

TRƯỜNG ĐẠI HỌC KIẾN TRÚC ĐÀ NẴNG

KHOA CÔNG NGHỆ THÔNG TIN

-----*****-----



ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

ĐỀ TÀI:

**XÂY DỰNG HỆ THỐNG IOT GIÁM SÁT VÀ
ĐIỀU KHIỂN MÔI TRƯỜNG CHUỒNG TRẠI**

Giảng viên hướng dẫn: **TS. LÊ VĂN MINH**

Sinh viên thực hiện: **NGUYỄN PHƯƠNG LY**

Mã số sinh viên: **2151220214**

Lớp: **21CT6**

Khóa: **2021 – 2026**

Đà Nẵng, 2025

TRƯỜNG ĐẠI HỌC KIẾN TRÚC ĐÀ NẴNG

KHOA CÔNG NGHỆ THÔNG TIN

-----*****-----



ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

ĐỀ TÀI:

**XÂY DỰNG HỆ THỐNG IOT GIÁM SÁT VÀ
ĐIỀU KHIỂN MÔI TRƯỜNG CHUỒNG TRẠI**

Giảng viên hướng dẫn: **TS. LÊ VĂN MINH**

Sinh viên thực hiện: **NGUYỄN PHƯƠNG LY**

Mã số sinh viên: **2151220214**

Lớp: **21CT6**

Khóa: **2021 – 2026**

Đà Nẵng, 2025

BẢN NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN
ĐỒ ÁN TỐT NGHIỆP

(GVHD nộp Bản nhận xét này về Văn phòng Khoa)

1. **Họ và tên sinh viên/ nhóm sinh viên được giao đề tài** (sĩ số trong nhóm...01...):
Họ và tên: Nguyễn Phương Ly..... MSSV: 2151220214...Lớp: 21CT6.....
Ngành : Công nghệ thông tin.....
Chuyên ngành : Công nghệ thông tin.....
2. **Tên đề tài:** Xây dựng hệ thống IoT giám sát và điều khiển môi trường chuồng trại..
.....
3. **Tổng quát về ĐA/KLTN:**
Số trang: Số chương:
Số bảng số liệu: Số hình vẽ:
Số tài liệu tham khảo: Phần mềm tính toán:
Số bản vẽ kèm theo: Hình thức bản vẽ:
Hiện vật (sản phẩm) kèm theo:
4. **Nhận xét:**
 - a) *Về tinh thần, thái độ làm việc của sinh viên:*
.....
.....
 - b) *Những kết quả đạt được của ĐA/KLTN:*
.....
.....
.....
.....
 - c) *Những hạn chế của ĐA/KLTN:*
.....
.....
.....
5. **Đề nghị:**
Được bảo vệ (hoặc nộp ĐA để chấm) ☐ Không được bảo vệ ☐
Đà Nẵng, ngày ... tháng ... năm
Giảng viên hướng dẫn
(Ký và ghi rõ họ tên)

TRƯỜNG ĐẠI HỌC KIẾN TRÚC ĐÀ NẴNG CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
KHOA CÔNG NGHỆ THÔNG TIN Độc lập – Tự do – Hạnh phúc
BẢN NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN/ ĐÁNH GIÁ
ĐỒ ÁN/ KHÓA LUẬN TỐT NGHIỆP
(GVPPB nộp Bản nhận xét này về Văn phòng Khoa)

6. **Họ và tên sinh viên/ nhóm sinh viên được giao đề tài** (sĩ số trong nhóm...01...):
Họ và tên: Nguyễn Phương Ly..... MSSV: 2151220214...Lớp: 21CT6.....

7. **Tên đề tài:** Xây dựng hệ thống IoT giám sát và điều khiển môi trường chuồng trại ...
.....

8. **Nhận xét:**

a) Những kết quả đạt được của ĐA/KLTN:

.....
.....
.....
.....
.....
.....

b) Những hạn chế của ĐA/KLTN:

.....
.....
.....
.....
.....
.....

9. **Đề nghị:**

Được bảo vệ ☐

Bổ sung thêm đề bảo vệ ☐

Không được bảo vệ ☐

10. **Các câu hỏi sinh viên cần trả lời trước Hội đồng:**

(1)
.....
(2)
.....
(3)
.....

Đà Nẵng, ngày ... tháng ... năm

Giảng viên phản biện

(Ký và ghi rõ họ tên)

TÓM TẮT

Tên đề tài: Xây dựng hệ thống IoT giám sát và điều khiển môi trường chuồng trại

Sinh viên thực hiện: Nguyễn Phương Ly

MSSV: 2151220214 - Lớp: 21CT6

Đề tài tập trung xây dựng hệ thống IoT có khả năng giám sát và điều khiển môi trường chuồng trại theo thời gian thực, giúp người chăn nuôi dễ dàng theo dõi và điều chỉnh các thông số môi trường nhằm đảm bảo điều kiện sinh trưởng ổn định cho vật nuôi. Hệ thống sử dụng vi điều khiển ESP32 làm bộ xử lý trung tâm, kết hợp với các cảm biến đo nhiệt độ, độ ẩm và độ rung để thu thập dữ liệu môi trường. Các thông số được hiển thị trực tiếp trên màn hình OLED và truyền lên ứng dụng Blynk thông qua kết nối Wifi, cho phép theo dõi từ xa trên điện thoại.

Hệ thống hỗ trợ hai chế độ điều khiển: thủ công và tự động. Ở chế độ tự động, quạt, đèn chiếu sáng và đèn sưởi sẽ được bật/tắt dựa trên các giá trị ngưỡng cài đặt từ ứng dụng Blynk. Ở chế độ thủ công, người dùng có thể điều khiển các thiết bị thông qua ứng dụng Blynk hoặc Remote hồng ngoại. Kết quả mong muốn là mô hình hoạt động ổn định, dễ sử dụng và có khả năng áp dụng tại các chuồng trại, góp phần giảm sức lao động và nâng cao hiệu quả chăm sóc vật nuôi.

LỜI NÓI ĐẦU

Để hoàn thành được đồ án tốt nghiệp này, em xin bày tỏ lòng biết ơn sâu sắc đến quý thầy cô trong Khoa Công Nghệ Thông Tin – Trường Đại học Kiến Trúc Đà Nẵng đã tận tình giảng dạy, truyền đạt cho em những kiến thức quý báu trong suốt quá trình học tập. Những kiến thức đó không chỉ là nền tảng để em hoàn thành tốt đồ án tốt nghiệp, mà còn là hành trang quan trọng giúp em tự tin bước vào con đường nghề nghiệp sau này.

Đặc biệt, em xin gửi lời cảm ơn chân thành và sự tri ân sâu sắc đến thầy **TS. LÊ VĂN MINH**, người đã trực tiếp hướng dẫn, tận tình chỉ bảo, định hướng và giúp đỡ em trong suốt quá trình thực hiện đồ án. Sự hỗ trợ và động viên của thầy là nguồn động lực to lớn giúp em vượt qua khó khăn và hoàn thành đồ án này.

Trong quá trình thực hiện đề tài, do kiến thức và kinh nghiệm thực tiễn còn hạn chế nên bài làm khó tránh khỏi những thiếu sót. Em rất mong nhận được những ý kiến đóng góp quý báu của quý thầy cô để em có thể hoàn thiện hơn và tích lũy thêm nhiều kinh nghiệm cho công việc sau này.

Em xin chân thành cảm ơn!

CAM ĐOAN

Em xin cam đoan rằng toàn bộ nội dung và kết quả đạt được trong đồ án tốt nghiệp là sản phẩm nghiên cứu của cá nhân em, được thực hiện dưới sự hướng dẫn tận tình của TS. Lê Văn Minh.

Đề tài không sao chép hay sử dụng kết quả của người khác dưới bất kỳ hình thức nào. Trong toàn bộ nội dung của đồ án, các phần trích dẫn, tham khảo đều được ghi rõ nguồn gốc và tuân thủ đúng quy định về trích dẫn tài liệu.

Em xin hoàn toàn chịu trách nhiệm trước nhà trường về tính trung thực và cam kết của mình đối với đồ án tốt nghiệp này.

Sinh viên thực hiện

MỤC LỤC

TÓM TẮT	
LỜI NÓI ĐẦU	i
CAM ĐOAN.....	ii
DANH SÁCH HÌNH ẢNH.....	vi
DANH SÁCH CÁC CỤM TỪ VIẾT TẮT	ix
MỞ ĐẦU	x
Chương 1: CƠ SỞ LÝ THUYẾT VÀ CÔNG NGHỆ SỬ DỤNG	1
1.1. Tổng quan về Internet of Thing (IoT).....	1
1.1.1. Khái niệm về IoT	1
1.1.2. Nguyên lý hoạt động.....	3
1.1.3. Ứng dụng của IoT trong đời sống.....	4
1.2. Tổng quan về ứng dụng Blynk.....	7
1.2.1. Giới thiệu về Blynk.....	7
1.2.2. Cách hoạt động của Blynk.....	9
1.2.3. Ứng dụng của Blynk trong hệ thống IoT.....	12
1.3. Ngôn ngữ lập trình Arduino	13
1.3.1. Khái niệm về ngôn ngữ lập trình Arduino	13
1.3.2. Ứng dụng của ngôn ngữ lập trình Arduino trong IoT	15
1.3.3. Các thư viện sử dụng trong hệ thống.....	15
1.4. Môi trường phát triển Arduino IDE	16
1.4.1. Giới thiệu Arduino IDE	16
1.4.2. Các chức năng chính của Arduino IDE	17
1.4.3. Ưu điểm và nhược điểm của Arduino IDE trong lập trình IoT	20
1.5. Giới thiệu về các thiết bị phần cứng.....	21
1.5.1. Thiết bị điều khiển	21

1.5.2. Thiết bị cảm biến	23
1.5.3. Thiết bị chấp hành	25
1.5.4. Thiết bị hiển thị.....	26
Chương 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG	28
2.1. Phân tích yêu cầu chức năng.....	28
2.1.1. Thu thập dữ liệu từ các cảm biến.....	28
2.1.2. Nhận tín hiệu điều khiển từ Remote hồng ngoại	28
2.1.3. Điều khiển thủ công qua ứng dụng Blynk	28
2.1.4. Thiết lập cảnh báo khi giá trị vượt ngưỡng	29
2.1.5. Hẹn giờ bật – tắt đèn chiếu sáng.....	29
2.1.6. Hiển thị dữ liệu lên ứng dụng Blynk và OLED	29
2.1.7. Gửi thông báo cảnh báo qua Gmail khi vượt ngưỡng	29
2.2. Thiết kế hệ thống.....	30
2.2.1. Thiết kế phần cứng	30
2.2.1.1. Vi điều khiển ESP32	31
2.2.1.2. Cảm biến DHT22	34
2.2.1.3. Module Relay 5V – 10A	35
2.2.1.4. Quạt tản nhiệt – 12V	36
2.2.1.5. Đèn LED dây dán 12V (Đèn chiếu sáng).....	38
2.2.1.6. Đèn sợi đốt loại C35 (Đèn sưởi)	39
2.2.1.7. OLED SSD1306 0.96 inch	41
2.2.1.8. Remote hồng ngoại và Remote IR1838	42
2.2.2. Thiết kế phần mềm	44
2.2.2.1. Cài đặt môi trường Arduino IDE	44
2.2.2.2. Thiết lập môi trường ESP32 trong Arduino IDE	46
2.2.3. Thiết lập ứng dụng trên app Blynk	48
2.3. Lập trình hệ thống (Code).....	53
Chương 3: KẾT QUẢ THỰC NGHIỆM HỆ THỐNG.....	65
3.1. Mô hình thực tế	65

3.2. Kết quả thực nghiệm.....	66
3.2.1. Kết quả đo và hiển thị trên Blynk và màn hình OLED	66
3.2.2. Kết quả cảnh báo khi vượt ngưỡng qua ứng dụng Blynk và Gmail	67
3.2.3. Kết quả chức năng điều khiển các thiết bị qua ứng dụng Blynk	70
3.2.4. Kết quả hẹn giờ bật – tắt đèn chiếu sáng	72
KẾT LUẬN	73
TÀI LIỆU THAM KHẢO.....	75

DANH SÁCH HÌNH ẢNH

Hình 1.1. Internet of Things	1
Hình 1.2. Nguyên lý hoạt động cơ bản của IoT.	3
Hình 1.3. Công nghệ nhà thông minh IoT.....	5
Hình 1.4. Theo dõi lộ trình đi của xe chở hàng.	5
Hình 1.5. Theo dõi tình trạng sinh trưởng của cây trồng.	6
Hình 1.6. Smart Watch – Thiết bị Internet of Bodies phổ biến nhất hiện nay.	6
Hình 1.7. Logo Blynk.	7
Hình 1.8 Blynk Console.	8
Hình 1.9. Blynk Apps	8
Hình 1.10. Blynk Library / Edgent.....	9
Hình 1.11. Sơ đồ luồng thực thi của chương trình Arduino.....	14
Hình 1.12. Logo Arduino IDE.....	17
Hình 1.13. Trình soạn thảo mã nguồn trong Arduino IDE.....	18
Hình 1.14. Chức năng Verify - Upload trong Arduino IDE.....	18
Hình 1.15. Giao diện Boards Manager.....	19
Hình 1.16. Giao diện Serial Plotter.	19
Hình 1.17. Giao diện Library Manager.	20
Hình 1.18. ESP8266 NodeMCU.	21
Hình 1.19. ESP8266 Wemos D1 Mini.	22
Hình 1.20. Arduino UNO.	22
Hình 1.21. Arduino Nano.	23
Hình 1.22. Cảm biến ánh sáng.....	24
Hình 1.23. Cảm biến siêu âm.	24
Hình 1.24. Servo Motor.....	25
Hình 1.25. Máy bơm Mini.....	26
Hình 1.26. LCD 16x2.	27

Hình 1.27. LED 7 đoạn.....	27
Hình 2.1. Sơ đồ thiết kế phần cứng.	30
Hình 2.2. Vi điều khiển ESP32.	32
Hình 2.3. Cảm biến DHT22.	34
Hình 2.4. Sơ đồ khối của cảm biến DHT22 với ESP32.	35
Hình 2.5. Module Relay 5V - 10A.	35
Hình 2.6. Sơ đồ khối của Module Relay 5V - 10A với ESP32.	36
Hình 2.7. Quạt tản nhiệt 12V.....	36
Hình 2.8. Sơ đồ khối điều khiển quạt tản nhiệt với ESP32 thông qua Module Relay ..	37
Hình 2.9. Đèn LED dây 12V.	38
Hình 2.10. Sơ đồ khối điều khiển Đèn LED dây 12V với ESP32 thông qua Module Relay 5V – 10A.	39
Hình 2.11. Đèn sợi đốt C35 - 40W.....	39
Hình 2.12. Sơ đồ khối điều khiển Đèn sợi đốt C35 - 40W với ESP32 thông qua Module Relay 5V – 10A.	40
Hình 2.13. OLED SSD1306 0.96 inch.	41
Hình 2.14. Sơ đồ khối nối OLED SSD1306 0.96 inch với ESP32.	42
Hình 2.15. Remote hồng ngoại và Remote IR1838.	42
Hình 2.16. Sơ đồ nối Remote IR1838 với ESP32.....	43
Hình 2.17. Tải ứng dụng Arduino IDE.....	44
Hình 2.18. Giao diện đồng ý tải Arduino IDE.	44
Hình 2.19. Chọn thành phần tải.....	45
Hình 2.20. Chọn nơi tải về.	45
Hình 2.21. Đóng ứng dụng khi tải xong.....	46
Hình 2.22. Trình soạn thảo Arduino IDE.....	46
Hình 2.23. Cài đặt ESP32 Broad.	47
Hình 2.24. Chọn Board ESP32.....	48
Hình 2.25. Chọn cổng COM.....	48
Hình 2.26. Tạo Template cho giao diện Blynk.	49

Hình 2.27. Tạo Device cho giao diện Blynk.	49
Hình 2.28. Thiết lập Datastream cho giao diện Blynk.	50
Hình 2.29. Tạo event cho hệ thống Blynk khi nhiệt độ thấp hơn ngưỡng.	50
Hình 2.30. Tạo event cho hệ thống Blynk khi nhiệt độ cao hơn ngưỡng.....	51
Hình 2.31. Thiết lập thông báo gửi về Gmail và ứng dụng Blynk.	51
Hình 2.32. Cài đặt thời gian gửi của hệ thống trên Blynk.....	52
Hình 2.33. Giao diện hệ thống trên App Blynk.....	52
Hình 3.1. Hệ thống khi chưa hoạt động.....	65
Hình 3.2. Hệ thống khi đang hoạt động.....	65
Hình 3.3. Kết quả đo và hiển thị trên Blynk.....	66
Hình 3.4. Kết quả đo và hiển thị trên màn hình OLED.....	67
Hình 3.5. Kết quả cảnh báo khi nhiệt độ thấp hơn ngưỡng nhiệt độ tối thiểu qua Blynk	68
Hình 3.6. Kết quả cảnh báo khi nhiệt độ thấp hơn ngưỡng nhiệt độ tối thiểu qua Gmail	68
Hình 3.7. Kết quả cảnh báo khi nhiệt độ cao hơn ngưỡng nhiệt độ tối đa qua Blynk. .	69
Hình 3.8. Kết quả cảnh báo khi nhiệt độ cao hơn ngưỡng nhiệt độ tối đa qua Gmail ..	69
Hình 3.9. Giao diện hệ thống sau khi điều chỉnh ngưỡng nhiệt độ.	70
Hình 3.10. Kết quả nhiệt độ cao hơn ngưỡng nhiệt độ tối đa - Quạt bật.	70
Hình 3.11. Kết quả nhiệt độ cao hơn ngưỡng nhiệt độ tối thiểu - Đèn sưởi bật.	71
Hình 3.12. Giao diện hệ thống khi ở chế độ thủ công.....	71
Hình 3.13. Thiết lập thời gian hẹn giờ bật - tắt đèn chiếu sáng.	72
Hình 3.14. Kết quả hẹn giờ của đèn chiếu sáng.	72

DANH SÁCH CÁC CỤM TỪ VIẾT TẮT

STT	Ký hiệu chữ viết tắt	Chữ viết tiếng anh	Chữ viết tiếng việt
1	IoT	Internet of Things	Internet vạn vật
2	RFID	Radio Frequency Identification	Nhận dạng bằng tần số vô tuyến
3	GPS	Global Positioning System	Hệ thống định vị toàn cầu
4	SDA	Serial Data (Line)	Đường truyền dữ liệu nối tiếp
5	SCL	Serial Clock (Line)	Đường xung nhịp nối tiếp
6	OLED	Organic Light Emitting Diode	Đi-ốt phát quang hữu cơ
7	I2C	Inter – Integrated Circuit	Giao thức truyền thông I2C
8	IDE	Integrated Development Environment	Môi trường phát triển tích hợp
9	PCB	Printed Circuit Board	Bảng mạch in
10	GPIO	General Purpose Input/Output	Chân vào/ra đa năng
11	LDR	Light Dependent Resistor	Điện trở phụ thuộc ánh sáng
12	DC	Direct Current	Dòng điện một chiều
13	AC	Alternating Current	Dòng điện xoay chiều
14	LCD	Liquid Crystal Display	Màn hình tinh thể lỏng
15	UART	Universal Asynchronous Receiver/Transmitter	Truyền thông nối tiếp không đồng bộ
16	SPI	Serial Peripheral Interface	Giao thức ngoại vi nối tiếp
17	AI	Artificial Interlligence	Trí tuệ nhân tạo

MỞ ĐẦU

1. Lý do chọn đề tài

Trong chăn nuôi hiện nay, việc đảm bảo môi trường chăn nuôi ổn định là yếu tố quan trọng ảnh hưởng trực tiếp đến sức khỏe, khả năng sinh trưởng và năng suất của vật nuôi. Tuy nhiên, ở nhiều mô hình chăn nuôi hộ gia đình và trang trại nhỏ, công tác giám sát và điều chỉnh môi trường vẫn chủ yếu dựa vào kinh nghiệm và quan sát thủ công của người chăn nuôi. Điều này dễ dẫn đến tình trạng nhiệt độ và độ ẩm không phù hợp, ánh sáng không đảm bảo, thông gió không hợp lý, khiến vật nuôi dễ bị sốc nhiệt, giảm sức đề kháng và làm tăng nguy cơ dịch bệnh, ảnh hưởng trực tiếp đến hiệu quả kinh tế.

Cùng với sự phát triển của cuộc cách mạng công nghiệp 4.0, công nghệ Internet of Things (IoT) cho phép các thiết bị cảm biến và điều khiển kết nối Internet, thu thập dữ liệu và truyền tải thông tin theo thời gian thực. Nhờ đó, người chăn nuôi có thể theo dõi các thông số môi trường chuồng trại như nhiệt độ, độ ẩm thông qua ứng dụng trên điện thoại, đồng thời điều khiển các thiết bị như quạt, đèn chiếu sáng, đèn sưởi từ xa một cách nhanh chóng và chính xác mà không cần trực tiếp có mặt tại chuồng trại.

Xuất phát từ những yêu cầu thực tế đó, đề tài “Xây dựng hệ thống IoT giám sát và điều khiển môi trường chuồng trại” được lựa chọn nhằm thiết kế và triển khai một hệ thống nhỏ gọn, dễ sử dụng, chi phí hợp lý và hoạt động hiệu quả. Hệ thống sử dụng vi điều khiển ESP32 để thu thập dữ liệu từ các cảm biến và truyền lên ứng dụng Blynk thông qua Wifi, cho phép người dùng theo dõi trạng thái môi trường và điều khiển tự động và tự động các thiết bị và nhận thông báo khi giá trị vượt ngưỡng từ điện thoại .

Đề tài giúp nâng cao hiệu quả quản lý chuồng trại, giảm sức lao động và kịp thời theo dõi môi trường, góp phần bảo vệ sức khỏe vật nuôi. Đồng thời, đề tài giúp sinh viên ứng dụng công nghệ IoT vào thực tiễn, hướng đến mô hình chăn nuôi tự động và hiện đại.

2. Mục tiêu đề tài

Mục tiêu chính của đề tài là xây dựng một hệ thống IoT có khả năng giám sát và điều khiển môi trường chuồng trại theo thời gian thực, giúp người chăn nuôi dễ dàng theo dõi các thông số môi trường và điều chỉnh thiết bị phù hợp nhằm đảm bảo điều kiện sinh trưởng ổn định cho vật nuôi.

Cụ thể, đề tài hướng đến các mục tiêu sau:

- Thiết kế và xây dựng hệ thống phần cứng sử dụng vi điều khiển ESP32 cùng các cảm biến đo nhiệt độ, độ ẩm đảm bảo thu nhập dữ liệu chính xác và hoạt động ổn định.
- Lập trình xử lý và truyền dữ liệu từ ESP32 lên nền tảng Blynk thông qua kết nối Wifi, đảm bảo các thông số được cập nhật liên tục theo thời gian thực.
- Xây dựng giao diện giám sát trên ứng dụng Blynk, cho phép người dùng quan sát các giá trị môi trường trực quan và chọn chế độ hoạt động cũng như điều khiển các thiết bị trên điện thoại.
- Thiết kế hệ thống điều khiển hai chế độ:
 - Chế độ tự động: Hệ thống tự động bật – tắt quạt, đèn sưởi dựa trên các giá trị ngưỡng được thiết lập trên điện thoại và bật tắt đèn chiếu sáng dựa trên hệ giờ từ điện thoại
 - Chế độ thủ công: Người dùng có thể điều khiển các thiết bị trực tiếp trên ứng dụng điện thoại hoặc điều khiển thông qua remote hồng ngoại.
- Thiết lập cảnh báo khi thông số môi trường vượt ngưỡng, giúp người chăn nuôi kịp thời xử lý để hạn chế ảnh hưởng đến sức khỏe vật nuôi.
- Đánh giá và kiểm thử hệ thống để rút ra nhận xét, ưu điểm, hạn chế và đề xuất hướng hoàn thiện trong tương lai.

Thông qua đề tài này, sinh viên chúng em có cơ hội vận dụng kiến thức về vi điều khiển, lập trình IoT, cảm biến và truyền dữ liệu không dây, đồng thời rèn luyện tư duy thiết kế hệ thống, tạo ra sản phẩm có tính ứng dụng thực tế cao trong lĩnh vực chăn nuôi.

3. Đối tượng và phạm vi nghiên cứu

a. Đối tượng nghiên cứu

Đề tài tập trung nghiên cứu và xây dựng hệ thống IoT giám sát và điều khiển môi trường chuồng trại dựa trên vi điều khiển ESP32, kết hợp với ứng dụng Blynk và điều khiển trực tiếp bằng Remote hồng ngoại.

Các đối tượng nghiên cứu chính bao gồm:

- Cảm biến nhiệt độ - độ ẩm (DHT22): dùng để đo và theo dõi nhiệt độ và độ ẩm không khí trong chuồng trại.
- Màn hình OLED (SSD1306): hiển thị trực tiếp các giá trị cảm biến và trạng thái hệ thống (thủ công - tự động) và tình trạng kết nối ứng dụng Blynk, hỗ trợ theo dõi tại chỗ.
- Vi điều khiển ESP32: thu thập dữ liệu cảm biến, xử lý và truyền dữ liệu qua Wifi đến ứng dụng Blynk.

- Module Relay: điều khiển hoạt động của quạt , đèn chiếu sáng và đèn sưởi
- Ứng dụng Blynk: giao diện để theo dõi và điều khiển từ xa theo thời gian thực
- Remote hồng ngoại: cho phép điều khiển thiết bị trực tiếp tại chuồng trại mà không cần dùng đến điện thoại.

b. Phạm vi nghiên cứu

Hệ thống giám sát các thông số môi trường (nhiệt độ, độ ẩm và độ rung) và điều khiển quạt, đèn chiếu sáng và đèn sưởi với hai chế độ hoạt động:

- Chế độ tự động: thiết bị được bật – tắt dựa trên ngưỡng môi trường được cài đặt trên điện thoại.
- Chế độ thủ công: người dùng có thể điều khiển các thiết bị thông qua ứng dụng Blynk hoặc Remote hồng ngoại.

Dữ liệu được truyền lên Blynk qua kết nối Wifi, không triển khai máy chủ riêng hoặc lưu trữ dữ liệu phân tích chuyên sâu.

Màn hình OLED hiển thị dữ liệu và trạng thái hệ thống tại chỗ, hỗ trợ quan sát khi không có Internet.

Mô hình thực hiện ở quy mô thử nghiệm, nhưng có thể mở rộng áp dụng thực tế cho hộ chăn nuôi hoặc chuồng trại.

Nội dung đề tài tập trung vào độ chính xác và ổn định của cảm biến, khả năng điều khiển thiết bị linh hoạt (tự động - thủ công) và giao diện hiển thị rõ ràng trên OLED và ứng dụng Blynk.

4. Nội dung báo cáo

Chương 1: Cơ sở lý thuyết và công nghệ sử dụng

Chương này trình bày các kiến thức nền tảng phục vụ cho việc xây dựng hệ thống IoT giám sát và điều khiển môi trường chuồng trại. Nội dung bao gồm tổng quan về IoT, nguyên lý hoạt động và các ứng dụng trong giám sát môi trường; giới thiệu nền tảng Blynk dùng để quan sát dữ liệu từ xa; ngôn ngữ lập trình Arduino cùng các thư viện hỗ trợ; phần mềm Arduino IDE; và các thiết bị phần cứng như ESP32, cảm biến DHT22, màn hình OLED, Remote IR1838, Module Relay, Quạt tản nhiệt, đèn LED, đèn sưởi. Chương giúp người đọc nắm được toàn bộ công nghệ và linh kiện dùng trong hệ thống.

Chương 2: Phân tích và thiết kế hệ thống

Chương này tập trung vào việc phân tích yêu cầu, mô hình hóa và thiết kế hệ thống giám sát. Nội dung bao gồm xác định chức năng cần có của hệ thống (thu thập dữ liệu, hiển thị, cảnh báo, điều khiển thiết bị); thiết kế phần cứng với các cảm biến và kết nối ESP32; thiết kế phần mềm gồm xử lý tín hiệu cảm biến, truyền dữ liệu lên Blynk và xây dựng giao diện hiển thị. Chương cũng trình bày mã lập trình chính để vận hành hệ thống. Đây là phần then chốt hình thành cấu trúc và nguyên lý hoạt động của mô hình.

Chương 3: Thi công và thực nghiệm hệ thống

Chương này mô tả quá trình lắp đặt mô hình thực tế và tiến hành kiểm thử hệ thống. Các kết quả đo lường từ cảm biến được so sánh với dữ liệu hiển thị trên Blynk và OLED, đánh giá khả năng đưa ra cảnh báo khi giá trị vượt ngưỡng và kiểm tra tính ổn định của hệ thống trong môi trường hoạt động. Cuối chương đưa ra kết luận chung, chỉ ra những hạn chế còn tồn tại và đề xuất hướng phát triển trong tương lai nhằm nâng cao độ chính xác và tính ứng dụng của hệ thống.

Chương 1: CƠ SỞ LÝ THUYẾT VÀ CÔNG NGHỆ SỬ DỤNG

1.1. Tổng quan về Internet of Thing (IoT)

1.1.1. Khái niệm về IoT

IoT (Internet of Things hay Internet vạn vật) là khái niệm mô tả mạng lưới các thiết bị vật lý, phương tiện, thiết bị gia dụng và các mục khác được kết nối với Internet và có khả năng thu thập, chia sẻ dữ liệu với nhau. Các thiết bị này thường được trang bị cảm biến, phần mềm, và các công nghệ khác để giao tiếp và trao đổi dữ liệu một cách tự động.



Hình 1.1. Internet of Things

Điểm nổi bật của IoT là tính kết nối và khả năng tương tác giữa các thiết bị, cho phép chúng tự động hoạt động, đưa ra quyết định hoặc kích hoạt hành động dựa trên dữ liệu thu thập được. IoT có ứng dụng rộng rãi trong nhiều lĩnh vực như nhà thông minh, y tế, nông nghiệp, sản xuất, và quản lý năng lượng.

Quá trình phát triển của IoT

IoT có quy trình phát triển tương đối dài để đạt đến mức độ phủ sóng mạnh mẽ như hiện nay.

Một số mốc thời gian quan trọng có thể kể đến như sau:

- **Năm 1990:** Máy nướng bánh mì gây tò mò khi lần đầu tiên được kết nối Internet. Một kỹ sư phần mềm tại Mỹ tên là John Romkey đã kết nối máy nướng bánh mì với mạng máy tính.
- **Năm 1999:** Kevin Ashton khi thuyết trình về một hệ thống cảm biến và nhận dạng qua tần số radio (RFID) gắn trên hàng hóa để quản lý chuỗi cung ứng đã lần đầu đưa ra thuật ngữ “Internet of Things” – khởi đầu cho khái niệm Internet vạn vật.
- **Năm 2000:** Công ty LG giới thiệu sản phẩm tủ lạnh có kết nối Internet đầu tiên trên thế giới được bán với mức giá 20.000 USD. Cho phép người tiêu dùng mua sắm thực phẩm trực tuyến và thực hiện cuộc gọi video.
- **Năm 2008:** Hội nghị quốc tế đầu tiên về chủ đề Internet vạn vật được tổ chức tại Zurich, Thụy Sĩ.
- **Năm 2009:** CISCO tuyên bố đây là thời điểm mà Internet vạn vật thực sự được khai sinh, khi số lượng thiết bị được kết nối Internet thậm chí vượt cả dân số thế giới.
- **Năm 2013:** Từ điển Oxford chính bổ sung định nghĩa “ Internet of Things” vào hệ thống.
- **Năm 2020:** Số lượng thiết bị được kết nối Internet trên thế giới ước tính đã vượt qua con số 20 tỷ.
- **Năm 2025:** Dự báo thế giới sẽ có 75 tỷ thiết bị sử dụng IoT.

Các đặc điểm chính của IoT

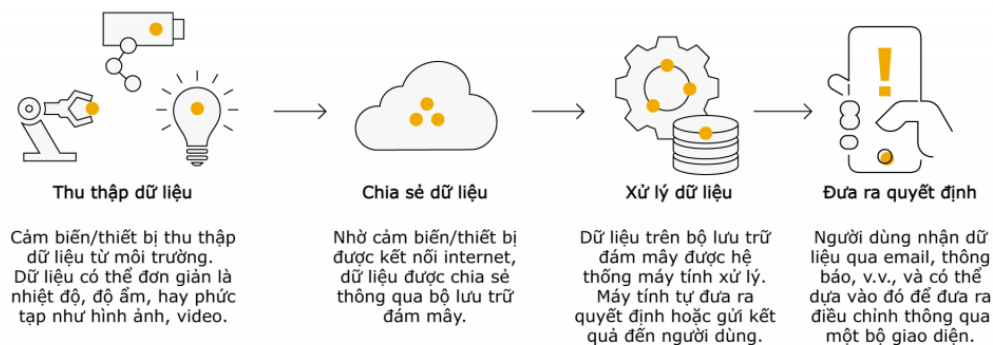
IoT (Internet of Things) có một số đặc điểm chính làm nổi bật công nghệ này và tác động đến cách nó được triển khai và sử dụng:

- **Kết nối liên tục:** Các thiết bị IoT luôn kết nối với nhau và với Internet, cho phép giao tiếp và trao đổi dữ liệu theo thời gian thực.
- **Tính tự động:** IoT cho phép các thiết bị thực hiện các hành động và đưa ra quyết định tự động dựa trên dữ liệu thu thập được mà không cần can thiệp của con người.
- **Tính tương tác:** Các thiết bị IoT có khả năng giao tiếp và tương tác với các thiết bị khác, tạo ra một hệ thống liên kết chặt chẽ và liên tục.

- **Đa dạng thiết bị:** IoT bao gồm nhiều loại thiết bị khác nhau, từ các thiết bị thông minh trong nhà đến các hệ thống công nghiệp, tất cả đều có khả năng kết nối và chia sẻ dữ liệu.
- **Tính mở rộng:** IoT có khả năng mở rộng dễ dàng, cho phép thêm các thiết bị và hệ thống mới mà không ảnh hưởng đến toàn bộ mạng lưới.
- **Thu thập và phân tích dữ liệu:** Các thiết bị IoT thường được trang bị cảm biến và phần mềm cho phép thu thập và phân tích dữ liệu, giúp đưa ra các quyết định dựa trên dữ liệu đó.
- **Tính an toàn và bảo mật:** Với sự kết nối liên tục và lượng dữ liệu lớn, IoT đòi hỏi các biện pháp an toàn và bảo mật mạnh mẽ để bảo vệ dữ liệu và người dùng.
- **Ứng dụng đa dạng:** IoT có thể được ứng dụng trong nhiều lĩnh vực khác nhau như nhà thông minh, y tế, sản xuất, nông nghiệp, giao thông, và năng lượng, giúp cải thiện hiệu suất và hiệu quả trong các lĩnh vực này.

1.1.2. Nguyên lý hoạt động

Nguyên lý hoạt động của IoT được xây dựng trên một chu trình gồm bốn bước: thu thập dữ liệu, truyền tải, phân tích xử lý, và đưa ra quyết định. Sự phối hợp chặt chẽ giữa các bước này giúp hệ thống vận hành thông minh, tự động và thích ứng hiệu quả với môi trường thực tế.



Hình 1.2. Nguyên lý hoạt động cơ bản của IoT.

Thu thập dữ liệu:

Ở giai đoạn đầu tiên, các cảm biến hoặc thiết bị đầu cuối (things) đảm nhiệm vai trò thu thập thông tin từ môi trường. Dữ liệu có thể là các thông số đơn giản như nhiệt độ, độ ẩm, ánh sáng, hoặc phức tạp hơn như hình ảnh, âm thanh và chuyển động. Đây là nguồn dữ liệu đầu vào quan trọng, giúp hệ thống nắm bắt được tình trạng thực tế và phục vụ cho các bước xử lý tiếp theo. Chất lượng dữ liệu ở bước này quyết định độ chính xác của toàn bộ hệ thống.

Chia sẻ (truyền) dữ liệu:

Sau khi được thu thập, dữ liệu sẽ được truyền đến bộ xử lý trung gian (Gateway) hoặc trực tiếp lên nền tảng đám mây thông qua giao thức kết nối như Wifi, Bluetooth, ZigBee, LoRa hoặc mạng di động 4G/5G. Giai đoạn này đảm bảo rằng dữ liệu được chuyển giao một cách liên tục, ổn định và an toàn để hệ thống có thể xử lý kịp thời. Đây cũng là bước giúp thiết bị IoT kết nối và tương tác với môi trường mạng rộng lớn hơn.

Xử lý và phân tích dữ liệu

Tại máy chủ hoặc nền tảng đám mây, dữ liệu được tổng hợp và xử lý bằng các thuật toán phù hợp. Hệ thống có thể thực hiện các tác vụ như lọc nhiễu, phân tích xu hướng, so sánh với ngưỡng, phát hiện bất thường hoặc đưa ra dự đoán. Giai đoạn này đóng vai trò quan trọng, bởi kết quả phân tích sẽ tạo cơ sở cho các hành động điều khiển tiếp theo. Đây chính là “bộ não” của hệ thống IoT, nơi dữ liệu thô được biến thành thông tin có ý nghĩa.

Đưa ra quyết định và phản hồi

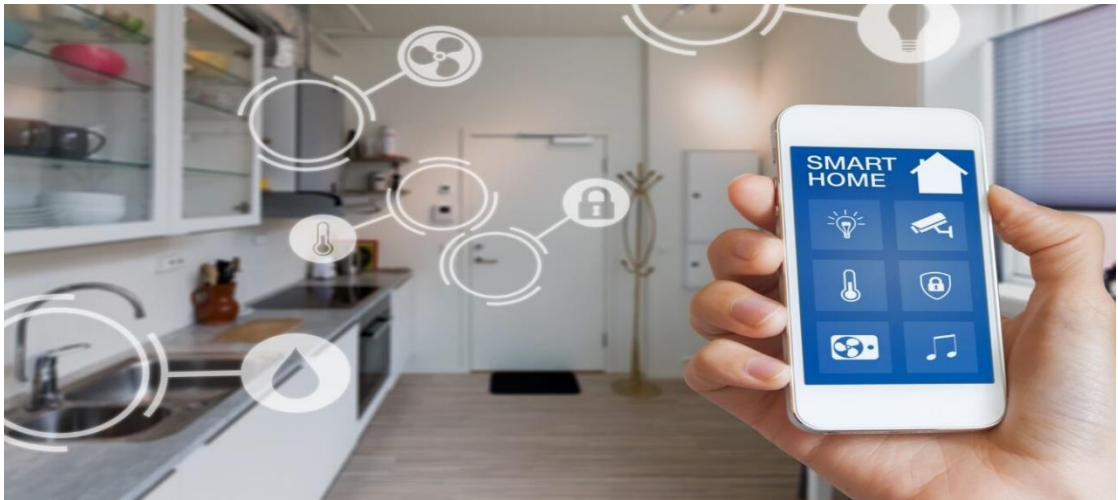
Dựa trên kết quả xử lý, hệ thống sẽ đưa ra các phản hồi tương ứng. Các hành động có thể bao gồm gửi cảnh báo đến người dùng thông qua ứng dụng hoặc email, hoặc tự động điều khiển thiết bị chấp hành như bật quạt, kích hoạt relay, điều chỉnh ánh sáng hay vận hành động cơ. Phản hồi có thể do hệ thống tự động quyết định, hoặc do con người dùng thao tác thông qua giao diện điều khiển. Giai đoạn này giúp hệ thống IoT duy trì khả năng hoạt động tự động, thông minh và kịp thời theo thời gian thực.

1.1.3. Ứng dụng của IoT trong đời sống

Trong những năm gần đây, Internet of Things (IoT) đã trở thành nền tảng quan trọng trong việc tự động hóa và hiện đại hóa nhiều lĩnh vực của đời sống và sản xuất. Thông qua việc sử dụng các cảm biến, kết nối mạng và hệ thống xử lý dữ liệu thông minh, IoT giúp nâng cao hiệu quả vận hành, giảm chi phí và hỗ trợ con người trong việc giám sát, điều khiển từ xa. Một số ứng dụng tiêu biểu của IoT có thể kể đến như sau:

Ứng dụng trong nhà thông minh (Smart Home)

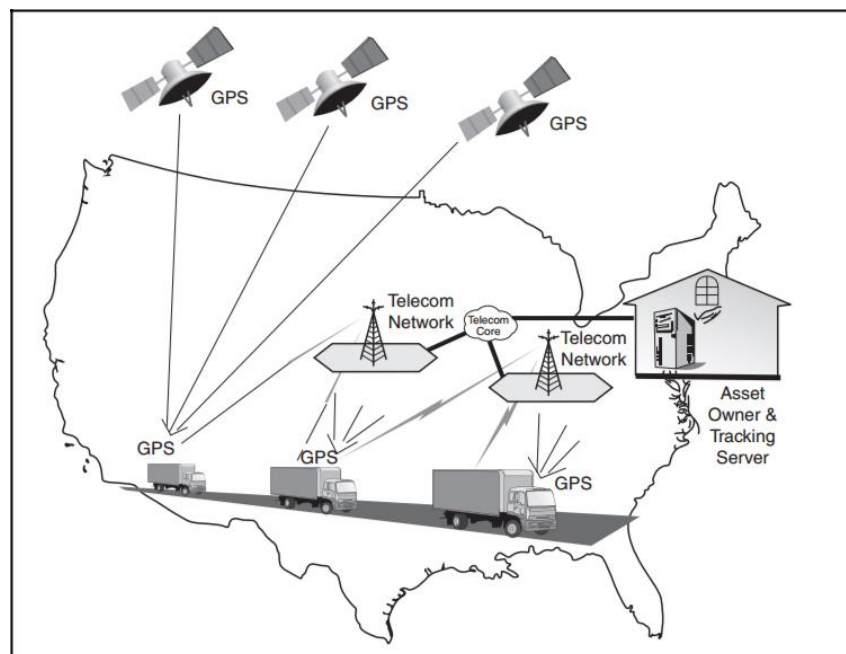
IoT cho phép kết nối, giám sát và tự động hóa các thiết bị trong gia đình như hệ thống chiếu sáng, rèm cửa, điều hòa, âm thanh và an ninh. Các thiết bị có thể hoạt động theo kịch bản được lập trình sẵn nhằm mang lại sự tiện nghi, tiết kiệm năng lượng và nâng cao mức độ an toàn cho người sử dụng. Ví dụ: Đèn chiếu sáng tự động nhờ cảm biến chuyển động, ...



Hình 1.3. Công nghệ nhà thông minh IoT.

Ứng dụng trong lĩnh vực vận tải

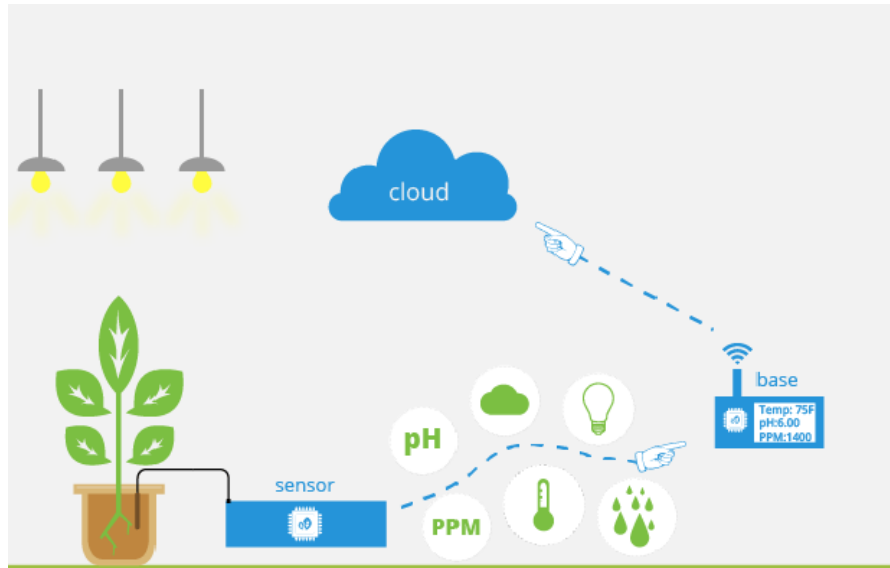
Trong lĩnh vực vận tải, IoT hỗ trợ giám sát phương tiện theo thời gian thực, giúp các doanh nghiệp tối ưu lộ trình và quản lý hiệu quả hoạt động vận chuyển. Dữ liệu thu thập từ các cảm biến và thiết bị GPS giúp kiểm soát tốc độ, thời gian di chuyển và mức tiêu hao nhiên liệu của từng phương tiện. Ví dụ: Xe tải gắn thiết bị GPS để theo dõi lộ trình và tốc độ, kho hàng sử dụng RFID để kiểm soát vị trí và số lượng hàng hóa, ...



Hình 1.4. Theo dõi lộ trình đi của xe chở hàng.

Ứng dụng trong lĩnh vực nông nghiệp

IoT góp phần nâng cao hiệu quả sản xuất nông nghiệp nhờ khả năng giám sát điều kiện môi trường và tự động điều chỉnh các hoạt động chăm sóc cây trồng. Các cảm biến đo độ ẩm, nhiệt độ, ánh sáng và độ pH giúp hệ thống đưa ra quyết định tưới tiêu, bón phân phù hợp với từng giai đoạn sinh trưởng. Ví dụ: Hệ thống tưới tự động, ...



Hình 1.5. Theo dõi tình trạng sinh trưởng của cây trồng.

Ứng dụng trong lĩnh vực y tế

IoT đóng vai trò quan trọng trong y tế nhờ khả năng theo dõi sức khỏe liên tục và hỗ trợ chẩn đoán từ xa. Các thiết bị đeo thông minh và cảm biến y tế thu thập dữ liệu sinh lý của bệnh nhân và gửi về hệ thống để bác sĩ giám sát và đưa ra quyết định kịp thời.

Ví dụ: Đồng hồ thông minh theo dõi nhịp tim, huyết áp, chất lượng giấc ngủ, ...



Hình 1.6. Smart Watch – Thiết bị Internet of Bodies phổ biến nhất hiện nay.

1.2. Tổng quan về ứng dụng Blynk

1.2.1. Giới thiệu về Blynk



Hình 1.7. Logo Blynk.

Blynk là gì?

Blynk là một nền tảng IoT (Internet of Things) giúp người dùng dễ dàng tạo ra các ứng dụng điều khiển từ xa cho các thiết bị thông minh. Điểm đặc biệt của Blynk chính là sự linh hoạt và dễ sử dụng.

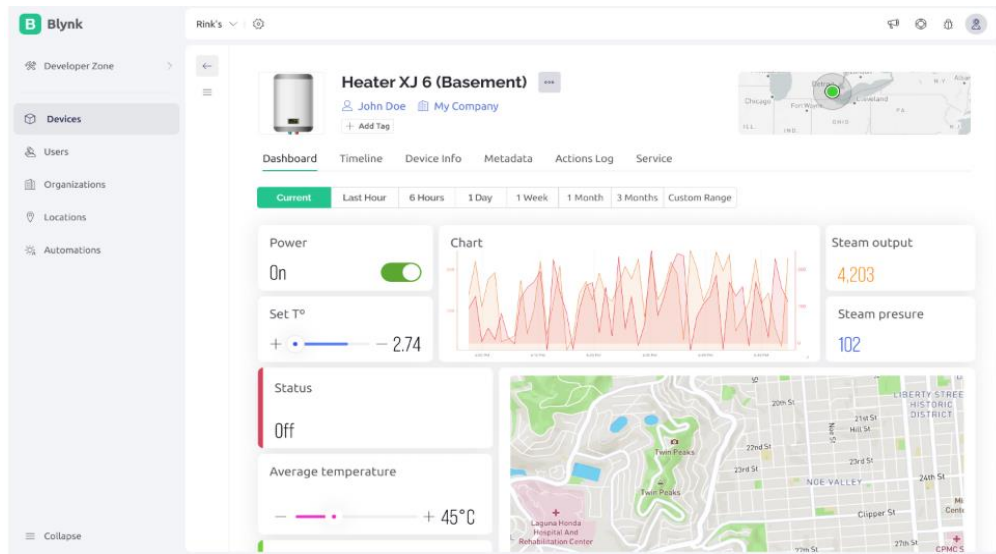
Với Blynk, người dùng có thể tạo ra các ứng dụng điều khiển thiết bị IoT chỉ trong vài phút mà không cần có kiến thức chuyên sâu về lập trình. Blynk cung cấp một loạt các công cụ và giao diện người dùng trực quan giúp người dùng dễ dàng kéo thả và tương tác với các thành phần điều khiển, biểu đồ, cảm biến, và nhiều tính năng khác.

Cụ thể, người dùng có thể tạo ra các ứng dụng để điều khiển đèn, quạt, cảm biến nhiệt độ, đo lường độ ẩm và thậm chí là máy rửa chén thông minh. Blynk hỗ trợ đa nền tảng, từ các thiết bị di động như điện thoại thông minh đến các thiết bị nhúng như Arduino, Raspberry Pi, ESP8266 và nhiều hơn nữa.

Điều này giúp Blynk trở thành một công cụ hữu ích không chỉ cho các nhà phát triển IoT chuyên nghiệp mà còn cho những người mới bắt đầu quan tâm đến thế giới IoT. Với sự tiện lợi và tính linh hoạt, Blynk đã và đang giúp mọi người dễ dàng thực hiện những ý tưởng sáng tạo của mình trong lĩnh vực IoT một cách hiệu quả.

Blynk cung cấp một hệ sinh thái đầy đủ gồm:

Blynk.Console



Hình 1.8 Blynk Console.

Blynk.Console là một ứng dụng website giàu tính năng, phục vụ cho nhiều loại người dùng khác nhau. Các chức năng chính của nó bao gồm:

- Cấu hình các thiết bị được kết nối trên nền tảng, bao gồm cài đặt ứng dụng.
- Quản lý thiết bị, dữ liệu, người dùng, tổ chức và vị trí.
- Giám sát và điều khiển thiết bị từ xa.

Blynk.Apps

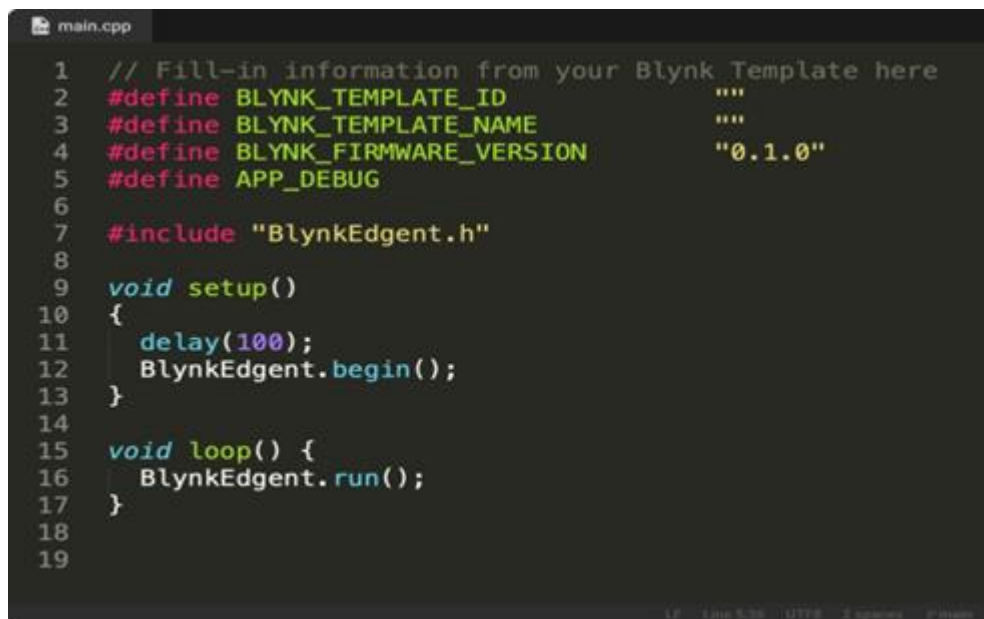


Hình 1.9. Blynk Apps

Blynk.Apps là ứng dụng di động iOS và Android đa năng có các chức năng chính sau:

- Giám sát và điều khiển từ xa các thiết bị được kết nối hoạt động với nền tảng Blynk.
- Cấu hình giao diện người dùng di động trong giai đoạn tạo mẫu và sản xuất.
- Tự động hóa hoạt động của thiết bị được kết nối.

Blynk.Edgent và thư viện Blynk



Hình 1.10. Blynk Library / Edgent.

Thư viện phần mềm dành cho vi điều khiển (như ESP8266, ESP32, Arduino...) giúp thiết bị kết nối dễ dàng với Blynk Cloud.

Nhờ đó, người dùng không cần tự xây dựng ứng dụng hoặc máy chủ phức tạp mà vẫn có thể tạo ra một hệ thống IoT hoàn chỉnh — chỉ cần lập trình phần cứng với thư viện Blynk và cấu hình giao diện điều khiển qua ứng dụng.

Blynk.Cloud

Blynk.Cloud là cơ sở hạ tầng máy chủ đóng vai trò là trung tâm của nền tảng IoT Blynk, liên kết tất cả các thành phần lại với nhau.

1.2.2. Cách hoạt động của Blynk

Blynk hoạt động dựa trên mô hình thiết bị (Device) – đám mây (Blynk Cloud) – ứng dụng (App/Console), trong đó toàn bộ dữ liệu và lệnh điều khiển được truyền qua lại

giữa các thành phần thông qua nền tảng Blynk Cloud. Toàn bộ quá trình có thể mô tả theo các bước sau:

Kết nối thiết bị với nền tảng Blynk:

Để kết nối thiết bị với nền tảng Blynk, trước hết cần chuẩn bị phần cứng như Arduino, ESP8266, ESP32, Raspberry Pi hoặc STM32, cùng môi trường lập trình tương ứng và thư viện Blynk. Khi tạo dự án mới trên Blynk.Console, hệ thống cấp một AuthToken duy nhất, đóng vai trò xác thực giúp thiết bị kết nối an toàn với Blynk Cloud.

Sau đó, người dùng cấu hình mạng cho thiết bị, phổ biến nhất là WiFi, ngoài ra có thể dùng Ethernet hoặc 3G/4G tùy phần cứng hỗ trợ. Trong chương trình, cần khai báo tên mạng và mật khẩu để thiết bị tự động kết nối Internet khi khởi động.

Thiết bị sẽ sử dụng AuthToken và các hàm của thư viện Blynk để xác thực và bắt đầu trao đổi dữ liệu với Cloud. Khi kết nối thành công, thiết bị có thể gửi dữ liệu cảm biến, nhận lệnh điều khiển và trạng thái sẽ hiển thị “online” trên Blynk.App hoặc Blynk.Console. Nếu gặp sự cố, nên kiểm tra lại thông tin mạng, AuthToken và kết nối Internet để hệ thống hoạt động ổn định.

Gửi và nhận dữ liệu qua Blynk Cloud:

Khi thiết bị đã kết nối thành công với Blynk Cloud, việc trao đổi dữ liệu giữa thiết bị và Cloud diễn ra theo cơ chế hai chiều. Thiết bị gửi dữ liệu lên Cloud thông qua các cảm biến được gắn sẵn, ví dụ như cảm biến nhiệt độ, độ ẩm, ánh sáng hoặc mức khí độc. Dữ liệu này có thể được gửi theo chu kỳ định sẵn hoặc theo yêu cầu của ứng dụng, giúp người dùng giám sát trạng thái môi trường và các thông số cảm biến theo thời gian thực.

Ngược lại, Blynk Cloud nhận các lệnh điều khiển từ người dùng và truyền xuống thiết bị. Khi thao tác trên Blynk.App hoặc Blynk.Console, chẳng hạn như bật/tắt rơ-le, thay đổi ngưỡng cảnh báo hay cập nhật cấu hình, lệnh sẽ được Cloud xử lý và gửi tiếp đến thiết bị tương ứng. Thiết bị nhận lệnh và thực hiện ngay các hành động cần thiết, nhờ đó việc điều khiển từ xa trở nên tiện lợi, chính xác và nhanh chóng.

Quá trình truyền dữ liệu giữa thiết bị và Blynk Cloud sử dụng giao thức riêng của Blynk (Blynk Protocol) hoặc có thể tích hợp các giao thức phổ biến như MQTT hay HTTP, tùy theo cấu hình của hệ thống. Dữ liệu được mã hóa và xử lý theo thời gian thực, đảm bảo tính bảo mật, chính xác và ổn định. Nhờ cơ chế này, hệ thống Blynk cho phép người dùng vừa giám sát các cảm biến, vừa điều khiển thiết bị từ xa mà không cần

kết nối trực tiếp, tạo điều kiện thuận lợi cho việc quản lý, giám sát và tự động hóa thiết bị trong nhiều ứng dụng IoT khác nhau.

Hiện thị và điều khiển trên ứng dụng:

Người dùng có thể truy cập hệ thống qua:

- Blynk.Console (Trình duyệt web) để theo dõi trạng thái thiết bị, biểu đồ dữ liệu, lịch sử đo và cấu hình hệ thống.
- Blynk.App (Ứng dụng điện thoại) để xem thông tin cảm biến, điều khiển thiết bị từ xa bằng giao diện trực quan (Widget).

Khi người dùng thao tác trên ứng dụng, ví dụ nhấn nút “Bật quạt”, lệnh sẽ được gửi tới Blynk Cloud, rồi chuyển tiếp xuống thiết bị tương ứng.

Ngược lại, khi cảm biến đo được giá trị mới, thiết bị gửi dữ liệu lên Cloud để ứng dụng hiển thị ngay lập tức.

Tự động hóa và cảnh báo:

Blynk cho phép thiết lập các kịch bản tự động để thiết bị phản hồi theo điều kiện môi trường. Người dùng có thể đặt ra các quy tắc như: khi nhiệt độ vượt 30°C thì hệ thống tự bật quạt hoặc gửi thông báo cảnh báo về điện thoại.

Các kịch bản này được thiết lập ngay trong Blynk.App hoặc Blynk.Console, và toàn bộ quá trình xử lý được thực hiện trên máy chủ Blynk Cloud, giúp thiết bị tự vận hành mà không cần can thiệp thủ công.

Quản lý và cập nhật thiết bị:

Blynk không chỉ hỗ trợ giám sát mà còn cho phép quản trị toàn bộ thiết bị từ xa:

- Dễ dàng quản lý danh sách thiết bị, phân quyền người dùng và tổ chức dự án.
- Hỗ trợ OTA (Over-The-Air), giúp cập nhật firmware mới, vá lỗi hoặc bổ sung tính năng mà không cần thao tác trực tiếp trên phần cứng.
- Theo dõi tình trạng hoạt động của thiết bị như online/offline, cường độ tín hiệu WiFi, mức pin và lịch sử hoạt động (logs).

Các khả năng này giúp hệ thống vận hành ổn định, dễ bảo trì, và tối ưu cho các mô hình IoT quy mô nhỏ đến lớn.

1.2.3. Ứng dụng của Blynk trong hệ thống IoT

Blynk là một nền tảng IoT mạnh mẽ cho phép kết nối, giám sát và điều khiển các thiết bị phần cứng thông qua Internet. Trong các hệ thống IoT hiện đại, Blynk được ứng dụng rộng rãi nhờ khả năng tích hợp linh hoạt, giao diện trực quan và hỗ trợ truyền dữ liệu thời gian thực. Một số ứng dụng tiêu biểu của Blynk trong hệ thống IoT bao gồm:

Giám sát dữ liệu thời gian thực (Real – Time Monitoring):

Blynk cho phép thu thập và hiển thị dữ liệu từ các cảm biến như nhiệt độ, độ ẩm, ánh sáng, độ rung hoặc chất lượng nước,... Người dùng có thể theo dõi các thông số này trực tiếp trên điện thoại hoặc giao diện Web Dashboard bất cứ lúc nào. Điều này giúp hệ thống hoạt động minh bạch, dễ quản lý và kịp thời phát hiện bất thường.

Điều khiển thiết bị từ xa (Remote Control):

Người dùng có thể điều khiển các thiết bị như quạt, đèn, máy bơm, relay, ... thông qua các widget như button, switch hoặc slider. Việc điều khiển từ xa thông qua Blynk giúp hệ thống vận hành linh hoạt, tiết kiệm thời gian và dễ dàng thao tác ngay cả khi người dùng không có mặt trực tiếp tại khu vực lắp đặt.

Tự động hóa theo điều kiện (Automation Based on Conditions):

Blynk hỗ trợ thiết lập các điều kiện tự động như bật quạt khi nhiệt độ vượt ngưỡng hoặc bật đèn khi độ sáng thấp. Kết hợp giữa lập trình trên vi điều khiển và Automation của Blynk Cloud, hệ thống có thể hoạt động tự động, chính xác và liên tục mà không cần sự can thiệp thủ công.

Lập lịch hoạt động (Scheduling):

Blynk cho phép người dùng đặt lịch cho các thiết bị hoạt động theo thời gian định sẵn, chẳng hạn như bật đèn vào buổi tối hoặc tưới cây và buổi sáng. Tính năng này đặc biệt hữu ích trong các mô hình nông nghiệp thông minh, chuồng trại thông minh hoặc các ứng dụng tiết kiệm năng lượng.

Cảnh báo và thông báo (Notifications and Alerts):

Blynk có khả năng thông báo đến điện thoại người dùng khi hệ thống phát hiện sự cố hoặc giá trị vượt ngưỡng.

Quản lý nhiều thiết bị và mở rộng hệ thống:

Blynk hỗ trợ kết nối nhiều thiết bị IoT trong cùng một dự án, cho phép quản lý tập trung và phân quyền sử dụng.

Xây dựng giao diện trực quan để sử dụng:

Blynk cung cấp giao diện kéo - thả giúp người dùng dễ dàng tạo Dashboard hiển thị dữ liệu cảm biến, đồ thị, trạng thái thiết bị, lịch hoạt động và các nút điều khiển. Việc thiết kế giao diện không yêu cầu kiến thức lập trình, giúp tiết kiệm thời gian triển khai và tăng tính thẩm mỹ cho hệ thống.

1.3. Ngôn ngữ lập trình Arduino

1.3.1. Khái niệm về ngôn ngữ lập trình Arduino

Định nghĩa ngôn ngữ lập trình Arduino

Ngôn ngữ lập trình Arduino là một ngôn ngữ đơn giản và cơ bản để lập trình các vi điều khiển Arduino – là một nền tảng phần cứng mã nguồn mở. Ngôn ngữ lập trình Arduino dựa trên ngôn ngữ C/C++, nhưng đã được tùy chỉnh và đơn giản hóa để phù hợp với vi điều khiển Arduino và các tác vụ điều khiển phần cứng.

Lịch sử và nguồn gốc phát triển

Arduino có nguồn gốc từ một dự án nghiên cứu tại Interaction Design Institute Ivrea (IDII) ở Ý. Nền tảng này được phát triển vào khoảng năm 2005 bởi nhóm sáng lập gồm Massimo Banzi, David Cuartielles, David Mellis, Tom Igoe và Gianluca Martino. Dự án ban đầu sử dụng vi điều khiển Atmel AVR và được xây dựng dựa trên ngôn ngữ lập trình Wiring, sau này phát triển thành Arduino Language dựa trên C/C++.

Tên gọi “Arduino” được lấy từ quán Bar di Re Arduino ở Ivrea – nơi nhóm sáng lập thường họp bàn ý tưởng. Ngay từ đầu, nhóm đã quyết định mở mã nguồn phần cứng và phần mềm, tạo nền tảng cộng đồng mạnh mẽ cho việc học tập, sáng tạo và chia sẻ.

Ngày nay, Arduino trở thành một trong những nền tảng phổ biến nhất cho các dự án IoT, robot, tự động hóa và giáo dục STEM, nhờ đặc tính đơn giản, dễ học và khả năng mở rộng mạnh mẽ của cộng đồng.

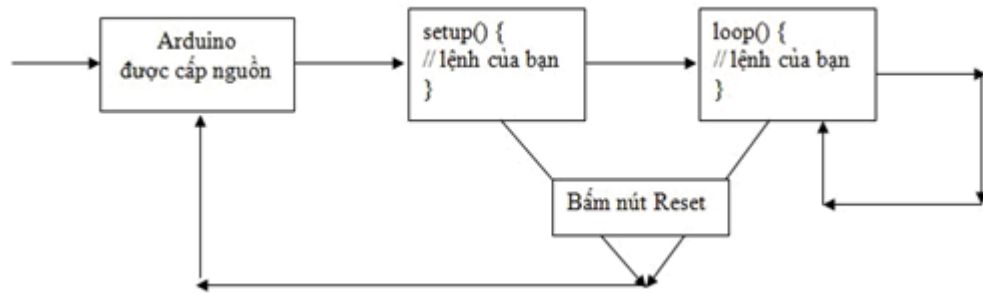
Chương trình Arduino có thể chia làm 3 phần: cấu trúc chương trình (structure), biến số (variable) và hằng số (constant), hàm và thủ tục (function).

Cấu trúc chương trình (Structure):

Một chương trình Arduino (gọi là sketch) thường có hai hàm bắt buộc:

- **setup():** hàm được gọi một lần khi thiết bị khởi động, dùng để thiết lập ban đầu (khởi tạo biến, chế độ pin,...).

- **loop():** hàm chính sẽ được gọi liên tục sau setup(), chứa logic lập cho chương trình chạy cho tới khi ngắt hoặc tắt máy.



Hình 1.11. Sơ đồ luồng thực thi của chương trình Arduino.

Ngoài ra, cấu trúc chương trình cũng cần khai báo trước các biến/hằng, thư viện nếu dùng và định nghĩa các hàm phụ nếu có.

Biến và hằng số (Variable & Constants):

Ngôn ngữ Arduino sử dụng biến để lưu trữ dữ liệu thay đổi trong khi chương trình chạy, và hằng số cho dữ liệu cố định.

Các kiểu dữ liệu thường gặp như int, float, char, boolean, cùng với các hằng số (#define) hoặc const

Việc dùng biến, hằng đúng cách giúp chương trình rõ ràng hơn, dễ bảo trì và tránh lỗi.

Hàm và thủ tục (Functions):

Hàm là khối mã thực hiện một nhiệm vụ cụ thể, có thể được gọi nhiều lần. Trong Arduino, có các hàm có sẵn có để điều khiển phần cứng, toán học, biến đổi kiểu dữ liệu, giao tiếp,...

Ví dụ một số nhóm hàm nổi bật:

- **Digital I/O:** pinMode(), digitalRead(), digitalWrite() – dùng để cấu hình và điều khiển các chân số.
- **Time & Delay:** millis(), micro(), delay() - quản lý thời gian và trễ trong chương trình.
- **Math:** sqrt(), sin(), cos(), pow() – xử lý toán học và số học nâng cao.
- **Giao tiếp:** Serial.print() – giao tiếp vật lý

Việc tổ chức hàm rõ ràng giúp mã nguồn gọn gàng, dễ test và tái sử dụng được.

1.3.2. Ứng dụng của ngôn ngữ lập trình Arduino trong IoT

Ngôn ngữ lập trình Arduino, dựa trên nền tảng C/C++, được thiết kế đơn giản và thân thiện, giúp người dùng dễ dàng viết chương trình điều khiển phần cứng. Trong các ứng dụng Internet of Things (IoT), Arduino đóng vai trò quan trọng trong việc thu thập dữ liệu từ cảm biến, xử lý thông tin và truyền dữ liệu lên Internet để giám sát và điều khiển từ xa.

Arduino thường được sử dụng trong các dự án như nhà thông minh, nông nghiệp thông minh, hệ thống giám sát môi trường, đo lường năng lượng, hoặc điều khiển thiết bị qua Internet. Với sự hỗ trợ của các thư viện có sẵn, lập trình viên có thể nhanh chóng kết nối với các module Wifi (như ESP8266, ESP32) hoặc nền tảng đám mây (như Blynk, ThingSpeak, Firebase...).

Ngôn ngữ Arduino còn hỗ trợ tích hợp nhiều loại cảm biến và module ngoại vi như cảm biến nhiệt độ, độ ẩm, ánh sáng, pH, TDS, OLED, relay... giúp xây dựng hệ thống IoT linh hoạt, tiết kiệm chi phí và dễ mở rộng. Nhờ đó, Arduino trở thành một công cụ mạnh mẽ và phổ biến trong nghiên cứu, học tập và phát triển các ứng dụng IoT thực tế.

1.3.3. Các thư viện sử dụng trong hệ thống

Trong các dự án IoT, Arduino và ESP32 thường được sử dụng để thu thập dữ liệu từ cảm biến, xử lý thông tin và kết nối với các thiết bị khác hoặc Internet. Để thực hiện những chức năng này, chương trình cần sử dụng một số thư viện hỗ trợ. Mỗi thư viện đảm nhiệm một vai trò riêng, từ giao tiếp với cảm biến, đo tín hiệu analog, đọc nhiệt độ, hiển thị dữ liệu trên màn hình, cho đến kết nối Wifi và tương tác với nền tảng IoT như Blynk. Việc hiểu rõ chức năng và ứng dụng của từng thư viện sẽ giúp lập trình viên tổ chức chương trình khoa học, tối ưu phần cứng và dễ dàng mở rộng dự án.

Sau đây là phần giới thiệu từng thư viện được sử dụng phổ biến trong các dự án IoT với Arduino/ESP32:

#include <Wire.h>

Thư viện Wire cho phép Arduino hoặc ESP32 giao tiếp với các thiết bị ngoại vi qua giao thức I2C, chỉ cần hai dây: SDA (dữ liệu) và SCL (đồng hồ). Nó giúp gửi và nhận dữ liệu từ cảm biến, màn hình OLED hay module ADC mà không phải viết code phức tạp, làm dự án gọn gàng và dễ quản lý.

#include <Wifi.h>

Thư viện Wifi quản lý kết nối Wifi cho ESP32, giúp thiết bị kết nối Internet để gửi và nhận dữ liệu từ xa. Nó cho phép giám sát cảm biến, điều khiển thiết bị qua cloud hoặc smartphone, là thành phần quan trọng trong các dự án IoT.

#include < BlynkSimpleEsp32.h>

Thư viện này kết nối ESP32 với ứng dụng Blynk, cho phép tạo giao diện điều khiển trên điện thoại, nhận dữ liệu cảm biến và điều khiển thiết bị từ xa. Nó giúp người dùng tương tác với hệ thống IoT dễ dàng mà không cần lập trình mạng phức tạp.

#include < DHT.h>

Thư viện dùng để đọc dữ liệu từ cảm biến DHT22. Nó cung cấp các hàm đo nhiệt độ và độ ẩm, phục vụ thuật toán tự động bật quạt hoặc đèn sưởi dựa trên ngưỡng nhiệt độ cài đặt.

#include < IRremote.h>

Thư viện hỗ trợ giải mã tín hiệu từ remote hồng ngoại, cho phép người dùng điều khiển thiết bị theo chế độ thủ công. Nhờ thư viện này, hệ thống nhận biết từng nút nhấn và xử lý chính xác ngay cả khi tín hiệu bị lặp.

#include < Adafruit_SSD1306.h>

Thư viện Adafruit_SSD1306 điều khiển màn hình OLED SSD1306, hiển thị chữ, số, biểu đồ hoặc biểu tượng. Nó giúp giám sát dữ liệu trực tiếp trên thiết bị mà không cần smartphone, thuận tiện trong các dự án IoT tại chỗ.

1.4. Môi trường phát triển Arduino IDE

1.4.1. Giới thiệu Arduino IDE

Arduino IDE là gì?

Arduino IDE (Môi trường Phát triển Tích hợp) là một nền tảng phần mềm được sử dụng để viết, biên dịch và nạp mã chương trình vào các bo mạch Arduino. Nó cung cấp một giao diện đơn giản để lập trình bằng ngôn ngữ tương tự C/C++, được thiết kế đặc biệt để tương tác với phần cứng Arduino.



Hình 1.12. Logo Arduino IDE.

IDE giúp người dùng viết các sketch (chương trình Arduino), kiểm tra mã nguồn, và nạp chương trình vào vi điều khiển chỉ với vài cú nhấp chuột.

Ngoài ra, Arduino IDE còn:

- Hỗ trợ thư viện phong phú: Cho phép dễ dàng kết nối các cảm biến, động cơ, màn hình và các module khác mà không cần viết lại mã từ đầu.
- Đa nền tảng sử dụng: Có thể chạy trên Windows, macOS, Linux và giúp lập trình viên dễ dàng sử dụng.
- Giao diện thân thiện với người mới: Ngay cả những người chưa biết lập trình cũng có thể bắt đầu nhanh chóng.
- Hỗ trợ cộng đồng lớn: Hàng ngàn ví dụ, hướng dẫn và forum giúp giải quyết vấn đề nhanh chóng.

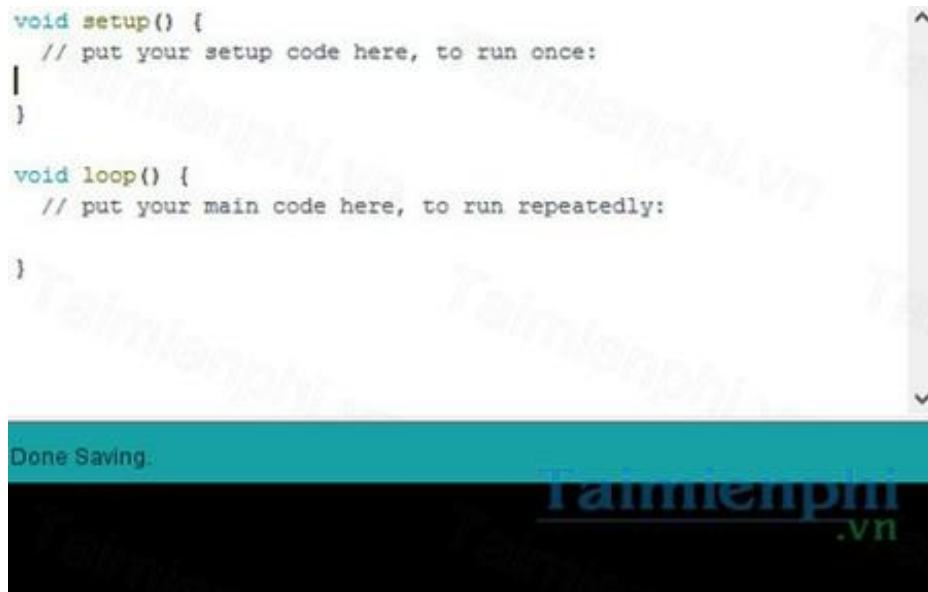
Nhờ những tính năng này, Arduino IDE trở thành công cụ lý tưởng cho cả người mới học điện tử lẫn các dự án chuyên sâu, từ robot, IoT, đến các thiết bị thông minh.

1.4.2. Các chức năng chính của Arduino IDE

Các chức năng chính của Arduino IDE bao gồm:

Trình soạn thảo văn bản:

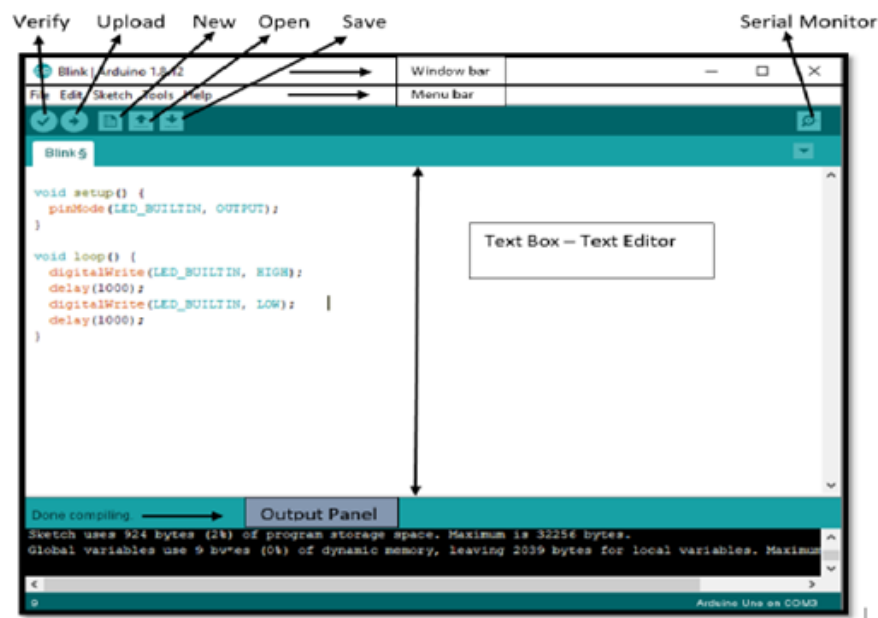
Arduino IDE cung cấp giao diện soạn thảo mã nguồn đơn giản, hỗ trợ tô màu cú pháp, gợi ý cấu trúc chương trình và hiển thị hai hàm mặc định `setup()` và `loop()`. Đây là nơi người dùng viết và chỉnh sửa toàn bộ chương trình điều khiển vi điều khiển.



Hình 1.13. Trình soạn thảo mã nguồn trong Arduino IDE.

Chức năng biên dịch chương trình:

Nút Verify giúp kiểm tra lỗi cú pháp và biên dịch chương trình thành mã máy. Nếu có lỗi IDE sẽ báo vị trí và loại lỗi ở khung thông báo phía dưới giúp người dùng sửa nhanh chóng.



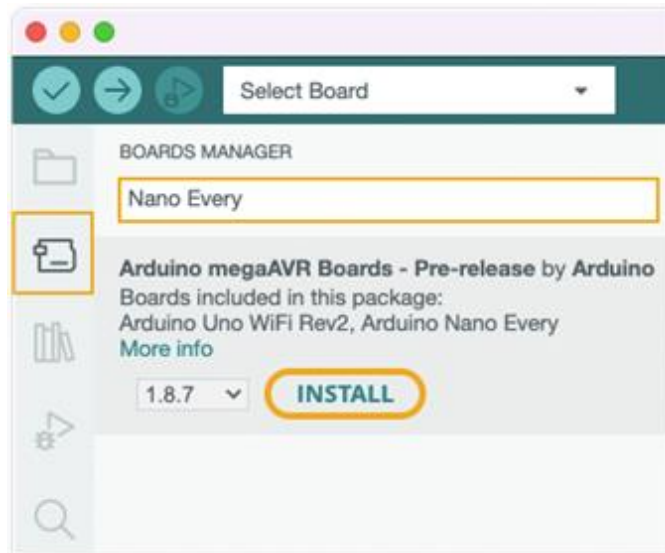
Hình 1.14. Chức năng Verify - Upload trong Arduino IDE.

Chức năng nạp chương trình:

Nút Upload cho phép nạp mã đã biên dịch vào bo mạch Arduino thông qua cổng USB. Quá trình upload tự động thực hiện bước biên dịch trước khi nạp chương trình.

Quản lý board mạch:

Arduino IDE hỗ trợ nhiều loại board khác nhau. Người dùng có thể cài đặt thêm board tại Boards Manager và lựa chọn đúng loại board trước khi dịch chương trình.



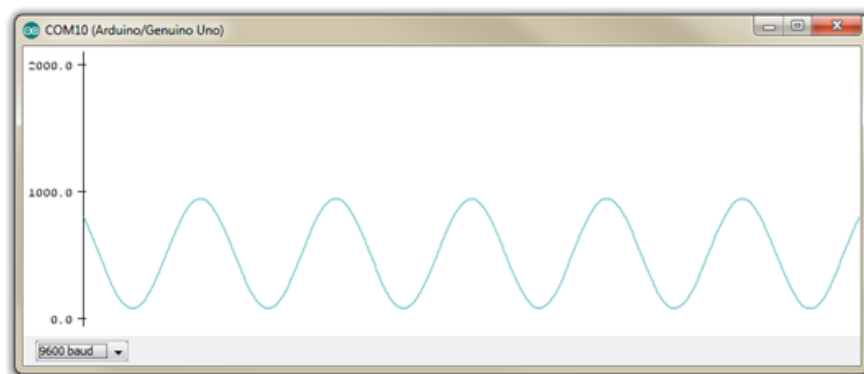
Hình 1.15. Giao diện Boards Manager.

Serial Monitor:

Serial Monitor cho phép hiển thị dữ liệu được gửi từ vi điều khiển về máy tính, hỗ trợ kiểm tra giá trị cảm biến và debug chương trình. Đây là công cụ thường được sử dụng trong các dự án IoT.

Serial Plotter:

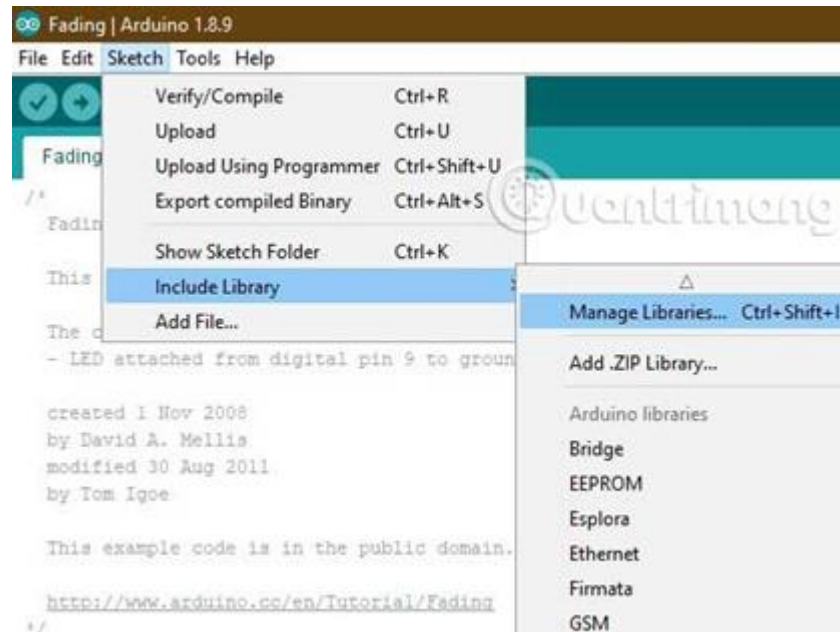
Serial Plotter giúp trực quan hóa dữ liệu dạng đồ thị theo thời gian, và đặc biệt khi theo dõi nhiệt độ, độ ẩm, ...



Hình 1.16. Giao diện Serial Plotter.

Quản lý thư viện:

Library Manager là công cụ cài đặt và quản lý các thư viện dùng cho cảm biến và module như DHT, Blynk,... giúp mở rộng chức năng lập trình.



Hình 1.17. Giao diện Library Manager.

1.4.3. Ưu điểm và nhược điểm của Arduino IDE trong lập trình IoT

Arduino IDE là phần mềm lập trình phổ biến cho các bo mạch Arduino và vi điều khiển nhờ giao diện đơn giản và dễ sử dụng. Bên cạnh những ưu điểm nổi bật, phần mềm cũng tồn tại một số hạn chế nhất định trong quá trình phát triển các dự án lớn hoặc yêu cầu kỹ thuật cao.

Ưu điểm của Arduino IDE:

- Arduino IDE có giao diện trực quan, dễ thao tác, phù hợp cho người mới tiếp cận lập trình nhúng và IoT.
- Phần mềm tích hợp các chức năng quan trọng như viết mã, biên dịch, nạp chương trình và theo dõi dữ liệu qua Serial Monitor, giúp quá trình phát triển diễn ra nhanh chóng.
- Hỗ trợ nhiều loại board mạch và thư viện mở rộng, kèm theo hệ thống ví dụ mẫu phong phú, giúp người dùng dễ dàng thử nghiệm và học tập.

Nhược điểm của Arduino IDE:

- Khả năng hỗ trợ lập trình nâng cao còn hạn chế, thiếu các tính năng như tự động gợi ý mã, phân tích lỗi nâng cao hoặc quản lý dự án lớn.

- Không hỗ trợ debug chuyên sâu, việc tìm lỗi chủ yếu dựa vào Serial Monitor nên đôi khi tốn thời gian.
- Tốc độ biên dịch chưa thật sự tối ưu và giao diện còn đơn giản so với các IDE hiện đại như VS Code hoặc PlatformIO.

1.5. Giới thiệu về các thiết bị phần cứng

1.5.1. Thiết bị điều khiển

Trong hệ thống IoT, MCU đóng vai trò “bộ xử lý trung tâm”, nhận dữ liệu từ cảm biến, xử lý thông tin và điều khiển các bộ phận chấp hành như relay, LED, quạt hay động cơ. MCU cũng đảm nhiệm truyền thông, gửi dữ liệu lên các nền tảng IoT (Blynk, MQTT, Firebase) và nhận lệnh từ người dùng. Nhờ khả năng kết nối – xử lý – điều khiển, MCU giữ vai trò điều phối toàn bộ hoạt động của hệ thống.

Các thiết bị điều khiển phổ biến trong IoT hiện nay là:

ESP8266 NodeMCU:

ESP8266 NodeMCU là một bo mạch phát triển (development board) dựa trên chip WiFi ESP8266, tích hợp vi điều khiển và module WiFi, thường dùng để xây dựng các dự án IoT. Nó có khả năng lập trình dễ dàng qua Arduino IDE và hỗ trợ giao tiếp WiFi, giúp thiết bị kết nối Internet mà không cần thêm module rời.



Hình 1.18. ESP8266 NodeMCU.

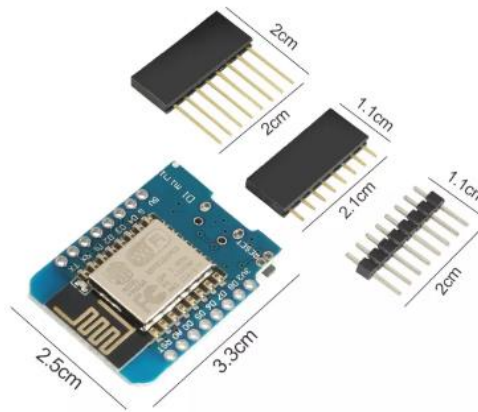
Ưu điểm: lập trình dễ với Arduino IDE, có mạch nạp USB tích hợp, phù hợp các mô hình IoT quy mô vừa.

Nhược điểm: kích thước board khá lớn, không thuận tiện cho thiết bị cần tối ưu diện tích hoặc PCB nhỏ.

Ứng dụng: triển khai dự án IoT phổ thông, điều khiển thiết bị qua Internet, hệ thống smart home, web server nhỏ và các mô hình prototyping yêu cầu nhiều GPIO.

ESP8266 Wemos D1 Mini:

ESP8266 Wemos D1 Mini là một bo mạch phát triển nhỏ gọn dựa trên chip WiFi ESP8266, thường dùng trong các dự án IoT. So với NodeMCU, Wemos D1 Mini có kích thước nhỏ hơn, vẫn tích hợp vi điều khiển và module WiFi, dễ lập trình qua Arduino IDE.



Hình 1.19. ESP8266 Wemos D1 Mini.

Ưu điểm: siêu nhỏ gọn, dễ tích hợp PCB, giá rẻ, tiêu thụ thấp, phù hợp thiết bị IoT mini.

Nhược điểm: ít GPIO hơn NodeMCU, khó cắm breadboard nếu không dùng header.

Ứng dụng: IoT mini và wearable, PCB có WiFi tích hợp, các cảm biến môi trường và ánh sáng, module điều khiển nhỏ như LED WS2812 hoặc công tắc mini.

Arduino Uno:

Arduino Uno là một bo mạch vi điều khiển phổ biến dựa trên chip ATmega328P.



Hình 1.20. Arduino UNO.

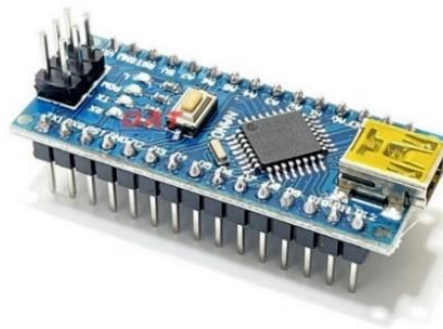
Ưu điểm: dễ thao tác, phù hợp người mới, cổng USB lớn bền, tương thích nhiều shield.

Nhược điểm: kích thước cổng kênh, không phù hợp thiết bị nhúng nhỏ.

Ứng dụng: bài học và thí nghiệm nhập môn, robot đơn giản, prototype với nhiều shield, dạy học STEM, hệ thống điều khiển cơ bản như đo nhiệt độ, độ ẩm, relay.

Arduino Nano:

Arduino Nano là phiên bản nhỏ gọn của Arduino Uno, cũng dựa trên chip ATmega328P.



Hình 1.21. Arduino Nano.

Ưu điểm: Kích thước nhỏ, dễ tích hợp vào các mạch nhỏ, lập trình đơn giản qua Arduino IDE.

Nhược điểm: Số lượng chân I/O ít hơn Uno, hạn chế trong các dự án cần nhiều thiết bị ngoại vi.

Ứng dụng: dự án nhúng, PCB, robot mini, IoT nhỏ, thiết bị wearable, tự động hóa nhỏ như máy bơm, LED, cảnh báo.

1.5.2. Thiết bị cảm biến

Cảm biến là thành phần không thể thiếu trong các hệ thống IoT, đảm nhiệm chức năng thu thập dữ liệu từ môi trường hoặc từ các thiết bị cần giám sát. Chúng chuyển đổi các đại lượng vật lý, hóa học hoặc môi trường thành tín hiệu điện (analog hoặc digital) để vi điều khiển xử lý. Nhờ đó, hệ thống IoT có thể theo dõi trạng thái, đánh giá điều kiện hoạt động và đưa ra các quyết định điều khiển tự động.

Các loại cảm biến được sử dụng rộng rãi trong IoT hiện nay bao gồm:

Cảm biến ánh sáng (LDR, BH1750):

Cảm biến ánh sáng đo cường độ sáng môi trường, giúp hệ thống nhận biết mức chiếu sáng hiện tại. Thường dùng LDR (tín hiệu analog) hoặc BH1750 (digital, chính xác cao), ứng dụng trong chiếu sáng tự động, smarthome và các dự án IoT.



Hình 1.22. Cảm biến ánh sáng.

Ưu điểm: dữ liệu nhanh, dễ tích hợp, LDR giá rẻ, BH1750 chính xác cao.

Nhược điểm: LDR không tuyến tính, dễ nhiễu; BH1750 cần I2C, phức tạp hơn.

Ứng dụng: chiếu sáng tự động, smarthome, giám sát ánh sáng, camera thông minh.

Cảm biến siêu âm (HC-SR04):

Cảm biến HC-SR04 đo khoảng cách đến vật thể bằng sóng siêu âm, không tiếp xúc trực tiếp.



Hình 1.23. Cảm biến siêu âm.

Ưu điểm: giá rẻ, chính xác, không tiếp xúc, dễ tích hợp robot và IoT.

Nhược điểm: không đo vật mỏng, mềm, nghiêng; khoảng cách hạn chế, dễ nhiễu môi trường ồn.

Ứng dụng: robot tránh vật cản, giám sát mực nước, bồn chứa, tự động hóa công nghiệp.

1.5.3. Thiết bị chấp hành

Thiết bị chấp hành trong IoT thực hiện các lệnh từ vi điều khiển, chuyển tín hiệu điện thành hành động vật lý như bật/tắt, quay, phát sáng hay cảnh báo. Chúng giúp hệ thống can thiệp trực tiếp vào môi trường, tự động hóa quy trình và phản hồi theo dữ liệu cảm biến.

Các thiết bị chấp hành phổ biến trong IoT hiện nay bao gồm:

Servo Motor:

Servo Motor giúp điều khiển góc quay chính xác, phù hợp cho robot, cửa tự động hoặc cơ cấu mở/đóng.



Hình 1.24. Servo Motor.

Ưu điểm: độ chính xác cao, lực kéo mạnh ở loại lớn.

Nhược điểm: giá cao hơn động cơ DC, tầm quay giới hạn ở loại servo góc.

Ứng dụng: cửa tự động, robot cánh tay, camera gimbal, cơ cấu cần điều khiển góc chính xác.

Máy bơm mini:

Máy bơm mini giúp bơm hoặc dẫn chất lỏng trong hệ thống tự động.



Hình 1.25. Máy bơm Mini.

Ưu điểm: giá rẻ, đáp ứng nhanh, lưu lượng ổn định.

Nhược điểm: không chạy liên tục lâu, cần lọc rác chống nghẹt.

Ứng dụng: tưới cây tự động, bơm nước hồ cá, thủy cảnh, điều khiển theo cảm biến độ ẩm hoặc lịch tưới.

1.5.4. Thiết bị hiển thị

Thiết bị hiển thị là thành phần quan trọng trong các hệ thống IoT, cho phép trình bày thông tin thu thập được từ cảm biến hoặc trạng thái hoạt động của thiết bị một cách trực quan. Chúng giúp người dùng quan sát dữ liệu, theo dõi quá trình vận hành và đưa ra các thao tác điều khiển phù hợp. Thiết bị hiển thị có thể thể hiện dữ liệu dưới dạng chữ, số, biểu tượng hoặc đồ họa tùy theo nhu cầu của hệ thống.

Các loại thiết bị hiển thị được sử dụng phổ biến trong IoT hiện nay bao gồm:

LCD 16x2:

LCD 16x2 là màn hình hiển thị ký tự gồm 2 dòng, mỗi dòng 16 ký tự, dùng để hiển thị các thông tin cơ bản như số liệu hoặc trạng thái thiết bị. Với giá thành thấp, dễ kết nối và độ ổn định cao, LCD 16x2 được sử dụng rộng rãi trong các dự án IoT và hệ thống nhúng.



Hình 1.26. LCD 16x2.

Ưu điểm: giá rẻ, dễ lập trình, hiển thị thông tin cơ bản trực tiếp.

Nhược điểm: không hiển thị hình ảnh/biểu đồ phức tạp, kích thước hạn chế.

Ứng dụng: giám sát nhiệt độ, độ ẩm, áp suất, thiết bị IoT, smarthome, mạch nhúng.

LED 7 đoạn:

LED 7 đoạn là bộ hiển thị số được tạo từ 7 thanh LED ghép lại, dùng để hiển thị các chữ số từ 0–9. Nhờ đơn giản và rẻ, nó thường xuất hiện trong đồng hồ số và các thiết bị điện tử cơ bản.



Hình 1.27. LED 7 đoạn.

Ưu điểm: hiển thị số rõ, giá rẻ, đơn giản, dễ điều khiển, tiết kiệm điện, bền.

Nhược điểm: chỉ hiển thị số, cần nhiều chân nếu không dùng drive, hạn chế thông tin/đồ họa.

Ứng dụng: đồng hồ số, bộ đếm/đo tốc độ/khoảng cách, hiển thị giá trị ADC, IoT hiển thị nhiệt độ, số lần đếm hoặc trạng thái hệ thống.

Chương 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

2.1. Phân tích yêu cầu chức năng

Hệ thống được xây dựng nhằm giám sát liên tục các thông số môi trường trong chuồng nuôi, đặc biệt là nhiệt độ và độ ẩm, giúp người dùng theo dõi điều kiện nuôi theo thời gian thực và phát hiện sớm các dấu hiệu bất thường. Trong thực tế, hệ thống đóng vai trò như một công cụ giám sát – điều khiển tự động: nó thu thập dữ liệu từ môi trường, hiển thị thông tin cho người sử dụng và tự động kích hoạt quạt, đèn sưởi hoặc đưa ra cảnh báo khi các giá trị vượt quá ngưỡng an toàn. Bên cạnh đó, hệ thống còn hỗ trợ điều khiển thủ công thông qua ứng dụng Blynk hoặc remote hồng ngoại, cho phép người dùng can thiệp trực tiếp vào các thiết bị khi cần thiết.

Nhờ kết hợp giữa khả năng giám sát, cảnh báo và điều khiển linh hoạt, người chăn nuôi có thể dễ dàng điều chỉnh điều kiện nuôi để đảm bảo sức khỏe và sự phát triển ổn định của vật nuôi. Từ những chức năng thực tế nêu trên, hệ thống được thiết kế với các mô-đun phần cứng và phần mềm phù hợp, đảm bảo hoạt động chính xác và ổn định. Các yêu cầu chức năng chính của hệ thống bao gồm:

2.1.1. Thu thập dữ liệu từ các cảm biến

Hệ thống thu thập dữ liệu môi trường theo thời gian thực thông qua cảm biến nhiệt độ – độ ẩm DHT22. ESP32 phải đọc và xử lý các giá trị đo được, sau đó cập nhật liên tục để phục vụ cho việc điều khiển tự động và hiển thị. Việc thu thập dữ liệu phải ổn định, chính xác và diễn ra theo chu kỳ định kỳ.

2.1.2. Nhận tín hiệu điều khiển từ Remote hồng ngoại

Hệ thống hỗ trợ chức năng nhận tín hiệu từ Remote hồng ngoại thông qua module thu IR. Khi người dùng nhấn các nút trên Remote, ESP32 giải mã tín hiệu nhận được và thực hiện các thao tác tương ứng như bật - tắt quạt, đèn sưởi hoặc đèn chiếu sáng. Chức năng này cho phép điều khiển trực tiếp và nhanh chóng ngay cả khi không sử dụng ứng dụng Blynk.

2.1.3. Điều khiển thủ công qua ứng dụng Blynk

Hệ thống cần hỗ trợ chế độ điều khiển thủ công thông qua giao diện của ứng dụng Blynk, cho phép người dùng bật/tắt trực tiếp các thiết bị như quạt, đèn sưởi và đèn chiếu sáng. Khi người dùng thao tác trên các nút điều khiển, ESP32 sẽ nhận lệnh ngay lập tức và thực hiện đúng chức năng tương ứng, bỏ qua chế độ tự động trong thời gian thủ công được kích hoạt. Chức năng này mang lại sự linh hoạt trong quá trình vận hành, giúp

người dùng có thể chủ động can thiệp vào hệ thống khi cần thiết, đặc biệt trong những tình huống yêu cầu thao tác nhanh hoặc ngoài ngưỡng tự động được cài đặt.

2.1.4. Thiết lập cảnh báo khi giá trị vượt ngưỡng

Hệ thống phải có khả năng phát hiện và cảnh báo khi nhiệt độ hoặc độ ẩm vượt quá ngưỡng an toàn. Các ngưỡng này (ngưỡng thấp – ngưỡng cao) có thể được người dùng tùy chỉnh trực tiếp trên ứng dụng Blynk, giúp hệ thống linh hoạt với từng môi trường nuôi khác nhau.

Khi giá trị đo được vượt ngưỡng, ESP32 sẽ tự động kích hoạt các thiết bị chấp hành như quạt làm mát hoặc đèn sưởi để đưa môi trường trở lại trạng thái ổn định. Đồng thời, ứng dụng Blynk có thể hiển thị thông báo cảnh báo để người dùng kịp thời theo dõi và xử lý.

2.1.5. Hẹn giờ bật – tắt đèn chiếu sáng

Hệ thống cần hỗ trợ chức năng hẹn giờ bật/tắt đèn chiếu sáng theo lịch do người dùng thiết lập trên ứng dụng Blynk. Người dùng có thể cài đặt thời gian bắt đầu, thời gian kết thúc và chọn các ngày trong tuần áp dụng lịch. ESP32 sẽ tự động kích hoạt đèn theo đúng lịch trình đã đặt, kể cả khi mất kết nối Internet trong thời gian ngắn. Chức năng này giúp tiết kiệm năng lượng và đảm bảo ánh sáng ổn định cho chuồng nuôi.

2.1.6. Hiển thị dữ liệu lên ứng dụng Blynk và OLED

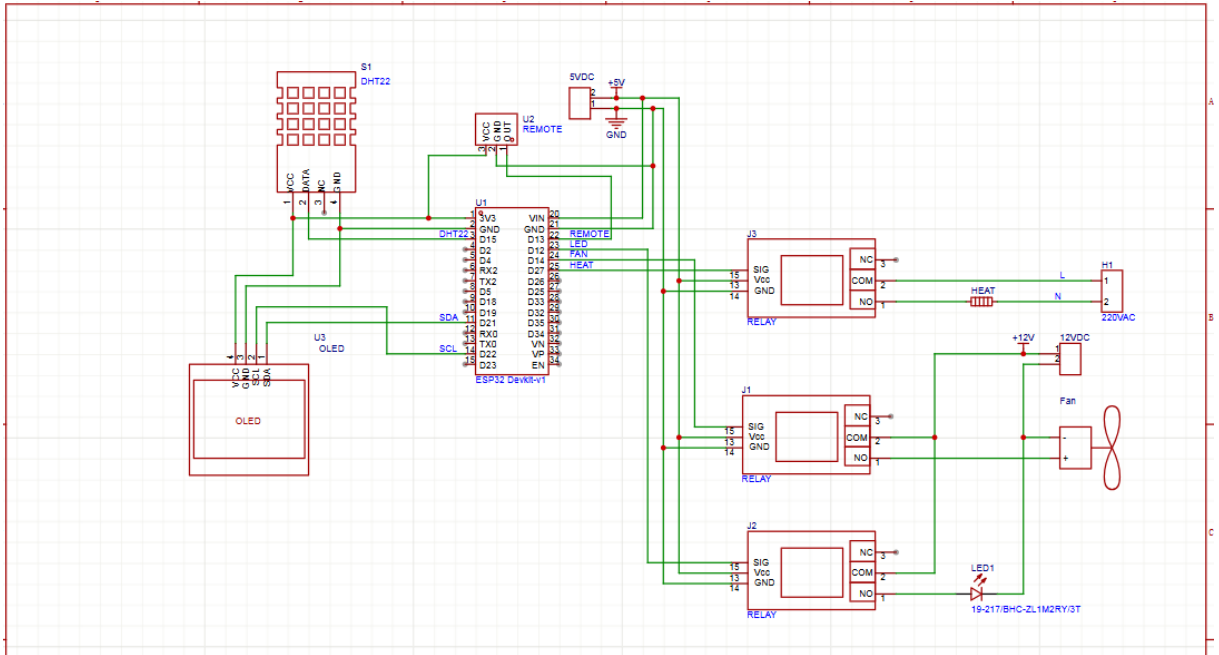
Hệ thống truyền dữ liệu thu thập được lên ứng dụng Blynk để người dùng theo dõi từ xa qua Internet. Đồng thời, nhiệt độ, độ ẩm, trạng thái thiết bị và chế độ hoạt động cũng cần được hiển thị trực tiếp trên màn hình OLED 0.96 inch. Việc hiển thị này giúp người dùng dễ dàng quan sát tình trạng hệ thống một cách trực quan.

2.1.7. Gửi thông báo cảnh báo qua Gmail khi vượt ngưỡng

Hệ thống cần có khả năng gửi cảnh báo qua Gmail khi nhiệt độ hoặc độ ẩm vượt quá ngưỡng an toàn do người dùng cài đặt. Khi phát hiện giá trị bất thường, ESP32 sẽ kích hoạt chức năng gửi thông báo thông qua dịch vụ của Blynk hoặc hệ thống IoT tích hợp, giúp người dùng nhận cảnh báo kịp thời ngay cả khi không mở ứng dụng. Chức năng này hỗ trợ giám sát từ xa hiệu quả hơn và nâng cao độ an toàn cho môi trường chuồng nuôi.

2.2. Thiết kế hệ thống

2.2.1. Thiết kế phần cứng



Hình 2.1. Sơ đồ thiết kế phần cứng.

Thiết bị xử lý:

ESP32 là vi điều khiển trung tâm của hệ thống, chịu trách nhiệm đọc dữ liệu từ cảm biến DHT22, xử lý và gửi lên Blynk để giám sát từ xa. ESP32 đồng thời giải mã tín hiệu từ Remote IR1838 để thực hiện điều khiển thủ công, hiển thị thông tin lên màn hình OLED và điều khiển các thiết bị chấp hành (quạt, đèn LED, sưởi) thông qua Module Relay.

Thiết bị cảm biến:

Cảm biến DHT22 đo nhiệt độ và độ ẩm không khí trong chuồng.

Thiết bị điều khiển từ xa:

Remote hồng ngoại IR1838 nhận tín hiệu từ remote để điều khiển các thiết bị quạt, đèn LED, sưởi hoặc chuyển đổi giữa các chế độ của hệ thống (thủ công/tự động).

Thiết bị hiển thị:

OLED SSD1306 hiển thị các thông số nhiệt độ, độ ẩm, trạng thái kết nối Blynk và trạng thái của hệ thống.

Thiết bị chấp hành:

Module Relay: Đóng/ngắt điện cho các thiết bị tải, là trung gian giữa ESP32 và quạt, đèn LED, sưởi.

Quạt tản nhiệt: Tự động bật khi nhiệt độ cao hơn ngưỡng điều chỉnh (khi ở hệ thống ở chế độ tự động) hoặc theo điều khiển từ Remote và Blynk (khi hệ thống ở chế độ thủ công).

Đèn LED chiếu sáng: Cung cấp ánh sáng và có thể bật/tắt theo lịch (khi hệ thống ở chế độ tự động) hoặc theo lệnh điều khiển qua remote và Blynk (khi hệ thống ở chế độ thủ công).

Sưởi: bật khi nhiệt độ thấp hơn ngưỡng điều chỉnh (khi hệ thống ở chế độ tự động) hoặc điều khiển từ Remote và Blynk (khi hệ thống ở chế độ thủ công).

2.2.1.1. Vi điều khiển ESP32

ESP32 là dòng vi điều khiển (Microcontroller) giá thành thấp và tiêu thụ năng lượng thấp, được tích hợp sẵn kết nối WiFi và Bluetooth dual-mode (gồm Bluetooth truyền thông và Bluetooth Low Energy – BLE). Vi điều khiển này sử dụng bộ xử lý Tensilica Xtensa LX6 với hai tùy chọn lõi đơn hoặc lõi kép, cho phép xử lý song song và đáp ứng tốt các tác vụ thời gian thực trong những ứng dụng IoT hiện đại. Bên trong ESP32 còn được tích hợp các thành phần như anten switch, RF balun, bộ khuếch đại công suất, bộ khuếch đại nhiễu thấp, bộ lọc và bộ quản lý năng lượng, giúp giảm số lượng linh kiện ngoại vi và nâng cao độ ổn định của toàn bộ hệ thống.

ESP32 được phát triển bởi Espressif Systems (Thượng Hải, Trung Quốc) và được sản xuất trên dây chuyền công nghệ 40 nm của TSMC. Đây là phiên bản kế thừa và nâng cấp đáng kể từ dòng ESP8266, được cải thiện về hiệu năng, khả năng kết nối không dây, độ an toàn bảo mật cũng như sự linh hoạt trong triển khai ứng dụng.

Lý do lựa chọn ESP32 trong hệ thống:

- Tích hợp WiFi và Bluetooth: Giúp kết nối trực tiếp với nền tảng IoT (như Blynk) mà không cần thêm module WiFi rời, từ đó giảm chi phí và tối ưu không gian phần cứng.
- Hiệu năng cao: Bộ xử lý lên đến 240 MHz đủ khả năng đọc nhiều cảm biến đồng thời (TDS, pH, DS18B20, DHT22, MPU6050), điều khiển relay, xử lý dữ liệu và giao tiếp với Blynk trong thời gian thực.

- Hỗ trợ nhiều giao thức giao tiếp: Bao gồm ADC, PWM, UART, SPI, I2C, phù hợp với hệ thống có nhiều thiết bị như quạt, đèn, màn hình OLED, Remote IR và các loại cảm biến.
- Tiêu thụ năng lượng thấp: Nhiều chế độ tiết kiệm năng lượng giúp hệ thống hoạt động ổn định trong thời gian dài.
- Hệ sinh thái phần mềm phong phú: Được hỗ trợ mạnh mẽ trên Arduino IDE, PlatformIO cùng nhiều thư viện và tài liệu cộng đồng, giúp quá trình lập trình, thử nghiệm và mở rộng trở nên dễ dàng hơn.
- Giá thành hợp lý: So với các vi điều khiển có tính năng tương đương, ESP32 mang lại hiệu suất rất tốt ở mức chi phí thấp, phù hợp cho các mô hình IoT trong đồ án tốt nghiệp.

Với những ưu điểm trên, ESP32 là lựa chọn phù hợp để làm bộ điều khiển trung tâm cho hệ thống, đảm bảo khả năng thu thập dữ liệu cảm biến, xử lý và truyền tải dữ liệu lên nền tảng điều khiển từ xa một cách ổn định, chính xác và hiệu quả.



Hình 2.2. Vi điều khiển ESP32.

Thông số kỹ thuật:

CPU và bộ nhớ:

- Vi xử lý (CPU): Xtensa® Dual-Core 32-bit LX6.
- Tốc độ xung nhịp: 160 MHz – 240 MHz.
- RAM: 520 KB SRAM.
- Flash: 4MB (SPI Flash tích hợp).

Kết nối không dây:

- Wifi: IEEE 802.11 b/g/n (2.4 GHz).
- Bluetooth: v4.2 BR/EDR và BLE (Bluetooth Low Energy).

Giao tiếp và I/O:

- GPIO: 34 chân (tùy bo có thể truy cập được ~25–30 chân).
- ADC: 12-bit, tối đa 18 kênh.
- DAC: 2 kênh (8-bit).
- PWM: Hỗ trợ trên tất cả GPIO.
- UART: 3 cổng.
- SPI: 3 cổng.
- I2C: 2 cổng.
- I2S: 2 cổng.
- CAN bus, IR (TX/RX), Touch Sensor (10 kênh).
- SD card/SDIO/MMC, Ethernet MAC (với DMA).

Cảm biến tích hợp:

- Cảm biến Hall (đo từ trường).
- Cảm biến nhiệt độ nội chip.
- 10 cảm biến chạm điện dung (touch sensor).

Bảo mật phần cứng:

- Hỗ trợ WPA/WPA2/WPA 3.
- Secure Boot và Flash Encryption.
- AES, SHA-2, RSA, ECC, RNG (bộ tăng tốc mã hóa phần cứng).
- 1024-bit OTP memory.

Nguồn và công suất:

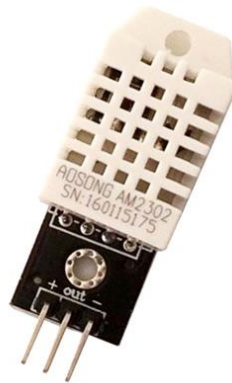
- Điện áp hoạt động: 2.2V – 3.6V (thường cấp 5V qua USB)
- Chân nguồn:
 - 5V: cấp nguồn từ USB.
 - 3.3V: đầu ra ổn định từ mạch ổn áp onboard.
- Nhiệt độ hoạt động: -40°C đến +85°C.

Ứng dụng:

- Hệ thống tự động hóa nhà thông minh.
- Cảm biến IoT và thu thập dữ liệu.
- Thiết bị đeo thông minh.
- Robot điều khiển từ xa.
- Ứng dụng phát triển phần mềm và phần cứng.

2.2.1.2. Cảm biến DHT22

DHT22 trong hệ thống được dùng để đo nhiệt độ và độ ẩm không khí chuồng trại. Nó cung cấp dữ liệu cho ESP32 để giám sát môi trường và tự động điều khiển thiết bị theo ngưỡng đặt trước.



Hình 2.3. Cảm biến DHT22.

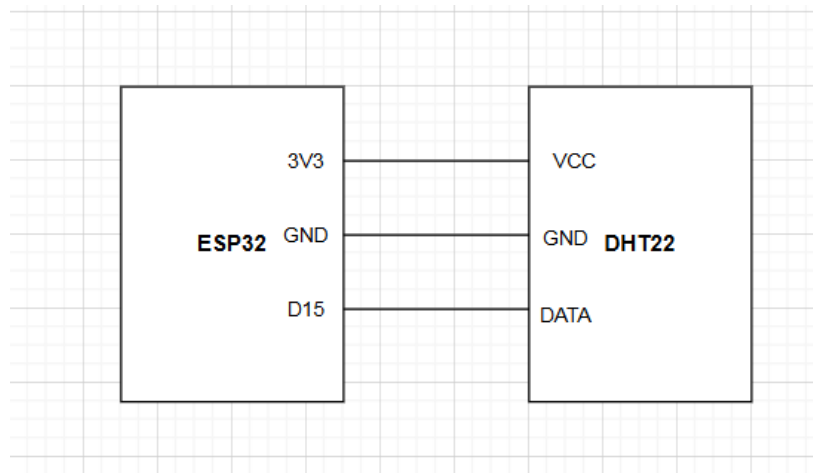
Thông số kỹ thuật:

- Điện áp hoạt động: 3.5V đến 5.5V.
- Dòng hoạt động: 0,3mA (đo) 60uA (chế độ chờ).
- Đầu ra: Dữ liệu nối tiếp.
- Phạm vi nhiệt độ: -40 ° C đến 80 ° C.
- Phạm vi độ ẩm: 0% đến 100%.
- Độ phân giải: Nhiệt độ và Độ ẩm đều là 16-bit.
- Độ chính xác: $\pm 0,5$ ° C và $\pm 1\%$.

Ứng dụng:

- Đo nhiệt độ và độ ẩm.
- Đo thời tiết địa phương.
- Kiểm soát khí hậu tự động.
- Giám sát môi trường.

Sơ đồ khối nối cảm biến DHT22 với ESP32:



Hình 2.4. Sơ đồ khối của cảm biến DHT22 với ESP32.

2.2.1.3. Module Relay 5V – 10A

Relay 5V–10A trong hệ thống được dùng để bật/tắt các thiết bị công suất lớn hơn như quạt, đèn sưởi, đèn LED. ESP32 chỉ cần xuất tín hiệu điều khiển 5V, còn lại Module Relay sẽ thực hiện đóng/cắt tải một cách an toàn nhờ cơ chế cách ly giữa mạch điều khiển và mạch điện AC/DC.



Hình 2.5. Module Relay 5V - 10A.

Thông số kỹ thuật:

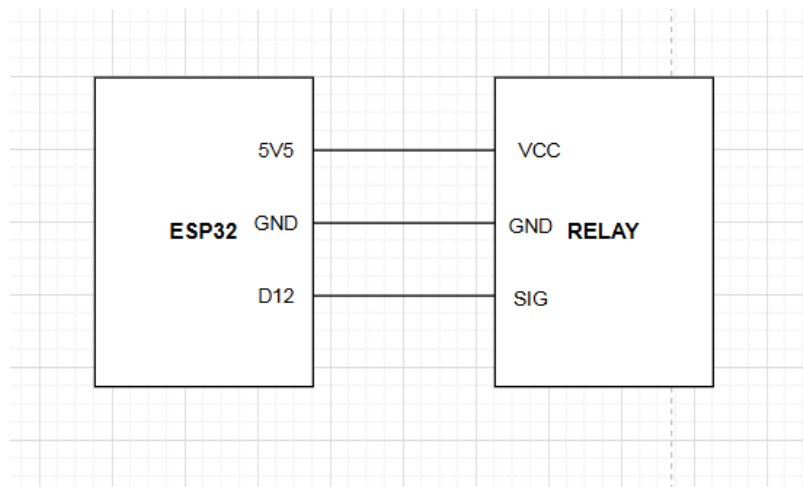
- Điện áp hoạt động: 5V DC.
- Dòng điện hoạt động max: 80mA.
- Điện áp kích tải AC max: 250V.
- Điện áp kích tải DC max: 30V.
- Dòng điện kích tải max: 10A.
- Công suất tải Max: 2500W.
- Nhiệt độ hoạt động: -20°C – 125°C.

- Chọn mức tín hiệu: kích 0 hoặc 1 qua jumper.

Ứng dụng:

- Điều khiển quạt thông gió.
- Điều khiển đèn sưởi/đèn nhiệt.
- Bật/tắt đèn chiếu sáng.
- Điều khiển máy bơm nước, van điện từ.
- Đóng/ngắt các thiết bị gia dụng trong mô hình nhà thông minh.

Sơ đồ khối nối Module Relay 5V – 10A với ESP32:



Hình 2.6. Sơ đồ khối của Module Relay 5V - 10A với ESP32.

2.2.1.4. Quạt tản nhiệt – 12V

Quạt tản nhiệt 12V trong hệ thống được dùng để làm mát và lưu thông không khí giúp giữ nhiệt độ môi trường ổn định.



Hình 2.7. Quạt tản nhiệt 12V.

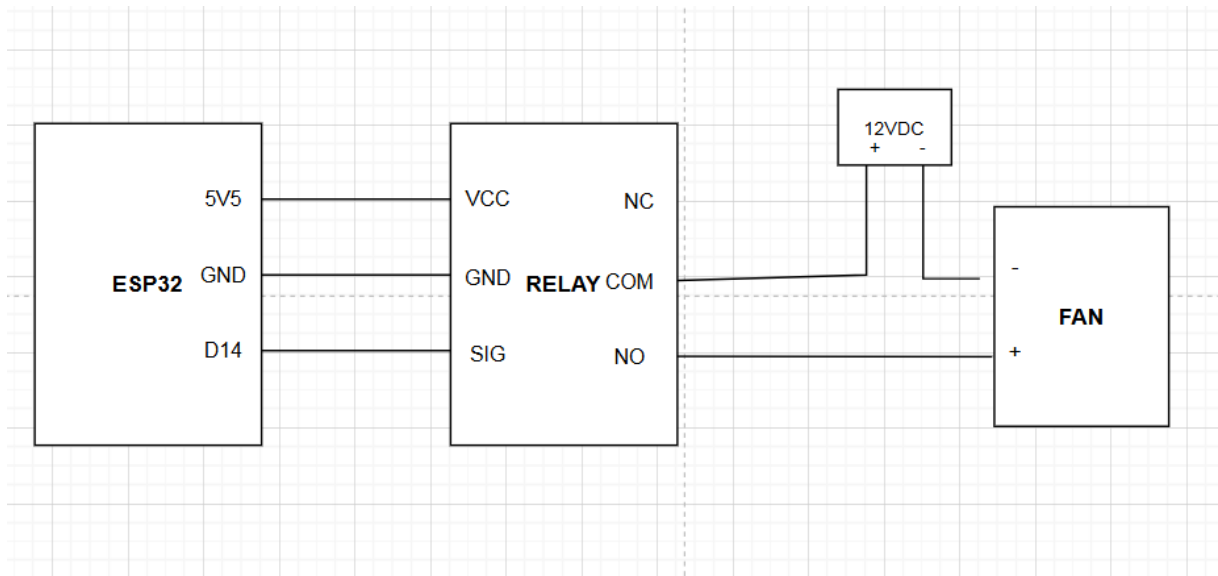
Thông số kỹ thuật:

- Điện áp hoạt động: 12V DC.
- Kích thước: 120x120x25mm.
- Tốc độ quay: ~2200 – 2800 RPM.
- Lưu lượng gió: ~70 – 90 CFM.
- Độ ồn: < 32dB.
- Dòng điện tiêu thụ: ~0.2 – 0.3A.
- Loại kết nối: 2 dây (đỏ – dương, đen – âm).
- Chất liệu: Nhựa ABS cao cấp.
- Tuổi thọ: Khoảng 50.000 giờ hoạt động liên tục.

Ứng dụng:

- Làm mát và giảm nhiệt cho thiết bị điện tử.
- Thông gió và lưu thông không khí trong các hệ thống IoT.
- Giải nhiệt cho tủ điện, máy móc nhỏ.
- Duy trì nhiệt độ ổn định trong mô hình chuồng nuôi thông minh.

Sơ đồ khối điều khiển Quạt tản nhiệt 12V với ESP32 thông qua Module Relay 5V – 10A:



Hình 2.8. Sơ đồ khối điều khiển quạt tản nhiệt với ESP32 thông qua Module Relay

2.2.1.5. Đèn LED dây dán 12V (Đèn chiếu sáng)

Đèn dây LED 12V trong hệ thống được dùng để chiếu sáng mô hình, giúp quan sát môi trường rõ ràng hơn.



Hình 2.9. Đèn LED dây 12V.

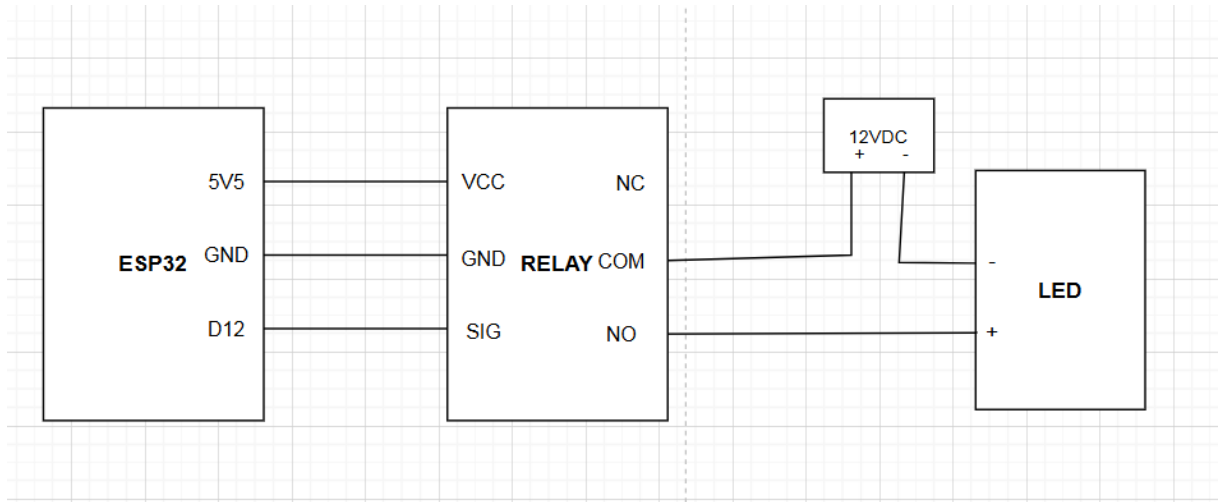
Thông số kỹ thuật:

- Điện áp hoạt động: 12V DC.
- Công suất: 4–14W/m (tùy loại 2835, 5050, 5630...).
- Mật độ LED: 30–120 LED/m.
- Màu ánh sáng: Trắng, vàng, RGB.
- Độ sáng: 300–1000 lm/m.
- Khả năng cắt: Cắt theo đoạn 3–5 cm.
- Chất liệu: Mạch dẻo FPC, lớp phủ chống bụi/ẩm (tùy loại).
- Nhiệt sinh ra: Thấp.
- Tuổi thọ: 20.000 – 50.000 giờ.

Ứng dụng:

- Chiếu sáng mô hình, hộp trưng bày và không gian nhỏ.
- Trang trí nội thất, viền tủ, kệ, bàn làm việc.
- Chiếu sáng trong các hệ thống IoT và thiết bị nhúng.
- Tạo ánh sáng phụ trong tủ điện hoặc khu vực làm việc.

Sơ đồ khối điều khiển Đèn LED dây 12V với ESP32 thông qua Module Relay 5V – 10A:



Hình 2.10. Sơ đồ khối điều khiển Đèn LED dây 12V với ESP32 thông qua Module Relay 5V – 10A.

2.2.1.6. Đèn sợi đốt loại C35 (Đèn sưởi)

Đèn sợi đốt C35 trong hệ thống được dùng như nguồn nhiệt sưởi, giúp giữ ấm môi trường khi nhiệt độ xuống thấp.



Hình 2.11. Đèn sợi đốt C35 - 40W.

Thông số kỹ thuật:

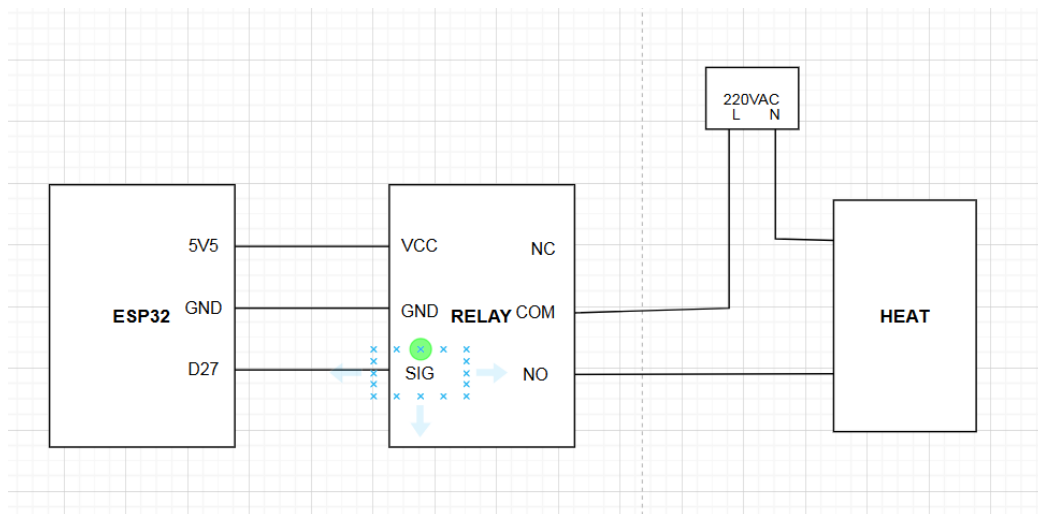
- Điện áp hoạt động: 220V AC.

- Công suất: 25W – 40W (tùy loại).
- Kiểu dáng: Dạng nến (Candle C35).
- Chất liệu bóng: Thủy tinh.
- Nguồn sáng: Dây tóc vonfram.
- Màu ánh sáng: Vàng ấm (Warm White).
- Đui đèn: E14 hoặc E27.
- Nhiệt tỏa ra: Trung bình – phù hợp sưởi nhẹ.
- Tuổi thọ: ~1.000 giờ.

Ứng dụng:

- Dùng trong chiếu sáng và trang trí nhờ ánh sáng vàng ấm.
- Tạo nguồn nhiệt nhẹ cho các mô hình hoặc không gian nhỏ.
- Sử dụng trong đèn ngủ, đèn nến, đèn trang trí nội thất.
- Ứng dụng trong một số mô hình IoT cần ánh sáng kết hợp tỏa nhiệt.

Sơ đồ khối điều khiển Đèn sợi đốt C35 - 40W với ESP32 thông qua Module Relay 5V – 10A:



Hình 2.12. Sơ đồ khối điều khiển Đèn sợi đốt C35 - 40W với ESP32 thông qua Module Relay 5V – 10A.

2.2.1.7. OLED SSD1306 0.96 inch

Màn hình OLED SSD1306 0.96 inch trong hệ thống được dùng để hiển thị nhiệt độ, độ ẩm, trạng thái kết nối Blynk và trạng thái hoạt động của hệ thống.



Hình 2.13. OLED SSD1306 0.96 inch.

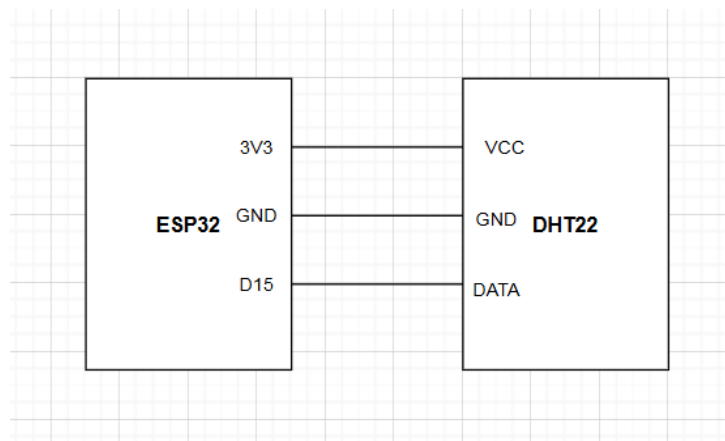
Thông số kỹ thuật:

- Điện áp sử dụng: 2.2~5.5VDC.
- Công suất tiêu thụ: 0.04w.
- Góc hiển thị: lớn hơn 160 độ.
- Số điểm hiển thị: 128x64 điểm.
- Độ rộng màn hình: 0.96 inch.
- Màu hiển thị: Trắng / Xanh Dương.
- Giao tiếp: I2C.
- Driver: SSD1306.

Ứng dụng:

- Hiển thị dữ liệu cảm biến trong các mô hình IoT.
- Hiển thị trạng thái hệ thống, cảnh báo và thông tin hoạt động.
- Dùng trong thiết bị nhúng, đồng hồ, máy đo mini.

Sơ đồ khối nối OLED SSD1306 0.96 inch với ESP32:



Hình 2.14. Sơ đồ khối nối OLED SSD1306 0.96 inch với ESP32.

2.2.1.8. Remote hồng ngoại và Remote IR1838

Remote hồng ngoại và module thu IR1838 trong hệ thống được dùng để điều khiển thiết bị thủ công từ xa, như bật/tắt quạt, đèn LED, sưởi và đổi trạng thái hoạt động của hệ thống.

Remote gửi mã lệnh hồng ngoại, IR1838 thu và chuyển mã này cho ESP32 xử lý, giúp hệ thống phản hồi nhanh và tiện lợi mà không cần dùng Blynk.



Hình 2.15. Remote hồng ngoại và Remote IR1838.

Thông số kỹ thuật:

Remote hồng ngoại:

- Tần số phát: 38kHz (chuẩn NEC).
- Số phím điều khiển: 21 phím.
- Khoảng cách điều khiển: 5 – 10 m.

- Nguồn cấp: Pin CR2025/CR2032.
- Góc phát sóng: $\sim 45\text{--}60^\circ$.
- Tín hiệu xuất: Mã lệnh dạng mã hóa NEC.

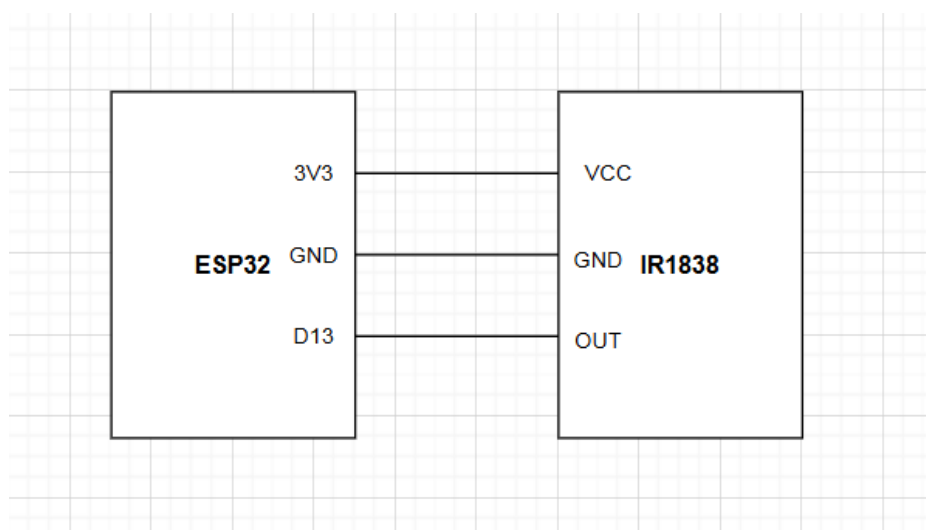
Remote IR1838:

- Điện áp hoạt động: 3.3V – 5VDC.
- Tần số nhận: 38 kHz.
- Giao tiếp: Digital Output (1 chân tín hiệu).
- Khoảng cách nhận: 5 – 15 m.
- Chân kết nối: VCC – GND – OUT.
- Tích hợp lọc nhiễu ánh sáng môi trường, độ nhạy cao.
- Tương thích: ESP32, Arduino, STM32.

Ứng dụng:

- Điều khiển từ xa cho các thiết bị trong hệ thống IoT (quạt, đèn, máy bơm...).
- Chuyển chế độ hoạt động của hệ thống như Auto/Manual.
- Điều hướng giao diện hiển thị trên màn hình OLED (menu, trang thông tin).
- Ứng dụng trong các thiết bị nhúng: robot điều khiển từ xa, xe dò đường.
- Giải pháp điều khiển dự phòng khi hệ thống mất kết nối Internet (ESP32 + Blynk).
- Học tập – nghiên cứu về giao thức truyền IR và mã hóa NEC.

Sơ đồ nối Remote IR1838 với ESP32:



Hình 2.16. Sơ đồ nối Remote IR1838 với ESP32.

2.2.2. Thiết kế phần mềm

2.2.2.1. Cài đặt môi trường Arduino IDE

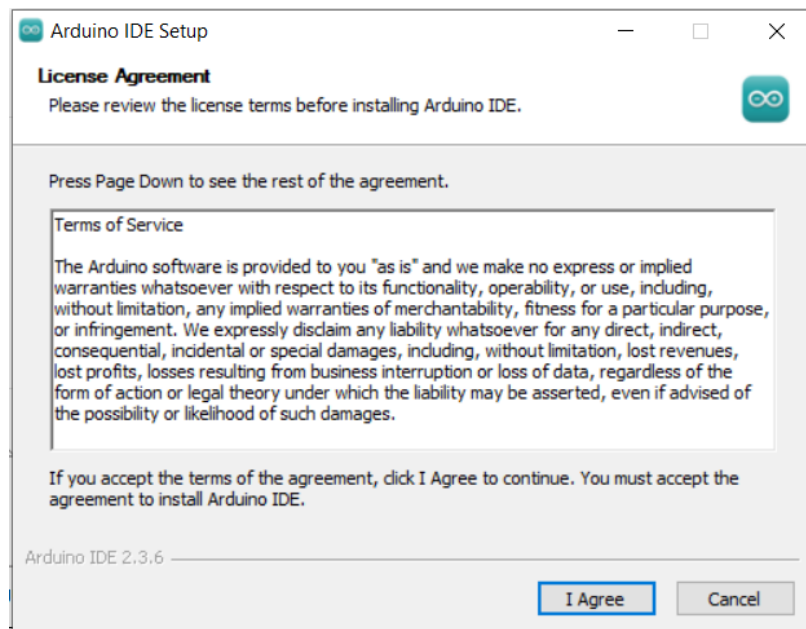
Các bước cài đặt môi trường Arduino IDE:

Bước 1: Truy cập link <https://www.arduino.cc/en/software/> sau đó chọn hệ điều hành bạn sử dụng và tải ứng dụng về.



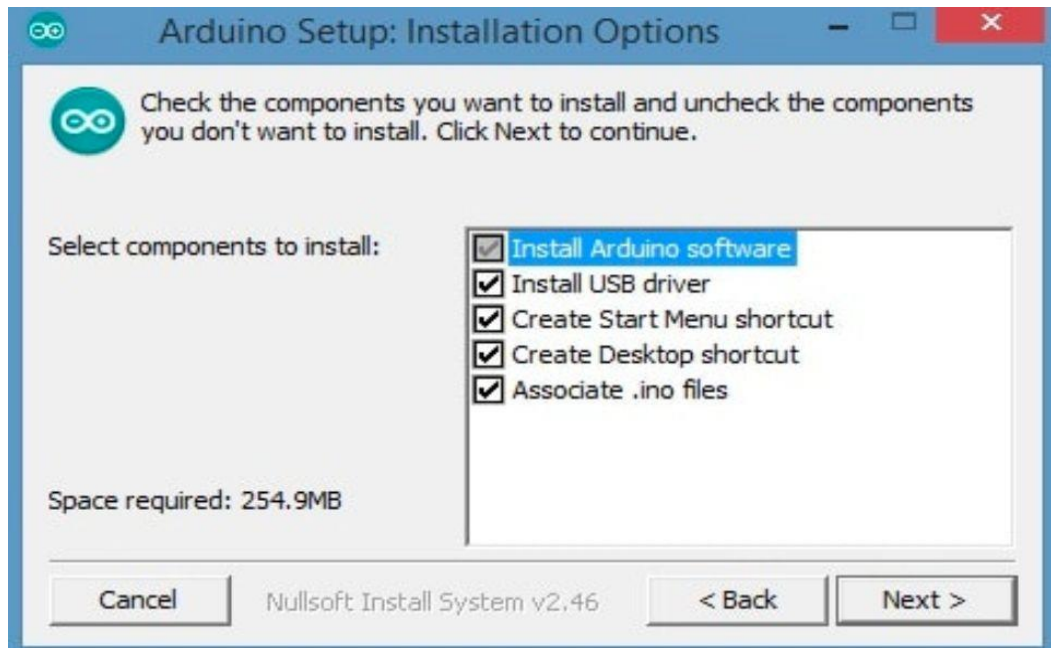
Hình 2.17. Tải ứng dụng Arduino IDE.

Bước 2: Mở phần mềm Arduino bạn vừa tải lên ấn I Agree với thỏa thuận cấp phép:



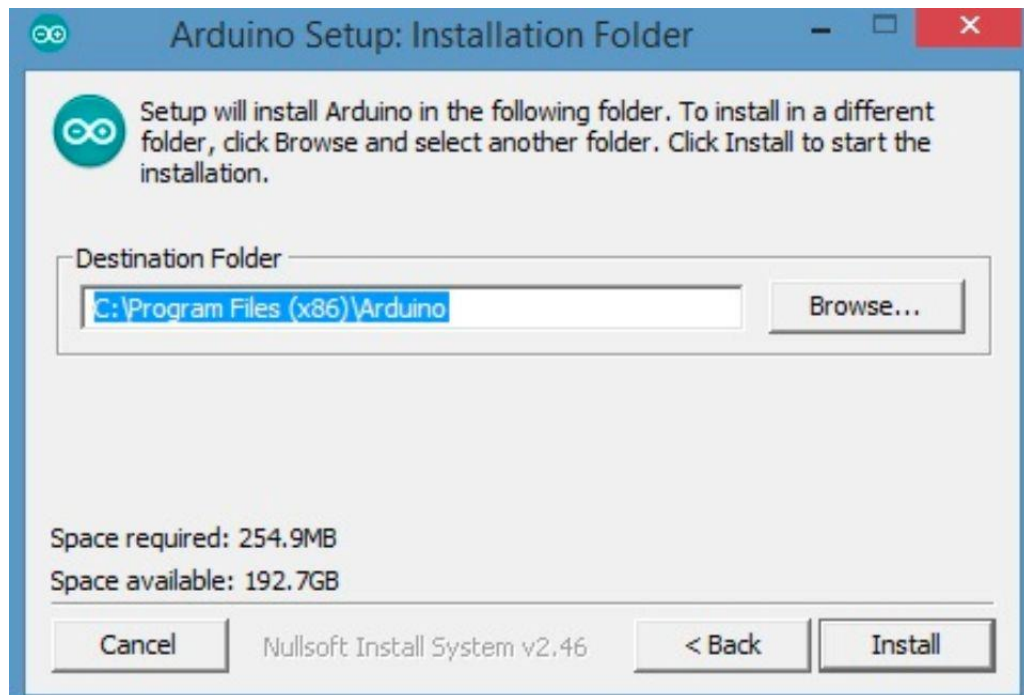
Hình 2.18. Giao diện đồng ý tải Arduino IDE.

Bước 3: Quyết định thành phần nào sẽ cài đặt, sau đó nhấp vào tiếp theo (Next).



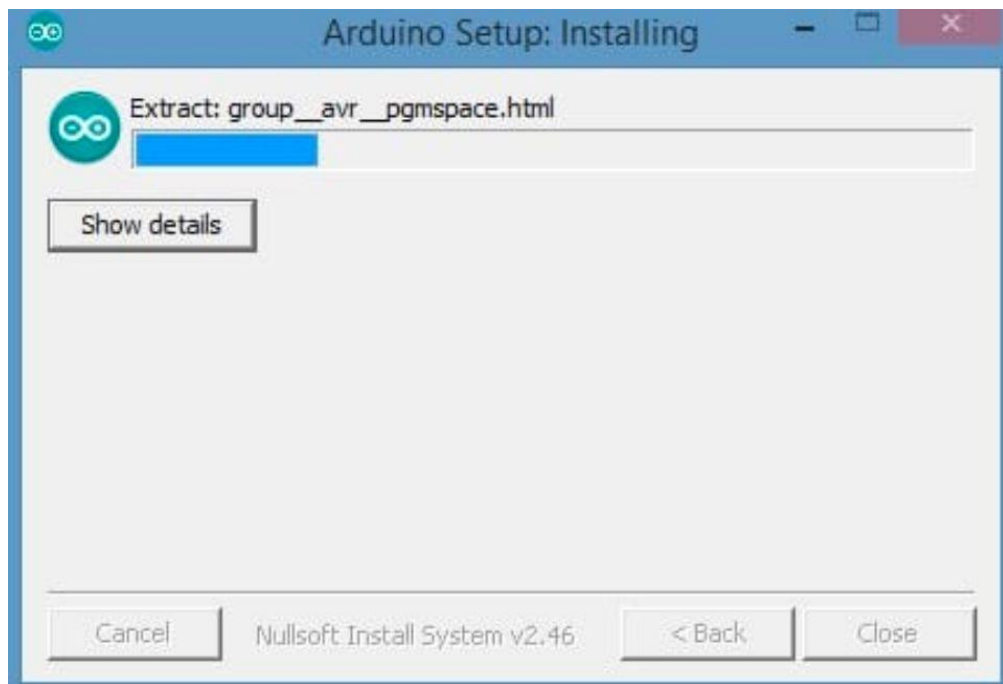
Hình 2.19. Chọn thành phần tải.

Bước 4: Chọn thư mục để cài đặt chương trình, sau đó nhấp vào cài đặt (Install).



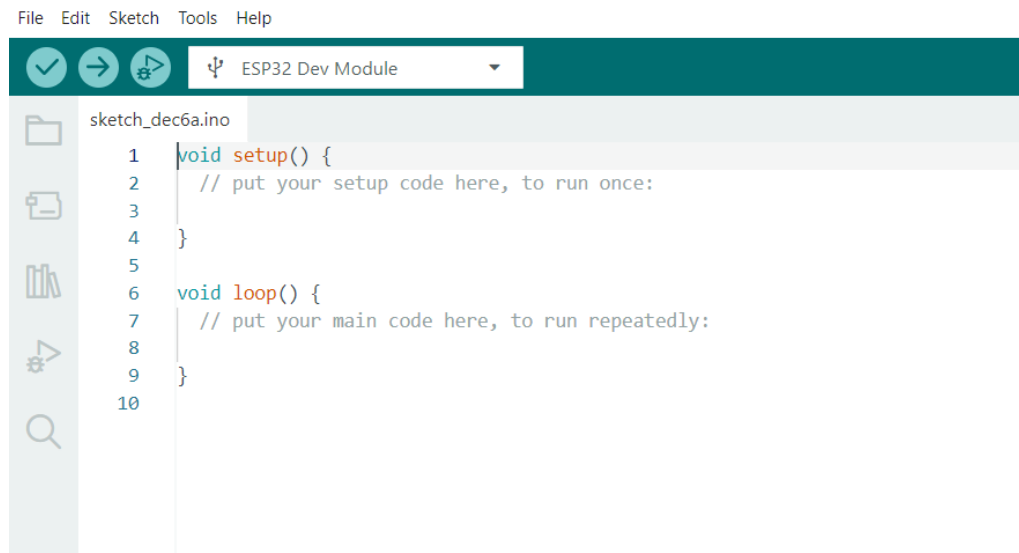
Hình 2.20. Chọn nơi tải về.

Bước 5: Đóng chương trình cài xong rồi nhấn “Close”.



Hình 2.21. Đóng ứng dụng khi tải xong.

Bước 6: Sau khi hoàn tất, bạn có thể tìm thấy lõi tắt Arduino trên màn hình và nhấp vào nó IDE sẽ mở ra và hiện ra trình soạn thảo.



Hình 2.22. Trình soạn thảo Arduino IDE.

2.2.2.2. Thiết lập môi trường ESP32 trong Arduino IDE

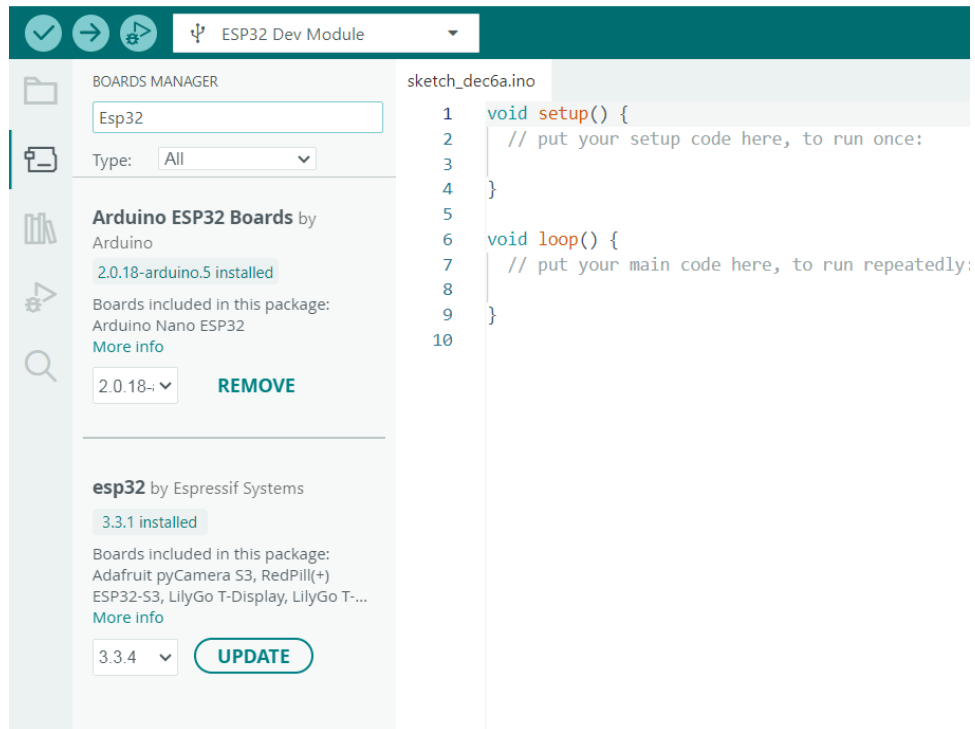
Hướng dẫn thiết lập môi trường ESP32 trong Arduino IDE:

Bước 1: Thêm ESP32 Board vào Arduino IDE.

- Mở Arduino IDE.
- Vào File → Preferences.
- Trong ô Additional Board Manager URLs, thêm đường link sau:
https://dl.espressif.com/dl/package_esp32_index.json .
- Nhấn OK để lưu lại.

Bước 2: Cài đặt ESP32 Board.

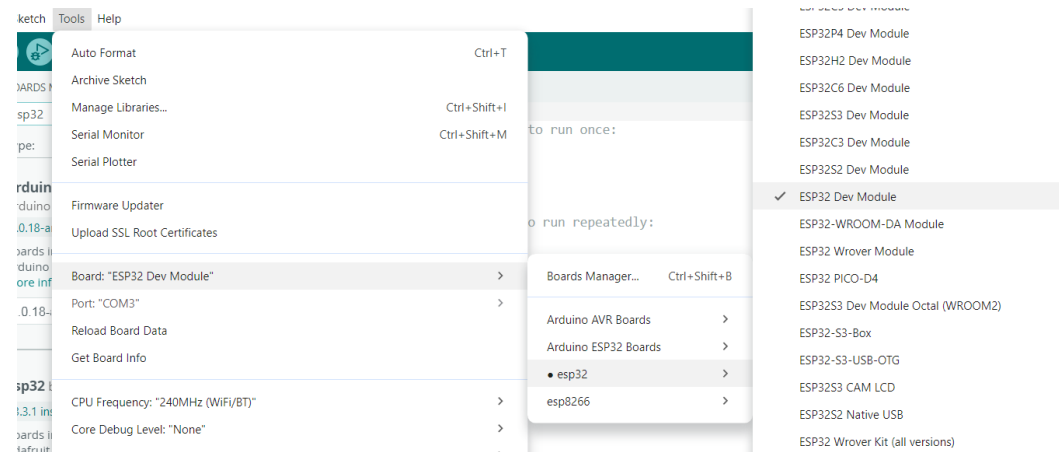
- Vào Tools → Board → Board Manager.
- Trong ô tìm kiếm gõ “esp32”.
- Chọn esp32 by Espressif Systems, nhấn Install.
- Chờ quá trình cài đặt hoàn tất.



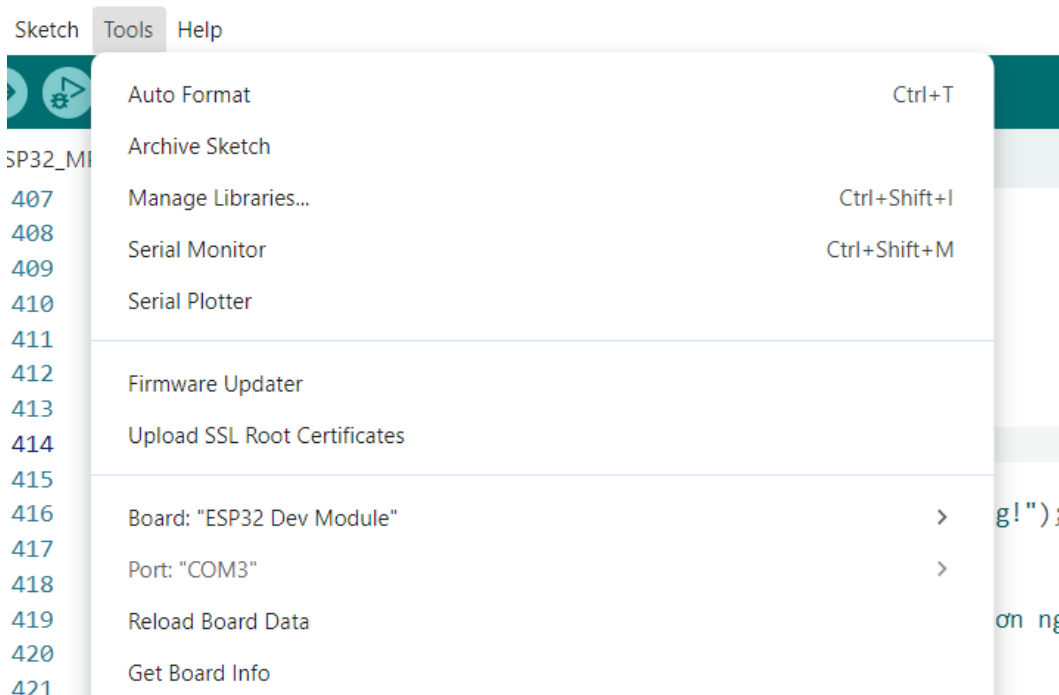
Hình 2.23. Cài đặt ESP32 Broad.

Bước 3: Chọn Board ESP32 và cổng COM.

- Vào Tools → Board → ESP32 Arduino → ESP32 Dev Module (hoặc board tương ứng với model sử dụng).
- Kết nối ESP32 với máy tính qua cáp USB.
- Vào Tools → Port → chọn cổng COM tương ứng với ESP32.



Hình 2.24. Chọn Board ESP32.



Hình 2.25. Chọn cổng COM.

2.2.3. Thiết lập ứng dụng trên app Blynk

Hướng dẫn thiết lập trên ứng dụng Blynk

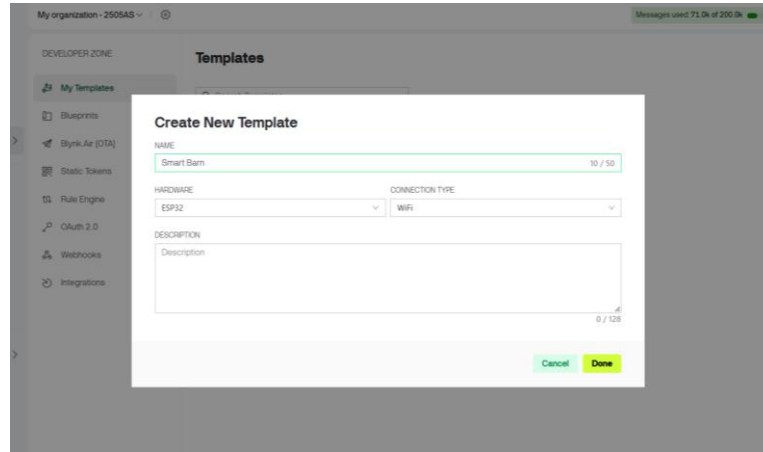
Bước 1: Tạo tài khoản Blynk IoT.

- Mở trình duyệt, truy cập: <https://blynk.io/>.
- Nhấn để đăng ký tài khoản mới hoặc đăng nhập nếu đã có tài khoản.

Bước 2: Tạo Template

- Trong Developer Zone → Nhấn My Template → + New Project.
- Điền thông tin:

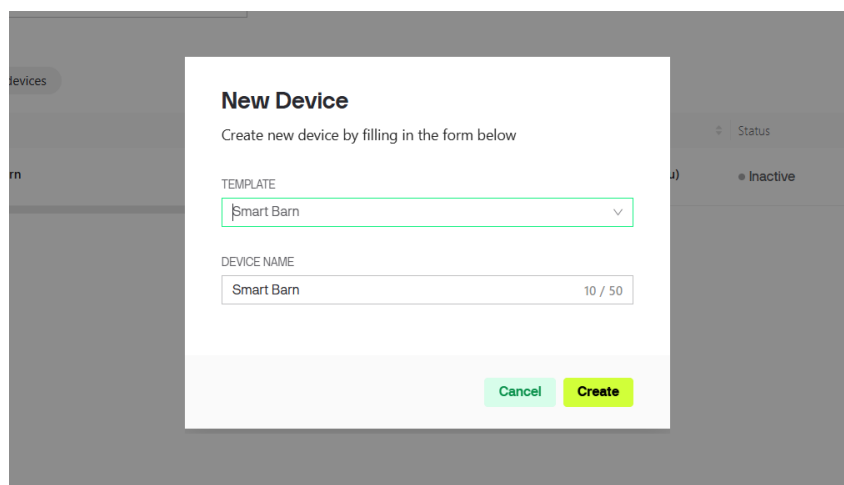
- Template Name: Smart Barn.
- Hardware: ESP32 Dev Board.
- Connection Type: Wifi.
- Nhấn Done để tạo Template.



Hình 2.26. Tạo Template cho giao diện Blynk.

Bước 3: Tạo Device

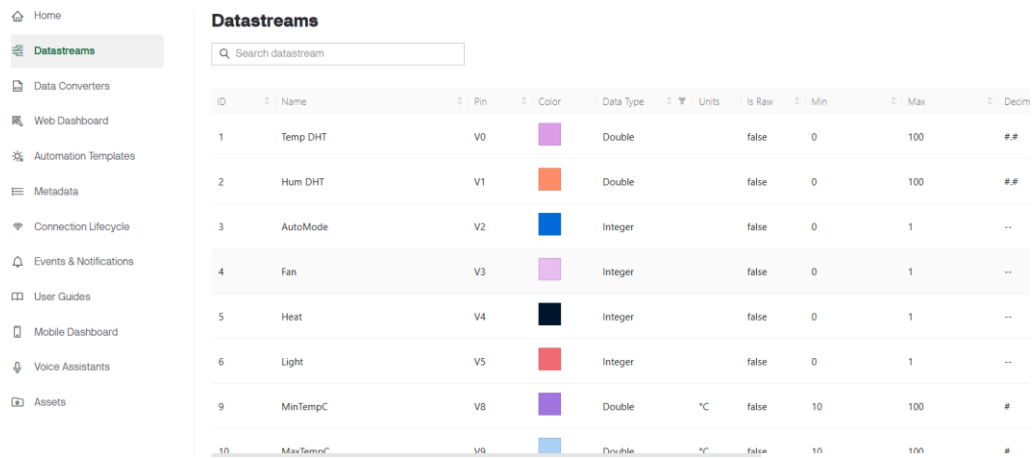
- Vào Devices → + New Device.
- Chọn Template bạn vừa tạo: Smart Barn.
- Điền tên Device: Smart Barn.
- Nhấn Create để tạo Device.
- Blynk sẽ cung cấp Auth Token → Lưu lại để ESP32 kết nối với Blynk.
(Auth Token là duy nhất cho từng Device, dùng trong code ESP32 để gửi và nhận dữ liệu).



Hình 2.27. Tạo Device cho giao diện Blynk.

Bước 4: Thêm Data Stream (Virtual Pin) vào Template.

- Trong Developer Zone → Nhấn My Template Smart Barn → Datastreams → Edit → New Datastreams để Datastreams cần thiết cho dự án.



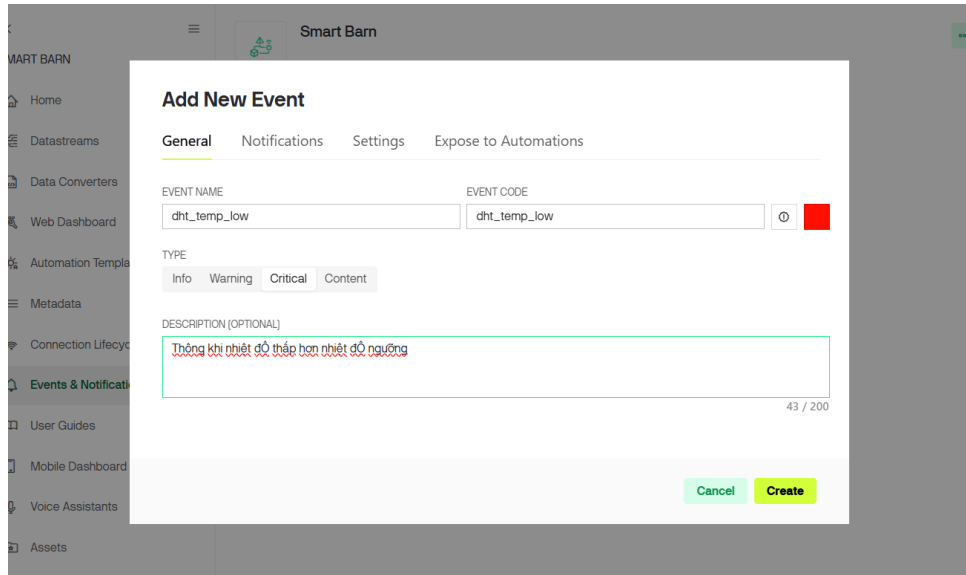
The screenshot shows the Blynk Datastreams interface. On the left is a sidebar with navigation options: Home, Datastreams (selected), Data Converters, Web Dashboard, Automation Templates, Metadata, Connection Lifecycle, Events & Notifications, User Guides, Mobile Dashboard, Voice Assistants, and Assets. The main area is titled 'Datastreams' and contains a search bar and a table of virtual pins.

ID	Name	Pin	Color	Data Type	Units	Is Raw	Min	Max	Decim
1	Temp DHT	V0	Purple	Double		false	0	100	##
2	Hum DHT	V1	Orange	Double		false	0	100	##
3	AutoMode	V2	Blue	Integer		false	0	1	--
4	Fan	V3	Purple	Integer		false	0	1	--
5	Heat	V4	Dark Blue	Integer		false	0	1	--
6	Light	V5	Red	Integer		false	0	1	--
9	MinTempC	V8	Purple	Double	°C	false	10	100	#
10	MaxTempC	V9	Blue	Double	°C	false	10	100	#

Hình 2.28. Thiết lập Datastream cho giao diện Blynk.

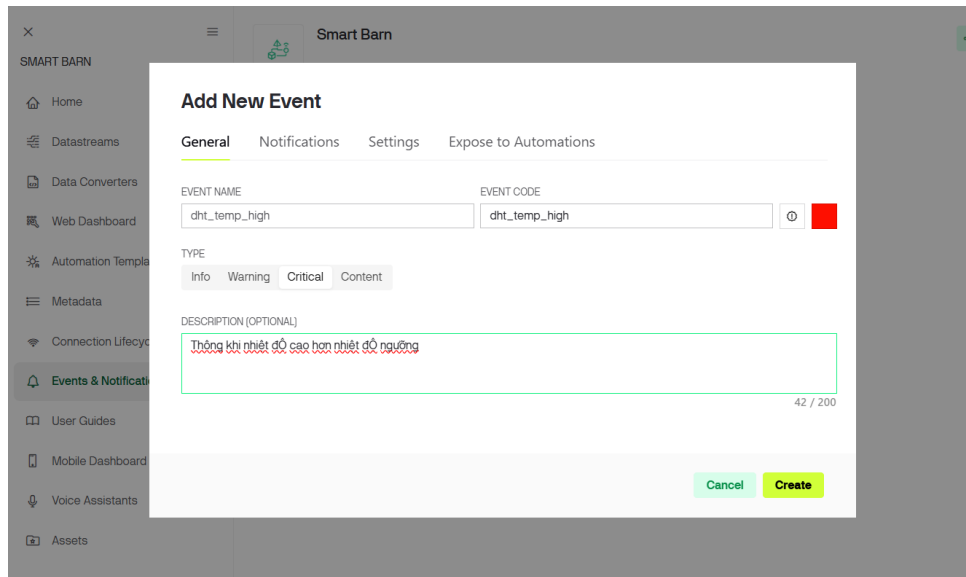
Bước 5: Tạo Event cảnh báo

- Trong Developer Zone → Nhấn My Template Smart Barn → Events & Notifications → Edit → Create Event → Custom.

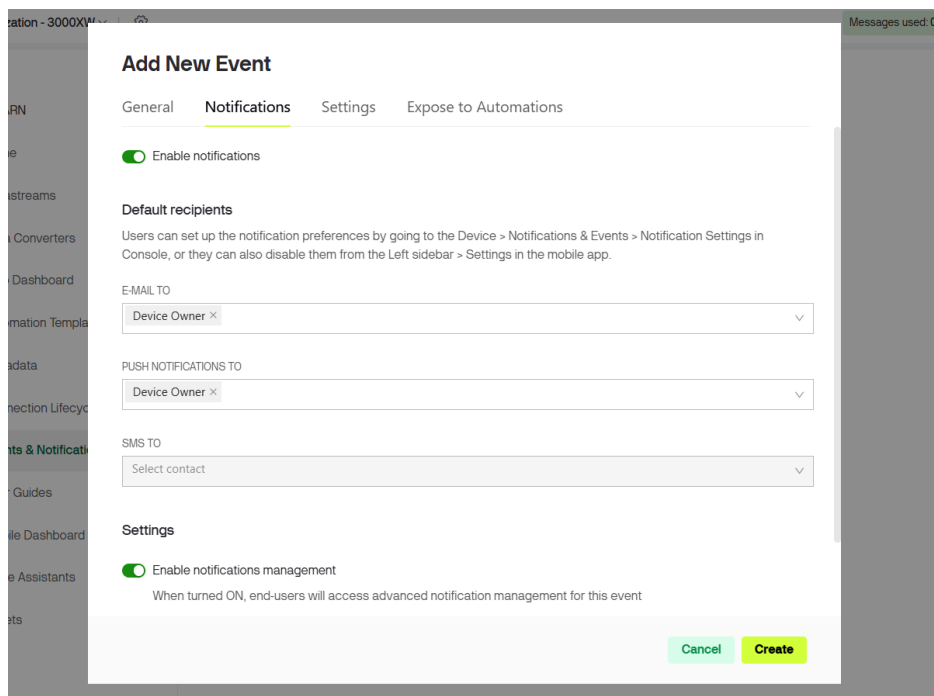


The screenshot shows the 'Add New Event' dialog box in the Blynk interface. The dialog has tabs for 'General', 'Notifications', 'Settings', and 'Expose to Automations'. The 'General' tab is active. It contains fields for 'EVENT NAME' (dht_temp_low), 'EVENT CODE' (dht_temp_low), and a 'TYPE' dropdown set to 'Info'. There is a 'DESCRIPTION (OPTIONAL)' text area with the text 'Thông khi nhiệt độ thấp hơn nhiệt độ ngưỡng'. At the bottom right are 'Cancel' and 'Create' buttons.

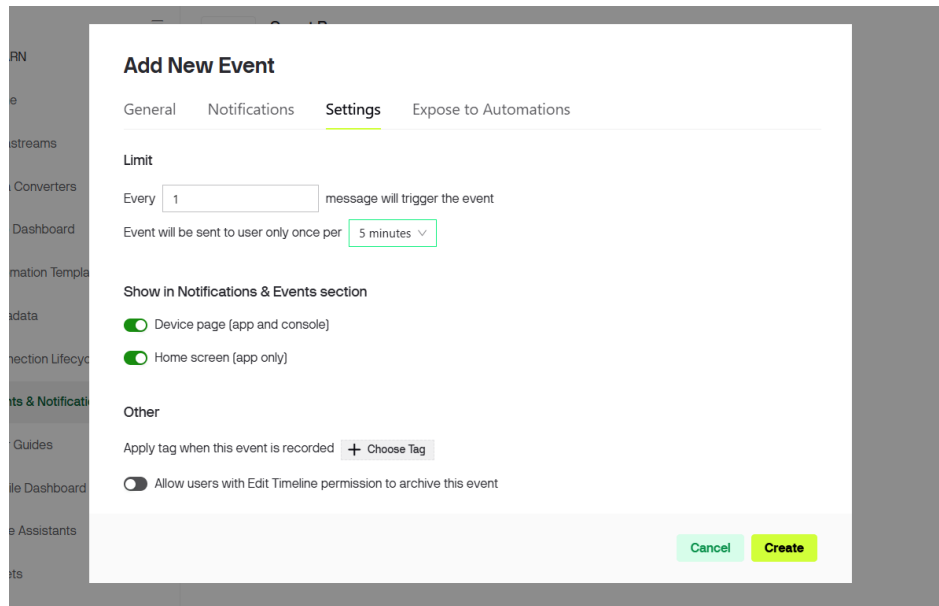
Hình 2.29. Tạo event cho hệ thống Blynk khi nhiệt độ thấp hơn ngưỡng.



Hình 2.30. Tạo event cho hệ thống Blynk khi nhiệt độ cao hơn ngưỡng.

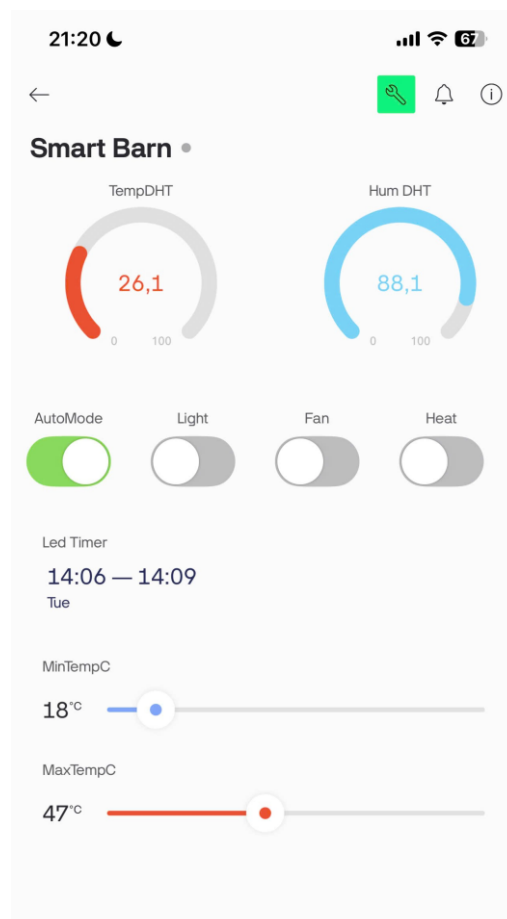


Hình 2.31. Thiết lập thông báo gửi về Gmail và ứng dụng Blynk.



Hình 2.32. Cài đặt thời gian gửi của hệ thống trên Blynk.

Kết quả giao diện của hệ thống trên ứng dụng Blynk:



Hình 2.33. Giao diện hệ thống trên App Blynk.

2.3. Lập trình hệ thống (Code)

```
#define BLYNK_TEMPLATE_ID "TMPL6oRdpm-VU"
#define BLYNK_TEMPLATE_NAME "Smart Barn"
#define BLYNK_AUTH_TOKEN "DQscV8UB95iA0OcaQUG2L17xX1FJkluo"
#define BLYNK_PRINT Serial

//Thư viện WiFi cho ESP32
#include <WiFi.h>
#include <WiFiClient.h>

//Thư viện Blynk cho ESP32
#include <BlynkSimpleEsp32.h>

//Time
#include <TimeLib.h>
#include <WidgetRTC.h> //Widget đồng bộ thời gian RTC từ Blynk
#include "DHT.h"

//Thư viện giao tiếp I2C và OLED
#include <Wire.h> //I2C
#include <IRremote.hpp>
#include <Adafruit_GFX.h> //Thư viện đồ họa
#include <Adafruit_SSD1306.h> //Điều khiển OLED

//Khai báo timer của Blynk để chạy hàm định kỳ
BlynkTimer timer;

//Tạo đối tượng RTC của Blynk
WidgetRTC rtc;

char ssid[] = "Tro 43 To Huu";
char pass[] = "43tohuutro";

//Cấu hình màn hình OLED
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET -1 // không dùng chân reset riêng (-1)
#define SCREEN_ADDRESS 0x3C

//Tạo đối tượng màn hình OLED
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire,
OLED_RESET);

//Remote hồng ngoại
```

```
#define IR_RECEIVE_PIN 13
//DHT22
#define DHTPIN 15
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
//Thiết bị
#define RELAY_LIGHT 12 //led
#define RELAY_FAN 14 //quạt
#define RELAY_HEAT 27 //suoi

float DHT22_C = 0.0;
float DHT22_H = 0.0;
float minTemp = 22.0;
float maxTemp = 30.0;
bool autoMode = true;
//trạng thái các thiết bị
int fanState = 0;
int heatState = 0;
int lightState = 0;
bool blynkIsConnected = false; //trạng thái kết nối Blynk
// biến cho hẹn giờ tự động
int lightStartSec = 0; // thời điểm bật (tính theo giây trong ngày)
int lightStopSec = 0; // thời điểm tắt
bool lightDays[8]; // mảng ngày (1..7)
bool lightHasSchedule = false; // true nếu đã nhận lịch ít nhất 1 lần

//Khai báo hàm
void Read_Sensors();
void Read_IRremote();
void controlDevice();
void auto_Control();
void manual_Control();
void Send_blynk();
void showOLED();
void tryWiFiConnectIfNeeded();
void tryBlynkConnectIfNeeded();
```

```
void auto_LightSchedule();
void showWarning(float tempAir);

BLYNK_WRITE(V2) { // Auto mode
  autoMode = param.asInt();
}
BLYNK_WRITE(V3) { // Fan manual
  if (autoMode == false) {
    fanState = param.asInt();
    controlDevice();
  }
}
BLYNK_WRITE(V4) { // Heat manual
  if (autoMode == false) {
    heatState = param.asInt();
    controlDevice();
  }
}
BLYNK_WRITE(V5) { // Light manual
  if (autoMode == false) {
    lightState = param.asInt();
    controlDevice();
  }
}
BLYNK_WRITE(V8) { // MinTempC
  minTemp = param.asFloat();
}
BLYNK_WRITE(V9) { // MaxTempC
  maxTemp = param.asFloat();
}
BLYNK_WRITE(V6) {
  TimeInputParam t(param); // tạo đối tượng đọc dữ liệu từ widget Time Input
  // --- In thông tin giờ ---
  Serial.print("Start: ");
  Serial.print(t.getStartHour());
  Serial.print(":");
```

```
Serial.print(t.getStartMinute());
Serial.print(" → Stop: ");
Serial.print(t.getStopHour());
Serial.print(":");
Serial.println(t.getStopMinute());

// --- Kiểm tra xem người dùng có chọn ngày nào không ---
bool hasDaySelected = false;
Serial.print("Days selected (Blynk): ");
for (int i = 1; i <= 7; i++) {
    if (t.isWeekdaySelected(i)) {
        Serial.print(String(i) + " "); // In ngày theo thứ tự Blynk (T2=1..CN=7)
        hasDaySelected = true;
    }
}

// Nếu không chọn ngày nào → coi như chọn cả tuần
if (!hasDaySelected) {
    Serial.println("(None) → Áp dụng cả tuần (CN-T7)");
    for (int i = 1; i <= 7; i++) lightDays[i] = true; //bật tất cả
} else {
    Serial.println();
    // Reset toàn bộ mảng trước khi gán lại
    for (int i = 1; i <= 7; i++) lightDays[i] = false;

    // Map chính xác giữa Blynk và TimeLib:
    lightDays[2] = t.isWeekdaySelected(1); // Thứ 2
    lightDays[3] = t.isWeekdaySelected(2); // Thứ 3
    lightDays[4] = t.isWeekdaySelected(3); // Thứ 4
    lightDays[5] = t.isWeekdaySelected(4); // Thứ 5
    lightDays[6] = t.isWeekdaySelected(5); // Thứ 6
    lightDays[7] = t.isWeekdaySelected(6); // Thứ 7
    lightDays[1] = t.isWeekdaySelected(7); // Chủ nhật

    // In kết quả để kiểm tra
    Serial.print("Days mapped to TimeLib: ");
```

```
for (int i = 1; i <= 7; i++) {
    if (lightDays[i]) Serial.print(String(i) + " ");
}
Serial.println();
}

// --- Lưu thời gian bật/tắt (giờ tính từ 0h00) ---
lightStartSec = t.getStartHour() * 3600 + t.getStartMinute() * 60;
lightStopSec = t.getStopHour() * 3600 + t.getStopMinute() * 60;
lightHasSchedule = true;

Serial.println("Lịch đèn đã lưu thành công!");
Serial.println("-----");
}

// Hàm gửi dữ liệu lên Blynk
void Send_blynk() {
    if (!Blynk.connected()) return;
    //DHT22
    Blynk.virtualWrite(V0, DHT22_C);
    Blynk.virtualWrite(V1, DHT22_H);
    //Thiết bị
    Blynk.virtualWrite(V2, autoMode);
    Blynk.virtualWrite(V3, fanState);
    Blynk.virtualWrite(V4, heatState);
    Blynk.virtualWrite(V5, lightState);
}

void setup() {
    Serial.begin(9600);

    Wire.begin(21, 22);
    dht.begin();
    IrReceiver.begin(IR_RECEIVE_PIN, ENABLE_LED_FEEDBACK);

    pinMode(RELAY_FAN, OUTPUT);
    pinMode(RELAY_HEAT, OUTPUT);
    pinMode(RELAY_LIGHT, OUTPUT);
}
```

```
//khởi tạo Oled
if (!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
  Serial.println(F("SSD1306 allocation failed"));
  for (;;)
    ;
}
display.clearDisplay();

//khởi tạo Wifi
WiFi.mode(WIFI_STA);
WiFi.begin(ssid, pass);

Blynk.config(BLYNK_AUTH_TOKEN);
Serial.println("Bắt đầu thử kết nối...");
rtc.begin(); // Bắt đầu đồng bộ thời gian thực từ Blynk RTC
Blynk.syncAll();

//Dùng timer
timer.setInterval(8000L, tryWiFiConnectIfNeeded);
timer.setInterval(9000L, tryBlynkConnectIfNeeded);
timer.setInterval(3000L, Read_Sensors);
timer.setInterval(500L, showOLED);
timer.setInterval(5000L, Send_blynk);
}

void loop() {
  timer.run(); // chạy các timer định kỳ

  if (Blynk.connected()) {
    Blynk.run();
  }
  auto_Control();
  manual_Control();
  Read_IRremote();
  delay(10);
}
```

```
}  
//Đọc cảm biến DHT22  
void Read_Sensors() {  
    float h = dht.readHumidity();  
    float t = dht.readTemperature();  
  
    if (isnan(h) || isnan(t)) {  
        //DHT22_C = DHT22_H = 0.0;  
        return;  
    }  
  
    DHT22_C = t;  
    DHT22_H = h;  
    //  
    showWarning(DHT22_C);  
}  
//Điều khiển Remote hồng ngoại  
void Read_IRremote() {  
    if (IrReceiver.decode()) {  
        // Bỏ qua tín hiệu lặp khi nhấn giữ  
        if (IrReceiver.decodedIRData.flags & IRDATA_FLAGS_IS_REPEAT) {  
            IrReceiver.resume(); //chuẩn bị khung nhận tiếp theo  
            return; // không xử lý lần này  
        }  
  
        Serial.println(IrReceiver.decodedIRData.command, HEX);  
        if (IrReceiver.decodedIRData.command == 0x1C) { // nút OK : Đk chế độ  
            autoMode = !autoMode;  
        }  
        if (!autoMode) {  
            if (IrReceiver.decodedIRData.command == 0x45) { // nút 1 : ĐK quạt  
                fanState = !fanState;  
            }  
            if (IrReceiver.decodedIRData.command == 0x46) { // nút 2 : ĐK đèn  
                heatState = !heatState;  
            }  
        }  
    }  
}
```

```
    if (IrReceiver.decodedIRData.command == 0x47) { // nút 3 : ĐK led
        lightState = !lightState;
    }
    controlDevice();
}
IrReceiver.resume(); //chuẩn bị nhận gói IR tiếp theo
}
}
//Hàm điều khiển thiết bị
void controlDevice() {
    digitalWrite(RELAY_FAN, fanState);
    digitalWrite(RELAY_HEAT, heatState);
    digitalWrite(RELAY_LIGHT, lightState);

    if (Blynk.connected()) {
        Blynk.virtualWrite(V3, fanState);
        Blynk.virtualWrite(V4, heatState);
        Blynk.virtualWrite(V5, lightState);
    }
}
//Chế độ tự động
void auto_Control() {
    if (autoMode) {
        if (DHT22_C < minTemp) {
            heatState = 1;
        } //
        else {
            heatState = 0;
        }
        if (DHT22_C > maxTemp) {
            fanState = 1;
        } //
        else {
            fanState = 0;
        }
    }
}
```

```
    auto_LightSchedule();
    controlDevice();
}
}
//Chế độ thủ công
void manual_Control() {
    Read_IRremote();
}
//Hàm hiển thị Oled
void showOLED() {
    display.clearDisplay();
    display.setTextSize(2);
    display.setTextColor(WHITE);

    if (autoMode) {
        display.setCursor(40, 0);
        display.println("AUTO");
    } //
    else {
        display.setCursor(30, 0);
        display.println("MANUAL");
    }

    display.setTextSize(1);
    display.println(" ");

    // DHT22
    // --- Nhiệt độ + ngưỡng ---
    display.print("T: ");
    display.print(DHT22_C, 1);
    display.print("C (");
    display.print(minTemp, 0);
    display.print(",");
    display.print(maxTemp, 0);
    display.println(")");
    display.println(" ");
```

```
// --- Độ ẩm ---
display.print("H: ");
display.print(DHT22_H, 1);
display.println("%");
display.println(" ");

//Trạng thái kết nối
if (blynkIsConnected == true) {
  display.println("  Ket Noi Blynk");
} //
else {
  display.println("  Mat Ket Noi Blynk");
}
display.display();
}

//Hàm thử kết nối Wifi
void tryWiFiConnectIfNeeded() {
  if (WiFi.status() != WL_CONNECTED) { //nếu chưa kết nối Wifi
    Serial.println("Mất WiFi -> đang thử kết nối lại...");
    WiFi.disconnect(); //Ngắt kết nối cũ nếu có
    WiFi.begin(ssid, pass); //Thử kết nối lại
  } else {
    Serial.println("WiFi OK: " + WiFi.localIP().toString());
  }
}

// Hàm thử kết nối Blynk
void tryBlynkConnectIfNeeded() {
  //kết nối khi có wifi và blynk bị ngắt
  if (WiFi.status() == WL_CONNECTED && !Blynk.connected()) {
    Serial.println("Thử kết nối Blynk...");
    if (Blynk.connect(5000)) { // thử trong 5s
      Serial.println("Kết nối Blynk thành công!");
      blynkIsConnected = true;
      rtc.begin(); //khởi động lại RTC
      //Blynk.syncAll();
    }
  }
}
```

```
Blynk.syncVirtual(V8, V9, V10, V11); // đồng bộ trạng thái
} else {
    Serial.println("Blynk thất bại, sẽ thử lại sau.");
    blynkIsConnected = false;
}
}
}
// Hẹn giờ tự động
void auto_LightSchedule() {
    if (!lightHasSchedule) return;

    // --- Lấy thời gian hiện tại ---
    int nowSec = hour() * 3600 + minute() * 60 + second();
    int today = weekday(); // CN=1, T2=2,...,T7=7

    // --- Kiểm tra hôm nay có trong lịch không ---
    bool dayActive = lightDays[today];
    bool allDaysActive = true;
    for (int i = 1; i <= 7; i++) {
        if (!lightDays[i]) {
            allDaysActive = false;
            break;
        }
    }

    if (!dayActive && !allDaysActive) {
        lightState = 0;
        return;
    }

    bool lightActive = false;

    if (lightStartSec < lightStopSec) {
        lightActive = (nowSec >= lightStartSec && nowSec < lightStopSec);
    } else {
        lightActive = (nowSec >= lightStartSec || nowSec < lightStopSec);
    }
}
```

```
}
lightState = lightActive ? 1 : 0;
digitalWrite(RELAY_LIGHT, lightState);

if (Blynk.connected()) Blynk.virtualWrite(V5, lightState);
}
//Hàm even
void showWarning(float tempAir) {
  if (isnan(tempAir)) return; // nếu đọc lỗi thì bỏ qua

  bool danger = false;

  // Nhiệt độ thấp hơn ngưỡng
  if (tempAir < minTemp) {
    danger = true;
    Serial.println("CẢNH BÁO: Nhiệt độ không khí THẤP hơn ngưỡng!");
    if (Blynk.connected()) {
      Blynk.logEvent("dht_temp_low", "Nhiệt độ không khí thấp hơn ngưỡng cài đặt (minTemp)!");
    }
  }
  // Nhiệt độ cao hơn ngưỡng
  else if (tempAir > maxTemp) {
    danger = true;
    Serial.println("CẢNH BÁO: Nhiệt độ không khí CAO hơn ngưỡng!");
    if (Blynk.connected()) {
      Blynk.logEvent("dht_temp_high", "Nhiệt độ không khí cao hơn ngưỡng cài đặt (maxTemp)!");
    }
  }
}
```

Chương 3: KẾT QUẢ THỰC NGHIỆM HỆ THỐNG

3.1. Mô hình thực tế

Ứng dụng đề tài vào hệ thống IoT nuôi gà Ri bình thường:



Hình 3.1. Hệ thống khi chưa hoạt động.

Hệ thống bao gồm ESP32, cảm biến DHT22, màn hình OLED, quạt tản nhiệt, bóng đèn sưởi, bộ nhận tín hiệu điều khiển từ Remote hồng ngoại. Khi hoạt động hệ thống sẽ gửi các thông số nhiệt độ, độ ẩm và trạng thái hoạt động của hệ thống lên cả Blynk và màn hình OLED, nếu vượt ngưỡng thì hệ thống sẽ tự động điều khiển các thiết bị phù hợp (Ở chế độ tự động).



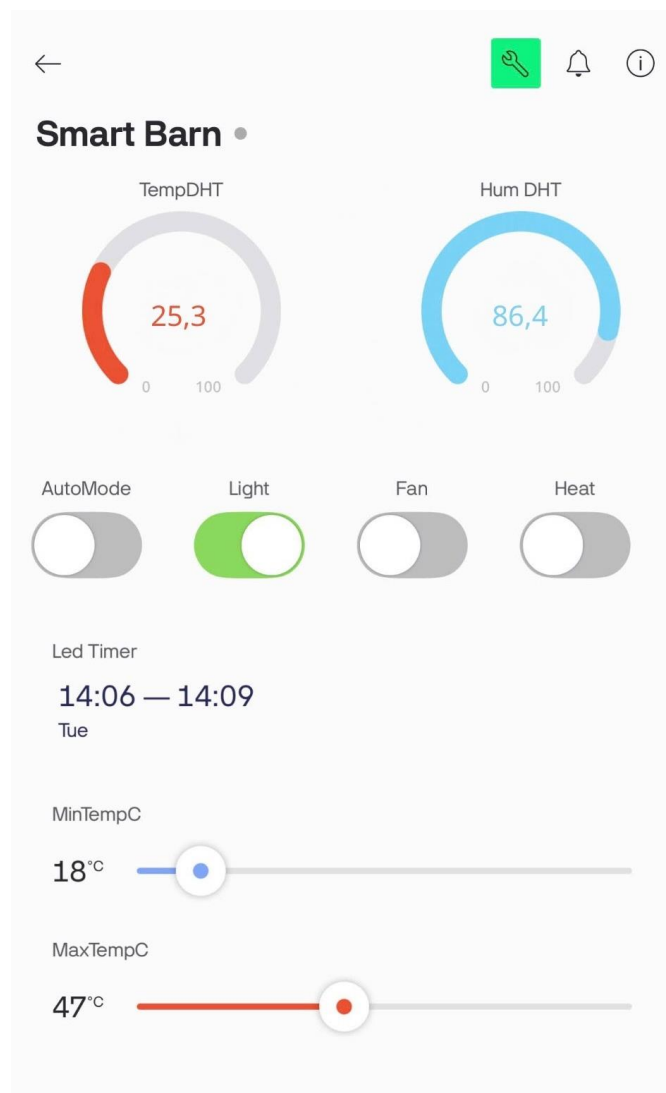
Hình 3.2. Hệ thống khi đang hoạt động.

3.2. Kết quả thực nghiệm

3.2.1. Kết quả đo và hiển thị trên Blynk và màn hình OLED

Kết quả đo và hiển thị trên Blynk:

Kết quả đo và hiển thị trên ứng dụng Blynk cho thấy các thông số trình bày rõ ràng, dễ hiểu qua các Widget, nhiệt độ và độ ẩm được hiển thị qua Gauge giúp người dùng quan sát một cách trực quan theo thời gian thực, trạng thái hệ thống và các thiết bị được hiển thị qua Button và các thanh điều chỉnh ngưỡng. Giao diện trình bày đơn giản giúp người dùng dễ dàng thao tác và sử dụng.



Hình 3.3. Kết quả đo và hiển thị trên Blynk.

Kết quả đo và hiển thị trên màn hình OLED:

Kết quả đo và hiển thị trên màn hình OLED cho thấy các thông số đó là trạng thái hoạt động của hệ thống (MANUAL/AUTO) và các chỉ số nhiệt độ (T), độ ẩm (H) và trạng thái kết nối Blynk của hệ thống.

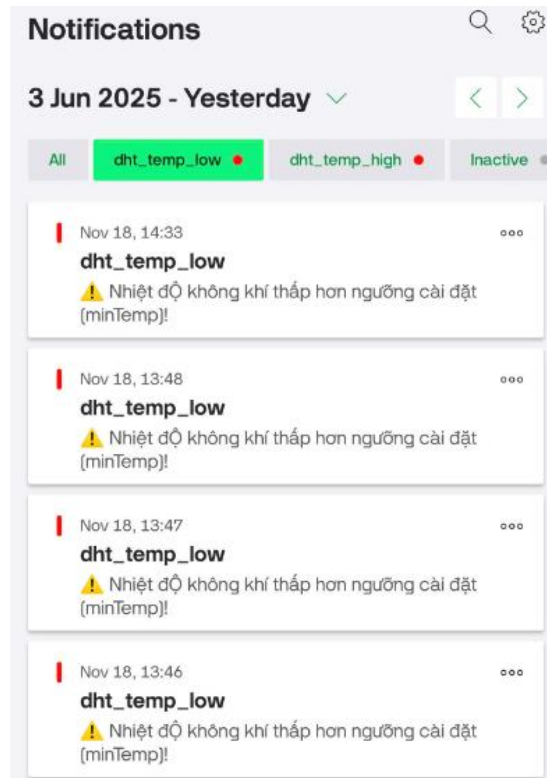


Hình 3.4. Kết quả đo và hiển thị trên màn hình OLED.

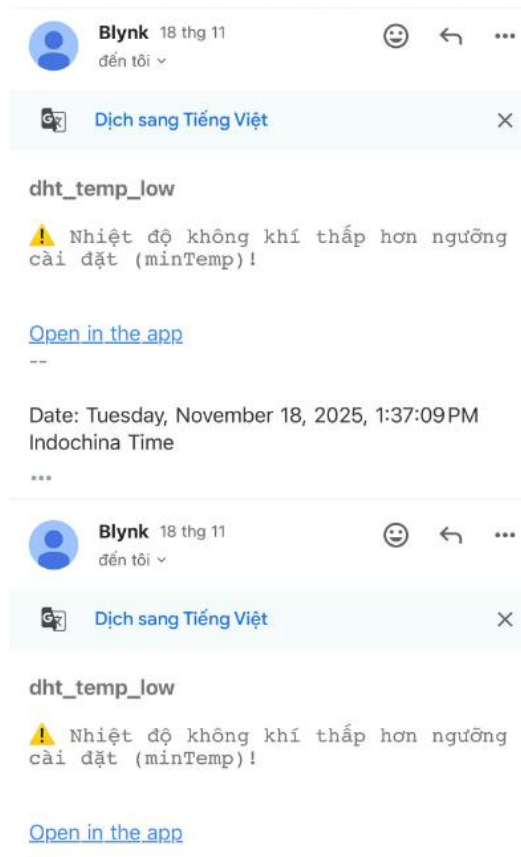
3.2.2. Kết quả cảnh báo khi vượt ngưỡng qua ứng dụng Blynk và Gmail

Kết quả cảnh báo khi nhiệt độ thấp hơn ngưỡng nhiệt độ tối thiểu:

Khi nhiệt độ hệ thống thấp hơn ngưỡng nhiệt độ tối thiểu cài đặt trên Blynk, hệ thống sẽ gửi cảnh báo về ứng dụng Blynk với thông báo “Nhiệt độ không khí thấp hơn ngưỡng cài đặt (minTemp)”. Và Blynk cũng đồng thời gửi cảnh báo về Gmail giúp người dùng theo dõi hệ thống kịp thời.



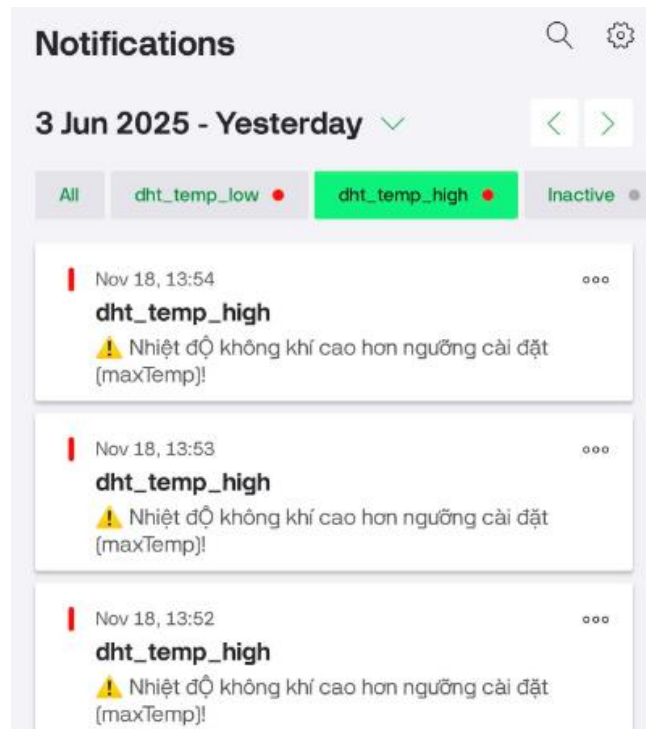
Hình 3.5. Kết quả cảnh báo khi nhiệt độ thấp hơn ngưỡng nhiệt độ tối thiểu qua Blynk



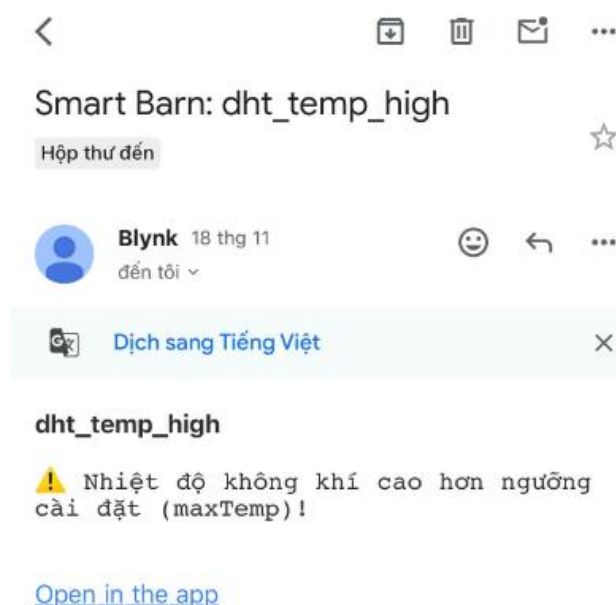
Hình 3.6. Kết quả cảnh báo khi nhiệt độ thấp hơn ngưỡng nhiệt độ tối thiểu qua Gmail

Kết quả cảnh báo khi nhiệt độ cao hơn ngưỡng nhiệt độ tối đa:

Khi nhiệt độ hệ thống cao hơn ngưỡng nhiệt độ tối đa cài đặt trên Blynk, hệ thống sẽ gửi cảnh báo về ứng dụng Blynk với thông báo “Nhiệt độ không khí cao hơn ngưỡng cài đặt (maxTemp)”. Và Blynk cũng đồng thời gửi cảnh báo về Gmail giúp người dùng theo dõi hệ thống kịp thời.



Hình 3.7. Kết quả cảnh báo khi nhiệt độ cao hơn ngưỡng nhiệt độ tối đa qua Blynk.

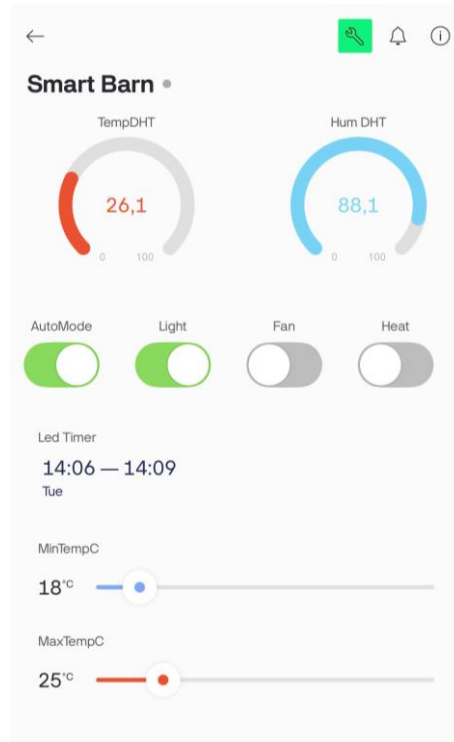


Hình 3.8. Kết quả cảnh báo khi nhiệt độ cao hơn ngưỡng nhiệt độ tối đa qua Gmail

3.2.3. Kết quả chức năng điều khiển các thiết bị qua ứng dụng Blynk

Chế độ tự động (AUTO):

Hệ thống hỗ trợ tự động bật/tắt quạt và đèn sưởi dựa trên ngưỡng nhiệt độ tối đa và tối thiểu. Khi nhiệt độ cao hơn ngưỡng, quạt sẽ tự bật; khi thấp hơn ngưỡng, đèn sưởi sẽ tự kích hoạt.



Hình 3.9. Giao diện hệ thống sau khi điều chỉnh ngưỡng nhiệt độ.

Khi nhiệt độ chuồng vượt quá ngưỡng nhiệt độ tối đa cho phép, hệ thống tự động kích hoạt quạt làm mát và trên ứng dụng Blynk, trạng thái của nút điều khiển Fan được chuyển sang chế độ bật.



Hình 3.10. Kết quả nhiệt độ cao hơn ngưỡng nhiệt độ tối đa - Quạt bật.

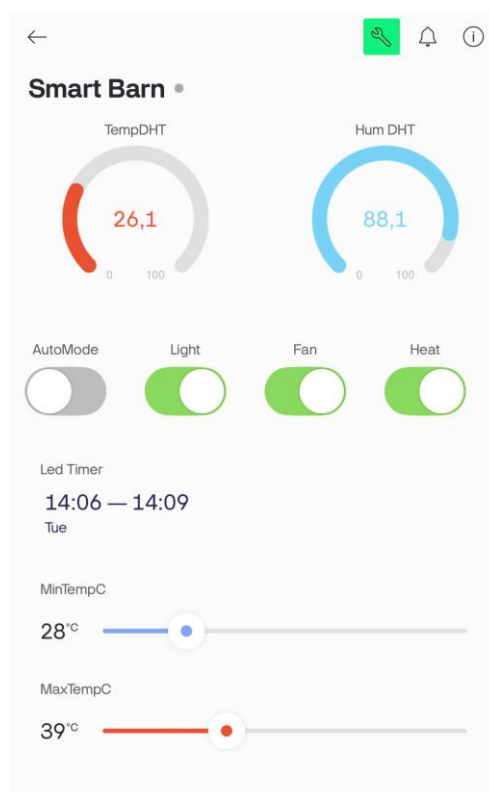
Khi nhiệt độ chuồng vượt quá ngưỡng nhiệt độ tối thiểu cho phép, hệ thống tự động kích hoạt đèn sưởi làm ấm và trên ứng dụng Blynk, trạng thái của nút điều khiển Heat được chuyển sang chế độ bật.



Hình 3.11. Kết quả nhiệt độ cao hơn ngưỡng nhiệt độ tối thiểu - Đèn sưởi bật.

Chế độ thủ công (MANUAL):

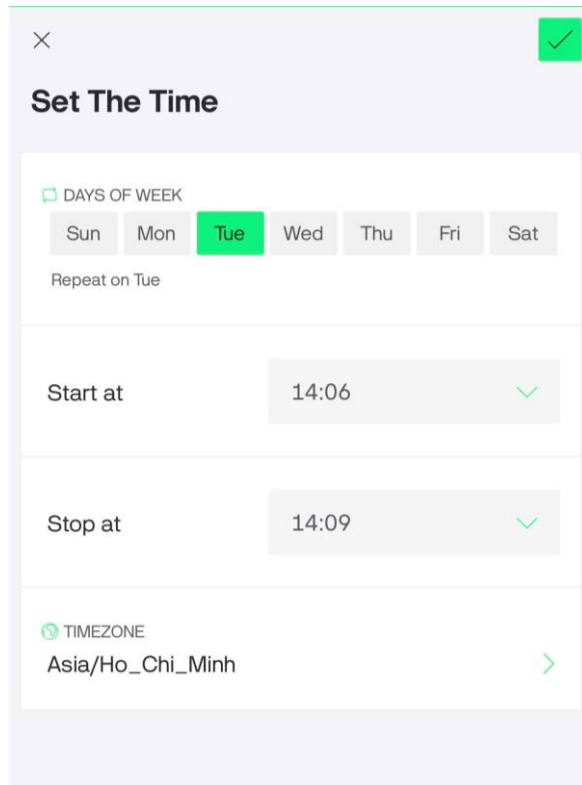
Người dùng cũng có thể chủ động bật/tắt quạt, đèn sưởi và đèn LED chiếu sáng thủ công ngay trên ứng dụng Blynk. Khi nhấn nút tắt AutoMode, hệ thống chuyển từ chế độ tự động sang chế độ thủ công. Lúc này, người dùng có thể điều khiển từng thiết bị bằng các nút bấm riêng, giúp quá trình vận hành linh hoạt và phù hợp theo nhu cầu thực tế.



Hình 3.12. Giao diện hệ thống khi ở chế độ thủ công.

3.2.4. Kết quả hẹn giờ bật – tắt đèn chiếu sáng

Hệ thống cho phép người dùng thiết lập lịch bật - tắt đèn chiếu sáng trực tiếp trên ứng dụng Blynk. Người dùng chọn các ngày trong tuần muốn kích hoạt lịch hẹn, sau đó cài đặt thời gian bật và thời gian tắt. Khi đến thời điểm đã cài đặt, đèn sẽ tự động bật hoặc tắt theo đúng lịch, giúp quá trình chiếu sáng được điều khiển hoàn toàn tự động và thuận tiện.



Hình 3.13. Thiết lập thời gian hẹn giờ bật - tắt đèn chiếu sáng.



Hình 3.14. Kết quả hẹn giờ của đèn chiếu sáng.

KẾT LUẬN

Kết luận:

Trong đề tài xây dựng hệ thống IoT giám sát và điều khiển môi trường chuồng trại, em đã triển khai thành công một hệ thống hoàn chỉnh đáp ứng đầy đủ các chức năng cốt lõi của một hệ thống giám sát – điều khiển môi trường chuồng trại tự động – thủ công. Hệ thống có khả năng thu thập dữ liệu từ các cảm biến môi trường, xử lý và hiển thị thông tin theo thời gian thực trên màn hình OLED và trên ứng dụng Blynk, giúp người dùng dễ dàng theo dõi các thông số quan trọng trong chuồng trại.

Bên cạnh đó, hệ thống hỗ trợ điều khiển thiết bị ở cả hai chế độ tự động và thủ công, đồng thời cho phép người dùng thao tác bằng remote hồng ngoại trong trường hợp không có kết nối Wifi. Hệ thống cũng tích hợp chức năng cảnh báo khi vượt ngưỡng và lập lịch bật – tắt đèn chiếu sáng, giúp tối ưu hoá quá trình vận hành chuồng trại và tạo môi trường sống phù hợp cho vật nuôi.

Việc sử dụng vi điều khiển ESP32 mang lại khả năng kết nối ổn định, truyền dữ liệu liên tục và nâng cao độ tin cậy trong quá trình hoạt động. Qua thực nghiệm, mô hình hoạt động ổn định, dễ sử dụng và đáp ứng tốt mục tiêu nghiên cứu đã đề ra.

Hạn chế:

Mặc dù hệ thống đã hoạt động hiệu quả, vẫn còn tồn tại một số hạn chế:

- Độ chính xác của cảm biến chưa cao, có thể bị ảnh hưởng bởi vị trí lắp đặt và điều kiện môi trường trong chuồng trại.
- Phụ thuộc vào kết nối Wifi, khi mất mạng hệ thống không thể cập nhật dữ liệu lên Blynk theo thời gian thực.
- Chưa lưu trữ dữ liệu lịch sử, chỉ hiển thị giá trị hiện tại mà chưa có khả năng thống kê hoặc phân tích dài hạn.
- Phạm vi cảm biến còn giới hạn, hệ thống mới chỉ đo nhiệt độ và độ ẩm, chưa mở rộng sang các yếu tố môi trường khác như khí độc (NH₃, CO₂), ánh sáng, chất lượng không khí...

Hướng phát triển:

Trong tương lai, hệ thống có thể được mở rộng và hoàn thiện theo các hướng sau:

- Hiệu chuẩn và nâng cấp cảm biến nhằm tăng độ chính xác và độ bền khi sử dụng trong môi trường khắc nghiệt.
- Bổ sung các loại cảm biến mới như NH₃, CO₂, cảm biến ánh sáng, cảm biến gió... để giám sát toàn diện hơn.
- Tích hợp tính năng lưu trữ dữ liệu lên Cloud (Firebase, Google Sheets, Server riêng) hoặc lưu vào thẻ nhớ để phục vụ phân tích lịch sử.
- Phát triển giao diện người dùng nâng cao trên web hoặc app riêng, thay thế cho Blynk và tăng khả năng tùy biến.
- Tự động hóa thông minh hơn, áp dụng thuật toán tối ưu hoặc AI đơn giản để dự đoán xu hướng môi trường và điều khiển thiết bị theo logic nâng cao.

TÀI LIỆU THAM KHẢO

- [1]. <https://www.pace.edu.vn/tin-kho-tri-thuc/iot-la-gi>
- [2]. <https://vina-aspire.com/internet-van-vat-iot-la-gi/>
- [3]. <https://base.vn/blog/iot-la-gi/>
- [4]. <https://fptshop.com.vn/tin-tuc/danh-gia/blynk-la-gi-184300>
- [5]. <https://docs.blynk.io/en>
- [6]. <https://stringee.com/vi/blog/post/lap-trinh-arduino-vi-sao-pho-bien>
- [7]. https://thinkrobotics.com/blogs/learn/what-is-arduino-ide-a-complete-guide-for-beginners?srsltid=AfmBOopSg5wR7FB-dW4fX4UP6_JJARPY-Mc8URd0TuTntTDWmfz0ydrC&utm_source=chatgpt.com
- [8]. https://3dpellet.com/tutorials/arduino/first-steps/arduino-ide/?utm_source=chatgpt.com
- [9]. <https://fptshop.com.vn/tin-tuc/danh-gia/phan-mem-arduino-ide-171894>
- [10]. <https://emeritus.org/blog/coding-arduino-programming-language/#advantages-of-the-arduino-programming-language>
- [11]. <https://dientuviet.com/gioi-thieu-esp32/>
- [12]. <http://arduino.vn/tutorial/1570-gioi-thieu-module-esp32-va-huong-dan-cai-trinh-bien-dich-tren-arduino-ide>
- [13]. <https://mlab.com.vn/cam-bien-tds-do-tong-chat-ran-hoa-tan-total-dissolved-solids-sensor-dfrobot-mlab-com-vn>
- [14]. <https://inno-corp.vn/cam-bien-do-do-ph-la-gi-ung-dung-va-nguyen-ly-hoat-dong/>
- [15]. <https://nshopvn.com/product/cam-bien-do-do-ph/>
- [16]. https://www.az-delivery.de/en/products/0-96zolldisplay?srsltid=AfmBOoo2wI17Vsv_hpzfHQ4s9rXYjgsOhdI4cvSzrWLO3ErUKJQRuBXd
- [17]. <https://hshop.vn/coi-buzzer-5vdc>

- [18]. <https://www.thegioiic.com/ads1115-mach-adc-16bit-i2c-2-5-5vdc>
- [19]. <https://community.blynk.cc/t/water-quality-monitoring-device-prototype/53868>
- [20]. <https://cimpleo.com/blog/arduino-ph-meter-using-ph-4502c/#wiring-diagram>
- [21]. <https://www.youtube.com/watch?v=GswTNPPCp70&t=826s>
- [22]. https://www.youtube.com/watch?v=Hjc_IFDs9jk&t=149s