

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 CHẤT LƯỢNG NƯỚC SINH HOẠT

Giảng viên hướng dẫn: **TH.S NGUYỄN THỊ VŨ THẢO**

Sinh viên thực hiện: **NGUYỄN THỊ THU AN**

Mã số sinh viên: **2151220163**

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 CHẤT
LƯỢNG NƯỚC SINH HOẠT**

Giảng viên hướng dẫn: **TH.S NGUYỄN THỊ VŨ THẢO**

Sinh viên thực hiện: **NGUYỄN THỊ THU ÂN**

Mã số sinh viên: **2151220163**

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.....):
(1)..... MSSV: Lớp:
(2)..... MSSV: Lớp:
(3)..... MSSV: Lớp:
Ngành :
Chuyên ngành :
2. **Tên đề tài:**
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)

BẢN NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN/ ĐÁNH GIÁ
ĐỒ ÁN/ KHÓA LUẬN TỐT NGHIỆP
(GVPB nộp Bản nhận xét này về Văn phòng Khoa)

6. **Họ và tên sinh viên/ nhóm sinh viên được giao đề tài** (số số trong nhóm.....):
(1)..... MSSV: Lớp:
(2)..... MSSV: Lớp:
(3)..... MSSV: Lớp:
7. **Tên đề tài:**
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 chất lượng nước sinh hoạt

Sinh viên thực hiện: Nguyễn Thị Thu Ân

MSSV: 2151220163 Lớp: 21CT6

Đề tài tập trung vào việc nghiên cứu và áp dụng công nghệ Internet of Things (IoT) nhằm xây dựng một hệ thống có khả năng giám sát chất lượng nước sinh hoạt theo thời gian thực. Hệ thống sử dụng vi điều khiển ESP32 kết hợp với các cảm biến pH, TDS và nhiệt độ để thu thập dữ liệu, sau đó truyền thông tin lên ứng dụng Blynk thông qua kết nối Wifi. Người dùng có thể quan sát trực tiếp các thông số chất lượng nước trên điện thoại và nhận cảnh báo tự động khi giá trị vượt ngưỡng an toàn.

Hệ thống được lập trình bằng ngôn ngữ C/C++ trên nền tảng Arduino IDE, tối ưu về hiệu năng, tính ổn định và khả năng mở rộng. Ngoài ra, đề tài còn hướng đến việc đảm bảo tính trực quan, dễ sử dụng và chi phí thấp, giúp mô hình có thể được triển khai rộng rãi trong hộ gia đình, khu dân cư hoặc các trạm cấp nước nhỏ.

Mô hình hoàn thiện góp phần nâng cao nhận thức về tầm quan trọng của việc kiểm soát chất lượng nước, đồng thời thể hiện khả năng ứng dụng công nghệ IoT trong các vấn đề thực tiễn của đời sống.

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 cô **Th.S Nguyễn Thị Vũ Thảo**, 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 cô 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 **Th.S Nguyễn Thị Vũ Thảo**.

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

{Chữ ký, họ và tên sinh viên}

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.....	2
1.1.3. Ứng dụng của IoT trong đời sống.....	4
1.2. Tổng quan về ứng dụng Blynk.....	6
1.2.1. Giới thiệu về Blynk.....	6
1.2.2. Cách hoạt động của Blynk.....	8
1.2.3. Ứng dụng của Blynk trong hệ thống IoT.....	11
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
1.3.4. Ưu và nhược điểm của Arduino trong lập trình IoT.....	16
1.4. Arduino IDE (Môi trường phát triển).....	18
1.4.1. Giới thiệu Arduino IDE	18
1.4.2. Các chức năng chính của Arduino IDE	19
1.4.3. Ưu và nhược điểm Arduino IDE	19
1.5. Giới thiệu về các thiết bị phần cứng.....	20

1.5.1. Thiết bị điều khiển	20
1.5.2. Thiết bị cảm biến	24
1.5.3. Thiết bị cảnh báo	27
1.5.4. Thiết bị hiển thị.....	28
Chương 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG	30
2.1. Phân tích yêu cầu chức năng.....	30
2.1.1. Thu thập dữ liệu từ các cảm biến.....	30
2.1.2. Hiển thị dữ liệu lên ứng dụng Blynk và OLED	30
2.1.3. Thiết lập cảnh báo khi giá trị vượt ngưỡng	30
2.2. Thiết kế hệ thống.....	31
2.2.1. Sơ đồ thiết kế phần cứng	31
2.2.1.1. Vi điều khiển ESP32	32
2.2.1.2. Cảm biến ADS1115	34
2.2.1.3. Cảm biến TDS (Total Dissolved Solids sensor)	35
2.2.1.4. Cảm biến pH.....	37
2.2.1.5. Cảm biến nhiệt độ DS18B20.....	38
2.2.1.6. OLED SSD1306	40
2.2.1.7. Buzzer(Còi)	41
2.2.1.8. Đèn LED(Light Emitting Diode)	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	44
2.2.3. Thiết lập ứng dụng trên Blynk.....	46
2.3. Lập trình hệ thống (Code).....	50
Chương 3: KẾT QUẢ THỰC NGHIỆM HỆ THỐNG	60
3.1. Mô hình thực tế	60
3.2. Kết quả thực nghiệm.....	63
3.2.1. Kết quả đo và hiển thị trên Blynk và màn hình OLED	63
3.2.2. Kết quả cảnh báo khi vượt ngưỡng qua ứng dụng Blynk.....	65

3.2.3. Kết quả chức năng điều khiển ngưỡng cảnh báo.....	68
3.2.4. Đánh giá hoạt động thực tế.....	75
KẾT LUẬN	76
TÀI LIỆU THAM KHẢO.....	77

DANH SÁCH HÌNH ẢNH

Hình 1.1.IoT	1
Hình 1.2 Nguyên lý hoạt động cơ bản của IoT	3
Hình 1.3 Logo Blynk	6
Hình 1.4 Blynk.Console	7
Hình 1.5 Blynk.App	8
Hình 1.6 Blynk Library / Edgent	8
Hình 1.7 Logo Arduino IDE	18
Hình 1.8 Arduino Uno	21
Hình 1.9 Arduino Nano	22
Hình 1.10 ESP8266 NodeMCU	22
Hình 1.11 ESP8266 Wemos D1 Mini	23
Hình 1.12 Raspberry Pi	24
Hình 1.13 Cảm biến nhiệt độ - độ ẩm DHT22	25
Hình 1.14 Cảm biến ánh sáng	25
Hình 1.15 Cảm biến khí MQ	26
Hình 1.16 Cảm biến siêu âm HC-SR04	27
Hình 1.17 Piezo Buzzer	27
Hình 1.18 LED RGB động	28
Hình 1.19 LCD 20x4	29
Hình 1.20 Màn hình LCD màu	29
Hình 2.1 Sơ đồ thiết kế phần cứng	31
Hình 2.2 Vi xử lý ESP32	33
Hình 2.3 Cảm biến ADS1115	34
Hình 2.4 Sơ đồ khối của cảm biến ADS1115	35
Hình 2.5 Cảm biến TDS (Total Dissolved Solids sensor)	36

Hình 2.6 Sơ đồ khối của cảm biến TDS	36
Hình 2.7 Cảm biến pH.....	37
Hình 2.8 Sơ đồ khối của cảm biến pH.....	38
Hình 2.9 Cảm biến nhiệt độ DS18B20	39
Hình 2.10 Sơ đồ khối của cảm biến nhiệt độ DS18B20.....	39
Hình 2.11 Màn hình OLED 1306	40
Hình 2.12 Sơ đồ khối của OLED 1306	41
Hình 2.13.Còi Buzzer	42
Hình 2.14 Sơ đồ khối của Buzzer.....	42
Hình 2.15 Đèn LED.....	43
Hình 2.16 Sơ đồ khối của LED	43
Hình 2.17 Trình soạn thảo và thư viện của Arduino IDE	44
Hình 2.18 Cài đặt ESP32 Board	45
Hình 2.19 Chọn board ESP32	46
Hình 2.20 Chọn cổng COM.....	46
Hình 2.21 Tạo Template cho giao diện Blynk	47
Hình 2.22 Tạo Device cho giao diện Blynk	47
Hình 2.23 Thiết lập Datastream cho giao diện Blynk	48
Hình 2.24 Tạo event cho hệ thống Blynk.....	48
Hình 2.25 Thiết lập thông báo gửi về email và ứng dụng Blynk	49
Hình 2.26 Cài đặt thời gian gửi của hệ thống trên Blynk.....	49
Hình 2.27 Giao diện hệ thống trên App Blynk.....	50
Hình 3.1 Mô hình mạch IoT khi chưa hoạt động	60
Hình 3.2 Mô hình mạch hệ thống IoT khi hoạt động	61
Hình 3.3 Hệ thống IoT giám sát chất lượng nước sinh hoạt	62
Hình 3.4 Kết quả đo hiển thị trên OLED khi bình thường.....	63
Hình 3.5 Kết quả đo hiển thị trên OLED khi vượt ngưỡng.....	63
Hình 3.6 Kết quả đo hiển thị trên Blynk	64
Hình 3.7 Kết quả cảnh báo khi vượt ngưỡng của TDS	65

Hình 3.8 Kết quả cảnh báo khi vượt ngưỡng của nhiệt độ	66
Hình 3.9 Kết quả cảnh báo khi vượt ngưỡng của pH khi vượt mức tối thiểu	67
Hình 3.10 Kết quả cảnh báo khi vượt ngưỡng của pH tối đa	68
Hình 3.11 Ngưỡng nhiệt độ trước chỉnh ngưỡng	69
Hình 3.12 Ngưỡng nhiệt độ sau chỉnh ngưỡng	69
Hình 3.13 Kết quả khi điều chỉnh ngưỡng nhiệt độ	70
Hình 3.14 Ngưỡng pH trước khi hiệu chỉnh.....	71
Hình 3.15 Ngưỡng pH tối thiểu khi chỉnh ngưỡng	71
Hình 3.16 Kết quả sau chỉnh ngưỡng pH tối thiểu.....	72
Hình 3.17 Ngưỡng pH tối đa khi chỉnh ngưỡng.....	73
Hình 3.18 Kết quả sau chỉnh ngưỡng pH tối đa	73
Hình 3.19 Ngưỡng TDS trước khi chỉnh.....	74
Hình 3.20 Ngưỡng TDS khi chỉnh ngưỡng	74
Hình 3.21 Kết quả sau chỉnh ngưỡng của TDS	75

DANH SÁCH CÁC CỤM TỪ VIẾT TẮT

Ký Hiệu	Tiếng Anh	Tiếng Việt
IoT	Internet of Things	Internet vạn vật
TDS	Total Dissolved Solids	Tổng chất rắn hòa tan
pH	potential of Hydrogen/ power of Hydrogen	Tiềm năng Hydro/lực Hydro
OLED	Organic Light-Emitting Diode	Điốt phát quang hữu cơ
LED	Light-Emitting Diode	Điốt phát quang
MQTT	Message Queuing Telemetry Transport	Vận chuyển/truyền tải dữ liệu đo đạc từ xa (viễn tải) thông qua việc xếp hàng đợi thông điệp
HTTP	Hypertext Transfer Protocol	Giao thức truyền tải siêu văn bản
LDR	Light Dependent Resistor	Cảm biến ánh sáng có điện trở thay đổi theo cường độ sáng chiếu vào
TFT	Thin-Film Transistor	Bóng bán dẫn màng mỏng
LCD	Liquid Crystal Display	Màn hình tinh thể lỏng

MỞ ĐẦU

1. Lý do chọn đề tài

Trong đời sống hiện nay, chất lượng nước sinh hoạt có ảnh hưởng trực tiếp đến sức khỏe, đời sống con người. Vì ở nhiều khu vực, nguồn nước thường bị ô nhiễm bởi các tạp chất hoặc không đạt tiêu chuẩn do ô nhiễm môi trường và hệ thống cấp nước xuống cấp.

Với sự phát triển của công nghệ Internet of Things (IoT), các thiết bị cảm biến có thể được kết nối Internet để thu thập và truyền dữ liệu tự động theo thời gian thực. Nhờ đó, việc giám sát các chỉ số môi trường như độ pH, độ dẫn điện (TDS) hay nhiệt độ nước có thể được thực hiện một cách nhanh chóng, chính xác và tiện lợi.

Từ thực tế đó, đề tài “Xây dựng hệ thống IoT giám sát chất lượng nước sinh hoạt” được thực hiện nhằm tạo ra một mô hình nhỏ gọn, dễ triển khai và hoạt động ổn định. Hệ thống sử dụng vi điều khiển ESP32 để thu thập dữ liệu từ các cảm biến pH, TDS, nhiệt độ, sau đó gửi dữ liệu lên ứng dụng Blynk thông qua mạng Wifi. Người dùng có thể quan sát trực tiếp chất lượng nước trên điện thoại và nhận cảnh báo khi giá trị vượt ngưỡng cho phép.

Hệ thống này hướng đến mục tiêu giúp người dân dễ dàng giám sát nguồn nước sinh hoạt hằng ngày, đồng thời có thể mở rộng để ứng dụng trong các khu dân cư, trường học, hoặc trạm cấp nước quy mô nhỏ. Đề tài cũng góp phần giúp sinh viên hiểu rõ hơn về cách ứng dụng công nghệ IoT trong các vấn đề thực tế của đời sống.

2. Mục tiêu đề tài

Mục tiêu chính của đề tài là xây dựng một hệ thống IoT có khả năng giám sát chất lượng nước sinh hoạt theo thời gian thực, giúp người dùng dễ dàng theo dõi các thông số cơ bản của nước và kịp thời phát hiện khi chất lượng nước không đạt tiêu chuẩn.

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

1. Thiết kế và xây dựng hệ thống phần cứng gồm vi điều khiển ESP32 và các cảm biến pH, TDS, nhiệt độ có khả năng thu thập dữ liệu chính xác và ổn định.
2. Lập trình xử lý và truyền dữ liệu từ ESP32 lên nền tảng Blynk thông qua kết nối Wifi, đảm bảo dữ liệu được cập nhật liên tục.

3. Xây dựng giao diện hiển thị trên ứng dụng Blynk, cho phép người dùng quan sát các giá trị đo được một cách trực quan trên điện thoại.
4. Thiết lập ngưỡng cảnh báo tự động, giúp người dùng nhận biết khi chất lượng nước vượt ngoài giới hạn an toàn (ví dụ: pH quá thấp hoặc TDS quá cao).
5. Đánh giá và kiểm thử hệ thống, từ đó rút ra nhận xét, ưu điểm và hạn chế để hoàn thiện mô hình.

Thông qua đề tài này, sinh viên chúng em có thể vận dụng kiến thức đã học về vi điều khiển, lập trình IoT, cảm biến và truyền dữ liệu không dây để tạo ra một sản phẩm có tính ứng dụng thực tế cao, phục vụ đời sống hằng ngày.

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 chất lượng nước sinh hoạt dựa trên nền tảng vi điều khiển ESP32 và ứng dụng Blynk.

Các đối tượng nghiên cứu chính bao gồm:

- Cảm biến pH: đo độ axit hoặc độ kiềm của nước.
- Cảm biến TDS (Total Dissolved Solids): xác định tổng lượng chất rắn hòa tan trong nước, phản ánh độ tinh khiết của nguồn nước.
- Cảm biến nhiệt độ DS18B20: đo nhiệt độ nước để hỗ trợ phân tích và giám sát điều kiện môi trường.
- Vi điều khiển ESP32: xử lý dữ liệu thu được từ các cảm biến và gửi lên Internet qua Wifi.
- Ứng dụng Blynk: làm giao diện hiển thị dữ liệu thời gian thực và cảnh báo khi các chỉ số vượt ngưỡng an toàn.

b. Phạm vi nghiên cứu

- Hệ thống chỉ thực hiện giám sát và cảnh báo các thông số cơ bản của nước (pH, TDS, nhiệt độ), không bao gồm chức năng xử lý hoặc lọc nước.
- Dữ liệu được truyền qua Wifi lên nền tảng Blynk, không triển khai máy chủ riêng hay lưu trữ dữ liệu chuyên sâu.
- Mô hình được thực hiện ở quy mô thử nghiệm, nhưng có thể mở rộng ứng dụng thực tế cho hộ gia đình, khu dân cư nhỏ hoặc các trạm cấp nước.
- Đề tài tập trung vào độ ổn định, độ chính xác của cảm biến và khả năng hiển thị, cảnh báo của hệ thống.

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 giám sát chất lượng nước dựa trên IoT. 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 TDS, cảm biến pH, cảm biến nhiệt độ, màn hình OLED, buzzer và LED. 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); 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.

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.



Hình 1.1.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

Hệ thống Internet of Things (IoT) thường hoạt động theo một quy trình gồm bốn bước chính: thu thập dữ liệu, chia sẻ dữ liệu, xử lý dữ liệu, và đưa ra quyết định. Quy trình này đảm bảo dữ liệu từ môi trường thực được ghi nhận, truyền tải, xử lý, và chuyển thành thông tin hữu ích phục vụ cho việc điều khiển, giám sát hoặc ra quyết định thông minh.

Thu thập dữ liệu:

Ở giai đoạn đầu tiên, các cảm biến hoặc thiết bị IoT được sử dụng để thu thập dữ liệu từ môi trường xung quanh. Các loại dữ liệu này có thể rất đa dạng từ những thông số đơn giản như nhiệt độ, độ ẩm, áp suất, ánh sáng, cho đến những dữ liệu phức tạp hơn như hình ảnh, video.

Nhờ các cảm biến này, hệ thống có thể ghi nhận tình trạng thực tế một cách liên tục, chính xác, tạo nên nền tảng thông tin cho các bước xử lý tiếp theo. Dữ liệu thu được từ bước này chính là “nguyên liệu đầu vào” của toàn bộ hệ thống IoT.

Chia sẻ dữ liệu:

Sau khi được thu thập, dữ liệu sẽ được truyền tải và chia sẻ đến các hệ thống khác thông qua mạng Internet. Các cảm biến và thiết bị IoT được kết nối mạng giúp dữ liệu

được gửi lên hệ thống lưu trữ đám mây (cloud storage), nơi dữ liệu được lưu giữ an toàn và có thể truy cập từ xa.

Bước chia sẻ dữ liệu này đóng vai trò cực kỳ quan trọng, vì nó giúp thông tin được tập trung hóa và đồng bộ hóa, tạo điều kiện thuận lợi cho việc xử lý và phân tích ở bước tiếp theo.

Xử lý dữ liệu:

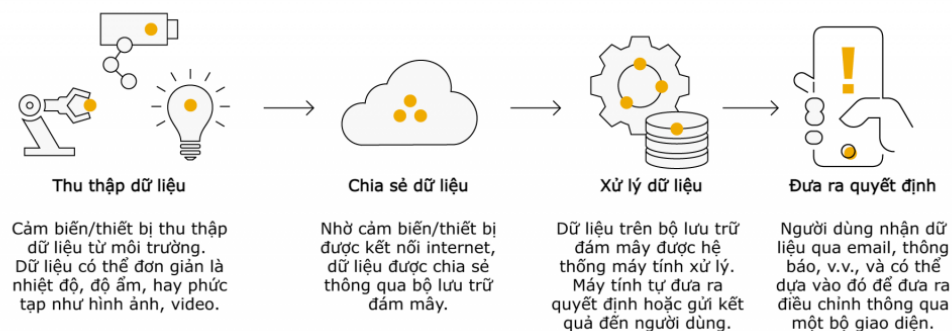
Khi dữ liệu đã được lưu trữ trên nền tảng đám mây, các hệ thống máy tính hoặc phần mềm phân tích dữ liệu sẽ tiến hành xử lý. Dữ liệu được làm sạch, sắp xếp, tổng hợp, và phân tích nhằm trích xuất ra các thông tin hữu ích.

Tùy theo yêu cầu, hệ thống có thể sử dụng thuật toán phân tích dữ liệu, trí tuệ nhân tạo (AI) hoặc các mô hình học máy (machine learning) để đưa ra kết quả, dự đoán hoặc cảnh báo.

Kết quả xử lý này sau đó được gửi đến người dùng hoặc các bộ phận điều khiển khác trong hệ thống.

Đưa ra quyết định:

Bước cuối cùng là đưa ra quyết định hoặc hành động dựa trên dữ liệu đã xử lý. Người dùng có thể nhận kết quả thông qua email, ứng dụng điện thoại, bảng điều khiển (dashboard) hoặc các hình thức thông báo khác.



Hình 1.2 Nguyên lý hoạt động cơ bản của IoT

Dựa vào thông tin đó, người dùng có thể đưa ra các điều chỉnh cần thiết (chẳng hạn như thay đổi thiết lập, kích hoạt hoặc tắt một thiết bị). Trong nhiều trường hợp, hệ thống còn có khả năng tự động ra quyết định, ví dụ như tự điều chỉnh nhiệt độ phòng, bật cảnh báo an ninh, hay tối ưu hóa hoạt động của thiết bị mà không cần con người can thiệp trực tiếp.

1.1.3. Ứng dụng của IoT trong đời sống

Trong lĩnh vực sản xuất:

Mục đích ứng dụng IoT là:

- Giám sát nhiệt độ, độ rung, mức điện tiêu thụ của máy móc trong thời gian thực.
- Phát hiện dấu hiệu bất thường, tự động điều chỉnh hoặc gửi cảnh báo để bảo trì kịp thời.

Dữ liệu thu thập & Phản hồi:

- Giám sát nhiệt độ, độ rung, mức tiêu thụ điện của máy móc trong thời gian thực.
- Phát hiện dấu hiệu bất thường, tự động điều chỉnh hoặc gửi cảnh báo để bảo trì kịp thời.

Ví dụ về thiết bị IoT:

- Hệ thống cảm biến rung động, áp suất, dòng điện.
- Robot công nghiệp.
- Hệ thống dự đoán bảo trì.

Trong lĩnh vực Y tế và chăm sóc sức khỏe:

Mục đích ứng dụng IoT là:

- Theo dõi sức khỏe bệnh nhân từ xa nhằm giảm tải cho bệnh viện hoặc cơ sở y tế.

Dữ liệu thu thập & Phản hồi:

- Theo dõi nhịp tim, huyết áp, nồng độ oxy trong máu.
- Gửi dữ liệu đến ứng dụng hoặc bác sĩ để kịp thời phát hiện các dấu hiệu nguy hiểm.

Ví dụ về thiết bị IoT:

- Máy đo huyết áp thông minh.
- Giường bệnh thông minh.
- Thiết bị theo dõi sức khỏe đeo tay.
- Robot phẫu thuật.

Trong lĩnh vực giao thông vận tải:

Mục đích ứng dụng IoT là:

- Giảm ùn tắc.

- Cải thiện an toàn đường bộ.
- Tối ưu hóa hiệu suất của phương tiện.

Dữ liệu thu thập & Phản hồi:

- Theo dõi lưu lượng giao thông, tốc độ xe cộ, mật độ xe.
- Tự động điều chỉnh đèn tín hiệu để giảm tắc nghẽn, cảnh báo hành vi lái xe nguy hiểm và cải thiện ý thức lái xe.

Ví dụ về thiết bị IoT:

- Hệ thống đèn giao thông thông minh.
- Bãi đỗ xe thông minh.
- Hệ thống theo dõi phương tiện.

Nhà thông minh:

Mục đích ứng dụng IoT là:

- Nâng cao sự tiện lợi.
- Tiết kiệm năng lượng.
- Tăng cường an ninh và giám sát.

Dữ liệu thu thập & Phản hồi:

- Cảm biến đo nhiệt độ, ánh sáng, âm thanh, chuyển động.
- Tự động điều chỉnh hoạt động theo điều kiện môi trường xung quanh, hoặc gửi thông báo để người dùng ra quyết định.

Ví dụ về thiết bị IoT:

- Hệ thống đèn, điều hòa, máy giặt, khóa cửa, bếp thông minh.
- Camera giám sát.

Trong lĩnh vực nông nghiệp:

Mục đích ứng dụng IoT là:

- Tự động hóa chăm sóc cây trồng.
- Dự báo sản lượng.
- Tối ưu hóa năng suất mùa vụ.

Dữ liệu thu thập & Phản hồi:

- Theo dõi điều kiện của đất (độ ẩm, pH, hàm lượng dinh dưỡng,...), ánh sáng, nhiệt độ.
- Cung cấp cho người nông dân dữ liệu sức khỏe cây trồng theo thời gian thực, đồng thời đề xuất tưới tiêu, bón phân hợp lý.

Ví dụ về thiết bị IoT:

- Hệ thống tưới tiêu tự động.
- Thiết bị theo dõi sức khỏe gắn trên vật nuôi.
- Máy bay không người lái (drone) phun thuốc bảo vệ thực vật.

1.2. Tổng quan về ứng dụng Blynk

1.2.1. Giới thiệu về Blynk

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.



Hình 1.3 Logo Blynk

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.

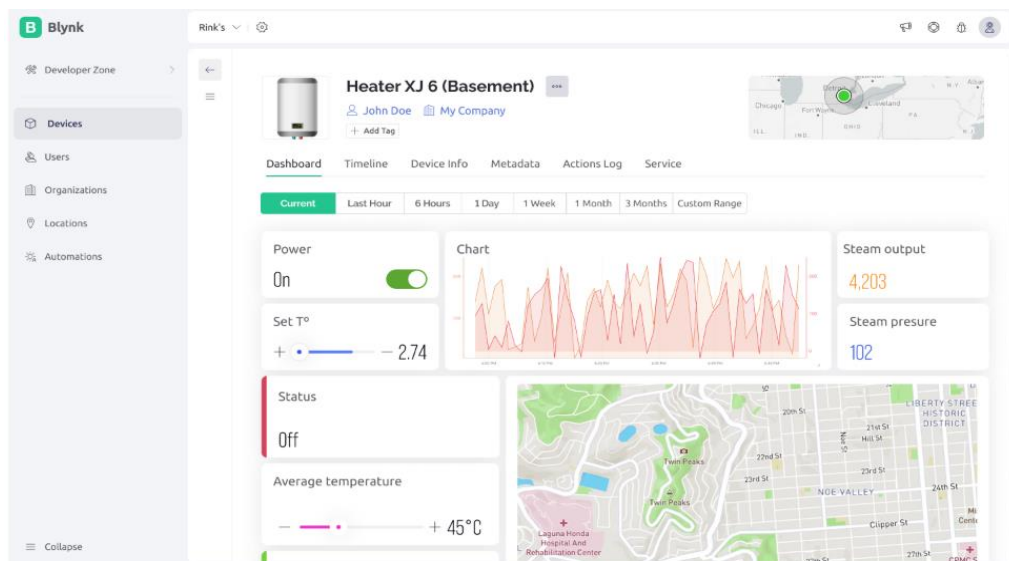
Cụ thể, người dùng có thể tạo ra các ứng dụng để điều khiển đèn, quạt, cảm biến nhiệt độ, đo lường độ ẩm và thậm chí là máy rửa chén thông minh. Blynk hỗ trợ đa nền tảng, từ các thiết bị di động như điện thoại thông minh đến các thiết bị nhúng như Arduino, Raspberry Pi, ESP8266 và nhiều hơn nữa.

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.Cloud: Máy chủ đám mây trung tâm để kết nối và xử lý dữ liệu giữa thiết bị và người dùng.

Blynk.Console: Giao diện web cho phép giám sát, quản lý thiết bị, người dùng, dữ liệu và tự động hóa.



Hình 1.4 Blynk.Console

Blynk.App: Ứng dụng di động trên iOS/Android dùng để điều khiển và theo dõi thiết bị theo thời gian thực.



Hình 1.5 Blynk.App

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.

```
1 // Fill-in information from your Blynk Template here
2 #define BLYNK_TEMPLATE_ID          ""
3 #define BLYNK_TEMPLATE_NAME        ""
4 #define BLYNK_FIRMWARE_VERSION     "0.1.0"
5 #define APP_DEBUG
6
7 #include "BlynkEdgent.h"
8
9 void setup()
10 {
11     delay(100);
12     BlynkEdgent.begin();
13 }
14
15 void loop() {
16     BlynkEdgent.run();
17 }
18
19
```

Hình 1.6 Blynk Library / Edgent

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.

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 bắt đầu, việc kết nối thiết bị với nền tảng Blynk yêu cầu chuẩn bị đầy đủ cả phần cứng lẫn phần mềm. Các thiết bị IoT phổ biến có thể sử dụng bao gồm Arduino, ESP8266, ESP32, Raspberry Pi hoặc STM32. Về phần mềm, cần cài đặt môi trường lập trình tương ứng như Arduino, đồng thời tích hợp thư viện Blynk thông qua Library Manager hoặc GitHub. Khi tạo dự án mới trên Blynk.Console, hệ thống sẽ cung cấp một mã Auth Token duy nhất cho thiết bị, đóng vai trò là thông tin xác thực giúp thiết bị giao tiếp an toàn với Blynk Cloud.

Sau khi chuẩn bị xong phần cứng và phần mềm, bước tiếp theo là cấu hình kết nối mạng cho thiết bị. Hầu hết các thiết bị IoT hiện nay hỗ trợ kết nối WiFi; với một số thiết bị khác có thể sử dụng Ethernet hoặc mạng di động 3G/4G. Trong code của thiết bị, người dùng cần khai báo tên mạng và mật khẩu, nhờ đó thiết bị có thể tự động kết nối khi khởi động, đảm bảo việc truyền dữ liệu đến Cloud diễn ra ổn định và liên tục.

Tiếp theo, thiết bị sử dụng mã Auth Token để kết nối với Blynk Cloud. Việc này được thực hiện thông qua các hàm của thư viện Blynk, giúp thiết bị xác thực danh tính và bắt đầu trao đổi dữ liệu với Cloud. Quá trình xác thực đảm bảo rằng chỉ những thiết bị hợp lệ mới có thể kết nối, từ đó nâng cao tính bảo mật và đáng tin cậy của hệ thống.

Khi kết nối thành công, thiết bị sẽ sẵn sàng nhận các lệnh điều khiển từ người dùng và gửi dữ liệu cảm biến lên Cloud. Trạng thái kết nối của thiết bị có thể được theo dõi trực tiếp trên Blynk.App hoặc Blynk.Console, nơi thiết bị sẽ hiển thị trạng thái “online”. Điều này cho thấy thiết bị đã sẵn sàng hoạt động trong hệ sinh thái Blynk và tương tác với các ứng dụng quản lý từ xa. Trong quá trình thiết lập, nếu xảy ra lỗi kết nối, cần kiểm tra lại thông tin mạng, xác thực Auth Token, cũng như khả năng truy cập Internet của thiết bị để đảm bảo 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 hỗ trợ tạo Automation — tức là các quy tắc hoạt động tự động:

- Ví dụ: khi cảm biến nhiệt độ $> 35^{\circ}\text{C}$ → tự động bật quạt hoặc gửi cảnh báo đến người dùng.
- Các automation này được cấu hình trực tiếp trong Blynk.Console hoặc Blynk.App, và được xử lý trên nền tảng Blynk Cloud mà không cần người dùng can thiệp thủ công.

Quản lý và cập nhật thiết bị:

Ngoài việc giám sát, Blynk còn cho phép quản lý và cập nhật thiết bị từ xa:

- Quản lý danh sách thiết bị, người dùng, và tổ chức.

- Cập nhật phần mềm (firmware) qua Internet (OTA – Over The Air) để sửa lỗi hoặc thêm tính năng mới.
- Kiểm tra trạng thái thiết bị (online/offline, pin, tín hiệu mạng) và ghi lại nhật ký hoạt động.

1.2.3. Ứng dụng của Blynk trong hệ thống IoT

Trong lĩnh vực sản xuất:

Mục đích ứng dụng IoT là:

- Giám sát trạng thái máy móc từ xa qua điện thoại/máy tính bảng.
- Nhận cảnh báo khi máy móc có dấu hiệu bất thường.
- Điều khiển hoặc tạm dừng máy tự động từ xa để bảo trì kịp thời.

Dữ liệu thu thập & Phản hồi:

- Nhiệt độ, độ rung, mức điện tiêu thụ của máy móc.
- Hiển thị dữ liệu theo thời gian thực trên ứng dụng Blynk.
- Gửi cảnh báo tự động nếu vượt ngưỡng.

Ví dụ về thiết bị IoT:

- Hệ thống cảm biến rung động, áp suất, dòng điện kết nối Blynk.
- Robot công nghiệp có thể điều khiển từ xa.
- Hệ thống dự đoán bảo trì tích hợp Blynk để cảnh báo kịp thời.

Trong lĩnh vực Y tế và chăm sóc sức khỏe:

Mục đích ứng dụng IoT là:

- Theo dõi sức khỏe bệnh nhân từ xa.
- Gửi dữ liệu đến bác sĩ, trung tâm y tế để phát hiện sớm các dấu hiệu nguy hiểm.

Dữ liệu thu thập & Phản hồi:

- Nhịp tim, huyết áp, nồng độ oxy trong máu.
- Hiển thị trực quan trên ứng dụng Blynk.
- Cảnh báo tức thời khi có bất thường.

Ví dụ về thiết bị IoT:

- Máy đo huyết áp, nhịp tim thông minh.

- Giường bệnh thông minh kết nối Blynk.
- Thiết bị theo dõi sức khỏe đeo tay.
- Robot phẫu thuật được giám sát từ xa.

Trong lĩnh vực giao thông vận tải:

Mục đích ứng dụng IoT là:

- Giám sát lưu lượng giao thông và tốc độ xe từ xa.
- Điều khiển hệ thống đèn tín hiệu thông minh, giảm tắc nghẽn.
- Cảnh báo hành vi lái xe nguy hiểm.

Dữ liệu thu thập & Phản hồi:

- Lưu lượng giao thông, mật độ xe, tốc độ xe.
- Hiện thị dữ liệu trên Blynk theo thời gian thực.
- Nhận cảnh báo và điều khiển đèn giao thông từ xa.

Ví dụ về thiết bị IoT:

- Hệ thống đèn giao thông thông minh.
- Bãi đỗ xe thông minh.
- Hệ thống theo dõi phương tiện và camera giám sát.

Nhà thông minh:

Mục đích ứng dụng IoT là:

- Điều khiển thiết bị điện tử từ xa.
- Tăng tiện nghi, tiết kiệm năng lượng và giám sát an ninh.

Dữ liệu thu thập & Phản hồi:

- Cảm biến nhiệt độ, ánh sáng, âm thanh, chuyển động.
- Hiện thị dữ liệu và trạng thái thiết bị trên Blynk.
- Tự động điều chỉnh thiết bị theo điều kiện môi trường hoặc gửi thông báo cho người dùng.

Ví dụ về thiết bị IoT:

- Đèn, điều hòa, máy giặt, khóa cửa, bếp thông minh.
- Camera giám sát và hệ thống cảnh báo an ninh.

Trong lĩnh vực nông nghiệp:

Mục đích ứng dụng IoT là:

- Giám sát cây trồng, đất, thời tiết từ xa.
- Điều khiển tưới tiêu, bón phân tự động.
- Dự báo sản lượng và tối ưu hóa năng suất.

Dữ liệu thu thập & Phản hồi:

- Theo dõi điều kiện của đất (độ ẩm, pH, hàm lượng dinh dưỡng...), ánh sáng, nhiệt độ.
- Hiện thị trạng thái cây trồng, cảnh báo bất thường trên Blynk.
- Điều khiển hệ thống tưới tiêu, thiết bị trên đồng ruộng từ xa.

Ví dụ về thiết bị IoT:

- Hệ thống tưới tiêu tự động.
- Thiết bị theo dõi sức khỏe vật nuôi.
- Drone phun thuốc bảo vệ thực vật.

1.3. Ngôn ngữ lập trình Arduino

1.3.1. Khái niệm về ngôn ngữ lập trình Arduino

Khái niệm:

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.

Chương trình Arduino có thể được chia làm 3 phần gồm:

Cấu trúc (structure):

Đây là bố cục tổng thể của chương trình Arduino, bao gồm hai hàm bắt buộc là `setup()` và `loop()`.

- `setup()`: Chứa các lệnh được thực hiện một lần khi mạch Arduino khởi động.
- `loop()`: Chứa các lệnh được thực hiện liên tục lặp lại.

Biến số (variable) và hằng số (constant):

- Biến số: Dùng để lưu trữ dữ liệu có thể thay đổi trong quá trình chạy chương trình, ví dụ: `int ledPin = 13;`
- Hằng số: Dùng để lưu trữ dữ liệu không thay đổi trong quá trình chạy, ví dụ: `const int threshold = 100;`

Hàm và thủ tục (function):

- Là các khối lệnh thực hiện một nhiệm vụ cụ thể, giúp chương trình trở nên rõ ràng và dễ quản lý.
- Ví dụ: Bạn có thể viết hàm để bật tắt đèn LED:

```
void blinkLED(int pin, int delayTime) {  
  
    digitalWrite(pin, HIGH);  
  
    delay(delayTime);  
  
    digitalWrite(pin, LOW);  
  
    delay(delayTime);  
  
}
```

Lịch sử và nguồn gốc phát triển:

Ngôn ngữ lập trình Arduino và nền tảng phần cứng Arduino có nguồn gốc từ một dự án nghiên cứu tại một trung tâm nghiên cứu và phát triển ở Ý. Dự án được khởi đầu vào năm 2003 bởi một nhóm các nhà khoa học và kỹ sư gồm Massimo Banzi, David Cuartielles, David Mellis, Tom Igoe và Gianluca Martino.

Tên "Arduino" được lấy từ tên một ngôi làng tại Ý, nơi các thành viên sáng lập nhóm có một buổi họp. Ban đầu, họ phát triển một board mạch phụ thuộc vào vi điều khiển của hãng Atmel và sử dụng ngôn ngữ lập trình Wiring. Sau đó, nhóm quyết định mở nguồn mã nguồn và phần cứng của dự án, cho phép mọi người có thể tham gia, phát triển và chia sẻ thông qua cộng đồng Arduino.

Từ khi ra mắt đến nay, Arduino đã trở thành một trong những nền tảng phổ biến nhất cho việc phát triển các dự án IoT, điện tử, robot và nhiều ứng dụng khác. Sự phát

triển và sự đóng góp từ cộng đồng đã giúp Arduino trở thành một công cụ hữu ích cho việc khám phá và sáng tạo trong lĩnh vực công nghệ.

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 <Adafruit_ADS1X15.h>

Thư viện này dùng để điều khiển module ADC ngoài ADS1115/ADS1015, giúp đo tín hiệu analog chính xác hơn so với ADC tích hợp của Arduino/ESP32. ADS1115 có độ phân giải 16 bit, rất hữu ích khi đọc cảm biến điện áp nhỏ hoặc nhạy, đảm bảo dữ liệu đo được đáng tin cậy cho 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 <OneWire.h>

Thư viện One Wire giao tiếp với các thiết bị dùng giao thức 1-Wire, phổ biến là cảm biến DS18B20. Chỉ cần một dây dữ liệu cho nhiều cảm biến, giúp hệ thống IoT gọn gàng, dễ lắp đặt và dễ mở rộng.

#include <DallasTemperature.h>

Thư viện này hỗ trợ đọc dữ liệu từ cảm biến DS18B20 dựa trên One Wire. Nó cho phép đọc nhiệt độ chính xác, hỗ trợ nhiều cảm biến trên cùng một bus và chuyển đổi đơn vị dễ dàng, rất phù hợp cho các ứng dụng giám sát môi trường.

#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.

1.3.4. Ưu và nhược điểm của Arduino trong lập trình IoT

Ưu điểm:

Năm lợi thế chính của ngôn ngữ lập trình Arduino là:

- **Dễ học và sử dụng:** Ngôn ngữ lập trình Arduino dựa trên C++, có cú pháp đơn giản và dễ hiểu, ngay cả người mới bắt đầu cũng có thể hiểu được.

- **Tính linh hoạt:** Bo mạch Arduino có thể được sử dụng trong nhiều dự án khác nhau, chẳng hạn như robot, tự động hóa gia đình và các ứng dụng IoT.
- **Được hỗ trợ rộng rãi:** Cộng đồng Arduino rất lớn và năng động, có rất nhiều tài nguyên và hướng dẫn giúp người dùng học tập và khắc phục sự cố.
- **Tiết kiệm chi phí:** Bo mạch và linh kiện Arduino có giá thành tương đối rẻ, phù hợp với những người đam mê và sinh viên.
- **Nguồn mở:** Phần mềm và phần cứng Arduino là mã nguồn mở, nghĩa là người dùng có thể truy cập vào mã nguồn và sửa đổi mã nguồn cho phù hợp với nhu cầu của mình.

Nhược điểm:

Ngôn ngữ lập trình Arduino cũng có một số nhược điểm cần lưu ý như sau:

- **Bộ nhớ và sức mạnh xử lý còn hạn chế:** So với các bo mạch vi điều khiển lớn hơn hoặc máy tính hoàn chỉnh, bo mạch Arduino có bộ nhớ và khả năng xử lý hạn chế. Điều này có thể hạn chế độ phức tạp và quy mô của các dự án được xây dựng bằng Arduino.
- **Hỗ trợ hạn chế cho một số giao thức truyền thông:** Bo mạch Arduino có hỗ trợ tích hợp cho một số giao thức truyền thông, chẳng hạn như USB, Ethernet và Wifi, nhưng không hỗ trợ các giao thức khác như Zigbee, Z-wave, Thread hoặc LoRa.
- **Hiệu suất thời gian thực hạn chế:** Bo mạch Arduino không được thiết kế cho các ứng dụng thời gian thực đòi hỏi độ chính xác cao về thời gian và khả năng phản hồi.
- **Tính năng bảo mật hạn chế:** Hơn nữa, bo mạch Arduino có các tính năng bảo mật hạn chế, có thể khiến chúng dễ bị tấn công mạng hoặc các loại tấn công mạng khác.
- **Độ chính xác còn hạn chế:** Tuy nhiên, một số bo mạch Arduino có độ chính xác hạn chế khi chuyển đổi tín hiệu analog sang tín hiệu số; điều này có thể ảnh hưởng đến độ chính xác của một số ứng dụng.
- **Khả năng mở rộng hạn chế:** Bo mạch Arduino không phù hợp cho các dự án quy mô lớn. Chúng cũng không được khuyến khích sử dụng cho các ứng dụng chuyên nghiệp hoặc công nghiệp.

1.4. Arduino IDE (Môi trường phát triển)

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.

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.



Hình 1.7 Logo Arduino IDE

1.4.2. Các chức năng chính của Arduino IDE

Soạn thảo mã nguồn (Code Editor):

Arduino IDE có trình soạn thảo giúp viết chương trình (Sketch) bằng ngôn ngữ C/C++ đơn giản hóa. Nó hỗ trợ tô màu cú pháp, xuống dòng tự động và tab indent, giúp code rõ ràng và dễ đọc.

Biên dịch và nạp chương trình(Compile & Upload):

IDE biên dịch code thành file nhị phân và nạp trực tiếp vào bo mạch Arduino hoặc ESP32. Điều này giúp lập trình viên kiểm tra lỗi cú pháp và chạy chương trình trên thiết bị một cách nhanh chóng.

Quản lý thư viện (Library Manager):

Arduino IDE cho phép cài đặt, xóa hoặc cập nhật thư viện hỗ trợ cảm biến, màn hình, Wifi và các thiết bị IoT khác. Thư viện giúp mở rộng chức năng phần cứng mà không cần viết code từ đầu.

Màn hình Serial Monitor:

Serial Monitor hiển thị dữ liệu từ Arduino gửi về máy tính, hỗ trợ gỡ lỗi và giám sát hoạt động chương trình. Người dùng cũng có thể gửi lệnh từ máy tính vào Arduino để tương tác trực tiếp.

Hỗ trợ nhiều bo mạch (Board Manager):

IDE tích hợp danh sách bo mạch Arduino phổ biến và cho phép cài thêm các board khác như ESP32, ESP8266 thông qua Board Manager. Điều này giúp lập trình trên nhiều loại phần cứng một cách dễ dàng.

Ví dụ:

Arduino IDE cung cấp nhiều ví dụ minh họa cách sử dụng cảm biến, module và các ứng dụng IoT. Những ví dụ này giúp người mới học nhanh chóng hiểu cách lập trình và áp dụng vào dự án thực tế.

1.4.3. Ưu và nhược điểm Arduino IDE

Ưu điểm:

- **Khả năng sử dụng:** Arduino IDE có giao diện thân thiện, giúp lập trình trở nên dễ dàng và thuận tiện cho cả người mới bắt đầu lẫn người có kinh nghiệm.

- **Thân thiện với người mới:** Thiết kế đơn giản giúp việc lập trình trở nên trực quan, thú vị và hiệu quả, phù hợp với những người lần đầu làm quen.
- **Dễ sử dụng:** Người dùng đánh giá Arduino IDE rất dễ dùng, đặc biệt thuận tiện cho người mới bắt đầu học lập trình.
- **Quản lý thư viện:** Hệ thống quản lý thư viện được đơn giản hóa, giúp người dùng dễ dàng cài đặt và sử dụng các thư viện cần thiết, nâng cao trải nghiệm lập trình.
- **Đơn giản và trực quan:** Thiết kế thân thiện giúp việc viết code cho các bo mạch Arduino và dự án IoT trở nên nhanh chóng, hiệu quả và dễ quản lý.

Nhược điểm:

- **Khó khăn khi debug:** Người dùng gặp khó khăn khi theo dõi và sửa lỗi do các công cụ debug của Arduino IDE còn hạn chế so với các môi trường lập trình nâng cao.
- **Khả năng tùy chỉnh hạn chế:** Người dùng gặp khó khăn khi thêm công cụ hoặc bo mạch ngoài để kiểm thử vi điều khiển.
- **Tính năng hạn chế:** So với các môi trường lập trình chuyên nghiệp, Arduino IDE có chức năng hạn chế, đặc biệt trong việc debug hoặc quản lý dự án phức tạp.
- **Vấn đề công cụ:** Người dùng đôi khi gặp trục trặc với các công cụ, gặp khó khăn khi thêm thư viện, công cụ hoặc bo mạch bên ngoài.

1.5. Giới thiệu về các thiết bị phần cứng

1.5.1. Thiết bị điều khiển

Trong một hệ thống IoT, thiết bị điều khiển (Controller/MCU – Microcontroller Unit) đóng vai trò như bộ não trung tâm, đảm nhiệm toàn bộ quá trình vận hành của hệ thống. Thiết bị này có nhiệm vụ thu thập dữ liệu từ các cảm biến, xử lý và phân tích các tín hiệu đầu vào để nhận biết trạng thái môi trường hoặc thiết bị. Đồng thời, nó điều khiển các phần tử chấp hành như đèn LED, động cơ hay relay dựa trên kết quả xử lý. Bên cạnh đó, MCU còn chịu trách nhiệm truyền dữ liệu lên internet hoặc các nền tảng IoT như Blynk, MQTT hay Firebase, cũng như tiếp nhận lệnh và cấu hình từ người dùng gửi xuống. Nhờ khả năng kết nối cảm biến – xử lý dữ liệu – truyền thông, thiết bị điều khiển trở thành trung tâm điều phối quan trọng, đảm bảo hệ thống IoT hoạt động ổn định và trơn tru.

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

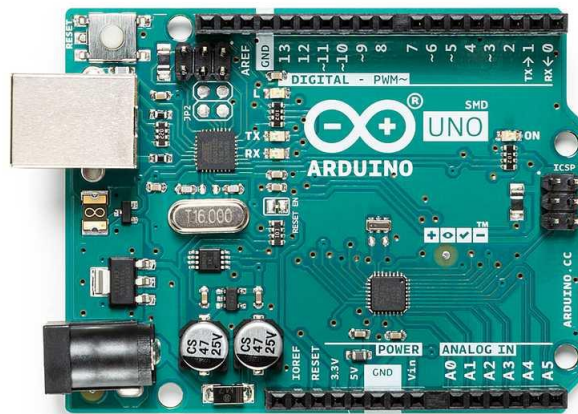
a) Arduino Uno :

Arduino Uno là bo mạch Arduino phổ biến nhất, dễ dùng cho người mới bắt đầu và phù hợp cho học tập, thực hành và làm prototype. Nó hoạt động ổn định, kết nối chắc chắn và hỗ trợ nhiều shield mở rộng, rất tiện khi làm các mô hình thử nghiệm hoặc dự án giáo dục.

Ưu điểm: Dễ sử dụng, bền, tiện kết nối và phù hợp cho các bài học STEM hoặc robot đơn giản.

Nhược điểm: Kích thước khá lớn, nên không thích hợp cho các sản phẩm nhỏ gọn.

Ứng dụng: Các bài học lập trình – điện tử, robot cơ bản, mô hình điều khiển (động cơ, relay, cảm biến), hoặc các dự án nguyên mẫu cần tính ổn định.



Hình 1.8 Arduino Uno

b) Arduino Nano:

Arduino Nano nhỏ gọn hơn nhiều so với Uno, vẫn dùng cùng loại vi điều khiển nhưng có lợi thế về kích thước và dễ gắn vào các thiết bị nhúng. Nó phù hợp cho những dự án cần mạch nhỏ, nhẹ hoặc lắp vào sản phẩm thực tế.

Ưu điểm: Nhỏ, rẻ, phù hợp cho mạch nhúng, và các sản phẩm cần tiết kiệm không gian.

Nhược điểm: Cổng USB nhỏ dễ hỏng hơn và không dùng được hầu hết shield lớn của Uno

Ứng dụng: Thiết bị đo mini, mạch in PCB, robot nhỏ, hệ thống IoT gọn lẹ, thiết bị đeo, hoặc các dự án cần nhiều đầu analog.



Hình 1.9 Arduino Nano

c) **ESP8266 NodeMCU:**

NodeMCU là bo phát triển dựa trên ESP8266, tích hợp sẵn WiFi và có thể lập trình trực tiếp bằng Arduino IDE. Kích thước của nó tương đối lớn nên thao tác thử nghiệm rất thuận tiện, đặc biệt phù hợp cho người mới làm IoT.

Ưu điểm: Dễ lập trình, có sẵn mạch nạp USB, hoạt động ổn định và phù hợp cho các dự án IoT cỡ vừa.

Nhược điểm: Kích thước khá lớn nên không phù hợp cho các thiết bị nhỏ gọn hoặc sản phẩm cần gắn trực tiếp lên PCB mini.

Ứng dụng: Các dự án IoT phổ thông như điều khiển thiết bị qua Internet, nhà thông minh, cảm biến nhiệt độ – độ ẩm, cảnh báo an ninh, xây dựng web server đơn giản hoặc dùng trong các bài thử nghiệm cần nhiều chân GPIO.

ESP8266 TYPE-C || Micro



Hình 1.10 ESP8266 NodeMCU

d) ESP8266 Wemos D1 mini:

Wemos D1 Mini là bản thu nhỏ của ESP8266, rất nhỏ gọn và phù hợp để gắn trực tiếp vào các sản phẩm IoT. Dù ít chân hơn NodeMCU, nó vẫn đáp ứng tốt hầu hết cảm biến và module IoT thông dụng.

Ưu điểm: Kích thước siêu nhỏ, dễ tích hợp vào mạch in, giá rẻ và tiêu thụ năng lượng thấp.

Nhược điểm: Ít chân GPIO và thao tác trên breadboard kém thuận tiện hơn.

Ứng dụng: Thiết bị IoT mini như cảm biến môi trường, công tắc thông minh, bộ điều khiển LED nhỏ gọn, wearable hoặc các dự án cần tiết kiệm tối đa không gian và năng lượng.



Hình 1.11 ESP8266 Wemos D1 Mini

e) Raspberry Pi:

Raspberry Pi là một máy tính thu nhỏ chạy Linux, mạnh hơn nhiều so với các vi điều khiển. Nó có thể xử lý đa nhiệm, chạy giao diện đồ họa, kết nối camera, màn hình, USB, mạng LAN/WiFi và đảm nhiệm những tác vụ phức tạp trong IoT hoặc hệ thống thông minh.

Ưu điểm: Hiệu năng mạnh, hỗ trợ nhiều thiết bị ngoại vi, chạy được server và các ứng dụng xử lý ảnh, AI hoặc dữ liệu lớn.

Nhược điểm: Giá cao hơn, tiêu thụ điện lớn và yêu cầu cài đặt phức tạp hơn so với Arduino hay ESP.

Ứng dụng: Hệ thống IoT nâng cao, camera giám sát, nhận dạng hình ảnh, smart home, robot, server web/MQTT hoặc các bài toán cần khả năng tính toán cao.



Hình 1.12 Raspberry Pi

1.5.2. Thiết bị cảm biến

Cảm biến là các thiết bị quan trọng trong hệ thống IoT, có vai trò thu thập dữ liệu từ môi trường hoặc thiết bị, chuyển đổi thành tín hiệu điện (analog hoặc digital) để vi điều khiển xử lý và ra quyết định. Thông qua cảm biến, hệ thống IoT có thể giám sát, đo lường và điều khiển tự động các thông số vật lý, hóa học, môi trường hoặc trạng thái của các thiết bị khác.

Các loại cảm biến phổ biến trong hệ thống IoT bao gồm:

a) Cảm biến nhiệt độ - độ ẩm (DHT11 – DHT22):

DHT11 và DHT22 là cảm biến đo nhiệt độ và độ ẩm phổ biến trong IoT, cho dữ liệu dạng digital và dễ kết nối với các vi điều khiển.

Ưu điểm: Dễ lập trình, giá rẻ, dữ liệu khá ổn định, DHT22 cho độ chính xác cao hơn.

Nhược điểm: Tốc độ phản hồi chậm và không phù hợp môi trường ẩm ướt kéo dài.

Ứng dụng: Giám sát môi trường, nhà thông minh, hệ thống HVAC, cảnh báo nhiệt độ - độ ẩm.



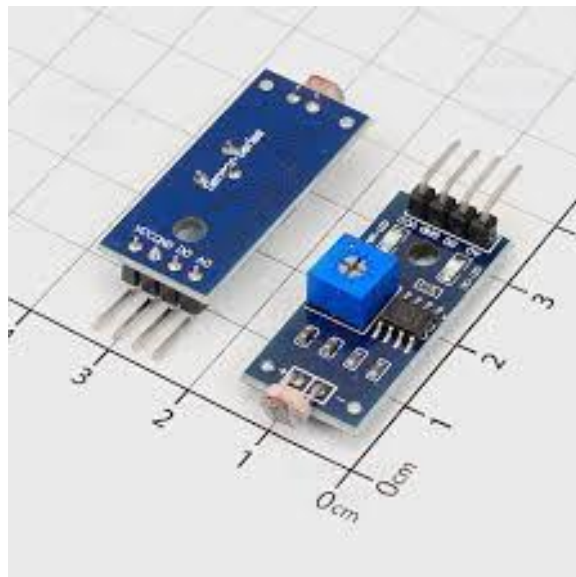
Hình 1.13 Cảm biến nhiệt độ - độ ẩm DHT22

b) Cảm biến ánh sáng (LDR, BH1750):

Cảm biến ánh sáng dùng để đo cường độ sáng môi trường, thường gặp ở hai loại: LDR với tín hiệu analog và BH1750 với tín hiệu digital qua I2C.

Ưu điểm: Dễ tích hợp; LDR rất rẻ, còn BH1750 cho độ chính xác cao và đọc dữ liệu ổn định.

Nhược điểm: LDR dễ nhiễu và không tuyến tính; BH1750 cần giao tiếp I2C nên phức tạp hơn một chút.



Hình 1.14 Cảm biến ánh sáng

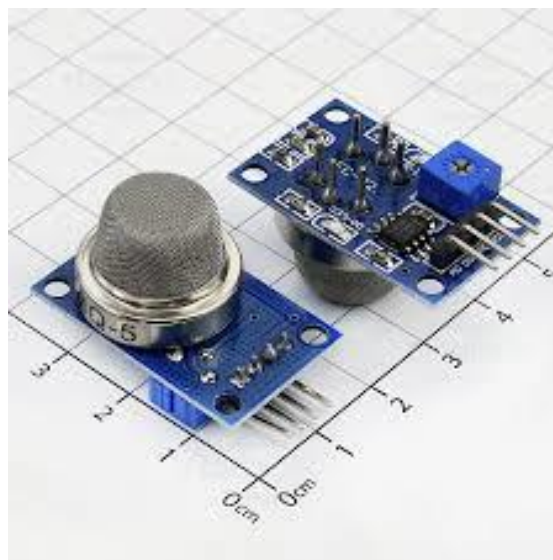
c) Cảm biến khí (MQ, CCS811):

Cảm biến khí được dùng để đo nồng độ các loại khí độc hại hoặc VOC trong không khí, với dạng analog (như MQ series) hoặc digital qua I2C (như CCS811).

Ưu điểm: Đo được nhiều loại khí, độ nhạy cao và dễ kết nối với vi điều khiển.

Nhược điểm: Cần hiệu chuẩn định kỳ và dễ bị ảnh hưởng bởi môi trường xung quanh.

Ứng dụng: Giám sát chất lượng không khí, cảnh báo khí độc, hệ thống nhà thông minh và các dự án IoT môi trường.



Hình 1.15 Cảm biến khí MQ

d) Cảm biến siêu âm (HC-SR04):

Cảm biến siêu âm là cảm biến đo khoảng cách bằng sóng siêu âm, cho kết quả digital và đo được vật thể mà không cần tiếp xúc.

Ưu điểm: Giá rẻ, đo khá chính xác, dễ dùng cho robot và các hệ thống IoT.

Nhược điểm: Khó đo vật mỏng hoặc mềm, dễ sai lệch trong môi trường nhiễu âm.

Ứng dụng: Robot tránh vật cản, đo mực nước, giám sát bồn chứa và hệ thống tự động hóa.



Hình 1.16 Cảm biến siêu âm HC-SR04

1.5.3. Thiết bị cảnh báo

Thiết bị cảnh báo là các module dùng để thông báo cho người dùng khi hệ thống IoT phát hiện sự cố hoặc giá trị cảm biến vượt ngưỡng an toàn. Chúng giúp tăng tính an toàn và giám sát thời gian thực. Trong các dự án IoT, có thể sử dụng các thiết bị như Relay cảnh báo đèn, còi điện tử, LED ma trận động, hoặc thông báo qua màn hình và ứng dụng di động.

a) Còi điện tử/Piezo Buzzer động:

Còi điện tử được dùng để phát âm thanh cảnh báo trong các hệ thống IoT và tự động hóa. Khi được kích hoạt, nó phát tiếng bíp hoặc âm liên tục tùy lập trình.

Ưu điểm: Âm thanh lớn, dễ nghe và rất dễ tích hợp vào các mạch điều khiển.

Nhược điểm: Chỉ báo bằng âm thanh và có thể gây khó chịu nếu hoạt động lâu.

Ứng dụng: Cảnh báo cháy, trộm, vượt ngưỡng cảm biến, báo hiệu trong nhà thông minh và hệ thống công nghiệp.



Hình 1.17 Piezo Buzzer

b) Đèn LED ma trận / LED RGB động:

LED ma trận được dùng để hiển thị cảnh báo bằng chữ, biểu tượng hoặc hiệu ứng ánh sáng, giúp thông tin trực quan hơn trong các hệ thống IoT.

Ưu điểm: Hiển thị rõ ràng, dễ tạo hiệu ứng và thể hiện trạng thái hệ thống.

Nhược điểm: Lập trình phức tạp hơn LED thường và tiêu thụ điện nhiều hơn.

Ứng dụng: Hiển thị chữ chạy, cảnh báo trạng thái thiết bị, các tín hiệu trong IoT công nghiệp và môi trường.



Hình 1.18 LED RGB động

1.5.4. Thiết bị hiển thị

a) LCD 16x2 / 20x4:

LCD là màn hình hiển thị ký tự, dùng để hiện thông tin cơ bản như số liệu hoặc trạng thái trong các dự án IoT.

Ưu điểm: Giá rẻ, dễ lập trình và hiển thị thông tin trực tiếp trên thiết bị.

Nhược điểm: Chỉ hiển thị ký tự, không phù hợp với hình ảnh hay biểu đồ phức tạp, kích thước hạn chế.

Ứng dụng: Giám sát nhiệt độ, độ ẩm, áp suất, các thiết bị IoT, nhà thông minh và mạch nhúng.



Hình 1.19 LCD 20x4

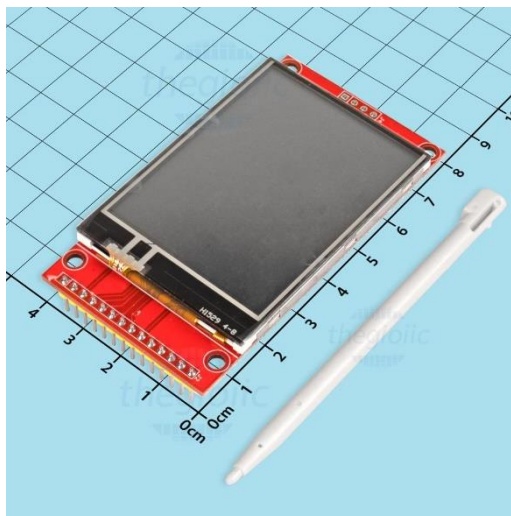
b) Màn hình TFT/LCD màu:

Màn hình màu dùng để hiển thị hình ảnh, đồ thị và giao diện trực quan trong các dự án IoT nâng cao.

Ưu điểm: Trực quan, hiển thị nhiều thông tin cùng lúc và hỗ trợ đồ họa, biểu đồ realtime.

Nhược điểm: Giá cao hơn LCD ký tự, tiêu thụ điện nhiều và lập trình phức tạp hơn.

Ứng dụng: Hiển thị dữ liệu cảm biến, hệ thống IoT nâng cao, smart home/farm, giám sát công nghiệp và máy móc tự động.



Hình 1.20 Màn hình LCD màu

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 nước sinh hoạt theo thời gian thực, giúp người dùng quan sát được sự thay đổi của chất lượng nước và nhanh chóng phát hiện các dấu hiệu bất thường. Trong thực tế, hệ thống đóng vai trò như một công cụ theo dõi tự động: nó thu thập dữ liệu từ nước sinh hoạt, hiển thị thông tin cho người sử dụng, đồng thời đưa ra cảnh báo khi giá trị vượt quá ngưỡng an toàn. Nhờ đó, người dùng có thể kịp thời xử lý hoặc điều chỉnh điều kiện môi trường theo nhu cầu.

Từ những chức năng thực tế nêu trên, hệ thống được thiết kế với các mô-đun phần cứng và phần mềm phù hợp để đảm bảo hoạt động chính xác, ổn định. Các yêu cầu chức năng chính của hệ thống bao gồm:

2.1.1. Thu thập dữ liệu từ các cảm biến

Chức năng thu thập dữ liệu cho phép hệ thống ghi nhận liên tục các thông số nước sinh hoạt như pH, nhiệt độ và TDS. Trong quá trình hoạt động, hệ thống tiến hành đo các chỉ số này theo thời gian thực, đảm bảo phản ánh chính xác tình trạng hiện tại của nguồn nước. Dữ liệu thu thập được được cập nhật liên tục, giúp người dùng theo dõi biến động theo thời gian và phát hiện sớm những dấu hiệu bất thường.

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

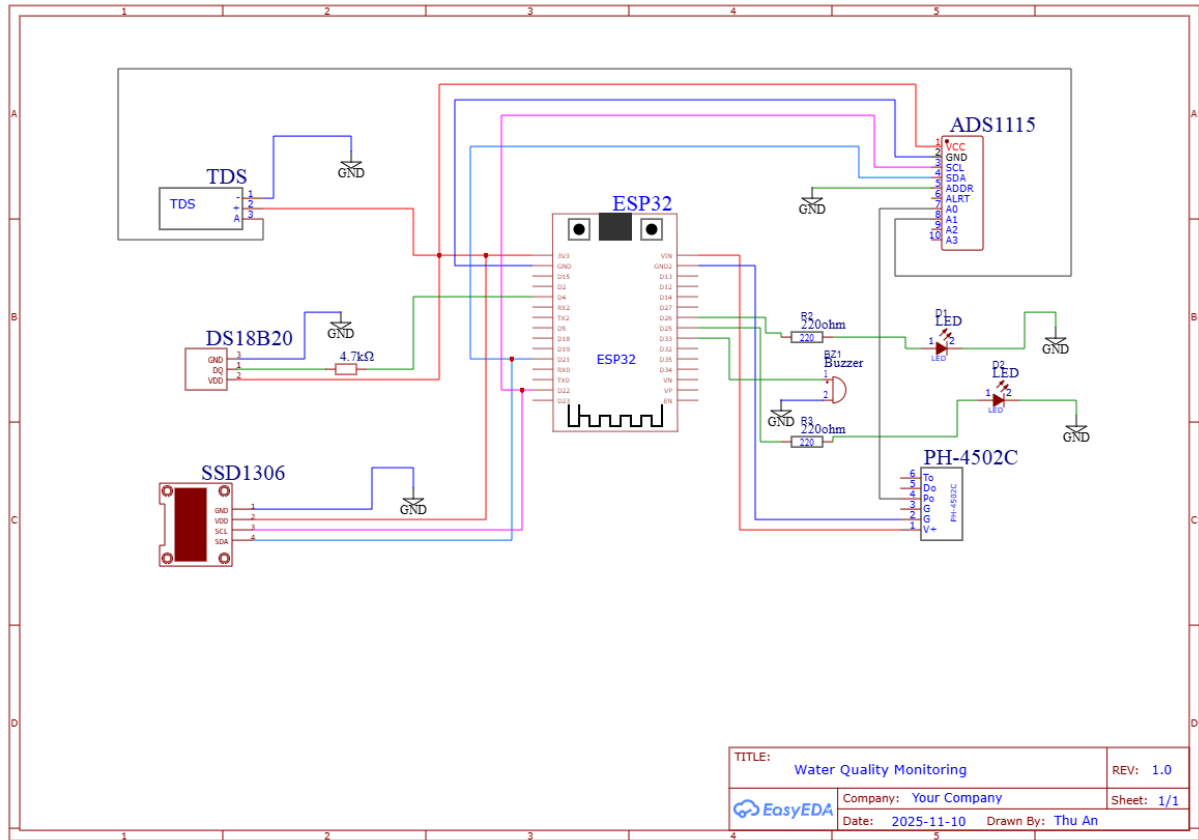
Sau khi dữ liệu được thu thập, hệ thống tiến hành hiển thị các thông số một cách trực quan. Trên màn hình tại vị trí lắp đặt, các thông số quan trọng được trình bày rõ ràng, giúp người dùng nhanh chóng nắm bắt tình trạng nguồn nước. Đồng thời, dữ liệu cũng được gửi lên ứng dụng di động Blynk, cho phép người dùng giám sát từ xa, và so sánh với mức chuẩn. Việc hiển thị liên tục và trực quan giúp nâng cao khả năng theo dõi, quản lý và ra quyết định kịp thời khi có những biến động bất thường.

2.1.3. Thiết lập cảnh báo khi giá trị vượt ngưỡng

Hệ thống cho phép người dùng thiết lập các ngưỡng an toàn cho từng thông số nước trên giao diện điện thoại. Khi các giá trị đo được vượt quá mức cho phép, hệ thống sẽ phát cảnh báo kịp thời, thông báo trực tiếp đến người dùng. Chức năng này giúp người dùng chủ động theo dõi chất lượng nước, nhận biết những nguy cơ tiềm ẩn và thực hiện các biện pháp xử lý phù hợp. Nhờ khả năng thiết lập và tùy chỉnh ngưỡng, hệ thống trở nên linh hoạt, đáp ứng nhu cầu kiểm soát chất lượng nước theo từng điều kiện thực tế.

2.2. Thiết kế hệ thống

2.2.1. Sơ đồ 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, nhận và xử lý dữ liệu từ các cảm biến, loại bỏ giá trị lỗi, chuyển đổi tín hiệu sang giá trị thực, gửi dữ liệu lên ứng dụng Blynk, đồng thời điều khiển hiển thị trên OLED và thiết bị cảnh báo.

Thiết bị cảm biến:

- Cảm biến pH: đo độ axit/bazơ của nước. Kết nối qua ADS1115 (kênh A0) để chuyển đổi tín hiệu analog sang ESP32.
- Cảm biến TDS: đo độ dẫn điện (ppm) của nước. Kết nối qua ADS1115 (kênh A1)
- Cảm biến nhiệt độ DS18B20: đo nhiệt độ nước (°C), giao tiếp qua bus 1-Wire tại GPIO4.
- ADS1115: module ADC 16-bit giúp chuyển đổi tín hiệu analog từ cảm biến pH và TDS sang dạng số chính xác cho ESP32

Thiết bị hiển thị:

OLED SSD1306: hiển thị các thông số pH, TDS, nhiệt độ, trạng thái Wifi, thông báo “Nước bình thường” hoặc “Vượt ngưỡng”.

Thiết bị cảnh báo:

- LED đỏ / xanh: báo trạng thái nước bình thường hay vượt ngưỡng.
- Còi: Cảnh báo âm thanh khi các chỉ số vượt ngưỡng.

2.2.1.1. Vi điều khiển ESP32

ESP32 là một series các vi điều khiển trên một vi mạch giá rẻ, năng lượng thấp có tích hợp Wifi và dual-mode Bluetooth (tạm dịch: Bluetooth chế độ kép). Dòng ESP32 sử dụng bộ vi xử lý Tensilica Xtensa LX6 có hai biến thể lõi kép và lõi đơn, và bao gồm các công tắc anten tích hợp, RF balun, bộ khuếch đại công suất, bộ khuếch đại thu nhiễu thấp, bộ lọc và module quản lý năng lượng.

ESP32 được chế tạo và phát triển bởi Espressif Systems, một công ty Trung Quốc có trụ sở tại Thượng Hải, và được sản xuất bởi TSMC bằng cách sử dụng công nghệ 40nm. ESP32 là sản phẩm kế thừa từ vi điều khiển ESP8266.

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:

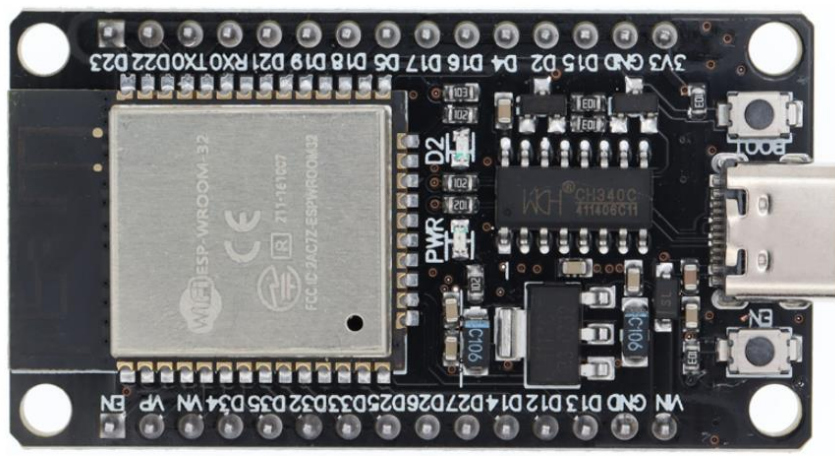
- Cảm biến Hall (đo từ trường)
- Cảm biến nhiệt độ nội chip
- 10 cảm biến chạm điện dung (touch sensor)

Bảo mật phần cứng:

- Hỗ trợ WPA/WPA 2/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



Hình 2.2 Vi xử lý ESP32

2.2.1.2. Cảm biến ADS1115

ADS1115 trong hệ thống IoT giám sát chất lượng nước giúp chuyển tín hiệu analog từ cảm biến pH và TDS sang số, giúp ESP32 đọc và xử lý chính xác dữ liệu để hiển thị, lưu trữ và cảnh báo kịp thời.

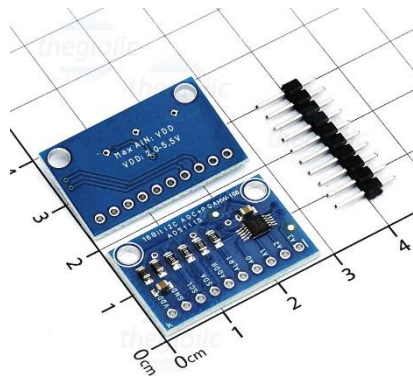
Đây là mạch chuyển đổi ADC 16-bit 4 kênh tích hợp bộ khuếch đại, hoạt động với nguồn 2–5,5V, đo được nhiều loại tín hiệu và thiết kế nhỏ gọn, dễ sử dụng, phù hợp cho các dự án vi điều khiển cần độ chính xác cao.

Thông số kỹ thuật:

- Tốc độ bit: 16 Bit.
- Kiểu giao tiếp: I2C.
- Kênh: 4 kênh AN0 AN1 AN2 AN3 hoặc 2 đầu vào riêng biệt.
- Điện áp vào: 2 ~ 5.5VDC.
- Dòng tiêu thụ: Chỉ 150uA.
- Đầu vào điều khiển chương trình, 7 loại đầu vào khoảng: -0.256V ~ 0.256V, -0.512V ~ 0.512V, -1.024V ~ 1.024 V, -2.048V ~ + 2.048V, -4.096V ~ 4.096 V, -6.144V ~ + 6.144V.
- Kích thước: 18 x 28mm.

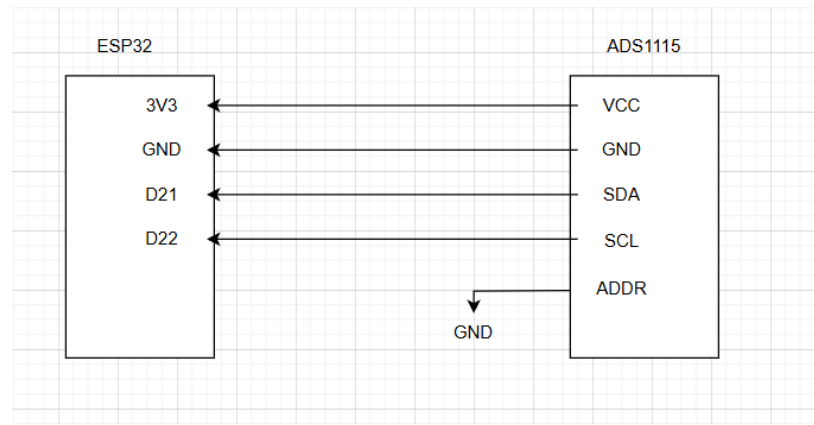
Ứng dụng:

- Đo các tín hiệu analog chính xác trong dự án vi điều khiển.
- Hệ thống đo điện áp, cảm biến điện tử, thiết bị đo lường.
- Kết hợp với Arduino, Raspberry Pi hoặc các vi điều khiển khác để mở rộng số kênh ADC.
- Các dự án cần độ chính xác cao và dải đo linh hoạt.



Hình 2.3 Cảm biến ADS1115

Sơ đồ khối của cảm biến ADS1115 với ESP32:



Hình 2.4 Sơ đồ khối của cảm biến ADS1115

2.2.1.3. Cảm biến TDS (Total Dissolved Solids sensor)

Trong hệ thống IoT giám sát chất lượng nước sinh hoạt, cảm biến TDS được sử dụng để đo liên tục độ dẫn điện của nước, từ đó xác định tổng chất rắn hòa tan và cung cấp dữ liệu chính xác cho hiển thị, lưu trữ và cảnh báo.

TDS (Total Dissolved Solids), hay tổng chất rắn hòa tan, là chỉ số dùng để đánh giá chất lượng nước, phản ánh hàm lượng các chất hữu cơ và vô cơ có trong nước.

Đơn vị đo thường là mg/l (milligrams/liter) hoặc ppm (parts per million).

Chỉ số TDS thấp cho thấy nước sạch, nhưng nếu quá thấp, nước có thể thiếu khoáng chất; ngược lại, TDS cao không đồng nghĩa với nước bẩn, vì đây chỉ là chỉ số tổng hợp về các hợp chất hóa học, không phải chỉ số ô nhiễm.

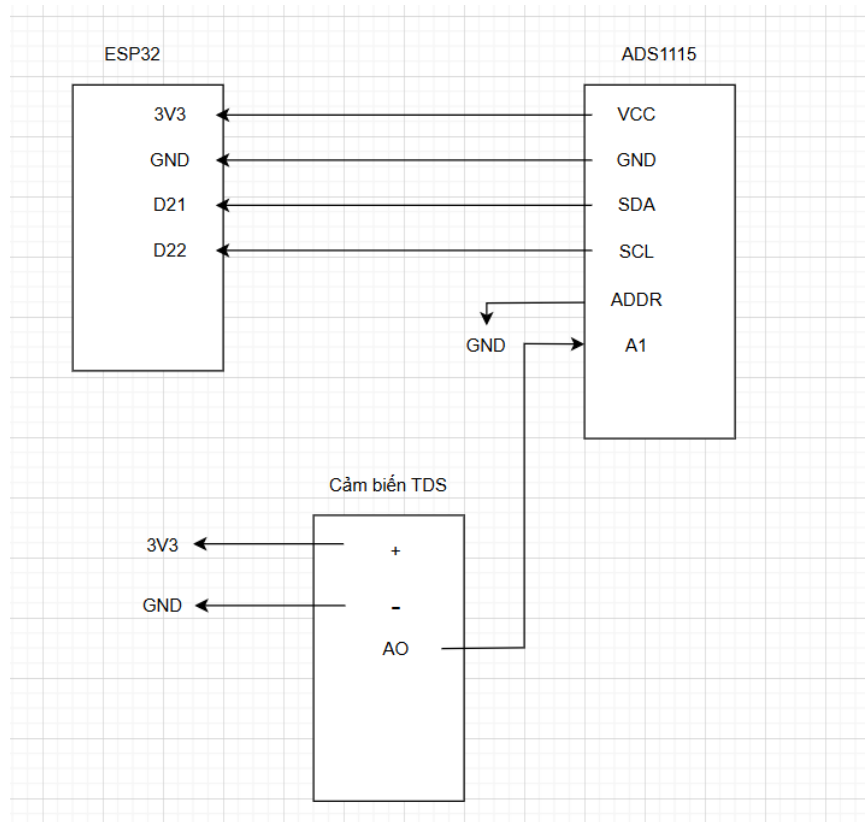
Thông số kỹ thuật:

- Điện áp hoạt động: 3.3 V - 5.5 V
- Dòng điện tiêu thụ: 3 ~ 6 mA
- Phạm vi đo TDS: 0 ~ 1000 ppm
- Độ chính xác đo TDS: $\pm 10\%$ f. S. (25 độ C)



Hình 2.5 Cảm biến TDS (Total Dissolved Solids sensor)

Sơ đồ khối của TDS với ESP32 qua ADS1115:



Hình 2.6 Sơ đồ khối của cảm biến TDS

2.2.1.4. Cảm biến pH

Cảm biến đo độ pH là thiết bị điện tử được sử dụng để đo độ pH của một môi trường nước sinh hoạt. Độ pH là chỉ số biểu thị độ axit hay bazơ của dung dịch, đo trên thang từ 0 đến 14:

- pH < 7: Môi trường axit
- pH = 7: Môi trường trung tính
- pH > 7: Môi trường bazơ

Cảm biến này thường được dùng trong phòng thí nghiệm, hệ thống đo môi trường, xử lý nước, và các dự án IoT nông nghiệp hoặc công nghiệp.

Thông số kỹ thuật:

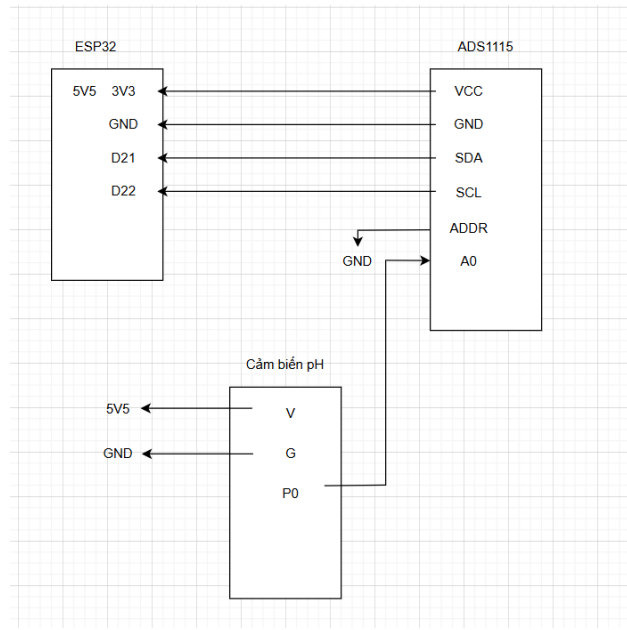
- Nguồn cấp: 5VDC.
- Dòng điện làm việc: 5 ~ 10mA.
- Công suất tiêu thụ: <0.5W.
- Thời gian đáp ứng: ≤5S.
- Thời gian ổn định: ≤60S.
- Tín hiệu trả về: Analog.
- Khoảng đo pH: 0 -14 pH.
- Khoảng nhiệt độ đo: 0 – 60 độ C.
- Độ chính xác: 0.1 pH (25 độ C).
- Tốc độ phản ứng: < 1 phút.

Lưu ý: Loại đầu dò này chỉ sử dụng trong phòng thí nghiệm, không được ngâm quá lâu trong mẫu chất lỏng cần đo. Đầu dò cần được rửa sạch sau mỗi lần đo và bảo quản trong dung dịch 3M KCL.



Hình 2.7 Cảm biến pH

Sơ đồ khối của pH với ESP32 qua ADS1115:



Hình 2.8 Sơ đồ khối của cảm biến pH

2.2.1.5. Cảm biến nhiệt độ DS18B20

Trong hệ thống IoT giám sát chất lượng nước sinh hoạt, DS18B20 đo liên tục nhiệt độ nước, cung cấp dữ liệu chính xác để hiển thị, lưu trữ và cảnh báo khi nhiệt độ vượt ngưỡng an toàn.

DS18B20 là cảm biến nhiệt độ 1 dây có thể lập trình, được dùng phổ biến để đo nhiệt độ trong các môi trường khác nhau, bao gồm dung dịch, hàm mỡ hoặc đất.

Cảm biến có phần vỏ chắc chắn, dễ lắp đặt và có thể chọn loại chống thấm nước. Nó đo được nhiệt độ từ -55°C đến 125°C với độ chính xác $\pm 0,5^{\circ}\text{C}$.

Mỗi cảm biến có một địa chỉ duy nhất và chỉ sử dụng một chân vi điều khiển để truyền dữ liệu, thuận tiện khi đo nhiệt độ nhiều vị trí.

Thông số kỹ thuật:

- Nguồn đầu vào: 3.3V.
- Dải đo nhiệt độ: -55 đến 125°C (-67 đến 257°F).
- Sai số: $\pm 0.5^{\circ}\text{C}$ khi đo ở dải $-10 - 85^{\circ}\text{C}$.
- Độ phân giải: người dùng có thể chọn từ 9 – 12 bits.
- Chuẩn giao tiếp: 1-Wire (1 dây).
- Có cảnh báo nhiệt khi vượt ngưỡng cho phép và cấp nguồn từ chân data.
- Thời gian chuyển đổi nhiệt độ tối đa : 750ms (khi chọn độ phân giải 12bit).

- Mỗi IC có một mã riêng (lưu trên EEPROM của IC) nên có thể giao tiếp nhiều DS18B20 trên cùng 1 dây.
- Ống thép không gỉ (chống ẩm, nước) đường kính 6mm, dài 50mm
- Đường kính đầu dò: 6mm.
- Chiều dài dây: 1m.

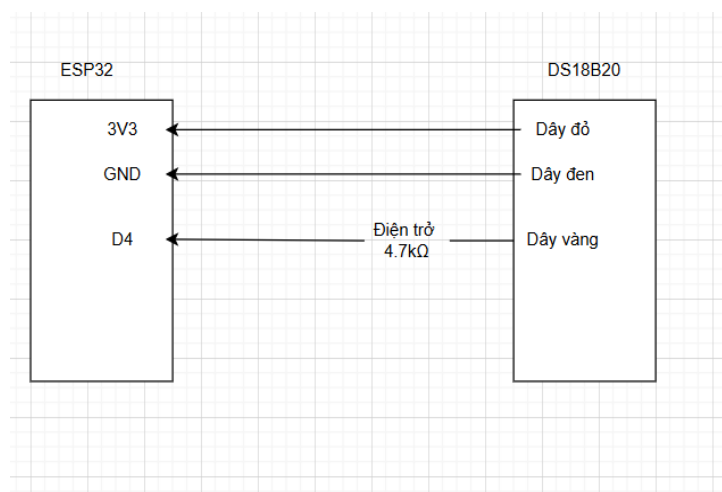
Ứng dụng:

- Giám sát nhiệt độ trong nhà thông minh(điều hòa,lò sưởi,tủ lạnh).
- Đo nhiệt độ nước, đất, môi trường hóa chất trong các dự án IoT nông nghiệp, công nghiệp.
- Sử dụng hệ thống cảnh báo quá nhiệt.
- Các dự án vi điều khiển Arduino, Raspberry Pi cần nhiều cảm biến cùng một bus.



Hình 2.9 Cảm biến nhiệt độ DS18B20

Sơ đồ khối của cảm biến nhiệt độ DS18B20 với ESP32:



Hình 2.10 Sơ đồ khối của cảm biến nhiệt độ DS18B20

2.2.1.6. OLED SSD1306

Trong hệ thống IoT giám sát chất lượng nước sinh hoạt, OLED dùng để hiển thị các thông số pH, TDS, nhiệt độ và trạng thái nước, giúp người dùng quan sát dữ liệu trực tiếp tại vị trí lắp đặt.

Màn hình OLED 0.96 inch của AZ Delivery có kích thước rất nhỏ với đường chéo màn hình 3 cm và dễ đọc nhờ độ tương phản cao. Màn hình bao gồm 128 x 64 điểm ảnh OLED, được điều khiển riêng lẻ bởi chip SSD1306 tích hợp.

Tính năng đặc biệt:

- Màn hình được điều khiển thông qua giao diện I2C và chỉ cần 4 kết nối.
- Kết hợp với hai thư viện Adafruit GFX và Adafruit SSD1306, cũng chứa các bản phác thảo mẫu, màn hình có thể được lập trình nhanh chóng.

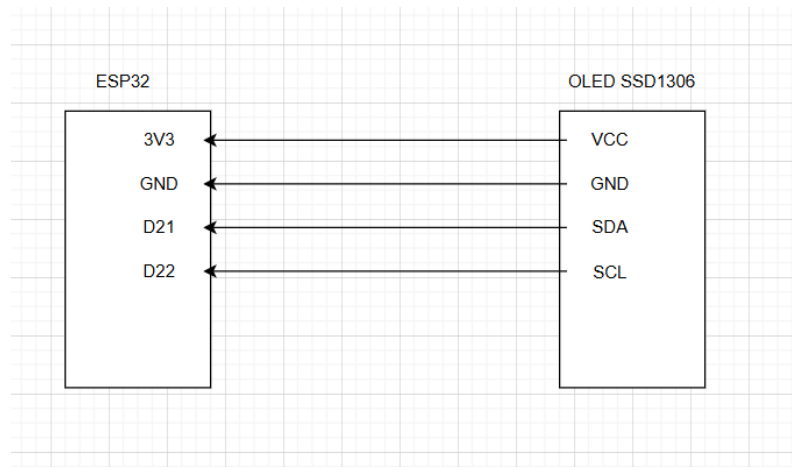
Thông số kỹ thuật:

- Bộ vi xử lý: SSD1306.
- Màu sắc: Trắng.
- Góc nhìn thấy được: $> 160^\circ$.
- Nghị quyết: 128 x 64 pixel.
- Điện áp hoạt động: 3.3V đến 5V.
- Nhiệt độ hoạt động: -30°C đến $+80^\circ\text{C}$.
- Tiêu thụ điện năng thấp: 0,04W khi hoạt động bình thường.
- Điều khiển: I2C/TWI.
- Gán mã pin: VCC, GND, SCL, SDA.
- Kích thước: xấp xỉ 27mm x 27mm x 4,1mm.



Hình 2.11 Màn hình OLED 1306

Sơ đồ khối của màn hình OLED SSD1306 với ESP32:



Hình 2.12 Sơ đồ khối của OLED 1306

2.2.1.7. Buzzer(Còi)

Trong hệ thống IoT giám sát chất lượng nước sinh hoạt, buzzer dùng để phát cảnh báo âm thanh khi các thông số nhiệt độ, TDS, pH vượt ngưỡng an toàn, giúp người dùng kịp thời nhận biết và xử lý tình trạng nước.

Còi Buzzer 5VDC là một thiết bị phát âm thanh điện tử nhỏ gọn, hoạt động với điện áp một chiều 5V (DC). Thiết bị này được sử dụng rộng rãi trong các mạch điện tử, hệ thống cảnh báo, báo hiệu và các ứng dụng nhúng như:

- Mạch báo cháy, báo trộm, báo nước, báo quá nhiệt.
- Thiết bị điện tử, đồng hồ điện tử, robot.
- Hệ thống điều khiển Arduino, Raspberry Pi, ESP32, STM32,...

Còi Buzzer 5VDC có tuổi thọ cao, hiệu suất ổn định, chất lượng tốt, được sản xuất nhỏ gọn phù hợp thiết kế với các mạch còi buzzer nhỏ gọn, mạch báo động.

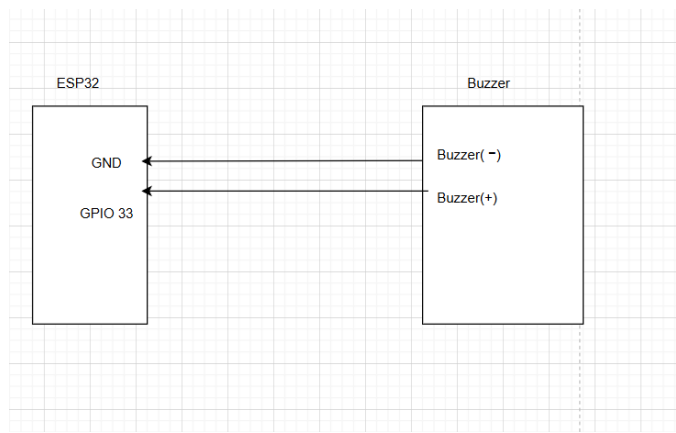
Thông số kỹ thuật:

- Nguồn : 3.5V - 5.5V.
- Dòng điện tiêu thụ: < 25 mA.
- Tần số cộng hưởng: 2300 Hz $\pm$ 500 Hz.
- Biên độ âm thanh: > 80 dB.
- Nhiệt độ hoạt động: -20 °C đến +70 °C.
- Kích thước: Đường kính 12mm, cao 9,7mm.



Hình 2.13. Còi Buzzer

Sơ đồ khối của Buzzer với ESP32



Hình 2.14 Sơ đồ khối của Buzzer

2.2.1.8. Đèn LED(Light Emitting Diode)

Trong hệ thống IoT giám sát chất lượng nước sinh hoạt, LED được dùng để cảnh báo hiệu trạng nước, LED xanh báo nước bình thường, LED đỏ báo vượt ngưỡng, giúp người dùng dễ dàng nhận biết tình trạng nước ngay tại chỗ.

LED (viết tắt của Light Emitting Diode, có nghĩa là điốt phát quang) là các điốt có khả năng phát ra ánh sáng hay tia hồng ngoại, tử ngoại. Cũng giống như điốt, LED được cấu tạo từ một khối bán dẫn loại p ghép với một khối bán dẫn loại n.

Thông số kỹ thuật:

- Chiều dài chân : >20mm.
- Đường kính : 5mm.
- Điện áp tham chiếu : 2.8-3.2V.
- Màu sắc hiển thị : đỏ, vàng, xanh lá, xanh dương.
- Dòng : 5mA – 40mA (Khuyến dùng 10-20mA là bền).

- Độ sáng: 2000 – 4000 mcd.

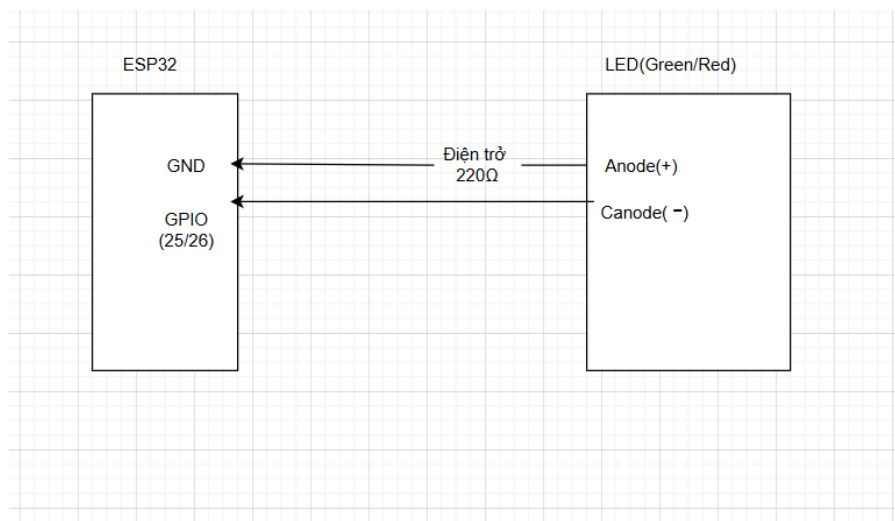
Ứng dụng:

- Ứng dụng đèn LED trong quảng cáo ngoài trời.
- Ứng dụng trong chiếu sáng và trang trí.
- Các bảng hiệu quảng cáo đã có sự phát triển với sự đa dạng về chủng loại cũng như phong cách. Nổi bật trong số đó là sự xuất hiện của đèn LED được các nhà quảng cáo rất “trọng dụng”.



Hình 2.15 Đèn LED

Sơ đồ khối của LED với ESP32



Hình 2.16 Sơ đồ khối của LED

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: Tải Arduino IDE

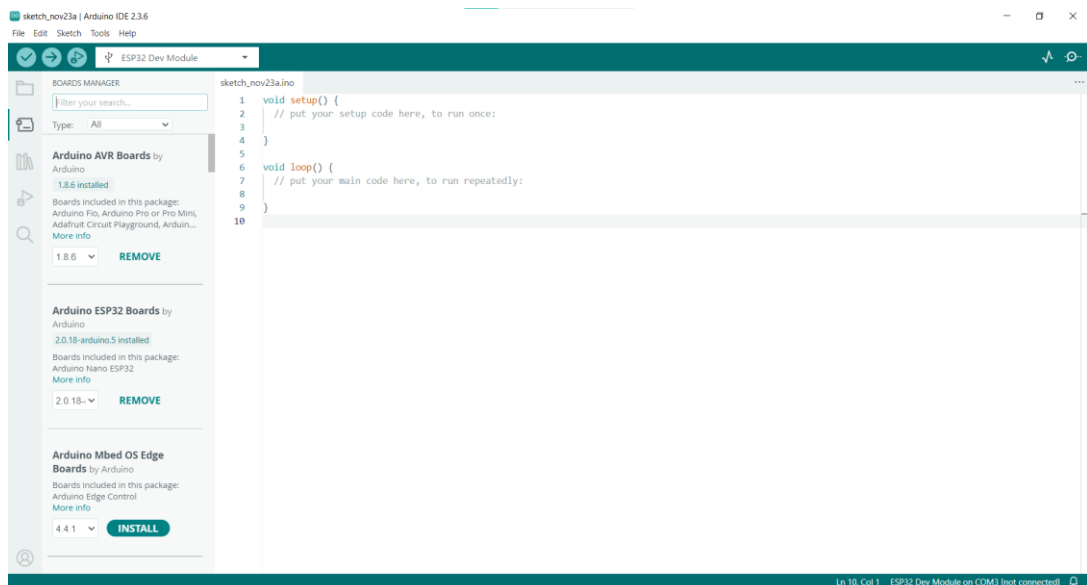
- Truy cập: <https://www.arduino.cc/en/software/>
- Chọn phiên bản tương thích với hệ điều hành: Windows, macOS, Linux và tải về.

Bước 2: Cài đặt Arduino IDE

- Windows: mở file .exe, nhấn Next → I Agree → Next → Install → Finish.
- macOS: Kéo Arduino IDE vào Applications.
- Linux: Giải nén file .tar.xz, mở Terminal và chạy ./install.sh.

Bước 3: Cài đặt các thư viện cần thiết.

- Vào Tools → Manage Libraries.
- Tìm và cài đặt các thư viện cần thiết cho dự án.



Hình 2.17 Trình soạn thảo và thư viện của Arduino IDE

2.2.2.2. Thiết lập môi trường ESP32 trong Arduino IDE

Hướng dẫn thiết lập môi trường ESP32 trong Arduino IDE:

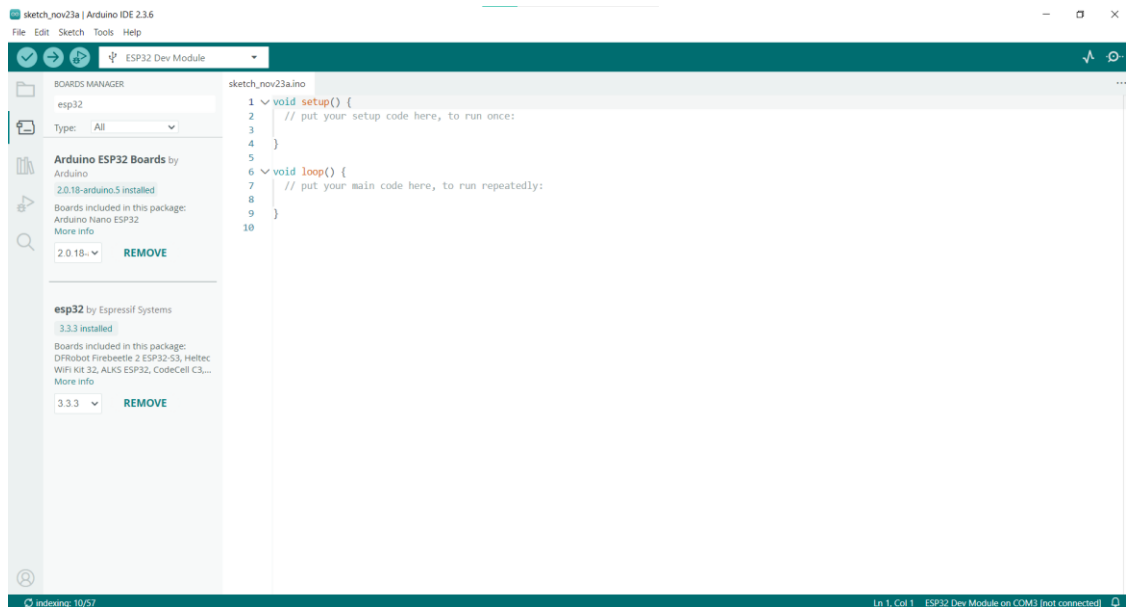
Bước 1: Thêm ESP32 Board vào Arduino IDE.

- 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.

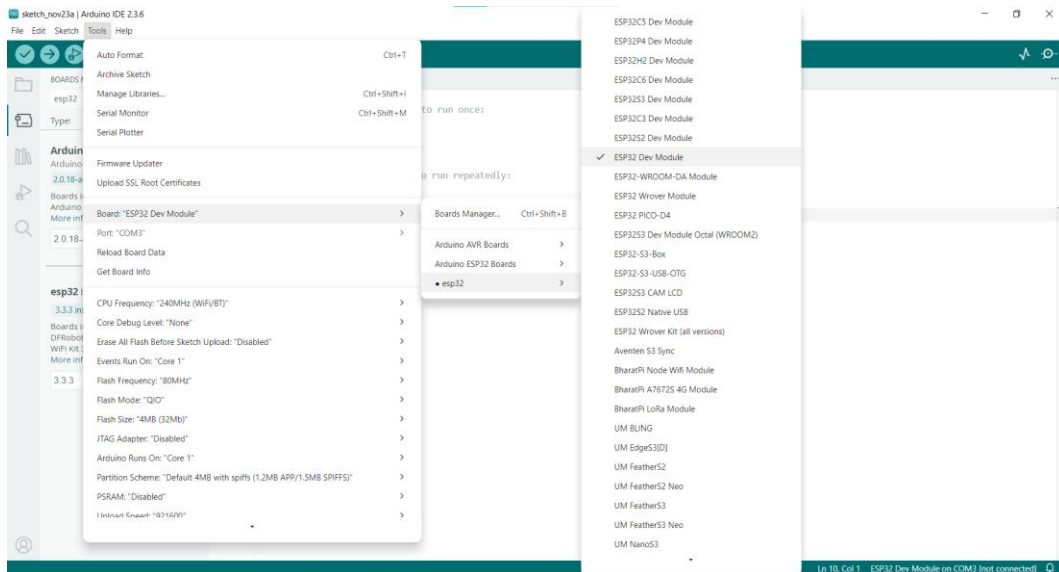
- Vào Tools → Board → Board Manager.
- Trong ô tìm kiếm gõ “esp32”.
- Chọn esp32 by Espressif Systems, nhấn Install.
- Chờ quá trình cài đặt hoàn tất.



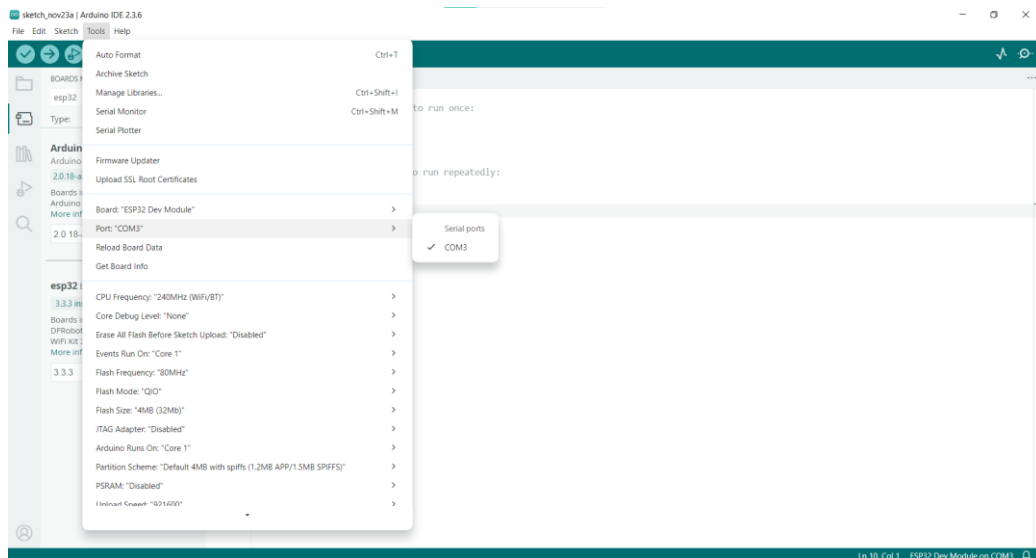
Hình 2.18 Cài đặt ESP32 Board

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 2.19 Chọn board ESP32



Hình 2.20 Chọn cổng COM

2.2.3. Thiết lập ứng dụng trên Blynk.

Hướng dẫn thiết lập ứng dụng trên Blynk:

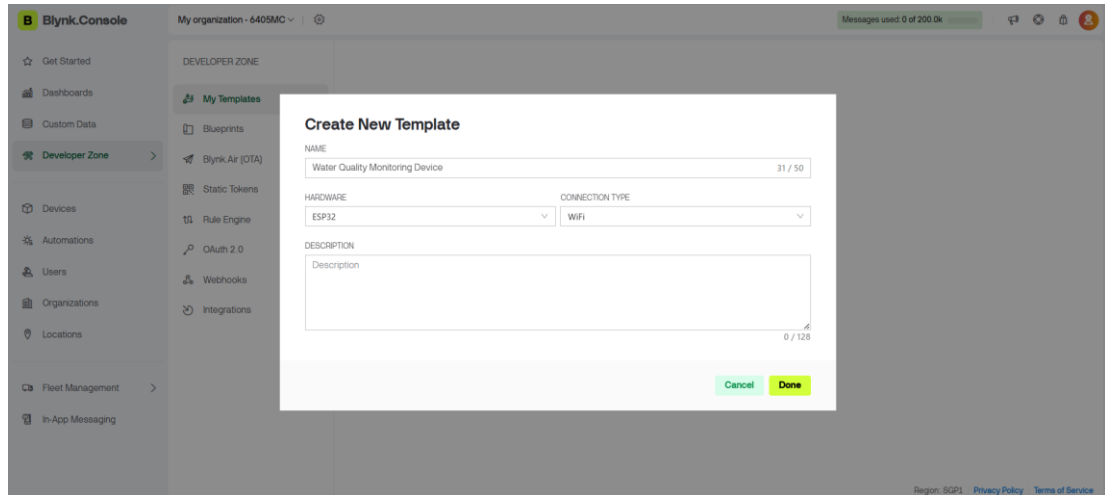
Bước 1: Tạo tài khoản Blynk IoT.

- Mở trình duyệt, truy cập: <https://blynk.io/> .
- Nhấn Sign Up để đăng ký tài khoản mới và Log In nếu đã có tài khoản.

Bước 2: Tạo Template

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

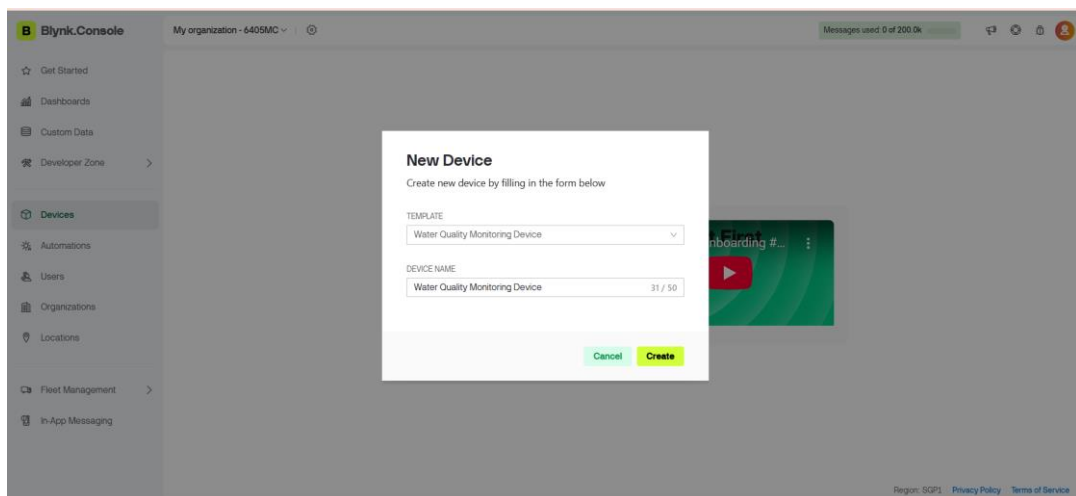
- Template Name: Water Quality Monitoring Device.
- Hardware: ESP32 Dev Board.
- Connection Type: Wifi.
- Nhấn Done để tạo Template.



Hình 2.21 Tạo Template cho giao diện Blynk

Bước 3: Tạo Device

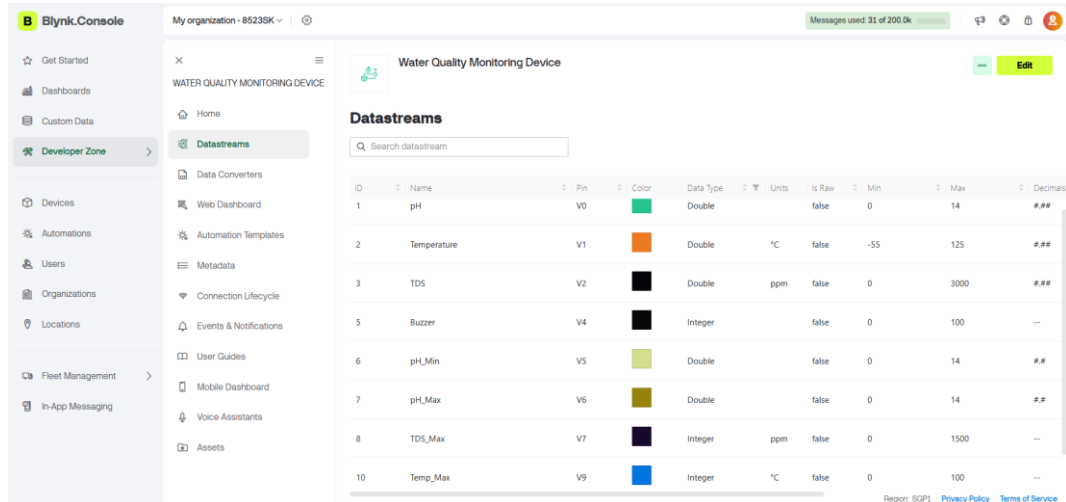
- Vào Devices → + New Device.
- Chọn Template bạn vừa tạo: Water Quality Monitoring Device.
- Điền tên Device: Water Quality Monitoring Device.
- Nhấn Create để tạo Device.
- Blynk sẽ cung cấp Auth Token → lưu lại để ESP32 kết nối với Blynk.
(Auth Token là duy nhất cho từng Device, dùng trong code ESP32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



Hình 2.22 Tạo Device cho giao diện Blynk

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

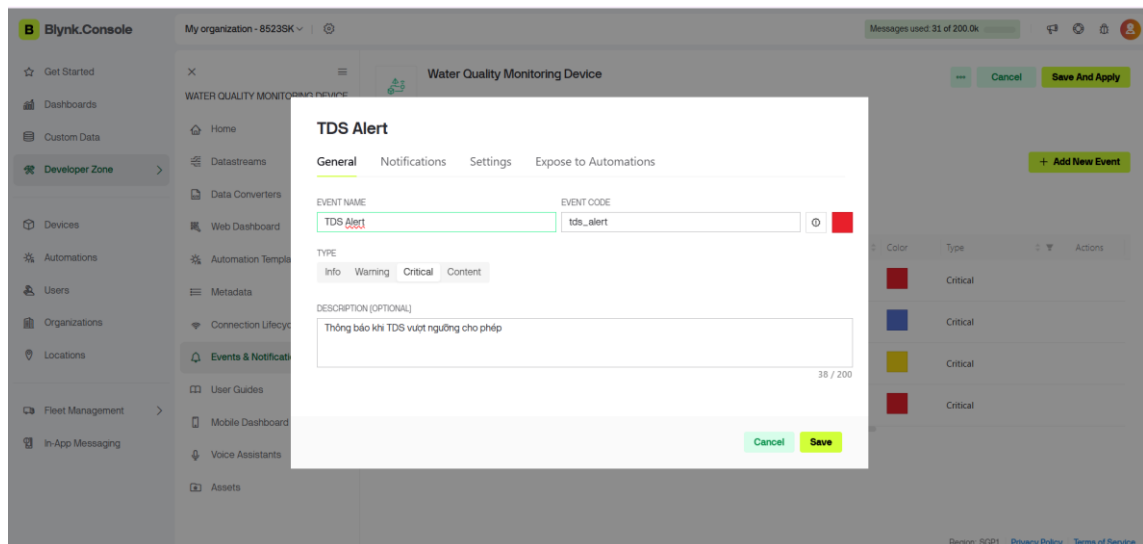
- Trong Developer Zone → nhấn My Template Water Quality Monitoring Device → Datastreams → Edit → New Datastreams để Datastreams cần thiết cho dự án.



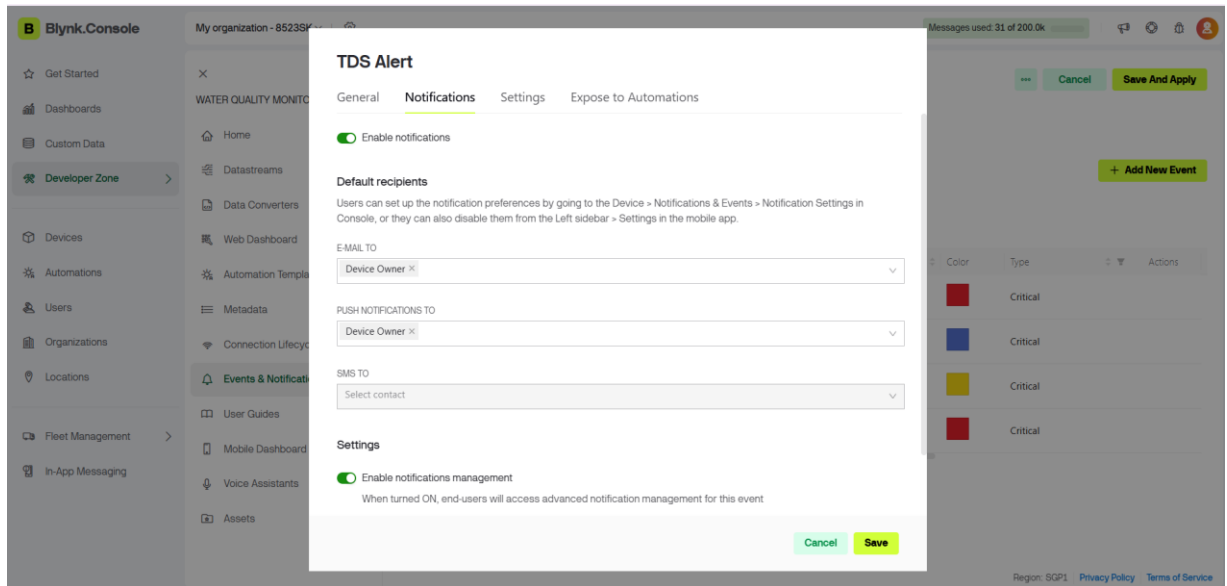
Hình 2.23 Thiết lập Datastream cho giao diện Blynk

Bước 5: Tạo Event cảnh báo

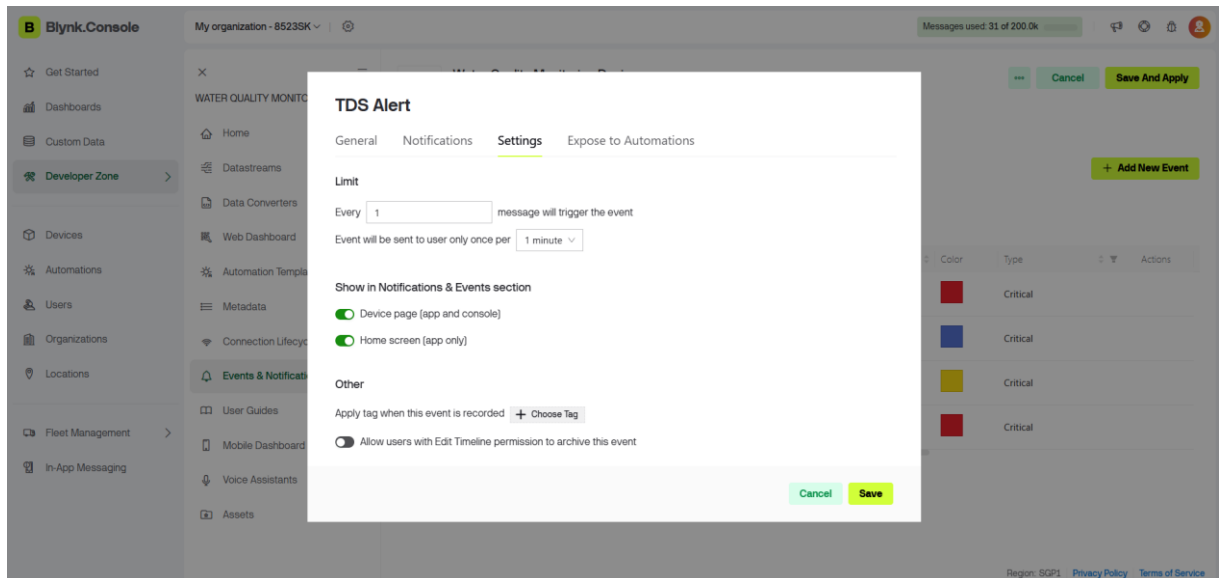
- Trong Developer Zone → nhấn My Template Water Quality Monitoring Device → Events & Notifications → Edit → Create Event → Custom.



Hình 2.24 Tạo event cho hệ thống Blynk

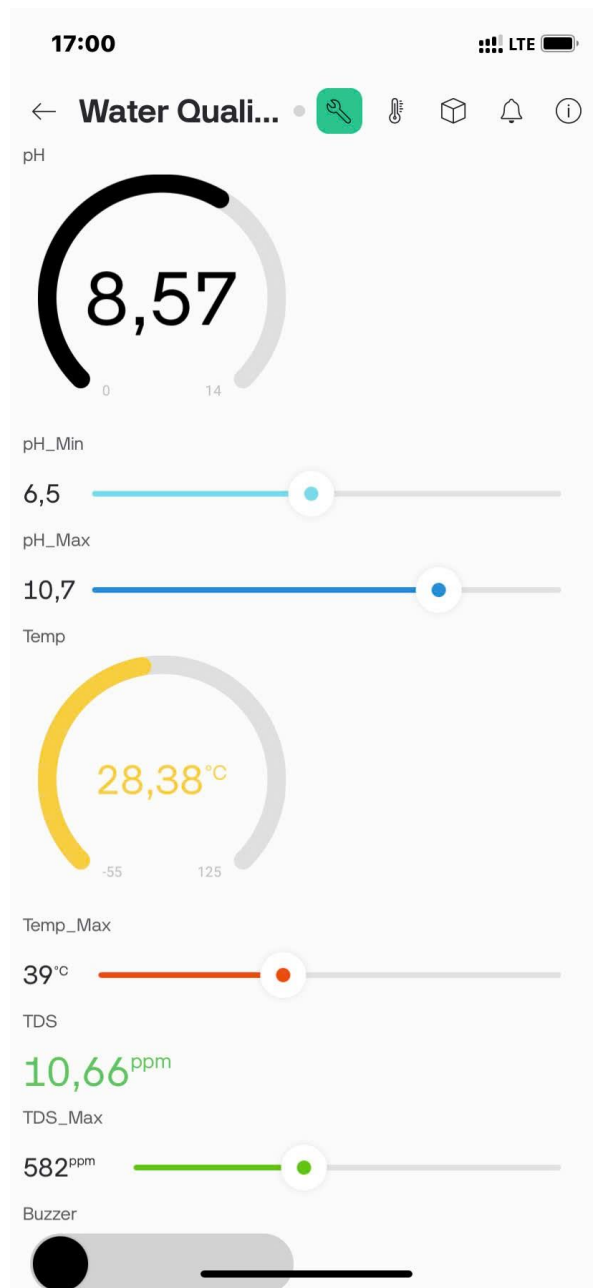


Hình 2.25 Thiết lập thông báo gửi về email và ứng dụng Blynk



Hình 2.26 Cài đặt thời gian gửi của hệ thống trên Blynk

Kết quả: Giao diện hệ thống trên ứng dụng Blynk.



Hình 2.27 Giao diện hệ thống trên App Blynk

2.3. Lập trình hệ thống (Code)

```
// ===== CẤU HÌNH BLYNK =====  
#define BLYNK_TEMPLATE_ID "TMPL6HYzeXEWM"  
#define BLYNK_TEMPLATE_NAME "Water Quality Monitoring Device"  
#define BLYNK_AUTH_TOKEN "fuJ2u7Md7JOf31r4kTSwTcV6w7NYGp9a"
```

```
// ===== THƯ VIỆN=====
#include <Wire.h>
#include <Adafruit_ADS1X15.h>
#include <WiFi.h>
#include <BlynkSimpleEsp32.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <Adafruit_SSD1306.h>

// ===== CẤU HÌNH WIFI =====
char ssid[] = "Tro 43 To Huu";
char pass[] = "43tohuutro";

// ===== CẤU HÌNH ADS1115 =====
Adafruit_ADS1115 ads; // khởi tạo ADC ADS1115

// ===== CẤU HÌNH OLED =====
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1); //
khởi tạo OLED

// ===== CẤU HÌNH DS18B20 =====
#define ONE_WIRE_BUS 4 // GPIO4 kết nối DS18B20
OneWire oneWire(ONE_WIRE_BUS); // khởi tạo giao tiếp 1-Wire
DallasTemperature sensors(&oneWire);

// ===== HẸN GIỜ BLYNK =====
BlynkTimer timer; // Timer gửi dữ liệu lên Blynk
BlynkTimer buzzerTimer; // Timer điều khiển nhấp nháy buzzer

// ===== THÔNG SỐ TDS + pH =====
float TDS_FACTOR = 0.5; // Hệ số tính TDS
float CALIBRATION_FACTOR = 1.0; // Hệ số hiệu chỉnh TDS
float LSB = 4.096F / 32767.0F; // Giá trị điện áp cho 1 đơn vị ADC (gain=1)
float PH_SLOPE = -5.88; // Hệ số tính pH
```

```
float PH_INTERCEPT = 22.32; // Hằng số tính pH

// ===== Ngưỡng có thể chỉnh trên Blynk =====
float pH_min = 6.5; // Ngưỡng pH thấp
float pH_max = 8.5; // Ngưỡng pH cao
int TDS_max = 500; // Ngưỡng TDS tối đa (ppm)
float TEMP_max = 35.0; // Ngưỡng nhiệt độ tối đa (°C)

// ===== PIN LED + BUZZER =====
#define LED_GREEN 25 // LED xanh báo nước bình thường
#define LED_RED 26 // LED đỏ báo vượt ngưỡng
#define BUZZER 33 // Buzzer cảnh báo

bool buzzerState = false; // Trạng thái on/off buzzer
bool buzzerActive = false; // Buzzer cần cảnh báo hay không
bool buzzerEnabled = true; // Cho phép buzzer từ Blynk

// ----- Nhận giá trị ngưỡng từ Blynk -----
BLYNK_WRITE(V5) { pH_min = param.asFloat(); Serial.printf("● pH_min = %.2f\n", pH_min); } // Nhận pH_min từ Blynk
BLYNK_WRITE(V6) { pH_max = param.asFloat(); Serial.printf("● pH_max = %.2f\n", pH_max); } // Nhận pH_max
BLYNK_WRITE(V7) { TDS_max = param.asInt(); Serial.printf("● TDS_max = %d ppm\n", TDS_max); } // Nhận TDS_max
BLYNK_WRITE(V9) { TEMP_max = param.asFloat(); Serial.printf("⚡ TEMP_max = %.2f°C\n", TEMP_max); } // Nhận TEMP_max

// ----- Đọc cảm biến pH -----
float readPH() {
    const int samples = 10; // // Lấy trung bình 10 mẫu
    long sum = 0;
    for (int i = 0; i < samples; i++) {
        sum += ads.readADC_SingleEnded(0); // Đọc ADC kênh 0 (pH)
        delay(10);
    }
}
```

```
float raw = sum / samples; // Giá trị trung bình
float voltage = raw * LSB; // Chuyển sang điện áp
float pH = PH_SLOPE * voltage + PH_INTERCEPT; // Tính pH
pH = constrain(pH, 0, 14); // Giới hạn pH từ 0–14

Serial.printf("ADC0=%0f | V=%0.3f V | pH=%0.2f\n", raw, voltage, pH);
return pH;
}

// ----- Đọc cảm biến TDS -----
float readTDS() {
    const int samples = 10;
    long sum = 0;
    for (int i = 0; i < samples; i++) {
        sum += ads.readADC_SingleEnded(1); // Đọc ADC kênh 1(TDS)
        delay(10);
    }

    float adc = (float)sum / samples; // Giá trị trung bình
    float voltage = adc * 0.125;      // mV
    float voltageV = voltage / 1000.0; // V

    float ec = (133.42 * pow(voltageV, 3)
                - 255.86 * pow(voltageV, 2)
                + 857.39 * voltageV);

    float tds = ec * TDS_FACTOR * CALIBRATION_FACTOR; // Tính TDS
    if (tds < 0) tds = 0; // Không cho âm

    Serial.printf("ADC1=%0f | V=%0.3f V | TDS=%0.2f ppm\n", adc, voltageV, tds);
    return tds;
}

// ----- Đọc cảm biến Nhiệt độ -----
float readTemperature() {
```

```
sensors.requestTemperatures();// Yêu cầu DS18B20 đọc
float tempC = sensors.getTempCByIndex(0); // Lấy giá trị nhiệt độ

// Loại bỏ lỗi khi cảm biến chưa sẵn sàng hoặc bị ngắt
if (tempC == DEVICE_DISCONNECTED_C || tempC == 85.0 || tempC < -10.0 ||
tempC > 100.0) {
    Serial.println("⚠ Cảm biến nhiệt độ lỗi hoặc chưa sẵn sàng!");
    return NAN; // Trả về NAN để các phần khác bỏ qua
}

Serial.printf(" 🌡 Nhiệt độ: %.2f °C\n", tempC);
return tempC;
}

// ----- Còi bíp bíp -----
void buzzerBeep() {
    if (buzzerActive && buzzerEnabled) {// Nếu cảnh báo và buzzer bật
        if (buzzerState) tone(BUZZER, 1000); // Phát tần số 1kHz
        else noTone(BUZZER); // Tắt buzzer
        buzzerState = !buzzerState; // Đảo trạng thái buzzer
    } else {
        noTone(BUZZER); // Không cảnh báo
    }
}

// ----- Hiển thị LED + Buzzer -----
void showWarning(float tds, float pH, float temp) {
    bool danger = false; // Flag cảnh báo

    // Kiểm tra TDS
    if (tds >= TDS_max) {
        danger = true;
        Serial.println("⚠ CẢNH BÁO: TDS vượt ngưỡng!");
        if (Blynk.connected())
```

```
Blynk.logEvent("tds_alert", "⚠ CẢNH BÁO: TDS vượt ngưỡng!");
}

// Kiểm tra pH
if (pH < pH_min) {
  danger = true;
  Serial.println("⚠ CẢNH BÁO: Nước có tính AXIT!");
  if (Blynk.connected())
    Blynk.logEvent("ph_alert", "⚠ CẢNH BÁO: pH thấp!");
}
else if (pH > pH_max) {
  danger = true;
  Serial.println("⚠ CẢNH BÁO: Nước có tính KIỀM!");
  if (Blynk.connected())
    Blynk.logEvent("ph_alert", "⚠ CẢNH BÁO: pH cao!");
}

// Kiểm tra nhiệt độ
if (!isnan(temp) && temp > TEMP_max) {
  danger = true;
  Serial.println("⚠ CẢNH BÁO: Nhiệt độ cao!");
  if (Blynk.connected())
    Blynk.logEvent("temp_alert", "⚠ CẢNH BÁO: Nhiệt độ vượt ngưỡng!");
}

if (!danger) Serial.println("✓ Nước bình thường.");

digitalWrite(LED_GREEN, !danger); // LED xanh nếu bình thường
digitalWrite(LED_RED, danger); // LED đỏ nếu vượt ngưỡng
buzzerActive = danger; // Bật tắt còi
}

// ----- hiển thị OLED -----
void showOLED(float pH, float tds, float temp) {
  display.clearDisplay();
```

```
display.setTextSize(1);
display.setTextColor(SSD1306_WHITE);

// ---- Hiển thị trạng thái Wifi ----
display.setCursor(0, 0);
if (WiFi.status() == WL_CONNECTED) {
    display.print("WiFi: OK");
} else {
    display.print("WiFi: Mat!");
}

// ---- hiển thị trạng thái pH trên OLED----
display.setCursor(0, 12);
display.printf("pH: %.2f ", pH); // hiển thị pH
if (pH < pH_min) display.print("pH Thap");
else if (pH > pH_max) display.print("pH Cao");

// ---- hiển thị trạng thái TDS khi vượt ngưỡng----
display.setCursor(0, 24);
display.printf("TDS: %.0fppm ", tds); // hiển thị tds
if (tds >= TDS_max) display.print("!!!!");

// ---- Hiển thị trạng thái nhiệt độ khi vượt ngưỡng ----
display.setCursor(0, 36);
if (!isnan(temp)) {
    display.printf("Temp: %.1fC ", temp); // hiển thị nhiệt độ
    if (temp > TEMP_max) display.print("!!!!");
} else {
    display.print("Temp: ---"); // Ẩn khi lỗi hoặc 85°C
}

// ---- Trạng thái tổng thể của hệ thống ----
display.setCursor(0, 50);
if ((pH < pH_min) || (pH > pH_max) || (tds >= TDS_max) || (!isnan(temp) && temp
> TEMP_max)) {
```

```
    display.println(" Vuot nguong!");
  } else {
    display.println(" Nuoc Binh thuong");
  }

  display.display();
}

// ----- Gửi dữ liệu lên Blynk -----
void sendData() {
  float pH = readPH();
  float tds = readTDS();
  float temp = readTemperature();

  showWarning(tds, pH, temp);
  showOLED(pH, tds, temp);

  if (Blynk.connected()) {
    Blynk.virtualWrite(V0, pH);
    Blynk.virtualWrite(V2, tds);
    Blynk.virtualWrite(V1, temp);
  }
}

// ----- Điều khiển bật/tắt còi từ app -----
BLYNK_WRITE(V4) {
  int state = param.asInt();
  buzzerEnabled = state;
  if (!buzzerEnabled) {
    noTone(BUZZER);
    Serial.println("🔊 Buzzer bị tắt thủ công!");
  } else {
    Serial.println("🔊 Buzzer được bật lại!");
  }
}
```

```
// ----- SETUP -----  
void setup() {  
  Serial.begin(115200);  
  
  if (!ads.begin()) {  
    Serial.println("Không tìm thấy ADS1115!");  
    while (1);  
  }  
  
  ads.setGain(GAIN_ONE);  
  pinMode(LED_GREEN, OUTPUT);  
  pinMode(LED_RED, OUTPUT);  
  pinMode(BUZZER, OUTPUT);  
  
  sensors.begin();  
  
  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {  
    Serial.println("⚠ Không tìm thấy OLED!");  
    while (1);  
  }  
  display.clearDisplay();  
  display.display();  
  
  WiFi.begin(ssid, pass); // kết nối wifi  
  Serial.print("Đang kết nối WiFi");  
  unsigned long startAttemptTime = millis();  
  while (WiFi.status() != WL_CONNECTED && millis() - startAttemptTime <  
10000) {  
    Serial.print(".");  
    delay(500);  
  }  
  Serial.println(WiFi.status() == WL_CONNECTED ? "\n✅ WiFi connected!" :  
"\n⚠ WiFi chưa kết nối!");  
  
  Blynk.config(BLYNK_AUTH_TOKEN); // cấu hình Blynk
```

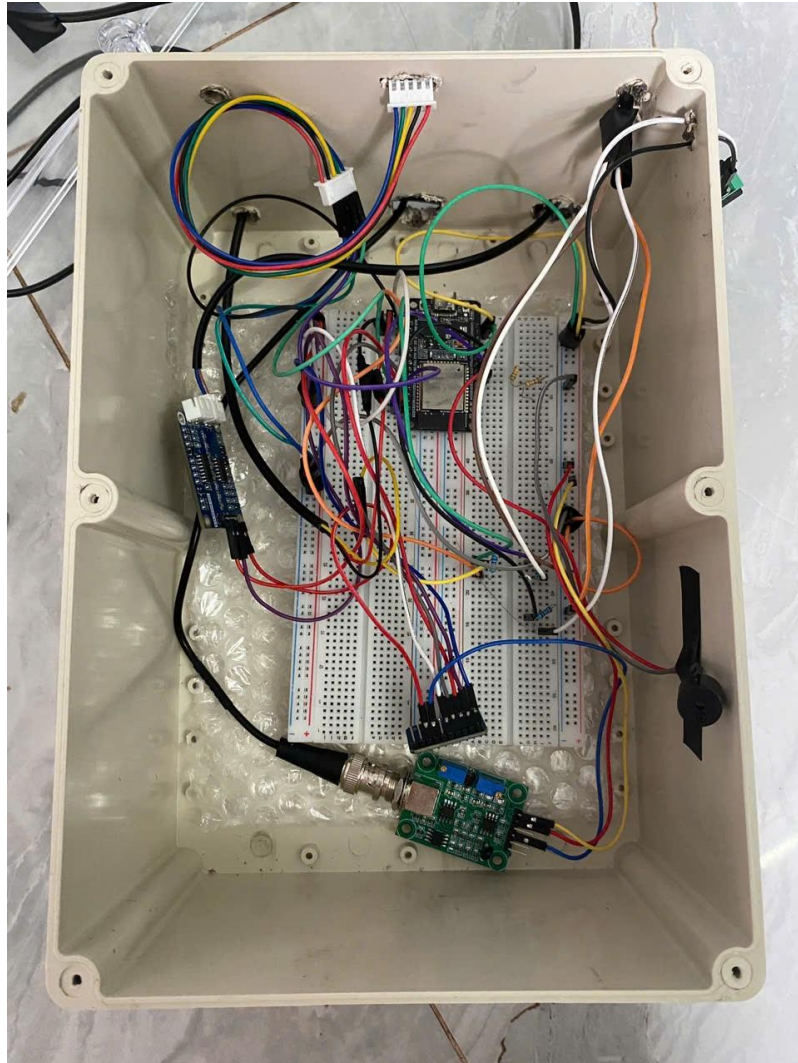
```
Blynk.connect(1000);

timer.setInterval(2000L, sendData); // gửi dữ liệu mỗi 2s
buzzerTimer.setInterval(300L, buzzerBeep); // Nhấp nháy còi mỗi 0.5s
}
// ----- LOOP -----
void loop() {
    if (WiFi.status() == WL_CONNECTED && !Blynk.connected())
Blynk.connect(1000); // giữ kết nối Blynk
    Blynk.run();
    timer.run();
    buzzerTimer.run();
}
```

Chương 3: KẾT QUẢ THỰC NGHIỆM HỆ THỐNG

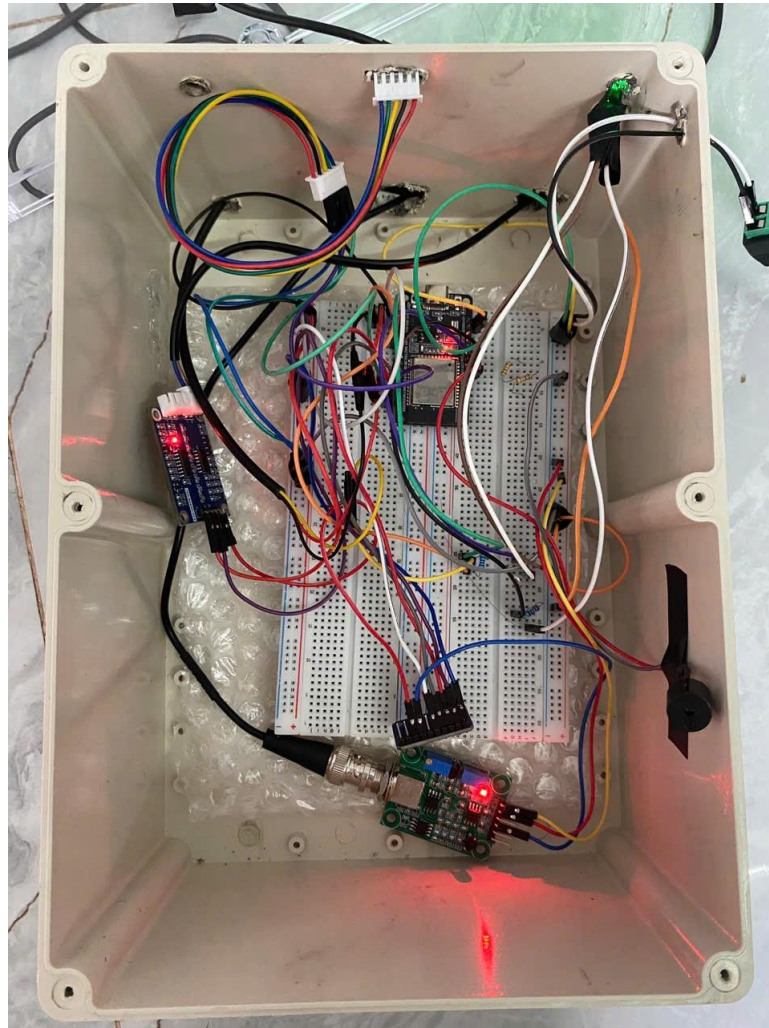
3.1. Mô hình thực tế

Mạch hệ thống IoT giám sát chất lượng nước sinh hoạt khi chưa hoạt động:



Hình 3.1 Mô hình mạch IoT khi chưa hoạt động

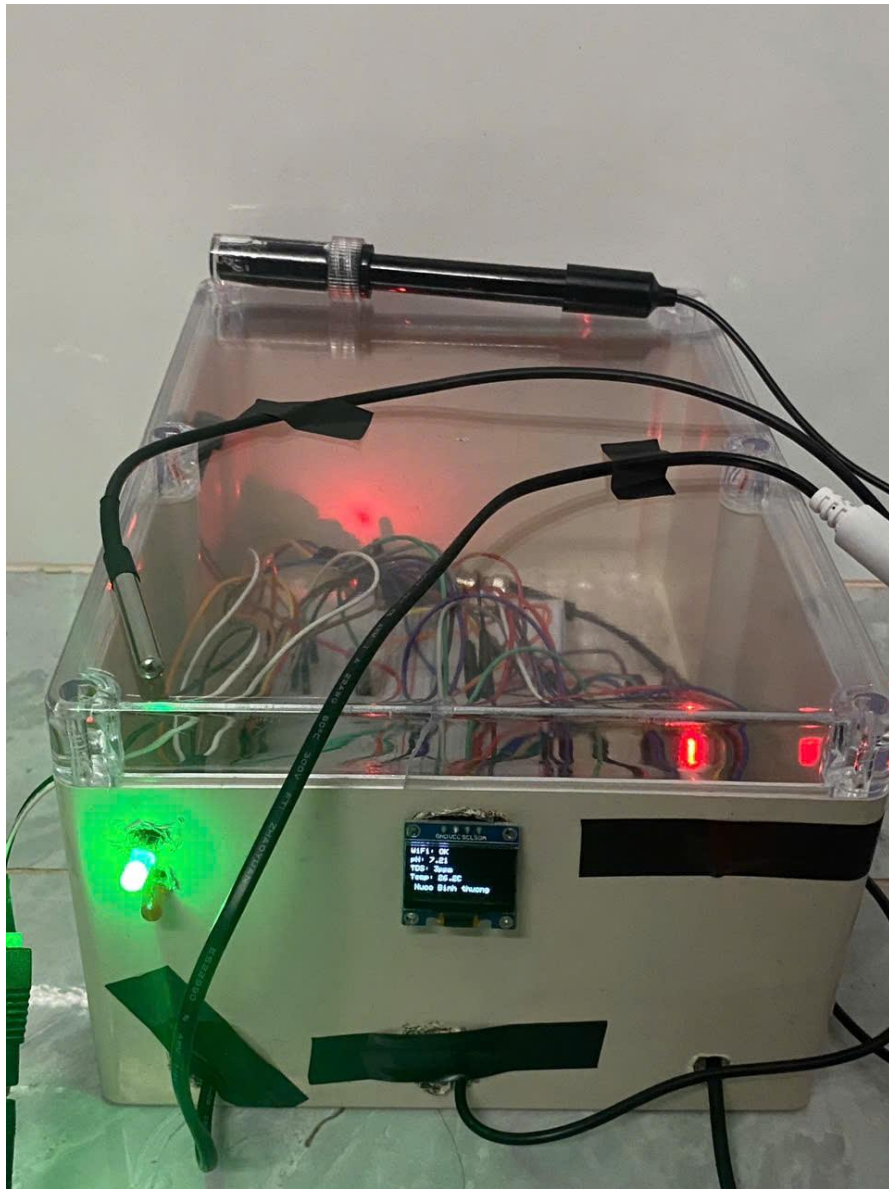
Mạch hệ thống IoT giám sát chất lượng nước sinh hoạt khi hoạt động:



Hình 3.2 Mô hình mạch hệ thống IoT khi hoạt động

Hệ thống IoT giám sát chất lượng nước sinh hoạt khi hoạt động:

Hệ thống IoT giám sát chất lượng nước sinh hoạt gồm ESP32, các cảm biến pH – TDS – nhiệt độ, màn hình OLED cùng LED và còi cảnh báo. Khi hoạt động, hệ thống đo các thông số nước và hiển thị lên OLED và ứng dụng Blynk. LED xanh sáng khi nước ở trạng thái bình thường, còn LED đỏ cùng còi sẽ kích hoạt khi các giá trị vượt ngưỡng.



Hình 3.3 Hệ thống IoT giám sát chất lượng nước sinh hoạt

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 OLED ở trạng thái nước bình thường:

Ở trạng thái nước bình thường, màn hình OLED hiển thị các thông số gồm: trạng thái kết nối WiFi, giá trị pH, TDS, nhiệt độ và thông báo “Nước bình thường”. Các giá trị được cập nhật liên tục, giúp người dùng dễ dàng theo dõi chất lượng nước tại chỗ.



Hình 3.4 Kết quả đo hiển thị trên OLED khi bình thường

Kết quả đo và hiển thị trên OLED ở trạng thái nước vượt ngưỡng:

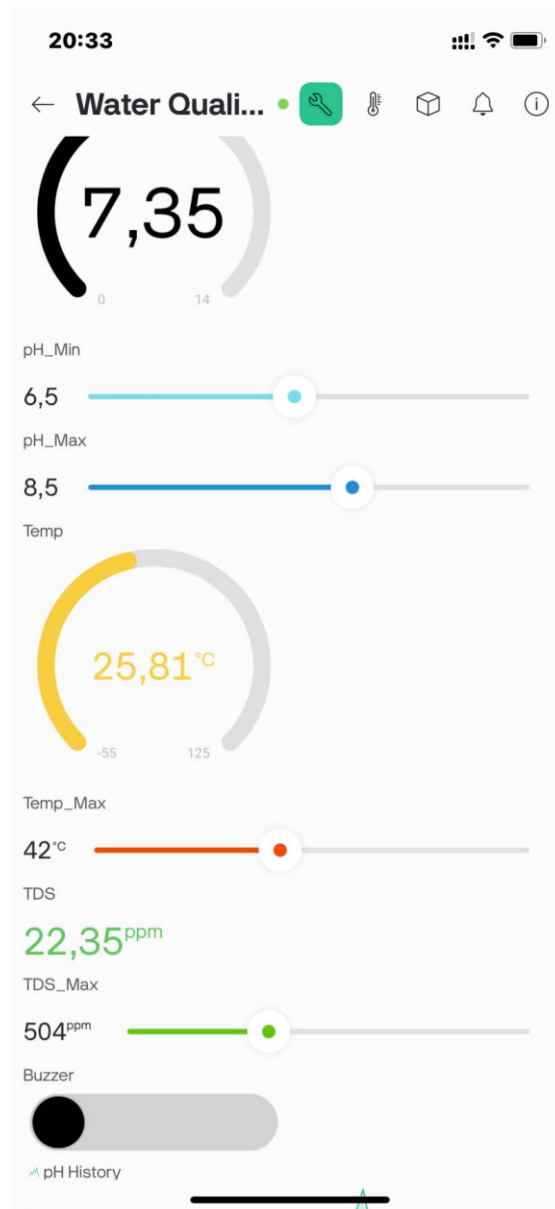
Ở trạng thái nước vượt ngưỡng, màn hình OLED bổ sung các dấu “!!!” bên cạnh những thông số vượt giới hạn an toàn, như TDS hoặc nhiệt độ. Riêng đối với pH, màn hình hiển thị thông báo cụ thể: “pH thấp” hoặc “pH cao”, đồng thời kèm theo cảnh báo “Vượt ngưỡng”. Nhờ đó, người dùng nhận biết được loại thông số nào đang bất thường và mức độ nguy hiểm của nguồn nước.



Hình 3.5 Kết quả đo hiển thị trên OLED khi vượt ngưỡng

Kết quả đo và hiển thị trên Blynk:

Kết quả hiển thị trên ứng dụng Blynk cho thấy các thông số được trình bày rõ ràng qua các widget: pH và nhiệt độ được hiển thị bằng Gauge giúp quan sát trực quan theo thời gian thực, trong khi TDS được hiển thị bằng Value Display dưới dạng giá trị số. Cách bố trí này giúp người dùng dễ dàng theo dõi chất lượng nước ngay trên điện thoại.

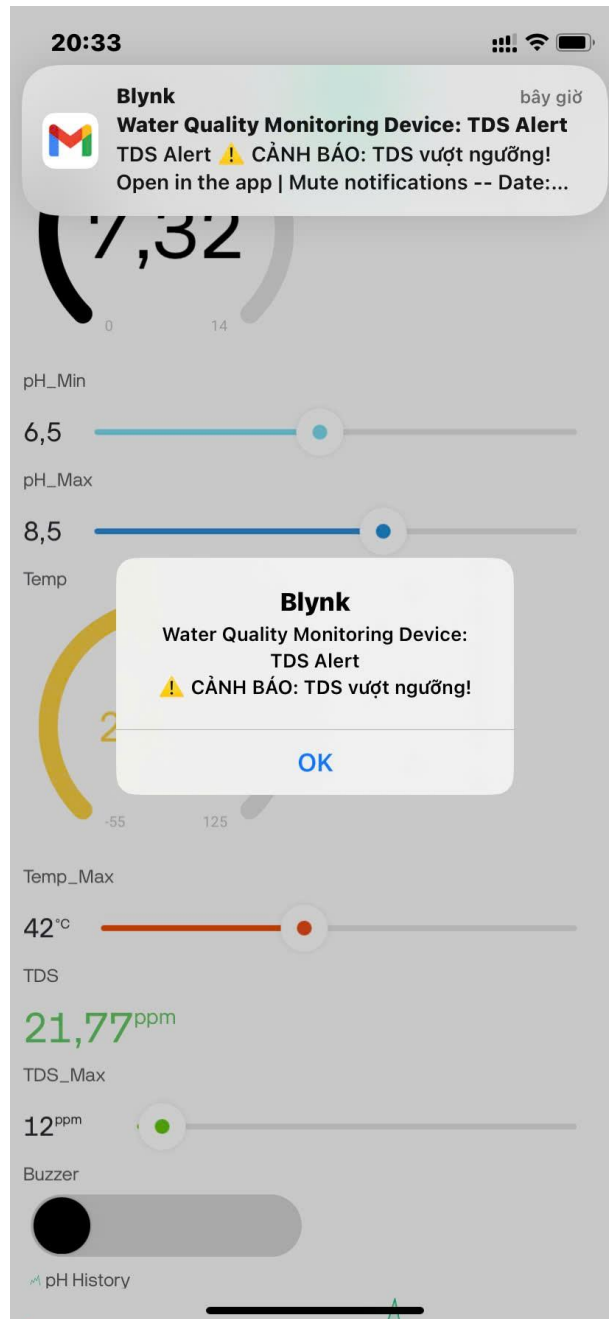


Hình 3.6 Kết quả đo hiển thị trên Blynk

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

Kết quả cảnh báo khi vượt ngưỡng của TDS:

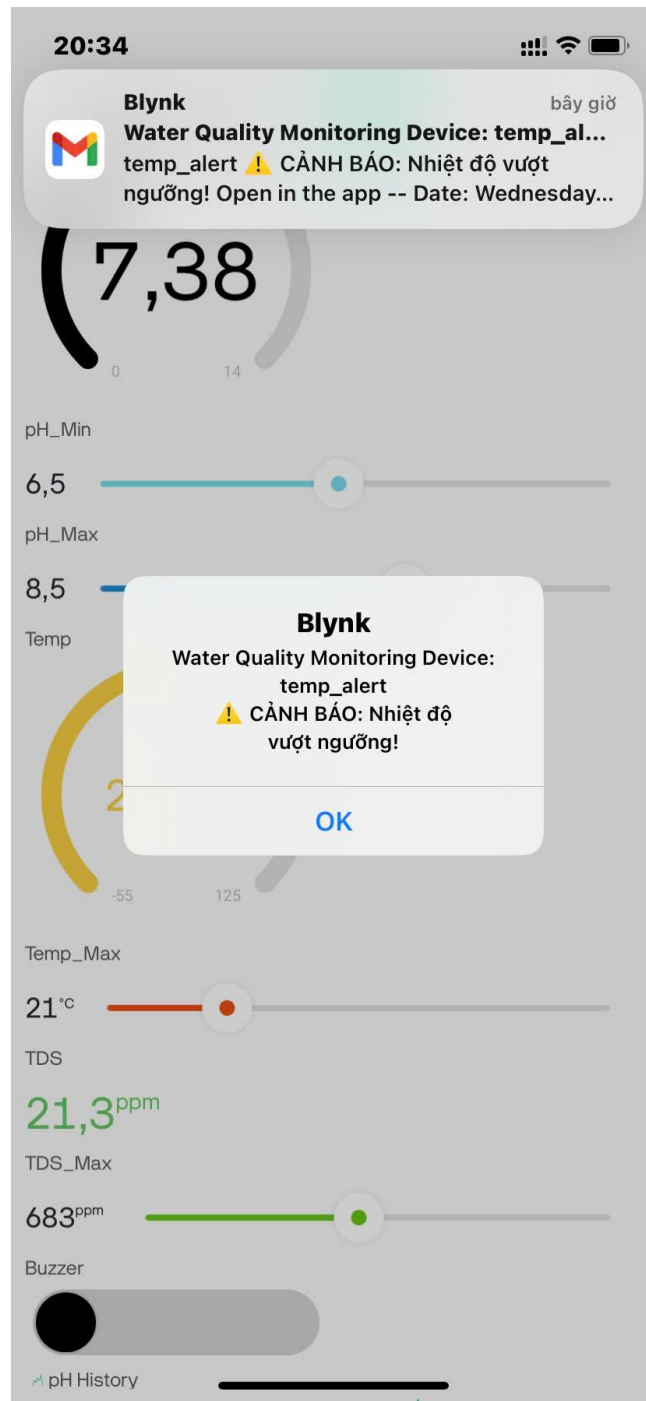
Khi giá trị TDS vượt quá ngưỡng cho phép, hệ thống gửi cảnh báo ngay lập tức lên ứng dụng Blynk với thông báo “CẢNH BÁO: TDS vượt ngưỡng!”. Đồng thời, Blynk cũng gửi email cảnh báo đến người dùng, giúp họ nhanh chóng nhận biết và xử lý kịp thời.



Hình 3.7 Kết quả cảnh báo khi vượt ngưỡng của TDS

Kết quả cảnh báo khi vượt ngưỡng của nhiệt độ:

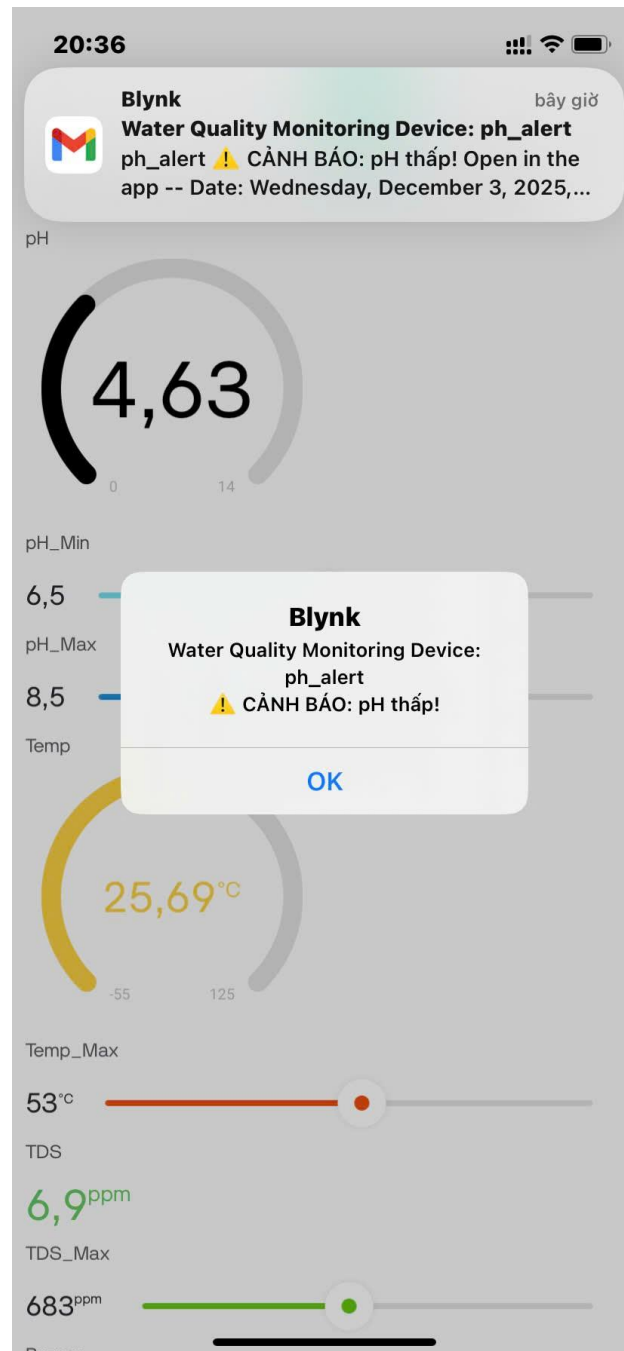
Khi nhiệt độ nước vượt ngưỡng cài đặt, hệ thống gửi cảnh báo ngay lập tức lên ứng dụng Blynk với thông báo “CẢNH BÁO: Nhiệt độ vượt ngưỡng!”. Bên cạnh đó, một email cảnh báo cũng được gửi đến người dùng để đảm bảo thông tin được tiếp nhận nhanh chóng.



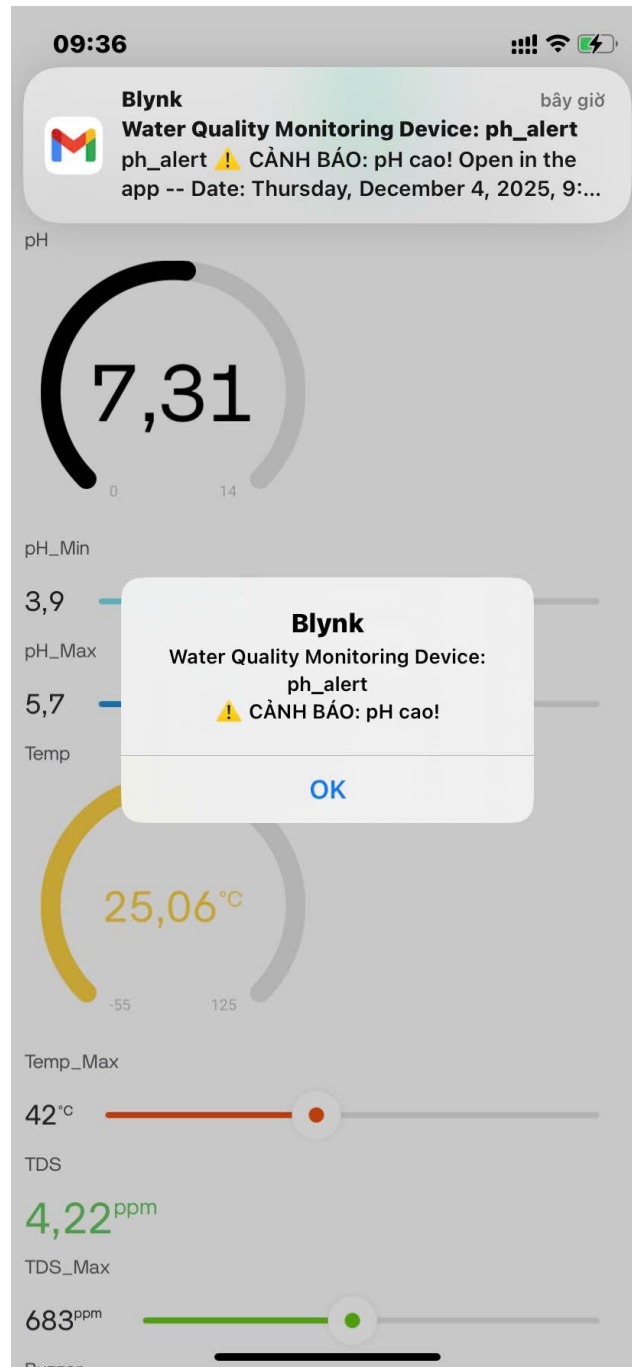
Hình 3.8 Kết quả cảnh báo khi vượt ngưỡng của nhiệt độ

Kết quả cảnh báo khi vượt ngưỡng của pH:

Khi giá trị pH vượt quá ngưỡng cho phép, hệ thống gửi cảnh báo lên ứng dụng Blynk với thông báo tương ứng: “CẢNH BÁO: pH thấp!” hoặc “CẢNH BÁO: pH cao!”. Đồng thời, Blynk gửi email cảnh báo đến người dùng, giúp họ kịp thời phát hiện sự bất thường của nguồn nước.



Hình 3.9 Kết quả cảnh báo khi vượt ngưỡng của pH khi vượt mức tối thiểu

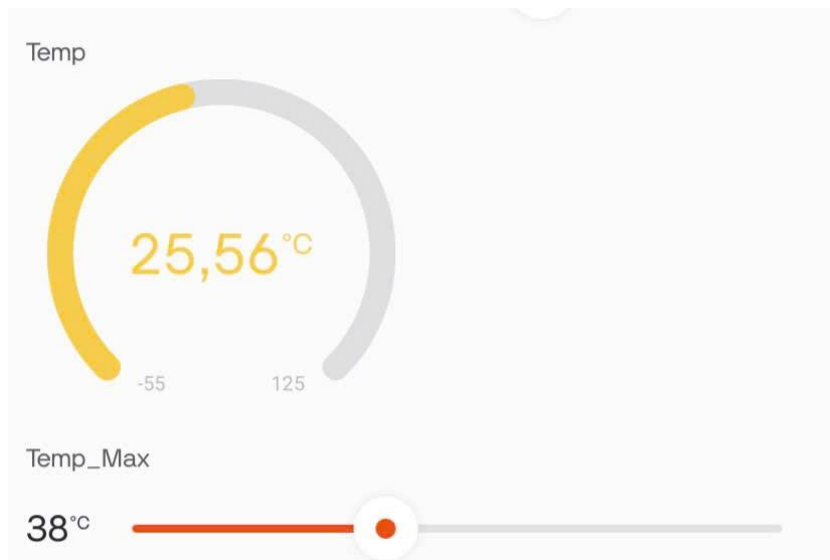


Hình 3.10 Kết quả cảnh báo khi vượt ngưỡng của pH tối đa

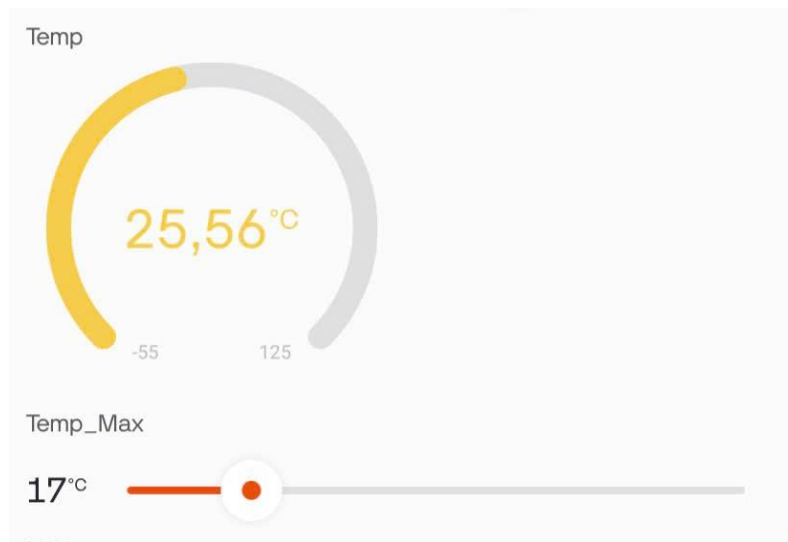
3.2.3. Kết quả chức năng điều khiển ngưỡng cảnh báo

Kết quả chức năng điều khiển ngưỡng của nhiệt độ:

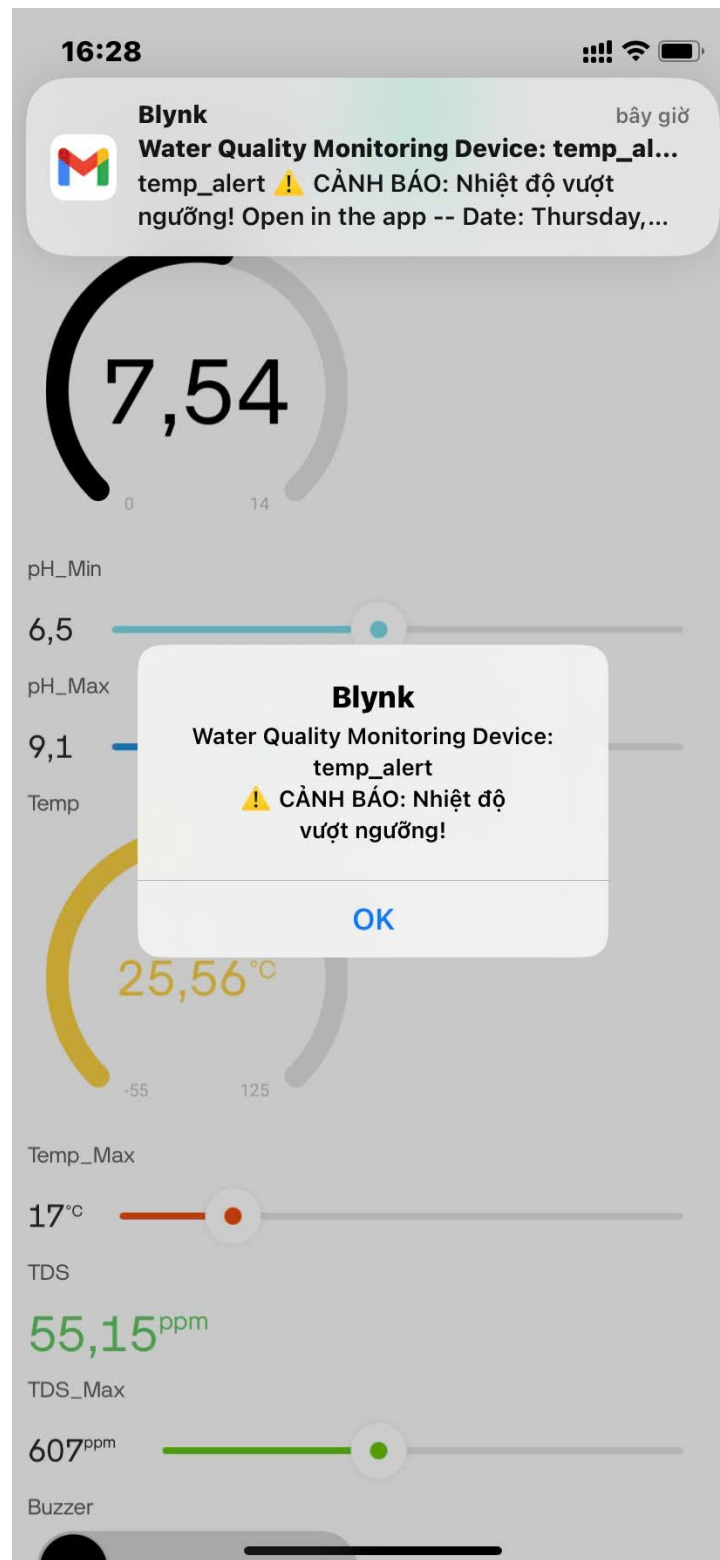
Đối với thông số nhiệt độ, người dùng có thể điều chỉnh ngưỡng cảnh báo trực tiếp trên ứng dụng Blynk. Ban đầu, ngưỡng được đặt ở mức 35°C. Khi người dùng thay đổi ngưỡng, hệ thống cập nhật ngay và dùng giá trị mới để so sánh với nhiệt độ đo được. Nếu nhiệt độ vượt quá ngưỡng đã đặt, hệ thống gửi cảnh báo lên ứng dụng Blynk và đồng thời gửi email, giúp người dùng kiểm soát chất lượng nước sinh hoạt và kịp thời.



Hình 3.11 Ngưỡng nhiệt độ trước chỉnh ngưỡng



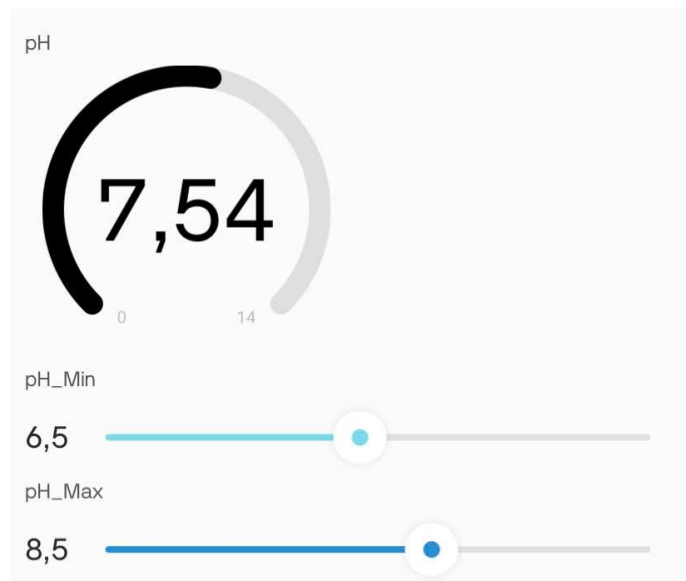
Hình 3.12 Ngưỡng nhiệt độ sau chỉnh ngưỡng



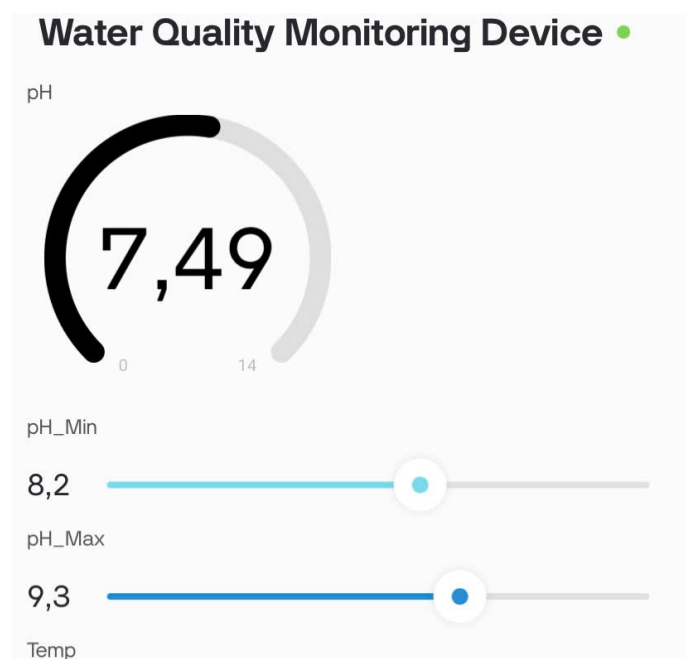
Hình 3.13 Kết quả khi điều chỉnh ngưỡng nhiệt độ

Kết quả chức năng điều khiển ngưỡng của pH:

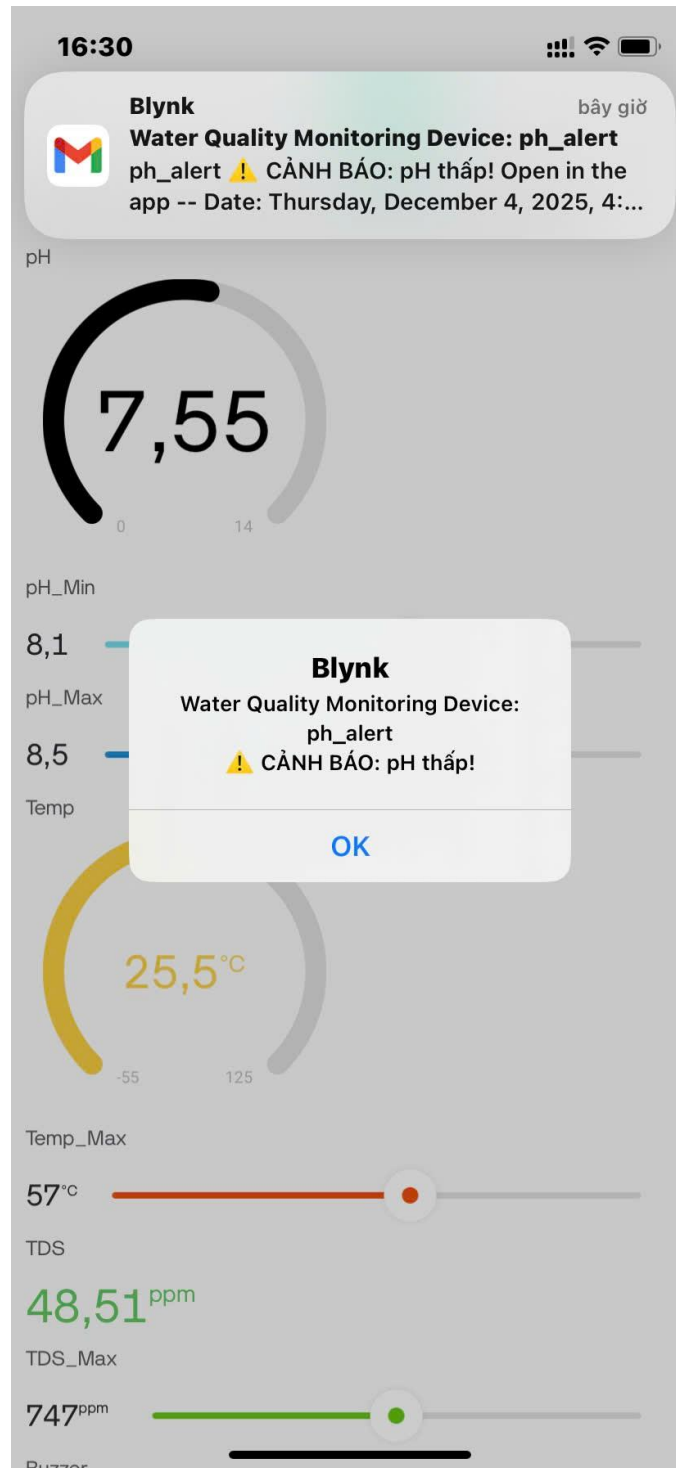
Người dùng có thể điều chỉnh ngưỡng pH tối thiểu và tối đa trực tiếp trên ứng dụng Blynk. Ban đầu, khoảng pH được đặt từ 6,5 đến 8,5. Khi người dùng thay đổi các giá trị ngưỡng này, hệ thống cập nhật ngay lập tức và so sánh với pH đo được. Nếu pH thấp hơn ngưỡng tối thiểu, hệ thống gửi cảnh báo “pH thấp!”; nếu pH vượt ngưỡng tối đa, hệ thống gửi cảnh báo “pH cao!” lên ứng dụng Blynk và đồng thời gửi email, giúp người dùng phát hiện kịp thời chất lượng nước bất thường.



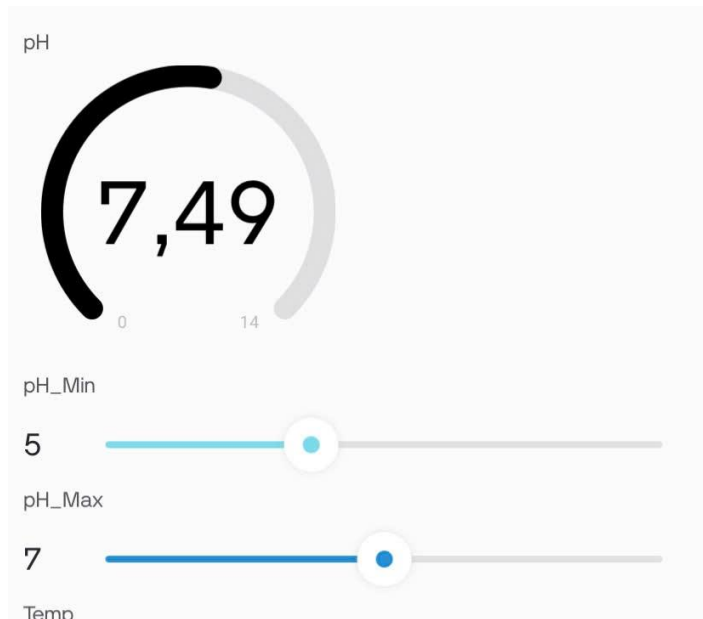
Hình 3.14 Ngưỡng pH trước khi hiệu chỉnh



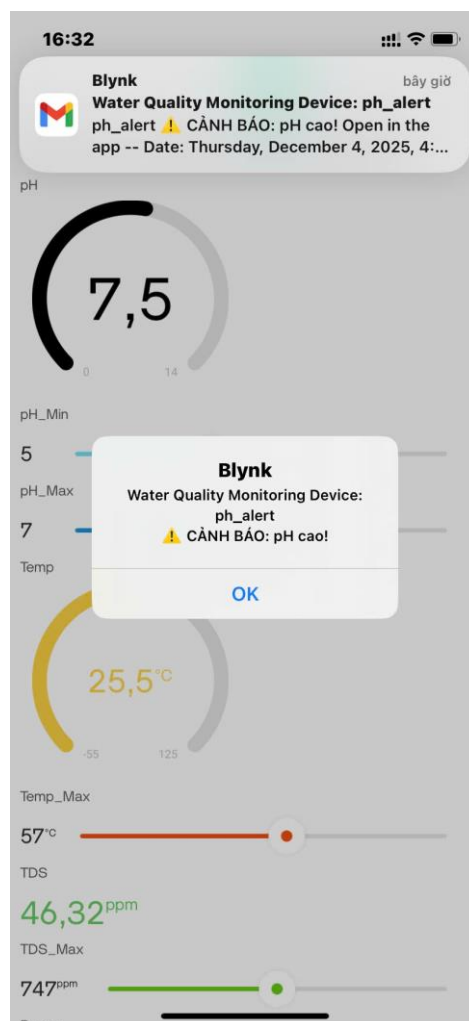
Hình 3.15 Ngưỡng pH tối thiểu khi chỉnh ngưỡng



Hình 3.16 Kết quả sau chỉnh ngưỡng pH tối thiểu



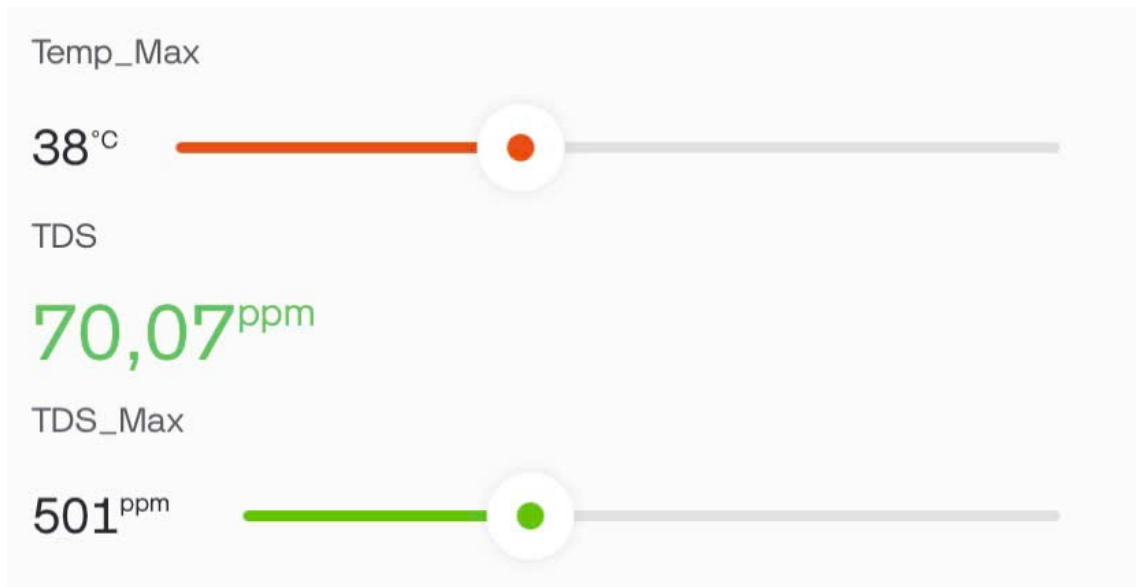
Hình 3.17 Ngưỡng pH tối đa khi chỉnh ngưỡng



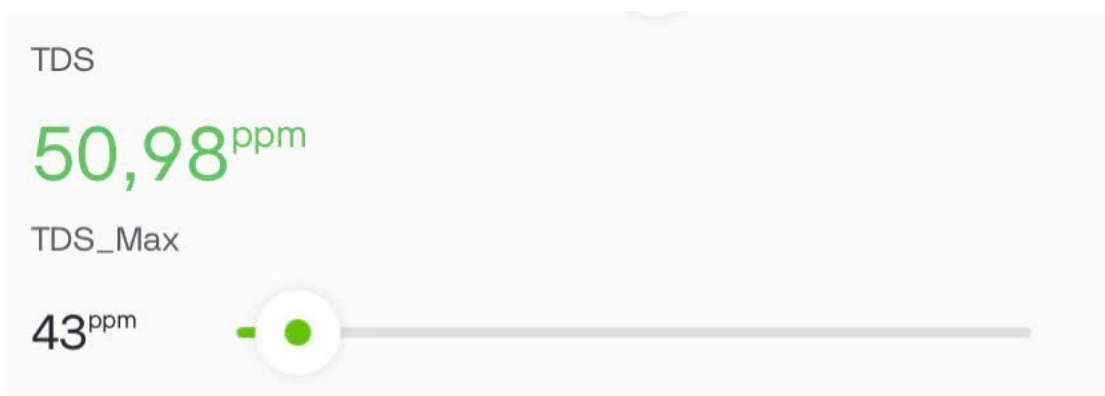
Hình 3.18 Kết quả sau chỉnh ngưỡng pH tối đa

Kết quả chức năng điều khiển ngưỡng của TDS:

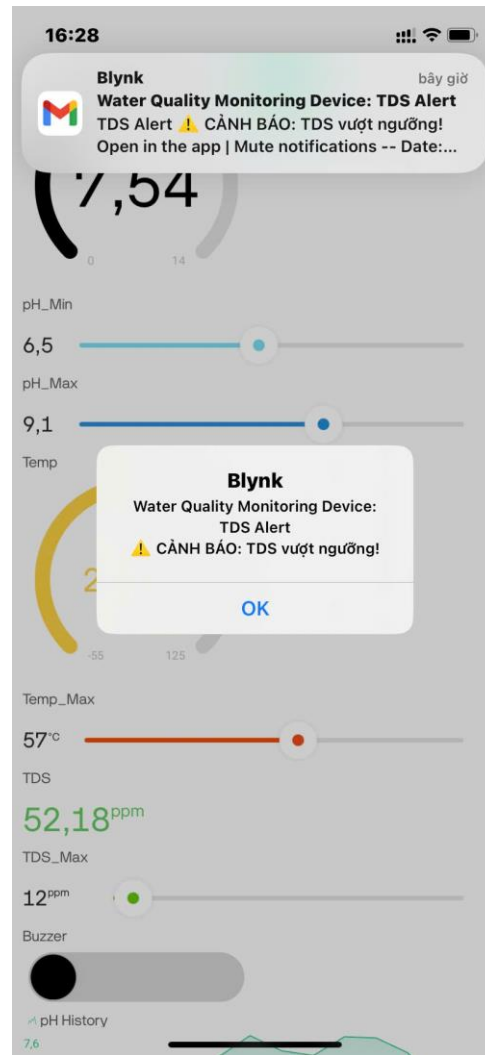
Người dùng có thể đặt và điều chỉnh ngưỡng TDS tối đa trực tiếp trên Blynk. Khi TDS đo được vượt quá ngưỡng mới, hệ thống lập tức gửi cảnh báo “TDS vượt ngưỡng!” lên ứng dụng và gửi email cho người dùng. Nhờ đó, việc giám sát chất lượng nước trở nên linh hoạt và kịp thời hơn.



Hình 3.19 Ngưỡng TDS trước khi chỉnh



Hình 3.20 Ngưỡng TDS khi chỉnh ngưỡng



Hình 3.21 Kết quả sau chỉnh ngưỡng của TDS

3.2.4. Đánh giá hoạt động thực tế

Qua quá trình vận hành thử, hệ thống hoạt động ổn định và phản hồi tốt với các thay đổi của chất lượng nước. Các cảm biến đo pH, TDS và nhiệt độ cho kết quả hiển thị chính xác trên màn hình OLED và ứng dụng Blynk. Chức năng cảnh báo hoạt động đúng yêu cầu: LED xanh báo trạng thái bình thường, LED đỏ và còi kích hoạt ngay khi thông số vượt ngưỡng. Ứng dụng Blynk gửi thông báo và email kịp thời, đồng thời việc điều chỉnh ngưỡng cảnh báo được thực hiện dễ dàng và cập nhật ngay lập tức. Nhìn chung, hệ thống đáp ứng tốt mục tiêu giám sát chất lượng nước sinh hoạt theo thời gian thực.

KẾT LUẬN

Sau thời gian tìm hiểu và thực hiện đề tài “Hệ thống IoT giám sát chất lượng nước sinh hoạt”, em đã xây dựng được một mô hình hoàn chỉnh có khả năng đo pH, TDS và nhiệt độ của nước, đồng thời hiển thị dữ liệu lên màn hình OLED và ứng dụng Blynk theo thời gian thực. Hệ thống cũng tích hợp cảnh báo bằng đèn LED, còi buzzer và thông báo qua Blynk khi các thông số vượt ngưỡng, giúp người dùng dễ dàng nhận biết tình trạng chất lượng nước.

Kết quả thực nghiệm cho thấy hệ thống hoạt động ổn định, các cảm biến đọc giá trị tương đối chính xác, giao diện Blynk thân thiện và thuận tiện cho việc theo dõi từ xa. Toàn bộ mạch được bố trí trong hộp nhựa và sử dụng nguồn adapter 5V, đảm bảo tính gọn gàng và an toàn khi vận hành.

Tuy nhiên, vì điều kiện thời gian và kiến thức còn hạn chế, hệ thống vẫn còn một số điểm chưa hoàn thiện như chưa tối ưu độ chính xác của cảm biến, chưa có chức năng lưu lại dữ liệu, và chỉ mới tập trung vào các thông số cơ bản nhất.

Hướng phát triển trong khả năng cá nhân có thể thực hiện sau này là:

- Cải thiện việc hiệu chỉnh cảm biến để giá trị đo ổn định hơn.
- Thêm chức năng lưu dữ liệu vào thẻ nhớ hoặc gửi lên cloud để xem lại lịch sử.
- Bổ sung giao diện hiển thị đẹp và dễ dùng hơn trên Blynk.

Dù còn những hạn chế nhất định, đề tài đã đạt được các mục tiêu ban đầu đề ra và giúp em củng cố thêm kiến thức về IoT, lập trình vi điều khiển, làm việc với cảm biến và xây dựng một hệ thống hoàn chỉnh.

TÀI LIỆU THAM KHẢO

<https://www.pace.edu.vn/tin-kho-tri-thuc/iot-la-gi>

<https://vina-aspire.com/internet-van-vat-iot-la-gi/>

<https://base.vn/blog/iot-la-gi/>

<https://fptshop.com.vn/tin-tuc/danh-gia/blynk-la-gi-184300>

<https://docs.blynk.io/en>

<https://stringee.com/vi/blog/post/lap-trinh-arduino-vi-sao-pho-bien>

https://thinkrobotics.com/blogs/learn/what-is-arduino-ide-a-complete-guide-for-beginners?srltid=AfmBOOpSg5wR7FB-dW4fX4UP6_JJARPY-Mc8URd0TuTntTDWmfz0ydrC&utm_source=chatgpt.com

https://3dpellet.com/tutorials/arduino/first-steps/arduino-ide/?utm_source=chatgpt.com

<https://fptshop.com.vn/tin-tuc/danh-gia/phan-mem-arduino-ide-171894>

<https://emeritus.org/blog/coding-arduino-programming-language/#advantages-of-the-arduino-programming-language>

<https://dientuviet.com/gioi-thieu-esp32/>

<http://arduino.vn/tutorial/1570-gioi-thieu-module-esp32-va-huong-dan-cai-trinh-bien-dich-tren-arduino-ide>

<https://mlab.com.vn/cam-bien-tds-do-tong-chat-ran-hoa-tan-total-dissolved-solids-sensor-dfrobot-mlab-com-vn>

<https://inno-corp.vn/cam-bien-do-do-ph-la-gi-ung-dung-va-nguyen-ly-hoat-dong/>

<https://nshopvn.com/product/cam-bien-do-do-ph/>

https://www.az-delivery.de/en/products/0-96zollldisplay?srltid=AfmBOoo2wI17Vsv_hpzfHQ4s9rXYjgsOhdI4cvSzwLO3ErUKJQRuBXd

<https://hshop.vn/coi-buzzer-5vdc>

<https://www.thegioiic.com/ads1115-mach-adc-16bit-i2c-2-5-5vdc>

<https://community.blynk.cc/t/water-quality-monitoring-device-prototype/53868>

<https://cimpleo.com/blog/arduino-ph-meter-using-ph-4502c/#wiring-diagram>

<https://www.youtube.com/watch?v=GswTNPPCp70&t=826s>

https://www.youtube.com/watch?v=Hjc_IFDs9jk&t=149s

https://www.g2.com/products/arduino-ide/reviews?qs=pros-and-cons&utm_source=chatgpt.com#pros