

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN ĐIỆN TỬ - VIỄN THÔNG



BÁO CÁO
KỸ THUẬT VI XỬ LÝ

Đề tài:

Giảng viên hướng dẫn: Hàn Huy Dũng

Sinh viên thực hiện:

Trần Duy Anh	20203872
Phạm Tiến Hùng	20203881
Nguyễn Bá Trung Hiếu	20200224
Phạm Quốc Huy	20200283
Đoàn Quang Lưu	20203884

Hà Nội, tháng 3 năm 2023

CONTENTS

Danh mục hình vẽ

Error! Bookmark not defined.

Danh mục bảng biểu

Error! Bookmark not defined.

I. IDEA:	4
1. Bối cảnh:	4
2. Các sản phẩm đã có:	5
II. SPECS:	7
1. Chỉ tiêu chức năng và ràng buộc dự án:	7
2. Chỉ tiêu phi chức năng:	7
III. Giải pháp: Kiến trúc/Thuật toán	8
1. Sơ đồ khối và giao thức:	8
2. Vai trò của các khối:	9
IV. Triển khai	9
1.1. Linh kiện:	9
2. Layout bảng mạch:	21
3. Lập trình trên các cảm biến đo	22
Bảng dữ liệu của nhiệt	26
V. Kiểm tra/Thử nghiệm	31
1. Kiểm tra chức năng của hệ thống:	31
VI. Kết luận	32
VII. Bảng công việc	32
VIII. Tài liệu tham khảo	32

I. IDEA:

1. Bối cảnh:

1.1. What: Vấn đề đang xảy ra ở đây?:

- Mối quan ngại ngày càng tăng về ô nhiễm không khí và tác động của nó đến sức khỏe con người và môi trường.
- Cần thiết bị để đo chất lượng không khí và cung cấp dữ liệu thời gian thực về ô nhiễm không khí, độ ẩm.

1.2. Why: Tại sao?

- Giải quyết vấn đề ô nhiễm không khí và tác động của nó đến sức khỏe con người và môi trường.
- Giúp con người đưa ra quyết định thông minh để bảo vệ bản thân và môi trường xung quanh.

1.3. Who: Ai gặp vấn đề?

- Công nhân và nhân viên làm việc trong các ngành công nghiệp ô nhiễm.
- Người dân sống trong khu vực ô nhiễm nặng.
- Các nhà khoa học và nhà nghiên cứu về môi trường

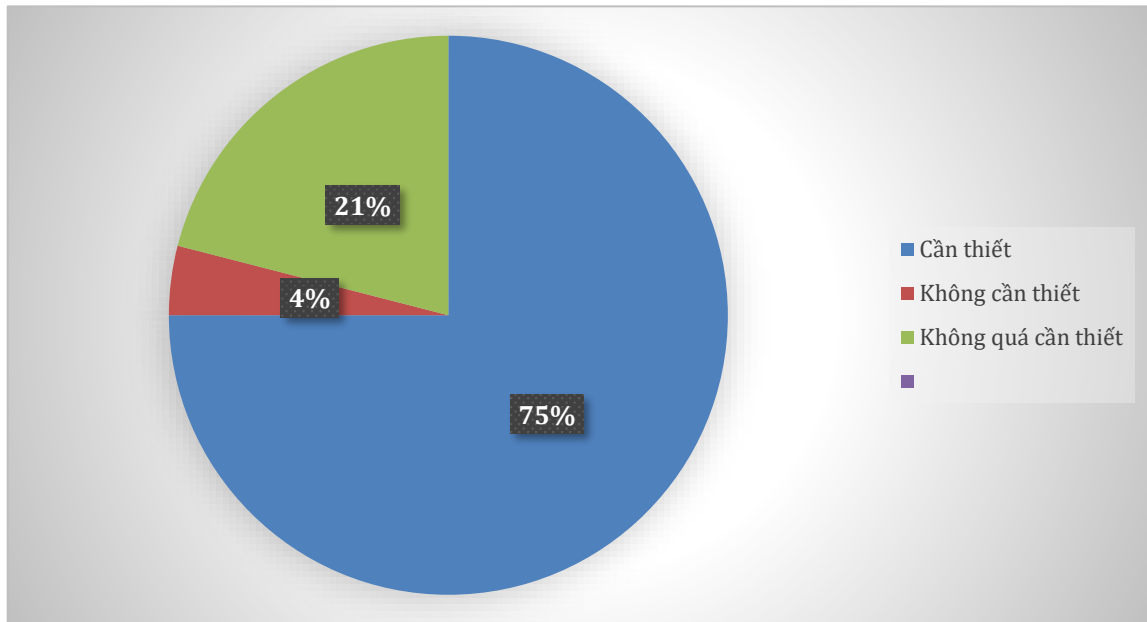
1.4. When: Khi nào gặp vấn đề?

- Khi sống hoặc làm việc trong các khu vực có chất lượng không khí kém.
- Khi bạn có bất kỳ triệu chứng nào của vấn đề sức khỏe liên quan đến ô nhiễm không khí.

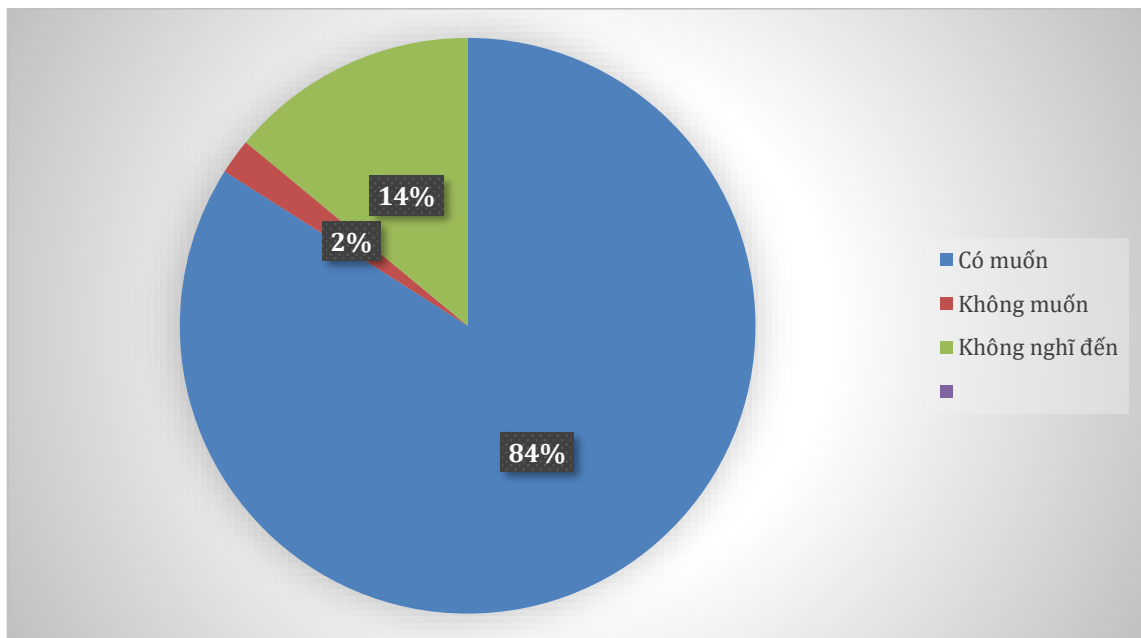
1.5. Where: Gặp ở đâu?

- Ngành sản xuất, công trình xây dựng, khai thác khoáng sản.
- Các khu công nghiệp, đô thị.
- Các khu vực đô thị lớn với mật độ dân cư cao, tuyến đường phức tạp và nhiều phương tiện giao thông.

1.6. How:



Hình 1: Khảo sát mức độ cần thiết của việc thay đổi tình trạng hiện tại



Hình 2: Khảo sát mong muốn sử dụng một thiết bị thay đổi tình trạng hiện tại

- Từ dữ liệu khảo sát, người dùng có xu hướng cảm thấy mức độ cần thiết của thay đổi tình trạng hiện tại là thực sự cần thiết quá cần thiết.

2. Các sản phẩm đã có:

Các đặc điểm	AirSense Smart Air Monitor	AirSense Laser Egg
Hình ảnh	 <p>Hình 3: AirSense Smart Air Monitor</p>	 <p>Hình 4: AirSense Laser Egg</p>
Kích thước	130 x 130 x 50 (mm)	58 x 58 x 70 mm
Giá thành	159.99\$	99.99\$
Tính năng	<p>Đo chất lượng không khí thông minh với cảm biến đa chức năng, cho phép đo nồng độ các chất độc hại trong không khí, bao gồm PM2.5, CO2, VOC và độ ẩm, nhiệt độ.</p> <p>Cung cấp thông tin chi tiết về sức khỏe và độc tính của các chất ô nhiễm, thông qua ứng dụng di động và kết nối Wi-Fi.</p>	<p>Đo nồng độ các chất độc hại trong không khí, bao gồm PM2.5, CO2 và VOC.</p> <p>Đo nhiệt độ và độ ẩm trong không khí.</p> <p>Kết nối với thiết bị di động thông qua Bluetooth để cập nhật dữ liệu đo đạc và theo dõi chất lượng không khí.</p>
Công suất	Nguồn cung cấp từ pin hoặc adapter	Pin sạc lithium-ion, có thể hoạt động liên tục trong khoảng 8 giờ và có thể sạc lại qua cổng USB.
Nhận xét ưu điểm	<p>Đo chính xác và đầy đủ về chất lượng không khí, độc tính của các chất ô nhiễm, hỗ trợ kết nối Wi-Fi và ứng dụng di động.</p> <p>Dễ sử dụng và thiết kế đẹp.</p>	<p>Thiết bị nhỏ gọn và dễ sử dụng, có thể đặt ở bất kỳ đâu trong phòng.</p> <p>Có tính năng đo đạc nồng độ các chất độc hại trong không khí, giúp người dùng kiểm soát và cải thiện chất lượng không khí trong nhà.</p>
Nhận xét nhược điểm	Giá thành khá cao.	Chỉ đo được một số chất độc hại chính trong không khí, còn lại có thể bỏ qua.

		Pin có thời gian sử dụng liên tục không lâu, khoảng 8 giờ, người dùng cần phải sạc lại thường xuyên để giữ cho thiết bị hoạt động.
--	--	--

Bảng 1: So sánh AirSense Smart Air Monitor và AirSense Laser Egg

II. SPECS:

1. Chỉ tiêu chức năng và ràng buộc dự án:

*** Input:**

- Điện áp vào sensor: 3.3V
- Tín hiệu vào Raspberry Pi 4: 5V-3A

*** Output:**

- Độ ẩm: 0-100%
- Nhiệt độ: 0 C – 50 C
- Bụi: 0 – 0,8mg/m³

2. Chỉ tiêu phi chức năng:

- Ngoại quan cơ khí:
 - + Kích thước : khoảng 15x25x6 cm
 - + Màu : đen
 - + Vật liệu : Nhựa

- Tiêu chuẩn cần tuân theo: ISO 9001:2015, Tiêu chuẩn quốc gia TCVN 9075:2011

- Môi trường hoạt động:

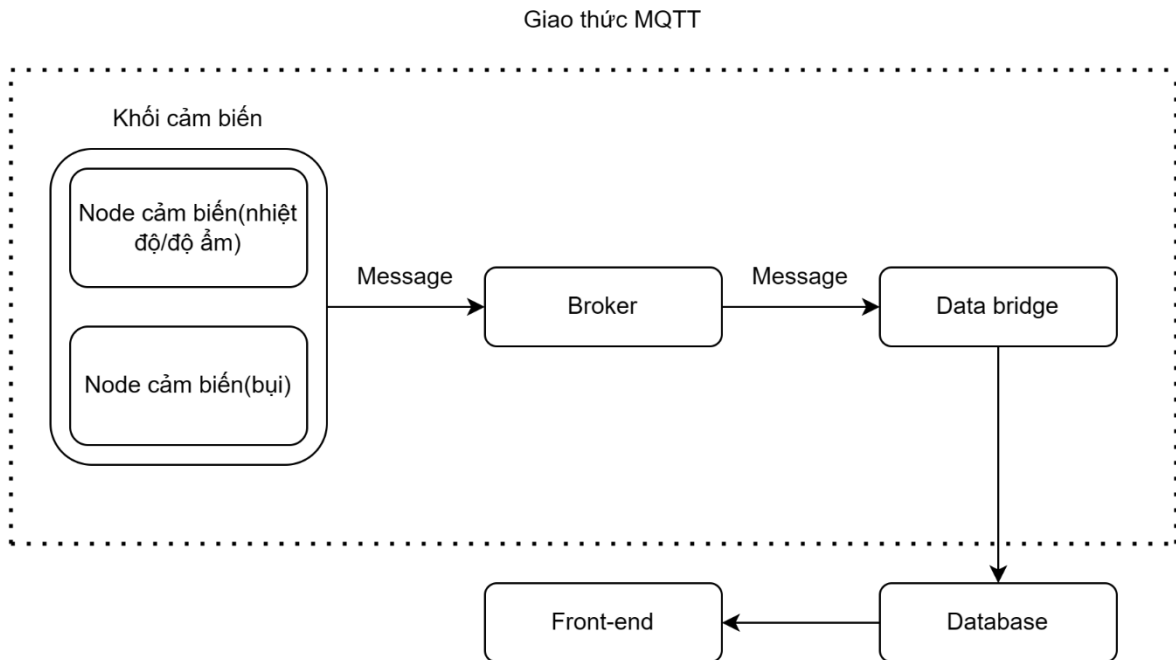
- Môi trường trong nhà
- Nhiệt độ: 0 đến 50 độ C
- Độ ẩm: 20% đến 80%
- Áp suất không khí: 86 đến 106 kPa

- Giá: 2.000.000VND

- Nhân lực: Kỹ sư phần cứng và phần mềm

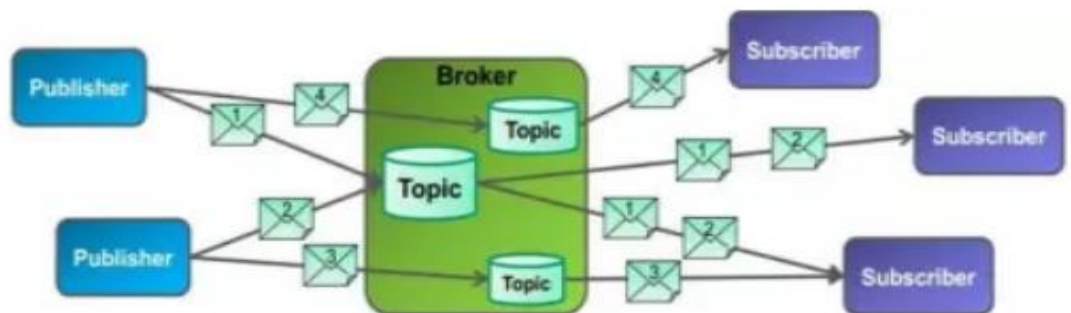
III. Giải pháp: Kiến trúc/Thuật toán

1. Sơ đồ khối và giao thức:



Hình 8: Sơ đồ khối

- Lựa chọn giao thức giữa MQTT client (Publisher/Subscriber) và MQTTServer (Broker):



+ Giao thức được sử dụng để truyền tải các message là giao thức MQTT(Message Queuing Telemetry Transport). MQTT được thiết kế để truyền tải dữ liệu trong mạng có băng thông thấp và không đáng tin cậy. Nó sử dụng giao thức TCP/IP để kết nối với các thiết bị trên mạng.

+MQTT Broker là trung tâm xử lý và điều phối các tin nhắn giữa các MQTT client. Nó có nhiệm vụ chịu trách nhiệm cho việc chuyển tiếp và định tuyến các tin nhắn giữa các MQTT publishers và MQTT subscribers. Các thông điệp được gửi từ MQTT clients đến MQTT Broker thông qua giao thức

TCP/IP, sau đó được lưu trữ trong một message queue để đợi cho các MQTT subscribers yêu cầu.

+ Trong quá trình truyền tải, MQTT sử dụng các topic để xác định các loại dữ liệu và các thiết bị sẽ nhận được dữ liệu đó. MQTT subscribers có thể đăng ký vào các topic để nhận các tin nhắn từ MQTT publishers. Khi một MQTT publisher gửi tin nhắn đến một topic, MQTT Broker sẽ chuyển tiếp tin nhắn đó đến tất cả các MQTT subscribers đã đăng ký vào topic đó.

+ Ở trên sơ đồ khối, publisher chính là khối cảm biến, còn subscriber là Data bridge.

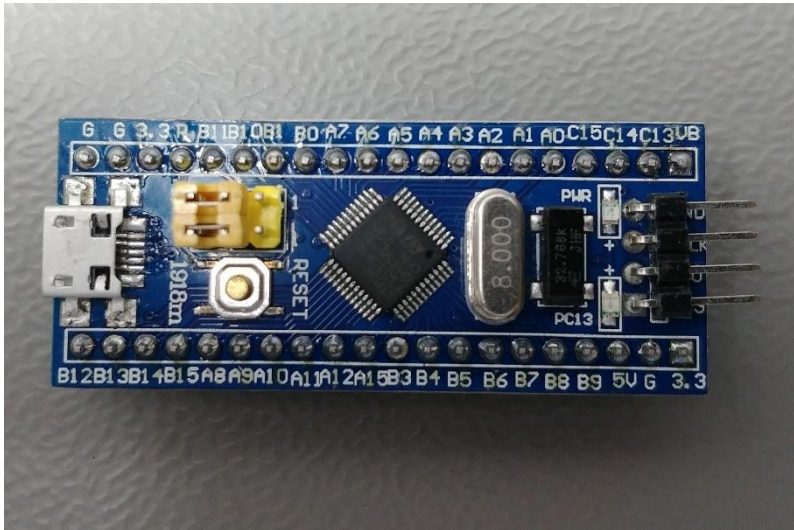
2. Vai trò của các khối:

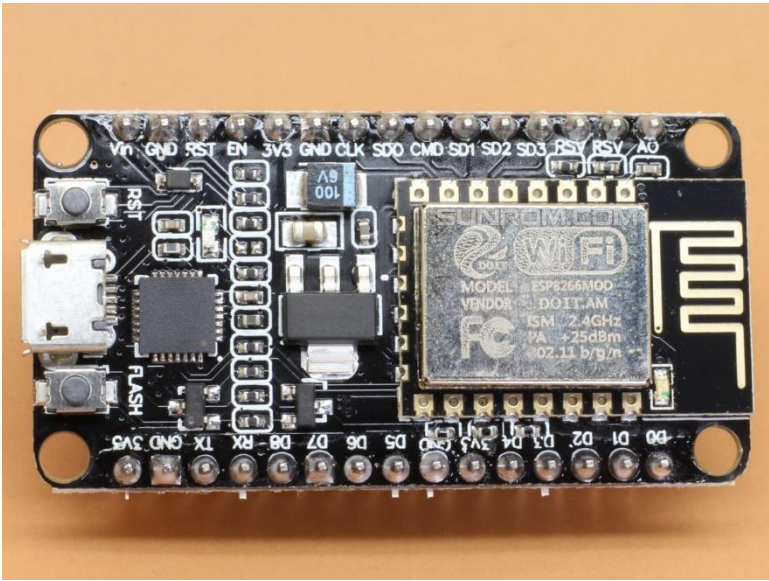
- **Khối cảm biến:**
 - Gửi thông tin dựa trên hiện trạng môi trường xung quanh (trong vùng hoạt động của khối cảm biến)
- **Broker:**
 - Broker nhận những thông tin subscribe (Subscriptions) từ publisher, nhận thông điệp, chuyển những thông điệp đến các Subscriber tương ứng dựa trên Subscriptions từ client.
- **Data bridge:**
 - Đưa luồng thông tin nhận được từ broker đến Database và phân loại, xử lý dữ liệu nhận được.
- **Database:**
 - Nhận thông tin nhận được từ data bridge và lưu trữ.
- ☒ **Front-end:**
 - Lấy được dữ liệu từ database và thể hiện dữ liệu cho người dùng.

IV. Triển khai

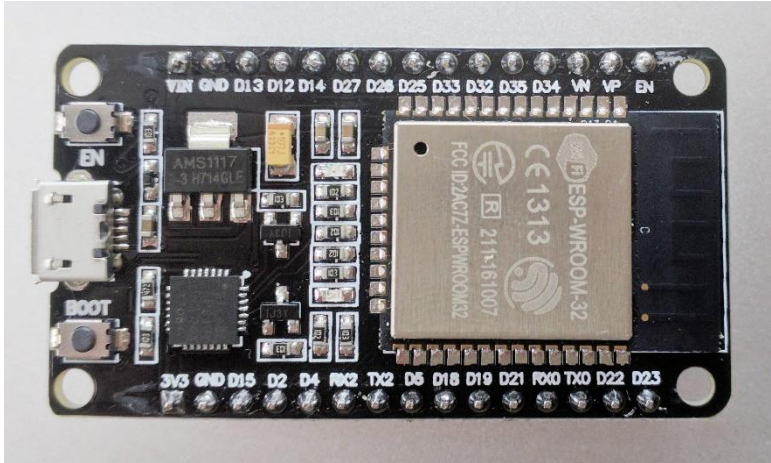
1.1. Linh kiện:

- **Các vi xử lý:**


	STM32F103C8T6
Hình ảnh	 <p>The image shows a blue printed circuit board (PCB) for the STM32F103C8T6 microcontroller. It features a central black integrated circuit (IC) with a gold-colored pin header on the left. A yellow USB-to-UART bridge chip is visible on the left side. A silver push-button labeled 'RESET' is located near the center. A small black component labeled 'PC13' is on the right. The board has two rows of gold pins along the top and bottom edges, labeled with pin numbers and functions like 'B1', 'B2', 'B3', 'B4', 'B5', 'B6', 'B7', 'B8', 'B9', '5V', 'G', '3.3', 'A1', 'A2', 'A3', 'A4', 'A5', 'A6', 'A7', 'A8', 'A9', 'A10', 'A11', 'A12', 'A13', 'A14', 'A15', 'A16', 'A17', 'A18', 'A19', 'A20', 'A21', 'A22', 'A23', 'A24', 'A25', 'A26', 'A27', 'A28', 'A29', 'A30', 'A31', 'A32', 'A33', 'A34', 'A35', 'A36', 'A37', 'A38', 'A39', 'A40', 'A41', 'A42', 'A43', 'A44', 'A45', 'A46', 'A47', 'A48', 'A49', 'A50', 'A51', 'A52', 'A53', 'A54', 'A55', 'A56', 'A57', 'A58', 'A59', 'A60', 'A61', 'A62', 'A63', 'A64', 'A65', 'A66', 'A67', 'A68', 'A69', 'A70', 'A71', 'A72', 'A73', 'A74', 'A75', 'A76', 'A77', 'A78', 'A79', 'A80', 'A81', 'A82', 'A83', 'A84', 'A85', 'A86', 'A87', 'A88', 'A89', 'A90', 'A91', 'A92', 'A93', 'A94', 'A95', 'A96', 'A97', 'A98', 'A99', 'A100', 'A101', 'A102', 'A103', 'A104', 'A105', 'A106', 'A107', 'A108', 'A109', 'A110', 'A111', 'A112', 'A113', 'A114', 'A115', 'A116', 'A117', 'A118', 'A119', 'A120', 'A121', 'A122', 'A123', 'A124', 'A125', 'A126', 'A127', 'A128', 'A129', 'A130', 'A131', 'A132', 'A133', 'A134', 'A135', 'A136', 'A137', 'A138', 'A139', 'A140', 'A141', 'A142', 'A143', 'A144', 'A145', 'A146', 'A147', 'A148', 'A149', 'A150', 'A151', 'A152', 'A153', 'A154', 'A155', 'A156', 'A157', 'A158', 'A159', 'A160', 'A161', 'A162', 'A163', 'A164', 'A165', 'A166', 'A167', 'A168', 'A169', 'A170', 'A171', 'A172', 'A173', 'A174', 'A175', 'A176', 'A177', 'A178', 'A179', 'A180', 'A181', 'A182', 'A183', 'A184', 'A185', 'A186', 'A187', 'A188', 'A189', 'A190', 'A191', 'A192', 'A193', 'A194', 'A195', 'A196', 'A197', 'A198', 'A199', 'A200', 'A201', 'A202', 'A203', 'A204', 'A205', 'A206', 'A207', 'A208', 'A209', 'A210', 'A211', 'A212', 'A213', 'A214', 'A215', 'A216', 'A217', 'A218', 'A219', 'A220', 'A221', 'A222', 'A223', 'A224', 'A225', 'A226', 'A227', 'A228', 'A229', 'A230', 'A231', 'A232', 'A233', 'A234', 'A235', 'A236', 'A237', 'A238', 'A239', 'A240', 'A241', 'A242', 'A243', 'A244', 'A245', 'A246', 'A247', 'A248', 'A249', 'A250', 'A251', 'A252', 'A253', 'A254', 'A255', 'A256', 'A257', 'A258', 'A259', 'A260', 'A261', 'A262', 'A263', 'A264', 'A265', 'A266', 'A267', 'A268', 'A269', 'A270', 'A271', 'A272', 'A273', 'A274', 'A275', 'A276', 'A277', 'A278', 'A279', 'A280', 'A281', 'A282', 'A283', 'A284', 'A285', 'A286', 'A287', 'A288', 'A289', 'A290', 'A291', 'A292', 'A293', 'A294', 'A295', 'A296', 'A297', 'A298', 'A299', 'A300', 'A301', 'A302', 'A303', 'A304', 'A305', 'A306', 'A307', 'A308', 'A309', 'A310', 'A311', 'A312', 'A313', 'A314', 'A315', 'A316', 'A317', 'A318', 'A319', 'A320', 'A321', 'A322', 'A323', 'A324', 'A325', 'A326', 'A327', 'A328', 'A329', 'A330', 'A331', 'A332', 'A333', 'A334', 'A335', 'A336', 'A337', 'A338', 'A339', 'A340', 'A341', 'A342', 'A343', 'A344', 'A345', 'A346', 'A347', 'A348', 'A349', 'A350', 'A351', 'A352', 'A353', 'A354', 'A355', 'A356', 'A357', 'A358', 'A359', 'A360', 'A361', 'A362', 'A363', 'A364', 'A365', 'A366', 'A367', 'A368', 'A369', 'A370', 'A371', 'A372', 'A373', 'A374', 'A375', 'A376', 'A377', 'A378', 'A379', 'A380', 'A381', 'A382', 'A383', 'A384', 'A385', 'A386', 'A387', 'A388', 'A389', 'A390', 'A391', 'A392', 'A393', 'A394', 'A395', 'A396', 'A397', 'A398', 'A399', 'A400', 'A401', 'A402', 'A403', 'A404', 'A405', 'A406', 'A407', 'A408', 'A409', 'A410', 'A411', 'A412', 'A413', 'A414', 'A415', 'A416', 'A417', 'A418', 'A419', 'A420', 'A421', 'A422', 'A423', 'A424', 'A425', 'A426', 'A427', 'A428', 'A429', 'A430', 'A431', 'A432', 'A433', 'A434', 'A435', 'A436', 'A437', 'A438', 'A439', 'A440', 'A441', 'A442', 'A443', 'A444', 'A445', 'A446', 'A447', 'A448', 'A449', 'A450', 'A451', 'A452', 'A453', 'A454', 'A455', 'A456', 'A457', 'A458', 'A459', 'A460', 'A461', 'A462', 'A463', 'A464', 'A465', 'A466', 'A467', 'A468', 'A469', 'A470', 'A471', 'A472', 'A473', 'A474', 'A475', 'A476', 'A477', 'A478', 'A479', 'A480', 'A481', 'A482', 'A483', 'A484', 'A485', 'A486', 'A487', 'A488', 'A489', 'A490', 'A491', 'A492', 'A493', 'A494', 'A495', 'A496', 'A497', 'A498', 'A499', 'A500', 'A501', 'A502', 'A503', 'A504', 'A505', 'A506', 'A507', 'A508', 'A509', 'A510', 'A511', 'A512', 'A513', 'A514', 'A515', 'A516', 'A517', 'A518', 'A519', 'A520', 'A521', 'A522', 'A523', 'A524', 'A525', 'A526', 'A527', 'A528', 'A529', 'A530', 'A531', 'A532', 'A533', 'A534', 'A535', 'A536', 'A537', 'A538', 'A539', 'A540', 'A541', 'A542', 'A543', 'A544', 'A545', 'A546', 'A547', 'A548', 'A549', 'A550', 'A551', 'A552', 'A553', 'A554', 'A555', 'A556', 'A557', 'A558', 'A559', 'A560', 'A561', 'A562', 'A563', 'A564', 'A565', 'A566', 'A567', 'A568', 'A569', 'A570', 'A571', 'A572', 'A573', 'A574', 'A575', 'A576', 'A577', 'A578', 'A579', 'A580', 'A581', 'A582', 'A583', 'A584', 'A585', 'A586', 'A587', 'A588', 'A589', 'A590', 'A591', 'A592', 'A593', 'A594', 'A595', 'A596', 'A597', 'A598', 'A599', 'A600', 'A601', 'A602', 'A603', 'A604', 'A605', 'A606', 'A607', 'A608', 'A609', 'A610', 'A611', 'A612', 'A613', 'A614', 'A615', 'A616', 'A617', 'A618', 'A619', 'A620', 'A621', 'A622', 'A623', 'A624', 'A625', 'A626', 'A627', 'A628', 'A629', 'A630', 'A631', 'A632', 'A633', 'A634', 'A635', 'A636', 'A637', 'A638', 'A639', 'A640', 'A641', 'A642', 'A643', 'A644', 'A645', 'A646', 'A647', 'A648', 'A649', 'A650', 'A651', 'A652', 'A653', 'A654', 'A655', 'A656', 'A657', 'A658', 'A659', 'A660', 'A</p>

	ESP8266
Hình ảnh	 <p>Hình 10: ESP8266</p>
Giá thành	~80.000VND
Chip	Wi-Fi SOC tích hợp
Kiến trúc	RISC 32-bit
Tốc độ xử lý	80MHz
Bộ nhớ	Flash 512KB hoặc 1MB SRAM 80KB
Giao tiếp	Wi-Fi 802.11 b/g/n UART, I2C, SPI, GPIO
Hỗ trợ	chế độ điện tiết kiệm
Điện áp hoạt động	2.5V-3.6V

Bảng 4: Thông số ESP8266

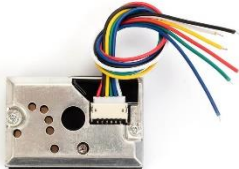
	ESP32
Hình ảnh	 <p>Hình 11: ESP8266</p>
Giá thành	~60.000VND
Kiến trúc	SoC hai nhân Xtensa LX6 32-bit
Tốc độ xử lý	240MHz
Bộ nhớ	Flash 4MB SRAM 520KB
Giao tiếp	Wi-Fi 802.11 b/g/n, Bluetooth 4.2 UART, I2C, SPI, GPIO
Hỗ trợ	chế độ điện tiết kiệm
Điện áp hoạt động	2.2V-3.6V

Bảng 5: Thông số ESP32


	Raspberry Pi 4
Hình ảnh	 <p>Hình 12: Raspberry Pi 4</p>
Giá thành	~1.300.000VND
Kiến trúc	SoC Broadcom BCM2711 Cortex-A72 (ARM v8) 64-bit
Tốc độ xử lý	1.5GHz
Bộ nhớ	RAM 1GB, 2GB, 4GB hoặc 8GB
Hỗ trợ độ phân giải	4K qua HDM
Giao tiếp	Gigabit Ethernet USB 3.0, USB 2.0, và USB-C
Hỗ trợ	Wi-Fi 802.11ac và Bluetooth 5.0 Truy cập vào cơ sở dữ liệu MySQL, PostgreSQL và MariaDB thông qua các thư viện Python hoặc Node.js.

Bảng 6: Thông số Raspberry Pi 4

- **Các cảm biến bụi**


	GP2Y10
Hình ảnh	 <p>Hình 13: GP2Y10</p>
Giá thành	~400.000VND
Khoảng đo	0.8 đến 2.8 μm
Độ chính xác	$\pm 15\%$ đối với kích thước hạt từ 0.5 μm đến 2.5 μm
Điện áp hoạt động	5VDC
Dòng tiêu thụ	33mA
Giao tiếp	Analog

Bảng 7: Thông số GP2Y10

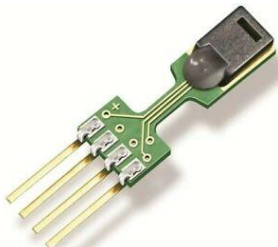
	PMS7003
Hình ảnh	 <p>Hình 14: PMS7003</p>
Giá thành	39.99\$
Kích thước hạt	1.0 μ m, 2.5 μ m, 10 μ m
Độ nhạy	0.3V/ (0.1mg/m ³)
Khoảng đo	0~500 μ g/m ³
Độ phân giải	1 μ g/m ³
Sai số	± 10 μ g/m ³ (0100 μ g/m ³), $\pm 10\%$ (100500 μ g/m ³)
Thời gian đáp ứng	<10 giây
Điện áp hoạt động	5VDC
Điện năng tiêu thụ	90mA (hoạt động), 2mA (chờ)
Giao tiếp	UART (TTL)

Bảng 8: Thông số PMS7003


- Các cảm biến nhiệt độ/độ ẩm

	DHT22
Hình ảnh	 <p>Hình 15: DHT22</p>
Giá thành	~40.000VND
Khoảng đo nhiệt độ	-40 đến 80 độ C (-40 đến 176 độ F)
Sai số đo nhiệt độ	±0.5 độ C
Khoảng đo độ ẩm	0 đến 100% RH
Độ phân giải	nhiệt độ: 0.1 độ C độ ẩm: 0.1% RH
Dòng điện hoạt động	2.5mA tại 5V
Điện áp hoạt động	3V đến 5.5V

Bảng 9: Thông số DHT22

	SHT71
Hình ảnh	 <p>Hình 16: SHT71</p>
Giá thành	~500.000VND
Độ chính xác nhiệt độ	$\pm 0,4$ độ C
Độ chính xác độ ẩm	$\pm 3\%RH$
Khoảng đo nhiệt độ	-40 đến 123,8 độ C
Khoảng đo độ ẩm	0 đến 100%RH
Độ phân giải nhiệt độ	0,01 độ C
Độ phân giải độ ẩm	0,04%RH
Điện áp hoạt động	2,1 đến 3,6V
Tiêu thụ điện năng	80 μW
Giao tiếp	2-wire digital serial interface

Bảng 10: Thông số SHT71

	DHT11
Hình ảnh	 <p>Hình 17: DHT11</p>
Giá thành	~20.000VND
Độ chính xác đo nhiệt độ	$\pm 2^{\circ}\text{C}$
Độ chính xác đo độ ẩm	$\pm 5\%$
Dải đo nhiệt độ	0 đến 50°C
Dải đo độ ẩm	20% đến 90%
Thời gian đọc dữ liệu.	khoảng 2 giây
Điện áp hoạt động	3V đến 5,5V

Bảng 11: Thông số DHT11

- **Lựa chọn phần cứng:**

Nhóm sử dụng các linh kiện sau:

- Cảm biến nhiệt độ/độ ẩm : **DHT11**: do giá thành rẻ hơn nhưng vẫn đảm bảo được yêu cầu chỉ tiêu chức năng

- Cảm biến bụi: **GP2Y10**: do đã có sẵn cảm biến

- Các vi xử lý: **ESP32, ESP8266**: do giá thành rẻ và đủ cho nhu cầu sử dụng của dự án

- Vi xử lý **Raspberry Pi 4 Model B**: RAM lớn và khả năng xử lý mạnh nên có thể đảm nhiệm vai trò của các khối Broker, Data bridge, Database và Front-end.

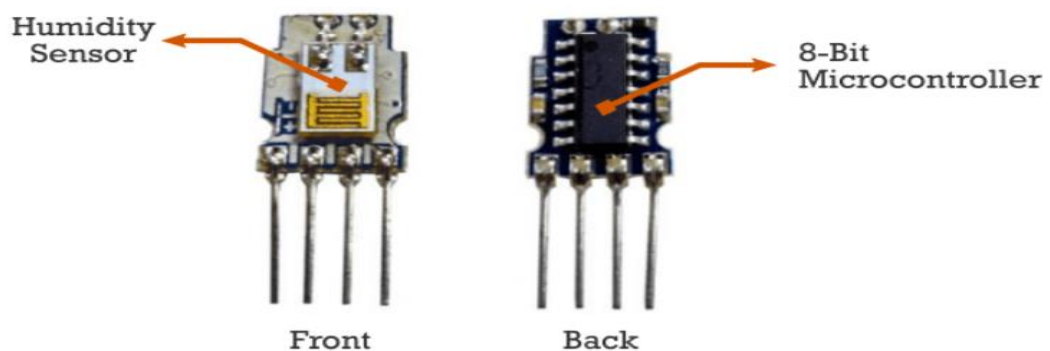
- **Lựa chọn phần mềm**

- Broker: **Mosquitto**: Hoàn toàn có thể thay bằng các broker khác như ActiveMQ, Mosca,...
- Data bridge: **Node-RED**: Lựa chọn Node-RED do Node-RED giúp việc sử dụng giao thức MQTT dễ dàng hơn
- Database: **InfluxDB**: hoàn toàn có thể thay thế bằng các database khác như MySQL, MongoDB hay MariaDB.
- Front-end : **Grafana**: giao diện đẹp, dễ sử dụng với nhiều phần mềm Database.

1.2. Khối cảm biến

1) Nguyên lý cảm biến:

- Cảm biến nhiệt DHT11:



+ Cảm biến DHT11 bao gồm một phần tử cảm biến độ ẩm điện dung và một điện trở nhiệt để cảm nhận nhiệt độ. Tự điện cảm biến độ ẩm có hai điện cực với chất nền giữ ẩm làm chất điện môi giữa chúng. Thay đổi giá trị điện dung xảy ra với sự thay đổi của các mức độ ẩm. IC đo, xử lý các giá trị điện trở đã thay đổi này và chuyển chúng thành dạng kỹ thuật số.

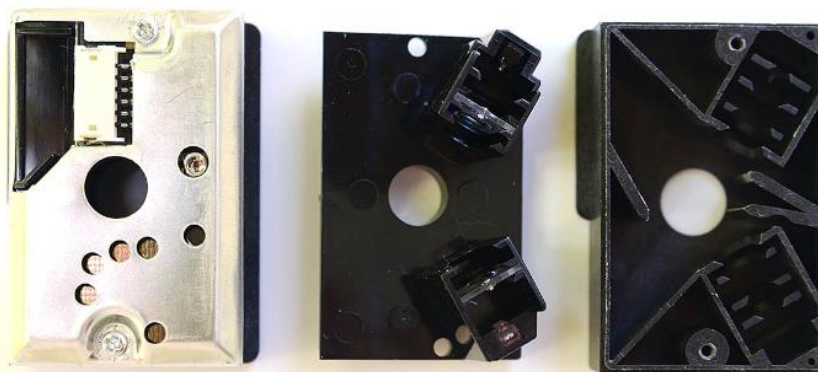
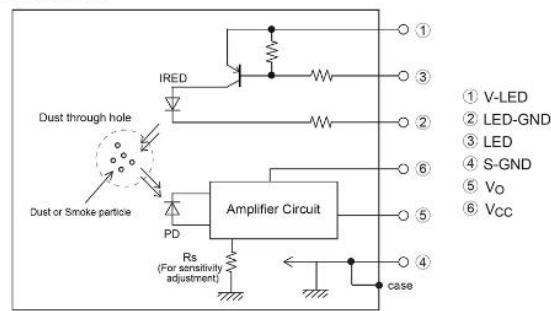
+ Để đo nhiệt độ, cảm biến này sử dụng một nhiệt điện trở có hệ số nhiệt độ âm, làm giảm giá trị điện trở của nó khi nhiệt độ tăng.

+ Cảm Biến Nhiệt Độ Và Độ Ẩm DHT11 là cảm biến rất thông dụng hiện nay vì chi phí rẻ và rất dễ lấy dữ liệu thông qua giao tiếp 1 wire (giao tiếp digital 1 dây truyền dữ liệu duy nhất). Bộ tiền xử lý tín hiệu tích hợp trong cảm biến giúp bạn có được dữ liệu chính xác mà không phải qua bất kỳ tính toán nào. So với cảm biến đời mới hơn là DHT22 thì DHT11 cho khoảng đo và độ chính xác kém hơn rất nhiều.

NTC Thermistor -100 °C Temperature 200 °C

- Cảm biến bụi:

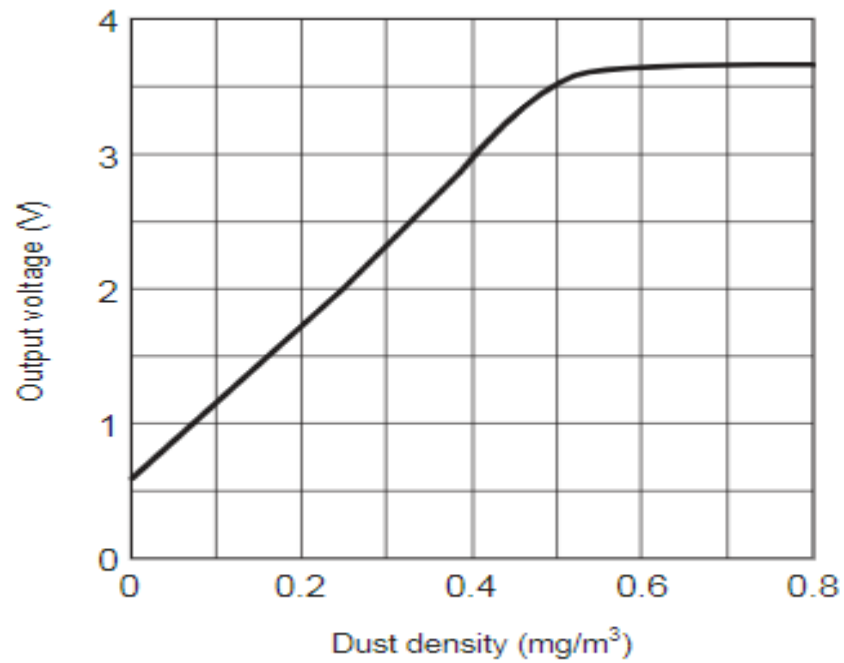
■ Internal schematic



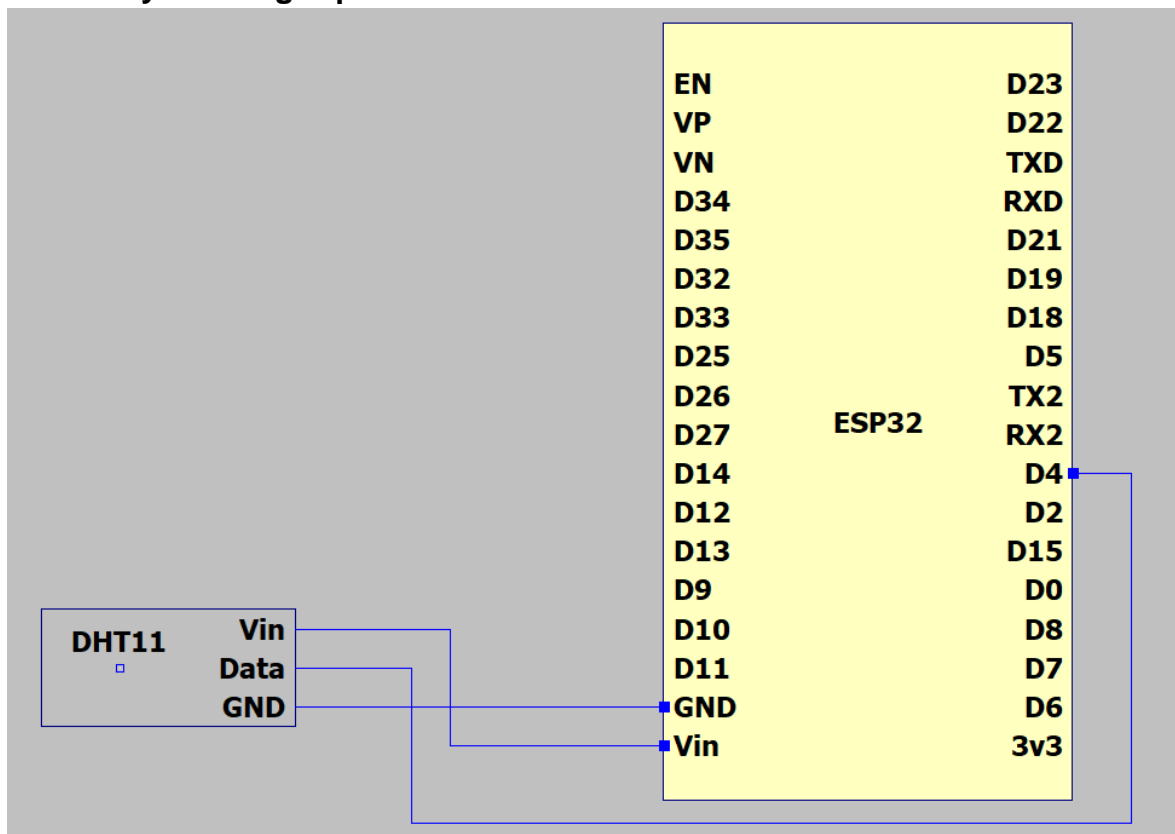
- Gồm 03 phần chính:
 - + IR LED
 - + Phototransistor
 - + Amplifier

- Có 02 bộ phận dùng để truyền và nhận hồng ngoại (IR LED và Phototransistor). 02 bộ phận này được đặt chệch góc với nhau. Khi có bụi bay vào, tia hồng ngoại từ IR LED sẽ bị dội vào Phototransistor, lúc này điện áp từ phototransistor sẽ được đưa đến mạch khuếch đại (Amplifier) và xuất ra chân Vo.

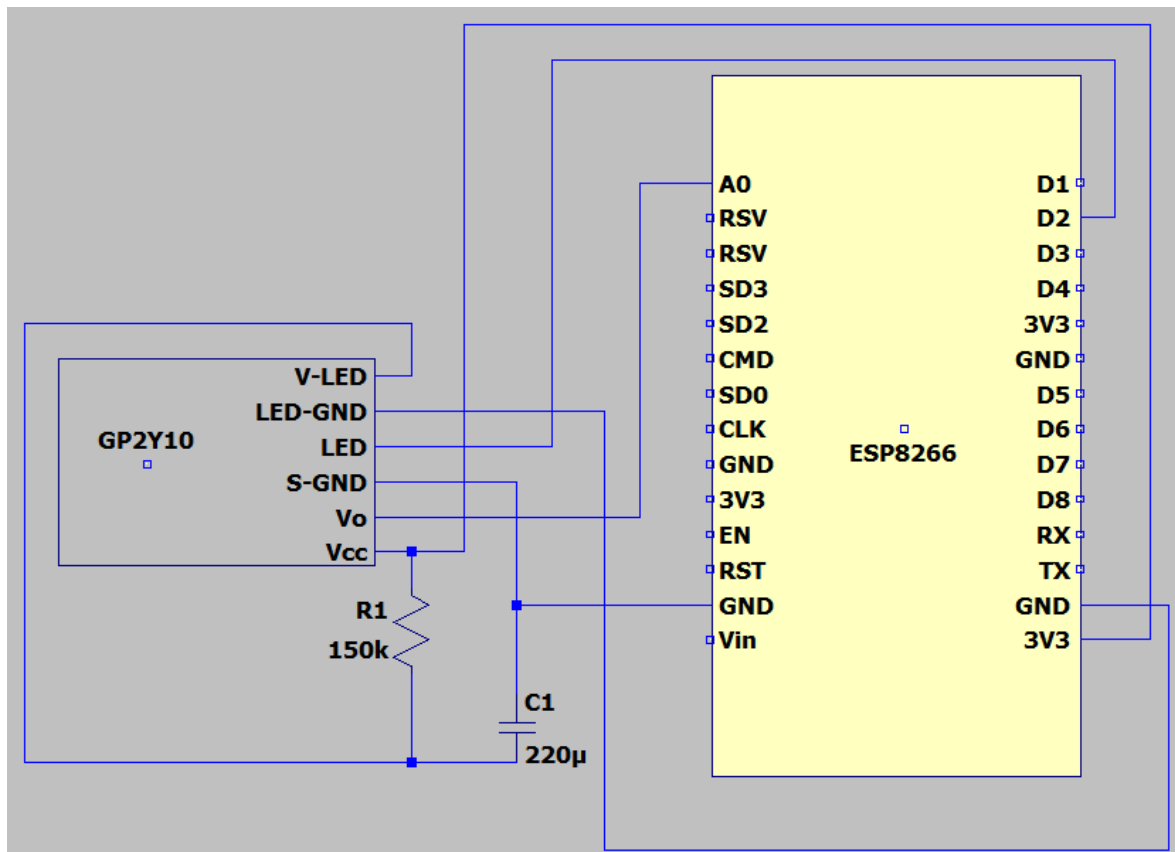
- Theo datasheet, một lần đo của cảm biến sẽ mất khoảng 10ms để đo. Trong 10ms sẽ có: 0.32ms: Trong khoảng thời gian này, IR LED sẽ được bật lên và chúng ta tiến hành đọc giá trị. Tuy nhiên, chỉ được phép đọc giá trị sau 0.28ms => cần làm trong 0.32ms này đó là: Bật IR LED. Delay 0.28ms. Tắt IR LED. Delay 0.04ms. 9.680ms: Thời gian này cảm biến sẽ không làm gì cả => chỉ cần delay 9.680ms.



2. Layout bảng mạch:



Sơ đồ mạch cảm biến nhiệt độ/độ ẩm



Sơ đồ cảm biến bụi

3. Lập trình trên các cảm biến đo

Các thư viện được sử dụng: WiFi.h, PubSubClient.h, ArduinoJson.h, DHTesp.h, ESP8266WiFi.h

Sử dụng ngôn ngữ lập trình: C

Sử dụng phần mềm: Arduino

+ Cảm biến nhiệt với ESP32:

```

bool getTemperature() {

    TempAndHumidity newValues = dht.getTempAndHumidity();
    if (dht.getStatus() != 0) {
        Serial.println("DHT11 error status: " + String(dht.getStatusString()));
        return false;
    }

    float heatIndex = dht.computeHeatIndex(newValues.temperature, newValues.humidity);
    float dewPoint = dht.computeDewPoint(newValues.temperature, newValues.humidity);
    float cr = dht.getComfortRatio(cf, newValues.temperature, newValues.humidity);

    String comfortStatus;
    switch(cf) {
        case Comfort_OK:           comfortStatus = "Comfort_OK";           break;
        case Comfort_TooHot:        comfortStatus = "Comfort_TooHot";        break;
        case Comfort_TooCold:       comfortStatus = "Comfort_TooCold";       break;
        case Comfort_TooDry:        comfortStatus = "Comfort_TooDry";        break;
        case Comfort_TooHumid:      comfortStatus = "Comfort_TooHumid";      break;
        case Comfort_HotAndHumid:   comfortStatus = "Comfort_HotAndHumid";   break;
        case Comfort_HotAndDry:     comfortStatus = "Comfort_HotAndDry";     break;
        case Comfort_ColdAndHumid:  comfortStatus = "Comfort_ColdAndHumid";  break;
        case Comfort_ColdAndDry:    comfortStatus = "Comfort_ColdAndDry";    break;
        default:                   comfortStatus = "Unknown:";               break;
    };
    Serial.println(" T:" + String(newValues.temperature) + " H:" + String(newValues.humidity)
    + " I:" + String(heatIndex) + " D:" + String(dewPoint) + " " + comfortStatus);
    return true;
}

```

+ Cảm biến bụi 8266:

```

digitalWrite(ledPower,LOW); // Bật IR LED
delayMicroseconds(samplingTime); //Delay 0.28ms
voMeasured = analogRead(measurePin); // Đọc giá trị ADC V0
delayMicroseconds(deltaTime); //Delay 0.04ms
digitalWrite(ledPower,HIGH); // Tắt LED
delayMicroseconds(sleepTime); //Delay 9.68ms

// Tính điện áp từ giá trị ADC
calcVoltage = voMeasured * (5.0 / 1024); //Điện áp Vcc của cảm biến (5.0 hoặc 3.3)

// Linear Equation http://www.howmuchsnow.com/arduino/airquality/
// Chris Nafis (c) 2012
dustDensity = 0.17 * calcVoltage - 0.1;

```

+ Cài đặt Wifi:


```

void setup_wifi() {

    delay(10);
    // We start by connecting to a WiFi network
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(ssid);

    WiFi.mode(WIFI_STA);
    WiFi.begin(ssid, password);

    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }

    randomSeed(micros());

    Serial.println("");
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
}

```

+ Gửi dữ liệu:

```

void loop() {

    if (!client.connected()) {
        reconnect();
    }
    client.loop();
    float temp;
    float humid;
    Serial.println("Cam bien nhiet DHT11 ");
    getTemperature();
    TempAndHumidity newValues = dht.getTempAndHumidity();

    temp = newValues.temperature;
    humid = newValues.humidity;
    StaticJsonDocument<1024> doc;
    char output[100];
    doc["temp"] = temp;
    doc["humid"] = humid;
    Serial.println("Read");

    serializeJson(doc, output);
    Serial.println(output);
    client.publish("home/esp32", output);
    Serial.println("Sent");
    delay(5000);

}

```

Bảng dữ liệu của nhiệt

4. Khối Server

- Bài toán cảm biến không khí đặt ra yêu cầu cần sử dụng trong một không gian không quá lớn như nhà ở, phòng làm việc,... Chính vì thế nhóm đã sử dụng một phần mềm builder các phần mềm trên nền docker tên là IOTstack.

- Trong tool này, chúng ta có thể build stack gồm các ứng dụng mà ta muốn cho một hệ thống IoT, bao gồm các ứng dụng cần sử dụng cho project này bao gồm

Mosquitto (MQTT Broker), Node-RED (Data Bridge), InfluxDB (Database), Grafana (Front-end).

- Trước tiên ta update các phần mềm và OS của Raspberry Pi bằng lệnh
“sudo apt update && upgrade”

- Tiếp đến ta thực hiện tải IOTstack:

- “curl -fsSL

- <https://raw.githubusercontent.com/Sensorslot/IOTstack/master/install.sh> | bash”

- sudo shutdown -r now”

- Khi Pi hoạt động trở lại thì chúng ta chuyển về directory chứa IOTstack để khởi động trình menu script:

- “cd IOTstack/

- ./menu.sh”

- Tại menu này ta sẽ thấy các lựa chọn để chọn các ứng dụng ta muốn để đưa vào file docker-compose.yml. Ta sẽ chọn các ứng dụng Mosquitto, Node-RED, InfluxDB, Grafana và Portainer.

- Ta có thể kiểm tra địa chỉ của các port bằng lệnh: *“docker-compose ps”*

4.1. MQTT Broker

- Với MQTT broker, nhóm quyết định sử dụng phần mềm Mosquitto

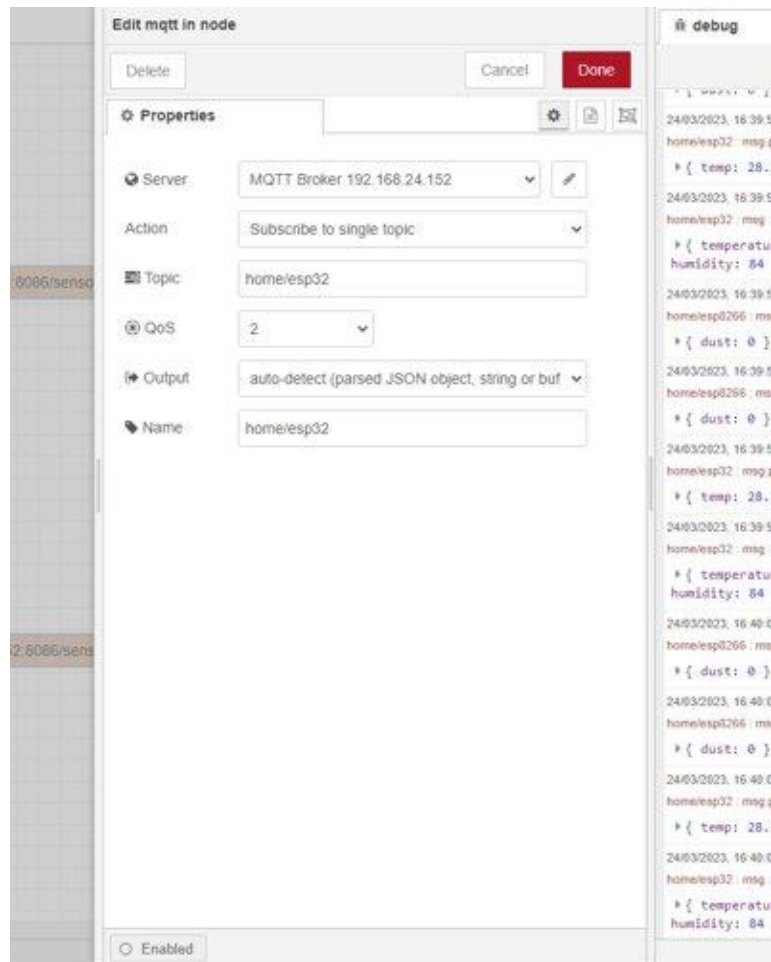
- Mosquitto là một MQTT Broker mã nguồn mở cho phép thiết bị truyền nhận dữ liệu theo giao thức MQTT version 5.0, 3.1.1 và 3.1 – Một giao thức nhanh, nhẹ theo mô hình publish/subscribe được sử dụng rất nhiều trong lĩnh vực Internet of Things. Mosquitto cung cấp một thư viện viết bằng ngôn ngữ C để triển khai các MQTT Client và có thể dễ dàng sử dụng bằng dòng lệnh: “mosquitto_pub” và “mosquitto_sub”.

- Ưu điểm:

- Ưu điểm nổi bật của Mosquitto là tốc độ truyền nhận và xử lý dữ liệu nhanh, độ ổn định cao, được sử dụng rộng rãi và phù hợp với những ứng dụng embedded.
 - Mosquitto rất nhẹ và phù hợp để sử dụng trên tất cả các thiết bị.
 - Ngoài ra, Mosquitto cũng được hỗ trợ các giao thức TLS/SSL (các giao thức nhằm xác thực server và client, mã hóa các message để bảo mật dữ liệu).

- Nhược điểm:

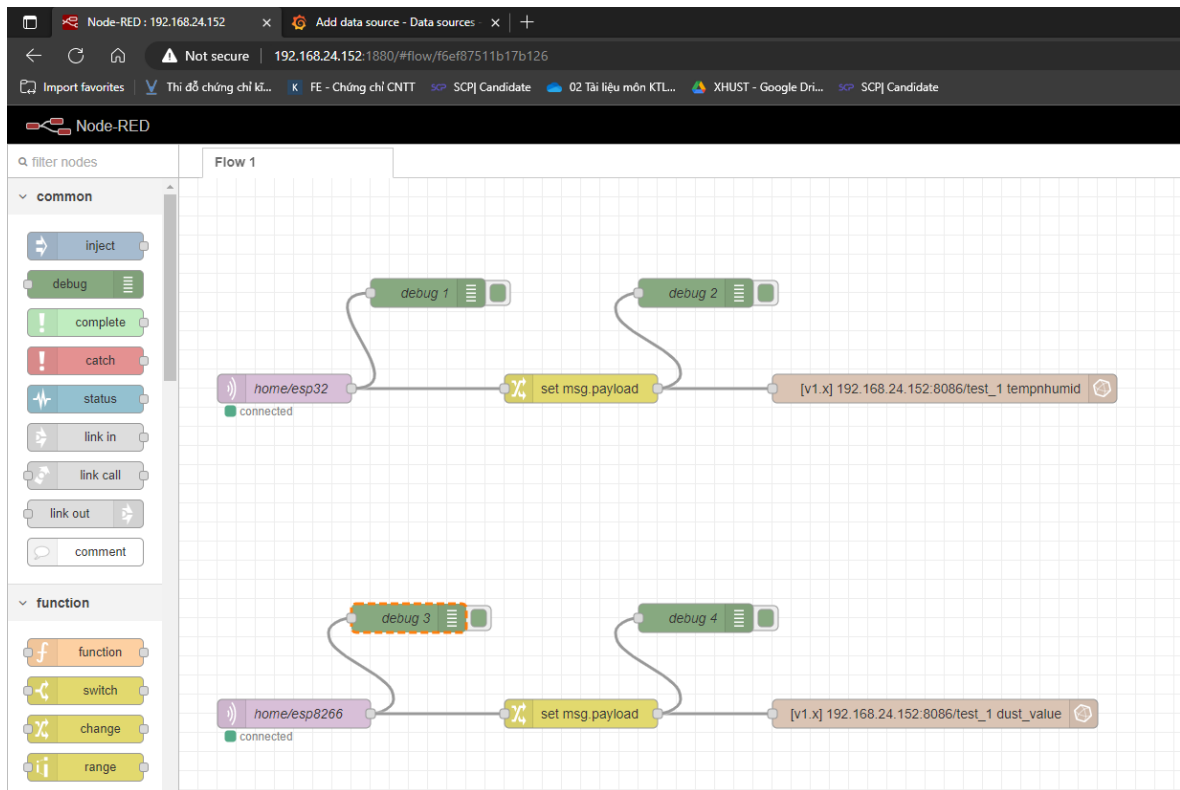
- Một số nhược điểm của mosquitto là khó thiết kế khi làm những ứng dụng lớn và ít phương thức xác thực thiết bị nên khả năng bảo mật vẫn chưa tối ưu.



4.2. Data Bridge:

- Node RED là một công cụ lập trình dùng để kết nối các thiết bị phần cứng, API và các dịch vụ trực tuyến với nhau. Về cơ bản, đây là một công cụ trực quan được thiết kế cho IoT (Internet of Things), nhưng cũng có thể được sử dụng cho các ứng dụng khác nhằm liên kết nhanh các luồng (flow) dịch vụ khác nhau.

- Node-RED cho phép người dùng kết hợp các dịch vụ Web và phần cứng bằng cách thay thế các tác vụ mã hóa cấp thấp phổ biến (như một dịch vụ đơn giản giao tiếp với một cổng nối tiếp) và điều này có thể được thực hiện với giao diện kéo thả trực quan. Các thành phần khác nhau trong Node-RED được kết nối với nhau để tạo ra một luồng (flow). Hầu hết mã lệnh (code) cần thiết được tạo tự động.



- Đầu tiên ta truy cập vào địa chỉ của database dưới dạng: địa chỉ host_server:8086
- Sau đó ta set các node MQTT là home/esp32 và home/esp8266 để nhận được các message từ các publishers là các node cảm biến.
- Tiếp đến ta set các node set msg.payload để chuyển các gói message về thành dạng JSON file.
- Cuối cùng là các JSON file được chuyển về node database được tạo ra từ trước. Ta cũng cần đồng thời tạo ra một database mới trước khi chuyển dữ liệu vào và tạo các measurement ở đây là các thông số đo. Mục đích của việc này là để tạo được các measurement nhằm hiển thị trên giao diện front-end của Grafana.

4.3. Database:

- InfluxDB là một database được tối ưu hóa để xử lý dữ liệu chuỗi thời gian (các dãy số được lập chỉ mục theo thời gian). InfluxDB được sử dụng để lưu các dữ liệu cho các trường hợp liên quan đến một lượng lớn time-stamped data, bao gồm DevOps monitoring, log data, application metrics, IoT sensor data, và real-time analytics. Nó có thể tự động xóa các dữ liệu cũ, không cần thiết và cung cấp một ngôn ngữ giống SQL để tương tác với dữ liệu.
- Influxdb có lợi thế lớn trong việc xử lý lượng time-stamped data lớn. Tuy vậy, giới hạn về kiểu dữ liệu là một hạn chế đối với Influxdb.
- Để tạo database ta cần nhập influx docker container và nhập database:

```
docker exec -it influxdb influx
CREATE DATABASE sensor_data
Quit
```

Edit influxdb out node

Delete Cancel Done

Properties

Name

Server [v1.x] 192.168.24.152:8086/sensor_dat

Measurement tempnhumid

☐ Advanced Query Options

Tip: If no retention policy is specified, **autogen** will be assumed.

Edit influxdb out node

Delete Cancel Done

Properties

Name

Server [v1.x] 192.168.24.152:8086/sensor_dat

Measurement dust_value

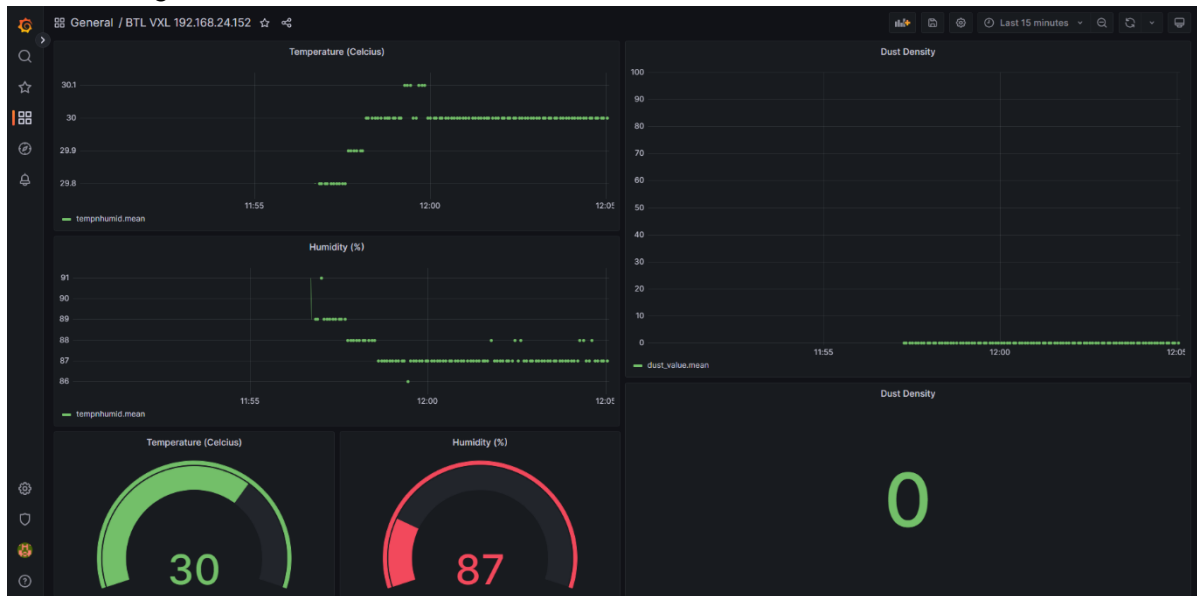
☐ Advanced Query Options

Tip: If no retention policy is specified, **autogen** will be assumed.

4.4. Front-end:

- Grafana là một nền tảng open-source chuyên phục vụ mục đích theo dõi và đánh giá các số liệu thu được. Grafana hỗ trợ rất nhiều loại dashboard và các loại graph để người dùng dễ dàng theo dõi.
- Ưu điểm của Grafana là có thể truy xuất dữ liệu theo dòng thời gian và có khả năng hiển thị trên nhiều loại đồ thị và biểu đồ. Vì vậy nên người dùng có thể đưa ra thông tin và cài đặt cảnh báo dễ dàng trên Grafana.
- Với Grafana, chúng ta có thể tiếp cận với các thao tác dễ dàng:
 - Cài đặt và đăng nhập vào Grafana

- Lựa chọn Datasource (trong bài tập lần này là InfluxDB)
- Tạo Dashboard để hiển thị dữ liệu từ database
- Hình ảnh giao diện của nhóm:



V. Kiểm tra/Thử nghiệm

1. Kiểm tra chức năng của hệ thống:

*Chuẩn bị:

- Đại lượng cần đo:
 - Độ bụi (mg/m³)
 - Nhiệt độ (°C), độ ẩm(%)
- Cách thiết lập hệ thống đo:
 - Kết nối vi xử lý qua máy tính đang sử dụng phần mềm Arduino IDE.
 - Thực hiện nối mạch như trên layout bảng mạch.
 - Phát sóng wifi 2.4GHz
 - Setup server và giao diện người dùng qua Node-RED và Grafana

*Kiểm tra :

- Các bước tiến hành:
 - Nạp code cho các vi xử lý được nêu trên
 - Setup server trên Raspberry Pi
 - Setup Node-RED và giao diện trên Grafana
 - Kiểm tra thông tin đo được của cảm biến qua giao diện Grafana.

Kết quả kiểm tra:



- Nhận xét: Hệ thống kết nối hoạt động đúng theo yêu cầu đề ra, cảm biến đo bụi có thông số vẫn chưa chuẩn, có thể là do cảm biến bị hỏng; thông số cảm biến nhiệt độ và độ ẩm khá đúng, gần với mức nhiệt độ thực tế.

VI. Kết luận

Thông qua việc thực hiện sản phẩm, chúng em đã tích lũy rất nhiều kiến thức thực tế về chuyên ngành Điện tử - Viễn thông, thầy Hàn Huy Dũng đã tạo cho chúng em niềm say mê học tập, tìm tòi kiến thức mới. Thầy còn giúp chúng em hoàn thiện khả năng tư duy phản biện, cách giải quyết vấn đề để đạt được mục tiêu đã đặt ra,... và hơn hết là nắm được quy trình thiết kế một sản phẩm Điện tử - Viễn thông.

Do vốn kiến thức còn hạn hẹp nên việc thực hiện ý tưởng còn nhiều hạn chế.

Nếu còn có gì sai sót, chúng em mong thầy giúp đỡ và tạo điều kiện để chúng em có thể hoàn thành một cách tốt nhất dự án này.

Chúng em xin chân thành cảm ơn!

VII. Bảng công việc

Tên	Phần công việc	Đánh giá mức độ hoàn thành và ý thức làm việc (%)
Trần Duy Anh	Khởi tạo server	90
Phạm Quốc Huy	Code kết nối các node	90
Phạm Tiến Hùng	Báo cáo, slides	75
Nguyễn Bá Trung Hiếu	Báo cáo, slides	60
Đoàn Quang Lưu	Code cảm biến	90

VIII. Tài liệu tham khảo

1. <https://www.arduino.cc/>
2. <https://learnembeddedsystems.co.uk/easy-raspberry-pi-iot-server>
3. https://www.youtube.com/watch?v=DO2wHl6JWQ&ab_channel=LearnEmbeddeSystems
4. <https://html.alldatasheet.com/html-pdf/1440068/ETC/DHT11/60/1/DHT11.html>

5. https://www.sparkfun.com/datasheets/Sensors/gp2y1010au_e.pdf
6. <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/>