**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT MÁY TÍNH**

**Phạm Văn Duy**

**Đặng Đình Khang**

**KHÓA LUẬN TỐT NGHIỆP**

**XÂY DỰNG CÁC THIẾT BỊ ĐIỀU KHIỂN VÀ CẢM BIẾN TRONG NHÀ TRÊN NỀN TẢNG BLUETOOTH SỬ DỤNG NRF52832**

**TP. HỒ CHÍ MINH, 2020**

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT MÁY TÍNH**

**Phạm Văn Duy – 15520169**

**Đặng Đình Khang - 15520336**

**KHÓA LUẬN TỐT NGHIỆP**

**XÂY DỰNG CÁC THIẾT BỊ ĐIỀU KHIỂN VÀ CẢM BIẾN TRONG NHÀ TRÊN NỀN TẢNG BLUETOOTH SỬ DỤNG NRF52832**

**GIẢNG VIÊN HƯỚNG DẪN**

**ThS. PHẠM MINH QUÂN**

**ThS. Nguyễn Duy Xuân Bách**

**TP. HỒ CHÍ MINH, 2020**

|  |  |
| --- | --- |
| ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH  **TRƯỜNG ĐẠI HỌC**  **CÔNG NGHỆ THÔNG TIN** | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc Lập - Tự Do - Hạnh Phúc** |
|  | *TP. HCM, ngày…..tháng…..năm……..* |

# NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP

**(CỦA CÁN BỘ HƯỚNG DẪN)**

**Tên khóa luận:**

**XÂY DỰNG CÁC THIẾT BỊ ĐIỀU KHIỂN VÀ CẢM BIẾN TRONG NHÀ TRÊN NỀN TẢNG BLUETOOTH SỬ DỤNG NRF52832**

**Nhóm sinh viên thực hiện:** **Cán bộ hướng dẫn:**

Phạm Văn Duy 15520169 ThS.Phạm Minh Quân

Đặng Đình Khang 15520336 ThS.Nguyễn Duy Xuân Bách

**Đánh giá khóa luận**

1. Về cuốn báo cáo:

1. Về nội dung nghiên cứu:

1. Về chương trình ứng dụng:

1. Về thái độ làm việc của sinh viên:

Đánh giá chung:

Điểm từng sinh viên:

Phạm Văn Duy: ……/10

Đặng Đình Khang: ……/10

|  |  |
| --- | --- |
|  | **Người nhận xét** |

|  |  |
| --- | --- |
| ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH  **TRƯỜNG ĐẠI HỌC**  **CÔNG NGHỆ THÔNG TIN** | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc Lập - Tự Do - Hạnh Phúc** |
|  | *TP. HCM, ngày…..tháng…..năm……..* |

# NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP

**(CỦA CÁN BỘ PHẢN BIỆN)**

**Tên khóa luận:**

**XÂY DỰNG CÁC THIẾT BỊ ĐIỀU KHIỂN VÀ CẢM BIẾN TRONG NHÀ TRÊN NỀN TẢNG BLUETOOTH SỬ DỤNG NRF52832**

**Nhóm sinh viên thực hiện:** **Cán bộ phản biện:**

Phạm Văn Duy 15520169

Đặng Đình Khang 15520336

**Đánh giá khóa luận**

1. Về cuốn báo cáo:

1. Về nội dung nghiên cứu:

1. Về chương trình ứng dụng:

1. Về thái độ làm việc của sinh viên:

Đánh giá chung:

Điểm từng sinh viên:

Phạm Văn Duy: ……/10

Đặng Đình Khang: ……/10

|  |  |
| --- | --- |
|  | **Người nhận xét** |

# LỜI CẢM ƠN

Để hoàn thành được khóa luận tốt nghiệp này, chúng em xin chân thành gửi lời cảm ơn đến quý thầy cô Trường Đại học Công nghệ Thông tin – Đại học Quốc gia Thành phố Hồ Chí Minh nói chung và quý thầy cô trong Khoa Kỹ thuật Máy tính nói riêng đã truyền đạt cho chúng em kiến thức và những kinh nghiệm quý báu trong suốt chặng đường 5 năm học vừa qua.

Xin gửi lời cảm ơn chân thành đến thầy ThS. Phạm Minh Quân và thầy ThS. Nguyễn Duy Xuân Bách, người đã dành những thời gian quý báu và kinh nghiệm của mình, trực tiếp hướng dẫn tận tình cho chúng em hoàn thành khóa luận tốt nghiệp.

Xin chân thành cảm ơn sự giúp đỡ của các bạn bè, anh chị, những người đã giúp đỡ chúng em tìm kiếm, thu thập thông tin trong suốt quá trình thực hiện khóa luận.

Một lần nữa xin chân thành cảm ơn đến tất cả những người đã giành thời gian, công sức giúp đỡ chúng em hoàn thành khóa luận tốt nghiệp. Trong quá trình thực hiện khóa luận, tất nhiên không tránh khỏi những sai lầm, thiếu sót, chúng em mong quý thầy cô, các bạn, anh chị bỏ qua và thứ lỗi.

Sinh viên thực hiện

**Phạm Văn Duy**

**Đặng Đình Khang**

Khoa Kỹ thuật Máy tính. Lớp KTMT2015

|  |  |
| --- | --- |
| ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH  **TRƯỜNG ĐẠI HỌC**  **CÔNG NGHỆ THÔNG TIN** | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc Lập - Tự Do - Hạnh Phúc** |
|  | *TP. HCM, ngày…..tháng…..năm……..* |

# ĐỀ CƯƠNG CHI TIẾT

**TÊN ĐỀ TÀI:** **XÂY DỰNG CÁC THIẾT BỊ ĐIỀU KHIỂN VÀ CẢM BIẾN TRONG NHÀ TRÊN NỀN TẢNG BLUETOOTH SỬ DỤNG NRF52832**

**Cán bộ hướng dẫn:** ThS. Phạm Minh Quân, ThS. Nguyễn Duy Xuân Bách

**Thời gian thực hiện:** Từ ngày 09/09/2019 đến ngày 07/02/2020

**Sinh viên thực hiện:**

Phạm Văn Duy – 15520169

Đặng Đình Khang– 15520336

**Nội dung đề tài:**

* Mục tiêu của đề tài:

Nghiên cứu, thiết kế các thiết bị điều khiển và cảm biến trong nhà trên nền tảng Bluetooth sử dụng nRF52832 cùng với ESP8266 và các cảm biến nhiệt độ, lửa, gas, module relay. Khai thác được khả năng kết nối tuyệt vời giữa các thiết bị lại với nhau theo cách đặc trưng của mạng Bluetooth mesh: kết nối theo kiểu many-to-many.

Ứng dụng thành công mô hình mạng Bluetooth mesh để điều khiển, giao tiếp giữa các thiết bị tích hợp công nghệ Bluetooth. Có thể giao tiếp với các thiết bị thông qua Smartphone. Thiết kế hệ thống hiện thực, cho ra sản phẩm demo.

* Nội dung chính:
* Tìm hiểu về Bluetooth và mô hình mạng Bluetooth mesh, tham khảo tài liệu về công nghệ cũng như các sản phẩm, hệ thống tương tự trong và ngoài nước.
* Nghiên cứu và thiết kế Gateway sử dụng chip BLE nRF52832, kết hợp với chip tích hợp wifi 2.4Ghz ESP8266.
* Nghiên cứu và thiết kế mạch cảm biến và công tắc tự động có tích hợp chip BLE nRF52832.
* Thiết lập giao thức mạng Bluetooth mesh giao tiếp giữa các thiết bị (Gateway, các mạch cảm biến và mạch công tắc).
* Xây dựng Server có quản lý Database giao tiếp với Gateway, đảm bảo khả năng kết nối, bảo mật đăng nhập và quản lý việc thông báo tình trạng các thiết bị trong nhà cho người dùng.
* Xây dựng ứng dụng Android để cho người dùng có thể thuận tiện điều khiển, theo dõi tình trạng của các thiết bị trong nhà mọi lúc, mọi nơi.
* Hoàn tất, đóng gói, kiểm thử và cải thiện hệ thống.
* Kết quả mong đợi:

Hoàn thành hệ thống bao gồm:

* Gateway (Tích hợp nRF52832 + esp8266).
* Devices:

+ Node 1: Mạch công tắc 1 (4-relays module)

+ Node 2: Mạch công tắc 2 (4-relays module)

+ Node 3: Mạch cảm biến Nhiệt độ (LM75)

+ Node 4: Mạch cảm biến Lửa (Flame sensor)

+ Node 5: Mạch cảm biến Gas (MQ-4)

* Server quản lý việc trao đổi dữ liệu giữa người dùng và hệ thống.
* Ứng dụng Android cho người dùng tương tác với hệ thống một cách trực quan, tiện lợi, có thông báo ngay khi có sự cố.
* Thử nghiệm hệ thống với khoảng cách thực tế và đưa ra đánh giá.

**Kế hoạch thực hiện:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tuần** | **Nội dung** | **Công việc** | **Người thực hiện** |
| **1** (03/09 - 09/09) | Tìm hiểu tổng quan về Bluetooth, mô hình mạng Bluetooth mesh. | Tham khảo tài liệu về công nghệ cũng như các sản phẩm, hệ thống tương tự trong và ngoài nước.  Phân tích tính cấp thiết của công nghệ, khả năng thực thi và đưa ra các giải pháp. | Nguyễn Thành Nam,  Lê Văn Thống |
| **2** (10/09 - 16/09) |
| **3** (17/09 - 23/09) | Tìm hiểu về nRF52832, ESP8266, Xác định cụ thể các loại cảm biến, module sẽ sử dụng: cảm biến nhiệt độ, lửa, gas, module relay 4 kênh. | Nghiên cứu, thiết kế Gateway tích hợp nRF52832 và ESP8266.  Nghiên cứu, thiết kế mạch cảm biến tích hợp nRF52832 với các cảm biến nhiệt độ, lửa, gas và mạch công tắc | Nguyễn Thành Nam,  Lê Văn Thống |
| **4** (24/09 - 30/09) |
| **5** (01/10 - 07/10) |
| **6** (08/10 - 14/10) |
| **7** (15/10 - 21/10) | Thiết lập mô hình mạng Bluetooth mesh để giao tiếp giữa các thiết bị. | Tìm hiểu, thiết lập môi trường, các công cụ và sử dụng bộ SDK do Nordic Semiconductor cung cấp. Ứng dụng bộ SDK của Nordic Semiconductor để hiện thực việc gửi nhận tin nhắn giữa các node, thiết lập mô hình mesh cơ bản với 3 nodes. | Nguyễn Thành Nam |
| **8** (22/10 - 28/10) |
| **9** (29/10 - 04/11) |
| **10** (05/11 - 11/11) | Thực hiện giao tiếp giữa các cảm biến nhiệt độ, lửa, gas và module relay 4 kênh với nRF52832. | Thực hiện giao tiếp giữa cảm biến nhiệt độ, module relay 4 kênh với nRF52832, gửi nhận dữ liệu thông qua mạng Bluetooth mesh. | Nguyễn Thành Nam |
| **11** (12/11 - 18/11) | Thực hiện giao tiếp giữa cảm biến lửa, gas với nRF52832. | Lê Văn Thống |
| **12** (19/11 - 25/11) | Báo cáo tiến độ giữa kỳ (23/11/2018) | | |
| **13** (26/11 - 02/12) | Xây dựng Server. Thực hiện giao tiếp giữa nRF52832 và ESP83266.  Thực hiện giao tiếp giữa ESP8266 với Server. | Xây dựng Server với ngôn ngữ Java, cho phép giao tiếp với ESP8266 qua giao thức WebSocket. | Lê Văn Thống |
| Thiết lập UART và chuẩn hóa dữ liệu trên nRF52832 để truyền nhận dữ liệu với ESP8266. | Nguyễn Thành Nam |
| **14** (03/12 - 09/12) | Xây dựng database cho Server | Thiết kế database lưu trữ dữ liệu bằng MongoDB. Xây dựng API cho Server, cho phép kết nối và truy xuất dữ liệu của Database. | Lê Văn Thống |
| **15** (10/12 - 16/12) | Xây dựng ứng dụng Android giao tiếp với server, quản lý, điểu khiển các thiết bị trong nhà. Thực hiện đóng gói các node thiết bị trong mạng mesh. | Đưa Server và Database hoạt động online. Thiết kế giao diện người dùng trên Android. Thiết kế mô hình 3D cho các node, thực hiện đóng gói các thiết bị. | Nguyễn Thành Nam |
| **16** (17/12 - 23/12) | Xây dựng API cho Android, cho phép giao tiếp với Server bằng websocket, cho phép cập nhật dữ liệu và hiển thị ra ngoài một cách realtime. | Lê Văn Thống |
| **17** (24/12 - 30/12) | Chạy thử nghiệm, kiểm tra và cải thiện hệ thống, tính toán, đo đạc kết quả thực nghiệm. | | Nguyễn Thành Nam, Lê Văn Thống |
| **18** (31/12 - 06/01) | Hoàn chỉnh nội dung báo cáo, quay video demo. | | Nguyễn Thành Nam, Lê Văn Thống |

|  |  |
| --- | --- |
| **Xác nhận của CBHD** | **TP. HCM, ngày….tháng ….năm…...**  **Sinh viên** |

MỤC LỤC

[TÓM TẮT KHÓA LUẬN 1](#_Toc534581510)

[MỞ ĐẦU 2](#_Toc534581511)

[Chương 1. TỔNG QUAN 3](#_Toc534581512)

[1.1. Tình hình trong nước. 3](#_Toc534581513)

[1.2. Tình hình nước ngoài 4](#_Toc534581514)

[1.3. Mục tiêu 5](#_Toc534581515)

[1.4. Đối tượng 6](#_Toc534581516)

[Chương 2. CƠ SỞ LÝ THUYẾT 7](#_Toc534581517)

[2.1. Vai trò mô hình mạng Bluetooth mesh 7](#_Toc534581518)

[2.2. Mô hình mạng Bluetooth mesh 8](#_Toc534581519)

[2.2.1. Tổng quan về mạng mesh 8](#_Toc534581520)

[2.2.2. Khái niệm về mạng Bluetooth mesh 8](#_Toc534581521)

[2.3. Tìm hiểu các cảm biến, thiết bị điều khiển trong nhà. 21](#_Toc534581522)

[2.4. Tìm hiểu về module Bluetooth nRF52832 22](#_Toc534581523)

[2.5. Tìm hiểu về module Wifi ESP8266 25](#_Toc534581524)

[2.6. Các thiết bị sử dụng trong đề tài. 25](#_Toc534581525)

[2.6.1. Module Bluetooth E732-2G4M04S. 25](#_Toc534581526)

[2.6.2. Module WiFi ESP8266 26](#_Toc534581527)

[2.6.3. Cảm biến nhiệt độ LM75. 26](#_Toc534581528)

[2.6.4. Cảm biến phát hiện lửa (flame sensor). 26](#_Toc534581529)

[2.6.5. Cảm biến phát hiện khí Gas (MQ4 sensor) 27](#_Toc534581530)

[2.6.6. Relay 4 kênh 5V 27](#_Toc534581531)

[2.6.7. Còi thạch anh 27](#_Toc534581532)

[2.7. Chuẩn giao tiếp I2C. 27](#_Toc534581533)

[2.8. Chuẩn giao tiếp UART. 29](#_Toc534581534)

[2.9. Chuẩn giao thức WebSocket. 31](#_Toc534581535)

[2.10. Cơ sở dữ liệu với MongoDB. 32](#_Toc534581536)

[2.11. Thư viện RTOS SDK cho ESP8266 37](#_Toc534581537)

[2.12. Thư viện nRF5 SDK và nRF5 SDK for Mesh 39](#_Toc534581538)

[2.12.1. Thư viện nRF5 SDK 39](#_Toc534581539)

[2.12.2. Thư viện nRF5 SDK for Mesh 39](#_Toc534581540)

[Chương 3. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 41](#_Toc534581541)

[3.1. Thiết kế Gateway tích hợp nRF52832 với ESP8266 41](#_Toc534581542)

[3.2. Thiết kế Node cảm biến nhiệt độ 41](#_Toc534581543)

[3.3. Thiết kế Node cảm biến phát hiện lửa 43](#_Toc534581544)

[3.4. Thiết kế Node cảm biến phát hiện khí Gas 44](#_Toc534581545)

[3.5. Thiết kế Node điều khiển Relay 4 kênh 45](#_Toc534581546)

[3.6. Thiết kế mô hình mạng mesh để giao tiếp giữa Gateway và Nodes 46](#_Toc534581547)

[3.7. Thiết kế Cơ sở Dữ liệu với MongoDB 52](#_Toc534581548)

[3.8. Thiết kế Server với Java 58](#_Toc534581549)

[3.9. Thiết kế ứng dụng Android 62](#_Toc534581550)

[Chương 4. KẾT QUẢ 69](#_Toc534581551)

[4.1. Các thiết bị trong hệ thống 69](#_Toc534581552)

[4.2. Đọc thành công các thông số cảm biến và điều khiển các node công tắc 72](#_Toc534581553)

[4.3. Thực nghiệm tổng quan hệ thống 75](#_Toc534581554)

[Chương 5. KẾT LUẬN – KIẾN NGHỊ 80](#_Toc534581555)

[5.1. Kết luận 80](#_Toc534581556)

[5.2. Kiến nghị 80](#_Toc534581557)

DANH MỤC HÌNH VẼ

[Hình 1. Bkav SmartHome 4](#_Toc534581386)

[Hình 2. Shenzhen Minew Technologies Co., Ltd. 5](#_Toc534581387)

[Hình 3. Sự phát triển của cấu trúc liên kết các thiết bị BLE 7](#_Toc534581388)

[Hình 4. Sơ đồ mạng mesh điển hình 9](#_Toc534581389)

[Hình 5. Node và thiết bị unprovisioned 10](#_Toc534581390)

[Hình 6. Các Elements trong một node 11](#_Toc534581391)

[Hình 7. Publish/Subscribe 13](#_Toc534581392)

[Hình 8. Relay node 14](#_Toc534581393)

[Hình 9. Proxy node 15](#_Toc534581394)

[Hình 10. Friend Node và Low Power Node 16](#_Toc534581395)

[Hình 11. Kiến trúc của Bluetooth mesh 16](#_Toc534581396)

[Hình 12. Invitation 19](#_Toc534581397)

[Hình 13. Quá trình trao đổi Public Keys 19](#_Toc534581398)

[Hình 14. Các bước xác thực 20](#_Toc534581399)

[Hình 15. Sơ đồ nguyên lý nRF52832 Nano Development Kit (pin map) 23](#_Toc534581400)

[Hình 16. Sơ đồ nguyên lý nRF52832 Nano Development Kit (Miscellaneous) 24](#_Toc534581401)

[Hình 17. PCB của mạch nRF52832 Nano Development Kit 24](#_Toc534581402)

[Hình 18. Sơ đồ kết nối giao tiếp I2C giữa các thiết bị. 28](#_Toc534581403)

[Hình 19. Sơ đồ hoạt động của giao tiếp I2C. 29](#_Toc534581404)

[Hình 20. Cấu trúc một Frame ví dụ trong giao tiếp UART 30](#_Toc534581405)

[Hình 21. Hình ảnh cơ chế kiếm tra bit even parity 30](#_Toc534581406)

[Hình 22. Yêu cầu Client gửi đến Server để thiết lập WebSocket 31](#_Toc534581407)

[Hình 23. Phản hồi Server gửi về cho Client 32](#_Toc534581408)

[Hình 24. Client gửi một dữ liệu “hello” đến Server 32](#_Toc534581409)

[Hình 25. Biểu đồ số lượng câu hỏi về MySQL(RDBMS) và MongoDB(NoSQL) 34](#_Toc534581410)

[Hình 26. Mongo DB 34](#_Toc534581411)

[Hình 27. Kiến trúc của thư viện nRF5 SDK for Mesh 40](#_Toc534581412)

[Hình 28. Kết nối nRF52832 với ESP8266 41](#_Toc534581413)

[Hình 29. Slave address của LM75 42](#_Toc534581414)

[Hình 30. Pointer register và các giá trị con trỏ tương ứng 42](#_Toc534581415)

[Hình 31. LM75 Temperature register 43](#_Toc534581416)

[Hình 32. Sơ đồ kết nối trong Node cảm biến lửa 44](#_Toc534581417)

[Hình 33. Sơ đồ kết nối trong Node cảm biến khí Gas 45](#_Toc534581418)

[Hình 34. Sơ đồ kết nối nRF52832 với module relay 4 kênh 46](#_Toc534581419)

[Hình 35. Generic OnOff Model 47](#_Toc534581420)

[Hình 36. Cấu trúc message cho các node công tắc 48](#_Toc534581421)

[Hình 37. Sơ đồ Publish/Subscribe cho các node 49](#_Toc534581422)

[Hình 38. Quá trình provisioning diễn ra trên tablet 51](#_Toc534581423)

[Hình 39. Cấu hình publish/subscribe cho node 52](#_Toc534581424)

[Hình 40. Mô hình CSDL trong dự án 53](#_Toc534581425)

[Hình 41. Logo website mlab.com 57](#_Toc534581426)

[Hình 42. Import thư viện WebSocket và MongoDB 59](#_Toc534581427)

[Hình 43. Logo của Amazon Web Services 62](#_Toc534581428)

[Hình 44. Logo “SmartHone” Android Application 62](#_Toc534581429)

[Hình 45. Giao diện LoginActivity 63](#_Toc534581430)

[Hình 46. Giao diện MainActivity tại “Home 1” fragment 63](#_Toc534581431)

[Hình 47. Alert Fire và Alert Gas Activity 64](#_Toc534581432)

[Hình 48. Giao diện profile fragment 64](#_Toc534581433)

[Hình 49. Giao diện HomeControl fragment 65](#_Toc534581434)

[Hình 50. Thông báo “Tài khoản đã được đăng nhập trên thiết bị khác” 66](#_Toc534581435)

[Hình 51. Giao diện cập nhật khi phát hiện rò rỉ gas 67](#_Toc534581436)

[Hình 52. Hai Notification hiển thị báo có lửa, và rò rỉ gas 67](#_Toc534581437)

[Hình 53. Nhấn Button reset để tắt chuông báo từ xa 68](#_Toc534581438)

[Hình 54. Giao diện hiển thị khi nhấn button "Call 114" trên notification 68](#_Toc534581439)

[Hình 55. Các thiết bị trong hệ thống 69](#_Toc534581440)

[Hình 56. Các thiết bị unprovisioned đang sẵn sàng để được provision 70](#_Toc534581441)

[Hình 57. Các thiết bị đã được provision và thực hiện cấu hình hoàn tất 71](#_Toc534581442)

[Hình 58. Kết nối node công tắc với bộ sạc laptop 72](#_Toc534581443)

[Hình 59. Bật/Tắt bộ sạc bằng app Smart Home 73](#_Toc534581444)

[Hình 60. Thử nghiệm node cảm biến lửa 74](#_Toc534581445)

[Hình 61. Thử nghiệm node cảm biến gas 74](#_Toc534581446)

[Hình 62. Vị trí Gateway trong thử nghiệm Relay message với 3 node 75](#_Toc534581447)

[Hình 63. Relay node trong thử nghiệm Relay message với 3 node 76](#_Toc534581448)

[Hình 64. Thực hiện gửi message từ node Fire 76](#_Toc534581449)

[Hình 65. Message được relay về từ node Fire 77](#_Toc534581450)

[Hình 66. Hai Switch trong vai trò làm Relay node và Relay node thay thế 78](#_Toc534581451)

[Hình 67. Vị trí đặt node cảm biến nhiệt độ 78](#_Toc534581452)

[Hình 68. Vị trí đặt node cảm biến Gas 79](#_Toc534581453)

DANH MỤC BẢNG

[Bảng 1. Module Bluetooth E732-2G4M04S 25](#_Toc534581377)

[Bảng 2. Module WiFi ESP8266 v12 26](#_Toc534581378)

[Bảng 3. Cảm biến nhiệt độ LM75 26](#_Toc534581379)

[Bảng 4. Cảm biến phát hiện lửa (Flame sensor) 26](#_Toc534581380)

[Bảng 5. Cảm biến phát hiện Gas (MQ4 sensor) 27](#_Toc534581381)

[Bảng 6. Relay 4 kênh 5V 27](#_Toc534581382)

[Bảng 7. Còi thạch anh 27](#_Toc534581383)

[Bảng 8. So sánh SQL (RDBMAS) với NoSQL 34](#_Toc534581384)

[Bảng 9. So sánh SQL DB schema và MongoDB schema 35](#_Toc534581385)

DANH MỤC TỪ VIẾT TẮT

|  |  |
| --- | --- |
| **Chữ viết tắt** | **Chữ tường minh** |
| IoT | Internet of Things |
| LED | Light Emitting Diode |
| GPIO | General-Purpose Input/Output |
| UART | Universal Asynchronous Receiver / Transmitter |
| ADC | Analog to Digital Converter |
| IC | Inter-Integrated Circuit |
| I2C | Integrated Circuit |
| PWM | Pulse-Width Modulation |
| API | Application Programming Interface |
| PCB | Printed Circuit Board |
| BLE | Bluetooth Low Engerny |
| SIG | Special Interest Group |
| CE | Conformité Européenne |
| CSDL | Cơ sở Dữ liệu |
| GAP | Generic Access Profile |
| GATT | Generic Attribute Profile |
| TTL | Time-to-live |
| LPN | Low Power Node |
| UUID | Universally Unique Identifier |
| PDUs | Protocol Data Units |
| ADV | Advertising |
| OOB | Out-of-Band |
| ROM | Read-Only Memory |
| RAM | Random Access Memory |
| MCU | Main Control Unit |
| SPI | Serial Peripheral Bus |
| DC | Direct Current |
| NFC | Near Field Communications |
| CPU | Central Processing Unit |
| AES | Advanced Encryption Standard |
| RTC | Real Time Control |
| SoC | System on a Chip |
| DMA | Direct Memory Access |
| SCL | Serial Clock |
| SDA | Serial Data |
| TCP | Transmission Control Protocol |
| IP | Internet Protocol |
| HTTP | Hypertext Transfer Protocol |
| RDBMS | Relational Database Management System |
| JSON | JavaScript Object Notation |
| NoSQL | Non-Relational SQL |
| RTOS | Real Time Operation System, |
| SDK | Software Development Kit |
| FIFO | First In - First Out |
| DFU | Device Upgrade Firmware |
| API | Application Programming Interface |
| PC | Personal Computer |
| MSB | Most Significant Bit |
| LSB | Least Significant Bit |
| AC | Alternating Current |
| AWS | Amazon Web Services |

TÓM TẮT KHÓA LUẬN

Internet of Thing (IoT) là một kịch bản của thế giới, khi mà mỗi đồ vật, con người được cung cấp một định danh của riêng mình, và tất cả có khả năng truyền tải, trao đổi thông tin, dữ liệu qua một mạng duy nhất mà không cần đến sự tương tác trực tiếp giữa người với người, hay người với máy tính. IoT đã phát triển từ sự hội tụ của công nghệ không dây, công nghệ vi cơ điện tử và Internet. Nói đơn giản là một tập hợp các thiết bị có khả năng kết nối với nhau, với Internet và với thế giới bên ngoài để thực hiện một công việc nào đó.

Đồ án “Xây dựng các thiết bị điều khiển và cảm biến trong nhà trên nền tảng Bluetooth sử dụng nRF52832” của nhóm là đồ án tập trung tìm hiểu, nghiên cứu và phát triển một hệ thống giúp điều khiển các thiết bị và cảm biến trong nhà trên nền tảng Bluetooth, đồng thời thiết lập, ứng dụng công nghệ mới nhất của Bluetooth là mô hình mạng Bluetooth mesh để giao tiếp giữa các thiết bị tích hợp Bluetooth, kết nối với người dùng thông qua mạng Internet, tạo thành một hệ thống IoT mạnh mẽ.

Hệ thống các thiết bị điều khiển và cảm biến trong nhà bao gồm Gateway, các cảm biến nhiệt độ, cảm biến phát hiện lửa, gas và các mạch công tắc giúp điều khiển bật tắt các thiết bị từ xa. Các thông tin từ cảm biến sẽ được cập nhật liên tục và được gửi tới Gateway. Thông qua mạng Bluetooth mesh, dữ liệu sẽ luôn được cập nhật tới Gateway một cách liên tục dù cho các thiết bị, cảm biến có đặt ở ngoài phạm vi kết nối Bluetooth tới Gateway. Tất cả các dữ liệu về nhiệt độ, báo cháy, báo rò rỉ gas cũng như mục điều khiển các công tắc sẽ được hiển thị trực tiếp trên điện thoại thông minh, đồng thời thông báo trực tiếp trên điện thoại khi có vấn đề về cháy hay rò rỉ gas trong nhà giúp cho người dùng giám sát tình trạng của ngôi nhà mọi lúc, mọi nơi.

MỞ ĐẦU

“Cách mạng công nghiệp 4.0” đang là xu thế của toàn cầu, là một quốc gia đang phát triển, Việt Nam cần phải nắm bắt và phát triển để không bị bỏ lại phía sau. Internet of Things là một “trụ cột chính” trong cuộc cách mạng này.

Bluetooth là chìa khóa mở ra cánh bước vào thời đại Internet of Things, với những tính năng ưu việt vượt trội so với các phương thức khác như hồng ngoại hay wifi. Đặc biệt là khi công nghệ Bluetooth Low Engerny (BLE) ra đời, thiết bị trong hệ thống có thể giao tiếp với nhau với mức tiêu hao năng lượng cực kỳ thấp. Tuy nhiên, Bluetooth là giao tiếp tầm ngắn, khoảng cách vài mét đổ lại, đây thực sự là thách thức để xây dựng một hệ thống quy mô lớn, khi mà đòi hỏi dữ liệu phải được truyền từ thiết bị này đến thiết bị khác với khoảng cách xa và độ tin cậy cao. Bài toán được giải quyết với mô hình mạng Bluetooth mesh, khi mà các thiết bị giao tiếp với nhau theo nguyên tắc “many to many” (m : m) tạo thành một mạng lưới với quy mô lớn, cho phép dữ liệu có thể “nhảy” giữa thiết bị này đến thiết bị khác cho đến khi nào tới được đích mong muốn, nhờ vậy khoảng cách giao tiếp không còn giới hạn.

Từ những vấn đề trên, nhóm quyết định thực hiện đồ án “Xây dựng các thiết bị điều khiển và cảm biến trong nhà trên nền tảng bluetooth sử dụng nRF52832” nhằm hiện thực khả năng kết nối một cách mạnh mẽ và rộng lớn của công nghệ mới ra mắt từ Special Interest Group (SIG) là Bluetooth mesh.

# TỔNG QUAN

## Tình hình trong nước.

Hiện nay, với tốc độ phát triển chóng mặt của công nghệ, việc biến các thiết bị trong nhà thành một hệ thống để có thể quản lý cũng như điều khiển từ xa đã không còn là bất khả thi. Việt Nam cũng không ngoại lệ, là một nước đang phát triển, với chủ trương đón đầu cách mạng 4.0. Vì vậy, nhiều tổ chức trong nước đã nghiên cứu, phát triển ra thị trường những hệ thống nhà thông minh “smart home”, có thể điều khiển các thiết bị trong nhà thông qua kết nối internet. Các cụm từ “smart home” hay “IoT” thường xuyên được nhắc đến, trở thành một xu hướng tiên tiến, hướng con người đến với cuộc sống tiện nghi và thoải mái do công nghệ mang lại. Để có thể nhìn thấy rõ hơn về sức mạnh “IoT” hay “smart home” tại Việt Nam, chúng ta có thể nhìn vào ngành công nghiệp với nhà thông minh đang trở thành “tiêu chuẩn” nhà hiện đại và là xu hướng trong xây dựng. Nhà thông minh cũng trở thành một tiêu chí để cạnh tranh, không khó để thấy các tòa nhà lớn hay các khu căn hộ, khu đô thị đều đang trưng các pano, quảng cáo về “căn hộ 4.0” hay “căn hộ thông minh”. Tỷ lệ tiếp cận internet của Việt Nam hiện nay là 64%[[1]](#footnote-1) dân số, nên nhu cầu tiếp nối công nghệ và học hỏi cái mới rất cao, nhu cầu về chủng loại sản phẩm thông minh trong tương lai 5-10 năm tới sẽ rất là lớn.

Hiện nay nói về đơn vị cung cấp hệ thống thì ngoài một số các nhà cung cấp Việt mới khởi nghiệp trong vài năm gần đây như Lumi hay Acis, thì tiên phong trong lĩnh vực “Smart home” tại Việt Nam là Bkav SmartHome với chục năm có mặt trên thị trường Việt Nam.



Hình 1. Bkav SmartHome

Bkav SmartHome chiếm khoảng 90%[[2]](#footnote-2) thị phần cao cấp, 80%2 thị phần thấp cấp, cũng là công ty phát riển nhà thông minh của Việt Nam có chứng nhận CE.

Giải pháp trong các nhà thông minh tại Việt Nam hiện chưa được nâng cấp, phần lớn chỉ kết nối trong một phạm vi nhỏ hoặc tương đối, muốn thêm bớt các thiết bị sẽ rất khó khăn để triển khai. Với sự ra đời của mạng Bluetooth mesh để khắc phục những tình trạng đó, hiện vẫn còn đang trong giai đoạn nghiên cứu tại Việt Nam và chưa đưa vào triển khai một cách rộng rãi, vì đây có thể xem là một công nghệ mới, cộng đồng lập trình viên vẫn đang phát triển, các tài liệu nghiên cứu bằng tiếng Việt gần như không có. Các công ty, tổ chức công nghệ trong nước cũng đã và đang nghiên cứu, phát triển để áp dụng công nghệ mới này.

## Tình hình nước ngoài

Hiện có rất nhiều dự án IoT về điều khiển các thiết bị trong nhà sử dụng công nghệ Bluetooth của nhiều công ty lớn nhỏ trên thế giới. Có thể kể đến như Minew Technologies (Trung Quốc) là một công ty công nghệ của Trung Quốc, phát triển các thiết bị “IoT”, đã nhiều năm phát triển và luôn đón đầu công nghệ, với các sản phẩm đều được thiết kế từ phần mềm đến phần cứng.



Hình 2. Shenzhen Minew Technologies Co., Ltd.

Các thiết bị công ty sản xuất cho phép giao tiếp qua Bluetooth v5.0, họ sử dụng Beacon để Broadcast dữ liệu về Gateway. Việc sử dụng Beacon thể hiện được sức mạnh của việc giao tiếp giữa các kết nối Bluetooth theo kiểu one-to-many, tuy nhiên với sự xuất hiện công nghệ mạng Bluetooth mesh, ta có thể khai thác được khả năng kết nối tuyệt vời giữa các thiết bị lại với nhau theo kiểu many-to-many. Theo Bluetooth SIG, hệ thống hiện tại có thể bao gồm tối đa 32.000 thiết bị, còn họ đã thử nghiệm thực tế với hệ thống quy mô 1.000 thiết bị. Mạng Bluetooth mesh cho phép dữ liệu có thể “nhảy” giữa thiết bị này đến thiết bị khác cho đến khi nào tới đích, nhờ vậy khoảng cách giao tiếp giữa các thiết bị vốn đã xa hơn với Bluetooth v5.0 lại còn xa hơn rất nhiều. Qua đó thấy được đây là công nghệ hoàn toàn mới, còn đang phát triển. Việc phát triển mạng Bluetooth mesh có thể xem là đón đầu của công nghệ.

## Mục tiêu

Các mục tiêu chính của đề tài:

* Nghiên cứu, xây dựng mô hình mạng Bluetooth mesh cho phép giao tiếp giữa các thiết bị trong nhà.
* Xây dựng hệ thống Server, CSDL và ứng dụng Android để quản lý, giao tiếp với hệ thống mạng Bluetooth mesh đã được thiết lập.
* Hiện thực thành công với hệ thống phòng, chống cháy trong nhà sử dụng mạng Bluetooth mesh. Cho phép người dùng có thể dễ dàng quản lý và điều khiển các thiết bị thông qua điện thoại Android.

## Đối tượng

Đối tượng dự án hướng đến là một mô hình hệ thống cảm biến, công tắc điều khiển từ xa, giao tiếp với nhau thông qua mô hình mạng Bluetooth mesh, kết hợp với hệ thống tracking xác định vị trí người dùng trong tòa nhà. Người dùng có thể dễ dàng kiểm tra và điều khiển qua smart phone.

Đối tượng sử dụng hệ thống là những căn hộ gia đình, công ty hay thậm xí là nhà máy, xí nghiệp lớn. Nhóm phát triển cốt lõi là hệ thống mạng Bluetooth mesh, vì thế, hệ thống mà nhóm phát triển là muốn hiện thực hóa mô hình Bluetooth mesh. Tương lai từ mô hình mạng Bluetooth mesh có thể dễ dàng thực hiện các chức năng khác tùy theo mục đích của người dùng. Do đó, đối tượng sử dụng sẽ cực kỳ rộng và đa dạng.

# CƠ SỞ LÝ THUYẾT

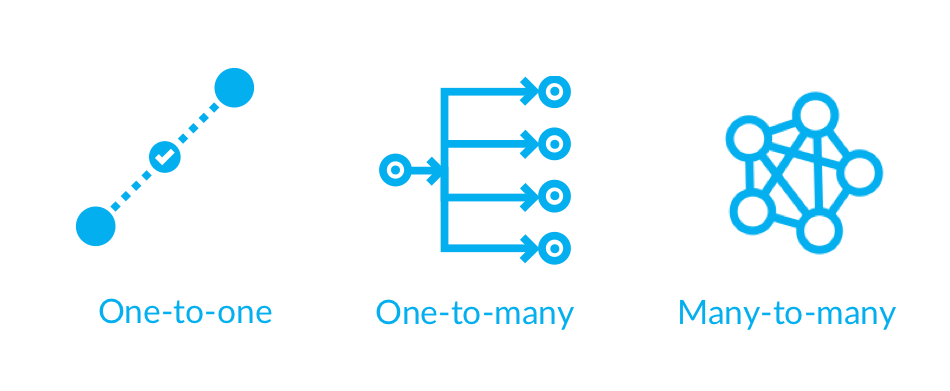
## Vai trò mô hình mạng Bluetooth mesh

Vào năm 2009, tổ chức Bluetooth SIG (Special Interest Group) cho ra đời tiêu chuẩn Bluetooth Low Energy (BLE) để giải quyết sự phát triển nhanh chóng của các dự án trong lĩnh vực Internet of Things (IoT) bao gồm cảm biến, thiết bị đeo, thiết bị y tế..., thiết bị trong hệ thống có thể giao tiếp với nhau với mức tiêu hao năng lượng cực kỳ thấp. Tuy nhiên, BLE thiếu khả năng hỗ trợ cấu trúc liên kết many-to-many (thường được gọi là mesh), trong đó nhiều thiết bị BLE có thể gửi tin nhắn cho nhau và chuyển tiếp tin nhắn đến các thiết bị khác trong mạng, tạo nên thách thức để xây dựng một hệ thống có quy mô lớn.

Các phiên bản trước của Bluetooth hỗ trợ hai cấu trúc liên kết khác nhau:

* Point-to-point (hay one-to-one): thiết lập kết nối chỉ cho 2 thiết bị BLE với nhau.
* Broadcast (hay one-to-many): thể hiện sức mạnh của kết nối một-nhiều giữa các thiết bị BLE, điển hình là Beacon.

Sự ra mắt của mạng Bluetooth mesh vào giữa năm 2017 đã mở ra một kỷ nguyên mới đánh dấu sự phát triển vượt bậc của công nghệ Bluetooth, cho phép triển khai xây dựng hệ thống các thiết bị Bluetooth có quy mô lớn, khi mà đòi hỏi dữ liệu phải được truyền từ thiết bị này đến thiết bị khác với khoảng cách xa và độ tin cậy cao.



Hình 3. Sự phát triển của cấu trúc liên kết các thiết bị BLE

## Mô hình mạng Bluetooth mesh

### Tổng quan về mạng mesh

Thuật ngữ "mesh network" dùng để chỉ một loại cấu trúc liên kết mạng không dây, trong đó mỗi nút mạng (node) tham gia phân phối dữ liệu trong mạng, bằng cách chuyển tiếp dữ liệu đến các node khác trong phạm vi. Có hai kỹ thuật chính được sử dụng để chuyển tiếp dữ liệu trên các mạng lưới, đó là định tuyến và tràn.

Định tuyến: mạng mesh sử dụng kỹ thuật định tuyến, gửi dữ liệu dọc theo một đường dẫn đã chọn, thông qua các node cụ thể để đến đích. Các bộ phận định tuyến có trách nhiệm xác định đường đi cho dữ liệu.

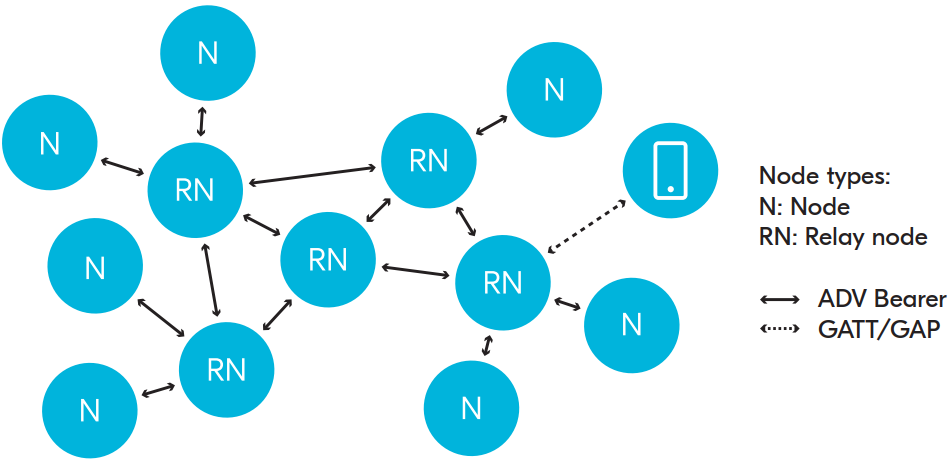
Cơ chế tràn: mạng mesh sử dụng cơ chế tràn, thực hiện phân phối dữ liệu sao cho tất cả các node nhận được trong phạm vi nhất định. Tiếp đó các node chuyển tiếp dữ liệu nhận được bằng cách phát lại dữ liệu để nó được nhận bởi một tập hợp các node trong phạm vi khác, những node được dữ liệu chỉ định hoạt động trên nó sẽ nhận nhiệm vụ xử lý dữ liệu đó. Với cơ chế tràn, tin nhắn sẽ có rất nhiều đường đi để tới đích, nó không phụ thuộc vào các bộ phận định tuyến. Vì vậy, về cơ bản ít rủi ro hơn so với định tuyến khi mà nó có thể gặp lỗi ở các điểm đơn lẻ. Điều này làm cho các mạng lưới sử dụng cơ chế tràn có độ tin cậy cao.

Hai lợi ích chính của mạng mesh:

* Mở rộng phạm vi: Các node có thể chuyển tiếp tin nhắn đến các node ở xa thông qua các node ở giữa chúng, điều này cho phép mở rộng phạm vi của mạng lưới và khoảng cách kết nối của các thiết bị với nhau.
* Khả năng tự phục hồi: Khi có một node bị sự cố, node khác sẽ vẫn tham gia vào việc truyền tải tin nhắn trong mạng lưới, giúp cho việc gửi tin nhắn giữa các node không bị gián đoạn.

### Khái niệm về mạng Bluetooth mesh

Mạng Bluetooth mesh cung cấp khả năng liên kết và giao tiếp giữa các thiết bị theo kiểu many-to-many (m : m) và được tối ưu hóa để tạo ra các mạng thiết bị quy mô lớn. Đó là lý tưởng phù hợp để xây dựng các hệ thống tự động hóa, mạng cảm biến, theo dõi tình hình tài nguyên và các giải pháp IoT khác với yêu cầu hàng chục, hàng trăm hoặc hàng ngàn thiết bị để giao tiếp với nhau.



Hình 4. Sơ đồ mạng mesh điển hình

**Quản lý tràn**

Mạng Bluetooth mesh sử dụng một kỹ thuật chuyển tiếp dữ liệu trên mạng lưới một cách chuyên biệt và được tối ưu hóa cao gọi là quản lý tràn.

**Heartbeats** – Thông điệp kiểu heartbeat được truyền tải bởi các node theo định kỳ, được sử dụng để cho các node khác rằng node hiện đang gửi message này còn hoạt động bình thường trong mạng.

**TTL** – Time-to-live được dùng để giới hạn số bước nhảy mà tin nhắn có thể đi qua nhiều node trong mạng lưới. TTL cho phép các nút thực hiện kiểm soát việc chuyển tiếp message và tiết kiệm năng lượng, bằng cách đảm bảo các message không được chuyển tiếp hơn mức cần thiết. Thông điệp Heartbeat cho phép các nút xác định giá trị TTL tối ưu sẽ là bao nhiêu cho mỗi thông báo được truyền đi.

**Message Cache** – Bộ đệm các message được thiết lập trên tất cả các node. Bộ đệm này chứa tất cả các message gần đây và nếu một message được tìm thấy trong bộ đệm, cho biết nút đã nhìn thấy và xử lý nó trước đó, nó sẽ bị loại bỏ ngay lập tức.

**Friendship** – Đề cập đến mối quan hệ giữa hai loại node là Friend và Low Power Node (LPN)

* LPN phần lớn thời gian radio bị tắt đi, bảo tồn năng lượng và không thể nhận các thông điệp trong mạng mesh dưới mọi lúc.
* Friend node là node lưu trữ các thông báo cho LPN để tiết kiệm năng lượng, do đó LPN có thể ngủ hầu hết thời gian và chỉ thức dậy khi cần thiết. Khi LPN thức dậy, thông qua Friend node, LPN sẽ kiểm tra các thông điệp được lưu trong bộ nhớ và xử lý nó.

**Subnets** - Mạng con là một cơ chế cho phép phân chia mạng lưới Bluetooth.

**Node**

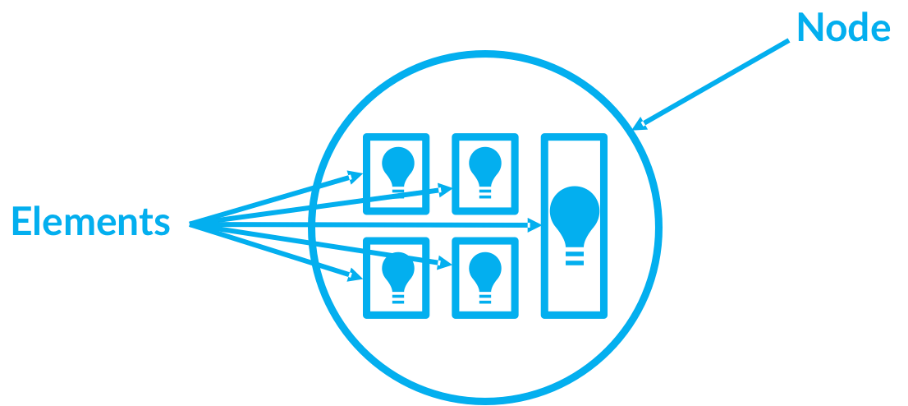
Node (nút) là một thiết bị đã tham gia mạng Bluetooth mesh. Các thiết bị không phải là một phần của mạng mesh được gọi là thiết bị unprovisioned. Khi một thiết bị unprovisoned được tiến hành provision, nó sẽ tham gia vào mạng mesh và trở thành một node.



Hình 5. Node và thiết bị unprovisioned

**Element**

Một element (phần tử) là một thực thể định địa chỉ được trong một thiết bị. Mỗi node phải chứa ít nhất một element và có thể tồn tại nhiều element trong một node và được kiểm soát một cách độc lập, thể hiện trên giá trị trạng thái khác nhau. Ví dụ, một giàn đèn bao gồm các bóng đèn có thể được bật tắt một cách độc lập với nhau.



Hình 6. Các Elements trong một node

**State**

Các element có thể ở trong các tình trạng khác nhau, được thể hiện thông qua state (trạng thái). Cụ thể như bật và tắt là trạng thái của bóng đèn trong giàn đèn.

**Message**

Trong mạng Bluetooth mesh, việc giao tiếp giữa các node được tổ chức theo “message-oriented”, các node thực hiện việc gửi các message để kiểm soát, điều khiển hoặc chuyển tiếp thông tin cho nhau.

Có 2 loại message chính thường được sử dụng đó là acknowledged và unacknowledged message.

Acknowledged message đòi hỏi phải có message trả về. Việc gửi trả message về có 2 mục đích: xác nhận rằng message đã được gửi đến đích và gửi trả dữ liệu cần thiết từ node nhận đến node đã gửi và đó thường là các message chứa trạng thái (status message).

Unacknowledged message là loại message mà k cần phải có message trả về.

**Address**

Message trong mạng Bluetooth mesh phải được gửi cũng như nhận từ một address (địa chỉ) nhất định.

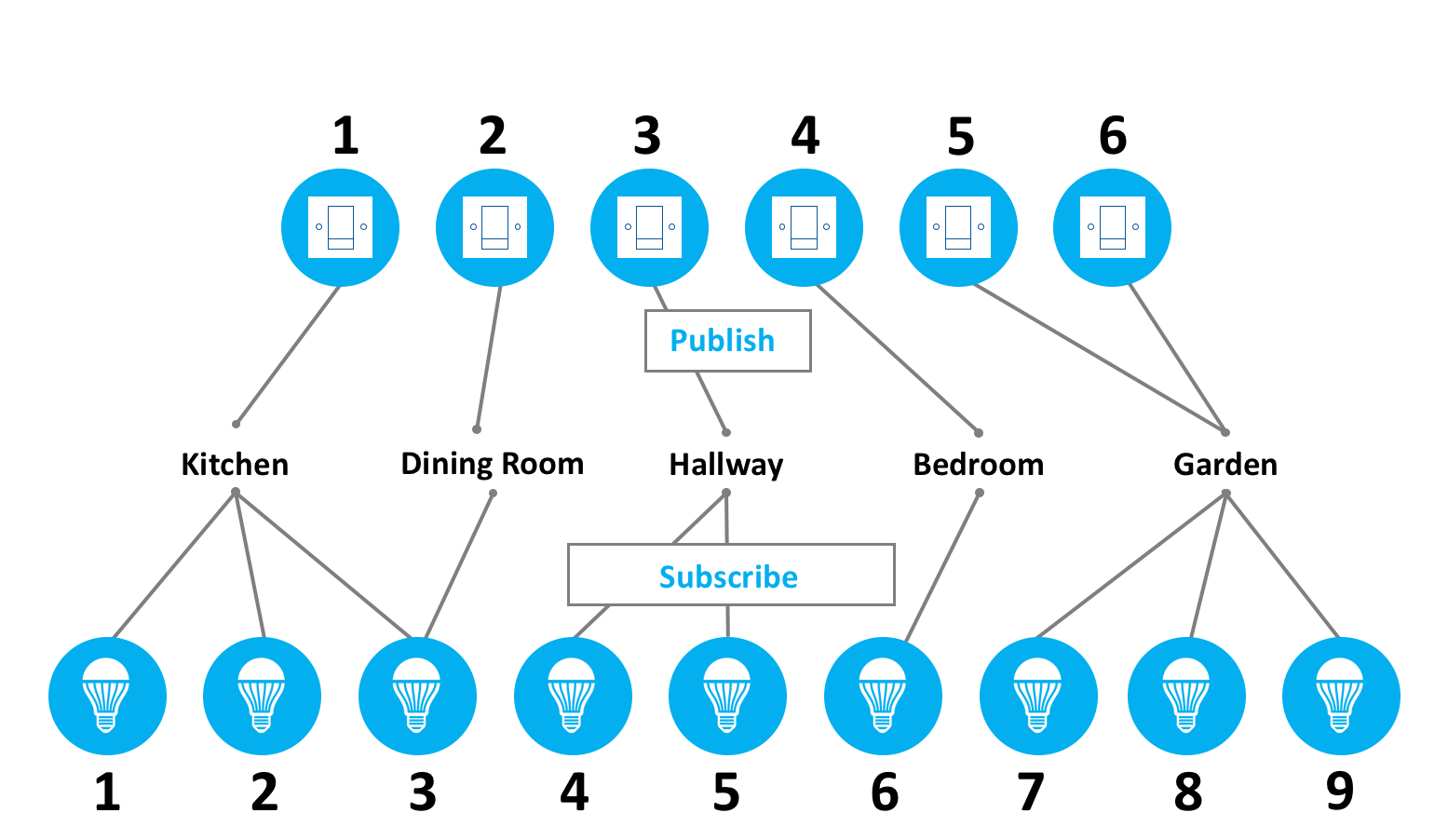
Có 3 loại địa chỉ:

* Unicast Addresses: là địa chỉ được gán cho mỗi thiết bị trong quá trình provisioning và đảm bảo rằng không có sự trùng lặp nào được phân bổ trong mạng.
* Group Addresses: là một địa chỉ multicast đại diện cho một hoặc nhiều element. Một group address (địa chỉ nhóm) được định nghĩa bởi Bluetooth SIG, được biết đến với tên gọi là SIG Fixed Group Address. Ngoài ra địa chỉ nhóm cũng có thể được định nghĩa bởi người dùng thông qua việc cấu hình bằng ứng dụng, gọi là Dynamic Group Address. Có thể có tới 16383 địa chỉ nhóm trong mạng mesh[[3]](#footnote-3).
* Virtual Addresses: có thể được gán cho một hoặc nhiều element, trải rộng ra một hoặc nhiều node. Nó có định dạng theo kiểu 128-bit UUID và có thể xem nó giống như là một nhãn cho element tương ứng. Địa chỉ ảo có thể được cấu hình sẵn tại thời điểm sản xuất. Có thể có tới 70 nghìn tỷ địa chỉ ảo trong một mạng lưới2.

**Publish/Subscribe**

Message được gửi và nhận trong mạng mesh tuân thủ theo mô hình giao tiếp đặc trưng là publish/subscribe.

Hành động gửi message đến một hoặc nhiều node khác được gọi là Publishing và sử dụng Publish Address làm địa chỉ. Việc cấu hình một node để có thể nhận được một số tin nhắn cần thiết nhất định gọi là Subscribing, thông thường subcribe tới group address hoặc virtual address.



Hình 7. Publish/Subscribe

Một node có thể subcribe nhiều address khác nhau cũng như các node có thể publish cùng chung một address. Trên hình, đèn thứ 3 vừa có thể subscribe tới switch thứ nhất ở phòng bếp, vừa có thể subscribe tới switch thứ 2 trong phòng ăn. Do vậy, người dùng có thể điều khiển bóng đèn thứ 3 này từ cả hai nơi. Tương tự, công tắc thứ 5 và thứ 6 có thể publish cùng chung một địa chỉ, do vậy, hai switch này cùng thực hiện việc điều khiển các bóng đèn 7, 8, 9 trong khu vườn.

**Model**

Model là một thuật ngữ quan trọng trong mạng Bluetooth mesh. Mỗi model xác định một vài hoặc tất cả các tính năng của element.

Server model: là tập hợp các states, state transitions, state bindings, và messages mà một element có chứa model có thể gửi hoặc nhận.

Client model: không xác định bất kỳ state nào, thay vào đó, nó chỉ định nghĩa các message như các tin nhắn GET, SET và STATUS để gửi đến một server model.

Control model: chứa cả server model và client model cho phép giao tiếp với các server model và client model khác.

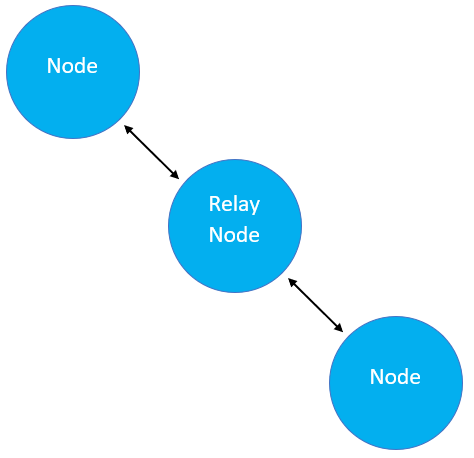
Một model có thể được hình thành bằng cách kế thừa từ một model khác. Model không cần kế thừa được gọi là root model. Việc thay đổi một model bằng cách thêm hoặc xóa một behavior là không được phép và chỉ nên kế thừa từ một model sẵn có để tùy chỉnh lại sao cho phù hợp với nhu cầu[[4]](#footnote-4).

**Các loại node trong mạng Bluetooth mesh**

Tất cả các node trong mạng mesh đều có thể truyền tải cũng như nhận các message, tuy nhiên, các node trọng mạng còn có thể hỗ trợ thêm một số tính năng chuyên biệt. Cụ thể có 4 loại tính năng là: Relay, Proxy, Friend, và Low Power. Mỗi node trong mạng có thể không hỗ trợ hoặc hỗ trợ tất cả các tính năng này.

* **Relay node**

Để thực hiện việc gửi nhận message giữa các node nằm ngoài phạm vi giao tiếp của nhau, ta cần phải có tính năng relay. Các relay node trong mạng Bluetooth mesh hỗ trợ tính năng relay, tức là có khả năng tái truyền tải message đã được broadcast của node khác.

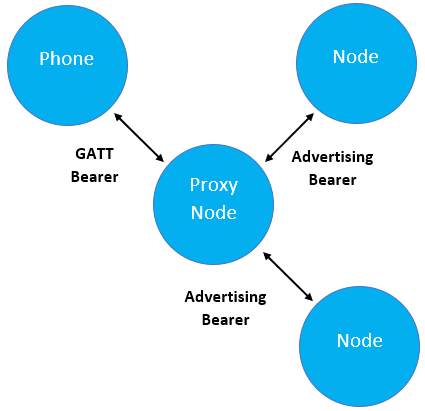


Hình 8. Relay node

Chính điều này giúp cho mạng mesh có khả năng mở rộng với phạm vi lớn, giúp cho message có thể truyền tải đến toàn bộ mạng lưới, vượt qua phạm vi ban đầu rất nhiều với khả năng luân chuyển message nhảy đến 127 bước[[5]](#footnote-5).

* **Proxy node**

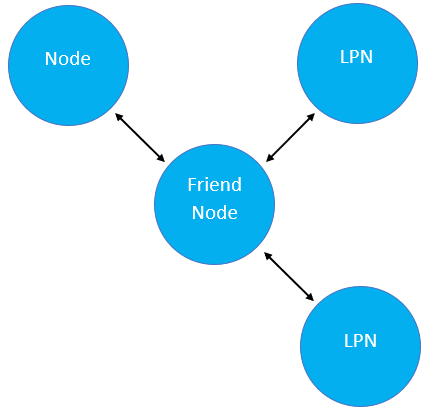
Với sự phát triển của công nghệ hiện nay, hầu hết các smartphone hoặc tablet đều hỗ trợ công nghệ BLE[[6]](#footnote-6). Để hiện thực hóa việc giao tiếp giữa mạng mesh với các thiết bị chưa hỗ trợ việc kết nối Bluetooth mesh (non-mesh-supported) này, ta cần phải có proxy node trong mạng mesh. Proxy node cung cấp khả năng truyền nhận mesh message giữa GATT(Generic Attributes) và Advertising bearers3.



Hình 9. Proxy node

* **Friend Node và Low Power Node**

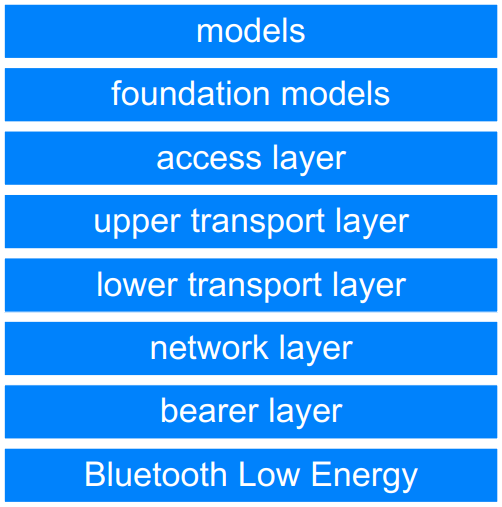
Low Power Node (LPN) thường có nguồn cung cấp năng lượng hạn chế như pin, vì vậy chúng cần tiết kiệm năng lượng bằng cách tắt radio thường xuyên nhất có thể. Các Friend Node có khả năng giúp một LPN hoạt động một cách hiệu quả, bằng cách lưu trữ các message dành cho node đó và chỉ gửi chúng khi LPN yêu cầu.



Hình 10. Friend Node và Low Power Node

**Kiến trúc hệ thống của Bluetooth mesh**

Bluetooth mesh được xây dựng trên nền tảng của BLE, sử dụng advertising và scanning state của BLE để hiện thực việc kết nối giữa các node trong mạng.



Hình 11. Kiến trúc của Bluetooth mesh

* **Bearer Layer**

Bearer layer xác định cách xử lý các mesh packet (Protocol Data Units – PDUs). Hiện có 2 loại bearers:

* Advertising Bearer: tận dụng các tính năng advertising và scanning của BLE để truyền tải và nhận các mesh PDUs.
* GATT Bearer: cho phép thiết bị không hỗ trợ Advertising Bearer giao tiếp một cách gián tiếp với các node của mạng mesh, sử dụng proxy protocol và điều này được thực hiện thông qua các proxy node.
* **Network Layer**

Network Layer định nghĩa các loại địa chỉ của message một cách đa dạng, định nghĩa một định dạng cho network message với khả năng cho phép transport layer PDUs sẽ được chuyển tiếp bởi Bearer Layer.

* **Lower Transport Layer**

Xử lý hai nhiệm vụ chính:

* Phân đoạn các packet từ layer trên (Upper Transport Layer)
* Tập hợp lại các packet từ layer bên dưới (Bearer Layer)
* **Upper Transport Layer**

Chịu trách nhiệm cho các chức năng sau:

* Mã hóa (Encryption)
* Giải mã (Decryption)
* Xác thực (Authentication)
* Kiểm soát việc luân chuyển thông điệp (Transport control message)
* **Access Layer**

Chịu trách nhiệm cho việc xác định làm thế nào để các ứng dụng có thể sử dụng được Upper Transport Layer. Access Layer xử lý các nhiệm vụ sau:

* Định dạng dữ liệu ứng dụng (Application data format)
* Mã hóa và giải mã (Encryption and decryption)
* Xác minh dữ liệu (Data verification)
* **Foundation Models Layer**

Chịu trách nhiệm triển khai các model liên quan đến việc cấu hình và quản lý mạng mesh.

* **Models Layer**

Quản lý việc triển khai các model, bao gồm việc thực hiện các behavior, message, state…

**Provisioning Process**

“Provisioning” là quá trình các thiết bị được tiến hành thêm vào mạng Bluetooth mesh. Sau khi được thực hiện quá trình provisioning, thiết bị đó sẽ được gọi là node. Thiết bị được sử dụng để thực hiện quá trình provisioning được gọi là provisioner và thường là smartphone, tablet hoặc PC. Provisioner chịu trách nhiệm cho việc tạo và phân phối các NetKeys.

Quá trình provisioning bao gồm 5 bước:

**Bước 1. Beaconing**

Một unprovisioned device, tức là một thiết bị chưa được thực hiện quá trình provisioning sẽ cho biết rằng nó sẵn sàng để được provision bằng cách advertising như một Unprovisioned Device beacon.

**Bước 2. Invitation**

Trong bước này, provisioner sử dụng bearer sẵn có để gửi invitation đến thiết bị sẽ được provision, dưới dạng Provisioning Invite PDU.



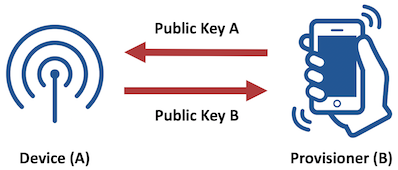
Hình 12. Invitation

Thiết bị Beaconing phản hồi với thông tin về chính nó trong Provisioning Capabilities PDU, cụ thể là:

* Số lượng các element mà thiết bị hỗ trợ.
* Bộ thuật toán bảo mật (security algorithm) được hỗ trợ.
* Tính khả dụng của các public key sử dụng công nghệ Out-of-Band (OOB).
* Khả năng của thiết bị này để tạo ra một giá trị cho người dùng.
* Khả năng của thiết bị này cho phép một giá trị được nhập vào từ người dùng.

**Bước 3. Exchanging Public Keys**

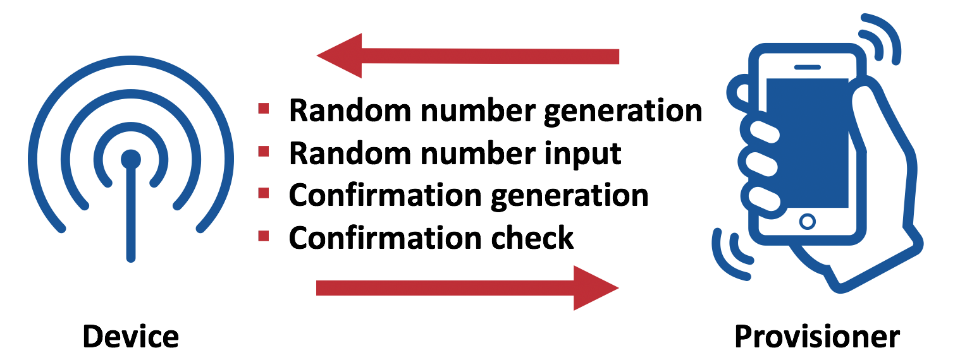
Provisioner và thiết bị sẽ được provision thực hiện việc trao đổi các public keys một cách trực tiếp hoặc sử dụng công nghệ Out-of-Band (OOB).



Hình 13. Quá trình trao đổi Public Keys

**Bước 4. Authentication**

Bước xác thực này thường yêu cầu một hành động của người dùng bằng cách tương tác với cả provisioner và thiết bị chưa được provision. Phương thức xác thực phụ thuộc vào khả năng của cả hai thiết bị được sử dụng.



Hình 14. Các bước xác thực

Trường hợp đầu tiên là output OOB, thiết bị chưa được xác định có thể xuất ngẫu nhiên một số hoặc một số có nhiều chữ số cho người dùng ở một số dạng, chẳng hạn như nhấp nháy đèn LED nhiều lần. Số đó sau đó được nhập vào thiết bị thực hiện quá trình provisioning thông qua một số phương thức nhập. Các trường hợp khác bao gồm một input OOB (trong đó con số sẽ được cung cấp bởi provisioner và được nhập vào thiết bị chưa được provision), static OOB hoặc hoàn toàn không có OOB.

Bất kể phương thức xác thực được sử dụng, xác thực cũng bao gồm bước tạo giá trị xác nhận và bước kiểm tra giá trị xác nhận.

**Bước 5. Distribution of the Provisioning Data**

Sau khi xác thực thành công, session key sẽ được tạo và được sử dụng để bảo đảm việc phân phối dữ liệu tiếp theo cần thiết để hoàn tất quy trình cung cấp, bao gồm NetKey, device key, unicast address gán cho thiết bị được provision bởi provisioner. Sau khi hoàn tất bước này, thiết bị sẽ hoàn thành quá trình provisioning và trở thành một node trong mạng mesh.

## Tìm hiểu các cảm biến, thiết bị điều khiển trong nhà.

Với mục tiêu hiện thực hóa mô hình mạng Bluetooth mesh trong thực tế, đồng thời nhận thấy cháy nổ đang là vấn đề gây nhức nhối trong xã hội khi nó mang lại hậu quả cực kỳ to lớn về người và của. Các biện pháp an toàn phòng, chống cháy nổ là hết sức cần thiết. Một hệ thống giúp phát hiện và cảnh báo khi có nguy cơ hoặc cháy xảy ra là rất hữu ích và thiết thực. Nhóm sẽ ứng dụng mô hình mạng Bluetooth mesh để xây dựng hệ thống phát hiện cảnh báo cháy trong nhà.

* Theo dõi, giám sát nhiệt độ trong nhà:

Việc theo dõi, giám sát nhiệt độ trong nhà với độ chính xác cao là cực kỳ quan trọng và cấp thiết, giúp chủ nhà có thể nhận biết được sự thay đổi bất thường, đột ngột của nhiệt độ mà có các biện pháp phòng cháy kịp thời. Nhóm sử dụng cảm biến LM75 với độ chính xác cao và tốc độ truyền dữ liệu nhanh để việc phát hiện dấu hiệu cháy được nhanh chóng.

* Phát hiện dấu hiệu của lửa:

Khi sự cháy xảy ra, điều tất yếu là phải có lửa và lửa có thể được phát hiện thông qua cảm biến lửa (flame sensor). Mọi vật có nhiệt độ lơn hơn 0oK đều tự phát ra tia hồng ngoại nhưng ở những bước sóng khác nhau, ví dụ như những remote điều khiển trong nhà có bước sóng hồng ngoại từ 0.75 đến 1.4 µm và đối với ngọn lửa bình thường sẽ có bước sóng từ 760 cho đến 1100 nm. Module cảm biến lửa sẽ dùng một con Led thu tín hiệu hồng ngoại để bắt tín hiệu hồng ngoại mà ngọn lửa đó phát ra, một dấu hiệu rõ ràng của sự cháy. Tín hiệu sẽ kích hoạt chuông báo cháy kêu đồng thời gửi thông báo cháy tới chủ nhân để kịp thời xử lý.

* Phát hiện dấu hiệu của khí Gas:

Rò rỉ khí Gas là một nguyên nhân gây cháy nổ mà chúng ta có thể nhận biết được và phòng chống từ xa. Với việc sử dụng cảm biến phát hiện khí Gas MQ4 giúp phát hiện khí Gas trong môi trường, cảm biến có độ nhạy cao, có thể điều chỉnh bằng biến trở và tốc độ truyền nhanh. Khi phát hiện khí Gas, cảm biến sẽ gửi tín hiệu bật chuông cảnh báo và gửi thông báo phát hiện rò rỉ khí Gas tới chủ nhân để kịp thời xử lý.

## Tìm hiểu về module Bluetooth nRF52832

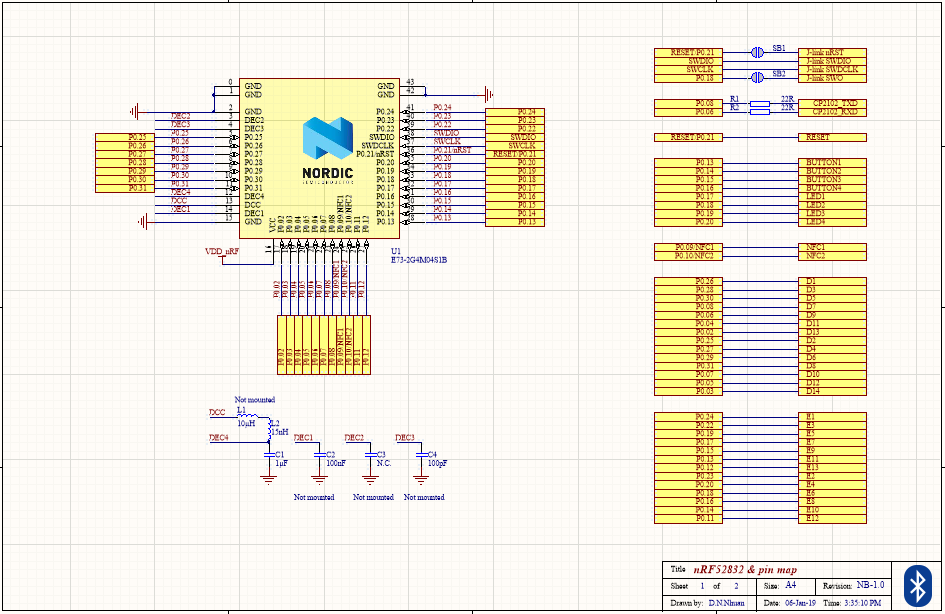
nRF52832 là dòng chip BLE được cung cấp từ Nordic, là một thành viên của dòng SoC nRF52 Series. nRF52832 sử dụng ARM Cortex-M4 hiệu suất cao hơn so với Cortex-M0 sử dụng trong dòng nRF51822. Ngoài ra nRF52832 còn tăng dung lượng ROM/RAM lên đến 512KB ROM và 64KB RAM để hỗ trợ các ứng dụng phức tạp đòi hỏi bộ nhớ lớn. Đặc biệt nRF52832 còn được tích hợp phần cứng NFC để hỗ trợ các ứng dụng nhận dữ liệu NFC và truyền lên smartphone thông qua BLE.

Giống như nRF51822, nRF52832 còn tích hợp nhiều ngoại vi analog và digital có thể hoạt động độc lập với MCU, hỗ trợ thay đổi chức năng của từng pin để giảm độ phức tạp khi thiết kế PCB. Đặc biệt, nRF52832 tích hợp chức năng EasyDMA cho các peripheral truyền/nhận dữ liệu như SPI, UART, I2C, PWM, v.v… giảm yêu cầu hoạt động của core MCU và từ đó giảm dòng tiêu thụ đáng kể so với nRF51822. Về nguồn cung cấp, nRF52832 hỗ trợ tích hợp DC-DC converter để tạo các nguồn cung cấp giá trị khác nhau từ 1 nguồn cung cấp duy nhất bên ngoài giúp giảm thiểu yêu cầu cho các chip nguồn phải sử dụng trên board sản phẩm.

Tính năng kỹ thuật nổi bật:

* Giải pháp Single-chip có thể xử lý đồng thời ứng dụng và giao thức BLE và NFC
* 32-bit ARM Cortex M4F CPU core
* 512KB Flash + 64kB RAM
* 3 tốc độ truyền dữ liệu (2Mbps/1Mbps/250kbps)
* Công suất phát đến +4dBm
* Hiệu suất nhận dữ liệu cao -93dBm
* Hỗ trợ thay đổi chức năng của 32 I/O pin
* Tích hợp 2 bộ NFC-A với khả năng wakeup-on-field và touch-to-pair
* 5 bộ 16/24-bit timer
* Hỗ trợ mã hóa AES tích hợp easyDMA
* 4 bộ PWM tích hợp easyDMA
* 8 kênh 8/9/10-bit ADC
* Tích hợp mạch so sánh
* Giao tiếp SPI, UART tích hợp easyDMA
* I2S tích hợp easyDMA
* 3 bộ RTC
* Nguồn cung cấp từ 1.8V đến 3.6V

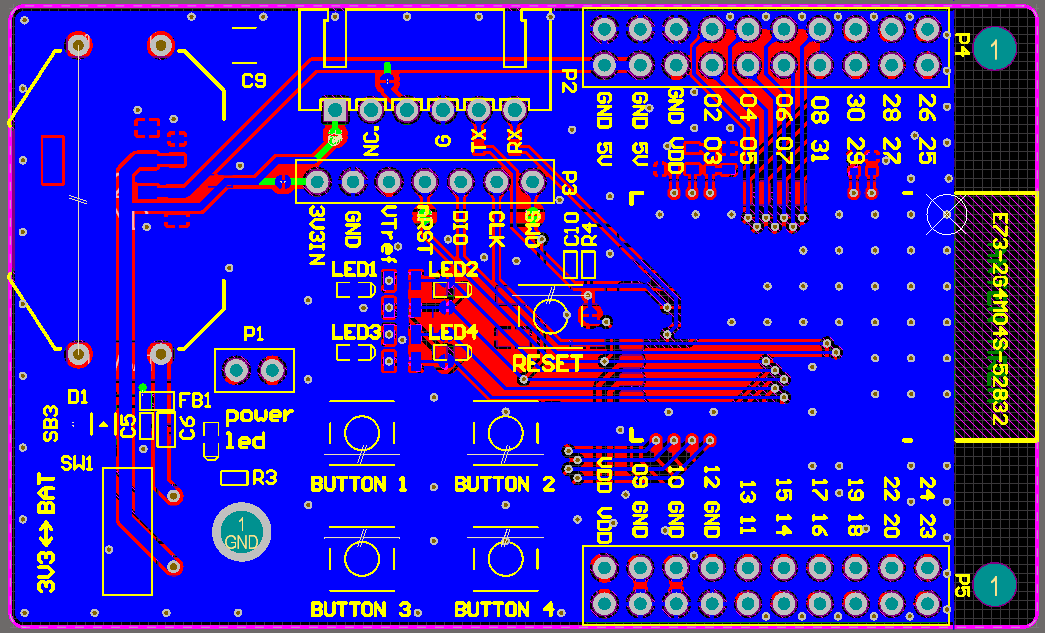
Nhóm sử dụng một Nano Development Kit tích hợp module Bluetooth E732-2G4M04S (RDIC: nRF52832) được thiết kế dựa trên nRF52832 Development Kit do Nordic cung cấp. Việc lập trình cho Kit được thực hiện thông qua J-Link và Embedded Segger Studio.



Hình 15. Sơ đồ nguyên lý nRF52832 Nano Development Kit (pin map)



Hình 16. Sơ đồ nguyên lý nRF52832 Nano Development Kit (Miscellaneous)



Hình 17. PCB của mạch nRF52832 Nano Development Kit

## Tìm hiểu về module Wifi ESP8266

ESP8266 là dòng chip tích hợp Wi-Fi 2.4Ghz có thể lập trình, được sản xuất bởi một công ty bán dẫn Trung Quốc có tên Espressif Systems.

Được phát hành đầu tiên vào tháng 8 năm 2014, đóng gói đưa ra thị trường dạng Mô dun ESP-01, được sản xuất bởi bên thứ 3: AI-Thinker. Có khả năng kết nối Internet qua mạng Wi-Fi một cách nhanh chóng và sử dụng rất ít linh kiện đi kèm. Với giá cả có thể nói là rất rẻ so với tính năng và khả năng ESP8266 có thể làm được.

ESP8266 có một cộng đồng các nhà phát triển trên thế giới rất lớn, cung cấp nhiều module lập trình mã nguồn mở giúp nhiều người mới có thể tiếp cận và xây dựng ứng dụng rất nhanh.

Cấu trúc phần cứng của chip ESP8266:

* Sử dụng 32-bit MCU core có tên là Tensilica
* Tốc độ System clock có thể set ở 80MHz hoặc 160MHz
* Tích hợp 50KB RAM để lưu dữ liệu ứng dụng khi chạy
* Có đầy đủ các ngoại vi chuẩn để giao tiếp như 17 GPIO, một slave SDIO, 3 SPI, một I2C, một I2S, 2 UART, 2 PWM.
* Tích hợp các mạch RF để truyền nhận dữ liệu ở tần số 2.4GHz
* Hỗ trợ các hoạt động truyền nhận các IP packages ở mức hardware như Acknowledgement, Frafmentation & Defragmentation, Aggregation…

## Các thiết bị sử dụng trong đề tài.

### Module Bluetooth E732-2G4M04S.

|  |  |
| --- | --- |
|  | Module E73-2G4M04S là một module Bluetooth được thiết kế bởi Chengdu Ebyte. Hỗ trợ chuẩn BLE 4.2 và BLE 5.0 (RDIC: nRF52832).  - Tần số: 2379 – 2496 MHz  - Điện áp hoạt động: 1.8 – 3.6 V  - Hỗ trợ đầy đủ GPIO và giai tiếp: I2C, UART, SPI. |

Bảng 1. Module Bluetooth E732-2G4M04S

### Module WiFi ESP8266

|  |  |
| --- | --- |
| Hình ảnh có liên quan | Module ESP8266 v12 truyền nhận WiFi dựa trên chip ESP8266 SoC của Espressif. Người dùng có thể lập trình lại firmware cho ESP8266 bằng các thư viện có sẵn.  - Nguồn: 3.3VDC.  - Giao tiếp: UART. |

Bảng 2. Module WiFi ESP8266 v12

### Cảm biến nhiệt độ LM75.

|  |  |
| --- | --- |
|  | LM75 là cảm biến nhiệt độ có độ chính xác cao, đạt tới độ chính xác 0.125oC. Dãy tín hiệu nhiệt độ từ -55oC đến +125oC.  - Nguồn: 2.8V – 5.5V  - Giao tiếp: I2C |

Bảng 3. Cảm biến nhiệt độ LM75

### Cảm biến phát hiện lửa (flame sensor).

|  |  |
| --- | --- |
|  | Là cảm biến phát hiện lửa trong khoảng cách 80cm, phát hiện ngọn lửa có bước sóng từ 760nm – 1100nm, có thể điều chỉnh độ nhạy bằng biến trở.  - Nguồn: 3.3 - 5VDC, 15mA.  - Điện áp ra: 3.3 – VDC, có cả Analog và Digital.  - Góc quét: 60 độ |

Bảng 4. Cảm biến phát hiện lửa (Flame sensor)

### Cảm biến phát hiện khí Gas (MQ4 sensor)

|  |  |
| --- | --- |
|  | Cảm biến Gas MQ4 dùng để phát hiện khí Gas trong môi trường, độ nhạy có thể điều chỉnh được bằng biến trở. MQ4 có thể phát hiện khí gas, metan.  - Nguồn hoạt động: 5V  - Loại dữ liệu ra: Analog, Digital |

Bảng 5. Cảm biến phát hiện Gas (MQ4 sensor)

### Relay 4 kênh 5V

|  |  |
| --- | --- |
|  | Module Relay 4 kênh 5V gồm 4 rơ le điện áp hoạt động ở mức 5VDC, đầu ra điều khiển hiệu điện thế tối đa ở mức 250VAC-10A, 125VSC-10A đối với dòng điện xoay chiều AC và 30VDC-10A, 28VDC-10A với dòng điện một chiều DC. |

Bảng 6. Relay 4 kênh 5V

### Còi thạch anh

|  |  |
| --- | --- |
|  | Còi thạch anh SFM-27 DC3-24V  - Nguồn: 3 - 24VDC  - Hiệu suất âm thanh: ≥ 85dB  - Tần số cộng hưởng: 3000 ± 500Hz |

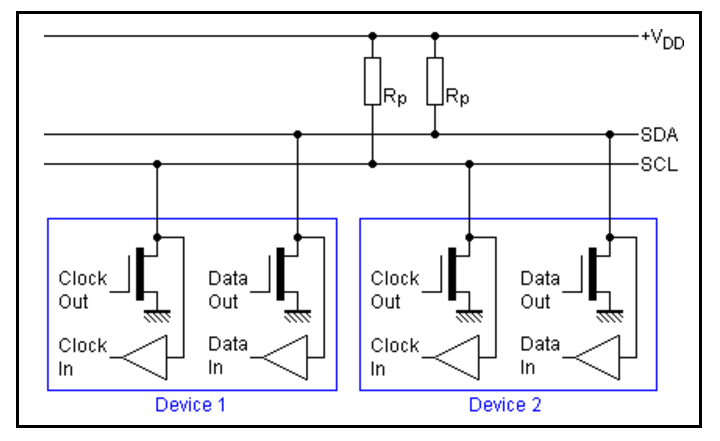
Bảng 7. Còi thạch anh

## Chuẩn giao tiếp I2C.

I2C, viết tắt của cụm từ “Inter – Integrated Circuit”, là loại bus nối tiếp được phát triển bởi hãng sản xuất linh kiện điện tử Philips. Ban đầu, loại bus này chỉ được dùng trong các linh kiện điện tử của Philips. Sau đó do tính ưu việt và đơn giản của nó, I2C đã được chuẩn hóa và được dùng rộng rãi trong các module truyền thông nối tiếp của vi mạch tích hợp ngày nay.

I2C sử dụng hai đường truyền tín hiệu:

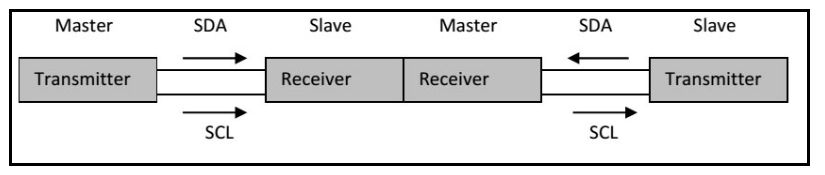
* Một đường xung clock (SCL) chỉ do Master phát đi (thông thường ở 100kHz và 400kHz. Mức cao nhất là 1MHz đến 3.4MHz).
* Một đường dữ liệu (SDA) theo hai hướng truyền và nhận dữ liệu.
* Sơ đồ kết nối như hình dưới.



Hình 18. Sơ đồ kết nối giao tiếp I2C giữa các thiết bị.

Mỗi dây SDA và SCL đều được nối với điện áp dương của nguồn cấp thông qua một điện trở kéo lên (pullup resistor). Sự cần thiết của điện trở kéo lên là vì chân giao tiếp I2C của thiết bị ngoại vi thường là dạng cực máng hở (opendrain hay opencollector). Giá trị của điện trở này khác nhau tùy vào từng thiết bị và chuẩn giao tiếp, thường dao động khoảng 1k đến 4.7k.

Một thiết bị hay IC khi kết nối với bus I2C, ngoài một địa chỉ (duy nhất) để phân biệt, nó còn được cấu hình là thiết bị chủ hay tớ. Sự phân biệt này là do trên một bus I2C thì quyền điều khiển thuộc về thiết bị chủ. Thiết bị chủ nắm vài trò tạo xung clock cho toàn hệ thống, khi giữa hai thiết bị chủ - tớ giao tiếp thì thiết bị chủ có nhiệm vụ tạo xung đồng hồ và quản lý địa chỉ của thiết bị tớ trong suốt quá trình giao tiếp. Thiết bị chủ giữ vai trò chủ động, còn thiết bị tớ giữ vai trò bị động trong việc giao tiếp.



Hình 19. Sơ đồ hoạt động của giao tiếp I2C.

Các chế độ hoạt động:

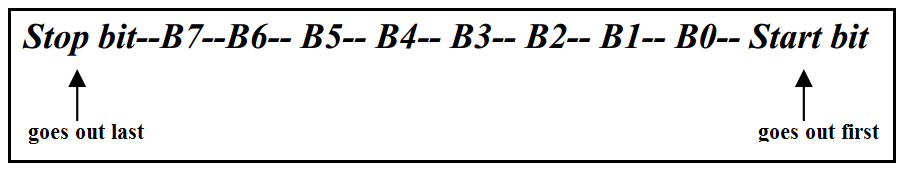
* Chế độ tiêu chuẩn (Standard mode)
* Là chế độ tiêu chuẩn ban đầu được phát hành vào đầu những năm 80.
* Nó có tốc độ dữ liệu tối đa 100kbps
* Sử dụng 7-bit địa chỉ, và 112 địa chỉ tớ.
* Chế độ nhanh (Fast mode)
* Tốc độ dữ liệu tối đa tăng lên đến 400kbps
* Để ngăn chặn gai tiếng ồn, ngõ vào thiết bị fast mode là schimitt – triggered.
* Chân SCL và SDA của thiết bị tớ I2C ở trạng thái trở kháng cao khi không cấp nguồn.
* Chế độ cao tốc (High-speed mode)
* Chế độ này đã được tạo ra chủ yếu để tăng tốc độ dữ liệu lên đến 36 lần nhanh hơn so với chế độ tiêu chuẩn. Nó cung cấp 1.7Mbps.

## Chuẩn giao tiếp UART.

UART (viết tắt của Universal Asynchronous Receiver/Transmitter) là phương thức truyền nhận tín hiệu bất đồng bộ, nghĩa là bên nhận và bên phát không cần phải có chung tín hiệu xung clock. Khi đó bên truyền muốn truyền dữ liệu sẽ gửi Start bit (bit 0) để báo bên nhận biết bắt đầu nhận dữ liệu và khi truyền xong dữ liệu thì stop bit (bit 1) sẽ được gửi để báo cho bên thu biết kết thúc quá trình truyền.

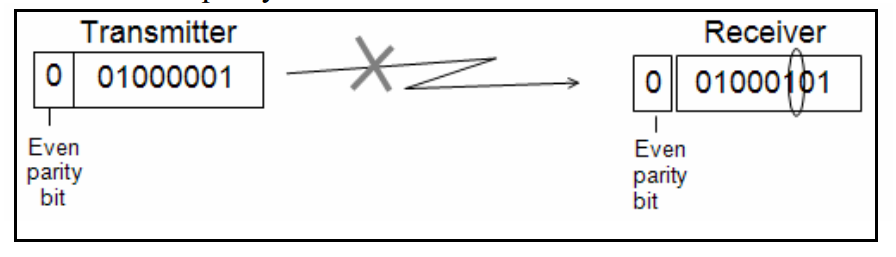
Tốc độ truyền (Baudrate): đơn vị bit per second (bps) là số bit truyền được trong 1s. Ở truyền nhận không đồng bộ thì các bên truyền nhận phải thống nhất Baudrate với nhau. Các thông số tốc độ Baudrate thường hay được sử dụng để giao tiếp là: 600, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 56000, 57600, 115200.

Frame: Dữ liệu được truyền và nhận phải có cấu hình giống nhau về khung truyền. Gồm 1 start bit (bit 0), data và 1 stop bit (bit 1), bít kiểm tra tính chẵn lẻ (Parity).



Hình 20. Cấu trúc một Frame ví dụ trong giao tiếp UART

Parity bit (bit chẵn lẻ): Dùng để kiểm tra dữ liệu truyền có đúng hay không. Có hai loại Parity đó là even parity (bit chẵn) và odd parity (bit lẻ). even parity nghĩa là số bit 1 trong data truyền cùng với bit parity luôn là là số chẵn, ngược lại nếu odd parity nghĩa là số bit 1 trng data truyền cùng với bit parity luôn là số lẻ. Bit parity không phải là bit bắt buộc trong frame vì thế chúng ta có thể loại bỏ bit này ra khỏi frame.



Hình 21. Hình ảnh cơ chế kiếm tra bit even parity

## Chuẩn giao thức WebSocket.

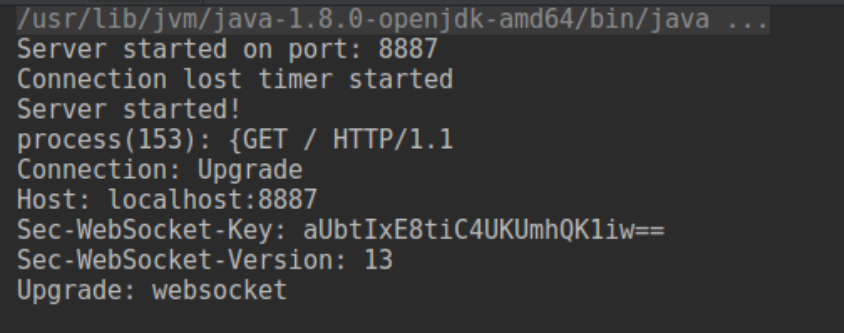
WebSocket là một giao thức cho phép truyền nhận dữ liệu hai chiều giữa Client và Server qua một kết nối TCP duy nhất. Nghĩa là khi có một Client khác kết nối đến Server, một kết nối TCP mới sẽ được tạo ra và Client này và Server có thể gửi nhận dữ liệu cho nhau trên kết nối TCP này. Có bao nhiêu Client kết nối đến Server là có bấy nhiêu kết nối TCP được tạo ra. WebSocket có thể hoạt động trên các cổng web tiêu chuẩn, nên không có rắc rối về việc mở cổng cho các ứng dụng hay lo lắng về việc bị chặn bởi các tường lửa hay Proxy Server.

Khác với giao thức HTTP là cần Client chủ động gửi yêu cầu cho Server thì Server mới gửi phản hồi về lại cho Client. Nghĩa là Server sẽ bị động chờ yêu cầu từ Server (Đối với HTTP 1.0 và 1.1). Giao thức WebSocket cho phép Server có thể chủ động gửi thông tin đến Client mà không cần có yêu cầu từ Client.

Tất cả dữ liệu giao tiếp giữa Client và Server được gửi trực tiếp qua duy nhất một kết nối TCP cố định nên thông tin được gửi đi một cách nhanh chóng và liên tục khi cần thiết. WebSocket làm giảm độ trễ bởi vì khi kết nối WebSocket được thiết lập, Server không cần phải chờ đợi yêu cầu từ Client, tương tự Client có thể gửi thông tin đến Server bất cứ lúc nào. Để có thể sử dụng được WebSocket thì hông phải chỉ cần công cụ trình duyệt hay ứng dụng hỗ trợ mà còn phải có Server WebSocket. Server Websocket được tạo ra bằng bất kì ngôn ngữ Server-side nào.

Cách thức hoạt động: Giao thức có hai bước

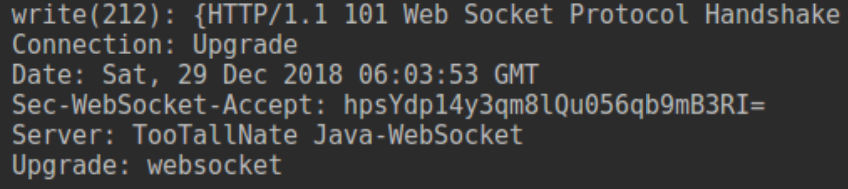
* Bước bắt tay: Phía Client sẽ gửi một yêu cầu nâng cấp đến Server



Hình 22. Yêu cầu Client gửi đến Server để thiết lập WebSocket

Để xác nhận việc kết nối, Client sẽ gửi một giá trị Sec-WebSocket-Key mã hóa bằng Based64 đến Server.

Sau đó Server sẽ nối chuỗi cố định *“258EAFA5-E914-47DA-95CA-C5AB0DC85B11″* vào Sec-WebSocket-Key để được chuỗi mới, sau đó mã hóa bằng Base64 và gửi phản hồi về lại Client với Status code là 101 kèm với giá trị Sec-WebSocket-Accept chính là chuỗi kết quả vừa tạo được.



Hình 23. Phản hồi Server gửi về cho Client

Client sẽ kiểm tra status code (phải bằng với 101) và giá trị Sec-WebSocket-Accpet xem có đúng kết quả mong đợi không. Nếu đúng, kết nối được thiết lập thành công.

* Bước truyền dữ liệu: sau khi kết nối TCP được khởi tạo thành công, Client và Server có thể gửi nhận dữ liệu cho nhau trên kết nối TCP đó, dữ liệu ở đây chính là các Frame.



Hình 24. Client gửi một dữ liệu “hello” đến Server

## Cơ sở dữ liệu với MongoDB.

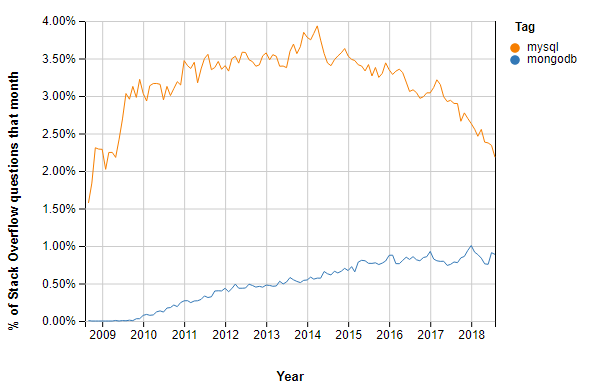
**Giới thiệu về NoSQL:**

NoSQL viết tắt bởi Non-Relational SQL (Không quan hệ), hay có nơi thường gọi là Not Only SQL. NoSQL là một khái niệm chỉ về những hệ cơ sở dữ liệu không sử dụng mô hình quan hệ RDBMS (Relational Database Management System), được phát triển trên Javascript Framework với kiểu dữ liệu JSON (theo cấu trúc “key: value”). NoSQL ra đời dựa trên những khuyết điểm, hạn chế của mô hình dữ liệu quan hệ RDBMS về tốc độ, tính năng, khả năng mở rộng, memory cache, …

Đặc điểm của hệ CSDL mới này bao gồm: schema-free, hỗ trợ mở rộng dễ dàng, API đơn giản, nhất quán cuối (eventual consistency), không giới hạn không gian dữ liệu, ...

|  |  |  |
| --- | --- | --- |
|  | **SQL (RDBMS)** | **NoSQL** |
| Model | - Ràng buộc.  - Lưu trữ dữ liệu dưới dạng các table.  - Có thể mở rộng theo chiều dọc. | - Không ràng buộc.  - Lưu trữ dữ liệu dưới dạng JSON (cấu trúc “key: value”).  - Có thể mở rộng theo chiều ngang. |
| Data | - Chuẩn hóa (Normalization), giảm thiểu sự thừa dữ liệu nhưng giảm tốc độ query.  - Cung cấp ngôn ngữ Join query rất mạnh mẽ, có thể lấy dữ liệu liên quan trong nhiều table bằng cách sử dụng một câu lệnh SQL. | - Denormalization (Không chuẩn hóa), giúp tăng tốc độ query nhanh hơn.  - Không trang bị ngôn ngữ để join query, chúng ta chỉ có thể lấy tất cả dữ liệu liên quan về và liên kết chúng bằng code trong chương trình. |
| Schema | Schema phải được design và implements trước đó, tuy cũng có thể update sau nhưng những thay đổi lớn sẽ rất phức tạp. | Sử dụng schema động cho dữ liệu phi cấu trúc, duex liệu được add mọi nơi và bất cứ lúc nào. |
| Performance | Phụ thuộc vào design. Design tốt hệ thống SQL hoạt động tốt, nhanh và ngược lại. | Tốc độ sẽ nhanh hơn RDBMS nhờ denormalization. |

Bảng 8. So sánh SQL (RDBMAS) với NoSQL



Hình 25. Biểu đồ số lượng câu hỏi về MySQL(RDBMS) và MongoDB(NoSQL)

**Tổng quan về MongoDB:**



Hình 26. Mongo DB

MongoDB là một trong những hệ cơ sở dữ liệu NoSQL phổ biến nhất. MongoDB được viết bằng ngôn ngữ C++ và hỗ trợ đa nền tảng, có thể chạy trên Window, Linux và Mac, … nó hỗ trợ hầu hết các ngôn ngữ lập trình phổ biến như C#, Java, PHP, Javascript, … và các môi trường phát triển khác nhau. MongoDB là cơ sở dữ liệu hướng tài liệu, MongoDB lưu trữ dữ liệu trong các document dưới dạng JSON với schema động rất linh hoạt. Nghĩa là bạn có thể lưu các bản ghi mà không cần lo lắng về cấu trúc dữ liệu như số trường, kiểu của trường đó. Tài liệu MongoDB tương tự các đối tượng trong JSON. Cũng chính do schema linh hoạt nên MongoDB có thể được dùng để lưu trữ các dữ liệu có cấu trúc phức tạp, đa dạng và không cố định (Big Data).

|  |  |
| --- | --- |
| SQL Database | MongoDB |
| Table | Collection |
| Row | Document |
| Column | Field |
| Joins | Embedded documents, linking |
| Primary key | Primary key (mặc định là \_id do chính MongoDB tạo) |

Bảng 9. So sánh SQL DB schema và MongoDB schema

Các thành phần trong một MongoDB:

* \_id: được hiểu là primary key (khóa chính) trong mỗi document, là trường bắt buộc phải. \_id là giá trị duy nhất trong document MongoDB. Giá trị \_id sẽ tự động được tạo ra khi ta thêm mới một document và nó là duy nhất trong cơ sở dữ liệu MongoDB. \_id sẽ là một số thập lục phân 12-byte với 4 bytes đầu là Timestamp hiện tại, 3 bytes tiếp là cho ID của thiết bị, 2 bytes tiếp nữa là process ID của MongoDB Server và 3 bytes cuối là giá trị gia tăng. Vì thế, \_id đảm bảo là duy nhất cho mỗi document được tạo ra.
* Collection: chứa các document, tương ứng như table trong cơ sở dữ liệu SQL, nhưng các collection không cần phải định nghĩa các cột, cách hàng hay kiểu dữ liệu trước như trong table của SQL.
* Document: mang ý nghĩa tương ứng với Row trong SQL, là một bản ghi thuộc một collection. Có cấu trúc theo dạng “key: value” giống trong JSON.
* Field: là một cặp “key: value” thuộc trong một document.

Các thuật ngữ hay dừng trong MongoDB:

* Database: Giống với trong SQL nói riêng RDBMS nói chung, Database chứa các collection của MongoDB, tương tự các table trong SQL. Mỗi Database có một tập tin riêng được lưu trữ trên bộ nhớ vật lý. Một MongoDB Server có thể lưu trữ nhiều Database.
* Cursor: là con trỏ đến tập kết quả khi truy vấn dữ liệu. Developer sử dụng để truy vấn và láy kết quả.
* Index (chỉ mục): Là chỉ số lưu trữ giá trị của một field cụ thể hoặc thiết lặp các fields, sắp xếp theo thứ tự các field này. Index hỗ trợ trong các truy vấn. Nếu không có index, MongoDB vẫn có thể truy vấn bằng cách quét tất cả các document của collection để chọn ra những document phù hợp với câu truy vấn, do vậy MongoDB sẽ phải xử lý một khối lượng lớn dữ liệu.

Dưới đây là ví dụ về một collection với hai document và các field trong MongoDB.

{

"\_id": {

"$oid": "5c19061cfb6fc04dd6ef4e14"

},

"name": "user",

"password": "user123",

"home": [{…}]

},

{

"\_id": {

"$oid": "5c19061cfb6fc04dd6ef4e15"

},

"name": "admin",

"password": "admin123",

"home": [{…}]

}

## Thư viện RTOS SDK cho ESP8266

Đây là một bản SDK sử dụng hệ điều hành thời gian thực (Real Time Operation System, viết tắt là RTOS) được port từ FreeRTOS (là phiên bản RTOS miễn phí được port cho nhiều MCU khác nhau) cho ESP8266. RTOS SDK cho ESP8266 do Espressif Systems phát triển và open source.

RTOS là một hệ điều hành được tạo ra nhằm thực hiện những công việc trong thời gian thực, xử lý dữ liệu mà không có sự chậm trễ của bộ đệm. RTOS có thể cho phép nhiều chương trình chạy cùng lúc (multi-tasking) nên thời gian phản hồi cực kỳ nhanh, độ chậm trễ rất nhỏ (tính bằng ms), có thể quản lý tài nguyên về phần cứng và cung cấp các dịch vụ cho các chương trình khác. RTOS cung cấp cơ chế phân chia các yêu cầu bộ nhớ và ngoại vi của MCU.

**Kernel**: có chức năng quản lý những task được tạo ra, quà đảm bảm chúng chạy cùng một lúc. Thật ra tại một thời điểm nhất định chỉ có một Task duy nhất được xử lý, bằng cách lưu lại trạng thái Task đang chạy, sau đó CPU sẽ load tiếp trạng thái của Task tiếp theo, sau đó xử lý Task tiếp theo đó, khi xử lý xong sẽ load lại trạng thái Task ban đầu và xử lý tiếp. Vì thời gian xử lý mỗi Task rất nhanh (trong khoảng thời gian ms) và CPU thực hiện các tác vụ rất nhanh nên dưới gốc nhìn của người dùng thì các Task như đang thực hiện cùng lúc một cách liên tục.

**Task**: là một chương trình chạy liên tục trong vòng lặp vô hạn và không bao giờ dừng lại. Một chương trình sẽ có nhiều Task khác nhau và được Kernel quản lý, sắp xếp trình tự xử lý thông qua cơ chế Scheduler bằng cách lưu trạng thái Task hiện tại và load trạng thái của Task tiếp theo. Một Task trong RTOS thường có các trạng thái như sau:

* RUNNING: Đang thực thi
* READY: Sẵn sàng để thực hiện
* WAITING/BLOCKED: chờ sự kiện
* INACTIVE: Không được kích hoạt

**Scheduler**: Là cơ chế lập lịch của Kernel, quyết định Task nào được quyền dành CPU để thực thi tiếp theo. Một số luật khi scheduling như sau:

* Cooperative: Một Task được thực thi khi và chi khi Task đang chạy dừng lại.
* Round-robin: Mỗi Task chỉ được thực thi trong một khoảng thời gian định trước gọi là Time slice và không có độ ưu tiên
* Priority base: Task nào có quyền ưu tiên cao nhất sẽ được thực thi trước tiên, nếu các Task có cùng độ ưu tiên thì sẽ theo cơ chế round-robin.
* Priority-based pre-emptive: Task nào có quyền ưu tiên thấp hơn phải nhường quyền thực thi cho Task có quyền ưu tiền cao hơn.

**Đồng bộ, chia sẻ tài nguyên giữa các Tasks**: Mỗi Task thực thi một tác vụ riêng của mình, chúng có không gian bộ nhớ riêng và hoàn toàn độc lập với nhau. Nhưng cần phải có một cơ chế để các Task có thể kết nối, đồng bộ để phối hợp trong một chương trình lớn và chia sẻ tài nguyên cho nhau để tránh xảy ra xung đột. Vì lý do đó một số khái niệm được sinh ra:

* Signal Events: Có chức năng đồng bộ các Task
* Message Queue: Có chức năng trao đổi tin nhắn giữa các Task trong hoạt động theo cơ chế Fist In-Fist Out (FIFO).
* Mail Queue: Có chức năng trao đổi dữ liệu giữa các Task sử dụng hàng đợi của khối bộ nhớ.
* Semaphores: Có chức năng truy xuất tài nguyên liên tục từ các Task khác nhau.
* Mutext: Có chức năng đồng bộ hóa truy xuất tài nguyên sử dụng Mutual Exclution.

## Thư viện nRF5 SDK và nRF5 SDK for Mesh

### Thư viện nRF5 SDK

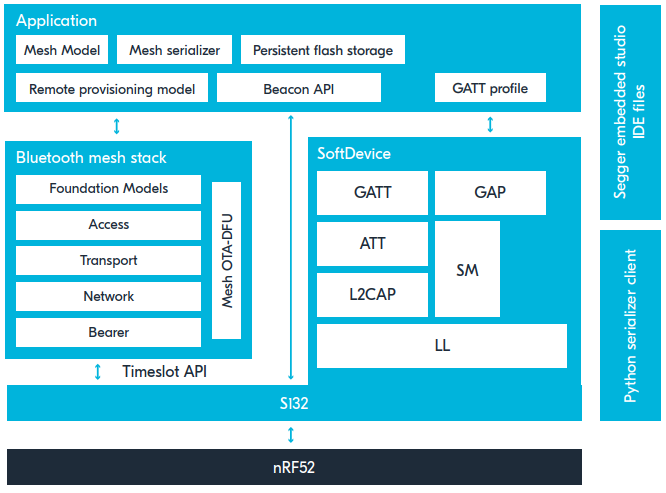
Đây là một bộ SDK rất quan trọng trong việc hỗ trợ xây dựng các ứng dụng một cách đầy đủ, đáng tin cậy và an toàn với việc hỗ trợ các dòng chip nRF52 và nRF51. Nó cung cấp cho các nhà phát triển rất nhiều module và ví dụ khác nhau bao gồm nhiều ví dụ về Bluetooth Low Energy, các Device Upgrade Firmware (DFU), GATT serializer và hỗ trợ các driver cho tất cả các thiết bị ngoại vi trên tất cả các thiết bị thuộc nRF5 Series.

### Thư viện nRF5 SDK for Mesh

Nordic Semiconductor cung cấp bộ nRF5 SDK for Mesh kết hợp với nRF52 Series để cung cấp giải pháp giúp các nhà phát triển triển khai các hệ thống mạng Bluetooth mesh. Mục tiêu của SDK là cung cấp cho các nhà phát triển một bộ API để giúp triển khai dễ dàng hơn và trừu tượng hóa mọi phức tạp không cần thiết trong cấu trúc mạng Bluetooth mesh.

Thư viện nRF5 SDK for Mesh cần phải kết hợp với phiên bản nhất định của thư viện nRF5 SDK để có thể triển khai lập trình và hiện đang được hỗ trợ cho việc hiện thực các dự án bằng cách sử dụng Cross-platform IDE chuyên biệt là Segger Embedded Studio. Ngoài ra bộ SDK còn cung cấp các ví dụ, hỗ trợ xây dựng các models một cách chuyên biệt.

Kiến trúc của thư viện nRF5 SDK for Mesh:



Hình 27. Kiến trúc của thư viện nRF5 SDK for Mesh

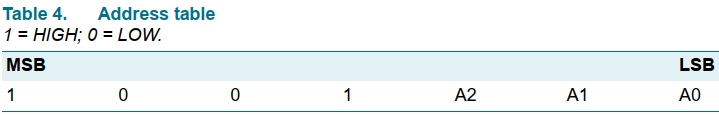
Bluetooth mesh stack sử dụng time-slot API để cho phép hoạt động đồng thời với các dịch vụ GATT/GAP. Nhờ vậy, việc biến smartphone, tablet, PC – các thiết bị hỗ trợ BLE trờ thành một phần của mạng Bluetooth mesh một cách dễ dàng.

# PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## Thiết kế Node cảm biến nhiệt độ

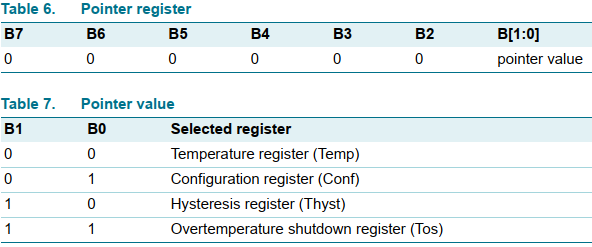
LM75B là cảm biến nhiệt độ do NXP Semiconductor sản xuất. LM75B thực hiện chuyển đổi trực tiếp nhiệt độ đo được thành giá trị lưu trong các thanh ghi của chip và có thể đọc bằng giao tiếp I2C. LM75B có thể hoạt động với điện áp nguồn từ 2.8V đến 5.5V, đo được dải nhiệt độ từ -55 °C đến +125 °C, sai số kết quả đo là 2 °C với dải nhiệt độ từ -25 °C đến +100 °C.

LM75 có ba bit LSB định nghĩa địa chỉ Slave trong thanh ghi Address là A2, A1, A0, cho phép tám cảm biến khác nhau được kết nối đồng thời trên cùng một bus mà không có xung đột địa chỉ và được thể hiện bằng ba chân A2, A1, A0 trên chip. Để định nghĩa địa chỉ Slave cho LM75B, ta sẽ kết nối các chân này với VCC hoặc GND để set mức logic 1 hoặc 0. Bốn bit MSB còn lại được định nghĩa sẵn, có giá trị là ‘1001’, bằng cách nối dây cứng bên trong LM75B. Thực hiện nối A2, A1, A0 cho GND, ta sẽ có được địa chỉ Slave là **0x48.**



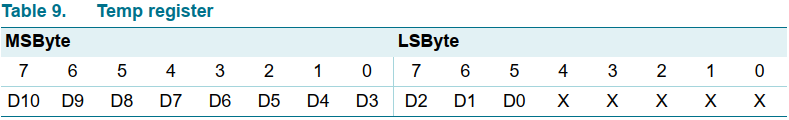
Hình 29. Slave address của LM75B

Thanh ghi Pointer của LM75B sẽ xác định lượt đọc ghi từ I2C sẽ thực hiện trên thanh ghi nào của LM75B. 6 Bit đầu của thanh ghi này đã được định nghĩa sẵn là 0. Để chọn thanh ghi để thực hiện đọc ghi, ta cần xác định giá trị 2 Bit cuối.



Hình 30. Thanh ghi Pointer và giá trị để xác định thanh ghi sẽ thao tác

Dữ liệu nhiệt độ trên LM75 được lưu trong thanh ghi nhiệt độ. Thanh ghi này chỉ cho phép đọc và bao gồm 2 bytes, 1-byte cho MSB và 1-byte cho LSB. Tuy nhiên, chỉ 11 bits của 2 bytes này được sử dụng để chứa dữ liệu nhiệt độ.



Hình 31. Thanh ghi Temp trong LM75B

Cách tính toán nhiệt độ dựa trên 11-bit dữ liệu của thanh ghi nhiệt độ:

* + Khi D10 = 0, nhiệt độ là một số dương, giá trị của nhiệt độ **(oC) = +(Temp data) x 0.125 oC.**
  + Khi D10 = 1, nhiệt độ là một số âm, giá trị của nhiệt độ **(oC) = -(bù 2 của Temp data) x 0.125 oC.**

Ví dụ ta có 11-bit dữ liệu là **000 1101 1000­2 =** 21610 x 0.125 = 27 oC.

Để đọc được giá trị nhiệt độ trong thanh ghi Temp, ta cần truyền dữ liệu là giá trị của thanh ghi Pointer với Bit R\W là write trong lần truyền I2C đầu tiên để xác định thanh ghi ta sẽ đọc là thanh ghi Temp. Trong lần truyền I2C tiếp theo ta sẽ truyền đi với Bit R\W là read và LM75B sẽ trả về 2 byte dữ liệu nhiệt độ..



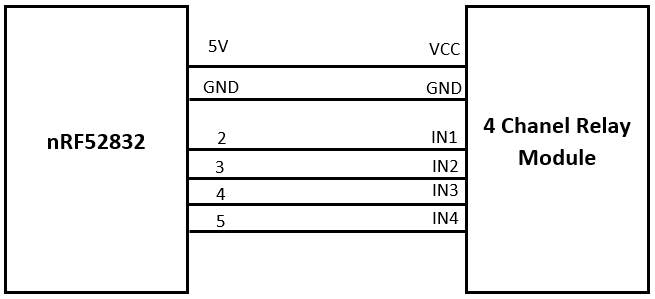
Hình 32. Cách giao tiếp I2C để đọc giá trị nhiệt độ

Node cảm biến nhiệt độ sẽ giao tiếp I2C với chip LM75B để đọc được nhiệt độ đo được. Kết quả sau khi được đọc từ chip sẽ được Node gửi đến Gateway để gửi lên Server với chu kỳ 5 giây.

## Thiết kế Node điều khiển bật tắt

Relay 5V có rơ-le giúp đóng ngắt nguồn cấp cho các thiết bị dùng điện thế cao AC hoặc DC. Rơ-le hoạt động tại điện áp 5V DC, chịu được hiệu điện thế lên đến 250V AC 10A. Trên module đã có sẵn mạch kích relay sử dụng transistor và IC cách ly quang giúp cách ly hoàn toàn mạch điều khiển (vi điều khiển) với rơ le bảo đảm vi điều khiển hoạt động ổn định. Có sẵn header rất tiện dụng khi kết nối với vi điều khiển. Relay sử dụng chân kích mức Thấp (0V), khi có tín hiệu 0V vào chân IN thì đèn LED ở đầu ra sẽ sáng lên. Nếu không, nó sẽ tắt.

Node Relay được thiết kế bao gồm nRF52832 và module relay, nguồn nuôi cho relay được nối trực tiếp từ nRF52832, do vậy ta phải dùng một adapter 5V 1A (hoặc 2A) để cấp nguồn trực tiếp cho nRF52832. Thực hiện kết nối các chân IN của module relay vào các chân pin của nRF52832 và điều khiển bằng cách kích mức thấp cho các chân mong muốn.



Hình 33. Sơ đồ kết nối nRF52832 với module relay 4 kênh

## Thiết kế Node Tracking

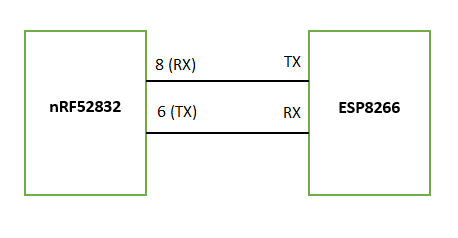
Node Tracking là những Node chịu trách nhiệm advertise tín hiệu BLE để phục vụ cho ứng Tracking nhưng bản thân nó vẫn là một Node trong mạng Mesh vẫn đảm bảo khả năng relay message.

Bình thường Node Tracking sẽ liên tục advertise tín hiệu theo chuẩn BLE bình thường để cho ứng dụng Tracking hoạt động. Khi nhận được một tín hiệu Bluetooth theo chuẩn Bluetooth Mesh, Node Tracking sẽ relay tín hiệu đó theo chuẩn Bluetooth Mesh như những Node khác trong mạng Mesh. Sau đó, Node Tracking lại tiếp tục advertise tín hiệu theo chuẩn BLE thông thường.

Tracking Node vẫn có thể gắn với các thiết bị và hoạt động một cách bình thường như các Node khác nhưng để phân biệt rõ chức năng, nhóm sẽ không gắn thiết bị lên các Node này.

## Thiết kế Gateway

Gateway được tích hợp 2 thành phần chính là một Node nRF52832 giao tiếp với module wifi ESP8266 thông qua chuẩn giao tiếp UART. Gateway cho phép người dùng có thể điều khiển, đọc được các dữ liệu trên các Node trong mạng Bluetooth Mesh thông qua Internet.



Hình 28. Sơ đồ kết nối nRF52832 với ESP8266

Node nRF52832 trong Gateway (Node Gateway) có nhiệm vụ truyền message điều khiển khi có yêu cầu từ Server và thu thập dữ liệu trạng thái của tất cả các Node trong mạng Mesh. Để quản lý các Node thiết bị và Tracking, Node Gateway sẽ thiết lập 9 model subscribe đến 9 Node trong mạng Mesh. Đối các Node thiết bị có điều khiển được, Node nRF52832 trong Gateway sẽ publish tới địa chỉ mà các Node thiết bị đã subscribe.

ESP8266 có nhiệm vụ gửi dữ liệu của các Node thiết bị được thu thập và truyền qua giao tiếp UART từ Node Gateway lên Server và nhận dữ liệu từ Server để triển khai vào mạng Bluetooth Mesh cũng bằng giao tiếp UART với Node Gateway. Dữ liệu từ Node Gateway gửi lên Server có định dạng là **“ID,value”** trong đó ID là định danh của thiết bị và value là giá trị trạng thái của thiết bị. Các yêu cầu điều khiển từ Server gửi đến Gateway cũng có định dạng như trên, Node Gateway sẽ phân tích ID, tìm ra Node thiết bị tương ứng trong mạng Mesh để truyền message.

Việc gửi và nhận dữ liệu từ Server của ESP8266 được thực hiện bằng giao thức MQTT. ESP8266 sẽ publish dữ liệu với topic “SWITCH\_STATUS” khi nhận được dữ liệu trạng thái của switch từ Node Switch, “SENSOR\_UPDATE” khi nhận được dữ liệu nhiệt độ từ Node cảm biến. Ở chiều nhận, ESP8266 sẽ subscribe đến topic “SWITCH\_CONTROL” khi có yêu cầu điều khiển từ Server.

## Thiết kế mô hình mạng Bluetooth Mesh

Hãng sản xuất chip nRF52832, Nordic Semiconductor có cung cấp một bộ SDK là nRF5 SDK for Mesh hỗ trợ việc xây dụng mạng Bluetooth Mesh. Bộ SDK này cung cấp cài đặt cho toàn bộ Stack, đặc tính, giao thức, các phân lớp theo chuẩn Bluetooth Mesh giúp cho những nhà phát triển chi cần tập trung vào tầng Application và tầng Model, hai tầng chính để phát triển ứng dụng trên nền tảng Bluetooth Mesh. Bộ SDK của hãng liên tục được cập nhật và luôn có sự hỗ trợ từ diễn đàn của hãng. Ứng dụng nhóm phát triển dựa trên phiên bản 3.2.0.

Các Node muốn tham gia mạng Bluetooth Mesh đều phải được Provision. Nhóm sử dụng phương thức PB-GATT thông qua smartphone làm phương thức Provision cho mạng Bluetooth Mesh. Ứng dụng nRF Mesh do Nordic cung cấp hỗ trợ việc Provision, quản lý các Node, dễ dàng thay đổi địa chỉ publish, subscribe của từng model trong các Node. Ngoài ra nRF Mesh còn có tính năng chia sẻ cấu hình của mạng Mesh cho nhiều smartphone khác nhau, cho phép nhiều smartphone có thể thao tác với cùng một mạng Mesh.

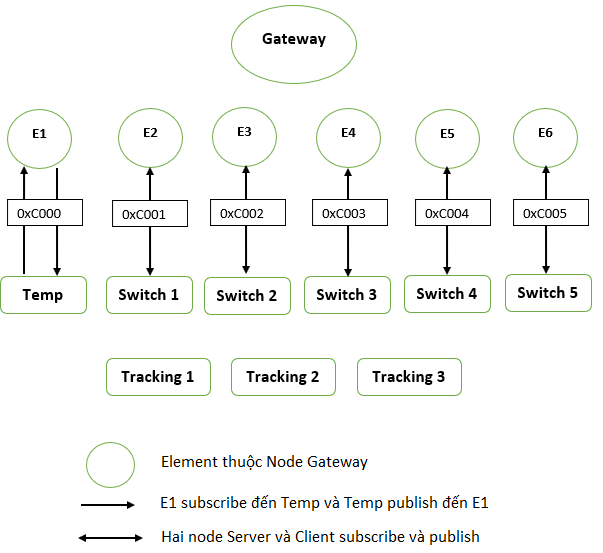
nRF5 SDK for Mesh phiên bản 3.2.0 định nghĩa sẵn model phục vụ việc bật tắt là Generic OnOff Model. Generic OnOff Model định nghĩa một state đơn giản cho việc bật tắt bao gồm hai giá trị là 0 và 1.



Hình 35. Generic OnOff Model

Dựa trên cài đặt của Generic OnOff Model, nhóm xây dựng một Model tương tự nhưng dữ liệu State là một giá trị Integer thay vì chỉ có 0 và 1, dữ liệu được dùng cho việc truyền nhiệt độ đọc từ cảm biến.

Sơ đồ thiết kế mạng Bluetooth Mesh:



Hình 36. Sơ đồ thiết kế mạng Bluetooth Mesh

Trong sơ đồ bao gồm một Node Gateway, một Node cảm biến nhiệt độ, 5 Node công tắc và 3 Node Tracking:

* Gateway bao gồm 6 Element chính, mỗi Element được gán với model tương ứng với loại dữ liệu cần truyền trong mạng Mesh. Tất cả các model được gán với Element trong Node Gateway đều subscribe tới các Node thiết bị nhằm mục đích nhận message trạng thái từ thiết bị. Các Element thực hiện điều khiển thiết bị, E2 đến E6 sẽ publish message đến các Node thiết bị.
* Tất cả các Node thiết bị đều publish tới các Element thuộc Node Gateway để gửi trạng thái của chúng đến Gateway. Các Node thiết bị được điều khiển bởi Gateway sẽ subscribe đến các Element để nhận message điều khiển từ Gateway.
* Các Node Tracking không được gắn thiết bị nào nên chúng sẽ không publish, subscribe nhưng chúng vẫn là một Node trong mạng Mesh và luôn đảm bảo chức năng relay message như các Node khác.
* Một Model trong Element đều có một ID, do đó một Node thiết bị khi publish, subscribe sẽ tương ứng với một Model Id. Giá trị ID này nằm trong cặp giá trị {nodeID:value} mà Gateway sẽ gửi lên Server.

Tất cả các Node trên đều có tính năng Relay, Proxy:

* Relay là tính năng cốt lõi của Bluetooth Mesh cho phép các Node có thể nhận và phát lại message. Do đó message từ một Node có thể truyền đến một Node hoàn toàn nằm ngoài vùng phủ sóng của chúng thông việc phát lại message từ các Node trung gian và phạm vi giao tiếp giữa các thiết bị được mở rộng ra rất nhiều.
* Proxy là tính năng cho phép một Node theo chuẩn Bluetooth Mesh có thể giao tiếp với các thiết bị không hỗ trợ Bluetooth Mesh thông qua GATT như smartphone. Tính năng proxy được nhóm sử dụng cho việc provision các Node vào mạng Mesh.

Với các Node Switch, sau khi thực hiện được lệnh điều khiển từ Gateway, chúng sẽ ngay lập gửi về trạng thái OnOff của thiết bị về lại cho Gateway dưới dạng STATUS message. Trạng thái của Switch nhằm mục đích để xác nhận kết quả điều khiển khi người dùng bật tắt thiết bị.

Dữ liệu trạng thái của thiết bị được gửi đi 1 lần trong mỗi 5 giây nhằm mục đích để xác nhận thiết bị vẫn còn hoạt động trong mạng Mesh. Nếu Server không nhận được bất cứ một giá trị trạng thái nào từ thiết bị trong một khoảng thời gian, nó sẽ gửi sự kiện Node không còn hoạt động đến ứng dụng Android.

## Thiết kế Server với Node.js

Node.js xuất thân từ ngôn ngữ JavaScript nhưng có thể chạy trên Server. Node.js hoạt động chỉ trên thread duy nhất và các function trong Node.js đều chạy dưới dạng Asynchronous. Node.js tiếp nhận các request và đăng ký cho các request đó một hàm callback thay vì chờ request đó hoàn tất. Khi request đó kết thúc, Server sẽ thực hiện một hàm callback. Node.js cho khả năng lập trình theo hướng sự kiện và callback nên rất phù hợp với các ứng dụng real-time, cần sự phản hồi nhanh và chính xác trong khi vẫn tiếp nhận rất nhiều request. Vì vậy, nhóm quyết định sử dụng Node.js để thiết kế Server.

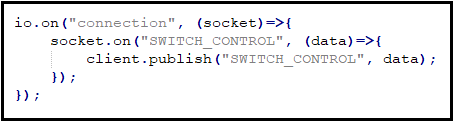
Server có ba chức năng chính:

* Chuyển tiếp yêu cầu điều khiển từ ứng dụng Android đến ESP8266 trong Node Gateway.
* Chuyển tiếp dữ liệu từ ESP8266 cho người dùng trên ứng dụng Android.
* Chức năng thứ ba của Server là phát hiện một Node không còn hoạt động và thông tin cho ứng dụng Android.

### Chuyển tiếp yêu cầu điều khiển từ Android đến Gateway

Socket.io là thư viện được xây dựng dựa trên giao thức Websockets cho phép giao tiếp giữa Client và Server. Điểm đặc biệt ở Socket.io là các API của thư viện hoạt động theo hướng sự kiện ở cả phía Client và Server. Cơ chế lắng nghe và phát sự kiện của Socket.io đáp ứng được tính real-time của ứng dụng, giúp việc phát triển ứng dụng IOT dễ dàng hơn.

Trong mô hình hệ thống của dự án, phía ứng dụng Android chỉ có yêu cầu điều khiển các thiết bị trong mạng Bluetooth Mesh. Nên Server lắng nghe sự kiện “SWITCH\_CONTROL” từ Android. Khi nhận được sự kiện “SWITCH\_CONTROL” được phát đi từ Android, Server sẽ gọi hàm callback sử dụng MQTT chuyển tiếp dữ liệu từ Android đến Gateway trong mạng Bluetooth Mesh.

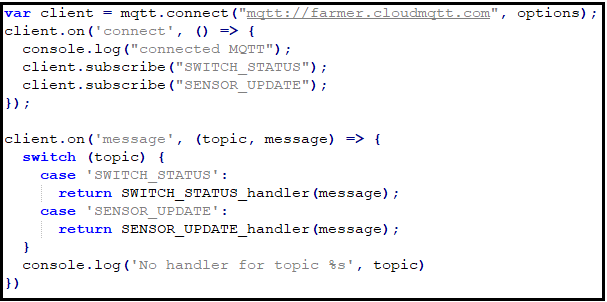


Hình. Server xử lý sự kiện từ Android

### Chuyển tiếp yêu cầu điều khiển từ Android đến Gateway

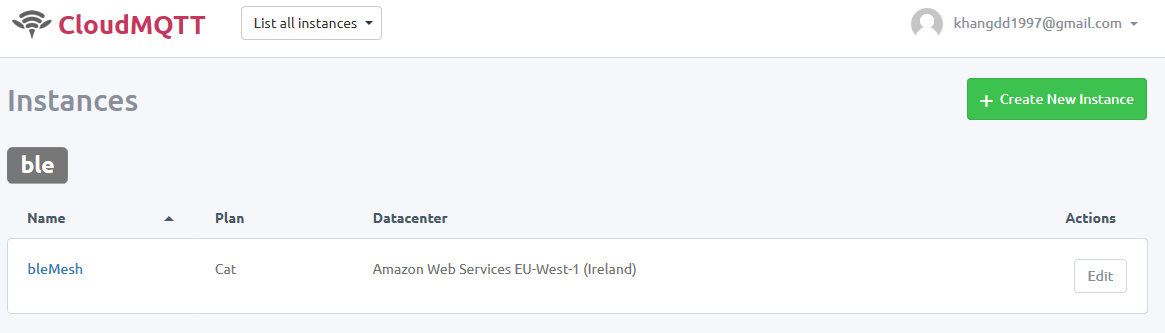
MQTT là một giao thức truyền thông điệp theo mô hình publish\subscribe sử dụng băng thông thấp và độ tin cậy cao, phù hợp với các ứng dụng điều khiển thiết bị. Node.js có hỗ trợ bộ thư viện MQTT thông qua lệnh cài đặt “*npm install mqtt”.* Các API trong thư viện này cũng hoạt động theo cách thức lắng nghe và phát sự kiện giống như Socket.io.

Trong mô hình hệ thống của dự án, phía Gateway (cụ thể là ESP8266) sẽ publish hai topic là “SWITCH\_STATUS” ứng với dữ liệu trạng thái của Node switch và “SENSOR\_UPDATE” ứng với dữ liệu nhiệt độ của Node cảm biến. Vì vậy, Server cần phải subscribe đến hai topic này để nhận được dữ liệu từ Gateway. Ứng với mỗi topic, Server sẽ gọi hàm xử lý và phát sự kiện Socket.io để thông báo cho Android.



Hình. Các hoạt động của MQTT trên Server

Để giao thức MQTT hoạt động giữa các Client, ta cần có một Broker như là một Server chuyên để xử lý các message, giao tiếp giữa các Client, thực hiện các tính năng theo chuẩn giao thức MQTT. CloudMQTT (<https://www.cloudmqtt.com/>) là một website cho phép đăng ký Broker miễn phí, Broker được host trên Server của Amazon và MQTT hoạt động khá ổn định.



Hình. Giao diện đăng ký Broker của CloudMQTT

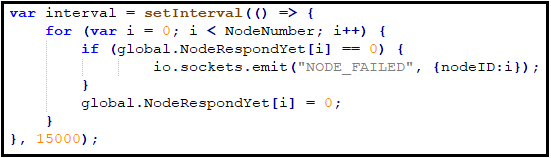
### Phát hiện Node không còn hoạt động trong mạng Mesh

Trong quá trình hoạt động, các Node thiết bị trong mạng Mesh sẽ liên tục gửi dữ liệu trạng thái của mình cho Node Gateway với mục đích thông tin và xác nhận Node đang hoạt động. Khi một Node bị hỏng hoặc lỗi, nó sẽ ngừng gửi dữ liệu trạng thái và Server sẽ chịu trách nhiệm phát hiện và thông báo cho Android.

Server sẽ khởi tạo một array NodeResponseYet có số lượng phần tử tương đương với số lượng thiết bị trong mạng Mesh. Giá trị của các phần tử sẽ là 0 tương ứng với Node chưa gửi dữ liệu trạng thái và 1 tương ứng Node đã gửi dữ liệu trạng thái. Việc gửi dữ liệu của các Node sẽ được Server ghi nhận trong quá trình xử lý MQTT từ ESP8266.

Để phát hiện Node ngừng gửi dữ liệu trạng thái, một timer sẽ chạy tuần hoàn trên Server với thời gian được thiết lập (15 giây). Cứ mỗi lần kết thúc 15 giây, Server kiểm tra array NodeResponseYet, Node không gửi trong suốt khoảng thời gian timer sẽ có giá trị là 0 và Server sẽ phát sự kiện Socket.io “NODE\_FAILED” cho Android kèm với ID của Node thiết bị.

Socket.io là thư viện được xây dựng dựa trên giao thức Websockets cho phép giao tiếp giữa Client và Server. Điểm đặc biệt ở Socket.io là các API của thư viện hoạt động theo hướng sự kiện ở cả phía Client và Server. Cơ chế lắng nghe và phát sự kiện của Socket.io đáp ứng được tính real-time của ứng dụng, giúp việc phát triển ứng dụng IOT dễ dàng hơn.

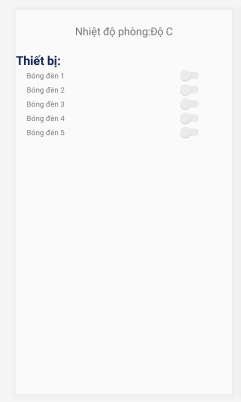


Hình. Server xử lý phát hiện Node không hoạt động

## Thiết kế ứng dụng Android cho ứng dụng điều khiển

Ứng dụng Android được thiết kế chuyên biệt cho một tòa nhà, do đó ứng dụng sẽ gồm 2 mục dành cho nhân viên tòa nhà (Staff) và khách (Guest). Mục nhân viên thực hiện ứng dụng điều khiển, có yêu cầu đăng nhập tài khoản của nhân viên nhằm đảm bảo chỉ có nhân viên tòa nhà được quyền quản lý các thiết bị. Mục Guest sẽ thực hiện chức năng Tracking và không yêu cầu đăng nhập.

Giao diện ứng dụng điều khiển:



Trong mạng Mesh các Node sẽ liên tục gửi dữ liệu trạng thái của mình, OnOff đối với Switch và nhiệt độ đối với Node cảm biến, cho Gateway và Server. Mục đích của việc gửi trạng thái của Node là để cập nhật tình trạng hiện tại của Node cũng như phát hiện Node không còn hoạt động. Giá trị của cảm biến nhiệt độ sẽ được cập nhật. Khi Server nhận được dữ liệu nhiệt độ, đồng thời là trạng thái của Node cảm biến nhiệt độ. Người dùng chỉ có thể điều khiển các Switch khi Server nhận được dữ liệu trạng thái của Node và gửi về cho Android.

Khi người dùng thực hiện điều khiển Switch, Android sẽ emit sự kiện Socket.io cho Server và Server gửi publish MQTT về Gateway để thực hiện điều khiển. Lúc này trạng thái của Switch trên sẽ giữ nguyên cho đến khi Node thực hiện OnOff và gửi kết quả theo dạng trạng thái về Gateway và Server chuyển tiếp cho Android thì trạng thái của Switch mới thay đổi.

## Thiết kế ứng dụng Android cho ứng dụng Tracking

Bản đồ tòa nhà E tầng 6:



Ứng dụng Tracking của nhóm sử dụng phương pháp Fingerprinting. Phương pháp Fingerprinting sử dụng một radio map chứa các tọa độ của bản đồ và tập giá trị RSSI đo được ngay tại điểm đó. Tập hợp nhiều điểm tọa độ trải rộng trên bản đồ sẽ tạo thành một radio map.

Bước đầu tiên để ứng dụng Tracking hoạt động được theo phương pháp này là xây dựng một radio map tại tầng 6 nhà E. Nhóm thiết kế một ứng dụng Android riêng phục vụ cho việc thu thập thông tin tọa độ và tập giá trị RSSI tương ứng với tọa độ đó.

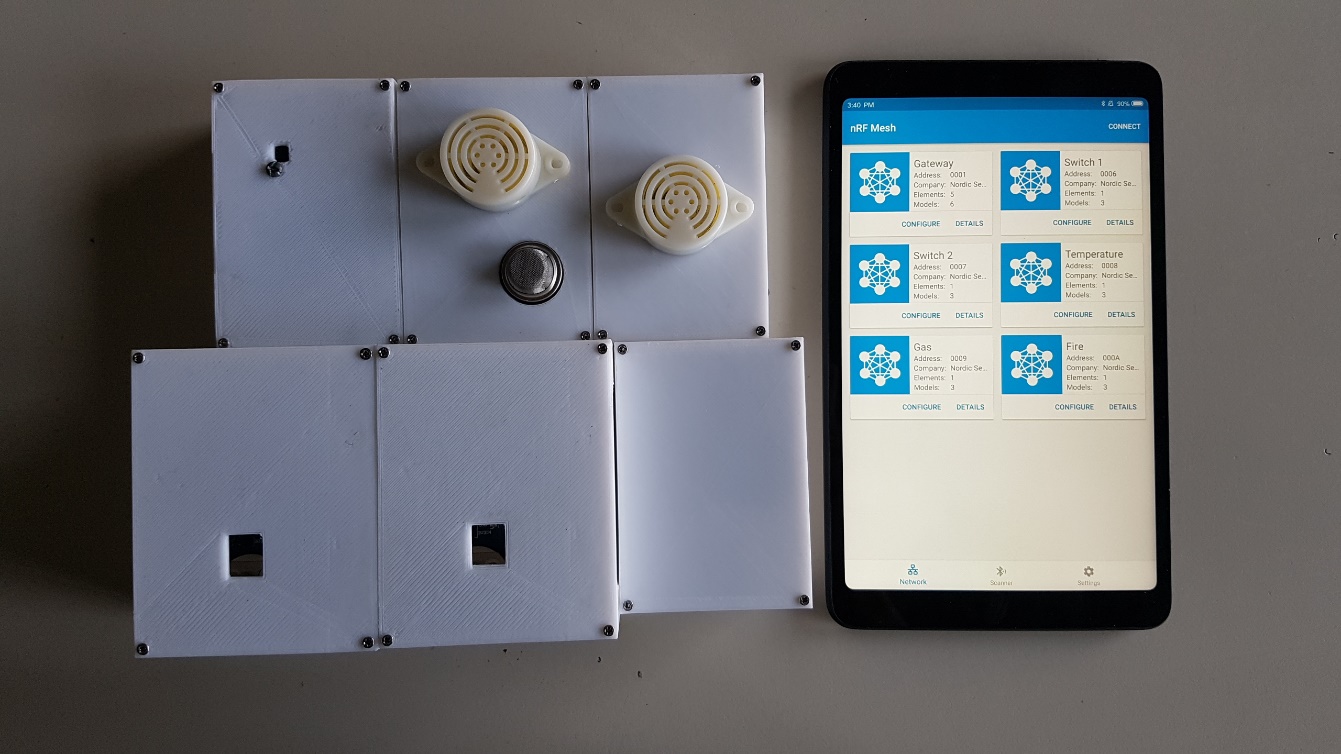
Ứng dụng Android cần phải thu thập được RSSI từ 2 Node Tracking trở lên để tập giá trị RSSI có thể đại diện cho một vị trí tọa độ trong bản đồ. Nhóm bố trí 3 Node Tracking trên tầng 6 nhà E. Mỗi Node sẽ advertise tên thiết bị theo cú pháp: “TRACKING E06 1” với TRACKING biểu thị cho Android nhận dạng Node dùng cho ứng dụng Tracking, E06 là số tầng của tòa nhà, 1 là ID của Tracking Node tại tầng đó. Nhóm sẽ phân bố các điểm tọa độ trên khắp bản đồ ví dụ (1m;1m) (2m;2m) … và đứng ngay tại vị trí đó để quét tập giá trị RSSI, sau đó lưu trên CSDL.

Sau khi xây dựng được radio map, ứng dụng Tracking theo phương pháp Fingerprinting sẽ hoạt động. Khi vào ứng dụng Tracking trên giao diện, Android sẽ tải radio map từ CSDL về. Sau đó Android sẽ quét tập giá trị RSSI tương tự như trong quá trình xây dựng radio map và đem so sánh với tập giá trị RSSI trong radio map để tìm ra tập giá trị gần giống nhất và suy ra được tọa độ tương ứng và hiển thị trên bản đồ ứng dụng Android.

# KẾT QUẢ

## Các thiết bị trong hệ thống

Mạng Bluetooth mesh được hiện thực là một hệ thống bao gồm 6 nodes: Gateway, 2 node công tắc, node cảm biến nhiệt độ, node cảm biến gas, node cảm biến lửa. Nhóm thực hiện thiết kế và in các mô hình 3D cho từng node để đóng gói thành mô hình thực tế phục vụ cho việc thử nghiệm hệ thống.



**Tempera-  
ture**

**Gas**

**Fire**

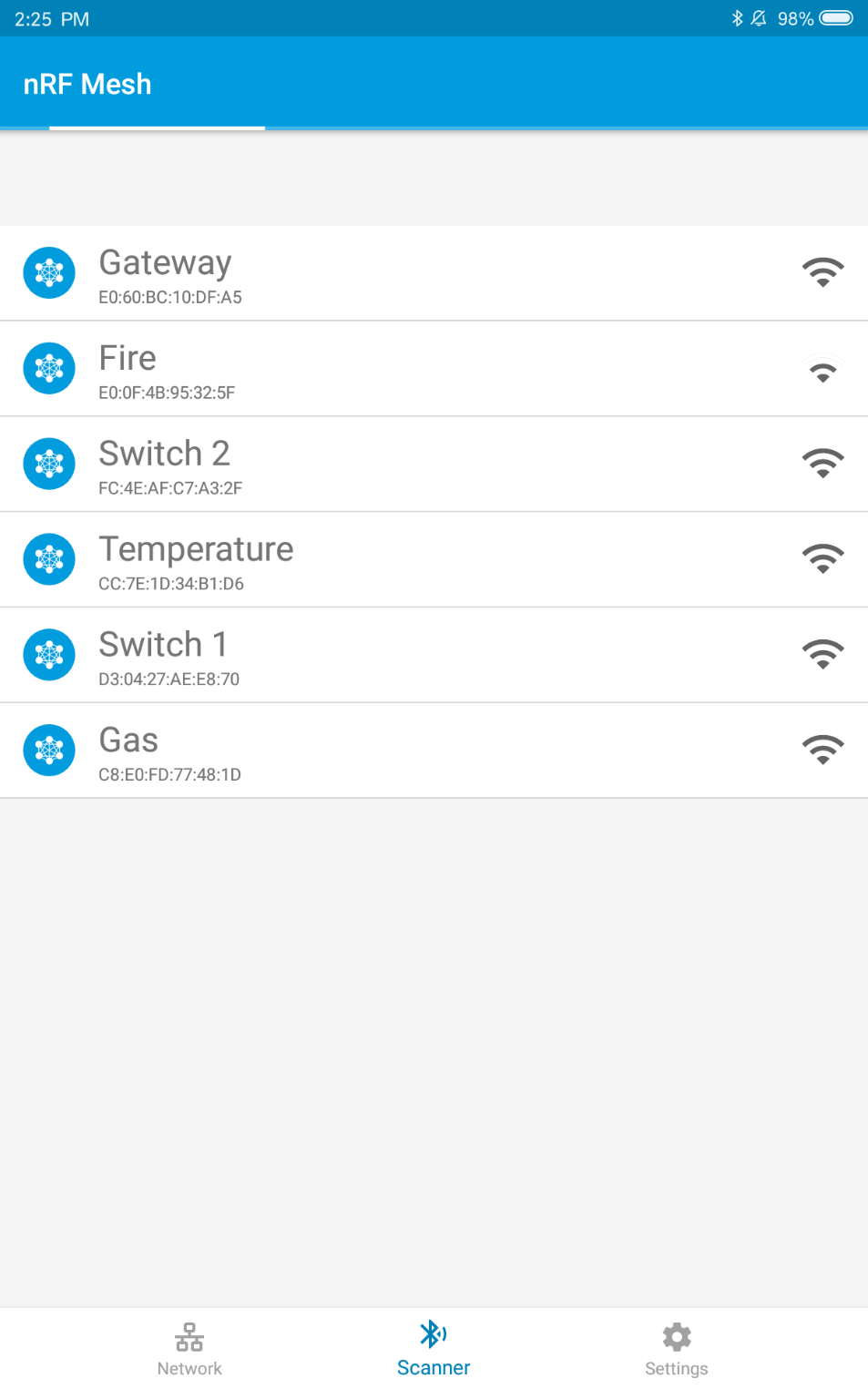
**Gateway**

**Switch 1**

**Switch 2**

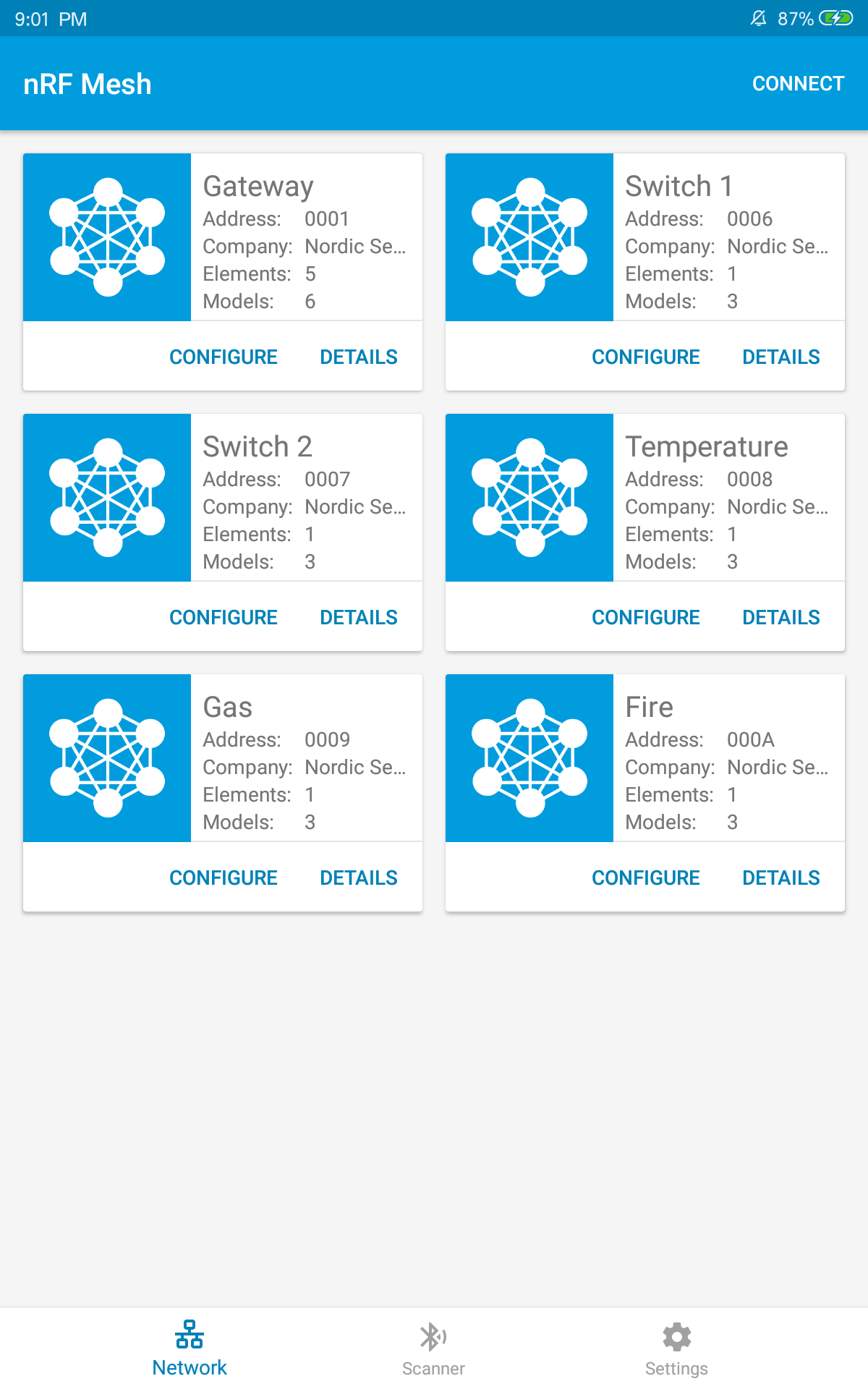
Hình 55. Các thiết bị trong hệ thống

Ngoài các node thì tablet cũng là một thiết bị quan trọng cần thiết trong việc lắp đặt hệ thống. Tablet đóng vai trò là provisioner và thực hiện quá trình provisioning, biến các thiết bị thành các node trong mạng mesh, đồng thời giúp cho việc cấu hình publish/subscribe giữa các node một cách dễ dàng.



Hình 56. Các thiết bị unprovisioned đang sẵn sàng để được provision

Tiến hành cấu hình Publish/Subscribe cho các node bằng úng dụng nRF Mesh theo cơ cấu Publish/Subscribe đã trình bày trong Hình 37. để hoàn tất hệ thống.

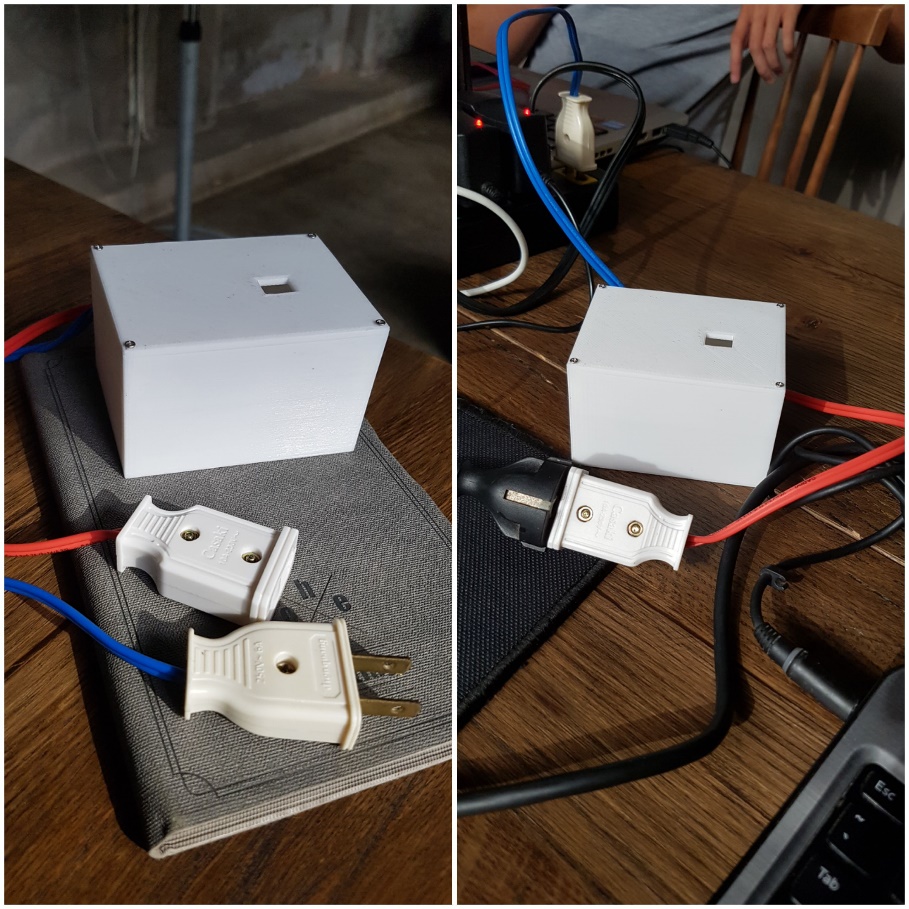


Hình 57. Các thiết bị đã được provision và thực hiện cấu hình hoàn tất

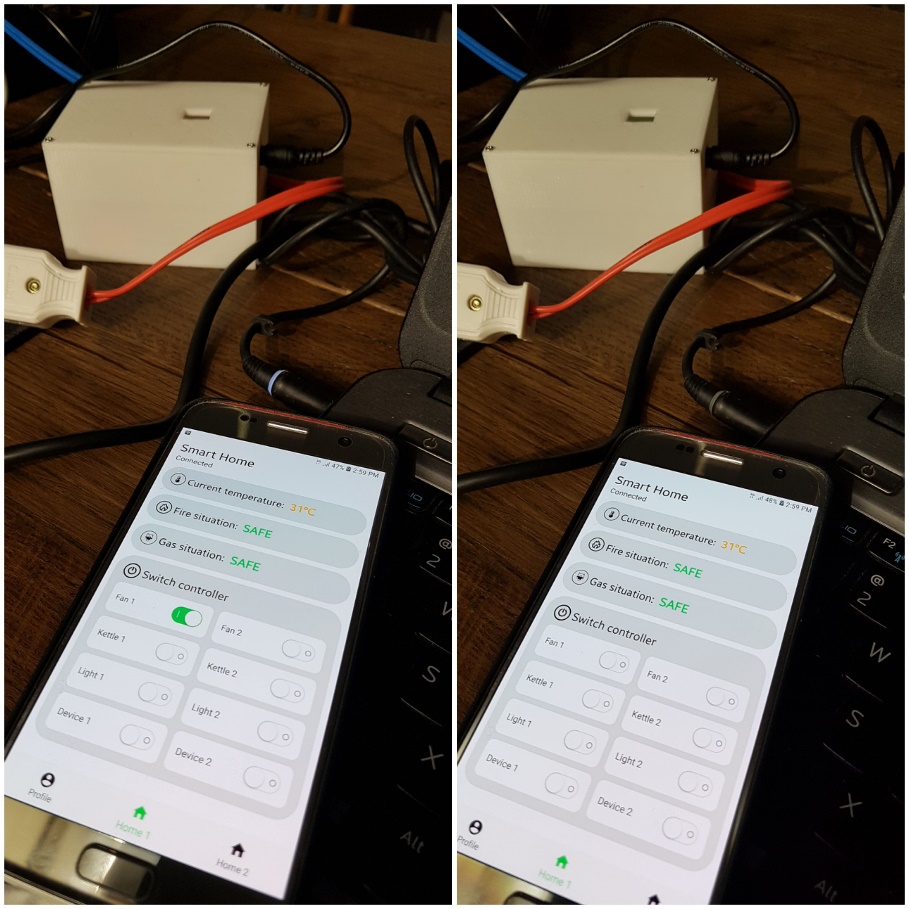
## Đọc thành công các thông số cảm biến và điều khiển các node công tắc

Tiến hành thực nghiệm với từng node:

* + Điều khiển node công tắc: thực hiện kết nối node công tắc với nguồn điện 220V thông qua module relay 4 kênh, cho phép điều khiển bật tắt các thiết bị điện xoay chiều. Cụ thể như trong hình dưới, nhóm kết nối bộ sạc máy tính có đèn LED báo nguồn với nguồn điện 220V thông qua node công tắc.

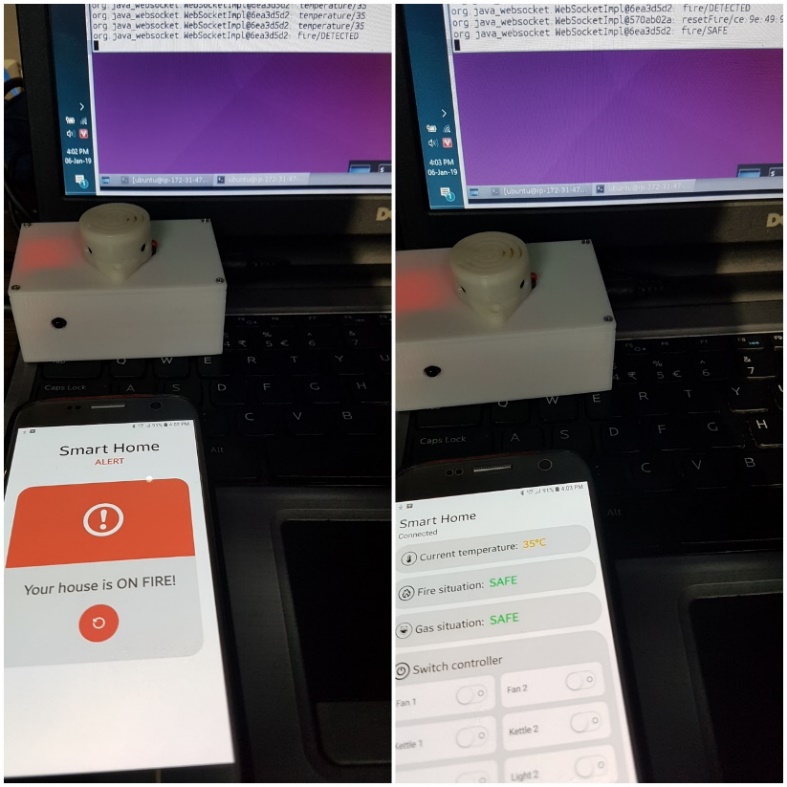


Hình 58. Kết nối node công tắc với bộ sạc laptop

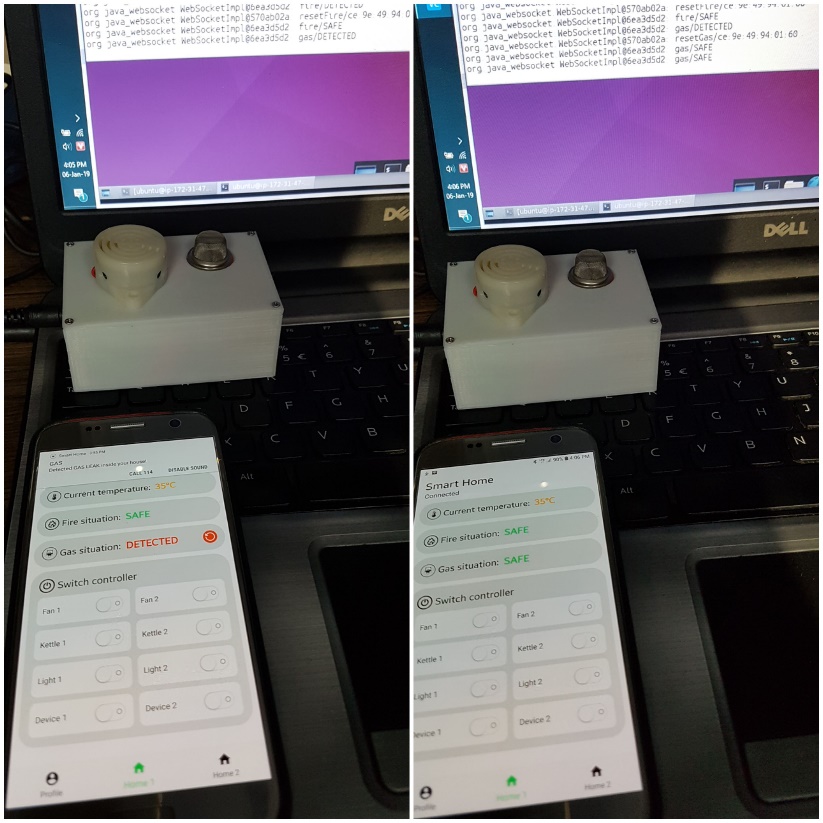


Hình 59. Bật/Tắt bộ sạc bằng app Smart Home

* + Đọc dữ liệu cảnh báo từ cảm biến lửa và cảm biến gas : thực hiện thử nghiệm gửi thông báo từ các node cảm biến và nhanh chóng cập nhật thông báo cho người dùng, kiểm tra trực tiếp tin nhắn trên Server:



Hình 60. Thử nghiệm node cảm biến lửa



Hình 61. Thử nghiệm node cảm biến gas

## Thực nghiệm tổng quan hệ thống

* + 1. **Thực nghiệm tính năng relay message với 3 node**

Để thực nghiệm tính năng truyền tải message với các node nằm ngoài phạm vi đối với Gateway, nhóm tiến hành thiết lập vị trí cho các node như sau:

* + **Gateway**: được đặt tại bàn học, kết nối gateway trực tiếp với Adapter và cần thiết kết nối Gateway với laptop bằng J-Link để kiểm tra cụ thể message trả về.



Hình 62. Vị trí Gateway trong thử nghiệm Relay message với 3 node

* + **Node bất kỳ làm Relay node**: được đặt tại cửa sổ hành lang, cách bàn học khoảng hơn 7m và cách nhau một bức tường. Node này đóng vai trò làm Relay node đảm nhiệm tính năng tái truyền tải message gửi từ node khác cho Gateway.



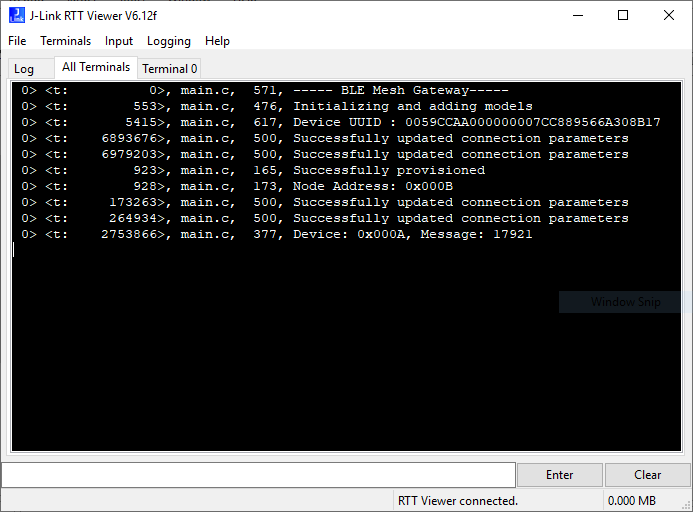
Hình 63. Relay node trong thử nghiệm Relay message với 3 node

* + **Node Fire**: được đặt trên một đường thẳng với Relay node, khoảng cách hơn 12m, được cấp nguồn thông qua pin CR2032 để tiện cho việc thử nghiệm. Dùng bật lửa để thực hiện gửi một message cảnh báo về tình trạng lửa đến Gateway.



Hình 64. Thực hiện gửi message từ node Fire

Để kiểm tra chính xác message đã được Relay thành công, ta sử dụng J-Link RTT Viewer. Message được relay về sẽ có thông tin về địa chỉ của Device đã gửi tương ứng là **0x000A**, chính là Node Address mà provisioner đã cung cấp cho Node Fire.



Hình 65. Message được relay về từ node Fire

* + 1. **Thực nghiệm toàn bộ hệ thống**

Tiến hành thực nghiệm toàn bộ hệ thống với tổng cộng 6 node đã được thiết kế. Do 2 node switch tiêu thụ điện năng khá cao nên nhóm đặt chúng ở vị trí Relay node và Relay node thay thế (cho thử nghiệm trường hợp node bị hỏng, xảy ra sự cố khiến node không hoạt động) và ngắt kết nối nguồn nuôi từ nRF52832 cho module relay 4 kênh, sử dụng pin CR2032 cấp nguồn cho 2 node switch này.



Hình 66. Hai Switch trong vai trò làm Relay node và Relay node thay thế

Node cảm biến nhiệt độ có thể hoạt động bình thường với nguồn là pin CR2032 nên được đặt ở cửa tầng 5, một vị trí vẫn trong phạm vi tiếp cận so với vị trí đặt node cảm biến lửa (Hình 64)



Hình 67. Vị trí đặt node cảm biến nhiệt độ

Node cuối cùng là node cảm biến Gas được đặt ở vị trí cửa tầng 4, nơi có thể thực hiện kết nối với node cảm biến nhiệt độ.



Hình 68. Vị trí đặt node cảm biến Gas

Hoàn thành thực nghiệm tổng quan hệ thống bằng cách thực hiện gửi cảnh báo cho người dùng từ node cảm biến gas, sau đó thực hiện hành động tắt chuông cảnh báo ở node cảm biến gas để xác nhận message đã được gửi thành công thông qua mạng Bluetooth mesh. Thử nghiệm Relay node thay thế bằng cách tắt một trong hai node switch, hệ thống hoạt động bình thường.

# KẾT LUẬN – KIẾN NGHỊ

## Kết luận

Hệ thống cơ bản hoàn thành. Các node thiết bị đã được đóng gói có thể lắp đặt và triển khai thành hệ thống mạng mesh trong nhà phục vụ cho việc điều khiển của người dùng một cách bình thường. Về phía ứng dụng cho người dùng cũng như server đã được triển khai khá hoàn chỉnh, giúp cho người dùng có thể điều khiển các thiết bị, cập nhật dữ liệu về nhiệt độ, cảnh báo về tình trạng gas, lửa một cách chính xác.

Nhóm đã tiến hành thực nghiệm tổng quan hệ thống và đo đạc, phân tích các thông số qua các lần thử nghiệm. Nhìn chung, việc truyền tín hiệu trong hệ thống mạng mesh khá ổn định, giúp cho các node ở phạm vi xa có thể cung cấp dữ liệu kịp thời, đồng thời có thể đảm bảo được dữ liệu khó có thể bị mất ngay cả khi có trường hợp một node xảy ra sự cố.

Với các chức năng như nhóm đã trình bày, hệ thống này đã đáp ứng được việc xây dựng một hệ thống mạng mesh cho các thiết bị điều khiển và cảm biến trong nhà trên nền tảng Bluetooth sử dụng nRF52832.

## Kiến nghị

Với thời gian tìm hiểu và phát triển đề tài trong khoảng 5 tháng, chắc chắn sẽ vẫn còn những trường hợp lỗi chưa được phát hiện. Do đó, đề tài cần có thêm nhiều thời gian để tìm hiểu và kiểm thử.

Đề tài cũng cần nghiên cứu để phát triển thành một hệ thống mạng mesh rộng lớn hơn nữa cho các thiết bị tích hợp công nghệ Bluetooth như kết nối đến các khu vực trong vườn, nhà kho, ga-ra….

TÀI LIỆU THAM KHẢO

1. Bluetooth SIG, “**Bluetooth Mesh Specification**”.
2. Bluetooth SIG, “**Bluetooth Mesh Model Specification**”
3. Bluetooth SIG, Martin Woolley, Sarah Schmidt, “**Mesh Technology Overview**”.
4. Bluetooth SIG, Martin Woolley, “**Bluetooth Mesh Developer Study Guide**”
5. Martin Woolley, “**In-Market Bluetooth Low Energy Devices and Bluetooth Mesh Networking**”, blog.bluetooth.com, September 05, 2017.
6. Mohammad Afaneh, “**The Ultimate Bluetooth Mesh Tutorial**”.
7. Nordic Support Community, https://devzone.nordicsemi.com

1. Theo số liệu của Pew Research Center khảo sát vào tháng 1/2018 [↑](#footnote-ref-1)
2. 2 Theo số liệu của báo điện tử news.zing.vn với tiêu đề “Nhà thông minh – xu hướng công nghiệp” [↑](#footnote-ref-2)
3. 2 Bluetooth SIG, “**Bluetooth Mesh Glossary of Terms**”, Limits section. [↑](#footnote-ref-3)
4. Martin Woolley, “**Mesh Technology Overview**”, Model section. [↑](#footnote-ref-4)
5. Bluetooth SIG, “**Bluetooth Mesh Glossary of Terms”**, Limits section. [↑](#footnote-ref-5)
6. 3 Martin Woolley, “**In-Market Bluetooth Low Energy Devices and Bluetooth Mesh Networking**” [↑](#footnote-ref-6)