

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**ĐỒ ÁN TỐT NGHIỆP**

**Xây dựng hệ thống quản lý thông tin tập trung trên  
giấy điện tử phiên bản 2**

**ĐỖ ANH LINH**

linh.da194314@sis.hust.edu.vn

**Ngành Kỹ thuật máy tính**

**Giảng viên hướng dẫn:** ThS. Nguyễn Đức Tiến

Chữ kí GVHD

**Khoa:** Kỹ thuật máy tính

**Trường:** Công nghệ Thông tin và Truyền thông

**HÀ NỘI, 06 / 2024**

# LỜI CẢM ƠN

Lời đầu tiên, em xin được gửi lời cảm ơn chân thành và sự tri ân sâu sắc đến tất cả các thầy cô trường Công nghệ Thông tin và Truyền thông – Đại học Bách khoa Hà Nội, đã giảng dạy và truyền đạt những bài học tâm huyết, những kiến thức vô cùng quý giá, cũng như hỗ trợ và tạo điều kiện thuận lợi cho em trong quá trình thực hiện đồ án tốt nghiệp. Đồ án tốt nghiệp không chỉ là một bước đầu mới trong hành trình học tập của em mà còn là cơ hội để em có thể phát triển và hoàn thiện bản thân.

Đặc biệt, em xin bày tỏ lòng biết ơn chân thành đến ThS. Nguyễn Đức Tiến là Giảng viên hướng dẫn và cũng là Cố vấn học tập của lớp Kỹ thuật Máy tính 02 - K64. Trong quá trình thực hiện đồ án, nhờ có sự giúp đỡ và góp ý của thầy đã góp phần giúp em phát triển thêm về hiểu biết cũng như là đóng góp quan trọng cho việc hoàn thành đồ án.

Lời cuối cùng em xin được cảm ơn gia đình, người thân và bạn bè đã luôn động viên, ủng hộ em trong suốt thời gian vừa qua. Cảm ơn những người bạn, người em ở Hội Sinh viên Đại học Bách khoa Hà Nội đã luôn sát cánh và hỗ trợ em trong suốt 5 năm học tập và hoạt động.

Với điều kiện thời gian hữu hạn và kiến thức của bản thân còn nhiều hạn chế, trong quá trình thực hiện đồ án dù em đã có nhiều cố gắng nhưng rất khó tránh khỏi những thiếu sót. Em rất mong nhận được sự đóng góp ý kiến và chỉ bảo từ các thầy cô để đồ án của em có thể hoàn thiện hơn.

*"Năm năm học không dài như bạn nghĩ, hãy chuyển động cùng chúng tôi!"*

# LỜI CAM KẾT

Họ và tên sinh viên: Đỗ Anh Linh

MSSV: 20194314

Điện thoại liên lạc: (+84) 36 347 1014

Email: linh.da194314@sis.hust.edu.vn

Lớp: Kỹ thuật Máy tính 02 - K64

Hệ đào tạo: Kỹ sư chính quy

Tôi – *Đỗ Anh Linh* – cam kết Đồ án Tốt nghiệp (ĐATN) là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của *ThS.Nguyễn Đức Tiên*. Các kết quả nêu trong ĐATN là trung thực, là thành quả của riêng tôi, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong ĐATN – bao gồm hình ảnh, bảng biểu, số liệu, và các câu trích dẫn – đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Tôi xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

*Hà Nội, ngày 30 tháng 6 năm 2024*

Tác giả ĐATN

*Đỗ Anh Linh*

# TÓM TẮT NỘI DUNG ĐỒ ÁN

Hiện nay, con người có rất nhiều cách để có thể cố định một lượng thông tin cần thiết trong một khoảng thời gian ngoài cuộc sống. Trong đó, có thể kể đến một trong những cách phổ biến nhất đó chính là sử dụng thiết bị trình chiếu, sử dụng các hình thức photocopy hay bảng tin... Tuy nhiên, các hình thức này đều có nhược điểm riêng và chưa có giải pháp. Ví dụ như là vấn đề năng lượng hay vấn đề về tính tồn tại cho các thiết bị màn hình, vấn đề về khả năng linh hoạt thông tin và quản lý dành cho các hình thức sử dụng giấy.

Chính vì vậy, giấy điện tử đã được ra đời giúp khắc phục các nhược điểm trên. Công nghệ giấy điện tử có khả năng tiêu thụ điện năng thấp và môi trường đọc thoải mái. Công nghệ này đặc biệt phù hợp với các môi trường đa dạng như trường học, bệnh viện, văn phòng và siêu thị. Đây cũng là cơ sở để em lựa chọn đề tài "**Xây dựng hệ thống quản lý thông tin tập trung trên giấy điện tử phiên bản 2**" với mục đích xây dựng một hệ thống quản lý thiết bị giấy điện tử phát triển trên nhiều nền tảng, đa giao thức và phù hợp với trải nghiệm người dùng cũng như phát triển từ những kết quả đã đạt được góp phần đưa ứng dụng của giấy điện tử vào thực tiễn.

Sinh viên thực hiện

*Đỗ Anh Linh*

## MỤC LỤC

<b>CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI.....</b>	<b>1</b>
1.1 Đặt vấn đề.....	1
1.2 Mục tiêu và phạm vi đề tài.....	2
1.3 Định hướng giải pháp.....	2
1.4 Bố cục đồ án .....	3
<b>CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU.....</b>	<b>4</b>
2.1 Khảo sát hiện trạng .....	4
2.1.1 Các hệ thống trên thị trường .....	4
2.1.2 Hệ thống đã được xây dựng.....	5
2.2 Tổng quan chức năng .....	6
2.2.1 Biểu đồ use case tổng quát .....	6
2.2.2 Biểu đồ use case phân rã "Điều khiển thiết bị" .....	8
2.2.3 Biểu đồ use case phân rã "Quản lý thông tin cá nhân".....	9
2.2.4 Biểu đồ use case phân rã "Quản lý thiết bị" .....	9
2.2.5 Biểu đồ use case phân rã "Quản lý dữ liệu".....	10
2.3 Quy trình nghiệp vụ .....	10
2.3.1 Biểu đồ luồng hoạt động chức năng "Tạo thiết bị" .....	11
2.3.2 Biểu đồ luồng hoạt động chức năng "Tạo dữ liệu" .....	12
2.4 Đặc tả chức năng .....	13
2.4.1 Đặc tả use case "Tạo thiết bị" .....	13
2.4.2 Đặc tả use case "Chỉnh sửa thiết bị qua Bluetooth" .....	14
2.4.3 Đặc tả use case "Chỉnh sửa thiết bị qua WiFi Adhoc" .....	14
2.4.4 Đặc tả use case "Xóa thiết bị" .....	16
2.4.5 Đặc tả use case "Tạo dữ liệu" .....	16

2.4.6 Đặc tả use case "Hiển thị dữ liệu" .....	17
2.4.7 Đặc tả use case "Xóa dữ liệu" .....	18
2.5 Yêu cầu phi chức năng .....	19
2.5.1 Yêu cầu về bảo mật .....	19
2.5.2 Yêu cầu về hiệu năng.....	19
2.5.3 Yêu cầu về khả năng mở rộng, nâng cấp và bảo trì .....	19
<b>CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG.....</b>	<b>20</b>
3.1 Front-end .....	20
3.1.1 Web người dùng - Trang quản lý.....	20
3.1.2 Ứng dụng trên điện thoại - Mobile App .....	21
3.2 Back-end.....	22
3.2.1 MongoDB .....	22
3.2.2 ExpressJS.....	23
3.2.3 Mosquitto MQTT.....	23
3.3 Thiết bị EPD .....	24
3.3.1 ESP32-C3 Supermini .....	24
3.3.2 WeAct-Epaper 2.9 Inch.....	24
3.4 Bluetooth Low Energy - BLE .....	24
3.5 PlatformIO.....	25
<b>CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG ....</b>	<b>26</b>
4.1 Thiết kế kiến trúc.....	26
4.1.1 Lựa chọn kiến trúc phần mềm .....	26
4.1.2 Thiết kế tổng quan.....	27
4.1.3 Thiết kế chi tiết gói .....	29
4.2 Thiết kế chi tiết.....	30
4.2.1 Thiết kế giao diện .....	30

4.2.2 Thiết kế lớp .....	36
4.2.3 Thiết kế cơ sở dữ liệu .....	47
4.3 Xây dựng ứng dụng.....	49
4.3.1 Thư viện và công cụ sử dụng.....	49
4.3.2 Kết quả đạt được .....	50
4.3.3 Minh họa các chức năng chính .....	50
4.4 Kiểm thử.....	55
4.5 Triển khai .....	59
<b>CHƯƠNG 5. CÁC GIẢI PHÁP VÀ ĐÓNG GÓP NỔI BẬT .....</b>	<b>63</b>
5.1 Hiển thị dữ liệu ảnh .....	63
5.1.1 Đặt vấn đề .....	63
5.1.2 Giải pháp .....	63
5.2 Chuyển các chế độ cho thiết bị EPD .....	65
5.3 Sử dụng công nghệ Bluetooth Low Energy (BLE) .....	66
5.3.1 Đặt vấn đề .....	66
5.3.2 Giải pháp .....	67
5.3.3 Kết quả đạt được .....	70
5.4 Sử dụng WiFi Adhoc cho thiết bị EPD.....	70
5.4.1 Đặt vấn đề .....	70
5.4.2 Giải pháp .....	71
5.5 Một số khó khăn gấp phải.....	71
5.5.1 PlatformIO .....	71
5.5.2 Bluetooth trên thiết bị di động .....	72
5.5.3 Sử dụng và kết nối WiFi .....	73

<b>CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN .....</b>	<b>74</b>
6.1 Kết luận .....	74
6.1.1 Kết quả đã đạt được.....	74
6.1.2 Hạn chế và thiếu sót .....	74
6.2 Hướng phát triển.....	75
<b>TÀI LIỆU THAM KHẢO.....</b>	<b>78</b>

## DANH MỤC HÌNH VẼ

Hình 2.1	Biểu đồ use case tổng quan . . . . .	7
Hình 2.2	Biểu đồ use case phân rã "Điều khiển thiết bị" . . . . .	8
Hình 2.3	Biểu đồ use case phân rã "Quản lý thông tin cá nhân" . . . . .	9
Hình 2.4	Biểu đồ use case phân rã "Quản lý thiết bị" . . . . .	9
Hình 2.5	Biểu đồ use case phân rã "Quản lý dữ liệu" . . . . .	10
Hình 2.6	Biểu đồ luồng hoạt động chức năng "Tạo thiết bị" . . . . .	11
Hình 2.7	Biểu đồ luồng hoạt động chức năng "Tạo dữ liệu" . . . . .	12
Hình 4.1	Kiến trúc tổng thể của hệ thống . . . . .	26
Hình 4.2	Kiến trúc MVCS . . . . .	28
Hình 4.3	MQTT Service . . . . .	28
Hình 4.4	Sơ đồ lớp chi tiết gói Views . . . . .	29
Hình 4.5	Sơ đồ lớp chi tiết gói Models . . . . .	29
Hình 4.6	Sơ đồ lớp chi tiết gói Controllers . . . . .	30
Hình 4.7	Sơ đồ lớp chi tiết gói Services . . . . .	30
Hình 4.8	Thiết kế màn hình chính . . . . .	32
Hình 4.9	Thiết kế màn tạo thiết bị mới . . . . .	33
Hình 4.10	Thiết kế màn tạo dữ liệu mới . . . . .	34
Hình 4.11	Thiết kế màn điều khiển kết nối Wifi Adhoc . . . . .	35
Hình 4.12	Biểu đồ lớp cho ca sử dụng "Tạo thiết bị" . . . . .	36
Hình 4.13	Biểu đồ trình tự cho ca sử dụng "Tạo thiết bị" . . . . .	36
Hình 4.14	Biểu đồ lớp cho ca sử dụng "Chỉnh sửa thiết bị qua Bluetooth" . . . . .	37
Hình 4.15	Biểu đồ trình tự cho ca sử dụng "Chỉnh sửa thiết bị qua Bluetooth" . . . . .	37
Hình 4.16	Biểu đồ lớp cho ca sử dụng "Chỉnh sửa thiết bị qua Wifi Adhoc" . . . . .	38
Hình 4.17	Biểu đồ trình tự cho ca sử dụng "Chỉnh sửa thiết bị qua Wifi Adhoc" . . . . .	38
Hình 4.18	Biểu đồ lớp cho ca sử dụng "Xóa thiết bị" . . . . .	39
Hình 4.19	Biểu đồ trình tự cho ca sử dụng "Xóa thiết bị" . . . . .	40
Hình 4.20	Biểu đồ lớp cho ca sử dụng "Tạo dữ liệu" . . . . .	41
Hình 4.21	Biểu đồ trình tự cho ca sử dụng "Tạo dữ liệu" . . . . .	42
Hình 4.22	Biểu đồ lớp cho ca sử dụng "Hiển thị dữ liệu" . . . . .	43
Hình 4.23	Biểu đồ trình tự cho ca sử dụng "Hiển thị dữ liệu" . . . . .	44
Hình 4.24	Biểu đồ lớp cho ca sử dụng "Xóa dữ liệu" . . . . .	45
Hình 4.25	Biểu đồ trình tự cho ca sử dụng "Xóa dữ liệu" . . . . .	46

Hình 4.26	Lược đồ cơ sở dữ liệu . . . . .	47
Hình 4.27	Màn hình Đăng nhập và Màn hình chính . . . . .	51
Hình 4.28	Màn hình "Devices dashboard" và Màn hình "Data dashboard" . . . . .	51
Hình 4.29	Màn hình "Device update" . . . . .	52
Hình 4.30	Màn hình "Data update" . . . . .	52
Hình 4.31	Màn hình "New Device" . . . . .	53
Hình 4.32	Màn hình "New Data" . . . . .	53
Hình 4.33	Màn hình "Scan QR" . . . . .	54
Hình 4.34	Các màn hình chức năng Wifi Adhoc . . . . .	54
Hình 4.35	Hiển thị chức năng của thiết bị EPD . . . . .	55
Hình 4.36	Hiển thị dữ liệu trên thiết bị EPD . . . . .	55
Hình 4.37	Hình ảnh của ESP32-C3 Supermini . . . . .	61
Hình 5.1	Trình tự xử lý dữ liệu ảnh . . . . .	64
Hình 5.2	Một số hiển thị dữ liệu ảnh trên thiết bị EPD . . . . .	64
Hình 5.3	Hiển thị gỡ lỗi trên thiết bị EPD . . . . .	65
Hình 5.4	Cấu trúc GATT Profile . . . . .	67
Hình 5.5	Hiển thị dữ liệu ảnh trên EPD . . . . .	70
Hình 5.6	Hiển thị dữ liệu ảnh bị lỗi trên EPD . . . . .	70
Hình 5.7	Thông tin mã nguồn từ PlatformIO . . . . .	72
Hình 5.8	Lỗi khi tải chương trình vào ESP32 . . . . .	72

## DANH MỤC BẢNG BIỂU

Bảng 2.1	So sánh 4 nhà cung cấp Electronic Paper Display . . . . .	5
Bảng 2.2	Đặc tả use case "Tạo thiết bị" . . . . .	13
Bảng 2.3	Đặc tả use case "Chỉnh sửa thiết bị qua Bluetooth" . . . . .	14
Bảng 2.4	Đặc tả use case "Chỉnh sửa thiết bị qua WiFi Adhoc" . . . . .	16
Bảng 2.5	Đặc tả use case "Xóa thiết bị" . . . . .	16
Bảng 2.6	Đặc tả use case "Tạo dữ liệu" . . . . .	17
Bảng 2.7	Đặc tả use case "Hiển thị dữ liệu" . . . . .	18
Bảng 2.8	Đặc tả use case "Xóa dữ liệu" . . . . .	19
Bảng 4.1	Thiết kế chi tiết cơ sở dữ liệu . . . . .	48
Bảng 4.2	Danh sách các thư viện và công cụ sử dụng . . . . .	49
Bảng 4.3	Chi tiết các trường form "Tạo dữ liệu" . . . . .	56
Bảng 4.4	Kết quả kiểm thử chức năng "Tạo dữ liệu" . . . . .	57
Bảng 4.5	Testing results of "Removing data" process . . . . .	57
Bảng 4.6	Chi tiết các trường form "Tạo thiết bị" . . . . .	58
Bảng 4.7	Kết quả kiểm thử chức năng "Tạo thiết bị" . . . . .	58
Bảng 4.8	Kết quả kiểm thử chức năng "Xóa thiết bị" . . . . .	58
Bảng 4.9	Bảng thông kê thông tin triển khai hệ thống . . . . .	59
Bảng 4.10	Chi tiết Server sử dụng . . . . .	60
Bảng 4.11	Thông số kỹ thuật ESP32-C3 Supermini . . . . .	61
Bảng 4.12	Lược đồ kết nối SPI . . . . .	62
Bảng 5.1	Các chế độ trên thiết bị EPD . . . . .	65
Bảng 5.2	So sánh Bluetooth Low Energy và Bluetooth Classic . . . . .	66
Bảng 5.3	Thiết kế Service cho BLE . . . . .	68
Bảng 5.4	Thiết kế Characteristic cho các Service . . . . .	68
Bảng 5.5	Mô tả thiết kế khôi dữ liệu ảnh . . . . .	69
Bảng 5.6	Tính năng của Web server trên thiết bị EPD . . . . .	71

## **DANH MỤC THUẬT NGỮ VÀ TỪ VIẾT TẮT**

<b>Thuật ngữ</b>	<b>Ý nghĩa</b>
AP	Điểm truy cập (Access Point)
API	Giao diện lập trình ứng dụng (Application Programming Interface)
BLE	Bluetooth năng lượng thấp (Bluetooth Low Energy)
EPD	Màn hình giấy điện tử (Electronic Paper Display)
GATT	Generic Attribute Profile
HTML	Ngôn ngữ đánh dấu siêu văn bản (HyperText Markup Language)
IaaS	Dịch vụ hạ tầng
IoT	Internet vạn vật (Internet of Things)

# CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

## 1.1 Đặt vấn đề

Hiện nay, thời đại công nghệ 4.0 đang ngày càng phát triển và có tác động lớn đến rất nhiều các ngành nghề, lĩnh vực khác nhau với lượng dữ liệu thông tin khổng lồ. Việc gia tăng dữ liệu này thúc đẩy các đổi mới về quản lý và hiển thị dữ liệu. Dựa vào việc tích hợp phân tích dữ liệu lớn, điện toán đám mây và IoT (Internet of Things) mà các doanh nghiệp đã có thể xử lý và trực quan hóa các tập dữ liệu phức tạp hiệu quả hơn.

Tuy nhiên, thách thức vẫn còn trong việc trình bày lượng thông tin khổng lồ này theo cách dễ tiếp cận, hấp dẫn và dễ hiểu đối với nhiều đối tượng. Con người đã sáng tạo ra rất nhiều cách để hiển thị một lượng thông tin cần thiết và trực quan hóa. Một trong những cách phổ biến nhất đó chính là sử dụng màn hình kỹ thuật số, sử dụng các hình thức photocopy hay bảng tin, v.v. Tuy nhiên, các hình thức này đều có nhược điểm riêng và chưa có giải pháp. Ví dụ như là vấn đề năng lượng hay vấn đề về tính tồn tại cho các thiết bị màn hình, vấn đề về khả năng linh hoạt thông tin, quản lý và chi phí sử dụng dành cho các hình thức sử dụng giấy. Việc chuyển đổi từ các phương pháp hiển thị truyền thống sang các giải pháp linh hoạt, tiết kiệm chi phí và tiết kiệm năng lượng hơn không chỉ là một lựa chọn mà còn là điều cần thiết để bắt kịp với bối cảnh đang phát triển nhanh chóng của thời đại kỹ thuật số.

Việc ra đời của màn hình giấy điện tử (EPD) chính là một trong những giải pháp khắc phục và tối ưu hơn bên cạnh các phương pháp truyền thống và màn hình kỹ thuật số. Những ưu điểm của EPD có thể kể đến đó chính là:

- **Tiết kiệm năng lượng:** So với việc sử dụng một lượng lớn năng lượng để hiển thị các thông tin lên trên màn hình kỹ thuật số, EPD sử dụng một lượng nhỏ và không cần cung cấp liên tục để hiển thị.
- **Linh hoạt:** Màn hình giấy điện tử có khả năng thay đổi dữ liệu hiển thị tùy theo cấu hình và yêu cầu của người dùng giống như màn hình kỹ thuật số.
- **Bất biến:** Màn hình giấy điện tử có thể hiển thị dữ liệu ngay cả khi không cung cấp năng lượng. Một trong những ưu điểm lớn của phương pháp hiển thị truyền thống.
- **Lưu trữ và quản lý:** Nhờ có các ưu điểm về tính Linh hoạt và Bất biến đã nêu trên mà EPD có thể giảm thiểu số lượng cần thiết để hiển thị thông tin. Từ đó, việc lưu trữ và quản lý trở nên tối ưu và dễ dàng hơn.

- **Chi phí:** Việc tiết kiệm năng lượng, lưu trữ và quản lý tối ưu hơn cũng giúp chi phí sử dụng trở nên ít tốn kém hơn so với các phương pháp truyền thống và kỹ thuật số.
- **Triển khai:** Hệ quả của tiết kiệm năng lượng và giao tiếp không dây là việc lắp rất đơn giản. Thiết bị chỉ cần gắn lên tường mà không cần dây nối. Tính sẵn sàng của hệ thống rất cao.

Vì vậy, EPD có thể được coi là một trong những giải pháp tận dụng tất cả những ưu điểm của các phương pháp nêu trên. Nhờ đó, màn hình giấy điện tử có thể ứng dụng trong tất cả lĩnh vực ứng dụng những phương pháp trên như là y tế, kinh doanh, công nghiệp và giáo dục.

## 1.2 Mục tiêu và phạm vi đề tài

Ứng dụng của giấy điện tử đã và đang được sử dụng trong nhiều hệ thống khác nhau, từ giao thông công cộng, hậu cần đến giáo dục và kinh doanh. Tuy nhiên, đây đa số là những hệ thống lớn yêu cầu xử lý nhiều nhiệm vụ chuyên biệt và yêu cầu hệ thống hiển thị phức tạp và tốn kém. Nói cách khác, hệ thống EPD được sử dụng trong các doanh nghiệp này được tối ưu hóa cao cho nhu cầu cụ thể của từng doanh nghiệp, khiến chi phí trở nên đắt đỏ và không phù hợp với các hệ thống nhỏ hơn. Bên cạnh đó, đối với các hệ thống kinh doanh nhỏ và vừa đã xuất hiện ở trên thị trường (ví dụ ứng dụng của giấy điện tử trong các cửa hàng CellphoneS tại Việt Nam), sự tiện ích và tính khả dụng trong nhiều bối cảnh chưa có.

Dựa trên những hạn chế này, đồ án nhằm mục đích phát triển hệ thống thiết bị EPD phiên tập trung nhiều hơn vào các doanh nghiệp không quá lớn và các trường hợp sử dụng. Màn hình giấy điện tử đã được thiết kế và xây dựng trước đó với phiên bản đầu tiên. Tuy nhiên, việc sử dụng còn nhiều hạn chế và quá trình xây dựng còn tồn tại một số vấn đề, do vậy đồ án này tập trung giải quyết các vấn đề đó cũng như xây dựng mở rộng thêm tính năng mới nhằm đến các mục tiêu chính sau: (i) Cung cấp cách hiển thị dữ liệu bằng cách sử dụng công nghệ hiển thị giấy điện tử ở các doanh nghiệp vừa và nhỏ; (ii) Cung cấp giao diện người dùng quản lý trên nhiều nền tảng và hệ thống back-end phù hợp với hầu hết các nhu cầu của mọi lĩnh vực, trước hết tập trung vào bán lẻ, văn phòng, bệnh viện, trường học và triển lãm; (iii) Tối ưu hóa các thiết bị EPD để đạt được hiệu suất tốt hơn và chất lượng cao hơn trong nhiều môi trường sử dụng khác nhau.

## 1.3 Định hướng giải pháp

Để đạt được mục tiêu đặt ra ở Phần 1.2, đồ án sẽ tập trung phát triển hệ thống theo mô hình client-server kết hợp với microservices từ hệ thống đã có và nâng cấp, mở rộng thêm các tính năng mới phù hợp với các mục đích, tình huống sử dụng.

Về phía frontend, đồ án sử dụng NextJS để phát triển giao diện người dùng trên ứng dụng web và React Native để xây dựng giao diện người dùng trên thiết bị di động (Mobile application) cụ thể là nền tảng Android. Giao diện giúp người dùng dễ dàng sử dụng và giao tiếp với các thiết bị EPD chạy trên module phát triển Supermini ESP32-C3. Trong đó, ứng dụng di động sẽ được cung cấp thêm tính năng giao tiếp với các thiết bị bằng công nghệ Bluetooth Low Energy (BLE).

Về phía backend, đồ án sử dụng ExpressJS và MongoDB để xây dựng máy chủ API chứa dữ liệu người dùng và Eclipse Mosquitto với tư cách là MQTT Broker để giao tiếp với các thiết bị EPD thông qua giao thức MQTT.

Về thiết bị EPD, đồ án sử dụng bo mạch Supermini ESP32-C3 để nhận và xử lý các yêu cầu hiển thị tới EPD cùng với màn hình giấy điện tử sử dụng là WeAct-Epaper 2.9 Inch. Thiết bị cũng sẽ được cấu hình thêm khả năng trao đổi dữ liệu qua BLE và tạo ra điểm truy cập WiFi (WiFi AP) để có thể cấu hình lại kết nối WiFi của thiết bị EPD.

### 1.4 **Bố cục đồ án**

Phần còn lại của báo cáo đồ án tốt nghiệp này được tổ chức như sau:

Chương 2 trình bày quá trình khảo sát và phân tích yêu cầu của đồ án, cung cấp tổng quan về chức năng và trình bày chi tiết một số tính năng nổi bật của hệ thống hiển thị EPD.

Dựa trên các phân tích trong Chương 2, Chương 3 sẽ tập trung vào các công nghệ chính để phát triển hệ thống. Hệ thống sẽ bao gồm ba phần chính: (i) Trang web quản lý thông tin và thiết bị hiển thị; (ii) Ứng dụng trên thiết bị di động; (iii) Server Node.js và MQTT Broker để xử lý giao tiếp giữa các thiết bị hiển thị và server, cũng như xử lý các yêu cầu từ trang web; (iv) Thiết bị EPD để nhận và hiển thị thông tin.

Chương 4 giới thiệu về thiết kế kiến trúc của từng thành phần, thiết kế cơ sở dữ liệu, quy trình phát triển ứng dụng và thử nghiệm hệ thống.

Chương 5 trình bày về những đóng góp và giải pháp nổi bật giúp hệ thống giải quyết các vấn đề hiện tại như quản lý thiết bị từ xa và một số khó khăn gặp phải khi phát triển hệ thống.

Chương 6 trình bày những đóng góp chính của đồ án và định hướng phát triển trong tương lai của hệ thống.

## CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

### 2.1 Khảo sát hiện trạng

#### 2.1.1 Các hệ thống trên thị trường

Hiện nay, thị trường có rất nhiều nhà sản xuất màn hình giấy điện tử tận dụng công nghệ EPD để cung cấp giải pháp cho nhiều doanh nghiệp khác nhau. Trong đó, em tập trung vào việc khảo sát bốn nhà cung cấp màn hình giấy điện tử: **E Ink**, **InkCase**, **Pervasive Displays** và **Zkong**. Việc so sánh, phân tích dựa vào các tiêu chí như hệ thống quản lý, chi phí, khả năng kết nối, ứng dụng tại Việt Nam, v.v. **E Ink** là một trong những công ty dẫn đầu trong công nghệ hiển thị giấy điện tử (e-paper), được sáng lập vào năm 1997 tách ra từ MIT Media Lab nhằm thương mại hóa công nghệ EPD và mực điện tử [1]. Họ cung cấp nhiều loại module hiển thị e-paper phù hợp cho các ứng dụng đa dạng. Các sản phẩm của họ được sử dụng trong nhiều lĩnh vực, bao gồm đọc và viết, giáo dục, kinh doanh và văn phòng, thiết bị di động và phụ kiện, bán lẻ, logistics và nhà máy, chăm sóc sức khỏe, giao thông vận tải, ô tô và thiết kế sáng tạo [2]. Họ cũng giới thiệu các ứng dụng theo yêu cầu của khách hàng và cung cấp các giải pháp tối ưu và tích hợp hệ thống cho các doanh nghiệp.

**InkCase** là một dòng sản phẩm từ công ty thiết kế điện tử tiêu dùng Gajah có trụ sở tại Singapore. Họ chuyên cung cấp các loại ốp bảo vệ cho điện thoại thông minh với các chức năng bổ sung và cũng tham gia vào các giải pháp hiển thị sáng tạo khác nhau, bao gồm hiển thị thông tin thông minh, chăm sóc sức khỏe và theo dõi [3].

**Pervasive Displays** là nhà sản xuất hàng đầu trong ngành hiển thị giấy điện tử, chuyên thiết kế, sản xuất và tiếp thị các màn hình e-paper. Họ sản xuất hơn 2 triệu đơn vị mỗi tuần, ổn định về tài chính và là một phần của công ty hiển thị lớn nhất thế giới [4]. Sản phẩm và dịch vụ của họ được tùy chỉnh cho các ứng dụng công nghiệp, với trọng tâm là thiết kế và sản xuất các màn hình tiêu thụ năng lượng cực thấp, và thường được đặt hàng với số lượng lớn bởi các hãng công nghiệp lớn khác.

**Zkong** là một công ty công nghệ cao được thành lập vào năm 2006 tại Chiết Giang, Trung Quốc. Công ty chủ yếu tham gia vào nghiên cứu, phát triển và kinh doanh phần cứng thông minh không dây và công nghệ IoT bán lẻ mới, cung cấp các dịch vụ đa dạng cho các lĩnh vực bán lẻ mới, siêu thị và cửa hàng bách hóa, làm đẹp và thời trang, điện tử tiêu dùng, chuỗi thực phẩm tươi sống, y tế, kho vận, và ngành công nghiệp ẩm thực, đồng thời không ngừng khám phá thêm nhiều lĩnh vực ứng dụng mới [5]. Các sản phẩm của công ty được bán rộng rãi tại nhiều quốc

## CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

gia trên toàn cầu, bao gồm cả Việt Nam.

Với những đánh giá nêu trên, có thể tóm tắt đặc điểm của 4 nhà sản xuất màn hình giấy điện tử trong bảng 2.1 dưới đây:

Tính năng	E Ink	InkCase	Pervasive Displays	Zkong
Chi phí	Cao	Cao	Cao	Trung bình
Lĩnh vực ứng dụng	Kinh doanh Giáo dục Y tế Logistics Vận tải Sản xuất phương tiện	Vận tải Kinh doanh	Logistic IoT Y tế Giáo dục Kinh doanh	Kinh doanh IoT Y tế Giáo dục Vận tải Sản xuất phương tiện
Hệ thống quản lý	✓	✓	✓	✓
Chỉnh sửa theo yêu cầu	✓	✗	✓	✓
Khả năng kết nối	Bluetooth Internet	Bluetooth Internet	Không công bố	Bluetooth Internet
Ứng dụng tại Việt Nam	Chưa có	Chưa có	Chưa có	Đã có

**Bảng 2.1:** So sánh 4 nhà cung cấp EPD

### 2.1.2 Hệ thống đã được xây dựng

Dựa vào những kết quả trước đó được xây dựng bởi Vũ Lê Nhật Minh trong đồ án "Xây dựng bảng thông tin sử dụng giấy điện tử ePaperboard" - phiên bản 1 của hệ thống, em có một số đánh giá như sau:

Về ưu điểm:

- Thiết bị có khả năng hiển thị Tiếng Việt trên màn hình so với các sản phẩm trên thị trường. Đây là một trong những cơ hội để đưa việc ứng dụng EPD trở nên khả thi tại Việt Nam.
- Thiết bị có tốc độ xử lý hiển thị nhanh hơn so với các thư viện có sẵn được xây dựng bởi các nhà sản xuất.
- Giao diện web ưa nhìn và thân thiện.
- Hệ thống hoạt động hiệu quả trong trường hợp khả thi.

Về nhược điểm:

- Thiết kế của Database sai lệch so với thiết kế trên EPD. Điều này dẫn đến việc sửa lỗi và mở rộng hệ thống gặp khó khăn.
- Thời gian cập nhật của thiết bị EPD khá lâu sau khi nhận được yêu cầu từ

server.

- Phiên bản trước đó chưa thực sự thân thiện đối với người dùng. Việc tạo mới phải dùng cáp USB là trở ngại lớn với việc tiếp cận hệ thống.
- Nếu như thiết bị EPD không thể kết nối WiFi và người dùng không có cáp USB thì thiết bị trở nên vô dụng, không thể thực hiện bất kỳ hành động nào.
- Một số bản mẫu của hệ thống còn chưa đủ ưa nhìn đối với một sản phẩm thương mại.

Sau khi khảo sát bốn hệ thống trên và các trường hợp sử dụng thực tế khác cũng như tham khảo từ kết quả của hệ thống phiên bản 1, em đã rút ra một số mục tiêu xây dựng hệ thống thiết bị hiển thị giấy điện tử (EPD), bao gồm thiết kế và xây dựng các thiết bị EPD, cũng như bao gồm một số tính năng cần thiết mới và cải tiến phù hợp với nhiều trường hợp sử dụng khác nhau.

Đầu tiên, tận dụng công nghệ giấy điện tử trong việc hiển thị thông tin. Với đặc điểm độc đáo của màn hình giấy điện tử, thiết bị EPD cần có khả năng hiển thị dữ liệu trong một khoảng thời gian dài mà không cần làm mới thường xuyên. Thiết bị cũng cần tiết kiệm năng lượng và có thể hoạt động trong thời gian hợp lý mà không cần sạc lại.

Thứ hai, quản lý các thiết bị EPD trong một hệ thống thông qua một trung tâm điều khiển, giao tiếp với các thiết bị EPD và máy chủ backend bằng giao thức MQTT, chuẩn kết nối Bluetooth BLE và các kết nối không dây khác chịu trách nhiệm nhận dữ liệu và hiển thị trên các bảng EPD.

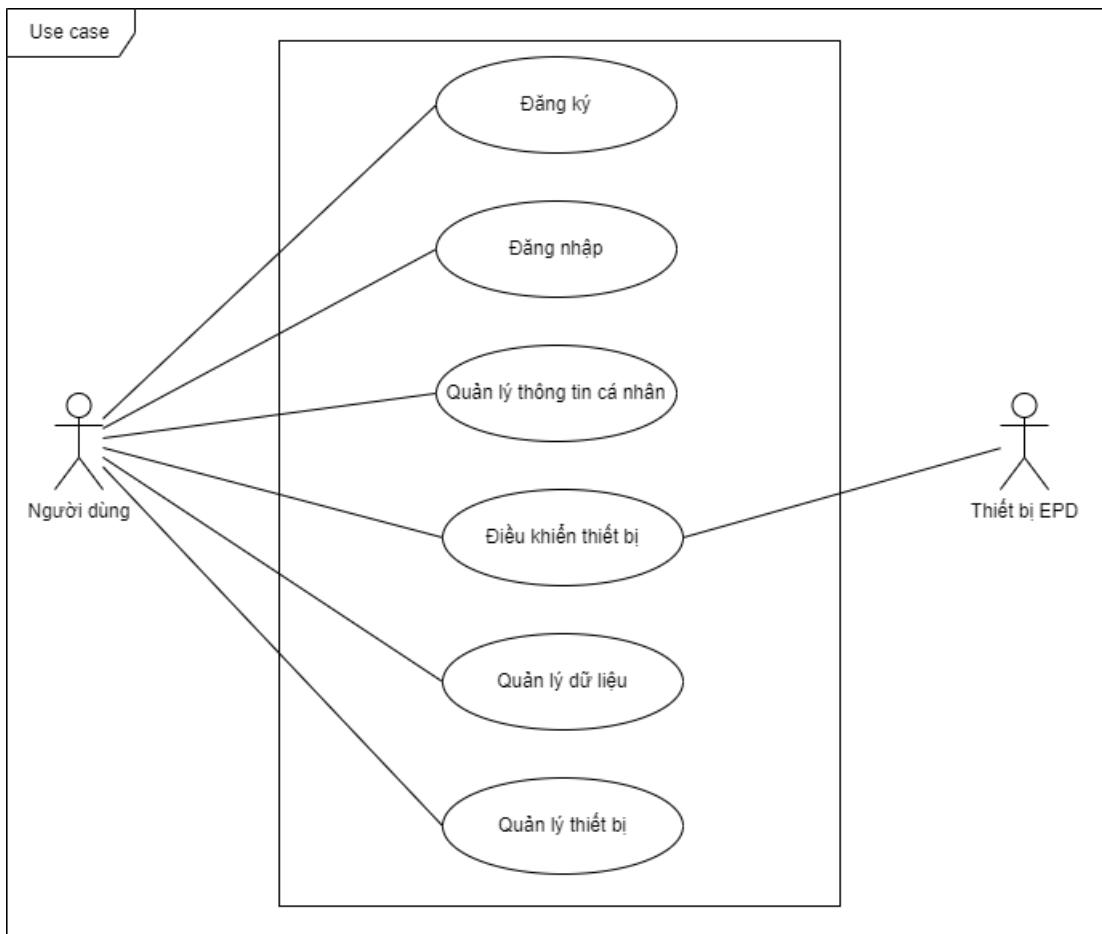
Cuối cùng, triển khai hệ thống thiết bị EPD trong một số trường hợp sử dụng ban đầu để kiểm tra hiệu suất và hiệu quả của hệ thống trong việc giải quyết các vấn đề thực tế. Các trường hợp sử dụng ban đầu bao gồm thẻ tên cho học sinh, nhân viên và khách hàng; thẻ giá trong các siêu thị mini và dịch vụ logistics; hiển thị dữ liệu ảnh; và biển báo phòng và các biển báo khác.

### 2.2 Tổng quan chức năng

#### 2.2.1 Biểu đồ use case tổng quát

Hình 2.1 là mô tả biểu đồ use case tổng quan của hệ thống

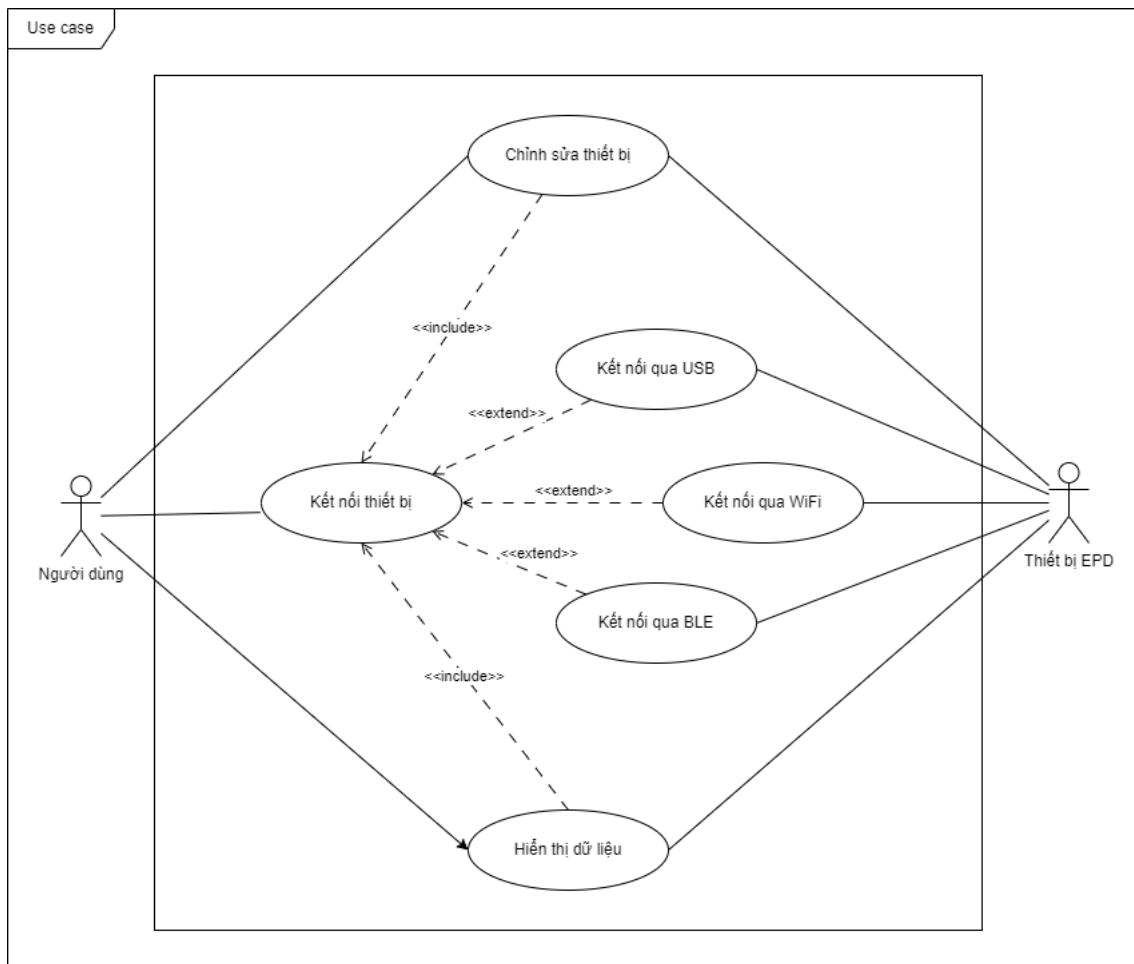
## CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU



**Hình 2.1:** Biểu đồ use case tổng quan

Hệ thống bao gồm hai tác nhân chính: Người dùng (User) và Thiết bị EPD (EPD devices). Người dùng (User) có các chức năng: Đăng ký, Đăng nhập, Quản lý thông tin cá nhân, Điều khiển thiết bị, Quản lý dữ liệu và Quản lý thiết bị. Trong khi đó, EPD đóng vai trò như một người dùng cuối, có chức năng nhận thông tin và hiển thị dữ liệu. Hai tác nhân tương tác với hệ thống qua giao thức MQTT hoặc BLE.

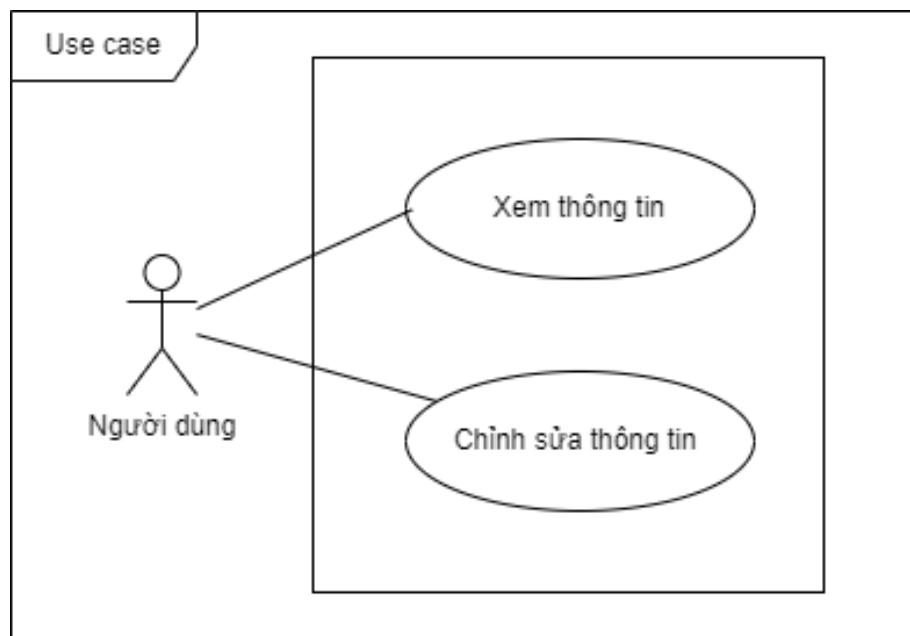
### 2.2.2 Biểu đồ use case phân rã "Điều khiển thiết bị"



**Hình 2.2:** Biểu đồ use case phân rã "Điều khiển thiết bị"

Hình 2.2 là mô tả phân rã use case "Điều khiển thiết bị". Để điều khiển thiết bị, trước tiên người dùng phải kết nối với thiết bị. Trong đó, việc kết nối với thiết bị có 3 cách đó là WiFi, BLE và USB. Trong phạm vi đồ án sẽ chủ yếu trình bày về tính năng sử dụng kết nối qua BLE. Sau đó, người dùng có thể thực hiện chỉnh sửa cấu hình và yêu cầu hiển thị dữ liệu. Thiết bị EPD sẽ nhận được yêu cầu của người dùng và thực hiện.

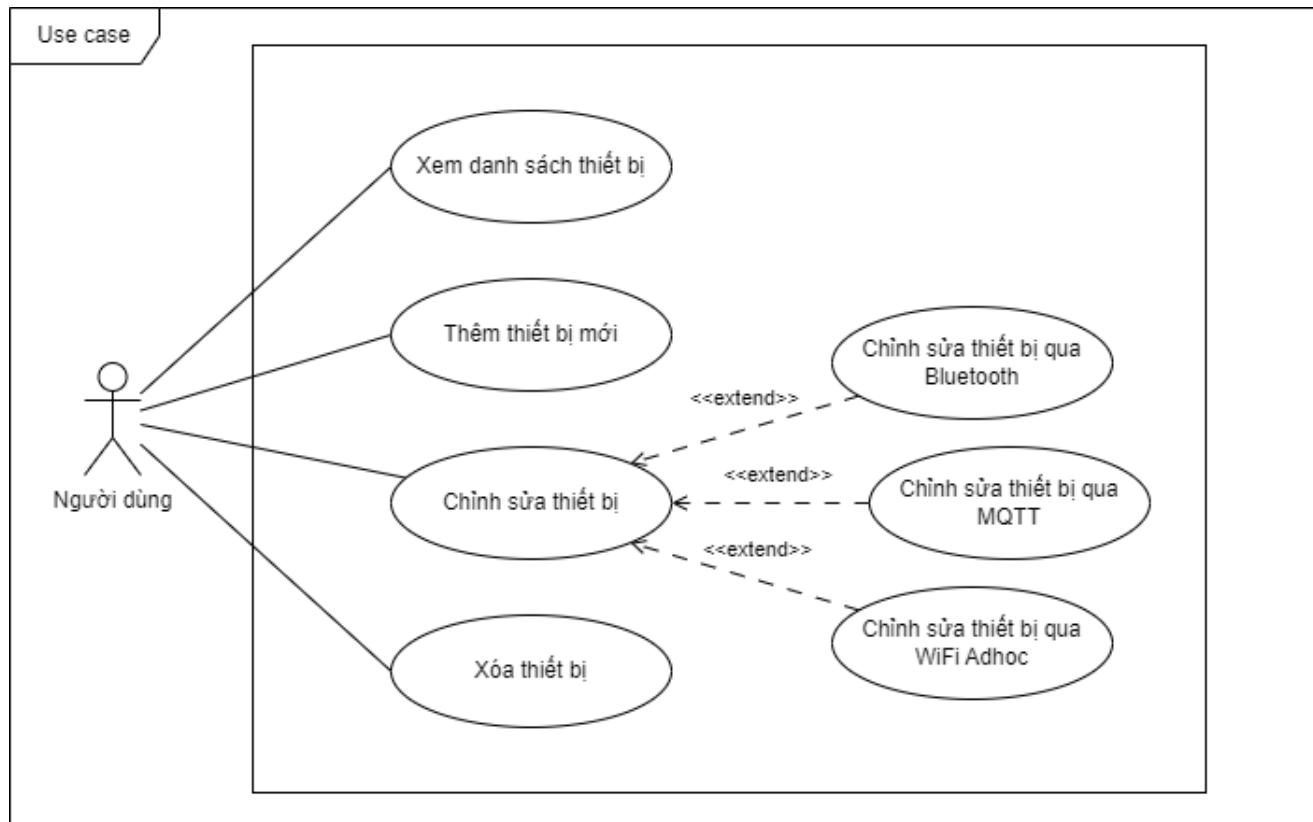
### 2.2.3 Biểu đồ use case phân rã "Quản lý thông tin cá nhân"



**Hình 2.3:** Biểu đồ use case phân rã "Quản lý thông tin cá nhân"

Hình 2.3 là mô tả phân rã use case "Quản lý thông tin cá nhân". Người dùng có thể xem hoặc thay đổi thông tin cá nhân của mình.

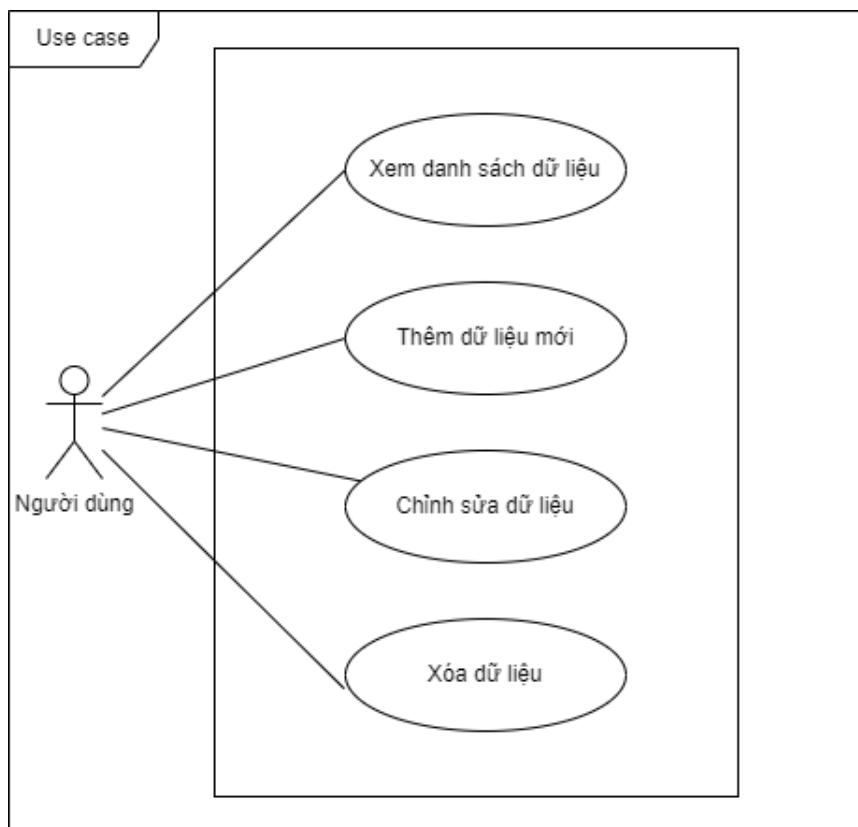
### 2.2.4 Biểu đồ use case phân rã "Quản lý thiết bị"



**Hình 2.4:** Biểu đồ use case phân rã "Quản lý thiết bị"

Hình 2.4 là mô tả phân rã use case "Quản lý thiết bị". Người dùng có thể xem danh sách thiết bị, tạo thiết bị mới, chỉnh sửa thiết bị và xóa thiết bị. Trong đó, việc chỉnh sửa thiết bị có thể thực hiện bằng cách sử dụng Bluetooth, MQTT và WiFi Adhoc.

### 2.2.5 Biểu đồ use case phân rã "Quản lý dữ liệu"



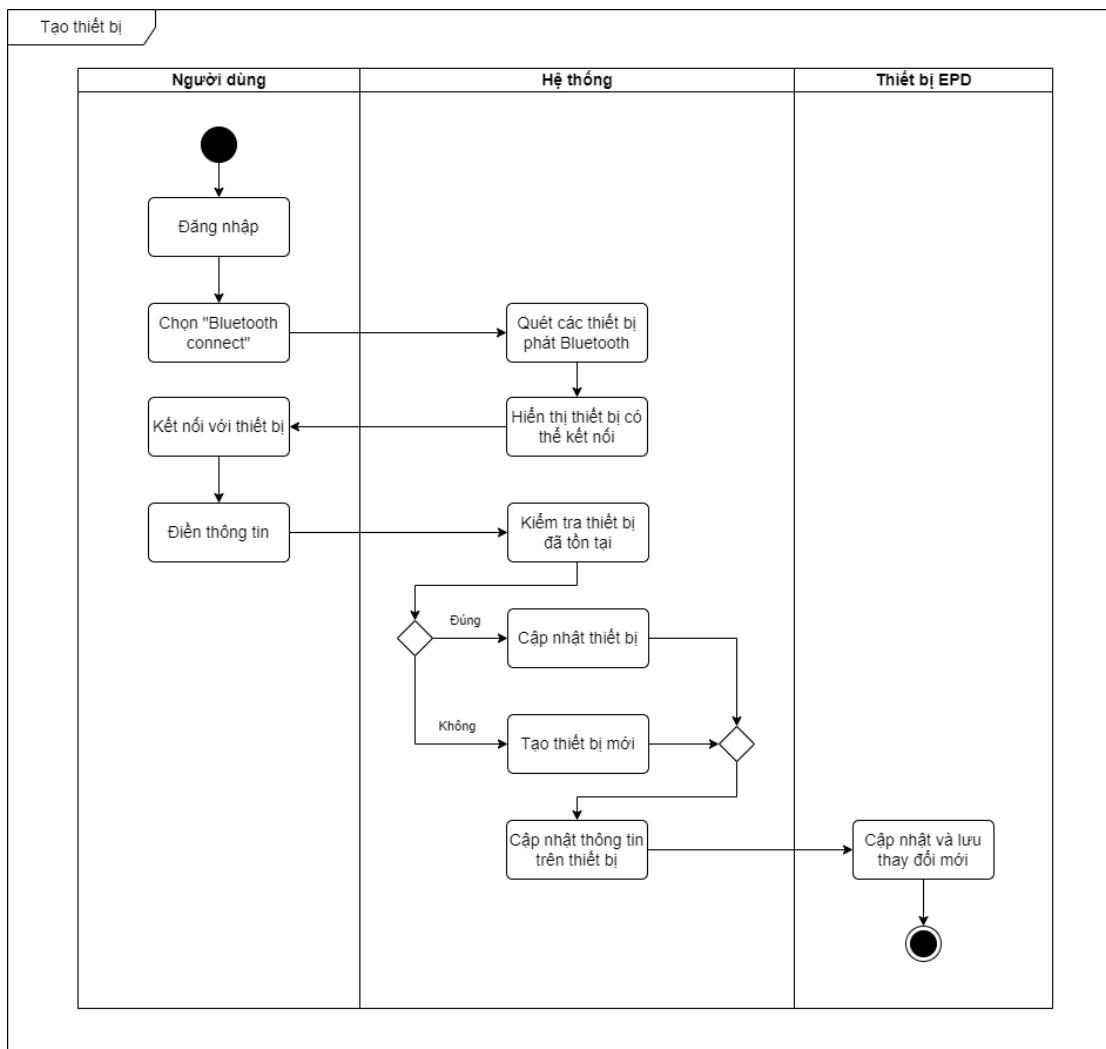
**Hình 2.5:** Biểu đồ use case phân rã "Quản lý dữ liệu"

Hình 2.5 là mô tả phân rã use case "Quản lý dữ liệu". Người dùng có thể xem danh sách dữ liệu, tạo dữ liệu mới, chỉnh sửa dữ liệu và xóa dữ liệu.

## 2.3 Quy trình nghiệp vụ

Hệ thống bao gồm nhiều quy trình nghiệp vụ, trong đó đáng chú ý là quy trình thêm dữ liệu và hiển thị nó trên các thiết bị EPD và quy trình thêm các thiết bị EPD vào hệ thống. Trong phiên bản 1, các quy trình đã được trình bày cách hệ thống giao tiếp với thiết bị qua cổng USB và MQTT một cách cụ thể. Do vậy, ở phiên bản 2 của hệ thống, báo cáo đồ án này sẽ trình bày cách hệ thống giao tiếp với thiết bị qua giao tiếp Bluetooth BLE.

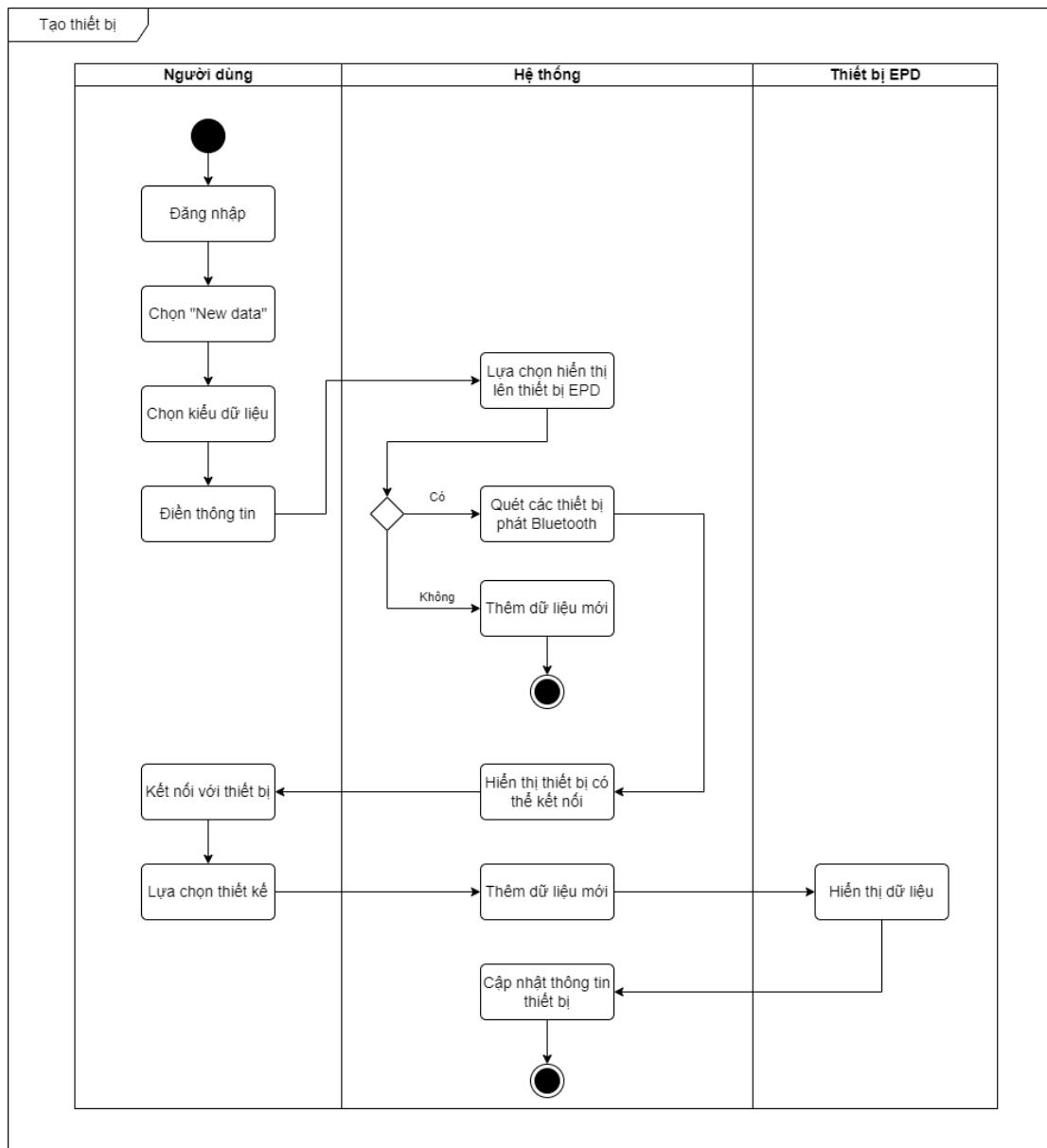
### 2.3.1 Biểu đồ luồng hoạt động chức năng "Tạo thiết bị"



**Hình 2.6:** Biểu đồ luồng hoạt động chức năng "Tạo thiết bị"

Hình 2.6 biểu diễn luồng hoạt động khi Người dùng tạo và đăng ký một thiết bị mới vào hệ thống. Quy trình này yêu cầu người dùng kết nối thiết bị EPD với thiết bị di động bằng Bluetooth, chọn từ danh sách các thiết bị có thể tìm thấy, sau đó điền thông tin cần thiết trước khi gửi về hệ thống. Hệ thống sẽ kiểm tra xem thiết bị đã tồn tại hay chưa và thực hiện cập nhật hoặc tạo mới. Sau đó, hệ thống gửi yêu cầu cập nhật thông tin cho thiết bị EPD.

### 2.3.2 Biểu đồ luồng hoạt động chức năng "Tạo dữ liệu"



**Hình 2.7:** Biểu đồ luồng hoạt động chức năng "Tạo dữ liệu"

Hình 2.7 mô tả luồng hoạt động Người dùng tạo dữ liệu mới. Trước tiên, người dùng chọn loại dữ liệu họ muốn hiển thị. Sau đó, họ thêm thông tin chi tiết tương ứng dựa trên loại dữ liệu đã chọn. Người dùng cũng có thể quyết định hiển thị trên thiết bị EPD hay không. Nếu họ chọn hiển thị, hệ thống sẽ cung cấp danh sách các thiết bị đang phát Bluetooth có thể tìm thấy và thiết bị đang kết nối với hệ thống để người dùng chọn. Người dùng sau đó sẽ cung cấp thêm thông tin về cách hiển thị dữ liệu trên thiết bị đã chọn. Khi tất cả thông tin cần thiết đã được cung cấp, hệ thống sẽ thêm dữ liệu mới và gửi các thông điệp qua Bluetooth. Thiết bị được chọn sẽ nhận và hiển thị dữ liệu.

## 2.4 Đặc tả chức năng

### 2.4.1 Đặc tả use case "Tạo thiết bị"

<b>Tên use case</b>	Tạo thiết bị		
<b>Tác nhân</b>	Người dùng, Hệ thống, Thiết bị EPD		
<b>Tiền điều kiện</b>	Người dùng đã đăng nhập vào hệ thống và thiết bị EPD đang bật chế độ BLE		
<b>Luồng sự kiện chính</b>	<b>STT</b>	<b>Thực hiện</b>	<b>Hành động</b>
	1	Người dùng	Chọn chức năng "Bluetooth connect"
	2	Hệ thống	Quét các thiết bị đang phát Bluetooth và hiển thị
	3	Người dùng	Chọn và kết nối với thiết bị EPD
	4	Hệ thống	Hiển thị form điền thông tin thiết bị EPD
	5	Người dùng	Điền thông tin thiết bị và xác nhận
	6	Hệ thống	Kiểm tra thiết bị đã tồn tại trên hệ thống hay chưa
	6.1	Hệ thống	Nếu thiết bị đã tồn tại, thực hiện cập nhật thông tin thiết bị
	6.2	Hệ thống	Nếu thiết bị chưa tồn tại, tạo thiết bị mới
	7	Hệ thống	Gửi thông tin mới tới thiết bị EPD
<b>Luồng sự kiện phát sinh</b>	<b>STT</b>	<b>Thực hiện</b>	<b>Hành động</b>
	3a	Hệ thống	Thông báo lỗi: Không thể kết nối tới thiết bị
<b>Hậu điều kiện</b>	<b>STT</b>	<b>Thực hiện</b>	<b>Hành động</b>
	5a	Hệ thống	Thông báo lỗi: Tất cả các trường không được để trống và mật khẩu wifi có độ dài ít nhất 8 ký tự
<b>Hậu điều kiện</b>	Hệ thống tạo mới thông tin thiết bị vào cơ sở dữ liệu		

**Bảng 2.2:** Đặc tả use case "Tạo thiết bị"

### 2.4.2 Đặc tả use case "Chỉnh sửa thiết bị qua Bluetooth"

<b>Tên use case</b>	Chỉnh sửa thiết bị qua Bluetooth		
<b>Tác nhân</b>	Người dùng, Hệ thống, Thiết bị EPD		
<b>Tiền điều kiện</b>	Người dùng đã đăng nhập vào hệ thống và thiết bị EPD đang bật chế độ BLE		
<b>Luồng sự kiện chính</b>	<b>STT</b>	<b>Thực hiện</b>	<b>Hành động</b>
	1	Người dùng	Chọn "Devices dashboard"
	2	Hệ thống	Hiển thị thông tin các thiết bị đã tồn tại trên hệ thống
	3	Người dùng	Chọn và kết nối với thiết bị EPD
	4	Hệ thống	Hiển thị form điền thông tin thiết bị EPD
	5	Người dùng	Điền thông tin thiết bị và xác nhận
	6	Hệ thống	Cập nhật thông tin mới của thiết bị trên hệ thống
	7	Hệ thống	Gửi thông tin mới tới thiết bị EPD
	8	Thiết bị EPD	Thiết bị thực hiện cập nhật thông tin mới
	9	Hệ thống	Thông báo thiết bị đã được cập nhật thành công
<b>Luồng sự kiện phát sinh</b>	<b>STT</b>	<b>Thực hiện</b>	<b>Hành động</b>
	3a	Hệ thống	Thông báo lỗi: Không thể kết nối tới thiết bị
	5a	Hệ thống	Thông báo lỗi: Tất cả các trường không được để trống và mật khẩu wifi có độ dài ít nhất 8 ký tự
<b>Hậu điều kiện</b>	Hệ thống cập nhật thông tin thiết bị vào cơ sở dữ liệu		

Bảng 2.3: Đặc tả use case "Chỉnh sửa thiết bị qua Bluetooth"

### 2.4.3 Đặc tả use case "Chỉnh sửa thiết bị qua WiFi Adhoc"

<b>Tên use case</b>	Chỉnh sửa thiết bị qua WiFi Adhoc
<b>Tác nhân</b>	Người dùng, Hệ thống, Thiết bị EPD
<b>Tiền điều kiện</b>	Người dùng đã đăng nhập vào hệ thống và thiết bị EPD đang bật chế độ WiFi Adhoc

## CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

	<b>STT</b>	<b>Thực hiện</b>	<b>Hành động</b>
<b>Luồng sự kiện chính</b>	1	Người dùng	Chọn chức năng "Scan"
	2	Người dùng	Chọn bắt đầu và quét mã QR code hiển thị trên thiết bị EPD
	3	Hệ thống	Kết nối tới WiFi AP của thiết bị EPD
	4	Người dùng	Chọn chức năng "Scan Wifi" trên giao diện hệ thống
	5	Hệ thống	Gửi yêu cầu lấy thông tin các mạng WiFi đến Thiết bị EPD
	6	Thiết bị EPD	Thực hiện quét các mạng WiFi và gửi thông tin về hệ thống
	7	Hệ thống	Hiển thị các thông tin mạng WiFi do thiết bị EPD tìm được
	8	Người dùng	Chọn mạng WiFi cần thiết bị EPD kết nối và điền các thông tin thiết bị EPD
	9	Người dùng	Bấm xác nhận
	10	Hệ thống	Gửi yêu cầu cập nhật thông tin mới đến Thiết bị EPD
	11	Thiết bị EPD	Thiết bị thực hiện cập nhật thông tin mới
	12	Người dùng	Chọn chức năng "Reset"
	13	Hệ thống	Gửi yêu cầu khởi động lại đến Thiết bị EPD
	14	Thiết bị EPD	Thiết bị thực hiện khởi động lại
	15	Thiết bị EPD	Kết nối vào mạng WiFi và gửi thông tin mới lên hệ thống
	16	Hệ thống	Thực hiện cập nhật thông tin mới của thiết bị trên hệ thống
	17	Hệ thống	Thông báo thiết bị đã được cập nhật thành công
<b>Luồng sự kiện phát sinh</b>	<b>STT</b>	<b>Thực hiện</b>	<b>Hành động</b>
	3a	Hệ thống	Thông báo lỗi: Không thể kết nối tới WiFi của thiết bị

	15a	Hệ thống	Thông tin sẽ không được gửi nếu không kết nối được vào mạng WiFi
<b>Hậu điều kiện</b>	Hệ thống cập nhật thông tin thiết bị vào cơ sở dữ liệu		

**Bảng 2.4:** Đặc tả use case "Chỉnh sửa thiết bị qua WiFi Adhoc"

#### 2.4.4 Đặc tả use case "Xóa thiết bị"

<b>Tên use case</b>	Xóa thiết bị		
<b>Tác nhân</b>	Người dùng, Hệ thống, Thiết bị EPD		
<b>Tiền điều kiện</b>	Người dùng đã đăng nhập vào hệ thống và thiết bị đã tồn tại trên hệ thống		
<b>Luồng sự kiện chính</b>	<b>STT</b>	<b>Thực hiện</b>	<b>Hành động</b>
	1	Người dùng	Chọn chức năng "Device dashboard"
	2	Hệ thống	Hiển thị thông tin các thiết bị đã tồn tại trên hệ thống
	3	Người dùng	Chọn thiết bị EPD
	4	Hệ thống	Hiển thị thông tin thiết bị EPD
	5	Người dùng	Chọn "Delete"
	6	Hệ thống	Hiển thị cảnh báo xóa thiết bị
	7	Người dùng	Xác nhận xóa thiết bị EPD
	8	Hệ thống	Xóa thiết bị khỏi hệ thống
	8.1	Hệ thống	Nếu thiết bị đang hiển thị dữ liệu, gửi thông điệp xóa dữ liệu qua MQTT
<b>Luồng sự kiện phát sinh</b>	8.2	Thiết bị EPD	Xóa hiển thị dữ liệu trên màn hình
	8.3	Hệ thống	Cập nhật lại trạng thái dữ liệu
<b>Hậu điều kiện</b>	Không		

**Bảng 2.5:** Đặc tả use case "Xóa thiết bị"

#### 2.4.5 Đặc tả use case "Tạo dữ liệu"

## CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

<b>Tên use case</b>	Tạo dữ liệu		
<b>Tác nhân</b>	Người dùng, Hệ thống, Thiết bị EPD		
<b>Tiền điều kiện</b>	Người dùng đã đăng nhập vào hệ thống và thiết bị EPD đang bật chế độ BLE hoặc kết nối tới hệ thống		
<b>Luồng sự kiện chính</b>	STT	Thực hiện	Hành động
	1	Người dùng	Chọn chức năng "New data"
	2	Hệ thống	Hiển thị các loại dữ liệu
	3	Người dùng	Chọn loại dữ liệu
	4	Hệ thống	Hiển thị form điền thông tin dữ liệu
	5	Người dùng	Điền thông tin dữ liệu và xác nhận
	5.1	Hệ thống	Nếu người dùng chọn hiển thị dữ liệu, hiển thị giao diện chọn thiết bị và thiết kế giao diện
	5.2	Người dùng	Chọn thiết bị và thiết kế giao diện
	6	Hệ thống	Gửi dữ liệu mới lên hệ thống
	6.1	Hệ thống	Nếu người dùng chọn hiển thị dữ liệu, gửi thông tin dữ liệu mới tới thiết bị EPD
<b>Luồng sự kiện phát sinh</b>	6.2	Thiết bị EPD	Thiết bị EPD thực hiện hiển thị dữ liệu mới
	6.3	Thiết bị EPD	Gửi trạng thái dữ liệu đến hệ thống
<b>Hậu điều kiện</b>	7	Hệ thống	Thông báo dữ liệu đã được tạo thành công
<b>Luồng sự kiện phát sinh</b>	STT	Thực hiện	Hành động
	5a	Hệ thống	Thông báo lỗi: Tất cả các trường không được để trống và mật khẩu wifi có độ dài ít nhất 8 ký tự
<b>Hậu điều kiện</b>	Hệ thống tạo mới thông tin thiết bị vào cơ sở dữ liệu		

**Bảng 2.6:** ĐẶC TẢ USE CASE "TẠO DỮ LIỆU"

### 2.4.6 Đặc tả use case "Hiển thị dữ liệu"

<b>Tên use case</b>	Hiển thị dữ liệu
<b>Tác nhân</b>	Người dùng, Hệ thống, Thiết bị EPD

<b>Tiền điều kiện</b>	Người dùng đã đăng nhập vào hệ thống và thiết bị EPD đang bật chế độ BLE hoặc kết nối tới hệ thống		
<b>Luồng sự kiện chính</b>	<b>STT</b>	<b>Thực hiện</b>	<b>Hành động</b>
	1	Người dùng	Chọn chức năng "Data dashboard"
	2	Hệ thống	Hiển thị danh sách dữ liệu
	3	Người dùng	Chọn dữ liệu cần hiển thị trong danh sách
	4	Hệ thống	Hiển thị thông tin dữ liệu
	5	Người dùng	Chọn "Display on EPD device"
	6	Hệ thống	Hiển thị giao diện chọn thiết bị và thiết kế giao diện
	7	Người dùng	Chọn thiết bị và thiết kế giao diện
	8	Hệ thống	Cập nhật trạng thái dữ liệu và thiết bị trên hệ thống
	9	Hệ thống	Gửi thông tin dữ liệu tới thiết bị EPD
	10	Thiết bị EPD	Thiết bị EPD thực hiện hiển thị dữ liệu
<b>Luồng sự kiện phát sinh</b>	<b>STT</b>	<b>Thực hiện</b>	<b>Hành động</b>
	5a	Hệ thống	Thông báo lỗi: Tất cả các trường không được để trống và mật khẩu wifi có độ dài ít nhất 8 ký tự
<b>Hậu điều kiện</b>	Hệ thống tạo mới thông tin thiết bị vào cơ sở dữ liệu		

**Bảng 2.7:** Đặc tả use case "Hiển thị dữ liệu"

#### 2.4.7 Đặc tả use case "Xóa dữ liệu"

<b>Tên use case</b>	Xóa dữ liệu		
<b>Tác nhân</b>	Người dùng, Hệ thống, Thiết bị EPD		
<b>Tiền điều kiện</b>	Người dùng đã đăng nhập vào hệ thống và dữ liệu đã tồn tại trên hệ thống		
<b>Luồng sự kiện chính</b>	<b>STT</b>	<b>Thực hiện</b>	<b>Hành động</b>
	1	Người dùng	Chọn chức năng "Data dashboard"
	2	Hệ thống	Hiển thị danh sách dữ liệu trên hệ thống

	3	Người dùng	Chọn dữ liệu cần xóa trong danh sách
	4	Hệ thống	Hiển thị thông tin dữ liệu
	5	Người dùng	Chọn "Delete"
	6	Hệ thống	Hiển thị cảnh báo xóa dữ liệu
	7	Người dùng	Xác nhận xóa dữ liệu
	8	Hệ thống	Xóa dữ liệu khỏi hệ thống
<b>Luồng sự kiện phát sinh</b>	8.1	Hệ thống	Nếu có thiết bị đang hiển thị dữ liệu, gửi thông điệp xóa dữ liệu qua MQTT
	8.2	Thiết bị EPD	Xóa hiển thị dữ liệu trên màn hình
	8.3	Hệ thống	Cập nhật lại trạng thái thiết bị EPD
	9	Hệ thống	Thông báo thiết bị đã bị xóa
<b>Hậu điều kiện</b>	<b>STT</b>	<b>Thực hiện</b>	<b>Hành động</b>
	8.3a	Hệ thống	Thông báo lỗi: Không thể xóa dữ liệu
		Không	

**Bảng 2.8:** Đặc tả use case "Xóa dữ liệu"

## 2.5 Yêu cầu phi chức năng

### 2.5.1 Yêu cầu về bảo mật

Server cần phải xác thực danh tính của người dùng trong mỗi request gửi lên. Chỉ những người dùng đã xác thực mới được phép truy cập sử dụng các chức năng và tài nguyên của hệ thống.

### 2.5.2 Yêu cầu về hiệu năng

Thiết bị EPD cần phải duy trì kết nối ổn định với hệ thống để có tốc độ giao tiếp tốt nhất. Ngoài ra, thiết bị cần được cung cấp đủ năng lượng để có thể đạt hiệu năng tốt nhất.

### 2.5.3 Yêu cầu về khả năng mở rộng, nâng cấp và bảo trì

Hệ thống vẫn đang trong giai đoạn phát triển nên cần thiết kế để dễ dàng sửa đổi và nâng cấp sao cho phù hợp mỗi khi có yêu cầu mới. Các tính năng nếu cần sửa đổi hoặc thêm mới cũng sẽ không ảnh hưởng gì đến sự hoạt động của hệ thống.

## CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG

### 3.1 Front-end

#### 3.1.1 Web người dùng - Trang quản lý

Nội dung sau đây được thực hiện bởi Vũ Lê Nhật Minh trong đồ án "Xây dựng bảng thông tin sử dụng giấy điện tử ePaperboard". Tác giả của đồ án này có trách nhiệm đọc, tìm hiểu và khai thác hệ thống hiện tại, đồng thời cũng không thực hiện chỉnh sửa nào đối với WebApp. Đóng góp của tác giả là triển khai lại WebApp trên máy chủ mới tại URL: <https://epaper.toolhub.app/>

##### a, NextJS

Next.js là một framework front-end React được phát triển dưới dạng open-source nhằm đơn giản hóa việc phát triển các ứng dụng một trang (SPAs) và các ứng dụng được render từ phía server [6]. Next.js cho phép phát triển giao diện người dùng bằng React components, đồng thời cung cấp thêm các tính năng và tối ưu hóa. Next.js nổi bật nhờ dễ sử dụng, tối ưu hóa hiệu suất và khả năng kết hợp giữa resnder phía máy chủ (SSR) và tạo trang web tĩnh, góp phần cải thiện trải nghiệm người dùng và tăng tốc độ tải trang. Next.js cũng hỗ trợ tự động phân tách mã, đảm bảo rằng mỗi trang chỉ tải JavaScript cần thiết [7].

Với tính năng render từ phía server (SSR) nổi tiếng và các tối ưu hóa khác, Next.js cải thiện đáng kể hiệu suất website và thời gian phản hồi, đồng thời giữ cho công việc phát triển dễ quản lý và có khả năng mở rộng. Framework này cũng được trang bị các công cụ để đo lường, theo dõi và hiển thị các số liệu hiệu suất trang tự động, chẳng hạn như Next.js Speed Insights và hook useReportWebVitals, cho phép các nhà phát triển liên tục duy trì và cải thiện hiệu suất của website [8].

Những lợi ích về hiệu suất này, bao gồm hiệu suất ứng dụng được cải thiện với SSR và quản lý mã hiệu quả, đã làm cho Next.js trở thành lựa chọn phù hợp cho đồ án.

##### b, TailwindCSS

Tailwind CSS là một utility-first CSS framework, mang lại nhiều lợi ích đáng kể cho việc tăng tốc thời gian render trong phát triển web. Một trong những lợi ích chính của nó là cách tiếp cận tạo ra file CSS nhỏ nhất có thể cho một dự án [9]. Kích thước của các file CSS ảnh hưởng trực tiếp đến tốc độ tải trang web. Các file CSS lớn hơn mất nhiều thời gian hơn để tải, dẫn đến thời gian tải trang tăng lên. Các trang web tải chậm có thể gấp phải tỷ lệ thoát cao hơn, giảm tương tác của người dùng và ảnh hưởng tiêu cực đến xếp hạng trên các công cụ tìm kiếm.

Tailwind giải quyết vấn đề này bằng cách chỉ tạo CSS cho các thành phần thực sự được sử dụng trong dự án. Kết hợp với việc tối giản hóa và nén mạng, điều này dẫn đến các file CSS cực kỳ nhỏ, thường dưới 10kB, ngay cả với các dự án lớn, điều này rất cần thiết cho việc đạt được hiệu suất cao [10].

Ngoài việc giảm kích thước file CSS, Tailwind CSS cũng nâng cao khả năng truy cập và hiệu suất của trang web bằng cách giảm lượng mã HTML. Điều này đạt được thông qua việc sử dụng các bộ chọn hiệu quả hơn, một khía cạnh cốt lõi của mục đích thiết kế Tailwind. Bằng cách tối ưu hóa HTML và CSS của một trang web, Tailwind CSS giúp cải thiện cả tốc độ và chất lượng trải nghiệm người dùng.

Tailwind cũng đóng góp đáng kể vào việc cải thiện các chỉ số quan trọng của trang web. Ví dụ, chỉ số First Contentful Paint (FCP), chỉ số quan trọng để đánh giá hiệu suất trang web, có thể giảm đến 36% so với các phương pháp tạo kiểu khác như styled components [11]. Những cải tiến trong chỉ số Speed Index và Largest Contentful Paint cho thấy rằng Tailwind CSS không chỉ tăng tốc thời gian phát triển mà còn nâng cao hiệu suất và khả năng phản hồi.

### 3.1.2 Ứng dụng trên điện thoại - Mobile App

#### a, React Native

React Native là một Framework do công ty công nghệ nổi tiếng Facebook phát triển cho phép các lập trình viên sử dụng Javascript để làm Mobile Apps trên cả Android và iOS với có cả trải nghiệm và hiệu năng như Native. React Native vượt trội ở chỗ chỉ cần viết một lần là có thể build ứng dụng cho cả iOS lẫn Android. Điều này giúp các nhà phát triển tiết kiệm được thời gian, công sức, chi phí mà vẫn giúp cho tốc độ ra sản phẩm cũng như cập nhật ứng dụng nhanh chóng. Có thể nói React Native là một Cross-platform để xây dựng một ứng dụng di động hiệu quả [12].

React Native tích hợp 2 thread là **Main Thread** và **JS Thread** cho ứng dụng mobile. Main Thread sẽ đảm nhận vai trò cập nhật giao diện và xử lý tương tác người dùng người dùng (UI). Trong khi đó, JS Thread sẽ thực thi và xử lý code Javascript. Hai luồng này hoạt động độc lập với nhau. Để tương tác được với nhau hai Thread sẽ sử dụng một Bridge (cầu nối) cho phép chúng giao tiếp mà không phụ thuộc lẫn nhau, chuyển đổi dữ liệu từ thread này sang thread khác. Dữ liệu từ hai Thread được vận hành khi tiếp nối dữ liệu cho nhau [13].

Ngoài những ưu điểm nêu trên, React Native được phát triển và cải tiến liên tục cũng như có cộng đồng người dùng lớn cùng với lượng thư viện phong phú, đa dạng. Do vậy, đây là một lựa chọn phù hợp với đồ án nhằm xây dựng một hệ thống đạt hiệu năng cao và có khả năng mở rộng.

## b, Redux

Hiện nay, việc phổ biến của React đã đi cùng với sự xuất hiện của rất nhiều thư viện hỗ trợ. Trong đó, quản lý trạng thái ứng dụng là một trong những yêu cầu mà các nhà phát triển cần đến để xây dựng các ứng dụng React. Một số thư viện đã ra đời nhằm giải quyết các yêu cầu đó như: Redux, Mobx, Recoil, Zustand, v.v. Trong đó, Redux là một trong những lựa chọn tốt nhất và phổ biến nhất dành cho ứng dụng hướng tới phát triển và khả năng mở rộng.

Redux là một thư viện JavaScript giúp tạo ra thành một lớp quản lý trạng thái của ứng dụng, phổ biến trong các ứng dụng React được dựa trên nền tảng tư tưởng của kiến trúc Flux do Facebook giới thiệu.

Redux tạo ra một luồng dữ liệu một chiều (unidirectional data flow), giúp quản lý trạng thái của ứng dụng một cách dễ dàng và dễ kiểm soát. Tích hợp Redux vào ứng dụng giúp tạo ra một cấu trúc rõ ràng và dễ bảo trì [14].

Với việc sử dụng Redux, các state của các ứng dụng sẽ được lưu tại một nơi gọi là **Store**, từ đó các component có thể dispatch (điều phối) state bị thay đổi thông qua Store này, và những thành phần nào cần biết về sự thay đổi đó có thể subscribe (đăng ký) vào Store.

Ưu điểm lớn nhất của Redux đó chính là việc quản lý trạng thái ứng dụng và theo dõi các state từ đó giúp việc tìm kiếm, lập trình và gỡ lỗi ứng dụng trở nên dễ dàng. Đối với một đồ án hướng tới việc mở rộng và phát triển thì Redux là một lựa chọn tốt nhất so với các thư viện quản lý trạng thái khác.

## 3.2 Back-end

### 3.2.1 MongoDB

MongoDB là một hệ thống cơ sở dữ liệu NoSQL mã nguồn mở có kiến trúc hướng tài liệu (document-oriented), được thiết kế nhằm lưu trữ một lượng lớn dữ liệu và cho phép làm việc với dữ liệu đó một cách hiệu quả. Hệ thống cơ sở dữ liệu NoSQL cung cấp giải pháp thay thế cho cơ sở dữ liệu quan hệ truyền thống sử dụng SQL. Trong MongoDB, dữ liệu được lưu trữ trong các document sử dụng cấu trúc giống JSON để biểu diễn và tương tác với dữ liệu [15].

Kiến trúc của cơ sở dữ liệu MongoDB bao gồm Database, Collection và Document. Database là vùng chứa vật lý cho dữ liệu và mỗi Database có một tập tin riêng lưu trữ trên bộ nhớ vật lý. Collection là một nhóm các database document nằm trong một cơ sở dữ liệu duy nhất. Các collection không cần định nghĩa các cột, các hàng hay kiểu dữ liệu trước. Document là một tập hợp các cặp key - value có thể được chỉ định được liên kết với các dynamic schema. Lợi ích của dynamic

schema là document trong một collection không nhất thiết phải có cùng cấu trúc hoặc các trường. Ngoài ra, các trường phổ biến trong document của collection có thể có nhiều loại dữ liệu khác nhau.

Ưu điểm lớn của MongoDB là tính linh hoạt lưu trữ dữ liệu dưới dạng hướng tài liệu JSON, dễ dàng mở rộng hệ thống bằng cách thêm node vào cluster và tốc độ truy vấn nhanh so với hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) [16]. Do đó, MongoDB là lựa chọn hàng đầu cho việc xây dựng một đồ án yêu cầu về hiệu năng và khả năng mở rộng cùng với nhiều loại dữ liệu khác nhau.

### 3.2.2 ExpressJS

ExpressJS là một framework ứng dụng web mã nguồn mở và phổ biến được xây dựng trên nền tảng Node.js. ExpressJS được sử dụng để thiết kế và phát triển các ứng dụng web một cách nhanh chóng. Việc xây dựng các ứng dụng web và API bằng ExpressJS khá đơn giản đối với các lập trình viên và nhà phát triển do không yêu cầu hiểu biết về nhiều ngôn ngữ ngoài JavaScript.

ExpressJS khá nhẹ, giúp tổ chức các ứng dụng web ở phía máy chủ thành một kiến trúc MVC hoàn hảo hơn. Nó còn hỗ trợ nâng cao các chức năng của NodeJS. Nếu không sử dụng ExpressJS, lập trình viên phải thực hiện rất nhiều lập trình phức tạp để xây dựng một API hiệu quả. ExpressJS đã giúp cho việc lập trình trong NodeJS trở nên dễ dàng hơn.

### 3.2.3 Mosquitto MQTT

Để cho phép giao tiếp và truyền dữ liệu theo thời gian thực giữa người dùng và thiết bị, hệ thống cũng cần hoạt động như một MQTT broker, hỗ trợ việc trao đổi thông tin giữa các bên. Điều này được thực hiện bằng cách sử dụng Mosquitto MQTT, một message broker mã nguồn mở cho phép (được cấp phép theo EPL/EDL) triển khai các phiên bản giao thức MQTT 5.0, 3.1.1 và 3.1. Mosquitto nhẹ và phù hợp để sử dụng trên tất cả các thiết bị từ máy tính bảng đơn tiêu thụ điện năng thấp đến các máy chủ hoàn chỉnh [17].

Giao thức MQTT cung cấp một phương pháp giao tiếp sử dụng mô hình publish/subscribe. Điều này khiến nó phù hợp cho việc nhắn tin trong Internet of Things, chẳng hạn như với các cảm biến tiêu thụ điện năng thấp hoặc các thiết bị di động như điện thoại, máy tính nhúng hoặc vi điều khiển.

Nó hoạt động như một trung gian cho việc nhắn tin bằng cách nhận và chuyển tiếp tin nhắn, trở thành một công cụ quan trọng để xử lý giao tiếp không đồng bộ giữa các thiết bị EPD và máy chủ. Việc xây dựng một hệ thống dành cho các doanh nghiệp nhỏ và vừa sẽ ưu tiên hiệu quả và nhỏ nhẹ nên Mosquitto MQTT là một lựa

chọn phù hợp.

### 3.3 Thiết bị EPD

#### 3.3.1 ESP32-C3 Supermini

Các thiết bị EPD trong đồ án sử dụng ESP32-C3 Supermini, một module Wi-Fi và Bluetooth LE đa năng và nhỏ gọn, thừa hưởng các tính năng từ các vi điều khiển ESP32. ESP32-C3 Supermini sử dụng ESP32-C3, một MCU dựa trên RISC-V 32-bit với 400KB SRAM có khả năng chạy lên đến 160 MHz làm lõi xử lý. Nó cũng tích hợp Wi-Fi 2.4 GHz và Bluetooth 5 (LE) với hỗ trợ tầm xa. Ngoài ra, nó còn có 22 chân GPIO (General Purpose Input/Output) lập trình được, hỗ trợ nhiều giao diện khác nhau, bao gồm ADC (Analog to Digital Converter), SPI (Serial Peripheral Interface), UART (Universal Asynchronous Receiver/Transmitter), I2C (Inter-Integrated Circuit), RMT (Remote Control), v.v. ESP32-C3 Supermini bao gồm cổng USB Type-C, 4MB Flash và hỗ trợ 12 chân IO, đủ cho hầu hết các tác vụ phát triển trong đồ án [18].

#### 3.3.2 WeAct-Epaper 2.9 Inch

Màn hình trong đồ án sử dụng module WeAct-Epaper 2.9 Inch, một giải pháp hiển thị linh hoạt và hiệu quả cho nhiều ứng dụng khác nhau. Nó có màn hình 2.9 inch với độ phân giải 296x128 pixel và hỗ trợ làm mới từng phần, cho phép tiêu thụ điện năng thấp và góc nhìn rộng. Màn hình hiển thị nội dung đen trắng lý tưởng cho các ứng dụng như thẻ giá, nhãn kệ hàng, và thiết bị công nghiệp. Màn hình được kết nối và giao tiếp với mô-đun ESP32-C3 Supermini qua giao diện SPI, đặc biệt nổi bật với khả năng tiêu thụ điện năng cực thấp, làm cho nó phù hợp với hệ thống này nơi hiệu quả năng lượng là rất quan trọng.

### 3.4 Bluetooth Low Energy - BLE

Bluetooth Low Energy (BLE) là một mạng không dây tương tự Bluetooth nhưng sử dụng năng lượng thấp và hoạt động ở băng tần ISM 2,4 GHz. BLE được thiết kế để sử dụng các thiết bị kết nối với nhau trong một phạm vi ngắn. Bluetooth Low Energy hiện nay được sử dụng cho những thiết bị thông minh như điện thoại, máy tính, tai nghe không dây, đồng hồ, v.v. và không ít những hệ thống IoT.

Không giống như Bluetooth Classic, BLE đạt tối đa chỉ 1Mbps và tiêu thụ chỉ 0,01 đến 0,5 watt. Tốc độ đó chỉ bằng một phần ba so với Bluetooth Classic và không quá một nửa công suất. BLE luôn duy trì chế độ ngủ, trừ khi có thiết bị kết nối tới so với Bluetooth Classic luôn ở trạng thái bật. Điều này làm cho nó tiêu thụ điện năng rất thấp, ít hơn khoảng 100 lần so với Bluetooth Classic, tùy thuộc vào ứng dụng. Chi tiết về công nghệ BLE so với Bluetooth Classic sẽ được trình bày ở phần 5.1.

### 3.5 PlatformIO

PlatformIO là một plugin có khả năng lập trình Arduino/ARM mbed, hỗ trợ tới hơn 800 board khác nhau ở thời điểm hiện tại, được sử dụng rộng rãi cho phát triển hệ thống IoT và nhúng [19]. Khả năng tương thích đa nền tảng của nó cho phép tích hợp với Visual Studio Code, mang lại sự linh hoạt cho các nhà phát triển.

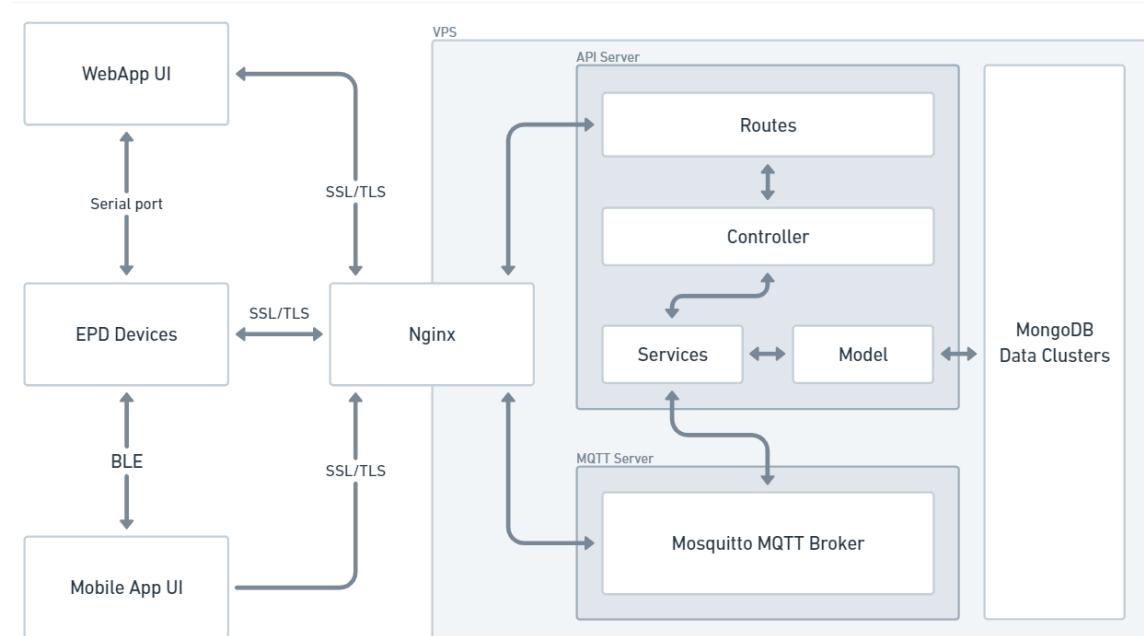
Một trong những tính năng nổi tiếng nhất của PlatformIO là bộ công cụ phát triển toàn diện, bao gồm trình quản lý thư viện mạnh mẽ giúp đơn giản hóa việc tích hợp và quản lý các thư viện bên ngoài, cùng với trình gỡ lỗi hợp nhất hỗ trợ nhiều công cụ gỡ lỗi. Nó cũng hỗ trợ nhiều môi trường phát triển, điều này đặc biệt có giá trị trong các tình huống phát triển phức tạp, cho phép các nhà phát triển tạo ra các ứng dụng linh hoạt có thể dễ dàng thích ứng và triển khai trên các nền tảng phần cứng khác nhau. Cuối cùng nhưng không kém phần quan trọng, PlatformIO còn xây dựng một cộng đồng lớn, làm phong phú nền tảng và cung cấp tài liệu phong phú, diễn đàn người dùng hoạt động, và kiến thức chia sẻ, biến nó thành một công cụ hữu ích cho cả người mới bắt đầu và lâu năm trong phát triển hệ thống nhúng.

Với sự hỗ trợ của PlatformIO, đồ án đã tiến bộ đáng kể cả về hiệu quả phát triển và độ bền kỹ thuật. Nó cung cấp một bộ công cụ toàn diện để quản lý, phát triển, và kiểm tra các board phát triển, được tích hợp với nhiều môi trường phát triển, làm cho quá trình phát triển và gỡ lỗi trở nên dễ dàng hơn.

## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

### 4.1 Thiết kế kiến trúc

#### 4.1.1 Lựa chọn kiến trúc phần mềm



Hình 4.1: Kiến trúc tổng thể của hệ thống

Hình 4.1 minh họa kiến trúc tổng quát của hệ thống. Hệ thống này sử dụng kết hợp giữa kiến trúc MVCS (Model-View-Controller-Service) và Microservice để tận dụng những ưu điểm đặc trưng của chúng. Kiến trúc kết hợp này tận dụng các lợi ích của cả hai kiến trúc, như khả năng mở rộng, tính linh hoạt và khả năng sử dụng các công nghệ khác nhau trong từng dịch vụ.

Kiến trúc microservices là một phương pháp phát triển phần mềm cấu trúc một ứng dụng thành một tập hợp các dịch vụ kết nối độc lập. Trong kiến trúc này, mỗi dịch vụ có thể được phát triển, triển khai và mở rộng độc lập, cho phép khả năng mở rộng, linh hoạt và nhanh nhẹn hơn so với các kiến trúc đơn khối, vì các nhóm có thể cập nhật hoặc sửa chữa các thành phần riêng lẻ mà không ảnh hưởng đến toàn bộ hệ thống trong trường hợp có lỗi.

Nhờ khả năng phù hợp với tính linh hoạt và phân tán của các ứng dụng IoT, kiến trúc microservice thường được sử dụng trong nhiều dự án IoT khác nhau. Nó cho phép các tổ chức xây dựng các ứng dụng linh hoạt và thích ứng hơn để xử lý các yêu cầu phức tạp của môi trường kinh doanh hiện đại.

Mặt khác, kiến trúc Model-View-Controller-Service (MVCS) là sự mở rộng của

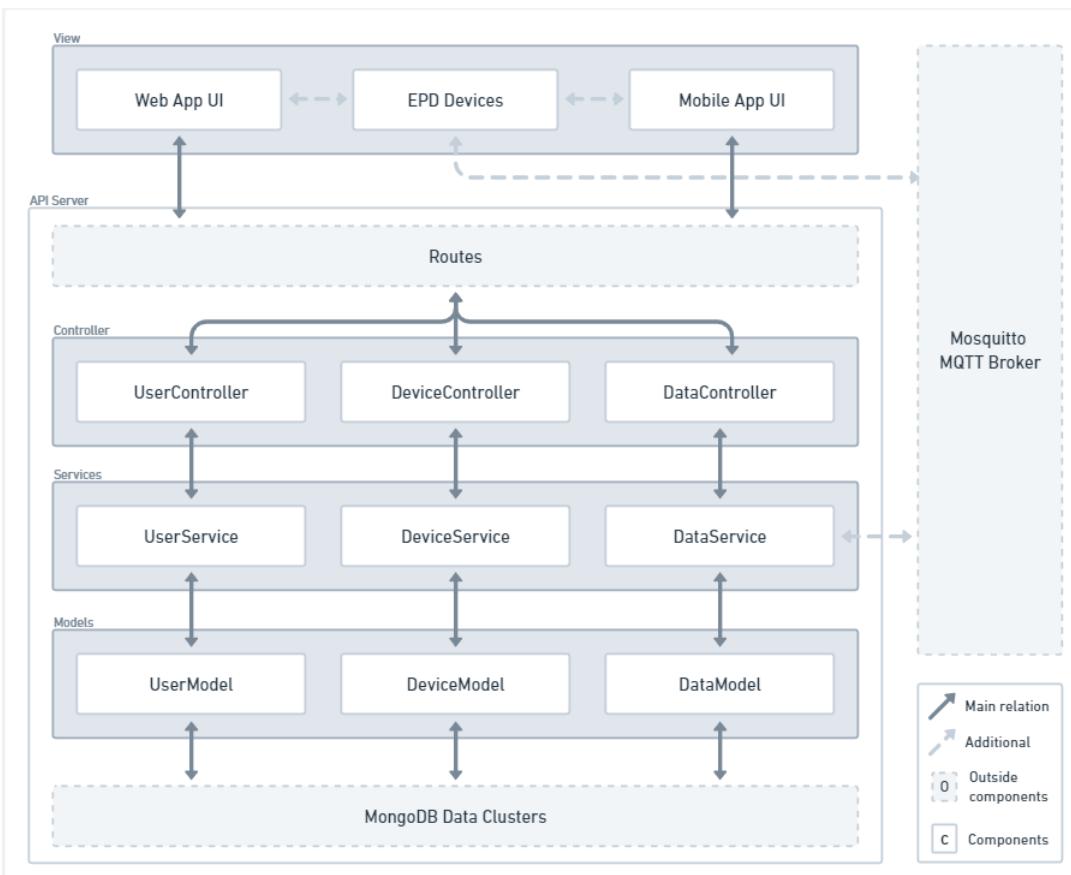
mô hình Model-View-Controller (MVC) truyền thống. Nó giới thiệu một lớp dịch vụ nằm giữa Controller và Model, chịu trách nhiệm chứa đựng logic và quy tắc kinh doanh. Lớp này trùu tượng hóa các luồng nghiệp vụ, cho phép các Controller tập trung vào việc xử lý các yêu cầu đến và giao việc xử lý cho các dịch vụ. Mẫu thiết kế này phù hợp với nhiều ứng dụng web, cung cấp một cách tiếp cận có cấu trúc và tổ chức để phát triển các ứng dụng phức tạp và giúp dễ dàng quản lý, bảo trì và mở rộng hệ thống theo thời gian.

Bằng cách kết hợp hai kiến trúc này, hệ thống được trang bị tốt hơn để xử lý bản chất động và phân tán của các ứng dụng IoT. Nó cũng cho phép các tổ chức xây dựng các ứng dụng linh hoạt và thích ứng hơn để đáp ứng các yêu cầu phức tạp của môi trường kinh doanh hiện đại trong khi đảm bảo khả năng mở rộng, độ tin cậy và sự giao tiếp liền mạch giữa các thành phần khác nhau.

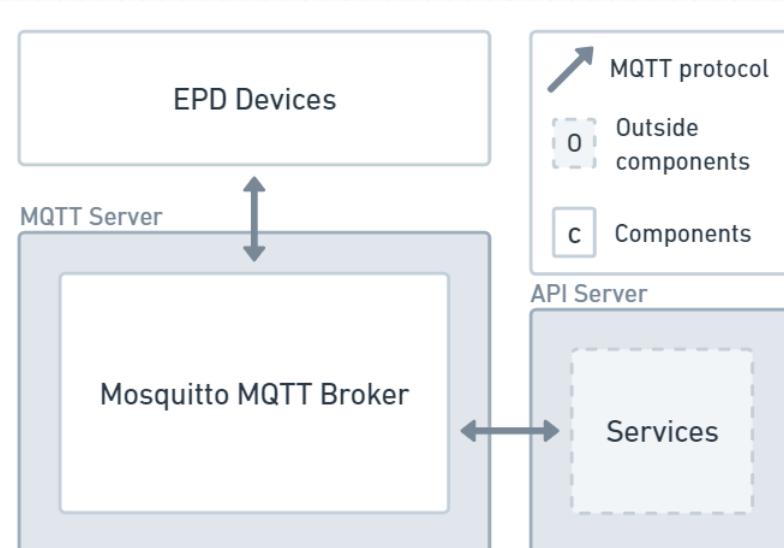
### 4.1.2 Thiết kế tổng quan

Hệ thống được chia thành các dịch vụ có thể triển khai độc lập, bao gồm API Server và MQTT Server. Các dịch vụ này giao tiếp thông qua giao thức MQTT. API Server tuân theo kiến trúc MVCS với ba thành phần chính: Controllers xử lý các yêu cầu đến và điều hướng các phản hồi, Services chứa các xử lý logic và Models giao tiếp với cơ sở dữ liệu, quản lý việc lưu trữ và truy xuất dữ liệu, trong khi Giao diện Web, Mobile app và các thiết bị EPD đóng vai trò như một thành phần View (Hình 4.2). MQTT Server cũng đóng vai trò như một broker để chuyển tiếp dữ liệu đến các thiết bị EPD (Hình 4.3).

## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG



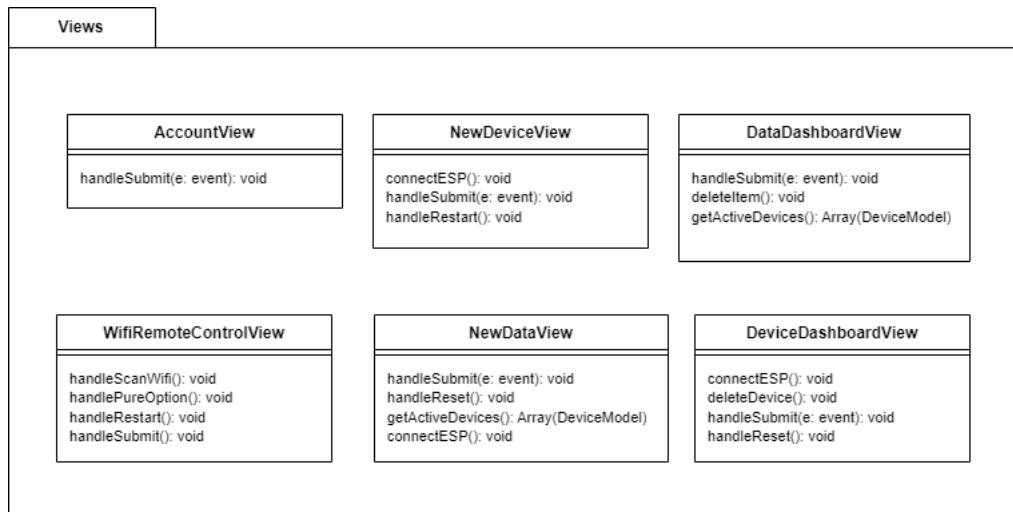
Hình 4.2: Kiến trúc MVCS



Hình 4.3: MQTT Service

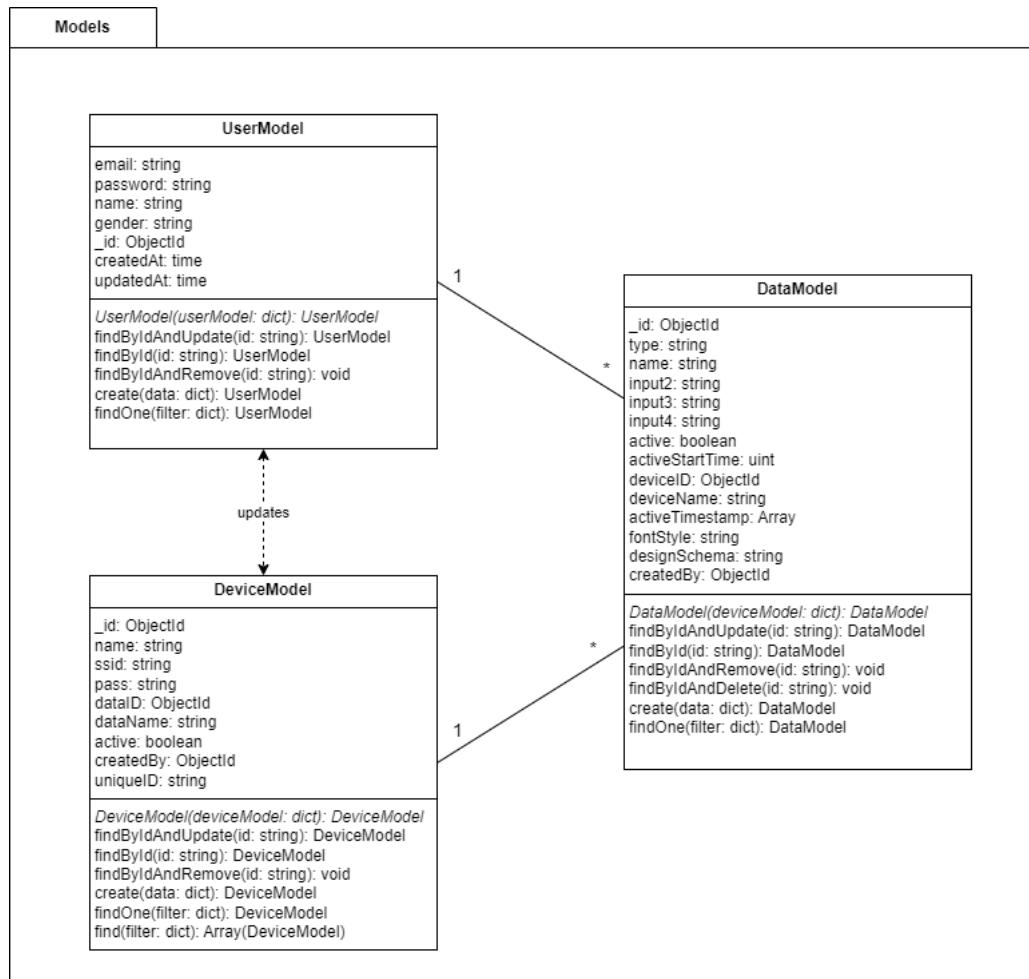
### 4.1.3 Thiết kế chi tiết gói

#### a, Gói Views



Hình 4.4: Sơ đồ lớp chi tiết gói Views

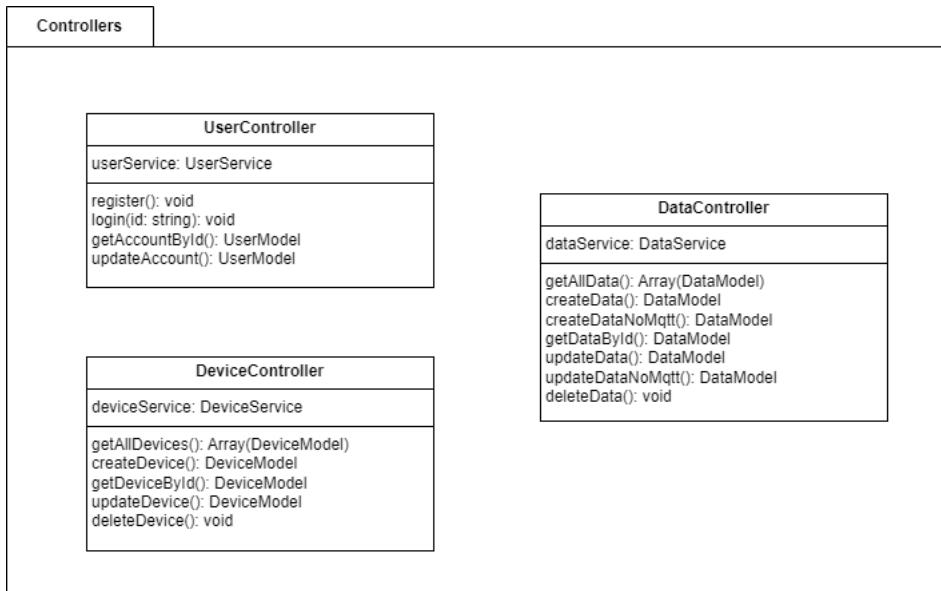
#### b, Gói Models



Hình 4.5: Sơ đồ lớp chi tiết gói Models

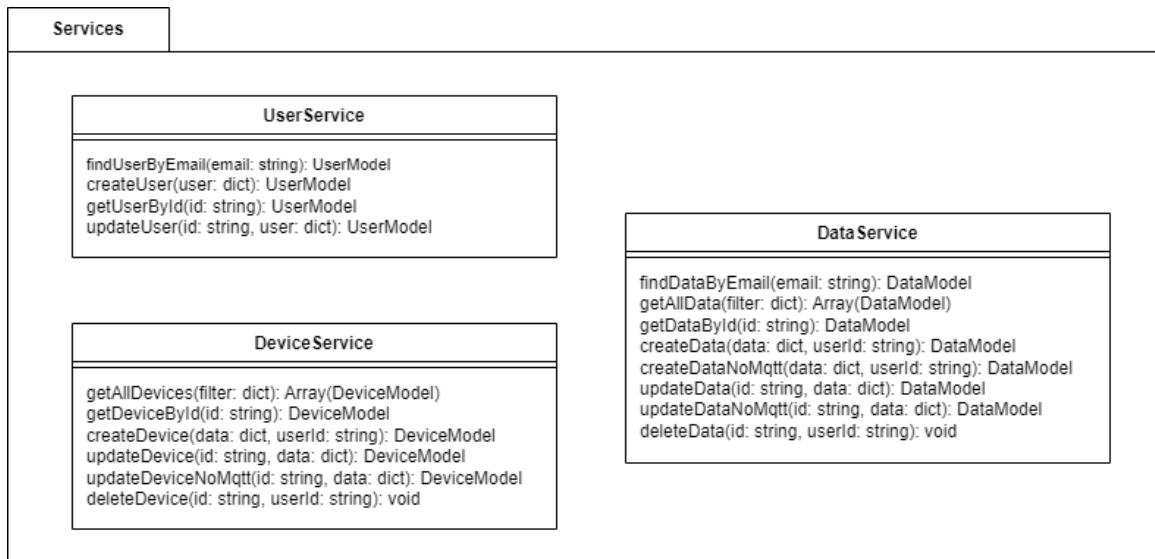
## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

### c, Gói Controllers



Hình 4.6: Sơ đồ lớp chi tiết gói Controllers

### d, Gói Services



Hình 4.7: Sơ đồ lớp chi tiết gói Services

## 4.2 Thiết kế chi tiết

### 4.2.1 Thiết kế giao diện

Trong phiên bản 1 của hệ thống, giao diện dành cho Web App đã được thiết kế và mô tả chi tiết. Do vậy, trong đồ án này sẽ mô tả giao diện sử dụng cho thiết bị di động (Mobile App).

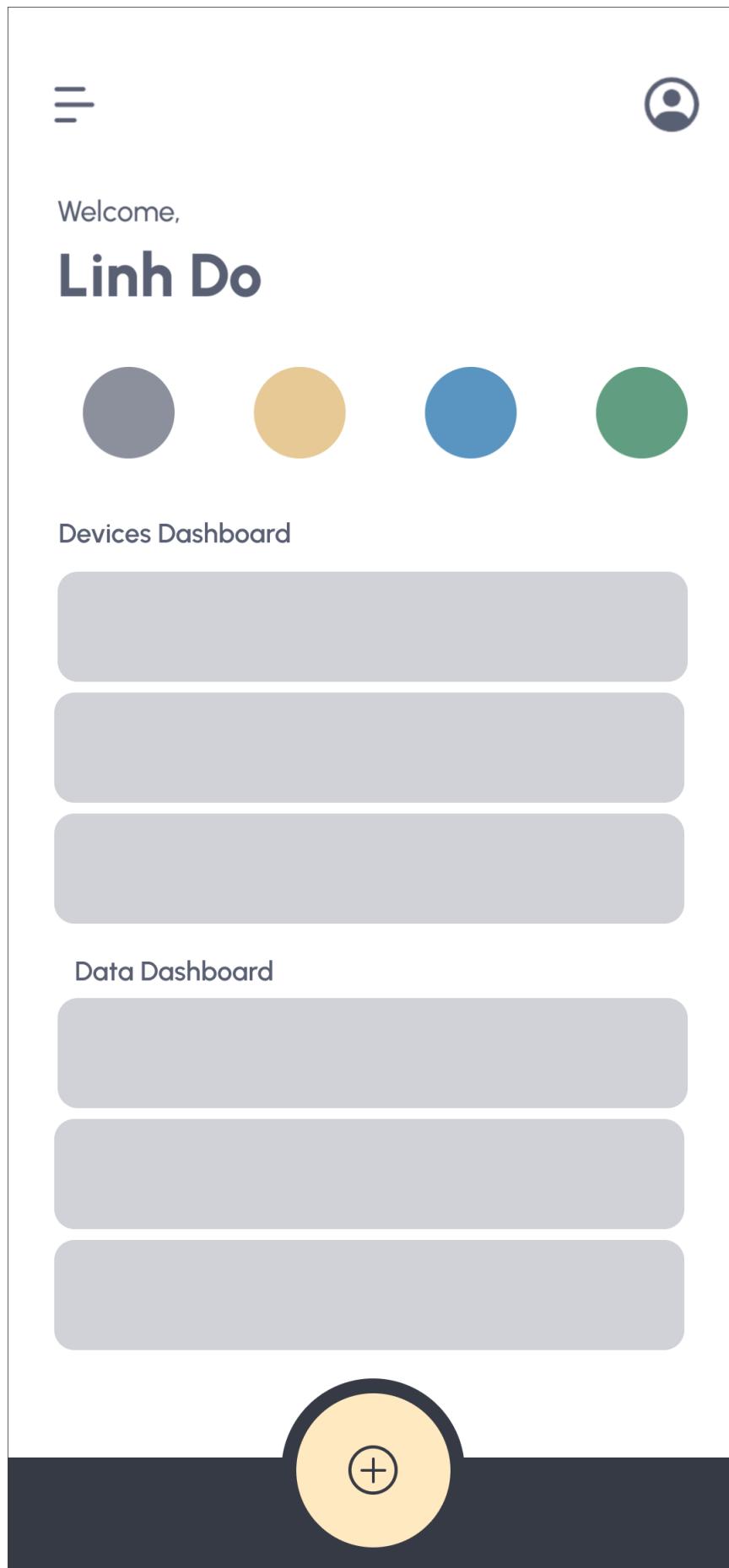
Đầu tiên, thiết kế sử dụng cho nền tảng Android với độ phân giải màn hình

## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

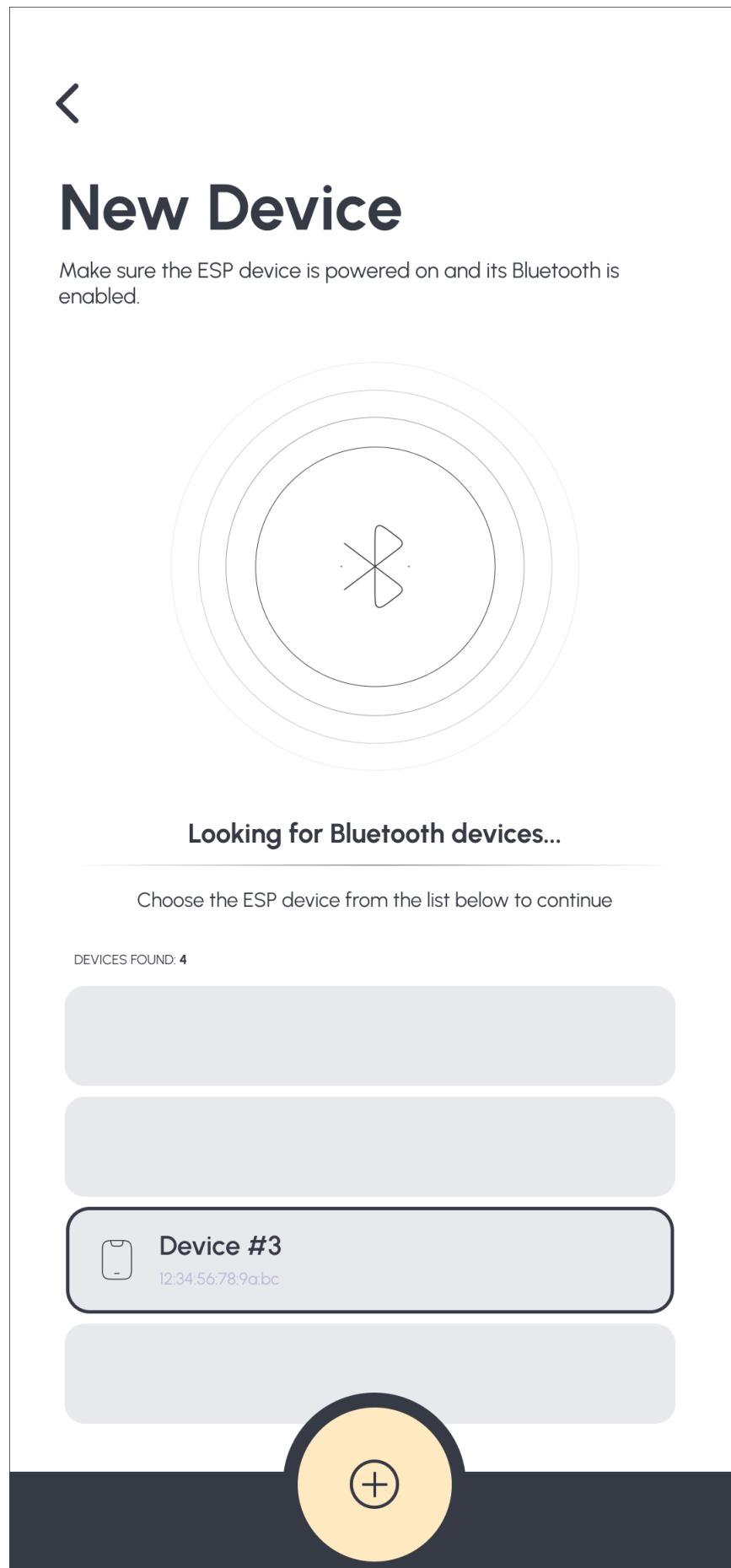
1080x2340 hoặc lớn hơn. Giao diện của hệ thống sử dụng màu trắng (mã màu #ffffff) làm nền cho các màn hình và sử dụng màu #5a6073 cho chữ, các thông tin hiển thị. Ngoài ra, một số màu như vàng nhạt (#ffdffa5), xanh lá (#79c5a1), xanh dương (#64a5d8) và đỏ nhạt (#ff8d9c) cũng được sử dụng để phân biệt các nút bấm và chức năng.

Thông tin phản hồi và thông báo lỗi sẽ được hiển thị tại hai modal là ở giữa màn hình và ở dưới cùng của màn hình. Các modal này sẽ bao gồm tiêu đề, nội dung và các nút chức năng. Tùy vào từng luồng mà các modal sẽ có những tính năng và thiết kế thêm.

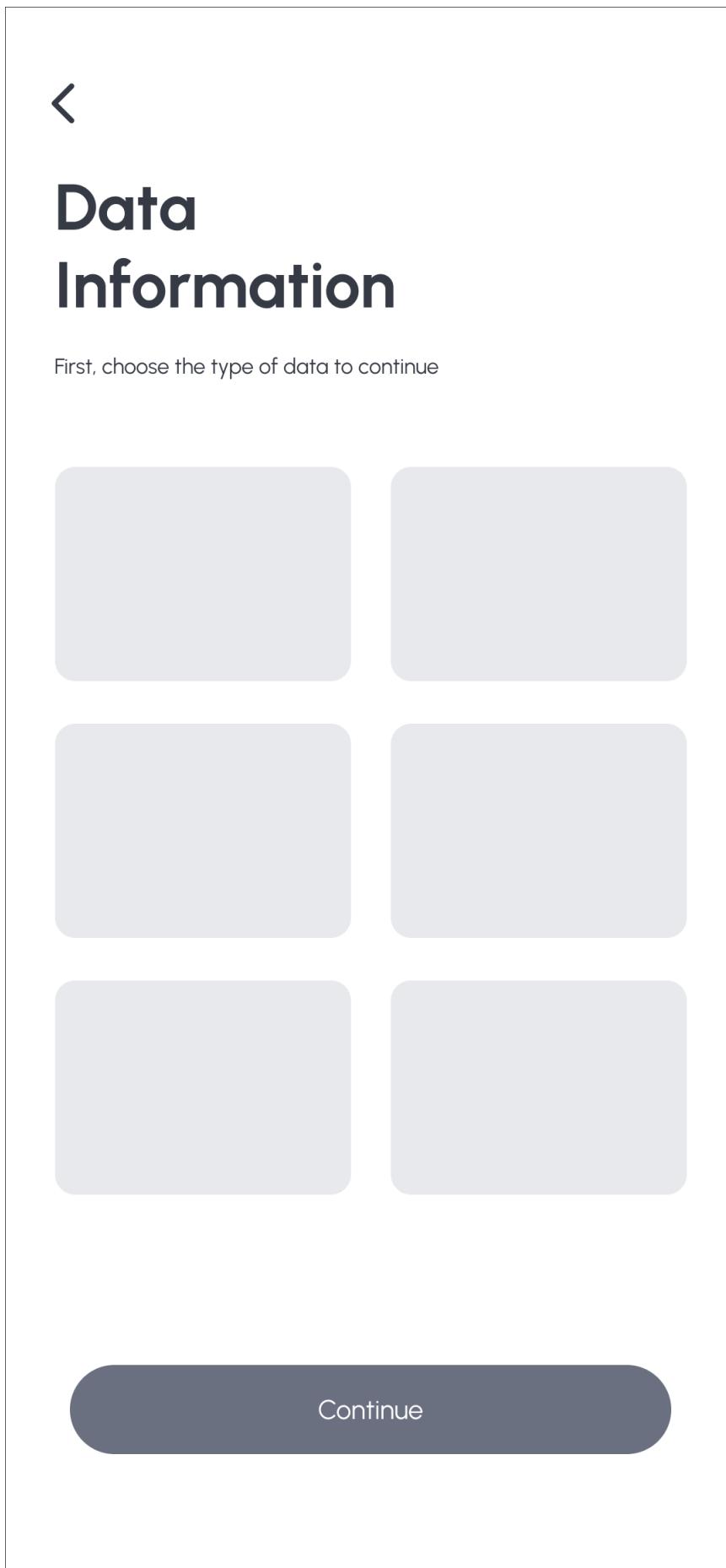
Một số thiết kế giao diện như sau:



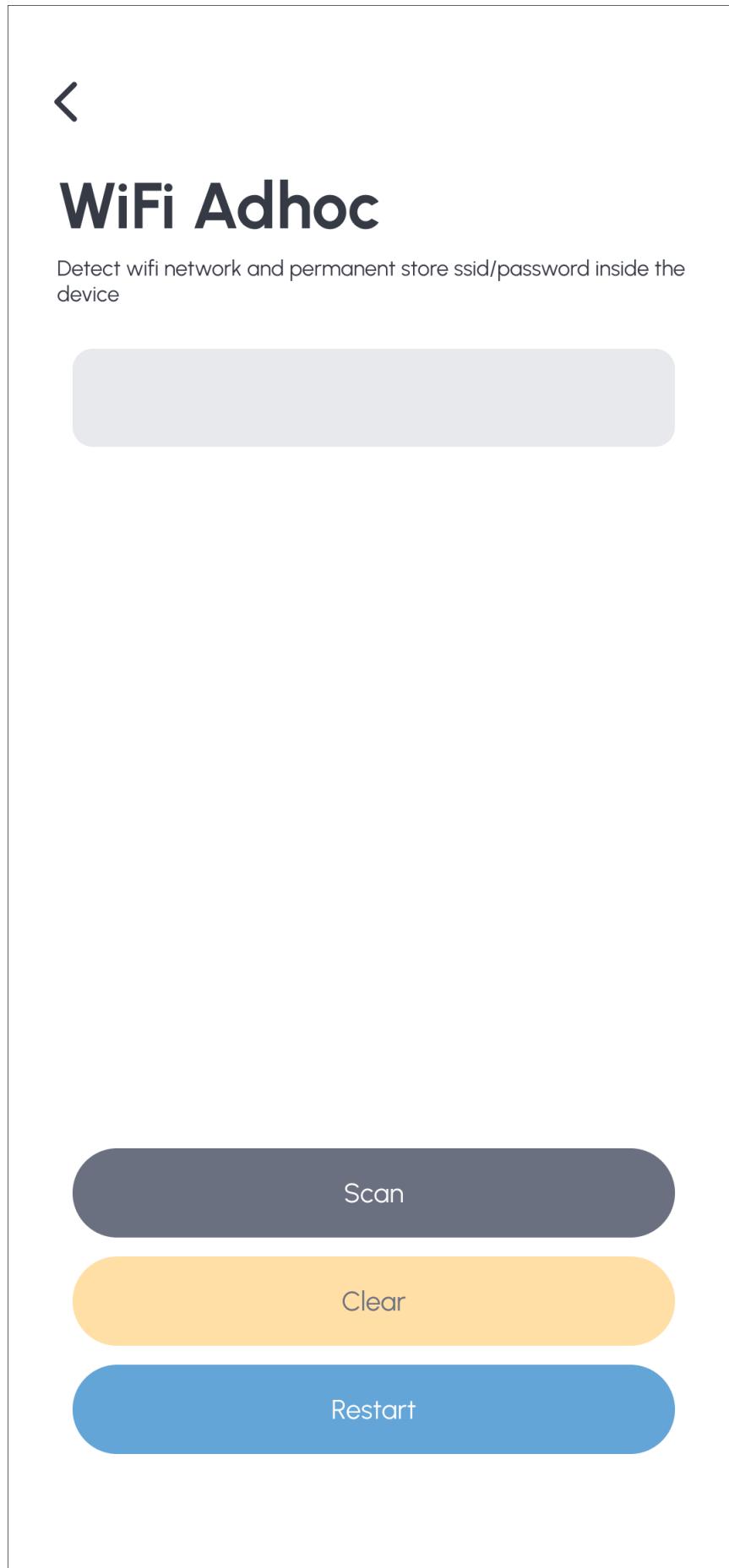
**Hình 4.8:** Thiết kế màn hình chính



**Hình 4.9:** Thiết kế màn tạo thiết bị mới



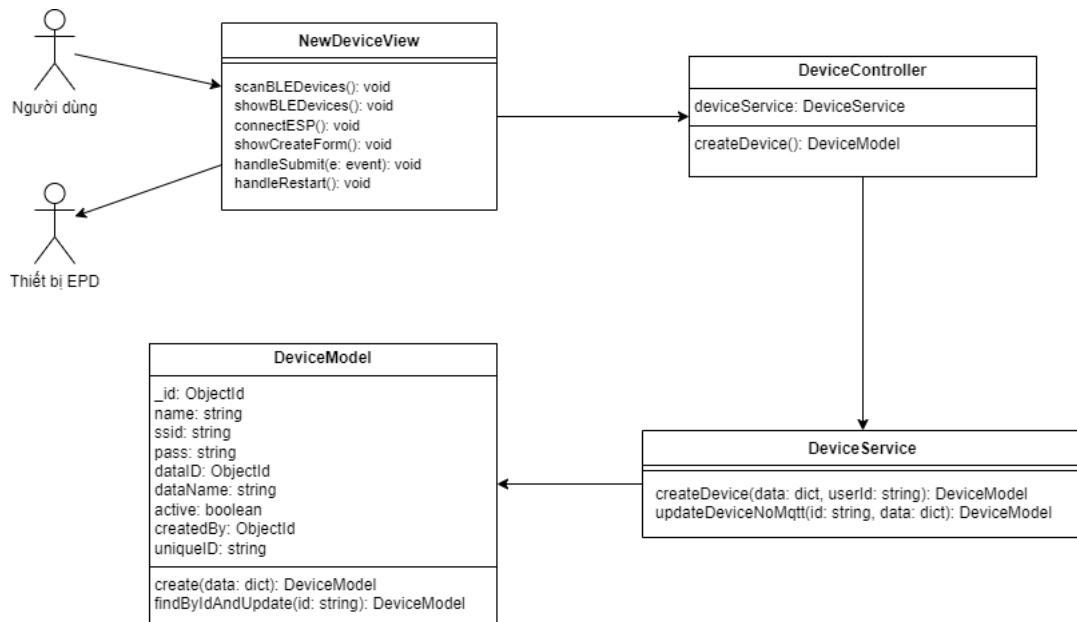
**Hình 4.10:** Thiết kế màn tạo dữ liệu mới



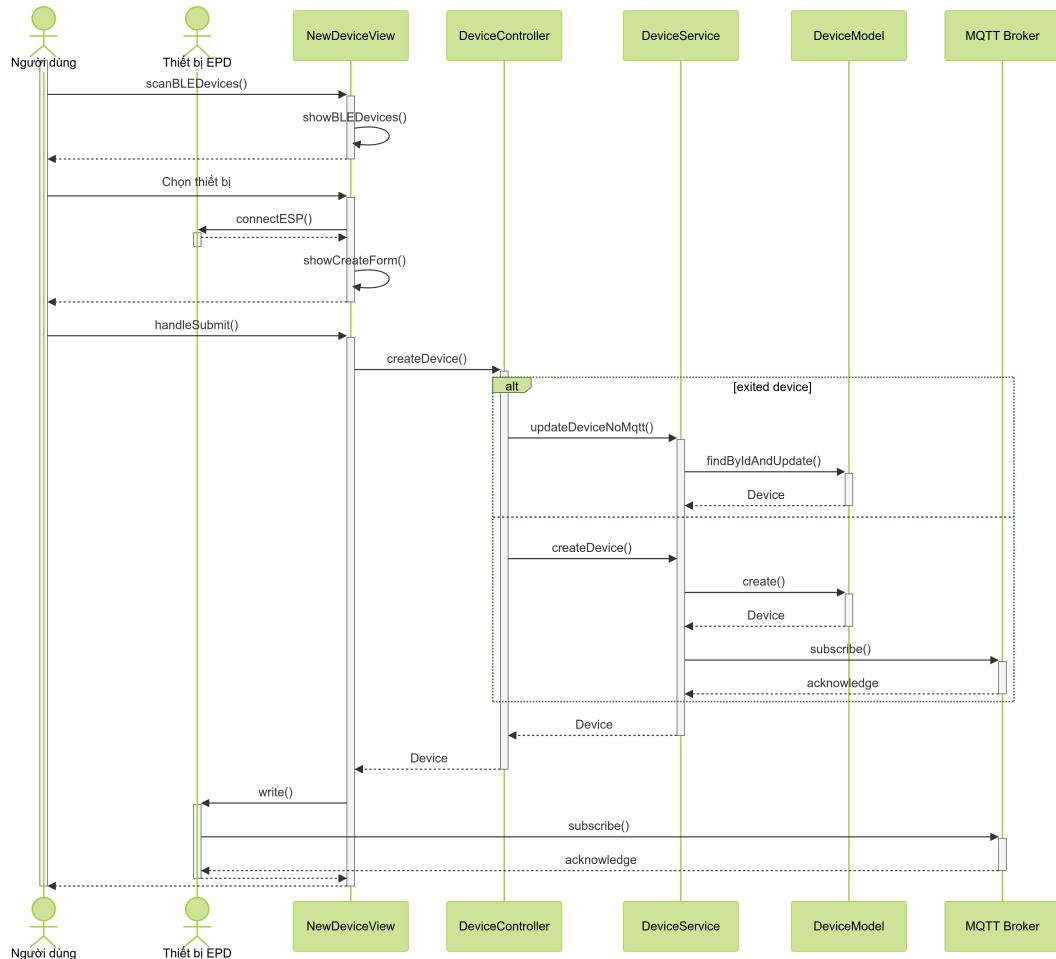
**Hình 4.11:** Thiết kế màn điều khiển kết nối WiFi Adhoc

#### 4.2.2 Thiết kế lớp

##### a, Biểu đồ lớp và biểu đồ trình tự cho ca sử dụng "Tạo thiết bị"

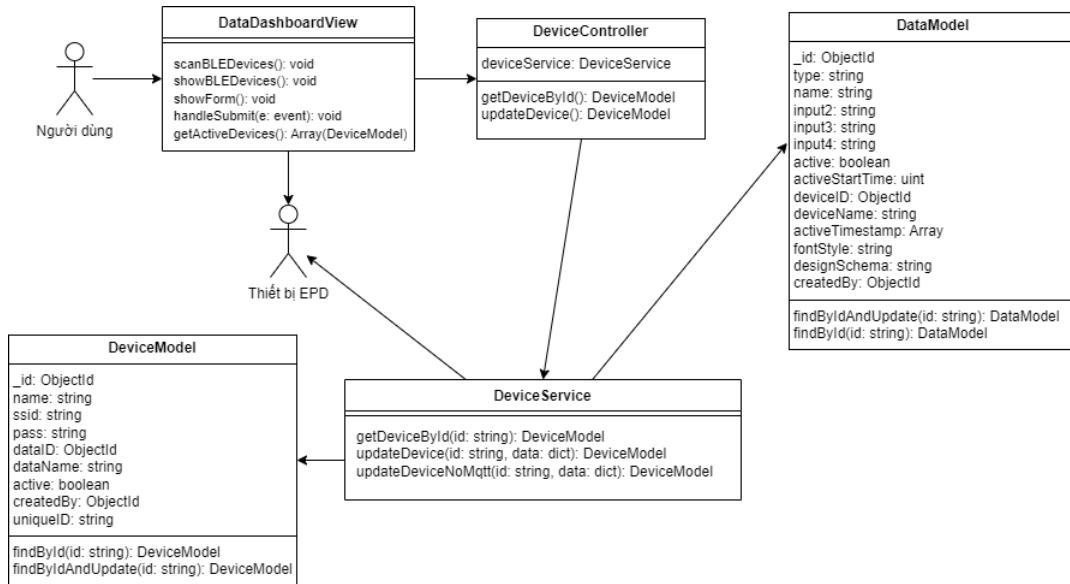


**Hình 4.12:** Biểu đồ lớp cho ca sử dụng "Tạo thiết bị"

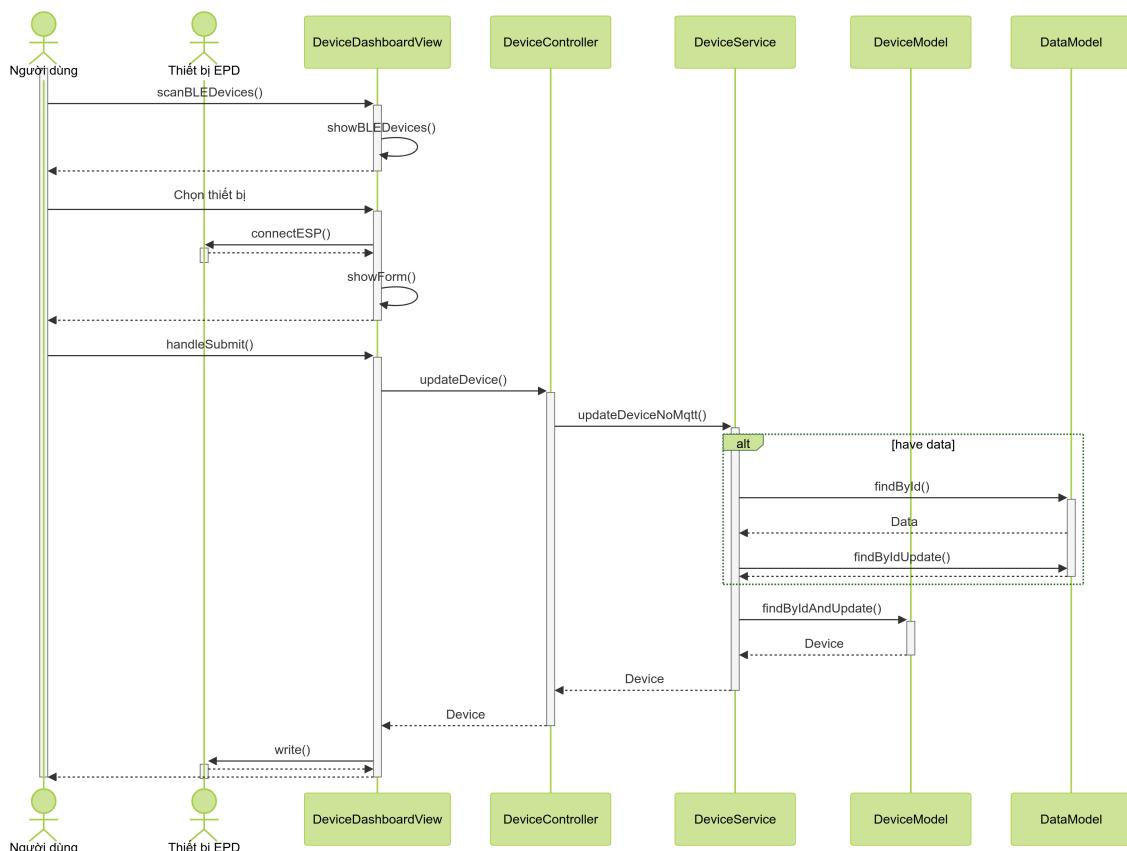


**Hình 4.13:** Biểu đồ trình tự cho ca sử dụng "Tạo thiết bị"

b, Biểu đồ lớp và biểu đồ trình tự cho ca sử dụng "Chỉnh sửa thiết bị qua Bluetooth"

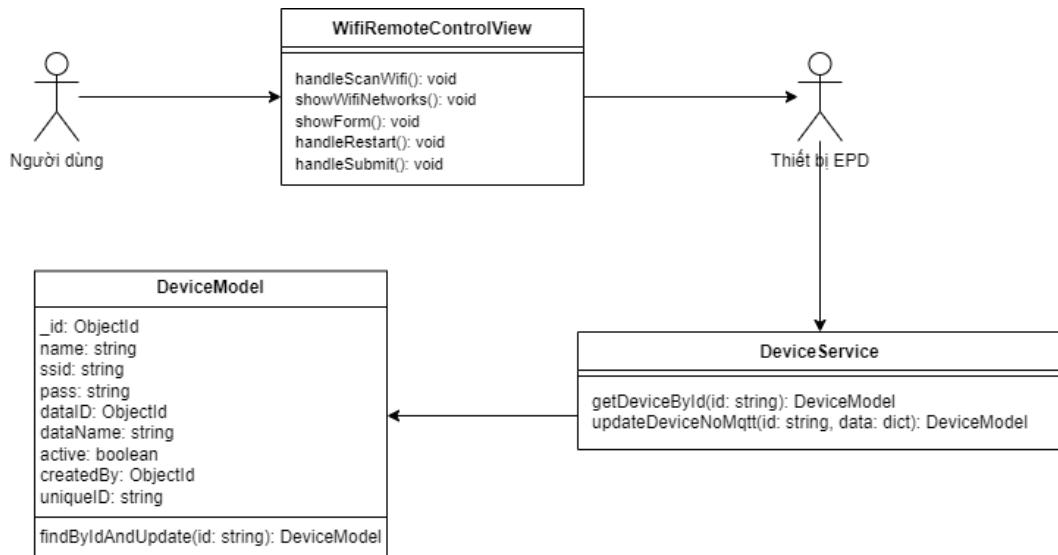


**Hình 4.14:** Biểu đồ lớp cho ca sử dụng "Chỉnh sửa thiết bị qua Bluetooth"

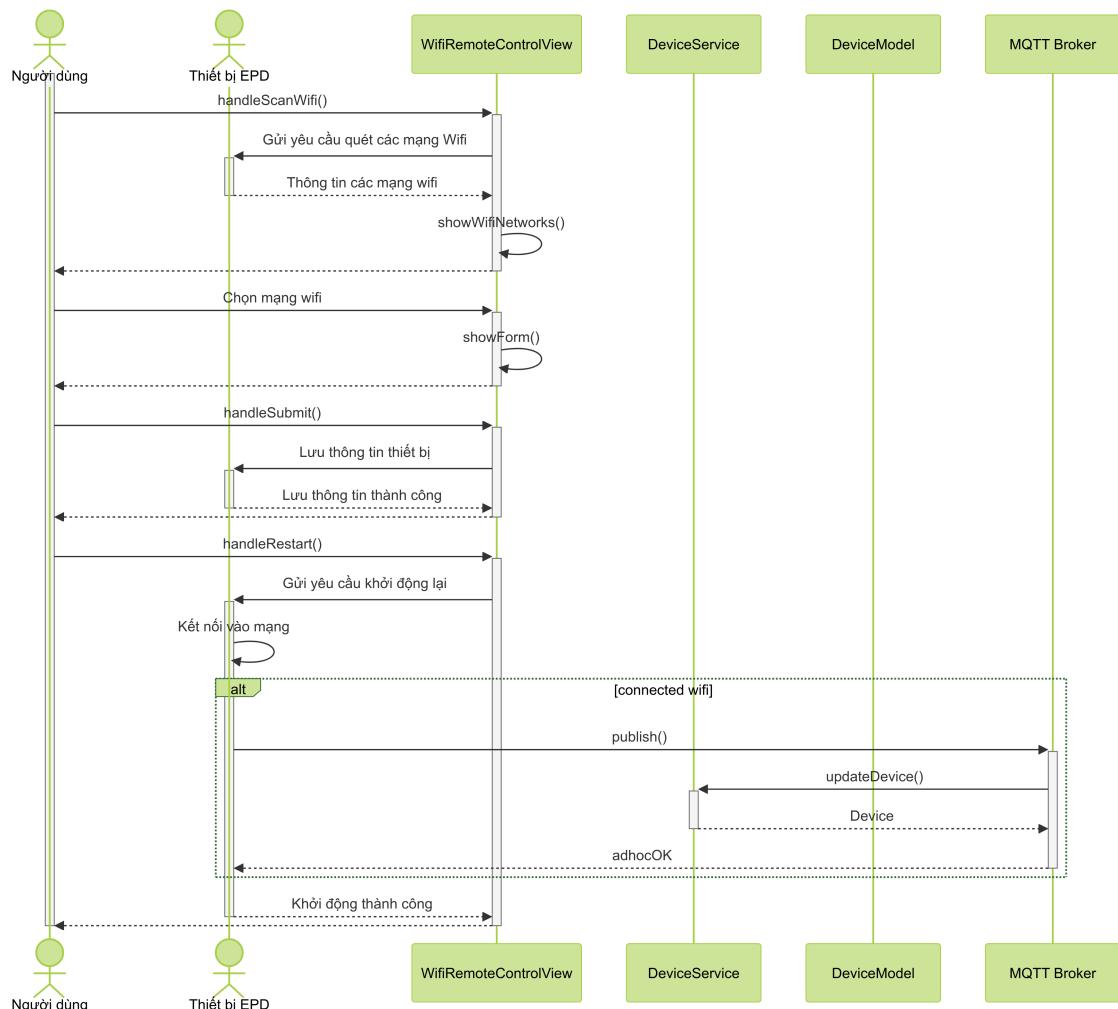


**Hình 4.15:** Biểu đồ trình tự cho ca sử dụng "Chỉnh sửa thiết bị qua Bluetooth"

c, Biểu đồ lớp và biểu đồ trình tự cho ca sử dụng "Chỉnh sửa thiết bị qua Wifi Adhoc"

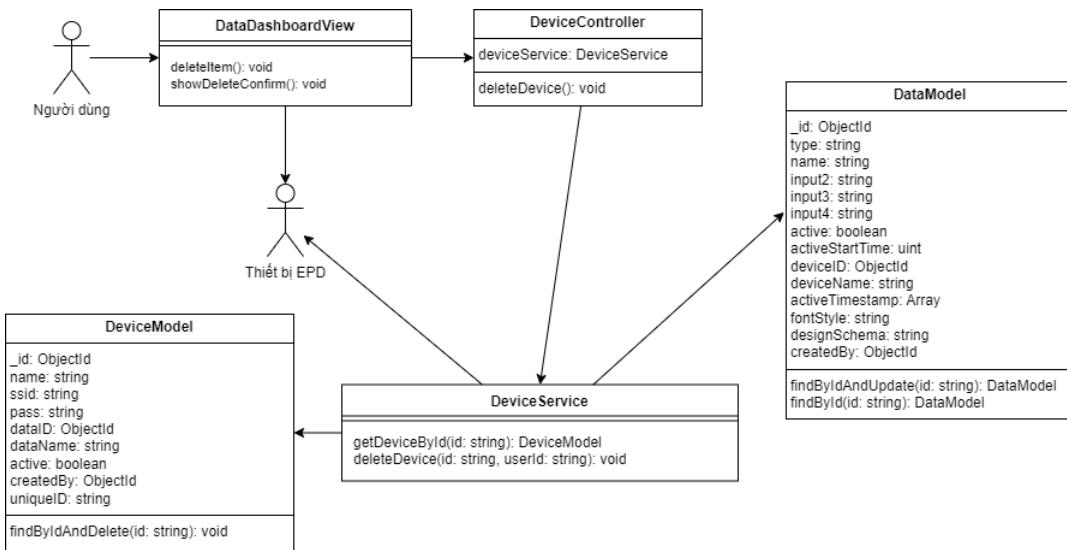


**Hình 4.16:** Biểu đồ lớp cho ca sử dụng "Chỉnh sửa thiết bị qua Wifi Adhoc"

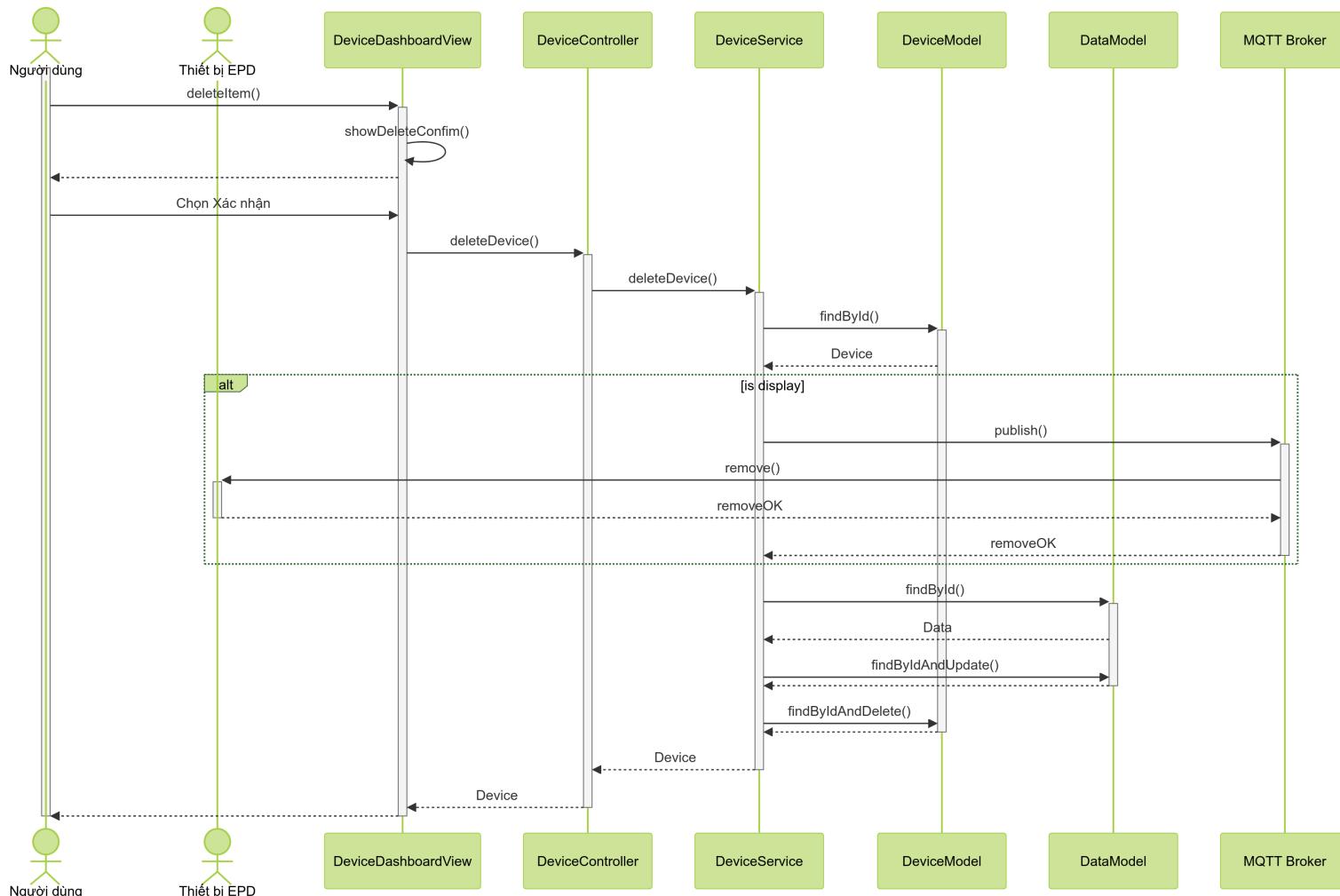


**Hình 4.17:** Biểu đồ trình tự cho ca sử dụng "Chỉnh sửa thiết bị qua Wifi Adhoc"

**d, Biểu đồ lớp và biểu đồ trình tự cho ca sử dụng "Xóa thiết bị"**

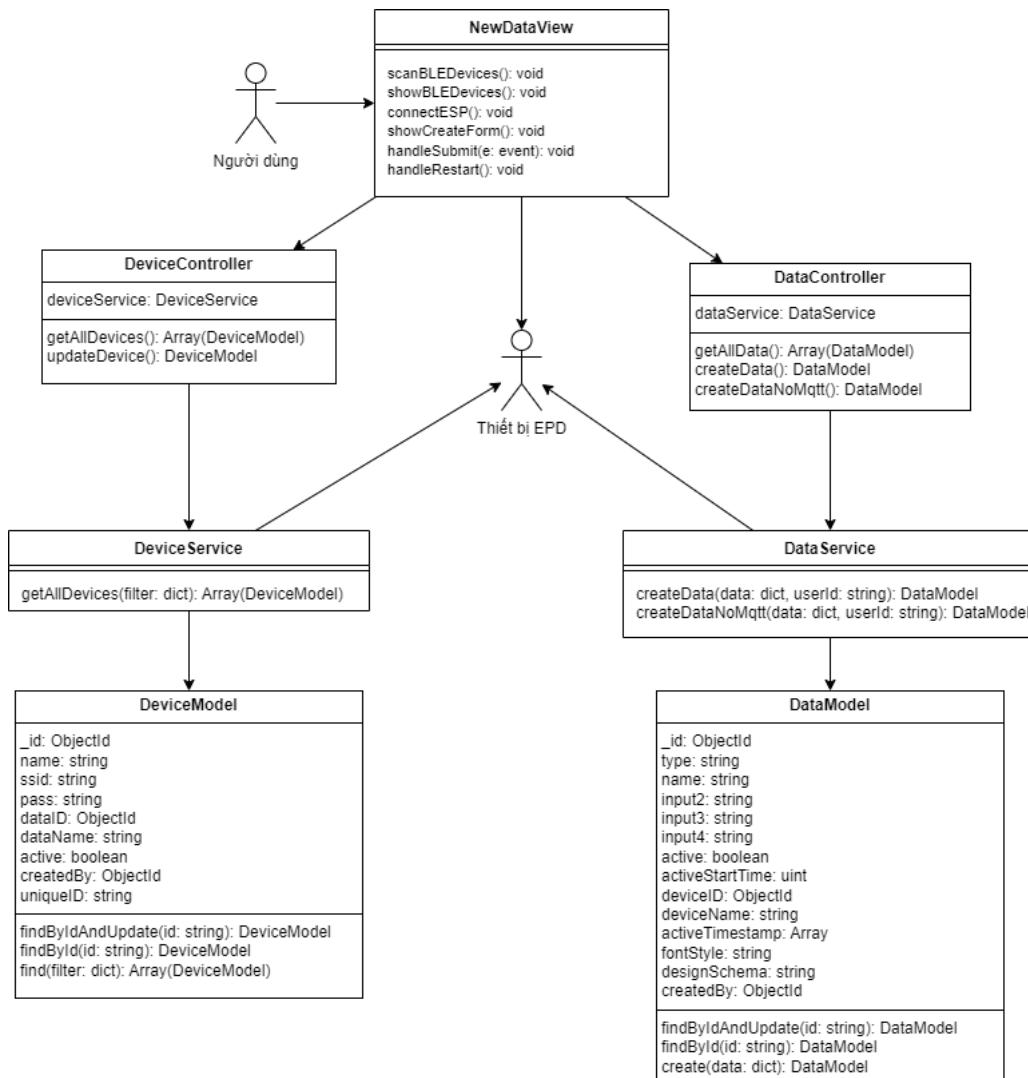


**Hình 4.18:** Biểu đồ lớp cho ca sử dụng "Xóa thiết bị"



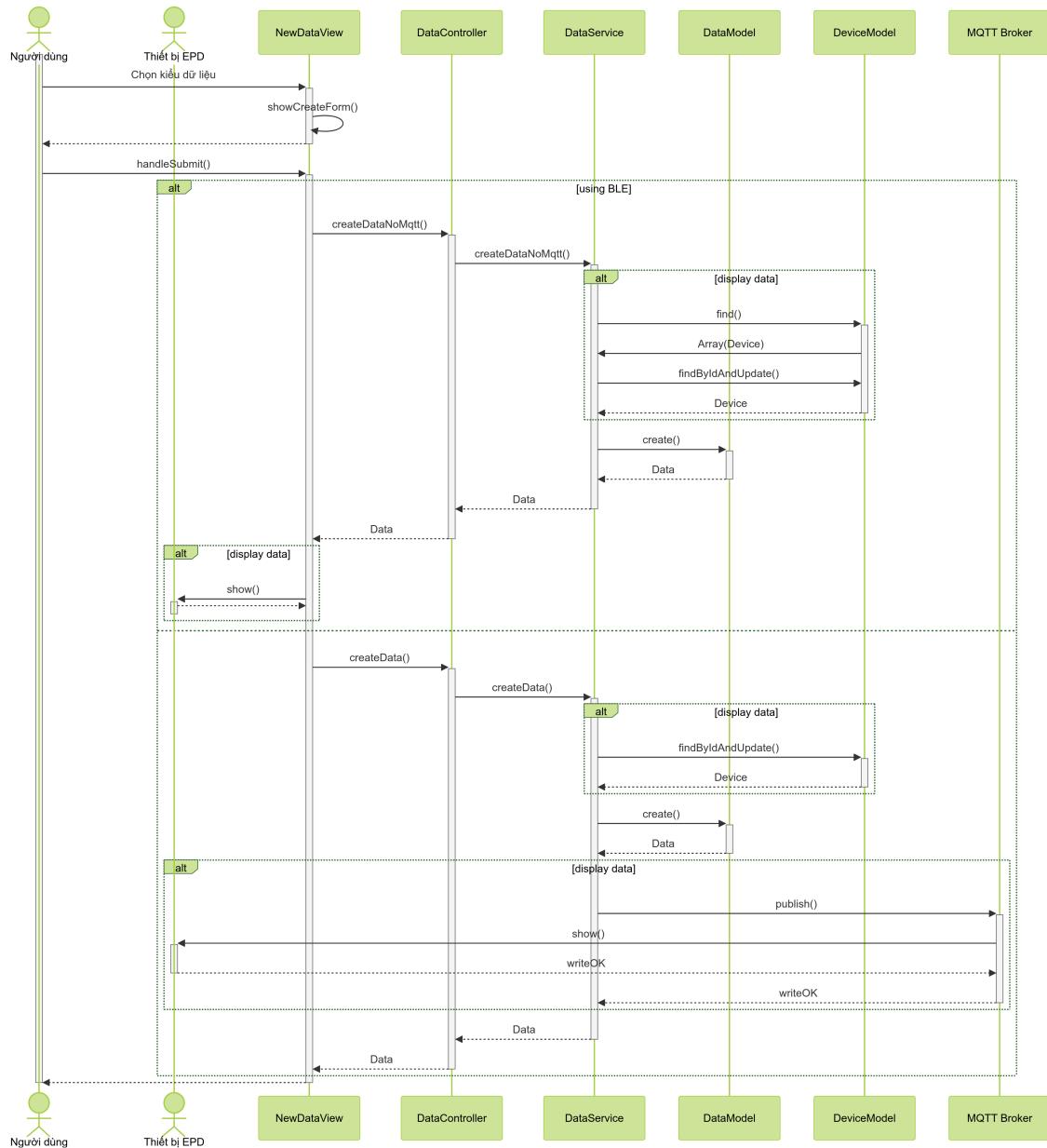
**Hình 4.19:** Biểu đồ trình tự cho ca sử dụng "Xóa thiết bị"

e, Biểu đồ lớp và biểu đồ trình tự cho ca sử dụng "Tạo dữ liệu"



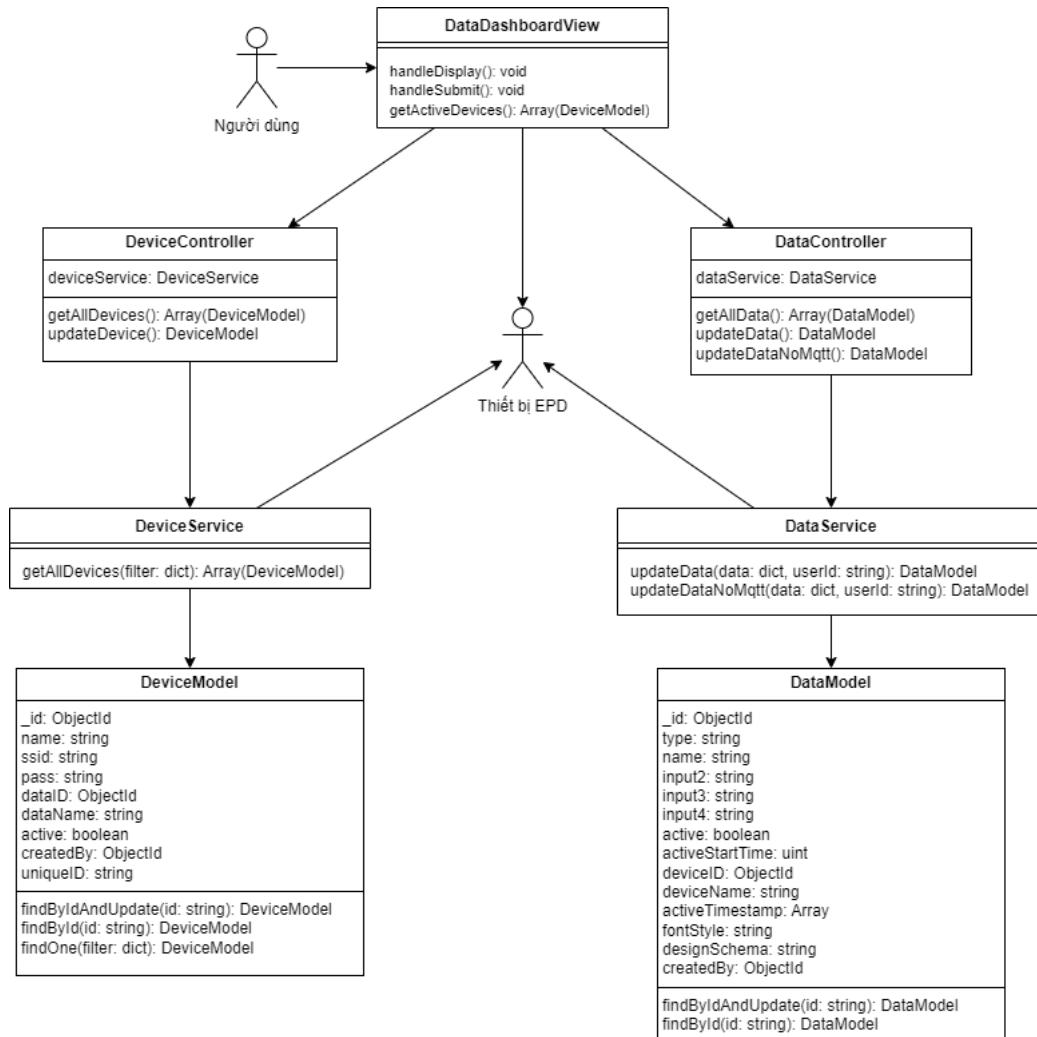
Hình 4.20: Biểu đồ lớp cho ca sử dụng "Tạo dữ liệu"

## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG



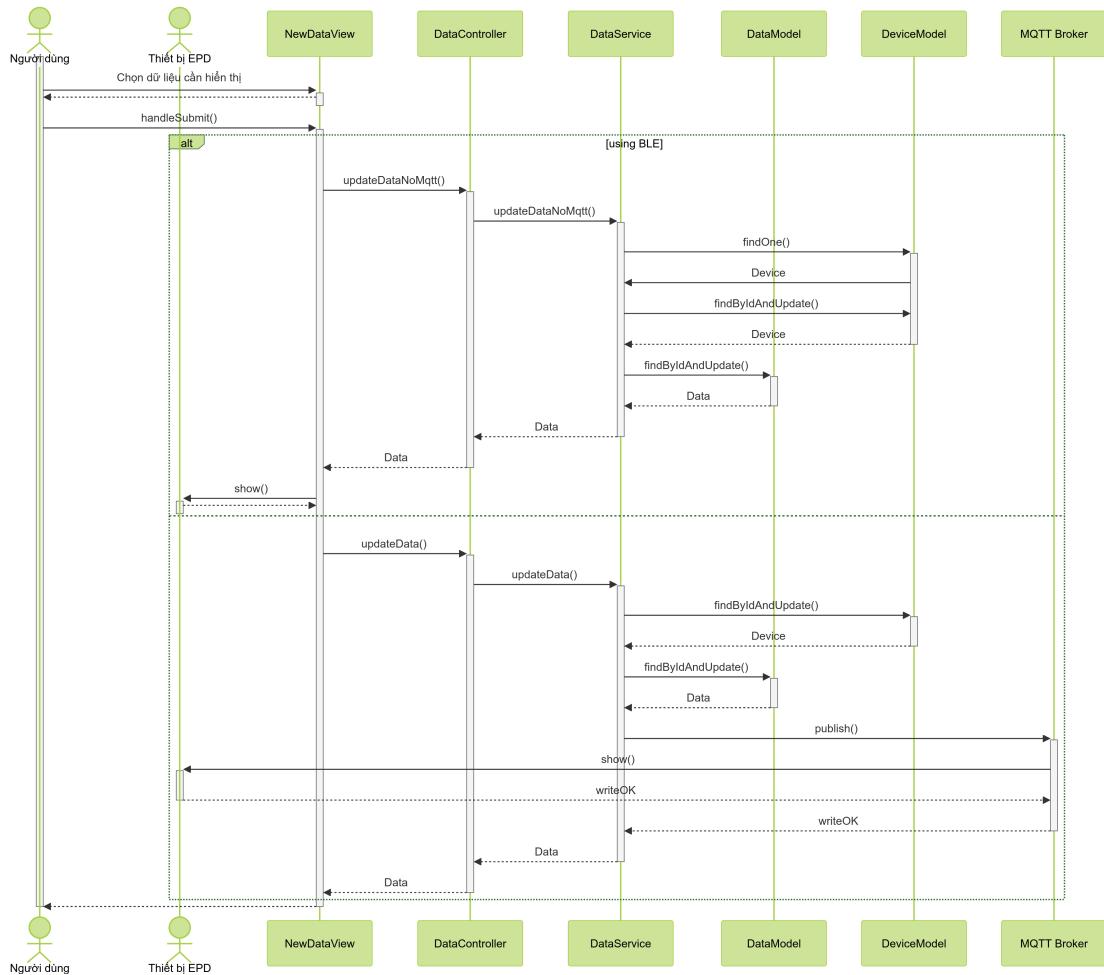
**Hình 4.21:** Biểu đồ trình tự cho ca sử dụng "Tạo dữ liệu"

f, Biểu đồ lớp và biểu đồ trình tự cho ca sử dụng "Hiển thị dữ liệu"



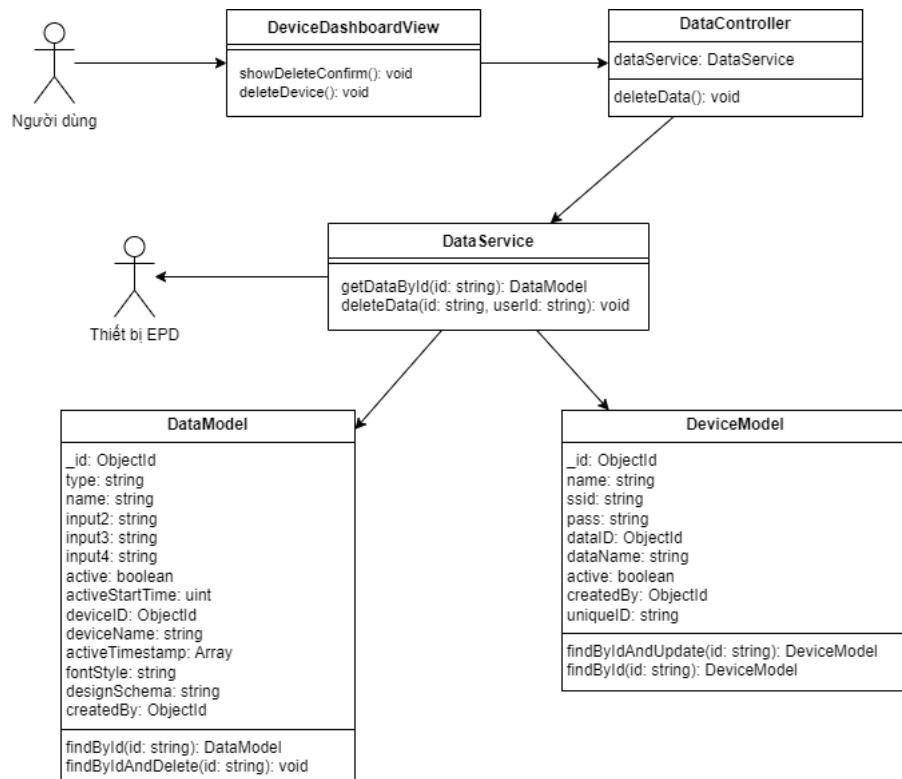
Hình 4.22: Biểu đồ lớp cho ca sử dụng "Hiển thị dữ liệu"

## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG



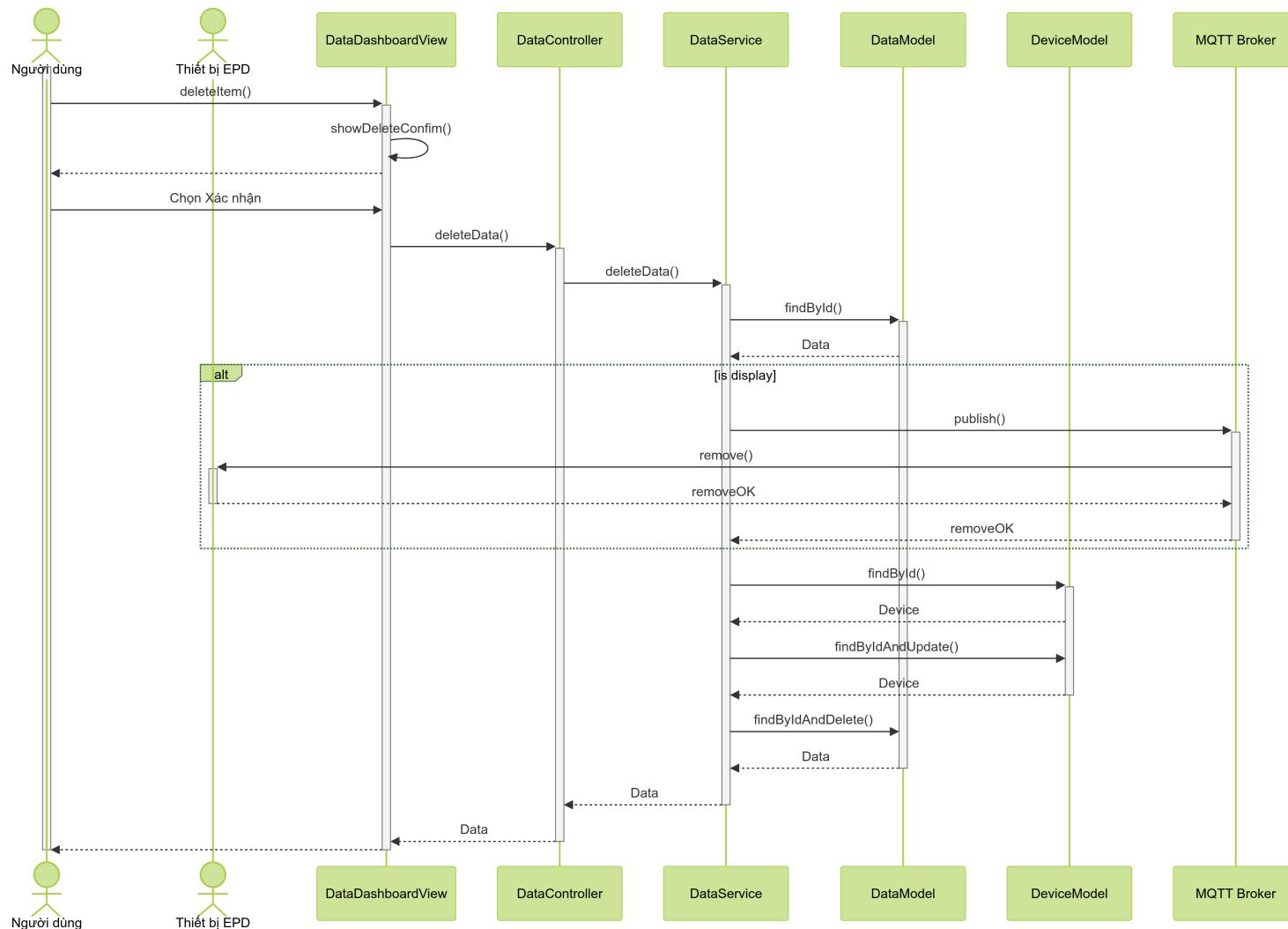
**Hình 4.23:** Biểu đồ trình tự cho ca sử dụng "Hiển thị dữ liệu"

**g, Biểu đồ lớp và biểu đồ trình tự cho ca sử dụng "Xóa dữ liệu"**



**Hình 4.24:** Biểu đồ lớp cho ca sử dụng "Xóa dữ liệu"

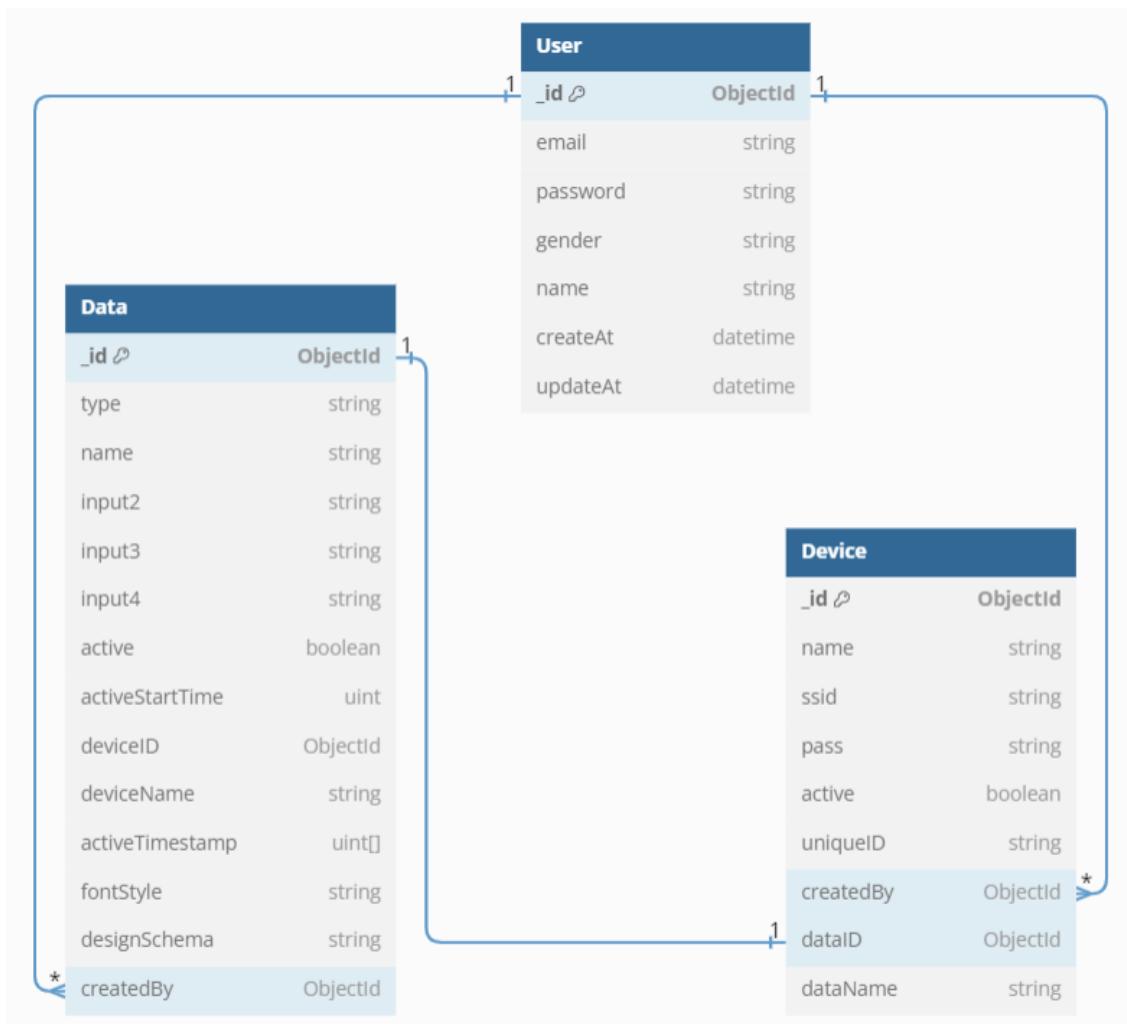
## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG



**Hình 4.25:** Biểu đồ trình tự cho ca sử dụng "Xóa dữ liệu"

### 4.2.3 Thiết kế cơ sở dữ liệu

#### a, Lược đồ cơ sở dữ liệu



**Hình 4.26:** Lược đồ cơ sở dữ liệu

**b, Thiết kế chi tiết cơ sở dữ liệu**

Collection	Tên trường	Kiểu dữ liệu	Bắt buộc	Mô tả
User	_id	ObjectId	*	ID của User
	email	string	*	Email
	password	string	*	Mật khẩu mã hóa
	name	string	*	User's name
	gender	string		Giới tính
	createdAt	datetime	*	Thời gian khởi tạo
	updatedAt	datetime	*	Thời gian cập nhật cuối
Data	_id	ObjectId	*	ID của Data
	type	string	*	Loại dữ liệu (Data)
	name	string	*	Tên dữ liệu
	input2	string		Thông tin thứ hai
	input3	string		Thông tin thứ ba
	input4	string		Thông tin thứ tư
	active	boolean	*	Trạng thái hiển thị trên thiết bị EPD
	activeStartTime	uint	*	Thời gian bắt đầu hiển thị
	deviceID	ObjectId		ID của thiết bị EPD
	deviceName	string		Tên của thiết bị EPD
	activeTimestamp	uint[]	*	Danh sách các thời gian hiển thị
	fontStyle	string		Phông chữ
	designSchema	string		Chủ đề thiết kế
Device	createdBy	ObjectId	*	ID của User tạo Data
	_id	ObjectId	*	ID của Device
	name	string		Tên thiết bị
	ssid	string	*	SSID của mạng đang kết nối
	pass	string	*	Mật khẩu của mạng đang kết nối
	dataID	ObjectId		ID của Data đang hiển thị
	dataName	string		Tên của dữ liệu
	active	boolean	*	Trạng thái kết nối
	uniqueID	string	*	ID riêng của thiết bị
	createdBy	ObjectId	*	ID của User tạo thiết bị

**Bảng 4.1:** Thiết kế chi tiết cơ sở dữ liệu

### 4.3 Xây dựng ứng dụng

#### 4.3.1 Thư viện và công cụ sử dụng

Danh sách các thư viện và công cụ được mô tả trong bảng 4.2 dưới đây.

Công cụ	Phiên bản	Mô tả	URL
Visual Studio Code (VSCode)	1.90.1	IDE lập trình chính	<a href="https://code.visualstudio.com">https://code.visualstudio.com</a>
PlatformIO	v6.1.15	Công cụ để lập trình hệ nhúng và ứng dụng IoT	<a href="https://platformio.org">https://platformio.org</a>
NodeJS	v21.5.0	Ngôn ngữ lập trình	<a href="https://nodejs.org">https://nodejs.org</a>
React Native	v0.73.6	Framework lập trình Mobile App	<a href="https://reactnative.dev/">https://reactnative.dev/</a>
Expo	v50.0.17	Công cụ mã nguồn mở hỗ trợ lập trình các ứng dụng React Native	<a href="https://expo.dev/">https://expo.dev/</a>
Redux	v50.0.17	Quản lý state trong ứng dụng React Native	<a href="https://redux.js.org/">https://redux.js.org/</a>
Next.JS	v13.4.5	Framework lập trình WebApp	<a href="https://nextjs.org">https://nextjs.org</a>
TailwindCSS	v3.3.2	Framework lập trình WebApp	<a href="https://tailwindcss.com">https://tailwindcss.com</a>
Express.JS	v4.18.2	Framework lập trình Backend	<a href="https://expressjs.com">https://expressjs.com</a>
Mongoose	v8.0.0	Thư viện mô hình hóa đối tượng cho MongoDB và Node.js	<a href="https://mongoosejs.com">https://mongoosejs.com</a>
MQTT NPM Package (mqtt)	v5.3.0	Thư viện giúp giao tiếp bằng giao thức MQTT cho Nodejs	<a href="https://www.npmjs.com/package/mqtt">https://www.npmjs.com/package/mqtt</a>
Node Version Manager (nvm)	v0.39.2	Công cụ giúp quản lý các phiên bản Nodejs	<a href="https://github.com/nvm-sh/nvm">https://github.com/nvm-sh/nvm</a>
Node Package Manager (npm)	v10.1.0	Công cụ tạo và quản lý các thư viện lập trình Javascript cho Node.js	<a href="https://www.npmjs.com">https://www.npmjs.com</a>
MongoDB Community Edition for Linux	v7.0.2	Cơ sở dữ liệu	<a href="https://www.mongodb.com/try/download/community">https://www.mongodb.com/try/download/community</a>
Mosquitto MQTT	v2.0.18	MQTT Broker mã nguồn mở	<a href="https://mosquitto.org/">https://mosquitto.org/</a>
ESP32 C3-Supermini		Module vi điều khiển nhỏ cho các thiết bị EPD	
WeAct Studio E-paper 2.9inch display		Module hiển thị năng lượng thấp	<a href="https://www.weact-tc.cn">https://www.weact-tc.cn</a>
Ubuntu Server		Server ảo (VPS)	<a href="https://nhanhoa.com/">https://nhanhoa.com/</a>
Nginx	nginx/1.18.0	Web Server mã nguồn mở	<a href="https://nginx.org">https://nginx.org</a>
GitHub		Quản lý mã nguồn	<a href="https://github.com">https://github.com</a>

**Bảng 4.2:** Danh sách các thư viện và công cụ sử dụng

### 4.3.2 Kết quả đạt được

Sau khi hoàn thành quá trình phát triển đồ án, bên cạnh nền tảng Web đã được phát triển ở phiên bản 1, ứng dụng hiện đã hoạt động ổn định trên nền tảng mới là Android cùng với giao diện dễ sử dụng.

So với phiên bản trước, một số thay đổi trong thiết kế cơ sở dữ liệu, thiết kế giao diện hiển thị của EPD và đặc biệt, xây dựng thêm các phương thức truyền dữ liệu và hiển thị mới. Hệ thống mới sẽ không cần bắt buộc phải sử dụng cổng USB mà còn có thể sử dụng Bluetooth để truyền dữ liệu. Đây là một trong những định hướng phát triển đã được nêu ra từ phiên bản trước đó. Trong phạm vi đồ án này, em sẽ chỉ thống kê những thông tin thay đổi và khác biệt so với phiên bản trước.

Về ứng dụng Mobile, ứng dụng được xây dựng trên nền tảng Android (có kích thước khoảng 40 MB) và đã có thể giao tiếp với hệ thống qua cả MQTT và Restful API. Tuy nhiên, khác biệt đó chính là không thể sử dụng cổng USB mà thay vào đó, em đã xây dựng thêm một cách giao tiếp khác dành cho thiết bị EPD và Mobile app đó chính là sử dụng công nghệ BLE. Chi tiết về ứng dụng của công nghệ BLE vào Mobile app được trình bày trong mục 5.3 của chương 5.

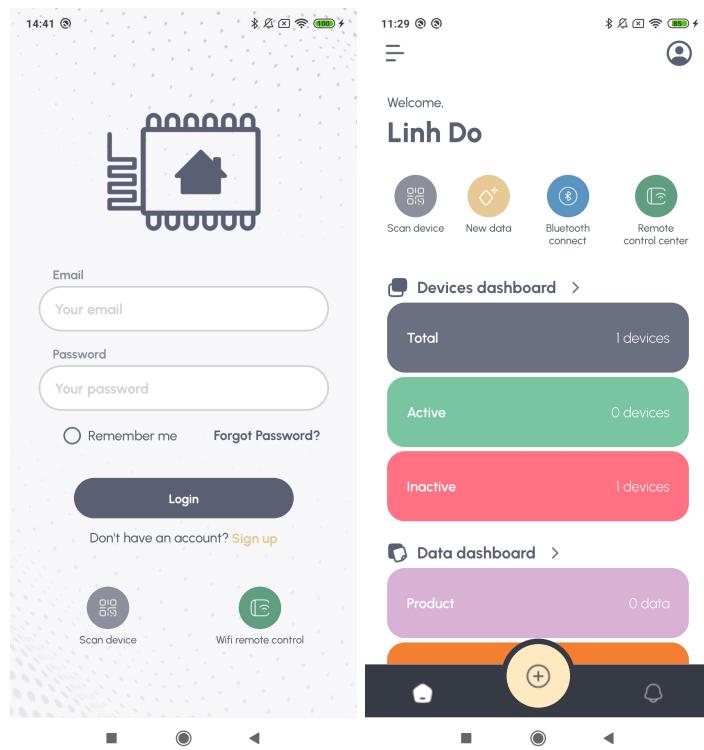
Về thiết bị EPD, mã nguồn đã được cải tiến thêm một số chức năng mới như sử dụng WiFi Adhoc, hiển thị dữ liệu ảnh và truyền nhận dữ liệu bằng BLE. Ngoài ra, em đã chỉnh sửa một số thay đổi nhằm tăng hiệu năng và giảm khả năng tiêu thụ pin. Chi tiết về một số cải tiến và thay đổi này sẽ được trình bày trong chương 5.

### 4.3.3 Minh họa các chức năng chính

Sau đây, em sẽ minh họa một số chức năng chính mà em đã xây dựng và phát triển trong đồ án:

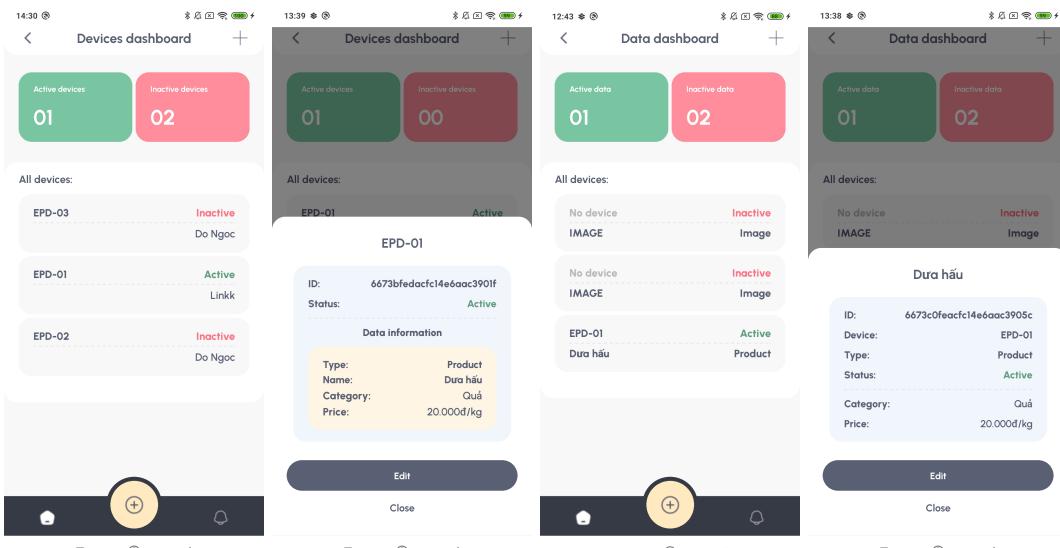
Đầu tiên, người dùng cần phải đăng nhập trước khi sử dụng. Tại màn hình đăng nhập, người dùng có thể truy cập vào màn hình "Scan QR" và "Wifi remote control"

## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG



**Hình 4.27:** Màn hình Đăng nhập và Màn hình chính

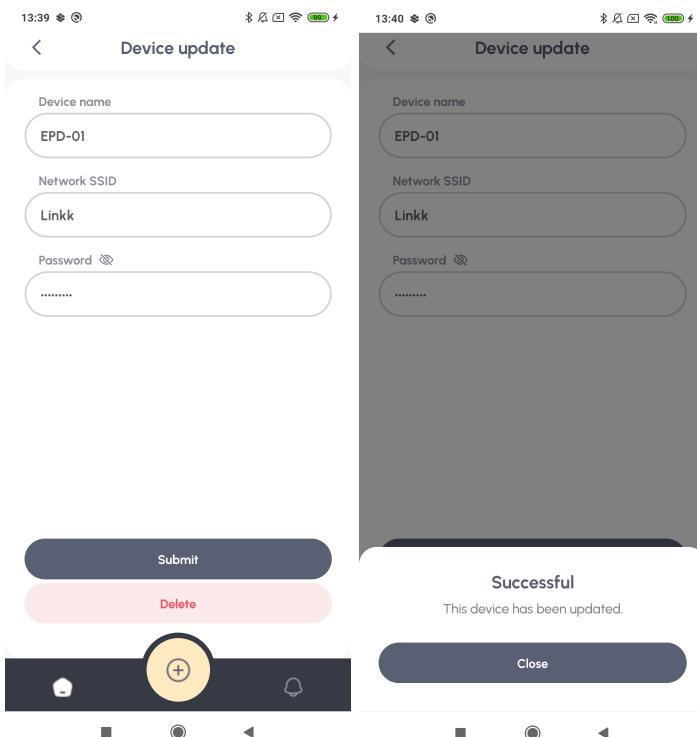
Tại màn hình chính, người dùng có thể truy cập vào xem danh sách thiết bị hoặc dữ liệu và xem chi tiết thông tin bằng cách bấm vào "Device dashboard" và "Data dashboard".



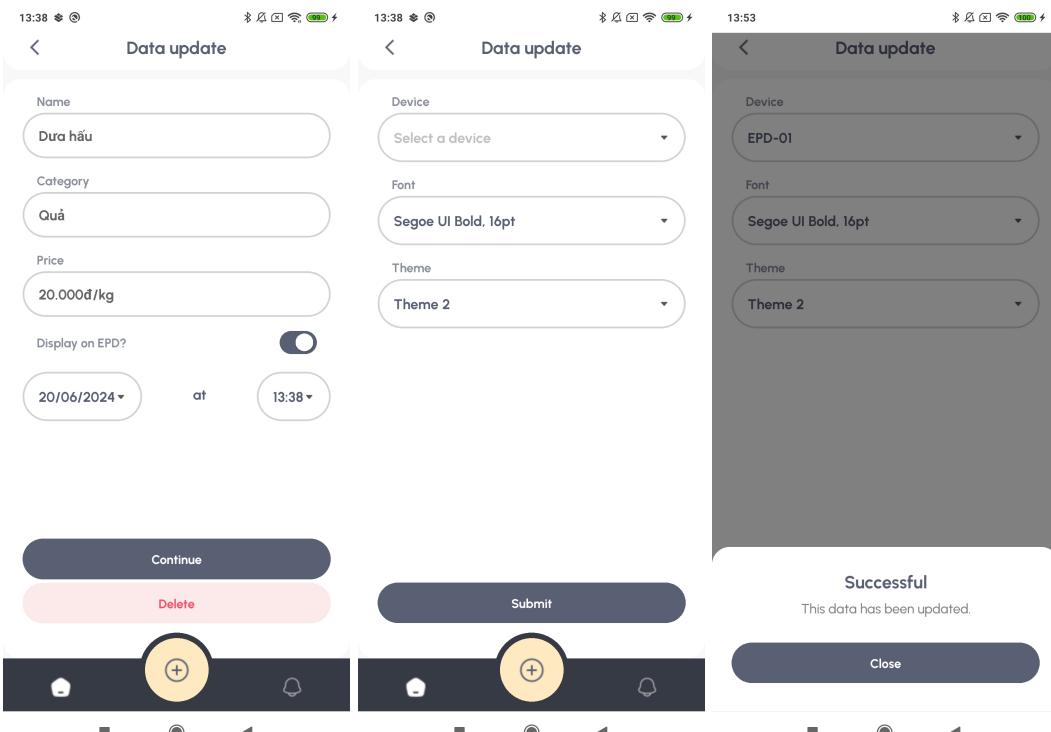
**Hình 4.28:** Màn hình "Devices dashboard" và Màn hình "Data dashboard"

Người dùng có thể bấm "Edit" để vào màn chỉnh sửa thông tin thiết bị ("Device update") và chỉnh sửa thông tin dữ liệu ("Data update"). Sau đó, điền đầy đủ thông tin và bấm "Submit".

## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG



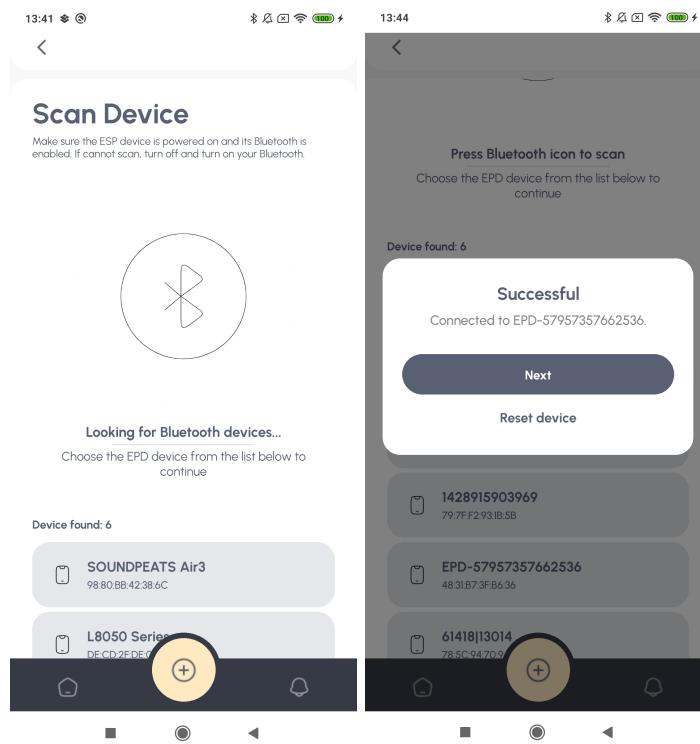
Hình 4.29: Màn hình "Device update"



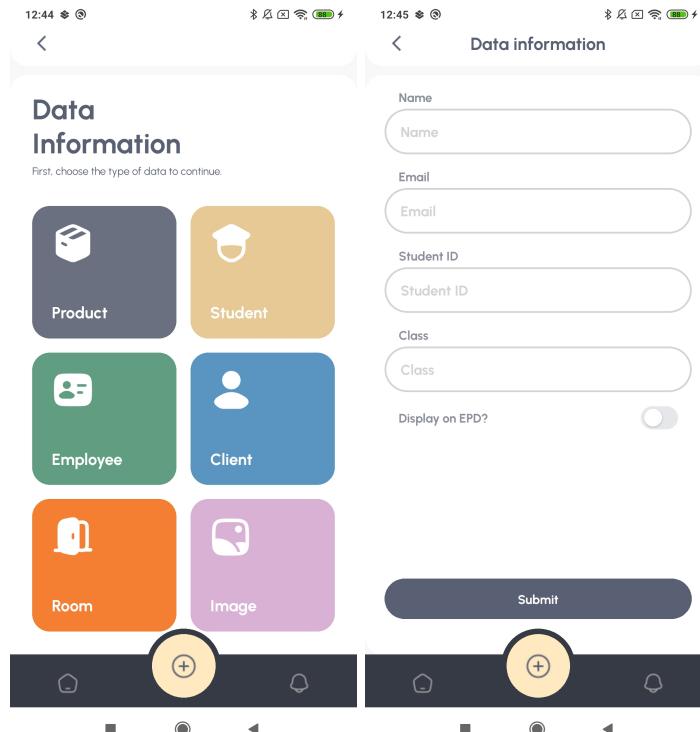
Hình 4.30: Màn hình "Data update"

Để tạo thiết bị và tạo dữ liệu mới người dùng có thể truy cập bằng thanh điều hướng ở dưới cùng của màn hình hoặc tại màn hình chính sử dụng các nút chức năng.

## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG



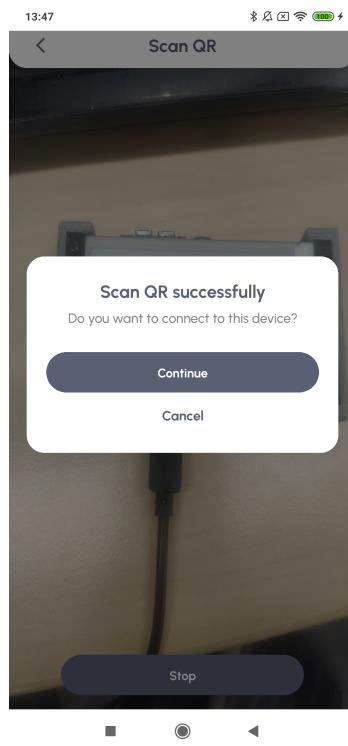
Hình 4.31: Màn hình "New Device"



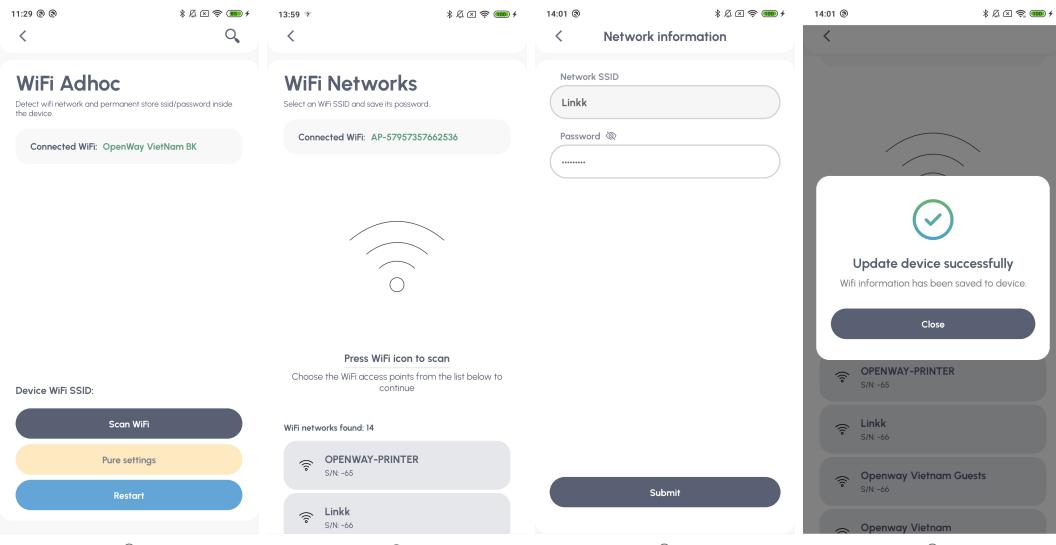
Hình 4.32: Màn hình "New Data"

Người dùng có thể truy cập màn "Scan QR" để quét mã QR chức năng hiển thị trên thiết bị EPD bao gồm Bluetooth và Wifi Adhoc. Dưới đây là minh họa cho chức năng sử dụng Wifi Adhoc.

## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG



**Hình 4.33:** Màn hình "Scan QR"



**Hình 4.34:** Các màn hình chức năng WiFi Adhoc

Thiết bị EPD sẽ hiển thị một số chức năng tùy chỉnh do người dùng lựa chọn bằng nút bấm.



**Hình 4.35:** Hiển thị chức năng của thiết bị EPD

Nếu thiết bị nhận được dữ liệu hoặc đang có dữ liệu thì nó sẽ hiển thị lên màn hình và lưu dữ liệu vào bộ nhớ Flash của ESP32. Sau đây là một trong những thiết kế hiển thị trên thiết bị EPD.



**Hình 4.36:** Hiển thị dữ liệu trên thiết bị EPD

#### 4.4 Kiểm thử

Đồ án sử dụng kỹ thuật kiểm thử hộp đen (Black box testing) để tiến hành kiểm thử trên toàn bộ hệ thống. Phần này sẽ trình bày kiểm thử các tính năng chính: thêm dữ liệu/thiết bị mới và xóa thiết bị/dữ liệu.

### **a, Kiểm thử chức năng "Tạo dữ liệu"**

Các đầu vào cần thiết của dữ liệu mới và phần chọn thiết bị trong quy trình này được mô tả trong bảng 4.3 dưới đây. Bảng 4.4 liệt kê các trường hợp kiểm thử đã được thực hiện và kết quả của chúng.

STT	Tên trường	Kiểu dữ liệu	Yêu cầu
1	Type	Text, Options	✓
2	Name	String, Input	✓
3	Email	String, Input	✗
4	Price	String, Input	✗
5	Department	String, Input	✗
6	Category	String, Input	✗
7	Student ID	String, Input	✗
8	Class	String, Input	✗
9	Employee ID	String, Input	✗
10	Address	String, Input	✗
11	Purpose	String, Input	✗
12	Manager	String, Input	✗
13	Status	String, Input	✗
14	Direction	String, Options	✓
15	Image data	String, Select	✓
16	Style	String, Options	✓
17	Active	Boolean, Check box	✓
18	Font Style	String, Options	Có nếu Active là true
19	Theme	String, Options	Có nếu Active là true
20	Device	String, Options	Có nếu Active là true

**Bảng 4.3:** Chi tiết các trường form "Tạo dữ liệu"

## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

STT	Test cases	Điều kiện	Kết quả mong đợi	Trạng thái
1	Người dùng điền đầy đủ thông tin	Tất các các trường được điền đầy đủ	Tạo dữ liệu mới thành công và hiển thị lên thiết bị EPD nếu Active là true	Đạt
2	Kiểm tra trường yêu cầu	Một số trường yêu cầu bị bỏ trống	Thông báo lỗi cho người dùng	Đạt
3	Dữ liệu ảnh	Tất các các trường không thiếu thông tin	Tạo dữ liệu ảnh thành công và hiển thị lên thiết bị EPD nếu Active là true	90%
3	Hiển thị lên thiết bị EPD	Thiết bị và dữ liệu hiển thị đúng	Dữ liệu hiển thị lên EPD đúng	Đạt
4	Không có thiết bị hoạt động	Tất cả thiết bị đều không hoạt động (inactive)	Thông báo lỗi yêu cầu người dùng kiểm tra lại thiết bị	Đạt
5	Xử lý dữ liệu cũ	Dữ liệu đang được hiển thị là dữ liệu cũ	Xóa dữ liệu cũ và hiển thị dữ liệu mới	Đạt
6	Lỗi thiết bị	Thiết bị lỗi khi nhận và hiển thị dữ liệu	Thông báo lỗi cho người dùng	Đạt

**Bảng 4.4:** Kết quả kiểm thử chức năng "Tạo dữ liệu"

### b, Kiểm thử chức năng "Xóa dữ liệu"

Kết quả kiểm thử chức năng "Xóa dữ liệu" được mô tả trong bảng 4.5 dưới đây.

STT	Test cases	Điều kiện	Kết quả mong đợi	Trạng thái
1	Xóa dữ liệu	Người dùng xác nhận xóa dữ liệu	Xóa dữ liệu khỏi hệ thống, cập nhật thông tin thiết bị đang hiển thị dữ liệu	Đạt
2	Xử lý lỗi thiết bị	Thiết bị lỗi khi xử lý yêu cầu	Xóa dữ liệu khỏi hệ thống và thông báo lỗi	Đạt

**Bảng 4.5:** Testing results of "Removing data" process

### c, Kiểm thử chức năng "Tạo thiết bị"

Các đầu vào cần thiết mà người dùng cần cung cấp để tạo một thiết bị mới được mô tả trong bảng 4.6 dưới đây.

## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

STT	Tên trường	Kiểu dữ liệu	Yêu cầu
1	Name	String, Input	✓
2	SSID	String, Input	✓
3	Password	String, Input	✓

**Bảng 4.6:** Chi tiết các trường form "Tạo thiết bị"

Kết quả kiểm thử được mô tả trong bảng 4.7 dưới đây.

STT	Test cases	Điều kiện	Kết quả mong đợi	Trạng thái
1	Người dùng điền đầy đủ thông tin	Tất cả các trường được điền đầy đủ	Thiết bị lưu thông tin và kết nối thành công tới Broker	Đạt
2	Kiểm tra trường yêu cầu	Một số trường yêu cầu bị bỏ trống	Thông báo lỗi cho người dùng	Đạt
3	Thiết bị kết nối lỗi	Thiết bị kết nối không thành công với Mobile qua Bluetooth	Thông báo lỗi cho người dùng	Đạt

**Bảng 4.7:** Kết quả kiểm thử chức năng "Tạo thiết bị"

### d, Kiểm thử chức năng "Xóa thiết bị"

Kết quả kiểm thử chức năng "Xóa thiết bị" được mô tả trong bảng 4.8 sau.

STT	Test cases	Điều kiện	Kết quả mong đợi	Trạng thái
1	Xóa thiết bị	Người dùng xác nhận xóa thiết bị	Xóa thiết bị khỏi hệ thống, cập nhật thông tin dữ liệu đang được thiết bị hiển thị	Đạt
2	Xử lý lỗi thiết bị	Thiết bị lỗi khi xử lý yêu cầu	Xóa thiết bị khỏi hệ thống và thông báo lỗi	Đạt

**Bảng 4.8:** Kết quả kiểm thử chức năng "Xóa thiết bị"

### e, Tổng kết

Hầu hết tất cả các trường hợp kiểm thử đều đạt như đã mô tả ở trên. Tuy nhiên, có duy nhất trường hợp tạo dữ liệu ảnh (bảng 4.4) là chưa đạt được 100%. Một số

lý giải có thể suy luận như sau:

- Do việc phân chia dữ liệu pixel ảnh thành nhiều khối (chunk) nên việc truyền ảnh giữa Mobile và thiết bị EPD có thể xảy ra hao hụt dữ liệu trong khối.
- Dữ liệu ảnh khá lớn điều này có thể dẫn đến tràn bộ nhớ của thiết bị EPD khiến cho dữ liệu ảnh hiển thị không đúng.
- Xử lý dữ liệu ảnh chưa chính xác tuyệt đối.

#### 4.5 Triển khai

Hệ thống đã được triển khai lên một máy chủ mới, bao gồm các thông tin như bảng 4.9.

Mô tả	URL
Webapp UI	<a href="https://epaper.toolhub.app">https://epaper.toolhub.app</a>
MongoDB Databases	<code>mongodb://mongo.epaper.toolhub.app:27017</code>
Back-end server	<a href="https://epaper.toolhub.app/api">https://epaper.toolhub.app/api</a>
MQTT Broker	<code>mqtts://mqtt.epaper.toolhub.app:1883</code>
OpenAPI Swagger	<a href="https://epaper.artsakh.ventures/api/swagger">https://epaper.artsakh.ventures/api/swagger</a>
Mobile app apk	<a href="https://epaper.artsakh.ventures/api/download-apk">https://epaper.artsakh.ventures/api/download-apk</a>

**Bảng 4.9:** Bảng thống kê thông tin triển khai hệ thống

##### a, Server

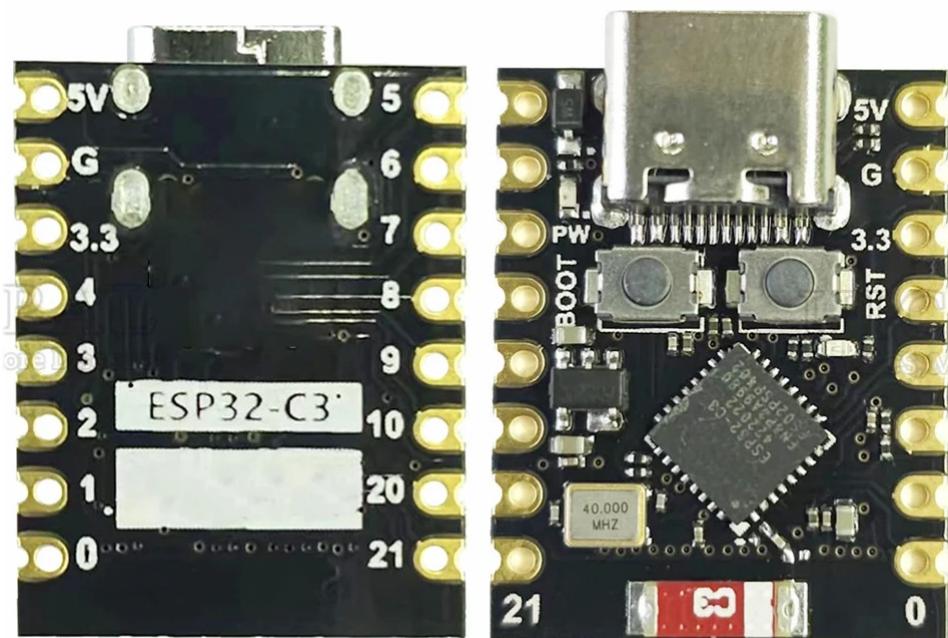
Cả API Server và MQTT Service đều được triển khai trên cùng một server. Thông tin cấu hình máy chủ được mô tả như bảng 4.10 sau.

<b>Chi tiết</b>	<b>Thông tin</b>
Địa chỉ IP	103.124.93.73
Hệ điều hành	Ubuntu 22.04.1 LTS x86_64
CPU	Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50GHz
RAM	1.5GB
Bộ nhớ	16GB
Services sử dụng	NginX, MongoDB, NextJS, NPM, Mosquito MQTT, PM2

**Bảng 4.10:** Chi tiết Server sử dụng

### **b, Thiết bị EPD**

Mã nguồn được chuyển và lưu trữ trong bộ nhớ flash của ESP32-C3 Supermini bằng cách sử dụng plugin PlatformIO tích hợp trong Visual Studio Code. Mã chiếm khoảng 1.5MB trong tổng bộ nhớ, hầu hết được lưu trữ trong bộ nhớ flash mà sẽ vẫn còn trên thiết bị ngay cả sau khi thiết bị bị tắt nguồn. Hình ảnh của ESP32-C3 Supermini và thông số kỹ thuật của đơn vị vi điều khiển ESP32-C3 được liệt kê trong hình 4.37 và bảng 4.11 dưới đây.



**Hình 4.37:** Hình ảnh của ESP32-C3 Supermini

Chi tiết	ESP32-C3 Supermini
Core	32-bit RISC-V single-core (SoC)
Maximum Clock Frequency	160MHz
RAM	400KB SRAM
Interfaces	GPIO, I2C, I2S, SPI, UART
Connectivities	BLE 5.0, Wi-Fi (2.4GHz)
Maximum transmit rate	150Mbps with Wi-Fi (2.4GHz), 2Mbps with BLE 5.0
Flash	4MB
ROM	384KB
Supply Voltage	3V - 3.6V

**Bảng 4.11:** Thông số kỹ thuật ESP32-C3 Supermini

Màn hình Epaper được kết nối và giao tiếp với ESP32-C3 Supermini thông qua các giao diện SPI. Chi tiết về các chân kết nối và mô tả của chúng được liệt kê

## CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

trong bảng 4.12 dưới đây.

Pin	ESP32-C3 Supermini	Mô tả
VCC	3V3	Power input (3.3V)
GND	GND	Ground
DIN	GPIO6	SPI MOSI pin, data input
SCLK	GPIO4	SPI CLK pin, clock signal input
CS	GPIO7	Chip selection, low active
DC	GPIO1	Data/command, low for commands, high for data
RST	GPIO2	Reset, low active
BUSY	GPIO3	Busy status output pin (means busy)

**Bảng 4.12:** Lược đồ kết nối SPI

## CHƯƠNG 5. CÁC GIẢI PHÁP VÀ ĐÓNG GÓP NỔI BẬT

Các chương trước của tài liệu này đã mô tả chi tiết kiến trúc và các công nghệ được sử dụng trong quá trình phát triển hệ thống. Tuy nhiên, nếu không có các giải pháp và ý tưởng rõ ràng, hệ thống có thể đã không có những nổi bật so với các hệ thống đã có và không phù hợp trong các môi trường khác nhau. Chương này nhằm làm rõ cách các vấn đề trong quá trình phát triển đồ án và những đổi mới đã đóng góp vào hệ thống so với phiên bản 1.

### 5.1 Hiển thị dữ liệu ảnh

#### 5.1.1 Đặt vấn đề

Đối với màn hình Epaper, ngoài việc hiển thị dữ liệu bằng dữ liệu cơ bản như văn bản, ký tự thì nó còn có thể hiển thị ảnh thông qua dữ liệu về các điểm ảnh. Ứng dụng của việc hiển thị dữ liệu ảnh có thể kể đến như hiển thị logo, hiển thị mã vạch, hiển thị mã QR code, v.v. Do vậy, để tận dụng tính năng này, trong phạm vi đồ án em đã xây dựng thêm tính năng hiển thị dữ liệu ảnh cho hệ thống. Đây cũng là một trong những điểm khác biệt của hệ thống so với phiên bản trước và hệ thống đã có trên thị trường.

#### 5.1.2 Giải pháp

##### a, Xử lý trên thiết bị EPD

Dữ liệu để hiển thị ảnh trên thiết bị EPD là dưới dạng mảng Byte. Trong đó, Số lượng điểm ảnh của màn WeAct-Epaper 2.9 Inch có kích thước  $128 \times 296$  là  $128 \times 296 = 37888(\text{bit})$  tương ứng với  $37888 \div 8 = 4736(\text{byte})$ . Mỗi Byte chứa thông tin của 8 điểm ảnh trên màn hình tương ứng với các bit. Do vậy dữ liệu gửi về từ các thiết bị cần phải được xử lý trước và chính xác.

Lý giải cho việc không thể xử lý trực tiếp dữ liệu trên thiết bị EPD là ESP32-C3 Supermini tuy rất mạnh nhưng xử lý một dữ liệu ảnh lớn là điều bất khả thi. Đặc biệt, kích thước dữ liệu ảnh gửi về từ người dùng không cố định và sẽ thường khá là lớn so với kích thước lớn nhất mà thiết bị hiển thị được (4736 bytes cho màn 2.9 inch). Do vậy, việc xử lý dữ liệu ảnh cần được thực hiện bởi các thiết bị mạnh mẽ hơn như PC, thiết bị di động, máy tính bảng, v.v.

Dữ liệu nhận được đã được mã hóa Base64 nhằm tránh mất mát trong quá trình gửi thông qua BLE hay MQTT Broker và xử lý. Từ đó, thiết bị EPD chỉ cần nhận và giải mã dữ liệu nhận được sau đó hiển thị lên trên màn hình.

### b, Xử lý trên thiết bị di động

Đối với thiết bị di động, dữ liệu ảnh được xử lý về dạng mảng Byte chứa giá trị của các điểm ảnh. Tuy nhiên, thiết bị EPD không thể hiển thị được ảnh màu ngoài đen - trắng nên cần phải sử dụng những thuật toán Dithering [20] đưa chúng về dạng ảnh dạng pixel đen hoặc trắng (1-bit Dithering).

Hình 5.1 mô tả trình tự xử lý dữ liệu ảnh trên Mobile.



**Hình 5.1:** Trình tự xử lý dữ liệu ảnh

Thuật toán Dithering sử dụng trong hệ thống bao gồm: Bayer dithering, Floyd – Steinberg dithering, Atkinson dithering. Mỗi thuật toán đều được ứng dụng trong từng trường hợp cụ thể khác nhau và được lựa chọn bởi người dùng.

### c, Kết quả đạt được

Dữ liệu ảnh đã được xử lý và hiển thị trên thiết bị EPD chính xác. Hình 5.2 dưới đây là một số hình ảnh của thiết bị EPD đang hiển thị ảnh.



**Hình 5.2:** Một số hiển thị dữ liệu ảnh trên thiết bị EPD

## 5.2 Chuyển các chế độ cho thiết bị EPD

### a, Đặt vấn đề

Thiết bị EPD trong phiên bản mới được nâng cấp và thêm vào tính năng BLE, WiFi Adhoc như đã đề cập ở các chương trước. Tuy nhiên, thiết bị chỉ cần duy trì một tính năng tại một thời điểm để tối ưu năng lượng sử dụng. Do vậy, thiết bị cần cung cấp khả năng chuyển đổi chế độ giúp người dùng có thể thực hiện thao tác này một cách dễ dàng.

### b, Giải pháp

Cách tiếp cận hiệu quả và đơn giản nhất đó chính là sử dụng nút bấm. Thiết bị đã được trang bị thêm hai nút bấm gắn với hai chân (pin) là 20 và 21. Tương ứng lần lượt sẽ là chuyển chế độ và bật/tắt hiển thị gõ lỗi trên EPD. Để tránh các lỗi trên bút bấm, thiết bị EPD được xây dựng thêm chương trình ngắn (interrupt) cho từng nút.

Thiết bị bao gồm ba chế độ được mô tả lần lượt như bảng 5.1 sau:

STT	Chế độ	Mô tả
1	Thường	Thiết bị chỉ kết nối vào mạng WiFi và giao tiếp qua Broker, cổng USB
2	Bluetooth Low Energy	Thiết bị sử dụng kết nối BLE, mạng WiFi và giao tiếp qua Broker, cổng USB
2	WiFi Adhoc	Thiết bị tạo WiFi Access Point và nhận thay đổi qua Web server

**Bảng 5.1:** Các chế độ trên thiết bị EPD

Đối với hiển thị gõ lỗi trên EPD, người dùng sẽ có thể thấy các thông tin mà thiết bị EPD đang thực hiện. Ví dụ như đang kết nối WiFi, đang kết nối tới Broker, hiển thị mã QR (BLE, WiFi Adhoc) của thiết bị EPD...



**Hình 5.3:** Hiển thị gõ lỗi trên thiết bị EPD

### **c, Kết quả đạt được**

Việc chuyển chế độ và bật/tắt hiển thị thực hiện chính xác trong tình huống người dùng không liên tục bấm các nút. Nếu người dùng thực hiện liên tục bấm các nút, chương trình trên thiết bị EPD tuy không bị lỗi nhưng sẽ xảy ra hiện tượng mất hiển thị hoặc hiển thị sai trên màn hình.

## **5.3 Sử dụng công nghệ Bluetooth Low Energy (BLE)**

### **5.3.1 Đặt vấn đề**

Hiện nay, sự phát triển của các ứng dụng IoT là vô cùng nhanh chóng và rộng rãi. Khi đó, các vấn đề về hiệu năng, năng lượng và đổi mới sẽ luôn là những điểm tạo nên sự khác biệt của một hệ thống so với trên thị trường. Chỉ cần một điểm khác biệt nhỏ cũng là tiền đề để đánh giá một hệ thống có ưu điểm tốt hơn một hệ thống khác. Vì vậy, như đã đề cập ở chương 3, đồ án sử dụng Công nghệ Bluetooth Low Energy thay vì Bluetooth Classic bởi những ưu điểm vượt trội của nó nhằm tạo ra một thiết bị có những khác biệt và nổi bật về hiệu năng, năng lượng so với các thiết bị khác. Để làm rõ hơn về sự khác biệt đó, đầu tiên chúng ta cần có cái nhìn tổng quan về ưu nhược điểm của BLE và Bluetooth Classic.

Tiêu chí	Bluetooth Low Energy	Bluetooth Classic
Tần số	2.4G	2.4G
Phạm vi hoạt động	100m (BLE 5.0)	Khoảng 10-100m
Tốc độ truyền dữ liệu	125Kbps – 2Mbps	1 – 3Mbps
MTU (Maximum Transmission Unit)	Lên tới 517 bytes	Lên tới 1021 bytes
Bảo mật	128-bit AES	64 bit, 128bit
Mức tiêu thụ năng lượng	Rất thấp	Thấp
Độ trễ	6ms	100ms
Công suất	0.01 – 0.5W	1W
Ứng dụng chủ yếu	IoT, cảm biến, thiết bị đeo, y tế	Truyền dữ liệu, âm thanh, thiết bị ngoại vi

**Bảng 5.2:** So sánh Bluetooth Low Energy và Bluetooth Classic

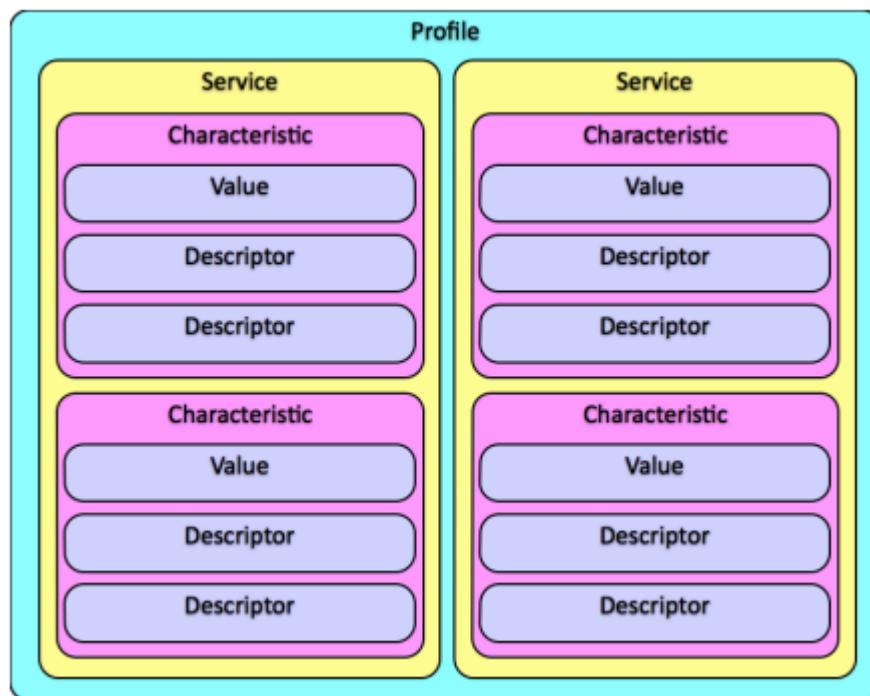
Từ so sánh trong bảng 5.2, có thể thấy việc sử dụng BLE là tối ưu hơn so với Bluetooth Classic dành cho các thiết bị IoT khi mà chúng không yêu cầu truyền nhiều dữ liệu lớn như ảnh, video, v.v. Tuy nhiên, trong hệ thống của đồ án, việc

hiển thị ảnh là cần thiết nên em vẫn phải có những thiết kế nhằm tận dụng những ưu điểm của BLE vào thiết bị mà không cần phải sử dụng đến cả Bluetooth Classic.

### 5.3.2 Giải pháp

#### a, Thiết kế truyền dữ liệu cho BLE

Generic Attribute Profile (GATT) là lớp có nhiệm vụ thiết lập chi tiết cách trao đổi tất cả profile và dữ liệu người dùng qua kết nối BLE. GATT sử dụng ATT và giao thức truyền của nó để trao đổi dữ liệu giữa các thiết bị. Dữ liệu truyền qua BLE là dữ liệu có cấu trúc được tổ chức phân cấp thành services và characteristics.



**Hình 5.4:** Cấu trúc GATT Profile

Do vậy, việc tổ chức, thiết kế các Service và Characteristic là rất quan trọng nhằm đảm bảo thông nhất giao tiếp giữa Mobile app và thiết bị EPD. Đối với thiết bị EPD, EPS32-C3 Supermini sẽ đóng vai trò là Peripheral device (server) nhằm định nghĩa các Services và Characteristics cũng như có thể quảng bá (advertise) cho các Central device (client) có thể quét và kết nối. Thiết bị di động sẽ đóng vai trò là các Central device.

Bảng 5.3 và bảng 5.4 dưới đây mô tả thiết kế Service và Characteristic cho thiết bị EPD.

## CHƯƠNG 5. CÁC GIẢI PHÁP VÀ ĐÓNG GÓP NỔI BẬT

Chi tiết	Device Service	Data Service
<b>UUID</b>	00001a10-0000-1000-8000-00805f9b34fb	00001a20-0000-1000-8000-00805f9b34fb
<b>Mô tả</b>	Bao gồm các Characteristics lưu trữ giá trị của thông tin Thiết bị	Bao gồm các Characteristics lưu trữ giá trị của thông tin Dữ liệu

**Bảng 5.3:** Thiết kế Service cho BLE

Service	Tên	UUID	Đọc	Ghi	Mã hóa
Device	ssid	00001a11-0000-1000-8000-00805f9b34fb	✓	✓	✓
	pass	00001a12-0000-1000-8000-00805f9b34fb	✓	✓	✓
	uniqueID	00001a13-0000-1000-8000-00805f9b34fb	✓	x	✓
	topicID	00001a14-0000-1000-8000-00805f9b34fb	✓	✓	✓
	restart	00001a1f-0000-1000-8000-00805f9b34fb	✓	✓	✓
Data	type	00001a2f-0000-1000-8000-00805f9b34fb	✓	✓	✓
	name	00001a21-0000-1000-8000-00805f9b34fb	✓	✓	✓
	input2	00001a22-0000-1000-8000-00805f9b34fb	✓	✓	✓
	input3	00001a23-0000-1000-8000-00805f9b34fb	✓	✓	✓
	input4	00001a24-0000-1000-8000-00805f9b34fb	✓	✓	✓
	image	00001a2b-0000-1000-8000-00805f9b34fb	✓	✓	✓
	font	00001a2d-0000-1000-8000-00805f9b34fb	✓	✓	✓
	schema	00001a2e-0000-1000-8000-00805f9b34fb	✓	✓	✓
	dataID	00001a2c-0000-1000-8000-00805f9b34fb	✓	✓	✓

**Bảng 5.4:** Thiết kế Characteristic cho các Service

Trong đó, hầu hết các characteristic tương đương với các trường được thiết kế

trong Cơ sở dữ liệu, khác biệt chỉ có 3 characteristic: topicID, restart và image.

- **topicID:** Lưu trữ giá trị của topic mà thiết bị subscribe tới MQTT Broker.
- **restart:** Lưu trữ giá trị của yêu cầu khởi động lại.
- **image:** Lưu trữ giá trị của các khối dữ liệu ảnh.

Việc thực hiện khởi tạo các Service và Characteristic như đã đề cập là trên thiết bị EPD. Chi tiết thực hiện trong hàm *BLE\_Init()* của mã nguồn nạp vào ESP32-C3 Supermini.

### b, Thiết kế truyền dữ liệu ảnh cho BLE

Đối với dữ liệu ảnh, kích thước lớn nhất dành cho màn hình WeAct-Epaper 2.9 Inch là 4736 bytes gấp khoảng 9 lần so với kích thước lớn nhất của BLE packet. Do vậy, dữ liệu ảnh cần phải được chia nhỏ và xử lý trước khi truyền.

Thiết kế ban đầu của các khối dữ liệu ảnh (chunk) bao gồm Payload và Flag được mô tả như sau:

- **Payload:** Tối đa 255 bytes - Chứa giá trị của một khối dữ liệu
- **Flag:** 1 bytes - Đánh dấu khối dữ liệu có phải là cuối cùng

Tuy nhiên, với cách tiếp cận này, nếu như truyền dữ liệu ở những trường hợp mất mát dữ liệu (ví dụ khoảng cách quá xa) sẽ không thể nhận biết. Dữ liệu ảnh nhận được sẽ bị sai lệch và hiển thị không chính xác trên màn hình.

Tiếp theo, một thiết kế khác dành cho các khối dữ liệu ảnh bao gồm Data length, Payload, Checksum và Flag:

Data length	Payload	Checksum	Flag
2 bytes	Tối đa 255 bytes	2bytes	1 byte

**Bảng 5.5:** Mô tả thiết kế khối dữ liệu ảnh

- **Data length:** 2 bytes - Kích thước của Payload.
- **Payload:** Tối đa 255 bytes - Chứa giá trị của một khối dữ liệu
- **Checksum:** 2 bytes - Kiểm tra lỗi
- **Flag:** 1 bytes - Đánh dấu khối dữ liệu có phải là cuối cùng

Hầu hết dữ liệu ảnh đã được nhận và hiển thị chính xác hơn so với thiết kế cũ. Tuy nhiên, một số tình huống vẫn xảy ra hiện tượng hiển thị dữ liệu ảnh không chính xác.

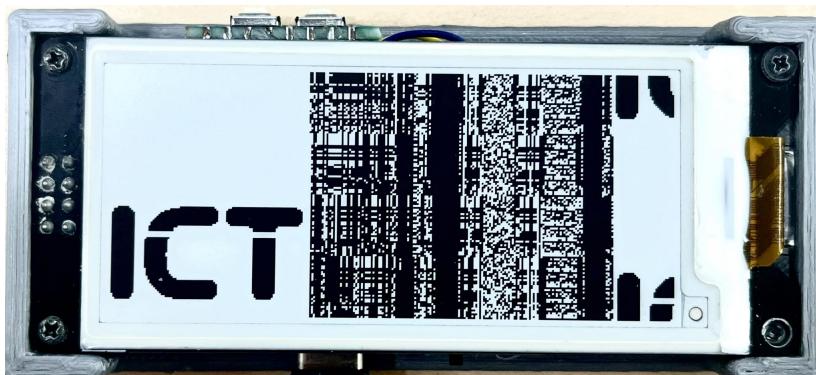
### 5.3.3 Kết quả đạt được

Thiết bị EPD đã có thể nhận và hiển thị dữ liệu ảnh một cách chính xác nhưng trong một số tình huống vẫn xảy ra hiện tượng hiển thị dữ liệu không chính xác. Dưới đây là hình ảnh của thiết bị EPD khi hiển thị dữ liệu chính xác và hiển thị dữ liệu bị lỗi.

Tuy nhiên, theo thử nghiệm và thống kê của em, xác xuất hiển thị sai của dữ liệu khá là thấp (khoảng 10%) và đã có cách khắc phục tạm thời. Người dùng chỉ cần gửi lại dữ liệu ảnh bị lỗi một lần nữa là kết quả hiển thị vẫn chính xác. Lý giải cho tình huống này có thể là do tràn bộ nhớ của thiết bị EPD khi liên tục gửi dữ liệu lớn qua BLE hoặc MQTT.



**Hình 5.5:** Hiển thị dữ liệu ảnh trên EPD



**Hình 5.6:** Hiển thị dữ liệu ảnh bị lỗi trên EPD

## 5.4 Sử dụng WiFi Adhoc cho thiết bị EPD

### 5.4.1 Đặt vấn đề

Khi một thiết bị đã đem vào sử dụng, các thiết bị không thể tránh khỏi việc bị mất kết nối mạng hoặc thay đổi môi trường sử dụng. Vì vậy, thiết bị cần phải có một tính năng cho phép người dùng không cần phải trực tiếp thực hiện trên thiết bị EPD như chuyển chế độ, cấu hình lại thiết bị, v.v là rất cần thiết nhằm tiết kiệm

thời gian và hạn chế việc gặp lỗi trong quá trình sử dụng thiết bị.

### 5.4.2 Giải pháp

Ngoài việc sử dụng tính năng BLE, thiết bị EPD đã được cải tiến, phát triển thêm tính năng phát WiFi Adhoc cho thiết bị EPD. Thiết bị EPD khi không kết nối được mạng sẽ đóng vai trò như một WiFi Access point. Các thiết bị xung quanh có khả năng kết nối WiFi sẽ có thể dò được thiết bị EPD và kết nối tới mạng WiFi của nó.

Để thay đổi thông tin mạng kết nối trên thiết bị EPD, em đã xây dựng một Web server trên ESP32-C3 Supermini nhằm lắng nghe những yêu cầu từ người dùng gửi đến. Một số tính năng của Web server như sau:

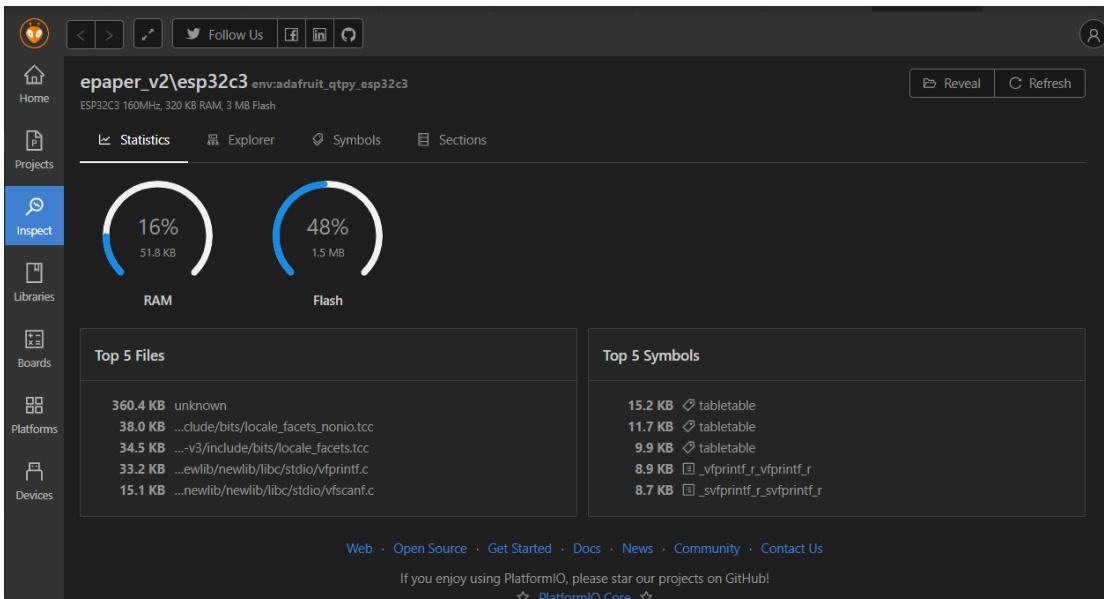
Entry point	Mô tả
http://192.168.15.1/enroll	Truy cập trang chủ
http://192.168.15.1/cgi/scan	Quét các mạng WiFi và trả về thông tin bao gồm SSID và RSSI
http://192.168.15.1/cgi/save?s=ssid&p=password	Lưu thông tin mạng vào thiết bị, ssid là SSID của mạng, password là mật khẩu của mạng do người dùng nhập
http://192.168.15.1/cgi/save?clear=true	Xóa thông tin mạng trong bộ nhớ
http://192.168.15.1/cgi/settings	Lấy thông tin mạng trong bộ nhớ
http://192.168.15.1/restart	Khởi động lại thiết bị

**Bảng 5.6:** Tính năng của Web server trên thiết bị EPD

## 5.5 Một số khó khăn gặp phải

### 5.5.1 PlatformIO

Trong phiên bản trước, kích thước của mã nguồn trên thiết bị EPD chưa đủ lớn. Do vậy, trong quá trình phát triển không gặp phải bất kỳ vấn đề nào với việc thực thi và tải lên chương trình vào ESP32. Tuy nhiên, sau khi thực hiện đồ án, kích thước mã chương trình khá lớn (1.5 MB) nên đã xảy ra vấn đề khi tải chương trình vào ESP32.



**Hình 5.7:** Thông tin mã nguồn từ PlatformIO

```
Checking size .pio\build\adafruit_qtpy_esp32c3\firmware.elf
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM: [==          ] 16.1% (used 52676 bytes from 327680 bytes)
Flash: [==Error: The program size (1426868 bytes) is greater than maximum allowed (1310720 bytes)
====** [checkprogsz] Explicit exit, status 1
=====] 108.9% (used 1426868 bytes from 1310720 bytes)
===== [FAILED] Took 14.70 seconds =====
Environment      Status      Duration
```

**Hình 5.8:** Lỗi khi tải chương trình vào ESP32

Nguyên nhân là do cấu hình mặc định của PlatformIO chỉ dành tối đa 1 MB cho kích thước mã nguồn. Vì vậy, để xây dựng một mã nguồn lớn hơn cần thay đổi cấu hình trong file *platformio.ini* bằng cách thêm thuộc tính [21]:

```
board_build.partitions = huge_app.csv
```

### 5.5.2 Bluetooth trên thiết bị di động

Trong quá trình thực hiện đồ án, em sử dụng thiết bị **Xiaomi Mi 9T** để xây dựng Mobile app. Tuy nhiên, quá trình phát triển đã xuất hiện một vấn đề gấp phải khi sử dụng Bluetooth kết nối với thiết bị EPD. Người dùng thi thoảng sẽ không thể quét được các thiết bị bằng Bluetooth của thiết bị di động. Người dùng sẽ phải khởi động lại Bluetooth trên thiết bị để có thể quét được các thiết bị đang phát Bluetooth. Hiện tại, vấn đề này em vẫn chưa có giải pháp cũng như chưa tìm được rõ nguyên nhân.

### 5.5.3 Sử dụng và kết nối WiFi

#### a, Cấu hình WiFi Adhoc cho ESP32

Đối với thiết bị EPD, việc phát tín hiệu WiFi ra một khoảng cách xa gấp rất nhiều vần đề. Trong quá trình phát triển tính năng, em đã gặp phải vấn đề khi không thể duy trì kết nối hoặc không thể kết nối đối với thiết bị phát WiFi (ESP32). Việc kết nối tới các thiết bị này mất một khoảng thời gian rất lâu.

Nguyên nhân chủ yếu là do cấu hình mặc định của việc phát tín hiệu WiFi là khá xa và tốn nhiều năng lượng trong khi thiết bị EPD không thể đảm bảo duy trì được việc phát tín hiệu đó. Hơn nữa, việc phát tín hiệu WiFi đi xa hiện tại không phù hợp với tính năng này vì nó chỉ là giải pháp nhanh chóng so với các tính năng chính. Do vậy, ESP32 cần được cấu hình công suất phát tín hiệu (Tx Power) giảm xuống cho phù hợp.

#### b, Kết nối mạng WiFi cho ESP32

Trong tình huống đột ngột mất mạng hoặc mất nguồn cấp năng lượng, thiết bị EPD không thể kết nối đến mạng WiFi kể cả sau khi mạng đã phục hồi và có thể truy cập mạng từ thiết bị khác như thiết bị di động, PC, v.v. Một trong những khắc phục đó chính là cần phải gọi hàm ngắt kết nối trước khi bắt đầu một kết nối mới đến mạng WiFi.

```
WiFi.mode(WIFI_STA);  
WiFi.disconnect();  
delay(10);  
WiFi.begin(ssid, password);
```

## CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

### 6.1 Kết luận

Đồ án "Xây dựng hệ thống quản lý thông tin tập trung trên giấy điện tử phiên bản 2" không chỉ cung cấp giải pháp cho các vấn đề mà còn mang lại những hiểu biết và giải pháp có giá trị. Trong đồ án văn này, quá trình phát triển cùng với các quy trình thiết kế và triển khai cũng được trình bày chi tiết. Mặc dù đồ án đã xây dựng và cải thiện hệ thống EPD trước đó với các chức năng cốt lõi, nhưng vẫn còn những thiếu sót và hạn chế cần khắc phục. Đồng thời, đồ án cũng đề xuất những hướng phát triển tiềm năng trong tương lai nhằm nâng cao hiệu quả và ứng dụng của hệ thống vào trong thực tiễn.

#### 6.1.1 Kết quả đã đạt được

Nhờ sự hướng dẫn của ThS. Nguyễn Đức Tiên, đồ án đã đạt được các mục tiêu và yêu cầu đã trình bày trong các chương trước. Đầu tiên, đồ án đã nghiên cứu nhu cầu hiển thị hiệu quả trong nhiều môi trường kinh doanh khác nhau, cùng với việc xem xét bốn nhà sản xuất EPD hàng đầu là E Ink, Inkcase, Pervasive Displays và Zkong, đánh giá sự độc đáo và điểm yếu của họ cũng như tham khảo đồ án "Xây dựng bảng thông tin sử dụng giấy điện tử ePaperboard" của tác giả Vũ Lê Nhật Minh để xác định các tính năng quan trọng và các cải tiến, chỉnh sửa cần thiết của các thiết bị EPD và hệ thống quản lý. Thứ hai, đồ án đã đi sâu vào phân tích các yêu cầu chức năng và phi chức năng, từ đó thiết kế chi tiết hệ thống quản lý. Từ những thiết kế này, đồ án đã phát triển các thiết bị EPD đáp ứng đầy đủ các yêu cầu và giao diện quản lý cho người dùng để quản lý thiết bị và dữ liệu trên cả nền tảng Web và Android. Hệ thống cũng đã được kiểm tra trong nhiều môi trường ví dụ và chứng minh tính hữu ích trong nhiều trường hợp sử dụng khác nhau.

Một trong những đóng góp quan trọng của đồ án này là ứng dụng công nghệ BLE để trao đổi dữ liệu và phát triển tính năng hiển thị hình ảnh. Điều này không chỉ giúp nâng cao hiểu biết về việc hiển thị trên bảng điện tử EPD và quản lý tài nguyên trong ESP32-C3 mà còn mang lại các ứng dụng thực tiễn trong việc tùy chỉnh hiển thị trên các thiết bị EPD cho người dùng.

#### 6.1.2 Hạn chế và thiếu sót

Mặc dù đã đạt được nhiều thành tựu, đồ án vẫn còn một số thiếu sót và các vấn đề chưa được giải quyết cần phải khắc phục. Thứ nhất, tuổi thọ pin của các thiết bị EPD vẫn còn kém ngay cả sau khi tối ưu hóa, do dung lượng hạn chế của pin. ESP32-C3 luôn hoạt động để duy trì kết nối với WiFi và BLE. Thứ hai, hệ thống

hiện đang ở giai đoạn nguyên mẫu, điều đó có nghĩa là nó chỉ phù hợp cho các trường hợp sử dụng quy mô nhỏ. Tuy nhiên, trong một môi trường kinh doanh thực tế, hệ thống sẽ cần phải được cải thiện đáng kể để đáp ứng nhu cầu rộng rãi của doanh nghiệp. Tiếp theo, giao diện của thiết bị di động chỉ mới được phát triển trên nền tảng Android. Cuối cùng, việc xử lý truyền dữ liệu ảnh giữa Mobile App và thiết bị EPD chưa thực sự hoàn chỉnh.

### 6.2 Hướng phát triển

Hiện tại, đồ án đã gần như hoàn thiện hệ thống đã thiết kế và xây dựng. Tuy nhiên, một số tính năng cần được chỉnh sửa và cải thiện hoàn chỉnh hơn.

Đầu tiên, hệ thống cần được chỉnh sửa chức năng hiển thị ảnh. Như đã đưa ra một số lý giải trong mục 4.4 của chương 4, chức năng này cần được phân tích rõ hơn và đưa ra các giải pháp thay thế để đạt được kết quả tốt nhất.

Thứ hai, thiết bị EPD cần được thiết kế lại để tối ưu hóa tuổi thọ pin và tăng tính thẩm mỹ. Hơn nữa, ESP32-C3 Supermini cần được tối ưu tiết kiệm năng lượng hơn trong khi vẫn đảm bảo hiệu suất. Cuối cùng, hệ thống cần cải thiện để đáp ứng các loại thiết bị EPD khác nhau, chẳng hạn như các kích thước và màu sắc khác nhau, để cung cấp sự đa dạng thiết bị trong hệ thống.

Thứ ba, người dùng trong hệ thống cần có một hệ thống phân cấp và quyền hạn rõ ràng hơn. Một giải pháp phi tập trung cụ thể mà hệ thống dự định phát triển trong tương lai là tạo ra người dùng quản trị viên (Admin) chịu trách nhiệm quản lý hệ thống. Người dùng loại này có thể quản lý máy chủ MongoDB và MQTT Broker của riêng họ với các tài khoản tùy chỉnh. Họ cũng có thể quản lý nhiều người dùng khác và phân quyền cho từng người dùng. Hơn nữa, họ có thể kiểm soát các thiết bị và dữ liệu của tất cả người dùng được quản lý nhưng không thể thêm, xóa hoặc sửa đổi chúng. Giải pháp phân cấp này rất hữu ích trong các doanh nghiệp lớn khi một công ty cần xử lý nhiều bộ phận con và dữ liệu, và một cách hiệu quả để quản lý người dùng và dữ liệu là rất quan trọng.

Thứ tư, giao diện người dùng cần được tối ưu hóa cho nhiều kích thước màn hình và thiết bị. Đặc biệt, hệ thống cần phát triển ứng dụng di động dành cho nền tảng IOS.

Cuối cùng, hệ thống cần cải thiện Web app và Backend. Đối với Web app, những tính năng mới được xây dựng trên Mobile app cần được nghiên cứu và phát triển. Đối với Backend, hệ thống cần tối ưu và sử dụng những công nghệ được hỗ trợ rộng rãi hơn nhằm tăng khả năng mở rộng cho hệ thống.

Trên đây là toàn bộ nội dung của đồ án "Xây dựng hệ thống quản lý thông tin

## CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

tập trung trên giấy điện tử phiên bản 2". Đồ án đã mang lại những hiểu biết sâu sắc về ứng dụng thực tiễn của màn hình điện tử trong nhiều tình huống khác nhau. Với thời gian, kỹ năng và kinh nghiệm hạn chế, em không thể tránh khỏi những thiếu sót trong quá trình phát triển đồ án. Em rất mong nhận được sự hướng dẫn và góp ý quý báu từ các thầy cô để hệ thống có thể ngày càng hoàn thiện và phát triển hơn trong tương lai.

## TÀI LIỆU THAM KHẢO

- [1] E Ink Company, *About*. [Online]. Available: <https://www.eink.com/about> (visited on 06/07/2024).
- [2] E-Ink, *E-ink: Multiple applications*. [Online]. Available: <https://www.eink.com/application> (visited on 06/07/2024).
- [3] InkCase, *Inkcase*. [Online]. Available: <https://inkcase.com> (visited on 06/07/2024).
- [4] A. Wang, *Pervasive displays - putting quality first in e-paper display design manufacture*. [Online]. Available: <https://www.epdtonthenet.net/article/183797/Putting-quality-first-in-e-paper-display-design-manufacture.aspx> (visited on 06/07/2024).
- [5] ZKONG Network Co., Ltd., *About*. [Online]. Available: <https://www.zkong.com/company-profile/> (visited on 06/07/2024).
- [6] NextJS, *Next.js by vercel - the react framework*. [Online]. Available: <https://nextjs.org/> (visited on 06/07/2024).
- [7] A. Trivedi, *Next.js vs. react: Guide to framework selection*. [Online]. Available: <https://dzone.com/articles/nextjs-vs-react-the-ultimate-guide-to-choosing-the> (visited on 06/07/2024).
- [8] T. Romesh, *Why choose next.js - top 5 performance benefits*. [Online]. Available: <https://cult.honeypot.io/reads/top-nextjs-performance-benefits/> (visited on 06/07/2024).
- [9] F. U. Blog, *Tailwind css: A utility-first framework for faster development*. [Online]. Available: <https://floatui.com/blog/tailwind-css-a-utility-first-framework-for-faster-development> (visited on 06/07/2024).
- [10] T. CSS, *Optimizing for production*. [Online]. Available: <https://tailwindcss.com/docs/optimizing-for-production> (visited on 06/07/2024).
- [11] S. Willows, *Css-in-js to tailwind: 36% better web vitals*. [Online]. Available: <https://sophiabits.com/blog/css-in-js-to-tailwind-better-web-vitals> (visited on 06/07/2024).
- [12] R. Native, *Docs*. [Online]. Available: <https://reactnative.dev/> (visited on 06/07/2024).
- [13] R. Native, *Threading model*. [Online]. Available: <https://reactnative.dev/architecture/threading-model> (visited on 06/07/2024).
- [14] Redux, *Docs*. [Online]. Available: <https://redux.js.org/> (visited on 06/07/2024).

- [15] MongoDB, *Docs*. [Online]. Available: <https://www.mongodb.com/> (visited on 06/07/2024).
- [16] MongoDB, *Relational vs. non-relational databases*. [Online]. Available: <https://www.mongodb.com/resources/compare/relational-vs-non-relational-databases> (visited on 06/07/2024).
- [17] M. MQTT, *About*. [Online]. Available: <https://mosquitto.org/> (visited on 06/07/2024).
- [18] E. Systems, *ESP32-C3 Series Datasheet*. Espressif Systems, 2023.
- [19] Quintagroup, *Platformio - open source ecosystem for iot development*. [Online]. Available: <https://quintagroup.com/cms/python/platformio> (visited on 06/07/2024).
- [20] H. coding academic, *Image dithering — how to deceive your eyes?* [Online]. Available: <https://hbyacademic.medium.com/how-to-deceive-your-eyes-image-dithering-172966d372b0> (visited on 06/07/2024).
- [21] PlatformIO, *Partition tables*. [Online]. Available: <https://docs.platformio.org/en/latest/platforms/espressif32.html#partition-tables> (visited on 06/07/2024).