

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH



BÁO CÁO

ĐỒ ÁN TỐT NGHIỆP KỸ THUẬT MÁY TÍNH

**HIỆN THỰC ỨNG DỤNG KHÁM SỨC KHỎE
DỰA TRÊN NỀN TẢNG KẾT NỐI VẠN VẬT**

NGÀNH KỸ THUẬT MÁY TÍNH

HỘI ĐỒNG: : HỘI ĐỒNG 3 KỸ THUẬT MÁY TÍNH

GVHD : TS. Lê Trọng Nhân

TKHD : Thầy Nguyễn Thiên Ân

—o0o—

SVTH1 : Bùi Quang Bằng - 2012681

SVTH2 : Nguyễn Tuấn Hùng - 2111386

SVTH3 : Trần Đình Phong - 2114404

THÀNH PHỐ HỒ CHÍ MINH - THÁNG 5 NĂM 2025

DÀI HỌC QUỐC GIA TP.HCM
TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA: KH & KT Máy tính
BỘ MÔN: KTMT

HỌ VÀ TÊN: BÙI QUANG BẰNG
HỌ VÀ TÊN: NGUYỄN TUẤN HÙNG
HỌ VÀ TÊN: TRẦN ĐÌNH PHONG
NGÀNH: KỸ THUẬT MÁY TÍNH

1. Đầu đề luận án:

- Hiện thực ứng dụng khám sức khỏe dựa trên nền tảng kết nối vạn vật
2. Nhiệm vụ (yêu cầu về nội dung và số liệu ban đầu):

- Tìm hiểu các thông tin cần đo, yêu cầu để tải và môi trường lập trình phần cứng, cài đặt FreeRTOS và phát triển thư viện lập trình.
- Lập trình thu thập dữ liệu từ hệ thống cảm biến và kết nối dữ liệu với server IoT.
- Xây dựng, hoàn thiện hệ thống cung cấp các hướng dẫn dưới dạng âm thanh và hình ảnh, và phát triển ứng dụng trên thiết bị di động.
- Tích hợp, vận hành thử nghiệm và đánh giá hiệu năng của hệ thống.

3. Ngày giao nhiệm vụ luận án: 15/01/2025

4. Ngày hoàn thành nhiệm vụ: 15/05/2025

5. Họ tên giảng viên hướng dẫn:

1) TS. LÊ TRỌNG NHÂN

HĐ3 KTHT - 311
CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

NHIỆM VỤ LUẬN ÁN TỐT NGHIỆP

MSSV: 2012681

MSSV: 2111386

MSSV: 2114404

Phản hướng dẫn:

Toàn bộ

Nội dung và yêu cầu LVTN đã được thông qua Bộ môn.

Ngày tháng năm
CHỦ NHIỆM BỘ MÔN
(Ký và ghi rõ họ tên)

Phạm Quốc Cường

GIẢNG VIÊN HƯỚNG DẪN CHÍNH
(Ký và ghi rõ họ tên)

TS. Lê Trọng Nhân

PHẦN DÀNH CHO KHOA, BỘ MÔN:

Người duyệt (chấm sơ bộ):

Đơn vị:

Ngày bảo vệ:

Điểm tổng kết:

Nơi lưu trữ luận án:

HĐ 3 LTMT

PHIẾU ĐÁNH GIÁ LUẬN VĂN/ ĐÒ ÁN TỐT NGHIỆP
(Dành cho người hướng dẫn)

1. Họ và tên SV: BÙI QUANG BẰNG

MSSV: 2012681

Ngành (chuyên ngành): Kỹ thuật Máy tính

Họ và tên SV: NGUYỄN TUẤN HÙNG

MSSV: 2111386

Ngành (chuyên ngành): Kỹ thuật Máy tính

Họ và tên SV: TRẦN ĐÌNH PHONG

MSSV: 2114404

Ngành (chuyên ngành): Kỹ thuật Máy tính

2. Đề tài: Hiện thực ứng dụng khám sức khỏe dựa trên nền tảng kết nối vạn vật

3. Họ tên người hướng dẫn: TS. Lê Trọng Nhán

4. Tổng quát về bản thuyết minh:

Số trang:

Số chương:

Số bảng số liệu:

Số hình vẽ:

Số tài liệu tham khảo:

Phần mềm tính toán:

Hiện vật (sản phẩm)

5. Những ưu điểm chính của LV/ĐATN:

Đề tài xây dựng 1 hệ thống giám sát sức khỏe dựa trên nền tảng kết nối vạn vật, với các tính năng chính như sau:

- Hệ thống cảm biến: Cảm biến cân nặng, SPO2, Nhiệt độ và nhịp tim được thu thập trực tiếp từ cảm biến. Bên cạnh đó, Camera sẽ được sử dụng để nội suy thêm thông tin về chiều cao, giới tính và độ tuổi.
- Kết nối với server IOT: Dữ liệu về sức khỏe sẽ được gửi lên server CoreIOT, phục vụ cho việc lưu trữ và phân tích dữ liệu
- Ứng dụng trên trạm thiết bị: Ứng dụng được phát triển trên nền tảng lập trình Python, cung cấp các UI cơ bản để hướng dẫn người dùng thao tác trên thiết bị

6. Những thiếu sót chính của LV/ĐATN:

Hệ thống vận hành chưa ổn định, vẫn còn 1 số lỗi nhỏ về logic

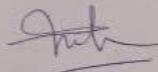
7. Đề nghị: Được bảo vệ Bổ sung thêm để bảo vệ Không được bảo vệ

8. Các câu hỏi SV phải trả lời trước Hội đồng:

a. Sinh viên hãy trình bày cách lấy dữ liệu từ hệ thống cảm biến của đề tài.

9. Đánh giá chung (bằng chữ: Xuất sắc, Giỏi, Khá, TB): Khá 6.5/10

Ký tên (ghi rõ họ tên)



Lê Trọng Nhán

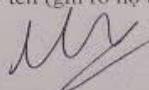
HÀ 34/TM - 311

Ngày 18 tháng 5 năm 2025

PHIẾU ĐÁNH GIÁ LUẬN VĂN/ ĐỒ ÁN TỐT NGHIỆP
(Dành cho người hướng dẫn/phản biện)

1. Họ và tên SV: **Bùi Quang Bằng**
MSSV: 2012681
Họ và tên SV: **Nguyễn Tuấn Hùng**
MSSV: 2111386
Họ và tên SV: **Trần Đình Phong**
MSSV: 2114404
2. Đề tài: **Hiện thực ứng dụng khám sức khỏe dựa trên nền tảng kết nối vạn vật**
3. Họ tên người hướng dẫn/phản biện: **Phạm Công Thái**
4. Tổng quát về bản thuyết minh:
Số trang:
Số bảng số liệu
Số tài liệu tham khảo:
Hiện vật (sản phẩm)
5. Những ưu điểm chính của LV/ĐATN:
 - Sinh viên làm chủ được phần cứng bao gồm các cảm biến, gateway, CPU và trạm cân cho hệ thống khám sức khỏe.
 - Sinh viên phát triển ứng dụng trên điện thoại để hỗ trợ theo dõi và thông kê dữ liệu sức khỏe bệnh nhân.
 - Sinh viên thể hiện khả năng áp dụng các kiến thức chuyên ngành đã được học để hiện thực đề tài.
6. Những thiếu sót chính của LV/ĐATN:
 - Các chức năng của hệ thống còn sơ sài, chưa có phân tích chuyên sâu để ứng dụng thực tiễn.
 - Giao diện trên ứng dụng điện thoại chưa hiển thị tốt các thông số sức khỏe và server chưa tối ưu cho lượng lớn người dùng.
 - Hệ thống vẫn chưa được hoàn thiện và vận hành hệ thống còn khó khăn.
7. Đề nghị: Được bảo vệ Bổ sung thêm để bảo vệ Không được bảo vệ
8. Các câu hỏi SV phải trả lời trước Hội đồng:
 - Sinh viên hãy cho biết trong môi trường thực tế thì nhóm có biện pháp gì để đảm bảo độ chính xác của cảm biến và làm thế nào để chứng minh các chỉ số đo được đạt tiêu chuẩn y tế ?
 - Mô hình mà nhóm đang thực hiện có rủi ro bảo mật nào khi triển khai trong thực tế ?
9. Đánh giá chung (bằng chữ: Xuất sắc, Giỏi, Khá, TB): **Giỏi** Điểm : 8 /10

Ký tên (ghi rõ họ tên)



Phạm Công Thái

Lời cam đoan

Nhóm sinh viên xin cam kết rằng toàn bộ đồ án môn học này được thực hiện dưới sự giám sát của TS. Lê Trọng Nhân tại Khoa Khoa học và Kỹ thuật Máy tính, trường Đại học Bách Khoa TP Hồ Chí Minh. Nhóm sinh viên đã ghi nhận cẩn thận và đầy đủ tất cả các nguồn tài liệu tham khảo bên ngoài đã sử dụng trong dự án này.

Nhóm sinh viên khẳng định hoàn toàn chịu trách nhiệm về tính xác thực và tính nguyên bản của nội dung, đảm bảo không có đạo văn. Nếu lời cam đoan sai sự thật, nhóm sinh viên chịu mọi trách nhiệm trước Ban Chủ nhiệm Khoa và Ban Giám hiệu Nhà trường

THÀNH PHỐ HỒ CHÍ MINH -
THÁNG 5 NĂM 2025

Nhóm sinh viên thực hiện đề tài,
Bùi Quang Bằng
Nguyễn Tuấn Hùng
Trần Đình Phong

Lời cảm ơn

Nhóm sinh viên xin gửi lời cảm ơn chân thành và sâu sắc tới TS. Lê Trọng Nhân và trợ giảng Nguyễn Thiên Ân. Nhờ sự hướng dẫn tận tình và sự hỗ trợ không ngừng của hai thầy, nhóm mới có thể hoàn thành đồ án môn học này.

Nhóm cũng muốn gửi lời cảm ơn đến các giảng viên đang kính của Khoa Khoa học và Kỹ thuật Máy tính, trường Đại học Bách Khoa TP Hồ Chí Minh vì đã cung cấp cho các thành viên trong nhóm kiến thức nền tảng vững chắc và các kỹ năng chuyên môn để hoàn thành chỉnh chu đồ án môn học.

Nhóm cũng chân thành cảm ơn các thành viên trong Lab vì đã cung cấp cũng như là hướng đi chọn công nghệ và thiết bị cho nhóm. Giúp cho nhóm có một môi trường để thực hiện một cách đồ án một cách thuận tiện nhất.

Cuối cùng, nhóm muốn cảm ơn các thành viên trong nhóm vì đã hợp tác hoàn thành đồ án một cách tốt nhất.

Tóm tắt

Sau những biến cố lớn gây ảnh hưởng lớn đến sức khỏe toàn cầu (Covid-19), đi cùng phương châm số hóa, hiện đại hóa của Việt Nam, nhóm đã quyết định thực hiện đồ án này nhằm đưa ra một trạm khám sức khỏe tổng quát có thể được sử dụng một cách dễ dàng, trực quan và dễ tiếp cận.

Toàn bộ hệ thống được chia ra làm ba phần hiện thực riêng biệt, bao gồm phần cứng, phần mềm hỗ trợ và phần mềm quản lý. Kết hợp với mô hình trạm được thiết kế hoàn chỉnh, khả năng đo nhiều loại thông số bằng các loại cảm biến và thiết bị để có thể thu thập thông tin về sức khỏe tổng quát của người sử dụng mà không tốn nhiều thời gian. Phần mềm để sử dụng hệ thống được hiện thực thông qua browser nhằm tiện lợi hơn cho việc hiện thực cũng như sử dụng.

Nhóm đã hiện thực được một hệ thống với trạm bao gồm khả năng tương tác với người dùng thông qua loa và màn hình cảm ứng, đo được nhiều loại dữ liệu về sức khỏe khác nhau bao gồm cân nặng, chiều cao, nhịp tim/phút, nồng độ oxy trong máu và nhiệt độ cơ thể của người dùng. Toàn bộ quá trình sử dụng tốn khoảng 2-3 phút cho mỗi lần đo toàn bộ tất cả các thông số, và gói gọn trong một trạm cân với kích cỡ nhỏ gọn và tiện lợi.

Qua đó, nhóm đã đưa ra một sản phẩm, với điều chỉnh và hoàn thiện hơn, có thể đưa ra thị trường, được sử dụng ở nơi công cộng, giúp cải thiện độ nhận thức về y tế của người sử dụng thông qua việc cho người sử dụng biết các thông số cơ bản về sức khỏe của bản thân chỉ trong vòng vài phút.

Mục lục

1	Giới thiệu đề tài	1
1.1	Động lực	1
1.2	Mục tiêu	3
1.3	Phạm vi	4
1.4	Cấu trúc đồ án	6
2	Thiết kế hệ thống	10
2.1	Thiết kế trùu tượng ban đầu	10
2.2	Kiến trúc hệ thống	13
3	Hiện thực hệ thống	36
3.1	Hiện thực phần cứng	36
3.2	Hiện thực giao diện màn hình khám sức khỏe	59
3.3	Ứng dụng theo dõi trên điện thoại	66
4	Thực nghiệm	85
4.1	Kiểm thử phần cứng	85
4.2	Kiểm thử Browser của hệ thống	88
4.3	Kiểm thử chức năng Ứng dụng theo dõi	89
5	Tổng kết và định hướng phát triển	94
5.1	Tổng kết và đánh giá tiến độ	94
5.2	Hướng phát triển trong tương lai	96
	Tài liệu tham khảo	98

Danh sách hình ảnh

Figure 2.1	Thiết kế hệ thống ở mức nguyên mẫu ban đầu	11
Figure 2.2	Sơ đồ kiến trúc của hệ thống.	14
Figure 2.3	Sơ đồ phân tầng kiến trúc hệ thống.	15
Figure 2.4	Module GY-906 chứa cảm biến MLX90614.	15
Figure 2.5	Module chứa cảm biến camera nhiệt MLX90640.	17
Figure 2.6	Module chứa cảm biến MAX30102.	17
Figure 2.7	Module chứa cảm biến MAX30100.	19
Figure 2.8	Cách kết nối giữa 2 thiết bị I2C.	20
Figure 2.9	Một frame tin nhắn của I2C.	21
Figure 2.10	Cân thông minh Xiaomi Smart Scale 2.	22
Figure 2.11	BLE Stack	24
Figure 2.12	Servo HerkuleX DRS-0201.	26
Figure 2.13	Cách kết nối dây giữa hai thiết bị UART	28
Figure 2.14	Module Yolo UNO với chipset ESP32-S3 N16R8.	29
Figure 3.1	Robot Ohmni Telepresence.	39
Figure 3.2	Biểu đồ 100 mẫu thử sử dụng giải thuật RF.	45
Figure 3.3	Biểu đồ 100 mẫu thử sử dụng giải thuật SparkFun.	45
Figure 3.4	FSM vận hành và kết nối BLE đến cân Xiaomi	47
Figure 3.5	Màn hình và hub chứa cảm biến + camera bên trên màn hình.	57
Figure 3.6	Đằng sau hub màn hình và hub chứa cảm biến + camera bên trên màn hình.	58

Figure 3.7	Hệ thống hoàn thiện, nhìn từ phía trước.	58
Figure 3.8	Sơ đồ của giao diện màn hình	59
Figure 3.9	Màn hình chờ	61
Figure 3.10	Người dùng khi ở vị trí hoàn hảo	62
Figure 3.11	Người dùng đang thấp hơn	62
Figure 3.12	Người dùng đang cao hơn	63
Figure 3.13	Màn hình đo chiều cao	64
Figure 3.14	Màn hình đo cân nặng	64
Figure 3.15	Màn hình đo nhiệt độ cơ thể	65
Figure 3.16	Màn hình đo nhịp tim và oxy trong máu	65
Figure 3.17	Giao diện Login	67
Figure 3.18	Đăng kí bước 1	68
Figure 3.19	Đăng kí bước 2	69
Figure 3.20	Đăng kí bước 3	70
Figure 3.21	Đăng kí thành công	71
Figure 3.22	Giao diện DashBoard	72
Figure 3.23	Giao diện Dashboard khi đo	73
Figure 3.24	Giá trị sensor khi đo	74
Figure 3.25	Giá trị trung bình	75
Figure 3.26	Giao diện lịch sử đo	76
Figure 3.27	Tùy chọn máy đo	77
Figure 3.28	Lịch sử đo chi tiết	78
Figure 3.29	Lịch sử đo chi tiết (tiếp tục)	79
Figure 3.30	Giao diện điều khiển	80
Figure 3.31	Tùy chọn máy đo	81
Figure 3.32	Giao diện cài đặt	82
Figure 3.33	Thông tin cá nhân	83
Figure 3.34	Thay đổi thông tin cá nhân	84

Figure 4.1	Hệ thống khi khởi động với các lệnh STATUS trả về.	85
Figure 4.2	Lệnh NACK được trả về khi lệnh nhập sai.	85
Figure 4.3	các lệnh xoay servo được nhận với ACK trả về.	86
Figure 4.4	các lệnh xoay servo được nhận với ACK trả về cùng với góc được trả về - góc xoay tối đa.	86
Figure 4.5	Góc xoay tối thiểu của servo.	86
Figure 4.6	Servo dừng ở một góc ngẫu nhiên.	87
Figure 4.7	ACK của lệnh lấy nhiệt độ môi trường xung quanh.	87
Figure 4.8	ACK của lệnh lấy nhiệt độ vật thể, nhưng không có vật thể trước cảm biến.	87
Figure 4.9	ACK của lệnh lấy nhiệt độ vật thể, có vật thể trước cảm biến (bàn tay người).	87
Figure 4.10	ACK của lệnh lấy dữ liệu từ cảm biến MAX30102, với dữ liệu được trả về, số đầu tiên là số nhịp tim/phút, số thứ 2 là chỉ số nồng độ oxy máu theo %.	88
Figure 4.11	ACK của lệnh lấy dữ liệu từ cân, với cân nặng theo kg được trả về.	88
Figure 4.12	Giao diện ứng dụng sau khi đăng nhập	91
Figure 4.13	Dữ liệu trên Firebase	92
Figure 4.14	Dữ liệu trên Ứng dụng	92
Figure 4.15	Giá trị trung bình của lần đo	93
Figure 4.16	Lịch sử đo chi tiết	93

Danh sách bảng

Table 3.1	Bảng so sánh nhiệt độ bề mặt da so với nhiệt độ lõi cơ thể của feevr (theo độ C) [16]	43
Table 3.2	Nồng độ oxy máu tiêu chuẩn. [17]	46
Table 3.3	Mức nhịp tim/phút (BPM) mong đợi theo độ tuổi. [45]	46
Table 3.4	Cấu các byte dữ liệu cân Xiaomi Smart Scale 2 gửi [48]	48
Table 3.5	Các lệnh giao tiếp qua cổng COM giữa CPU và IoT Gateway	54

Chương 1

Giới thiệu đề tài

Trong chương 1, tổng quan sơ lược, dành cho việc mô tả những mục tiêu cũng như nhiệm vụ của đồ án này, cùng với khung sườn của toàn bộ đồ án.

1.1 Động lực

Trong những năm gần đây, lĩnh vực y tế toàn cầu đang chứng kiến những biến chuyển sâu sắc cả về mặt thách thức lẫn cơ hội. Một trong những cú sốc lớn nhất chính là đại dịch COVID-19, khi chỉ trong vòng hai năm (2019–2021), tuổi thọ trung bình toàn cầu giảm từ 73,1 tuổi xuống còn 71,4 tuổi – mức thấp nhất trong vòng một thập kỷ, làm tiêu tan thành quả tích lũy từ nhiều năm cải thiện y tế công cộng [1]. Song song đó, tuổi thọ khỏe mạnh toàn cầu (HALE – Healthy Life Expectancy) cũng giảm từ 63,5 xuống 61,9 tuổi [2]. Đây là lần đầu tiên từ năm 1950 đến nay mà thế giới chứng kiến sự sụt giảm liên tiếp về tuổi thọ, một cảnh báo nghiêm trọng cho hệ thống y tế toàn cầu [3].

Không chỉ ảnh hưởng về mặt con số, đại dịch còn làm lộ rõ những điểm yếu cố hữu của ngành y, từ tình trạng quá tải bệnh viện, thiếu hụt nhân sự y tế, đến khả năng quản lý khủng hoảng và chia sẻ dữ liệu y tế. Theo thống kê của World Bank, hơn 90% quốc gia trên thế giới bị gián đoạn dịch vụ y tế cơ bản trong đại dịch [4]. Trong khi các quốc gia phát triển có thể ứng phó bằng cách tăng cường số hóa, y tế từ xa và sử dụng các nền tảng quản lý thông minh, thì nhiều quốc

gia đang phát triển, bao gồm cả Việt Nam, lại gặp nhiều khó khăn do hạn chế về hạ tầng công nghệ, thiếu hụt nguồn nhân lực chuyên môn và sự chênh lệch trong phân bố y tế giữa thành thị và nông thôn.

Việt Nam, dù đã có nhiều bước tiến trong lĩnh vực y tế, vẫn đang đối mặt với áp lực từ tốc độ già hóa dân số nhanh chóng, tỷ lệ mắc các bệnh không lây nhiễm (như tiểu đường, tim mạch, béo phì) đang ngày càng tăng, và các dịch bệnh mới nổi lên ngày càng nhiều [5]. Theo báo cáo của Bộ Y tế, hơn 70% gánh nặng bệnh tật ở Việt Nam hiện nay đến từ các bệnh không lây nhiễm, trong khi đây là nhóm bệnh có thể phòng ngừa sớm bằng các công cụ theo dõi sức khỏe cá nhân [6].

Chuyển đổi số trong y tế được xem là hướng đi tất yếu để giải quyết các bài toán nêu trên. Việc ứng dụng các công nghệ mới như trí tuệ nhân tạo (AI), cảm biến y sinh, dữ liệu lớn (Big Data), và đặc biệt là Internet vạn vật (IoT) đang giúp ngành y chuyển mình từ mô hình "chữa bệnh bị động" sang "phòng bệnh chủ động" [7]. Những công nghệ này không chỉ cho phép giám sát tình trạng sức khỏe theo thời gian thực, mà còn giúp bệnh nhân và bác sĩ đưa ra quyết định dựa trên dữ liệu chính xác. IoT đặc biệt đóng vai trò quan trọng trong việc kết nối các thiết bị đo lường từ máy đo huyết áp, nồng độ oxy máu, nhịp tim đến các nền tảng lưu trữ và phân tích dữ liệu từ xa [8].

Tuy nhiên, một thực tế đáng quan ngại là các giải pháp chăm sóc sức khỏe thông minh tại Việt Nam vẫn còn rất hạn chế, chủ yếu do chi phí cao, thiếu tích hợp phần cứng – phần mềm, và thiết kế chưa tối ưu cho người dùng phổ thông. Trên thị trường hiện nay, đa số các thiết bị y tế cá nhân đều là hàng nhập khẩu với chi phí cao, trong khi nhu cầu về các hệ thống chăm sóc sức khỏe đơn giản, hiệu quả, giá thành rẻ và phù hợp với trình độ công nghệ của người dân vẫn chưa được đáp ứng đầy đủ [9].

Chính từ những vấn đề thực tế đó, nhóm chúng em nhận thấy cần thiết phải xây dựng một giải pháp y tế dựa trên nền tảng IoT, không chỉ đóng vai trò như một công cụ hỗ trợ theo dõi sức khỏe tại nhà, mà còn góp phần vào việc hình thành hệ sinh thái y tế thông minh tại Việt Nam. Hệ thống cần đảm bảo các tiêu chí: dễ sử dụng, chi phí thấp, có khả năng mở rộng tính năng, và đặc biệt là tích hợp dữ liệu vào các nền tảng y tế số để hỗ trợ bác sĩ trong chẩn đoán và điều trị.

Thông qua đồ án này, nhóm hướng tới việc xây dựng một mô hình ứng dụng khám sức khỏe tổng quát thông minh, dựa trên cảm biến đo sinh hiệu và vi điều khiển kết nối mạng, phục vụ nhu cầu theo dõi sức khỏe định kỳ, tầm soát bệnh sớm và hướng đến mô hình y tế cá nhân hóa. Đây không chỉ là một sản phẩm công nghệ, mà còn là bước khởi đầu cho việc hiện thực hóa một tầm nhìn y tế số toàn diện hơn trong tương lai gần tại Việt Nam.

1.2 Mục tiêu

Đề tài được triển khai qua hai giai đoạn, trong đó mỗi giai đoạn đều có mục tiêu cụ thể phù hợp với mức độ phát triển và hoàn thiện hệ thống.

Ở giai đoạn đầu, mục tiêu của nhóm chủ yếu là xây dựng nền tảng kiến thức và thực hiện những bước triển khai cơ bản để tạo tiền đề cho hệ thống khám sức khỏe IoT. Đầu tiên, nhóm đã nghiên cứu và làm rõ các khái niệm cốt lõi liên quan đến Internet of Things (IoT) trong lĩnh vực y tế, bao gồm cách hoạt động của hệ thống IoT, các thành phần cấu thành, cũng như các chuẩn giao tiếp dữ liệu phổ biến như MQTT, HTTP, CoAP. Bên cạnh đó, việc tìm hiểu nguyên lý hoạt động và ứng dụng thực tế của các cảm biến y tế như cảm biến đo nhịp tim, SpO2, nhiệt độ... là yêu cầu thiết yếu nhằm phục vụ bước triển khai phần cứng. Tiếp theo, nhóm thực hiện lựa chọn các loại cảm biến phù hợp và tiến hành kết nối chúng với vi điều khiển, thiết lập luồng thu thập và xử lý dữ liệu sức khỏe cơ

bản. Đồng thời, phần mềm điều khiển cũng được phát triển để thu thập và truyền dữ liệu lên hệ thống trung tâm. Ngoài ra, nhóm cũng đã triển khai một ứng dụng di động nhằm phục vụ cho việc theo dõi việc khám từ xa cũng như truy xuất lịch sử các lần đo. Cuối cùng, một giai đoạn đánh giá đã được nhóm thực hiện để kiểm nghiệm hiệu suất hoạt động của thiết bị, độ chính xác của cảm biến, và độ ổn định của đường truyền dữ liệu.

Tiếp nối những kết quả đạt được từ giai đoạn đầu, giai đoạn hai của đề tài tập trung vào việc mở rộng và hoàn thiện hệ thống theo hướng ứng dụng thực tiễn. Ở giai đoạn này, nhóm đặt mục tiêu phát triển một mô hình hệ thống khám sức khỏe thông minh hoàn chỉnh, có khả năng tích hợp đa dạng cảm biến và quản lý dữ liệu người dùng hiệu quả. Cụ thể, hệ thống sẽ tiếp tục sử dụng vi điều khiển để xử lý dữ liệu đo được từ các cảm biến và hiển thị tạm thời thông tin lên màn hình LCD hoặc OLED. Dữ liệu sau đó được truyền về máy tính qua giao tiếp nối tiếp, rồi chuyển tiếp lên nền tảng lưu trữ đám mây như Firebase để đảm bảo tính sẵn sàng và truy cập mọi lúc mọi nơi. Ngoài phần cứng, nhóm còn xây dựng phần mềm quản lý người dùng, bao gồm thiết kế cơ sở dữ liệu, phát triển API, và xây dựng giao diện ứng dụng web hoặc mobile cho phép người dùng theo dõi kết quả sức khỏe của mình. Qua đó, hệ thống không chỉ dừng lại ở mức mô hình kỹ thuật mà còn hướng đến việc ứng dụng trong thực tế, chẳng hạn tại các phòng khám, trạm y tế hoặc trong các chương trình khám sức khỏe cộng đồng lưu động, góp phần tự động hóa bước khám ban đầu và nâng cao hiệu quả công tác khám chữa bệnh.

1.3 Phạm vi

Đồ án tập trung vào việc thiết kế và xây dựng một hệ thống khám sức khỏe sơ bộ, ứng dụng công nghệ Internet vạn vật (IoT) để hỗ trợ người dùng tự theo dõi các chỉ số sinh lý cơ bản một cách tự động và thuận tiện. Hệ thống hướng đến việc cải thiện nhận thức về sức khỏe cá nhân trong cộng đồng, đồng thời góp

phần hình thành thói quen theo dõi sức khỏe định kỳ, đặc biệt trong bối cảnh các bệnh mãn tính như tăng huyết áp, tim mạch, béo phì hay đái tháo đường ngày càng phổ biến.

Quá trình nghiên cứu và triển khai được giới hạn trong một số khía cạnh kỹ thuật, công nghệ và ứng dụng cụ thể, phù hợp với quy mô và mục tiêu của đề án:

1. **Chỉ số sinh lý cơ bản:** Hệ thống tập trung vào việc đo và xử lý các thông số sức khỏe phổ biến như nhịp tim, nhiệt độ cơ thể, nồng độ oxy trong máu (SpO2), chiều cao, cân nặng và chỉ số khối cơ thể (BMI). Các chỉ số chuyên sâu hơn như huyết áp tâm thu/tâm trương, điện tâm đồ (ECG), glucose máu hay dữ liệu hình ảnh y tế (CT, MRI) đòi hỏi thiết bị chuyên dụng và tiêu chuẩn y tế khắt khe, nên chưa được tích hợp trong giai đoạn này.
2. **Kiến trúc truyền thông nội bộ:** Dữ liệu thu thập được từ cảm biến sẽ được truyền đến một thiết bị trung gian như máy tính hoặc vi điều khiển mạnh (ESP32), sau đó được xử lý sơ bộ và gửi đến nền tảng lưu trữ đám mây như Firebase. Từ đó, ứng dụng di động hoặc web có thể truy xuất dữ liệu theo thời gian thực. Tuy nhiên, hệ thống chưa kết nối với cơ sở dữ liệu y tế quốc gia hoặc các hệ thống bệnh viện thực tế do yêu cầu bảo mật, chuẩn giao tiếp HL7/FHIR và sự phức tạp trong triển khai.
3. **Ứng dụng theo dõi sức khỏe đơn giản:** Giao diện người dùng hiện tại được xây dựng ở mức cơ bản, chủ yếu để hiển thị dữ liệu tức thời, biểu đồ trực quan và nhật ký theo dõi các chỉ số qua thời gian. Các tính năng nâng cao như cảnh báo nguy cơ, phân tích xu hướng sức khỏe, đánh giá mức độ thể trạng, hoặc gợi ý hành vi cải thiện sức khỏe chưa được phát triển, nhưng có thể là hướng mở cho tương lai.
4. **Giới hạn phần cứng và triển khai thực tế:** Do điều kiện kỹ thuật, kinh phí và thời gian có hạn, hệ thống được thử nghiệm trên số lượng nhỏ người

dùng, trong môi trường giả lập (lớp học, phòng lab, hoặc gia đình). Tính năng bảo trì hệ thống, mở rộng đồng thời cho nhiều người dùng, và tối ưu hóa năng lượng cho thiết bị phần cứng và thiết bị di động chưa được xử lý toàn diện. Những yếu tố này sẽ được cân nhắc nếu triển khai ở quy mô thương mại hoặc y tế thực tiễn.

5. Bảo mật và quyền riêng tư dữ liệu: Đề án chỉ áp dụng các biện pháp bảo mật cơ bản như xác thực bằng mã người dùng và mã hóa đường truyền khi gửi dữ liệu. Các vấn đề chuyên sâu về quyền riêng tư y tế cá nhân, kiểm soát truy cập nâng cao, tuân thủ quy định bảo vệ dữ liệu (như GDPR hay HIPAA) chưa được triển khai đầy đủ, và sẽ là yêu cầu bắt buộc nếu hệ thống tiến tới ứng dụng thực tiễn.

Với phạm vi được giới hạn và xác định rõ ràng, đề tài nhằm xây dựng một hệ thống nguyên mẫu có thể hoạt động ổn định, dễ sử dụng, dễ triển khai, và đóng vai trò như một bước đệm để mở rộng hoặc tích hợp vào các hệ thống chăm sóc sức khỏe lớn hơn trong tương lai.

1.4 Cấu trúc đồ án

Để đảm bảo tính hệ thống, dễ tiếp cận và thể hiện rõ ràng tiến trình hiện thực hóa đề tài, nội dung của bài báo cáo được tổ chức theo một cấu trúc logic gồm năm chương chính. Mỗi chương không chỉ phản ánh một giai đoạn quan trọng trong quá trình nghiên cứu và triển khai hệ thống mà còn đóng vai trò kết nối liền mạch giữa cơ sở lý thuyết và thực tiễn kỹ thuật. Bộ cục này giúp đảm bảo sự kết nối chặt chẽ giữa nền tảng lý thuyết, phân tích công nghệ và quá trình hiện thực, tạo nên một hành trình xuyên suốt, mạch lạc và có định hướng trong suốt toàn bộ quá trình phát triển hệ thống. Dưới đây là nội dung từng chương cụ thể:

- 1. Chương 1 - Giới thiệu đề tài** trình bày tổng quan về bối cảnh thực hiện đồ án, động lực và ý nghĩa thực tiễn của đề tài trong bối cảnh y tế hiện nay – nơi mà nhu cầu theo dõi sức khỏe cá nhân ngày càng gia tăng sau đại dịch

Covid-19. Nội dung chương đề cập đến những thay đổi sâu rộng trong nhận thức của cộng đồng về tầm quan trọng của y tế dự phòng và vai trò của công nghệ trong việc phổ cập hóa các dịch vụ khám sức khỏe. Từ đó, nhóm đặt ra mục tiêu phát triển một hệ thống khám sức khỏe tổng quát, dễ tiếp cận, thân thiện với người dùng nhưng vẫn đảm bảo độ chính xác cao trong việc thu thập và xử lý thông tin sinh học. Ngoài ra, chương 1 cũng trình bày rõ ràng phạm vi đồ án, những giới hạn kỹ thuật và nguồn lực mà nhóm phải tuân theo trong quá trình thực hiện. Cuối cùng, chương định hướng cấu trúc nội dung toàn bộ tài liệu nhằm giúp người đọc nắm bắt mục tiêu cụ thể của từng chương tiếp theo.

2. **Chương 2 - Thiết kế hệ thống** tập trung vào việc thiết kế tổng thể của hệ thống. Ban đầu, chương 2 mô tả kiến trúc trừu tượng của hệ thống khám sức khỏe dựa trên IoT, gồm các thành phần chính: các cảm biến đầu cuối (đảm nhiệm thu thập thông số sinh học từ người dùng), bộ vi xử lý trung tâm (xử lý và tổng hợp dữ liệu), máy chủ (lưu trữ và phân tích thông tin) và các thiết bị người dùng như máy tính hay điện thoại thông minh (hiển thị kết quả). Sơ đồ khái của hệ thống được trình bày chi tiết, cho thấy cách thức tương tác giữa các module và luồng dữ liệu di chuyển từ lớp cảm biến đến lớp hiển thị. Tiếp theo, chương phân tích các lựa chọn thiết bị và nền tảng phần mềm phù hợp với yêu cầu đề tài. Chương còn giải thích cơ sở chọn lựa các cảm biến cụ thể (như cảm biến đo nhịp tim, nhiệt độ, độ bão hòa oxy...), đồng thời trình bày các thành phần phần cứng kèm theo như vi điều khiển và bộ thu phát tín hiệu. Đối với phần mềm, chương này giới thiệu các công cụ và môi trường phát triển được sử dụng (ví dụ: ngôn ngữ lập trình nhúng, hệ điều hành thời gian thực, nền tảng phát triển ứng dụng di động hoặc web). Những quyết định thiết kế này được phân tích để minh họa cách thức hệ thống đáp ứng các tiêu chí về tính ổn định, khả năng mở rộng và hiệu quả xử lý trong việc thu thập và truyền tải dữ liệu.
3. **Chương 3 - Hiện thực hệ thống** cung cấp các cơ sở lý thuyết và kiến thức

nền tảng cần thiết cho hệ thống, đồng thời trình bày chi tiết quá trình triển khai. Đầu tiên, chương giới thiệu khái niệm và nguyên lý cơ bản: nguyên lý hoạt động của các loại cảm biến sinh trắc học, kiến trúc mạng IoT và các giao thức truyền thông dữ liệu phổ biến (chẳng hạn MQTT, HTTP, Bluetooth). Những kiến thức này làm cơ sở để người đọc hiểu cách thức mà hệ thống khám sức khỏe hoạt động từ góc độ công nghệ. Tiếp theo, chương trình bày chi tiết quá trình xây dựng hệ thống từ phần cứng đến phần mềm. Phần thiết kế phần cứng mô tả cách thức kết nối các cảm biến với bộ vi điều khiển và các thành phần ngoại vi, đảm bảo thu thập dữ liệu chính xác. Phần lập trình phần mềm đề cập đến các bước phát triển chương trình trên vi xử lý để xử lý dữ liệu cảm biến, cũng như tạo dựng giao diện người dùng trên màn hình và ứng dụng di động. Ngoài ra, chương còn phân tích các giao thức truyền thông cụ thể được áp dụng, đồng thời minh họa luồng dữ liệu giữa các lớp khác nhau trong hệ thống, giúp làm rõ quy trình làm việc nội bộ của hệ thống.

4. **Chương 4 - Thực nghiệm** mô tả quy trình kiểm thử và đánh giá hiệu năng của hệ thống sau khi hoàn thiện. Mục tiêu của chương là chứng minh rằng hệ thống hoạt động ổn định và đáp ứng được các yêu cầu đã đề ra. Nhóm đã thiết lập nhiều kịch bản thử nghiệm khác nhau: bao gồm kiểm tra độ ổn định của cảm biến trong quá trình đo liên tục, kiểm tra khả năng truyền dữ liệu không dây và khả năng phản hồi của hệ thống khi có nhiều yêu cầu đồng thời từ người dùng. Trong mỗi thử nghiệm, các chỉ số về hiệu suất được thu thập và phân tích. Cụ thể, nhóm tác giả đánh giá tốc độ truyền dữ liệu giữa các thiết bị, thời gian phản hồi của hệ thống khi người dùng thực hiện các thao tác, và độ chính xác của các chỉ số sinh trắc học (so với phép đo chuẩn). Kết quả thử nghiệm được trình bày và thảo luận một cách chi tiết, từ đó rút ra những nhận xét về điểm mạnh và hạn chế của hệ thống. Dựa trên các phân tích này, chương cũng đưa ra các khuyến nghị nhằm tối ưu hóa và cải thiện chất lượng hệ thống trong tương lai.

5. Chương 5 - Tổng kết và định hướng phát triển tổng kết lại toàn bộ quá trình thực hiện đồ án và đánh giá kết quả đạt được. Trước tiên, chương khái quát lại các mục tiêu chính và nội dung quan trọng đã triển khai trong các chương trước. Nhóm sẽ đưa ra những nhận xét và đánh giá về mức độ hoàn thành các yêu cầu đề ra: ví dụ như hệ thống đã thu thập thành công các chỉ số sinh trắc học cơ bản, hoạt động ổn định và cung cấp thông tin hữu ích cho người dùng. Đánh giá cũng chỉ ra những vấn đề còn tồn tại hoặc các chức năng chưa được phát triển đầy đủ trong phiên bản hiện tại của hệ thống. Tiếp theo, chương này đề xuất hướng phát triển trong tương lai dựa trên kết quả nghiên cứu. Cụ thể, nhóm đề xuất mở rộng các tính năng của hệ thống (như tích hợp thêm nhiều loại cảm biến hoặc phát triển thuật toán xử lý dữ liệu nâng cao), cải tiến hiệu năng (tăng tốc độ xử lý, nâng cấp giao thức truyền thông) và nghiên cứu ứng dụng công nghệ mới (tích hợp trí tuệ nhân tạo hoặc phân tích dữ liệu lớn để nâng cao khả năng phân tích sức khỏe). Những đề xuất và định hướng này được đề ra với mục tiêu nhằm hoàn thiện sản phẩm và nâng cao giá trị ứng dụng của hệ thống trong tương lai.

Chương 2

Thiết kế hệ thống

Chương 2 sẽ trình bày về thiết kế hệ thống, cấu trúc hoạt động của hệ thống, những lựa chọn về mặt thiết bị, nền tảng phần mềm mà hệ thống sẽ sử dụng dựa trên những thiết kế đã được quyết định.

2.1 Thiết kế trừu tượng ban đầu

Với những yêu cầu dự án như đã nói trên, ở giai đoạn thiết kế ban đầu, nhóm đã thiết kế hệ thống với nguyên mẫu trừu tượng theo biểu đồ sau:

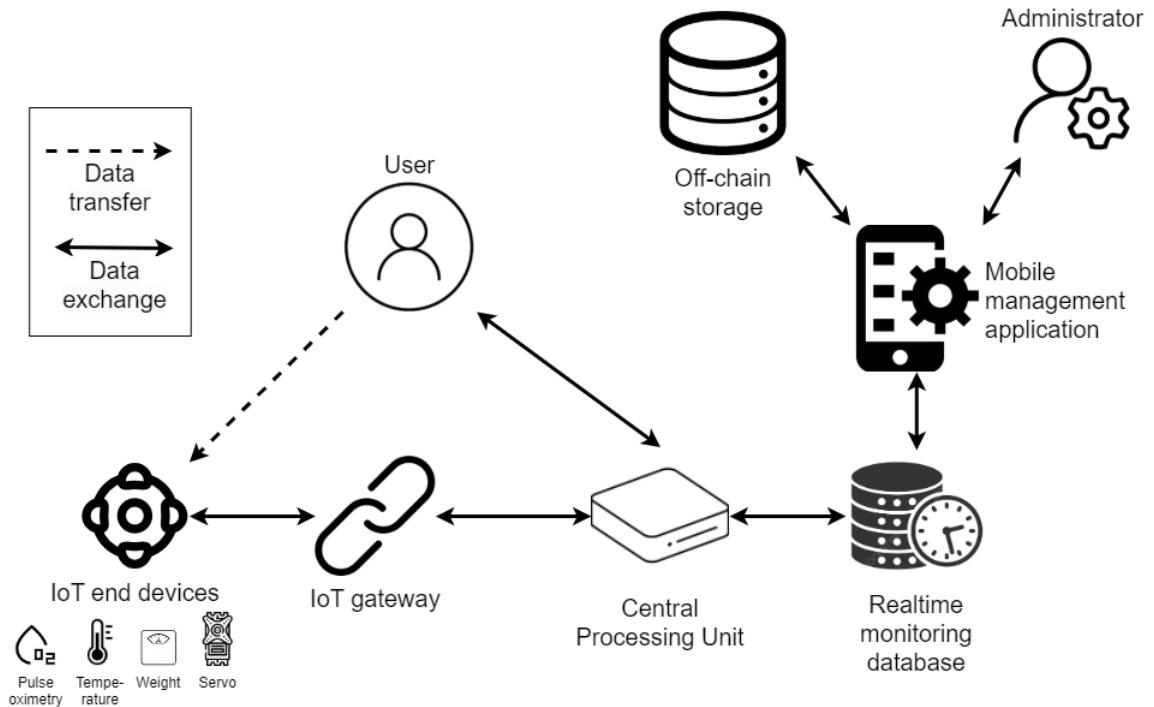


Figure 2.1: Thiết kế hệ thống ở mức nguyên mẫu ban đầu

Ở biểu đồ trên, hệ thống được thiết kế ban đầu với 8 module chính, gồm:

- **Các thiết bị IoT đầu cuối (IoT end devices):** Là các thiết bị, cảm biến dùng để thu thập thông số, dữ liệu đo được từ người sử dụng và giao tiếp các dữ liệu thu thập được này lên các hệ thống khác. Các thiết bị này bao gồm các loại cảm biến, các thiết bị kích hoạt, các vật dụng thông minh hay các thiết bị đeo thông minh. Một số ví dụ điển hình có thể nói đến các cảm biến đo nhiệt độ, đo độ ẩm, lượng mưa, các thông số sức khỏe từ thiết bị đeo, cảm biến đo độ ẩm đất, đo pH của đất trong các ứng dụng nông nghiệp. Đối với hệ thống hiện tại của dự án này, các thông số chúng ta cần thu thập bao gồm nhiệt độ, nồng độ oxy trong máu, nhịp tim và cân nặng của người dùng.
- **IoT gateway:** Đóng vai trò làm trung gian nhận dữ liệu từ các thiết bị IoT đầu cuối để đưa dữ liệu lên nền tảng lưu trữ dữ liệu đám mây. Ở đây, dữ liệu từ các thiết bị IoT được xử lý, sau đó truyền đến các hệ thống khác hoặc dịch vụ điện toán đám mây. Trong dự án này, IoT Gateway sẽ đóng vai trò

trung gian để điều khiển và tương tác giữa các thiết bị đầu cuối IoT với đơn vị xử lý trung tâm (Central Processing Unit). Các chức năng IoT gateway ngoài xử lý và gửi dữ liệu còn đảm bảo liên lạc giữa các thiết bị, giữa các hệ thống, đảm bảo tính bảo mật, quản lý cấu hình các thiết bị, tối ưu hóa lượng thông tin được đưa lên đám mây và chẩn đoán khi hệ thống gặp sự cố. [10]

Trong hệ thống này, IoT Gateway sẽ đóng vai trò trung gian giữa CPU và các thiết bị IoT đầu cuối, là thiết bị hỗ trợ trong việc giao tiếp, thu thập dữ liệu và điều khiển các thiết bị IoT đầu cuối cũng như giao tiếp các dữ liệu thu thập được từ các thiết bị trên để giao tiếp với CPU.

- **Central Processing Unit:** Central Processing Unit - Đơn vị xử lý trung tâm, trong hệ thống này là một máy tính (PC) thông thường, sẽ là trung tâm xử lý chính của toàn bộ hệ thống, đảm nhận việc phụ trách tương tác với người dùng (qua phần mềm và màn hình cảm ứng); tương tác với IoT Gateway để tiếp nhận và trao đổi dữ liệu từ các thiết bị IoT đầu cuối, cũng như giao tiếp với server và database thời gian thực để lưu trữ cũng như kiểm soát dữ liệu và các khía cạnh khác của hệ thống.
- **Cơ sở dữ liệu theo dõi thời gian thực (Realtime monitoring database):** Là nơi lưu trữ dữ liệu, làm trung gian lưu trữ và xử lý các luồng dữ liệu trong thời gian thực, thường là ngay lập tức sau khi dữ liệu được thu thập và gửi trước khi dữ liệu này được các ứng dụng bậc cao hơn nhận và xử lý sâu hơn. Loại hình cơ sở dữ liệu này đáp ứng nhu cầu về dữ liệu trong thời gian thực cho nền tảng khác thông qua các công cụ chuyên dụng cho việc đáp ứng cung cấp và xử lý dữ liệu trong khoảng thời gian cần thiết của ứng dụng.
- **Ứng dụng trên điện thoại cho việc quản lý (Mobile management application):** Trung gian cho người dùng quản trị hệ thống giao tiếp và tương tác với toàn bộ hệ thống thông qua dữ liệu thời gian thực nhận được từ đơn

vị xử lý trung tâm thông qua cơ sở dữ liệu thời gian thực.

- **Điểm lưu trữ ngoại hệ thống (Off-chain storage):** Lưu trữ các loại dữ liệu bên ngoài hệ thống xử lý, thường là những cơ sở dữ liệu truyền thống hay là các giải pháp lưu trữ đám mây được cung cấp sẵn. Hệ thống này thường dùng để lưu trữ lượng dữ liệu lớn hoặc các loại dữ liệu có độ nhạy cảm cao không cần lưu trữ trên hệ thống do giới hạn về mặt dung lượng hay vấn đề bảo mật. Hệ thống này cung cấp một giải pháp có thể mở rộng để lưu trữ lượng lớn dữ liệu tạo ra bởi các thiết bị IoT, giải tỏa lượng dữ liệu cần lưu trữ trên hệ thống xử lý, giúp truy cập dữ liệu nhanh hơn, giảm giá thành về mặt lưu trữ.

Trong hệ thống IoT, lưu trữ ngoại hệ thống thường được dùng để lưu trữ thông tin dữ liệu thu thập từ các cảm biến, dữ liệu cấu hình thiết bị, nhật ký lịch sử sử dụng thiết bị, thông tin người dùng, ... mà không cần thiết phải lưu trữ trên hệ thống xử lý nhưng vẫn cần thiết cho việc phân tích dữ liệu. Một số ví dụ điển hình có thể là lưu trữ dữ liệu trong hệ thống công nghiệp để phân tích và dự đoán trước khoảng thời gian để bảo trì.

- **Người dùng (User):** Tương tác với hệ thống, sử dụng hệ thống để lấy dữ liệu, tiếp nhận thông tin, ...
- **Người dùng quản trị (Administrator):** Điều khiển và theo dõi hệ thống thông qua Đơn vị xử lý trung tâm lẫn App quản lý thông qua điện thoại.

2.2 Kiến trúc hệ thống

Sau khi đã đưa ra được phương án thiết kế ban đầu như trên, bước tiếp theo nhóm sẽ tiến hành phân tích các phương án về mặt thiết bị và nền tảng phù hợp cho từng module và đưa ra sơ đồ về kiến trúc như sau:

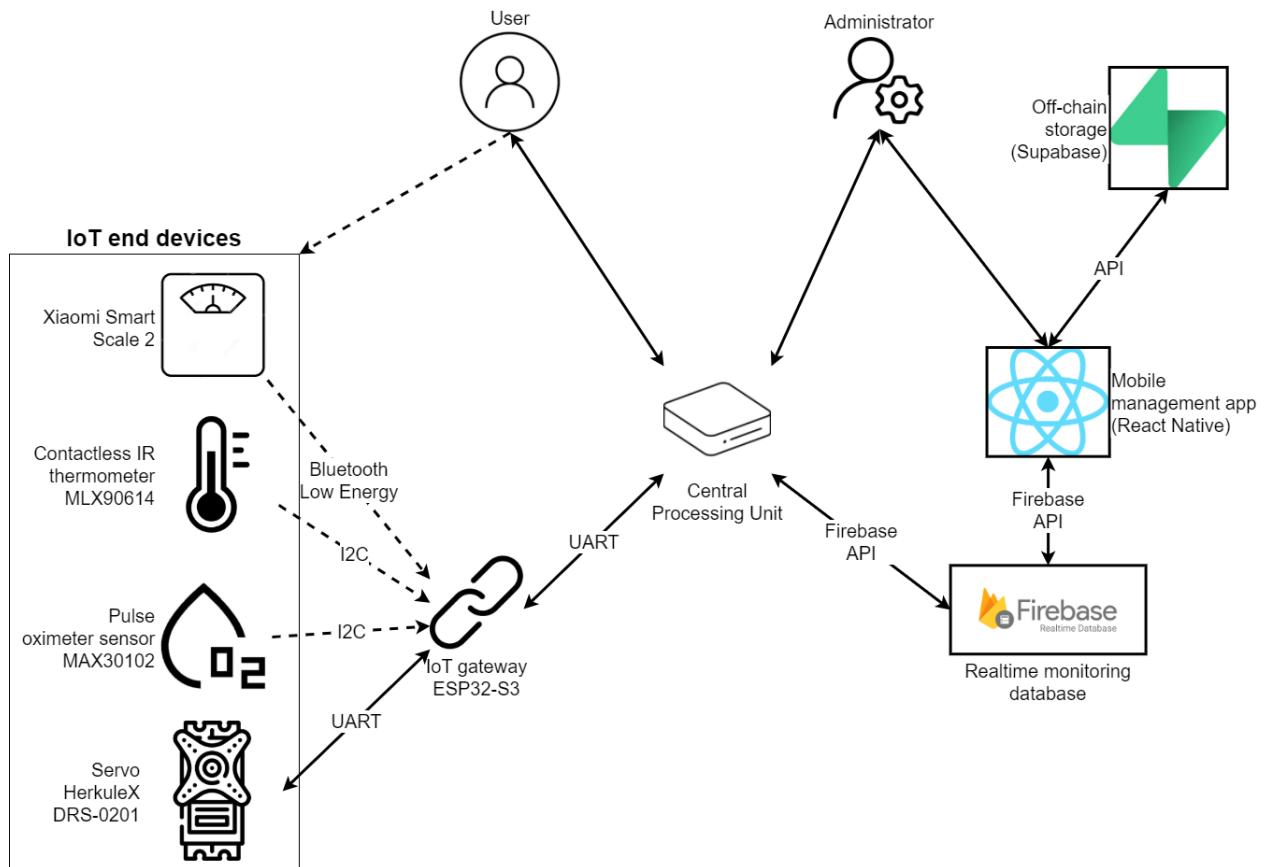


Figure 2.2: Sơ đồ kiến trúc của hệ thống.

Qua sơ đồ kiến trúc hệ thống được đưa ra như trên, các hệ thống của kiến trúc sẽ được phân tầng như sau:

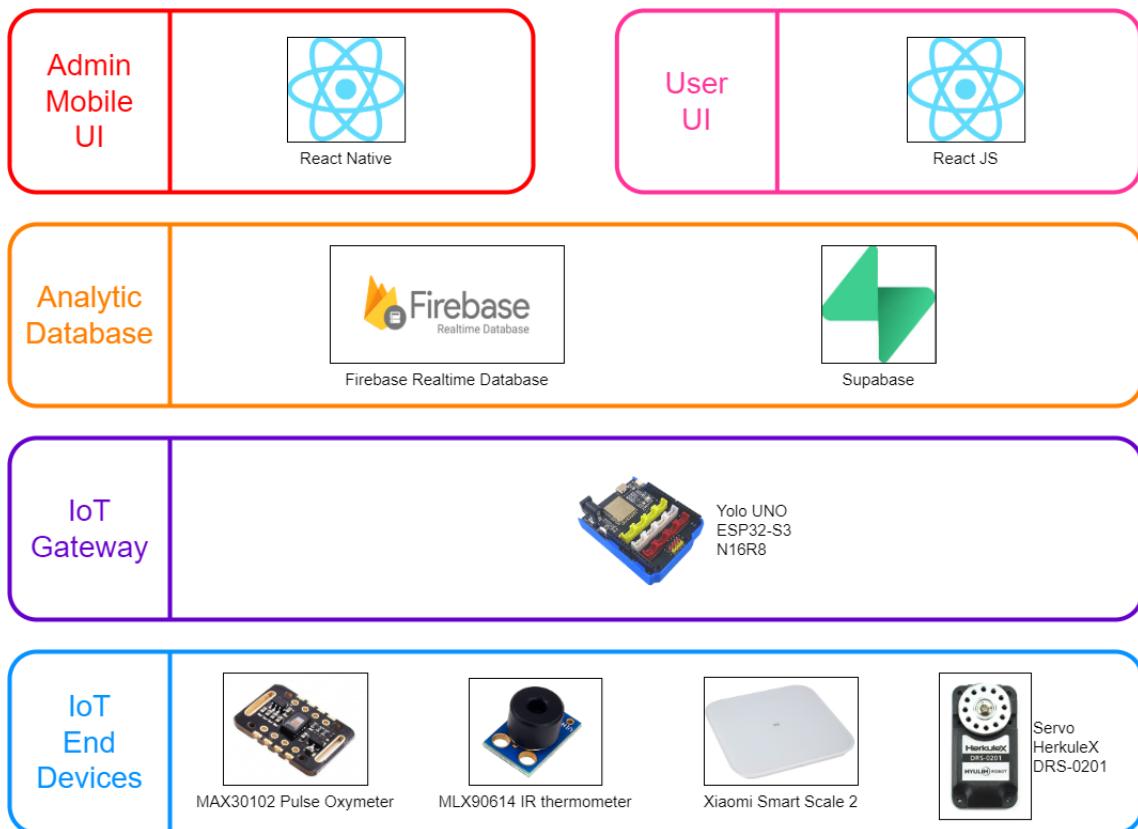


Figure 2.3: Sơ đồ phân tầng kiến trúc hệ thống.

Với từng module đã đưa ra như trên, để đáp ứng được yêu cầu của đề tài, các thiết bị đã được lựa chọn và giao tiếp với nhau như sau:

- **Đối với các thiết bị đầu cuối IoT:**
 - Đo nhiệt độ: Nhiệt kế hồng ngoại không chạm MLX90614.



Figure 2.4: Module GY-906 chứa cảm biến MLX90614.

Giới thiệu: module GY-906-BCC này có trung tâm là cảm biến nhiệt độ hồng ngoại không tiếp xúc từ Melexis: MLX90614. Được thiết kế để đo nhiệt độ mà không cần tiếp xúc trực tiếp, cùng với một bộ ADC 17 bit và xử lý tín hiệu điện tử mạnh mẽ giúp cảm biến này có độ chính xác và phân giải cao. Ngoài ra, kết nối thông qua I2C giúp cảm biến này có thể dễ tích hợp vào hệ thống. Cảm biến này có rất nhiều ứng dụng như đo nhiệt độ cơ thể hay cảm biến chuyển động.

Một số thông số kỹ thuật của cảm biến:

- * Điện thế đầu vào: 3-5V
- * Nhiệt độ đo: -40°C đến +125°C đối với nhiệt độ cảm biến và -70°C đến +380°C đối với nhiệt độ vật thể.
- * Độ chính xác: 0.5°C (trong mức nhiệt độ 0°C đến +50°C)
- * Mức phân giải: 0.02 °C
- * Giao tiếp: I2C
- * Trường nhìn (FoV): 35°
- * Ngoài ra cảm biến còn có chức năng “Bù trừ Gradien nhiệt” để nhiệt độ đo được tương đối ổn định so với sự thay đổi từ môi trường.

Lý do lựa chọn: nhiệt kế hồng ngoại MLX90614 đã có mặt trên thị trường từ lâu, giá thành rẻ, dễ tiếp cận, được hỗ trợ rộng rãi trên các nền tảng khác nhau. Cộng với độ chính xác cao (0.5°C trong mức nhiệt độ 0°C đến +50°C) và giao tiếp thông qua I2C giúp cho việc kết nối dễ dàng và thuận tiện hơn.

Phương án bị loại bỏ: Camera nhiệt tầm xa MLX90640 IR array



Figure 2.5: Module chứa cảm biến camera nhiệt MLX90640.

Lý do loại bỏ: mặc dù việc sử dụng camera nhiệt tầm xa có thể giúp việc đo nhiệt độ được trực quan hơn, (có thể) chính xác hơn và tầm xa hơn so với cảm biến hồng ngoại tầm gần trên, việc dữ liệu đưa vào lớn (vì dữ liệu được đưa vào là hình ảnh so với thông số) làm tăng lượng dữ liệu phải xử lý, cộng với giá thành cao (đắt hơn khoảng 10 lần) khiến cho việc sử dụng cảm biến này trong scope của dự án là không hợp lý.

- **Đo nhịp tim và nồng độ oxy trong máu: Cảm biến nhịp tim và nồng độ oxy trong máu MAX30102**



Figure 2.6: Module chứa cảm biến MAX30102.

Giới thiệu: module trên sử dụng IC MAX30102 - một IC hiện đại (kế thừa từ MAX30100) đo nhịp tim và nồng độ oxy trong máu đến từ Analog Devices. IC này kết hợp 2 đèn LED, một cảm biến ánh sáng, quang học được tối ưu hóa cũng như là xử lý tín hiệu analog nhiễu thấp để cảm biến được các tín hiệu về nhịp tim và nồng độ oxy trong máu. sau cửa sổ của IC là 2 đèn LED, một đèn LED đỏ và một đèn LED hồng ngoại. Ở phía bên kia là một cảm biến ánh sáng cực nhạy. Lý thuyết hoạt động của IC này là dựa trên việc chiếu sáng một đèn LED, cảm biến lượng ánh sáng phản chiếu lại cảm biến, và qua đó tính toán những thông số cần thiết.

Một số thông số kỹ thuật của cảm biến:

- * Điện thế đầu vào: 3.3-5V
- * Dòng điện tiêu thụ: $\sim 600\mu\text{A}$ (khi đo) / $\sim 0.7\mu\text{A}$ (trong chế độ chờ)
- * Bước sóng đèn LED đỏ: 660nm
- * Bước sóng đèn LED hồng ngoại: 880nm
- * Nhiệt độ hoạt động: -40°C đến $+85^\circ\text{C}$
- * Giao tiếp: I2C

Lý do lựa chọn: là một sản phẩm chuyên dụng dùng để đo nhịp tim và nồng độ oxy trong máu, MAX30102 là sản phẩm đời sau của cảm biến MAX30100, vì là đời sau nên cảm biến có một số cải thiện so với MAX30100, tuy nhiên giá thành không quá khác biệt, độ chính xác vẫn ổn định cộng với một số trực trắc khi sử dụng MAX30100 nên nhóm đã đưa ra quyết định sử dụng cảm biến này

Phương án bị loại bỏ: Cảm biến nhịp tim và nồng độ oxy trong máu MAX30100



Figure 2.7: Module chứa cảm biến MAX30100.

Lý do loại bỏ: như đã nói trên, ngoài trực trắc trong việc sử dụng cảm biến này khi tích hợp vào hệ thống, do là thế hệ trước của cảm biến MAX30102, cảm biến MAX30100 còn có một số bất lợi về mặt kỹ thuật như:

- * FIFO: cảm biến MAX30102 có bộ nhớ lưu trữ 32 mẫu dữ liệu, trong khi đó MAX30100 chỉ có thể lưu trữ 16 mẫu dữ liệu.
- * Độ phân giải ADC: MAX30102 có thể đọc tối 18 bit dữ liệu (lên đến 262144). Trong khi đó MAX30100 chỉ có độ phân giải ADC là 16 bit, nên MAX30102 có thể cảm nhận được những di chuyển nhỏ hơn.
- * Độ rộng xung của đèn LED: MAX30102 độ rộng xung của đèn LED nhỏ hơn MAX30100, giúp tiêu tốn năng lượng ít hơn.

Giới thiệu về giao thức giao tiếp I2C

Để kết nối với 2 cảm biến này, chúng ta sử dụng **giao thức I2C**.

I2C là một giao thức giao tiếp serial 2 dây sử dụng một dây serial dữ liệu (SDA) và một dây serial clock (SCL). Giao thức này hỗ trợ đa thiết bị nhận và truyền dữ liệu trên một cùng một bus giao tiếp. Giao tiếp

được gửi bằng gói byte với mỗi thiết bị có một địa chỉ độc nhất.

I2C là một giao thức được sử dụng rộng rãi do nhiều lý do khác nhau. Mặc dù tốc độ và khoảng cách giao tiếp thấp, giao thức này chỉ cần 2 dây để giao tiếp. Như nhiều giao thức khác, thì I2C cũng chứa đường dây serial cho dữ liệu và clock. I2C có thể kết nối với đa dữ liệu trên 1 bus với chỉ 2 đường dây này. Thiết bị điều khiển có thể giao tiếp với mọi thiết bị thông qua một địa chỉ I2C độc nhất được truyền thông qua đường serial dữ liệu. I2C qua đó rất đơn giản và tiết kiệm chi phí để các nhà sản xuất thiết bị tích hợp vào sản phẩm. [11]

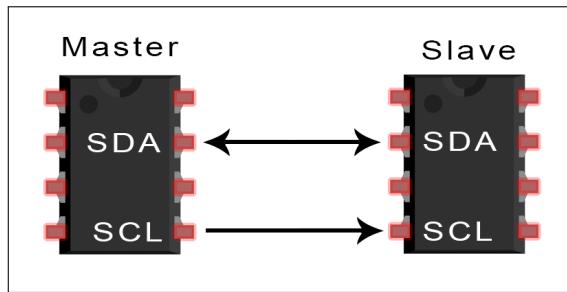


Figure 2.8: Cách kết nối giữa 2 thiết bị I2C.

I2C có lên đến 5 mức tốc độ khác nhau, với 2 mức tốc độ là Standard-mode (100kbps) và Fast-mode (400kbps) là hai mức tốc độ phổ biến nhất.

Chế độ I2C	Tốc độ bit tối đa
Standard-mode	100kbps
Fast-mode	400kbps
Fast-mode Plus	1Mbps
High-speed mode	3.4Mbps
Ultra-Fast mode	5Mbps

Cách gửi tin của I2C:

Với I2C, dữ liệu được truyền dưới dạng các *tin nhắn*. Các tin nhắn được chia ra thành các frame dữ liệu nhỏ. Mỗi tin nhắn có chứa một frame

địa chỉ chứa địa chỉ ở dạng nhị phân của slave, và có ít nhất một frame dữ liệu chứa dữ liệu cần truyền. Tin nhắn cũng bao gồm các điều kiện bắt đầu và dừng, các bit đọc/ghi, và các bit ACK/NACK giữa mỗi frame dữ liệu:

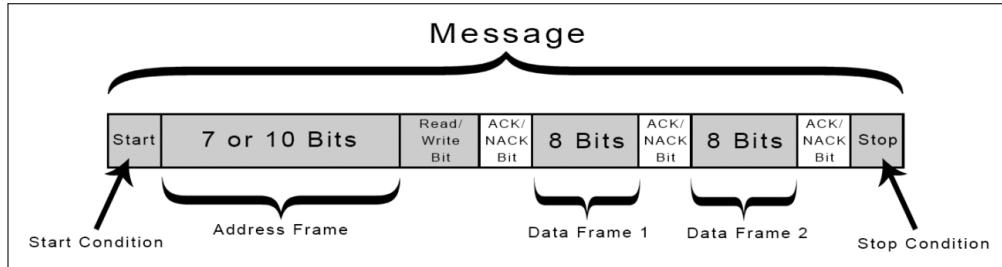


Figure 2.9: Một frame tin nhắn của I2C.

Điều kiện bắt đầu (Start Condition): Dây SDA đổi từ tích cực mức cao sang tích cực mức thấp TRƯỚC khi dây SCL đổi từ cao sang thấp.

Điều kiện kết thúc (Stop Condition): Dây SDA đổi từ tích cực mức thấp sang tích cực mức cao SAU khi dây SCL đổi từ thấp sang cao.

Frame địa chỉ (Address Frame): Một chuỗi từ 7-10 bit độc nhất cho mỗi slave để xác định slave khi master muốn giao tiếp với thiết bị này.

Bit Đọc/Ghi (Read/Write Bit): Một bit duy nhất xác định master đang gửi dữ liệu cho slave (tích cực mức thấp) hay yêu cầu dữ liệu từ slave (tích cực mức cao).

Bit ACK/NACK: Mỗi frame trong tin nhắn này sẽ được phản hồi với một bit acknowledge/no-acknowledge. Nếu frame địa chỉ hoặc frame dữ liệu được nhận thành công, một bit ACK sẽ được trả về cho thiết bị gửi từ thiết bị nhận. [12]

Qua đó, chúng ta có thể tóm gọn lại các ưu và nhược điểm của I2C như sau:

Ưu điểm:

- * Chỉ sử dụng 2 dây.
- * Truyền dữ liệu có đồng bộ với dây clock.

- * Hỗ trợ nhiều master và slave trên cùng 2 dây.
- * Có bit ACK/NACK để xác nhận gói tin.
- * Phần cứng không phức tạp như UART.
- * Được sử dụng rộng rãi và phổ biến.

Bất lợi:

- * Gửi dữ liệu Half-Duplex, do đó tốc độ chậm hơn UART và SPI.
- * Kích cỡ của frame dữ liệu bị giới hạn ở mức 8 bit.
- * Phần cứng cần hiện thực phức tạp hơn SPI.

– Đo cân nặng: Cân Xiaomi Smart Scale 2



Figure 2.10: Cân thông minh Xiaomi Smart Scale 2.

Giới thiệu: cân Xiaomi Smart Scale 2 là một thiết bị smarthome giúp theo dõi cân nặng, với cảm biến thép mangan độ chính xác cao, ngoài ra hỗ trợ Bluetooth 5.0 và Bluetooth Low Energy giúp kết nối dễ dàng. Ngoài cân cơ thể người cân cũng có chức năng cân nặng đồ vật.

Một số đặc điểm kỹ thuật:

- * Sử dụng 3 pin AAA
- * Trọng lượng cân từ 100gr đến 150kg
- * Cảm biến thép mangan độ chính xác cao, sai số ~50gr

- * Chất liệu nhựa ABS, mặt kính cùng với màn hình hiển thị LCD

- * Kết nối Bluetooth 5.0

Lý do lựa chọn: là một sản phẩm mới, với cách hoạt động đơn giản của hệ thống Bluetooth của cân Xiaomi Smart Scale 2, công nghệ mới (Bluetooth 5.0), giá thành rẻ dễ tiếp cận (~280.000 VNĐ), dễ sử dụng và đã kết nối, đã kết nối và nhận tín hiệu thành công bằng những nền tảng bluetooth IoT khác nhau.

Bluetooth Low Energy - BLE là gì?



Bluetooth LE là một giao thức Bluetooth được thiết kế cho các ứng dụng tiêu thụ cực thấp điện năng. Bản chất của Bluetooth là một công nghệ kết nối tầm gần sử dụng sóng radio làm trung gian giao tiếp. Tiêu chuẩn đầu tiên của công nghệ này là Bluetooth Classic, ban đầu được thiết kế để cung cấp sự kết nối không dây giữa các điện thoại hay những thiết bị khác. Được giới thiệu vào năm 2010 như là một phần của tiêu chuẩn Bluetooth 4.0, BLE được tối ưu hóa cho các ứng dụng có độ tiêu thụ điện năng cực thấp, và phục vụ cho thị trường các thiết bị sử dụng điện cung cấp từ pin. [13]

Các đặc điểm tiết kiệm điện năng của BLE:

- * Sóng radio của BLE không gửi tin liên tục, chỉ kích hoạt khi cần thiết.
- * Thời gian kết nối sóng radio của BLE ngắn, do số lượng kênh tần số radio ít.
- * Sóng radio của BLE sử dụng các gói tin bé hơn.

- * Tối ưu lượng pin sử dụng bằng cách giải phóng điện của pin theo xung.
- * Giao thức BLE có thiết kế không cân bằng, thiết bị có hiệu năng thấp sẽ có khối lượng công việc ít.

Stack của giao thức BLE

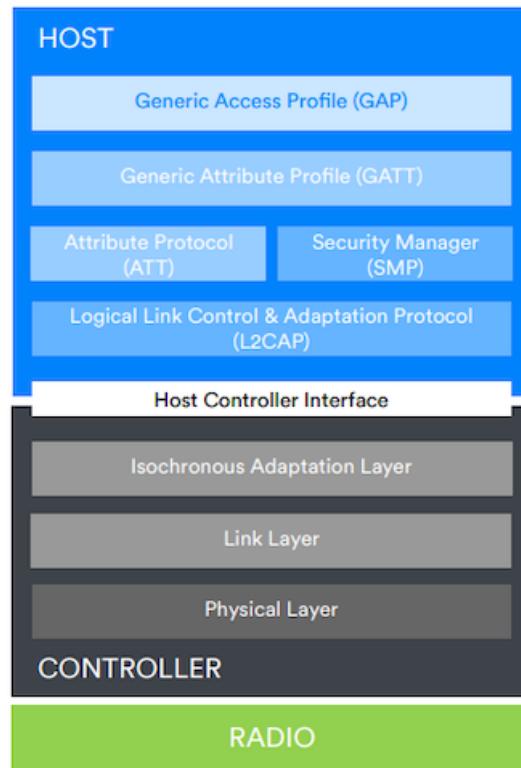


Figure 2.11: BLE Stack

Trong stack này, các lớp giao thức chúng ta cần quan tâm nhất chính là ATT, GATT và GAP.

- * **Attribute Protocol (ATT):** xác định cách một server tiết lộ dữ liệu cho một client như thế nào và cấu trúc của dữ liệu.
- * **Generic Attribute Protocol (GATT):** xác định định dạng của dữ liệu được thiết bị BLE tiết lộ, ngoài ra còn xác định những quy trình cần thiết để truy cập dữ liệu được thiết bị tiết lộ. Trong GATT có 2 vai trò (Roles) chính:
 - **Server** là thiết bị tiết lộ dữ liệu mà thiết bị kiểm soát hoặc chứa

và một số khía cạnh khác của mình các thiết bị khác có thể kiểm soát.

- **Client** là thiết bị tương tác với Server với mục tiêu đọc dữ liệu được Server tiết lộ và/hoặc kiểm soát hành vi của Server. Một thiết bị BLE có thể vừa là Server vừa là Client.

Ngoài ra, GATT còn 2 thành phần quan trọng bao gồm Services và Characteristic, với mỗi thành phần có một mã đặc biệt có tên là UUID riêng:

- **Services** là nhóm một hoặc nhiều các Thuộc tính (Attribute - một thuật ngữ cho bất kỳ loại dữ liệu được Server tiết lộ). Services dùng để nhóm các Attribute liên quan nhau lại để thực hiện một chức năng cụ thể của Server.
- **Characteristic** LUÔN là một phần của Service, thể hiện một phần của dữ liệu/thông tin mà Server muốn tiết lộ cho Client.

Trong BLE, có 6 loại tương tác với Characteristic:

1. Command (Mệnh lệnh)
2. Request (Yêu cầu)
3. Response (Phản hồi)
4. Notification (Thông báo)
5. Indication (Chỉ định)
6. Confirmation (Xác nhận)

* **Generic Access Profile (GAP):** cung cấp một framework xác định các thiết bị BLE tương tác với nhau như thế nào. GAP bao gồm:

- Vai trò của những thiết bị BLE, gồm:
 1. Broadcaster: thiết bị phát Advertisement và không nhận packet hoặc cho phép kết nối.
 2. Observer: thiết bị nghe những Advertisement từ các thiết bị khác nhưng không thiết lập kết nối với một thiết bị đang quảng bá.

3. Central: thiết bị khám phá và lắng nghe những thiết bị khác đang quảng bá. Central cũng có thể kết nối với một thiết bị đang quảng bá.

4. Peripheral: thiết bị phát Advertisement và nhận kết nối từ các thiết bị Central

- Advertisement (Những quảng bá) (Broadcast, Discovery, thông số Advertisement, dữ liệu Advertisement)
- Thiết lập kết nối (khởi động kết nối, chấp nhận kết nối, thông số kết nối)
- Bảo mật

Lưu ý rằng các thiết bị cũng có thể có nhiều vai trò cùng lúc. [14]

– **Servo điều khiển màn hình tương tác: Servo HerkuleX DRS-0201**



Figure 2.12: Servo HerkuleX DRS-0201.

Giới thiệu: Thông qua việc tái sử dụng robot của hãng Ohmnilabs, servo HerkuleX mà hãng đã sử dụng sẵn là một servo rất hiện đại và có tính module cao, tích hợp motor, hộp giảm tốc, mạch điều khiển và giao tiếp tích hợp toàn bộ trong một servo. Với motor DC kim loại có chổi

không lõi, servo này có thể tạo ra lực momen xoắn lên đến 24kg.cm ở mức điện thế 7.4V. Với việc mỗi servo có một địa chỉ riêng, hoặc người dùng điều chỉnh địa chỉ đó, servo này có thể sử dụng bus để điều khiển lên đến 253 servo cùng lúc. Mạch điều khiển thông minh của servo có nhiều chức năng và chế độ điều khiển, cho phép điều chỉnh tốc độ, báo trạng thái, giao tiếp UART full duplex, cảm biến quá tải, bảo vệ quá mạch, vv.

Một số đặc điểm kỹ thuật:

- * Motor DC kim loại có chổi không lõi
- * Khoảng góc xoay: 320° xoay liên tục
- * Độ phân giải (Góc xoay nhỏ nhất): 0.325°
- * Momen xoắn dừng: 24kg.cm (7.4v)
- * Tốc độ xoay tối đa: 0.147s/60° (7.4v)
- * Cân nặng: 60g
- * Giao tiếp: UART Full duplex(mức TTL), Packet nhị phân, Bus Multi Drop
- * Điều khiển tối đa lên đến 253 servo cùng lúc thông qua địa chỉ

Giới thiệu về kết nối UART

Servo HerkuleX DRS-0201 sử dụng giao thức UART (Universal Asynchronous Receiver-Transmitter), là một giao thức truyền tin serial cho phép 2 thiết bị giao tiếp với nhau.

Giao thức UART là một giao thức đơn giản và phổ biến, bao gồm hai đường truyền dữ liệu độc lập là Tx (truyền) và Rx (nhận). Dữ liệu được truyền và nhận qua các đường truyền này dưới dạng các khung dữ liệu (data frame) có cấu trúc chuẩn, với một bit bắt đầu (start bit), một số bit dữ liệu (data bits), một bit kiểm tra chẵn lẻ (parity bit) và một hoặc nhiều bit dừng (stop bit).

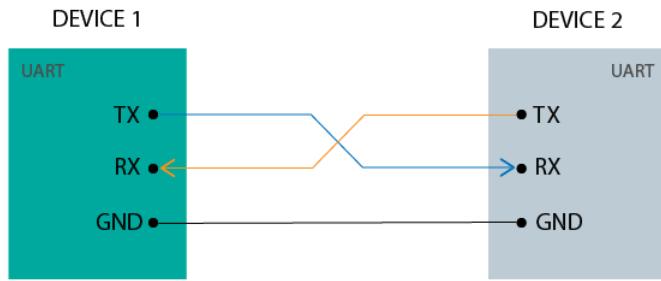


Figure 2.13: Cách kết nối dây giữa hai thiết bị UART

Không giống như SPI hay I2C (cả 2 đều có đồng bộ), UART là giao thức không đồng bộ, nên UART không có tín hiệu clock để đồng bộ việc truyền dữ liệu giữa các thiết bị. Tuy nhiên, cả hai thiết bị phải có sự đồng bộ về baud rate (tốc độ truyền dữ liệu).

Thông thường, tốc độ truyền của UART được đặt ở một số chuẩn, chẳng hạn như 9600, 19200, 38400, 57600, 115200 baud và các tốc độ khác. Tốc độ truyền này định nghĩa số lượng bit được truyền qua mỗi giây. Các tốc độ truyền khác nhau thường được sử dụng tùy thuộc vào ứng dụng và hệ thống sử dụng.

Ưu điểm và nhược điểm của giao tiếp UART:

- Ưu điểm

- * Chỉ cần dùng 2 dây truyền dữ liệu và chung đất
- * Không cần đến tín hiệu clock
- * Có 2 bit chẵn lẻ nên có thể kiểm tra lỗi dễ dàng
- * Cấu trúc gói dữ liệu có thể thay đổi được miễn là cả 2 bên đều được thiết lập để giao tiếp với nhau
- * Phương pháp giao tiếp UART có nhiều tài liệu hướng dẫn và cũng là bộ truyền dữ liệu đang được sử dụng rộng rãi hiện nay

- Nhược điểm

- * Kích thước của khung dữ liệu giới hạn tối đa là 9 bit, khá nhỏ so với nhu cầu sử dụng

- * Không được hỗ trợ nhiều hệ thống master và slave, chỉ có 2 thiết bị liên lạc với nhau
 - * Tốc độ truyền của mỗi giao tiếp UART phải nằm trong khoảng 10% của nhau
- **Đối với IoT gateway: ESP32-S3 N16R8**

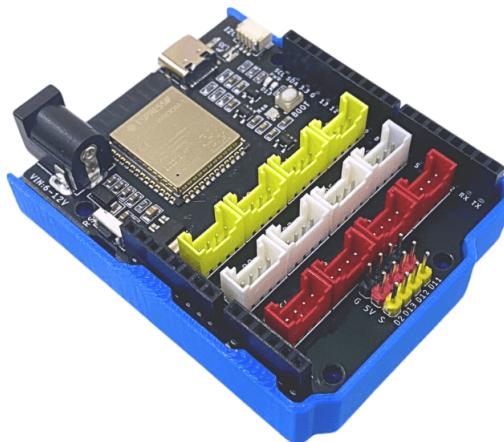


Figure 2.14: Module Yolo UNO với chipset ESP32-S3 N16R8.

Giới thiệu: ESP32-S3 là một MCU 2 nhân Xtensa LX7, với xung nhịp 240MHz. Với 512KB SRAM, MCU này còn đi kèm với kết nối WiFi 2.4 GHz, 802.11 b/g/n và Bluetooth 5 (LE) hỗ trợ cho việc kết nối tầm xa. MCU còn có 45 GPIO lập trình được và tập hợp nhiều kết nối hữu dụng. ESP32-S3 hỗ trợ cả flash SPI octal lớn hơn, nhanh hơn và PSRAM với dữ liệu có thể điều chỉnh được và cache tập lệnh.

Một số thông số kỹ thuật của MCU:

- Lõi: Xtensa dual-core 32-bit LX7 CPU, tần số lên đến 240MHz
- Bộ nhớ: 384 KB ROM / 512KB SRAM / 16MB Flash / 8MB PSRAM
- Tích hợp Wifi 2.4 GHz chuẩn 802.11 b/g/n
- Bluetooth (LE) 5.0

- Điện áp hoạt động: Từ 3V đến 3.6V DC
- Số chân I/O: 45 GPIO
- 2x12-bit ADC (lên đến 20 kênh)
- Giao diện giao tiếp: 2 giao diện I2C / 2 giao diện I2S / 4 giao diện SPI / 3 giao diện UART / 1 giao diện USB OTG
- Bảo mật: 4096 bit OTP / AES, SHA, RSA, ECC, RNG / Khởi động an toàn, Mã hóa Flash, Chữ ký số, Mô-đun HMAC
- Nhiệt độ hoạt động: -40 ~ 65 °C
- Trọng lượng: 12g

Lý do lựa chọn: Khả năng xử lý mạnh mẽ, kết nối rộng, giá thành rẻ cùng với hỗ trợ rộng rãi thông qua nền tảng Arduino phổ biến là những lý do lớn nhất vì sao nhóm lại quyết định chọn ESP32-S3 cho dự án này. ESP32-S3 là một trong những MCU hỗ trợ nền tảng Arduino mà nhóm quen thuộc, qua đó loại bỏ khâu học hỏi và tìm hiểu. Ngoài ra, kiến trúc quen thuộc của ESP32, tương đồng với chip đời trước là ESP8266 giúp cho việc sử dụng MCU này dễ dàng và tiện lợi hơn. Mặc dù mạnh mẽ, nhưng hiện nay giá thành của MCU này lại tương đối rẻ và dễ tiếp cận, phù hợp hơn với scope của dự án. Cộng với khả năng kết nối rộng, sự kết hợp giữa BLE và WiFi, thỏa mãn được yêu cầu của hệ thống, và khả năng xử lý thông tin dữ liệu mạnh với 2 nhân Xtensa nên MCU này là một lựa chọn phù hợp cho yêu cầu của hệ thống.

- **Đối với cơ sở dữ liệu theo dõi thời gian thực: Firebase Realtime Database**



Giới thiệu: Firebase Realtime Database là một cơ sở dữ liệu được lưu trữ trên đám mây. Dữ liệu được lưu trữ dưới dạng JSON và đồng bộ hóa trong thời gian thực tới tất cả mọi thiết bị kết nối. Khi xây dựng những ứng dụng liên nền tảng, tất cả mọi thiết bị client chia sẻ chung một Realtime Database sẽ đều nhận được cập nhật với dữ liệu mới nhất.

Ngoài ra, nền tảng này còn cung cấp một hệ thống ngôn ngữ chuyên dụng cho việc bảo mật - Firebase Realtime Database Security Rules, xác định cách dữ liệu được cấu trúc hay lưu trữ, viết hoặc ghi như thế nào, được truy cập ra sao. Cộng với dựa trên nền tảng cơ sở dữ liệu NoSQL và qua đó có những khả năng và tối ưu hóa khác so với một cơ sở dữ liệu thông thường. Hệ thống API của Firebase Realtime Database còn được thiết kế để cho phép các hoạt động có thể đáp ứng một cách nhanh chóng, qua đó giúp xây dựng một hệ thống thời gian thực có thể phục vụ nhiều người dùng mà không gây ảnh hưởng đến độ tốc độ phản hồi của hệ thống.

Các khả năng chính:

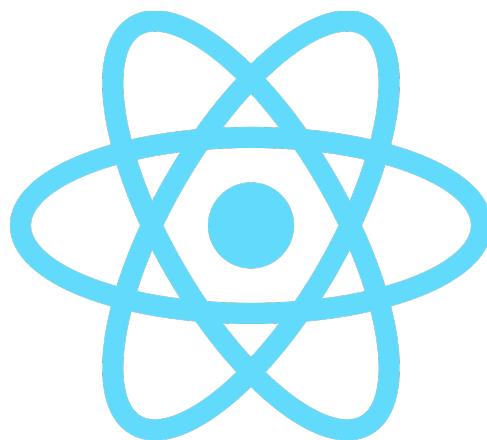
- Thời gian thực: thay vì các request HTTP thông thường, Firebase Realtime Database sử dụng đồng bộ dữ liệu-mỗi khi dữ liệu thay đổi, các thiết bị kết nối cũng nhận được cập nhật trong thời gian ngắn
- Ngoại tuyến: Các ứng dụng vẫn có thể giữ được khả năng phản hồi tốt do Firebase Realtime Database lưu trữ dữ liệu trên đĩa. Khi đã có thể

tái kết nối lại với nền tảng, thiết bị có thể cập nhật các thay đổi đã bỏ lỡ, qua đó đồng bộ hóa với trạng thái hệ thống hiện tại.

- Các thiết bị Client có thể truy cập dễ dàng: Firebase Realtime Database có thể truy cập trực tiếp từ một thiết bị cầm tay hay trình duyệt web; qua đó loại bỏ việc phải sử dụng một server ứng dụng riêng biệt. Bảo mật dữ liệu cũng có thể hiện thực dễ dàng thông qua Firebase Realtime Database Security Rules.

Lý do lựa chọn: Firebase RTDB được lựa chọn vì ở thời điểm thực hiện, Firebase RTDB là một trong những nền tảng cung cấp các công cụ, giải pháp lưu trữ thời gian thực, bảo mật toàn diện và tương đối dễ sử dụng, tích hợp và hiện thực nhất. Vì là một dịch vụ được cung cấp trên nền tảng đám mây, các dịch vụ được cung cấp mà không cần phải phụ thuộc vào một nền tảng lưu trữ nội bộ và có khả năng scalability tốt hơn. Ngoài ra, sự quen thuộc khi hiện thực hệ thống cũng là một nguyên nhân cho sự lựa chọn này.

- **Đối với ứng dụng điện thoại: React Native**



Giới thiệu: ReactJS là một thư viện JavaScript phổ biến sử dụng để xây dựng giao diện người dùng (UI). Với triết lý "learn once, write anywhere", ReactJS cho phép các nhà phát triển xây dựng ứng dụng web và di động một cách hiệu quả và linh hoạt. Một số đặc điểm nổi bật của ReactJS bao gồm:

- Component-Based Architecture: ReactJS sử dụng các thành phần (component) tái sử dụng được, giúp việc phát triển và duy trì code dễ dàng hơn.
- Virtual DOM: Tăng cường hiệu suất bằng cách chỉ cập nhật những phần của giao diện thực sự thay đổi.
- JSX Syntax: Kết hợp JavaScript và HTML, giúp viết code trực quan hơn.
- Môi trường đa dạng: Hỗ trợ việc tích hợp với nhiều công nghệ khác trong hệ sinh thái.

Lý do lựa chọn: Nhóm lựa chọn Android studio thay vì các nền tảng khác bởi các lợi ích sau:

- Thư viện hỗ trợ giao tiếp với serial port: Reactjs có thư viện như node-serialport để giao tiếp với cổng nối tiếp. ReactJS thường kết hợp với Node.js để xử lý phần backend, do đó việc gửi dữ liệu qua serial port sẽ hoạt động hiệu quả khi sử dụng Node.js.
- Tích hợp Firebase: ReactJS cung cấp nhiều thư viện hỗ trợ Firebase, như Firebase SDK hoặc React-Firebase-Hooks, dễ dàng kết nối và quản lý dữ liệu. có thể thiết lập các tính năng như lưu trữ dữ liệu cảm biến, hoặc hiển thị dữ liệu thời gian thực từ Firebase trên giao diện Reactjs.
- Xử lý đồng bộ: Kết hợp việc gửi dữ liệu đến serial port và Firebase trong một flow hợp lý, đảm bảo rằng dữ liệu được xử lý đồng bộ và không bị trễ. Sử dụng cú pháp async/await trong JavaScript được đảm bảo các yêu cầu xử lý tuần tự.

• **Đối với lưu trữ ngoại hệ thống: Supabase**

Trong đồ án này, hệ thống được thiết kế sử dụng Supabase như một nền tảng lưu trữ dữ liệu và cung cấp dịch vụ Backend. Supabase là một giải pháp mã nguồn mở phát triển dựa trên cơ sở dữ liệu quan hệ PostgreSQL,

được tích hợp sẵn nhiều chức năng hiện đại như xác thực người dùng, quản lý tệp tin, đồng bộ dữ liệu thời gian thực và serverless functions. Việc lựa chọn Supabase thay thế cho các hệ quản trị NoSQL truyền thống đến từ nhu cầu về tính toàn vẹn dữ liệu, khả năng mở rộng, cũng như sự thuận tiện trong việc triển khai và bảo trì hệ thống.

Một điểm mạnh nổi bật khác của Supabase là khả năng theo dõi dữ liệu theo thời gian thực. Bằng cách hỗ trợ cơ chế lắng nghe sự kiện thông qua WebSocket, hệ thống có thể tự động phản hồi với các thay đổi phát sinh từ cơ sở dữ liệu mà không cần thực hiện kiểm tra định kỳ (polling), từ đó cải thiện hiệu năng và giảm độ trễ trong truyền tải thông tin.

Ngoài ra, Supabase còn tích hợp sẵn hệ thống xác thực người dùng với các cơ chế bảo mật hiện đại như xác thực qua email, OTP để đảm bảo dữ liệu chỉ được truy xuất bởi đúng đối tượng có quyền.



Một số ưu điểm nổi bật:

- Đáp ứng tốt các yêu cầu về bảo mật và tính toàn vẹn dữ liệu.
- Hạn chế thời gian triển khai backend nhờ nhiều tính năng tích hợp sẵn.
- Hỗ trợ tốt cho các ứng dụng có yêu cầu cập nhật thời gian thực.
- Có thể tự triển khai trên máy chủ riêng hoặc sử dụng phiên bản cloud tùy theo nhu cầu.

Lý do lựa chọn: Với những ưu điểm trên, Supabase được đánh giá là giải pháp phù hợp cho các dự án phát triển nhanh, cần đảm bảo khả năng mở rộng, đồng thời vẫn duy trì tính ổn định và bảo mật của hệ thống. Việc sử dụng Supabase trong dự án này không chỉ giúp rút ngắn thời gian phát triển mà còn tạo điều kiện thuận lợi để mở rộng các tính năng trong tương lai mà không cần thay đổi kiến trúc lưu trữ ban đầu.

Chương 3

Hiện thực hệ thống

Trong chương 3, chương này sẽ tập trung vào việc áp dụng những phương án thiết kế về hệ thống, phần cứng được trình bày ở chương trước vào việc lựa chọn các nền tảng giao tiếp giữa các thiết bị, giữa các lớp hiện thực với nhau và lập trình cho phần cứng, phần mềm để tạo ra một hệ thống hoàn chỉnh.

3.1 Hiện thực phần cứng

Quá trình hiện thực phần cứng của dự án bao gồm:

- Kết nối, thu thập dữ liệu từ các thiết bị IoT đầu cuối.
- Xử lý dữ liệu thu thập được.
- Đẩy dữ liệu lên cơ sở dữ liệu theo dõi theo thời gian thực.

Trong quá trình hiện thực trên phải đảm bảo được tính vẹn toàn của dữ liệu, độ chính xác cũng như tối ưu về mặt hiệu năng và độ ổn định của hệ thống. Qua đó, quá trình hiện thực phần cứng được thực hiện thông qua các bước thực hiện như sau:

1. Xác định các module phụ cần hiện thực

2. Hiện thực và tối ưu hóa từng module phụ

3. Kết hợp các module phụ thành thể thống nhất-hệ thống hoàn chỉnh

Qua đó, bước đầu tiên cần thực hiện chính là việc xác định các module phụ mà phần cứng của hệ thống bao gồm những gì. Và toàn bộ hệ thống sẽ được tích hợp thông qua các hàm cần sử dụng, các thư viện phụ thuộc và các định nghĩa được hiện thực trong thư viện `system_op.h` và `system_op.c` nhằm giảm thiểu sự rối loạn trong file `main`.

3.1.1 Nền tảng lập trình

Để thực hiện lập trình, dự án sẽ sử dụng nền tảng lập trình PlatformIO trên Visual Studio Code cho việc lập trình toàn bộ các phần lập trình liên quan đến phần cứng của dự án.



PlatformIO được lựa chọn là nền tảng cho việc lập trình phần vì nhiều lý do như sau:

- Trang chủ của PlatformIO nói rằng: "PlatformIO là một môi trường phát triển tích hợp rộng, thân thiện với người dùng với các bộ phát triển chuyên nghiệp, cung cấp những tính năng mạnh mẽ và hiện đại để tăng tốc cũng như đơn giản hóa quy trình tạo ra và bàn giao sản phẩm nhúng."
- PlatformIO có thể dễ dàng cài đặt trên phần mềm lập trình Visual Studio Code thông dụng, hỗ trợ đa nền tảng lập trình như Arduino hay ESP-IDF của ESP32. Với ngôn ngữ lập trình dựa trên nền tảng C/C++ thông dụng cho việc lập trình nhúng, PlatformIO hỗ trợ tốt cho việc lập trình này.
- PlatformIO Core (CLI), trái tim của PlatformIO, bao gồm rất nhiều công cụ hữu ích cho việc lập trình một hệ thống nhúng như:

- Hệ thống build đa nền tảng
- Hệ thống quản lý package thống nhất
- Quản lý thư viện
- Công cụ tìm kiếm các thư viện phụ thuộc
- Công cụ theo dõi cổng Serial
- Các công cụ tích hợp (các IDE đám mây / desktop và Tích hợp liên tục)

3.1.1.1 Bước 1: Xác định các module phụ cần hiện thực

Với yêu cầu của dự án, thông qua các thiết bị và nền đã lựa chọn ra ở phần thiết kế, các module phụ cần hiện thực trước khi đến khâu ghép các module phụ này thành hệ thống hoàn chỉnh bao gồm:

- Kết nối đến cảm biến nhiệt độ MLX90614 và cảm biết nhịp tim + nồng độ oxy máu MAX30102. Tối ưu hóa hoạt động của từng cảm biến (nếu có thể).
- Kết nối BLE với cân Xiaomi Smart Scale 2 và lấy dữ liệu.
- Kết nối và điều khiển servo HerkuleX DRS-0201. Tối ưu hóa tốc độ xoay của servo.
- Thiết lập giao thức trao đổi thông tin với Đơn vị xử lý trung tâm (CPU).

Với các module phụ nhỏ và rời rạc trên, hiện thực từng module một cách riêng biệt, sau đó đến giai đoạn cuối tổng hợp lại sẽ giúp cho hệ thống giảm thiểu khả năng gặp lỗi, dễ dàng chẩn đoán lỗi ở module nào, dễ dàng tối ưu hóa, kiểm thử hệ thống ở những module cần thiết.

Tất cả các module nhỏ này sẽ được hiện thực thông qua các library bọc (library wrapper) với các function được thiết kế để đơn giản hóa việc sử dụng code và giúp cho code sạch và dễ hiểu hơn. Các thư viện này sẽ có phần _op ở phần đuôi tên.

3.1.1.2 Bước 2: Hiện thực và tối ưu hóa từng module phụ

Trong quá trình hiện thực hệ thống, với sự hỗ trợ của thầy Lê Trọng Nhân, thầy đã cung cấp robot **Ohmni Telepresence** của Ohmnilabs để sử dụng một phần làm thành trạm cân hoàn chỉnh, cùng với việc thầy hỗ trợ về mặt hoàn thiện phần để trạm cân cũng như các loại chi phí. Với robot này, nhóm sẽ sử dụng tất cả từ phần cột trở lên, bao gồm loa, màn hình, camera, servo xoay assembly màn hình + camera, và các phần khác. Phần nhóm không sử dụng là phần base có chứa các loại mạch cấp nguồn, xử lý, pin, bánh xe, ...



Figure 3.1: Robot Ohmni Telepresence.

Hệ thống bao gồm:

- Một màn hình IPS cảm ứng, kích cỡ 10.1", độ phân giải 1280x800. Sử dụng nguồn cấp từ cổng Micro USB cũng như sử dụng cổng này cho tín hiệu cảm ứng, sử dụng cổng HDMI để truyền tín hiệu hình ảnh.
- Một loa ở phần giữa thân máy, kết nối thông qua USB-A.
- Một camera See3CAM_CU135 của hãng e-con Systems, kết nối thông qua USB-C.

- Servo HerkuleX DRS-0201 đã giới thiệu trên.

3.1.1.2.a Kết nối đến 2 cảm biến MLX90614 và MAX30102 và tối ưu hóa cảm biến.

Thực hiện kết nối đến 2 cảm biến:

- Cảm biến nhiệt độ hồng ngoại MLX90614: kết nối đơn giản bằng I2C thông qua thư viện Adafruit_MLX90614.h tại repository [Adafruit-MLX90614-Library](#) trên nền tảng GitHub.
- Cảm biến nhịp tim và nồng độ oxy máu MAX30102: kết nối thông qua bus I2C. Tuy nhiên, để sử dụng cảm biến này có 2 thư viện:
 - Thư viện MAX30105.h của repository [SparkFun_MAX3010x_Sensor_Library](#) trên GitHub.
 - Thư viện max30102.h của repository [MAX30102_by_RF](#) trên GitHub.

Qua [nguồn tài liệu](#) của tác giả thư viện [MAX30102_by_RF](#), thư viện có một số đặc điểm nổi trội hơn so với thư viện ban đầu của hãng SparkFun như sau:

- Thay vì sử dụng thư viện SoftI2C phụ thuộc vào phần cứng thì chuyển sang thư viện Wire được sử dụng rộng rãi và tổng quát hơn.
- Có giải thuật tính toán kết quả với độ chuẩn xác cao hơn rất nhiều so với giải thuật của SparkFun (sẽ được trình bày sau).

Tuy nhiên, trong quá trình kết nối 2 thiết bị, có **một vấn đề gặp phải khi kết nối cả hai cảm biến MLX90614 và MAX30102 trên cùng 1 bus I2C**. Điểm quan trọng chính là, lựa chọn đầu tiên của dự án này để đo nhịp tim và nồng độ oxy trong máu là cảm biến MAX30100, và vấn đề này đều hiện hữu trên cả hai cảm biến MAX30100 lẫn MAX30102.

Khi kết nối cả hai cảm biến trên vào cùng một bus I2C, 2 cảm biến MAX30100 lẫn MAX30102 đều chiếm toàn bộ bus I2C, khiến cho cảm biến MLX90614

không thể hoạt động hoặc gửi dữ liệu, hoặc ngược lại, 2 model cảm biến MAX sẽ không thể tương tác vì báo lỗi không có bus I2C. Nguyên nhân vì sao hiện vẫn chưa được nắm rõ, do datasheet của cả 2 IC này đều không nhắc đến việc này.

Phương hướng giải quyết: Mặc dù sử chung 1 bus I2C sẽ dẫn đến vấn đề nói trên, ESP32-S3 hỗ trợ lên đến 2 bus I2C, do đó, bằng việc mở thêm 1 bus I2C và kết nối cảm biến MLX90614 giúp cho cả 2 cảm biến đều hoạt động bình thường và có thể gửi dữ liệu cho ESP32-S3.

Cấu trúc lập trình của hai cảm biến MLX90614 và MAX30102:

- **Cảm biến MLX90614:** Cảm biến MLX90614 sẽ có một wrapper library tương ứng là MLX90614_op. Trong thư viện này sẽ bao gồm các phần sau:

- Các define chân I2C và địa chỉ của cảm biến, bao gồm các define cho 2 chân SDA, SCL và địa chỉ I2C của cảm biến:

```
1 #define MLX90614_SDA 7 // MLX90614 I2C SDA pin  
2 #define MLX90614_SCL 6 // MLX90614 I2C SCL pin  
3 #define MLX90614_ADR 0x5A // MLX90614 I2C address
```

- function uint8_t initMLX90614(): Một function để khởi động cảm biến MLX90614, và mở một bus I2C riêng, trả về true khi khởi động thành công và false khi khởi động không thành công.
 - function void updateTempData(): Function cập nhật dữ liệu nhiệt độ mới nhất.
 - function float getObjTemp(): Function trả về nhiệt độ của vật thể trước cảm biến hiện tại.
 - function float getAmbTemp(): Function trả về nhiệt độ nhiệt độ xung quanh (nhiệt độ môi trường) quanh cảm biến.

- **Cảm biến MAX30102:** Cảm biến MAX30102 sẽ có một wrapper library tương ứng là MAX30102_op. Trong thư viện này sẽ bao gồm các phần sau:

- Các define chân I2C và địa chỉ của cảm biến, bao gồm các define cho 2 chân SDA, SCL và chân Interrupt của cảm biến MAX30102:

```

1 #define MAX30102_INT 10 // MAX30102 interrupt pin
2 #define MAX30102_SDA 11 // MAX30102 I2C SDA pin
3 #define MAX30102_SCL 12 // MAX30102 I2C SCL pin

```

- function `uint8_t initMAX30102()`: Một function để khởi động cảm biến MAX30102, sử dụng bus I2C mặc định của ESP32-S3, kích hoạt chân nhận Interrupt của cảm biến, trả về `true` khi khởi động thành công và `false` khi khởi động không thành công.
- function `void updatePoxData()`: Cập nhật dữ liệu của cảm biến MAX30102, trong hàm này, lệnh sẽ được gọi liên tục, ghi nhận dữ liệu nhận được từ cảm biến MAX30102 khi chân Interrupt được kích hoạt, sau đó tính toán dữ liệu nhận được từ các mẫu đã lấy được, sử dụng giải thuật của RF.
- function `int getBPM()`: Trả về giá trị BPM (nhịp tim/phút) sau khi đã tính toán được.
- function `float getSpO2()`: Trả về giá trị SpO2 (nồng độ oxy máu) sau khi đã tính toán được.
- function `bool validBPM()`: Kiểm tra xem dữ liệu BPM nhận được từ cảm biến có phải là dữ liệu sử dụng được/hợp lệ hay không. True khi dữ liệu đúng, và false khi dữ liệu không hợp lệ.
- function `bool validSpO2()`: Kiểm tra xem dữ liệu SpO2 nhận được từ cảm biến có phải là dữ liệu sử dụng được/hợp lệ hay không. True khi dữ liệu đúng, và false khi dữ liệu không hợp lệ.

Kiểm tra độ chính xác và tối ưu hóa dữ liệu từ 2 cảm biến:

- **Cảm biến nhiệt độ hồng ngoại MLX90614:** Với cảm biến MLX90614, việc kiểm tra được thực hiện thông qua việc sử dụng nước đá, nước sôi

và đo nhiệt độ trán của người trưởng thành khỏe mạnh ở điều kiện thông thường.

Kết quả:

- Đo nước đá: 0 độ C
- Đo nước sôi: ~100 độ C
- Đo nhiệt độ trán: Dao động ở mức 32.8 độ C - 33.8 độ C. Dựa theo bảng so sánh nhiệt độ trán so với nhiệt độ cơ thể của hằng feevr dưới đây, nhiệt độ như vậy được xem là nhiệt độ bình thường và không có gì đáng lo ngại. [15]

Table 3.1: Bảng so sánh nhiệt độ bề mặt da so với nhiệt độ lõi cơ thể của feevr (theo độ C) [16]

Nhiệt độ bề mặt da	Nhiệt độ lõi cơ thể	Độ khác biệt
Nhiệt độ bề mặt da dưới 32.5 độ C là do những yếu tố ngoại cảnh(*)		
32.5	36.5	4.0
32.9	36.7	3.7
33.4	36.8	3.5
33.8	37.0	3.2
34.3	37.2	2.9
34.7	37.3	2.6
35.1	37.5	2.4
35.6	37.7	2.1
36.0	37.8	1.8
36.5	38.0	1.5
36.9	38.2	1.3
37.3	38.3	1.0
37.5	38.5	1.0
37.7	38.7	1.0
37.8	38.8	1.0

38.0	39.0	1.0
38.2	39.2	1.0
38.3	39.3	1.0
38.5	39.5	1.0
38.7	39.7	1.0
38.8	39.8	1.0
	Nhiệt độ trung bình	
	Cần cẩn trọng	
	Tiếp cận ngưỡng sốt	
	Cần đo đặc kỹ hơn	

(*): Các yếu tố ngoại cảnh bao gồm (nhưng không giới hạn) nhiệt độ không khí trong phòng / ngoài trời, lượng cồn trong cơ thể, tập luyện thể dục, gió lùa và / hoặc ánh sáng trực tiếp.

→ Suy ra, kết quả của cảm biến nhiệt độ MLX90614 tương đối chính xác.

- **Cảm biến nhịp tim và nồng độ oxy máu MAX30102:** Đôi với cảm biến nhịp tim và nồng độ oxy máu, trước tiên chúng ta sẽ so sánh mức độ biến thiên và độ lệch của dữ liệu giữa 2 giải thuật, giải thuật của SparkFun so với giải thuật RF. Dưới đây là 2 biểu đồ thể hiện sự so sánh giữa dữ liệu thu thập được từ 2 giải thuật khác nhau, cả hai đều có chung thiết lập và lấy 100 mẫu thử.

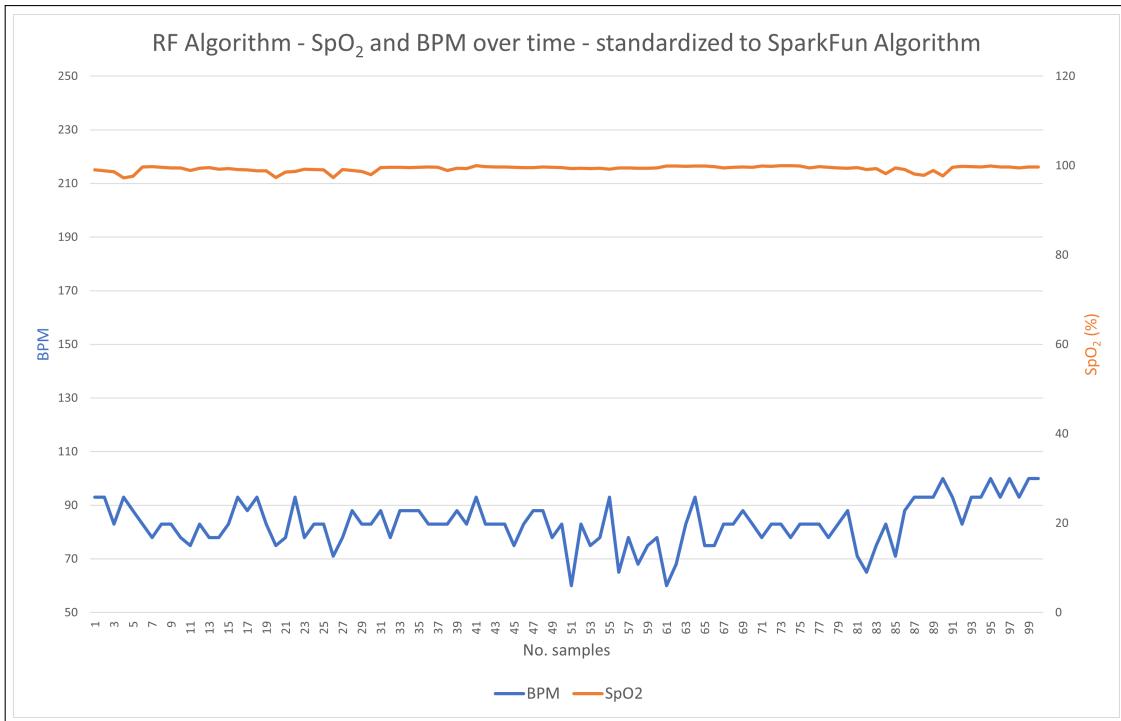


Figure 3.2: Biểu đồ 100 mẫu thử sử dụng giải thuật RF.

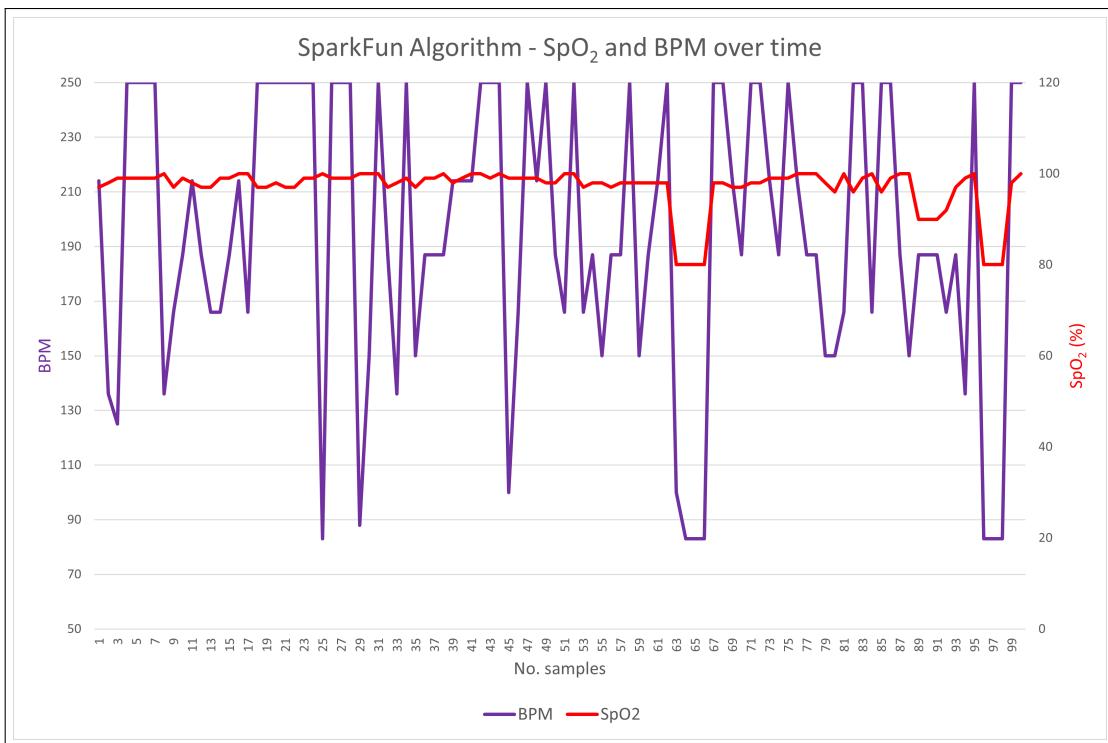


Figure 3.3: Biểu đồ 100 mẫu thử sử dụng giải thuật SparkFun.

Thông qua 2 biểu đồ trên, ta có thể thấy rõ sự khác biệt giữa 2 giải thuật, với giải thuật RF có độ ổn định về mặt dữ liệu tốt hơn rất nhiều so với giải thuật

SparkFun, đặc biệt là ở thông số nhịp tim / phút (BPM), ngoài ra cũng có sự bất ổn ở thông số độ bão hòa Oxy (SpO2) với giải thuật của SparkFun.

Về khía cạnh độ chính xác, tiếp tục đo thông số của người trưởng thành khỏe mạnh ở điều kiện thông thường, với SpO2 loanh quanh trong khoảng 97% - 100%, trong khoảng mức nồng độ oxy máu thông thường của người lớn khỏe mạnh. [17][18][19]

Table 3.2: Nồng độ oxy máu tiêu chuẩn. [17]

	Nồng độ oxy máu	Phân tích khí máu động mạch (ABG)
Mức khỏe mạnh	95 - 100%	75 - 100 mmHg
Cần sự can thiệp y tế	<95%	<74 mmHg

Đối với nhịp tim, theo biểu đồ của giải thuật RF, nhịp tim dao động ở mức 60-100, so với biểu đồ sau đây của trang heart.org; thì nhịp tim này là mức nhịp tim/phút tương đối bình thường của người khỏe mạnh khi có và không làm những hoạt động có cường độ trung bình (50-70% mức nhịp tim/phút tối đa) và hoạt động thể chất cường độ cao (70-85% mức nhịp tim/phút tối đa). [45][46]

Đặc biệt, khi sử dụng máy đo huyết áp có chức năng đo nhịp tim có độ chính xác ở mức y khoa từ hãng Omron cho một người lớn tuổi có bệnh nền và nhịp tim tương đối thấp (60-65 bpm), kết quả của cảm biến sử dụng giải thuật của RF cũng có độ chính xác gần như tương đương, chênh lệch khoảng 2-3 bpm so với máy đo huyết áp.

Table 3.3: Mức nhịp tim/phút (BPM) mong đợi theo độ tuổi. [45]

Độ tuổi	Mức nhịp tim 50-85%	Mức nhịp tim (dự đoán) tối đa
20 tuổi	100-170 bpm	200 bpm
30 tuổi	95-162 bpm	190 bpm

35 tuổi	93-157 bpm	185 bpm
40 tuổi	90-153 bpm	180 bpm
45 tuổi	88-149 bpm	175 bpm
50 tuổi	85-145 bpm	170 bpm
55 tuổi	83-140 bpm	165 bpm
60 tuổi	80-136 bpm	160 bpm
65 tuổi	78-132 bpm	155 bpm
70 tuổi	75-128 bpm	150 bpm

→ Suy ra, kết quả của cảm biến nhịp tim và nồng độ oxy máu, sử dụng giải thuật RF có độ chính xác khá cao.

3.1.1.2.b Kết nối BLE với cân Xiaomi Smart Scale 2

Thiết lập kết nối BLE với cân Xiaomi Smart Scale 2

Máy trạng thái cho chu trình ESP32-S3 kết nối, tiếp nhận và xử lý dữ liệu từ cân Xiaomi Smart Scale 2 như sau:

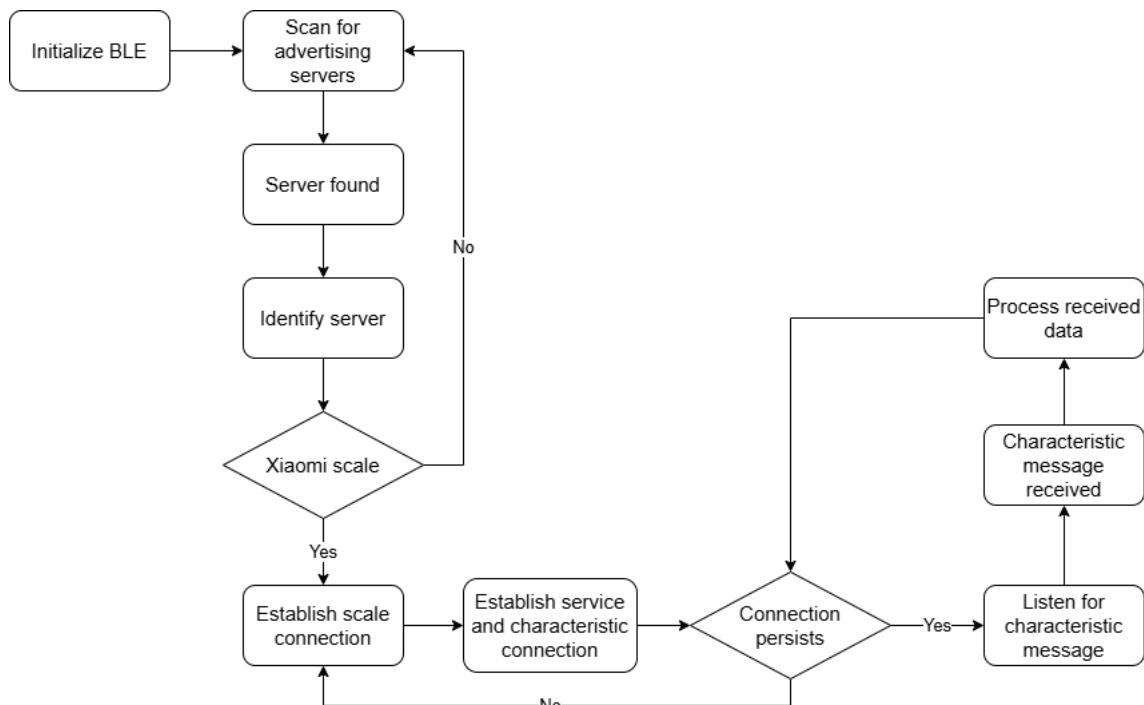


Figure 3.4: FSM vận hành và kết nối BLE đến cân Xiaomi

Trong hệ thống kết nối BLE với cân Xiaomi Smart Scale 2, như đã giới thiệu về các lớp giao tiếp trên, cân Xiaomi Smart Scale 2 hoạt động như một Broadcaster, liên tục gửi dữ liệu đo được ra môi trường xung quanh, trong khi đó, ESP32-S3 là một Client. Dữ liệu mà cân Xiaomi gửi ra môi trường xung quanh có một số đặc điểm thú vị như sau:

- Kết nối đến cân *không hề* có mã hóa.
- Cân gửi dữ liệu cân nặng thông qua frame quảng bá (advertisement) của Bluetooth.
- Cân nặng được gửi dưới định dạng little endian.

Do đó, tất cả mọi thiết bị có khả năng kết nối Bluetooth đều có thể nhận dữ liệu từ cân Xiaomi Smart Scale 2 nếu ở trong tầm tín hiệu advertisement của cân.

[48]

Dữ liệu dưới dạng hex chưa qua xử lý sẽ nhìn như sau: 0x62 0xAC 0x49 0xE0 0x07 0x0C 0x14 0x0D 0x1C 0x04 qua mỗi lần thay đổi. Cấu trúc các byte của dữ liệu mà cân gửi khi được giải mã sẽ như sau:

Table 3.4: Cấu trúc các byte dữ liệu cân Xiaomi Smart Scale 2 gửi [48]

byte số	Chức năng
0	byte trạng thái: Bit 0: đơn vị lbs Bit 1-3: không rõ chức năng Bit 5: cân nặng ổn định Bit 6: không rõ Bit 7: cân nặng bỏ khỏi cân
1-2	cân nặng (little endian)
3-4	năm (little endian)
5	tháng
6	ngày

7	giờ
8	phút
9	giây

Qua đó, hệ thống sẽ hoạt động như FSM trên, liên tục nhận thông tin từ cân khi có dữ liệu được quảng bá. Để tìm kiếm và nhận dạng cân Xiaomi chúng ta cần sử dụng, hệ thống sử dụng 2 cách nhận dạng như sau:

- Kết nối thông qua tên thiết bị: MI SCALE2 là tên cân Xiaomi chúng ta cần kết nối. Tuy nhiên, việc kết nối thông qua tên thiết bị có thể sẽ gặp vấn đề nếu như có nhiều thiết bị có cùng tên trong cùng 1 khu vực.
- Kết nối thông qua địa chỉ MAC: địa chỉ MAC độc nhất của kênh hệ thống sử dụng là 5c:64:f3:70:cf:69. Việc kết nối qua địa chỉ MAC, tuy phải biết trước địa chỉ MAC của thiết bị, nhưng vì địa chỉ MAC là độc nhất đối với mỗi thiết bị, việc này sẽ giúp kết nối đến với chính thiết bị (cân) mà chúng ta cần sử dụng.

Để nhận dữ liệu được cân Xiaomi quảng bá, chúng ta cần các số thông số về GATT để thiết lập kết nối và nghe dữ liệu được cân quảng bá:

- Service UUID: 0000181d-0000-1000-8000-00805f9b34fb
- Characteristic UUID: 00002a9d-0000-1000-8000-00805f9b34fb [36]

Sau khi đã thiết lập được kết nối với cân, các byte dữ liệu không cần thiết chúng ta sẽ bỏ qua. Ở đây, chúng ta chỉ cần quan tâm đến byte cân nặng và và bit thứ 5 (bit cân nặng ổn định) của byte 0 (byte trạng thái). Để tính toán và xác định cân nặng, chúng ta có 2 khía cạnh bao gồm:

- **Công thức để tính cân nặng (kg):** $(byte[1] + byte[2] * 256) * 0.005$ [36]
- Đặt một flag để xác nhận cân nặng đã ổn định khi bit 5 của byte 0 trở thành 1 - cân nặng đã ổn định.

2 thông số trên sẽ được xử lý **mỗi khi** có 1 frame được nhận từ cân Xiaomi. Qua đó, theo FSM như trên, hệ thống sẽ lặp đi lặp lại quá trình này liên tục.

Cấu trúc lập trình của kết nối BLE đến cân Xiaomi:

Để sử dụng BLE, wrapper library của quá trình sử dụng BLE sẽ mang tên `scale_ble`

- Các define liên quan đến việc sử dụng BLE bao gồm:
 - `scaleConnectionMode`: Chế độ kết nối đến cân. 0 để kết nối bằng địa chỉ MAC, và 1 để kết nối bằng tên thiết bị.
 - `xiaomiScaleMAC`: Địa chỉ MAC của cân.
 - `xiaomiScaleName`: Tên thiết bị của cân
 - `xiaomiScaleServUUID`: Service UUID của cân
 - `xiaomiScaleCharUUID`: Characteristic UUID của cân

```
1 #define scaleConnectionMode 0
2
3 #define xiaomiScaleMAC      "5c:64:f3:70:cf:69"
4 #define xiaomiScaleName     "MI SCALE2"
5 #define xiaomiScaleServUUID "0000181d
6   -0000-1000-8000-00805f9b34fb"
7 #define xiaomiScaleCharUUID "00002a9d
8   -0000-1000-8000-00805f9b34fb"
```

- function `uint8_t validSendWeight()`: Kiểm tra xem cân nặng có phải là cân nặng ổn định và sẵn sàng để gửi hay không, trả về `true` khi dữ liệu đúng và `false` khi ngược lại.
- function `void resetWeightSendFlag()`: Đã gửi xong dữ liệu đến CPU, reset flag gửi dữ liệu.
- function `uint8_t onScale()`: Kiểm tra xem trên cân có còn cân nặng hay không.

- function float getDefWeight(): Trả về dữ liệu cân nặng ổn định của cân Xiaomi.
- function bool connectToScale(): Kết nối đến cân bằng BLE. Trả về True nếu kết nối thành công, False khi ngược lại, kết nối không thành công. Hàm này sẽ được gọi liên tục trong vòng lặp để kiểm tra và kết nối đến cân nếu kết nối bị ngắt.
- function void initBLE(): Khởi động BLE với các thông số đã thiết lập sẵn.
- function uint8_t scaleConnectStatus(): Kiểm tra trạng thái kết nối hiện tại của cân. True nếu cân vẫn kết nối, False nếu không có kết nối.
- function void turnOffBLE(): Hàm sử dụng để tắt BLE đi.

3.1.1.2.c Kết nối và điều khiển servo HerkuleX DRS-0201. Tối ưu hóa tốc độ xoay của servo.

Kết nối đến servo HerkuleX DRS-0201

Trong hệ thống này, như hệ thống của Ohmnilabs, servo trên được sử dụng để xoay màn hình nhằm điều chỉnh vị trí camera phù hợp cho giải thuật nhận diện khuôn mặt. Với khả năng hỗ trợ 3 giao tiếp UART cùng lúc của ESP32-S3, servo này sẽ sử dụng một cổng HardwareSerial cùng với thư viện [HerkulexServo](#) trên GitHub. Servo sẽ kết nối thông qua một HardwareSerial với baud rate 115200, giao thức UART 8N1 (1 start bit, 8 data bit, không có bit parity và 1 stop bit).

Một số đặc điểm nổi bật trong quá trình hiện thực servo:

- Servo sử dụng nguồn 12V.
- Do giới hạn về mặt vật lý (sự tồn tại của module camera và màn hình) nên góc quay của servo được giới hạn. Giới hạn trên là 690 và giới hạn dưới là 375 (theo góc hoạt động của servo).

- Điểm bắt đầu của servo là 500.
- Do servo và camera hoạt động theo cách scan để đặt gương mặt vào giữa, servo sẽ xoay theo phương thức scan, xoay góc nhỏ liên tục qua mỗi lần lặp của code. Sau quá trình thử nghiệm liên tục, và tìm ra tốc độ quay phù hợp cho servo có đủ momen xoắn để nâng và giữ được màn hình, được định nghĩa ở trong file thư viện `servo_op.h` ở `#define SERVO_SPD` là 40; thấp hơn sẽ khiến cho servo gấp vần đẽ trong quá trình xoay, và người dùng có thể điều chỉnh cao hơn nếu muốn xoay nhanh hơn.
- Mặc dù trong thư viện `HerkulexServo` có một function mang tên `setBrake` (đặt/bật phanh); nhưng do sức nặng của màn hình và module cảm biến/-camera khiến cho servo không thể giữ nguyên vị trí cho dù có sử dụng lệnh `setBrake` (tất cả assembly màn hình và cụm cảm biến sẽ trôi/quay dần dần xuống dưới); do đó, để giữ nguyên vị trí, hệ thống sẽ đặt vị trí stop và cố gắng quay servo đến vị trí dừng sau mỗi vòng lặp để giữ nguyên vị trí.

Cấu trúc lập trình của kết nối BLE đến cân Xiaomi:

Để sử dụng servo, wrapper library sẽ mang tên `servo_op`

- Các define liên quan đến việc sử dụng servo bao gồm:
 - `SERVO_PIN_RX`: Chân Rx kết nối đến ESP32-S3, nối vào chân Tx của Servo
 - `SERVO_PIN_TX`: Chân Tx kết nối đến ESP32-S3, nối vào chân Rx của Servo
 - `SERVO_ID`: Địa chỉ của servo
 - Các mã của chế độ quay của servo, bao gồm quay lên (`SERVO_MOVE_UP`), quay xuống (`SERVO_MOVE_DOWN`), dừng quay (`SERVO_MOVE_STOP`) và quay về vị trí mặc định (`SERVO_MOVE_DEFAULT`).
 - Các define vị trí cố định của servo, bao gồm vị trí khởi đầu (vị trí mặc định) (`SERVO_START_POS`), vị trí quay cao nhất (`SERVO_MAX_UP_POS`) và vị trí quay xuống góc thấp nhất (`SERVO_MAX_DN_POS`).

- SERVO_SPD define tốc độ quay của servo, với 40 là tốc độ tối thiểu (như đã nói trên) để servo có đủ lực momen xoắn để nâng màn hình.

```

1 // Setup pins and IDs
2 #define SERVO_PIN_RX      9
3 #define SERVO_PIN_TX      8
4 #define SERVO_ID          0x03
5
6 // Move mode flags
7 #define SERVO_MOVE_UP      1
8 #define SERVO_MOVE_DOWN    2
9 #define SERVO_MOVE_STOP    3
10 #define SERVO_MOVE_DEFAULT 4
11
12 // Positional defines
13 #define SERVO_START_POS    500
14 #define SERVO_MAX_UP_POS   690
15 #define SERVO_MAX_DN_POS   375
16
17 #define SERVO_SPD        40

```

- function void servo_init(): Mở cổng UART để bắt đầu liên lạc với servo thông qua 2 chân Rx và Tx đã nêu trên.
- function void servo_updateSerial(): Cập nhật thông tin của servo qua serial. Lệnh này sẽ được gọi liên tục, và gọi nhiều nhất có thể.
- function uint16_t servo_getPos(): Trả về vị trí hiện tại của Servo.
- function void servo_setMoveFlag(uint8_t flagnum): Đặt chế độ quay của servo, sử dụng các define servo như đã nói trên.
- function uint8_t servo_isMoving(): Kiểm tra xem servo có đang di chuyển / quay hay không. True nếu servo đang quay, và False nếu servo đang dừng.

3.1.1.2.d Thiết lập giao thức trao đổi thông tin với Đơn vị xử lý trung tâm (CPU)

Để giao tiếp với CPU, IoT Gateway (ESP32-S3) sẽ sử dụng cổng COM (UART), ở baud rate 115200 và giao tiếp thông qua một loạt các tin nhắn được tiêu chuẩn hóa để giao tiếp về mặt dữ liệu và mệnh lệnh với Đơn vị xử lý trung tâm.

Việc tiếp nhận các lệnh được gửi qua UART và xử lý các lệnh đó sẽ được thực hiện ở trong thư viện `system_op`, thông qua hàm `void serial_receive()`. Ngoài ra, trong thư viện `system_op` còn có hàm `uint8_t system_busy()` để kiểm tra xem hệ thống có đang thực hiện một tác vụ khác hay không.

Sau đây là tập hợp các lệnh và giao tiếp được gửi qua cổng COM giữa CPU và IoT Gateway.

Table 3.5: Các lệnh giao tiếp qua cổng COM giữa CPU và IoT Gateway

Lệnh / trả lời	Mô tả
Các lệnh gửi từ CPU	
Nhiệt độ:	
TEMP-GETOBJ	Yêu cầu nhiệt độ của vật thể (object temperature) từ cảm biến nhiệt độ MLX90614
TEMP-GETAMB	Yêu cầu nhiệt độ của môi trường xung quanh (ambient temperature) từ cảm biến nhiệt độ MLX90614
Nhip tim và nồng độ oxy máu:	
POX-GETDATA	Yêu cầu dữ liệu (nhip tim và nồng độ oxy máu) từ cảm biến MAX30102
Cân nặng:	
SCALE-GETWEIGHT	Yêu cầu cân nặng nhận được từ cân Xiaomi Smart Scale 2 thông qua BLE.
Servo:	

SERVO-MOVEUP	Quay servo hướng lên trên
SERVO-MOVEDN	Quay servo hướng xuống dưới
SERVO-STOP	Dừng servo ở vị trí hiện tại
SERVO-MOVEDEFAULT	Quay servo trở về vị trí default (500).
SERVO-GETANGLE	Yêu cầu góc hiện tại của servo.
Các lệnh trả về từ ESP32:	
ACK-[lệnh CPU gửi]	ACK ESP32 gửi trả về để xác nhận mệnh lệnh từ CPU. ACK sẽ được gửi khi nhận bất cứ lệnh nào hợp lệ.
NACK-SYNTAXERROR-[lệnh CPU gửi]	NACK ESP32 gửi trả về do lệnh không đúng cú pháp.
Nhiệt độ:	
TEMP-START	Thông báo bắt đầu kết nối đến cảm biến nhiệt độ MLX90614.
TEMP-STATUS-0	Cảm biến nhiệt độ MLX90614 đang tắt / không kết nối được.
TEMP-STATUS-1	Cảm biến nhiệt độ MLX90614 đang hoạt động bình thường.
OBJ-[nhiệt độ vật thể]	Trả về dữ liệu nhiệt độ vật thể hiện tại, được yêu cầu bởi lệnh TEMP-GETOBJ.
AMB-[nhiệt độ môi trường]	Trả về dữ liệu nhiệt độ môi trường xung quanh hiện tại, được yêu cầu bởi lệnh TEMP-GETAMB.
Nhip tim và nồng độ oxy máu:	
POX-START	Thông báo bắt đầu kết nối đến cảm biến nhịp tim và nồng độ oxy máu MAX30102.
POX-STATUS-0	Cảm biến MAX30102 hiện tại đang tắt / không hoạt động.

POX-STATUS-1	Cảm biến MAX30102 hiện tại đang hoạt động bình thường.
POX-[chỉ số BPM]-[chỉ số SpO2]	Trả về dữ liệu theo yêu cầu, BPM là chỉ số nhịp tim/phút, SpO2 là chỉ số nồng độ oxy trong máu, được yêu cầu bởi lệnh POX-GETDATA.
Cân nặng:	
SCALE-ERROR-NODATA	Thông báo lỗi không có dữ liệu mới nào để gửi trả.
SCALE-ERROR-NOSERVICE	Thông báo lỗi không thể kết nối đến UUID Service của cân.
SCALE-ERROR-NOCHAR	Thông báo lỗi không thể kết nối đến UUID Characteristic của cân.
SCALE-BLE-START	Bắt đầu kết nối BLE đến cân.
SCALE-STATUS-0	Mất kết nối BLE đến cân.
SCALE-STATUS-1	Có kết nối BLE đến cân.
WEIGHT-[chỉ số cân nặng]	Trả về giá trị cân nặng nhận được qua BLE từ cân Xiaomi Smart Scale 2, được yêu cầu từ lệnh SCALE-GETWEIGHT.
Servo:	
SERVO-START	Bắt đầu kết nối đến servo.
SERVO-STATUS-1	Có kết nối thành công đến servo.
SERVO-STATUS-0	Mất kết nối đến servo.
ANGLE-[chỉ số góc hiện tại]	Trả về góc hiện tại của servo, được yêu cầu bởi lệnh SERVO-GETANGLE.

3.1.1.3 Kết hợp các module phụ thành thể thống nhất-hệ thống hoàn chỉnh

Trong quá trình lắp ráp cuối cùng để hoàn chỉnh hệ thống, 2 cảm biến nhiệt độ MLX90614 và cảm biến nhịp tim + nồng độ oxy máu MAX30102 được tích hợp vào hub chứa camera ở phía trên màn hình cảm ứng, với cảm biến nhiệt độ bên phải và cảm biến nhịp tim + nồng độ oxy máu bên trái camera.

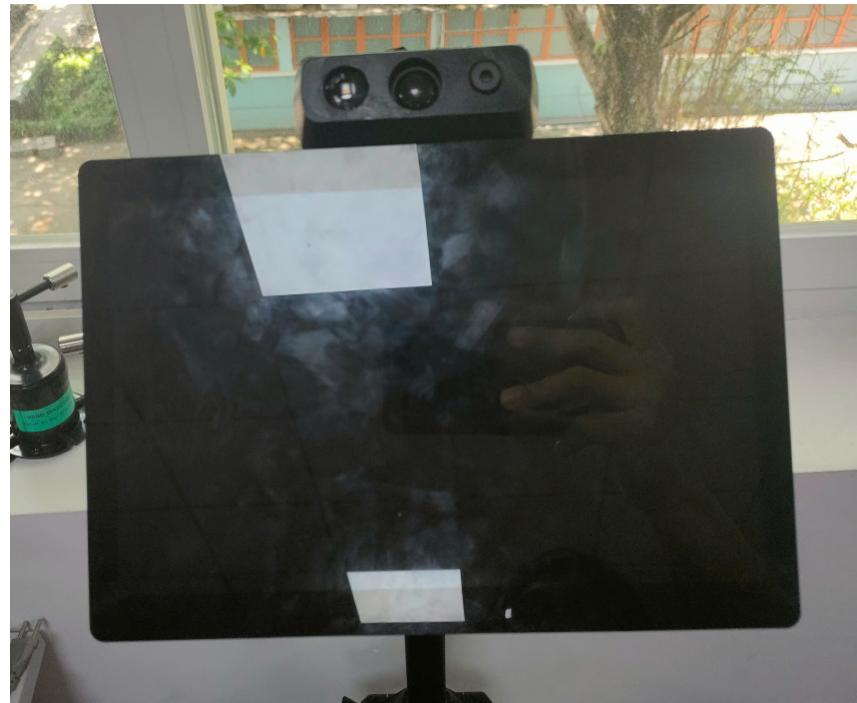


Figure 3.5: Màn hình và hub chứa cảm biến + camera bên trên màn hình.

Do số lượng dây nhiều, nên phần đằng sau màn hình vẫn chưa thể lắp ráp hoàn toàn.

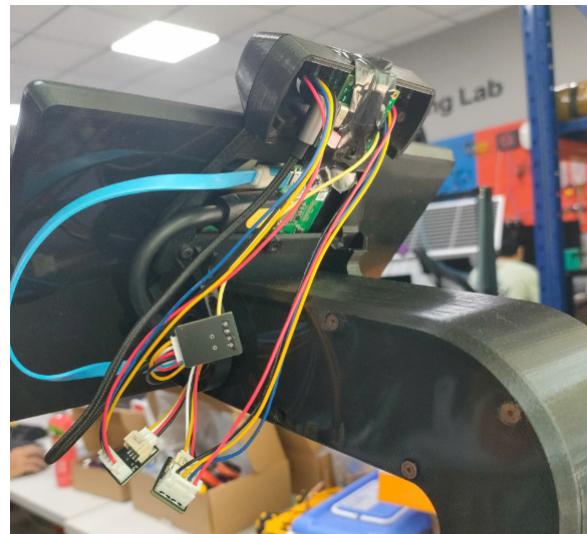


Figure 3.6: Đằng sau hub màn hình và hub chứa cảm biến + camera bên trên màn hình.



Figure 3.7: Hệ thống hoàn thiện, nhìn từ phía trước.

3.2 Hiện thực giao diện màn hình khám sức khỏe

3.2.1 Mục tiêu giao diện

Ứng dụng được thiết kế với giao diện trực quan, giúp người dùng dễ dàng tiếp cận với hệ thống đo lường sức khỏe. Trong quá trình sử dụng, ứng dụng cung cấp các hình ảnh trực quan và âm thanh hướng dẫn, giúp người dùng thực hiện từng bước đo lường một cách chính xác và thuận tiện. Giao diện được tối ưu để hỗ trợ cả người dùng mới lẫn người dùng đã quen thuộc với quy trình khám sức khỏe.

Dưới đây là sơ đồ của giao diện màn hình:

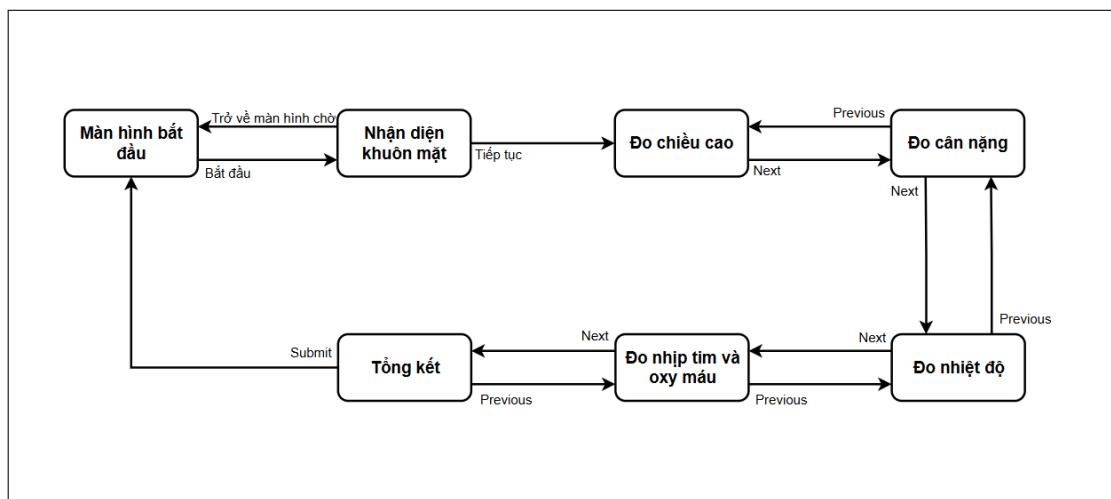


Figure 3.8: Sơ đồ của giao diện màn hình

Hệ thống vận hành theo mô hình **tuần tự với vòng lặp tuyến tính**, nghĩa là các bước trong quá trình khám sức khỏe được thực hiện theo trình tự cố định. Sau khi hoàn thành một chu trình đo lường, dữ liệu sẽ được xử lý và lưu trữ, sau đó hệ thống quay về trạng thái ban đầu để chuẩn bị cho lần đo tiếp theo. Điều này đảm bảo rằng ứng dụng hoạt động ổn định, chính xác và có khả năng lặp lại một cách hiệu quả.

3.2.2 Module chính trong ứng dụng

Để vận hành giao diện một cách hiệu quả sẽ gồm có các module phối hợp chặt chẽ để thực hiện nhiệm vụ đo lường, xử lý và lưu trữ dữ liệu. Các module chính bao gồm:

1. Router:

- Sử dụng thư viện BrowserRouter của ReactJS để điều hướng giữa các màn hình.[18]
- Kiểm tra điều kiện cần thiết trước khi chuyển sang bước kế tiếp.

2. Send-SerialPort:

- Sử dụng thư viện serialport của ReactJS để gửi lệnh điều khiển phần cứng ESP32.[19]
- Đảm bảo ESP32 thực hiện đúng nhiệm vụ theo yêu cầu từ hệ thống.

3. Listen-SerialPort:

- Cũng sử dụng thư viện serialport của ReactJS nhưng với chức năng lắng nghe dữ liệu từ ESP32.[19]
- Khi dữ liệu đo lường được gửi về, hệ thống thực hiện các bước tính toán trước khi hiển thị lên màn hình.

4. Nhận diện khuôn mặt:

- Sử dụng thư viện TensorFlow và các thư viện hỗ trợ khác để xử lý nhận diện.
- Điều chỉnh vị trí khuôn mặt sao cho luôn nằm giữa camera, hỗ trợ việc tính toán chiều cao và điều chỉnh màn hình hiển thị sao cho phù hợp với người dùng.

5. SendFirebase:

- Sử dụng API của Firebase để gửi dữ liệu đã đo lường được lên hệ thống lưu trữ. [20]
- Đảm bảo dữ liệu có thể được lưu trữ, phân tích và truy xuất khi cần thiết.

3.2.3 Giao diện tương tác người dùng

3.2.3.1 Màn hình chờ

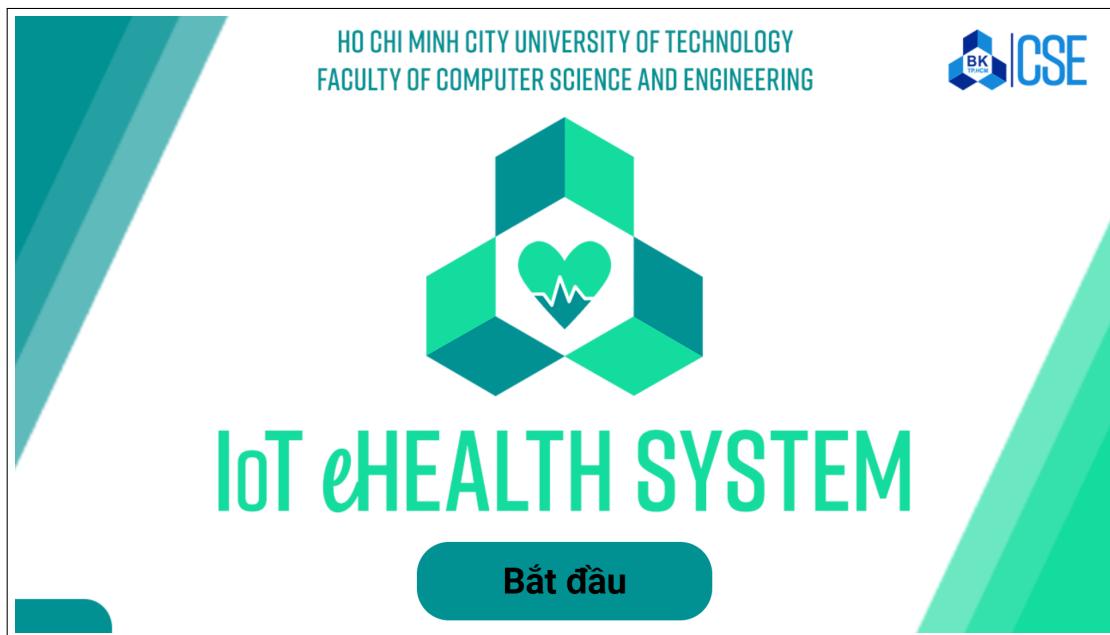


Figure 3.9: Màn hình chờ

Đây là màn hình chờ khi người dùng tương tác với trạm khám sức khỏe. Tại đây hiển thị nhiều thông tin và logo cũng một nút bấm "Bắt đầu" để bắt đầu đo đạc. Cùng lúc đó khi nhấn nút "Bắt đầu" camera sẽ xoay tới vị trí mặc định ban đầu phù hợp đa số người dùng.

3.2.3.2 Màn hình nhận diện khuôn mặt

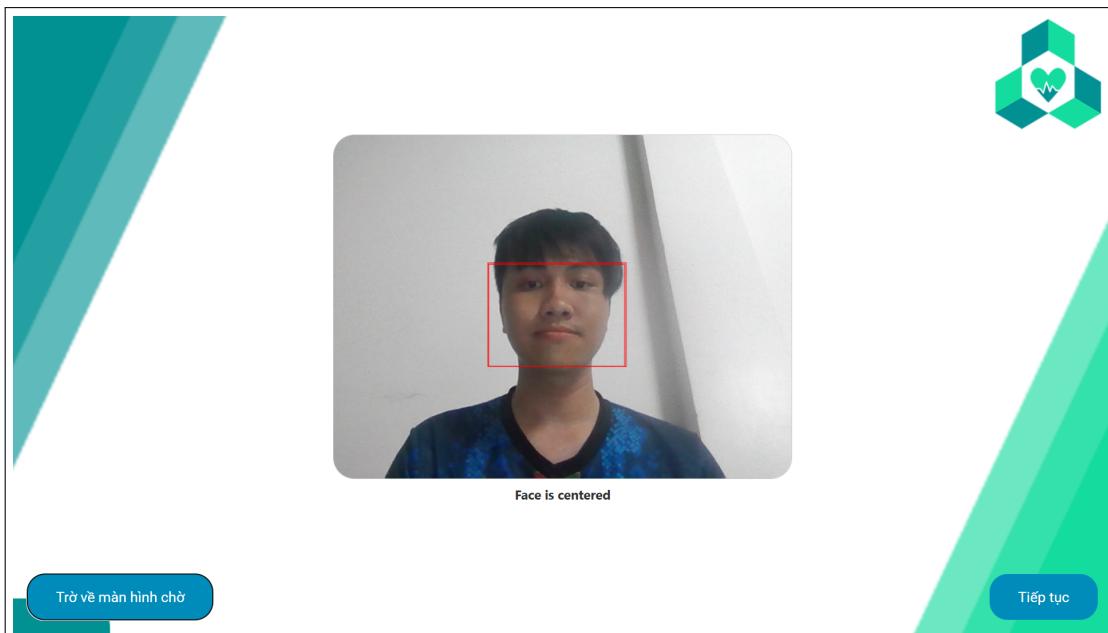


Figure 3.10: Người dùng khi ở vị trí hoàn hảo

Đây là màn hình nhận diện khuôn mặt. Trong hình khi người dùng ở vị trí trung tâm của camera thì nút bấm "Tiếp tục" sẽ hiển thị. Từ đó người dùng sẽ bắt đầu đo đạc các chỉ số sức khỏe

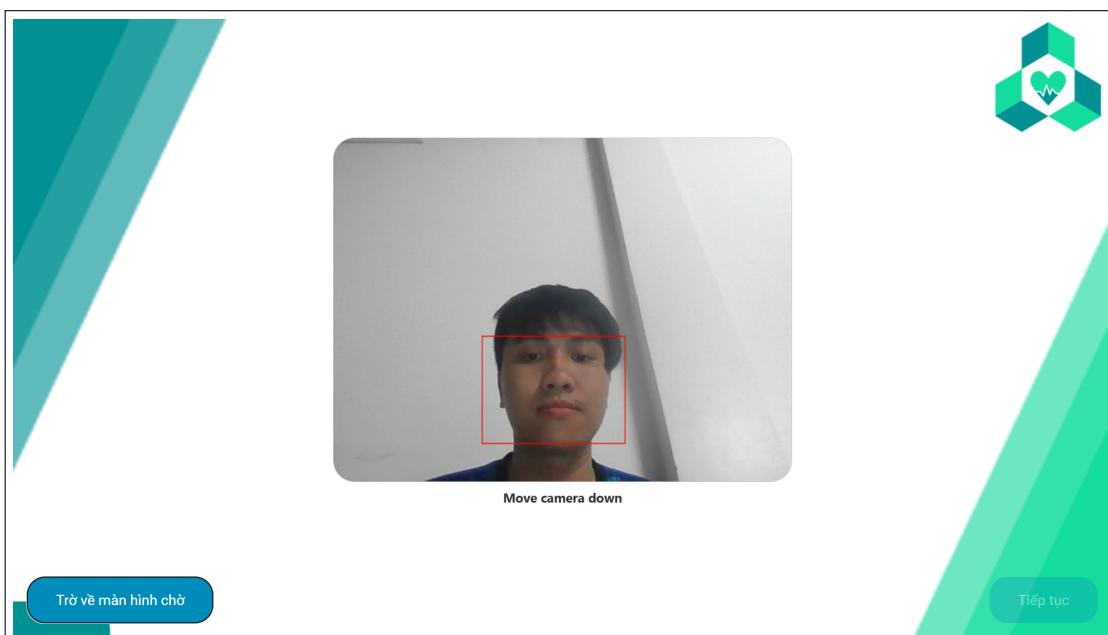


Figure 3.11: Người dùng đang thấp hơn

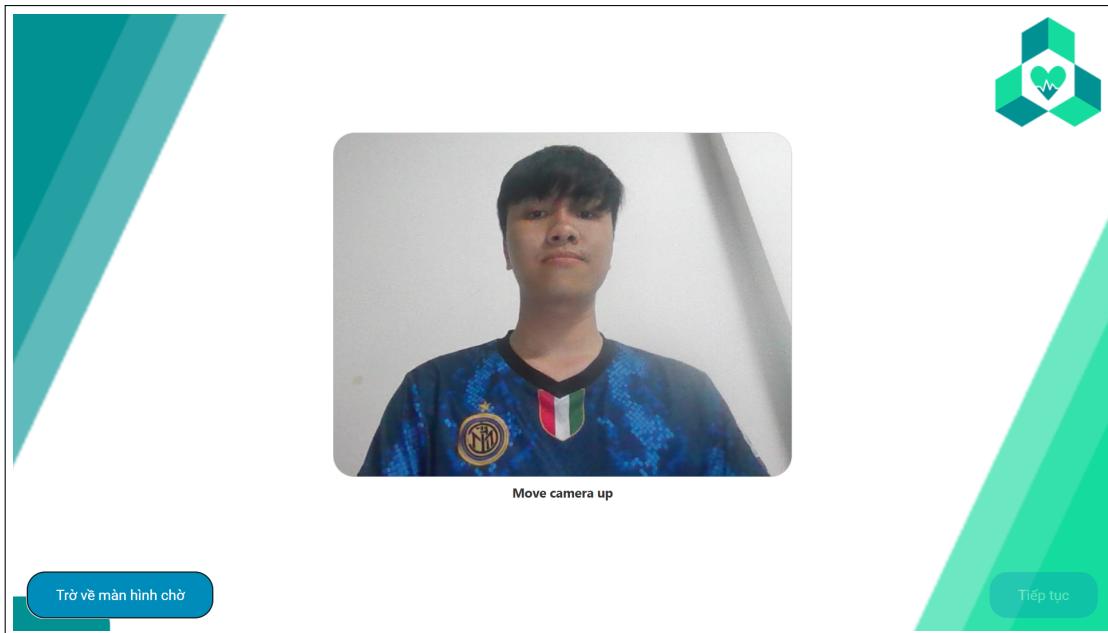


Figure 3.12: Người dùng đang cao hơn

Còn đây là hai trường hợp khi cao hơn hoặc thấp hơn vị trí trung tâm của camera. Nút bấm "Tiếp tục" sẽ bị ẩn đi cũng lúc đó camera sẽ di chuyển dần đi lên hoặc dần đi xuống để nhận diện khuôn mặt người dùng. Ngoài ra, có nút bấm "Trở về màn hình chờ" có tác dụng đưa camera trở lại vị trí ban đầu khi người dùng có những sự cố ngoài ý muốn xảy ra.

3.2.3.3 Màn hình đo chỉ số



Figure 3.13: Màn hình đo chiều cao

Tại màn hình này sẽ có âm thanh tự động hướng dẫn được phát ra loa, nếu muốn nghe lại người dùng có thể nhấn icon sound trên màn hình. Khi đã nắm rõ hướng dẫn, người dùng nhấn nút "Bắt đầu". Sau 1 đến 2 giây, chiều cao sẽ được hiển thị trên màn hình.

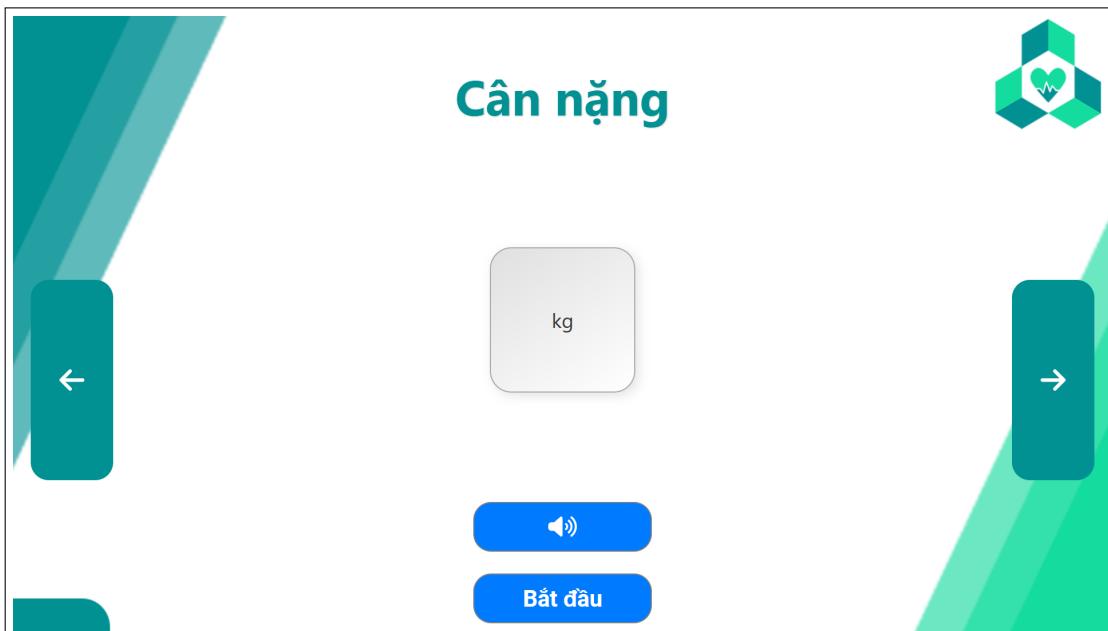


Figure 3.14: Màn hình đo cân nặng

Phần âm thanh hướng dẫn cũng sẽ tự động bật và có thể nhấn icon sound để nghe lại giống bên trang đo chiều cao. Khi người dùng đã hiểu hướng dẫn, người dùng nhấn nút "Bắt đầu". Sau 1 đến 2 giây, cân nặng sẽ được hiển thị trên màn hình.

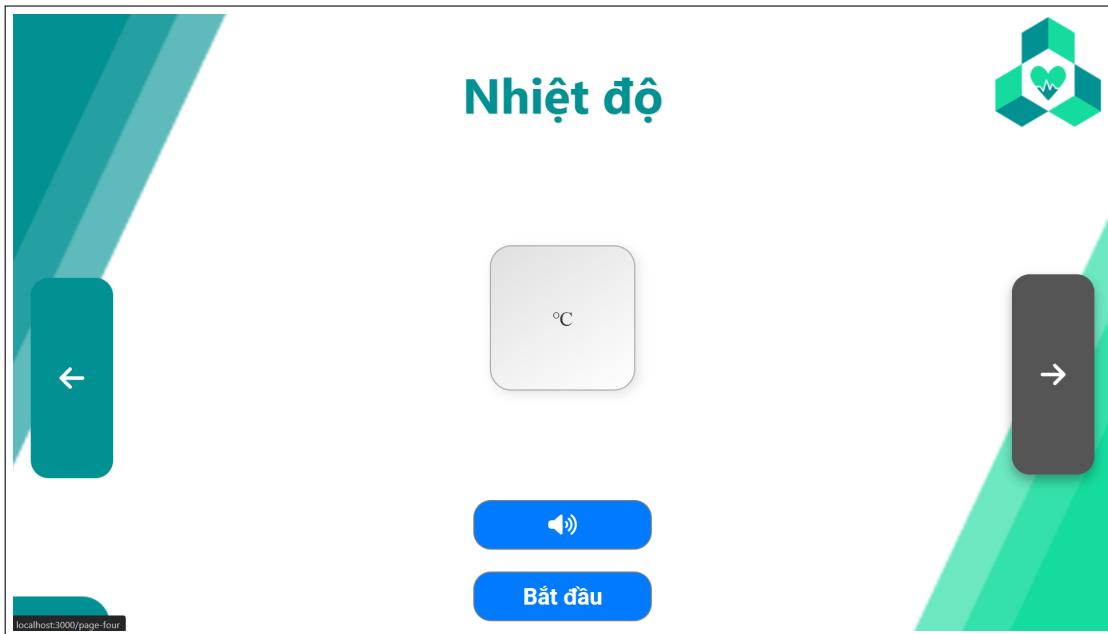


Figure 3.15: Màn hình đo nhiệt độ cơ thể

Khi người dùng đã hiểu hướng dẫn, người dùng nhấn nút "Bắt đầu". Sau 10 đến 15 giây, nhiệt độ người dùng sẽ được hiển thị trên màn hình.

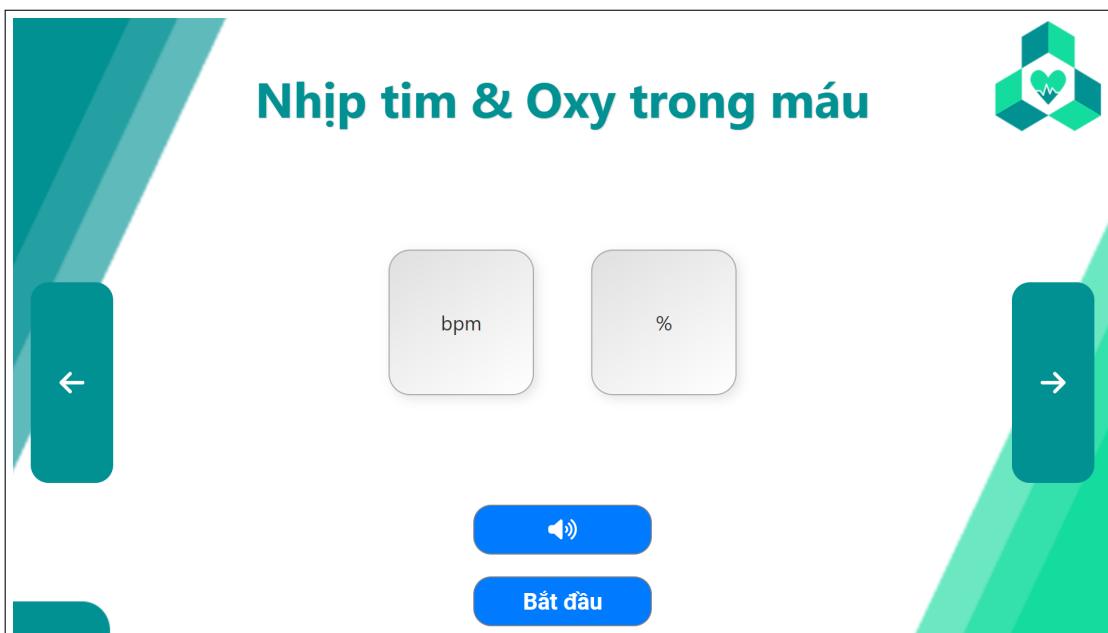


Figure 3.16: Màn hình đo nhịp tim và oxy trong máu

Khi người dùng đã hiểu hướng dẫn, người dùng nhấn nút "Bắt đầu". Sau 30 đến 45 giây, đồng thời nhịp tim và oxy trong máu sẽ được hiển thị trên màn hình.

3.3 Ứng dụng theo dõi trên điện thoại

Ứng dụng này được tạo ra nhằm hỗ trợ bác sĩ trong việc theo dõi và thống kê dữ liệu sức khỏe bệnh nhân, bao gồm 4 chức năng chính:

- I. Đăng kí/Đăng nhập.
- II. Thực hiện đo chỉ số sức khỏe.
- III. Xem lịch sử đo.
- IV. Điều khiển trạng thái thiết bị.
- V. Thay đổi thông tin cá nhân.

3.3.1 Đăng kí/Đăng nhập

1. Khi bắt đầu, ứng dụng sẽ chuyển tới màn hình yêu cầu người dùng đăng nhập (hoặc đăng kí nếu chưa có tài khoản).

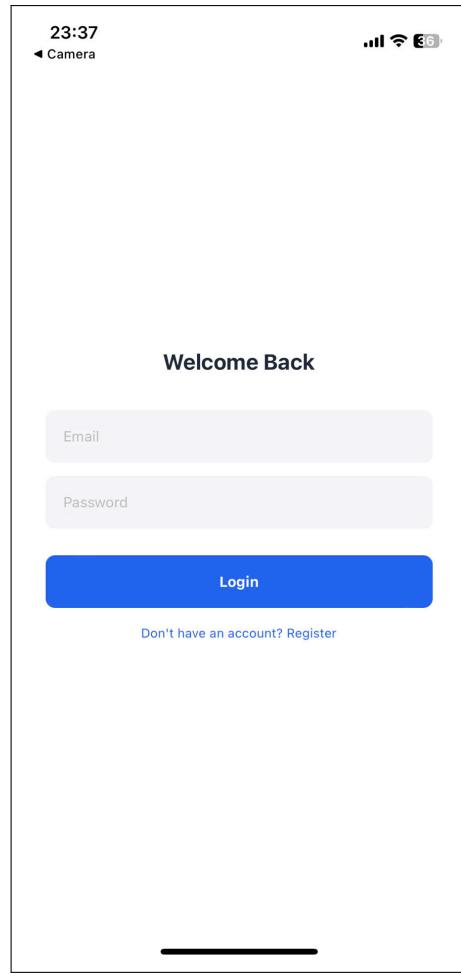


Figure 3.17: Giao diện Login

2. Nếu người dùng đã có tài khoản, nhập thông tin và chọn "Login", trong trường hợp người dùng chưa có tài khoản, chọn "Register" để đăng kí tài khoản mới.
3. Ở đây người dùng được yêu cầu nhập gmail và mật khẩu để đăng kí.

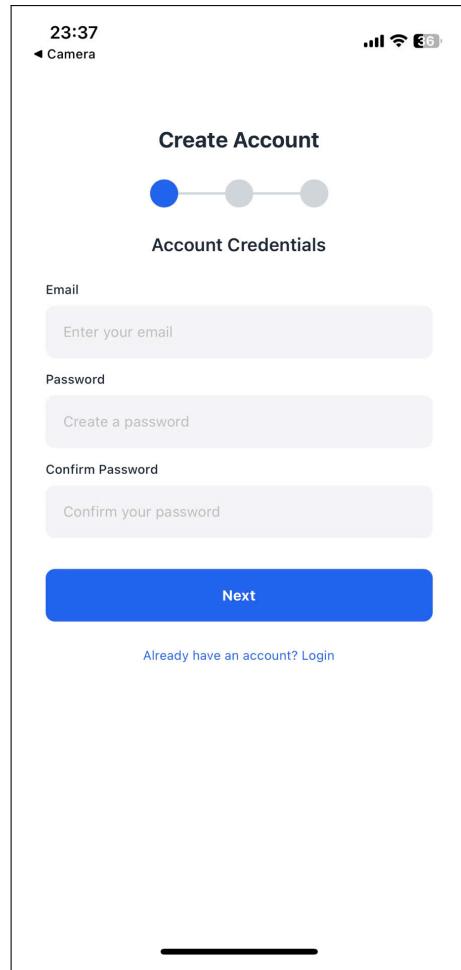


Figure 3.18: Đăng ký bước 1

4. Sau đó người dùng được yêu cầu nhập họ và tên.

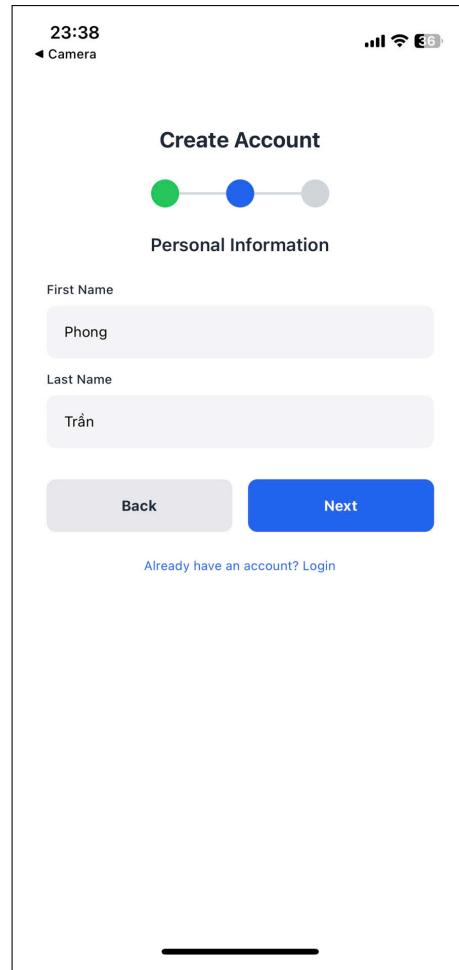


Figure 3.19: Đăng ký bước 2

5. Tiếp theo người dùng được yêu cầu nhập dữ liệu liên quan tới ngành nghề bác sĩ

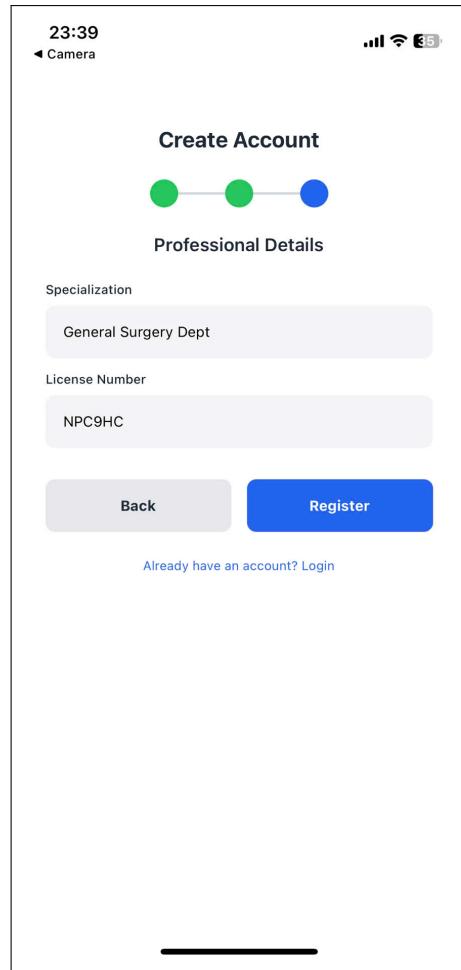


Figure 3.20: Đăng kí bước 3

6. Cuối cùng, sau khi đăng kí (đăng nhập), ứng dụng sẽ xuất hiện giao diện dưới đây, từ bây giờ người dùng có thể sử dụng toàn bộ tính năng của app.

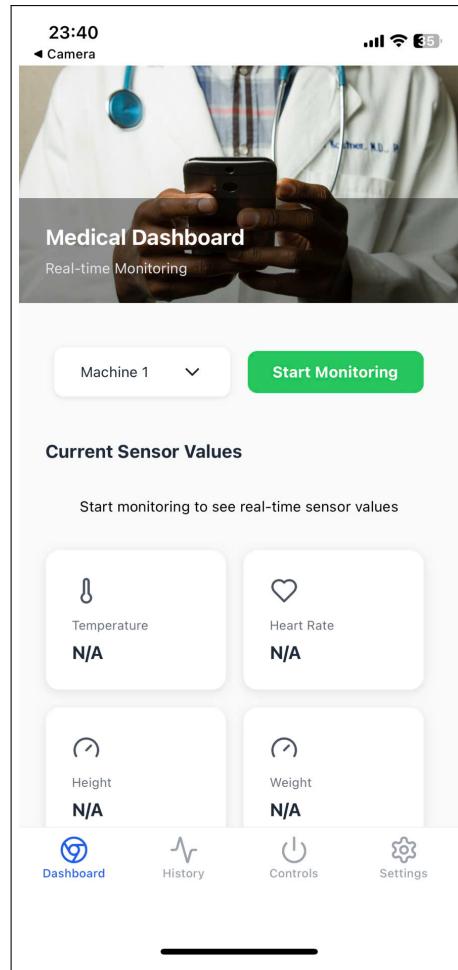


Figure 3.21: Đăng ký thành công

3.3.2 Thực hiện đo chỉ số sức khỏe

1. Trong giao diện Dashboard, người dùng có thể kết nối ứng dụng tới máy đo tùy chọn để thực hiện đo.

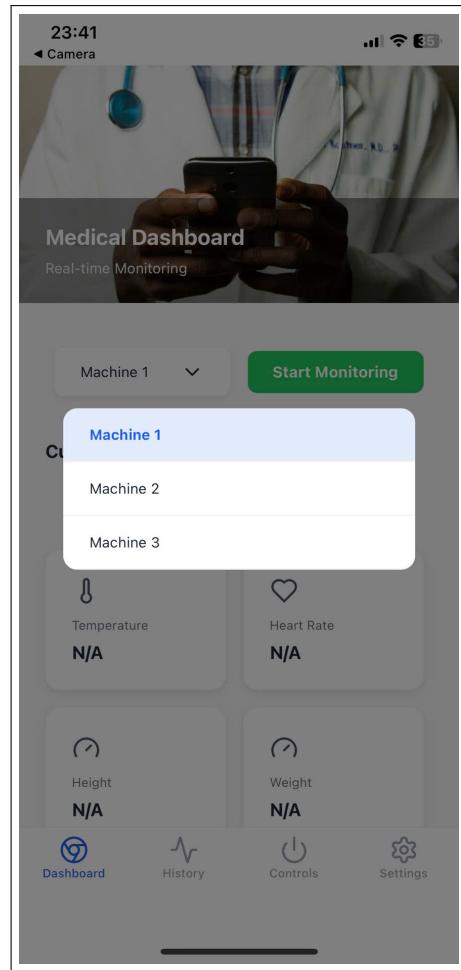


Figure 3.22: Giao diện DashBoard

2. Sau đó, người dùng chọn "Start Monitoring" để có thể bắt đầu đo.

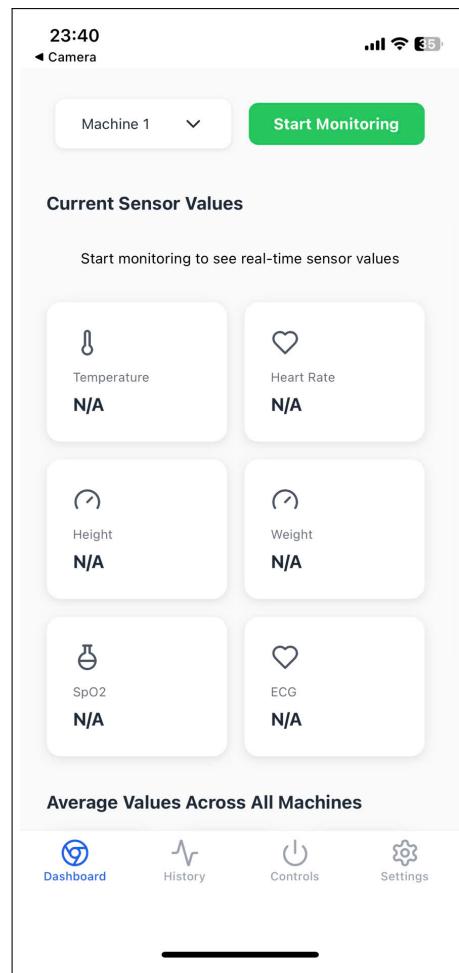


Figure 3.23: Giao diện Dashboard khi đo

3. Ứng dụng sẽ hiển thị giá trị từ các sensor trong quá trình đo.

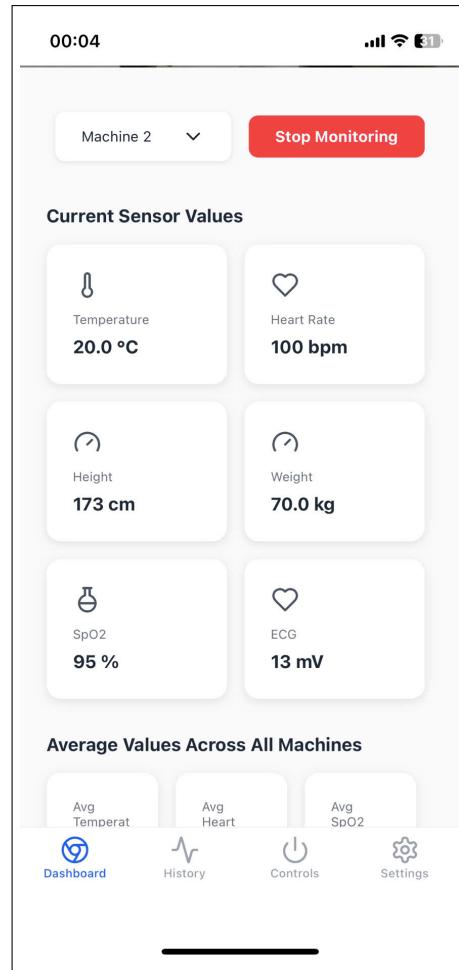


Figure 3.24: Giá trị sensor khi đo

4. Sau khi thực hiện đo xong, người dùng chọn "Stop Monitoring" để kết thúc lần đo. Khi đó lịch sử đo sẽ được cập nhật và màn hình sẽ hiển thị kết quả trung bình của các sensor.

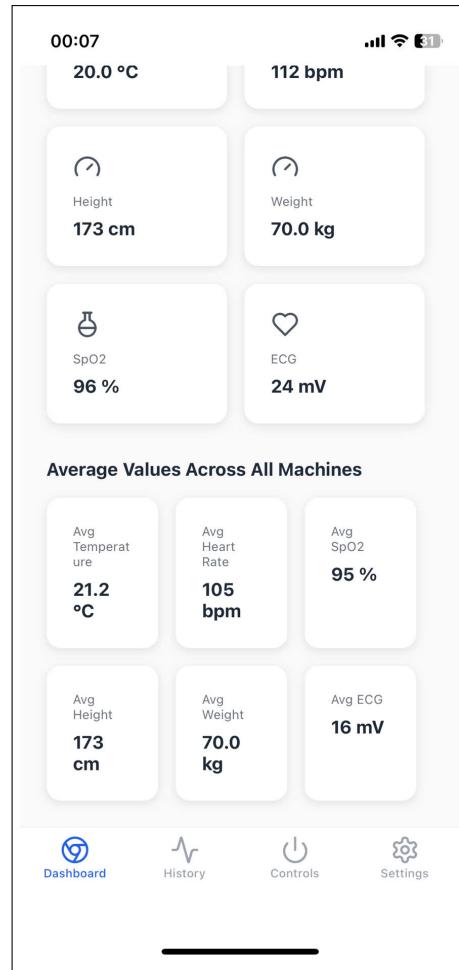


Figure 3.25: Giá trị trung bình

3.3.3 Xem lịch sử đo

- Để có thể xem lịch sử đo, ở giao diện chính người dùng chọn biểu tượng "History" ở menu bar, lúc này màn hình xuất hiện giao diện dưới đây.

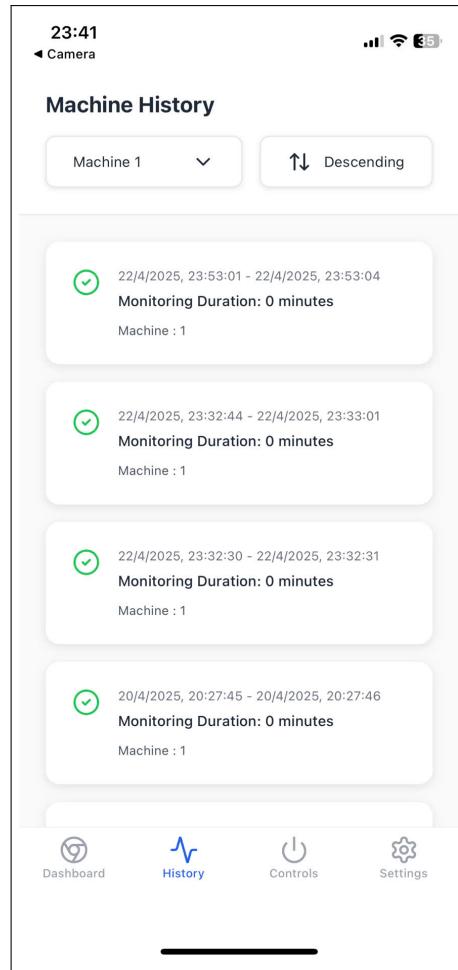


Figure 3.26: Giao diện lịch sử đo

2. Ở đây, người dùng có thể tùy chọn máy đo để xem lịch sử đo của máy đó.
Lúc này giao diện sẽ hiển thị toàn bộ lịch sử đo của một máy đo được chọn.

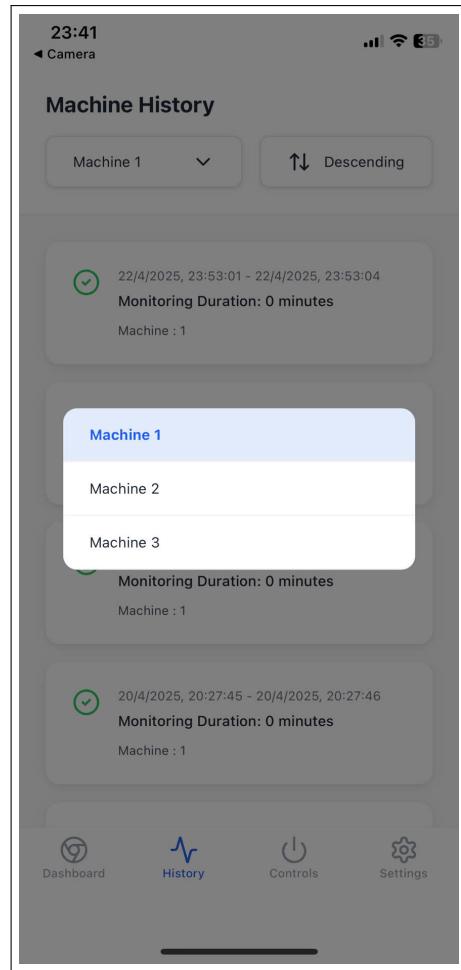


Figure 3.27: Tùy chọn máy đo

3. Để có thể xem chi tiết người dùng chọn vào lần đo muốn xem, lúc này giao diện sẽ hiển thị thông tin chi tiết của lần đo được chọn.



Figure 3.28: Lịch sử đo chi tiết

4. Người dùng kéo xuống để có thể xem toàn bộ giá trị của lần đo được chọn.

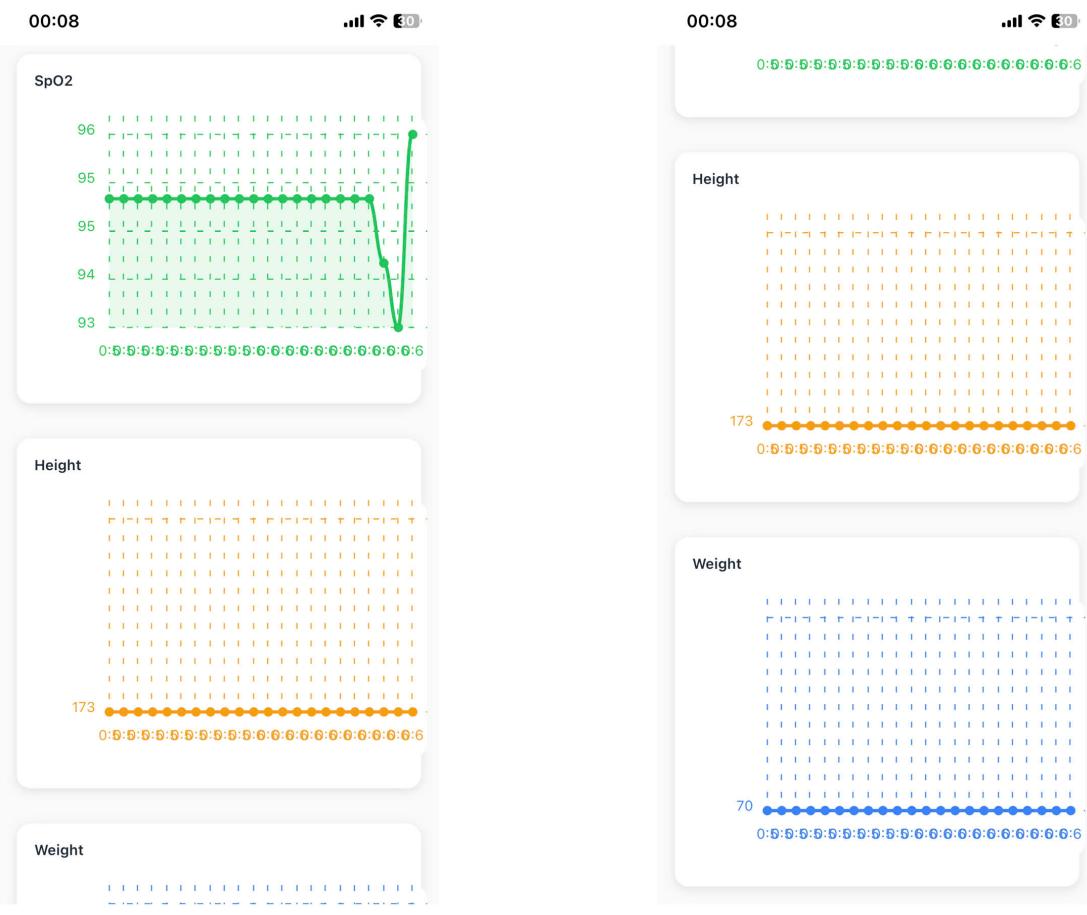


Figure 3.29: Lịch sử đo chi tiết (tiếp tục)

3.3.4 Điều khiển trạng thái thiết bị

- Để có thể điều khiển trạng thái (bật/tắt) của sensor, người dùng chọn "Controls" ở Menu Bar.

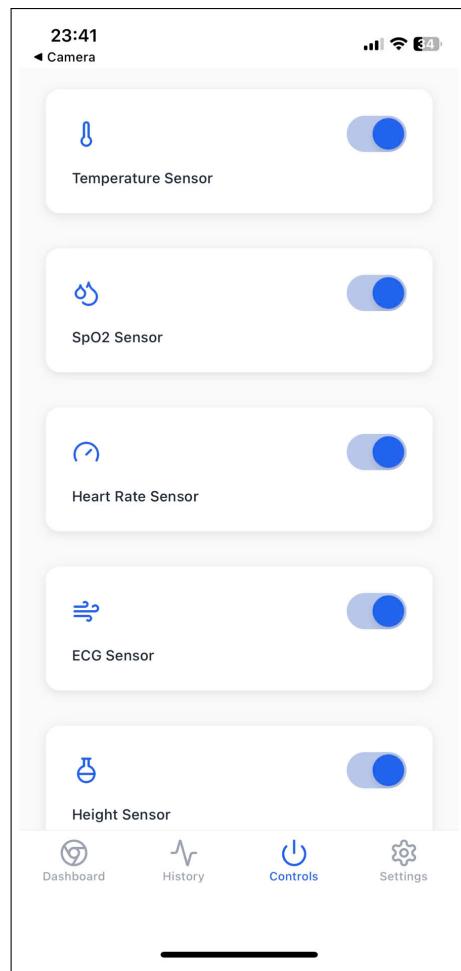


Figure 3.30: Giao diện điều khiển

- Tại đây, người dùng có thể tùy chọn máy đo muốn điều khiển trạng thái sensor.

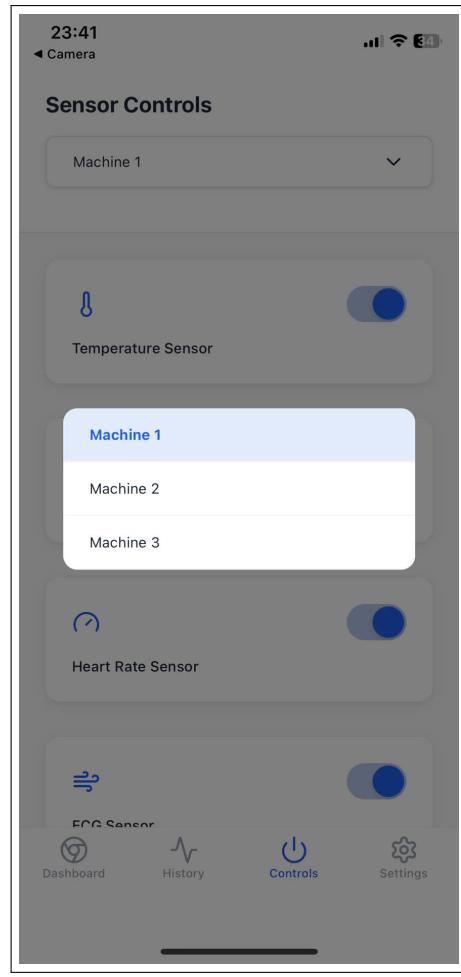


Figure 3.31: Tùy chọn máy đo

3. Bây giờ người dùng có thể bật/tắt sensor bằng cách nhấn button của sensor đó.

3.3.5 Thay đổi thông tin cá nhân

1. Để thay đổi thông tin cá nhân, người dùng chọn phần "Setting" ở Menu Bar, màn hình xuất hiện giao diện dưới đây.

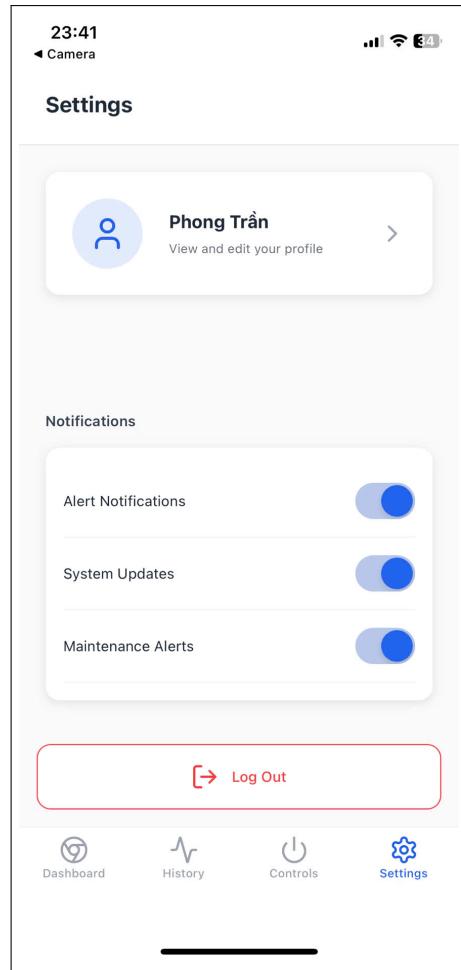


Figure 3.32: Giao diện cài đặt

2. Ở đây người dùng có thể bật tắt cảnh báo, thông báo update, bảo trì,... Để có thể tiến hành thay đổi thông tin, chọn "View and edit your profile", màn hình xuất hiện giao diện dưới đây.

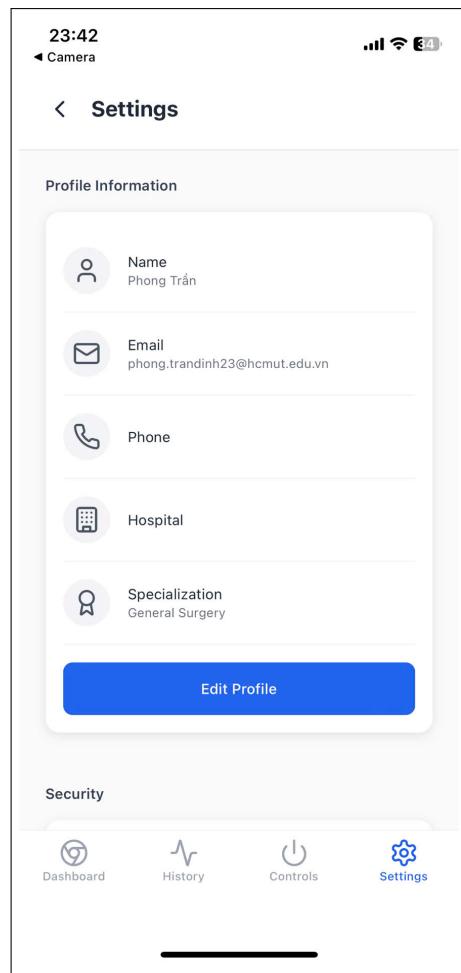


Figure 3.33: Thông tin cá nhân

3. Tiếp theo người dùng chọn "Edit Profile", thay đổi thông tin theo ý muốn và chọn "Save Changes" để lưu lại những thay đổi.

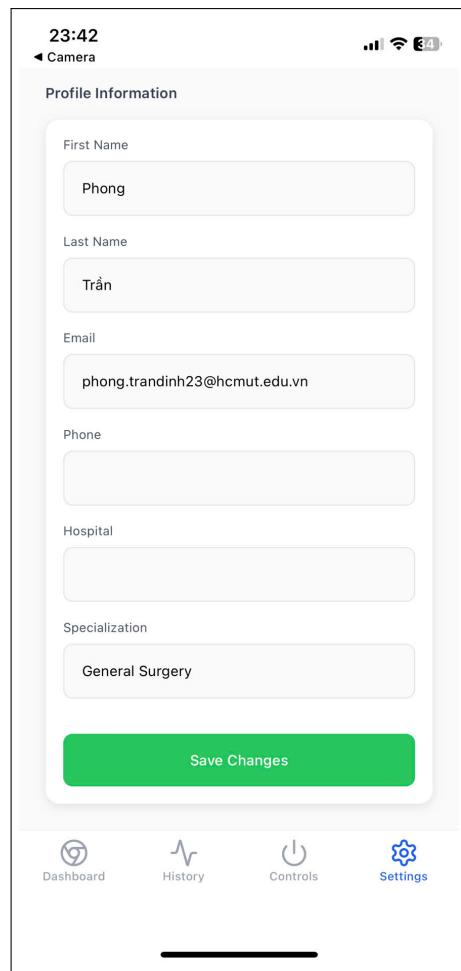


Figure 3.34: Thay đổi thông tin cá nhân

Chương 4

Thực nghiệm

4.1 Kiểm thử phần cứng

Kiểm thử phần cứng sẽ được thực hiện bằng cách theo dõi phản hồi của ESP32-S3 thông qua Serial.

- Test khởi động hệ thống và kết nối đến các thiết bị đầu cuối IoT.

```
POX-START  
POX-STATUS-1  
TEMP-START  
TEMP-STATUS-1  
TEMP-STATUS-1  
SERVO-START  
SERVO-STATUS-1  
SCALE-BLE-START  
SCALE-STATUS-1
```

Figure 4.1: Hệ thống khi khởi động với các lệnh STATUS trả về.

```
NACK-SYNTAXERROR-ADF  
NACK-SYNTAXERROR-ET  
NACK-SYNTAXERROR-N  
NACK-SYNTAXERROR-A  
NACK-SYNTAXERROR-ADF  
NACK-SYNTAXERROR-NA
```

Figure 4.2: Lệnh NACK được trả về khi lệnh nhập sai.

- Test xoay servo lên, xuống và về điểm default, cũng như gọi lệnh trả về góc hiện tại.

```
ACK-SERVO-MOVEUP
ACK-SERVO-MOVEDN
SCALE-STATUS-1
ACK-SERVO-MOVEUP
ACK-SERVO-MOVEDN
ACK-SERVO-MOVEDEFAULT
```

Figure 4.3: các lệnh xoay servo được nhận với ACK trả về.

```
ACK-SERVO-GETANGLE
ANGLE-698
ANGLE-698
ANGLE-698
ANGLE-698
ANGLE-698
SCALE-STATUS-0
SCALE-STATUS-1
ACK-SERVO-MOVEDFAULT
ACK-SERVO-MOVEUP
ACK-SERVO-GETANGLE
ANGLE-700
ANGLE-700
ANGLE-700
ANGLE-700
ANGLE-700
```

Figure 4.4: các lệnh xoay servo được nhận với ACK trả về cùng với góc được trả về - góc xoay tối đa.

```
ACK-SERVO-MOVEDN
ACK-SERVO-GETANGLE
ANGLE-365
ANGLE-365
ANGLE-365
ANGLE-365
ANGLE-365
```

Figure 4.5: Góc xoay tối thiểu của servo.

```
ACK-SERVO-GETANGLE  
ANGLE-536  
ANGLE-536  
ANGLE-536  
ANGLE-536  
ANGLE-536
```

Figure 4.6: Servo dừng ở một góc ngẫu nhiên.

- Test lấy nhiệt độ từ cảm biến MLX90614.

```
ACK-TEMP-GETAMB  
AMB-26.49  
AMB-26.49  
AMB-26.49  
AMB-26.51  
AMB-26.49
```

Figure 4.7: ACK của lệnh lấy nhiệt độ môi trường xung quanh.

```
ACK-TEMP-GETOBJ  
OBJ-26.53  
OBJ-26.29  
OBJ-26.35  
OBJ-26.49  
OBJ-26.71
```

Figure 4.8: ACK của lệnh lấy nhiệt độ vật thể, nhưng không có vật thể trước cảm biến.

```
ACK-TEMP-GETOBJ  
OBJ-33.19  
OBJ-33.87  
OBJ-34.33  
OBJ-34.29  
OBJ-33.73
```

Figure 4.9: ACK của lệnh lấy nhiệt độ vật thể, có vật thể trước cảm biến (bàn tay người).

- Test lấy dữ liệu nồng độ oxy máu và nhịp tim từ cảm biến MAX30102.

```
ACK-POX-GETDATA
SCALE-STATUS-0
POX-115-99.36
POX-115-99.59
POX-125-98.08
POX-125-99.15
POX-125-97.61
SCALE-STATUS-1
SCALE-STATUS-0
SCALE-STATUS-1
NACK-SYNTAXERROR-
ACK-POX-GETDATA
POX-107-99.62
POX-115-99.21
POX-107-99.45
SCALE-STATUS-0
POX-100-98.93
POX-93-99.42
```

Figure 4.10: ACK của lệnh lấy dữ liệu từ cảm biến MAX30102, với dữ liệu được trả về, số đầu tiên là số nhịp tim/phút, số thứ 2 là chỉ số nồng độ oxy máu theo %.

- Test lấy dữ liệu từ cân Xiaomi.

```
ACK-SCALE-GETWEIGHT
WEIGHT-76.50
WEIGHT-76.50
WEIGHT-76.50
WEIGHT-76.50
WEIGHT-76.50
```

Figure 4.11: ACK của lệnh lấy dữ liệu từ cân, với cân nặng theo kg được trả về.

4.2 Kiểm thử Browser của hệ thống

Bài kiểm thử hệ thống

Mục tiêu: Kiểm tra các chức năng đo lường và phản hồi dữ liệu của hệ thống, bao gồm đo chiều cao, cân nặng, nhiệt độ, nhịp tim và oxy máu. Đồng thời kiểm tra khả năng gửi dữ liệu lên Firebase với tốc độ mạng ổn định.

Các bước thực hiện:

1. Kiểm tra tín hiệu UART:

- Gửi tín hiệu từ nguồn đến ESP32.
 - Đảm bảo thời gian phản hồi tối đa 0,5 giây.
2. Kiểm tra đo lường chiều cao và cân nặng:
- Đo chiều cao với thời gian phản hồi từ 1 đến 2 giây.
 - Đo cân nặng với thời gian phản hồi từ 1 đến 2 giây.
3. Kiểm tra đo lường nhiệt độ:
- Kích hoạt cảm biến nhiệt độ.
 - Đảm bảo thời gian phản hồi từ 10 đến 15 giây.
4. Kiểm tra đo nhịp tim và oxy máu:
- Kích hoạt cảm biến nhịp tim và oxy máu.
 - Đảm bảo thời gian phản hồi từ 30 đến 45 giây.

5. Kiểm tra gửi dữ liệu lên Firebase:
- Gửi dữ liệu lên Firebase sau khi thu thập đầy đủ giá trị.
 - Đảm bảo thời gian gửi từ 1 đến 2 giây với tốc độ mạng ổn định.

Kết quả mong đợi:

- Hệ thống phản hồi theo thời gian được chỉ định cho từng chức năng.
- Không có lỗi hoặc sai sót trong quá trình truyền và xử lý dữ liệu.

4.3 Kiểm thử chức năng Ứng dụng theo dõi

Bài kiểm thử hệ thống

Mục tiêu: Kiểm tra các chức năng chính của ứng dụng di động, bao gồm lấy dữ liệu đo lường theo thời gian thực, hiển thị dữ liệu đúng định dạng, cập nhật lịch sử đo theo thời gian, và đảm bảo độ ổn định khi truyền nhận dữ liệu.

Các bước thực hiện:

1. Kiểm tra đăng nhập người dùng:

- Đảm bảo tài khoản hợp lệ có thể đăng nhập được trên hệ thống.
- Đảm bảo thời gian phản hồi tối đa là 3 giây và điều hướng tới Dashboard sau khi đăng nhập.

2. Kiểm tra lấy dữ liệu từ Firebase:

- Đảm bảo ứng dụng đã kết nối và sẵn sàng nhận dữ liệu từ Firebase.
- Đảm bảo độ trễ hiển thị dữ liệu mới không quá 3 giây so với thời điểm dữ liệu được cập nhật trên Firebase.

3. Kiểm tra hiển thị dữ liệu đo:

- Đảm bảo giao diện hiển thị đầy đủ các chỉ số đo bao gồm: Chiều cao, cân nặng, nhiệt độ, nhịp tim, oxy máu.
- Đảm bảo dữ liệu hiển thị chính xác theo dữ liệu được lưu trữ trên Firebase.

4. Kiểm tra việc cập nhật lịch sử đo:

- Đảm bảo lịch sử đo được cập nhật ngay khi lần đo kết thúc.
- Đảm bảo dữ liệu lịch sử đo chi tiết không bị sai lệch, biểu đồ không bị giật hay đứng hình.

5. Kiểm tra tốc độ phản hồi khi mạng ổn định:

- Kiểm thử khả năng nhận và cập nhật dữ liệu trong các điều kiện mạng yếu và ổn định.
- Đảm bảo dữ liệu luôn được đồng bộ liên tục mà không bị mất kết nối hoặc lỗi dữ liệu.

Kết quả mong đợi:

- Ứng dụng đăng nhập thành công và lấy dữ liệu đúng với thông tin trên firebase.
- Dữ liệu đo lường cập nhật gần như đồng thời với độ trễ nhỏ.
- Lịch sử đo cập nhật đúng giá trị lần đo, biểu đồ hiển thị đúng và trơn tru.
- Không có lỗi hoặc sai lệch trong quá trình chuyển nhận dữ liệu giữa Firebase và ứng dụng.

Kết quả thực tế:

1. Ứng dụng cho phép tài khoản hợp lệ đăng nhập, sau khi đăng nhập ứng dụng hiển thị giao diện dashboard.

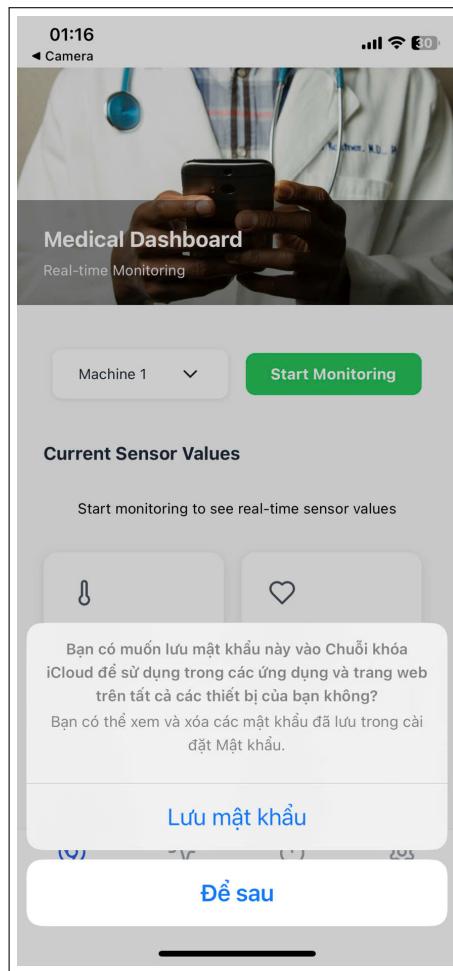


Figure 4.12: Giao diện ứng dụng sau khi đăng nhập

2. Dữ liệu cập nhật gần như đồng thời với Firebase trong điều kiện mạng ổn

định. Trong điều kiện mạng yếu, dữ liệu trên ứng dụng cập nhật trễ từ 3 đến 4 giây sau khi dữ liệu trên Firebase được cập nhật.

3. Giao diện hiển thị đầy đủ các chỉ số đo hiện có, và đúng với dữ liệu được lưu trữ trên Firebase.



```
https://platformio-demo-default.firebaseioapp.com/machine1.json
```

```
{
  "machine1": {
    "id": "1",
    "ecg": "26",
    "heart_rate": "115",
    "height": "173",
    "spo2": "96",
    "temp": "29",
    "weight": "65"
  }
}
```

Figure 4.13: Dữ liệu trên Firebase

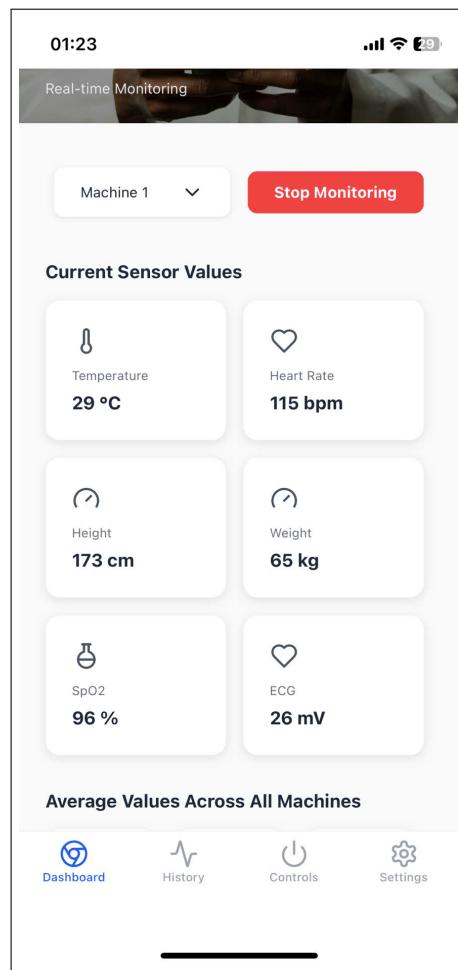


Figure 4.14: Dữ liệu trên Ứng dụng

4. Lịch sử đo được cập nhật ngay sau khi kết thúc lần đo, đúng với giá trị trung bình của lần đo đó.

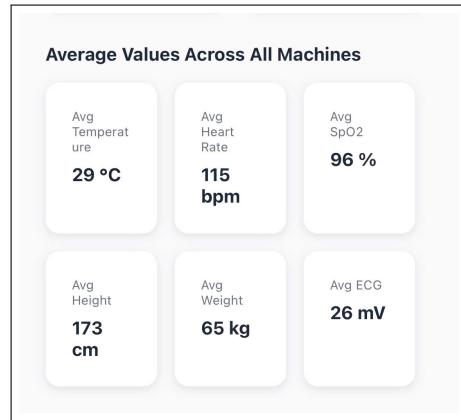


Figure 4.15: Giá trị trung bình của lần đo

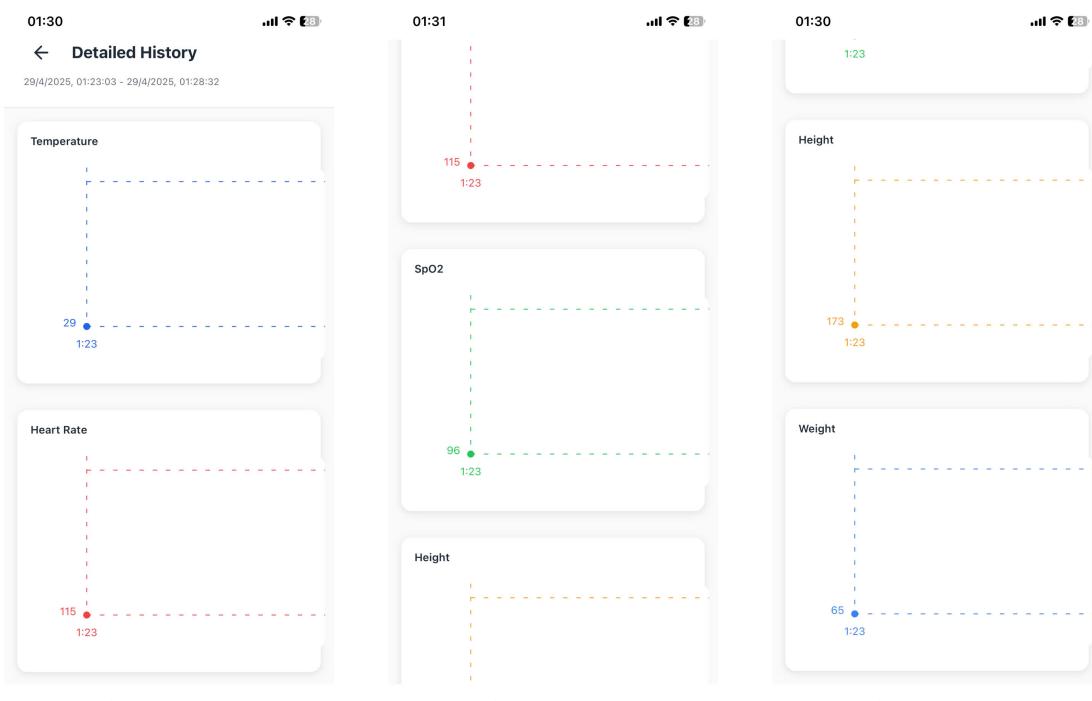


Figure 4.16: Lịch sử đo chi tiết

Chương 5

Tổng kết và định hướng phát triển

5.1 Tổng kết và đánh giá tiến độ

Trong 15 tuần nghiên cứu, hiện thực và kế thừa từ giai đoạn 1, nhóm đồ án đã hoàn thành một số mục tiêu quan trọng như:

Về phần cứng:

- Được làm quen và sử dụng các loại phần cứng mới
- Kết nối được đến các cảm biến với độ chính xác đã được tối ưu hóa.
- Kết nối được đến servo và sử dụng servo một cách mượt mà
- Kết nối BLE được đến cân Xiaomi Smart Scale 2, thu thập và lọc được dữ liệu từ cân.
- Lắp ráp hoàn chỉnh trọng lượng.

Về phần mềm:

- Hoàn thiện ứng dụng giao diện tương tác với người dùng sử dụng ReactJS, mang lại trải nghiệm thân thiện và hiệu quả.

- Mang lại những trải nghiệm hình ảnh và âm thanh cho người dùng khi sử dụng trạm cân.
- Xây dựng backend để gửi UART tới hệ thống ESP32, đảm bảo giao tiếp ổn định giữa các thành phần.
- Điều khiển camera để nhận diện khuôn mặt người dùng với độ hoàn thiện cao
- Nạp code vào mini PC, giúp hệ thống có thể hoạt động liên tục gần như bán tự động.
- Lưu trữ và gửi dữ liệu lên Firebase với khả năng cập nhật 24/24, đảm bảo rằng dữ liệu luôn được bảo quản và luôn có thể truy xuất.

Về ứng dụng:

- Xây dựng ứng dụng thân thiện người dùng: Hoàn thiện ứng dụng với các chức năng cơ bản như: thực hiện đo đạc, hiển thị lịch sử đo, điều khiển trạng thái thiết bị theo thời gian thực nhằm hỗ trợ bác sĩ tiến hành khám bệnh nhân một cách thuận tiện hơn.
- Xây dựng ứng dụng đa nền tảng: Ứng dụng có thể triển khai được trên nhiều thiết bị Android và IOS với nhiều kích thước màn hình khác nhau, không bị lỗi hiển thị mà vẫn đảm bảo các chức năng đều hoạt động ổn định.

Tuy đã đạt được nhiều thành tựu quan trọng nhưng do còn nhiều hạn chế về kỹ năng và thời gian nên nhóm cũng đã đối diện với không ít khó khăn như:

Về phần cứng:

- Vẫn chưa tối ưu hóa hoàn toàn hệ thống về mặt vận hành.
- Chưa thể tích hợp nhiều thiết bị đo các loại thông số sức khỏe vào hệ thống hơn như điện tâm đồ (ECG), đo huyết áp, đo đường huyết, vân vân.
- Hệ thống vẫn chưa được lắp ráp hoàn chỉnh do số lượng dây của hệ thống quá nhiều.

Về phần mềm:

- Khi hệ thống gặp sự cố như mất nguồn hoặc vấn đề liên quan đến kết nối dây, backend cần phải được reset lại, gây ảnh hưởng đến sự liên tục của hệ thống.
- Code truy xuất camera yêu cầu phải có một khoảng thời gian delay trước khi hình ảnh được hiển thị.
- Backend chưa đạt mức tự động hoàn toàn, dẫn đến trải nghiệm người dùng chưa thật sự tối ưu.

Về ứng dụng:

- Ứng dụng hiện tại chỉ dừng ở mức thu thập và hiển thị dữ liệu đo lường, chưa thực hiện phân tích để phát hiện bất thường và cảnh báo kịp thời cho người dùng.
- Ứng dụng chưa có chức năng phân tích dữ liệu chuyên sâu bằng AI để tự động đưa ra chẩn đoán sơ bộ hoặc lời khuyên chăm sóc sức khỏe.
- Cơ sở dữ liệu và server hiện tại mới chỉ đủ đáp ứng nhu cầu thử nghiệm, chưa tối ưu cho lượng lớn người dùng đồng thời.

5.2 Hướng phát triển trong tương lai

Với những hạn chế và khó khăn kể trên, nhóm đồ án có một số hướng phát triển để cải thiện hệ thống trong tương lai bao gồm:

Về phần cứng:

- Tối ưu hóa mặt vận hành của ESP32-S3, đo thông số nhanh hơn, tích hợp RTOS, ...
- Tích hợp thêm các loại cảm biến và thiết bị đo thông số sức khỏe khác.

- Giảm thiểu lượng dây dợ mà hệ thống đang có, giúp cho hệ thống sạch sẽ và gọn gàng hơn.

Về phần mềm:

- Đảm bảo backend luôn hoạt động ở trạng thái tốt nhất, bao gồm việc tăng khả năng tự phục hồi khi mất kết nối hoặc gặp sự cố nguồn, nhằm hạn chế tối đa việc phải reset thủ công.
- Hoàn thiện tính tự động hóa trên browser và cải thiện trải nghiệm tương tác, giúp người dùng cảm thấy thuận tiện và dễ dàng hơn trong việc sử dụng.

Về ứng dụng:

- Phát triển chức năng phân tích dữ liệu để phát hiện các chỉ số bất thường và gửi thông báo cảnh báo ngay lập tức tới người dùng.
- Nghiên cứu và tích hợp các mô hình học máy hoặc trí tuệ nhân tạo, tận dụng các API như Gemini hoặc ChatGPT nhằm tự động phân tích tình trạng sức khỏe và đề xuất các biện pháp phù hợp.
- Thiết kế lại cơ sở dữ liệu, tối ưu backend để có thể phục vụ nhiều người dùng đồng thời với tốc độ truy cập ổn định.

Tài liệu tham khảo

- [1] World Health Organization, *COVID-19 eliminated a decade of progress in global level of life expectancy*, Source: <https://www.who.int/news-room/detail/24-05-2024-covid-19-eliminated-a-decade-of-progress-in-global-level-of-life-expectancy>
- [2] Báo Quân đội nhân dân, *Tuổi thọ trung bình toàn cầu giảm vì Covid-19*, Source: <https://www.qdnd.vn/y-te/tin-tuc/tuoi-tho-trung-binh-toan-cau-giam-vi-covid-19-778370>
- [3] PMC PubMed Central, *Global and National Declines in Life Expectancy*, Source: <https://PMC.ncbi.nlm.nih.gov/articles/PMC10270701/>
- [4] World Bank Group, *Universal Health Coverage*, Source: <https://www.worldbank.org/en/topic/universalhealthcoverage>
- [5] Báo Nhân Dân, *Nhận diện những bệnh không lây nhiễm nguy hiểm hàng đầu ở Việt Nam*, Source: <https://nhandan.vn/nhan-dien-nhung-benh-khong-lay-nhiem-hang-dau-o-viet-nam-post733296.html>
- [6] vienktxh Hà Nội, *Bệnh không lây nhiễm chiếm 70 % gánh nặng bệnh tật ở Việt Nam*, Source: <https://vienktxh.hanoi.gov.vn/nghien-cuu-traodoi/benh-khong-lay-nhiem-chiem-70-ganh-nang-benh-tat-o-viet-nam-59459.html>

- [7] BASE.VN, *Ứng dụng AI trong Y tế: Bước tiến mới cho ngành chăm sóc sức khỏe*, Source: <https://base.vn/blog/ai-trong-y-te/>
- [8] IBM, *What is the Internet of Things (IoT)?*, Source: <https://www.ibm.com/think/topics/internet-of-things>
- [9] QSystems, *IoT trong chăm sóc sức khỏe - Các thách thức (Phần 2)*, Source: <https://qsystemsco.com/iot-trong-cham-soc-suc-khoe-cac-thach-thuc.html>
- [10] nextech, *IoT Gateway là gì? Các chức năng của IoT Gateway*, Source: <https://nextech.vn/dinh-nghia-iot-gateway/>
- [11] A Basic Guide to I2C, Source: <https://www.ti.com/lit/an/sbaa565/sbaa565.pdf>
- [12] Circuit Basics, *Basics of the I2C Communication Protocol*, Source: <https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol>
- [13] How to Hash Passwords with bcrypt in Node.js, , Source: <https://www.freecodecamp.org/news/how-to-hash-passwords-with-bcrypt-in-nodejs/>
- [14] NovelBits, *Bluetooth Low Energy (BLE): A Complete Guide*, Source: <https://novelbits.io/bluetooth-low-energy-ble-complete-guide/>
- [15] MDPI, *A Comparative Study of Forehead Temperature and Core Body Temperature under Varying Ambient Temperature Conditions*, Source: <https://www.mdpi.com/1660-4601/19/23/15883>
- [16] feevr, *Visual presentation of temperature*, Source: <https://d1tzzns6d79su2.cloudfront.net/uploads/e3c3/2296/feevr-temp-justification.pdf>

- [17] healthline, *Is My Blood Oxygen Level Normal?*, Source: <https://www.healthline.com/health/normal-blood-oxygen-level>
- [18] Asthma+ Lung UK, *Oxygen level testing*, Source: <https://www.asthmaandlung.org.uk/symptoms-tests-treatments/tests/oxygen-level-tests>
- [19] Cleveland Clinic, *Blood Oxygen Level*, Source: <https://my.clevelandclinic.org/health/diagnostics/22447-blood-oxygen-level>
- [20] npm, *serialport* , Source: <https://www.npmjs.com/package/serialport>
- [21] w3schools, *React Router* , Source: https://www.w3schools.com/react/react_router.asp
- [22] Firebase, *Add Firebase to your JavaScript project*, Source: <https://firebase.google.com/docs/web/setup>
- [23] Node.js Tutorial, Source: <https://www.mongodb.com/docs/guides/>
- [24] Web Dev Simplified, *Building a REST API with Express, Node, and MongoDB*, Source: <https://www.mongodb.com/resources/languages/express-mongodb-rest-api-tutorial>
- [25] JWT, *Introduction to JSON Web Tokens*, Source: <https://jwt.io/introduction>
- [26] Medium, *Node.js, Express, MongoDB, and Mongoose: Understanding the MVC Model*, Source: <https://medium.com/@gecno/node-js-express-mongodb-and-mongoose-understanding-the-mvc-model-678952631ea3>
- [27] Medium, *Building a RESTful API with Node.js, Express, and MongoDB using MVC Architecture*, Source:

https://medium.com/@Prathmesh_Chavan/building-a-restful-api-with-node-js-express-and-mongodb-using-mvc-architecture-c418143a882a

- [28] StackoverFlow, *Which Pins to use for second I2C-Bus on Adafruit Featherwing*, Source: <https://stackoverflow.com/questions/73064367/which-pins-to-use-for-second-i2c-bus-on-adafruit-featherwing>
- [29] Random Nerd Tutorials, *ESP32 I2C Communication: Set Pins, Multiple Bus Interfaces and Peripherals (Arduino IDE)*, Source: <https://randomnerdtutorials.com/esp32-i2c-communication-arduino-ide/>
- [30] ElectronicWings, *MAX30100 Pulse Oximeter Interfacing with Arduino*, Source: <https://www.electronicwings.com/arduino/max30100-pulse-oximeter-interfacing-with-arduino>
- [31] ElectronicWings, *Interfacing MAX30102 Pulse Oximeter and Heart Rate Sensor with Arduino*, Source: <https://lastminuteengineers.com/max30102-pulse-oximeter-heart-rate-sensor-arduino-tutorial>
- [32] Arduino.cc, *Code modification for measuring Heart Pulse Rate with MAX30102 pulse sensor*, Source: <https://forum.arduino.cc/t/code-modification-for-measuring-heart-pulse-rate-with-max30102-pulse-sensor/1056605>
- [33] SparkFun *MAX301x Particle Sensor Library*, Source: https://github.com/sparkfun/SparkFun_MAX3010x_Sensor_Library
- [34] *Pulseoximeter using MAX30102*, Source: <https://github.com/adeelahmad94/pulseoximeter-max30102>

[35] *MAX30102-by-RF*, Source: <https://github.com/aromring/MAX30102-by-RF>

[36] pangodream, *Read a Xiaomi Mi Smart scale using an ESP32*, Source: <https://www.pangodream.es/read-xiaomi-mi-smart-scale-using-an-esp32>

[37] Firebase, *Add Firebase to your Android project* , Source: <https://firebase.google.com/docs/android/setup>

[38] Developers, *PdfDocument*, Source: <https://developer.android.com/reference/android/graphics/pdf/PdfDocument>

[39] *MPAndroidChart*, Source: <https://github.com/PhilJay/MPAndroidChart>

[40] All about circuits, *Exploring the Basics of Bluetooth Low Energy: A Beginners Guide To BLE*, Source: <https://www.allaboutcircuits.com/technical-articles/exploring-the-basics-of-bluetooth-low-energy-a-beginners-guide-to-ble/>

[41] NovelBits, *Bluetooth Low Energy (BLE): A Complete Guide*, Source: <https://novelbits.io/bluetooth-low-energy-ble-complete-guide/>

[42] LastMinuteEngineers, *Interfacing MAX30102 Pulse Oximeter and Heart Rate Sensor with Arduino* , Source: <https://lastminuteengineers.com/max30102-pulse-oximeter-heart-rate-sensor-arduino-tutorial/>

[43] World Health Organization, *World health statistics 2024* , Source: <https://iris.who.int/bitstream/handle/10665/376869/9789240094703-eng.pdf?sequence=1>

[44] PMC PubMed Central, *A brief report on the normal range of forehead temperature as determined by*

noncontact, handheld, infrared thermometer, Source:
<https://PMC7115295/>

[45] American Heart Association, *Target Heart Rates Chart*, Source:
<https://www.heart.org/en/healthy-living/fitness/fitness-basics/target-heart-rates>

[46] Penn Medicine, *What's a Good Heart Rate for My Age?*,
Source: <https://www.chestercountyhospital.org/news/health-eliving-blog/2023/january/whats-a-good-heart-rate-for-my-age>

[47] Forbes, *Normal Resting Heart Rate By Age (Chart)*, Source:
<https://www.forbes.com/health/wellness/normal-heart-rate-by-age/>

[48] OliE (oliexdev), *Xiaomi Bluetooth Mi Scale*, Source:
<https://github.com/oliexdev/openScale/wiki/Xiaomi-Bluetooth-Mi-Scale>

[49] Pangodream, *Read a Xiaomi Mi Smart scale using an ESP32*, Source:
<https://www.pangodream.es/read-xiaomi-mi-smart-scale-using-an-esp32>