HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO CUỐI KỲ IOT

HỆ THỐNG THEO DÕI SỨC KHỎE THEO THỜI GIAN THỰC.

LỚP CNPM01

Sinh viên	Mã sinh viên
Nguyễn Thị Hiền	B22DCCN289
Vương Thu Thảo	B22DCCN805
Nguyễn Thị Thu Phương	B22DCCN637

Hà Nội 2025

Mục lục

Chương 1 : Giới thiệu đề tài

I. Tổng quan về dự án

1. Bối cảnh và lý do chọn đề tài

- Trong bối cảnh xã hội hiện đại, nhu cầu theo dõi sức khỏe cá nhân ngày càng trở nên quan trọng, đặc biệt là đối với người lớn tuổi, người có bệnh nền hoặc những người thường xuyên vận động. Việc theo dõi các chỉ số sức khỏe như nhịp tim và trạng thái vận động một cách liên tục có thể giúp phát hiện sớm các dấu hiệu bất thường, từ đó có biện pháp can thiệp kịp thời, giảm thiểu rủi ro cho sức khỏe.
- Tuy nhiên, các thiết bị y tế chuyên dụng thường có giá thành cao và không phải lúc nào cũng tiện lợi để sử dụng hàng ngày. Nhận thấy tiềm năng của công nghệ Internet vạn vật (IoT) trong việc giải quyết vấn đề này, nhóm chúng em đã quyết định thực hiện đề tài xây dựng "hệ thống theo dõi sức khỏe theo thời gian thực" .

2. Ý nghĩa và ứng dụng thực tế

- Theo dõi sức khỏe: Cung cấp một giải pháp chi phí thấp để người dùng có thể tự theo dõi nhịp tim và mức độ vận động của mình theo thời gian thực.
- Cảnh báo sớm: Hệ thống có khả năng phát hiện khi nhịp tim vượt ngưỡng an toàn và đưa ra cảnh báo tức thì, giúp người dùng và người thân yên tâm hơn.
- Ứng dụng AI: Tích hợp trí tuệ nhân tạo để phân tích dữ liệu, không chỉ hiển thị thông số mà còn đưa ra dự đoán về trạng thái sức khỏe, mang lại giá trị cao hơn cho người dùng.
- Dự án này hướng tới việc xây dựng một hệ thống hoàn chỉnh từ phần cứng thu thập dữ liệu đến phần mềm phân tích và hiển thị, mang lại một công cụ hữu ích và dễ tiếp cận cho mọi người.

II. Mục tiêu hệ thống

- Mục tiêu của hệ thống là xây dựng một nền tảng theo dõi sức khỏe thông minh theo thời gian thực, kết hợp giữa dữ liệu sinh học (nhịp tim) và dữ liệu chuyển động (gia tốc) nhằm đánh giá và giám sát tình trạng sức khỏe của người dùng một cách chính xác, liên tục và tự động. Cụ thể, hệ thống hướng đến các mục tiêu thiết thực :
- Thu thập dữ liệu thời gian thực:

- Sử dụng cảm biến MAX30102 để đo nhịp tim và nồng độ oxy trong máu (SpO_2).
- Sử dụng cảm biến ADXL345 để ghi nhận chuyển động, tư thế hoặc rung động bất thường của cơ thể.
- Dữ liệu từ hai cảm biến được vi điều khiển ESP32 xử lý sơ bộ và gửi liên tục về server thông qua giao thức MQTT.

- Xử lý và phân tích dữ liệu thông minh:
  - Server tiếp nhận dữ liệu, lưu trữ và xử lý theo thời gian thực.
  - Ứng dụng mô hình trí tuệ nhân tạo (AI) để nhận dạng trạng thái sức khỏe hiện tại của người dùng dựa trên đặc trưng nhịp tim và chuyển động (ví dụ: bình thường, hoạt động mạnh, hoặc cảnh báo sức khỏe bất thường).
- Phát hiện và cảnh báo sớm:
  - Khi mô hình phát hiện dấu hiệu bất thường (như nhịp tim quá cao/thấp, mất cân bằng, té ngã), hệ thống sẽ tự động gửi cảnh báo đến người dùng thông qua giao diện web và thông báo tức thời.
  - Mục tiêu là giúp người dùng hoặc người thân kịp thời can thiệp trong các tình huống khẩn cấp.
- Hiển thị và giám sát trực quan:
  - Cung cấp giao diện web cho phép người dùng đăng nhập và theo dõi sức khỏe cá nhân.
  - Dữ liệu được hiển thị dưới dạng biểu đồ thời gian thực, giúp người dùng dễ dàng nhận biết xu hướng và biến động của các chỉ số sức khỏe.
- Hệ thống được thiết kế nhằm kết hợp phần cứng IoT và trí tuệ nhân tạo để tạo ra một công cụ hỗ trợ chăm sóc sức khỏe thông minh, có khả năng giám sát – phân tích – cảnh báo liên tục, góp phần nâng cao ý thức và an toàn sức khỏe cho người dùng trong đời sống hàng ngày.

III. Phạm vi đề tài

- Hệ thống "Theo dõi sức khỏe theo thời gian thực" được thiết kế nhằm giám sát sức khỏe theo thời gian thực. Tuy nhiên, phạm vi triển khai của đề tài này được giới hạn để phù hợp với thời gian và thiết bị hiện có:

1. Phần cứng

- Sử dụng ESP32 làm bộ vi điều khiển trung tâm.
- ESP32 kết nối với cảm biến nhịp tim MAX30102 và cảm biến gia tốc ADXL345 để thu thập dữ liệu BPM (nhịp tim) và gia tốc 3 trục.
- Người dùng cần đặt ngón tay lên thiết bị để đo nhịp tim

2. Phần mềm

- Backend: sử dụng Flask để nhận dữ liệu từ ESP32 qua WiFi, xử lý bằng mô hình AI và gửi kết quả cho web.
- Frontend: xây dựng giao diện web để hiển thị dữ liệu cảm biến (bảng dữ liệu, biểu đồ) và trạng thái sức khỏe theo thời gian thực.
- Phân quyền người dùng:

    - Tài khoản Quản lý: quản lý dữ liệu của bác sĩ, bệnh nhân
    - Tài khoản Bác sĩ: xem dữ liệu sức khỏe của bệnh nhân
    - Tài khoản Bệnh nhân: chỉ xem dữ liệu và cảnh báo của bản thân.

3. Giới hạn phạm vi

- Hệ thống chỉ hoạt động trên web phân quyền.
- Dữ liệu được lưu và xử lý trực tiếp trên server Flask, không dùng cơ sở dữ liệu đám mây quy mô lớn.
- Hệ thống hiện tại chỉ tập trung vào các chỉ số:

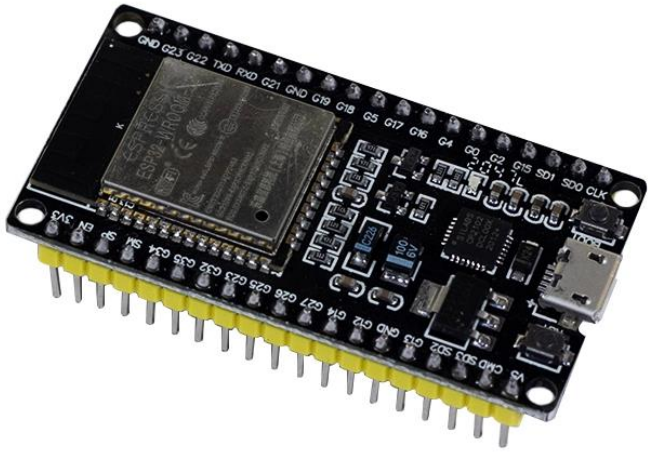
    - Nhịp tim (BPM)
    - Gia tốc 3 trục
    - Chưa mở rộng sang các loại cảm biến khác như ECG, nhiệt độ cơ thể, huyết áp…

Chương 2: Các công nghệ, lý thuyết áp dụng

I. Phần cứng

1. Vi điều khiển ESP32

- ESP32 là một vi điều khiển hiệu năng cao do Espressif Systems phát triển, được tích hợp sẵn WiFi và Bluetooth. Nhờ đó, thiết bị này trở thành một trong những lựa chọn phổ biến nhất cho các ứng dụng Internet of Things (IoT), đặc biệt là trong các hệ thống cần kết nối mạng, xử lý tín hiệu cảm biến và truyền dữ liệu thời gian thực.



Hình ảnh 1: Vi điều khiển ESP32

- Đặc điểm kỹ thuật nổi bật:

- Bộ xử lý dual-core Xtensa 32-bit, tốc độ lên đến 240 MHz.
- Bộ nhớ trong 512 KB RAM, có thể mở rộng qua SPI Flash.
- Hỗ trợ đa dạng giao tiếp: I2C, SPI, UART, PWM, ADC, DAC, GPIO.
- Tích hợp WiFi (802.11 b/g/n) và Bluetooth v4.2 BLE.
- Điện áp hoạt động: 2.2V – 3.6V, tiêu thụ năng lượng thấp (hỗ trợ deep-sleep mode).

- Ưu điểm:

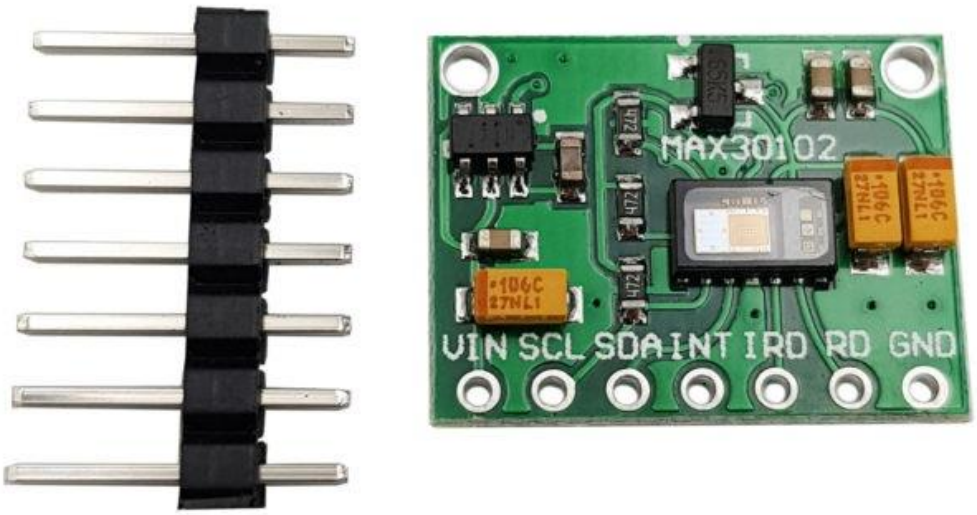
- Tốc độ xử lý cao, đảm bảo khả năng thu thập dữ liệu theo thời gian thực.
- Dễ dàng lập trình bằng Arduino IDE hoặc ESP-IDF.

- Hỗ trợ thư viện phong phú, thuận tiện cho việc phát triển nhanh.
- Tiêu thụ điện năng thấp, phù hợp cho thiết bị đeo hoặc hoạt động liên tục.

- ESP32 là trung tâm xử lý của thiết bị, có nhiệm vụ đọc dữ liệu từ các cảm biến và gửi lên server. ESP32 được lựa chọn vì tích hợp sẵn Wi-Fi, hiệu năng cao và hỗ trợ đa nhiệm với FreeRTOS, cho phép đọc dữ liệu từ nhiều cảm biến cùng lúc một cách hiệu quả.

2. Cảm biến MAX30102

- MAX30102 là cảm biến quang học được sử dụng để đo nhịp tim và nồng độ oxy trong máu (SpO_2). Cảm biến này tích hợp sẵn hai đèn LED (đỏ và hồng ngoại), photodiode, các bộ lọc quang học, và mạch khuếch đại tín hiệu trong cùng một module nhỏ gọn.



Hình ảnh 2: Cảm biến MAX30102

- Nguyên lý hoạt động:

- Khi đèn LED phát sáng, một phần ánh sáng sẽ bị mạch máu hấp thụ và phần còn lại phản xạ lại về photodiode.
- Khi tim co bóp và giãn ra, lượng máu chảy qua mao mạch thay đổi → làm thay đổi cường độ ánh sáng phản xạ.
- Cảm biến ghi nhận sự thay đổi này để tính toán nhịp tim (BPM) và SpO_2 (% oxy bão hòa trong máu).

- Ưu điểm:

- Độ nhạy cao, tiêu thụ điện năng thấp.

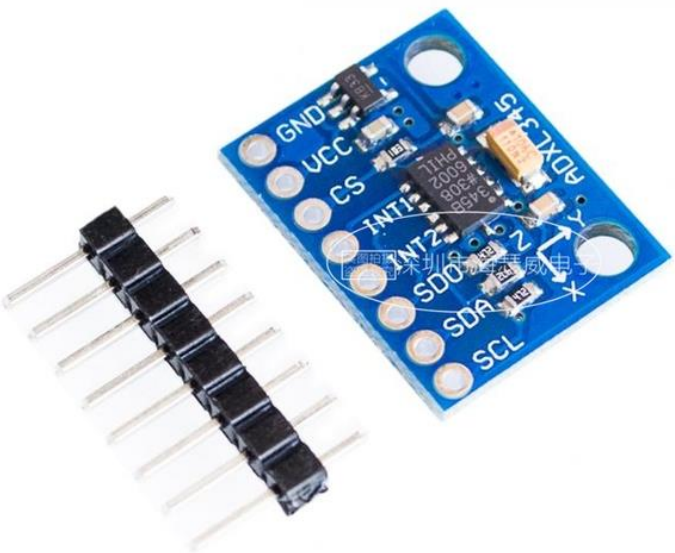
- Dễ dàng giao tiếp qua I2C.
- Có khả năng hoạt động ổn định trong môi trường ánh sáng yếu.

- Ứng dụng trong dự án:

- Ngón tay người dùng sẽ đặt vào cảm biến để đo liên tục.
- ESP32 sẽ đọc dữ liệu từ MAX30102 định kỳ (mỗi 5 giây).
- Dữ liệu được xử lý sơ bộ bằng bộ lọc Kalman hoặc moving average để giảm nhiễu.
- Sau đó gửi dữ liệu qua MQTT đến Flask server để lưu trữ, hiển thị và phân tích AI.

3. Cảm biến ADXL345

- ADXL345 là một cảm biến gia tốc điện dung vi cơ (MEMS) 3 trục (X, Y, Z) nhỏ gọn và tiêu thụ điện năng thấp. Nó có khả năng đo lường cả gia tốc tĩnh (như trọng lực) và gia tốc động (do chuyển động, va chạm), làm cho nó trở thành một lựa chọn lý tưởng để theo dõi trạng thái vật lý của người dùng.



Hình ảnh 3: Cảm biến ADXL345

- Nguyên lý hoạt động: Bên trong ADXL345 là một cấu trúc vi cơ cực nhỏ được treo lơ lửng. Khi có gia tốc tác động lên cảm biến, cấu trúc này sẽ di chuyển. Sự dịch chuyển này làm thay đổi khoảng cách giữa các bản cực của các tụ điện vi mô, dẫn đến sự thay đổi về điện dung. Mạch điện tử bên trong cảm biến sẽ đo lường sự thay đổi điện dung này và chuyển đổi nó thành giá trị gia tốc số trên 3 trục X, Y, và Z.
- Ứng dụng và xử lý tín hiệu trong dự án: ADXL345 đóng vai trò cung cấp "bối cảnh" cho dữ liệu nhịp tim, giúp hệ thống AI hiểu được người dùng đang làm gì.

- Cảm biến được cấu hình để có thể ghi nhận một dải rộng các chuyển động, từ những cử động nhẹ nhàng cho đến những tác động mạnh và đột ngột, chẳng hạn như khi người dùng bị ngã. Dữ liệu gia tốc trên cả ba trục (X, Y, Z) được đọc và phân tích liên tục để xác định cường độ vận động tổng thể. Phép đo này cho phép hệ thống đánh giá mức độ hoạt động của người dùng một cách chính xác.

- Dữ liệu về chuyển động này là một yếu tố then chốt, giúp hệ thống phân tích AI trở nên thông minh hơn. Bằng cách kết hợp thông tin nhịp tim với mức độ hoạt động, hệ thống có thể phân biệt rõ ràng các kịch bản khác nhau:

- Nhịp tim cao kèm theo vận động mạnh: Được hiểu là trạng thái bình thường khi người dùng đang tập thể dục hoặc hoạt động thể chất.
- Nhịp tim cao nhưng không có vận động đáng kể: Được xem là một dấu hiệu tiềm ẩn bất thường, cần được cảnh báo.

II. Liên kết phần cứng với phần mềm

1. Giao thức I2C

- Giao thức I2C (Inter-Integrated Circuit) là một giao thức truyền thông nối tiếp hai dây, đồng bộ, hai chiều, được sử dụng để giao tiếp tốc độ thấp và khoảng cách ngắn giữa các chip và thiết bị trong một hệ thống. Giao thức này hoạt động theo mô hình chủ-tớ, sử dụng hai dây chính là SCL (đường xung nhịp) và SDA (đường dữ liệu), cho phép kết nối nhiều thiết bị trên cùng một bus bằng cách sử dụng các địa chỉ riêng biệt cho từng thiết bị.
- Cách thức hoạt động

- Dây tín hiệu:

    - SCL (Serial Clock Line): Dây này truyền tín hiệu xung nhịp đồng hồ, do thiết bị chủ tạo ra để đồng bộ hóa quá trình truyền dữ liệu.
    - SDA (Serial Data Line): Dây này dùng để gửi và nhận dữ liệu giữa thiết bị chủ và các thiết bị tớ.

- Cấu trúc chủ-tớ:
    - Thiết bị chủ (Master): Khởi tạo và điều khiển quá trình truyền, gửi địa chỉ của thiết bị tớ mà nó muốn giao tiếp.
    - Thiết bị tớ (Slave): Thiết bị phản hồi khi địa chỉ của nó được gọi. Có thể có nhiều thiết bị tớ trên một bus I2C.
- Cơ chế địa chỉ hóa: Mỗi thiết bị tớ trên bus có một địa chỉ duy nhất để thiết bị chủ có thể chọn và giao tiếp với nó, giống như lấy số điện thoại của một người cụ thể trong danh bạ.
- Điện trở kéo lên (Pull-up Resistor): Cần thiết để giữ cho các đường SDA và SCL ở mức logic cao (trạng thái mặc định) khi không có thiết bị nào chủ động kéo chúng xuống mức thấp.

- Ưu điểm của I2C

- Tiết kiệm chân IO: Chỉ cần hai dây để kết nối nhiều thiết bị, giúp tiết kiệm tài nguyên trên vi điều khiển.
- Hỗ trợ nhiều chủ/nhiều tớ: Có thể kết nối nhiều thiết bị tớ với một thiết bị chủ, hoặc thậm chí có nhiều thiết bị chủ trên một bus (dù chúng phải luân phiên sử dụng bus).
- Thiết kế đơn giản: Chỉ cần hai dây tín hiệu cho toàn bộ hệ thống.

- Trong hệ thống theo dõi sức khỏe, giao thức I²C được sử dụng để ESP32 giao tiếp với hai cảm biến MAX30102 và ADXL345. ESP32 đóng vai trò Master, còn MAX30102 và ADXL345 là các thiết bị Slave. Chúng hoạt động dựa trên luồng giao tiếp:

1. ESP32 gửi tín hiệu START trên bus I²C.
2. ESP32 gửi địa chỉ thiết bị MAX30102, ADXL345 cùng bit đọc/ghi (R/W).
3. Cảm biến có địa chỉ trùng sẽ ACK (xác nhận).
4. ESP32 gửi hoặc đọc dữ liệu từ thanh ghi của cảm biến.
5. Khi hoàn tất, ESP32 gửi tín hiệu STOP để kết thúc truyền.

- I2C là cầu nối giữa tầng cảm biến phần cứng và tầng xử lý IoT trong hệ thống, giúp ESP32 thu thập dữ liệu chính xác, tiết kiệm dây nối, và đảm bảo độ ổn định trước khi truyền đi qua MQTT. I2C cho phép truyền dữ liệu song song từ nhiều thiết bị chỉ với hai dây SDA và SCL, giúp tiết kiệm phần cứng, dễ mở rộng, và đảm bảo tính ổn định trong việc thu thập dữ liệu sinh học phục vụ cho xử lý AI ở tầng server.

2. Giao thức truyền dẫn Wifi

- Giao thức truyền dẫn Wi-Fi (IEEE 802.11) là một bộ các quy tắc và tiêu chuẩn cho phép các thiết bị kết nối không dây và truyền dữ liệu qua sóng radio, thay vì kết nối có dây. Nó hoạt động dựa trên các tiêu chuẩn IEEE 802.11 để thiết lập mạng nội bộ (LAN) không dây, sử dụng bộ phát sóng (router) để giải mã và gửi dữ liệu qua internet.
- Cách thức hoạt động:

- Truyền sóng Radio: Tương tự như điện thoại di động, Wi-Fi sử dụng sóng radio (sóng vô tuyến) để truyền thông tin.
- Card mạng không dây: Thiết bị của bạn có một card mạng không dây sẽ chuyển dữ liệu thành tín hiệu radio.
- Ăng-ten và bộ giải mã (Router): Tín hiệu radio được truyền đi qua ăng-ten của thiết bị, sau đó được bộ giải mã (router) nhận và giải mã.
- Kết nối Internet: Router sẽ gửi dữ liệu đã giải mã đến Internet thông qua kết nối Ethernet có dây.
- Giao tiếp hai chiều: Khi nhận dữ liệu từ Internet, router sẽ mã hóa chúng thành tín hiệu radio để card mạng không dây của máy tính nhận và xử lý.

- Trong hệ thống theo dõi sức khỏe thời gian thực, Wi-Fi chính là cầu nối giữa thiết bị thu thập dữ liệu (ESP32) và server.

- ESP32 kết nối vào mạng Wi-Fi nội bộ thông qua SSID và mật khẩu.
- Khi kết nối thành công, ESP32 sẽ có địa chỉ IP trong mạng cục bộ.
- ESP32 gửi dữ liệu cảm biến (nhịp tim từ MAX30102, gia tốc từ ADXL345) đến server Flask qua MQTT (chạy trên nền TCP/IP qua Wi-Fi).

- Server xử lý, phân tích bằng mô hình AI, sau đó gửi cảnh báo hoặc phản hồi ngược lại cho ESP32 (cũng qua Wi-Fi).

- Wi-Fi đóng vai trò là tầng truyền dẫn vật lý và mạng giúp các giao thức ở tầng ứng dụng (như MQTT, HTTP) hoạt động được. Không có Wi-Fi, ESP32 không thể gửi dữ liệu cảm biến lên server Flask, do đó đây là giao thức truyền dẫn cốt lõi của toàn hệ thống.

3. Giao thức truyền tin MQTT

- MQTT là một giao thức nhắn tin nhẹ (lightweight) dựa trên mô hình xuất bản/đăng ký (Pub/Sub), được thiết kế cho Internet Vạn Vật (IoT) và các thiết bị có tài nguyên hạn chế. Giao thức này cho phép các thiết bị truyền và nhận dữ liệu hiệu quả qua các mạng có băng thông thấp hoặc không ổn định, bằng cách sử dụng một trình môi giới (broker) làm trung gian để phân phối tin nhắn dựa trên các chủ đề (topics). [1]

- Cách thức hoạt động:

- Mô hình Pub/Sub: Các thiết bị (máy khách) không kết nối trực tiếp với nhau.
    - Người xuất bản (Publisher): Thiết bị gửi dữ liệu đến trình môi giới, chỉ định chủ đề mà dữ liệu đó thuộc về.
    - Người đăng ký (Subscriber): Thiết bị quan tâm đến một chủ đề cụ thể sẽ đăng ký chủ đề đó với trình môi giới.
    - Trình môi giới (Broker): Máy chủ trung tâm này nhận tin nhắn từ những người xuất bản và chuyển tiếp chúng đến tất cả những người đăng ký quan tâm đến chủ đề đó.
- Chủ đề (Topics): Dữ liệu được tổ chức theo các chuỗi ký tự (chủ đề), giúp trình môi giới lọc tin nhắn và chỉ gửi đến các máy khách đã đăng ký.

- Đặc điểm chính:

- Gọn nhẹ: Giảm thiểu lượng dữ liệu tiêu thụ trên các thiết bị có tài nguyên hạn chế và mạng có băng thông thấp.
- Dựa trên TCP/IP: Hoạt động trên nền tảng TCP/IP, đảm bảo kết nối hai chiều, đáng tin cậy và không mất dữ liệu.

- Hiệu quả cho IoT: Lý tưởng cho các ứng dụng IoT, nơi các thiết bị cần giao tiếp với đám mây hoặc với nhau trong điều kiện mạng không ổn định.
- Tách biệt và tập trung: Mô hình Pub/Sub tách biệt hoàn toàn người gửi và người nhận, với trình môi giới làm trung gian kết nối.
- Truyền thông hai chiều: Cho phép cả dữ liệu từ thiết bị lên đám mây và lệnh từ đám mây xuống thiết bị.

- Trong hệ thống MQTT sẽ hoạt động như sau

1. ESP32 thu thập dữ liệu nhịp tim (MAX30102) và gia tốc (ADXL345).
2. Dữ liệu được publish lên MQTT Broker
3. Flask server đóng vai trò subscriber, nhận dữ liệu này để xử lý bằng mô hình AI.
4. Flask sau khi phân tích sẽ publish ngược lại kết quả dự đoán/cảnh báo lên topic khác
5. Web client (dashboard) subscribe topic cảnh báo để hiển thị real-time cho người dùng.

→ Nhờ cơ chế Publish/Subscribe này, mọi thành phần đều hoạt động tách biệt, không cần kết nối trực tiếp, giúp hệ thống linh hoạt và dễ mở rộng.

- Trong hệ thống theo dõi sức khỏe theo thời gian thực, giao thức MQTT được sử dụng để truyền dữ liệu giữa thiết bị ESP32 và máy chủ Flask. ESP32 đóng vai trò Publisher, gửi dữ liệu cảm biến (nhịp tim và gia tốc) lên MQTT Broker. Flask server đóng vai trò Subscriber, nhận dữ liệu, xử lý bằng mô hình AI và gửi cảnh báo ngược lại qua MQTT. Việc sử dụng MQTT giúp hệ thống đạt được tốc độ truyền nhanh, độ trễ thấp, phù hợp cho ứng dụng IoT real-time yêu cầu cảnh báo sức khỏe kịp thời. [2]

III. Phần mềm

1. Frontend - Giao diện người dùng

a. Vai trò của frontend:

- Hiển thị dữ liệu cảm biến:
  - Hiện IR Value, BPM, Avg BPM
  - Hiện gia tốc 3 trục X, Y, Z và tổng gia tốc
- Biểu đồ trực quan

- Vẽ đường thay đổi nhịp tim và gia tốc theo thời gian (Chart.js)

- Cảnh báo sức khỏe

- Hiện thông báo màu sắc: Bình thường, Cảnh báo, Nguy hiểm

- Tương tác người dùng

- Là nơi bác sĩ hoặc người chăm sóc quan sát dữ liệu
- Có thể bổ sung nút tải dữ liệu, nút gửi cảnh báo khẩn cấp…

b. Các công nghệ sử dụng

- HTML (HyperText Markup Language)

- Ngôn ngữ đánh dấu siêu văn bản, dùng để tạo cấu trúc cơ bản của các trang web.
- Xây dựng các phần tử như tiêu đề, bảng, biểu mẫu (form), nút bấm, khu vực hiển thị dữ liệu từ cảm biến.
- Trong hệ thống này, HTML được dùng để tạo khung giao diện theo dõi nhịp tim, gia tốc và các biểu đồ.

- CSS (Cascading Style Sheets)

- Dùng để tạo kiểu dáng, bố cục và màu sắc cho giao diện web.
- Giúp giao diện trở nên thân thiện, trực quan, dễ nhìn với người dùng.
- Tối ưu cho trải nghiệm người dùng trên nhiều kích thước màn hình (responsive).

- JavaScript (JS)

- Ngôn ngữ lập trình chạy trên trình duyệt, dùng để tạo các chức năng tương tác động cho trang web.
- Trong hệ thống
    - Gửi yêu cầu (AJAX/Fetch) đến server Flask để lấy dữ liệu thời gian thực từ ESP32.
    - Cập nhật dữ liệu nhịp tim, gia tốc lên biểu đồ mà không cần tải lại trang.
    - Quản lý các sự kiện người dùng (ví dụ: nút đăng nhập, nút đăng xuất, chọn khoảng thời gian hiển thị dữ liệu).
    - Kết hợp thư viện hiển thị biểu đồ (như Chart.js) để trực quan hóa dữ liệu.

2. Mô hình AI

- Hệ thống áp dụng AI sử dụng mạng nơ-ron để phân loại trạng thái sức khỏe dựa trên dữ liệu nhịp tim và gia tốc từ ESP32. Mô hình giúp đánh giá sức khỏe tự động, cung cấp cảnh báo tức thời cho giao diện web.

a. Vai trò của AI trong hệ thống

- AI được dùng để phân tích dữ liệu cảm biến (nhịp tim từ MAX30102 và gia tốc từ ADXL345) nhằm:

- Phân loại tình trạng sức khỏe của người dùng:
  - Bình thường / Hoạt động nhẹ / Hoạt động mạnh
  - Cảnh báo sức khoẻ không ổn định / Cảnh báo nguy hiểm
- Cung cấp dữ liệu đầu ra cho Frontend để hiển thị cảnh báo trực quan.

b. Dữ liệu đầu vào cho AI

- AvgBPM – Nhịp tim trung bình từ MAX30102.
- A_total – Tổng gia tốc tính từ 3 trục X, Y, Z của ADXL345.
- Các dữ liệu này được thu liên tục từ ESP32, lưu lại để huấn luyện mô hình.

c. Cách AI hoạt động

- Tiền xử lý dữ liệu:
  - Làm sạch dữ liệu, chuẩn hóa giá trị BPM và A_total.
  - Gán nhãn trạng thái sức khỏe dựa trên quy tắc ban đầu để huấn luyện.
- Xây dựng mô hình học máy:
  - Sử dụng TensorFlow/Keras để tạo mạng nơ-ron nhiều lớp
  - Đầu vào: 2 đặc trưng (AvgBPM, A_total).
  - Đầu ra: Xác suất của các nhãn sức khỏe.
- Huấn luyện và tối ưu:
  - Sử dụng EarlyStopping để tránh quá khớp (overfitting).
  - Dùng class_weight để cân bằng dữ liệu giữa các nhãn.
- Triển khai:
  - Mô hình sau huấn luyện được lưu thành file.

- Flask backend tải mô hình này và dự đoán tình trạng người dùng theo thời gian thực.

d. Kết quả AI cung cấp

- Trạng thái sức khỏe: Bình thường, Hoạt động nhẹ, Hoạt động trung bình, Hoạt động mạnh, Cảnh báo sức khỏe không ổn định, Hoạt động mạnh bất thường, Nhịp tim cao bất thường, Hoạt động thể chất mạnh, Hoạt động cực độ, Sức khỏe siêu tốt, Cảnh báo nguy hiểm, Không xác định

3. Xử lý backend và cơ sở dữ liệu

- Vai trò của Flask: Flask được sử dụng làm web server backend cho toàn hệ thống, đảm nhận:

- Kết nối với MQTT broker để nhận và gửi dữ liệu.
- Tải mô hình AI để xử lý dự đoán.
- Cung cấp giao diện web (web client) cho người dùng.
- Kết nối với cơ sở dữ liệu để lưu trữ thông tin sức khỏe.

- Luồng xử lý dữ liệu

- Nhận dữ liệu từ MQTT: Flask subscribe topic 1 để lắng nghe dữ liệu cảm biến.
- Xử lý AI: Dữ liệu nhận được sẽ được chuẩn hóa, đưa vào mô hình AI để dự đoán trạng thái: hoạt động bình thường, hoặc cảnh báo dựa trên nhịp tim quá cao, chuyển động bất thường
- Phản hồi qua MQTT: Flask gửi kết quả dự đoán trở lại MQTT topic 2
- Hiển thị lên web: Flask render kết quả lên giao diện HTML, đồng thời cập nhật liên tục dữ liệu theo thời gian thực

- Hệ thống sử dụng SQL để lưu trữ dữ liệu:

- Thông tin người dùng (tài khoản, phân quyền).
- Dữ liệu cảm biến (nhịp tim, gia tốc, thời gian đo).
- Lịch sử dự đoán và cảnh báo.

Chương 3: Các tính năng dự kiến triển khai

I. Thông tin các đối tượng được xử lý và mối quan hệ giữa chúng

- Hệ thống quản lý 3 đối tượng chính bao gồm: bệnh nhân, bác sĩ, quản lý
- Bệnh nhân

  - Có các thông tin cá nhân sau: mã định danh, họ và tên, mật khẩu, số điện thoại, địa chỉ, ngày tháng năm sinh
  - Các thông tin liên quan đến sức khỏe bao gồm: chỉ số nhịp tim hiện tại (BPM), chỉ số nhip tim trung bình (Avg_BPM), chỉ số hồng ngoại (IR_value), dữ liệu gia tốc theo 3 trục (Accel_X, Accel_Y, Accel_Z), độ lớn gia tốc tổng hợp (A_total)
  - 1 bệnh nhân được theo dõi bởi 1 bác sĩ
  - Mỗi bệnh nhân được gắn một thiết bị theo dõi sức khỏe để thu thập dữ liệu cảm biến và gửi về hệ thống theo thời gian thực

- Bác sĩ

  - Có các thông tin cá nhân sau: mã định danh, họ và tên, mật khẩu, số điện thoại, địa chỉ, email, ngày tháng năm sinh, chức danh, khoa chuyên môn
  - Một bác sĩ có thể theo dõi nhiều bệnh nhân.
  - Mỗi bác sĩ thuộc về một người quản lý.

- Quản lý

  - Có các thông tin cá nhân sau: mã định danh, họ và tên, mật khẩu, số điện thoại, địa chỉ, chức danh, email, ngày tháng năm sinh
  - Một quản lý có thể quản lý nhiều bác sĩ và nhiều bệnh nhân.
  - Quản lý có quyền cao nhất trong hệ thống, có thể xem, thống kê, phân quyền người dùng và cấu hình hệ thống.

- Mối quan hệ giữa các đối tượng:

  - Bác sĩ - bệnh nhân có mối quan hệ 1 - N: Một bác sĩ có thể theo dõi nhiều bệnh nhân, nhưng mỗi bệnh nhân chỉ được theo dõi bởi một bác sĩ.

- Quản lý - bác sĩ có mối quan hệ 1 - N: Một quản lý có thể quản lý nhiều bác sĩ, mỗi bác sĩ chỉ thuộc về một quản lý
- Quản lý - bệnh nhân có mối quan hệ 1 - N: Một quản lý có thể giám sát nhiều bệnh nhân, mỗi bệnh nhân chỉ thuộc về một quản lý.

- Mô tả mối quan hệ giữa thiết bị và các tác nhân

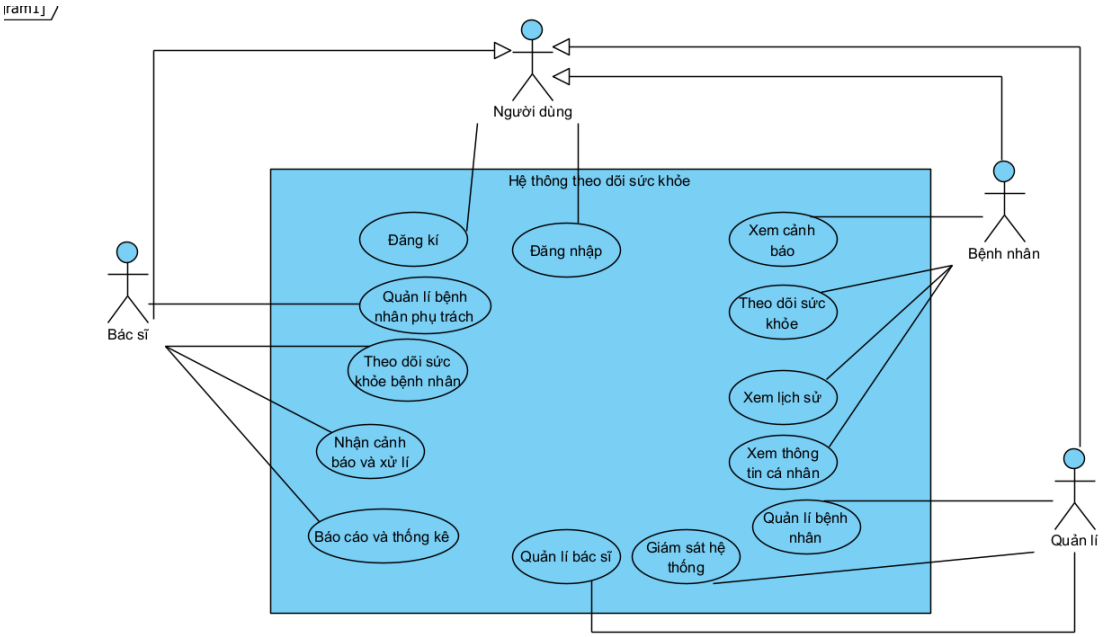
- Mỗi bệnh nhân gắn với 1 thiết bị IoT ESP32, tích hợp cảm biến MAX30102 (đo nhịp tim, IR) và ADXL345 (đo gia tốc).
- Thiết bị gửi dữ liệu cảm biến qua MQTT Broker đến Flask Server.
- Flask lưu dữ liệu vào bảng thông tin sức khỏe tương ứng với bệnh nhân
- Mô hình AI trên server phân tích và ghi kết quả cảnh báo nếu có dấu hiệu bất thường.
- Bác sĩ có thể xem dữ liệu và cảnh báo của tất cả bệnh nhân mà mình phụ trách.
- Quản lý có thể truy cập toàn bộ dữ liệu hệ thống, thống kê, và phân quyền người dùng.

II. Yêu cầu chức năng

1. Yêu cầu chức năng của hệ thống

- Hệ thống phải đảm bảo các yêu cầu:

- Theo dõi và cập nhật liên tục dữ liệu cảm biến.
- Lưu trữ lịch sử dữ liệu sức khỏe của từng bệnh nhân.
- Phát hiện và cảnh báo kịp thời khi có dấu hiệu bất thường.
- Quản lý người dùng (bệnh nhân, bác sĩ, quản lý) với quyền truy cập phù hợp.
- Thống kê, báo cáo và cấu hình hệ thống linh hoạt.



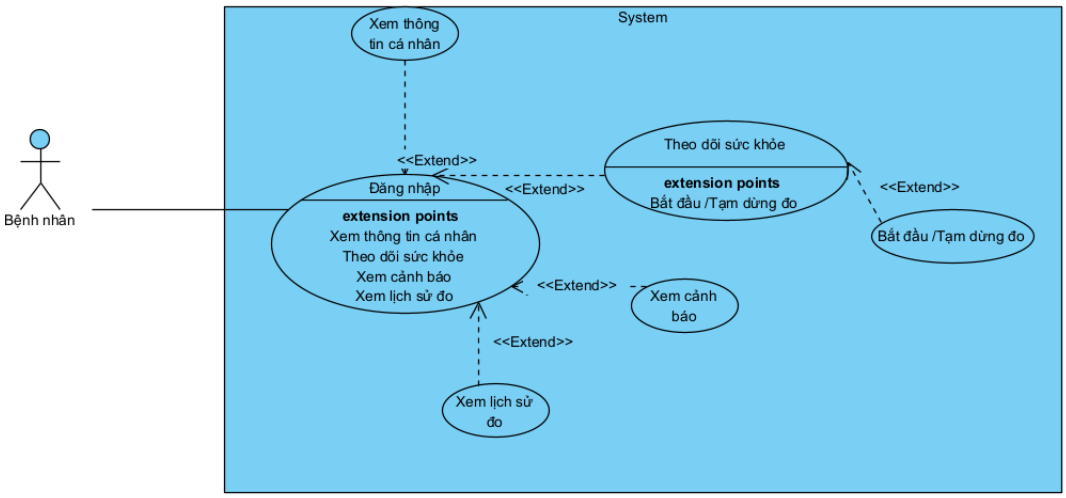
Hình ảnh 4: Sơ đồ use case của hệ thống

2. Chức năng của bệnh nhân

Nhóm chức năng	Mô tả chi tiết
Đăng nhập, đăng ký	- Đăng nhập vào hệ thống bằng mã định danh và mật khẩu. - Thay đổi mật khẩu cá nhân.
Xem thông tin cá nhân	- Hiển thị thông tin cơ bản (họ tên, ngày sinh, số điện thoại, địa chỉ).
Theo dõi sức khỏe	- Xem các chỉ số sức khỏe theo thời gian thực: nhịp tim (BPM), IR_value, gia tốc (Accel_X, Y, Z, A_total). - Xem biểu đồ thống kê theo ngày, tuần, tháng.
Bắt đầu/ tạm dừng đo	- Bệnh nhân có thể bật chế độ đo hoặc tạm dừng đo tùy ý
Xem cảnh báo	- Nhận thông báo khi có bất thường (nhịp tim cao/thấp, chuyển động đột ngột).

Xem lịch sử	- Xem lịch sử đo nhịp tim, chuyển động, và các cảnh báo trước đây.

- Sơ đồ UseCase

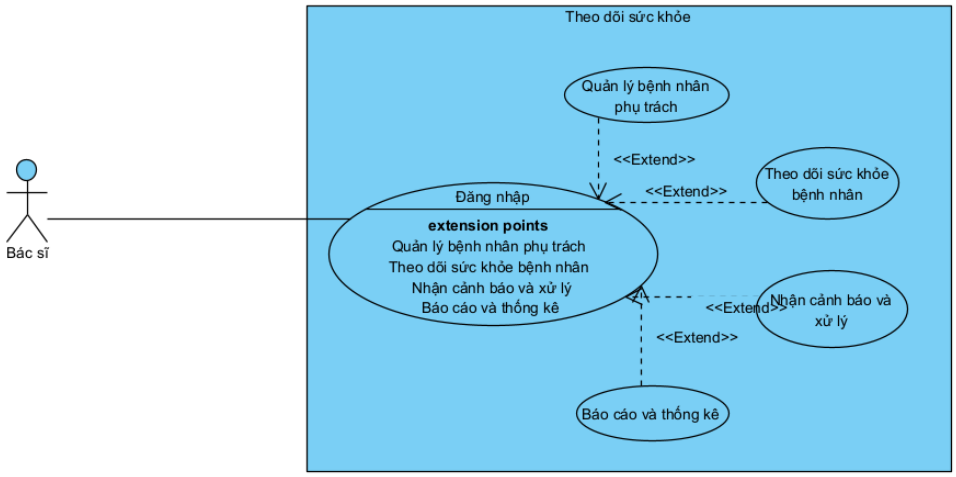


3. Chức năng của bác sĩ

Nhóm chức năng	Mô tả chi tiết
Đăng nhập	- Đăng nhập, thay đổi thông tin, đổi mật khẩu.
Quản lý bệnh nhân phụ trách	- Xem danh sách bệnh nhân phụ trách. - Thêm hoặc xác nhận bệnh nhân mới. - Xem hồ sơ cá nhân và dữ liệu sức khỏe của từng bệnh nhân.
Theo dõi sức khỏe bệnh nhân	- Xem dữ liệu thời gian thực (BPM, IR, gia tốc). - Xem biểu đồ biến thiên nhịp tim, vận động. - Phân tích xu hướng sức khỏe.

Nhận cảnh báo và xử lý	- Nhận thông báo khi bệnh nhân có dấu hiệu bất thường. - Ghi nhận và xác nhận đã xem cảnh báo. - Gửi hướng dẫn xử lý hoặc khuyến nghị cho bệnh nhân.
Báo cáo và thống kê	- Xem thống kê sức khỏe trung bình theo bệnh nhân hoặc nhóm bệnh nhân. - Xuất báo cáo sức khỏe định kỳ.

- Sơ đồ Usecase:

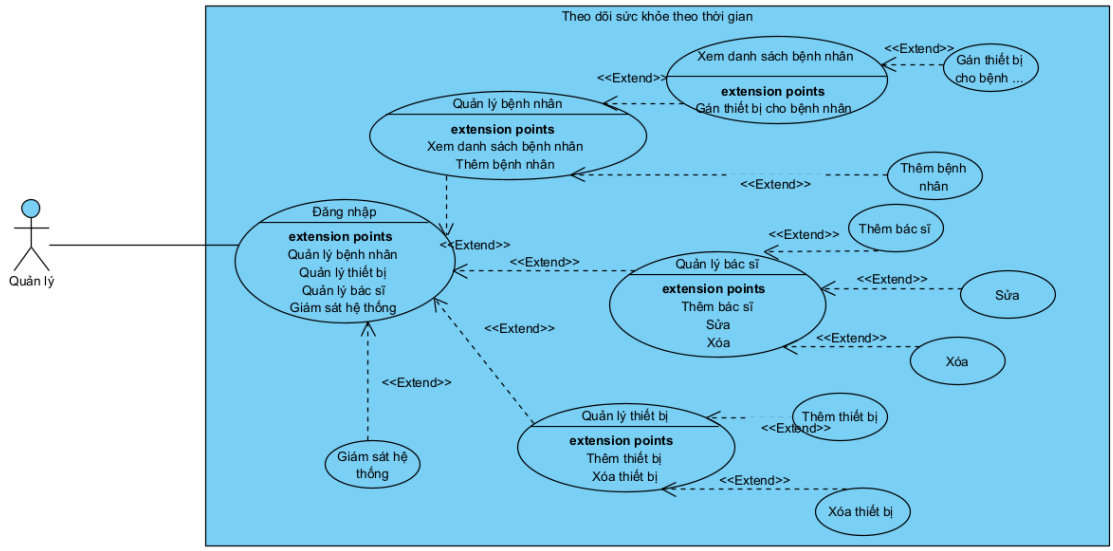


4. Chức năng của quản lý

Nhóm chức năng	Mô tả chi tiết
Đăng nhập	- Đăng nhập bằng tài khoản quản lý.
Quản lý bác sĩ	- Thêm, sửa, xóa thông tin bác sĩ. - Phân công bác sĩ phụ trách bệnh nhân.
Quản lý bệnh nhân	- Theo dõi danh sách toàn bộ bệnh nhân. - Gán bệnh nhân cho bác sĩ cụ thể. - Xem dữ liệu sức khỏe và cảnh báo của mọi bệnh nhân.

Giám sát hệ thống	- Theo dõi tình trạng hoạt động của các thiết bị IoT. - Kiểm tra kết nối MQTT, tình trạng gửi dữ liệu.
Thống kê – Báo cáo tổng hợp	- Thống kê số lượng bệnh nhân, bác sĩ, cảnh báo theo thời gian. - Xem biểu đồ sức khỏe tổng thể hệ thống. - Xuất báo cáo theo ngày, tuần, tháng.

- Sơ đồ Usecase



III. Yêu cầu phi chức năng

1. Yêu cầu về hiệu suất

- Đây là các yêu cầu về tốc độ và khả năng phản hồi của hệ thống. Đối với một hệ thống theo dõi sức khỏe, hiệu suất là yếu tố cực kỳ quan trọng.
- Độ trễ truyền dữ liệu: Dữ liệu nhịp tim từ cảm biến phải được gửi đến server qua MQTT broker trong vòng dưới 1 giây. Điều này đảm bảo dữ liệu được cập nhật gần như thời gian thực.
- Thời gian xử lý cảnh báo: Khi phát hiện nhịp tim bất thường, hệ thống phải gửi cảnh báo đến người dùng (qua giao diện web hoặc thông báo) trong vòng tối đa 2 giây.

- Tần suất gửi dữ liệu: Thiết bị cảm biến cần gửi dữ liệu nhịp tim đến server với tần suất đều đặn, mỗi 5 giây một lần, để đảm bảo theo dõi liên tục mà không gây quá tải cho mạng.

2. Yêu cầu về bảo mật

- Mã hóa dữ liệu: Toàn bộ dữ liệu truyền trên mạng giữa thiết bị, MQTT broker và server phải được mã hóa bằng TLS/SSL để ngăn chặn việc nghe lén.
- Xác thực và phân quyền:

- Mỗi thiết bị phải có một danh tính (ví dụ: client ID, username/password hoặc chứng chỉ) duy nhất để xác thực với MQTT broker.
- Sử dụng cơ chế phân quyền (ACL - Access Control List) trên MQTT broker để đảm bảo thiết bị chỉ có thể publish và subscribe vào các topic được phép. Ví dụ, một thiết bị chỉ có thể gửi dữ liệu lên topic của chính nó và không thể nghe lén dữ liệu từ các thiết bị khác.

- Bảo mật phía server: Server ứng dụng phải có các biện pháp bảo vệ chống lại các cuộc tấn công phổ biến như SQL injection, XSS,...

3. Yêu cầu về độ tin cậy

- Độ sẵn sàng (Uptime): Hệ thống (bao gồm cả MQTT broker và server) phải đạt độ sẵn sàng là 99.9%. Điều này có nghĩa là hệ thống chỉ có thể "sập" trong một khoảng thời gian rất ngắn trong một năm.
- Khả năng phục hồi sau lỗi: Nếu thiết bị mất kết nối mạng, nó phải có khả năng tự động kết nối lại và gửi dữ liệu đã được lưu trữ tạm thời (nếu có) khi có kết nối trở lại.
- Đảm bảo gửi tin (Quality of Service - QoS): Khi sử dụng MQTT, nên xác định mức độ QoS:

- QoS 0 (At most once) là mức phù hợp với dữ liệu được gửi theo cơ chế “gửi một lần rồi thôi” mà không cần xác nhận từ phía server. Điều này giúp giảm độ trễ, tăng tốc độ truyền và tiết kiệm tài nguyên, đặc biệt phù hợp với các dữ liệu cảm biến gửi liên tục theo thời gian thực. Việc mất một vài gói tin không ảnh hưởng lớn đến quá trình giám sát tổng thể.

- QoS 2 (Exactly once): Nên được sử dụng cho các tin nhắn cảnh báo quan trọng để đảm bảo cảnh báo được gửi đi và nhận được chính xác một lần.

4. Yêu cầu về khả năng mở rộng

- Hệ thống cần có khả năng xử lý khi số lượng người dùng hoặc thiết bị tăng lên.
- Khả năng xử lý đồng thời: Hệ thống phải có khả năng xử lý dữ liệu từ ít nhất 100 thiết bị gửi tin đồng thời mà không làm giảm hiệu suất.
- Mở rộng MQTT Broker: MQTT broker được chọn phải có khả năng clustering (tạo cụm) để có thể mở rộng khi số lượng kết nối tăng lên.

5. Yêu cầu về tính khả dụng

- Giao diện trực quan: Giao diện web phải hiển thị biểu đồ nhịp tim rõ ràng, dễ đọc và các cảnh báo phải nổi bật.
- Thời gian học hỏi: Người dùng mới có thể hiểu và sử dụng các chức năng chính trong vòng 5 phút.

6. Yêu cầu về khả năng bảo trì

- Ghi log : Hệ thống phải ghi lại các sự kiện quan trọng để hỗ trợ gỡ lỗi.
- Mã nguồn rõ ràng: Mã nguồn cần được viết theo chuẩn và có comment giải thích.
- Cấu hình tách biệt: Các thông tin cấu hình nên được lưu trong file riêng thay vì viết cứng trong code.

7. Yêu cầu về chi phí và năng lượng

- Chi phí phần cứng: Chi phí để chế tạo một thiết bị cảm biến hoàn chỉnh (vi điều khiển, cảm biến, vỏ hộp,...) cần được giữ ở mức thấp, ví dụ dưới 500.000 VNĐ, để sản phẩm có thể tiếp cận được nhiều người dùng.
- Chi phí vận hành: Ưu tiên sử dụng các dịch vụ MQTT broker và hosting miễn phí hoặc chi phí thấp cho giai đoạn thử nghiệm và quy mô nhỏ.
- Tiêu thụ năng lượng: Thiết bị phải được tối ưu để tiêu thụ năng lượng ở mức thấp nhất, đảm bảo có thể hoạt động liên tục bằng pin trong ít nhất 24 giờ sau mỗi lần sạc đầy.

- Chế độ tiết kiệm năng lượng: Phần mềm trên thiết bị cần có các chế độ ngủ (sleep mode) thông minh, tự động giảm tần suất đo hoặc tắt các module không cần thiết khi người dùng không hoạt động để kéo dài thời gian sử dụng pin.

Chương 4: Phân tích thiết kế

I. Tổng quan thiết kế hệ thống

1. Giới thiệu về cấu trúc tổng thể hệ thống.

- Hệ thống được xây dựng theo mô hình IoT tiêu chuẩn, bao gồm các thành phần chính:
    - Cảm biến: thu thập thông tin sinh học (nhịp tim) và thông tin chuyển động của người dùng.
    - Vi điều khiển ESP32: xử lý sơ bộ tín hiệu cảm biến, kết nối WiFi, truyền dữ liệu lên MQTT Broker, đồng thời nhận các tín hiệu điều khiển từ ứng dụng.
    - MQTT Broker (HiveMQ): đóng vai trò trung gian truyền thông, nhận dữ liệu từ ESP32 và phân phối đến web client, đồng thời chuyển lệnh điều khiển từ web xuống lại cho ESP32.
    - Hệ thống Web + Mô hình AI: nhận dữ liệu, phân tích bất thường, hiển thị kết quả, lưu dữ liệu người dùng, và gửi tín hiệu cảnh báo khi cần.
- Kiến trúc này đảm bảo tính linh hoạt, mở rộng, phù hợp cho thời gian thực, và đặc biệt tối ưu cho các ứng dụng IoT yêu cầu truyền thông nhẹ.

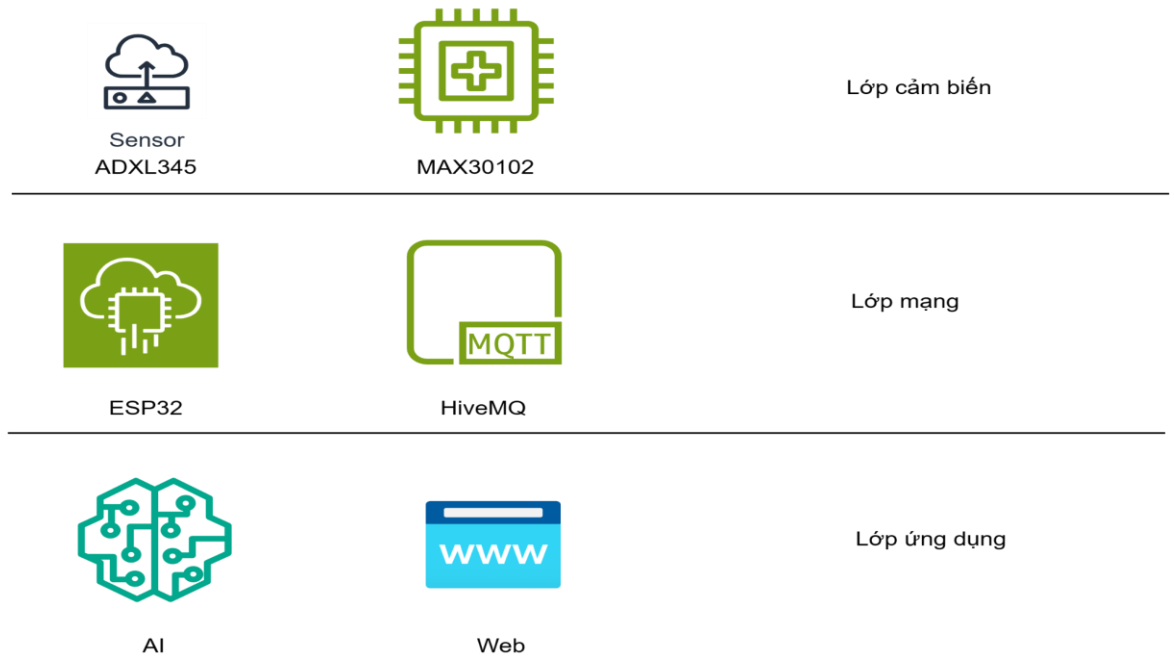
2. Mô hình 3 lớp IoT

- Hệ thống tuân theo mô hình IoT ba lớp tiêu chuẩn:

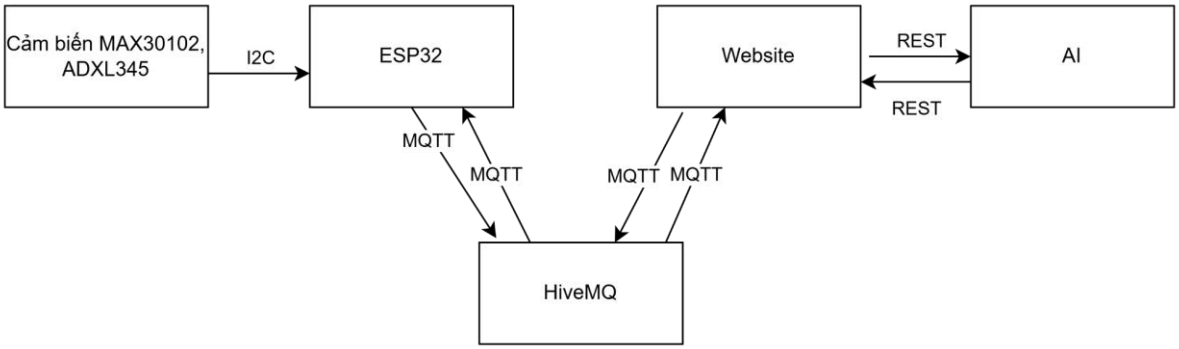
+ Lớp cảm biến (Sensing Layer)
    - Bao gồm cảm biến nhịp tim, cảm biến chuyển động.
    - Thu thập dữ liệu thực tế từ người dùng.
    - Truyền dữ liệu dạng analog/digital vào ESP32 để xử lý.

+ Lớp mạng (Network Layer)
    - ESP32 thực hiện:
        - Kết nối WiFi
        - Gửi và nhận dữ liệu qua MQTT Broker (HiveMQ)
    - HiveMQ đảm nhận truyền thông hai chiều giữa ESP32 và Web.

+ Lớp ứng dụng (Application Layer)
    - Giao diện web hiển thị dữ liệu theo thời gian thực.
    - Mô hình AI phân tích tín hiệu và đưa ra cảnh báo.
    - Web gửi tín hiệu điều khiển bật/tắt đèn LED cho ESP32 khi có bất thường.



3. Sơ đồ khối hệ thống (Block Diagram)



- Luồng hoạt động tổng thể: từ thu thập dữ liệu đến hiển thị kết quả: Quá trình vận hành của hệ thống diễn ra theo các bước:

- Bước 1: Cảm biến đo nhịp tim và chuyển động rồi gửi tín hiệu đến ESP32.
- Bước 2: ESP32 xử lý sơ bộ dữ liệu, đóng gói dưới dạng JSON và publish lên HiveMQ thông qua topic đã quy định.
- Bước 3: Web Application subscribe topic và nhận dữ liệu thời gian thực.
- Bước 4: Web truyền dữ liệu sang mô hình AI để phân tích bất thường.
- Bước 5: Mô hình AI trả kết quả lại cho Web, bao gồm đánh giá mức độ nguy hiểm.
- Bước 6: Web hiển thị kết quả và lưu vào cơ sở dữ liệu.
- Bước 7: Nếu phát hiện dấu hiệu bất thường, Web publish cảnh báo qua MQTT.
- Bước 8: ESP32 nhận tín hiệu cảnh báo và bật đèn LED cảnh báo cho người dùng.

- Luồng truyền thông đảm bảo tính ổn định, tức thời, hai chiều, phục vụ hoàn chỉnh cho bài toán giám sát sức khỏe từ xa.

II. Thiết kế phần cứng

1. Chi tiết các linh kiện

- Vi điều khiển esp32 CH340 30P Wifi + Bluetooth type C
    - Chip xử lý ESP32 (Xtensa Dual-Core 32-bit LX6, tốc độ lên đến 240MHz)
    - Kết nối WiFi và Bluetooth (Classic + BLE 4.2) – đáp ứng đa dạng nhu cầu giao tiếp không dây.
    - Cổng giao tiếp Type-C – dễ dàng kết nối và cấp nguồn, tương thích với hầu hết máy tính hiện nay.
    - Tích hợp chip nạp CH340 – driver phổ biến, dễ cài đặt, tương thích tốt với Windows/Linux/Mac.
    - 30 chân I/O – hỗ trợ giao tiếp SPI, I2C, UART, ADC, DAC, PWM… phù hợp nhiều loại cảm biến và module.

- Cảm biến MAX30102
    - Module MAX30102 là cảm biến đo nhịp tim (Heart Rate) và nồng độ oxy trong máu (SpO_2) chất lượng cao, được sản xuất bởi Maxim Integrated.
    - Module tích hợp LED hồng ngoại (IR), LED đỏ, cùng bộ thu quang (photodiode) và mạch xử lý tín hiệu chuyên dụng, giúp đọc chính xác xung mạch máu dưới da.
    - Cảm biến hoạt động dựa trên nguyên lý đo hấp thụ ánh sáng của máu theo từng nhịp tim, từ đó tính toán tần số nhịp tim và độ bão hòa oxy trong máu (SpO_2).
    - Với kích thước nhỏ gọn, độ nhạy cao, và giao tiếp I2C, MAX30102 được sử dụng rộng rãi trong thiết bị đeo tay, vòng theo dõi sức khỏe, smartwatch, và các dự án IoT y tế.
    - Sơ đồ chân: Cảm biến có 5 chân chính:
        - VIN: Nguồn cấp 3.3V hoặc 5V.
        - GND: Nối mass.
        - SCL: Chân xung giao tiếp I2C.
        - SDA: Chân dữ liệu I2C.
        - INT: Chân ngắt (tùy chọn, có thể không dùng).

- Cảm biến ADXL345
    - Cảm biến gia tốc góc ADXL345 là gia tốc kế ba trục nhỏ, mỏng, công suất thấp, cung cấp các phép đo gia tốc có độ phân giải cao (13 bit) lên tới ±16 g. Dữ liệu đầu ra kỹ thuật số ở định dạng bổ sung hai bit 16 bit và có thể được truy cập thông qua giao diện kỹ thuật số SPI (3 dây hoặc 4 dây) hoặc I2C.
    - Chức năng các chân
        - GND: Chân cấp nguồn 0V
        - VCC : chân nguồn dương 3V – 6V
        - CS : Lựa chọn chip
        - INT1 : Chân ngắt 1
        - INT2 : Chân ngắt 2

    - SDO : Ngõ ra dữ liệu
    - SDA : Chân dữ liệu I2C
  - SCL : Chân tạo xung I2C

- Dây cáp nạp type C
  - Dây cáp nạp type C 30cm là cáp kết nối có một đầu đực của USB 2.0 và một đầu đực type C. Dây cáp USB này dùng để kết nối esp với máy tính. Được bọc kỹ lưỡng giúp chống nhiễu được tốt hơn. Sử dụng trong vấn đề nạp chương trình và sử dụng truyền nhận dữ liệu giữa các thiết bị như máy ảnh, điện thoại,…
  - Thông số cơ bản dây cáp : dài 30cm, dây dẫn làm bằng đồng, kết nối từ USB sang Type C.

- Bảng mạch Breadboard MB-102 830 lỗ 165x55x10mm
  - Breadboard MB-102 830 lỗ 165x55x10mm được sử dụng để gắn các module hoặc linh kiện điện tử, kết nối chúng với nhau bằng các loại dây cắm, dây nối test board giúp test, kiểm tra tính năng 1 cách dễ dàng trước khi tạo thành các thành phẩm hoàn chỉnh.
  - Cấu tạo của breadboard rất đơn giản:
    - Khu vực trung tâm chính của breadboard là một khối gồm hai cột.
    - Mỗi cột được tạo thành từ nhiều hàng.
    - Mỗi hàng được nối điện theo từng hàng.
    - Dọc hai bên là hai bus dọc để cấp điện vào cột bên trong.

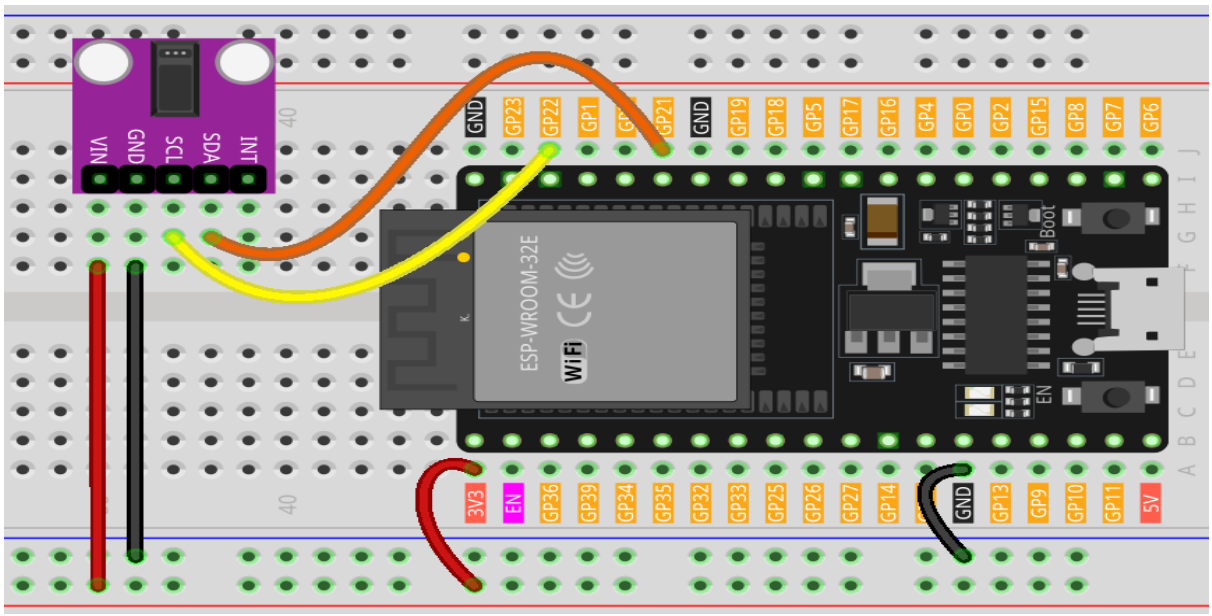
- Dây nối đực - đực
  - Dây Cắm Breadboard đực đực (M-M Jumper Wire) được sử dụng với Breadboard để kết nối các module và linh kiện điện tử với nhau, dây có chất lượng tốt với lõi đồng giúp dây mềm, khó đứt và không bị oxy hóa so với các loại chất lượng kém giá rẻ lõi nhôm trên thị trường (dây cứng, cắt ra thấy bị oxy hóa và rất dễ đứt).
  - Thông số kỹ thuật: Dây lõi đồng nhiều sợi có độ dẫn điện cao, mối tiếp xúc chắc chắn, độ dài 20cm, có nhiều màu sắc khác nhau.

2. Sơ đồ chân (Pin Connection Table)

- Bảng nối chân giữa esp32 với max30102

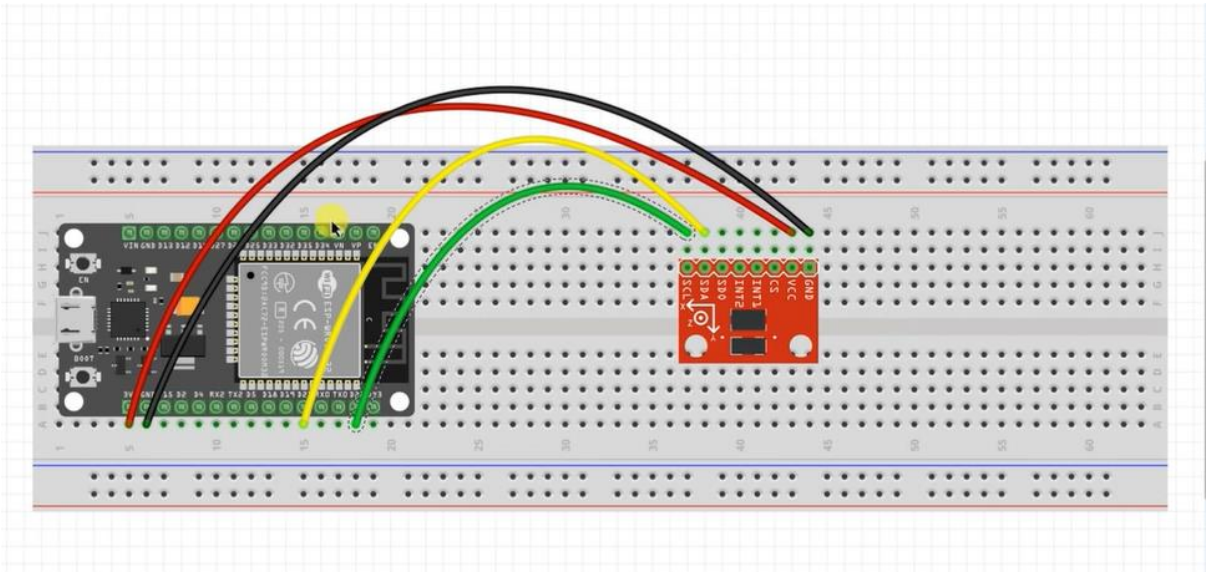
Chân của esp32	Chân của max30102
3V3	VIN
GND	GND

GPIO 21	SDA
GPIO 22	SCL

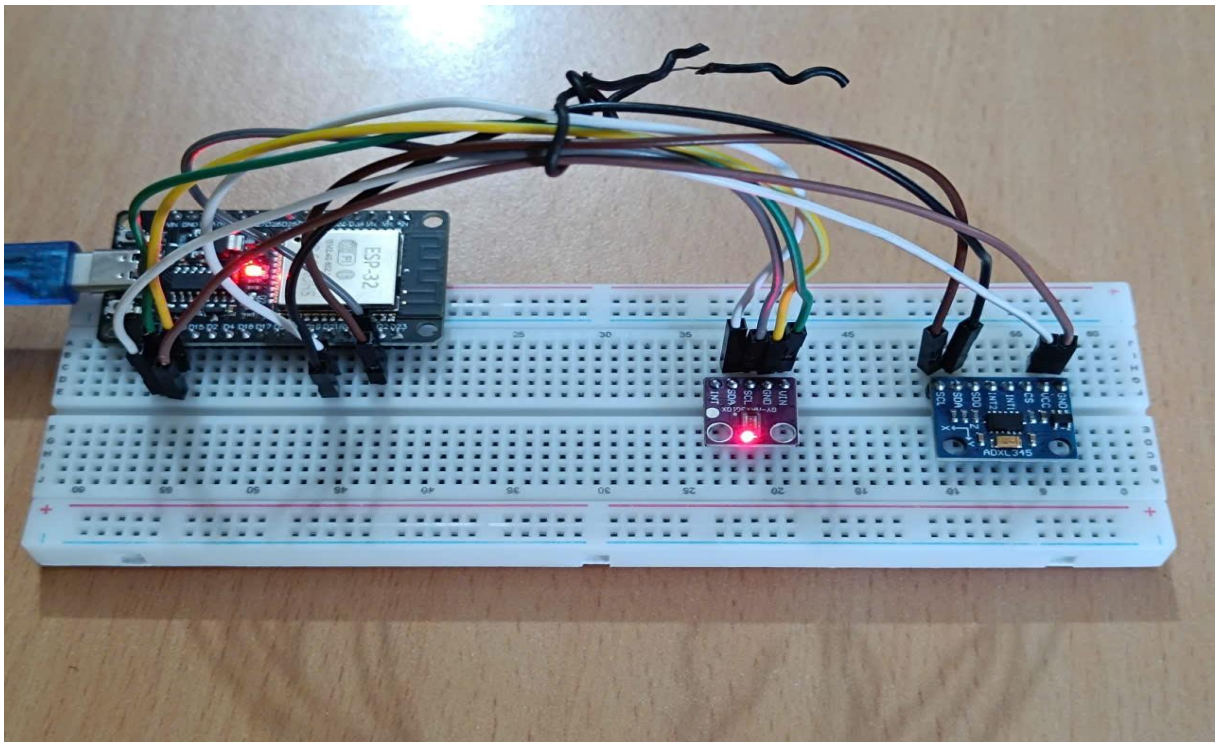


- Bảng nối chân giữa esp32 với adxl345

Chân của esp32	Chân của adxl345
3V3	VCC
GND	GND
GPIO 21	SDA
GPIO 22	SCL



- Thực hiện việc nối chân cho 2 cảm biến, ta thu được hình ảnh thực tế



3. Nguyên lý hoạt động phần cứng:

a. Tổng quan hệ thống

- Hệ thống phần cứng gồm ba khối chính:
  - Khối cảm biến – gồm cảm biến nhịp tim MAX30102 và cảm biến gia tốc ADXL345.
  - Khối xử lý trung tâm – vi điều khiển ESP32 đảm nhiệm việc đọc dữ liệu, xử lý và truyền đi.
  - Khối truyền thông & hiển thị – ESP32 kết nối WiFi gửi dữ liệu đến máy chủ thông qua giao thức MQTT

- Tất cả các linh kiện được cấp nguồn 3.3V từ ESP32 và giao tiếp thông qua bus I2C (chung hai dây SDA – SCL).

b. Nguyên lý hoạt động của từng khối

- Vi điều khiển ESP32
    - ESP32 là vi điều khiển tích hợp WiFi và Bluetooth, đóng vai trò trung tâm của hệ thống.
    - Nó đọc dữ liệu cảm biến thông qua giao tiếp I2C, xử lý tín hiệu cơ bản (lọc nhiễu, tính toán nhịp tim, gia tốc), rồi truyền dữ liệu lên website thông qua giao thức MQTT
    - Ngoài ra, ESP32 có thể thực hiện cảnh báo trực tiếp bật LED khi có những cảnh báo bất thường

- Cảm biến nhịp tim MAX30102
    - MAX30102 là cảm biến quang học dùng phương pháp đo quang thể tích (PPG – Photoplethysmography).
    - Module có đèn LED đỏ và hồng ngoại chiếu ánh sáng qua đầu ngón tay; điốt quang thu lại lượng ánh sáng phản xạ từ mạch máu.
    - Khi tim co bóp, thể tích máu trong mao mạch thay đổi → cường độ ánh sáng phản xạ thay đổi → cảm biến thu nhận tín hiệu này.
    - ESP32 đọc tín hiệu ánh sáng qua giao tiếp I2C, xử lý bằng thuật toán để xác định số nhịp tim (BPM – Beats Per Minute).
    - Tín hiệu đầu ra dạng sóng PPG được hiển thị trên website

- Cảm biến chuyển động ADXL345
    - ADXL345 là cảm biến gia tốc 3 trục (X, Y, Z).
    - Dựa trên sự thay đổi điện dung trong cấu trúc MEMS bên trong, ADXL345 đo được gia tốc và hướng chuyển động.
    - Khi người đeo di chuyển (lắc tay, té ngã, hoạt động mạnh...), giá trị gia tốc thay đổi → ESP32 nhận dạng được trạng thái chuyển động.

- Liên kết giữa các khối
    - Cả hai cảm biến được kết nối song song trên bus I2C (SDA – SCL).
    - ESP32 liên tục gửi lệnh đọc dữ liệu từ các địa chỉ I2C khác nhau:
        - MAX30102: địa chỉ 0x57
        - ADXL345: địa chỉ 0x53
    - Các cảm biến truyền dữ liệu về ESP32, sau đó vi điều khiển xử lý hoặc gửi trực tiếp lên server để hiển thị.

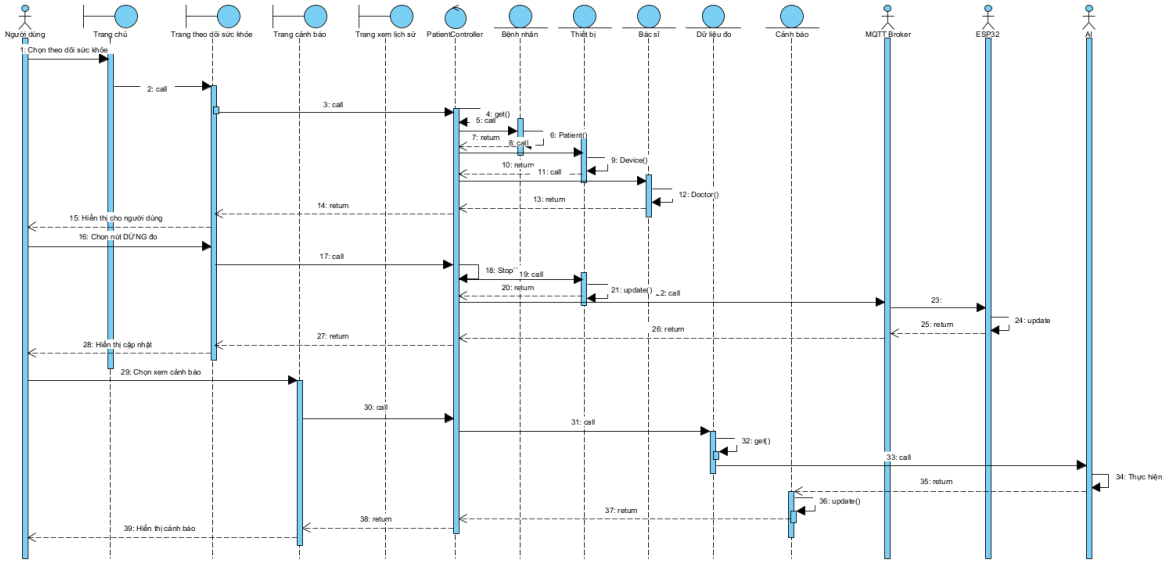
c. Nguyên lý hoạt động tổng thể

- Khi hệ thống khởi động, ESP32 khởi tạo kết nối I2C và WiFi.
- MAX30102 bắt đầu phát ánh sáng, đo tín hiệu phản xạ từ ngón tay → gửi dữ liệu dạng sóng quang đến ESP32.
- ADXL345 liên tục đo gia tốc → gửi dữ liệu ba trục X, Y, Z về ESP32.
- ESP32 tính toán nhịp tim, nhận diện chuyển động, lọc nhiễu, sau đó:
    - Gửi dữ liệu qua mqtt để hiển thị kết quả trên trang web
    - Gửi cảnh báo (ví dụ: nhịp tim cao, bất động lâu) nếu vượt ngưỡng.
- Tất cả dữ liệu được lưu vào cơ sở dữ liệu và hiển thị theo thời gian thực.

III. Thiết kế phần mềm

1. Luồng hoạt động của hệ thống

a. Luồng hoạt động của bệnh nhân

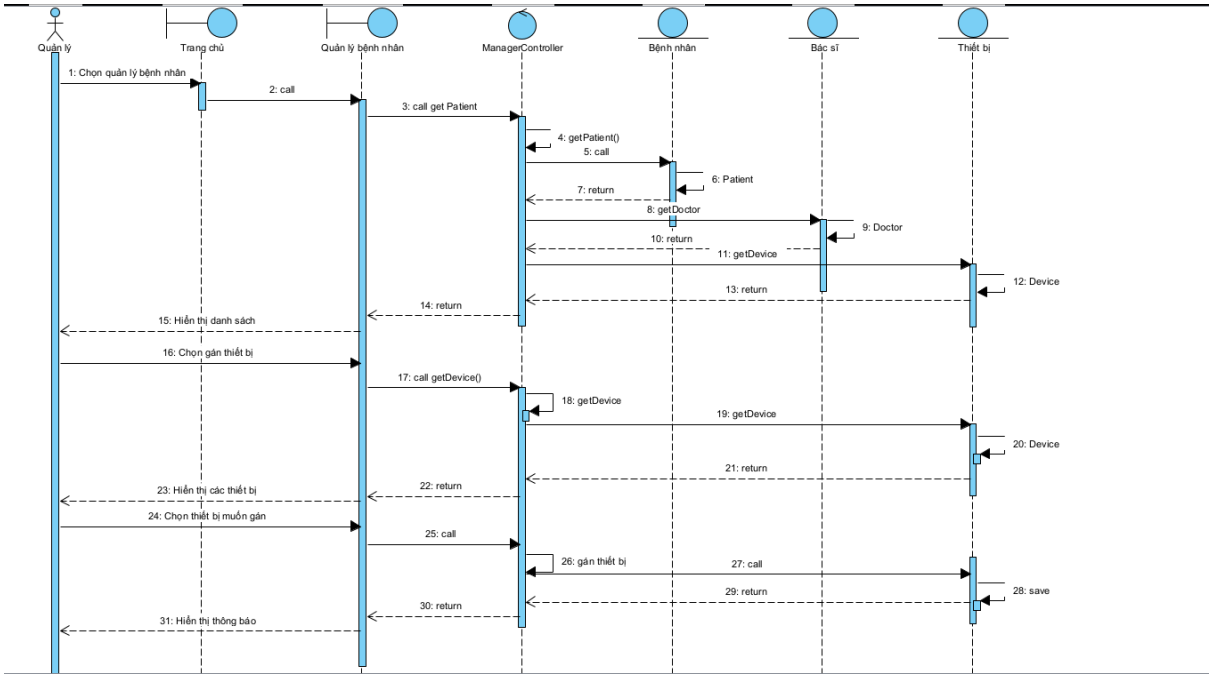


1. Tại giao diện Trang chủ người dùng chọn theo dõi sức khỏe
2. Trang chủ gọi đến Trang theo dõi sức khỏe
3. Trang theo dõi sức khỏe gọi đến PatientController
4. PatientController thực hiện gọi hàm get để lấy thông tin
5. Thực hiện truy vấn đến bảng cơ sở dữ liệu của bệnh nhân
6. Thực hiện đóng gói dữ liệu
7. Trả về cho PatientController
8. Truy vấn đến bảng cơ sở dữ liệu Thiết bị
9. Thực hiện đóng gói dữ liệu
10. Trả về cho PatientController
11. Truy vấn cơ sở dữ liệu đến bảng bác sĩ
12. Thực hiện đóng gói dữ liệu
13. Trả về cho PatientController
14. Thực hiện join các bảng và trả về cho trang theo dõi sức khỏe
15. Thực hiện hiển thị thông tin cho người dùng

16. Người dùng nhấn nút dừng đo
17. Giao diện gọi đến PatientController
18. Gọi hàm stop để thực hiện dừng đo
19. Truy vấn đến cơ sở dữ liệu của thiết bị
20. Thực hiện cập nhật CSDL
21. Thực hiện trả về PatientController
22. Gọi đến MQTT Broker
23. Gọi đến ESP32
24. ESP32 cập nhật trạng thái
25. Trả về cho MQTT Broker
26. Dữ liệu trả về cho PatientController
27. Trả về cho trang theo dõi sức khỏe
28. Hiển thị thông báo cho người dùng
29. Người dùng chọn xem cảnh báo
30. Trang cảnh báo gọi đến PatientController
31. Truy vấn đến cơ sở dữ liệu của Dữ liệu đo
32. Thực hiện đóng gói dữ liệu
33. Dữ liệu được đưa đến mô hình AI
34. Thực hiện đưa ra các cảnh báo
35. Truy vấn đến cơ sở dữ liệu bảng Cảnh báo
36. Thực hiện đóng gói dữ liệu
37. Trả về cho PatientController
38. Trả về cho trang hiển thị cảnh báo
39. Hiển thị giao diện cho người dùng.

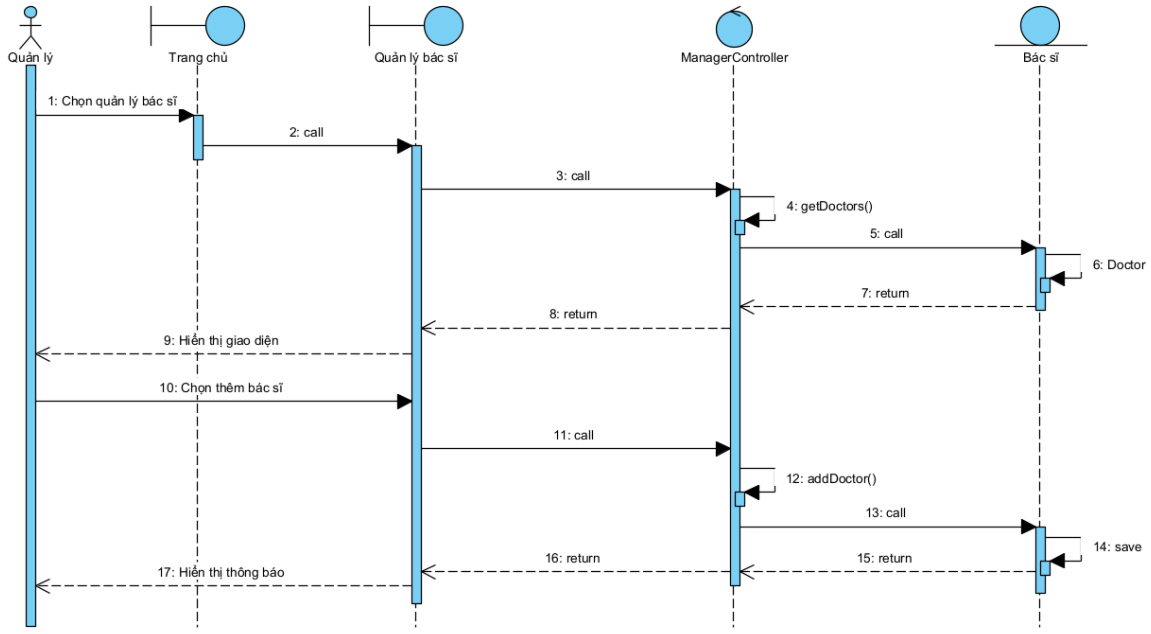
b. Luồng hoạt động của quản lý

- Quản lý bệnh nhân



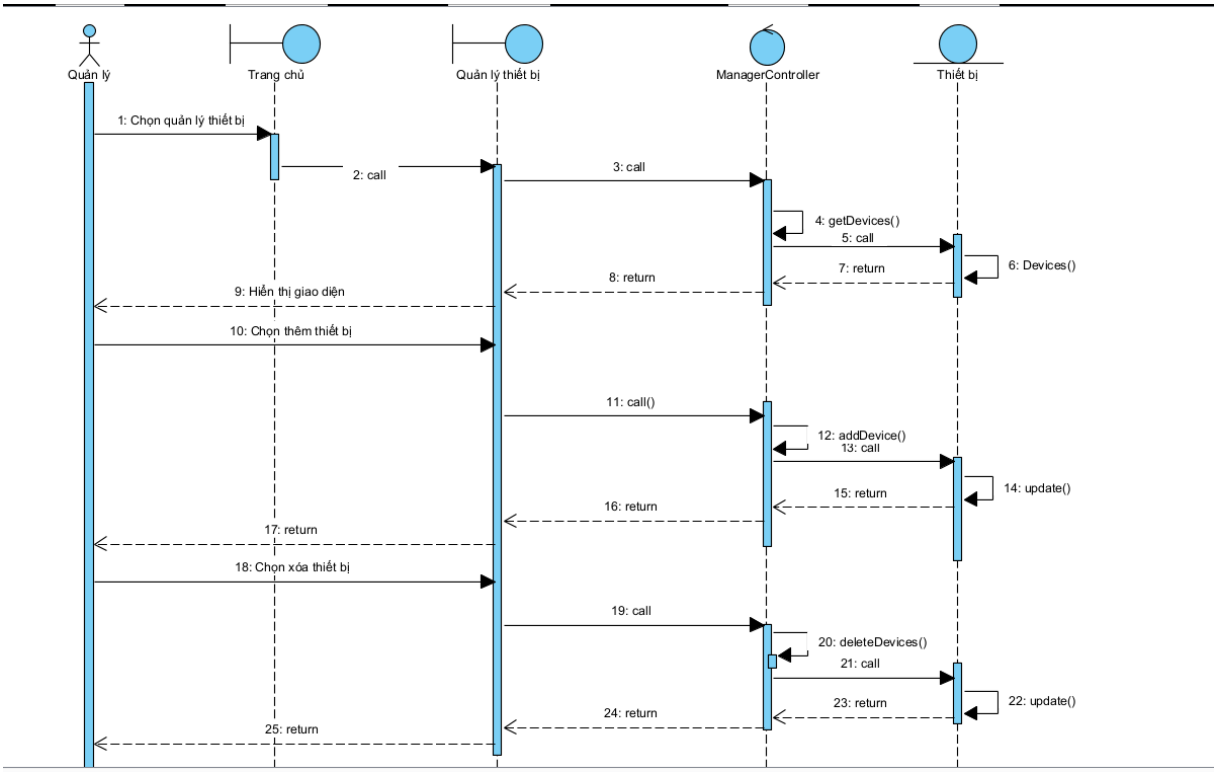
+Thực hiện các luồng chính để hiển thị danh sách bệnh nhân , và thực hiện gán thiết bị cho bệnh nhân.

- Quản lý bác sĩ



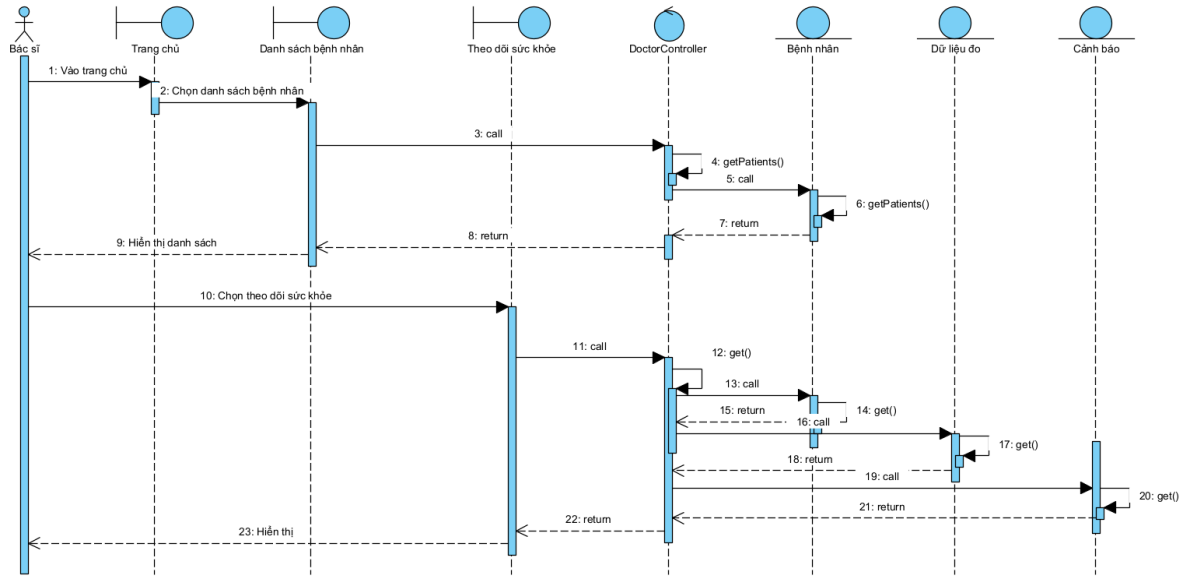
+Thực hiện các luồng chính về danh sách bác sĩ và thêm bác sĩ.

- Quản lý thiết bị



+ Thể hiện luồng chính là danh sách các thiết bị, thêm thiết bị, xóa thiết bị.

c. Luồng hoạt động của bác sĩ



+ Thực hiện xem danh sách các bệnh nhân mình quản lý và theo dõi sức khỏe của từng bệnh nhân.

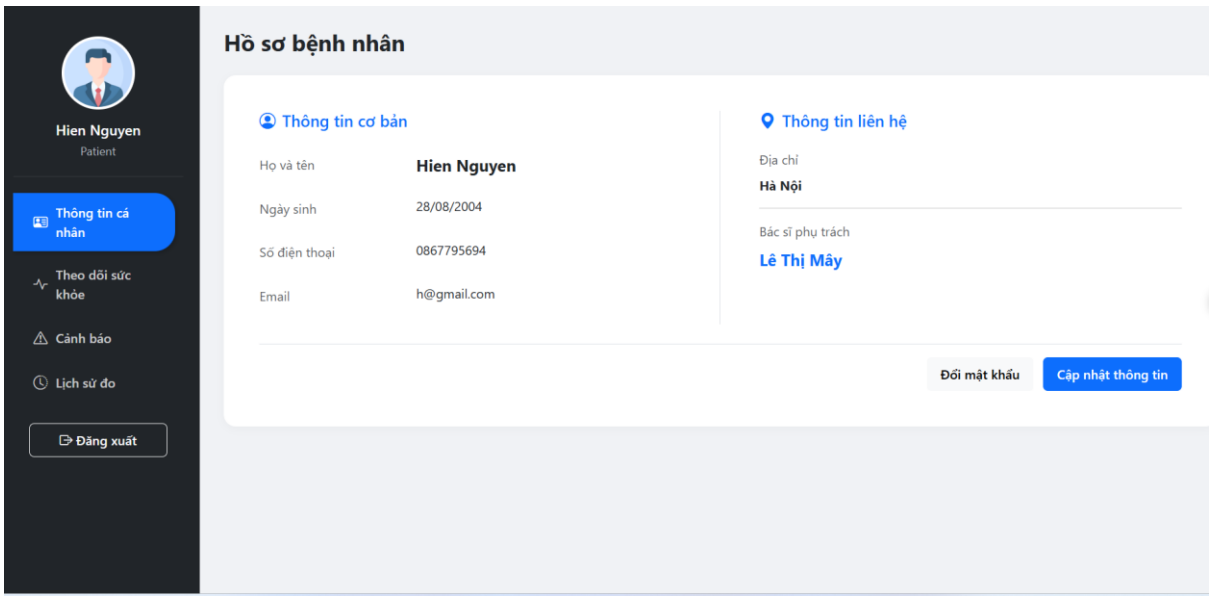
2. Thiết kế giao diện (UI Design):

a. Giao diện của bệnh nhân

- Các màn hình chính:

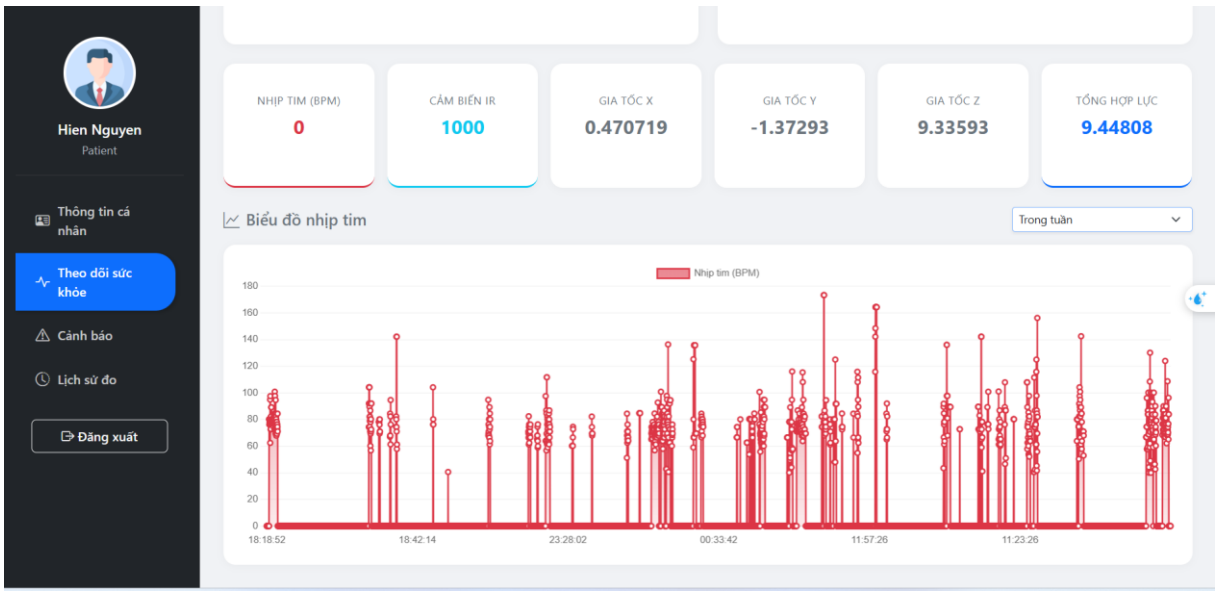
+ Trang chủ / Dashboard:

- Hiển thị thông tin cá nhân: tên, mã định danh, tuổi, giới tính.
- Biểu đồ dữ liệu sức khỏe theo thời gian: nhịp tim, huyết áp, nhiệt độ, oxy máu.



+ Lịch sử đo / Dữ liệu IoT:

- Bảng hoặc biểu đồ hiển thị dữ liệu các lần đo trước.
- Tùy chọn lọc theo ngày/tháng.



+ Thông báo / Cảnh báo:

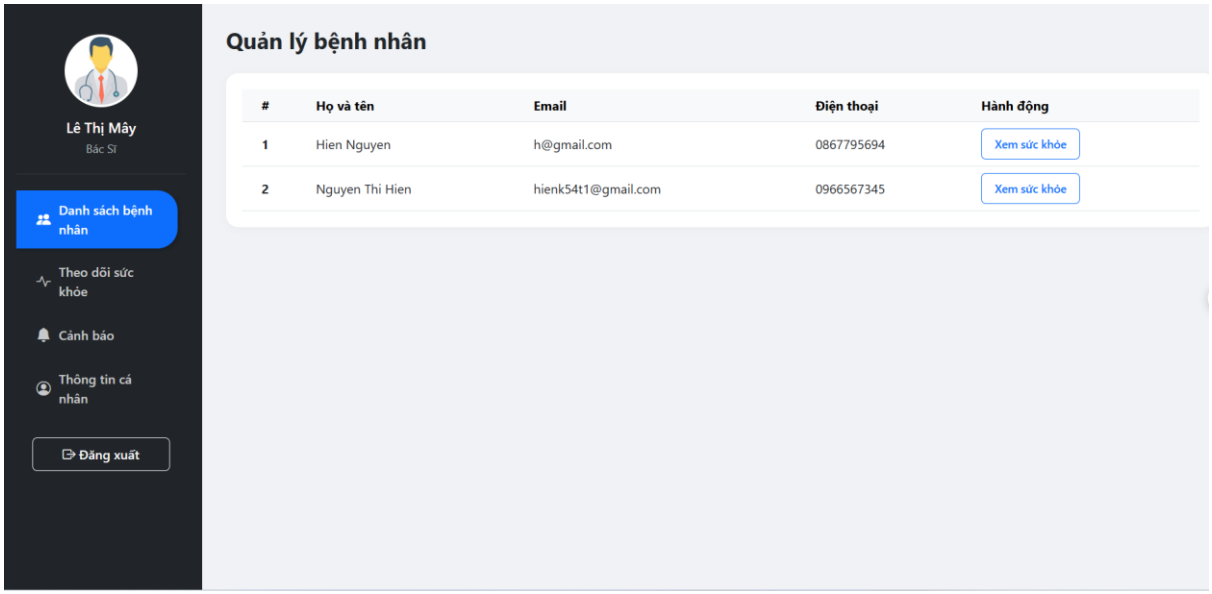
- Hiển thị cảnh báo nếu dữ liệu vượt ngưỡng.
- Ví dụ: “Nhịp tim cao trên 130 – cần kiểm tra y tế”.

b. Giao diện của bác sĩ

- Các màn hình chính:

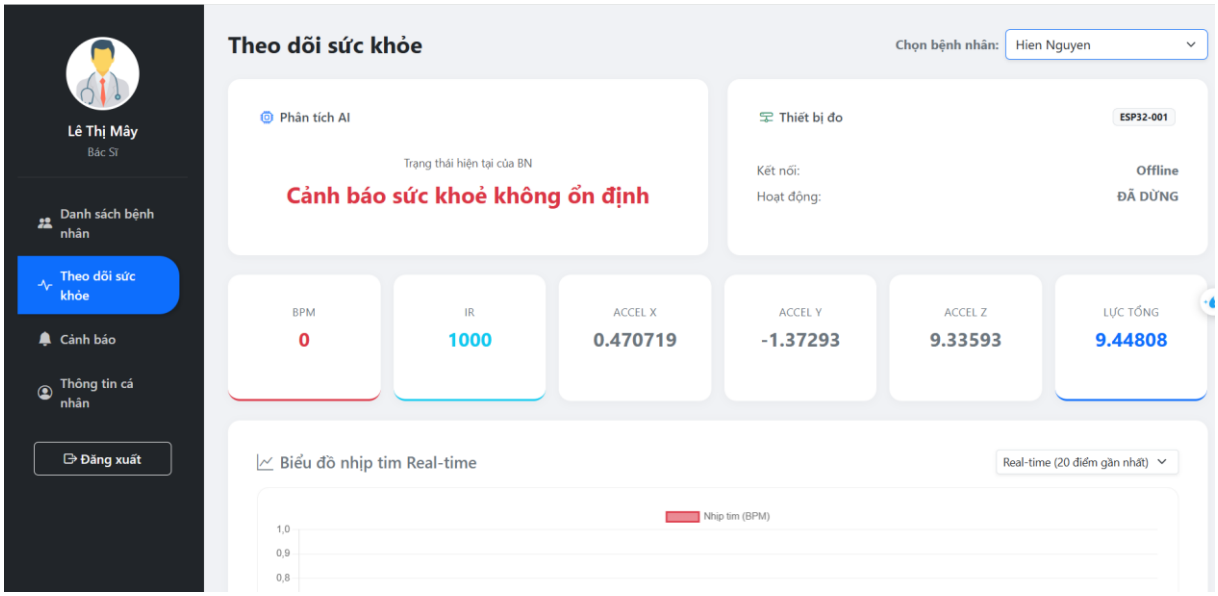
+ Dashboard tổng quan:

- Danh sách bệnh nhân được quản lý.
- Thống kê nhanh: số bệnh nhân bình thường / cảnh báo / nguy hiểm.



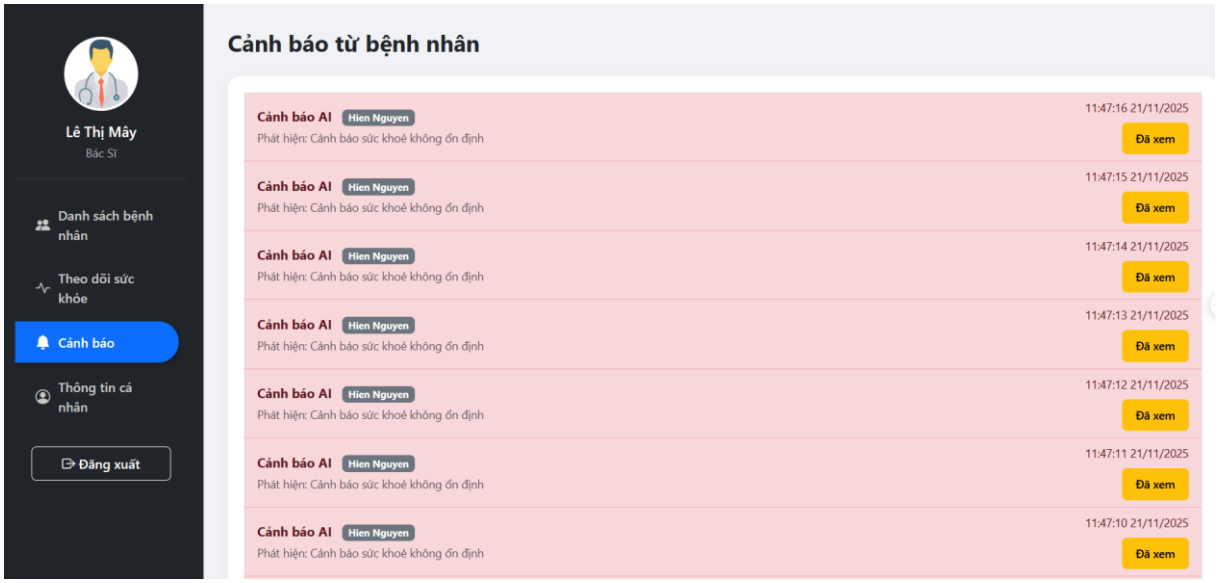
+ Chi tiết bệnh nhân:

- Xem biểu đồ dữ liệu sức khỏe theo thời gian.
- Nhận thông báo cảnh báo từ hệ thống.
- Ghi chú hoặc tư vấn cho bệnh nhân.



+ Quản lý cảnh báo:

- Xem cảnh báo sức khỏe của bệnh nhân

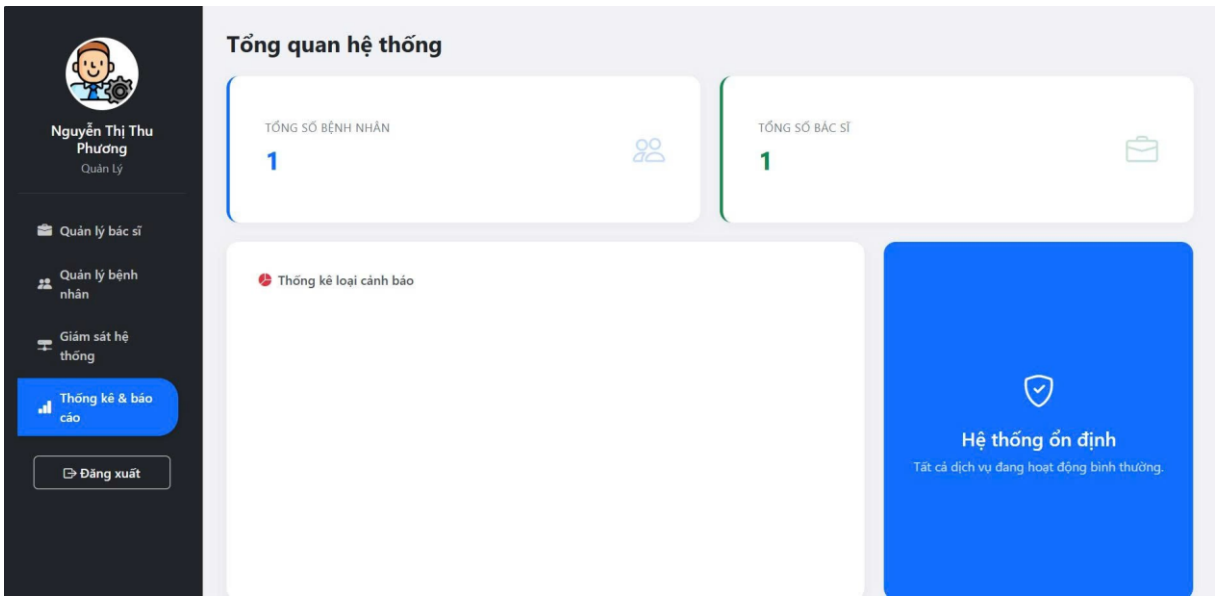


c. Giao diện của quản lý

- Các màn hình chính:

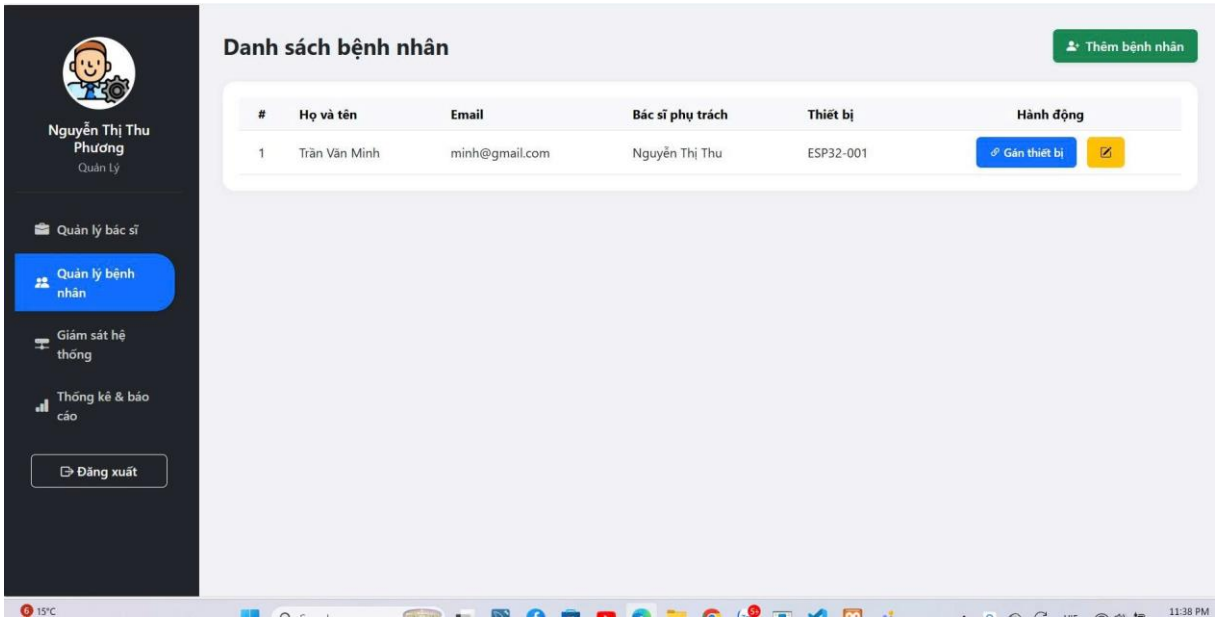
+ Dashboard tổng quan:

- Số lượng bệnh nhân, bác sĩ, thiết bị hoạt động.
- Biểu đồ thống kê số bệnh nhân theo trạng thái sức khỏe.



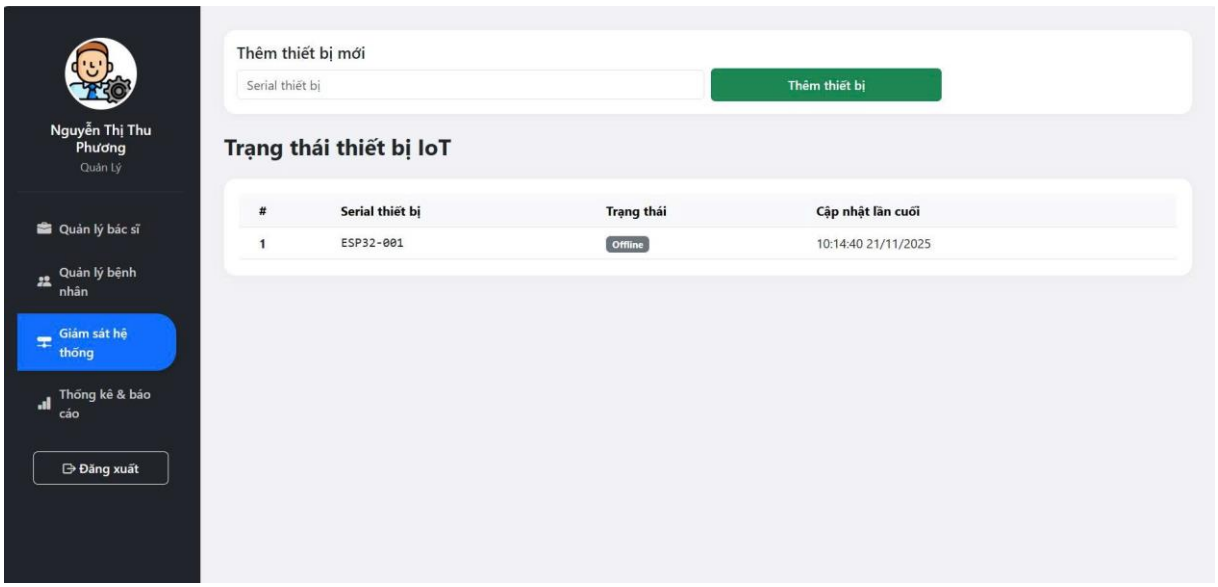
+ Quản lý người dùng:

- Danh sách bệnh nhân và bác sĩ.
- Thêm/sửa/xóa người dùng.



+ Quản lý thiết bị IoT:

- Danh sách thiết bị, trạng thái kết nối.
- Thêm/sửa/xóa thiết bị, gán thiết bị cho bệnh nhân.



+ Báo cáo & thống kê:

- Xuất dữ liệu, biểu đồ theo ngày/tháng.

3. Thiết kế CSDL

- Cơ sở dữ liệu bao gồm 6 bảng chính:

- Bảng 1 - Managers:

+ Lưu trữ thông tin về các quản lý

+ Các trường chính:

- manager_id – Khóa chính
- full_name – Họ tên quản lý.
- email – Địa chỉ email, duy nhất.
- password_hash – Mật khẩu
- phone_number – Số điện thoại.
- address – Địa chỉ.
- date_of_birth – Ngày sinh.
- title – Chức vụ.
- created_at – Thời điểm tạo tài khoản

- Bảng 2 - Doctors:

+ Lưu trữ thông tin bác sĩ và liên kết với quản lý phụ trách.

+ Các trường chính:

- doctor_id (Khóa chính)
- manager_id (Khóa ngoại tham chiếu đến Managers)
- full_name: Họ tên bác sĩ
- email (Duy nhất)
- password_hash

- title: Chức vụ
- specialty (chuyên khoa).
- created_at – Thời điểm tạo tài khoản

+ Quan hệ: Mỗi bác sĩ có thể được quản lý bởi một quản lý.

- Bảng 3 - Devices:

+ Quản lý các thiết bị theo dõi sức khỏe.

+ Các trường chính:

- device_id (Khóa chính)
- device_serial: Mã thiết bị (Duy nhất)
- status (trạng thái online/offline/error)
- last_seen: Lần cuối nhận tín hiệu
- created_at – Thời điểm tạo bản ghi.
- is_measuring – Thiết bị đang đo hay không (0/1).

- Bảng 4 - Patients:

+ Lưu trữ thông tin chi tiết của bệnh nhân.

+ Các trường chính:

- patient_id (Khóa chính)
- doctor_id (Khóa ngoại tham chiếu đến Doctors)
- manager_id (Khóa ngoại tham chiếu đến Managers)
- device_id (Khóa ngoại tham chiếu đến Devices, Duy nhất)
- full_name, email (Duy nhất)
- password_hash
- phone_number – Số điện thoại.
- email – Email duy nhất
- date_of_birth.
- address – Địa chỉ.
- created_at – Thời điểm tạo tài khoản
- current_health_status – Tình trạng sức khỏe hiện tại

+ Quan hệ: Mỗi bệnh nhân được theo dõi bởi một bác sĩ, thuộc một quản lý và được gán một thiết bị duy nhất.

- Bảng 5 - HealthData:

+ Lưu dữ liệu sức khỏe thô và dự đoán AI được gửi từ thiết bị.

+ Các trường chính:

- data_id (Khóa chính)
- patient_id (Khóa ngoại tham chiếu đến Patients)
- timestamp: Thời điểm ghi nhận data
- bpm : Nhịp tim tức thời
- avg_bpm: Nhịp tim trung bình

- ir_value : Giá trị hồng ngoại từ cảm biến MAX30102.
- accel_x, accel_y, accel_z: Gia tốc theo 3 trục từ ADXL345.
- a_total: Tổng gia tốc
- predict_status: Tình trạng sức khỏe được mô hình AI dự đoán

+ Quan hệ: Khi bệnh nhân bị xóa → dữ liệu sức khỏe cũng bị xóa

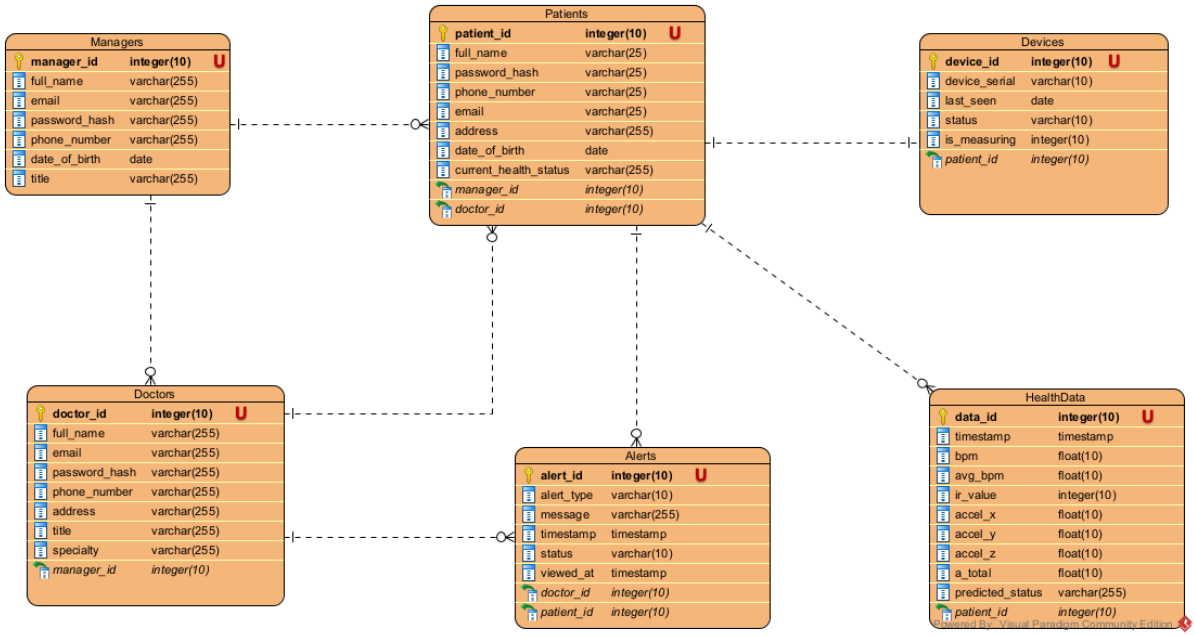
- Bảng 6 - Alerts:

+ Ghi lại các cảnh báo sức khỏe được tạo ra cho bệnh nhân.

+ Các trường chính:

- alert_id (Khóa chính)
- patient_id (Khóa ngoại tham chiếu đến Patients)
- related_data_id (Khóa ngoại tham chiếu đến HealthData)
- alert_type: Loại cảnh báo
- message: Nội dung cảnh báo
- status (trạng thái new/viewed/handled)
- viewed_by_doctor_id (Khóa ngoại tham chiếu đến Doctors).

- Thiết kế CSDL



4. Mô hình AI:

- Mục đích: Mô hình được xây dựng để dự đoán trạng thái sức khỏe của bệnh nhân theo thời gian thực:

+ Input: Nhịp tim trung bình (AvgBPM) và Tổng gia tốc (A_total).
+ Output: Trạng thái sức khỏe hiện tại

- Thu thập dữ liệu: Dữ liệu được đọc từ file sensor_data.csv và trải qua các bước tiền xử lý nghiêm ngặt để đảm bảo chất lượng cho việc huấn luyện mô hình. File sensor_data.csv chứa dữ liệu thu thập từ các cảm biến, bao gồm các cột:

+ BPM: Nhịp tim tức thời. Giá trị này phản ánh số nhịp đập của tim trong một phút tại thời điểm đo. Các giá trị đầu tiên là 0 có thể do cảm biến chưa ổn định hoặc chưa đo được nhịp.
+ AvgBPM: Nhịp tim trung bình. Đây là chỉ số quan trọng dùng để đánh giá trạng thái sức khỏe tổng quát hơn so với BPM tức thời, giúp loại bỏ nhiễu.
+ IR_value: Giá trị hồng ngoại. Thường dùng trong cảm biến nhịp tim quang học (như MAX30100/MAX30102) để phát hiện sự thay đổi thể tích máu, từ đó tính toán nhịp tim. Nó cũng có thể dùng để nhận biết xem người dùng có đang đeo thiết bị hay không.
+ Accel_X: Gia tốc trục X. Đo độ nghiêng hoặc chuyển động theo phương ngang (trái/phải).
+ Accel_Y: Gia tốc trục Y. Đo độ nghiêng hoặc chuyển động theo phương dọc (lên/xuống hoặc trước/sau)
+ Accel_Z: Gia tốc trục Z. Đo độ nghiêng hoặc chuyển động theo phương thẳng đứng (vuông góc với mặt đất).
+ A_total: Tổng gia tốc toàn phần. Đây là chỉ số chính dùng để xác định mức độ hoạt động thể chất (ngồi yên, đi bộ, chạy, ngã...).

- Xử lý dữ liệu
  - Tạo nhãn (Labeling): Dữ liệu được gán nhãn dựa trên một hàm logic classify_health_state. Hàm này định nghĩa các quy tắc phức tạp kết hợp AvgBPM và A_total để phân loại trạng thái, ví dụ: "Bình thường", "Hoạt động nhẹ", "Nhịp tim cao bất thường", "Cảnh báo nguy hiểm", v.v..
  - Mã hóa & Chuẩn hóa:
    - Các nhãn văn bản (ví dụ: "Bình thường") được chuyển đổi thành số nguyên bằng LabelEncoder.
    - Dữ liệu đầu vào (AvgBPM, A_total) được chuẩn hóa bằng StandardScaler để đưa về cùng một thang đo, giúp mô hình hội tụ tốt hơn.
    - Cả LabelEncoder và StandardScaler đều được lưu lại (label_encoder.pkl, scaler.pkl) để sử dụng trong lúc dự đoán thực tế.
  - Phân chia dữ liệu: Dữ liệu được chia thành 80% cho huấn luyện (train) và 20% cho kiểm thử (test).
  - Xử lý mất cân bằng: compute_class_weight được sử dụng để tính trọng số cho các lớp, giải quyết vấn đề mất cân bằng dữ liệu (khi một số trạng thái sức khỏe hiếm gặp hơn các trạng thái khác).

- Kiến trúc mô hình: Mô hình là một mạng nơ-ron tuần tự (keras.Sequential), bao gồm:

Model: "sequential_1"

Layer (type)	Output Shape	Param #
dense_4 (Dense)	(None, 32)	96
batch_normalization_1 (BatchNormalization)	(None, 32)	128
dense_5 (Dense)	(None, 16)	528
dropout_1 (Dropout)	(None, 16)	0
dense_6 (Dense)	(None, 8)	136
dense_7 (Dense)	(None, 6)	54

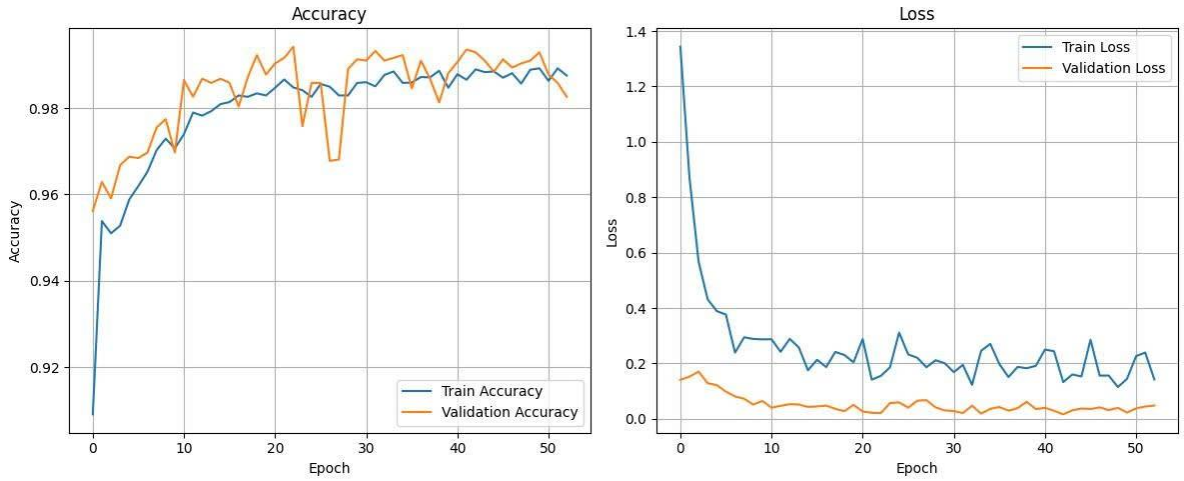
- Lớp đầu vào: Lớp Dense với 32 đơn vị và hàm kích hoạt relu, nhận đầu vào có kích thước là 2 (tương ứng với 2 đặc trưng).
- Lớp ẩn:
  - Một lớp BatchNormalization để ổn định quá trình huấn luyện.
  - Một lớp Dense với 16 đơn vị và hàm kích hoạt relu.
  - Một lớp Dropout với tỷ lệ 0.2 để giảm thiểu overfitting (học vẹt).
  - Một lớp Dense với 8 đơn vị và hàm kích hoạt relu.
- Lớp đầu ra: Lớp Dense với số lượng đơn vị bằng số lớp nhãn, sử dụng hàm kích hoạt softmax để đưa ra xác suất cho mỗi trạng thái sức khỏe.

- Huấn luyện
- Trình tối ưu (Optimizer): adam.
- Hàm mất mát (Loss): sparse_categorical_crossentropy (phù hợp với bài toán phân loại đa lớp với nhãn là số nguyên).
- Chỉ số đo lường: accuracy (độ chính xác).
- Dừng sớm (EarlyStopping): Mô hình sử dụng EarlyStopping để theo dõi val_loss (mất mát trên tập validation) và dừng huấn luyện nếu hiệu suất không cải thiện sau 10 epochs, đồng thời khôi phục trọng số tốt nhất.

- Kết quả: Sau khi huấn luyện, mô hình được đánh giá trên tập test độc lập để kiểm tra độ chính xác tổng quát. Mô hình có độ chính xác trên tập test lên tới 99.29%

```
Mô hình đã được lưu thành công!
97/97 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9950 - loss: 0.0129
Độ chính xác trên tập test: 0.9929
```

- Biểu đồ lịch sử huấn luyện (Accuracy và Loss) được vẽ lại để trực quan hóa quá trình học của mô hình, cho thấy sự hội tụ của độ chính xác trên cả tập train và validation.



- Mô hình sau khi huấn luyện được lưu vào tệp health_model.h5 để sẵn sàng cho việc triển khai và dự đoán.

IV. Thiết kế giao thức truyền thông MQTT

1. Tổng quan về giao thức MQTT và HiveMQ

- Trước tiên, MQTT (Message Queuing Telemetry Transport) là một giao thức nhắn tin tiêu chuẩn, gọn nhẹ được thiết kế cho việc giao tiếp máy với máy (M2M) hoặc Internet of Things (IoT). Nó hoạt động dựa trên mô hình publish/subscribe (xuất bản/đăng ký), cho phép các thiết bị (client) gửi dữ liệu (publisher) đến một máy chủ trung gian (broker) và các thiết bị khác (subscriber) có thể đăng ký để nhận dữ liệu từ các kênh (topic) mà chúng quan tâm.
- HiveMQ là một nền tảng phần mềm triển khai chức năng của một MQTT broker. Nó cung cấp một giải pháp chuyên nghiệp, có thể mở rộng và có độ sẵn sàng cao để kết nối hàng triệu thiết bị IoT một cách hiệu quả và an toàn. [3]
- Các đặc điểm chính của HiveMQ:
    - Độ tin cậy và khả năng mở rộng: HiveMQ được thiết kế để xử lý lượng lớn dữ liệu từ hàng triệu thiết bị mà vẫn duy trì độ trễ thấp và độ tin cậy cao.
    - Hỗ trợ doanh nghiệp: Nó cung cấp các tính năng nâng cao phù hợp cho các ứng dụng công nghiệp và doanh nghiệp, bao gồm khả năng tích hợp dễ dàng với các hệ thống backend khác.
    - Triển khai linh hoạt: HiveMQ có thể được triển khai tại chỗ (on-premise) hoặc dưới dạng dịch vụ đám mây được quản lý hoàn toàn (HiveMQ Cloud).
- Tóm lại, nếu MQTT là ngôn ngữ giao tiếp thì HiveMQ là một trong những máy chủ dịch vụ thông minh và mạnh mẽ nhất sử dụng ngôn ngữ đó để điều phối thông tin giữa các thiết bị IoT.

2. Kiến trúc hệ thống MQTT

a. ESP32 (Publisher & Subscriber)

- Đọc dữ liệu từ cảm biến MAX30102 (đo nhịp tim) và ADXL345 (gia tốc kế).
- Đóng vai trò Publisher, gửi dữ liệu đo được lên MQTT Broker theo các topic định nghĩa sẵn (health/data).
- Đồng thời, ESP32 cũng đóng vai trò Subscriber, lắng nghe các topic phản hồi cảnh báo từ hệ thống (health/alert) để thực hiện các hành động như bật LED cảnh báo

b. HiveMQ (MQTT Broker)
- Đóng vai trò là máy chủ trung gian chịu trách nhiệm quản lý, phân phối và định tuyến dữ liệu giữa các thiết bị IoT và các ứng dụng khách.
- Broker giúp tách biệt hoàn toàn giữa bên gửi (publisher) và bên nhận (subscriber), đảm bảo hệ thống hoạt động ổn định, mở rộng và dễ bảo trì.

c. Trang web / Ứng dụng giám sát (Subscriber & Publisher)
- Đóng vai trò Subscriber, đăng ký nhận dữ liệu cảm biến từ các topic để hiển thị thông tin nhịp tim, chuyển động và cảnh báo bất thường cho người dùng.
- Đồng thời, trang web cũng đóng vai trò Publisher, gửi các thông điệp cảnh báo người dùng lên broker khi có các dấu hiệu bất thường về sức khỏe.

3. Thiết kế topic MQTT

Thiết bị	Vai trò	Topic	Kiểu dữ liệu	Ý nghĩa
esp32	Publisher	health/data	{"device_serial":"1", "bpm":60.00, "avg_bpm":60.00, "ir_value":60.00, "accel_x":0.43, "accel_y":-0.67, "accel_z":9.57}	Gửi dữ liệu do 2 cảm biến đo được
esp32	Subscriber	health/alert	LED ON	Nhận cảnh báo từ trang web
Trang web	Subscriber	health/data	{"device_serial":"1", "bpm":60.00, "avg_bpm":60.00, "ir_value":60.00, "accel_x":0.43, "accel_y":-0.67, "accel_z":9.57}	Nhận dữ liệu đo được từ cảm biến

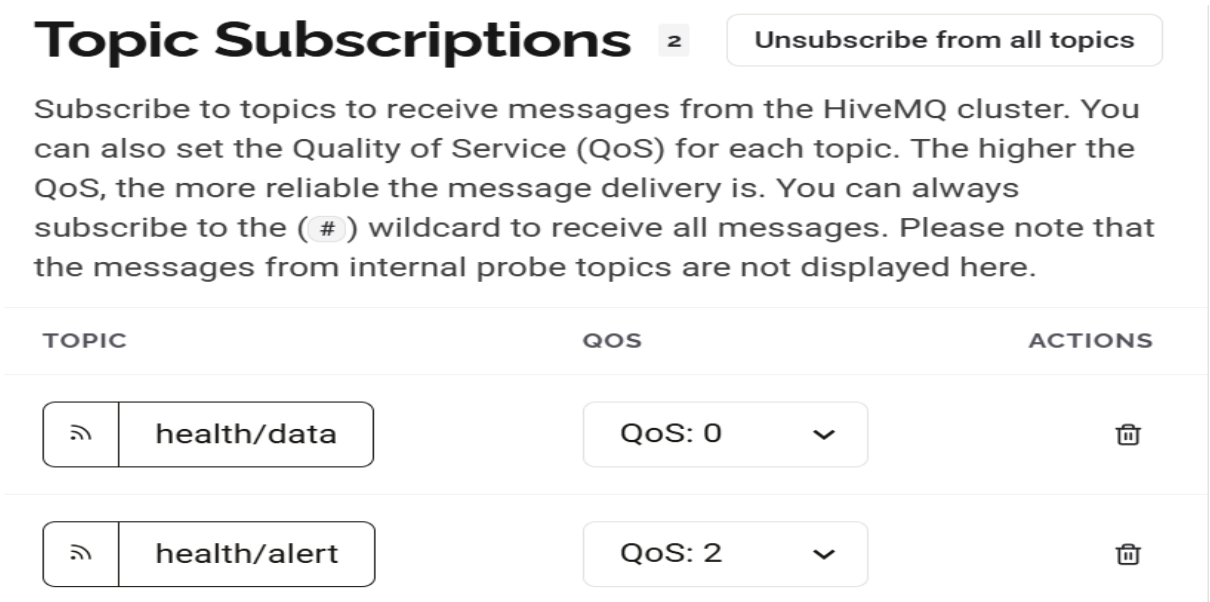
Trang web	Publisher	health/alert	LED ON	Gửi cảnh báo cho esp32

- Đăng ký topic trên hive mq:
+ Topic health/data được đăng kí với QoS 0: Đây là dữ liệu cảm biến gửi liên tục theo thời gian thực (heart rate, movement) nên việc dùng QoS 0 (At most once) hoàn toàn phù hợp vì:

- Dữ liệu được cập nhật liên tục → mất 1–2 gói không ảnh hưởng lớn.
- Giảm tải cho ESP32 và broker.
- Độ trễ thấp nhất, phù hợp với streaming data.
- Các hệ thống IoT thực tế (như đo nhiệt độ, nhịp tim, độ ẩm) gần như luôn dùng QoS 0 để tối ưu tốc độ.

+ Đăng kí topic health/alert với QoS 2: vì topic này đóng vai trò gửi cảnh báo bất thường (alert) từ web → ESP32, kích hoạt LED cảnh báo hoặc các hành động an toàn. Đây là dữ liệu quan trọng, cần đảm bảo không được mất gói, vì:

- Nếu cảnh báo gửi thất bại → ESP32 không bật đèn → người dùng không nhận biết sự cố.
- Đây là loại dữ liệu hiếm khi gửi, nhưng mang tính sống còn.
- Như vậy dùng QoS 2 (Exactly once) sẽ: đảm bảo gói tin tới ESP32 đúng 1 lần, tránh gửi trùng lặp lệnh, tránh sai lệch, đảm bảo độ an toàn cao nhất trong truyền thông MQTT.



4. Luồng truyền thông của hệ thống

- Bước 1: Cảm biến đo nhịp tim và chuyển động rồi gửi tín hiệu đến ESP32.
- Bước 2: ESP32 publish dữ liệu lên HiveMQ Broker qua topic health/data
- Bước 3: HiveMQ Broker phân phối dữ liệu đến các subscriber đăng ký trước đó
- Bước 4: Web gửi dữ liệu cho mô hình AI để phân tích
- Bước 5: Mô hình AI xử lý và trả về kết quả cho web
- Bước 6: Web hiển thị thông tin phân tích theo thời gian thực
- Bước 7: Nếu mô hình phát hiện dấu hiệu bất thường, web sẽ publish thông điệp cảnh báo lên HiveMQ qua topic health/alert
- Bước 8: ESP32 nhận thông điệp cảnh báo, sau đó bật LED cảnh báo

Chương 5: Kết quả đạt được và hướng phát triển

I. Kết quả đạt được

- Đề tài "Hệ thống theo dõi sức khỏe theo thời gian thực" đã xây dựng được một mô hình hoàn chỉnh kết hợp giữa phần cứng IoT, giao thức truyền thông MQTT, nền tảng HiveMQ và mô hình AI để dự đoán, giám sát và cảnh báo tình trạng sức khỏe người dùng theo thời gian thực.
- Trong phạm vi thực hiện, nhóm đã đạt được các kết quả chính sau:
  - Thu thập và xử lý dữ liệu cảm biến thực tế: ESP32 đọc tín hiệu từ cảm biến MAX30102 và ADXL345, truyền dữ liệu ổn định qua giao thức MQTT.
  - Xây dựng hệ thống giao tiếp hai chiều: ESP32 vừa gửi dữ liệu cảm biến lên HiveMQ (Publisher), vừa nhận tín hiệu điều khiển bật LED cảnh báo từ Web (Subscriber).
  - Phát triển giao diện web: Hiển thị dữ liệu đo được, biểu đồ thời gian thực, đồng thời nhận và phản hồi thông báo từ mô hình AI.
  - Tích hợp mô hình AI: Mô hình học máy được huấn luyện để nhận biết dấu hiệu bất thường trong dữ liệu nhịp tim và chuyển động, hỗ trợ việc ra quyết định tự động.
  - Cảnh báo tức thời: Khi phát hiện tình huống nguy hiểm, hệ thống tự động bật đèn cảnh báo và thông báo cho người dùng.
  - Hệ thống cho thấy khả năng hoạt động ổn định, có độ trễ thấp và tính chính xác cao trong việc thu thập – phân tích dữ liệu. Qua đó, đề tài chứng minh tiềm năng ứng dụng của IoT kết hợp AI trong lĩnh vực theo dõi sức khỏe và cảnh báo sớm.

II. Hướng phát triển

- Mặc dù hệ thống đã hoạt động tốt trong phạm vi mô phỏng và thử nghiệm, nhóm nhận thấy vẫn còn nhiều tiềm năng mở rộng trong tương lai:

+ Mở rộng số lượng cảm biến:

  - Bổ sung cảm biến nhiệt độ cơ thể, cảm biến ECG hoặc cảm biến oxy (SpO_2) chính xác cao để phân tích sâu hơn về tình trạng sức khỏe.
  - Kết hợp cảm biến GPS để theo dõi vị trí khi phát hiện té ngã hoặc tình huống khẩn cấp.

+ Cải thiện mô hình AI:

  - Thu thập dữ liệu thực tế nhiều hơn để huấn luyện mô hình AI chính xác và cá nhân hóa theo từng người dùng.
  - Ứng dụng Deep Learning (RNN, LSTM) để phân tích chuỗi tín hiệu thời gian và dự đoán xu hướng sức khỏe trong tương lai.

+ Phát triển ứng dụng di động (Mobile App): Tạo app Android/iOS kết nối trực tiếp với MQTT để hiển thị dữ liệu, nhận cảnh báo và lưu trữ lịch sử sức khỏe.

+ Tối ưu hiệu năng và bảo mật:

- Sử dụng giao thức MQTTS (MQTT over SSL/TLS) để mã hóa dữ liệu khi truyền qua mạng.
- Tối ưu mã nguồn ESP32 để tiết kiệm năng lượng, hướng đến khả năng hoạt động lâu dài bằng pin.

+ Triển khai thực tế:

- Xây dựng nguyên mẫu đeo tay hoàn chỉnh (wearable device).
- Thử nghiệm trên nhóm người dùng thật trong các tình huống khác nhau để đánh giá độ chính xác và độ tin cậy.

III. Kết luận

- Qua quá trình nghiên cứu và triển khai, đề tài đã xây dựng được một mô hình hoàn chỉnh, kết hợp đồng bộ giữa phần cứng IoT, giao thức truyền thông MQTT, nền tảng HiveMQ và mô hình AI. Hệ thống đã chứng minh khả năng thu thập dữ liệu cảm biến theo thời gian thực, truyền tải ổn định, phân tích chính xác và phản hồi kịp thời trong các tình huống bất thường.
- Những kết quả đạt được cho thấy hệ thống vận hành ổn định, có độ trễ thấp và đáp ứng tốt yêu cầu giám sát liên tục. Dữ liệu nhịp tim và chuyển động được thu thập, xử lý và phân tích một cách hiệu quả, trong khi cơ chế cảnh báo tự động đảm bảo hỗ trợ người dùng trong các tình huống tiềm ẩn nguy hiểm.
- Từ góc độ tổng thể, đề tài đã chứng minh tính khả thi và tiềm năng ứng dụng của việc tích hợp IoT – MQTT – AI trong các hệ thống theo dõi sức khỏe thông minh. Đây có thể coi là nền tảng quan trọng để mở rộng và phát triển thành các giải pháp chăm sóc sức khỏe tiên tiến, hướng tới tính tự động hóa cao và khả năng hỗ trợ người dùng trong đời sống thực tế.
- Hệ thống không chỉ có ý nghĩa thực tiễn trong việc chăm sóc cá nhân mà còn mở ra hướng ứng dụng rộng hơn trong các lĩnh vực như y tế thông minh, nhà thông minh, và hỗ trợ người cao tuổi.

Tài liệu tham khảo

[1] MQTT, "MQTT," [Online]. Available: https://mqtt.org/.

[2] N. C. Trinh, Bài giảng Phát triển hệ thống và ứng dụng IOT.

[3] HiveMQ, "HiveMQ," [Online]. Available: https://www.hivemq.com/.