HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO GIỮA 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

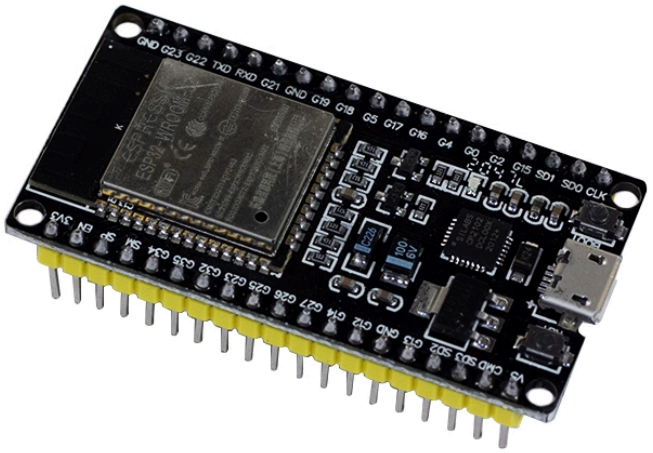
- Hệ thống chỉ hoạt động trên web phân quyền.
- Dữ liệu được lưu và xử lý trực tiếp trên server Flask, không dùng cơ sở dữ liệu đám mây quy mô lớn.
- Hệ thống hiện tại chỉ tập trung vào các chỉ số:
    - Nhịp tim (BPM)
    - Gia tốc 3 trục
    - Chưa mở rộng sang các loại cảm biến khác như ECG, nhiệt 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.



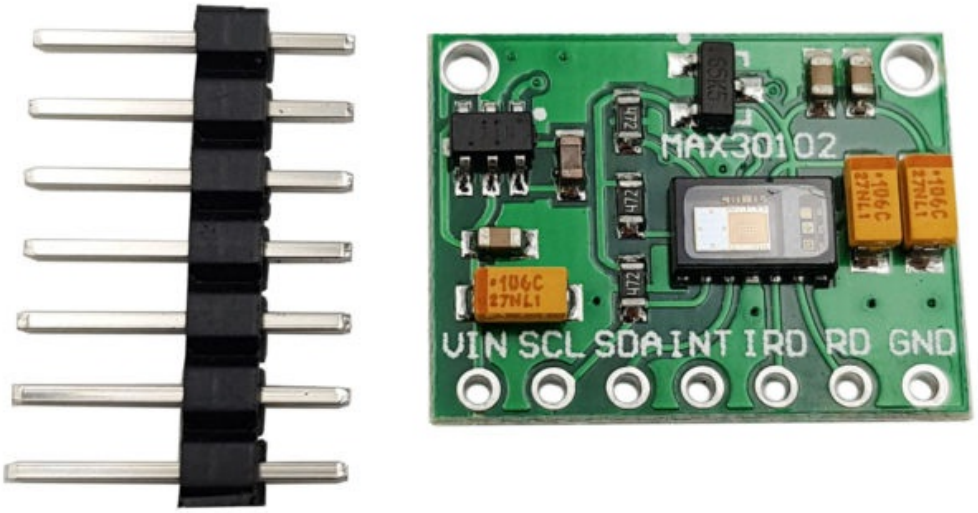
Hình ảnh 1: Vi điều khiển ESP32

- 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

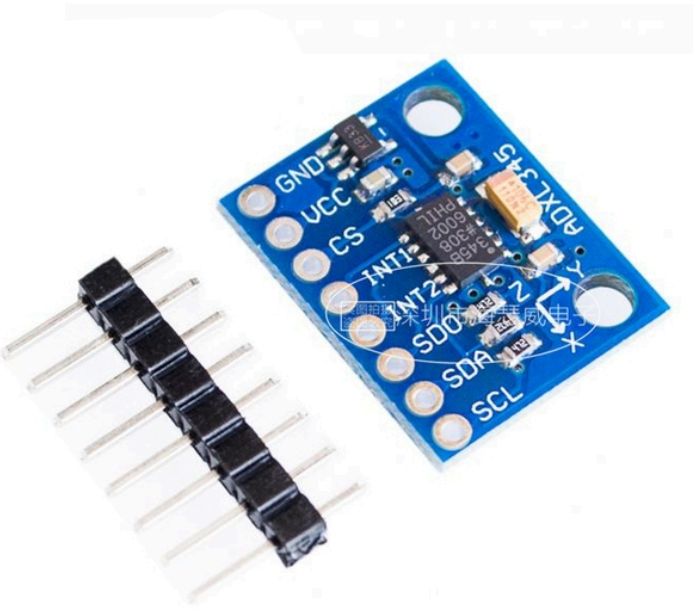
- Nguyên lý hoạt động:
  - Khi đèn LED phát sáng, một phần ánh sáng sẽ bị mạch máu hấp thụ và phần còn lại phản xạ lại về photodiode.
  - Khi tim co bóp và giãn ra, lượng máu chảy qua mao mạch thay đổi → làm thay đổi cường độ ánh sáng phản xạ.
  - Cảm biến ghi nhận sự thay đổi này để tính toán nhịp tim (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 1-2 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).

- 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.

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 nhịp 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.

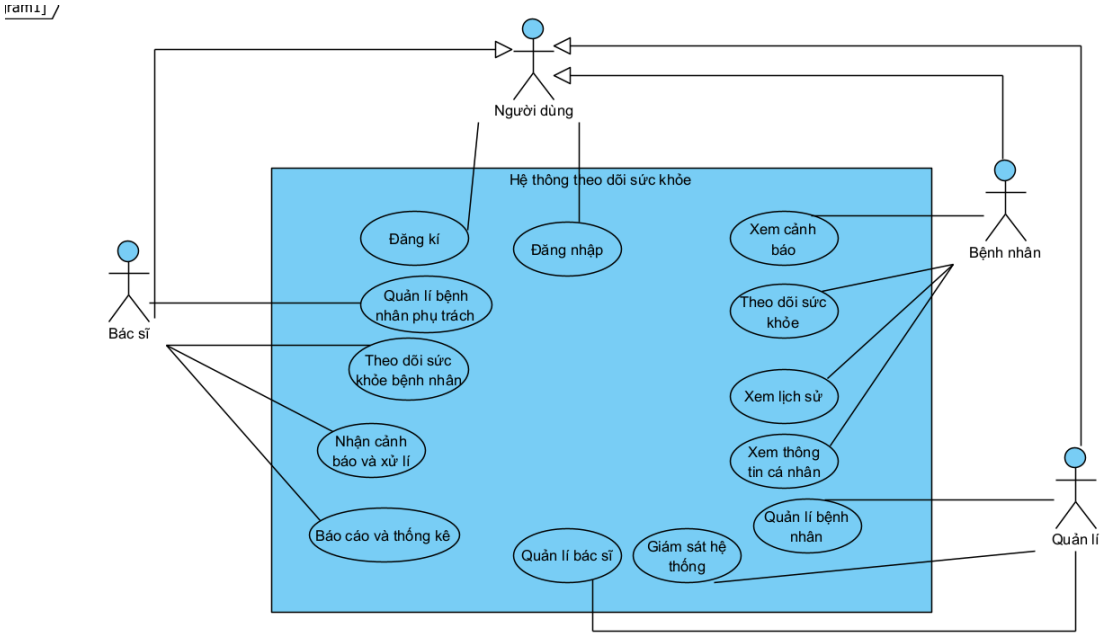
- Quản lý có thể truy cập toàn bộ dữ liệu hệ thống, thống kê, và phân quyền người dùng.

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.
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.

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.

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.

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, ví dụ 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 1 (At least once): Nên được sử dụng cho việc gửi dữ liệu nhịp tim để đảm bảo tin nhắn đến được server ít nhất một lần. Điều này chấp nhận khả năng tin nhắn bị lặp lại, nhưng đảm bảo không mất dữ liệu.
- 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 chia công việc và kế hoạch triển khai

Giai đoạn	Mục tiêu chính	Nội dung công việc	Người phụ trách
1. Khởi động & Thiết kế	Xác định yêu cầu, kiến trúc hệ thống	- Phân tích đề tài, xác định đầu vào & đầu ra. - Hoàn thành tài liệu SRS	Cả nhóm
2. Phát triển IoT Device	Lấy dữ liệu cảm biến và truyền MQTT	- Lập trình ESP32 đọc dữ liệu từ MAX30102, ADXL345. - Gửi dữ liệu qua MQTT đến broker. - Thử nghiệm độ ổn định kết nối WiFi.	Cả nhóm
3. Xây dựng Backend Flask	Nhận dữ liệu, lưu vào SQL, xử lý ban đầu	- Flask nhận dữ liệu từ MQTT (paho-mqtt). - Lưu dữ liệu vào SQL. - Bắt đầu tiền xử lý (lọc nhiễu, chuẩn hóa).	Cả nhóm
4. Huấn luyện & Tích hợp AI	Dự đoán trạng thái sức khỏe	- Thu thập dữ liệu thật để huấn luyện mô hình	Cả nhóm

		- Dùng TensorFlow/Keras huấn luyện mô hình AI. - Xuất model và tích hợp Flask để dự đoán real-time. - Sinh cảnh báo theo nhãn dự đoán.	
5. Phát triển Giao diện Web	Hiển thị dữ liệu real-time & cảnh báo	- Xây dựng web dashboard bằng Flask + HTML + JS + Chart.js. - Hiển thị biểu đồ BPM và trạng thái sức khỏe. - Hoàn thiện các tính năng đối với mỗi người dùng	Cả nhóm
6. Kiểm thử hệ thống tổng thể	Đảm bảo hoạt động ổn định	- Chạy thử ESP32 gửi dữ liệu thật → MQTT → Flask → Web. - Test cảnh báo AI real-time. - Kiểm tra hiển thị web nhiều tài khoản.	Cả nhóm
7. Hoàn thiện & Báo cáo	Chuẩn bị nộp cuối kỳ	- Chuẩn hóa code. - Hoàn thành báo cáo, mô hình thiết bị	Cả nhóm