TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

VÀ TRUYỀN THÔNG VIỆT – HÀN

**KHOA KỸ THUẬT MÁY TÍNH & ĐIỆN TỬ**



**BÁO CÁO**

**ĐỒ ÁN CHUYÊN NGÀNH II**

***Đề Tài*:** **XÂY DỰNG HỆ THỐNG GIÁM SÁT SỨC KHỎE THÔNG MINH**

**Sinh viên thực hiện     : Thái Văn Hòa**

**Lớp : 21IR**

**Niên khoá : 2021 - 2026**

**Giảng viên hướng dẫn : THS. Lê Kim Trọng**

***Đà Nẵng, ngày 01 tháng 12 năm 2024***

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

VÀ TRUYỀN THÔNG VIỆT – HÀN

**KHOA KỸ THUẬT MÁY TÍNH & ĐIỆN TỬ**



**BÁO CÁO**

**ĐỒ ÁN CHUYÊN NGÀNH II**

***Đề Tài*:** **XÂY DỰNG HỆ THỐNG GIÁM SÁT SỨC KHỎE THÔNG MINH**

**Sinh viên thực hiện     : Thái Văn Hòa**

**Lớp : 21IR**

**Niên khoá : 2021 - 2026**

**Giảng viên hướng dẫn : THS. Lê Kim Trọng**

***Đà Nẵng, ngày 01 tháng 12 năm 2024***

# NHẬN XÉT

**(Của Giảng viên hướng dẫn)**

....................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

Đà Nẵng,….. tháng … năm 2024

Giảng viên hướng dẫn

# LỜI CẢM ƠN

Lời đầu tiên cho phép em gửi lời cảm ơn tới các Thầy Cô giáo, các cán bộ công tác tại Trường ĐH Công Nghệ Thông Tin Và Truyền Thông Việt Hàn đã tạo mọi điều kiện giúp đỡ em trong thời gian xây dựng và hoàn thành báo cáo

Đặc biệt em xin bày tỏ lòng biết ơn sâu sắc tới cô **THS. Lê Kim Trọng** giảng viên hướng dẫn môn Đồ Án Chuyên Ngành I đã tận tình giúp đỡ, chỉ bảo về nghiệp vụ và trực tiếp hướng dẫn em trong suốt quá trình hoàn thành báo cáo này.

Tuy nhiên do thời gian có hạn và cùng với nhiều nguyên nhân khác, mặc dù em đã nỗ lực hết mình xong đồ án của em vẫn còn mắc phải những thiếu sót và hạn chế. Em rất mong nhận được sự thông cảm và chỉ bảo của các Thầy Cô cùng tất cả các bạn.

Em xin chân thành cảm ơn!

***Đà Nẵng, 01 tháng 12 năm 2024***

*Sinh viên thực hiện*

Thái Văn Hòa

**MỤC LỤC**

[NHẬN XÉT iii](#_Toc184105843)

[LỜI CẢM ƠN iv](#_Toc184105844)

[DANH MỤC HÌNH VẼ vi](#_Toc184105845)

[DANH MỤC CÁC TỪ VIẾT TẮT vii](#_Toc184105846)

[MỞ ĐẦU viii](#_Toc184105847)

[CHƯƠNG 1 - TỔNG QUAN VỀ ĐỀ TÀI 1](#_Toc184105848)

[1.1 Tổng quan 1](#_Toc184105849)

[1.1.1 Lý do chọn đề tài 1](#_Toc184105850)

[1.1.2 Mục tiêu và nội dung thực hiện đề tài 1](#_Toc184105851)

[1.1.3 Phương pháp nghiên cứu của đề tài 2](#_Toc184105852)

[1.2 Kết quả và ý nghĩa đề tài 2](#_Toc184105853)

[1.2.1. Kết quả 2](#_Toc184105854)

[1.2.2. Ý nghĩa đề tài 2](#_Toc184105855)

[1.3 Công cụ 3](#_Toc184105856)

[1.3.1. Phần mềm lấp trình 3](#_Toc184105857)

[1.3.2. Phần mềm vẽ mạch Altium 5](#_Toc184105858)

[1.3.3. Flutter 7](#_Toc184105859)

[CHƯƠNG 2 – CƠ SỞ LÝ THUYẾT 9](#_Toc184105860)

[2.1 Giới thiệu chung về ESP32 9](#_Toc184105861)

[2.1.1 Cấu Trúc của ESP32 9](#_Toc184105862)

[2.1.2 Nguyên Lý Hoạt Động 10](#_Toc184105863)

[2.1.3 Ứng Dụng của ESP32 11](#_Toc184105864)

[2.1.4 Lợi Ích của Việc Sử Dụng ESP32 13](#_Toc184105865)

[2.2 Web server 14](#_Toc184105866)

[CHƯƠNG 3 – THIẾT KẾ & LẬP TRÌNH HỆ THỐNG 14](#_Toc184105867)

[3.1 Các Module và các linh kiện cần có 14](#_Toc184105868)

[3.1.1. ESP32 14](#_Toc184105869)

[3.1.2. Cảm biến nhiệt độ DS18B20 15](#_Toc184105870)

[3.1.3. Màn hình OLED I2C 0,96 inch 16](#_Toc184105871)

[3.1.4. Relay 5V 16](#_Toc184105872)

[3.2 Thiết kế phần cứng 18](#_Toc184105873)

[3.2.1. Vẽ schematic trên Altium 18](#_Toc184105874)

[3.2.2. Vẽ mạch PCB 19](#_Toc184105875)

[3.3 Các thư viện cần dùng 21](#_Toc184105876)

[3.3.1. WiFi.h 21](#_Toc184105877)

[3.3.2. AsyncTCP.h 21](#_Toc184105878)

[3.3.3. ESPAsyncWebServer.h 21](#_Toc184105879)

[3.3.4. ESPDash.h 21](#_Toc184105880)

[3.3.5. OneWire.h và DallasTemperature.h 21](#_Toc184105881)

[3.3.6. SPI.h và Wire.h 21](#_Toc184105882)

[3.3.7. Adafruit\_GFX.h và Adafruit\_SSD1306.h 21](#_Toc184105883)

[3.4 Mô phỏng thực tế hệ thống 22](#_Toc184105884)

[3.4.1. Sơ dồ mạch 22](#_Toc184105885)

[3.4.2. Mô phỏng testboard 23](#_Toc184105886)

[3.5 Sơ đồ và cách thức hoạt động của hệ thống 23](#_Toc184105887)

[3.5.1. Sơ đồ hoạt động của hệ thống 24](#_Toc184105888)

[3.5.2. Cách thức hoạt động của hệ thống 24](#_Toc184105889)

[3.6 Sản phẩm hoàn thiện 25](#_Toc184105890)

[3.6.1. Mạch PCB hoàn chỉnh 25](#_Toc184105891)

[3.6.2. Giao diện dashboard web server 26](#_Toc184105892)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 27](#_Toc184105893)

[TÀI LIỆU THAM KHẢO 28](#_Toc184105894)

[PHỤ LỤC 29](#_Toc184105895)

# DANH MỤC HÌNH VẼ

[Hình 1. 1: Phần mềm Arduno IDE 3](#_Toc167246005)

[Hình 1. 2: Phần mềm Altium Design 4](#_Toc167246006)

[Hình 2. 1: Datasheet ESP32 5](#_Toc167245351)

[Hình 2. 2: Các loại ESP32 7](#_Toc167245352)

[Hình 2. 3: Web Server 8](#_Toc167245353)

[Hình 3. 1: ESP32 9](#_Toc167247149)

[Hình 3. 2: Cảm biến nhiệt độ DS18B20 10](#_Toc167247150)

[Hình 3. 3: Màn hình OLED I2C 11](#_Toc167247151)

[Hình 3. 4: Relay 12](#_Toc167247152)

[Hình 3. 5: Schematic 13](#_Toc167247153)

[Hình 3. 6: Mạch PCB ở lớp Top layer 14](#_Toc167247154)

[Hình 3. 7: Mạch PCB ở lớp Bottom layer 15](#_Toc167247155)

[Hình 3. 8: Mạch PCB 3D 15](#_Toc167247156)

[Hình 3. 9: Sơ đồ mạch 17](#_Toc167247157)

[Hình 3. 10: Mô phỏng hệ thống trên testboard 17](#_Toc167247158)

[Hình 3. 11: Sơ đồ hoạt động hệ thống 18](#_Toc167247159)

[Hình 3. 12: Sản phẩm hoàn thiện 19](#_Toc167247160)

[Hình 3. 13: Giao diện web server trên laptop 20](#_Toc167247161)

[Hình 3. 14: Giao diện web server trên diện thoại 20](#_Toc167247162)

# DANH MỤC CÁC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| STT | Cụm từ | Viết tắt |
| 1 | Espressif System Platform | ESP |
| 2 | Internet of Think | IoT |

# MỞ ĐẦU

Trong thời đại số hóa ngày nay, việc sử dụng công nghệ để cải thiện chất lượng cuộc sống đã trở thành một xu hướng không thể tránh khỏi. Đặc biệt, việc tạo ra các hệ thống thông minh, nhờ vào sự phát triển của các công nghệ như ESP32 và Web Server, đã mở ra những cơ hội mới cho chúng ta để tạo ra những giải pháp tối ưu cho cuộc sống hàng ngày.

ESP32 là một chip có khả năng kết nối không dây và nhiều tính năng khác, đã trở thành một công cụ quan trọng trong việc phát triển các hệ thống thông minh. Bên cạnh đó, Web Server, một nền tảng cho phép người dùng tạo ra các ứng dụng IoT một cách dễ dàng và nhanh chóng, cũng đã đóng góp một phần quan trọng trong việc tạo ra những giải pháp IoT hiệu quả.

Đề tài sẽ tập trung vào việc khám phá sức mạnh của ESP32 và Web Server, cũng như cách chúng có thể được sử dụng để tạo ra các hệ thống thông minh. Chúng ta sẽ đi sâu vào việc tìm hiểu về cách hoạt động của ESP32 và Web Server, cũng như cách chúng có thể được lập trình để tạo ra các hệ thống thông minh.

Kết quả , chúng ta sẽ áp dụng những kiến thức đã học để thiết kế và lập trình một hệ thống thông minh cụ thể, minh họa cho khả năng và tiềm năng của ESP32 và Web Server trong việc tạo ra các giải pháp IoT hiệu quả. Thông qua đề tài này, ta sẽ có thể mở rộng hiểu biết về công nghệ và tạo ra những giải pháp tốt nhất cho cuộc sống của chúng ta.

# CHƯƠNG 1 - TỔNG QUAN VỀ ĐỀ TÀI

## Tổng quan

### Lý do chọn đề tài

Trong bối cảnh sự phát triển nhanh chóng của công nghệ thông tin và Internet of Things (IoT), việc ứng dụng các thiết bị thông minh trong lĩnh vực chăm sóc sức khỏe ngày càng trở nên phổ biến và quan trọng. Một trong những xu hướng lớn hiện nay là việc theo dõi và giám sát sức khỏe từ xa, giúp người dùng có thể chủ động kiểm soát sức khỏe của mình mọi lúc, mọi nơi. Sự phát triển của các vi điều khiển như ESP32, cùng với các cảm biến sinh học như MAX30102, đã mở ra cơ hội cho việc tạo ra các hệ thống giám sát sức khỏe thông minh.

Đề tài này được chọn với mục tiêu xây dựng một hệ thống giám sát sức khỏe cá nhân dựa trên nền tảng ESP32, có khả năng đo nhịp tim và nồng độ oxy trong máu, giúp cung cấp dữ liệu tức thời cho người sử dụng và bác sĩ để theo dõi tình trạng sức khỏe. Hệ thống này cũng được thiết kế để gửi thông báo cho người dùng khi các chỉ số vượt quá ngưỡng an toàn, hỗ trợ phát hiện sớm các vấn đề sức khỏe.

### Mục tiêu và nội dung thực hiện đề tài

**Mục tiêu:**

* Xây dựng hệ thống giám sát sức khỏe từ xa sử dụng ESP32: Sử dụng ESP32 để kết nối cảm biến và hiển thị thông tin về nhịp tim, mức oxy trong máu, cùng các thông số khác của người sử dụng.
* Tích hợp hệ thống với ứng dụng di động: Phát triển một ứng dụng di động (sử dụng Flutter) để nhận dữ liệu và thông báo từ hệ thống theo dõi sức khỏe.
* Mô phỏng và kiểm tra hệ thống: Mô phỏng các thành phần hệ thống, kiểm tra độ chính xác của cảm biến và khả năng hoạt động của hệ thống trong môi trường thực tế.

**Nội dung thực hiện:**

1. Tìm hiểu và nghiên cứu ESP32 và cảm biến MAX30102: Các thành phần của hệ thống, cách thức hoạt động của từng thiết bị, và các thuật toán cần thiết để xử lý dữ liệu.
2. Thiết kế phần cứng của hệ thống: Thiết kế sơ đồ mạch, PCB và các kết nối điện tử.
3. Lập trình hệ thống: Phát triển phần mềm điều khiển ESP32, xử lý dữ liệu cảm biến, và phát triển ứng dụng di động để hiển thị kết quả.
4. Mô phỏng và kiểm tra: Kiểm tra hoạt động của hệ thống thông qua mô phỏng và thực tế.

### Phương pháp nghiên cứu của đề tài

Phương pháp nghiên cứu

**- Nghiên cứu lý thuyết:** Tìm hiểu các kiến thức nền tảng về ESP32, cảm biến MAX30102, các thuật toán xử lý tín hiệu sinh học, và giao thức truyền thông dữ liệu.

**- Thực nghiệm và mô phỏng:** Thiết kế mạch điện, lập trình ESP32 và ứng dụng di động, kiểm tra hệ thống bằng phần mềm mô phỏng trước khi triển khai thực tế.

**- Kiểm tra và đánh giá:** Đo lường độ chính xác của cảm biến và hệ thống, đánh giá hiệu suất và độ tin cậy của hệ thống trong các tình huống khác nhau.

## Kết quả và ý nghĩa đề tài

### 1.2.1. Kết quả

Sau khi thực hiện nghiên cứu và phát triển, hệ thống hoàn thiện có thể đo lường các chỉ số sức khỏe quan trọng như nhịp tim và mức độ oxy trong máu bằng cảm biến MAX30102. Dữ liệu thu được từ cảm biến được xử lý và hiển thị trên màn hình OLED 0.96 inch, đồng thời hệ thống gửi thông báo qua ứng dụng di động được phát triển bằng Flutter. Các thông báo này có thể cảnh báo người dùng nếu các chỉ số vượt quá ngưỡng an toàn, ví dụ như nhịp tim quá nhanh hoặc nồng độ oxy trong máu thấp.

### 1.2.2. Ý nghĩa đề tài

Hệ thống giám sát sức khỏe từ xa sử dụng ESP32 có ý nghĩa quan trọng trong việc nâng cao chất lượng chăm sóc sức khỏe, đặc biệt trong bối cảnh dịch bệnh hoặc khi không thể tiếp xúc trực tiếp với bác sĩ. Hệ thống này không chỉ giúp người sử dụng theo dõi sức khỏe cá nhân một cách chủ động, mà còn hỗ trợ phát hiện sớm các dấu hiệu của bệnh tật, từ đó có thể can thiệp kịp thời. Hệ thống cũng có thể được tích hợp vào các ứng dụng y tế từ xa trong tương lai, giúp cải thiện chất lượng dịch vụ y tế.

## Công cụ

### 1.3.1. Phần mềm lấp trình

Thony là một IDE (Integrated Development Environment) dành cho việc lập trình Python, được thiết kế đặc biệt cho người mới bắt đầu và các ứng dụng nhúng. Thony là một công cụ rất phù hợp khi phát triển các ứng dụng IoT, đặc biệt là khi lập trình cho các vi điều khiển như ESP32. Với giao diện đơn giản và dễ sử dụng, Thony giúp lập trình viên nhanh chóng viết mã, kiểm tra và gỡ lỗi ứng dụng mà không cần phải thiết lập một môi trường phức tạp.

A black and white symbol

Description automatically generated

Hình 1. 1: Phần mềm Arduno IDE

Thony cung cấp một môi trường phát triển hoàn chỉnh với các tính năng như hỗ trợ gỡ lỗi trực tiếp, xem và điều khiển các biến, và kiểm tra mã nguồn ngay lập tức. Đặc biệt, Thony hỗ trợ việc lập trình Python trực tiếp cho ESP32, giúp quá trình phát triển ứng dụng IoT trở nên đơn giản và nhanh chóng.

**Các tính năng nổi bật của Thony**

1. **Giao diện người dùng đơn giản và thân thiện**: Thony có giao diện người dùng trực quan, dễ sử dụng, giúp người mới học lập trình hoặc lập trình viên có ít kinh nghiệm trong phát triển nhúng có thể làm quen nhanh chóng. Với các cửa sổ như trình soạn thảo mã, bảng điều khiển shell, và bảng điều khiển gỡ lỗi, người dùng có thể dễ dàng kiểm tra và thực hiện các thay đổi trong mã nguồn.
2. **Hỗ trợ lập trình Python cho ESP32**: Thony hỗ trợ việc lập trình Python cho các thiết bị nhúng như ESP32 thông qua MicroPython. Điều này giúp lập trình viên dễ dàng viết các ứng dụng Python trực tiếp trên ESP32 mà không cần phải cài đặt các công cụ phức tạp. MicroPython là một phiên bản nhẹ của Python, được thiết kế đặc biệt cho các vi điều khiển, giúp ESP32 có thể chạy các ứng dụng Python mà không chiếm quá nhiều tài nguyên.
3. **Chạy và gỡ lỗi mã trực tiếp trên thiết bị**: Thony hỗ trợ gỡ lỗi trực tiếp trên ESP32, cho phép lập trình viên kiểm tra mã nguồn và theo dõi các biến trong khi chương trình đang chạy. Thông qua chức năng này, người lập trình có thể dễ dàng phát hiện và khắc phục lỗi trong quá trình phát triển, giúp giảm thời gian debug và cải thiện chất lượng mã.
4. **Hỗ trợ console (chạy lệnh trực tiếp)**: Thony tích hợp một console (cửa sổ điều khiển) cho phép người dùng chạy các lệnh Python trực tiếp. Điều này giúp lập trình viên có thể kiểm tra nhanh chóng các đoạn mã nhỏ, kiểm tra các thư viện hoặc thử nghiệm các tính năng của ESP32 mà không cần phải biên dịch toàn bộ chương trình.
5. **Tích hợp với MicroPython**: MicroPython là một phiên bản Python tối ưu hóa cho vi điều khiển. Nó có thể chạy trực tiếp trên các vi điều khiển như ESP32, cho phép người lập trình sử dụng các thư viện Python phổ biến để tương tác với phần cứng. Thony hỗ trợ giao tiếp và lập trình trực tiếp với MicroPython trên ESP32, giúp phát triển ứng dụng nhúng trở nên dễ dàng và nhanh chóng.
6. **Công cụ gỡ lỗi mạnh mẽ**: Thony cung cấp một công cụ gỡ lỗi mạnh mẽ giúp lập trình viên theo dõi và phân tích mã nguồn, giúp phát hiện lỗi ngay khi chương trình chạy. Các công cụ gỡ lỗi này bao gồm khả năng dừng lại tại các điểm cụ thể trong mã (breakpoint), xem giá trị của các biến, và theo dõi luồng chương trình.

### 1.3.2. Phần mềm vẽ mạch Altium

Altium Designer là phần mềm thiết kế mạch điện tử tích hợp (EDA) mạnh mẽ, được phát triển bởi Altium Limited, giúp người dùng tạo ra các thiết kế mạch điện tử và PCB với độ chính xác cao. Phần mềm này cung cấp một môi trường phát triển tích hợp, hỗ trợ tất cả các giai đoạn thiết kế mạch, từ sơ đồ nguyên lý (schematic), thiết kế PCB, đến kiểm tra và mô phỏng mạch. Altium Designer không chỉ giúp đơn giản hóa quy trình thiết kế mà còn tối ưu hóa hiệu suất mạch, đảm bảo tính ổn định và hiệu quả khi triển khai sản phẩm.



Hình 1. 2: Phần mềm Altium Design

**Các tính năng nổi bật của Altium Designer**

1. **Tích hợp toàn bộ quy trình thiết kế**: Altium Designer cung cấp một môi trường thiết kế tích hợp, cho phép người dùng thực hiện tất cả các bước trong quy trình thiết kế mạch điện tử từ vẽ sơ đồ nguyên lý, thiết kế PCB, đến kiểm tra và mô phỏng mạch trong một phần mềm duy nhất. Điều này giúp giảm thiểu sự sai sót trong quá trình chuyển giao dữ liệu giữa các công cụ khác nhau và tăng cường sự đồng bộ trong dự án.
2. **Vẽ sơ đồ nguyên lý (Schematic) mạnh mẽ**: Altium cung cấp công cụ vẽ sơ đồ nguyên lý dễ sử dụng, cho phép thiết kế mạch điện với các thành phần, linh kiện điện tử tiêu chuẩn. Các linh kiện có sẵn trong thư viện phong phú của Altium, giúp người dùng nhanh chóng chọn lựa các linh kiện cần thiết cho dự án. Phần mềm hỗ trợ các công cụ để kiểm tra kết nối mạch, giúp giảm thiểu lỗi thiết kế ngay từ giai đoạn sơ đồ.
3. **Thiết kế mạch PCB chính xác**: Sau khi hoàn thành sơ đồ nguyên lý, Altium Designer cho phép người dùng chuyển đổi thiết kế sang mạch PCB. Phần mềm cung cấp các công cụ mạnh mẽ để thiết kế bố trí PCB, bao gồm khả năng điều chỉnh vị trí các linh kiện, dây nối (trace), và các thông số khác như chiều rộng đường dẫn (trace width), khoảng cách giữa các linh kiện, đảm bảo thiết kế có tính khả thi và đáp ứng yêu cầu về kích thước và hiệu suất.
4. **Kiểm tra DRC (Design Rule Check) và ERC (Electrical Rule Check)**: Altium Designer tích hợp các công cụ kiểm tra lỗi DRC và ERC, giúp phát hiện các vấn đề trong thiết kế như ngắn mạch, sai sót trong kết nối hoặc các lỗi về điện. Những kiểm tra này giúp đảm bảo rằng thiết kế đáp ứng các quy định kỹ thuật và không có lỗi nghiêm trọng trước khi sản xuất PCB.
5. **Mô phỏng mạch và kiểm tra**: Altium cung cấp các công cụ mô phỏng để kiểm tra hoạt động của mạch trong điều kiện thực tế, giúp xác định các vấn đề tiềm ẩn trước khi thực hiện sản xuất. Công cụ này rất hữu ích trong việc kiểm tra các mạch phức tạp hoặc các hệ thống nhúng như ESP32, khi mà việc thử nghiệm trực tiếp trên phần cứng có thể gặp nhiều khó khăn.
6. **Thư viện linh kiện phong phú và dễ dàng quản lý**: Altium Designer cung cấp một thư viện linh kiện điện tử phong phú với các thành phần phổ biến trong ngành công nghiệp. Các linh kiện có sẵn trong thư viện có thể được sử dụng ngay lập tức, hoặc người dùng có thể tự tạo thêm linh kiện nếu cần. Thư viện này giúp tiết kiệm thời gian trong quá trình thiết kế và đảm bảo tính chính xác của các linh kiện trong thiết kế.
7. **Chế độ xem 3D và khả năng xuất tệp Gerber**: Altium Designer cho phép người dùng xem mạch PCB dưới dạng mô hình 3D, giúp hình dung rõ ràng hơn về cách các linh kiện sẽ được lắp ráp trên PCB. Bên cạnh đó, phần mềm còn hỗ trợ xuất tệp Gerber và các tài liệu sản xuất khác, giúp chuyển giao thiết kế cho nhà sản xuất PCB.

### 1.3.3. Flutter

Flutter là một framework phát triển ứng dụng di động (cross-platform) nổi bật, hỗ trợ việc xây dựng ứng dụng cho cả hai hệ điều hành phổ biến là Android và iOS. Flutter được Google phát triển với mục tiêu mang đến khả năng phát triển ứng dụng với hiệu suất cao và giao diện người dùng đẹp mắt, trong khi vẫn đảm bảo tiết kiệm thời gian và chi phí phát triển.

A grey text on a white background

Description automatically generated

Flutter sử dụng ngôn ngữ lập trình Dart, cung cấp một bộ công cụ mạnh mẽ cho việc phát triển ứng dụng với tốc độ nhanh và khả năng tạo giao diện người dùng (UI) tinh vi, dễ dàng thay đổi và tối ưu hóa. Điều này đặc biệt quan trọng trong việc xây dựng các ứng dụng yêu cầu tính năng động và giao diện người dùng mượt mà, như trong các ứng dụng giám sát sức khỏe.

**Lợi ích khi sử dụng Flutter**

1. **Phát triển ứng dụng cross-platform**: Flutter cho phép lập trình viên phát triển một ứng dụng duy nhất có thể chạy trên cả iOS và Android, điều này giúp tiết kiệm thời gian và chi phí phát triển. Việc sử dụng một mã nguồn chung còn giảm bớt sự phức tạp khi cần duy trì nhiều phiên bản khác nhau của ứng dụng cho từng hệ điều hành.
2. **Hiệu suất cao**: Một trong những điểm mạnh của Flutter là hiệu suất vượt trội so với các framework cross-platform khác như React Native. Flutter biên dịch trực tiếp sang mã máy (native code) cho phép ứng dụng chạy với hiệu suất gần như tương đương với ứng dụng native. Điều này đặc biệt quan trọng trong các ứng dụng cần xử lý nhiều dữ liệu hoặc yêu cầu độ mượt mà cao, chẳng hạn như ứng dụng giám sát sức khỏe.
3. **Hot Reload**: Tính năng Hot Reload của Flutter cho phép lập trình viên thấy ngay lập tức sự thay đổi khi cập nhật mã nguồn mà không cần phải khởi động lại ứng dụng. Điều này không chỉ giúp rút ngắn thời gian phát triển mà còn giúp kiểm tra, thử nghiệm và điều chỉnh giao diện người dùng nhanh chóng.
4. **Tích hợp dễ dàng với các thư viện và API**: Flutter cung cấp các thư viện mạnh mẽ để làm việc với giao thức HTTP, kết nối Bluetooth, WebSockets và các API khác, giúp việc thu thập và hiển thị dữ liệu từ các cảm biến như MAX30102 hoặc hệ thống theo dõi sức khỏe trở nên dễ dàng. Ngoài ra, Flutter hỗ trợ tích hợp với các nền tảng như Firebase, giúp lưu trữ và quản lý dữ liệu trực tuyến một cách hiệu quả.
5. **Giao diện người dùng (UI) đẹp mắt và tùy chỉnh**: Flutter đi kèm với một bộ widget phong phú, cho phép tạo ra các giao diện người dùng rất linh hoạt và đẹp mắt. Điều này đặc biệt quan trọng trong các ứng dụng giám sát sức khỏe, nơi mà người dùng cần dễ dàng truy cập thông tin qua giao diện trực quan và dễ sử dụng. Flutter hỗ trợ phát triển các giao diện với kiểu dáng và màu sắc tùy chỉnh, cho phép người dùng có trải nghiệm sử dụng linh hoạt.

**Ứng dụng Flutter trong phát triển hệ thống giám sát sức khỏe**

Trong đề tài nghiên cứu, Flutter sẽ được sử dụng để phát triển ứng dụng di động kết nối với hệ thống giám sát sức khỏe từ xa, nhận dữ liệu từ các cảm biến như MAX30102 (đo nhịp tim và nồng độ oxy trong máu) qua ESP32. Cụ thể:

* + **Thông báo và cảnh báo**: Với Flutter, việc tích hợp thông báo push sẽ giúp người dùng nhận được cảnh báo tức thì khi có sự thay đổi đột ngột trong các chỉ số sức khỏe (ví dụ như nhịp tim quá cao hoặc nồng độ oxy trong máu quá thấp).
  + **Dữ liệu theo thời gian thực**: Ứng dụng có thể nhận dữ liệu theo thời gian thực từ hệ thống, hiển thị các biểu đồ hoặc thông số thống kê về tình trạng sức khỏe trong suốt một khoảng thời gian nhất định, giúp người dùng theo dõi sức khỏe của mình một cách dễ dàng.
  + **Tính năng báo cáo và lưu trữ dữ liệu**: Flutter có thể kết nối với các dịch vụ lưu trữ dữ liệu đám mây (ví dụ như Firebase) để lưu trữ lịch sử các chỉ số sức khỏe, từ đó giúp người dùng hoặc bác sĩ có thể tham khảo lại dữ liệu trong các cuộc kiểm tra sức khỏe sau này.

# CHƯƠNG 2 – CƠ SỞ LÝ THUYẾT

## Giới thiệu chung về ESP32

### 2.1.1 Cấu Trúc của ESP32

**ESP32** là một vi xử lý mạnh mẽ, tích hợp Wi-Fi và Bluetooth, được thiết kế cho các ứng dụng IoT. Cấu trúc của ESP32 bao gồm các thành phần chính sau:

**+Bộ Xử Lý (CPU)**: ESP32 sử dụng **dual-core** với **Tensilica Xtensa LX6** 32-bit, hoạt động ở tần số tối đa 240 MHz, giúp xử lý các tác vụ đa nhiệm và yêu cầu tính toán cao.

**+Bộ Nhớ (Memory)**:

* + **RAM**: Khoảng 520 KB - 512 KB (tùy phiên bản).
  + **Flash Memory**: Lưu trữ chương trình và dữ liệu người dùng (4MB - 16MB).
  + **ROM**: Lưu trữ các chương trình khởi động.

**+Kết Nối Không Dây**:

* + **Wi-Fi**: Hỗ trợ chuẩn **IEEE 802.11 b/g/n** (2.4 GHz).
  + **Bluetooth**: Hỗ trợ **Bluetooth Classic** và **Bluetooth Low Energy (BLE)**.

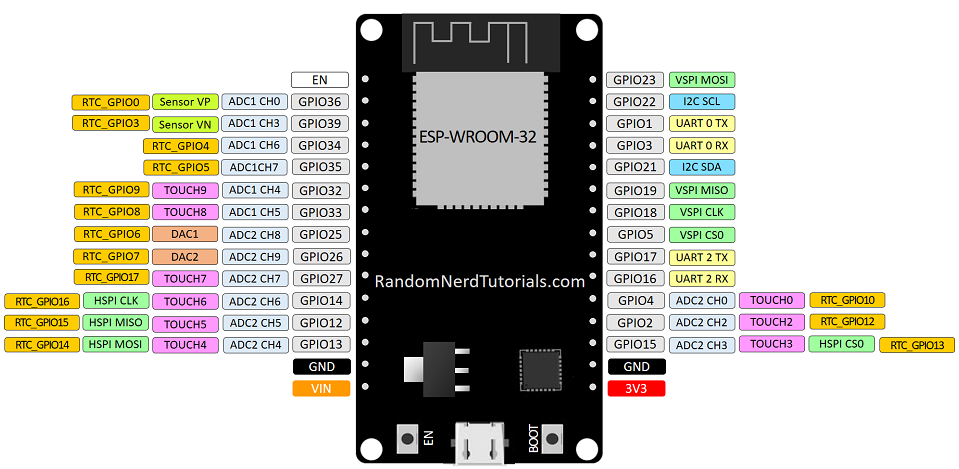
**+I/O và Giao Tiếp**:

* + **GPIO**: Khoảng 34 chân có thể cấu hình cho các ứng dụng cảm biến, điều khiển thiết bị.
  + Hỗ trợ các giao thức như **UART, SPI, I2C** cho kết nối với các cảm biến và thiết bị ngoại vi.

**+Tiết Kiệm Năng Lượng**:

* + **Chế độ Deep Sleep** và **Light Sleep** giúp tiết kiệm năng lượng, phù hợp với các ứng dụng sử dụng pin.

**+Cảm Biến và GPS**: Một số phiên bản của ESP32 hỗ trợ kết nối với cảm biến và định vị GPS, lý tưởng cho các ứng dụng theo dõi vị trí.



Hình 2. 1: Datasheet ESP32

### 2.1.2 Nguyên Lý Hoạt Động

**1. Xử Lý Tín Hiệu và Lệnh**

ESP32 sử dụng bộ vi xử lý **dual-core** (2 lõi xử lý) với nền tảng **Tensilica Xtensa LX6**, cho phép xử lý các lệnh song song và đa nhiệm. Bộ vi xử lý này thực hiện các tác vụ tính toán, xử lý tín hiệu từ cảm biến hoặc các thiết bị ngoại vi, và thực thi các lệnh phần mềm được lập trình trước.

**2. Kết Nối Không Dây (Wi-Fi và Bluetooth)**

**- Wi-Fi**: ESP32 kết nối với mạng không dây thông qua chuẩn **IEEE 802.11 b/g/n**. Khi được cấu hình, ESP32 có thể kết nối với các mạng Wi-Fi để truyền và nhận dữ liệu, ví dụ như gửi thông tin sức khỏe từ cảm biến đến một máy chủ hoặc ứng dụng di động.

**- Bluetooth**: ESP32 hỗ trợ **Bluetooth Classic** và **Bluetooth Low Energy (BLE)**. Với BLE, ESP32 có thể giao tiếp với các thiết bị Bluetooth khác như điện thoại di động hoặc các cảm biến theo dõi sức khỏe, với mức tiêu thụ năng lượng thấp.

**3. Quản Lý Dữ Liệu và Cảm Biến**

ESP32 có thể thu thập và xử lý dữ liệu từ các cảm biến qua các chân **GPIO** hoặc qua các giao thức như **I2C** hoặc **SPI**. Ví dụ, nó có thể đọc tín hiệu từ cảm biến nhịp tim, nhiệt độ, hoặc độ ẩm, và xử lý chúng để hiển thị thông tin sức khỏe cho người dùng.

**4. Điều Khiển Các Thiết Bị Ngoại Vi**

ESP32 có thể điều khiển các thiết bị như động cơ, đèn LED, hoặc còi báo động thông qua các tín hiệu **PWM (Pulse Width Modulation)**. Điều này cho phép ESP32 ứng dụng trong các hệ thống tự động hóa hoặc cảnh báo.

**5. Tiết Kiệm Năng Lượng**

ESP32 có các chế độ **Deep Sleep** và **Light Sleep** để tiết kiệm năng lượng. Khi không cần hoạt động, ESP32 có thể vào chế độ ngủ sâu, chỉ giữ lại các chức năng cần thiết như bộ định thời (timer) hoặc giao tiếp Bluetooth, giúp tiết kiệm năng lượng cho các ứng dụng sử dụng pin.

**6. Đồng Bộ và Lưu Trữ Dữ Liệu**

Thông qua **Flash memory** và **Firebase** (hoặc các dịch vụ đám mây khác), ESP32 có thể lưu trữ và đồng bộ hóa dữ liệu từ các cảm biến, đảm bảo dữ liệu luôn được cập nhật và lưu trữ trong thời gian thực, sẵn sàng để người dùng hoặc hệ thống quản lý truy xuất.

### 2.1.3 Ứng Dụng của ESP32

**Internet of Things (IoT)**

ESP32 là một trong những vi điều khiển phổ biến nhất trong các dự án IoT, nhờ vào khả năng kết nối Wi-Fi và Bluetooth tích hợp. Nó có thể dễ dàng giao tiếp với các cảm biến, bộ điều khiển, và hệ thống đám mây để thu thập và chia sẻ dữ liệu từ xa. Ví dụ:

* **Hệ thống giám sát môi trường**: ESP32 có thể kết nối với cảm biến nhiệt độ, độ ẩm, CO2, hoặc bụi mịn để thu thập dữ liệu và gửi đến một máy chủ hoặc ứng dụng trên điện thoại thông qua Wi-Fi hoặc Bluetooth.
* **Hệ thống điều khiển ánh sáng thông minh**: ESP32 có thể điều khiển các đèn LED hoặc hệ thống chiếu sáng thông qua một ứng dụng di động

**Nhà thông minh (Smart Home)**

ESP32 có thể được sử dụng để phát triển các ứng dụng nhà thông minh, nơi người dùng có thể điều khiển các thiết bị trong nhà như đèn, quạt, hệ thống điều hòa nhiệt độ, hoặc các thiết bị gia dụng khác thông qua smartphone hoặc giọng nói. Một số ví dụ:

* **Điều khiển cửa thông minh**: ESP32 có thể kết nối với một cơ cấu khóa cửa điện tử, cho phép người dùng mở khóa từ xa thông qua một ứng dụng di động hoặc qua các lệnh giọng nói.
* **Hệ thống báo động an ninh thông minh**: ESP32 có thể được tích hợp với cảm biến chuyển động, camera an ninh, và các thiết bị khác để giám sát và gửi cảnh báo khi có sự xâm nhập

**Thiết bị đeo và giám sát sức khỏe**

ESP32 cũng được sử dụng trong các thiết bị đeo như đồng hồ thông minh hoặc các thiết bị giám sát sức khỏe. Chúng có thể đo các chỉ số sức khỏe như nhịp tim, mức độ oxy trong máu, nhiệt độ cơ thể và truyền tải dữ liệu này đến điện thoại hoặc máy tính:

* **Đồng hồ thông minh**: ESP32 có thể được sử dụng để phát triển đồng hồ thông minh có tính năng đo nhịp tim, theo dõi giấc ngủ, hoặc kết nối với các thiết bị IoT khác trong hệ sinh thái nhà thông minh.
* **Giám sát sức khỏe từ xa**: Các thiết bị sử dụng ESP32 có thể kết nối với các cảm biến sinh học (như cảm biến ECG, cảm biến SpO2) để theo dõi các chỉ số sức khỏe và gửi dữ liệu này đến bác sĩ hoặc hệ thống quản lý sức khỏe từ xa.

**Hệ thống tự động hóa công nghiệp**

Trong các ứng dụng tự động hóa công nghiệp, ESP32 có thể giúp giám sát các yếu tố như nhiệt độ, độ ẩm, và các tín hiệu đầu vào từ các cảm biến trong môi trường sản xuất. Các ví dụ bao gồm:

* **Hệ thống giám sát và điều khiển từ xa**: ESP32 có thể kết nối với các cảm biến nhiệt độ, áp suất, và độ ẩm trong các dây chuyền sản xuất và gửi dữ liệu về trung tâm điều khiển, giúp giám sát tình trạng của các máy móc hoặc dây chuyền.
* **Điều khiển động cơ và máy móc**: ESP32 có thể điều khiển các động cơ hoặc máy móc trong môi trường công nghiệp thông qua tín hiệu điều khiển, giúp tự động hóa các quy trình sản xuất.

**Robot và xe tự lái**

ESP32 có thể được sử dụng trong các dự án chế tạo robot hoặc xe tự lái, nhờ khả năng kết nối với cảm biến và các thiết bị ngoại vi khác. Ví dụ:

* **Robot tự động**: ESP32 có thể được tích hợp với các cảm biến khoảng cách, cảm biến âm thanh, và các cảm biến khác để giúp robot di chuyển và thực hiện các nhiệm vụ tự động trong môi trường cụ thể.
* **Xe tự lái**: ESP32 có thể được sử dụng để kết nối và điều khiển các cảm biến như camera, cảm biến LIDAR, và các hệ thống định vị GPS trong xe tự lái.

**Thiết bị giải trí và trò chơi**

ESP32 cũng được ứng dụng trong các thiết bị giải trí như các trò chơi điện tử hoặc các thiết bị âm thanh thông minh. Các ví dụ bao gồm:

* **Điều khiển trò chơi**: ESP32 có thể được sử dụng để phát triển các bộ điều khiển trò chơi không dây, kết nối với các thiết bị thông qua Bluetooth.
* **Hệ thống âm thanh không dây**: ESP32 có thể được sử dụng trong các hệ thống âm thanh thông minh, cho phép điều khiển từ xa qua smartphone hoặc tích hợp với các thiết bị gia đình thông minh khác.



Hình 2. 2: Các loại ESP32

### 2.1.4 Lợi Ích của Việc Sử Dụng ESP32

* Tiết Kiệm Năng Lượng

Với chế độ Deep Sleep, ESP32 tiết kiệm năng lượng, giúp kéo dài tuổi thọ pin cho các thiết bị đeo hoặc cảm biến trong các hệ thống giám sát sức khỏe, đặc biệt trong các ứng dụng yêu cầu hoạt động lâu dài như theo dõi sức khỏe 24/7.

* Khả Năng Xử Lý Dữ Liệu Tín Hiệu Analog

ESP32 có khả năng xử lý tín hiệu analog từ các cảm biến sinh học như cảm biến nhịp tim, SpO2, nhiệt độ cơ thể, và chuyển đổi chúng thành tín hiệu số để phân tích và lưu trữ, giúp theo dõi và giám sát sức khỏe người dùng.

* Độ Chính Xác Cao và Đa Dạng Cảm Biến

ESP32 hỗ trợ kết nối với nhiều loại cảm biến khác nhau như Max30102 (đo nhịp tim và SpO2), cảm biến nhiệt độ, điện tâm đồ (ECG), và nhiều loại cảm biến sinh học khác, giúp xây dựng một hệ thống giám sát sức khỏe toàn diện.

* Cập Nhật và Cảnh Báo Thời Gian Thực

Với khả năng kết nối internet qua Wi-Fi, ESP32 có thể gửi dữ liệu sức khỏe lên các dịch vụ đám mây hoặc ứng dụng di động, giúp bác sĩ hoặc người dùng nhận thông tin sức khỏe theo thời gian thực và có thể cảnh báo khi có dấu hiệu bất thường.

* Tích Hợp với Các Hệ Thống Y Tế

ESP32 có khả năng tích hợp với các hệ thống phần mềm y tế hiện đại, giúp đồng bộ hóa và phân tích dữ liệu từ nhiều nguồn (cảm biến, thiết bị y tế) để đưa ra các cảnh báo hoặc chẩn đoán chính xác hơn

## Fluter

**Flutter và Khả Năng Xây Dựng Giao Diện Người Dùng (UI)**

**Flutter** là một framework phát triển ứng dụng di động mạnh mẽ được phát triển bởi Google, sử dụng ngôn ngữ lập trình Dart. Flutter cho phép xây dựng các ứng dụng có giao diện người dùng mượt mà và đẹp mắt cho các nền tảng di động như Android và iOS, với mã nguồn duy nhất.

Trong lĩnh vực giám sát sức khỏe, Flutter có thể được sử dụng để hiển thị các chỉ số sức khỏe như nhịp tim, mức độ bão hòa oxy (SpO2), huyết áp, và các chỉ số khác. Flutter cung cấp một loạt các widget mạnh mẽ và linh hoạt, giúp xây dựng giao diện người dùng trực quan và dễ sử dụng.

**Quản Lý Dữ Liệu Sức Khỏe**

Ứng dụng giám sát sức khỏe thường phải thu thập dữ liệu từ các cảm biến sức khỏe hoặc thiết bị ngoại vi, như máy đo nhịp tim, cảm biến bão hòa oxy, hoặc các thiết bị theo dõi sức khỏe đeo tay. Flutter hỗ trợ tích hợp với các cảm biến qua các plugin và thư viện, cho phép lấy dữ liệu từ các thiết bị Bluetooth hoặc cảm biến trên điện thoại.

Các ứng dụng giám sát sức khỏe cũng có thể lưu trữ và quản lý dữ liệu qua các nền tảng đám mây như **Firebase**, giúp đồng bộ hóa dữ liệu giữa các thiết bị và cung cấp thông tin sức khỏe cho người dùng bất cứ lúc nào.

**Hiển Thị Thông Tin Sức Khỏe Trên Giao Diện**

Để hiển thị các chỉ số sức khỏe, Flutter sử dụng các widget linh hoạt như **Text**, **ListView**, và **Chart**. Các chỉ số sức khỏe có thể được trình bày dưới dạng các biểu đồ, bảng thống kê, hoặc các chỉ số động cập nhật theo thời gian thực.

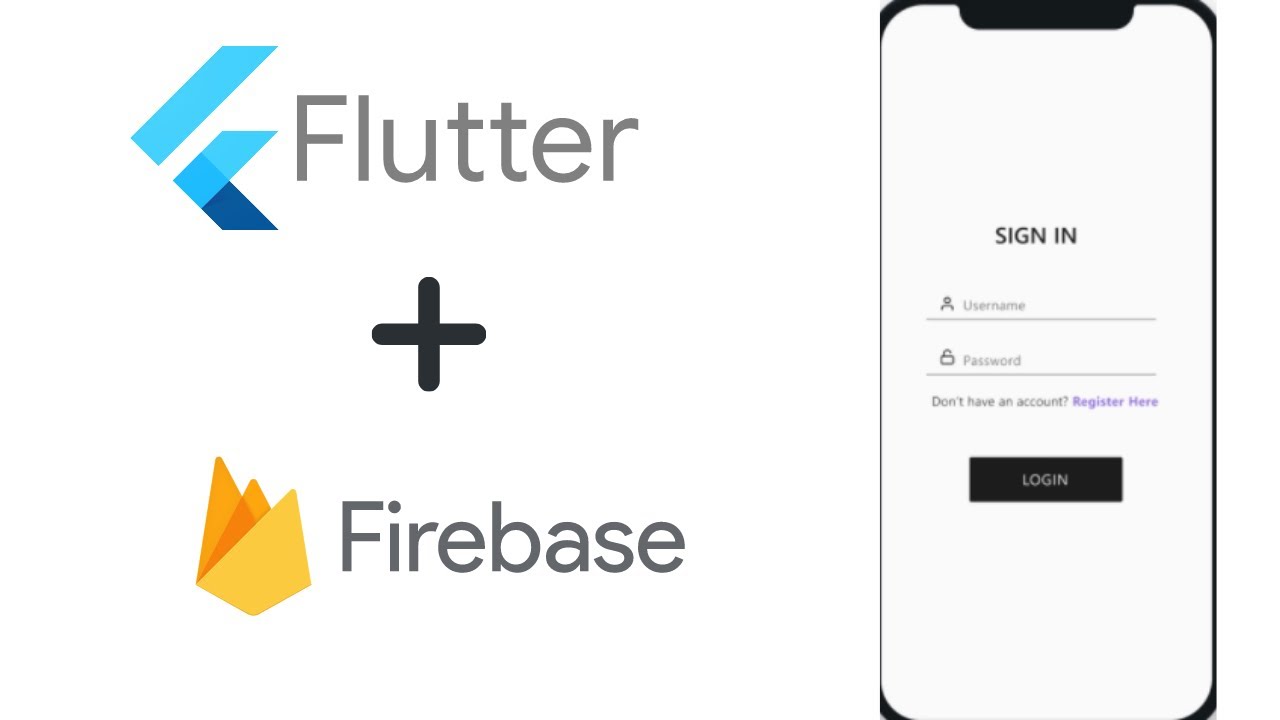
Flutter hỗ trợ việc tạo ra giao diện có thể tự động cập nhật khi có sự thay đổi dữ liệu, thông qua các tính năng như **StreamBuilder** và **FutureBuilder**, giúp hiển thị dữ liệu sức khỏe mới nhất cho người dùng mà không cần phải làm mới toàn bộ ứng dụng.

**Đồng Bộ Dữ Liệu Sức Khỏe với Firebase**

Firebase là một nền tảng mạnh mẽ cho việc lưu trữ và đồng bộ hóa dữ liệu trong thời gian thực. Với **Firebase Firestore**, các ứng dụng Flutter có thể lưu trữ dữ liệu sức khỏe của người dùng (như nhịp tim, huyết áp, chỉ số SpO2) và đồng bộ hóa chúng với các thiết bị khác trong thời gian thực. Firebase Authentication có thể được sử dụng để quản lý và xác thực người dùng, bảo mật dữ liệu sức khỏe cá nhân.

**Cập Nhật Dữ Liệu Sức Khỏe Theo Thời Gian Thực**

Một tính năng quan trọng của các ứng dụng giám sát sức khỏe là khả năng cập nhật dữ liệu theo thời gian thực. Flutter hỗ trợ việc sử dụng các **Stream** và **Future** để cập nhật giao diện người dùng khi có thay đổi trong dữ liệu. Điều này rất hữu ích khi hiển thị các chỉ số như nhịp tim, mức độ bão hòa oxy, huyết áp, giúp người dùng theo dõi tình trạng sức khỏe của mình liên tục và kịp thời.



Hình 2. 3: Web Server

# CHƯƠNG 3 – THIẾT KẾ & LẬP TRÌNH HỆ THỐNG

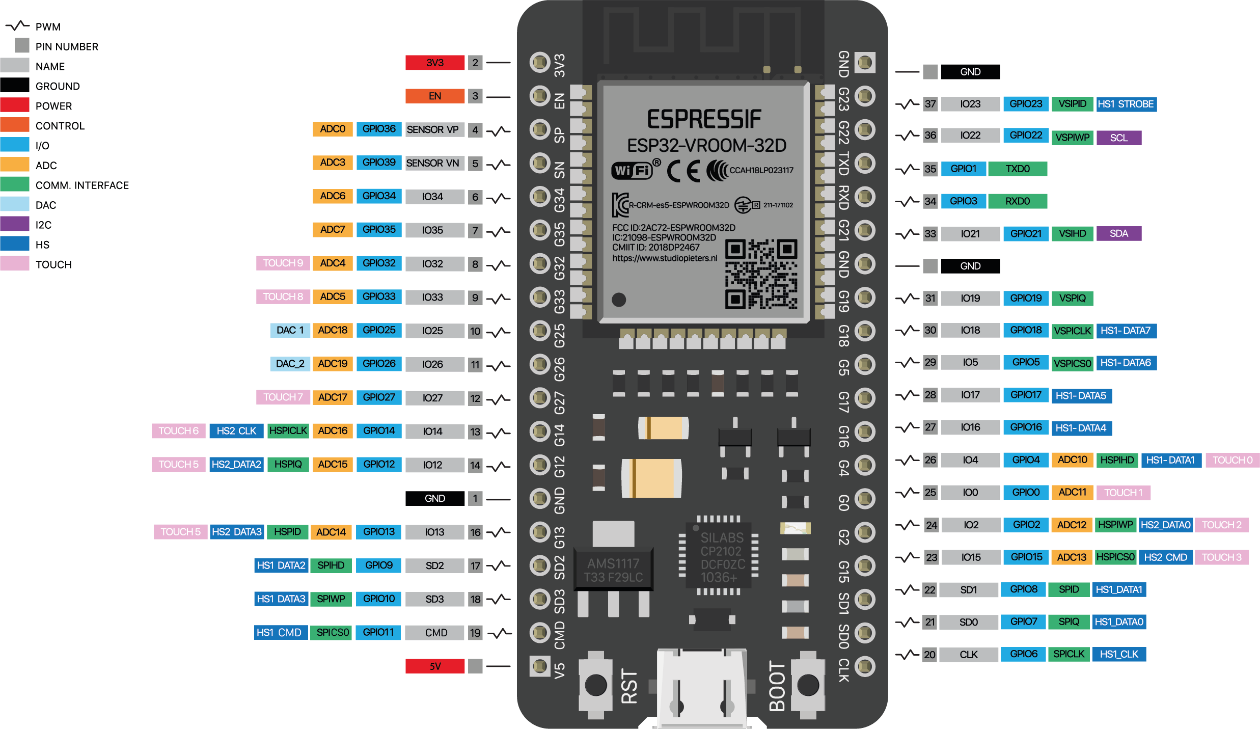
* 1. Các Module và các linh kiện cần có

### 3.1.1. ESP32

ESP32 là một bộ vi điều khiển thuộc danh mục vi điều khiển trên chip công suất thấp và tiết kiệm chi phí. Hầu hết tất cả các biến thể ESP32 đều tích hợp Bluetooth và Wi-Fi chế độ kép, làm cho nó có tính linh hoạt cao, mạnh mẽ và đáng tin cậy cho nhiều ứng dụng.

Nó là sự kế thừa của vi điều khiển NodeMCU ESP8266 phổ biến và cung cấp hiệu suất và tính năng tốt hơn. Bộ vi điều khiển ESP32 được sản xuất bởi Espressif Systems và được sử dụng rộng rãi trong nhiều ứng dụng khác nhau như IoT, robot và tự động hóa.

ESP32 cũng được thiết kế để tiêu thụ điện năng thấp, lý tưởng cho các ứng dụng chạy bằng pin. Nó có hệ thống quản lý năng lượng cho phép nó hoạt động ở chế độ ngủ và chỉ thức dậy khi cần thiết, điều này có thể kéo dài tuổi thọ pin rất nhiều.



Hình 3. 1: ESP32

### 3.1.2. Module MAX30102

**MAX30102** là một module cảm biến quang học chuyên dụng để đo các chỉ số sức khỏe như **nhịp tim (Heart Rate - HR)** và **nồng độ oxy trong máu (Blood Oxygen Saturation - SpO2)**. Nó sử dụng công nghệ quang học, cụ thể là **phương pháp phân tích quang học phản xạ** để đo lường các thông số sinh học này, với độ chính xác và khả năng tiêu thụ năng lượng thấp.

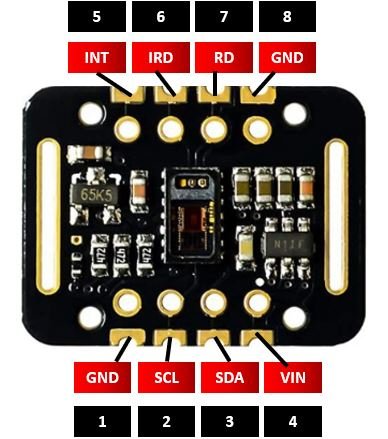
**Cấu Tạo và Thành Phần**

**LEDs**: MAX30102 sử dụng hai loại LED để phát ánh sáng vào da, từ đó đo được độ phản xạ ánh sáng. Các LED này bao gồm:

* **LED đỏ** (Red LED): Sử dụng để đo lượng oxy trong máu (SpO2).
* **LED hồng ngoại (IR LED)**: Sử dụng để đo nhịp tim và giúp phân biệt giữa oxy và các phân tử khác trong máu.

**Cảm Biến Quang Học**: Được sử dụng để nhận ánh sáng phản xạ từ các mạch máu dưới da, giúp đo được nhịp tim và SpO2. Cảm biến này rất nhạy và có thể đo được các tín hiệu mờ từ máu.

**Mạch Khuếch Đại Tín Hiệu**: Để đảm bảo tín hiệu thu được đủ mạnh để xử lý, MAX30102 sử dụng mạch khuếch đại tín hiệu để làm rõ sự biến đổi của ánh sáng phản xạ.



Hình 3. 2: Cảm biến nhiệt độ DS18B20

### 3.1.3. Màn hình OLED I2C 0,96 inch

màn hình Oled 0.96inch sử dụng giao tiếp I2C cho chất lượng đường truyền ổn định và rất dễ giao tiếp chỉ với 2 chân GPIO.

Sơ đồ chân:

- VCC: là nguồn cấp cho màn hình có thể là 3.3V hoặc 5V

- GND: chân nối đất

- SCL: chân xung clock

- SDA: chân dữ liệu nối tiếp

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm

Mô tả được tạo tự động

Hình 3. 3: Màn hình OLED I2C

### 3.1.4. Còi Buzzer

Còi **buzzer** là một thiết bị điện tử phát ra âm thanh khi có tín hiệu điện cung cấp cho nó. Buzzer thường được sử dụng trong các ứng dụng cảnh báo, thông báo, hoặc tín hiệu âm thanh, ví dụ như trong các hệ thống báo động, thiết bị điện tử, và các ứng dụng IoT.

**Kết Nối và Điều Khiển**

**Điện Áp Hoạt Động**: Thông thường, buzzer hoạt động ở các mức điện áp từ 3V đến 5V (tùy loại). Còi buzzer chủ động có thể hoạt động trực tiếp từ nguồn điện, trong khi buzzer thụ động cần một tín hiệu điều khiển (PWM hoặc sóng vuông).

**Chân Kết Nối**: Buzzer có hai chân cơ bản:

* **Chân Dương (VCC)**: Kết nối với nguồn dương (thường là 3.3V hoặc 5V).
* **Chân Âm (GND)**: Kết nối với đất.
* Nếu là buzzer thụ động, chân điều khiển có thể kết nối với vi điều khiển thông qua một tín hiệu PWM (Pulse Width Modulation) để điều khiển tần số âm thanh.



Hình 3. 4: Relay

* 1. Thiết kế phần cứng

### 3.2.1. Vẽ schematic trên Altium

Sơ đồ nguyên lý mô tả mạch điện sử dụng **ESP32 WROOM 32D V1.1**, module cảm biến nhịp tim và oxy trong máu **MAX30102**, màn hình hiển thị **OLED 0.96 inch** và một **loa/buzzer** để báo hiệu. Các linh kiện được kết nối thông qua giao thức **I2C** và GPIO.

A diagram of a circuit board

Description automatically generated

Hình 3. 5: Schematic

**ESP32 WROOM 32D**

* **Vai trò:** Là vi điều khiển chính của hệ thống, nhận dữ liệu từ cảm biến và hiển thị trên màn hình OLED, đồng thời điều khiển loa báo hiệu.
* **Nguồn cấp:** Pin 3.3V được sử dụng để cấp nguồn cho ESP32.
* **Giao tiếp:**

**D21 (SDA)** và **D22 (SCL):** Giao tiếp I2C với MAX30102 và OLED.

**D15:** Điều khiển loa/buzzer.

**Cảm biến MAX30102**

* **Vai trò:** Đo nhịp tim và độ bão hòa oxy trong máu (SpO2).
* **Nguồn cấp:**

Chân **VCC** được kết nối với nguồn 3.3V.

Chân **GND** được nối đất.

* **Giao tiếp:**

**SDA:** Nối với chân D21 của ESP32.

**SCL:** Nối với chân D22 của ESP32.

* **Chân INT (ngõ ra ngắt):** Không được sử dụng trong sơ đồ này.
* **Chân RD (nạp điện trở):** Được đấu với đất để đảm bảo hoạt động ổn định.

**Màn hình OLED 0.96 inch**

* **Vai trò:** Hiển thị dữ liệu đo từ cảm biến MAX30102 (như nhịp tim và SpO2).
* **Nguồn cấp:**

Chân **VCC** được kết nối với nguồn 3.3V.

Chân **GND** được nối đất.

* **Giao tiếp:** Giao thức I2C thông qua chân:

**SDA (chân 5):** Kết nối với D21 trên ESP32.

**SCL (chân 6):** Kết nối với D22 trên ESP32.

**Loa/Buzzer**

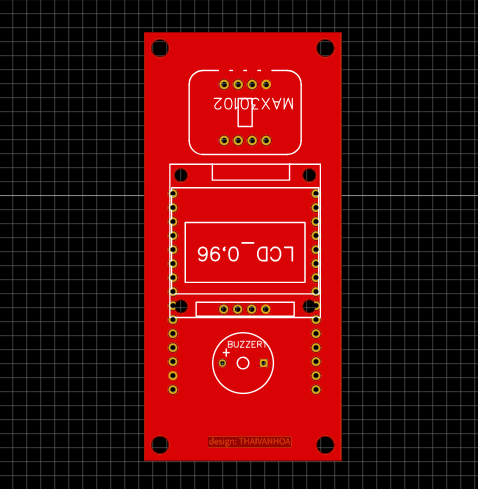
* **Vai trò:** Phát âm thanh cảnh báo khi phát hiện các bất thường hoặc sự kiện cần thông báo.
* **Nguồn cấp:**

Chân 1 của loa nối với GPIO D15 trên ESP32.

Chân 2 nối đất.

* **Nguyên lý hoạt động:** Loa được kích hoạt bởi tín hiệu HIGH từ chân D15 của ESP32.

### 3.2.2. Vẽ mạch PCB



Hình 3. 6: Mạch PCB ở lớp Top layer

* Lớp Top Layer được sử dụng để định vị và đặt các linh kiện trên bề mặt của mạch PCB. Các linh kiện bao gồm vi điều khiển, resistor, capacitor, ICs và các linh kiện khác.
* Đặt và Định Vị Linh Kiện: Lớp Top Layer được sử dụng để đặt và định vị các linh kiện trên bề mặt của mạch PCB. Điều này bao gồm việc xác định vị trí và hướng của các linh kiện, bố trí chúng sao cho phù hợp với yêu cầu kỹ thuật và thiết kế.
* Tạo Vias và Pads: Lớp Top Layer cũng chứa các pads (đế linh kiện) và vias (lỗ thông) để tạo kết nối giữa các linh kiện trên cùng một lớp và giữa các lớp khác nhau của mạch PCB.
* Tạo Vùng Mạch và Trở Kháng: Trên lớp này, vùng mạch (copper pour) được tạo ra để cung cấp đường dẫn điện hoặc để kiểm soát trở kháng của dây dẫn. Các vùng mạch cũng có thể được sử dụng để tạo ra một môi trường đất tốt hơn cho mạch.

A blue circuit board with yellow dots and wires

Description automatically generated

Hình 3. 7: Mạch PCB ở lớp Bottom layer

* Lớp bottom layer dùng để hàn các linh kiện điện tử vào mạch PCB
* **Lớp bottom (mặt dưới)** chứa các đường mạch (trace) kết nối giữa các chân GPIO của module ESP32 với các chân header mở rộng.
* Đây là lớp dây dẫn đồng được định tuyến cẩn thận để đảm bảo kết nối tín hiệu ổn định và giảm nhiễu..

A blue rectangular object with white text

Description automatically generated

Hình 3. 8: Mạch PCB 3D

* Dùng để hiển thị các bộ phận bố cục của mạch và các linh kiện cần có để dễ hình dung và lắp ráp mạch .
  1. Cài phần mềm để chạy code micropython

### 3.3.1. Cài đặt Thony

**1. Chuẩn bị**

**- Thonny IDE**: Cài đặt Thonny theo hướng dẫn ở trên ([thonny.org](https://thonny.org/)).

**- ESP32 Board**: Kết nối ESP32 với máy tính qua cáp USB.

**- Driver USB to UART**: Đảm bảo driver đã được cài đặt để máy tính nhận diện được ESP32 (CP2102 hoặc CH340).

**2. Flash Firmware bằng Thonny**

**+ Kết nối ESP32 với máy tính**:

* Kết nối ESP32 qua cáp USB.
* Mở Thonny IDE.

**+ Cấu hình Thonny cho MicroPython**:

* Trong Thonny, đi tới **Tools > Options > Interpreter**.
* Tại mục **Interpreter**, chọn:

**MicroPython (ESP32)**.

**Port**: Chọn cổng COM tương ứng với ESP32 (ví dụ: COM3 trên Windows hoặc /dev/ttyUSBx trên Linux/Mac).

**+ Flash firmware trực tiếp**:

* Vào **Tools > Manage devices**.
* Nhấn **Install or update firmware**.
* Hộp thoại **Install firmware** xuất hiện:

**Port**: Chọn cổng kết nối của ESP32.

**Firmware**: Nhấn **Install the latest MicroPython firmware**.

Thonny sẽ tự động tải firmware phù hợp cho ESP32 từ trang MicroPython.

* Nhấn **Install** và chờ quá trình cài đặt hoàn tất.
  1. Mô phỏng thực tế hệ thống

### 3.4.1. Sơ dồ mạch

A circuit board with wires

Description automatically generated

Hình 3. 9: Sơ đồ mạch

### 3.4.2. Mô phỏng testboard

**A circuit board with wires and a wire

Description automatically generated**

Hình 3. 10: Mô phỏng hệ thống trên testboard

* 1. Sơ đồ và cách thức hoạt động của hệ thống

### 3.5.1. Sơ đồ hoạt động của hệ thống

A diagram of a process

Description automatically generated

Hình 3. 11: Sơ đồ hoạt động hệ thống

### 3.5.2. Cách thức hoạt động của hệ thống

Hệ thống hoạt động bằng cách thu thập dữ liệu từ cảm biến MAX30102, một cảm biến quang học chuyên dùng để đo nhịp tim (Heart Rate) và nồng độ oxy trong máu (SpO2). Cảm biến này sử dụng đèn LED và photodiode để phát hiện sự thay đổi ánh sáng xuyên qua mạch máu, sau đó gửi tín hiệu đo được đến vi điều khiển ESP32 qua giao thức I2C. Dữ liệu nhận được sẽ được ESP32 xử lý, bao gồm lọc nhiễu và tính toán các chỉ số quan trọng như nhịp tim và SpO2. Kết quả sau khi xử lý sẽ được hiển thị trực tiếp trên màn hình OLED để người dùng theo dõi thời gian thực, đồng thời kích hoạt buzzer để phát âm thanh cảnh báo khi các chỉ số vượt ngưỡng an toàn.

ESP32 cũng đóng vai trò như một trung tâm truyền tải dữ liệu. Sau khi xử lý, các chỉ số đo lường được gửi lên Firebase Realtime Database thông qua kết nối Wi-Fi. Tại đây, dữ liệu được lưu trữ theo thời gian thực, sẵn sàng để ứng dụng di động truy cập. Ứng dụng trên điện thoại di động lấy dữ liệu từ Firebase thông qua API và hiển thị dưới dạng biểu đồ trực quan, đồng thời cung cấp các thông báo cảnh báo nếu phát hiện bất thường. Hệ thống này hoạt động liên tục, đảm bảo cung cấp dữ liệu sức khỏe chính xác, kịp thời và hỗ trợ người dùng theo dõi tình trạng sức khỏe ở bất cứ đâu.

* 1. Sản phẩm hoàn thiện

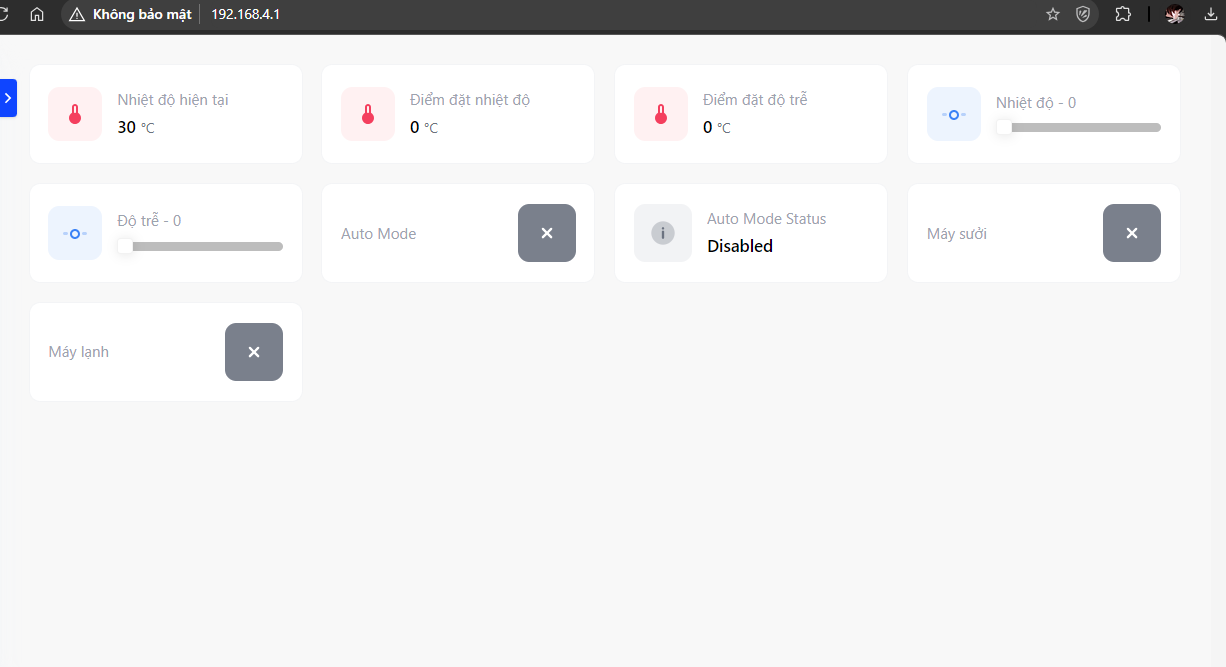
### 3.6.1. Sản phẩm hoàn chỉnh

A circuit board with wires and a black screen

Description automatically generated

Hình 3. 12: Sản phẩm hoàn thiện

### 3.6.2. Giao diện APP



Hình 3. 13: Giao diện web server trên laptop

A screenshot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generated

Hình 3. 14: Giao diện web server trên diện thoại

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

**Kết quả đạt được**

* **Đo lường và hiển thị thông tin chính xác**: Hệ thống đã thu thập và xử lý chính xác các dữ liệu về nhịp tim và SpO2 từ cảm biến MAX30102, sau đó hiển thị các chỉ số này trên màn hình OLED để người dùng dễ dàng theo dõi trạng thái sức khỏe trong thời gian thực.
* **Cảnh báo khi vượt ngưỡng**: Khi các chỉ số nhịp tim hoặc SpO2 vượt quá các ngưỡng đã được xác định (ví dụ: nhịp tim quá cao hoặc SpO2 quá thấp), hệ thống tự động kích hoạt buzzer để cảnh báo người dùng, giúp phát hiện sớm các vấn đề sức khỏe.
* **Truyền tải dữ liệu lên Cloud (Firebase)**: Hệ thống đã thành công trong việc truyền tải dữ liệu sức khỏe lên Firebase thông qua kết nối Wi-Fi. Điều này cho phép người dùng theo dõi và lưu trữ các chỉ số sức khỏe trong thời gian thực.
* **Hiển thị dữ liệu qua ứng dụng di động**: Dữ liệu từ Firebase được ứng dụng di động truy xuất và hiển thị dưới dạng biểu đồ trực quan, giúp người dùng dễ dàng theo dõi tình trạng sức khỏe của mình ở bất cứ đâu.
* **Tính năng lưu trữ và theo dõi lâu dài**: Firebase giúp lưu trữ dữ liệu trong thời gian dài, hỗ trợ người dùng theo dõi sự thay đổi về sức khỏe theo thời gian và đưa ra các phân tích, nhận định cần thiết.
* Mã nguồn : https://github.com/ThaiVanHoa/health-pulse-esp32-app.git

**Hạn chế**

* **Độ chính xác của cảm biến**: Cảm biến MAX30102 có thể bị ảnh hưởng bởi một số yếu tố như ánh sáng môi trường, độ ẩm hoặc tình trạng tiếp xúc không chính xác với da, dẫn đến sai lệch trong việc đo nhịp tim và SpO2. Điều này có thể làm giảm độ chính xác của các phép đo.
* **Giới hạn của kết nối Wi-Fi**: Hệ thống phụ thuộc vào kết nối Wi-Fi để truyền tải dữ liệu lên Firebase. Nếu tín hiệu Wi-Fi yếu hoặc không ổn định, việc truyền tải dữ liệu sẽ gặp khó khăn, dẫn đến mất mát thông tin hoặc gián đoạn trong quá trình theo dõi sức khỏe.
* **Tầm xa của buzzer**: Mặc dù buzzer có thể phát ra cảnh báo khi các chỉ số vượt ngưỡng, nhưng tầm phát âm của buzzer trong môi trường ồn ào có thể không đủ để người dùng nghe thấy cảnh báo, đặc biệt là trong các tình huống khẩn cấp.
* **Thiếu tính năng phân tích nâng cao**: Hệ thống hiện chỉ cung cấp dữ liệu cơ bản như nhịp tim và SpO2 mà không có các tính năng phân tích hoặc dự đoán các vấn đề sức khỏe tiềm ẩn. Điều này có thể khiến hệ thống thiếu sự hỗ trợ chuyên sâu cho người dùng, như khả năng nhận diện sớm các bệnh lý liên quan đến tim mạch hoặc hô hấp.
* **Hiệu suất và tuổi thọ của cảm biến**: Cảm biến MAX30102 có thể bị giảm hiệu suất theo thời gian sử dụng, do đó việc cần phải kiểm tra và thay thế cảm biến định kỳ là điều cần thiết để đảm bảo độ chính xác và ổn định của các phép đo.
* **Khả năng mở rộng**: Hệ thống hiện tại chỉ đo nhịp tim và SpO2, nếu muốn mở rộng để theo dõi các chỉ số sức khỏe khác như nhiệt độ cơ thể, huyết áp, hoặc điện tâm đồ (ECG), hệ thống sẽ cần phải nâng cấp cả phần cứng lẫn phần mềm.

**Hướng phát triển**

* **Cải thiện độ chính xác**: Sử dụng cảm biến cao cấp hơn hoặc kết hợp nhiều cảm biến để tăng độ chính xác, như cảm biến huyết áp hoặc ECG.
* **Phân tích dữ liệu nâng cao**: Áp dụng AI và machine learning để phân tích dữ liệu và dự đoán tình trạng sức khỏe, theo dõi xu hướng sức khỏe dài hạn.
* **Cảnh báo thông minh**: Phát triển cảnh báo qua SMS, email hoặc thông báo đẩy khi các chỉ số vượt ngưỡng nguy hiểm. Thêm tính năng cảnh báo theo vị trí hoặc phát hiện thay đổi bất thường.
* **Mở rộng chỉ số sức khỏe**: Theo dõi thêm các chỉ số như huyết áp, ECG và nhiệt độ cơ thể để cung cấp cái nhìn toàn diện về sức khỏe.
* **Kết nối và lưu trữ dữ liệu tối ưu**: Sử dụng công nghệ kết nối mới như 5G hoặc LoRa để cải thiện truyền tải dữ liệu, cũng như tối ưu hóa lưu trữ và phân tích dữ liệu.
* **Tích hợp với nền tảng sức khỏe**: Kết nối với Google Fit, Apple Health để chia sẻ dữ liệu, đồng thời cho phép bác sĩ theo dõi và tư vấn từ xa.
* **Trải nghiệm người dùng tốt hơn**: Cải thiện giao diện ứng dụng và hỗ trợ đa ngôn ngữ.
* **Tiết kiệm năng lượng**: Tối ưu hóa phần cứng và phần mềm để kéo dài tuổi thọ pin, phù hợp với các thiết bị di động hoặc đeo.

# TÀI LIỆU THAM KHẢO

[1] https://docs.espressif.com/

[2] https://docs.wokwi.com/

[3] https://www.arduino.cc/reference/en/libraries/

# PHỤ LỤC