UNIVERSIDAD AUTÓNOMA DEL ESTADO DE QUINTANA ROO

DIVISIÓN DE CIENCIAS, INGENIERÍA Y TECNOLOGÍA

Monitoreo de signos vitales mediante aplicaciones de IoT

TESIS

PARA OBTENER EL GRADO DE

INGENIERO EN SISTEMAS DE ENERGÍA

PRESENTA

Kevin Alberto Estrada

DIRECTOR

Jesús Orifiel Alvarez Ruiz

ASESORES

Set Jubal Castillo Ávila
Freddy Ignacio Chan Puc
Jorge Ovidio Aguilar Aguilar
Emmanuel Torres Montalvo

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE QUINTANA ROO

ÁREA DE TITULACIÓN

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE QUINTANA ROO
DIVISIÓN DE CIENCIAS, INGENIERÍA Y TECNOLOGÍA

CHETUMAL QUINTANA ROO, MÉXICO, OCTUBRE DE 2024

UNIVERSIDAD AUTÓNOMA DEL
ESTADO DE QUINTANA ROO

DIVISIÓN DE CIENCIAS, INGENIERÍA Y TECNOLOGÍA

TRABAJO DE TESIS TITULADO

Monitoreo de signos vitales mediante aplicaciones de IoT

ELABORADO POR

Kevin Alberto Estrada

BAJO SUPERVISIÓN DEL COMITÉ DEL PROGRAMA DE LICENCIATURA Y APROBADO COMO REQUISITO PARCIAL PARA OBTENER EL GRADO DE:

INGENIERO EN SISTEMAS DE ENERGÍA

COMITÉ SUPERVISOR

DIRECTOR: JESUS ORIFIEL ALVAREZ RUIZ
DIRECTOR

SUPERVISOR: SET JUBAL CASTILLO AVILA
SUPERVISOR

SUPERVISOR: FREDDY IGNACIO CHAN PUC
SUPERVISOR

SUPERVISOR SUPLENTE: JORGE OVIDIO AGUILAR AGUILAR
SUPERVISOR

SUPERVISORA SUPLENTE: EMMANUEL TORRES MONTALVO
SUPERVISOR

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE QUINTANA ROO
DIVISIÓN DE CIENCIAS, INGENIERÍA Y TECNOLOGÍA

CHETUMAL QUINTANA ROO, MÉXICO, OCTUBRE DE 2024

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE QUINTANA ROO
ÁREA DE TITULACIÓN

Dedicatoria

Esta tesis está dedicada con sincera gratitud y amor a quienes han sido esenciales en este camino:

- A mi padre, Rigoberto Estrada, cuyos esfuerzos incansables y orientación, que han sido la base de este logro. Su apoyo constante y la provisión de herramientas esenciales han sido fundamentales para mi éxito.
- A mi madre, Perla Estrada, por su ánimo y apoyo, que han proporcionado fuerza y estabilidad inquebrantables.
- A mi hermana, Karina Estrada, por su apoyo constante que ha ofrecido un gran consuelo.
- A mi sobrino, Eliseo Estrada, por su presencia alegre, que ha aportado una inmensa luminosidad a mi vida, incluso a una edad tan tierna.
- A mis abuelos, Jose Vivas y Eva Vivas, por creer en mí, lo que ha impulsado mi determinación para alcanzar mis metas.
- A mi novia, Yasari Novelo, cuyo apoyo continuo, cuidado y motivación han sido cruciales a lo largo de este proceso.

Agradecimientos

Quiero comenzar expresando mi más profundo agradecimiento a Dios, sin quien nada de esto habría sido posible. Él me otorgó las fuerzas, los recursos y la sabiduría necesarios para alcanzar esta significativa meta en mi vida.

A mi familia, por su apoyo incondicional y su fe constante en mí. A mis padres, Rigoberto Estrada y Perla Estrada, por estar siempre disponibles para ayudar en todo lo necesario. A mi hermana, Karina Estrada, por su atención constante para alcanzar mis objetivos. A mis abuelos, Jose Vivas, Eva Vivas, Antonio Vivas y Hercilia Estrada por su creencia inquebrantable en mi potencial. Y a mi tía, Lucinda Estrada del Cid por ofrecer su apoyo en los momentos más críticos.

Un agradecimiento especial a mi novia, Yasari Novelo, una persona extraordinaria en todos los sentidos. Ella ha sido una fuente constante de motivación y apoyo, siempre animándome a seguir adelante con mis metas. Aprecio profundamente su paciencia y los sacrificios que ha hecho para acompañarme en este viaje.

Quiero expresar mi sincero agradecimiento a mi director de tesis, M.M. Jesús Orifiel Alvarez Ruiz, quien ha estado siempre dispuesto a brindar orientación y a facilitar este proceso de manera excepcional. Igualmente, agradezco a mis asesores de tesis, al Dr. Freddy Ignacio Chan Puc y al Mtro. Set Jubal Castillo Ávila, por su ayuda constante y su disponibilidad para resolver cualquier duda.

Un reconocimiento especial al Mtro. Heriberto López Platas y a la Mtra. Gloria del Carmen Bocarando Lobato, quienes me trataron como parte de su familia desde el primer día. Su apoyo incondicional ha sido fundamental para mi crecimiento personal y académico. Espero poder devolverles el favor de alguna manera, tal como ellos hicieron conmigo. Nunca me sentí lejos de casa gracias a su calidez y generosidad.

Finalmente, agradezco a Dyel Acosta por su compañía a lo largo de este trayecto. Se convirtió en un gran amigo durante este proceso y siempre me alentó a seguir adelante.

Índice

CAPÍTULO I-INTRODUCCIÓN 11

1.1 Resumen 12

1.2 Objetivos 13

1.2.1 Objetivo general 13

1.2.2 Objetivos específicos 13

1.3 Justificación 14

1.4 Alcance y limitaciones 14

1.4.1 Alcance 14

1.4.2 Limitaciones 15

CAPÍTULO II- ANTECEDENTES 16

2.1 Estadísticas de problemas cardiovasculares en México y en el mundo 16

2.2 Dispositivos de monitoreo para el ritmo cardiaco 16

2.3 Métodos para la medición de ritmo cardiaco 17

2.4 Monitoreo remoto del ritmo cardiaco 18

CAPÍTULO III- MARCO TEÓRICO 20

3.1 Signos Vitales 20

3.2 NodeMCU ESP-32 20

3.3 Oxímetro 21

3.4 XD-58C 22

3.5 MAX30102 23

3.6 Amazon Alexa 25

3.7 Arduino IoT Cloud 26

3.8 Arduino IDE 26

3.9 Arduino IoT Remote 27

3.10 Comunicación $I2C$ 28

CAPÍTULO IV-Desarrollo del proyecto 29

4.1 Diseño e implementación del circuito 29

4.1.1 Sensor XD-58C 29

4.1.2 Sensor MAX30102 30

4.2 Programación del dispositivo 32

4.2.1 Medición de la frecuencia cardíaca 32

4.2.2 Medición de saturación de oxígeno 34

4.3 Pruebas y resultados con diferentes sensores 37

4.3.1 Prueba de frecuencia cardiaca y saturación de oxígeno 37

4.3.2 Resultados de frecuencia cardiaca con el sensor XD-58C 39

4.3.3 Resultados de frecuencia cardiaca y saturación de oxígeno con el sensor MAX30102 44

4.4 Conexión con la plataforma Arduino IoT Cloud 50

4.4.1 Creación de cuenta 50

4.4.2 Configuración de "Thing" y vinculación con microcontrolador 51

4.4.3 Integración y ejecución del programa 58

4.4.4 Creando un Dashboard en Arduino IoT Cloud 62

4.5 Monitoreo de los signos vitales mediante la aplicación IoT Remote. 65

4.6 Conexión con la IA Alexa 66

CAPÍTULO V- Conclusiones y trabajos futuros. 68

5.1 Conclusión 68

5.2 Trabajo futuro 69

Anexos 70

Anexo A. Código en Arduino IDE para el sensor XD-58C 70

Anexo B. Código en Arduino IDE para el sensor MAX30102 71

Anexo C. Código en Arduino IoT Cloud para Max_Heart (Thing 1) 76

Anexo D. Código en Arduino IoT Cloud para AlexaReadsHeartRate (Thing 2) ... 82

Referencias 89

Índice de figuras

Figura 1: Dispositivo de recopilación de datos y circuitos internos. 17

Figura 2: Recopilación de señales PPG. 17

Figura 3: Dispositivo PPG con oxímetro de pulso de referencia. 18

Figura 4: Diseño de circuito para sensor PPG portátil. 18

Figura 5: Conexión en el dispositivo portátil integrando el NodeMCU ESP-32 y MAX30102. 19

Figura 6: Comunicación entre el dispositivo portátil con aplicación móvil. 19

Figura 7: ESP-WROOM-32 21

Figura 8: Oxímetro de modelo YM203 22

Figura 9: Sensor XD-58C. 23

Figura 10: Sensor MAX30102. 24

Figura 11: Amazon Alexa. 25

Figura 12: Plataforma Arduino IoT Cloud. 26

Figura 13: Arduino IDE. 27

Figura 14: Arduino IoT Remote. 27

Figura 15: Comunicación I2C. 28

Figura 16: Esquema electrónico de la conexión del XD-58C al NodeMCU ESP32. 29

Figura 17: Conexión física del XD-58C al NodeMCU ESP32. 30

Figura 18: Esquema electrónico de la conexión del MAX30102 al NodeMCU ESP32. 31

Figura 19: Terminales de la ESP32 que soportan protocolo I2C. 31

Figura 20: Conexión física del MAX30102 al NodeMCU ESP32. 32

Figura 21: Biblioteca Wire 33

Figura 22: Biblioteca MAX30100lib 33

Figura 23: Medición de la frecuencia cardíaca y saturación de oxígeno con un oxímetro. 37

Figura 24: Medición de la frecuencia cardíaca con el sensor de pulso XD-58C. 38

Figura 25: Medición de la frecuencia cardíaca y saturación de oxígeno con el sensor de pulso MAX30102. 38

Figura 26: Ritmo cardíaco (en reposo) vs No. de mediciones de oxímetro YM203 y sensor XD-58C. 41

Figura 27: Ritmo cardíaco (ejercicio ligero) vs No. de mediciones de oxímetro YM203 y sensor XD-58C. 43

Figura 28: Ritmo cardíaco (en reposo) vs No. de mediciones de oxímetro YM203 y MAX30102..... 46

Figura 29: Saturación de oxígeno (en reposo) vs No. de mediciones de oxímetro YM203 y MAX30102. 46

Figura 30: Ritmo cardíaco (ejercicio ligero) vs No. de mediciones de oxímetro YM203 y MAX30102. 48

Figura 31: Saturación de oxígeno (ejercicio ligero) vs No. de mediciones de oxímetro YM203 y MAX30102. 49

Figura 32: Página principal de Arduino IoT Cloud. 50

Figura 33: Página de inicio de sesión de Arduino IoT Cloud. 51

Figura 34: Creación de Thing en Arduino IoT Cloud. 51

Figura 35: Max_Heart (Thing 1). 52

Figura 36: Configuración del dispositivo. 52

Figura 37: ESP32 Dev Module. 52

Figura 38: Muschi. 53

Figura 39: Clave secreta. 53

Figura 40: Configuración de Red. 54

Figura 41: Configuración de variables. 54

Figura 42: Variable Heart beat. 55

Figura 43: Variable ESpO2. 55

Figura 44: Variable Temp. 56

Figura 45: Max_Heart vinculado con Muschi. 56

Figura 46: Variables compatible con Alexa. 57

Figura 47: Variable heartrate. 57

Figura 48: AlexaReadsHeartRate (Thing 2). 58

Figura 49: Sketch de Arduino IoT Cloud para Max_Heart. 58

Figura 50: Arduino Create Agent. 59

Figura 51: Drivers de Arduino Create Agent. 59

Figura 52: Editor de Arduino IoT Cloud. 60

Figura 53: Opción de subir código. 60

Figura 54: Opción de subir biblioteca. 60

Figura 55: Monitor serial del Editor para Max_Heart. 61

Figura 56: Sketch de Arduino IoT Cloud para AlexaReadsHeartRate. 61

Figura 57: Monitor serial del Editor para AlexaReadsHeartRate. 62

Figura 58: Creando un Dashboard. 62

Figura 59: Max Dashboard. 63

Figura 60: Uso de widgets en Dashboard. 63

Figura 61: Enlace entre un widget y una variable. 63

Figura 62: Variable Temp 64

Figura 63: Max Dashboard. 64

Figura 64: a) Inicio de sesión. b) y c) Max Dashboard. d) NodeMCU ESP32. 65

Figura 65: a) - e) Conexión del dispositivo a Alexa a través de Arduino IoT Cloud y f) Valor de frecuencia cardíaco mostrado en Amazon Alexa en tiempo real. 67

Índice de Tablas

Tabla 1: Rangos normales y anormales de frecuencia cardíaca y saturación de oxígeno. 20

Tabla 2: Características eléctricas del microcontrolador ESP-WROOM-32 21

Tabla 3: Especificaciones técnicas del oxímetro YM203. 22

Tabla 4: Especificaciones técnicas del sensor XD-58C. 23

Tabla 5: Especificaciones técnicas del sensor MAX-30102. 25

Tabla 6: Valores de frecuencia cardíaca del sensor XD-58C y error porcentual **en reposo** en comparación con un oxímetro. 39

Tabla 7: Valores de frecuencia cardíaca del sensor XD-58C y error porcentual después de realizar un **ejercicio ligero** en comparación con un oxímetro. 41

Tabla 8: Valores de frecuencia cardíaca y saturación de oxígeno del sensor MAX30102 y error porcentual **en reposo** en comparación con un oxímetro. 44

Tabla 9: Valores de frecuencia cardíaca y saturación de oxígeno del sensor MAX30102 y error porcentual después de realizar un **ejercicio ligero** en comparación con un oxímetro. 47

CAPÍTULO I-INTRODUCCIÓN

Según la Secretaría de Salud cada año 220 mil personas fallecen debido a enfermedades del corazón (Salud, 2024). Por otro lado, la Organización Mundial de la Salud (OMS) menciona que, en México, el 19% de mujeres y hombres de 30 a 69 años mueren de enfermedades cardiovasculares, y se estima que el 70.3% de la población adulta vive con al menos un factor de riesgo cardiovascular. En todo el mundo las enfermedades cardiovasculares son la principal causa de defunción en el mundo y según se cobran 17.9 millones de vidas cada año. (Salud O. M., 2024) La mayoría de estas personas se encuentran con esta fe inoportuna al quedarse solos en casa sin supervisión. Este problema crea la necesidad de una monitorización adecuada de los signos vitales sobre todo en personas mayores. Comprender el rango normal de la frecuencia cardíaca y la saturación de oxígeno, que son los signos vitales por examinar, es esencial para monitorear el bienestar humano y detectar posibles anomalías. La frecuencia cardíaca en reposo (FCR), que refleja una menor actividad de bombeo de sangre durante los períodos de calma, suele oscilar entre 60 y 100 latidos por minuto (bpm) (Klevjer et al., 2023). De manera similar, la saturación de oxígeno (SpO_2) en una persona sana con pulmones normales debería estar entre el 95% y el 100% (Panja et al., 2023). La actividad física regular y el ejercicio son factores clave para mantener un ritmo cardíaco saludable y una saturación de oxígeno dentro del rango normal. Al medir los signos vitales, ciertas áreas como las muñecas, el costado del cuello, los dedos y especialmente el lóbulo de la oreja garantizan una medición más precisa (De Oliveira Costa et al., 2023). Especialmente en personas de edad avanzada, la monitorización de la frecuencia cardíaca y saturación de oxígeno se vuelve cada vez más importante debido a los elevados riesgos asociados con las enfermedades cardiovasculares relacionadas con la edad. En esta era moderna, el avance tecnológico ha permitido el uso del Internet de las Cosas (IoT) para brindar una solución a este problema, haciendo uso de sensores de frecuencia cardiaca y saturación de oxígeno (Li et al., 2024).

En este contexto más amplio, esta tesis no solo se alinea con la necesidad imperativa de un seguimiento regular de la salud para los adultos mayores, sino que también tiene como propósito integrar de manera fluida los sensores XD-58C y MAX30102, aprovechando soluciones tecnológicas avanzadas para la monitorización integral de los signos vitales y la comunicación con plataformas como Amazon Alexa. Esta tesis comienza con una exploración detallada de las especificaciones técnicas de estos sensores, incluidas pruebas preliminares realizadas dentro del entorno de Arduino IDE. Simultáneamente, se evaluará la funcionalidad y compatibilidad del microcontrolador NodeMCU ESP-32. El proyecto incluye un análisis comparativo meticuloso con un oxímetro comercial para evaluar el rendimiento de cada sensor. A medida que avanza el proyecto, el enfoque se traslada al desarrollo de código en Arduino, resaltando en la compatibilidad y funcionalidad sin problemas con la placa NodeMCU ESP-32. Las pruebas y comparaciones individuales de los sensores XD-58C y MAX30102 guiarán la selección del sensor más preciso. Por último, los pasos posteriores implican la integración de estos sensores seleccionados con la plataforma Arduino IoT Cloud, facilitando la monitorización remota a través de la aplicación Arduino IoT Remote para una mayor accesibilidad.

1.1 Resumen

La frecuencia cardíaca y la saturación de oxígeno son indicadores vitales de la salud cardiovascular. Su importancia es especialmente alta en la población de edad avanzada, donde las complicaciones cardiovasculares representan un riesgo sustancial. Este trabajo aborda esta preocupación apremiante proponiendo el monitoreo de signos vitales mediante aplicaciones de IoT. La investigación destaca la importancia de comprender los rangos normales de estos signos vitales y el papel de la actividad física en el mantenimiento del bienestar cardiovascular. Aprovechando los avances en la tecnología de IoT, el estudio tiene como objetivo monitorear de manera efectiva los signos vitales en personas mayores y, en última instancia, mejorar su calidad de vida. Para lograr este objetivo, el estudio práctico compara el rendimiento de dos sensores, el XD-58C y el MAX30102, para determinar su idoneidad para una posible integración futura en un

dispositivo portátil. A través de un meticuloso análisis comparativo que involucró a un individuo en dos escenarios (descanso y ejercicio ligero), se descubrió que el sensor MAX30102 presentaba una precisión superior en la medición de la frecuencia cardíaca. Posteriormente, el sensor seleccionado se integró con la plataforma Arduino IoT Cloud para una comunicación fluida con Amazon Alexa, lo que permite un monitoreo continuo de la salud del corazón, a pesar de encontrar limitaciones mínimas.

1.2 Objetivos

1.2.1 Objetivo general

Desarrollar un sistema de monitoreo remoto de frecuencia cardiaca utilizando tecnologías de sensores avanzadas y plataformas IoT, para la supervisión de adultos mayores.

1.2.2 Objetivos específicos

- Realizar una investigación sobre sistemas de monitoreo similares con la finalidad de realizar una comparativa con el proyecto a desarrollar.
- Elegir los dispositivos adecuados (sensores, microcontroladores, entre otros) para implementar el prototipo.
- Realizar la programación del microcontrolador ESP32 para que opere en conjunto con los sensores XD-58C y MAX30102.
- Evaluar el rendimiento de los sensores XD-58C y MAX30102 para una monitorización precisa de la frecuencia cardíaca comparándolo con un Oxímetro Comercial.
- Realizar un análisis comparativo entre los sensores XD-58C y MAX30102 para determinar su idoneidad para la integración en el sistema de monitoreo.
- Integrar perfectamente el sensor seleccionado con el microcontrolador NodeMCU ESP-32 y las plataformas IoT, estableciendo un canal de comunicación efectivo para los datos monitoreados, particularmente con plataformas de inteligencia artificial como Amazon Alexa.

1.3 Justificación

Este proyecto surge del reconocimiento de una brecha notable en la tecnología actual de dispositivos de monitorización de signo vitales portátiles. Si bien estos dispositivos monitorean eficazmente la frecuencia cardíaca y saturación de oxígeno, a menudo carecen de una conexión crucial con sistemas de inteligencia artificial (IA), como Amazon Alexa o aplicaciones de monitoreo remoto. Esta integración tiene un potencial significativo para revolucionar los sistemas de salud, particularmente para las personas mayores que a menudo enfrentan un riesgo elevado de problemas relacionados con el corazón. Al establecer una comunicación adecuada entre los dispositivos portátiles de monitoreo de la frecuencia cardíaca y la IA, se puede desarrollar un sistema sofisticado capaz no solo de rastrear la salud del corazón sino también de analizar patrones de datos para proporcionar advertencias tempranas de problemas potenciales. Este avance tecnológico puede beneficiar enormemente a la sociedad al permitir que los familiares y o personas que tienen bajo su supervisión adultos mayores monitoreen de forma remota la salud cardíaca de sus seres queridos, independientemente de su ubicación física. Este seguimiento continuo y la intervención oportuna tienen el potencial de reducir significativamente la carga de enfermedades relacionadas con el corazón y mejorar la calidad de vida general de las personas mayores.

1.4 Alcance y limitaciones

1.4.1 Alcance

El alcance de este trabajo abarca el monitoreo de signos vitales del usuario mediante aplicaciones de IoT. A través de un análisis comparativo, se identificará el sensor de frecuencia cardíaca más adecuado y se integrará con un microcontrolador ESP32 de la marca Espressif Systems. Además, se dirigirá el esfuerzo a establecer una comunicación fluida entre el sensor de frecuencia cardíaca y saturación de oxígeno y un sistema de IA. Este esfuerzo tiene como objetivo sentar las bases para un futuro dispositivo sofisticado, capaz de proporcionar monitoreo y análisis en tiempo real de los signos vitales en estudio. Sin embargo, es importante señalar que el proyecto no abarcará el diseño,

fabricación y montaje del dispositivo portátil debido a restricciones y limitaciones logísticas.

1.4.2 Limitaciones

La integración de los sensores de signos vitales con la IA Amazon Alexa: Los datos de la frecuencia cardiaca se lograron monitorear de manera remota a través de la plataforma Arduino IoT Cloud sin ningún problema. El detalle fue la integración con la aplicación de Alexa al momento de invocar por comandos de voz la variable “ritmo cardiaco” esta no lo permitía, debido a que se encuentra limitada en cuanto a su programación y desarrollo. Solamente permitía invocar variables como temperatura, lo que llevó a invocar por voz “temperatura", pero en realidad eran valores de la variable “ritmo cardiaco”.

El tiempo para la implementación del prototipo de un dispositivo portátil ergonómico para monitoreo de frecuencia cardíaca: por cuestiones de tiempo, el diseño e implementación del prototipo quedarán como proyectos a futuro.

CAPÍTULO II- ANTECEDENTES

2.1 Estadísticas de problemas cardiovasculares en México y en el mundo

La naturaleza imperativa de desarrollar un sistema de monitoreo remoto de la frecuencia cardíaca para adultos mayores se hace evidente al considerar las asombrosas estadísticas que rodean las enfermedades cardiovasculares. Anualmente, sólo en México, 220 mil personas perecen a enfermedades relacionadas con el corazón (Salud, 2024). A nivel mundial, las enfermedades cardiovasculares cobran la asombrosa cifra de 17.9 millones de vidas cada año, según la Organización Mundial de la Salud (Salud O. M., 2024). Estas cifras subrayan la importancia crítica del monitoreo continuo de los signos vitales, particularmente entre las poblaciones vulnerables.

2.2 Dispositivos de monitoreo para el ritmo cardiaco

En respuesta a esta necesidad apremiante, las tecnologías de sensores avanzadas y las plataformas de IoT presentan soluciones prometedoras. Los dispositivos de muñeca hechos a medida, equipados con sensores como el sensor de frecuencia cardíaca MAX30102, el sensor SP02, MPU6500 y el chip magnetómetro GY273, constituyen la piedra angular para la recopilación de datos precisa (ver figura 1) (Nandi et al., 2024). Además, la integración de biosensores en dispositivos portátiles, como defiende Costa et al. (2023), agiliza el seguimiento continuo y la detección temprana de problemas.

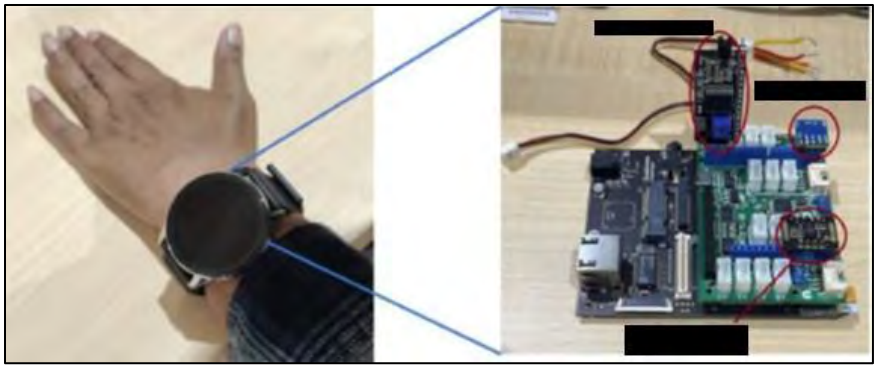


Figura 1: Dispositivo de recopilación de datos y circuitos internos.

2.3 Métodos para la medición de ritmo cardiaco

El uso de fotopletismografía (PPG), conocidos por sus siglas en inglés como Photoplethysmography, surge como componente fundamental en la monitorización de la frecuencia cardíaca al medir la luz reflejada a través de la sangre. Abuzairi et al. (2024) enfatizan la importancia de las consideraciones de diseño para contrarrestar la interferencia de luz externa en las señales de PPG, garantizando así la integridad de la señal. Además, las ideas de Ngoc-Thang et al. (2022) con respecto a algoritmos robustos de adquisición y procesamiento de señales son invaluables para una estimación precisa de la frecuencia cardíaca utilizando señales PPG (ver figuras 2 y 3).

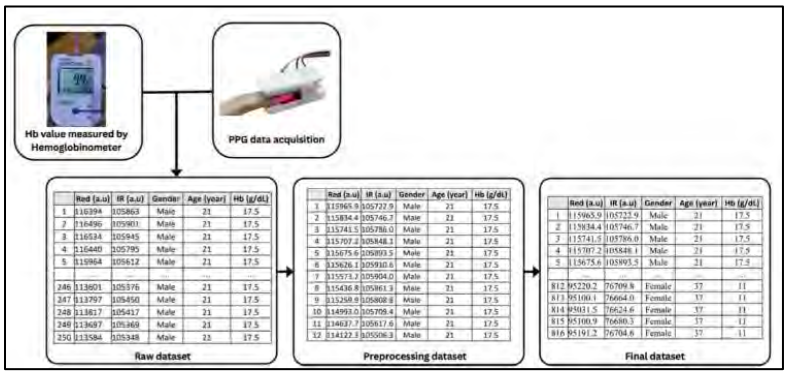


Figura 2: Recopilación de señales PPG.

Figura 3: Dispositivo PPG con oxímetro de pulso de referencia.

2.4 Monitoreo remoto del ritmo cardiaco

La importancia de las aplicaciones de IoT en la atención sanitaria, tal como lo explican Li et al. (2024), subraya el potencial transformador de los sistemas de monitorización de la frecuencia cardíaca compatibles con IoT. La tecnología IoT facilita el seguimiento remoto de los usuarios, mejorando así la accesibilidad y la comodidad (Panja et al., 2023). Además, la investigación innovadora sobre sensores PPG portátiles para la monitorización no invasiva de la frecuencia cardíaca, como lo demostraron Miller et al. (2023), subraya la versatilidad y resistencia de dichos sistemas, sentando las bases para soluciones sanitarias innovadoras (ver figura 4).

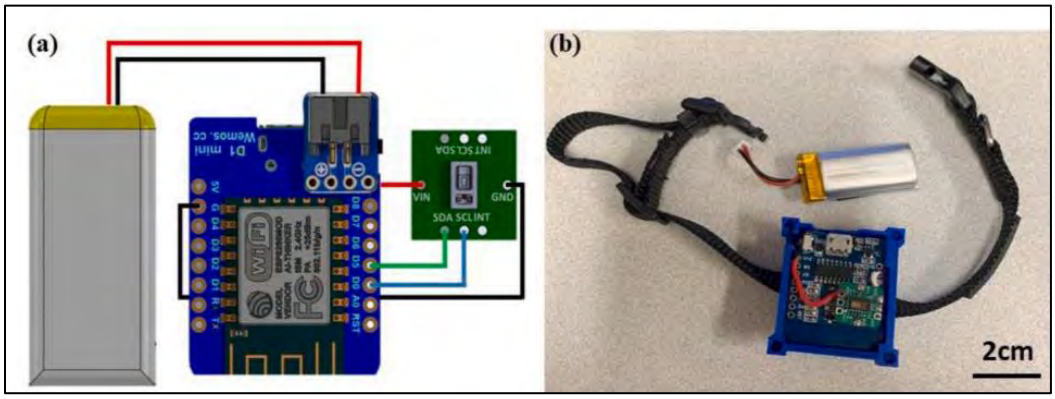


Figura 4: Diseño de circuito para sensor PPG portátil.

Además, Bernadus et al. (2019) ofrecen información crucial sobre los componentes de hardware y software esenciales para desarrollar dispositivos IoT portátiles para monitorear los signos vitales (ver figura 5). Al integrar perfectamente sensores como el XD-58C y MAX30102 con microcontroladores avanzados como NodeMCU ESP-32, el sistema garantiza una comunicación y transmisión de datos indetectables (ver figura 6).

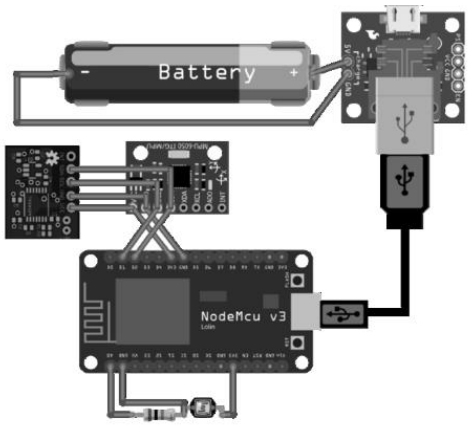


Figura 5: Conexión en el dispositivo portátil integrando el NodeMCU ESP-32 y MAX30102.

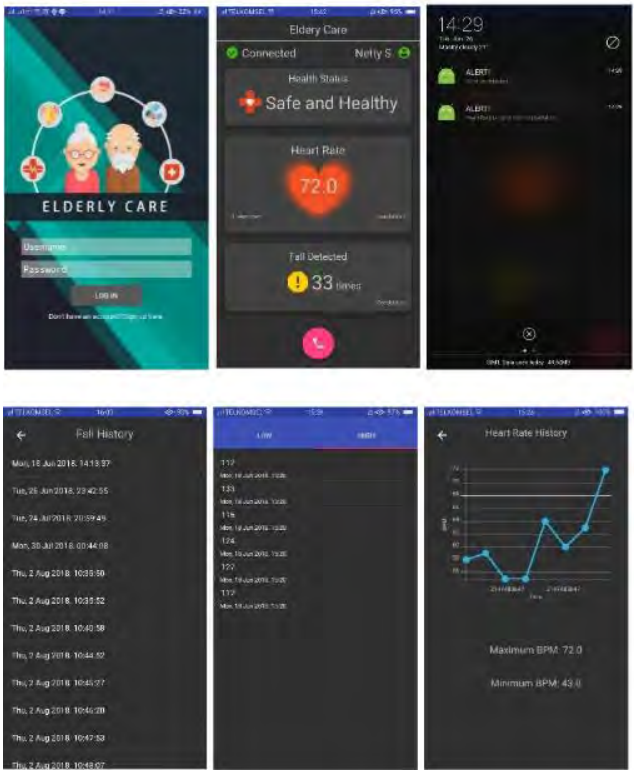


Figura 6: Comunicación entre el dispositivo portátil con aplicación móvil.

CAPÍTULO III- MARCO TEÓRICO

3.1 Signos Vitales

Los signos vitales monitorean funciones clave del cuerpo, como el pulso, la saturación de oxígeno, la temperatura, la presión arterial y la tasa de respiración. Los valores normales pueden variar según la edad, el índice de masa corporal y otros factores (Professional, 2024). La tabla 1 presenta los rangos normales y anormales para los signos vitales estudiados.

Tabla 1: Rangos normales y anormales de frecuencia cardíaca y saturación de oxígeno.

Signo Vital	Categoría	Rango
Frecuencia cardiaca (bpm)	Bradicardia	< 60 bpm
	Normal	60 – 100 bpm
	Taquicardia	> 100 bpm
Saturación de oxígeno (%)	Hipoxemia	< 90 %
	Normal	95 – 100 %
	Hipoxia	> 100 %

3.2 NodeMCU ESP-32

La placa NodeMCU ESP-32 surge como un potente microcontrolador que emplea el módulo ESP-WROOM-32 (ver figura 7) reconocido por sus capacidades en aplicaciones Wi-Fi y Bluetooth (Kashif, 2023). Con dos núcleos de CPU controlados de forma autónoma, frecuencias de reloj ajustables y una amplia gama de periféricos. Este microcontrolador destaca en integración electrónica, conectividad y operaciones de bajo consumo. Su versatilidad y especificaciones robustas lo convierten en una excelente opción para proyectos que exigen una comunicación inalámbrica confiable y eficiente (Espressif Systems, 2018). La tabla 2 describe brevemente las clasificaciones eléctricas máximas absolutas del microcontrolador.

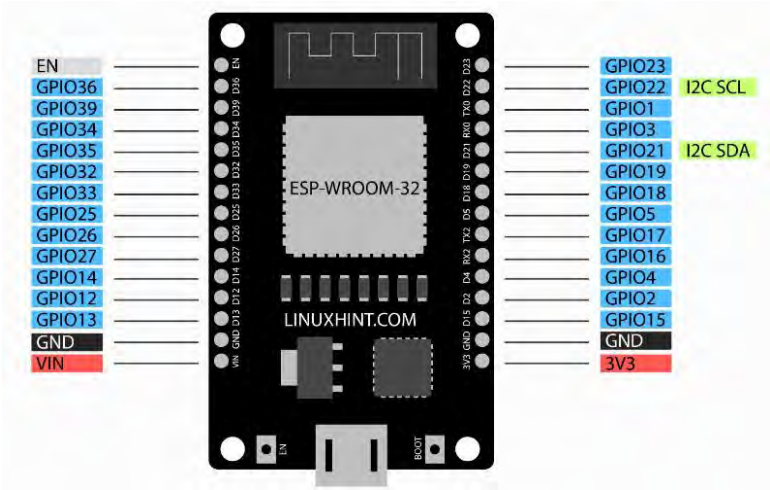


Figura 7: ESP-WROOM-32

Tabla 2: Características eléctricas del microcontrolador ESP-WROOM-32

Parameter	Symbol	Min	Typ	Max	Unit
Power supply	VDD	2.7	3.3	3.6	V
Minimum current delivered by power supply	I_{VDD}	0.5	-	-	A
Input low voltage	V_{IL}	-0.3	-	$0.25 \times V_{IO}$[1]	V
Input high voltage	V_{IH}	$0.75 \times V_{IO}$[1]	-	V_{IO}[1]+0.3	V
Input leakage current	I_{IL}	-	-	50	nA
Input pin capacitance	C_{pad}	-	-	2	pF
Output low voltage	V_{OL}	-	-	$0.1 \times V_{IO}$[1]	V
Output high voltage	V_{OH}	$0.8 \times V_{IO}$[1]	-	-	V
Maximum output drive capability	I_{MAX}	-	-	40	mA
Storage temperature range	T_{STR}	-40	-	85	°C
Operating temperature range	T_{OPR}	-40	-	85	°C

3.3 Oxímetro

Un oxímetro de pulso es un dispositivo no invasivo que mide los niveles de oxígeno en sangre y el pulso cardiaco, detectando cambios de manera eficiente. Se adhiere a partes del cuerpo, comúnmente al dedo, y se utiliza en diversos entornos, incluidos consultorios médicos y hogares, y ofrece información valiosa sobre la saturación de oxígeno y la salud cardiovascular (Gotter, 2023). El oxímetro para utilizar es el modelo YM023 (ver figura

8), y en la tabla 3 se detallan sus especificaciones técnicas (YM203 Multifunctional Bluetooth CE ROHS Monitor Fingertip Pulse Oximeter, n.d.).

Figura 8: Oxímetro de modelo YM203

Tabla 3: Especificaciones técnicas del oxímetro YM203.

Especificaciones Técnicas	
Modelo	YM203
Grupo	De mediana edad y viejo
Rango de medición Pr.	30 Bpm-250 bpm (resolución inferior a 1 bpm)
Temperatura de trabajo	10ºC-40ºC
Tiempo de medición	8 segundos
Rango de medición de SpO2	0%-100% (la resolución es 1%); Precisión de SpO2: 70%
Precisión de Pr.	±2bpm o ±2% (Seleccione más grande)
Rango de frecuencia del pulso	30bpm-250bpm

3.4 XD-58C

El sensor de pulso XD-58C (ver figura 9) es un sensor óptico de frecuencia cardíaca rentable diseñado para ser compatible con Arduino, ESP32, ESP8266 o Raspberry Pi, y ofrece una solución práctica para medir la frecuencia cardíaca durante actividades físicas. El sensor se coloca en uno de los dedos de la mano y aprovechando un sensor óptico detecta cambios en el flujo sanguíneo a través del reflejo de una luz infrarroja, lo

que se traduce en variaciones correspondiente a cada latido del corazón. Reconocido por su diseño fácil de usar, el XD-58C sirve como un sensor plug-and-play adecuado para diversas aplicaciones, incluidos proyectos estudiantiles, iniciativas artísticas, actividades deportivas y desarrollo de juegos, integrando perfectamente datos de frecuencia cardíaca en tiempo real en proyectos Arduino (XD-58C Heart Pulse Sensor Module, 2022). Las especificaciones técnicas del sensor de frecuencia cardíaca XD-58C se detallan en la tabla 4 (Sensor De Pulso Cardíaco XD58C Ritmo XD-58C, n.d.).

Figura 9: Sensor XD-58C.

Tabla 4: Especificaciones técnicas del sensor XD-58C.

Especificaciones Técnicas	
Modelo	XD-58C
Medición	Pulso cardíaco
Alimentación	3.3 ~ 5V
Distribución de Pines	“+”: VCC (3.3~ 5V) “-”: GND “S”: Señal de Salida Análoga
Corriente	4mA
Longitud de Onda del led	609 nm
Salida	Análoga
Número de cables	3

3.5 MAX30102

El MAX30102 (ver figura 10), una versión mejorada del sensor MAX30100, que funciona como monitor de frecuencia cardíaca y oxímetro de pulso (Microcontrollers Lab, 2022). Este sensor integra dos LED, un fotodetector con óptica optimizada y componentes de

procesamiento de señales de bajo ruido, lo que lo hace adecuado para aplicaciones de microcontroladores como Arduino, ESP32 y ESP8266 NodeMCU para desarrollar un dispositivo eficaz para monitorear la frecuencia cardíaca y la saturación de oxígeno. El sensor de oxígeno y frecuencia cardíaca MAX30102 emplea principios de absorción de luz para medir la saturación de oxígeno (SpO_2) y la frecuencia cardíaca (HR) a través de tecnología de reflexión de luz roja e infrarroja. Al utilizar LED infrarrojos y rojos junto con un filtro fotónico, el sensor genera una luz específica que penetra la piel y los tejidos rojos, y la luz infrarroja llega más profundamente a la piel. Los fotodiodos capturan la luz reflejada de la sangre que fluye en los vasos debajo de la piel, convirtiéndola en datos analógicos, que luego se amplifican y procesan digitalmente para obtener lecturas de frecuencia cardíaca y saturación de oxígeno (AUTODESK Instructables, 2023). La tabla 5 proporciona una descripción general completa de las especificaciones técnicas del sensor de frecuencia cardíaca MAX30102 (Naylamp Mechatronics, n.d.).

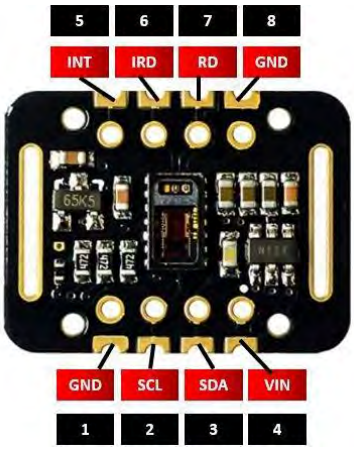


Figura 10: Sensor MAX30102.

Tabla 5: Especificaciones técnicas del sensor MAX-30102.

Especificaciones Técnicas	
Modelo	MAX30102
Medición	Pulso cardíaco y nivel de oxígeno
Alimentación	3.3 ~ 5V
Frecuencia de muestreo	50Hz – 3200 Hz
Consumo máximo de corriente	6mA
Longitud de onda máxima del LED IR	880 nm
Longitud de onda máxima del LED rojo	660 nm
Rango de temperatura	-40°C to +85°C
Precisión de temperatura	±1˚C

3.6 Amazon Alexa

Amazon Alexa es una IA activada por voz disponible en más de 100 millones de dispositivos de Amazon y otros fabricantes (ver figura 11). Permite crear interacciones de voz naturales, mejorando la experiencia del usuario con la tecnología cotidiana (Alexa, n.d.).

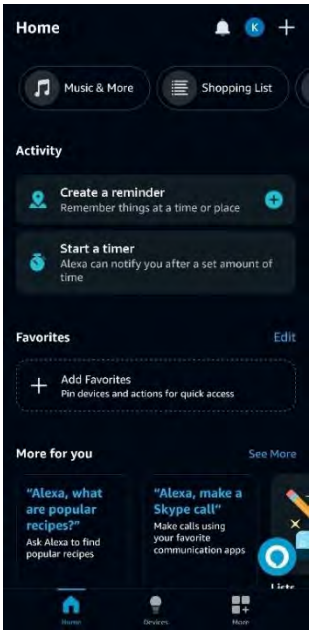


Figura 11: Amazon Alexa.

3.7 Arduino IoT Cloud

Arduino IoT Cloud (ver figura 12) es una plataforma de desarrollo de código abierto que facilita la creación de sistemas interconectados en la nube (Arduino Cloud, n.d.). Diseñada para desarrolladores y entusiastas, ofrece una alternativa asequible con bibliotecas en crecimiento, soporte comunitario y documentación técnica. Con funciones como la generación automática de paneles y soporte TLS completo, simplifica la programación y configuración, permitiendo la creación rápida de dispositivos operativos. Admite diversos métodos de interacción, como API REST HTTP y MQTT, y se enfoca en democratizar el desarrollo de IoT, con aplicaciones potenciales en agricultura, movilidad, monitoreo y seguridad (T-Bem, 2019). Además, una función destacada de Arduino IoT Cloud es su capacidad para facilitar la comunicación entre un microcontrolador, como el NodeMCU ESP-32, y Alexa. Esta característica amplía las posibilidades de interconexión y abre nuevas oportunidades para integrar dispositivos en la plataforma de voz de Alexa.

Figura 12: Plataforma Arduino IoT Cloud.

3.8 Arduino IDE

Arduino IDE (ver figura 13) es una herramienta esencial para programar placas de Arduino y microcontroladores. Permite escribir, compilar y cargar código de manera intuitiva y eficiente. Además, proporciona acceso a bibliotecas y ejemplos útiles para desarrollar proyectos (About Arduino, n.d.).

Figura 13: Arduino IDE.

3.9 Arduino IoT Remote

La aplicación Arduino IoT Remote (ver figura 14) permite a los usuarios controlar sus paneles de control de Arduino Cloud e integrar los sensores de su teléfono en su sistema IoT. Está disponible para dispositivos Android y iOS, requiere una cuenta de Arduino y se puede descargar desde la App Store o Google Play (Söderby, n.d.).

Figura 14: Arduino IoT Remote.

3.10 Comunicación I^2C

I^2C (Inter-Integrated Circuit) es un protocolo de comunicación serial que funciona con solo dos cables: SDA para la transmisión de datos y SCL para la sincronización del reloj (ver figura 15). Permite conectar múltiples dispositivos a un único maestro, con cada esclavo identificado por una dirección única de 7 o 10 bits. Los datos se intercambian en tramas que contienen información de dirección, datos y control, con la sincronización mantenida por la señal de reloj controlada por el maestro. Esta configuración asegura una comunicación confiable entre los dispositivos conectados (Campbell, 2021).

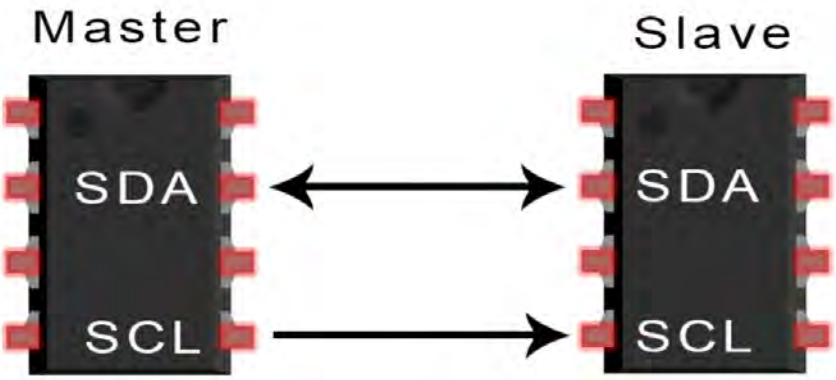


Figura 15: Comunicación I2C.

CAPÍTULO IV-Desarrollo del proyecto

4.1 Diseño e implementación del circuito

4.1.1 Sensor XD-58C

El sensor XD-58C monitorea el pulso cardiaco, usando un LED emisor y un sensor receptor. Dependiendo de la luz reflejada debida a la sangre que circula a través de las venas del cuerpo es que se define la salida de la señal del sensor. Para la conexión física se conectan las terminales a la NodeMCU ESP32 como se observa en la figura 16. Una terminal está conectada a tierra (GND), otra a la fuente de 3.3V, y la tercera, correspondiente a la señal de salida del sensor, se conecta al GPIO 4 que es un puerto digital.

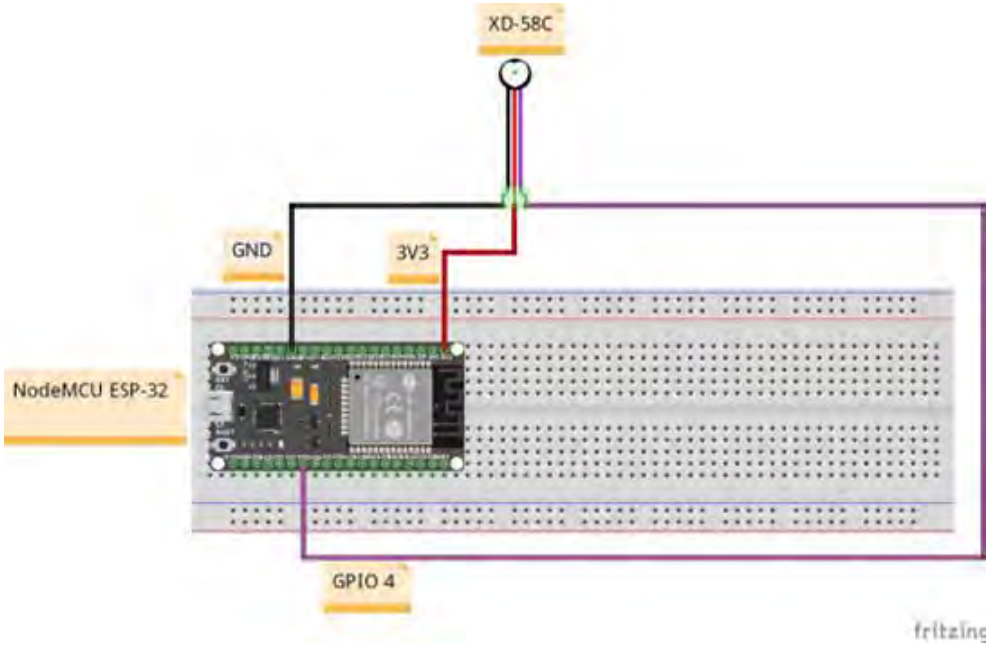


Figura 16: Esquema electrónico de la conexión del XD-58C al NodeMCU ESP32.

A continuación, se muestra la implementación del circuito (ver figura 17). Las conexiones se realizan utilizando un protoboard. Se utiliza cable dupont para la conexión y se aplica termofit para reforzar las terminales del sensor y así evitar que se desolden por la manipulación durante las pruebas.

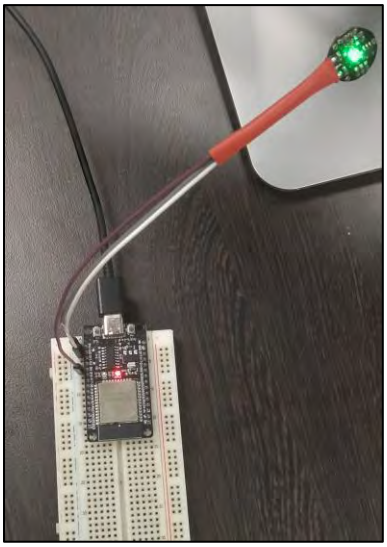


Figura 17: Conexión física del XD-58C al NodeMCU ESP32.

4.1.2 Sensor MAX30102

El sensor MAX30102 es un dispositivo que combina un pulsioxímetro y un monitor de frecuencia cardiaca. Cuenta con un led infrarrojo y un detector fotoeléctrico, trabajando bajo el mismo principio de la luz reflejada por la sangre circundante. Este sensor cuenta con 8 terminales, de los cuales solo se requieren 4 para conectarse a la NodeMCU ESP32: GND, Vin, SDA y SCL. La terminal GND se conecta a tierra, Vin a la fuente de 3.3V, SDA al GPIO 21 y SCL al GPIO 22. Es importante recordar que el sensor MAX30102 utiliza comunicación I^2C, ya que SDA y SCL deben conectarse a los GPIO 21 y 22, respectivamente, que son puertos que soportan este protocolo de comunicación (ver figuras 18 y 19).

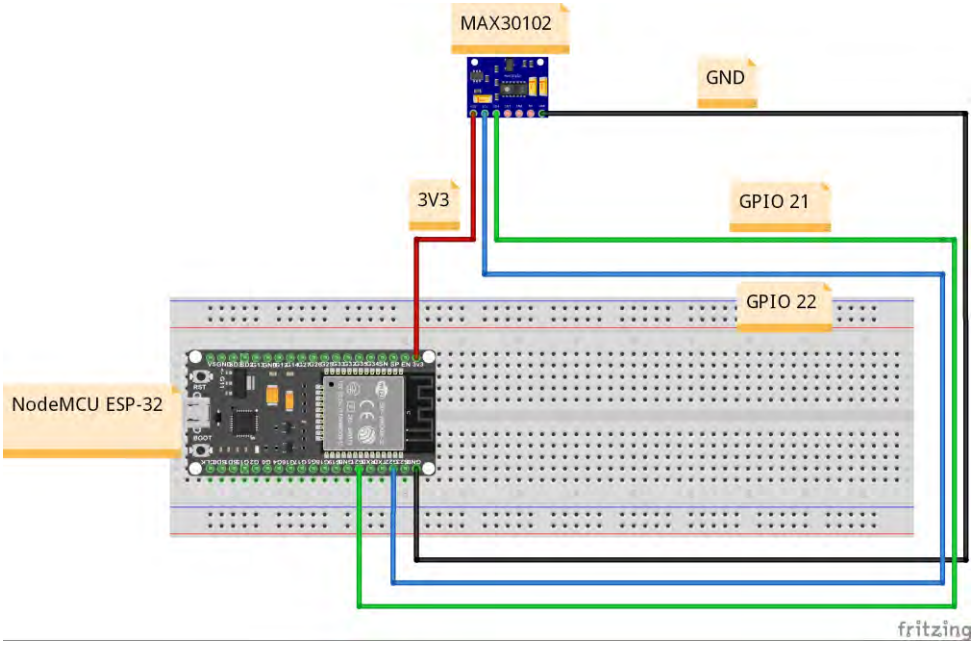


Figura 18: Esquema electrónico de la conexión del MAX30102 al NodeMCU ESP32.

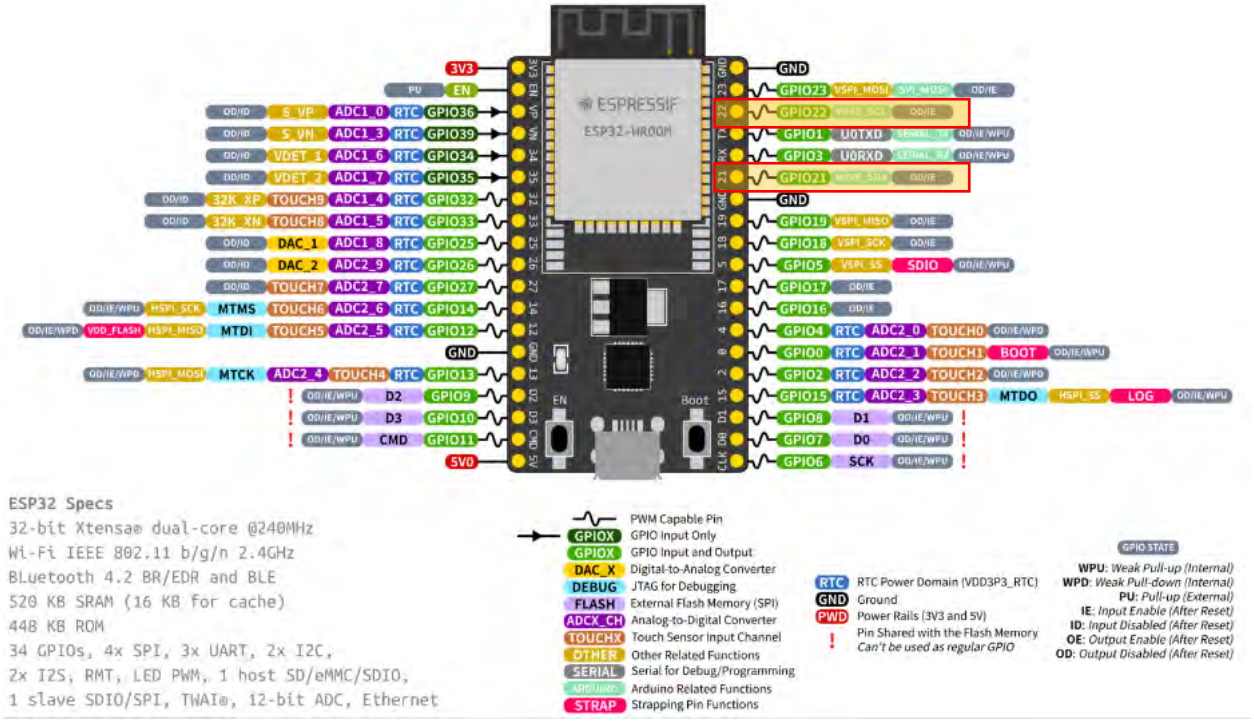


Figura 19: Terminales de la ESP32 que soportan protocolo I2C.

La implementación del circuito se muestra en la figura 20. Al igual que con el sensor XD-58C, todas las conexiones se realizan utilizando un protoboard.

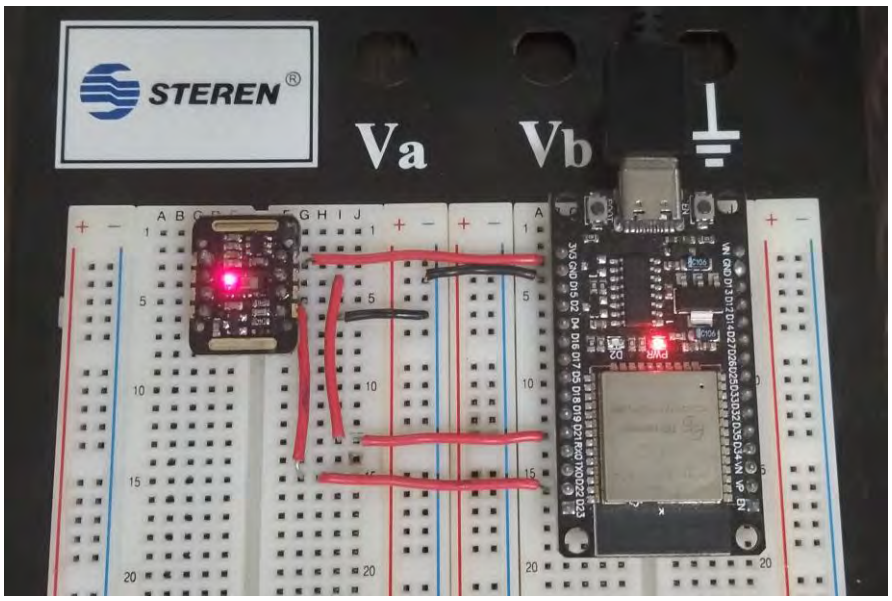


Figura 20: Conexión física del MAX30102 al NodeMCU ESP32.

4.2 Programación del dispositivo

4.2.1 Medición de la frecuencia cardíaca

Para medir la frecuencia cardíaca, es necesario entender la siguiente ecuación:

$$bpm = \frac{60\ [s]}{pulso\ [ms]} \div \frac{1\ [s]}{1000\ [ms]} = \frac{60\ [s]}{pulso\ [ms]} \times \frac{1000\ [ms]}{1\ [s]}$$

$$bpm = \frac{60000}{pulso}\ [lpm] \ldots\ldots\ldots ecuación\ 1$$

Donde:

bpm = frecuencia cardiaca [bmp].

pulso = Intervalo entre los latidos del corazón [ms].

La variable bpm calculará el valor de la frecuencia cardíaca cuando reciba el valor de la variable pulso, medido en milisegundos [ms]. Es importante entender que hay 60,000 milisegundos en un minuto, por lo que la fórmula se divide entre 1000 [ms]/[s] para convertir los milisegundos a segundos. Finalmente, se obtiene la ecuación 1, que proporciona un resultado en latidos por minuto (bpm). Se utiliza el entorno de Arduino IDE para programar el NodeMCU ESP32 y probar los sensores XD-58C y MAX30102. No es necesario utilizar ninguna biblioteca para el sensor XD-58C. Sin embargo, para el sensor MAX30102, se requieren tres bibliotecas:

- #include <Wire.h>,
- #include <MAX30105.h>,
- y #include "heartRate.h".

La biblioteca #include <Wire.h> Es esencial porque el sensor MAX30102 para que entre en funcionamiento la comunicación I^2C (ver figura 21). Esta biblioteca se encuentra en el menú Archivo > Ejemplos > Wire del Arduino IDE. Por otro lado, las bibliotecas #include <MAX30105.h> y #include "heartRate.h" son específicas para el sensor MAX30102 y se pueden encontrar en el 'Gestor de Bibliotecas' del Arduino IDE (ver figura 22).

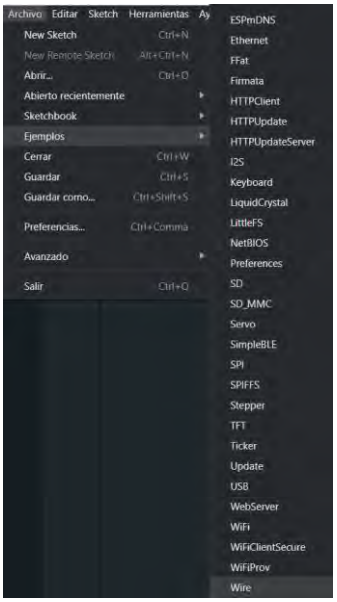


Figura 21: Biblioteca Wire

Figura 22: Biblioteca MAX30100lib

A continuación, se muestra cómo se usa la ecuación 1 para calcular la frecuencia cardíaca del sensor XD-58C. El código completo para este cálculo se encuentra en el **Anexo A**:

```
    bpm = 60000 / pulso; // Calculo
del ritmo cardíaco
```

De manera similar, para el sensor MAX30102, la frecuencia cardíaca se calcula de la siguiente forma. En lugar de la variable pulso, se utiliza la variable delta, que cumple una función equivalente. Los detalles completos del código están en el **Anexo B**:

```
    beatsPerMinute = 60 / (delta /
1000.0);
```

4.2.2 Medición de saturación de oxígeno

Para medir la saturación de oxígeno, se utilizan las siguientes ecuaciones, que son empleadas en la calibración del algoritmo de SpO_2 de Maxim.

$$SpO_2 = aR^2 + bR + c \ldots\ldots\ldots. ecuación\ 2$$

Donde,

$$R = \frac{AC_{red}/DC_{red}}{AC_{ired}/DC_{ired}} \ldots\ldots\ldots. ecuación\ 3$$

y

SpO_2 = Porcentaje de saturación de oxígeno en la sangre [%].

a, b y c = coeficientes de calibración.

AC_{red} = Componente de corriente alterna de la señal de luz roja.

DC_{red} = Componente de corriente directa de la señal de luz roja.

AC_{ired} = Componente de corriente alterna de la señal de luz infrarroja.

DC_{ired} = Componente de corriente directa de la señal de luz infrarroja.

La proporción R derivado de la *ecuación 3* es esencial en el cálculo de SpO_2 con el algoritmo de Maxim. Esta relación captura la relación entre las señales pulsátiles de la luz roja y la infrarroja. Luego, la *ecuación 2* aplica esta relación en una fórmula polinómica para estimar la SpO_2, utilizando coeficientes de calibración a, b y c para garantizar lecturas precisas que reflejen las condiciones del mundo real (Guidelines for SPO2 Measurement, 2023). Este método combina el análisis de señales con datos de calibración para ofrecer mediciones precisas de la saturación de oxígeno en sangre.

En el siguiente código, se calcula la saturación de oxígeno a partir de las señales de luz roja (red) e infrarroja (ir). Primero, se actualizan los valores promedio (ave) de estas señales usando un filtro pasa-bajo para suavizar las fluctuaciones:

```
    avered = avered * frate + (double)red * (1.0 - frate); // Nivel de
rojo promedio por filtro de low pass
    aveir = aveir * frate + (double)ir * (1.0 - frate); // Nivel IR
promedio por filtro de low pass
```

Posteriormente, se calcula la suma de los cuadrados de las diferencias entre los valores actuales y los promedios para cada señal. Este cálculo ayuda a medir la variabilidad pulsátil de las señales de luz roja e infrarroja:

```
    sumredrms += (fred - avered) * (fred - avered); // Suma cuadrada del
componente alternativo del nivel rojo
    sumirrms += (fir - aveir) * (fir - aveir); // Suma cuadrada del
componente alternativo del nivel IR
```

En este contexto, el valor rms (raíz cuadrática media) representa la parte pulsátil o AC (corriente alterna) de las señales, que fluctúa con cada latido del corazón. Por otro lado, el valor promedio representa la parte constante o DC (corriente continua), que se mantiene estable y no varía con el latido.

La relación entre estas dos componentes se refleja en la fórmula para calcular la razón R (ver *ecuación 3*):

```
double R = (sqrt(sumredrms)/ avered) / (sqrt(sumirrms) / aveir);
```

Aquí, el valor rms se compara con el promedio para ambas señales de luz roja e infrarroja. Esta relación es crucial para la estimación de la saturación de oxígeno, ya que muestra cómo la componente pulsátil de la señal varía en relación con la parte base constante.

Finalmente, la razón R se utiliza para calcular la saturación de oxígeno (SpO_2) mediante la *ecuación 2*. Los coeficientes a, b, y c se determinan a través de calibraciones empíricas, como se detalla en los gráficos de calibración del algoritmo de Maxim. Para obtener más información sobre cómo se obtienen estos coeficientes, consulte el apartado de referencias (Guidelines for SPO2 Measurement, 2023).

```
SpO2 = -23.3 * (R - 0.4) + 100; // Calcular SpO2
```

En esta última línea, ESpO2 se actualiza aplicando un filtro pasa-bajo a la nueva lectura de SpO2. Este filtro ayuda a suavizar las fluctuaciones y proporciona un valor más estable y confiable de SpO2. El coeficiente FSpO2 determina cuánto se mezcla el nuevo valor de SpO2 con el valor anterior de ESpO2, ajustando así la respuesta del sistema a las variaciones en la saturación de oxígeno (ver **Anexo B**).

```
ESpO2 = FSpO2 * ESpO2 + (1.0 - FSpO2) * SpO2;//filtro low pass
```

4.3 Pruebas y resultados con diferentes sensores

4.3.1 Prueba de frecuencia cardiaca y saturación de oxígeno

La primera parte de la prueba consistió en medir el ritmo cardíaco y la saturación de oxígeno con los sensores XD-58C y MAX30102 mientras el individuo, un hombre de 25 años en buen estado de salud, estaba sentado y relajado. Es importante destacar que el sensor XD-58C mide únicamente la frecuencia cardíaca, mientras que el sensor MAX30102 proporciona tanto la frecuencia cardíaca como la saturación de oxígeno. Se colocó un oxímetro de marca YM203 (ver figura 23) en el dedo índice de la mano derecha del individuo y el sensor XD-58C (ver figura 24) en el dedo índice de la mano izquierda. Se tomaron 50 valores en intervalos de 10 segundos para asegurar la estabilización del ritmo cardíaco. Posteriormente, se repitió el experimento utilizando el sensor MAX30102 (ver figura 25), manteniendo constante la posición del oxímetro y colocando el dedo índice de la mano izquierda sobre el sensor MAX30102.

Figura 23: Medición de la frecuencia cardíaca y saturación de oxígeno con un oxímetro.

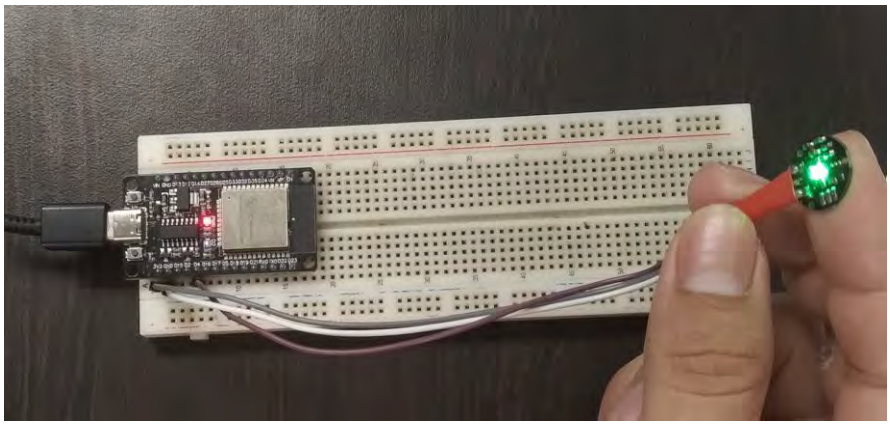


Figura 24: Medición de la frecuencia cardíaca con el sensor de pulso XD-58C.

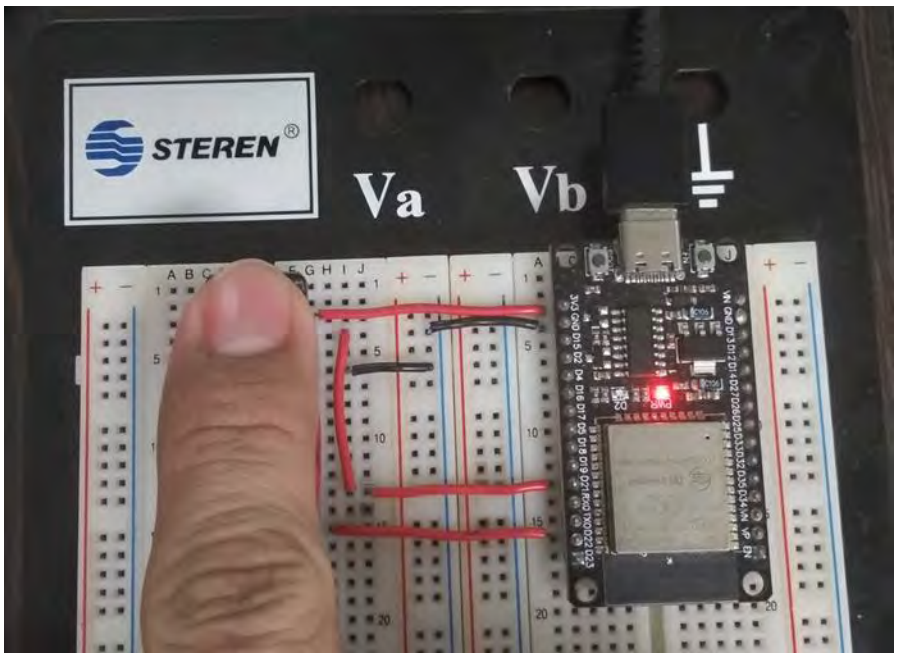


Figura 25: Medición de la frecuencia cardíaca y saturación de oxígeno con el sensor de pulso MAX30102.

En la segunda parte de la prueba, el individuo caminó a paso ligero durante 45 minutos. Luego, se midió la frecuencia cardíaca con el sensor XD-58C de la misma manera que en la primera parte. Después de obtener los valores con el sensor XD-58C, se esperaron 30 minutos antes de repetir el ejercicio ligero. Finalmente, se midieron nuevamente la frecuencia cardíaca y la saturación de oxígeno, pero ahora con el sensor MAX30102, siguiendo un procedimiento similar al de la primera parte de la prueba.

4.3.2 Resultados de frecuencia cardiaca con el sensor XD-58C

Los resultados del ritmo cardíaco obtenidos con el sensor XD-58C en reposo y después de realizar un ejercicio ligero se organizan en las tablas 6 y 7, respectivamente. Se calculó el porcentaje de error entre los valores del oxímetro y el sensor XD-58C. El porcentaje de error mostrado en la columna 4 de la tabla, es calculado con la diferencia entre el valor de referencia (en este caso el oxímetro YM203 siendo un dispositivo comercial se considera que ha sido probado antes de salir al mercado) y el valor obtenido con el sensor XD-58.

Tabla 6: Valores de frecuencia cardíaca del sensor XD-58C y error porcentual **en reposo** en comparación con un oxímetro.

No.	Oxímetro de pulso	XD-58C	% Error
	Ritmo cardíaco (bpm)	Ritmo cardíaco (bpm)	Ritmo cardíaco (bpm)
1	68	83	22%
2	64	79	23%
3	62	61	2%
4	65	28	57%
5	67	48	28%
6	64	57	11%
7	70	47	33%
8	52	30	42%
9	66	60	9%
10	71	79	11%
11	70	53	24%
12	64	47	27%
13	67	49	27%
14	67	64	4%
15	72	90	25%
16	68	52	24%
17	71	74	4%
18	72	78	8%

19	67	55	18%
20	72	90	25%
21	73	74	1%
22	70	83	19%
23	72	87	21%
24	75	96	28%
25	72	94	31%
26	67	53	21%
27	66	84	27%
28	62	41	34%
29	67	41	39%
30	68	52	24%
31	69	55	20%
32	73	60	18%
33	67	40	40%
34	66	43	35%
35	76	45	41%
36	62	44	29%
37	73	41	44%
38	62	63	2%
39	74	47	36%
40	73	60	18%
41	62	41	34%
42	62	41	34%
43	67	43	36%
44	67	60	10%
45	67	55	18%
46	67	60	10%
47	67	51	24%
48	65	43	34%
49	62	52	16%
50	62	64	3%

Se grafican las señales con los datos obtenidos con el oxímetro comercial elegido como punto de comparación y el sensor XD-58. Esto con la finalidad de facilitar una comparación visual clara entre los valores del oxímetro y del sensor obtenidas con el sujeto en reposo (ver figuras 26 y 27).

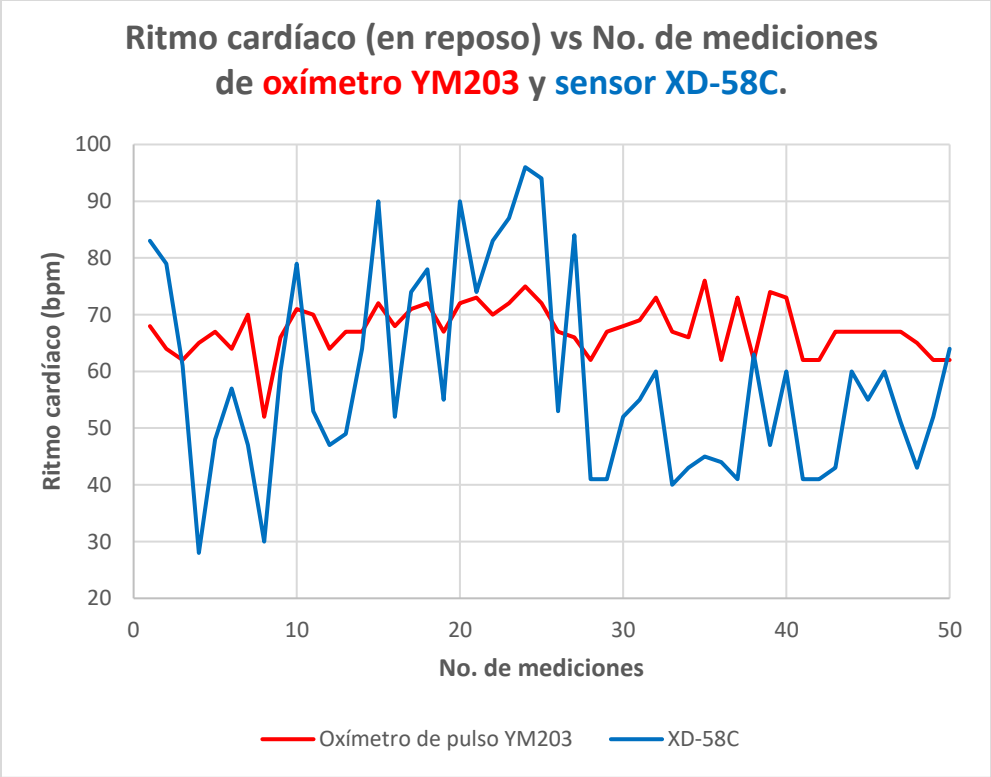


Figura 26: Ritmo cardíaco (en reposo) vs No. de mediciones de oxímetro YM203 y sensor XD-58C.

Tabla 7: Valores de frecuencia cardíaca del sensor XD-58C y error porcentual después de realizar un **ejercicio ligero** en comparación con un oxímetro.

No.	Oxímetro de pulso	XD-58C	% Error
	Ritmo cardíaco (bpm)	Ritmo cardíaco (bpm)	Ritmo cardíaco (bpm)
1	73	50	32%
2	80	60	25%
3	82	61	26%
4	93	99	6%
5	77	61	21%
6	80	63	21%
7	81	61	25%
8	81	96	19%
9	80	63	21%
10	94	92	2%
11	90	92	2%
12	79	89	13%
13	91	85	7%
14	101	95	6%

15	90	89	1%
16	76	73	4%
17	86	92	7%
18	74	75	1%
19	73	83	14%
20	89	96	8%
21	101	99	2%
22	86	67	22%
23	98	97	1%
24	74	98	32%
25	73	65	11%
26	72	65	10%
27	74	74	0%
28	81	68	16%
29	85	68	20%
30	80	62	23%
31	74	52	30%
32	80	78	3%
33	81	81	0%
34	101	95	6%
35	85	93	9%
36	81	85	5%
37	71	89	25%
38	73	79	8%
39	74	98	32%
40	88	81	8%
41	89	86	3%
42	77	74	4%
43	90	67	26%
44	96	98	2%
45	76	52	32%
46	80	67	16%
47	79	68	14%
48	80	72	10%
49	74	70	5%
50	73	65	11%

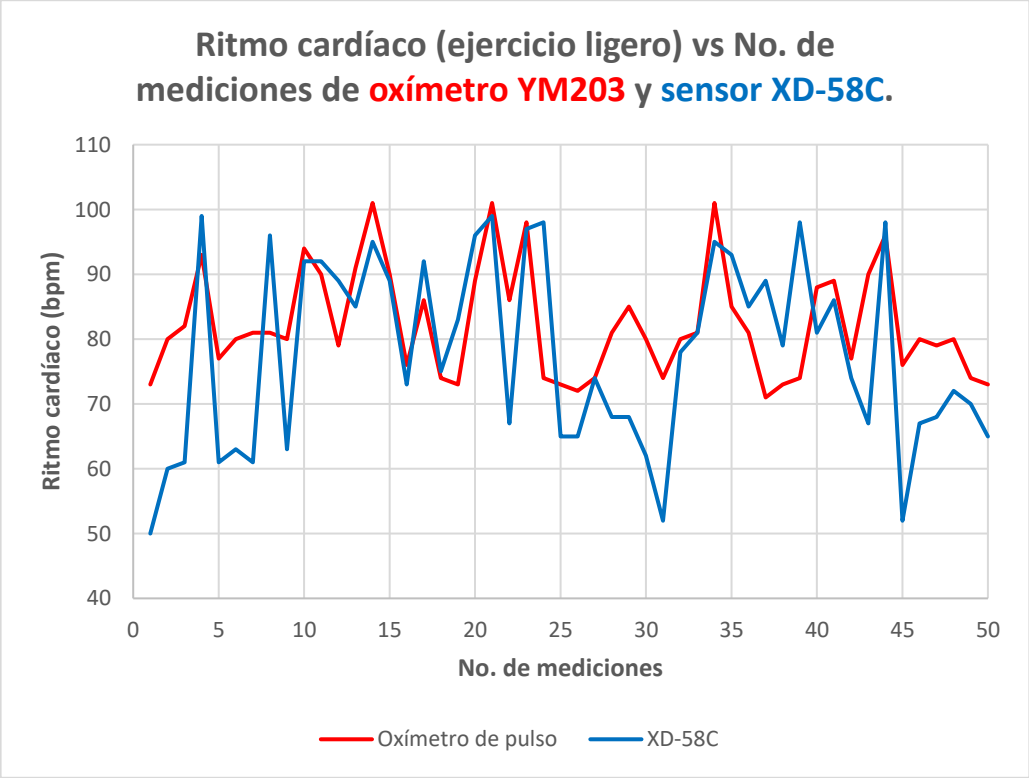


Figura 27: Ritmo cardíaco (ejercicio ligero) vs No. de mediciones de oxímetro YM203 y sensor XD-58C.

Al examinar detenidamente los resultados de las mediciones de frecuencia cardíaca entre el oxímetro YM203, comúnmente utilizado en entornos médicos, y el sensor de pulso XD-58C, surgen distinciones notables tanto en el ritmo cardíaco en reposo como después de realizar ejercicio ligero. En la tabla 6, que ilustra las mediciones en reposo, se observan variaciones sustanciales, destacando casos notables donde el sensor XD-58C mostró diferencias significativas (ver figura 26), alcanzando un error máximo del 57% en una sola medición. De manera similar, la tabla 7, que aborda las mediciones postejercicio, revela divergencias evidentes (ver figura 27), con un error máximo del 32% en una medición.

Estas disparidades observadas sugieren firmemente que el sensor XD-58C no cumple con la entrega de mediciones precisas al compararse con el estándar establecido por el oxímetro. Este hallazgo subraya la importancia crítica de optar por sensores más precisos, capaces de proporcionar valores más cercanos a los obtenidos con el oxímetro comercial, especialmente en aplicaciones médicas donde se trabaja con la salud del paciente. En consecuencia, se recomienda explorar alternativas más precisas para garantizar mediciones confiables y pertinentes en entornos clínicos y de atención médica.

4.3.3 Resultados de frecuencia cardiaca y saturación de oxígeno con el sensor MAX30102

Similar al sensor XD-58C, los resultados del ritmo cardíaco y la saturación de oxígeno obtenidos con el sensor MAX30102 en reposo y después de realizar un ejercicio ligero se presentan en las tablas 8 y 9, respectivamente. Se calculó el porcentaje de error entre los valores del oxímetro y los del sensor MAX30102. Además, se graficaron ambos conjuntos de datos para cada situación, permitiendo una comparación visual clara entre los valores del oxímetro y del sensor (ver figuras 28 a 31).

Tabla 8: Valores de frecuencia cardíaca y saturación de oxígeno del sensor MAX30102 y error porcentual **en reposo** en comparación con un oxímetro.

No.	Pulse Oximeter		MAX30102		% Error	
	Ritmo cardíaco (bpm)	SpO2 (%)	Ritmo cardíaco (bpm)	SpO2 (%)	Ritmo cardíaco (bpm)	SpO2 (%)
1	67	97	67	98.47	0%	1.52%
2	67	96	66	98.47	1%	2.57%
3	73	96	70	98.21	4%	2.30%
4	67	98	68	98.21	1%	0.21%
5	67	96	68	98.21	1%	2.30%
6	65	95	64	98.29	2%	3.46%
7	65	94	65	98.57	0%	4.86%
8	62	94	62	98.75	0%	5.05%
9	68	95	68	98.75	0%	3.95%
10	67	94	68	98.75	1%	5.05%
11	65	96	64	94.67	2%	1.39%
12	62	99	60	94.67	3%	4.37%
13	66	97	67	97.23	2%	0.24%
14	66	98	68	97.18	3%	0.84%
15	65	98	64	93.97	2%	4.11%

16	63	98	62	95.97	2%	2.07%
17	73	98	74	96.65	1%	1.38%
18	66	94	66	93.28	0%	0.77%
19	67	98	67	96.72	0%	1.31%
20	67	96	68	97.82	1%	1.90%
21	63	95	64	97.82	2%	2.97%
22	64	96	64	97.82	0%	1.90%
23	62	98	61	99.20	2%	1.22%
24	70	97	72	98.81	3%	1.87%
25	67	99	66	99.20	1%	0.20%
26	71	99	71	99.12	0%	0.12%
27	73	97	75	99.12	3%	2.19%
28	68	99	68	99.10	0%	0.10%
29	62	97	61	99.55	2%	2.63%
30	62	97	62	99.72	0%	2.80%
31	62	96	61	99.63	2%	3.78%
32	69	99	69	99.75	0%	0.76%
33	67	99	69	99.75	3%	0.76%
34	61	97	61	99.63	0%	2.71%
35	63	99	64	99.63	2%	0.64%
36	66	98	66	99.63	0%	1.66%
37	67	97	66	99.54	1%	2.62%
38	68	98	66	99.54	3%	1.57%
39	62	99	62	99.75	0%	0.76%
40	63	97	64	99.70	2%	2.78%
41	67	96	66	99.70	1%	3.85%
42	69	99	70	99.86	1%	0.87%
43	60	99	61	99.70	2%	0.71%
44	67	96	66	99.70	1%	3.85%
45	67	99	66	99.66	1%	0.67%
46	65	98	66	99.81	2%	1.85%
47	67	98	66	99.81	1%	1.85%
48	65	99	66	99.58	2%	0.59%
49	68	98	69	99.39	1%	1.42%
50	74	97	75	98.79	1%	1.85%

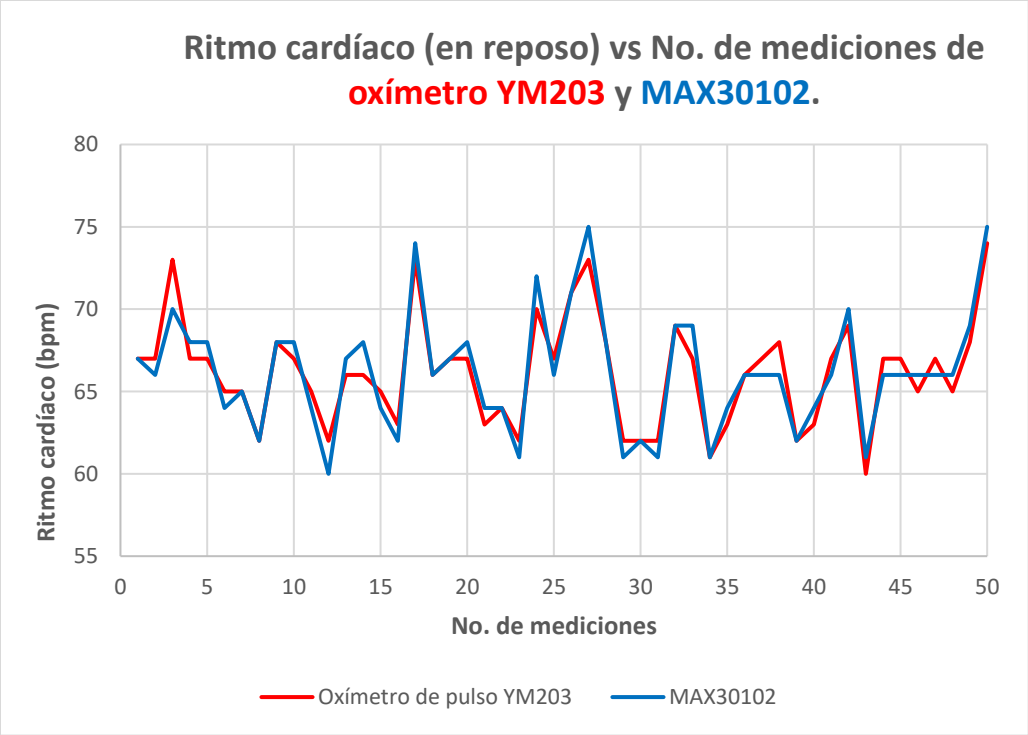


Figura 28: Ritmo cardíaco (en reposo) vs No. de mediciones de oxímetro YM203 y MAX30102.

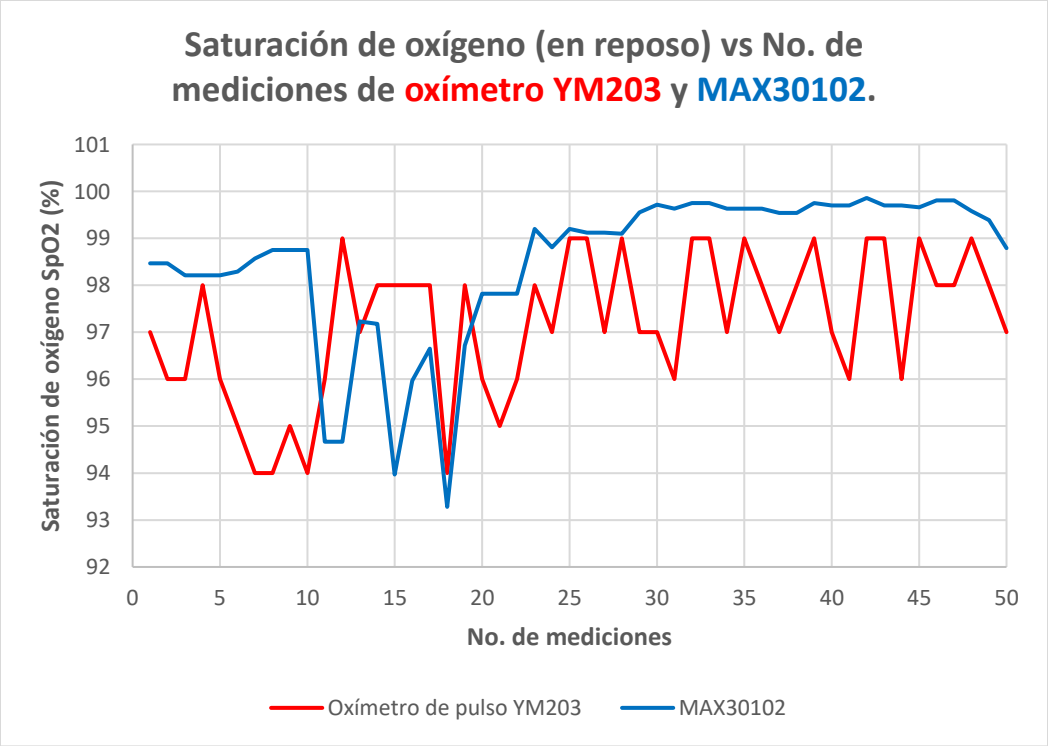


Figura 29: Saturación de oxígeno (en reposo) vs No. de mediciones de oxímetro YM203 y MAX30102.

Tabla 9: Valores de frecuencia cardíaca y saturación de oxígeno del sensor MAX30102 y error porcentual después de realizar un **ejercicio ligero** en comparación con un oxímetro.

No.	Pulse Oximeter		MAX30102		% Error	
	Ritmo cardíaco (bpm)	SpO2 (%)	Ritmo cardíaco (bpm)	SpO2 (%)	Ritmo cardíaco (bpm)	SpO2 (%)
1	81	93	81	94.33	0%	1.43%
2	81	94	81	94.33	0%	0.35%
3	89	98	86	100.50	3%	2.55%
4	80	99	80	96.98	0%	2.04%
5	75	96	76	94.33	1%	1.74%
6	96	94	96	94.33	0%	0.35%
7	81	97	81	98.65	0%	1.70%
8	81	95	79	98.65	2%	3.84%
9	85	96	83	95.35	2%	0.68%
10	74	95	72	95.35	3%	0.37%
11	81	95	83	95.35	2%	0.37%
12	81	99	82	100.71	1%	1.73%
13	89	99	89	100.78	0%	1.80%
14	80	95	79	98.65	1%	3.84%
15	73	97	74	98.65	1%	1.70%
16	80	97	81	98.65	1%	1.70%
17	83	97	82	98.86	1%	1.92%
18	81	97	82	99.09	1%	2.15%
19	98	96	99	98.87	1%	2.99%
20	83	96	82	99.09	1%	3.22%
21	80	94	80	98.03	0%	4.29%
22	83	98	82	98.06	1%	0.06%
23	80	96	81	98.06	1%	2.15%
24	80	97	82	98.06	3%	1.09%
25	80	98	80	97.71	0%	0.30%
26	71	97	71	99.30	0%	2.37%
27	78	96	77	99.30	1%	3.44%
28	82	95	82	99.48	0%	4.72%
29	87	94	86	98.07	1%	4.33%
30	74	96	74	99.33	0%	3.47%
31	80	96	80	99.57	0%	3.72%
32	81	97	82	99.57	1%	2.65%
33	78	96	76	99.57	3%	3.72%
34	81	95	84	99.30	4%	4.53%
35	73	95	72	97.43	1%	2.56%
36	81	96	82	98.64	1%	2.75%
37	81	96	81	98.64	0%	2.75%
38	76	95	75	98.72	1%	3.92%

39	73	98	73	96.36	0%	1.67%
40	75	96	75	95.10	0%	0.94%
41	82	94	81	94.42	1%	0.45%
42	86	97	88	96.50	2%	0.52%
43	89	96	89	96.50	0%	0.52%
44	87	95	88	96.50	1%	1.58%
45	81	96	83	100.25	2%	4.43%
46	77	98	77	100.56	0%	2.61%
47	78	98	78	97.32	0%	0.69%
48	77	96	77	93.62	0%	2.48%
49	73	96	72	100.00	1%	4.17%
50	76	99	76	97.32	0%	1.70%

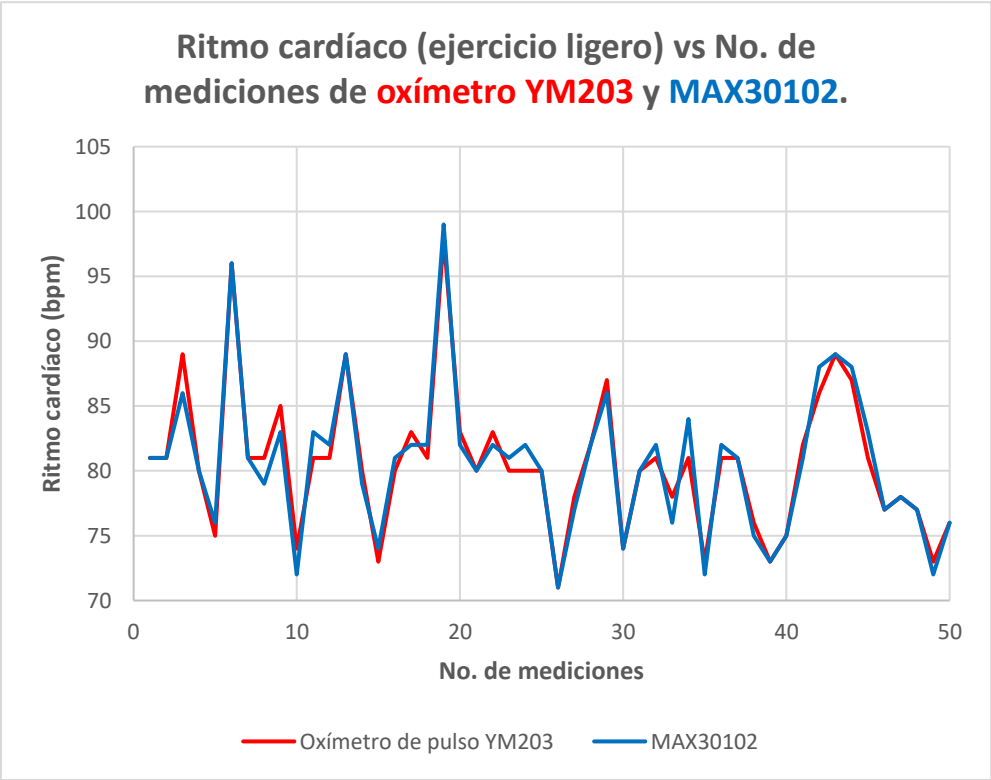


Figura 30: Ritmo cardíaco (ejercicio ligero) vs No. de mediciones de oxímetro YM203 y MAX30102.

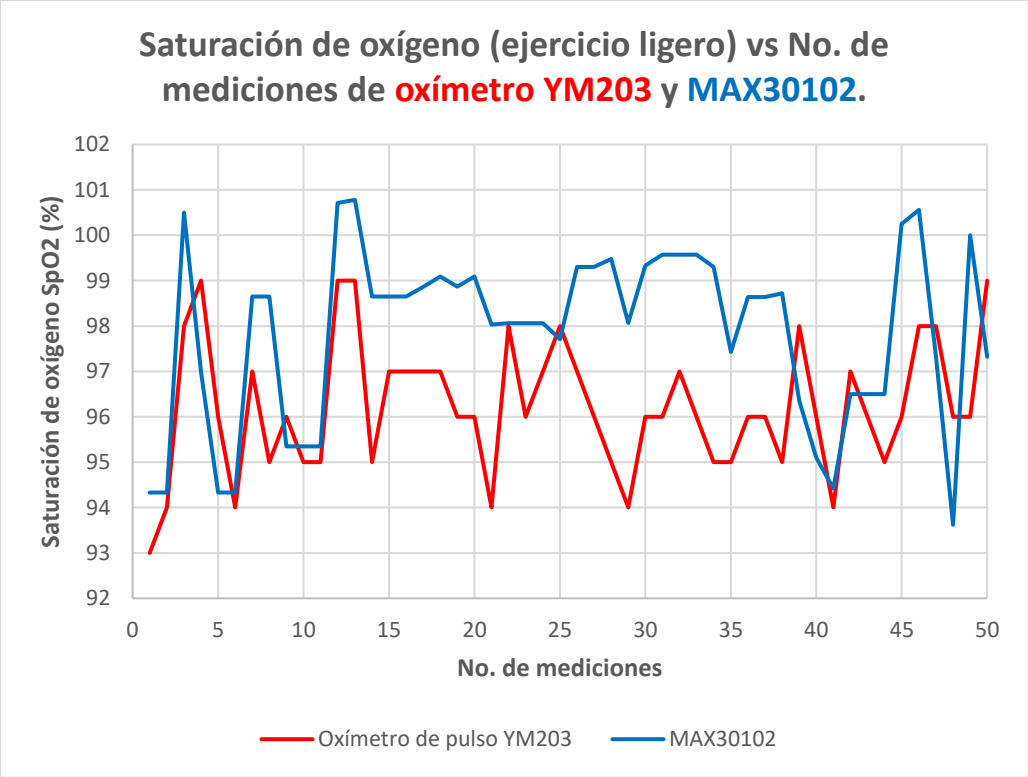


Figura 31: Saturación de oxígeno (ejercicio ligero) vs No. de mediciones de oxímetro YM203 y MAX30102.

Dado que el sensor de pulso XD-58C arrojó valores imprecisos, se consideró la alternativa del sensor MAX30102. Al analizar los resultados, surgieron notables discrepancias en términos de errores porcentuales entre ambos sensores, especialmente en las mediciones de frecuencia cardíaca. En comparación, el MAX30102 presenta un error porcentual máximo del 4% en las mediciones de frecuencia cardíaca en reposo, en relación con la medida estándar proporcionada por el oxímetro (ver tabla 8). Este valor se mantiene constante después de realizar ejercicio ligero. La discrepancia relativamente menor (ver figuras 28 y 30) sugiere que el MAX30102 podría ofrecer mediciones más precisas y coherentes en comparación con el XD-58C. Cabe destacar que el error entre el oxímetro y el sensor MAX30102 en términos de niveles de oxígeno, tanto en reposo como después de realizar ejercicio ligero, resultó ser muy pequeño (ver figuras 29 y 31), como se evidencia en la tabla 9. Esta precisión adicional subraya la relevancia clínica de la exactitud en aplicaciones médicas y de salud. Gracias a esta

mayor precisión, se optó por utilizar el sensor MAX30102 en la siguiente etapa del proyecto, que implica establecer la comunicación con Alexa.

4.4 Conexión con la plataforma Arduino IoT Cloud

4.4.1 Creación de cuenta

Una de las principales características de la plataforma Arduino IoT Cloud es su capacidad para monitorear variables como la frecuencia cardíaca y la saturación de oxígeno a través de la nube. Esto permite acceder a estos datos desde cualquier lugar mediante una conexión Wi-Fi. Además, la plataforma facilita la integración entre el NodeMCU ESP32 y la inteligencia artificial Alexa. Para acceder a la plataforma y crear una cuenta gratuita, se utilizó el siguiente enlace: https://cloud.arduino.cc/ (Arduino Cloud, n.d.). La figura 32 muestra la página principal de acceso a la plataforma, y la figura 33 ilustra la página de registro. Para una guía completa sobre cómo acceder y utilizar la plataforma, se recomienda el siguiente video tutorial, que explica el proceso paso a paso: https://www.youtube.com/watch?v=UFCmTZUoZ1M (DroneBot Workshop, 2021).

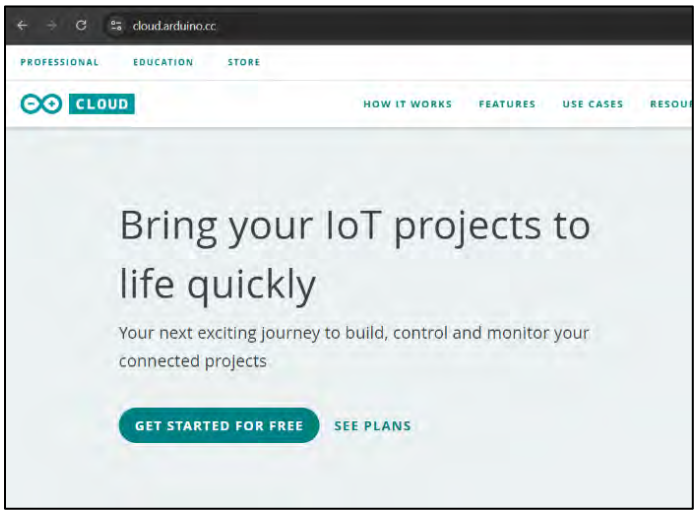


Figura 32: Página principal de Arduino IoT Cloud.

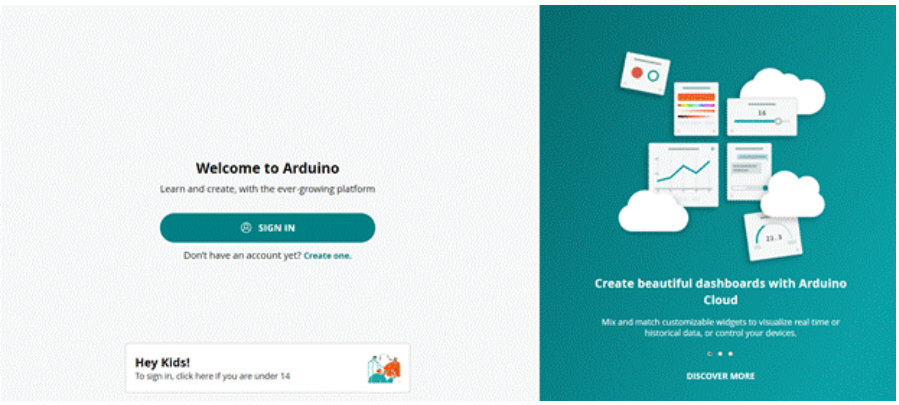


Figura 33: Página de inicio de sesión de Arduino IoT Cloud.

4.4.2 Configuración de "Thing" y vinculación con microcontrolador

Una vez creada la cuenta, se procedió a crear el primer “Thing” (ver figura 34) y se le asignó el nombre “Max_Heart” (ver figura 35).

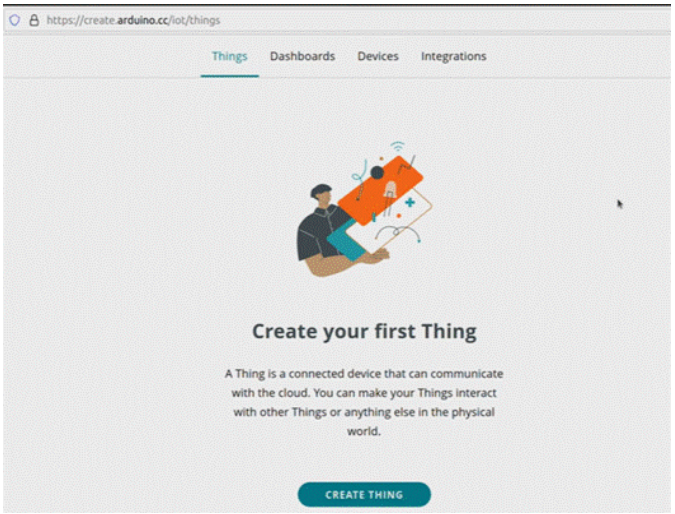


Figura 34: Creación de Thing en Arduino IoT Cloud.

Figura 35: Max_Heart (Thing 1).

El siguiente paso consistió en asociar un dispositivo (ver figura 36), el cual fue el ESP32 Dev Module (ver figura 37), dado que se estaba utilizando un microcontrolador NodeMCU ESP32. El dispositivo se denominó “Muschi” (ver figura 38).

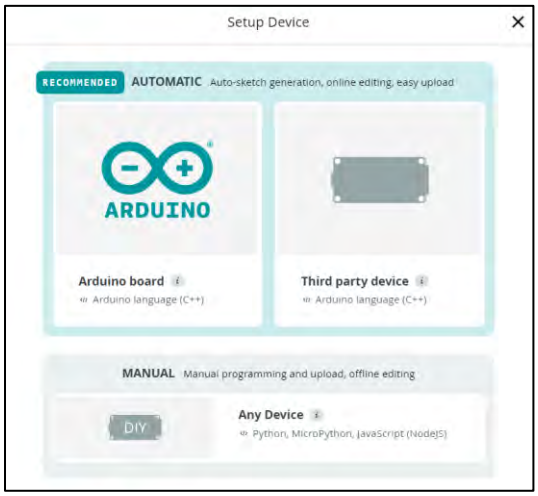


Figura 36: Configuración del dispositivo.

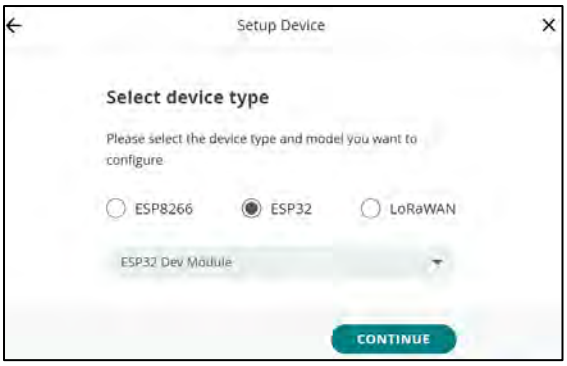


Figura 37: ESP32 Dev Module.

Figura 38: Muschi.

Es importante mencionar que, al asociar este dispositivo, se proporciona una clave secreta necesaria para configurar la red y conectar el dispositivo (ver figura 39).

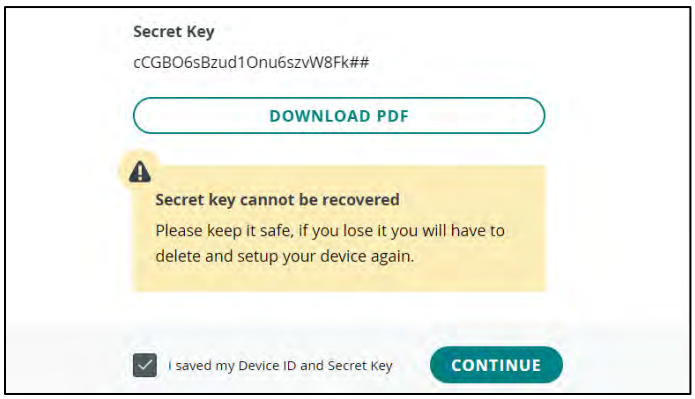


Figura 39: Clave secreta.

Posteriormente, se configura la red WiFi a la que se conectaría el microcontrolador (ver figura 40).

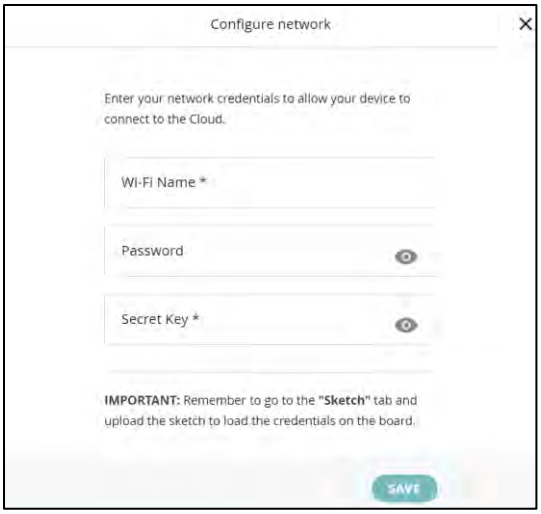


Figura 40: Configuración de Red.

Luego, se procedió a crear las variables que mediría el microcontrolador (ver figura 41). La primera variable se denominó “Heart_beat” y se utilizó para medir el ritmo cardíaco de un individuo. La segunda variable, llamada “ESpO2”, se empleó para medir el nivel de oxígeno en la sangre. Además, se añadió una nueva variable llamada “Temp”, que se utiliza para medir la temperatura ambiental y proporcionar una mejor calibración para el sensor. Las especificaciones de estas variables se pueden consultar en las figuras 42 a 44.

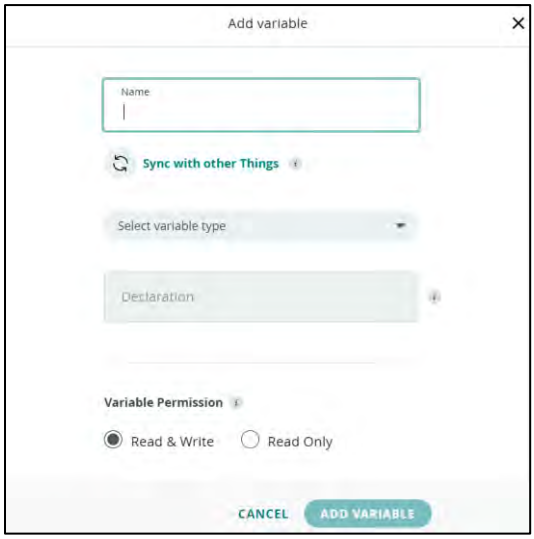


Figura 41: Configuración de variables.

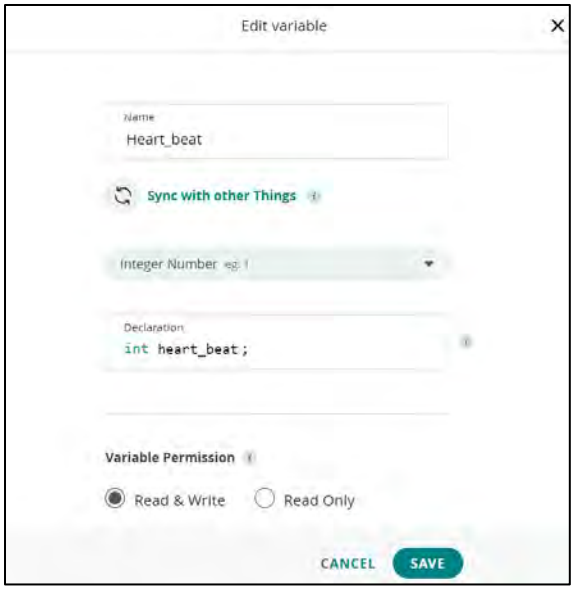


Figura 42: Variable Heart beat.

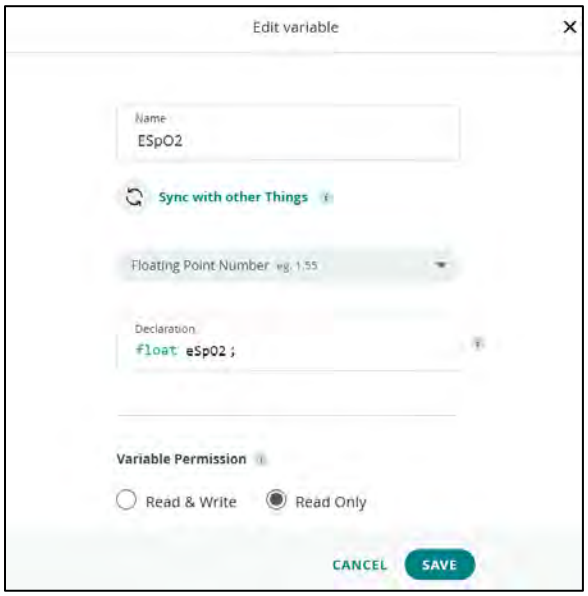


Figura 43: Variable ESpO2.

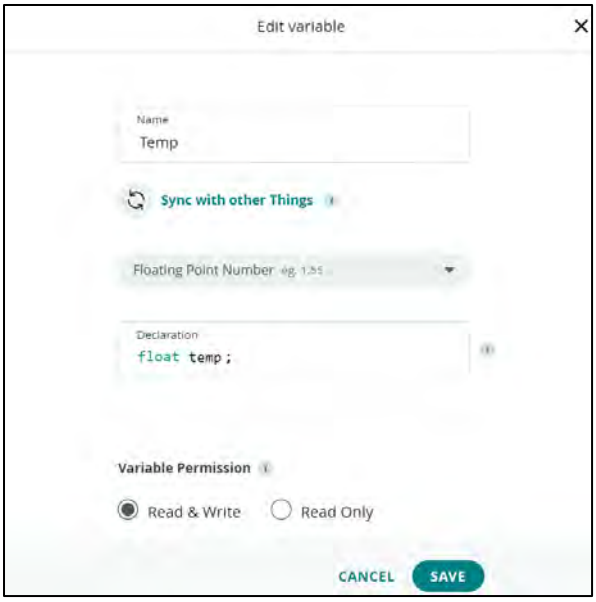


Figura 44: Variable Temp.

La figura 45 muestra claramente las variables en “Max_Heart” y el dispositivo vinculado. Es importante destacar que “Max_Heart” se utiliza exclusivamente para el monitoreo de las variables a través de la aplicación móvil IoT Remote, mediante el dashboard de Arduino IoT Cloud. La conexión con la IA Alexa no es posible, ya que el tipo de variables no es compatible con Alexa (ver figura 46).

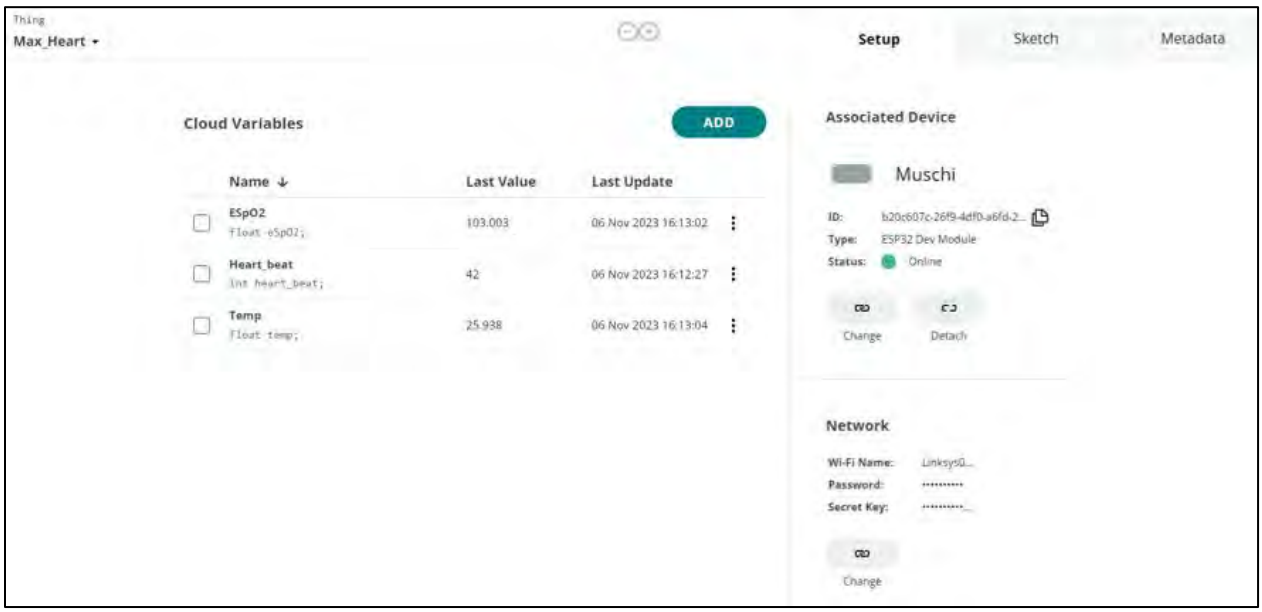


Figura 45: Max_Heart vinculado con Muschi.

Para conectar con Alexa, se creó un segundo “Thing” llamado “AlexaReadsHeartRate”. Se siguieron los mismos pasos que anteriormente para crear la variable “heartrate” (ver figura 47) y se asoció al mismo microcontrolador ya agregado a la plataforma (ver figura 48). Es crucial mencionar que el tipo de variable incluida es compatible con Alexa. Dado que las opciones de variables compatibles con Alexa son limitadas, se eligió “Temperature Sensor (°C)” para establecer la conexión con Alexa.

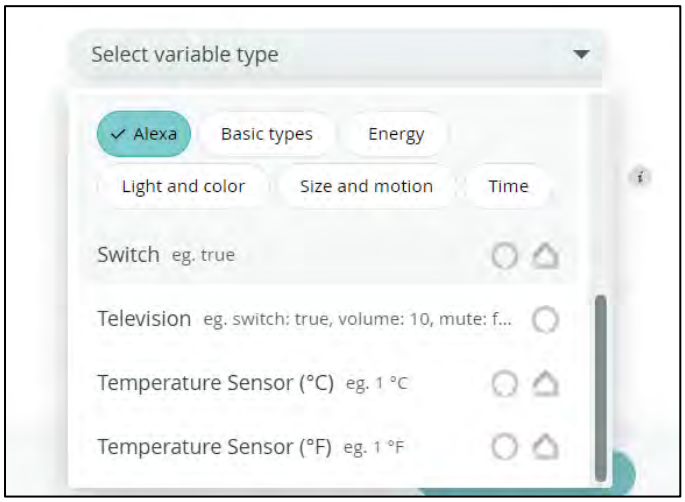


Figura 46: Variables compatible con Alexa.

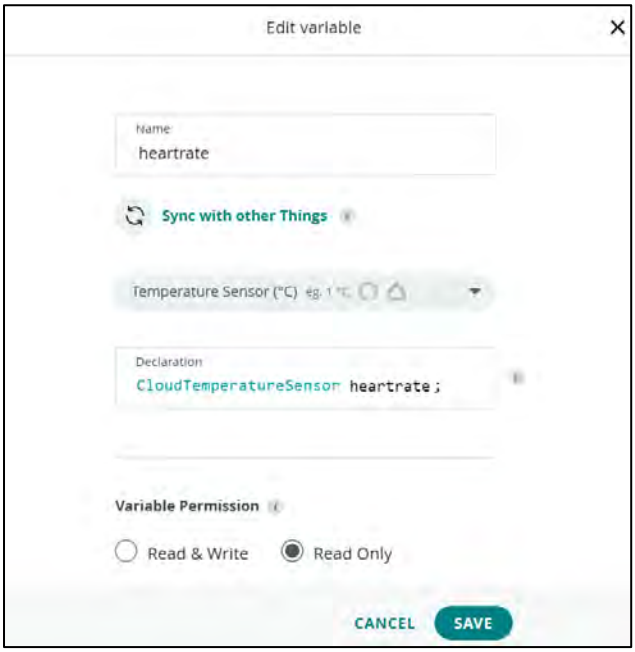


Figura 47: Variable heartrate.

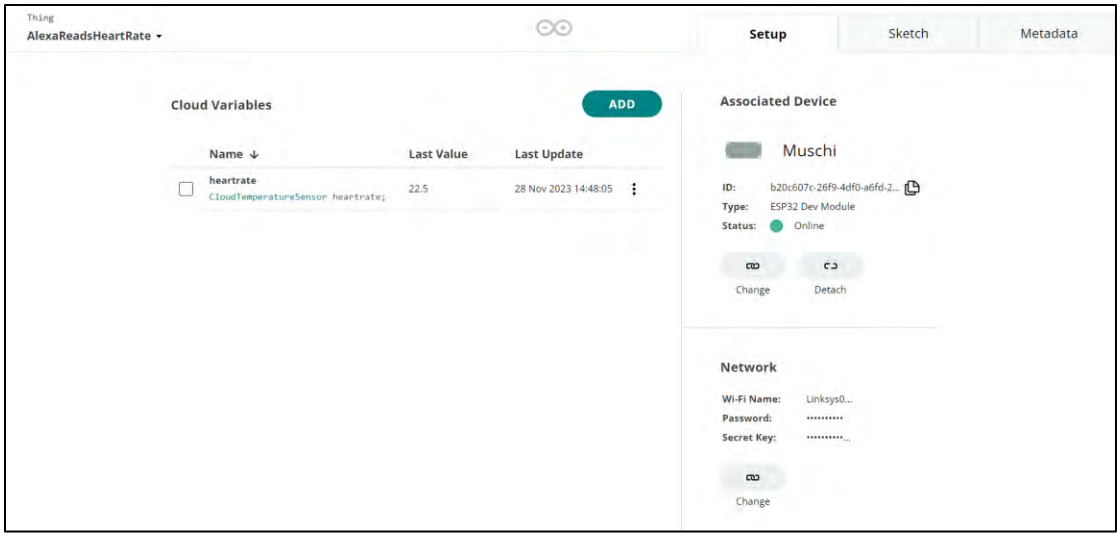


Figura 48: AlexaReadsHeartRate (Thing 2).

4.4.3 Integración y ejecución del programa

En la figura 49 se puede observar que, al crear las variables de “Max_Heart”, estas se guardaron automáticamente en el apartado de “Sketch” de la plataforma Arduino IoT Cloud. Este es el área de trabajo de la plataforma en uso.

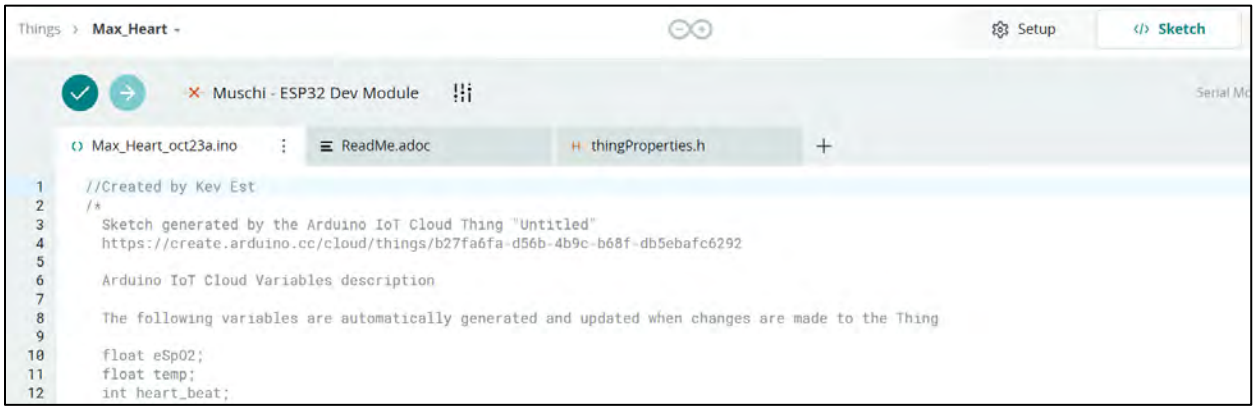


Figura 49: Sketch de Arduino IoT Cloud para Max_Heart.

Antes de proceder a editar en la plataforma, es necesario descargar e instalar el “Arduino Create Agent” y los drivers correspondientes (ver figuras 50 y 51). Una vez instalado, no será necesario volver a instalarlo al utilizar el editor.

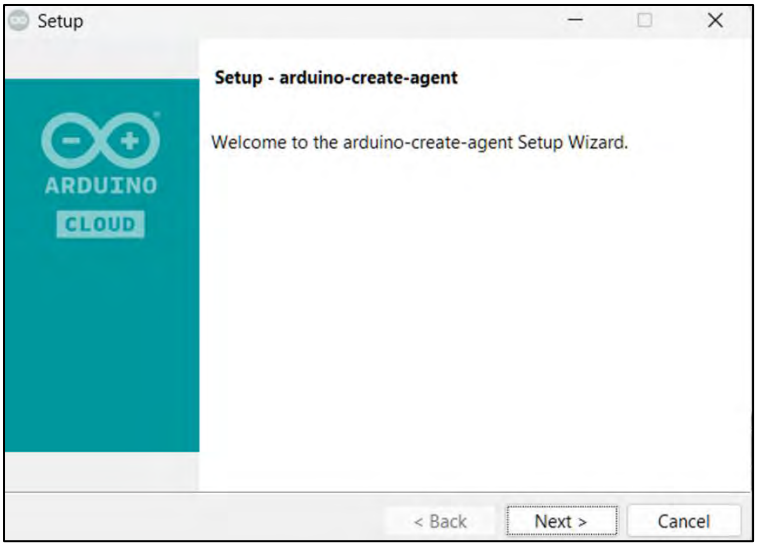


Figura 50: Arduino Create Agent.

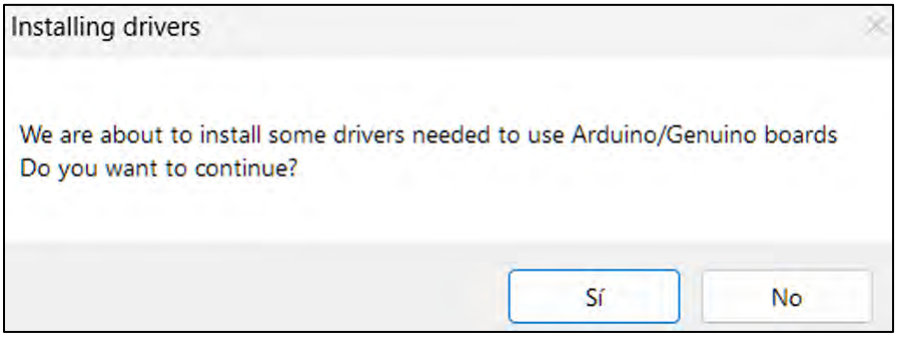


Figura 51: Drivers de Arduino Create Agent.

Es importante resaltar que la plataforma ofrece un editor (ver figura 52) con una interfaz similar a la de Arduino IDE. En este editor, se pueden subir tanto códigos desarrollados en Arduino IDE como bibliotecas que no están disponibles en la plataforma (ver figura 53 y 54). Se cargó el código ya preparado a la ESP32, que se encuentra en el **Anexo B**, junto con las bibliotecas correspondientes.

Figura 52: Editor de Arduino IoT Cloud.

Figura 53: Opción de subir código.

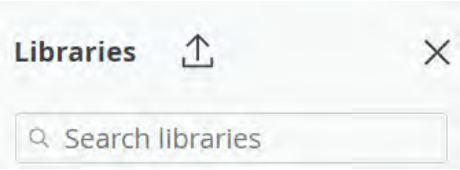


Figura 54: Opción de subir biblioteca.

Es importante señalar que, una vez subido el código, se deben realizar algunas modificaciones, ya que la plataforma proporciona una plantilla para ingresar el código. Esto se debe a que la plataforma incluye una función para conectar el NodeMCU ESP32 a Wi-Fi. El código modificado para “Max_Heart” se muestra en el **Anexo C.**

En esta etapa, el microcontrolador debe estar conectado al ordenador mediante el puerto USB y también al sensor MAX30102. El proceso de verificación y compilación del código en el editor sigue el procedimiento estándar. Una vez verificado y compilado, se puede observar cómo el dispositivo se conecta a la red configurada. Los datos de las variables medidas, como la frecuencia cardíaca (Avg BPM), la saturación de oxígeno (Oxygen %) y la temperatura (Temp), se pueden ver en el monitor serial del editor (ver figura 55).

Es importante resaltar que, en este momento, el microcontrolador ya está conectado a la nube de Arduino IoT Cloud a través de Wi-Fi. En otras palabras, el microcontrolador puede desconectarse del ordenador y conectarse a una fuente de alimentación externa, y continuará generando las mediciones de frecuencia cardíaca, saturación de oxígeno y temperatura.

```
8►►►
```

```
ng...
Place your index finger on the sensor
with steady pressure.
***** Arduino IoT Cloud - configuration
info *****
Device ID: b20c607c-26f9-4df0-a6fd-
2239aa558728
MQTT Broker: mqtts-
up.iot.arduino.cc:8884
WiFi.status(): 255
Connection to "Linksys07982" failed
Retrying in  "4000" milliseconds
Infrared=33219,  Red=112224,
Temp=26°C,  BPM=45.70,  Avg BPM=11.25,
Oxygen % = 25.78%
Infrared=33005,  Red=111466,
Temp=26°C,  BPM=39.71,  Avg BPM=21.00,
Oxygen % = 45.14%
Infrared=32979,  Red=111551,
Temp=26°C,  BPM=67.42,  Avg BPM=37.75,
Oxygen % = 45.14%
Connected to "Linksys07982"
Connected to Arduino IoT Cloud
Thing ID: f11e7dcd-31bb-4ce1-bf4e-
d31d4668b503
Infrared=32827,  Red=110488,
Temp=27°C,  BPM=90.77,  Avg BPM=60.25,
Oxygen % = 56.09%
```

```
Infrared=32820,  Red=110181,
Temp=26°C,  BPM=64.52,  Avg BPM=77.
Oxygen % = 74.49%
Infrared=32804,  Red=110150,
Temp=26°C,  BPM=68.89,  Avg BPM=71.25,
Oxygen % = 74.49%
Infrared=32833,  Red=110100,
Temp=26°C,  BPM=66.82,  Avg BPM=69.50,
Oxygen % = 81.97%
Infrared=32852,  Red=110073,
Temp=26°C,  BPM=68.73,  Avg BPM=66.50,
Oxygen % = 81.97%
Infrared=32863,  Red=110069,
Temp=27°C,  BPM=83.45,  Avg BPM=71.25,
Oxygen % = 81.97%
Infrared=32856,  Red=110174,
Temp=26°C,  BPM=71.86,  Avg BPM=72.00,
Oxygen % = 86.96%
Infrared=33071,  Red=110267,
Temp=26°C,  BPM=77.22,  Avg BPM=74.75,
Oxygen % = 92.05%
```

Figura 57: Monitor serial del Editor para AlexaReadsHeartRate.

4.4.4 Creando un Dashboard en Arduino IoT Cloud

El “Max_Heart” se utilizó para crear un dashboard que permitió monitorear en tiempo real las variables de frecuencia cardíaca, saturación de oxígeno y temperatura del ambiente, tanto a través del Arduino IoT Cloud como en la aplicación Arduino IoT Remote. Los pasos para crear el dashboard fueron los siguientes:

1. Se accedió a la sección de “Dashboards” (ver figura 58).

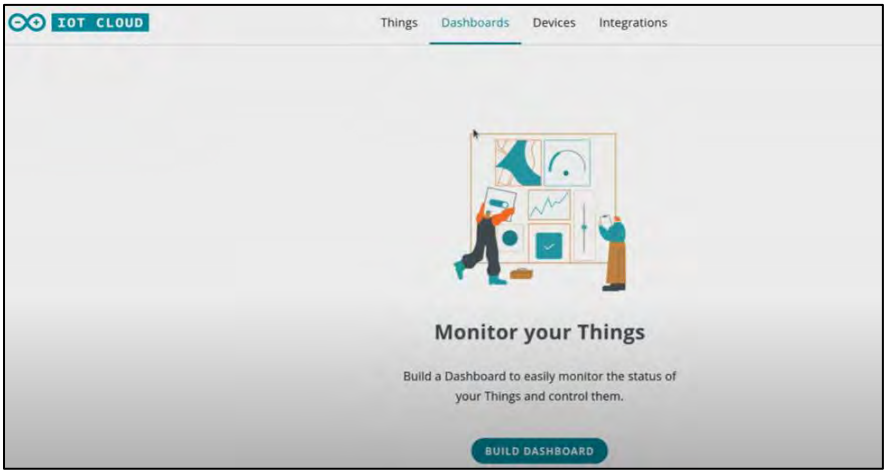


Figura 58: Creando un Dashboard.

2. Se nombró el dashboard como “Max Dashboard” (ver figura 59).

Figura 59: Max Dashboard.

3. Se agregó un widget para la variable de temperatura (Temp) (ver figura 60).

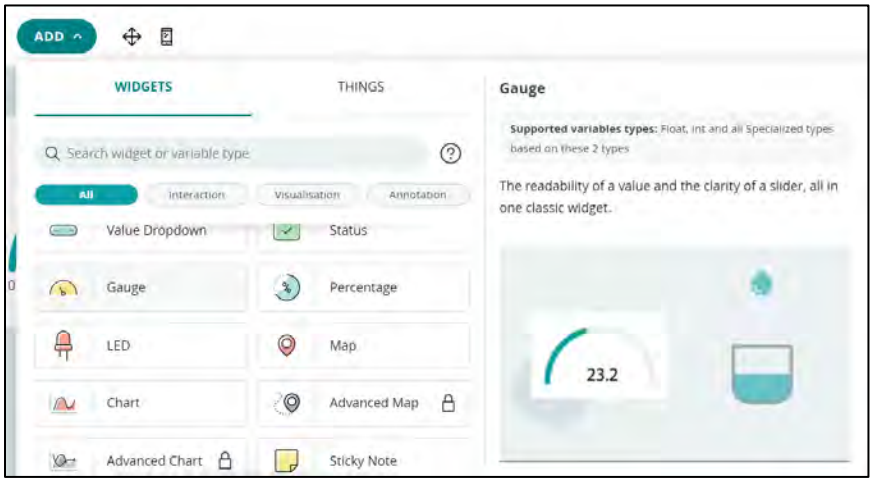


Figura 60: Uso de widgets en Dashboard.

4. Se vinculó el widget con la variable “Temp” existente en “Max_Heart” (ver figura 61).

Figura 61: Enlace entre un widget y una variable.

5. Se configuró el widget, estableciendo los valores mínimo y máximo de la variable para definir un rango representativo (ver figura 62).

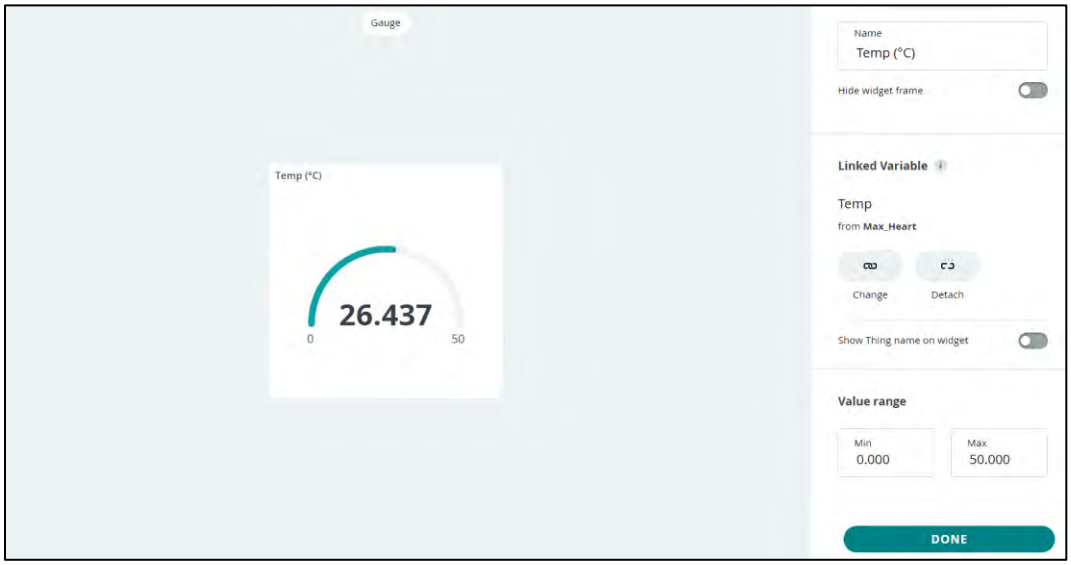


Figura 62: Variable Temp .

6. Se repitió el proceso para las variables “Heart_beat” y “ESpO2” (ver figura 63).

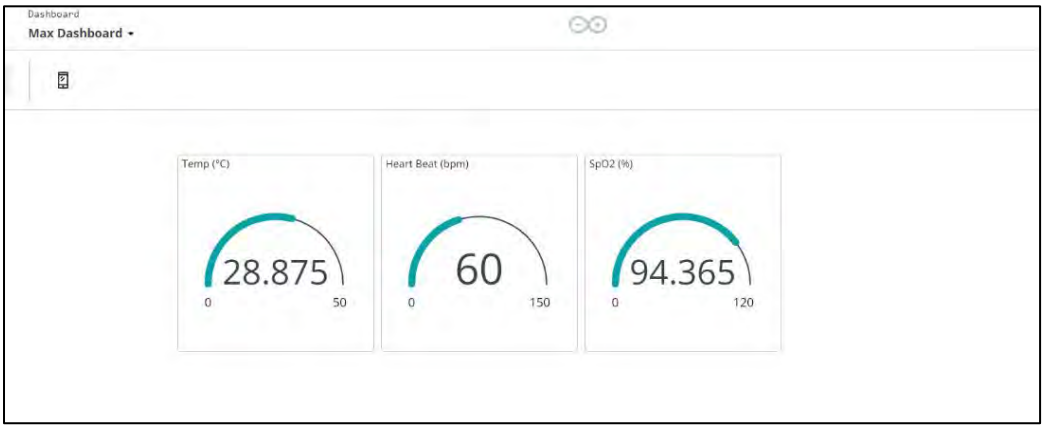
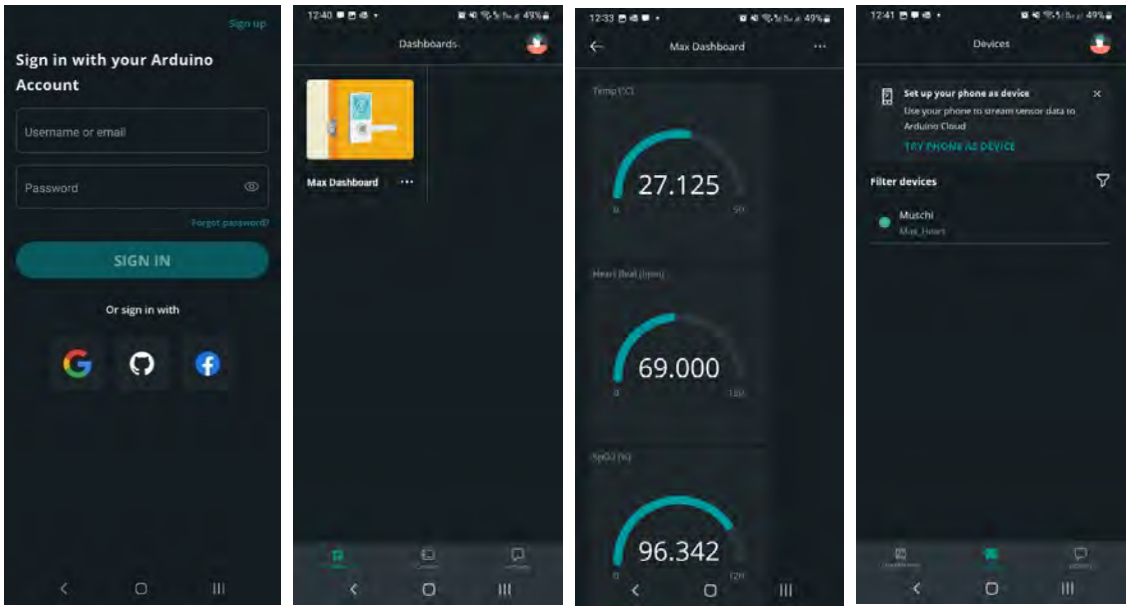


Figura 63: Max Dashboard.

Es importante mencionar que los valores de estas variables se actualizaron en tiempo real mientras el microcontrolador estuvo conectado a la nube de Arduino IoT Cloud.

4.5 Monitoreo de los signos vitales mediante la aplicación IoT Remote.

La aplicación IoT Remote de Arduino ofrece una interfaz intuitiva para el monitoreo eficiente de signos vitales en tiempo real. Al abrir la aplicación IoT Remote de Arduino, descargada desde Google Play Store, se deben ingresar los datos de la cuenta de Arduino IoT Cloud (ver figura 64 a)). En el apartado de dashboards, se encontrará el dashboard “Max Dashboard” previamente creado (ver figura 64 b)). Al acceder a este dashboard, se puede ver claramente que los datos se actualizan en tiempo real mientras el NodeMCU ESP32 esté conectado a la nube (ver figura 64 d)). Los valores que se muestran en la figura 64 c) pueden compararse con los de la figura 63, ya que fueron capturados en la misma instancia. La diferencia entre ellos es mínima debido a las actualizaciones continuas de los datos.



a) b) c) d)

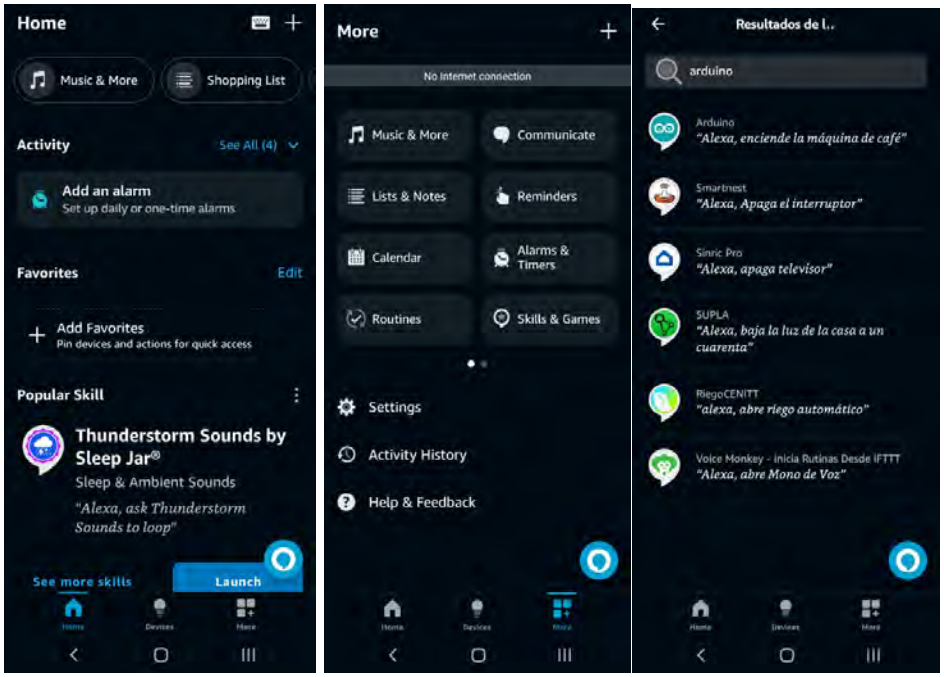
Figura 64: a) Inicio de sesión. b) y c) Max Dashboard. d) NodeMCU ESP32.

Esta aplicación facilita el monitoreo de los signos vitales de cualquier persona en cualquier lugar, siempre que exista una conexión Wi-Fi con acceso a Internet. Resulta

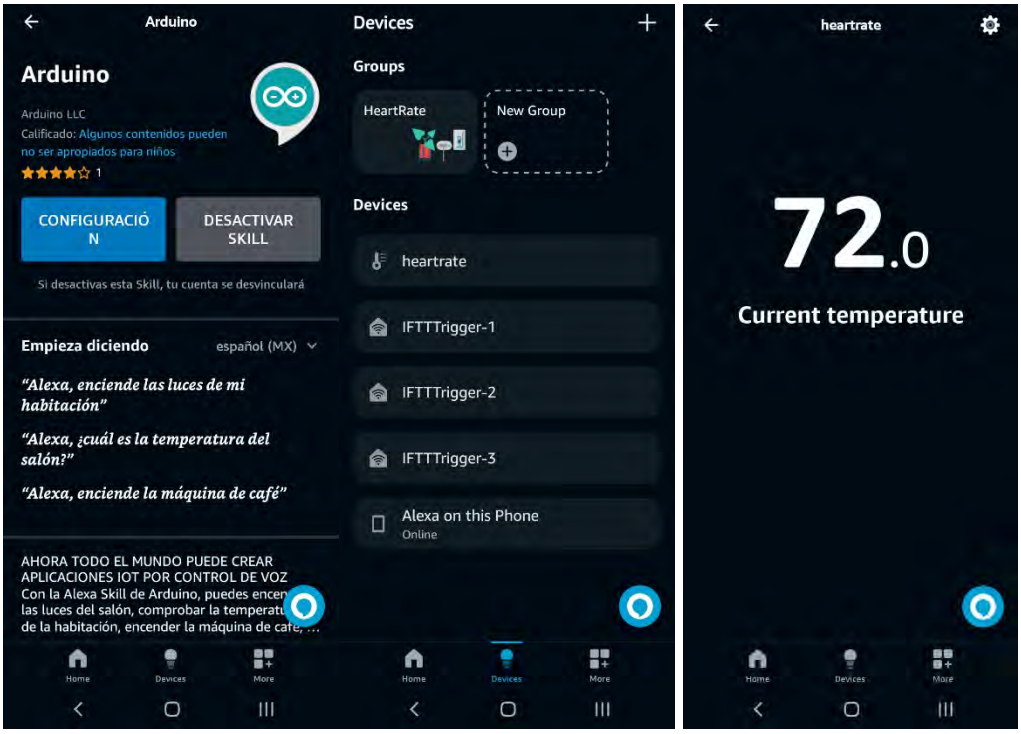
particularmente valiosa para la supervisión de individuos vulnerables, como los de la tercera edad, quienes son más susceptibles a problemas cardíacos.

4.6 Conexión con la IA Alexa

Amazon Alexa se descargó desde Google Play Store y se creó una cuenta para acceder a la aplicación. A continuación, se accedió a la sección “More” y se seleccionó “Skills & Games”. Se buscó “Arduino” y se hizo clic en la habilidad “Arduino” de Arduino LLC. Posteriormente, se activó y vinculó la habilidad Arduino en la aplicación con Amazon Alexa. Una vez completado el proceso de vinculación, el dispositivo fue identificado por la variable “heartrate” y se estableció la comunicación (ver figura 65 a) – e)). Es importante destacar que Amazon Alexa reconoce la variable “heartrate” en lugar del nombre del Thing “AlexaReadsHeartRate” o del dispositivo “Muschi”. La variable “heartrate” se configuró como un sensor de temperatura (°C) debido a la limitación de variables compatibles con Alexa, siendo esta la opción más adecuada. Esta restricción es la razón principal por la que solo se creó una variable para la frecuencia cardíaca y no se incluyó una para la saturación de oxígeno.



a) **b)** **c)**



d) e) f)

Figura 65: a) - e) Conexión del dispositivo a Alexa a través de Arduino IoT Cloud y f) Valor de frecuencia cardíaco mostrado en Amazon Alexa en tiempo real.

Por lo tanto, para consultar el valor de la frecuencia cardíaca (ver figura 65 f)) a través de Alexa, el usuario debe formular la pregunta de la siguiente manera: "Alexa, what is the temperature of the room?" en lugar de “Alexa, what is my heart rate?”. Aunque se adquirió el “Maker Plan” de la plataforma para obtener acceso a más variables, se constató que tanto los planes gratuitos como los de pago están limitados por la misma cantidad de variables compatibles con Alexa. Se espera que en el futuro la plataforma Arduino IoT Cloud amplíe la compatibilidad con más variables para resolver esta limitación.

La conexión del dispositivo NodeMCU ESP32, denominado “Muschi”, fue exitosa gracias a la plataforma Arduino IoT Cloud, que facilita la comunicación entre el NodeMCU ESP32 y Alexa. Este es un logro significativo, ya que demuestra el avance de la tecnología IoT al permitir el monitoreo en tiempo real de signos vitales mediante inteligencia artificial desde cualquier lugar. Esto tiene el potencial de prevenir enfermedades cardiovasculares, especialmente en las personas mayores, que son las más vulnerables.

CAPÍTULO V- Conclusiones y trabajos futuros.

5.1 Conclusión

En conclusión, el sensor de pulso XD-58C mostró errores significativos, con márgenes de hasta el 57% en mediciones en reposo y del 32% después de ejercicio ligero. En contraste, el sensor MAX30102 presentó un error máximo del 4% en las mediciones de frecuencia cardíaca, demostrando una mayor precisión y coherencia. Por ello, se recomienda el MAX30102 por su exactitud superior y menor discrepancia con los valores del oxímetro, especialmente en aplicaciones médicas.

Además, el monitoreo de los signos vitales, como la frecuencia cardíaca y la saturación de oxígeno, se realizó con éxito mediante la plataforma Arduino IoT Cloud y su aplicación IoT Remote. Esta aplicación permitió el seguimiento simultáneo de ambas variables, y la comunicación entre el NodeMCU ESP32, denominado “Muschi”, y Amazon Alexa también fue efectiva, aunque se limitó al monitoreo en tiempo real de la frecuencia cardíaca.

Una limitación significativa de la plataforma Arduino IoT Cloud es su compatibilidad restringida con las variables de Alexa. Debido a esta restricción, se utilizó la variable de sensor de temperatura, lo que resultó en la interpretación de la variable "heartrate" como temperatura. A pesar de esta limitación, el logro subraya el avance de la tecnología IoT, que facilita la supervisión de signos vitales mediante inteligencia artificial y aplicaciones accesibles desde cualquier lugar, con un potencial significativo para prevenir enfermedades cardiovasculares, especialmente en personas mayores.

La aplicación desarrollada,con Arduino IoT Cloud y Amazon Alexa, demuestra un gran potencial para el monitoreo en tiempo real de signos vitales en adultos mayores. Gracias a su interfaz intuitiva y la capacidad de recibir datos actualizados continuamente, es ideal para supervisar a personas vulnerables. Aunque actualmente solo consulta la frecuencia cardíaca, se espera que futuras actualizaciones amplíen su funcionalidad, mejorando su eficacia en la prevención de enfermedades cardiovasculares.

5.2 Trabajo futuro

- En trabajos futuros, se contempla el diseño y la fabricación de un dispositivo portátil y de fácil uso, específicamente para personas mayores. Este dispositivo debe estar diseñado de manera ergonómica para fácil empleo en partes del cuerpo que no provoquen incomodidad al usuario. Este dispositivo permitirá a los usuarios monitorear su frecuencia cardíaca en tiempo real de manera sencilla a través de la aplicación especializada de IoT.
- Se plantea la posibilidad de desarrollar una aplicación web o móvil propia para no depender de las aplicaciones comerciales, ya que como se vio a lo largo del proyecto estas presentan algunas limitaciones en cuanto a necesidades muy específicas se refiere.
- Las pruebas del proyecto se realizaron usando fuentes de voltaje variable de laboratorio, otro reto será diseñar la fuente de energía para la alimentación del circuito, ya que esta debe contar con dimensiones pequeñas, que sea recargable y sobre todo que tenga la autonomía suficiente para el monitoreo consecutivo.
- De igual manera, el diseño del PCB para darle continuidad al proyecto, ya que todas las pruebas se realizaron en protoboard y al momento de la implementación se requiere de un circuito impreso.

Anexos

Anexo A. Código en Arduino IDE para el sensor XD-58C

```
// Código del sensor de ritmo cardíaco XD-58C para el NodeMCU ESP-32

const int led = 2; // LED pin
const int sensor = 4; // Sensor pin
int pulso; // Valor del sensor
int limite = 600; // Valor umbral
int bpm; // Ritmo cardíaco (latidos/minuto)

 void setup()
 {
  Serial.begin(115200);// Comunicación serial

  pinMode(sensor, INPUT); // Modo del pin 'sensor' en INPUT
  pinMode(led, OUTPUT); //Modo del pin ´led' en OUTPUT
  digitalWrite(led, LOW); // Apagar led
}

void loop()
 {
  pulso = analogRead(sensor); // leer el valor del sensor

  // Detección de pulso
  if (pulso > limite) // Valor del sensor mayor al limite
  {
    digitalWrite(led, HIGH); // Encender led
    delay(100); // Esperar 100ms

    digitalWrite(led, LOW); // Apagar led
    bpm = 60000 / pulso; // Calculo del ritmo cardíaco
    /*Serial.print("Heart rate: ");
    Serial.print(bpm);// Imprime el valor de ritmo cardíaco
```

```
    Serial.println(" BPM");*/


// Crea un mensaje con información de ritmo cardíaco
    String message = "Heart Rate: " + String(bpm) + " BPM";
    Serial.println(message);// Imprime el mensaje de ritmo cardíaco en el monitor
serie
delay (1000);

  } else {

    Serial.println("No se detectó pulso cardíaco");
  }
  delay(1000);// Esperar 1000ms
  }
```

Anexo B. Código en Arduino IDE para el sensor MAX30102

```
// Código del sensor de ritmo cardíaco y nivel de oxigeno MAX30102 para el NodeMCU
ESP-32.

// Librerias necesarias para el ritmo cardiaco y el nivel de oxígeno
#include <Wire.h>
#include "MAX30105.h"
#include "heartRate.h"

MAX30105 particleSensor;

// Ritmo cardíaco
const byte RATE_SIZE = 4; // Aumentar para obtener más promedios (4).
byte rates[RATE_SIZE]; // Matriz de ritmo cardíacos
byte rateSpot = 0;
long lastBeat = 0; // Tiempo en la que ocurrío el último latido.
float beatsPerMinute;
int heartrate;
```

```
//SpO2 (nivel de oxigeno)
double avered    = 0;
double aveir     = 0;
double sumirrms  = 0;
double sumredrms = 0;
int    i         = 0;
int    Num       = 100;  // Calcular SpO2 según este intervalo de muestreo
int    temp;             // Utilizado para compensar cambios ambientales
float  ESpO2;            // Valor inicial de SpO2 estimado
double FSpO2     = 0.7;  // Factor de filtrado para SpO2 estimado
double frate     = 0.95; // Filtro de paso bajo para valor de IR/LED rojo para
eliminar el componente de CA.
#define TIMETOBOOT 3000  // Tiempo de espera (ms) para generar SpO2.
#define SCALE      88.0  // Ajustar para mostrar los latidos del corazón y SpO2 en la
misma escala
#define SAMPLING   100 //25 //5     // Para ver los latidos del corazón con mayor
precisión, configurar SAMPLING en 1
#define FINGER_ON  30000 // Si la señal roja es inferior a esto, indica que su dedo
no está en el sensor.
#define USEFIFO

// Void Setup
void setup()
{
  Serial.begin(115200);
  Serial.println("Inicializando...");

  // Inicializar sensor
  if (!particleSensor.begin(Wire, I2C_SPEED_FAST)) // Utilice el puerto I2C
predeterminado, velocidad de 400 kHz
  {
    Serial.println("No se encontró MAX30105.Por favor revise el
cableado/alimentación.");
    while (1);
  }
```

```
  // Ritmo cardíaco
  Serial.println("Coloque su dedo índice sobre el sensor con presión constante.");

  particleSensor.setup(); // Configurar el sensor con la configuración predeterminada
  particleSensor.setPulseAmplitudeRed(0x0A); // Ajustar el LED rojo a nivel bajo para
indicar que el sensor está funcionando
  particleSensor.setPulseAmplitudeGreen(0); // Apague el LED verde



  particleSensor.enableDIETEMPRDY();// Habilita la interrupción DIETEMPRDY (es
requerido).
}


//Void Loop
void loop()
{

  //SpO2
  uint32_t ir, red, green;
  double fred, fir;
  double SpO2 = 0; // SpO2 (sin procesar) antes del filtrado de low pass

#ifdef USEFIFO
  particleSensor.check(); // Comprueba el sensor, lee hasta 3 muestras

  while (particleSensor.available()) {// Comprueba la presencia de nuevos datos.
#ifdef MAX30105
   red = particleSensor.getFIFORed(); //Sparkfun's MAX30105
   ir  = particleSensor.getFIFOIR();  //Sparkfun's MAX30105
#else
   red = particleSensor.getFIFOIR();  //why getFOFOIR output Red data by MAX30102 on
MH-ET LIVE breakout board
   ir  = particleSensor.getFIFORed(); //why getFIFORed output IR data by MAX30102 on
MH-ET LIVE breakout board
```

```
#endif

// Cálculo de SpO2 PARTE 1
    i++;
    fred = (double)red;
    fir  = (double)ir;
    avered = avered * frate + (double)red * (1.0 - frate); // Nivel de rojo promedio
por filtro de low pass
    aveir = aveir * frate + (double)ir * (1.0 - frate); // Nivel IR promedio por
filtro de low pass
    sumredrms += (fred - avered) * (fred - avered); // Suma cuadrada del componente
alternativo del nivel rojo
    sumirrms += (fir - aveir) * (fir - aveir); // Suma cuadrada del componente
alternativo del nivel IR


    // Temperatura para compensar cambios ambientales
    float temp = particleSensor.readTemperature();

  // Ritmo cardíaco
  //long irValue = particleSensor.getIR();

  if (checkForBeat(ir) == true)
  {
    // Se detecta un latido
    long delta = millis() - lastBeat;
    lastBeat = millis();

    beatsPerMinute = 60 / (delta / 1000.0);

    if (beatsPerMinute < 255 && beatsPerMinute > 20)
    {
      rates[rateSpot++] = (byte)beatsPerMinute; // Almacene esta lectura en la matriz
      rateSpot %= RATE_SIZE; // Variable de ajuste

      // Toma promedio de lecturas
```

```
    heartrate = 0;
    for (byte x = 0 ; x < RATE_SIZE ; x++)
      heartrate += rates[x];
    heartrate /= RATE_SIZE;



  // Ritmo cardíaco, temperatura y SpO2
  if (ir < FINGER_ON){ // No hay dedo en el sensor
         Serial.println("No se detectó ningún dedo");
         break;
      }
      if(ir > FINGER_ON){
         //Temperature = mlx.readObjectTempC();

          // Imprime valores de ritmo cardíaca, temperatura y
SpO2.
          Serial.print("Infrarojo=");
          Serial.print(ir);
          Serial.print(",  ");
          Serial.print("Rojo=");
          Serial.print(red);
          Serial.print(",  ");
          Serial.print("Temp=");
          Serial.print(temp);
          Serial.print("°C,  ");
          Serial.print("BPM=");
          Serial.print(beatsPerMinute);
          Serial.print(",  ");
          Serial.print("Avg BPM=");
          Serial.print(heartrate);
          Serial.print(",  ");
          Serial.print("Oxígeno % = ");
          Serial.print(ESpO2);
          Serial.println("%");
  }
```

```
  }


  }

// Cálculo de SpO2 PARTE 2
    if ((i % Num) == 0) {
      double R = (sqrt(sumredrms)/ avered) / (sqrt(sumirrms) / aveir);

      // Serial.println(R);
      SpO2 = -23.3 * (R - 0.4) + 100; // Calcular SpO2
      ESpO2 = FSpO2 * ESpO2 + (1.0 - FSpO2) * SpO2;//filtro low pass
      //  Serial.print(SpO2);Serial.print(",");Serial.println(ESpO2);
      sumredrms = 0.0; sumirrms = 0.0; i = 0;
      break;
    }
    particleSensor.nextSample(); //Listo con esta muestra. Pasar a la siguiente
muestra
   // Serial.println(SpO2);

}
#endif
```

Anexo C. Código en Arduino IoT Cloud para Max_Heart (Thing 1)

```
/*
  Sketch generated by the Arduino IoT Cloud Thing "Max_Heart"
  https://create.arduino.cc/cloud/things/b27fa6fa-d56b-4b9c-b68f-db5ebafc6292

  Arduino IoT Cloud Variables description

  The following variables are automatically generated and updated when changes are made to the Thing

  float eSpO2;
  float temp;
  int heart_beat;
```

```
  Variables which are marked as READ/WRITE in the Cloud Thing will also have functions
  which are called when their values are changed from the Dashboard.
  These functions are generated with the Thing and added at the end of this sketch.
*/

#include "thingProperties.h"

#include <Wire.h>

// SparkFun MAX3010x Pulse and Proximity Sensor Library - Version: Latest
#include <MAX30105.h>
#include <heartRate.h>


MAX30105 particleSensor;

//Heart Rate Monitor
const byte RATE_SIZE = 4; //Increase this for more averaging. 4 is good.
byte rates[RATE_SIZE]; //Array of heart rates
byte rateSpot = 0;
long lastBeat = 0; //Time at which the last beat occurred
float beatsPerMinute;
//int heart_beat;

//SpO2 Monitor
double avered    = 0;
double aveir     = 0;
double sumirrms  = 0;
double sumredrms = 0;
int    i         = 0;
int    Num       = 100;  // calculate SpO2 by this sampling interval
//int    temp;            // body temperature
//float  eSpO2;           // initial value of estimated SpO2
double FSpO2     = 0.7;  // filter factor for estimated SpO2
double frate     = 0.95; // low pass filter for IR/red LED value to eliminate AC component
#define TIMETOBOOT 3000  // wait for this time(msec) to output SpO2
```

```
#define SCALE      88.0  // adjust to display heart beat and SpO2 in the same scale
#define SAMPLING   100 //25 //5     // if you want to see heart beat more precisely, set SAMPLING to 1
#define FINGER_ON  30000 // if red signal is lower than this, it indicates your finger is not on the sensor
#define USEFIFO


void setup() {
  // Initialize serial and wait for port to open:
  Serial.begin(9600);
  // This delay gives the chance to wait for a Serial Monitor without blocking if none is found

   Serial.println("Initializing...");

  delay(1500);

  // Defined in thingProperties.h
  initProperties();

// Initialize sensor
  if (!particleSensor.begin(Wire, I2C_SPEED_FAST)) //Use default I2C port, 400kHz speed
  {
    Serial.println("MAX30105 was not found. Please check wiring/power. ");
    while (1);
  }


  //Heart Rate Monitor
  Serial.println("Place your index finger on the sensor with steady pressure.");

  particleSensor.setup(); //Configure sensor with default settings
  particleSensor.setPulseAmplitudeRed(0x0A); //Turn Red LED to low to indicate sensor is running
  particleSensor.setPulseAmplitudeGreen(0); //Turn off Green LED



  //Temperature Monitor
  particleSensor.enableDIETEMPRDY();//Enable the temp ready interrupt. This is required.
```

```
  // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  /*
    The following function allows you to obtain more information
    related to the state of network and IoT Cloud connection and errors
    the higher number the more granular information you'll get.
    The default is 0 (only errors).
    Maximum is 4
 */
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}

//Void Loop
void loop() {
  ArduinoCloud.update();
  // Your code here

  //SpO2 Monitor
  uint32_t ir, red, green;
  double fred, fir;
  double SpO2 = 0; //raw SpO2 before low pass filtered

#ifdef USEFIFO
  particleSensor.check(); //Check the sensor, read up to 3 samples

  while (particleSensor.available()) {//do we have new data
#ifdef MAX30105
   red = particleSensor.getFIFORed(); //Sparkfun's MAX30105
   ir  = particleSensor.getFIFOIR();  //Sparkfun's MAX30105
#else
   red = particleSensor.getFIFOIR();  //why getFOFOIR output Red data by MAX30102 on MH-ET LIVE breakout
board
   ir  = particleSensor.getFIFORed(); //why getFIFORed output IR data by MAX30102 on MH-ET LIVE breakout
board
#endif
```

```
  i++;
  fred = (double)red;
  fir  = (double)ir;
  avered = avered * frate + (double)red * (1.0 - frate); //average red level by low pass filter
  aveir = aveir * frate + (double)ir * (1.0 - frate); //average IR level by low pass filter
  sumredrms += (fred - avered) * (fred - avered); //square sum of alternate component of red level
  sumirrms += (fir - aveir) * (fir - aveir);//square sum of alternate component of IR level



  //Temperature Monitor
  temp = particleSensor.readTemperature();


//Heart Rate Monitor
//long irValue = particleSensor.getIR();

if (checkForBeat(ir) == true)
{
  //We sensed a beat!
  long delta = millis() - lastBeat;
  lastBeat = millis();

  beatsPerMinute = 60 / (delta / 1000.0);

  if (beatsPerMinute < 255 && beatsPerMinute > 20)
  {
    rates[rateSpot++] = (byte)beatsPerMinute; //Store this reading in the array
    rateSpot %= RATE_SIZE; //Wrap variable

    //Take average of readings
    heart_beat = 0;
    for (byte x = 0 ; x < RATE_SIZE ; x++)
      heart_beat += rates[x];
    heart_beat /= RATE_SIZE;
```

```
 //Heart Rate, Temperature, and SpO2 Monitor
 if (ir < FINGER_ON){ // no finger on the sensor
       Serial.println("No finger detected");
       break;
    }
    if(ir > FINGER_ON){
       //Temperature = mlx.readObjectTempC();

        //Prints Values of Heart Rate, Temperature, and SpO2.
        Serial.print("Infrared=");
        Serial.print(ir);
        Serial.print(",  ");
        Serial.print("Red=");
        Serial.print(red);
        Serial.print(",  ");
        Serial.print("Temp=");
        Serial.print(temp);
        Serial.print("°C,  ");
        Serial.print("BPM=");
        Serial.print(beatsPerMinute);
        Serial.print(",  ");
        Serial.print("Avg BPM=");
        Serial.print(heart_beat);
        Serial.print(",  ");
        Serial.print("Oxygen % = ");
        Serial.print(eSpO2);
        Serial.println("%");
  }
}


}

 if ((i % Num) == 0) {
   double R = (sqrt(sumredrms)/ avered) / (sqrt(sumirrms) / aveir);
```

```
      // Serial.println(R);
      SpO2 = -23.3 * (R - 0.4) + 100; //Calculating SpO2 (view this link: https://www.analog.com/en/technical-
articles/guidelines-for-spo2-measurement--maxim-integrated.html)
      eSpO2 = FSpO2 * eSpO2 + (1.0 - FSpO2) * SpO2;//low pass filter
      //  Serial.print(SpO2);Serial.print(",");Serial.println(ESpO2);
      sumredrms = 0.0; sumirrms = 0.0; i = 0;
      break;
    }
    particleSensor.nextSample(); //We're finished with this sample so move to next sample
   // Serial.println(SpO2);

}
#endif



}
```

Anexo D. Código en Arduino IoT Cloud para AlexaReadsHeartRate (Thing 2)

```
/*
  Sketch generated by the Arduino IoT Cloud Thing "AlexaReadsHeartRate"
  https://create.arduino.cc/cloud/things/f11e7dcd-31bb-4ce1-bf4e-d31d4668b503

  Arduino IoT Cloud Variables description

  The following variables are automatically generated and updated when changes are made to the Thing

  CloudTemperatureSensor heartrate;

  Variables which are marked as READ/WRITE in the Cloud Thing will also have functions
  which are called when their values are changed from the Dashboard.
  These functions are generated with the Thing and added at the end of this sketch.
*/
```

```
#include "thingProperties.h"

#include <Wire.h>

// SparkFun MAX3010x Pulse and Proximity Sensor Library - Version: Latest
#include <MAX30105.h>
#include <heartRate.h>


MAX30105 particleSensor;

//Heart Rate Monitor
const byte RATE_SIZE = 4; //Increase this for more averaging. 4 is good.
byte rates[RATE_SIZE]; //Array of heart rates
byte rateSpot = 0;
long lastBeat = 0; //Time at which the last beat occurred
float beatsPerMinute;
//int heart_beat;

//SpO2 Monitor
double avered    = 0;
double aveir     = 0;
double sumirrms  = 0;
double sumredrms = 0;
int    i         = 0;
int    Num       = 100;  // calculate SpO2 by this sampling interval
int    temp;             // body temperature
float  eSpO2;            // initial value of estimated SpO2
double FSpO2     = 0.7;  // filter factor for estimated SpO2
double frate     = 0.95; // low pass filter for IR/red LED value to eliminate AC component
#define TIMETOBOOT 3000  // wait for this time(msec) to output SpO2
#define SCALE      88.0  // adjust to display heart beat and SpO2 in the same scale
#define SAMPLING   100 //25 //5     // if you want to see heart beat more precisely, set SAMPLING to 1
#define FINGER_ON  30000 // if red signal is lower than this, it indicates your finger is not on the sensor
#define USEFIFO


void setup() {
```

```
  // Initialize serial and wait for port to open:
  Serial.begin(9600);
  // This delay gives the chance to wait for a Serial Monitor without blocking if none is found

   Serial.println("Initializing...");

  delay(1500);

  // Defined in thingProperties.h
  initProperties();

 // Initialize sensor
  if (!particleSensor.begin(Wire, I2C_SPEED_FAST)) //Use default I2C port, 400kHz speed
  {
    Serial.println("MAX30105 was not found. Please check wiring/power. ");
    while (1);
  }


  //Heart Rate Monitor
  Serial.println("Place your index finger on the sensor with steady pressure.");

  particleSensor.setup(); //Configure sensor with default settings
  particleSensor.setPulseAmplitudeRed(0x0A); //Turn Red LED to low to indicate sensor is running
  particleSensor.setPulseAmplitudeGreen(0); //Turn off Green LED



  //Temperature Monitor
  particleSensor.enableDIETEMPRDY();//Enable the temp ready interrupt. This is required.


  // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  /*
    The following function allows you to obtain more information
    related to the state of network and IoT Cloud connection and errors
```

```
    the higher number the more granular information you'll get.
    The default is 0 (only errors).
    Maximum is 4
 */
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}

//Void Loop
void loop() {
  ArduinoCloud.update();
  // Your code here

  //SpO2 Monitor
  uint32_t ir, red, green;
  double fred, fir;
  double SpO2 = 0; //raw SpO2 before low pass filtered

#ifdef USEFIFO
  particleSensor.check(); //Check the sensor, read up to 3 samples

  while (particleSensor.available()) {//do we have new data
#ifdef MAX30105
   red = particleSensor.getFIFORed(); //Sparkfun's MAX30105
   ir  = particleSensor.getFIFOIR();  //Sparkfun's MAX30105
#else
   red = particleSensor.getFIFOIR();  //why getFOFOIR output Red data by MAX30102 on MH-ET LIVE breakout
board
   ir  = particleSensor.getFIFORed(); //why getFIFORed output IR data by MAX30102 on MH-ET LIVE breakout
board
#endif

    i++;
    fred = (double)red;
    fir  = (double)ir;
    avered = avered * frate + (double)red * (1.0 - frate); //average red level by low pass filter
    aveir = aveir * frate + (double)ir * (1.0 - frate); //average IR level by low pass filter
```

```
  sumredrms += (fred - avered) * (fred - avered); //square sum of alternate component of red level
  sumirrms += (fir - aveir) * (fir - aveir);//square sum of alternate component of IR level



  //Temperature Monitor
  temp = particleSensor.readTemperature();


 //Heart Rate Monitor
 //long irValue = particleSensor.getIR();

 if (checkForBeat(ir) == true)
 {
  //We sensed a beat!
  long delta = millis() - lastBeat;
  lastBeat = millis();

  beatsPerMinute = 60 / (delta / 1000.0);

  if (beatsPerMinute < 255 && beatsPerMinute > 20)
  {
   rates[rateSpot++] = (byte)beatsPerMinute; //Store this reading in the array
   rateSpot %= RATE_SIZE; //Wrap variable

   //Take average of readings
   heartrate = 0;
   for (byte x = 0 ; x < RATE_SIZE ; x++)
    heartrate += rates[x];
    heartrate /= RATE_SIZE;



  //Heart Rate, Temperature, and SpO2 Monitor
  if (ir < FINGER_ON){ // no finger on the sensor
       Serial.println("No finger detected");
       break;
    }
    if(ir > FINGER_ON){
```

```
        //Temperature = mlx.readObjectTempC();


         //Prints Values of Heart Rate, Temperature, and SpO2.
         Serial.print("Infrared=");
         Serial.print(ir);
         Serial.print(",  ");
         Serial.print("Red=");
         Serial.print(red);
         Serial.print(",  ");
         Serial.print("Temp=");
         Serial.print(temp);
         Serial.print("°C,  ");
         Serial.print("BPM=");
         Serial.print(beatsPerMinute);
         Serial.print(",  ");
         Serial.print("Avg BPM=");
         Serial.print(heartrate);
         Serial.print(",  ");
         Serial.print("Oxygen % = ");
         Serial.print(eSpO2);
         Serial.println("%");


    }
  }


}

  if ((i % Num) == 0) {
    double R = (sqrt(sumredrms)/ avered) / (sqrt(sumirrms) / aveir);

    // Serial.println(R);
    SpO2 = -23.3 * (R - 0.4) + 100; //Calculating SpO2 (view this link: https://www.analog.com/en/technical-articles/guidelines-for-spo2-measurement--maxim-integrated.html)
    eSpO2 = FSpO2 * eSpO2 + (1.0 - FSpO2) * SpO2;//low pass filter
```

```
      //  Serial.print(SpO2);Serial.print(",");Serial.println(ESpO2);
      sumredrms = 0.0; sumirrms = 0.0; i = 0;
      break;
    }
    particleSensor.nextSample(); //We're finished with this sample so move to next sample
   // Serial.println(SpO2);

}
#endif

}
```

Referencias

- Abuzairi, T., Vinia, E., Yudhistira, M. A., Rizkinia, M., & Eriska, W. (2024). A dataset of hemoglobin blood value and photoplethysmography signal for machine learning-based non-invasive hemoglobin measurement. *Data in Brief*, *52*, 109823. https://doi.org/10.1016/j.dib.2023.109823
- *Alexa*. (n.d.). Retrieved September 5, 2024, from https://developer.amazon.com/en-GB/alexa
- Arduino Cloud. (n.d.). *Arduino Cloud | Build, Control, monitor your IoT Projects*. Retrieved December 5, 2023, from https://cloud.arduino.cc/
- AUTODESK Instructables. (2023, June 5). *Guide to using MAX30102 heart rate and oxygen sensor with Arduino*. Retrieved December 5, 2023, from https://www.instructables.com/Guide-to-Using-MAX30102-Heart-Rate-and-Oxygen-Sens/
- Bernadus, T. F., Subekti, L. B., & Bandung, Y. (2019). *IoT-Based Fall Detection and Heart Rate Monitoring System for Elderly Care*. https://doi.org/10.1109/iciss48059.2019.8969845
- Costa, C., Faria, J. M., Guimarães, D., Matos, D., Moreira, A. H., Morais, P., Vilaça, J. L., & Carvalho, V. (2023). A wearable monitoring device for COVID-19 biometric symptoms detection. *IRBM*, *44*(6), 100810. https://doi.org/10.1016/j.irbm.2023.100810

- DroneBot Workshop. (2021, August 1). *Arduino IoT Cloud 2021 - Getting Started with Arduino & ESP32* [Video]. YouTube. Retrieved September 9, 2023, from https://www.youtube.com/watch?v=UFCmTZUoZ1M
- Espressif Systems. (2018). *ESP32-WROOM-32 (ESP-WROOM-32) Datasheet*. Retrieved December 7, 2023, from https://www.mouser.com/datasheet/2/891/esp-wroom-32_datasheet_en-1223836.pdf
- Gotter, A. (2023, April 3). *Pulse oximetry: Purpose, uses, and how to take a reading*. Healthline. Retrieved December 6, 2023, from https://www.healthline.com/health/pulse-oximetry
- *Guidelines for SPO2 measurement*. (2023, May 22). Analog Devices. Retrieved September 10, 2024, from https://www.analog.com/en/resources/technical-articles/guidelines-for-spo2-measurement--maxim-integrated.html
- Kashif, B. (2023). *ESP32 Pinout Reference – A complete guide*. Linuxhint. Retrieved December 7, 2023, from https://linuxhint.com/esp32-pinout-reference/
- Li, C., Wang, J., Wang, S., & Zhang, Y. (2024). A review of IoT applications in healthcare. *Neurocomputing*, *565*, 127017. https://doi.org/10.1016/j.neucom.2023.127017
- Microcontrollers Lab. (2022, May 12). *MAX30102 Pulse Oximeter and Heart Rate Sensor with Arduino*. Retrieved December 5, 2023, from https://microcontrollerslab.com/max30102-pulse-oximeter-heart-rate-sensor-arduino/
- Miller, M., Byfield, R., Crosby, M., Schiltz, P., Johnson, P. J., & Lin, J. (2023). A wearable photoplethysmography sensor for non-invasive equine heart rate monitoring. *Smart Agricultural Technology*, *5*, 100264. https://doi.org/10.1016/j.atech.2023.100264

- Nandi, P., Anupama, K., Agarwal, H., Patel, K., Bang, V., Bharat, M., & Guru, M. V. (2024). Inertial measurement and heart-rate sensor-based dataset for geriatric fall detection using custom built wrist-worn device. *Data in Brief*, *52*, 109812. https://doi.org/10.1016/j.dib.2023.109812
- Naylamp Mechatronics. (n.d.). *Pulsioxímetro MAX30102*. Naylamp Mechatronics - Perú. Retrieved December 5, 2023, from https://naylampmechatronics.com/biomedico/444-pulsioximetro-max30102.html
- Ngoc-Thang, B., Nguyen, T. M. T., Truong, T. T., Nguyen, B. L., & Nguyen, T. T. (2022). A dynamic reconfigurable wearable device to acquire high quality PPG signal and robust heart rate estimate based on deep learning algorithm for smart healthcare system. *Biosensors and Bioelectronics. X*, *12*, 100223. https://doi.org/10.1016/j.biosx.2022.100223
- Panja, S., Chattopadhyay, A. K., Nag, A., & Singh, J. P. (2023). Fuzzy-logic-based IoMT framework for COVID19 patient monitoring. *Computers & Industrial Engineering*, *176*, 108941. https://doi.org/10.1016/j.cie.2022.108941
- Professional, C. C. M. (2024, May 1). *Vital signs*. Cleveland Clinic. Retrieved September 5, 2024, from https://my.clevelandclinic.org/health/articles/10881-vital-signs
- Salud, O. M. (2024, mayo 10). *OMS*. Retrieved from https://www.who.int/es/health-topics/cardiovascular-diseases#tab=tab_1
- Salud, S. d. (2024, Mayo 10). *Gobierno de México*. Retrieved from https://www.gob.mx/salud/prensa/490-cada-ano-220-mil-personas-fallecen-debido-a-enfermedades-del-corazon

- *Sensor de pulso cardíaco XD58C Ritmo XD-58C. (n.d.). Tecneu. Retrieved December 7, 2023, from https://www.tecneu.com/products/sensor-de-pulso-cardiaco-xd-58c*
- *Söderby, K. (n.d.). IoT Remote App. Arduino Cloud. Retrieved October 6, 2023, from https://docs.arduino.cc/arduino-cloud/iot-remote-app/getting-started/*
- *T-Bem, S. (2019, May 7). Arduino IoT Cloud: La Nueva plataforma. TiendaDeElectronica. Retrieved December 6, 2023, from https://tiendadeelectronica.mx/blog/arduino-iot-cloud-la-nueva-plataforma/#:~:text=Arduino%20IoT%20Cloud%20permitir%C3%A1%20a,Massimo%20Banzi%2C%20cofundador%20de%20Arduino.*
- *XD-58C Heart Pulse Sensor Module*. (2022, July 28). Einstronic Enterprise. Retrieved December 5, 2023, from https://einstronic.com/product/xd-58c-heart-pulse-sensor-module/
- *YM203 Multifunctional Bluetooth CE ROHS Monitor Fingertip Pulse Oximeter*. (n.d.). Made-in-China. Retrieved December 5, 2023, from https://honggao888.en.made-in-china.com/product/DwLmBPVvficM/China-Ym203-Multifunctional-Bluetooth-CE-RoHS-Monitor-Fingertip-Pulse-Oximeter.html