



UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO
INSTITUTO DE CIENCIAS BÁSICAS E INGENIERÍA

MAESTRÍA EN INTERNET DE LAS COSAS

PROYECTO TERMINAL

ESTACIÓN DE MONITOREO Y SEGUIMIENTO DEL ÍNDICE DE MASA CORPORAL (IMC) EN ESTUDIANTES DE BACHILLERATO CON IOT

**Para obtener el grado de
Maestro en Internet de las Cosas**

PRESENTA

Julio César Gómez Gayosso

Director (a)

Mtro. Melecio Sánchez Ruiz

Codirector (a)

Dr. Ismael Domínguez Jiménez

Comité tutorial

Dr. Juan Bacilio Guerrero Escamilla

Dr. Juan Carlos González Islas

Mtro. Melecio Sánchez Ruiz

Mineral de la Reforma, Hgo., a 20 de mayo de 2025



UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO
INSTITUTO DE CIENCIAS BÁSICAS E INGENIERÍA

MAESTRÍA EN INTERNET DE LAS COSAS

PROYECTO TERMINAL

ESTACIÓN DE MONITOREO Y SEGUIMIENTO DEL ÍNDICE DE MASA CORPORAL (IMC) EN ESTUDIANTES DE BACHILLERATO CON IOT

**Para obtener el grado de
Maestro en Internet de las Cosas**

PRESENTA

Julio César Gómez Gayosso

Director (a)

Mtro. Melecio Sánchez Ruiz

Codirector (a)

Dr. Ismael Domínguez Jiménez

Comité tutorial

Dr. Juan Bacilio Guerrero Escamilla

Dr. Juan Carlos González Islas

Mtro. Melecio Sánchez Ruiz

Mineral de la Reforma, Hgo., a 20 de mayo de 2025



Universidad Autónoma del Estado de Hidalgo

Instituto de Ciencias Básicas e Ingeniería

School of Engineering and Basic Sciences

Área Académica de Computación y Electrónica

Department of Electronics and Computer Science

Mineral de la Reforma, Hgo., a 20 de mayo de 2025

No. De Control: ICBI-AACyE/638/2025

Asunto: Autorización de impresión

MTRA. OJUKY DEL ROCÍO ISLAS MALDONADO
DIRECTORA DE ADMINISTRACIÓN ESCOLAR DE LA UA EH

El Comité Tutorial del **PROYECTO TERMINAL** del programa educativo de posgrado titulado **"ESTACIÓN DE MONITOREO Y SEGUIMIENTO DEL ÍNDICE DE MASA CORPORAL (IMC) EN ESTUDIANTES DE BACHILLERATO CON IOT"**, realizado por el sustentante **JULIO CÉSAR GÓMEZ GAYOSSO** con número de cuenta **091124** perteneciente al programa de **MAESTRÍA EN INTERNET DE LAS COSAS**, una vez que se ha revisado, analizado y evaluado el documento recepcional de acuerdo a lo estipulado en el Artículo 110 del Reglamento de Estudios de Posgrado, tiene a bien extender la presente:

AUTORIZACIÓN DE IMPRESIÓN

Por lo que la sustentante deberá cumplir los requisitos del Reglamento de Estudios de Posgrado y con lo establecido en el proceso de grado vigente.

Atentamente
"Amor, Orden y Progreso"

El Comité Tutorial

Dr. Juan Bacilio Guerrero Escamilla
Miembro del comité

Dr. Juan Carlos González Islas
Miembro del comité

Mtro. Melecio Sánchez Ruiz
Miembro del comité

Cop. Archivo
NSR



ÁREA ACADÉMICA DE
COMPUTACIÓN Y ELECTRÓNICA

Ciudad del Conocimiento, Carretera Pachuca-Tulancingo Km. 4.5 Colonia Carboneras, Mineral de la Reforma, Hidalgo, México. C.P. 42184
Teléfono: 52 (771) 71 720 00 Ext. 40052, 40053
sacye_icbi@uaeh.edu.mx, jesus_ordaz@uaeh.edu.mx



uaeh.edu.mx

Dedicatoria

A la Escuela Preparatoria Número Dos de la UAEH. Siempre es grato contribuir con el desarrollo de esta unidad Académica, un espacio único y muy representativo de mi Alma Mater. En especial a sus estudiantes, razón de ser de esta casa de estudios.

Al Colegio de Posgrado y al Instituto de Ciencias Básicas e Ingeniería, siempre impulsando que las garzas extendamos las alas y levantemos el vuelo.

A quienes con su apoyo incondicional son fuente de inspiración para el desarrollo de este proyecto.

"El viaje de mil millas comienza con un solo paso."
Lao Tse.

Agradecimientos

Al núcleo Académico de la Maestría en Internet de las Cosas por compartir sus conocimientos, experiencias y recomendaciones, fundamentales para nuestra formación profesional.

Al Director, Codirector y comité tutorial de este proyecto, sin duda el apoyo como guías en la elaboración del mismo fue clave para culminar con éxito.

A los compañeros de la primera generación de la Maestría en Internet de las Cosas por su acompañamiento y trabajo colaborativo realizado.

A mis amigos y compañeros de trabajo que siempre estuvieron para que este proyecto fuera posible.

A toda mi familia quienes siempre me han otorgado el respaldo, apoyo incondicional, aliento, ánimo y las fuerzas para hacer frente y superar los retos de la vida.

A mi amada Lore por su paciencia y comprensión.

***“La tecnología tiene el potencial de resolver
los grandes desafíos de la humanidad, como la
escasez de recursos y la enfermedad.”***

Ray Kurzweil

Resumen

La detección temprana de alteraciones nutricionales en estudiantes de bachillerato representa un reto urgente en el contexto educativo y de salud pública. Este trabajo presenta el diseño, desarrollo e implementación de un sistema basado en Internet de las Cosas (IoT) para el monitoreo automatizado del Índice de Masa Corporal (IMC), con la finalidad de identificar casos de desnutrición y sobrepeso de manera oportuna. A través del uso de sensores y plataformas digitales, se logró registrar de forma eficiente y precisa los parámetros antropométricos de una muestra de estudiantes, permitiendo conocer su estado nutricional mediante un proceso experimental. El sistema no solo optimiza la toma de decisiones en el área médica escolar mediante el monitoreo de la variable, sino que también se destaca como una herramienta sólida para la intervención preventiva y la mejora del rendimiento académico. Además, este enfoque abre nuevas posibilidades para el análisis de tendencias generacionales mediante el uso de series temporales, aportando información valiosa para el diseño de políticas educativas y de salud más efectivas. El proyecto sienta las bases para futuras aplicaciones en diagnóstico predictivo mediante algoritmos de inteligencia artificial, y su eventual evolución hacia plataformas comerciales de salud pública y medicina deportiva.

Palabras clave: Internet de las Cosas (IoT), Índice de Masa Corporal (IMC), monitoreo remoto, sistemas embebidos, microcontroladores.

Abstract

Early detection of nutritional disorders in high school students represents an urgent challenge in both educational and public health contexts. This work presents the design, development, and implementation of an Internet of Things (IoT)-based system for the automated monitoring of Body Mass Index (BMI), aimed at identifying cases of malnutrition and overweight in a timely manner. Through the use of sensors and digital platforms, anthropometric parameters of a sample of students were efficiently and accurately recorded, allowing for the assessment of their nutritional status through an experimental process. The system not only optimizes decision-making in school medical settings by monitoring this variable, but also stands out as a robust tool for preventive intervention and the improvement of academic performance. Furthermore, this approach opens new possibilities for analyzing generational trends through time series data, providing valuable information for the design of more effective educational and health policies. The project lays the groundwork for future applications in predictive diagnostics using artificial intelligence algorithms and its eventual evolution into commercial platforms for public health and sports medicine.

Keywords: Internet of Things (IoT), Body Mass Index (BMI), remote monitoring, embedded systems, microcontrollers.

Contenido

<i>Dedicatoria</i>	<i>V</i>
<i>Agradecimientos</i>	<i>VI</i>
<i>Resumen</i>	<i>VII</i>
<i>Abstract</i>	<i>VIII</i>
<i>Índice de figuras</i>	<i>XIII</i>
<i>Índice de tablas</i>	<i>XV</i>
<i>Capítulo 1 Introducción</i>	<i>1</i>
1.1. Descripción del problema	<i>2</i>
1.2. Antecedentes	<i>2</i>
1.3. Marco contextual	<i>3</i>
1.4. Motivación	<i>4</i>
1.5. La afectación en el rendimiento y aprovechamiento escolar provocado por los padecimientos a causa de la desnutrición y obesidad	<i>4</i>
1.6. La prevención de afectaciones a la salud en estudiantes de bachillerato	<i>4</i>
1.7. La obesidad en estudiantes, como factor en su rendimiento	<i>5</i>
1.8. Desestimación del IMC como indicador de salud en los adolescentes	<i>5</i>
1.9. Preguntas de Investigación	<i>6</i>
1.10. Hipótesis	<i>6</i>
1.11. Propuesta de solución	<i>6</i>
1.12. Justificación	<i>7</i>
1.13. Objetivo General	<i>8</i>
1.14. Objetivos específicos	<i>8</i>
<i>Capítulo 2 Estado del arte</i>	<i>9</i>
2.1. Aportaciones relevantes de IoT al cuidado de la salud	<i>9</i>
2.2. Desarrollos tecnológicos actuales	<i>11</i>
2.3. Desafíos en la Implementación de IoT en Salud	<i>13</i>
<i>Capítulo 3 Metodología</i>	<i>15</i>
3.1. Metodología utilizada.....	<i>15</i>
<i>Capítulo 4 Desarrollo de la plataforma experimental</i>	<i>17</i>
4.1. Análisis de contexto	<i>17</i>

4.1.1.	Necesidades	17
4.1.2.	Restricciones	17
4.1.3.	Usuarios	17
4.1.4.	Estudio de factibilidad	18
4.1.4.1.	Participantes en el desarrollo del estudio de factibilidad	18
4.1.4.2.	Instrumento de evaluación para estudio de factibilidad	18
4.1.4.3.	Interpretación de resultados	20
4.2.	Análisis de requerimientos	23
4.2.1.	Requerimientos funcionales	23
4.2.2.	Requerimientos no funcionales	25
4.2.3.	Factibilidad técnica	26
4.2.4.	Cálculo de costos aproximados de componentes sugeridos	26
4.2.5.	Seguridad	27
4.2.5.1.	Control de acceso roto	27
4.2.5.2.	Fallas criptográficas	28
4.2.5.3.	Errores de configuración en la seguridad	28
4.2.5.4.	Deficiencias en el registro de seguridad y monitoreo	29
4.3.	Análisis de tecnología	29
4.3.1.	Microcontroladores como eje funcional del Internet de las cosas (IoT)	30
4.3.2.	Comparativa de microcontroladores	31
4.3.3.	El microcontrolador ESP8266	32
4.3.4.	Sensores para medición de distancia en IoT	33
4.3.4.1.	Sensor de distancia VL53L0X	34
4.3.5.	Sensores para medición de peso en IoT	34
4.3.5.1.	Sensor de peso HX711	35
4.3.6.	Protocolos de comunicación IoT	36
4.3.6.1.	MQTT (Message Queuing Telemetry Transport)	36
4.3.6.2.	CoAP (Constrained Application Protocol)	37
4.3.6.3.	AMQP (Advanced Message Queuing Protocol)	37
4.3.6.4.	HTTP (Hypertext Transfer Protocol)	38
4.3.6.5.	DDS (Data Distribution Service)	38
4.3.6.6.	JMS (Java Message Service)	38
4.3.6.7.	XMPP (Extensible Messaging and Presence Protocol)	39
4.3.6.8.	Protocolo MQTT y microcontrolador ESP8266 beneficios y ventajas	39
4.3.6.9.	Comparativa y selección del broker	40
4.3.6.10.	AWS IoT Core	41
4.3.7.	Aplicación de bases de datos no relacionales en IoT	41
4.3.7.1.	Características de las bases de datos no relacionales	42
4.3.7.2.	Aplicación de NoSQL en el contexto de IoT	42
4.3.7.3.	Desafíos y consideraciones clave	43
4.3.8.	Bases de datos de AWS	43
4.3.8.1.	Amazon DynamoDB	44
4.3.8.2.	Amazon Timestream	44
4.3.9.	Visualización de datos en la nube con Amazon Managed Grafana	44
4.3.10.	Tecnologías Inalámbricas para IoT	45
4.3.10.1.	Wi-Fi (802.11) y el ecosistema Internet de las Cosas (IoT)	46
4.3.10.2.	ZigBee	46
4.3.10.3.	LoRaWAN (Long Range Wide Area Network)	46

4.3.11.	El IDE de Arduino y su integración en plataformas IoT	47
4.4.	Diseño del sistema	48
4.4.1.	Arquitectura utilizada para el desarrollo del prototipo	48
4.4.2.	Diagrama del prototipo (Arquitectura de capas IoT)	48
4.4.3.	Elementos utilizados para la transferencia de datos a la Nube	49
4.4.4.	Diagrama de sistema embebido	50
4.4.5.	Diagrama a bloques del prototipo	53
4.4.6.	Diagrama de proceso	53
4.4.7.	Descripción general del proceso	55
4.5.	Construcción del sistema	56
4.5.1.	Integración del Hardware	56
4.5.2.	Gabinets de protección del sistema	56
4.5.3.	Programación del microcontrolador	58
4.5.3.1.	Librerías incluidas	58
4.5.3.2.	Calibración del HX711	58
4.5.3.3.	Configuración de servicios de red y protocolo de comunicación	59
4.5.3.4.	Función de conexión de red WI-FI	59
4.5.3.5.	Reconexión del broker MQTT en caso de pérdida de comunicación	60
4.5.3.6.	Carga de certificados. DER de AWS	60
4.5.3.7.	Función de interrupción para activar los sensores	62
4.5.3.8.	Función para inicializar el sensor VL53L0X y su operación	63
4.5.3.9.	Funciones para inicializar el sensor HX711 y su operación	63
4.5.3.10.	Publicación del mensaje a tópico MQTT en formato JSON	64
4.5.3.11.	Función Setup	65
4.5.3.12.	Función loop	66
4.5.3.13.	Monitor serie	68
4.5.4.	Amazon Web Services	69
4.5.4.1.	Consola de AWS	69
4.5.4.2.	Conexión de objetos (Things) en AWS IoT Core	70
4.5.4.3.	Pruebas de conexión MQTT en AWS IoT Core	73
4.5.4.4.	Ingesta de datos con AWS Timestream	75
4.5.4.5.	Consultas en Timestream	80
4.5.4.6.	Almacenamiento permanente con AWS DynamoDB	82
4.5.4.7.	Consultas en DynamoDB	85
4.5.4.8.	Tablero (Dashboard) en Amazon Managed Grafana	87
Capítulo 5	Análisis y discusión de resultados.....	93
5.1.	Metodología y muestreo	93
5.2.	Análisis de los resultados	97
5.3.	Apreciación crítica de los resultados	99
5.4.	Impacto del Sistema IoT en la Gestión de la Salud Escolar	100
5.5.	Limitaciones del estudio	100
5.6.	Recomendaciones.....	101
Capítulo 6	Conclusiones y trabajo futuro	103

6.1. Conclusiones	103
6.2. Trabajo futuro	104
<i>Referencias</i>	<i>106</i>
<i>Glosario</i>	<i>118</i>
<i>Apéndice A</i>	<i>121</i>
<i>Apéndice B</i>	<i>125</i>
<i>Apéndice C</i>	<i>127</i>

Índice de figuras

Figura 1. Modelo propuesto para el desarrollo de proyectos IoT.).	16
Figura 2. Distribución porcentual sobre si existe afectación al rendimiento escolar debido al estado de salud.	20
Figura 3. Distribución porcentual sobre la consideración a la prevención.	20
Figura 4. Distribución porcentual sobre la necesidad de evitar la obesidad y desnutrición.	21
Figura 5. Distribución porcentual sobre el control de problemas de nutrición con ayuda de la tecnología.	21
Figura 6. Distribución porcentual sobre la disminución en la deserción escolar debido al monitoreo de la salud.	21
Figura 7. Distribución porcentual sobre la mejora en la salud mediante el desarrollo de una estación de monitoreo.	22
Figura 8. Distribución porcentual sobre la necesidad de toma de decisiones inmediata en salud.	22
Figura 9. Distribución porcentual sobre el favorecimiento de la salud mediante monitoreo.	22
Figura 10. Distribución porcentual sobre el alertamiento para la prevención en la salud.	23
Figura 11. Distribución porcentual sobre la aceptación de herramientas tecnológicas en apoyo al servicio médico.	23
Figura 12. Diagrama de arquitectura en capas del prototipo.	49
Figura 13. Estructura de comunicación.	50
Figura 14. Diagrama de conexión de sistema embebido.	50
Figura 15. Diagrama de conexión celdas de carga.	51
Figura 16. Diagrama a bloques de la solución.	53
Figura 17. Diagrama del proceso de la solución.	54
Figura 18. Modelo de gabinete para impresión 3D del ESP8266.	57
Figura 19. Modelo de gabinete para impresión 3D del sensor VL53L0X.	57
Figura 20. Modelo de gabinete para impresión 3D del sensor HX711.	57
Figura 21. Pseudocódigo de calibración HX711.	58
Figura 22. Pseudocódigo Configuración de servicios de red.	59
Figura 23. Pseudocódigo de conexión de red WI-FI.	60
Figura 24. Pseudocódigo de reconexión del broker.	60
Figura 25. Carpeta "data" adjunta al código fuente.	61
Figura 26. Archivos de certificados AWS en formato .der.	61
Figura 27. Lista de herramientas con la librería FS.h.	62
Figura 28. Pseudocódigo activación de sensores.	63
Figura 29. Pseudocódigo que inicializa el sensor VL53L0X.	63
Figura 30. Pseudocódigo que inicializa el sensor HX711.	64
Figura 31. Pseudocódigo que calcula la masa.	64
Figura 32. Pseudocódigo publicación de mensaje MQTT.	65
Figura 33. Pseudocódigo función setup.	66
Figura 34. Pseudocódigo, Memoria, ID y MAC.	66
Figura 35. Pseudocódigo función Loop.	67
Figura 36. Pseudocódigo publicación de mensaje.	68
Figura 37. Monitor serie IDE Arduino.	68

Figura 38. Medición y publicación de datos vista del monitor serie IDE Arduino.	69
Figura 39. Consola AWS.	70
Figura 40. Ventana de inicio de AWS IoT Core.	71
Figura 41. Ventana de objetos creados en AWS IoT Core.	72
Figura 42. Ventana de certificados de seguridad (certificados .DER) creados en AWS IoT Core.	73
Figura 43. Ventana de políticas creadas en AWS IoT Core.	73
Figura 44. Prueba MQTT en AWS IoT Core.	74
Figura 45. Vista de suscripción MQTT en AWS IoT Core.	74
Figura 46. Ventana de inicio de AWS de Timestream.	76
Figura 47. Base de datos imc_IoT de la ingesta, creada en Timestream.	77
Figura 48. Tabla imc_Data_IoT de la ingesta, creada en Timestream.	78
Figura 49. Creación de regla de interacción de la ingesta, en Timestream.	80
Figura 50. Tabla consulta (Query) SQL, en Timestream.	81
Figura 51. Ventana de inicio de AWS DynamoDB.	83
Figura 52. Tabla imc_IoT_Data, creada en AWS DynamoDB.	83
Figura 53. Creación de regla de interacción de la ingesta, en AWS DynamoDB.	85
Figura 54. Selección de Tabla imc_IoT_Data, en AWS DynamoDB.	86
Figura 55. Consulta de datos en Tabla imc_IoT_Data, AWS DynamoDB.	86
Figura 56. Ventana de inicio de Amazon Managed Grafana.	87
Figura 57. Espacio de trabajo de Amazon Managed Grafana.	88
Figura 58. Selección de región y fuente de datos Amazon Managed Grafana.	88
Figura 59. Consulta de datos T-SQL en Amazon Managed Grafana.	90
Figura 60. Vista de Dashboard en el navegador de PC. Tomada de AWS (2025).	92
Figura 61. Diagrama de muestreo, almacenamiento y procesamiento de datos muestreados.	94
Figura 62. Aplicación de la configuración con 2 estudiantes de bachillerato.	95
Figura 63. IMC registrado de 80 estudiantes.	98
Figura 64. Distribución de los datos de IMC de 80 estudiantes.	98
Figura 65. Distribución del IMC en estudiantes de Bachillerato.	99
Figura 66. Publicación Padi Vol.10 No. Especial 3.	127

Índice de tablas

Tabla 1. Niveles, etapas y descripción Modelo propuesto para el desarrollo de proyectos IoT...	16
Tabla 2. Participantes, partes interesadas.....	18
Tabla 3. Costos estimados de los componentes para la realización del prototipo.	27
Tabla 4. Comparativa de microcontroladores.....	31
Tabla 5. Comparativa de sensores de distancia.....	33
Tabla 6. Comparativa de sensores de peso.	35
Tabla 7. Comparativa de brokers de uso gratuito y de pago.	40
Tabla 8. Características de DynamoDB y Timestream.....	43
Tabla 9. Características de Amazon Managed Grafana.....	45
Tabla 10. Comparativo de redes inalámbricas para IoT.....	45
Tabla 11. Conexión de cableado rojo C/Red.	52
Tabla 12. Conexión de cableado blanco +/-Wht.	52
Tabla 13. Conexión de cableado negro -/Blk.....	52
Tabla 14. Conexión ESP8266 - HX711.....	52
Tabla 15. Conexión ESP8266 - VL53L0X.....	53
Tabla 16. Características de los gabinetes de los componentes IoT.....	57
Tabla 17. Carcasas de componentes.	57
Tabla 18. Datos de la muestra e IMC calculado.....	96
Tabla 19. Análisis de datos.	97
Tabla 20. Riesgos de enfermedad por IMC.....	97

Capítulo 1 Introducción

En la actualidad, las instituciones educativas enfrentan desafíos significativos relacionados con la salud de los estudiantes, particularmente en lo que respecta a trastornos alimenticios como la desnutrición y la obesidad. Estas condiciones no solo afectan el bienestar físico de los adolescentes, sino que también inciden directamente en su rendimiento académico y desarrollo integral (León De la Cruz et al., 2023).

La incorporación del Internet de las Cosas (IoT) en el ámbito de la salud ha transformado la manera en que se monitorean y gestionan diversos indicadores de bienestar, incluyendo el Índice de Masa Corporal (IMC). Esta tecnología permite una supervisión continua y precisa de parámetros corporales, facilitando intervenciones tempranas y personalizadas. Además, la conectividad mejora la precisión en el seguimiento del IMC y empodera a los individuos en la gestión de su salud, promoviendo hábitos más saludables y una mayor adherencia a tratamientos. (Mae, J., Oey, E., & Kristiady, F. S., 2020).

En el contexto educativo, la implementación de tecnologías IoT para el monitoreo de la salud estudiantil no solo mejora la precisión en la recopilación de datos, sino que también permite una intervención oportuna por parte del personal médico escolar. Esto contribuye a la prevención de enfermedades y al fomento de hábitos saludables desde una edad temprana.

Este estudio se centra en explorar cómo las herramientas tecnológicas, específicamente las soluciones basadas en IoT, pueden contribuir al monitoreo y mejora del IMC en estudiantes de nivel medio superior. Al integrar dispositivos de medición y plataformas de análisis de datos, se busca proporcionar una visión más clara y precisa del estado de salud de los alumnos, facilitando intervenciones oportunas y personalizadas.

Además de abordar la implementación técnica de un prototipo IoT, esta investigación considera las implicaciones éticas, sociales y educativas de su aplicación. Al comprender la intersección entre tecnología y salud en el ámbito escolar, se aspira a ofrecer soluciones prácticas que promuevan hábitos saludables y mejoren el desempeño académico de los estudiantes.

1.1. Descripción del problema

En numerosos contextos, la carencia de herramientas tecnológicas automatizadas y asequibles limita la posibilidad de que las personas supervisen regularmente su Índice de Masa Corporal (IMC), especialmente en espacios con escasos recursos médicos o con escaso nivel de educación en salud preventiva. Asimismo, los métodos convencionales de recolección de datos, ya sean manuales o semiautomáticos, dificultan la integración de información en tiempo real y la generación de alertas o recomendaciones personalizadas basadas en patrones previos.

El desarrollo de tecnologías basadas en el Internet de las Cosas (IoT) abre la puerta a soluciones inteligentes que permiten recopilar, transmitir y analizar datos fisiológicos al instante. Sin embargo, aún persiste una carencia en la implementación de estaciones automatizadas, y eficientes que midan parámetros como el peso y la estatura, calculen el IMC y proporcionen un monitoreo constante, remoto y accesible. La integración de esta información a plataformas en la nube facilitaría su uso tanto por los usuarios como por los profesionales de la salud.

En este sentido, se plantea la necesidad de diseñar e implementar una estación de monitoreo del IMC apoyada en tecnología IoT, que ofrezca a los usuarios la posibilidad de registrar y seguir la evolución de su estado corporal de manera automática, y en tiempo real. Esta solución debe ser económica, fácil de operar y accesible, para potenciar su utilidad en la promoción de estilos de vida saludables y en la prevención de enfermedades vinculadas al sobrepeso o desnutrición.

1.2. Antecedentes

Durante el proceso de formación en el estudiante de bachillerato, es importante el seguimiento del histórico, que permita identificar las posibles afectaciones en su estado de salud. Uno de los primeros registros del estado de salud se realiza con el examen médico, requisito indispensable para que el estudiante pueda inscribirse en el primer ciclo escolar. Con anterioridad, el registro de peso y estatura se realizaba en documentación impresa que finalmente quedaba archivada masivamente.

En la actualidad se cuenta con una plataforma en donde los datos adquiridos en el examen médico, así como los antecedentes sobre padecimientos en el estudiante son registrados de forma manual por el servicio médico universitario, generando un expediente digital. No existe un registro histórico generacional en donde se observe el comportamiento de las series en el tiempo de la medición de estatura y peso que puedan causar posibles afectaciones en los estudiantes a lo largo de su trayectoria escolar.

1.3. Marco contextual

En el contexto escolar, particularmente en la Escuela Preparatoria Número Dos, dependiente de la Universidad Autónoma del Estado de Hidalgo, el análisis de los síntomas psicosomáticos adquiere una relevancia particular, ya que el entorno educativo no solo cumple una función formativa, sino que también actúa como un espacio donde se manifiestan diversas expresiones del bienestar o malestar físico y emocional del estudiante. Es frecuente que algunos alumnos presenten síntomas psicosomáticos como cefaleas, trastornos gastrointestinales, fatiga o alteraciones del sueño cuya etiología no se encuentra en una causa orgánica identificable, sino en factores psicológicos, emocionales o incluso nutricionales.

De manera específica, problemas como la desnutrición y la obesidad en la adolescencia, que constituyen dos manifestaciones de la malnutrición, pueden estar estrechamente vinculados con la aparición de estos síntomas. La desnutrición puede generar debilidad generalizada, dificultades cognitivas y bajo rendimiento escolar, lo que puede traducirse en una experiencia académica frustrante y estresante para el estudiante. Por otro lado, la obesidad no solo acarrea riesgos físicos, sino también psicológicos, como la baja autoestima, el aislamiento social o el acoso escolar, factores que pueden desencadenar respuestas psicosomáticas en el alumnado.

La Organización Mundial de la Salud ha señalado que la coexistencia de desnutrición y obesidad en la adolescencia representa una “doble carga” de problemas de salud relacionados con la nutrición, lo que requiere medidas integradas para prevenir y controlar ambas condiciones. Asimismo, la OMS (2022) destaca la importancia de promover una alimentación saludable desde etapas tempranas de la vida como estrategia clave para garantizar el crecimiento y desarrollo adecuados, y prevenir trastornos relacionados con el estado nutricional.

En el ámbito nacional, estudios realizados por la Universidad Autónoma del Estado de Hidalgo (UAEH) han evidenciado la presencia simultánea de desnutrición y obesidad en la población escolar mexicana. Según Cortés-Castillo et al. (2022), esta situación refleja la necesidad de rediseñar políticas públicas que refuercen estrategias integrales de alimentación en México. Además, investigaciones desarrolladas por Hernández-Díaz et al. (2014) identificaron una prevalencia significativa de sobrepeso y riesgo de obesidad en escolares, lo que subraya la urgencia de implementar intervenciones efectivas en el entorno educativo.

En consecuencia, comprender la relación entre el estado nutricional de los estudiantes, su salud emocional y su desempeño académico es fundamental para diseñar intervenciones integrales en el ámbito escolar. Estas deben considerar no solo el rendimiento académico, sino también la

salud física y mental del estudiante, promoviendo ambientes escolares saludables, programas de alimentación adecuados y apoyo psicosocial continuo.

1.4. Motivación

En el nivel de educación media superior, la atención a la salud del estudiantado representa una problemática creciente. Muchos planteles carecen de personal médico suficiente para atender de manera oportuna y adecuada los distintos padecimientos que afectan a los jóvenes de entre 15 y 20 años, etapa crítica en su desarrollo físico, emocional y académico. Esta limitación en la infraestructura de salud escolar compromete el bienestar integral del alumnado y puede traducirse en consecuencias como ausentismo, bajo rendimiento y deserción escolar.

Frente a este escenario, resulta imperativo establecer un método que permita optimizar los recursos médicos disponibles en las instituciones de bachillerato. La implementación de una estrategia que aligere la carga de trabajo del personal de salud no solo facilitaría la atención inmediata, sino que también permitiría la generación y análisis sistemático de datos sobre el estado de salud de los estudiantes. Esta información sería clave para diseñar acciones preventivas y correctivas, orientadas a preservar la salud del estudiantado y reducir la inasistencia escolar derivada de problemas médicos.

1.5. La afectación en el rendimiento y aprovechamiento escolar provocado por los padecimientos a causa de la desnutrición y obesidad

Con base en la investigación desarrollada por (Jukes & McGuire, 2002), es necesaria la inversión en la nutrición, la cuál es fundamental para el desarrollo del intelecto y el aprendizaje, así como para el rendimiento cognitivo, de igual refieren que el 53 % de niños en edad escolar, presentan deficiencia de hierro y presentan anemia. Asistir al colegio sin desayunar provoca una disminución del rendimiento de manera considerable.

1.6. La prevención de afectaciones a la salud en estudiantes de bachillerato

El plan educativo nacional, refiere que la salud del estudiante es de suma importancia para el desempeño físico y mental de estudiantes, sobre todo en edad escolar de la adolescencia donde los adolescentes sufren cambios constantes. Las afectaciones están determinadas por los estilos de vida, la alimentación correcta, la actividad física, la higiene personal y los factores psicosociales. El programa escuela y salud, señala como objetivo principal, realizar intervenciones anticipatorias y monitoreo continuo que ejerzan mayor control en la causa de los problemas de salud.

1.7. La obesidad en estudiantes, como factor en su rendimiento

El gobierno de México proporciona datos basados sobre la encuesta nacional de salud y nutrición, y refiere que uno de cada cinco niños tiene sobrepeso u obesidad, 30 millones de adultos presentan obesidad o sobrepeso, así como 18.5 millones con sobrepeso y 11.4 millones con obesidad, ocupando nuestro país el primer lugar en obesidad infantil y el segundo lugar en adultos. Señala también la necesidad de tratamiento integral proporcionado por especialistas como psicólogos, nutriólogos, médicos y expertos en actividad física.

1.8. Desestimación del IMC como indicador de salud en los adolescentes

A pesar de ser uno de los indicadores más accesibles y útiles para evaluar el estado nutricional, el Índice de Masa Corporal (IMC) continúa siendo subestimado en muchos entornos escolares de México, especialmente a nivel medio superior. Esta falta de atención sistemática al IMC ha contribuido a que se normalicen tanto el sobrepeso como la desnutrición en la adolescencia, dejando pasar señales tempranas de alerta que podrían evitar complicaciones mayores.

El Índice de Masa Corporal (IMC) se ha consolidado como una herramienta clave para evaluar el estado nutricional de la población, particularmente en contextos escolares donde la salud física incide directamente en el rendimiento académico y el bienestar emocional. En el caso de los estudiantes adolescentes, etapa marcada por importantes cambios físicos y psicológicos, el monitoreo constante del IMC permite detectar oportunamente señales de riesgo asociadas tanto a la obesidad como a posibles desórdenes alimenticios. Sin embargo, el seguimiento del IMC rara vez forma parte de un protocolo regular en las escuelas, y muchas veces se percibe como un dato aislado, sin conexión con la prevención de enfermedades crónicas o trastornos alimenticios (Jiménez-Cruz et al., 2019).

La obesidad, entendida como una acumulación excesiva de grasa corporal, suele estar estrechamente relacionada con hábitos alimenticios inadecuados, sedentarismo y factores psicosociales que se agudizan en la adolescencia (World Health Organization [WHO], 2021). Un IMC elevado en esta etapa no solo representa un riesgo para enfermedades crónicas a futuro — como diabetes tipo 2 o hipertensión—, sino también una afectación significativa en la autoestima, la socialización y la asistencia escolar (Flores-Ramos et al., 2020).

Por otro lado, la vigilancia del IMC también contribuye a la detección temprana de patrones que puedan vincularse con desórdenes alimenticios como la anorexia o la bulimia. En muchos casos, una disminución abrupta del IMC puede estar reflejando una relación conflictiva del estudiante

con su cuerpo o su alimentación, lo cual requiere una intervención multidisciplinaria oportuna (González-Macías et al., 2022).

El IMC no debe ser considerado únicamente como un número, sino como un indicador útil en la promoción de estilos de vida saludables, la prevención de enfermedades y la mejora del entorno educativo, su desestimación representa una barrera para la intervención temprana, sobre todo si se considera que México enfrenta altos índices de obesidad infantil y juvenil a nivel mundial. Reconocer el valor del IMC como herramienta preventiva, y no solo diagnóstica, es fundamental para diseñar políticas escolares que fomenten estilos de vida saludables desde edades tempranas.

1.9. Preguntas de Investigación

- a) ¿Qué impacto tiene la implementación de un sistema automatizado de monitoreo de IMC en la detección oportuna de alteraciones nutricionales en estudiantes de bachillerato?
- b) ¿De qué manera un registro constante y digitalizado de parámetros como peso y estatura puede influir en la mejora de la salud estudiantil y en la disminución de inasistencias escolares que afectan el rendimiento escolar?
- c) ¿Cómo contribuye el acceso inmediato a reportes médicos automatizados en la toma de decisiones dentro del área de servicio médico escolar, y qué efectos podría tener en el rendimiento académico de los estudiantes?

1.10. Hipótesis

La implementación de un sistema automatizado de monitoreo de variables antropométricas como peso y estatura, mediante sensores integrados en el servicio médico escolar, contribuiría significativamente a mejorar la detección temprana de problemas de salud relacionados con la desnutrición y el sobrepeso en estudiantes de bachillerato.

Lo anterior parte de la premisa de que las afectaciones en la salud tienen un impacto directo en el desempeño escolar, y que la falta de un monitoreo constante y sistemático dificulta la toma de decisiones clínicas informadas. Por tanto, se espera que la incorporación de tecnologías de medición automática facilite la recolección y análisis de datos clínicos, optimizando los procesos de atención médica preventiva dentro del entorno escolar.

1.11. Propuesta de solución

En este trabajo se propone una solución orientada a mejorar los procesos de recopilación y análisis de información clínica en instituciones de educación media superior, con la finalidad del monitoreo constante en los servicios médicos escolares. En este contexto, la integración de

tecnologías emergentes, como el Internet de las Cosas (IoT), ofrece nuevas oportunidades para mejorar los servicios de salud en instituciones educativas. La propuesta consiste en el desarrollo e implementación de una estación automatizada de monitoreo de salud que utilice tecnologías del Internet de las Cosas (IoT) para registrar y gestionar datos biométricos fundamentales, como el peso y la estatura del estudiantado.

El sistema planteado permitirá la detección temprana de indicadores asociados a enfermedades no transmisibles, como el sobrepeso y la obesidad, mediante la integración de sensores adecuados y módulos de comunicación que posibiliten la transmisión en tiempo real de los datos recolectados. Esta información será almacenada y analizada utilizando herramientas estadísticas, con el propósito de generar reportes clínicos automatizados y apoyar la elaboración de expedientes médicos individualizados. De esta manera, se busca dotar al personal de salud escolar con una herramienta tecnológica que facilite el monitoreo sistemático del estado nutricional de los estudiantes y, con ello, contribuir a la implementación de estrategias preventivas y de atención oportuna dentro de la comunidad educativa.

En suma, se plantea la creación de una infraestructura mínima pero funcional que, mediante el uso de dispositivos IoT, cumpla con los requerimientos técnicos y operativos necesarios para su integración en el entorno escolar. Esta solución tecnológica no solo permitirá un control más preciso de la salud estudiantil, sino que también fomentará una cultura institucional orientada a la prevención y al bienestar integral. Además, se alinea con las tendencias actuales en salud digital, que promueven el uso de tecnologías innovadoras para mejorar la calidad y eficiencia de los servicios de salud (Bhushan et al., 2021).

1.12. Justificación

La falta de monitoreo sistemático que impide la detección oportuna de afecciones comunes en adolescentes estudiantes de bachillerato, como la desnutrición y el sobrepeso, condiciones que afectan significativamente su bienestar físico, emocional y académico, atendiendo la necesidad en el reforzamiento de estrategias preventivas desde el entorno escolar, a partir de una alternativa innovadora, escalable y pertinente, que atiende una necesidad crítica en el ámbito educativo institucional orientada al bienestar y al uso responsable de tecnologías emergentes.

El monitoreo de salud escolar, atiende una necesidad crítica en el ámbito educativo. Este proyecto no solo busca mejorar la calidad de vida de los estudiantes mediante acciones preventivas, sino también fomentar una cultura institucional orientada al bienestar y al uso responsable de tecnologías emergentes.

Este proyecto se encuentra alineado con el Programa Escuela y Salud, el cual establece como objetivo principal el fortalecimiento del monitoreo y la prevención de los problemas de salud más frecuentes en estudiantes de educación media superior (Secretaría de Educación Pública [SEP], 2020). Asimismo, responde a los lineamientos del Plan Nacional de Desarrollo y de la Nueva Escuela Mexicana, que reconocen la salud física y emocional como un eje central en el proceso educativo.

1.13. Objetivo General

Desarrollar una estación de monitoreo y seguimiento del Índice de Masa Corporal (IMC) basada en tecnologías de Internet de las Cosas (IoT), para la captura automática de datos antropométricos, su almacenamiento seguro en la nube y la visualización en tiempo real, con el propósito del control continuo del estado nutricional y la toma de decisiones en el ámbito de la salud preventiva.

1.14. Objetivos específicos

- I. Diseñar una estación de monitoreo físico equipada con sensores IoT capaz de medir peso y estatura con precisión y confiabilidad.
- II. Implementar una estación para el monitoreo, que mediante una infraestructura mínima de IoT, cumpla con los requerimientos necesarios para la comunidad en instituciones de educación media superior.
- III. Calcular el Índice de Masa Corporal (IMC) y clasificar los resultados con base en parámetros establecidos.
- IV. Desarrollar una interfaz de consulta para el monitoreo del estado nutricional de la población estudiantil.
- V. Implementar un sistema de resguardo de información biométrica recolectada a través de datos obtenidos por sensores IoT, mediante almacenamiento en la nube, para tener un seguimiento histórico y consulta posterior.

Capítulo 2 Estado del arte

2.1. Aportaciones relevantes de IoT al cuidado de la salud

Actualmente se desarrollan múltiples trabajos relacionados con IoT en apoyo a la salud, el cuidado de la misma, así como diagnóstico. El enfoque en la industria del cuidado de la salud, así como la evaluación de las condiciones de esta y el monitoreo, son de manera particular temas en donde se centran los trabajos actuales, así como la obtención de datos y su almacenamiento para su posterior uso en diagnósticos.

En temas de obesidad, IoT hoy en día es muy recurrente pues la ocupación por el monitoreo masivo de las personas es motivo de elaboración de trabajos y desarrollo de dispositivos para el monitoreo en tiempo real, así como el registro a la par de esta.

El proyecto “Continua Health Alliance”, desarrollado en Estados Unidos, que integró sensores de monitoreo domiciliario (presión arterial, glucosa, frecuencia cardíaca) conectados mediante protocolos estandarizados a plataformas de datos interoperables. Esta iniciativa adoptó una metodología basada en desarrollo modular, que permitió escalar el sistema desde aplicaciones en hogares individuales hasta clínicas comunitarias. La clave estuvo en combinar estándares abiertos con una arquitectura centrada en la privacidad del paciente (Steele et al., 2012).

Trabajos desarrollados por (Gupta, Akansksha & Swarnalatha, 2020), emplean un microcontrolador Arduino para el monitoreo del oxígeno (MAX30100 sensor), presión de la sangre y temperatura (sensor LM35), utilizando un dashboard para la entrega de la información, sin la integración de la medición de estatura y peso.

La integración de un sensor de temperatura DSB18B20, los sensores de ritmo cardíaco KY-039 y AD8232 fueron implementados en el desarrollo prototipo de quiosco para monitoreo de salud, proyecto realizado por (Kumar, 2018).

En Asia, la Universidad Nacional de Singapur implementó un sistema de monitoreo continuo de signos vitales para estudiantes universitarios, con énfasis en la prevención del estrés térmico y la fatiga. Utilizando una metodología ágil, el equipo diseñó una red de sensores vestibles conectados a una app móvil, que alertaba automáticamente al personal médico si se detectaban parámetros anómalos. El enfoque iterativo permitió mejorar la precisión y la usabilidad del sistema a partir de pruebas piloto dentro del campus universitario (Tan et al., 2019).

Tecnología RFID es utilizada en el desarrollo de estaciones prototipos para el apoyo a espacios rurales en la india para el monitoreo de síntomas que tienen que ver con VIH, malaria, tuberculosis, diarrea, así como también la desnutrición en niños menores de 5 años, para ello utilizan sensores de temperatura, presión sanguínea, ritmo cardiaco, estatura y peso. Proyecto desarrollado por (Jacob & Menon, 2019).

La propuesta de implementación de cómputo en la niebla para la administración óptima de los datos en el cuidado de la salud, solucionando la problemática en variaciones de los datos provocados por la latencia y los grandes volúmenes de datos, así como errores de transmisión mediante cómputo en la nube, son referidos por (De Moura et al, 2020).

El proyecto “SmartHealth” desarrollado en Europa, liderado por el University College London, se enfocó en el desarrollo de dispositivos portátiles para la detección temprana de caídas en adultos mayores. La metodología empleada fue Design Thinking, centrando el diseño en las necesidades emocionales y físicas de los usuarios. A través de sesiones de co-creación con cuidadores y pacientes, se desarrollaron prototipos que incorporaban sensores de movimiento, geolocalización y una interfaz de alerta simple y amigable. Esta estrategia facilitó una mayor aceptación del dispositivo y una menor tasa de abandono tecnológico (Cook et al., 2020).

Un modelo de quiosco presentado por (Bagula et al, 2019), sugiere atención médica cibernética basada en IoT, implementando sensores de medición de saturación de oxígeno, sensor de flujo nasal de aire, sensor de posición del paciente (acelerómetro), glucómetro, electrocardiograma, presión en la sangre, electromiograma y actividad electrodérmica (características eléctricas en la piel), con tecnología IoT zigbee, LoRa, y machine learning, basado en un estudio previo de las necesidades para diagnóstico clínico, sin embargo el modelo no se ha desarrollado ni probado, pues está en función de la disponibilidad de los sensores y la aplicación de las técnicas de aprendizaje automático. Únicamente se utilizaron sensores para, Arduino y se tomaron muestras con instrumentos de medición convencionales donde se obtuvieron variaciones considerables.

Una investigación propone un sistema novedoso de monitoreo de la salud para los estudiantes, basado en inteligencia ambiental e internet de las cosas que permite la mejora en su rendimiento, el cual sugiere la implementación de un modelo basado en 3 capas, capa de IoT, Capa de nube y capa de monitoreo del estudiante, todo ello con apoyo de máquinas inteligentes, redes de sensores aplicaciones para teléfonos inteligentes para la colecta de datos, la capa de nube apoyada en tecnologías 4G, 5G, LTE y la capa de monitoreo, la cual otorga información y

advertencias sobre el estado de salud a los estudiantes, así como a los padres y a las partes interesadas. (Hong-tan et al, 2021).

En un modelo de machine learning basado también en 3 capas, propone un sistema para verificar signos vitales y detectar cambios biológicos y en el comportamiento de los estudiantes con IoT, se realiza predicción de datos con algoritmos de clasificación SVM (Support Vector Machines), DT (Decision Trees), RF (Random Forest), y MLP (Multiple Layer Perceptron) (Souri et al, 2020).

Las tecnologías IoT han posibilitado el desarrollo de dispositivos médicos conectados que transforman la forma en que se brinda atención a los pacientes. Dispositivos como glucómetros, monitores de ritmo cardíaco y presión arterial, y otros wearables, transmiten datos biométricos en tiempo real a los profesionales de la salud, lo que permite una intervención temprana en caso de anomalías (Sun et al., 2024).

Un ejemplo innovador es el uso de implantes inteligentes como el “Persona IQ”, desarrollado por Zimmer Biomet en colaboración con Canary Medical. Este dispositivo, implantado durante una cirugía de rodilla, monitorea el progreso del paciente, incluyendo métricas como el rango de movimiento y la actividad diaria, transmitiendo los datos al médico a través de una aplicación móvil (Financial Times, 2024).

Una tendencia notable en este campo es la integración de inteligencia artificial (IA) y aprendizaje automático en los sistemas IoT, lo que permite analizar grandes volúmenes de datos clínicos para detectar patrones y predecir enfermedades antes de que se manifiesten síntomas (Toxigon, 2024). Asimismo, se está explorando el uso de la tecnología blockchain para proteger los datos sensibles de los pacientes, creando registros descentralizados y resistentes a manipulaciones (Khatoon et al., 2023).

El aprendizaje federado también ha emergido como una técnica prometedora. Este enfoque permite a los dispositivos IoT entrenar modelos de IA sin compartir datos sensibles, lo que mejora la privacidad y seguridad de la información clínica (Yang et al., 2021).

2.2. Desarrollos tecnológicos actuales

a. Beurer BF1000 - Báscula Diagnóstica Bluetooth

- **Características:**

- Mide peso, grasa corporal, agua corporal, masa muscular, masa ósea, IMC y calorías (AMR/BMR).

- Interfaz intuitiva y multilinguaje (DE, EN, FR, ES, IT).
- Revestimiento ITO y electrodos de acero inoxidable.
- Capacidad de almacenamiento para 10 usuarios + modo invitado.
- Memoria para 30 mediciones.
- Conexión mediante la app Health Manager (Android/iOS).
- Capacidad de carga: 200 kg.
- **Uso recomendado:** Hogares, gimnasios y centros de entrenamiento.
- **Desventajas:** No permite la explotación avanzada de datos.
- **Precio:** \$8,270 MXN.

b. Quiosco Integrado de Revisión de Salud SH-N15

- **Características:**
 - Pantalla táctil LCD de 15 pulgadas.
 - Informes de diagnóstico en pantalla e impresión.
 - Medición de altura, peso, IMC, presión arterial y frecuencia cardíaca.
 - Medición de oxígeno en sangre, azúcar en sangre y colesterol.
 - Pruebas de visión, salud mental y alcohol.
 - Compatible con múltiples idiomas.
 - Movable con ruedas.
- **Uso recomendado:** Centros de salud y espacios comerciales.
- **Desventajas:** No almacena datos para consultas posteriores.

Precio: US\$ 1,299.00.

c. Quiosco SH-V20 - Báscula Inteligente

- **Características:**

- Reconocimiento facial inteligente (opcional).
 - Pantalla LCD táctil de 10.2 pulgadas + pantalla publicitaria de 19 pulgadas.
 - Impresión automática de resultados mediante impresora térmica.
 - Medición de altura, peso, IMC, presión arterial, frecuencia cardíaca y grasa corporal.
 - Transferencia de datos vía Bluetooth, Wi-Fi y 3G.
 - Opcional: Sistema de pago por monedas y microcréditos.
- **Uso recomendado:** Clínicas, gimnasios y negocios con autoservicio.
 - **Desventajas:** No cuenta con base de datos para seguimiento de usuarios.
 - **Precio:** US\$ 1,599.00.

2.3. Desafíos en la Implementación de IoT en Salud

A pesar de los avances, la adopción del IoT en salud enfrenta varios retos. La privacidad y la seguridad de los datos siguen siendo preocupaciones fundamentales, especialmente ante el incremento de ciberataques dirigidos a sistemas médicos (Kumar et al., 2023). Además, la interoperabilidad entre diferentes dispositivos y plataformas tecnológicas representa un obstáculo crítico, ya que no todos los sistemas están diseñados para comunicarse entre sí de forma eficiente (Sharma et al., 2024).

La falta de estandarización, regulaciones específicas y políticas claras para la integración del IoT en contextos clínicos también limita su expansión y aceptación generalizada en muchos entornos de salud.

Las aplicaciones de IoT en salud, sobrepeso y desnutrición continúan en expansión, con desarrollos que van desde sensores individuales hasta sistemas complejos de monitoreo y análisis de datos mediante machine learning. Sin embargo, desafíos como la integración de múltiples sensores, la precisión de las mediciones y la escalabilidad de las soluciones aún requieren mayor investigación y validación experimental.

La tecnología IoT está desempeñando un papel crucial en la transformación del sistema de salud global, ofreciendo soluciones avanzadas para la monitorización remota, el tratamiento personalizado y la atención preventiva. No obstante, para alcanzar su máximo potencial, es imprescindible superar los desafíos relacionados con la seguridad, la interoperabilidad y la

regulación. Las investigaciones recientes muestran que la evolución del IoT en salud continuará, especialmente con el respaldo de IA, blockchain y machine learning, construyendo así un sistema más inteligente y centrado en el paciente.

Capítulo 3 Metodología

Para el desarrollo de proyectos de Internet de las Cosas (IoT), es inminente el uso de metodologías que permitan la facilidad de la implementación, integración y escalabilidad de estos sistemas. Debido a la gran diversidad y al carácter multidisciplinario de aplicaciones basadas en soluciones IoT, así como de la gran variedad de software y hardware, (Luis-García & Torres Gómez, 2024), existe una gran diversidad de metodologías de desarrollo, que buscan cumplir con los desafíos técnicos y operativos de la actualidad en el diseño desarrollo e implementación de las aplicaciones.

Las metodologías para el desarrollo de IoT varían según las características del proyecto, el contexto tecnológico y las necesidades del cliente. Las metodologías ágiles y DevOps son ideales para proyectos que requieren flexibilidad y rapidez, mientras que enfoques como la metodología en cascada o el modelo basado en ciclos de vida son más adecuados para proyectos más estructurados o con menos cambios durante el desarrollo. La elección de la metodología debe alinearse con los objetivos y la naturaleza del proyecto IoT en cuestión, además se requiere que esta permita gestionar la complejidad técnica, la diversidad de dispositivos y la evolución constante de los requisitos, sin embargo, las metodologías preferidas por muchos desarrolladores son modelos de desarrollo ágil y en cascada.

3.1. Metodología utilizada

El modelo utilizado para el desarrollo del presente proyecto proporciona una guía estructurada para abordar proyectos de IoT, facilitando la integración de herramientas de software en cada etapa de desarrollo. Este modelo propuesto por León Torres-Restrepo et al. (2020), en la Figura 1, es un modelo que consta de 5 niveles, centra su enfoque en la etapa de análisis de la tecnología, las tres primeras etapas enfatizan en la conceptualización del método, las últimas dos hacen énfasis en el desarrollo de la solución. Esta metodología simplifica la construcción de sistemas basados en IoT, comparado con los modelos que tradicionalmente se utilizan, aunque evidentemente tiene influencias de los modelos de prototipado ágil, modelo de desarrollo en cascada y ciclo de vida de sistemas, este enfoque es bastante estructurado y tiene amplia similitud con una metodología adaptada para proyectos IoT desde una perspectiva de ingeniería de sistemas, combinando de manera acertada elementos clásicos con necesidades específicas de IoT, lo cual lo hace muy claro para gestionar proyectos desde cero, mantener equilibrio entre

estructura y flexibilidad, permite la facilidad de la documentación y la escalabilidad, además de que es ideal para la aplicación en contextos institucionales.

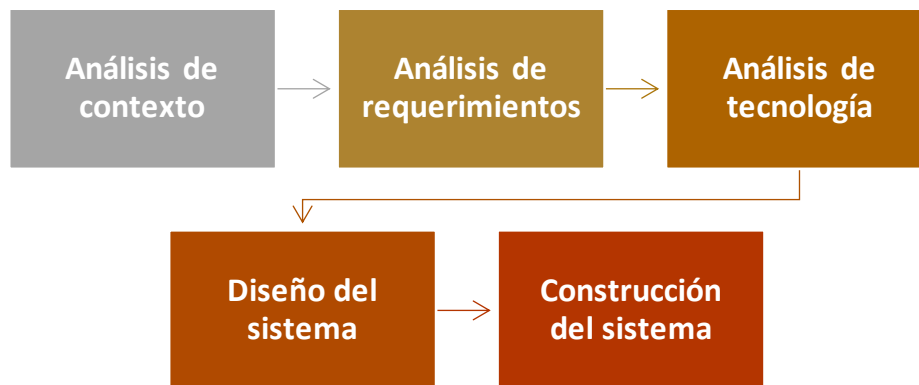


Figura 1. Modelo propuesto para el desarrollo de proyectos IoT. Adaptado de León Torres-Restrepo et al. (2020).

En la Tabla 1 se detallan cómo se relacionan los 5 niveles con una metodología estructurada para IoT:

Tabla 1. Niveles, etapas y descripción Modelo propuesto para el desarrollo de proyectos IoT. Adaptado de León Torres-Restrepo et al. (2020).

Nivel	Etapas	Descripción
1. Análisis de contexto	Exploración del problema	Se estudia el entorno donde se aplicará el sistema (ej. ciudad, industria, hogar, agricultura). Se detectan necesidades, usuarios, restricciones.
2. Análisis de requerimientos	Requisitos funcionales y no funcionales	Se define qué debe hacer el sistema (funciones, sensores, conectividad) y cómo (rendimiento, seguridad, costo, etc.).
3. Análisis de tecnología	Selección de componentes y plataformas	Se eligen sensores, actuadores, microcontroladores (ESP32, Arduino, etc.), protocolos (MQTT, Wi-Fi, LoRa), plataformas cloud, etc.
4. Diseño del sistema	Diseño técnico y funcional	Se crea el diagrama del sistema (arquitectura IoT), flujos de datos, esquemas de hardware y lógica de programación.
5. Construcción del sistema	Implementación del prototipo	Se ensamblan los componentes, se programa el firmware, se conectan servicios, se realiza testing inicial.

Capítulo 4 Desarrollo de la plataforma experimental

4.1. Análisis de contexto

El proyecto es una propuesta de solución para la problemática que se presenta en instituciones de educación media superior, en lo particular al interior de la Escuela Preparatoria Número 2, dependiente de la Universidad Autónoma del Estado de Hidalgo, el área operativa de la solución a desarrollar es la unidad de servicio médico de la institución, sin embargo, existe posibilidad de replicar la solución en el resto de las Escuelas e Institutos de la UAEH, así mismo, se estima también la posibilidad de que con la solución mínima funcional, existan posibilidades de desarrollar un modelo de negocio para atender la problemática dentro de la industria en diversas corporaciones.

4.1.1. Necesidades

- El monitoreo de variables como peso y estatura para el cálculo del IMC que es determinante para la detección de casos de desnutrición y obesidad.
- Entrega de los valores de las variables descritas en tiempo real.
- Almacenamiento de las mediciones a través de una base de datos en la nube.
- Descripción gráfica del histórico a partir de las series en el tiempo de las variables monitoreadas, para analizar el comportamiento y la tendencia generacional para su utilización de manera general.

4.1.2. Restricciones

- Uso de la base de datos de alumnos, académicos y administrativos, así como datos de expediente clínico para su tratamiento en la nube.
- Las variables calculadas con apoyo del sistema, solo son alimentadas a los expedientes clínicos por el titular del servicio médico de la institución.

Lo anterior referido en la base legal del *Manual de procedimientos de la Dirección de Servicio Médico Universitario* de la UAEH (Universidad Autónoma del Estado de Hidalgo, 2024).

4.1.3. Usuarios

La comunidad perteneciente a la institución de educación media superior, haciendo énfasis en el alumnado.

- Estudiantes
- Docentes
- Administrativos

4.1.4. Estudio de factibilidad

Es indispensable conocer si el desarrollo del proyecto propuesto es posible ser realizado, definiendo el problema a resolver, siendo útil en la solución del problema y apoyando o beneficiando al usuario final, a fin de contribuir con los objetivos de la organización, su misión y visión, además de determinar el presupuesto necesario para su desarrollo e implementación.

La determinación de las condiciones de desarrollo, de operación y la importancia que tiene la participación de las partes interesadas para la elaboración del proyecto son factores determinantes para la elaboración del estudio de factibilidad.

4.1.4.1. Participantes en el desarrollo del estudio de factibilidad

La Tabla 2 muestra el listado de las partes interesadas en el desarrollo del proyecto.

Tabla 2. Participantes, partes interesadas. Fuente: Elaboración propia.

1	Julia María Magdalena Rodríguez Sosa	Directora de la institución
2	Martha Elba Rojas Islas	Orientación Educativa
3	Idalia Ortiz Castillo	Trabajo Social
4	Diana María Arteaga Pozos	Psicología
5	José Juan Arista Hernández	Servicios Estudiantiles
6	Leobardo Leandro Peralta Guerrero	Responsable de Deportes
7	Daisy Karina Bravo Arellanos	Titular unidad médica Escuela Preparatoria No. 2

4.1.4.2. Instrumento de evaluación para estudio de factibilidad

El instrumento de evaluación para el estudio de factibilidad aplicado es la encuesta.

• Objetivo de la Encuesta

La encuesta es una herramienta de evaluación bastante útil para la captura y el aprovechamiento de datos que los usuarios, permite la creación de indicadores propios para obtener resultados y tomar decisiones. La encuesta está encaminada a la obtención de información que permita la verificación de la necesidad de la solución a la problemática que se presenta en el monitoreo de

la salud de estudiantes de bachillerato y la explotación de los datos que se obtienen con el monitoreo, a fin de prevenir situaciones de riesgo antes de que estas ocurran.

- **Población objetivo**

La población a la que se dirige la encuesta es a la representación de la comunidad de estudiantes de bachillerato, particularmente estudiantes de la Escuela Preparatoria Número Dos, así como personal académico y administrativo, sin embargo, es de suma importancia la opinión de la parte operativa del servicio médico universitario de la institución, así como de las áreas académicas correspondientes como partes interesadas descritas en la Tabla 2.

- **Encuesta**

- 1.- ¿Considera que el rendimiento escolar se ve afectado por la salud del estudiante?
- 2.- ¿Considera que es indispensable la cultura de la prevención en la salud antes de dar lugar a la atención para corregir las afectaciones?
- 3.- ¿Considera necesario evitar los problemas de obesidad y desnutrición en estudiantes?
- 4.- ¿Cree que los problemas psicológicos originados por situaciones derivadas por problemas de obesidad y desnutrición se pueden controlar de manera anticipada con ayuda de la tecnología?
- 5.- ¿Cree que la deserción escolar disminuiría de manera considerable si existiera un sistema de monitoreo para el cuidado de la salud de los estudiantes?
- 6.- ¿Considera que el desarrollo de una estación que permita el monitoreo constante de estatura y peso favorecerá la salud de los estudiantes?
- 7.- ¿Considera que sea necesario la generación inmediata de un reporte del estado de salud del estudiante, para el apoyo en la toma de decisiones en el cuidado de la misma?
- 8.- ¿Cree que los datos obtenidos del monitoreo constante de estatura y peso facilitarían el seguimiento continuo del estado de salud de un estudiante?
- 9.- ¿Considera que un sistema de alertamiento mediante una web app, podría anticipar al estudiante y al servicio médico sobre una posible afectación en la salud?
- 10.- ¿Considera que las herramientas tecnológicas como las anteriormente mencionadas, apoyarían y agilizarían la atención por parte del servicio médico universitario?

Las opciones de respuesta para esta encuesta son sí y no.

4.1.4.3. Interpretación de resultados

De las 10 preguntas anteriormente aplicadas, se obtuvo un nivel de aceptación considerable como respuesta por parte de los usuarios, 3 preguntas obtuvieron un 80% de aceptación, 5 preguntas el 90% y 2 preguntas el 100%, lo cual refleja de manera benéfica, que la muestra de la población a la que se aplicó la encuesta, está de acuerdo con la solución propuesta para la atención a la problemática planteada sobre la detección oportuna de desnutrición y obesidad, así como dar seguimiento al estado de salud del estudiante.

Las gráficas de los resultados de la encuesta se ilustran con las Figuras 2 – 11.

¿Considera que el rendimiento escolar se ve afectado por la salud del estudiante?
10 respuestas

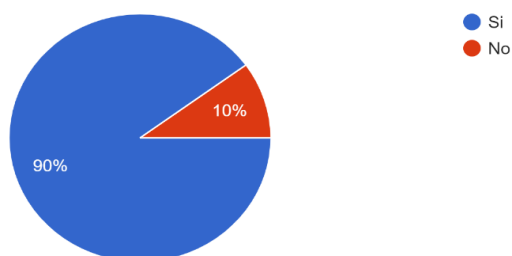


Figura 2. Distribución porcentual sobre si existe afectación al rendimiento escolar debido al estado de salud. Fuente: Elaboración propia.

¿Considera que es indispensable la cultura de la prevención en la salud antes de dar lugar a la atención para corregir las afectaciones?
10 respuestas

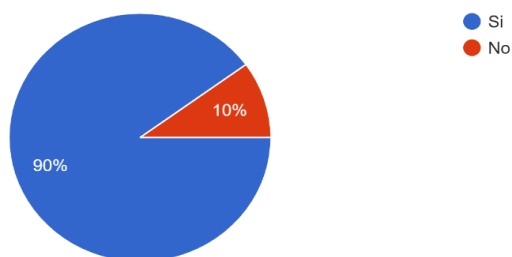


Figura 3. Distribución porcentual sobre la consideración a la prevención. Fuente: Elaboración propia.

¿Considera necesario evitar los problemas de obesidad y desnutrición en estudiantes?
10 respuestas

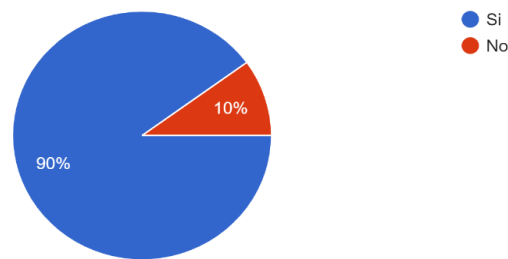


Figura 4. Distribución porcentual sobre la necesidad de evitar la obesidad y desnutrición. Fuente: Elaboración propia.

¿Cree que los problemas psicológicos originados por situaciones derivadas por problemas de obesidad y desnutrición se pueden controlar de manera anticipada con ayuda de la tecnología?
10 respuestas

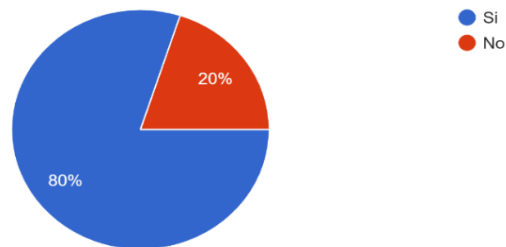


Figura 5. Distribución porcentual sobre el control de problemas de nutrición con ayuda de la tecnología. Fuente: Elaboración propia.

¿Cree que la deserción escolar disminuiría de manera considerable si existiera un sistema de monitoreo para el cuidado de la salud de los estudiantes?
10 respuestas

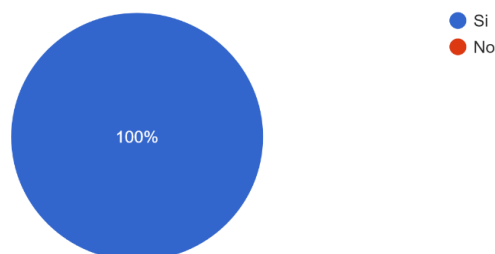


Figura 6. Distribución porcentual sobre la disminución en la deserción escolar debido al monitoreo de la salud. Fuente: Elaboración propia.

¿Considera que el desarrollo de una estación que permita el monitoreo constante de estatura y peso favorecerá la salud de los estudiantes?

10 respuestas

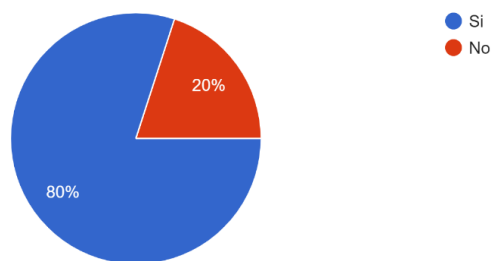


Figura 7. Distribución porcentual sobre la mejora en la salud mediante el desarrollo de una estación de monitoreo. Fuente: Elaboración propia.

¿Considera que sea necesario la generación inmediata de un reporte del estado de salud del estudiante, para el apoyo en la toma de decisiones en el cuidado de la misma?

10 respuestas

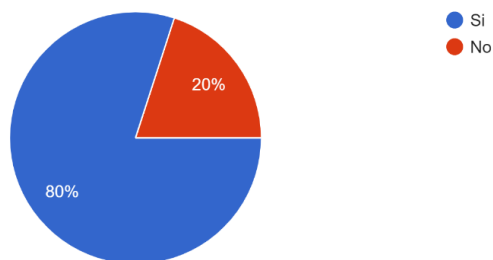


Figura 8. Distribución porcentual sobre la necesidad de toma de decisiones inmediata en salud. Fuente: Elaboración propia.

¿Cree que los datos obtenidos del monitoreo constante de estatura y peso facilitarían el seguimiento continuo del estado de salud de un estudiante?

10 respuestas

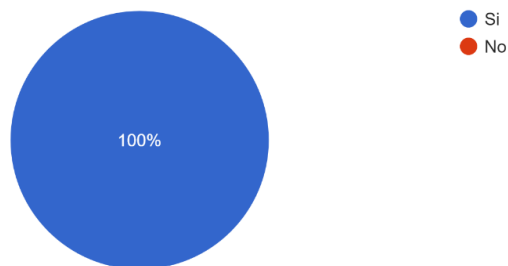


Figura 9. Distribución porcentual sobre el favorecimiento de la salud mediante monitoreo. Fuente: Elaboración propia

¿Considera que un sistema de alertamiento mediante una web app, podría anticipar al estudiante y al servicio médico sobre una posible afectación en la salud?

10 respuestas

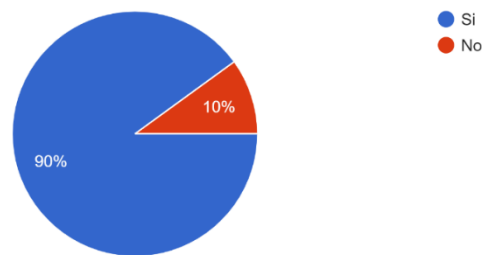


Figura 10. Distribución porcentual sobre el alertamiento para la prevención en la salud. Fuente: Elaboración propia.

¿Considera que las herramientas tecnológicas como las anteriormente mencionadas, apoyarían y agilizarían la atención por parte del servicio médico universitario?

10 respuestas

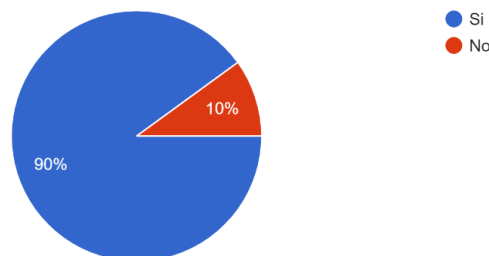


Figura 11. Distribución porcentual sobre la aceptación de herramientas tecnológicas en apoyo al servicio médico. Fuente: Elaboración propia.

4.2. Análisis de requerimientos

En la presente metodología se realiza un análisis de lo que se requiere, basado en la exploración del problema planteado en el análisis de contexto, tomando en cuenta las necesidades por naturaleza propia del entorno, las necesidades de la operabilidad del prototipo, así como de aspectos de costos y seguridad.

4.2.1. Requerimientos funcionales

En el diseño y desarrollo de sistemas basados en Internet de las Cosas (IoT), identificar claramente los requerimientos funcionales representa un paso clave para garantizar que la solución cumpla con su propósito de manera eficiente. Estos requerimientos describen las acciones concretas que el sistema debe ser capaz de realizar, desde la recolección de datos

hasta su procesamiento y visualización. En este caso, se plantea un sistema enfocado en el monitoreo de parámetros físicos del usuario, como parte de un enfoque preventivo y orientado al bienestar personal. Cada función responde a una necesidad práctica, permitiendo que el sistema no solo recopile información relevante, sino que también la convierta en conocimiento útil y accesible para quien lo utilice.

A continuación, se muestran los requerimientos funcionales detectados:

- **Medición de la masa corporal del usuario**

El sistema debe ser capaz de registrar con precisión la masa corporal del usuario utilizando sensores adecuados. Esta funcionalidad es fundamental, ya que proporciona uno de los datos base para el posterior cálculo del Índice de Masa Corporal (IMC). La medición debe realizarse de forma automática una vez que el sistema esté activado, mediante celdas de carga conectadas a una base que servirá de soporte y punto de apoyo al usuario.

- **Medición de la estatura del usuario**

Además de la masa, el sistema debe medir la estatura del usuario mediante sensores de distancia o tecnologías similares. Esta medición debe ser sencilla, rápida y cómoda para el usuario, permitiendo una experiencia fluida durante el proceso de registro de esta variable.

- **Cálculo del índice de masa corporal (IMC) del usuario**

A partir de las mediciones de masa y estatura, el sistema debe calcular de forma automática el IMC del usuario. Este valor sirve como indicador básico del estado físico general, permitiendo al usuario identificar posibles desviaciones en su salud relacionadas con el peso.

- **Registro y envío de datos a una plataforma online**

Una vez obtenidos y procesados los datos, el sistema debe contar con la capacidad de almacenarlos de manera segura y enviarlos a una plataforma en línea. Este paso es esencial para permitir el seguimiento histórico, el análisis de tendencias y el acceso remoto por parte del usuario o de profesionales de la salud.

- **Permitir la activación de los sensores mediante interruptor**

Para brindar mayor control al usuario, el sistema debe incluir un mecanismo manual o físico como un interruptor que permita iniciar el proceso de medición cuando el usuario así lo decida. Esto evita mediciones automáticas no deseadas y mejora la interacción con el dispositivo.

- **La conectividad debe ser Wi-Fi**

La transmisión de datos hacia la plataforma en línea debe realizarse a través de una conexión Wi-Fi. Esta forma de conectividad permite un enlace estable y de amplio alcance, facilitando la sincronización con servicios en la nube y asegurando que la información esté disponible en tiempo real.

- **Visualización de datos mediante un dashboard**

Los resultados obtenidos (masa, estatura e IMC) deben visualizarse de forma clara en un dashboard o panel digital accesible al usuario. Este panel debe presentar los datos de forma comprensible, permitiendo al usuario consultar su estado actual, fomentando el autocuidado y el seguimiento de su progreso.

4.2.2. Requerimientos no funcionales

Además de las funciones específicas que el sistema debe cumplir, existen ciertos aspectos técnicos y operativos que, aunque no están directamente relacionados con las tareas principales, son igual de importantes para su correcto desempeño. Estos se conocen como requerimientos no funcionales y tienen como objetivo asegurar que el sistema IoT sea eficiente, confiable, fácil de usar y sostenible en el tiempo. En esta etapa del diseño, se consideran factores como la usabilidad, el consumo energético, la escalabilidad y los recursos disponibles, ya que todos ellos influyen directamente en la calidad de la experiencia del usuario y en la viabilidad del proyecto.

Para el presente proyecto se identifican los siguientes requerimientos no funcionales.

- **Interfaz intuitiva para el usuario**

Es fundamental que el sistema cuente con una interfaz clara, sencilla y fácil de usar. La experiencia del usuario debe ser lo más fluida posible, permitiendo que cualquier persona independientemente de su nivel técnico pueda interactuar con el sistema sin necesidad de instrucciones complejas. Este aspecto no solo mejora la usabilidad, sino que también fomenta la adopción del sistema en contextos cotidianos.

- **Bajo consumo energético**

El diseño debe priorizar la eficiencia energética para prolongar la vida útil de los componentes, especialmente si se alimentan mediante baterías o fuentes limitadas. Un bajo consumo energético no solo reduce los costos operativos, sino que también hace que el sistema sea más sustentable, lo cual es especialmente relevante en soluciones tecnológicas modernas.

- **Fácil instalación y mantenimiento**

La simplicidad en la instalación y el mantenimiento del sistema es clave para su implementación exitosa. El dispositivo debe ser fácilmente ensamblable, con componentes accesibles y un diseño que no requiera conocimientos técnicos avanzados para su puesta en marcha o actualización. Esto favorece su uso en instituciones.

- **Condiciones óptimas para la escalabilidad**

El sistema debe estar preparado para crecer o adaptarse a nuevas necesidades en el futuro. Esto implica que su arquitectura permita integrar nuevos sensores, ampliar la capacidad de almacenamiento o adaptarse a distintas plataformas sin necesidad de rediseñar por completo la solución. La escalabilidad garantiza que el sistema siga siendo útil a largo plazo.

- **Alta confiabilidad y precisión en los datos**

Dado que se trabaja con datos relacionados con la salud y el bienestar, es esencial que las mediciones sean consistentes, precisas y confiables. La integridad de la información recolectada fortalece la utilidad del sistema y genera confianza en los usuarios, permitiéndoles tomar decisiones basadas en datos reales y verificables.

- **Presupuesto limitado**

Uno de los retos del proyecto es ajustarse a un presupuesto reducido. Por ello, es necesario buscar soluciones tecnológicas que ofrezcan un equilibrio entre costo, calidad y funcionalidad. Esta restricción impulsa el uso de materiales accesibles y estrategias de diseño eficientes, sin comprometer el desempeño general del sistema.

4.2.3. Factibilidad técnica

Para que la solución a desarrollar funcione de manera efectiva, es necesario se incluyan los siguientes requerimientos técnicos, a fin de evitar un mal funcionamiento o la interrupción del servicio.

- Energía eléctrica ininterrumpida, se requiere de una unidad UPS de respaldo.
- Un punto de acceso inalámbrico (WAP) que asegure la conectividad para el envío de datos.
- Equipo de cómputo.

4.2.4. Cálculo de costos aproximados de componentes sugeridos

La Tabla 3 muestra el costo aproximado de los componentes para el desarrollo del prototipo.

Tabla 3. Costos estimados de los componentes para la realización del prototipo. Fuente: Elaboración propia.

Cantidad	Componente	Costo
1	Módulo HX711 + 4 celdas de carga	\$399.00
1	Sensor de distancia laser VL53L0X	\$80.00
1	1 microcontrolador NODE MCU ESP8266	\$150.00
1	Base vidrio templado	\$260.00
1	Soporte de Metal Ajustable	\$295.00
1	Soporte Para Sensor VL53L0X	\$60.00
1	Servicio AWS IoT anual estimado	\$1,800
1	Cableado del sistema	\$900.00
1	Mantenimiento y Soporte	\$2,200
1	Integración e instalación	\$3,000
1	Servicio de luz bimestral	\$300.00
1	Servicio de internet mensual	\$400.00
	Total	\$9,844.00

4.2.5. Seguridad

La seguridad en los sistemas IoT representa uno de los principales desafíos en el desarrollo de soluciones tecnológicas confiables y sostenibles. A medida que estos sistemas se integran en entornos cada vez más complejos, se hace imprescindible identificar posibles vulnerabilidades que puedan comprometer la integridad, disponibilidad o confidencialidad de la información. En este sentido, el marco de referencia proporcionado por el portal OWASP Top Ten (OWASP, 2023) resulta esencial para reconocer las amenazas más comunes en aplicaciones web y aplicar medidas efectivas de prevención. A continuación, se presentan las vulnerabilidades más relevantes, junto con sus causas, consecuencias y estrategias recomendadas para su mitigación.

4.2.5.1. Control de acceso roto

Causas:

Esta vulnerabilidad se origina cuando no se aplican de manera adecuada los principios de denegación por defecto. Esto permite que ciertos usuarios puedan acceder a información o funciones sin contar con los privilegios necesarios, ya sea por elevación de privilegios, mala gestión de roles o ausencia de controles efectivos. (Fernández-Medina et al., 2022).

Efecto:

Entre los principales riesgos se encuentra la exposición de datos sensibles, la manipulación de cuentas y perfiles de usuario, así como el acceso no autorizado a metadatos o tokens JSON utilizados en sesiones activas.

Prevención:

Es fundamental establecer políticas de denegación por defecto, registrar intentos fallidos de acceso y activar alertas ante comportamientos sospechosos. Además, la incorporación de una unidad de control de acceso bien estructurada facilita la supervisión y reduce los riesgos de exposición indebida.

4.2.5.2. Fallas criptográficas**Causas:**

Las deficiencias en la implementación de mecanismos criptográficos suelen estar relacionadas con el uso de algoritmos débiles o desactualizados, la transmisión de datos sin cifrado, o el uso inadecuado de contraseñas como claves de cifrado. También influyen prácticas incorrectas en la generación de aleatoriedad y la falta de cumplimiento de estándares criptográficos. (ENISA, 2021).

Efecto:

Estas fallas comprometen la confidencialidad de información sensible, como contraseñas, datos financieros o registros médicos, los cuales suelen estar protegidos por marcos legales como los avisos de privacidad.

Prevención:

Se recomienda clasificar los datos almacenados o transmitidos para identificar aquellos que requieren protección especial. Asimismo, es importante evitar el almacenamiento innecesario de datos confidenciales y utilizar protocolos modernos como TLS con confidencialidad directa. El uso de cifrado autenticado y algoritmos actualizados contribuye a reforzar la seguridad del sistema.

4.2.5.3. Errores de configuración en la seguridad**Causas:**

Estos errores derivan comúnmente de configuraciones inadecuadas en el perímetro de seguridad, servicios en la nube o incluso permisos excesivos. Abrir puertos, mantener servicios

innecesarios activos o utilizar cuentas con privilegios predeterminados sin modificación aumenta significativamente el riesgo de intrusión.

Efecto:

Se genera una mayor exposición a ataques dirigidos a bases de datos, tanto relacionales como no relacionales. También pueden derivarse problemas por falta de actualización en certificados de seguridad o por el uso de software obsoleto, lo que incrementa la vulnerabilidad del sistema.

Prevención:

Es necesario aplicar buenas prácticas desde la instalación inicial del sistema, incluyendo la revisión periódica de certificados, parches de seguridad y permisos. También debe asegurarse el envío de encabezados de seguridad adecuados a los clientes y mantener el software actualizado conforme a los estándares recomendados.

4.2.5.4. Deficiencias en el registro de seguridad y monitoreo

Causas:

Esta vulnerabilidad aparece cuando el sistema carece de mecanismos adecuados para registrar y supervisar actividades relevantes. La escasa visibilidad de accesos no autorizados, junto con la falta de alertas y respuestas automatizadas ante incidentes, puede dejar brechas sin detectar durante largos períodos (González-Granadillo & Alcaraz, 2020).

Efecto:

La ausencia de registros de eventos como intentos de inicio de sesión, fallos del sistema o actividades anómalas dificulta la detección oportuna de amenazas. Esto puede permitir la fuga de información sin generar ninguna alerta activa, comprometiendo seriamente la seguridad del entorno.

Prevención:

Es esencial implementar una codificación segura para los registros, desarrollar un plan efectivo de respuesta ante incidentes, y establecer monitoreo continuo con generación de alertas en tiempo real. También debe garantizarse el control de accesos, la validación del lado del servidor y la conservación segura de los registros relevantes para su análisis posterior.

4.3. Análisis de tecnología

El análisis tecnológico cumple un papel fundamental dentro del proceso de desarrollo, ya que permite seleccionar de manera informada los componentes y plataformas que mejor se ajustan a

los requerimientos definidos en la fase anterior. Esta etapa exige una evaluación comparativa minuciosa, tanto del hardware como del software, con el objetivo de tomar decisiones que garanticen la funcionalidad y viabilidad del sistema. En este sentido, es indispensable realizar una selección detallada de elementos como sensores, actuadores y microcontroladores, considerando no solo sus características técnicas, sino también su compatibilidad con distintos protocolos de comunicación y su integración con plataformas especializadas en el desarrollo de soluciones IoT. Enfatizar en este análisis es clave para lograr un diseño sólido, que permita construir un sistema robusto, eficiente y sostenible en el tiempo.

4.3.1. Microcontroladores como eje funcional del Internet de las cosas (IoT)

En el contexto del desarrollo tecnológico actual, los microcontroladores se posicionan como componentes esenciales para la arquitectura del Internet de las Cosas (IoT, por sus siglas en inglés). Un microcontrolador es un sistema embebido en un solo chip que integra una unidad central de procesamiento (CPU), memoria (RAM, ROM o EEPROM) y periféricos de entrada/salida programables. Esta integración compacta permite controlar tareas específicas de forma autónoma, eficiente y con bajo consumo energético, lo que resulta crucial para aplicaciones distribuidas en IoT (Silva et al., 2020).

Lo que hace especialmente valioso al microcontrolador dentro del ecosistema IoT es su capacidad para interactuar con sensores y actuadores, que permiten la interacción eficiente entre dispositivos y sistemas digitales en un entorno o ambiente físico. Estos dispositivos compactos ejecutan tareas específicas con capacidad de ahorro de energía eléctrica y a un costo reducido, además de procesar información localmente y comunicar los datos relevantes a través de protocolos como MQTT, HTTP o CoAP.

A diferencia de los microprocesadores, que están diseñados para tareas de propósito general, los microcontroladores están optimizados para ejecutar tareas de carácter particular, lo que reduce la complejidad del sistema y aumenta su eficiencia. Este detalle técnico, aunque sutil, tiene un enorme impacto práctico: facilita el diseño de dispositivos autónomos y económicamente viables, incluso en escenarios de bajo presupuesto, lo cual es clave para la expansión masiva del IoT.

La evolución de plataformas accesibles como Arduino, ESP32 o STM32 ha democratizado el desarrollo de soluciones basadas en microcontroladores, permitiendo que tanto investigadores como desarrolladores independientes puedan experimentar, prototipar e implementar sistemas conectados sin necesidad de infraestructura compleja. Esta apertura tecnológica ha acelerado la

innovación en múltiples sectores, desde la salud y la agricultura, hasta la industria manufacturera y las ciudades inteligentes.

En el contexto del IoT, los microcontroladores facilitan la recopilación y procesamiento de datos en tiempo real, permitiendo la comunicación entre sensores, actuadores y plataformas de gestión. Esta capacidad es esencial para aplicaciones que requieren respuestas rápidas y autónomas, como la automatización industrial, la gestión energética y los sistemas de salud inteligentes (Ashok et al., 2023).

El crecimiento exponencial del IoT también ha impulsado la demanda de microcontroladores más potentes y versátiles. En este contexto, los microcontroladores no solo deben ser capaces de gestionar grandes volúmenes de datos, sino también garantizar la seguridad y la interoperabilidad entre dispositivos. Choudhary, 2024).

4.3.2. Comparativa de microcontroladores

La Tabla 4, muestra una comparativa de los microcontroladores de mayor demanda para el desarrollo de soluciones con IoT en la actualidad.

Tabla 4. Comparativa de microcontroladores. Fuente: Elaboración propia.

Característica	ESP8266	ESP32	STM32
Conectividad inalámbrica	Wi-Fi 2.4 GHz	Wi-Fi 2.4 GHz, Bluetooth 4.2/5.x	Algunas series con Bluetooth LE, LoRa, etc.
Procesador	Tensilica L106 de 32 bits	Xtensa LX6 de doble núcleo	ARM Cortex-M (M0/M3/M4/M7, etc.)
Frecuencia de operación	Hasta 160 MHz	Hasta 240 MHz	24 MHz a 800 MHz (según el modelo)
Memoria SRAM	160 KiB	Hasta 520 KiB	Varía según el modelo
Consumo energético	Muy bajo con modos de suspensión	Ultra bajo, múltiples estados de energía	Muy bajo en las series L y U
Ecosistema de desarrollo	Arduino, NodeMCU, RTOS SDK	ESP-IDF, Arduino, MicroPython	STM32CubeMX, Keil, IAR, etc.
Aplicaciones comunes	Sensores remotos, automatización del hogar e institucional.	Wearables, gateways, sistemas embebidos	Automatización industrial, salud, automotriz

4.3.3. El microcontrolador ESP8266

En el entorno actual del Internet de las Cosas (IoT), elegir el microcontrolador adecuado puede marcar la diferencia entre una solución eficiente y una de alto consumo o complejidad innecesaria. Entre los más populares, el ESP8266 de *Espressif Systems* se ha posicionado como uno de los más accesibles y prácticos para proyectos que requieren conectividad Wi-Fi confiable y de bajo costo.

El ESP8266 es un sistema en chip (SoC) que combina eficiencia, conectividad y bajo consumo energético. Su flexibilidad para operar de forma autónoma o como complemento de otros microcontroladores mediante comandos AT, lo convierte en una pieza central de muchas soluciones de automatización, monitoreo ambiental o control remoto (*Espressif Systems*, s.f.-a). Además, su compatibilidad con plataformas de desarrollo populares como Arduino y NodeMCU lo hacen muy accesible incluso para desarrolladores principiantes.

El ESP8266 destaca por su eficiencia y accesibilidad, siendo una opción ideal para proyectos de IoT que requieren conectividad Wi-Fi básica con bajo consumo energético y de modo económico. Además, soporta características como conectividad en la nube, siendo adecuado para una amplia gama de aplicaciones.

A diferencia del ESP8266, el ESP32 ofrece mayor potencia y versatilidad, adecuado para aplicaciones que demandan múltiples formas de conectividad y procesamiento más intensivo. (*Espressif Systems*, s.f.-b).

Por otro lado, la familia STM32 proporciona una amplia gama de opciones que permiten escalar en rendimiento y funcionalidades, siendo adecuada para aplicaciones más complejas o específicas que requieren un control más detallado del hardware y un ecosistema de desarrollo más amplio. (*STMicroelectronics*, s.f.).

Por lo anteriormente señalado, el ESP8266 es el microcontrolador ideal para el desarrollo del presente proyecto debido a la conectividad Wi-Fi que se requiere únicamente, su fácil programación en el IDE de Arduino, su capacidad de ahorro en consumo energético dado que el sistema a desarrollar no se encontrará muestreando datos de manera continua, sino que tendrá periodos de suspensión, su procesamiento sencillo durante la aplicación, además de ser el de más bajo costo.

4.3.4. Sensores para medición de distancia en IoT

Los sensores ultrasónicos, como el HC-SR04, destacan por su bajo costo y facilidad de uso, aunque presentan limitaciones en ambientes ruidosos o con superficies irregulares, ideales para aplicaciones educativas y de prototipado rápido (Nakamura et al., 2019).

Los sensores infrarrojos como el GP2Y0A21YK0F, son útiles en distancias cortas, pero su desempeño puede verse afectado por las condiciones lumínicas. En contraste, los sensores LIDAR (Light Detection and Ranging), como el VL53L0X y VL53L1X basada en tiempo de vuelo (*ToF*) y el TFMMini-S, ofrecen alta precisión y mejor adaptabilidad a entornos complejos, siendo ideales para aplicaciones más exigentes, además de ser inmunes al ruido acústico y se adaptan bien a sistemas embebidos debido a su tamaño reducido y bajo consumo energético (Sicari et al., 2015).

Estos sensores no solo permiten recopilar datos sobre el entorno físico, sino también mejorar la eficiencia, la seguridad y la autonomía de los sistemas conectados. Su correcta selección e implementación representa un elemento clave para el éxito de cualquier solución IoT, considerando tanto los requerimientos técnicos como el contexto de uso.

La Tabla 5 muestra una comparación técnica de los sensores de distancia más utilizados en IoT.

Tabla 5. Comparativa de sensores de distancia. Fuente: Elaboración propia

Sensor	Tecnología	Rango	Precisión	Interfaz	Consumo	Ventajas
HC-SR04	Ultrasónica	2 cm – 400 cm	±3 mm	GPIO (digital)	Bajo	Económico, fácil de usar
VL53L0X	LIDAR (ToF)	3 mm – 200 cm	±3%	I2C	Muy bajo	Muy preciso y compacto
VL53L1X	LIDAR (ToF)	3 mm – 400 cm	±1%	I2C	Bajo	Mayor rango y precisión que VL53L0X
GP2Y0A21YK0F	Infrarrojo (analógico)	10 cm – 80 cm	±5%	Analógica	Medio	Estable, fácil de leer
TFMini-S	LIDAR (ToF)	30 cm – 12 m	±1%	UART / I2C	Bajo-medio	Largo alcance, buena precisión
MB1040 (MaxBotix)	Ultrasónica	20 cm – 765 cm	±1%	Analógica / PWM / Serial	Bajo	Alta calidad, varias salidas

4.3.4.1. Sensor de distancia VL53L0X

El sensor VL53L0X LIDAR (Light Detection and Ranging), basado en tecnología *Time-of-Flight*, destaca por su capacidad de medir distancias cortas con gran precisión. Gracias a su reducido tamaño y su interfaz I2C, es especialmente adecuado para proyectos de IoT que utilizan microcontroladores compactos como el ESP8266, lo que facilita su integración en dispositivos móviles, wearables y sistemas robóticos, sin sacrificar precisión ni eficiencia energética (STMicroelectronics, 2016).

Esta combinación permite implementar soluciones eficientes de detección de proximidad en aplicaciones como automatización del hogar, control de acceso o sistemas interactivos, todo ello manteniendo una arquitectura ligera y de bajo costo.

El uso del VL53L0X con el ESP8266 se ha vuelto común debido a la facilidad de comunicación entre ambos dispositivos y la amplia disponibilidad de librerías en plataformas como Arduino, lo que facilita su programación e integración en entornos conectados a la nube.

4.3.5. Sensores para medición de peso en IoT

La implementación de sensores de peso en soluciones de IoT ha revolucionado la forma en que se gestionan procesos en sectores como la agricultura, el comercio, la automatización industrial y la salud. Estos sensores permiten cuantificar de forma precisa la carga o masa de objetos, proporcionando datos que pueden ser transmitidos, almacenados y analizados en tiempo real, lo cual es relevante para la toma de decisiones automatizadas.

Entre las tecnologías disponibles, el uso de galgas extensiométricas junto a módulos de acondicionamiento como el HX711 destaca por su notable equilibrio entre bajo costo, alta resolución (24 bits) y amplio soporte comunitario. Este sensor, se ha convertido en el estándar de facto para desarrollos educativos y prototipos funcionales.

Por otro lado, soluciones como el NAU7802 ofrecen mayor estabilidad operativa y comunicación por I2C, siendo preferidas en entornos más exigentes. Alternativas más avanzadas como SparkFun OpenScale o el Loadstar iLoad Mini integran funcionalidades adicionales como monitoreo directo por USB y calibración de fábrica, lo que facilita su uso en aplicaciones comerciales, aunque a un costo considerablemente mayor.

La diversidad de sensores permite una amplia gama de aplicaciones, desde básculas inteligentes conectadas hasta sistemas de pesaje automatizado en tiempo real. Esta versatilidad convierte a los sensores de peso en IoT en una herramienta crítica para aumentar la eficiencia operativa, reducir errores humanos y avanzar hacia procesos autónomos (Yick et al., 2008).

Una comparativa técnica detallada de los principales sensores se muestra en la Tabla 6.

Tabla 6. Comparativa de sensores de peso. Fuente: Elaboración propia.

Sensor	Tipo / Principio	Rango	Precisión	Interfaz	Ventajas Destacadas	Desventajas
HX711 + Célula de carga	Galga extensiométrica + ADC de 24 bits	0–50 kg (según celda)	±0.01 kg	Digital (2 pines GPIO)	Muy económico Alta resolución (24 bits) Gran soporte comunitario Compatible con ESP8266/ESP32/Arduino o Librerías disponibles y fáciles de usar	Necesita calibración manual Sensible al ruido eléctrico sin buen filtrado
NAU7802 + Célula de carga	Galga extensiométrica + ADC 24-bit	0–50 kg (según celda)	Alta (mayor estabilidad que HX711)	I2C	Alta precisión Fácil integración con microcontroladores modernos Más estable en ambientes industriales	Menos documentación Más costoso que HX711
TAL220 (usada con HX711 o NAU7802)	Celda de carga (strain gauge)	0–5 kg	±1% F.S.	N/A (requiere ADC)	Robusta, económica y ampliamente usada	No funciona sola, requiere amplificador
SparkFun OpenScale	Sensor + ADC + microcontrolador	Depende de la celda	Alta	UART / USB	Solución integrada Fácil monitoreo desde PC Diseñada para prototipos rápidos	Costo más alto Menos flexible en personalización
Loadstar iLoad Mini	Sensor digital con galga interna	0–100 kg	Muy alta (precisión calibrada)	USB / Serial	Precisión industrial Plug-and-play No requiere calibración manual	Muy costoso Sobrecalificado para proyectos hobby

4.3.5.1. Sensor de peso HX711

El módulo HX711 utilizado en sistemas de medición automatizada, procesos industriales e industria médica, es una alternativa precisa y económica para la medición de peso en aplicaciones IoT como básculas inteligentes, sistemas de dispensado automático y monitoreo de

carga en tiempo real, destacando especialmente por su integración con microcontroladores como el ESP8266. Gracias a su alta resolución (24 bits) y su compatibilidad con celdas de carga de hasta 50 kg, permite desarrollar sistemas inalámbricos de pesaje con conectividad Wi-Fi. Aunque requiere calibración manual, su implementación resulta sencilla debido a la amplia disponibilidad de recursos y librerías disponibles en plataformas como Arduino y ESP-IDF, que facilitan su integración incluso para desarrolladores con poca experiencia, lo que lo convierte en una solución escalable y confiable en entornos automatizados (Mikhaylov, Plevritakis & Tervonen, 2013).

4.3.6. Protocolos de comunicación IoT

El Internet de las Cosas (IoT) ha transformado la manera en que interactuamos con el mundo digital, permitiendo la comunicación y la interacción entre dispositivos conectados. Uno de los elementos fundamentales para el funcionamiento eficiente de los sistemas IoT es la elección del protocolo de comunicación adecuado. Existen varios protocolos de comunicación, cada uno diseñado para abordar necesidades específicas en términos de fiabilidad, consumo energético, alcance y velocidad de transmisión. Este análisis profundiza en los principales protocolos utilizados en IoT, destacando sus características, ventajas, desventajas y aplicaciones comunes.

4.3.6.1. MQTT (Message Queuing Telemetry Transport)

MQTT (Transporte de Telemetría por Cola de Mensajes) es un protocolo basado en el modelo de **publicación/suscripción**, que se caracteriza por su ligereza y su eficiencia en entornos con recursos limitados, como dispositivos IoT. Este protocolo funciona sobre TCP/IP y ofrece tres niveles de Calidad de Servicio (QoS) que permiten ajustar la fiabilidad de la transmisión según las necesidades de la aplicación. La arquitectura de MQTT se centra en un servidor central, denominado **broker**, que gestiona la distribución de los mensajes entre los dispositivos conectados.

MQTT es un protocolo *M2M* (machine to machine) de tipo *message queue*, creado por el Dr. Andy Stanford Clark IBM y Arlen Nipper Arcom para conectar mecanismos en la industria del petróleo. (HiveMQ, 2023)

Una de las principales ventajas de MQTT es su escalabilidad, ya que permite manejar un gran número de dispositivos de manera eficiente. Su bajo consumo de ancho de banda lo hace adecuado para redes con alta latencia o en situaciones donde el acceso a la red es intermitente. Sin embargo, MQTT depende de un **broker** central, lo que puede ser un inconveniente si este servidor experimenta fallos o se vuelve inaccesible. Además, para asegurar una comunicación confiable, es necesario garantizar que los dispositivos estén correctamente configurados con el **broker** (Wytrębowicz, Cabaj & Krawiec, 2021).

Aplicaciones comunes: Dispositivos con recursos limitados, redes con alta latencia, IoT industrial, y en aplicaciones de monitoreo de salud o automotriz, donde la eficiencia de la transmisión es crítica.

4.3.6.2. CoAP (Constrained Application Protocol)

CoAP (Protocolo de Aplicación para Entornos Restringidos) es otro protocolo orientado a la comunicación en dispositivos con recursos limitados. A diferencia de MQTT, CoAP utiliza el modelo **cliente/servidor** y opera sobre *UDP*, lo que reduce la sobrecarga en la comunicación. CoAP se destaca por su capacidad de integrar fácilmente las aplicaciones IoT con la web, ya que sigue un modelo similar a HTTP, pero es más eficiente para ambientes de baja capacidad.

Al utilizar *UDP*, CoAP permite una mayor velocidad de transmisión y menor latencia, pero con un compromiso en términos de fiabilidad. Este protocolo incluye características como la capacidad de realizar retransmisiones automáticas, y su compatibilidad con *IPv6* y *6LoWPAN* lo hace muy adecuado para redes de sensores en entornos industriales y de automatización del hogar (Gavrilov et al., 2022).

Aplicaciones comunes: Redes de sensores, automatización del hogar, monitoreo de dispositivos IoT con recursos limitados y aplicaciones en ciudades inteligentes.

4.3.6.3. AMQP (Advanced Message Queuing Protocol)

AMQP (Protocolo Avanzado de Encolamiento de Mensajes) es un protocolo orientado a mensajes que proporciona un conjunto más avanzado de características que MQTT, tales como enrutamiento flexible, transacciones confiables y seguridad robusta. Este protocolo es ideal para aplicaciones que requieren un alto nivel de fiabilidad en la transmisión de datos, como en el caso de sistemas financieros o aplicaciones empresariales críticas. AMQP opera sobre TCP y garantiza la entrega de los mensajes, incluso en caso de fallos de la red o desconexión de los dispositivos.

A pesar de sus ventajas, AMQP tiene un mayor consumo de recursos que otros protocolos como MQTT y CoAP, lo que lo hace menos adecuado para entornos con dispositivos de bajo consumo energético. Sin embargo, su capacidad para manejar mensajes complejos y asegurar la entrega confiable lo convierte en la opción preferida para aplicaciones industriales de misión crítica (Sun-Chul et al., 2022).

Aplicaciones comunes: Entornos industriales, aplicaciones empresariales, sistemas bancarios y en áreas donde la seguridad y fiabilidad de la comunicación son esenciales.

4.3.6.4. HTTP (Hypertext Transfer Protocol)

HTTP (Protocolo de Transferencia de Hipertexto) es ampliamente utilizado en entornos IoT debido a su compatibilidad con redes IP y su integración con servicios web *RESTful*. Aunque fue diseñado originalmente para la navegación web, su sencillez y soporte global lo han convertido en un estándar para la comunicación entre dispositivos IoT y servidores en la nube (Fielding & Taylor, 2002). Su modelo **cliente-servidor** se adapta bien a arquitecturas donde los dispositivos recolectan datos que son enviados periódicamente a servicios centrales para su procesamiento.

Aplicaciones comunes: Dispositivos inteligentes en hogares conectados (termostatos, cámaras IP), sensores ambientales que reportan datos a plataformas basadas en la nube, control y monitoreo remoto a través de dashboards web, Interfaces REST para dispositivos que requieren interoperabilidad rápida.

4.3.6.5. DDS (Data Distribution Service)

DDS (Servicio de Distribución de Datos) ha sido diseñado específicamente para sistemas distribuidos que requieren comunicación en tiempo real y alta confiabilidad. A diferencia de HTTP, DDS implementa un modelo **publicación/suscripción**, que permite que los dispositivos IoT publiquen datos a múltiples suscriptores sin necesidad de intermediarios, lo cual reduce la latencia y mejora la escalabilidad (Schmidt et al., 2018). Su capacidad para priorizar el tráfico y controlar la calidad del servicio lo convierte en una opción robusta para entornos críticos.

Aplicaciones comunes: Sistemas avanzados de conducción autónoma y vehículos conectados, redes industriales con sensores de alta frecuencia y bajo retardo, automatización en infraestructura crítica (energía, petróleo, salud), plataformas de robótica colaborativa e industria 4.0.

4.3.6.6. JMS (Java Message Service)

JMS (Servicio de Mensajería de Java) fue concebido principalmente para aplicaciones empresariales dentro del ecosistema Java, su modelo de mensajería asincrónica también encuentra espacio en soluciones IoT que requieren intercambio fiable de mensajes entre dispositivos y plataformas backend. Funciona bien cuando se necesita desacoplar el envío y recepción de datos, facilitando así arquitecturas escalables con capacidad de buffering (Oracle, 2023).

Aplicaciones comunes: Gestión de flotas logísticas donde los dispositivos informan en tiempo real, integración de sensores en entornos empresariales (como edificios inteligentes), coordinación de dispositivos IoT con middleware empresarial.

4.3.6.7. XMPP (Extensible Messaging and Presence Protocol)

XMPP (Protocolo Extensible de Mensajería y Presencia) Basado en XML, es particularmente valioso en entornos IoT donde se requiere mensajería en tiempo real entre dispositivos, con capacidad de mantener sesiones activas y presencia. Su estructura extensible permite la personalización para adaptarse a distintas necesidades IoT, desde redes de sensores hasta aplicaciones móviles conectadas (Saint-Andre, 2011).

Aplicaciones comunes: Comunicación **M2M** en hogares inteligentes o fábricas conectadas, plataformas de mensajería para dispositivos portátiles o wearables, coordinación entre nodos en redes distribuidas de sensores, juegos y dispositivos interactivos conectados entre sí.

4.3.6.8. Protocolo MQTT y microcontrolador ESP8266 beneficios y ventajas

El protocolo MQTT (*Message Queuing Telemetry Transport*) es ampliamente utilizado en el campo del Internet de las Cosas (IoT) debido a su ligereza y eficiencia en redes con dispositivos limitados. Cuando se emplea en conjunto con microcontroladores como el ESP8266, el protocolo ofrece una solución económica y escalable para una variedad de aplicaciones IoT. El ESP8266, se conecta fácilmente a redes Wi-Fi, además facilita la comunicación remota de dispositivos sin cables, lo que lo convierte en una opción popular para proyectos IoT, y desde luego el protocolo de comunicación ideal para implementar en el desarrollo del presente proyecto.

Una de las principales ventajas de usar MQTT con el ESP8266 es su bajo consumo de recursos. MQTT es un protocolo basado en el modelo de **publicación/suscripción** y de mensajería **push**, de tal manera que los mensajes son organizados jerárquicamente a manera de tópicos, de esta forma uno de los clientes puede publicar un mensaje en cada tópico y otros clientes pueden suscribirse a estos mismos, y de esta manera minimizar el tráfico de red al permitir que los dispositivos se conecten a un servidor denominado *broker* central, el cual será el encargado de entregar los mensajes, eliminando la necesidad de mantener conexiones directas entre ellos. Los clientes inician una conexión TCP/IP con el *broker*, el cual mantiene un registro de los clientes conectados. Esta conexión se mantiene abierta hasta que el cliente la finaliza. Por defecto, MQTT emplea el puerto **1883 y el 8883** cuando funciona sobre *TLS*. Esto es especialmente útil en entornos donde el ancho de banda es limitado o la conectividad es intermitente (Srinivas et al., 2021). Además, el protocolo es altamente eficiente en términos de latencia y fiabilidad, lo que permite que las aplicaciones IoT, como la automatización del hogar o el monitoreo industrial, operen en tiempo real sin problemas significativos de retraso.

Otro beneficio clave es la facilidad de implementación. Tanto MQTT como el ESP8266 tienen una amplia documentación y soporte comunitario, lo que hace que la integración de ambos sea relativamente sencilla, incluso para desarrolladores sin mucha experiencia (Rahman & Ahmed, 2022). Esto permite a los usuarios construir sistemas IoT personalizados y rentables con un tiempo de desarrollo reducido.

En términos de aplicaciones, la combinación de MQTT y el ESP8266 es especialmente efectiva para proyectos de monitoreo remoto, automatización de sistemas y control de dispositivos conectados. La capacidad del ESP8266 para gestionar redes Wi-Fi y la flexibilidad de MQTT para comunicarse con una gran cantidad de dispositivos hacen que esta combinación sea ideal para soluciones IoT escalables, como en ciudades inteligentes o sistemas de control industrial a gran escala (Kumari & Kumar, 2021).

4.3.6.9. Comparativa y selección del broker

Un *broker* actúa como un intermediario que gestiona la comunicación entre los dispositivos conectados. Su función principal es recibir, filtrar y distribuir mensajes entre los distintos nodos de la red, asegurando que cada dato llegue a su destino correcto. Gracias a su arquitectura ligera y eficiente, los *brokers* permiten que los dispositivos trabajen de forma autónoma y coordinada, incluso en entornos con conectividad limitada. Esta figura técnica se vuelve esencial en sistemas donde la fluidez y fiabilidad de la información definen el éxito de la solución implementada. (IEEE Computer Society, 2024).

La Tabla 7 muestra una comparación de las características fundamentales entre diversos *broker* de uso gratuito o prueba y de pago, destacando los beneficios y ventajas de **AWS IoT Core**.

Tabla 7. Comparativa de brokers de uso gratuito y de pago. Fuente: Elaboración propia

Característica	Brokers de Uso Libre (EMQX, Mosquitto)	Brokers de Pago (AWS IoT Core)
Costo	Gratuito	Basado en uso (Pay-as-you-go)
Escalabilidad	Limitada; requiere configuración manual	Alta; escalado automático
Gestión de dispositivos	Básica; requiere herramientas adicionales	Integrada; gestión centralizada
Seguridad	Configurable; depende del usuario	Seguridad avanzada integrada
Integración con servicios	Limitada; requiere desarrollo adicional	Amplia; integración nativa con servicios de AWS
Soporte técnico	Comunidad; limitado	Soporte profesional 24/7

4.3.6.10. AWS IoT Core

AWS IoT Core se distingue como un broker de pago que ofrece múltiples ventajas sobre las opciones de uso libre, especialmente en términos de escalabilidad, seguridad y gestión (AWS, 2024).

- **Escalabilidad automática:** Gestiona millones de dispositivos sin la necesidad de infraestructura adicional. Esto permite que las soluciones IoT crezcan de manera flexible y sin problemas, adaptándose a las necesidades cambiantes.
- **Seguridad avanzada:** Ofrece autenticación mutua, cifrado de datos y cumplimiento con normas de seguridad rigurosas, lo que asegura que los datos y dispositivos estén protegidos en todo momento.
- **Integración nativa con AWS:** Se integra fácilmente con otros servicios de AWS como *Lambda*, *DynamoDB*, *S3*, *CloudWatch* y *Machine Learning*, lo que facilita el procesamiento, almacenamiento y análisis de datos sin complicaciones.
- **Gestión centralizada:** AWS IoT Core ofrece una consola unificada que permite monitorear y gestionar dispositivos, políticas y flujos de datos desde un único lugar, simplificando la administración de soluciones IoT grandes y complejas.
- **Soporte técnico:** A diferencia de los *brokers* libres, AWS IoT Core incluye soporte profesional las 24 horas, lo que asegura que las empresas puedan resolver rápidamente problemas y optimizar sus soluciones IoT.

Dadas estas características y ventajas sobre los *brokers* gratuitos, además, de acuerdo al análisis de requerimientos, **AWS IoT Core** es el servicio óptimo para el desarrollo del presente proyecto.

4.3.7. Aplicación de bases de datos no relacionales en IoT

Las bases de datos no relacionales han dejado de ser una novedad para convertirse en una herramienta indispensable en el Internet de las Cosas. Su capacidad para gestionar datos diversos, adaptarse a entornos cambiantes y ofrecer un rendimiento sostenido frente a cargas elevadas las posiciona como una opción estratégica para desarrolladores. En un mundo cada vez más conectado, contar con una infraestructura de datos ágil y robusta no es solo una ventaja, sino una necesidad.

El cúmulo de información generado por dispositivos interconectados, ha evidenciado las limitaciones de los sistemas de gestión de bases de datos tradicionales, que no siempre están diseñados para manejar la diversidad y el dinamismo de los datos del IoT. Ante este panorama, las bases de datos no relacionales también conocidas como **NoSQL**, han cobrado relevancia

como una alternativa eficiente para almacenar, procesar y consultar datos en entornos distribuidos y de alta demanda.

4.3.7.1. Características de las bases de datos no relacionales

A diferencia de las bases de datos relacionales, que se estructuran a partir de esquemas rígidos y tablas interrelacionadas, las bases de datos **NoSQL** ofrecen una arquitectura más flexible. Están diseñadas para gestionar tanto datos estructurados como no estructurados o semiestructurados, lo que permite adaptarse mejor a entornos cambiantes y necesidades específicas. Su escalabilidad horizontal y su alta disponibilidad las hacen especialmente útiles en aplicaciones donde el rendimiento y la velocidad son esenciales (Yadav, 2024).

Existen diversos tipos de bases de datos **NoSQL**, cada uno con ventajas particulares:

- **Bases de datos orientadas a documentos:** organizan la información en estructuras como *JSON* o *BSON*, lo que facilita la representación de datos complejos.
- **Bases de datos clave-valor:** permiten acceder rápidamente a los datos a través de una clave única.
- **Bases de datos orientadas a columnas:** optimizadas para consultar grandes volúmenes de datos por columnas, lo cual mejora el rendimiento analítico.
- **Bases de datos de grafos:** ideales para modelar relaciones complejas, como las que se presentan en redes sociales o sistemas de recomendación.

4.3.7.2. Aplicación de NoSQL en el contexto de IoT

Uno de los principales retos del IoT es la necesidad de capturar y procesar datos generados por múltiples fuentes como sensores, dispositivos móviles o sistemas embebidos de manera eficiente y en tiempo real. Aquí es donde las bases de datos **NoSQL** encuentran un terreno fértil para su aplicación. Gracias a su diseño flexible, permiten no solo almacenar información heterogénea, sino también escalar fácilmente cuando aumenta el volumen de datos (Singh, 2019).

Como ejemplo de bases de datos **NoSQL** se encuentra *MongoDB*, una base de datos orientada a documentos, que ha sido ampliamente adoptada en proyectos IoT por su facilidad para almacenar registros provenientes de sensores con estructuras variables. Otras bases como *Apache Cassandra* y *Redis* han probado ser particularmente eficaces en situaciones donde se requiere disponibilidad continua y baja latencia, como en la automatización industrial o el monitoreo en tiempo real de infraestructuras críticas.

4.3.7.3. Desafíos y consideraciones clave

A pesar de sus ventajas, implementar bases de datos *NoSQL* en arquitecturas IoT no está exento de desafíos. Uno de los aspectos más relevantes es la consistencia eventual, un modelo que prioriza la disponibilidad y la tolerancia a particiones, pero que puede llevar a estados transitorios en los que los datos no estén completamente sincronizados. Además, el manejo de esquemas dinámicos requiere un diseño cuidadoso para evitar la pérdida de integridad o problemas de compatibilidad a largo plazo (Yadav, 2024).

Por ello, al elegir una solución *NoSQL* para un sistema IoT, es importante evaluar no solo la capacidad técnica, sino también la madurez del proyecto, los requerimientos de latencia, la complejidad del modelo de datos y las necesidades futuras de escalabilidad.

4.3.8. Bases de datos de AWS

Amazon Web Services (AWS), cuenta con dos servicios sumamente completos para la manipulación del amplio volumen y la naturaleza dinámica de los datos, su gestión y almacenamiento en la nube, estos servicios son ***DynamoDB y Timestream***, tecnologías que, gracias a sus bondades y extensa funcionalidad, fortalecen el desarrollo de este proyecto.

La Tabla 8, muestra sus principales características:

Tabla 8. Características de DynamoDB y Timestream. Fuente: Elaboración propia

Característica	DynamoDB	Timestream
Tipo de base de datos	NoSQL (clave-valor y documentos)	Series temporales
Arquitectura	Distribuida y sin servidor	Sin servidor con almacenamiento jerárquico
Casos de uso principales	Aplicaciones web, móviles, juegos, IoT	Monitoreo en tiempo real, análisis de métricas
Escalabilidad	Automática, manejo de millones de solicitudes	Escalabilidad automática para ingesta y consultas
Integración con AWS	Alta (Lambda, API Gateway, etc.)	Alta (QuickSight, Grafana, etc.)
Gestión del ciclo de vida	Manual mediante TTL y políticas de replicación	Automática según políticas definidas

4.3.8.1. Amazon DynamoDB

Amazon DynamoDB es un servicio de base de datos *NoSQL* de alta disponibilidad completamente gestionado, que proporciona un rendimiento consistente y predecible a cualquier escala. Su arquitectura distribuida y sin servidor permite manejar cargas de trabajo intensivas, ofreciendo baja latencia y alta disponibilidad. **DynamoDB** es particularmente adecuado para aplicaciones que requieren operaciones rápidas y escalables, como juegos en línea, comercio electrónico y sistemas de IoT.

DynamoDB manejó picos de hasta 89.2 millones de solicitudes por segundo, manteniendo una latencia de un solo dígito en milisegundos durante eventos de alta demanda, como el Prime Day de Amazon en 2021 (Elhemali et al., 2022). Esta capacidad de escalar automáticamente sin intervención manual lo convierte en una opción robusta para aplicaciones críticas.

4.3.8.2. Amazon Timestream

Amazon Timestream es un servicio de base de datos diseñado específicamente para almacenar y analizar datos de series temporales, como métricas de sensores y registros de eventos. Su arquitectura sin servidor permite la ingesta y consulta de grandes volúmenes de datos con eficiencia y rapidez. **Timestream** gestiona automáticamente el ciclo de vida de los datos, manteniendo los datos recientes en almacenamiento en memoria y trasladando los datos históricos a almacenamiento optimizado en costos (Amazon Web Services, 2023). Esta funcionalidad es especialmente beneficiosa para aplicaciones de monitoreo en tiempo real, donde la capacidad de analizar tendencias y patrones rápidamente es relevante. Además, **Timestream** ofrece funciones analíticas integradas y se integra fácilmente con herramientas de visualización como *Amazon QuickSight* y *Grafana*.

4.3.9. Visualización de datos en la nube con Amazon Managed Grafana

Amazon Managed Grafana representa una solución robusta y flexible para la visualización de datos en entornos de nube. Su integración con servicios de AWS y fuentes de datos externas, junto con su enfoque en la seguridad y la personalización en el análisis de datos la convierten en una herramienta poderosa. Esta plataforma, basada en el proyecto de código abierto **Grafana**, ofrece una solución completamente gestionada para la visualización de métricas, registros y trazas, facilitando la observabilidad de sistemas distribuidos en entornos de nube.

Amazon Managed Grafana es sin duda una herramienta integral para la construcción de dashboards para la visualización de datos. La Tabla 9, especifica sus principales características.

Tabla 9. Características de Amazon Managed Grafana. Fuente: Elaboración propia

Categoría	Descripción
Naturaleza del servicio	Solución completamente gestionada basada en Grafana. AWS se encarga de escalabilidad, mantenimiento y disponibilidad.
Integración de datos	Compatible con CloudWatch, X-Ray, Prometheus, Timestream, OpenSearch y servicios externos como Datadog.
Control de acceso	Soporta IAM Identity Center y SAML 2.0 para gestionar el acceso y los permisos de usuarios.
Plugins y personalización	Permite el uso de plugins oficiales y de la comunidad para ampliar capacidades.
Casos de uso principales	-Monitoreo de aplicaciones, Observabilidad de microservicios, Visualización de datos industriales e IoT
Ventajas	- Visualización unificada de métricas, registros y trazas, reducción de carga operativa, escalabilidad automática
Consideraciones	-Costos por usuario activo, requiere configuración manual de dashboards, gestión responsable de permisos y roles

4.3.10. Tecnologías Inalámbricas para IoT

La Tabla 10 muestra un comparativo de las principales tecnologías de redes inalámbricas para IoT, destacando por su fácil accesibilidad, infraestructura y popularidad la tecnología **Wi-Fi**

Tabla 10. Comparativo de redes inalámbricas para IoT. Fuente: Elaboración propia

Tecnología	Rango de Cobertura	Consumo Energético	Velocidad de Transmisión	Topología	Casos de Uso Comunes	Ventajas Clave
Wi-Fi (802.11)	50–100 metros (urbano)	Alto (excepto 802.11ah)	Alta (hasta 600 Mbps o más)	Estrella / Infraestructura	Hogares inteligentes, oficinas, edificios conectados	Alta velocidad, infraestructura ampliamente disponible
Zigbee	10–100 metros	Bajo	Media-baja (hasta 250 Kbps)	Malla (Mesh)	Domótica, iluminación, sensores ambientales	Bajo consumo, topología flexible, ideal para redes densas
LoRaWAN	2–15 km (rural)	Muy bajo	Muy baja (0.3–50 Kbps)	Estrella	Agricultura, ciudades inteligentes, monitoreo remoto	Largo alcance, muy bajo consumo, buena penetración en interiores

4.3.10.1. Wi-Fi (802.11) y el ecosistema Internet de las Cosas (IoT)

En el contexto del Internet de las Cosas (IoT), la conectividad inalámbrica juega un papel central en la transmisión de datos entre dispositivos inteligentes. Dentro de este panorama, las redes **Wi-Fi**, basadas en el estándar IEEE 802.11, se han consolidado como una opción accesible y ampliamente adoptada, especialmente en entornos domésticos y urbanos. Su uso en IoT se debe, en gran parte, a su infraestructura ya establecida, su amplio soporte en dispositivos comerciales y su capacidad para ofrecer velocidades de transmisión elevadas en distancias relativamente cortas (Zeadally et al., 2020).

Aunque no fueron diseñadas específicamente para dispositivos de baja potencia, las versiones más recientes del estándar, como 802.11ah (*también conocida como Wi-Fi HaLow*), han sido optimizadas para escenarios IoT, ofreciendo un menor consumo energético y un mayor alcance en comparación con sus predecesores (Farhan et al., 2021). Este tipo de redes resulta especialmente útil en aplicaciones como el monitoreo ambiental, la automatización del hogar y la gestión de edificios inteligentes, donde se requiere conectividad robusta y la integración con redes IP ya existentes.

Sin embargo, es importante considerar que el uso de **Wi-Fi** en IoT presenta limitaciones en términos de consumo energético y congestión en bandas de frecuencia compartidas. Por ello, su adopción debe evaluarse cuidadosamente en función del tipo de dispositivo, la cantidad de nodos conectados y los requerimientos de calidad de servicio.

4.3.10.2. ZigBee

ZigBee es un protocolo basado en el estándar IEEE 802.15.4, que se utiliza principalmente para redes de área personal (PAN) de bajo consumo. Este protocolo utiliza una topología de red en malla, lo que significa que los dispositivos pueden retransmitir señales a otros dispositivos, ampliando el alcance de la red. **ZigBee** es muy eficiente en términos de consumo energético, lo que lo convierte en una opción popular para aplicaciones de automatización del hogar y control industrial.

A pesar de su eficiencia energética, **ZigBee** tiene limitaciones en términos de alcance, ya que generalmente funciona mejor en distancias cortas y requiere un buen diseño de red para garantizar una cobertura adecuada (Mohan & Banakar, 2020).

4.3.10.3. LoRaWAN (Long Range Wide Area Network)

LoRaWAN (Red de Área Amplia de Largo Alcance) es un protocolo diseñado para comunicaciones a larga distancia, ideal para aplicaciones que requieren la transmisión de

pequeños paquetes de datos en áreas geográficas amplias. **LoRaWAN** opera sobre una arquitectura de red en estrella y es extremadamente eficiente en términos de consumo energético, lo que permite que los dispositivos conectados funcionen durante largos períodos sin necesidad de recarga.

El principal beneficio de **LoRaWAN** es su capacidad para cubrir grandes distancias (hasta 15 kilómetros en áreas rurales) mientras mantiene un consumo de energía bajo. Sin embargo, **LoRaWAN** está limitado en términos de velocidad de transmisión y volumen de datos, lo que lo hace menos adecuado para aplicaciones que requieren alta capacidad de transferencia de datos (Jouhari et al., 2022).

4.3.11. El IDE de Arduino y su integración en plataformas IoT

El entorno de desarrollo integrado (**IDE**) de **Arduino** es considerado ampliamente como una herramienta fundamental en el desarrollo de plataformas de Internet de las Cosas (IoT), destacándose por su facilidad de uso y su capacidad de integración con módulos como el ESP8266 mediante el lenguaje de programación C++.

El **IDE de Arduino** es una aplicación multiplataforma que permite escribir y cargar programas en placas compatibles con Arduino. Su interfaz sencilla y su estructura basada en funciones básicas facilitan el aprendizaje y la implementación de proyectos, incluso para aquellos sin experiencia previa en programación. Además, su naturaleza de código abierto permite a los desarrolladores extender su funcionalidad y adaptarla a diversas necesidades (Kavedia et al., 2020).

La integración del ESP8266 con el **IDE de Arduino** ha ampliado las posibilidades en el desarrollo de soluciones IoT. Su compatibilidad con el **IDE de Arduino** permite programarlo utilizando el lenguaje C++, lo que simplifica el proceso de desarrollo a costos accesibles (Mittapally & Karra, 2023).

El uso combinado del **IDE de Arduino** y el ESP8266 ha demostrado ser eficaz en aplicaciones como el monitoreo de salud de pacientes en tiempo real, y transmitirlos a una plataforma IoT para su análisis (Brown Macheso & Meela, 2021).

La combinación del **IDE de Arduino** y el ESP8266, junto con el lenguaje de programación C++, ofrece una solución accesible y eficiente para el desarrollo de aplicaciones IoT, facilitando la creación de prototipos y la implementación de sistemas en diversos campos.

4.4. Diseño del sistema

La fase de diseño en el desarrollo de soluciones basadas en Internet de las Cosas (IoT) representa un momento clave donde se define no solo la funcionalidad técnica del sistema, sino también su viabilidad, escalabilidad y sostenibilidad a largo plazo. En esta etapa, se conjuntan cuidadosamente los dispositivos, protocolos de comunicación, plataformas de integración y esquemas de seguridad, tomando en cuenta las necesidades reales del entorno donde se implementará la solución.

Elaborar el diseño con visión clara permite anticipar desafíos, optimizar recursos y garantizar que el sistema responda eficazmente a las condiciones del mundo físico. Además, en contextos sensibles como la salud, un diseño bien fundamentado puede marcar la diferencia entre una solución útil y una falla operativa crítica. Por ello, más allá de los componentes tecnológicos, el diseño debe ser un proceso reflexivo, centrado en las personas y adaptado a los contextos específicos donde se desplegará.

4.4.1. Arquitectura utilizada para el desarrollo del prototipo

La arquitectura del Internet de las Cosas (IoT) se organiza en cuatro capas principales: ***dispositivo, red, servicios y aplicación.***

- ***La capa de dispositivo*** incluye sensores y actuadores que recopilan datos del entorno físico.
- ***La capa de red*** permite la transmisión eficiente y segura de datos mediante tecnologías como Wi-Fi y Zigbee (Serpanos & Wolf, 2018; Sethi & Sarangi, 2017).
- ***La capa de servicios*** gestiona funciones esenciales como el procesamiento en tiempo real y el almacenamiento, permitiendo una arquitectura flexible y reutilizable (Al-Fuqaha et al., 2015).
- ***La capa de aplicación*** ofrece soluciones adaptadas al usuario final, destacando en áreas como la salud, donde facilita el monitoreo remoto y la gestión de datos médicos (Islam et al., 2015).

Esta estructura por capas mejora la eficiencia, escalabilidad y capacidad de adaptación de los sistemas IoT a diferentes contextos.

4.4.2. Diagrama del prototipo (Arquitectura de capas IoT)

El proyecto principalmente se define dentro de las capas de arquitectura IoT que a continuación se refieren:

La capa de aplicación: está compuesta por la interfaz de usuario, para la monitorización, entrega y análisis de la información.

La capa de servicios y apoyo: aquí se encuentran los servicios en la nube, con el apoyo también de la capa de sesión que asegura la conexión entre los distintos puntos.

La capa de red: está determinada por la infraestructura de comunicaciones inalámbrica (WI-FI), hacia el transceptor (ESP8266), apoyados por los protocolos de comunicaciones para el envío y recepción de datos.

La capa de dispositivo: está conformada por el sistema embebido a manera de nodo por el microcontrolador y los sensores.

La Figura 12 muestra las cuatro capas de diseño de este prototipo:



Figura 12. Diagrama de arquitectura en capas del prototipo. Fuente: Elaboración Propia.

4.4.3. Elementos utilizados para la transferencia de datos a la Nube

Forma de conexión y comunicación de los dispositivos IoT:

- Modelo dispositivo a la nube, utilizando tecnología **WI-FI 802.11 b/g/n**.
- La comunicación del sistema embebido a la red es mediante **IP**, posteriormente a la Nube utilizando servicio de **pub/sub MQTT**.
- Intercambio de datos a través del backend: mediante los servicios en la nube.
- Permite que el usuario visualice el monitoreo a través de un dashboard.

A continuación, se muestra el modelo de comunicación del prototipo en la Figura 13.

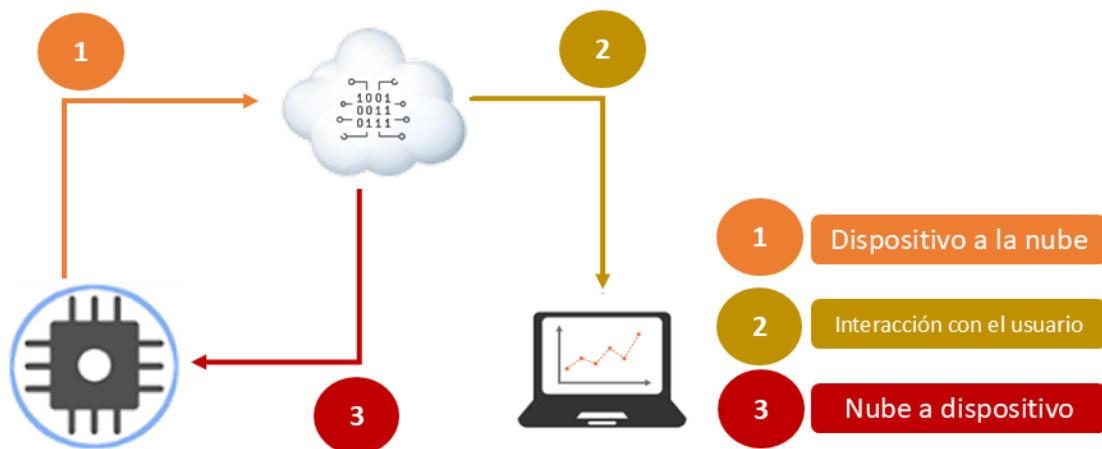


Figura 13. Estructura de comunicación. Fuente: Elaboración Propia.

4.4.4. Diagrama de sistema embebido

El sistema embebido para el desarrollo del proyecto IoT consta de los siguientes componentes:

- 1 microcontrolador ESP8266.
- 4 celdas de carga de 50 Kg conectadas en puente Wheatstone.
- 1 módulo amplificador HX711.
- 1 sensor de distancia LIDAR VL53L0X.

El esquema de conexión de los componentes del sistema es como se muestra en la Figura 14.

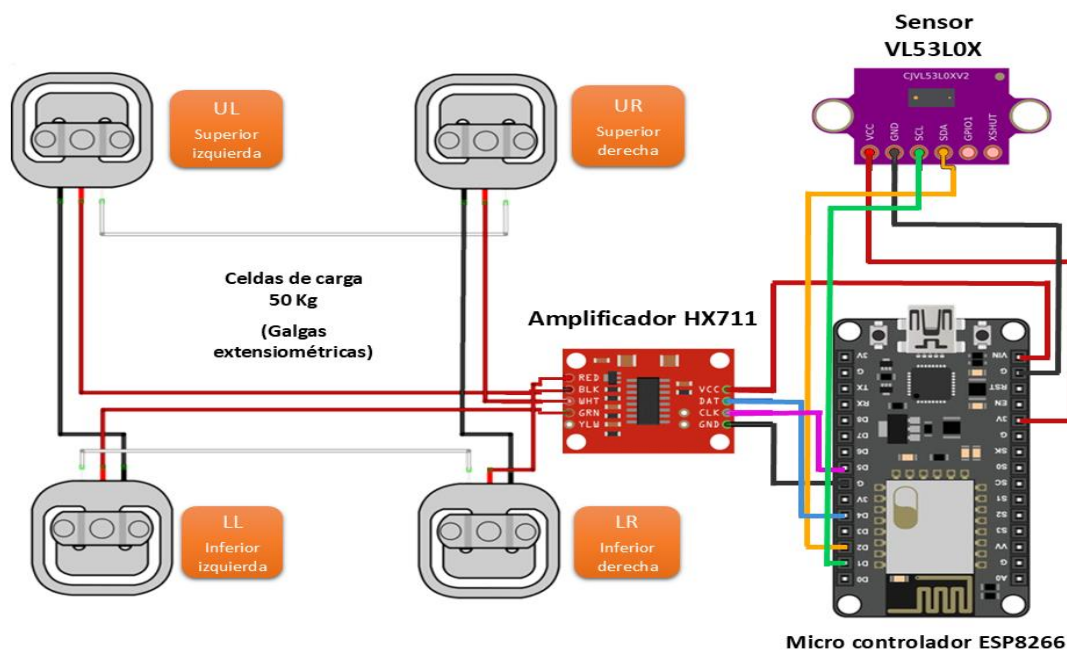


Figura 14. Diagrama de conexión de sistema embebido. Fuente: Elaboración Propia.

- **Conexión en puente Wheatstone de las celdas de carga con el módulo amplificador HX711**

Las celdas de carga se conectan en conjunto a través de una configuración llamada **puente Wheatstone**. El **puente de Wheatstone** es un circuito eléctrico que sirve para medir con mucha precisión cambios muy pequeños en la resistencia. Es muy útil cuando se conectan sensores, como los que detectan presión o fuerza, porque permite notar incluso las variaciones más mínimas. (Mohamad et al., 2020).

La Figura 15 muestra el esquema de conexión de las 4 celdas de carga para el prototipo.

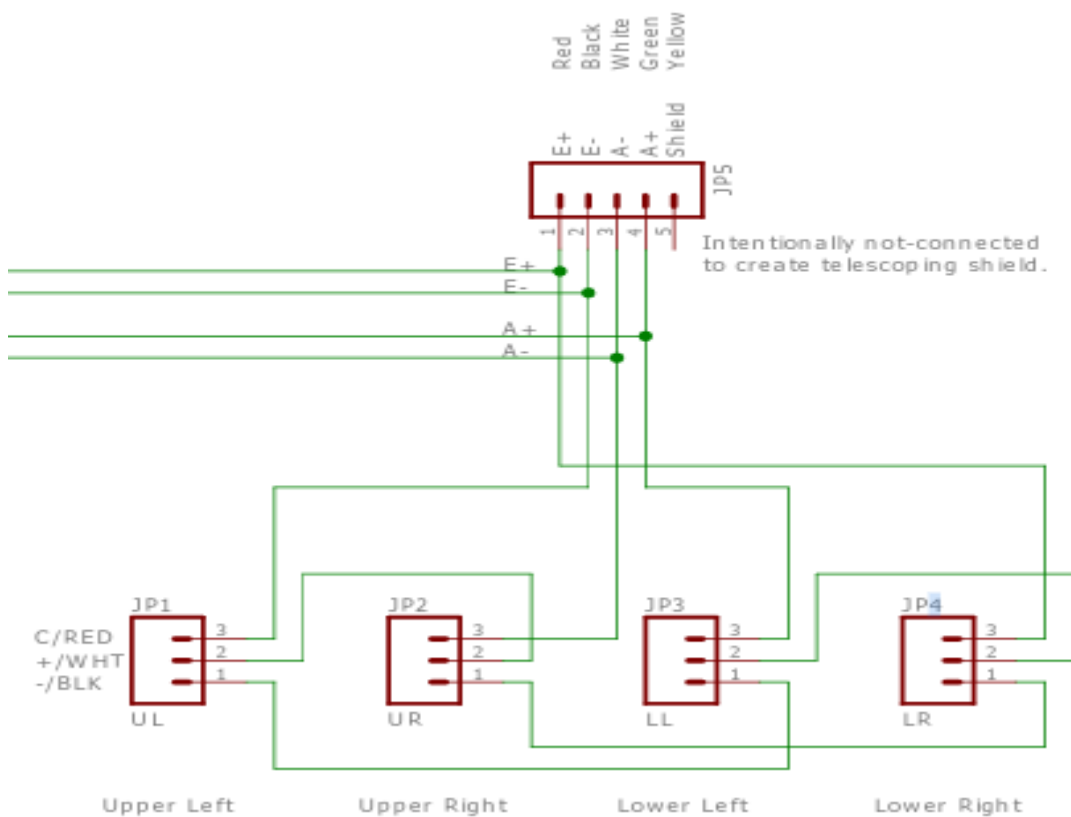


Figura 15. Diagrama de conexión celdas de carga tomada de SparkFun Electronics (2022).

Las celdas de carga cuentan con 3 cables para su conexión: **C/Red (rojo)**, **+/Wht (blanco)**, **-/Blk (negro)**.

De acuerdo con el diagrama de la Figura 15, el cableado **rojo C/Red** debe conectarse como se muestra en la Tabla 11.

Tabla 11. Conexión de cableado rojo C/Red. Fuente: Elaboración Propia.

Celda de carga	Pin de HX711
UL Superior izquierda	E - (Black)
UR Superior derecha	A - (White)
LL Inferior izquierda	A + (Green)
LR Inferior derecha	E + (Red)

El cableado **blanco +/Wht** debe conectarse de la forma como lo indica la Tabla 12.

Tabla 12. Conexión de cableado blanco +/Wht. Fuente: Elaboración Propia.

Celda de carga	Celda de carga
UL Superior izquierda	UR Superior derecha
LL Inferior izquierda	LR Inferior derecha

El cableado **negro -/Blk** será conectado como se indica en la Tabla 13.

Tabla 13. Conexión de cableado negro -/Blk. Fuente: Elaboración Propia.

Celda de carga	Celda de carga
UL Superior izquierda	LL Inferior izquierda
UR Superior derecha	LR Inferior derecha

- **Conexión del micro controlador ESP8266 con amplificador HX711**

La Tabla 14 muestra la configuración de conexión física del **ESP8266** con el **HX711**.

Tabla 14. Conexión ESP8266 - HX711. Fuente: Elaboración Propia.

Pin de ESP8266	Pin de HX711
GND	GND
VIN 5V	VCC
D4	DAT
D5	CLK

- **Conexión del micro controlador ESP8266 con sensor VL53L0X**

La Tabla 15 indica la configuración de conexión física del **ESP8266** con el **VL53L0X**.

Tabla 15. Conexión ESP8266 - VL53L0X. Fuente: Elaboración Propia.

Pin de ESP8266	Pin de VL53L0X
GND	GND
3V	VCC
D1	SCL
D2	SDA

4.4.5. Diagrama a bloques del prototipo

La Figura 16, presenta un diagrama a bloques que resume la arquitectura del sistema IoT, destacando la interacción entre sensores, unidad de procesamiento, red de comunicación y plataforma en la nube para el análisis de datos.

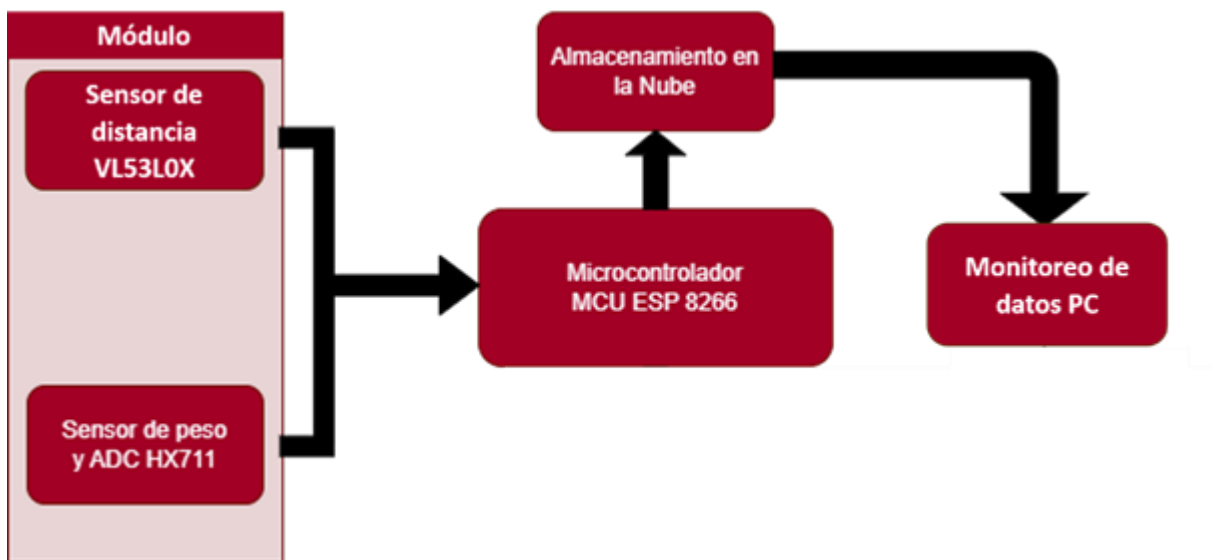


Figura 16. Diagrama a bloques de la solución. Fuente: Elaboración Propia.

4.4.6. Diagrama de proceso

La Figura 17 muestra el diagrama de proceso aplicado al prototipo IoT, detallando el flujo desde la solicitud de servicio por el usuario, la recolección de datos biométricos por sensores, su transmisión segura, el procesamiento en la nube y el uso de la información para el monitoreo y toma de decisiones clínicas.

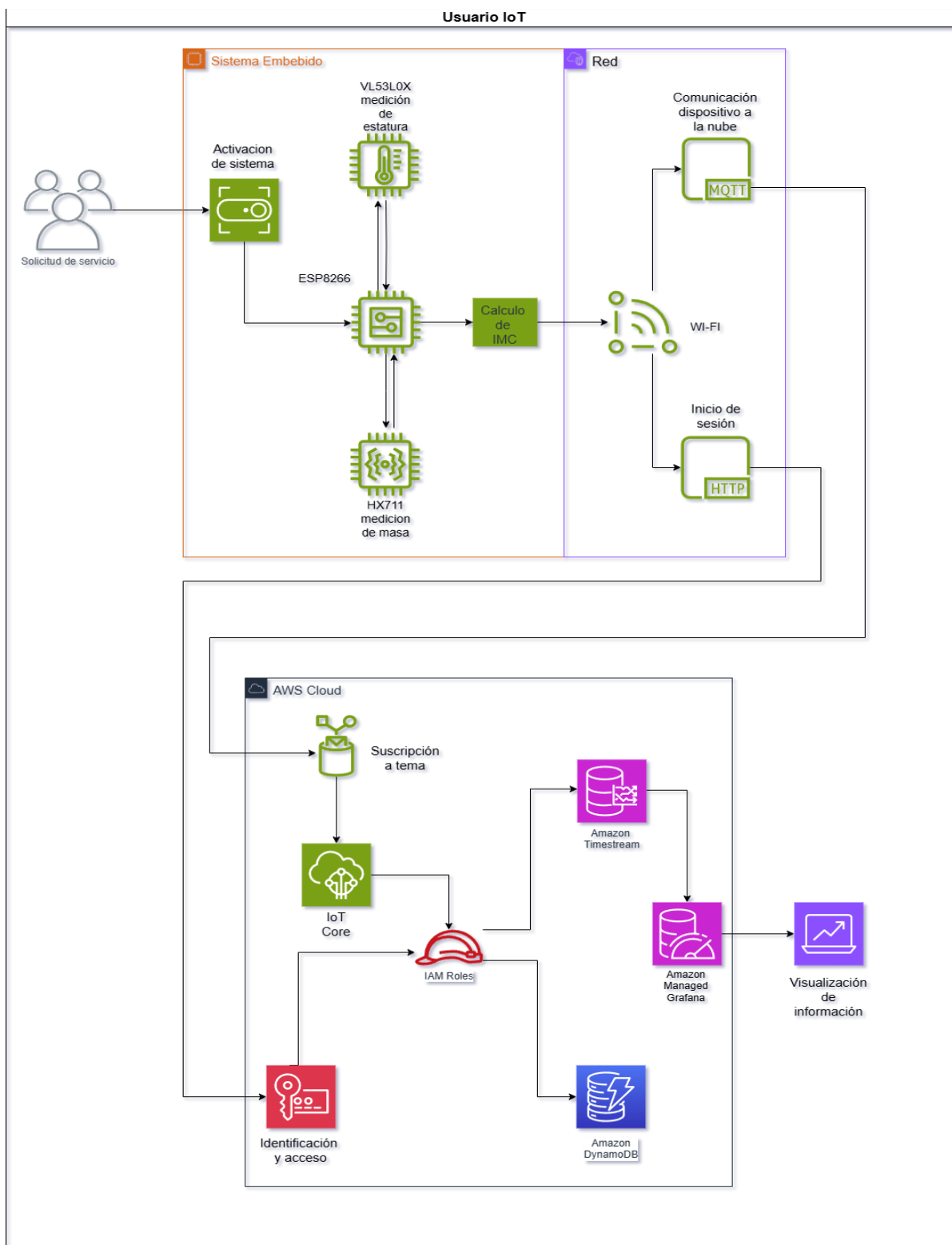


Figura 17. Diagrama del proceso de la solución. Fuente: Elaboración Propia.

4.4.7. Descripción general del proceso

- 1) El usuario realiza la solicitud de servicio.
- 2) Mediante interruptor se realiza activación de sistema.
- 3) El sistema embebido realiza la medición de estatura a través del sensor VL53L0X y de la masa del usuario mediante el sensor HX711.
- 4) El microcontrolador realiza el cálculo del IMC del usuario.
- 5) El microcontrolador conectado a la red WI-FI establece comunicación con el servicio de procesamiento y almacenamiento en la nube de AWS (Amazon Web Services) con apoyo de los protocolos HTTP y MQTT.
- 6) El protocolo HTTP realiza la conexión segura mediante identificación de credenciales, así como llaves de acceso para iniciar y mantener una sesión que asegure la integridad de los datos durante el envío y recepción.
- 7) El protocolo MQTT realiza conexión a la nube mediante el servicio de publicación y suscripción al tema donde se realizará el envío de los datos captados por el sistema embebido.
- 8) El broker IoT Core de AWS realiza la conexión al sistema embebido mediante suscripción al tema donde se publicarán los datos.
- 9) IAM Role servicio de identificación a través de roles de usuario realiza la administración del paso e ingreso de manera segura a las bases de datos no relacionales de AWS.
- 10) Amazon DynamoDB realiza el almacenamiento permanentemente los datos enviados del sistema embebido a la nube para su posterior análisis.
- 11) Amazon Timestream realiza el almacenamiento temporal de los datos publicados en la nube mediante IoT Core para su análisis y entrega en tiempo real.
- 12) Amazon Managed Grafana muestra finalmente los datos captados y calculados por el sistema embebido en tiempo real, mediante una interfaz gráfica de usuario llamada dashboard o tablero.

4.5. Construcción del sistema

La fase de construcción en sistemas IoT no es solo una tarea técnica, sino una labor de articulación entre hardware, software y humano. Es el punto donde la tecnología deja de ser una idea para convertirse en una presencia palpable y significativa. (Ghosh et al., 2022).

Esta etapa requiere una interrelación estrecha entre disciplinas como la electrónica, el desarrollo de software y la gestión de redes. Pero más allá del aspecto técnico, la construcción también implica una sensibilidad particular: comprender cómo las personas interactuarán con los dispositivos y prever su impacto en contextos reales. Así, el desarrollo no solo se mide en eficiencia funcional, sino también en su capacidad para integrarse de manera armónica en la vida cotidiana. (Ullah et al., 2024).

4.5.1. Integración del Hardware

El hardware del proyecto se conecta tal y como es referido en la sección **4.4.4 Diagrama de Sistema Embebido**. Para la conexión física de todos los dispositivos, celdas de carga 50 kg, sensores VL53L0X, HX711 y el microcontrolador ESP8266, se utiliza cable de cobre AWG calibre 24, con terminales o jumper Dupont tipo hembra, que facilitan la conexión de los sensores al microcontrolador. Alternativamente se utiliza cable UTP categoría 6 blindado que ofrece algunas ventajas sobre el uso de cable AWG como son: flexibilidad en diferentes entornos y facilidad de instalación, resultando una opción económica y versátil, sin perder funcionalidad.

4.5.2. Gabinetes de protección del sistema

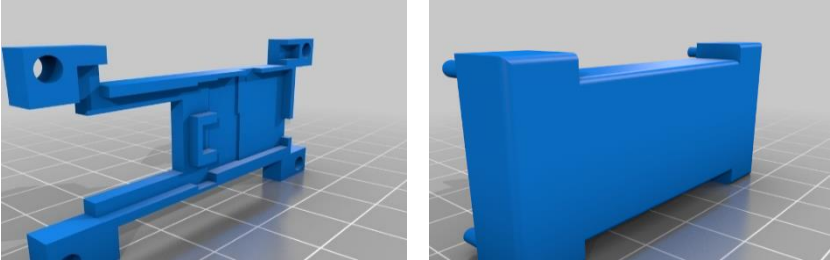
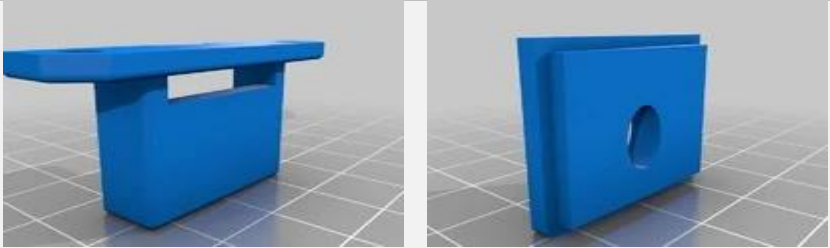
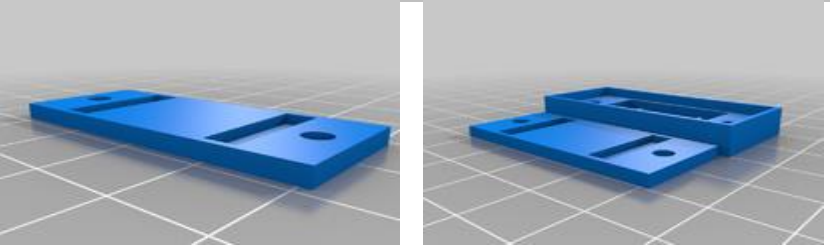
Para asegurar la protección física de los dispositivos IoT utilizados en el sistema embebido, se realizó la impresión 3D de los gabinetes para los sensores VL53L0X, HX711 y el microcontrolador ESP8266. Los modelos de los gabinetes se obtuvieron en el sitio web **Thingiverse**. Los modelos 3D disponibles en **Thingiverse**, están enfocados en carcasas para componentes electrónicos comunes en proyectos de IoT y electrónica, estos modelos son de **licencia Creative Commons (CC)**, con **atribución 4.0 Internacional no comercial**, lo que significa que pueden ser compartidos, adaptados y redistribuidos sin restricciones para fines no comerciales en el prototipado de proyectos IoT. El uso de licencias **Creative Commons** permite a los autores compartir sus obras bajo condiciones específicas de uso, promoviendo la reutilización legal de contenidos (Creative Commons, 2022).

Las Tablas 16 y 17 muestran las principales características de los diseños utilizados.

Tabla 16. Características de los gabinetes de los componentes IoT. Fuente: Elaboración Propia.

Diseño	Descripción	Compatibilidad y Montaje	Autor	Enlace
Carcasa para ESP8266 NodeMCU (CH340)	Carcasa protectora para el microcontrolador ESP8266 NodeMCU. Ofrece una estructura básica para uso en proyectos electrónicos.	Diseño sencillo; no se detallan sistemas de anclaje específicos.	Cavefish (2023)	Thing: 5873652
Caja para sensor Adafruit VL53L0X	Diseño adaptado para alojar el sensor de distancia VL53L0X de Adafruit. Requiere poca impresión y es fácilmente personalizable.	Incluye soporte de montaje, aunque sin detalles técnicos explícitos.	ajh1138 (2019)	Thing: 3434410
Caja para módulo HX711	Caja compacta para el módulo de celda de carga HX711. Ideal para proyectos estructurados sobre perfiles de aluminio.	Montaje específico en perfiles 2020; permite espacio interno para conexiones.	ProfHankD (2021)	Thing: 4975809

Tabla 17. Carcasas de componentes tomada de (Thingiverse, 2024).

ESP8266 Node MCU Case		Figura 18. Modelo de gabinete para impresión 3D del ESP8266.
Adafruit VL53L0X Case With Mount		Figura 19. Modelo de gabinete para impresión 3D del sensor VL53L0X.
HX711 Board Case		Figura 20. Modelo de gabinete para impresión 3D del sensor HX711.

4.5.3. Programación del microcontrolador

Para la correcta ejecución de los procesos de muestreo en el microcontrolador ESP8266, es sumamente importante la programación del mismo, la ejecución de las rutinas pertinentes como parte del software necesario en el sistema embebido, realizan en conjunto con el hardware una tarea específica, de manera dedicada o especializada.

El entorno de desarrollo integrado seleccionado es **Arduino IDE** debido a su facilidad y flexibilidad de uso.

A continuación, se muestran las rutinas de programación desarrolladas en el **IDE de Arduino**.

4.5.3.1. Librerías incluidas

En el entorno de desarrollo **Arduino IDE**, una **librería** (o *library*) es un conjunto de archivos de código que facilita el uso de componentes, sensores, módulos o funciones complejas sin tener que programarlas desde cero.

Es necesario incluir las librerías **ESP8266WiFi.h**, **HX711.h** y **Adafruit_VL53L0X.h** para la programación del prototipo. Cabe referir que el lenguaje de programación utilizado es **C++** y como editor se utilizó **Arduino IDE**. *Para visualizar el código, refiérase al Apéndice A.*

4.5.3.2. Calibración del HX711

Para el correcto funcionamiento del sensor HX711 se realiza previamente el cálculo del factor de calibración, este cálculo se lleva a cabo con un programa creado por *Nathan Seidle de SparkFun Electronics* y es de código libre y facilita la configuración precisa de celdas de carga mediante Arduino (SparkFun Electronics, 2023).

```
INICIO

SI hay datos disponibles en el puerto serial ENTONCES
    Leer un carácter y guardarlo en la variable temp

    SI temp es igual a '+' ENTONCES
        Aumentar el valor de factor_calibracion en 10
    SINO SI temp es igual a '-' ENTONCES
        Disminuir el valor de factor_calibracion en 10
    FIN SI

FIN SI

FIN
```

Figura 21. Pseudocódigo de calibración HX711. Fuente: Elaboración Propia.

Este programa debe iniciarse con la báscula sin peso alguno sobre la base de cristal templado para comenzar la calibración. Posteriormente debe aplicarse un peso conocido, preferentemente uno a dos kilogramos, con las teclas '+' y '-' como se muestra en la Figura 21. Se puede aumentar o disminuir el factor de calibración según sea el caso hasta igualar el peso conocido del objeto en la báscula como se muestra en las líneas 68 – 73 del código para la calibración. **Para visualizar el código completo, refiérase al Apéndice B.**

4.5.3.3. Configuración de servicios de red y protocolo de comunicación

Debe indicarse en el programa el nombre público de la red WI-Fi SSID y la contraseña de acceso. Para establecer comunicación con el broker MQTT es necesario indicar la dirección del mismo, la dirección proporcionada corresponde al servicio de AWS IoT Core. Además, como se indica en la Figura 22 es indispensable indicar el puerto de comunicación que es el **8883**, y el nombre de identificación del cliente conectado a través de la red WI-Fi.

```
// Definir el tema MQTT a utilizar (por ejemplo: "esp8266/datos")
Definir mqtt_topic = "esp8266/datos"

// Crear clientes para WiFi y MQTT
Crear cliente WiFi seguro (espClient)
Crear cliente MQTT usando:
    - dirección del servidor AWS
    - puerto 8883 (estándar para MQTT seguro)
    - función callback
    - cliente WiFi seguro

FIN
```

Figura 22. Pseudocódigo Configuración de servicios de red. Fuente: Elaboración Propia.

4.5.3.4. Función de conexión de red WI-Fi

Este bloque de programa realiza la rutina que conecta el microcontrolador a la red WI-Fi, como se muestra en la Figura 23 y es llamada siempre que se inicia el sistema embebido.

```

Esperar 10 milisegundos

Establecer tamaño de búfer para el cliente seguro (512, 512)

Imprimir una línea vacía en el monitor serial

Imprimir "Conectando a WiFi..." en el monitor serial

Imprimir el nombre de la red WiFi (SSID)

Iniciar conexión WiFi usando el SSID y la contraseña

Mientras el estado del WiFi no sea "conectado":
    Esperar 500 milisegundos
    Imprimir un punto (".") en el monitor serial

```

Figura 23. Pseudocódigo de conexión de red WI-FI. Fuente: Elaboración Propia.

4.5.3.5. Reconexión del broker MQTT en caso de pérdida de comunicación

La rutina que se muestra en la Figura 24, asegura la reconexión del broker en caso de algún fallo de comunicación para el envío de mensajes al servicio de AWS IoT Core. En caso de fallo o desconexión, realiza un nuevo intento después de 5 segundos.

```

Si el cliente logra conectarse con el identificador "ESP8266_thing":
    Imprimir "Conectado al broker MQTT"
    Publicar el mensaje "hola" en el tema mqtt_topic
    Suscribirse al tema "inTopic"

Sino:
    Imprimir "Error, rc=" seguido del estado del cliente
    Imprimir "Intentando de nuevo en 5 segundos..."

    Crear un búfer de 256 caracteres
    Obtener el último error SSL del cliente WiFi seguro y almacenarlo en el búfer
    Imprimir "WiFiClientSecure SSL error: " seguido del contenido del búfer

```

Figura 24. Pseudocódigo de reconexión del broker. Fuente: Elaboración Propia.

4.5.3.6. Carga de certificados. DER de AWS

Los certificados en formato **.DER** de AWS (Amazon Web Services), son archivos digitales codificados en formato binario (***Distinguished Encoding Rules***). que contienen certificados **X.509**. Se utilizan principalmente en sistemas que requieren comunicación segura a través de

SSL/TLS como el servicio de AWS IoT Core, y así validar la identidad de los dispositivos y servicios de manera segura, además de asegurar que los datos transmitidos a la nube de AWS, estén protegidos mediante el cifrado en la comunicación. Estos archivos son los componentes necesarios para establecer una conexión segura (*HTTPS*) entre un cliente y un servidor en AWS (Amazon Web Services, 2023).

Para el desarrollo del proyecto, adjunto al archivo de código fuente en Arduino, se debe cargar al microcontrolador una carpeta llamada '**data**', que contiene los archivos en formato **.der** como se observa en la Figura 25, Figura 26 y son:

Ca.der → Certificate Authority (**Autoridad de Certificación**), verifica la identidad de un servidor o cliente y emite certificados digitales.

Cert.der → Certificate (**Certificado del Servidor/Cliente**), es el certificado digital asociado a un dominio, utilizado para establecer una conexión segura (*HTTPS*) con el servidor o cliente.

Private.der → Llave privada asociada al certificado, de suma importancia para la seguridad, no debe compartirse con terceros y se utiliza para descifrar la información y autenticar la identidad del servidor o cliente.

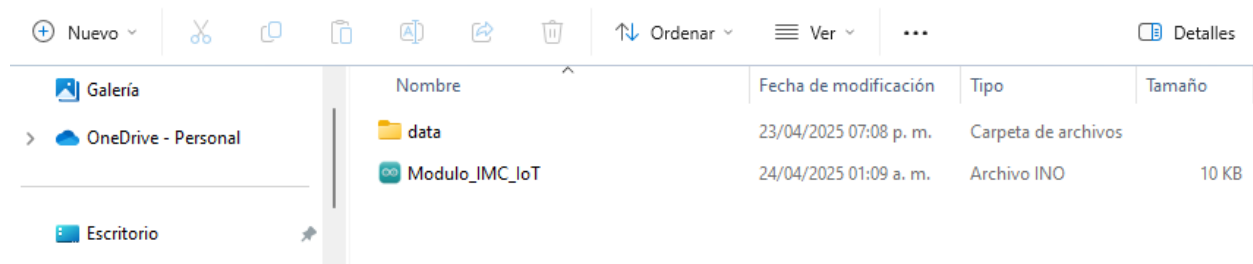


Figura 25. Carpeta "data" adjunta al código fuente. Fuente: Elaboración Propia.

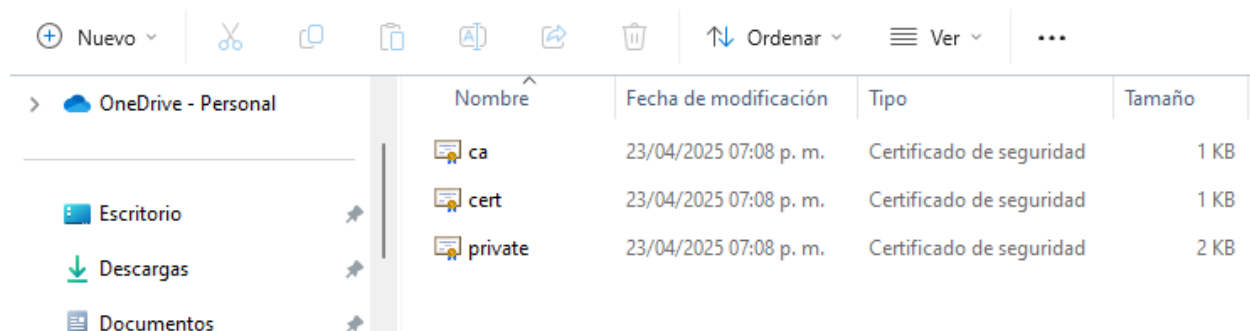


Figura 26. Archivos de certificados AWS en formato .der. Fuente: Elaboración Propia.

La carpeta '**data**' debe ser cargada al ESP8266, para ello es imprescindible instalar la librería **FS.h** que contiene objetos y funciones para la gestión de ficheros en **SPIFFS**. **SPIFFS (SPI Flash File System)** es un sistema de archivos utilizado en memorias flash conectadas por **SPI** en dispositivos embebidos y con escasa cantidad de RAM, como el ESP8266. (Espressif Systems, 2024).

Cuando la librería **SPIFFS FS.h** está debidamente instalada en el IDE de Arduino, en el menú de herramientas deben aparecer los comandos de carga **LittleFS Data Upload** y **Sketch Data Upload** como se muestra en la Figura 27.

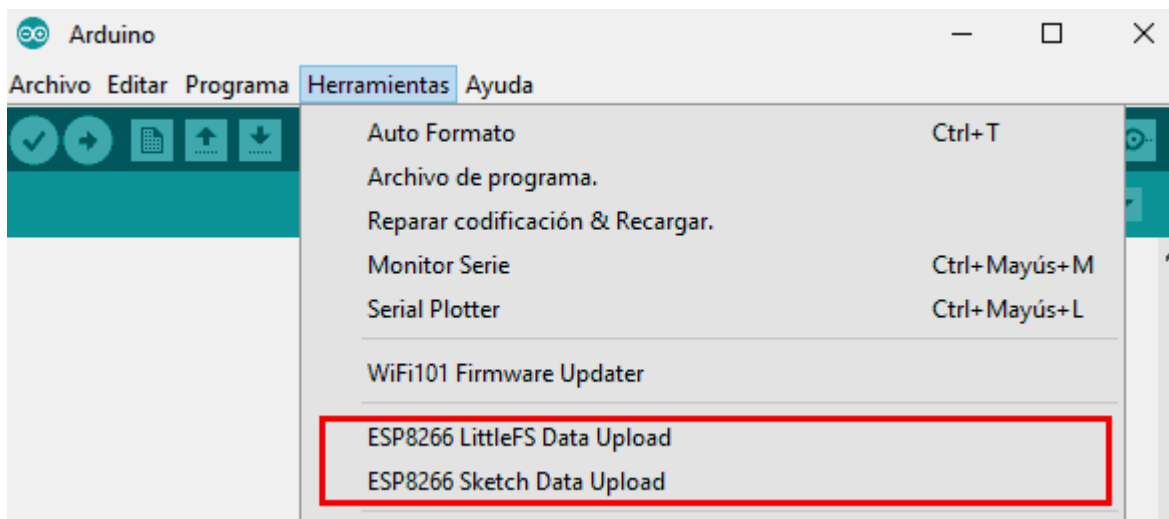


Figura 27. Lista de herramientas con la librería FS.h instalada en el IDE de Arduino (2025).

Una vez que se ejecuta el comando '**ESP8266 Sketch Data Upload**', todos los archivos que se encuentren dentro de la carpeta '**data**' se cargarán directamente en la memoria **SPIFFS** del dispositivo. Es importante tener en cuenta que este proceso sobrescribirá cualquier información previa almacenada en este espacio.

4.5.3.7. Función de interrupción para activar los sensores

Dado que el dispositivo solo realiza la medición de datos bajo solicitud del usuario, el sistema embebido permanecerá en estado de espera hasta que el usuario haga dicha petición, para ello la rutina correspondiente llamada mediante una función (Figura 28), permite el control de interrupción mediante un pulsador, que activa los sensores del sistema para realizar la tarea.

```

Declarar una variable booleana "sensorActivo" como falsa (el sensor inicia desactivado)

Procedimiento activarSensor:
    Establecer sensorActivo como verdadero (activar el sensor cuando se presione el botón)

Procedimiento interruptLoad:
    Configurar el pin del botón como entrada con resistencia pull-up

    Asignar una interrupción al pin del botón:
        - Activar la función activarSensor cuando ocurra un flanco de bajada (presión del botón)

    Imprimir "Sistema iniciado. Sensor inactivo." en el monitor serial

```

Figura 28. Pseudocódigo activación de sensores. Fuente: Elaboración Propia.

4.5.3.8. Función para inicializar el sensor VL53L0X y su operación

La rutina que a continuación se presenta, inicializa el sensor de distancia que calcula la estatura del usuario en milímetros para posteriormente convertirla a metros, una vez que se activa el sistema, la función es llamada para realizar la medición y el cálculo de esta variable. La función **loadVL53L0X()** que se muestra en la Figura 29 activa el sensor, además realiza la medición y los cálculos correspondientes.

```

Procedimiento loadVL53L0X:
    Mientras la comunicación Serial no esté lista:
        Esperar 1 milisegundo

    Imprimir "Adafruit VL53L0X prueba" en el monitor serial

    Si el sensor VL53L0X no se inicializa correctamente:
        Imprimir "falla en la carga del sensor VL53L0X"
        Detener el programa en un bucle infinito

```

Figura 29. Pseudocódigo que inicializa el sensor VL53L0X. Fuente: Elaboración Propia.

4.5.3.9. Funciones para inicializar el sensor HX711 y su operación

El HX711 es un amplificador **ADC**, esto significa que la medición de la masa en las celdas de carga se realiza de manera analógica y posteriormente es convertido a formato digital, la función **loadHX711()** (Figura 30), activa el **ADC (Convertidor analógico a digital)** del HX711, y la función **masaHX711()** en la Figura 31, realiza el cálculo de la masa del usuario en kilogramos,

para que esto pueda operar de manera correcta es necesario el factor de calibración obtenido previamente.

```
Procedimiento loadHX711:

    Si está definida la constante DEBUG_HX711 Entonces
        Imprimir en Serial: "[HX7] Inicio del sensor HX711"
    Fin Si

    Iniciar la báscula con los pines LOADCELL_DOUT_PIN y LOADCELL_SCK_PIN

    Configurar la escala de la báscula con el valor CALIBRACION

    Realizar la tara (poner a cero la báscula)

    Imprimir línea en blanco en Serial
```

Figura 30. Pseudocódigo que inicializa el sensor HX711 Fuente: Elaboración Propia.

```
Procedimiento masaHX711:

    Si está activada la depuración de HX711 (DEBUG_HX711) Entonces
        Imprimir en Serial: "[HX7] Leyendo: "

        Leer el peso desde la báscula (con 2 decimales) y mostrarlo en Serial

        Asignar a la variable imcscale el valor leído desde la báscula

        Imprimir en Serial: " Kg"
        Imprimir salto de línea en Serial
    Fin Si

Fin Procedimiento
```

Figura 31. Pseudocódigo que calcula la masa. Fuente: Elaboración Propia.

4.5.3.10. Publicación del mensaje a tópico MQTT en formato JSON

La función ***publishMessage()***, almacena las variables globales en cadenas de caracteres y números flotantes según sea el tipo de dato, para posteriormente ser convertidos y enviados en formato ***JSON*** al tema de publicación del broker MQTT llamado ***“esp8266/datos”***. Un objeto ***JSON (JavaScript Object Notation)*** (Figura 32), es un formato de intercambio de datos ligero muy útil en el envío y almacenamiento de datos de manera estructurada entre un servidor y una aplicación o diferentes partes de un programa, además de su multicompatibilidad con diversos lenguajes de programación y su uso extenso en proyectos de IoT.

```

Convertir el identificador del chip (chipid) a una cadena llamada espIdStr
Convertir la dirección MAC (mac_Id) a una cadena llamada macIdStr

Asignar el valor de la estatura a la variable estaturaMtr
Asignar el valor del peso (masa corporal) a la variable masaKg
Asignar el valor del índice de masa corporal a la variable imcKgm

Crear un mensaje en formato JSON con los siguientes datos:
- chip_Id: espIdStr
- mac_Id: macIdStr
- Estatura en metros (estaturaMtr, con 2 decimales)
- Masa en kilogramos (masaKg, con 2 decimales)
- IMC en kg/m² (imcKgm, con 2 decimales)
El mensaje se guarda en la variable `msg` con formato controlado (snprintf)

```

Figura 32. Pseudocódigo publicación de mensaje MQTT. Fuente: Elaboración Propia.

4.5.3.11. Función Setup

En la Figura 33 la función **setup()** en el sketch o programa del IDE de Arduino, se ejecuta solo una vez al iniciar el sistema, sirve para definir configuraciones iniciales como los puertos de entrada, la velocidad del puerto serie, las variables y librerías de inicio y los sensores o módulos conectados. En la ejecución de la función **setup()** del presente proyecto, además de llamar a las funciones de configuración inicial de los sensores VL53L0X y HX711, se monta también el sistema de archivos **SPIFFS (SPI Flash File System)**, para permitir la lectura de los certificados de AWS almacenados en la memoria interna del microcontrolador ESP8266.

La carga de archivos y ciertas operaciones, así como el uso de diversas funciones y **bibliotecas como WiFi**, pueden hacer uso del **heap**, el cual es una región de la memoria RAM del microcontrolador que es útil para la asignación dinámica de memoria durante la ejecución del programa, pero su uso irracional, puede comprometer la estabilidad del sistema embebido (Barrett & Pack, 2020). En el setup la función **getFreeHeap()**, que permite monitorear la memoria disponible del microcontrolador.

Finalmente se incluyen las funciones **getChipId()** y **macAddress()**, para obtener la **ID** y la dirección física única **MAC (Media Access Control)** del microcontrolador (Figura 34).


```

Iniciar comunicación serie a 115200 baudios

Llamar a la función interruptLoad() para configurar las interrupciones
Llamar a la función loadVL53L0X() para iniciar el sensor de distancia
Llamar a la función loadHX711() para inicializar el sensor de peso

Habilitar salida de depuración en el monitor serie

Llamar a la función setup_wifi() para conectarse a la red WiFi

```

Figura 33. Pseudocódigo función setup. Fuente: Elaboración Propia.

```

// Mostrar memoria disponible en el ESP8266
Mostrar en consola el texto "Heap: "
Mostrar en consola el valor de la memoria libre obtenida mediante ESP.getFreeHeap()

// Obtener y mostrar la ID del chip ESP8266
Mostrar en consola el texto "chipId: "
Obtener el chip ID del ESP8266 y guardarlo como texto en la variable chipid
Mostrar en consola el valor de chipid

// Obtener y mostrar la dirección MAC del ESP8266
Obtener la dirección MAC del ESP8266 y guardarla en el arreglo mac
Formatear la dirección MAC como cadena de texto en formato XX:XX:XX:XX:XX:XX
Guardar la dirección MAC formateada en la variable mac_Id
Mostrar en consola el valor de mac_Id

```

Figura 34. Pseudocódigo, Memoria, ID y MAC. Fuente: Elaboración Propia.

4.5.3.12. Función loop

Esta función se ejecuta de manera repetitiva en la ejecución del programa, bajo la condición de que el microcontrolador ESP8266 identificado como cliente, se encuentre conectado a la nube mediante infraestructura de red WI-FI, pues el servicio de IoT Core de AWS así lo demanda, por eso es significativo que la red WI-FI sea persistente (Espressif Systems, 2023). En la Figura 35 se observa que, si por alguna razón la conexión se pierde, la función **loop()**, llamará a la rutina **reconnect()** para recuperar la conexión. Toda vez que el cliente se encuentre conectado al

servidor de AWS, la función **loop()** verificará si existen solicitudes de activación de los sensores por parte del usuario, esta verificación la realiza mediante una variable de control llamada **sensorActivo**, la cual puede tener uno de dos posibles valores (**falso**, **verdadero**). Al comienzo de la ejecución de la función **loop()**, la variable **sensorActivo** está declarada como “**falso**”, si el usuario mediante interrupción activa los sensores, la variable **sensorActivo**, cambia su valor a “**verdadero**” e invocará las funciones **estaturaVL53L0X()** y **masaHX711()**, que realizan las mediciones de estatura y masa respectivamente, con ambos datos obtenidos de los sensores, se realiza el cálculo del **IMC (Índice de Masa Corporal)** dentro de la variable **indmascorp**. Los datos obtenidos y el valor calculado del **IMC**, son mostrados en el monitor serie del IDE de Arduino y posteriormente, son publicados como mensaje en formato JSON a través de la función **publishMessage()**, utilizando el protocolo MQTT al servicio de AWS IoT Core (Figura 36).

Finalmente, los sensores quedan deshabilitados, terminando el ciclo de la función **loop()** y en espera de una nueva solicitud de servicio por parte del usuario.

```
INICIO DE LOOP

SI el cliente MQTT no está conectado ENTONCES
    Llamar a la función reconnect()
FIN SI

Ejecutar client.loop()

Obtener el tiempo actual en milisegundos y guardarlo en 'now'

SI (now - lastMsg) es mayor o igual a 10000 ENTONCES
    lastMsg ← now

    SI sensorActivo ES VERDADERO ENTONCES
        Llamar a estaturaVL53L0X()    // Mide la estatura
        Llamar a masaHX711()          // Mide la masa corporal

        Calcular el IMC:
        indmascorp ← imcscale / (estatura * estatura)
```

Figura 35. Pseudocódigo función Loop. Fuente: Elaboración Propia.

```

Mostrar en consola:
    "Peso = ", imcscale
    "IMC = ", indmascorp

Llamar a publishMessage()      // Publicar mensaje MQTT

Mostrar en consola la memoria libre (heap)
Mostrar mensaje de evento detectado

sensorActivo ← FALSO          // Desactiva el sensado
SINO
    Mostrar mensaje: "Sensado inactivo. Presiona el botón para activarlo."
    Esperar 1000 milisegundos
FIN SI

```

Figura 36. Pseudocódigo publicación de mensaje. Fuente: Elaboración Propia.

4.5.3.13. Monitor serie

Al ejecutar el programa en el IDE de Arduino, mediante la ventana de monitor serie en la Figura 37, se puede visualizar la actividad del sistema embebido, desde la conexión con la red, asignación de dirección IP, Carga de certificados y llave privada de AWS, conexión con el broker MQTT, el status de los sensores (activo o inactivo) y la medida y cálculo de las variables, así como su envío y registro en el servicio de AWS como se muestra en la Figura 38.

```

COM5

..scandone
state: 0 -> 2 (b0)
.state: 2 -> 3 (0)
state: 3 -> 5 (10)
add 0
aid 5
cnt

connected with TELECABLE-BB93, channel 11
dhcp client start...
.ip:192.168.1.3,mask:255.255.255.0,gw:192.168.1.1
.

WiFi conectado!
IP asignada: 192.168.1.3
Heap: 35304
Success to open cert file
cert loaded
Success to open private cert file
private key loaded
Success to open ca
ca loaded
Heap: 31976
chipId: 6260859
ec:fa:bc:5f:88:7bIntentando conexión MQTT...Conectado al broker MQTT
Sensado inactivo. Presiona el botón para activarlo.
Sensado inactivo. Presiona el botón para activarlo.

```

Figura 37. Monitor serie IDE Arduino. (2025).

```

Sensado inactivo. Presiona el botón para activarlo.
Realizando medicion... Distancia en (mm): 28
Distancia en (mm): 28.00
Distancia en (m): 0.03
Estatura en (m): 1.97
[HX7] Leyendo: 98.45 Kg
Peso= 97.38
IMC = 25.04
Mensaje publicado: {"chip_Id" : "6260859", "mac_Id" : "ec:fa:bc:5f:88:7b", "Estatura en m" : "1.97", "Masa en kg" : "97.38", "IMC en kg/m2" : "25.04"}
Heap: 26536
¡Evento detectado! Sensado detenido.
Sensado inactivo. Presiona el botón para activarlo.

```

Figura 38. Medición y publicación de datos vista del monitor serie IDE Arduino (2025).

4.5.4. Amazon Web Services

Amazon Web Services (AWS) se ha convertido en una pieza clave de la infraestructura digital moderna. Desde su lanzamiento en 2006, ha pasado de ser una simple solución de almacenamiento y computación a ofrecer más de 200 servicios en la nube, lo que ha permitido a empresas de todos los tamaños acceder a tecnologías antes impensables (Amazon Web Services, 2023).

Su éxito se basa en la escalabilidad, la seguridad y la confiabilidad. Compañías como Netflix o la NASA lo usan precisamente por su capacidad de adaptarse a cualquier carga de trabajo sin interrupciones (Miller, 2022). Además, el modelo de pago por uso ha hecho que muchas organizaciones migren a AWS buscando optimizar costos.

Sin embargo, no todo es positivo. Algunos analistas advierten sobre la fuerte dependencia tecnológica que puede generar AWS y los retos que implica gestionar su compleja estructura de precios (Smith, 2021).

AWS no solo ha revolucionado el mundo tecnológico, sino que también ha cambiado la forma en que las organizaciones piensan sus operaciones digitales. Comprender su impacto es esencial para cualquier profesional de las tecnologías emergentes en el presente.

4.5.4.1. Consola de AWS

La consola de Amazon Web Services (AWS) que se observa en la Figura 39, es una interfaz gráfica basada en la web que permite a los usuarios interactuar de forma intuitiva con los distintos servicios que ofrece la plataforma. A través de ella, es posible lanzar servidores, configurar bases de datos, gestionar redes o monitorear el rendimiento de aplicaciones, todo desde un único lugar y sin necesidad de utilizar líneas de comando.

Esta herramienta ha sido clave en la democratización del acceso a la nube, ya que permite a usuarios técnicos y no técnicos explorar y utilizar servicios complejos de manera accesible. Su diseño centrado en la experiencia del usuario facilita la curva de aprendizaje, convirtiéndola en una puerta de entrada al vasto ecosistema de AWS (Amazon Web Services, 2023).

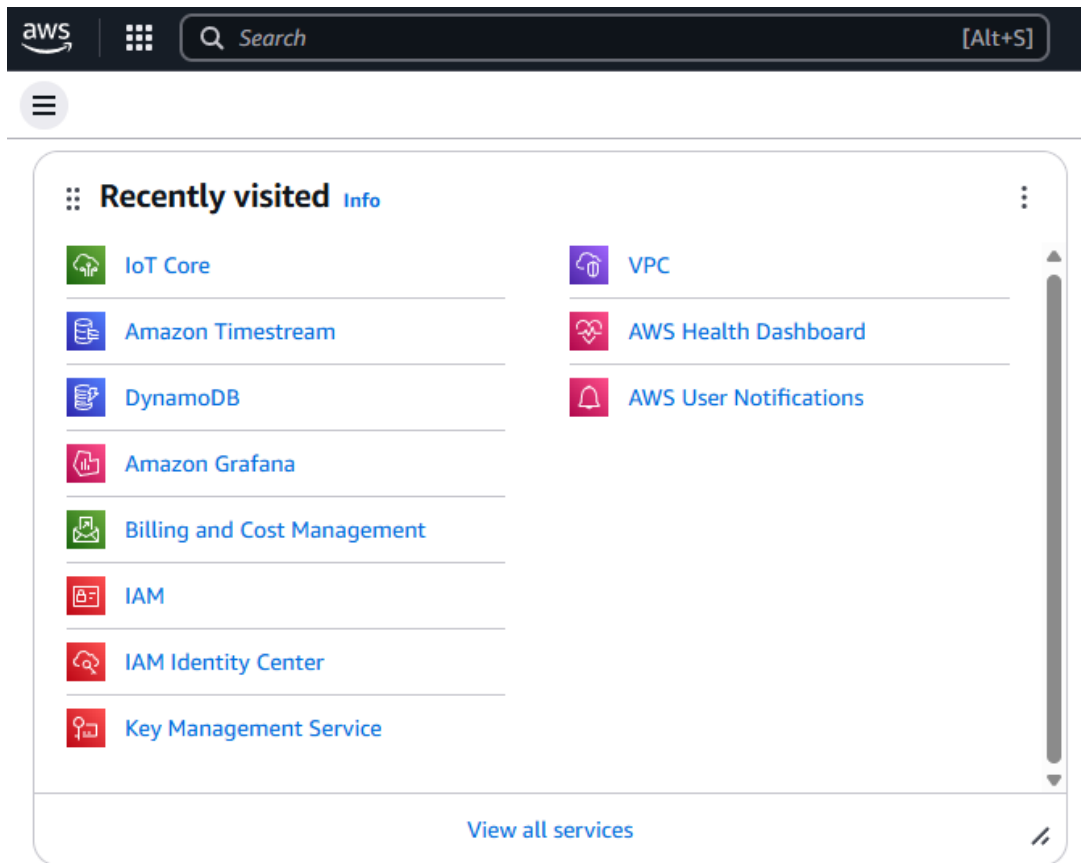


Figura 39. Consola AWS. Tomada de AWS (2025).

4.5.4.2. Conexión de objetos (Things) en AWS IoT Core

IoT Core mostrado en la ventana de la Figura 40, es un servicio de AWS que facilita la conexión de dispositivos físicos u objetos (*things*) como microcontroladores, sensores, electrodomésticos, vehículos y cámaras a internet de manera segura para su interactividad y comunicación entre estos, o con aplicaciones en la nube.

El microcontrolador ESP8266 el cual administra el sistema embebido del proyecto es un objeto que debe estar conectado a internet y al servicio de IoT Core para poder gestionar, administrar, autenticar y cifrar los mensajes de los datos obtenidos por los sensores con seguridad principalmente. Para realizar todas estas funciones, lo primero que debe hacerse es agregar un objeto (*thing*) nuevo en IoT Core.

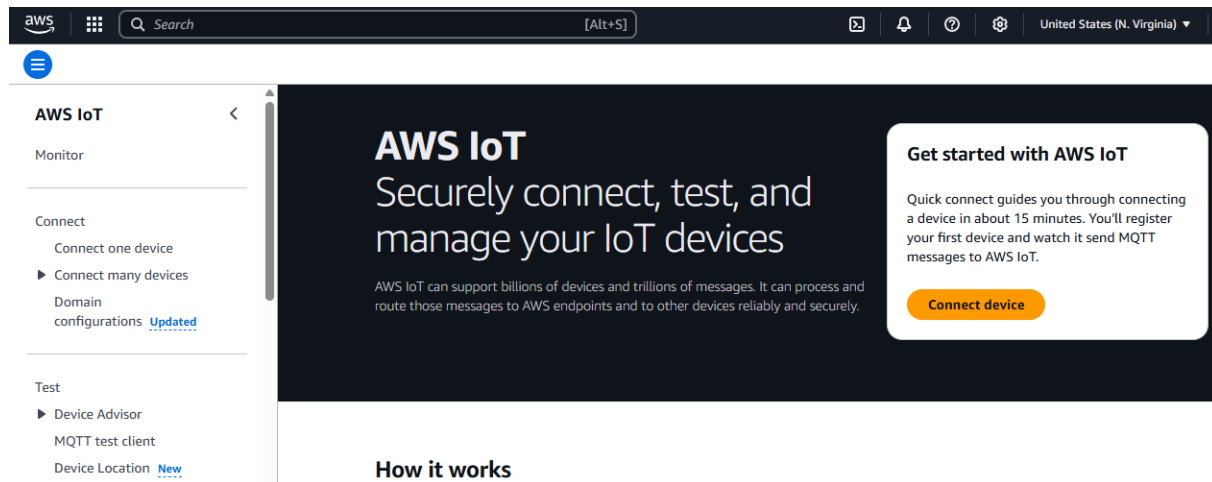


Figura 40. Ventana de inicio de AWS IoT Core. Tomada de AWS (2025).

Para agregar el ESP8266 como objeto al servicio de IoT Core se debe hacer lo siguiente.

i. Accesar a la consola de AWS IoT Core

Ingresar a la cuenta de AWS y dirigirse al servicio IoT Core desde la consola principal.

ii. Crear un "Thing" (dispositivo)

En el panel izquierdo como se ilustra en la Figura 41, seleccionar "Manage" > "Things" > "Create things".

Elegir entre crear un solo dispositivo o varios a la vez. Para comenzar, se recomienda la opción de crear un solo objeto.

iii. Asignar un nombre y tipo al "Thing"

Introducir un nombre representativo (por ejemplo: SensorTemperatura01) y, si se desea, asignar un tipo o grupo para facilitar su gestión. Para el caso de este proyecto el objeto ha sido nombrado como **ESP8266_thing**.

iv. Crear certificados de seguridad

AWS IoT Core necesita autenticar los dispositivos. En este paso, se puede generar automáticamente un certificado y clave privada, así como el archivo de política de conexión y descargar los archivos. **certificados .der** mostrados en la Figura 42.

v. Adjuntar una política al certificado

Las políticas definen que permisos tiene el dispositivo (por ejemplo: publicar o suscribirse a ciertos tópicos). Las políticas mostradas en la Figura 43, se pueden definir aquí mismo, es recomendable

marcar con * **“todos”** para que el objeto **ESP8266_thing** pueda acceder a todos los servicios y acciones.

vi. Conectar el "Thing"

Una vez creado el objeto y configurada su seguridad, AWS ofrecerá una guía con ejemplos de código y opciones para conectarlo a través de MQTT, HTTP o WebSockets.

vii. Probar la conexión

Desde la pestaña de MQTT test client, se puede enviar o recibir mensajes simulados para asegurarse de que el dispositivo está correctamente conectado.

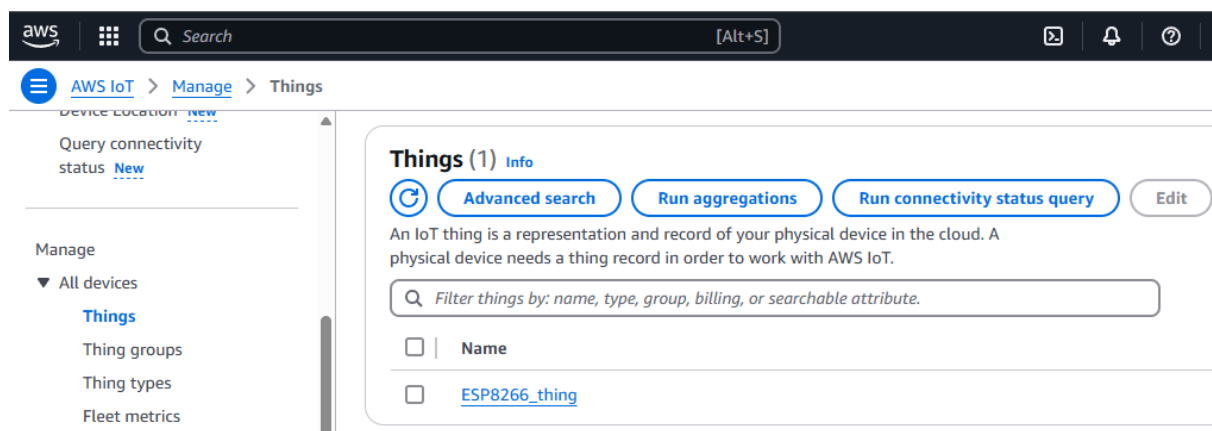


Figura 41. Ventana de objetos creados en AWS IoT Core. Tomada de AWS (2025).

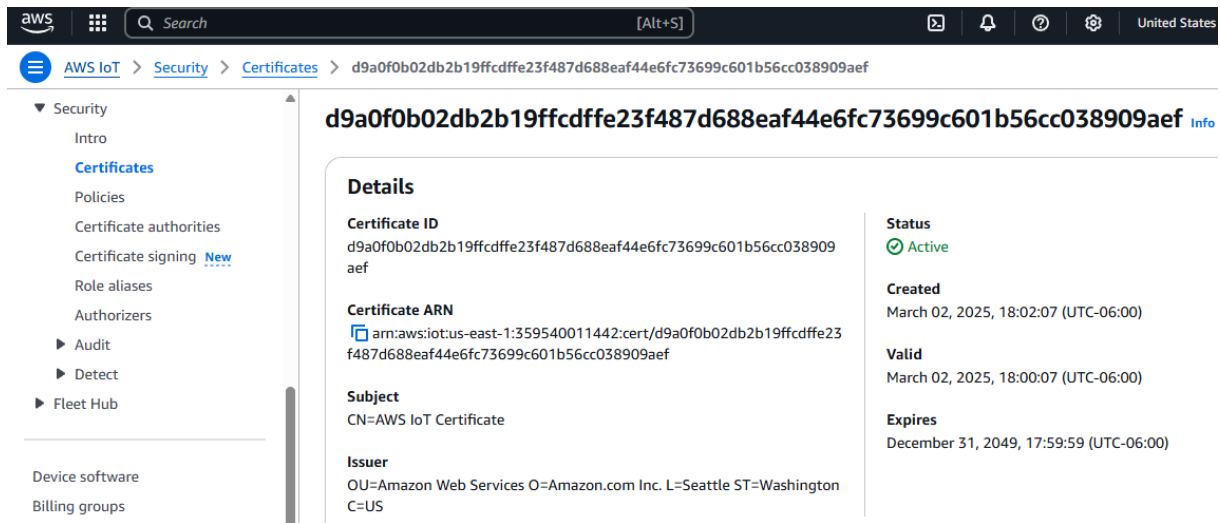


Figura 42. Ventana de certificados de seguridad (certificados .DER) creados en AWS IoT Core. Tomada de AWS (2025).

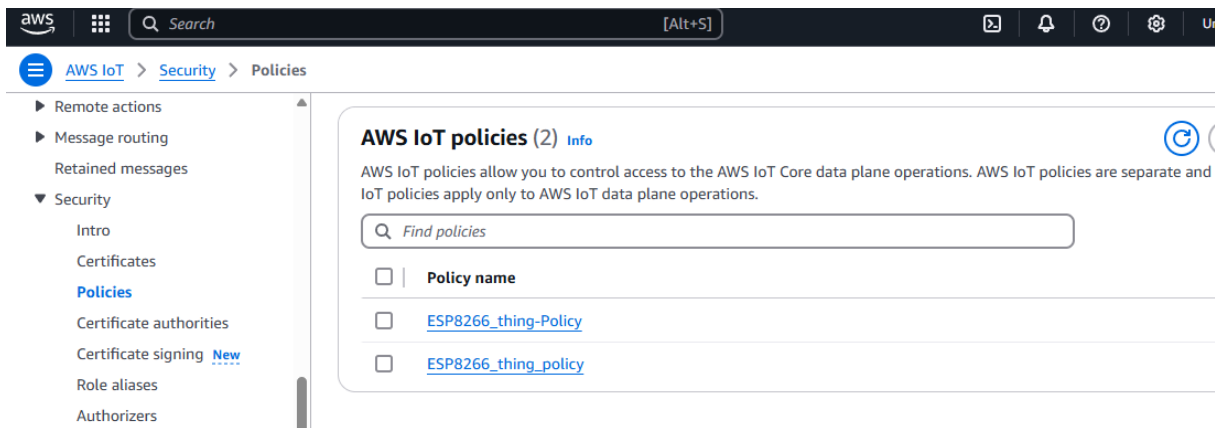


Figura 43. Ventana de políticas creadas en AWS IoT Core. Tomada de AWS (2025).

4.5.4.3. Pruebas de conexión MQTT en AWS IoT Core

Una vez que se ha creado el objeto (**thing**), los certificados de seguridad y las políticas de acciones permitidas con el objeto y los datos, es necesario realizar una prueba de comunicación mediante protocolo MQTT desde el sistema embebido al servicio de IoT Core. Para realizar una prueba se deben seguir los siguientes pasos:

i. Ingresar a la consola de AWS

Acceder con credenciales y dirigirse al servicio IoT Core desde el buscador del panel principal.

ii. Abrir el cliente de prueba MQTT

En el menú izquierdo, seleccionar "Test" y luego haz clic en "MQTT test client".

iii. Suscribirse a un tópico

Hacer clic en "Subscribe to a topic".

Escribir un nombre de t pico, por ejemplo: dispositivos/temperatura. Para el caso del presente proyecto el tema es: **esp8266/datos** como se observa en la Figura 44.

Presionar "Subscribe" para comenzar a escuchar mensajes que lleguen a ese canal.

iv. Publicar un mensaje en el mismo t pico

Cambiar a la pesta a "Publish to a topic".

Escribir el mismo nombre de t pico, ejemplo: (dispositivos/temperatura). Nuevamente escribir el nombre del tema: **esp8266/datos** (Figura 45).

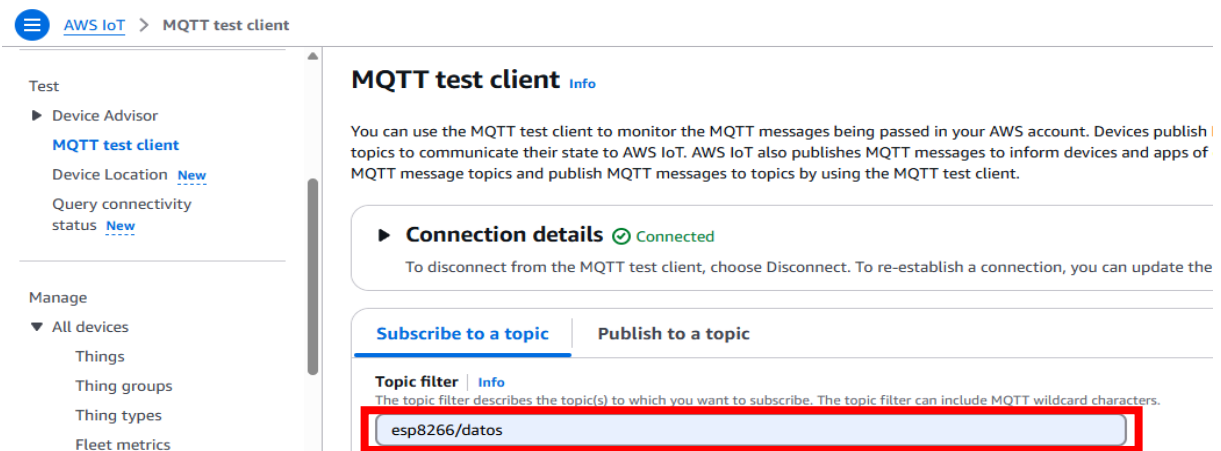


Figura 44. Prueba MQTT en AWS IoT Core. Tomada de AWS (2025).

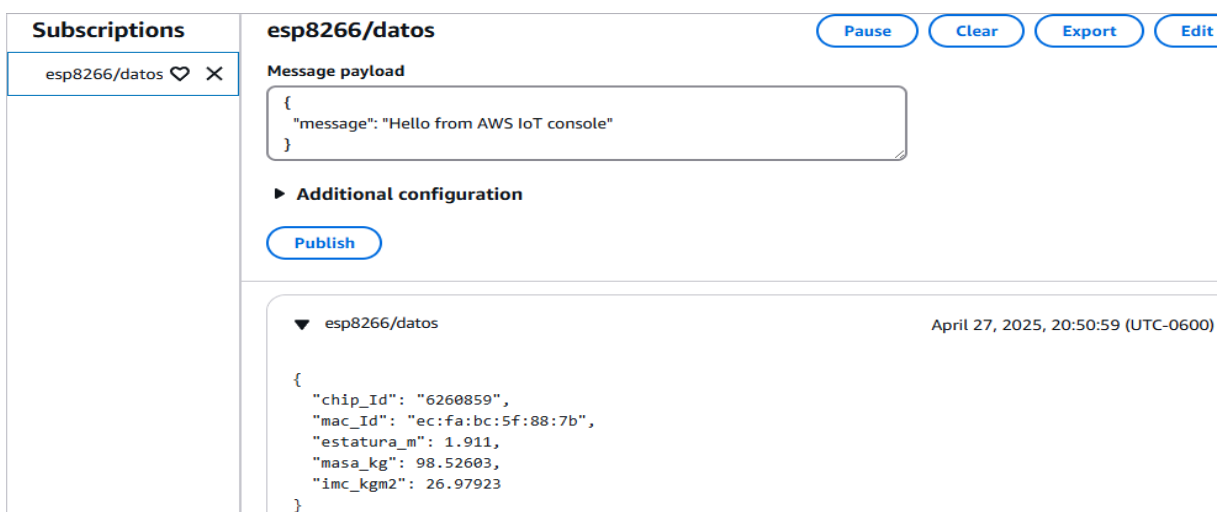


Figura 45. Vista de suscripci n MQTT en AWS IoT Core. Tomada de AWS (2025).

4.5.4.4. Ingesta de datos con AWS Timestream

AWS Timestream es una base de datos especializada en manejar datos de series temporales, como las lecturas de sensores, métricas de rendimiento o registros de actividad.

Cuando se habla de ingesta de datos en Timestream, se hace referencia al proceso de capturar y almacenar datos que vienen acompañados de una marca de tiempo (**timestamp**). Estos datos se adquieren a través del servicio de suscripción mediante el protocolo MQTT, particularmente del tema **esp8266/datos**. Cada punto de información contiene no solo el valor medido como el peso y la masa del usuario, sino también el momento exacto en que fue registrado. Esta característica es fundamental en aplicaciones como monitoreo industrial, dispositivos IoT, análisis de logs y sistemas de salud.

La ventaja de *Timestream* es que permite enviar datos desde distintas fuentes, como *AWS IoT Core*, *Lambda* o *Kinesis*, de forma directa y sin tener que preocuparse por la infraestructura que hay detrás. Además, almacena la información de forma optimizada y con compresión automática, lo que permite realizar consultas rápidas, incluso sobre millones de registros.

El sistema embebido IoT mide la estatura, la masa del usuario y realiza el cálculo del IMC. Timestream recibe estos valores, los organiza por fecha y hora, y los deja listos para ser visualizados o analizados. Todo esto, sin necesidad de diseñar complejos esquemas de base de datos o preocuparse por el mantenimiento. AWS Timestream no solo permite capturar lo que ocurre, sino también entender cuándo ocurre, lo que es esencial para tomar decisiones informadas en tiempo real. A continuación, se muestra la metodología para crear una base de datos en AWS Timestream:

i. Accesar a la consola de AWS

Iniciar sesión en la cuenta de AWS y seleccionar "*Timestream*".

ii. Clic en "*Databases*"

Una vez dentro del panel de Timestream que se observa en la Figura 46, ir al menú lateral izquierdo y seleccionar "Databases".

iii. Crear una nueva base de datos

Pulsar el botón "**Create database**"

En este apartado, se debe indicar si se desea una base de datos standard (estándar) o magnetic (almacenamiento optimizado para largo plazo). La base de datos generada para la ingesta de datos de este proyecto es estándar.

La opción estándar es ideal para la mayoría de los casos de uso en tiempo real.

iv. Asignar un nombre

Escribir un nombre único y representativo para la base de datos. La base de datos creada lleva por nombre ***imc_loT*** como se aprecia en la Figura 47.

v. Configurar etiquetas (opcional)

Se pueden añadir etiquetas (***tags***) para clasificar o identificar mejor los recursos, especialmente útil en proyectos grandes o de varios equipos.

vi. Confirmación de la creación

Clic en ***"Create database"***. En unos segundos, la base de datos estará lista para ser usada.

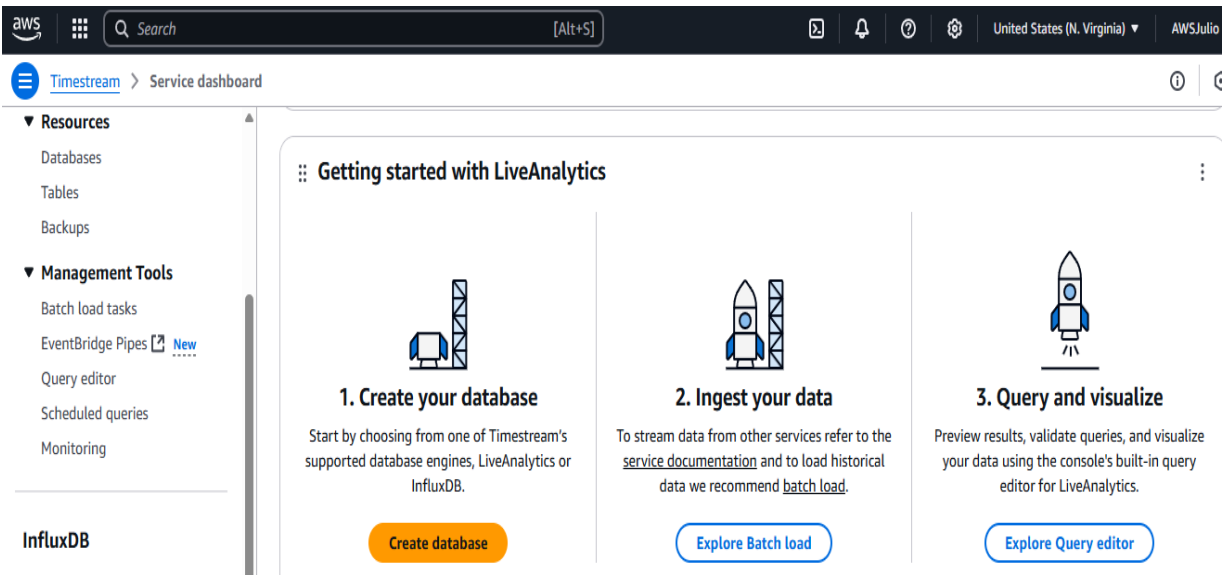


Figura 46. Ventana de inicio de AWS de Timestream. Tomada de AWS (2025).

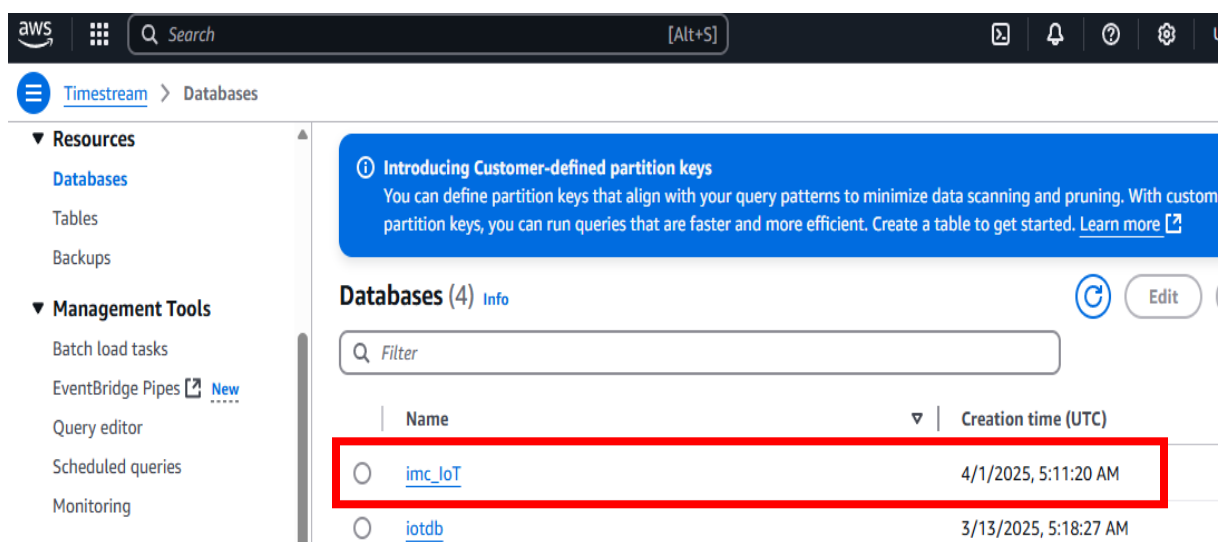


Figura 47. Base de datos imc IoT de la ingesta, creada en Timestream. Tomada de AWS (2025).

Después de generar la base de datos se debe crear una tabla dentro de ella, donde se definirán los períodos de retención y se almacenarán los datos. A continuación, se describen los pasos:

i. Acceder a la consola de Timestream

Ingresar a la consola de AWS, seleccionar el servicio Amazon Timestream, y asegurarse de estar en la región correcta donde se creó la base de datos. La región utilizada en el desarrollo de este proyecto es: **us-east_1 United States (N. Virginia)**. Es importante no perder de vista este dato.

ii. Ingresar a la sección "Tables"

En el panel izquierdo, clic en "Tables" y luego presionar el botón "Create table".

iii. Seleccionar la base de datos

Elegir la base de datos en la que se desea crear la tabla desde el menú desplegable.

En este caso es: **imc IoT**.

iv. Asignar un nombre a la tabla

Introducir un nombre claro y descriptivo. Para el desarrollo de este proyecto la tabla tiene el nombre de: **imc_Data IoT** como se observa en la Figura 48.

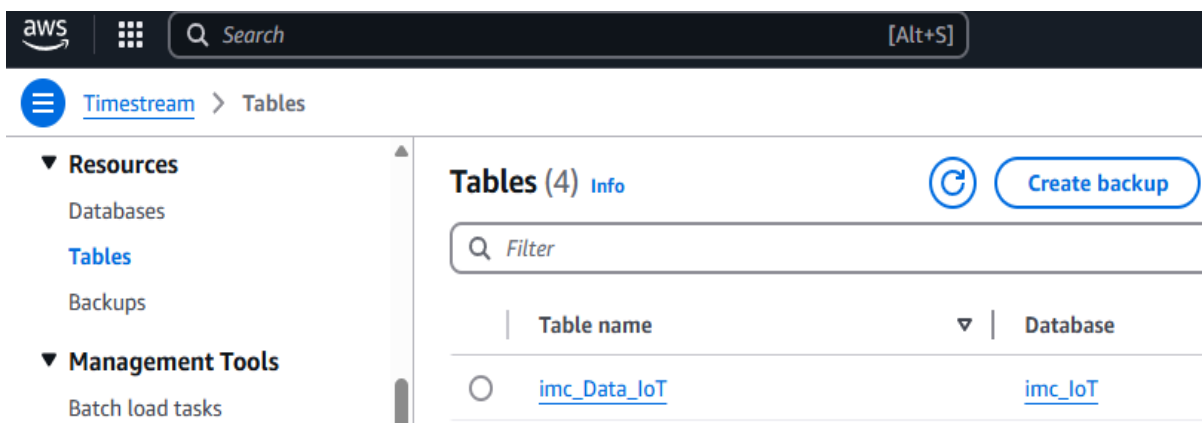


Figura 48. Tabla `imc_Data_IoT` de la ingesta, creada en Timestream. Tomada de AWS (2025).

Es necesario establecer los períodos de retención para los datos en memoria y en almacenamiento magnético:

- **Memory store retention:** determina cuánto tiempo se mantienen los datos en almacenamiento rápido (por ejemplo: 1 hora a 8760 horas).
- **Magnetic store retention:** define cuánto tiempo se conservarán los datos en almacenamiento a largo plazo (por ejemplo: 1 día a 200 años).

v. (Opcional) Agregar etiquetas

Se pueden añadir etiquetas clave-valor para organizar los recursos, especialmente útil en entornos con múltiples usuarios o proyectos.

vi. Crear la tabla

Clic en **"Create table"**. En segundos, la tabla estará lista para comenzar a recibir datos.

Importante: Las tablas en Timestream no requieren definir un esquema fijo; el servicio detecta automáticamente los campos enviados con los datos, lo que lo hace flexible y fácil de escalar.

Para que la ingesta de datos pueda ser almacenada satisfactoriamente dentro de la tabla `imc_Data_IoT`, en la base de datos `imc_IoT` de **Timestream**, es indispensable aplicar una **regla de interacción entre AWS IoT Core y Timestream** para indicarle a este servicio que se debe hacer con los datos (mensajes) ingresados a través del *broker*. Las reglas de interacción permiten que los datos que llegan desde dispositivos IoT, se almacenen automáticamente en una tabla de Timestream para análisis posterior.

Los pasos para crear una regla de interacción son los siguientes:

i. Acceder a la consola de AWS IoT Core

Ingresar a la cuenta de AWS y seleccionar el servicio IoT Core.

ii. Seleccionar el menú “Act” (Actions)

En el menú lateral, clic en ***“Message Routing”*** y después ***“Rules”***, luego seleccionar ***“Create rule”***.

iii. Asignar un nombre a la regla

Escribir un nombre descriptivo para la regla, por ejemplo: ReglaTemperaturaTimestream. También es posible agregar una breve descripción para identificar su propósito. La regla definida para este proyecto es nombrada como: ***imc_rule_IoT*** como se aprecia en la Figura 49.

iv. Definir la consulta SQL

En el campo de consulta, introducir la instrucción SQL que extraerá los datos del mensaje MQTT que se desea enviar a Timestream, la instrucción SQL para enviar los datos a la tabla ***imc_Data_IoT*** es la siguiente:

```
SELECT estatura_m, masa_kg, imc_kgm2 FROM 'esp8266/datos'
```

En donde:

estatura_m → es la variable que almacena el valor del mensaje estatura

masa_kg → es la variable que almacena el valor del mensaje masa

imc_kgm2 → es la variable que almacena el valor del mensaje IMC

Todos los datos provienen del tema ***'esp8266/datos'*** donde se publican los mensajes a través del broker en IoT Core.

v. Agregar una acción de Timestream

Clic en ***“Add action”*** y seleccionar ***“Insert into Amazon Timestream table”***.

vi. Configurar la acción Timestream

- Base de datos: seleccionar la base de datos creada ***imc_IoT***
- Tabla: elegir la tabla correspondiente. ***imc_Data_IoT***

- Nombre de dimensión: indica una etiqueta clave, como dispositivo. Para este proyecto se han definido 2 dimensiones: **chip_id** → para identificar al microcontrolador y **mac_id** → que almacena la dirección física del microcontrolador.
- Nombre de medida: el nombre de la medida es → **\${timestamp()}**
- Unidad de medida: está dada en milisegundos → **MILLISECONDS**
- Tipo de medida: El tipo de medida es cadena de caracteres → **\$ {STRING}**

vii. Crear un rol de IAM

AWS IoT necesita permisos para escribir en Timestream. Si no existe un rol IAM existente, elegir la opción **"Create a new role"** y dejar la configuración automática de AWS, el nombre del rol definido es: **IMC_timestream_role**.

viii. Guardar la regla

Clic en **"Create rule"** para finalizar el proceso. La regla quedará activa y comenzará a almacenar los datos que coincidan con la consulta definida.

Edit rule: imc_rule_IoT

Rule properties

Rule description - optional
Enter a description to provide additional details about the rule to others.

Calculo de estatura, peso e IMC

SQL statement info

SQL version
The version of the SQL rules engine to use when evaluating the rule.

2016-03-23

SQL statement
Enter a SQL statement using the following: `SELECT <Attribute> FROM <Topic Filter> WHERE <Condition>`. For example: `SELECT temperature FROM 'iot/topic' WHERE temperature > 50`. To learn more, see [AWS IoT SQL Reference](#)

```
1 SELECT estatura_m, masa_kg, imc_kgm2 FROM 'esp8266/datos'
```

Figura 49. Creación de regla de interacción de la ingesta, en Timestream. Tomada de AWS (2025).

4.5.4.5. Consultas en Timestream

Para visualizar los datos almacenados se debe realizar una consulta como se visualiza en la Figura 50. Para realizar una consulta se presentan los siguientes pasos:

i. Acceder a la consola de AWS

Iniciar sesión en la cuenta de AWS y abrir el servicio Amazon Timestream.

ii. Ir a la sección "Query editor"

En el panel lateral izquierdo, clic en **"Query editor"**, es aquí donde se pueden escribir y ejecutar consultas SQL sobre los datos de series temporales.

iii. Seleccionar la base de datos y tabla

En el editor, seleccionar la base de datos y la tabla que contienen los datos que se desean consultar. Esto se puede hacer desde el panel desplegable o directamente en la consulta.

iv. Escribe la consulta SQL

En el área del editor, introducir la instrucción SQL para extraer los datos. Por ejemplo, para obtener las mediciones más recientes, se introduce la siguiente instrucción:

“SELECT * FROM "imc_IoT"."imc_Data_IoT" WHERE time between ago(15m) and now() ORDER BY time DESC LIMIT 10”

Esta consulta devuelve las lecturas de estatura, masa e IMC registradas en los últimos 15 minutos en la **base de datos imc_IoT** y la **tabla imc_Data_IoT**, ordenadas por tiempo, con un límite de 10 elementos.

v. Ejecuta la consulta

Clic en el botón **“Run query”**. Los resultados aparecerán en la parte inferior del editor, en formato de tabla.

vi. Analizar o exporta los resultados (opcional)

Se podrán revisar los resultados directamente o copiarlos para análisis externo. Timestream también permite conectar estas consultas a herramientas como Amazon QuickSight para visualización avanzada.

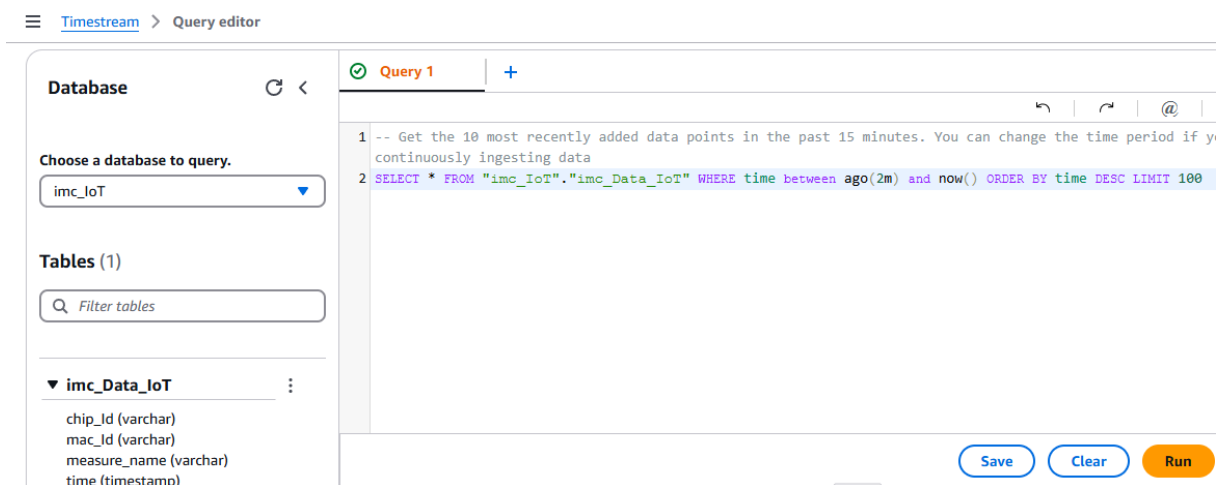


Figura 50. Tabla consulta (Query) SQL, en Timestream. Tomada de AWS (2025).

4.5.4.6. Almacenamiento permanente con AWS DynamoDB

AWS DynamoDB mostrado en la Figura 51, está diseñado para ofrecer alta disponibilidad, rendimiento constante y baja latencia, lo que lo hace ideal para aplicaciones web, móviles, y de IoT que manejan grandes volúmenes de datos y requieren respuestas rápidas.

Lo más destacado de **DynamoDB** es su capacidad para manejar millones de solicitudes por segundo sin que el rendimiento se vea afectado. Esto se logra mediante un modelo de datos clave-valor, donde se pueden almacenar y acceder a datos de manera rápida y sencilla. Además, **DynamoDB** ofrece características como escalabilidad automática, seguridad integrada, y backup continuo, lo que facilita la gestión de bases de datos sin necesidad de intervención manual.

A continuación, se muestran los pasos para crear una tabla en el servicio **DynamoDB** de AWS de forma manual a través de la consola.

- i. **Acceder a AWS Management Console**
 - Ingresar al sitio **AWS Console**.
 - Autenticarse con credenciales de usuario.
- ii. **Ingresar al servicio DynamoDB**
 - En la barra de búsqueda de servicios, escribir **“DynamoDB”** y seleccionarlo.
- iii. **Iniciar la creación de una nueva tabla**
 - Clic en el botón **“Create table”**.
- iv. **Configurar los datos básicos de la tabla**
 - **Table name:** Ingresar el nombre que tendrá la tabla (por ejemplo: Usuarios). Para este proyecto el nombre de la tabla es **imc_loT_Data** como se observa en la Figura 52.
 - **Partition key:** Definir el nombre del atributo que funcionará como clave primaria (por ejemplo: UsuarioID). De manera similar que la tabla de AWS Timestream, el nombre del atributo de la tabla creada es **chip_Id** tipo **String**.
 - **(Opcional) Sort key:** Definir un atributo adicional para ordenar registros dentro de una misma partición, para este caso si es requerido y es denominado **TS** de tipo **Number**.
- v. **Configurar las opciones avanzadas (opcional)**
 - **Seleccionar el modo de capacidad:**
 - **On-demand** (pago por uso automático). Elegido para la tabla **imc_loT_Data** de este proyecto.
 - **Provisioned** (capacidad fija con configuración de unidades de lectura y escritura).
 - Configurar índices secundarios, si desea habilitar búsquedas alternativas.

- Activar otras opciones como copias de seguridad automáticas o encriptación de datos, si es necesario.

vi. **Finalizar la creación de la tabla**

- Revisar la configuración realizada.
- Clic en el botón **"Create table"**.

vii. **Esperar la activación de la tabla**

- Esperar a que el estado de la tabla cambie a **"Active"**, momento en el cual estará disponible para operaciones de inserción, lectura o actualización de datos

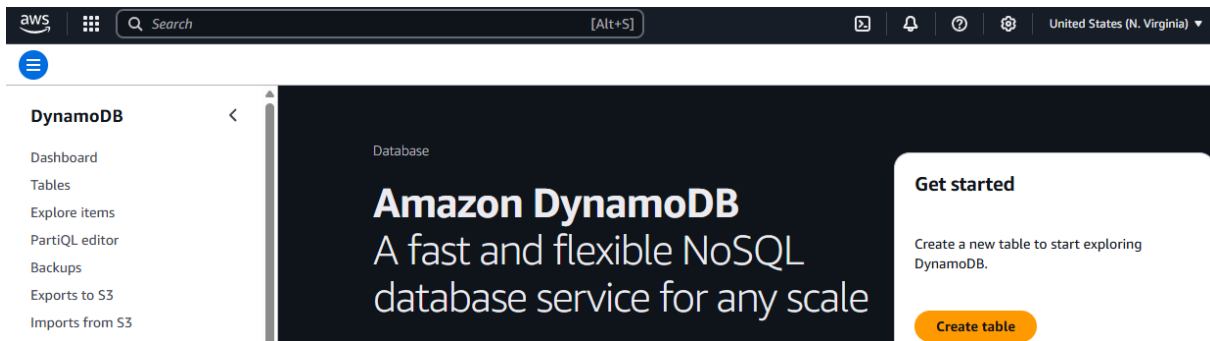


Figura 51. Ventana de inicio de AWS DynamoDB. Tomada de AWS (2025).

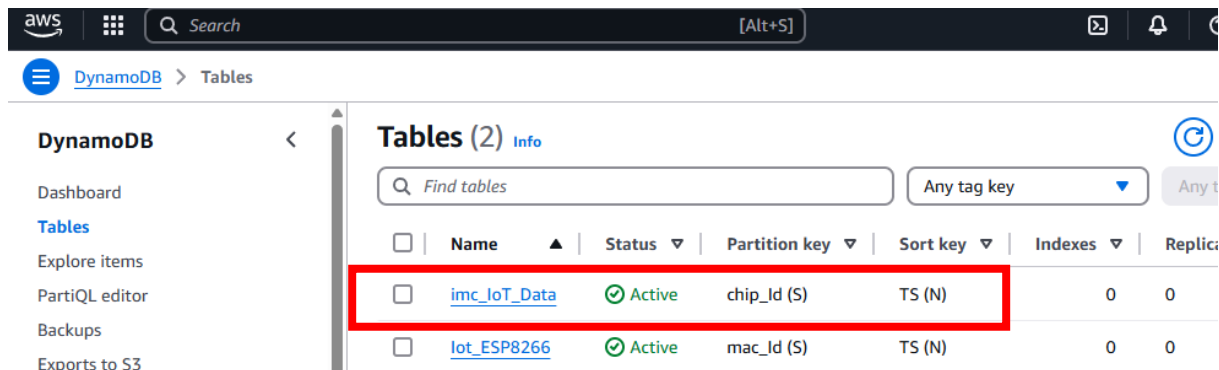


Figura 52. Tabla imcIoT_Data, creada en AWS DynamoDB. Tomada de AWS (2025).

Para que la ingesta de datos se pueda llevar a cabo de manera satisfactoria a la tabla **imcIoT_Data**, se debe generar una regla de interacción entre **AWS IoT Core** y **AWS DynamoDB**.

Para crear una regla se debe realizar lo siguiente:

i. **Acceder a la consola de AWS IoT Core**

Ingresar a la cuenta de AWS y seleccionar el servicio IoT Core.

ii. Seleccionar el menú “Act” (Actions)

En el menú lateral, clic en “**Message Routing**” y después “**Rules**”, luego seleccionar “**Create rule**”.

iii. Asignar un nombre a la regla

Escribir un nombre descriptivo para la regla, también es posible agregar una breve descripción para identificar su propósito. La regla definida para este proyecto es nombrada como: **imc_loT_rule** tal como se observa en la Figura 53.

iv. Definir la consulta SQL

En el campo de consulta, introducir la instrucción SQL que extraerá los datos del mensaje MQTT que se desea enviar a **DynamoDB**, la instrucción SQL para enviar los datos a la tabla **imc_loT_Data** es la siguiente:

```
SELECT *, timestamp() AS TS FROM 'esp8266/datos'
```

En donde:

timestamp() → es una función en AWS IoT Core que obtiene la hora y fecha actuales del evento (cuando se recibe el mensaje MQTT).

TS → es un alias, que significa que el valor retornado por **timestamp()** se llamará **TS**. En otras palabras, se asignará el nombre **TS** a la columna que contiene la marca de tiempo.

Todos los datos provienen del tema **'esp8266/datos'** donde se publican los mensajes a través del *broker* en IoT Core.

v. Agregar una acción de DynamoDBv2

Clic en “**Add action**” y seleccionar **DynamoDBv2** " Split message into multiple columns of a DynamoDB table". Que a diferencia de DynamoDB, DynamoDBv2 inserta los datos de la ingesta en columnas diferentes.

vi. Configurar la acción DynamoDBv2

- Tabla: elegir la tabla correspondiente. **imc_loT_Data**.
- Definir el **Partition key** del elemento: indica una etiqueta clave, como dispositivo. Para este proyecto se han definido **chip_Id** → para identificar al microcontrolador.
- Tipo de medida de **Partition Key** : El tipo de medida es cadena de caracteres → **STRING**

vii. Crear un rol de IAM

AWS IoT necesita permisos para escribir en **DynamoDBv2**. Si no existe un rol IAM existente, elegir la opción **"Create a new role"** y dejar la configuración automática de AWS, el nombre del rol definido es: **imc_loT_role**.

ix. Guardar la regla

Clic en **"Create rule"** para finalizar el proceso. La regla quedará activa y comenzará a almacenar los datos que coincidan con la consulta definida.

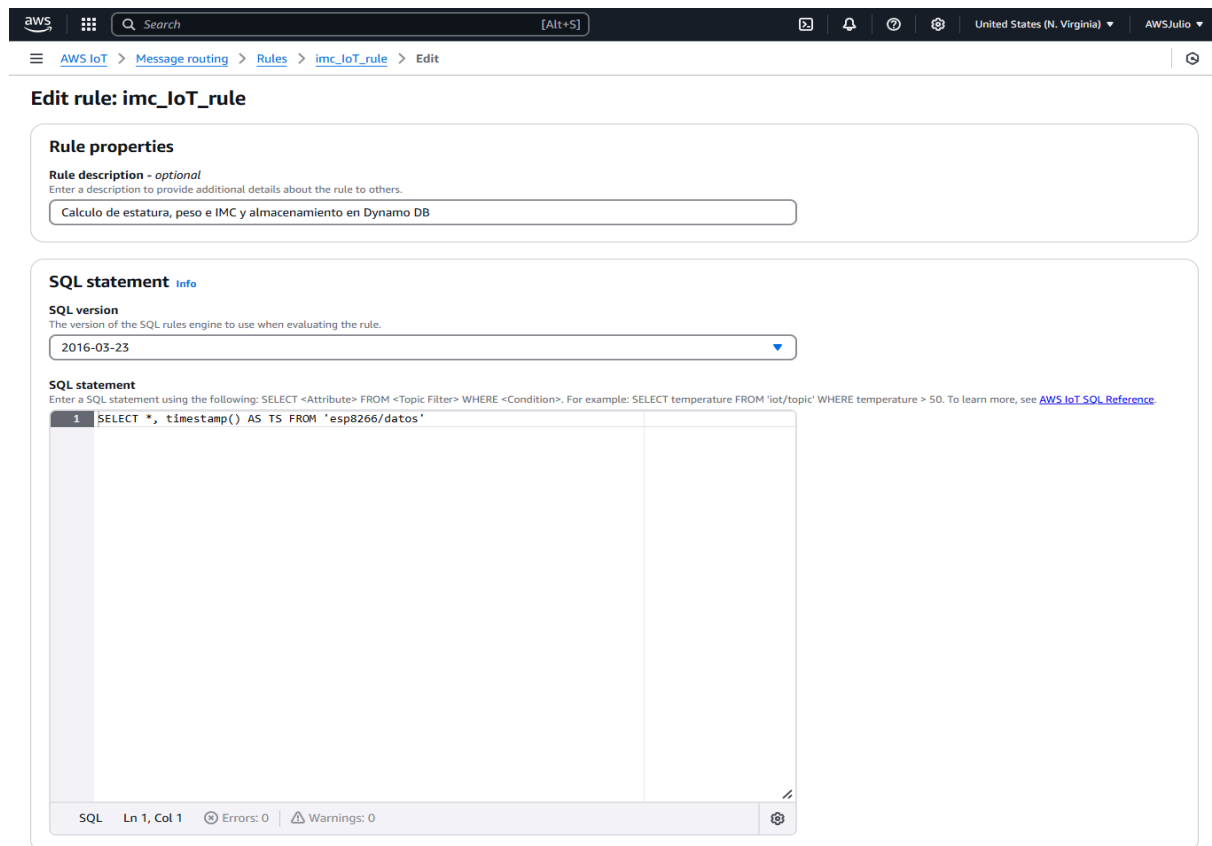


Figura 53. Creación de regla de interacción de la ingesta, en AWS DynamoDB. Tomada de AWS (2025).

4.5.4.7. Consultas en DynamoDB

DynamoDB es una base de datos **NoSQL** basada en índices y claves. Por tal motivo, los datos para realizar una consulta dependen de los campos clave y de cómo esté estructurada la tabla.

Para realizar una consulta en la tabla creada para este proyecto, se debe tomar en cuenta principalmente el campo clave (**partition key**) **chip_Id** o identificador del microcontrolador, y la

clave de clasificación (**sort key**) determinada por la marca de tiempo **TS** que es retornado por la función timestamp()).

Pasos para realizar consultas en **DynamoDB**:

- i. **Iniciar sesión** en la consola de AWS.
- ii. **Abrir DynamoDB** desde la consola.
- iii. **Seleccionar la tabla** en la que se desea hacer la consulta (Figura 54).
- iv. **Seleccionar la opción consultar ítems**, debe indicarse la opción **Query** en Scan or query items.
- v. **Configura la consulta** usando la **Partition Key**, y si es necesario, la **SortKey**, proyección de atributos y filtros. **Chip_id** y **TS** respectivamente.
- vi. **Ejecutar la consulta** y revisar los resultados (Figura 55).
- vii. **Opcional:** Realizar acciones adicionales como paginación o nuevas consultas.

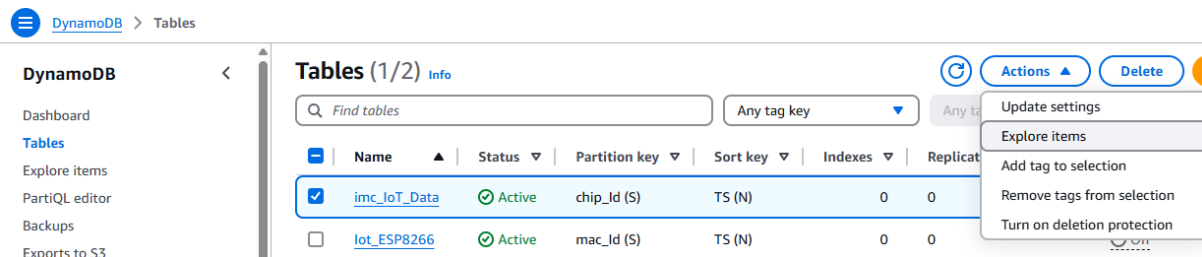


Figura 54. Selección de Tabla imc_IoT_Data, en AWS DynamoDB. Tomada de AWS (2025).

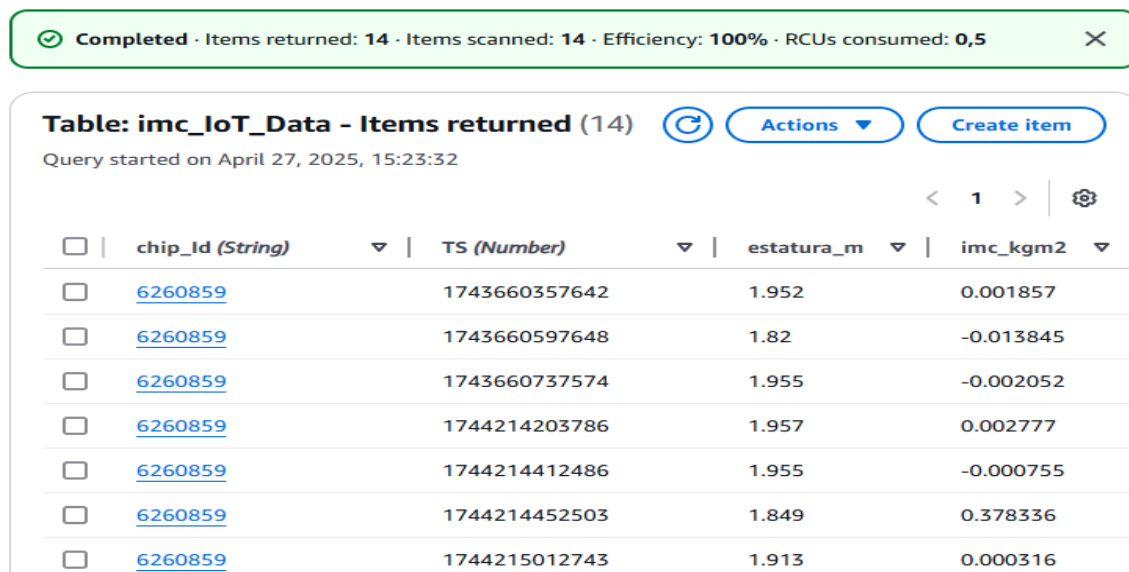


Figura 55. Consulta de datos en Tabla imc_IoT_Data, AWS DynamoDB. Tomada de AWS (2025).

4.5.4.8. Tablero (Dashboard) en Amazon Managed Grafana

El monitoreo de las variables obtenidas en tiempo real por los sensores HX711 Masa y VL53L0X Estatura, así como el cálculo del IMC es visible de manera instantánea por el usuario de IoT gracias al tablero (**dashboard**) creado. Para que el monitoreo de las variables se realice, el apoyo de la base de datos **imc_iot** de AWS Timestream es de gran importancia, ya que funge como fuente de datos que serán visibles en el tablero.

Amazon Managed Grafana mostrado en la Figura 56, facilita la visualización y el análisis colaborativo de datos. En el sector salud, este servicio se presenta como una herramienta de bastante utilidad para mejorar la gestión de datos, la toma de decisiones clínicas y el monitoreo en tiempo real de indicadores de salud. Su capacidad de integrar múltiples fuentes de datos y presentarlas en paneles interactivos facilita una gestión más eficiente y precisa de los recursos y servicios médico.

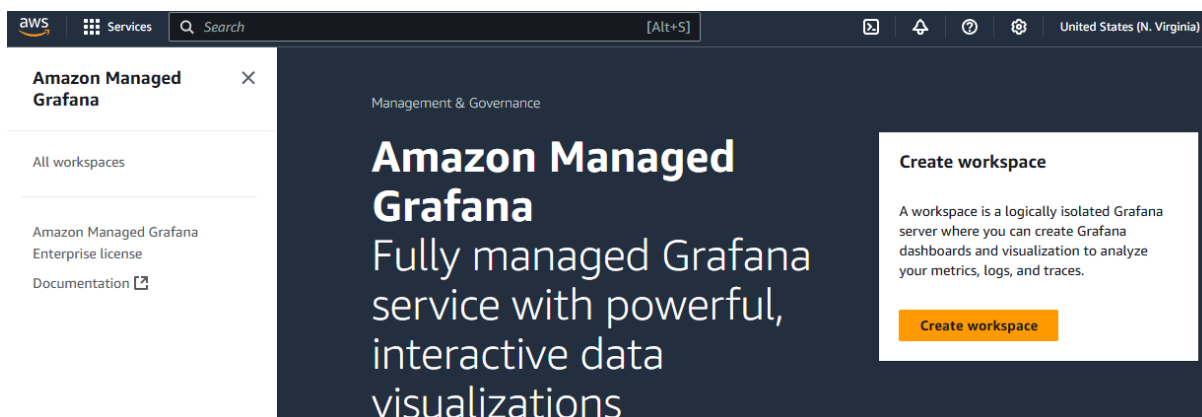


Figura 56. Ventana de inicio de Amazon Managed Grafana. Tomada de AWS (2025).

Los pasos para generar un **Dashboard** en **Amazon Managed Grafana** utilizando Amazon Timestream como Fuente de Datos son los siguientes:

i. Acceder a la Consola de AWS

Iniciar sesión en la Consola de Administración de AWS.

ii. Configurar Amazon Managed Grafana

En la consola de AWS, buscar **Amazon Managed Grafana**. Crear un espacio de trabajo como se ilustra en la Figura 57, seleccionando la región indicada en la ventana de la Figura 58, configurando la red y definiendo las opciones de autenticación, no perder de vista la región utilizada para este proyecto **us-east-1 United States (N. Virginia)**.

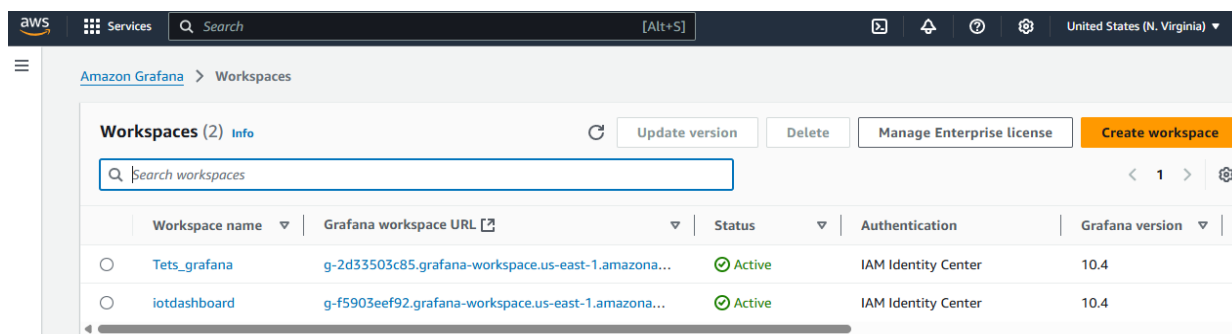


Figura 57. Espacio de trabajo de Amazon Managed Grafana. Tomada de AWS (2025).

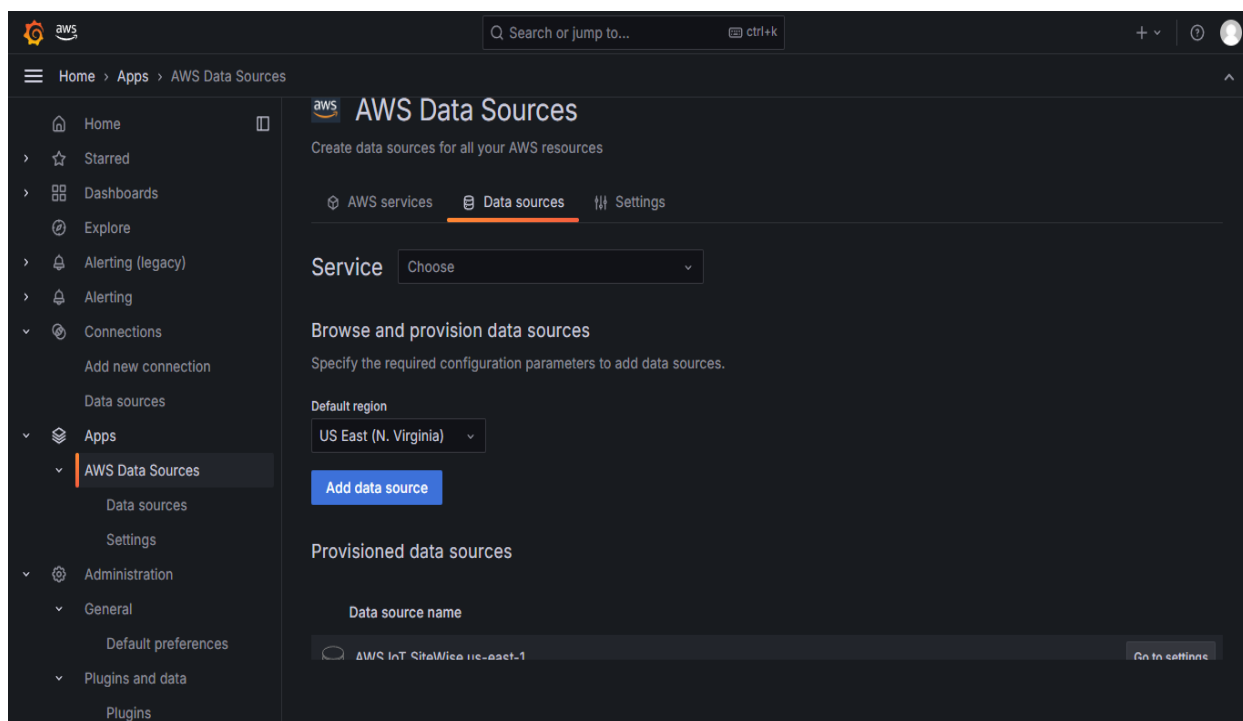


Figura 58. Selección de región y fuente de datos Amazon Managed Grafana. Tomada de AWS (2025).

iii. Conectar Amazon Timestream como Fuente de Datos

- En la consola de **Amazon Managed Grafana**, acceder a la sección Configuration (Configuración) > Data Sources (Fuentes de datos).
- Clic en Add data source (Agregar fuente de datos) y selecciona Amazon Timestream de la lista de opciones.
- Proporcionar los detalles necesarios para autenticar la conexión a Timestream (como las credenciales de IAM) y seleccionar la base de datos de Timestream que se desean consultar.

iv. Crear un Nuevo Dashboard

- En el panel principal de Grafana clic en Create (Crear) > Dashboard.
- Se abrirá una pantalla en blanco. Clic en Add Panel (Agregar Panel) para empezar a construir el panel de datos.

v. Configurar la Consulta de Timestream

- En el panel recién creado, seleccionar **Query** (Consulta) y en la fuente de datos, elegir Amazon Timestream como se ilustra en la Figura 59.
- Escribir la consulta T-SQL de Timestream para extraer los datos que se desean visualizar. Se pueden escribir consultas personalizadas que seleccionen métricas o registros específicos de la base de datos de Timestream. Para el **dashboard** de este proyecto se han definido 3 consultas diferentes y son:

- **Consulta de la variable estatura:**

Nombre de la columna que contiene la variable: **'estatura_m'**

- **Instrucción de consulta en T-SQL:**

```
SELECT * FROM $ database.$ table where measure name like 'estatura_m'
```

Grafana selecciona todos los datos que pertenecen a la base de datos de AWS Timestream identificada como **__\$database** y que están localizados en la tabla identificada como **__\$table**, donde el nombre del campo a medir coincide con la columna llamada **'estatura_m'**.

- **Consulta de la variable masa:**

Nombre de la columna que contiene la variable: **'masa_kg'**

- **Instrucción de consulta en T-SQL:**

```
SELECT * FROM $ database.$ table where measure name like 'masa_kg'
```

Grafana selecciona todos los datos que pertenecen a la base de datos de AWS Timestream identificada como **__\$database** y que están localizados en la tabla identificada como **__\$table**, donde el nombre del campo a medir coincide con la columna llamada **'masa_kg'**.

- **Consulta de la variable IMC:**

Nombre de la columna que contiene la variable: **'imc_kgm2'**

- **Instrucción de consulta en T-SQL:**

```
SELECT * FROM $ database.$ table where measure name like 'imc_kgm2'
```

Grafana selecciona todos los datos que pertenecen a la base de datos de AWS Timestream identificada como **__\$database** y que están localizados en la tabla identificada como **__\$table**, donde el nombre del campo a medir coincide con la columna llamada **'imc_kgm2'**.

- Ajustar la consulta según los parámetros necesarios, como el timestamp, las dimensiones (por ejemplo, una instancia específica), o las métricas para este caso **estatura en m**, **masa en kg** e **IMC en kgm²**.

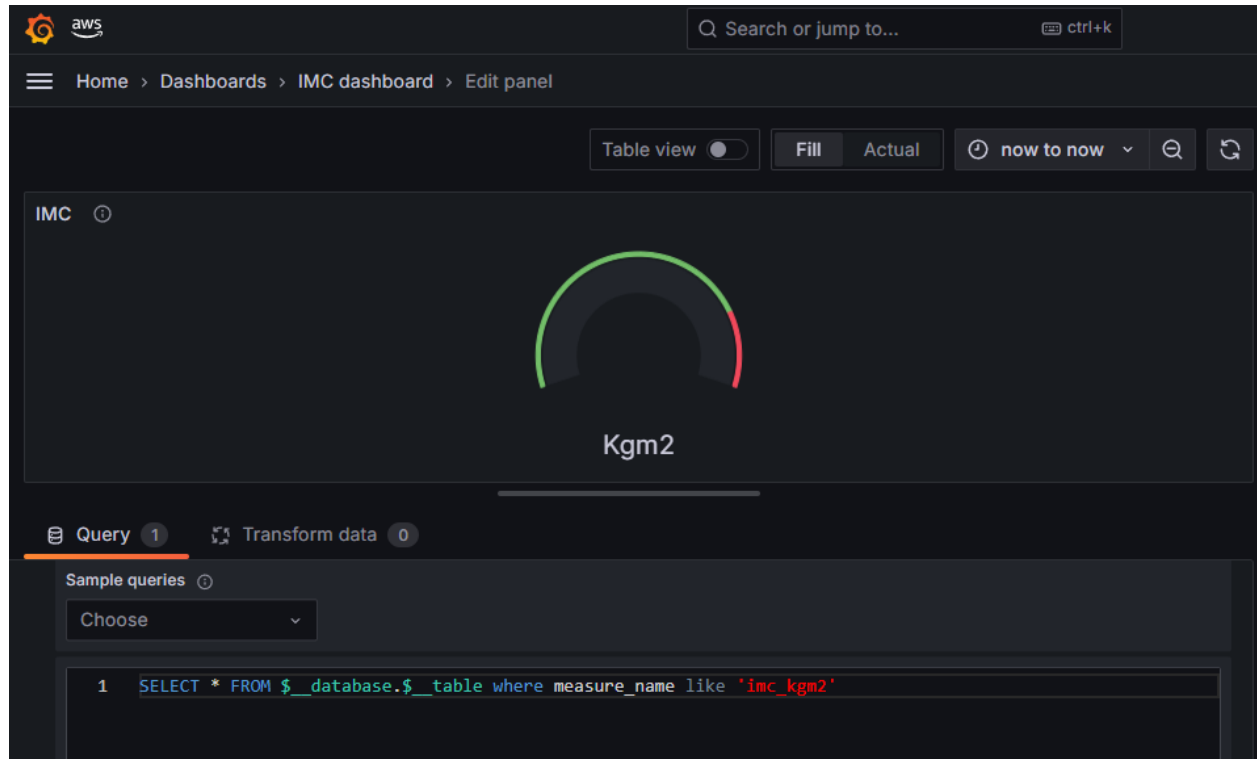


Figura 59. Consulta de datos T-SQL en Amazon Managed Grafana. Tomada de AWS (2025).

vi. Ajustar la Visualización del Panel

- Después de configurar la consulta, seleccionar el tipo de visualización que mejor representen los datos (gráfico de líneas, barras, tabla, etc.). Para las visualizaciones de los paneles de las tres consultas se utiliza el tipo **Gauge (indicador)**.
- Personalizar la visualización cambiando los colores, etiquetas, leyendas, y otras opciones de diseño para hacer los datos más comprensibles.

vii. Configurar Alertas para el Panel

- Si se desea se pueden configurar alertas para notificar a los usuarios sobre ciertos eventos (por ejemplo, si los valores superan un umbral crítico), para ello ir a la sección de **Alert** (Alerta) en la configuración del panel.
- Definir las condiciones de la alerta.

viii. **Agregar Más Paneles**

- Si se requiere visualizar más métricas o tipos de datos, repetir el paso **iv** para agregar más paneles.
- Es posible combinar diferentes visualizaciones de métricas de Timestream en un único **dashboard** para obtener una visión integral de los datos.

ix. **Personalizar el Dashboard con Variables**

- Para hacer el **dashboard** más interactivo, se pueden agregar variables que permitan a los usuarios seleccionar dinámicamente diferentes valores. Por ejemplo, puede agregarse una variable para elegir diferentes instancias de IoT o sensores que se visualizan en el **dashboard**.
- Ingresar a Dashboard Settings (Configuración del Dashboard) > Variables y agregar las variables correspondientes.

x. **Configurar el Rango de Tiempo y Filtros**

- En la parte superior del **dashboard**, seleccionar el rango de tiempo para la visualización de los datos. Podrán elegirse opciones predeterminadas como “Últimos 30 minutos” o personalizar un rango de tiempo específico, para este caso es “**Auto**”, el **dashboard** se actualizará automáticamente cuando nuevas variables sean enviadas a la base de datos **imc_loT** de AWS Timestream.
- Agregar filtros para que los usuarios puedan elegir diferentes parámetros de los datos, como fechas, métricas o categorías. Para que el dashboard, ilustrado en la Figura 60, se actualice automáticamente con los últimos datos obtenidos, la configuración del filtro es “**now to now**” en el rango de tiempo y fecha a desplegar en el panel.

xi. **Guardar y Compartir el Dashboard**

- Una vez que el **dashboard** está configurado, clic en el ícono guardar en la parte superior derecha.
- Asignar un nombre y una descripción al **dashboard** y guardar los cambios El **dashboard** ha sido nombrado como: **IMC dashboard**.
- Si se desea compartir, ajustar los permisos de acceso para permitir que otros usuarios de **Grafana** visualicen el **dashboard**. También se puede exportar el **dashboard** o compartir el enlace directo.

xii. **Monitoreo y Actualización del Dashboard**

- Revisar el desempeño del **dashboard** a medida que se usa. Si es necesario, ajustar las consultas, paneles o alertas para mejorar la visibilidad y la respuesta ante eventos.

- Asegurarse de que las métricas sean actualizadas correctamente y realizar mejoras continuas para mantener la relevancia de los datos mostrados.

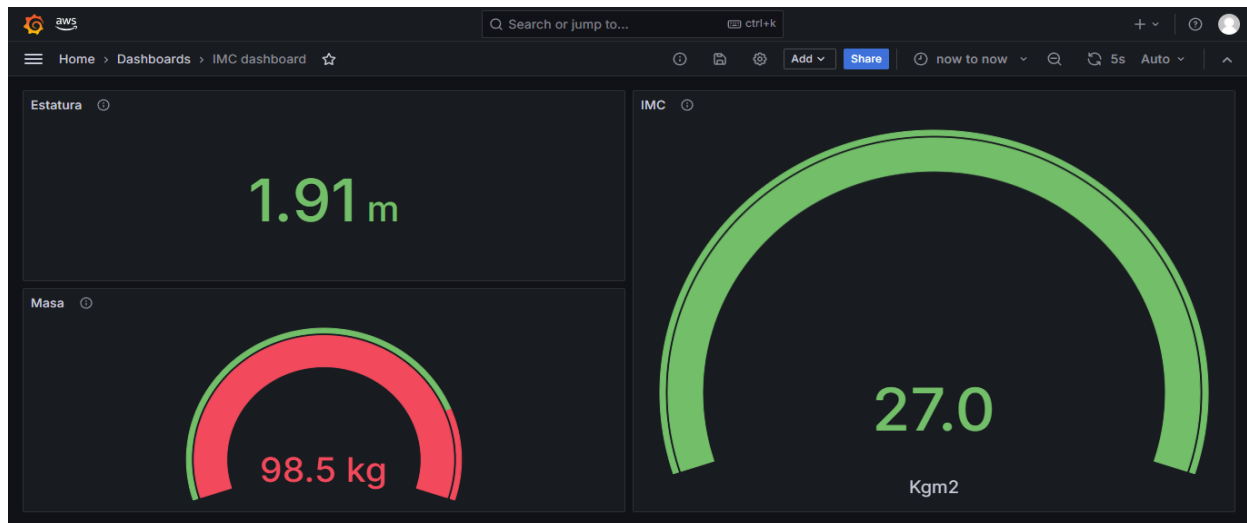


Figura 60. Vista de Dashboard en el navegador de PC. Tomada de AWS (2025).

Capítulo 5 Análisis y discusión de resultados

En un estudio previo publicado en el boletín científico *Pädi* del Instituto de Ciencias Básicas e Ingenierías ICBI. Apoyados de una versión preliminar del prototipo para este proyecto, da cuenta de la importancia de la implementación de tecnologías basadas en Internet de las Cosas (IoT) en el ámbito escolar y destaca de manera relevante la oportunidad tangible para fortalecer la vigilancia de la salud estudiantil. En un momento en el que las tasas de sobrepeso, obesidad y desnutrición infantil siguen en aumento, herramientas de monitoreo como los sistemas IoT permiten identificar riesgos de manera temprana, favoreciendo intervenciones que impactan tanto en la salud física como en el rendimiento académico (Gómez-Gayosso, J. C., Suárez-Cansino, J., López-Morales, V., & Franco-Árcega, A., 2022). ***Para visualizar el resumen de la publicación, refiérase al Apéndice C.***

Reconociendo la importancia de actuar de forma preventiva, en la implementación práctica del presente proyecto, a manera de estudio, se buscó medir el Índice de Masa Corporal (IMC) de estudiantes de bachillerato mediante este sistema IoT, y analizar los datos obtenidos en función de estándares internacionales y nacionales, particularmente las categorías de riesgo establecidas por el Instituto de Seguridad y Servicios Sociales de los Trabajadores del Estado (ISSSTE, 2022).

5.1. Metodología y muestreo

La investigación llevada a cabo en el presente proyecto es de ***tipo clínica***, ya que se centra en la evaluación y mejora de aspectos relacionados con la salud humana, específicamente en el monitoreo del Índice de Masa Corporal (IMC) mediante tecnologías basadas en el Internet de las Cosas (IoT). Este enfoque permite abordar problemas reales del ámbito sanitario, orientados al bienestar físico y al seguimiento del estado nutricional de los individuos.

El tipo de estudio es de carácter ***experimental***, dado que se diseña, implementa y pone a prueba una estación de monitoreo con el propósito de observar los efectos y beneficios de su uso en un entorno controlado. A través de la intervención directa con dispositivos tecnológicos, se busca recolectar datos, evaluar resultados y establecer relaciones causales entre el uso de la solución propuesta y la mejora en el control del estado nutricional. Este diseño experimental permite validar la eficacia del sistema y su aplicabilidad en contextos clínicos y preventivos.

En cuanto al **enfoque metodológico**, la investigación es de naturaleza **cuantitativa**, puesto que se basa en la obtención, procesamiento y análisis de datos numéricos recolectados a través de sensores integrados en los dispositivos IoT. Variables como el peso corporal, la estatura y el IMC se registran de manera sistemática, lo que permite realizar un tratamiento estadístico riguroso para validar los resultados obtenidos. Este enfoque garantiza la objetividad, precisión y confiabilidad en la interpretación de los datos, y sustenta la formulación de conclusiones basadas en evidencia empírica.

Se realizó el monitoreo del IMC en una muestra de 80 estudiantes adolescentes, que cursan el primer semestre de bachillerato en la Escuela Preparatoria Número Dos de la UAEH, de edades comprendidas entre los 15 y 18 años. Utilizando el sistema IoT desarrollado en este proyecto. Se capturaron en tiempo real las mediciones de peso y estatura, automatizando el cálculo del IMC y su almacenamiento en una plataforma AWS en línea. La figura 61 muestra el diagrama de muestreo de la solución, y la figura 62 la aplicación de la configuración con 2 estudiantes.

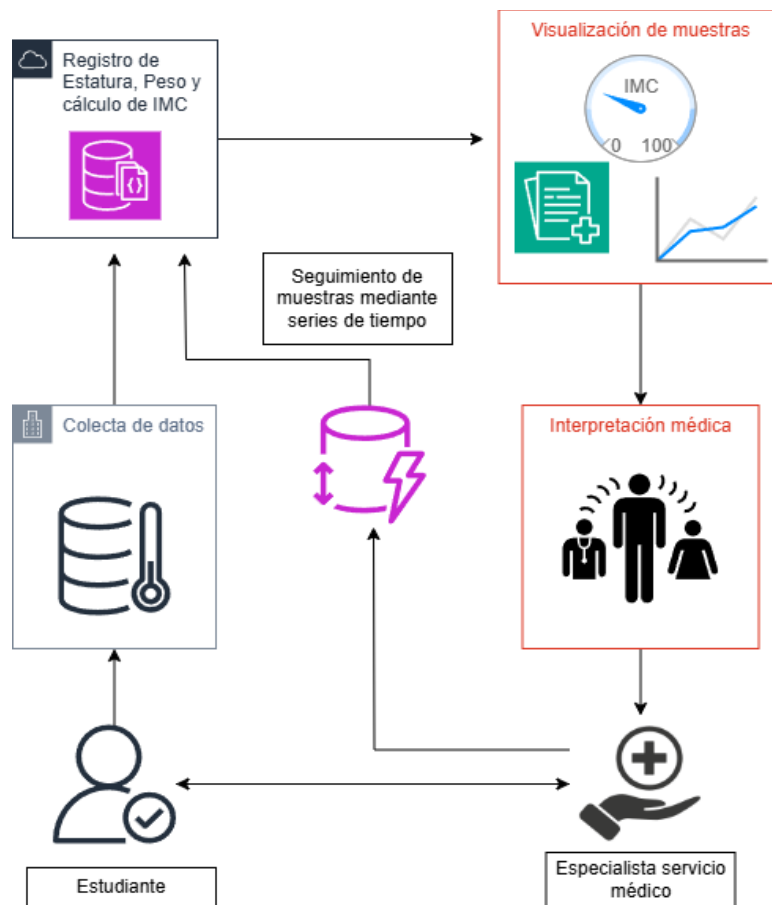


Figura 61. Diagrama de muestreo, almacenamiento y procesamiento de datos muestreados.
Fuente: Elaboración Propia.

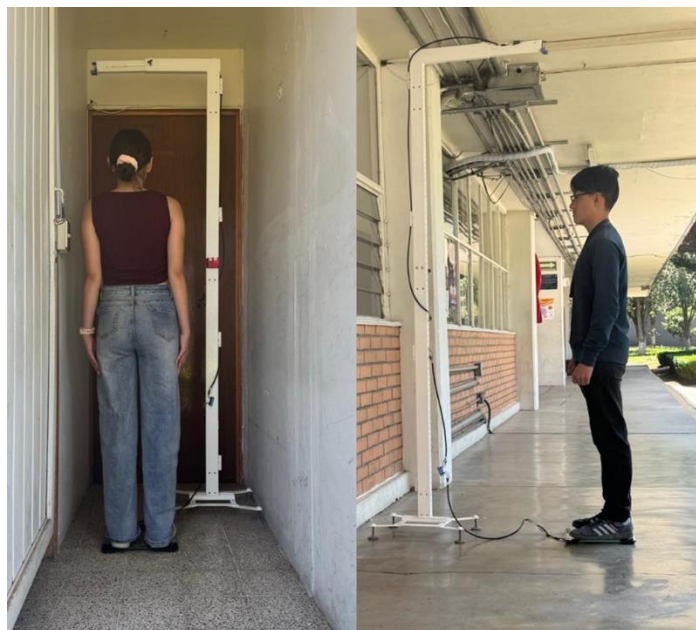


Figura 62. Aplicación de la configuración con 2 estudiantes de bachillerato. Fuente: Elaboración Propia.

La clasificación de los datos se basó en los rangos definidos por la Organización Mundial de la Salud (OMS, 2023), complementados con la tabla de riesgos de enfermedad por IMC publicada por el ISSSTE (2022). Posteriormente, se analizaron las tendencias y se interpretaron los resultados en relación con su impacto potencial en la salud y el desempeño académico de los estudiantes.

Descripción de los Datos de la muestra:

Se realizó una muestra por cada estudiante obteniendo 80 muestras en total

- **Número total de estudiantes:** 80
- **Edades intervalo de estudio:** 15 -18 años
- **Media de las edades de la muestra de estudiantes:** 16.5 años
- **Variables registradas:**
 - Peso (kg)
 - Altura (m)
 - IMC
 - Clasificación del IMC (Bajo peso, Peso normal, Sobrepeso, Obesidad.)

La tabla 18 contiene los datos muestreados con el correspondiente cálculo de IMC.

Tabla 18. Datos de la muestra e IMC calculado. Fuente: Elaboración propia

Estudiante	Peso (kg)	Altura (m)	IMC	Clasificación IMC	Estudiante	Peso (kg)	Altura (m)	IMC	Clasificación IMC
1	65.59971	1.845241	19.26618	Peso normal	41	51.7121	1.822976	15.56074	Bajo peso
2	97.28929	1.749319	31.79266	Obesidad	42	72.23473	1.858437	20.91465	Peso normal
3	85.25967	1.632359	31.99721	Obesidad	43	46.89137	1.627201	17.70966	Bajo peso
4	77.92622	1.525423	33.48905	Obesidad	44	95.01262	1.544021	39.85429	Obesidad
5	53.58103	1.624393	20.30621	Peso normal	45	59.2329	1.591174	23.39525	Peso normal
6	53.5797	1.630073	20.16443	Peso normal	46	81.43873	1.670843	29.17156	Sobrepeso
7	48.1946	1.791842	15.01062	Bajo peso	47	62.14411	1.827206	18.61337	Peso normal
8	92.63969	1.755023	30.07679	Obesidad	48	73.60374	1.844292	21.63916	Peso normal
9	78.06133	1.854885	22.6883	Peso normal	49	75.06907	1.502781	33.24066	Obesidad
10	83.94399	1.688886	29.42991	Sobrepeso	50	55.167	1.704299	18.99275	Peso normal
11	46.13215	1.547838	19.25541	Peso normal	51	98.32715	1.666964	35.38513	Obesidad
12	98.34504	1.785298	30.85539	Obesidad	52	87.63231	1.588843	34.7138	Obesidad
13	90.78435	1.804314	27.88603	Sobrepeso	53	96.67244	1.547946	40.34512	Obesidad
14	56.67865	1.724511	19.05845	Peso normal	54	94.2155	1.635046	35.24215	Obesidad
15	55.00037	1.808387	16.81833	Bajo peso	55	77.8845	1.877164	22.10277	Peso normal
16	55.08725	1.697518	19.11711	Peso normal	56	95.70308	1.629281	36.05237	Obesidad
17	61.73332	1.709093	21.13432	Peso normal	57	49.86709	1.707516	17.10347	Bajo peso
18	73.8616	1.671016	26.45193	Sobrepeso	58	55.77906	1.781208	17.58094	Bajo peso
19	68.75698	1.510168	30.14855	Obesidad	59	47.4875	1.645452	17.53917	Bajo peso
20	61.0176	1.543157	25.62331	Sobrepeso	60	62.89317	1.888713	17.63078	Bajo peso
21	78.65191	1.512572	34.37774	Obesidad	61	66.37725	1.884979	18.68126	Peso normal
22	52.67216	1.754564	17.10971	Bajo peso	62	59.9242	1.600713	23.38704	Peso normal
23	61.06796	1.625742	23.10521	Peso normal	63	90.58056	1.698899	31.38338	Obesidad
24	65.1499	1.703428	22.45257	Peso normal	64	64.62143	1.620351	24.61264	Peso normal
25	70.08385	1.863027	20.19202	Peso normal	65	60.4514	1.613936	23.20778	Peso normal
26	88.18468	1.599717	34.45933	Obesidad	66	74.84828	1.514755	32.62099	Obesidad
27	55.98206	1.664153	20.21447	Peso normal	67	52.75083	1.743826	17.34695	Bajo peso
28	73.28289	1.80222	22.56248	Peso normal	68	89.12083	1.701072	30.79882	Obesidad
29	77.5828	1.591519	30.62962	Obesidad	69	49.10029	1.520592	21.23532	Peso normal
30	47.55477	1.530792	20.29372	Peso normal	70	99.27878	1.611459	38.23122	Obesidad
31	78.41497	1.615901	30.03099	Obesidad	71	87.47346	1.863306	25.19462	Sobrepeso
32	54.37883	1.564489	22.21698	Peso normal	72	55.92936	1.595825	21.96188	Peso normal
33	48.57784	1.871879	13.86381	Bajo peso	73	45.30372	1.557958	18.66476	Peso normal
34	97.1887	1.823248	29.23642	Sobrepeso	74	89.85038	1.695781	31.24499	Obesidad
35	98.10976	1.753362	31.91312	Obesidad	75	83.87715	1.89426	23.37569	Peso normal
36	89.46185	1.848584	26.17938	Sobrepeso	76	85.09539	1.596822	33.37283	Obesidad
37	61.75376	1.821469	18.61315	Peso normal	77	87.41987	1.768854	27.93998	Sobrepeso
38	50.37197	1.574628	20.31575	Peso normal	78	49.07246	1.804648	15.0679	Bajo peso
39	82.63282	1.857024	23.9617	Peso normal	79	64.71562	1.595055	25.43652	Sobrepeso
40	69.20839	1.715737	23.51026	Peso normal	80	51.3728	1.791287	16.01043	Bajo peso

5.2. Análisis de los resultados

El análisis de los datos reflejó una **media de IMC de 24.85**, de las 80 muestras tomadas de los participantes, ubicada en el límite superior del rango de peso normal. Sin embargo, la dispersión de los resultados evidenció problemáticas relevantes, como se resume en la Tabla 19.

Tabla 19. Análisis de datos. Fuente: Elaboración propia.

Categoría de IMC	Número de estudiantes	Porcentaje (%)	Riesgo de enfermedad según ISSSTE
Peso bajo (<18.5)	13	16.25%	Aumentado
Peso normal (18.5-24.99)	33	41.25%	Normal
Sobrepeso (25.0-29.99)	10	12.50%	Aumentado
Obesidad leve (30.0-34.99)	18	22.5%	Moderado
Obesidad media (35.0-39.99)	5	6.25%	Elevado
Obesidad mórbida (>40)	1	1.25%	Severo

La presencia de más del 40% de estudiantes fuera del rango de peso normal plantea un desafío significativo para las políticas de salud escolar. Particularmente, el hallazgo de casos de obesidad mórbida (1.25%) señala un riesgo severo que demanda atención inmediata (ISSSTE, 2022). La Tabla 20, muestra el grado de riesgo de acuerdo a la condición determinada por el IMC.

Tabla 20. Riesgos de enfermedad por IMC adaptada de ISSSTE (2022).

IMC	Condición	Riesgo de enfermedad
Menor a 18.9	peso bajo (Desnutrición)	Aumentado
18.50 a 24.99	peso normal	Normal
25.00 a 29.99	sobrepeso	Aumentado
30.00 a 34.99	obesidad leve	Moderado
35.00 a 39.99	obesidad media	Elevado
Mayor a 40.0	obesidad mórbida	Severo

Para visualizar la evolución y distribución de los datos de IMC, se construyó una gráfica ilustrada en la Figura 63, la cual permite observar las variaciones individuales de los estudiantes a lo largo del proceso de medición.

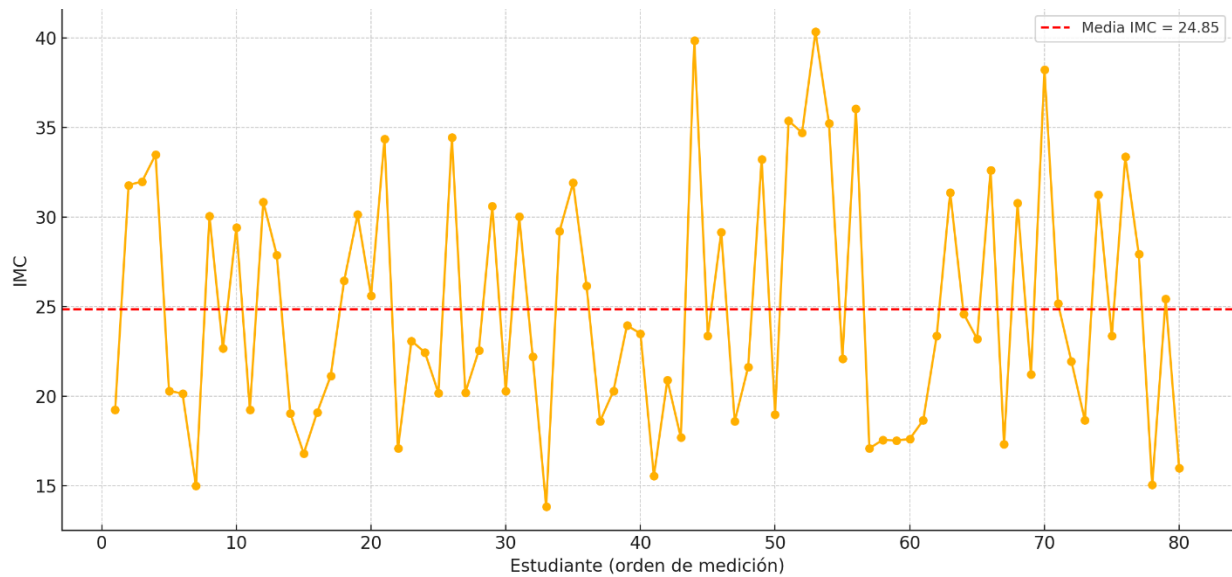


Figura 63. IMC registrado de 80 estudiantes. Fuente: Elaboración propia.

Descripción:

- Cada punto representa el IMC de un estudiante medido en orden consecutivo.
- La línea roja discontinua indica el valor promedio del IMC de la muestra (24.85).
- Se observan fluctuaciones considerables, reflejando casos de bajo peso, sobrepeso y obesidad.

La figura 64 muestra la distribución de IMC (Índice de Masa Corporal) de las 80 muestras tomadas.

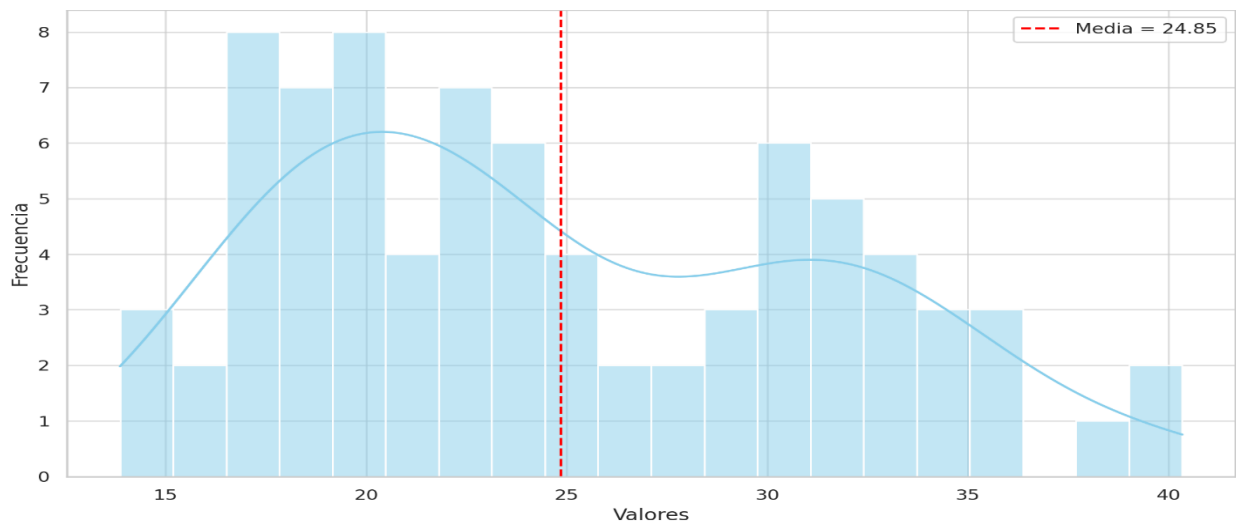


Figura 64. Distribución de los datos de IMC de 80 estudiantes Fuente: Elaboración propia.

Análisis de la distribución:

- **Desviación estándar:** 6.68
- **Prueba de normalidad** (Shapiro-Wilk):
 - **Estadístico W:** 0.9480
 - **Valor p:** 0.0027
- El histograma con la curva de densidad muestra una distribución asimétrica, con una ligera cola hacia la derecha.
- El valor p de la prueba de Shapiro-Wilk es menor a 0.05, lo que indica que **los datos no siguen una distribución normal** de forma estadísticamente significativa.

5.3. Apreciación crítica de los resultados

Al analizar los datos de la gráfica en la Figura 65, se observa que 16.25% de los estudiantes presentan peso bajo, situación que, de acuerdo con la Organización Mundial de la Salud (OMS, 2023), puede derivar en deficiencias nutricionales severas y afectar el desarrollo académico y emocional de los adolescentes.

Por otro lado, la suma de sobrepeso y obesidad alcanza el 42.5% de la muestra, lo cual coincide con lo reportado en otros estudios nacionales e internacionales que asocian estas condiciones con un incremento de enfermedades metabólicas, baja autoestima y disminución en el rendimiento escolar (PAHO, 2022).

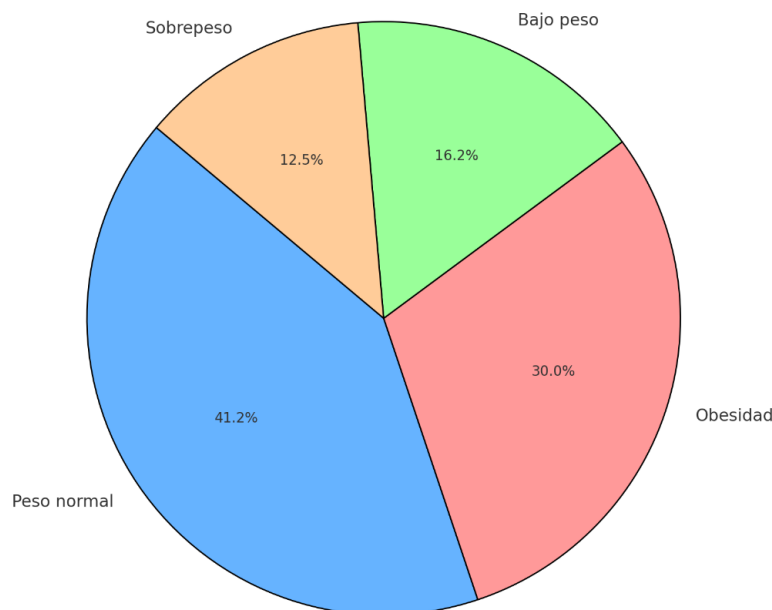


Figura 65. Distribución del IMC en estudiantes de Bachillerato. Fuente: Elaboración propia.

Descripción:

- La mayor proporción corresponde a estudiantes en peso normal (41.25%).
- Obesidad representa un 30% de la muestra.
- Bajo peso y sobrepeso suman el resto, indicando la necesidad de intervención.

5.4. Impacto del Sistema IoT en la Gestión de la Salud Escolar

El sistema IoT implementado no solo permitió una recolección eficiente y automatizada de datos, sino que también facilitó el análisis en tiempo real, alertando a los responsables escolares sobre posibles factores de riesgo de manera anticipada. Esta capacidad de respuesta inmediata, resulta crucial para establecer programas de intervención dirigidos y oportunos.

Los resultados obtenidos a través del sistema IoT evidencian que la implementación de tecnologías de monitoreo es una estrategia eficaz para la prevención de riesgos asociados a la malnutrición en estudiantes de bachillerato. Detectar de manera temprana alteraciones en el IMC permite diseñar estrategias de intervención más personalizadas, con el objetivo de mejorar tanto la salud integral como el rendimiento académico de los estudiantes.

5.5. Limitaciones del estudio

Si bien la implementación del sistema IoT para el monitoreo del Índice de Masa Corporal (IMC) en estudiantes de nivel medio superior ha demostrado ser una herramienta innovadora y efectiva, es importante reconocer una serie de limitaciones que condicionaron el alcance y la generalización de los resultados obtenidos.

En primer lugar, la muestra utilizada en el presente estudio fue limitada en tamaño y geográficamente circunscrita a una institución educativa específica, lo cual restringe la extrapolación de los hallazgos a otras poblaciones estudiantiles con diferentes contextos socioeconómicos, culturales o alimentarios. Para validar la efectividad del sistema a gran escala, sería necesario replicar el estudio en múltiples escuelas con diversidad regional y demográfica.

En segundo lugar, el sistema de medición automatizada dependió del correcto funcionamiento de los sensores IoT y de la conectividad a Internet, lo que implicó ciertos desafíos técnicos. En algunas ocasiones se presentaron problemas de calibración de los dispositivos de medición, así como interrupciones temporales en la transmisión de datos, lo cual pudo haber afectado la precisión o la completitud de ciertos registros. Estas fallas, aunque mínimas, subrayan la

importancia de realizar mantenimientos periódicos y contar con protocolos de verificación manual como respaldo.

Una tercera limitación está relacionada con el uso exclusivo del IMC como indicador nutricional. Si bien este índice es ampliamente utilizado por su practicidad, no considera directamente aspectos como la masa muscular, la distribución de grasa corporal o el estado de hidratación, lo cual podría conducir a interpretaciones imprecisas en algunos casos. La inclusión de otros indicadores complementarios, como el porcentaje de grasa corporal o medidas de circunferencia, permitiría una evaluación más holística del estado nutricional.

Asimismo, la temporalidad del estudio fue relativamente corta para establecer conclusiones robustas sobre tendencias longitudinales. Aunque se logró iniciar una base de datos histórica, el período de observación no fue lo suficientemente extenso como para identificar patrones sostenidos de cambio en el estado nutricional. Estudios futuros de mayor duración permitirían aprovechar completamente el potencial del sistema para el análisis por series temporales.

Finalmente, se debe señalar que el enfoque cuantitativo, si bien útil para el diagnóstico y segmentación de datos, no permitió explorar en profundidad los factores psicosociales, conductuales o familiares que inciden en el estado nutricional de los estudiantes. Para una comprensión más integral del problema, es pertinente complementar el análisis con estudios cualitativos que incluyan entrevistas, encuestas o grupos focales con los estudiantes y sus familias.

Aunque el sistema IoT propuesto ofrece una plataforma sólida y prometedora para el monitoreo del IMC, la presente investigación reconoce sus limitaciones técnicas, metodológicas y contextuales, las cuales deberán ser consideradas en futuras investigaciones e implementaciones a mayor escala.

5.6. Recomendaciones

A partir de los hallazgos obtenidos y las limitaciones identificadas en el presente estudio, se proponen las siguientes recomendaciones orientadas tanto a futuras investigaciones como a la implementación práctica del sistema IoT para el monitoreo del Índice de Masa Corporal (IMC) en entornos escolares:

1. **Ampliar la muestra y diversificar el contexto:** Se recomienda replicar el estudio con una muestra más amplia y representativa que incluya estudiantes de distintos niveles educativos, regiones geográficas y contextos socioculturales. Esto permitiría validar la eficacia del sistema IoT en distintos entornos y fortalecer la generalización de los resultados.
2. **Incorporar más variables de evaluación nutricional:** Para obtener un perfil más completo del estado nutricional de los estudiantes, es conveniente integrar otros indicadores como el porcentaje de grasa corporal, el índice cintura-estatura, la frecuencia cardíaca en reposo o incluso biomarcadores asociados a deficiencias nutricionales. Esta ampliación fortalecería el valor diagnóstico del sistema.
3. **Asegurar la sostenibilidad tecnológica:** Se sugiere invertir en infraestructura tecnológica adecuada y estable, así como en personal capacitado en el uso y mantenimiento del sistema. Esto garantizaría la continuidad operativa del monitoreo y la confiabilidad de los datos recolectados.
4. **Prolongar el seguimiento longitudinal:** Es fundamental extender el periodo de observación a varios ciclos escolares para analizar la evolución del estado nutricional en el tiempo y evaluar el impacto de estrategias de intervención institucionales o comunitarias.
5. **Articular el sistema con políticas educativas y de salud escolar:** Se recomienda integrar formalmente el uso del sistema IoT en los programas de salud escolar, de manera que su uso no quede limitado a iniciativas aisladas, sino que forme parte de una estrategia estructural y sostenida de vigilancia nutricional y promoción de la salud.
6. **Fortalecer la alfabetización digital y nutricional:** Se sugiere acompañar la implementación del sistema con programas educativos dirigidos tanto a estudiantes como a docentes, orientados a fomentar hábitos saludables y el uso responsable de la tecnología para el autocuidado.
7. **Establecer protocolos de actuación ante alertas nutricionales:** Finalmente, es esencial definir rutas de acción claras cuando se detecten desviaciones en el IMC. Esto implica coordinar la atención médica, el acompañamiento psicológico y las intervenciones nutricionales, garantizando una respuesta integral y oportuna.

Capítulo 6 Conclusiones y trabajo futuro

6.1. Conclusiones

La implementación de un sistema basado en tecnologías del Internet de las Cosas (IoT) para el monitoreo del Índice de Masa Corporal (IMC) en estudiantes de bachillerato representa una innovación significativa en el abordaje preventivo y personalizado de la salud escolar. A partir del análisis de los datos recolectados durante el presente estudio, se estableció un perfil cuantitativo que permite comprender con mayor precisión el estado nutricional de la población estudiada y el impacto potencial de la tecnología implementada.

Los valores registrados indican una media de IMC de 24.85, ubicándose al límite del rango saludable. Esta cifra, en combinación con una desviación estándar de 6.68, sugiere la existencia de una dispersión significativa en los datos, reflejando una amplia variabilidad en los estados nutricionales de los estudiantes. Algunos individuos presentan condiciones compatibles con desnutrición, mientras que otros superan los umbrales del sobrepeso e incluso obesidad. Esta heterogeneidad confirma la necesidad de un monitoreo sistemático y continuo, capaz de adaptar intervenciones según cada caso específico.

Adicionalmente, la prueba de normalidad de Shapiro-Wilk ($p = 0.0027$) proporciona evidencia de que los datos no siguen una distribución normal, como resultado se obtuvo una distribución bimodal. Esta desviación estadísticamente significativa indica la presencia de subgrupos con características particulares que no pueden ser explicadas bajo un modelo estándar. Este hallazgo refuerza la utilidad del sistema IoT, ya que permite detectar desviaciones no evidentes a simple vista, brindando alertas tempranas y facilitando la toma de decisiones informadas.

Desde una perspectiva longitudinal, el sistema no solo posibilita la observación de valores puntuales de IMC, sino que también construye una base de datos histórica que documenta la evolución del estado nutricional de los estudiantes a lo largo del tiempo. Esta característica resulta esencial para identificar patrones de riesgo sostenidos, evaluar la efectividad de programas nutricionales, y anticipar problemas de salud crónicos derivados de estilos de vida poco saludables.

Además del valor clínico, el sistema proporciona una ventaja estratégica a nivel institucional. Al digitalizar y automatizar la recolección de datos antropométricos (peso y estatura), se mejora la eficiencia operativa y se reduce la carga administrativa del personal docente y médico. La disponibilidad inmediata de información en la nube facilita la generación de reportes individuales y grupales, promoviendo una respuesta ágil ante posibles desviaciones.

En términos pedagógicos, los beneficios del sistema se extienden al ámbito académico. Una mejor salud física se asocia con una mayor asistencia, mejor concentración y, por ende, mejor rendimiento escolar. Asimismo, la participación activa del alumnado en el monitoreo de su propio estado de salud contribuye al desarrollo de una conciencia corporal más informada y responsable, promoviendo la adopción de hábitos saludables desde edades tempranas.

Finalmente, cabe destacar que la integración de un sistema IoT para el seguimiento del IMC no debe entenderse únicamente como una solución tecnológica, sino como una estrategia de salud pública preventiva, orientada a transformar la manera en que se gestiona la salud estudiantil. Esta herramienta empodera a los distintos actores escolares desde el personal médico hasta los directivos y docentes para construir entornos educativos más saludables, resilientes y sostenibles.

6.2. Trabajo futuro

A partir de la implementación inicial del sistema IoT para el monitoreo del IMC y la detección temprana de alteraciones nutricionales en estudiantes de bachillerato, se abre un amplio horizonte de desarrollo técnico, científico y social. El siguiente paso natural en la evolución de este sistema consiste en integrar algoritmos de **machine learning** y modelos de **diagnóstico predictivo**, que permitan no solo registrar y alertar sobre estados actuales, sino también anticiparse a riesgos futuros con base en patrones históricos y características individuales.

El uso de técnicas de **ciencia de datos** permitirá analizar volúmenes cada vez mayores de información recopilada a lo largo del tiempo, tanto a nivel individual como colectivo, y detectar correlaciones que no siempre resultan evidentes a simple vista. Estos modelos, al ser entrenados con datos reales, podrán ofrecer evaluaciones más precisas, ajustadas a la realidad de contextos específicos como el entorno escolar, deportivo o clínico, generando una medicina más personalizada y preventiva. Con el tiempo, la calidad y riqueza de estos datos permitirá desarrollar perfiles de riesgo altamente fiables, útiles para orientar campañas de intervención nutricional o médica desde edades tempranas.

Además, se contempla la evolución del sistema hacia una solución comercial orientada al sector salud y al consumidor general. Esto podría concretarse mediante el desarrollo de quioscos inteligentes de monitoreo, ubicados en espacios públicos, instituciones educativas, centros deportivos o clínicas, que permitan la medición rápida y no invasiva de signos vitales como frecuencia cardíaca, presión arterial, oxigenación, temperatura, entre otros, complementados con el cálculo automatizado del IMC. Estos dispositivos brindarían diagnósticos orientativos e inmediatos, así como recomendaciones personalizadas para acudir a servicios médicos especializados o iniciar tratamientos preventivos, conectando a los usuarios con redes de atención mediante plataformas digitales seguras.

En este contexto, la utilidad del sistema no se limitaría a la detección de sobrepeso o desnutrición, sino que podría extenderse hacia una vigilancia más amplia del estado de salud, siendo una herramienta clave en la transición hacia un modelo de salud predictiva, preventiva y centrada en el paciente. Esto incluye también su aplicación en la medicina del deporte, donde el monitoreo constante de parámetros fisiológicos y el análisis en tiempo real del rendimiento físico permitirían diseñar planes de entrenamiento adaptados, prevenir lesiones y optimizar la recuperación de atletas, tanto amateurs como profesionales.

En definitiva, el desarrollo futuro de este sistema apunta a consolidar una plataforma integral de salud digital que combine la precisión de la tecnología con un enfoque profundamente humano, centrado en la prevención, la educación en salud y el empoderamiento de las personas sobre su propio bienestar. Esta visión no sólo responde a las necesidades actuales, sino que anticipa un futuro en el que el acceso a herramientas de diagnóstico y orientación médica de calidad esté al alcance de todos, sin importar el nivel socioeconómico ni el contexto geográfico.

Referencias

- Aguilar-Medina, E. M., Osorio-Macías, M. L., & Delgado-Velázquez, J. A. (2023). *Prototipado Rápido con Tecnologías IoT para Aplicaciones Educativas*. *Revista Ingeniería*, 27, 45–59. <https://www.revista.ingenieria.uady.mx/ojs/index.php/ingenieria/article/view/345>
- Amazon Web Services. (2023). *Amazon Timestream Developer Guide*. Recuperado de <https://docs.aws.amazon.com/timestream/latest/developerguide/AWS Documentation+3AWS Documentation+3AWS Documentation+3>
- Amazon Web Services. (2023). *Overview of Amazon Web Services*. AWS Documentation. <https://docs.aws.amazon.com/whitepapers/latest/aws-overview/aws-overview.pdf>
- Amazon Web Services. (2023). *X.509 Certificates for AWS IoT*. <https://docs.aws.amazon.com/iot/latest/developerguide/x509-client-certs.html>
- Arango Astorga, P., & García García, Y. (2022). *Internet De Las Cosas En El Ámbito De La Atención Médica: Tendencias Y Desafíos*. *Revista Cubana de Informática Médica*, 14(1). Recuperado de <https://revinformatica.sld.cu/index.php/rcim/article/view/452>
- Ashok, J., Tallapragada, V. V. S., Padmaja, D. L., Sathish, E., Karthikeyan, S., & Kumar, J. P. (2023). *Modelling and Implementation of Microcontroller System Robotic Devices Through Digital Twins with the Modernization of Complex Systems*. *SN Computer Science*, 4(497). <https://doi.org/10.1007/s42979-023-01938-3>
- Atzori, L., Iera, A., & Morabito, G. (2010). *The Internet of Things: A survey*. *Computer Networks*, 54(15), 2787–2805. <https://doi.org/10.1016/j.comnet.2010.05.010>
- AWS. (2024). *AWS IoT Core*. Recuperado de <https://aws.amazon.com/iot-core/>
- Bagula M.F., Bagula H., Mandava M., Kakoko Lubamba C., Bagula A. (2019) *Cyber-Healthcare Kiosks for Healthcare Support in Developing Countries*. In: Mendy G., Ouya S., Dioum I., Thiaré O. (eds) *e-Infrastructure and e-Services for Developing Countries*. AFRICOMM 2018. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 275. Springer, Cham. https://link.springer.com/chapter/10.1007/978-3-030-16042-5_18
- Barrett, S. F., & Pack, D. J. (2020). *Embedded Systems: Design and Applications with the 68HC12 and HCS12*. Pearson.

- Bhushan, B., Sahoo, P., & Satapathy, S. (2021). *Internet of Things in Healthcare: Theory and Applications*. Springer.
- Brown Macheso, P. S., & Meela, A. G. (2021). *IoT Based Patient Health Monitoring using ESP8266 and Arduino*. International Journal of Computer Communication and Informatics, 3(2), 75-83. <https://doi.org/10.34256/ijcci2127>
- Cavefish. (2023). *ESP8266 NodeMCU (CH340) Case*. Thingiverse. <https://www.thingiverse.com/thing:5873652>
- Choudhary, A. (2024). *Internet of Things: A Comprehensive Overview, Architectures, Applications, Simulation Tools, Challenges, and Future Directions*. Discover Internet of Things, 4(31). <https://doi.org/10.1007/s43926-024-00084-3>
- Cook, G., Duncan, M., & Mitchell, R. (2020). *Design Thinking for Wearable Health Devices: A Human-Centered Approach to Fall Detection Systems*. International Journal of Medical Informatics, 140, 104150. <https://doi.org/10.1016/j.ijmedinf.2020.104150>
- Cortés-Castillo, L. C., Sánchez-Cruz, M. L., & Méndez-López, A. (2022). *Prevalencia De Malnutrición En Escolares: Una Realidad Dual En Hidalgo, México*. Revista del Instituto de Ciencias Básicas e Ingeniería, 10(2), 33–45. Recuperado de <https://repository.uaeh.edu.mx/revistas/index.php/atotonilco/article/download/5267/9469>
- Creative Commons. (2022). *About The Licenses*. <https://creativecommons.org/licenses/>
- De Moura Costa, H.J., da Costa, C.A., da Rosa Righi, R. et al. (2020). *Fog computing in health: A systematic literature review*. Health Technol. 10, 1025–1044 (2020). <https://link.springer.com/article/10.1007/s12553-020-00431-8#citeas>
- Elhemali, M., Gallagher, N., Gordon, N., Idziorek, J., Krog, R., Lazier, C., ... & Vig, A. (2022). *Amazon DynamoDB: A Scalable, Predictably Performant, and Fully Managed NoSQL Database Service*. Proceedings of the 2022 USENIX Annual Technical Conference. Recuperado de <https://www.usenix.org/conference/atc22/presentation/elhemali>
- EMQ. (2024). *AWS IoT Core vs EMQX Platform: A Practical Guide for IoT Solution Selection*. Recuperado de <https://www.emqx.com/en/resources/aws-iot-core-vs-emqx-platform-guide>

- ENISA. (2021). *Algorithms, Key Size And Parameters Report – 2021 Recommendations*. European Union Agency for Cybersecurity. <https://www.enisa.europa.eu/publications>
- Espressif Systems. (2024). *SPIFFS — ESP-IDF Programming Guide*. <https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/storage/spiffs.html>
- Espressif Systems. (s.f.-a). *ESP8266 Wi-Fi Modules*. Recuperado el 18 de abril de 2025, de <https://www.espressif.com/en/products/modules/esp8266/>
- Espressif Systems. (s.f.-b). *ESP32 Wi-Fi & Bluetooth SoC*. Recuperado el 18 de abril de 2025, de <https://www.espressif.com/en/products/socs/esp32>
- Farhan, M., Majeed, A., & Rashid, M. (2021). *Wi-Fi HaLow: The Future of IoT*. IEEE Internet of Things Magazine, 4(2), 30–35. <https://doi.org/10.1109/IOTM.001.2100034>
- Fernández-Medina, E., Rodríguez, A., & Trujillo, J. (2022). *Seguridad En El Desarrollo De Software Seguro*. Editorial Académica Española.
- Fielding, R. T., & Taylor, R. N. (2002). *Principled Design Of The Modern Web Architecture*. ACM Transactions on Internet Technology (TOIT), 2(2), 115–150. <https://doi.org/10.1145/514183.514185>
- Flores-Ramos, I., García, M. A., & López-Ruiz, M. (2020). *Obesidad y Rendimiento Académico En Adolescentes: Una Revisión Sistemática*. Revista Mexicana de Psicología, 37(1), 45–52.
- G2. (2024). *Compare AWS IoT Core vs. EMQX*. Recuperado de <https://www.g2.com/compare/aws-iot-core-vs-emqx>
- García, R., & Muñoz, A. (2019). *IoT in Optometry: Potential and Application*. Springer.
- Gavrilov, D., et al. (2022). *Using IoT Protocols in Real-Time Systems: Protocol Analysis and Evaluation of Data Transmission Characteristics*. Journal of Computer Networks and Communications. <https://doi.org/10.1155/2022/7368691>
- Ghosh, A., Adinyira, E., Agyekum, K., & Poku-Boansi, M. (2022). *Emerging industry 4.0 and Internet of Things (IoT) technologies in the Ghanaian construction industry: sustainability, implementation challenges, and benefits*. Environmental Science and Pollution Research. <https://doi.org/10.1007/s11356-022-24764-1>

- Gómez-Gayosso, J. C., Suárez-Cansino, J., López-Morales, V., & Franco-Árcega, A. (2022). *Sistema IOT Y Validación Estadística Para Monitoreo De Salud De Estudiantes*. *Pädi Boletín Científico De Ciencias Básicas E Ingenierías Del ICBI*, 10(Especial3), 103-111. <https://doi.org/10.29057/icbi.v10iEspecial3.9004>
- González-Granadillo, G., & Alcaraz, C. (2020). *Cybersecurity monitoring for critical infrastructures*. Springer.
- González-Macías, L., Ortega-Castillo, L., & Pérez-Rojas, R. (2022). *Desórdenes Alimenticios En Estudiantes De Bachillerato: Señales De Alerta Y Estrategias De Prevención*. *Revista Latinoamericana de Salud Escolar*, 14(2), 89–97.
- Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). *Internet Of Things (IoT): A Vision, Architectural Elements, And Future Directions*. *Future Generation Computer Systems*, 29(7), 1645–1660. <https://doi.org/10.1016/j.future.2013.01.010>
- Guerrero-Ulloa, G., Rodríguez-Domínguez, C., & Hornos, M. J. (2023). *Agile Methodologies Applied to the Development of IoT-Based Systems: A Review*. *Sensors*, 23(2), 790. <https://doi.org/10.3390/s23020790>
- Gupta, D. Akansksha, P. Swarnalatha, R. (2020). *Integrated healthcare monitoring device for obese adults using internet of things (IoT)* Keywords: Blood pressure Heart rate Internet of things SpO2 Storage of medical data Temperature International Journal of Electrical and Computer Engineering (IJECE), 2020 https://www.academia.edu/43719434/Integrated_healthcare_monitoring_device_for_obese_adults_using_internet_of_things_IoT_Keywords_Blood_pressure_Heart_rate_Internet_of_things_SpO2_Storage_of_medical_data_Temperature?auto=citations&from=cover_page
- He, Y., He, J., & Wen, N. (2023). *The Challenges Of IoT-Based Applications In High-Risk Environments*, health and safety industries in the Industry 4.0 era using decision-making approach. *Journal of Innovation & Knowledge*, 8(2). <https://doi.org/10.1016/j.jik.2023.100347>
- Hernández-Díaz, S., Roldán-Cruz, D. F., & López-Pérez, R. (2014). *Estado Nutricional Y Riesgo De Obesidad En Escolares Del Nivel Básico En Hidalgo*. *UAEH Investigación y Ciencia*, 22(2), 45–51. Recuperado de <https://www.uaeh.edu.mx/investigacion/productos/5242>

- HiveMQ. (2023). *Introducing the MQTT Protocol – MQTT Essentials: Part 1*. Recuperado de <https://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt/>
- HiveMQ. (2024). *HiveMQ vs. AWS IoT Core: A Comparative Analysis for IoT Messaging*. Recuperado de <https://www.hivemq.com/blog/hivemq-vs-aws-iot-core-comparative-analysis-iot-messaging/>
- HiveMQ. (2024). *The Trusted Enterprise-Grade MQTT Broker*. Recuperado de <https://www.hivemq.com/products/mqtt-broker/>
- Hong-tan, L., Cui-hua, K., Muthu, B., & Sivaparthipan, C. B. (2021). *Big Data And Ambient Intelligence In Iot-Based Wireless Student Health Monitoring System*. Aggression and Violent Behavior, 101601. doi:10.1016/j.avb.2021.101601
<https://www.sciencedirect.com/science/article/abs/pii/S1359178921000550>
- Hornos, M. J., & Quinde, M. (2024). *Development Methodologies For Iot-Based Systems: Challenges and research directions*. Springer Nature Switzerland AG. Recuperado de <https://www.researchgate.net/publication/382998653>
- IEEE Computer Society. (2024). *Are MQTT Brokers the Foundation of Future IoT Networks?*. Recuperado de <https://www.computer.org/publications/tech-news/trends/mqtt-brokers/>
- Instituto de Seguridad y Servicios Sociales de los Trabajadores del Estado (ISSSTE). (2022). *Tabla de clasificación del IMC y riesgos asociados*. Gobierno de México. Recuperado de <https://www.gob.mx/issste/articulos/que-es-el-indice-de-masa-corporal>
- Instituto de Seguridad y Servicios Sociales de los Trabajadores del Estado (2022), “*La Obesidad se puede prevenir*”, Gobierno de México, Blog, Recuperado de: <https://www.gob.mx/issste/articulos/la-obesidad-se-puede-prevenir>
- Jacob, S.; Menon, V. (2019) *MEDICO-A Simple IoT Integrated Medical Kiosk for the Rural People*. Preprints 2019, 2019030069 (doi: 10.20944/preprints201903.0069.v1). <https://www.preprints.org/manuscript/201903.0069/v1>
- Jiménez-Cruz, A., Bacardí-Gascón, M., & Castillo-Ruiz, O. (2019). *Obesidad En Adolescentes Mexicanos: Una Problemática De Salud Pública Y El Papel De Las Escuelas En Su Prevención*. Revista de Salud Pública y Nutrición, 18(2), 45–52.

- Jouhari, M., et al. (2022). *A Survey on Scalable LoRaWAN for Massive IoT: Recent Advances, Potentials, and Challenges*. arXiv preprint arXiv:2202.11082. <https://arxiv.org/abs/2202.11082>
- Jukes, M & McGuire, J. (2002). *“Nutrición, La Base Para el Desarrollo”*, Nutrición y Educación, Ginebra:SCN, United Nations System Standing Committee on Nutrition, Recuperado de: https://www.unscn.org/files/Publications/Briefs_on_Nutrition/Brief2_SP.pdf
- Kavedia, M., Bhambure, S. V., & Shetye, S. S. (2020). *Arduino a Development Tools for Embedded System and IOT*. ResearchGate. https://www.researchgate.net/publication/357203278_Arduino_a_Development_Tools_for_Embedded_System_and_IOT
- Kumar, A., & Gupta, R. (2021). *Model-Based Development in IoT: A Comprehensive Review*. Journal of Intelligent Manufacturing, 32(6), 1797–1810. <https://doi.org/10.1007/s10845-021-01889-x>
- Kumar, A., Gupta, D. N., & Kumar, R. (2023). *Integration of Big Data and IoT in the Modern Era. En N. Sindhwani, R. Anand, M. Niranjanamurthy, D. C. Verma, & E. B. Valentina (Eds.), IoT Based Smart Applications* (pp. 167-229). Springer, Cham. https://doi.org/10.1007/978-3-031-04524-0_8
- Kumar, N. (2018). *Health Kiosk Using IOT, Electronics and Communication Engineering, National Institute of Technology, Durgapur India*, nkp.20150077@btech.nitdgp.ac.in, <https://innovate.mygov.in/wp-content/uploads/2018/09/mygov1535788439415380.pdf>
- Kumari, S., & Kumar, S. (2021). *Design And Analysis Of Iot Based Smart Systems For Automation And Monitoring*. Sensors and Actuators A: Physical, 320, 112510. <https://doi.org/10.1016/j.sna.2021.112510>
- Lampertius T. (2019). *Adafruit VL53L0X Case*. Thingiverse. <https://www.thingiverse.com/thing:3434410>
- León De la Cruz, J. Y., Carhuajulca Saavedra, L. F., Jaramillo Vega, M. A., Torres Fabian, S. F., García Ramos, W., & Álvarez Huari, M. Y. (2023). *Impacto de la nutrición en el rendimiento académico: Una revisión narrativa*. Ciencia Latina, 7(5). https://doi.org/10.37811/cl_rcm.v7i5.8476

- León Torres-Restrepo, L., Martínez-Rebollar, A., González-Mendoza, M., Estrada-Esquivel, H., & Vargas-Agudelo, F. (2020). *Method for introducing IoT project development using free software tools*. *Research in Computing Science*, 149(12), 57–64. https://rcs.cic.ipn.mx/2020_149_12/Method%20for%20Introducing%20IoT%20Project%20Development%20Using%20Free%20Software%20Tools.pdf
- Longo, E., & Redondi, A. E. C. (2023). *Design And Implementation Of An Advanced MQTT Broker For Distributed Pub/Sub Scenarios*. *Computer Networks*, 224, 109601. <https://doi.org/10.1016/j.comnet.2023.109601>
- Luis-García, L. C., & Torres Gómez, A. R. (2024). *Desarrollo de aplicaciones IoT: metodologías y estrategias*. *European Public & Social Innovation Review*, 9, Artículo 1375. <https://doi.org/10.31637/epsir-2024-1375>
- Mae, J., Oey, E., & Kristiady, F. S. (2020). *IoT based body weight tracking system for obese adults in Indonesia using realtime database*. *IOP Conference Series: Earth and Environmental Science*, 426, 012143. <https://doi.org/10.1088/1755-1315/426/1/012143>
- Martínez-Torres, J., Rodríguez-López, A., & Castillo-Rivera, M. (2021). *Aplicación Del Internet De Las Cosas En El Monitoreo De Salud Escolar: una revisión de casos*. *Tecnología y Sociedad*, 8(3), 60–74.
- Mikhaylov, K., Plevritakis, N., & Tervonen, J. (2013). *Performance Analysis And Comparison Of Low-Power Wireless Technologies For Medical Applications*. *EURASIP Journal on Wireless Communications and Networking*, 2013(1), 1–15. <https://doi.org/10.1186/1687-1499-2013-61>
- Miller, R. (2022). *Cloud Giants: How AWS Became A \$100 Billion Business*. TechCrunch. <https://techcrunch.com/2022/04/12/aws-100-billion-business/>
- Mittapally, S. K., & Karra, R. C. (2023). *Functions And Performance Of Sensors For Slope Monitoring In Opencast Coal Mines – Laboratory Experimentation*. *Petroleum Science and Technology*. https://taylorandfrancis.com/knowledge/Engineering_and_technology/Computer_science/ESP8266
- Mohamad, M., Soin, N., & Ibrahim, F. (2020). *Effect of Different Wheatstone Bridge Configurations on Sensitivity and Linearity of MEMS Piezoresistive Intracranial Pressure Sensors*. *Journal*

- of Electronic Voltage and Application, 1(2), 14-19.
<https://publisher.uthm.edu.my/ojs/index.php/jeva/article/view/7078>
- Mohan Tukade, T., & Banakar, R. (2020). *Comparative Study of IoT Protocols*. PJAE, 17(7).
https://www.researchgate.net/publication/348279462_COMPARATIVE_STUDY_OF_IOT_PROTOCOLS_PJAE_17_7_2020_COMPARATIVE_STUDY_OF_IOT_PROTOCOLS
- Mohan, P., & Al-Khatib, K. (2020). *Evaluating Traditional Software Development Models in IoT: Insights from Waterfall Applications*. Journal of Software Engineering and Applications, 13(4), 187–200. <https://doi.org/10.4236/jsea.2020.134010>
- Muzaffar, S., & Elfadel, I. A. M. (2022). *Introduction*. In *Secure, Low-Power IoT Communication Using Edge-Coded Signaling* (pp. 1–10). Springer, Cham. https://doi.org/10.1007/978-3-030-95914-2_1
- Nakamura, K., Kuroda, K., & Mamiya, Y. (2019). *Ultrasonic Sensors and Their Applications*. Springer.
- Oracle. (2023). *Java Message Service (JMS) API*. Recuperado de <https://docs.oracle.com/javaee/7/tutorial/jms.htm>
- Organización Mundial de la Salud (OMS). (2023). *Obesidad Infantil Y Adolescente*. Recuperado de <https://www.who.int/es/news-room/fact-sheets/detail/obesity-and-overweight>
- Organización Mundial de la Salud. (2017). *El Doble Reto De La Malnutrición: Priorizar Las Políticas De Nutrición Para Combatir La Desnutrición Y El Sobrepeso Al Mismo Tiempo*. OMS.
- Organización Mundial de la Salud. (2022). *Fortalecer Las Políticas Y Programas Para La Nutrición En La Infancia Y Adolescencia: Informe Técnico*. OMS. <https://iris.who.int/bitstream/handle/10665/362047/9789240054547-spa.pdf>
- OWASP. (2023). *OWASP Top Ten Web Application Security Risks*. Open Web Application Security Project. <https://owasp.org/www-project-top-ten/>
- Pan American Health Organization (PAHO). (2022). *Schools As Settings For Promoting Health And Well-Being: Health promoting schools initiative*. Recuperado de <https://www.paho.org/en/documents/schools-settings-promoting-health-and-well-being-health-promoting-schools-initiative>

- ProfHankD. (2021). *HX711 Module Case*. Thingiverse.
<https://www.thingiverse.com/thing:4975809>
- Rahman, M., & Ahmed, A. (2022). *Efficient Iot System Design Using MQTT Protocol And ESP8266 For Remote Monitoring*. *Internet Of Things*, 14, 100426.
<https://doi.org/10.1016/j.iot.2021.100426>
- Saint-Andre, P. (2011). *Extensible Messaging and Presence Protocol (XMPP): Core*. IETF.
<https://tools.ietf.org/html/rfc6120>
- Schmidt, D. C., Corsaro, A., & Corman, D. (2018). *Data Distribution Service (DDS) for Real-Time Systems*. Object Management Group. <https://www.omg.org/spec/DDS>
- Secretaría de Salud y Secretaría de Educación Pública (2007), “Programa De Acción Específico 2007-2012”, Escuela y Salud, p. 9, Plan educativo Nacional, UNAM, Recuperado de:
https://www.planeducativonacional.unam.mx/CAP_12/Text/12_05a.html
- Serpanos, D., & Wolf, M. (2018). *Internet-of-Things (IoT) Systems: Architectures, Algorithms, Methodologies*. Springer. <https://doi.org/10.1007/978-3-319-69715-4>
- Sicari, S., Rizzardi, A., Grieco, L. A., & Coen-Porisini, A. (2015). *Security, Privacy And Trust in Internet of Things: The road ahead*. *Computer Networks*, 76, 146–164.
<https://doi.org/10.1016/j.comnet.2014.11.008>
- Silva, B. N., Khan, M., & Han, K. (2020). *Towards Sustainable Smart Cities: A Review Of Trends, Architectures, Components And Open Challenges In Smart Cities*. *Sustainable Cities and Society*, 61, 102408. <https://doi.org/10.1016/j.scs.2020.102408>
- Singh, A. (2019). *The IoT Applications and Consideration for NoSQL Database*. SSRN. Recuperado de https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3399649
[SSRN+1SSRN+1](https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3399649)
- Smith, J. (2021). *The Risks Of Cloud Dependence: A Critical Look At AWS*. *Journal of Cloud Computing*, 9(2), 45-58. <https://doi.org/10.1007/s13677-021-00247-y>
- Smith, J., & Lee, H. (2021). *DevOps Practices in IoT Deployment: Challenges and Solutions*. *IEEE Internet of Things Journal*, 8(10), 8256–8270. <https://doi.org/10.1109/JIOT.2021.3070692>
- Souri, Alireza; Ghafour, Marwan Yassin; Ahmed, Aram Mahmood; Safara, Fatemeh; Yamini, Ali; Hoseyninezhad, Mahdi (2020). *A New Machine Learning-Based Healthcare Monitoring*

- Model For Students Condition Diagnosis In Internet Of Things Environment. Soft Computing*, (), -. doi:10.1007/s00500-020-05003-6
<https://link.springer.com/article/10.1007/s00500-020-05003-6>
- SparkFun Electronics. (2023). *SparkFun HX711 Calibration [Código fuente]*. GitHub.
[https://github.com/sparkfun/HX711-Load-Cell-Amplifier/blob/master/firmware/SparkFun HX711 Calibration/SparkFun HX711 Calibrati on.ino](https://github.com/sparkfun/HX711-Load-Cell-Amplifier/blob/master/firmware/SparkFun%20HX711%20Calibration/SparkFun%20HX711%20Calibration.ino)
- SparkFun Electronics. (s.f.). *SparkFun Load Sensor Combinator v1.1 [Ficha técnica]*.
[https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/SparkFun%20Load%20Sensor %20Combinator%20v1.1.pdf](https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/SparkFun%20Load%20Sensor%20Combinator%20v1.1.pdf)
- Srinivas, V., et al. (2021). *Performance Evaluation Of MQTT Protocol For Iot Applications In Constrained Networks*. Journal of Computer Networks and Communications, 2021, 7084026. <https://doi.org/10.1155/2021/7084026>
- Steele, R., Lo, A., Secombe, C., & Wong, Y. K. (2012). *Elderly Persons' Perception and Acceptance of Using Wireless Sensor Networks to Assist Healthcare Services*. International Journal of Medical Informatics, 78(12), 788–801.
<https://doi.org/10.1016/j.ijmedinf.2009.08.001>
- STMicroelectronics. (2016). *VL53L0X Time-of-Flight Ranging Sensor*. Retrieved from
<https://www.st.com/resource/en/datasheet/vl53l0x.pdf>
- STMicroelectronics. (s.f.). *STM32 32-bit Arm Cortex MCUs*. Recuperado el 18 de abril de 2025, de <https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html>
- Sun-Chul, J., et al. (2022). *Comprehensive Exploration of IoT Communication Protocol: CoAP, MQTT, HTTP, LoRaWAN and AMQP*. Springer Professional.
<https://www.springerprofessional.de/comprehensive-exploration-of-iot-communication-protocol-coap-mqt/50201226>
- Suvini P. Amaraweera and Malka N. Halgamuge (2019). *Chapter 8 Internet of Things in the Healthcare Sector: Overview of Security and Privacy Issues*. In: Security, Privacy and Trust in the IoT Environment, Zaigham Mahmood Editor, Springer, pp. 153 – 179

- Tan, J., Yeo, C. H., & Lim, L. H. (2019). *Development of an IoT-Based Health Monitoring System for Student Wellness in Tropical Environments*. *Sensors*, 19(10), 2281. <https://doi.org/10.3390/s19102281>
- Thingiverse. (2024). *Free 3D Models For 3D Printing*. Retrieved April 23, 2025, from <https://www.thingiverse.com/>
- Thingiverse. (2024). *Thingiverse: Digital Designs for Physical Objects*. <https://www.thingiverse.com/>
- Ullah, F., Olatunji, O., & Qayyum, S. (2024). *A Scoping Review Of Green Internet Of Things In Construction And Smart Cities: Current Applications, Adoption Strategies And Future Directions*. *Smart and Sustainable Built Environment*. <https://doi.org/10.1108/SASBE-11-2023-0349>
- Universidad Autónoma del Estado de Hidalgo. (2022). *Aviso De Privacidad Integral*. <https://www.uaeh.edu.mx/aviso-privacidad-integral.html>
- Universidad Autónoma del Estado de Hidalgo. (2024). *Manual de Procedimientos de La Dirección de Servicio Médico Universitario*. https://www.uaeh.edu.mx/adminyserv/dir_generales/serv_estudiantiles/direccion_servuni_versitario/docs/docs_administrativos/2024/mp-direccion-servicio-medico-universitario.pdf
- Universidad Autónoma del Estado de Hidalgo. (s.f.). *Misión y visión de la UAEH*. <https://www.uaeh.edu.mx/excelencia/mision.htm>
- Vigilancia Tecnológica del Internet de las Cosas (IoT) aplicado en el sector salud. (2023). *Repositorio Universidad Distrital Francisco José de Caldas*. <https://repository.udistrital.edu.co/bitstreams/e6cb27af-a6eb-40d3-8ee2-f3a2cec35da8/download>
- Wang, K., et al. (2021). *TinyML Meets IoT: A Comprehensive Survey*. *Internet of Things*, 14, 100377. <https://doi.org/10.1016/j.iot.2021.100377>
- Wang, K., Zhao, X., & Lin, J. (2024). *TinyML Meets IoT: Recent Advances and Applications*. *Internet of Things*, 14, 100377. <https://doi.org/10.1016/j.iot.2024.100377>
- World Health Organization (WHO). (2021). *Obesity and overweight*. <https://www.who.int/news-room/fact-sheets/detail/obesity-and-overweight>

- Wuest, T., & Thoben, K. D. (2021). *Lean Manufacturing and the Internet of Things: Synergies and Challenges*. Computers in Industry, 129, 103464. <https://doi.org/10.1016/j.compind.2021.103464>
- Wytrębowicz, J., Cabaj, K., & Krawiec, J. (2021). *Messaging Protocols for IoT Systems—A Pragmatic Comparison*. Sensors, 21(20), 6904. <https://doi.org/10.3390/s21206904>
- Yadav, H. (2024). *Structuring SQL/NoSQL Databases for IoT Data*. International Journal of Machine Learning and Artificial Intelligence, 5(5), 1-12. Recuperado de <https://ijmlai.in/index.php/ijmlai/article/view/27>
- Yick, J., Mukherjee, B., & Ghosal, D. (2008). *Wireless Sensor Network Survey*. Computer Networks, 52(12), 2292–2330. <https://doi.org/10.1016/j.comnet.2008.04.002>
- Zeadally, S., Mbakwe, J., & Bello, O. (2020). *Wireless Technologies For Iot: Performance, Challenges, And Trends*. International Journal of Communication Systems, 33(9), e4374. <https://doi.org/10.1002/dac.4374>
- Zhang, Y., & Wang, J. (2019). *Design And Implementation Of A Low-Power Microcontroller-Based Iot Device*. International Journal of Embedded Systems, 11(2), 127–135. <https://doi.org/10.1504/IJES.2019.097261>
- Zhao, Z., Chen, W., Wang, C., & Wang, J. (2020). *A Review of Lidar Technology And Its Applications In Unmanned Aerial Vehicles*. In Remote Sensing, 12(2), 1–20. <https://doi.org/10.3390/rs12020282>
- Zhou, Y., & Li, F. (2022). *Rapid Prototyping for IoT Application Development: A Case Study Approach*. Journal of Systems and Software, 182, 111123. <https://doi.org/10.1016/j.jss.2022.111123>

Glosario

A

- **AMQP (Advanced Message Queuing Protocol):** Protocolo de mensajería orientado a mensajes que permite la comunicación entre sistemas distribuidos, facilitando la interoperabilidad y la fiabilidad en la transmisión de datos.
- **Amazon DynamoDB:** Base de datos NoSQL totalmente gestionada por AWS, diseñada para aplicaciones que requieren baja latencia y escalabilidad.
- **Amazon Timestream:** Base de datos de series temporales de AWS optimizada para almacenar y analizar datos que cambian con el tiempo, como los generados por dispositivos IoT.
- **AWS IoT Core:** Servicio de Amazon Web Services que permite conectar dispositivos IoT a la nube de forma segura, facilitando la recopilación y análisis de datos.
- **Arduino IDE:** Entorno de desarrollo integrado que permite escribir, compilar y cargar código en placas de desarrollo como Arduino y ESP8266.

C

- **CoAP (Constrained Application Protocol):** Protocolo de internet diseñado para dispositivos con recursos limitados, que permite la comunicación eficiente en redes IoT.

D

- **DDS (Data Distribution Service):** Protocolo de comunicación que permite el intercambio de datos en sistemas distribuidos en tiempo real, utilizando un modelo de publicación/suscripción.
- **DevOps:** Conjunto de prácticas que integran el desarrollo de software (Dev) y las operaciones de tecnología (Ops), con el objetivo de acortar el ciclo de vida del desarrollo y proporcionar entregas continuas de alta calidad.

H

- **HTTP (Hypertext Transfer Protocol):** Protocolo de comunicación utilizado en la World Wide Web para la transferencia de información entre clientes y servidores.

- **HX711**: Convertidor analógico a digital (ADC) de 24 bits, comúnmente utilizado para leer datos de celdas de carga en aplicaciones de medición de peso.

I

- **IDE (Integrated Development Environment)**: Aplicación que proporciona herramientas integradas para el desarrollo de software, como edición de código, compilación y depuración.
- **IMC (Índice de Masa Corporal)**: Medida utilizada para evaluar la relación entre el peso y la altura de una persona, utilizada como indicador general de salud.
- **IoT (Internet de las Cosas)**: Red de dispositivos físicos interconectados que recopilan y comparten datos a través de internet.

J

- **JMS (Java Message Service)**: API de Java que permite la creación, envío, recepción y lectura de mensajes entre sistemas distribuidos.
- **JSON (JavaScript Object Notation)**: Formato ligero de intercambio de datos, fácil de leer y escribir para humanos, y fácil de parsear y generar para máquinas.

L

- **Lean**: Metodología de desarrollo que busca maximizar el valor para el cliente minimizando el desperdicio, enfocándose en la eficiencia y la mejora continua.
- **LoRaWAN (Long Range Wide Area Network)**: Protocolo de red de área amplia de largo alcance diseñado para dispositivos IoT que requieren comunicaciones a larga distancia con bajo consumo de energía.

M

- **MQTT (Message Queuing Telemetry Transport)**: Protocolo de mensajería ligero basado en el modelo de publicación/suscripción, ideal para dispositivos con recursos limitados y redes de baja banda ancha.
- **Microcontrolador**: Circuito integrado que contiene un procesador, memoria y periféricos de entrada/salida, utilizado para controlar dispositivos electrónicos.

N

- **NoSQL:** Tipo de base de datos que permite el almacenamiento y recuperación de datos no estructurados o semi-estructurados, ofreciendo flexibilidad y escalabilidad.

P

- **Prototipado Rápido:** Técnica de desarrollo que permite la creación rápida de un modelo funcional de un producto para validar conceptos y funcionalidades.

S

- **Sensor VL53L0X:** Sensor de distancia basado en tecnología de tiempo de vuelo (ToF) que mide la distancia al objeto más cercano mediante un láser infrarrojo.
- **Sensor HX711:** Convertidor analógico a digital utilizado para leer señales de celdas de carga, comúnmente empleado en básculas electrónicas.

W

- **Wi-Fi (802.11):** Conjunto de estándares de comunicación inalámbrica que permiten la conexión de dispositivos a redes locales y a internet.
- **Waterfall (Metodología en Cascada):** Modelo de desarrollo de software lineal y secuencial, donde cada fase debe completarse antes de pasar a la siguiente.

Apéndice A

- Código de programación en C++ del prototipo generado en Arduino IDE.

❖ Librerías incluidas

```
1 #include "FS.h"
2 #include "HX711.h"
3 #include <ESP8266WiFi.h>
4 #include <PubSubClient.h>
5 #include <NTPClient.h>
6 #include <WiFiUdp.h>
7 #include <ArduinoJson.h>
8 #include "Adafruit_VL53L0X.h"
```

❖ Factor de calibración HX711

```
13 //----- Parámetro para calibrar el peso y el sensor-----
14
15 #define CALIBRACION 20780.0
16
```

❖ Configuración de servicios de red

```
24 //----- Configuración de la red WiFi-----
25 const char* ssid = "SSID";
26 const char* password = "*****";
27 WiFiUDP ntpUDP;
28 NTPClient timeClient(ntpUDP, "pool.ntp.org");
29
30 //----- Configuración del servidor MQTT (AWS)-----
31 const char * AWS_endpoint =
32 lasdhlhauouxaukaxq.iot.us-east-1.amazonaws.com; //MQTT broker IP
33 void callback(char * topic, byte * payload, unsigned int length) {
34     Serial.print("Mensaje ingresado: [");
35     Serial.print(topic);
36     Serial.print("] ");
37     for (int i = 0; i < length; i++) {
38         Serial.print((char) payload[i]);
39     }
40     Serial.println();
41 }
```



```

42 const char* mqtt_topic = "esp8266/datos";
43
44 //-----Cliente WiFi y MQTT-----
45 WiFiClientSecure espClient;
46 PubSubClient
47 client(AWS_endpoint, 8883, callback, espClient); //MQTT port 8883 - standard

```

❖ Conexión de red WI-FI

```

69 //-----Función para conectar a WiFi-----
70 void setup_wifi() {
71     delay(10);
72     espClient.setBufferSizes(512, 512);
73     Serial.println();
74     Serial.println("Conectando a WiFi...");
75     Serial.println(ssid);
76     WiFi.begin(ssid, password);
77     while (WiFi.status() != WL_CONNECTED) {
78         delay(500);
79         Serial.print(".");
80     }

```

❖ Reconexión del broker MQTT

```

//-----Función para reconectar MQTT en caso de desconexión-----
92 -
93 void reconnect() {
94     while (!client.connected()) {
95         Serial.print("Intentando conexión MQTT...");
96         if (client.connect("ESP8266_thing")) {
97             Serial.println("Conectado al broker MQTT");
98             client.publish(mqtt_topic,"hola");
99             client.subscribe("inTopic");
100         } else {
101             Serial.print("Error, rc=");
102             Serial.print(client.state());
103             Serial.println(" Intentando de nuevo en 5 segundos...");
104             char buf[256];
105             espClient.getLastSSLError(buf, 256);
106             Serial.print("WiFiClientSecure SSL error: ");
107             Serial.println(buf);
108             delay(5000);
109         }
110     }

```

```
111 }
```

❖ Función de interrupción de sensores

```
148 //-----control de interrupcion de sensado-----  
149 volatile bool sensorActivo = false; // Sensado comienza inactivo  
150 void IRAM_ATTR activarSensado() {  
151     sensorActivo = true; // Habilitar sensado al presionar el botón  
152 }  
153 //-----Función para carga de interrupcion de sensado-----  
154 -----  
155 void interruptLoad() {  
156     pinMode(BOTON_PIN, INPUT_PULLUP);  
157  
158     // Configurar interrupción en el botón (Flanco de bajada)  
159     attachInterrupt(digitalPinToInterrupt(BOTON_PIN), activarSensado,  
160     FALLING);  
161  
162     Serial.println("Sistema iniciado. Sensado inactivo.");
```

❖ Inicializar sensor VL53L0X

```
164 //-----Función para inicializar el sensor VL53L0X estatura----  
165 -----  
166 void loadVL53L0X() {  
167     while (! Serial) {  
168         delay(1);  
169     }  
170  
171     Serial.println("Adafruit VL53L0X prueba");  
172     if (!lox.begin()) {  
173         Serial.println(F("falla en la carga del sensor VL53L0X"));  
174         while(1);  
175     }
```

❖ Inicializar sensor HX711

```
202 //----- Función para inicializar el ADC HX711 sensor báscula--  
203 -----  
204 void loadHX711() {  
205     #ifdef DEBUG_HX711  
206         Serial.println("[HX7] Inicio del sensor HX711");  
207     #endif  
208     bascula.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
```

```
208     bascula.set_scale(CALIBRACION);  
209     bascula.tare();  
210     Serial.println("");  
211 }
```

Apéndice B

- **Código de calibración del sensor HX711 desarrollado en C++ en Arduino IDE. Creado por Nathan Seidle (SparkFun Electronics, 2023).**

```
1  /*
2   Este código fue creado por Nathan Seidle de SparkFun Electronics
3
4   Este programa permite calibrar la báscula. Se utiliza para determinar el
5   factor de calibración.
6   También puede servir para generar el factor cero útil en básculas que tienen
7   una
8   masa permanente.
9
10  Para empezar la configuración de la báscula se debe iniciar este programa sin
11  tener ningún peso encima de la báscula.
12  Una vez empiecen a aparecer medidas de peso en el monitor serie, colocar algún
13  objeto con un peso conocido sobre la báscula.
14  Luego, en el monitor serie teclear + o - (en símbolo) para ajustar el factor de
15  calibración (factor_calibracion) hasta que la salida coincida
16  con el peso conocido del objeto.
17  Por ejemplo, se puede poner un kilo de algún objeto que se sabe con exactitud
18  que
19  pesa un kilo.
20  Una vez el peso coincida, anotar el valor del factor de calibración
21  (factor_calibracion).
22  El factor de calibración puede ser muy diferente al utilizado en el ejemplo.
23  Puede que sea muy positivo o muy negativo. Todo esto depende de la
24  configuración
25  del sistema de báscula, la célula de carga y los sensores de carga.
26 */
27
28 #include "HX711.h"
29
30 // Pin de datos y de reloj
31 byte pinData = 2;
32 byte pinClk = 14;
33
34 HX711 bascula;
35
36 // Parámetro para calibrar el peso y el sensor
37 float factor_calibracion = 20780.0; //Este valor es inicial
38 float peso;
```

```

36 void setup() {
37   Serial.begin(9600);
38   Serial.println("HX711 programa de calibracion");
39   Serial.println("Quita cualquier peso de la bascula");
40   Serial.println("Una vez empiece a mostrar informacion de medidas, coloca un
41   peso conocido encima de la bascula");
42   Serial.println("Presiona + para incrementar el factor de calibracion");
43   Serial.println("Presiona - para disminuir el factor de calibracion");
44   // Iniciar sensor
45   bascula.begin(pinData, pinClk);
46   // Aplicar la calibración
47   bascula.set_scale();
48   // Iniciar la tara
49   // No tiene que haber nada sobre el peso
50   bascula.tare();
51   // Obtener una lectura de referencia
52   long zero_factor = bascula.read_average();
53   // Mostrar la primera desviación
54   Serial.print("Zero factor: ");
55   Serial.println(zero_factor);
56 }
57 void loop() {
58   // Aplicar calibración
59   bascula.set_scale(factor_calibracion);
60   // Mostrar la información para ajustar el factor de calibración
61   Serial.print("Leyendo: ");
62   Serial.print(bascula.get_units(), 3);
63   Serial.print(" kgs");
64   Serial.print(" factor_calibracion: ");
65   Serial.print(factor_calibracion);
66   Serial.println();
67   // Obtener información desde el monitor serie
68   if (Serial.available())
69     char temp = Serial.read();
70     if (temp == '+')
71       factor_calibracion += 10;
72     else if (temp == '-')
73       factor_calibracion -= 10;
74   }
75 }

```

Apéndice C

- Sistema IoT y validación estadística para monitoreo de salud de estudiantes (Gómez-Gayosso, J. C., Suárez-Cansino, J., López-Morales, V., & Franco-Árcega, A., 2022).



DESDE 2013
<https://repositorio.uaeh.edu.mx/revistas/index.php/icbi/issue/archive>
Pädi Boletín Científico de Ciencias Básicas e Ingenierías del ICBI



Publicación Semestral Pädi Vol. 10 No. Especial 3 (2022) 103-111

Sistema IoT y validación estadística para monitoreo de salud de estudiantes

IoT system and statistical validation for monitoring students' health

J. C. Gómez-Gayosso^a, J. Suárez-Cansino^{b,*}, V. López-Morales^b, A. Franco-Árcega^b

^aUniversidad Autónoma del Estado de Hidalgo, Instituto de Ciencias Básicas e Ingeniería, Área Académica de Computación y Electrónica, Maestría en Internet de las Cosas, Colonia Carboneras, CP 42184, Mineral de la Reforma, Hidalgo, México.

^bUniversidad Autónoma del Estado de Hidalgo, Instituto de Ciencias Básicas e Ingeniería, Área Académica de Computación y Electrónica, Cuerpo Académico de Computación Inteligente, Colonia Carboneras, CP 42184, Mineral de la Reforma, Hidalgo, México.

Figura 66. Publicación Pädi Vol.10 No. Especial 3 Tomada de Boletín científico ICBI (2022).

• Resumen

La combinación del Internet de las Cosas y de la Estadística para Ciencias de los Datos resulta bastante interesante, sobre todo cuando se aplican dentro del área educativa. Este trabajo propone el desarrollo de una estación de salud (quiosco) y del análisis de los datos biométricos obtenidos a través de un procedimiento experimental con el cual obtener valores de variables antropométricas que sean útiles para indicar el estado de salud de estudiantes de la Preparatoria 2, perteneciente a la Universidad Autónoma del Estado de Hidalgo. En una primera fase, el trabajo propone la medición de peso y estatura de las y los estudiantes, así como el Índice de Masa Corporal, el cual se emplea tradicionalmente para determinar sobrepeso y su relación con otros tipos de enfermedades. Los resultados experimentales muestran que adquirir esta información requiere de sensores de peso y estatura, para cuyo análisis estadístico existen modelos que permiten realizar pruebas de normalidad y homogeneidad de varianzas, las cuales resultan útiles para determinar el tipo de estadística a usar.

