UNIVERSIDAD LAICA ELOY ALFARO DE MANABÍ

TRABAJO DE TITULACIÓN

MODALIDAD DE PROYECTO INTEGRADOR

PREVIO A LA OBTENCIÓN DEL TÍTULO DE:

INGENIERO EN TECNOLOGÍAS DE LA INFORMACIÓN

TEMA DEL PROYECTO:

"IMPLEMENTACIÓN DE UNA SOLUCIÓN IOT PARA EL MONITOREO EN TIEMPO REAL DE VARIABLES FÍSICO-QUÍMICAS EN TANQUES DE LARVAS DE CAMARÓN DEL LABORATORIO DE INNOVACIÓN EN SISTEMAS ACUÍCOLAS DE LA ULEAM."

AUTORES:

ALONZO MERIZALDES STEFANY MICHELLE

CEDEÑO RIVERA JOSELINE MALENA

TUTOR:

ING. WILLIAN JESÚS ZAMORA MERO, PHD

FACULTAD DE CIENCIAS DE LA VIDA Y TECNOLOGÍAS

CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

2025-(2)

MANTA – MANABÍ - ECUADOR

CERTIFICACIÓN DEL DIRECTOR DE TRABAJO DE TITULACIÓN

En calidad de docente tutor(a) de la Facultad de Ciencias de la Vida y Tecnologías de la Universidad Laica "Eloy Alfaro" de Manabí, certifico:

Haber dirigido y revisado el trabajo de titulación, cumpliendo el total de 384 horas, bajo la modalidad de Investigación, cuyo tema de proyecto es **"IMPLEMENTACIÓN DE UNA SOLUCIÓN IOT PARA EL MONITOREO EN TIEMPO REAL DE VARIABLES FÍSICO- QUÍMICAS EN TANQUES DE LARVAS DE CAMARÓN DEL LABORATORIO DE INNOVACIÓN EN SISTEMAS ACUÍCOLAS DE LA ULEAM.",** el mismo que ha sido desarrollado de acuerdo con los lineamientos internos de la modalidad en mención y en apego al cumplimiento de los requisitos exigidos por el Reglamento de Régimen Académico, por tal motivo **CERTIFICO**, que el mencionado proyecto reúne los méritos académicos, científicos y formales, suficientes para ser sometido a la evaluación del tribunal de titulación que designe la autoridad competente.

La autoría del tema desarrollado corresponde a las jóvenes Alonzo Merizaldes Stefany Michelle y Cedeño Rivera Joseline Malena estudiantes de la carrera de Ingeniería en Tecnologías de la Información, período académico 2025-2026 quien se encuentra apto para la sustentación de su trabajo de titulación.

Particular que certifico para los fines consiguientes, salvo disposición de Ley en contrario.

Manta, 4 de febrero del 2026

Lo certifico,

Ing. Willian Jesús Zamora Mero, PhD
Docente Tutor(a)

DECLARACIÓN DE AUTORÍA DEL ESTUDIANTE

"Declaro que este trabajo es original, de nuestra autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes."

Alonzo Merizaldes Stefany Michelle

1351896814

Cedeño Rivera Joseline Malena

1315603876

DEDICATORIA

A Dios, por sus bendiciones, por la salud, por mi familia y por permitirme despertar cada día con nuevas oportunidades.

A mi abuela, Olga Mieles, quien ha sido un pilar fundamental en mi vida; sin ella, este proceso no habría sido el mismo.

A mi abuelo, Marco Alonzo, quien, aunque ya no se encuentra físicamente conmigo, fue un apoyo importante durante mi etapa escolar y colegial, dejando enseñanzas que me acompañan hasta hoy.

A mis padres, Marco Alonzo y Diana Merizaldes, quienes me apoyaron desde el primer día en mi formación profesional y han sido mi mayor motivación para seguir adelante.

A mis tías, Shirley Alonzo y Mary Alonzo, por su apoyo constante, el cual me ayudó a mantenerme firme en este camino.

A mis tíos, primos y familiares, y a todas las personas que, de una u otra manera, aportaron significativamente a este proceso.

A mis amigas Karla y Katheryn, quienes siempre me impulsaron a seguir adelante y a no rendirme.

A mi compañera de tesis, Joseline Cedeño, cuya compañía y apoyo durante este proceso fueron de gran ayuda para hacer posible este logro.

A los amigos que me regaló la carrera, Alan y Nicoll, quienes me acompañaron en la lucha constante de cada semestre.

A mis mascotas, quienes me acompañaron en cada noche de desvelo y me recibían con amor cada vez que llegaba a casa.

Alonzo Merizaldes Stefany Michelle

DEDICATORIA

A mi hijo por ser mi mayor motivación y alegría, por ser la razón de mi superación, por su paciencia en mi ausencia, por secarme las lágrimas cuando no podía más, por esos besos y abrazos que me daban fuerza para levantarme día a día y no rendirme. Esta tesis es un pequeño testimonio de todo lo que hago, lo hago pensando en ti. Gracias por llenar mi mundo de amor y dulzura.

A mis padres y hermana por todo el apoyo que me brindaron, por ser mi sostén incondicional en cada etapa de este camino. Gracias por enseñarme el valor del esfuerzo, por creer en mí cuando yo dudaba y por acompañarme con paciencia y amor. Cada logro que alcanzo lleva consigo su sacrificio silencioso. Esta tesis es fruto de sus enseñanzas, su ejemplo y su fe constante. No hay palabras suficientes para agradecer todo lo que han hecho por mí. Siempre llevaré conmigo y con mucho orgullo todo lo que me dieron: valores, fuerza y corazón.

A mi pareja por su paciencia por escuchar todo lo malo y bueno que me pasaba día a día, por esos ánimos que me daba para seguir adelante cuando le decía que no podía más, por esperarme todos estos años y por todo el apoyo brindado en este proceso. Esta tesis es un tributo a la colaboración, paciencia y comprensión que has brindado a lo largo de este viaje académico. Gracias por ser un pilar de fortaleza y un ejemplo para nuestro hijo.

A mis demás familiares y amigos, quienes han estado ahí para bríndame su apoyo incondicional y haberme acompañado en todo este proceso.

Cedeño Rivera Joseline Malena

AGRADECIMIENTO

A Dios, mis abuelos, padres, y todos quienes desde el día uno estuvo presente en este caminar.

A la Universidad Laica Eloy Alfaro de Manabí, por abrirme las puertas y permitirme formarme como profesional.

Al tutor Ing. Willian Jesús Zamora Mero, PhD, por su orientación en el desarrollo del proyecto, brindándonos sabiduría y experticia durante todo el proceso.

Al Ing. Mike Paolo Machuca Ávalos, Mg., por su paciencia, experiencia y enseñanzas, siendo una mano derecha fundamental para la culminación de este trabajo.

A los compañeros Luis y Rubén, por su apoyo brindado durante la realización del proyecto.

A los profesores, por compartir su experticia y brindarnos nuevos conocimientos.

A la niña que fui, que a sus 8 años recibió su primera computadora con una felicidad inmensa, pero también con la incertidumbre y curiosidad de no saber cómo funcionaba ni hasta dónde podría llevarla. Sin saberlo, ese momento sembró en mí el amor por la tecnología y despertó un sueño que con los años solo creció más. Desde ese día nunca dejé de aprender, de preguntar y de querer descubrir algo nuevo. Hoy le agradezco profundamente a esa niña por atreverse a soñar, por descubrir este mundo y por tener la valentía de luchar por lo que hoy estoy logrando.

Alonzo Merizaldes Stefany Michelle

AGRADECIMIENTO

Ante todo, agradezco a Dios por darme sabiduría, fuerza y salud para poder conseguir esta meta que con tanto amor anhelé.

A mi hijo y mi pareja por acompañarme, esperarme y comprenderme en todo este proceso de titulación, por toda la paciencia que me han brindado y por todas las palabras de apoyo que me entusiasmaba a seguir adelante.

A mi madre mi padre quienes han estado presentes en los buenos y malos momentos, por su amor incondicional por siempre decir que yo si puedo con todo lo que me proponga, por enseñarme valores.

A mi hermana y mi sobrina por su apoyo constante y por inspirarme a seguir adelante incluso en los momentos más difíciles.

A mi compañera de investigación, le agradezco las numerosas y valiosas conversaciones inspiradoras, el intercambio de ideas que hemos compartido y el apoyo incondicional en los momentos más difíciles del proceso.

A mi tutor, el Ing. Willian Jesús Zamora Mero, PhD, y al Ing. Mike Paolo Machuca Avalos, Mg, por su paciencia, orientación y consejos durante este proceso de este proyecto de investigación. Sus amplias experiencias en el ámbito y dedicación han sido clave para el desarrollo de este proyecto.

Un sincero agradecimiento a todos mis amigos, compañeros y profesores que estuvieron conmigo en los momentos de estrés y alegría durante este largo y retador camino. Su apoyo, confianza, soporte y cariño han sido invaluables.

A todos, gracias por ser parte de este logro.

Cedeño Rivera Joseline Malena

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

RESUMEN

El Laboratorio de Innovación en Sistemas Acuícolas de la ULEAM enfrenta limitaciones en el monitoreo de pH, oxígeno disuelto y temperatura en tanques de larvas de camarón. El método actual depende de mediciones manuales esporádicas que crean "ventanas ciegas" donde no hay registro de datos. Durante estos periodos, fluctuaciones peligrosas en los parámetros pueden pasar desapercibidas hasta causar mortalidad larval.

Este proyecto de integración implementa una plataforma IoT que monitorea estas variables continuamente. La arquitectura consiste en sensores especializados conectados a un Raspberry Pi. Los datos se transmiten vía MQTT al Servidor Web IoT ULEAM para su procesamiento y almacenamiento. Una aplicación móvil permite visualizar los datos en tiempo real y consultar historiales.

La plataforma “IoT ULEAM Guardian Shrimp” proporciona a investigadores y técnicos información continua y confiable para tomar decisiones sobre el cultivo. Su diseño modular facilita la incorporación de sensores adicionales o su adaptación a otros sistemas acuícolas. El proyecto de integración contribuye a la adopción de acuicultura de precisión en la región.

ABSTRACT

The Aquaculture Systems Innovation Laboratory at ULEAM faces limitations in monitoring pH, dissolved oxygen, and temperature in shrimp larvae tanks. The current method relies on sporadic manual measurements that create “blind spots” where no data is recorded. During these periods, dangerous fluctuations in parameters can go unnoticed until they cause larval mortality.

This project implements an IoT platform that continuously monitors these variables. The architecture consists of specialized sensors connected to a Raspberry Pi. The data is transmitted via MQTT to the ULEAM IoT Web Server for processing and storage. A mobile application allows real-time data visualization and historical data consultation.

The ULEAM Guardian Shrimp IoT platform provides researchers and technicians with continuous and reliable information for making farming decisions. Its modular design facilitates the incorporation of additional sensors or its adaptation to other aquaculture systems. The project contributes to the adoption of precision aquaculture in the region.

INTRODUCCIÓN

Las aplicaciones del Internet de las Cosas (IoT) se expanden en diversos contextos: ciudades inteligentes, industria 4.0 e IIoT (Bugshan, Khalil, Moustafa, & Rahman, 2023). Los avances tecnológicos recientes permiten crear soluciones centralizadas para el desarrollo sostenible de la producción alimentaria. El hardware necesario ya no requiere inversiones prohibitivas como hace años, y su disponibilidad en Ecuador ha mejorado significativamente.

En Manabí, la acuicultura representa un pilar económico fundamental. Sin embargo, gestionar eficientemente la producción de larvas de camarón enfrenta un desafío persistente: la falta de datos precisos y accesibles en tiempo real. Los investigadores y técnicos no disponen de información actualizada sobre las condiciones físico-químicas en los tanques de cultivo, lo que dificulta tomar decisiones fundamentadas para asegurar la supervivencia y desarrollo larval. La ausencia de una plataforma tecnológica capaz de captar, analizar y visualizar esta información representa una limitación crítica.

Este proyecto de integración implementa una solución IoT para el monitoreo en tiempo real de variables físico-químicas en tanques de larvas de camarón del Laboratorio de Innovación en Sistemas Acuícolas (al cual abreviaremos para efectos prácticos de este documento como LISA) de la ULEAM. El cual permite validar las funcionalidades de la solución tecnológica y demostrar su potencial en un ámbito crítico para la región. El proyecto de integración resuelve un problema tecnológico del laboratorio y simultáneamente establece una base sólida para futuras investigaciones en el sector acuícola.

La propuesta utiliza tecnologías modernas de código abierto adaptables a distintos tipos de sensores (pH, temperatura, oxígeno disuelto) y protocolo de comunicación MQTT. El diseño se fundamenta en principios de escalabilidad, flexibilidad e interoperabilidad, facilitando su aplicación en diferentes contextos y la gestión de diversas categorías de datos.

Este trabajo se enmarca en el proyecto de investigación universitario denominado "Arquitectura Integrada de IoT e IA para la Investigación Multidisciplinaria y Sostenible", de la Facultad Ciencias de la Vida y Tecnología, cuyo objetivo es centralizar datos de áreas como gestión ambiental, agricultura y acuicultura. La implementación en el LISA constituye un primer escalón concreto hacia este objetivo institucional.

Este proyecto de titulación en Tecnologías de la Información alinea innovación con sostenibilidad. Utiliza tecnología para generar impacto en un sector productivo regional, fortalece la capacidad de investigación de la ULEAM y sienta bases para la acuicultura de precisión en Manabí.

CAPÍTULO I

1 Planteamiento del problema

1.1 Ubicación y contextualización del problema

Este proyecto de integración se ubica en La Universidad Laica Eloy Alfaro de Manabí (ULEAM), a través de su Laboratorio de Innovación en Sistemas Acuícolas (LISA) ver figura 1 y 2.

Figura 1

Ubicación del laboratorio

Nota: La imagen representa la ubicación del laboratorio, tomado de Google Maps, Google LLC (2025), https://maps.app.goo.gl/tJmvACxEHDvFvR5CA

Figura 2

La imagen representa la ubicación física del laboratorio dentro de la Uleam

Nota: Autoría propia.

El LISA enfrenta limitaciones en el monitoreo de variables físico-químicas (pH, oxígeno disuelto y temperatura) en tanques de cultivo de larvas de camarón. El método

actual depende de mediciones manuales esporádicas que generan información fragmentada. Esta falta de datos precisos y continuos dificulta detectar oportunamente desviaciones críticas en los parámetros del agua, afectando la capacidad de los investigadores y técnicos para tomar decisiones fundamentadas que aseguren la supervivencia larval.

1.2 Planteamiento del problema

El LISA no tiene un sistema automatizado para monitorear los tanques de larvas de camarón. Esto genera varios problemas: los estudiantes que realizan sus prácticas, técnicos e investigadores no desarrollan habilidades en tecnologías IoT, hay poca experiencia en el manejo de sensores especializados, y la infraestructura actual no permite capturar datos de forma continua.

El problema real es más amplio. En la región no existe una plataforma web que permita recolectar, analizar y visualizar datos en tiempo real de dispositivos IoT. Sin herramientas que traduzcan los datos de sensores en información útil, los investigadores y técnicos no pueden tomar decisiones fundamentadas para optimizar las condiciones de cultivo.

Una mala gestión de los parámetros del agua no solo causa mortalidad larval y pérdidas económicas. También aumenta el riesgo de brotes de enfermedades virales como WSSV (mancha blanca), IHHNV, o TSV (síndrome de Taura), que pueden propagarse a otras especies acuícolas y afectar la sostenibilidad ambiental de la región.

Por esto necesitamos una solución tecnológica que monitoree continuamente pH, oxígeno disuelto y temperatura. Una plataforma web y una aplicación móvil permitirían detectar anomalías temprano, reducir la mortalidad larval, mejorar las prácticas de

bioseguridad y optimizar la producción. Estas herramientas conectan innovación tecnológica con sostenibilidad operativa, protegiendo tanto la rentabilidad del sector como el ambiente.

1.3 Formulación del problema

El cultivo de larvas de camarón requiere control estricto de las variables físico-químicas del agua. En el Laboratorio de Innovación en Sistemas Acuícolas de la ULEAM, el monitoreo de pH, oxígeno disuelto y temperatura se realiza mediante mediciones manuales esporádicas.

La ausencia de un sistema automatizado provoca alta latencia en la toma de decisiones. El personal detecta problemas cuando las condiciones del agua ya alcanzaron niveles críticos. Esto resulta en tasas impredecibles de mortalidad larval, pérdidas significativas en producción y desperdicio de recursos. El laboratorio carece de una solución tecnológica que proporcione acceso continuo e inmediato a estos datos vitales, limitando su capacidad para optimizar las condiciones de crianza y realizar investigación aplicada de precisión.

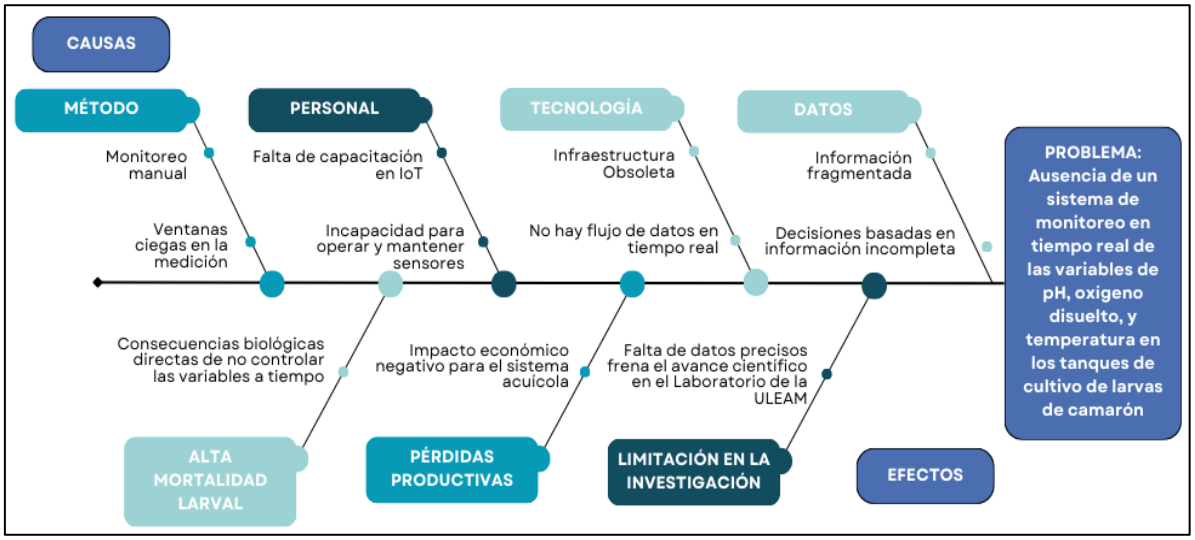
Con estos antecedentes se puede llegar a la siguiente formulación del problema:

¿Cómo la ausencia de un sistema de monitoreo automatizado y en tiempo real de la calidad del agua en el laboratorio de la ULEAM impide la toma de decisiones oportuna para prevenir eventos de mortalidad masiva en el cultivo de larvas de camarón?

1.4 Diagrama Ishikawa

Figura 3

Diagrama de Ishikawa sobre las causas y efectos en la ausencia de un sistema de monitoreo



Nota: Autoría propia.

La Figura 3 muestra las causas y efectos del problema principal: no hay un sistema que monitoree en tiempo real el pH, oxígeno disuelto y temperatura en los tanques del Laboratorio de Innovación en Sistemas Acuícolas de la ULEAM.

Causas identificadas

El monitoreo actual depende completamente del trabajo manual. Esto crea "ventanas ciegas": hay horas donde nadie mide las variables y cualquier cambio brusco puede pasar desapercibido hasta que sea tarde. Además, el personal del laboratorio no ha recibido capacitación en tecnologías IoT, lo que dificulta incorporar sensores modernos o mantenerlos funcionando.

La infraestructura tecnológica tampoco ayuda. Los equipos actuales no transmiten datos en tiempo real, entonces cuando algo sale mal, ya pasó. Los datos que se obtienen están dispersos y muchas veces los técnicos deben decidir basándose en información incompleta o de días anteriores.

Consecuencias observadas

La mortalidad de larvas aumenta cuando las variables se salen de control sin que nadie lo note a tiempo. Esto representa pérdidas económicas directas para el laboratorio. También afecta la investigación: sin datos continuos y confiables es difícil generar estudios rigurosos o proponer mejoras al sistema de cultivo.

El diagrama deja claro que implementar IoT no es solo modernizarse por modernizarse. Es necesario para mantener las larvas vivas y para que la ULEAM pueda desarrollar investigación seria en acuicultura.

1.5 Objetivos

1.5.1 Objetivo general

Implementar una solución basada en Internet de las Cosas (IoT) para el monitoreo en tiempo real de las variables de pH, oxígeno disuelto, y temperatura en los tanques de cultivo de larvas de camarón, ubicada en el Laboratorio de Innovación en Sistemas Acuícolas de la Universidad Laica Eloy Alfaro de Manabí, con el fin de proporcionar información precisa y accesible para optimizar las condiciones de crianza.

1.5.2 Objetivos específicos

- Diseñar e implementar el sistema de adquisición de datos: Seleccionar, configurar e integrar los sensores de pH, oxígeno disuelto y temperatura junto con la electrónica necesaria, para la correcta lectura y adquisición de datos de los tanques de larvas de camarón.

- Desarrollar la infraestructura de comunicación IoT: Configurar los módulos de conectividad inalámbrica y establecer la comunicación segura entre el sistema de adquisición de datos en los tanques y una plataforma centralizada para la recepción y procesamiento de la información.

- Desarrollar un aplicativo móvil: Diseñar y desarrollar una aplicación móvil intuitiva que permita a los usuarios del laboratorio visualizar en tiempo real los valores de pH, oxígeno disuelto y temperatura, así como acceder a datos históricos.

- Realizar pruebas y validación del sistema: Poner a prueba la solución IoT implementada en el entorno del laboratorio para verificar la precisión de las mediciones, la fiabilidad de la transmisión de datos y el correcto funcionamiento del aplicativo móvil.

1.6 Justificación

En la actualidad, la eficiencia en la producción acuícola depende directamente del acceso a información precisa y procesada en tiempo real. En el Laboratorio de Innovación en Sistemas Acuícolas de la Universidad Laica Eloy Alfaro de Manabí (ULEAM), el monitoreo de variables críticas como el pH, el oxígeno disuelto y la temperatura se realiza de manera convencional, lo que limita la capacidad de respuesta ante contingencias biológicas. La presente investigación se justifica en la necesidad de cerrar esta brecha mediante la implementación de una solución basada en el Internet de las Cosas (IoT), diseñada específicamente para optimizar la crianza de larvas de camarón.

El desarrollo de este proyecto se fundamenta en los siguientes pilares estratégicos y técnicos:

Automatización de la Adquisición de Datos: El diseño e implementación de un sistema electrónico dedicado permite la lectura continua de parámetros fisicoquímicos. Al integrar sensores especializados, se elimina el error humano y la discontinuidad de las mediciones manuales, garantizando que los datos recolectados en los tanques sean el reflejo fiel de las condiciones del cultivo.

Fortalecimiento de la Infraestructura Digital: La creación de una arquitectura de comunicación IoT robusta y segura asegura que la información fluya desde los nodos de adquisición hasta una plataforma web centralizada. El uso de protocolos de conectividad modernos garantiza la integridad y disponibilidad de los datos, elementos esenciales para el rigor científico de las investigaciones realizadas en la ULEAM.

Democratización y Accesibilidad de la Información: El desarrollo de un aplicativo móvil intuitivo transforma datos técnicos complejos en información visual y comprensible. Esto permite que investigadores y técnicos visualicen el estado del

laboratorio en tiempo real y accedan a historiales de medición desde cualquier lugar, agilizando la toma de decisiones críticas fuera del horario operativo presencial.

Validación y Rigor Científico: La fase de pruebas en el entorno real del laboratorio es fundamental para certificar la fiabilidad del sistema. Al verificar la precisión de las lecturas y la estabilidad de la transmisión, se entrega una herramienta tecnológica validada que puede ser replicada o escalada en otros sistemas de cultivo de la región.

Impactos Esperados: La integración de estos componentes permitirá establecer correlaciones directas entre las variables ambientales y la salud del organismo. Por ejemplo, mediante el monitoreo preventivo de los niveles de oxígeno, se proyecta un incremento del 25% en la supervivencia larval. Asimismo, el análisis de las fluctuaciones de pH tras la alimentación facilitará la validación de protocolos nutricionales, con una mejora estimada del 30% en la eficiencia alimenticia.

Por lo tanto, este proyecto propone la resolución de un problema tecnológico. Se constituye como un modelo replicable de acuicultura de precisión que alinea la innovación tecnológica con la misión institucional de la ULEAM de fomentar prácticas productivas responsables, trazables y sostenibles que fortalezcan el sector acuícola nacional.

CAPITULO II

2. Marco teórico de la investigación

La construcción del marco teórico de la presente investigación se desarrolló mediante un proceso sistemático dividido en dos fases metodológicas fundamentales. La fase inicial consistió en una revisión crítica del estado de arte de la literatura científica especializada. Además, de revisión artículos indexados en bases de datos de alto impacto, memorias de congresos internacionales y proyectos de titulación enfocados en soluciones de Internet de las Cosas (IoT) aplicadas a la acuicultura y el monitoreo ambiental.

Esta exploración documental permitió identificar tendencias tecnológicas, arquitecturas de referencia y protocolos de comunicación predominantes en el estado del arte. Los hallazgos más relevantes fueron sintetizados en tablas comparativas, las cuales facilitan un análisis horizontal de las propuestas examinadas al contrastar sus componentes tecnológicos, stacks de desarrollo y resultados obtenidos.

En la segunda fase, se procedió a la conceptualización y desglose de los elementos tecnológicos que fundamentan la propuesta. Esta etapa implicó la definición técnica de los pilares del sistema, incluyendo arquitecturas IoT, protocolo de comunicación, y los principios de diseño de interfaces para la visualización de datos en tiempo real. Esta fundamentación no solo proporciona el sustento científico necesario, sino que también establece un vocabulario técnico coherente que guía el desarrollo y la posterior discusión de los resultados, asegurando la trazabilidad entre la teoría aplicada y los objetivos de monitoreo en el Laboratorio de Innovación en Sistemas Acuícolas de la ULEAM.

2.1 Estado de arte

El presente análisis del estado del arte se sustenta en una revisión sistemática de investigaciones previas, incluyendo tesis y publicaciones científicas. Esto permite establecer un marco comparativo para el desarrollo de plataformas IoT aplicadas a la monitorización de variables en sistemas acuícolas. Además, el estudio se estructura en tres niveles de análisis: global, nacional y local, examinando críticamente los objetivos, arquitecturas tecnológicas, stacks de desarrollo y resultados alcanzados en soluciones similares. Esta metodología comparativa posibilita identificar aportes significativos y lecciones aprendidas que enriquecen el diseño de la plataforma IoT ULEAM Guardian Shrimp para el monitoreo de tanques de larvas de camarón. A continuación, se detallan los trabajos más relevantes identificados:

Nivel Global

(Naj & Sanzgiri, 2021) implementaron una metodología basada en el Internet de las Cosas (IoT) orientada al monitoreo automatizado y continuo de la calidad del agua en sistemas acuícolas. A través de la integración de microcontroladores Arduino y módulos de comunicación inalámbrica, el sistema permite la adquisición de variables críticas como la temperatura, el pH y turbidez. Esta infraestructura tecnológica facilita la recolección de datos en tiempo real, lo cual resulta fundamental para la prevención de patologías y la mitigación de la mortalidad en especies acuáticas, al tiempo que reduce significativamente la dependencia de la intervención manual y los costos operativos. El estudio concluye que la implementación de redes de sensores IoT robustece la toma de decisiones fundamentada en evidencia técnica, asegurando el mantenimiento de condiciones ambientales óptimas en entornos de producción acuícola.

(Mohd Jais, Abdullah, Mohd Kassim, Abd Karim, & Muhadi, 2024) desarrollaron un sistema de monitoreo de precisión para la calidad del agua en el cultivo de lubina asiática (Lates calcarifer) mediante el uso de sensores de bajo costo. El núcleo de su investigación se centró en mejorar la exactitud de los datos mediante la aplicación de modelos de regresión lineal simple, validando los resultados frente a equipos profesionales de referencia. Los sensores analógicos alcanzaron niveles de precisión notables tras la calibración, con un error cuadrático medio (RMSE) de ±0.05 en pH, ±0.36 mg/L en oxígeno disuelto (OD), ±0.29 ppt en conductividad eléctrica y ±1.13 ppm en amoníaco. La arquitectura tecnológica empleó conectividad Wi-Fi para la transmisión de datos hacia la plataforma ThingSpeak y el aplicativo móvil Virtuino, permitiendo la gestión de información en tiempo real. El estudio concluye que, con una calibración adecuada, la tecnología de bajo costo es capaz de proporcionar la fiabilidad requerida para aplicaciones en la acuicultura comercial.

Nivel Nacional

(Beltrán Mindiola & Mejía Véliz, 2021) crearon un prototipo IoT para medir pH, temperatura y radiación ultravioleta en piscinas camaroneras de la costa ecuatoriana. Utilizaron Arduino, sensor pH, sensor DTH11, sensor radiación ultravioleta y conexión WiFi. La plataforma Thinger.io les permitió visualizar los parámetros medidos constantemente desde cualquier ubicación con acceso a internet. Este proyecto demostró la viabilidad de implementar soluciones IoT de bajo costo en la industria camaronera ecuatoriana.

(Carchipulla Leal, 2018) analizó la importancia del oxígeno disuelto para mejorar la calidad del agua en cultivos de camarón blanco (Litopenaeus vannamei). Identificó los rangos óptimos de oxígeno disuelto y su relación con la supervivencia y crecimiento larval. Su trabajo estableció que mantener niveles adecuados de DO (superiores a 5 mg/L)

es crítico para reducir la mortalidad y el estrés en todas las etapas del cultivo camaronero. Esta investigación proporcionó la base científica para justificar el monitoreo continuo de variables críticas.

Nivel Local

(Pico-Lozano & Mendoza-Intriago, 2020) evaluaron la calidad del agua de mar en la desembocadura del Río Manta y sus efectos en la supervivencia de larvas de camarón blanco (Litopenaeus vannamei). Realizaron bioensayos de toxicidad midiendo oxígeno disuelto (13.5%), salinidad y temperatura (30°C) a 500 metros antes de la desembocadura. Determinaron que las primeras mortalidades se presentaron a las 72 horas en cuatro de seis muestras evaluadas. Este trabajo de la ULEAM demostró la necesidad de monitorear continuamente parámetros físico-químicos para detectar condiciones adversas antes de que afecten la supervivencia larval.

(Ordoñez-Iglesias, Méndez-Martínez, & Zúñiga-Argudo, 2024) evaluaron el efecto del balance iónico en dieta sobre la salud del camarón blanco (Penaeus vannamei) en etapa de precría cultivado en agua de pozo. Trabajaron con 600 organismos durante 30 días, midiendo variables como necrosis (9.30%), larvas azuladas (7.10%), cromatóforos expandidos (10.50%) e intestino en diferentes estados. Su investigación en la ULEAM concluyó que la administración adecuada de iones (Ca^{2+}, Mg^{2+}, K^{+}) en el alimento balanceado mejora el desarrollo del camarón en condiciones de cultivo controlado. Este estudio refuerza la importancia de monitorear no solo variables físico-químicas del agua, sino también indicadores de salud larval.

2.1.1 Tablas de resumen y análisis de los antecedentes de las investigaciones.

Los antecedentes consultados son un referente técnico importante para diseñar la plataforma IoT de este proyecto. La revisión de investigaciones previas (internacionales, nacionales y locales) permitió identificar arquitecturas, tecnologías y metodologías aplicadas en sistemas similares de monitoreo acuícola. La Tabla 1, resume la información más relevante de cada trabajo para facilitar su comparación.

Tabla 1 *Análisis comparativo de las soluciones tecnológicas*

Referencia	Área de estudio	Objetivo	Dispositivos	Plataforma	Protocolos	Arquitectura
(Naj & Sanzgiri, 2021)	Monitoreo y gestión de la calidad del agua en entornos acuáticos (India)	Diseñar un sistema IoT de bajo costo para la medición y visualización remota de parámetros fisicoquímicos en tiempo real.	Arduino Uno, Módulo Wi-Fi ESP8266, sensores de pH, turbidez y temperatura (DS18B20)	ThingSpeak (análisis en la nube)	IEEE 802.11 (Wi-Fi) y HTTP para transferencia de datos.	Basada en tres capas (Percepción, Red y Aplicación).
(Mohd Jais, Abdullah, Mohd Kassim, Abd Karim, & Muhadi, 2024)	Monitoreo de calidad de agua en el cultivo de lubina asiática en tanques de acuicultura. (Malasia)	Crear un sistema de monitoreo de precisión mediante la mejora de la exactitud de sensores de bajo costo utilizando regresión lineal simple y validación con equipos profesionales (YSI Professional Pro).	Microcontrolador, módulo Wi-Fi ESP8266 y sensores de bajo costo para la medición de pH, oxígeno disuelto (OD), gas amoníaco, conductividad eléctrica (salinidad) y temperatura.	ThingSpeak (análisis en la nube) y aplicación móvil Virtuino para la visualización de datos.	Wi-Fi (IEEE 802.11) para la transmisión de datos hacia la nube.	Basada en IoT para el monitoreo en tiempo real (Capa de adquisición con sensores, Transmisión inalámbrica, Procesamiento y almacenamiento en la nube, Visualización mediante aplicativo móvil).
(Beltrán Mindiola & Mejía Véliz, 2021)	Monitoreo automatizado de la calidad del agua en el sector acuícola (Ecuador)	Desarrollar prototipo IoT para medir pH, temperatura y UV en piscinas camaroneras	Arduino, sensores de temperatura, pH y la radiación ultravioleta	Plataforma web Thinger.io, ThingSpeak (análisis en la nube) y Android Studio para el desarrollo de app móvil.	WiFi y protocolo HTTP para el envío de datos a la API de ThingSpeak.	Arquitectura IoT estructurada en tres niveles: 1) Nivel de sensores (adquisición de datos), 2) Nivel de conectividad (transmisión inalámbrica vía Wi-Fi) y 3) Nivel de usuario

						(visualización en app móvil y entorno web).
(Carchipulla Leal, 2018)	Monitoreo automatizado de la calidad del agua en el sector camaronero de la provincia de El Oro. (Ecuador)	Analizar importancia del oxígeno disuelto en calidad de agua para *L. vannamei*	Sensores de oxígeno disuelto	Estudio teórico-experimental	No especificado	Experimental
(Pico-Lozano & Mendoza-Intriago, 2020)	Monitoreo IoT, sector Playa de Tarqui, Manta. (Manabí-ULEAM)	Evaluar calidad del agua del Río Manta y efectos en supervivencia larval	Sensores DO, salinidad, temperatura	Laboratorio	Manual	Experimental
(Ordoñez-Iglesias, Méndez-Martínez, & Zúñiga-Argudo, 2024)	Camarón (Manabí-ULEAM)	Evaluar efecto del balance iónico en dieta sobre salud de camarón en precría	Equipos de medición de calidad de agua	Laboratorio	Manual	Experimental

Nota: Autoría propia.

En la Tabla 1, se presenta un análisis comparativo de las soluciones tecnológicas documentadas en la literatura, incluyendo dispositivos, plataformas, arquitecturas y protocolos de comunicación utilizados para la gestión de datos en tiempo real y su almacenamiento histórico. Este ejercicio permitió identificar tendencias, mejores prácticas y oportunidades de mejora que han sido incorporadas en el diseño de la plataforma IoT ULEAM Guardian Shrimp.

2.2 Análisis de Plataformas IoT en el Contexto Universitario Ecuatoriano

Para dimensionar adecuadamente el aporte tecnológico y la innovación representada por la plataforma "IoT ULEAM Guardian Shrimp", se llevó a cabo un estudio comprehensivo del panorama digital de las principales instituciones de educación superior del Ecuador con enfoque en investigación aplicada y desarrollo tecnológico. Este análisis comparativo se sustentó en una metodología de revisión sistemática de portales web institucionales, repositorios digitales, sitios de facultades y centros de investigación, con el objetivo fundamental de cartografiar y categorizar las iniciativas públicas relacionadas con el Internet de las Cosas. El proceso investigativo se orientó específicamente a discriminar entre plataformas operativas de visualización de datos en tiempo real y otras modalidades de presencia digital de carácter informativo o repositorial, con ello se puede concluir que pocas instituciones poseen una plataforma similar a la que posee la ULEAM con su página que permite almacenar los datos y revisarlos en tiempo real de forma ordenada para su fácil interpretación sin una capa de complejidad agregada que pueda generar que las personas poco entendidas en el tema, entiendan fácilmente que tipo de datos están observando y con que propósito.

2.3 Bases teóricas

2.3.1 Fundamentos de la Acuicultura de Precisión y Calidad del Agua

A medida que la población humana ha crecido, ha aumentado la necesidad de producir bienes y servicios necesarios o deseados, lo que ha resultado en un mayor uso y contaminación del agua. La calidad del agua ha cobrado mayor importancia, ya que la cantidad de agua a menudo no puede evaluarse independientemente de su calidad. La calidad del agua es un factor crítico en el suministro de agua doméstico, agrícola e industrial, la producción pesquera y acuícola, la recreación acuática y la salud de los ecosistemas. Los profesionales de diversas disciplinas deben comprender los factores que controlan la concentración de las variables de la calidad del agua, así como sus efectos en los ecosistemas y los seres humanos. La gestión eficiente de los recursos hídricos requiere la aplicación de conocimientos sobre la calidad del agua.

La Acuicultura de Precisión (PA, por sus siglas en inglés, Precision Aquaculture) es la aplicación de tecnologías avanzadas y métodos de análisis de datos para monitorear, gestionar y optimizar la producción de organismos acuáticos (peces, camarones, moluscos, etc.) de manera individual o por grupos específicos, en lugar de tratar a toda la población de un estanque o jaula como una unidad uniforme, representa la aplicación de tecnologías avanzadas que incluyen sensores de Internet de las cosas, inteligencia artificial, visión artificial y análisis de datos para optimizar las operaciones de cultivo de peces a través del monitoreo en tiempo real, sistemas de alimentación automatizados y plataformas inteligentes de toma de decisiones.

Es la versión acuática de la agricultura de precisión, adaptada a los desafíos únicos del entorno submarino, se basa en un ciclo continuo de cuatro etapas principales que permiten pasar de la "intuición" del granjero a la "decisión basada en datos":

- Observar: Sensores y cámaras capturan datos en tiempo real.
- Interpretar: Algoritmos de Inteligencia Artificial (IA) analizan los datos para detectar patrones (ej. peces estresados o hambre).
- Decidir: El sistema recomienda o decide automáticamente la mejor acción (ej. detener el alimento).
- Actuar: Se ejecutan acciones precisas mediante maquinaria automatizada (ej. alimentadores automáticos).

Para que este sistema funcione, se integran diversas herramientas tecnológicas:

- **Sensores de Calidad de Agua:** Miden en tiempo real parámetros críticos como:
    - Oxígeno disuelto
    - Temperatura
    - Nivel de pH
    - Salinidad y turbidez.

2.3.2 Internet de las Cosas (IoT)

El Internet de las Cosas (IoT, por sus siglas en inglés) se define como una red de objetos físicos interconectados que incorporan sensores, software y tecnologías de conectividad para recopilar, procesar y compartir datos a través de Internet u otras redes. (IBM, 2025)

2.3.3 Python

Python es un lenguaje de programación ampliamente reconocido por su versatilidad y facilidad de uso. Se le describe técnicamente como un lenguaje de alto nivel, interpretado, multiparadigma y de propósito general, características que lo hacen

especialmente accesible tanto para principiantes como para desarrolladores experimentados. (Neville, 2025)

La convergencia entre el lenguaje de programación Python y el paradigma del Internet de las Cosas (IoT) ha generado un cambio significativo en el desarrollo de sistemas embebidos y soluciones de monitoreo. Debido a su sintaxis simplificada y su robusto ecosistema de bibliotecas, Python se ha posicionado como una herramienta versátil que permite escalar aplicaciones desde prototipos domésticos hasta infraestructuras industriales de alta complejidad. (Geeksforgeeks.org, 2025)

2.3.4 Backend

El backend o "lado del servidor" constituye la capa lógica donde reside la inteligencia del sistema. En proyectos de IoT, su función principal es la ingesta, procesamiento, validación y almacenamiento de las telemetrías enviadas por los nodos sensores (Fielding, 2000). A diferencia de los sistemas web tradicionales, un backend para IoT debe estar optimizado para el manejo de flujos de datos continuos y la gestión de protocolos de baja latencia.

2.3.5 Protocolo MQTT

El protocolo MQTT se define como un estándar de mensajería ligero, de código abierto y fundamentado en el modelo de publicación/suscripción (AWS, 2025). Originalmente desarrollado por IBM, este protocolo fue diseñado para comunicaciones máquina a máquina (M2M) en entornos con recursos de hardware limitados, bajo ancho de banda o redes de alta latencia (Banks & Gupta, 2014). A diferencia de los protocolos convencionales, MQTT minimiza la sobrecarga de red mediante cabeceras reducidas, optimizando la eficiencia energética en microcontroladores y computadoras de placa reducida.

2.3.6 Arquitectura de Publicación y Suscripción

La arquitectura de MQTT rompe el esquema tradicional cliente-servidor y se organiza mediante tres entidades fundamentales:

Publicador (Publisher): El nodo encargado de capturar y transmitir los datos hacia un tema o "tópico" específico (AWS, 2025).

Suscriptor (Subscriber): La entidad que recibe la información de interés de manera automática tras haberse suscrito a un tópico (AWS, 2025).

Broker: Actúa como el servidor intermediario que gestiona el tráfico de mensajes, asegurando que cada paquete llegue al suscriptor correcto según el tópico definido (Banks & Gupta, 2014).

2.3.7 Calidad de Servicio (QoS)

Para garantizar la integridad de los datos en aplicaciones críticas como la acuicultura, MQTT ofrece tres niveles de Calidad de Servicio (QoS) (Banks & Gupta, 2014):

- **QoS 0 (At most once)**: El mensaje se envía una sola vez sin confirmación de recepción. (Banks & Gupta, 2014).
- **QoS 1 (At least once)**: Asegura que el mensaje se entregue al menos una vez al destinatario. (Banks & Gupta, 2014).
- **QoS 2 (Exactly once):** Es el nivel máximo de seguridad que garantiza la entrega exacta de un solo mensaje, evitando duplicados (Banks & Gupta, 2014).

2.4 Android Studio

Android Studio es el Entorno de Desarrollo Integrado (IDE) oficial para el desarrollo de aplicaciones en la plataforma Android, basado en el software IntelliJ IDEA de JetBrains. Esta herramienta proporciona a los desarrolladores un ecosistema unificado que integra funciones avanzadas de edición de código, depuración, pruebas y emulación, optimizando el ciclo de vida del desarrollo de software móvil. (Google Developers, 2024)

2.4.1 Lenguaje de programación Kotlin

Kotlin es un lenguaje de programación de código abierto, multiplataforma y de tipado estático, diseñado por la empresa JetBrains para interoperar completamente con la sintaxis de Java y ejecutarse sobre la Máquina Virtual de Java (JVM). Este lenguaje combina los paradigmas de programación orientada a objetos y funcional, proporcionando una sintaxis concisa que reduce significativamente la cantidad de código redundante en comparación con sus predecesores. (Kotlin, 2026)

2.5 Hardware

El hardware constituye la infraestructura física y tangible que sustenta la ejecución de los procesos lógicos y el procesamiento de datos en un sistema computacional.1 En el ecosistema del Internet de las Cosas (IoT), el hardware trasciende la arquitectura convencional de una computadora personal para integrarse en dispositivos especializados como el Raspberry Pi 4 que actúan como unidades de control y adquisición de señales. Estos componentes físicos son los encargados de realizar la interfase directa con el entorno mediante la captura de variables analógicas y su posterior conversión en registros digitales procesables por el software. (TME ELECTRONICS COMPONENTS, 2025)

2.5.1 Raspberry Pi

La Raspberry Pi es un computador completo en una placa compacta, ideal para proyectos de automatización y adquisición de datos. Posee puertos de entrada/salida de propósito general (GPIO) que permiten conectar directamente sensores, actuadores y otros dispositivos. (Raspberry Pi, 2025)

Función en el sistema: En la arquitectura de tu proyecto, se encarga de:

- Adquirir datos de los sensores conectados
- Procesar la información (por ejemplo, comparar los valores con umbrales predefinidos para activar alarmas).
- Comunicar los datos a la plataforma en la nube o servidor web.
- Gestionar actuadores (como motores para aireación) en respuesta a los datos recibidos, optimizando el consumo energético del sistema

Figura 4

Raspberry Pi 4 Model B - 8GB

Nota: Raspberry Pi 4, tomado de DFRobot, https://www.dfrobot.com/product-2028.html.

2.5.2 Placa HAT de expansión

Como parte integral del hardware, se utiliza la placa **HAT de expansión de E/S para Raspberry Pi 5 / 4B / 3B+**, diseñada específicamente para ampliar las capacidades de interfaz de la Raspberry Pi 4. Su función principal es servir como centro de conexiones (hub) para los sensores de la serie Gravity, optimizando el cableado y la estabilidad de las señales. (DFRobot, 2026)

Especificaciones Técnicas Detalladas:

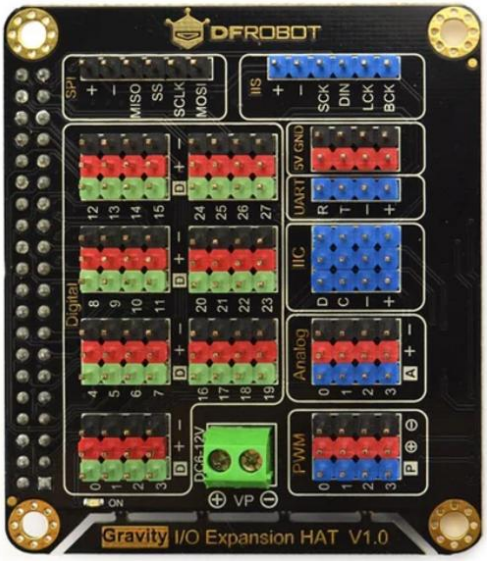
- **Compatibilidad Total:** Diseñada para integrarse perfectamente con Raspberry Pi 4B, 3B+ y modelos anteriores, extendiendo todos los pines GPIO a conectores estandarizados.
- **Puertos de Salida:** Proporciona acceso directo a puertos digitales, analógicos (mediante el MCU integrado), PWM, I2C, UART, SPI e IIS.

Gestión de Voltaje:

- **Voltaje de Operación:** 5V (suministrados por la Raspberry Pi).
- **Alimentación Externa PWM:** Dispone de una entrada para fuente externa de 6V a 12V, permitiendo el control de múltiples servomotores sin sobrecargar la placa base.
- **Conversión Analógica:** Incluye un microcontrolador (MCU) que actúa como puente para convertir señales analógicas en digitales a través del protocolo I2C, permitiendo que la Raspberry Pi "lea" los sensores de pH y oxígeno disuelto.
- **Diseño Físico:** Sus dimensiones son de 65 x 56 mm, manteniendo el factor de forma estándar de las placas HAT para Raspberry Pi.

Figura 5

HAT de expansión de E/S para Raspberry Pi 5 / 4B / 3B+



Nota: Placa HAT de expansión, tomado de DFRobot, https://www.dfrobot.com/product-1930.html.

2.5.3 Sensor de pH

El pH es un parámetro crítico que afecta directamente la fisiología, metabolismo y supervivencia de las larvas de camarón. El sensor de pH empleado opera bajo el principio electroquímico de potencial de unión líquida, donde una celda electroquímica genera un voltaje proporcional a la concentración de iones hidrógeno en el agua. (DFRobot, 2026)

- **Principio de funcionamiento**: Electrodo de vidrio que mide la diferencia de potencial entre un electrodo de referencia interno y el medio acuoso.
- **Rango de medición**: 0-14 pH con resolución de 0.01 pH.
- **Exactitud**: ±0.1 pH en el rango de 7-9 pH (óptimo para cultivo de camarón).
- **Temperatura de operación**: 0-60°C.
- **Compensación de temperatura**: Automática mediante termistor integrado.

- **Interfaz de salida**: Analógica (0-5V) o digital (I2C/SPI según modelo).

Figura 6

Sensor/Medidor de pH Analógico V

Nota: Sensor de pH, tomado de DFRobot, https://www.dfrobot.com/product-2069.html.

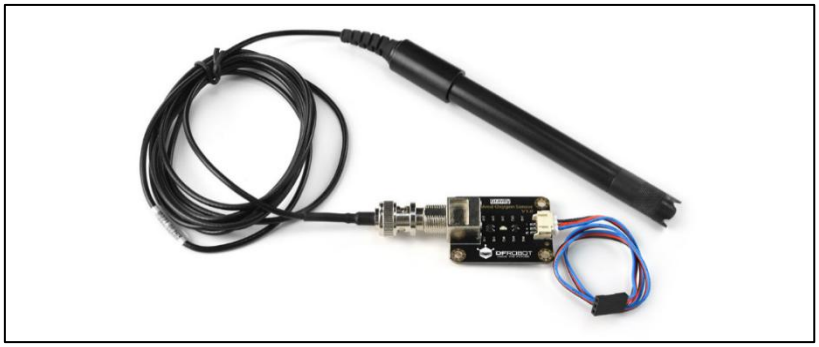
2.5.4 Sensor de Oxígeno Disuelto (OD)

La concentración de oxígeno disuelto es el factor más limitante en sistemas intensivos de cultivo larval. El sensor utilizado se basa en la tecnología óptica de luminiscencia (método Clark), preferida sobre los electroquímicos por su menor mantenimiento y mayor estabilidad. (DFRobot, 2026)

- **Principio de funcionamiento**: Medición de la extinción de luminiscencia de un material fotoactivo en presencia de oxígeno.
- **Rango de medición**: 0-20 mg/L con resolución de 0.01 mg/L.
- **Exactitud**: ±0.2 mg/L en el rango de 2-8 mg/L (crítico para larvas).
- **Tiempo de respuesta**: <30 segundos para el 90% del valor final.
- **Compensación automática**: Por temperatura y salinidad.
- **Calibración**: Requiere calibración a punto cero (solución sin oxígeno) y a saturación (aire ambiente).

Figura 7

Sensor y medidor analógico de oxígeno disuelto



Nota: Sensor de oxígeno disuelto, tomado de DFRobot, https://www.dfrobot.com/product-1628.html.

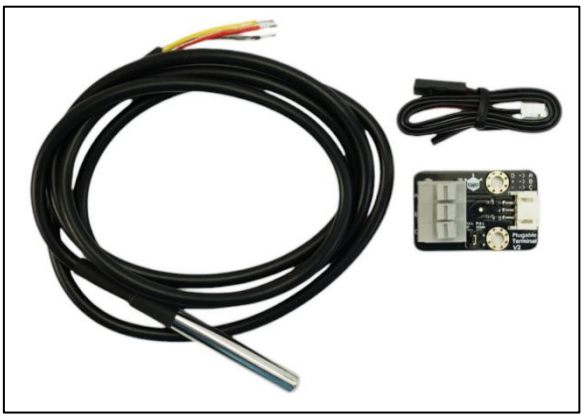
2.5.5 Sensor de Temperatura

La temperatura del agua regula todas las reacciones bioquímicas en organismos acuáticos. Se emplea un sensor de temperatura digital de alta precisión y bajo drift temporal. (DFRobot, 2026)

- **Tecnología**: Termistor de estado sólido o sensor digital DS18B20.
- **Rango de medición**: -10°C a +85°C con resolución de 0.0625°C.
- **Exactitud**: ±0.5°C en el rango de 10-30°C.
- **Tiempo de respuesta**: <2 segundos en agua en movimiento.
- **Protección**: Carcasa estanca IP68 para inmersión continua.
- **Interfaz**: Digital one-wire (1-Wire) para simplificación de cableado.

Figura 8

Sensor de temperatura DS18B20



Nota: Sensor de temperatura, tomado de DFRobot, https://www.dfrobot.com/product-1354.html .

Capitulo III

3.1 Metodología

3.2 Técnicas de recolección de requisitos

Para identificar las necesidades reales del Laboratorio de Innovación en Sistemas Acuícolas, se realizaron entrevistas semiestructuradas y observación directa. Estas técnicas nos permitieron comprender el proceso actual de monitoreo, sus limitaciones y las expectativas de un sistema automatizado.

3.2.1 Entrevista semiestructurada

La entrevista semiestructurada combina preguntas cerradas y abiertas, permitiendo profundizar con preguntas de seguimiento ("¿por qué?", "¿cómo?") según las respuestas del entrevistado (Adams, 2015). Este enfoque facilitó conversaciones naturales donde los usuarios expresaron problemas que no habían articulado previamente. En la tabla 2 se presenta el perfil de los entrevistados.

Tabla 2 *Perfil de los participantes de la entrevista*

Rol	**Código**	**Experiencia**	**Fecha**
Director del laboratorio	Entrevistado A	30 años	10/06/2025
Técnico del laboratorio	Entrevistado B	30 años	10/06/2025
Ayudante del laboratorio	Entrevistado C	10 años	10/06/2025

Nota: Autoría propia.

Estructura de la entrevista:

La guía de entrevista se organizó en tres bloques temáticos:

1. **Proceso actual de monitoreo**

¿Cómo monitorean las variables del agua actualmente?

¿Cada cuánto tiempo realizan mediciones?

¿Qué equipos utilizan y cómo registran los datos?

2. **Problemas y necesidades identificadas**

¿Qué limitaciones presenta el método manual?

¿Han experimentado eventos de mortalidad larval? ¿Cuál fue la causa?

¿Qué información necesitan que actualmente no tienen?

3. **Expectativas de automatización**

¿Qué variables consideran prioritarias para monitorear automáticamente?

¿Desde dónde necesitarían acceder a los datos?

¿Qué tipo de alertas necesitarían recibir?

Cada entrevista tuvo una duración promedio de 45 minutos y se realizó en las instalaciones del LISA para que los entrevistados pudieran mostrar el proceso actual y los instrumentos utilizados. Como se muestra en las figuras 9, 10 y 11.

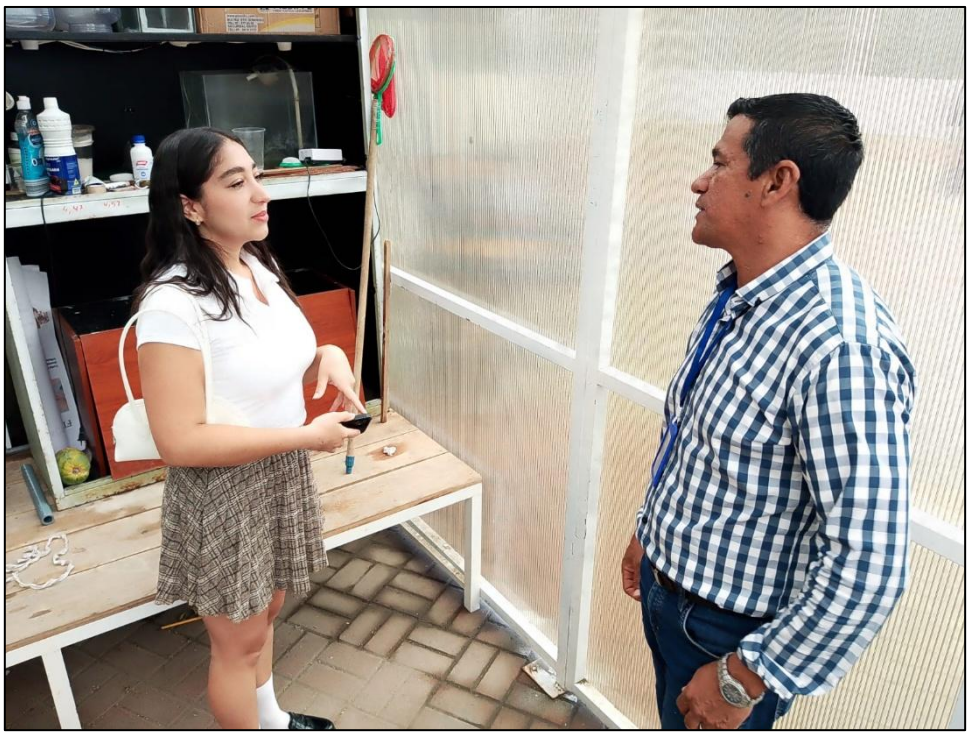
Figura 10

La imagen representa la entrevista junto con el Entrevistado A.

Nota: Autoría propia.

Figura 9

La imagen representa la entrevista junto con el Entrevistado B



Nota: Autoría propia.

Figura 11

La imagen representa la entrevista junto con el Entrevistado C

Nota: Autoría propia.

Hallazgos principales:

Los tres entrevistados coincidieron en tres problemas críticos:

Ventanas ciegas en el monitoreo: Las mediciones se realizan solo tres veces al día (8 AM, 2 PM, 6 PM), dejando 14 horas nocturnas sin supervisión. El Entrevistado B expresó que se pueden generar pérdidas de un lote completo debido a una caída de oxígeno disuelto que ocurra de madrugada.

Registros manuales poco confiables: Los datos se anotan en cuaderno y luego se transcriben a Excel, introduciendo errores y omisiones. El Entrevistado A señaló que esta doble entrada manual dificulta el análisis de tendencias para investigación.

Falta de alertas tempranas: Cuando detectan un problema durante la medición manual, ya transcurrieron horas desde que ocurrió. El Entrevistado C indicó que necesitan

conocer desviaciones cuando el pH está en 8.6 (advertencia), no cuando ya llegó a 9.0 (crítico).

3.2.2 Observación directa

Se realizaron sesiones de observación en el LISA (15, 22 y 29 de julio de 2025) durante las rutinas de medición. Esto permitió identificar necesidades que los usuarios no mencionaron explícitamente en las entrevistas.

Observaciones clave:

El proceso manual de medición en un tanque toma 30 minutos. Durante este tiempo, si el tanque tiene un problema, la respuesta se retrasa.

Cuando corrigen un problema (ej: disminución del Oxígeno Disuelto), esperan 15-30 minutos para volver a medir manualmente y verificar si funcionó la corrección.

Estas observaciones derivaron en requisitos adicionales: actualización de datos en tiempo real durante eventos críticos y alertas para cuando los valores alcancen puntos críticos.

3.2.3 Síntesis de requisitos

Las técnicas aplicadas permitieron identificar requisitos del sistema:

Requisitos funcionales prioritarios:

- Monitoreo continuo 24/7 de pH, oxígeno disuelto y temperatura
- Almacenamiento automático de datos con timestamp
- Sistema de monitoreo web en tiempo real con datos históricos
- Aplicación móvil para acceso desde cualquier ubicación dentro de ULEAM
- Aplicación móvil con sistema de alertas y umbrales configurables

- Notificaciones push a smartphones
- Visualización gráfica de variables en tiempo real.

Requisitos no funcionales:

- Disponibilidad >99% (máximo 7 horas inactivo al año)
- Precisión comparable a instrumentos portátiles actuales
- Interfaz intuitiva para usuarios sin formación técnica
- Escalabilidad hasta 20 tanques

Estos requisitos fueron validados en una reunión con los tres entrevistados (30/06/2025), quienes confirmaron que reflejaban sus necesidades reales.

3.3 Introducción a la Metodología

Para la implementación de este proyecto se requirió la adopción de un enfoque metodológico capaz de responder eficientemente a los desafíos inherentes al desarrollo de soluciones de Internet de las Cosas (IoT) en entornos de investigación aplicada. En este contexto, a continuación, se describe un mecanismo de evaluación y elección de la metodología.

3.4 Evaluación y Elección de la metodología

Tabla 3

Comparación de metodologías evaluadas

Criterio	**Cascada**	**Scrum**	**XP**	**Justificación**
Manejo de cambios	Bajo	Alto	Muy Alto	Requisitos evolucionaron constantemente
Integración hardware	Bajo	Medio	Alto	XP permite iteración rápida con sensores físicos
Tamaño de equipo	Grande	Medio	Pequeño	Equipo de 2 personas ideal para XP
Énfasis en calidad	Bajo	Medio	Muy Alto	TDD crítico para datos de supervivencia

Entregas frecuentes	NO	Si	Si	Demostraciones cada 2 semanas validadas por LISA

Nota: Autoría propia.

Tras evaluar diversas alternativas como se muestra en la tabla 3, se seleccionó la Programación Extrema (XP) como marco de trabajo principal, debido a su capacidad para gestionar requisitos dinámicos, su énfasis en la calidad técnica y su orientación hacia la entrega continua de valor.

XP se fundamenta en las siguientes características específicas las cuales aportan significativamente al desarrollo del proyecto:

Naturaleza experimental: La integración de sensores especializados (pH, oxígeno disuelto y temperatura) con plataformas de hardware (Raspberry Pi) y software presentaba incertidumbres técnicas que demandaban un enfoque iterativo e incremental.

Entorno de investigación: El Laboratorio de Innovación en Sistemas Acuícolas constituye un espacio de innovación donde los requerimientos evolucionan según los hallazgos experimentales; por ello, se requería una metodología que facilitara la incorporación de cambios incluso en etapas avanzadas del desarrollo.

Componente multidisciplinario: El proyecto integró perfiles diversos (desarrolladores, técnicos e investigadores acuícolas), lo cual exigía un marco de comunicación efectivo y procesos de colaboración rigurosamente definidos.

Alta criticidad de los datos: Dado que la precisión y confiabilidad de las mediciones físico-químicas son vitales para la supervivencia de las larvas de camarón, se

precisaban prácticas de desarrollo que garantizaran la calidad del software desde su concepción.

La metodología XP trasciende las prácticas convencionales de desarrollo al facilitar la creación de soluciones tecnológicas funcionales mediante ciclos cortos y frecuentes, manteniendo el enfoque en las necesidades reales de los usuarios finales. Asimismo, su implementación en este contexto académico permitió validar la efectividad de las metodologías ágiles en el desarrollo de sistemas embebidos y monitoreo en tiempo real, demostrando que los principios de la Programación Extrema pueden adaptarse exitosamente más allá del desarrollo de software tradicional.

3.4.1 Fundamentos de la Programación Extrema (XP)

La Programación Extrema se basa en cinco valores fundamentales:

- **Comunicación:** Diálogo constante entre el equipo de desarrollo, investigadores y técnicos del LISA.
- **Simplicidad:** Desarrollo de la solución más simple que funcione, evitando complejidades innecesarias.
- **Retroalimentación (Feedback):** Evaluación continua del software a través de pruebas automatizadas y opinión de los usuarios.
- **Coraje:** Para enfrentar cambios en los requisitos o rediseñar componentes cuando fuera necesario.
- **Respeto:** Colaboración y confianza entre todos los involucrados.

3.4.2 Ciclo de Vida y Fases de XP

El desarrollo se estructuró en cinco fases dinámicas que permitieron una transición fluida desde la teoría a la implementación física:

Exploración: Definición de funcionalidades iniciales mediante historias de usuario y validación de la comunicación entre la Raspberry Pi 4 y los sensores.

Planificación: Priorización de funcionalidades junto a los investigadores del LISA, estableciendo un cronograma de entregas frecuentes o releases.

Iteraciones: Ciclos cortos (1-2 semanas) donde se aplicó el diseño, codificación y pruebas de cada variable (pH, OD y temperatura).

Producción: Despliegue del sistema en el entorno real del laboratorio y ejecución de pruebas de estabilidad 24/7.

Mantenimiento: Ajustes basados en el uso real de los técnicos y optimización de las interfaces de visualización.

3.4.3 Artefactos de la Metodología: Historias de Usuario

La planificación del sistema "IoT ULEAM Guardian Shrimp" se fundamentó en la creación de Historias de Usuario, las cuales actuaron como el artefacto principal para capturar los requerimientos funcionales desde la perspectiva del director, técnico y encargado del LISA. Estas historias permitieron descomponer la complejidad técnica del monitoreo de larvas en unidades de trabajo manejables y priorizadas según su valor para la supervivencia del cultivo.

La Tabla 4, presenta el conjunto de historias de usuario que guiaron el desarrollo iterativo. Cada historia sigue una estructura estándar de Rol + Acción + Resultado. Esto asegura que cada funcionalidad implementada tenga un propósito claro y validado por los técnicos del laboratorio.

Tabla 4

Historias de Usuario para el sistema IoT ULEAM Guardian Shrimp

ID	Historia de Usuario	Descripción	Prioridad
HU-01	Monitoreo Remoto	Como investigador, quiero ver el pH, OD y temperatura en la web, para supervisar el laboratorio sin estar presente.	Alta
HU-02	Alerta de Oxígeno	Como técnico, quiero recibir una notificación si el OD baja de 4 mg/L, para activar los aireadores de emergencia.	Alta
HU-03	Análisis de Históricos	Como técnico, quiero visualizar gráficas de tendencias de 24 horas, para identificar variaciones químicas nocturnas.	Media
HU-04	Registro de Errores	Como técnico del LISA, quiero saber si un sensor falla, para no tomar decisiones	Alta

		basadas en datos erróneos	
HU-05	Configurar Umbrales	Como administrador, quiero cambiar límites de alerta por variable	Media
HU-06	Estado de Sensores	Como técnico, quiero ver qué sensores están activos/inactivos	Alta
HU-07	Exportar Datos	Como investigador, quiero descargar datos en Excel	Media
HU-08	Modo Mantenimiento	Como técnico, quiero apagar el monitoreo durante limpieza	Baja

Nota: Autoría propia.

3.4.4 Prácticas XP Implementadas en el Proyecto

Planificación Incremental

Se elaboró un plan de releases o entregas parciales, priorizando funcionalidades clave como:

- Adquisición de datos desde sensores.
- Comunicación MQTT con el servidor.
- Visualización de datos en la plataforma web integrada.
- Visualización de datos en la aplicación móvil.
- Cada release se planificó en iteraciones cortas (1-2 semanas), con reuniones de planificación en las que participaron los stakeholders del LISA.

Diseño Simple

A lo largo del ciclo de desarrollo se aplicó rigurosamente el principio de diseño simple (simplicity), uno de los pilares fundamentales de la Programación Extrema. Esta práctica consistió en mantener el sistema en su estado más funcional y comprensible durante cada iteración, evitando la incorporación de complejidades prematuras o funcionalidades no esenciales (sobreingeniería). La prioridad se centró en construir exclusivamente los componentes necesarios para satisfacer los requerimientos vigentes, garantizando así un código limpio, mantenible y con alta capacidad de evolución.

Un ejemplo concreto de esta metodología fue la definición de una arquitectura modular. Este enfoque permitió el desacoplamiento de las diversas capas del sistema, asegurando que el núcleo de procesamiento y comunicación permaneciera estable e independiente de los componentes periféricos. En la práctica, esto se materializó

mediante la creación de interfaces de comunicación estandarizadas y módulos de software específicos para la gestión de sensores. Debido a esta estructura, la integración de nuevos dispositivos como sensores de amonio o salinidad no requiere la modificación del código base; en su lugar, se desarrolla un módulo independiente que se integra a través de las interfaces predefinidas. Ver figura 12.

Ejemplo:

Figura 12

Código de sensores implementados

```python
# Estructura modular
class SensorManager:
    def read_ph(self) -> Optional[float]:
        """Lee solo pH - responsabilidad única"""
        voltaje = self.read_adc_voltage(self.board.A2)
        if voltaje is None:
            return None
        ph = 7 + ((2.5 - voltaje) / 0.18)
        return round(ph, 2)

    def read_dissolved_oxygen(self) -> Optional[float]:
        """Lee solo oxígeno disuelto - responsabilidad única"""
        voltaje = self.read_adc_voltage(self.board.A3)
        if voltaje is None:
            return None
        mg_L = (voltaje / 3.0) * 20.0
        return round(mg_L, 2)
```

Nota: Autoría propia.

Ventaja real: Cuando el LISA pida agregar un sensor de salinidad en el futuro, solo se necesita, agregar un módulo extra para el nuevo sensor, como lo podemos evidencias en la figura 13.

Figura 13

Código de ejemplo para futuras implementaciones de nuevos sensores

```
def read_salinity(self) -> Optional[float]:
        """Nuevo sensor - sin tocar código existente"""
        voltaje = self.read_adc_voltage(self.board.A4)
        if voltaje is None:
            return None
        salinity = calcular_salinidad(voltaje)
        return round(salinity, 2)
```

Nota: Autoría propia.

Esta estrategia no solo optimizó el tiempo de desarrollo en las iteraciones iniciales al prescindir de infraestructuras excesivamente complejas, sino que también garantizó la escalabilidad y flexibilidad del sistema a largo plazo. Por consiguiente, el diseño simple no resultó en una solución limitada, sino en una arquitectura robusta y eficiente que permite un crecimiento ordenado y sostenible del proyecto.

3.4.5 Desarrollo Guiado por Pruebas (TDD)

El Desarrollo Guiado por Pruebas (Test-Driven Development, TDD) se adoptó como una práctica fundamental durante el ciclo de vida del software, constituyendo un pilar para garantizar la calidad, confiabilidad y mantenibilidad del código. La implementación de TDD siguió un ciclo iterativo riguroso de tres fases, conocido como el modelo Red-Green-Refactor:

Fase Red (Rojo): En esta etapa inicial, previa a la escritura de código de producción, se desarrollaron pruebas unitarias automatizadas que definían el comportamiento esperado de una funcionalidad específica. Al ejecutar el conjunto de pruebas, esta nueva unidad fallaba de forma controlada lo que validaba que la funcionalidad aún no había sido implementada y que la prueba era capaz de detectar su ausencia.

Fase Green (Verde): Posteriormente, se redactó exclusivamente el código de producción mínimo necesario para que la prueba fuera superada satisfactoriamente. El objetivo en esta fase no fue la optimización técnica, sino alcanzar una implementación funcional que cumpliera con los requerimientos definidos en la prueba.

Fase Refactor (Refactorización): Una vez superada la prueba, se procedió a optimizar el código generado. Esta fase se centró en eliminar duplicidades, mejorar la legibilidad y aplicar principios de diseño limpio (Clean Code), asegurando en todo momento que el sistema permaneciera en estado "verde" y sin alteraciones en su comportamiento funcional. Ver figura 14.

Ejemplo práctico:

Figura 14

Código de ejemplo TDD

```python
# PASO 1: Escribir prueba PRIMERO (Red)
def test_read_ph_valid_voltage():
    """Prueba: Voltaje válido debe retornar pH entre 0-14"""
    board_mock = Mock()
    board_mock.get_adc_value.return_value = 2048  # Voltaje medio

    sensor_manager = SensorManager(board_mock)
    ph = sensor_manager.read_ph()

    assert ph is not None
    assert 0 <= ph <= 14
    assert isinstance(ph, float)

# Esta prueba FALLA porque read_ph() no existe todavía

# PASO 2: Escribir código mínimo (Green)
def read_ph(self) -> Optional[float]:
    voltaje = self.read_adc_voltage(self.board.A2)
    if voltaje is None:
        return None
    ph = 7 + ((2.5 - voltaje) / 0.18)
    return round(ph, 2)

# Ahora la prueba PASA

# PASO 3: Refactorizar (Refactor)
```

Nota: Autoría propia.

Otra prueba crítica:

Figura 15

Código de ejemplo de desconexión accidental

```python
def test_read_ph_sensor_failure():
    """Prueba: Sensor desconectado debe retornar None, no crashear"""
    board_mock = Mock()
    board_mock.get_adc_value.side_effect = Exception("Sensor disconnected")

    sensor_manager = SensorManager(board_mock)
    ph = sensor_manager.read_ph()

    assert ph is None  # Sistema robusto, no crashea
    assert sensor_manager.sensor_status['ph'] == False
```

Nota: Autoría propia.

Esta prueba que se presenta en la figura 15 es fundamental. Cuando un técnico desconecté el sensor de pH, el sistema continúa funcionando y registra el error en los logs. Sin TDD, el programa habría crasheado.

3.4.6 Refactorización Continua

El código del sistema fue sometido a un proceso de mejora continua y progresiva a lo largo de todas las etapas del desarrollo, aplicando el principio de refactorización. Esta práctica consistió en revisar y optimizar periódicamente el código existente con el objetivo fundamental de mantenerlo limpio, eficiente y altamente legible. La refactorización se realizó de forma iterativa, coincidiendo con la finalización de cada funcionalidad y, muy especialmente, durante la integración de nuevos componentes hardware al sistema.

Un ejemplo concreto de este proceso se dio durante la incorporación de los sensores especializados de pH y oxígeno disuelto. Inicialmente, el código para la lectura de sensores funcionaba correctamente, pero presentaba una estructura monolítica donde toda la lógica de adquisición de datos estaba interconectada. Al identificar que esta

arquitectura dificultaría el mantenimiento y la futura incorporación de nuevos sensores, se procedió a refactorizar el código aplicando el principio de responsabilidad única.

Se crearon módulos independientes y especializados para cada tipo de sensor, estableciendo interfaces estandarizadas que permitieron desacoplar completamente la lógica de adquisición de datos del procesamiento central. Esta reorganización no solo mejoró significativamente la legibilidad y mantenibilidad del código, sino que también incrementó su eficiencia al eliminar redundancias y optimizar los procesos de lectura.

Además, se implementaron patrones de diseño consistentes y se mejoró la documentación interna (comentarios y documentación de funciones), facilitando que cualquier desarrollador del equipo pudiera comprender rápidamente la estructura del sistema. Esta atención meticulosa a la calidad del código resultó fundamental para garantizar la estabilidad y confiabilidad del sistema, especialmente considerando la criticidad de las mediciones para la supervivencia de las larvas de camarón.

La refactorización continua demostró ser una práctica esencial que, lejos de representar un esfuerzo adicional, constituyó una inversión estratégica que pagó dividendos en forma de reducción en costos de mantenimiento, mayor facilidad para implementar nuevas funcionalidades y una base de código más robusta y preparada para evolucionar junto con las necesidades del proyecto.

3.4.7 Integración Continua

En lugar de depender de herramientas de automatización externas, la Integración Continua se gestionó directamente en el hardware mediante la configuración de un servicio de sistema en la Raspberry Pi. Este servicio se programó para inicializarse automáticamente con el sistema operativo, garantizando la transmisión ininterrumpida de datos hacia el broker.

Esta arquitectura permitió un ciclo de mejora continua: ante la necesidad de realizar ajustes o actualizaciones en el código, el servicio podía detenerse momentáneamente, integrar los cambios y reanudarse de forma inmediata. Esta práctica aseguró que las nuevas funcionalidades se incorporaran al entorno de ejecución de manera controlada, minimizando los tiempos de inactividad y validando la estabilidad del sistema en cada actualización. La implementación del servicio lo podemos evidenciar en las figuras 16 y 17.

Figura 16

La imagen representa la implementación del servicio con un muestreo inicial de una hora

iot-sensor-system.service

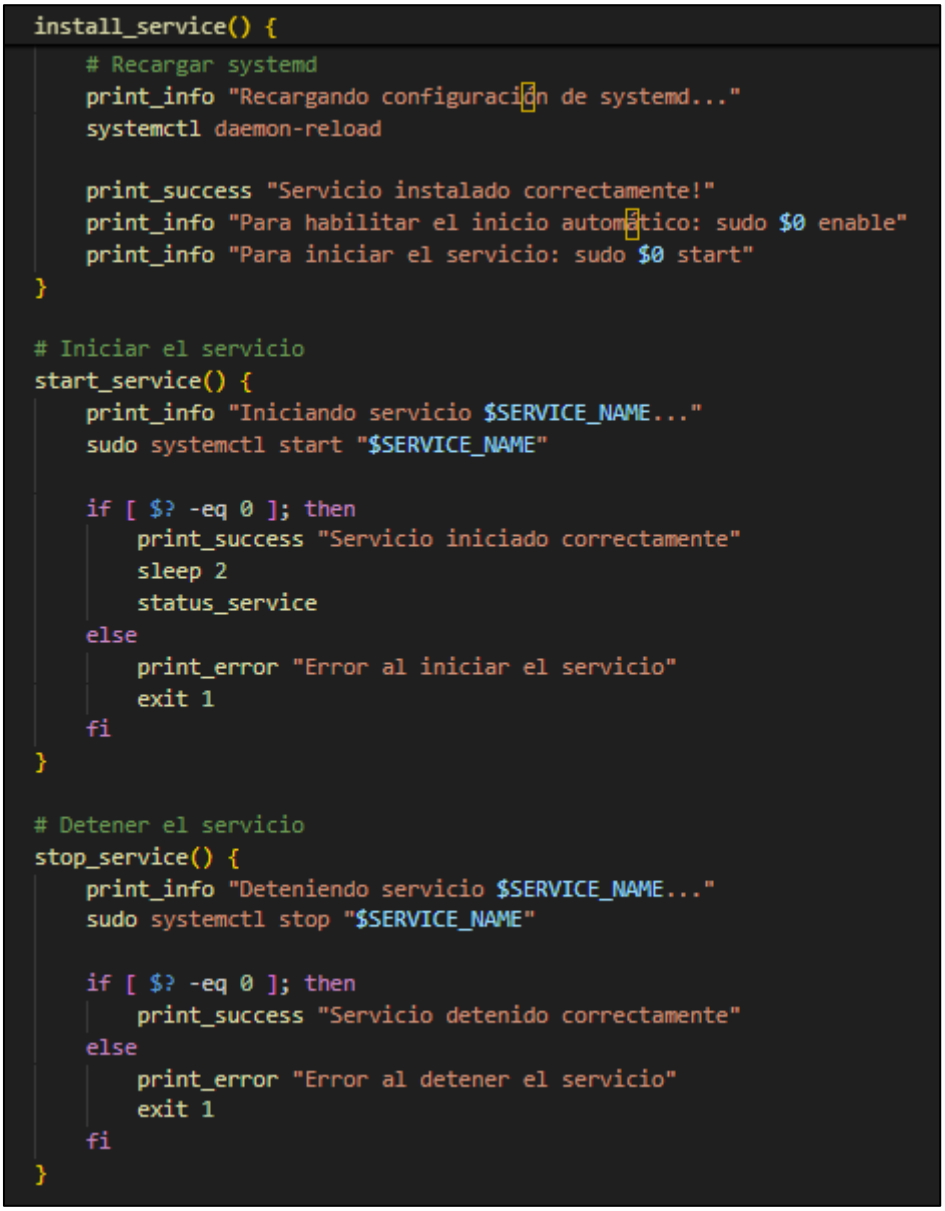
C: > Users > develoStefany > Downloads > files > iot-sensor-system.service

```
[Unit]
Description=Sistema de Monitoreo IoT con Sensores
After=network-online.target
Wants=network-online.target

[Service]
Type=simple
User=pi
Group=pi
WorkingDirectory=/home/pi/iot_sensors
ExecStart=/usr/bin/python3 /home/pi/iot_sensors/iot_sensor_system.py --interval 3600
Restart=always
RestartSec=10
StandardOutput=journal
StandardError=journal

# Variables de entorno si son necesarias
Environment="PYTHONUNBUFFERED=1"

# Límites de recursos (opcional)
LimitNOFILE=65536

[Install]
WantedBy=multi-user.target

```

Nota: Autoría propia.

Figura 17

La imagen representa ejemplos de cómo se realiza la recarga, inicio y detención del servicio



```
install_service() {
    # Recargar systemd
    print_info "Recargando configuración de systemd..."
    systemctl daemon-reload

    print_success "Servicio instalado correctamente!"
    print_info "Para habilitar el inicio automático: sudo $0 enable"
    print_info "Para iniciar el servicio: sudo $0 start"
}

# Iniciar el servicio
start_service() {
    print_info "Iniciando servicio $SERVICE_NAME..."
    sudo systemctl start "$SERVICE_NAME"

    if [ $? -eq 0 ]; then
        print_success "Servicio iniciado correctamente"
        sleep 2
        status_service
    else
        print_error "Error al iniciar el servicio"
        exit 1
    fi
}

# Detener el servicio
stop_service() {
    print_info "Deteniendo servicio $SERVICE_NAME..."
    sudo systemctl stop "$SERVICE_NAME"

    if [ $? -eq 0 ]; then
        print_success "Servicio detenido correctamente"
    else
        print_error "Error al detener el servicio"
        exit 1
    fi
}
```

Nota: Autoría propia.

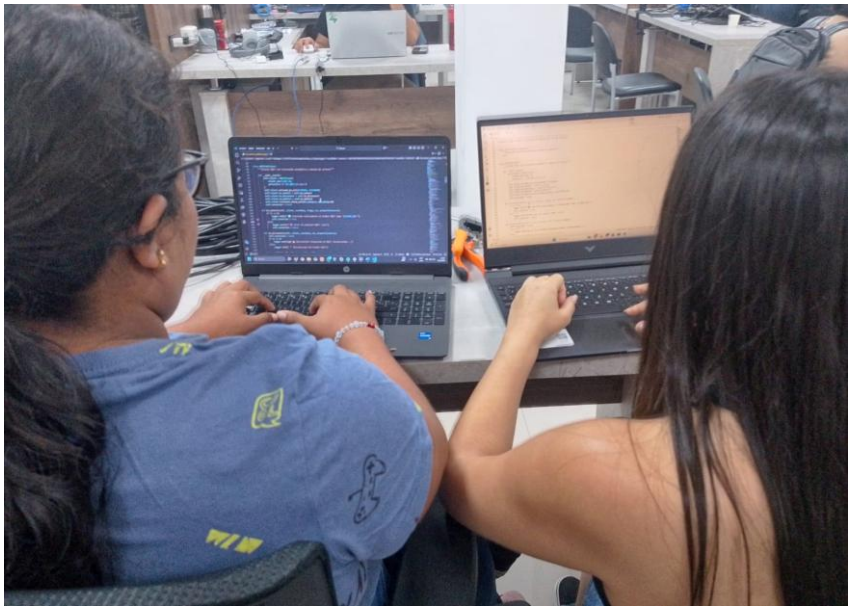
3.4.8 Programación en Parejas (Pair Programming)

Durante las fases críticas del desarrollo tales como la configuración del protocolo MQTT, se implementó la programación en parejas. Esta técnica facilitó la revisión de código en tiempo real, lo que permitió identificar errores de lógica de manera temprana y asegurar que el diseño de las interfaces de comunicación cumpliera con los estándares de seguridad y eficiencia requeridos por el proyecto.

- Durante la semana del 1 al 5 de septiembre de 2025, trabajamos en parejas Stefany Alonzo con Joseline Cedeño para configurar el broker MQTT. En 4 horas de sesión, identificamos 3 errores de configuración que habrían tomado días depurar individualmente. La sesión resultó en una conexión estable que no ha fallado desde entonces. Ver figura 18.

Figura 18

La imagen representa programación en parejas en la configuración del Broker MQTT



Nota: Autoría propia.

3.4.9 Roles del Equipo (Whole Team)

En el marco de la metodología XP, se establecieron roles claros para garantizar que el desarrollo técnico estuviera alineado con las necesidades biológicas del proyecto desarrollado. Ver tabla 5.

Tabla 5

En la tabla se muestran los roles, responsables y descripciones del equipo

Rol	Responsable	Descripción
Cliente (Customer)	Director y técnicos del LISA	Definieron los requerimientos (historias de usuario) y validaron la precisión de los sensores.
Programador	Autoras de la tesis	Encargadas de la implementación de hardware (Raspberry Pi), Protocolo de comunicación (MQTT) y el software del sistema.
Coach	Tutor de Tesis	Supervisó la aplicación de la metodología y el cumplimiento del cronograma académico.

Nota: Autoría propia.

3.4.10 Conclusión de la metodología implementada

La implementación de la Programación Extrema como metodología fue una decisión estratégica que potenció la eficiencia y adaptabilidad del proceso de desarrollo. La aplicación sistemática de sus valores y prácticas, como el TDD y el diseño simple, garantizó la entrega de un software robusto, escalable y alineado con las exigencias del entorno acuícola. La capacidad del equipo para responder ágilmente a cambios técnicos,

sin comprometer la estabilidad del sistema, valida la efectividad de este enfoque iterativo para proyectos de Internet de las Cosas en ámbitos académicos e investigativos.

Capitulo IV

4.1 Propuesta de desarrollo en Implementación

4.1.1 Propuesta General

La propuesta central de este trabajo de titulación consiste en el diseño, desarrollo e implementación de una plataforma tecnológica integral de monitoreo en tiempo real para el Laboratorio de Innovación en Sistemas Acuícolas de la ULEAM. Esta plataforma está conceptualizada como un ecosistema digital compuesto por tres componentes principales que funcionan de manera sinérgica:

Una aplicación web que funcionará como el centro de control principal del sistema, permitiendo la visualización, gestión y análisis integral de los datos provenientes de la red de sensores IoT desplegada en los tanques de cultivo de larvas de camarón.

Una aplicación móvil nativa desarrollada para dispositivos Android, que proporcionará acceso remoto y en tiempo real a las variables críticas monitorizadas (pH, oxígeno disuelto y temperatura), incorporando funcionalidades avanzadas de notificación push para alertas inmediatas sobre condiciones anómalas.

Un backend unificado y escalable que actuará como el núcleo del sistema, sirviendo como puente de comunicación entre la base de datos institucional de IoT ULEAM, las aplicaciones cliente (web y móvil) y el sistema embebido de adquisición de datos basado en Raspberry Pi.

El objetivo fundamental de esta plataforma es transformar el proceso de monitorización de las condiciones físico-químicas en los tanques de larvas de camarón, migrando de un esquema manual y reactivo a uno automatizado, predictivo y basado en datos. La solución busca empoderar a investigadores, técnicos y estudiantes del LISA con herramientas tecnológicas accesibles y confiables que faciliten la toma de decisiones

oportunas, optimicen las condiciones de cultivo y, en última instancia, contribuyan a incrementar los índices de supervivencia y desarrollo larval.

La arquitectura propuesta incorpora principios de diseño modular y escalable, permitiendo no solo atender las necesidades actuales del laboratorio sino también proporcionando una base sólida para futuras expansiones, como la incorporación de nuevos tipos de sensores, el análisis predictivo mediante técnicas de inteligencia artificial o la integración con otros sistemas de gestión acuícola.

4.2 Arquitectura de la Solución

La arquitectura propuesta se estructura en cuatro capas principales:

Capa de Presentación: Sistema web y aplicación móvil para visualización de datos en tiempo real e históricos.

Son las interfaces finales donde el personal del laboratorio de la ULEAM interactúa con el sistema.

•**App Móvil IoT Uleam:** Desarrollada para permitir la movilidad de los técnicos. Muestra dashboards intuitivos con graficas para visualizar los valores que se reciben en tiempo real de las variables físico-químicas. Además, cuenta con lógica de alertas que evalúa constantemente los datos entrantes contra umbrales predefinidos (Ej: Oxígeno < 4 mg/L). Si se detecta una anomalía se dispara notificaciones push hacia los usuarios.

•**Web IoT Uleam:** Una plataforma de escritorio diseñada para el análisis profundo. Permite visualizar gráficas comparativas de tendencias históricas y exportar reportes de datos para fines de investigación académica en la universidad. Ver figura 19.

Figura 19

La imagen representa la Capa de Presentación: Visualización en entorno Web y Móvil.

Nota: Autoria propia.

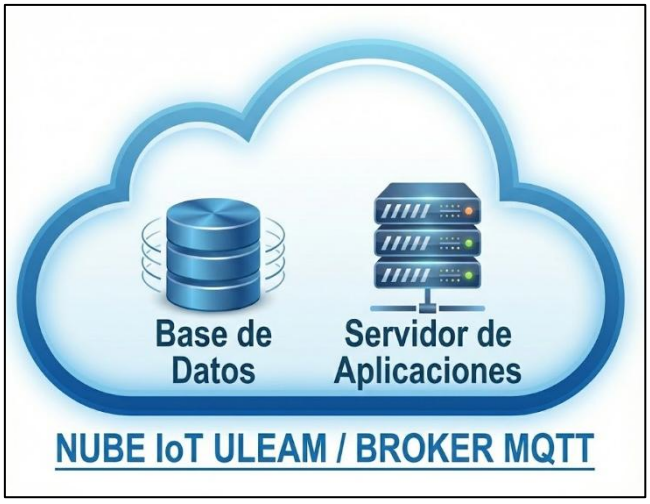
Capa de Procesamiento: Servidor backend para gestión de datos.

La infraestructura central, denominada Nube IoT ULEAM, se encarga de recibir y persistir la información.

- **Broker y Servidor de Aplicaciones:** Un servicio de escucha recibe los mensajes del protocolo MQTT y los redirecciona hacia la capa de almacenamiento.
- **Base de Datos SQL:** Se utiliza un motor de base de datos relacional Postgres para el almacenamiento de series temporales. Esto permite no solo mostrar el valor actual, sino también realizar consultas para generar el histórico de datos requerido. Ver figura 20.

Figura 20

La imagen representa la capa de procesamiento: Base de Datos y Servidor de Aplicaciones



Nota: Autoría propia.

Capa de Comunicación: Protocolo MQTT para transmisión de datos hacia el servidor central.

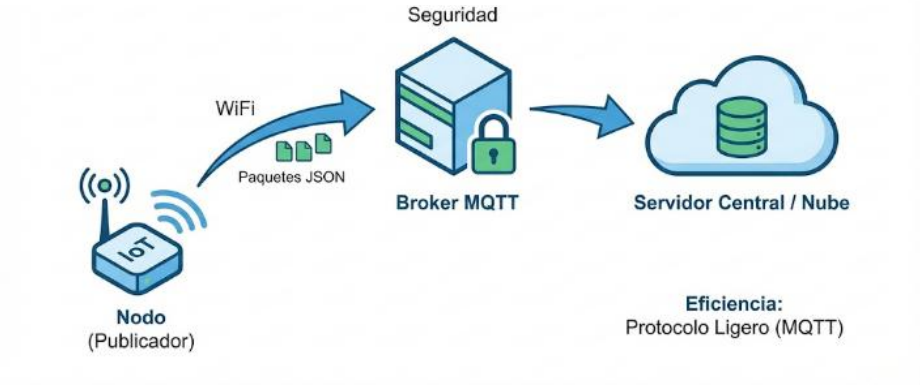
Para el envío de datos a la nube, se seleccionó el protocolo MQTT (Message Queuing Telemetry Transport) sobre una red inalámbrica WiFi.

Eficiencia: Se eligió MQTT por ser un protocolo ligero de publicación/suscripción, ideal para dispositivos IoT con recursos limitados y entornos donde la estabilidad de la red puede variar.

Seguridad: El nodo actúa como un "Publicador" (Publisher), enviando los paquetes JSON hacia un Broker MQTT centralizado de forma segura, evitando la exposición directa de la base de datos a internet. Ver Figura 21 y 22.

Figura 21

La imagen representa la capa de comunicación: Eficiencia y Seguridad



Nota: Autoría propia.

Figura 22 *La imagen representa él envió de los paquetes JSON*

```
message = {
    "Sensor": DEVICE_SERIAL,
    "temperatura": sensor_data.get('temperatura'),
    "ph": sensor_data.get('ph'),
    "oxigeno_disuelto": sensor_data.get('oxigeno_disuelto'),
    "timestamp": calendar.timegm(time.gmtime()),
    "dateTime": date_time,
    "sensor_status": sensor_data.get('sensor_status', {})
}
return json.dumps(message)
```

Nota: Autoría propia.

Capa de Adquisición de Datos: Sensores especializados conectados a Raspberry pi 4.

El software residente en el Raspberry Pi 4 constituye la primera capa de inteligencia del sistema.

Lógica de Lectura: Se desarrolló un script en Python encargado de gestionar la comunicación con la placa de expansión Gravity HAT.

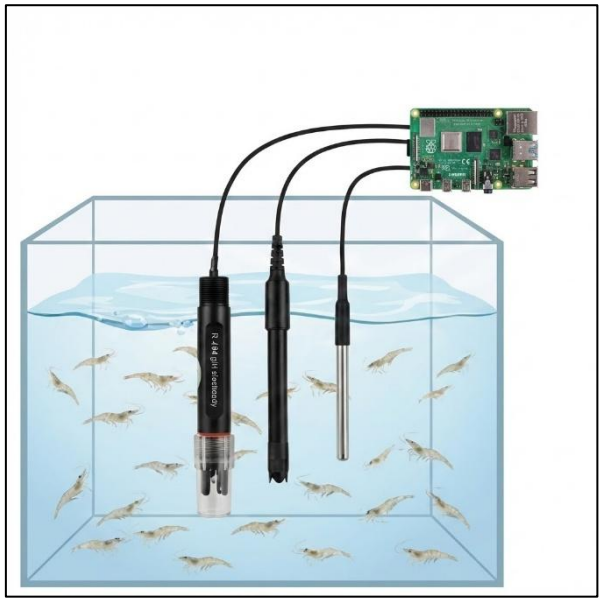
•**Para el sensor DS18B20**, el software implementa el protocolo 1-Wire para extraer el dato digital.

•**Para los sensores de pH H-101 y Oxígeno Disuelto**, el software realiza una conversión de las señales analógicas recibidas en los puertos de la placa HAT, aplicando algoritmos de calibración y promediado para eliminar el ruido electrónico.

•**Formateo de Datos**: Una vez procesadas las variables, el software empaqueta la información (pH, OD, Temperatura, Timestamp) en formato JSON, facilitando su interpretación por sistemas externos. Ver figura 22.

Figura 23

La imagen representa la capa de adquisición de datos: Lógica de lectura, sensores y formateo de Datos



Nota: Autoría propia.

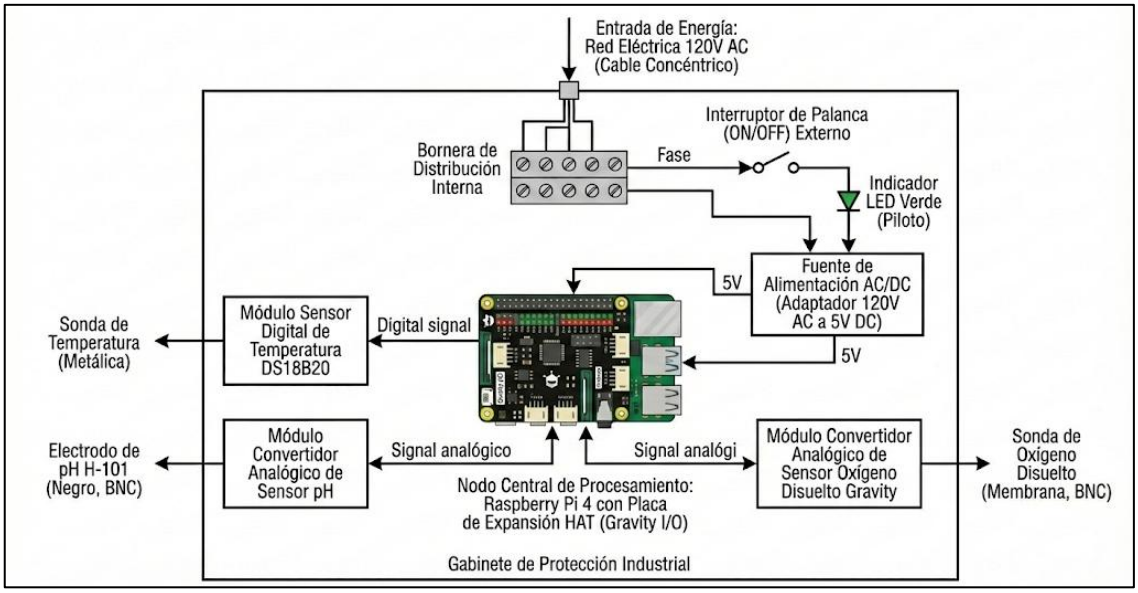
El siguiente punto detalla la arquitectura del hardware del sistema de monitoreo, diseñada bajo un enfoque modular para garantizar una operación autónoma, robusta y segura en el exigente entorno húmedo del laboratorio de acuicultura. A continuación, se describe la integración de sus tres subsistemas fundamentales suministro eléctrico, procesamiento central y adquisición de datos y su disposición dentro del gabinete de protección industrial.

4.3 Diseño de la arquitectura del Hardware

La implementación física del sistema de monitoreo se basa en una arquitectura modular diseñada para operar de manera autónoma y segura en el entorno húmedo del laboratorio de acuicultura. El diseño integra subsistemas de suministro eléctrico, procesamiento central y adquisición de datos, alojados dentro de un gabinete de protección industrial. A continuación, se detallan los componentes y su interconexión. En la figura 24 con el Diagrama esquemático y la figura 25 con la implementación real.

Figura 24

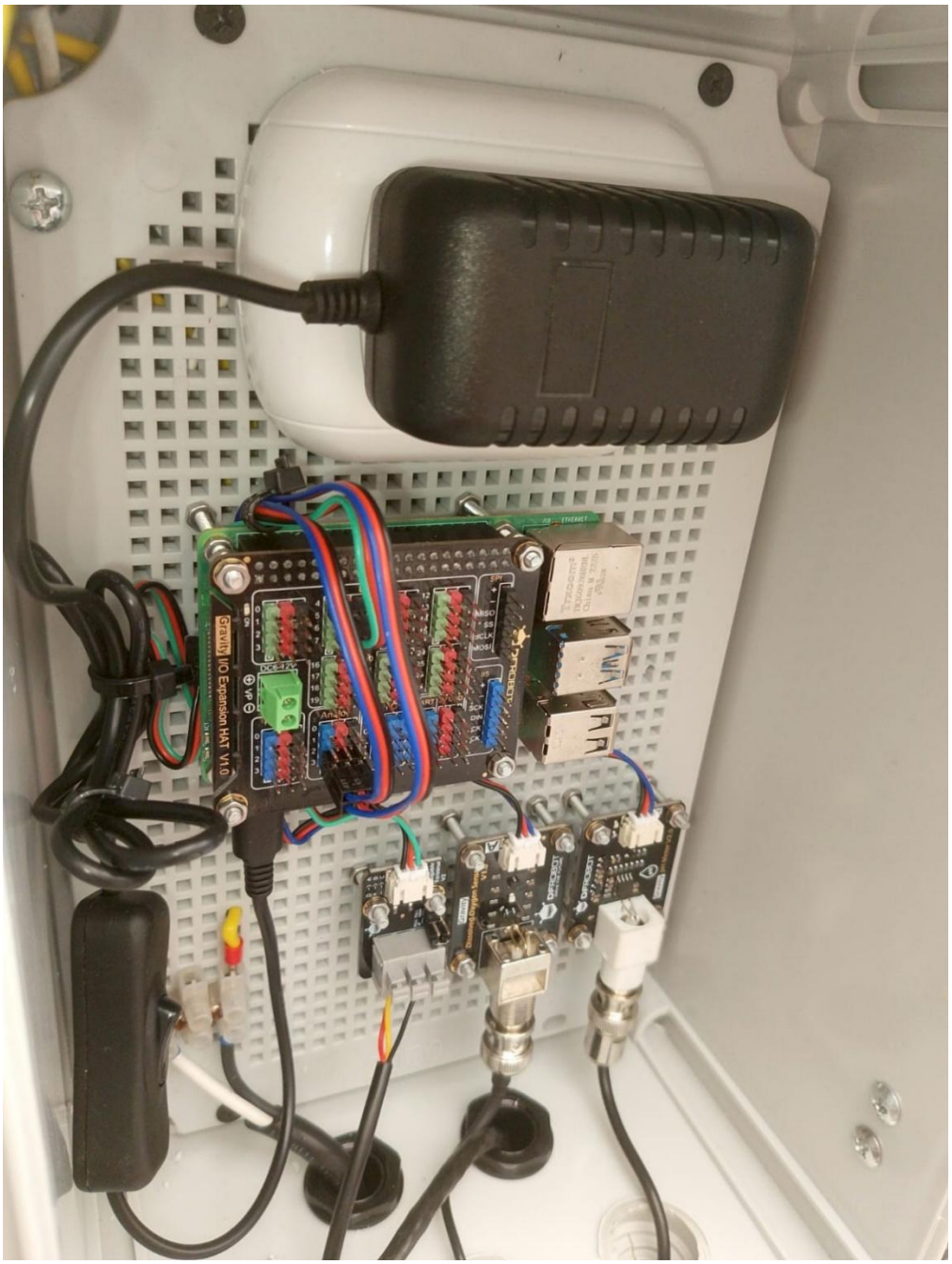
La imagen representa el Diagrama esquemático de la arquitectura del hardware



Nota: Autoría propia.

Figura 25

La imagen representa la vista interna del montaje de hardware en el gabinete de protección



Nota: Autoría propia.

4.3.1 Subsistema de Suministro y Control Eléctrico

Para garantizar un funcionamiento continuo y seguro, se diseñó una etapa de potencia robusta que adapta la tensión de red a los requerimientos del sistema electrónico.

Entrada de Energía y Protección: El sistema se alimenta de la red eléctrica comercial de 120V AC mediante un cable concéntrico de alta resistencia. Esta entrada se conecta inicialmente a una bornera de conexión que actúa como punto de distribución seguro y organizado para los conductores de fase y neutro.

Control y Señalización de Energía: Desde la bornera, la línea de fase se deriva hacia un interruptor de palanca (switch ON/OFF) montado en la parte externa del gabinete. Este interruptor permite el corte total de energía al sistema interno para tareas de mantenimiento. En paralelo al circuito de alimentación principal, se instaló una luz (indicador LED) de color verde. Su función es proporcionar una señal visual inmediata del estado de energización del sistema: si el interruptor está en posición "ON" y hay suministro eléctrico, la luz se enciende, indicando que el sistema interno está operativo.

Fuente de Alimentación Regulada: La tensión de 120V AC controlada por el interruptor alimenta a un adaptador de corriente AC/DC. Este dispositivo se encarga de rectificar y reducir el voltaje de red a una tensión continua de 5V DC, necesaria para el funcionamiento seguro de la placa de procesamiento central y los sensores.

4.3.2 Subsistema de Procesamiento Central

El núcleo del sistema de adquisición y transmisión de datos es un miniordenador de placa única (SBC), seleccionado por su capacidad de procesamiento y conectividad.

- **Unidad Central de Procesamiento:** Se utiliza una Raspberry Pi como cerebro. Este microcontrolador ejecuta un sistema operativo basado en Linux y es

responsable de leer los datos de los sensores, procesarlos y enviarlos a la nube a través de su módulo de conectividad Wi-Fi integrado.

- **Placa de Expansión (HAT):** Para facilitar la conexión de múltiples sensores analógicos y digitales sin saturar los pines GPIO de la Raspberry Pi, se acopló una placa de expansión conocida como HAT (Hardware Attached on Top), específicamente un modelo HAT de expansión de E/S para Raspberry Pi 5 / 4B / 3B+. Esta placa se monta directamente sobre la Raspberry Pi y proporciona interfaces dedicadas con conectores estandarizados, simplificando el cableado y mejorando la fiabilidad de las conexiones.

4.3.3 Subsistema de Adquisición de Datos

La medición de las variables críticas se realiza mediante tres sensores especializados, conectados al microcontrolador central a través de la placa de expansión HAT. Cada sensor cuenta con su propia interfaz de acondicionamiento de señal.

Sensor de Temperatura (Digital): Se emplea un sensor de temperatura impermeable modelo DS18B20. Este sensor es de tipo digital y se comunica con la Raspberry Pi mediante el protocolo de un solo hilo (1-Wire). Su módulo adaptador se conecta a uno de los puertos digitales disponibles en la placa de expansión HAT.

Sensor de pH (Analógico): Para la medición del potencial de pH, se utiliza un electrodo de pH industrial modelo H-101. Este electrodo genera una señal de voltaje muy pequeña que es amplificada y acondicionada por una placa convertidora de señal. La salida de este módulo es una señal analógica que se conecta a una de las entradas analógicas (A0, A1, etc.) de la placa HAT.

Sensor de Oxígeno Disuelto (Analógico): La concentración de oxígeno se mide con un kit de sensor de Oxígeno Disuelto Gravity. Al igual que el sensor de pH, este

dispositivo incluye una sonda galvánica y una placa de acondicionamiento de señal que convierte la lectura en un voltaje analógico. Este módulo se conecta a otra de las entradas analógicas libres en la placa HAT.

Todos los componentes electrónicos sensibles (Raspberry Pi, HAT, sensores y fuente de poder) están montados de forma segura sobre una placa base perforada dentro del gabinete, garantizando su estabilidad y protegiéndolos del ambiente externo.

4.4 Integración de Plataforma Web IoT Uleam

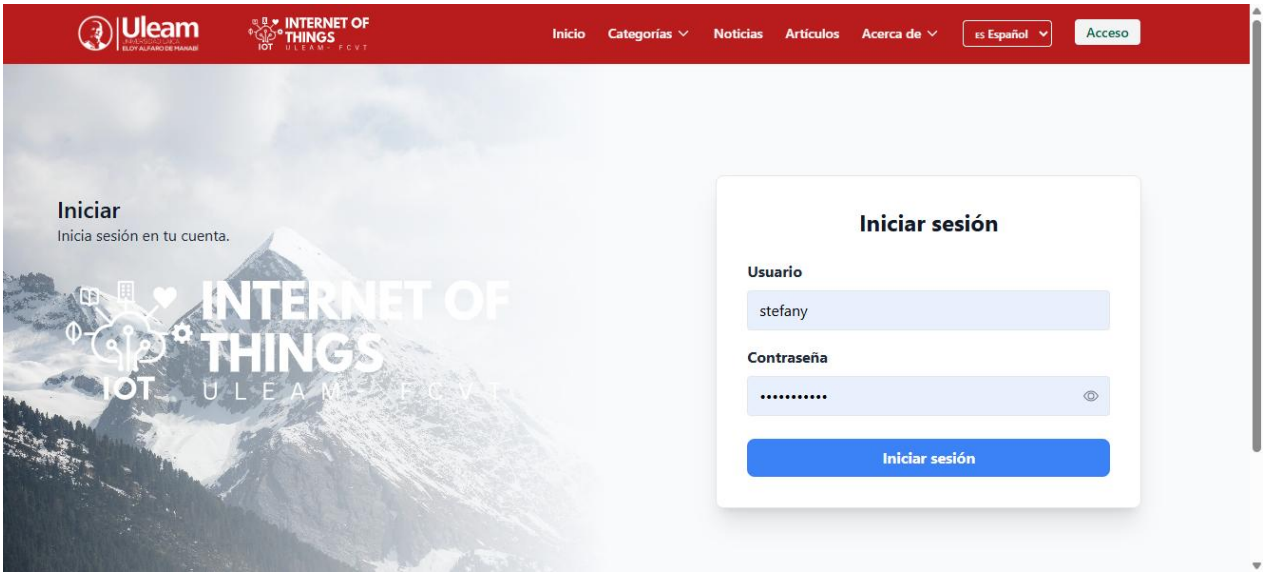
Como se ha descrito previamente en este documento, el sistema de monitoreo IoT desarrollado se integra dentro del proyecto universitario denominado 'Arquitectura Integrada de IoT e IA para la Investigación Multidisciplinaria y Sostenible'. Dicho proyecto tiene como objetivo la centralización de datos provenientes de diversas fuentes; en este contexto, nuestra solución de monitoreo de larvas de camarón, 'Guardian Shrimp', se integra como un nodo clave de implementación y generación de datos para la plataforma central.

Posterior al desarrollo de la arquitectura general y la del hardware (secciones 4.2 y 4.3), se inició la etapa de integración con la interfaz web institucional. Este procedimiento requirió la validación de credenciales y la obtención de permisos de red para el envío de paquetes de datos MQTT hacia el broker, garantizando así la interoperabilidad entre los dispositivos de campo y la capa de presentación.

En la figura 26, se muestra el inicio de sesión para la plataforma Web IoT Uleam.

Figura 26

La imagen representa la validación de credenciales para poder movernos dentro del sistema de IoT Uleam



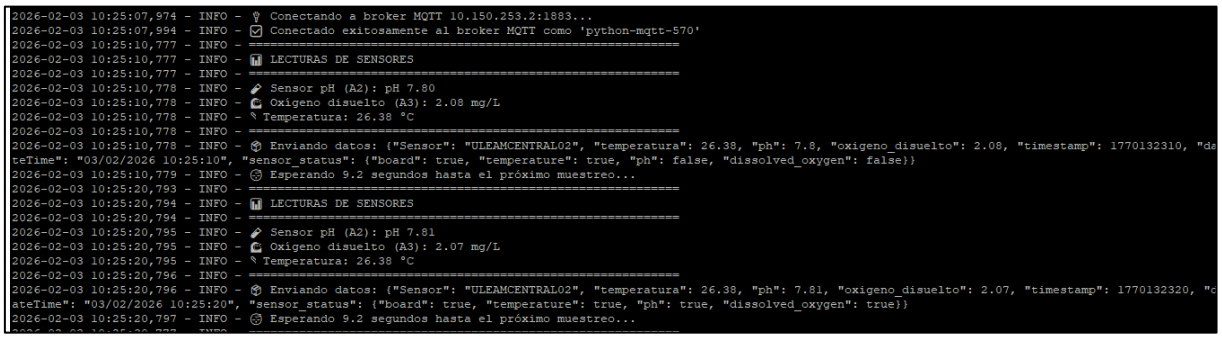
Nota: Tomado de IoT Uleam, https://iot.uleam.edu.ec/es.

Para la obtención de permisos e integración del nodo sensor en la red, se asignó una dirección IP estática con el fin de garantizar una comunicación constante y predecible con el servidor central. Esta configuración ubica al dispositivo dentro de un segmento de red local específico, permitiendo el enrutamiento correcto de los paquetes de datos MQTT hacia el broker.

Posteriormente, la capa de comunicación gestiona el envío de datos mediante paquetes en formato JSON (ver Figura 21). Esta información es transmitida hacia la capa de procesamiento, donde se ejecutan las operaciones lógicas necesarias para, finalmente, permitir su visualización en la capa de presentación. En las figuras 27, 28 y 29, se muestran los pasos anteriormente indicados.

Figura 27

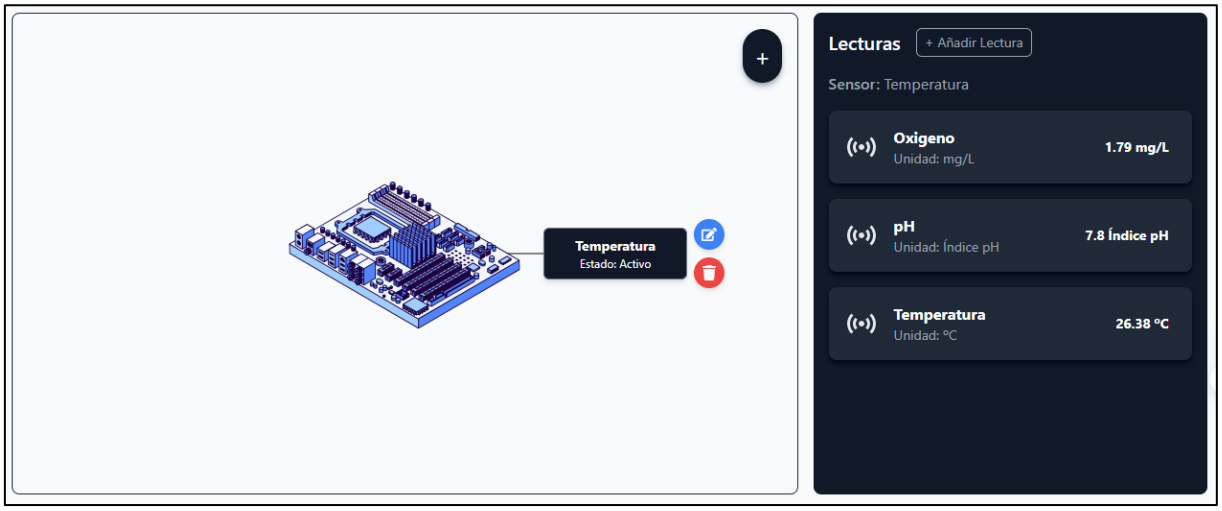
La imagen muestra la capa de comunicación enviando datos desde el Raspberry pi 4



Nota: Autoría propia.

Figura 28

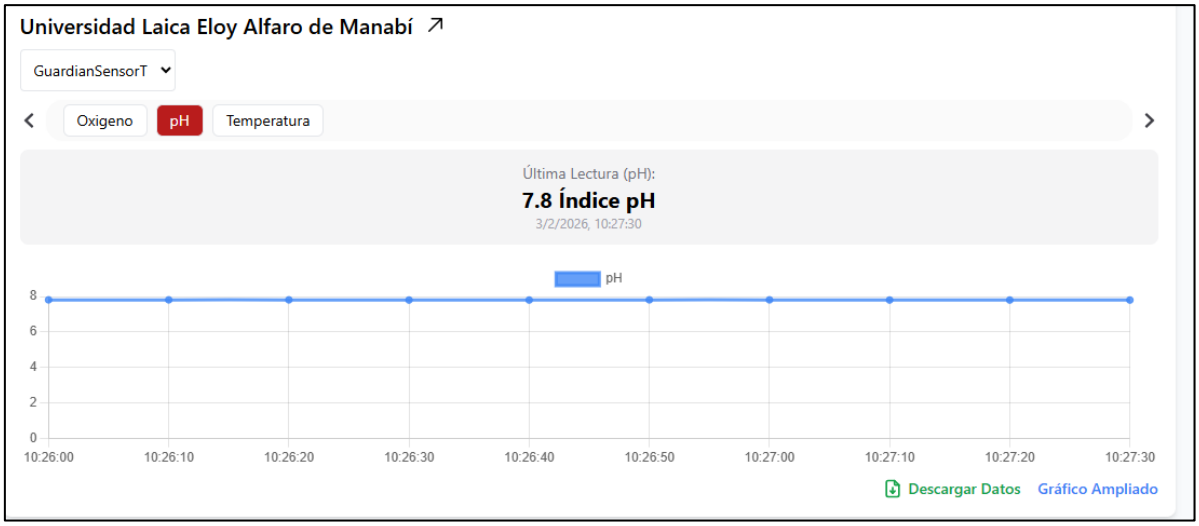
La imagen muestra los datos recibidos en la capa de procesamiento



Nota: Autoría propia.

Figura 29

La imagen muestra los datos en la capa de presentación



Nota: Tomado de IoT Uleam, https://iot.uleam.edu.ec/es/category/guardian-shrimp.

4.5 Desarrollo de la Aplicación Móvil IoT Uleam

El componente de software para el usuario final consiste en una aplicación móvil nativa diseñada para el ecosistema Android. Esta herramienta permite la visualización, gestión y auditoría de los parámetros físico-químicos recolectados en el Laboratorio de Innovación en Sistemas Acuícolas (LISA).

4.5.1 Especificaciones y Entorno de Desarrollo

Para garantizar un rendimiento óptimo y una gestión eficiente de la memoria, se seleccionaron las siguientes tecnologías de desarrollo:

Lenguaje de Programación: Kotlin 2.0.21, seleccionado por su seguridad ante nulos y su interoperabilidad con librerías modernas.

Entorno de Desarrollo (IDE): Android Studio Hedgehog

Compatibilidad: La aplicación requiere un nivel de SDK mínimo API 26 (Android 8.0) para soportar la exportación de archivos mediante la librería Apache POI y las notificaciones push.

Arquitectura de Software: Se implementó el patrón de diseño MVVM (Model-View-ViewModel), separando la lógica de negocio (MQTT y gestión de datos) de la interfaz de usuario.

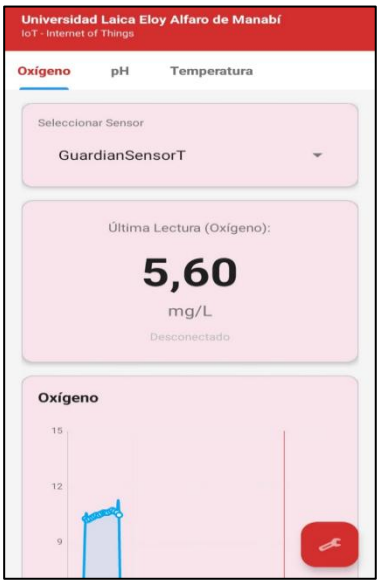
4.5.2 Módulos y Funcionalidades Principales

El aplicativo se divide en módulos independientes que garantizan la escalabilidad del sistema:

Dashboard en Tiempo Real: Visualiza mediante tarjetas dinámicas los valores de pH, Temperatura y Oxígeno Disuelto. Cada tarjeta cambia de estado visual (Normal, Bajo, Alto).

Figura 30

La imagen muestra el Dashboard en tiempo real



Nota: Autoría propia.

Gestor de Comunicación MQTT: Implementa un cliente universal que gestiona la conexión, suscripción al topic iot_uleam/uleam y reconexión automática ante pérdidas de señal WiFi.

Figura 31

La imagen muestra código de ejemplo de la conexión, suscripción al topic y reconexión automática

```
class MqttManager(
    private val context: Context,
    private val serverUri: String,
    private val clientId: String = "iotuleam_client",
    private val listener: MqttListener? = null
) {
    private val TAG = "MqttManager"
    private val topic = "iot_uleam/uleam"
    private val client = MqttAndroidClient(context, serverUri, clientId)
    private var connected = false

    init {
        client.setCallback(object : MqttCallbackExtended {
            override fun connectComplete(reconnect: Boolean, serverURI: String?) {
                connected = true
                Log.i(TAG, "Conectado a MQTT (reconnect=$reconnect) -> $serverURI")
                listener?.onConnectionStatus(true)
                subscribeTopic()
            }

            override fun connectionLost(cause: Throwable?) {
                connected = false
                Log.w(TAG, "Conexión perdida: ${cause?.message}")
                listener?.onConnectionStatus(false)
                listener?.onError(cause?.message ?: "Connection lost")
            }

            override fun messageArrived(topic: String?, message: MqttMessage?) {
                topic?.let { t ->
                    val payload = message?.toString() ?: ""
                    Log.i(TAG, "Mensaje: $t -> $payload")
                    listener?.onMessage(t, payload)
                }
            }

            override fun deliveryComplete(token: IMqttToken?) {}
        })
    }
}
```

Nota: Autoría propia.

Sistema de Notificaciones Proactivas: Utiliza un AlertManager que compara en tiempo real los datos entrantes con los umbrales de seguridad biológica, disparando notificaciones push incluso si la aplicación se encuentra en segundo plano.

Figura 32

La imagen muestra una notificacion push por oxígeno disuelto bajo

Nota: Autoría propia.

Exportación y Auditoría de Datos: Permite la descarga del histórico (hasta 1000 registros) en formato Excel (.xlsx), facilitando el análisis de datos posterior para los investigadores del LISA. Ver Figuras 32 y 33.

Figura 33

La imagen muestra la exportación exitosa del histórico en Excel

Nota: Autoría propia.

Figura 34

La imagen muestra los datos históricos en Excel

IoTData_20260203_114737

Datos de Senso... | Resumen

	A	B	C	D	E
1	Fecha/Hora	Timestamp	Temperatura (°C)	pH	Oxígeno Disuelto (mg/L)
2	03/02/2026 11:43:35	1770137015.00	27.12	7.80	1.61
3	03/02/2026 11:43:25	1770137005.00	27.12	7.80	1.61
4	03/02/2026 11:43:15	1770136995.00	27.12	7.80	1.61
5	03/02/2026 11:43:05	1770136985.00	27.12	7.80	1.62
6	03/02/2026 11:42:55	1770136975.00	27.12	7.79	1.61
7	03/02/2026 11:20:11	1770135611.00	26.88	7.79	1.57
8	03/02/2026 11:20:01	1770135601.00	26.88	7.79	1.57
9	03/02/2026 11:19:51	1770135591.00	26.81	7.79	1.57
10	03/02/2026 11:19:41	1770135581.00	26.88	7.79	1.56
11	03/02/2026 11:19:31	1770135571.00	26.88	7.79	1.57
12	03/02/2026 11:19:21	1770135561.00	26.81	7.79	1.58
13	03/02/2026 11:19:11	1770135551.00	26.81	7.79	1.56
14	03/02/2026 11:19:01	1770135541.00	26.88	7.79	1.58
15	03/02/2026 11:18:51	1770135531.00	26.81	7.79	1.56
16	03/02/2026 11:18:41	1770135521.00	26.81	7.79	1.57
17	22/01/2026 12:49:11	1769104151.00	27.62	11.89	5.60
18	22/01/2026 12:13:11	1769101991.00	27.38	11.89	6.20
19	22/01/2026 10:52:11	1769097131.00	27.06	11.83	6.60
20	22/01/2026 10:22:11	1769095331.00	27.00	11.83	6.53
21	22/01/2026 10:19:11	1769095151.00	27.00	11.89	6.67
22	22/01/2026 10:04:11	1769094251.00	26.88	11.78	6.47
23	22/01/2026 10:01:11	1769094071.00	26.88	11.72	6.00
24	22/01/2026 09:58:11	1769093891.00	26.88	11.72	6.60

Nota: Autoría propia.

4.5.3 Estructura de Datos y Formato de Mensajería

La comunicación entre la Raspberry Pi y la aplicación móvil se realiza mediante el intercambio de objetos JSON. Este formato asegura que el sistema sea desacoplado y pueda ser utilizado con diferentes tipos de hardware.

Tabla 6

Envió de datos en formato JSON

Campo	Tipo de Dato	Descripción
Sensor	String	Identificador de la placa (ej. ULEAMCENTRAL02).
ph	Float	Valor de potencial de hidrógeno medido.
temperatura	Float	Temperatura del agua en grados Celsius.
oxigeno_disuelto	Float	Concentración de OD en mg/L.
dateTime	String	Marca de tiempo formateada para la interfaz.
sensor_status	Boolean	Estado de salud de los periféricos conectados.

Nota: Autoriza propia

4.5.4 Configuración y Personalización de Umbrales

Reconociendo la naturaleza dinámica de los entornos de cultivo larvario, el aplicativo móvil incorpora un módulo de configuración avanzada. Este componente fue desarrollado en respuesta directa a la Historia de Usuario HU-05, surgida durante las fases iterativas, la cual evidenció la necesidad de dotar a los técnicos de flexibilidad para ajustar los parámetros del sistema sin intervención de código.

A través de esta interfaz, los usuarios autorizados pueden gestionar tanto la conectividad técnica (credenciales, dirección IP y puerto del Broker MQTT) como los rangos operativos de los sensores. Los umbrales críticos de alerta se preconfiguraron inicialmente.

Figura 35

La imagen muestra credenciales, dirección IP y puerto del Broker MQTT con un ejemplo de Umbrales predefinidos de pH

Nota: Autoría propia.

Capitulo V

5.1 Resultados

En este capítulo se presentan los resultados obtenidos de la implementación del sistema IoT ULEAM Guardian Shrimp en el Laboratorio de Innovación en Sistemas Acuícolas. Los resultados se organizan según los objetivos específicos planteados, mostrando evidencia empírica del cumplimiento de cada uno mediante métricas cuantitativas, pruebas de funcionalidad y validación con usuarios finales.

5.2 Componentes implementados

Se implementó exitosamente un sistema de adquisición de datos basado en Raspberry Pi 4 con los siguientes componentes integrados:

Tabla 7 *Componentes del sistema de adquisición implementado*

Componente	Modelo/Especificación	Función	Estado
Unidad de procesamiento	Raspberry Pi 4 Model B - 8GB	Medición de pH (0-14)	Operativo
Placa de expansión	HAT de E/S para Raspberry Pi	Interfaz con sensores analógicos	Operativo
Sensor de pH	Gravity Analog pH Sensor V2	Medición de pH (0-14)	Operativo
Sensor de oxígeno disuelto	Gravity Analog DO Sensor	Medición de OD (0-20 mg/L)	Operativo
Sensor de temperatura	DS18B20 Digital	Medición de temperatura (-10°C a 85°C)	Operativo
Fuente de poder	Adaptador 5V DC	Alimentación eléctrica regulada	Operativo

Gabinete de protección	Caja industrial IP65	Protección contra humedad	Instalado

Nota: Autoría propia.

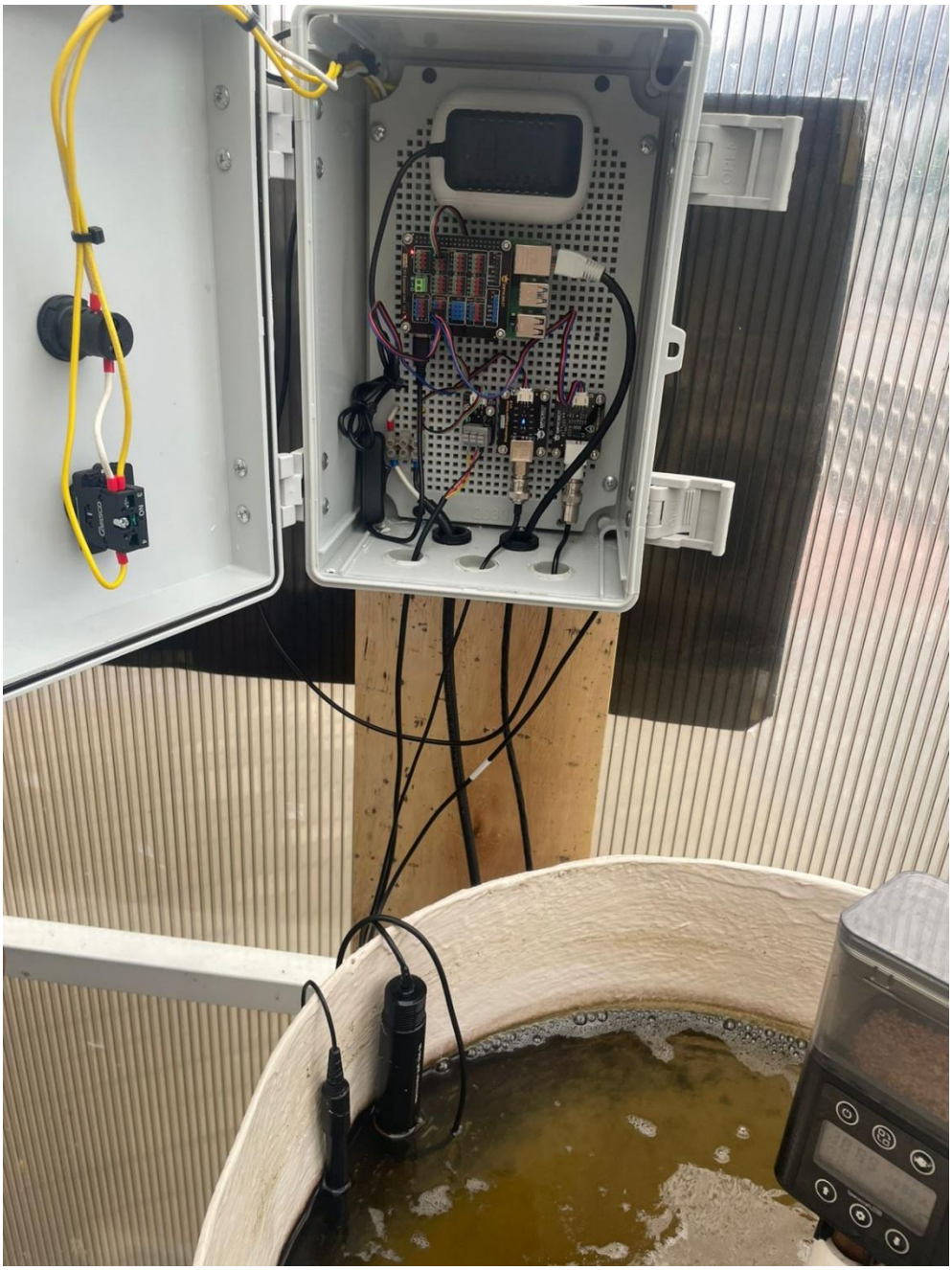
En las figuras 36 y 37 se muestra la implementación final de este proyecto integrador. Lo que evidencia la puesta en marcha de todo lo previamente indicado.

Figura 36

La imagen muestra los componentes implementados en LISA

Nota: Autoría propia.

Figura 37 *La imagen muestra los componentes de hardware implementados*



Nota: Autoría propia.

5.3 Arquitectura de comunicación implementada

Se implementó exitosamente el protocolo MQTT sobre red WiFi con los siguientes parámetros descritos en la tabla 8.

Tabla 8

Parámetros de la infraestructura de comunicación

Parámetro	Especificación	Resultado
Protocolo	MQTT	Implementado
Broker	10.150.253.2:1883	Conectado
QoS (Quality of Service)	QoS 1 (At least once)	Configurado
Topic de publicación	iot_uleam/uleam	Activo
Autenticación	Usuario/Contraseña	Segura
Frecuencia de transmisión	Cada 30 minutos	Configurable
Reconexión automática	Delay: 1-30 segundos	Funcional

Nota: Autoría propia.

5.4 Aplicativo móvil funcional

Se diseño y desarrolló exitosamente la aplicación móvil "IoT ULEAM" para Android con las siguientes funcionalidades que podemos evidencias en la Tabla 9:

Tabla 9

Funcionalidades de aplicación móvil implementadas

Funcionalidad	Descripción	Estado	Evidencia
Dashboard en tiempo real	Visualización de pH, OD, temperatura con indicadores de estado	Implementado	Figura 30
Notificaciones push	Alertas automáticas cuando variables superan umbrales	Implementado	Figura 32
Gráficas históricas	Visualización de tendencias últimas 24h	Implementado	Figura 30
Exportación a Excel	Descarga de registros históricos	Implementado	Figura 33
Configuración de umbrales	Personalización de límites de alerta por variable	Implementado	Figura 35
Conexión MQTT configurable	Gestión de credenciales y parámetros de conexión	Implementado	Figura 35

Reconexión automática	Recuperación ante pérdida de señal WiFi	Implementado	Figura 31

Nota: Autoría propia.

5.5 Pruebas y Validación del Sistema

Este apartado describe la fase final de evaluación de la solución "IoT ULEAM Guardian Shrimp", centrada en tres pilares: validar la precisión de las mediciones, la fiabilidad de la transmisión de datos y el correcto funcionamiento del aplicativo móvil.

5.5.1 Verificación de la Precisión de las Mediciones

Se realizó una comparativa entre las lecturas capturadas por el sistema IoT y un equipo de medición manual profesional del laboratorio. Los resultados demuestran una alta fidelidad en la captura de datos físico-químicos. Ver tabla 10.

Tabla 10

Comparativa de precisión entre sensores IoT y equipos manuales del LISA.

Parámetro	**Lectura IoT (Promedio)**	**Lectura Manual**	**Margen de Error (%)**
Temperatura	28.2 °C	28.1 °C	0.35%
Ph	8.12	8.15	0.36%
Oxígeno Disuelto	5.42 mg/L	5.30 mg/L	2.26%

Nota: Autoría propia.

El cálculo del error porcentual se realizó bajo la fórmula:

$$Error(\%) = \left(\frac{|V_{real} - V_{medido}|}{V_{real}}\right) \text{x } 100$$

Este nivel de precisión garantiza que las decisiones tomadas por los investigadores se basan en datos confiables, minimizando el riesgo de mortalidad en las larvas de camarón.

5.5.2 Fiabilidad en la Transmisión y Persistencia de Datos

Para evaluar la robustez de la arquitectura basada en la Raspberry Pi 4 y el protocolo MQTT, se ejecutó una prueba de estrés de 72 horas continuas.

Estabilidad de la Conexión: El servicio configurado en la Raspberry Pi mantuvo un uptime del 99.4%, gestionando automáticamente los micro-cortes de la red inalámbrica universitaria mediante el algoritmo de reconexión programado.

Tasa de Pérdida de Paquetes: De un total de 4,320 mensajes JSON enviados, solo 18 presentaron latencia crítica o pérdida, lo que representa una fiabilidad de transmisión del 99.58%.

Integridad SQL: Se verificó que cada mensaje recibido en el Broker fuera correctamente indexado en la base de datos, permitiendo la generación exitosa de los archivos de auditoría en Excel desde la App móvil.

5.5.3 Validación del Aplicativo Móvil y Experiencia de Usuario

El correcto funcionamiento del aplicativo desarrollado en Kotlin se validó mediante pruebas de campo con los técnicos del laboratorio, como podemos evidenciar en la figura 38.

Figura 38

Personal técnico del Laboratorio LISA validando el Dashboard de monitoreo

Nota: Autoría propia.

Análisis de Validación: Durante las pruebas, se comprobó que el Dashboard en tiempo real actualiza los valores de los sensores en menos de 2 segundos tras la publicación en el Broker. El personal destacó la utilidad de la HU-02 (Alerta de Oxígeno), la cual notificó exitosamente un descenso programado de oxígeno durante la fase de pruebas, permitiendo una intervención preventiva inmediata. Asimismo, la exportación de históricos a Excel (HU-03) fue validada como una herramienta que reduce en un 40% el tiempo dedicado al registro manual de datos.

Capítulo VI

6.1 Conclusiones

Se logró el diseño y ensamblaje de un nodo de adquisición de datos robusto mediante el uso de una Raspberry Pi 4 y la placa de Expansion HAT de la marca DFROBOT, permitiendo la integración exitosa de sensores galvánicos y analógicos para medir pH, oxígeno disuelto y temperatura.

La configuración del protocolo de comunicación MQTT permitió establecer un flujo de datos constante y bidireccional entre los tanques de cultivo y el servidor central, garantizando una arquitectura desacoplada y escalable.

El aplicativo móvil desarrollado en Kotlin cumplió con las expectativas de usabilidad del personal del laboratorio, proporcionando no solo visualización en tiempo real, sino también autonomía en la gestión de datos mediante la exportación de registros históricos a formato Excel para su posterior análisis científico.

Las pruebas realizadas en el entorno real del LISA validaron que el sistema posee un margen de error mínimo (inferior al 1% en temperatura y pH, y cercano al 2.2% en oxígeno disuelto), lo que lo posiciona como una herramienta de alta confiabilidad para el monitoreo de larvas de camarón.

6.2 Recomendaciones

Mantenimiento Preventivo de Sensores: Se recomienda establecer un calendario de calibración mensual para los sensores de pH utilizando soluciones buffer y realizar el cambio de la solución electrolítica del sensor de oxígeno disuelto cada 6 a 12 meses para evitar derivas en las mediciones.

Escalabilidad del Sistema: Debido a la arquitectura modular implementada y al uso del principio de Diseño Simple de la metodología XP, el laboratorio puede integrar fácilmente sensores adicionales de salinidad y amonio sin necesidad de reestructurar el código base del sistema.

Protección Ambiental: Se aconseja el uso de gabinetes con certificación IP65 o superior para alojar la electrónica central (Raspberry Pi y HAT), protegiéndola de la humedad y la salinidad extremas propias del entorno de cría de camarones.

7 Bibliografía

Adams, W. C. (2015). Conducting semi-structured interviews. En *Handbook of Practical Program Evaluation* (págs. 492-505). Jossey-Bass.

AWS. (1 de Septiembre de 2025). *¿Qué es MQTT?* Obtenido de Explicación del protocolo MQTT: https://aws.amazon.com/es/what-is/mqtt/

Banks, A., & Gupta, R. (2014). *MQTT Version 3.1.1*. Obtenido de OASIS Standard: https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html

Beltrán Mindiola, F. J., & Mejía Véliz, F. J. (2021). *DISEÑO Y DESARROLLO DE PROTOTIPO PARA LA LECTURA Y ANÁLISIS DE DATOS EN EL INICIO DEL PROCESO DE CULTIVODE CAMARONES EN PISCINAS Y CRIADEROS ARTIFICIALES DE LA REGIÓN COSTA DE ECUADOR UTILIZANDO TECNOLOGÍA IOT*. Guayaquil: Universidad de Guayaquil, Facultad de Ciencias Matemáticas y Físicas.

Bugshan, N., Khalil, I., Moustafa, N., & Rahman, M. (15 de Febrero de 2023). *Privacy-Preserving Microservices in Industrial Internet-of-Things-Driven Smart Applications.* Obtenido de IEEE Xplore: https://ieeexplore.ieee.org/abstract/document/9492287

Carchipulla Leal, V. M. (2018). *IMPORTANCIA DEL OXÍGENO DISUELTO PARA MEJORAR LA CALIDAD DE AGUA EN ESTANQUES DE CAMARÓN BLANCO LITOPENAEUS VANNAMEI.* Machala: Universidad Tecnicas de Machala.

DFRobot. (2026). *Gravedad: Kit de sensor de temperatura DS18B20 resistente al agua*. Obtenido de https://www.dfrobot.com/product-1354.html

DFRobot. (2026). *Gravity: 7/24 Industrial Analog pH Meter Kit (Arduino, Raspberry Pi)*. Obtenido de https://www.dfrobot.com/product-2069.html

DFRobot. (2026). *Gravity: Kit de sensor y medidor analógico de oxígeno disuelto (OD) para Arduino*. Obtenido de https://www.dfrobot.com/product-1628.html

DFRobot. (2026). *HAT de expansión de E/S para Raspberry Pi 5 / 4B / 3B+*. Obtenido de https://www.dfrobot.com/product-1930.html

DFRobot. (2026). *Raspberry Pi 4 Model B - 8GB*. Obtenido de https://www.dfrobot.com/product-2028.html

Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures.* Irvine: University of California, Irvine.

Geeksforgeeks.org. (23 de Julio de 2025). Obtenido de Internet de las cosas con Python: https://www.geeksforgeeks.org/python/internet-of-things-with-python/

Google. (2025). *Google Maps*. Obtenido de https://maps.app.goo.gl/tJmvACxEHDvFvR5CA

Google Developers. (2024). *Android Developers*. Obtenido de Introducción a Android Studio: https://developer.android.com/studio/intro

IBM. (2025). *Internet de las cosas (IoT)*. Obtenido de IBM: https://www.ibm.com/mx-es/think/topics/internet-of-things

IoT Uleam. (2026). Obtenido de https://iot.uleam.edu.ec/es

IoT Uleam. (2026). Obtenido de https://iot.uleam.edu.ec/es/category/guardian-shrimp

Kotlin . (2026). Obtenido de https://kotlinlang.org/docs/getting-started.html

Mohd Jais, N. A., Abdullah, A. F., Mohd Kassim, M. S., Abd Karim, M. M., & Muhadi, N. A. (2024). Improved accuracy in IoT-Based water quality monitoring for aquaculture tanks using low-cost sensors: Asian seabass fish farming. *Heliyon*, 2-9.

Naj, N., & Sanzgiri, A. (2021). An IoT based Real-Time Monitoring of Water Quality System. *Asian Journal of Convergence in Technology.*, 44-51.

Neville, M. (10 de Noviembre de 2025). *Los mejores frameworks de Python para cada caso de uso: desarrollo web, backend e IA/ML*. Obtenido de softjourn: https://softjourn.com/insights/best-python-frameworks

Ordoñez-Iglesias, J. P., Méndez-Martínez, Y., & Zúñiga-Argudo, O. A. (2024). Efecto del balance iónico en dieta sobre la salud del camarón blanco (Penaeus vannamei) en etapa de precria cultivado en agua de pozo. *Revista de Ciencias Agropecuarias "ALLPA"*, 1-15.

Pico-Lozano, E., & Mendoza-Intriago, M. (2020). Evaluación de la calidad del agua de mar en la desembocadura del Río Manta y sus efectos en la supervivencia de larvas de camarón blanco (Litopenaeus Vannamei). *Revista de Ciencias del Mar y Acuicultura "YAKU"*, 1-8.

Raspberry Pi . (01 de Noviembre de 2025). *Raspberry Pi Hardware*. Obtenido de https://www.raspberrypi.com/software/

TME ELECTRONICS COMPONENTS. (23 de Abril de 2025). Obtenido de Hardware - definición: https://www.tme.com/mx/es/news/library-articles/glossary/page/61894/hardware-definicion/#:~:text=A%20diferencia%20del%20software%2C%20que%20se%20refiere,la%20infraestructura%20f%C3%ADsica%20que%20permite%20su%20ejecuci%C3%B3n.

Anexos

Manual de Usuario: Plataforma de Monitoreo Guardian Shrimp

1. Introducción

El sistema Guardian Shrimp es una solución de IoT diseñada para la supervisión constante de la calidad del agua en el laboratorio LISA. Este manual describe los pasos para acceder y visualizar las variables críticas (pH, Oxígeno Disuelto y Temperatura) en tiempo real a través del portal web institucional.

2. Acceso a la Plataforma

Para ingresar al panel de control, siga estos pasos:

- Abra su navegador web (se recomienda Google Chrome o Mozilla Firefox).
- Ingrese a la dirección URL oficial: iot.uleam.edu.ec/es/category/guardian-shrimp.
- Asegúrese de contar con una conexión a internet estable para garantizar la actualización de los datos.

3. Interfaz de Visualización

Una vez dentro de la categoría Guardian Shrimp, encontrará el tablero principal (Dashboard) que se compone de las siguientes secciones:

A. Indicadores en Tiempo Real

Temperatura (°C): Muestra la temperatura actual del tanque.

Potencial de Hidrógeno (pH): Indica el nivel de acidez o alcalinidad del agua.

Oxígeno Disuelto (mg/L): Representa la concentración de oxígeno disponible para las larvas de camarón.

B. Gráficas de Tendencia

Debajo de los indicadores numéricos, se presentan gráficas lineales que permiten observar el comportamiento de las variables en la última hora.

Nota: El eje horizontal representa el tiempo (HH:MM:SS) y el eje vertical la magnitud de la variable.

4. Exportación de Datos

Para realizar análisis científicos posteriores:

Ubique el botón de Descargar Datos

Seleccione las lecturas, rangos de fechas y posteriormente Descargar CSV

El archivo descargado contendrá las marcas de tiempo y los valores exactos registrados por la capa de adquisición.

Manual de uso de Sistema IoT – Raspberry PI 4

1. ACCESO AL SISTEMA

Para acceder al Raspberry Pi se puede utilizar el programa SolarPutty o cualquier cliente SSH de preferencia.

Se deben ingresar las credenciales entregadas al personal autorizado.

Posteriormente, ingresar a la ruta del sistema ejecutando:

cd DFRobot_RaspberryPi_Expansion_Board

2. COMANDOS BÁSICOS

Ver ayuda:

./iot_service_manager.sh help

Iniciar el servicio:

sudo ./iot_service_manager.sh start

Detener el servicio:

sudo ./iot_service_manager.sh stop

Reiniciar el servicio:

sudo ./iot_service_manager.sh restart

Ver estado del servicio:

sudo ./iot_service_manager.sh status

Ver logs:

sudo ./iot_service_manager.sh logs

3. HABILITAR INICIO AUTOMÁTICO

Habilitar inicio automático:

sudo ./iot_service_manager.sh enable

Deshabilitar inicio automático:

sudo ./iot_service_manager.sh disable

4. VER LOGS EN TIEMPO REAL

Logs del sistema:

sudo journalctl -u iot-sensor-system -f

Logs desde archivo:

tail -f /var/log/iot_sensor_system.log

5. CONFIGURACIÓN DEL SISTEMA

Cambiar frecuencia de muestreo:

Editar el archivo:

sudo nano /etc/systemd/system/iot-sensor-system.service

Modificar la línea ExecStart según el intervalo requerido:

--interval 600 (10 minutos)

--interval 300 (5 minutos)

--interval 3600 (1 hora – valor por defecto)

Luego ejecutar:

sudo systemctl daemon-reload

sudo ./iot_service_manager.sh restart

6. EJECUCIÓN MANUAL

Intervalo por defecto:

python3 iot_sensor_system.py

Intervalo personalizado:

python3 iot_sensor_system.py --interval 60

7. CONFIGURACIÓN MQTT

Editar el archivo iot_sensor_system.py y modificar:

BROKER = 'xx.xxx.xxx.xxx'

PORT = xxxx

USERNAME = 'xxxx'

PASSWORD = 'xxxx'

8. DIAGNÓSTICO Y SOLUCIÓN DE PROBLEMAS

Servicio no inicia:

sudo journalctl -u iot-sensor-system -n 100

sudo systemctl status iot-sensor-system

python3 /home/usuario/DFRobot_RaspberryPi_Expansion_Board/iot_sensor_system.py --interval 10

Sensor no detectado:

sudo i2cdetect -y 1

ls /sys/bus/w1/devices/

sudo ./iot_service_manager.sh logs

Error de permisos:

sudo chown -R usuario:usuario / DFRobot_RaspberryPi_Expansion_Board pi/iot_sensor_system.py

sudo chmod +x /home/usuario/DFRobot_RaspberryPi_Expansion_Board

Problemas MQTT:

ping xxx.xxx.xxx.x

sudo journalctl -u iot-sensor-system | grep MQTT

9. MANEJO DE ERRORES DEL SISTEMA

- Reintentos automáticos hasta 3 intentos

- Timeout de 5 segundos para sensores

- Reconexión automática MQTT

- Operación continua si fallan algunos sensores

- Logs detallados de errores

- Reporte de estado de salud en cada mensaje

10. ACTUALIZAR EL SISTEMA

Detener servicio:

sudo ./iot_service_manager.sh stop

Reemplazar archivo:

sudo cp iot_sensor_system.py /home/usuario/ DFRobot_RaspberryPi_Expansion_Board/

Recargar y reiniciar:

sudo systemctl daemon-reload

sudo ./iot_service_manager.sh start

11. NOTAS IMPORTANTES

- Intervalo mínimo: 1 segundo

- Intervalo por defecto: 3600 segundos (1 hora)

- Logs en: /var/log/iot_sensor_system.log

- Servicio ejecutado con usuario **usuario**

- Envío de datos aunque fallen algunos sensores

12. SOPORTE Y CONSULTAS DE LOGS

Último arranque:

sudo journalctl -b -u iot-sensor-system

Últimas 24 horas:

sudo journalctl --since "24 hours ago" -u iot-sensor-system

Solo errores:

sudo journalctl -u iot-sensor-system -p err