INSTITUTO TECNOLÓGICO SUPERIOR DE TEZIUTLÁN

Tesis

SISTEMA DE MONITOREO Y CONTROL DE VARIABLES AMBIENTALES A INVERNADEROS UTILIZANDO TECNOLOGÍAS WEB

PRESENTA:

GIOVANI ZAMITIZ CÓRDOBA

CON NÚMERO DE CONTROL

21TE0007P

PARA OBTENER EL GRADO ACADÉMICO DE:

MAESTRA EN SISTEMAS COMPUTACIONALES

CLAVE DEL PROGRAMA ACADÉMICO

MPSCO-0127

DIRECTOR (A) DE TESIS:

Adriana Pérez López

CO- DIRECTOR DE TESIS:

Luis Alberto Espejo Ponce

"La Juventud de hoy, Tecnología del Mañana"

TEZIUTLÁN, PUEBLA, JULIO 2023

AGRADECIMIENTOS

Quiero expresar mi más sincero agradecimiento al Consejo Nacional de Ciencia y Tecnología (CONACYT) por brindarme la invaluable oportunidad de recibir una beca para cursar la Maestría en Sistemas Computacionales. Es un honor y un privilegio ser seleccionado para formar parte de este programa, y no puedo expresar con palabras lo agradecido que estoy por esta maravillosa oportunidad.

Asimismo, me gustaría expresar mi gratitud a mis profesores y mentores, quienes han sido una fuente constante de inspiración y conocimiento a lo largo de mi carrera. Su apoyo y orientación han sido fundamentales para mi crecimiento académico y personal.

Por último, pero no menos importante, quiero agradecer a mi familia y amigos por su apoyo incondicional durante este proceso. Su aliento y amor han sido un pilar fundamental en mi vida, y sin ellos, este logro no hubiera sido posible.

DEDICATORIA PERSONAL

A mi amada abuela Jovita Peralta Calderón.

Esta dedicatoria es en honor a ti, querida abuela. Recuerdo tus últimas palabras, donde mostraste interés en el tiempo que tomaría completar mis estudios de licenciatura. Aunque no pudiste presenciar mi titulación, te dedico el logro de mi posgrado en sistemas computacionales.

También quiero dedicar este logro a mis queridas mascotas Atka, Amaguq y Denahi, quienes fallecieron durante mis estudios de maestría, ellos fueron testigos de mis momentos de alegría y frustración, siempre brindándome su amor incondicional y su lealtad. A través de sus travesuras y cariño, encontré consuelo y compañía en los momentos más difíciles.

A todos ustedes, mi abuela y mis adoradas mascotas, les agradezco profundamente por su amor y apoyo incondicional. Este logro es un tributo a su influencia en mi vida.

Con amor y gratitud,

Giovani Zamitiz Córdoba.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

Este trabajo presenta la implementación exitosa de un sistema innovador que demuestra la eficiencia y el potencial del uso combinado de la plataforma de servicios en la nube de Amazon Web Services (AWS) y la potente tarjeta embebida Raspberry Pi en un entorno de invernadero. La investigación se llevó a cabo en el invernadero de la licenciatura de Biología del Instituto Tecnológico Superior de Tecnológico de Zacapoaxtla, con el objetivo de abordar un área de oportunidad en la región de Teziutlán, donde la agricultura protegida aún no se ha desarrollado ampliamente. El sistema desarrollado en este proyecto abarca una amplia gama de componentes electrónicos, diseño y programación personalizados para el monitoreo y control de variables ambientales en invernaderos agrícolas. Para lograr esto, se empleó la potencia de procesamiento y las capacidades de conectividad de la tarjeta Raspberry Pi 4, junto con los servicios altamente escalables y confiables de AWS, que incluyen la comunicación, el almacenamiento y la visualización de datos. Uno de los aspectos destacados de este proyecto es el diseño y la implementación de shields personalizados, que simplifican la conexión y protección de los sensores utilizados en el sistema. Estos shields fueron diseñados específicamente para adaptarse a las necesidades del invernadero y garantizar un funcionamiento óptimo de los sensores en un entorno agrícola. Para validar la efectividad del sistema implementado, se llevó a cabo una prueba utilizando el cultivo de suculentas piedras de luna en el invernadero. Además, se desarrollaron códigos de inicio y reset para automatizar el proceso y facilitar la operación del sistema. Los resultados de la prueba demostraron una excelente capacidad de control y monitoreo de variables ambientales, como la humedad del suelo y la temperatura, utilizando la combinación de AWS y Raspberry Pi. La evaluación del rendimiento de AWS en el sistema reveló un flujo de información eficiente entre los distintos servicios utilizados.

ABSTRACT

This work presents the successful implementation of an innovative system that demonstrates the efficiency and potential of the combined use of Amazon Web Services (AWS) cloud service platform and the powerful embedded Raspberry Pi board in a greenhouse environment. The research was carried out at the Biology greenhouse of the Tecnológico de Zacapoaxtla, aiming to address an opportunity area in the Teziutlan region where protected agriculture has not been widely developed yet. The system developed in this project encompasses a wide range of customized electronic components, design, and programming for monitoring and controlling environmental variables in agricultural greenhouses. To achieve this, the processing power and connectivity capabilities of the Raspberry Pi 4 board were employed, along with the highly scalable and reliable services of AWS, including communication, storage, and data visualization. One of the highlights of this project is the design and implementation of customized shields that simplify the connection and protection of sensors used in the system. These shields were specifically designed to meet the needs of the greenhouse and ensure optimal sensor performance in an agricultural environment.To validate the effectiveness of the implemented system, a test was conducted using the cultivation of moonstone succulents in the greenhouse. Additionally, startup and reset codes were developed to automate the process and facilitate system operation. The test results demonstrated excellent control and monitoring capability of environmental variables, such as soil moisture and temperature, using the combination of AWS and Raspberry Pi. The evaluation of AWS performance in the system revealed an efficient flow of information among the different services used.

INTRODUCCIÓN

Esta tesis se enfoca en la presentación del desarrollo de un sistema destinado a la automatización de invernaderos, haciendo uso de tecnologías web y sistemas embebidos. Actualmente, existe un mayor interés en la calidad de los cultivos para el consumo diario, llevando a la tendencia de que muchos intentan cultivar sus propios alimentos sin conocer los cuidados específicos de cada tipo hortalizas. Sin embargo, la calidad se puede interpretar de muchas maneras; los consumidores no confían en los alimentos cultivados tradicionalmente por la falta de información de ellos. Anaya D. (Anaya, 2020)comenta en su tesis "Prototipo de invernadero automatizado INDOOR 2, que los consumidores gradualmente van prefiriendo los alimentos orgánicos y sanos, esto lleva a consumir productos que fueron cultivados en agricultura protegida como invernaderos, macrotúneles o mallas de sombra.

Por otra parte, del 100% del espacio destinado la agricultura en territorio mexicano, sólo el 0.26% está ocupado por agricultura protegida, que, a pesar del poco espacio a nivel nacional, generó el 7% de valor de producción en el año 2021, dato estadístico que comparte el Servicio de Información Agroalimentaria y Pesquera (SIAP) de dependencia gubernamental. Enfocándonos a la región de Teziutlán del estado de Puebla, se tiene un espacio de 963.421 de Km2 destinado al sector agrícola, donde sólo el 0.007% está ocupado con agricultura protegida, cifra que está muy por debajo de la media nacional.

Es por ello que se propone una mejora para la automatización en los invernaderos para poder subir el índice de producción en los mismos, optimizando de mejor manera las variables ambientales que la planta requiere para su crecimiento con base en tecnologías web. Así generando una utilidad más rentable al agricultor, en base a una mejor calidad de producto para el consumidor.

CAPÍTULO I GENERALIDADES DEL PROYECTO

1.1 Marco teórico

1.1.1 Marco conceptual

Un marco conceptual es un esquema preliminar de un marco teórico sustentando la investigación, no solo contiene compuestos teóricos para comprender conceptos, sino que también constituye un método de recolección de datos que define o establece los límites de las categorías. de análisis que se pueden utilizar (Reidl-Martínez, 2012). Contando con los siguientes conceptos:

I.1.1.1 Sistema

Un sistema puede describirse como un conjunto coherente de reglas o principios interrelacionados que están lógicamente vinculados entre sí y se aplican a un determinado ámbito o tema (Real academia española, 2021). Proporciona una metodología para construir el modelo y especifica los procedimientos y técnicas a utilizar en los siguientes pasos:

1. Observar el comportamiento tal como se aplica al sistema real.
2. Caracterización de procesos y/o componentes esenciales.
3. Identificación de los arreglos de retroalimentación indispensables para explicar su comportamiento.
4. Estructura tu modelo cuantificando relaciones y atributos.
5. Adaptación del modelo a ordenador.
6. Implementación del modelo en un entorno de simulación.

Basado en la investigación, se seleccionó un sistema abierto. Tiene la propiedad de importar y procesar elementos (energía, materia, información) del medio ambiente.

Establece un intercambio permanente con el entorno que determina el equilibrio, la viabilidad y la continuidad. (Arnold & Osorio, 1998).

I.1.1.2 Modelo

El modelo es una herramienta creada con el propósito de reconocer y evaluar interacciones complejas en un sistema. En ocasiones, un sistema real puede requerir varios modelos para su representación, ya que esto depende de los objetivos del facilitador y de su capacidad para identificar las relaciones pertinentes que están asociadas con dichos objetivos. El modelado genérico de simplificación más conocido es el esquema de entrada-salida (input-output) como menciona Marcelo (1998).

I.1.1.3 Entrada-salida (Input-output)

- Input: La entrada de un sistema abierto implica la necesidad de recursos provenientes de su entorno. Estos recursos son captados por sensores de señales analógicas o digitales con fines de investigación.

- Output: Definida a la salida generada por un sistema. En general, los productos pueden ser categorizados en diferentes tipos de outputs, dependiendo de su propósito, como funciones, retroalimentación y servicios (Arnold & Osorio, 1998). Se concretan como disparadores o actuadores que efectúan cambios en el proceso de cambio o modificación del estado controlado. El término activador es un elemento modificador que proporciona energía para llevar a cabo una acción o comando de control.

I.1.1.4 Que es un sensor

Ramón Pallás comentó: El término sensor deriva del sentido medio de obtener conocimiento de cantidades físicas que son imperceptibles a los sentidos debido a su naturaleza o tamaño. Dada la amplia variedad de sensores disponibles para medir

diversas magnitudes físicas, resulta imperativo llevar a cabo una catalogación sistemática de los mismos con el fin de continuar nuestra investigación de manera racional. Con este propósito en mente, hemos establecido los siguientes criterios para clasificar y organizar los sensores (Pallás A, 2003, p. 6).

- Aporte de energía
    - Modulador o Activo: El componente de este sistema es responsable de generar la mayor parte de la potencia presente en la señal de salida, obtenida de una fuente auxiliar.
    - Generador o Pasivo: La fuente de energía que proporciona la potencia de salida puede ser un generador o un sistema pasivo, ya que la potencia de entrada determina la cantidad de potencia disponible.

- Señal de salida.
    - Analógico: En los sensores analógicos la salida varía continuamente a nivel macroscópico y la información es en amplitud y frecuencia.
    - Digital: Con un sensor digital, la salida cambia en saltos o pasos discretos. Los modelos digitales están ausentes para muchas de las dimensiones físicas más importantes, pero han mostrado mayor confiabilidad y precisión.

- Modo de funcionamiento.
    - Deflexión: Un sensor que funciona por deflexión. Las cantidades medidas producen efectos físicos que producen efectos similares pero opuestos.
    - Comparación: Para los sensores que funcionan por comparación, la deflexión debe mantenerse en cero aplicando un efecto contrario al producido por la cantidad que se mide.

En la Tabla 1 se resumen las taxonomías se representan en términos muy generales con ejemplos de sensores en cada clase.

Tabla 1. Clasificación de sensores (Pallás A, 2003, p. 7).

Criterio	Clase	Ejemplo
Aporte de energía	Moduladores	Termistor
	Generadores	Termopar
Señal de salida	Analógicos	Potenciómetro
	Digitales	Codificador de posición
Modo de operación	De deflexión	Dinamómetro
	De comparación	Bascula

I.1.1.5 Señal analógica

Una señal analógica consiste en cambios eléctricos que evolucionan en variables físicas (temperatura, humedad, luminosidad, etc.) ejemplo en la Figura 1.

Figura 1. Señal analógica.

Estas variables pueden representarse en forma de corriente, voltaje o carga, que varía constantemente entre los límites superior e inferior. Cuando estos límites coinciden con los soportados por el dispositivo (el sensor), la señal se normaliza (escala) para hacer un mejor uso de la relación señal-ruido del sensor (Miyara,

2004). El ruido es un conjunto de perturbaciones que alteran la señal debido a las distorsiones provocadas por el sistema y es un componente muy importante de la limitación del ancho de banda de un sistema de comunicación.

I.1.1.6 Señal digital

Las señales digitales son variaciones eléctricas con dos niveles que se alteran en el tiempo transfiriendo información según un código previamente establecido. Cada nivel eléctrico representa uno de dos símbolos: verdadero(V) o falso(F), 0 o 1, voltaje bajo o voltaje alto, etc. Los niveles específicos dependen mucho de los componentes específicos, por ejemplo, para la familia de los sensores CMOS (complementary metal-oxide-semiconductor), los valores dependen de la alimentación, para una alimentación de 5v (voltaje), teniendo los dos niveles, reconociendo de 0v a 2.25v para un valor 0 y de 2.75v a 5v para el valor 1. Este ejemplo muestra su gran inmunidad al ruido para las señales digitales que tiene la particularidad de solo tener 2 estados por lo que permite transmitir, representar y almacenar información binaria (Miyara, 2004). Su forma característica de las señales digitales es una onda cuadrada o pulsos como se muestra en la Figura 2, teniendo los siguientes parámetros:

- Duración o ancho de pulso.
- Velocidad de repetición por segundo.
- Pulso alto.

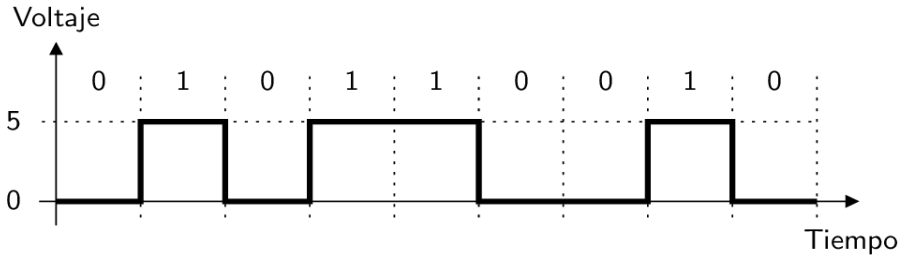


Figura 2. Señal digital.

I.1.1.7 Sistema de medición

La Real Academia Española afirma que la medida es "una comparación cuantitativa con sus propias unidades para ver cuántas veces está la segunda en la primera" (Real academia española, 2021). En electrónica, puede pensarse que la medida consiste en tres elementos básicos como se muestra en la Figura 3. Sistema de medición y sus componentes (Bolton, 2017).

- Sensor: responde a una magnitud que se mide y da una señal a una salida asociada a esta magnitud.
- Procesador de señal: recibe la señal enviada desde el sensor y la procesa en el estado adecuado para representarla visualmente o en el sistema de control.
- Sistema de visualización: donde se muestra la salida del procesador de señal, puede ser, por ejemplo, una flecha animada, balanzas o displays digitales.

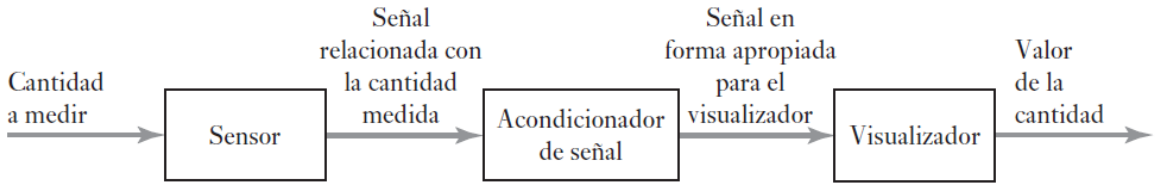


Figura 3. Sistema de medición y sus componentes (Bolton, 2017).

I.1.1.8 Sistema de control

Un sistema de control es un programa que puede utilizarse para:

1. Controlar una variable por una cantidad determinada, por ejemplo, un sistema de calefacción cuya temperatura está controlada por un valor determinado.

2. Control de una secuencia concreta de eventos, por ejemplo, en una lavadora, donde y cuando el ciclo está fijado, por ejemplo "color", entonces un ciclo de lavado concreto es controlado por la máquina, es una secuencia de eventos.

3. Controlar si sucede o no un evento, por ejemplo, un seguro en una máquina en donde no puede ser operada hasta que esté en posición el dispositivo de seguridad.

Existen dos tipos básicos de sistema de control, de lazo abierto y lazo cerrado. El de lazo abierto se caracteriza por no recibir ninguna retroalimentación o información sobre el estado de la variable, por lo regular son utilizados cuando la variable es predecible, ya que se puede calcular las veces o el tiempo en que se debe de repetir el ciclo para completar su proceso, teniendo la ventaja de ser sencillos por lo que el costo es bajo, sin embargo, con frecuencia son imprecisos ya que no poseen una corrección de errores.

El control de lazo cerrado es un sistema más completo ya que recibe una retroalimentación sobre los estados que va tomando la variable, esto se logra con la colocación de sensores que envían información de puntos clave del proceso para que así pueda actuar de manera automática, tienen la ventaja de ser bastantes precisos para emparejar el valor real con el requerido, aunque son más complejos y costosos debido a la cantidad de componentes incluidos (Bolton, 2017, pp. 9–12).

I.1.1.9 Temperatura ambiente

La temperatura es una variable importante en el cultivo y desarrollo de las plantas, afectando constantemente a los organismos. La mayoría de los procesos biológicos se aceleran con el aumento de la temperatura, que puede ser algo perjudicial, y la hiperventilación resultante es desfavorable, ya que significa que el crecimiento de los frutos carece de energía, lo que comportará una reducción del tamaño del producto (Anaya, 2020).

I.1.1.10 Humedad del suelo

El significado de la humedad del suelo es determinar el valor del agua total en una cantidad determinada de suelo conocida, que puede expresarse en porcentaje, volumen o peso. La humedad suficiente en el suelo es una condición muy importante para el crecimiento de las plantas, puesto que actúa no sólo como restaurador de humedad sino también como regulador de temperatura. Si la planta pierde demasiada agua, las estomas se cierran, parando la fotosíntesis, y si esto ocurre no podrá consumir dióxido de carbono (C02) (Anaya, 2020).

I.1.1.11 Iluminación

Las plantas necesitan principalmente luz y agua para la fotosíntesis; Sin embargo, las cantidades pueden variar según las especies: algunas plantas requieren mucha luz, mientras que otras pueden sobrevivir con poca luz. J. López (2017) en su artículo "La importancia de la luz para las plantas" afirma que demasiada y poca luz pueden tener efectos negativos, y demasiada luz puede hacer que las hojas pierdan su aspecto verde y se vuelvan blancas con manchas amarillas y marrones. en los bordes. Por el contrario, cuando la planta absorbe menos luz de la que necesita, la planta parece débil. Rara vez florece, y las flores son pequeñas y caen antes de terminar de crecer. Los tallos son frágiles y delgados, y las hojas se vuelven amarillas y finalmente se derrumban.

1.1.2 Invernadero

El invernadero representa una herramienta clave de la agricultura protegida (AP) y detalla dos aspectos importantes, el primero es la eficiencia para condicionar algunos de los principales elementos del clima dentro de los límites determinados de acuerdo con los requerimientos fisiológicos de un cultivo en específico; la segunda es la funcionalidad, definida como el conjunto de requisitos que permiten la mejor

eficiencia del invernadero, tanto del punto de vista económico como técnico. Se justifica mediante el aspecto de calidad que se está viviendo a nivel mundial, donde el mercado es más exigente en el producto demandando una mejor presentación, calidad y certificación, ya que el consumidor observa las diferencias entre el producto que se presenta con respecto a otros, esto hace que los cultivos en invernaderos se encuentren en un estándar de alto nivel (Fundación Produce Sinaloa, 2006). En México los primeros invernaderos de tipo comercial se colocaron en los años cincuenta y setenta en varias partes del país, elaborados con construcciones de herrería, concreto y cristal destinadas a la producción de plantas ornamentales, con el paso del tiempo han ido evolucionando tanto en los materiales de construcción como en su diseño, en la actualidad se encuentra una amplia gama donde el productor agrícola pueda escoger el que se ajuste con sus requerimientos de producción.

Sin embargo, se utilizará un invernadero tipo túnel o semicilíndrico, el cual se caracteriza por la forma de su estructura metálica y su cubierta (Figura 4). Este tipo de invernadero es el más usado por su mayor capacidad de control en las variables ambientales, su rapidez de instalación por ser estructuras ya prefabricadas y su gran resistencia a fuertes vientos (Santizo V., 2011).

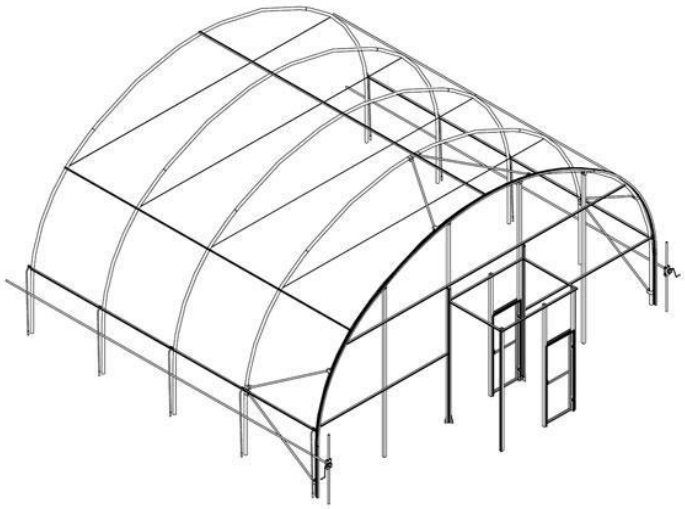


Figura 4. Invernadero tipo túnel.

Las ventajas de un invernadero tipo túnel:

- Buena ventilación.
- Buen reparto de la luminosidad en el interior.
- Buena estanqueidad a la lluvia y al aire.
- Fácil instalación.
- Permite la instalación de ventilación y facilita su acondicionamiento.
- Estructuras con pocos obstáculos.

1.1.3 Internet de las cosas

IoT (Internet Of Things) significa cosas conectadas que están equipadas con sensores, software y otras tecnologías que les permiten transmitir y recibir datos – hacia y desde otras cosas (SAP Insights, 2022), cómo se observa en la Figura 5.

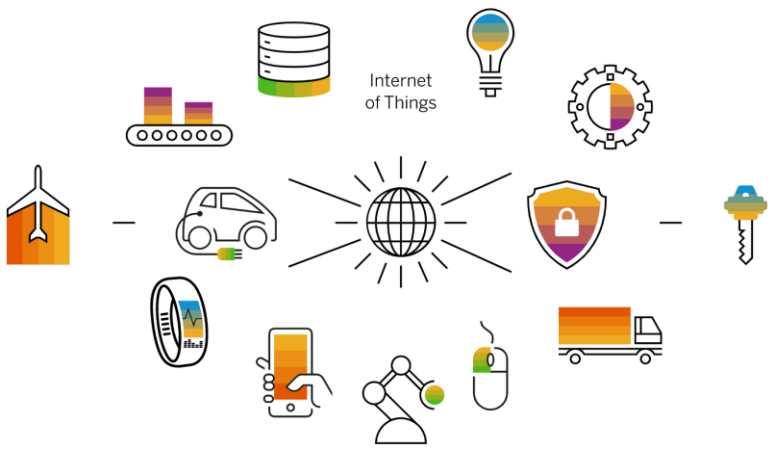


Figura 5. Esquema general de IoT (SAP Insights, 2022).

La red de área doméstica consta de diferentes dispositivos basados en IoT, como electrodomésticos y sensores que tienen interfaces de comunicación que facilitan el intercambio de información. Estos dispositivos permiten recopilar información sobre el estado de los electrodomésticos, consumo de energía eléctrica, condiciones

ambientales de la habitación (temperatura, humedad, e iluminación natural), el comportamiento de los usuarios, entre otros (Paredes-Valverde et al., 2020). Los beneficios de implementar sistemas IoT en el proyecto, son evidentes desde el primer momento: capacidad de conectarse a la red, ahorro energético, comunicación con el entorno directo, intercambio de información de forma rápida y en tiempo real y procesos más sostenibles. Las características y los servicios de IoT son increíblemente diversos, pero es importante identificar la información crítica que realmente tiene valor.

1.1.4 Servicios Web

Los Servicios web son aplicaciones que permiten una flexibilidad mejor a los procesos de negocio con ayuda de módulos que pueden publicar, describir, localizar y usar a través de la red. Estos servicios reflejan un enfoque Service Oriented Architecture (SOA), esta orientación se basa principalmente en crear aplicaciones mediante el desarrollo e implementación de servicios en la red, también en la activación de aplicaciones disponibles con la finalidad de realizar una tarea destinada (IBM, 2022). Algunas de las ventajas para la implementación de esta tecnología son las siguientes:

- Proporcionan interoperabilidad entre aplicaciones de software independientemente de sus características o de la plataforma en la que estén instaladas.
- Los servicios web promueven estándares y protocolos basados en texto que facilitan el acceso al contenido y la comprensión de su funcionamiento.
- Mediante el uso de HTTP, los servicios web pueden aprovechar los sistemas de seguridad de firewall sin necesidad de cambiar las reglas de filtrado.
- Los servicios y el software de diferentes empresas en diferentes ubicaciones geográficas se pueden combinar fácilmente para brindar un servicio integrado.

I.1.4.1 Amazon Web Services (AWS)

Amazon Web Service ofrece una amplia gama de productos basados en la nube, que incluyen bases de datos, cómputo, IoT (Internet de las cosas), análisis, herramientas para desarrolladores, almacenamiento, redes, aplicaciones empresariales, seguridad y tiempo de actividad en segundos, algunos servicios tienen una tarifa de uso o un período de prueba de hasta un año. Una de las principales ventajas de AWS es la capacidad de reemplazar toda la infraestructura y planificar la compra de servidores y otra infraestructura de TI, proporcionando una plataforma rentable basada en la nube. La alta precisión y la escalabilidad respaldan a muchas organizaciones en todo el mundo (Amazon, 2022b). Dentro de sus productos cuenta con AWS IoT Core, un servicio en la nube administrado que permite a los dispositivos conectados se puedan comunicar de manera fácil y segura con otros dispositivos y aplicaciones interactivas mediante software en la nube (Figura 6).

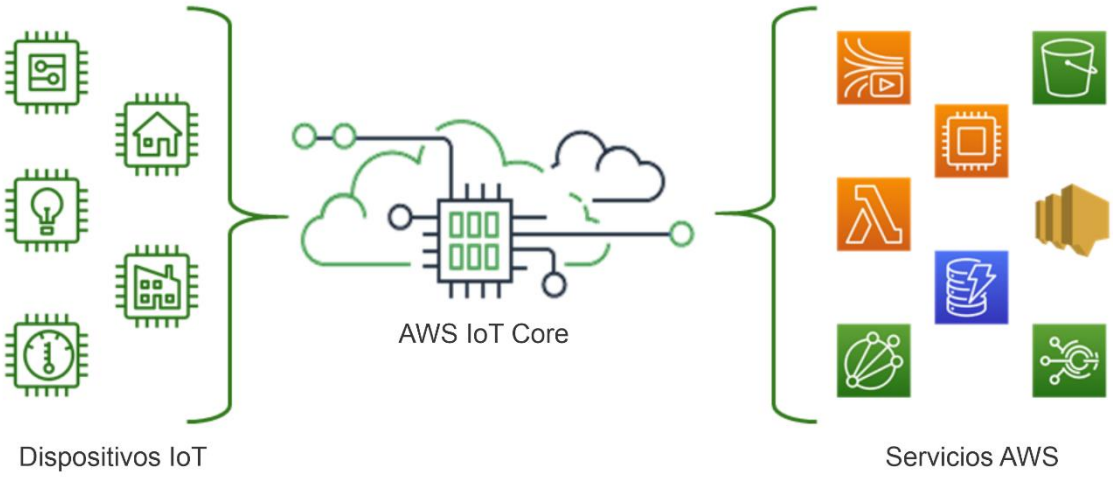


Figura 6. Esquema de AWS IoT (Amazon, 2022a).

El servicio admite millones de dispositivos y billones de mensajes, y es capaz de procesarlos y encaminarlos a los puntos finales de AWS, por lo que es fácil de utilizar AWS lamda, Amazon Kinesis, Amazon S3, Amazon CloudWatch, AWS CloadTrail,

Amazon SageMarket, Amazon Dynamics Database, y otros servicios. Cree aplicaciones IoT con Amazon QuickSight para manipular, recopilar, analizar y utilizar datos generados por los dispositivos conectados. El AWS IoT Core le permite elegir la opción de solución más adecuada para gestionar y apoyar los dispositivos IoT en el campo gracias a la compatibilidad con los siguientes protocolos (Amazon, 2022a):

- MQTT (Cola de mensajes y transmisión de telemetría).
- MQTT sobre WSS (Websockets Secure).
- HTTPS (Protocolo de transferencia de hipertexto - Seguro).
- LoRaWAN (Long Range Wide Area Network).

1.1.5 Lenguaje de programación Python

Python es un lenguaje de programación de alto nivel utilizado para desarrollar todo tipo de aplicaciones. Es un lenguaje interpretado, a diferencia de otros lenguajes como Java o .NET., esto significa que las aplicaciones escritas en Python no necesitan ser compiladas para ejecutarse, ni ser traducidas a lenguaje de máquina ya que son ejecutadas directamente por la computadora usando un programa llamado intérprete. Es un lenguaje fácil de leer y escribir porque es muy similar al lenguaje humano. Además, es un lenguaje multiplataforma de código abierto, por lo que es gratuito, permitiendo desarrollar software sin límites. Con el tiempo, Python ha ganado seguidores gracias a su simplicidad y amplia gama de posibilidades (Becas Santander, 2022).

Una de las mayores ventajas de Python es que puede usar para muchos propósitos, es una buena opción para el desarrollo de software porque permite que los desarrolladores usen excelentes marcos como Django y Flask, además, Python es un lenguaje multiparadigma que admite programación estructurada, funcional y orientada a objetos. Entre las ventajas más relevantes que cuenta es la gran variedad de bibliotecas y marcos diferentes, la biblioteca estándar de Python es muy

rica con muchos módulos integrados, encontrando otras bibliotecas disponibles en el directorio de paquetes de Python (PyPI). En ciencia de datos destacan bibliotecas como TensorFlow, PyTorch o NumPy que manejan funciones matemáticas y científicas (El Tutorial de Python — Documentación de Python - 3.11.0, n.d.).

1.1.6 Sistemas embebidos

Los sistemas integrados o embebidos incluyen hardware informático y software integrado como uno de los componentes principales del sistema informático dedicado a una aplicación o producto. Es un sistema independiente o parte de un sistema más grande, con su software generalmente está integrado en la ROM (memoria de solo lectura), por lo que no requiere almacenamiento secundario como una computadora (Arevalo et al., 2009), sistema integrado consta de tres componentes principales:

- Hardware.
- Software principal o aplicación principal. Este software o aplicación realiza una tarea específica o posiblemente un conjunto de tareas.
- Un sistema operativo que, además de proporcionar un mecanismo para ejecutar procesos, permite monitorear aplicaciones. Muchos sistemas integrados requieren propiedades en tiempo real en el sistema operativo.

Los sistemas embebidos tienen ciertas características que los distinguen de otros sistemas informáticos. Los más importantes se describen a continuación.

1. Ciertas operaciones. Los sistemas integrados normalmente ejecutan un programa en particular repetidamente. Por el contrario, el sistema de escritorio ejecuta una variedad de programas como: B. Hojas de cálculo, juegos, etc. Además, se agregan

nuevos programas con frecuencia. Por supuesto, puede ocurrir una excepción. Es posible que el programa del sistema integrado se haya actualizado a una versión más nueva. Por ejemplo, los teléfonos móviles pueden actualizarse de alguna manera.

2. Límite estricto. Todos los sistemas informáticos tienen limitaciones en sus métricas de diseño, pero los sistemas integrados son muy potentes. Una métrica de diseño es una medida de alguna característica de implementación, como: Costo, tamaño, rendimiento, consumo de energía. Los sistemas integrados generalmente deben ser de bajo costo, ocupar poco espacio y tener un buen rendimiento para procesar datos en tiempo real para prolongar la vida útil de la batería y evitar la necesidad de refrigeración adicional de los componentes.

3. Reactivos y tiempo real. Muchos sistemas integrados necesitan reaccionar a los cambios ambientales, además de la computación en tiempo real sin latencia. Por ejemplo, en el, el módulo de Control del automóvil monitorea continuamente los sensores de velocidad y freno del y reacciona ante cualquier eventualidad. Ante un estímulo anómalo, el módulo de control debe realizar cálculos con precisión y rapidez para garantizar que brinde resultados en tiempo y forma, violar este tiempo puede resultar en la pérdida de control del auto. Por el contrario, el sistema de escritorio se centra en realizar cálculos con una frecuencia no especificada y sus retrasos no provocan el tiempo de inactividad del sistema (Arevalo et al., 2009).

I.1.6.1 Raspberry -PI 4

La placa integrada Raspberry Pi es una computadora de bajo costo y tamaño compacto que se conecta a un monitor de computadora o TV, así como a periféricos como mouse y teclados estándar. La tarjeta se ejecuta en el sistema operativo basado en Linux Raspberry Pi OS (anteriormente Raspbian) para navegar por Internet, reproducir vídeos de alta definición, Scratch y Python.

El Raspberry Pi Model B 4 es la incorporación más reciente a la línea y ofrece una velocidad de procesador, un rendimiento, una memoria y una conectividad mejorados en comparación con el Raspberry Pi 3 Model B de la generación anterior compatible con versiones anteriores. Las características clave de la versión incluyen un potente procesador de cuatro núcleos de 6 bits, soporte para dos pantallas de resolución 4K completa a través de un par de puertos micro HDMI, diferentes versiones de RAM (2 GB, GB y 8 GB) e incluye puerto Gigabit Ethernet como se muestra en la Figura 7. Raspberry-pi 4 (Raspberry Pi, 2019).

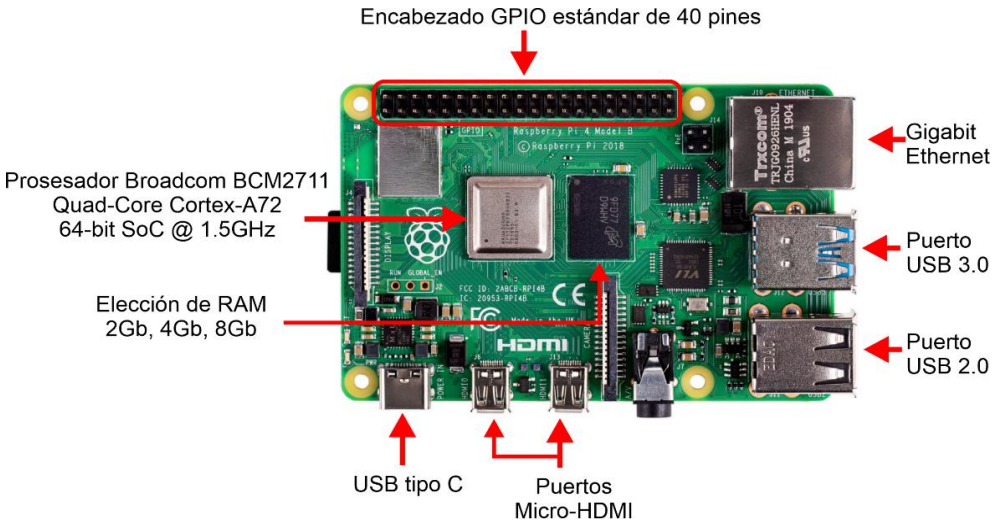


Figura 7. Raspberry-pi 4 (Raspberry Pi, 2019).

1.1.7 Sensores

Los sensores son una parte importante para la recolección de las mediciones dentro del microclima en el invernadero, por ello se optaron por sensores que sean resistentes a las condiciones del medio ambiente para evitar la corrosión de los equipos y el reemplazo continuo de los mismos, siendo los siguientes:

I.1.7.1 Sensor de temperatura DS18B20

Figura 8. Sensor DS18B20.

El sensor de temperatura digital DS18B20 (Figura 8) de Dallas Semiconductor está recubierto con un sellador de alta conductividad térmica para garantizar la precisión del sensor con una variación de temperatura mínima. El sensor admite una interfaz de "bus de un cable" (1-Wire) en el rango de temperatura de -55 °C a +125 °C, con una precisión de ±0,5 °C en el rango de -10 °C a +85 ° C. Temperatura ambiente, la transmisión directa en formato digital mejora enormemente la interferencia en el sistema, adecuado para la medición de temperatura en condiciones adversas (Xi 'an Gavin, 2008). El sensor de temperatura digital DS18B20 tiene un número único y el colector de temperatura identifica el sensor de los demás si se utiliza en paralelo. Sus características son las siguientes:

- Rango de alimentación: 3.0v to 5.5v.
- Rango de medición de temperatura: -55 °C a + 125 °C (-67°F a +257°F).
- Temperatura de almacenamiento: -55 °C a + 125 °C (-67°F a +257°F).
- Rango de precisión: -10 °C a + 85 °C ±0.5 °C.
- Recubierto de acero inoxidable.
- Resistente al agua.
- Tamaño de la funda: 6mm x 50mm.

Cada pin del chip está separado por tubo termo retráctil para evitar corto circuitos, mostrando el componente dentro de una sonda (Figura 9).

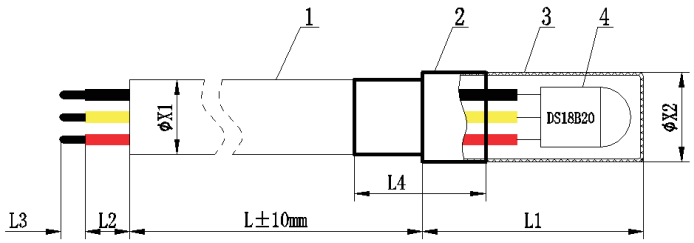


Figura 9. Partes de la sonda con el sensor DS18B20.

1. Cable PVC de 3 hilos con diámetro de 5 mm.
2. Tubo termo retráctil de 6 mm de diámetro.
3. Sonda de acero inoxidable de 6 mm de diámetro con 50 mm de largo.
4. sensor DS18B20.
    a. Rojo: alimentación.
    b. Amarillo: entrada/salida de señal digital.
    c. Negro: tierra (GND).

I.1.7.2 Sensor de humedad del suelo HD-38

Figura 10. sensor HD-38.

El sensor de humedad del suelo HD-38 (Figura 10) es un dispositivo electrónico desarrollado específicamente para diseñadores que necesitan un sensor de alta calidad que pueda mostrar datos de manera más precisa y confiable.

Una diferencia llamativa de este modelo es que está fabricado con materiales de alta calidad que garantizan una mayor durabilidad. La sonda del sensor de humedad del

suelo HD-38 incluye una punta de metal de 90 mm de largo que se conecta fácilmente al suelo y un cable de conexión de 1 metro de largo. Por ejemplo, para facilitar la conexión a un Arduino o Raspberry, incluye un módulo que lee los datos generados por la sonda y envía la información al microcontrolador a través de una interfaz digital o analógica (OXDEA, 2020). Sus especificaciones son las siguientes:

- Rango de alimentación: 3.0v to 12v.
- Sensibilidad ajustable mediante potenciómetro.
  Salida digital y analógica.
- Recubierto de acero inoxidable.
- Fácil instalación.
- Corriente de funcionamiento: < 20 mA.
- Dimensiones: 36 x 15 x 7 mm.

I.1.7.3 Sensor de calidad del aire MQ135

Figura 11. Sensor MQ135.

El material sensible del sensor de gas MQ135 (Figura 11) es dióxido de estaño (SnO_2), que tiene baja conductividad en aire limpio. En presencia del gas objetivo contaminante, la conductividad del sensor aumenta al aumentar la concentración de gas. Se puede convertir el cambio de conductividad en una señal de salida correspondiente a la concentración de gas a través de un circuito simple. El sensor de gas MQ135 tiene una alta sensibilidad a los vapores de las series de amoníaco, sulfuro y benceno y también puede monitorear gases tóxicos como el smog. Puede

detectar muchos gases tóxicos y es un sensor económico adecuado para una variedad de aplicaciones (Winsen, 2015). Sus especificaciones son las siguientes:

- Rango de alimentación: 5v ±0.1v.
- Rango de detección de 10 ~ 1000ppm (partes por millón) de gas de amoníaco, tolueno, hidrógeno, humo.
- Encapsulación estándar baquelita, Tapa metálica.
- Salida digital y analógica.
- Recubierto de acero inoxidable.
- Fácil instalación.
- Consumo ≤950mW.

1.1.8 Suculentas piedras de luna

Las suculentas Piedra de Luna, también conocidas como Pachyphytum oviferum, son plantas nativas de México. Tienen hojas gruesas y redondeadas que se asemejan a pequeñas piedras lunares. Para un cultivo exitoso, es importante considerar las condiciones ambientales adecuadas (Figura 12).

Figura 12. Suculenta piedra de luna.

En cuanto a la luz, estas suculentas requieren exposición adecuada a la luz solar, preferentemente en lugares brillantes con luz indirecta o moderada. Evitando la luz solar intensa durante las horas más calurosas para evitar quemaduras en las hojas.

En términos de temperatura, prefieren climas moderados, evitando temperaturas extremas. La temperatura ideal para estas suculentas oscila entre los 18 °C y 34 °C.

El suelo es un factor clave. Utilizando un sustrato bien drenado que permita un rápido escurrimiento del agua. En cuanto al riego, estas suculentas prefieren un régimen moderado. Permitiendo que el suelo se seque por completo entre riegos para evitar problemas de pudrición de raíces. La humedad del suelo debe mantenerse entre 0% y 30%.

1.2 Planteamiento del Problema

En la economía interna de un país, la industria del sector agrario es un importante pilar porque, además de aumentar el Producto Interior Bruto (PIB), también crea oportunidades de empleo directo y contribuye a reducir la inflación. El sector agrícola de México representó 2,35% del PIB nacional en 2019. La agricultura mexicana también tiene una presencia internacional importante, siendo el 8º exportador y el 11º productor del mundo (Gómez, 2020). Debido a una serie de eventos en los últimos años, como la pandemia del virus SARS-CoV-2. (Covid-19), que afectó a la producción y suministro de algunos alimentos.

Es evidente que México está atravesando un período muy complejo en cuanto a la productividad, añadiendo que en el año 2021 ha habido factores climáticos desfavorables como la sequía en algunos puntos del otro factor estresando la actividad ganadera (SIAP, 2021). Pese a las condiciones desfavorables, el sector agrícola mexicano produjo 268,6 millones de toneladas, un 1,3% más que el pasado año, con una superficie de 21,7 millones de hectáreas (SIAP, 2022).

Sin embargo, el clima es un factor esencial en la agricultura que afecta al crecimiento y desarrollo de diversos cultivos. Según el INEGI (Instituto Nacional Mexicano de

Estadística y Geografía), el país tiene hasta ocho climas distintos, destacando tres grandes grupos debido a su amplia expansión territorial, tal y como muestra en la Tabla 2. Tres principales climas del territorio mexicano (Gómez, 2020).

Tabla 2. Tres principales climas del territorio mexicano (Gómez, 2020).

Clima	Ubicación Geográfica	Rango de temperaturas	Rango de precipitación media
Templado húmedo y templado subhúmedo.	Parte central del territorio mexicano.	10 a 22ºC	Media entre los 600 y 4.000 mm de lluvia.
cálido húmedo y cálido subhúmedo.	Se encuentra en la parte sur, sudeste y parte de la costa este.	media anual de 26ºC relativamente constante	Media entre los 1000 y 4.000 mm de lluvia.
Seco y muy seco	Mayoritariamente en la parte norte y centro del país	18-22ºC, aunque puede llegar a ser extrema en zonas desérticas.	Media entre los 100 y 600 mm de lluvia.

A pesar de la variedad de climas que cuenta el país, México tiene alrededor del 9% del territorio nacional (1,964 miles de km2) ocupado para el sector agrícola, lo que en superficie continental se traduce como 181.51 miles de km2, este espacio se clasifica en dos apartados: Cultivo a cielo abierto y cultivo en agricultura protegida, en la Tabla 3 se especifica el área destinada como el valor de producción de cada

una desde un nivel Nacional, estatal y regional, especificando a Puebla como entidad federativa, en la región de Teziutlán (SIAP, 2021).

Tabla 3. Tabla de comparación entre cultivo a cielo abierto y AP.

Criterio	Nacional	Estatal - Puebla	Región de Teziutlan
Territorio destinado	181.503 miles de km^2	8,625 miles de km^2	963.421 de km^2
Territorio para el cultivo a cielo abierto	181.037 miles de km^2	8,612 miles de km^2	963.350 de km^2
Territorio para el cultivo en agricultura protegida	472.539 de km^2	13.1956 de km^2	0.071 de km^2
Valor de producción en millones de pesos (MDP)	692,828.56 MDP	19,690.71 MDP	2,198.82 MDP
Valor de producción en cielo abierto	644,342.44 MDP	18,424.69 MDP	2,194.39 MDP
Valor de producción en agricultura protegida	48,486.11 MDP	1,266.02 MDP	4.427 MDP

Como podemos apreciar en la tabla, solo el 0.26% del territorio a nivel nacional está destinado al cultivo en AP, no obstante, el valor de producción generado es del 7% lo que podemos representar que por cada kilómetro cuadrado de AP genera. Pasando al nivel estatal se observa un comportamiento similar del 0.15% del territorio propuesto a AP con un valor de producción del 6.43%. Por último, a nivel regional se establece sólo el 0.007% de la superficie para AP con un valor de producción del 0.202%, lo que indica que en la región de Teziutlán se encuentra muy debajo de la media en la utilización de invernaderos, malla de sombra o macro túnel (SIAP, 2021). Es por ello que la importancia de poner en práctica la agricultura

protegida, tecnología que ayude a controlar mejor las variables ambientales, aumentará el valor de producción con menos área destinada.

1.3 Justificación

El dominio del entorno cultural es un factor importante para promover la salud humana. Según Guadalupe García S. (García S, 2020) en su artículo "Tecnificación De Invernaderos Para La Producción De Jitomate (Solanum Lycopersicum L.), En Tetela De Ocampo, Puebla, México". En el pais, la demanda de alimentos es alta, lo que conlleva la explotación del agua y la tierra, puesto que la agricultura de conservación pretende aprovechar al máximo estos recursos. Como ya se sabe, la mayoría de los invernaderos están diseñados y construidos para ofrecer un entorno global ideal a gran escala. La gestión de estas variables ambientales tiene un principio básico sencillo, añadiendo complejidad según las necesidades y circunstancias; Sin embargo, la complejidad viene con un precio elevado o caro, que afecta directamente al uso generalizado. En términos comerciales, no todos los proveedores rastrean y controlan las variables ambientales, principalmente por la falta de precios razonables (Anaya, 2020).

El proyecto proporcionará una mejor condición en el crecimiento del cultivo seleccionado. Representará un beneficio hacia el cultivo como el agricultor ya que al incorporar el soporte con las tecnologías web y el uso de sistemas embebidos, tecnologías que no se han llegado a incorporar en la región de Teziutlán. Facilitando el control de las variables del ambiente donde se delimitará un margen para las condiciones óptimas del cultivo a selección. Esto no solo tendrá un impacto donde se mejorará la calidad del producto, si no también se tendrá la optimización de los recursos durante su tiempo de desarrollo hasta la cosecha.

1.4 Hipótesis

Con base al área de oportunidad se espera obtener un crecimiento óptimo para el cultivo en un invernadero tipo túnel de pequeños productores, donde incorpore tecnologías web con soporte de sistemas embebidos.

1.5 Objetivo general

Desarrollar e implementar sistema de monitoreo y control de variables ambientales a invernaderos tipo túnel en la región de Teziutlán, Pue. Con base en tecnologías web con la finalidad de mejorar el crecimiento de los cultivos.

1.6 Objetivos específicos

- Diseñar un sistema de instrumentación de variables ambientales.
- Generar la conexión de un sistema embebido con los Servicios web.
- Desarrollar un sistema de gestión de datos que permita almacenar y consultar el historial de mediciones.
- Realizar pruebas de control y medición para verificar el correcto crecimiento de los cultivos.

1.7 Alcances y limitaciones

1.7.1 Alcances

- Optimización de las variables ambientales necesarias para el crecimiento de las plantas en el invernadero.
- Mejora en la calidad de los productos cultivados en el invernadero.
- Incremento en la rentabilidad para los agricultores al generar una producción más eficiente.

- Implementación de un sistema de monitoreo y control de variables ambientales en invernaderos tipo túnel en la región de Teziutlán, Puebla.
- Conexión de un sistema embebido con servicios web para facilitar el control y gestión de datos.

1.7.2 Limitaciones

- El proyecto se enfoca en invernaderos tipo túnel y no abarca otros tipos de estructuras agrícolas protegidas.
- La implementación del sistema puede requerir una inversión inicial significativa en tecnologías web y sistemas embebidos.
- Las pruebas de control y medición pueden requerir tiempo y recursos adicionales.
- El sistema de automatización puede tener limitaciones en términos de la variedad de cultivos que se pueden manejar de manera óptima.
- La automatización no puede acelerar significativamente el tiempo de desarrollo de los cultivos debido a los períodos específicos de crecimiento y madurez requeridos por cada planta.

CAPÍTULO II ESTADO DEL ARTE

2.1 Trabajos relacionados

A continuación, se presentan trabajos de varios proyectos nacionales e internacionales, relacionados con el problema de búsqueda de la gestión de variables ambientales mediante el uso de tecnologías móviles, web y múltiples sistemas embebidos.

2.1.1 Internacionales

Jorge Arcenio Q. (Jorge Arcenio, 2021) implementó un proyecto titulado "Sistema para el manejo de datos climáticos de pequeñas producciones agrícolas bajo invernadero: un acercamiento a la agricultura inteligente". Se basa en recoger variables ambientales mediante sensores baratos y procesarlas mediante una placa Arduino UNO, conectada a una aplicación web mediante una base de datos FireBase en tiempo real. El proyecto se implementó en un invernadero de tomate (Solanum lycopersicum) en Ubaque, Cundinamarca, Colombia. Evidencia de que el prototipo funciona, la vida útil de Arduino se estima en 3-5 años y la vida útil del sensor se estima en 1 año, por lo que debe sustituirse constantemente, porque el sensor no es adecuado para un uso masivo, mientras que la API web se comunica con la aplicación móvil y se ha simplificado la API de Firebase.

Continuando con la investigación de Blanco Rico y Martínez Zarzuela de la universidad de Valladolid, España (Blanco Rico & Martínez Zarzuela, 2021) con el título "Sistema de telemedida y sensorización mediante red LoRaWAN", basado desarrollo de un sistema de monitorización de parámetros ambientales utilizando tecnologías emergentes de IoT. Trabajando en colaboración con la empresa Energibid, buscaban expandir su alcance más allá de la telemedida eléctrica y

adentrarse en la monitorización de variables como el consumo de agua, gas y variables ambientales como temperatura, humedad y calidad del aire. El objetivo principal había sido desarrollar un sistema óptimo y adaptado a las necesidades propuestas, centrándose en la calidad del aire en un colegio para garantizar una ventilación adecuada en las aulas. El sistema había utilizado tecnología LoRa para establecer comunicaciones inalámbricas de larga distancia, se había implementado utilizando dispositivos de adquisición de datos con Arduino y se había presentado un prototipo completo con una aplicación móvil para visualizar los datos obtenidos. El sistema había cumplido con los requisitos propuestos y se había demostrado su fiabilidad en situaciones adversas, sentando las bases para futuras mejoras y extensiones. En resumen, se había desarrollado un sistema de monitorización de parámetros ambientales utilizando tecnología IoT, específicamente enfocado en la calidad del aire en un colegio. El sistema ha ofrecido ventajas en términos de automatización, control y reducción de costes, lo que ha brindado oportunidades para futuros desarrollos y mejoras en el campo de la telemedida.

En trabajos más recientes, se tiene el de Francisco Valdez, que lleva por título "Sistema de riego para invernaderos con interfaz web utilizando Arduino Yun" de Quito, Ecuador (Valdez L. Francisco, 2022). El sistema se basa en la utilización de una tarjeta Arduino UNO con un Shield Dragino YUN, este componente está relacionado específicamente con el registro de los parámetros ambientales que afectan al invernadero, como la humedad, temperatura y la luz. Se presentan conceptos relacionados con los sistemas de riego, la plataforma Arduino, los sensores utilizados y los componentes del protocolo I2C. Además, se presenta el diseño de un sistema que combina sensores y un módulo INA3221 en un solo dispositivo. Asimismo, se presenta el diseño de un algoritmo que gestiona la recolección y organización de los datos recibidos del sistema sensor, el cual es evaluado en un ambiente controlado para observar cambios en los parámetros ambientales y verificar su funcionamiento. Como resultados, demostró que al utilizar

su metodología obtuvo un margen de error por debajo del 5%, tomando en cuenta el tipo de sensores a utilizar y las variables a medir.

2.1.2 Nacionales

En la investigación a nivel nacional se cuenta con el trabajo de Ortiz Aguilar y colaboradores (Ortiz-Aguilar et al., 2022), con su artículo llamado "Diseño IoT de invernadero para el control de variables mediante técnicas de inteligencia artificial", presentando un prototipo de modelo basado en Inteligencia Artificial e Internet de las Cosas para controlar el sistema de riego de un invernadero. El enfoque propuesto abarca diversas etapas, como el diseño, monitoreo, agrupación, clasificación y control. Se desarrolló un modelo a escala que permitió probar las técnicas de inteligencia artificial, y posteriormente se adaptó a la infraestructura del Instituto Tecnológico Superior de Purísima del Rincón (ITSPR), incorporando una base de datos e interfaces electrónicas. Los datos recopilados se almacenaron en una base de datos y se aplicó una clasificación no supervisada utilizando el algoritmo K-means. A partir de esto, se logró predecir los valores adecuados para mantener el invernadero en óptimas condiciones, siendo uno de los objetivos principales de este proyecto la optimización de recursos, como el agua.

Igualmente, Edgar Carrillo y colaboradores (Carrillo et al., 2022) realizaron su tesis llamada "Propuesta Técnica de un control en un invernadero industrial tipo multitunel y monitoreo de variables a través de un sistema IOT", proporcionando una visión general del cultivo protegido y los parámetros climáticos necesarios para el crecimiento de las fresas, analizando los antecedentes del cultivo de fresas y las tecnologías utilizadas en proyectos similares, enfocándose en la propuesta técnica, incluyendo la selección de instrumentos y equipos. Explicando la lógica utilizada en el controlador lógico programable (PLC) y el envío de datos mediante tecnologías IoT. En resumen, el proyecto buscó mejorar la producción de fresas en un

invernadero en Baja California mediante la automatización y el control de los parámetros ambientales. Se utilizan diferentes tecnologías, como sensores, PLC, software de programación y plataformas en la nube, para lograr este objetivo. Los resultados incluyen una producción más constante y estable, así como la posibilidad de monitorear y analizar los datos en tiempo real para la toma de decisiones. las temperaturas ya que solo introducen aire caliente o frío, mas no la extracción del mismo. En la parte de control la implementación del Raspberry facilitó el crecimiento de las hortalizas seleccionadas para la investigación.

Continuando con la investigación, se referencia el trabajo tesis llamada "Desarrollo de una Red de Sensores para el Monitoreo en Ambiente Web de Parámetros Físico-Químicos en Invernaderos de Plantas Ornamentales" del maestro Rodrigo D. (Delgadillo-Gaytan, 2019) donde realizó un sistema propuesto que se compone de módulos, incluyendo nodos de sensores para las mediciones y su transmisión mediante tecnología wifi y Xbee, y una estación de monitoreo encargada de recibir, procesar y mostrar la información, así como enviarla a la base de datos. La implementación de esta red de nodos demostró ser una solución eficaz para reducir los tiempos de producción y mantener un control adecuado de los parámetros ambientales en el invernadero. En esta tesis se centra en el desarrollo de una red de sensores para el monitoreo y control de parámetros en un invernadero, con el objetivo de mejorar la productividad de los procesos agrícolas. Mediante la implementación de esta red de nodos, se logra una disminución de los tiempos de producción y un control más preciso de los parámetros ambientales, lo que resulta en un aumento de la producción de plantas ornamentales. Se destaca la importancia de la tecnología de agricultura de precisión y se concluye que el sistema implementado proporciona un monitoreo más eficiente y constante del invernadero.

La implementación de sistemas avanzados para el monitoreo de invernaderos agrícolas, utilizando tecnología de vanguardia, resulta altamente beneficioso para

registrar y dar seguimiento a las variables ambientales con mayor precisión como se ve en los trabajos antes mencionados. Aunque la tecnología de redes y los teléfonos móviles han avanzado a nivel internacional, su utilización en el ámbito agrícola a nivel nacional es aún un área de mejora. Esto brinda una oportunidad para desarrollar sistemas más complejos y capacitados, con el objetivo de mejorar la competitividad en este campo.

2.2 Análisis comparativo de los trabajos relacionados.

Estos trabajos muestran el uso de diversas tecnologías y enfoques para mejorar la automatización, control y monitoreo de los parámetros ambientales en invernaderos agrícolas. Los resultados obtenidos incluyen la funcionalidad de los prototipos, la fiabilidad de las comunicaciones, la optimización de recursos, la producción más constante y el control preciso de los parámetros, mostrado en la Tabla 4. Análisis comparativo del estado del arte.

Tabla 4. Análisis comparativo del estado del arte.

Trabajo	Tecnologías	Objetivos	Resultados
(Jorge Arcenio, 2021)	Arduino UNO, Firebase, sensores económicos.	Monitorear datos climáticos en invernadero.	Prototipo funcional, vida útil estimada de Arduino y sensores, simplificación de la API web.
(Blanco Rico & Martínez Zarzuela, 2021)	LoRaWAN, Arduino, aplicación móvil.	Monitorización de variables ambientales en un colegio.	Sistema óptimo, comunicaciones inalámbricas de larga distancia, prototipo completo, fiabilidad demostrada

(Valdez L. Francisco, 2022)	Arduino UNO, Shield Dragino YUN, sensores.	Registro y control de parámetros ambientales en invernadero.	Metodología con margen de error <5%, cambios observados en los parámetros ambientales.
(Ortiz-Aguilar et al., 2022)	Inteligencia Artificial, IoT, base de datos.	Control de riego en invernadero mediante IA.	Modelo a escala, técnicas de IA aplicadas, optimización de recursos.
(Carrillo et al., 2022)	PLC, IoT, sensores, Cloud.	Control de riego en invernadero mediante IA.	Producción más constante, monitoreo en tiempo real, toma de decisiones informada.
(Delgadillo-Gaytan, 2019)	Wifi, Xbee, sensores, base de datos.	Monitoreo y control de parámetros en invernadero de plantas ornamentales.	Disminución de tiempos de producción, control preciso de parámetros, aumento de la producción.

Estos avances tienen el potencial de aumentar la productividad y eficiencia en la agricultura, brindando oportunidades para futuros desarrollos en este campo.

2.3 Propuesta de solución

En este apartado, se presenta la propuesta de solución para abordar el problema planteado, que consiste en desarrollar un sistema de monitoreo y control de variables ambientales en invernaderos utilizando Raspberry Pi 4 como dispositivo de adquisición de datos y la plataforma de servicios en la nube de Amazon Web Services (AWS) para el procesamiento, almacenamiento y visualización de los datos. El sistema se basará en las siguientes tecnologías de AWS: IoT Core, Lambda, DynamoDB, Timestream y Grafana.

2.3.1 Diagrama de arquitectura

A continuación, se muestra el diagrama de arquitectura del sistema propuesto (Figura 13):

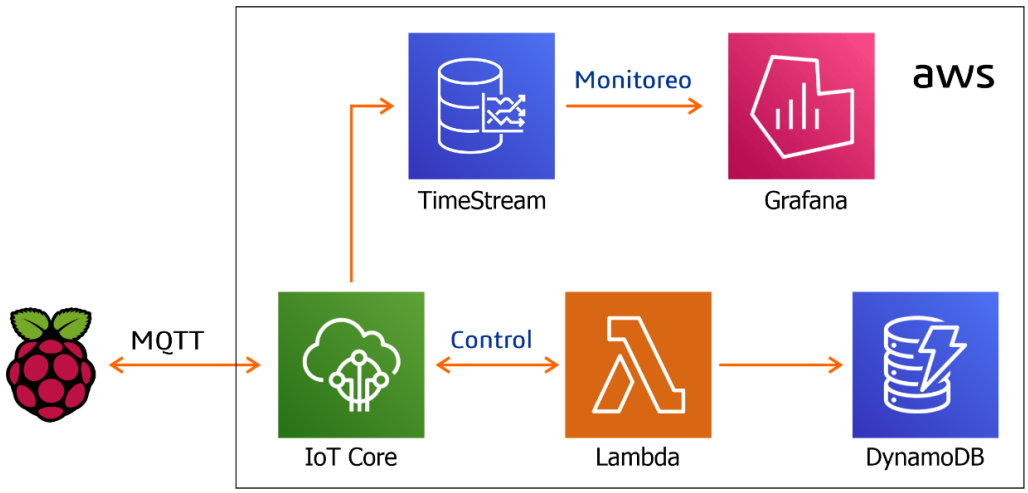


Figura 13. Diagrama de monitoreo y control desde Raspberry 4 con AWS.

En el diagrama, se puede observar la interacción entre los diferentes componentes del sistema, desde la adquisición de datos por parte de Raspberry Pi 4, el envío y recepción de datos a AWS IoT Core, el procesamiento de datos mediante funciones Lambda, el almacenamiento en DynamoDB y Timestream, hasta la visualización de los datos en Grafana.

2.3.2 Ventajas y beneficios

La implementación de este sistema de monitoreo y control de variables ambientales en invernaderos utilizando Raspberry Pi 4 y AWS ofrece las siguientes ventajas y beneficios:

- Monitoreo en tiempo real: Permite el monitoreo continuo de las variables ambientales clave en los invernaderos, proporcionando una visión precisa del entorno y la posibilidad de tomar acciones inmediatas en caso de condiciones anormales.
- Automatización y control: Permite la automatización de procesos y acciones basadas en los datos recopilados, como la activación de sistemas de riego, la regulación de la temperatura o la ventilación, optimizando el uso de recursos y mejorando la eficiencia.
- Almacenamiento y análisis de datos: Permite el almacenamiento estructurado y escalable de datos históricos y en tiempo real, lo que facilita el análisis posterior, la generación de informes y la toma de decisiones fundamentadas.
- Accesibilidad y visualización intuitiva: Proporciona una interfaz de usuario amigable a través de Grafana, que permite a los usuarios visualizar y analizar los datos de manera intuitiva, facilitando la comprensión del estado del invernadero y la identificación de tendencias.
- Escalabilidad y flexibilidad: El sistema puede adaptarse a diferentes tamaños y tipos de invernaderos, permitiendo la incorporación de más sensores y funcionalidades según las necesidades específicas de cada caso.

En resumen, la propuesta de solución consiste en desarrollar un sistema integral de monitoreo y control de variables ambientales en invernaderos utilizando Raspberry Pi 4 y la plataforma de servicios en la nube de AWS. El sistema permitirá la adquisición de datos, el procesamiento, almacenamiento y visualización de los mismos, ofreciendo ventajas significativas en términos de monitoreo en tiempo real, automatización, análisis de datos y accesibilidad.

CAPÍTULO III METODOLOGÍA Y DESARROLLO

La metodología seguida para la implementación del sistema de monitoreo y control de variables ambientales en invernaderos tipo túnel, utilizando la Raspberry Pi 4 y los servicios de AWS, se describe de la siguiente manera:

1. Selección de sensores: Se seleccionaron tres sensores específicos para medir diferentes variables ambientales en el invernadero. Los sensores elegidos son el DS18B20 para medir la temperatura, el HD-38 para medir la humedad y el MQ-135 para medir la calidad del aire.

2. Protocolos empleados para las señales: Se utilizó el protocolo PyFirmata para establecer la comunicación entre la Raspberry Pi 4 y los sensores. PyFirmata es una biblioteca de Python que permite controlar y leer datos de dispositivos electrónicos utilizando el protocolo Firmata. Además, para el sensor de temperatura DS18B20 se empleó el protocolo 1-wire, que es un protocolo de comunicación de bajo nivel diseñado para dispositivos con un solo cable de datos.

3. Generación de código fuente en Raspberry: Se generó el código fuente necesario para leer los datos de los sensores y poder interpretarlos. Se utilizaron bibliotecas relevantes para interactuar con los sensores seleccionados, es por eso que el código se desarrolló en Python utilizando programación paralela, así aprovechando las capacidades de programación de la Raspberry Pi 4.

4. Comunicación con AWS y Raspberry para el control y monitoreo: Se utilizó el conjunto de servicios de AWS, como IoT Core, Lambda, DynamoDB, Timestream y Grafana, para la comunicación y el almacenamiento de los datos del sistema. AWS IoT Core se utilizó para enviar los datos recopilados por la Raspberry Pi 4 a la nube, mientras que los otros servicios se utilizaron para almacenar, procesar y visualizar los datos en tiempo real.

5. Creación de Shields: Se diseñaron y crearon shields personalizados para la Raspberry Pi 4, con el objetivo de facilitar la conexión y protección de los sensores. Estos shields permitieron una fácil integración de los sensores seleccionados, garantizando una conexión segura y confiable.

6. Montaje del sistema en el invernadero: Una vez que se tuvieron los sensores, la Raspberry Pi 4 y los shields preparados, se procedió a montar el sistema en el invernadero. Los sensores se ubicaron estratégicamente para obtener mediciones representativas de las variables ambientales y se conectaron a través de los shields. A su vez se instaló el sistema de control de riego, utilizando como prueba un invernadero del Instituto tecnológico superior de Zacapoaxtla durante la estadía, con el cultivo de suculentas piedras de luna (Pachyphytum Oviferum).

7. Generación de códigos de set y reset: Se crearon códigos de set y reset para asegurar que el sistema se iniciara correctamente al encender la Raspberry Pi 4 y se pudiera reiniciar en caso de ser necesario. Estos códigos automatizaron el proceso de inicio y reseteo del sistema, evitando la necesidad de intervención manual.

Mediante la aplicación de esta metodología, se buscó garantizar un proceso de desarrollo eficiente y efectivo, culminando en un sistema robusto y funcional que permitiera el monitoreo y control de las variables ambientales en los invernaderos agrícolas.

3.1 Selección de sensores

En este estudio, se centra en analizar los beneficios y la importancia de implementar sistemas de sensores de temperatura, humedad del suelo y calidad del aire en un invernadero. Se investiga cómo estos sensores pueden ayudar a mejorar el control ambiental, aumentar la eficiencia del riego y optimizar el uso de recursos en general. Explorando diferentes tecnologías de sensores disponibles en el mercado actualmente y examinando sus características, ventajas y limitaciones. Se analizo los métodos utilizados para la recolección y el procesamiento de los datos generados por los mismos.

3.1.1 Variable de Temperatura

El primer campo a estudiar es la variable de temperatura tanto externa como interna del invernadero, requiriendo un dispositivo viable para este uso, realizando la comparación de varios sensores del mercado como se muestra en la Tabla 5. Comparación de sensores de medición de temperatura., resaltando sus rangos de medición, protocolo de comunicación, resistencia a condiciones climáticas y costo en pesos mexicanos.

Tabla 5. Comparación de sensores de medición de temperatura.

Sensor	Rango de medición	Protocolo de comunicación	Resistencia a condiciones climáticas	Costo aproximado (MXN)
DS18B20 (Xi 'an Gavin, 2008)	-55 °C a +125 °C	1-Wire	Resistente al agua y sellado	$100.00
LM35 (Texas Instruments, 1999)	-55 °C a +150 °C	Analógico	Sensible a entornos hostiles	$80.00
DHT22 (alldatasheet, n.d.)	-40 °C a +80 °C	1-Wire o I2C	Sensible a alta humedad	$120.00
PT100 (ZIEHL, n.d.)	-200 °C a +850 °C	Analógico	Requiere acondicionamiento especial	$200.00
TMP36 (Analog Devices inc., 2002)	-40 °C a +125 °C	Analógico	Sensible a cambios bruscos	$50.00

Destacando el sensor DS18B20 por varios motivos. En primer lugar, tiene un amplio rango de medición de temperatura, desde -55 °C hasta +125 °C, lo que permite su uso en una amplia variedad de aplicaciones. Además, utiliza un protocolo de comunicación 1-Wire, lo que simplifica su integración en el sistema de monitoreo con señales digitales, siendo compatible con la Tarjeta Raspberry, utilizando 5 para

este proyecto (4 internos y 1 externo). También es resistente al agua y está sellado, lo que le otorga mayor durabilidad y resistencia en condiciones climáticas adversas, lo cual es particularmente importante en un entorno de invernadero donde puede haber humedad y cambios de temperatura significativos. En cuanto al costo, el DS18B20 tiene un valor de $100 pesos mexicanos aproximadamente, lo que lo hace una opción accesible para su implementación.

3.1.2 Variable de Humedad del suelo

El segundo a estudiar es la variable de humedad del suelo dentro del invernadero para cada cultivo. Existen diversos sensores en el mercado que ofrecen diferentes características y funcionalidades. A continuación, se presenta una tabla comparativa de sensores de humedad del suelo (Tabla 6), resaltando sus rangos de medición, protocolo de comunicación, resistencia a condiciones climáticas y costo en pesos mexicanos.

Tabla 6. Comparación de sensores para la medición de humedad del suelo.

Sensor	Rango de medición	Protocolo de comunicación	Resistencia a condiciones climáticas	Costo aproximado (MXN)
HD-38 (OXDEA, 2020)	0% a 100%	Analógico	Resistente al agua y sellado	$250.00
TRH200 (Ascon Tecnologic s.r.l, 2020)	0% a 100%	RS485	Sensible a la inmersión en agua	$350.00

VH400 (Vegetronix, n.d.)	0% a 100%	Analógico	Sensible a la salinidad del suelo	$300.00
EC-5 (Decagon Devices, 2012)	0% a 100%	Digital	Sensible a la compactación del suelo	$400.00
SM200 (Delta-T Devices Ltd, 2006)	0% a 100%	SM2C/dHH2	Sensible a altas temperaturas	$450.00

En comparación con los otros sensores, el HD-38 destaca por su amplio rango de medición de humedad del suelo, que va desde 0% hasta 100%, esto proporciona una gran precisión y sensibilidad en la detección de la humedad. Además, utiliza una interfaz de comunicación analógica, lo que facilita su integración en el sistema de monitoreo. El HD-38 también es resistente al agua y está sellado, lo que lo hace adecuado para su uso en condiciones climáticas adversas y entornos húmedos, como los invernaderos. En cuanto al costo, tiene un valor de $250 pesos mexicanos, lo que lo convierte en una opción rentable y accesible para el monitoreo de la humedad del suelo. Aunque los otros sensores también tienen sus propias características y ventajas, como la capacidad de medir el contenido volumétrico de agua o la sensibilidad a diferentes condiciones del suelo, destacando como una opción confiable y precisa para el monitoreo de la humedad del suelo, el cual fue escogido para este proyecto utilizando 4 sensores HD-38.

3.1.3 Variable de Calidad del Aire

La calidad del aire afecta directamente la salud de las plantas, ya que altos niveles de contaminantes pueden comprometer su crecimiento y rendimiento. Por lo tanto, contar con un sensor preciso y confiable para monitorear la calidad del aire es fundamental para garantizar un ambiente adecuado y saludable para las plantas. A continuación, se presenta una tabla (Tabla 7) comparativa de diferentes sensores:

Tabla 7. Comparación de sensores para la medición de calidad del aire.

Sensor	medición	Protocolo de comunicación	Resistencia a condiciones climáticas	Costo aproximado (MXN)
MQ-135 (Winsen, 2015)	CO2, gases y compuestos orgánicos.	Analógico	Sensible a altas temperaturas.	$150.00
BME680 (BOSCH, 2017)	VOCs, temperatura, humedad, presión.	I2C, SPI	Sensible a la humedad y polvo.	$300.00
CCS811 (ams AG, 2016)	TVOCs y CO2.	I2C	Sensible a la humedad.	$200.00
PMS5003 (PLANTOWER, 2016)	Partículas PM2.5 y PM10	UART	Sensible a la humedad y polvo.	$250.00
SGP30 (Sension, 2017)	TVOCs y CO2 equivalentes	I2C	Sensible a la humedad.	$450.00

3.2 Protocolos empleados para las señales

En este proyecto, se utilizan varios protocolos de comunicación para facilitar la interpretación de señales provenientes de diferentes tipos de sensores y manejó de información hacia la nube. Estos protocolos se seleccionaron cuidadosamente en función de sus capacidades y compatibilidad con los dispositivos utilizados. Se emplea PyFirmata para la comunicación entre Raspberry Pi y Arduino y el protocolo 1-wire para la comunicación entre Raspberry Pi y sensores DS18B20. Estas metodologías permiten la adquisición de datos en tiempo real y la transferencia eficiente de los mismos a través de distintas plataformas. Explicando como fue el método empleado para cada protocolo.

3.2.1 1-wire

Se utiliza la configuración por default del protocolo en la tarjeta Raspberry, realizando el acondicionamiento de los sensores de temperatura digital, utilizando el mismo pin para la conexión en paralelo de cinco sensores DS18B20, como se muestra a continuación (Figura 14). Usando una resistencia de un kilo ohm entre el hilo de comunicación (Azul) de los sensores y la alimentación de 5 volts (Rojo).

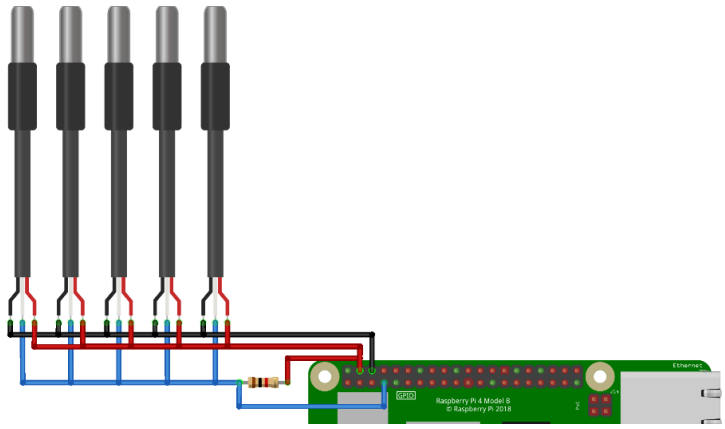


Figura 14. Conexión de sensores DS18B20 con Raspberry.

Estableciendo la conexión de los sensores se obtiene el id único de cada sensor para tener las lecturas independientes, ubicada en la dirección "/sys/bus/w1/devices/" en la Raspberry, como se muestra en la Figura 15.

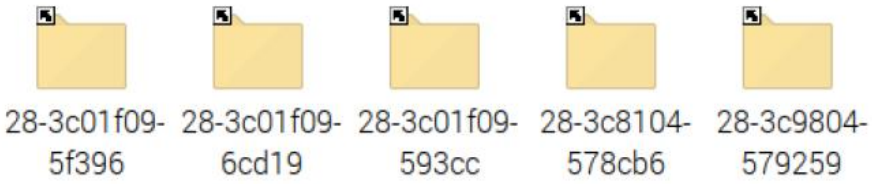


Figura 15. Carpeta con id único de cada sensor DS18B20.

Con este método de comunicación del protocolo 1-wire se establece los sensores de manera adecuada para la captura de temperaturas en el invernadero y exterior.

3.2.2 PyFirmata

Otro protocolo usado en este proyecto es PyFirmata que combina la Raspberry Pi y el Arduino para aprovechar sus capacidades complementarias. La Raspberry Pi, una computadora, se utiliza para tareas de procesamiento y control digital, mientras que el Arduino se emplea por su capacidad de leer señales analógicas. Se establece una comunicación serial entre ambos dispositivos para superar la limitación de la Raspberry Pi en la lectura de señales analógicas. Para facilitar la transferencia de datos y comandos entre la Raspberry Pi y el Arduino, se utiliza la biblioteca PyFirmata. Esta biblioteca permite enviar instrucciones desde la Raspberry Pi al Arduino para que este último pueda leer las señales analógicas de los sensores conectados. Los datos recopilados se transmiten nuevamente a la Raspberry Pi para su posterior procesamiento y análisis. Cabe mencionar que la Raspberry Pi no incluye de forma predeterminada la biblioteca PyFirmata, por lo que es necesario realizar un proceso de instalación específico en la Raspberry Pi para utilizarla y establecer una comunicación efectiva con el Arduino, siendo el siguiente:

Para garantizar un entorno estable y seguro en la Raspberry Pi, es fundamental realizar actualizaciones del sistema operativo y las bibliotecas utilizadas. Por este motivo, como parte del proceso de configuración, se llevó a cabo una búsqueda de actualizaciones previas utilizando comandos en la consola. Permitiendo mantener el sistema operativo y las bibliotecas al día, reduciendo así los riesgos de posibles errores, vulnerabilidades de seguridad y problemas de compatibilidad. La búsqueda de actualizaciones se realiza mediante comandos específicos que permiten verificar si existen nuevas versiones de los paquetes instalados en la Raspberry Pi. Estos comandos, ejecutados en la consola (Figura 16), se encargan de buscar y descargar las actualizaciones disponibles desde los repositorios oficiales. Una vez descargadas, las actualizaciones son instaladas en el sistema, garantizando así que se aprovechen las mejoras de rendimiento y funcionalidad que ofrecen.

```
rasp@raspberrypi:~ $ sudo apt-get update
rasp@raspberrypi:~ $ sudo apt-get upgrade
```

Figura 16, comandos para actualizar bibliotecas y programas en Raspberry.

La librería de PyFirmata se basa en el intérprete de Python, por lo tanto, es necesario contar con la instalación de la versión 3 del lenguaje. Para la instalación de Python 3 y pip en la Raspberry Pi se ejecutan los siguientes comandos en la terminal como se muestra en la Figura 17.

```
rasp@raspberrypi:~ $ install python3
rasp@raspberrypi:~ $ install python3-pip
```

Figura 17. instalación de Python3 y su gestor de paquetes, pip.

La instalación de Python 3 y pip es un paso crucial para garantizar el correcto funcionamiento de PyFirmata como una comunicación efectiva entre la Raspberry Pi y el Arduino.

- Terminando la instalación de los programas antes mencionados, se procede a instalar PyFirmata en tarjeta. Utilizando la terminal de la Raspberry Pi para ejecutar el siguiente comando en la terminal (Figura 18):

```
rasp@raspberrypi:~ $ sudo pip3 install pyfirmata
```

Figura 18. Instalación de PyFirmata

Terminando la instalación de la biblioteca PyFirmata en la Raspberry Pi, es necesario utilizar código que permita la recolección de datos analógicos desde el Arduino y su envío a la Raspberry mediante el protocolo Firmata. Para este proyecto, se utiliza uno de los códigos precargados en el IDE de Arduino. A continuación, se describen los pasos para realizar esta configuración:

Se conecta el Arduino al ordenador mediante un cable USB e iniciando IDE de Arduino. Una vez abierto el IDE de Arduino, se accede al menú "Archivo" en la parte superior de la ventana y se selecciona la opción "Ejemplos". Posteriormente, en el submenú "Ejemplos", se busca la carpeta "Firmata" y se abre el ejemplo llamado "StandardFirmata" (Figura 19). Este ejemplo proporciona un programa predefinido en el Arduino que establece la comunicación utilizando el protocolo Firmata.

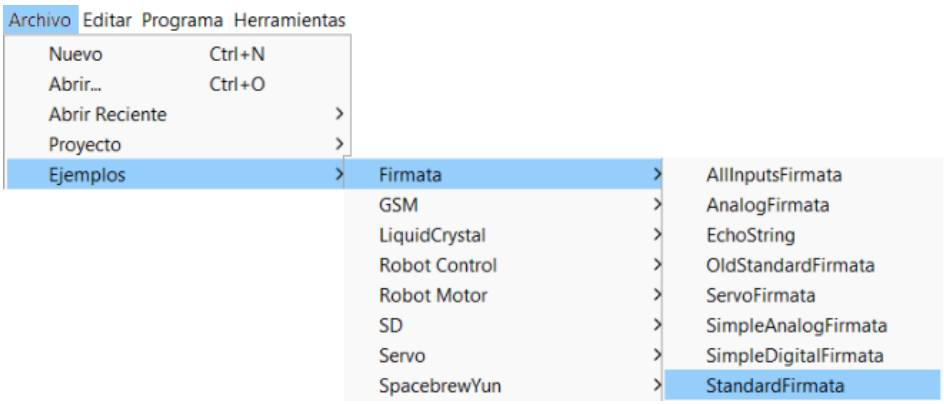


Figura 19. Ruta del Sketch Firmata.

Continuando la carga se secciona placa correspondiente al Arduino Nano. Para hacer esto, se accede al menú "Herramientas" en la parte superior de la ventana y se selecciona la opción "Placa", eligiendo la opción que corresponde a "Arduino Nano". Esta selección asegura que el IDE de Arduino compile y cargue el código correctamente para la placa específica que se está utilizando (Figura 20).

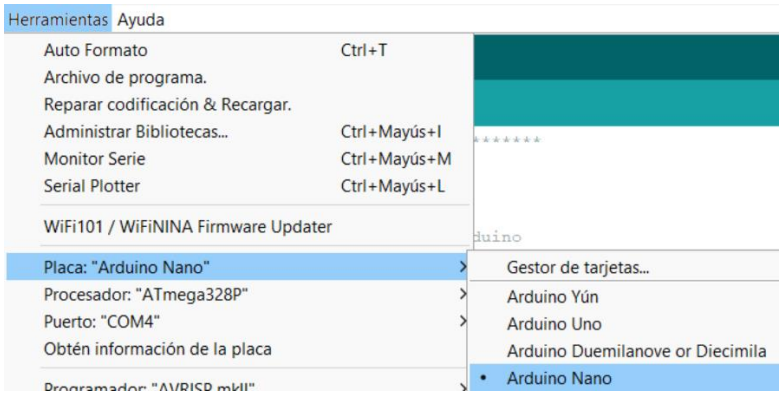


Figura 20. Selección de Arduino Nano.

Pasando con eso se selecciona el puerto serie al que está conectado el Arduino Nano. Se accede nuevamente al menú "Herramientas" y se selecciona la opción "Puerto". mostrando una lista de los puertos serie disponibles en el computador.

Seleccionando el puerto serie al que está conectado el Arduino Nano (Figura 21). La identificación del puerto puede variar dependiendo del sistema operativo utilizado.

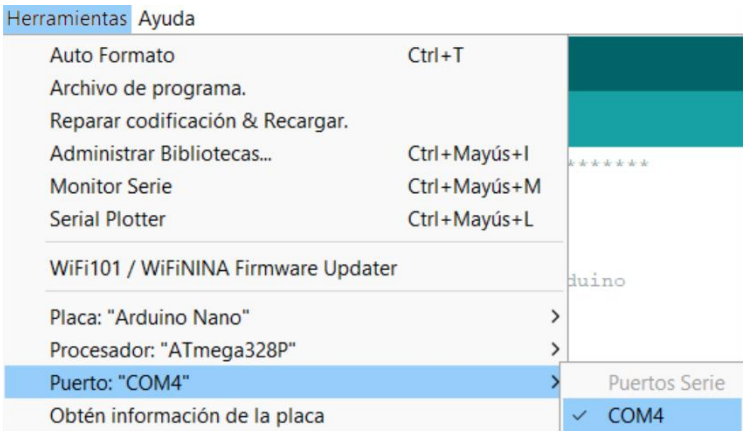


Figura 21. Selección de COM del Arduino Nano.

Por último, para cargar el sketch en la placa Arduino Nano, se hace clic en el botón "Subir" o "Cargar" ubicado en la parte superior de la ventana del IDE de Arduino. Al hacer clic en este botón, el código se compila y se carga en la placa Arduino Nano a través del puerto serie seleccionado previamente.

Estableciendo ya una conexión con el Arduino conectado a la Raspberry Pi mediante el puerto USB para controlar sus pines desde Python. Aprovechando la potencia de la Raspberry Pi y la capacidad de lectura analógica del Arduino en un entorno de desarrollo integrado en este proyecto como se observa en la Figura 22. Conexión entre Raspberry y Arduino con cuatro sensores HD-38 y un MQ-135.

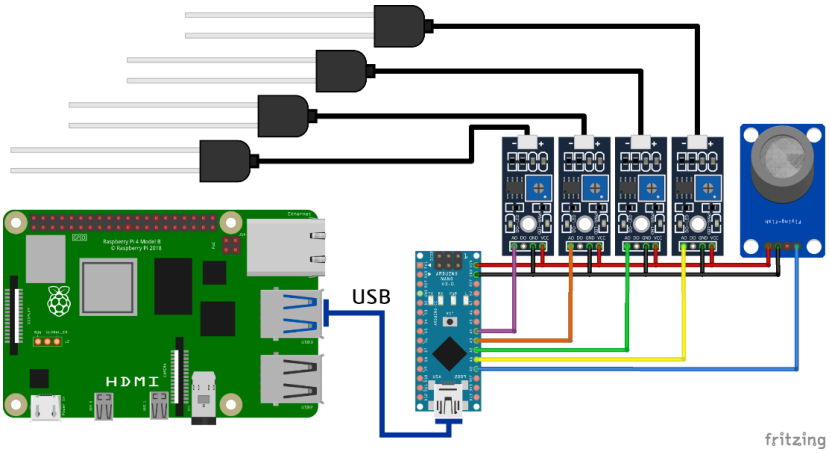


Figura 22. Conexión entre Raspberry y Arduino con cuatro sensores HD-38 y un MQ-135.

Explicando la Figura 22, la comunicación entre las tarjetas se establece por el puerto USB, los sensores se conectan en los pines analógicos que va desde el "A0" al "A4" del Arduino, para los señores de lecturas analogías.

Con el mismo protocolo de PyFirmata se emplean los pines de salida ya que desde Arduino su voltaje de salida es de cinco volts, permitiendo activar los relevadores que estarán accionando los actuadores cuando sean necesarios por el control. Utilizando cuatro pines digitales desde el "D4" al "D7" (Figura 23) adecuándolos a los relevadores.

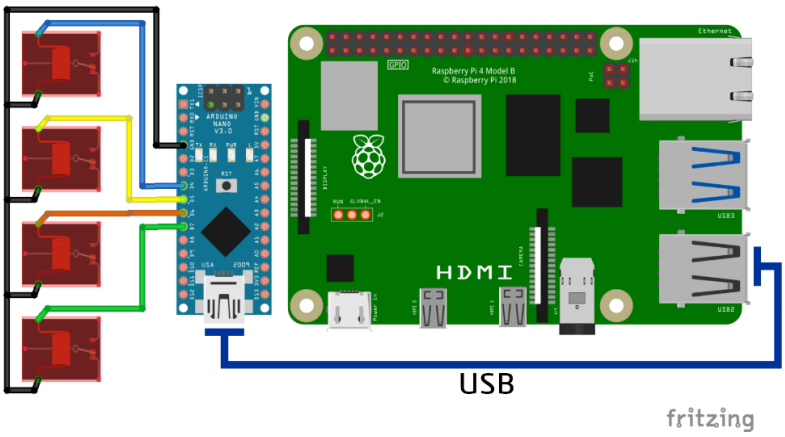


Figura 23. Conexión de relevadores con Arduino y Raspberry.

3.3 Generación de código fuente en Raspberry

Este código fue desarrollado para implementar un monitoreo y control de sensores en una placa Raspberry Pi utilizando el lenguaje de programación Python. En este código, se importan varias librerías, se configuran los pines GPIO y la conexión con un dispositivo Arduino. Además, se definen funciones para medir la temperatura, la humedad y controlar otros sensores conectados al Arduino. Se utiliza un bucle infinito para realizar las mediciones en intervalos regulares y se utiliza un ThreadPoolExecutor para ejecutar las funciones de medición en paralelo. Al final de cada ciclo de medición, se espera un período antes de realizar la siguiente ronda de mediciones. Explicando a continuación el código por partes:

3.3.1 Importación de librerías

En la sección de importación de librerías, se describen todas las librerías necesarias para el correcto funcionamiento del programa. A continuación, se explica brevemente cada una de las librerías utilizadas (Figura 24. Librerías utilizadas):

```
import time
import ssl
import json
import _thread
import RPi.GPIO as GPIO
import logging
import os
import glob
from pyfirmata import Arduino,util
import time
import datetime;
import calendar
```

Figura 24. Librerías utilizadas

- time: Proporciona funciones relacionadas con la gestión del tiempo.
- ssl: Permite la comunicación segura mediante el protocolo SSL.
- json: Permite trabajar con datos en formato JSON para el envió a AWS.
- _thread: Permite la ejecución de múltiples hilos de ejecución.
- RPi.GPIO: Facilita el acceso y la configuración de los pines GPIO de la Raspberry Pi.
- logging: Permite realizar registros (logs) en el programa.
- os: Proporciona funciones para interactuar con el sistema operativo.
- glob: Accede a realizar búsquedas y manipulaciones de rutas de archivos.
- pyfirmata: Permite la comunicación con Arduino nano a través del protocolo Firmata.
- datetime: Provee funciones para trabajar con fechas y horas.

Estas librerías son importadas para brindar funcionalidades necesarias en la implementación de la comunicación de la Raspberry Pi como el manejo del tiempo, la seguridad de la comunicación, el procesamiento de datos JSON, la ejecución de hilos, la interacción con los pines GPIO y el registro de eventos.

3.3.2 Lectura de temperatura

En esta parte del código se configuran los sensores de temperatura y lee los valores de cada uno de ellos. Proporciona una manera de acceder y procesar estos datos para su posterior uso en aplicaciones que requieran información sobre la temperatura ambiente. Explicándolo a continuación:

Se empezó por la configuración, estableciendo las rutas de los sensores de temperatura mediante el protocolo 1-Wire (Figura 25. Declaración del protocolo 1-Wire para temperaturas.). Cada sensor se identifica con una dirección única y se asigna una variable para almacenar su ruta en el sistema de archivos de la Raspberry

Pi, localizándose en el directorio "/sys/bus/w1/devices/" como se observa en la Figura 26.

```
os.system('modprobe w1-gpio')
os.system('modprobe w1-therm')
```

Figura 25. Declaración del protocolo 1-Wire para temperaturas.

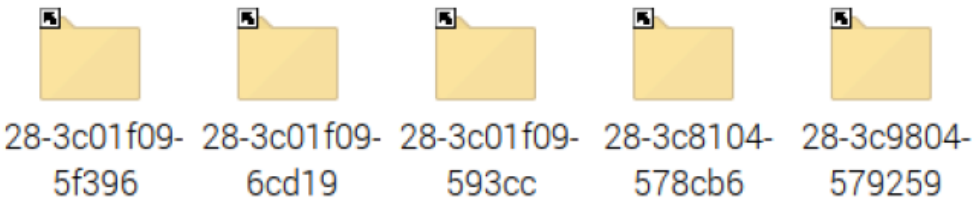


Figura 26. Id del forlder de cada sensor DS18B20.

Estas rutas de sensores son asignadas a las variables "Temp1", "Temp2", "Temp3", "Temp4" y "Temp_Ext", para poder acceder a los archivos que contienen los valores de temperatura de cada sensor (Figura 27).

```
Temp1='/sys/bus/w1/devices/28-3c01f096cd19'
Temp2='/sys/bus/w1/devices/28-3cfff095588b'
Temp3='/sys/bus/w1/devices/28-3c01f09593cc'
Temp4='/sys/bus/w1/devices/28-3c01f095f396'
Temp_Ext='/sys/bus/w1/devices/28-3c9804579259'
```

Figura 27. Dirección completa de cada sensor DS18B20.

Posteriormente, se procede a realizar la lectura de los sensores de temperatura, almacenándolos en una lista llamada "folderT". Se utiliza un bucle para recorrer cada sensor y obtener el valor correspondiente. En cada iteración del bucle, se abre el archivo de lectura específico del sensor en una lista secundaria llamada "dispositivo", se leé la primera línea que contiene el valor de la temperatura con la variable "fila"

y se almacena en la lista "leer_fila". Luego, el archivo se cierra para posterior usarlo en una función, mostrando el fragmento del código en la Figura 28.

```
folderT=[Temp1, Temp2, Temp3, Temp4, Temp_Ext]
dispositivo = []
leer_fila = []

for i in range(5):
    dispositivo.append(folderT[i]+'/w1_slave')
    fila = open(dispositivo[i], "r")
    leer_fila.append(fila.readline())
    fila.close()
```

Figura 28. Lectura de filas de los sensores.

Continuando con el código, se definió una función llamada "temperatura()" que se encarga de procesar las lecturas de temperatura de los sensores. Dentro de esta función, se realiza otro bucle para recorrer la lista de sensores y realizar las operaciones necesarias para obtener la temperatura real a partir de los valores leídos en el bucle anterior. Se busca la cadena de texto "t=" en la segunda línea del archivo de lectura para identificar el valor de temperatura. Luego se realiza la conversión y el cálculo necesario para obtener la temperatura en grados Celsius. Finalmente, el valor de temperatura se redondea a dos decimales y se agrega a la lista "temp" (Figura 29).

```
temp = []
def temperatura():
    for i in range(5):
        fila = open(dispositivo[i], "r")
        linea=fila.readlines()
        temAux = linea[1].find('t=')
        temp_raw = linea[1][temAux+2:]
        temp_c = float(temp_raw) / 1000.0
        temp.append(round(temp_c, 2))
        fila.close()
    print('Temperatura_full: ', temp)
    temp.clear()
```

Figura 29. Lectura de temperatura en los sensores.

3.3.3 Lectura de humedad del suelo

El propósito general de este fragmento de código es obtener información sobre la humedad utilizando sensores conectados a la placa Arduino nano que está anclada a la Raspberry por el protocolo PyFirmata y mostrar los valores medidos en la lista "Sensor". Esta información es utilizada para el monitoreo y control de la humedad en un entorno determinado dentro de un invernadero para un sistema de riego automático.

Lo primero fue establecer una conexión entre el programa y la placa Arduino. La ruta "/dev/ttyUSB0" especifica el puerto serie al que está conectada la placa. Al asignar esta conexión a la variable "arduino", se puede utilizar para enviar y recibir datos desde la placa en el resto del código (Figura 30).

```
arduino=Arduino("/dev/ttyUSB0")
```

Figura 30. Asignación del puerto serial para Arduino y Raspberry.

La línea "it=util.Iterator(arduino)" creo un objeto "Iterator" utilizando la biblioteca pyfirmata. Este objeto se utiliza para recibir actualizaciones de los pines de entrada analógicos y digitales de la placa Arduino. Al pasar la instancia "arduino" como argumento, se establece la conexión del objeto "Iterator" con la placa. Por último, "it.start()" inicia el objeto "Iterator", lo cual activa la actualización continua de los pines de entrada. Esto significa que los pines estarán listos para leer valores en tiempo real (Figura 31).

```
it=util.Iterator(arduino)
it.start()
```

Figura 31. Creación e inicio del iterador.

Esta funcionalidad es esencial para monitorear o interactuar con los cuatro sensores HD-38 y los relevadores conectados a la placa Arduino. Las siguientes líneas del código, se declaran las variables para la humedad mediante la sintaxis "arduno.get("a/d:0:i/0")", definiendo el tipo de señal analógica o digital, el pin asignado y si se usara de entrada o de salida. Basándose en este método de definen cuatro variables para la humedad, configurando los pines de entrada analógicas desde el pin "0" al "3" en la placa Arduino. Como se muestra en la Figura 32. Declaracion de variables para humedad.

```
Humedad1=arduino.get_pin("a:0:i")
Humedad2=arduino.get_pin("a:1:i")
Humedad3=arduino.get_pin("a:2:i")
Humedad4=arduino.get_pin("a:3:i")
```

Figura 32. Declaracion de variables para humedad.

Después de configurar los pines se alacenas las variables en una lista llamada "humedadS" para usarla posteriormente, se define una función llamada "medirH()" que se encarga de medir la humedad utilizando los sensores conectados a los pines analógicos. La función itera sobre los sensores de humedad (humedadS), lee los valores analógicos correspondientes y los convierte en un rango de 0 a 100 representando la humedad del suelo en cada sensor. Luego, se almacenan los valores en una lista llamada Sensor y se imprimen por pantalla como se mientras en la Figura 33. Función "medirH()".

```
humedadS=[Humedad1,Humedad2,Humedad3,Humedad4]
Sensor=[]
def medirH():
    for i in range (4):
        Sensor.append(round((humedadS[i].read()*(-100)+100),0))
    print (Sensor)
    Sensor.clear()
```

Figura 33. Función "medirH()".

3.3.4 Lectura de sensores extras

Esta parte de código se encarga de recolectar la lectura del sensor de calidad del aire y a su vez se deja adaptado para la conexión de otros 2 sensores digitales para una futura adaptación o expansión. Se uso de igual manera pines de Arduino por el protocolo PyFitmata usando el petodo get_pin(), definiendo tres variables "Aire", "Aux1" y "Aux2" que representan pines de entrada en Arduino (Figura 34. Declaración de variables para sensores extras.).

```
Aire=arduino.get_pin("a:4:i")
Aux1=arduino.get_pin("d:2:i")
Aux2=arduino.get_pin("d:3:i")
```

Figura 34. Declaración de variables para sensores extras.

Para poder contener estas variables, se crea una lista llamada "listAux", misma que se usa en la función "medirA()" que se define con el objetivo de medir los valores de los pines de entrada. Primero, se realiza una pausa de 0.5 segundos utilizando "time.sleep(0.5)" para asegurar una estabilidad en las mediciones. Luego, se imprime el contenido de "listAux", que son los valores actuales de los pines de entrada. Finalmente, la lista se limpia con el método "clear()", como se muestra en el código de la Figura 35.

```
listAux=[Aire,Aux1,Aux2]
def medirA():
    time.sleep(0.5)
    print (listAux)
    listAux.clear()
```

Figura 35. Función "MedirA".

Completando así el método de recolección de datos en los sensores.

3.3.5 Inicio del programa en paralelo

En esta sección del código inicia el bucle principal permite la ejecución continua del programa, donde se obtiene el tiempo actual, se envían tareas a los hilos para medir la humedad, otros sensores y la temperatura, y se espera un tiempo determinado antes de iniciar una nueva iteración. Esto permite realizar mediciones periódicas y controlar los diferentes componentes del sistema donde se explica a continuación (Figura 36).

```
while True:
    tiempo_act = int(time.mktime(datetime.datetime.utcnow().timetuple()))
    executor = ThreadPoolExecutor(max_workers=3)
    executor.submit(medirH)
    executor.submit(medirA)
    executor.submit(temperatura)
    time.sleep(30)
```

Figura 36. Bucle de inicio el programa

Obtención del tiempo actual en formato Unix:

Se utiliza la función "time.mktime(datetime.datetime.utcnow().timetuple())" para obtener el tiempo actual en formato Unix. Esto se realiza para tener una referencia de tiempo que se utiliza posteriormente en el programa.

Creación del objeto ThreadPoolExecutor:

Se crea un objeto "ThreadPoolExecutor" con la línea "executor = ThreadPoolExecutor(max_workers=3)". Este objeto permite ejecutar tareas en paralelo utilizando múltiples hilos de ejecución. En este caso, se especifica que el número máximo de hilos serán tres, lo que significa que se pueden ejecutar hasta tres tareas simultáneamente.

Envío de tareas a los hilos:

Se utilizan los métodos "executor.submit()" para enviar las tareas a los hilos. En este caso, se envían tres tareas:

- "executor.submit(medirH)": Esta tarea llama a la función medirH (), que mide la humedad de los sensores y guarda los valores en una lista.
- "executor.submit(medirA)": Esta tarea realiza la medición del sensor del aire y el acomplo de otros dos sensores ().
- "executor.submit(temperatura)": Esta tarea llama a la función temperatura (), que lee los sensores de temperatura y guarda los valores en una lista.

Al enviar estas tareas a los hilos, se permite que se ejecuten en paralelo y se optimice el tiempo de ejecución.

Espera de 30 segundos:

Después de enviar las tareas a los hilos, se utiliza "time.sleep(30)" para pausar la ejecución del programa durante 30 segundos. Esto significa que el programa esperará ese tiempo antes de comenzar una nueva iteración del bucle principal con la finalidad de saturar él envió a AWS IoT Core.

Una vez creado el código se hace la prueba para verificar el correcto funcionamiento de recolección de información (Figura 37. Lectura de sensores.).

```
temperaturas: [18.33, 18.46, 18.67, 18.35, 16.34]
HumedadS: [32, 43, 26, 30]
SensoresEx: [0.0061, , null, null]
```

Figura 37. Lectura de sensores.

3.4 Comunicación con AWS para el control y monitoreo

La comunicación segura y escalable entre dispositivos IoT y la nube es esencial para el funcionamiento exitoso de un sistema de IoT. En este contexto, AWS IoT Core se destaca como un servicio en la nube de AWS especialmente diseñado para proporcionar una plataforma confiable para la conexión y administración de dispositivos IoT. En esta sección, vamos a describir el proceso de integración de la Raspberry Pi con AWS IoT Core utilizando el control mediante funciones Lambda y almacenamiento de datos en DynamoDB, además del monitoreo a través de TimeStream y Grafana. Este enfoque permitirá enviar los datos recolectados por los sensores a través de la nube de AWS para su posterior análisis y procesamiento. A continuación, se detalla el método utilizado.

3.4.1 Configuración de AWS IoT Core con Raspberry Pi

Para lograr configurar AWS IoT Core y establecer una conexión segura entre la Raspberry Pi y la nube de AWS, se inicia sesión en AWS Management Console, se siguieron los siguientes pasos:

Figura 38. Creación de objeto para la Raspberry.

En la Figura 38, se puede observar la creación de un Thing (Dispositivo) en la consola de AWS IoT Core para representar una Raspberry Pi. Este paso es crucial para

establecer una conexión segura y eficiente entre la Raspberry Pi y los servicios en la nube de AWS. Al crear el Thing, se le otorga un nombre único, en este caso, "raspberry", que permitirá identificar y diferenciar este dispositivo de otros en el entorno de IoT. Además, se asigna un identificador único de tipo de objeto denominado "pi", que ayuda a clasificar y organizar los dispositivos de acuerdo con sus características y funcionalidades. La creación de este objeto es fundamental, ya que marca el inicio del registro y la configuración del dispositivo en la plataforma de AWS IoT Core. A medida que se completa el proceso.

Continuando, se procedió a configurar los certificados y claves necesarios para garantizar la autenticación y encriptación de la comunicación entre la Raspberry Pi y AWS IoT Core. Estos certificados, que son únicos y se generan específicamente para esta conexión, desempeñan un papel fundamental en la creación de una conexión segura. Para llevar a cabo esta configuración, se accedió al menú de seguridad dentro de la plataforma de AWS IoT Core. Dentro del apartado de certificados en el menú de seguridad, se siguió un proceso cuidadoso para generar los certificados y claves correspondientes (Figura 39).

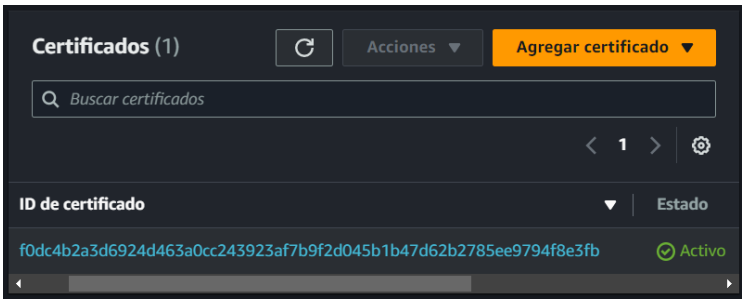


Figura 39. Creación de certificados y claves.

Estos elementos son esenciales para establecer una comunicación segura y confiable entre la Raspberry Pi y la nube de AWS. Al configurar los certificados y claves necesarios, se logró establecer una capa adicional de seguridad en la comunicación

entre la Raspberry Pi y AWS IoT Core. La autenticación y encriptación proporcionadas por estos certificados garantizan que solo los dispositivos autorizados puedan acceder y comunicarse con la nube de AWS, protegiendo así los datos y la integridad de la conexión.

Ya teniendo creado el Thing (dispositivo) en AWS IoT Core, es necesario asociar los certificados y claves correspondientes al dispositivo. Para asociar los certificados al Thing, se deben seguir ciertos pasos cargando el certificado en AWS IoT Core creado anteriormente. Esto implica proporcionar el certificado y la clave pública al servicio. AWS IoT Core verifica la validez del certificado y lo registra en su sistema. Después de registrar el certificado, se asocia al Thing correspondiente. Esta asociación vincula el certificado y sus claves al dispositivo específico en AWS IoT Core. De esta manera, cuando el dispositivo se comunica con AWS IoT Core, el servicio puede utilizar el certificado y las claves para autenticar y validar la identidad del dispositivo (Figura 40).

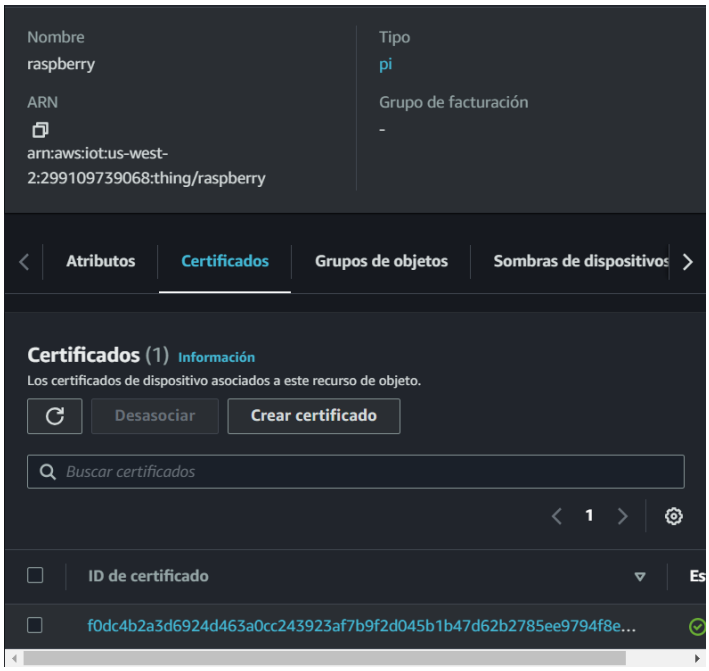


Figura 40. Asociación del Objeto y los certificados.

Una vez que el certificado y la clave privada están asociados con el Thing registrado, se pueden crear políticas de acceso en AWS IoT Core. Estas políticas determinan los permisos y restricciones para el dispositivo en la nube de AWS. Las políticas de acceso definen qué acciones puede realizar el Thing, como enviar y recibir datos, suscribirse a temas MQTT (Message Queuing Telemetry Transport), publicar mensajes, entre otras operaciones. Estas políticas son esenciales para establecer una capa adicional de seguridad y control en la comunicación entre el dispositivo y la nube (Figura 41).

Figura 41. Creación de la política.

los certificados se asocian al Thing registrado en AWS IoT Core para establecer una comunicación segura. Las claves criptográficas y los certificados X.509 permiten autenticar y cifrar la comunicación entre el dispositivo y la nube, mientras que las políticas de acceso determinan los permisos y restricciones para el Thing en AWS IoT Core.

Una vez que se ha configurado la conexión entre la Raspberry Pi y AWS IoT Core, se establecieron diferentes suscripciones para recibir datos específicos. Al suscribirse a este tema, la Raspberry Pi recibirá los datos en un formato específico, como JSON, que luego se puede procesar y utilizar según sea necesario. Creando tres suscripciones: "raspi/temps", "raspi/humedad" y "raspi/extra" (Figura 42).

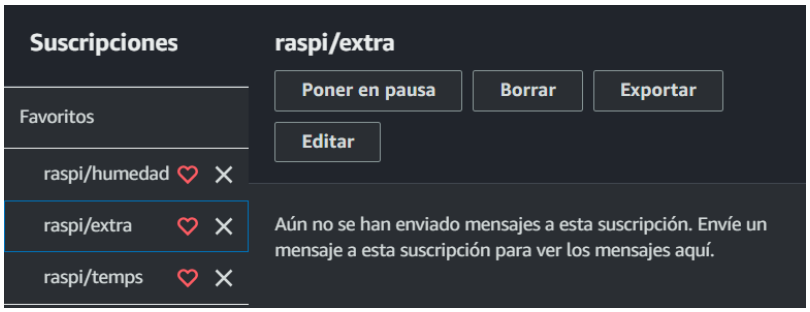


Figura 42. Subscripciones desde AWS IoT Core.

La suscripción "raspi/temps" permite a la Raspberry Pi recibir datos relacionados con la temperatura. Siendo útil para monitorear y controlar la temperatura en tiempo real dentro y fuera del invernadero. La suscripción "raspi/humedad" está diseñada para recibir datos de humedad del suelo. Además de las suscripciones mencionadas anteriormente, se cuenta con una suscripción extra llamada "raspi/extra", esta suscripción se usa para los sensores extras en el sistema, contando ahí con el sensor de calidad del aire y los dos espacios disponibles para sensores digitales. Teniendo la comunicación establecida lista para la Raspberry con IoT Core, se debe de agregar unas líneas de código en el programa de la Raspberry para poder tener el enlace explicándolo en el siguiente punto.

3.4.2 Anexo de código en Raspberry con IoT Core

Se definiendo una función de callback llamada "on_connect". Esta función se ejecuta cuando se logra establecer una conexión exitosa entre la Raspberry y AWS IoT Core. Es decir, cuando ambos dispositivos se comunican de manera satisfactoria. Para ello se crea un objeto cliente MQTT utilizando la biblioteca Paho MQTT en Python. Este cliente MQTT se configuro con certificados de seguridad (creados anteriormente), mismos que se descargaron y se guardaron en la misma carpeta del programa, este utiliza una versión específica del protocolo TLS para garantizar la seguridad de la conexión. Finalmente, se conecta el cliente MQTT al endpoint de AWS IoT Core que se ha especificado. Se establece un límite de tiempo de 60 segundos para establecer

la conexión. Si no se logra establecer la conexión dentro de este período de tiempo, se considera que la conexión ha fallado. En la Figura 43, se ilustra este proceso de conexión MQTT segura entre la Raspberry y AWS IoT Core.

```
import paho.mqtt.client as mqtt

def on_connect(client, userdata, flags, rc):
    print("Connected with result code "+str(rc))
client = mqtt.Client()
client.on_connect = on_connect
client.tls_set(ca_certs='/home/rasp/aws/rootCA.pem',
               certfile='/home/rasp/aws/certificate.pem.crt',
               keyfile='/home/rasp/aws/llave.pem.key>
client.tls_insecure_set(True)
client.connect("a231wyu012dljk-ats.iot.us-west-2.amazonaws.com", 8883, 60)
```

Figura 43. Fragmento de código para lograr el enlace Raspberry-AWS.

Contando ya con el enlace, se agregaron líneas de código en las diferentes funciones para su publicación respectivamente. En la función "temperaturas()", se agrega lo siguiente que publica un mensaje en el tópico "raspi/temps". El mensaje contiene los siguientes datos: "idDevice" (identificador del dispositivo), "tiempo_act" (tiempo actual), "T_int1", "T_int2", "T_int3", "T_int4 (temperaturas internas), "T_intF" (temperatura interna promedio), y "T_Ext" (temperatura externa). Estos datos se envían en formato JSON y se establece un nivel de calidad de servicio (QoS) de 0, sin retención del mensaje (Figura 44).

```
client.publish("raspi/temps",payload=json.dumps({'idDevice': idDevice,
                                                 "tiempo_act": tiempo_act,
                                                 'T_int1':round(temp[0]+0.01,2),
                                                 'T_int2':round(temp[1]+0.01,2),
                                                 'T_int3':round(temp[2]+0.01,2),
                                                 'T_int4':round(temp[3]+0.01,2),
                                                 'T_intF':(round(((temp[0]+temp[1]+temp[2]+temp[3])/4),2)),
                                                 'T_Ext':round(temp[4]+0.01,2)}), qos=0, retain=False)
```

Figura 44. Publicación en "raspi/temps".

En la función "medirH()", se publica un mensaje en el tópico "raspi/humedad". El mensaje incluye los siguientes datos: "idDevice" (identificador del dispositivo), "tiempo_act" (tiempo actual), "H1", "H2", "H3", "H4" (Sensores de humedad). Al igual que en el caso anterior, los datos se envían en formato JSON y se utiliza un nivel de calidad de servicio (QoS) de 0, sin retención del mensaje (), Figura 45.

```
client.publish("raspi/humedad", payload=json.dumps({'idDevice': idDevice,
                                                    "tiempo_act": tiempo_act,
                                                    'H1':Sensor[0],
                                                    'H2':Sensor[1],
                                                    'H3':Sensor[2],
                                                    'H4':Sensor[3]}), qos=0, retain=False)
```

Figura 45. Publicación en "raspi/humedad".

En la función "medirA()", se realiza una publicación en el tópico "raspi/extra". El mensaje contiene los siguientes datos: "idDevice" (identificador del dispositivo), "tiempo_act" (tiempo actual), "Aire" (lectura de sensor de aire), "Aux1" y "Aux2" (Espacios disponibles). Al igual que en los casos anteriores, los datos se envían en formato JSON con un nivel de calidad de servicio (QoS) de 0, sin retención del mensaje (), Figura 46.

```
client.publish("raspi/extra", payload=json.dumps({'idDevice': idDevice,
                                                  "tiempo_act": tiempo_act,
                                                  'Aire':listAux[0].read(),
                                                  'Aux1':listAux[1].read(),
                                                  'Aux2':listAux[2].read()}), qos=0, retain=False)
```

Figura 46.Publicación en "raspi/extra"

Una vez que se agregaron las líneas para la suscripción de envió se realiza la prueba en cada una de las suscripciones para ver que el envío de datos es el correcto. Mostrando la Figura 47 para la función de "temperatura", la Figura 48 para la función de "medirH" y la Figura 49 para la función de "medirA".

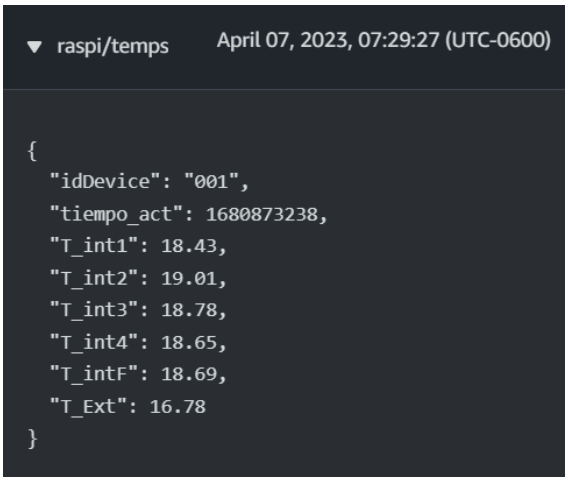


Figura 47. Envió de datos a "raspi/temps".

Figura 48. Envió de datos a "raspi/humedad".

Figura 49. Envió de datos a "raspi/extra".

Se crea un código adicional llamado "aws-rasp.py" que se encarga de enviar las instrucciones desde IoT Core para activar los relevadores. Este código tiene una estructura similar a la conexión MQTT a AWS, como se muestra en la Figura 43, Además, se añade la función "on_connect()" que se muestra en la Figura 50.

```
def on_connect(client, userdata, flags, rc):
    print("Conectado con resultado: " + str(rc))
    client.subscribe("raspi/comand")
```

Figura 50. Envió de datos de AWS a Raspberry.

Para identificar los pines de los relevadores directamente con Arduino, se crearon 7 variables, estas variables se utilizarán junto con la metodología "arduino.digital[(número del relevador)].write(0/1)". Mediante esta metodología, se envía una señal de 0 o 1 al pin correspondiente al número asignado (Figura 51).

```
Relay1=10
Relay2=9
Relay3=8
Relay4=7
Relay5=6
Relay6=5
Relay7=4
```

Figura 51. Declaración de variables a relevadores.

Continuando con el código se utilizó otra función “on_message” que es la encargada de verificar que mensaje se recibió desde la suscripción “raspi/comand” (creada como las anteriores) como se muestra en la Figura 52.

```
def on_message(client, userdata, msg):
    if msg.payload.decode() == "Relay1_on":
        a=1.1
        print (a)
        arduino.digital[Relay1].write(1)
```

Figura 52. Declaración de la función "on_menssage"

Es esta función se crearon múltiples "if" donde evalúa cual es el mensaje recibido, estos mensajes tiene la estructura de la activación o desactivación de cada relevador, con la frase "Relay(numero)_on/off" e imprimiendo en consola la variable "a" que tiene la estructura del número del relevador y si se activó o se desactivo (Figura 51. Declaración de variables a relevadores.). Mostrando más a detalle todos los "if" en la siguiente Tabla 8. Condiciones para cada mensaje recibido desde "raspi/comand".

Tabla 8. Condiciones para cada mensaje recibido desde "raspi/comand"

if	Mensaje recibido	Acción	Valor de "a"
1	Relay1_on	Arduino.digital[Relay1].write(1)	1.1
2	Relay1_off	Arduino.digital[Relay1].write(0)	1.0
3	Relay2_on	Arduino.digital[Relay2].write(1)	2.1
4	Relay2_off	Arduino.digital[Relay2].write(0)	2.0
5	Relay3_on	Arduino.digital[Relay3].write(1)	3.1
6	Relay3_off	Arduino.digital[Relay3].write(0)	3.0
7	Relay4_on	Arduino.digital[Relay4].write(1)	4.1
8	Relay4_off	Arduino.digital[Relay4].write(0)	4.0

9	Relay5_on	Arduino.digital[Relay5].write(1)	5.1
10	Relay5_off	Arduino.digital[Relay5].write(0)	5.0
11	Relay6_on	Arduino.digital[Relay6].write(1)	6.1
12	Relay6_off	Arduino.digital[Relay6].write(0)	6.0
13	Relay7_on	Arduino.digital[Relay7].write(1)	7.1
14	Relay7_off	Arduino.digital[Relay7].write(0)	7.0

Por último, se establecen las funciones de callback para la conexión y recepción de mensajes en el cliente MQTT. La función "on_connect" se ejecuta cuando el cliente se conecta con éxito al servidor MQTT, mientras que la función "on_message" se ejecuta al recibir un mensaje MQTT. Estas funciones realizan acciones específicas según el contenido del mensaje. Finalmente, el cliente entra en un bucle principal (client.loop_forever()) que permite mantener la conexión y recibir mensajes de manera continua, asegurando que el programa esté siempre disponible para procesar los mensajes entrantes (Figura 53. Funciones de callback para aws-rasp.py.).

```
client.on_connect = on_connect
client.on_message = on_message

client.loop_forever()
```

Figura 53. Funciones de callback para aws-rasp.py.

3.4.3 Control Con AWS lambda y DynamoDB

En este apartado de la metodología aprovechamos las herramientas de AWS "Lambda" y "DynamoDB", para un control efectivo y una recolección de datos adecuada. Mediante la implementación de AWS Lambda, se ejecuta el código necesario en respuesta a eventos específicos, mientras que DynamoDB nos proporciona una solución escalable y de alto rendimiento para almacenar y recuperar los datos recopilados. Gracias a estas herramientas, se crea un "data set" en DynamoDB que nos permitirá realizar análisis y obtener información valiosa para futuras investigaciones.

Comenzando con la configuración desde la consola de AWS Lambda, se crearon tres funciones utilizando el lenguaje de programación Python 3.9 para realizar diversas tareas. Estas funciones se llaman "TempsData", "HumedadData" y "ExtraData". Cada una de estas funciones desempeña un papel crucial en el proceso de recopilación y análisis de datos (Figura 54. Funciones creadas en AWS Lambda.).

Nombre de la función	Descripción	Tipo de paquete	Tiempo de ejecución
TempsData	-	Zip	Python 3.9
HumedadData	-	Zip	Python 3.9
ExtraData	-	Zip	Python 3.9

Figura 54. Funciones creadas en AWS Lambda.

La función "TempsData" se encarga de recopilar y procesar datos relacionados con la temperatura. Por otro lado, la función "HumedadData" está diseñada específicamente para recopilar y analizar datos relacionados con la humedad del suelo en cada sensor HD-38. Por último, la función "ExtraData" tiene un propósito

más amplio y versátil, encargándose de recopilar el sensor de calidad del aire y extendiéndose para más sensores si el sistema llega a crecer.

Pasando a la consola de IoT Core, se configuran reglas de enrutamiento de mensajes basadas en las suscripciones "raspi/temps", "raspi/humedad" y "raspi/extra". Estas reglas son esenciales para procesar y direccionar los mensajes recibidos desde los dispositivos conectados. A continuación, se detalla el proceso que se llevó para la configuración de las reglas en la consola de IoT Core. Lo primero en la sección de "Enrutamiento de mensajes", se encuentra la categoría de "reglas" donde se crearon tres reglas distintas para cada suscripción. Estas reglas se configuran de forma independiente y se denominan "TempRule", "HumedadRule" y "ExtraRule". Cada una de estas reglas requiere una configuración individual que incluye una instrucción SQL para filtrar y enrutar de manera flexible y avanzada los mensajes MQTT recibidos. Esta instrucción se utiliza para procesar y enviar los datos a AWS Lambda. La instrucción SQL utilizada es la siguiente: "SELECT * FROM <Topic Filter>". En esta instrucción, el asterisco "*" indica todos los valores recibidos en la suscripción, y "<Topic Filter>" representa la suscripción específica. De este modo, se configuraron las tres reglas de la siguiente manera como se presenta en la Figura 55, Figura 56 y Figura 57.

```
SELECT * FROM 'raspi/temps'
```

Figura 55. Instrucción SQL para TempsRule.

```
SELECT * FROM 'raspi/humedad'
```

Figura 56. Instrucción SQL para HumedadRule.

```
SELECT * FROM 'raspi/exras'
```

Figura 57. Instrucción SQL para ExtraRule.

Continuando con el proceso de configuración de reglas, se llevó a cabo una declaración de acciones específicas con AWS Lambda. En esta etapa, se realizó la selección cuidadosa de las funciones creadas previamente y se las vinculó con sus respectivas reglas. Para la regla denominada "TempRule", se asignó la función "TempsData" como acción correspondiente. De manera similar, para la regla "HumedadRule", se estableció la función "HumedadData" como acción asociada. Por último, para la regla adicional denominada "ExtraRule", se enlazó la función "ExtraData" como acción correspondiente. Esta configuración aseguro que cada regla esté correctamente asociada con la función específica requerida para el procesamiento de los datos capturados y enviados como se muestra en: Figura 58, Figura 59 y Figura 60.

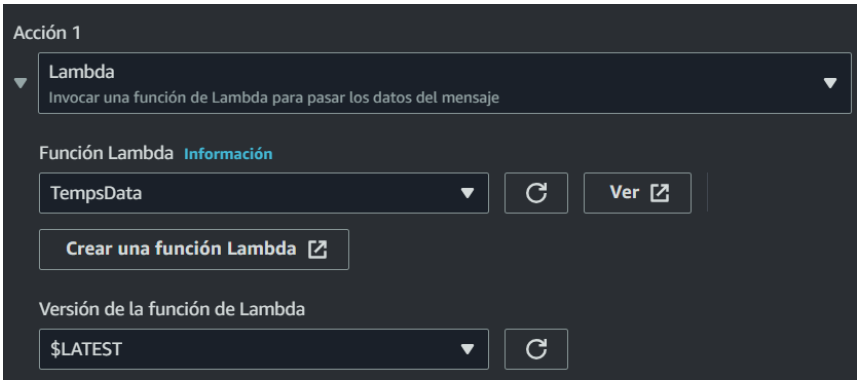


Figura 58. Acción de regla con TempsRule.

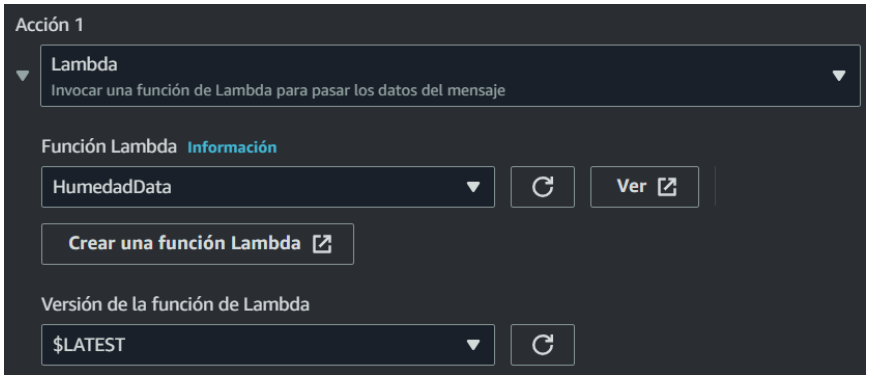


Figura 59. Acción de regla con HumedadRule.

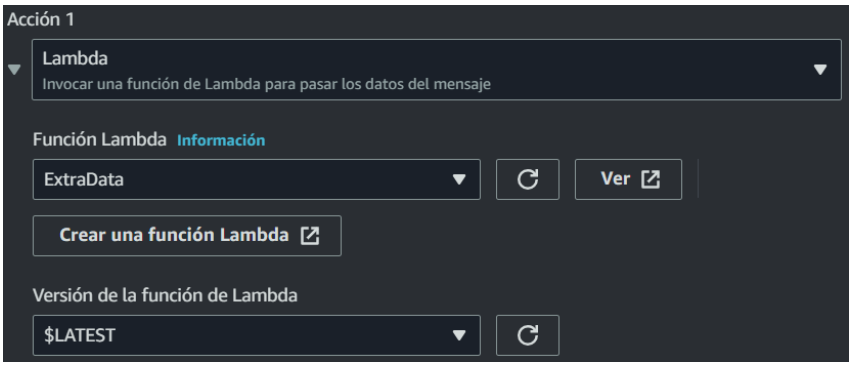


Figura 60. Acción de regla con ExtraRule.

Una vez finalizada la configuración de las reglas para el direccionamiento de mensajes hacia las funciones Lambda, se procede con el control del proceso mediante la generación de código en cada función y el envío de datos a DynamoDB. Este paso crucial asegura la correcta ejecución de las funciones, garantizando la captura y procesamiento de los datos de manera eficiente. Al completar este flujo de trabajo, se establece un sistema integrado y efectivo que permite la gestión y almacenamiento de la información de manera óptima en DynamoDB.

Siguiendo la metodología desarrollada, se generaron tres tablas en AWS DynamoDB con el propósito de almacenar y gestionar los datos generados por las funciones implementadas en Lambda. Estas tablas están vinculadas a las funciones correspondientes, facilitando así un flujo eficiente de información. La tabla "Rasp_Temperaturas" se conecta a través de la función "TempsData", mientras que la tabla "Rasp_Humedad" se enlaza con "HumedadData", y la tabla "Rasp_Extra" se conecta con "ExtraData". En cada caso, al crear estas tablas en DynamoDB, se asigna un nombre específico, como "Rasp_Temperaturas", donde se estableció una clave de partición denominada "tiempo_act", que se utiliza para organizar y acceder a los datos relacionados con la temperatura. Este mismo proceso se repite para las demás tablas, donde también se utiliza "tiempo_act" como clave de partición en

todas ellas como se muestra en la Figura 61. Creación de Tabla "Rasp_Temperaturas". en el apartado de Tablas / crear tabla.

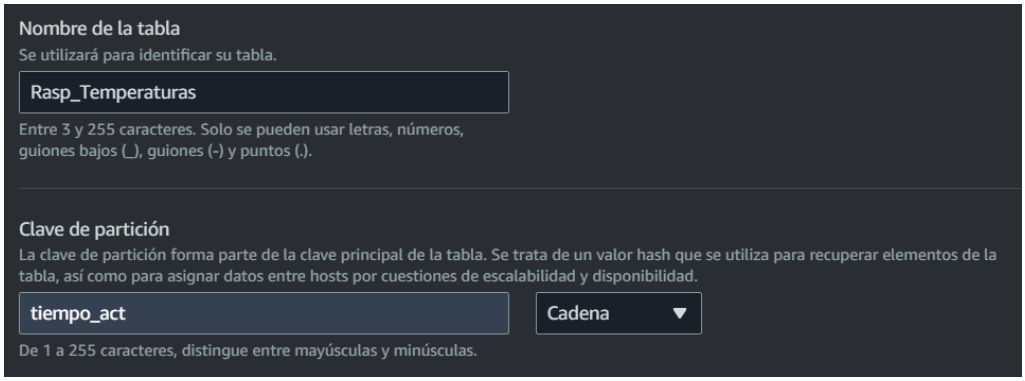


Figura 61. Creación de Tabla "Rasp_Temperaturas".

Continuando con la creación de las tres tablas, se utilizó la configuración predeterminada de DynamoDB, como se ilustra en la Figura 62.

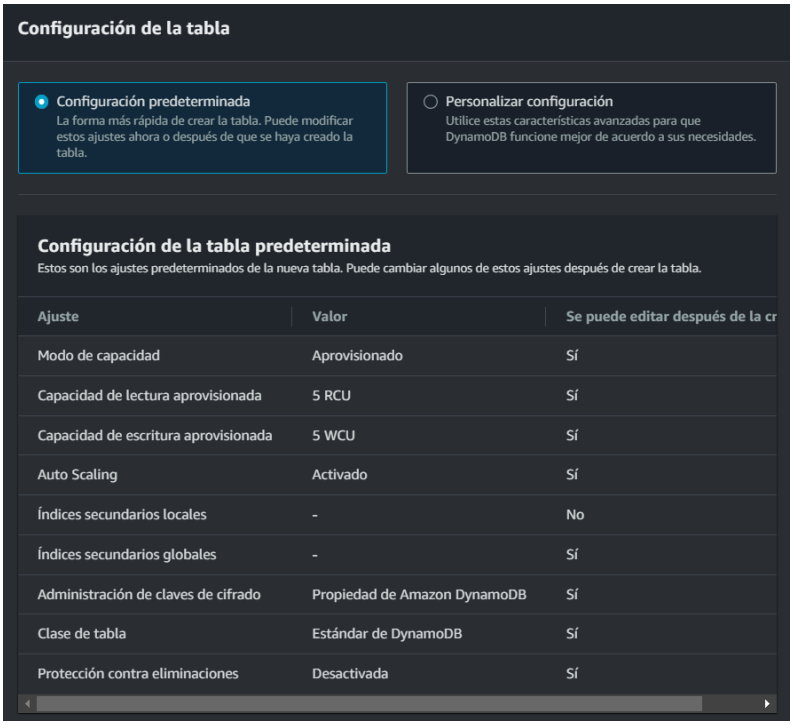


Figura 62. Configuración predeterminada de las tablas.

Una vez que las tablas fueron creadas, se procedió a implementar el código en cada una de las funciones utilizando el lenguaje Python. Estas funciones comparten similitudes en su estructura hasta el punto en que reciben los datos y los envían a DynamoDB. Sin embargo, hay una función en particular, "HumedadData", que difiere de las otras dos (TempsData y ExtraData), ya que está involucrada en el control de riego basado en la humedad registrada para un cultivo específico. Las tres funciones hacen uso de las siguientes bibliotecas: "datetime", que permite trabajar con fechas y horas; "boto3", que facilita la interacción con los servicios de AWS; y "json", utilizada para el manejo de datos en formato JSON. Estas bibliotecas son declaradas en el código, como se muestra en la Figura 63.

```
from datetime import datetime
import boto3
import json
```

Figura 63. Declaración de bibliotecas en Lambda.

Se crea función lambda_handler, siendo el punto de entrada principal para ejecutar la función Lambda. Se recibe una entrada de datos llamada "event" y se proporciona información contextual a través de "context". Se utiliza la biblioteca "boto3" para crear un cliente de DynamoDB. Dentro de la función, se extrae el valor del campo "tiempo_act" del evento de entrada (envió de la regla), el cual representa una marca de tiempo en formato UNIX. Esta marca de tiempo se convierte a un formato de fecha y hora legible utilizando la función "datetime.utcfromtimestamp(t_unix).strftime('%Y-%m-%d %H:%M:%S')", guardándose en la variable "t_actual", aplicando para las tres funciones (Figura 64).

```
def lambda_handler(event, context):
    client = boto3.client('dynamodb')
    t_unix=event.get('tiempo_act')
    t_actual=datetime.utcfromtimestamp(t_unix).strftime('%Y-%m-%d %H:%M:%S')
```

Figura 64. Inicio de la función lambda_handler() con dynamobd.

Contando con el inicio de lambda_handler(), para la función "TempsData" se realiza un arreglo de las entradas de los sensores. Los valores se redondean a dos decimales y se almacenan en variables adicionales (Sensor1 al Sensor5). Esta estructura de variables organizadas nos permite tener valores más precisos, los cuales serán enviados a DynamoDB. Además, esta estructura proporciona la flexibilidad de agregar controles adicionales en el programa en el futuro, como se muestra en la Figura 65.

```
sensor1=round(event.get('T_int1'),2)
sensor2=round(event.get('T_int2'),2)
sensor3=round(event.get('T_int3'),2)
sensor4=round(event.get('T_int3'),2)
sensor5=round(event.get('T_Ext'),2)
```

Figura 65. Arreglo de los valores de temperatura.

Asimismo, se sigue un proceso similar para establecer la configuración de la función "HumedadData" sin el redondeo. En este caso, las variables correspondientes a los sensores (Sensor1 al Sensor4) se utilizarán para el control de riego en el cultivo. La representación del código se puede observar en la Figura 66.

```
sensor1=event.get('H1')
sensor2=event.get('H2')
sensor3=event.get('H3')
sensor4=event.get('H4')
```

Figura 66. Arreglo de los valores de humedad del suelo.

Continuando con la implementación del código en cada función, se utiliza el cliente de DynamoDB junto con el método "put_item()" para insertar nuevos elementos en la tabla correspondiente. Para lograr esto, se creó un objeto "Item" que contiene los datos que se almacenaran, donde cada campo se especifica como un par clave-valor. A través del cliente de DynamoDB, se realiza la llamada al método "put_item()" proporcionando el nombre de la tabla y el objeto "Item", lo cual

garantiza un almacenamiento adecuado de los datos. Este proceso asegura que las variables ambientales registradas por diferentes sensores se guarden correctamente en las tablas de DynamoDB. Cada función tiene su propia tabla y los datos se estructuran de acuerdo a las necesidades de cada una. Esto se representa visualmente en: Figura 67,Figura 68 y Figura 69, donde se ilustran las estructuras de las tablas y cómo se almacenan los datos correspondientes a cada función.

```
response = client.put_item(
    TableName = 'Rasp_Temperaturas',
    Item = {
        'tiempo_act': {'S': t_actual},
        'T_int1': {'N':str(sensor1)},
        'T_int2': {'N':str(sensor2)},
        'T_int3': {'N':str(sensor3)},
        'T_int4': {'N':str(sensor4)},
        'T_Ext': {'N':str(sensor5)}
    }
)
```

Figura 67. Uso de "client.put_item()" para "TempsData".

```
response = client.put_item(
    TableName = 'Rasp_Humedad',
    Item = {
        'tiempo_act': {'S': t_actual},
        'H1': {'N':str(sensor1)},
        'H2': {'N':str(sensor1)},
        'H3': {'N':str(sensor3)},
        'H4': {'N':str(sensor4)}
    }
)
```

Figura 68. Uso de "client.put_item()" para "HumedadData".

```
response = client.put_item(
    TableName = 'Rasp_Extra',
    Item = {
        'tiempo_act': {'S': event['tiempo_act']},
        'Aire': {'N': event(str('Aire')]},
        'Aux1': {'BOOL': event['Aux1']},
        'Aux2': {'BOOL': event['Aux2']}
    }
)
```

Figura 69. Uso de "client.put_item()" para "ExtraData".

Luego de enviar los datos desde IoT Core, pasando por Lambda y llegando a DynamoDB, se llevó a cabo un proceso de prueba en cada tabla para asegurarse de que los datos se estén enviando correctamente (Figura 70, Figura 71 y Figura 72).

tiempo_act	T_Ext	T_int1	T_int2	T_int3	T_int4
2023-04-24 17:0…	16.57	18.57	18.7	18.63	18.63
2023-04-25 23:0…	17.76	19.32	19.39	19.39	19.39
2023-04-25 23:3…	16.89	19.2	19.32	19.32	19.32
2023-04-26 00:2…	15.51	19.95	20.13	20.01	20.01
2023-04-26 00:5…	15.32	20.07	20.26	20.07	20.07

Figura 70. Valores en tabla "Rasp_Temperaturas".

tiempo_act	H1	H2	H3	H4
2023-04-24 1…	47	0	49	0
2023-04-25 2…	46	0	48	0
2023-04-26 0…	45	0	48	0
2023-04-26 0…	45	0	47	0
2023-04-26 0…	45	0	47	0

Figura 71. Valores en tabla "Rasp_Humedad".

tiempo_act	Aire	Aux1	Aux2
26/04/2023, 16…	0.0044	null	null
26/04/2023, 16…	0.0043	null	null
26/04/2023, 16…	0.0044	null	null
26/04/2023, 16…	0.0044	null	null
26/04/2023, 16…	0.0043	null	null

Figura 72. Valores en tabla "Rasp_Extras".

Después de haber configurado la comunicación entre Lambda y DynamoDB, se agregó un fragmento de código a la función "HumedadData" para gestionar los valores de humedad del suelo en las suculentas piedra luna. Este código utiliza la biblioteca boto3 para crear un cliente de AWS IoT denominado "iot" y realiza comprobaciones basadas en los valores de los sensores: sensor1, sensor2, sensor3 y sensor4. Cuando un sensor alcanza un valor igual o superior a 25, se envía un mensaje a través del tema "raspi/comand" indicando que se debe encender un relevador específico (Relay1/2/3/4). Por otro lado, si el valor de un sensor es igual a 0, se publica un mensaje en el mismo tema para apagar el relevador correspondiente. Esta estructura de código permite controlar los relevadores según las condiciones de humedad del suelo detectadas por los sensores, estableciendo así una conexión entre la lógica de la función Lambda y las acciones físicas en el entorno de las suculentas piedra luna. Ilustrando más a detalle en la Figura 73.

```
iot = boto3.client('iot-data')
if sensor1 >= 25:
    iot.publish(topic="raspi/comand", payload=("Relay1_on"), qos=0, retain=False)
if sensor1 == 0:
    iot.publish(topic="raspi/comand", payload=("Relay1_off"), qos=0, retain=False)
if sensor2 >= 25:
    iot.publish(topic="raspi/comand", payload=("Relay2_on"), qos=0, retain=False)
if sensor2 == 0:
    iot.publish(topic="raspi/comand", payload=("Relay2_off"), qos=0, retain=False)
if sensor3 >= 25:
    iot.publish(topic="raspi/comand", payload=("Relay3_on"), qos=0, retain=False)
if sensor3 == 0:
    iot.publish(topic="raspi/comand", payload=("Relay3_off"), qos=0, retain=False)
if sensor4 >= 25:
    iot.publish(topic="raspi/comand", payload=("Relay4_on"), qos=0, retain=False)
if sensor4 == 0:
    iot.publish(topic="raspi/comand", payload=("Relay4_off"), qos=0, retain=False)
```

Figura 73. Código para el control del riego.

En resumen, la metodología de control con los servicios de AWS proporcionó una solución escalable y de alto rendimiento para la recopilación y control de variables

ambientales. Además, esta metodología allana el camino para el monitoreo utilizando herramientas de AWS TimeStream y Grafana.

3.4.4 Monitoreo con TimeStream y Grafana

El monitoreo con TimeStream y Grafana es parte de la metodología que permitió recopilar y visualizar datos en un entorno IoT de manera eficiente. En este enfoque, se utilizan las reglas de direccionamiento de IoT Core para capturar y enrutar los mensajes provenientes de los sensores hacia TimeStream. Una vez que los datos se encuentran en TimeStream, se pueden realizar consultas y análisis para obtener información relevante. Además, se utilizo3 Grafana para crear paneles interactivos y representar gráficamente los datos almacenados en TimeStream, facilitando la comprensión y el monitoreo de los diferentes aspectos del sistema IoT.

Se comenzó el apartado de monitoreo creando tres bases de datos en TimeStream: "TempsDB", "humedadDB" y "ExtraDB". En la creación de estas bases de datos, se configuró únicamente el nombre, dejando el cifrado en su configuración predeterminada, tal como se muestra en la Figura 74. Este proceso se repitió para las otras dos bases de datos creadas.

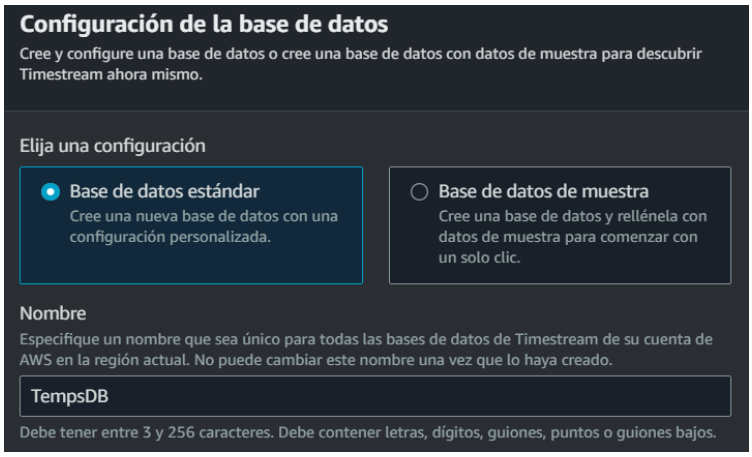


Figura 74. Creación de Base de datos en TimeStream.

Continuando con esto se crearon diversas tablas en TimeStream para organizar los datos provenientes de IoT Core. Cada tabla se configuró de manera similar, utilizando la clave de partición por defecto de TimeStream y estableciendo un tiempo de retención de 12 horas para los datos. La única diferencia entre las tablas radicó en la elección de la base de datos y el nombre correspondiente a cada una, Como se muestra en la Figura 75. Creación de la tabla para Temperatura interna 1. Además, en la tabla 1 se muestra un resumen de todas las tablas creadas, incluyendo su base de datos.

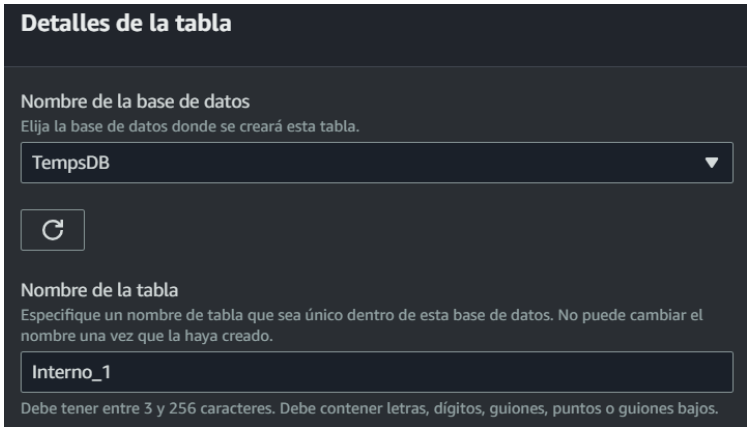


Figura 75. Creación de la tabla para Temperatura interna 1.

En la implementación de las tablas en DynamoDB para recibir datos de IoT Core, se configuraron reglas de direccionamiento de mensajes específicos como el apartado de control. En este caso, se seleccionó un atributo de la suscripción en la instrucción SQL. Esta estructura se puede observar en la Figura 76. Instruccion SQL para las reglas..

```
SELECT T_Int1 FROM 'raspi/temps'
```

Figura 76. Instruccion SQL para las reglas.

Continuando con ello se selecciona la acción de regla "Timestream table" para realizar la escritura de un mensaje en una tabla. La base de datos utilizada se llama "TempsDB", y dentro de ella se encuentra una tabla llamada "Interior_1". Cada registro en esta tabla contiene un conjunto de dimensiones, siendo "idDevice" el nombre de la dimensión utilizada. El valor de esta dimensión se obtiene de la variable "${idDevice}". Además, se selecciona el rol de IAM denominado "HumTimeStream", el cual cuenta con los permisos necesarios para el intercambio de datos entre estos dos servicios Figura 77.

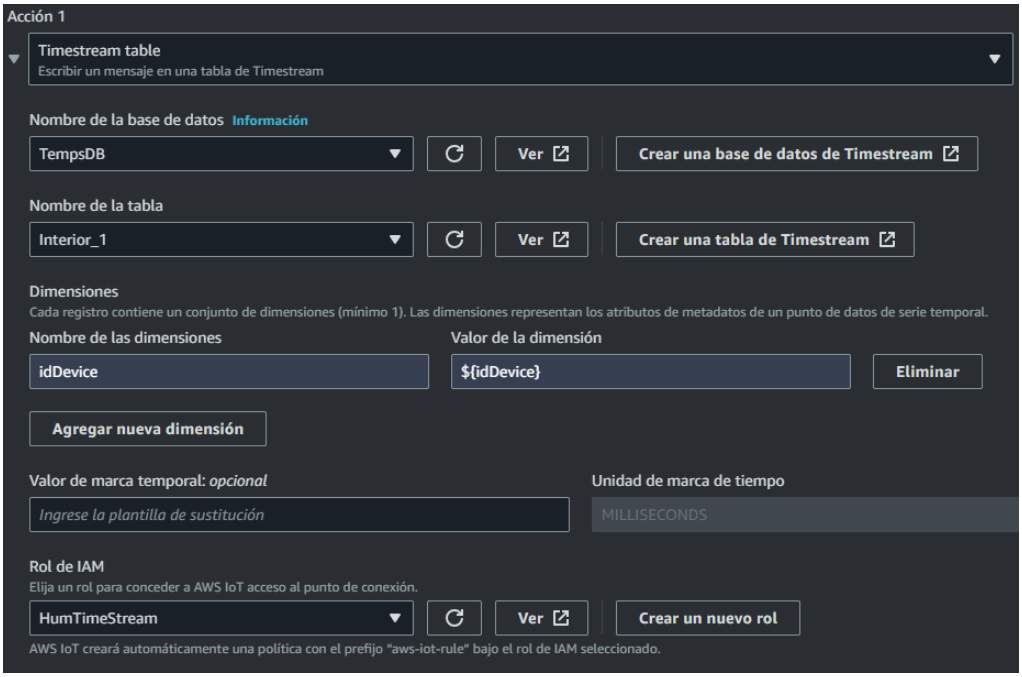


Figura 77. Configuración de las reglas de direccionamiento.

Para completar el proceso de creación de la regla de adición a TimeStream, se comparten todos los valores necesarios en la Tabla 9. Tablas creadas en AWS TimeStream con sus suscripciones de las reglas de direccionamiento., siguiendo la misma metodología de creación para las demás tablas.

Tabla 9. Tablas creadas en AWS TimeStream con sus suscripciones de las reglas de direccionamiento.

Tabla	Base de datos	Instrucción SQL	Regla
Interna_1	TempsDB	SELECT T_Int1 FROM 'raspi/temps'	Temp_int1_TS
Interna_2	TempsDB	SELECT T_Int2 FROM 'raspi/temps'	Temp_int2_TS
Interna_3	TempsDB	SELECT T_Int3 FROM 'raspi/temps'	Temp_int3_TS
Interna_4	TempsDB	SELECT T_Int4 FROM 'raspi/temps'	Temp_int4_TS
Interior_Promedio	TempsDB	SELECT T_IntF FROM 'raspi/temps'	Temp_intF_TS
Temperatura_ext	TempsDB	SELECT T_Ext FROM 'raspi/temps'	TempExterior_TS
Air	ExtraDB	SELECT Aire FROM 'raspi/extra'	Aire_TS
H_1	HumedadDB	SELECT H1 FROM 'raspi/humedad'	Humedad1_TS
H_2	HumedadDB	SELECT H2 FROM 'raspi/humedad'	Humedad2_TS
H_3	HumedadDB	SELECT H3 FROM 'raspi/humedad'	Humedad3_TS
H_4	HumedadDB	SELECT H4 FROM 'raspi/humedad'	Humedad4_TS

Completando la creación de las tablas se pasó a crear un área de trabajo en Grafana, específicamente para el área denominada "Invernadero_1". En primer lugar, se realizó el inicio de sesión en Grafana utilizando las credenciales adecuadas con el usuario que se ha estado trabajando. Se accedió a la sección "Áreas de trabajo" y se procedió a crear un nuevo espacio. Al asignarle el nombre "Invernadero_1", se estableció su identidad, se configuraron los permisos para determinar los niveles de acceso y las personas autorizadas a ingresar. Finalmente, se guarda la configuración y se obtiene acceso al área de trabajo recién creada a través del panel de navegación lateral. Al seguir esta metodología, se logró la creación efectiva y personalizada del área de trabajo "Invernadero_1" en Grafana, lo que brinda la capacidad de colocar

los valores de TimeStream y analizar los datos relacionados con el invernadero de manera óptima. Dicha área de trabajo se observa en la Figura 78.

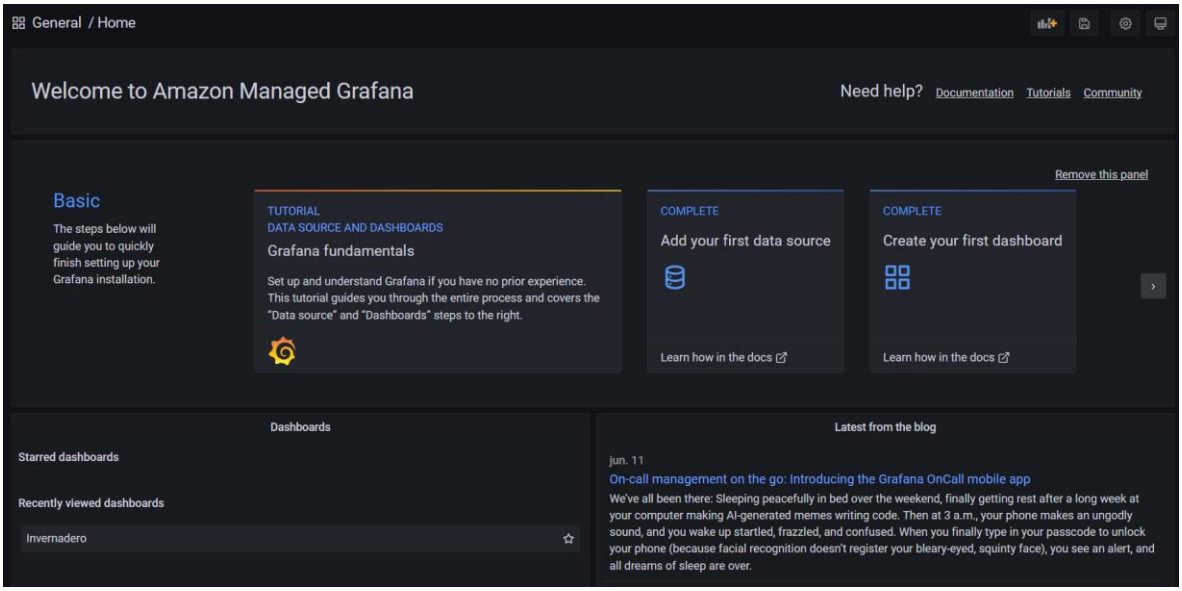


Figura 78. Área de trabajo de Grafana.

Dentro de Grafana, se ha creado un Dashboard para la visualización de gráficas e indicadores relacionados con diferentes variables y su comportamiento. Para lograr esto, se siguieron ciertos pasos. En primer lugar, se selecciona el servicio donde se tomarán los datos en este caso TimeStream, continuando se selecciona la base de datos que se ha creado con la tabla correspondiente. En la tabla, se elige la variable que se va a medir, tal como se muestra en la Figura 79 para la temperatura interna uno. A continuación, se envía una instrucción SQL "SELECT * FROM $__database.$__table order by time", para visualizar el último dato registrado en la tabla. Este proceso se repite de manera similar para cada una de las once tablas creadas en TimeStream, solo cambiando la base de datos, la tabla y el valor de medición, dejando la misma instrucción SQL.

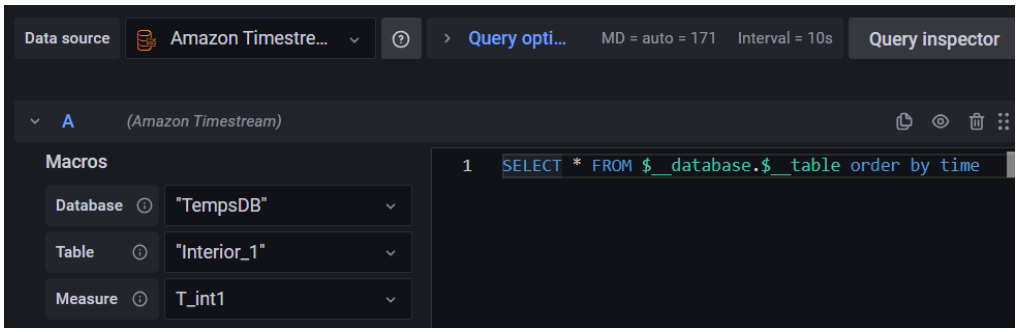


Figura 79. Selección de tablas de TimeStream para temperatura interna 1

Después de definir los datos que se mostrarán, se procedió a seleccionar el tipo de gráfico o medidor adecuado. En el caso de las temperaturas internas, se optó por utilizar el tipo de gráfico "Gauge", el cual fue configurado con un rango de medición de 0°C a 40°C. Figura 80 ilustra los cuatro medidores que representan las diferentes temperaturas internas.

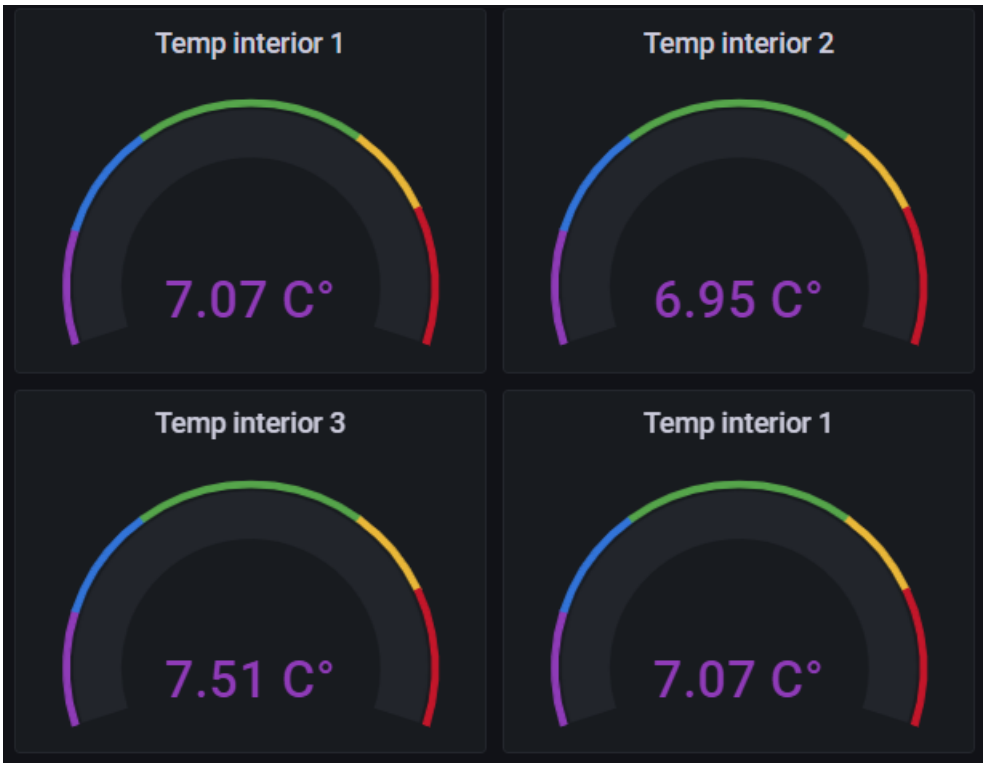


Figura 80. Medidas de las cuatro temperaturas internas.

Siguiendo con las temperaturas, se emplearon gráficas de tipo "Time series" para visualizar los datos de las tablas de "Temperatura interna promedio" y "Temperatura externa". Estas gráficas permiten analizar el comportamiento de las temperaturas en diferentes rangos de tiempo, desde intervalos de cinco minutos hasta semanas completas. El propósito de estas representaciones gráficas es observar y comprender la evolución de las temperaturas en el tiempo. Las figuras 1 y 2 muestran ejemplos de estas gráficas respectivamente.

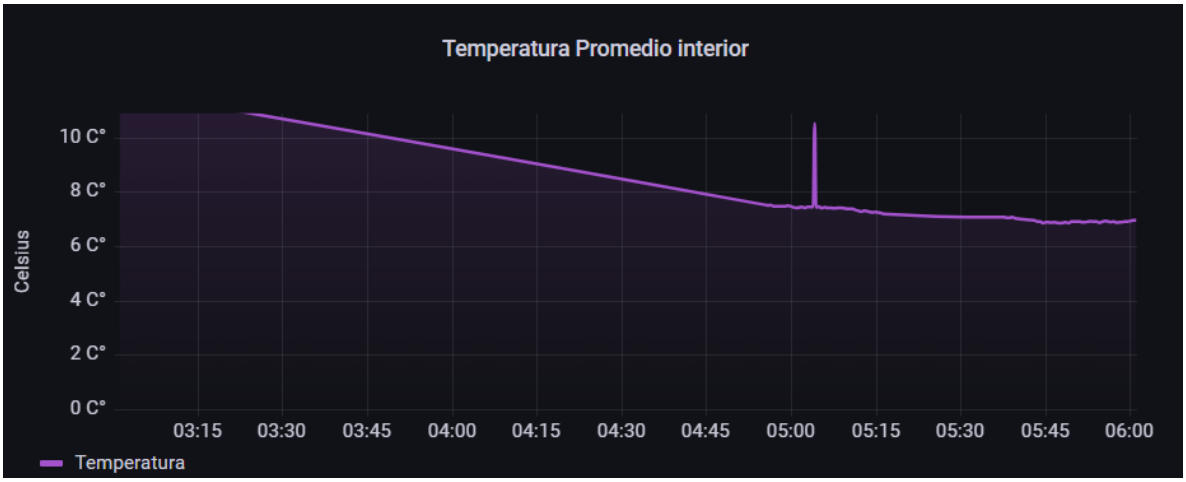


Figura 81. grafica de temperatura interna

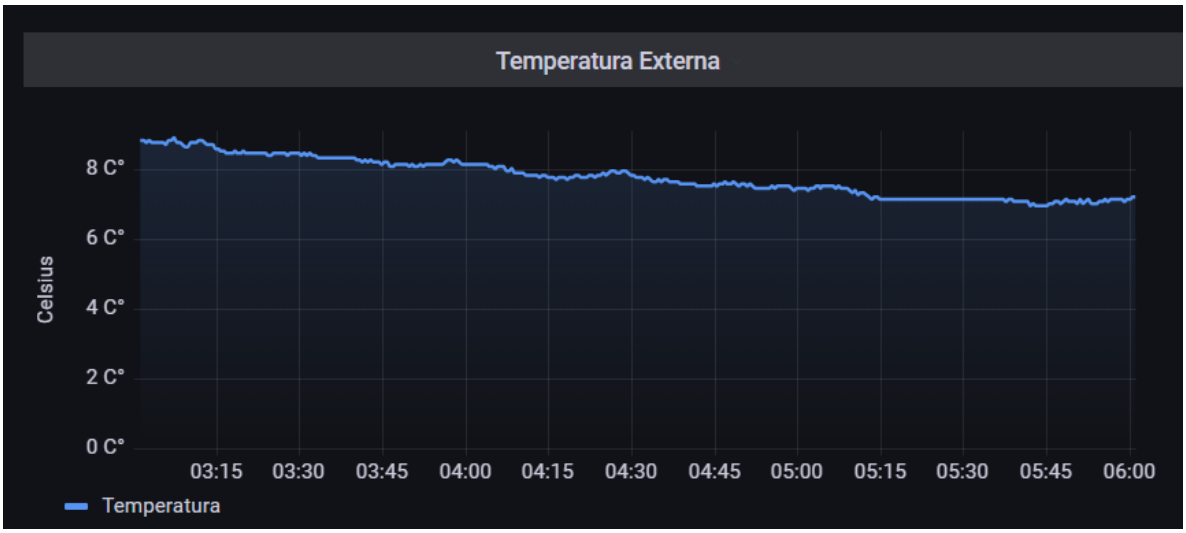


Figura 82. grafica de temperatura externa

En la etapa siguiente, se llevó a cabo la instalación y configuración de los medidores encargados de medir la humedad del suelo. Estos medidores se representaron visualmente en forma de gráficas tipo "stat", las cuales muestran el valor de cada sensor a través de un número grande que indica el valor actual. Además, se utilizó la misma representación gráfica para mostrar la calidad del aire. Figura 83muestra las cuatro gráficas correspondientes a los sensores de humedad del suelo y calidad del aire.

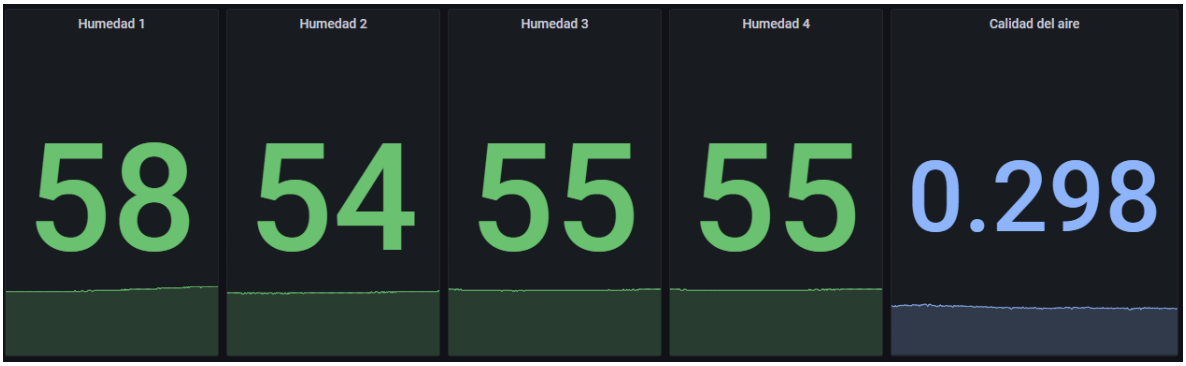


Figura 83. Gráficas para los sensores de humedad del suelo y calidad del aire.

Después de haber completado la configuración del monitoreo utilizando IoT Core, TimeStream y Grafana para visualizar las variables ambientales en el invernadero, el siguiente paso consiste en la creación de shields para la Raspberry Pi y la instalación del sistema. Estos shields proporcionarán protección y funcionalidad adicional a la Raspberry Pi, permitiendo su integración en el entorno del invernadero de manera segura y eficiente.

3.5 Creación de Shield para Raspberry con Arduino

La creación de un Shield para Raspberry Pi con Arduino es un proceso en el cual se desarrolla un dispositivo complementario que se conecta directamente a la Raspberry Pi y proporciona funcionalidades adicionales. El Shield actúa como una extensión de la Raspberry Pi, permitiendo una conexión directa y una comunicación eficiente entre ambos dispositivos. Los 7 relevadores integrados en el Shield ofrecen la capacidad de controlar diferentes dispositivos o cargas eléctricas de manera independiente, lo que brinda flexibilidad y versatilidad en la automatización de procesos. La alimentación externa incorporada en el Shield garantiza una fuente de energía confiable y estable para los componentes conectados. Esto es especialmente importante cuando se trabaja con cargas eléctricas más grandes o cuando se requiere un suministro de energía separado para un mejor rendimiento.

El slot para Arduino Nano en el Shield permite conectar y utilizar el Arduino Nano junto con la Raspberry Pi. Esto proporciona una mayor capacidad de procesamiento y control, ya que el Arduino Nano puede encargarse de tareas específicas mientras la Raspberry Pi se dedica a otras funciones, creando así un sistema más eficiente y distribuido. Además, incluye pines específicos para los sensores DS18B20, estos sensores de temperatura digital se pueden conectar directamente al Shield, lo que facilita la implementación y la lectura de datos de temperatura en el proyecto.

3.5.1 Diseño PCB

Para crear el completo se realizó el diagrama PCB en el software Fritzing (Figura 84), agregando más componentes de los utilizados en el proyecto, como 3 relevadores más y todos los pines disponibles del Arduino.

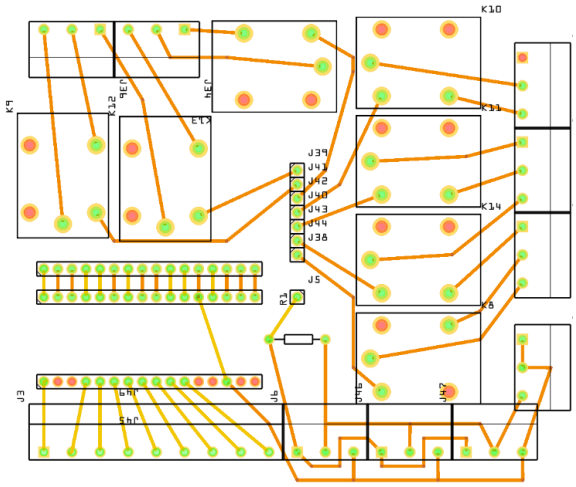


Figura 84. Diseño PCB de Shield

En la figura anterior se muestra como fue el acomodo de los 7 relevadores con sus pines de control, cada uno conectado a dos clemeros para la línea de activación. Para los sensores DS18B20 se contemplaron 3 espacios para conectarlos con la finalidad de ampliar la recolección de datos de temperatura. Los pines analógicos del Arduino están puenteados cada uno con un clemero para asegurar la comunicación y no falsear al igual una línea completa de todos sus pines digitales para disponer de ellos su en un momento se pretenden usar. A su vez se crea una segunda Shield donde se conectarán los sensores analógicos, o más si se emplean esta placa cuenta con siete slots para diferentes sensores en este caso los cuatro HD-38 y el sensor MQ-135, dejando dos libres y cuatro juegos de pines para alimentación a cinco volts (Figura 85).

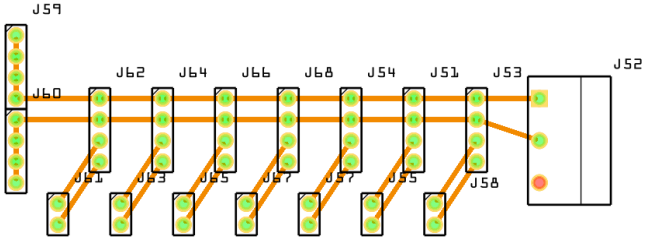


Figura 85. Complemento de Shield.

Estas Shield pasaron por un reacondicionamiento de líneas mediante CorelDraw, vectorizando las líneas y definiéndose de manera correcta para poder maquinar la placa quedando como se muestra en la Figura 86. Vectorización de líneas en los Shields.

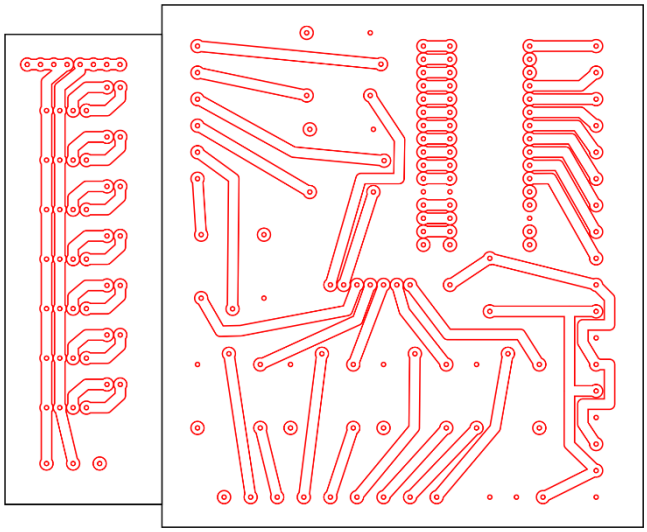


Figura 86. Vectorización de líneas en los Shields

Para lograr un tamaño compacto y eficiente, se acopló el diseño dentro de un espacio de diez por diez centímetros para el Shield más grande, y se redujo a un espacio de tres por nueve centímetros para el Shield más pequeño. Para delimitar las áreas de grabado en la placa fenólica, se utilizaron vectores de color negro, mientras que los vectores rojos indicaron las pistas que serían grabadas en la placa (Figura 86).

3.5.2 Perfilado de vectores

Una vez finalizada la etapa de vectorización de los diseños, se procede a utilizar el software Aspire para llevar a cabo los perfilados de los vectores. Este software proporciona una amplia gama de herramientas y funciones que permiten obtener resultados precisos y de alta calidad en el proceso de perfilado. Para llevar a cabo

esta tarea, se utiliza una broca especializada de grabado, diseñada específicamente para este propósito, así como una broca de perforado adecuada. Llevando a cabo tres procesos de perfilado para cada uno de los Shields desarrollados. Cada perfilado tuvo un propósito específico y se realizaron considerando diferentes parámetros. El primer perfilado consistió en el grabado de líneas en la placa con una profundidad de 0.2 mm. Esta etapa permitió marcar las rutas y conexiones requeridas para el circuito (Figura 87).

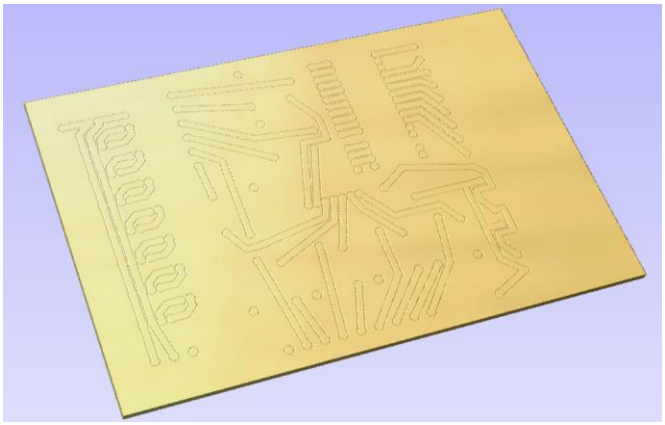


Figura 87. Simulación del perfilado para grabar de pistas en Shields.

Posteriormente, se llevó a cabo el segundo perfilado, el cual fue destinado al perforado de los pines a una profundidad de 2 mm en toda la placa. Estos pines proporcionan los puntos de soldadura para los componentes electrónicos. Esta fase es crucial para garantizar una correcta conexión y fijación de los elementos en la placa (Figura 88).

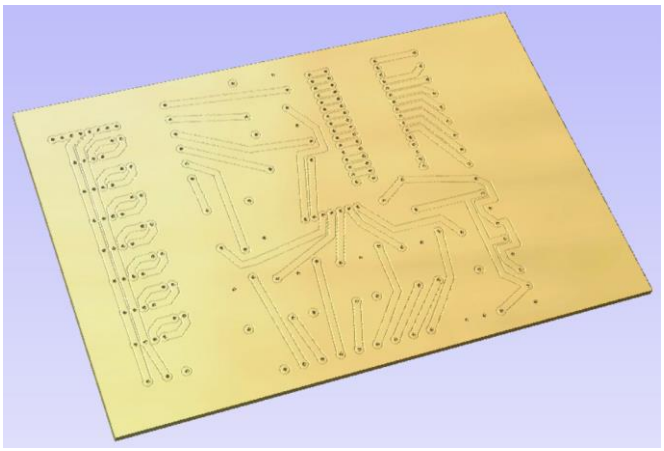


Figura 88. Simulación del perfilado dos para el perforado de pines.

Finalmente, se procedió con el tercer perfilado, el cual tuvo como objetivo delimitar la placa fenólica que serviría como base para el circuito. Mediante este proceso, se crearon bordes precisos y definidos en la placa, lo que contribuye a su estabilidad y protección (Figura 89).

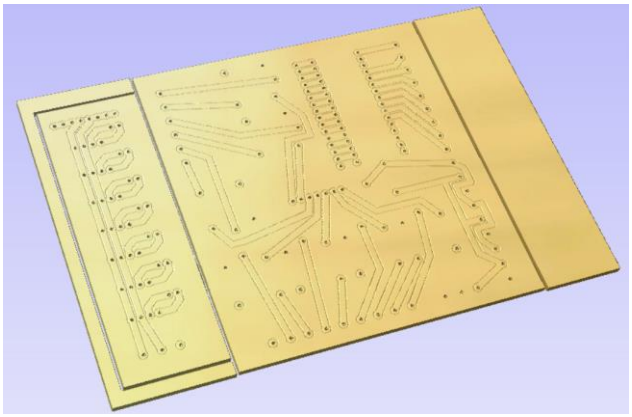


Figura 89. Simulación del perfilado tres para la delimitación y tamaño de cada Shield.

Cabe destacar que cada perfilado se realizó cuidadosamente, siguiendo los parámetros establecidos en el software Aspire. Esto garantizó la precisión y calidad requerida en cada etapa del proceso de fabricación de los Shields. Los resultados

obtenidos en estos perfilados fueron fundamentales para continuar con la creación de las Shields.

3.5.3 Grabado de Shields

Continuando se procedió con el siguiente paso del proceso de fabricación, que consistió en el grabado de los perfilados en la placa fenólica utilizando una máquina CNC 3018 PRO (Figura 90). Para controlar la máquina, se empleó el software Grblcontrol, también conocido como Candle. Este software proporciona una interfaz intuitiva que permite configurar los parámetros de grabado y controlar el movimiento de la máquina CNC.

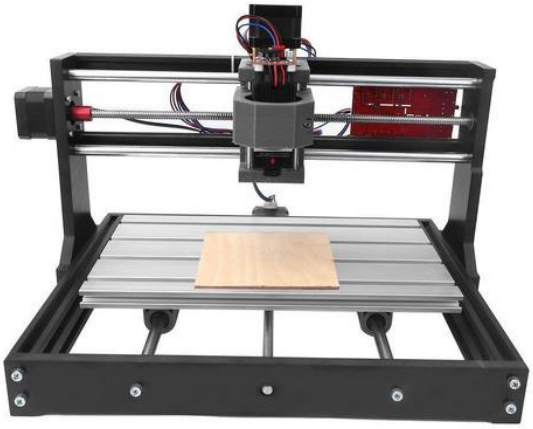


Figura 90. CNC 3018 Pro

Antes de iniciar el grabado, se realizó una simulación en el software para verificar la correcta ubicación de los perfilados y evitar cualquier error en el proceso. La simulación mostró una representación visual de cómo se llevaría a cabo el grabado en la placa fenólica, lo cual permitió ajustar los parámetros necesarios para obtener resultados precisos y de alta calidad como se muestra a continuación (Figura 91).

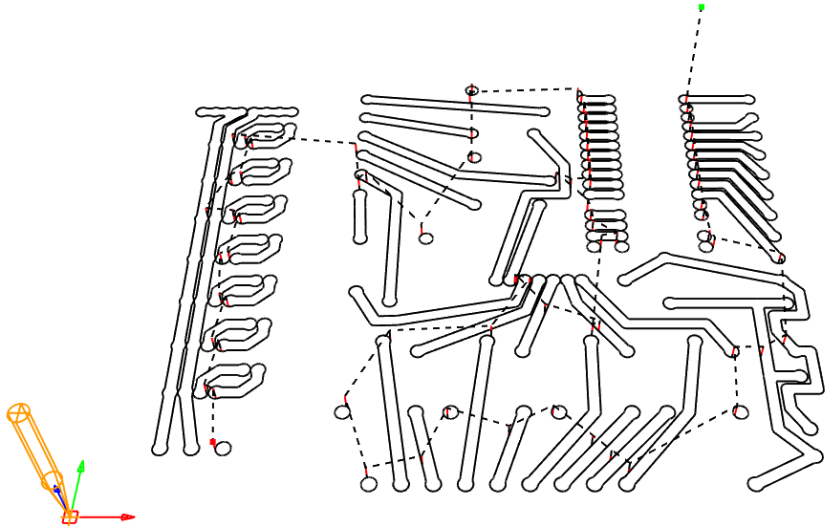


Figura 91. Simulación de GrblControl de perfilados

Una vez verificada la simulación y configurados los parámetros adecuados, se procedió a realizar el grabado de los perfilados en la placa fenólica. La máquina CNC, controlada por el software GrblControl, se encargó de seguir el diseño establecido previamente, grabando con precisión los contornos y detalles requeridos (Figura 92).

Figura 92. Grabado de placa fenólica con CNC 3018 Pro

Gracias a la combinación del CNC 3018 PRO y el software GrblControl (Candle), se logró obtener resultados satisfactorios en el grabado de los perfilados en la placa fenólica. Este proceso permitió la creación de los espacios necesarios para alojar los componentes electrónicos de forma precisa y segura como se muestra en la Figura 93.

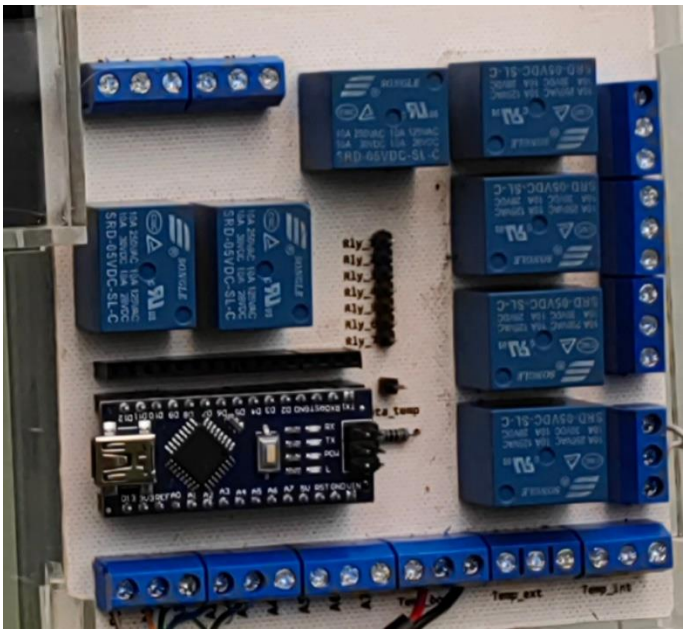


Figura 93. Shield concluida.

3.6 Montaje del sistema en el invernadero

Después de completar el sistema de control y monitoreo, así como la creación de las protecciones necesarias, se llevó a cabo la instalación del sistema en un invernadero perteneciente a la licenciatura en Biología del Instituto Tecnológico Superior de Zacapoaxtla (ITSZ). El propósito principal de este sistema es monitorear y controlar el crecimiento de las suculentas piedras de luna, las cuales son objeto de investigación en esta institución educativa. Con esta implementación, se busca obtener datos precisos y realizar un seguimiento efectivo del desarrollo de este cultivo en particular.

3.6.1 Instalación del sistema

El invernadero seleccionado para este proyecto es un invernadero tipo túnel con las siguientes dimensiones: quince metros de largo, cuatro metros de ancho y cuatro metros de alto (ver Figura 94). Con el propósito de monitorear y controlar las condiciones ambientales dentro del invernadero, se instaló el sistema de las tarjetas embebidas encargadas de la recolección de datos.

Figura 94. Invernadero de la licenciatura de Biología del ITSZ.

Dentro del invernadero, se colocó estratégicamente un contenedor de acrílico diseñado exclusivamente para este propósito. En dicho contenedor se encuentran alojadas la tarjeta Raspberry y la shield con Arduino Nano. Para asegurar su protección, el contenedor se ubicó en un soporte de PTR dentro del invernadero, garantizando que no esté expuesto a riesgos como la humedad o caídas accidentales. Además, se seleccionó cuidadosamente la ubicación del contenedor para asegurar una óptima conexión WiFi con el instituto, maximizando así la calidad de la conexión (Figura 95).

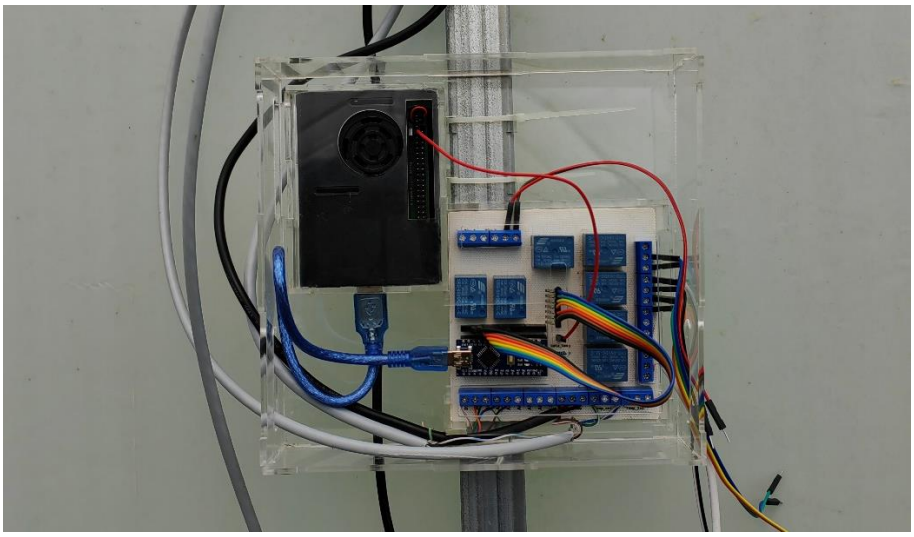


Figura 95. Colocación de soporte de acrílico con el sistema incluido y conectado.

Una vez instalado el soporte en el invernadero, se procedió a realizar las conexiones necesarias para asegurar la comunicación entre los diferentes componentes. Se conectaron los puertos de comunicación del Arduino a los relés, así como las líneas de comunicación hacia los sensores. Esta interconexión permite que el Arduino pueda recibir y enviar señales a los relés y sensores de manera efectiva, facilitando así el monitoreo y control de las condiciones ambientales dentro del invernadero.

3.6.2 Localización de sensores

Continuando con la instalación, se procedió a instalar los nueve sensores necesarios. El proceso se inició con la instalación de los cuatro sensores de humedad dentro de las plantas. Esta conexión se realizó de manera ágil y eficiente gracias a la presencia de una Shield adicional, lo que facilitó la instalación y la identificación de cada uno de estos sensores, como se muestra detalladamente en la Figura 96.

Figura 96. Instalación de sensores de humedad del suelo con la Shield extra.

En cuanto a los sensores de temperatura, se han instalado los cuatro sensores distribuidos estratégicamente en varias partes del invernadero, con el objetivo de abarcar un área de muestreo más amplia y obtener múltiples lecturas. El primer sensor se ha colocado cerca de las plantas que se están monitoreando, mientras que el segundo se encuentra en proximidad del plástico que cubre el invernadero. Los dos sensores restantes se han ubicado en la parte superior del mismo, de manera que se recolecten datos de una zona más extensa dentro del invernadero, tal como se muestra en las Figuras Figura 97, Figura 98 y Figura 99. Esta disposición estratégica de los sensores permite obtener mediciones más representativas de las condiciones de temperatura en diferentes áreas del invernadero.

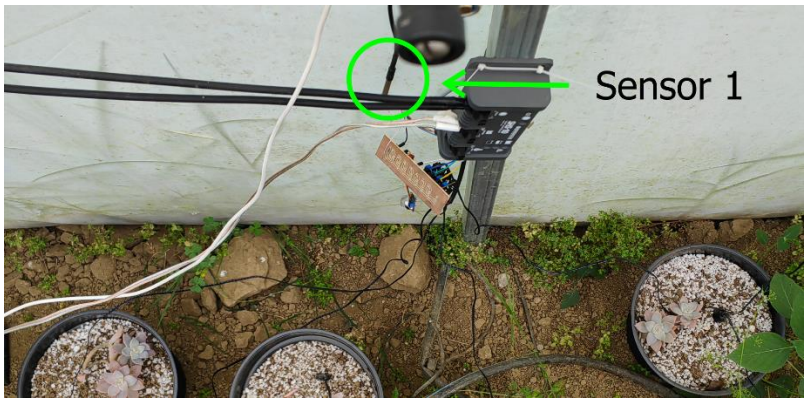


Figura 97. Ubicación del sensor de temperatura 1.

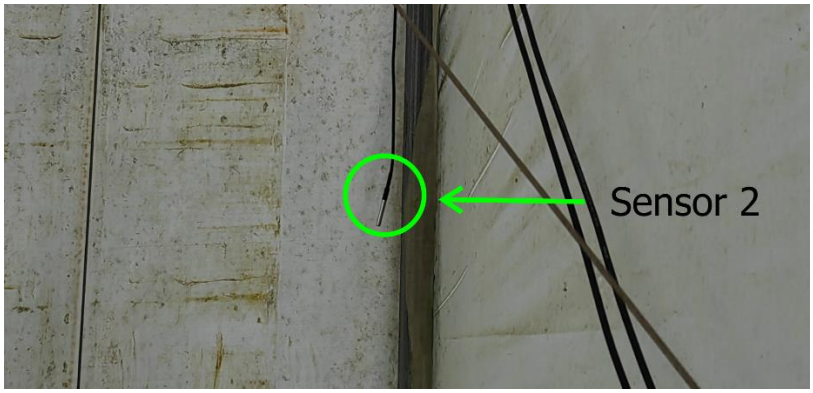


Figura 98. Ubicación del sensor de temperatura 2.

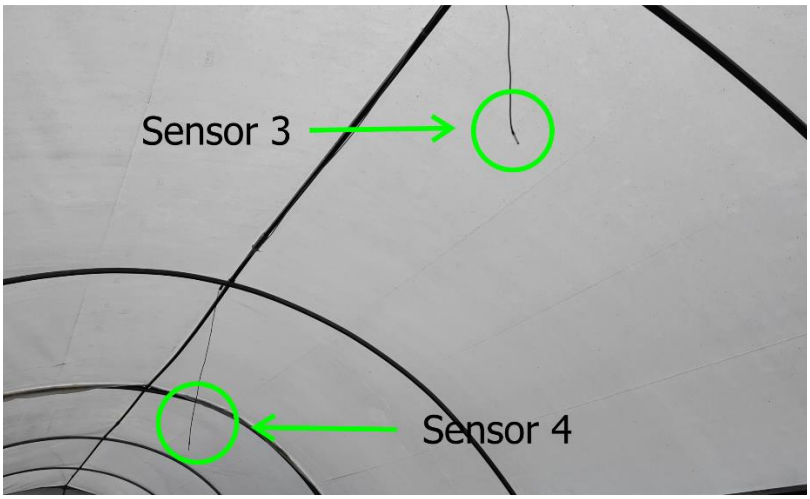


Figura 99. Ubicación del sensor de temperatura 3 y 4.

En una etapa posterior, se realizó la adición del sensor de calidad del aire a la shield, junto con los sensores de humedad del suelo, tal como se muestra en detalle en la Figura 100. Esta incorporación permitió ampliar la capacidad de monitoreo del invernadero, brindando información adicional sobre las condiciones ambientales internas.

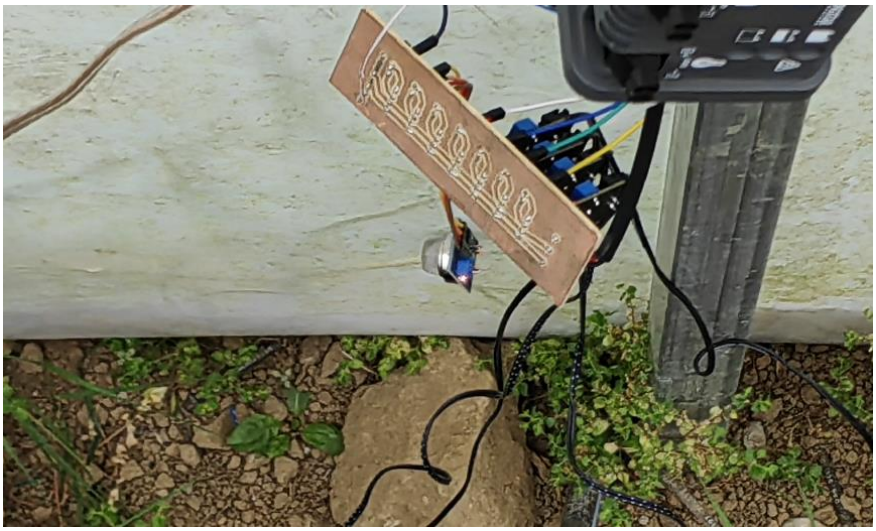


Figura 100. Sensor de calidad del aire.

La instalación de los sensores ha sido fundamental para establecer un sistema de recolección de datos altamente sólido y preciso, lo cual desempeña un papel crucial en el monitoreo y control efectivo de las condiciones ambientales en el invernadero. Ahora, nos adentramos en la siguiente fase del proceso: la alimentación del sistema. Durante esta etapa, se implementarán las medidas necesarias para garantizar un suministro adecuado de energía y recursos al sistema, asegurando su funcionamiento continuo y óptimo.

3.6.3 Alimentación del sistema

En cuanto a la alimentación del sistema, se tomó la decisión de utilizar energías limpias, específicamente la energía solar, como fuente de suministro para todo el sistema. Para este propósito, se instaló un panel solar de 50 watts que proporciona un suministro constante de energía. El panel solar se ubicó estratégicamente en la parte exterior del invernadero, tal como se muestra en la figura 1, aprovechando al máximo la radiación solar disponible. Esta elección asegura una fuente de alimentación sostenible y renovable para el funcionamiento continuo del sistema.

Figura 101. Colocación del panel solar.

la energía generada por el panel solar se dirige hacia un controlador de carga solar encargado de gestionar su distribución. A través de este controlador, la energía recolectada se envía hacia una batería de 12 volts (Figura 102). Posteriormente, la batería se conecta a un inversor de corriente continua a corriente alterna, el cual suministra energía al sistema de control y al sistema de riego (Figura 103).

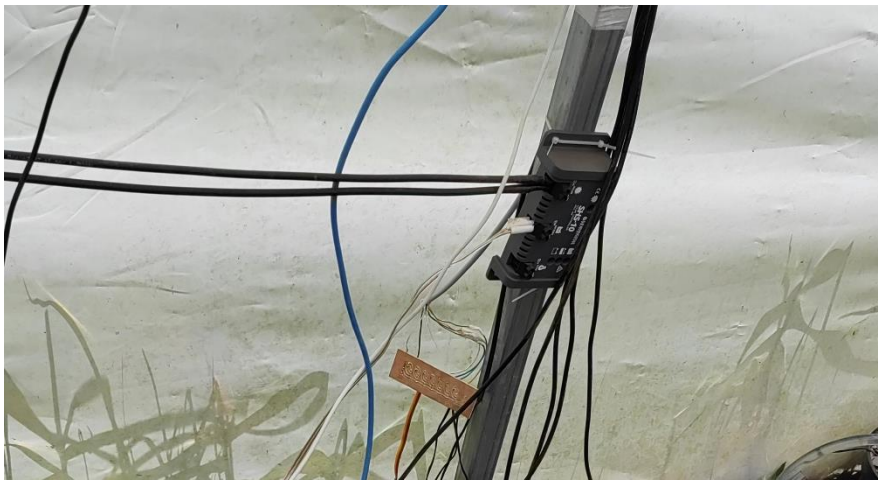


Figura 102. Instalación del controlador de carga solar.

Figura 103. Instalación del inversor.

De esta manera, se establece un flujo eficiente de energía que garantiza una alimentación constante y confiable de los componentes necesarios durante las 24 horas del día, sin importar si es de día o de noche. Esto asegura un funcionamiento ininterrumpido del sistema, permitiendo un monitoreo y control continuo para optimizar el crecimiento y la productividad de las plantas en el invernadero.

3.6.4 Instalación del control de riego

además de la instalación del sistema de recolección de datos, se llevó a cabo la implementación de un sistema de riego por goteo para cada una de las plantas en el invernadero. Para ello, se procedió en primer lugar a la instalación de mangueras de media pulgada, conectadas en paralelo mediante uniones tipo "T", tal como se ilustra en la Figura 104. Estas uniones están vinculadas a válvulas solenoides que controlan el flujo de agua hacia cada planta de manera individual.

Figura 104. Conexión de sistema hidráulico con válvulas solenoides.

Las válvulas solenoides utilizadas en conjunto con la bomba de agua operan con una alimentación de 12 voltios, lo cual asegura una adecuada funcionalidad y coordinación. Una vez activada por medio del control desde AWS Lambda, la bomba suministra agua al sistema para el riego. Para garantizar un rendimiento óptimo, el sistema de riego por goteo está diseñado con mangueras sin retorno hacia el contenedor de agua. No obstante, se ha incorporado una válvula manual adicional para evitar la sobrecarga de la bomba debido a una presión excesiva de agua. Esta estrategia de configuración, ilustrada en la figura 1, permite un control preciso y eficiente del riego en el invernadero, asegurando el suministro adecuado de agua a las plantas.

Figura 105. Instalación del sistema de control de riego, con retorno al contenedor.

Para asegurar un funcionamiento eficiente, se conectaron las válvulas a los relevadores correspondientes a cada cultivo. Esta configuración permite que las electroválvulas y los sensores trabajen de manera independiente pero coordinada, en beneficio de las plantas. De esta forma, se logra un control preciso y optimizado de las condiciones ambientales, favoreciendo el crecimiento y desarrollo adecuado de las plantas en el invernadero.

3.7 Generación de códigos de set y reset

Una de las configuraciones cruciales en este sistema es la implementación del inicio automático. Dada la posibilidad de desconexiones de red o interrupciones en la alimentación eléctrica, se han establecido tres códigos de inicio y reinicio del sistema en intervalos de tiempo predefinidos mediante instrucciones ".service". Esta medida garantiza que el sistema se reactive automáticamente en caso de fallos, asegurando así su continuidad operativa y evitando posibles inconvenientes por pérdida de conexión o interrupciones de energía. De esta manera, se logra una mayor confiabilidad y disponibilidad del sistema en el invernadero.

3.7.1 Inicio de comunicación Raspberry-AWS

El primer aspecto fundamental es el inicio automático del programa responsable de enviar los datos desde la Raspberry a la nube. Para lograr esta funcionalidad, se utilizó el archivo de instrucciones de Linux ".service", como se muestra visualmente en la Figura 106. Este enfoque permite que el programa se inicie automáticamente al arrancar el sistema operativo de la Raspberry, asegurando así la continuidad en la transmisión de datos sin necesidad de intervención manual.

```
GNU nano 5.4
[Unit]
Description=Protocole_RASP-AWS
After=network.target

[Service]
Type=simple
User=rasp
WorkingDirectory=/home/rasp/aws/
ExecStartPre=/bin/sleep 60
ExecStart=/usr/bin/python3 /home/rasp/aws/rasp-aws.py
Restart=always

[Install]
WantedBy=multi-user.target
```

Figura 106. Inicio de programa "rasp-aws.py".

Explicando la Figura 106 se define el servicio y su comportamiento. "After=network.target" Especifica que debe iniciarse después de que la red esté disponible. "Type=simple" indicando que el servicio es un proceso simple. "User=rasp" especifica que el servicio se ejecutará con el usuario "rasp". "WorkingDirectory" estableciendo el directorio de trabajo del servicio, donde se encuentra el archivo "rasp-aws.py". "ExecStartPre" proporciona una instrucción opcional para ejecutar antes del inicio del servicio, en este caso, se espera 60 segundos antes de iniciar el programa. "ExecStart" especifica el comando que se ejecutará para iniciar el programa "rasp-aws.py" utilizando Python 3. "Restart=always" nos indica que el servicio se reiniciará automáticamente en caso

de que se detenga o se produzca un error. En resumen, este código de arranque configuro un servicio en la Raspberry Pi para ejecutar el programa "rasp-aws.py" que enviará datos a AWS. El servicio se iniciará automáticamente al arrancar el sistema operativo y se reiniciará en caso de que se detenga o falle. Asegurando él envió de datos.

3.7.2 Inicio de comunicación AWS-Raspberry

El siguiente código tiene como objetivo iniciar el programa "aws-rasp.py" en una Raspberry Pi. Este programa es fundamental para el control del riego en un sistema y para activar los relés necesarios. Se ha diseñado de manera que haya una sincronización adecuada entre el programa y los servicios de AWS. evitando posibles errores de comunicación. El código comienza después de un periodo de sesenta y cinco segundos, permitiendo que otros procesos se inicien correctamente antes de ejecutar "aws-rasp.py". Esta demora garantiza que la Raspberry Pi esté lista para recibir y enviar datos sin interrupciones. El propósito principal del programa "aws-rasp.py" es enviar señales de activación a los relés. Esto se logra mediante el envío de comandos específicos que son recibidos a través de la suscripción "raspi/comand" en AWS. Estos comandos son interpretados por el programa y se ejecutan las acciones correspondientes para activar los relés de manera adecuada.

```
[Unit]
Description=Protocole_AWS-RASP
After=network.target

[Service]
Type=simple
User=rasp
WorkingDirectory=/home/rasp/aws/
ExecStartPre=/bin/sleep 65
ExecStart=/usr/bin/python3 /home/rasp/aws/aws-rasp.py
Restart=always

[Install]
WantedBy=multi-user.target
```

Figura 107. Inicio de programa "aws-rasp.py".

A continuación, se muestra la Figura 107 que proporciona el código completo del programa "aws-rasp.py". Esta representación visual del código puede ser útil para comprender mejor su implementación y funcionalidad. Este código es muy similar al anterior con la instrucción ".service", pero se diferencia en el tiempo de inicio para asegurar una correcta secuencia de operaciones.

3.7.3 Reseteo del sistema

Se ha creado un programa utilizando el formato de archivo ".service" en Linux con el propósito de reiniciar automáticamente el sistema cada seis horas. Esta implementación se llevó a cabo para prevenir posibles errores o fallos en la recolección y envío de datos cuando hay una alta latencia en la conexión.

```
[Unit]
Description=Reiniciar Raspberry Pi

[Service]
Type=simple
ExecStart=/bin/bash -c "sleep 3h && sudo reboot"

[Install]
WantedBy=multi-user.target
```

Figura 108. Codigo de reseteo.

El código en cuestión se basa en una unidad de servicio de systemd que configura el reinicio programado. En la sección [Unit], se proporciona una breve descripción del servicio, indicando que su función principal es reiniciar la Raspberry Pi. En la sección [Service], se especifica que el tipo de servicio es "simple", lo que significa que el proceso se ejecuta en primer plano sin demonización. El comando utilizado en el campo "ExecStart" es "/bin/bash -c", el cual abre una nueva instancia de la

shell de bash. A continuación, se establece un tiempo de espera de tres horas utilizando el comando "sleep 3h". Después de ese período, se ejecuta el comando "sudo reboot" para reiniciar el sistema. En la sección [Install], se indica que el servicio debe ser habilitado para el nivel de ejecución "multi-user.target". Esto significa que el servicio se iniciará automáticamente cuando el sistema arranque en el modo multiusuario. La finalidad de este código es evitar que el sistema se bloquee o experimente problemas en la recolección y envío de datos debido a la latencia de la conexión. Al reiniciar cada tres horas, se busca mantener la estabilidad y asegurar que los datos se envíen de manera adecuada, incluso en condiciones de red menos óptimas. Al reiniciar regularmente, se proporciona una oportunidad para restablecer cualquier estado inconsistente y garantizar un funcionamiento continuo y confiable del sistema.

CAPÍTULO IV RESULTADOS

En este capítulo, se presentan los resultados de la implementación del sistema que demuestra la eficiencia del uso combinado de AWS y Raspberry Pi. Se aborda el desarrollo del sistema propuesto, describiendo sus componentes, dispositivos electrónicos utilizados, diseño, programación, entre otros aspectos relevantes. Además, se exponen los resultados obtenidos a través de las pruebas realizadas con el sistema. Para ilustrar el funcionamiento y evaluar su desempeño, se utilizó un caso de estudio específico: un cultivo de suculentas piedra luna que ha estado en crecimiento durante un período de cuatro semanas. Obteniendo lo siguiente:

4.1 Evaluación de la eficacia del sistema

En esta investigación, se realizó un estudio sobre los efectos de las variables ambientales en un entorno de cultivo durante un período de 4 semanas. Se analizaron dos variables clave: *la humedad del suelo y la relación entre la temperatura externa e interna del invernadero*. El objetivo principal fue evaluar cómo estas variables afectan el crecimiento y desarrollo de las plantas. En relación a la humedad del suelo, se estableció un rango de variación máximo del 25% entre los valores máximos y mínimos para cada planta en el invernadero. Se buscó comprender cómo esta variación influye en la disponibilidad de agua y, por consiguiente, en el desarrollo de las plantas. Por otro lado, se estudió la relación entre la temperatura externa e interna del invernadero. Se registraron y compararon los datos de temperatura exterior e interior para analizar cómo las condiciones ambientales externas impactan en la estabilidad y el control de la temperatura dentro del invernadero. Esto permitió evaluar la eficiencia del sistema de control de temperatura implementado. A lo largo de las 4 semanas de estudio, se recopilaron datos detallados sobre las variaciones de humedad del suelo y las diferencias de temperatura entre el entorno externo e interno. Estos datos brindaron una base sólida para evaluar los efectos de estas variables en el crecimiento y desarrollo de las plantas en el invernadero.

Partiendo de las lecturas de los sensores de humedad del suelo, se ha observado un eficiente funcionamiento de los actuadores, los cuales se han activado de manera adecuada para mantener la humedad del suelo en el rango ideal para las suculentas, que es del 1% al 25%. Esto se puede apreciar en la gráfica de la Figura 109, donde se muestra el comportamiento a lo largo de una semana en cada una de las cuatro plantas, evidenciando una consistencia en el modo de lectura y activación de las válvulas solenoides para el riego.

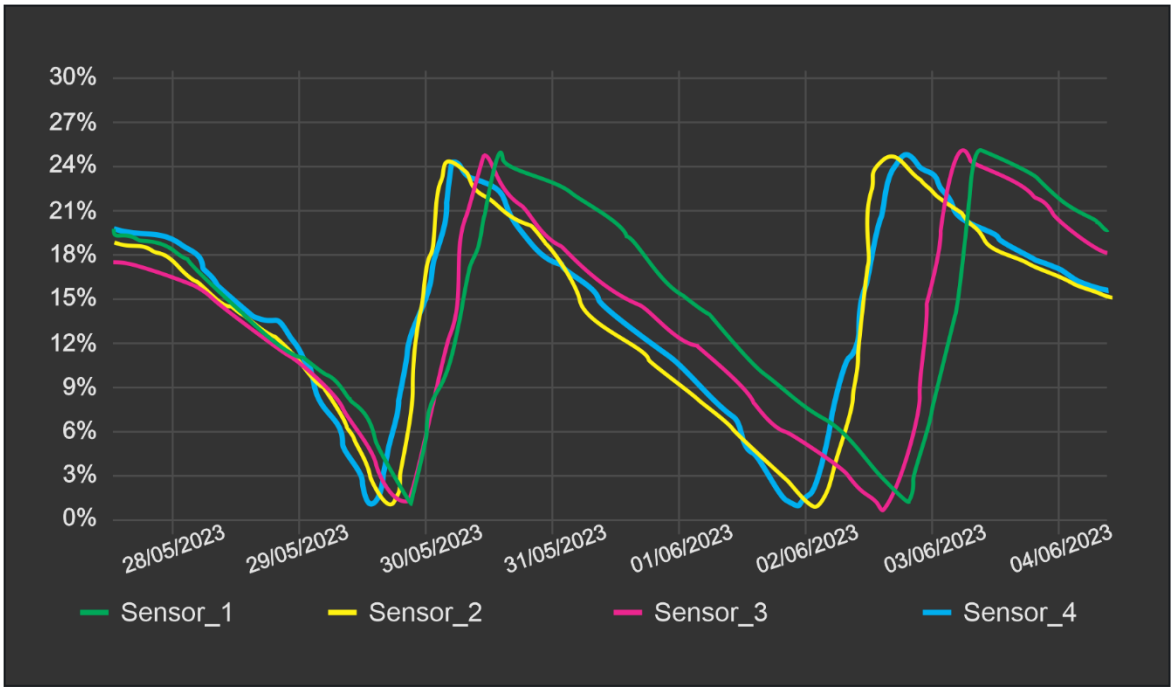


Figura 109. Graficas del comportamiento en la humedad del suelo.

A continuación, se observa que el comportamiento de los sensores de humedad está demostrando una eficiencia notable. Las variaciones en el tiempo entre la humedad y el riego pueden deberse al tipo de planta y al sistema de goteo utilizado. A pesar de esto, el sistema en general está funcionando de manera adecuada, reflejando su eficiencia de forma precisa. En pruebas anteriores, se identificó un problema en la recopilación de datos que arrojaba un valor de "-2186". Para solucionar este error, se implementó un método consistente en mejorar la conexión a tierra en la shield

adicional encargada de alimentar y comunicar con estos sensores. Como resultado, los sensores han demostrado no presentar corrosión en las terminales de cada punta del sensor HD-38, lo que prolonga su vida útil.

En cuanto a la Shield, también se ha mantenido sin fallas el sensor de calidad del aire, como se puede apreciar en la Figura 110. Este sensor ha mantenido un rango de medición entre 0.25 y 0.45, lo cual se encuentra dentro de los límites tolerables establecidos.

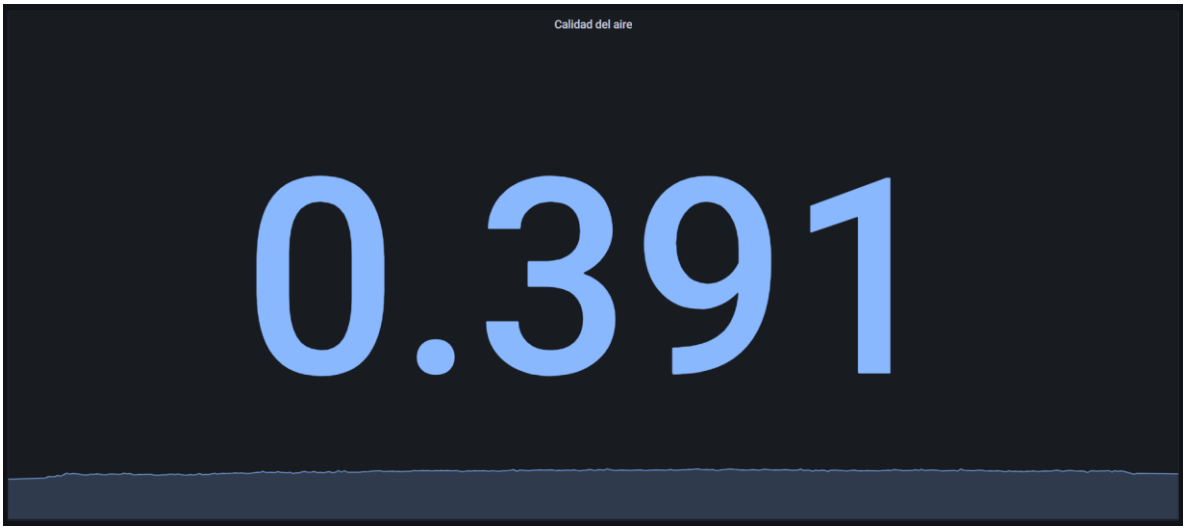


Figura 110. Gráfica de sensor de calidad del aire.

Pasando a los sensores de temperatura, se encontraron datos que revelan un panorama intrigante. Las gráficas que representan las temperaturas internas muestran un fenómeno interesante, anqué, en general, se mantienen estables, el resultado que cada sensor registra variaciones de temperatura de hasta 5 grados Celsius según su ubicación. Esta variabilidad nos brinda una perspectiva amplia y minuciosa de las temperaturas internas.

Durante el transcurso del día, en el periodo promedio de 11 a 14 horas, el sensor tres sobresale al registrar la temperatura más alta. Además, al examinar las temperaturas intermedias, encontramos que los sensores cuatro y uno presentan una diferencia aproximada de 3 a 2 grados Celsius en sus mediciones. Por último, el sensor dos muestra una tendencia distinta, ya que sus mediciones revelan temperaturas más bajas en comparación con los otros tres sensores como se puede ilustrar en la Figura 111.

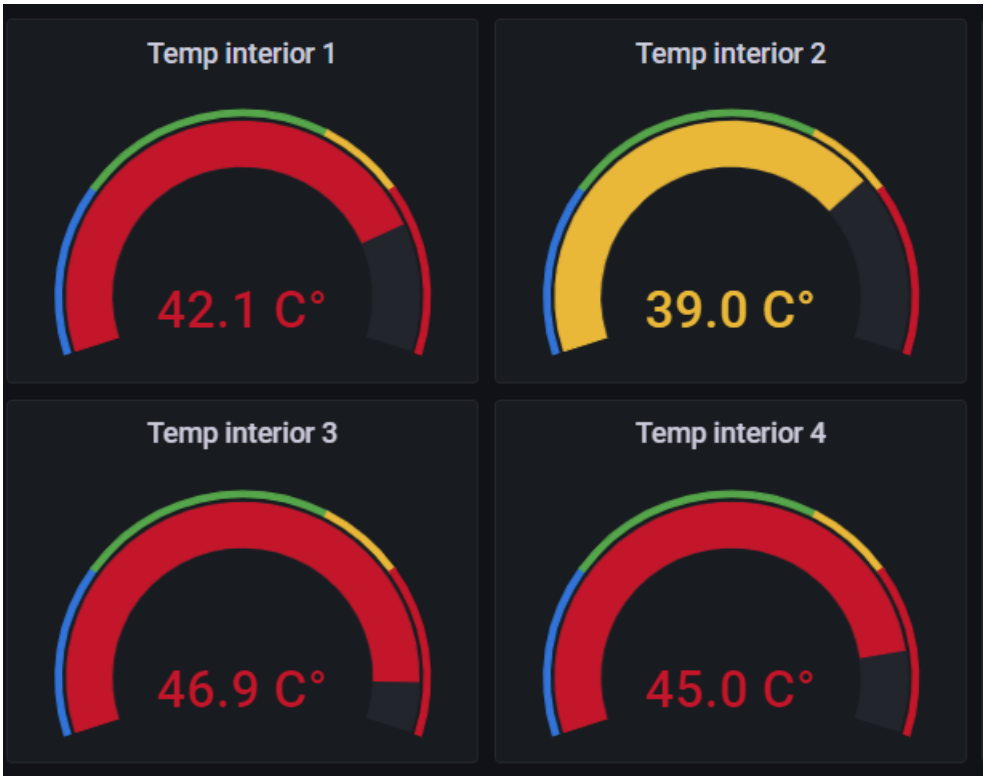


Figura 111. Toma de temperaturas internas en horario de entre 11 a 14 horas.

En un escenario diferente, durante la noche, se puede observar un comportamiento contrastante. En este caso, los cuatro sensores muestran una diferencia de temperatura que no supera los 2 grados Celsius entre ellos (Figura 112). Esto indica que las condiciones cambian notablemente entre el día y la noche en relación con la radiación solar dentro del invernadero. Estos hallazgos brindan la oportunidad de

implementar diversas medidas para controlar y equilibrar aún más las temperaturas. Es fascinante notar cómo el ciclo diurno y nocturno influye en el comportamiento térmico dentro del invernadero. Durante el día, la radiación solar incide directamente sobre el espacio, generando un aumento significativo en las temperaturas internas. Sin embargo, cuando la noche llega y el sol se oculta, la ausencia de esta fuente de calor provoca un descenso en las temperaturas.

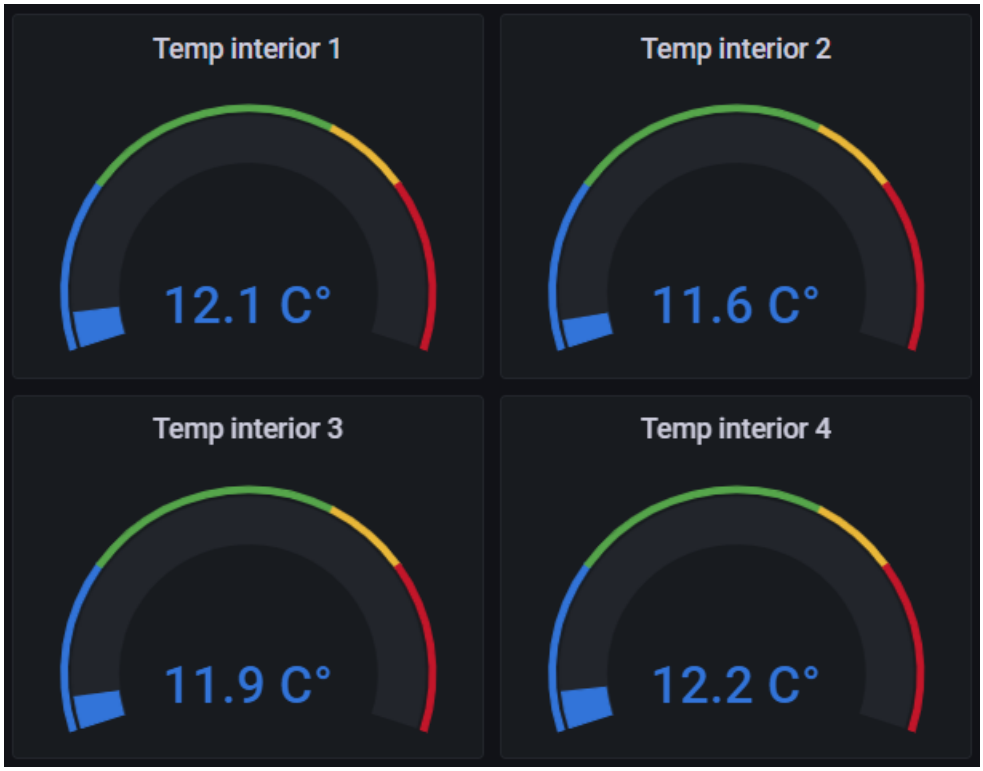


Figura 112. Temperaturas internas de noche.

En el análisis final, es relevante comparar el promedio de las temperaturas internas con las temperaturas externas. Se observa un cambio significativo durante el día, con una diferencia de más de 10 grados Celsius entre el interior y el exterior del invernadero. Este dato resalta la importancia de controlar y regular las variables de temperatura dentro del ambiente controlado del invernadero. A medida que avanza la noche, se observa una estabilización en un mismo rango de temperatura tanto en

el interior como en el exterior. Esta convergencia nocturna se puede apreciar claramente en la Figura 113 y Figura 114, donde se representan gráficamente las mediciones. Este hallazgo brinda la oportunidad de ajustar y controlar de manera más efectiva las variables de temperatura dentro del invernadero durante las horas nocturnas y reducirlas en las horas diurnas.

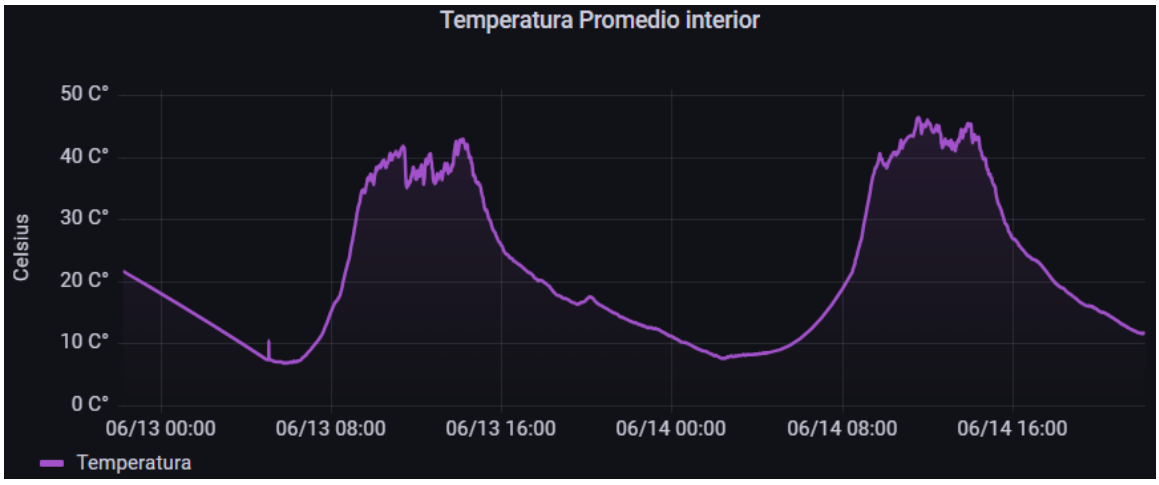


Figura 113. Mediciones de temperatura interna promedio.

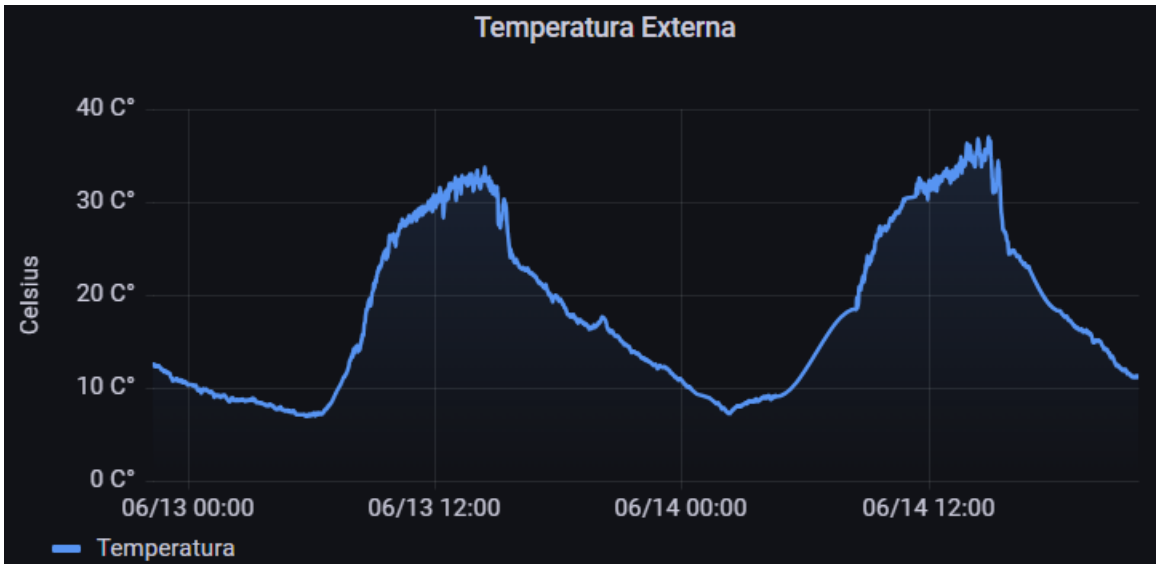


Figura 114. mediciones de temperatura externa.

Estos hallazgos subrayan la importancia de un control preciso de las variables de temperatura dentro del invernadero. El conocimiento de las diferencias entre las temperaturas internas y externas en diferentes momentos del día brinda oportunidades para optimizar los sistemas de calefacción, ventilación y aislamiento térmico. Esto permitirá mantener condiciones ideales para el crecimiento de las plantas y maximizar la eficiencia energética en el invernadero.

4.2 Rendimiento de AWS al sistema

Utilizando IoT Core como intermediario de control y monitoreo, junto con servicios como Lambda, DynamoDB, Timestream y Grafana, se logró evaluar el rendimiento de AWS en el sistema. Esta arquitectura basada en la nube proporcionó una solución escalable y eficiente para el control y monitoreo de los dispositivos IoT, así como el almacenamiento, análisis y visualización de los datos generados. Los resultados obtenidos a través de esta implementación permitieron obtener una visión clara del rendimiento del sistema y posibilitaron tomar decisiones informadas para su mejora continua.

4.2.1 Control

En la parte de control, se puede observar un flujo de información eficiente desde IoT Core hacia Lambda para la ejecución de las funciones correspondientes. Esta integración sin problemas entre los servicios de AWS asegura que los datos sean enviados y procesados de manera efectiva, permitiendo una respuesta rápida y precisa del sistema. Utilizando el monitoreo proporcionado por AWS CloudWatch, se puede evaluar la frecuencia de activación de cada función a lo largo del día.

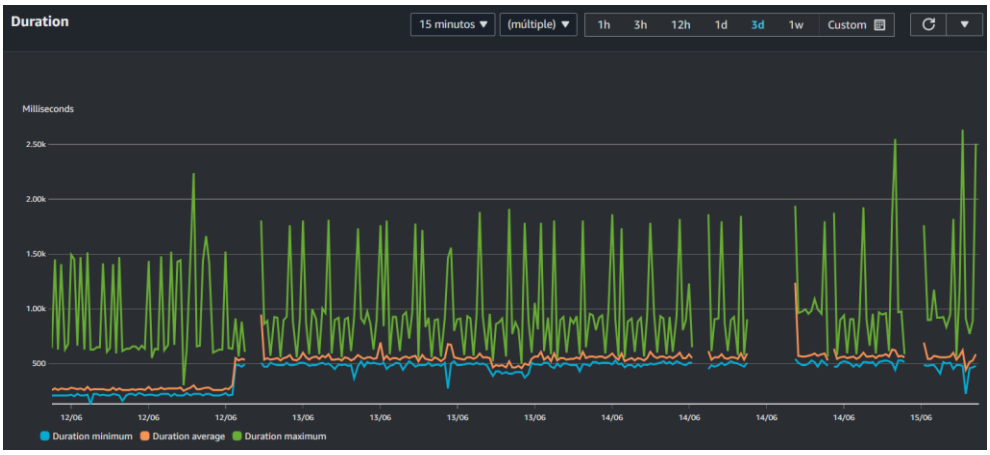


Figura 115. Duración de ejecución en AWS Lambda para "Humedad Data".

Además de evaluar la frecuencia de activación, también es importante analizar el tiempo de ejecución de las funciones Lambda. En la Figura 115, se puede ver que el tiempo de ejecución varía entre un máximo de 2500 y un mínimo de 230 milisegundos, a diferencia de la gráfica de la Figura 116 donde su rango de ejecución varia de los 2170 a los 133 milisegundos. Esta variabilidad puede atribuirse a diversos factores, como la carga del sistema, la complejidad de la función y la cantidad de datos procesados. Es importante tener en cuenta que estos tiempos de ejecución se encuentran dentro de los límites permitidos y, dado que se encuentra en la capa gratuita, no representa ningún problema para el uso de esta función.

Figura 116. Duración de ejecución en AWS Lambda para "TempsData".

La fluctuación en el tiempo de ejecución puede deberse a la naturaleza dinámica del sistema, donde la cantidad de datos y las tareas a realizar varia en diferentes momentos del día. Además, el uso de recursos compartidos en el entorno de ejecución de Lambda también influye en la variabilidad observada. Sin embargo, es importante destacar que estos tiempos de ejecución están dentro de rangos aceptables y no afectan negativamente el rendimiento general del sistema.

Por otra parte, es importante destacar el rendimiento y la eficacia del servicio DynamoDB en el sistema. Desde su creación de las tablas, se ha estado recopilando información de manera continua. A lo largo de aproximadamente 7 semanas de ejecución, la tabla "Rasp_Temperaturas" ha acumulado un total de 25,270 registros, con un tamaño de aproximadamente 2 megabytes. Por otro lado, la tabla "Rasp_Humedad" ha registrado 20,093 registros, ocupando 1.2 megabytes de espacio. Estos datos demuestran que, a pesar de tener el mismo tiempo de ejecución, existe una variación en la cantidad de datos enviados, aunque individualmente sean muy pequeños.

Uno de los aspectos significativos al analizar DynamoDB es la latencia en la escritura de datos. Mediante el análisis de las gráficas correspondientes (Figura 117 y Figura 118), se observa que la latencia se mantiene en un rango de 30 a 15 milisegundos. Esto indica que existe una respuesta rápida y eficiente por parte de Lambda al interactuar con DynamoDB para almacenar los datos recopilados. La baja latencia en la escritura asegura una alta disponibilidad y capacidad de respuesta del sistema, lo cual es crucial para mantener un flujo continuo de información y garantizar una operación fluida.

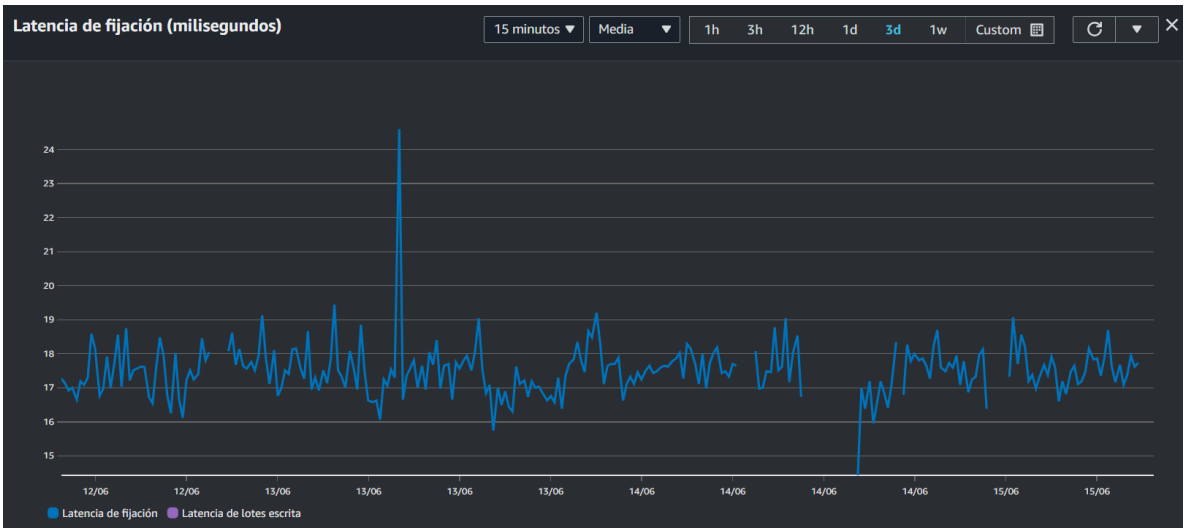


Figura 117. Gráfico de latencia en la escritura de datos para la tabla "Rasp_Temperaturas".

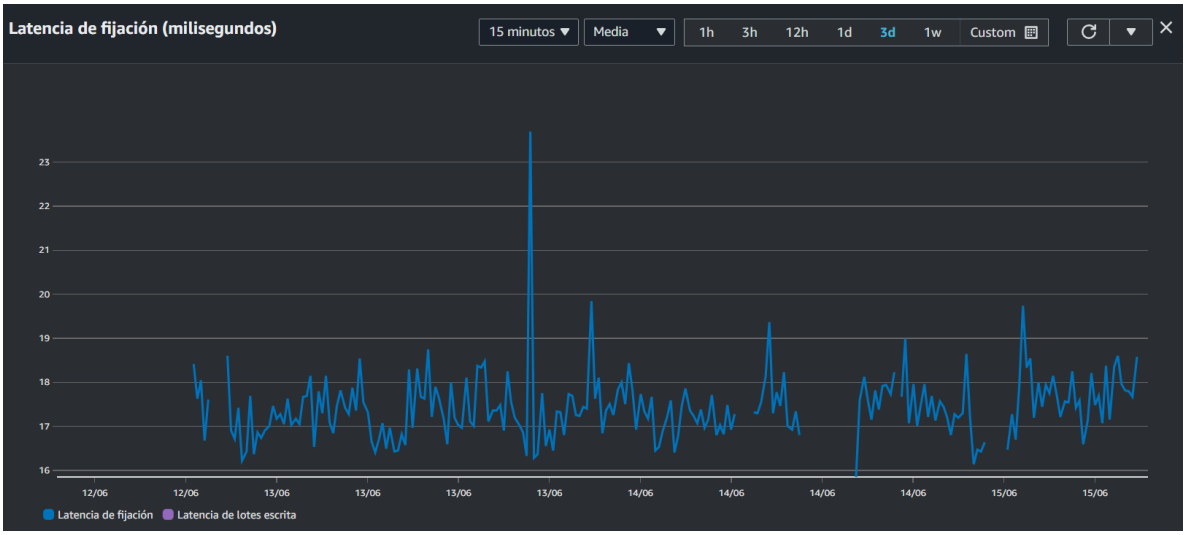


Figura 118. Gráfico de latencia en la escritura de datos para la tabla "Rasp_Humedad".

La eficacia de DynamoDB en la gestión y almacenamiento de datos es evidente, ya que ha sido capaz de manejar un volumen considerable de registros sin afectar significativamente el rendimiento del sistema. La estructura altamente escalable y flexible de DynamoDB permitiendo gestionar estas cantidades de información de manera eficiente, adaptándose a las necesidades cambiantes del sistema.

4.2.2 Monitoreo

En la parte del monitoreo, se observan los resultados obtenidos en AWS TimeStream utilizando AWS CloudWatch para evaluar la latencia de ingestión P95. La latencia de ingestión P95 es una medida en milisegundos que resulta fundamental para evaluar la capacidad de respuesta y eficiencia del servicio al procesar y almacenar puntos de datos. Una latencia P95 baja indica un rendimiento óptimo en términos de velocidad de ingestión, lo que permite una toma de decisiones y análisis de datos más rápidos en aplicaciones en tiempo real.

Al analizar los resultados en la base de datos "TempsDB", se puede observar la latencia de ingreso de datos en un rango máximo de aproximadamente 120 milisegundos, mientras que la latencia mínima se sitúa en alrededor de 35 milisegundos. Estos datos se representan gráficamente en la Figura 119, donde cada color representa una variable de las tablas, "Interior_1", "Interior_2", "Interior_3", "Interior_4", "Interior_promedio" y "Temperatura_Exterior". Estas variables reflejan las mediciones de temperatura registradas en el sistema. Brindando una visión clara de la latencia de ingreso de datos en cada una de estas variables y permite realizar un seguimiento detallado de su rendimiento.

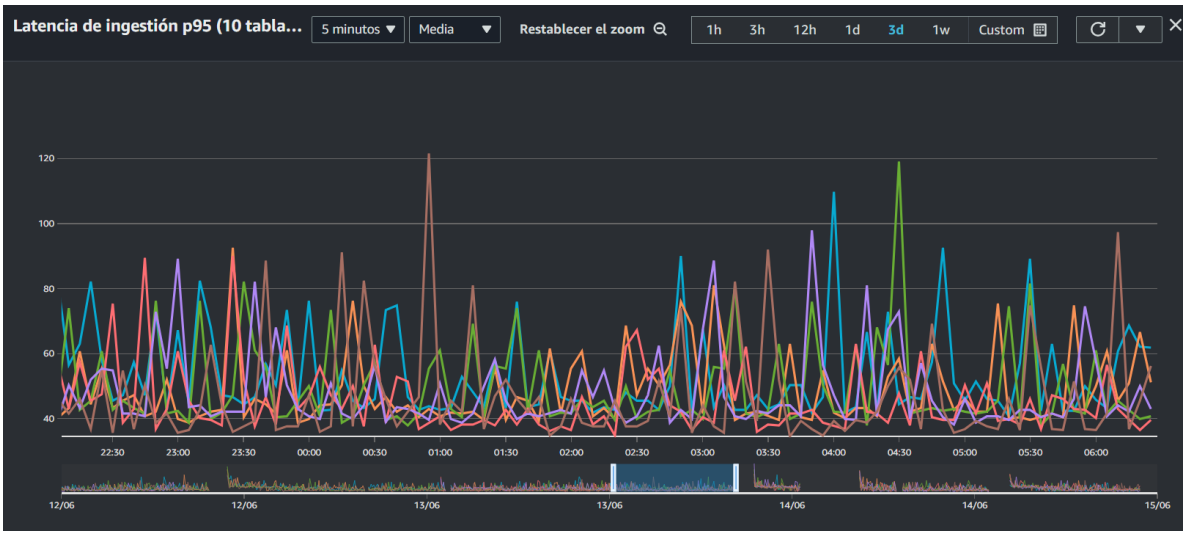


Figura 119 Latencia de Ingestión P95 de Temperaturas Registradas en el Sistema.

De manera similar, en la base de datos "HumedadDB", se observan resultados similares en cuanto a la latencia de ingreso de datos. Aquí, también se encuentran rangos de latencia máxima alrededor de 120 milisegundos y latencia mínima de aproximadamente 35 milisegundos. La Figura 120 muestra esta información en forma gráfica, representando las cuatro tablas de la base de datos, denotadas como "H_1", "H_2", "H_3" y "H_4". Estas tablas contienen las mediciones de humedad registradas en el sistema. Al observar la gráfica, se puede analizar y comparar fácilmente la latencia de ingreso de datos en cada una de estas variables.

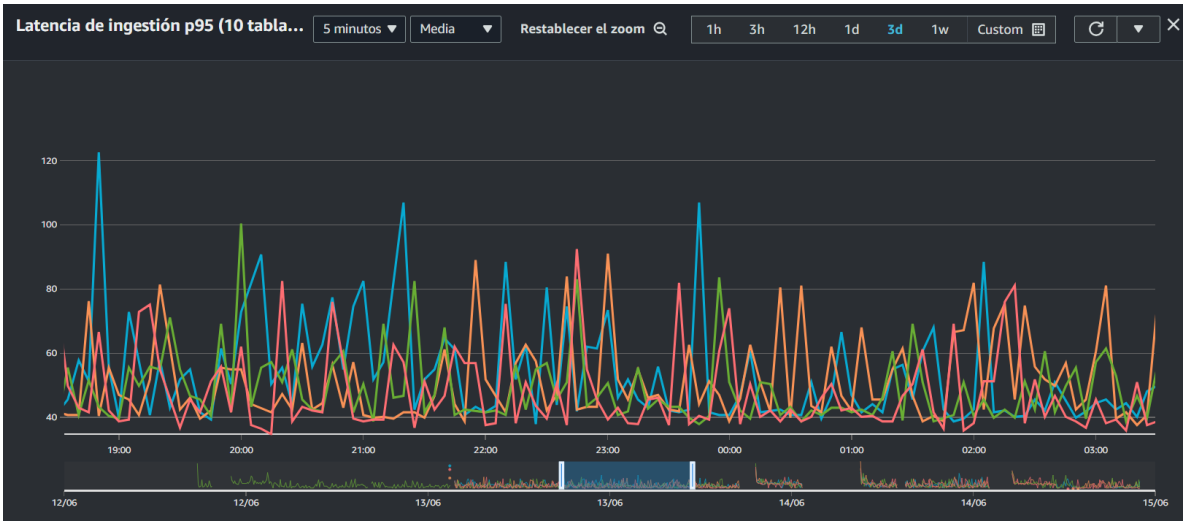


Figura 120. Latencia de Ingestión P95 de Humedad Registrada en el Sistema.

Los resultados obtenidos indican que la latencia de ingestión P95 en ambas bases de datos se encuentra dentro de un rango aceptable. Esto significa que el servicio proporcionado por AWS TimeStream desde IoT Core presenta una respuesta rápida y eficiente en la ingestión de datos. La capacidad de recibir y procesar los puntos de datos en un tiempo mínimo permite un monitoreo en tiempo real y una toma de decisiones ágil en aplicaciones que dependen de estos datos.

En cuanto a la visualización de los datos recopilados desde AWS TimeStream, Grafana ha demostrado ser una herramienta eficiente al presentarlos de manera clara y completa. En la Figura 121, se muestra el panel completo que brinda una vista general de los diferentes aspectos monitoreados, como las temperaturas internas, la temperatura exterior, la humedad del suelo en cada planta como la calidad del aire. Esta visualización en forma de panel proporciona una representación gráfica intuitiva y fácil de interpretar de los datos registrados.

Figura 121. Panel de Monitoreo del invernadero por AWS Grafana.

4.2.3 Costo de los servicios

Algo sumamente relevante que merece ser resaltado es el costo de los servicios de AWS una vez se superan los límites de uso gratuito. En este caso, al revisar detenidamente la factura correspondiente al mes de mayo de 2023, se puede apreciar que el costo incurrido es relativamente mínimo, ascendiendo a tan solo $0.09 dólares, lo cual equivale aproximadamente a $1.54 pesos mexicanos, considerando el tipo de cambio actual.

Si se observa la Tabla 10 adjunta, se nota que los gastos más significativos corresponden al servicio de AWS TimeStream, mientras que el gasto más insignificante se encuentra en el servicio de IoT Core. Por otro lado, es importante mencionar que los servicios de Lambda, DynamoDB y Grafana se mantienen dentro del rango de uso gratuito, lo que significa que no han generado costos adicionales en esta ocasión.

Tabla 10. factura del mes de mayo del 2023.

Servicio	Descripción de servicio	Cantidad de uso	Importe en USD
TimeStream	Amazon Timestream USW2-DataIngestion-Bytes	0.134 GB	0,07 USD
TimeStream	Amazon Timestream USW2-DataScanned-Bytes	0.913 GB	0,01 USD
TimeStream	Amazon Timestream USW2-MemoryStore-ByteHrs	0.064 GB-Hours	0,00 USD
IoT Device Defender	$1.10 per thousand devices	5 Devices	0,01 USD
CloudWatch	Amazon CloudWatch	8 Alarms	0,00 USD
DynamoDB	$0.00 per hour for 25 units of read capacity for a month (free tier)	2232 ReadCapacityUnit-Hrs	0,00 USD
IoT	AWS IoT USW2-ActionsExecuted	192,656 ActionsExecuted	0,00 USD

IoT	AWS IoT USW2-ConnectionMinutes	19,757 Minutes	0,00 USD
	AWS IoT USW2-Messages	50,988 Messages	0,00 USD
Lambda	AWS Lambda USW2-Lambda-GB-Second	1,100.037 Second	0,00 USD
Grafana	mazon Managed Grafana EU-Grafana:FreeTrialUser	1 Users	0,00 USD

En comparación con la factura del mes de mayo de 2023 que fue mencionada anteriormente, la factura correspondiente al mes de abril del mismo año muestra una diferencia notable. Durante ese período, debido a que se encontraban en la fase de pruebas de los servicios, los costos se elevaron a $4.90 dólares, lo que equivale a aproximadamente $84.06 pesos mexicanos al tipo de cambio actual. En este caso, se estaban evaluando múltiples opciones de graficación y control de variables. Dentro de las aplicaciones de gasto en esta fase de pruebas, destaca el servicio de IoT SiteWise, que generó un costo de $2.25 dólares, convirtiéndose así en el servicio más caro considerado durante estas evaluaciones. Sin embargo, debido a su elevado costo, se decidió no utilizarlo en la implementación final del proyecto. Para obtener una visión más detallada de los costos y servicios evaluados durante esta fase, mostrada en la Tabla 11 adjunta.

Tabla 11. Factura del mes de abril del 2023.

Servicio	Descripción de servicio	Cantidad de uso	Importe en USD
IoT SiteWise	AWS IoT SiteWise USW2-AWSIoTSiteWise-Computations	2,387,595 Computations	1,19 USD

IoT SiteWise	AWS IoT SiteWise USW2-AWSIoTSiteWise-Messaging	1,322,474 Messages	1,32 USD
IoT SiteWise	AWS IoT SiteWise USW2-AWSIoTSiteWise-Storage	21.804 GB-Hours	0,01 USD
IoT Events	AWS IoT Events EU-Alarms	1 Alarms	0,10 USD
IoT Events	AWS IoT Events USW2-Alarms	8 Alarms	0,80 USD
IoT Events	AWS IoT Events USW2-Messages	2793 Messages	0,04 USD
TimeStream	Amazon Timestream USW2-DataIngestion-Bytes	0.064 GB	0,03 USD
TimeStream	Amazon Timestream USW2-DataScanned-Bytes	71.544 GB	0,72 USD
TimeStream	Amazon Timestream USW2-MemoryStore-ByteHrs	0.05 GB-Hours	0,00 USD
CloudWatch	$0.10 per alarm metric month (standard resolution)	3,465 Alarms	0,35 USD
CloudWatch	$0.50 per rule per month for CW-Rule-DynamoDB	0.546 Rule-Hour	0,27 USD
IoT Device Defender	$1.10 per thousand devices	2 Devices	0,00 USD
SageMaker	Amazon SageMaker CreateVolume-Gp2	0.276 GB-Mo	0,04 USD

DynamoDB	$0.00 per hour for 25 units of read capacity for a month (free tier)	1572 ReadCapacityUnit-Hrs	0,00 USD
IoT	AWS IoT USW2-ActionsExecuted	250,000 ActionsExecuted	0,00 USD
IoT	AWS IoT USW2-ConnectionMinutes	21,665 Minutes	0,00 USD
IoT	AWS IoT USW2-Messages	183,628 Messages	0,00 USD
Lambda	AWS Lambda USW2-Lambda-GB-Second	95,749.812 Second	0,00 USD
Grafana	mazon Managed Grafana EU-Grafana:FreeTrialUser	1 Users	0,00 USD

Es importante tener en cuenta que, durante las etapas de pruebas y evaluaciones, los costos pueden ser más altos debido a la exploración de diversas opciones y configuraciones. A medida que el proyecto avanza y se definen las soluciones más adecuadas, se espera que los costos se optimicen y se mantengan dentro de un rango más favorable en futuras facturas. Durante el proceso de optimización de los recursos de cada servicio, se realizaron mejoras significativas en el uso de AWS TimeStream. Inicialmente, se tenía una única tabla que contenía todos los valores de las variables, lo que generaba un costo adicional al consultar todas las variables al mismo tiempo. Para resolver esta situación, se implementó una separación de tablas por variable, lo que permitió reducir los costos asociados a las consultas. Al realizar esta optimización en la estructura de la base de datos, se logró minimizar el tiempo y los recursos necesarios para acceder a los datos específicos de cada variable en AWS TimeStream. Como resultado, se obtuvo un rendimiento más eficiente y se redujo el costo general del servicio. Estas acciones demuestran la

importancia de realizar una evaluación continua de los recursos utilizados y aplicar ajustes en función de los resultados obtenidos. Al optimizar los recursos de cada servicio, se logra maximizar su eficiencia y, al mismo tiempo, se reduce el impacto en el costo total del proyecto.

En resumen, el sistema implementado para el control del crecimiento de los cultivos ha demostrado una efectividad notable en su funcionamiento. Gracias a su capacidad para mantener condiciones óptimas de humedad del suelo, calidad del aire y temperatura, se ha logrado un manejo eficiente de los cultivos y se han obtenido resultados satisfactorios. La utilización de tecnologías como AWS (Amazon Web Services) y Raspberry Pi ha sido fundamental en el éxito de esta solución. Estas herramientas han permitido no solo el monitoreo en tiempo real de las condiciones ambientales, sino también un control preciso y automatizado de los parámetros necesarios para el desarrollo saludable de las plantas.

Además, se pudo validar que la hipótesis planteada, la cual se buscaba lograr un crecimiento óptimo de los cultivos en un invernadero tipo túnel mediante la integración de tecnologías web y sistemas embebidos. La implementación de estas tecnologías ha permitido mantener las condiciones apropiadas dentro del invernadero, lo cual ha resultado en un correcto desarrollo de los cultivos. El monitoreo y control preciso de las condiciones ambientales ha contribuido al crecimiento saludable de las plantas, confirmando así la efectividad de la hipótesis inicialmente propuesta. El uso de AWS ha brindado una plataforma sólida y escalable para el almacenamiento y procesamiento de los datos recopilados por el sistema. La capacidad de esta infraestructura en la nube ha permitido gestionar grandes volúmenes de información de manera eficiente, asegurando la disponibilidad y seguridad de los datos en todo momento. Además, la flexibilidad de AWS ha permitido adaptar y ampliar el sistema de acuerdo con las necesidades cambiantes del cultivo, garantizando su funcionamiento óptimo en todo momento.

Por otro lado, Raspberry Pi ha desempeñado un papel crucial como dispositivo de control y comunicación en el sistema. Gracias a su capacidad de procesamiento y conectividad, se ha logrado un control eficiente de los diferentes actuadores y sensores del sistema. Raspberry Pi ha demostrado ser una solución rentable y versátil, que se puede adaptar fácilmente a las necesidades específicas de cualquier proyecto agrícola. Además de su eficiencia técnica, la implementación de este sistema ha brindado beneficios económicos y ambientales significativos. Al mantener condiciones óptimas de humedad y el sondeo de la temperatura, se ha logrado un uso más eficiente de los recursos hídricos y energéticos, reduciendo el desperdicio y el impacto ambiental asociado. Esto se traduce en una mayor productividad y rentabilidad para los agricultores, así como en una producción agrícola más sostenible y respetuosa con el medio ambiente.

CAPÍTULO V CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

Las conclusiones destacan la eficacia del sistema implementado en el invernadero para controlar variables ambientales clave, la eficiencia de DynamoDB como base de datos, la efectividad de AWS y Raspberry Pi para el control y monitoreo de cultivos, la utilidad de Grafana para la visualización de datos y el potencial de reducción de costos a través de mejoras continuas.

El sistema implementado en el invernadero ha demostrado ser eficaz en el control de variables ambientales clave, como la temperatura, la humedad del suelo y la calidad del aire. La utilización de sensores y sistemas de control ha permitido mantener condiciones óptimas para el crecimiento de las plantas, lo cual es fundamental para obtener buenos resultados en la producción agrícola.

La implementación de DynamoDB como base de datos ha sido exitosa, proporcionando una solución escalable y confiable para el almacenamiento y recuperación de datos generados por el sistema. La baja latencia en la escritura de datos y la capacidad para manejar grandes volúmenes de registros han contribuido al rendimiento general del sistema.

El uso de AWS y Raspberry Pi ha demostrado ser una combinación efectiva para el control y monitoreo de los cultivos. Estas tecnologías han permitido mantener condiciones óptimas de humedad, calidad del aire, así como su sondeo en temperatura, lo cual es fundamental para el éxito de la agricultura moderna.

La visualización de datos a través de Grafana ha brindado una comprensión clara y detallada del rendimiento del invernadero en términos de control de temperatura y humedad. Esta herramienta ha facilitado la toma de decisiones informadas y ha optimizado el rendimiento del sistema.

A medida que se realizan mejoras y ajustes en los servicios utilizados, como en el caso de AWS TimeStream, se puede lograr una reducción significativa en los costos asociados. Esto es beneficioso para el desarrollo y éxito del proyecto a largo plazo.

5.2 Recomendaciones

Lo que se recomienda es continuar mejorando el sistema de control y monitoreo implementado en el invernadero, considerando aspectos como la temperatura, humedad, calidad del aire y otros factores ambientales relevantes. Esto permitirá optimizar el crecimiento de las plantas, maximizar la eficiencia energética y mejorar los resultados generales del proyecto. Algunas recomendaciones son:

- Para optimizar el control de la temperatura interna del invernadero, se sugiere considerar la implementación de sistemas de ventilación o refrigeración adicionales. Esto ayudará a reducir las variaciones de temperatura durante el día y mantener un ambiente más estable para el crecimiento de las plantas.

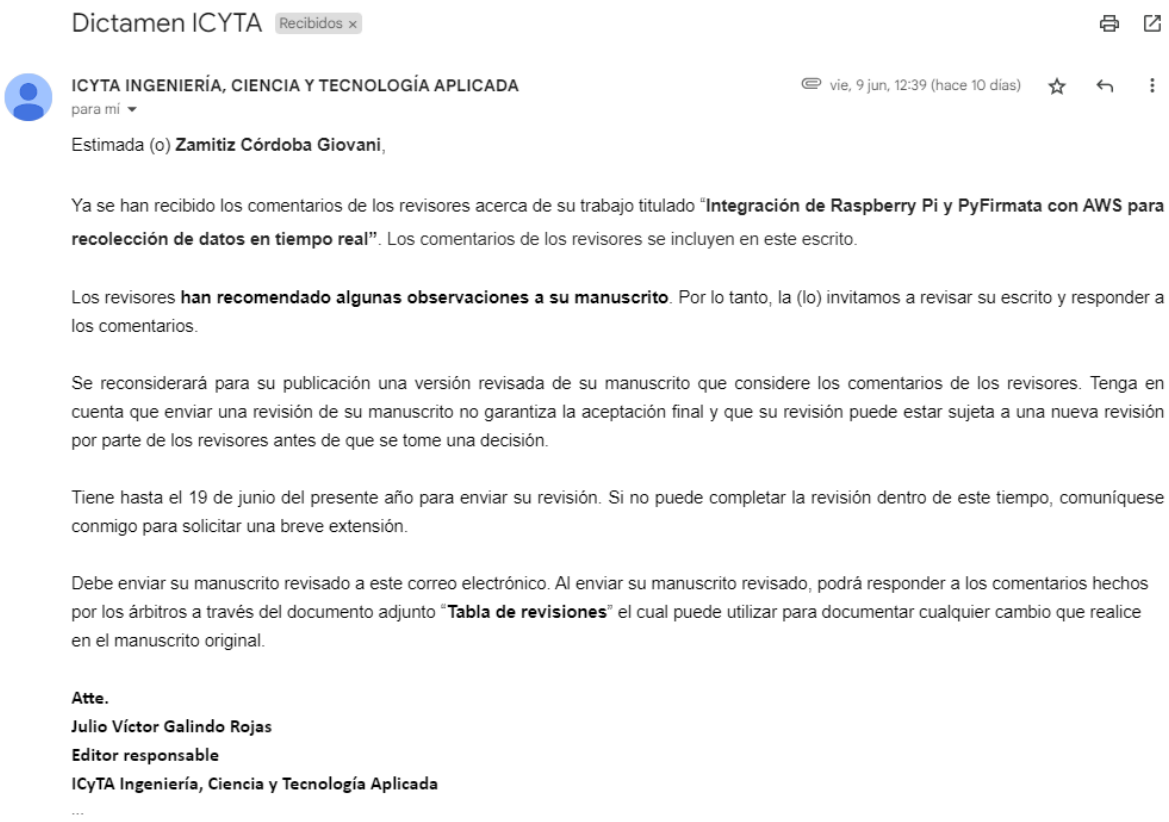
- Es importante realizar un análisis más detallado de las lecturas de los sensores de calidad del aire para identificar posibles fuentes de contaminación y tomar medidas preventivas. Esto garantizará un ambiente saludable para las plantas y minimizará los riesgos de enfermedades o plagas.

- Considerar la integración de otros sistemas de monitoreo, como el monitoreo de la luz y los niveles de dióxido de carbono, para obtener una imagen más completa de las condiciones ambientales dentro del invernadero y tomar decisiones basadas en datos más precisos.

- Explorar la posibilidad de utilizar técnicas de análisis de datos y aprendizaje automático para detectar patrones y tendencias en los datos recopilados. Esto puede proporcionar información adicional sobre el rendimiento del invernadero y ayudar en la toma de decisiones estratégicas.

PRODUCTOS ACADÉMICOS

Anexando el envió de articulo a la revista ICYTA esperando a la respuesta de aceptación. Donde se aran las correcciones solicitadas para ser candidato a la aceptación de la publicación.



Dictamen ICYTA Recibidos ×

ICYTA INGENIERÍA, CIENCIA Y TECNOLOGÍA APLICADA vie, 9 jun, 12:39 (hace 10 días)

para mí

Estimada (o) **Zamitiz Córdoba Giovani**,

Ya se han recibido los comentarios de los revisores acerca de su trabajo titulado "**Integración de Raspberry Pi y PyFirmata con AWS para recolección de datos en tiempo real"**. Los comentarios de los revisores se incluyen en este escrito.

Los revisores **han recomendado algunas observaciones a su manuscrito**. Por lo tanto, la (lo) invitamos a revisar su escrito y responder a los comentarios.

Se reconsiderará para su publicación una versión revisada de su manuscrito que considere los comentarios de los revisores. Tenga en cuenta que enviar una revisión de su manuscrito no garantiza la aceptación final y que su revisión puede estar sujeta a una nueva revisión por parte de los revisores antes de que se tome una decisión.

Tiene hasta el 19 de junio del presente año para enviar su revisión. Si no puede completar la revisión dentro de este tiempo, comuníquese conmigo para solicitar una breve extensión.

Debe enviar su manuscrito revisado a este correo electrónico. Al enviar su manuscrito revisado, podrá responder a los comentarios hechos por los árbitros a través del documento adjunto "**Tabla de revisiones**" el cual puede utilizar para documentar cualquier cambio que realice en el manuscrito original.

Atte.
Julio Víctor Galindo Rojas
Editor responsable
ICyTA Ingeniería, Ciencia y Tecnología Aplicada
...

REFERENCIAS

alldatasheet. (n.d.). *DHT11 Humidity & Temperature Sensor*. Retrieved March 7, 2023, from https://pdf1.alldatasheet.com/datasheet-pdf/view/1440068/ETC/DHT11.html

Amazon. (2022a). *AWS IoT Core: Guía para desarrolladores*. https://docs.aws.amazon.com/es_es/iot/latest/developerguide/iot-dg.pdf#what-is-aws-iot

Amazon. (2022b). *Overview of Amazon Web Services*. https://docs.aws.amazon.com/whitepapers/latest/aws-overview/aws-overview.pdf

ams AG. (2016). *CCS811 Ultra-Low Power Digital Gas Sensor for Monitoring Indoor Air Quality*. https://cdn.sparkfun.com/assets/learn_tutorials/1/4/3/CCS811_Datasheet-DS000459.pdf

Analog Devices inc. (2002). *Low Voltage Temperature Sensors TMP35/TMP36/TMP37*. www.analog.com

Anaya, D. A. (2020). *PROTOTIPO INVERNADERO AUTOMATIZADO INDOOR 2*. Universidad de la Costa.

Arevalo, P., Jesus, M. E., & Soria, E. H. (2009). *Sistemas Embebidos Wireless surveillance of elect ronic paramet ers t hrough T CP/IP prot ocol by means of WiFi*.

Arnold, M., & Osorio, F. (1998). *Introducción a los Conceptos Básicos de la Teoría General de Sistemas*.

Ascon Tecnologic s.r.l. (2020). *RELATIVE HUMIDITY AND HUMIDITY AND TEMPERATURE TRANSMITTERS • WALL MOUNTING*. TRH200. https://www.ascontecnologic.com/prodotto/trh200/?lang=es#toggle-id-2

Becas Santander. (2022). *¿Qué es Python?* https://www.becas-santander.com/es/blog/python-que-es.html

Blanco Rico, G., & Martínez Zarzuela. (2021). *Sistema de telemedida y sensorización mediante red LoRaWAN*.

Bolton, W. (2017). *Mecatrónica. Sistemas de control electrónico en la ingeniería mecánica y eléctrica. Un enfoque multidisciplinario.* (6th ed.).

BOSCH. (2017). *BME680 Low power gas, pressure, temperature & humidity sensor*. https://cdn-shop.adafruit.com/product-files/3660/BME680.pdf

Carrillo, E., Meza, J., & Hernandez, F. A. (2022). *Propuesta Técnica de un control en un invernadero industrial tipo multitúnel y monitoreo de variables a través de un sistema IOT*.

Decagon Devices. (2012). *EC-5 Soil Moisture Sensor User's Manual*. https://www.ictinternational.com/content/uploads/2014/03/EC-5-Manual-v2.pdf

Delgadillo-Gaytan, R. (2019). *Desarrollo de una Red de Sensores para el Monitoreo en Ambiente Web de Parámetros Físico-Químicos en Invernaderos de Plantas Ornamentales*. Instituto Tegnológico de Colima.

Delta-T Devices Ltd. (2006). *User Manual for the SM200 Soil Moisture Sensor*. www.delta-t.co.uk

El tutorial de Python — documentación de Python - 3.11.0. (n.d.). Retrieved December 1, 2022, from https://docs.python.org/es/3/tutorial/index.html

Fundación Produce Sinaloa, A. C. (2006). *Produccion de hortalizas bajo invernadero*.

García S, G. (2020). *TECNIFICACIÓN DE INVERNADEROS PARA LA PRODUCCIÓN DE JITOMATE (Solanum lycopersicum L.), EN TETELA DE OCAMPO PUEBLA, MÉXICO.* Benemérita Universidad Autónoma De Puebla.

Gómez, O. Y. (2020). *invernaderos y sistemas de riego en México*.

IBM. (2022, October 13). *Servicios web*. https://www.ibm.com/docs/es/was/9.0.5?topic=services-web

Jorge Arcenio, Q. P. (2021). *Sistema para el manejo de datos climáticos de pequeñas producciones agrícolas bajo invernadero: un acercamiento a la agricultura inteligente*. Universidad Nacional de Colombia.

López, J. R. (2017, October 18). *La importancia de la luz en las plantas*. https://www.catalunyaplants.com/la-importancia-de-la-luz-en-las-plantas/

Miyara, F. (2004). *CONVERSORES D/A Y A/D*. http://www.fceia.unr.edu.ar/enica3

Ortiz-Aguilar, L., Hernández-Silva, L., Muñoz-López, B., & Cortes-Ruiz, A. (2022). Diseño IoT de invernadero para el control de variables mediante técnicas de inteligencia artificial. *Research in Computing Science*, *151*(6), 173–186.

OXDEA. (2020). *Sensor de Humedad de Suelo-Higrómetro Anticorrosivo HD-38*. Https://Oxdea.Gt/Product/Sensor-de-Humedad-de-Suelo-Higrometro-Anticorrosivo-Hd-38/.

Pallás A, R. (2003). *Sensores y acondicionadores de señal* (S. A. MARCOMBO, Ed.; 4th ed.).

Paredes-Valverde, M. A., Machorro-Cano, I., Alor-Hernández, G., Rodríguez-Mazahua, L., Sánchez-Cervantes, J. L., & Olmedo-Aguirre, J. O. (2020). HEMS-IoT: A big data and machine learning-based smart home system for energy saving. *Energies*, *13*(5). https://doi.org/10.3390/en13051097

PLANTOWER. (2016). *Digital universal particle concentration sensor PMS5003 series data manual*. https://www.aqmd.gov/docs/default-source/aq-spec/resources-page/plantower-pms5003-manual_v2-3.pdf

Raspberry Pi. (2019). *Raspberry Pi 4 Tech Specs*. Https://Www.Raspberrypi.Com/Products/Raspberry-Pi-4-Model-b/Specifications/.

Real academia española. (2021). *medir*. Diccionario de La Lengua Española. https://dle.rae.es/medir

Real academia Española. (2021). *sistema*. Diccionario de La Lengua Española. https://dle.rae.es/sistema

Reidl-Martínez, L. M. (2012). Marco conceptual en el proceso de investigación. In *Inv Ed Med* (Vol. 1, Issue 3). www.elsevier.com.mx

Santizo V., H. (2011). *Diseño y construcción de invernaderos para la producción de hortalizas*.

SAP Insights. (2022). *¿Qué es IoT y cómo funciona?* https://www.sap.com/spain/insights/what-is-iot-internet-of-things.html?gclid=Cj0KCQiA4aacBhCUARIsAI55maEWm_01BZ3JT-VEr5FNPDZV5wusgfGwUduQvPxywTEOBFccNvLJcdIaAjMbEALw_wcB&gclsrc=aw.ds

Sension. (2017). *SGP30 Sensirion Gas Platform*. www.sensirion.com

SIAP. (2021). *Anuario Estadístico de la Producción Agrícola*. https://nube.siap.gob.mx/cierreagricola/

SIAP. (2022). *Panorama agroalimentario 2022*.

Texas Instruments. (1999). *LM35 LM35 Precision Centigrade Temperature Sensors*. https://www.ti.com/lit/ds/symlink/lm35.pdf?ts=1685211865231&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FLM35%253Futm_source%253Dgoogle%2526utm_medium%253Dcpc%2526utm_campaign%253Dasc-null-null-GPN_EN-cpc-pf-google-wwe%2526utm_content%253DLM35%2526ds_k%253DLM35%2BDatasheet%2526DCM%253Dyes%2526gclid%253DCjwKCAjw1MajBhAcEiwAagW9MdWkkTE5uPwiXLro0NIBi_vJYiRL1lGOxnNnzwoT3PCoYItVqGlMJxoCPxgQAvD_BwE%2526gclsrc%253Daw.ds

Valdez L. Francisco. (2022). *Sistema De Riego Para Invernaderos Con Interfaz Web Utilizando Un Arduino Yun*.

Vegetronix. (n.d.). *Stop Over-Watering With Soil Moisture Sensors*. VH400 Low-Cost Soil Moisture Sensors. Retrieved May 27, 2023, from https://vegetronix.com/soil-moisture-sensor

Winsen. (2015). *MQ135*. www.winsen-sensor.com

Xi 'an Gavin, E. T. Co. (2008). *DS18B20 Temperature Sensor*. https://www.gaimc.com/Uploads/Download/Temperature/GTS200.pdf

ZIEHL. (n.d.). *Pt100-Temperature-Relay Type TR250*. www.ziehl.de