ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

IMPLEMENTACIÓN DE UN PROTOTIPO DE CONTROL PARA UN CRIADERO DE POLLOS UTILIZANDO UN SOC, UNA APLICACIÓN PARA ANDROID Y UN *DASHBOARD* WEB

IMPLEMENTACIÓN DE UN PROTOTIPO DE SISTEMA DE CONTROL DE LA CALEFACCIÓN DE UN CRIADERO DE POLLOS

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO SUPERIOR EN REDES Y TELECOMUNICACIONES

MELANY DAYANA VITAR CRUZ

DIRECTOR: LEANDRO ANTONIO PAZMIÑO ORTIZ

DMQ, marzo 2024

CERTIFICACIONES

Yo, Melany Dayana Vitar Cruz declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

Melany Dayana Vitar Cruz

melany.vitar@epn.edu.ec

melany24cv@hotmail.com

Certifico que el presente trabajo de integración curricular fue desarrollado por Melany Dayana Vitar Cruz, bajo mi supervisión.

Leandro Antonio Pazmiño Ortiz
DIRECTOR

leandro.pazmino@epn.edu.ec

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

Melany Dayana Vitar Cruz

C.I: 171995230-9

DEDICATORIA

A mis padres, Sandra y Dario, quienes siempre me han apoyado tanto emocional como económicamente durante todo el transcurso de mis estudios, siendo la roca que me mantiene firme a pesar de los problemas y obstáculos que se presentaron.

A mi hermano, Ricardo, que, a pesar de ser menor, me apoyó en lo que podía de forma desinteresada, además de animarme a continuar hasta finalizar mis estudios.

Melany.

AGRADECIMIENTO

A la Escuela Politécnica Nacional y a la Escuela de Formación de Tecnólogos por proporcionarme el entorno académico propicio para explorar y desarrollar este mis conocimientos. Los recursos y la infraestructura de la institución fueron fundamentales para mi desarrollo tanto profesional como personal.

Quiero expresar mi reconocimiento a todos aquellos que, de una u otra manera, contribuyeron con su apoyo y conocimientos, ya sea brindando consejos técnicos o compartiendo experiencias.

Agradezco especialmente a mi director de tesis, el Ing. Leandro Pazmiño, por su orientación experta y su paciencia. Sus conocimientos y dirección fueron esenciales para llevar a cabo este proyecto de manera exitosa.

Melany.

ÍNDICE DE CONTENIDOS

RESUMEN

El presente proyecto aborda el desarrollo de un prototipo controlador de temperatura y humedad en un criadero de pollos. La monitorización de datos se realiza a través de la plataforma creada en *Arduino IoT Cloud* y la aplicación móvil *IoT Remote*. El funcionamiento básico se centra en la lectura de la temperatura y humedad proporcionada por un sensor, activando respuestas específicas en función de condiciones predeterminadas. Cuando la temperatura desciende fuera de un rango establecido, se activa el calefactor, al igual que cuando la humedad disminuye. El ventilador se enciende en caso de que la temperatura aumente o cuando la humedad excede un cierto límite. Los datos y la representación gráfica se visualizan mediante una aplicación *Android* y un *dashboard web*.

En la primera sección, se establecen los objetivos del proyecto junto con el componente desarrollado. Se detallan los conceptos clave necesarios para la implementación del sistema.

La segunda sección describe la metodología seguida para cumplir con los objetivos planteados, proporcionando una secuencia clara en el desarrollo del proyecto.

En la tercera sección, se presentan los resultados obtenidos en el logro de cada objetivo. Se detallan los requisitos del proyecto, la justificación en la selección de hardware y software, el diseño del código junto con la aplicación en *Arduino IoT Cloud.* Además, se incluye la implementación total del sistema en una maqueta que simula un criadero de pollos, seguida de pruebas que demuestran el funcionamiento del prototipo.

Finalmente, se exponen las conclusiones y recomendaciones derivadas del desarrollo del proyecto, así como las fuentes de investigación que respaldaron teóricamente el trabajo. Se incluyen anexos que contienen el video de las pruebas y el código final del sistema.

PALABRAS CLAVE: ESP32, sensor, *Arduino IoT Cloud, IoT Remote,* control de temperatura y humedad.

ABSTRACT

This project focuses on the development of a prototype for regulating temperature and humidity within a chicken hatchery. Data monitoring is facilitated through a platform created in Arduino IoT Cloud and the IoT Remote mobile application. The core functionality revolves around monitoring temperature and humidity levels provided by a sensor, triggering specific responses based on predefined conditions. Activation of the heater occurs when the temperature falls outside a predetermined range, while a decrease in humidity also prompts its activation. Conversely, the fan is activated when the temperature surpasses a certain threshold or when humidity exceeds predefined limits. Data visualization, along with graphical representation, is facilitated through both an Android application and a web dashboard.

The first section of the document delineates the project objectives and the components developed, elucidating key concepts essential for system implementation.

Subsequently, the second section expounds upon the methodology adopted to attain the outlined objectives, providing a systematic progression in the project's development.

In the third section, the results achieved in meeting each objective are presented. This includes detailing project requirements, rationale for the selection of hardware and software, code design, and its application within Arduino IoT Cloud. Furthermore, the complete implementation of the system on a model simulating a chicken hatchery is illustrated, followed by tests showcasing the prototype's functionality.

Finally, conclusions and recommendations stemming from the project's development are outlined, alongside research sources that theoretically underpinned the work. Annexes containing the test video and the final system code are also included.

KEYWORDS: *ESP32, sensor, Arduino IoT Cloud, IoT Remote, temperature and humidity control.*

1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

El monitoreo y control de criaderos de pollos “artesanales” se ha llevado a cabo de manera manual, con información limitada sobre el cuidado adecuado de las aves. Además, mantener las condiciones óptimas resulta laborioso. Sin embargo, gracias al Internet de las cosas (IoT), es posible desarrollar sistemas que permitan un control automático, eficiente y práctico de las condiciones del criadero. Esto se logra mediante la automatización de tareas continuas de monitoreo, utilizando comunicación inalámbrica hacia un servidor y permitiendo la observación del estado a través de una interfaz *web* y una aplicación móvil.

En el presente proyecto, se implementará un prototipo IoT para automatizar tareas como el monitoreo de las condiciones ambientales de un criadero de pollos, como la temperatura y la humedad. A partir de los datos obtenidos por el sensor, se llevarán a cabo ciertas tareas, como el encendido y apagado del ventilador y calentador, según lo establecido en el código. Esto transforma el criadero en un entorno independiente e inteligente. Además, se busca la escalabilidad, por lo que la plataforma *web* utilizada cuenta con características diseñadas para este propósito.

Para lograr este propósito, se seleccionó el SoC ESP32 tras una comparación detallada de las principales características y requisitos necesarios para el prototipo. La placa y los periféricos se vincularon a *Arduino IoT Cloud*, que funcionaría como servidor para la recopilación, procesamiento y representación gráfica de los datos. De esta manera, la visualización de los datos se presenta de forma amigable para el usuario.

La automatización de las acciones se logró al especificar en el código las tareas a realizar en función de ciertas condiciones predefinidas. Para llevar a cabo estas funciones de manera precisa, se emplearon dispositivos activadores para controlar tanto el calentador como el ventilador. Los datos recopilados se pueden observar a través de la plataforma *web* y la aplicación móvil, permitiendo al usuario supervisarlos en cualquier momento y lugar para un monitoreo continuo.

La implementación del sistema se realizó con un enfoque en la eficiencia, considerando la ubicación óptima para los elementos clave como el ventilador, el calefactor y el sensor de temperatura y humedad. Durante las pruebas de funcionamiento, se obtuvieron datos importantes para determinar el correcto funcionamiento del prototipo y que cumpla con los objetivos planteados, simulando las mejores condiciones para las aves.

1.1 Objetivo general

Implementar un prototipo de control para un criadero de pollos utilizando un SoC, una aplicación para Android y un *dashboard* web.

1.2 Objetivos específicos

- Identificar los requerimientos para el diseño del prototipo.
- Seleccionar el hardware y software acorde a los requerimientos establecidos.
- Diseñar el prototipo del sistema de control.
- Implementar el prototipo del sistema de control.
- Realizar pruebas de funcionamiento del prototipo.

1.3 Alcance

El propósito principal de este proyecto es la implementación de un sistema inteligente de control de humedad y temperatura en un criadero de pollos. El sistema se enfocará en el monitoreo y automatización de diversas funciones, como el encendido y apagado de la ventilación y calefacción, tomando decisiones basadas en los datos obtenidos por el sensor ambiental. El diseño del prototipo contendrá las siguientes características específicas:

- Monitoreo de condiciones ambientales como: temperatura y humedad.
- Control del sistema de calefacción en el criadero.
- Monitoreo remoto a través de una aplicación para dispositivos Android.
- Visualización de datos y control a través de un *dashboard web*.

1.4 Marco Teórico

Criadero de Pollos

Un criadero de pollos, entendido como un establecimiento dedicado a la reproducción y cría de aves, desempeña un papel sumamente importante en la producción de carne y huevos. La diversidad de criaderos abarca desde pequeñas granjas locales hasta complejas instalaciones industriales, cada uno con enfoques específicos, ya sea para la obtención de carne, la producción de huevos o la reproducción de aves de corral [1].

Estos criaderos, independientemente de su escala, se encuentran especialmente diseñados para mantener las condiciones ambientales óptimas. La regulación de factores como la temperatura y humedad es esencial para garantizar el bienestar y la máxima productividad de las aves. Mientras que las instalaciones más grandes pueden

requerir de tecnologías industriales para la calefacción y ventilación, las granjas locales pueden implementar métodos de automatización más accesibles y eficientes para su tamaño [1].

Es importante destacar que las condiciones ambientales deben ser adaptadas a la etapa de vida que se encuentren las aves. Desde la cría hasta la fase de producción, los criaderos ajustan meticulosamente las condiciones con el fin de satisfacer las necesidades específicas de cada etapa. Por ejemplo, los pollitos recién nacidos pueden requerir temperaturas mucho más altas y cuidados especiales en comparación con las aves adultas, por lo que las condiciones ambientales se deben adaptarse a ello [1].

SoC: *System on a chip*

Un SoC, o Sistema en un Chip en español, es un circuito integrado que incorpora en su estructura un sistema electrónico completo. Estos chips suelen incluir una unidad central de procesamiento (CPU), puertos de entrada y salida, memoria interna, así como bloques de entrada y salida analógica, entre otras características. Existen varios tamaños y funciones, dependiendo del tipo de chip y las necesidades específicas que se tengan [2].

ESP32 devKit

El ESP32, mostrado en la Figura 1.1, se destaca como un microcontrolador accesible y de bajo consumo energético, siendo una opción versátil y poderosa en el campo de la tecnología. Distinguiéndose especialmente por su conectividad inalámbrica *Wi-Fi* y *Bluetooth* integrada, se ha convertido en un pilar fundamental para una variedad de proyectos del Internet de las cosas (IoT). Equipado con dos núcleos Xtensa LX6 que opera a una frecuencia de reloj de hasta 240 (MHz), el ESP32 ofrece una grna capacidad de procesamiento para diversas aplicaciones [3].

Tiene compatibilidad con una amplia gama de periféricos y sensores, permitiendo una integración flexible en diversos entornos y aplicaciones. Lo notable de este microcontrolador es la posibilidad de programarlo en varios lenguajes populares como Python, C++, JavaScript y Arduino IDE, entre otros, lo que facilita su adaptación a diferentes necesidades y preferencias de desarrollo [3].

Además de su potencia de procesamiento y conectividad, el ESP32 se beneficia de una documentación extensa y un sólido soporte técnico, lo que lo hace atractivo para desarrolladores e investigadores que buscan implementar soluciones tecnológicas avanzadas en el campo de la automatización, sistemas embebidos, control remoto y monitoreo, entre otros campos de aplicación [3].

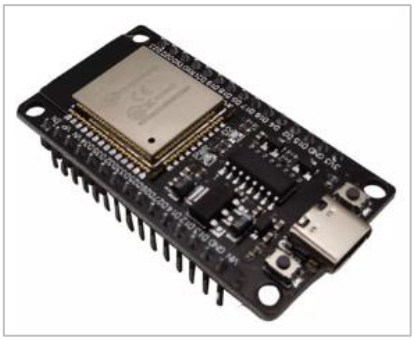

Figura 1.1 Módulo ESP32 *[4]*

Arduino IoT Cloud

Arduino IoT Cloud, ver Figura 1.2*,* es una plataforma diseñada sobre el entorno de *Arduino,* reconocida especialmente por su accesibilidad y facilidad de uso, y se centra en aplicaciones IoT. Su ecosistema no solo permite el monitoreo de sensores, sino que también ofrece una amplia gama de *widgets* que simplifican y mejoran la visualización de datos, facilitando la interacción con los dispositivos conectados [5].

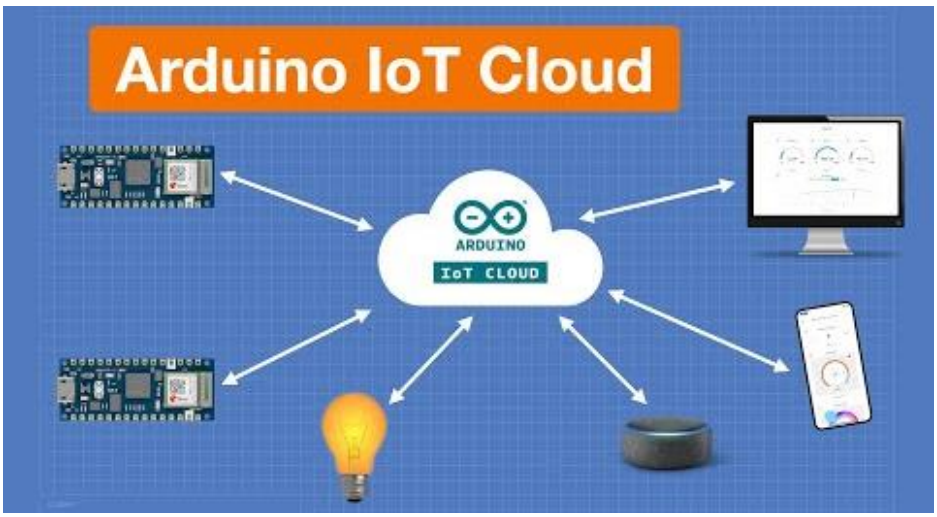

Figura 1.2 Plataforma *Arduino IoT Cloud [5]*

Además, posee la capacidad vincularse con múltiples placas basadas en *Arduino,* lo que facilita a la automatización de periféricos. La plataforma proporciona herramientas sólidas para el registro, análisis y detallada representación gráfica de los datos recolectados por los sensores. La generación automática de la programación al configurar dispositivos u objetos, denominados ‘cosas’ en la plataforma, agiliza considerablemente el proceso de desarrollo y configuración [5].

Una de sus características distintivas radica en la capacidad de lectura y escritura de variables, junto con la detección de eventos, lo que facilita la creación de aplicaciones personalizadas y la implementación de acciones basadas en distintos escenarios. Adicionalmente, se destaca por su enfoque en la seguridad, proporcionando protección

a los dispositivos mediante un sólido sistema de autenticación basada en certificados X.509 [5].

Aplicación *Android*

Las aplicaciones móviles, o también conocidas como *apps*, son programas diseñados específicamente para funcionar en dispositivos móviles como celulares o *tablets*. Gracias a su versatilidad, permite a los usuarios realizar una amplia gama de tareas, ya sean profesionales, educativas, de entretenimiento o para acceder a diversos servicios, de manera ágil y simplificada. Las aplicaciones móviles para *Android* generalmente se encuentran disponibles en la *PlayStore* y pueden ser gratuitas o de pago. A través de estas *apps,* las organizaciones pueden gestionar procesos y aplicaciones usando el *Software* como servicio (SaaS) alojado en la nube, lo que facilita una comunicación efectiva con plataformas web. Esto es especialmente útil en productos combinados, como *Arduino IoT cloud* y *IoT Remote* [6].

El funcionamiento se basa en la conexión entre la aplicación móvil y la nube de *Arduino Cloud,* posteriormente conectándose con la placa como se muestra en la Figura 1.3.

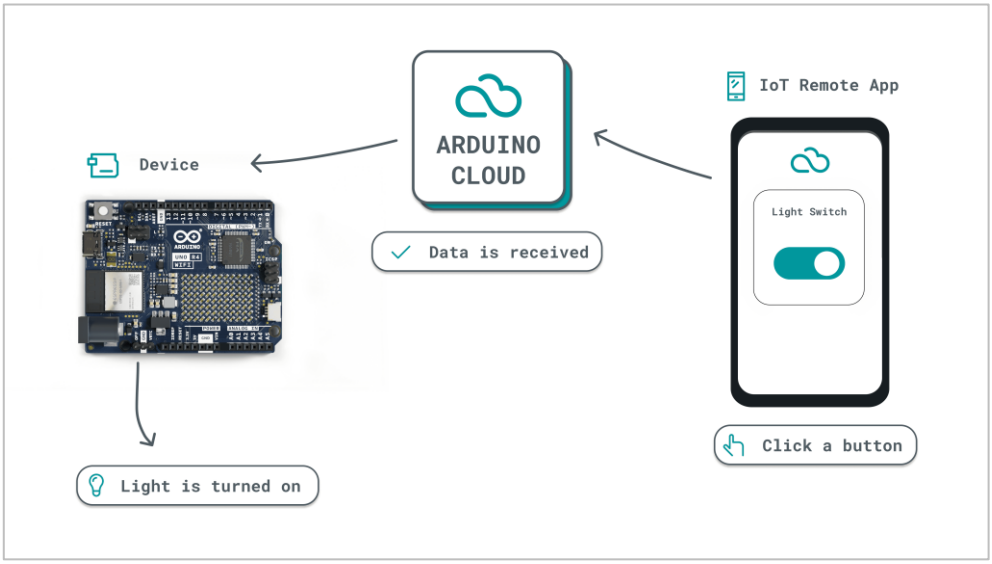

Figura 1.3 Conexión de la aplicación a la placa

Una vez que la placa se comunicó con la nube devuelve los resultados o datos hacia la aplicación móvil como se puede observar en la Figura 1.4.

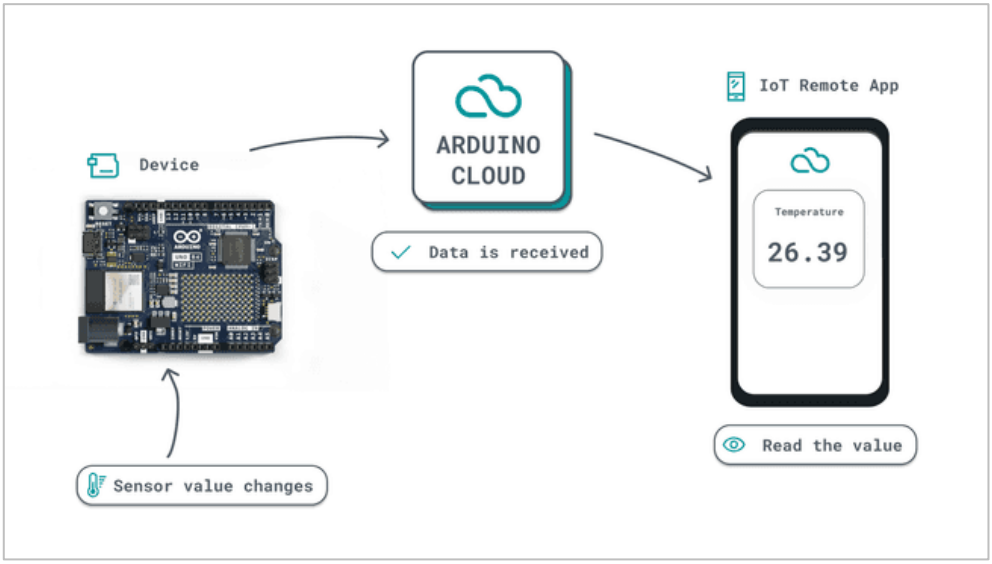

Figura 1.4 Conexión de la placa a la aplicación

Sensor DHT22

El sensor DHT22 es un dispositivo económico y práctico que combina dos sensores sumamente importantes: uno de humedad y otro de temperatura. Su estructura básica consta de tres pines distintivos, cada uno con una función específica, como se muestra en la Figura 1.5. El primer pin corresponde a la conexión a tierra (GND), el segundo pin a la salida de datos, encargado de transmitir la información recopilada al microcontrolador o dispositivo receptor. Finalmente, el tercer pin gestiona la alimentación del sensor, requerida en un rango de voltaje entre 3.3 a 5.5 (V) para asegurar su correcto funcionamiento [7].

Este sensor opera con una tecnología simplificada de un solo bus, lo que implica el uso una única línea de comunicación para el intercambio y control de datos. Esta línea, mediante un protocolo de comunicación eficiente, transmite tanto la información de humadas como de temperatura al sistema receptor. Ambas mediciones son capturadas con una resolución de 16 (bits), lo que garantiza una precisión y nivel de detalle notables en las lecturas obtenidas [7].

Su bajo costo como su tamaño compacto lo hacen ideal para integrarse en sistemas domóticos, facilitando el control inteligente en diferentes entornos. Además, su versatilidad y precisión lo convierten en una herramienta valiosa para el monitoreo ambiental y meteorológico [7].

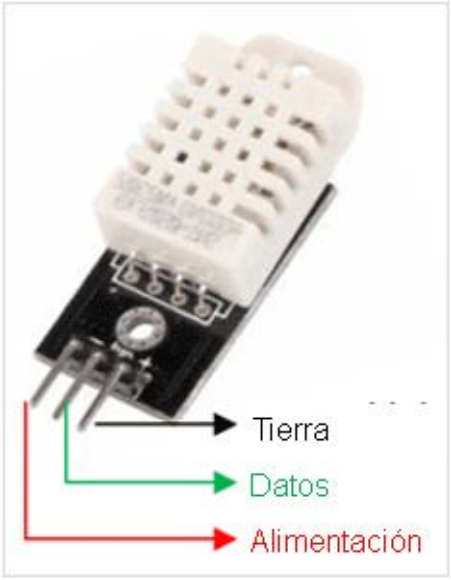

Figura 1.5 Sensor DHT22 *[8]*

Módulo relé

Los relés, componentes esenciales en circuitos eléctricos, desempeñan un papel sumamente importante en el mundo de la automatización al controlar el flujo de corriente en los circuitos. Funcionan como interruptores que se activan o desactivan según la señal eléctrica recibida, empleando un electroimán para abrir o cerrar contactos eléctricos [9]. Sus pines se pueden observar en la Figura 1.6.

Los relés electromagnéticos, comúnmente usados, constan de una bobina, un núcleo móvil y contactos eléctricos. Funcionan mediante la atracción magnética de la bobina, moviendo el núcleo y cerrando o abriendo los contactos. Dada su versatilidad se aplican en diversos campos como el control de dispositivos eléctricos en sistemas industriales, la automatización de procesos, sistemas de seguridad y protección, control de temperatura y aplicaciones en la domótica [9].

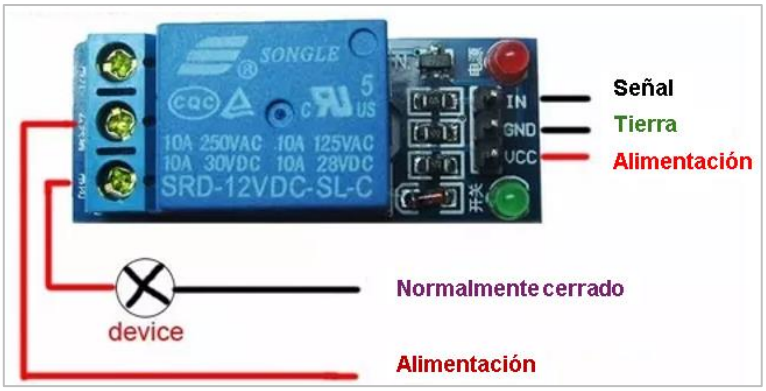

Figura 1.6 Módulo relé *[9]*

Foco Emisor de Calor de Cerámica

Las bombillas de cerámica infrarroja, comúnmente empleadas en terapias de calor y entornos de criaderos, juegan un papel fundamental en el cuidado de animales como pollos, lechones y reptiles. Estas lámparas de bajo consumo, al emitir únicamente calor a través de ondas infrarrojas y sin producir luz visible, son particularmente apreciadas por su idoneidad para su uso nocturno, evitando así perturbar el ciclo de sueño de los animales. El casquillo cerámico estándar E27 es el apropiado para su conexión [10], [11].

La bombilla de cerámica convexa infrarroja de fundición hueca se destaca por su construcción resistente a altas temperaturas, alcanzando hasta 530 (°C), y su potencia de salida de hasta 250 (W). Además de su uso en terapias de calor, estas bombillas son especialmente beneficiosas para mantener las condiciones de temperatura adecuadas en terrarios de reptiles, incubadoras de pollos o criaderos de lechones [10], [11].

Fuente de Alimentación

Una fuente de alimentación es un dispositivo encargado de transformar la corriente alterna (CA) proveniente de la red doméstica, la cual normalmente tiene un voltaje de 120 V en Ecuador, en corriente continua (CC). Esta transformación permite ajustar el voltaje de salida a valores como 5 V o 12 V, según los requerimientos específicos de los dispositivos conectados. La elección de la fuente dependerá directamente de las necesidades de alimentación de los dispositivos [12].

Las fuentes lineales reguladas son ideales para su uso en un SoC. Estas fuentes constan de cuatro fases las cuales se las puede observar en la Figura 1.7: la fase de transformación, donde se busca reducir la tensión alterna; la fase de rectificación, encargada de convertir la tensión alterna (AC) a tensión continua (DC); la fase de filtrado, que regula los picos que puedan presentarse en la tensión; y, finalmente, la fase de regulación, que mantiene estable la tensión en los terminales de salida [13].

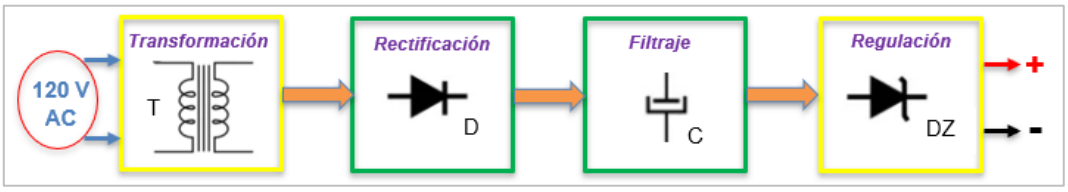

Figura 1.7 Fases de una fuente de alimentación *[13]*

2 METODOLOGÍA

El desarrollo del prototipo constó de cinco faces esenciales, las cuales permitieron la implementación del sistema de control de temperatura y humedad de un criadero de pollos. Las faces mencionadas se las puede observar en la Figura 2.1.

Figura 2.1 Metodología del proyecto

Al inicio del proyecto, se describió el alcance del mismo, estableciendo las características principales a tener en cuenta en el diseño y desarrollo. Esta fase permitió avanzar en el desarrollo de los objetivos previamente planteados. Para abordar los diversos requisitos, se llevó a cabo una investigación en fuentes confiables con el propósito de determinar los elementos necesarios para el diseño del prototipo.

En la sección de selección de *hardware* y *software*, al desarrollar el hardware seleccionado, se detallaron las características principales del SoC ESP32 en comparación con otro modelo de la misma serie. Se resaltaron las mejores cualidades de ambos SoC para determinar cuál sería el más adecuado para el prototipo. De manera similar, se eligió el sensor DHT22 después de compararlo con otro sensor de la misma serie. Los demás elementos, como el ventilador, el foco de cerámica de calor y los relés, fueron seleccionados considerando la facilidad de integración y sus características superiores.

En cuanto a la sección de *software,* se llevó a cabo una comparación entre plataformas compatibles con el ESP32, evaluando los planes ofrecidos por dichas plataformas. La elección se basó en aquella que proporcionara las mejores características y capacidad de escalabilidad.

Una vez determinados los elementos de *software* y *hardware,* se procedió con el diseño del prototipo. Esto implicó establecer el esquema de funcionamiento del proyecto y sus respectivas conexiones. Se demostró la creación de la cuenta en la plataforma seleccionada, la vinculación de los periféricos y la placa, además de la elaboración del código. En este código se especificaron las condiciones a tomar, como encender o apagar el calefactor y el ventilador en función de si la temperatura es mayor o menor a

un rango determinado. Del mismo modo, se establecieron condiciones para controlar la humedad, permitiendo encender o apagar el calefactor y ventilador en base a un rango específico de porcentaje de humedad. Se configuró la interfaz gráfica tanto de la *web* como de la aplicación móvil, tal como se plateó en el plan de proyecto.

En la sección de implementación del sistema, se llevó a cabo una investigación para determinar los lugares más idóneos tanto para el ventilador, calefactor y el sensor de temperatura y humedad. Fue necesario realizar un correcto cableado, así como su protección para evitar daños. Durante el desarrollo de las pruebas de funcionamiento, se obtuvieron los datos necesarios para realizar las correcciones pertinentes en la maqueta y lograr una distribución óptima de los elementos.

3 RESULTADOS

En la siguiente sección se presenta los parámetros utilizados para identificar los requerimientos necesarios en la sección de *hardware* y *software,* se muestra el diseño e implementación del prototipo, y finalmente, se evidencia las pruebas de funcionamiento. El propósito de esta sección es demostrar el cumplimiento de los objetivos establecidos en el plan de titulación.

3.1 Identificación de los requerimientos para el diseño del prototipo

Selección del SoC

El SoC es la parte principal en el funcionamiento del prototipo, ya que se encarga de tareas críticas como la lectura de los datos provenientes de los sensores de temperatura y humedad. Además, debe ser capaz de gestionar la delegación de tareas a los dispositivos de control, mantenerse alerta ante los cambios en el entorno avícola y garantizar la compatibilidad con los periféricos necesarios. Para maximizar la practicidad del prototipo, se busca un SoC asequible que sea completamente compatible con los periféricos y capaz de comunicarse de manera efectiva tanto con la interfaz web como con la aplicación de Android.

Recopilación de los datos ambientales

La recolección de datos es especialmente crucial en el diseño de un prototipo de control ambiental, ya que los datos obtenidos son fundamentales para automatizar una serie de decisiones que aseguren un entorno óptimo para las aves. La información recolectada brinda una visión detallada de las condiciones ambientales, la cual es utilizada por los distintos dispositivos de control ambiental para ajustarse automáticamente. Además,

esta información permite identificar posibles problemas en la infraestructura que alberga a las aves y facilita el posicionamiento estratégico de los dispositivos de control para obtener resultados óptimos.

Selección de dispositivo inteligente de medición de temperatura y humedad

La selección de un dispositivo inteligente para medir la temperatura y humedad implica la consideración de diversos aspectos clave. La precisión en las mediciones, la facilidad de comprensión de los datos recopilados, la durabilidad en entornos avícolas y la compatibilidad con el SoC que controla el sistema son criterios fundamentales en este proceso de selección.

Estos dispositivos de medición, generalmente sensores, suelen estar equipados con conectividad que les permite recopilar datos en tiempo real sobre las condiciones ambientales dentro del criadero. Esta capacidad de recolectar información actualizada es crucial para un control preciso del ambiente avícola. Dado que las aves son particularmente sensibles a los cambios ambientales, contar con datos precisos y en tiempo real es esencial para mantener condiciones estables y adecuadas para su desarrollo y bienestar.

Selección de dispositivos control

La selección de dispositivos de control es un aspecto crítico en el desarrollo del prototipo, similar a la recolección de datos, ya que estos dispositivos responden directamente a la información obtenida y son responsables de regular los aspectos ambientales hacia sus estados óptimos. Los dispositivos de control juegan un papel fundamental al activar o desactivar elementos que regulan la calidad del ambiente en el criadero. Para tomar decisiones informadas, es crucial considerar la precisión en su funcionamiento, la facilidad de instalación y la integración con el sistema general.

Estos dispositivos no solo reaccionan a los datos recopilados, sino que también son responsables de mantener las condiciones adecuadas para el bienestar de las aves. Al tener en cuenta criterios como la eficiencia energética y la precisión en sus acciones, se garantiza un ambiente óptimo y confortable para el desarrollo saludable de los pollos en el criadero.

Automatización de procesos

La automatización de procesos busca minimizar la intervención humana y asegurar respuestas inmediatas de los dispositivos de regulación ambiental. Esto tiene como objetivo reducir el estrés innecesario en el entorno de las aves. La automatización

implica la regulación de sistemas como calefacción, ventilación y control de la humedad ambiental. El propósito es mantener un ambiente óptimo para el crecimiento y bienestar de los pollos.

Para lograr una automatización eficiente, es fundamental considerar el lenguaje de programación y su correcta implementación, así como garantizar la compatibilidad con el SoC al que se integrará. La elección adecuada del lenguaje de programación y la correcta configuración garantizarán el funcionamiento preciso y eficaz de los sistemas automatizados, asegurando así un entorno estable y saludable para las aves en el criadero.

Identificación de la interfaz de usuario

El objetivo de la interfaz de usuario es presentar los datos recopilados de forma que sean comprensibles para el usuario. Se busca facilitar su revisión en cualquier momento, por lo que se pretende implementar tanto una interfaz de escritorio como una aplicación móvil compatible con todos los sistemas operativos.

3.2 Selección del hardware y software acorde a los requerimientos establecidos

Selección del *hardware*

Debido a la gran cantidad de placas, plataformas y sensores que existen en el mercado, para elegir los dispositivos más adecuados se requiere un detallado análisis de las principales características, teniendo en cuenta los aspectos más relevantes y requeridos para la elaboración del prototipo.

Selección del SoC para controlar el circuito

El microcontrolador considerado es el ESP32 de 32 pines mostrado en la Figura 3.1, un módulo compacto, de reducido tamaño y fácil manejo. Este microcontrolador está especialmente diseñado para aplicaciones IoT, lo que lo hace ideal para implementaciones específicas, como el control de sensores y otros periféricos. Su conectividad integrada *Wi-Fi* y *Bluetooth* lo vuelven especialmente útil para el acceso remoto y la comunicación con dispositivos móviles y sus respectivas aplicaciones [14].

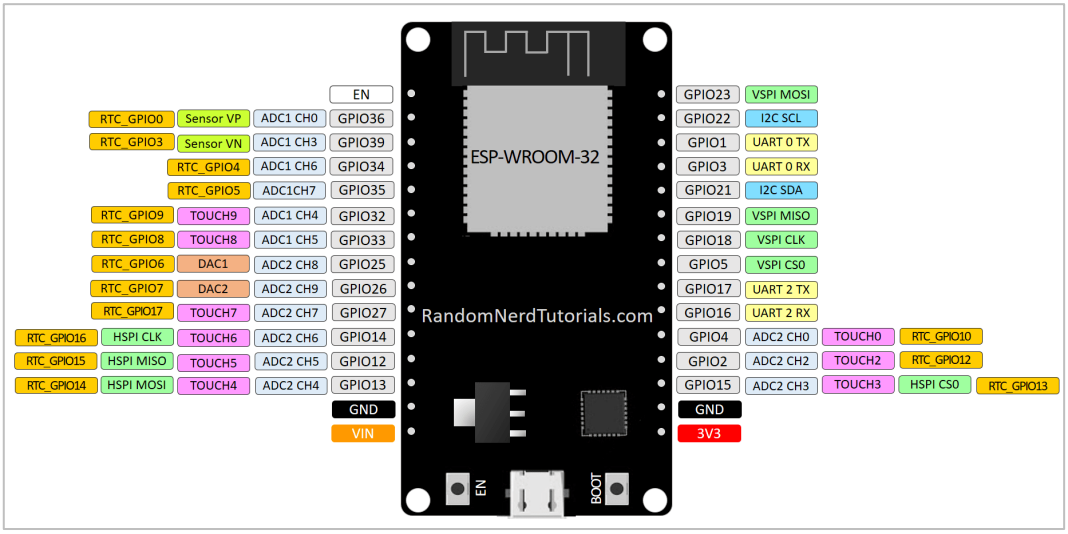

Figura 3.1 ESP32 pines *[15]*

Con la finalidad de evidenciar las características más importantes para la elaboración del prototipo, en la Tabla 3.1 se realiza una comparación entre ESP32 y ESP8266.

Tabla 3.1 Comparación ESP32 vs. ESP8266 *[15]*

	ESP32	**ESP8266**
Procesador	Xtensa Dual-Core LX6 con 600 DMIPS	Xtensa Single-bit L106
Núcleos de procesamiento	2	1
RAM	520 (kB)	160 (kB)
ROM	448 kB para arranque y funciones básicas	Sin ROM programable
Velocidad *Wi-Fi*	Hasta 150 (Mbps)	Hasta 72,2(Mbps)
Protocolos Wi-Fi	802,11 b/g/n	802,11 b/g/n
Bluetooth	Bluetooth v4.2 BR/EDR/BLE	No
Pines GPIO	36	17
Canales ADC	8	1 de 10 (bits)
Sensor de temperatura	Si	No
Protocolos de red	IPv4, IPv6, SSL, TCP/UDP/HTTP/FTP/MQTT	IPv4, TCP/UDP/HTTP/MQTT

Tomando en consideración las características descritas anteriormente, se consideró que la mejor opción es el módulo ESP32, puesto que al tener mejores cualidades puede funcionar correctamente con los diferentes periféricos necesarios para el desarrollo del

prototipo. Además, al poseer más núcleos, puede realizar más eficientemente las tareas asignadas [16].

También es la mejor opción ya que es capaz de obtener datos analógicos como lo son las mediciones de humedad y temperatura, posee mejores velocidades en *Wi-Fi,* lo que es sumamente necesario para una pronta respuesta al regular las condiciones ambientales, al igual que puede conectarse con varios protocolos de red [16].

A pesar de tener un consumo energético ligeramente superior, esto se compensa con las mejoras en sus características y eficiencia.

Selección del dispositivo inteligente para medir la temperatura y humedad

Al seleccionar el dispositivo inteligente para medir la temperatura y humedad, se consideró el sensor DHT22 debido a que en su estructura integra ambos tipos de sensores requeridos por el prototipo, cuenta también con un conversor analógico – digital lo que lo hace ideal para la recopilación de datos y comunicación con el ESP32. Además, es lo suficientemente compacto, tiene una respuesta rápida y un precio accesible [17].

En la Figura 3.2 se observa los dos sensores, tanto el sensor normal y su versión en módulo, para el prototipo se optó por la versión en módulo.

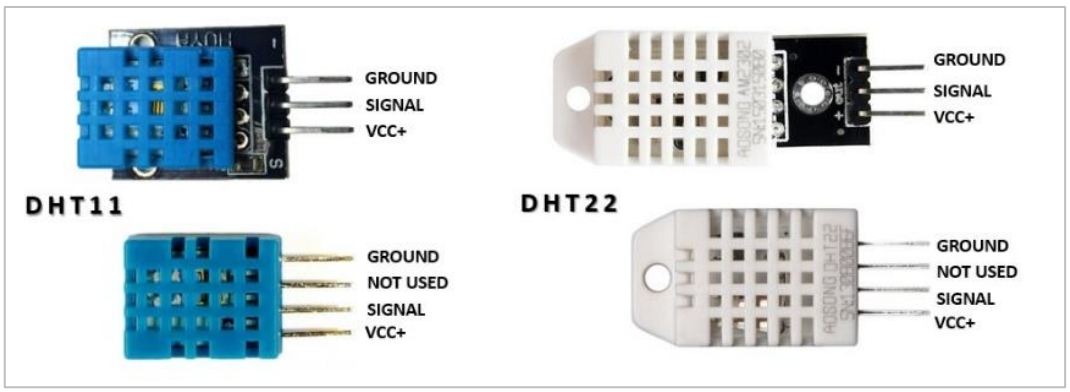

Figura 3.2 Sensor DHT11 vs. DHT22 *[17]*

En la Tabla 3.2 se llevó a cabo una comparativa entre dos modelos del sensor DHT, facilitando la evidencia de las principales características de ambos sensores.

Tabla 3.2 DHT11 vs DHT22 *[17]*

	DHT11	**DHT22**
Rango de temperatura	0 – 50(°C)	-40 – 125(°C)
Precisión en temperatura	±2(°C)	±0.5(°C)
Rango de humedad	20 – 80%	0 – 100%
Precisión en humedad	±5%	±2%
Frecuencia de muestreo	1(Hz) Cada segundo	0.5(Hz) Cada 2 segundos
Voltaje	3 – 5(V)	3 – 5(V)
Corriente	2.5(mA)	2.5(mA)
Medidas	15.5 x 12 x 5.5(mm)	15.1 x 25 x 7.7(mm)

Si bien ambos sensores podrían cumplir con el propósito del prototipo, al seleccionar el DHT22 se priorizó las mediciones, tanto de temperatura como de humedad. Dado que en un criadero de pollos las condiciones ambientales son críticas para el bienestar de las aves y para asegurar que los dispositivos de control ambiental respondan adecuadamente a los datos obtenidos por el sensor.

Selección de dispositivos de acción

Al elegir un dispositivo de acción o activación, se consideró su compatibilidad con el SoC ESP32, su velocidad de respuesta, su capacidad de soportar voltajes de hasta 110(V) y su facilidad de integración en el formato del circuito. Por esta razón, se optó por el módulo relé de un solo canal, con un voltaje de funcionamiento de 5(V) DC y una corriente de funcionamiento de 10(A), capaz de soportar hasta 250(V) AC, necesario para alimentar los dispositivos que regulan la temperatura. Además, su tiempo de activación es apenas de 10(ms), lo que garantiza un tiempo de respuesta reducido [18].

Selección de dispositivos de regulación de temperatura y humedad

Como dispositivo de regulación de las condiciones ambientales, se eligió un ventilador y un foco de cerámica. El disipador de calor seleccionado fue un ventilador con un voltaje de funcionamiento de 12(V), por lo que se requirió un adaptador de voltaje adicional para conectarlo directamente al tomacorriente doméstico.

El generador de calor elegido fue un foco de calor cerámico diseñado específicamente para su uso en criaderos de pollos. Con un voltaje de funcionamiento de 110(V), se optó por este tipo de bombilla debido a que no emite luz, sino únicamente calor. Esto lo hace ideal para que las aves puedan descansar correctamente durante la noche son interrupciones. Además, tiene una vida útil de hasta 10,000 horas de funcionamiento [19].

Selección del *software* de monitoreo y control

Plataforma de monitoreo y aplicación *Android*

Las plataformas que se tomaron en cuenta fueron *Adafruit IO* y *Arduino IoT Cloud,* las cuales tienen características similares. A continuación, se muestran las principales características de ambas plataformas:

Adafruit IO es un servicio orientado al *IoT,* su base de datos es compatible con varios módulos SoC como *Arduino, Raspberry Pi, Adafruit Feather Huzzah, ESP,* entre otros. Su característica más destacable es su capacidad de soportar los lenguajes de programación más populares. Se puede crear bibliotecas personalizadas o utilizar las que se encuentran en la plataforma. En ella se encuentran varias herramientas de administración, monitoreo y control de datos, esta característica permite el monitoreo de los datos en tiempo real [20].

Además, tiene la facilidad de integrarse con aplicaciones de terceros lo que permite guardar los datos no solo en la nube de *Adafruit IO,* sino también en los dispositivos móviles [20].

Arduino IoT Cloud es una plataforma diseñada por *Arduino*, robusta y adaptable en proyectos *IoT*. Ofrece una amplia gama de herramientas y ventajas, con un enfoque centrado en la versatilidad, simplicidad y usabilidad, lo que lo hace ideal tanto para usuarios avanzados como para principiantes. Es especialmente adecuada para usuarios familiarizados con el entorno de *Arduino* [5].

Una de sus características más destacables es su interfaz intuitiva y su facilidad para conectar y comunicar dispositivos de diferentes marcas y fabricantes, permitiendo una integración completa en entornos de hardware mixto [5].

Ambas plataformas cuentan con una versión gratuita con ciertas limitaciones, como la asociación de apenas 5 objetos, alojamiento de datos en la nube y visualización de datos tanto en entornos para escritorio o dispositivos móviles. La diferencia clave radica en que *Arduino IoT Cloud* tiene una aplicación más sencilla de entender con comunicación directa hacia la nube pudiendo modificar su interfaz desde la *web* para hacerla más

amigable para el usuario final. La aplicación de *Adafruit IO* también es intuitiva, pero puede ser un poco más compleja de usar, sobre todo porque tiene más funciones que un usuario común no comprendería del todo [5] [20].

El módulo ESP32, al estar basado en *Arduino*, es más compatible con *Arduino IoT Cloud*, lo que facilita su programación sin la necesidad de requerir documentación adicional para comprender su API, a diferencia de *Adafruit IO*, que requiere más estudio para implementarlo adecuadamente.

Ambas plataformas son viables para el desarrollo del prototipo, ya que cuentan con versiones gratuitas y características similares, como interfaces amigables con el usuario y medidas estándar de seguridad, como el cifrado de datos mediante HTTPS y la autenticación de usuarios y dispositivos.

A pesar de ser plataformas similares, se optó por *Arduino IoT Cloud* para la recolección y monitoreo de datos, ya que se adapta mejor a las necesidades del prototipo, es más compatible con el SoC elegido, tiene una API más intuitiva y fácil de comprender con conocimientos previos en Arduino. Además, en caso de requerir un plan de pago, Arduino IoT Cloud ofrece más variedad y planes más económicos, con características superiores según se evidencia en la Tabla 3.3.

Tabla 3.3 Plan de paga *Adafruit IO* vs *Arduino IoT Cloud [21] [22]*

	Arduino IoT Cloud	***Adafruit IO***
Número de objetos	25	Ilimitados
Tableros	Ilimitados	Ilimitados
Retención de datos	90 días	60 días
Compartición de tableros	Sí	No
Compilaciones	Ilimitadas	-
Almacenamiento de código	Ilimitado	-

3.3 Diseño del prototipo de sistema de control

Esquema organizacional del proyecto

En la Figura 3. ***3*** y en la Figura 3. ***4*** se presenta el esquema y la organización que muestra la composición del prototipo y los dispositivos que este controla. El prototipo está mayormente conformado por el microcontrolador ESP32, que actúa como el cerebro del prototipo, encargándose de controlar todas las interacciones y de gestionar la comunicación con los servicios de *Arduino IoT Cloud*, tales como el panel de control *web* y la aplicación móvil *IoT Remote*. Además de responder automáticamente a condiciones específicas descritas en el código, la arquitectura utilizada es el desarrollo del prototipo es de tipo centralizado.

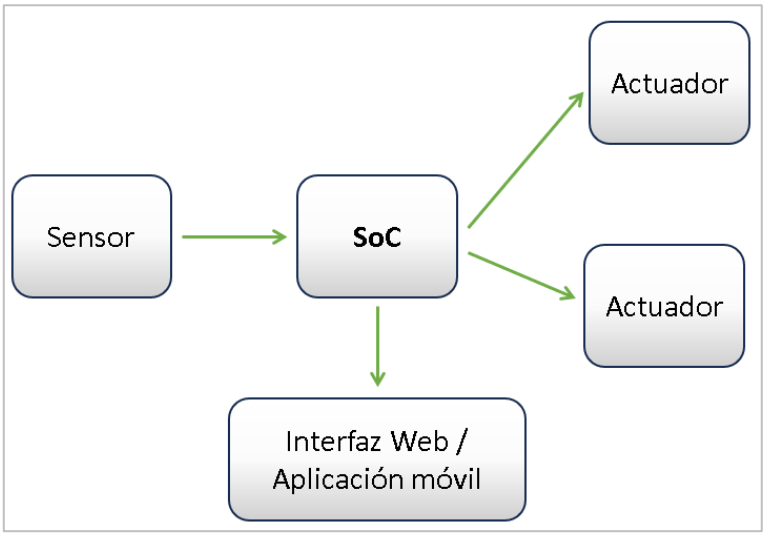

Figura 3. 3 Arquitectura organizacional

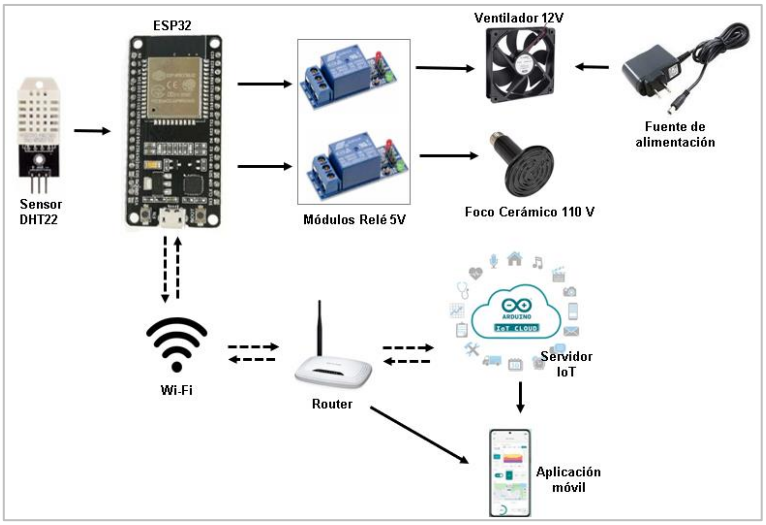

Figura 3. 4 Esquema de funcionamiento del prototipo

En la Figura 3. ***4*** se observa que el ESP32 recolecta la información de temperatura y humedad proporcionada por el sensor DHT22. Lugo, procesa los datos obtenidos y actúa en consecuencia, enviando las respectivas órdenes a los relés. Al mismo tiempo, durante este proceso, el ESP32 establece la conexión a la red *WiFi* para comunicarse con los servidores de *Arduino IoT Cloud*. Una vez que la conexión está establecida, la

plataforma *web* recibe los datos y los presenta como representaciones gráficas. De esta manera, se pueden visualizar los diversos datos tanto en la plataforma *web* como en la aplicación móvil.

Creación del ambiente en *Arduino IoT Cloud*

La placa ESP32, al ser una placa basada en Arduino, permite una asociación relativamente sencilla con la plataforma de *Arduino IoT Cloud*. El primer paso consiste en la creación de una cuenta en la plataforma. Dado que se trata de un prototipo, se optó por la versión gratuita. Para ello, se accedió a la dirección *arduino.cc,* lo que mostró la página principal de la plataforma, como se puede observar en la Figura 3.5.

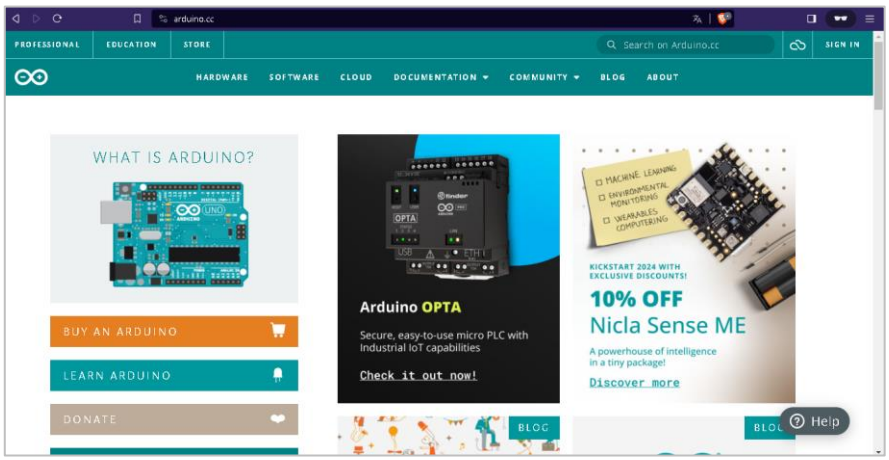

Figura 3.5 Página de inicio de *Arduino IoT Cloud*

Se procedió a crear la cuenta gratuita ingresando unos pocos datos, como la fecha de nacimiento, la dirección de correo electrónico, un nombre de usuario y una contraseña, tal como se puede observar en la Figura 3.6.

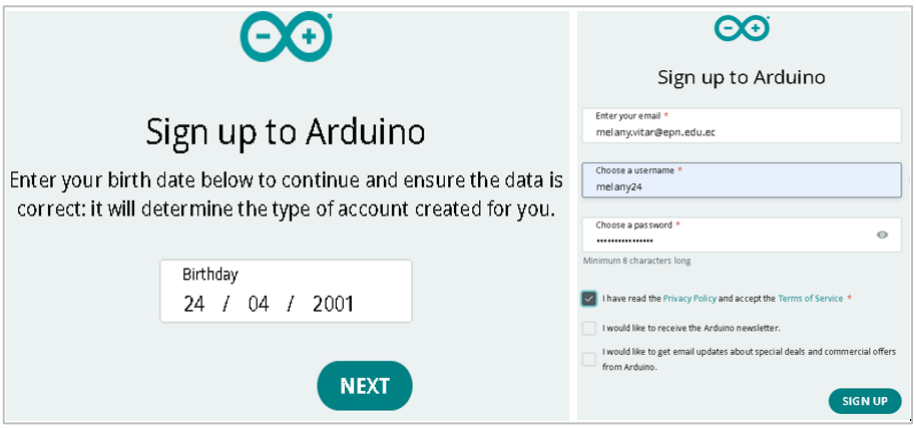

Figura 3.6 Creación de cuenta en *Arduino IoT Cloud*

Finalmente, en la Figura 3.7 se muestra el perfil de usuario dentro de la plataforma, donde es posible vincular placas, dispositivos, y acceder a otros servicios y opciones.

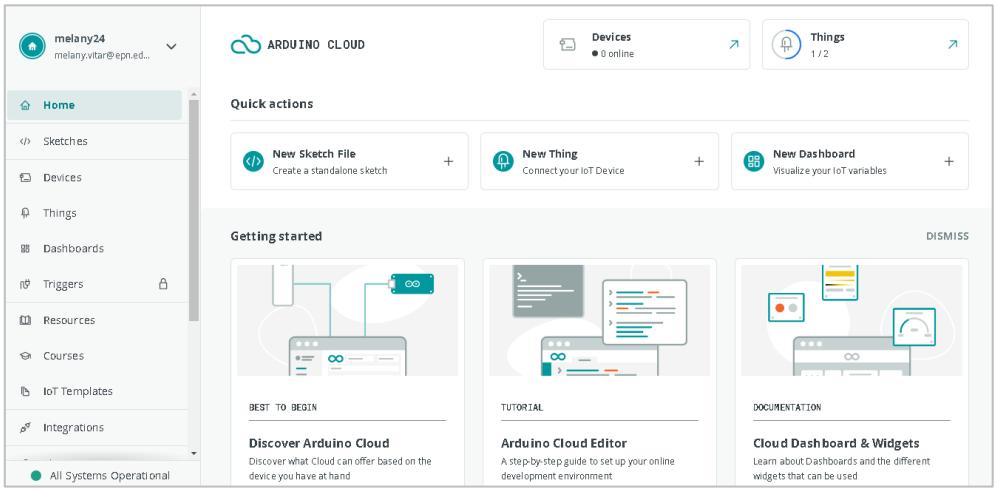

Figura 3.7 Perfil de usuario en *Arduino IoT Cloud*

Vinculación de placa y dispositivos a *Arduino IoT Cloud*

Dentro del entorno de la plataforma web, se ingresó a la opción "*Things"*, dónde, para iniciar la creación de un proyecto, es necesario crear un elemento o "*Thing*". Por lo tanto, se seleccionó la opción "*Create thing",* ubicada en la parte superior derecha como se muestra en la Figura 3.8.

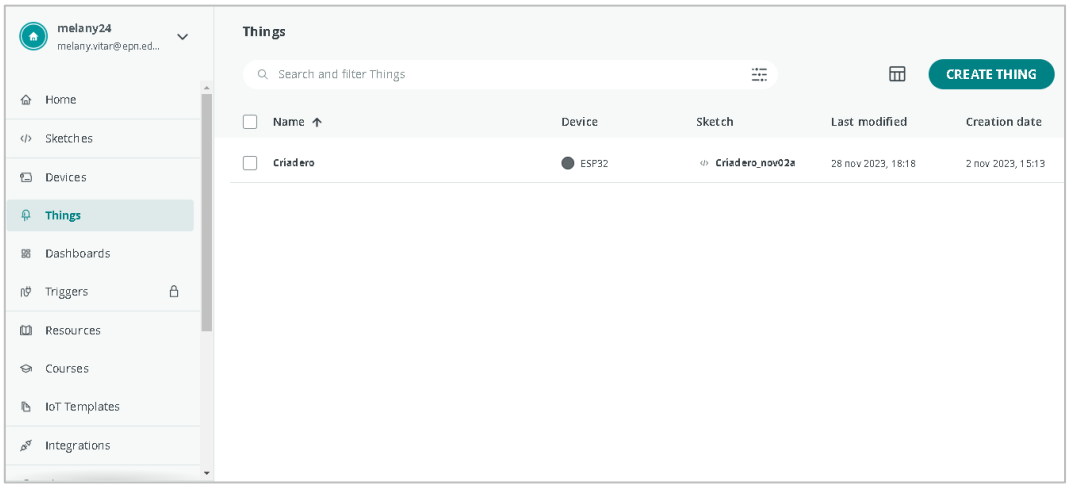

Figura 3.8 Creación de un *Thing*

Una vez creado, se despliegan tres opciones, como se aprecia en la Figura 3.9; la primera de ellas se emplea para la declaración de variables, que en este prototipo serían humedad22, temperatura22, relay1_gpio y relay2_gpio. La segunda opción se utiliza para vincular una placa al sistema, siendo en este caso es la placa del ESP32. Por último, la tercera opción se emplea para la conexión a la red *WiFi.*

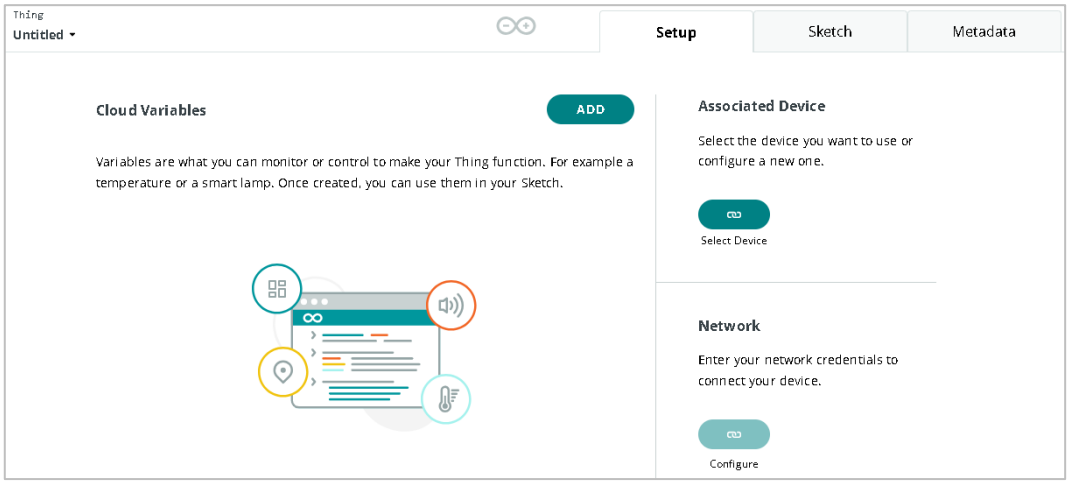

Figura 3.9 Entorno dentro de un *Thing*

En la Figura 3.10 se puede observar el cómo se realiza la creación de las variables las cuales corresponden a los periféricos, se debe ingresar el nombre de la variable, el tipo de variable y el qué puede hacer la variable, este mismo proceso se realizó para todas las variables.

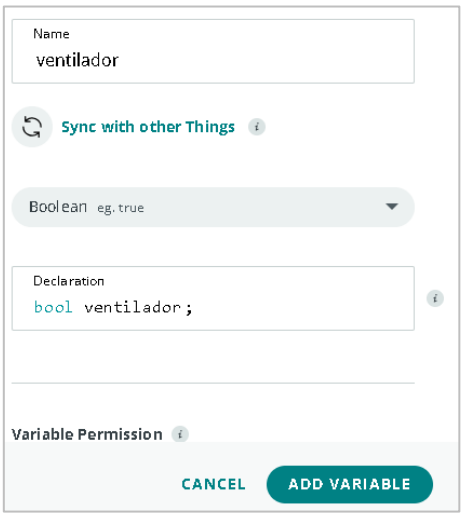

Figura 3.10 Creación de variables

Para la vinculación de la placa a la plataforma, es necesario seleccionar el tipo de placa, es decir, optar por la opción de placa basada en Arduino. A continuación, se deplegarán tres casillas, entre las cuales se debe elegir “ESP32”. Posteriormente, se presentarán diversas opciones relacionadas con los modelos disponibles para la placa, de las cuales se seleccionó “ESP32 *Dev Module*”. Finalmente, se asigna un nombre al dispositivo recién vinculado. Todo el proceso se lo puede observar en la Figura 3.11.

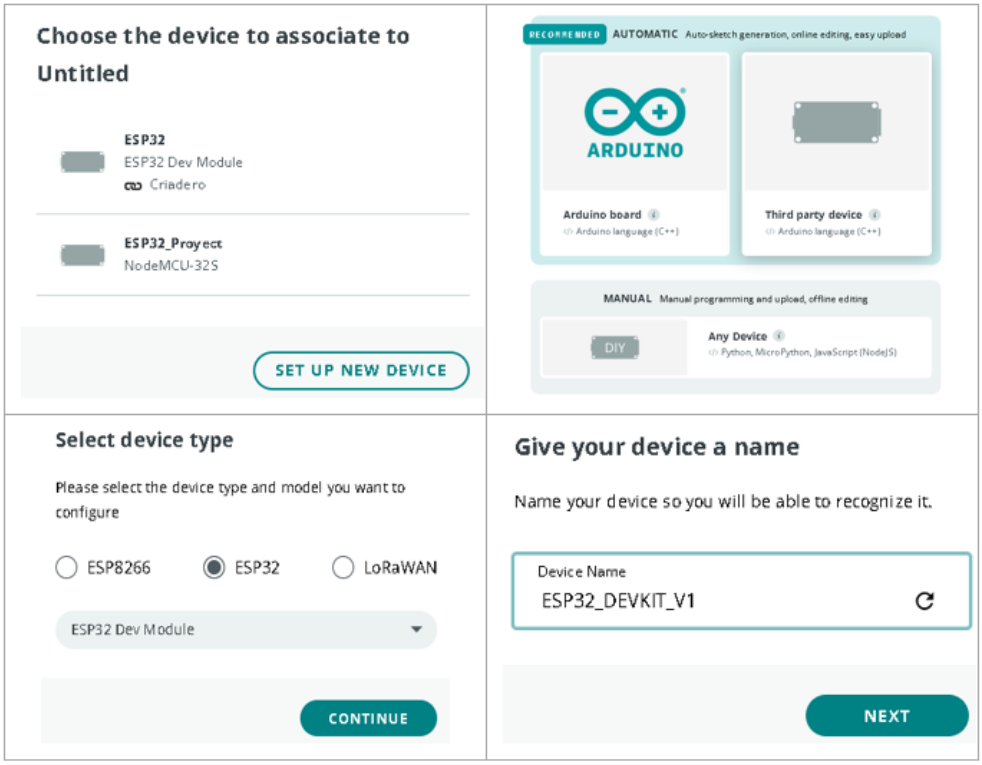

Figura 3.11 Proceso de vinculación del SoC

Completada la vinculación de la placa, aparecerá un mensaje indicando que la vinculación del SoC fue exitosa. Además, se mostrará una clave que se utilizará para la conexión *Wi-Fi*. También se proporsionaba un archivo PDF descargable que contiene tanto la clave como el ID del SoC. Es crucial guardar adecuadamente el PDF o, alternativamente, la clave, ya que no se mostrará nuevamente y será necesario vincular el dispositivo de nuevo en caso de pérdida. En la Figura 3.12 se puede observar lo mencionado.

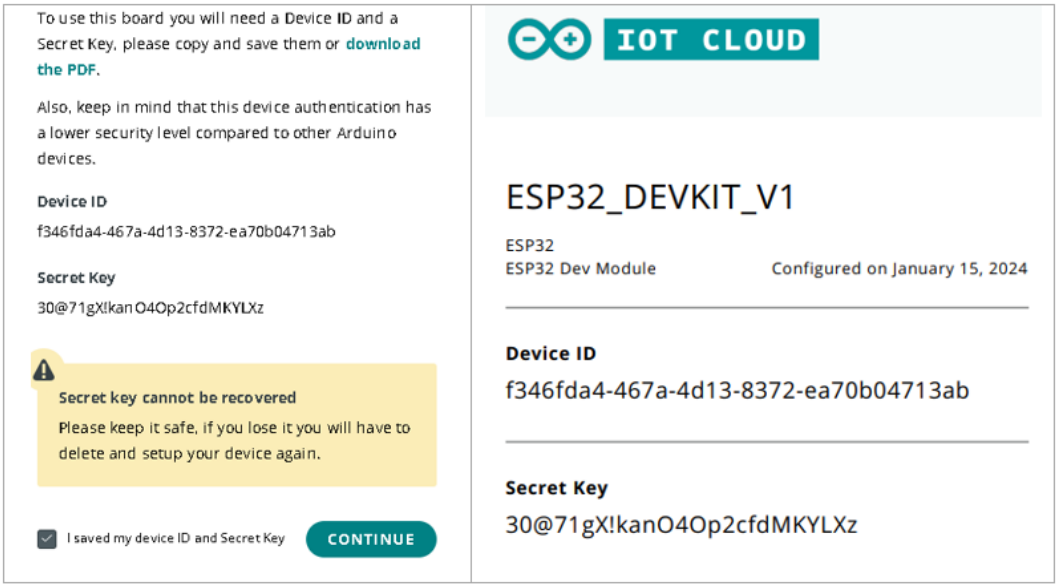

Figura 3.12 ID y clave del SoC

En la Figura 3.13 se observan los datos de la red a la cual se va a conectar y la respectiva clave obtenida anteriormente en el PDF descargable.

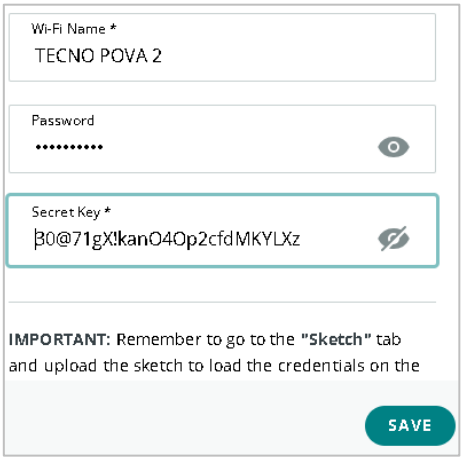

Figura 3.13 Vinculación y conexión *WiFi* del SoC

Configuración y desarrollo del código

Al ingresar a la interfaz de configuración, aparecerá un mensaje solicitando la instalación del agente *Arduino Create Agent,* como se muestra en la Figura 3. ***14***. Una vez instalado, es necesario asegurarse de que la aplicación esté ejecutándose antes de acceder al editor de código de la plataforma. Para verificar la ejecución exitosa, se puede observar el ícono en la barra de tareas, que debería estar de color gris o, en su defecto, blanco. Si se presenta en un tono gris claro, significa que el *software* no está activo. Este agente es especialmente importante para que la plataforma *web* pueda reconocer la placa en a la que se subirá el código.

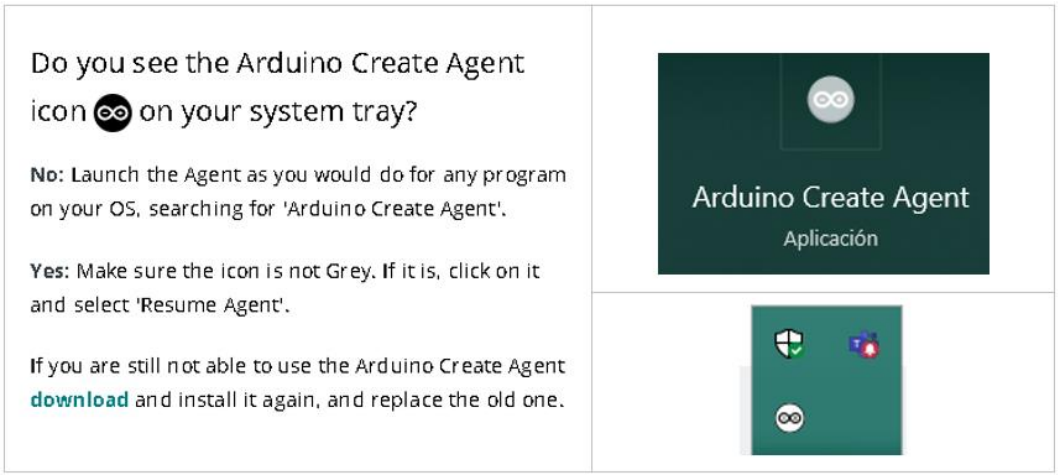

Figura 3. 14 Instalación del *software Arduino Create Agent*

En la Figura 3.15 se mustra el editor de código de *Arduino IoT Cloud*. Posteriormente a la instalación del agente, es posible comenzar a desarrollar el código sin problemas.

Como primer paso, se debe descargar la respectiva librería “DHT Sensor Library for ESPX” , la única librería especial necesaria para el propósito del prototipo.

Figura 3.15 Editor de código de *Arduino IoT Cloud*

Para la elaboración de código, se incluyeron dos librerías, como se puede observar en Figura 3.16. La primera librería fue “DHTesp.h”, la cual sirve para la lectura de datos obtenidos por el sensor de temperatura y humedad DHT22 y es compatible con la placa ESP32. La segunda, “thingProperties.h”, se incluye automáticamente al ingresar al editor de código de *Arduino IoT Cloud*. Además, se incorporan automáticamente, en forma de comentario, las variables y el tipo, las cuales fueron declaradas anteriormente siguiendo el procedimiento que se observa en la Figura 3.10.

```
#include <DHTesp.h> // Librería de ESP32
#include "thingProperties.h" // Archivo de configuraciones automáticas

//Variables usadas en el desarrollo del proyecto

/*
  float humedad22;
  float temperatura22;
  bool cooler;
  bool foco;
*/
```

Figura 3.16 Inclusión de librerías y comentario de variables

Lo siguiente en el código fue la creación del objeto denominado “DHTesp dht”. Este objeto se utiliza para encapsular los datos obtenidos de las variables ‘dht’ y vincularlos con los datos del sensor. En Figura 3.17, además de observarse la creación del objeto, también se definen las variables ‘relay1_gpio 25’ y ‘relay2_gpio 27’ que se utilizarán para controlar los relés. Asimismo, se determina el pin al que estará conectado cada relé.

```
DHTesp dht; //Creación del objeto dht
#define relay1_gpio 25 // Relé 1 conectado al pin25
#define relay2_gpio 27 // Relé 2 conectado al pin27
```

Figura 3.17 Creación del objeto dht y definición de las variables *relay*

En la Figura 3.18 se observa la sección donde se determina la inicialización del código con el 'void setup'. La línea 'Serial.begin(9600)' indica el inicio de la comunicación serial a algún puerto abierto. Caso contrario, si no se encuentra el puerto abierto, se establece un tiempo de espera de 1500 milisegundos, como se observa en la línea 'delay(1500)'.

En la línea 'ArduinoCloud.begin(ArduinoIoTPreferredConnection)', se establece que la placa ESP32 pueda iniciar la conexión hacia la nube de Arduino, recopilando los datos y realizando las acciones necesarias según lo determinado en el resto del código.

La línea 'dht.setup(5, DHTesp::DHT22)' indica que el objeto 'dht', es decir, el sensor, se encuentra conectado en el pin 5 y que el sensor conectado es el DHT22, utilizando la librería específica para la placa ESP32.

En las líneas 'pinMode(relay1_gpio, OUTPUT)' y 'pinMode(relay2_gpio, OUTPUT)' se establece el tipo de comunicación para esos pines. En estos casos, los pines son de tipo salida, ya que no recolectan información, simplemente se utilizan para accionarse según lo automatizado.

```
//Inicialización del código

void setup() {

  Serial.begin(9600); //Inicio de la comunicación serial con el puerto abierto
  delay(1500); //Tiempo de espera por si no se encuentra el puerto

  initProperties();

  ArduinoCloud.begin(ArduinoIoTPreferredConnection); //Conección con Arduino Cloud

  setDebugMessageLevel(2); //Establece el tipo de mensaje para arduino cloud
  ArduinoCloud.printDebugInfo(); // Muestra la información en el monitor serie

  dht.setup(5, DHTesp::DHT22); //Indica el tipo de sensor y pin conectado
  pinMode(relay1_gpio, OUTPUT); //Tipo de comunicación de los relés
  pinMode(relay2_gpio, OUTPUT);
}
```

Figura 3.18 Inicialización del código

En la Figura 3.19, se puede observar la fución 'void loop', la cual se ejecutará continuamente en un bucle. En la línea 'ArduinoCloud.update()', se establece que la plataforma *Arduino IoT Cloud* se actualizará constantemente con los nuevos datos obtenidos del ESP32. Las líneas 'temperatura22 = dht.getTemperature()' y 'humedad22 = dht.getHumidity()' sirven para obtener los datos de temperatura y humedad, respectivamente, por parte del sensor y guardarlos en el objeto 'dht'.

```
void loop() {
  ArduinoCloud.update(); //Actualización continua de Arduino cloud

//Obtención temperatura y humedad y los guarda en sus respectivas variables
  temperatura22 = dht.getTemperature();
  humedad22 = dht.getHumidity();
```

Figura 3.19 Función de bucle

En la Figura 3.20, se puede observar la configuración utilizada para automatizar la tarea de control del ventilador. En la línea 'if (temperatura22 > 26.50 || humedad22 < 65)', se establece la condición en la cual, si la temperatura (almacenada en la variable 'temperatura22') es mayor a 26.50 o la humedad (almacenada en la variable 'humedad22') es mayor a 65, el ventilador se enciende. La activación del ventilador se realiza a través del relé asociado a la variable 'relay1_gpio'. En caso de que no se cumplan estas condiciones, el ventilador se desactiva.

```
// Control de los relés basado en la temperatura y humedad
// Enciende el ventilador si la temperatura supera 26.5°C o si la humedad supera 65%

if (temperatura22 > 26.50 || humedad22 > 65) {
  digitalWrite(relay1_gpio, HIGH);
} else {
  digitalWrite(relay1_gpio, LOW);  // Caso contrario se desactiva
}
```

Figura 3.20 Condición para el control del ventilador

Por su parte en la Figura 3.21 se puede observar la configuración que se realizó para controlar el calefactor, es decir la variable 'relay2_gpio', en este caso, si la temperatura es menor a 25.50 o la humedad es menor a 55, el calefactor de activará en caso contrario el calefactor se mantendrá inactivo. En la última línea se puede observar un 'delay(5000)' el cual indica el tiempo de espera hasta volver a ejecutarse el código y repetir el bucle.

```
  // Enciende el foco si la temperatura es menor a 25.5°C o si la humedad disminuye de 55%

  if (temperatura22 < 25.50 || humedad22 < 55) {
    digitalWrite(relay2_gpio, HIGH);
  } else {
    digitalWrite(relay2_gpio, LOW);   // Desactivación
  }

  delay(5000);
}
```

Figura 3.21 Condición para el control del calefactor

Para tener una mejor idea gráfica del funcionamiento del prototipo, en la Figura 3.22 se presenta el diagrama de flujo, donde se muestran las decisiones y las acciones a realizarse respecto a la lectura del sensor.

Ya desarrollado el código, se verificó que la sintaxis y la coherencia fueran correctas. En la Figura 3.23se puede evidenciar que el código está exitosamente escrito, además, se muestra el tamaño en memoria del código.

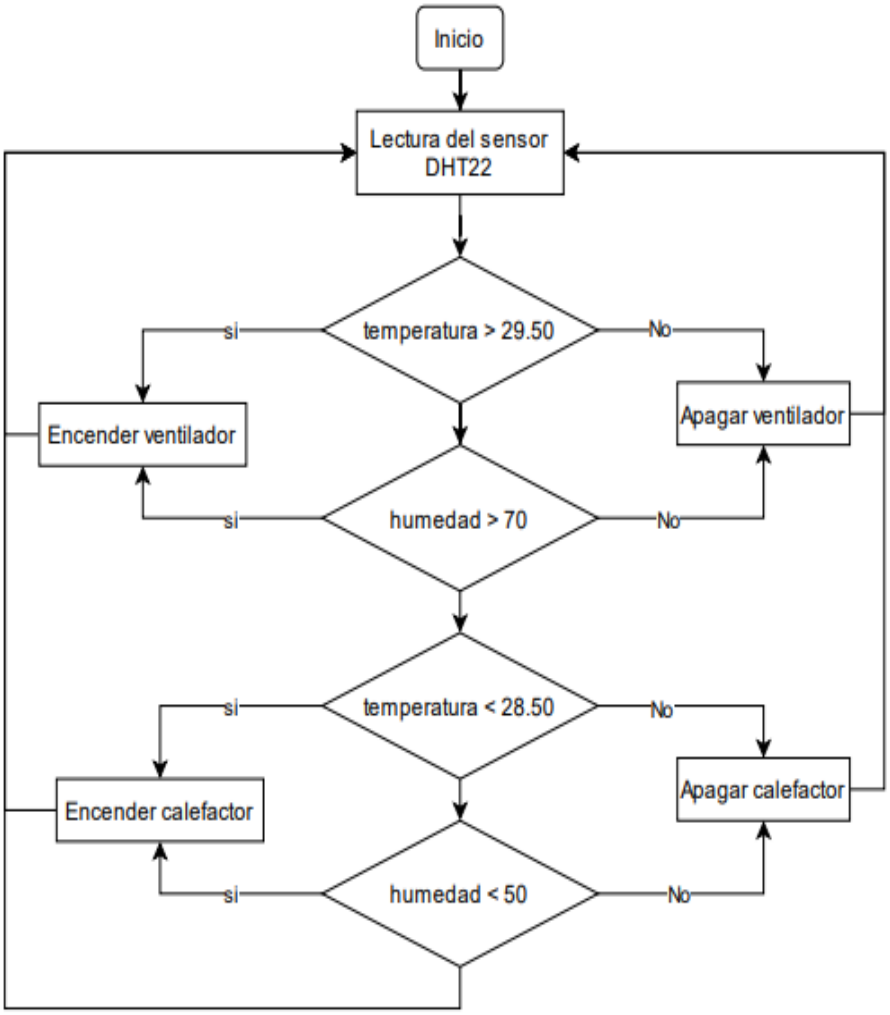

Figura 3.22 Diagrama de flujo de control del prototipo

```
/usr/local/bin/arduino-cli compile --fqbn
esp32:esp32:esp32:CPUFreq=240,DebugLevel=none,EraseFlash=none,EventsCore=1,FlashFreq=80,FlashMode=qio,FlashSi
--build-cache-path /tmp --output-dir /tmp/159003040/build --build-path /tmp/arduino-build-
527AEF132869D6ACAE33535FE6BE43AE /tmp/159003040/Criadero_nov02a
Sketch uses 1115489 bytes (85%) of program storage space. Maximum is 1310720 bytes.
Global variables use 51448 bytes (15%) of dynamic memory, leaving 276232 bytes for local variables.
Maximum is 327680 bytes.
```

Figura 3.23 Verificación del código

Carga del código y verificación del puerto ‘COM’

Para poder subir el código a la placa, en el caso del navegador ‘Brave’, se debe desactivar las protecciones de la página, como se puede observar en la Figura 3.24. De lo contrario, la plataforma no encontrará el puerto al que está conectada la placa y será imposible cargar el código.

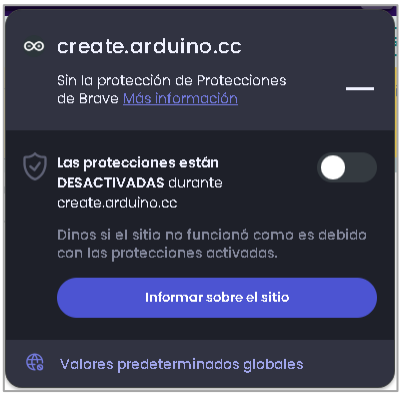

Figura 3.24 Desactivar protecciones del navegador

Para asegurarse de que el puerto esté activo, se puede revisar el 'Administrador de dispositivos', donde se pueden ver los puertos activos conectados. Como se muestra en la Figura 3.25, el puerto utilizado para la conexión de la placa es el 'COM3'.

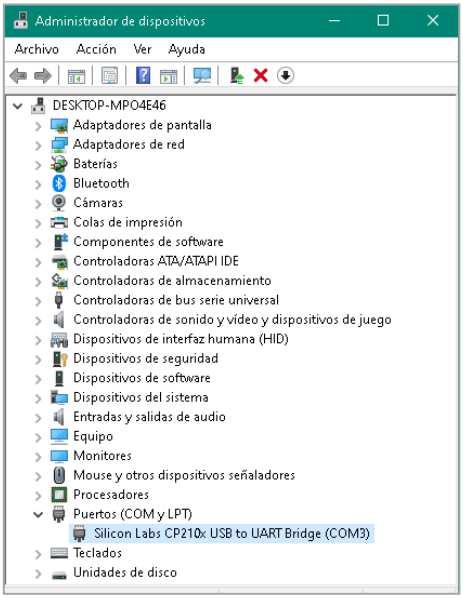

Figura 3.25 Puerto 'COM' en el administrador de dispositivos

En el editor de código, también es necesario determinar el tipo y modelo de placa con se puede ver en la Figura 3.26. Además, se debe seleccionar el puerto al que está conectado, en este caso, el puerto 'COM3'.

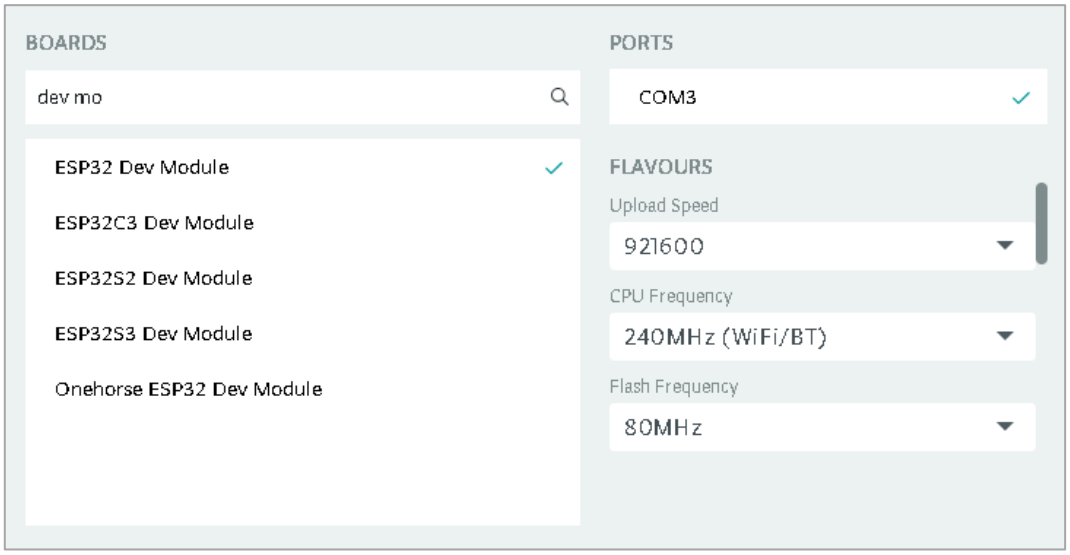

Figura 3.26 Selección de la placa y puerto conectado

Al cargar el código a la placa, se realiza nuevamente el proceso de verificación. Este proceso puede demorar un poco. Una vez finalizada la verificación y la carga, se mostrará un mensaje de finalización, como se puede observar en la Figura 3.27.

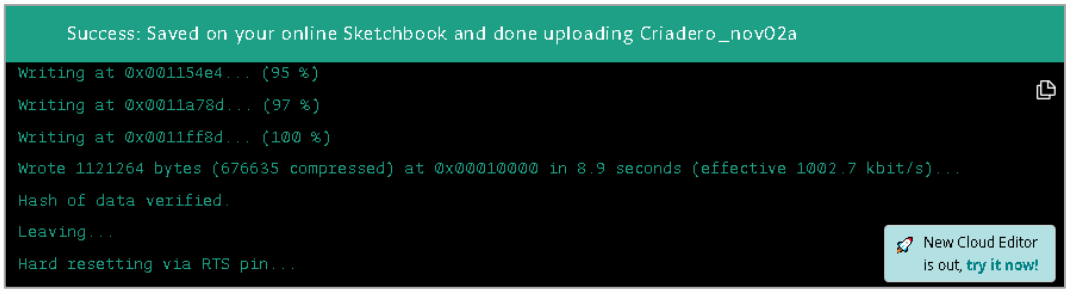

Figura 3.27 Carga del código a la placa

Verificación de la conexión y configuración de la interfaz *web* y móvil

En el área ‘Thing’, se encuentra una sección que indica el estado de la placa, es decir, muestra si está conectado a la plataforma y la información de la red inalámbrica a la que está conectada la placa. A la izquierda de la Figura 3.28, se muestra el estado ‘Offline’, lo que indica que la placa aún no se ha conectado a la plataforma. Por otra parte, al lado derecho se muestra el estado ‘Online’, lo que hace referencia que la placa ya se encuentra en comunicación con la plataforma.

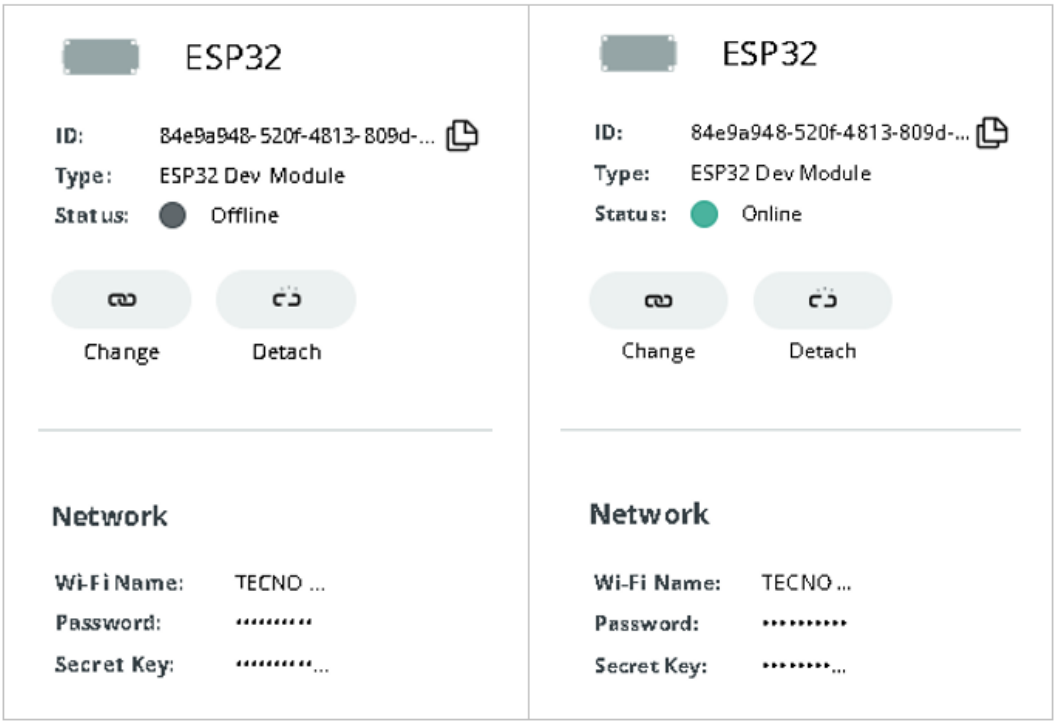

Figura 3.28 Estado de la conexión de la placa

Si se desea observar de forma más dinámica la conexión, se puede hacer desde el monitor serie, como se muestra en la Figura 3.29. En ella, se puede observar el estado de la conexión a través del monitor serie de la aplicación 'Arduino IDE'. Como se puede ver, la conexión comienza obteniendo la configuración de *Arduino IoT Cloud*. Posteriormente, se obtiene el ID del dispositivo y se realiza la conexión MQTT. Luego, intenta conectarse a la red *WiFi* que se ha establecido anteriormente en la configuración. Si la conexión falla, puede intentar conectarse nuevamente después de 4 segundos. Una vez que ha logrado conectarse a la red, muestra el mensaje de conexión con *Arduino IoT Cloud* y, finalmente, se obtiene el *ID Thing*.

```
***** Arduino IoT Cloud - configuration info *****
19:25:20.215 -> Device ID: 84e9a948-520f-4813-809d-64795e84183f
19:25:20.301 -> MQTT Broker: mqtts-up.iot.arduino.cc:8884
19:25:20.348 -> WiFi.status(): 255
19:25:25.416 -> Connection to "TECNO POVA 2" failed
19:25:25.465 -> Retrying in  "4000" milliseconds
19:25:30.420 -> Connected to "TECNO POVA 2"
19:25:59.463 -> ArduinoIoTCloudTCP::handle_WaitDeviceConfig device waiting for valid thing_id
19:26:20.319 -> Connected to Arduino IoT Cloud
19:26:20.319 -> Thing ID: 4b168e48-fc3c-416d-8f61-87c71c19866c
```

Figura 3.29 Visualización del estado de la conexión por monitor serie

En la Figura 3.30, se observa la pestaña de 'Dashboards', utilizada para crear la visualización tanto de la plataforma *web* como de la aplicación móvil. Y en la Figura 3.31 muestra el entorno sin ninguna modificación; los márgenes que se presentan en color gris representan el área en la que se debe trabajar y será la que se muestre en la interfaz gráfica.

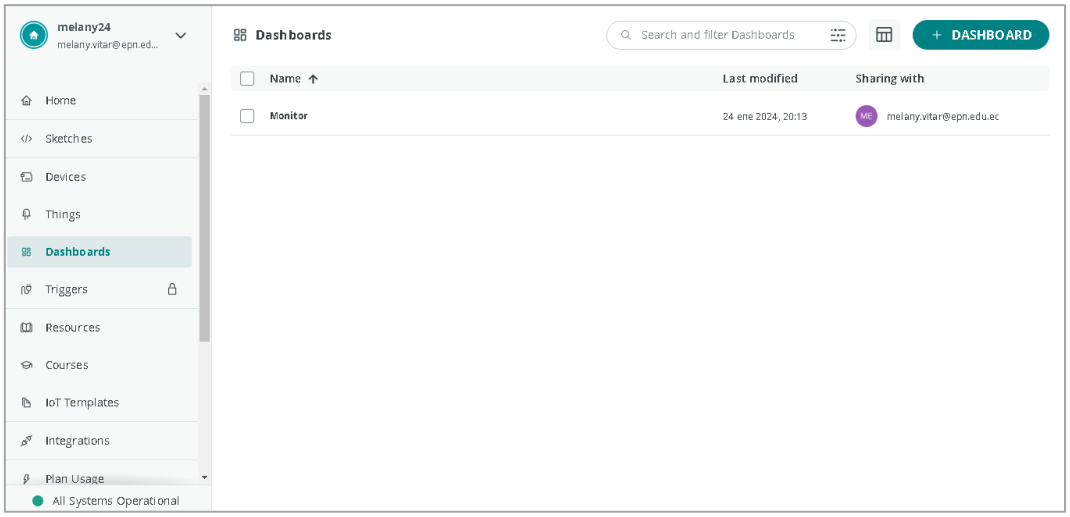

Figura 3.30 Creación del *Dashboard*

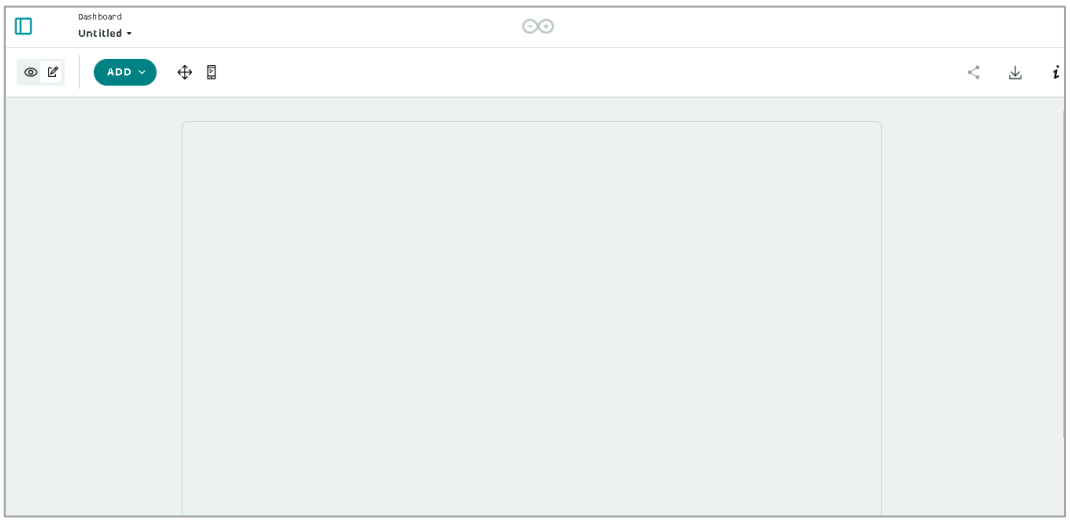

Figura 3.31 *Dashboard* sin modificaciones

Para diseñar la interfaz gráfica, se tienen varias opciones dependiendo de las necesidades y las variables declaradas en el código. En la Figura 3.32, se muestra una lista con las opciones disponibles en la versión gratuita. Si se desean gráficas más avanzadas, se deberá actualizar a la versión de pago. Para el prototipo, se utilizaron *widgets* relacionados con el porcentaje asociado a la humedad, un indicador para la temperatura y una gráfica estadística para ambas variables. De esta forma, se puede tener una mejor percepción y registro de los cambios en la humedad y temperatura.

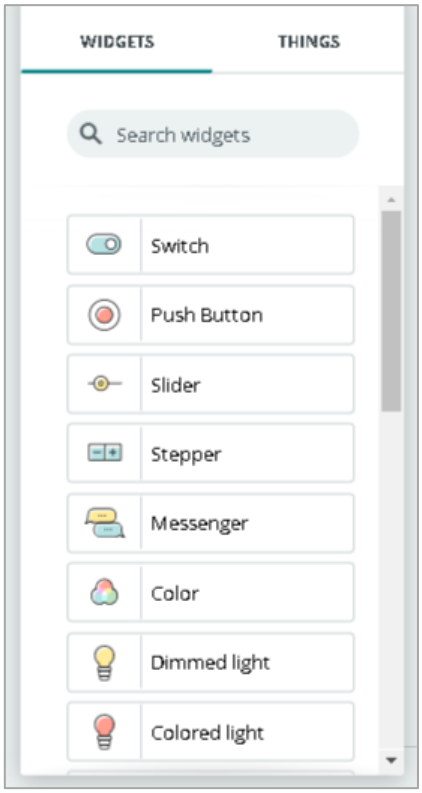

Figura 3.32 Lista de *widgets*

Ya elegidos los *widgets* necesarios, se procedió a vincularlos con las respectivas variables. Dentro de la configuración de las variables, se puede modificar el nombre y, si se desea, realizar algún cambio de color al llegar a un cierto valor, como se puede observar en la Figura 3.33.

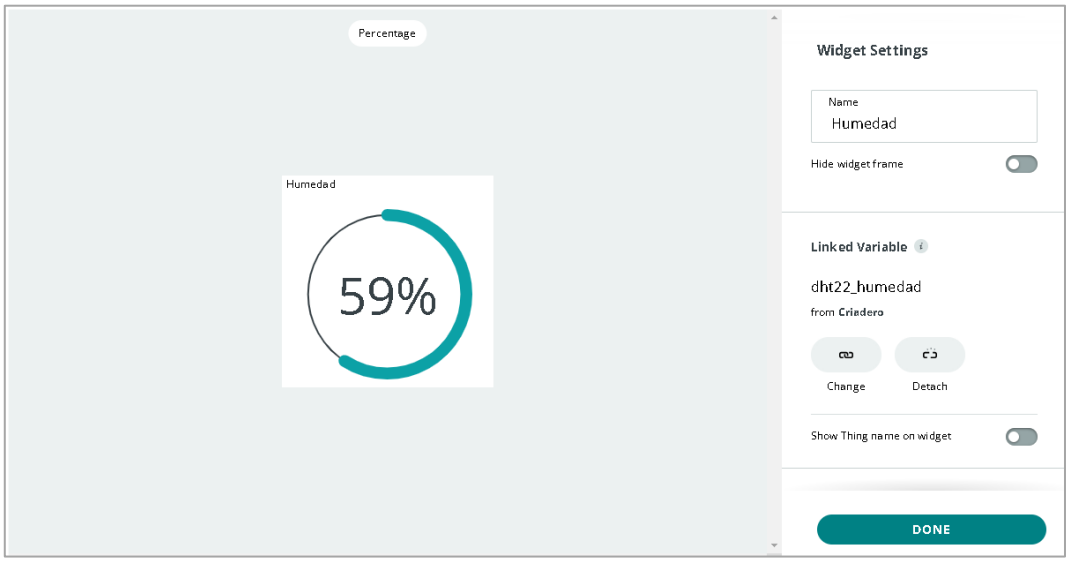

Figura 3.33 Configuración de los *widgets*

Una vez que configurados los *widgets necesarios,* se procedió a ubicarlos en el área de presentación teniendo la vista que se muestra en la Figura 3.34.

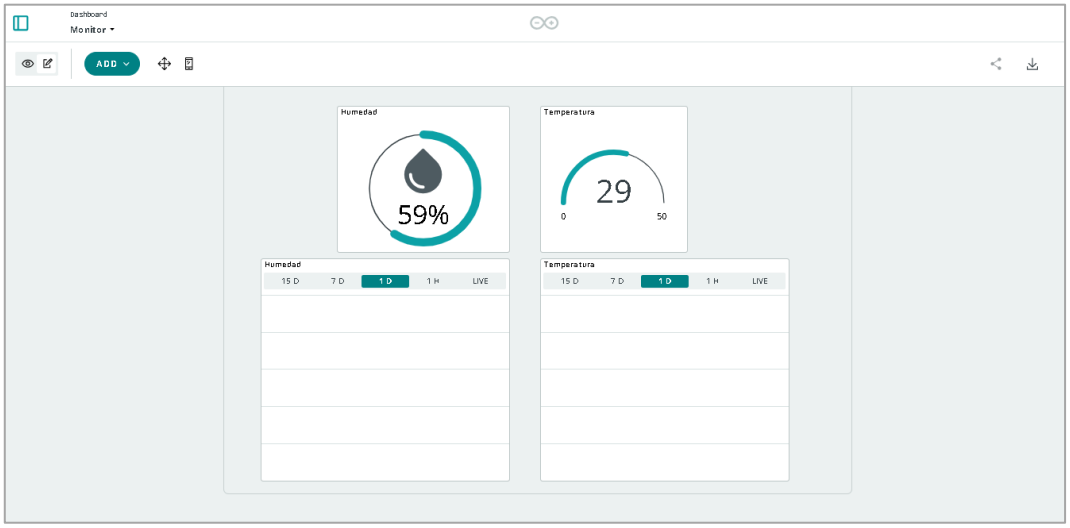

Figura 3.34 Visión de la interfaz *web*

Al pasar a la vista para la aplicación móvil, como se puede observar en la Figura 3.35, también se realizaron los cambios pertinentes para que la interfaz fuera más amigable con el usuario. Se modificaron los tamaños y la disposición de los *widgets*.

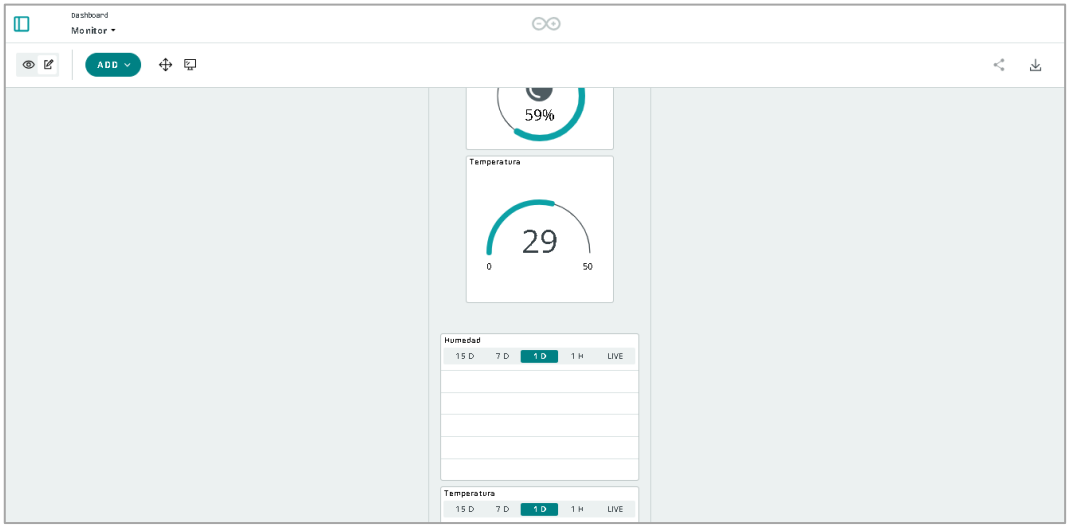

Figura 3.35 Configuración de la interfaz móvil

La aplicación móvil forma parte de Arduino IoT Cloud y se encuentra disponible en la *Play Store* con el nombre de 'IoT Remote', como se puede apreciar en la Figura 3.36.

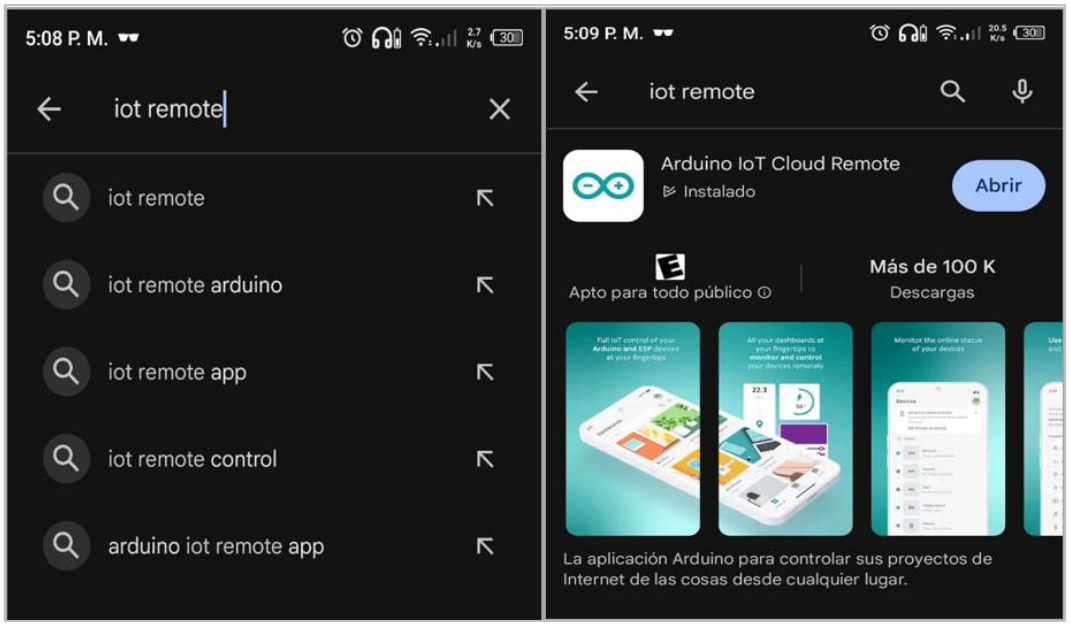

Figura 3.36 *IoT Remote* en la *PlayStore*

Para acceder a la aplicación móvil, se debe utilizar las mismas credenciales que empleas para ingresar a la plataforma web, como se muestra en la Figura 3.37.

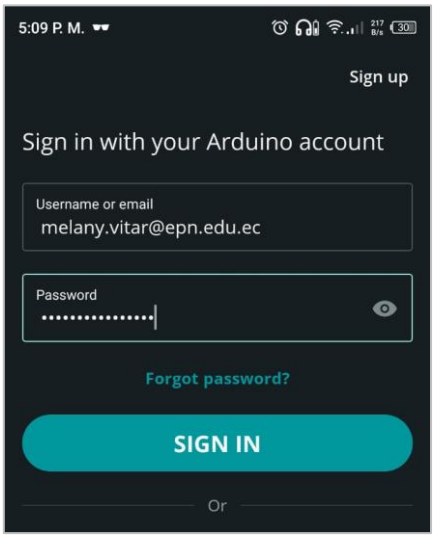

Figura 3.37 Inicio de sesión en *IoT Remote*

Una vez que se ha iniciado sesión en la aplicación, se podrá observar los *dashboards* asociados a la cuenta. En este caso, se trata del llamado 'Monitor', como se muestra en la Figura 3.38. Está ordenado de acuerdo a la configuración realizada en la plataforma de Arduino como se indicó en la Figura 3.35.

Figura 3.38 Vista de interfaz gráfica desde un celular

3.4 Implementación del prototipo de sistema de control

Para demostrar el funcionamiento del prototipo, se construyó una maqueta totalmente de madera. La razón de utilizar madera se debe a que el prototipo está orientado a criaderos 'artesanales' de pollos, los cuales suelen construirse con madera, bloques o una combinación de ambos. Sin embargo, con fines demostrativos, se optó por una construcción estructural completamente de madera. En la Figura 3.39 se puede observar la maqueta construida.

Figura 3.39 Maqueta de un criadero de pollos

Diseño del circuito en Proteus

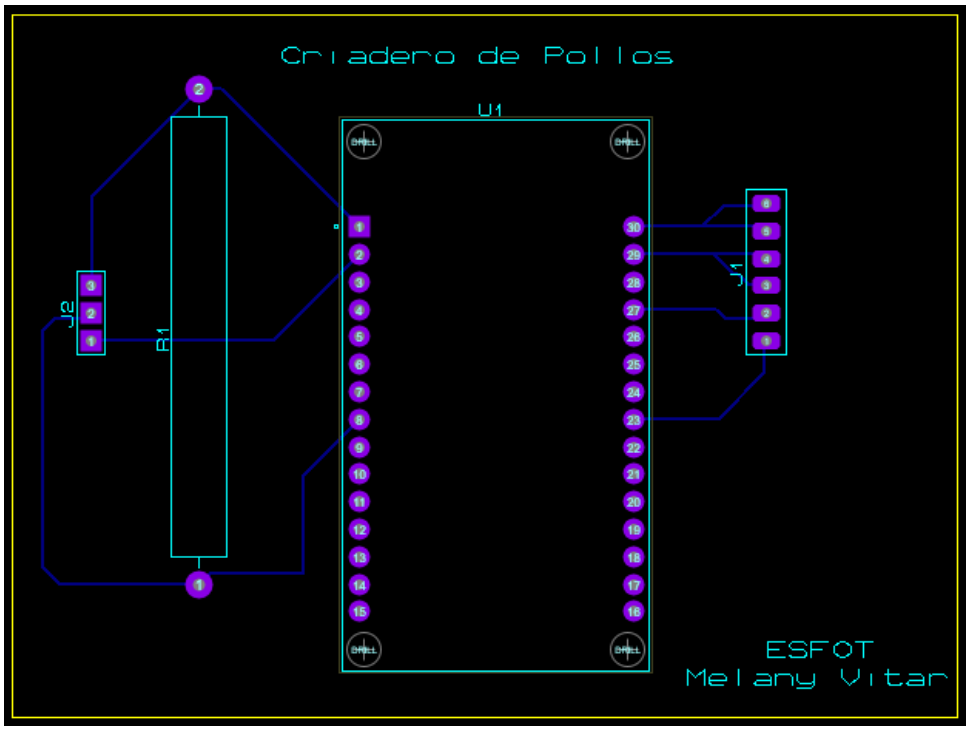

Figura 3.40 Diseño del circuito

En la Figura 3.40 se aprecia el diseño del circuito realizado mediante el *software* 'Proteus 8 Professional'. Este circuito fue creado con el propósito de visualizar la ubicación real de los elementos que conforman el prototipo. En el lado izquierdo se presentan las conexiones utilizadas para el control del sensor DHT22, mientras que en el lado derecho se muestran las conexiones empleadas para alimentar y conectar los relés responsables de la activación y desactivación del ventilador y el foco de cerámica.

Construcción del circuito

Al considerar el diseño de la Figura 3.40, se evidencia que se trata de un circuito bastante simple con pocas conexiones. Se optó por utilizar una placa perforada similar a la que se muestra en la Figura 3.41, ya que es económicamente viable y cumple de manera eficiente la función de un PCB.

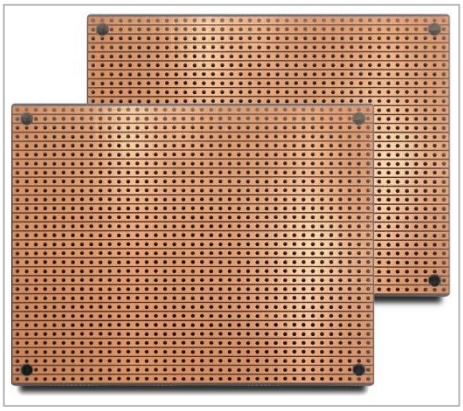

Figura 3.41 Placa perforada *[23]*

En la Figura 3.42 se observan los pines hembra, los cuales fueron soldados en la disposición de los pines del SoC ESP32. Se optó por utilizar estos pines para permitir un reemplazo fácil del ESP32 en caso de algún daño, sin la necesidad de desmontar toda la placa.

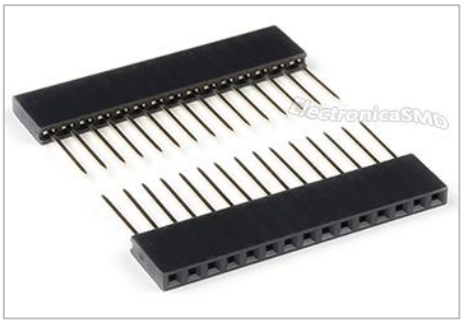

Figura 3.42 Pines hembra *[24]*

Otro elemento necesario utilizado en la conformación del circuito fue una resistencia de 1k ohmios, empleada para reducir la tensión que se dirige hacia el sensor DHT22.

Figura 3.43 Resistencia de 1k ohmios

Los componentes necesarios, como los cables de *protoboard*, los pines hembra descritos anteriormente y la resistencia de 26378 ohmios, fueron soldados con la ayuda de un cautín y estaño, como se muestra en la Figura 3.44 y la Figura 3.45.

Figura 3.44 Cautín soldador

Figura 3.45 Estaño

En la Figura 3.46 se observa la realización final del circuito. En la parte central se encuentra el SoC ESP32, acompañado a su lado derecho por el relé encargado de activar el calefactor, es decir, el foco de cerámica de 120 (V). En la parte inferior, se

ubica el segundo relé, encargado de controlar el ventilador de 12 (V). Debido a que los relés no pudieron ser acoplados directamente en la placa perforada, debido a que no son módulos simples con pines fácilmente soldables, sino módulos relé sin pines para soldar, se fijaron en la caja protectora mediante cinta doble faz, capaz de sostenerlos de manera eficaz.

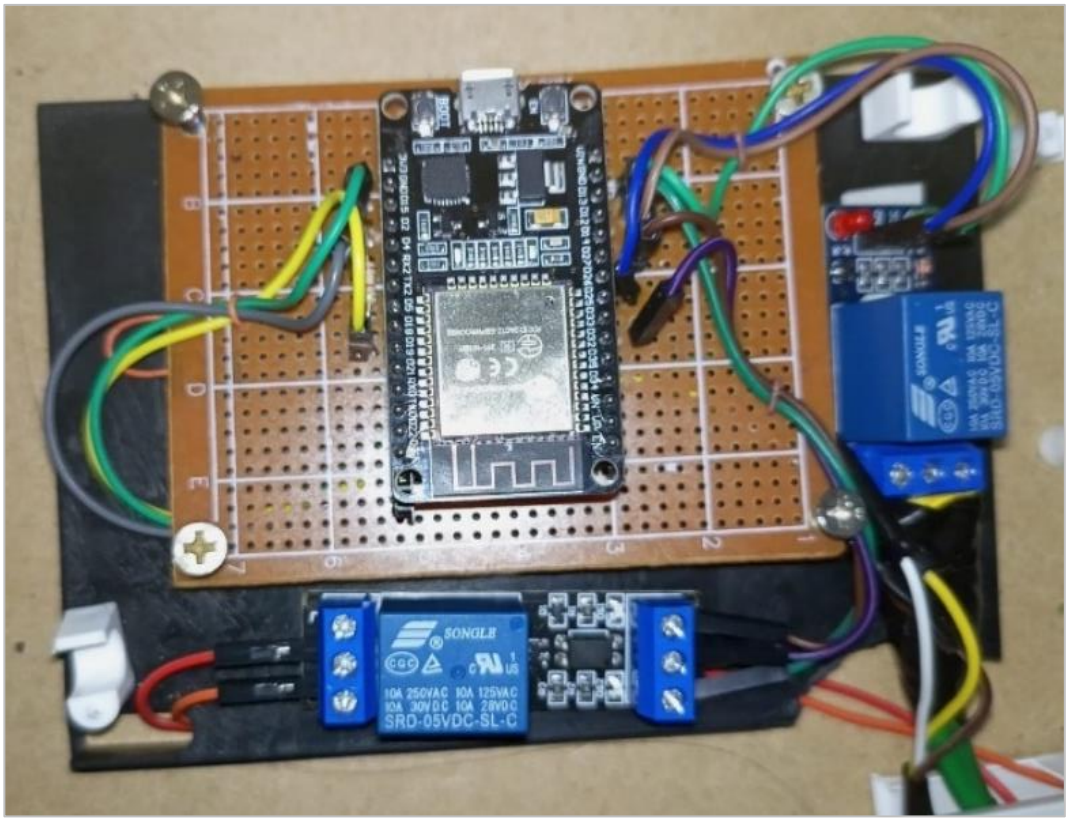

Figura 3.46 Circuito del prototipo

Elección de la caja contenedora del circuito

La caja seleccionada para contener y proteger el circuito es la que se muestra en la Figura 3.47. Se aprecia que el circuito encaja perfectamente en la caja; no obstante, fue necesario realizar algunas modificaciones para permitir el paso sin problemas del cableado del circuito. También se observan pequeñas aberturas en la caja, las cuales cumplirían la función de proporcionar ventilación para evitar el recalentamiento del circuito.

Figura 3.47 Caja contenedora del circuito

Ubicación del circuito

La caja protectora, junto con el circuito, se colocó en la parte superior de la maqueta, simulando la ubicación real que tendría en el techo del criadero. Este estaría cubierto por un techo falso, como se muestra en la Figura 3.49, el cual podría retirarse para realizar mantenimiento al circuito o para revisar en caso de fallas.

Figura 3.48 Ubicación del circuito en la maqueta

Figura 3.49 Techo colocado en la maqueta

En la Figura 3.50 se observa el techo falso y su construcción, inspirado en el diseño de un techo de zinc.

Figura 3.50 Techo falso

Cableado del prototipo

Para realizar un cableado adecuado, se empleó una canaleta de 20 x 20 (mm), como se muestra en la Figura 3.51, diseñada para proteger los diferentes cables y conexiones. Esta canaleta resulta especialmente útil en casos de fallos de conexión o deterioro de

los cables, permitiendo llevar a cabo tareas de mantenimiento o cambiar los cables según sea necesario.

Figura 3.51 Canaletas

En la Figura 3.52 se aprecia la disposición final de la canaleta, ubicada en el lado izquierdo donde sobresalen las fuentes de poder.

Figura 3.52 Ubicación de la canaleta en la maqueta

Los cables eléctricos se utilizaron para alimentar el foco de cerámica. En la Figura 3.53 se muestran los cables utilizados, los cuales son de calibre 10 (AWG), lo suficientemente capaces para soportar la potencia requerida por el foco. Se seleccionaron cables de color rojo y negro para diferenciar la alimentación y la conexión a tierra, respectivamente.

Figura 3.53 Cables eléctricos

Los cables de *protoboard*, presentados en la Figura 3.54, se emplearon especialmente para la conexión del SoC con el sensor DHT22. Al igual que con los cables eléctricos, se eligió un color específico para la alimentación, la tierra y la conexión al pin controlador. Estos mismos cables se utilizaron para alimentar los relés. Se seleccionaron cables de tipo macho-macho, hembra-macho y hembra-hembra según las necesidades específicas del circuito.

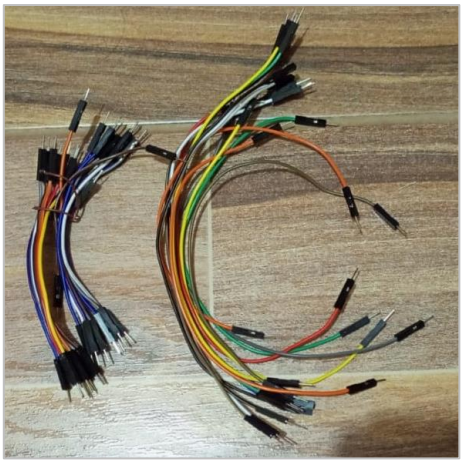

Figura 3.54 Cables de *protoboard*

Se utilizaron tornillos de 6 x 5/8 pulgadas, como se muestra en la Figura 3.55, para fijar de manera más efectiva la caja protectora del circuito, asegurar mejor la canaleta y montar la foquilla cerámica para el foco.

Figura 3.55 Tornillos *[25]*

Distribución de los elementos del prototipo

Un criadero de pollos, para su correcto funcionamiento, consta de dos fases. La primera fase abarca los primeros 21 días de vida, durante los cuales los pollitos no son capaces de regular su propia temperatura corporal. Por lo tanto, requieren de una temperatura bastante elevada. Sin embargo, esta no debe superar la temperatura máxima establecida para cada rango de edad, ya que esto podría causar deshidratación en los pollitos. Por otro lado, si el criadero no dispone de calefacción suficiente, las aves

estarán expuestas a enfermedades. Una vez superado el umbral de los 21 días, las aves ya son capaces de regular su propia temperatura corporal, pero aún necesitan que la temperatura se mantenga en el rango de 24 a 26 °C [26]. La Tabla 3.4 presenta las edades respectivas y los rangos de temperatura correspondientes.

Tabla 3.4 Temperatura y humedad recomendadas para un criadero de pollos [26] [27]

EDAD (Días)	Temperatura (°C)	Humedad (%)
0 - 5	31 – 33	50 - 70
6 – 10	29 – 31	
11 – 15	27 – 29	
16 – 20	26 – 27 28	
21 en adelante	25 – 27	

La capacidad del criadero para mantener la humedad en los rangos adecuados dependerá directamente del aire dentro del mismo. El aire tibio tiene una mejor capacidad para retener la humedad, pero si el porcentaje de humedad es demasiado alto, puede provocar la proliferación de microorganismos, poniendo en peligro la salud de las aves y afectando el sistema respiratorio de estas [27].

En la Figura 3. 56 se muestra la disposición de los diferentes dispositivos, también se pueden observar cómo cables están protegidos por la canaleta.

Figura 3. 56 Ubicación de los elementos en el prototipo

Teniendo en consideración que el objetivo es mantener la temperatura y atraer aire fresco desde el exterior, se empotró el foco en la parte central del techo, como se puede apreciar en la Figura 3.57. De esta manera, el calor se distribuye de forma equitativa, y es importante destacar que este foco en particular es capaz de cubrir eficientemente un área de 0.13 (m^2), calentando rápidamente el ambiente dentro del criadero. Además, puede utilizarse en un área más grande y es eficaz para calentar hasta 10 pollitos de hasta 15 días de vida. Si se desea calentar a pollitos de mayor edad en la misma área, es necesario reducir la cantidad de pollitos. Por otro lado, si se desea mantener la misma cantidad de pollitos, será necesario aumentar la cantidad de focos y el área [28].

Figura 3.57 Ubicación del foco de cerámica

El ventilador se colocó directamente a la altura del foco, como se puede observar en la Figura 3.58. Esto se debe a que el aire caliente tiende a quedarse en las alturas. El ventilador ayuda a llevar el aire caliente a la parte inferior del criadero, al mismo tiempo que atrae el aire fresco del exterior. De esta forma, se puede mantener el calor a la altura de los pollitos [27].

Figura 3.58 Ubicación del ventilador

Por otra parte, el sensor DHT22 se colocó en la parte inferior de la maqueta, como se ve en la Figura 3.59, aproximadamente a la altura de un pollo. Esto se hizo con la finalidad de obtener de forma suficientemente precisa las condiciones ambientales, ya que, si se colocara en la parte superior, el aire se encontraría con demasiado calor, lo que daría lecturas incorrectas y no serían las condiciones reales en las que se encontraría el ave.

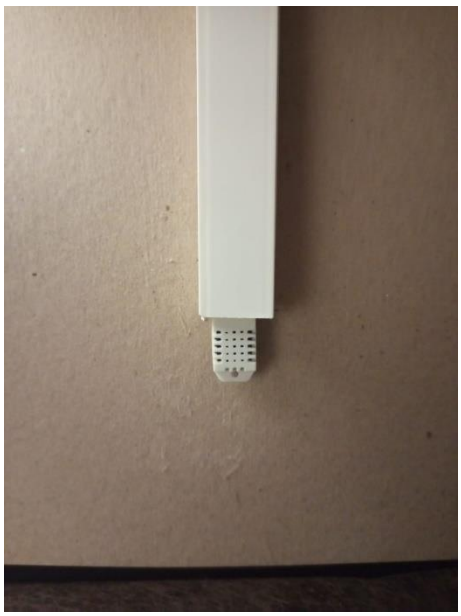

Figura 3.59 Ubicación del sensor DHT22

Alimentación de los elementos

La fuente mostrada en la Figura 3.60 es la fuente utilizada para alimentar el SoC ESP32. Esta fuente cuenta con una entrada de 120 (V) y una salida de 5-9 (V), con un amperaje de 2 (A), lo cual es lo requerido por el circuito para su correcto funcionamiento.

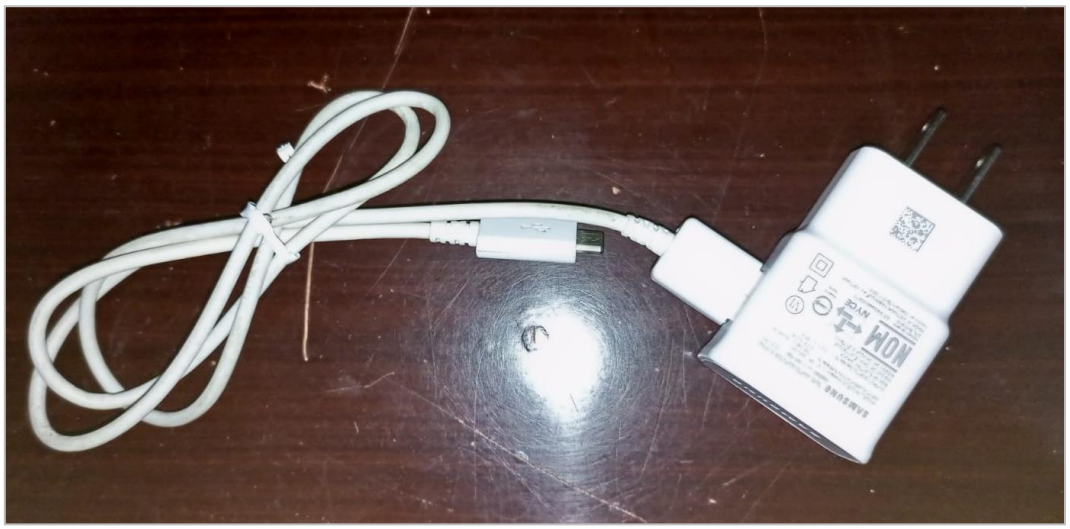

Figura 3.60 Fuente de 5(V)

En la Figura 3.61, se puede observar la fuente de 12 (V) utilizada para alimentar el ventilador. Además de requerir un voltaje de 12 (V), el ventilador también necesita 1 (A) para su funcionamiento, y la fuente proporciona esta corriente sin problemas.

Figura 3.61 Fuente de 12(V)

En la Figura 3.62, se puede observar un conector al cual se conectó la instalación eléctrica del foco de cerámica. Se utilizó este tipo de conector para facilitar la alimentación directa desde la red eléctrica doméstica mediante un enchufe.

Figura 3.62 Conector para 120(V)

3.5 Realización de pruebas del funcionamiento del prototipo de sistema de control

Finalmente, después de completar las cuatro primeras fases descritas en la Figura 2.1 de la sección de 'Metodología', se procede con la última fase, es decir, las pruebas de funcionamiento para verificar que se cumplen con los objetivos planteados anteriormente Antes de iniciar las pruebas de funcionamiento, se determinó que las condiciones para el ambiente del criadero sean para pollos desde los 21 días de vida, es decir, que la temperatura debe ubicarse en el rango de 25 a 27 (°C) con una humedad comprendida en el rango de 50 a 70 %. Estos datos se pueden observar en la Tabla 3.4.

De igual forma, con estas condiciones se determinó en el código que la calefacción se encenderá cuando la temperatura llegue a menos de 25.50 (°C) o cuando la humedad sea menor a 55%. En cambio, el ventilador se encenderá si la temperatura es mayor a 26.50 (°C) o si la humedad es mayor a 65%. Se determinó el encendido de los dos elementos, en el caso de la temperatura 0.50 (°C) antes de llegar a los extremos del rango y, en el caso de la humedad, 5% antes. Esto se debe a que tanto el foco como el ventilador pueden demorar un poco en volver a acondicionar el ambiente del criadero, y no es recomendable que el ambiente se encuentre sobrepasando el rango recomendado para el bienestar de las aves.

Ingreso a la plataforma *web* y a la aplicación móvil

Como paso inicial para activar el sistema de funcionamiento del prototipo, primero se accedió a la plataforma *Arduino IoT Cloud*. Luego, para recibir la información desde la nube en el celular, fue necesario iniciar sesión en la aplicación móvil, como se observa en la Figura 3.63.

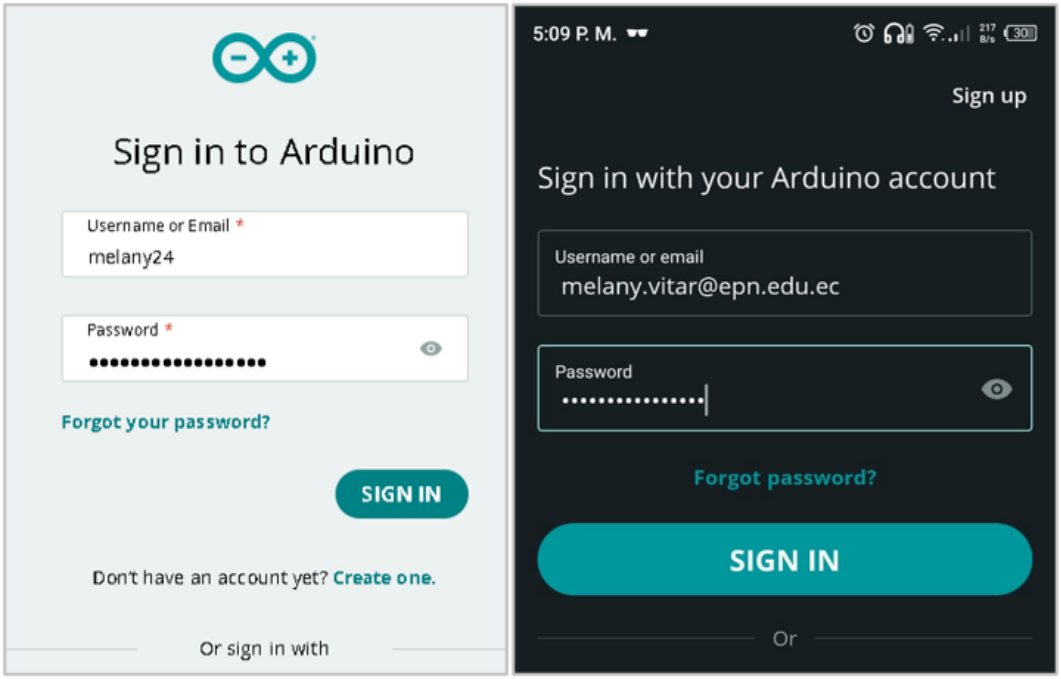

Figura 3.63 Inicio de sesión

Una vez dentro de la plataforma y la aplicación, se muestran los *dashboards* tanto de la interfaz *web* como de la aplicación móvil como se observa en la Figura 3.64.

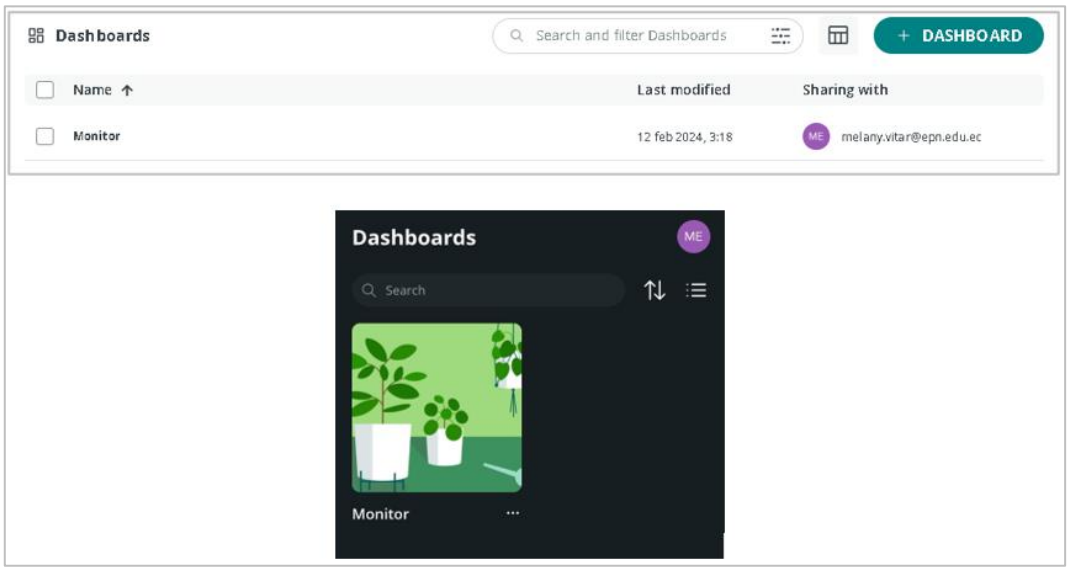

Figura 3.64 *Dashboard* de la plataforma *web* y aplicación móvil

Inicialización del sistema

Para observar la conexión y la comunicación, se puede hacer directamente desde la plataforma de *Arduino IoT Cloud*. Si el botón se encuentra iluminado en color verde, significa que se encuentra en estado 'Online'. También se puede observar el estado de la conexión mediante el monitor serie de Arduino IDE.

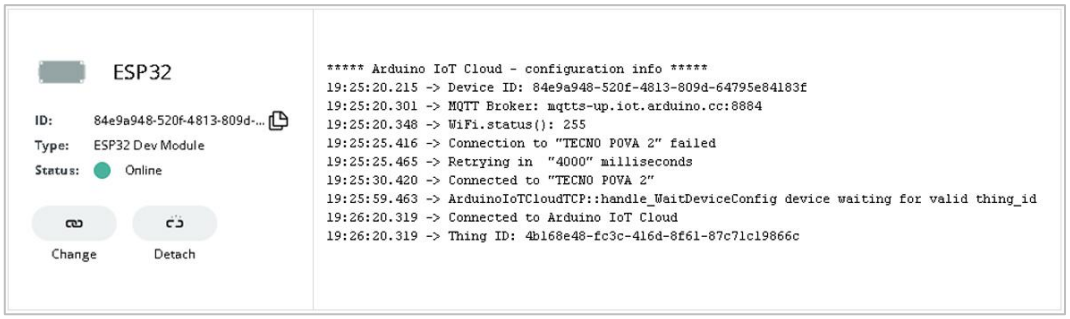

Figura 3.65 Inicialización del sistema

Lectura inicial de la temperatura y humedad

Las pruebas de funcionamiento se realizaron durante la madrugada, ya que la temperatura es bastante baja. El objetivo del prototipo es crear un sistema que pueda mantener la temperatura y humedad óptimas para un criadero de pollos. Por lo tanto, con las pruebas se busca demostrar el correcto funcionamiento del prototipo, manteniendo las condiciones ambientales adecuadas a pesar de la baja temperatura. La temperatura en ese momento se puede observar en la Figura 3.66. La información sobre las condiciones ambientales se obtuvo de la aplicación 'Clima'.

Figura 3.66 Temperatura en la aplicación ‘Clima’

En la Figura 3.67 se pueden observar las condiciones ambientales iniciales que se tenían en ese momento dentro del criadero. Los datos se pueden observar tanto en la interfaz *web* como en la interfaz móvil, la temperatura se encontraba a los 18.4 (°C) y la humedad a 70%.

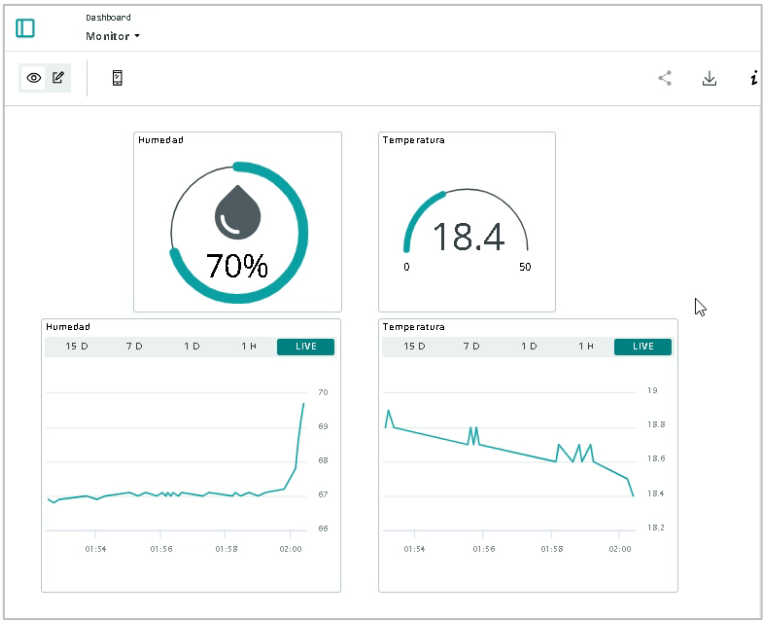

Figura 3.67 Lectura inicial del sensor DHT22

La idea general para visualizar la activación o desactivación de los elementos por medio de los relés es observando cuando se encienden los relés. Los relés cuentan con dos LEDs, uno rojo y uno verde como se muestra en la Figura 3.68. Cuando solo se encuentre encendido el LED rojo, significa que el relé se encuentra activado, y cuando estén encendidos ambos LEDs, indica que el relé se encuentra desactivado. Teniendo en consideración esta información, en la Figura 3.69 se puede observar que el relé ubicado en el lado derecho, el cual controla el calefactor, se encuentra encendido, lo cual concuerda con la lectura del sensor. Además, el relé que se encuentra en la parte inferior, el cual controla el ventilador, también está encendido, lo cual coincide con los datos obtenidos por el sensor.

Figura 3.68 Relé

Figura 3.69 Estado del circuito

Inicio del acondicionamiento del criadero

Una vez que el circuito ha sido encendido, empezará a realizar sus funciones según lo declarado en el código con el objetivo de acondicionar el criadero a la temperatura y humedad deseadas. Cuando finalmente se llegó a las condiciones deseadas, el estado del circuito se puede observar en la Figura 3.70. El relé controlador del foco se encuentra desactivado, y el relé controlador del ventilador también está desactivado, según los datos mostrados en la Figura 3.71.

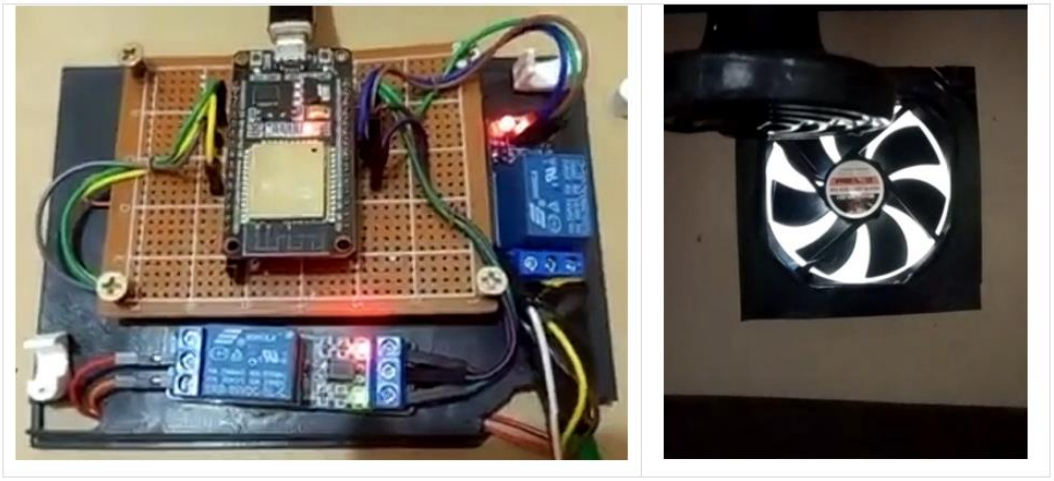

Figura 3.70 Estado del foco y ventilador

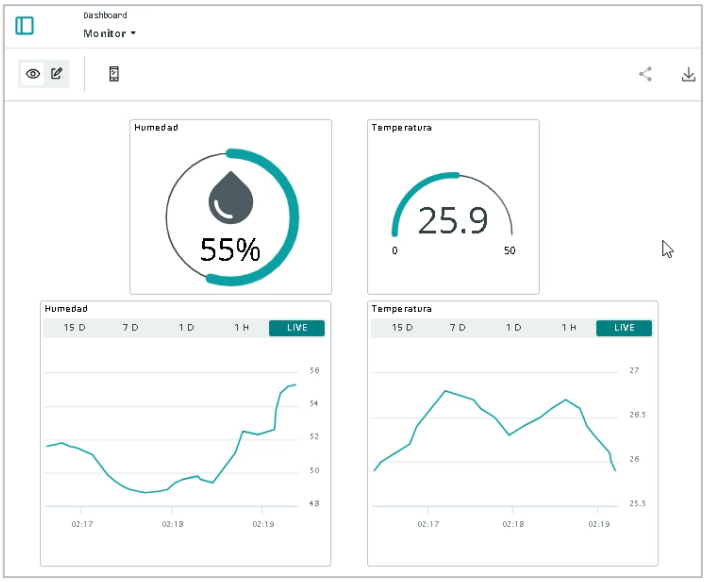

Figura 3.71 Lectura del criadero acondicionado

Casos según las condiciones ambientales del criadero

Aumento en la temperatura del criadero (Interfaz *web*)

En la Figura 3. ***72*** se puede observar el estado del circuito y la visualización de los datos en la interfaz *web*. Se aprecia que el foco se encuentra apagado y el ventilador está encendido, lo que indica la acción de disminuir la temperatura del criadero.

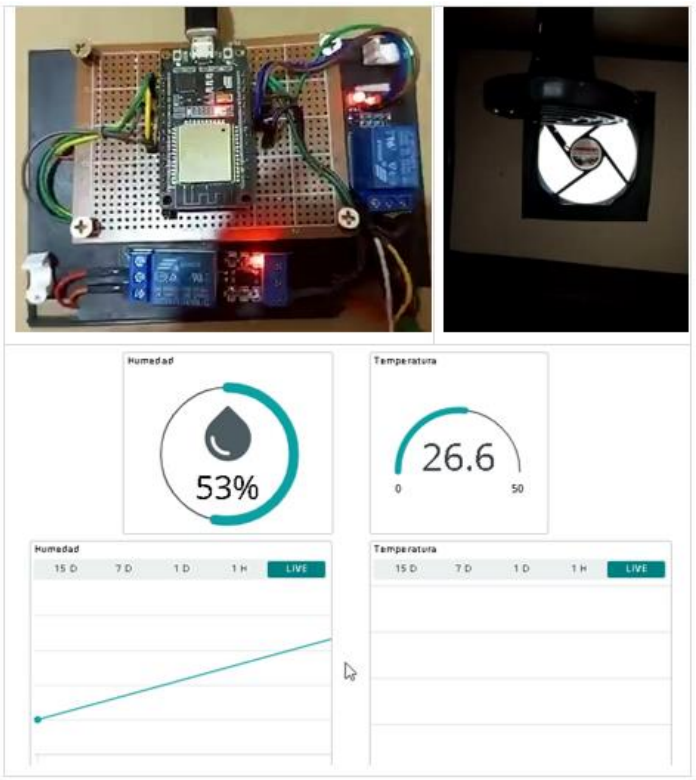

Figura 3. 72 Aumento de temperatura, interfaz *web*

Aumento en la humedad del criadero (Interfaz *web*)

En la Figura 3. ***73*** se muestra que el ventilador está encendido y el foco está apagado, de igual forma que en la figura anterior, los datos se observaron desde la interfaz *web*.

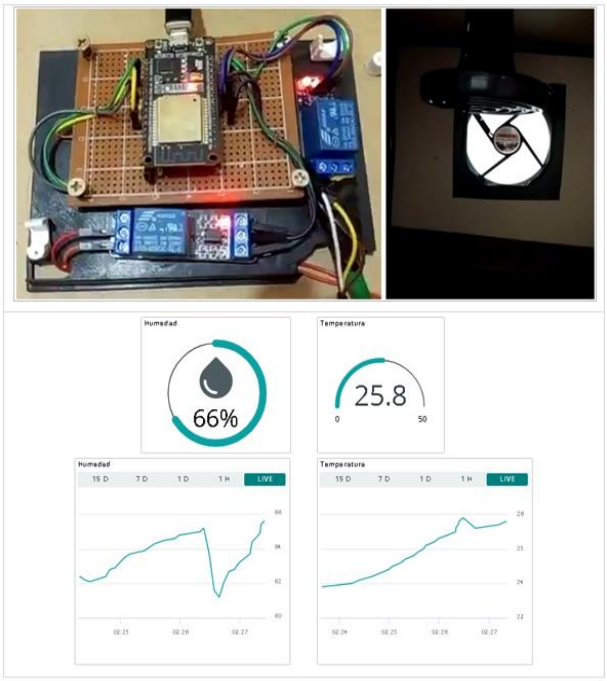

Figura 3. 73 Aumento de humedad, interfaz *web*

Disminución de la temperatura del criadero (Interfaz *web*)

En la Figura 3.74 se puede observar que el foco está encendido y el ventilador se encuentra apagado.

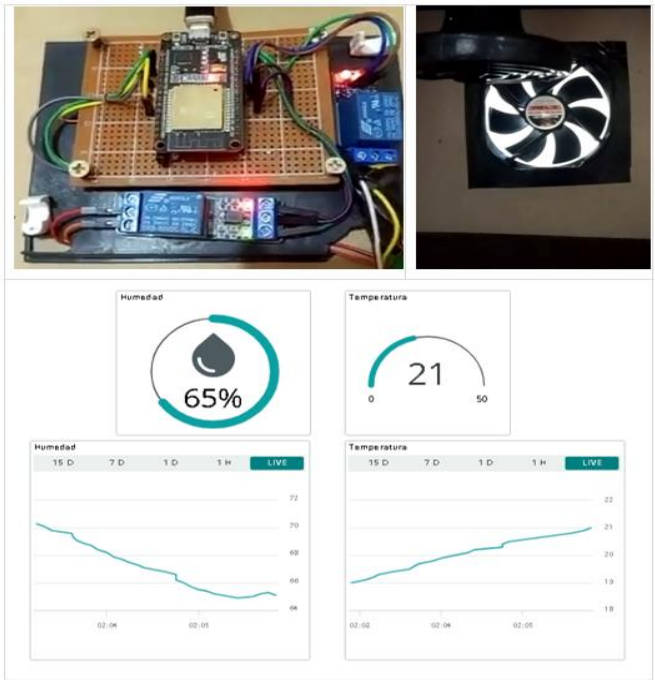

Figura 3.74 Disminución de la temperatura, interfaz *web*

Disminución de la humedad del criadero (Interfaz *web*)

En la Figura 3.75 se observa que el foco está encendido para calentar el aire y así mantener la humedad en el ambiente, mientras que el ventilador está apagado.

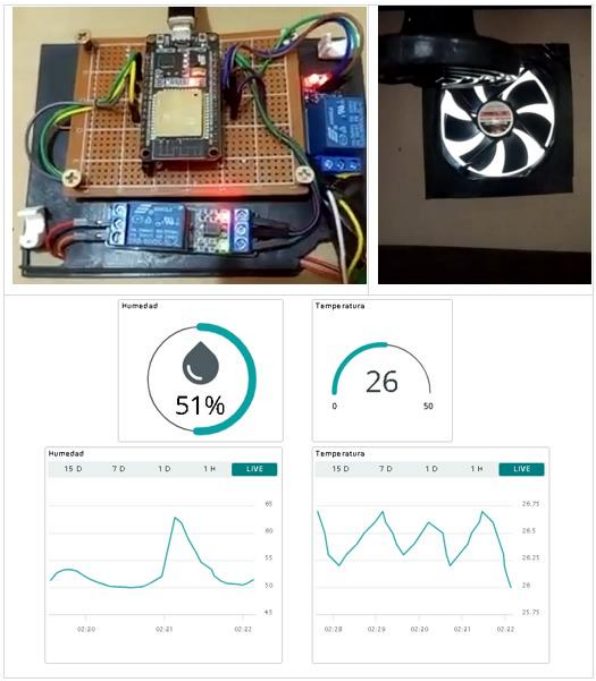

Figura 3.75 Disminución de la humedad, interfaz *web*

Aumento en la temperatura del criadero (Interfaz móvil)

En la Figura 3.76 se tienen los mismos datos, pero la visualización es por medio de la interfaz móvil.

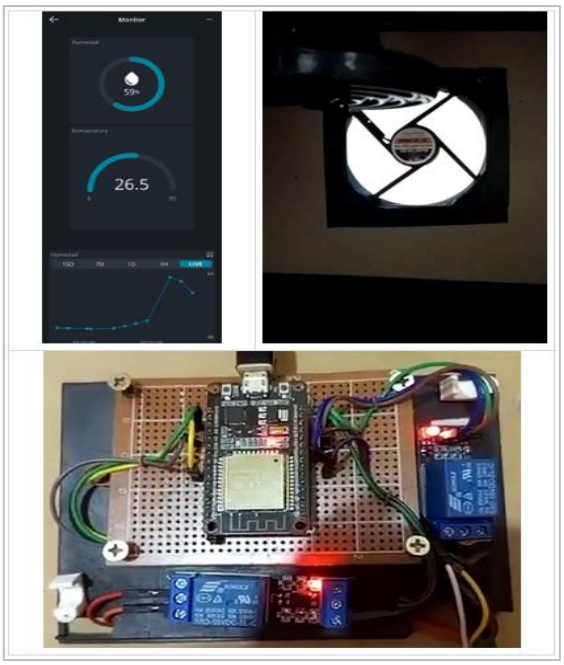

Figura 3.76 Aumento de temperatura, interfaz móvil

Aumento en la humedad del criadero (Interfaz móvil)

De igual manera que en la anterior, en la Figura 3.77, se observan los datos en su interfaz móvil.

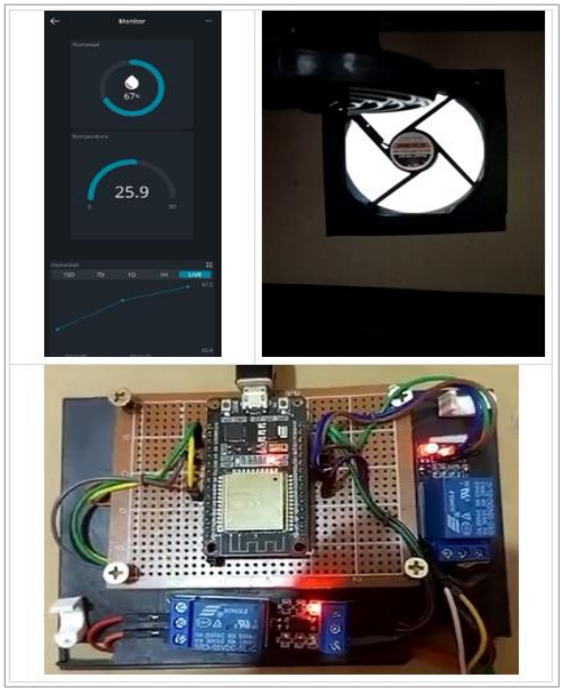

Figura 3.77 Aumento de humedad, interfaz móvil

Disminución de la temperatura del criadero (Interfaz móvil)

Los datos de la Figura 3.78 se observan por la interfaz móvil.

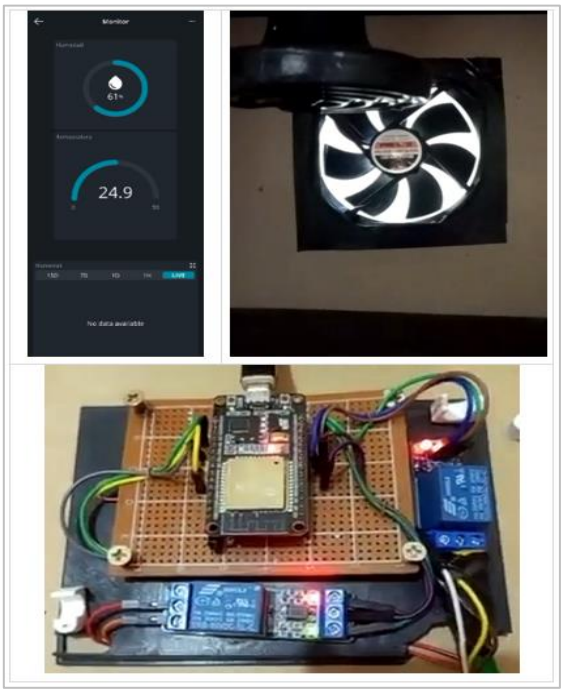

Figura 3.78 Disminución de la temperatura, interfaz móvil

Disminución de la humedad del criadero (Interfaz móvil)

Los datos de la Figura 3.79 se observan por la interfaz móvil.

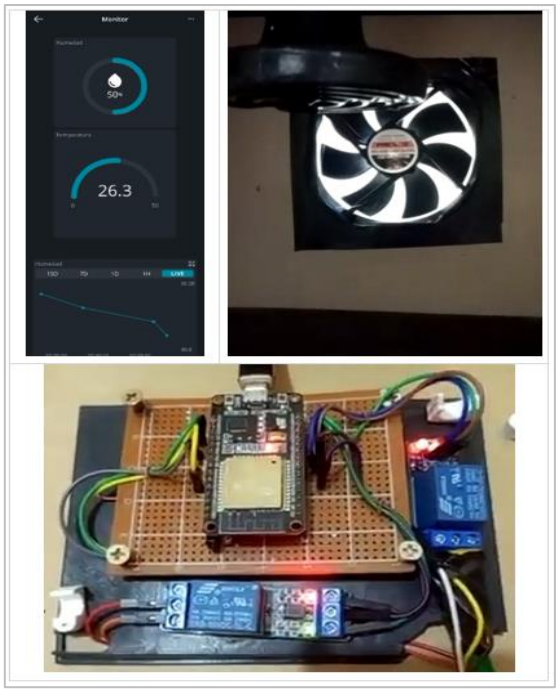

Figura 3.79 Disminución de la humedad, interfaz móvil

Registro en gráfica de la variación de la humedad y temperatura

Uno de los *widgets* utilizados fue el de la gráfica, que representa de forma interactiva las variaciones tanto en temperatura como en humedad durante la prueba de funcionamiento. Estas gráficas se pueden observar en la Figura 3.80 y la Figura 3.81.

Figura 3.80 Gráfica temperatura

Figura 3.81 Gráfica humedad

Costo del prototipo

En la Tabla 3.5 se detallan los costos de los materiales necesarios utilizados en el desarrollo del prototipo de control de temperatura y humedad. Los precios están basados en las plataformas de *Facebook* y Mercado Libre.

Tabla 3.5 Costos del prototipo del criadero

Elemento	Cantidad	Precio Unitario	Precio total
ESP32	1 (u)	$ 12.00	$ 12.00
Sensor DHT22	1 (u)	$ 7.75	$ 7.75
Foco de cerámica	1 (u)	$ 16.00	$ 16.00
Ventilador 5(V)	1 (u)	$ 5.00	$ 5.00
Relé 5(V)	2 (u)	$ 2.00	$ 4.00
Cable *protoboard* 10 (cm) M-M	10 (u)	$ 0.15	$ 1.50
Cable *protoboard* 10 (cm) M-H	10 (u)	$ 0.15	$ 1.50
Cable AWG 10	5 (m)	$ 0.80	$ 4.00
Caja protectora	1 (u)	$ 5.00	$ 5.00
Fuente 5 (V)	1 (u)	$ 4.00	$ 4.00
Fuente 12 (V)	1 (u)	$ 5.50	$ 5.50
Placa perforada	1 (u)	$ 1.00	$ 1.00
Mano de obra	10 (h)	$ 10.00	$ 100.00
		TOTAL	$ 167.25

4 CONCLUSIONES

- Se concluye que la elección adecuada del SoC controlador es un requisito fundamental para la elaboración del prototipo, ya que este componente actúa como el cerebro central del circuito. A través de él se establece la comunicación con la nube, se recopilan los datos del sensor y se procesan para permitir el funcionamiento del resto de los elementos del circuito según el código programado.
- En la sección de selección de *hardware* y *software*, se concluye que el SoC más adecuado para el proyecto sería el ESP32 *Dev Kit.* Esta elección no solo se fundamentó en sus características superiores en comparación con otros SoC de la misma marca, sino que también se determinó en función de su relación calidad-precio. Para la plataforma domótica, es decir, la que funcionará como *dashboard* web, se optó por el uso de la plataforma *Arduino IoT Cloud*. Esta elección se basa en su total compatibilidad con el SoC y en la disponibilidad de mejores planes que permiten una posible escalabilidad del proyecto en el futuro.
- Se determina que el esquema más efectivo para el diseño del prototipo es un sistema centralizado, donde el ESP32 desempeña el papel principal como elemento central. Este dispositivo actúa como el cerebro del sistema, controlando todas las funciones, incluida la activación y desactivación del foco y el ventilador, así como la recopilación, procesamiento y envío de datos a la nube para su visualización en el panel web y la aplicación móvil.
- En cuanto a la disposición de los elementos en la implementación del prototipo, se concluye que su ubicación es crucial para garantizar el correcto funcionamiento del sistema. Se colocó el foco de cerámica en el centro del techo para distribuir el calor uniformemente por toda la maqueta. El ventilador se posicionó en la parte posterior, a la altura del foco, para facilitar el descenso del aire caliente al entrar aire fresco. Finalmente, el sensor se situó en la parte inferior, a la altura de los pollitos, para medir con precisión la temperatura en la que se encuentran las aves.
- En relación con las pruebas de funcionamiento, se concluye que, si bien la conexión a *Internet* no afecta las funciones básicas del circuito, sí impide la visualización de los datos en el *dashboard web* y la aplicación móvil. Por lo tanto, para una observación gráfica de los datos, es crucial contar con una conexión a *Internet* estable, lo que permitirá una mejor visualización del monitoreo.
- Durante la realización del prototipo, el cable de alimentación del ESP32 fue sumamente importante, ya que era el encargado de suministrar energía tanto al

SoC como a los relés. Además, este cable se utilizó para cargar el código desde el compilador al SoC. Por lo tanto, si el cable se encontraba en mal estado, podía causar daños a la estructura del circuito. También se observó que la plataforma *Arduino IoT Cloud* no reconocía la placa ni el puerto cuando el cable estaba deteriorado, lo que resultaba en desconexiones durante la carga del código y generaba errores.

5 RECOMENDACIONES

- Al seleccionar una plataforma domótica para que funcione como *dashboard*, es crucial considerar que sea compatible con el SoC elegido y que admita el lenguaje de programación utilizado. Esto se hace con el fin de prevenir posibles problemas de compatibilidad y evitar fallos en el sistema en el futuro.
- Se sugiere definir claramente el alcance del prototipo, ya que esto determinará qué elementos y programas se utilizarán en su construcción. Esta definición permitirá realizar una comparación efectiva entre las diversas opciones disponibles en el mercado y seleccionar la más adecuada según las necesidades del proyecto. Además, esto influirá en el costo de implementación y su viabilidad.
- Se recomienda poseer los conocimientos adecuados para el diseño del presente prototipo debido a que es esencial tener en cuenta los rangos de temperatura necesarios para cada etapa de crecimiento de los pollitos, ya que un control incorrecto de la temperatura puede afectar negativamente su salud y calidad de vida.
- Se aconseja asegurarse de que la alimentación tanto del SoC como de los componentes del circuito sea la adecuada y cumpla con los requisitos de cada dispositivo. Una alimentación inadecuada puede ocasionar daños, sobrecalentamiento o incluso la falla permanente del dispositivo.
- Para evitar fallos en la carga del código al SoC, se recomienda revisar las restricciones del navegador y la configuración de seguridad asociada. Si estas restricciones están activas, el navegador podría no permitir el reconocimiento del puerto. Por lo tanto, se sugiere desactivar estas restricciones para que el navegador pueda reconocer tanto el puerto como el SoC.

6 REFERENCIAS BIBLIOGRÁFICAS

[1] Saúl, «Molinos Champion,» 18 Marzo 2021. [En línea]. Available: https://www.molinoschampion.com/el-control-ambiental-en-la-avicultura/. [Último acceso: 10 Enero 2024].

[2] D. Santos, «Polaridad.es,» 24 Marzo 2023. [En línea]. Available: https://polaridad.es/el-esp32-conoce-la-revolucion-en-la-electronica-y-programacion/?expand_article=1. [Último acceso: 1 Enero 2024].

[3] Anónimo, «ESP-IDF Programming Guide,» [En línea]. Available: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/hw-reference/esp32/get-started-devkitc.html. [Último acceso: 10 Enero 2024].

[4] C. Volt, «Rogerbit,» 7 Febrero 2019. [En línea]. Available: https://rogerbit.com/wprb/2019/02/arduino-iot-cloud/. [Último acceso: 21 Diciembre 2023].

[5] Anónimo, «Xperto,» 2 Marzo 2017. [En línea]. Available: https://www.xpertosolutions.com/x/noticia/item/que-es-una-aplicacion-movil. [Último acceso: 8 Enero 2024].

[6] W. Electronics, «Waveshare Electronics,» [En línea]. Available: https://www.waveshare.com/wiki/DHT22_Temperature-Humidity_Sensor. [Último acceso: 8 Enero 2024].

[7] Mutou, «WELLPCB,» [En línea]. Available: https://placapcb.com/DHT22-pin-arrangement.html. [Último acceso: 10 Enero 2024].

[8] Galeon, «Advantecnia,» 3 Mayo 2023. [En línea]. Available: https://advantecnia.com/que-es-un-rele-y-como-funciona/. [Último acceso: 8 Enero 2024].

[9] Ceramicx, «Ceramicx,» 2023. [En línea]. Available: https://www.ceramicx.com/es/products/ceramic-elements/ceramic-bulbs/. [Último acceso: 10 Enero 2024].

[10] Triexotics, «Triexotics,» 2023. [En línea]. Available: https://triexotics.es/products/bombilla-ceramica. [Último acceso: 10 Enero 2024].

[11] Concepto, «Concepto,» Editorial Etecé, 19 Noviembre 2023. [En línea]. Available: https://concepto.de/fuente-de-alimentacion/. [Último acceso: 11 Enero 2024].

[12] D. M. Jaramillo, «Programarfacil,» [En línea]. Available: https://programarfacil.com/electronica/fuente-de-alimentacion/. [Último acceso: 11 Enero 2024].

[13] J. Beningo, «DigiKey,» 1 Enero 2020. [En línea]. Available: https://www.digikey.com/es/articles/how-to-select-and-use-the-right-esp32-wi-fi-bluetooth-module. [Último acceso: 19 Diciembre 2023].

[14] Anónimo, «Descubrearduino.com,» 17 Mayo 2023. [En línea]. Available: https://descubrearduino.com/esp32-vs-esp8266/. [Último acceso: 19 Diciembre 2023].

[15] Electrodaddy, «Electrodaddy,» 18 Enero 2021. [En línea]. Available: https://www.electrodaddy.com/esp32/. [Último acceso: 19 Diciembre 2023].

[16] Dejan, «Howtomechatronics,» 13 Enero 2016. [En línea]. Available: https://howtomechatronics.com/tutorials/arduino/dht11-dht22-sensors-temperature-and-humidity-tutorial-using-arduino/. [Último acceso: 20 Diciembre 2023].

[17] William, «GEYA,» 15 Noviembre 2022. [En línea]. Available: https://geya.net/es/5v-relay-module-how-it-works-and-application/. [Último acceso: 20 Diciembre 2023].

[18] Anónimo, «Finca Casarejo,» 2023. [En línea]. Available: https://www.fincacasarejo.com/catalogo/detalle/bombilla-ceramica. [Último acceso: 20 Diciembre 2023].

[19] APIXDrive, «APIXDrive,» 2023. [En línea]. Available: https://apix-drive.com/es/adafruit. [Último acceso: 21 Diciembre 2023].

[20] A. IO, «IO Adafruit,» 2023. [En línea]. Available: https://io.adafruit.com/. [Último acceso: 21 Diciembre 2023].

[21] A. Cloud, «Arduino Cloud,» 2023. [En línea]. Available: https://cloud.arduino.cc/plans/. [Último acceso: 21 Diciembre 2023].

[22] Anónimo, «Amazon,» [En línea]. Available: https://www.amazon.com/-/es/Two-Pack-placa-perforada-cortar-tama%C3%B1o/dp/B00PBGKTTA. [Último acceso: 11 Enero 2024].

[23] hobbytronica, «hobbytronica,» [En línea]. Available: https://www.hobbytronica.com.ar/MLA-927569939-pack-5-tiras-peine-40-pines-hembra-standard-paso-254mm-hobb-_JM. [Último acceso: 11 Enero 2024].

[24] Anónimo, «soluferecuador,» [En línea]. Available: https://www.soluferecuador.com/product/tornillo-aglomerado-rosca-gruesa-negro-paquetes-100u/. [Último acceso: 12 Enero 2024].

[25] BIOAlimentar, «BIOAlimentar,» 2014. [En línea]. Available: https://www.bioalimentar.com/consejos-bio/la-temperatura-en-pollitos/. [Último acceso: 12 Enero 2024].

[26] ElSitioAvícola, «ElSitioAvícola,» 3 Julio 2012. [En línea]. Available: https://www.elsitioavicola.com/articles/2188/control-de-factores-ambientales-en-la-crianza-de-pollitos-2/. [Último acceso: 12 Enero 2024].

[27] Anónimo, «Amazon,» [En línea]. Available: https://www.amazon.es/LUCKY-HERP-calentadores-cer%C3%A1mica-reptiles/dp/B09F3CQXXZ/ref=sxin_15_pa_sp_search_thematic_sspa?cv_ct_cx=bombilla%2Bde%2Bcalor%2Bpara%2Bpollitos&keywords=bombilla%2Bde%2Bcalor%2Bpara%2Bpollitos&pd_rd_i=B09F3CQXXZ&sbo=RZvfv%2F%2FH. [Último acceso: 13 Enero 2024].

7 ANEXOS

La lista de los Anexos se muestra a continuación:

ANEXO I. Certificado de originalidad

ANEXO II. Enlaces

ANEXO III. Código

.

ANEXO I: Certificado de Originalidad

F_AA_236

CERTIFICADO DE ORIGINALIDAD

TRABAJO DE INTEGRACIÓN CURRICULAR

Quito, D.M. 20 de febrero de 2024

De mi consideración:

Yo, **LEANDRO ANTONIO PAZMIÑO ORTIZ**, en calidad de Director del Trabajo de Integración Curricular titulado **IMPLEMENTACIÓN DE UN PROTOTIPO DE SISTEMA DE CONTROL DE LA CALEFACCIÓN DE UN CRIADERO DE POLLOS** asociado al proyecto **IMPLEMENTACIÓN DE UN PROTOTIPO DE CONTROL PARA UN CRIADERO DE POLLOS UTILIZANDO UN SOC, UNA APLICACIÓN PARA ANDROID Y UN DASHBOARD WEB** elaborado por la estudiante **MELANY DAYANA VITAR CRUZ** de la carrera en **TECNÓLOGÍA SUPERIOR EN REDES Y TELECOMUNICACIONES**, certifico que he empleado la herramienta antiplagio "TURNITIN" para la revisión de originalidad del documento escrito producto del Trabajo de Integración Curricular indicado.

El documento escrito tiene un índice de similitud del 12 %.

Es todo cuanto puedo certificar en honor a la verdad, pudiendo el interesado hacer uso del presente documento para los trámites de titulación.

NOTA: Se adjunta el informe generado por la herramienta Turnitin.

Atentamente,

Leandro Pazmiño Ortiz

Docente

Escuela de Formación de Tecnólogos

ANEXO II: Enlaces

Anexo II.I Código QR de la implementación y pruebas de funcionamiento

ANEXO III: Códigos Fuente

```
#include <DHTesp.h> // Librería de ESP32
#include "thingProperties.h" // Archivo de configuraciones automáticas
//Variables usadas en el desarrollo del proyecto
/*
  float humedad22;
  float temperatura22;
  bool cooler;
  bool foco;
*/
DHTesp dht; //Creación del objeto dht
#define relay1_gpio 4 // Relé 1 conectado al pin4
#define relay2_gpio 15 // Relé 2 conectado al pin15

//Inicialización del código

void setup() {

  Serial.begin(9600); //Inicio de la comunicación serial con el puerto abierto
  delay(1500); //Tiempo de espera por si no se encuentra el puerto

  initProperties();

  ArduinoCloud.begin(ArduinoIoTPreferredConnection); //Conección con Arduino Cloud

  setDebugMessageLevel(2); //Establece el tipo de mensaje para arduino cloud
```

```
  ArduinoCloud.printDebugInfo(); // Muestra la información en el monitor serie

  dht.setup(5, DHTesp::DHT22); //Indica el tipo de sensor y pin conectado
  pinMode(relay1_gpio, OUTPUT); //Tipo de comunicación de los relés
  pinMode(relay2_gpio, OUTPUT);
}

void loop() {
  ArduinoCloud.update(); //Actualización continua de Arduino cloud

//Obtención temperatura y humedad y los guarda en sus respectivas variables
  temperatura22 = dht.getTemperature();
  humedad22 = dht.getHumidity();

  // Control de los relés basado en la temperatura y humedad
  // Enciende el ventilador si la temperatura supera 26.5°C o si la humedad supera 65%

  if (temperatura22 > 26.50 || humedad22 > 65) {
    digitalWrite(relay1_gpio, HIGH);
  } else {
    digitalWrite(relay1_gpio, LOW);  // Caso contrario se desactiva
  }

  // Enciende el foco si la temperatura es menor a 25.5°C o si la humedad disminuye de
55%

  if (temperatura22 < 25.50 || humedad22 < 55) {
```

```
    digitalWrite(relay2_gpio, HIGH);
  } else {
    digitalWrite(relay2_gpio, LOW);   // Desactivación
  }

  delay(5000);
}
```