ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

“DISEÑO Y PROTOTIPO DE UN SISTEMA BASADO EN EL INTERNET DE LAS COSAS (IOT) PARA EL CONTROL Y MONITOREO EN TIEMPO REAL DE LA TEMPERATURA Y LA HUMEDAD EN LOS EQUIPOS DE REFRIGERACIÓN DEL ÁREA DE VACUNACIÓN Y FARMACIA DE UN HOSPITAL.”

EXAMEN DE GRADO

Previo a la obtención del título de:

MAGISTER EN TELECOMUNICACIONES

Presentado por:

OLIVIER MARLON AGUILAR MOSQUERA

GUAYAQUIL – ECUADOR

AÑO 2020

AGRADECIMIENTO

Agradezco a Dios por la vida y la familia que me ha regalado, por darme salud y las fuerzas necesarias para poder realizar este proyecto. A mis padres, por darme su apoyo incondicional en todo momento. A mí familia en general por darme su confianza y apoyo en cada una de las etapas de mi vida profesional.

A todos los Docentes de la MET VI, por brindarnos sus conocimientos en cada uno de los módulos impartidos y orientarnos en el ámbito profesional.

Finalmente agradezco a mis compañeros que me permitieron compartir con ellos momentos muy gratos, brindándome su apoyo y sus conocimientos que me permitieron superar cada uno de los módulos exitosamente. Gracias a todos.

Olivier Marlon Aguilar Mosquera

DEDICATORIA

Este proyecto va dedicado a Dios Todopoderoso por bendecirme y darme la oportunidad de ver realizado unos de mis objetivos, a mis padres por creer en mí y haberme dado la educación y motivarme en seguir adelante, a mis hermanos por su gran cariño y múltiples consejos y a toda mi familia que me dio la oportunidad de avanzar y luchar por una de mis metas.

Olivier Marlon Aguilar Mosquera

TRIBUNAL DE EVALUACIÓN

MSc. Jorge Brito

PROFESOR EVALUADOR

Ph.D. María Antonieta Álvarez

PROFESOR EVALUADOR

DECLARACIÓN EXPRESA

“La responsabilidad y la autoría del contenido de este Trabajo de Titulación, me corresponde exclusivamente; y doy mi consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual.”

Olivier Marlon Aguilar Mosquera

RESUMEN

Unos de los principales problemas que provocan el deterioro de los medicamentos es la variación de la temperatura y la humedad en las áreas donde se almacenan. En especial las vacunas que deben permanecer en un rango de temperatura adecuada de (2° a 8°C) para evitar que estas pierdan su efectividad. La dificultad de tener un monitoreo constante sobre la temperatura en que permanecen estos medicamentos puede provocar un daño irreversible en los mismos, generando un comportamiento adverso del medicamento. Comprometiendo el estado de salud del paciente como son eventos supuestamente atribuidos a la vacunación e inmunización (ESAVI), es cualquier cuadro clínico patológico asociado temporalmente a la vacunación o inmunización no existiendo necesariamente relación causal, motivando el inicio de la investigación epidemiológica; Es todo cuadro clínico que implica uno o más de los siguientes criterios: a). Hospitalización. b). Riesgo de muerte. c). Discapacidad. d). Fallecimiento. Puede clasificarse como: Evento coincidente, evento relacionado con error programático u operacional, evento relacionado con los componentes propios de la vacuna y evento no concluyente.

En base a esta necesidad en el siguiente proyecto se plantea el diseño y prototipo de un sistema basado en el internet de las cosas (iot) para el control y monitoreo en tiempo real de la temperatura y la humedad en los equipos de refrigeración del área de vacunación y farmacia de un hospital, el cual permita prevenir los posibles fallos en el voltaje eléctrico en el sistema de refrigeración donde se conservan los medicamentos y las diferentes vacunas en las áreas de farmacia en un Hospital. Para el desarrollo y el correcto funcionamiento de este proyecto es necesario utilizar un sensor DTH11, que permita obtener los datos de la temperatura y humedad en tiempo real de los equipos de refrigeración. El seguimiento de estos datos nos permitió generar una alarma mediante notificaciones SMS empleando la red GSM, para lo cual se usará el módulo SIM900. Estos datos serán almacenados en la base de datos Influxdb, Para el análisis y representación de los datos se utilizará Grafana, que permitirá observar el comportamiento de la temperatura y la humedad en tiempo real mediante gráficos.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

1. INTRODUCCIÓN

En la actualidad uno de los principales objetivos de las entidades de salud es prevenir y reducir los riesgos, errores y daños que sufren los pacientes durante la prestación de la asistencia sanitaria. Una piedra angular de la disciplina es la mejora continua basada en el aprendizaje a partir de los errores y eventos adversos, y de esta forma mantener la integridad del paciente y brindar una atención segura. Además coordinar con los diferentes procesos sobre el uso de aplicaciones tecnológicas que permitan solucionar problema en el área de salud, estos son unos de los grandes retos que enfrentan la mayoría de los hospitales [1].

En Ecuador, el Ministerio de Salud Pública reconoce que la seguridad del paciente es fundamental para prestar servicios sanitarios esenciales de calidad. De hecho, existe un claro consenso de que los servicios de salud de calidad en todo el mundo deben ser eficaces y seguros y estar centrados en las personas. Además, para que los beneficios de una atención sanitaria de calidad sean efectivos, los servicios de salud deben prestarse de manera oportuna, equitativa, integrada y eficiente. Tomando en consideración este criterio se puede indicar que la calidad de atención en su salud son aspectos fundamentales para el cambio de la cultura organizacional. En este entorno normativo, se definen las características y condiciones óptimas de la atención que se debe brindar a cada uno de los pacientes que acudan a un centro de salud. Prevenir posibles eventos adversos, detectar riesgos, actuar a tiempo, corregir los errores y aprender de ellos, es el nuevo desafío de la salud publica ecuatoriana.

Según la normativa vigente del Ministerio de Salud Pública, en el manual sobre el correcto almacenamiento de los medicamentos y el control adecuado de la temperatura y humedad, refiere lo siguiente : “El personal de bodega debe

realizar la lectura de los termohigrómetros y registrar diariamente la temperatura y humedad relativa, en la mañana entre las 08h00 y 09h00, y en la tarde entre las 14h00 y 15h00, incluidas las observaciones que amerite completando el formato de Registro de temperatura y humedad relativa ambiental y de cadena de frío."[2]

La persona que realiza la lectura debe verificar que la temperatura y humedad relativa se encuentren dentro de los límites adecuados, a no ser que el proveedor indique una temperatura específica. Se consideran como límites adecuados los siguientes:
Temperatura ambiente: Entre 15 a 25 °C, nunca más de 30 ºC.
Temperatura de refrigeración: 2 a 8 °C
Humedad relativa: Entre 50 % y 70 %

Generalmente el registro de estos valores se lo realiza de forma manual por la persona encargada de farmacia o bodega, el cual está ubicado cerca de los equipos de refrigeración, esto permite obtener datos que son subjetivos y falta de veracidad y confianza. Por lo general no se realiza un registro permanente de los datos por lo que no se puede conocer de manera correcta la variación de estos datos en el tiempo.[2]

El propósito principal de este proyecto es desarrollar un sistema basado en el Internet de las Cosas (IoT), que permita el monitoreo y control en tiempo real de los valores de la temperatura y humedad de los equipos de refrigeración de medicamentos de un Hospital. Con el fin notificar mediante SMS de manera oportuna cuando estos equipos no estén trabajando de manera adecuada y los rangos de temperatura y humedad estén fuera de los correctos, de esta manera poder prevenir que los medicamentos (vacunas) se deterioren o pierdan su efectividad y causen algún daño en la salud.

1.1. Descripción del problema

Los componentes activos de los medicamentos biológicos, son típicamente proteínas y/o polipéptidos, en los cuales la conservación de la conformación molecular y de la actividad biológica depende de fuerzas covalentes y no covalentes; en consecuencia, son sensibles a los factores ambientales (temperatura, humedad y luz), a la oxidación, al contenido iónico y a la ruptura por cizalladura, por lo tanto, se requieren condiciones bien definidas de almacenamiento con el fin de asegurar la conservación de la actividad biológica y evitar su degradación.

La variación de voltaje de la energía eléctrica son una de las variables que influye en la temperatura y la humedad en las áreas donde se almacena medicamentos (vacunas), en este sentido hay que tomar en cuenta la capacidad de un ingrediente farmacéutico activo o producto farmacéutico terminado, de mantener a través del tiempo sus propiedades originales dentro de las especificaciones establecidas, en relación a su calidad, seguridad y eficacia, estos son unos de los principales inconvenientes para su deterioro, en especial las vacunas que deben permanecer en un rango de temperatura adecuada para evitar que estas no pierdan su efectividad, ya que pueden sufrir congelamiento si están a temperaturas muy bajas o perder sus propiedades si están a temperaturas muy altas. La dificultad de tener un monitoreo constante sobre la temperatura en la que permanecen estos medicamentos puede provocar un daño irreversible en los mismos generando un comportamiento adverso del medicamento.

Otro de los problemas que se presentan son las exposiciones de corto tiempo del medicamento biológico a temperaturas no recomendadas de almacenamiento, es decir demasiado altas o bajas, teniendo en cuenta que pueden ser inevitables en algún momento, en particular durante la manipulación y transporte o el uso en zonas climáticas con altas temperaturas. Debido a que las vacunas son de clase heterogénea de medicamentos que contienen sustancias inmunogénicas

capaces de inducir en el huésped una inmunidad específica, activa y de protección contra las enfermedades infecciosas.

Actualmente según la normativa vigente del Ministerio de Salud Pública sobre el correcto almacenamiento de los medicamentos y el control adecuado de la temperatura y humedad refiere lo siguiente: “Si la temperatura y humedad relativa dentro de la bodega están fuera de los límites establecidos (2° a 8°C), se tomará las siguientes acciones: Si es superior, se debe aumentar la ventilación regulando los ventiladores y/o equipos de aire acondicionado, los extractores de aire o abriendo las ventanas. Si es inferior, se debe aumenta la temperatura regulando los equipos de aire acondicionado. Si se detectará el mal funcionamiento de los equipos de ventilación, se debe comunicar inmediatamente al responsable de bodega, para que tome acciones inmediatas.[2]

Todo este proceso de verificación es realizado manualmente provocando lentitud y pérdida de tiempo a la hora de tomar una decisión. Además, podemos indicar que en ocasiones se presentan cortes de energía eléctrica y los equipos de refrigeración pueden permanecer máximo 4 horas sin suministro de energía, si se excede este tiempo y no se ha tomado una acción para la protección de los medicamentos, estos pueden deteriorarse.

1.2. Justificación

El propósito del presente proyecto es de poder tener estabilidad y asegurar que los medicamentos biológicos sean de calidad y que cuenten con perfiles de seguridad y eficacia definidos, que se mantengan hasta el final de la vida útil o durante el periodo de almacenamiento en las condiciones recomendadas de temperatura y humedad; se pretende garantizar que los medicamentos biológicos mantengan la calidad, seguridad y eficacia durante el tiempo de vida útil asignado, en las condiciones de almacenamiento establecidas, para lo cual se realizará las acciones de inspección, vigilancia y control correspondientes.

Por otro lado el proyecto se plantea el diseño y desarrollo de un sistema basado en IoT [3], el cual permita prevenir los posibles fallos en el tratamiento y conservación de los medicamentos en las áreas de farmacia y vacunación en un Hospital. Esto se logrará mediante el monitoreo constante y permanente de los valores de temperatura y humedad de los equipos de refrigeración que contienen los medicamentos. Según la normativa vigente del Ministerio de Salud Pública estos deben estar entre 2 a 8 °C. De esta manera se podrá asegurar que se mantengan en la temperatura adecuada, protegiendo sus propiedades químicas que recomienda el fabricante para que no pierdan su efectividad.

Para el desarrollo y el correcto funcionamiento de este proyecto es necesario utilizar un sensor DTH11 que permita obtener los datos de la temperatura y humedad, ya que cumple con las condiciones de funcionamiento del equipo a monitorear, es decir que su rango de trabajo está entre (2 a 8 °C) y humedad entre (50% a 70%). Este sensor estará conectado al módulo ESP32 dev, el módulo ESP32 se conectará de manera Inalámbrica a la red local y permitirá transmitir los datos a un servidor local alojado en la Raspberry Pi utilizando el protocolo de comunicación MQTT [4].

El seguimiento de estos datos nos permitirá crear un sistema de alarma, el cual facilitará el monitoreo para conocer cuando la temperatura este por encima o por debajo de los rangos óptimos (2 a 8 ° C), para ello se va a utilizar un Módulo Shield Quad-band Sim900, que facilita el envío de notificaciones mediante SMS usando la red GSM, dichas notificaciones serán enviadas a la persona encargada del área de farmacia.

Así mismo la información obtenida por el sensor será almacenada en una base de datos (influxdb), así mismo se empleará Node-Red para el tratamiento de los datos, y para en análisis y representación de los datos se utilizará Grafana, que permitirá observar el comportamiento de la temperatura y la humedad de manera gráfica y en tiempo real.

Por último, se realizó una serie de pruebas diseñadas para obtener información sobre la estabilidad de un medicamento biológico con el fin de definir su vida útil y su período de utilización bajo condiciones de envase y almacenamiento especificadas, para determinar el impacto de los factores ambientales extremos tales como luz y temperatura. Estas pruebas se realizan generalmente como parte de un programa de estabilidad. Estas se utilizan para establecer condiciones de protección de envase y contenedores, y respaldan las leyendas de etiquetado.

1.3. Objetivos:

1.3.1. Objetivo general

Desarrollar un sistema basado en el Internet de las Cosas (IoT), que permita el monitoreo y control en tiempo real de los valores de la temperatura de los equipos de refrigeración de medicamentos de un Hospital, con el fin notificar mediante SMS de manera oportuna y prevenir que estos se deterioren o pierdan su efectividad y causen algún daño en la salud.

1.3.2. Objetivos Específicos

- Analizar el uso de IoT en el área de salud, obtener datos sobre la temperatura adecuada que deben permanecer los medicamentos para evitar su deterioro.
- Diseñar e implementar un modelo que permita determinar la tecnología más conveniente para la implementación de un sistema basado en (IoT) para el control de la temperatura, usando sensores DTH11, Módulo Sim900, Módulo esp32 y Raspberryp, para el control de la temperatura en estos ambientes.
- Evaluar los resultados de las pruebas realizadas luego de terminar el diseño del prototipo para mejorar la eficiencia en el control de los medicamentos que se encuentran en refrigeración.

1.4. Marco Teórico

1.4.1. Centro de Recursos de Cadena de Frio OPS

Según la OPS (Organización Panamericana de Salud), para un adecuado almacenamiento de las vacunas establece que “Dependiendo del tipo de la vacuna hay dos rangos de temperaturas para el almacenamiento de estas: Vacunas que son sensibles al congelamiento deben almacenarse a temperaturas entre 2°C a 8°C. Las vacunas producidas con sepas víricas y/o liofilizadas pueden almacenarse a temperaturas entre -15°C y -25°C”, [5].

1.4.2. Módulo de adquisición de humedad y temperatura DTH11

El DHT11 es un tipo de sensor que permite realizar una medición de la temperatura y humedad de forma simultánea.

Estos tipos de sensores disponen de un procesador interno que realiza el proceso de medición, proporcionando la obtención de los datos mediante una señal digital, por lo que resulta muy sencillo obtener la medición desde un microprocesador como Arduino. DHT11 se caracteriza principalmente por tener la señal digital graduada, la cual permite tener una alta fiabilidad en el transcurso del tiempo. Puede medir la humedad que va en el rango desde 20% hasta 90% y la temperatura va en el rango de 0ºC a 50ºC, [6].

En la Tabla 1 se detallan las características más relevantes del sensor DTH11.

Alimentación	3Vdc ≤ Vcc ≤ 5Vdc
Rango de medición de temperatura	0 a 50 °C
Precisión de medición de temperatura	±2.0 °C
Resolución Temperatura	0.1°C
Rango de medición de humedad	20% a 90% RH
Precisión de medición de humedad	4% RH
Resolución Humedad	1% RH
Tiempo de censado	1 seg

Tabla 1 Especificaciones técnicas del sensor DTH11

1.4.3. Modulo ESP32 DEVKIT V1 30 Pines

La placa de desarrollo Esp32 DevKit v1 incorpora un CPU que corre a dos núcleos de hasta 240Mhz los que pueden ser controlados de manera independientemente. Además, dispone de comunicación vía Wifi y bluetooth. Lo cual facilita la creación de proyectos rápidamente con la plataforma ESP32. El ESP32 fue desarrollado por la empresa ESPRESSIF Systems una compañía de origen chino [7].

En la Tabla 2 se muestra las características y especificaciones técnicas del módulo ESP32.

Voltaje de Alimentación (USB):	5V DC
Voltaje de Entradas/Salidas:	3.3V DC
Consumo de energía	5µA en modo de suspensión
CPU principal:	Tensilica Xtensa 32-bit LX6
Desempeño:	Hasta 600 DMIPS
Frecuencia de Reloj:	hasta 240Mhz
Procesador secundario:	Permite hacer operaciones básicas en modo de ultra bajo consumo
Wifi:	802.11 b/g/n/e/i (802.11n @ 2.4 GHz hasta 150 Mbit/s)
Bluetooth:	4.2 BR/EDR BLE Modo de control dual
Memoria:	448 KByte ROM, 520 KByte SRAM, 6 KByte SRAM en RTC y QSPI admite múltiples chips flash /SRAM

Tabla 2 Especificaciones del Módulo ESP32

1.4.4. Módulo Shield Quad-band Sim900

El módulo SIM900 es un Shield compacto y confiable, está basado en el chip SIM900, compatible con Arduino Uno y Mega. Trabaja en frecuencias gsm/gprs de 850/900/1800/1900 MHz, para realizar llamadas de voz, envió de sms y fax. Su configuración se realiza mediante el protocolo UART, empleando comandos AT. Por defecto la velocidad UART está establecida en 19200 baudios. Para esta función posee un jumper, para poder seleccionar los pines digitales mediante los cuales queremos realizar la comunicación (D0-D3), hay un switch en la placa que nos permite seleccionar entre una conexión UART o un puerto de debug, el SIM900 se puede conectar directamente a un pc vía un chip FTDI232. [8].

Se puede observar en la Tabla 3 las características técnicas del Módulo SIM900

Conexión	Puerto serial
Quad-Band	850 MHz, 900 MHz, 1800 MHz y 1900 MHz
GPRS	Multi-slot clase 10/8
GPRS	Mobile station clase B
Compatible	GSM fase 2/2+
Clase 4	2 W (AT) 850 MHz a 900 MHz
Clase 1	1 W (AT) 1800 MHz a 1900 MHz
TCP/UP	Embebido
Soporta	Soporta RTC
Consumo	1.5 mA. A 2.0 mA

Tabla 3 Especificaciones Técnicas del Módulo SIM900

1.4.5. Tarjeta Arduino UNO

Arduino UNO es una placa basada en el microcontrolador ATmega328P. Tiene 14 pines de entrada/salida digital (de los cuales 6 pueden ser usando con PWM), 6 entradas analógicas, un cristal de 16Mhz, conexión USB, conector jack de alimentación, terminales para conexión ICSP y un botón de reinicio. Tiene toda la electrónica necesaria para que el microcontrolador opere, simplemente hay que conectarlo a la energía por el puerto USB o con un transformador AC-DC, [9].

En la Tabla 4 se describe las características técnicas de la tarjeta Arduino UNO.

Microcontrolador:	ATmega328
Voltaje de operación	5 V
Voltaje de alimentación (recomendado)	7-12 V
Pines Digitales E/S	14 (6 PWM)
Pines Análogos (entrada)	6
Memoria flash	32 KB
Memoria SRAM	2 KB
Memoria EEPROM	1 KB
Velocidad del reloj	16 MHz

Tabla 4 Especificaciones técnicas tarjeta Arduino UNO

1.4.6. Raspberry Pi 4

El Raspberry Pi es un dispositivo cuyo software dispone de características muy similares a las de un computador, pero con la ventaja que tiene un tamaño reducido. Fue desarrollado por la fundación Raspberry Pi con la finalidad de otorgar la oportunidad de desarrollar una gran variedad de proyectos y fomentar el aprendizaje en las instituciones educativas. El sistema operativo oficial de la Raspberry Pi es Raspbian, que es una versión adaptada del Debian/GNU Linux. [10].

A continuación, se detallan las características y especificaciones de la Rasberry Pi 4 ver Tabla 5.

Procesador	ARM Cortex-A72
Frecuencia De Reloj	1,5 GHz
Gpu	Video Core VI (con soporte para OpenGL ES 3.x)
Memoria	1 GB / 2 GB / 4 GB LPDDR4 SDRAM
Conectividad	Bluetooth 5.0, Wi-Fi 802.11ac, Gigabit Ethernet
Puertos	GPIO 40 pines 2 x micro HDMI 2 x USB 2.0 2 x USB 3.0 CSI (cámara Raspberry Pi) DSI (pantalla tácil) Micro SD Conector de audio jack USB-C (alimentación)
Video Y Sonido	Hasta 600 DMIPS
Multimedia	hasta 240Mhz
Soporte De Tarjeta Sd	Permite hacer operaciones básicas en modo de ultra bajo consumo
Alimentación	802.11 b/g/n/e/i (802.11n @ 2.4 GHz hasta 150 Mbit/s)
Temperatura De Funcionamiento	4.2 BR/EDR BLE Modo de control dual

Tabla 5 Especificaciones Técnicas Raspberry Pi 4

1.4.7. Protocolo MQTT

MQTT (Message Queue Telemetry Transport) es un protocolo de transporte de mensajes Cliente/Servidor basado en publicaciones y subscripciones de “tópicos”. Cada vez que un mensaje es publicado será recibido por el resto de los dispositivos adheridos a un tópico del protocolo. Al igual que el Protocolo de transferencia de hipertexto (HTTP), MQTT se basa en el Protocolo de control de transmisión (TCP) e IP como sus capas subyacentes. La fiabilidad de los mensajes en MQTT está a cargo de tres niveles de calidad de servicio (QoS). El nivel de QoS 0 significa que un mensaje se entrega como máximo una vez y no se requiere confirmación de recepción. El nivel 1 de QoS significa que cada

mensaje se entrega al menos una vez y se requiere la confirmación de la recepción del mensaje. En el nivel 2 de QoS, se utiliza un mecanismo de protocolo de enlace de cuatro vías para la entrega de un mensaje exactamente una vez.[11].

1.4.8. Grafana.

Grafana es una herramienta desarrollada en software libre, específicamente con licencia Apache 2.0, ideada por Torkel Ödegaard (encargado actualmente de su desarrollo y mantenimiento) y creada en enero de 2014. Grafana es un software de análisis y visualización de código abierto. Este software permite consultar, visualizar, alertar y explorar sus métricas sin importar dónde estén almacenadas. En un lenguaje sencillo, proporciona herramientas para convertir los datos de su base de datos en gráficos y visualizaciones muy sencillas y fácil de interpretar, [12].

1.4.9. Alcance del Proyecto

Para el desarrollo de esta solución se usará una Raspberry Pi, ya que por sus características cumple con las condiciones necesarias para el funcionamiento de este proyecto, su tamaño pequeño y compatibilidad con el sistema operativo Linux lo hacen un equipo muy potente. Este dispositivo usara el protocolo MQTT para la publicación de los datos.

La alimentación de la Rasberry es de 5 Voltios y puede ser conectada a un adaptador de voltaje USB el mismo que estará conectado a una toma de corriente de 110 voltios, el consumo de este dispositivo es mínimo. Para la alimentación del Módulo SIM900 y de la tarjeta Arduino UNO se usarán adaptadores de corrientes de 5 Voltios 1 Amperio, y para la alimentación del Módulo ESP32 se usará un adaptador micro USB de 5 Voltios, todos estos adaptadores estarán conectados a una toma de corriente de 110 Voltios.

Este proyecto está diseñado para trabajar en un área de banco de vacunas o farmacia, donde se usa un refrigerador HYC-260. El prototipo de control de temperatura tendrá dos modos de operación: el primero será cuando no se disponga de internet en el área, en este caso lo datos de temperatura y humedad seguirán siendo monitoreados y la alerta solo será enviada por mensajes de texto, no se podrá tener una representación gráfica en tiempo real de los valores obtenidos. En el segundo caso es el cual se disponga de Internet, se podrá representar de manera gráfica los datos en forma remota y en tiempo real, además se notificará mediante mensaje de texto cualquier alerta en el cambio de la temperatura o humedad.

CAPÍTULO 2

2. DISEÑO DEL PROTOTIPO

Para el diseño del prototipo, se va a basar en el siguiente esquema, donde se describe la ubicación del sensor DTH11, el mismo que se encontrará ubicado en la parte interna del equipo de refrigeración, este estará conectado a un módulo ESP32, este módulo enviará por medio de red Wifi los datos al servidor Mosquitto que se encontrará instalado en nuestra Raspberry pi, los datos obtenidos serán almacenados en una base de datos (Influxdb) y representados en una PC mediante Grafana. En el módulo ESP32 estará también conectado a una tarjeta Arduino UNO que permitirá la comunicación con el módulo SIM900 y el control de los rangos de temperatura, para él envió del mensaje de alerta a un teléfono móvil del usuario o responsable del área.

En la siguiente Figura 2.1 se muestra el esquema del prototipo del proyecto.

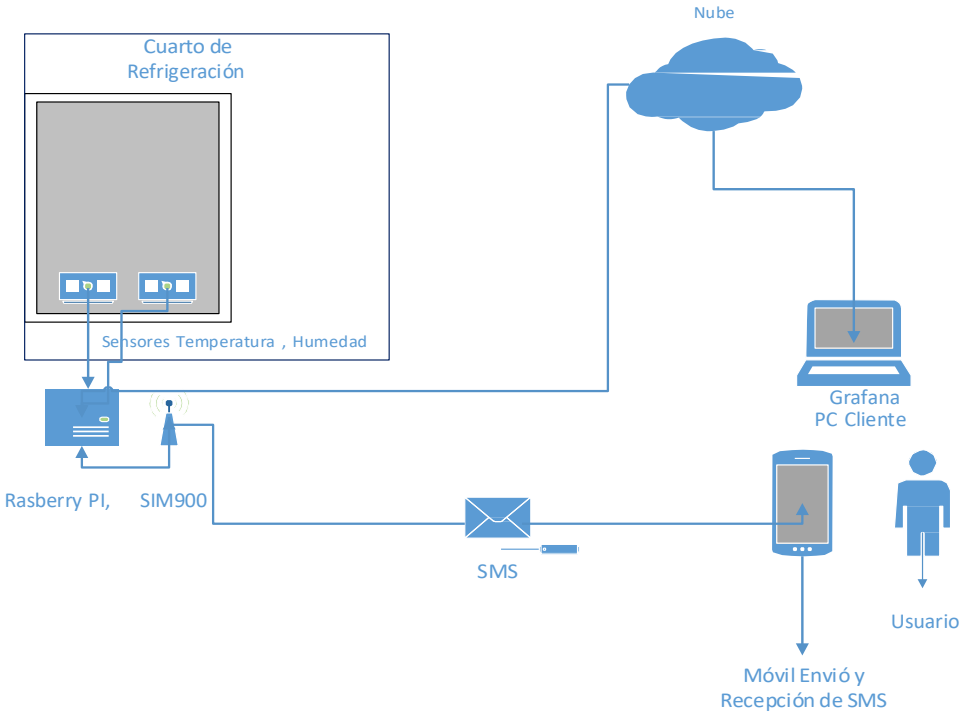

Figura 2.1 Esquema del Funcionamiento del Sistema

2.1. Diagrama de Bloques.

En la Figura 2.2 se visualiza la estructura que se seguirá en la elaboración del prototipo.

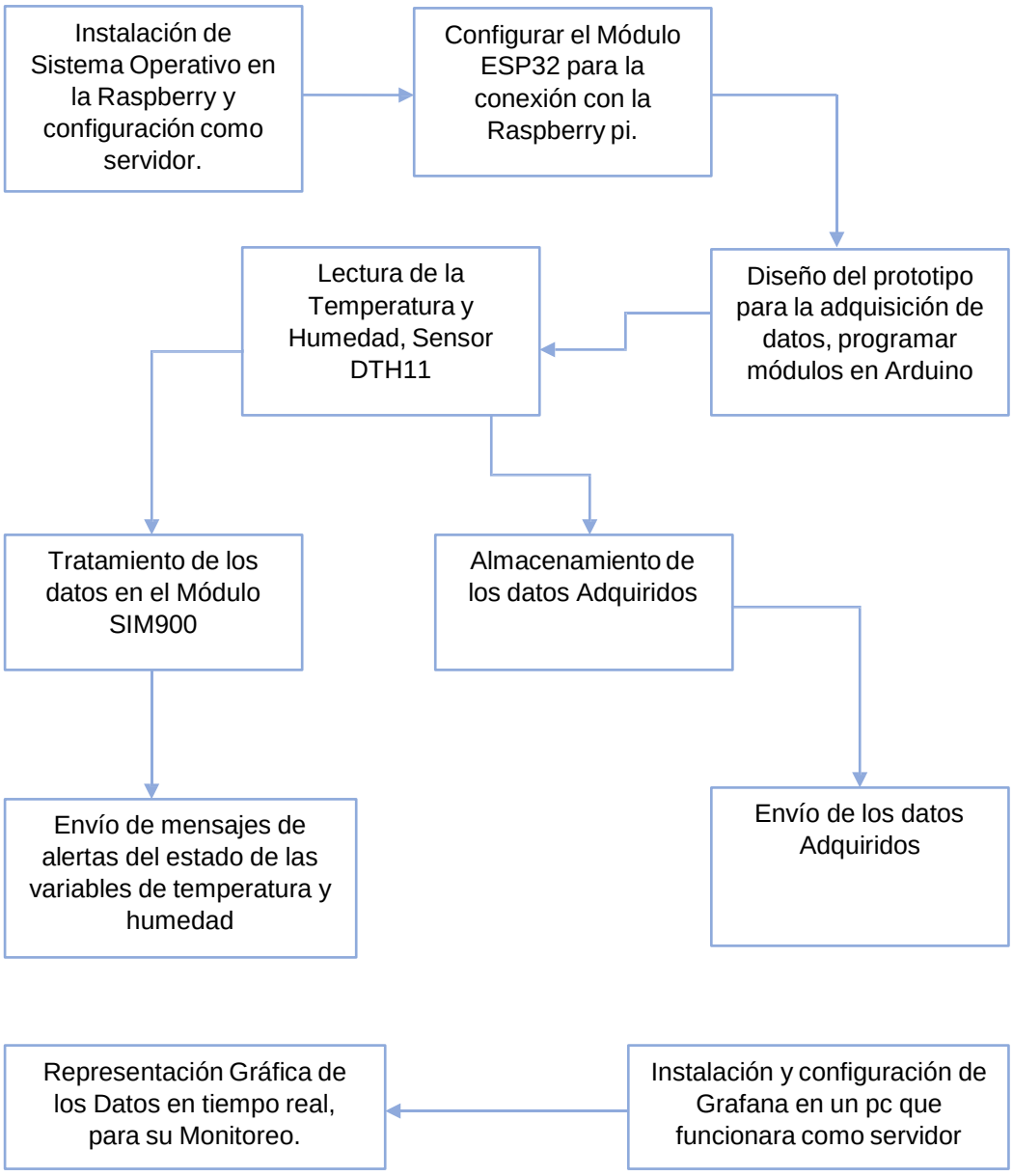

Figura 2.2 Diagramas de bloques principal del Sistema.

2.2. Especificación del Proceso de Diseño del Prototipo

Para el diseño del prototipo se usó un sensor DTH11 el mismo que tiene tres pines para su conexión, el pin izquierdo es el VCC (3.3 o 5 Voltios), el segundo pin es el que se usa para la lectura de los datos, el mismo que estará conectado al pin GPIO N° 5 del módulo ESP32, el tercer pin es el GND y está conectado al pin 2 del Módulo ESP32. Se utilizará un Módulo ESP32, el cual va a permitir la publicación de los datos mediante el protocolo de comunicación MQTT, para esto se crearán varios tópicos, uno para los datos de la temperatura y uno para los datos de la humedad, estos datos serán enviado mediante la red wifi a nuestra Raspberry pi, en el cual se estará ejecutando nuestro servidor Mosquitto.

En el módulo ESP32 dispone de 30 pines, 24 son pines GPIO digitales, de los cuales usaremos el pin N° 12 y 13 como salida, los mismos que estarán conectados a una tarjeta Arduino Uno al pin 13 y 6 respectivamente, dichos pines serán configurados para poder recibir los datos de temperatura y humedad. Con la obtención de estos datos mediante programación en la tarjeta Arduino UNO se podrá tener un control en base al rango de temperatura permitida (2° a 8°) en los medicamentos (vacunas), si el rango excede los valores permitidos usaremos un módulo SIM900 para realizar las notificaciones de alerta. Este módulo SIM900 va a estar conectado al ping 7,8,9 del Arduino UNO para la trasmisión y recepción de los datos. El Módulo SIM900 será el encargado de recibir esta información por parte del Arduino UNO, este módulo tendrá una tarjeta SIM la que se conectara a la red celular GSM, para poder recibir y enviar mensajes SMS, ya que por medio de este tipo de mensajes se va a generar una alerta, esto será controlado mediante comandos AT.

2.3. Hardware utilizado.

Se ha mencionado en el apartado anterior el hardware y como estará conectado, en este apartado se explica un poco más a detalle su configuración y funcionamiento.

El sistema está formado por dispositivos electrónicos programables, los cuales son los siguientes:

- Raspberry pi en la versión 4
- Sensor DTH11
- Módulo ESP32 devkit v1
- Módulo Shield Quad-band Sim900 Gsm Gprs
- Modulo Arduino UNO
- Fuentes de voltaje regulable DC de 5v.

2.4. Entorno de programación y software utilizado

2.4.1. Configuración de Raspberry pi 4.

La Raspberry pi 4 es un ordenador de bajo coste de un tamaño de una tarjeta de crédito, para poder proceder con la configuración del dispositivo se le inserta una tarjeta SD en la cual se instala el sistema operativo, en nuestro caso vamos a usar el sistema operativo Raspian. La configuración del Raspberry pi es de forma remota mediante el puerto SSH. La Raspberry pi va a funcionar como servidor para ello le vamos a instalar mediante comandos el bróker Mosquitto, que es un servidor de mensajes que implementa el protocolo MQTT y es muy usado en IoT, ya que permite la publicación y suscripción de varios dispositivos. En la Figura 2.3 se muestra la instalación del servidor mosquitto en nuestra raspberry pi.

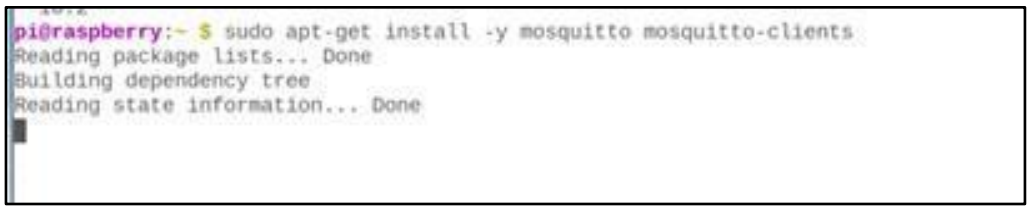

Figura 2.3 Resultado del código de instalación de Mosquitto**.**

2.4.2. Configuración ESP32 dev en Arduino.

Para la configuración del módulo ESP32 vamos a usa el software Arduino en la versión 1.8, para poder programas este módulo vamos a necesitar instalar algunas librerías y paquetes adicionales para ello nos dirigimos a Tools->Board->Boards Manager y Buscamos e Instalamos el módulo "ESP8266 Community". En la siguiente Figura 2.4 se observa la descarga de las librerías necesarias para el funcionamiento de los módulos.

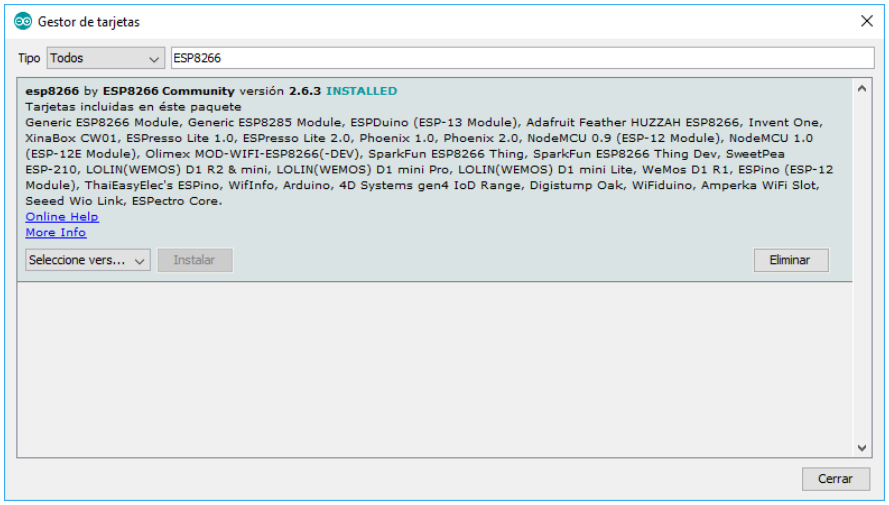

Figura 2.4 Instalación de librería ESP8266

Ahora ya tenemos listo el ambiente para la programación de nuestro módulo. La programación en este módulo nos va a permitir obtener los valores de la temperatura y humedad proporcionados por el sensor DTH11, de esta manera vamos a configurar nuestro módulo ESP32, el cual va a apuntar a la dirección IP de nuestra Raspberry pi para que los datos sean publicados en este servidor para su uso posteriormente.

2.4.3. Configuración de la tarjeta Arduino UNO

Para la configuración de nuestra Tarjeta Arduino UNO vamos a usar el software Arduino en la versión 1.8. La programación en esta tarjeta va a permitir controlar mediante comandos AT las acciones en el Módulo

SIM900, de esta forma podemos tomar decisiones en base a los datos obtenidos de la temperatura y la humedad.

2.4.4. Configuración de Node-Red, InfluxDB y Grafana

En esta parte vamos a realizar la Integración de Node Red, InfluxDB y Grafana, Para ello vamos a tener una máquina virtual con el sistema operativo UBUNTU, una vez configurada nuestra máquina virtual vamos a poner en marcha los siguientes servicios. En la siguiente Figura 2.5 se observa cómo se inician cada uno de los servicios indicados anteriormente.

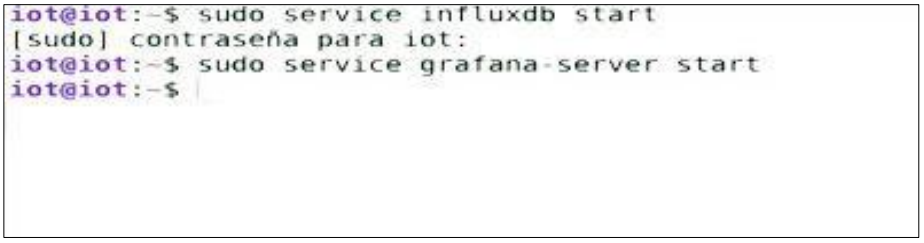

Figura 2.5 Inicio de Servicios**.**

Una vez que tenemos iniciados los servicios procedemos a acceder a nuestra herramienta de desarrollo nod-red, apuntando a la siguiente dirección 127.0.0.1:1880, en la barra de direcciones de nuestro navegador. Para poder vincular node-red con nuestra base de datos en influxdb, debemos instalar el siguiente complemento: **node-red-contrib-influxdb** como se observa en la Figura 2.6.

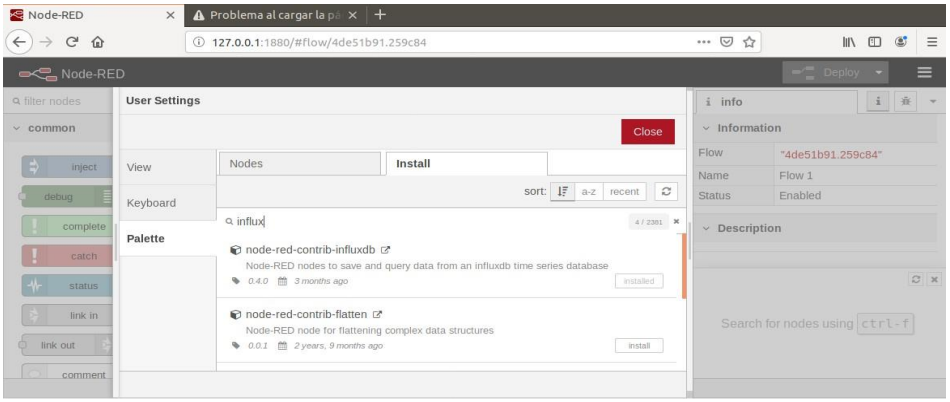

Figura 2.6 Instalación del componente influxdb

Una vez que tenemos instalado nuestro componente para la configuración de la base de datos, vamos a agregar los componentes que usaremos en este proyecto en nuestro área de trabajo del Node-Red: insertaremos dos nodos MQTT, uno para la temperatura y otro para la humedad, luego insertamos dos nodos de funciones que nos permitan convertir los valores recibidos en formato string a formato float, insertamos dos nodos influxdb para ir almacenando en nuestra base de datos los valores obtenidos de la temperatura y humedad respectivamente.

Ahora procedemos con la configuración de cada uno de los nodos, empezaremos con el nodo mqtt-broker. Para que el nodo mqtt-broker pueda comunicarse con la Raspberry pi que está funcionando como servidor, es necesario apuntar a su dirección IP para poder leer los datos de temperatura y humedad. Como se puede observar en la Figura 2.7 en nuestro caso estamos trabajando con un IP local 192.168.1.28.

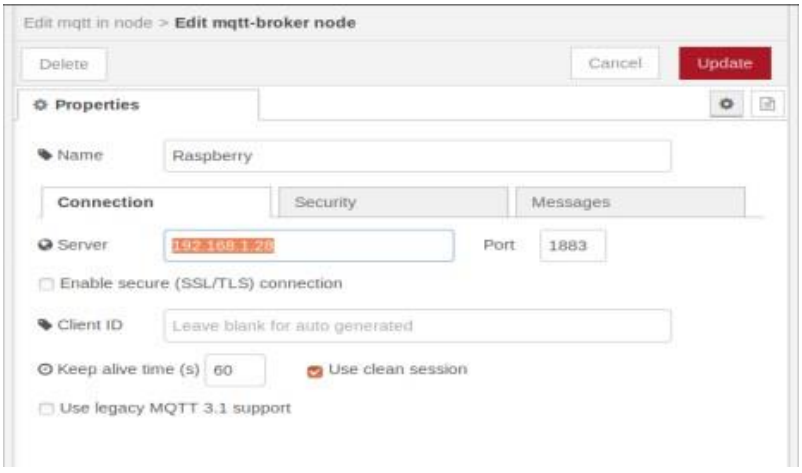

Figura 2.7 Configuración nodo mqtt-broker.

Continuando con la configuración de los nodos mqtt-broker, en nuestro caso vamos a tener dos, uno para la temperatura y uno para la humedad. Configuramos los tópicos de suscripción iot/temp/# y el tópico de suscripción iot/hum/#. En la Figura 2.8 se observa la configuración del

tópico de temperatura y en la Figura 2.9 se observa la configuración del tópico de humedad.

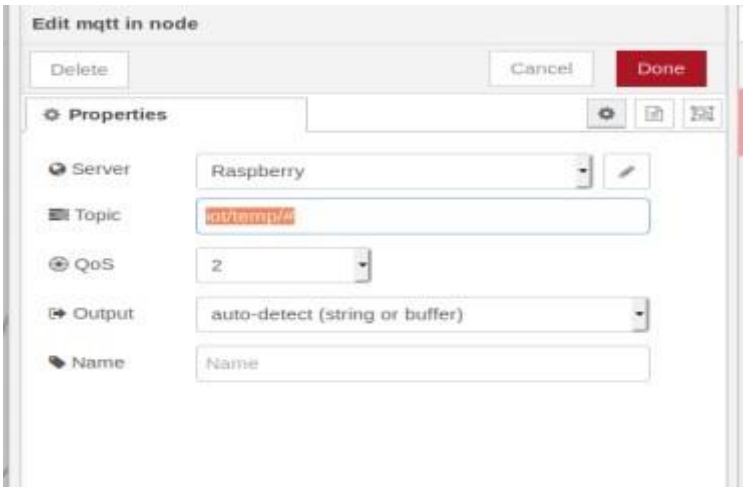

Figura 2.8 Configuración tópico temperatura.

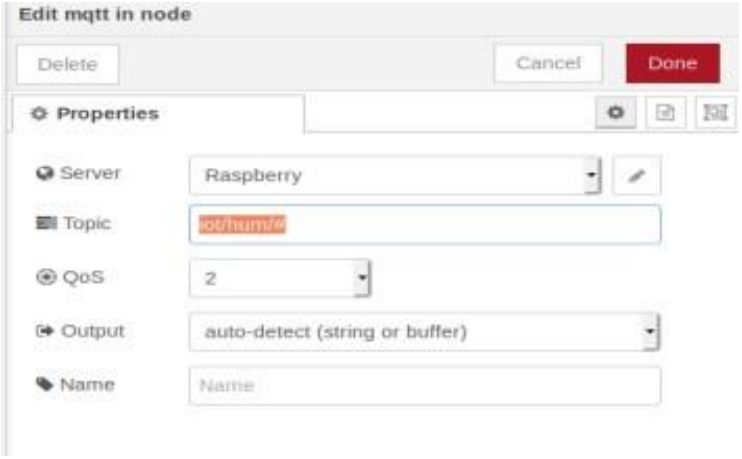

Figura 2.9 Configuración tópico humedad

Los datos que vamos a obtener de la publicación de nuestro sensor DTH11 se encuentra en un formato string, por lo que es necesario aplicar una conversión mediante código de programación para convertirlos en datos numérico, ya que es necesario para su posterior uso en la representación gráfica. Para esto insertaremos dos nodos de funciones y lo configuramos como podemos observar en la Figura 2.10.

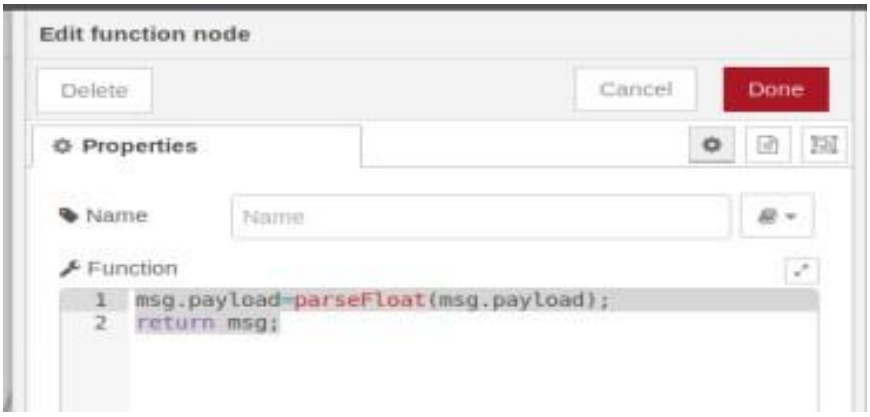

Figura 2.10 Configuración nodo funciones.

Una vez que tenemos neutros datos en formato float, vamos a realizar la configuración de los nodos influxdb para que puedan almacenar estos datos en la base de datos base_taller que hemos creado para este proyecto, le asignamos un nombre a las variables que guardaran las mediciones, en nuestro caso “temp” para temperatura y “hum” para la humedad. En la Figura 2.11 se observa la configuración de las variables.

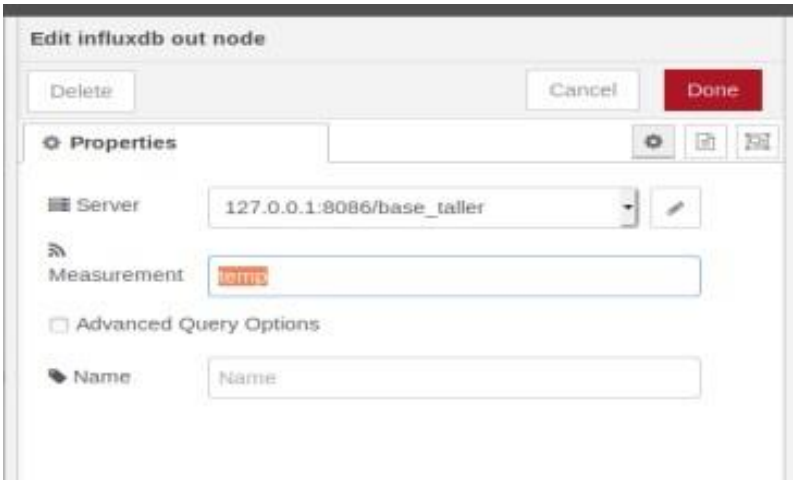

Figura 2.11 Configuración del nodo influxdb

Una vez que ya hemos realizado todas la configuraciones para obtener los datos y almacenarlos en nuestra base de datos, se observa en la Figura 2.12 el diseño de nuetro node-red.

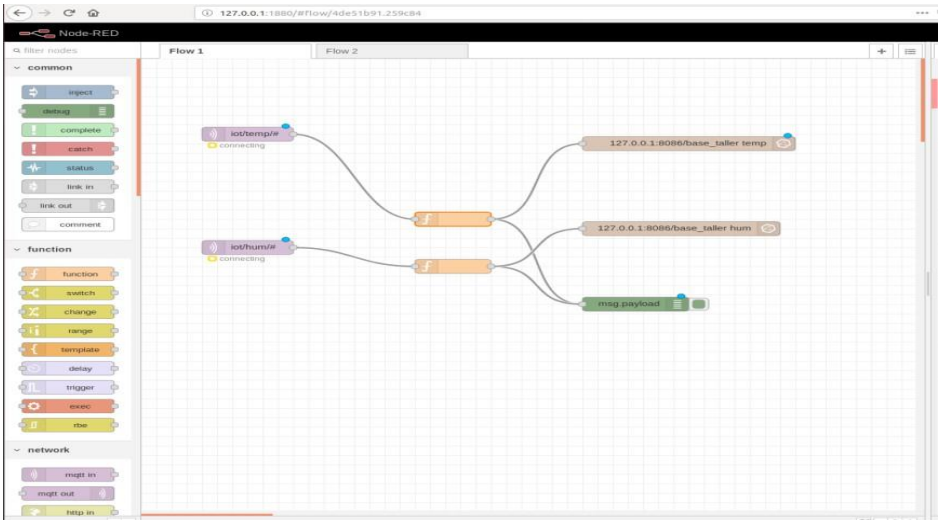

Figura 2.12 Diseño del porgrama en node-red.

Ahora procedemos a configurar Grafana, que es un sofware libre basado en Apache que permite la visualización y el formato de datos métricos. Para acceder a nuestro servidor local vamos a colocar la siguiente dirección **http://localhost:3000** con lo cual nos va a cargar la pantalla de inicio de sesión a nuestro sistema, iniciamos sesión y procedemos a configura la base de datos, para ello solo es necesario modificar la consulta SQL, por lo que tendremos lo siguiente: **SELECT value FROM temp, hum.** En la Figura 2.13 se observa el resultado de la consulta SQL de los datos en Grafana.

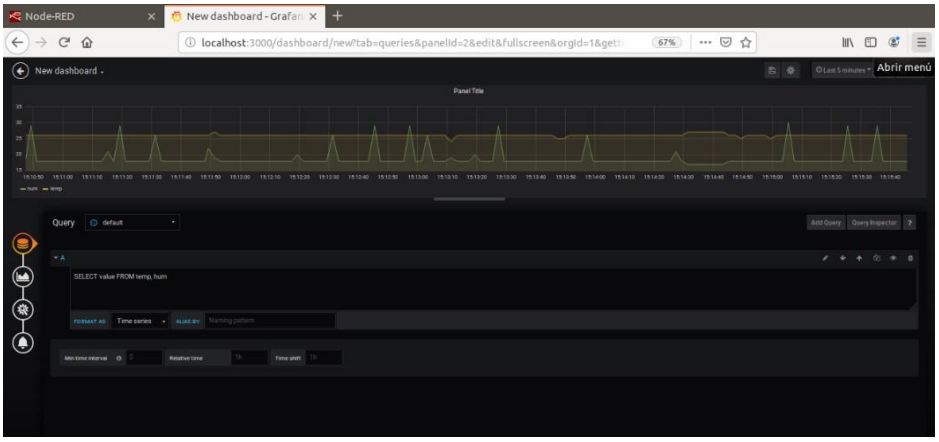

Figura 2.13 Representación datos en Grafana.

2.5. Implementación del Prototipo

Para la implementación de esta solución de Internet de las Cosas (IoT), se realizó las pruebas en un refrigerador de referencia cuyo modelo es HYC-260, el cual garantiza las temperaturas de 2° a 8° C.

Se desarrolla el montaje de los elementos electrónicos para el funcionamiento y puesta en marcha del sistema de monitoreo de temperatura y humedad vía remota. En la Figura 2.14, se aprecia la conexión de nuestra placa Arduino Uno con la tarjeta SIM900, además nuestro modulo ESP32 conectado a la tarjeta Arduino UNO, en esta fase ya se encuentra conectado el sensor DTH11 de temperatura y humedad. Con la implementación de este circuito ya podemos obtener los valores reales de las mediciones de temperatura y humedad, y poder generar alerta mediante notificación de mensajes SMS y el monitoreo en tiempo real de los valores mediante Grafana.

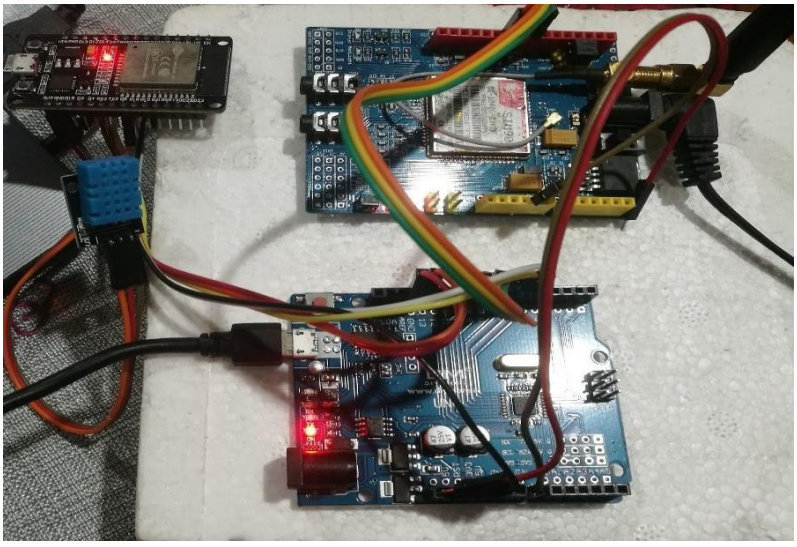

Figura 2.14 Funcionamiento del prototipo**.**

CAPÍTULO 3

3. ANÁLISIS DE RESULTADOS

3.1. Pruebas de funcionamiento del prototipo

Una vez realizado el diseño y la implementación es necesario realizar las respectivas pruebas de funcionamiento. Vamos a verificar como primer punto que los equipos encienda de manera correcta, para ello vamos a usar el siguiente tipo de alimentación eléctrica. La Tarjeta SIM900 será alimentada por una fuente externa de 5 V y 1A, nuestra Raspberry pi será alimentada por 5 V mediante el puerto C, El Módulo ESP32 dev estará conectado mediante el puerto micro USB 5V, y la tarjeta Arduino UNO será alimentado con 5 V mediante un adaptador USB. Una vez realizadas las conexiones se pudo comprobar que todos los equipos encendieron sin ningún inconveniente.

3.2. Pruebas del censado de temperatura y Humedad.

Se verificó que el sensor DTH11 realice el envío de los datos de manera correcta y sin pérdida de los mismo, estos datos son obtenidos de manera correcta en el módulo ESP32 dev. Los datos obtenidos del sensor tienen una diferencia de 1°C con respecto a los obtenidos por el termómehigrómetro usado por en personal de bodega para su medición, con lo que podemos indicar que el sensor DTH11 a pesar de obtener valores muy buenos no es el más apropiado en estos ambientes donde se manejan grados de temperaturas muy bajos.

En las pruebas realizadas se pudo observar que cada vez que existe una apertura de la puerta del equipo de refrigeración, los niveles de temperatura se elevan entre (2°a 3°) más, y una vez realizada el cerrado de la puerta demora de 5 a 10 minutos en estabilizarse a la temperatura programada. Esta información es importante porque nos permitirá estimar los tiempos en que se realicen las

notificaciones mediante SMS. En la Figura 3.1 se observa valores obtenidos en una medición con el sensor DTH11.

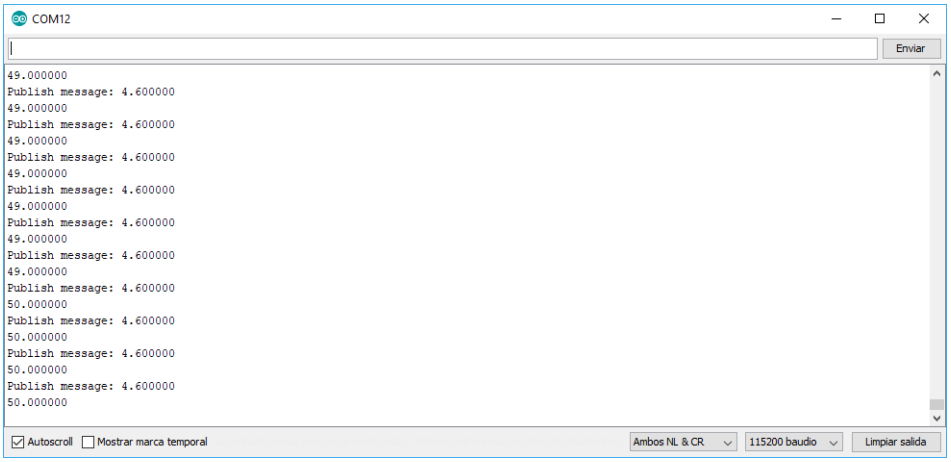

Figura 3.1 Datos de temperatura y humedad.

3.3. Adquisición de los datos vía remota.

Se inserto un chip de una operadora telefónica móvil local en el módulo SIM900 el cual se registró a la red GSM de manera adecuada. Se realizaron pruebas para la verificación del envió de las notificaciones de la tarjeta SIM900 al celular mediante SMS, en primera instancia el principal problema que se presentó es que una vez que la temperatura no estuviera en el rango adecuado se enviaban notificaciones de inmediato, es decir no había configurado un periodo de tiempo adecuado para poder realizar nuevamente las notificaciones, lo que generaba un gasto innecesario de mensajes de texto, esto se corrigió mediante la programación de la tarjeta Arduino UNO.

La configuración del servidor Grafana para la representación de los datos de manera gráfica se lo hizo en una máquina virtual VirtualBox, por lo que no se presentó inconveniente con la recepción de la información. El servidor mosquitto que se encuentra instalado en la Raspberry pi presentó algunos inconvenientes para la Publicación/Suscripción de los tópicos, por lo que se realizó una nueva instalación de este servicio en la Raspberry pi permitiendo solventar el problema.

En las pruebas de medición se obtuvieron los siguientes valores obtenidos por el sensor DTH11 y representados en Grafana, comparados con los valores obtenidos por un termómehigrómetro, ver en la Tabla 6 los valores obtenidos por el sensor tienden a ser superior por 1°C con respecto a los valores obtenidos por el termómehigrómetro.

	LECTURA DTH11 REPRESENTADO GRAFANA	LECTURA TERMÓMEHIGRÓMETRO
ITEM	**°C**	**°C**
1	9	8
2	9	8
3	8	7
4	8	7
5	8	7
6	8	6
7	7	6
8	7	5
9	6	5
10	6	5
11	6	5
12	6	5
13	5	4
14	5	4
15	5	4

Tabla 6 Datos de temperatura Adquiridos.

Una vez realizadas las pruebas y solventados los inconvenientes, se verifica que el prototipo de medición de temperatura y humedad cumple con el objetivo propuesto para el cual fue diseñado.

3.4. Evidencia gráfica de las pruebas de funcionamiento

Una vez realizadas las pruebas se realiza una representación gráfica de los datos obtenidos y el correcto funcionamiento del prototipo. En la Figura 3.2 se observa

los datos obtenidos por el sensor DTH11, la medición se realizó en horas de la mañana y están representadas en el monitor de arduino.

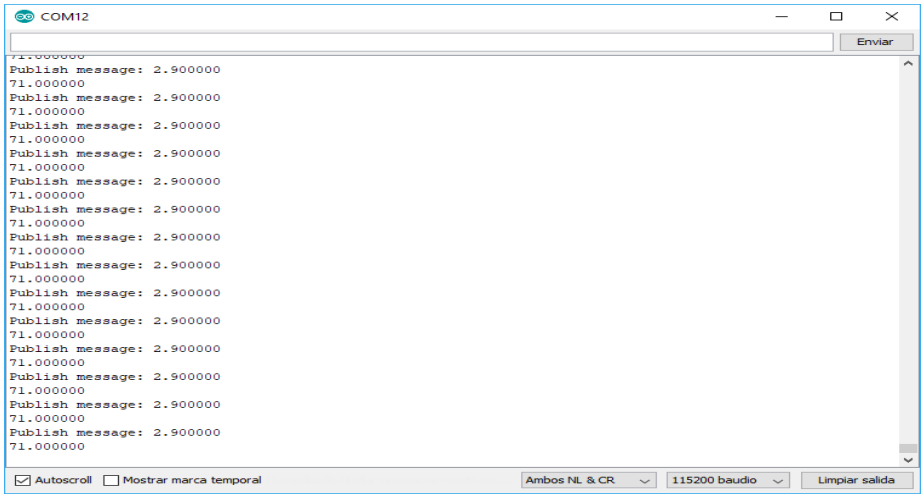

Figura 3.2 Datos Temperatura y Humedad

En la siguiente Figura 3.3 se observa el comportamiento de los datos obtenidos durante un periodo de tiempo, la gráfica evidencia que la variación en los valores es mínima y se mantiene la temperatura entre 2°C y 3°C, la humedad representa una variación mayor de 50% a 70%, este escenario se da siempre que la puerta del equipo de refrigeración permaneció cerrada.

Figura 3.3 Representación de Datos en Grafana

En la siguiente Figura 3.4 se observa el sensor DTH11 en nuestro equipo de refrigeración.

Figura 3.4 Sensor DTH11 en equipo HYC-260

Cuando los valores de la temperatura o humedad sobrepasan los límites establecidos automáticamente se genera la alerta, como podemos observar en la Figura 3.5 se muestra el resultado mediante monitor de Arduino y en la Figura 3.6 se muestra la recepción de la alerta vía SMS a un teléfono móvil.

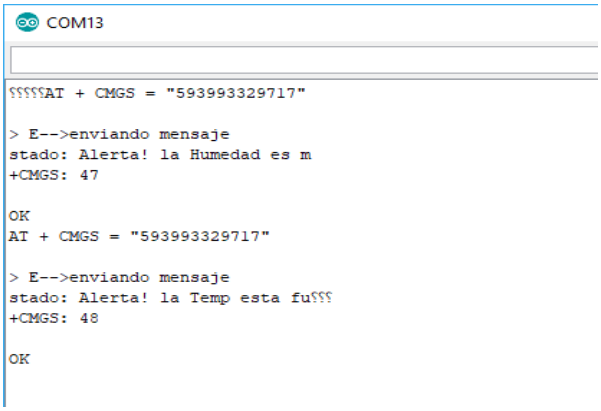

Figura 3.5 Monitoreo de resultado en Arduino

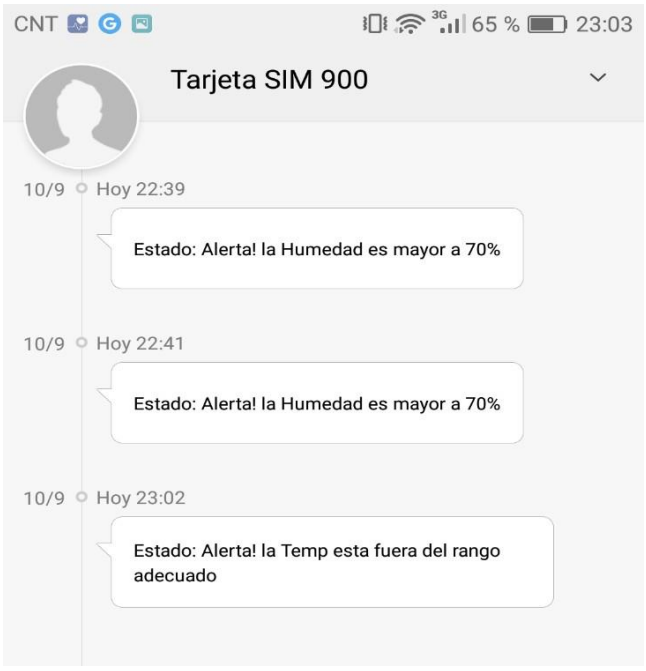

Figura 3.6 Recepción de alerta vía SMS a teléfono móvil

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

Podemos concluir que el uso de Internet ha incrementado de manera exponencial, de igual manera la necesidad de que nuestros dispositivos sean cada vez más inteligentes, para lo cual se implementa sensores que permitan obtener datos relevantes y poder tomar decisiones de manera remota. En base a estas necesidades en el siguiente proyecto se diseñó un prototipo que permita monitorear de manera remota el estado de la temperatura y humedad de en los medicamentos (vacunas), en un hospital.

Dentro de las pruebas realizada en la medición de los datos se pudo observar que es importante que la puerta del equipo de refrigeración no sea abierta de una manera frecuente ya que esto provoca un cambio de temperatura de manera muy drástica.

Con el uso de esta tecnología se pudo evidenciar las ventajas de tener equipos inteligentes, ya que se logró obtener datos confiables de la humedad y la temperatura del equipo de refrigeración. De igual manera se logró tener un control del registro de las variables para efectos de supervisiones o toma de decisiones. Sin embargo, para poder tener un mejor funcionamiento del sistema con mayor exactitud y precisión de los datos es necesario la toma de una mayor cantidad de muestras en periodos de tiempos más prolongados, así como el uso de un sensor más amigable a este tipo de ambientes. Además, con la implementación del sistema de alarma mediante SMS usando la red GSM, nos permite detectar fallos de manera oportuna, facilitando a la persona encargada tomar acciones de manera rápida permitiendo precautelar el correcto almacenamiento del medicamento y evitando que esta sufra daños con respecto a las normativas correspondientes.

RECOMENDACIONES

Dentro del manejo de los equipos de refrigeración se recomienda que solo sean manipulados cuando sea necesario, ya que la apertura y cierre de la puerta genera que la temperatura varíe de manera muy rápida perjudicando el estado de los medicamentos (vacunas).

Además, es recomendable que los dispositivos: Raspberry pi, tarjeta SIM900, Modulo ESP32 dev y Arduino Uno estén conectados a un sistema de alimentación ininterrumpida UPS, con el fin de que los equipos sigan operativos cuando de sufra algún corte de energía eléctrica, permitiendo el correcto monitoreo. En caso de no disponer de servicio de internet el sistema continúa funcionando mediante las notificaciones SMS ya que esto trabaja usando la red celular.

El uso del sensor DTH11 para la obtención de los datos de temperatura y humedad resulto muy útil, ya que los datos obtenidos siempre se mantuvieron estables, sin embargo, en comparación con los datos obtenidos por el termómehigrómetro se obtuvo una variación entre 0.5 a 1 °C, por lo que es recomendable probar con otros tipos de sensores que sean más adecuados en estos ambientes, con la finalidad de disminuir el margen de error y tener una información más precisa.

BIBLIOGRAFÍA

[1] M. de la P. S. Maritza Roa, “Lineamientos para la implementación de la Política de Seguridad del Paciente”, pp. 15–52, 2012.

[2] MSP, “Guía para la recepción y almacenamiento de medicamentos en el Ministerio de Salud Pública”, pp. 24–26, 2009.

[3] A. Hanah, R. Farook, S. J. Elias, M. R. A. Rejab, M. F. M. Fadzil, y Z. Husin, “IoT Room Control And Monitoring System Using Rasberry Pi”, en *2019 4th International Conference and Workshops on Recent Advances and Innovations in Engineering (ICRAIE)*, 2019, pp. 1–4.

[4] D. Eridani, K. T. Martono, y A. A. Hanifah, “MQTT Performance as a Message Protocol in an IoT based Chili Crops Greenhouse Prototyping”, en *2019 4th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, 2019, pp. 184–189.

[5] O. Organización Panamericana de la Salud, OPS, “Centro de recursos de la cadena de frío”.

[6] AOSONG. (s.f. de s.f. de s.f.)., “Obtenido de Temperature and humidity module DHT11 Product Manual”, pp. 1–8.

[7] E. 12) Cdmxelectronica (2020, “Tarjeta De Desarrollo ESP32 ESP-32S. [Online]”, 2020.

[8] Carrod Electrónica Online S. de R.L., “Módulo GPRS/GSM Shield con Antena Integrada SIM900”, 2014.

[9] E. 12) Cdmxelectronica (2020, “Arduino UNO R3 ATMEGA328P”, 2020.

[10] Raspberry Pi Foundation, “Raspberry Pi 4 Model B”.

[11] D. Thangavel, X. Ma, A. Valera, H. Tan, y C. K. Tan, “Performance evaluation of MQTT and CoAP via a common middleware”, en *2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2014, pp. 1–6.

[12] Grafana Labs, “Grafana-La plataforma abierta para análisis y seguimiento”, 2020.

ANEXOS

Algoritmo Aplicación principal -ESP32

```
//#include <ESP8266WiFi.h>
#include <WiFi.h>
#include <PubSubClient.h>
#include "DHT.h"
#define DHTTYPE DHT11 // DHT 11
const char* ssid = "NETLIFE-MORA";
const char* password = "CONTRASENA1234";
const char* mqtt_server = "192.168.1.28";
const int DHTPin = 5;
int ledPin = 13; // GPIO13 nuevo
int ledHum = 12; // GPIO12 nuevo

WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char    msgtmp[50];
char   msghum[50];
float valuetmp = 0;
float valuehum = 0;

DHT dht(DHTPin, DHTTYPE);
static char celsiusTemp[7];
static char fahrenheitTemp[7];
static char humidityTemp[7];

void setup() {
   Serial.begin(115200);
```

```
    delay(10);
    pinMode(ledPin, OUTPUT); // nuevo
    digitalWrite(ledPin, LOW);//nuevo
    pinMode(ledHum, OUTPUT); // nuevo
    digitalWrite(ledHum, LOW);//nuevo
    dht.begin();
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(ssid);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
       delay(500);
       Serial.print(".");
    }
    Serial.println("");
    Serial.println("WiFi connected");
    client.setServer(mqtt_server, 1883);

}
void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    String clientId = "ESP8266Client-";
    clientId += String(random(0xffff), HEX);
    if (client.connect(clientId.c_str())) {
    Serial.println("connected");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
```

```
      delay(5000);
    }
  }
}

void loop() {
        float h = dht.readHumidity();
        float t = dht.readTemperature();
        float f = dht.readTemperature(true);

        if (isnan(h) || isnan(t) || isnan(f)) {
            Serial.println("Failed to read from DHT sensor!");
            strcpy(celsiusTemp,"Failed");
            strcpy(fahrenheitTemp, "Failed");
            strcpy(humidityTemp, "Failed");
        } else{
            float hic = dht.computeHeatIndex(t, h, false);
            dtostrf(hic, 6, 2, celsiusTemp);
            float hif = dht.computeHeatIndex(f, h);
            dtostrf(hif, 6, 2, fahrenheitTemp);
            dtostrf(h, 6, 2, humidityTemp);
        }if (!client.connected()) {
            reconnect();
        }
        client.loop();
//////nuevo
        // Evalua petición temp
        int value = LOW;
        if (t>8||t<2) {
          digitalWrite(ledPin, HIGH);
```

```
  value = HIGH;
 }
 else{
  digitalWrite(ledPin, LOW);
  value = LOW;
 }
 // Evalua petición Humedad
 int valueH = LOW;
 if (h<50||h>70) {
  digitalWrite(ledHum, HIGH);
  valueH = HIGH;
 }
 else{
  digitalWrite(ledHum, LOW);
  valueH = LOW;
 }
//////
 long now = millis();
 if (now - lastMsg > 2000) {
  lastMsg = now;
  valuetmp = t;
  valuehum = h;
  snprintf (msgtmp, 75, "%lf", valuetmp);
  snprintf (msghum, 75, "%lf", valuehum);
  Serial.print("Publish message: ");
  Serial.println(msgtmp);
  Serial.println(msghum);
  client.publish("iot/temp", msgtmp);
  client.publish("iot/hum", msghum);
 } }
```

Algoritmo Aplicación SIM900

```
#include <SoftwareSerial.h>
SoftwareSerial SIM900(7, 8);//Configuración de los pines serial
char caracter=0;// Variable para guardar los caracteres
int led =5;
long tiempoT = 0;
long tiempoH = 0;
String estado="";
int leerPin = 13;  // LED connected to digital pin 13 nuevo
int val = LOW;

int leerHum = 6;
int valH = LOW;

int conta=0;
int conta1=0;

void setup() {
  SIM900.begin(19200); //Arduino se comunica con el SIM900 a una velocidad de 19200bps
  Serial.begin(19200); //Velocidad del puerto serial de Arduino
  delay(20000);//Tiempo prudencial para el escudo inicie sesión de red con tu operador
  pinMode(led, OUTPUT);
  pinMode(leerPin, INPUT);     // sets the digital pin 13 as input nuevo
  pinMode(leerHum, INPUT);
  SIM900.print("AT+CMGF=1\r");// comando AT para configurar el SIM900 en modo texto
  delay(200);
  SIM900.print("AT+CNMI=2,2,0,0,0\r");
```

```
  delay(200);

}
void loop()
{
  long now = millis();
  val = digitalRead(leerPin);//nuevo
  if (val == HIGH){//&&(conta<1)){
        if (now - tiempoT > 300000) {
           tiempoT = now;
        //digitalWrite(led, LOW);
        estado="Alerta! la Temperatura esta fuera del rango adecuado 2° a  8°C";
        envioMensaje(estado);}
    //conta=conta+1;
  }///nuevo
  else if (val == LOW){
    //digitalWrite(led, LOW);
    digitalWrite(leerPin, LOW);
    val = LOW;
    //conta=0;
  }
  valH = digitalRead(leerHum);//nuevo
 if (valH == HIGH){//&&(conta1<1)){
    if (now - tiempoH > 300000) {
       tiempoH = now;
    estado="Alerta! la Humedad es mayor a 70%";
    envioMensaje(estado);}
 }///nuevo
 else if (valH == LOW){
    digitalWrite(leerHum, LOW);
```

```
    valH = LOW;
  }


 if(SIM900.available() >0) {//Verificamos si hay datos disponibles desde el SIM900
   caracter=SIM900.read(); // Leemos los datos y los almacenamos en la variable mensaje
   Serial.print(caracter); //Imprime los datos entrantes uno a uno en el terminal serial
  }
}
void envioMensaje(String estado) {
  SIM900.println("AT + CMGS = \"593993329717\"");// el número a enviar el mensaje
  delay(200);
  SIM900.println("Estado: " + estado);// texto a enviar
  delay(200);
   SIM900.println((char)26);
  delay(200);
  SIM900.println();
}
```