



UNL

Universidad
Nacional
de Loja



Carrera de Ingeniería en
Sistemas / Computación

UNIVERSIDAD NACIONAL DE LOJA

FACULTAD DE LA ENERGÍA, LAS INDUSTRIAS Y LOS
RECURSOS NATURALES NO RENOVABLES

CARRERA DE INGENIERÍA EN SISTEMAS

**DESARROLLO DE UNA SOLUCIÓN INFORMÁTICA QUE
PERMITA ADMINISTRAR UN HOGAR INTELIGENTE
HACIENDO USO DE HARDWARE LIBRE**

**TESIS DE GRADO PREVIA A LA OBTENCIÓN
DEL TÍTULO DE INGENIERO EN SISTEMAS**

AUTOR:

ALEX NIXON SALINAS GRANDA

DIRECTOR:

ING. GASTÓN RENÉ CHAMBA ROMERO, MG. SC.

LOJA – ECUADOR

2020

CERTIFICACIÓN DEL DIRECTOR



unl

Universidad
Nacional
de Loja

Facultad
de la Energía, las Industrias y los
Recursos Naturales No Renovables

CARRERA DE SISTEMAS

Certificado Nro. 001-GRCR-CIS-FEIRNNR-UNL

Ing. Gastón Rene Chamba Romero Mg.SC., **DIRECTOR DE TESIS.**

CERTIFICA:

Que el estudiante: Alex Nixon Salinas Granda C.I: **1105968141**, aprobó la asignatura de Trabajo de Titulación en el **décimo** Ciclo correspondiente al período académico: **15 de abril de 2019** al **6 de septiembre de 2019**; respecto del desarrollo de su tesis de grado titulada **“DESARROLLO DE UNA SOLUCIÓN INFORMÁTICA QUE PERMITA ADMINISTRAR UN HOGAR INTELIGENTE HACIENDO USO DE HARDWARE LIBRE”**, opción de titulación escogida dentro del período académico de culminación de sus estudios, siendo las **09H00** del **14 de febrero del 2020**, se certifica que se ha cumplido con el cien por ciento (**100%**) del trabajo de titulación y está en condiciones de continuar con los procesos administrativos que correspondan.

Loja, 14 de febrero del 2020

Gastón Rene Chamba Romero
DIRECTOR DE TESIS

AUTORÍA

Yo, **ALEX NIXON SALINAS GRANDA** declaro ser autor del presente Trabajo de Titulación y eximo expresamente a la Universidad Nacional de Loja y a sus representantes jurídicos de posibles reclamos o acciones legales, por el contenido de la misma.

Adicionalmente acepto y autorizo a la Universidad Nacional, la publicación de mi Trabajo de Titulación en el Repositorio Institucional – Biblioteca Virtual.

Autor: Alex Nixon Salinas Granda



Firma:

Cédula: 1105968141

Fecha: Loja, 14 de febrero del 2020

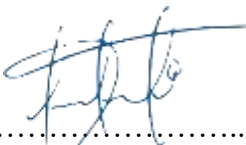
CARTA DE AUTORIZACIÓN DEL TRABAJO DE TITULACIÓN POR PARTE DEL AUTOR, PARA LA CONSULTA, REPRODUCCIÓN PARCIAL O TOTAL Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO.

Yo, **ALEX NIXON SALINAS GRANDA**, declaro ser la autora de la tesis titulada **“DESARROLLO DE UNA SOLUCIÓN INFORMÁTICA QUE PERMITA ADMINISTRAR UN HOGAR INTELIGENTE HACIENDO USO DE HARDWARE LIBRE”**, como requisito para optar el grado de: **INGENIERO EN SISTEMAS**; Autorizo al Sistema Bibliotecario de la Universidad Nacional de Loja para que con fines académicos, muestre al mundo la producción intelectual de la Universidad, a través de la visibilidad de su contenido en el Repositorio Digital Institucional:

Los usuarios pueden consultar el contenido de este trabajo en el RDI, en las redes de información del país y el exterior, con las cuales tenga convenio la Universidad.

La Universidad Nacional de Loja, no se responsabiliza por plagio o copia de la tesis que realice un tercero.

Para constancia de esta autorización, en la ciudad de Loja a los 10 días del mes de Julio del dos mil veinte.

Firma:


Autor: Alex Nixon Salinas Granda

Cedula: 1105968141

Dirección: Sector Época – (Av. Paltas entre Suecia e Irán)

Correo Electrónico: ansalinasg@gmail.com

Celular: 0986661807

DATOS COMPLEMENTARIOS:

Director de Tesis: Ing. Gastón René Chamba Romero Mg.Sc.

Tribunal de Grado: Ing. Pablo Fernando Ordoñez Ordoñez Mg. Sc.

Ing. María del Cisne Ruilova Sánchez Mg.Sc.

Ing. Francisco Xavier Álvarez Pineda Mg.Sc.

AGRADECIMIENTO

A Dios, por la vida que nos ha concedido a mi familia y a mí, y por todas las oportunidades que nos brinda.

A mi madre por el apoyo incondicional que me ha brindado, por hacer un trabajo casi perfecto y maravilloso conmigo; además por sacrificarse como nadie más por mí ayudándome a culminar mi carrera profesional.

Al Ing. Gastón Chamba quien con su experiencia y conocimiento fue mi guía en la elaboración y desarrollo del presente Trabajo de Titulación.

A los docentes de la carrera por los conocimientos impartidos en mi formación académica y por ultimo a todas las personas quienes me han apoyado no solo en los estudios sino también en cosas de la vida para crecer como persona y que me han ayudado a llegar hasta este punto y convertirme en un profesional.

DEDICATORIA

El presente Trabajo de Titulación se lo dedico a mi pequeña hija, a mi madre, hermanos y a todas las personas que siempre estuvieron ayudándome de una u otra manera durante mis años de estudio.

TABLA DE CONTENIDO

CERTIFICACIÓN DEL DIRECTOR.....	II
AUTORÍA	III
CARTA DE AUTORIZACIÓN DEL TRABAJO DE TITULACIÓN POR PARTE DEL AUTOR, PARA LA CONSULTA, REPRODUCCIÓN PARCIAL O TOTAL Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO.....	IV
AGRADECIMIENTO.....	V
DEDICATORIA	VI
1. TÍTULO.....	1
2. RESUMEN	2
SUMMARY	3
3. INTRODUCCIÓN	4
4. REVISIÓN DE LITERATURA	6
4.1 Internet De Las Cosas (IoT).....	6
4.1.1 Domótica	7
4.2 Protocolos de la Capa de Aplicación	9
4.2.1 Protocolos de la capa de aplicación para IoT.	10
4.3 Software Libre	22
4.3.1 Jeedom.....	22
4.3.2 Domoticz	26
4.3.3 Home Assistant.....	27
4.3.4 Node-RED	27
4.4 Hardware Libre.	28
4.4.1 Raspberry Pi	29
4.4.2 Arduino.....	30
4.4.3 Adafruit.....	33
4.5 Tecnologías y Herramientas utilizadas.	34
4.5.1 IDE de desarrollo Arduino.	34
4.5.2 Fritzing.....	35

4.5.3	MobaXterm.....	35
4.6	Trabajos Relacionados.....	35
5.	MATERIALES Y MÉTODOS.....	39
5.1	Materiales.....	40
5.2	Métodos.....	42
5.2.1	Método Deductivo	42
5.2.2	Método Bibliográfico	42
5.2.3	Método Analítico.....	43
5.3	Técnicas.....	43
5.3.1	Observación.....	43
5.3.2	Entrevista.....	43
5.3.3	Encuesta.....	43
5.4	Metodología.....	44
6.	RESULTADOS.....	46
6.1	Realizar una comparación técnica de los diferentes dispositivos de Hardware Libre y Software aplicado a Domótica.....	46
6.1.1	Requisitos Funcionales.....	46
6.1.2	Requisitos No Funcionales	47
6.1.3	Software Libre.....	47
6.1.4	Protocolo de comunicación para la solución Informática.....	48
6.1.5	Broker MQTT.....	49
6.1.6	Hardware Libre a Utilizar.....	51
6.2	Desarrollar el módulo de que permita la integración y administración de los componentes eléctricos y electrónicos del prototipo de la vivienda inteligente.....	52
6.2.1	Diseño de Alto Nivel	52
6.2.2	Diseño en Detalle.....	61
6.2.3	Estructura de los Tópicos	65
6.2.4	Diseño y Programación de los Sketches MQTT.....	65
6.3	Desarrollar un prototipo para la solución informática que permita administrar la programación de eventos para el hogar inteligente.....	78

6.4	Probar y evaluar el funcionamiento del prototipo del sistema de control del hogar inteligente en un entorno simulado.....	87
6.4.1	Pruebas.....	87
6.4.2	Reporte de Pruebas	96
6.4.3	Resultados de las Pruebas.....	104
7.	DISCUSIÓN	112
7.1	Desarrollo de la propuesta alternativa.....	112
7.2	Valoración técnica económica ambiental	115
8.	CONCLUSIONES	118
9.	RECOMENDACIONES	119
10.	BIBLIOGRAFÍA	120
11.	ANEXOS.....	124
	Anexo 1: Especificación de Requerimientos.....	124
	Anexo 2: Instalación y Configuración de Mosquitto.....	135
	Anexo 3: Instalación y Configuración de Node-RED.....	137
	Anexo 4: Implementación de las Funciones, Formularios en Node-RED para la Autenticación.....	143
	Anexo 5: Implementación.....	148
	Anexo 6: Manual de Usuario.....	151
	Anexo 7: Encuesta Para la Evaluación del Sistema Domótico.....	155
	Anexo 8: Código Fuente.....	158

ÍNDICE DE FIGURAS

Figura 1: Dispositivos Conectados a Internet y Evolución Futura	7
Figura 2: Aspectos que cubre la Domótica	8
Figura 3:Tendencias De Estándares de Mensajería de Comunicación	10
Figura 4: Arquitectura MQTT	12
Figura 5: Calidad de Servicio Nivel 0	12
Figura 6: Calidad de Servicio Nivel 1	13
Figura 7: Calidad de Servicio Nivel 2	13
Figura 8: Arquitectura Protocolo CoAP	15
Figura 9: Arquitectura AMQP	17
Figura 10: Arquitectura DDS	19
Figura 11: Interfaz de Escenario de Jeedom	22
Figura 12: Interfaz del Historial de Jeedom	23
Figura 13: Interfaz de la Visualización de Jeedom	23
Figura 14: Interfaz de las Iteraciones de Jeedom	24
Figura 15: Interfaz de Actualizaciones y RespalDOS de Jeedom	24
Figura 16: Interfaz de la Tienda de Jeedom	25
Figura 17: NODEMCU v1.0	33
Figura 18: ADAFRUIT HUZZAN ESP8266 BREAKOUT	34
Figura 19: Proceso de la metodología para el desarrollo de software y hardware	45
Figura 20: Diagrama de Casos de Uso	52
Figura 21: Diagrama de secuencia Autenticación	58
Figura 22: Diagrama de secuencia Sensores	59
Figura 23: Diagrama de Secuencia de los Actuadores	59
Figura 24. Diagrama de Secuencia de Gestión de Seguridad.....	60
Figura 25. Arquitectura del Sistema.....	61
Figura 26. Diagrama de Clases.....	62
Figura 27. Diagrama de Flujo Actuadores.....	63
Figura 28. Diagrama de Flujo Sistema de Seguridad	64
Figura 29: Estructura Jerárquica de los Tópicos que conformaran el Sistema.....	65
Figura 30: Características Ubuntu Server 18.04.....	67
Figura 31: Preferencias de Arduino.....	67
Figura 32: Configuración para la gestión de tarjetas ESP8266.....	68

Figura 33: Configuración para la programación del ESP8266.....	69
Figura 34: Configuración del Puerto Serial.....	69
Figura 35: Diseño de la Conexión para los LED's	70
Figura 36: Diseño de la Conexión del Sensor de Temperatura y Humedad DHT11	70
Figura 37: Diseño de la conexión para el motor de las puertas.....	71
Figura 38: Diseño de la conexión para los ventiladores.....	72
Figura 39: Diseño de la conexión Sensor PIR.....	72
Figura 40: Definición de Librerías	73
Figura 41: Definición de Pines GPIO.....	73
Figura 42: Definición de Variables y Constantes.....	74
Figura 43: Función Para la conexión a la Red Local.....	74
Figura 44: Función Callback	75
Figura 45: Función Reconexión	76
Figura 46: Método Setup.....	77
Figura 47: Método loop.....	77
Figura 48: Flujo de Autenticación.....	78
Figura 49: Flujo de Tiempo de Sesión.....	79
Figura 50: Estructura del Layout.....	79
Figura 51: Dashboard de Autenticación	80
Figura 52: Configuración bróker MQTT- Node-RED	81
Figura 53. Configuración del nodo MySQL.....	81
Figura 54. Estructura de la base de Datos en MySQL.....	82
Figura 55: Flujo de Iluminación	82
Figura 56. Funciones para guardar las acciones sobre la iluminación en MySQL.....	83
Figura 57: Apertura y Cierre de Puertas.....	83
Figura 58. Funciones para guardar las acciones sobre la puerta en MySQL.....	84
Figura 59: Flujo Control de Ventilación.....	84
Figura 60. Función para guardar las acciones sobre el ventilador en MySQL.....	85
Figura 61: Lectura de Temperatura y Humedad.....	85
Figura 62. Función para guardar la lectura del sensor DHT11 en MySQL.....	85
Figura 63: Flujo Control de Seguridad.....	86
Figura 64. Función para guardar la lectura del sensor PIR en MySQL.....	86
Figura 65: Interfaz del Sistema Domótico.....	87
Figura 66. Prueba Soporte datos entrada 1	98

Figura 67. Prueba de soporte de datos de entrada 2	98
Figura 68. Prueba de soporte de datos de entrada 3	99
Figura 69. Caso de Prueba Inicio de Sesión	99
Figura 70. Caso de Prueba Cierre de Sesión	100
Figura 71. Caso de Prueba cantidad de Usuarios por sesión	100
Figura 72. Caso de Prueba Encendido y Pagado de luces	101
Figura 73. Caso de Prueba Encendido y Apagado de Ventiladores	101
Figura 74. Caso de Prueba Apertura y Cierre de Puerta.....	102
Figura 75. Caso de Prueba Lectura de Temperatura y Humedad Ambiente	102
Figura 76. Caso de Prueba Gestión de Seguridad 1	103
Figura 77. Caso de Prueba Gestión de Seguridad 2	103
Figura 78: Resultados Evaluación del sistema Pregunta 1	105
Figura 79: Resultados Evaluación del sistema Pregunta 2	106
Figura 80: Resultados Evaluación del sistema Pregunta 3	106
Figura 81: Resultados Evaluación del sistema Pregunta 4	107
Figura 82: Resultados Evaluación del sistema Pregunta 5	107
Figura 83: Resultados Evaluación del sistema Pregunta 6	108
Figura 84: Resultados Evaluación del sistema Pregunta 7	108
Figura 85: Resultados Evaluación del sistema Pregunta 8	109
Figura 86: Resultados Evaluación del sistema Pregunta 9	110

ÍNDICE DE TABLAS

Tabla I. Resumen de Protocolos de Comunicación Para IoT	21
Tabla II. Características de Raspberry Pi 3 Modelo B+	30
Tabla III. Materiales Usados	40
Tabla IV. Descripción de Requerimientos Funcionales	46
Tabla V. Descripción de Requerimientos No Funcionales.....	47
Tabla VI. Características de los Protocolos MQTT y COAP	48
Tabla VII. Descripción de Casos de Uso N°1	53
Tabla VIII. Descripción de Casos de Uso N°2.....	53
Tabla IX. Descripción de Casos de Uso N°3.....	54
Tabla X. Descripción de Casos de Uso N°4.....	55
Tabla XI. Descripción de Casos de Uso N°5.....	56
Tabla XII. Descripción de Casos de Uso N°6	56
Tabla XIII. Descripción de Casos de Uso N°7	57
Tabla XIV. Elementos que formaran parte del Sistema Domótico	66
Tabla XV. Casos de Prueba.....	89
Tabla XVI. Caso de Prueba Inicio de Sesión	90
Tabla XVII. Caso de Prueba Cierre de Sesión	91
Tabla XVIII. Caso de Prueba Cantidad de Usuarios autenticados al mismo tiempo.	91
Tabla XIX. Caso de Prueba Encendido y Apagado de Luces	92
Tabla XX. Caso de Prueba Encendido y apagado de Ventiladores.....	92
Tabla XXI. Caso de Prueba Cierre y Apertura de Puertas	93
Tabla XXII. Caso de Prueba Lectura de Temperatura y Humedad Ambiente	93
Tabla XXIII. Caso de Prueba de Gestión de Seguridad.	94
Tabla XXIV. Verificación de Pantallas	97
Tabla XXV. Valoración Económica del Proyecto.....	115

1. TÍTULO

**“DESARROLLO DE UNA SOLUCIÓN
INFORMÁTICA QUE PERMITA
ADMINISTRAR UN HOGAR
INTELIGENTE HACIENDO USO DE
HARDWARE LIBRE”**

2. RESUMEN

El presente Trabajo de Titulación tiene como objetivo el diseño e implementación de una solución informática que permita administrar un hogar inteligente a través de la utilización de herramientas de bajo coste o de uso libre. En si consiste en el desarrollo de un prototipo de sistema domótico de fácil manejo por parte de los usuarios, teniendo como punto principal el control de la iluminación, ventilación, gestión de puertas, gestión de temperatura y seguridad de la vivienda, con la capacidad de monitoreo constante y remoto.

Por otra parte, se hace uso de la plataforma NodeRED que es una herramienta Open-Source basada en Node.js que hace más fácil la integración del hardware con el software de manera rápida y sencilla a través de una programación visual basada en nodos. El sistema puede controlar los actuadores y recibir datos de los sensores por medio de un servidor web; para la comunicación de los mismos se hace uso del protocolo ligero MQTT (Message Queue Telemetry Transport).

Asimismo, se emplea el microcontrolador NodeMCU V1 basada en el Chip ESP8266 que facilita la conexión y comunicación de los sensores y actuadores de manera muy económica e inalámbrica debido a que es una placa de desarrollo totalmente abierta a nivel de software y hardware.

SUMMARY

The objective of this Degree Project is to design and implement a computer solution that allows managing a smart home through the use of low-cost or free-use tools. In itself it consists in the development of a prototype of a home automation system that is easy for users to use, having as its main point the control of lighting, ventilation, door management, temperature management and home security, with the ability to constant and remote monitoring.

On the other hand, the NodeRED platform is used, which is an Open-Source tool based on Node.js that makes it easier to integrate hardware with software quickly and easily through visual node-based programming. The system can control the actuators and receive data from the sensors through a web server; For the communication of the same, the light protocol MQTT (Message Queue Telemetry Transport) is used.

Likewise, the NodeMCU V1 microcontroller based on the ESP8266 Chip is used, which facilitates the connection and communication of the sensors and actuators in a very economic and wireless way, since it is a completely open development board at the software and hardware level.

3. INTRODUCCIÓN

Hoy en día existe una gran demanda en la sociedad por el confort, la gestión de energía eléctrica, la seguridad personal y de bienes existentes en el hogar. Cada vez toma más fuerza; por lo tanto, la domótica ofrece una solución dirigida a todo tipo de viviendas de una manera muy intuitiva, de calidad y que puede ser manejada por cualquier persona, generando así un fuerte impacto gracias a sus ventajas que brinda y sobre todo por la capacidad de acoplarse a dispositivos remotos, tales como Smartphone, Tablets, entre otros dispositivos inteligentes.

En la actualidad gran cantidad de hogares han empezado a incorporar una serie de automatismos, la mayoría son basados en el control de iluminación, puertas o seguridad, todos estos instalados de forma individual sin estar conectados a un solo dispositivo central, razón por la cual se estaría refiriendo a un hogar automatizado mas no domotizado. Por ello se propone una solución tecnológica con la ayuda de la plataforma Node-RED, el protocolo MQTT y la placa NodeMCU ESP8266, mismos que permitirán al usuario gestionar actividades de su vivienda de forma remota siempre y cuando tenga acceso a internet, de esta forma se crea un prototipo de sistema domótico de bajo costo y de fácil uso, mejorando significativamente la calidad de vida de los usuarios.

Para cumplir con éxito del presente trabajo de Titulación se plantearon los siguientes objetivos específicos que nos ayudaran abordar todo el trabajo.

1. Realizar una comparación técnica de los diferentes dispositivos de Hardware Libre y Software aplicado a Domótica.
2. Desarrollar el módulo que permita la integración y administración de los componentes eléctricos y electrónicos del prototipo de la vivienda inteligente.
3. Desarrollar un prototipo para la solución informática que permita administrar la programación de eventos para el hogar inteligente.
4. Probar y evaluar el funcionamiento del prototipo del sistema de control del hogar inteligente en un entorno simulado.

Además, cuenta con varias secciones, entre ellas la revisión de estado del arte de la domótica, un estudio de soluciones existentes en el mercado tanto de hardware y software libre utilizados en la domótica, detallando sus características y sus aplicaciones, así mismo se realiza una revisión sobre los protocolos de comunicación a nivel de la capa de aplicación del modelo OSI para IoT. La sección de materiales y

métodos abarca los diferentes materiales, métodos, técnicas y metodología usada para el desarrollo del presente trabajo de titulación. Otra sección es la de Resultados que tiene la siguiente estructura: Estudio de hardware y software aplicado a domótica, desarrollo de modulo para integrar los dispositivos eléctricos y electrónicos, desarrollo de la solución informática, prueba y evaluación. El estudio de hardware libre y software aplicado a domótica consistió en realizar un análisis del hardware y software libre que se utiliza para realizar sistemas domóticos de manera segura, fácil y económica. Para desarrollar el módulo para integrar los componentes eléctricos y electrónicos, se lo realizo componente por componente realizando pruebas de su respectivo funcionamiento. Al desarrollar la solución informática se lo realizo utilizando la herramienta de programación visual NodeRED la cual permitió crear los flujos de trabajo para la conexión entre el software y el hardware. En la prueba y evaluación de la solución informática se integró todos los módulos y se puso en funcionamiento todo el sistema para luego proceder a manejar todo el sistema enviando diferentes instrucciones de diferentes lugares para ver el comportamiento del mismo.

También están las secciones de Discusión, Conclusiones y Recomendaciones, en la sección de Discusión se presenta una explicación de cómo se realizó todo el proceso que se llevó a cabo para cumplir los objetivos planteados, en la sección de Conclusiones se expone las deducciones de las experiencias obtenidas durante el cumplimiento de los objetivos y en la última Sección de Recomendaciones se describen algunas sugerencias para un mejor desarrollo de temas similares.

4. REVISIÓN DE LITERATURA

4.1 Internet De Las Cosas (IoT).

El término "Internet de las cosas" o Internet of Things (IoT) fue introducido por primera vez en 1999 por Kevin Ashton. Ashton visualiza que el mundo físico puede ser conectado vía internet con sensores capaces de brindar información necesaria y en tiempo real para beneficiar la calidad de vida. [1]. Hoy en día el termino IoT ha ganado popularidad en escenarios en los que la conectividad a Internet y la capacidad de cómputo se extienden a un sinnúmero de objetos, dispositivos, sensores y artículos de uso diario, ofrece la promesa de oportunidades para desarrollar nuevos servicios e integrar diferentes dominios de aplicación [2].

La IoT trata sobre la conexión de objetos físicos con componentes electrónicos integrados, software, sensores y conectividad permitiendo así poder medir desde la temperatura de una habitación hasta medir el nivel de tráfico de vehículos en una ciudad [3]. La IoT ha logrado que la internet sea sensorial (temperatura, presión, vibración, luz, humedad, estrés, etc.) además, se expande a lugares donde se pensaba que eran inalcanzables como por ejemplo un paciente puede ingerir pequeños sensores que están conectados a internet con los cuales ayudan a los médicos a diagnosticar y determinar causas de ciertas enfermedades. En general se puede colocar sensores microscópicos en plantas, animales, humanos, fenómenos geológicos y conectarlos a internet para obtener de manera masiva información que es necesaria no solo para sobrevivir sino para seguir mejorando continuamente en un futuro [4]. También, permite ver y controlar cosas a distancia por la infraestructura de la red existente en la actualidad, lo que resulta preciso, eficiente y con beneficios económicos ya que optimiza el tiempo que toma la intervención de un humano. La IoT con la ayuda de sensores y actuadores engloba una variedad de tecnologías tales como hogares inteligentes, transporte inteligente, ciudades inteligentes, redes inteligentes, etc., [5].

Como se muestra en la Figura 1. Las predicciones según CISCO para el 2020 se estima un total de 50 millones de objetos conectados cuando la población mundial alcance los 7.6 millones de habitantes por lo que tendremos muchos más dispositivos conectados que personas [6].

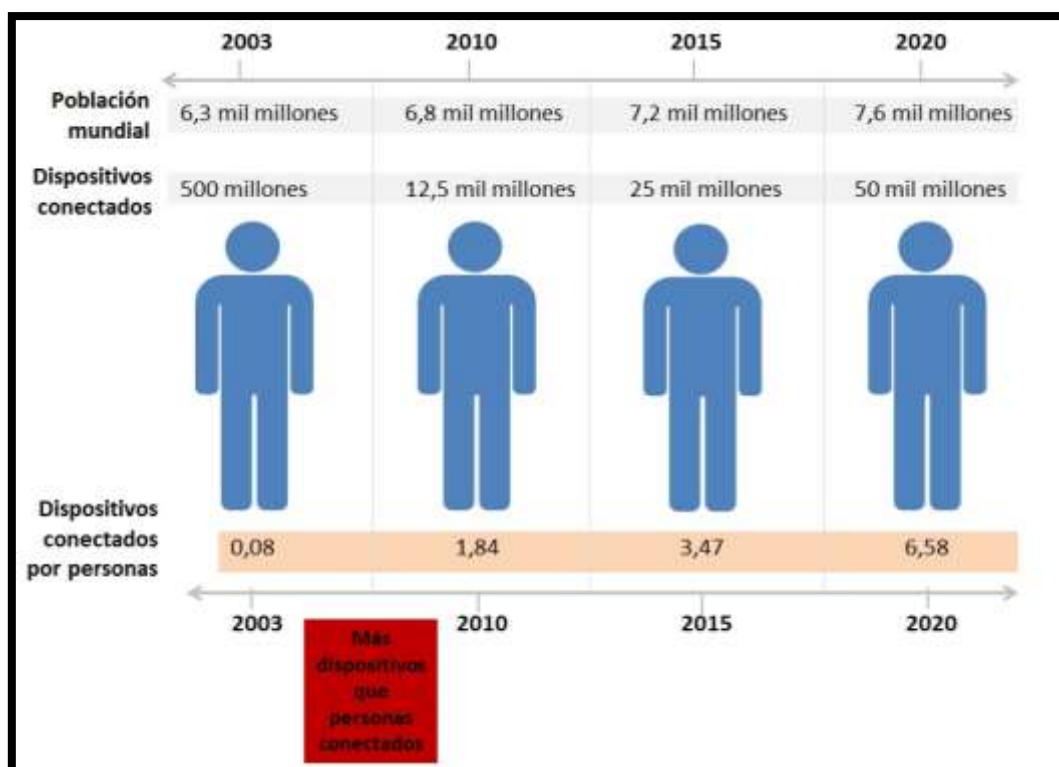


Figura 1: Dispositivos Conectados a Internet y Evolución Futura [4]

De acuerdo a estudios y estimaciones, entre 25 a 30 personas emigran a las principales ciudades a cada minuto, se estima que alrededor de 843 millones de ciudadanos serán ciudadanos residentes en zonas urbanas extensas para el año 2050. Por tal motivo para mantener activa esta masiva cantidad de personas nace la necesidad de crear técnicas más eficientes para dar cabida a las complejidades inherentes y aumentar la eficiencia de los procedimientos actuales llevados en las ciudades. Es por eso que la próxima generación se basara en tener disponibilidad de comunicaciones y tener una buena infraestructura es de aquí que nace requerimientos para lograr esto los cuales son: Gobierno inteligente (A través de aplicaciones innovadoras y en línea), energía, medio ambiente, transporte, tecnologías de información y comunicación, edificios, hospitales de salud y educación inteligentes. Dadas las tendencias de la continua urbanización, los retos asociados deben ser examinados cuidadosamente con el fin de garantizar un crecimiento sostenible y evitar futuras complicaciones no deseadas [7].

4.1.1 Domótica

La domótica es un conjunto de tecnologías capaces de recoger información de una serie de sensores y/o dispositivos diferentes, para procesarla y actuar acorde a la situación enviando instrucciones a diferentes actuadores que están incorporados al sistema

inteligente mejorando así la calidad de vida de las personas [6]. El termino domótica es utilizado para referirse a la automatización de una vivienda que permite controlarla y administrarla tanto local como remotamente mediante el uso de sensores, actuadores y tecnologías de información [8].

La popularidad de la Domótica ha aumentado enormemente en los últimos años debido a una mayor accesibilidad y simplicidad a través de la conectividad de los Smartphone y tabletas. Con la ayuda de estos dispositivos se podrá controlar sistemas de entretenimiento en el hogar, riego de jardines, alimentación de mascotas, cambiar ambientes para diferentes eventos (como fiestas), optimizar el consumo de energía eléctrica, optimizar la seguridad del hogar etc., [9].

La domótica ofrece servicios en cinco diferentes áreas (Ver Figura 2):

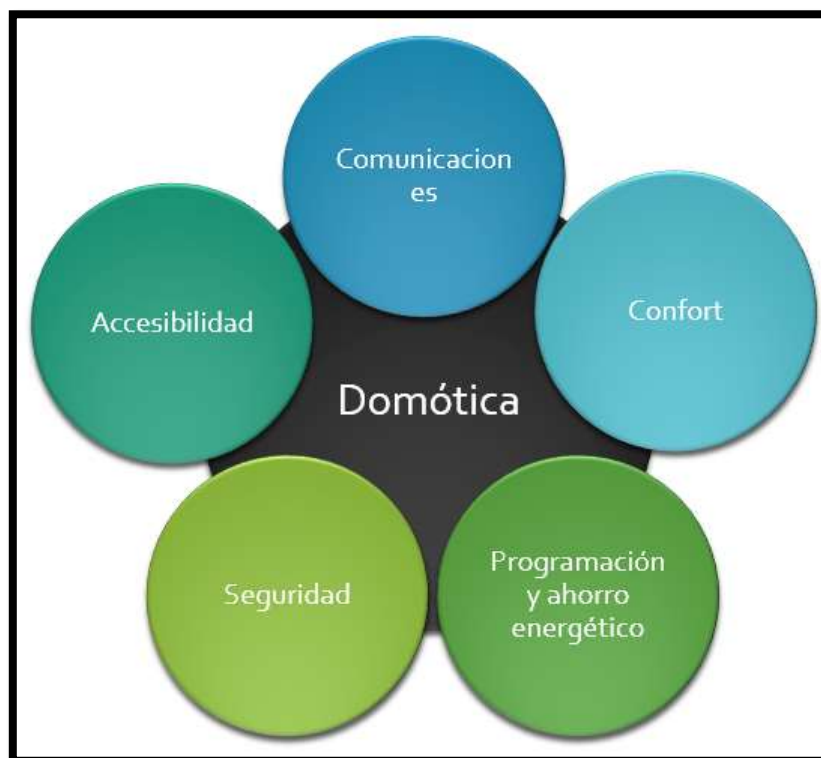


Figura 2: Aspectos que cubre la Domótica [8].

- **Seguridad y Vigilancia:** Consiste en resguardar la seguridad personal y de los bienes protegiéndolos de intrusos a toda hora de tal manera que funcione de manera automática con la ayuda de cámaras, sensores de presencia y alarmas, además nos proporciona seguridad contra accidentes detectando inicios de incendios, fugas de agua, humo etc.

- **Comunicaciones:** Permite la comunicación entre los distintos sensores y dispositivos que conforman la red de la vivienda logrando así controlar por ejemplo el encendido luces, ventiladores, el riego del jardín etc. Además, podemos controlarlas no solo desde el interior de la vivienda sino de cualquier parte del mundo en el que contemos con acceso a internet.
- **Confort:** Facilita al usuario obtener un mayor nivel de comodidad en cada actividad a desarrollar dentro del hogar permitiendo por ejemplo controlar a distancia la apertura de puertas, control de alarmas, control de luces, persianas, electrodomésticos etc.
- **Accesibilidad:** Este aspecto consiste en dar la facilidad de acceso dentro del hogar a personas que sufran de alguna discapacidad o problema de algún tipo.
- **Programación y Ahorro Energético.** Optimiza inteligentemente los diferentes recursos disponibles en el hogar como la electricidad, sistema de riego, persianas, por ejemplo, levantar persianas a una hora programada o apagar las luces cuando exista una gran cantidad de luz natural etc., [8][9].

4.2 Protocolos de la Capa de Aplicación

Un gran desafío de la IoT es la interoperabilidad. En Una encuesta reciente de Nexus, el setenta y siete por ciento (77%) de los entrevistados consideró que la interoperatividad es el mayor de los desafíos de la IoT. Conectar dispositivos industriales con tecnologías de información y plataformas IoT es un tema considerable. Existen Varios Protocolos para complementar esto: algunos son privados y otros que son estándares abiertos [10].

Una de las características principales de los dispositivos conectados a la red es que están equipados de baterías, tienen un mínimo de almacenamiento y capacidades de proceso. Debido a estas restricciones, la comunicación entre estos dispositivos acarrea varios desafíos como son:

- Direccionamiento e identificación
- Comunicaciones con bajo consumo de energía.
- Protocolos de comunicación eficientes y de bajos requerimientos de memoria.
- Alta velocidad y comunicación sin pérdida de datos.
- Movilidad.

Los dispositivos IoT, generalmente se conecta a Internet a través de la pila TCP/IP, esta pila es compleja y requiere de gran cantidad de energía. También se pueden conectar localmente a través de redes no IP, consumen poca energía y se conectaba través de una pasarela (Gateway). Algunas Redes no IP como, Bluetooth, RFID, NFC son populares, pero tienen poco alcance. Para alcanzar dicho alcance es necesario modificar la pila TCP/IP con el objetivo de reducir el consumo de energía [11].

4.2.1 Protocolos de la capa de aplicación para IoT.

En la capa de aplicación se proporciona la interfaz y servicios que soportan las aplicaciones de usuario; es decir, suministra las herramientas que el usuario puede ver. Adicionalmente gestiona los servicios relacionados como la transferencia de archivos y consulta de base de datos. También permite que los dispositivos se intercomunicen a través de procesos, dicho intercambio de información de estos procesos se efectúa por medio de algún protocolo de la capa de aplicación.

En esta capa existen muchos protocolos para IoT que facilitan y simplifican la programación de aplicaciones a los programadores y proveedores de servicios. Estos protocolos permiten comunicar el Hardware con el Software y que según un estudio realizado por la fundación de Eclipse (Ver Figura 3) en la actualidad son muy utilizados entre los principales podemos destacar:

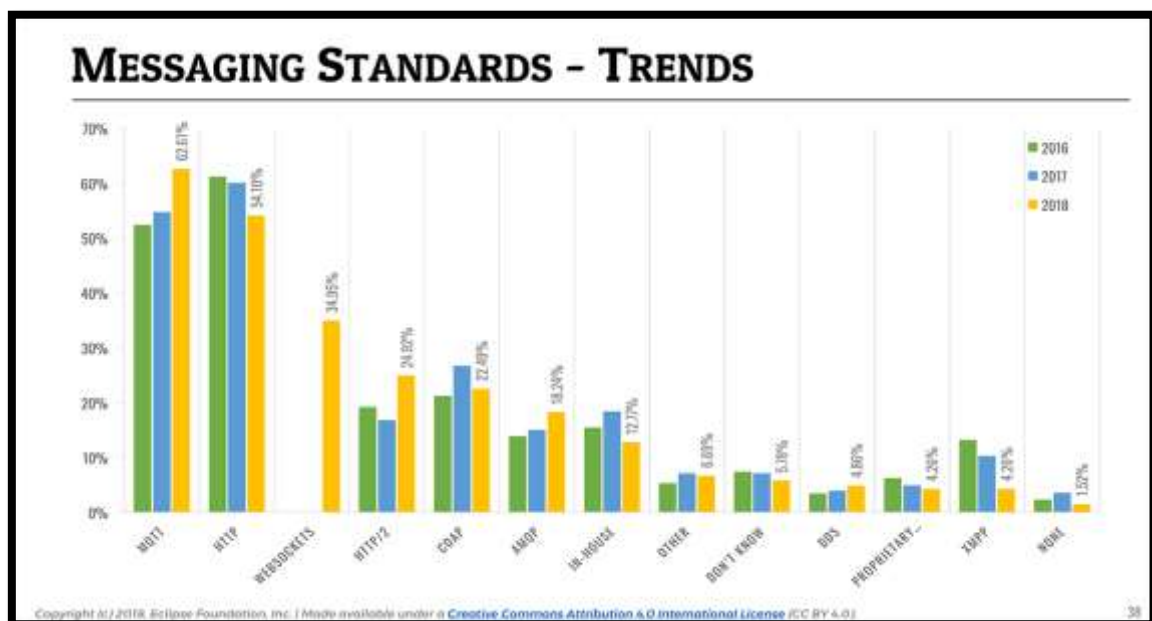


Figura 3: Tendencias De Estándares de Mensajería de Comunicación [12]

4.2.1.1 MQTT (Message Queuing Telemetry Transport, 'Cola de mensajes telemetría y transporte')

MQTT fue desarrollado por el Dr. Andy Stanford-Clark de IBM y Arlen Nipper de Arcom (actualmente Eurotech) en el año 1999 para conectar diferentes dispositivos de monitorización que fueron utilizados en la industria petrolera cuyos servidores estaban ubicados en distintos lugares [8]. MQTT es un protocolo publicación/ suscripción diseñado para SCADA (Supervisory Control And Data Acquisition, 'Supervisión, Control y Adquisición de Datos') y redes remotas. Se centra en un mínimo encabezado de 2 bytes y comunicaciones confiables de tal manera que utiliza un mínimo ancho de banda. MQTT posee puertos TCP/IP estandarizados para poder utilizarlo. El puerto 1883 está reservado con IANA (Internet Assigned Numbers Authority) para su uso básico y el puerto 8883 se lo utiliza para realizar el cifrado de los datos, aunque esto supone una sobrecarga significativa de la red por lo que no es recomendable la utilización de esta alternativa.

La estructura de los "tópicos" es jerárquica y los niveles temáticos se separan con el carácter "/"; por ejemplo "/casa/sala/sensor"; además si se quiere obtener la información de todos los sensores solo se utiliza el carácter "#" quedando el ejemplo de la siguiente manera: "/casa/sala/#" y ya no es necesario definir cada uno de los tópicos "/casa/sala/temperatura" y "/casa/sala/humedad" (un tema para el sensor de temperatura y el otro para el sensor de humedad).

- **Arquitectura**

MQTT se compone de tres elementos (Ver Figura 4): Subscriptores, Publicadores y el Bróker. Esto quiere decir que un dispositivo se registra como subscribers en un tema en especial para que sea informado por el bróker, cuando el Publisher publique temas de su interés. El Publisher transmite información a las entidades interesadas (subscribers) a través del bróker y por último el Bróker comprueba que ambos subscribers y Publisher estén autorizados.

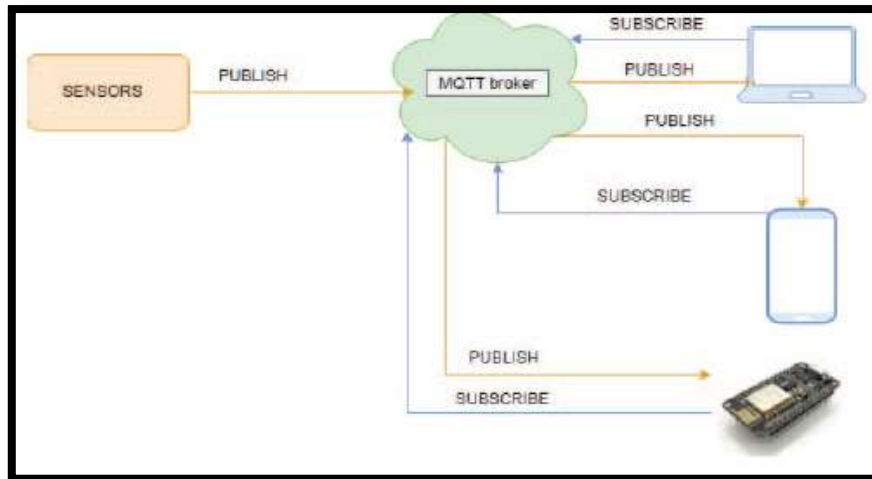


Figura 4: Arquitectura MQTT [13]

- **Niveles de QoS**

MQTT aborda principalmente tres propiedades de calidad de servicio los cuales son utilizados para gestionar de manera óptima la comunicación, en función de la aplicación en que se la utilice. Cabe mencionar que existen niveles superiores de calidad de servicio que permiten comunicaciones más confiables, pero esto también conlleva a que se utilizara más recursos de ancho de banda por el incremento de la latencia. A continuación, se detalla cada uno de los niveles de calidad de servicio en el proceso de publicación de datos.

- **QoS 0:** Con el término “**Como máximo una vez**” (Ver Figura 5), que es el nivel más básico. Sin embargo, este nivel se suma la importancia de la aplicación, donde el usuario puede enviar el mensaje en la forma más rápida utilizando el protocolo MQTT sin esperar las respuestas, en otras palabras, se envía una secuencia de paquetes PUBLISH al bróker para un tema determinado y este lo reenvía una sola vez hacia los clientes suscritos a dicho “topic”.

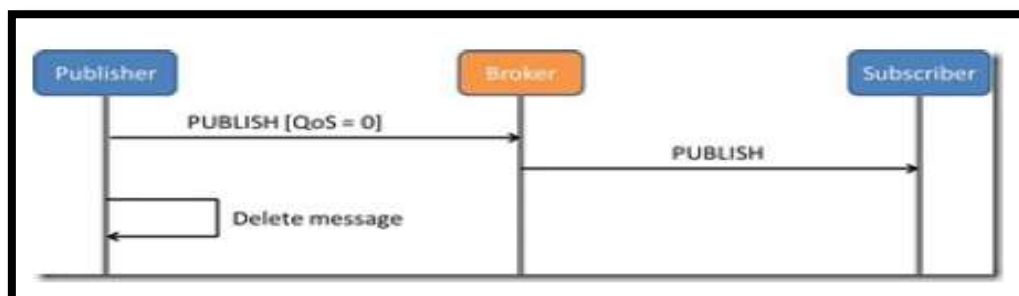


Figura 5: Calidad de Servicio Nivel 0 [6].

- **QoS 1:** es el siguiente nivel con el término **“al menos una vez Entrega”** (Ver Figura 6) en la que el cliente o el servidor deben enviar al menos un mensaje, independientemente de los mensajes duplicados. Además, se utilizará un acuse de recibo para indicar la correcta recepción de la información. De este modo, si no se recibe este acuse, el bróker volverá a enviar el contenido original de la publicación; dando lugar, como inconveniente, a una posible duplicidad de la información.

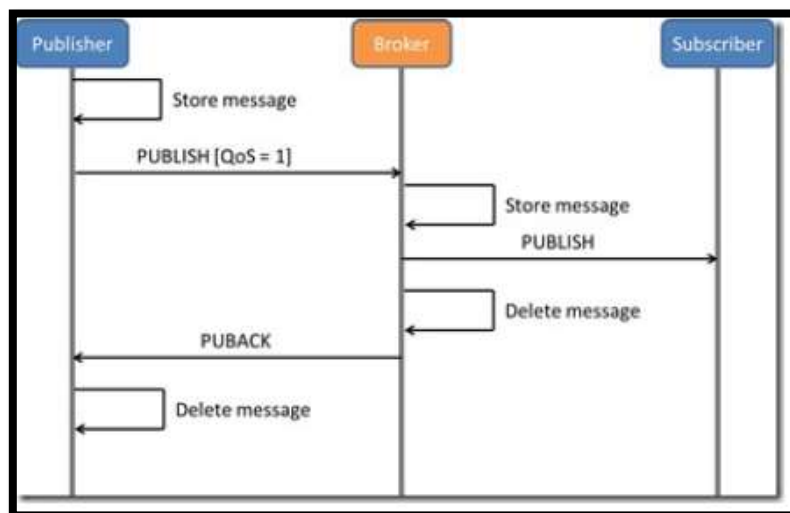


Figura 6: Calidad de Servicio Nivel 1 [6].

- **QoS 2:** es el último con más alto nivel de calidad de servicio, conocida como **“exactamente una vez la entrega”** (Ver Figura 7) en el que los mensajes se transfieren sólo una vez sin permitir duplicar la información. Este proceso lo realiza mediante dos duplas de mensajes La primera de ellas, está compuesta por los paquetes PUBLISH/PUBREC y la segunda por PUBREL/ PUBCOMP.

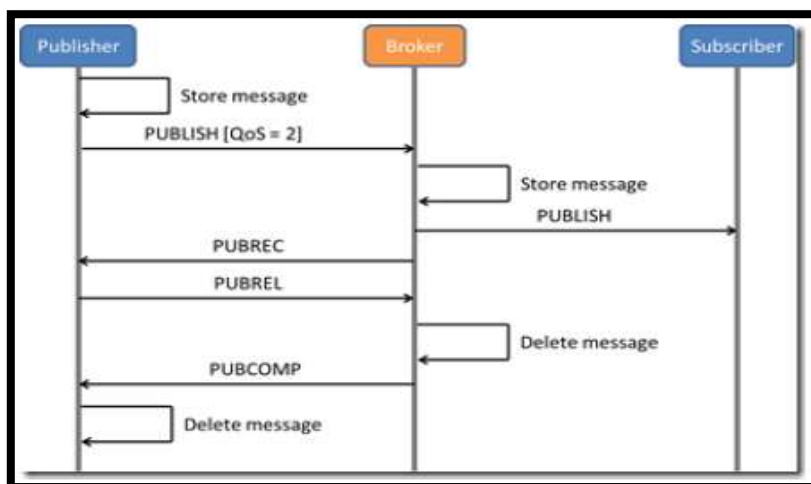


Figura 7: Calidad de Servicio Nivel 2 [6].

- **Seguridad**

MQTT dará flexibilidad de añadir la función de seguridad para aplicaciones para cifrar los datos que envía y recibe, con el fin de mantener el protocolo ligero y simple. En MQTT existen tres conceptos fundamentales en cuanto a la seguridad estos son: Identidad, autenticación, y autorización. La identidad consiste en dar nombre al cliente que se va a dar acceso. La autenticación consiste en probar la identidad del cliente y la Autorización consiste en gestionar los derechos que se otorgan al cliente. Todas estas acciones las realiza mediante el protocolo SSL.

- **Ventajas**

- El paradigma que usa es publicar / suscribirse paradigma que le da baja sobrecarga y hace este es un protocolo ligero.
- Como utiliza un corredor para transferir los mensajes entre la autenticación del cliente pueden ser implementado en el corredor que da seguridad a los datos.
- La función QoS del protocolo ayuda en el manejo de las redes poco confiables se hace mediante el uso de diferentes niveles de calidad de servicio para diferentes tipos de mensajes.

- **Aplicaciones**

- Domótica
- Aplicaciones a nivel empresarial

4.2.1.2 CoAP (Constrained Application Protocol, 'Protocolo de aplicación restringida').

Fue creado por IETF (Internet Engineering Task Force, 'Grupo de Trabajo de Ingeniería de Internet') para proveer la compatibilidad de HTTP con una mínima carga. CoAP es similar a HTTP, pero usa UDP/multicast en lugar de TCP. Además, simplifica el encabezado HTTP y reduce el tamaño de cada requerimiento. CoAP se utiliza en dispositivo de borde en donde HTTP sería demandante de recursos, y a menudo, las plataformas de IoT lo utilizan como tercer protocolo, después de HTTP y MQTT. Similar a HTTPS [10]. Una de las principales metas de CoAP es diseñar un protocolo web genérico para los requisitos especiales de entornos restringidos, considerando

especialmente las necesidades energéticas, automatización de edificios y otras aplicaciones M2M.

CoAp es un protocolo con el mensaje de petición/respuesta, este mantiene el tamaño del mensaje tan pequeño como sea posible y compatible con otro mecanismo de retransmisión. En otras palabras, los usuarios CoAP pueden utilizar GET, PUT, POST o eliminar los métodos para gestionar los recursos identificados en la red.

- **Arquitectura**

CoAP mantiene la arquitectura Cliente/ Servidor (Ver Figura 8) y soporta operaciones GET, PUT, POST, DELETE. Además, extiende el modelo de request añadiendo la función observe que permite al usuario recibir cambios de un recurso solicitado al servidor.

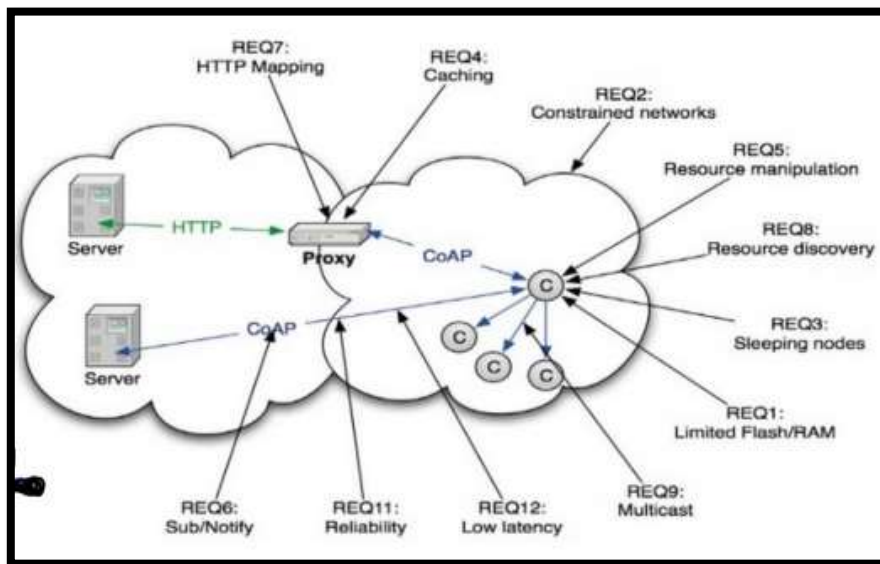


Figura 8: Arquitectura Protocolo CoAP [14].

- **Niveles de Q&S**

Esta capa define cuatro tipos de mensajes: CON (Confirmable), NO (no confirmable), ACK (acuse de recibo) y RST (reset). Los mensajes CON se utilizan para asegurar una comunicación fiable, y exigen una confirmación desde el lado del receptor con un mensaje ACK. Precisamente esta característica que marca si los mensajes necesitan el reconocimiento es lo que permite la diferenciación de QoS en el COAP, aunque de forma limitada [15].

- **Seguridad**

CoAP usa DTLS (Datagram Transport Layer Security, 'seguridad en la capa de transporte datagrama') para proteger las comunicaciones. La mayor parte de las modificaciones en comparación con TLS incluyen características que impiden la terminación de conexión en caso de pérdida o fuera de los paquetes de pedidos. A modo de ejemplo, existe la posibilidad de retransmitir los mensajes de saludo. proceso de apretón de manos es muy similar a la de TLS, con el intercambio de cliente y servidor de mensajes 'hola', pero con la posibilidad adicional de un servidor para enviar una consulta de verificación para asegurarse de que el cliente estaba enviando su mensaje 'hola' a partir la dirección de origen auténtico. Este mecanismo ayuda a prevenir los ataques de denegación de servicio [15].

- **Ventajas**

- Simplifica la cabecera HTTP y reduce el tamaño de cada solicitud.
- CoAP se emplea en dispositivos perimetrales donde HTTP consumiría demasiados recursos.
- Es el tercer protocolo admitido por plataformas de IoT.

- **Aplicaciones**

- Ciudades Inteligentes
- Red Inteligente
- Automatización de edificios
- Comunicaciones en grupo.
- Logística de transporte.

4.2.1.3 AMQP (Advanced Message Queuing Protocol).

Es otro protocolo de estándar abierto de tipo publicación /suscripción. Una gran cualidad de AMQP es que su modelo es robusto. A diferencia de MQTT, AMQP puede garantizar transacciones de mensajes completas, aunque en aplicaciones de IoT no siempre se requiere estas cualidades. El objetivo es asegurar que la información se transporte de forma segura y eficientes entre aplicaciones, entre organizaciones, a través de entornos distribuidos de computación en nube y dentro de infraestructura móviles. AMQP está

diseñado para la interoperabilidad entre una amplia gama de aplicaciones y sistemas diferentes ya que permite a diferentes plataformas que están implementadas en diferentes idiomas poder intercambiar mensajes [11][15].

- **Arquitectura**

La arquitectura de AMQP contiene el intermediario de tres componentes y un receptor. La comunicación entre el editor y el suscriptor será a través de intercambios con la ayuda del corredor. El editor es responsable de producir y reenviar mensajes al agente. El Bróker proporciona dos servicios: uno es el intercambio y la otra es la de mantener las colas. Los intercambios se utilizan para reenviar los mensajes a las colas específicas siguiendo reglas predefinidas. Considerando que, las colas pueden almacenar mensajes y transmitirlos al receptor y el Receptor almacenará dichos mensajes [14].

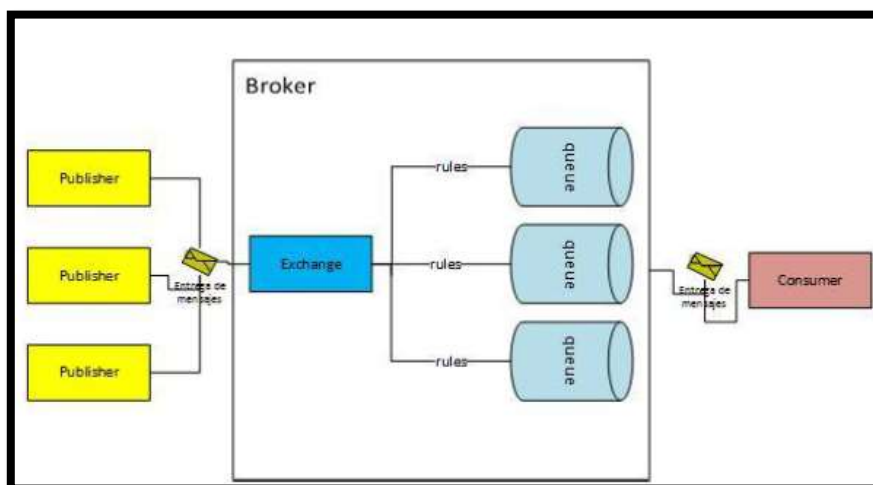


Figura 9: Arquitectura AMQP[11].

- **Niveles de Q&S**

AMQP utiliza TCP para un transporte fiable, y además proporciona tres niveles diferentes de QoS, al igual que MQTT (Ver Figura 5,6,7) [15].

- **Seguridad**

La seguridad que proporciona el protocolo AMQP para los datos lo hace mediante el uso del protocolo TLS para el cifrado, y para la autenticación mediante el uso de SASL (Simple Autenticación y capa de seguridad) [14].

- **Ventajas**
 - Altamente fiable para compras Online y comunicación de datos.
 - Garantiza la entrega de los mensajes.

- **Aplicaciones**
 - Es utilizado por muchos bancos famosos como JP Morgan y el Deutsche Börse para transmisiones de datos más pesados.
 - AMQP se utiliza en UIDAI, el gobierno de la India para la recolección y mantenimiento de datos de 1,2 mil millones de personas.
 - Se utiliza en los servicios de computación en la nube de la NASA para la nebulosa y Red Hat Linux para sus comunicaciones internas.
 - National Science Foundation está utilizando AMQP para transmitir datos desde los barcos a la costa [14].

4.2.1.4 DDS (Data Distribution Service).

Este protocolo de datos fue diseñado por el grupo de gestión de objetos (OMG). Su comunicación es de forma multicast (M2M) dentro de la IoT, se ejecuta sobre el protocolo UDP. DDS proporciona el intercambio de mensajes simultáneos, dicho de otra manera puede enviar miles de mensajes por unidad de tiempo a diferentes receptores sin perder dicha información [14]. DDS es un protocolo de publicación / suscripción en tiempo real centrados en datos para sistemas distribuidos altamente dinámicos. El uso de los datos es fundamentalmente anónimo, ya que los editores no preguntan sobre que consume sus datos. A diferencia de otros protocolos este no depende de un Broker y se basa en la idea de “espacio global de datos” donde los productores escriben los datos y los consumidores leen este mensaje. Un problema del DDS es que no se ha utilizado ampliamente, aunque esto puede cambiar con las nuevas tecnologías también cabe mencionar que existen implementaciones de DDS de código abierto listas para ser probadas, como OpenDDS [11][15].

- **Arquitectura**

Las principales entidades en la arquitectura DDS incluyen: Dominio, Participante del Dominio, Tema, editor, un suscriptor, un grabador de datos y un lector de datos. DDS

tiene una arquitectura de un corredor-menos (Ver Figura 10) que contiene el sistema objeto de datos entre los abonados y el editor. Estos editores y suscriptores se conectan a través de una red, los temas están relacionados con los escritores de datos y lectores de datos. Cuando el editor transmite datos, a continuación, los datos se pasan al escritor de datos. Considerando que, lectores de datos son capaces de almacenar datos de diferentes usuarios [11][14][15].

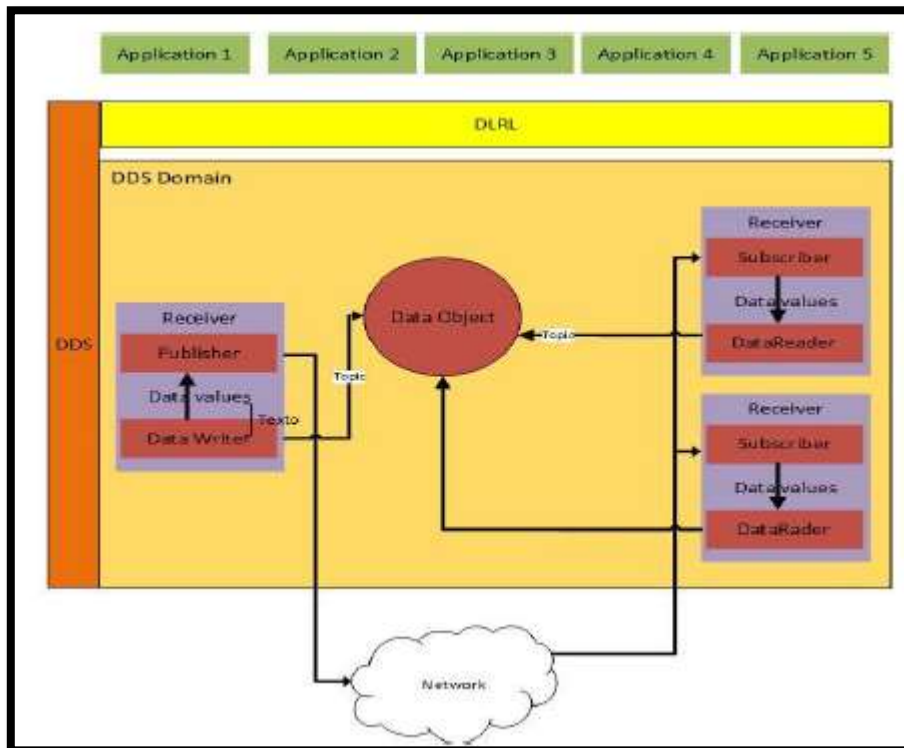


Figura 10: Arquitectura DDS [11].

- **Niveles de Q&S**

DDS cuenta con un excelente nivel de calidad de servicio y fiabilidad ya que cuenta con 23 niveles. DDS proporciona un amplio conjunto de parámetros de QoS, por ejemplo, durabilidad, vida útil, presentación, confiabilidad, priorización, urgencia y plazos [11][14][15].

- **Seguridad**

La seguridad DDS define un amplio modelo de seguridad y una interfaz de plugin de servicio (SPI).

DDS implementa varias soluciones. Basado en un protocolo de transporte de TLS puede utilizarse en caso de que TCP sea el protocolo de transporte, o el protocolo DTLS en caso de que se utilice UDP.

Del mismo modo, en el caso de TLS, DTLS supone una sobrecarga excesiva en entornos restringidos, para los cuales se han propuesto mecanismos mejorados. La especificación de seguridad sigue siendo una especificación abierta para DDS y se espera que las nuevas adiciones se implementarán en el futuro [14][15].

- **Ventajas**

- Una de las características más destacadas del protocolo DDS es su escalabilidad
- Su paradigma es enfocado a los datos y no a los mensajes.
- Una de las ventajas de usar DDS es un amplio conjunto de políticas de QoS ofrecidas.

- **Aplicaciones**

- Sectores industriales como en las redes de ferrocarril.
- Control de tráfico aéreo.
- Energía inteligente.
- Servicios médicos.
- Automatización militar y aeroespacial.

Tabla I. Resumen de Protocolos de Comunicación Para IoT

Protocolo	Características	Entorno de Trabajo	Ventajas	Desventajas	Aplicaciones
MQQT	Bajo consumo de recursos. Comunicación M-M.	Protocolo basado en PUB-SUB. Su principal objetivo es recoger datos y transportarlos en la infraestructura de IoT	Ahorra energía y Memoria. Bajo Consumo de energía. Usa poco ancho de banda .	Conexión TCP de larga duración. Nombre de asunto son cadenas largas.	Domótica, aplicaciones de nivel empresarial..
AMQP	Cola de mensajes e interoperable.	Diseñado para soportar gran variedad de mensajería y patrones de comunicación.	Altamente fiable Garantiza la entrega de los mensajes.	Solamente trabaja en mayores anchos de banda	Mensajería de negocios y en la industria bancaria.
COAP	Solicitud de respuesta síncrona, comunicación 1 – 1 o M-M	Utilizado en dispositivos electrónicos para comunicarse. interactivamente a través de una red.	Comunicación M-M, Descubrimiento de recursos.	Estándar inferior comparado al estándar MQTT	Los hogares inteligentes, redes inteligentes, automatizaciones de construcción.
DDS	Es un protocolo de publicación/ suscripción en tiempo real. Proporciona el intercambio de mensajes simultáneos en un intervalo de tiempo.	Es centrados en datos utilizados en sistemas distribuidos altamente dinámicos	Es escalable. Su paradigma es enfocado a los datos y no a los mensajes. Tiene un amplio conjunto de políticas de QoS.	DDS no se ha utilizado ampliamente.	Sectores industriales como en las redes de ferrocarril. Control de tráfico aéreo. Energía inteligente. Servicios médicos.

4.3 Software Libre

En esta sección se analizará algunas opciones de Software Open-Source aplicado a la automatización de hogares, es de bajo costo y flexible ya que puede ser fácilmente ser modificado y adaptado a las necesidades de cualquier proyecto.

Debido al avance tecnológico actual en esta área de la IoT, hay que saber diferenciar, comparar la necesidad y la finalidad del uso del sistema antes de buscar una alternativa debido a que en la actualidad existen un gran número de propuestas diferentes con un funcionamiento parecido.

A continuación, se realizará un estudio de algunas alternativas más utilizadas según la revisión de literatura realizada.

4.3.1 Jeedom

Jeedom es un software de código abierto especializado para gestionar la automatización de su hogar. Se compone de una parte central (llamada núcleo) que se encarga de las funciones básicas que son:

- **Escenario.**

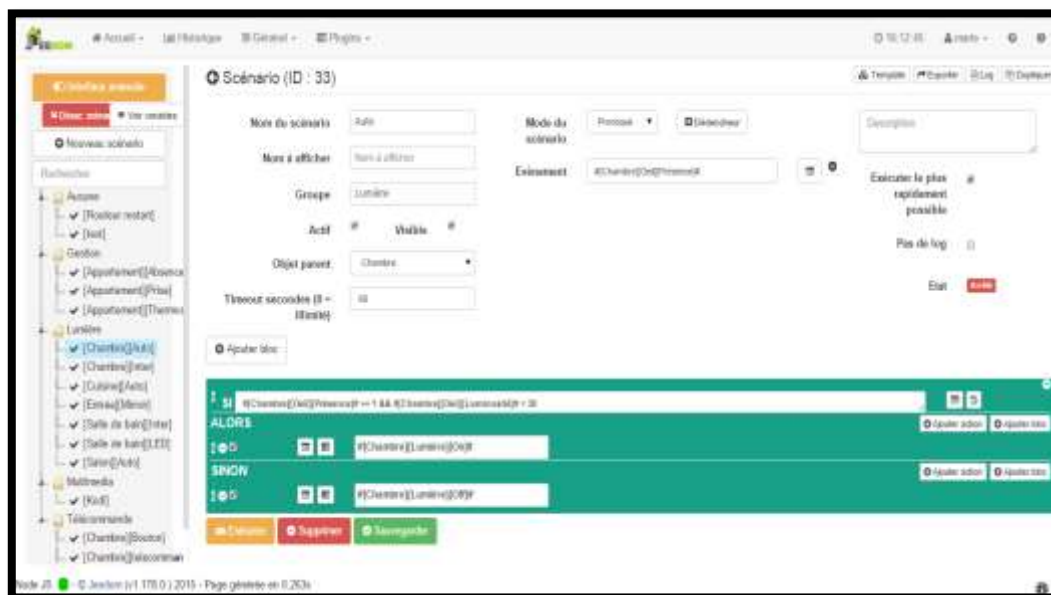


Figura 11: Interfaz de Escenario de Jeedom [16].

- **Historial**

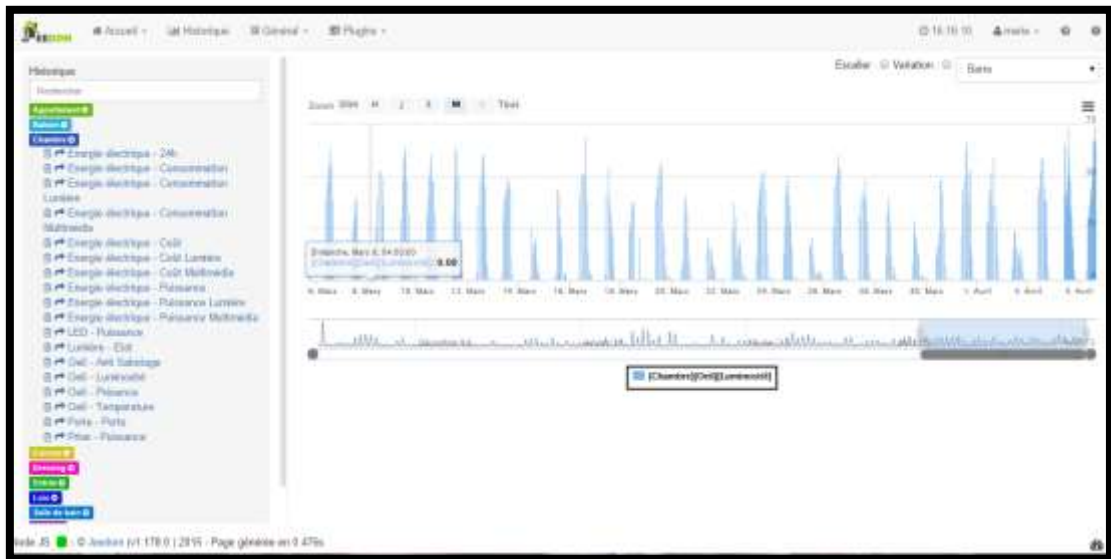


Figura 12: Interfaz del Historial de Jeedom [16].

- **Visualización**



Figura 13: Interfaz de la Visualización de Jeedom [16].

- **Iteraciones**

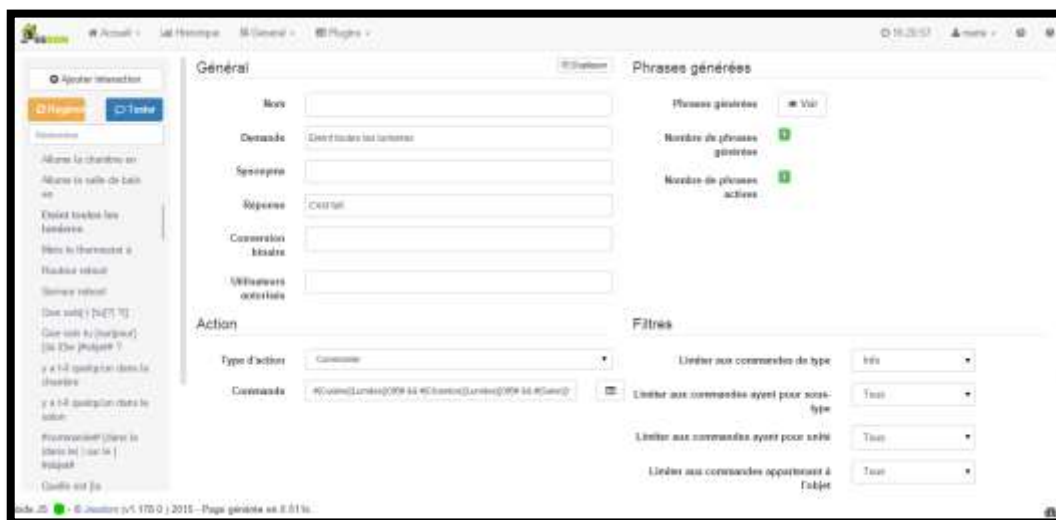


Figura 14: Interfaz de las Iteraciones de Jeedom [16].

- **Actualización y Respaldo.**

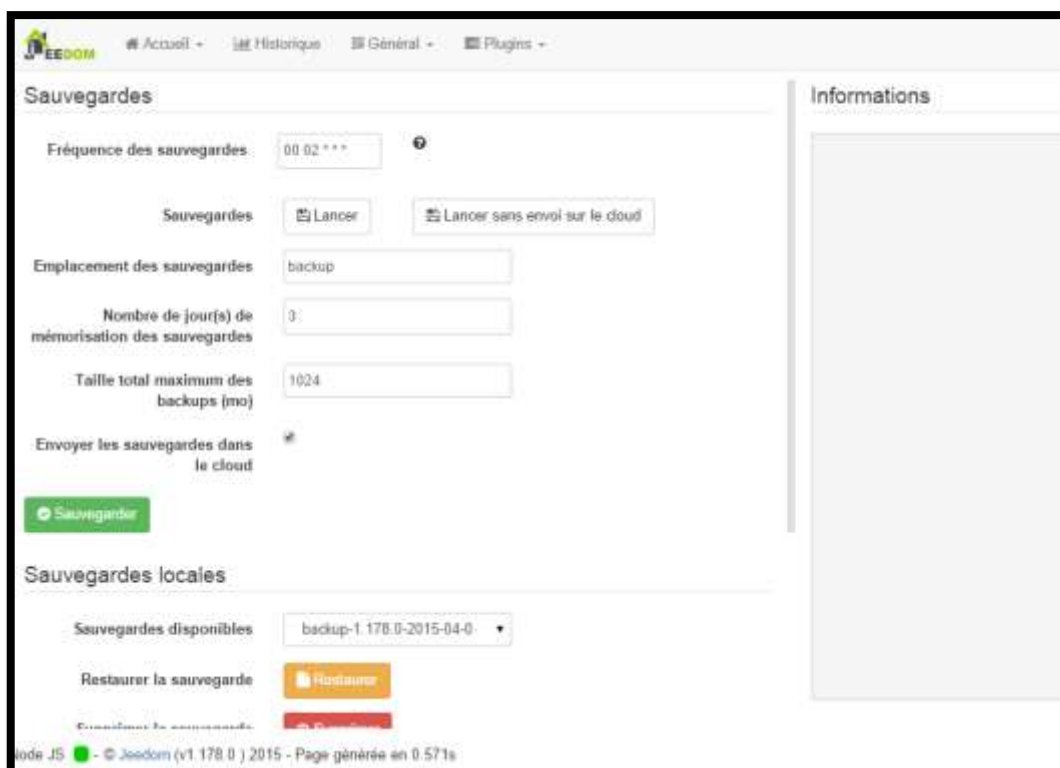


Figura 15: Interfaz de Actualizaciones y Respaldos de Jeedom [16]

En este núcleo se pueden agregar varios complementos a través de la Tienda (Ver Figura 16) de Jeedom tales como:



Figura 16: Interfaz de la Tienda de Jeedom [16].

- Protocolo de automatización del hogar (Z-Wave, RFXcom, EnOcean, etc).
- Protocolo IP (KNX, Xpl, etc).
- Objeto comunicante (Nest, Netatmo, etc).
- Nivel alto (alarma, termostato, etc).
- Interfaz (widget).
- Organización (agenda, Google agenda).
- Desarrollo (Escena).

Jeedom es una alternativa bastante completa, la cual ofrece muchas posibilidades como la gestión de la seguridad de personas y bienes. Gestión de electricidad, climatización etc. Su comunicación la realiza mediante voz, mensajes de texto, aplicaciones móviles tanto para dispositivos “Android” e “iOS” [6].

Además, cuenta con cuatro características importantes que son:

- Open Source:** Es de código abierto de tal manera que garantiza su transparencia en los procesos. Además, cualquier persona con conocimientos de programación puede ayudar a la mejora del mismo [6][16][17].
- Multi-Protocolo:** Jeedom es compatible con muchos protocolos como Z-Wave, EnOcean, KNX, Legrand Bus, RFXcom, RTS, Chacon, Edisio, MQTT etc. El

sistema de complementos, a través de Market Jeedom, garantiza la compatibilidad con muchos protocolos actuales y futuros [6][16][17].

- c) **Autónomo:** Jeedom no requiere acceso a servidores externos para funcionar. Toda su instalación se gestiona localmente y, por lo tanto, usted es el único que tiene acceso para garantizar la total confidencialidad [6][16][17].
- d) **Personalizable:** Gracias a su flexibilidad y a los muchos parámetros de personalización, cada usuario puede crear su propia domótica Jeedom. Con la ayuda de widgets, vistas y diseño, tiene total libertad para crear su propia interfaz[6][16][17].

4.3.2 Domoticz

Es un sistema de código abierto diseño de sistema de domótica para controlar varios dispositivos y recibir información de varios sensores.

Por ejemplo, este sistema se puede usar con:

- a) Interruptores de luz.
- b) Sensores de puerta.
- c) Timbres.
- d) Dispositivos de seguridad.
- e) Sensores meteorológicos como: medidores UV, lluvia y viento.
- f) Sensores de temperatura.
- g) Medidores de pulso.
- h) Medidores de voltaje.

Este sistema está diseñado para operar en varios sistemas operativos, cuenta con una interfaz gráfica hecha en html5 escalable y que se adapta automáticamente para dispositivos móviles y de escritorio compatible con todos los navegadores web.

Domoticz es un sistema de automatización de casas con licencia GPL v3 desarrollado en C++, cuenta con una interfaz web soportando muchos protocolos como el MQTT. Además, cuenta con una gama de Plugins que ayudan a adoptarse a las necesidades del cliente [6][17][18].

4.3.3 Home Assistant.

Es otra alternativa Open-Source desarrollada en Python 3 con licencia Apache 2.0 perfecto para ejecutarse en una Raspberry pi o un servidor local que tiene como objetivo el control local y la privacidad. Para conseguir la automatización de la vivienda establece una serie de reglas ante posibles eventos ocasionales o programados.

Al igual que la mayoría de alternativas de código abierto, posee manuales paso a paso de instalación, configuración y utilización.

Home Assistant permite controlar todos sus dispositivos sin almacenar ninguno de sus datos en la nube. Así mismo rastrea y controla todos los dispositivos en el hogar y automatiza el control.

Algunas características principales con las que cuenta este sistema son:

- a) Es gratis y de código abierto.
- b) Optimizado para dispositivos integrados como Raspberry pi.
- c) Domótica 100% local
- d) Fácil instalación y actualizaciones
- e) Interfaz web de gestión integrada.
- f) Cree y restaure copias de seguridad completas de toda la configuración con facilidad [19][17][6].

4.3.4 Node-RED

Es una herramienta de programación visual para conectar hardware, API y servicios de formas nuevas y fáciles. Node-RED proporciona un editor basado en un navegador que facilita la conexión de flujos mediante un sinnúmero de nodos con los que cuenta la paleta y que se pueden implementar con tan solo un clic, comenzó su vida a principios de 2013 como un proyecto paralelo de Nick O'Leary y Dave Conway-Jones del grupo de servicios de tecnología emergente de IBM[13].

Lo que comenzó como una prueba de concepto para visualizar y manipular mapeos entre temas MQTT, se convirtió rápidamente en una herramienta mucho más general que se podía extender fácilmente en cualquier dirección. Sus características principales son:

- **Edición de flujo basado en un navegador:** Como ya se mencionó Node-Red proporciona un editor de flujos mediante un sinnúmero de nodos disponibles en la paleta y que se los puede implantar con tan solo un clic. Las funciones de java script se pueden crear dentro mediante la utilización de un editor de texto enriquecido. También incorpora una biblioteca que permite guardar funciones, plantillas o flujos útiles para la reutilización.
- **Construido en Node.js:** Node-RED es basado en el entorno de desarrollo Node.js es por eso que el tiempo de ejecución es liviano de tal manera que aprovecha al máximo su modelado sin bloqueo controlado por eventos. Calidad que lo hace ideal para ejecutarse en hardware de bajo costo como la Raspberry pi o en la nube. Cuenta con más de 225000 módulos en el repositorio de paquetes de node.js y es muy fácil extender dicho rango de nodos para agregar nuevas funcionalidades y capacidades.
- **Desarrollo social:** Los flujos que se crean en esta herramienta se almacenan utilizando un formato JSON de tal manera que se pueden importar y exportar fácilmente para compartir con otros. Además, cuenta con una biblioteca en línea que le permite compartir sus mejores flujos con el mundo [20].

4.4 Hardware Libre.

El hardware Libre al igual que el Software libre ofrece al usuario cuatro libertades: Libertado de uso, de estudio y modificación, de distribución, y de redistribución de las mejoras. Sin embargo, su empleo no es tan directo ya que compartir diseños en hardware es más complicado. No hay definición exacta, pero se puede definirla mediante la clasificación según la naturaleza que son dos tipos:

Hardware Estático: Conjunto de elementos materiales de los sistemas electrónicos.

Hardware Reconfigurable: Es aquel que se describe mediante un lenguaje HDL (Hardware Description Language, Lenguaje de descripción hardware) y que permite especificar todo detalle de su estructura y funcionalidad [21].

En este punto se analizará tipos de hardware reconfigurable que sirven para implementar o desarrollar un sistema Domótico.

4.4.1 Raspberry Pi

La Raspberry Pi es un ordenador del tamaño de una tarjeta de crédito. Ordenador de placa reducida (SBC) o placa única (2006). Producto de la Fundación Raspberry Pi (UK), organización caritativa. Su administrador Eben Upton se puso en contacto con profesores, académicos y entusiastas para crear un ordenador que animara a los niños y niñas a aprender informática. Soportado por el Laboratorio de Computadores de la Universidad de Cambridge y la empresa Broadcom[22][23].

Operativo sin ningún problema, ser un centro multimedia para el hogar o una VPN entre más cosas. Es capaz de hacer muchas y cosas además es muy barato, que está disponible en cualquier lado. Con la ayuda de esto mini ordenadores las personas de todo el mundo puedan aprender, crear, innovar, investigar y resolver problemas a la vez que se divierten [6].

4.4.1.1 Raspberry Pi 3 Modelo B+

La Raspberry pi 3 Modelo B+ cuenta con muchas características como una velocidad de procesamiento excelente además se encuentra una infinidad de sistemas operativos disponibles, cada uno de ellos se dividen con base en sus aplicaciones, características y especificaciones.

- **Características.**

Tabla II. Características de Raspberry Pi 3 Modelo B+

Procesador	Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC
Frecuencia De Reloj	1,4 GHz
Memoria	1GB LPDDR2 SDRAM
Conectividad Inalámbrica	2.4GHz / 5GHz IEEE 802.11.b/g/n/ac Bluetooth 4.2, BLE
Conectividad De Red	Gigabit Ethernet over USB 2.0 (300 Mbps de máximo teórico)
Puertos	40 pines GPIO HDMI 4 x USB 2.0 CSI (cámara Raspberry Pi) DSI (pantalla tácil) Toma auriculares y vídeo compuesto. Micro SD y USB (alimentación) Power-over-Ethernet (PoE)
Fecha De Lanzamiento	14/3/2018
Precio Aproximado	\$ 68

4.4.2 Arduino.

Arduino es un microcontrolador de 8 bits de Atmel Open Source, esto significa que el usuario puede estudiar y realizar cambios en el hardware. Las placas Arduino pueden leer señales en las entradas, procesarlas y enviar a las salidas. Para que la placa trabaje es necesario enviar un conjunto de instrucciones mediante un lenguaje de programación Arduino y el software Arduino (IDE).

Arduino nació en el Instituto de Diseño de Interacción Ivrea como una herramienta fácil para la creación rápida de prototipos, dirigida a estudiantes sin experiencia en electrónica y programación. Tan pronto como llegó a una comunidad más amplia esta comenzó a cambiar para poder adaptarse a nuevas necesidades y desafíos siendo así una buena oferta para aplicaciones IoT. Arduino se ejecuta en Mac, Windows, Linux y son microcontroladores de bajo costo siendo de gran ventaja para maestros, estudiantes y

aficionados que están interesados en desarrollo de la robótica y comienzo de la programación.

Con los años, Arduino ha sido el cerebro de muchos proyectos desde objetos cotidianos hasta objetos científicos complejos. Todas las placas Arduino son completamente de código abierto, lo que permite a los usuarios construirlas de forma independiente y eventualmente adaptarlas a sus necesidades particulares [24].

Algunas características relevantes de Arduino son:

- **Barato:** Arduino son microcontroladores sumamente baratos en comparación a otros microcontroladores.
- **Multiplataforma:** El software de Arduino (IDE) se ejecuta en la mayoría de sistemas operativos como Windows, Linux y Mac Os.
- **Entorno de programación simple y claro:** El IDE de Arduino es fácil de usar para principiantes y flexible como para que usuarios avanzados también aprovechen las ventajas de estos microcontroladores.
- **Software de código abierto y extensible:** Arduino publica herramientas de código abierto a través de las bibliotecas de C++ de tal manera que se puede ampliar y las personas que deseen pueden comprender detalles técnicos de estas herramientas pueden hacerlo mediante el lenguaje de programación AVR C en la que está basada.
- **Hardware de código abierto y extensible:** Arduino se publica bajo la licencia Creative Commons, por lo que los diseñadores de circuitos experimentados pueden hacer su propia versión del módulo, extenderlo y mejorarlo.

4.4.2.1 ESP8266.

El módulo Wi-Fi ESP8266 es autónomo con una pila de protocolos TCP / IP integrada que puede brindar acceso a su red WiFi (o el dispositivo puede actuar como un punto de acceso)[24][13]. Satisface las diferentes necesidades de los usuarios en implementaciones de eficiencia energética, diseños compactos y altamente relacionados con IoT.

Una característica presente en todas las versiones de este módulo es la posibilidad de conectarle diferentes sensores externos u otros dispositivos mediante los pines GPIOs.

Además, este chip se presenta mediante diferentes módulos ESP8266, los cuales poseen algunas diferencias como la distribución número de pines GPIOs, etc., [6].

- **NodeMCU.**

NODEMCU es una placa de desarrollo de bajo costo que consolida pines GPIO, I2C, UART, PWM, ADC y Wi-Fi. Con tecnología de suministro de 3.3V y ESP8266 junto con un regulador de tensión y USB en serie[25].

Las características más relevantes de este kit de desarrollo son que es abierto, interactivo, programable, de bajo costo, simple, inteligente e incorpora la tecnología Wi-Fi. Con él se permite control de hardware mediante una API avanzada, consiguiendo una reducción del trabajo de configuración y manipulación hardware mediante lenguaje Arduino [6].

Existen dos versiones de NodeMCU la v0.9 y la v1.0, a continuación, analizaremos la versión más actual.

- **NodeMCU V1.**

Presenta una actualización del chip ESP8266 utilizando de un ESP-12 a un ESP-12E. Tiene grandes capacidades de procesamiento y almacenamiento que permiten integrar sensores y dispositivos a través de sus pines GPIO. Esta versión incorpora el chip CH340 para la conversión USB- Serial.

- **Características:**

- Código abierto, Interactivo, Programable, De Bajo Costo, Sencillo e Inteligente.
- Compatible con Arduino.
- USB
- 11 GPIO, cada GPIO puede ser PWM, I2C, 1 cable.
- Módulo Wi-Fi certificado FCC
- Antena PCB.

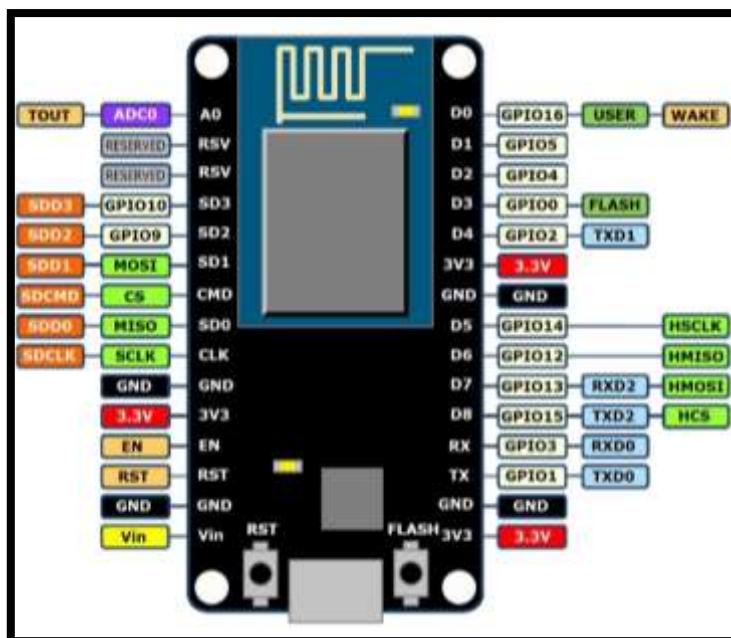


Figura 17: NODEMCU v1.0 [6]

4.4.3 Adafruit.

Adafruit fue fundada en 2005 por el hacker e ingeniero del MIT, Limor "Ladyada" Fried. Su objetivo era crear el mejor lugar en línea para aprender electrónica y crear los productos mejor diseñados para fabricantes de todas las edades y niveles de habilidad [26].

4.4.3.1 Adafruit HUZZAH ESP8266 BREAKOUT.

Es un microcontrolador WiFi del tamaño de un bocado a un precio considerable. El procesador ESP8266 de ESPRESSIF es un controlador de 80MHz con un front-end Wi-Fi completo con una pila TCP/IP con soporte DNS. Es muy fácil de usar y cuenta con un regulador a bordo de 500Ma, 3.3 V o cambio de nivel. Cuenta con una antena integrada y muchos pines [27].

- **Características:**
 - Salida de 3.3 v y regulador de 500mA
 - Dos entradas de alimentación protegidas con diodos
 - 64 KIB de RAM de instrucción
 - 96KiB de RAM de Datos.
 - 4 mb DE QIO Flash
 - Una entrada analógica

- 9 pines GPIO
- 2 pines UART
- 2 entradas de alimentación de 3-6 V
- Reinicio, habilitación, desactivación de LDO.

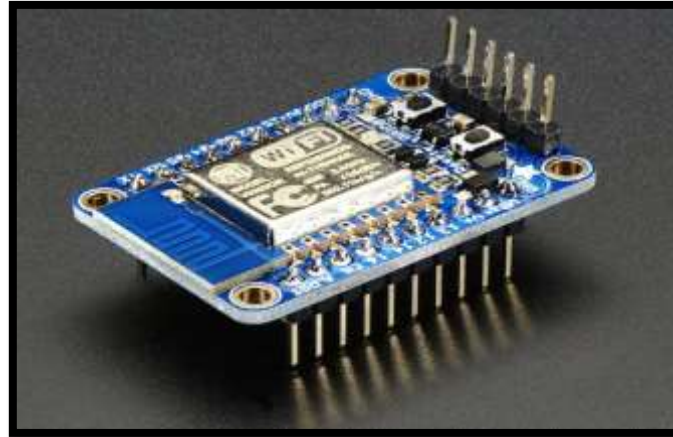


Figura 18: ADAFRUIT HUZZAN ESP8266 BREAKOUT [27].

4.5 Tecnologías y Herramientas utilizadas.

4.5.1 IDE de desarrollo Arduino.

Arduino es una compañía de hardware y software Open Source. Esto se refiere al usuario que diseña y utiliza tarjetas de desarrollo basados en microcontroladores. Dichas tarjetas de desarrollo se las conoce como módulos Arduino que son plataformas de código abierto.

El enfoque de programación de estas tarjetas se lo hace con el IDE de Arduino, que utiliza el lenguaje de programación C de tal manera que da acceso a una gran biblioteca que está constantemente en crecimiento gracias a la comunidad de código abierto.

Este IDE Arduino cuenta con una ventana en blanco donde se puede codificar inmediatamente. En primer lugar, se debe configurar los ajustes de la tarjeta y el puerto que permita cargar el código. Además, se puede utilizar otros lenguajes de programación y aplicaciones populares en Arduino como Java, Python Matlab, Perl, Visual Basic, etc.

El entorno de desarrollo de Arduino es sencillo e intuitivo, además, puede descargarse desde la página oficial de Arduino gratuitamente en este caso la versión 1.8.8. que es la más actual se utilizará para la codificación del microcontrolador [36].

4.5.2 Fritzing

Fritzing es una iniciativa de hardware de código abierto que permite que la electrónica sea accesible de manera creativa para cualquier persona. Ofrece a los usuarios documentar, diseñar y fabricar placas de circuitos impresos profesionales.

Fritzing se instala con una biblioteca de piezas y se puede agregar más si se desea. En Fritzing las partes se organizan en contenedores a los que se puede acceder desde la paleta del lado derecho con tan solo arrastrarla al área del boceto para después usarla. Además, ofrece la creación de piezas que se necesita, así como también compartirlas [37]. En el presente trabajo se utilizará esta herramienta para el diseño de los circuitos para integrar los sensores y actuadores a la placa NodeMCU.

4.5.3 MobaXterm

Es un software con muchas herramientas indispensables para la informática remota. Proporciona muchas funciones que se adaptan a los programadores, web masters, administradores de tecnologías de la información y prácticamente a todos los usuarios que necesitan manejar sus trabajos remotos de una manera más simple. MobaXterm proporciona herramientas utilizadas para la red remota (SSH, X11, RDP, VNC, FTP, MOSH, etc.) y, comando Unix (bash, ls, cat, sed, grep, awk, rsync, etc) para el escritorio Windows. Existen muchas ventajas al tener una aplicación de red con todas estas herramientas, por ejemplo, acceder por medio de SSH para conectarse a un servidor remoto y automáticamente aparecerá un navegador SFTP para editar directamente los archivos alojados en dicho servidor. Así mismo, todas las aplicaciones remotas se mostraran sin problema alguno en su escritorio de Windows[38]. Por todas las características expuestas se utilizará esta herramienta para acceder a un servidor remoto por medio de SSH que se adquirió para montar nuestra aplicación y poder acceder a ella desde cualquier parte del mundo.

4.6 Trabajos Relacionados.

A continuación, se describen estudios relacionados con el presente trabajo. Estos trabajos sirven de ayuda para afianzar que herramientas, materiales y software son los más comunes, confiables y que operan bajo recursos limitados dentro de la industria de la IoT. En el estudio [17] manifiesta que debido a la mejora de los protocolos inalámbricos, el desarrollo de servicios en la nube y el menor costo del hardware en los últimos años ha

sido esencial para una nueva era de las casas inteligentes. Por eso en este artículo se estudia los diferentes Protocolos de comunicación para verificar cual es el más efectivo para automatizar un hogar. Una vez terminado el estudio se concluyó que con la utilización de protocolo MQTT se puede incluir a dispositivos con recursos limitados y que se encuentren en lugares muy remotos. Según el artículo [25] manifiesta que, durante el desarrollo de la industria en automatización y conectividad inalámbrica, todos los dispositivos dentro del hogar pueden estar conectados. mejorando el confort, la eficiencia energética, seguridad interior, ahorro de costes del hogar. En tal caso, El consumo de energía y el ancho de banda de la red se convierten en una de las principales preocupaciones. Por tal motivo se necesita un dispositivo de baja potencia que transmita mensajes a través de un protocolo menos detallado. Debido a la ubicua disponibilidad de Wifi, todos los aparatos dentro del hogar pueden ser conectados a través de una pasarela común. Este trabajo presenta una visión general de un protocolo ligero de mensaje de telemetría de colas, Protocolo de transporte (MQTT). En el prototipo, intenta implementar MQTT junto con un chip ESP8266. Los sensores y actuadores están conectados al ESP8266 y se ha establecido un agente MQTT basado en Mosquitto para monitorear y controlar de manera remota.

Este estudio [28] tiene como objetivo proporcionar un sistema inteligente que reduzca la carga de trabajo del personal que trabaja, contribuyendo con servicios adicionales, integración con el entorno del hogar mediante la contratación de un agente de la comunidad. El entorno doméstico consta de un controlador inteligente para el hogar, sensores, dispositivos que transfieren información con el fin de mejorar la seguridad. El extremo de la comunidad consiste en un servidor central, elemento como una computadora personal que tiene la capacidad de asociarse con los dispositivos que se encuentran en ubicaciones remotas. Los dispositivos de interfaz se emplean para evitar la confusión entre la funcionalidad y la interfaz de usuario. El documento apunta tanto a MQTT como a servicios HTTP. El MQTT desea implementar servicios en sistemas domésticos inteligentes. El HTTP gobierna la transferencia de datos basados en la ubicación.

En el artículo [29] se discute y crea un sistema domótico basado en MQTT, utilizando los sensores y usando el Raspberry pi modelo B + como la puerta de enlace de red, aquí se implementó el protocolo MQTT para la transferir y recibir datos del sensor y finalmente

acceder a esos datos de algunos sensores, también se implementa un ACL (lista de control de acceso) para proporcionar un método de encriptación para los datos y finalmente monitorear estos datos en una página web. El Raspberry pi se utiliza como una puerta de enlace o servidor principal de todo el sistema, que tiene varios sensores conectados a través de cables o inalámbricamente.

Según el artículo [30] se ha desarrollado un sistema revolucionario que compite por cambiar la arquitectura del software de la domótica actual existente en el mercado. Este sistema utiliza el protocolo MQTT y la programación visual Node-RED brindando así un sistema confiable y eficiente, además brinda una visualización para cambiar los requisitos del sistema controlando por completos las entradas, salidas de los sensores a través del servidor web y por último brindan notificaciones por correo electrónico o redes sociales, funciones las cuales se las puede acceder desde cualquier partes del mundo de forma remota o por medio de SSH, por lo tanto se presenta una automatización del hogar avanzada a través de Node-RED y MQTT.

El objetivo del documento [9] es diseñar un sistema de automatización para el hogar seguro al que se pueda acceder desde una posición global. En este sistema propuesto, se utiliza una Raspberry pi como puerta de enlace entre el panel de control web y los dispositivos reales del sistema, así como los sensores. Para un punto de vista de comunicación más rápido en este sistema, se utiliza el protocolo MQTT que accede a los datos y dispositivos de los sensores desde cualquier lugar del mundo. Los sensores y dispositivos se están comunicando a la Raspberry Pi a través de la conexión por cable.

En el estudio [6] propone poner al alcance de todas las personas la domótica sin importar su clase social por lo cual hace uso de hardware libre y de bajo costo, además hace uso de herramientas OpenSource o Código libre y gratuito que son de gran escalabilidad y compatibilidad con cualquier hardware. En este caso hace uso del microcontrolador Raspberry Pi, la cual tiene una gran variedad de funcionalidad y por tal motivo lo utilizará para montar el bróker MQTT que servirá como medio de comunicaciones entre sensores y el cliente. Cabe destacar que se utilizará un protocolo de comunicación bastante utilizado y ligero para IoT que permitirá la comunicación de los sensores, actuadores y el servidor siendo automáticamente y en tiempo real cada una de sus funcionalidades.

El propósito del estudio [31] es realizar un prototipo de una Smart Home utilizando el paradigma IoT y añadiendo otras nuevas que facilitan la interacción de los usuarios con los dispositivos, en este caso utilizando lenguaje natural desde los Smartphone, para esto utilizo la plataforma de IBM Node-RED que ayuda con la integración del hardware con el software, así mismo utilizo el protocolo MQTT para la comunicación de los datos entre estos dispositivos y por último implemento un chatbot en Telegram para la interacción del usuario de forma intuitiva.

En [13] nos manifiesta que la domótica representa el control remoto de electrodomésticos, enchufes eléctricos, calefacción, sistemas de refrigeración, de seguridad, etc. Y que con el aumento de todos estos dispositivos conectados nace un problema de procesamiento de datos por la cantidad masiva de los mismos es de aquí es que para vencer estos problemas de procesamiento de información hace la utilización del protocolo MQTT que es un protocolo ligero el cual ayuda significativamente a reducir este problema ya que opera bajo recursos limitados como el ancho de banda, procesamiento, etc. Además, hace uso de la plataforma IoT Node-RED que es muy eficiente para enlazar cualquier cantidad de dispositivos IoT y poderlos controlar y monitorear desde cualquier parte del mundo. Por tal motivo este trabajo se centra específicamente en crear un sistema eficaz de automatización del hogar utilizando hardware de bajo costo o libre que contengan Wi-Fi. Llegando a la conclusión que estas herramientas o sistemas serán el futuro de la innovación.

5. MATERIALES Y MÉTODOS

El trabajo de titulación (TT) se realizó en la carrera de Ingeniería en Sistemas, de la facultad de la Energía, Las Industrias y los Recursos Naturales no Renovables de la Universidad Nacional de Loja, se plantearon lineamientos necesarios para el desarrollo del mismo, conjuntamente con el docente de la materia de Trabajo de Titulación y el Director del trabajo. Para cumplir el objetivo general del TT que es "Desarrollar una solución informática que permita administrar un hogar inteligente haciendo uso de Hardware Libre" se desarrollaron actividades planteadas por cada objetivo específico. A continuación, se detallan los objetivos específicos con sus respectivas actividades.

- **Realizar una comparación técnica de los diferentes dispositivos de Hardware Libre y Software aplicado a Domótica.**
 - ✓ Revisión de Literatura acerca del tema planteado.
 - ✓ Estudio de bondades de hardware libre Raspberry PI, Arduino y Adafruit.
 - ✓ Estudio de los diferentes softwares Open-Source usados en la domótica.
- **Desarrollar el módulo de que permita la integración y administración de los componentes eléctricos y electrónicos del prototipo de la vivienda inteligente.**
 - ✓ Fabricación de la maqueta que simula la realidad de una vivienda.
 - ✓ Diseño del Circuito
 - ✓ Generación de Código de diferentes módulos
 - ✓ Acoplamiento de módulos para conformar sistema
 - ✓ Integración de los diferentes componentes eléctricos y electrónicos en la maqueta.
- **Desarrollar un prototipo para la solución informática que permita administrar la programación de eventos para el hogar inteligente:**
 - ✓ Definir Metodología
 - ✓ Documentar Requerimientos
 - ✓ Elaboración de Casos de Usos
 - ✓ Diseñar la Solución Informática
 - ✓ Desarrollo de la solución informática.
- **Probar y evaluar el funcionamiento del prototipo del sistema de control del hogar inteligente en un entorno simulado:**
 - ✓ Testeo de funcionamiento de la solución Informática

- ✓ Comprobación de que los sensores y actuadores funcionen adecuadamente.

Para el cumplimiento del primer objetivo se realizó una revisión de literatura sobre temas relacionados al TT, ayudando como apoyo y justificación del mismo. Para el desarrollo del trabajo se utilizó las siguientes herramientas Mosquitto, IDE de desarrollo Arduino, NodeRED, Fritzing. que nos ayudaron a cumplir el segundo y tercer objetivo. Los diferentes materiales, métodos y técnicas científicas utilizadas para cumplir los objetivos planteados y la metodología utilizada para el desarrollo y revisión de literatura se describen a continuación.

5.1 Materiales

En el desarrollo del presente TT se emplearon materiales y herramientas que permitieron llevar a cabo el desarrollo de los objetivos, en la TABLA III son presentados los materiales y herramientas utilizadas.

Tabla III. Materiales Usados

Hardware	
Elementos	Funciones
Computadora	<ul style="list-style-type: none">- Codificar los diferentes Sketches de Arduino que posteriormente son grabados en el chip ESP8266.- Acceso al servidor para el levantamiento del servicio.- Redacción del trabajo.
NODEMCU	<ul style="list-style-type: none">- Conexión de los sensores e intercambio de información entre los mismos y el cliente (dispone de un chip ESP8266 para la conexión WIFI).
Sensor PIR	<ul style="list-style-type: none">- Detectar la presencia física.
Sensor DHT11	<ul style="list-style-type: none">- Detectar la temperatura y humedad ambiente.
Servo-Motor	<ul style="list-style-type: none">- Abrir y cerrar las puertas de la maqueta.
Relés	<ul style="list-style-type: none">- Gestionar los ventiladores de la maqueta.
Led	<ul style="list-style-type: none">- Simular las luces de una vivienda real.

Driver Puente H L298N	- Girar los motores en sentido horario y anti horario.
Software	
<i>Software</i>	<i>Funciones</i>
MobaXterm	- Acceder mediante SSH al servidor montado en Ubuntu 18.04. y poder levantar el servicio donde está montada la aplicación.
Eclipse Mosquitto	- Broker MQTT para la comunicación entre cliente y sensores.
Node js	- Instalar la plataforma de programación grafica Node-Red.
Node-Red	- Desarrollo del cliente MQTT.
IDE de desarrollo Arduino	- Codificación de programas y subirlos a las diferentes placas que son compatibles con el mismo.
Fritzing	- Diseño las conexiones electrónicas de sensores y actuadores.
Varios	
	Funciones
Protoboard	- Conexión de los sensores con el microcontrolador NODEMCU.
Cables de Prototipado.	- Distribución de corriente entre sensores, actuadores y microcontroladores.
Maqueta	- Integra todos los circuitos, para la simulación del prototipo de hogar inteligente.
Internet	- Búsqueda de información de apoyo para la elaboración del presente TT.
Trasporte	- Movilización a lugares que permitieron terminar el TT.
Soldadura	- Conexión estable de los sensores, actuadores al microcontrolador.

Cables USB	- Enviar el código y proporcionar energía al microcontrolador.
UPS	- Suministra energía a los actuadores y sensores.

5.2 Métodos

5.2.1 Método Deductivo

Este método va de lo general a lo particular. Es decir, este método permitió determinar la estructura de la solución informática y los procesos que fueron necesarios para el cumplimiento de los objetivos del presente TT. Los pasos empleados en este trabajo son los siguientes:

- Establecer el tema a desarrollar.
- Definir los objetivos a realizar.
- Elaboración de la Revisión de literatura.
- Seleccionar las herramientas y materiales para el desarrollo del TT.
- Desarrollar la solución informática para nuestro TT.
- Evaluar los resultados y la solución informática.

5.2.2 Método Bibliográfico

La utilización de este método permitió recolectar la información necesaria de libros, blogs, tesis, artículos científicos, revistas y demás material relacionado con la tecnología, protocolos de comunicación y software para automatizar un hogar. También para determinar cuáles serán las mejores herramientas para el desarrollo del mismo. Los pasos que se siguieron para la obtención de la información fueron los siguientes:

- Los estudios para sustentar este TT deben tener como característica principal su año de publicación mayor al año 2014.
- Para la recolección de información se seleccionó como fuente de búsqueda a Google Scholar, IEEE, Scopus y Scielo, debido a que son repositorios de información confiables y permite obtenerlos en orden de importancia sin limitaciones de idioma. Además, nos permite una búsqueda avanzada con palabras claves acerca del tema a investigar.

- Se eliminó los estudios duplicados y se procedió a la selección de los estudios revisando todo el documento y eligiendo que estudio tiene mejor resultados.

5.2.3 Método Analítico

Este método ayudo a revisar de forma separada y ordenada toda la recolección de información sobre herramientas utilizadas para el desarrollo de sistemas domóticos. Los pasos empleados para este proceso fueron:

- Primeramente, se seleccionó las herramientas más utilizadas para el desarrollo de sistemas domóticos basándonos en los casos de éxito. Luego se procedió a detallar cada una de las características de las mismas.
- Luego, se hizo un análisis de las características encontradas de estas herramientas con otras existentes en el mercado para después seleccionar las herramientas más idóneas que ayudaron al desarrollo del presente trabajo de acuerdo a los objetivos planteados.

Este método también se lo utilizo para realizar el análisis de los resultados obtenido al aplicar la encuesta para evaluar el presente sistema de tal manera que se pueda concluir si tiene buena acogida por parte de los usuarios.

5.3 Técnicas.

5.3.1 Observación

Esta técnica se la utilizo para realizar la evaluación de los resultados de cada objetivo planteado garantizando así que la solución propuesta sea confiable, segura y de bajo costo en relación a otras soluciones informáticas existentes en el mercado.

5.3.2 Entrevista.

Con el uso de esta técnica se logró definir y afianzar el uso herramientas, materiales y requerimientos con los cuales se va a desarrollar la solución informática.

5.3.3 Encuesta.

Esta técnica fue de gran ayuda para la realización del presente TT, debido a que permitió obtener información que ayudo a ver si es viable o no el presente trabajo así mismo ayudo

a fijar los requerimientos con los que va a contar la aplicación. Además, se la utilizo para evaluar el prototipo del sistema propuesto de tal manera que el usuario exprese el nivel de satisfacción al experimentar como funciona un sistema domótico.

5.4 Metodología.

Para el desarrollo del presente trabajo se utilizó como base la metodología propuesta por A. Pérez [32], para el desarrollo de hardware y software pretendiendo dar solución a problemas reales donde resulta ineludible la iteración de ambos. La presente metodología está apoyada en el ciclo de desarrollo clásico en V, que contiene un sinnúmero de técnicas y métodos propuestos para mejorar la confiabilidad en el proceso su ciclo de vida. Una de las características más relevantes de la metodología es que llevara invertir más tiempo en la etapa de diseño, fase en la que los rediseños son más baratos y los cambios más efectivos de tal manera que se reduce en coste final del producto. En consecuencia, la materialización del producto final será más rápida y cumplirá con todos los requisitos propuestos. A continuación, se describe cada una de las fases de la metodología empleada:

- Definición de especificaciones: Esta fase consistió primeramente en definir y documentar los diferentes requisitos funcionales y no funcionales con los que contara el sistema, seguidamente se procedió a realizar un análisis tanto de hardware como software libre usado para el desarrollo de un sistema domótico para proceder a elegir la mejor opción entre todas las alternativas existentes en el mercado. Asimismo, se realizó una comparativa de los beneficios que presentan los diferentes protocolos de comunicación a nivel de la capa de aplicación para IoT para que nos facilite el intercambio de mensajes entre los sensores, actuadores y el servidor.
- En el Diseño Global del sistema se procedió a realizar un diseño general del mismo mediante diagramas UML. Para el diseño general del sistema utilizaremos en diagrama de casos de uso en el cual se detalla las iteraciones entre el usuario y el sistema. Además, se utilizará los diagramas de secuencia que permitirá visualizar los pasos que el usuario deberá hacer para interactuar con el usuario.
- En la fase de Diseño En Detalle se especificará la arquitectura de la propuesta, así mismo se hará uso de los diagramas de estado y clases, de tal forma que se tenga clara la arquitectura cada uno de los bloques descritos en la fase anterior y facilite el entendimiento del funcionamiento de la misma. También, se procede al diseño

6. RESULTADOS

En esta sección se presentan los resultados obtenidos del proceso mostrado en la Figura 19, donde se detallan las actividades de la metodología utilizada para el cumplimiento de los objetivos del presente Trabajo de Titulación.

6.1 Realizar una comparación técnica de los diferentes dispositivos de Hardware Libre y Software aplicado a Domótica.

En este apartado se expondrá de la manera más detallada posible, los requisitos fundamentales con los que debe cumplir el sistema (Ver Anexo 1), para así poder satisfacer las posteriores etapas de diseño e implementación.

A continuación, se detallan los requisitos funcionales y no funcionales con los que debe cumplir la solución informática.

6.1.1 Requisitos Funcionales.

En la Tabla IV, se presentan los requerimientos funcionales del sistema domótico, los cuales son base para el desarrollo del mismo:

Tabla IV. Descripción de Requerimientos Funcionales

CODIGO	DESCRIPCIÓN	CATEGORIA
RF001	Autenticación de Usuarios	Evidente
RF002	Cierre de Sesión	Evidente
RF003	Control de Cantidad de Usuarios que acceden al sistema al mismo tiempo.	Evidente
RF004	Control de luces.	Evidente
RF005	Control de Ventiladores	Evidente
RF006	Apertura y Cierre de Puertas	Evidente
RF007	Lectura de Temperatura	Evidente
RF008	Lectura de Humedad	Evidente
RF009	Gestión de seguridad	Evidente

6.1.2 Requisitos No Funcionales

A continuación, se detallan los requerimientos no funcionales que son de gran importancia para el sistema:

Tabla V. Descripción de Requerimientos No Funcionales

CODIGO	DESCRIPCIÓN
RNF001	Protocolo
RNF002	Funcionamiento
RNF003	Disponibilidad
RNF004	Usabilidad.

6.1.3 Software Libre.

De acuerdo a la revisión de literatura y los trabajos relacionados (Ver Sección 6 de la Revisión de Literatura) se ha encontrado cuatro alternativas que cumplen con los requisitos establecidos en el presente trabajo de titulación, mismos que son: JEEDOM, DOMETICZ, HOME ASSISTANT y Node-RED. Con la información encontrada se constata que todos son confiables, comerciales y sobre todo son de fácil configuración e implementación con diferente hardware existente, asimismo, existe gran cantidad de documentación y foros que ayudan a resolver cualquier inquietud y problemas que se presenten en el sistema.

A continuación, se analizará la herramienta HOME ASSISTANT y Node-RED que son los más utilizados para posterior al análisis seleccionar la mejor opción para el desarrollo del presente trabajo. Para el análisis se procedió a la instalación y configuración de ambos sistemas de tal manera que se pueda comprobar cuáles son sus puntos débiles y fuertes de cada uno de estos sistemas con respecto a los siguientes puntos presentados.

- **Instalación:** En cuanto a la instalación la herramienta Node-RED fue mucho más fácil en consideración a Home Assistant; no dio problema alguno, en cambio Home Assistant mantuvo errores en cuanto a la versión de Python por lo que con una versión anterior a la 3.5.1 o con la nueva versión 3.8.2 el sistema dio errores a la hora de instalar.

- **Interfaz UI:** Home Assistant permite un mayor orden con respecto a Node-RED. Aunque Node-RED permite mayor libertad de diseño.
- **Integración de Protocolos:** A la hora de integrar un protocolo de comunicaciones Node-Red lo hace mucho más fácil con respecto a Home Assistant, ya que este basta con arrastrar el nodo desde la paleta y configurarlo, mientras que Home Assistant hay que configurarlo en el archivo `configuration.yaml` y después integrarlo.
- **Escalabilidad:** Node-Red permite una mayor escalabilidad con respecto a Home Assistant debido a que es una herramienta para IoT y no un sistema dedicado a la automatización del hogar como Home Assistant. Node-Red permite integrar una infinidad de funcionalidades más. incluso se puede integrar el mismo Home Assistant en el. Por otro lado, Home Assistant activa comandos basados en configuraciones de usuario basados en el comportamiento anterior.

Por lo expuesto anteriormente se evidencia las ventajas que tiene Node-RED ya que es más fácil de crear y modificar flujos en cambio Home Assistant carece de las características que requieren las automatizaciones más complejas. Podría terminar con múltiples Scripts, Automatizaciones, etc.; tratando de hacer una automatización compleja que podría hacer en una sola aplicación simple como Node-Red.

6.1.4 Protocolo de comunicación para la solución Informática.

Uno de los requisitos del presente trabajo es utilizar un protocolo de comunicación M2M. Para seleccionar este protocolo nos basamos en el estudio hecho por la Fundación Eclipse que según la Figura 3 nos muestra que hasta el año 2018 los protocolos de comunicación más utilizados para IoT son MQTT y COAP. A continuación (Ver Tabla VI), se detallara sus características más importantes de estos protocolos para seleccionar el más idóneo.

Tabla VI. Características de los Protocolos MQTT y COAP

CARACTERISTICAS	MQTT	COAP
Protocolo Base	TCP	UDP
Modelo De Comunicación	Publicación- Suscripción	Pregunta- Respuesta
Consumo de energía	Mayor que COAP	Menor que MQTT

Tamaño de Cabecera	2 bytes	4 bytes
Tipos de Mensajes	Asíncronos	Asíncronos y síncronos
Fiabilidad	3 niveles de QoS	Mensajes confirmables, no confirmables, retransmisiones.
Seguridad	SSL	DTLS
Otro	Útil para conexiones en localizaciones remotas.	Baja latencia, baja saturación pero con problemas en NAT.
Implementación	Fácil de implementar pero complejo para añadir extensiones	Pocas librerías existentes y soporte.

Una vez identificadas las ventajas y desventajas de estos protocolos se optan por utilizar MQTT debido a que es un protocolo reconocido, de fácil implementación y con alta fiabilidad en conexiones remotas, características mismas con las que no cuenta el protocolo COAP.

6.1.5 Broker MQTT

Una vez definido la utilización del protocolo MQTT se procederá a buscar un bróker MQTT que nos permita recibir mensajes por parte del cliente, los procese y envíe los mensajes a los suscritores de ese Tópico. Asimismo, que gestione la red de tal manera que cuando el cliente envíe mensajes al bróker constantemente este reciba una respuesta inmediata del mismo.

Dentro del mercado podemos encontrar una infinidad de alternativas entre las principales tenemos al bróker Mosquitto, Mosca, HiveMQ, ActiveMQ, VerneMQ y JoramMQ. Para determinar la mejor opción se va a analizar dos de los más utilizados en los Trabajos Relacionados (Ver Sección 6 de la Revisión de Literatura); entre ellos tenemos al bróker Mosquitto y Mosca. A continuación, se detallan las características más relevantes de estos.

Mosca: En si es un corredor de Node.js MQTT que se puede usar como servicio independiente o incrustado en otra aplicación Node.js. Su popularidad en GitHub es de 3.1 K de estrellas.

Mosquitto: Es un agente de mensajes MQTT, es liviano y adecuado para el uso de todos los dispositivos desde la computadora personal hasta servidores completos. Su popularidad se evidencia en GitHub con un total de 3.5 K de estrellas.

Por lo expuesto anteriormente ambos agentes proporcionan una comunicación mediante el protocolo MQTT por lo cual para el presente proyecto vamos a seleccionar al mismo por la popularidad alcanzada en el repositorio GitHub. Debido a que refleja la satisfacción de otras personas al utilizar dicho agente. Entonces, para el desarrollo del presente TT se utilizará el Bróker Mosquitto a continuación se lo detallará.

6.1.5.1 Eclipse Mosquitto

El Broker Mosquitto es un intermediario de mensajes de código abierto (con licencia BSD) que implementa el protocolo de transporte de telemetría MQTT versión 3.1. MQTT proporciona un método ligero para llevar a cabo la mensajería utilizando un modelo de publicación / suscripción. Este lo hace adecuado para mensajes de "máquina a máquina", sensores de baja potencia o dispositivos móviles; como teléfonos, computadoras integradas o microcontroladores como el Arduino [34][35]. Su servidor escucha en los siguientes puertos:

- 1883: MQTT, sin cifrar.
- 8883: MQTT, encriptado.
- 8884: MQTT, cifrado, requiere certificado de cliente.
- 8080: MQTT sobre WebSockets, sin cifrar.
- 8081: MQTT sobre WebSockets, encriptado.

Los puertos cifrados son compatibles con TLS v1.2, v1.1 o v1.0 con certificados x509 y requieren soporte de cliente para conectarse.

Debido a que este bróker es de código abierto se puede descargar e instalar en diferentes sistemas operativos como Windows, Linux, Mac OS, etc. Los equipos que vayan a contener el bróker Mosquitto deben permanecer activos las 24 horas del día, ya que puede

existir una publicación y suscripción por parte de los clientes en cualquier momento. Por ejemplo, se puede instalar en un Router debido a que este permanece encendido todo el tiempo o en cualquier dispositivo dependiendo del proyecto [35].

6.1.6 Hardware Libre a Utilizar.

En este apartado, se realizará una comparación técnica de dos microcontroladores que son los más usados para proyectos IoT, son de bajo coste y sobre todo que cumplen con los requisitos establecidos por el presente TT. Estos son el microcontrolador NODEMCU y la Raspberry Pi 3. Sus características más relevantes son:

NodeMCU ESP8266: Este microcontrolador es similar al Arduino, aunque más potente debido a que cuenta con un wifi incorporado, su programación se la hace mediante el lenguaje de programación Arduino y permite construir casi cualquier proyecto desde electrodomésticos inteligentes hasta robots autónomos. Además, su costo aproximado es de diez dólares americanos.

Raspberry PI 3: Este minicomputador es más potente y completo con respecto al NodeMCU, además cuenta con muchas más funcionalidades como Wi-Fi, bluetooth, entradas GPIO, entrada de sonido, video, etc.; tiene mucha popularidad hoy en día ya que ejecuta casi cualquier cosa, desde máquinas expendedoras hasta paneles de automóviles. Al igual que el NodeMCU se puede construir cualquier cosa teniendo mucho más control sobre la forma en que se hace las cosas y su valor aproximado es de setenta dólares americanos.

Una vez expuestas sus características más relevantes se ha optado por seleccionar al microcontrolador NodeMCU ya que se depende de algunos factores entre ellos tenemos el costo y la comodidad para trabajar. Debido a que el sistema propuesto va a ser un prototipo y este microcontrolador brinda casi los mismos beneficios que la Raspberry Pi.

6.2 Desarrollar el módulo de que permita la integración y administración de los componentes eléctricos y electrónicos del prototipo de la vivienda inteligente.

En esta sección primeramente se procede al diseño e implantación del sistema domótico a nivel de hardware.

6.2.1 Diseño de Alto Nivel

A continuación, se procederá al diseño del sistema de forma general de tal forma que se pueda tener una visión de cómo va a funcionar. Para ello se hará uso de los diagramas UML.

6.2.1.1 Casos de Uso.

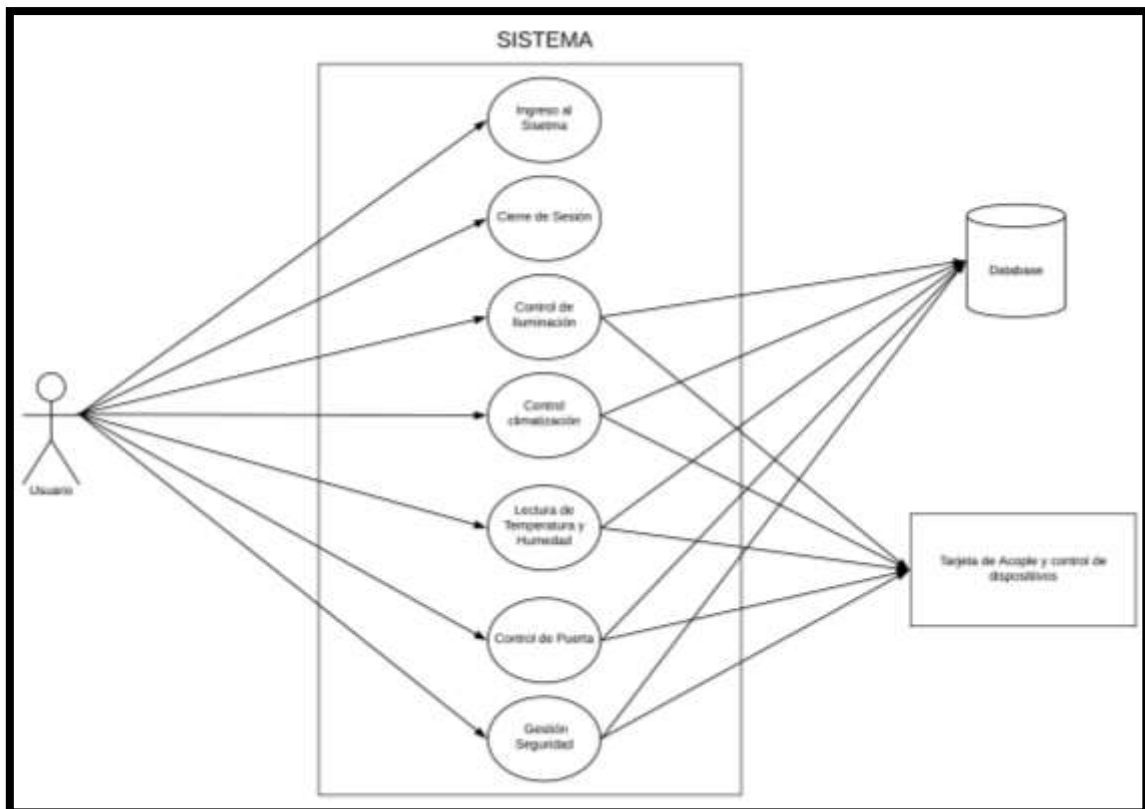


Figura 20: Diagrama de Casos de Uso

6.2.1.2 Especificación de casos de Uso

A continuación, se hace la descripción de cada caso (Ver Figura 20) de uso del Sistema Domótico.

Tabla VII. Descripción de Casos de Uso N°1

CU-01	Ingreso al Sistema
Requerimiento Relacionado.	RF001- RF003
Objetivo en Contexto	El usuario debe autenticarse para acceder al sistema.
Precondición	-
Final Exitoso	Usuario Autenticado
Final Fallido	Usuario Rechazado
Actores Principales	Usuario
Actores secundarios	-
Evento de Inicio	El usuario ingresa al sistema para autenticarse.
Flujo Principal.	<ol style="list-style-type: none"> 1. Al cargar el sistema presenta formulario de autenticación. 2. El usuario deberá digitar su usuario y contraseña. 3. Al presionar el botón "Aceptar" el sistema verificara la autenticidad de los datos ingresados. Si son válidos se inicia sesión. El inicio de sesión implica desplegar el menú principal de control del hogar inteligente. 4. Al presionar el botón "Cancelar" finaliza el caso de uso.
Flujo alternativo.	<ol style="list-style-type: none"> 1. Si la identificación de usuario o la contraseña no son válidas, el sistema presentara cuadros de dialogo como "usuario no existe" o "Contraseña Incorrecta". 2. Si un usuario está en sesión activa otro usuario no podrá ingresar de tal manera que el sistema notificara que está en uso.

Tabla VIII. Descripción de Casos de Uso N°2

CU-02	Cierre de Sesión
Requerimiento Relacionado.	RF002
Objetivo en Contexto	Cerrar sesión Automáticamente.
Precondición	El usuario esta con la sesión activa.
Final Exitoso	Cierre de sesión

Final Fallido	Sesión activa
Actores Principales	Usuario
Actores secundarios	-
Evento de Inicio	El usuario esta con la sesión activa.
Flujo Principal.	<ol style="list-style-type: none"> 1. El usuario al iniciar sesión el sistema comenzará un temporizador de 5 minutos, tiempo en el cual el usuario podrá controlar el hogar inteligente. 2. Cuando falten 15 segundos para terminar la sesión el sistema notificara que está a punto de terminarse. 3. Al concluir el tiempo establecido el sistema cerrara la sesión automáticamente mostrando un cuadro der dialogo de “sesión expirada”.
Flujo alternativo.	<ol style="list-style-type: none"> 1. El usuario al iniciar sesión en el sistema y una vez que termine con la tarea propuesta pinche en el botón cerrar sesión. 2. El sistema cerrara la sesión mostrando el formulario de autenticación.

Tabla IX. Descripción de Casos de Uso N°3

CU-03	Control de Iluminación
Requerimiento Relacionado.	RF004
Objetivo en Contexto	Encender y Apagar luces
Precondición	Usuario Autenticado – microcontrolador conectado.
Final Exitoso	Encendido y Apagado de las Luces
Final Fallido	No se ejecuta acción de encendido y apagado de luces
Actores Principales	Usuario
Actores secundarios	Base de Datos – Microcontrolador
Evento de Inicio	El usuario se encuentra con la sesión activa
Flujo Principal.	<ol style="list-style-type: none"> 1. El usuario accede al sistema 2. El usuario se autentica 3. El usuario ejecuta acción de encender y apagar luces. Dichas acciones se guardaran en una base de datos en terminología booleana 0 “apagar”

	y 1 “encender” con la fecha y hora de dicha acción para posteriores estudios.
Flujo alternativo.	1. El microcontrolador se encuentra sin energía o internet entonces no se podrá subscribir al tema publicado por el sistema de tal forma que no se podrá tener un final exitoso del caso de uso.

Tabla X. Descripción de Casos de Uso N°4

CU-04	Control de Climatización
Requerimiento Relacionado.	RF005
Objetivo en Contexto	Encender y Apagar ventiladores
Precondición	Usuario Autenticado - Lectura de temperatura ambiente- microcontrolador conectado.
Final Exitoso	Encendido y Apagado de ventiladores
Final Fallido	No se ejecuta acción de encendido y apagado de la ventilación
Actores Principales	Usuario
Actores secundarios	Base de Datos – Microcontrolador
Evento de Inicio	El usuario se encuentra con la sesión activa
Flujo Principal.	<ol style="list-style-type: none"> 1. El usuario accede al sistema 2. El usuario se autentica 3. El usuario ejecuta acción de encender y apagar ventilación. Dichas acciones se guardaran en una base de datos en terminología booleana 0 “apagar” y 1 “encender” con la fecha y hora de dicha acción para posteriores estudios.
Flujo alternativo.	<ol style="list-style-type: none"> 1. Si la temperatura sobrepasa los 21°C entonces el sistema accionara los ventiladores automáticamente caso contrario los apagara. 2. El microcontrolador se encuentra sin energía o internet entonces no se podrá subscribir al tema publicado por el sistema de tal forma que no se podrá tener un final exitoso del caso de uso.

Tabla XI. Descripción de Casos de Uso N°5

CU-05	Lectura de temperatura y Humedad	
Requerimiento Relacionado.	RF007 –RF008	
Objetivo en Contexto	Leer la temperatura y Humedad	
Precondición	Usuario Autenticado – Microcontrolador conectado	
Final Exitoso	Poder obtener información de temperatura y humedad	
Final Fallido	No se puede obtener temperatura y humedad ambiente.	
Actores Principales	Usuario	
Actores secundarios	Base de Datos – Microcontrolador	
Evento de Inicio	El usuario se encuentra con la sesión activa	
Flujo Principal.	<ol style="list-style-type: none"> 1. El usuario acceder al sistema 2. El usuario se autentica 3. El usuario puede visualizar los valores obtenidos del sensor tanto de temperatura como de humedad. Dichos valores se guardan en una base de datos con fecha y hora de cada lectura que hace el sensor. 	
Flujo alternativo.	<ol style="list-style-type: none"> 1. El microcontrolador se encuentra sin energía o internet entonces no podrá publicar el tema para que el sistema pueda subscribirse entonces no se podrá tener un final exitoso del caso de uso. 	

Tabla XII. Descripción de Casos de Uso N°6

CU-06	Control de Puerta	
Requerimiento Relacionado.	RF006	
Objetivo en Contexto	Abrir y Cerrar Puerta	
Precondición	Usuario Autenticado – Microcontrolador conectado	
Final Exitoso	Apertura y cierre de puerta	
Final Fallido	No se puede abrir ni cerrar la puerta	
Actores Principales	Usuario	
Actores secundarios	Base de Datos – Microcontrolador	

Evento de Inicio	El usuario se encuentra con la sesión activa
Flujo Principal.	<ol style="list-style-type: none"> 1. El usuario acceder al sistema 2. El usuario se autentica 3. El usuario ejecuta acción de abrir o cerrar puerta. Dichas acciones se guardaran en una base de datos en terminología booleana 0 "cerrar" y 1 "abrir" con la fecha y hora de dicha acción para posteriores estudios.
Flujo alternativo.	<ol style="list-style-type: none"> 1. En caso de que la temperatura sobrepase los 26 °C el sistema accionara la apertura inmediata de la puerta para ventilación y evacuación del lugar. 2. El microcontrolador se encuentra sin energía o internet entonces no podrá publicar el tema para que el sistema pueda subscribirse entonces no se podrá tener un final exitoso del caso de uso.

Tabla XIII. Descripción de Casos de Uso N°7

CU-07	Control de Puerta
Requerimiento Relacionado.	RF009
Objetivo en Contexto	Asegurar el bienestar de las personas y bienes
Precondición	Microcontrolador conectado
Final Exitoso	Actividades del hogar actúan frente a emergencias.
Final Fallido	No ejecuta acciones automáticamente.
Actores Principales	Microcontrolador
Actores secundarios	Usuario
Evento de Inicio	El microcontrolador está recibiendo datos de los sensores.
Flujo Principal.	<ol style="list-style-type: none"> 1. El microcontrolador está conectado y con acceso a internet. 2. El microcontrolador recibe datos de los sensores tanto de temperatura como de presencia. En el caso de recibir un dato de presencia física por parte del sensor PIR este activara la alarma y enviara notificaciones al correo de dicha detección. Por otro lado si se recibe la lectura de temperatura superior a los 26°C este activara una alarma, encenderá luces por la ruta de evacuación, enviara notificación al correo de dicha situación y abrirá inmediatamente la puerta para evacuación y ventilación al tratarse de un incendio.

Flujo alternativo.

1. El microcontrolador se encuentra sin energía o internet entonces no podrá accionar los diferentes actuadores de tal manera que se tendrá un final fallido del caso de uso.

6.2.1.3 Diagrama de secuencia del Sistema.

En este apartado se detallará los casos de uso de tal manera que se pueda planificar y comprender la funcionalidad de cada módulo del sistema.

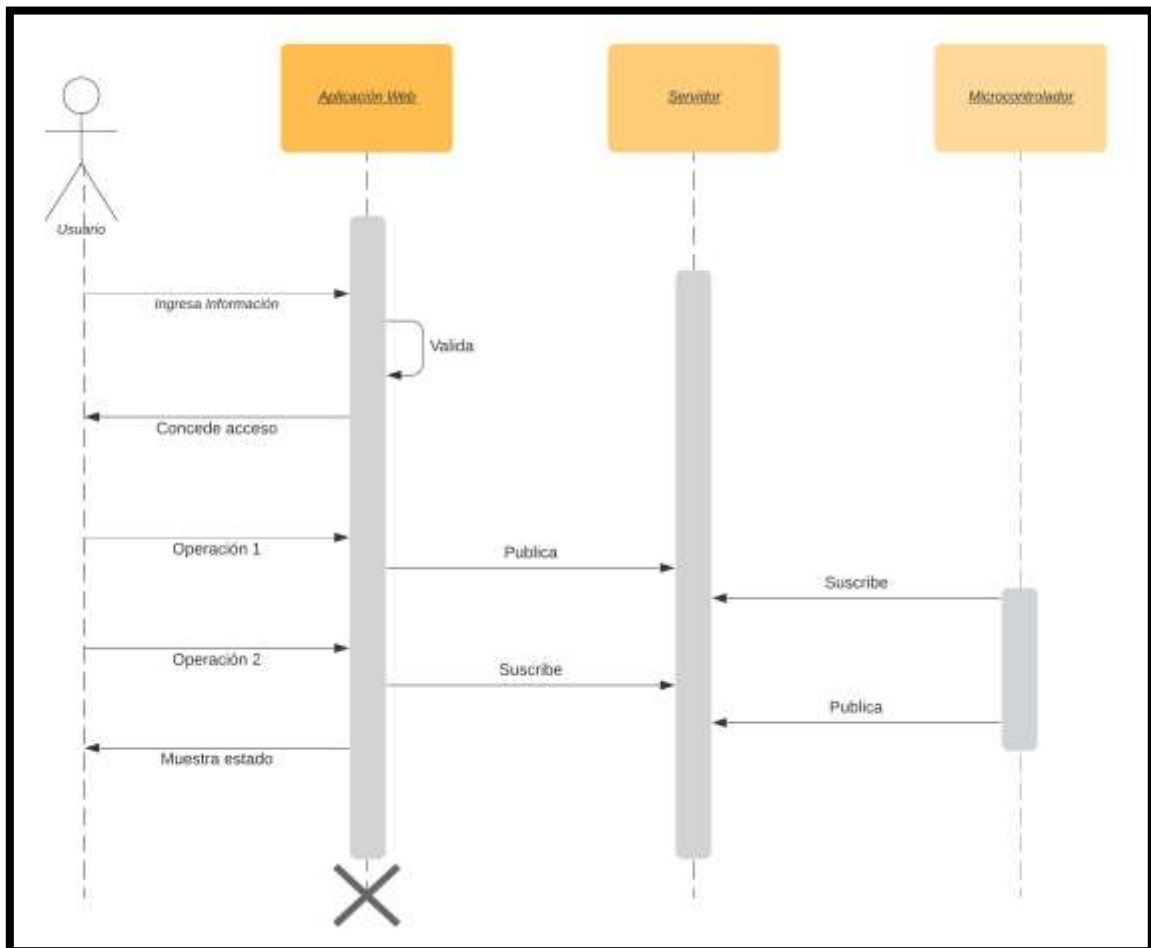


Figura 21: Diagrama de secuencia Autenticación

En la Figura 21. se muestra el proceso del módulo de autenticación del sistema domótico. Primeramente, el usuario ingresa la información requerida por la aplicación WEB y la envía, después la aplicación valida los datos ingresados por el usuario, si los datos están correctos devuelve al usuario la interfaz donde podrá realizar las diferentes operaciones que permite el sistema, caso contrario vuelve a pedir que ingrese los datos de autenticación.

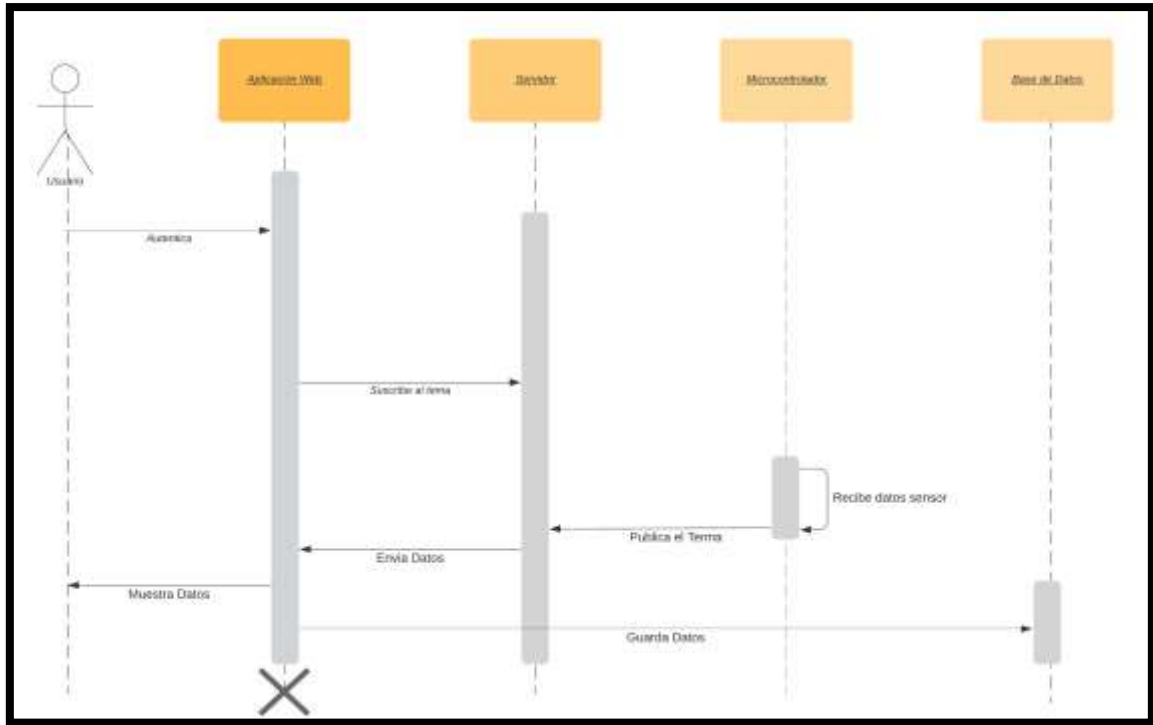


Figura 22: Diagrama de secuencia Sensores

En la Figura 22 se muestra la secuencia para obtener la lectura de datos de Temperatura y humedad que hace constantemente la aplicación Web al servidor de comunicaciones, luego el servidor se comunica con el microcontrolador que a la vez recibe los datos del sensor para posterior a esto reenviar dichos valores hasta la interfaz gráfica. De la misma forma cada dato recibido será guardado en una base de datos para estudios posteriores.

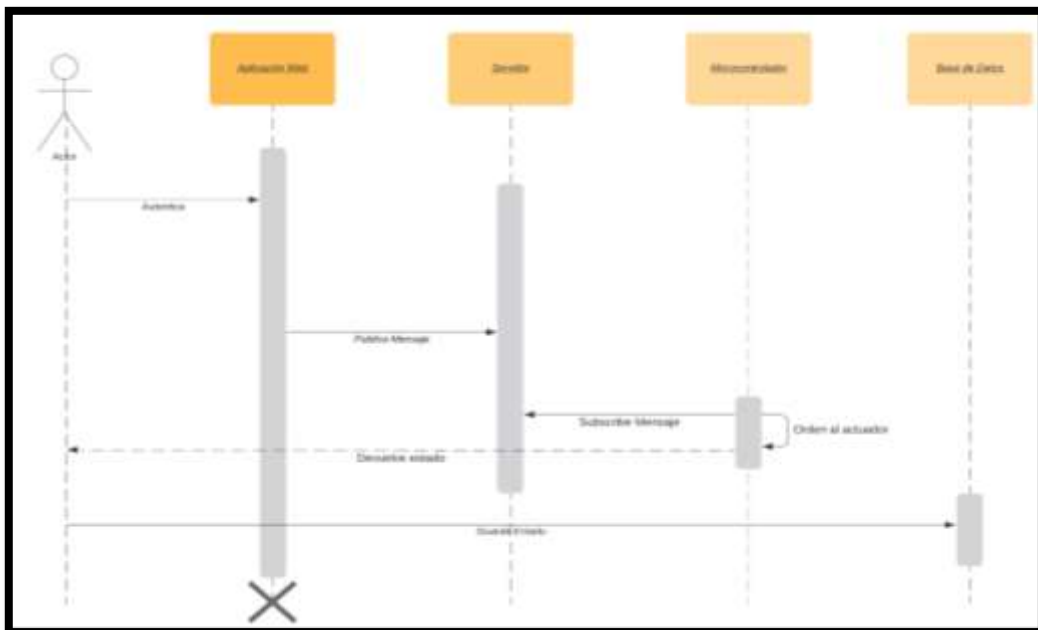


Figura 23: Diagrama de Secuencia de los Actuadores

En la Figura 23 se muestra el proceso que el usuario debe hacer para modificar el estado de un actuador por ejemplo en el sistema se podrá encender o apagar luces, abrir o cerrar puertas y encender o apagar ventiladores. Primero el usuario inicia sesión, después mediante la interfaz gráfica del sistema ordenara el cambio de estado de cualquier actuador, dicha instrucción se efectuará con la ayuda del servidor de comunicaciones que ordenara al microcontrolador el cambio del mismo. Por otro lado, cada instrucción que se realiza será guardada en una base de datos con fecha y hora de la acción realizada.

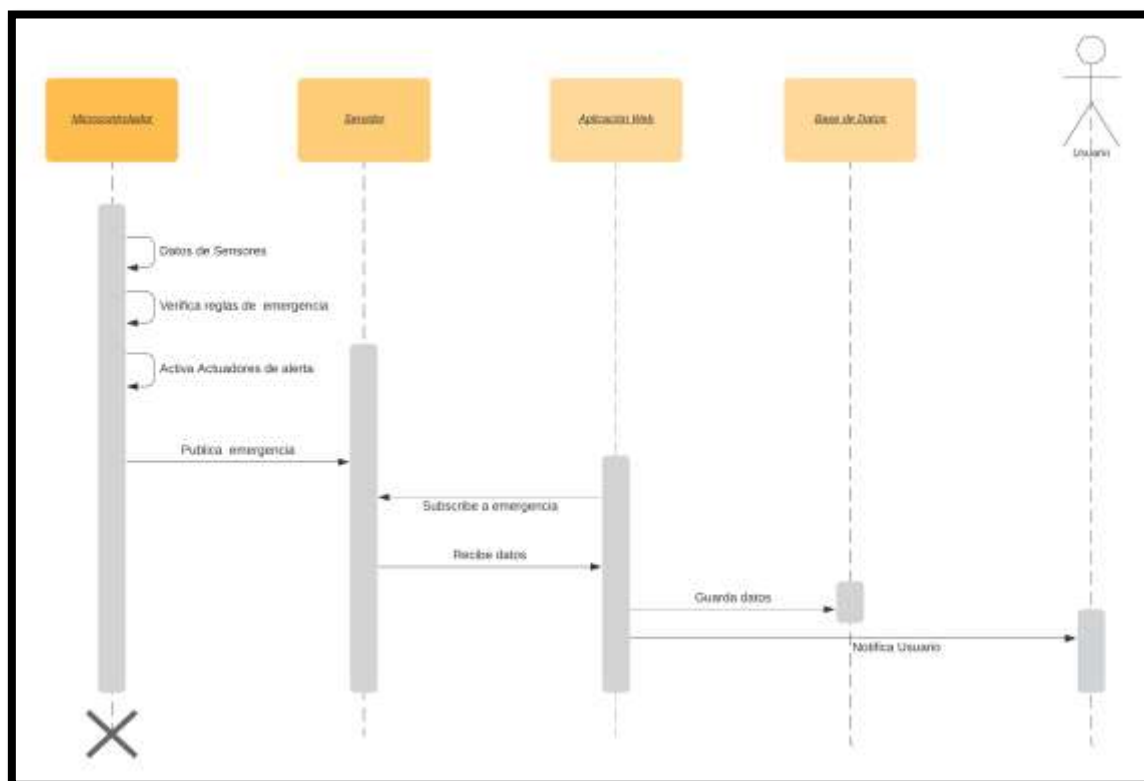


Figura 24. Diagrama de Secuencia de Gestión de Seguridad

En la Figura 24. se detalla el procedimiento que se efectúa para que el sistema actúe en caso de una emergencia. Primeramente, el microcontrolador está constantemente recibiendo datos por parte de los sensores; una vez que recibe un dato que este establecido en una regla de emergencia este activara los diferentes actuadores como la sirena, luces de evacuación y apertura de la puerta; que ayudaran a asegurar el bienestar personal y de bienes según sea el escenario de la emergencia. Posterior a esto publicara dicha emergencia a la interfaz gráfica de la aplicación web y notificara mediante correo electrónico a los usuarios.

6.2.2 Diseño en Detalle.

A continuación, lo que pretende es complementar lo ya alcanzado, aumentando la claridad y la comprensión de cómo va a funcionar el sistema domótico propuesto.

6.2.2.1 Arquitectura del sistema.

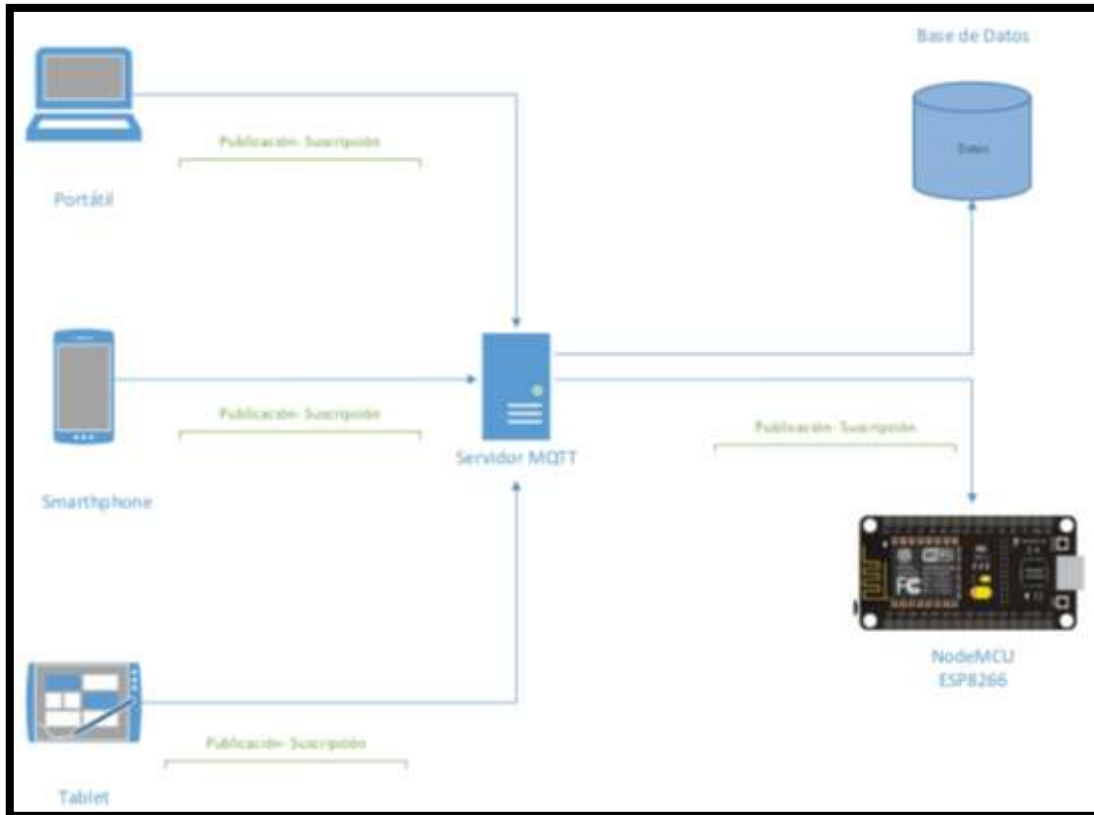


Figura 25. Arquitectura del Sistema

En la Figura 25. se muestra la arquitectura del sistema propuesto. Se compone de cuatro elementos que son: Un publicador que transmite información a entidades interesadas (Suscriptores), un suscriptor que se registra a temas especiales por parte del publicador, Un bróker MQTT el cual comprobaba que tanto el publicador como el suscriptor estén autorizados y por ultimo una base de datos en la cual se guardara toda la información generada por los publicadores y suscriptores.

6.2.2.2 Diagrama de clases.

En la Figura 26. se representa la estructura y el comportamiento de cada uno de los objetos del sistema y sus relaciones con los demás objetos con el fin de facilitar la comprensión de como interactúa el sistema con los diferentes sensores y actuadores.

"Desarrollo De Una Solución Informática Que Permita Administrar Un Hogar Inteligente Haciendo Uso De Hardware Libre"

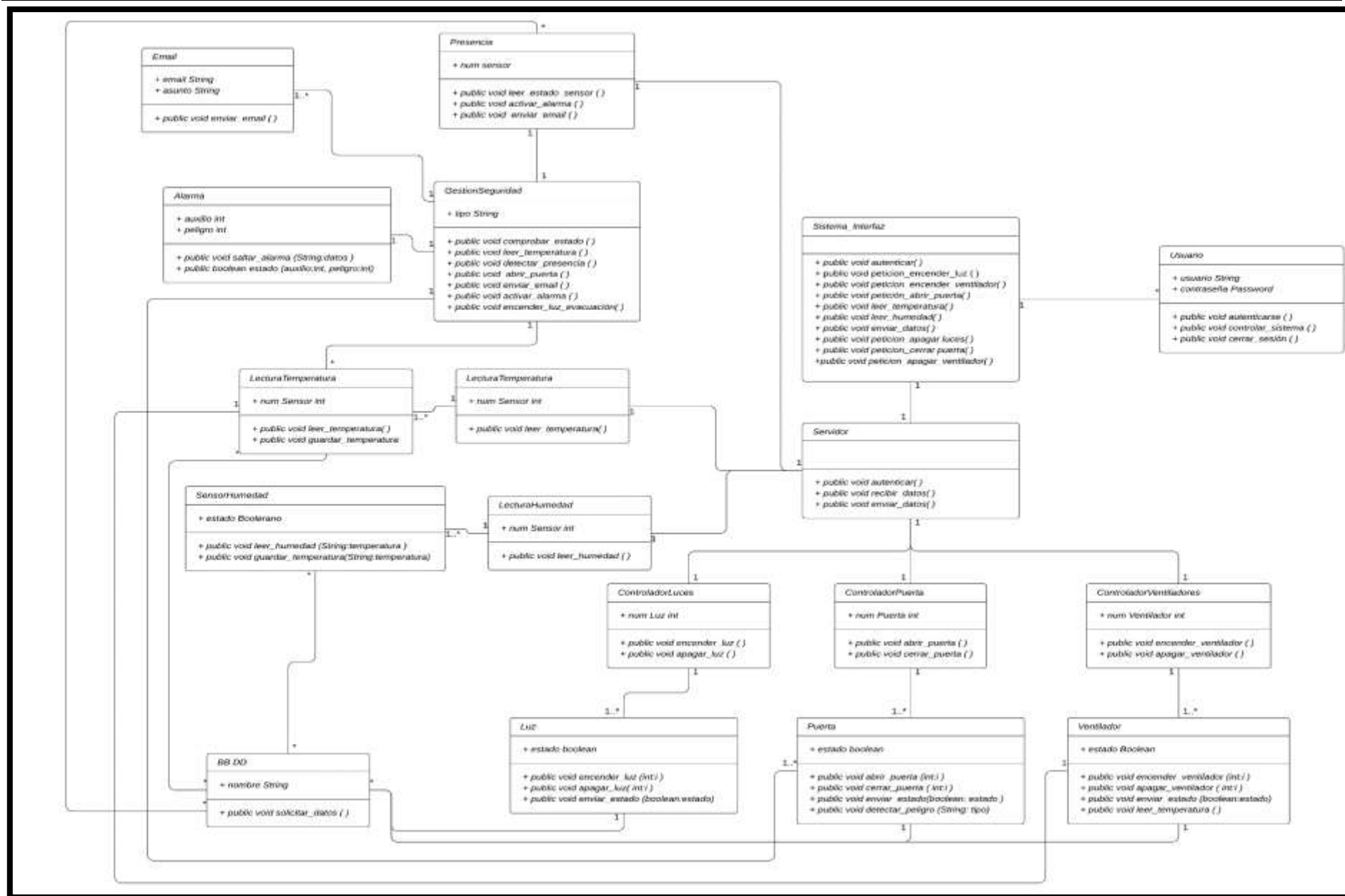


Figura 26. Diagrama de Clases

6.2.2.3 Diagrama de Flujo.

En la Figura 27. se muestra el flujo que se lleva a cabo para poder controlar los diversos actuadores tales como, luces, ventilación y puerta. Primeramente, el usuario debe autenticarse si sus credenciales son válidas accederá a la interfaz de control caso contrario volverá a pedir dichas credenciales. Así mismo, permite la lectura de la temperatura y humedad ambiente; cuando la lectura de temperatura sea mayor a 22 °C se activará automáticamente el ventilador caso contrario se desactivará.

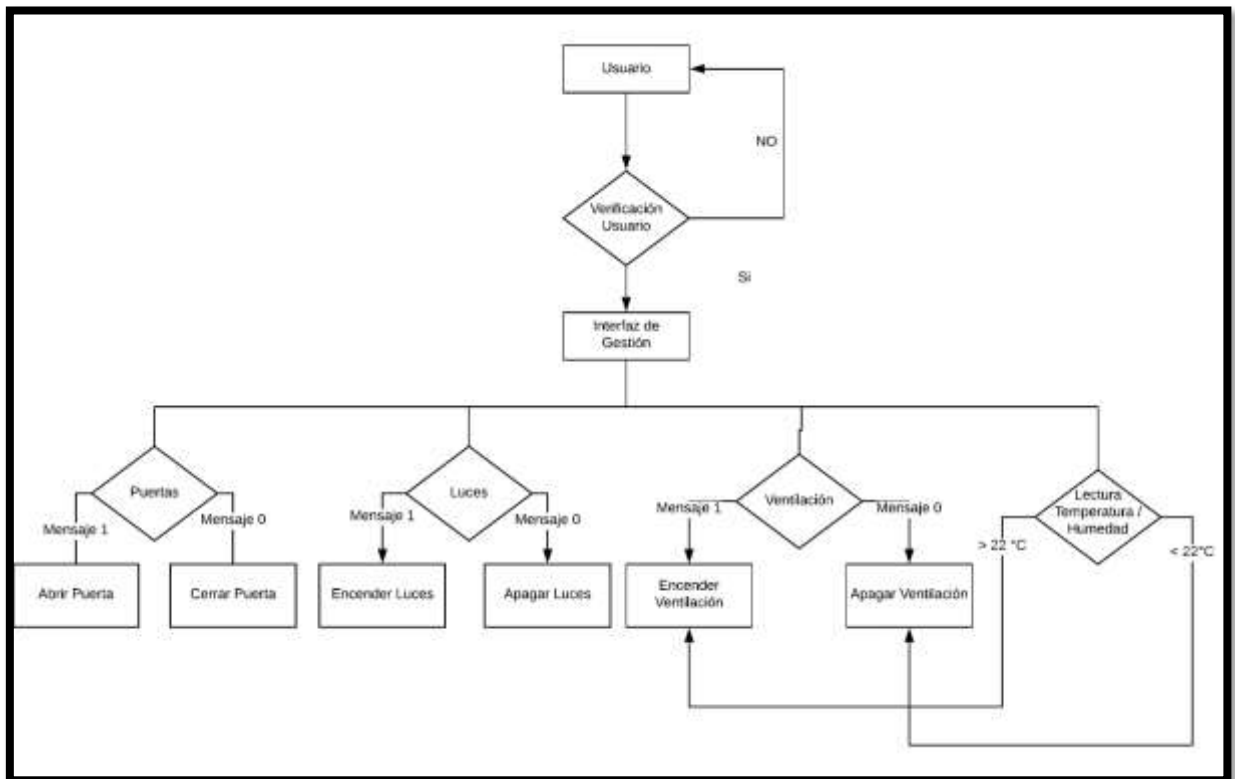


Figura 27. Diagrama de Flujo Actuadores

En la Figura 28. de detalla el flujo que se efectuara en el módulo de seguridad el cual accionara cuando los sensores tanto como de presencia como de temperatura reciban datos que representes peligro. Por ejemplo, en el caso del sensor de presencia cuando este detecte movimiento enviará un 1 lógico que disparará una alarma y enviará notificaciones al usuario de dicho evento. En el caso del sensor de temperatura cuando este obtenga una lectura por encima de los 26 °C lo tomará como principio de incendio por lo cual activará una alarma, encenderá luces por la ruta de evacuación, abrirá la puerta automáticamente para la evacuación y posterior a esto enviará notificaciones mediante correo electrónico de dicho incidente.

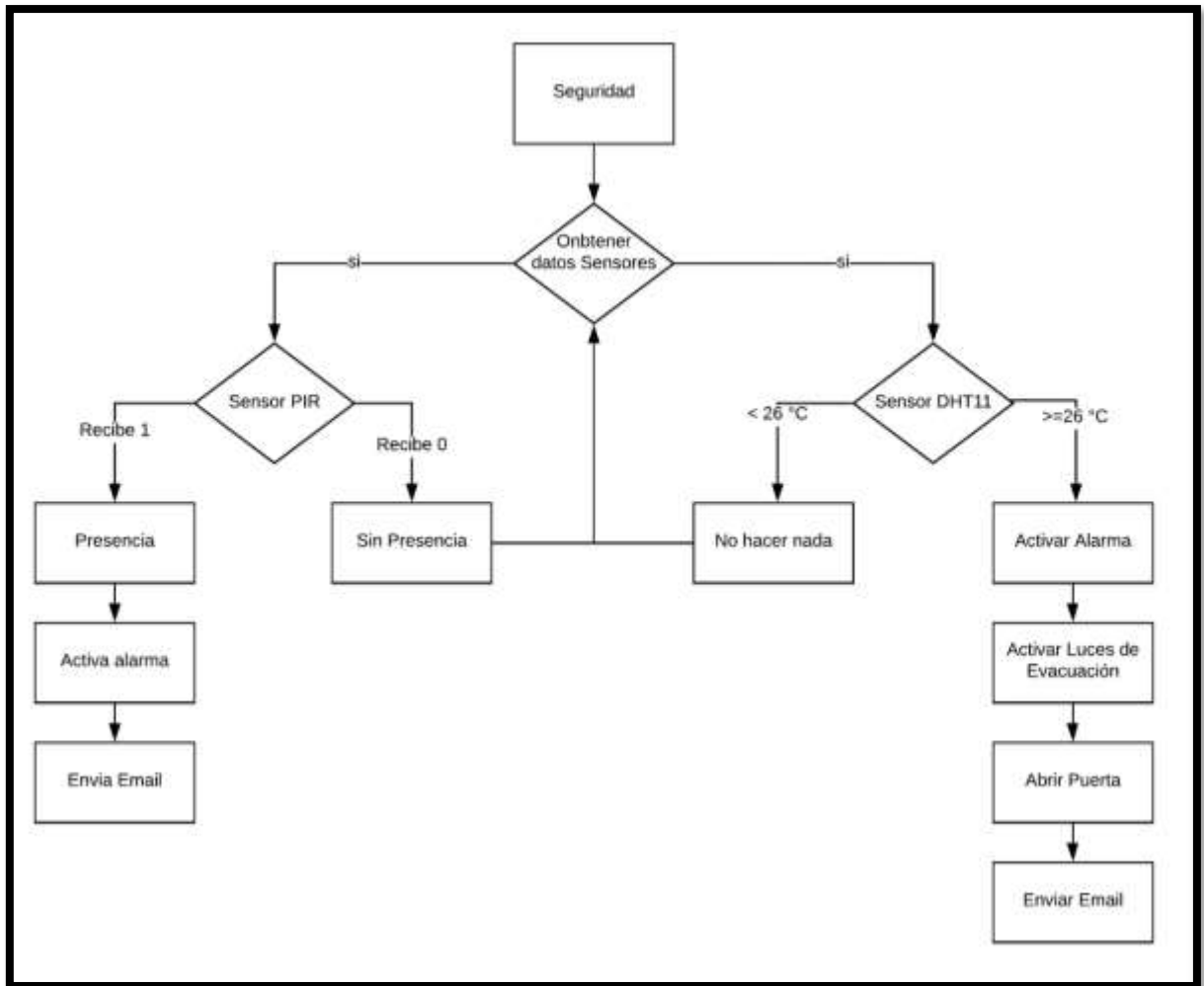


Figura 28. Diagrama de Flujo Sistema de Seguridad

6.2.3 Estructura de los Tópicos

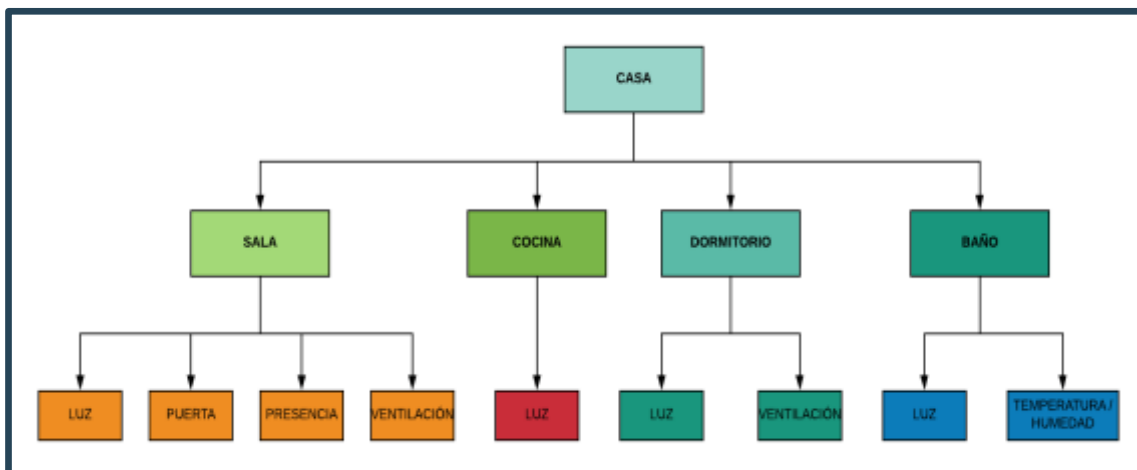


Figura 29: Estructura Jerárquica de los Tópicos que conformaran el Sistema

Como anteriormente se mencionó los tópicos dentro de MQTT se manejan de forma jerárquica y se separa cada nivel con el símbolo “/” de esta forma se crean jerarquías de clientes que publican y reciben datos. De esta forma un nodo puede subscribirse a un “Topic” concreto o a varios si es necesario. En la Figura 29. se muestra la jerarquía de los topics que contendrá nuestro sistema, como se puede observar el nodo padre sería CASA de cual se desprenden los nodos hijos SALA – COCINA – DORMITORIO Y BAÑO, a la vez estos tienen otros nodos hijos. A continuación, se detallan todos los topics que estarán presente en el sistema.

- casa/sala/luz
- casa/sala/puerta
- casa/sala/presencia
- casa/sala/ventilación
- casa/cocina/luz
- casa/dormitorio/luz
- casa/dormitorio/ventilación
- casa/baño/luz
- casa/baño/temperatura
- casa/baño/humedad

6.2.4 Diseño y Programación de los Sketches MQTT.

Primeramente, se identifica la cantidad de materiales necesarios que conformaran el sistema domótico en general. A continuación, se detallan (ver Tabla XIV).

Tabla XIV. Elementos que formaran parte del Sistema Domótico

Cantidad	Nombre	Descripción
2	NODEMCU ESP8266	Como unidad de control.
1	Sensor temperatura y humedad DHT11	Leer temperatura y accionar ventiladores.
4	Leds	Simulan la luz eléctrica.
1	Motor	Para apertura y cierre de puertas
2	Ventiladores	Mantener temperatura deseada.
1	Sirena	Como alarma ante la detección de intrusos
1	Sensor de Presencia	Como Detector de Intrusos
1	UPS	Como fuente de energía externa
2	Módulos de Relés dobles.	Para controlar los ventiladores
1	Modulo H L298N	Para cambiar el sentido del giro del motor.

Una vez detallados todos los materiales a utilizar se procede a la instalación y configuración del servidor MQTT y el lenguaje de desarrollo Arduino.

6.2.4.1 Instalación del bróker Mosquitto.

En esta sección primeramente se procede a la instalación y configuración del bróker MQTT (Ver Anexo 2) en el servidor virtual contratado (Ver Figura 30) que permitirá la comunicación desde cualquier parte del mundo.

Luego si todo el proceso se ha concluido satisfactoriamente sin ningún error se debe tener el servidor Mosquitto funcionando y escuchando peticiones MQTT en su puerto por defecto 1883.

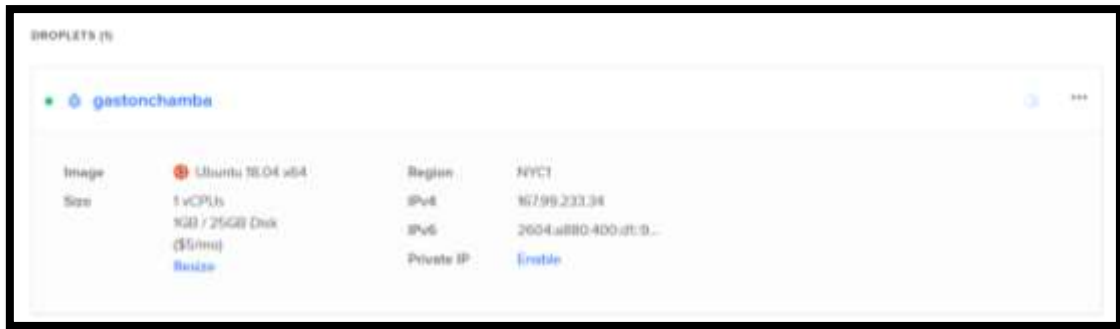


Figura 30: Características Ubuntu Server 18.04

6.2.4.2 Configuración del IDE de desarrollo Arduino.

En esta sección se configurará el IDE de desarrollo Arduino que se lo utilizará para la programación de los Sketches diseñados y posterior a esto cargarlos a la placa NodeMCU ESP8266.

Primero se descarga e instala el IDE de desarrollo Arduino de su página oficial [39]. Luego se ejecuta el programa, una vez abierto se procede a configurarlo ya que por defecto no trae las configuraciones correspondientes para poder programar el chip NodeMCU ESP8266. Dichas configuraciones están disponibles en [40].

Después vamos al menú superior a “Archivo” en el desplegable que nos muestra escogemos la opción “Preferencias” (Ver Figura 31) hacemos clic en ella, luego nos muestra una ventana donde nos muestra toda la configuración del IDE.

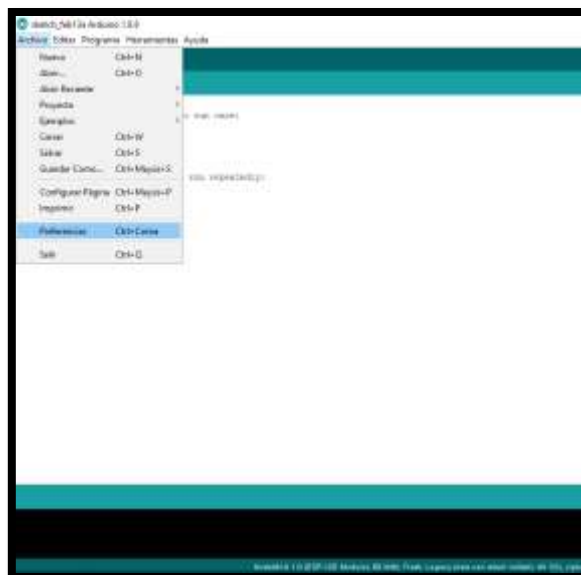


Figura 31: Preferencias de Arduino

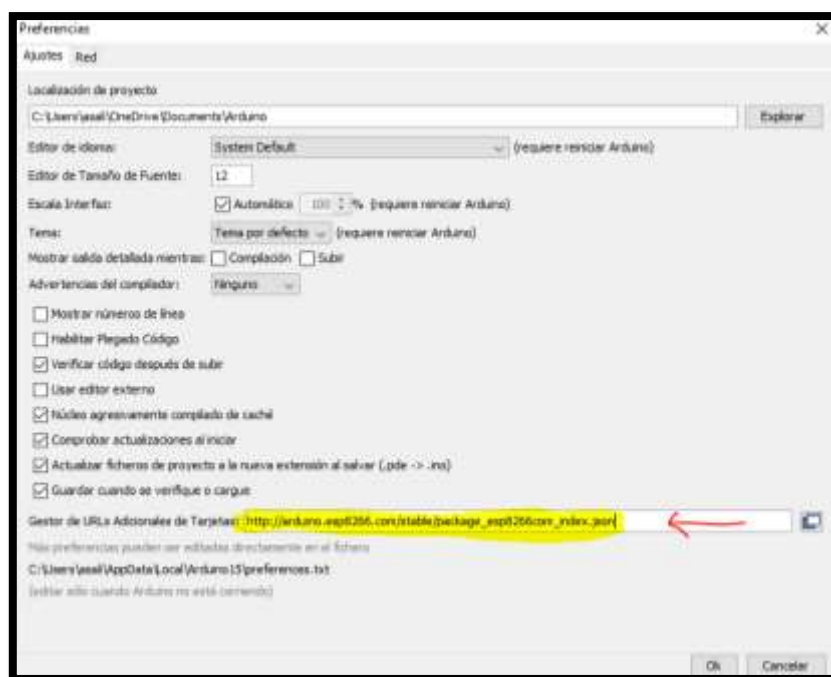


Figura 32: Configuración para la gestión de tarjetas ESP8266

En la ventana de preferencias vamos a la opción “Gestor de URL’s Adicionales de Tarjetas” en la cual vamos a copiar y a pegar el enlace encontrado en el repositorio de GitHub (Ver Figura 32). En este caso es:

https://arduino.esp8266.com/stable/package_esp8266com_index.json

Una vez realizado este proceso satisfactoriamente ya podremos gestionar las tarjetas basadas en ESP8266. Para instalar la tarjeta NODEMCU vamos a “Herramientas” □ “Placa” □ “Gestor de Tarjetas”, una vez hecho esto se mostrará una ventana en la cual podremos buscar las diferentes tarjetas Arduino. En este caso buscaremos la tarjeta “ESP8266” como resultado de esta búsqueda nos mostrara diferentes tarjetas, pero nosotros escogeremos la NODEMCU.

Para añadirla al entorno de desarrollo Arduino haremos clic en ella y en instalar. Luego de esto debemos escoger la placa de desarrollo en “Herramientas” □ “Placa” □ y navegar hacia abajo hasta que nos muestre la opción “NodeMCU 1.0 (ESP-12E Module)” (Ver Figura 33).

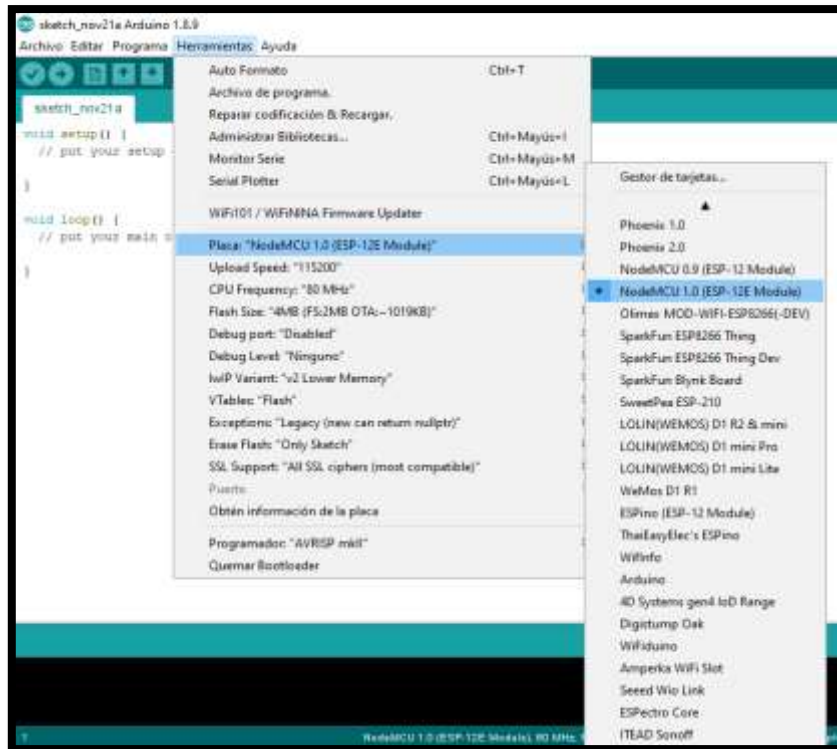


Figura 33: Configuración para la programación del ESP8266

Una vez elegida la placa procedemos a conectarla y comprobar que el puerto de comunicación serial se ha detectado correctamente en “Herramientas” □ “Puerto COMx” (Ver Figura 34).

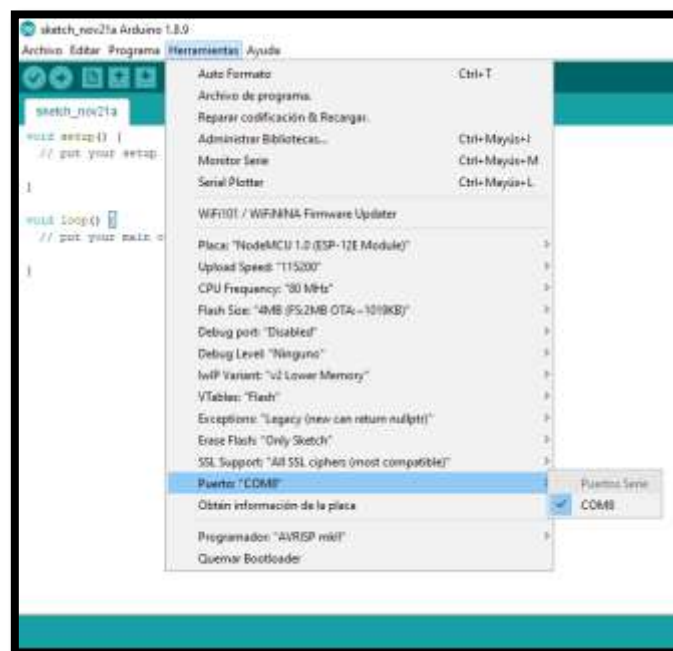


Figura 34: Configuración del Puerto Serial

6.2.4.3 Distribución de conexiones de los leds.

En la Figura 35 se muestra la distribución para la conexión de los leds, como en este caso utilizaremos cuatro leds entonces procederemos a definir qué pines de la placa utilizaremos para su respectivo control. Para la conexión utilizaremos los pines GPIO0, GPIO1, GPIO2, GPIO3 de tal manera que el positivo del led vaya conectado a lo 3V de salida de la placa y el negativo del LED vaya conectado a la salida GPIO de la misma.

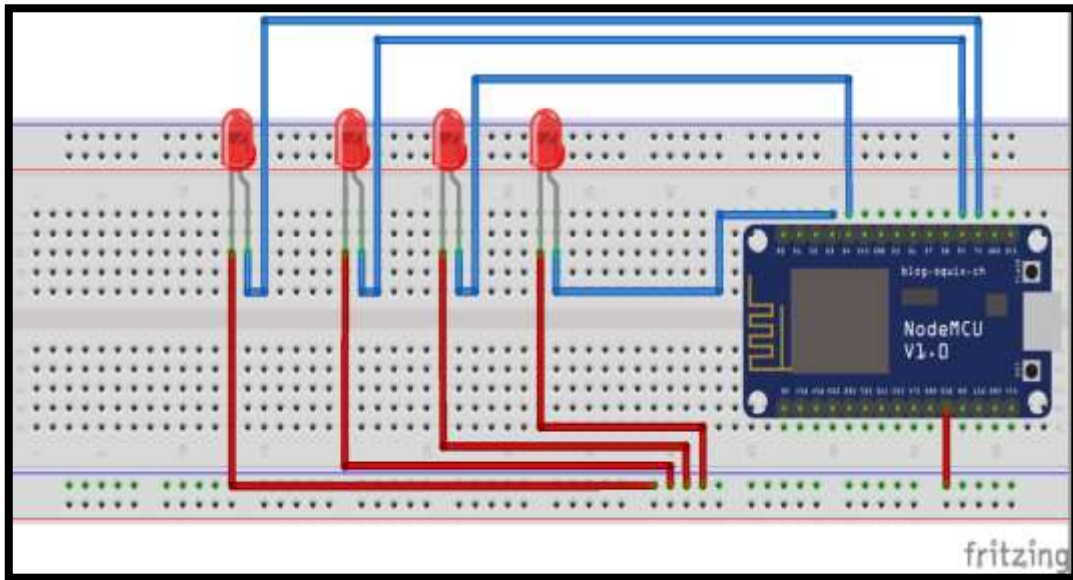


Figura 35: Diseño de la Conexión para los LED's

6.2.4.4 Distribución de la conexión del sensor de Temperatura- Humedad

En la Figura 36 se muestra la distribución de las conexiones del sensor de temperatura DHT11. Para nuestro caso utilizaremos la salida GPIO16 para la lectura de los datos.

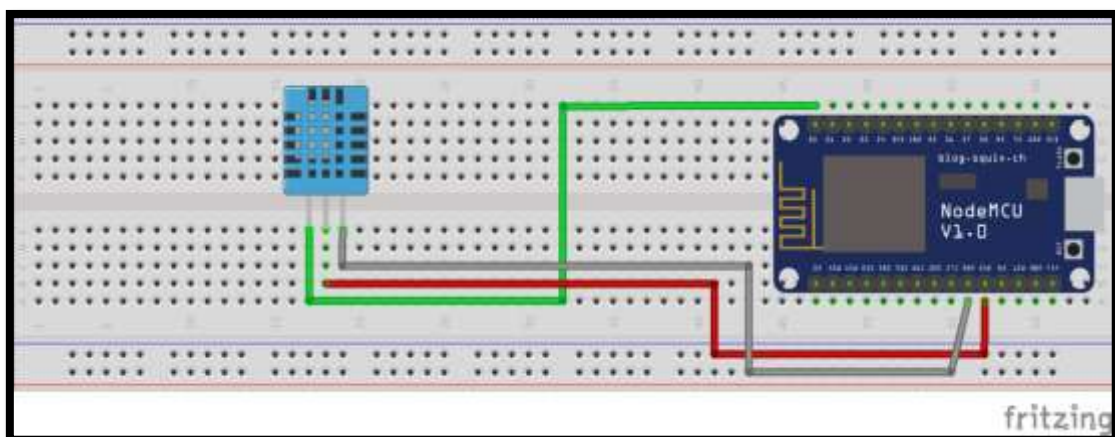


Figura 36: Diseño de la Conexión del Sensor de Temperatura y Humedad DHT11

6.2.4.5 Distribución del motor para abertura y cierre de puertas.

En la Figura 37 se muestra la distribución de las conexiones para girar el motor en ambos sentidos con la ayuda del módulo H-Bridge L298N. Para este proceso utilizaremos los pines GPIO13 para abrir la puerta y el GPIO15 para cerrarla. Además, se utilizará un UPS con fuente de energía externa que ayudará al motor para que se ponga en marcha.

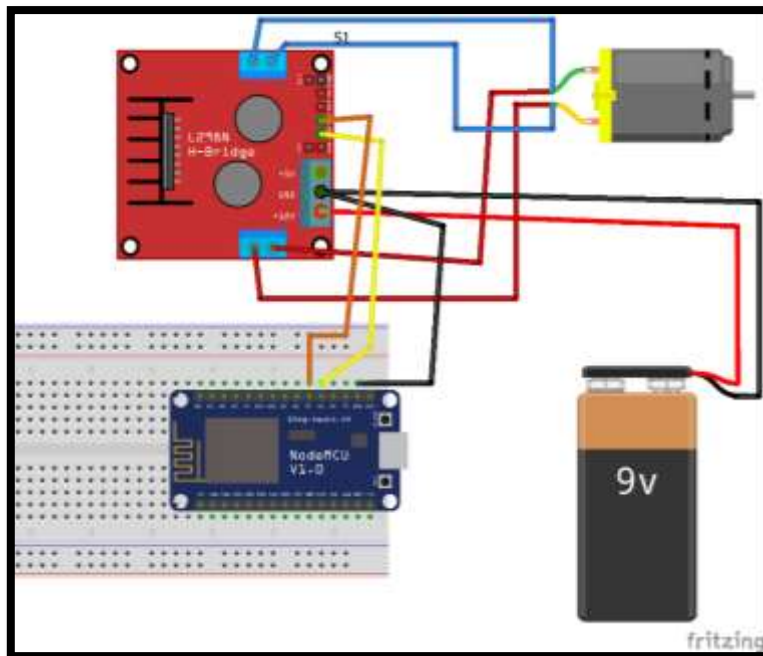


Figura 37: Diseño de la conexión para el motor de las puertas.

6.2.4.6 Distribución de los ventiladores.

En la Figura 38 muestra la conexión de los elementos que conformaran el sistema de ventilación. Para esto se hará uso de un relé doble y un UPS que ayudaran para que el ventilador se ponga en funcionamiento. Para nuestro ejemplo utilizaremos el pin GPIO4. Además, utilizaremos el pin GPIO5 para accionar el otro ventilador automáticamente cuando la temperatura sea mayor a 22 °C.

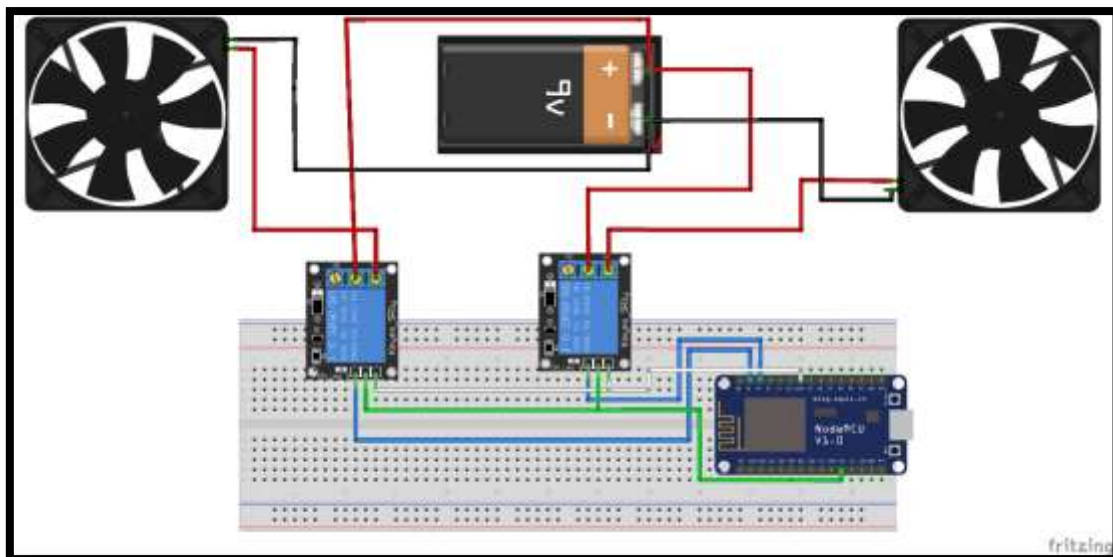


Figura 38: Diseño de la conexión para los ventiladores

6.2.4.7 Distribución sensor de presencia.

En la Figura 39 se muestra el sketch de la conexión del sensor PIR el cual va accionar por medio de un relé a una sirena en el instante que detecte presencia física. Para esta conexión utilizamos una alarma, el UPS, un relé y el sensor PIR, el cual va estar conectado al GPIO14 y cuando detecte presencia va a enviar una señal al GPIO16 el cual va a accionar el relé y por ende la alarma sonara.

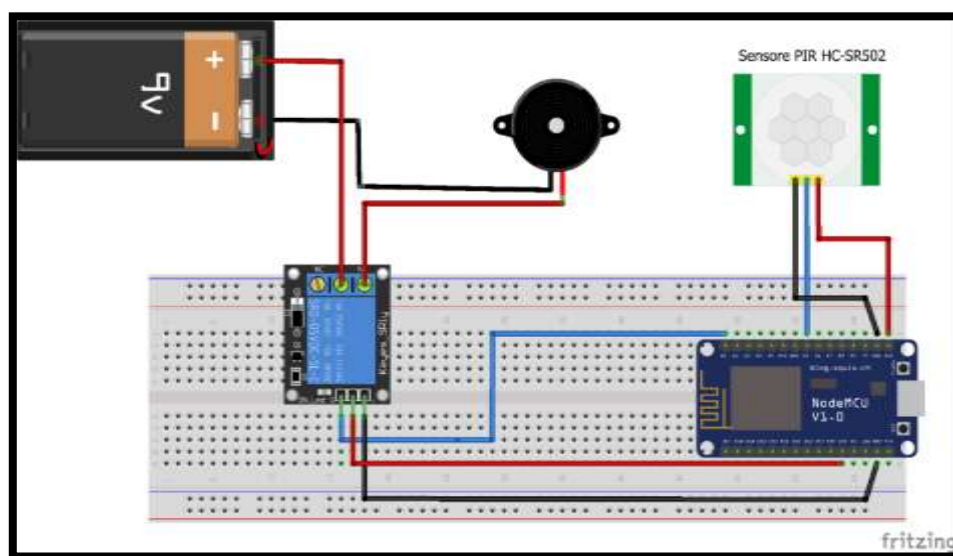


Figura 39: Diseño de la conexión Sensor PIR

A continuación, se puede estudiar los sketches realizados y se podrá codificarlo en con la ayuda del IDE de Desarrollo Arduino.

El funcionamiento del programa se realiza de la siguiente manera, primeramente, se realiza la lectura del sensor, tanto de temperatura como humedad y se envían los datos con MQTT al servidor. Asimismo, desde el cliente MQTT se enviarán los valores 0(OFF)/1(ON) con este protocolo a un tópico determinado para cambiar el estado de los actuadores según lo requiramos. Cabe mencionar que el sensor de presencia lo activaremos manualmente dando energía al NodeMCU donde estará alojado todo el módulo de seguridad y este accionará la alarma en caso de detección de presencia, además, enviará datos por medio del protocolo MQTT el cual servirá para enviar notificaciones al correo electrónico de la presencia detectada.

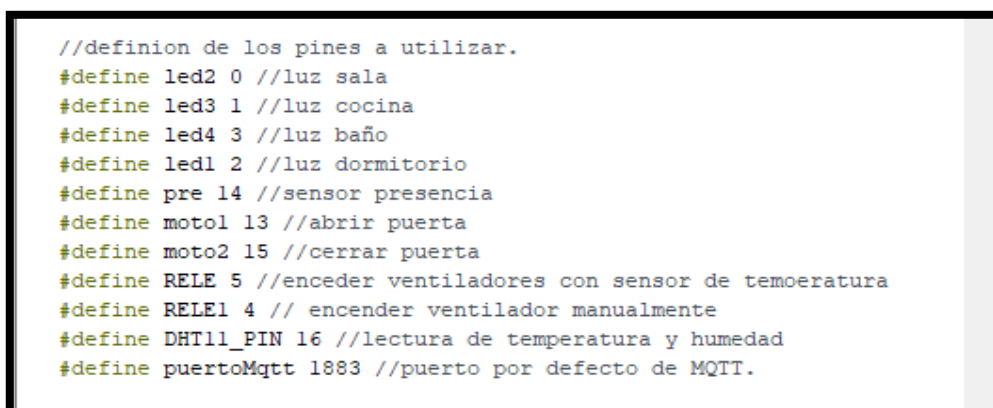
Primeramente, se debe añadir las librerías necesarias para el funcionamiento de los diferentes elementos en este caso son: ESP8266, WiFiClient, DHT, PubSubClient y definir los pines GPIO del NodeMCU a utilizar por cada elemento (Ver Figura 40). Luego se procede a poner los valores de la red Wifi y del servidor MQTT. Finalmente se escriben las instrucciones para cada sensor y actuador correspondiente al tópico que se desea.



```
mqqt_led$  
//Definición de librerías utilizadas.  
#include <ESP8266WiFi.h> //librería para poder utilizar la placa NodeMCU  
#include <PubSubClient.h> //librería MQTT para hacer publicaciones y Subscripciones a un topico.  
#include <dht.h> //librería para utilizar el sensor temperatura DHT11  
dht DHT;
```

Figura 40: Definición de Librerías

Una vez definidas las librerías se procede a definir los pines que vamos a utilizar.



```
//definición de los pines a utilizar.  
#define led2 0 //luz sala  
#define led3 1 //luz cocina  
#define led4 3 //luz baño  
#define led1 2 //luz dormitorio  
#define pre 14 //sensor presencia  
#define motol 13 //abrir puerta  
#define moto2 15 //cerrar puerta  
#define RELE 5 //encender ventiladores con sensor de temoeratura  
#define RELE1 4 // encender ventilador manualmente  
#define DHT11_PIN 16 //lectura de temperatura y humedad  
#define puertoMqtt 1883 //puerto por defecto de MQTT.
```

Figura 41: Definición de Pines GPIO

Una vez definido los pines (Ver Figura 41) a utilizar se procede a establecer las variables necesarias para la conexión inalámbrica (Ver Figura 42), primeramente, definimos las contantes de conexión wifi de la red local y después la del servidor MQTT, que no es más que la IP, y el puerto de escucha, en ocasiones es necesario un usuario y contraseña del servidor MQTT según como este configurado. Además, definimos ciertas variables que ayudaran a guardar datos de lectura del sensor de temperatura y humedad para enviarlos al tópicó especificado.

```
WiFiClient clienteWifi;//este cliente se encarga de la comunicacion con el wifi
PubSubClient clienteMQTT(clienteWifi);//este utiliza el cliente anterior para hacer poder cr
//si pasan por el hackerspace Xibalba pues ya tienen la clave
const char * ssid = "Alex";
const char * claveWifi = "12345678";
const char * brokerMqtt = "167.99.233.34";// ip del broker sin http ni nada solo los numeros

uint32_t ultimoIntentoReconexion;
uint32_t timerEnvioDatos;
long lastMsg = 0; //Variable para enviar datos de Temperatura
long lastMs = 0; //Variable para enviar datos de Humedad
char msg[50]; //Variable para recibir los datos obtenidos del sensor de Temperatura
char ms[50]; //Variable para recibir los datos obtenidos del sensor de Humedad
int value = 0;
```

Figura 42: Definición de Variables y Constantes

Asimismo, se crea una función “conectarWiFi” (Ver Figura 43) que permitirá la conexión a nuestra red local y mostrara por el puerto serial el estado de la conexión.

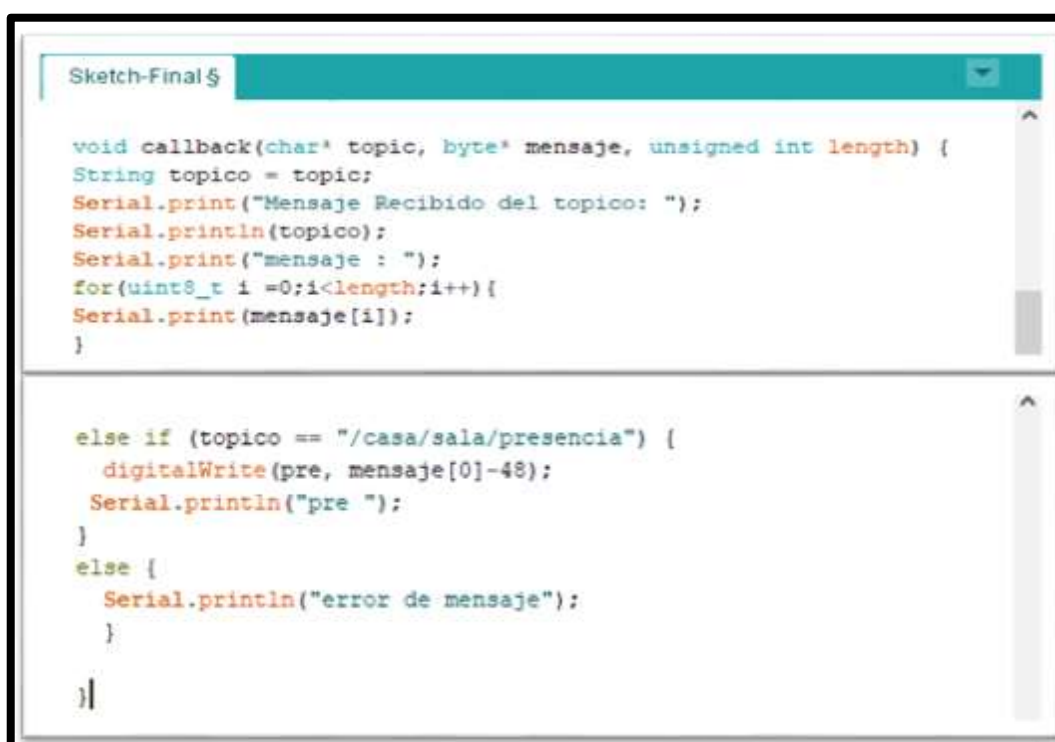
```
void conectarAlWifi() {

  WiFi.begin(ssid, claveWifi);
  Serial.print("conectando a");
  Serial.println(ssid);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("Wifi Conectado ");
  Serial.println("direccion IP: ");
  Serial.println(WiFi.localIP());
}
```

Figura 43: Función Para la conexión a la Red Local

Ahora se procederá a desarrollar el método “callback ()” (Ver Figura 44) que es el encargado de controlar los mensajes de suscripciones hechas por algún cliente MQTT. Además, nos muestra por el monitor serial el recibimiento de estos mensajes que los muestra carácter a carácter mediante la utilización de un bucle para finalmente comparar el mensaje recibido con las opciones que definimos en este caso son “0 y 1” para proceder al encendido o apagado de un actuador. La comparación de estos mensajes lo hace con la instrucción “digitalWrite (GPIOx, mensaje [0] -48)”, aquí le restamos -48 para que el valor sea 0 o 1 dicho en otras palabras le restamos el valor ASCII para hacerlo un entero.



```
Sketch-Final$

void callback(char* topic, byte* mensaje, unsigned int length) {
String topico = topic;
Serial.print("Mensaje Recibido del topico: ");
Serial.println(topico);
Serial.print("mensaje : ");
for(uint8_t i =0;i<length;i++){
Serial.print(mensaje[i]);
}

else if (topico == "/casa/sala/presencia") {
digitalWrite(pre, mensaje[0]-48);
Serial.println("pre ");
}
else {
Serial.println("error de mensaje");
}

}
}
```

Figura 44: Función Callback

La función “reconexión ()” (Ver Figura 45) nos permite conectar al servidor para poder publicar y subscribirnos en un tema en específico. Además, nos subscribimos a los tópicos definidos en la Figura 29 para poder controlar cada uno de los actuadores que conformaran el sistema. Asimismo, muestra el estado de la conexión MQTT y en caso de un fallo vuelve a reconectar al servidor MQTT.


```
mqtt_led $
boolean reconexion() {
  Serial.print("Conectando al broker mqtt");
  //intentando conectar al broker
  if (clienteMQTT.connect("ESP8266Client")) {
    Serial.println("Conectado");
    //publicamos que estamos conectados
    clienteMQTT.publish("/conexion", "Conectado");
    //nos suscribimos a los topicos para controlar los actuadores
    clienteMQTT.subscribe("/casa/cocina/luz");
    clienteMQTT.subscribe("/casa/sala/luz");
    clienteMQTT.subscribe("/casa/dormitorio1/luz");
    clienteMQTT.subscribe("/casa/dormitorio2/luz");
    clienteMQTT.subscribe("/casa/dormitorio3/luz");
    clienteMQTT.subscribe("/casa/dormitorio4/luz");
    clienteMQTT.subscribe("/casa/bano1/luz");
    clienteMQTT.subscribe("/casa/bano2/luz");
    clienteMQTT.subscribe("/casa/sala/ventilador");
    clienteMQTT.subscribe("/casa/dormitorio1/ventilador");
  } else {
    Serial.print("falló, rc=");
    Serial.print(clienteMQTT.state());
  }
  return clienteMQTT.connected();
}
```

Figura 45: Función Reconexión

A continuación, se explica los métodos “void setup()” y “void loop()”. El primer método permite configurar el funcionamiento de la placa NodeMCU V1.0. y la segunda para crear un bucle infinito de instrucciones deseadas.

El método “void setup()” (Ver Figura 46) lo utilizamos en este trabajo para definir como salidas los pines empleados anteriormente, en este caso definiremos como pines de salidas a todos los pines a los cuales están conectados los actuadores como los leds, ventiladores, motores y relés. En esta función también le decimos cual es el servidor y el puerto al que se debe conectar y se llama la función callback.

```
Sketch-Final $  
  
void setup() {  
  Serial.begin(115200);  
  Serial.println("iniciando programa");  
  
  pinMode(motol, OUTPUT);  
  pinMode(moto2, OUTPUT);  
  pinMode(RELE, OUTPUT);  
  pinMode(RELE1, OUTPUT);  
  pinMode(led2, OUTPUT);  
  pinMode(led3, OUTPUT);  
  pinMode(led4, OUTPUT);  
  pinMode(led1, OUTPUT);  
  pinMode(pre, OUTPUT);  
  conectarAlWifi();  
  //le decimos cual es el servidor y el puerto al que se debe conectar  
  clienteMQTT.setServer(brokerMqtt, puertoMqtt);  
  //le decimos como se llama la funcion de callback  
  clienteMQTT.setCallback(callback);  
}
```

Figura 46: Método Setup

En el método “void loop()” (Ver Figura 47) comprobamos si el cliente MQTT se ha conectado, si no es así lo reconecta. Dicho de otra manera, permite al cliente MQTT procesar los mensajes entrantes y mantener siempre activa la conexión. Además, se realiza aquí las publicaciones de los datos de temperatura y humedad a los tópicos definidos anteriormente; por ultimo comprueba si la temperatura es mayor a 22°C para accionar el funcionamiento del ventilador caso contrario lo desactiva.

```
Sketch-Final $  
  
clienteMQTT.loop();  
long now = millis();  
  
if (now - lastMsg > 7000) {  
  lastMsg = now;  
  int chk = DHT.read11(DHT11_PIN);  
  String msg= msg+ DHT.temperature;  
  char message[50];  
  msg.toCharArray(message, 50);  
  Serial.println(message);  
  //publish sensor data to MQTT broker  
  clienteMQTT.publish("/casa/sala/temperatura",message);  
  
  String ms= ms+ DHT.humidity;  
  char messag[50];  
  ms.toCharArray(messag, 50);  
  Serial.println(messag);  
  clienteMQTT.publish("/casa/bano/humedad",messag);  
}
```

Figura 47: Método loop

6.3 Desarrollar un prototipo para la solución informática que permita administrar la programación de eventos para el hogar inteligente.

En esta fase se desarrollará del Cliente MQTT con ayuda de la plataforma IoT Node-RED que permitirá la integración del hardware y el software de manera sencilla.

Primeramente, se procederá a la instalación y configuración de la plataforma Node-RED siguiendo los pasos del Anexo 3. Una vez instalado ejecutamos el servicio y accedemos por medio de un navegador WEB a la dirección donde tenemos levantado el servidor de Node-RED en este caso es:

<http://167.99.233.34:1880/>

Una vez que accedemos al servidor se muestra un editor de flujos en el cual permitirá crear la interfaz del sistema rápidamente.

A continuación, se creará el flujo de Autenticación para el acceso a la aplicación, para esto se hará uso de los nodos disponibles en la paleta del lado izquierdo.

Primeramente, se hará uso del nodo "Form" que pertenecerá a un Grupo llamado "Signin" que permitirá crear el formulario de autenticación, los datos necesarios para el formulario serán Usuario y contraseña, luego crearemos una función en la cual definimos los usuarios con sus respectivas credenciales, también se controlará la validación de las mismas una vez que las ingresen, caso contrario mostrará notificaciones como usuario incorrecto o contraseña incorrecta. Un punto importante también de esta función es que permite controlar la cantidad de usuarios que acceden a la misma vez, limitándolo a uno por lo que si otra persona desea ingresar el sistema notificara que está en Uso. La codificación de cada una de estas funciones está disponible en el Anexo 4. En la Figura 48 se muestra el flujo de la autenticación.

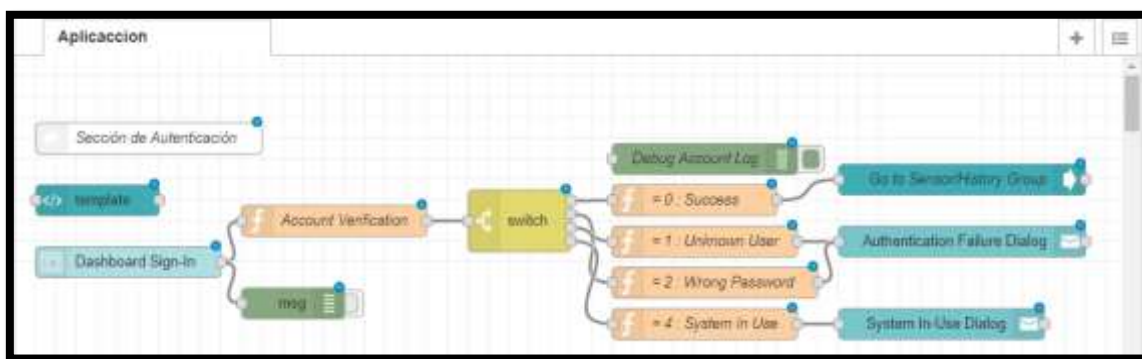


Figura 48: Flujo de Autenticación

Seguidamente se procede a crear el flujo que nos permitirá controlar el tiempo de sesión que estará disponible el sistema ver Figura 49. Para esto hacemos uso de un nodo de entrada Inject para crear el intervalo de tiempo de la sesión activa del sistema. Además, se crea una función en la cual se controla dicho tiempo y una vez terminado el tiempo redireccioné a la interfaz de inicio de sesión y la otra función hace un cálculo para cuando este punto de terminar el tiempo de sesión notifique al usuario. Las instrucciones de las funciones se detallan en el Anexo 4.

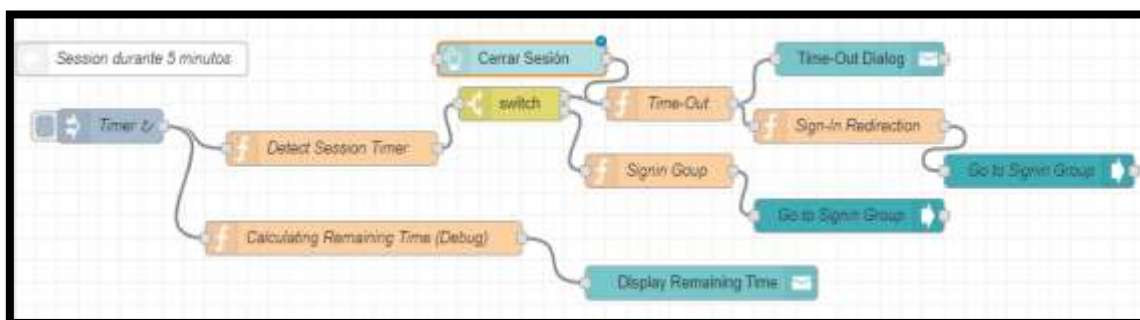


Figura 49: Flujo de Tiempo de Sesión

Una vez finalizada la creación de los flujos de autenticación y de tiempo de Sesión se procede a la creación de los layouts que serán visibles en el dashboard ver Figura 50. Una vez terminado este proceso accedemos a la dirección de la aplicación en este caso es: <http://167.99.233.34:1880/ui/> una vez que ingresamos a esta dirección se mostrara el resultado esperado. Ver Figura 51.

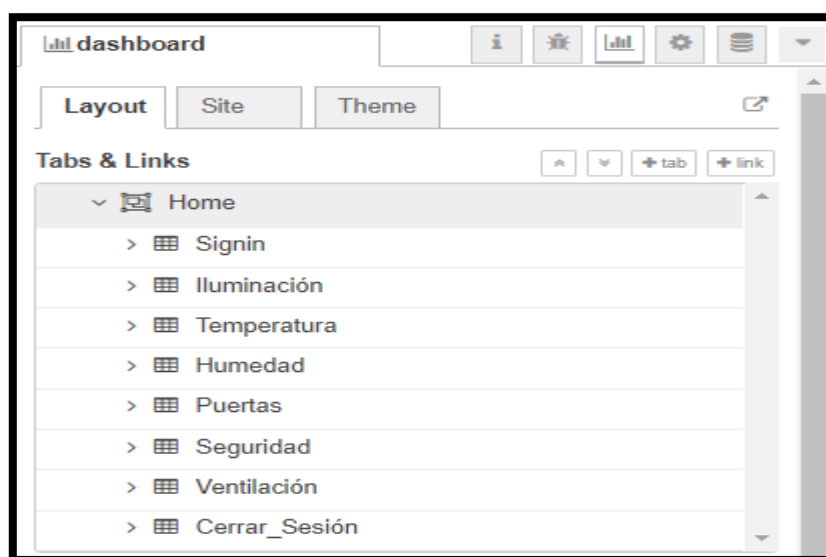


Figura 50: Estructura del Layout.

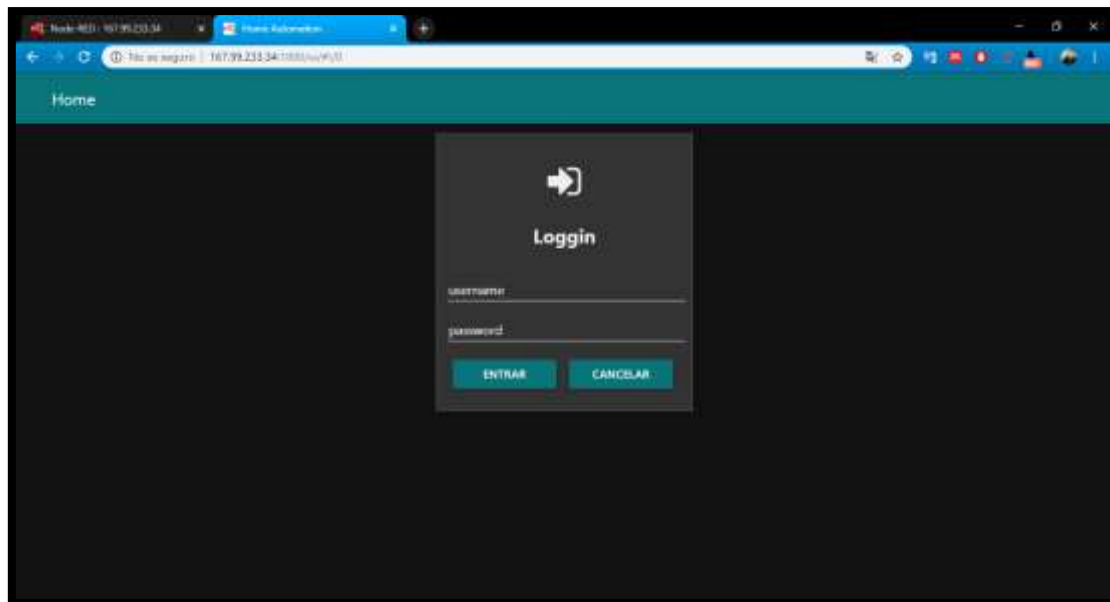


Figura 51: Dashboard de Autenticación

Una vez finalizado el módulo de autenticación procederemos a crear los flujos que permitan controlar los diferentes actuadores y obtener la lectura del sensor utilizado.

Primero crearemos el flujo de control de la iluminación ver Figura 55 para esto hacemos uso de un Toogle Switch que nos permitirá enviar mensajes de 0 o 1 a los diferentes tópicos establecidos con anterioridad. Asimismo, utilizamos el nodo de salida MQTT que nos permitirá publicar mensajes en un tema en específico; en este caso será la iluminación. La configuración de este nodo de lo hace de la siguiente manera: Como vamos a crear nuestro primer flujo es necesario crear un nuevo bróker MQTT (Ver Figura 52), para ello se da clic en el nodo “mqtt” y en la ventana emergente vamos a “server” “Add new bróker-mqtt” . y damos clic en editar una vez en la ventana de edición vamos a la pestaña de “Conexion” y en el cuadro de server escribimos la dirección IP de donde está montado el servidor MQTT, y le damos un nombre en “Name” para identificarlo. También es necesario definir el puerto por el cual está escuchando el servidor MQTT en este caso será por el puerto 1883. Una vez completado estos pasos damos clic en “Add” y listo ya tenemos configurado el broker MQTT para mantener las comunicaciones desde nuestra aplicación al servidor y posterior a esto tener acceso al control de sensores y actuadores.

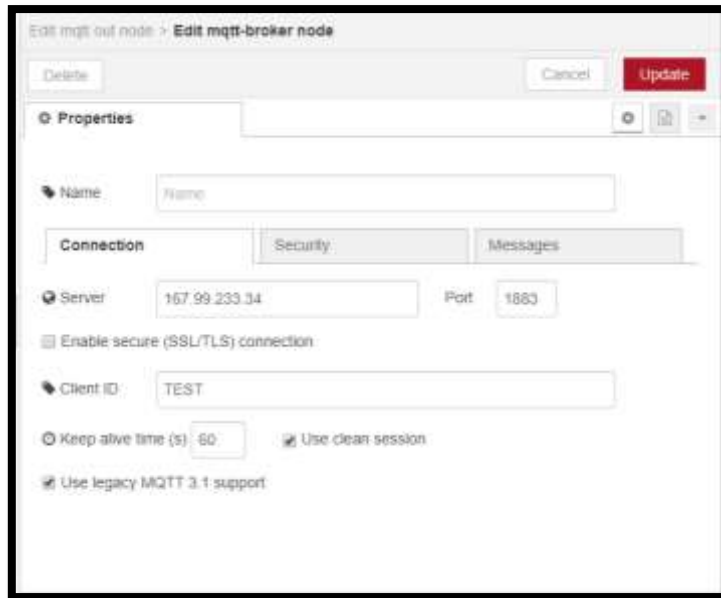


Figura 52: Configuración bróker MQTT- Node-RED

Otro nodo que hay que configurar es el de la base de datos. Primeramente, instalamos el nodo MySQL, luego de esto arrastramos el nodo al área de trabajo y lo configuramos de la siguiente manera.(Ver Figura 53.)

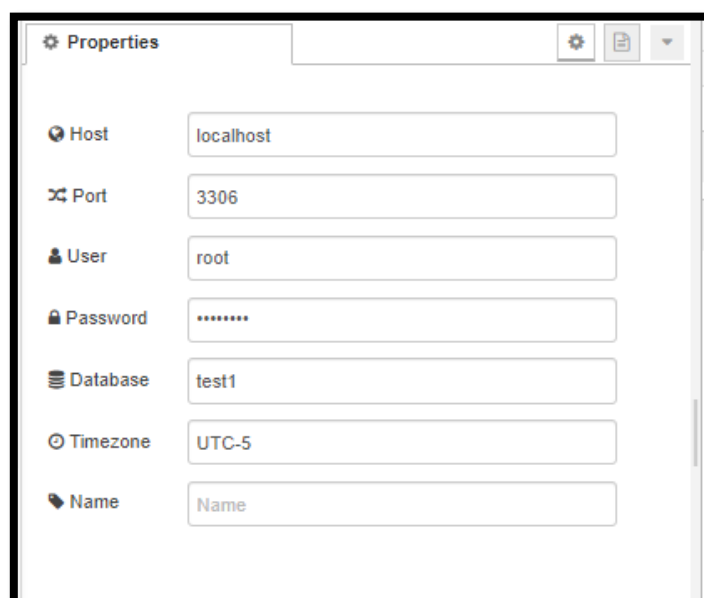


Figura 53. Configuración del nodo MySQL

Cabe recalcar que para utilizar la base de datos MySQL hacemos uso del sistema de gestión de base de datos llamado XAMPP. Sistema que se instala en el servidor virtual

contratado. Para luego Crear la base de datos y una tabla para cada uno de las acciones que se va a guardar.

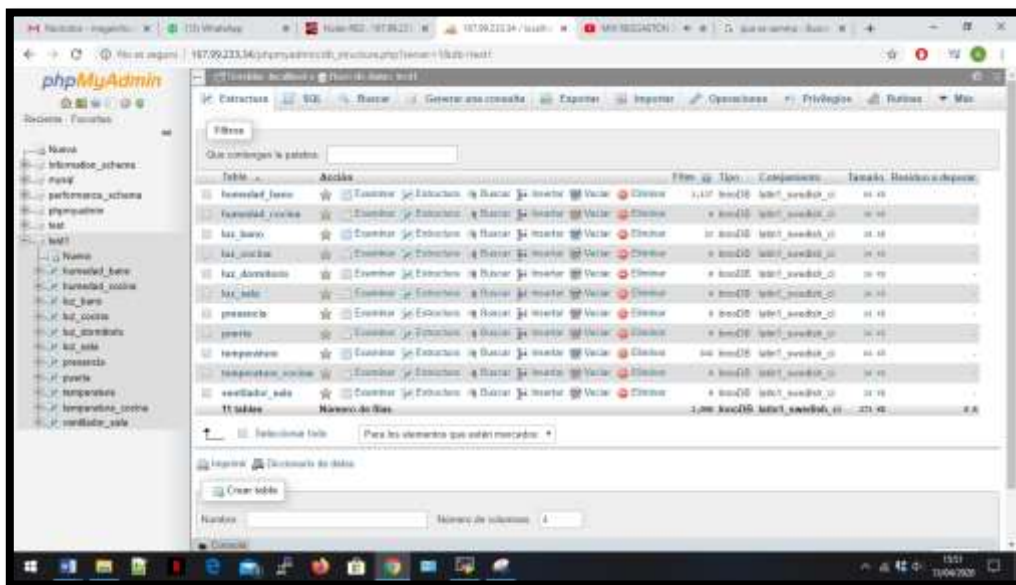
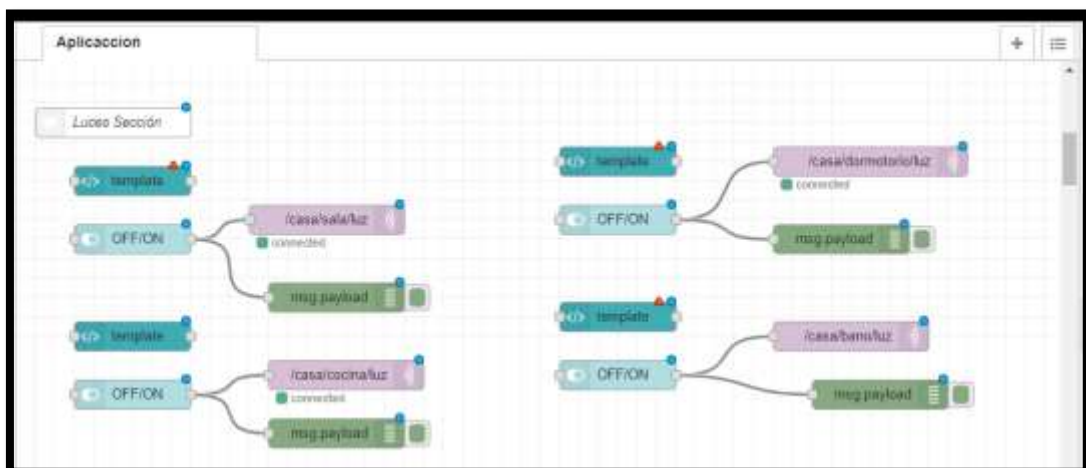
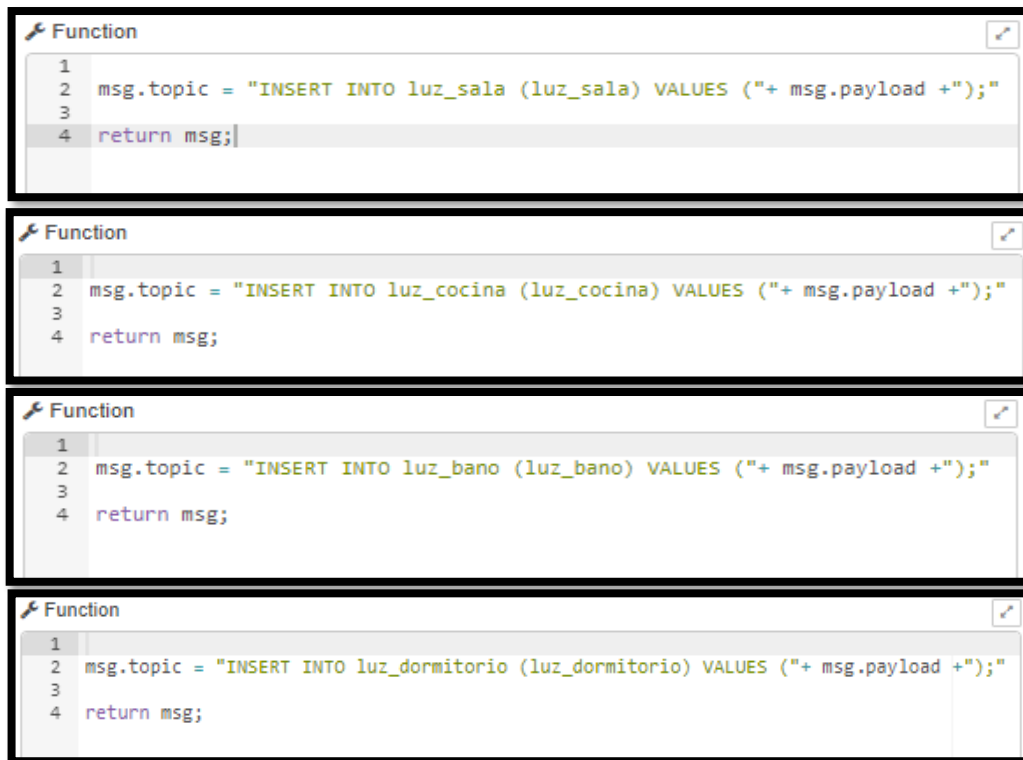


Figura 54. Estructura de la base de Datos en MySQL

Ahora bien, se procede a desarrollar el flujo de la iluminación para esto añadimos los cuatro nodos “mqtt” en los cuales la configuración solo va a cambiar en el tópicos ya que es aquí donde añadimos la jerarquía del tópicos al cual vamos a publicar el mensaje. También es necesario elegir un nivel de calidad de servicio del protocolo para esto le dejaremos en el nivel QoS1 que asegurara la entrega de los mensajes como mínimo una vez. Cabe mencionar que también es necesario configurar el switch para que envíe como datos números enteros ya sea 1 o 0 misma configuración que será general para el resto de Switchs utilizados en la presente aplicación. Estos primero cuatro switchs van a pertenecer al grupo Iluminación creado en el Layout (Ver Figura 50).



Por otro lado, se crea las funciones que permitirán guardar cada acción que se realice en una base de datos llamada test1, tomando en cuenta que 0 va a ser igual a apagado y 1 va a ser igual a encendido (Ver Figura 56..).



```
Function
1
2 msg.topic = "INSERT INTO luz_sala (luz_sala) VALUES (" + msg.payload + ");";
3
4 return msg;

Function
1
2 msg.topic = "INSERT INTO luz_cocina (luz_cocina) VALUES (" + msg.payload + ");";
3
4 return msg;

Function
1
2 msg.topic = "INSERT INTO luz_bano (luz_bano) VALUES (" + msg.payload + ");";
3
4 return msg;

Function
1
2 msg.topic = "INSERT INTO luz_dormitorio (luz_dormitorio) VALUES (" + msg.payload + ");";
3
4 return msg;
```

Figura 56. Funciones para guardar las acciones sobre la iluminación en MySQL.

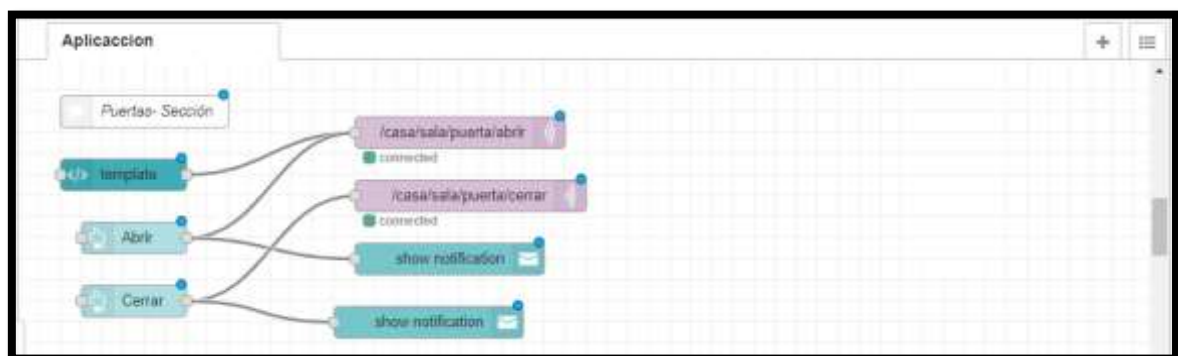


Figura 57: Apertura y Cierre de Puertas

Una vez finalizado todo este proceso procedemos a crear el siguiente flujo que controlará la apertura y cierre de las puertas (Ver Figura 57). Para ello se agrega dos nodos “mqtt” que al igual que en el flujo de iluminación llevara la misma configuración y lo único que cambiara serán los tópicos. Para poder gestionar las puertas se hará uso de dos nodos “Button” uno para abrir la puerta y el otro para cerrarla de tal forma que el que abre

publica al t pico dedicado a accionar el motor para abrir y el otro publica en el t pico dedicado a accionar el motor para cerrarla. Estos botones van a pertenecer al grupo llamado Puertas creado en el Layout (Ver Figura 50).

Asimismo, se crea la funci n que permitir n guardar cada acci n que se realice sobre la puerta en una base de datos, tomando en cuenta que 0 va a ser igual a cerrada y 1 va a ser igual a abierta (Ver Figura 58.).

```
Function
1
2 msg.topic = "INSERT INTO puerta (puerta_abierta) VALUES (" + msg.payload + ")"
3
4 return msg;
```

```
Function
1
2 msg.topic = "INSERT INTO puerta (puerta_cerrada) VALUES (" + msg.payload + ");"
3
4 return msg;
```

Figura 58. Funciones para guardar las acciones sobre la puerta en MySQL

Seguidamente se crear  el flujo que permita controlar la ventilaci n (Ver Figura 59) para esto se hace uso de un switch y el nodo de salida "mqtt". Este switch va a pertenecer al grupo Ventilaci n creado en el Layout (Ver Figura 50).

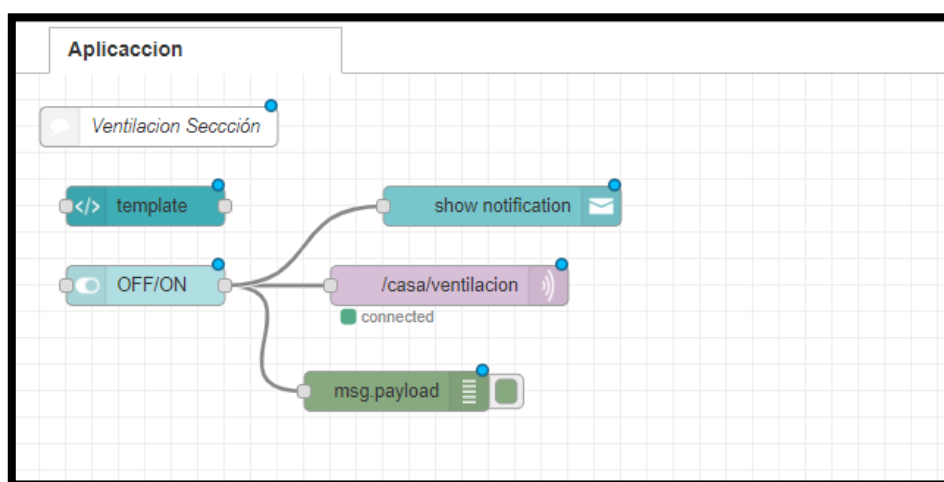


Figura 59: Flujo Control de Ventilaci n

Luego se crea la funci n que permitir n guardar cada acci n que se realice sobre el ventilador en la base de datos, tomando en cuenta que 0 va a ser igual a apagado y 1 va a ser igual a encendido (Ver Figura 60.).

```
Function
1
2 msg.topic = "INSERT INTO ventilador_sala (ventilador_sala) VALUES (" + msg.payload + ");";
3
4 return msg;
```

Figura 60. Función para guardar las acciones sobre el ventilador en MySQL

En este apartado se desarrollará el flujo que nos permita visualizar la temperatura y la humedad ambiente (Ver Figura 61). En este caso utilizaremos los nodos “MQTT” pero de entrada que nos permitirá subscribirnos al sensor para obtener dichos valores. Además, se hará uso de los nodos “gauge” que nos brindan una interfaz dinámica para la lectura de los datos. Luego de esto se crea una función llamada “Email Temperatura Excedida” (Ver Anexo 4) función que controla si la temperatura ambiente es mayor a 26 grados este envía un correo electrónico como notificación mostrando una alerta y la lectura de la temperatura actual, este proceso se lo realiza con la ayuda del nodo email de Node-RED.

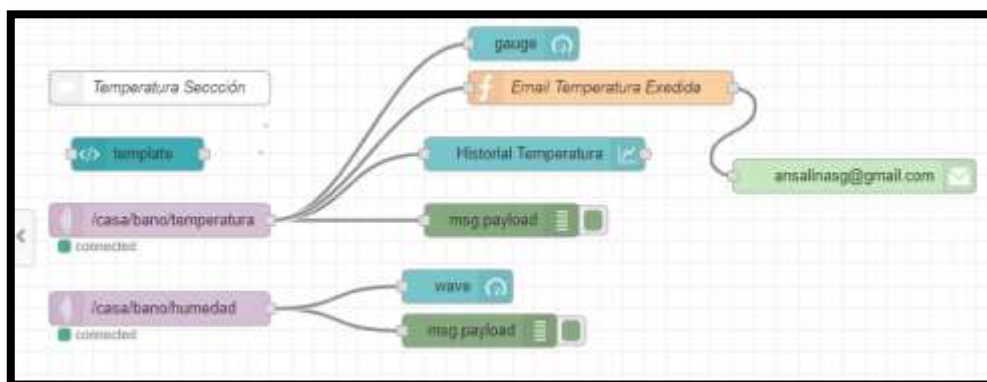


Figura 61: Lectura de Temperatura y Humedad

Para guardar la lectura de los recibidos por parte del sensor en la base de datos se hace uso de las siguientes funciones (Ver Figura 62.).

```
Function
1 temperatura = msg.payload
2
3 msg.topic = "INSERT INTO temperatura (temperatura_dormitorio) VALUES (" + temperatura + ");";
4
5 return msg;
```

```
Function
1 humedad = msg.payload
2
3 msg.topic = "INSERT INTO humedad_bano (humedad_bano) VALUES (" + humedad + ");";
4
5 return msg;
```

Figura 62. Función para guardar la lectura del sensor DHT11 en MySQL

Finalmente se crea el flujo que conformara nuestro sistema domótico. Este flujo será el de gestión de seguridad (Ver Figura 63). En este caso se utilizará el modo MQTT de la paleta input para recibir los datos desde el sensor ya sea 1 o 0 mostrándolos gráficamente por medio de un gauge. Además, se crea una función que permita enviar notificaciones mediante correo electrónico y por medio de un cuadro de dialogo en la interfaz web del sistema domótico cuando el sensor detecte presencia física. Este flujo va a pertenecer al grupo llamado Seguridad creado el Layout (Ver Figura 50). Cabe recalcar que las demás funciones que ayudan a la gestión de seguridad se las hace internamente en el microcontrolador.

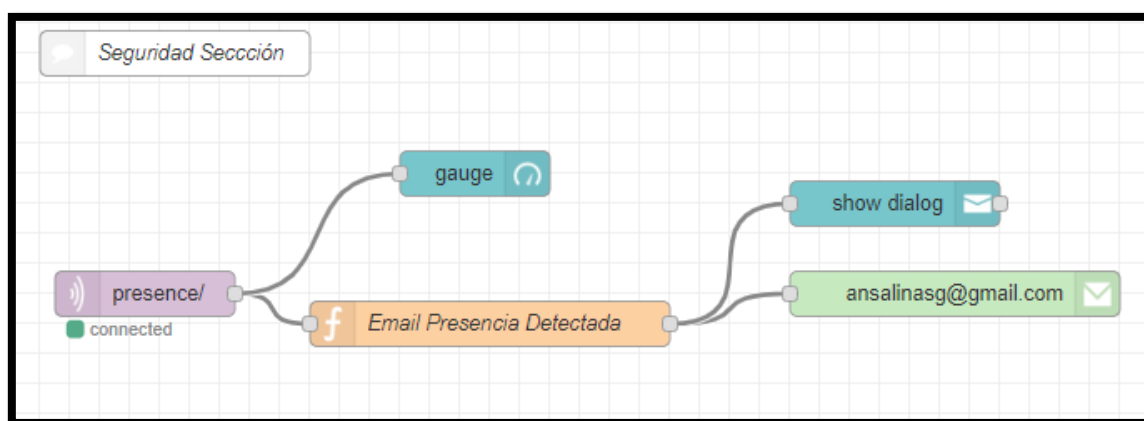


Figura 63: Flujo Control de Seguridad

Luego de crear el flujo de presencia se procede a crear la función para guardar la lectura del sensor de presencia tomando en cuenta que el valor 1 es cuando el sensor detecta movimiento y 0 cuando no hay detección de presencia (Ver Figura 64.).

```
Function
1 presencia = msg.payload
2
3 msg.topic = "INSERT INTO presencia (presencia) VALUES (" + presencia +");"
4
5 return msg;
```

Figura 64. Función para guardar la lectura del sensor PIR en MySQL

Una vez finalizado todo este proceso se accede al dashboard para comprobar el funcionamiento de cada uno de los flujos. La vista final se observa en la Figura 65 una vez que ingresamos las credenciales creadas en el flujo de autenticación.

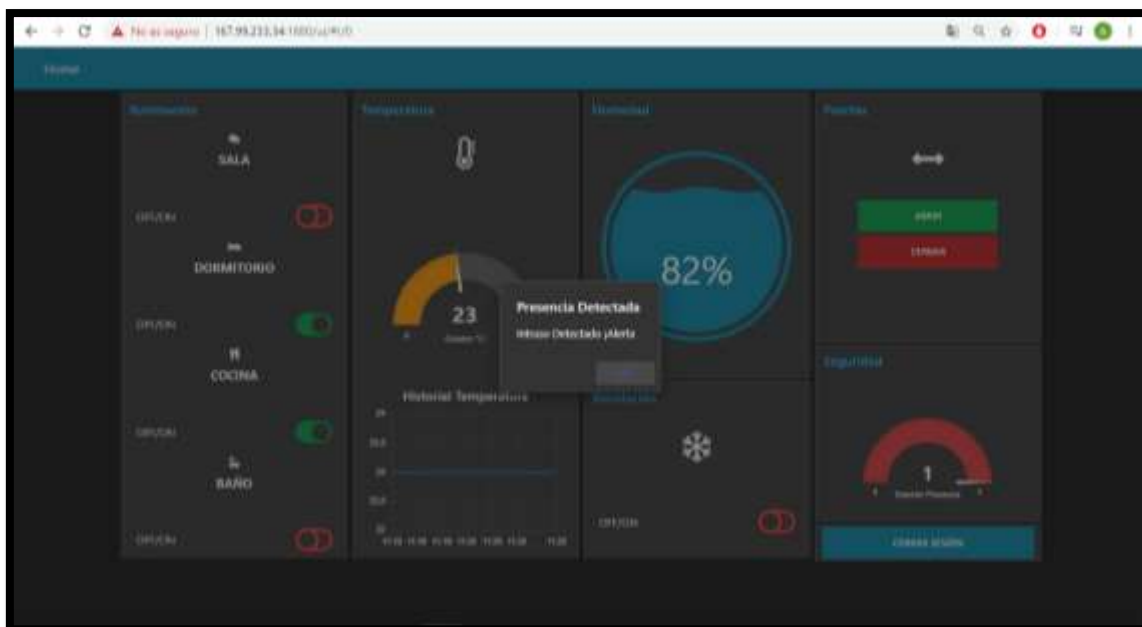


Figura 65: Interfaz del Sistema Domótico

6.4 Probar y evaluar el funcionamiento del prototipo del sistema de control del hogar inteligente en un entorno simulado.

En este apartado se detallará las pruebas realizadas durante el desarrollo del sistema domótico, cuyo fin es el de detectar cualquier fallo y asegurar que el sistema funcione correctamente.

Las pruebas se realizaron durante el proceso de desarrollo del sistema, los test se hicieron de acuerdo a los requerimientos definidos para el mismo por lo tanto fue necesario validar el correcto funcionamiento de cada uno de los módulos para poder avanzar con el siguiente hasta finalizar con todos los requerimientos.

6.4.1 Pruebas.

Para la realización de las pruebas se tomó como base el estándar IEEE 829 que sirve para pruebas de sistemas y para la realización de los documentos de la misma, permitiendo plasmar todos los aspectos claves de las pruebas. El proceso de prueba determinara si el producto se ajusta a los requisitos y si el sistema satisface su uso previsto y las necesidades del usuario.

6.4.1.1 Identificador único del Documento.

DOC-01

6.4.1.2 Introducción.

El documento a continuación, describe el alcance, la aproximación, los recursos, la planificación y las actividades necesarias. Primeramente, identifica elementos de prueba, luego las características que deben probarse, las tareas de prueba y por ultimo lo que hará cada tarea. Para las pruebas fue necesario alojar la aplicación en un servidor virtual basado en el sistema operativo Ubuntu Server 18.04 en el cual se instaló la plataforma Node-Red para el desarrollo del cliente y el servidor de comunicaciones MQTT que permitirá la comunicación del hardware con el software. Así mismo esto servirá para mejorar la experiencia de los usuarios que van a ser parte de las pruebas.

6.4.1.3 Elementos

Los elementos a ser probados son tres:

- Aplicación Web de control del hogar inteligente denominado Home en su versión 1.0
- Servidor de comunicaciones MQTT

6.4.1.4 Características a probar

Las características a probar están relacionadas con:

- Interfaz.
- Funcionalidad.
- Soporte.

Interfaz: Que cuente con todas las pantallas definidas en la especificación de Requerimientos (Ver Anexo 1).

Funcionalidad: De acuerdo a lo especificado en la especificación de requerimientos las funciones a ser probadas se detallan a continuación (Ver Tabla XV).

Tabla XV. Casos de Prueba

Código	Caso de Prueba	Detalle	Requerimiento
CP01	Inicio de Sesión	Ingreso mediante un usuario y contraseña.	RF001
CP02	Cierre de Sesión	Cierre de sesión manual y automático	RF002
CP03	Cantidad de usuario autenticados al mismo tiempo	Cantidad de usuarios que ingresan a la misma vez a la aplicación	RF003
CP04	Funcionamiento de Leds	Encendido y apagado de luces mediante el sistema.	RF004
CP05	Funcionamiento de Ventiladores	Encendido y apagado de ventiladores mediante el sistema.	RF005
CP06	Funcionamiento de la Puerta	Giro del motor en sentido horario y anti horario por medio del sistema.	
CP07	Obtención de valores de temperatura y humedad.	Obtener temperatura y humedad ambiente del lugar donde se encuentra el sensor.	RF007- RF008
CP08	Detectar presencia	Detección de presencia mediante sensor PIR	RF009

6.4.1.5 Características que no se deben probar.

Las características que no se deben probar están relacionadas con:

- Errores relacionados con el tiempo.
- Condiciones de errores no detectadas.
- Condiciones especiales de los datos.

- Incapacidad de soportar un gran volumen de carga o fallas de software.

6.4.1.6 Enfoque.

De acuerdo al criterio de los involucrados en el presente TT las pruebas se realizarán en un ambiente simulado y debido a la capacidad operativa y de costo que implica las pruebas en un ambiente real.

Se tomará como usuarios de pruebas a 15 personas de diferentes viviendas del sector Época de la ciudad de Loja.

El éxito de las pruebas estará determinado por esta población y de ser necesario se repetiría varias veces las pruebas hasta obtener el cien por ciento de éxito en pruebas.

La forma de medir el éxito de las pruebas será mediante una encuesta contestada por los involucrados en las pruebas.

6.4.1.7 Criterios de aprobación.

En este apartado se describe los criterios de éxito para evaluar los resultados de las pruebas.

Tabla XVI. Caso de Prueba Inicio de Sesión

CP01-CR01	
Identificador	
Descripción	Prueba del Funcionamiento del inicio de sesión.
Requisito del Sistema	RF001
Estado Inicial	Usuario Sin Autenticar
Estado Final	Usuario Autenticado
Procedimiento de la Prueba	<ul style="list-style-type: none">- Acceder a la aplicación- Ingresar usuario y contraseña- Validar credenciales- Ingresar usuario Incorrecto- Notificación de Usuario Incorrecto- Ingresar clave incorrecta- Notificación de usuario Incorrecta

Tabla XVII. Caso de Prueba Cierre de Sesión

Identificador	CP02-CR02
Descripción	Prueba del Cierre de Sesión Automático
Requisito del Sistema	RF002
Estado Inicial	Sesión activa
Estado Final	Sesión terminada
Procedimiento de la Prueba	<ul style="list-style-type: none"> - Acceder a la aplicación. - Autenticarse - Esperar cinco minutos - Notificaciones de Sesión por Terminar - Cierre de la aplicación

Tabla XVIII. Caso de Prueba Cantidad de Usuarios autenticados al mismo tiempo.

Identificador	CP03-CR03
Descripción	Prueba de cantidad de usuario autenticados al mismo tiempo.
Requisito del Sistema	RF003
Estado Inicial	Usuario Autenticado
Estado Final	Sistema en Uso
Procedimiento de la Prueba	<ul style="list-style-type: none"> - Acceder a la aplicación - Autenticarse - Abrir otro navegador e intentar acceder - Notificación de sistema en uso

Tabla XIX. Caso de Prueba Encendido y Apagado de Luces

Identificador	CP04-CR04
Descripción	Prueba de funcionamiento de Leds.
Requisito del Sistema	RF004
Estado Inicial	Led Apagado
Estado Final	Led Encendido
Procedimiento de la Prueba	<ul style="list-style-type: none"> - Se conecta el microcontrolador - Acceder a la aplicación - Autenticarse - Se realiza peticiones de modificar estado del Led de acuerdo a la instalación de la casa deseada. - El led cambia de estado - La acción realizada se guarda en una base de datos.

Tabla XX. Caso de Prueba Encendido y apagado de Ventiladores.

Identificador	CP05-CR05
Descripción	Prueba de funcionamiento de Ventiladores.
Requisito del Sistema	RF005
Estado Inicial	Ventilador Apagado
Estado Final	Ventilador Encendido
Procedimiento de la Prueba	<ul style="list-style-type: none"> - Se conecta el microcontrolador - Acceder a la aplicación - Autenticarse - Se realiza peticiones de modificar estado del ventilador. - El ventilador cambia de estado - Se aumenta temperatura igual o mayor a 21 °C

	<ul style="list-style-type: none"> - El ventilador cambia de estado apagado a encendido - Se disminuye la temperatura menos de 21 °C cambia de estado encendido a apagado. - La acción realizada se guarda en una base de datos.
--	---

Tabla XXI. Caso de Prueba Cierre y Apertura de Puertas

CP06-CR06	
Identificador	
Descripción	Prueba de funcionamiento del motor que accionar la puerta
Requisito del Sistema	RF006
Estado Inicial	Motor estado inicial
Estado Final	Motor estado final
Procedimiento de la Prueba	<ul style="list-style-type: none"> - Se conecta el microcontrolador - Acceder a la aplicación - Autenticarse - Se realiza petición de modificar estado de motor. - El motor cambia al estado solicitado. - Dicha acción es guardada en una base de datos.

Tabla XXII. Caso de Prueba Lectura de Temperatura y Humedad Ambiente

CP07- CR07	
Identificador	
Descripción	Prueba de funcionamiento del sensor DHT11
Requisito del Sistema	RF007 –RF008

Estado Inicial	Sensor DHT11 no envía y guarda datos
Estado Final	Sensor DHT11 envía y guarda datos.
Procedimiento de la Prueba	<ul style="list-style-type: none"> - Se conecta el microcontrolador - Acceder a la aplicación - Autenticarse - Visualizar información de temperatura y humedad en sección de Temperatura Ambiente. - El sensor envía datos de temperatura y humedad correctamente mostrándolos en un gráfico. - Los datos recibidos al sistema son guardados en una base de datos.

Tabla XXIII. Caso de Prueba de Gestión de Seguridad.

CP08-CR08	
Identificador	
Descripción	Prueba de funcionamiento sensor de presencia
Requisito del Sistema	RF009
Estado Inicial	Gestiona seguridad ante eventos inesperados
Estado Final	No gestiona seguridad
Procedimiento de la Prueba	<ul style="list-style-type: none"> - Se conecta el microcontrolador. - El primer escenario el sensor detecta movimiento activando una alarma y envía notificación al correo electrónico de dicha presencia. - El valor se guarda en la base de datos con fecha y hora de la detección.

	<ul style="list-style-type: none">- El segundo escenario consiste en que cuando el sensor de temperatura detecta una temperatura mayor a 26°C dispara una alarma, enciende luces de evacuación desde el lugar donde se detectó dicha temperatura elevada, abre la puerta automáticamente, envía notificación mediante correo electrónico de dicho evento y guarda en base de datos los valores de temperatura y apertura de la puerta.
--	--

6.4.1.8 Criterios de Suspensión.

Los motivos por los cuales se suspendan las pruebas contemplan:

- No disponibilidad de usuario.
- Errores relacionados con el servidor.

6.4.1.9 Pruebas de entregables.

Los resultados de las pruebas son los documentos que entregara el equipo de pruebas al final del proceso de las pruebas.

- Informe de pruebas de aceptación.

6.4.1.10 Necesidades ambientales.

En cuanto a: SOFTWARE y HARDWARE

- Dispositivo inteligente ya sea: Portátil, Smartphone, Tablet y que cuentes con un navegador Web con acceso a Internet.

6.4.1.11 Responsabilidades.

Alex Nixon Salinas Granda: Pruebas del Sistema y Documentación.

6.4.1.12 Necesidades del personal y formación.

Familiarizado con el uso de un dispositivo inteligente.

6.4.1.13 Riesgos y contingencias.

Posibles riesgos:

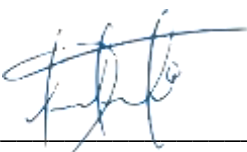
- No contar con el equipo necesario para realizar las pruebas.
- No contar con la conexión a internet.

Contingencias:

- Se proveerá al usuario un equipo necesario para las pruebas.
- Se proveerá al usuario conexión a internet estable.

6.4.1.14 Aprobaciones.

Las pruebas están avaladas por el encargado de las pruebas y su documentación.



Alex Nixon Salinas Granda

Encargado de las Pruebas



Ing. Gastón Chamba

Director del presente TT

6.4.2 Reporte de Pruebas

Como se estableció en el documento anterior las pruebas están vinculadas a los ejes de funcionalidad, soporte e interfaz.

6.4.2.1 Pruebas de Interfaz.

Este tipo de pruebas tiene como objetivo comprobar que la pantalla desarrollada cumple con los requisitos.

- **Verificación de pantallas.**

A continuación, se describe una tabla en la cual se puede verificar que el sistema posee las mismas pantallas que se definieron en los requisitos.

Tabla XXIV. Verificación de Pantallas

Requisito	Pantalla del sistema	Cumplimiento
Inicio de Sesión	Formulario de Inicio de sesión.	✓
Control de Actuadores y lectura de sensores.	Sección de Control y de lectura de sensores.	✓
Notificaciones	Sección de Notificaciones.	✓

6.4.2.2 Pruebas de Soporte.

Para comprobar que la aplicación realiza un adecuado control sobre los errores comunes y presentados al usuario.

PS01: Errores de datos de entrada

Para comprobar que la aplicación realiza un adecuado control sobre la información ingresada hacia esta, se desarrollaron pruebas específicas sobre las pantallas que poseen entrada de datos.

En la aplicación la pantalla que posee ingreso de datos es el inicio de sesión.

A continuación, un ejemplo de control que se realiza para verificar que los campos de entradas sean llenados correctamente.

- ✓ **Error al enviar formulario con datos vacíos:** en este caso la aplicación pide completar los campos (Ver Figura 66.).

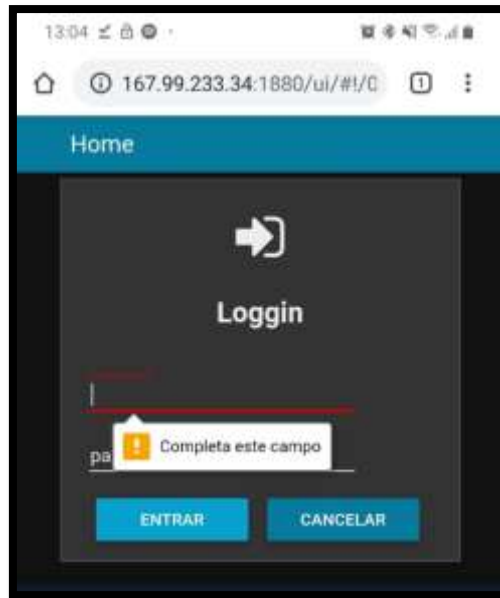


Figura 66. Prueba Soporte datos entrada 1

- ✓ **Error al ingresar usuario:** en este caso el sistema notifica que la autentificacion ha fallado y que dicho usuario no existe (Ver Figura 67.) .

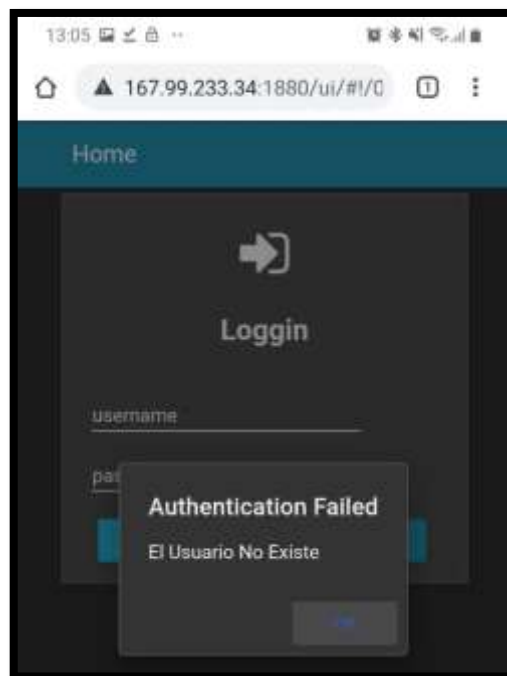


Figura 67. Prueba de soporte de datos de entrada 2

- ✓ **Error al ingresar contraseña:** en este caso el sistema notifica que la contraseña ingresada es invalida por lo que debería ingresarla nuevamente (Ver Figura 68.).

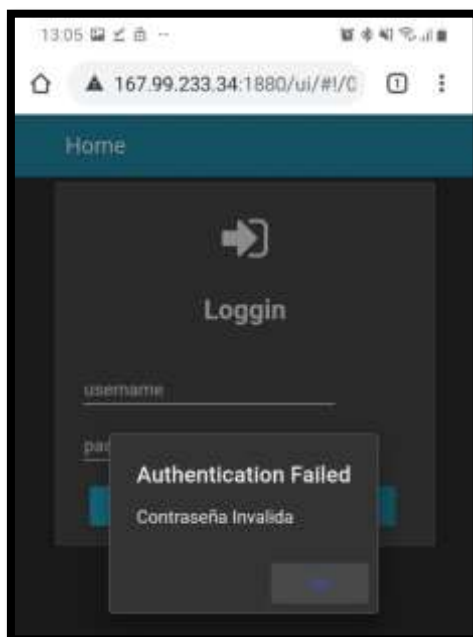


Figura 68. Prueba de soporte de datos de entrada 3

6.4.2.3 Pruebas de Funcionalidad.

En cuanto a las pruebas funcionales se procedió a verificar los ítems descritos en los criterios de aceptación del plan de pruebas.

- **Inicio de sesión.**

La ejecución de inicio de sesión se muestra en la Figura 69.

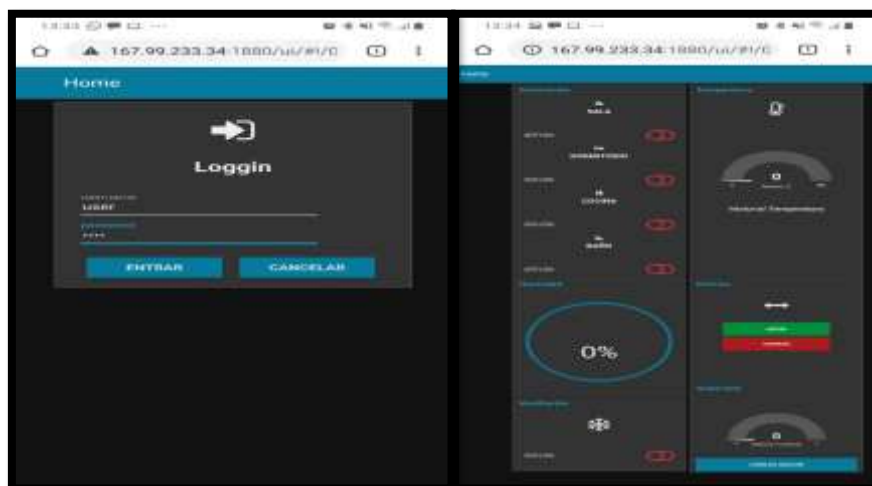


Figura 69. Caso de Prueba Inicio de Sesión

- **Cierre de sesión:** La ejecución del cierre de sesión se muestra en la Figura 70.

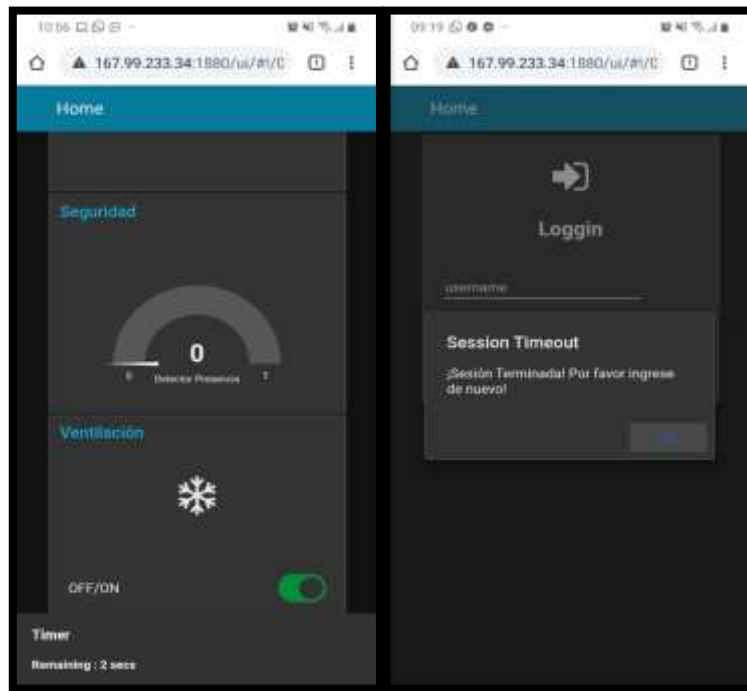


Figura 70. Caso de Prueba Cierre de Sesión

- **Sistema en uso:** La ejecución de cantidad de usuarios por sesión se muestra en la Figura 71.

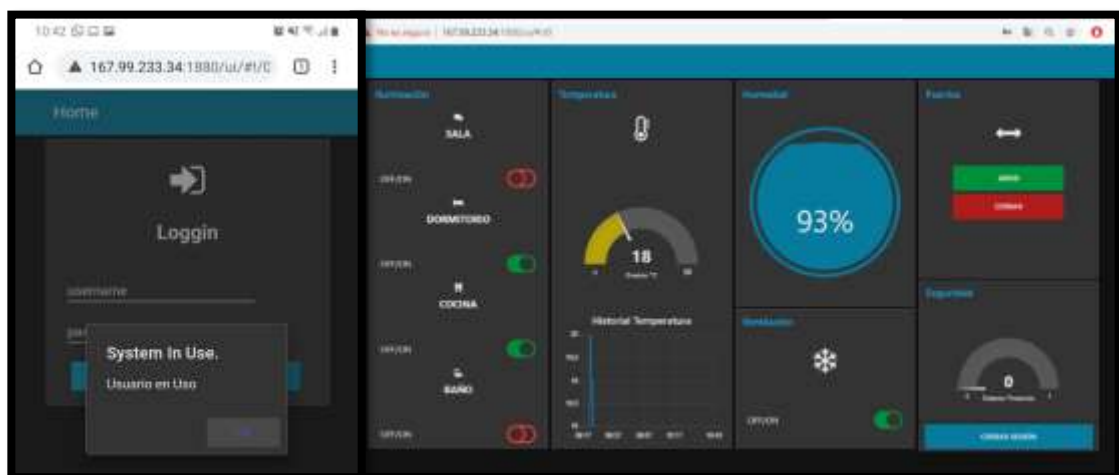


Figura 71. Caso de Prueba cantidad de Usuarios por sesión

- **Encendido y Apagado de luces:** La ejecución encendido y apagado de luces se muestra en la Figura 72.

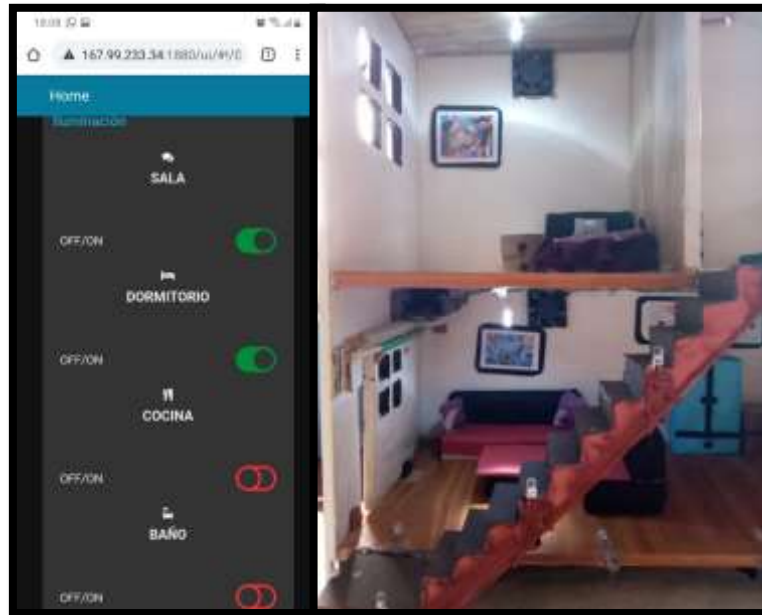


Figura 72. Caso de Prueba Encendido y Pagado de luces

- **Encendido y apagado de ventiladores:** La ejecución de encendido y apagado de luces se muestra en la Figura 73.

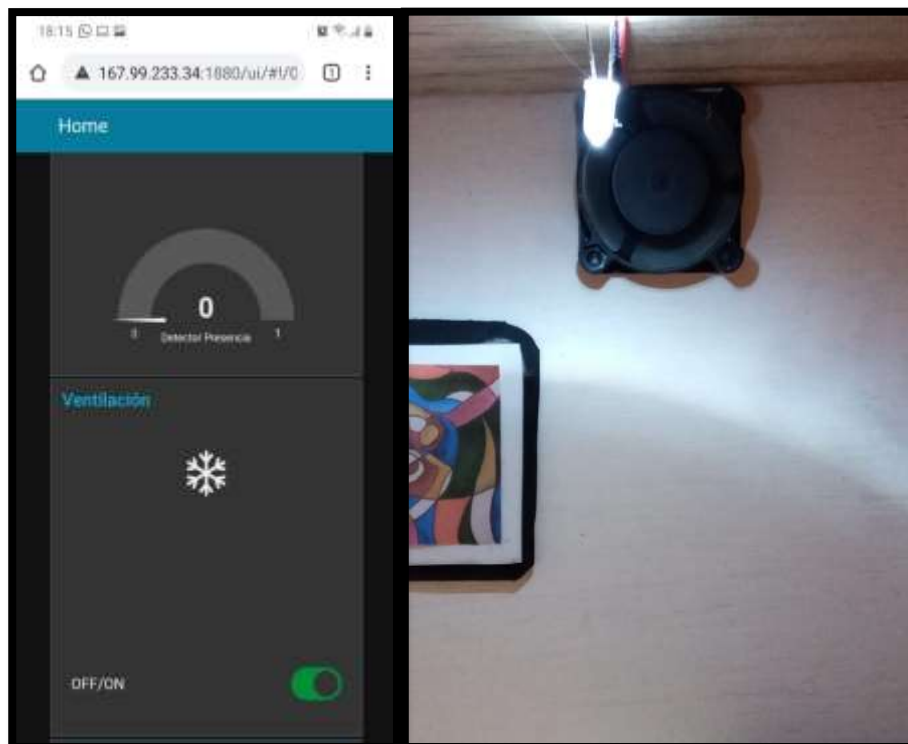


Figura 73. Caso de Prueba Encendido y Apagado de Ventiladores

- **Apertura y cierre de puertas:** La ejecución del cierre y apertura de la puerta se muestra en Figura 74.

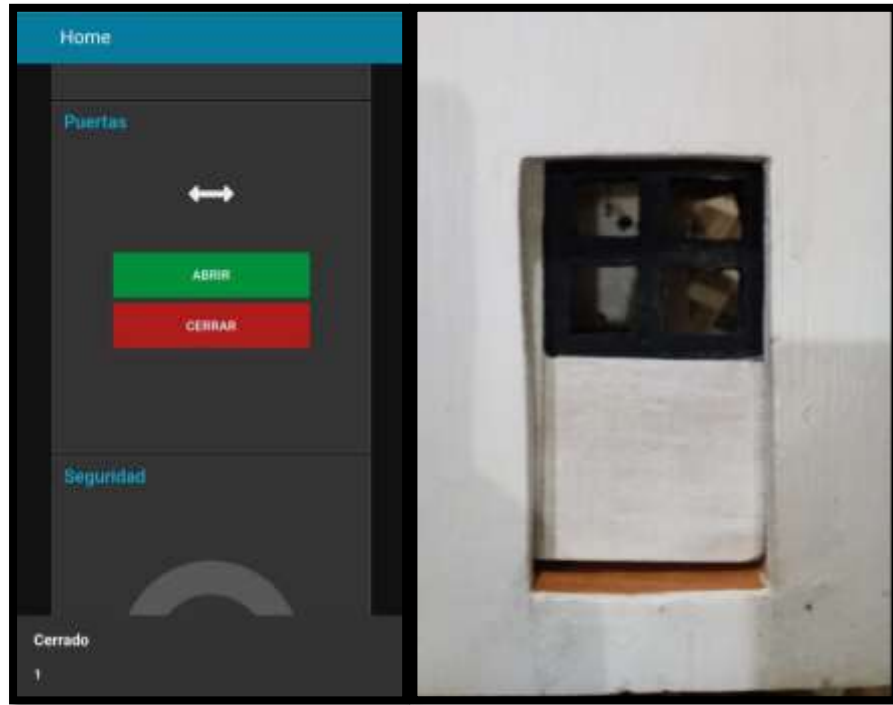


Figura 74. Caso de Prueba Apertura y Cierre de Puerta

- **Lectura de Temperatura y Humedad Ambiente:** La lectura de humedad y temperatura ambiente se muestra en Figura 75.



Figura 75. Caso de Prueba Lectura de Temperatura y Humedad Ambiente

- **Gestión de seguridad:** Los eventos de la gestión de seguridad se muestra en la Figura 76. y Figura 77.

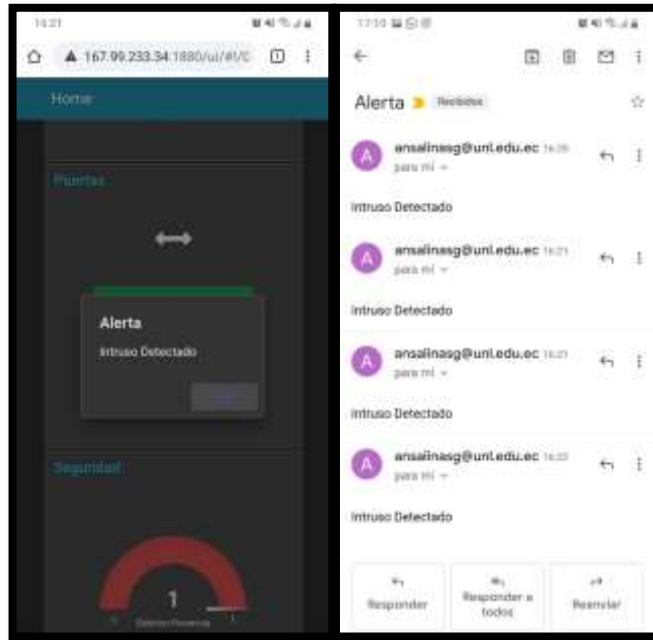


Figura 76. Caso de Prueba Gestión de Seguridad 1

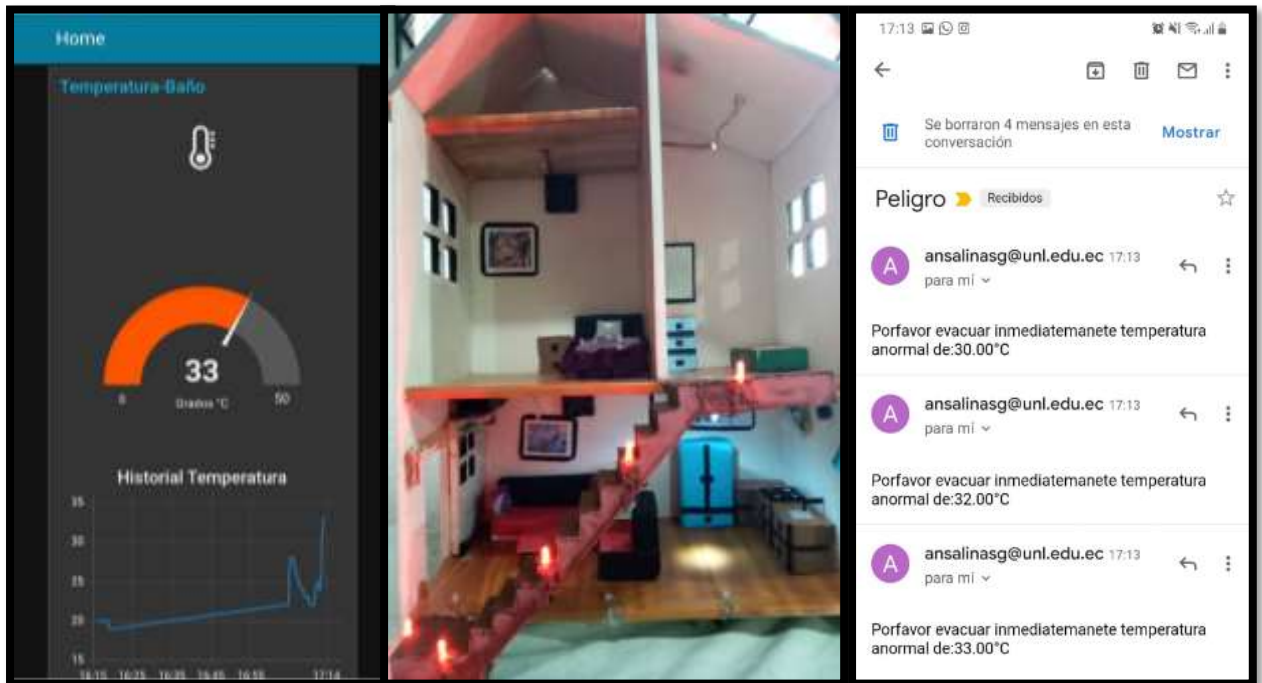


Figura 77. Caso de Prueba Gestión de Seguridad 2

6.4.3 Resultados de las Pruebas.

En la siguiente tabla se detallan los resultados de las pruebas ejecutadas según los casos de prueba.

Caso de Prueba	Requerimiento	Criterio de Aceptación	Resultado
CP01	RF001	CR01	Exitoso
CP02	RF002	CR02	Exitoso
CP03	RF003	CR03	Exitoso
CP04	RF004	CR04	Exitoso
CP05	RF005	CR05	Exitoso
CP06	RF006	CR06	Exitoso
CP07	RF007 –RF008	CR07	Exitoso
CP08	RF009	CR07	Exitoso

Luego de hacer ejecutado todos los casos de prueba se concluye que cada una de las pruebas realizadas culminaron con éxito total en todos los ítems dados, así como resultado en una versión estable de la arquitectura informática.

6.4.3.1 Pruebas de Usabilidad.

Una forma de definir si un sistema de software es usable, es evaluando si cumple con las características de usabilidad. La usabilidad se define como el enlace de un producto o servicio usado por usuario específicos para lograr un objetivo con efectividad, eficiencia y satisfacción en un contexto específico. Las pruebas de usabilidad comprenden tres técnicas que permiten medir la efectividad, eficiencia y satisfacción.

Pruebas con usuarios: permite que ciertos usuarios realicen tareas específicas para que a partir de sus ejecuciones evalúen la usabilidad y se identifiquen errores.

Observación: Permite visualizar las actividades de los usuarios mediante la grabación de sus ejecuciones u observación dedicada.

Pregunta a los usuarios: mediante el uso de cuestionarios o entrevistas sobre la satisfacción y opinión en el uso del sistema.

Basándonos en lo antes mencionado se procede a evaluar la usabilidad del sistema domótico considerando los parámetros de eficiencia y eficacia.

Para ello primeramente se establece la muestra a ser evaluada.

Para escoger la muestra se escogió al azar 15 personas de diferentes viviendas del sector Época de la ciudad de Loja. Se justifica este tamaño de la muestra por la dificultad de trasladar el prototipo a muchas más viviendas para su respectiva demostración.

Luego de definir la muestra se procedió a diseñar la encuesta (Ver Anexo 6) para ello se utilizó la herramienta en línea de google que permite crear formularios totalmente gratis, así como visualizar los resultados.

Los casos de pruebas a utilizar son los especificados en las pruebas de funcionalidad ya que para probar características de usabilidad no se requieren de caos de pruebas específicos.

Luego de aplicar la encuesta a 15 personas se obtuvieron los siguientes resultados.

- **¿Cómo considera la interfaz web del sistema domótico? (ej.: botones, zonas de contenido, colores, etc.).**

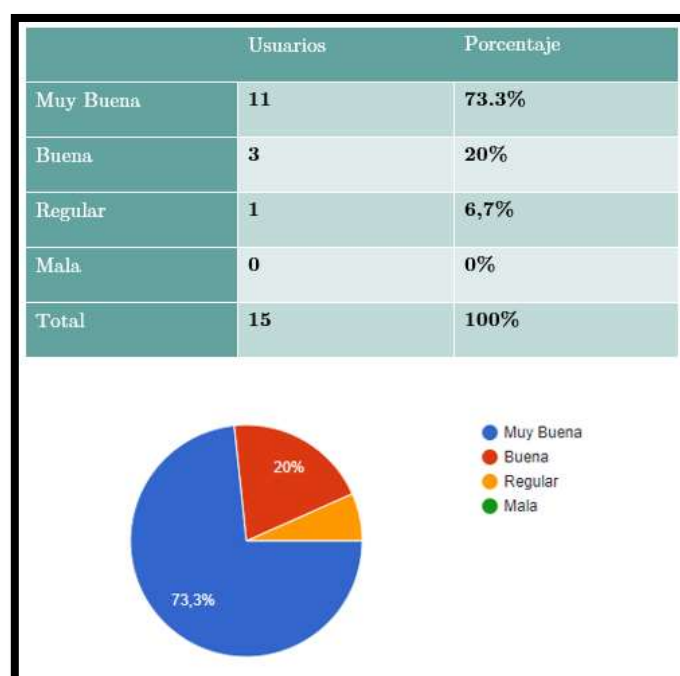


Figura 78: Resultados Evaluación del sistema Pregunta 1

Del 100% de encuestados el 73.3 % manifiesta que la interfaz web del sistema domótico es muy buena, el 20 % manifiesta que es buena y el 6,7% dice que es regular (Ver Figura 78). Por lo tanto, se llega a la conclusión de que a la mayoría le gusta la interfaz gráfica presentada.

- **Accesibilidad:** ¿Las acciones que solicita el sistema domótico (activación de luces, apertura de puerta, encendido de ventiladores, etc.) son fáciles de ejecutar?

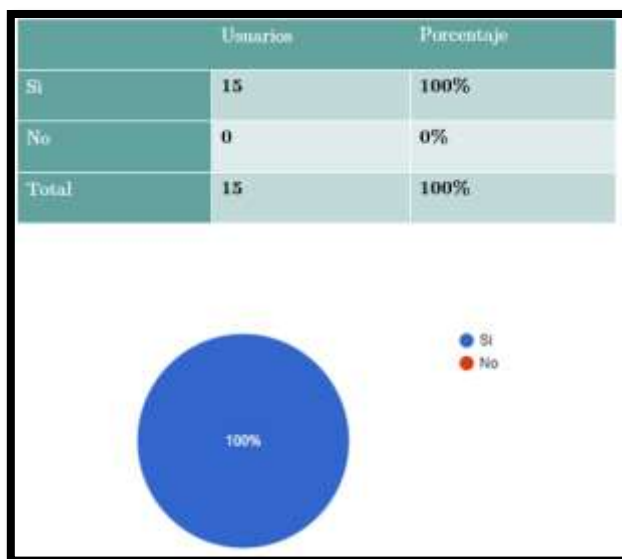


Figura 79: Resultados Evaluación del sistema Pregunta 2

Del total de la población encuestada el 100% manifiesta que las acciones que solicita el sistema domótico son fáciles de ejecutar por lo cualquier persona podría utilizarlo (Ver Figura 79).

- **Sistema de Identificación:** Se identifican fácilmente las figuras, hipertextos, las zonas activas y el tipo de acción que se debe ejecutar en el sistema domótico.

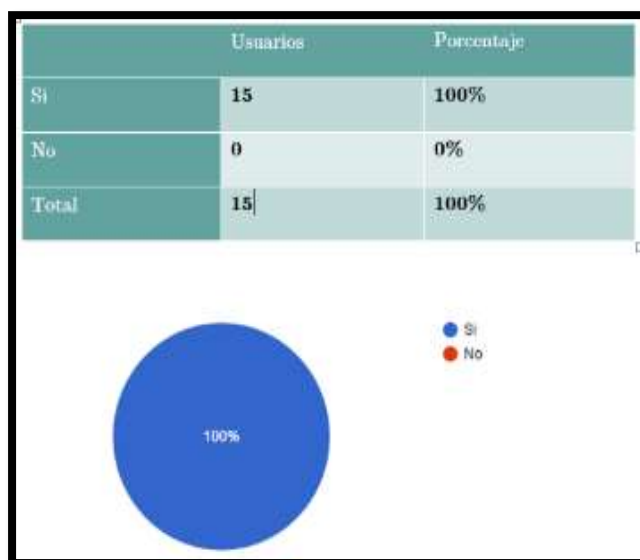


Figura 80: Resultados Evaluación del sistema Pregunta 3

- **Fiabilidad del sistema:** Se presentaron errores durante la operación del sistema domótico.

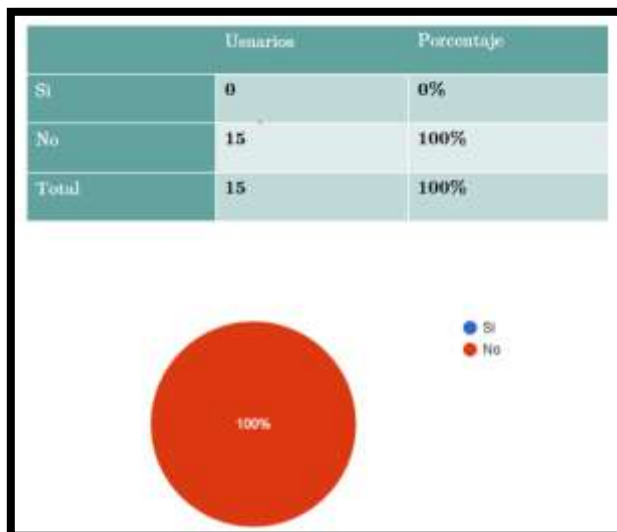


Figura 81: Resultados Evaluación del sistema Pregunta 4

De los 15 usuarios encuestados el 100% de ellos manifiestan que no tuvieron ningún error al ejecutar alguna de las acciones existentes en el sistema domótico (Ver Figura 81). Por lo que se supone que tiene un nivel elevado de confiabilidad.

- **Intuición:** Los procedimientos de navegación por el sistema domótico, así como la ejecución de tareas son fáciles de realizar:

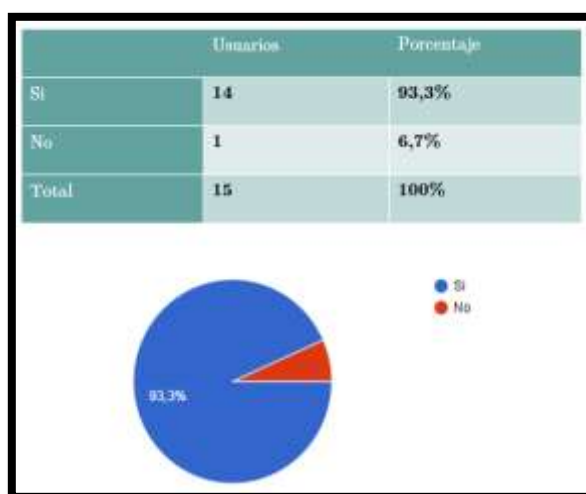


Figura 82: Resultados Evaluación del sistema Pregunta 5

Del total de 15 personas encuestadas el 93,3 % de estas asegura que la navegación y ejecución de tareas son fáciles de realizar en el sistema domótico y el restante manifiesta

que se les dificulta realizar las tareas (Ver Figura 82). Dicho esto, se concluye que casi en su totalidad podrán utilizar el sistema domótico propuesto.

- **¿Cómo considera la densidad del contenido que se presenta en el sistema Domótico?**

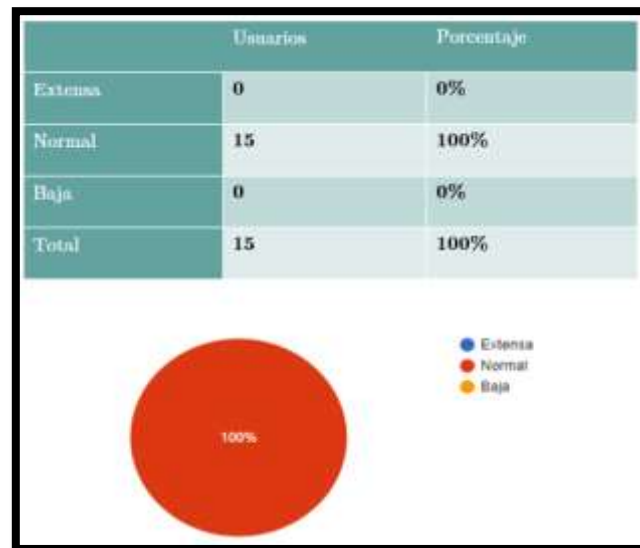
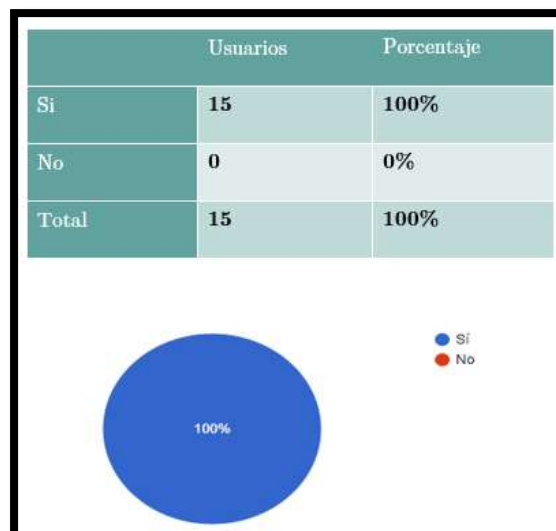


Figura 83: Resultados Evaluación del sistema Pregunta 6

De las encuestas realizadas a los usuarios el 100% manifiesta que la densidad del contenido que presenta el sistema domótico como por ejemplo botones, gráficos, cuadros etc., (Ver Figura 83). Es normal por lo que no existe aglomeración de información o tareas que dificulten la operatividad del mismo.

- **¿La información que se presenta en el sistema domótico es fácil de entender y memorizar?**



De las encuestas aplicadas el 100% manifiesta que la información presentada por el sistema domótico es fácil de entender y memorizar de tal manera que con cada vez que ingresen al sistema les será más fácil realizar una tarea (Ver

del sistema Pregunta 7).

- **¿Le gustaría implementar este sistema Domótico en su Hogar?**

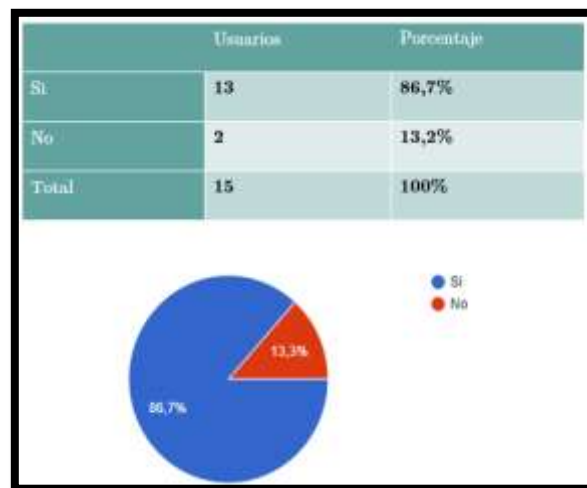


Figura 85: Resultados Evaluación del sistema Pregunta 8

Del 100% de encuestados el 86.7% manifiesta que le gustaría implementar este sistema domótico en sus hogares y el 13,2% asegura que no (Ver Figura 85). Dicho lo anterior se concluye que tiene un nivel de aceptabilidad alto.

- **¿Cómo Calificaría el Sistema Domótico Propuesto?**

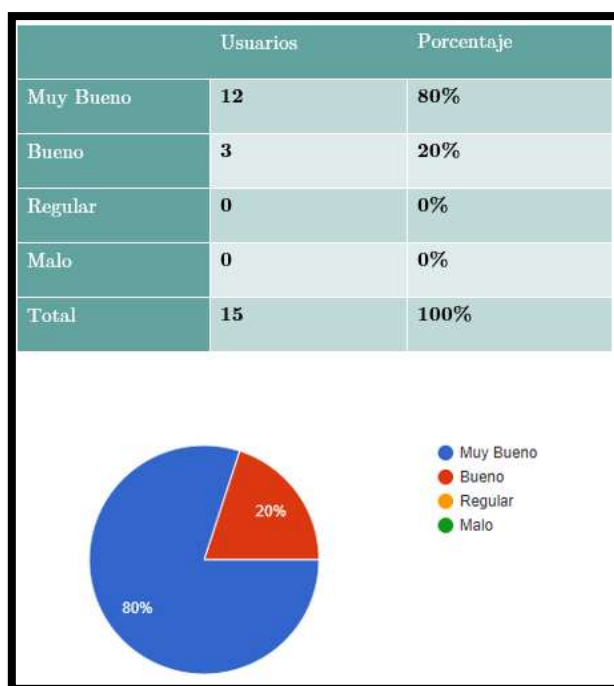


Figura 86: Resultados Evaluación del sistema Pregunta 9

De las encuestas realizadas a los usuarios el 80 % de ellos califican al sistema propuesto como Muy bueno y el 20 % lo califica como Bueno (Ver Figura 86) por lo que se concluye que el sistema domótico tiene una gran acogida por parte de ellos considerándolo óptimo para mejorar su calidad de vida.

- **¿Escriba una opinión general sobre el sistema domótico o Sugerencias?**

En cuanto a la pregunta sobre escriba una opinión o sugerencia sobre el sistema domótico los usuarios manifestaron textualmente que les parece interesante y que la mayoría de persona deberían implementar este sistema porque les da facilidad de acceder y controlar todos desde fuera y dentro de casa, que les permite controlar y darles seguridad a sus hogares, también manifiestan que la ayuda de esta tecnología se puede ayudar a que la vida cotidiana sea más cómoda y con beneficios positivos, otra opinión es que es muy interesante y que sería conveniente implementarlo tanto en viviendas como en empresas y centros educativos ya que estas funciones serian de gran utilidad. Estas son algunas de las opiniones positivas recibidas por los usuarios. Por otro lado, también se recibió algunas sugerencias entre ellas tenemos que sería indispensable que el sistema sea tolerante a fallos como la presentación de algún corte de energía eléctrica, además que sería aún más interesante la implementación de las cámaras para verificar que cada una de las acciones que se realizan desde el sistema son ejecutadas correctamente cuando se encuentre fuera de su hogar, y por ultimo hubo un usuario el cual manifestó que al sistema

lo utilizaría solo cuando se encuentre dentro o cerca de ella por no puede confiar en que el sistema va a funcionar al cien por ciento.

7. DISCUSIÓN

7.1 Desarrollo de la propuesta alternativa

El presente Trabajo de Titulación denominado "Desarrollo De Una Solución Informática Que Permita Administrar Un Hogar Inteligente Haciendo Uso De Hardware Libre" se culminó desarrollándolo y ejecutando el prototipo.

Con la aplicación de métodos, técnicas y la metodología, se pudo concluir en su totalidad con el objetivo general y los objetivos específicos, a continuación, se detalla el cumplimiento de cada uno:

Objetivo 1: Realizar Una Comparación Técnica De Los Diferentes Dispositivos De Hardware Libre Y Software Aplicado A Domótica.

Antes de realizar una comparación técnica tanto de hardware como software se procedió a analizar sobre las diferentes tecnologías que existen hoy en día para automatizar un hogar, Asimismo se determinó y documento los requisitos con los cuales va a contar el presente sistema, una vez identificados estos requisitos se procedió a realizar una búsqueda y recopilación de información sobre softwares de código abierto más relevantes para desarrollo de aplicaciones IoT y por ende para automatizar un hogar. Por otra parte, se realizó una búsqueda de información sobre hardware Open-Source para el desarrollo de aplicaciones IoT. Después de realizar esta búsqueda de información se efectuó una comparación técnica tanto de software como de hardware para seleccionar las herramientas más idóneas que ayudaran al desarrollo de la solución informática.

Al finalizar esta comparación técnica se llegó a la conclusión que la mejor opción para desarrollar el sistema domótico es el microcontrolador NodeMCU ESP8266 ya que permite una comunicación mediante Wi-Fi y cuenta con pines GPIO que ayudarán a la conexión de los sensores y actuadores. Además, se determinó el uso del protocolo MQTT para las comunicaciones; siendo este un protocolo óptimo que opera bajo recursos limitados como: el ancho de banda y consumo de energía, igualmente, para la integración del hardware con el software se utilizó la plataforma de programación visual Node-RED debido a que es fácil de usar e integra un sin número de protocolos entre ellos el MQTT. Por otro lado, permite ver el flujo que tendrá nuestro sistema, asimismo, integra plugins de terceros que ayudarán a extender las funcionalidades del sistema sin problema alguno.

Todo lo antes mencionado se detalla en la Sección de Resultados (Ver Tablas XVII, XIX, XX, XXI). Primera Fase: Realizar una comparación técnica de los diferentes dispositivos de Hardware Libre y Software aplicado a Domótica.

Objetivo 2: Desarrollar el módulo que permita la integración y administración de los componentes eléctricos y electrónicos del prototipo de la vivienda inteligente.

Para el desarrollo del módulo que permitan integrar los componentes eléctricos y electrónicos del prototipo del hogar inteligente, se usó como base la metodología en V propuesta por A. Martínez para el desarrollo de sistemas que integran hardware y software, el cual pretende dar solución al problema en donde resulta ineludible la interacción de ambos. El presente trabajo se basó en esta metodología puesto que no existe una metodología estándar que ayude al desarrollo de estos sistemas. Una vez determinada la metodología se procedió a diseñar los circuitos que se codificaran y luego se cargaran en la placa NodeMCU. Para conseguir este diseño se hizo uso del programa Fritzing mismo que permitió pasar de prototipos visuales a un producto final simulado en la maqueta. Para lograr lo dicho con anterioridad se procedido a programar componente por componente de tal manera que se vaya probando el funcionamiento de cada uno de estos para finalmente integrarlos formando un solo modulo. Para la codificación e implementación se utilizó algunas herramientas como el IDE de desarrollo Arduino, la placa NodeMCU, sensores y actuadores. Adicional se hizo uso de algunas librerías que permitieron la comunicación entre éstos dispositivos con el Bróker MQTT, otras librerías para poder programar en la placa NodeMCU ESP8266 y también para la lectura del sensor de temperatura y Humedad DHT11. Una vez programado y cargado el código a la placa se procedió a verificar por medio del puerto COM08 el funcionamiento de los mismos luego, se verifica por medio de la consola si las placas reciben y envían información. En el desarrollo de este proceso se encontró algunos problemas en cuanto a la modificación de los actuadores, pero fueron solucionados satisfactoriamente.

Objetivo 3: Desarrollar un prototipo para la solución informática que permita administrar la programación de eventos para el hogar inteligente.

Para cumplir este objetivo primeramente se instaló y configuro la plataforma Node-Red. Posterior a ello, se procedió al desarrollo del cliente MQTT; para esto, fue necesario instalar paquetes adicionales necesarios para crear la interfaz gráfica. Luego de la instalación de estos paquetes se inició con la creación del flujo principal del sistema, es

decir, la autenticación; para esto se creó varias funciones que sirvieron para controlar el acceso de los usuarios permitiendo así, que una sola persona pueda manipular el sistema mientras se encuentre autenticada. Una vez terminado el flujo de la autenticación se procedió a la creación del resto de flujos que permitirán el control total de los actuadores y sensores que integran el sistema. Para la creación de estos flujos se realizó la configuración de algunos aspectos como son el servidor MQTT el cual va a enviar y a la vez recibir información, también configurar el nivel de QoS del Protocolo MQTT y los diferentes tópicos en los cuales se va hacer una suscripción o una publicación. Una vez terminado todo este proceso para todos los actuadores y sensores se prosiguió a integrarlos dentro de la autenticación anteriormente creada.

Objetivo 4: Probar y evaluar el funcionamiento del prototipo del sistema de control del hogar inteligente en un entorno simulado.

Para realizar las pruebas del sistema domótico primeramente se buscó trabajos relacionados en los que hayan hecho pruebas de sistemas semejantes. Luego se diseñó la prueba para el sistema basándose en las pruebas que han ejecutado en dichos trabajos. Seguidamente se creó una tabla en la cual se especifica primeramente el nombre de la prueba, luego el requerimiento al que hace referencia, después el procedimiento de la prueba, asimismo, el estado inicial y final del componente al que se le realizara la prueba; y por último el resultado de la prueba. Previo a esto, llevamos a cabo cada una de las pruebas según los requerimientos funcionales y los no funcionales.

Para llevar a cabo lo dicho con anterioridad se conecta el microcontrolador y se levanta el servidor donde se encuentra alojada la aplicación. Luego aplicamos las pruebas según los requerimientos. Una vez ejecutada la prueba se procede a documentar el resultado de cada una de las estas. Finalmente, se ejecuta la evaluación de la aplicación. Para realizar este proceso se hizo uso de un cuestionario, el cual se lo aplicó a una muestra de 15 personas de diferentes viviendas. Este proceso se lo aplico realizando una demostración del funcionamiento del sistema en el entorno simulado y una vez explicado el funcionamiento y puesto en marcha el manejo del mismo por parte de los usuarios. Luego se procedió a aplicarles la encuesta en la cual el usuario podía expresar su nivel de satisfacción al interactuar con un sistema domótico; así mismo, podía calificar la estructura, operación y contenido del mismo, por otro lado, podía dejar una recomendación u opinión general del mismo de tal manera que ayude a mejorar dicho

sistema y medir el nivel de impacto del mismo. En este proceso se evidenció que parte de las personas se mostraron muy impresionados del avance de la tecnología; y de la misma forma, otro grupo de personas manifestaron que no podían confiar ciegamente en la tecnología por lo cual no dejarían el sistema tradicional de manejo de actividades dentro del hogar, y que se debería implementar más funcionalidades que aumente el nivel de confiabilidad; entre estas sugerencias están: implementar el servicio de streaming, aumento de sensores y un sistema de generación de energía adicional que se active cuando la energía eléctrica esté ausente.

Finalmente, en base a lo descrito se puede contestar la pregunta de investigación **“Haciendo Uso De Nuevas Tecnologías Y Con La Implementación De Elementos De Domótica, Es Posible Proporcionar A La Ciudadanía Un Sistema Configurable Y De Bajo Costo Para Administrar Un Hogar Inteligente”**. Se puede observar que es posible brindar una solución informática de bajo costo para poder administrar un hogar inteligente a la ciudadanía en general sin importar su estatus económico. Información que fue estimada de acuerdo a la construcción del prototipo. A más de ello, los resultados obtenidos corroboran que los objetivos planteados fueron alcanzados y cumplidos en su totalidad, logrando en su conjunto la culminación exitosa del Trabajo de Titulación.

7.2 Valoración técnica económica ambiental

El desarrollo del presente trabajo de titulación implicó una inversión económica, puesto que se utilizó recursos para realizar los objetivos planteados en este trabajo, en la Tabla XXV se detallan los costos de los recursos utilizados.

Tabla XXV. Valoración Económica del Proyecto

Recurso Humano			
Equipo de Trabajo	Horas	Precio/Hora	Valor Total
Alex Salinas	960	\$ 5.00	\$4.800,00
Director	20	\$ 15.00	\$ 300,00
Subtotal			\$ 5.100,00
Recurso Hardware			

*"Desarrollo De Una Solución Informática Que Permita Administrar
Un Hogar Inteligente Haciendo Uso De Hardware Libre"*

Descripción	Cantidad	Valor U.	Valor Total
Computadora	1	\$800,00	\$800,00
Impresora	1	\$60,00	\$60,00
NodeMCU	2	\$10,00	\$20,00
Sensor DHT11	1	\$3,50	\$3,50
Sensor PIR	1	\$4,00	\$4,00
Motor	1	\$1,00	\$1,00
Módulo Relé	2	\$4,00	\$8,00
Módulo L298N	1	\$4,00	\$4,00
Ventiladores	2	\$1,00	\$2,00
Leds	4	\$0,25	\$1,00
Subtotal			\$903,50
Recurso Software			
Descripción	Cantidad	Valor U.	Valor Total
Node-Red	1	\$0,00	\$0,00
IDE de Desarrollo Arduino	1	\$0,00	\$0,00
Mosquitto	1	\$0,00	\$0,00
Fritzing	1	\$0,00	\$0,00
MobaXterm	1	\$0,00	\$0,00
Subtotal			\$0,00
Recursos Varios			
Descripción	Cantidad	Valor U.	Valor Total
Internet	50	\$0.60	\$30,00
Transporte	30	\$0.30	\$9,00
Maqueta	1	\$30,00	\$30,00
Protoboard	2	\$4,00	\$8,00
Cable de Prototipado	30	\$0.25	\$7,50
UPS	1	\$10,00	\$10,00

*"Desarrollo De Una Solución Informática Que Permita Administrar
Un Hogar Inteligente Haciendo Uso De Hardware Libre"*

Subtotal	\$325,50
TOTAL	\$6.329,00

El coste del proyecto corresponde con el tiempo de ejecución del mismo, cabe mencionar que los costes fueron solventados por el autor de este trabajo.

8. CONCLUSIONES

Para finalizar el presenta trabajo de titulación, se describen las siguientes conclusiones obtenidas.

- Se desarrolló el sistema domótico utilizando hardware libre como el NodeMCU Basado el en Chip ESP8266 que nos permite crear proyectos IoT de la manera más sencilla posible, reduciendo significativamente el costo.
- Para la comunicación del hardware con el software se utilizó el protocolo MQTT con el fin de que el sistema domótico opere bajo recursos limitados como el ancho de banda, CPU y memoria RAM.
- Para la elaboración del cliente MQTT se utilizó Software OpenSource como NodeRED el cual permitió integrar el hardware con el Software de manera rápida y sencilla, ya que su programación es basada en flujos lo que permitió tener una estructura ordenada de los procesos existentes en la aplicación.
- Para el diseño y desarrollo del sistema domótico se empleó la metodología propuesta por A. Pérez misma que permitió determinar las necesidades y definir los requerimientos de los usuarios de manera más eficiente y adecuada ya que fue idónea para este tipo de proyectos en los cuales se integra hardware y software.
- El sistema domótico propuesto ayuda a mejorar la seguridad personal y de bienes debido a que, si se presenta alguna actividad anormal dentro de la vivienda, el sistema notifica o alerta de modo que podemos actuar inmediatamente con la ayuda de los actuadores que se activan para contrarrestar la emergencia.

9. RECOMENDACIONES

En esta sección se definieron algunas recomendaciones que sean de ayuda para la realización del presente proyecto.

- Para el diseño e implantación de un sistema domótico se recomienda detallar bien que elementos son los que se pretende controlar y monitorear de tal manera que se indague sobre los beneficios de los distintos usuarios y buscar soluciones acordes con el presupuesto que se cuente.
- Para la implantación de un sistema domótico en un entorno simulado se recomienda que este sea de un tamaño considerable para que no exista una aglomeración de cableado que provoque confusión ante alguna falla. Además, para exista comodidad a la hora de instalar los actuadores.
- Se recomienda agregarle a la aplicación Web tonos de colores llamativos, imágenes, sonidos, notificaciones de tal manera que cualquier persona pueda utilizar y entender el sistema.
- Se recomienda utilizar APIs de terceros para agregar más funcionalidades al sistema propuesto como por ejemplo notificaciones mediante una red social e inteligencia artificial.
- Si se quiere ampliar el número de sensores y actuadores al sistema se recomienda utilizar un Arduino Mega conjuntamente con el Chip ESP8266 ya que este ofrece una cantidad considerable de pines GPIO por lo que podemos conectar más de estos o también se puede hacer uso de la Raspberry Pi con la cual se puede integrar el servicio de cámaras por su facilidad de implementación.

10. BIBLIOGRAFÍA

- [1] M. Sánchez and G. Ramoscelli, "Creación de valor a partir del Internet de las cosas: Estudio exploratorio en la Provincia de Buenos Aires," *Visión Futur.*, vol. 22, no. 1, pp. 0–0, 2018.
- [2] C. L. Rose Karen, Eldridge Scott, "LA INTERNET DE LAS COSAS — LA INTERNET DE LAS COSAS — UNA BREVE RESEÑA," 2015.
- [3] P. S. Mendoza, K. Á. Hernández, C. V. Núñez, and D. J. Molinares, "Internet de las cosas y la salud centrada en el hogar," *Salud Uninorte*, vol. 32, no. 2, pp. 337–351, 2016.
- [4] D. Evans, "Internet de las cosas Internet de las cosas Cómo la próxima evolución de Internet lo cambia todo," 2011.
- [5] M. Leki and G. Gardaševi, "IoT sensor integration to Node-RED platform," no. March, pp. 21–23, 2018.
- [6] A. García, "Plataforma domótica basada en Raspberry Pi y el protocolo MQTT," *Esc. Técnica Super. Ing. Informática y Telecomunicación*, 2017.
- [7] S. Chanthakit and C. Rattanapoka, "Mqtt based air quality monitoring system using node MCU and node-red," *Proceeding 2018 7th ICT Int. Student Proj. Conf. ICT-ISPC 2018*, pp. 1–5, 2018.
- [8] G. Charry, "Diseño e implementación de un sistema domótico, monitoreado y controlado remotamente en tiempo real mediante una aplicación android para smartphone," p. 94, 2016.
- [9] R. Sagar, A. Sonawane, M. S. Pawar, M. V. Kadam, and A. Waghela, "IOT based Home Automation and Security System," *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 6, no. 3, pp. 821–824, 2017.
- [10] P. A. Semle, "Protocolos IIoT para considerar," pp. 32–35, 2016.
- [11] A. González, "IoT : Dispositivos , tecnologías de transporte y aplicaciones," 2017.
- [12] B. Cabé, "IoT Developer Survey 2018," *Eclipse Found.*, no. April, p. 51, 2018.
- [13] D. Kumar, R. K. Maurya, and K. Dwivedi, "Iot based home automation using Node-RED," *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 12, pp. 5044–5047, 2019.

- [14] G. R. N. M. S. P. Santhoshkumar, "IoT using Data Protocols: Issues and Performance Rathinam Technical Campus , Coimbatore , India," vol. 5, no. 10, pp. 735–739, 2017.
- [15] J. Dizdarević, F. Carpio, A. Jukan, and X. Masip-Bruin, "A survey of communication protocols for internet of things and related challenges of fog and cloud computing integration," *ACM Computing Surveys*, vol. 51, no. 6. pp. 1–30, 2019.
- [16] "Jeedom." [Online]. Available: <https://www.jeedom.com/site/fr/>. [Accessed: 17-Oct-2019].
- [17] I. Froiz-Míguez, T. M. Fernández-Caramés, P. Fraga-Lamas, and L. Castedo, "Design, implementation and practical evaluation of an iot home automation system for fog computing applications based on MQTT and ZigBee-WiFi sensor nodes," *Sensors (Switzerland)*, vol. 18, no. 8, pp. 1–42, 2018.
- [18] "Domoticz." [Online]. Available: <https://www.domoticz.com/>. [Accessed: 17-Oct-2019].
- [19] "Home Assistant." [Online]. Available: <https://www.home-assistant.io/>. [Accessed: 17-Oct-2019].
- [20] "Node-RED." [Online]. Available: <https://nodered.org/#features>. [Accessed: 23-Oct-2019].
- [21] I. González, J. González, and F. Gómez-arribas, "Hardware libre : clasificación y desarrollo de hardware reconfigurable en entornos GNU / Linux," no. February 2015.
- [22] V. Suárez and G. Hacklab, "Introducción a Raspberry Pi."
- [23] P. Jos and C. Claudel, "Raspberry Pi," 2016.
- [24] D. Cuartielles, "Arduino."
- [25] R. K. Kodali and S. R. Soratkal, "MQTT based home automation system using ESP8266," in *IEEE Region 10 Humanitarian Technology Conference 2016, R10-HTC 2016 - Proceedings*, 2017, no. December 2016.
- [26] Pycom Lopy, "Adafruit Industries, electrónica y kits de bricolajé unicos y divertidos," *Adafruit*, 2017. [Online]. Available: <https://www.adafruit.com/about>.

- [Accessed: 17-Oct-2019].
- [27] Learn.adafruit.com, "Overview | Adafruit HUZZAH ESP8266 breakout | Adafruit Learning System," 2015. [Online]. Available: <https://learn.adafruit.com/adafruit-huzzah-esp8266-breakout>. [Accessed: 17-Oct-2019].
- [28] S. Neeraja and B. S. S. Tejesh, "Implementation of an Efficient Smart Home System using MQTT," *Int. Res. J. Eng. Technol.*, vol. 4, no. 3, pp. 1848–1852, 2017.
- [29] Y. Upadhyay, A. Borole, and D. Dileepan, "MQTT based secured home automation system," *2016 Symp. Colossal Data Anal. Networking, CDAN 2016*, 2016.
- [30] A. Agarwal, R. Singh, A. Gehlot, G. Gupta, and M. Choudhary, "IOT Enabled Home Automation through Nodered and MQTT," *Int. J. Control Theory Appl.*, vol. 1, p. 18, 2017.
- [31] J. J. Brun, "Smart Home usando IoT y Chatbots," 2018.
- [32] A. Perez, O. Berreteaga, A. R. De Olano, A. Urkidi, and J. Perez, "Una metodología para el desarrollo de hardware y software embebidos en sistemas críticos de seguridad," *5ta. Conf. Iberoam. en Sist. Cibern. e Informatica, CISCI 2006, Memorias*, vol. 1, pp. 297–302, 2006.
- [33] M. Martínez, "Desarrollo de proyectos IoT utilizando Raspberry Pi como plataforma," p. 134, 2017.
- [34] G. Rocher and J.-Y. Tigli, "Tutorial: MQTT (Message Queuing Telemetry Transport)," vol. 33, no. 0, pp. 1–11, 2017.
- [35] Mosquitto, "Eclipse Mosquitto," *Eclipse Mosquitto*, 2018. [Online]. Available: <https://mosquitto.org/>. [Accessed: 23-Oct-2019].
- [36] D. Wheat and D. Wheat, "Arduino Software," in *Arduino Internals*, 2011, pp. 89–97.
- [37] Friends-of-Fritzing foundation, "Fritzing Fritzing," *fritzing.org*, 2017. [Online]. Available: <https://fritzing.org/home/>. [Accessed: 23-Oct-2019].
- [38] "MobaXterm free Xserver and tabbed SSH client for Windows." [Online]. Available: <https://mobaxterm.mobatek.net/>. [Accessed: 23-Oct-2019].

- [39] "Arduino - Software." [Online]. Available: <https://www.arduino.cc/en/Main/Software#>. [Accessed: 21-Nov-2019].
- [40] I. GitHub, "GitHub - esp8266/Arduino: ESP8266 core for Arduino," 2018. [Online]. Available: <https://github.com/esp8266/Arduino>. [Accessed: 21-Nov-2019].

11. ANEXOS

Anexo 1: Especificación de Requerimientos.

ESPECIFICACIÓN DE REQUISITOS DE SOFTWARE SRS 830

Proyecto: Desarrollo de una solución informática que permita administrar un hogar Inteligente haciendo uso de hardware Libre.

El presente documento es aprobado por el cliente referenciando a continuación:

A handwritten signature in blue ink, appearing to read 'Gastón Chamba', is written over a horizontal line. The signature is stylized and cursive.

Ing. Gastón Chamba

**DOCENTE DE LA CARRERA
DE INGENIERIA EN SISTEMAS DE LA
FACULTAD DE ENERGÍA- UNL**

DOCUMENTO DE ESPECIFICACIÓN DE REQUERIMIENTOS DE SOFTWARE IEEE 830

INTRODUCCIÓN

- **Propósito**

El presente documento tiene como propósito definir las especificaciones funcionales, no funcionales para el desarrollo de una solución informática que permita administrar un hogar inteligente haciendo uso de hardware libre.

- **Ámbito del Sistema**

El proyecto trata de acercar la domótica a cualquier persona, sin importar su estatus social. Es por eso que el objetivo del siguiente proyecto es desarrollar un prototipo de sistema domótico utilizando herramientas gratuitas o de bajo coste como: NodeRED que es un software IoT que permitirá la conexión fácil con el hardware; y el microcontrolador NODEMCU ESP8266 que es un hardware que permitirá la comunicación y conexión de los actuadores y sensores. Con el fin de automatizar y controlar ciertas actividades o tareas que se llevan dentro del hogar como:

- Encendido y apagado de luces.
- Control de climatización.
- Apertura y cierre de puertas.
- Detección de intrusos.
- Lectura de temperatura y humedad.

- **Definición, Acrónimos y Abreviaturas**

La siguiente tabla describe las definiciones, acrónimos y abreviaturas claves que se van a utilizar en el documento.

Término	Significado
Node-Red	Es una herramienta de desarrollo basada en flujo para programación visual desarrollada originalmente por IBM para conectar dispositivos de hardware, API y servicios en línea como parte de Internet de las cosas

NodeMCU ESP8266	Es una plataforma IoT de código abierto. Incluye el firmware que se ejecuta en el SoC Wi-Fi ESP8266 y el hardware que se basa en el módulo ESP-12.
Usuario	Persona que utiliza el sistema
Administrador.	Es la persona que tiene la responsabilidad de implementar, configurar, mantener, monitorizar, documentar y asegurar el correcto funcionamiento de un sistema
Software	Soporte lógico de un sistema informático, que comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas, en contraposición a los componentes físicos que son llamados hardware.
Hardware	Se refiere a las partes físicas, tangibles, de un sistema informático, sus componentes eléctricos, electrónicos, electromecánicos y mecánicos
Domótica	Sistemas capaces de automatizar una vivienda o edificación de cualquier tipo, aportando servicios de gestión energética, seguridad, bienestar y comunicación
RF	Requisito Funcional
RNF	Requisito No Funcional
Open-Source	Es un modelo de desarrollo de software basado en la colaboración abierta.

- **Referencia.**

Estándar IEEE 830-1998

- **Visión General Del Documento**

El objetivo de este documento es dar a conocer una descripción completa del funcionamiento del sistema domótico. Se encuentra dividido en tres secciones. En la primera sección se realiza una introducción al documento en donde se detallan el propósito del documento, el ámbito del sistema, referencias y la visión general del documento. En la segunda sección se presenta una descripción general del sistema

domótico teniendo como puntos importantes la perspectiva del producto, Funciones del Producto, Características de los Usuarios, Restricciones y Dependencias. Finalmente, en la última sección se muestra una descripción de los requerimientos del sistema domótico.

- **Descripción General.**

- **Perspectiva del Producto.**

El sistema domótico estará desarrollado bajo herramientas de hardware y software Open-Source, así mismo estará implementado en un servidor virtual que estará disponible las 24 horas para su control y monitoreo del mismo.

- **Funciones del Producto.**

El sistema domótico desarrollado permitirá realizar las siguientes funcionalidades:

- Control de luces. - Para que el usuario controle la iluminación a distancia.
- Lectura de temperatura. - Alerta de incidencias, y para mantener un clima adecuado.
- Control de puertas. - Incrementación de seguridad de la vivienda en caso de una emergencia como incendios y control de la misma a distancia.
- Control de ventilación. - Permitiendo disfrutar de la temperatura ideal en todo momento.
- Detector de presencia. - Para detectar y alertar sobre intrusos en la vivienda.

- **Características del Usuario**

A continuación, la interacción del usuario con el Sistema Domótico.

Tipo de usuario	Usuario Normal
Formación	Ninguna formación en específico.
Habilidades	Conocer el manejo de las TIC's
Actividades	Utilizar el sistema domótico.

Tipo de usuario	Administrador
Formación	Técnico en Sistemas
Habilidades	Administración de Hardware y Software
Actividades	Administración General del Sistema.

- **Restricciones.**
 - Para la implementación del Sistema domótico se utilizará Herramientas Open-Source.
 - El Sistema Domótico será compatible con los navegadores Chrome, Firefox y Microsoft Edge.
 - La comunicación del sistema domótico será mediante el uso de un protocolo M2M.
 - El sistema domótico será alojado en un servidor virtual en la nube.
- **Suposiciones y Dependencias.**
 - **Suposiciones**

Se supone que los requisitos planteados son correctos y estables, lo cual garantizará un sistema de calidad y eficiente.
 - **Dependencias**

El funcionamiento del Sistema domótico depende de la conexión a internet estable y energía eléctrica.
 - **Requisitos**

En esta sección se detallan los requisitos que son necesarios para que el sistema funcione adecuadamente, los requisitos deben ser claros, no ambiguos para poder desarrollar un buen diseño y desarrollo del sistema.

- **Requisitos Funcional**

- **Requisito Funcional 1**

Requisito		RF001
Nombre del Requisito		Autenticación de Usuarios
Prioridad		Alta
Descripción		El sistema deberá presentar un formulario de autenticación. Debe presentar notificaciones en casos de usuario o contraseña incorrecta. Deberá contar con un botón de aceptar y otro de cancelar.

- **Requisito Funcional 2**

Requisito		RF002
Nombre del Requisito		Cierre de Sesión
Prioridad		Alta
Descripción		El sistema deberá controlar el tiempo de permanencia de usuarios. No siendo mayor a 5 min.

- **Requisito Funcional 3**

Requisito		RF003
Nombre del Requisito		Control de Cantidad de Usuarios que acceden al sistema al mismo tiempo.
Prioridad		Alta
Descripción		El sistema deberá controlar que solo un usuario pueda acceder a la vez. El sistema deberá mostrar notificación en caso de que el sistema esté en uso.

➤ **Requisito Funcional 4**

Requisito		RF004
Nombre del Requisito	del	Control de Iluminación
Prioridad		Alta
Descripción		El sistema presentar una interfaz de control de iluminación por cada una de las instalaciones de la casa. La interfaz deberá contar con un switch que simule un interruptor. Cada acción realizada por el usuario debe ser guardada automáticamente en una base de datos con fecha y hora de la acción.

➤ **Requisito Funcional 5**

Requisito		RF005
Nombre del Requisito	del	Control de Ventiladores
Prioridad		Media
Descripción		El sistema presentar una interfaz de control de ventilación mediante un switch. Los ventiladores deben encenderse automáticamente cuando la temperatura sea mayor a 21 °C caso contrario estarán apagados. Cada encendido ya pagado de dichos ventiladores deben ser registrados en una base de datos con fecha y hora de la actividad realizada.

➤ **Requisito Funcional 6**

Requisito		RF006
Nombre del Requisito	del	Control de Puertas
Prioridad		Media
Descripción		El sistema debe presentar una interfaz para la apertura y cierre de las puertas mediante el uso de botones. Así mismo, cada acción que realiza el usuario debe ser registrada en una base de datos con fecha y hora de la actividad realizada.

➤ **Requisito Funcional 7**

Requisito RF007	
Nombre del Requisito	Lectura de Temperatura
Prioridad	Media
Descripción	El sistema debe presentar una interfaz en la cual permita visualizar la temperatura ambiente actual y los valores recibidos se deben guardar en una base de datos con fecha y hora del valor recibido.

➤ **Requisito Funcional 8**

Requisito RF008	
Nombre del Requisito	Lectura de Humedad
Prioridad	Media
Descripción	El sistema debe presentar una interfaz en la cual permita visualizar la humedad ambiente actual y cada valor recibido se deberá guardar en una base de datos con fecha y hora del valor recibido.

➤ **Requisito Funcional 9**

Requisito RF009	
Nombre del Requisito	Gestión de Seguridad
Prioridad	Alta
Descripción	El sistema debe contar con un sensor de presencia el cual accione una alarma cuando detecte movimiento y envíe notificaciones al correo de dicha detección. Así mismo, cuando se detecte una temperatura anormal superior a los 26 °C se debe abrir la puerta automáticamente para evacuar, disparar la alarma, encender luces de evacuación y notificar de dicho suceso mediante correo electrónico. Por otro lado, se deben guardar estos registros en una base de datos con fecha y hora del suceso suscitado.

- **Requisitos No Funcionales**
 - **Requisito No Funcional 1**

Requisito RNF001	
Nombre del Requisito	Protocolo de comunicación
Prioridad	Alta
Descripción	El sistema domótico debe tener la capacidad de enviar y recibir información de forma automática de los diferentes servicios que existan en el hogar mediante una comunicación maquina a máquina, para ello dichos servicios deben estar configurados para tal propósito.

- **Requisito No Funcional 2**

Requisito RNF002	
Nombre del Requisito	Funcionamiento sistema
Prioridad	Alta
Descripción	El sistema domótico deberá mantener un funcionamiento constante, de tal manera que analicé los datos recibidos, que actúe en tiempo real independientemente de las acciones que se realicen de manera remota.

- **Requisito No Funcional 3**

Requisito RNF003	
Nombre del Requisito	Disponibilidad del Sistema
Prioridad	Alta
Descripción	El sistema domótico debe de estar conectado al Internet, es decir que se pueda acceder desde cualquier parte del mundo y en cualquier momento. Por tanto, es necesario la utilización de una conexión a la red local disponible en la vivienda.

➤ **Requisito No Funcional 4**

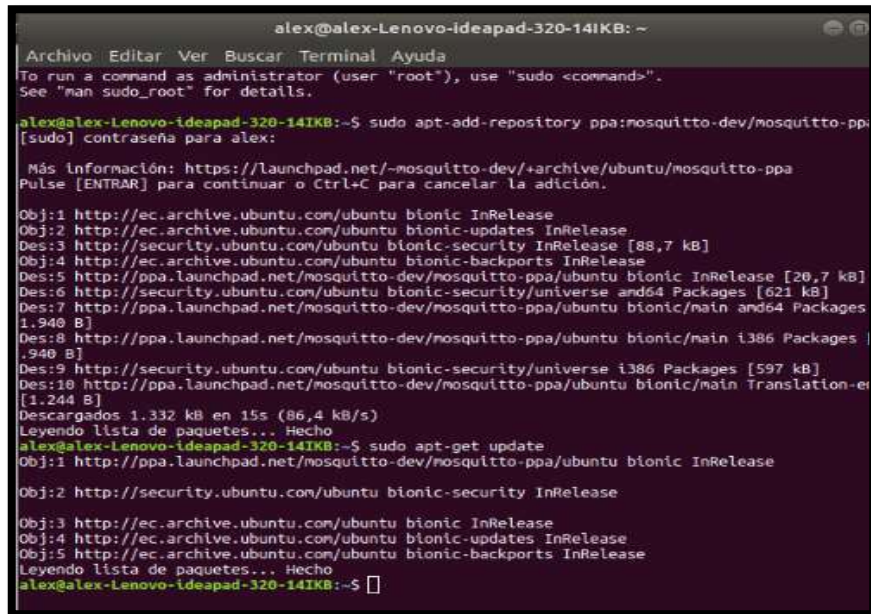
Requisito		RNF004
Nombre del Requisito		Usabilidad del Sistema
Prioridad		Alta
Descripción		Para administrar y controlar el sistema domótico será necesario la utilización de cualquier dispositivo que cuente con un navegador WEB sin importar el sistema operativo existente en el dispositivo

Anexo 2: Instalación y Configuración de Mosquito.

- Actualizar los repositorios de Ubuntu.

```
$ sudo apt-add-repository ppa:mosquitto-dev/mosquitto-ppa
```

```
$ sudo apt-get update
```



```
alex@alex-Lenovo-ideapad-320-14IKB: ~
Archivo Editar Ver Buscar Terminal Ayuda
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

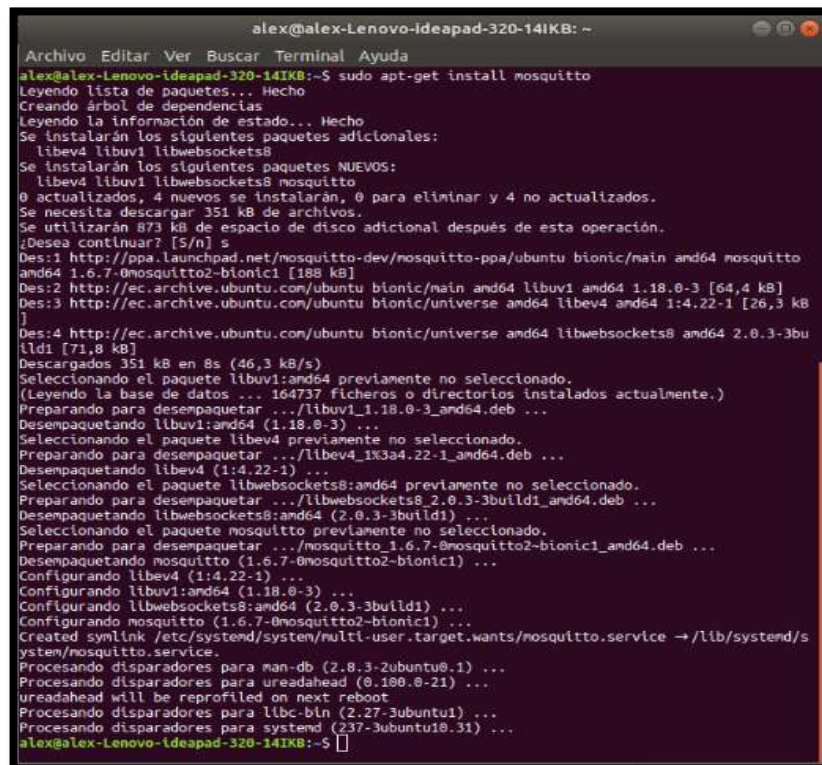
alex@alex-Lenovo-ideapad-320-14IKB:~$ sudo apt-add-repository ppa:mosquitto-dev/mosquitto-ppa
[sudo] contraseña para alex:

Más información: https://launchpad.net/~mosquitto-dev/+archive/ubuntu/mosquitto-ppa
Pulse [ENTRAR] para continuar o Ctrl+C para cancelar la adición.

Obj:1 http://ec.archive.ubuntu.com/ubuntu bionic InRelease
Obj:2 http://ec.archive.ubuntu.com/ubuntu bionic-updates InRelease
Des:3 http://security.ubuntu.com/ubuntu bionic-security InRelease [88,7 kB]
Obj:4 http://ec.archive.ubuntu.com/ubuntu bionic-backports InRelease
Des:5 http://ppa.launchpad.net/mosquitto-dev/mosquitto-ppa/ubuntu bionic InRelease [20,7 kB]
Des:6 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 Packages [621 kB]
Des:7 http://ppa.launchpad.net/mosquitto-dev/mosquitto-ppa/ubuntu bionic/main amd64 Packages
1.940 B]
Des:8 http://ppa.launchpad.net/mosquitto-dev/mosquitto-ppa/ubuntu bionic/main i386 Packages [
1.940 B]
Des:9 http://security.ubuntu.com/ubuntu bionic-security/universe i386 Packages [597 kB]
Des:18 http://ppa.launchpad.net/mosquitto-dev/mosquitto-ppa/ubuntu bionic/main Translation-e
[1.244 B]
Descargados 1.332 kB en 15s (86,4 kB/s)
Leyendo lista de paquetes... Hecho
alex@alex-Lenovo-ideapad-320-14IKB:~$ sudo apt-get update
Obj:1 http://ppa.launchpad.net/mosquitto-dev/mosquitto-ppa/ubuntu bionic InRelease
Obj:2 http://security.ubuntu.com/ubuntu bionic-security InRelease
Obj:3 http://ec.archive.ubuntu.com/ubuntu bionic InRelease
Obj:4 http://ec.archive.ubuntu.com/ubuntu bionic-updates InRelease
Obj:5 http://ec.archive.ubuntu.com/ubuntu bionic-backports InRelease
Leyendo lista de paquetes... Hecho
alex@alex-Lenovo-ideapad-320-14IKB:~$
```

- Una vez actualizado los repositorios se procede a instalar el bróker Mosquitto con el siguiente comando.

```
$ sudo apt-get install mosquitto
```

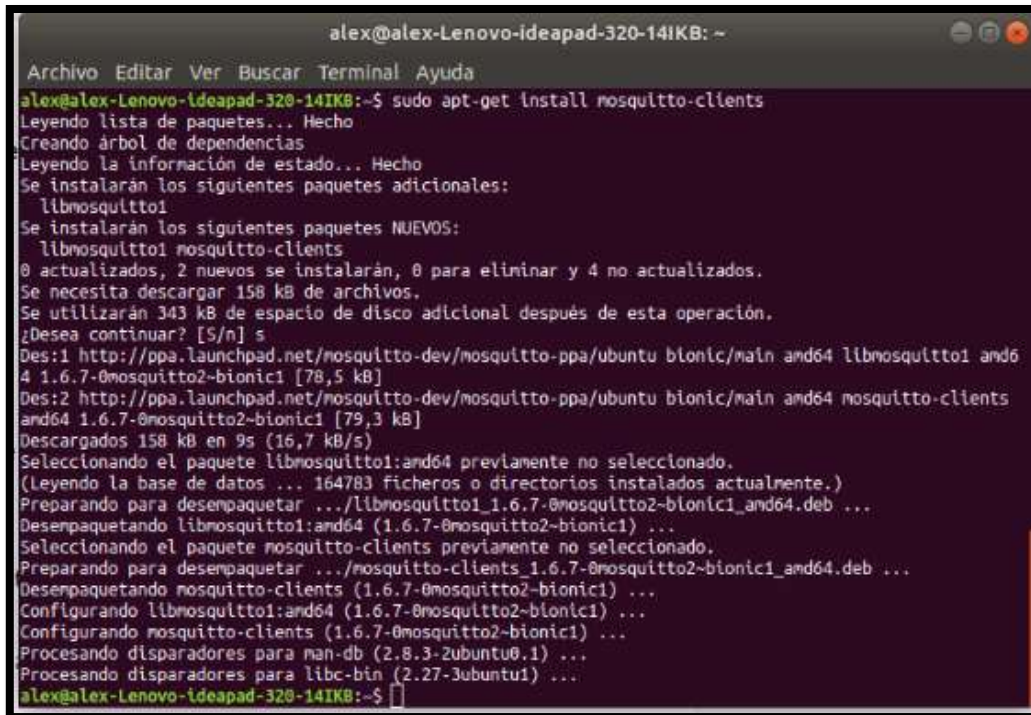


```
alex@alex-Lenovo-ideapad-320-14IKB: ~
Archivo Editar Ver Buscar Terminal Ayuda

alex@alex-Lenovo-ideapad-320-14IKB:~$ sudo apt-get install mosquitto
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
 libev4 libuv1 libwebsockets8
Se instalarán los siguientes paquetes NUEVOS:
 libev4 libuv1 libwebsockets8 mosquitto
0 actualizados, 4 nuevos se instalarán, 0 para eliminar y 4 no actualizados.
Se necesita descargar 351 kB de archivos.
Se utilizarán 873 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
Des:1 http://ppa.launchpad.net/mosquitto-dev/mosquitto-ppa/ubuntu bionic/main amd64 mosquitto
amd64 1.6.7-0mosquitto2-bionic1 [188 kB]
Des:2 http://ec.archive.ubuntu.com/ubuntu bionic/main amd64 libuv1 amd64 1.18.0-3 [64,4 kB]
Des:3 http://ec.archive.ubuntu.com/ubuntu bionic/universe amd64 libev4 amd64 1:4.22-1 [26,3 kB]
]
Des:4 http://ec.archive.ubuntu.com/ubuntu bionic/universe amd64 libwebsockets8 amd64 2.0.3-3bu
ild1 [71,0 kB]
Descargados 351 kB en 8s (46,3 kB/s)
Seleccionando el paquete libuv1:amd64 previamente no seleccionado.
(Leyendo la base de datos ... 164737 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../libuv1_1.18.0-3_amd64.deb ...
Desempaquetando libuv1:amd64 (1.18.0-3) ...
Seleccionando el paquete libev4 previamente no seleccionado.
Preparando para desempaquetar .../libev4_1x3a4.22-1_amd64.deb ...
Desempaquetando libev4 (1:4.22-1) ...
Seleccionando el paquete libwebsockets8:amd64 previamente no seleccionado.
Preparando para desempaquetar .../libwebsockets8_2.0.3-3build1_amd64.deb ...
Desempaquetando libwebsockets8:amd64 (2.0.3-3build1) ...
Seleccionando el paquete mosquitto previamente no seleccionado.
Preparando para desempaquetar .../mosquitto_1.6.7-0mosquitto2-bionic1_amd64.deb ...
Desempaquetando mosquitto (1.6.7-0mosquitto2-bionic1) ...
Configurando libev4 (1:4.22-1) ...
Configurando libuv1:amd64 (1.18.0-3) ...
Configurando libwebsockets8:amd64 (2.0.3-3build1) ...
Configurando mosquitto (1.6.7-0mosquitto2-bionic1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/mosquitto.service → /lib/systemd/s
ystem/mosquitto.service.
Procesando disparadores para man-db (2.8.3-2ubuntu0.1) ...
Procesando disparadores para ureadahead (0.100.0-21) ...
ureadahead will be reprofiled on next reboot
Procesando disparadores para libc-bin (2.27-3ubuntu1) ...
Procesando disparadores para systemd (237-3ubuntu10.31) ...
alex@alex-Lenovo-ideapad-320-14IKB:~$
```

- Finalmente se procede a instalar el cliente Mosquito para ellos se lo hace con el siguiente comando.

\$ sudo apt-get install mosquito-clients



```
alex@alex-Lenovo-ideapad-320-14IKB: ~
Archivo Editar Ver Buscar Terminal Ayuda
alex@alex-Lenovo-ideapad-320-14IKB:~$ sudo apt-get install mosquito-clients
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
 libmosquitto1
Se instalarán los siguientes paquetes NUEVOS:
 libmosquitto1 mosquito-clients
0 actualizados, 2 nuevos se instalarán, 0 para eliminar y 4 no actualizados.
Se necesita descargar 158 kB de archivos.
Se utilizarán 343 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
Des:1 http://ppa.launchpad.net/mosquitto-dev/mosquitto-ppa/ubuntu bionic/main amd64 libmosquitto1 amd64 1.6.7-0mosquitto2-bionic1 [78,5 kB]
Des:2 http://ppa.launchpad.net/mosquitto-dev/mosquitto-ppa/ubuntu bionic/main amd64 mosquito-clients amd64 1.6.7-0mosquitto2-bionic1 [79,3 kB]
Descargados 158 kB en 9s (16,7 kB/s)
Seleccionando el paquete libmosquitto1:amd64 previamente no seleccionado.
(Leyendo la base de datos ... 164783 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../libmosquitto1_1.6.7-0mosquitto2-bionic1_amd64.deb ...
Desempaquetando libmosquitto1:amd64 (1.6.7-0mosquitto2-bionic1) ...
Seleccionando el paquete mosquito-clients previamente no seleccionado.
Preparando para desempaquetar .../mosquitto-clients_1.6.7-0mosquitto2-bionic1_amd64.deb ...
Desempaquetando mosquito-clients (1.6.7-0mosquitto2-bionic1) ...
Configurando libmosquitto1:amd64 (1.6.7-0mosquitto2-bionic1) ...
Configurando mosquito-clients (1.6.7-0mosquitto2-bionic1) ...
Procesando disparadores para man-db (2.8.3-2ubuntu0.1) ...
Procesando disparadores para libc-bin (2.27-3ubuntu1) ...
alex@alex-Lenovo-ideapad-320-14IKB:~$
```

Anexo 3: Instalación y Configuración de Node-RED.

Para instalar Node-RED primeramente se debe instalar Node.Js debido a que Node-RED es desarrollado en base a Node.JS. a continuación, se instalará Node.js.

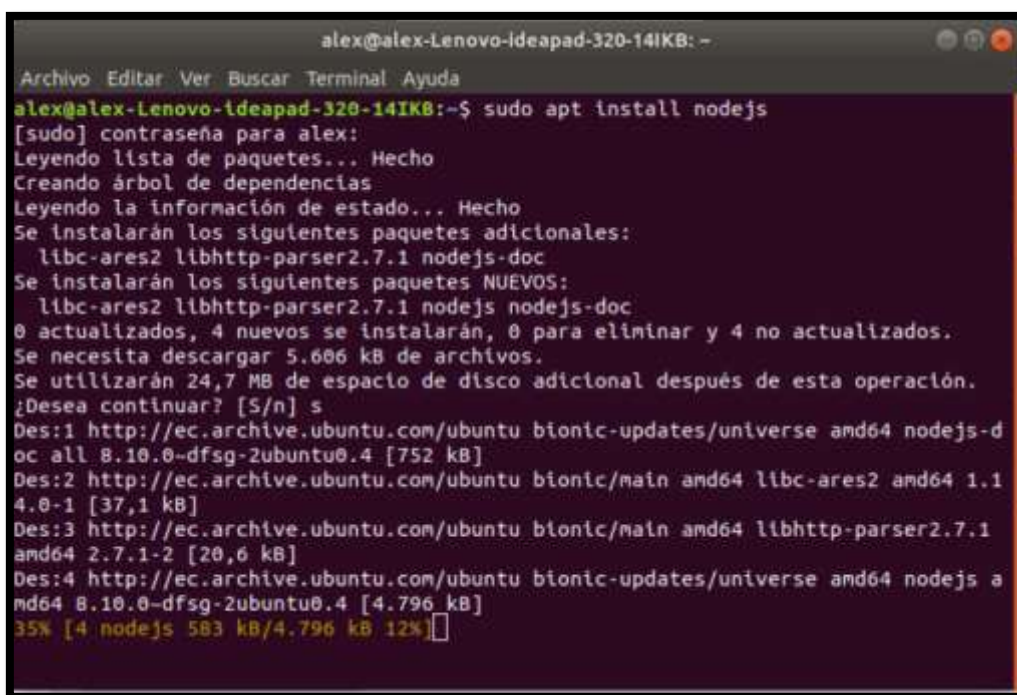
- **Instalación de Node.Js en Ubuntu 18.04.**

Primeramente, se actualiza la lista de paquetes del sistema con el siguiente comando:

```
$ sudo apt-get update
```

Luego se instala Node.Js desde los repositorios con el siguiente comando:

```
$ sudo apt install nodejs
```



Después de proceder a instalar el administrador de paquetes de Node.js npm con la siguiente línea de comandos:

```
$ sudo apt install npm
```

```
alex@alex-Lenovo-ideapad-320-14IKB: ~  
Archivo Editar Ver Buscar Terminal Ayuda  
alex@alex-Lenovo-ideapad-320-14IKB:~$ sudo apt install npm  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
Se instalarán los siguientes paquetes adicionales:  
build-essential dpkg-dev fakeroot g++ g++-7 gcc gcc-7 gyp javascript-common  
libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl  
libasan4 libatomic1 libc-dev-bin libc6-dev libcilkrts5 libfakeroot  
libgcc-7-dev libitm1 libjs-async libjs-inherits libjs-jquery libjs-node-uuid  
libjs-underscore liblsan0 libmpx2 libpython-stdlib libquadmath0  
libssl1.0-dev libstdc++-7-dev libtsan0 libubsan0 libuv1-dev linux-libc-dev  
make manpages-dev node-abbrev node-ansi node-ansi-color-table node-archy  
node-async node-balanced-match node-block-stream node-brace-expansion  
node-builtin-modules node-combined-stream node-concat-map node-cookie-jar  
node-delayed-stream node-forever-agent node-form-data node-fs.realpath  
node-fstream node-fstream-ignore node-github-url-from-git node-glob  
node-graceful-fs node-gyp node-hosted-git-info node-inflight node-inherits  
node-ini node-is-builtin-module node-isexe node-json-stringify-safe  
node-lockfile node-lru-cache node-mime node-minimatch node-mkdirp  
node-mute-stream node-node-uuid node-nopt node-normalize-package-data  
node-npmlog node-once node-osenv node-path-is-absolute node-pseudomap  
node-qs node-read node-read-package-json node-request node-retry node-rimraf  
node-semver node-sha node-slide node-spdx-correct node-spdx-expression-parse  
node-spdx-license-ids node-tar node-tunnel-agent node-underscore
```

Luego para verificar si se instaló correctamente se digita el siguiente comando en la terminal para ver que versión de Node.js se ha instalado.

```
$ nodejs -v
```

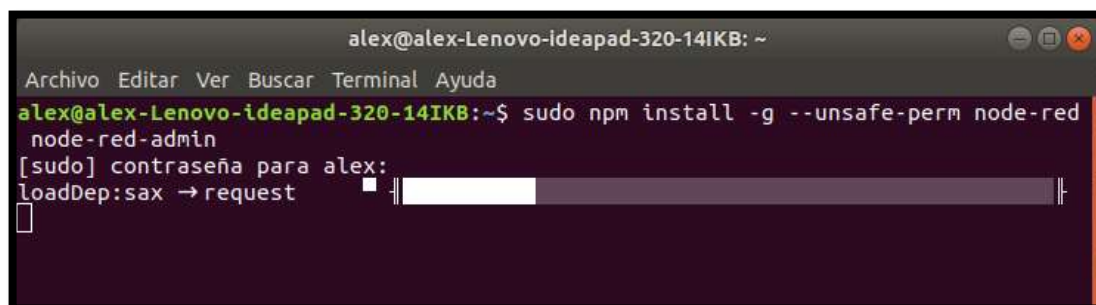
```
alex@alex-Lenovo-ideapad-320-14IKB: ~  
Archivo Editar Ver Buscar Terminal Ayuda  
alex@alex-Lenovo-ideapad-320-14IKB:~$ nodejs -v  
v8.10.0  
alex@alex-Lenovo-ideapad-320-14IKB:~$
```

Una vez verificado que versión de Node.js se tiene instalado se procede a la instalación de la herramienta Node-RED.

➤ **Instalación de Node-RED en Ubuntu 18.04**

Para la instalación de Node-RED se emplea el administrador de paquetes de Node.js npm. Y se llama a una utilidad auxiliar de Node-RED. El comando es el siguiente.

```
$ sudo npm install -g --unsafe-perm node-red node-red-admin
```



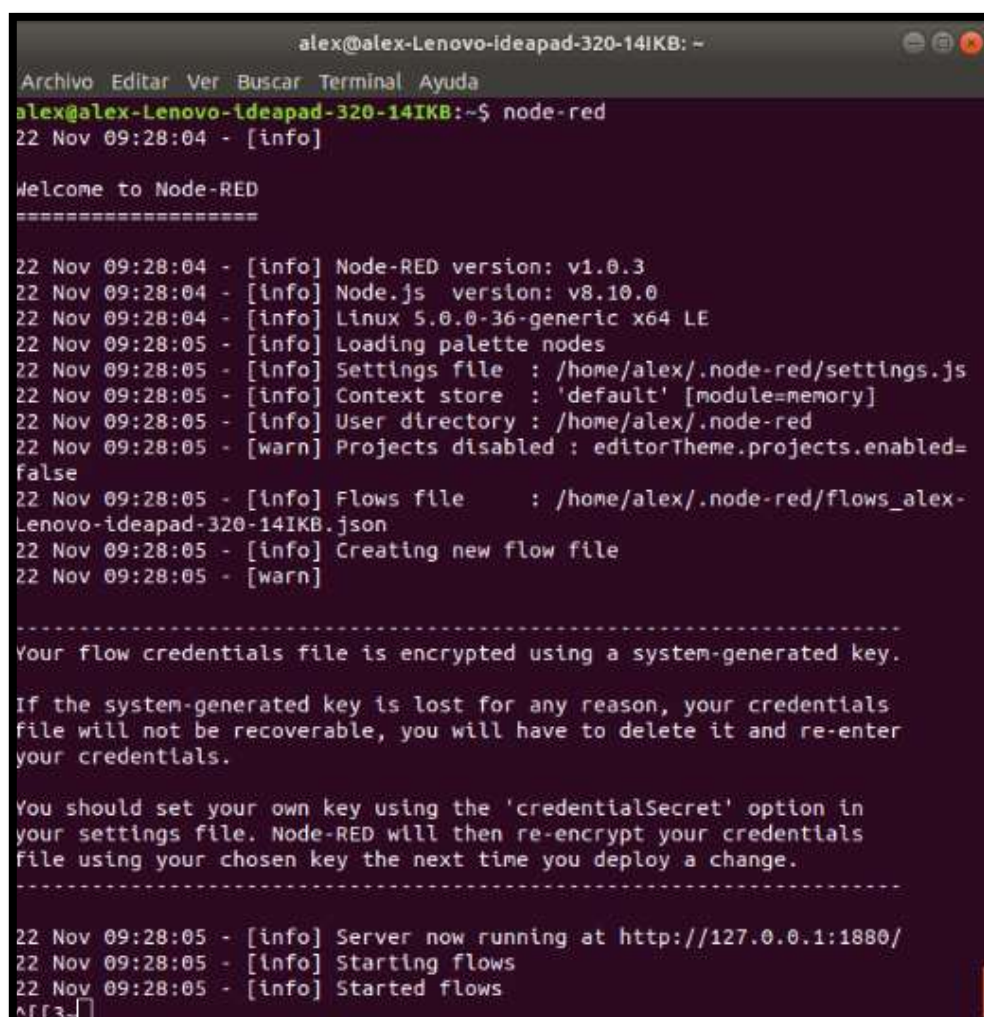
```
alex@alex-Lenovo-ideapad-320-14IKB: ~  
Archivo Editar Ver Buscar Terminal Ayuda  
alex@alex-Lenovo-ideapad-320-14IKB:~$ sudo npm install -g --unsafe-perm node-red  
node-red-admin  
[sudo] contraseña para alex:  
loadDep:sax → request
```

Una vez que se termine de instalar la herramienta se procede a instalar el dashboard que nos permitirá crear nuestra interfaz gráfica de la aplicación. Para instalarla se lo hace con la siguiente línea de comandos:

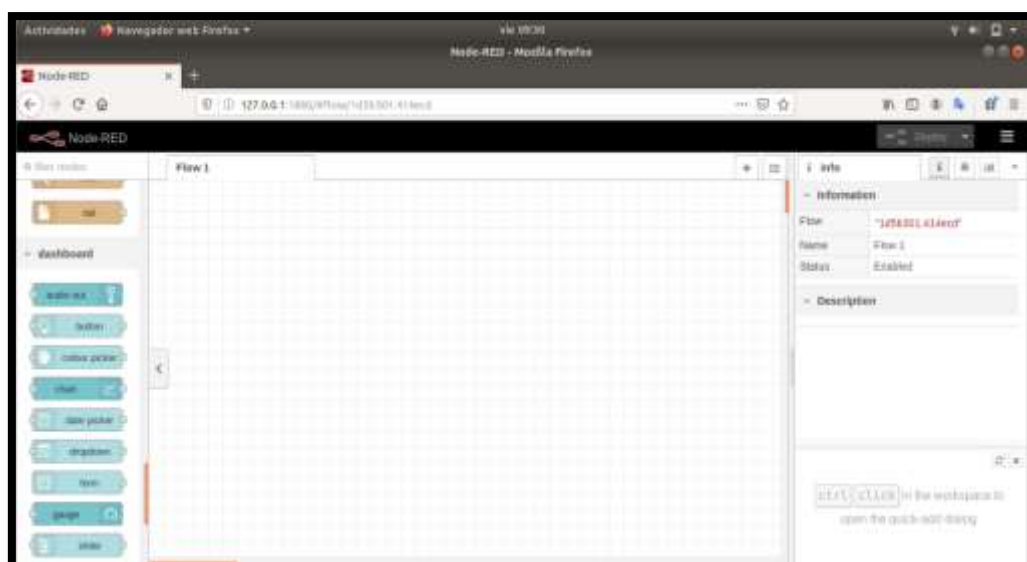
```
$ npm i node-red/node-red-dashboard
```

Finalmente se pone a correr el servicio de Node-RED con el siguiente comando.

```
$node-red
```



```
alex@alex-Lenovo-ideapad-320-14IKB: ~  
Archivo Editar Ver Buscar Terminal Ayuda  
alex@alex-Lenovo-ideapad-320-14IKB:~$ node-red  
22 Nov 09:28:04 - [info]  
  
Welcome to Node-RED  
=====  
  
22 Nov 09:28:04 - [info] Node-RED version: v1.0.3  
22 Nov 09:28:04 - [info] Node.js version: v8.10.0  
22 Nov 09:28:04 - [info] Linux 5.0.0-36-generic x64 LE  
22 Nov 09:28:05 - [info] Loading palette nodes  
22 Nov 09:28:05 - [info] Settings file : /home/alex/.node-red/settings.js  
22 Nov 09:28:05 - [info] Context store : 'default' [module=memory]  
22 Nov 09:28:05 - [info] User directory : /home/alex/.node-red  
22 Nov 09:28:05 - [warn] Projects disabled : editorTheme.projects.enabled=false  
22 Nov 09:28:05 - [info] Flows file : /home/alex/.node-red/flows_alex-Lenovo-ideapad-320-14IKB.json  
22 Nov 09:28:05 - [info] Creating new flow file  
22 Nov 09:28:05 - [warn]  
  
-----  
Your flow credentials file is encrypted using a system-generated key.  
  
If the system-generated key is lost for any reason, your credentials file will not be recoverable, you will have to delete it and re-enter your credentials.  
  
You should set your own key using the 'credentialSecret' option in your settings file. Node-RED will then re-encrypt your credentials file using your chosen key the next time you deploy a change.  
-----  
  
22 Nov 09:28:05 - [info] Server now running at http://127.0.0.1:1880/  
22 Nov 09:28:05 - [info] Starting flows  
22 Nov 09:28:05 - [info] Started flows  
^[[Z]
```

➤ **Configuración de Autenticación para acceder al dashboard de Node-RED.**

En esta sección se procederá a proteger Node-Red ya que por defecto no trae ninguna protección y cualquier persona puede acceder mediante su IP al editor e implementar cambios en los flujos que hemos creado. A continuación, se detallan los pasos para crear esta seguridad mediante una autenticación basada en nombre de usuario y contraseña:

Primeramente, se accede mediante la terminal al directorio “. node-red” luego buscamos en el directorio el archivo settings.js el cual lo vamos a editar con el comando “nano settings.js” una vez que ejecutamos este comando podremos editar este archivo. Dentro del mismo buscamos donde nos dice “Securing Node-Red” las cuales copiamos y pegamos en la parte inferior para posteriormente editarlas. Ahora bien, si nos fijamos en “username” podemos poner el usuario que queramos y en “password” vamos a tener que generar la contraseña para eso seguiremos otro proceso detallando a continuación.

```
// If you installed the optional node-red-dashboard you can set it's path
// relative to httpRoot
//ui: { path: "ui" },

// Securing Node-RED
// -----
// To password protect the Node-RED editor and admin API, the following
// property can be used. See http://nodered.org/docs/security.html for details.
//adminAuth: {
//  type: "credentials",
//  users: [{
//    username: "admin",
//    password: "$2a$08$zZWtXTja0fB1pzD4sHcMyOCMyz2Z6dNbM6t18sJogENOMcxWV9DN.",
//    permissions: "*"
//  }]
//},

adminAuth: {
  type: "credentials",
  users: [{
    username: "admin",
    password: "$2a$08$UIfh2gjiC1lIRk.AI.WxT.gFSMKukQhiSroL5Fbcp4qWxtT2Djqb2",
    permissions: "*"
  }]
},
```

Para generar el hash de contraseña se usa la herramienta “node-red-admin”, para esto primeramente abrimos otra consola y procedemos a instalarla con la siguiente línea de comandos:

\$ npm install -g node-red-admin

```
root@gastonchamba:~# sudo npm install -g node-red-admin
/usr/local/bin/node-red-admin -> /usr/local/lib/node_modules/node-red-admin/node-red-admin.js
> bcrypt@3.0.7 install /usr/local/lib/node_modules/node-red-admin/node_modules/bcrypt
> node-pre-gyp install --fallback-to-build
node-pre-gyp WARN Using request for node-pre-gyp https download
bcrypt libcrypto Success: "/usr/local/lib/node_modules/node-red-admin/node_modules/bcrypt/lib/binding/bcrypt_lib.node" is installed via remote:
/usr/local/lib
├── node-red-admin@0.1.5
│   ├── bcrypt@3.0.7
│   ├── nan@2.14.0
│   ├── node-pre-gyp@0.13.0
│   │   ├── debug@3.2.0
│   │   ├── ms@2.1.2
│   │   ├── npm-packlist@1.4.6
│   │   ├── ignore-walk@3.0.3
│   │   ├── npmlog@4.1.2
│   │   ├── are-we-there-yet@1.1.5
│   │   ├── readable-stream@2.3.6
│   │   └── process-nextick-args@2.0.1
│   ├── rimraf@2.7.1
│   ├── glob@7.1.6
│   ├── inherits@2.0.4
│   ├── semver@5.7.1
│   ├── tar@4.4.13
│   ├── chownr@1.1.3
│   ├── fs-minipass@1.2.7
│   ├── minipass@2.9.0
│   ├── minilib@1.3.3
│   ├── yallist@3.1.1
│   └── colors@1.4.0
```

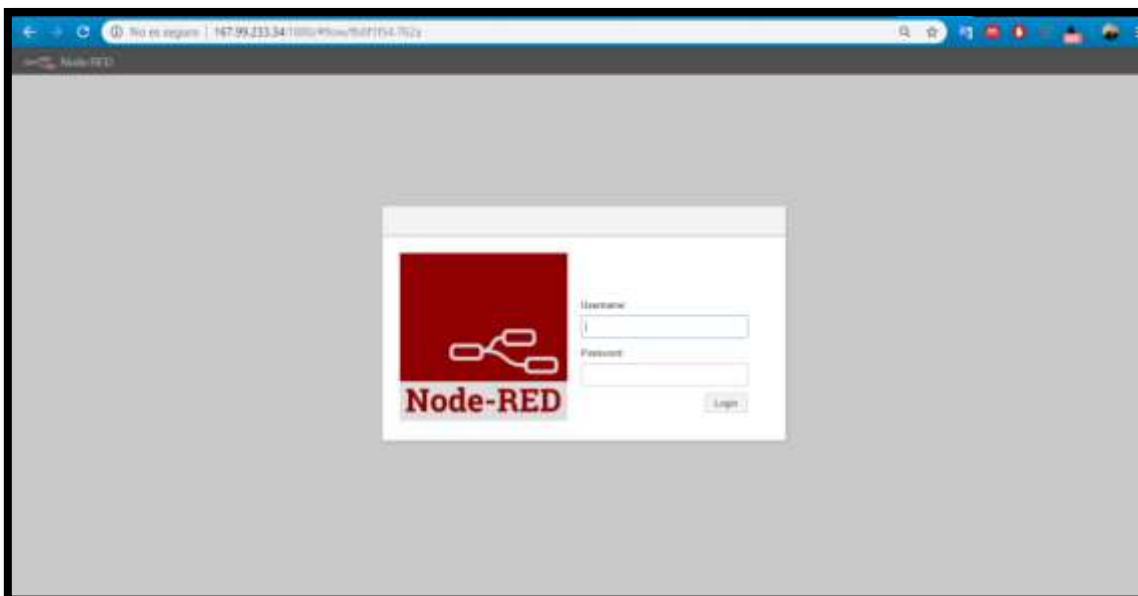
Una vez instalada la herramienta se ejecuta el siguiente comando el cual nos va a generar la contraseña.

\$ node-red-admin hash-pw

Una vez ejecutado este comando la herramienta va a pedir la contraseña que se desea usar para acceder al tablero de Node-RED y luego imprimirá el hash que se puede copiar en el archivo de configuración anteriormente abierto para editar.

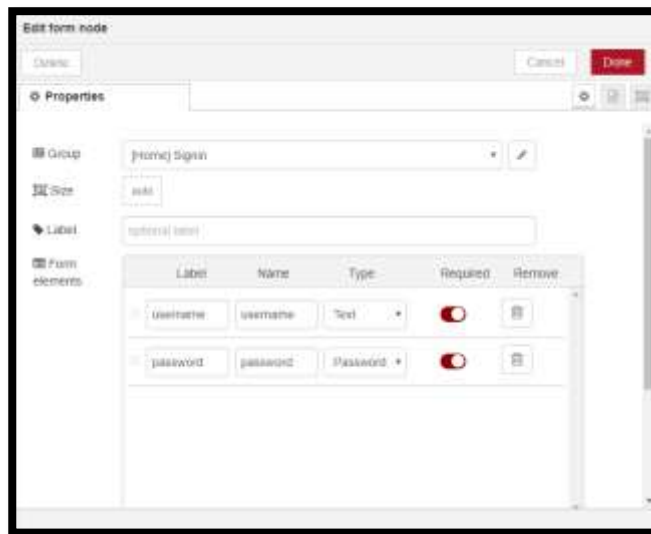
```
root@gastonchamba:~# node-red-admin hash-pw
Password:
$2b$08$Ii5jIZ1E3G6P9wqM0jv0V0B8p02pPpwLM3iBzSMmYmT8T7iUmXSd0
root@gastonchamba:~#
```

Finalmente guardamos el archivo editado y ejecutamos Node-Red y accedemos mediante su IP. Y listo nos mostrara una interfaz donde debemos ingresar el usuario y contraseña que hemos configurado y por ende accederemos al editor y posterior a eso se cierra la sesión.



Anexo 4: Implementación de las Funciones, Formularios en Node-RED para la Autenticación.

- **Formulario Sing-In.** Este formulario nos servirá para enviar datos de usuario y contraseña antes de acceder a nuestro sistema de control domótico.



- **Función Account Verification.** Esta función nos permite crear los usuarios que el sistema aceptar para acceder al mismo. Además permite controlar si las credenciales están bien o no enviándonos notificaciones de los campos que están mal. Otra funcionalidad es que nos permite controlar que solo un usuario pueda acceder a la misma vez bloqueando el acceso para el resto que quiera ingresar.

```
1- var accounts = flow.get("accounts") || [
2   { username : "admin", password : "admin"},
3   { username : "user", password : "user"}];
4
5 var username = msg.payload.username ;
6 var password = msg.payload.password ;
7
8 msg.payload = 1;
9
10- accounts.forEach(function ( account ){
11-   if ( account.username == username ) {
12-     msg.payload = 2;
13-     if ( account.password == password ) {
14-       msg.payload = 0;
15-     }
16-   }
17- });
18
19- IF ( msg.payload == 0 ) {
20   var currentsocketid = flow.get("clientid") || undefined;
21   if ( currentsocketid != undefined && currentsocketid != msg.socketid ) msg.payload = 3;
22 }
23 // keep the original socketid from msg.socketid;
24 return msg;
```

- **Función = 0: Success.** Esta función se activa una vez que el usuario y contraseña este correctos, de tal manera que una vez bien ingresados nos va a permitir acceder al control completo de nuestro hogar mostrando toda la información y control de la misma.

```
1  /* activate session timer */
2  var sessionTimer = flow.get("sessionTimer") || 0;
3  var currTime = Date.now();
4  flow.set("sessionTimer", currTime);
5  flow.set("clientid", msg.socketid);
6  /* ui-control payload */
7  msg.payload = { group: {
8      show : ["Home_Puertas",
9              "Home_Temperatura",
10             "Home_Iluminación",
11             "Home_Seguridad",
12             "Home_Humedad",
13             "Home_Ventilación",
14             "Home_Cerrar_Sesión"],
15      hide : ["Home_Signin"]
16    }
17  };
18
19  return msg;
```

- **Función = 1: Unknown User.** Esta función se activa si el usuario ingresado no es correcto de tal manera que notificara que dicho usuario no existe.

```
1  msg.payload = "El Usuario No Existe"
2  return msg;
```

- **Función = 2: Wrong Password.** Esta función se activa si la contraseña ingresada no es correcta de tal manera que notificara que la contraseña ingresada es invalida.

```
1  msg.payload = "Contraseña Invalida";
2  return msg;
```

- **Función = 4: System In Use.** Esta función se activa si el sistema está en uso por otras credenciales de tal manera que notificara que el sistema está en uso y que intente en unos minutos.

```
1  msg.payload = "Usuario en Uso ;; Intente en breves minutos";
2  return msg;
```

- **Función Detect Session Timer.** Esta función permite controlar el tiempo que tendrá de duración la sesión y una vez cumplido con el tiempo la cierra automáticamente.

```
1 var sessionTimer = flow.get("sessionTimer") || 0;
2 var currTime = Date.now();
3 var SESSION_TIMEOUT = 30000; //
4
5 ▾ if ( sessionTimer === 0 /* Inactive, No user signed in */ ) {
6     msg.payload = 2;
7 ▾ } else {
8 ▾     if ( currTime - sessionTimer > SESSION_TIMEOUT ) {
9         msg.payload = 0;
10 ▾     } else {
11         msg.payload = 1;
12 ▾     }
13 ▾ }
14 return msg;
```

- **Función Calculating Remaining Time (Debug).** Esta función permite crear una regla que ayuda a notificar al usuario cuando el sistema esté a punto de cerrar su sesión. Creando un contador y vaya notificando cuando falten 10 s, 5 s, y 2 s.

```
1 var sessionTimer = flow.get("sessionTimer") || 0;
2 var currTime = Date.now();
3 var remainingSecs = Math.floor((300000 - (currTime - sessionTimer))/1000) + 1;
4
5 ▾ if ( sessionTimer == 0 /* Inactive, No user signed in */ ) {
6     return null;
7 ▾ } else {
8
9     if ( remainingSecs <= 7 ) msg.payload = "Remaining : " + remainingSecs + " secs";
10    else return null;
11 ▾ }
12
13 msg.socketid = flow.get("clientid") || "B1234";
14 return msg;
```

- **Función Time-Out.** Esta función se activa cuando el tiempo de sesión se ha agotado de tal manera que cierra la sesión y nos muestra un cuadro de dialogo que la sesión ha expirado y que ingrese las credenciales nuevamente.

```
1 /* activate session timer */
2 var sessionTimer = flow.get("sessionTimer") || 0;
3 flow.set("sessionTimer", 0);
4
5 msg.socketid = flow.get("clientid") || undefined;
6 msg.payload = "¡Sesión expirada! Por favor ingrese de nuevo!";
7 return msg;
```

- **Función Sing-In Redirection.** Esta función oculta el menú una vez que se termine el tiempo de sesión.

```
1 var msg = {};
2
3 msg.socketid = flow.get("clientid") || undefined;
4 msg.payload = { group: {
5   hide : ["Home_Puertas",
6           "Home_Temperatura",
7           "Home_Iluminación",
8           "Home_Seguridad",
9           "Home_Humedad",
10          "Home_Ventilación",
11          "Home_Cerrar_Sesión"],
12   show : ["Home_Signin"]
13 }
14 };
15
16 flow.set("clientid", undefined);
17 return msg;
```

- **Función Signin Goup.** Esta función muestra todo el menú durante el tiempo predeterminado de la sesión.

```
1 var msg = {};
2 msg.payload = { group: {
3   hide : ["Home_Puertas",
4           "Home_Temperatura",
5           "Home_Iluminación",
6           "Home_Seguridad",
7           "Home_Humedad",
8           "Home_Ventilación",
9           "Home_Cerrar_Sesión"],
10  show : ["Home_Signin"]
11 }
12 };
13 return msg;
```

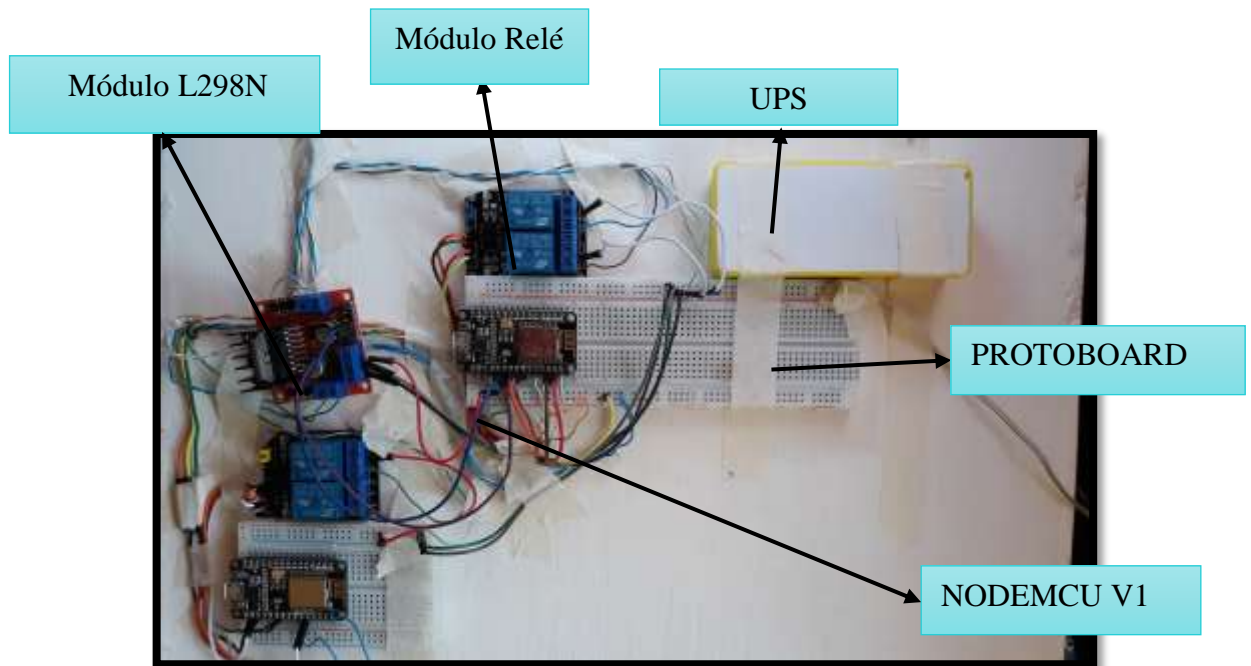
- **Función Email Temperatura Excedida:** En esta función se crea una condición en la cual si la temperatura ambiente es mayor a 25 °C este va a enviar un mensaje de alerta al correo personal incluyendo la lectura actual del sensor.

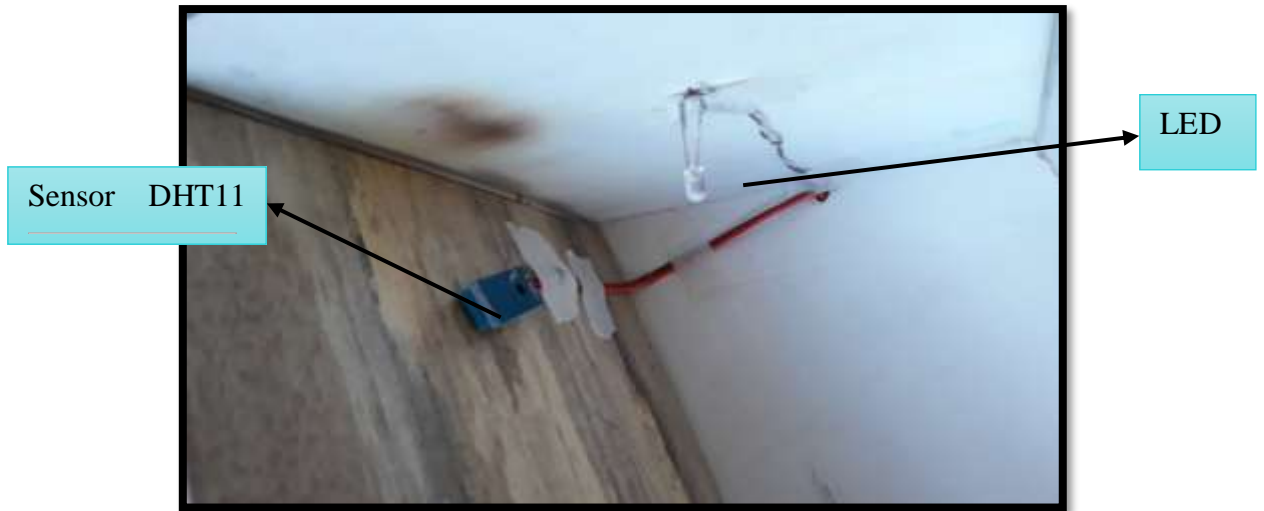
```
1 if(msg.payload > 25)
2 {
3   msg.topic = "Alerta Temperatura Excedida";
4   msg.payload = "Revisar Instalaciones Podria estarse produciendo un incendio su Temperatura actual es : " + msg.payload + "°C";
5   return msg;
6 }
```

- **Función Email Presencia Detectada:** en esta función se crea la condición en la cual si se recibe un 1 lógico que equivale a la detección de presencia física por parte del sensor este muestre notificación en la interfaz gráfica del sistema y además envíe notificación por medio de correo electrónico.

```
1 if(msg.payload == 1)
2 {
3   msg.topic = "Presencia Detectada";
4   msg.payload = "Intruso Detectado ¡Alerta" ;
5   return msg;
6 }
```


Anexo 5: Implementación.





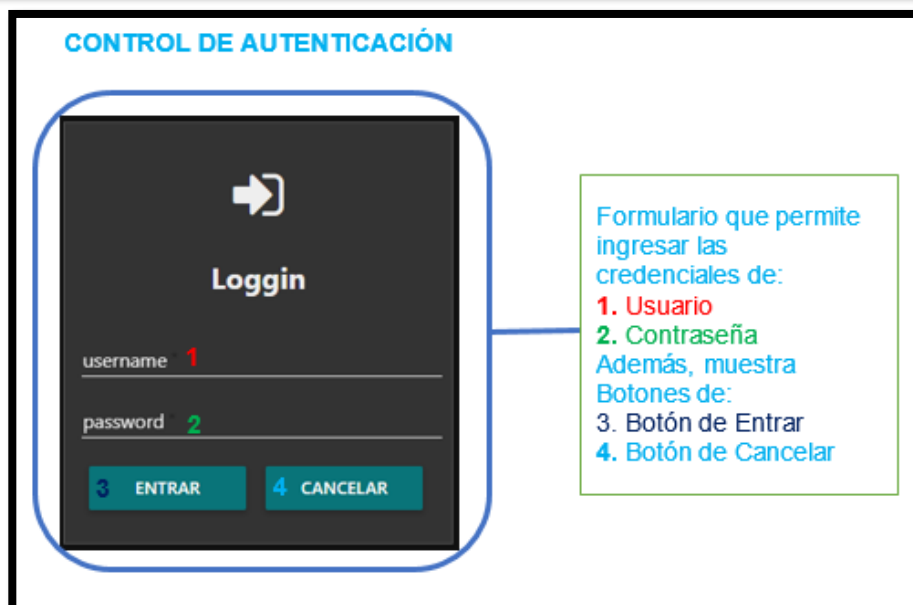
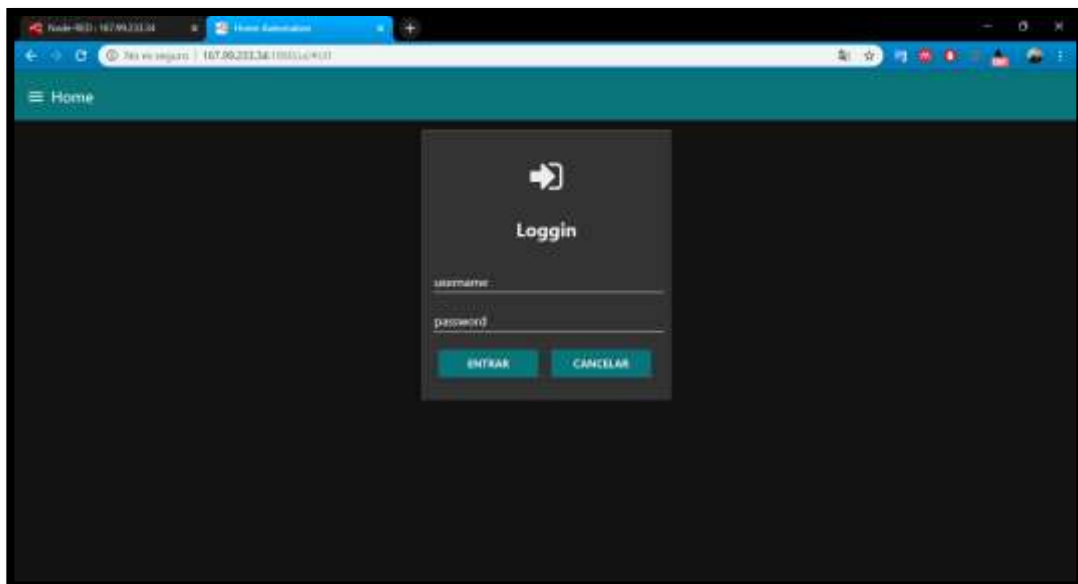


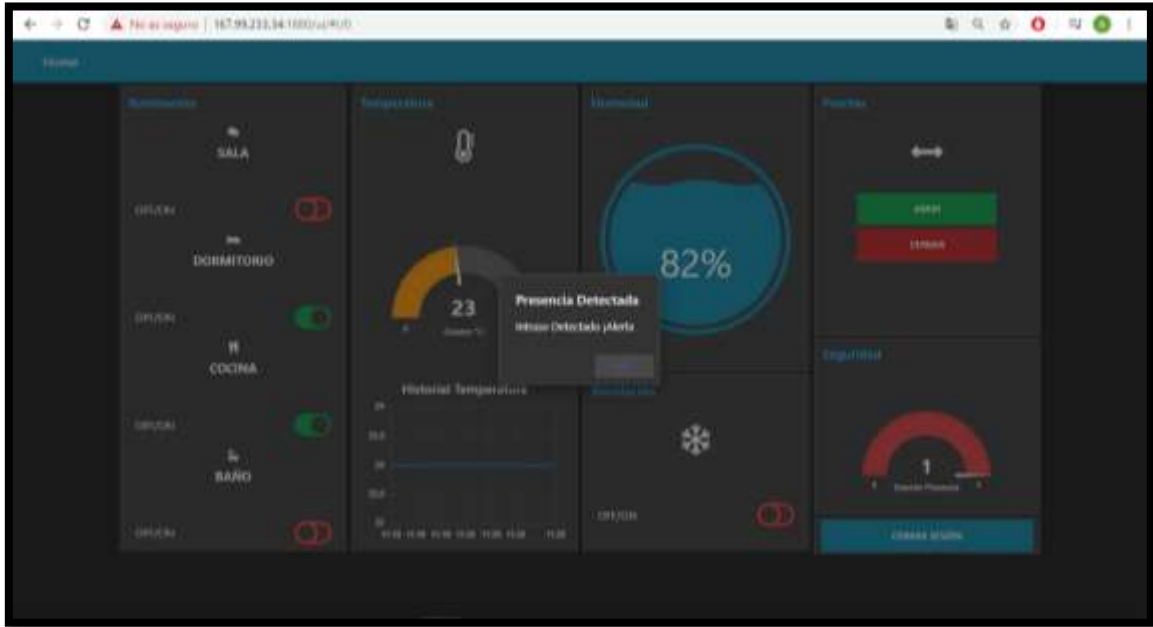
Anexo 6: Manual de Usuario.

Mediante un navegador Web accedemos a la siguiente dirección:

<http://167.99.233.34:1880/ui>

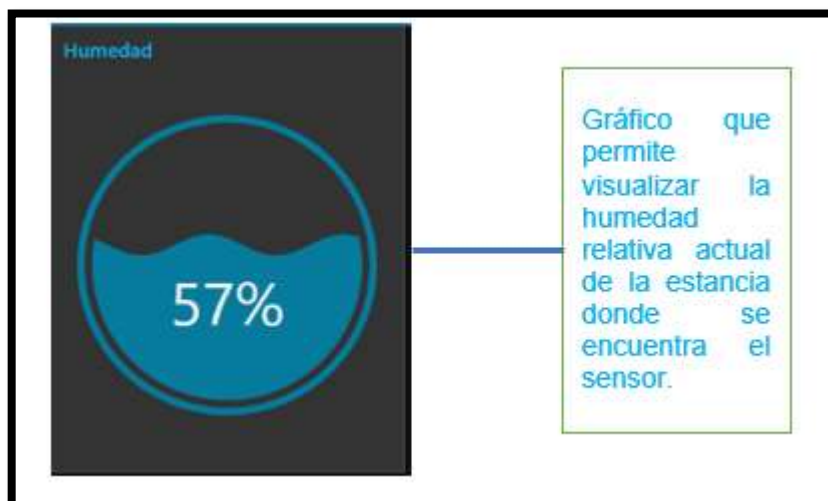
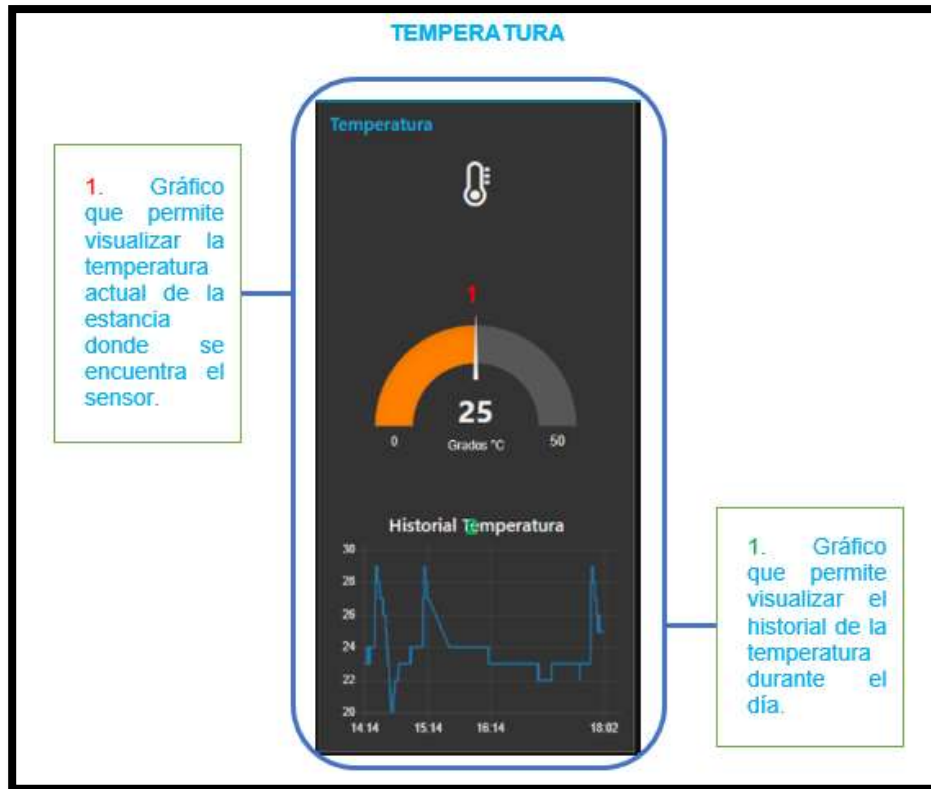
Una vez que se ingresa mostrará un control de acceso en el cual pedirá que se ingrese un usuario y una contraseña. En este caso, se utilizará las siguientes credenciales para acceder al menú de control: Usuario: user y Contraseña: user.

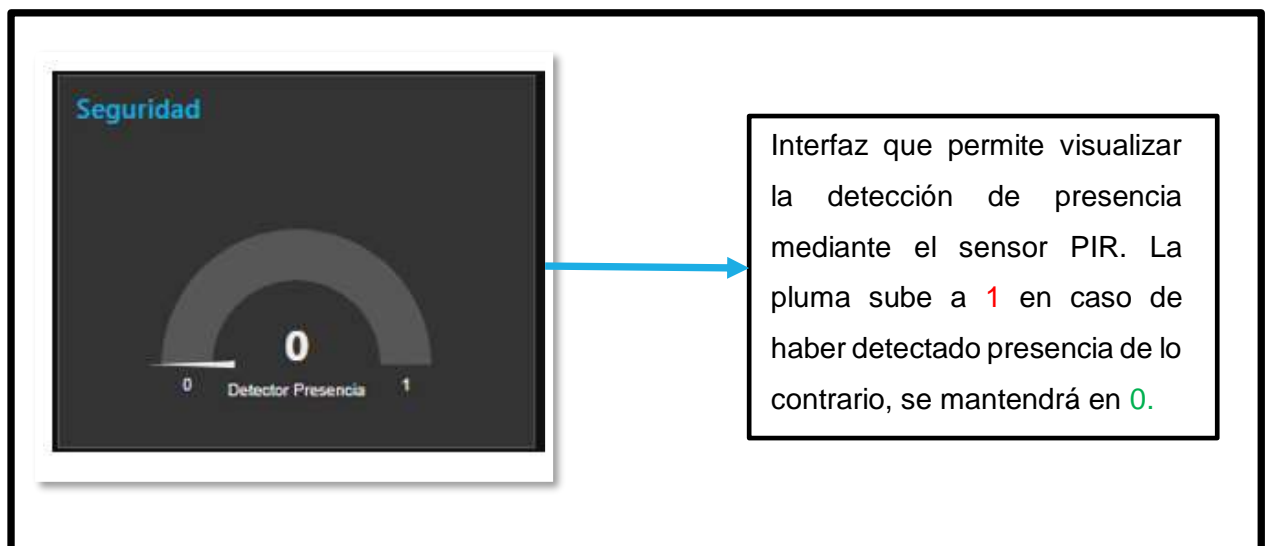
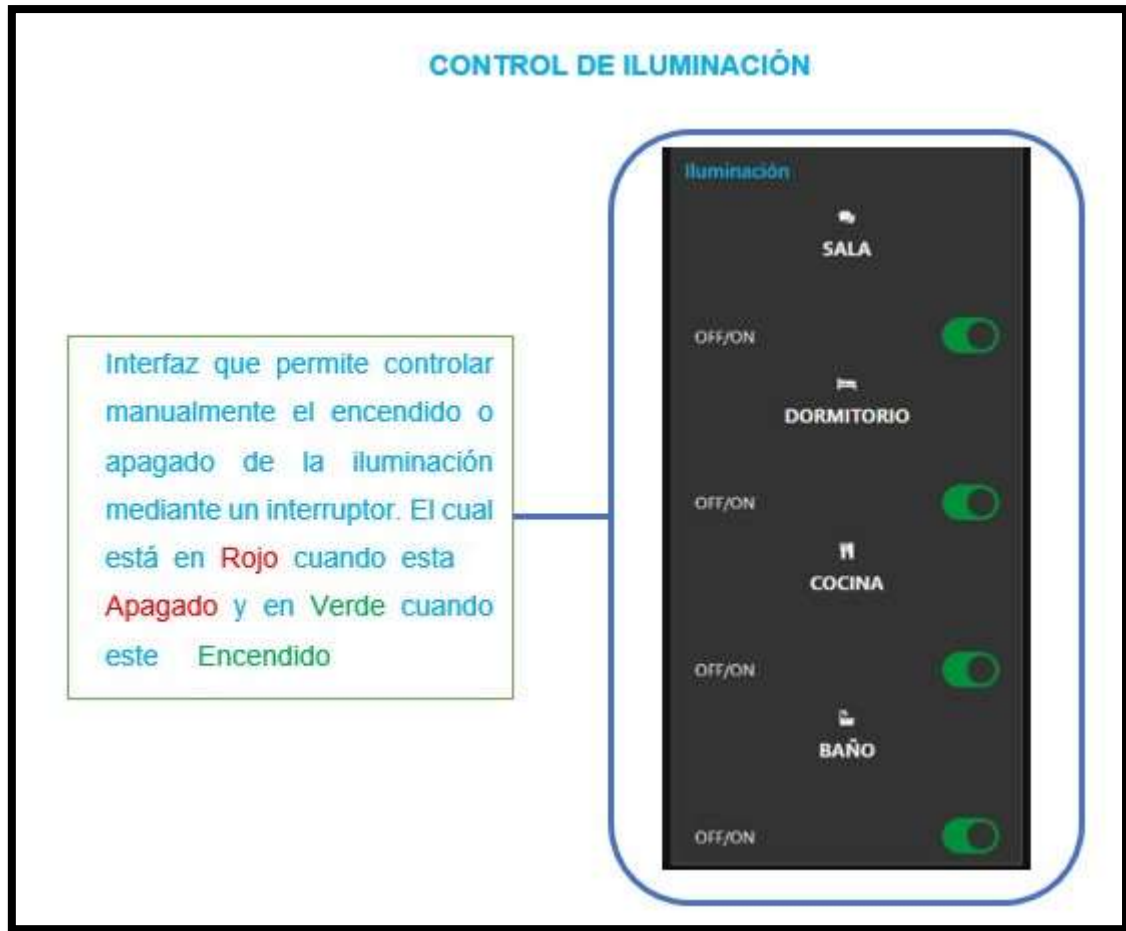




Esta Sección permite controlar la apertura y cierre de la puerta.

1. Botón verde Para Abrir
2. Botón rojo para cerrar





Anexo 7: Encuesta Para la Evaluación del Sistema Domótico.

Encuesta General Sobre la Usabilidad del Sistema Domótico

El objetivo de la siguiente encuesta es conocer el grado de satisfacción de los/as usuarios/as con los beneficios que ofrece el sistema domótico. Así mismo servirá como medio para medir el nivel de aceptación del mismo mediante los criterios de evaluación de usabilidad.

***Obligatorio**

1. Dirección de correo electrónico *

Datos Personales

2. ¿Ingrese Su Nombre? *

3. ¿Ingrese su numero de Cédula? *

Características del Sistema Domótico

1. Estructura de la Aplicación

4. 1.1 ¿Como considera la interfaz web del sistema domótico?.(ejm: botones, zonas de contenido, colores, etc). *

Marca solo un óvalo.

- Muy Buena
- Buena
- Regular
- Mala

2. Operación de la Aplicación

5. 2.1 Accesibilidad: ¿Las acciones que solicita el sistema domótico (activación de luces, apertura de puerta, encendido de ventiladores, etc) son fáciles de ejecutar? *

Marca solo un óvalo.

- Si
- No

17/12/2019

Encuesta General Sobre la Usabilidad del Sistema Domótico

6. 2.2 Sistema de Identificación: Se identifican fácilmente las figuras, hipertextos, las zonas activas y el tipo de acción que se debe ejecutar en el sistema domótico. *

Marca solo un óvalo.

- Si
 No

7. 2.3 Fiabilidad del sistema: Se presentaron errores durante la operación del sistema domótico. *

Marca solo un óvalo.

- Si
 No

8. 2.4 Intuición: Los procedimientos de navegación por el sistema domótico, así como la ejecución de tareas son fáciles de realizar: *

Marca solo un óvalo.

- Si
 No

3. Contenido del sistema Domótico

9. 3.1 ¿Como considera la densidad del contenido que se presenta en el sistema Domotico? *

Marca solo un óvalo.

- Extensa
 Normal
 Baja

10. 3.2 La información que se presenta en el sistema domótico es fácil de entender y memorizar? *

Marca solo un óvalo.

- Sí
 No

4. Experiencia de Usuario

11. 4.1 Le gustaría implementar este sistema Domótico en su Hogar? *

Marca solo un óvalo.

- Si
 No

12. 4.2 ¿Como Calificaría el Sistema Domótico Propuesto? *

Marca solo un óvalo.

- Muy Bueno
- Bueno
- Regular
- Malo

13. 4.3 Escriba una Opinión General Sobre el Sistema Domótico o Sugerencias. *

Anexo 8: Código Fuente

SketchFinal.ino

```
//Definición de librerías utilizadas.

#include <ESP8266WiFi.h> //librería para poder utilizar la placa NodeMCU

#include <PubSubClient.h> //librería MQTT para hacer publicaciones y Subscripciones a un
topico.

#include<dht.h> //librería para utilizar el sensor temperatura DHT11

dht DHT;

//definicion de los pines a utilizar.

#define led1 0 //luz sala

#define led2 1 //luz cocina

#define led3 3 //luz baño

#define led4 2 //luz dormitorio

#define moto1 13 //abrir puerta

#define moto2 15 //cerrar puerta

#define RELE 5 //encender ventiladores con sensor de temperatura

#define RELE1 4 // encender ventilador manualmente

#define led5 14 // luces emergencia

#define DHT11_PIN 16 //lectura de temperatura y humedad

#define puertoMqtt 1883 //puerto por defecto de MQTT.

WiFiClient clienteWifi;//este cliente se encarga de la comunicacion con el wifi

PubSubClient clienteMQTT(clienteWifi);//este utiliza el cliente anterior para hacer poder crear
la conexion mqtt

//si pasan por el hackerspace Xibalba pues ya tienen la clave

const char * ssid = "Tvcable_maria rodriguez";

const char * claveWifi = "mariarodriguez31";

const char * brokerMqtt = "167.99.233.34";// ip del bróker sin http ni nada solo los números

uint32_t ultimoIntentoReconexion;

uint32_t timerEnvioDatos;
```

```
long lastMsg = 0; //Variable para enviar datos de Temperatura

long lastMs = 0; //Variable para enviar datos de Humedad

char msg[50]; //Variable para recibir los datos obtenidos del sensor de Temperatura

char ms[50]; //Variable para recibir los datos obtenidos del sensor de Humedad

void conectarAlWifi() {

WiFi.begin(ssid, claveWifi);

Serial.print("conectando a");

Serial.println(ssid);

while (WiFi.status() != WL_CONNECTED) {

    delay(500);

    Serial.print(".");

}

Serial.println("");

Serial.println("Wifi Conectado ");

Serial.println("direccion IP: ");

Serial.println(WiFi.localIP());

}

void callback(char* topic, byte* mensaje, unsigned int length) {

String topico = topic;

Serial.print("Mensaje Recibido del topico: ");

Serial.println(topico);

Serial.print("mensaje : ");

for(uint8_t i =0;i<length;i++){

Serial.print(mensaje[i]);

}

if (topico == "/casa/sala/puerta/abrir") {

    digitalWrite(moto1, mensaje[0]-48);

    digitalWrite (moto2, HIGH);

    digitalWrite (moto1, LOW);

    delay(2200);

}
```

```
digitalWrite (moto2, LOW);  
  
}  
  
else if (topico == "/casa/sala/puerta/cerrar") {  
  
    digitalWrite(moto2, mensaje[0]-48);  
  
    digitalWrite (moto1, HIGH);  
  
    digitalWrite (moto2, LOW);  
  
    delay(2200);  
  
    digitalWrite (moto1, LOW);  
  
}  
  
else if (topico == "/casa/sala/ventilacion") {  
  
    digitalWrite(RELE1, mensaje[0]-48);  
  
    Serial.println("RELE1");  
  
}  
  
else if (topico == "/casa/sala/luz") {  
  
    digitalWrite(led1, mensaje[0]-48);  
  
    Serial.println("led 1 ");  
  
}  
  
else if (topico == "/casa/cocina/luz") {  
  
    digitalWrite(led2, mensaje[0]-48);  
  
    Serial.println("led 2 ");  
  
}  
  
else if (topico == "/casa/bano/luz") {  
  
    digitalWrite(led3, mensaje[0]-48);  
  
    Serial.println("led 3 ");  
  
}  
  
else if (topico == "/casa/dormitorio/luz") {  
  
    digitalWrite(led4, mensaje[0]-48);  
  
    Serial.println("led 4 ");  
  
}  
  
else {
```

```
Serial.println("error de mensaje");

}

}

boolean reconexion() {
Serial.print("Conectando al broker mqtt");

//intentando conectar al broker
if (clienteMQTT.connect("ESP8266Client")) {
Serial.println("Conectado");

//publicamos que estamos conectados
clienteMQTT.publish("/conexion", "Conectado");

//nos suscribimos a los topicos para controlar los ledes
clienteMQTT.subscribe("/casa/sala/puerta/abrir");
clienteMQTT.subscribe("/casa/sala/puerta/cerrar");
clienteMQTT.subscribe("/casa/sala/ventilacion");
clienteMQTT.subscribe("/casa/sala/luz");
clienteMQTT.subscribe("/casa/cocina/luz");
clienteMQTT.subscribe("/casa/bano/luz");
clienteMQTT.subscribe("/casa/dormitorio/luz");

delay (300);

} else {
Serial.print("falló, rc=");

Serial.print(clienteMQTT.state());
}

return clienteMQTT.connected();
}

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

Serial.println("iniciando programa");

pinMode(moto1, OUTPUT);

pinMode(moto2, OUTPUT);
```

```
pinMode(RELE, OUTPUT);
pinMode(RELE1, OUTPUT);
pinMode(led1, OUTPUT);
pinMode(led2, OUTPUT);
pinMode(led3, OUTPUT);
pinMode(led4, OUTPUT);
pinMode(led5, OUTPUT);
conectarAlWifi();

clienteMQTT.setServer(brokerMqtt, puertoMqtt); //le decimos cual es el servidor y el puerto al
que se debe conectar

clienteMQTT.setCallback(callback); //le decimos como se llama la funcion de callback
}

void loop() {
if (!clienteMQTT.connected()) {
if (millis() - ultimoIntentoReconexion > 5000) {
ultimoIntentoReconexion = millis();
// Attempt to reconnect
if (reconexion()) {
ultimoIntentoReconexion = 0;
} }
} else {
//cliente conectado
} clienteMQTT.loop();

long now = millis();
if (now - lastMsg > 5000) {
lastMsg = now;
int chk = DHT.read11(DHT11_PIN);
String msg= msg+ DHT.temperature;
char message[50];
msg.toCharArray(message,50);
```

```
Serial.println(message);

//publish sensor data to MQTT broker

clienteMQTT.publish("/casa/bano/temperatura",message);

String ms= ms+ DHT.humidity;

char messag[50];

ms.toCharArray(messag,50);

Serial.println(messag);

clienteMQTT.publish("/casa/bano/humedad",messag); }

int temp = DHT.temperature;

if(temp>=26){

    Serial.println("Ventiladores Encendidos");

    digitalWrite(RELE,LOW);

    temp = DHT.temperature;

    delay(2000);

} else

digitalWrite(RELE,HIGH);

int eme = DHT.temperature;

if(eme>=23){

    Serial.println("Luces de Emergencia Activadas");

    digitalWrite(led5, HIGH);

    delay(1000);

    digitalWrite(led5, LOW);

    delay(1000);

    eme = DHT.temperature;

    delay(2000);

} else

digitalWrite(led5,HIGH);

}
```


SensorPir.ino

```
#include <ESP8266WiFi.h>

#include <PubSubClient.h>

#include<dht.h> //librería para utilizar el sensor temperatura DHT11

  dht DHT;

#define inputPin 13

#define Status 16

#define led6 14

#define DHT11_PIN1 4 //lectura de temperatura y humedad2

const char* ssid = "Tvcable_maria rodriguez";

const char* password = "mariarodriguez31";

const char* mqtt_server = "167.99.233.34";

long lastMsg = 0; //Variable para enviar datos de Temperatura

long lastMs = 0; //Variable para enviar datos de Humedad

char msg[50]; //Variable para recibir los datos obtenidos del sensor de Temperatura

char ms[50]; //Variable para recibir los datos obtenidos del sensor de Humedad

String topic = "presence/";

int lastPirValue = LOW;

WiFiClient espClient;

PubSubClient client(espClient);

void setup_wifi() {

  pinMode(led6, OUTPUT);

  delay(10);

  // We start by connecting to a WiFi network

  Serial.println();

  Serial.print("Connecting to ");

  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
```

```
    delay(500);

    Serial.print("."); }

    randomSeed(micros());

    Serial.println("");

    Serial.println("WiFi connected");

    Serial.println("IP address: ");

    Serial.println(WiFi.localIP());
}

void reconnect() {
    // Loop until we're reconnected
    while (!client.connected()) {

        Serial.print("Attempting MQTT connection...");

        // Create a random client ID
        String clientId = "ESP8266Client-";

        clientId += String(random(0xffff), HEX);

        // Attempt to connect
        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");

            // Wait 5 seconds before retrying
            delay(5000);

        }

    }

}

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

    setup_wifi();
```

```
client.setServer(mqtt_server, 1883);

pinMode(inputPin, INPUT);

pinMode(Status, OUTPUT);

}

void loop() {

  if (!client.connected()) {

    reconnect();

  }

  client.loop();

  int pirValue = digitalRead(inputPin);

  if (pirValue != lastPirValue) {

    if (pirValue == HIGH) {

      digitalWrite(Status,LOW);

      String message = String(pirValue);

      client.publish(topic.c_str(), message.c_str());

    } else {

      digitalWrite(Status,HIGH);

      String message = String(pirValue);

      client.publish(topic.c_str(), message.c_str());

    }

    Serial.println(pirValue);

    lastPirValue = pirValue;

    delay(1000);

  }

  long now = millis();

  if (now - lastMsg > 9000) {

    lastMsg = now;

    int chk = DHT.read11(DHT11_PIN1);

    String msg= msg+ DHT.temperature;

    char message[50];
```

```
msg.toCharArray(message,50);

Serial.println(message);

//publish sensor data to MQTT broker

client.publish("/casa/cocina/temperatura",message);

String ms= ms+ DHT.humidity;

char messag[50];

ms.toCharArray(messag,50);

Serial.println(messag);

client.publish("/casa/cocina/humedad",messag);

}

int eme = DHT.temperature;

if(eme>=26){

    Serial.println("Luces de Emergencia Activadas");

    digitalWrite(led6, HIGH);

    delay(1000);

    digitalWrite(led6, LOW);

    delay(1000);

    eme = DHT.temperature;

    delay(2000);

} else

digitalWrite(led6,HIGH);

}
```