UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO

CARRERA DE INGENIERÍA DE SISTEMAS

MONITORIZACIÓN Y CONTROL DE SEGURIDAD EN HOGARES VULNERABLES EN EL SECTOR DE CHILLOGALLO DESDE DISPOSITIVOS MÓVILES MULTIPLATAFORMA, HACIENDO USO DE NODE-RED Y RASPBERRY PI4

Trabajo de titulación previo a la obtención del título de Ingenieros de Sistemas

AUTORES: **BORIS RAÚL CUTOS MANOTOA**
MIGUEL ÁNGEL RECALDE CHÁVEZ

TUTOR: **MANUEL RAFAEL JAYA DUCHE**

Quito-Ecuador

2022

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

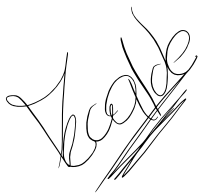
Nosotros, Boris Raúl Cutos Manotoa con documento de identificación N° 1720777919 y Miguel Ángel Recalde Chávez con documento de identificación Nº 1720195914 manifestamos que:

Somos los autores y responsables del presente trabajo: y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana puedan usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación

Quito, 09 marzo del 2022

Atentamente,

Boris Raúl Cutos Manotoa
1720777919



Miguel Ángel Recalde Chávez
1720195914

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Nosotros, Boris Raúl Cutos Manotoa con documento de identificación N.º 1720777919 y Miguel Ángel Recalde Chávez con documento de identificación Nº 1720195914, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del Proyecto Técnico: "Monitorización y control de seguridad en hogares vulnerables en el sector de Chillogallo desde dispositivos móviles multiplataforma, haciendo uso de Node-Red y Raspberry PI4", el cual ha sido desarrollado para optar por el título de: Ingenieros en Sistemas, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Quito, 09 marzo del 2022

Atentamente,

Boris Raúl Cutos Manotoa
1720777919

Miguel Ángel Recalde Chávez
1720195914

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Manuel Rafael Jaya Duche con documento de identificación Nº 1710631035 docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: MONITORIZACIÓN Y CONTROL DE SEGURIDAD EN HOGARES VULNERABLES EN EL SECTOR DE CHILLOGALLO DESDE DISPOSITIVOS MÓVILES MULTIPLATAFORMA, HACIENDO USO DE NODE-RED Y RASPBERRY PI4, realizado por Boris Raúl Cutos Manotoa con documento de identificación N.º 1720777919 y Miguel Ángel Recalde Chávez con documento de identificación N.º 1720195914, obteniendo como resultado final al trabajo de titulación bajo la opción de Proyecto Técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Quito, 09 Marzo del 2022

Atentamente,

_____

Ing. Manuel Rafael Jaya Duche, Mgtr

1710631035

DEDICATORIA

En estas líneas quiero agradecer a todas las personas que hicieron posible esta investigación y que de alguna manera estuvieron conmigo en los momentos difíciles, alegres y tristes. Estas palabras son para ustedes.

A mis padres Miguel Ángel Recalde Arauz y Mayra Lucia Chávez Castillo, por todo su amor, comprensión y apoyo, pero sobre todo gracias infinitas por la paciencia que me han tenido. No tengo palabras para agradecerles las incontables veces que me brindaron su apoyo en todas las decisiones que he tomado a lo largo de mi vida, unas buenas, otras malas, otras locas. Gracias por darme la libertad de desenvolverme como ser humano.

Gracias a mis Abuelitos, orgullosamente y con la cara muy en alto agradezco a Humberto Chávez Gordon y Clara luz Etelvina Castillo mi mayor inspiración, gracias a mis Abuelitos he concluido con mi mayor meta.

De igual forma, agradezco a nuestro director de Tesis, que gracias a sus consejos y correcciones hoy puedo culminar este trabajo. A los Profesores que me han visto crecer como persona, y gracias a sus conocimientos hoy puedo sentirme dichoso.

A todos mis amigos, vecinos y futuros colegas que me ayudaron de una manera desinteresada, gracias infinitas por toda su ayuda y buena voluntad.

AGRADECIMIENTO

Dicen que la mejor herencia que nos pueden dejar los padres son los estudios, sin embargo, no creo que sea el único legado del cual yo particularmente me siento muy agradecido, mis padres me han permitido trazar mi camino y caminar con mis propios pies. Ellos son mis pilares de la vida, les dedico este trabajo de titulación. Gracias mamita Mayra y papito Miguel.

Me gustaría agradecer en estas líneas la ayuda que muchas personas y colegas me han apoyado durante el proceso de formación universitaria, a quienes contribuyeron con esta investigación y redacción de este trabajo, a mi tutor, Rafael Jaya, por haberme orientado en todos los momentos que necesitamos sus consejos.

Así mismo, deseo expresar mi reconocimiento al comité central del barrio Santa Rosa en Chillogallo, mismo que nos brindó todas las atenciones e información brindada a lo largo de esta indagación.

A todos mis amigos, vecinos y futuros colegas que me ayudaron de una manera desinteresada, gracias infinitas por toda su ayuda y buena voluntad.

A la Universidad Politécnica Salesiana por ser la sede de todo el conocimiento adquirido en estos años.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El presente proyecto se enfoca en brindar un servicio de seguridad a viviendas de Chillogallo, integrando tecnologías de comunicación, monitorización y administración de componentes en un entorno controlado. Se ubicarán sensores de contacto, movimiento y golpe, que determinan el cambio de estado, en una puerta, ventana o espacio abierto. Estos sensores identificarán cuando un inmueble está siendo vulnerado, ya que los índices de inseguridad en la ciudad se han incrementado a nivel general, el crecimiento urbano ha dado paso a la proliferación de la delincuencia.

El sistema se gestionará desde un lenguaje de alto nivel llamado NODE RED que permite procesar señales de alerta generadas por el sistema y facilita la comunicación de forma de *bróker* de comunicación entre los circuitos y una aplicación móvil de gestión.

Con el presente proyecto, se brinda una alternativa tecnológica, versátil y cómoda para el control de los domicilios, brindando así un medio confiable y de control remoto para resguardar la propiedad privada, aumentando en gran medida el tiempo de acción de un ciudadano cuando sus vienes están siendo violentados.

ABSTRACT

This project focuses on providing a security service to homes in Chillogallo, integrating communication technologies, monitoring and component management in a controlled environment. Contact, movement and blow sensors will be located, which will determine the change of state, in a door, window or open space. These sensors will be identified when a property is being violated, since insecurity rates in the city have increased at a general level, urban growth has given way to the disappearance of crime.

The system will be managed from a high-level language called NODE RED that allows the processing of alert signals generated by the system and facilitates communication in the form of a communication broker between the circuits and a mobile management application.

With this project, a technological, versatile and comfortable alternative is provided for the control of homes, thus providing a reliable and remote control means to protect private property, greatly increasing the time of action of a citizen when they come. they are being violent.

INTRODUCCIÓN

ANTECEDENTES

Dentro de lo que a la delincuencia refiere, se puede definir varios niveles de la misma, pero el punto en el cual se centra el presente trabajo, refiere a los asaltos a domicilios, que es en sí, uno de los casos más sensibles de los cuales la sociedad aqueja, ya que fuera del síntoma negativo que representa un robo y/o asalto, un domicilio se referencia al trabajo de gran parte de la vida de los propietarios, y el sentimiento que este deja tras el hurto, deja mellas profundas en los dueños del inmueble.

Considerando que el robo de viviendas se define como *"la entrada ilícita y con fuerza sobre las cosas a una casa de habitación"* (Ministerio de Justicia y Paz, 2020), se ha tomado la ciudad capital Quito, para poder visualizar cuales fueron los números de atracos a domicilios, suscitados en los últimos años y en base a lo mencionado, el Señor coronel de la Policía de E.M Enrique Fernando Bautista Espín, comandante del distrito de Policía Eloy Alfaro, en calidad de custodio de la información referente a casos denunciados de robos a domicilios, compartió la información que se resume en la tabla 1

Tabla 1 *Índice de robos a domicilio*

COD. CIRCUITO	CIRCUITO	2014	2015	2016	2017	2018	2019	27_nov 2020
17D06C08	MENA	54	80	68	48	44	36	21
17D06C06	SOLANDA	53	36	32	14	13	15	10
17D07C01	CHILLOGALLO	75	74	64	72	51	44	29
17D07C04	QUITUMBE	37	30	26	30	36	33	17

Nota: Detalle de estadísticas sobre los números de robos en 4 barrios de Quito considerando los periodos entre 2014 y 2020.
Fuente: (Policía Nacional del Ecuador, 2020)

En un estudio realizado por el Instituto Nacional de Estadísticas y Censos (INEC), con una unidad de análisis a personas entre los 26 años que poseen viviendas propias en el área urbana del Ecuador (exceptuando la región insular), con una representatividad, nacional urbano, 24 provincias, 177 ciudades (con más de 2.000 habitantes). A los que se les realizó una encuesta obteniendo los siguientes resultados:

Cuatro de cada cien hogares han sido víctimas de robo a sus domicilios, se pone en consideración adicionalmente información relevante proporcionada por la unidad de delitos con la cual se está sustentado la problemática que repercute a la sociedad Quiteña, el hampa es de

los principales males que han aquejado a sus ciudadanos, siendo Chillogallo un suburbio que presenta uno de los mayores índices de delincuencia en la ciudad capital del Ecuador (Instituto Nacional de Estdítica y Censos, 2019).

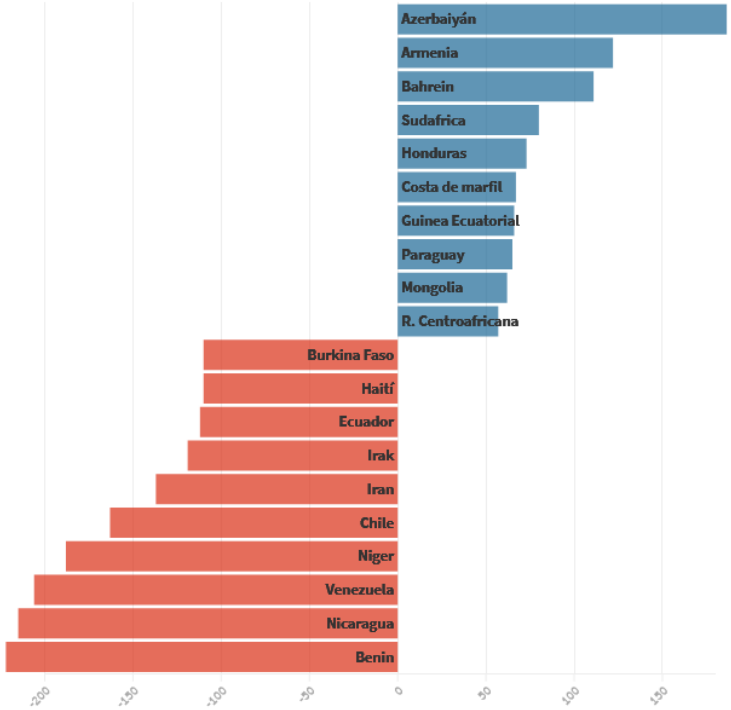
JUSTIFICACIÓN DEL PROYECTO

La delincuencia y la inseguridad forma parte del argot popular, a tal punto que la asimilación de este por la sociedad ha sido una necesidad, de lo contrario, la misma sociedad estaría enfrentándose a una verdad que posee historia y una forma consolidada en alta o baja organización dentro de las estructuras de toda urbe mundial, la delincuencia representa costos tanto individuales como sociales: pérdida de vidas, seguridad, propiedad, productividad; en definitiva, obstaculiza el crecimiento económico. En los últimos años, la opinión pública sobre el incremento de la delincuencia es una de las principales preocupaciones de América Latina y el Caribe, la región tiene el 9% de la población mundial y el 33% de sus homicidios. (Jaitman, y otros, 2017)

De acuerdo con la información de los últimos 10 años, Ecuador es uno de los países en América Latina que posee uno de los más altos índices de inseguridad y delincuencia a nivel mundial. El Índice de Paz Global que elabora el Institute for Economics & Peace (2020) expuso que cinco de los diez países que sufrieron mayores retrocesos en el último año pertenecen al continente americano, con Nicaragua y Venezuela a la cabeza como se detalla en la Figura 1, en el cual se representa en azul, los países que descontaron más puntos en relación con el índice de 2019 y se volvieron más pacíficos, mientras que en rojo, los países que sumaron más puntos entre un año y otro, lo que supone que se volvieron menos pacíficos. De esta manera, América Central y Sudamérica, tuvieron el mayor retroceso entre los años 2019 y 2020, y es explicado por un incremento de la milicia y un deterioro de seguridad interna de los países de las regiones mencionadas (Infobae, 2020).

En la Figura 1 se visualiza la puntuación de Ecuador, entre los países con puntuación negativa, en cuanto a seguridad a nivel mundial, haciendo alusión a que existen puntos de mejora que no han sido tomados en consideración por las autoridades competentes.

Figura 1 *Índice de Paz Global para el año 2020*



Nota: La puntuación de Ecuador es negativa en cuanto a seguridad a nivel mundial
Fuente: Global Peace Index (Institute for Economics and Peace, 2020)

OBJETIVOS DEL PROYECTO

Objetivo General

Monitorización y control de la seguridad en hogares vulnerables en el sector de Chillogallo desde dispositivos móviles multiplataforma, haciendo uso de Node-Red y Rasberry Pi4.

Objetivos Específicos

Desarrollar un sistema que permita al propietario de un hogar conocer el monitoreo y control de su bien inmueble.

Investigar y analizar el estado del arte con respecto al funcionamiento del protocolo MQTT de mensajería publicación/suscripción, de un sistema de control de la seguridad en hogares vulnerables.

Verificar la importancia de permitir al propietario de un domicilio controlar la seguridad de su hogar, por medio de la verificación el estado de los medios de acceso que el hogar posee.

Diseñar e implementar una aplicación multiplataforma gratuita, con la que el propietario de un domicilio pueda conocer en tiempo real el estado de su hogar.

Interpretar y analizar resultados para brindar un sistema de seguridad económico (gratuito en su inicio), a los moradores del sector de Chillogallo, de cuyos ingresos no les permita implementar un sistema de seguridad privado.

METODOLOGÍA

La metodología con la que se efectúo este proyecto fue SCRUM por las siguientes razones:

El equipo de Scrum lleva a cabo una sesión de planificación de Sprint en la que las tareas necesarias para completar los elementos de la lista de deseos se dividen en partes pequeñas y más fáciles de administrar.

El equipo se reúne todos los días para una breve reunión de Scrum (a menudo denominada Daily Standup) donde cada miembro del equipo comparte actualizaciones diarias, lo que ayuda al equipo y al gerente (tutor) de proyecto a evaluar el progreso del proyecto.

Scrum puede ayudar a los equipos a completar los entregables del proyecto de manera rápida y eficiente

El esfuerzo individual de cada miembro del equipo es visible durante las reuniones diarias de scrum.

Las partes interesadas y el propietario del producto realizan una revisión al final de cada sprint

Este es el ciclo que sigue un equipo Scrum en un proyecto de desarrollo de productos. Los tres roles mencionados anteriormente: el propietario del producto, el equipo Scrum y el Scrum Máster juntos juegan un papel importante en el ejercicio de este marco.

Cabe mencionar que la metodología Scrum viene dada para aplicarse en grandes o pequeños grupos de trabajo y es perfectamente acoplable a los equipos que deseen utilizarla, en otras palabras, es independientemente del número de integrantes de una célula de trabajo.

CAPÍTULO I

1. MARCO TEÓRICO

1.1. FUNDAMENTOS DE DOMÓTICA

El término domótica hace referencia al grupo de sistemas que permiten controlar y gestionar los procesos de una vivienda, agregando servicios de bienestar y comodidad para los propietarios del inmueble además de brindar mecanismos necesarios de comunicación con el sistema de la vivienda, los mismos que se integran a través de redes de comunicación con el objetivo de monitorear el hogar desde su interior o exterior (Palacios, 2020). El campo de la Domótica se ha transformado en una tendencia cuya demanda se fundamenta en construir viviendas integrando nuevas tecnologías, a través de sistemas sencillos y gestionables, de tal manera que garantice la seguridad, privacidad, y que sea un proceso transparente en el hogar y para los usuarios. Así, la domótica involucra la convergencia de los campos de gestión de las viviendas con la ingeniería informática, electrónica, la electricidad y las comunicaciones (Palacios, 2020).

1.1.1. Objetivos de la domótica.

La instalación domótica busca gestionar cuatro aspectos fundamentales en una vivienda:

Confort: se refiere a la automatización de los servicios que conforman los detalles de un hogar que permiten la comodidad del habitad de una casa como puede ser la refrigeración de los alimentos, control de cortinas y persianas, calefacción, etc. Lo cual se logra a través de pulsadores o por la creación de eventos programados que son ejecutados por el usuario en un momento determinado (Palacios, 2020).

Seguridad: es la configuración de sistemas automatizados que puede contener alarmas de tipo personales o técnicas, sensores de cercanía, movimiento, luz, agua, gas, fallos, además de cámaras de vigilancia etc. (Palacios, 2020).

Energía: en este campo los sistemas domóticos supervisan el consumo de energía con termostatos y relojes programadores para optimizar el uso de energía eléctrica, calefacción, refrigeración, etc. (Palacios, 2020).

Comunicaciones: el sistema domótico que será instalado en la vivienda se interconecta a la red de área local (LAN) con la finalidad de gestionar los dispositivos que en el sistema se encuentre. Esto permite el monitoreo del domicilio por su propietario desde cualquier lugar en

el que este se encuentre, además de permitirle un control a distancia del sistema (Palacios, 2020).

1.1.2. Dispositivos que conforman un sistema domótico

El diseño de un sistema domótico depende de su aplicación, por lo que su dimensión puede estar conformada por un único dispositivo, que realiza una sola acción, hasta amplios sistemas que automatizan prácticamente la mayoría de las instalaciones en el interior de un área residencial. De esta manera en un sistema domótico participan los siguientes actores:

***Controlador**:* este elemento gestiona el sistema domótico de acuerdo con la programación efectuado y los datos recibidos. Puede existir uno o múltiples controladores en todo el sistema.

***Actuador**:* se encarga de recibir una orden desde controlador (este es un dispositivo), tiene la capacidad de efectuar acciones sobre un dispositivo o un sistema completo como puede ser encender, apagar, apertura o cierre, etc.

***Sensor**:* el sistema domótico se encarga de monitorear el entornos interiores y exteriores de cualquier lugar que se desee, con el fin de recolectar información útil que es transmitido hacia al sistema. Entre los sensores más comunes que se instalan en una vivienda se puede mencionar: sensores de humo, temperatura, agua, etc.

***Bus**:* es el medio por el cual se transmite la información entre los distintos dispositivos que conforman el sistema domótico. Este medio de transmisión puede ser mediante cableado, redes de otros sistemas como redes inalámbricas, redes eléctricas, redes de telefónica o redes de datos.

***Interfaces**:* son dispositivos y formatos que permiten mostrar los datos generados por el sistema domótico hacia los usuarios u otros sistemas, permitiendo la interacción entre sistemas.

Figura 2. *Dispositivos que integran un sistema domótico*

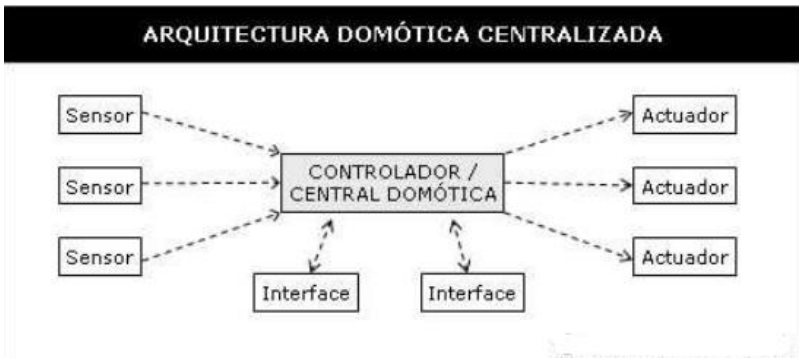
Nota: Detalle de los dispositivos que integran un sistema domótico
Fuente: (Barrio Ruiz, 2017)

1.1.3. Tipos de Arquitecturas

Se define una arquitectura dependiendo de la distribución y la ubicación de los dispositivos de control que contiene un sistema domótico. Los tipos de arquitectura más empleados se describen a continuación:

Arquitectura centralizada: se establece este tipo de arquitectura cuando existe solo un control máster que transmite la información a los dispositivos conectados como interfaces y actuadores según la configuración programada y la información que recibe de los dispositivos, usuarios y sistemas interconectados. De esta manera, el medio de transmisión es del tipo estrella ya que en su centro se encuentra la unidad central de control y no existe comunicación entre sensores y actuadores (Huidrobo Moya & Milan Tejedor, 2010).

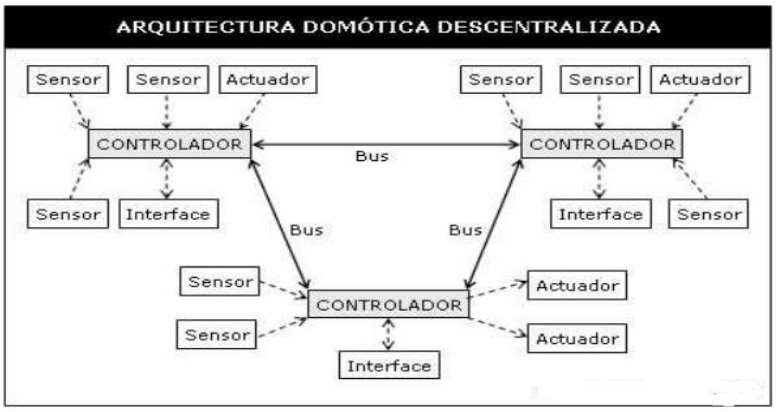
Figura 3. *Arquitectura domótica centralizada*



Nota: En la Figura **3***3 se muestra los elementos de esta arquitectura.*

Arquitectura descentralizada: se considera una arquitectura descentralizada cuando se instalan dentro del sistema más de un controlador, los mismos que tienen la capacidad de transmitir información entre los nodos dentro de la red y a los dispositivos conectados a controladores (Huidrobo Moya & Milan Tejedor, 2010).

Figura 4. *Arquitectura domótica descentralizada*

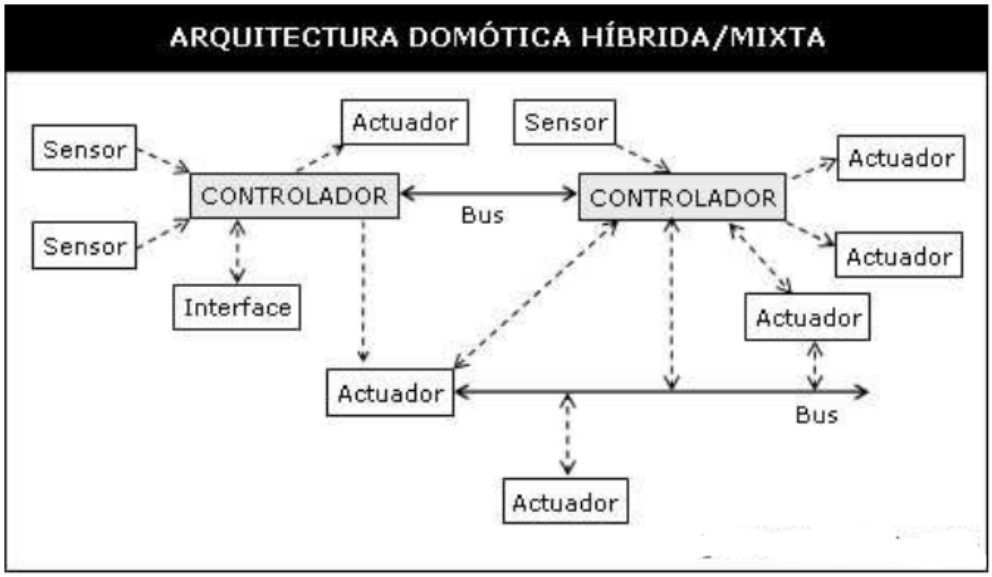


Nota: Detalla la arquitectura descentralizada.
Fuente: (Barrio Ruiz, 2017)

Arquitectura Híbrida: es una mezcla de arquitecturas de sistemas centralizados y distribuidos. Con esta arquitectura es posible disponer de un control máster o múltiples controles descentralizados. Además, las actuadores, sensores e interfaces también pueden cumplir la función de controladores, como lo es un sistema distribuido, permitiendo de esta manera a los dispositivos actuar de acuerdo con la configuración y la información que recolecta por sí

mismo, evitando así transmitir datos hacia otro controlador (Huidrobo Moya & Milan Tejedor, 2010).

Figura 5. *Arquitectura domótica híbrida*



Nota: Se muestra una distribución de la arquitectura híbrida.
Fuente: (Barrio Ruiz, 2017)

1.2. INTERNET DE LAS COSAS (IoT) Y DOMÓTICA

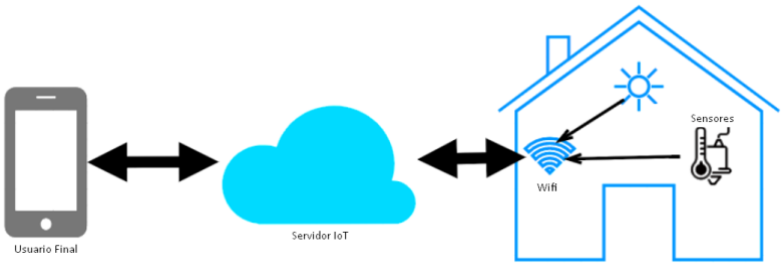
Aunque no existe una definición única para IoT que sea aceptable dentro de la comunidad mundial de académicos, investigadores, profesionales, innovadores, desarrolladores y personas corporativas, una definición que describe con exactitud a IoT es la siguiente: "IoT es una red abierta e integral de objetos inteligentes que tienen la capacidad de auto organizarse, compartir información, datos y recursos, reaccionando y actuando ante situaciones y cambios en el entorno" (Madakam, Ramaswamy, & Tripathi, 2015).

De esta manera, el IoT también se puede entender como una red global que permite la comunicación entre humano a humano, humano a dispositivo y dispositivo a dispositivo, empleando a Internet.

Internet es una red global compuesta de cientos de millones de redes públicas, privadas, empresariales, gubernamentales y académicas, que permiten la comunicación a nivel mundial mediante la implementación de tecnologías de redes ópticas, inalámbricas y electrónicas (Madakam, Ramaswamy, & Tripathi, 2015).

Actualmente, el IoT se ha iniciado a afectar el diario vivir de los seres humanos, brindando la capacidad de integrar tecnología en la mayor cantidad de actividades del día a día. Así, las viviendas (ver Figura 6)son los sitios donde se consume la mayor parte del tiempo, por lo que su automatización, usando un Smartphone, y poder controlar todo el entorno remotamente actualmente es totalmente viable debido a dos casos esenciales mencionados a continuación: La tecnología se comparte mediante el concepto de Open Source y segundo, reducción de los costes y del espacio físico que ocupan los circuitos integrados. De esta manera, constituir un hogar totalmente inteligente se lo realiza con el fin de conseguir comodidad, medicina mas económica, y sistemas de seguridad confiables (Karimi & Krit, 2018).

Figura 6. *Diagrama ilustrativo del sistema de monitoreo del hogar utilizando IoT*



Nota: Sistema de monitoreo
Fuente: (Al-Kuwari, y otros, 2018)

1.2.1. Protocolo de comunicación MQTT en IoT

MQTT (Message Queue Telemetry Transport) es un protocolo para el intercambio de mensajes entre un Cliente y un Servidor basado en subscripciones y publicaciones. Este protocolo proporciona funciones de mensajería sólidas para comunicarse con sistemas y dispositivos remotos, y también minimiza el ancho de banda de la red y los requisitos de recursos del dispositivo. El protocolo está diseñado para dispositivos en entornos restringidos, como sistemas integrados, teléfonos celulares y sensores con capacidad de procesamiento y memoria limitadas, y para sistemas que están conectados a redes no confiables (Cal Calleja, 2019).
Los elementos que participan en la comunicación MQTTP son los siguientes:

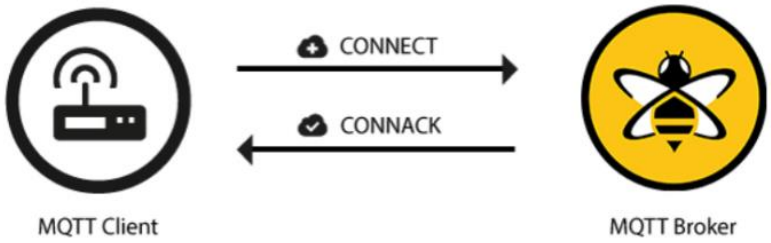
Cliente MQTT: es cualquier tipo de dispositivo que tiene la capacidad de ejecutar una biblioteca MQTT y que tiene la capacidad de conectarse a un agente MQTT por medio de una red. Por ejemplo, el cliente MQTT puede ser un dispositivo muy pequeño con recursos limitados que se conecta a través de una red inalámbrica y tiene una biblioteca mínima. El cliente MQTT también puede ser una computadora típica que ejecuta un cliente MQTT gráfico con fines de

prueba. Básicamente, cualquier dispositivo que habla MQTT sobre una pila TCP / IP puede llamarse cliente MQTT (Cal Calleja, 2019).

Bróker: es responsable de recibir todos los mensajes, filtrar los mensajes, determinar quién está suscrito a cada mensaje y enviar el mensaje a estos clientes suscritos. El bróker también guarda los datos de las sesiones y suscripciones persistentes de los clientes, incluidos los paquetes de comunicación perdidos. Otra de las actividades del bróker es la validación de acceso y concesión de permisos de los clientes (Cal Calleja, 2019).

Conexión MQTT: el protocolo MQTT opera sobre TCP / IP. De esta manera una conexión MQTT que se ejecuta en medio del cliente y el bróker. Para empezar la conexión, el cliente emite el mensaje CONNECT hacia el bróker. El bróker envía su mensaje de respuesta CONNACK además del código de estado. Una vez que establecida conexión, el bróker mantiene vigente dicha conexión hasta que el cliente emita el comando de desconexión o la conexión se viera interrumpida (Lekić & Gardašević, 2018).

Figura 7 *Diagrama de conexión MQTT entre un cliente MQTT y un Broker MQTT*



Nota: Diagrama de conexión MQTT
Fuente: (Cal Calleja, 2019)

1.3. PLATAFORMA DE HARDWARE PARA IoT

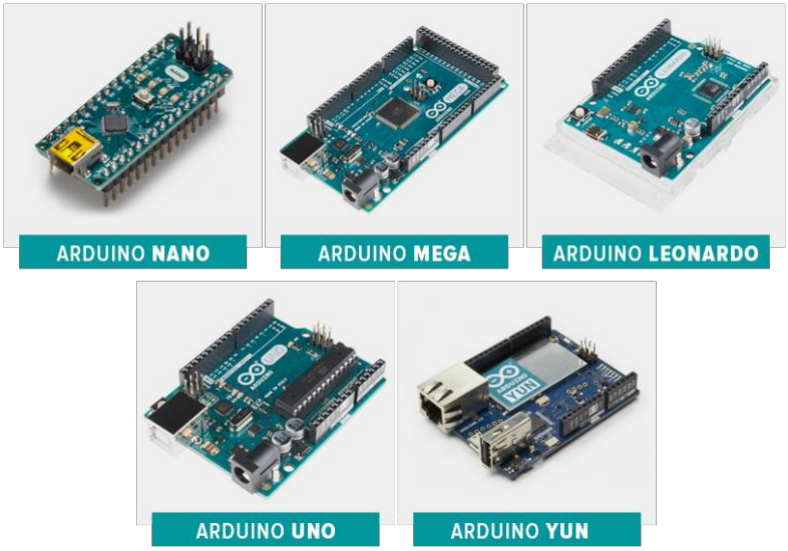
En el mercado se ofertan una gran variedad de SOC (System on a *Chip*) para aplicaciones de IoT, ya que son dispositivos que buscan integrar la mayor parte de los módulos de un computador o cualquier otro sistema informático o electrónico en un solo circuito integrado. Además, los SOC son dispositivos de bajo coste, bajo consumo de potencia, reducido tamaño. A continuación, se describe dos de las plataformas de hardware más usados en aplicaciones IoT.

1.3.1. Arduino

Es una placa electrónica de hardware libre, conformada por un microcontrolador reprogramable (varía de acuerdo con múltiples modelos ofertados en el mercado) y un set de pines integrados, que permiten la conexión de actuadores y sensores con microcontrolador de la placa, para

interpretación y manipulación de las señales emitidas. Debido a que Arduino es una las placas electrónicas populares a nivel mundial, se han fabricado varios modelos. La diferencia entre cada uno de los modelos radica en sus características como tamaño, número de pines Entrada/Salida, modelo de microcontrolador, etc. (Arduino, 2020).

Figura 8. *Modelos Arduino ofertados en el mercado*



Nota: Se muestra algunos de los modelos Arduino fabricados y ofertados en el mercado.
Fuente: (Arduino, 2020)

Entre las principales ventajas de implementar proyectos de IoT sobre las placas Arduino son:

Libre y extensible: al ser una placa electrónica de hardware libre, Arduino es libre y extensible. Esto se traduce a que cualquier individuo es libre de modificar, mejorar o ampliar el diseño del hardware (Arduino, 2020).

Amplia comunidad: arduino cuenta con una gran comunidad que brinda soporte a los diferentes modelos. Esto permite que exista una vasta documentación para comenzar a trabajar sobre cualquier plataforma (Arduino, 2020).

Multiplataforma: arduino puede ser instalado y manejado desde las diversas plataformas tradicionales como Windows, Mac y Linux.

Lenguaje de programación intuitivo: arduino cuenta con su propio lenguaje de programación. El mismo está basado en C++, lo cual lo convierte en un lenguaje de fácil comprensión y con una curva de aprendizaje relativamente alta (Arduino, 2020).

Bajo costo: el costo de las placas electrónicas es relativamente económico. Incluso el usuario final podría construirla por sí mismo, ya que, al ser libre, existe todo tipo de documentación que contribuye a lograr dicho objetivo (Arduino, 2020).

1.3.2. Raspberry PI

Plantea una alternativa de hardware para proyectos que ofertan IoT, debido a que es un ordenador de precio moderado y de cómoda portabilidad debido a su tamaño. Raspberry Pi tiene la capacidad de efectuar los mismos procesos de computo que un ordenador de escritorio, como navegar por la web, edición de documentos de texto incluso es compatible para juegos de video. Esta plataforma se desarrolló con el fin de que el público investigue múltiples campos de la informática de manera cómoda y llamativa, integrando en sus capacidades el uso de dispositivos electrónicos combinados con lenguajes de programación de alto nivel como Scratch, Java, JavaScript o Phyton (Rasperry, 2020).

La plataforma ha sido aplicada en varios campos de la ingeniería por lo que se lo considerado en el desarrollo de varios proyectos como implementación de una estación meteorológica, estación de videojuegos, centro multimedia, chatbot, etc. Además, en el mercado se oferta distintos modelos de placa Raspberry,

Figura 9. *Modelos de placas Raspberry PI ofertados en el mercado*

	Raspberry Pi 3B	Raspberry Pi 3B+	Raspberry Pi 4B	Raspberry Pi Zero	Raspberry Pi Zero W	Raspberry Pi A+	Raspberry Pi 3A+
Procesador	Broadcom BCM2837	Broadcom BCM283780	Broadcom BCM271180	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM283780
CPU	1.2Ghz Quad Core	1.4Ghz Quad Core	1.5Ghz Quad Core	1Ghz Single Core	1Ghz Single Core	700MHz Single Core	1.4Ghz Quad Core
GPU	400MHz	400MHz	500MHz	No aplica	No aplica	400MHz	400Mhz
RAM	1GB	1GB	1GB/2GB/4GB	512MB	512MB	512MB	512MB
USB	4 USB 2.0	4 USB 2.0	2 USB 2.0/ 2 USB 4.0	1 Micro USB	1 Micro USB	1 USB 2.0	1 USB 2.0
Video	Jack, HDMI	Jack, HDMI	Jack,2 Micro HDMI	Micro HDMI	Micro HDMI	Jack, HDMI	Jack, HDMI
Audio	Jack, HDMI	Jack, HDMI	Jack, micro HDMI	Micro HDMI	Micro HDMI	Jack, HDMI	Jack, HDMI
Memoria	Micro SD	Micro SD	Micro SD	Micro SD	Micro SD	Micro SD	Micro SD
Wireless	Wifi 2.4 Ghz Bluetooth 4.1BLE	Wifi 2.4 y 5 Ghz Bluetooth 4.2BLE	Wifi 2.4 y 5 Ghz Bluetooth 5.0BLE	No aplica	Wifi 2.4 Ghz Bluetooth 4.0	No aplica	Wifi 2.4 y 5 Ghz Bluetooth 4.2BLE
Ethernet	Ethernet 10/100	Gigabit Ethernet 300	Gigabit Ethernet 1000	No aplica	No aplica	No aplica	No aplica
Alimentación	5V 2.5Amp	5V 2.5Amp	5V 3Amp	5V 2Amp	5V 2Amp	5V 2Amp	5V 2.5Amp
GPIO	40 Pines	40 Pines	40 Pines	40 Pines	40 Pines	40 Pines	40 Pines
Dimensiones	85x56x17 mm	85x56x17	88x58x19 mm	65x30x5	65x30x5	65x56x12	65x56x12 mm

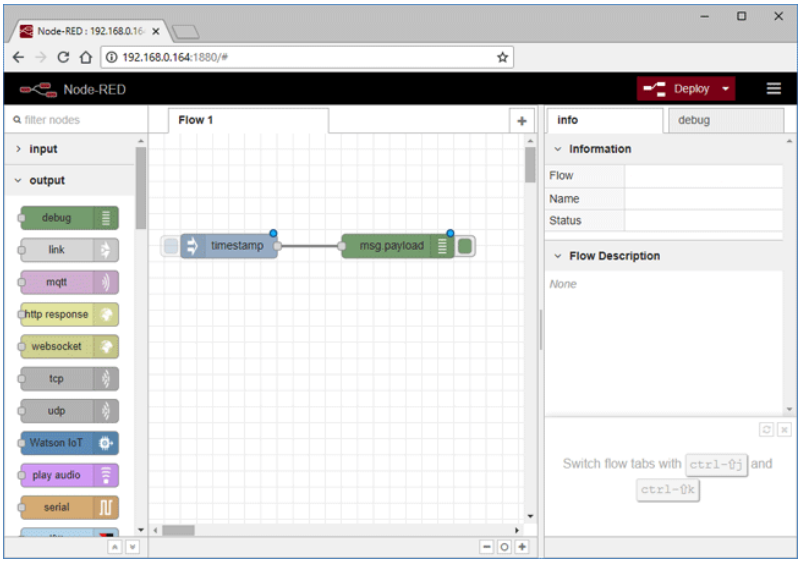
Nota: Se muestra los modelos más demandados.
Fuente: (MCI Electronics, 2020)

1.4. NODE – RED

Diseñada por IBM en conjunto con el departamento de Servicios de Tecnologías Emergentes. Al contrario de lo que pueda parecer, es un software sencillo e intuitivo de utilizar y extremadamente compatible con Raspberry Pi. Node-red es una potente herramienta utilizada para la comunicación entre distintos dispositivos de hardware y múltiples servicios de una forma eficiente y veloz, además de simplificar la programación gracias a su interfaz gráfica de programación, mitigando la complejidad que surge se desea integrar hardware con servicios

externos. Node-RED está creado a partir de NodeJS y la librería de JavaScript D3.js (Lekić & Gardašević, 2018).

Figura 10. *Entorno de Node - RED*



Nota: Nodos de comunicación input – output
Fuente: (Lekić & Gardašević, 2018)

1.4.1. NodeJS

Ideado como un entorno de ejecución de JavaScript orientado a eventos asíncronos, Node.js está diseñado para el desarrollo de aplicaciones network escalables, proporciona la potencia suficiente para que Node-RED sea fiable y escalable. NodeJS es un software muy potente que permite la programación en JavaScript del lado del servidor. Su ventaja más importante es que está optimizado para poder tratar múltiples conexiones concurrentes de una manera óptima (Lekić & Gardašević, 2018).

1.4.2. D3.js

D3.js es el encargado de la interfaz web. Node-RED es accesible a través de un navegador, es decir, sólo necesitamos acceder a una página web para poder crear nuestros propios sistemas. No hace falta instalar ningún entorno de desarrollo o IDE. En esta aplicación web podremos arrastrar y conectar nodos con diferentes funcionalidades para crear flujos de datos que hagan una determinada cosa (Lekić & Gardašević, 2018).

1.5. FIREBASE

Firebase es una plataforma utilizada para diferentes propósitos, de las cuales la principal es el almacenamiento de datos de manera flexible y con una escalabilidad en almacenamiento configurable, de esta manera se garantiza que los usuarios de esta plataforma no se queden sin espacio para almacenar toda la información que se desee guardar.

Creada por Google, posee uno de los mejores soportes a nivel mundial, además de garantizar una experiencia al usuario incomparable, sus tiempos de respuesta con cualquier dispositivo o programa que desee acoplarse a la plataforma. Posee una ampla gama de herramientas de entre las cuales se pueden destacar 3 de las múltiples que se ofertan en Firebase.

- Realtime Database
- Cloud Messaging
- Funciones acoplables los nodos de datos que se crean en la Realtime Database.

1.5.1. Realtime Database.

Como su nombre lo indica, es una base de datos que trabaja en tiempo real. De manera no relacional esta base de datos permite que los datos de cualquier proveedor puedan ser almacenados sin necesidad de tener una secuencia o dependencia de algún parent, concepto muy utilizado en las bases de datos relacionales.

Este tipo de base de datos tiene la capacidad de almacenar su información de manera ligera similar a un formato JSON a manera de nodos, únicamente es necesario conocer que datos son los que se van a almacenar en cada nodo, el orden lógico es implícito al usar Realtime Database en la plataforma de Firebase, ya que esta se encargará de asignar un identificador único a cada grupo de información, además es innecesario la declaración de los tipos de variables y las longitudes que estas van a requerir en la Realtime Database ya que esta se encargará de declarar en la memoria virtual toda la información de los datos a almacenar

Figura 11. *Nodos de almacenamiento de datos Firebase*

Nota: Nodo de almacenamiento HouseInputs
Elaborado por: Boris Cutos, Miguel Recalde

Posee una API sin costo que permite acceder a todos los métodos ofertados por el protocolo HTTP. De los más conocidos:

- PUT
- GET
- POST

Estos métodos son los que permitirán interactuar con la información de la Realtime Database para poder administrarla a voluntad de la persona encargada de manipular dichos datos. Para acceder a la API, únicamente es necesario obtener el nodo al que se va a acceder y hacer un click sobre el mismo para conocer la URL y proceder con la interacción de los nodos.

Figura 12. *URL del Nodo de Realtime Database*

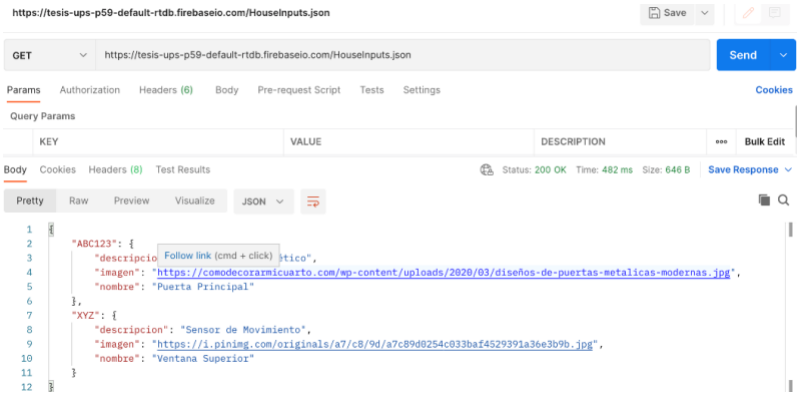
Nota: Dirección de almacenamiento notificaciones push
Elaborado por: Boris Cutos y Miguel Recalde

Como se puede observar en la imagen figura 12, se muestra que la URL del nodo en la que se encuentra el grupo de información referente a los accesos de un domicilio es la siguiente:

https://tesis-ups-p59-default-rtdb.firebaseio.com/HouseInputs

Con el conocimiento de la URL, nada más basta con acceder a ella con cualquier cliente HTTP. Como lo puede ser POSTMAN. A continuación, un ejemplo de un GET de la información almacenada en el nodo expuesto en la Figura 12. URL del Nodo de Realtime Database.

Figura 13. *Petición GET realizada al Nodo de información en FIREBASE*



Nota. Se utiliza el cliente PostMan para referir una petición de tipo Get a la API de FIREBASE
Realizado por: Boris Cutos y Miguel Recalde.

En la Figura 13 se presenta como la información es obtenida desde Firebase. Esta petición está siendo realizada mediante un GET por medio del protocolo HTTP, mismo que retorna una cadena de caracteres mediante el formato JSON.

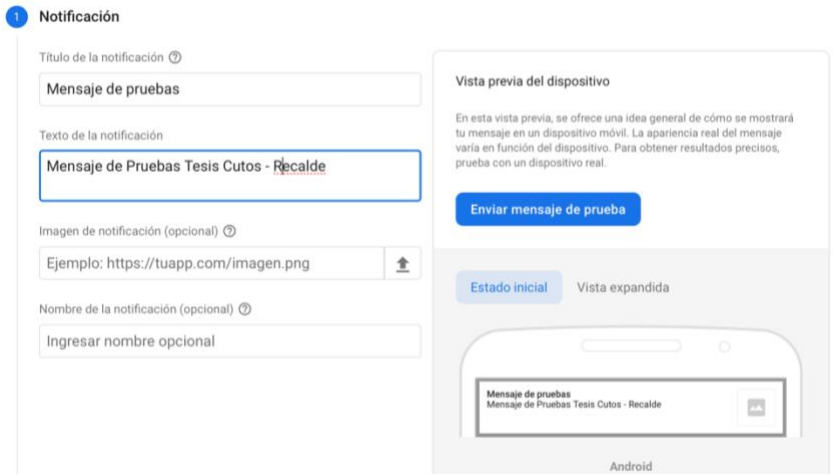
Una vez que la información se encuentra en el formato anteriormente presentado, únicamente basta que se la manipule a conveniencia de la persona o grupo de desarrolladores que se encuentren trabajando con la plataforma de Firebase para la presentación o muestra de la información que se encuentra requiriendo. De la misma manera se puede trabajar con un método POST o un método PUT.

Firebase o, la funcionalidad de Firebase que se está utilizando, Realtime Database, tiene herramientas que permiten que en tiempo real se obtengan métricas de las actividades que se están realizando, notificando a través de la consola de administración a los clientes o a su vez enviando notificaciones de tipo PUSH a los clientes que se encuentren suscritos a las notificaciones, o se encuentren en la lista de las funciones que hacen notificaciones a los clientes.

1.5.2. Cloud Messaging.

Es una de las funcionalidades más llamativas y potentes que posee Firebase, la cual consiste en enviar notificaciones de tipo PUSH a todos los clientes que se encuentren suscritos dentro del pool de dispositivos (móviles y web)

Figura 14. *Consola de Notificaciones PUSH en Firebase*



Nota: Consola de Notificaciones PUSH en Firebase
Realizado por: Boris Cutos y Miguel Recalde, mediante la consola de notificaciones PUSH de Firebase (Firebase Google, 2021)

Una vez seteado todos los valores necesarios en la notificación, se deberá presionar el botón de "Enviar mensaje de prueba" para que el motor lógico de Firebase proceda a la(s) notificaciones.

1.5.3. Funciones Firebase

Las funciones de Firebase son de diferentes usos, todo depende de las necesidades del equipo que se encuentre trabajando con los nodos de la Realtime Database descritos en las secciones anteriores. Uno de los usos en los cuales se pueda aplicar el módulo de funciones de Firebase, es notificar a todos los clientes (web o móvil) cuando una nueva inserción de data se ha realizado cualquiera de los nodos de la base de datos.

Uno de los usos más comunes para las funciones de notificaciones por cada inserción de data sobre los nodos de la Realtime Database, son los lives chats. Cada vez que se haga un ingreso de información, se podrá notificar a los escuchas que existe un nuevo mensaje en la cola que ameritan una revisión.

Otro de los usos (para este proyecto en específico) sería enviar notificaciones a los escuchas cada cuando se haga una inserción en el nodo de la Realtime Database, siempre que esta

información proceda de un circuito cerrado, que avise que alguno de los accesos del circuito sea vulnerados o abiertos. A continuación, un ejemplo.

Figura 15. *Funciones que interactúan con Firebase*

```
const functions = require("firebase-functions");
const admin = require("firebase-admin");

admin.initializeApp(functions.config().functions);

let newData;
exports.messageTrigger1 = functions.database.ref('SecurityDomotics/{id}').onCreate(async (snapshot, context) => {
    if (snapshot.empty) {
        console.log("No Devices");
        return;
    }
    newData = snapshot.data;
    let tokens = [
        "fCoahPj6QNeCWgLn4P19mn:APA91bFFqT_e6RuqJoKjXKYuvGoPX_aVhrOSUqPOP9Ehp5fIZBruPXb28Vf8nPFIGA6msBY0WlSa36eqQoAga7YOin54c9ot_tsmiQrINTqOC-
    ];
    let payload = {
        notification: {
            title: 'AMENAZA DETECTADA',
            body: 'Forzaron la seguridad del domicilio',
            sound: 'default',
        },
        data: {
            product: 'EL acceso se vulnero',
        }
    };
    try {
        const resp = await admin.messaging().sendToDevice(tokens, payload);
        console.log("se envio el mensaje");

    } catch (err) {
        console.log("error al enviar tu nota");
    }
});
```

Nota: Función para la conexión a la base de datos
Elaborado por: Boris Cutos y Miguel Recalde.

En la función expuesta en la Figura 14 se puede observar que se realiza una llamado a la base de datos.

```
functions.database.ref('SecurityDomotics/{id}').
```

Se instancia una nueva conexión a la base de datos y esta quedará a la escucha de cada cuando se realice una actividad en particular como la creación, eliminación o actualización de la Data dentro del nodo de la Realtime Database. (Firebase Google, 2021)

1.6. LENGUAJE DE PROGRAMACION DART

Es un lenguaje de código abierto desarrollado por Google permitiendo a la comunidad de desarrolladores utilizar un paradignma de conocimiento general el cual es la programación orientada a objetos, lo que simplifica muchos de los conceptos tradicionales para la programación de aplicaciones móviles, ya que unifica los paradigmas de los lenguajes comúnmente conocidos como lo son HTML y JS, normalmente utilizados para el desarrollo de proyectos web, y últimamente aplicados en programación de aplicaciones para teléfonos.

Dart es un lenguaje de programación fuertemente tipado. Es decir, cada valor que usa en su sintaxis tiene su tipo de dato primitivo, ya sea una cadena o un número, estos deben de ser reconocidos de manera clara al momento de compilar el código.

A continuación, se detalla los tipos de datos primitivos que usa dart, y como estos esta estructurados.

Tabla 2 *Tipos de datos primitivos Dart*

Tipo de Dato	Bits	Obs
Integer	Desde -2^{63} Hasta $2^{63} - 1$	N/A
Double	64-bit	Especificados en los estándares de la IEEE 754
Strings	N/A	Estandarizados por la codificación UTF-16
Booleans	N/A	Puede tener solo dos estados TRUE y FALSE

Nota: Tamaño en bits para el tipo de datos
Realizado por: Boris Cutos y Miguel Recalde*, en base a la documentación de Dart* (dart, 2021)

Asimismo, los conceptos básicos ya conocidos por cualquier individuo que tiene nociones de programación serán de lo más sencillos de comprender como por ejemplo los conceptos de funciones, ya que estas son un conjunto de declaraciones que realizan una tarea especifica, mismas que están organizados en bloques lógicos de código que se pueden leer, mantener y reutilizar.

Dart al igual que los lenguajes de programación de alto nivel posee un Framework con altas prestaciones que permiten una experiencia de programación centralizada, cómoda y de alto rendimiento para los desarrolladores de software. Flutter es el framework por excelencia que maneja Dart, este permite la encapsulación de líneas extensas de código en un solo bloque denominado “Widget” mismo que puede ser reutilizado con un concepto fácil de entender, el cual es la programación orientada a objetos. (dart, 2021)

La programación orientada a objetos tiene una amplia trayectoria dentro de los lenguajes de alto nivel en el área del desarrollo de software ya que implementa un paradigma simple y fácil de entender por cualquier programador. El paradigma de la POO permite realizar una abstracción de la realidad, permitiendo la reutilización de muchos de los componentes del código a través de múltiples instancias que pueden ser definidas en los “n” lugares en los cuales estas instancias sean requeridas, permitiendo una codificación mas limpia a la hora de realizar una programación más estructurada y simple de entender.

La programación orientada a Objetos dentro de Dart tiene una variante adicional, la cual permite que esta sea encapsulada dentro de un concepto propio de Dart, la cual son los widgets. Los Widgets utilizados en Dart son comúnmente para representar un componente con una funcionalidad especifica, que esta puede ser utilizada o consumida por otros Widgets y que pueden comportarse de maneras distintas, este concepto es comúnmente es conocido como herencia y polimorfismo es lenguajes como Java.

La facilidad con la cual Dart ha implementado las mejores funcionalidades de otros lenguajes muy conocidos por su potencia al momento de ejecución, define el éxito con en el cual Dart se ha posicionado en el mercado, imponiéndose incluso como una opción más eficaz y eficiente que React Native o Ionic.

1.6.1. FLUTTER (flutter, 2021)

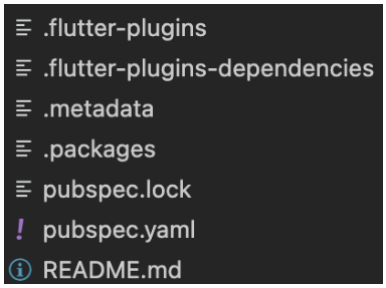
Flutter es el framework por excelencia diseñado para la interacción con Dart, optimizando la sintaxis utilizada por su lenguaje raíz. Posee el mejor tiempo de ejecución dentro de los frameworks ofertados en el mercado además de integrar múltiples herramientas gratuitas fáciles y livianas de utilizar para depurar y debuggear el código.

Flutter transforma el proceso de desarrollo de aplicaciones desde una perspectiva cómoda y eficiente para los desarrolladores ya que posee librerías fáciles y cómodas de usar e instalar desde un repositorio virtual, con el soporte de cientos de desarrolladores a nivel mundial que realizan constantes contribuciones a la comunidad. (pub.dev, 2021)

Para implementar una o más dependencias (librerías) para el proyecto que se esté desarrollando, únicamente basta con añadir la librería correspondiente a la necesidad de los programadores, mediante una sencilla búsqueda dentro del repositorio virtual de Google Flutter “https://pub.dev”.

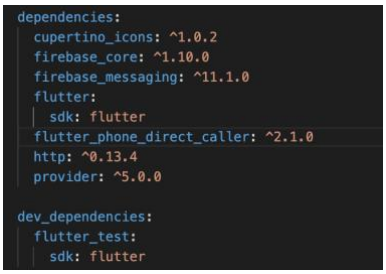
Para añadir una nueva librería es necesario abrir el archivo “pubspec.yaml”, ubicado en la raíz del proyecto Flutter creado y añadir la dependencia.

Figura 16. *Archivos de configuración en la raíz del proyecto Flutter*.



Nota: Archivos de configuración
Elaborado por: Boris Cutos y Miguel Recalde

Figura 17. *Dependencias del proyecto Flutter.*



Nota: Dependencias para la creación de app
Elaborado por: Boris Cutos y Miguel Recalde

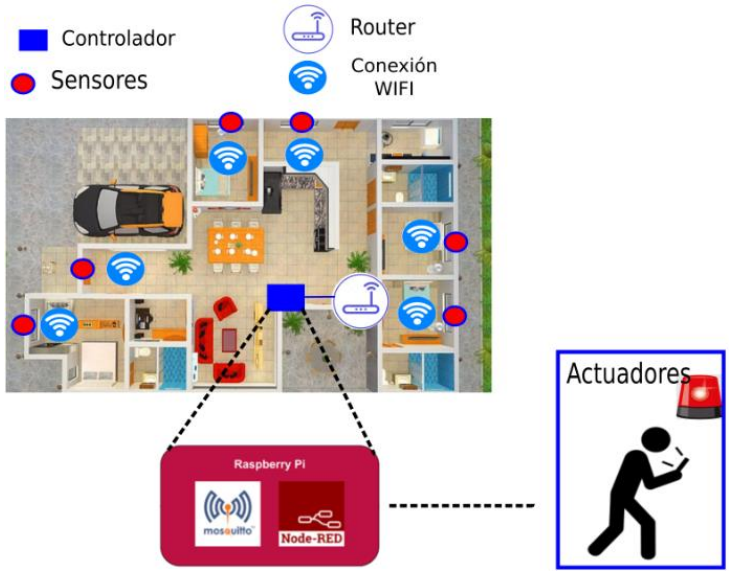
El uso de estas dependencias permite crear aplicaciones mucho más versátiles y rápidas en tiempos de desarrollo ya que la necesidad de crear algo desde cero, a menos que sea estrictamente demandante, es totalmente innecesario.

CAPÍTULO II

2. DISEÑO DEL SISTEMA DOMÓTICO DE SEGURIDAD

En esta sección se muestra el diagrama del prototipo electrónico que permitirá brindar seguridad a viviendas a través de una arquitectura de domótica centralizada, es decir se establece una entidad central denominada controlador encargada de recibir las notificaciones provenientes de los diversos sensores, para procesar la información y posteriormente emitir órdenes hacia los actuadores. En la Figura 18 se resumen cada uno de los componentes del sistema de domótica diseñado para el monitoreo de seguridad en una vivienda.

Figura 18. *Diagrama del sistema domótico de seguridad para una vivienda*



Nota: Diagrama del sistema domótico
Elaborado por: Boris Cutos, Miguel Recalde

El sistema domótico estará conformado por los siguientes tres elementos:

Nodos Sensores: estos dispositivos serán ubicados de forma estratégica dentro de la vivienda y su función principal es la de monitorear el estado de las puertas y ventanas. Cada nodo sensor se constituye por un módulo ESP8266 el cual es un microcontrolador encargado de monitorear cada uno de los sensores que se encuentran a su cargo. En varios de los pines del microcontrolador se instalarán interruptores magnéticos, sensores de detección de movimiento y sensores de detección de vibración, con el objetivo de generar mensajes de notificación que se enviarán al controlador del sistema domótico a través del protocolo MQTT (Message Queue Telemetry Transport) utilizando la red WIFI de la vivienda.

Controlador: Para el controlador del sistema domótico se emplea la Raspberry PI 4 B sobre la cual se instala las siguientes funcionalidades:

- Sobre la Raspberry PI 4B se configura un servidor MQTT, el cual se encarga de gestionar los mensajes de alerta provenientes de cada uno de los nodos sensores que han sido instalados dentro de la vivienda cuando algunos de los interruptores magnéticos, sensores de movimiento o vibración detecten una situación sospechosa.
- Además, se instala el software Nod-RED con el objetivo de notificar a los actuadores del sistema domótico que se ha violado la seguridad de la vivienda.

Actuadores e interfaces: los actuadores del sistema domótico son los Smartphones que permitirán notificar a los usuarios que la seguridad de sus viviendas ha sido vulnerada, a través de una aplicación móvil multiplataforma diseñada en este proyecto de titulación.

2.1. SENSORES DE MONITOREO DE VENTANAS Y PUERTAS

Cada uno de los sensores se conforman por el microcontrolador del módulo ESP8266 el cual monitorea el estado de los interruptores magnéticos y enviará notificaciones al controlador del sistema domótico. A continuación, se detallará las especificaciones técnicas de cada uno de los elementos que conforman el sensor

2.1.1. Interruptor Magnético

En el mercado se oferta una variedad de dispositivos para la funcionalidad de la detección de apertura de puertas y ventas, tanto a nivel de viviendas como a nivel industrial. De esta manera, con el objetivo de brindar seguridad dentro de áreas residenciales se ha optado por diseñar un prototipo electrónico con interruptores magnéticos MC-38 que se instalan cerca de los cerrojos de puertas y ventanas. En la Figura 19, se muestra el sensor MC-38.

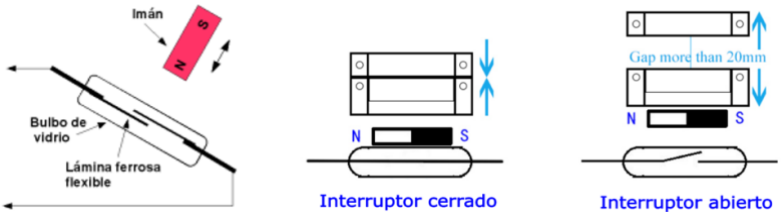
Figura 19. *Sensor MC-38: Interruptor Magnético*

Nota: Sensor magnético
Fuente: (AV Electronics, 2021)

El principio de funcionamiento de este sensor se basa en el uso de contactos de lengüeta, cuyas placas contribuyen a un contacto eléctrico debido a la presencia de un campo magnético, creado mediante un bulbo de vidrio con gas inerte diseñado. La superficie de la placa ha sido tratada

con materiales ferrosos adecuado para circuitos de baja corriente o alta inductancia, creando señales de salida en base al estado en el que se encuentra circuito, dicho de otra forma, si las placas se encuentran juntas, estas generan en el circuito un estado cerrado, pero si estas se separan, se interpreta como un estado abierto.

Figura 20. *Principio de funcionamiento del interruptor magnético*



Nota: Principio de funcionamiento del sensor MC-38.
Fuente: (Deeter Electronics, 2021)

La salida de un sensor de campo magnético puede ser un voltaje de salida que se relaciona linealmente con la fuerza del campo magnético que incide en el detector, o binaria en la que la salida toma dos estados, el primero cuando la fuerza del campo magnético medida es mayor a un umbral particular establecido, y el segundo cuando la intensidad del campo magnético medida se encuentra por debajo del umbral. En la **Error! Reference source not found.** se resume las especificaciones técnicas del sensor MC-38.

Tabla 3 *Especificaciones técnicas del sensor MC -38*

Especificaciones técnicas	
Voltaje de operación	3.3 [V] a 100 [V] DC
Amperaje	0.5 [A]
Distancia activación interruptor	15 - 25 mm
Dimensiones del interruptor	34 x 41 x 6.5 mm

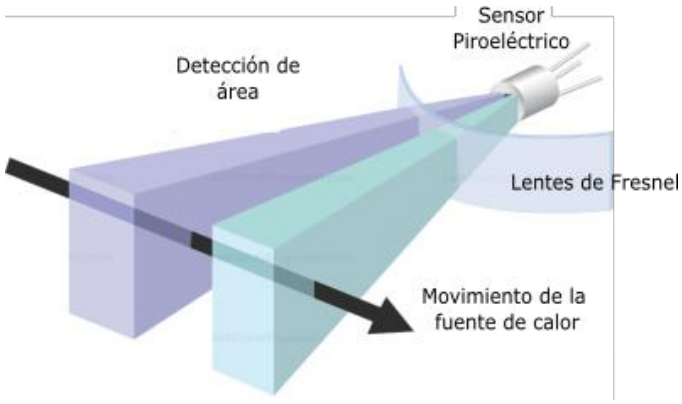
Nota: Resumen de las especificaciones técnicas del Datasheet del sensor MC-38
Elaborado por: Boris Cutos, Miguel Recalde

2.1.2. Sensor de detección de movimiento

Para la detectar cualquier movimiento en la vivienda, se ha optado por utilizar el módulo sensor infrarojo HC-SR501. Este sensor tiene la funcionalidad de cuantificar la radiación electromagnética infrarroja (RI) de los cuerpos presentes dentro de su campo de visión. Básicamente consta de dos partes principales: un sensor piroeléctrico y un lente Fresnel que enfoca las señales infrarrojas en el sensor piroeléctrico. Un sensor piroeléctrico contiene dos ranuras rectangulares por donde atraviesa la radiación infrarroja hacia dos electrodos sensores de infrarrojos separados, uno responsable de producir una salida positiva y el otro una salida

negativa con el objetivo de cuantificar un cambio en los niveles de RI y no en los niveles de RI ambientales (Jost, 2019). En la Figura 21 se ilustra el principio de detección de movimiento con sensor infrarrojo.

Figura 21. *Principio de funcionamiento de un sensor infrarrojo HC-SR501*



Nota: principio de detección de movimiento con sensor infrarrojo
Fuente: (LAST MINUTE ENGINEERS, 2021)

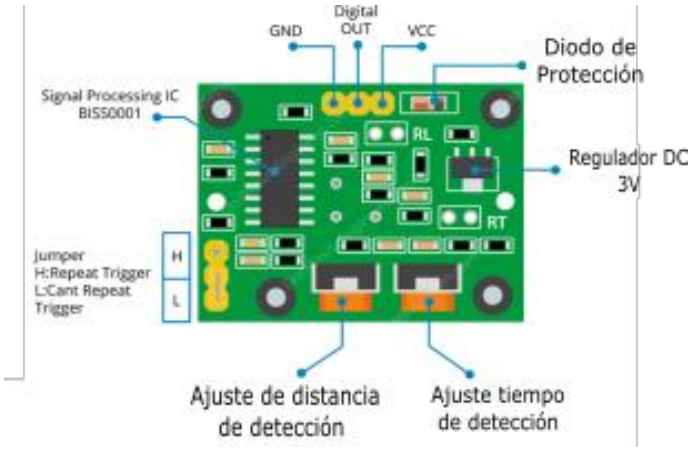
Se debe tener en consideración que para el uso correcto del sensor infrarrojo HC-SR501, es necesario configurar los siguientes parámetros de acuerdo a los requerimientos de la vivienda:

Sensibilidad: El sensor puede detectar el movimiento en un rango de distancia que va desde 3 metros hasta aproximadamente 7 metros. Aunque la topología de su habitación puede afectar el rango real que alcance.

Tiempo: A través de este parámetro se establece cuánto tiempo permanecerá en estado ALTO la señal de salida después de la detección. Como mínimo es de 3 segundos, como máximo es de 300 segundos o 5 minutos.

En la Figura 22 se observa la estructura del sensor de detección IR y los parámetros anteriores se modifican por los potenciómetros ajuste de distancia de detección y ajuste de tiempo de detección y las demás especificaciones técnicas de este sensor se detallan en los anexos de este proyecto de titulación.

Figura 22. *Estructura del sensor de detección RI*

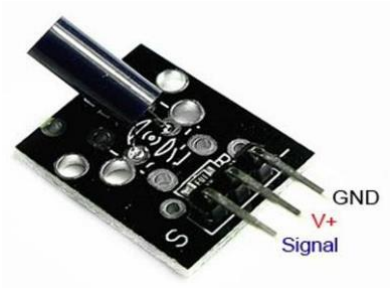


Nota: Estructura del sensor de detección IR
Fuente: (LAST MINUTE ENGINEERS, 2021)

2.1.3. Sensor de detección de vibración

En el sistema domótico de este proyecto de titulación integra además sensores que permitan la detección de las ondas de impacto asociadas con la rotura de una ventana o puerta. De esta manera, cuando se detecta una onda de choque grande, el sensor se activa y posteriormente se envíe un mensaje de alerta al sistema de alarma para informarle sobre la situación. Para esta funcionalidad se ha optado por el sensor digital KY002, el cual se conectará a los pines digitales del módulo ESP8266 y actuará como interruptor en un estado cerrado cuando detecta una vibración o una fuerza externa, proporcionando un cambio de voltaje en la señal digital de baja en su pin de salida. Además, KY002 es un sensor de vibración capacitivo con una resistencia de 10kΩ. El consumo de corriente es de 10 mA y el voltaje máximo permitido que admite es de hasta 12 V. En la figura 23 se indica la estructura de pines del sensor KY002.

Figura 23. *Estructura física del sensor de vibración KY002*



Nota: Sensor de vibración KY002
Fuente: (Leantec. Robotics. Arcade. Electronics, 2019)

2.1.4. Diagrama de Circuito de detección de apertura de puertas y ventanas

Para el diseño de los nodos sensores del sistema domótico se considera el uso del módulo NodeMCU ESP8266, el cual es un entorno de desarrollo de software y hardware de código abierto que se ha desarrollado sobre un chip SoC (System on a Chip) de bajo costo, que además incluye elementos cruciales que caracterizan a una computadora como CPU, RAM, tarjeta de red WIFI e incluso un moderno sistema operativo con un conjunto de herramientas de desarrollo de software (Yuan, 2021). Estas características convierten al módulo NodeMCU ESP8266 en una excelente opción para la creación de proyectos de Internet de las cosas (IoT) en todos los ámbitos. Las especificaciones técnicas de módulo NodeMCU se resumen en la **Error! Reference source not found.**

Tabla 4 *Especificaciones técnicas NodeMCU ESP8266*

Especificaciones técnicas NodeMCU ESP8266

Velocidad de reloj	80 MHz
Voltaje de entrada	5 - 10 [V]
Voltaje de operación	3.3 V
Flash Memory / SRAM	4 MB / 64 KB
Pines digitales I/O	11
Pines Analógicos	1
Puertos Seriales UART/SPI/I2C	SI
WIFI	Estándar 802.11 b/g/n
Temperatura de operación	-40C - 125C
Puerto USB	Micro USB
Puerto USB a Serial	CP2102

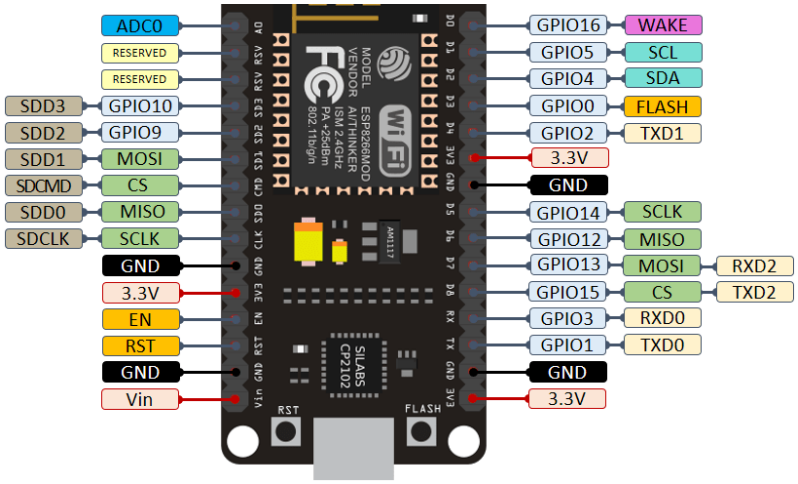
Nota: Resumen de las especificaciones técnicas del Datasheet de NodeMCU ESP8266
Elaborado por: Boris Cutos, Miguel Recalde

Para la conexión de los interruptores magnéticos, sensores magnéticos y sensores de vibración al módulo NodeMCU ESP8266 se considera los pines de propósito general (GPIO). Estos pines pueden ser configurados vía software con interfaces de entrada para que puedan reconocer una tensión de 0V o 3.3V. Si son configurados como interfaces de salida, la corriente máxima que proporciona cada GPIO es de 12[mA].

De los 17 pines GPIO que posee este módulo es recomendable configurar hasta 8 de ellos como pines de entrada o salida, porque se consideran 6 pines (GPIO 6 a GPIO11) para conectar el chip de memoria Flash. Mientras que los GPIO 1 y 3 se utilizan como pines de transmisión y recepción (RX/TX) del puerto serie de hardware (UART), por lo que, en la mayoría de los

casos, no es recomendable tampoco utilizarlos como pines de entrada/salida mientras envía / recibe datos en serie (ESP8266 Shop , 2021). En la Figura 24 se detalla gráficamente la posición de los pines del módulo NodeMCU ESP8266.

Figura 24. *Distribución de pines para el módulo NodeMCU ESP8266*



Nota. La funcionalidad de los pines se detalla en el Apéndice
Fuente: (Hardwarelibre, 2021)

Para el diseño del nodo sensor se conectará al módulo NodeMCU ESP8266 los sensores que se detallan en la **Error! Reference source not found.**.

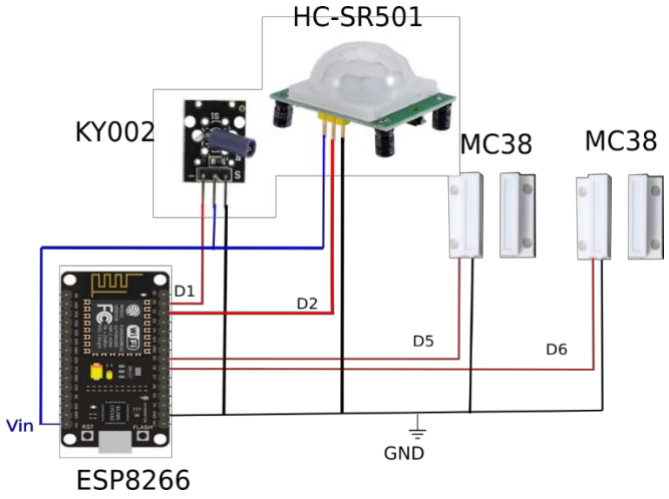
Tabla 5 *Pines del módulo NodeMCU ESP8266 que se consideran para conectar los sensores*

Puerto digital en el NodeMCU ESP8266	Número GPIO	Sensores digitales
D1: Pin de entrada	GPIO 5	Sensor de vibración KY002
D2: Pin de entrada	GPIO 4	Módulo infrarojo HC-SR501
D5: Pin de entrada	GPIO 14	Interruptor Magnético MC-38
D6: Pin de entrada	GPIO 12	Interruptor Magnético MC-38

Nota: Pines del módulo NodeMCU ESP8266
Elaborado por: Boris Cutos, Miguel Recalde

En la Figura 25 se detalla el diagrama circuital para un nodo sensor que se instalará en regiones estratégicas de la vivienda. También es necesario considerar la activación de las resistencias internas pull-up de cada pin que se configura como interfaz de entrada con el propósito de controlar el voltaje y corriente que ingresa a los pines cuando se activa alguno de los sensores. Todos los pines del módulo NodeMCU ESP8266 desde GPIO 0 hasta GPIO15 tienen una resistencia pull-up incorporada y el pin GPIO16 tiene una resistencia pull-down incorporada.

Figura 25. *Diagrama del circuito para el monitoreo del estado de puertas y ventanas*



Nota: Diagrama circuital para un nodo sensor
Elaborado por: Boris Cutos, Miguel Recalde

2.2. CONTROLADOR DEL SISTEMA DOMÓTICO

2.2.1. Raspberry PI 4 modelo B como controlador del sistema domótico

Actualmente, la Raspberry Pi se ha convertido en una opción conveniente de plataforma de hardware para el diseño de proyectos relacionados con la IoT, por poseer las características de un ordenador de un tamaño portable y de bajo coste. La Raspberry Pi actuará como el controlador del sistema domótico, dado que contiene dispositivos de entrada y salida como cualquier computador, así como también un conjunto de pines GPIO (General Purpose Input/Output), y estos son los mismos que brindan la capacidad de conectar actuadores y sensores.

Las características técnicas de la Raspberry PI 4 B se detallan en la Tabla 6.

Tabla 6. *Especificaciones técnicas de la placa Raspberry PI 4 modelo B*

Especificaciones técnicas Raspberry PI 4 B	
CPU + GPU	Quad core 64-bit ARM-Cortex A72 running at 1.5GHz
RAM	2 GB
WI-FI	5 - 10 [V]
BLUETOOTH	802.11 b/g/n/ac Wireless LAN
ETHERNET	Bluetooth 5.0 BLE
GPIO	1 puerto Gigabit Ethernet
HDMI	Cabecera extendida de 40 pines
USB	SI (2 puertos que admiten pantallas duales con una resolución de hasta 4Kp60)
WIFI	2 puertos 2.0 y 2 puertos 3.0
PUERTO CSI (CAMERA SERIAL INTERFACE)	SI (Permite conectar una cámara)
PUERTO DSI (DISPLAY SERIAL INTERFACE)	SI (Permite conectar una pantalla táctil)
PUERTO MICRO SD	SI (Para cargar el SO y almacenar datos)
PUERTO MICRO USB	SI (Para alimentar la placa 5V / 2.5 DC)

POE (POWER OVER ETHERNET)	SI

Nota. Resumen de las especificaciones técnicas del Datasheet de Raspberry PI 4 modelo B
Elaborado por: Boris Cutos, Miguel Recalde

Dado las características de RAM y CPU que contiene la Raspberry PI4 se lo emplea como el controlador central del sistema domótico de seguridad para ello se instala el sistema operativo Raspbian con el propósito de configurar un servidor MQTT para la gestión de mensajes de notificación provenientes de cada nodo sensor instalado y el software NOD-RED el mismo que permitirá interconectar el servidor MQTT y la base en tiempo real Firebase para el envío de notificaciones a la aplicación móvil.

2.3. COMUNICACIÓN ENTRE SENSORES Y CONTROLADOR USANDO MQTT

Para la interacción entre los sensores de monitoreo y el controlador, se ha seleccionado el protocolo MQTT, debido a que es flexible y portable lo permitiéndolo ser ejecutado por dispositivos con limitados recursos, usando redes con altas o bajas prestaciones, ajustándose perfectamente con el paradigma de Internet de las Cosas. Considerando que MQTT también puede ser implementado sobre el modelo TCP/IP, la interconexión de los dispositivos que conforman el sistema domótico de seguridad planteado en este proyecto se realizará sobre la red WI-FI instalada en cada una de las viviendas.

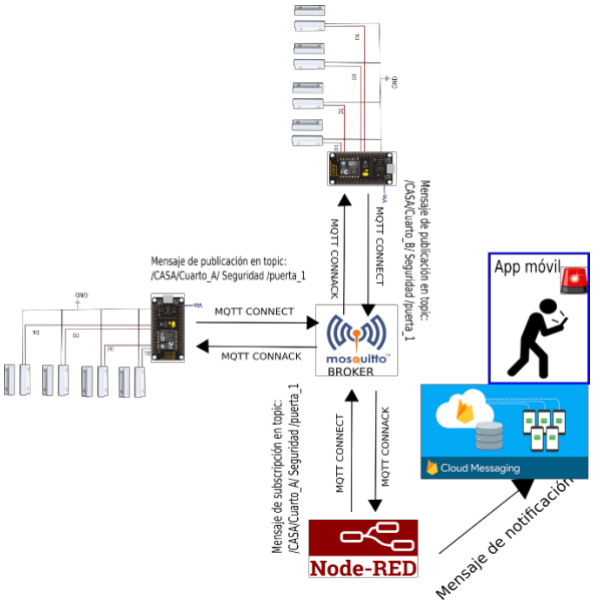
En la

Figura 26 se muestra la arquitectura del sistema domótico, donde se establece al controlador del sistema como el Bróker MQTT y como clientes MQTT a cada uno de los sensores de monitoreo del estado de puertas y ventanas. Es decir, cada uno de los nodos sensores se conectan al bróker MQTT enviando mensajes CONNECT con su identificador único de cliente y le notifica al bróker si se trata de una sesión será o no persistente.

En caso de que la sesión sea persistente, el bróker almacenará las suscripciones y los mensajes pertinentes a las suscripciones que se han perdido considerando Calidad de Servicio (QoS). En

el caso de que el bróker necesite autenticación, este será tomado del mensaje que posee la información de user y password, pero se debe tener en consideración que esta información viene en texto plano y lo que lo hace inseguro por no poseer cifrados. Para solucionar esto se hace uso de TLS o certificados SSL (HiveMQ, 2021).

Figura 26. *Diagrama de comunicación del sistema domótico de seguridad*



Nota: Diagrama de la arquitectura del sistema domótico
Elaborado por: Boris Cutos, Miguel Recalde

El protocolo MQTT contiene 14 tipos de mensajes, pero normalmente un cliente MQTT la mayor parte del tiempo emplea mensajes **CONNECT, PUBLISH, SUBSCRIBE Y UNSUBSCRIBE**. En la Tabla **7**, se detalla los distintos tipos de mensajes del protocolo MQTT.

Tabla 7. Tipos de mensaje en el protocolo MQTT

Tipo de mensaje	Dirección del flujo	Descripción del mensaje
CONNECT	Cliente a servidor	Petición de conexión cliente a servidor
CONNACK	Servidor a cliente	Respuesta a un mensaje CONNECT
PUBLISH	Cliente a servidor Servidor a cliente	Mensaje de Publicación
PUBACK	Cliente a servidor Servidor a cliente	Respuesta a un mensaje PUBLISH
PUBREC	Cliente a servidor	Publicación recibida

	Servidor a cliente	
PUBREL	Cliente a servidor Servidor a cliente	Publicación actualizada
PUBCOM	Cliente a servidor Servidor a cliente	Publicación completa
SUBSCRIBE	Cliente a servidor	Petición de subscripción del cliente
SUBACK	Servidor a cliente	Respuesta a un mensaje SUBSCRIBE
UNSUBSCRIBE	Cliente a servidor	Petición de no subscripción
UNSUBACK	Servidor a cliente	Respuesta a un mensaje UNSUBSCRIBE
PINGREQ	Cliente a servidor	PING Request
PINGRESP	Servidor a cliente	PING Response
DISCONNECT	Cliente a servidor	Cliente está desconectado

Nota: Detalle de los distintos tipos de mensajes del protocolo MQTT
Fuente: (Corinne , Kate, & Alexander , 2021)

los clientes enviando mensajes CONNACK, el cual incluye un código para indicar el estado de la conexión comunicando además si el cliente ya tenia una sesión establecida o no. En la Tabla 8 se muestra los distintos tipos de códigos que puede incluir un mensaje CONNACK.

Tabla 8. *Códigos de mensaje CONNACK enviados por el bróker*

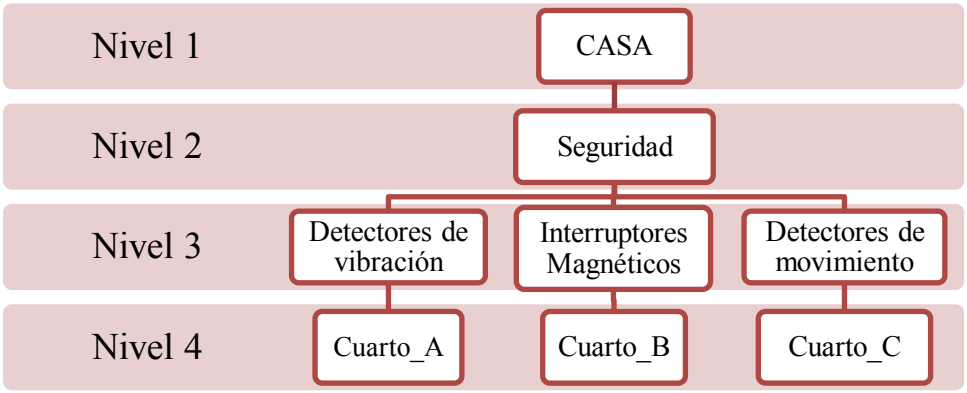
Código CONNACK	Estado de la conexión
0	Conexión aceptada
1	Conexión rechazada, versión de protocolo erróneo
2	Conexión rechazada, identificador rechazado
3	Conexión rechazada, servidor no disponible
4	Conexión rechazada, usuario o contraseña incorrectos
5	Conexión rechazada, no autorizado

Nota: Tipos de códigos que puede incluir un mensaje CONNACK
Fuente: (Corinne , Kate, & Alexander , 2021)

Una vez establecida conexión de tipo MQTT, el bróker utiliza el identificador para clasificar y distribuir los flujos de comunicación a los subscritores del sistema, dicho de otra forma, el bróker una vez que recibe un flujo desde el publicador, verifica identificador del flujo entrante con el contenido del flujo de los suscritores, que se han suscrito a algún topic y reenvía el flujo a los suscriptores a dicho topic.

Cada uno de los topic, se jerarquizan con el distintivo “slash” (‘/’), permitiendo una estructura ordenada y legible, lo que facilita agregar o retirar elementos de la red del sistema sin afectar a los otros elementos. En la Figura **27** se muestra un ejemplo de la jerarquización de los topic de cada uno de los sensores que se instalarán en una vivienda para monitorear el estado de las puertas y ventas, para la cual se ha estructurado 4 niveles de organización de acuerdo con la ubicación dentro de la vivienda.

Figura 27. *Jerarquía del topic para los sensores de monitoreo de puertas y ventanas*



Nota: Jerarquización de los topic de cada sensor
Elaborado por: Boris Cutos, Miguel Recalde

La jerarquización de los topics de la Figura **27** es representado por el bróker en el siguiente formato:

- /CASA/Seguridad/Detector_vibración/Cuarto_A/

Los topics se organizan en 4 niveles para construir un sistema domótico escalable de tal manera que un futuro se pueda añadir otro tipo de sensores de acuerdo con la situación que se desee monitorear dentro de la vivienda. De esta manera el Nivel 2 de la jerarquía está destinado para desplegar otras áreas del sistema domótico como monitoreo de temperatura, optimización de energía eléctrica o control automático de encendido o apagado de dispositivos dentro de la vivienda. El nivel 3 se destina al tipo de sensor a utilizar para cada área de monitoreo. Mientras que el nivel 4 del topic hace referencia a la ubicación de los sensores en el interior de la vivienda.

2.3.1. Configuración del bróker Mosquitto sobre la Raspberry PI 4

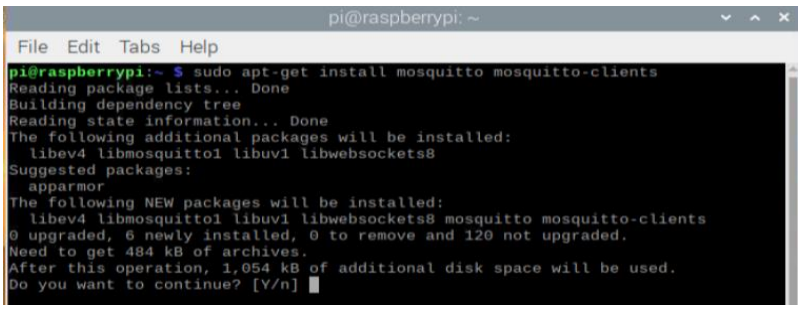
Para configurar el modelo cliente/subscriptor en el sistema domótico de seguridad se implementa el bróker Eclipse Mosquitto en la Raspberry PI, el cual es liviano y desarrollado en C, y se lo utiliza en todos los últimos releases del protocolo MQTT, por lo que se puede instalarlo en un dispositivo de capacidades ligeras, hasta un computador de alto rendimiento. Para la instalación de Eclipse Mosquitto sobre el sistema operativo Raspbian únicamente se debe ejecutar los siguientes tres comandos:

- ***sudo apt update***
- ***sudo apt upgrade***

- ***sudo apt-get install mosquitto mosquitto-clients***

En la **Error! Reference source not found.** se muestra la terminal de Raspbian insertando el comando "**sudo apt-get install mosquitto mosquitto-clients"** para la instalación de Eclipse Mosquitto.

Figura 28. *Instalación de Eclipse Mosquitto*



Nota: Comando para instalar Eclipse Mosquitto
Elaborado por: Boris Cutos, Miguel Recalde

Si se desea ejecutar Eclipse Mosquitto después de arrancar el sistema, es necesario insertar en la consola de Raspbian el comando "**sudo systemctl enable mosquitto.service**" como se indica en la Figura 29.

Figura 29. *Ejecución de Eclipse Mosquitto al reinicio de Raspberry*

Nota: Ejecución de Eclipse Mosquitto
Elaborado por: Boris Cutos, Miguel Recalde

2.3.2. Seguridad en la conexión MQTT:

Un aspecto importante en las comunicaciones en escenarios de IoT es el nivel de seguridad que se emplea en el establecimiento de las conexiones entre los dispositivos. Es posible establecer una conexión MQTT segura a través de los siguientes enfoques:

Autenticación Usuario/Contraseña: El protocolo MQTT proporciona sistemas de seguridad intuitivos basados en user y password. El cliente tiene la opción de enviar un nombre de usuario y una contraseña cuando se conecta a un bróker de MQTT. Cuando el nombre de usuario y la

contraseña se establecen en el cliente, la información se envía al bróker en texto plano. Este texto es vulnerable a escuchas clandestinas y proporciona una manera fácil para que los atacantes obtengan las credenciales. La transmisión segura de nombres de usuario y contraseñas requiere cifrado de transporte (The HiveMQ Team , 2015).

Seguridad SSL / TLS: TLS es un protocolo criptográfico que en el proceso de autenticación negocia varios parámetros para establecer una comunicación cifrada entre el cliente y el servidor, generando así un canal de comunicación seguro, garantizando que el contenido de su comunicación no pueda ser leído ni alterado por terceros. Para ello se considera una Autoridad de Certificación (CA) encargada de la generación de certificados digitales que serán considerados para el establecimiento de cada una de las conexiones cifradas. El formato de estos certificados está especificado por el estándar X.509 (The HiveMQ Team, 2015).

Lista de control de acceso (ACL): Las ACL son la función que permite controlar el acceso a los topics realizando comprobaciones en los mensajes PUBLISH y SUBSCRIBE enviados desde el cliente MQTT. De esta manera los diferentes eventos tienen tipos de ACL asociados y cada ACL tiene un topic, una prioridad y se puede configurar para permitir o denegar la realización de un evento (The HiveMQ Team, 2015).

Dado que el protocolo MQTTT es un protocolo que se ha desarrollado específicamente para recursos limitados de computación en IoT, el SSL/TLS no es una opción óptima para el sistema domótico de seguridad planteado en este proyecto por las siguientes razones:

- Dependiendo de los cifrados utilizados, TLS es muy intensivo en computación y requiere varios kilobytes de memoria, lo cual reduciría el rendimiento del módulo ESP8266 porque es un dispositivo con recursos de computación limitado.
- Dado que se instalará varios sensores, se generan muchas reconexiones hacia el bróker MQTT, la cual provoca una sobrecarga muy significativa en cada reanudación de la sesión, especialmente si está publicando mensajes MQTT muy pequeños.

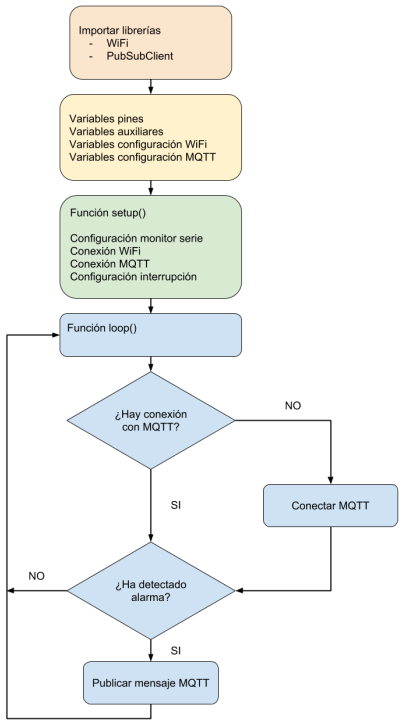
El esquema de seguridad planteado para el sistema domótico es realizar la autenticación usuario/contraseña entre los nodos sensores y el bróker Mosquitto complementado con el uso de Lista de Control de Acceso a cada uno de los topics donde los nodos sensores publican los mensajes.

2.3.3. Configuración cliente MQTT en el módulo NodeMCU ESP8266

Sobre los módulos NodeMCU ESP8266 se debe instalar la librería de Arduino PubSubClient, la misma que permitirá configurarlos como clientes MQTT y de esta manera tengan la capacidad de publicar mensajes y suscribirse a un topic o varios para recibir mensajes.

En la Figura **30** se muestra el diagrama de flujo de tareas que el microcontrolador del módulo NodeMCU ESP8266 ejecutará constantemente, para esto, su configuración se lo realiza por medio de lenguaje de programación C++ utilizando el IDE de Arduino.

Figura 30. *Diagrama de flujo de tareas del microcontrolador ESP8266*



Nota: Flujo de tareas del microcontrolador
Elaborado por: Boris Cutos, Miguel Recalde

De acuerdo con el diagrama de la Figura 30, lo primero que debe establecer el microcontrolador es la conexión a la red local WIFI para poder estar en comunicación con el servidor MQTT. En la Figura 31 se muestra el código para el establecimiento de la conexión WIFI, para lo cual se ha creado la función **setup_wifi ()**.

Figura 31. *Código para el establecimiento de conexión WIFI*

```
NodoSensor

void setup_wifi() {
  /*
  Esta función permite establecer la conexión local WIFI.
  */
  delay(10);
  Serial.println();
  Serial.print("Conectando a ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi conectado");
  Serial.println("dirección IP: ");
  Serial.println(WiFi.localIP());
}
```

Nota: Nodo de configuración wifi
Elaborado por: Boris Cutos, Miguel Recalde

Después de que el módulo NodeMCU ESP8266 se ha conectado a la red local, se dispara el proceso MQTT con Mosquitto, este último funciona como bróker de conexión. En la Figura 32 se muestra la función con la cual el módulo envía un mensaje CONNECT hacia el bróker MQTT.

Figura 32. *Código de Arduino para la conexión MQTT*

```
NodoSensor §

void setup_mqtt_conexion(){
  /*
   Esta función configura la conexión con el servidor mqtt
   */
  clientMqtt.setServer(servidorMqtt, 1883);
  Serial.println("[MQTT]Conectado al servidor MQTT");


  }
```

Nota: Función de configuración conexión con el servidor MQTT
Elaborado por: Boris Cutos, Miguel Recalde

En caso de que el módulo NodeMCU ESP8266 no se pueda conectar al primer intento al bróker, se ha creado la función **reconectarMqtt()** la misma que se encargará del proceso de reconexión la misma que lo volverá a intentar cada 5 segundos. En la Figura 33 se muestra el contenido de la función **reconectarMqtt().**

Figura 33. *Código para la reconexión al bróker MQTT*

```
NodoSensor

void reconectarMqtt() {
  /*
  Esta función se encarga de reconectar el sensor al servidor
  si el intento de conexión inicial ha fallado. Este intento
  se realiza cada 5 segundos.
  */
  while (!clientMqtt.connected()) {
    Serial.println("[MQTT]Esperando conexión con MQTT...");
    // Intento para conectar
    if (clientMqtt.connect("ALARMA-CASA-SEGURIDAD")) {
      Serial.println("[MQTT]Conectado");
      } else {
        Serial.print("[MQTT]Fallo, rc=");
        Serial.print(clientMqtt.state());
        Serial.println(" se intentará o travez tras 5 segundos");
        // Esperamos 5 segundos
        delay(5000);
```

Nota: Función encargada de reconectar el sensor al servidor
Elaborado por: Boris Cutos, Miguel Recalde

Cuando se establezca correctamente la comunicación entre el nodo sensor y el bróker MQTT, el módulo NodeMCU ESP8266 debe monitorear constantemente los pines donde se han conectado los interruptores magnéticos y cuando se vulnere uno de estos interruptores se publique un mensaje MTTQ hacia el bróker utilizando el topic asignado. En la Figura 34 se detalla la función **test_state_interruptor()** que se empleará para el monitoreo de los interruptores y publicación de mensajes hacia el bróker.

Figura 34. *Código para el monitoreo de los interruptores magnéticos*

```
NodoSensor §
void test_state_interruptor(const byte puertaVentana, const char* topicPublicar, const char* smsPublicar){
  /*
  Esta función revisa el estado de los interruptores magnéticos y publica mesajes MQTT en el canal
  correspondiente
  Variables de entrada:
  1. const byte puertaVentana: Es el pin donde se conecta el interruptor
  2. const char* topicPublicar: Corresponde al canal o topic donde publicar los mensajes
  3. const char* smsPublicar: Es el mensaje a publicar en el canal
  */
  if (digitalRead(puertaVentana) == LOW) {
    Serial.println("Switch Closed");
  }
  else {
    Serial.println("Switch Open");
    Serial.println(smsPublicar);
    clientMqtt.publish(topicPuerta1, mensajePuerta1);// publicación del mensaje.
    }
```

*Nota: Función **test_state_interruptor()** que se empleará para el monitoreo de los interruptores*
Elaborado por: Boris Cutos, Miguel Recalde

Para verificar si los mensajes enviados por el módulo NodeMCU ESP8266 son recibidos por el bróker MQTT instalado en la Raspberry PI, por medio de una terminal de Raspbian se realiza una suscripción al topic donde el módulo NodeMCU ESP8266 está publicando sus mensajes, de esta manera en el terminal se mostrará los mensajes receptados. Para la suscripción a un topic con el bróker Eclipse Mosquitto, se ingresa el siguiente comando:

mosquitto_sub -h BROKER -t TOPIC.

En donde, **BROKER** corresponde a la dirección IP del bróker MQTT, **TOPIC** es el canal al cual se desea suscribir para recibir los mensajes provenientes del módulo NodeMCU ESP8266. En la Figura 35 se ilustra el contenido de los mensajes enviados por el sensor.

Figura 35. *Subscripción al topic de los sensores para escuchar notificaciones*

```
pi@raspberrypi: ~
File  Edit  Tabs  Help
pi@raspberrypi:~ $ mosquitto_sub -h 192.168.100.90 -t CASA/Cuarto_A/Seguridad/puerta_1
Puerta 1 ha sido vulnerada
Puerta 1 ha sido vulnerada
Puerta 1 ha sido vulnerada
Puerta 1 ha sido vulnerada
Puerta 1 ha sido vulnerada
Puerta 1 ha sido vulnerada
Puerta 1 ha sido vulnerada
```

Nota: Topic de los sensores para escuchar notificaciones
Elaborado por: Boris Cutos, Miguel Recalde

2.4. FUNCIONALIDAD DE NOD-RED DENTRO DEL SISTEMA DOMÓTICO

A través de Nod-RED la Raspberry PI constituye una red que especifica la interacción entre los distintos elementos del sistema domótico de seguridad, utilizando la programación basada en flujos, en donde cada elemento que conforma el sistema se denomina nodo. De esta manera, el

flujo de datos entre los distintos nodos sigue la siguiente lógica: Nod-Red se suscribe a los topics donde el módulo NodeMCU ESP8266 transmite las notificaciones de alerta para luego enviar mensajes de alerta a los Smartphones a través de una aplicación móvil. A continuación, se describe la instalación de Nod-RED en la Raspberry PI y se detalla el flujo de datos que intercomunicará cada uno de los elementos del sistema domótico.

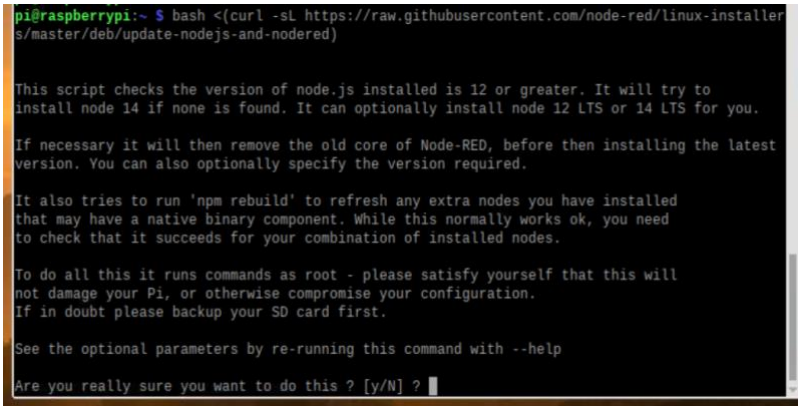
2.4.1. Instalación de Nod-Red

Dado que el software Nod – Red ya se encuentra pre-instalado en el S.O. de Raspbian es necesario actualizarlo insertando el siguiente comando en la terminal:

bash <(curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered)

En la Figura 36 se muestra el proceso de actualización de Nod-RED en Raspbian

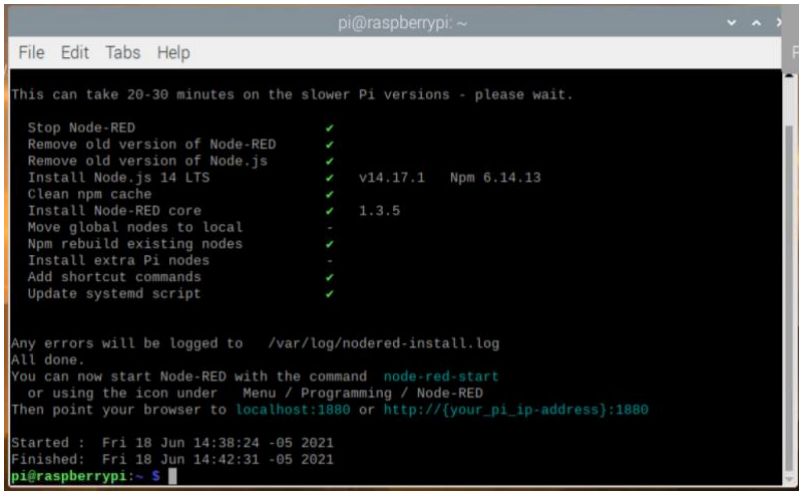
Figura 36. *Actualización del Software Nod-RED*



Nota: actualización de Nod-RED en Raspbian
Elaborado por: Boris Cutos, y Miguel Recalde

En la Figura 37 se ilustra el resultado de la actualización del software Node-Red y de NodeJS

Figura 37. *Proceso de actualización de Node-Red y NodeJS*



Nota: Resultado de la actualización del software Node-Red
Elaborado por: Boris Cutos, y Miguel Recalde

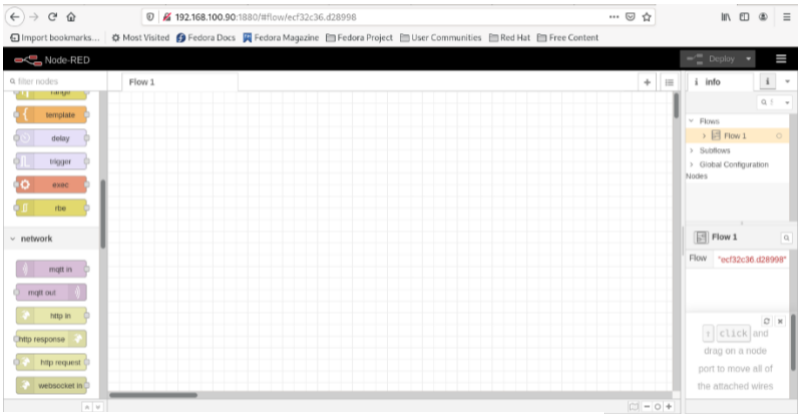
Al finalizar la instalación de Node-Red, se configura para que se inicie automáticamente después de que la Raspberry PI se reinicie. Este se consigue insertando el comando "**sudo systemctl enable nodered.service"** en el terminal de Raspbian como se ilustra en la Figura 38.

Figura 38. *Ejecución de Node-RED al reinicio de Raspberry*

Nota: Configuración para que iniciar automáticamente
Elaborado por: Boris Cutos y Miguel Recalde

Para acceder al software Nod-Red se lo realiza utilizando un navegador web insertando el socket **https://ip_address_raspberry:port** donde i**p_address_raspberry** corresponde a la dirección IP de la Raspberry PI y ***port*** es el puerto 1880 que Nod-Red utiliza por defecto.
En la Figura 39 se muestra la interfaz de Nod-Red al cual se ha accedido por un navegador web con el socket https://192.168.100.90:1880. Se recomienda asignar una dirección IP estática a la interfaz de red de la Raspberry PI.

Figura 39. *Interfaz del software Nod-RED*



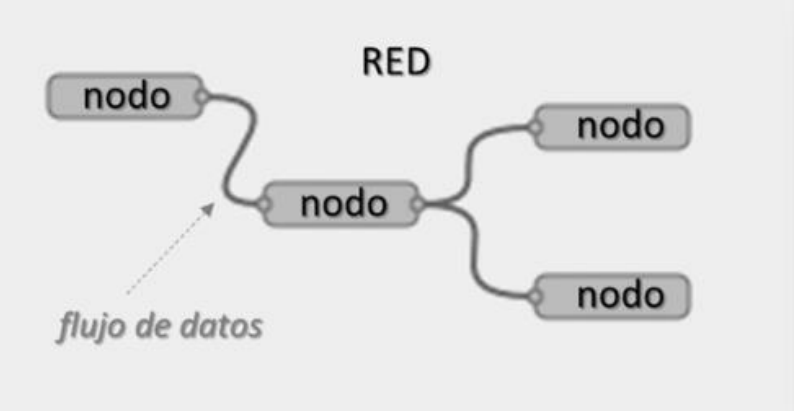
Nota: Interfaz de Nod-Red
Elaborado por: Boris Cutos, Miguel Recalde

2.4.2. Configuración de Nod-RED en el sistema domótico de seguridad

NODE - RED es una herramienta de programación de código libre (Open Source) sobre Node.js desarrollado por el equipo de Emerging Technology Services de IBM (IBM, 2021) y actualmente forma parte de OpenJS Foundation (The Linux Foundation, 2021).

NODE-RED se basa en herramientas de programación basadas en flujos ("flow-based programming (FBP) tools"), lo que permite conectar distintos dispositivos de hardware, API (Interfaz de Programación de Aplicaciones) y servicios en línea, ya que el comportamiento de cualquier aplicación se considera como una red de múltiples nodos, donde cada nodo se compone de codificación específica para un comportamiento único como: recibir, capturar o procesar datos, posteriormente enviarlos hacia uno o varios nodos configurados dentro del sistema de la aplicación, estableciéndose una red donde existe flujo de datos entre nodos (Nod-RED, 2021).

Figura 40. *Elementos de una red utilizando la programación basada en flujos*



Nota: flujos permite representar redes de procesos que intercambian datos
Fuente: (Sancho, 2020)

Como se ilustra en la Figura 40, la programación basa en flujos permite representar redes de procesos que intercambian datos a través de conexiones predefinidas, es decir, cada proceso es representado por nodos que conectados mediante los puertos configurados en cada uno de ellos, para establecer el un canal de datos continuo entre nodos. Cada nodo de la red debe confugurado un solo puerto de entrada y puede tener mas de un puerto de salida.

Nod-RED trabaja con tres tipos de nodos que son:

Nodos iniciadores de flujo: Supervisan de forma periódica el lanzamiento de nuevos eventos como: rangos de tiempo y la entrada de un archivo al hold de directorios, la entrada de un nuevo flujo de MQTT subscrito, etc. Además, estos nodos están atentos de una petición, de la misma manera que los API REST o Websockets (Sancho, 2020).

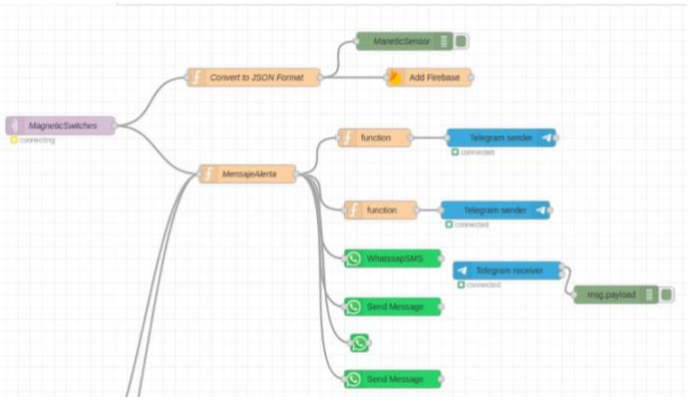
Nodos intermedios: Procesa la información recibida del flujo de datos, ejecutan acciones sobre la mencionada información y envían los resultados a los demás nodos dentro del circuito a través de los puertos configurados para la salida de los datos. Entre las acciones que ejecutan estos nodos se puede mencionar: El formateo de datos (JSON, YAML, CSV, XML), consumo de APIs, persistencia en RAM, bases de datos (SQL/NoSQL), descarte de mensajes mediante condiciones establecidas para el procesamiento de la información, etc. (Sancho, 2020).

Nodos terminales: Estos nodos se ubican al final de los flujos de datos, disparando eventos cuando el flujo de la trama los alcanza, por lo que estos nodos solo tienen un puerto de entrada. Las acciones que ejecutan estos nodos se encuentran: invocación de otros servicios expuestos externamente, publicación y seguimiento de los datos, generación de eventos de notificación en sistemas de mensajería social (Twitter, Telegram), etc. (Sancho, 2020).

En la Figura 41 se tiene un flujo de nodos que ha sido trazado en la interfaz de Nod-RED para mostrar los mensajes enviados por los sensores instalados en puertas y ventanas. Los mensajes de cada uno de los sensores son publicados en canales MQTT diferentes, es necesario en el

NodeRED establecer nodos iniciadores de flujo que se suscriban a dichos canales para poder receptar los mensajes y darle un posterior tratamiento de formato en el cual sea aceptado en la base de datos alojada en la plataforma de Firebase.

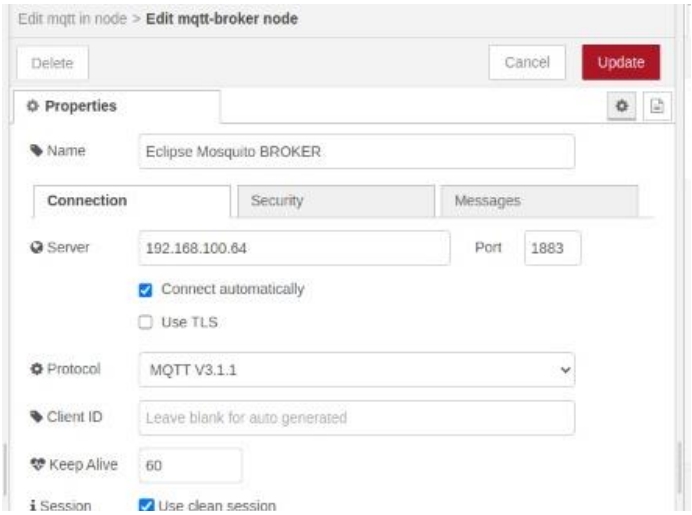
Figura 41. *Configuración de nodos para recibir las notificaciones MQTT enviados por los sensores.*



Nota: flujo de nodos que ha sido trazado en la interfaz de Nod-RED
Elaborado por: Boris Cutos, Miguel Recalde

A través de los parámetros del nodo de inicio de flujo MQTT, es posible conectarse a un canal MQTT en específico. En la Figura 42 se muestra que para configurar este nodo es necesario especificar la dirección IP donde se aloja el bróker MQTT, el PATH del canal MQTT y además se debe ingresar el usuario y contraseña en la sección de seguridad para la correcta autenticación del nodo con el bróker MQTT.

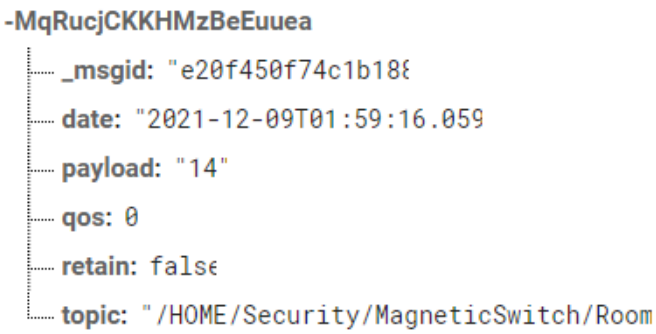
Figura 42. *Parámetros de configuración en el nodo inicio de flujo MQTT*



Nota: Configurar este nodo es necesario especificar la dirección IP donde se aloja el bróker MQTT
Elaborado por: Boris Cutos, Miguel Recalde

El mensaje enviado desde los sensores se transforma en un formato JSON con la información proveniente del header y payload del paquete MQTT enviado, como se ilustra en la Figura 43.

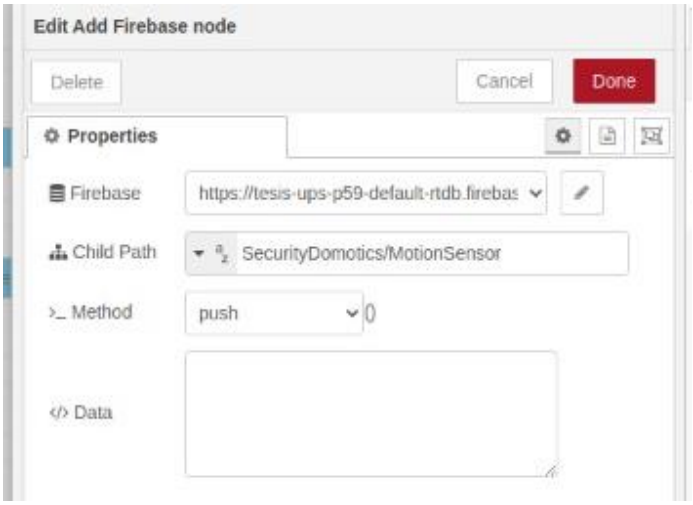
Figura 43. *Información del header y payload de los mensajes MQTT*



Nota: Mensaje enviado desde los sensores se transforma en un formato JSON
Elaborado por: Boris Cutos, Miguel Recalde

Para el envío del mensaje de notificación a la base de datos de tiempo real de Firebase, se utiliza un nodo terminal personalizado, el mismo que como parámetros de configuración es la especificación de la URL que apunta a la base de datos y la operación CRUD HTTP a ejecutar. En la Figura 44 se indica las propiedades del nodo de Firebase personalizado en la plataforma de NOD-RED.

Figura 44. *Parámetros de configuración nodo intermedio para conexión a Firebase*



Nota: Propiedades del nodo de Firebase
Elaborado por: Boris Cutos, Miguel Recalde

CAPÍTULO III

3. DISEÑO DE LA APLICACIÓN MÓVIL MUTIPLATAFORMA

Todo el código fuente detallado se encuentra disponible en el Bitbucket asignado para la Universidad Politécnica Salesiana

3.1. CONFIGURACIÓN DEL ENTORNO DE DESARROLLO FLUTTER

Para realizar las configuraciones necesarias del entorno de desarrollo en el cual se va a levantar la aplicación de control domótico, se tiene que considerar en que Sistema Operativo se va a trabajar. Esto según las consideraciones de la documentación ofertada por Flutter (Flutter, 2021)

3.1.1. Instalación de Flutter en Mac OS

En primer lugar, se tiene que de descargar el comprimido que se encuentra en la página oficial de Flutter mismo que posee los Releases actualizados (Flutter, 2021).

Una vez descargado el comprimido con la metadata necesaria para arrancar el entorno de desarrollo, se deberá ejecutar los siguientes comandos en el directorio en el cual se descargó el archivo.

```
cd ~/development
unzip ~/Downloads/flutter_macos_2.5.3-stable.zip
```

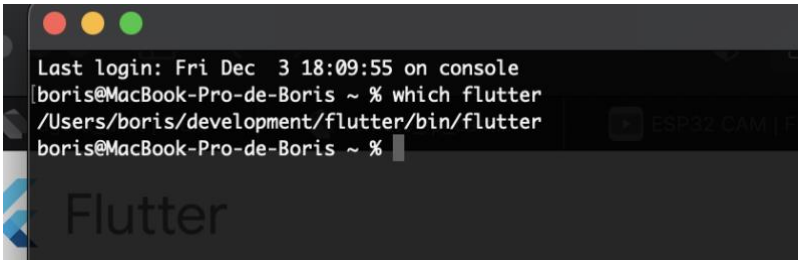
A continuación, se tiene que añadir los *tools* de Flutter al PATH del sistema

```
$ export PATH="$PATH:`pwd`/flutter/bin"
```

Este último comando añade al PATH del sistema las configuraciones de Flutter para que este pueda arrancar todas sus herramientas de entorno en el Sistema Operativo.
Una vez realizado las configuraciones anteriormente mencionadas, se puede realizar una búsqueda de donde se encuentra los Binarios de Flutter mediante el comando, ejecutado desde la terminal del Equipo.

which flutter

Figura 45. *Comando que muestra el directorio de instalación de Flutter*



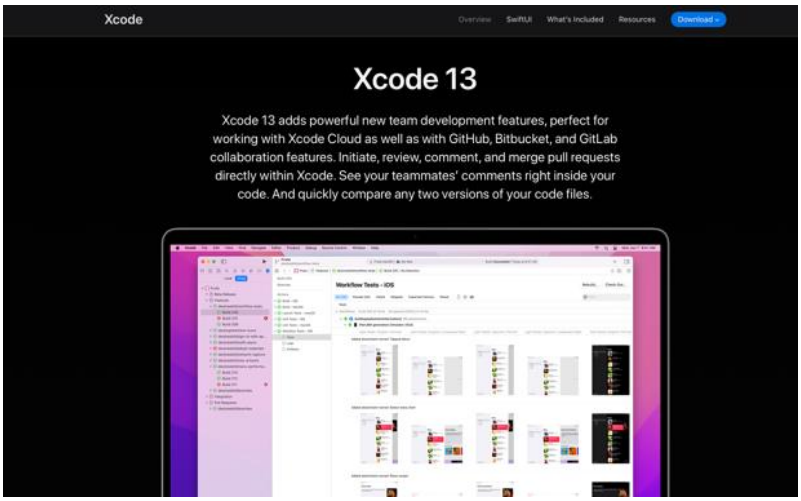
Nota: Directorio de instalación de Flutter
Elaborado por: Boris Cutos y Miguel Recalde

3.1.2. Configuración del entorno de desarrollo IOS

Flutter al permitir a los desarrolladores trabajar en Nativo mediante los conceptos de SDKs de cada una de las plataformas de desarrollo de aplicaciones móviles, el mismo código que funciona para IOS, funcionara para Android, es decir, el Archive (App) generado tras la compilación funcionara en todos los dispositivos IOS o ANDROID del mercado.

Lo primero que se debe de realizar es instalar Xcode, el cual contiene el entorno de desarrollo exclusivo de IOS. Para poder descargar la última versión disponible, se debe de acceder a la Mac App Store y descargar el dmg. El enlace a continuación.

figura 46. *Pantalla de descarga de Xcode IOS*



Nota: Entorno de desarrollo Xcode 13
Elaborado por: Boris Cutos y Miguel Recalde, a través de
https://apps.apple.com/us/app/xcode/id497799835

A continuación, lo que se realizará es configurar Xcode desde la terminal integrada del equipo en el que se está instalando. Ejecutar los siguientes comandos

```
$ sudo xcode-select --switch /Applications/Xcode.app/Contents/Developer
$ sudo xcodebuild -runFirstLaunch
```

Seguido de la anterior acción, se deberá aceptar las políticas de Apple para instalar su software (Xcode) ejecutando el siguiente comando en la terminal de la computadora.

```
$ sudo xcodebuild -license
```

3.1.3. Configurar un simulador IOS

Abrir una terminal y ejecutar el siguiente comando

```
open -a Simulator
```

Se debe tener en cuenta las siguientes consideraciones.

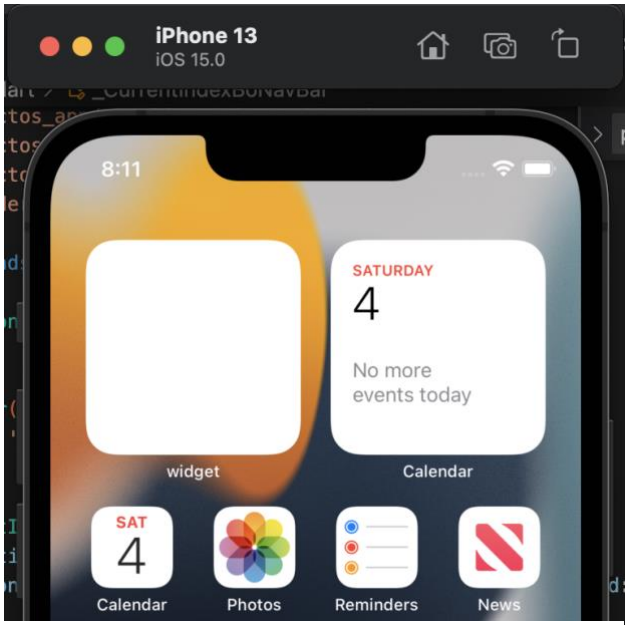
Tabla 9. *Consideraciones del computador para los simuladores*

Arquitectura	Requerimientos del dispositivo	Observaciones
64 bits	Iphone5 en adelante	Tener en consideración la capacidad y resolución de la máquina en la que se va a utilizar el simulador (pixeles) adicional las pulgadas de monitor para optimizar recursos de la computadora.

Nota. Se muestra los requerimientos mínimos y observaciones a ser tomados en consideración para levantar un simulador IOS
Elaborador por: Boris Cutos y Miguel Recalde

Para la aplicación que servirá de administrador del sistema domótico se seleccionó una simulación de un IPhone 13 tomando como referencia la Tabla 9. Consideraciones del computador para los simuladores

Figura 47. *Simulador IOS*



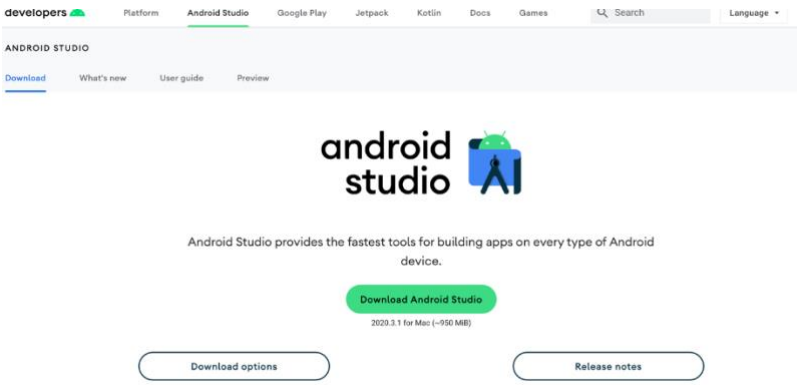
Nota: Simulación de un IPhone 13
Elaborado por: Boris Cutos y Miguel Recalde

3.1.4. Configurar un simulador Android

Para la configuración se deberá descargar Android Studio, desde el sitio oficial de Android y dependiendo del Sistema Operativo, se deberá realizar la descarga el archivo compatible. (Android, 2021).

Las configuraciones de un dispositivo virtual Android tienden a ser un poco más elaboradas que en IOS, por lo cual se explicará más detalladamente la configuración del simulador.

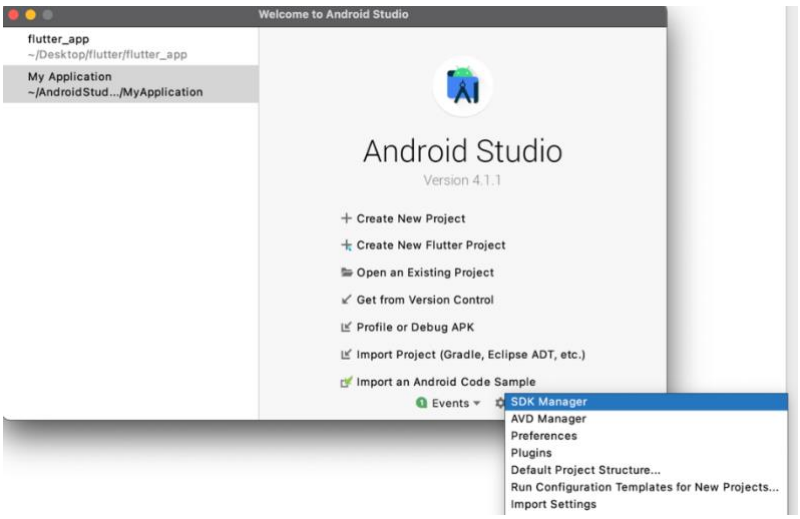
Figura 48. *Pantalla de descarga de Android Studio*



Nota: Descarga de Herramienta Android Studio
Elaborador por: Boris Cutos y Miguel Recalde,
mediante la URI de descarga de Android Studio

Una vez descargado el IDE de desarrollo con todos sus componentes, se procederá a iniciar el programa de Android Studio y se procederá a instalar la última versión de Android SDK mediante el Wizard de configuración que el mismo programa oferta.

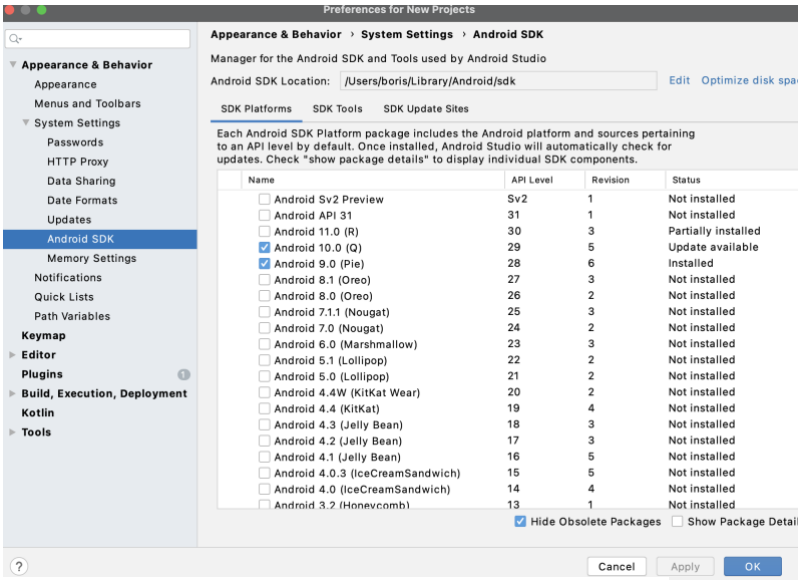
Figura 49. *Wizard de Android para añadir dependencias*



Nota. Se muestra el menú para la configuración de SDK Android mediante el Wizard incorporado en el IDE
Elaborado por: Boris Cutos y Miguel Recalde

Una vez dentro del Wizard, se procederá a seleccionar la versión de Android, y se procederá la instalación automáticamente después de presionar "OK" tal cual se muestra en la Figura 50. Versiones del Android SDK disponibles mediante el Wizard

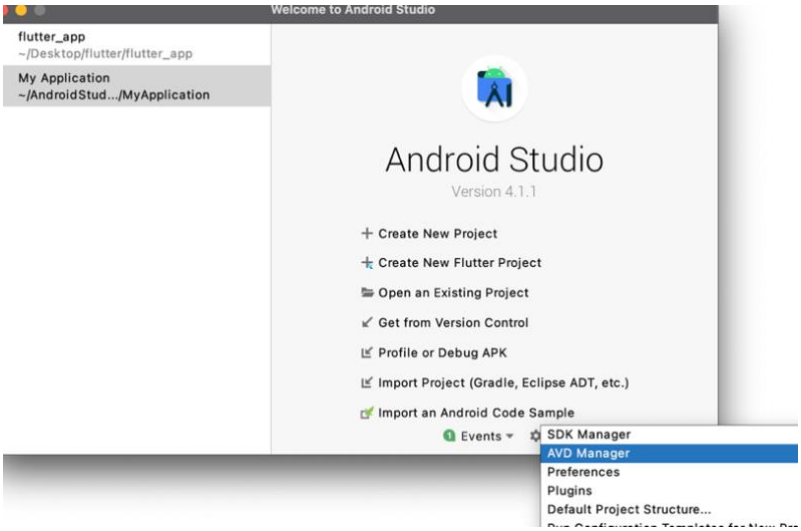
Figura 50. *Versiones del Android SDK disponibles mediante el Wizard*



Nota: Para el desarrollo de la aplicación administradora del sistema domótico, se tomará en consideración Android 10.
Elaborado por: Boris Cutos y Miguel Recalde

Una vez instalada la versión de Android SDK pertinentes para el desarrollo de la aplicación, se procederá a la configuración del simulador mediante el Wizard de Android Studio.

Figura 51. *Wizard para la configuración del simulador Android*



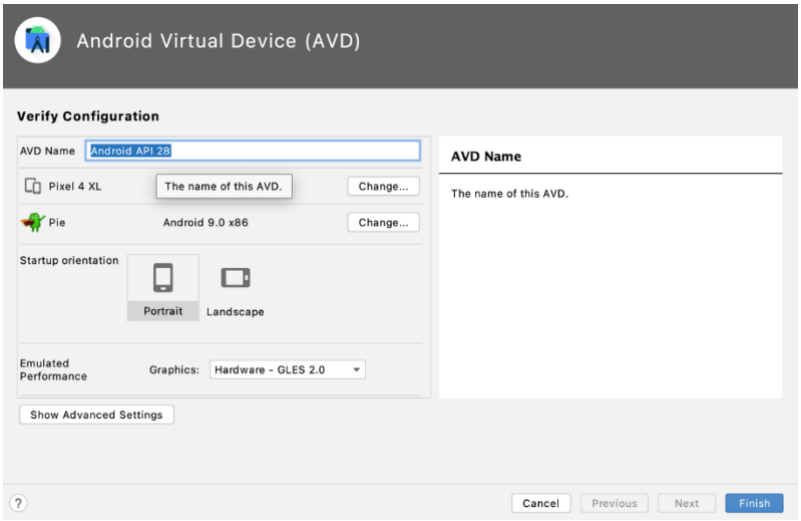
Nota. Se muestra el Wizard para la configuración de un nuevo simulador Android
Elaborado por: Boris Cutos y Miguel Recalde.

Una vez dentro del Wizard

Figura *49. Wizard de Android para añadir dependencias*

Figura 49se procede a seleccionar el nombre del dispositivo, además de las configuraciones básicas que serán de ser tomadas en consideración. En la figura 53 hay un ejemplo de las configuraciones.

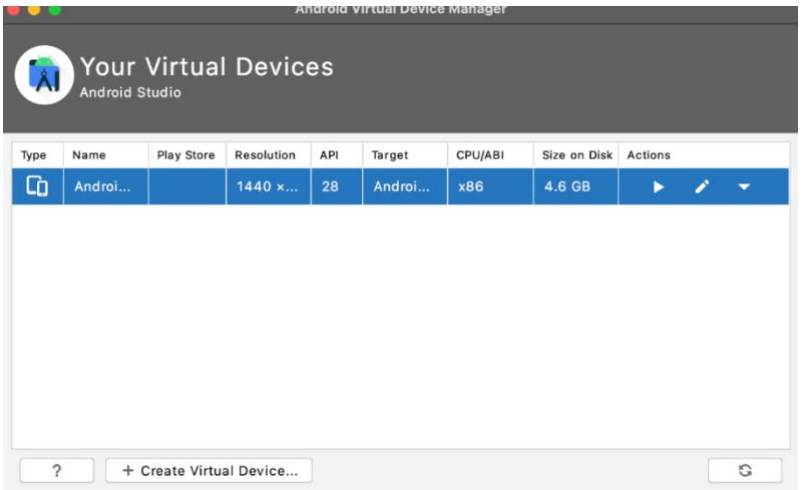
Figura 52. *Configuraciones de un simulador Android mediante Wizard*



Nota: Configuración inicial para Android Studio
Elaborado por: Los Autores*, mediante una captura del Wizard de Android Studio*

Una vez realizadas las configuraciones detalladas en los pasos anteriores, se podrá visualizar el dispositivo para ser utilizado.

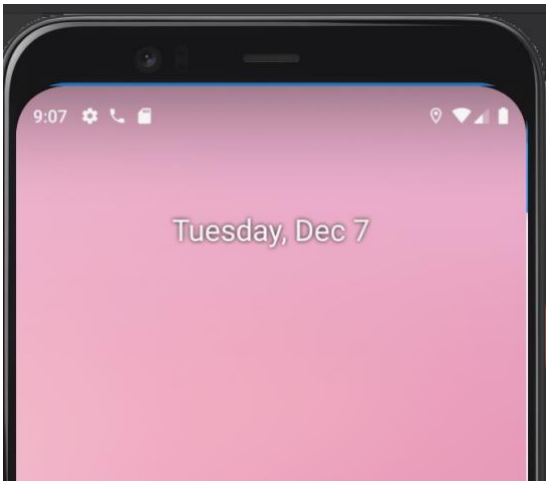
Figura 53 *Simuladores Android disponibles*



Nota: Interface simulador Android
Elaborado por: Boris Cutos, Miguel Recalde.

En la figura 53 se puede revisar el arranque del simulador.

Figura 54. *Simulador Android*



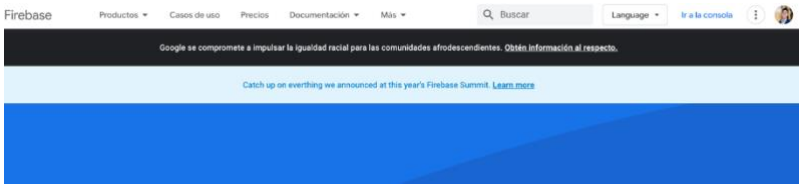
Nota: Simulador de sistema Android
Elaborado por: Boris Cutos, Miguel Recalde

3.2. FIREBASE

3.2.1. Creación del proyecto de FIREBASE

Para la configuración de Firebase, basta con crear una cuenta de Google, misma que se deberá ser asociada con alguno de los desarrolladores a cargo del proyecto y acceder a la consola de administración.

Figura 55. *Consola de Administración Firebase*



Nota: Creación de la consola de administración Firebase
Elaborado por: Boris Cutos, Miguel Recalde *Mediante la captura de imagen de Google Firebase*

Una vez dentro de la consola de Firebase se necesita seleccionar un nuevo proyecto en el cual se va a llevar el control de la información que va a ser almacenada en las bases de datos

Figura 56. *Agregar un Nuevo proyecto Firebase*

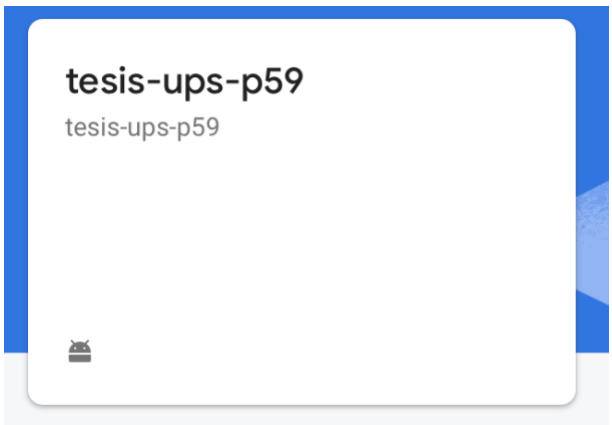
Nota: Consola Firebase – Nuevo Proyecto
Elaborado por: Boris Cutos y Miguel Recalde, *mediante una foto captura de la consola de Firebase*

Una dentro de la creación de un nuevo proyecto de Firebase (**Figura 56.** Agregar un Nuevo proyecto Firebase) se deberá llenar la información que solicita el nuevo proyecto como lo son

- Nombre del proyecto.
- Seleccionar si se desea o no un control de Google Analytics.
- Seleccionar la cuenta de Google por default y proceder a la creación del Proyecto.

Una vez creado el proyecto, se podrá tenerlo disponible dentro de la consola de Firebase

Figura 57. *Proyecto de Tesis en la plataforma Firebase*

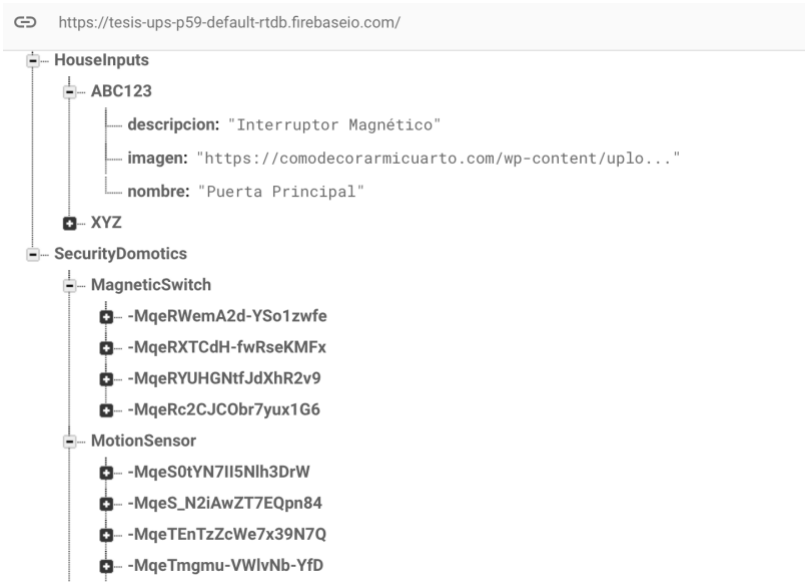


Nota. Se muestra un nuevo proyecto creado en Firebase con las configuraciones necesarias para crear bases de datos, funciones, y servicios de notificaciones.
Elaborado por: Boris Cutos y Miguel Recalde, *mediante la foto captura de la Consola de Administración Firebase*

Una vez dentro de la aplicación creada, procederemos a crear los nodos correspondientes que serán los encargados de almacenar la información de los accesos a los domicilios y de las notificaciones recibidas desde NODE Red como se muestra en la figura 58.

Se muestra los nodos "HouseInputs" que almacenará los accesos a los domicilios, "SecurityDomotics" es el nodo que almacenará la información de las incidencias que vienen desde Node Red

Figura 58. *Nodos de almacenamiento del Bróker Node Red*



Nota. Se almacena la información de las incidencias
Elaborado por: Boris Cutos y Miguel Recalde, mediante la creación de los nodos de Base de Datos para almacenamiento de información en Firebase.

3.3. DISEÑO DE LA APLICACIÓN MOVIL

3.3.1. Módulo de Login

El módulo de ***Login*** está programado de tal forma que el algoritmo se encarga de validar que dichos campos sean considerados de acuerdo con formatos válidos. Está diseñado con un formulario que valida que la contraseña sea únicamente números y que el ***user*** sea únicamente correos electrónicos figura 58, todo esto con el manejo de sentencias propias de los ***TextFields*** que el framework posee como el manejo de campos específicos como el correo electrónico y de otros valores que tienen que ser validados de manera más específica como lo son las expresiones regulares para el Field de las contraseñas

Figura 59. *Código para el análisis de los campos obligatorios en el Login*

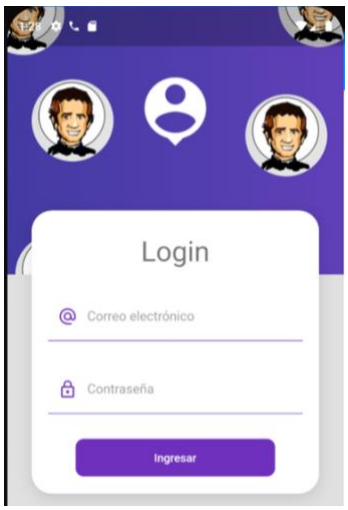
```
String pattern = r'^(([^<>()[\]\\.,;:\s@\"]+(\.[^<>()[\]\\.,;:\s@\"]+)*)|(\".+\"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$';
RegExp regExp = new RegExp(pattern);

return regExp.hasMatch(value ?? '')
? null
: 'El valor ingresado no luce como un correo';
```

Nota: Creación de código funcional para la validación de los formularios de Ingreso del Login
Elaborado por: Boris Cutos y Miguel Recalde,

La pantalla de Login es un Frame de componentes variados en los que se incluyen “Labels”, “Layouts”, “Fields”, etc. Cada componente tiene una codificación exclusiva para su funcionamiento

Figura 60. *Pantalla de Login de la aplicación del Sistema Domótico*



Nota: Ingreso al sistema Domótico
Elaborado por: Boris Cutos y Miguel Recalde, mediante la codificación de la funcionalidad de Login con el lenguaje de programación Dart

3.3.2. Módulo de notificaciones PUSH desde Firebase

El módulo de notificaciones PUSH consta de la librería "Firebase 9.0.2" que deberá de ser importada desde el repositorio virtual de FLUTTER (pub.dev, 2021) mismo que contiene las dependencias correspondientes para conectarse a las REALTIME Database y recibir las notificaciones cada cuanto se dispare una advertencia desde el módulo de NODE RED (cada que haya una inserción). La codificación esta realizada en lenguaje Dart, mismo que realiza una apertura de un escucha para los tres estados que posee una aplicación móvil, la cuales son las siguientes:

- Activa
- En segundo Plano
- Modo OFF

Los estados anteriormente mencionados, vienen definidas por la documentación de Flutter (flutter.dev, 2021). En los tres casos mencionados, se recibirá una notificación alertando al usuario de que existe una incidencia en su domicilio cuando alguno de los sensores haya detectado algún problema (siempre y cuando la aplicación este encendida).

Una vez que se atienda la notificación entrante de que existe algún problema, esta mostrará una pantalla en la que se muestra una alerta y con la que el usuario pueda interactuar y que lo direccionará a la consola principal en la que el usuario podrá tomar alguna acción en concreto

Figura 61. *Pantalla de Alerta*

Nota: Pantalla de alerta sistema domótico
Elaborado por: Boris Cutos y Miguel Recalde.

En la consola principal se podrá observar cuál de los accesos del domicilio ha sido vulnerado ya que en la parte superior izquierda del recuadro que corresponde al acceso del domicilio se mostrará un mensaje parpadeante que indica que ese acceso ha sido vulnerado.

Figura 62. *Consola Principal del sistema Domótico en la Aplicación Móvil*

Nota: Indicadores de accesos vulnerados
Elaborado por: Boris Cutos y Miguel Recalde*, mediante la creación de la pantalla en la Aplicación con Dart y Flutter*

La pantalla principal o consola de administración del sistema domótico mencionada en la Figura 62, se puede visualizar que existe en su parte inferior existe un menú en el que figuran tres opciones de selección las cuales son:

- Teléfono
- Home
- Activar / Desactivar

Cada una de las opciones mencionadas tienen un comportamiento particular con el cual se puede interactuar con las opciones disponibles en la aplicación como las mencionadas en el alcance del proyecto, como lo son las llamas de emergencia, verificar el estado de la aplicación. El diseño del menú viene dado por Widgets predefinidos en el Framework Flutter, cuyo nombre es “BottomNavigationBar”. Este provee un entorno con funciones claras y predefinidas que permite la carga de componentes de acuerdo con la necesidad del usuario, a continuación, se muestra el código de cada uno de los componentes.

Figura 63. *Código del menú inferior de la consola de administración*

```
BottomNavigationBar(
    items: [
      BottomNavigationBarItem(
        icon: GestureDetector(
            onTap: () async {
              const number = '911'; //set the number here
              await FlutterPhoneDirectCaller.callNumber(number);
            },
            child: Icon(Icons.mobile_friendly)),
        label: 'Teléfono',
      ),
      BottomNavigationBarItem(icon: Icon(Icons.house_rounded), label: 'Home'),
      BottomNavigationBarItem(
          icon: Icon(Icons.power_settings_new_outlined),
          label: 'Activar / Desactivar'),
    ],
    selectedItemColor: Colors.amber,
    currentIndex: this._selectedIndex,
    onTap: (value){
      print ('veamos que sale $value');
      uiprovider.selectedItem = value;
    },
  );
```

Nota. El código muestra todos los componentes de los cuales se basa el menú inferior de la consola principal.
Elaborado por: Boris Cutos y Miguel Recalde, mediante el framework Flutter

La programación de la **Figura 63.** Código del menú inferior de la consola de administración, muestra en sí mismo subcomponentes que se explicarán a detalle su funcionamiento más adelante, sin embargo, se abordarán a breves rasgos en esta sección. "BottomNavigationBarItem" conforma una programación simple de entender, ya que en estos se definen los ***items*** a mostrar en el menú inferior. Adicional se puede definir en ellos mismos comportamientos específicos de cada componente mediante funciones ***onTap()*** que se comportarán de acuerdo a la programación que se encuentren especificadas en ellas (pub.dev, 2021).

3.3.3. Funcionalidad de llamadas telefónicas desde la aplicación móvil

Para la funcionalidad de la utilización de las llamadas telefónicas se realizó la importación del paquete "flutter_phone_direct_caller" (pub.dev, 2021), mismo que posee todas las características para utilizar las herramientas del teléfono anfitrión, entre ellas la app de telefonía integrada.

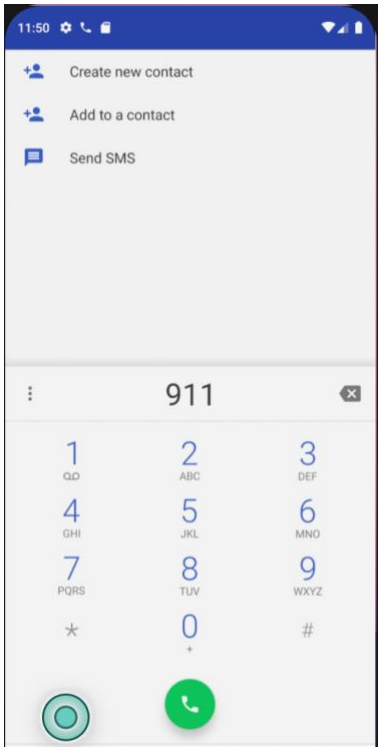
Figura 64. *Ítem Telefónico del menú inferior*

```
BottomNavigationBarItem(
        icon: GestureDetector(
            onTap: () async {
              const number = '911'; //set the number here
              await FlutterPhoneDirectCaller.callNumber(number);
            },
            child: Icon(Icons.mobile_friendly)),
        label: 'Teléfono',
      ),
```

Nota. Se describe la funcionalidad del código para el Ítem Teléfono
Elaborado por: Boris Cutos y Miguel Recalde

La librería "flutter_phone_direct_caller" permite definir entre en sus propiedades el número al cual se va a designar por defecto para marcar cada cuando se necesite realizar una llamada telefónica mismo que es el "911" número internacional de emergencias. Se puede verificar que en las líneas de código se **setea** el número de emergencias dentro de la función de comportamiento del Widget, ***onTap()*** de la Figura 64. Ítem Telefónico del menú inferior. Adicional, se define un **await()** para que se dispare una espera hasta que la funcionalidad del teléfono se despliegue

Figura 65. *Aplicación de llamadas del Teléfono Host*



Nota. Se invoca el aplicativo de llamadas desde la aplicación móvil de control domótico.
Elaborado por: Boris Cutos y Miguel Recalde

3.3.4. Módulo de notificaciones en la aplicación domótica

Las notificaciones se definen mediante un módulo desarrollado en lenguaje Dart, sin intervención de las funcionalidades que nos brinda el framework Flutter. Lo que se pretende es crear una funcionalidad que permita estar en modo escucha de los eventos que se encuentren dentro de la Real Time Database de Flutter, para ser más específicos, de las inserciones que se realicen en los nodos, ya que estos son los que almacenarán la información pertinente a los eventos que se generen de parte de los sensores manejados por Node Red. El almacenamiento de los eventos en los nodos de la Real Time Database de Firebase, se lo explica en la sección **3.2.0** ***Creación del proyecto de Firebase.***

El código de Dart que conforma las funciones de escuchas para los estados de la aplicación mencionados en la sección **3.3.2** *Módulo de notificaciones PUSH desde Firebase,* se los detalla a continuación.

Figura 66. *Código de la clase que maneja las funciones de escucha de notificaciones PUSH*

```
class PushNotificationsServices {
  static FirebaseMessaging messaging = FirebaseMessaging.instance;
  static String? token;
  static StreamController<String> _messageStream =
      new StreamController.broadcast();

  static Stream<String> get messagesStream => _messageStream.stream;

  static Future _onBackgroudHandler(RemoteMessage message) async {
    print('backgorund handler ${message.messageId}');
    print(message.data);
    _messageStream.add(message.data['product'] ?? 'no data');
  }

  static Future _onMessageHandler(RemoteMessage message) async {
    print('_onMessageHandler handler ${message.messageId}');
    print(message.data);
    _messageStream.add(message.data['product'] ?? 'no data');
  }

  static Future _onMessageOpenAppHandler(RemoteMessage message) async {
    print('_onMessageOpenAppHandler handler ${message.messageId}');
    print(message.data);
    _messageStream.add(message.data['product'] ?? 'no data');
  }

  static Future initializeApp() async {

    await Firebase.initializeApp();
```

```
    token = await FirebaseMessaging.instance.getToken();
    print("el token es $token");
    FirebaseMessaging.onBackgroundMessage(_onBackgroudHandler);
    FirebaseMessaging.onMessage.listen(_onMessageHandler);
    FirebaseMessaging.onMessageOpenedApp.listen(_onMessageOpenAppHandler);
  }

  static closeStreams() {
    _messageStream.close();
  }
}
```

Nota: Clase que maneja las funciones de escucha
Elaborado por: Boris Cutos y Miguel Recalde, mediante el lenguaje de programación Dart

En la clase expuesta en la se puede evidenciar las funciones que se harán cargo de la escucha de las notificaciones que vendrán desde Firebase.

Para el correcto funcionamiento de la clase expuesta, se tiene que importar la dependencia "firebase_messaging" y "firebase_core" (pub.dev, 2021)

Figura 67. *Dependencias en Flutter*

```
dependencies:
  cupertino_icons: ^1.0.2
  firebase_core: ^1.10.0
  firebase_messaging: ^11.1.0
  flutter:
    sdk: flutter
  flutter_phone_direct_caller: ^2.1.0
  http: ^0.13.4
  provider: ^5.0.0
```

Nota: Dependencias de Flutter
Elaborado por: Boris Cutos y Miguel Recalde, mediante la importación de dependencias desde pub.dev (pub.dev, 2021)

3.3.5. Función _onMessageOpenAppHandler()

La función "*_onMessageOpenAppHandler"* se encuentra escuchando las notificaciones siempre y cuando la aplicación se encuentre abierta, es decir, cuando el usuario se encuentre utilizándola.

Los métodos vienen dados por la propia dependencia importada desde *"firebase_messaging"*. Lo único necesario es definirlo dentro de la función, y ya el propio *core* de la dependencia, se encargará de procesar las peticiones que vengan desde Firebase

3.3.6. Función _onMessageHandler()

Esta función permite que la aplicación se encuentre escuchando las notificaciones cuando la aplicación se encuentre en segundo plano, es decir la aplicación no se encuentra siendo ocupada por el usuario.

3.3.7. Función _onBackgroudHandler()

Esta función permite que la aplicación se encuentre escuchando las notificaciones cuando la aplicación se encuentre cerrada, es decir la aplicación no se encuentra siendo ocupada por el usuario y este tampoco la ha abierto.

CAPITULO IV

4. PRUEBAS Y RESULTADOS

Las pruebas se realizaron sobre tres beneficiarios, mismos que fueron dos familias en el sector de Chillogallo, cumpliendo así el alcance definido al levantar el presente proyecto. Las instalaciones beneficiadas fueron dos domicilios del sector de Chillogallo.

Las instalaciones del sistema de Seguridad y Monitorización Domótico se tomaron en cuenta en base a los accesos críticos que poseen los domicilios, siendo estos en los que se instalaron dos interruptores magnéticos, un sensor de golpe y un sensor de movimiento, todo esto monitoreado desde la aplicación móvil multiplataforma.

Para el muestreo de resultados se tomó en consideración pruebas durante tres días. En los días de pruebas se testearon los sensores, que estos estén llegando con las notificaciones pertinentes a la aplicación móvil y que la información se almacene en Firebase para llevar un registro de los eventos disparados en el sistema.
Las métricas para el análisis de los días de prueba fueron

- Estado de los dispositivos
- Tiempo de respuesta entre comunicación del sistema con NodeRED y Firebase
- Tiempo de comunicación entre Firebase y la aplicación Móvil

Las pruebas se las realizaron utilizando Google Analytics. Este es una plataforma ofertada por Google que permite el análisis de todos los componentes que se anclen con este, y está orientado a trabajar particularmente bien con Firebase.

4.4. PRUEBAS SISTEMA DOMÓTICO EN EL DOMICILIO FAMILIA VIZCAÍNO.

Para la Familia Vizcaíno, se instaló el sistema en puntos de comodidad para los propietarios, y con la sugerencia de las mejores prácticas para la instalación de dispositivos electrónicos. Los Accesos fueron:

- Sensores Magnéticos
    - La puerta principal del domicilio
    - La puerta posterior del domicilio.
- Sensor de Golpe
    - La ventana lateral Junto al Acceso principal
- Sensor de movimiento.

4.4.1. Sensores Magnéticos

Figura 68. *Acceso principal del domicilio*

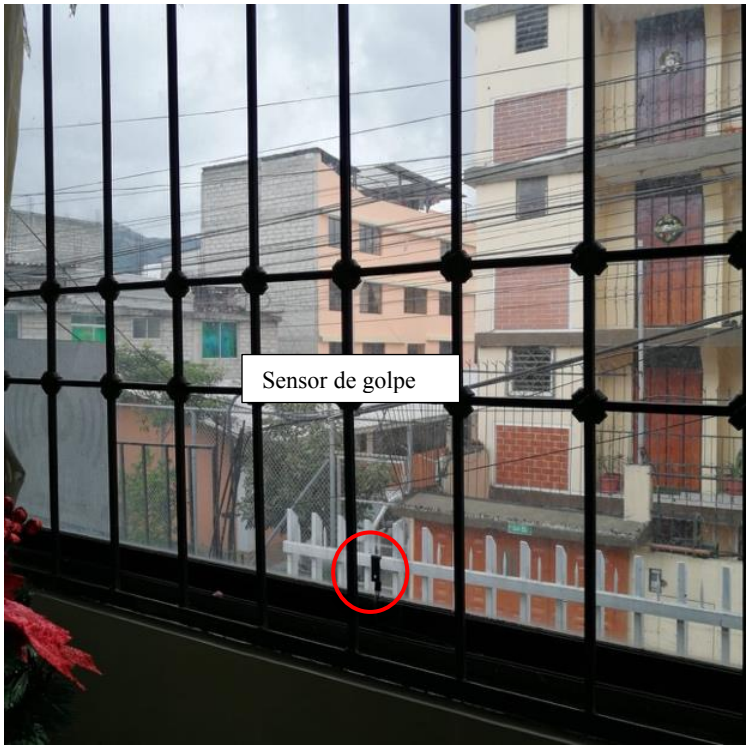
Elaborado por: Boris Cutos y Miguel Recalde

Figura 69. *Acceso posterior del domicilio*

Elaborado por: Boris Cutos y Miguel Recalde

4.4.2. Sensor de Golpe

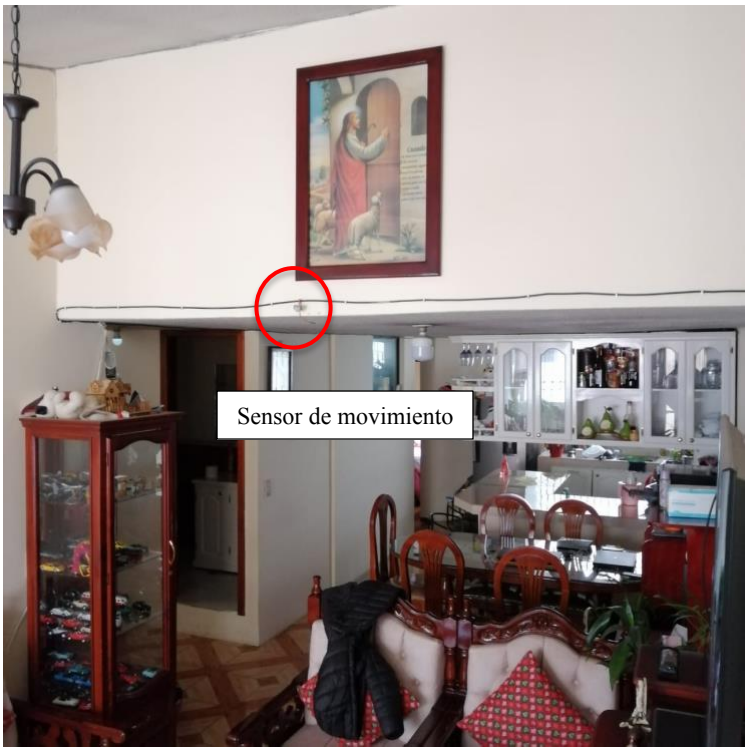
Figura 70. *Ventana Posterior al acceso principal del domicilio*



Elaborado por: Boris Cutos y Miguel Recalde

4.4.3. Sensor de Movimiento

Figura 71. *Sala de estar del domicilio*



Elaborado por: Boris Cutos y Miguel Recalde

4.4.4. Resultado de las pruebas

Las métricas para el análisis de los días de prueba fueron

- Estado de los dispositivos
- Tiempo de respuesta entre comunicación del sistema con NodeRED y Firebase
- Tiempo de comunicación entre Firebase y la aplicación Móvil

Tabla 10. *Resultados del domicilio de la familia Vizcaíno*

Dispositivos	Accesos del domicilio controlados	Días de prueba (noviembre 2021)								
		14-nov			15-nov			16-nov		
		Funcionamiento del sensor	Tiempo de comunicación con Firebase (seg)	Tiempo de respuesta con la aplicación Móvil (seg)	Funcionamiento del sensor	Tiempo de comunicación con Firebase (seg)	Tiempo de respuesta con la aplicación Móvil (seg)	Funcionamiento del sensor	Tiempo de comunicación con Firebase (seg)	Tiempo de respuesta con la aplicación Móvil (seg)
Sensores magnéticos	Puerta principal	ok	1.326	0.943	ok	0.837	1.519	ok	0.838	0.97
	Puerta Posterior	ok	1.576	0.926	ok	0.711	0.702	ok	0.926	0.75
Sensor de Golpe	Ventana Lateral (Junto a la puerta principal)	ok	0.937	0.945	ok	0.901	0.970	ok	0.957	0.798
Sensor de Movimiento	Sala de estar	ok	2.489	0.951	ok	1.651	0.838	ok	1.598	2.681

Elaborado por: Boris Cutos y Miguel Recalde, mediante la toma de resultados de Google Analytics

4.5. PRUEBAS SISTEMA DOMÓTICO EN EL DOMICILIO FAMILIA PLAZARTE

Para la Familia Plazarte, se instaló el sistema en puntos de comodidad para los propietarios, y con la sugerencia de las mejores prácticas para la instalación de dispositivos electrónicos. Se tomó en consideración de que la familia Plazarte posee un taller de confecciones en su domicilio por lo cual no poseen una sala de estar, pero en esta se tomó como punto principal para el sensor de movimiento. Los Accesos fueron:

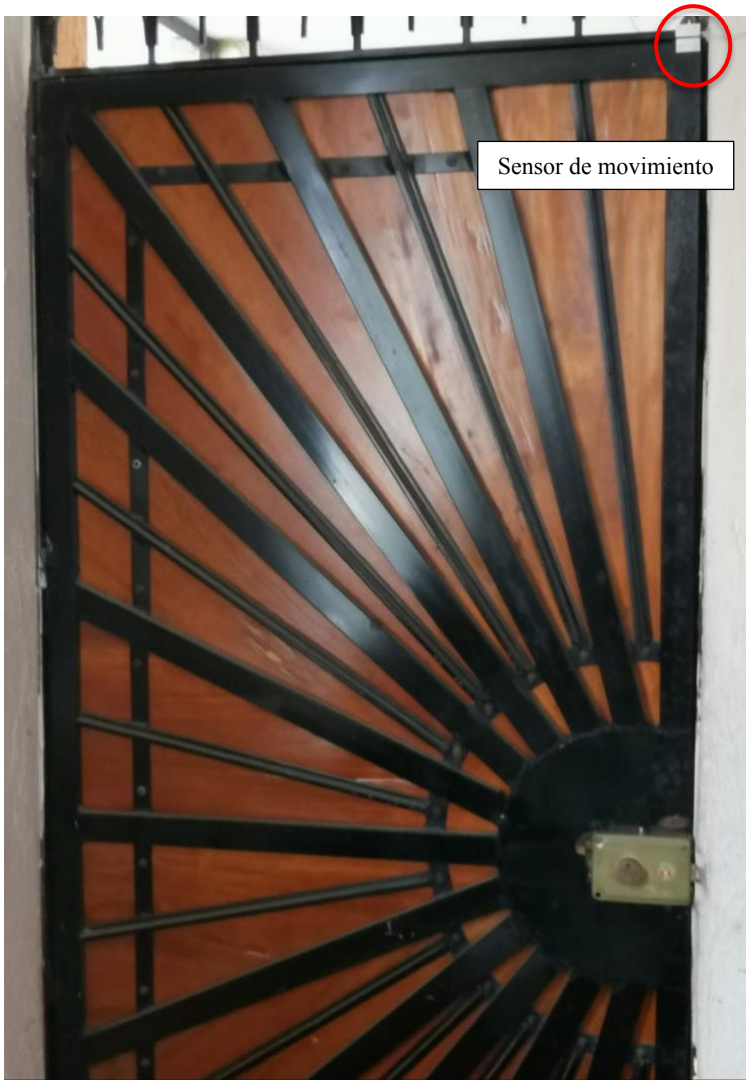
- Sensores Magnéticos
    - La puerta principal del domicilio
    - La antepuerta de seguridad del domicilio.
- Sensor de Golpe
    - La ventana posterior del domicilio
- Sensor de movimiento.
    - El taller de confecciones del domicilio

Figura 72. *Caja de circuito de la Familia Plazarte*

Elaborado por: Boris Cutos y Miguel Recalde

4.5.1. Sensores Magnéticos

Figura 73. *Puerta Principal del domicilio*



Elaborado por: Boris Cutos y Miguel Recalde

Figura 74. *Antepuerta del domicilio*

Elaborado por: Boris Cutos y Miguel Recalde

Figura 75. *Antepuerta del domicilio*

Elaborado por: Boris Cutos y Miguel Recalde

4.5.2. Sensor de Golpe

Figura 76. *Ventana posterior del domicilio*

Elaborado por: Boris Cutos y Miguel Recalde

4.5.3. Sensor de movimiento

Figura *77.* *Talles de confecciones del domicilio*

Elaborado por: Boris Cutos y Miguel Recalde

4.5.4. Resultado de Pruebas

Las métricas para el análisis de los días de prueba fueron

- Estado de los dispositivos
- Tiempo de respuesta entre comunicación del sistema con NodeRED y Firebase
- Tiempo de comunicación entre Firebase y la aplicación Móvil

Tabla 11. *Resultado de pruebas en el domicilio de la familia Plazarte*

Dispositivos	Accesos del domicilio controlados	Días de prueba (noviembre 2021)								
		21-nov			22-nov			23-nov		
		Funciona_ miento del sensor	Tiempo de comunicación con Firebase (seg)	Tiempo de respuesta con la aplicación Móvil (seg)	Funciona_ miento del sensor	Tiempo de comunicación con Firebase (seg)	Tiempo de respuesta con la aplicación Móvil (seg)	Funciona_ miento del sensor	Tiempo de comunicación con Firebase (seg)	Tiempo de respuesta con la aplicación Móvil (seg)
Sensores magnéticos	Puerta principal	ok	1.563	0.893	ok	0.948	0.719	ok	0.936	0.772
	Antepuerta Principal	ok	1.274	0.918	ok	0.735	0.861	ok	0.941	0.983
Sensor de Golpe	Ventana posterior del Domicilio	ok	0.928	0.879	ok	0.993	0.744	ok	0.922	0.931
Sensor de Movimiento	Taller de confecciones	ok	2.911	0.813	El sensor se quemó por un apagón	-	-	ok	1.591	1.201

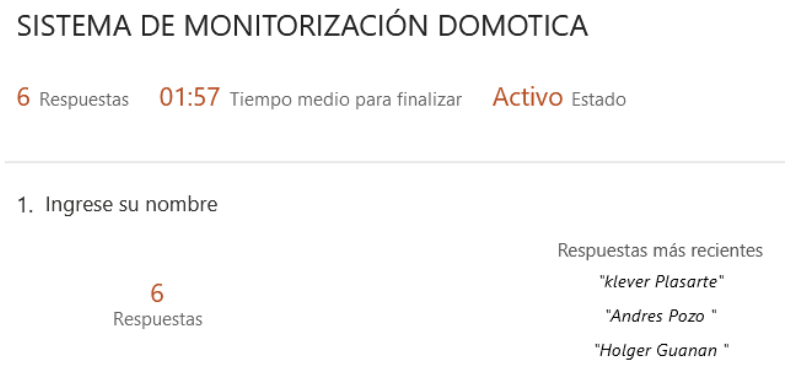
Elaborado por: Boris Cutos y Miguel Recalde, mediante la toma de resultados de Google Analytics

4.6. ENCUESTA

4.6.0. Análisis e interpretación de la encuesta

Encuesta dirigida a los propietarios y usuarios del sistema de monitorización y control domótico. Con el propósito de recopilar información necesaria para el proyecto con tecnología Node-Red a fin de mejorar la seguridad en hogares vulnerables.

Figura 78 *Participantes de la encuesta*



Elaborado por: Boris Cutos y Miguel Recalde, mediante Microsoft Form

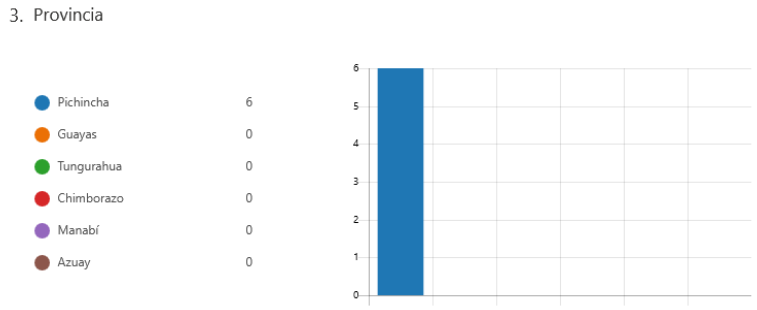
Se realizó la encuesta a las personas beneficiarias del sistema de monitorización domótico y usuarios dentro del domicilio, obteniendo 6 resultados con un promedio de tiempo de casi 2 minutos.

Figura 79. *Edades promedio de encuestados*

Elaborado por: Boris Cutos y Miguel Recalde, mediante Microsoft Form

Al ser un sistema que necesita de un dispositivo móvil los usuarios no sobrepasan los 53 años, podemos observar que las edades están entre los 25 – 43 años con un 66% lo que nos hace suponer que es potencialmente útil para personas a fin a la tecnología.

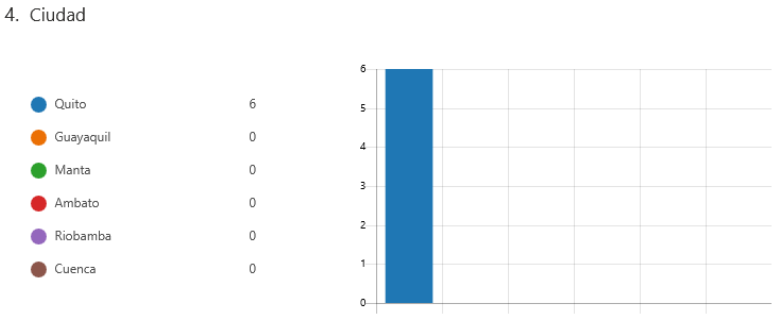
Figura 80. *Encuesta de Provincias*



Elaborado por: Boris Cutos y Miguel Recalde, mediante Microsoft Form

Todas las personas encuestadas se encuentran en la provincia de pichincha, claro esta es posible la implementación en cualquier lugar que cuente con una conexión a internet.

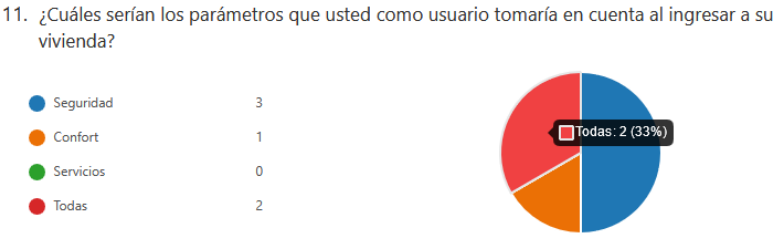
Figura 81. *Ciudad encuestada*



Elaborado por: Boris Cutos y Miguel Recalde, mediante Microsoft Form

La locación se la hizo según el área delimitada por el tema.

Figura 82. *Accesos para considerar dentro de la vivienda*



Elaborado por: Boris Cutos y Miguel Recalde, mediante Microsoft Form

CONCLUSIONES

Se identificó que los dispositivos consumen menos de 1.5 vatios, por lo que el usuario no tiene que preocuparse por el consumo de energía eléctrica; además, el consumo de ancho de banda en la red es mínimo, únicamente se produce al acceder remotamente.
El diseño del sistema de monitorización domótica e inalámbrica genera seguridad, debido a que posee componentes electrónicos de control, transmisión y recepción de datos con una precisión del 100%. Recibiendo notificaciones entre los dispositivos y la aplicación de 0.838 segundos obteniendo una comunicación muy efectiva.

El sistema domótico diseñado en este proyecto de titulación utiliza el protocolo de comunicación MQTT para la interacción de los distintos sensores, lo cual le da una gran ventaja sobre la gestión de los mismos al crear varios canales de comunicación en el bróker MQTT sobre la red WIFI, además dado que el protocolo MQTT se puede implementar sobre dispositivos de limitados recursos de memoria como es el módulo ESP8266, es posible reducir el costo de fabricación del sistema domótico, haciéndolo accesible a la mayoría de los hogares del sector de Chillogallo.

Para el diseño del sistema domótico se consideró enlazar los interruptores magnéticos, sensores de detección de vibración y sensores de detección de movimiento al módulo electrónico ESP8266, a lo cual en este proyecto se denomina un nodo sensor, con el propósito de crear un sistema domótico escalable con la Rasberry PI como su controlador central, consiguiendo de esta manera darle a los usuarios una flexibilidad para la instalación de los nodos sensores en zonas estratégicas de las viviendas por lo que es necesario garantizar una conexión WIFI estable ya que es el medio de comunicación principal de cada uno de los nodos al controlador.

El presente proyecto de tesis logró la implementación del Raspberry pi 4 como un servidor cómodo y compacto. El tamaño de este hace que no ocupe un espacio notable en el hogar, lo que brinda discreción, y la estética del domicilio no se ve afectada, incluso con los sensores que a esta van conectados. La Raspberry cuenta con características internas lo suficientemente potentes para ejecutar el servicio de comunicación lo que permite migrar de dispositivos extremadamente grandes y notables a componentes pequeños y fáciles de administrar e instalar, lo que además representa una notoria satisfacción de los propietarios de los domicilios

beneficiados, ya que el impacto tecnológico incluso en las dimensiones de los aparatos electrónicos tiene una afectación positiva en los usuarios. Las Pruebas desarrolladas sobre el servidor demuestra que el funcionamiento es el esperado, ya que este es el eslabón principal en receptar peticiones y emitir una señal mediante comunicación MQTT a los clientes conectados mediante la aplicación móvil.

Se menciona que el mercado no está saturado con sistemas domóticos. Los habitantes del sector de Chillogallo y el público en general muestran el interés esperado inicialmente al levantar el presente proyecto de monitorización domótica.

RECOMENDACIONES

Para un funcionamiento idóneo es importante que los dispositivos cuenten con una buena señal de internet conectando al dispositivo Raspberry mediante un cable de red y al Arduino situándolo cerca de la antena wifi del departamento o casa, los sensores tienen que estar ubicados en lugares altos, con buena circulación de viento y evitando ser identificados a simple vista

El sensor de movimiento tiene un radio de espectro de entre 3 y 7 metros fuera de estos rangos no vamos a poder recibir una buena lectura de las señales emitidas, la aplicación clasifica el tipo de notificación que revive y de que área de la casa viene permitiendo al usuario habilitar o deshabilitar en caso de que así lo amerite.

Los dispositivos donde se va a instalar la aplicación deben contar con un sistema operativo actualizado, igual o superior a Android 19. Para sistemas IOS se recomiendo una versión igual o superior a la iOS 10. Con esto garantizamos que la App funcione correctamente.

Para la instalación del sistema de control domótico, se tiene que considerar el área de cobertura de la red inalámbrica, esta debe llegar a todos los dispositivos (Raspberry pi 4 – Arduino) con el fin de que exista una buena comunicación. La realización de una inspección en sitio casa o local comercial es muy recomendada para que la cobertura inalámbrica sea la adecuada.

Para este proyecto se utilizó el protocolo MQTT, con el cual el sistema domótico es escalable en la instalación de nuevos sensores que cuantifiquen otras áreas como la temperatura de la vivienda, detector de humo, encendido automático de lámparas, etc., por lo que se recomienda a posteriores proyectos considerar la implementación de nuevas topologías para configurar sistemas domóticos más complejos.

LIMITACIONES

Encontrar la mejor tecnología en Raspberry y Ardunio para ser utilizado en el sistema domótico, ya que el Ecuador no posee los mejores insumos para implementaciones de sistemas de esta índole. Los costos son elevados y los sensores mas adecuados (de mejor calidad) para un despliegue de garantía del sistema domótico suelen sumar más valor al costo total, sin contar con el costo de desarrollo de la aplicación móvil, por lo que una implementación normal de este sistema seria un desembolso de unos 400 dólares americanos para un domicilio, lo cual puede ser excesivo para familias de bajos recursos, incluso a sabiendas que en su presupuesto la seguridad no es lo primordial, o dicho de otra manera, no se tiene una cultura en que la seguridad mediante la tecnología sea algo en lo que se quiera invertir.

PROSPECTIVAS

El presente sistema tiene la capacidad de crecer implementando nuevas funcionalidades como nuevos sensores, con mejores capacidades, inclusive reduciendo los costos de fabricación del sistema si se compran los implementos al por mayor.

El sistema puede ser implementado en la Universidad politécnica Salesiana para gestionar los ingresos a las aulas, de cuando esta se abrió y cuando se cerro, e incluso para precautelar los vienes inmuebles de ciertos lugares de la universidad en los cuales se tengan insumos costosos. Se puede adicionar al sistema cámaras de seguridad y sensores de movimiento, y la ventaja del sistema es que será lo suficientemente capaz de mantener un control centralizado y fácil de administrar y mas llamativo aun, es que con el gran Data Center que posee la Universidad, se podrá tener un centro de control y almacenamiento de los datos del sistema Domótico con las mejores tecnologías del mercado.

LISTA DE REFERENCIAS

Al-Kuwari, M., Ramadan, A., Ismael, Y., Al-Sughair, L., Gastli, A., & Benammar, M. (Abril de 2018). Smart-home automation using IoT-based sensing and monitoring platform. *2018 IEEE 12th International Conference on Compatibility, Power Electronics and Power Engineering (CPE-POWERENG 2018)*, 1-6. Obtenido de http://www.ieeeprojectmadurai.in/IEEE%202019%20EMBEDDED%20BASEPEPERS/Smart-Home%20Automation%20using%20IoT-based%20Sensing%20and%20Monitoring%20Platform.pdf

Android. (02 de 11 de 2021). *developer.android.com*. Obtenido de https://developer.android.com/studio

Arduino. (2020). *What is Arduino?* Obtenido de https://www.arduino.cc/

AV Electronics. (20 de abril de 2021). *Interruptor Magnético MC 38*. Obtenido de Tienda de dispositivos electrónicos AV Electronics: https://avelectronics.cc/producto/interruptor-magnetico/

Barrio Ruiz, J. (2017). *Sistema domótico abierto de bajo coste para la mejora de la eficiencia energética.* Universitat Politécnica de Catalunya. Obtenido de https://upcommons.upc.edu/bitstream/handle/2117/107028/javier.barrio_122863.pdf

Cal Calleja, J. (2019). *Control Domó}tico basado en el protocolo MQTT*. Valladolid: Universidad de Valladolid. Obtenido de https://core.ac.uk/download/pdf/228074048.pdf

Corinne , B., Kate, B., & Alexander , G. (enero de 2021). *TechTarget*. Obtenido de MQTT (MQ Telemetry Transport) : https://internetofthingsagenda.techtarget.com/definition/MQTT-MQ-Telemetry-Transport

Deeter Electronics. (mayo de 2021). *Reed Switch – How it Works*. Obtenido de https://www.deeterelectronics.com/reed-switch-how-it-works/

Escobar Gallardo, E., & Villazón, A. (2018). Sistema de monitoreo energético y control domótico basado en tecnología internet de las cosas. *{Investigación y Desarrollo, 18*(1), 103--116.

ESP8266 Shop . (junio de 2021). *Online Shop & Blog. Explore the World of the IoT*. Obtenido de ESP8266 NodeMCU Pinout: https://esp8266-shop.com/esp8266-guide/esp8266-nodemcu-pinout/

Firebase. (12 de 10 de 2021). *firebase.google.com*. Obtenido de firebase: https://firebase.google.com/docs?gclid=Cj0KCQiA47GNBhDrARIsAKfZ2rADZok0tYX3n796Je2T_eCqJTPq-gkcQ8nEpsIzn-WeB6b77kwx0pUaAk9lEALw_wcB&gclsrc=aw.ds

Flutter. (19 de 11 de 2021). *docs.flutter.dev*. Obtenido de https://docs.flutter.dev/get-started/install

Flutter. (19 de 11 de 2021). *docs.flutter.dev*. Obtenido de https://docs.flutter.dev/get-started/install/macos

flutter.dev. (03 de 10 de 2021). *FlutterFire*. Obtenido de https://firebase.flutter.dev/docs/messaging/usage

Hardwarelibre. (junio de 2021). Obtenido de NodeMCU: la plataforma de IoT de código abierto: https://www.hwlibre.com/nodemcu/

HiveMQ. (2021). *MQTT Essentials.* Obtenido de https://www.hivemq.com/mqtt-essentials

Huidrobo Moya , J. M., & Milan Tejedor, R. (2010). *Manual de domótica.* España: Creaciones Copyright SL.

IBM. (2021). *IBM Emerging Technologies*. Obtenido de https://research.ibm.com/labs/uk/

Infobae. (20 de junio de 2020). *América Latina, la región que más cayó en el ranking mundial de países pacíficos*. Obtenido de https://www.infobae.com/america/america-latina/2020/06/20/america-latina-la-region-que-mas-cayo-en-el-ranking-mundial-de-paises-pacificos

Institute for Economics and Peace. (2020). *Global peace index. Institute for Economics and Peace*.

Instituto Nacional de Estdítica y Censos. (2019). *Encuesta de Victimización y Percepción de Inseguridad*. Obtenido de https://anda.inec.gob.ec/anda/index.php/catalog/673

Jaitman, L., Caprirolo, D., Granguillhome Ochoa, R., Keefer, P., Leggett, T., Lewis, J., . . . Torre, I. (2017). *The Costs of Crime and Violence: New Evidence and Insights in Latin America and the Caribbean* . Washington, DC: Banco Interamericano de Desarrollo. doi: http://dx.doi . org/10.18235/0000615

Jost, D. (29 de Julio de 2019). *FIERCE Electronics*. Obtenido de What is an IR sensor?: https://www.fierceelectronics.com/sensors/what-ir-sensor

Karimi, K., & Krit, S.-d. (2018). Systems and technologies for Smart Homes/Smart Phones: A study and comparison. *Proceedings of the Fourth International Conference on Engineering y MIS 2018*, 1--7. Obtenido de https://dl.acm.org/doi/abs/10.1145/3234698.3234706?casa_token=xmfn4e-

6EtkAAAAA%3A43yJ6-cysWhugE8cwzHSgJQSPssGsg4qKKeY9lfYPvQTwviiwvGuien_l-nRRf7w96gv1hsFzknnNiE

LAST MINUTE ENGINEERS. (2021). *How HC-SR501 PIR Sensor Works & Interface It With Arduino*. Obtenido de https://lastminuteengineers.com/pir-sensor-arduino-tutorial/

Leantec. Robotics. Arcade. Electronics. (Junio de 2019). *MODULO INTERRUPTOR DE VIBRACION KY-002 VIBRATION SWITCH*. Obtenido de https://leantec.es/tienda/modulo-de-interruptor-de-vibracion-ky-002-vibration-switch/

Lekić, M., & Gardašević, G. (2018). IoT sensor integration to Node-RED platform. *2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH)*. doi:10.1109/INFOTEH.2018.8345544

Madakam, S., Ramaswamy, R., & Tripathi, S. (2015). Internet of Things (IoT): A Literature Review. *Journal of Computer and Communications*, 164-173. doi:10.4236/jcc.2015.35021

MCI Electronics. (2020). *Raspberry Pi 3 Modelo B+ Catalogo*. Obtenido de https://www.mcielectronics.cl/shop/product/raspberry-pi-3-modelo-b-25300

Ministerio de Justicia y Paz. (14 de noviembre de 2020). *Robo a casa habitación*. Obtenido de Observatorio de la volencia: http://observatorio.mj.go.cr/tipo-de-hecho-violento/robo-casa-de-habitacion

Nod-RED. (2021). *Node-RED Low-code programming for event-driven applications*. Obtenido de https://nodered.org/#get-started

Palacios, J. L. (2020). La protección del usuario en la domótica y las facilidades que ofrece. *Revista Científica Sinapsis*. doi:https://doi.org/10.37117/s.v1i16.276

Policía Nacional del Ecuador. (2020). *Índice de robos a vivienda*. Obtenido de https://www.policia.gob.ec/

pub.dev. (03 de 10 de 2021). Obtenido de pub.dev: https://pub.dev/packages/firebase

Rasperry. (2020). *What is a Raspberry Pi?* Obtenido de https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi

Sancho, P. (20 de abril de 2020). *Techedge Group*. Obtenido de Fundamentos de Node-RED: https://www.techedgegroup.com/es/blog/fundamenos-node-red

The HiveMQ Team . (20 de Abril de 2015). *MQTT Security Fundamentals*. Obtenido de Authentication with Username and Password - MQTT Security Fundamentals: https://www.hivemq.com/blog/mqtt-security-fundamentals-authentication-username-password

The HiveMQ Team. (11 de mayo de 2015). *MQTT Security Fundamentals*. Obtenido de TLS/SSL - MQTT Security Fundamentals: https://www.hivemq.com/blog/mqtt-security-fundamentals-tls-ssl/

The HiveMQ Team. (4 de mayo de 2015). *MQTT Security Fundamentals*. Obtenido de Authorization - MQTT Security Fundamentals: https://www.hivemq.com/blog/mqtt-security-fundamentals-authorization/

The Linux Foundation. (2021). *OpenJS Foundation*. Obtenido de https://openjsf.org/

Yuan, M. (18 de junio de 2021). *IMB Developer*. Obtenido de Conozca NodeMCU y su placa DEVKIT: https://developer.ibm.com/es/tutorials/iot-nodemcu-open-why-use/