ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN

Diseño e implementación de un parqueadero automatizado en el edificio principal de la FIEC utilizando una red de sensores basada en IoT.

PROYECTO INTEGRADOR

Previo a la obtención del Título de:

INGENIERO EN TELECOMUNICACIONES

Autores:

Christian Anibal Campoz Chiliquinga

Jimmy Marcelo Guillin Camacho

GUAYAQUIL - ECUADOR

Año: 2022

DEDICATORIA

Este proyecto está dedicado a mi madre Gilma Irlanda Guillin Camacho por darme la vida y ofrecerme incondicionalmente su apoyo.

A mis queridos tíos Edith Guillin y Washington Guillin que han sido mi orgullo y fuente de inspiración, gracias a ellos por haber siempre confiado en mí.

A mi abuelita Julieta Camacho por haberme inculcado buenos valores y principios que han hecho de mí una gran persona y por cuidarme con todo su amor y cariño.

Jimmy Marcelo Guillin Camacho.

Este trabajo se lo dedico a varios seres importantes que estuvieron a lo largo de este camino: A mis padres Olger Campoz y Vitalia Chiliquinga quienes día a día con amor se esforzaron para sacarme adelante. A mis hermanas Kerly & Melanie que siempre me escucharon y aconsejaron.

A mis abuelos maternos Fernando & Pastora. En especial a mi abuelito paterno Raúl que ya no está conmigo pero que fue el impulso que necesitaba en este trayecto; sé que desde el cielo está orgulloso, ya que estoy cumpliendo lo que me pidió en sus últimos días: “deja mi apellido muy alto”.

A Allison, mi compañera de vida que en estos últimos años ha estado a mi lado, dándome luz en los momentos más oscuros y por su puesto a Dios que sin él nada de esto hubiera sido posible.

Christian Anibal Campoz Chiliquinga.

AGRADECIMIENTOS

A lo largo de este período he conocido personas maravillosas que han aportado con una parte esencial para mi crecimiento personal y profesional.

Quiero agradecer a todos las personas que han formado parte de esta etapa principalmente a mi familia por todo el esfuerzo y apoyo brindado.

A mi profesor el Ph.D. Francisco Novillo quien nos ha acompañado en el desarrollo de este proyecto.

A mi tutora la Ph.D. Patricia Chávez por la paciencia y los consejos brindamos para la elaboración del proyecto.

A mis amigos Williams, Carlos, Christopher, Teresa, Anita, Fátima y Christian con quienes he compartido momentos increíbles y me han motivado a seguir adelante.

Finalmente, quiero dar un agradecimiento especial a Ruby Sáenz mi mejor amiga que supo darme su mano y levantarme en una de las peores situaciones que atravesaba mi vida.

Jimmy Marcelo Guillin Camacho.

Dios pone en la vida a las personas correctas en el momento correcto y es por eso por lo que agradezco de todo corazón a todas las personas que formaron parte de esta etapa de mi vida, primos, tíos, amigos, a mi compañero Jimmy Guillin ya que sin él este trabajo no hubiera sido posible.

A mi profesor el Ph.D. Francisco Novillo quien nos ha acompañado en el desarrollo de este proyecto y a mi tutora la Ph.D. Patricia Chávez, que a pesar de la distancia ha sabido brindarnos su ayuda y su paciencia con la elaboración de este trabajo.

Un agradecimiento especial a mis amigos que han estado a lo largo de esta carrera: Karla, Manuel y Gilliana, sin ellos no hubiera sido tan agradable y amena esta etapa.

Christian Anibal Campoz Chiliquinga.

DECLARACIÓN EXPRESA

"Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; Christian Anibal Campoz Chiliquinga y Jimmy Marcelo Guillin Camacho damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"

---------- ----------

Christian Anibal Campoz Chiliquinga **Jimmy Marcelo Guillin Camacho**

EVALUADORES

Firmado electrónicamente por:
FRANCISCO VICENTE NOVILLO PARALES

PhD. Francisco Vicente Novillo Parales

PROFESOR DE LA MATERIA

PhD. Patricia Ximena Chávez Burbano

PROFESOR TUTOR

RESUMEN

El acceso a una plaza de estacionamiento en el área de FIEC resulta complicado, debido a que, existen una cantidad significativa de funcionarios que trabajan en la facultad y a su vez cada uno de ellos dispone de un automóvil para su movilización generando así que un alto número de automotores circulen en busca de una plaza de parqueo.

Esto provoca problemas a los usuarios que utilizan este lugar para estacionar sus vehículos, tales como retraso a sus lugares de trabajo, gasto innecesario de combustible, pérdida de tiempo al no encontrar un lugar disponible causando frustración e incomodidad.

Para solucionar el problema se implementará un modelo de parqueadero automatizado mediante el diseño de una red de sensores inalámbricos en el área del parqueadero del decanato de la FIEC, en dónde cada plaza de parqueo tendrá un sensor de proximidad HC-SR04, el cual permitirá determinar si ese espacio está ocupado, de esta manera se puede monitorear y adquirir datos en tiempo real usando una computadora de placa única raspberry pi 3 con el protocolo MQTT.

Los resultados de la lectura de los sensores y el conteo realizado en los nodos se visualizan en una pantalla LED que mostrará el número de lugares de parqueo disponibles. Adicionalmente, los datos son enviados a la nube para informar en tiempo real a los usuarios el número de lugares disponibles a través de un bot de Telegram llamado “Parqueadero FIEC”.

ABSTRACT

Access to an available parking space in the FIEC area is complicated. There is a significant number of officials who work in this faculty and each of them has a car for their mobilization, thus generating a high number of vehicles circulating in search of a parking space.

This continuous search causes problems for users who employ this area to park their vehicles, such as delays to their places of work, unnecessary fuel consumption, loss of time by not finding an available space, causing frustration and inconvenience.

To solve the problem, an automated parking model will be implemented by designing a wireless sensor network in the FIEC deanery parking area, where each parking space will have a HC-SR04 proximity sensor, which will allow determining if that space is occupied, this way you can monitor and acquire data in real time using a raspberry pi 3 single board computer with the MQTT protocol.

The results of the reading of the sensors and the count made in the nodes are displayed on a LED screen that will show the number of available parking spaces. Additionally, the data is sent to the cloud to inform users in real time of the number of places available through a Telegram bot called "FIEC Parking".

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

CAPÍTULO 1

1. Introducción

En la actualidad las telecomunicaciones son un factor importante y necesario, que promueven desarrollo y mejora para la sociedad. El desarrollo de este campo avanza a pasos agigantados ya que en un futuro no muy lejano serán de gran importancia para futuras generaciones que día a día necesitan de servicios de telecomunicaciones más avanzados y sofisticados, de esta manera poder intercomunicar al mundo [1]. Gradualmente se han convertido en un medio de vital importancia, que conecta a la sociedad y hace que la interacción entre las personas sea más eficiente y sencilla, empleando dispositivos que requieren intervención humana limitada. Las telecomunicaciones se han convertido en un medio de vital importancia, que actualmente conecta a la sociedad y hace que la interacción entre ellas sea más eficiente y sencilla, esto usando un sinnúmero de dispositivos que, con previa intervención humana, realizan acciones que hacen posible avanzar a la par con la tecnología.

Las telecomunicaciones se han vuelto un factor esencial para todas las personas, empresas y las agencias gubernamentales, con el incremento exponencial de usuarios conectados a los sistemas de telecomunicaciones, se producirá una mayor demanda en la capacidad y necesidad de comunicación [2]. Varios lugares privados y públicos usan actualmente dispositivos para automatizar procesos de telecomunicaciones, por ejemplo, saber por medio de IoT el estado de una red o el número de lugares disponibles en una zona específica.

Las placas de desarrollo como arduino, raspberry pi y demás, tienen la posibilidad de vincular grandes variedades de componentes a cualquier elemento a través de la detección,

almacenamiento, procesamiento e integración de información descubierta por los sensores. Los componentes mencionados anteriormente forman una red que es "consciente" de su entorno y puede determinar, varios aspectos, entre ellos: si un objeto está estático o en movimiento [3]. En este caso para la aplicación esta red de sensores va a ser capaz de identificar la presencia o ausencia de un vehículo en un estacionamiento.

En el presente trabajo se va a diseñar e implementar una red de sensores inalámbricas (WSN, por sus siglas en inglés, Wireless Sensor Network) basada en IoT, los elementos que forman parte de esta red son: sensores de ultrasonido cuya funcionalidad se fundamenta en el envío de una señal a una frecuencia alta, que no es percibido por el oído humano. Este pulso se refleja en los obstáculos que se encuentran dentro del rango de alcance y es receptado por el sensor, de esta forma se calcula el tiempo entre el lanzamiento del pulso y la llegada de este, para luego determinar la distancia a la que se encuentra un obstáculo [4]. De esta manera, es posible detectar si existe o no un vehículo dentro de una determinada plaza de parqueo, la información en forma de pulso es recibida por un nodo sensor el cuál consta de un microcontrolador ESP32, este nodo a su vez es programado para convertir dicha señal en un número entero, para posteriormente efectuar un conteo de total de los estacionamientos disponibles mediante el número de sensores conectados en un mismo nodo.

Los nodos sensores son clientes MQTT, que actúan como publicadores, encargándose de enviar el dato obtenido del total de aparcamientos disponibles en cada nodo hacia el bróker MQTT a través de un tópico especificado. El bróker MQTT se implementa en una raspberry pi 3 modelo B y es el que recolecta la información de todos los nodos y la procesa utilizando el ambiente de programación node RED para obtener un total general de estacionamientos

disponibles, enviando esta información a la pantalla LCD por comunicación I2C y a la aplicación móvil Telegram.

1.1. Definición del problema

El acceso a una plaza de estacionamiento en el área junto al edificio principal de la FIEC resulta complicado. Debido a que, existen una cantidad significativa de funcionarios y estudiantes que acuden a la facultad y a su vez disponen de automóviles para su movilización generando así un alto número de automotores circulen en busca de una plaza de parqueo. Esto provoca problemas principales al conductor como la pérdida de tiempo al no hallar un parqueadero generando a su vez frustración e incomodidad además otro problema que se genera afectando el ámbito económico pérdida de dinero al gastar innecesariamente combustible.

El proyecto pretende solucionar problemas principales que afectan a los funcionarios y estudiantes que acuden a la FIEC como la pérdida de tiempo al no hallar un parqueadero generando a su vez frustración e incomodidad además otro problema que se genera afectando el ámbito económico pérdida de dinero al gastar innecesariamente combustible. Estos problemas afectan a varias instituciones que dispongan de un lugar de parqueo para sus empleados.

1.2. Justificación del problema

Con el incremento constante del número de vehículos adquirir una plaza de parqueo es casi imposible, es por esto qué nace la necesidad de implementar un parqueadero inteligente que permita identificar de manera fácil las plazas de estacionamiento que se encuentran disponibles. Además de contribuir a la reducción por contaminación de CO2 en el medio ambiente causado por los automóviles al pasar mayor tiempo los motores encendidos.

Por otro lado, también se pretende solucionar problemas de impacto social que afectan a los estudiantes de forma indirecta como el retraso en sus clases y aprendizaje cuando los profesores llegan tarde.

1.3. Objetivos

Este proyecto tiene como objetivo principal "Desarrollar un sistema de automatización para el parqueo de decanato de la Facultad de Ingeniería en Electricidad y Computación a través de una red de sensores basada en IoT." Para alcanzar esta meta, se plantean tres objetivos específicos.

- Identificar el área de trabajo para implementación del sistema.
- Diseñar el sistema que permitirá automatizar el parqueadero.
- Implementar el sistema de automatización para parqueadero.

1.4. Propuesta de Solución

Para solucionar el problema planteado se implementará un parqueadero inteligente. Los sensores serán ubicados en lugares adecuados para evitar averías debido a los distintos factores físicos del medio ambiente, asegurando la detección de los vehículos en cada plaza de parqueo. De esta manera se puede monitorear el parqueadero y adquirir datos en tiempo real, los datos adquiridos se visualizan en una pantalla LED instalada en la entrada del parqueadero para conocer el número de lugares disponibles y a su vez se puede visualizar la disponibilidad de parqueaderos mediante la aplicación de telegram.

1.5. Metodología

Para este proyecto se ha empleado la metodología de gestión de proyectos conocida como "Waterfall" (cascada). El proyecto fue dividido en procesos que se ejecutaron en forma secuencial: inicio, planificación, ejecución, control, y cierre. En el inicio se definieron el problema, y el

alcance y las limitaciones de la solución. Durante la planificación se estudió el área donde se implementará la solución, se seleccionó el protocolo de comunicación que se ajusta al área estudiada y se diseñó el sistema de la red de sensores. En la etapa de ejecución se realizaron implementaciones de prueba de los diferentes componentes del sistema. Estas implementaciones se emplean en el proceso de control para las pruebas de campo a baja escala del sistema de sensores y su interacción con los otros componentes de la solución. Adicionalmente se realizan las correcciones de los errores encontrados en las pruebas realizadas. Finalmente, en la etapa de cierre se realizó la implementación a gran escala del prototipo y se unificó la documentación de todo el proceso.

CAPÍTULO 2

2. Marco Teórico

En el presente capítulo se detallan los fundamentos teóricos de la solución propuesta. Uno de los conceptos más importantes a entender es la red de sensores, y la definición de los elementos para su implementación: placa de desarrollo Esp32 Wi-Fi, sensor ultrasónico JSN-SR04T, pantalla LED, y protocolo de comunicación I2C que permitirá la intercomunicación entre los elementos del sistema.

2.1. Placa de desarrollo ESP32 Wi-Fi

El módulo ESP32 fue elegido por qué ofrece una solución Wi-Fi integrada y certificada, es decir, proporciona radio comunicación alámbrica e inalámbrica. Adicionalmente, tiene un procesador integrado de dos núcleos cuyas frecuencias de operación son programables dentro del rango 80-240 [MHz], e interfaces de tipo externo tales como: I2C, SPI, UART, I2S, Ethernet, tarjetas SD e interfaces capacitivas y táctiles [3].

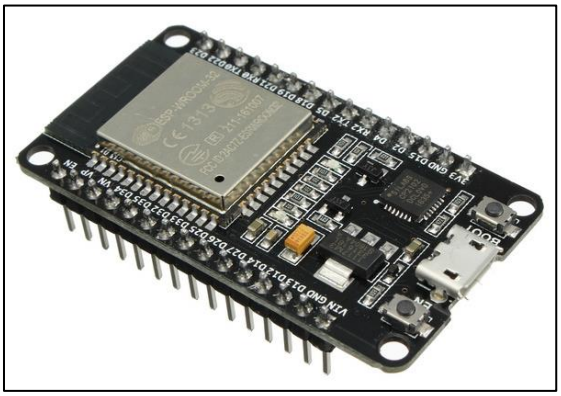


Figura 2.1. Nodo MCU ESP32 [6].

A continuación, se detallan los elementos del protocolo de comunicación I2C que va a ser usado para poder establecer conexión entre los sensores y la pantalla LED del sistema.

2.2. Señal de Reloj

Señal que sólo refleja dos estados lógicos: ALTO (‘1’) y BAJO (‘0’) [4], representados con una onda cuadrada [5]. Se emplea para coordinar las acciones de dos o más circuitos.

2.3. Protocolo I2C

El protocolo I2C usa un puerto de comunicación serial, por el cual se determina el paquete de datos y las conexiones físicas para la transmisión de bits entre 2 terminales digitales. Dicho protocolo permite enlazar mediante sus dos líneas hasta un máximo 127 terminales esclavos, permitiendo velocidades de 100, 400 y 1000 [kbits/s] e incluye la confirmación de datos recibidos. Uno de los dispositivos que se pueden conectar son los sensores digitales propuestos [6].

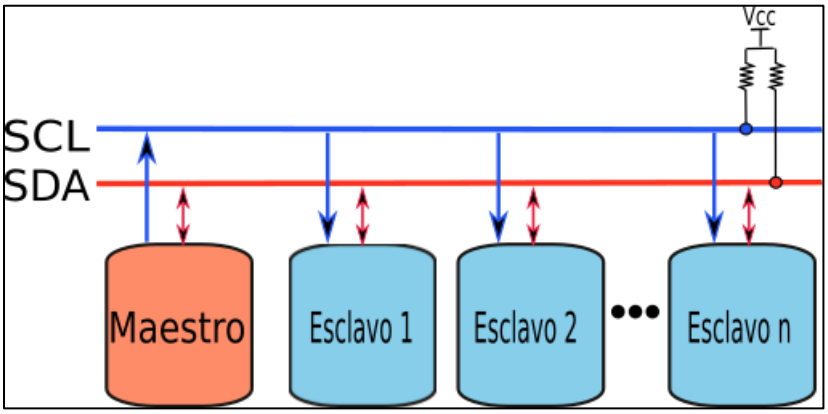


Figura 2.2. Conexión dispositivos bus I2C [10].

En la figura 2.2 se puede observar las conexiones entre dispositivos usando el protocolo I2C: la línea SDA que realiza la función de transmitir datos y la línea SCL que se encarga de

transmitir la señal de reloj. En conjunto las dos líneas controlan el dispositivo maestro, el cual maneja a los dispositivos esclavos [7].

El funcionamiento de este protocolo es sencillo. Cuando la línea SDA cambia de valor alto a valor bajo durante un valor alto de la línea SCL, se inicia la transferencia de datos. La línea SDA transfiere un bit por cada pulso de la línea SCL (figura 2.3). Cuando la línea SDA cambia de valor bajo a valor alto durante un valor alto de la línea SCL, la transmisión de datos se detiene.

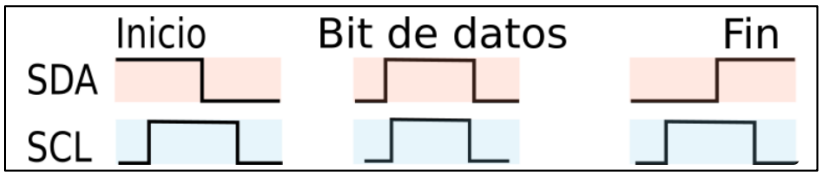


Figura 2.3. Protocolo I2C [10].

El bit de reconocimiento es de nivel alto colocado en el bus por el transmisor, un valor bajo colocado por el receptor [7].

2.4. Protocolo MQTT

Es un protocolo creado para IoT que permite enfocarse en la conectividad máquina a máquina (M2M por sus siglas en inglés) y se basa en TCP/IP. Se destaca frente al protocolo HTTP, pues este último en cada transmisión se ejecuta través de un enlace o conexión, en MQTT las conexiones se mantiene abiertas y se "reutilizan" en otras comunicaciones. [8].

MQTT funciona como un servicio de mensajería “push” en la cual los destinatarios no necesitan usar la aplicación para recibir las notificaciones. Haciendo uso del modelo “publicador/suscriptor” (pubsub)., cuando el dispositivo recibe nueva información notifica al usuario [9]. Los clientes requieren conectarse a un servidor central llamado bróker.

Para elegir los mensajes enviados a cada cliente, los mensajes se organizan en secciones jerárquicas. Un cliente remite un mensaje sobre un determinado tópico y el bróker o servidor lo reenvía a otros clientes que se han suscrito a dicho tópico.

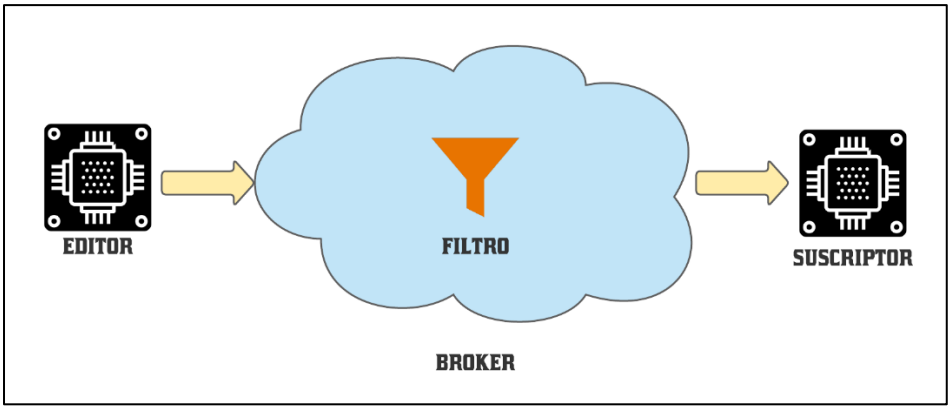


Figura 2.4. Sistema pub-sub (publisher-subscriber).

El MQTT Bróker es clave para la comunicación al ser el programa encargado de recibir los mensajes y distribuirlos en un sistema pubsub. Adicionalmente, los broker Open Source son la más amplia distribución en el área doméstica [10] En la figura 4 se puede observar el sistema publisher-subscriber, este servicio en la nube de MQTT totalmente administrado proporciona un agente de MQTT público que se puede utilizar para el aprendizaje, las pruebas o la creación de prototipos de MQTT [10].En la figura 2.4 se puede observar un sistema publisher-subscriber. El servicio en la nube proporciona un agente de MQTT público que se puede utilizar para el aprendizaje, las pruebas o la creación de prototipos de MQTT [10].

Para este sistema no existe protección de privacidad para el Bróker de acceso público, cualquier dispositivo puede publicar y suscribirse a temas en él. Este agente público de MQTT es un clúster EMQ X Enterprise de dos nodos [11].

2.5. Topologías

Una topología representa la manera en la que se encuentra estructurada una red tanto a nivel lógico como físico [12]. Es decir, es la disposición de la red, incluyendo sus nodos y líneas de conexión. [13]

La topología lógica se refiere a la naturaleza de las rutas de la señal que van de un nodo a otro, mientras que la topología física es la disposición geométrica real de las estaciones de trabajo. En muchos casos, la topología lógica es la misma que la topología física. Sin embargo, varias redes están establecidas físicamente por una distribución en estrella, pero actúan lógicamente como redes de bus o anillo [13].

Existen varios tipos de topología, pero nos centraremos en la topología estrella que es la que se usa para interconectar los nodos sensores al servidor. Cada terminal se conecta de forma directa al servidor sin intermediarios ya se sea de forma alámbrica o inalámbrica [14]. Además, al aumentar los nodos de la topología, esta maneja una buena comunicación a pesar del gran volumen de nodos de red, sin importar la distancia al que estos se encuentren.

2.6. Wifi 2.4 GHz

Una de las conexiones inalámbricas más usadas en la actualidad es el Wifi, esta tecnología es de vital importancia tanto en hogares, empresas, unidades educativas, demás sectores, la mayoría de los dispositivos con acceso directo a Internet y sus servicios, poseen funciones internas de componentes inteligentes enlazados en la red local y usan un router como central principal de operación [15].

Desde la creación de las redes Wifi, la banda de 2.4 GHz ha sido la más usada en la mayoría de los casos. La banda de 2.4 GHz está particionada en 13 canales con 22 MHz cada uno, por los

cuales viaja el tráfico de red de tus conexiones Wifi, es decir si la banda de frecuencia fuera una autopista de 13 carriles, donde su enrutador es un automóvil que circula por uno de esos carriles. En la banda de 2.4 GHz, estos canales son muy estrechos y se superponen a los canales vecinos [16].

Además, esta banda se utiliza para transmitir las comunicaciones de aparatos de radiofrecuencia, periféricos inalámbricos, microondas entre otros muchos otros dispositivos, tales como los módulos usados para la implementación de este proyecto [16].

2.7. Node RED

Es un software de código abierto donde mediante su entorno de programación visual permite conectar nodos predeterminados estableciendo parámetros respectivos y siguiendo un flujo determinado con la finalidad de vincular terminales de hardware, API y servicios en línea conformando un sistema IoT [17].

Ahora si se usa como herramienta de visualización, esta proporciona fácilmente la captura de varios sucesos de un escenario real, permitiendo adicionar un nivel de inteligencia en nodos que efectúan procesos y conversión de datos en nodos establecidos para integrar estos acontecimientos con cualquier sistema de mensajería MQTT [18].

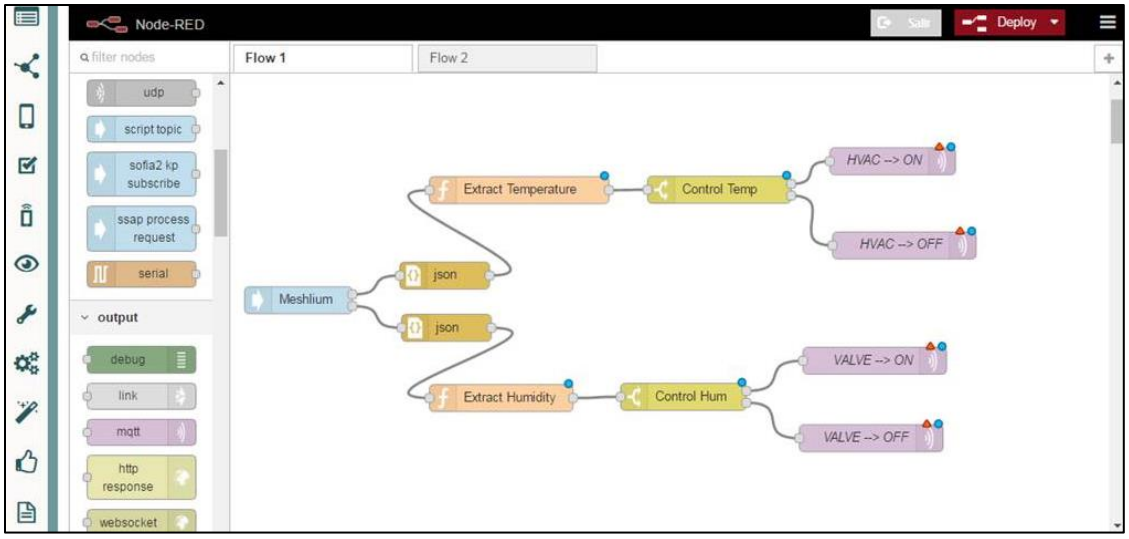


Figura 2.5. Entorno gráfico de programación Node Red [23].

2.8. Protocolo de Comunicación Ethernet

El protocolo de red multinivel o más conocido como Ethernet / IP, es usado para automatizar procesos en las aplicaciones, se basa en TCP / IP. Para realizar configuración, acceso y control de los dispositivos que van a automatizarse usa hardware y software Ethernet. Este protocolo clasifica los nodos dentro de tipos predefinidos de dispositivos, con sus acciones específicas. Por otro lado, el protocolo de red se encuentra apoyado en el protocolo de control e información (CIP) usado en DeviceNet y ControlNet [19].

2.9. Bot de Telegram

La aplicación de mensajería Telegram posee una herramienta denominada bots [20]. que permite automatizar chats para que el usuario obtenga información en tiempo real, sin necesidad de la intervención de personas en el origen. Para este proyecto se usó el bot en el chat denominado "Parqueadero FIEC" el cual ofrece información de disponibilidad de lugares e información de los creadores.

2.10. Raspberry Pi

La Raspberry Pi es una computadora en una placa a la cual se pueden conectar periféricos para la interacción hombre-máquina. Incluye un procesador SoC, memoria de acceso aleatorio (RAM), puertos de audio y video tanto de entrada como salida, conexión a la red, ranura para tarjeta de almacenamiento SD, señal de reloj, puerto de corriente, conexiones para dispositivos externos micro de gama baja, reloj e interruptor para encenderlo o apagarlo [21]. En este caso la Raspberry contiene la programación del sistema de conteo de espacios disponibles y no disponibles del parqueo, esta información la obtiene en conjunto con los sensores instalados en cada uno de los lugares de estacionamiento.

El sistema operativo Raspberry Pi se basa en Debian (distribución de Linux), que es un sistema operativo de código abierto permitiendo personalización e instalación sin pagar licencias [22]. Cabe recalcar que se pueden usar en otros sistemas operativos tales como Ubuntu Mate, Snappy Ubuntu Core, Windows 10 IoT Core, Pinet, etc.

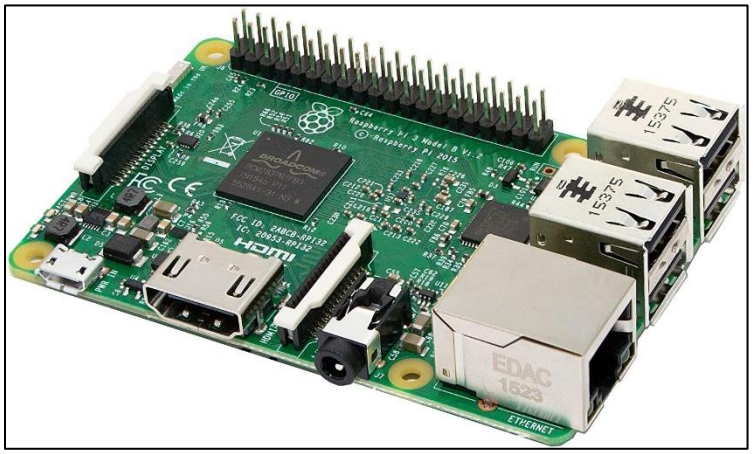


Figura 2.6. Raspberry Pi 3 Modelo B [28].

2.11. Enrutador TP-Link modelo TL-WR840N

Este enrutador emplea conexiones Wi-Fi en la banda de frecuencia de 2.4 GHz con máxima tasa de datos de 60 Mbps [23]. Este dispositivo proporciona conexión de red a la Raspberry Pi, para que esta a su vez envíe la información a la plataforma y el Bot de Telegram pueda informar al usuario acerca de la disponibilidad de lugares de estacionamiento.

Figura 2.7. Router TP LINK TL-WR840N Parte Frontal (izquierda) y Parte Trasera (derecha) [30].

2.12. Sensor Ultrasónico hcsr04

Es un componente que incorpora un par de transductores ultrasónicos que se usan en conjunto para calcular la distancia entre el sensor y un obstáculo. Un transductor emite un "haz" de ondas ultrasónicas y el otro capta las ondas reflejadas. El tiempo que tardan las ondas de sonido en viajar hacia y desde un objeto se puede utilizar para determinar la distancia entre la fuente de sonido y el objeto [24].

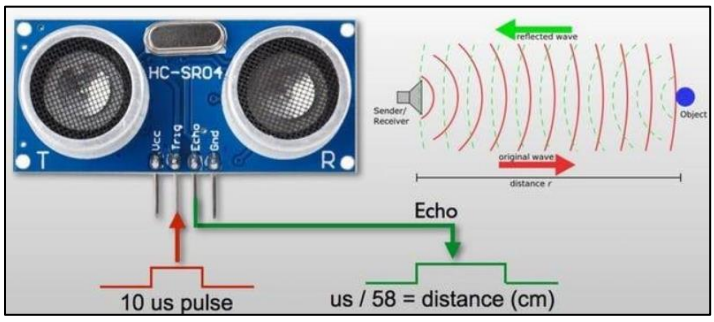


Figura 2.8. Sensor HC-SR04 /Rango de operación [32].

2.13. Base Shield esp32

Esta base está diseñada para ser compatible con las placas base keyestudio esp32, esta placa expande la cantidad de dos filas de pines de 2.54 mm del módulo principal que alimentan sensores ultrasónicos HC-SR04 con un voltaje de CC de 3.3V y un interruptor DIP, para controlar la energía [25].

Tabla 2.1. Parámetros de funcionamiento del sensor ultrasónico.

Parámetro	Valores
Tensión de alimentación (DC)	6-9V
Corriente de funcionamiento	60mA
Potencia máxima	0,3W
Temperatura	De -25℃ a +65℃
Dimensiones	30mm x 20mm

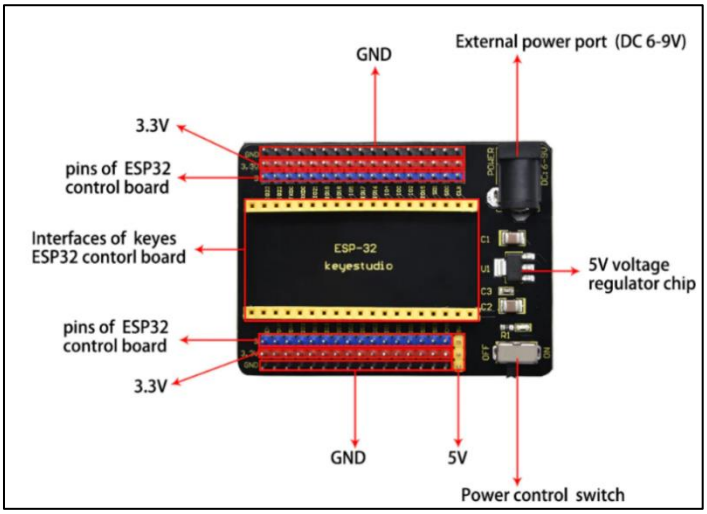


Figura 2.9. Base shield KEYESTUDIO para módulo ESP32 [34].

CAPÍTULO 3

3. Diseño de la solución

El diseño del prototipo del parqueadero automatizado se lo realizó tomando en cuenta que en el aparcamiento ubicado junto al edificio principal de la FIEC no cuenta con un sistema inteligente que permita verificar la disponibilidad de un lugar de parqueo, lo cual genera que los funcionarios y estudiantes que acuden a la facultad pierdan tiempo buscando un lugar de estacionamiento y dinero al gastar innecesariamente combustible, debido al alto número de vehículos que circulan buscando una plaza de parqueo.

El sistema desarrollado nos permitirá conocer el número de lugares de parqueo disponibles a través de una pantalla LED ubicada en la entrada del parqueadero, adicionalmente se podrá verificar la disponibilidad a través de una aplicación móvil.

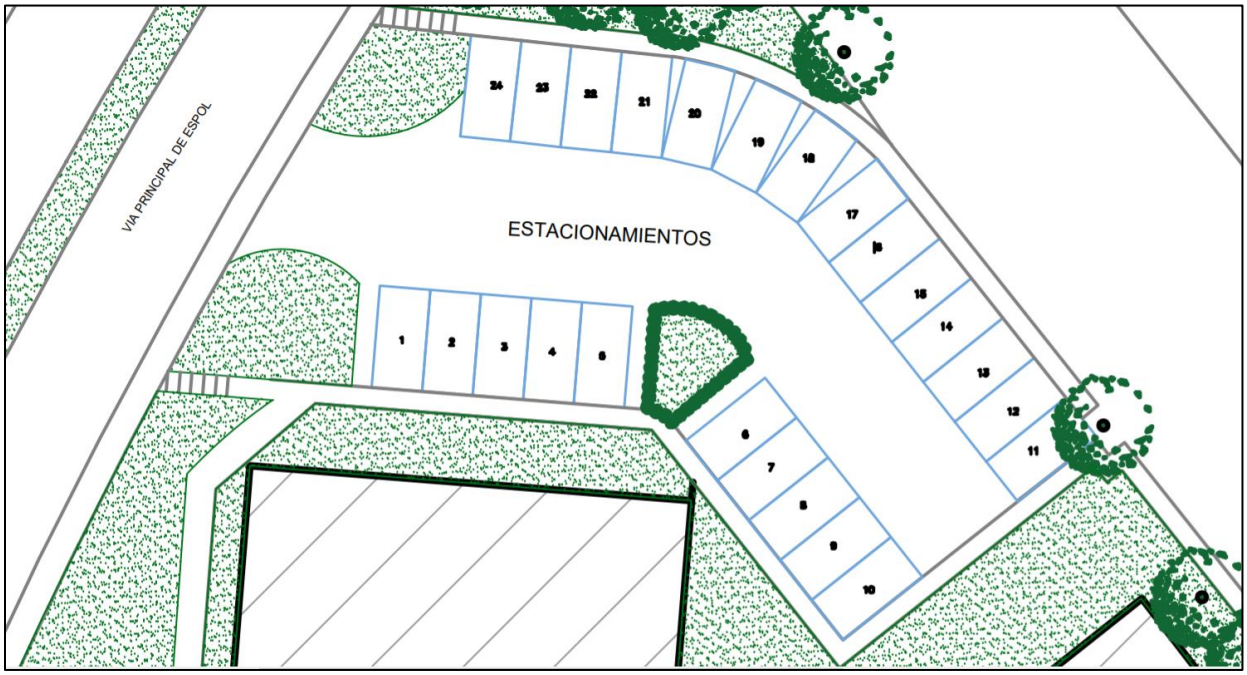


Figura 3.1. Esquemático de escenario de implementación en AUTOCAD.

Figura 3.2. Parqueadero FIEC.

En la figura 3.2 podemos observar el escenario real en el cual va a estar implementado de ser necesario el sistema de red de sensores. El parqueadero ubicado cerca del decanato de la FIEC cuenta con 24 espacio numerados y repartidos equitativamente, es decir 12 a cada lado.

Se eligió este lugar a que existe una gran cantidad de personal administrativo, profesores, estudiantes que poseen un automotor y que al asistir a su lugar de trabajo requieren de una plaza de parque libre y muchas de las veces les resulta complicado encontrar un parqueo disponible por la gran demanda de los usuarios.

Para implementar la red de sensores es necesario considerar factores como: Ubicación, infraestructura, y condiciones climáticas; esta última es importante ya que en la época de invierno se debe procurar que los dispositivos no se mojen, puesto que esto puede ocasionar daños en el sistema al igual que soportar condiciones altas de temperatura ya que al ser un espacio abierto formado en su mayoría de material asfaltico tiende a sobre calentarse.

3.1. Descripción del sistema

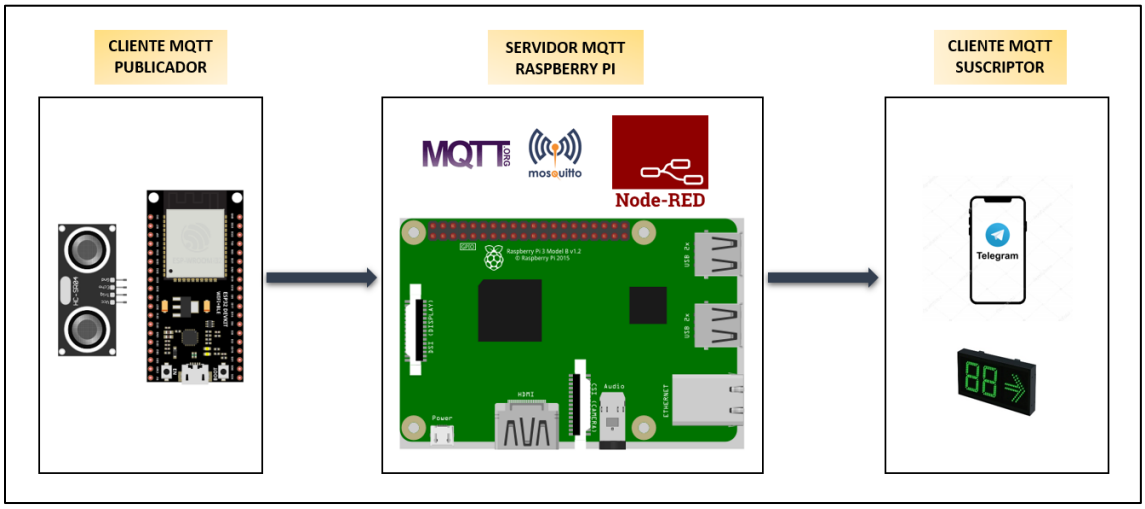


Figura 3.3. Diagramas de bloques de la arquitectura MQTT del sistema implementado.

Para el desarrollo del sistema se ha utilizado la arquitectura MQTT, la misma que consta de un publicador que emite la información, un suscriptor que recibe la información y un servidor MQTT que distribuya la información a los clientes que se encuentren conectados a este. Para establecer comunicación entre estos componentes se debe tener en cuenta el concepto topic o tema en español, puesto que los emisores y receptores deben estar suscritos a un topic común.

El prototipo diseñado consta de las partes mostradas en el diagrama de bloques de la figura 3.3, dónde el publicador MQTT es un nodo sensor que consta de un sensor ultrasónico y un módulo ESP32 el cual emite una señal a través de Wifi cuando detecta un vehículo, para la comunicación Wifi entre el publicador y el servidor se utiliza un gateway al cual deben estar conectados tanto el publicador como el servidor MQTT, el servidor MQTT está implementado en una raspberry PI 3 y es una entidad intermediaria que permite que los clientes de MQTT se comuniquen, este servidor al recibir los mensajes publicados por los clientes, filtra los mensajes por topic y los distribuye a los suscriptores, finalmente el suscriptor MQTT es el recibe la información correspondiente a la

disponibilidad de las plazas de parqueo desde el servidor, esta información se transmite a través de Internet para visualizarla en la aplicación móvil Telegram a través de la interacción en un chat Bot, adicionalmente dicha información se visualiza en un letrero LED que se encuentra conectado a la raspberry pi.

3.2. Planteamiento inicial

Los sensores ultrasónicos serán colocados en el borde de la acera estratégicamente de tal manera que la señal emitida será reflejada correctamente para la detección del vehículo, luego las señales adquiridas por estos sensores son procesadas por el módulo ESP32 el mismo que cuenta con wifi integrado para conectar a la red local del Gateway y poder realizar el intercambio de datos con el servidor MQTT, luego el servidor MQTT también se conectará a la red local para adquirir los datos de los sensores y enviarlos a través de internet a dispositivos remotos y poder visualizarlos mediante una aplicación móvil.

3.3. Diagramas de funcionamiento

Los sensores ultrasónicos serán colocados en el borde de la acera estratégicamente de tal manera que la señal emitida será reflejada correctamente para la detección del vehículo, luego las señales adquiridas por estos sensores son procesadas por el módulo ESP32 el mismo que cuenta con wifi integrado para conectar a la red local del Gateway y poder realizar el intercambio de datos con el servidor MQTT, luego el servidor MQTT también se conectará a la red local para adquirir los datos de los sensores y enviarlos a través de internet a dispositivos remotos y poder visualizarlos mediante una aplicación móvil.

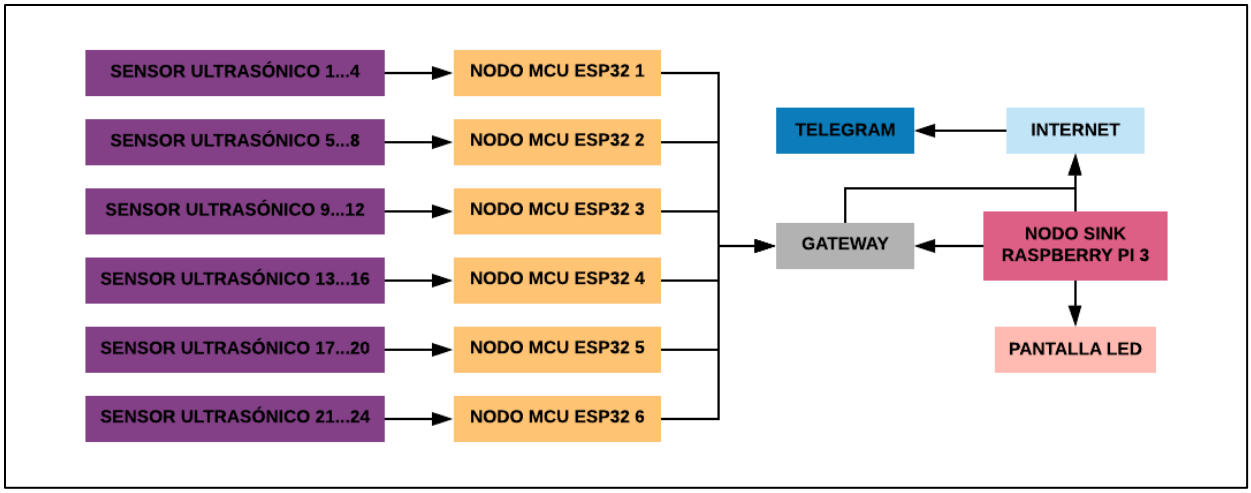


Figura 3.4. Diagrama de funcionamiento del sistema de sensores.

La figura 3.4 muestra el funcionamiento general del sistema de parqueadero inteligente, donde se conecta 4 sensores ultrasónicos por cada MCU Esp32, estos nodos a su vez se conectan al Gateway a través de Wifi usando la banda de 2.4 GHz, de la misma manera el nodo sink se conecta al Gateway mediante un cable Ethernet para de esta forma adquirir los datos de los nodos utilizando el protocolo de comunicación MQTT. Por último, una vez procesados los datos de los nodos en el nodo sink se muestra la información de los estacionamientos disponibles en una pantalla led además el nodo sink se conecta a la red de internet para presentar dicha información por la aplicación celular Telegram.

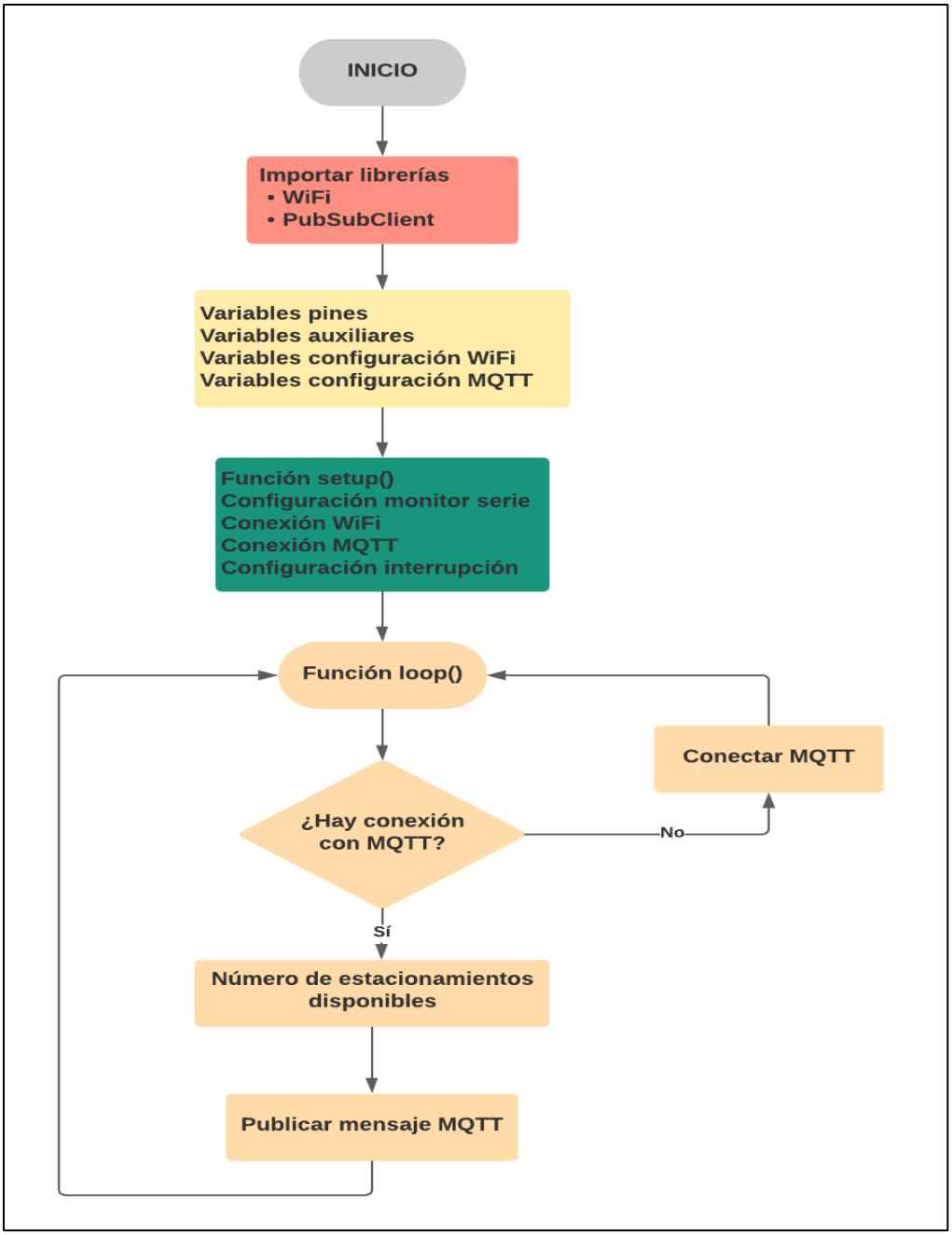


Figura 3.5. Diagrama de funcionamiento del node MCU Esp32.

La figura 3.5 describe el funcionamiento del nodo sensor Esp32, este nodo microcontrolador utiliza la librería Wi-Fi para conectarse inalámbricamente a red local wifi que proporcionada por el Gateway, además se usa la librería PubSubClient la misma que proporciona un cliente para realizar mensajes simples de publicación / suscripción con servidor MQTT

implementado en la raspberry pi, una vez declarado las bibliotecas y variables a utilizarse se configuran las funciones utilizadas para la conexión Wi-Fi, MQTT y para el funcionamiento del sensor ultrasónico. A continuación, se define dentro de la función Loop un bucle que verifica si existe conexión MQTT se obtiene el resultado de los estacionamientos disponibles en ese nodo y se publica a través de un mensaje MQTT, caso contrario al no existir conexión el microcontrolador vuelve a ejecutar la función Loop hasta que se logre establecer correctamente la comunicación MQTT.

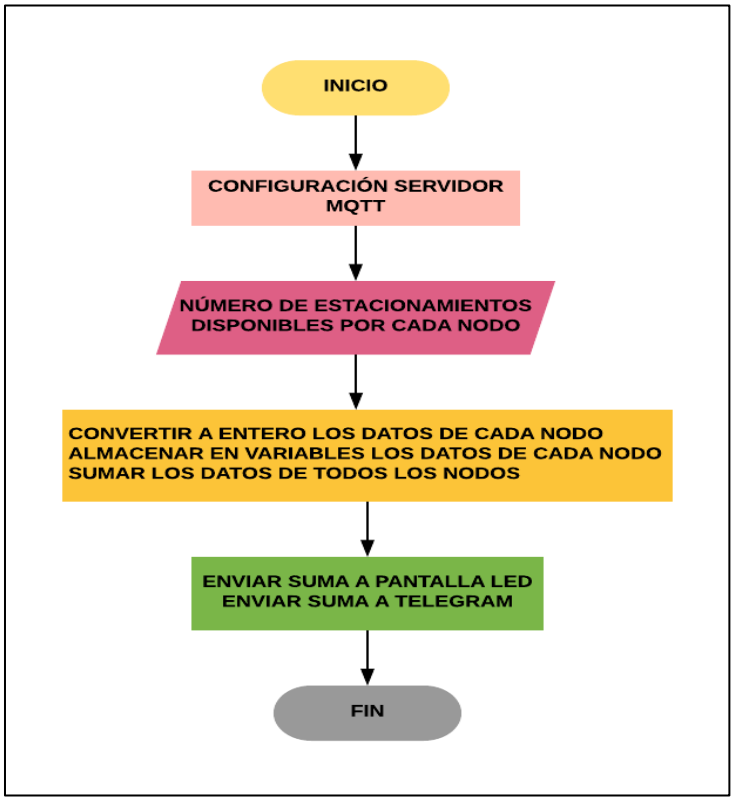


Figura 3.6. Diagrama de funcionamiento del nodo sink (raspberry pi 3 model B).

En la figura 3.6 se detalla el proceso que realiza el nodo sink para lo cual se utiliza una raspberry pi donde se aloja el servidor MQTT. Este nodo utiliza un motor de flujos con enfoque IoT llamado node RED, donde en primer lugar se configuran los parámetros del servidor MQTT tales como, la dirección IP, el puerto y el topic. Luego, se recibe el dato del número de

estacionamientos disponibles de cada nodo, se lo convierte a un número entero en caso de no serlo, después el dato convertido se lo almacena en una variable respectivamente para finalmente sumar el dato de cada nodo y obtener el número de estacionamientos disponibles total conformado por todos los nodos. Por último, la suma total se lo muestra en una pantalla LED o se envía el dato a Telegram siempre y cuando el usuario haya iniciado un chat con el bot del parqueadero.

CAPÍTULO 4

4. Resultados y análisis

Para comprobar la eficiencia y efectividad del sistema de red de sensores se van a utilizar dos distintos escenarios: el primero una maqueta en la cual se van a simular pruebas con 24 sensores HCSR04, el segundo escenario es un parqueadero real ubicado en la FIEC, en el cual solamente se usará un sensor y se realizará las mismas pruebas que en el primer escenario.

Las pruebas para realizarse son importantes ya que gracias a ellas se podrá saber el tiempo de actualización de los datos en pantalla considerando diferentes situaciones tales como clima, distancia y velocidad de conexión al punto de red. Todas las pruebas para realizarse se harán con el objetivo de demostrar que la red de sensores y su implementación son viables para todos los usuarios.

4.1. Implementación en maqueta

Inicialmente el escenario elegido para la implementación del proyecto era el parqueadero ubicado en las instalaciones de la Facultad de Electricidad y Computación, pero por la situación actual de la pandemia Covid-19 no fue posible obtener los permisos necesarios para la ejecución del proyecto. Por tal razón se optó por ejecutar el proyecto a mínima escala haciendo uso de una maqueta en la cual se implementó de manera correcta el sistema con todas las funciones mencionadas a lo largo del desarrollo del presente documento.

A continuación, se detalla cada uno de los procesos realizados para la implementación de la red de sensores, para finalmente obtener como resultado el objetivo del proyecto.

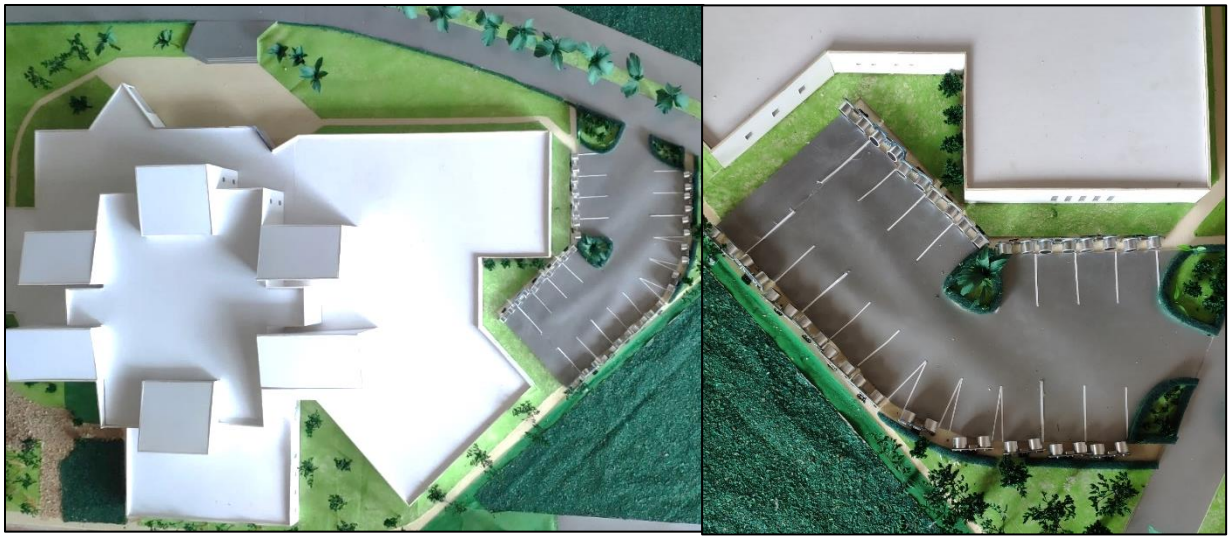


Figura 4.1. Maqueta diseñada para el escenario de implementación.

En la figura 4.1 se observa la maqueta construida a escala de la facultad FIEC, esta posea una réplica del estacionamiento usado para realizar las pruebas. Cada estacionamiento posee un sensor HCSR04 el cual está encargado de detectar la presencia o ausencia de un obstáculo, en este caso el vehículo. En la parte interior de la maqueta está el cableado de los sensores, los cuales conectan con la placa principal que contiene cada uno de los módulos ESP32, estos módulos por medio del protocolo MQTT se comunica inalámbricamente a la raspberry pi 3 modelo b y esta a su vez con el router que es el responsable de la actualización de datos en pantalla y en la aplicación de Telegram.

4.2. Programación de nodos en los módulos ESP32

Node Red es la interfaz de programación visual que ayudó al desarrollo del código de programación de cada uno de los nodos a los cuales van conectados los sensores hcsr04, el total de módulos esp32 son cinco, y a cada módulo está conectado 4 sensores ultrasónicos.

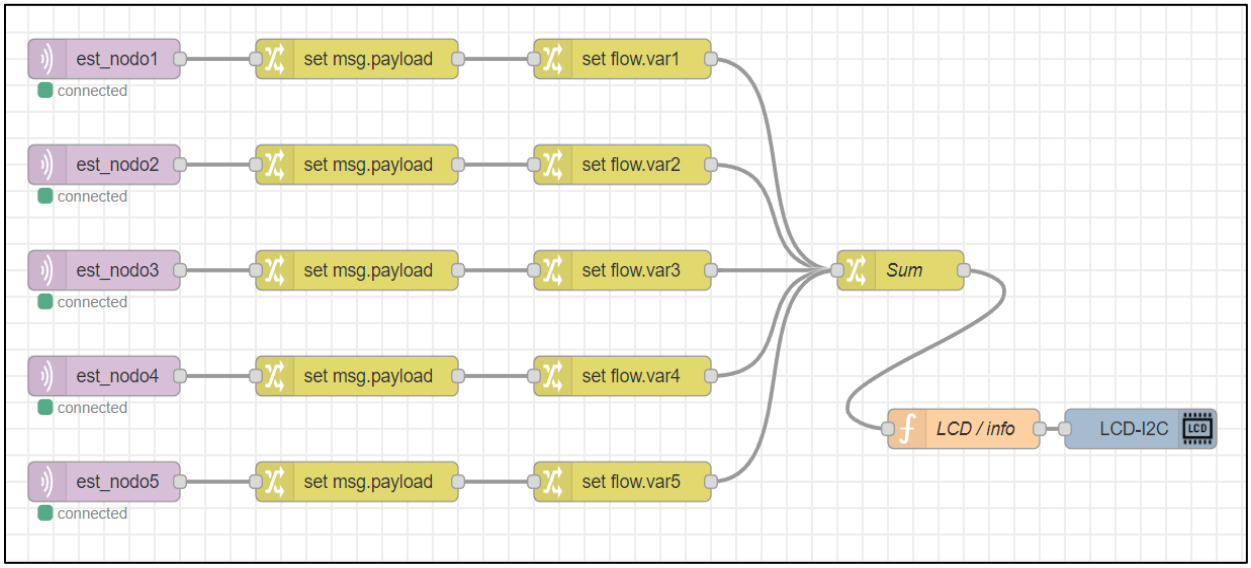


Figura 4.2. Programación en node RED recolección de datos y envío a la pantalla LCD.

En la figura 4.2 podemos observar los nodos (est_nodo1, est_nodo2…, est_nodo5) conectados con el tópico especificado con un nombre, de tal manera que este pueda ser reconocido. Posteriormente a través de la variable de configuración se trae la información recolectada por cada uno de los nodos a través de los módulos y sensores hcsr04; esta variable obtenida por los elementos mencionados anteriormente se convierte a un número entero, cada uno de los nodos (5 en total), para finalmente realizar el conteo total de estacionamientos disponibles a través del nodo Sum.

Figura 4.3. Pantalla LCD conectado a raspberry pi mediante el protocolo I2C.

En la figura 4.3 se observa el resultado de lugares disponibles, este resultado se lo obtiene con la función suma, la cual acumula todas las variables y se muestra como resultado un conteo de todos los estacionamientos disponibles. Finalmente, a través del protocolo I2C el resultado como número entero que varía entre 0 y 20 es transmitido a la pantalla LCD, esta pantalla fue programada con leguaje basado en Java script y está conectada a la Raspberry.

4.3. Programación del Bot de Telegram

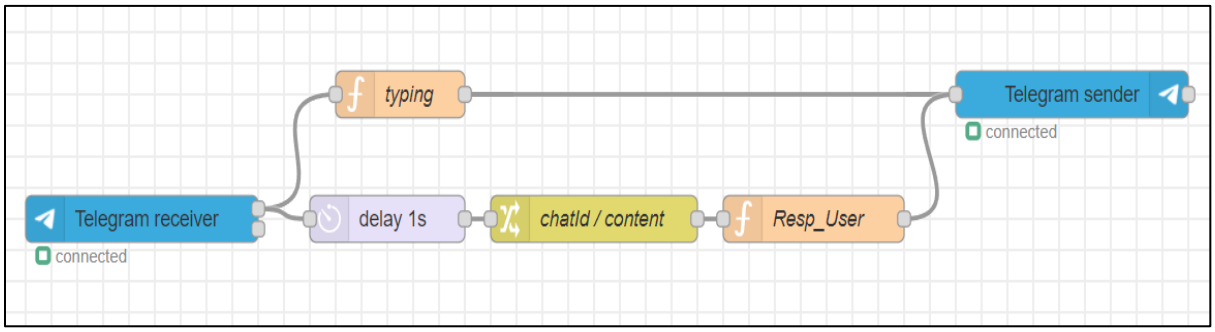


Figura 4.4. Programación del bot en node RED para su uso con la app móvil Telegram.

En la figura 4.4 se puede observar los nodos que forman parte de la programación visual del Bot de Telegram, el primer nodo es Telegram receiver, el cual recibe una de las entradas enviadas por el usuario en el chat Bot de la aplicación, el código contiene los parámetros del uso de app, creadores y el número de disponibilidad de lugares de parqueo. A continuación, se muestra la interfaz del chat Bot denominado Parqueadero FIEC, el cual va a permitir al usuario saber en tiempo real el número de parqueos y de esta manera evitar visitar ese parqueadero en el caso de que esté presente cero lugares disponibles.

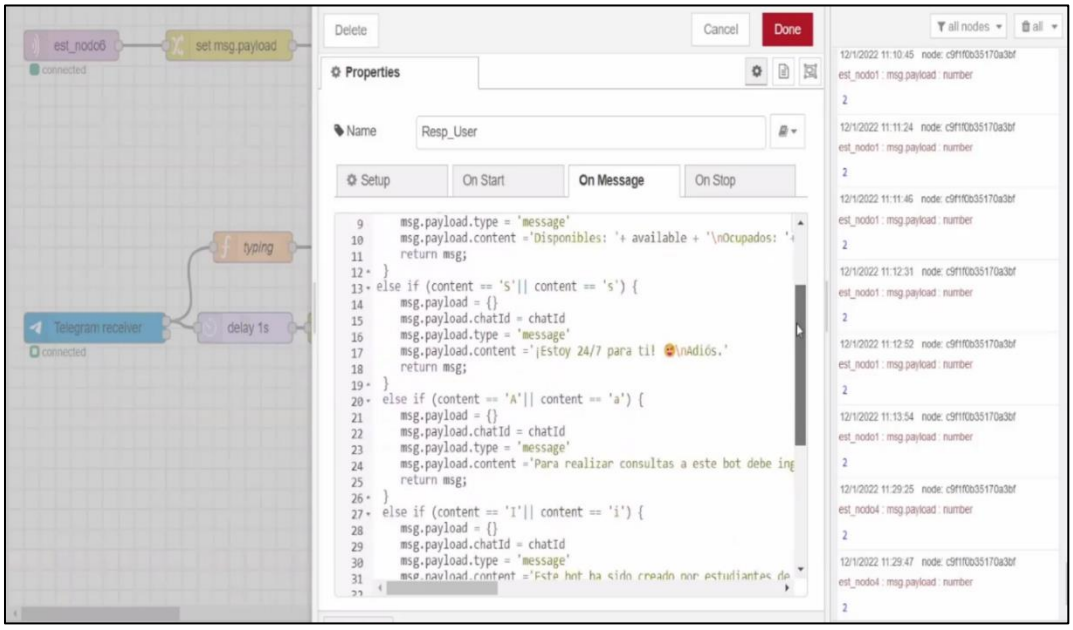


Figura 4.5. Vista -Código de configuración de parámetros Chat Bot Telegram.

En la figura 4.5 se observan cada uno de los nodos envían información de tal manera que el Bot proporciona en tiempo real la disponibilidad, cabe recalcar que, sobre la maqueta, escenario el cual fue elegido para colocar los sensores, se realizaron varias pruebas en las cuales el tiempo de actualización del número entero en la pantalla es de 2 segundos luego de haber colocado o quitado el obstáculo, el cual representa el coche en cada estacionamiento.

Es importante mencionar que la conectividad a la red no influye mayormente en el funcionamiento del sistema, es decir la actualización en el gateway sigue siendo eficiente con una conexión “lenta”, esto debido a que los elementos encargados de la lectura e interpretación de datos requieren un ancho de banda mínimo.

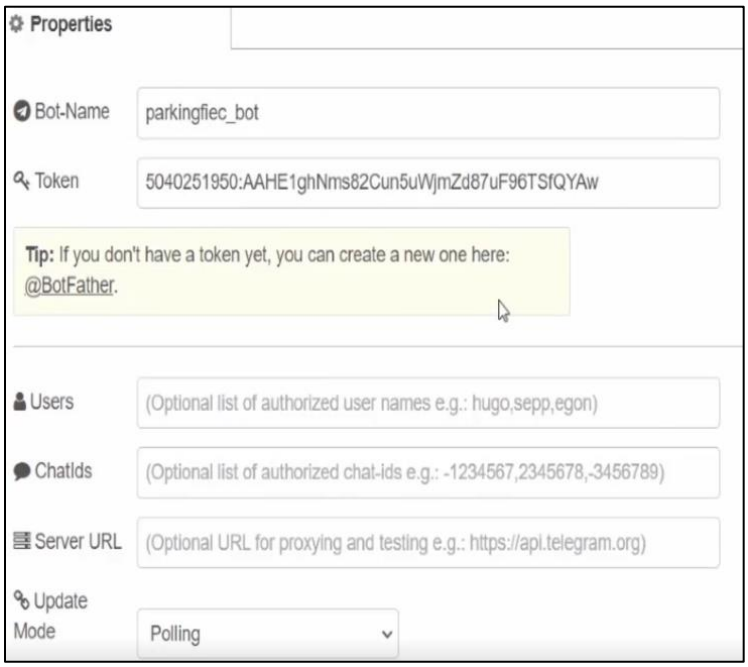


Figura 4.6. Propiedades y parámetros del Bot Telegram.

En la figura 4.6 se puede observar las propiedades usadas para la creación del Bot en la aplicación de Telegram. El nombre del chat interactivo disponible para todo usuario que cuente con la aplicación es “ParkingFIEC”, el cual tiene una respuesta inmediata e interactúa con el usuario dando a conocer información del número de paqueos disponibles en tiempo real. Con este Bot, el usuario va a tener la posibilidad de saber si existe una plaza de estacionamiento disponible, mucho antes de llegar al parqueadero.

4.4. Protoboard con módulo ESP32

En la figura 4.7 se puede verificar las conexiones realizadas sobre la protoboard, existen 5 módulos ESP32, los cuales son los encargados de realizar el control de las señales recibidas por parte de cada uno de los sensores. Cabe recalcar que los módulos ESP32 trabajan de manera individual, representando cinco nodos con 4 sensores cada uno, la función en la programación denominada “suma”, es la responsable de enviar el total de estacionamientos disponibles.

Los módulos esp32 están conectados a la raspberry por medio del módulo Wifi. Cada uno de los pines sirven para alimentar los sensores el cual tiene las entradas de voltaje, tierra, echo y trigger, estos dos últimos responsables de enviar una señal por el pin echo en forma de onda y si existe un obstáculo esta rebota y reingresa por el pin trigger.

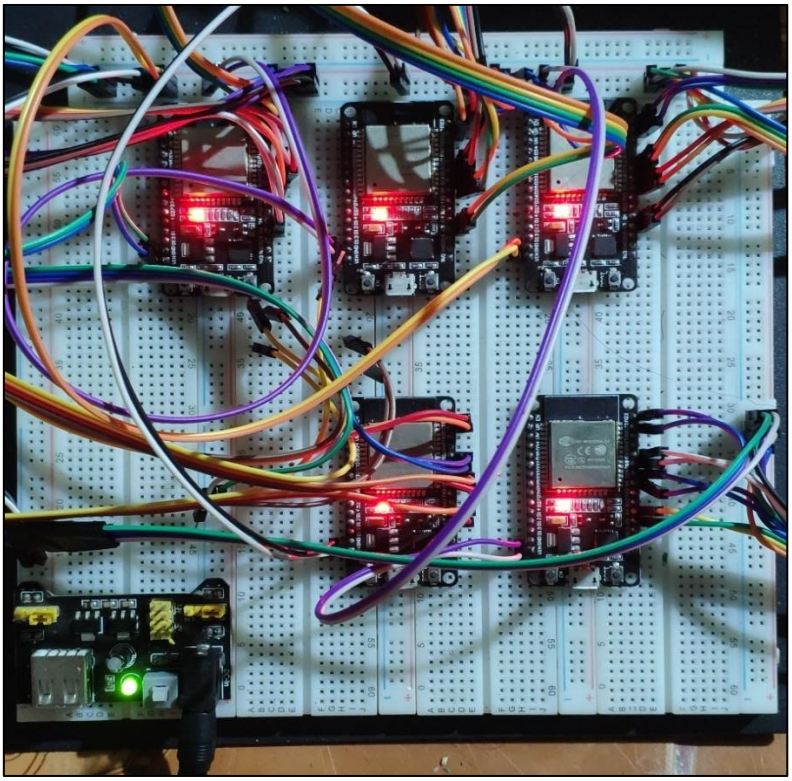


Figura 4.7. Protoboard conectada con módulos ESP32 y sensores HCSR04.

4.5. Pruebas de Campo

El presente proyecto fue aplicado a baja escala en una maqueta en la cual tuvo un correcto funcionamiento, es por eso qué, para comprobar la eficacia y correcto desempeño del sistema de red de sensores, se procedió a realizar pruebas de campo; este proceso se llevó a cabo en uno de los parqueaderos de FIEC ESPOL.

Para esta prueba se realizó la reprogramación completa de un solo nodo formado por un módulo ESP32 (responsable del procesamiento de la señal enviada) y por un único sensor HCSR04 que se encarga de enviar la señal con existencia o ausencia de un vehículo en el espacio de estacionamiento.

Figura 4.8. Pruebas realizadas en el parqueadero de la FIEC con presencia de un automotor.

De la misma manera que se realizó en las pruebas a baja escala, se procedió a verificar si el único estacionamiento usado se encuentra o no disponible, dicha disponibilidad se representa en forma de número entero procesado por la Raspberry pi 3 modelo b y visualizada en una pequeña

pantalla LED. Las pruebas se realizaron en un ambiente real con una temperatura promedio de 27°C y dichos datos obtenidos, comparados con los datos de las pruebas realizadas en el modelo a escala fueron similares.

Figura 4.9. Pruebas realizadas en el parqueadero de la FIEC con ausencia del automotor.

A continuación, en la tabla 4.1, se puede observar un total de 20 pruebas realizadas al sensor ubicado en la acera del estacionamiento, el mismo fue configurado con un rango de medición de 5 a 100 [cm], es decir podrá detectar la presencia de un objeto siempre y cuando este dentro de este rango. Para esta comparativa se tomó en cuenta el tiempo que tarda en llegar el dato luego de que detecte la presencia o ausencia de un obstáculo, a la raspberry y a la pantalla LED.

También se puede observar un promedio de 1.17 [seg] para el tiempo de respuesta de la Raspberry y una media de respuesta de 2.15[seg] de la pantalla LED, es decir el tiempo que tarda en actualizar el valor numérico. Tal como se puede observar son valores pequeños lo que hace que

la respuesta del sistema sea eficaz ya que se logra en cortos periodos de tiempo medido en segundos.

Tabla 4.1. Tiempo de demora de la actualización de los estacionamientos disponibles en la pantalla.

TIEMPO DE ACTUALIZACIÓN DE DATOS EN PANTALLA				
Distancia de Gateway a nodo [cm]	**T1[s]**	**T2[s]**	**T3[s]**	**PROM[s]**
4	1,47	1,74	1,35	1,520
8	1,85	2,28	2,15	2,093
12	2,13	1,8	1,74	1,890
16	2,38	1,85	2,74	2,323
20	2,31	2,25	2,01	2,190
24	1,91	2	1,81	1,907
28	1,81	1,86	2	1,890
32	2,5	2,27	2,33	2,367
36	2,45	2,52	2,82	2,597
40	1,9	2,6	2,5	2,333
44	2	2,6	2,45	2,350
48	2,1	1,96	2,45	2,170
52	1,8	2,56	2,64	2,333
56	1,9	2,69	2,52	2,370
60	1,79	2,65	2,37	2,270
64	1,85	2,56	1,98	2,130
68	2,14	2,58	1,59	2,103
72	2,2	2,48	2,14	2,273
76	2,35	2,87	2,02	2,413
80	2,26	2,42	2,09	2,257

Seguidamente, en la figura 4.10 se puede observar un cuadro estadístico en el cual están presentes gráficamente los valores de tiempo de respuesta tanto de la raspberry como de actualización de la raspberry. Es evidente que la pendiente de tiempo de respuesta de la Raspberry es más uniforme esto debido a que los tiempos oscilan en un rango más pequeño, es decir entre (0.98-1.35) [seg], mientras que la pendiente representada por el tiempo de actualización del valor numérico que el usuario observa por medio de la pantalla LED es menos uniforme y tiene altos y bajos algo

pronunciados, esto porque el rango de tiempo de respuesta en los cuales oscilan estos valores es (1.19-1.54) [seg].

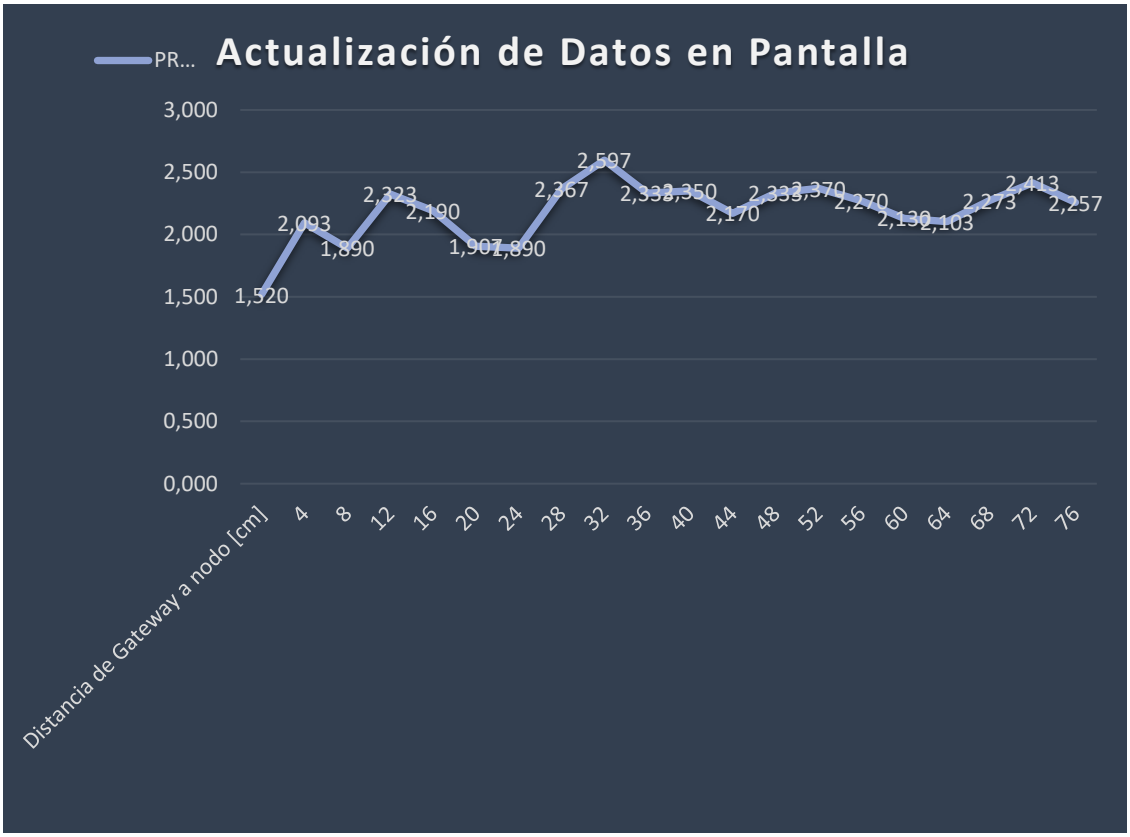


Figura 4.10. Gráfica de tiempos de actualización de datos en pantalla.

CAPÍTULO 5

5. Conclusiones y recomendaciones

En el presente capítulo con el análisis de resultados de las pruebas realizadas, se concluye la efectividad y viabilidad de la implementación del proyecto. Se analizan diferentes factores que hacen que el sistema de red de sensores sea una opción útil para el usuario, en este caso estudiantes, profesores y personal administrativo.

Las conclusiones se las hace en base a las pruebas realizadas en los dos escenarios, es decir la maqueta y el parqueadero real en la FIEC, cabe recalcar que las únicas diferencias son las condiciones ambientales de cada uno de los ambientes y el número de sensores implementados: 24 para la maqueta y 1 para el estacionamiento en FIEC.

Las recomendaciones se dan en base a la experiencia adquirida durante la implementación, tomando errores de programación, cambio de arquitectura en la red y sus nodos, hasta factores del escenario que pueden influir en el funcionamiento de los sensores.

5.1. Conclusiones

Hoy en día, el gran número de estudiantes y profesores que acuden a la universidad de manera presencial y poseen un vehículo, necesitan de un lugar de estacionamiento. Muchas de las veces estos se encuentran a su capacidad máxima lo que causa malestar en los usuarios, es por esto qué se implementó un sistema inteligente sobre una maqueta, el cual está formado por sensores HCSR04, raspberry pi 3 modelo b, módulo ESP32 y una pantalla LED, los cuales en conjunto forman la de red de sensores. La escritura y codificación de cada uno de los elementos del sistema se realizó en lenguaje de programación basado en Java y Node red y protocolo I2C, por lo tanto,

la respuesta de los elementos físicos fue inmediatas, las lecturas en pantalla no mostraban datos erróneos.

La implementación y pruebas realizadas en dos distintitos escenarios del sistema de red de sensores, permitió saber con exactitud el número de lugares de estacionamiento, sin importar la distancia a la cual se encontraba el sensor y el enrutador, por lo tanto, se logró que el usuario ahorre tiempo al saber la disponibilidad de lugares de estacionamiento, ya que los datos en pantalla se actualizaban en un promedio de 2.59 [seg].

El Bot fue creado en la plataforma Telegram, el sistema requiere de un ancho de banda mínimo de aproximadamente 4 kbps para su funcionamiento y actualización de datos en la aplicación, por lo tanto, cada uno de los usuarios que dispongan de un ancho de banda igual o superior al mencionado, podrán consultar correctamente la información sobre el número de lugares disponibles en la pantalla LCD y a través del chat automático de Telegram sin ningún problema.

El sistema de red de sensores implementado ayuda al ahorro de energía. Dado que, los componentes de los nodos sensores que van ubicados en los estacionamientos tienen un consumo total de 35 mA pueden ser alimentado por un banco 10 baterías de 9 V y 12000 mAh conectadas en paralelo para una duración de 142.86 días. Por otro lado, es necesario alimentar la Raspberry Pi 3 y el enrutador Tp-Link mediante la línea eléctrica, ya que estos se deben encontrar específicamente dentro de un lugar con acceso a la red de electricidad.

Con este sistema el usuario evitará ingresar innecesariamente al parqueadero si no existen espacios de parqueo disponible, puesto que la información se visualizará en la pantalla LCD y podrá ser consultada a través de Telegram de manera rápida y oportuna, por lo que se presupone se traduzca en un ahorro de tiempo y dinero para el usuario.

5.2. Recomendaciones

La implementación al ser realizada sobre un entorno que no presenta condiciones climáticas hostiles no requirió de elementos que soporten en ambientes exteriores, por lo que si se implementa en un parqueadero real se recomienda cajas protectoras con resistencia IP68, esto para poder evitar el ingreso de agua que puede afectar o incluso dañar el sensor y por ende el resto del sistema.

Usar como entorno de programación Node Red fue de gran utilidad, ya que esta interfaz acorta varias líneas de codificación, además este entorno permite modificar de manera sencilla los procesos previamente establecidos, es decir aumentar o disminuir el número de sensores por nodo dependiendo de las necesidades del usuario.

Para la implementación en un escenario real es necesario usar un router de mayor alcance de cobertura, al igual disponer una conexión a internet estable e IP estáticas, ya que si se posee IP dinámica esta cambia y por ende se debe realizar el redireccionamiento de direcciones IP, cada vez que se reinicia el sistema.

BIBLIOGRAFÍA

[1] B. Ki-moon, "cepal.org," 18 Mayo 2010. [Online]. Available: https://www.cepal.org/es/articulos/2010-dia-mundial-telecomunicaciones-la-sociedad-la-informacion. [Accessed 3 Noviembre 2021].

[2] ISP GRUP, "ispgrup.cat," 18 Agosto 2021. [Online]. Available: https://www.ispgrup.cat/la-telecomunicacion-mundo-beneficios/#:~:text=La%20telecomunicaci%C3%B3n%20permite%20establecer%20la%20comunicaci%C3%B3n%20a%20distancia.&text=La%20telecomunicaci%C3%B3n%20contribuye%20al%20desarrollo,vital%20para%20cualquier%20%C3. [Accessed 3 Noviembre 2021].

[3] Unión Internacional de Telecomunicaciones, "Unión Internacional de Telecomuniaciones UIT," Febrero 2008. [Online]. Available: https://www.itu.int/itunews/manager/display.asp?lang=es&year=2008&issue=08&ipage=24&ext=html. [Accessed 30 Enero 2022].

[4] L. Llamas, "luisllamas.es," 16 Junio 2015. [Online]. Available: https://www.luisllamas.es/medir-distancia-con-arduino-y-sensor-de-ultrasonidos-hc-sr04/. [Accessed 30 Enero 2022].

[5] J. Beningo, "Digi-Key Electronics," 21 Enero 2020. [Online]. Available: https://www.digikey.com/es/articles/how-to-select-and-use-the-right-esp32-wi-fi-bluetooth-module. [Accessed 16 Noviembre 2021].

[6] HETPRO, "Herramientas Tecnológicas Profesionales," 10 Abril 2018. [Online]. Available: https://hetpro-store.com/TUTORIALES/senal-digital/. [Accessed 2021 Noviembre 2021].

[7] L. Alegsa, "Alegsa.com.ar," 31 Mayo 2018. [Online]. Available: https://www.alegsa.com.ar/Dic/se%C3%B1al_de_reloj.php. [Accessed 16 Noviembre 2021].

[8] HETPRO, "Herrameintas Tecnologías Profesionales," 22 Febrero 2018. [Online]. Available: https://hetpro-store.com/TUTORIALES/i2c/. [Accessed 16 Noviembre 2021].

[9] Un Electrónica, "unelectronica.github.io," 10 Noviembre 2019. [Online]. Available: https://unelectronica.github.io/I2C_esp8266/. [Accessed 16 Noviembre 2021].

[10] L. LLamas, "luisllamas.es," 17 Abril 2019. [Online]. Available: https://www.luisllamas.es/que-es-mqtt-su-importancia-como-protocolo-iot/. [Accessed 22 Diciembre 2021].

[11] Websa100, "SEoptimer," [Online]. Available: https://www.seoptimer.com/es/blog/notificaciones-push-que-son-por-que-usarlas/. [Accessed 22 Diciembre 2021].

[12] L. Llamas, "luisllamas.es," 23 Noviembre 2019. [Online]. Available: https://www.luisllamas.es/principales-broker-mqtt-open-source-para-proyectos-iot/. [Accessed 2022 Enero 4].

[13] broker.emqx.io, "emqx.com," 4 Octubre 2013. [Online]. Available: https://www.emqx.com/en/mqtt/public-mqtt5-broker. [Accessed 4 Enero 2022].

[14] E. Limones, "Open Webinars," 7 Abril 2021. [Online]. Available: https://openwebinars.net/blog/topologia-de-redes-informaticas/. [Accessed 16 Noviembre 2021].

[15] M. Rouse, "Computer Weekly.es," Abril 2021. [Online]. Available: https://www.computerweekly.com/es/definicion/Topologia-de-red. [Accessed 16 Noviembre 2021].

[16] L. Nungaray, "ALEPH," 7 Abril 2021. [Online]. Available: https://aleph.org.mx/como-funciona-la-topologia-de-bus. [Accessed 16 Noviembre 2021].

[17] P. Rodriguez, "xakata smart home," 22 Abril 2021. [Online]. Available: https://www.xatakahome.com/la-red-local/que-red-wifi-mejor-usar-cada-caso-2-4-ghz-vs-5-ghz. [Accessed 4 Enero 2022].

[18] R. Andrés, "Computer Hoy," 18 Agosto 2019. [Online]. Available: https://computerhoy.com/reportajes/tecnologia/wifi-24ghz-vs-5ghz-diferencia-velocidad-cuando-debes-elegir-cada-475141. [Accessed 4 Enero 2022].

[19] Aprendiendo Arduino, "aprendiendoarduino.com," 19 Abril 2021. [Online]. Available: https://www.aprendiendoarduino.com/cursos/node-red-developer-2021-nivel-i/. [Accessed 4 Enero 2022].

[20] P. Sancho, "Techedge," 20 Abril 2020. [Online]. Available: https://www.techedgegroup.com/es/blog/fundamenos-node-red. [Accessed 4 Enero 2022].

[21] SIEMON, "siemon.com," 3 Octubre 2013. [Online]. Available: https://www.siemon.com/es/home/support/education/white-papers/03-10-13-ethernet-ip#:~:text=Ethernet%2FIP%20es%20un%20protocolo,controlar%20dispositivos%20de%20automatizaci%C3%B3n%20industrial.. [Accessed 4 Enero 2022].

[22] I. Ramírez, "xakatamovil," 1 Junio 2021. [Online]. Available: https://www.xatakamovil.com/listas/bots-telegram-como-encontrarlos-23-opciones-recomendadas. [Accessed 4 Enero 2022].

[23] E. Rodriguez De Luis, "xakata.com," 15 Julio 2018. [Online]. Available: https://www.xataka.com/makers/cero-maker-todo-necesario-para-empezar-raspberry-pi. [Accessed 5 Enero 2022].

[24] R. Solé, "Profesional Review," 18 Juliio 2021. [Online]. Available: https://www.profesionalreview.com/2021/07/18/que-es-raspberry-pi/. [Accessed 5 Enero 2022].

[25] Y. Fernández, "xakata.com," 18 Marzo 2018. [Online]. Available: https://www.xataka.com/basics/wifi-2-4g-y-5g-cuales-son-las-diferencias-y-cual-elegir. [Accessed 5 Enero 2022].

[26] Geek Factory, "geekfactory.mx," 16 Mayo 2014. [Online]. Available: https://www.geekfactory.mx/tutoriales-arduino/hc-sr04-con-arduino-sensor-de-distancia-ultrasonico/. [Accessed 5 Enero 2022].

[27] KESYESTUDIO, "alliexpres.com," 14 Julio 2021. [Online]. Available: https://es.aliexpress.com/item/4000203575061.html?spm=a2g0o.search0304.0.0.4248496bk5gY70&algo_pvid=a6107729-90af-47e1-b9ff-94e9a8252d56&algo_exp_id=a6107729-90af-47e1-b9ff-94e9a8252d56-8. [Accessed 5 Enero 2022].

[28] "Comisión de la Banda Ancha de las Naciones Unidas fija objetivos mundiales para poner en línea a 3800 millones de habitantes desconectados," ITU // La Comisión de la Banda Ancha para el Desarrollo Sostenible lanza metas de 2025 para "Conectar la otra mitad", 23 Enero 2018. [Online]. Available: https://www.itu.int/es/mediacentre/Pages/2018-PR01.aspx. [Accessed Junio 2021].

ANEXOS

Anexo 1. Código General para nodos MCU ESP32.

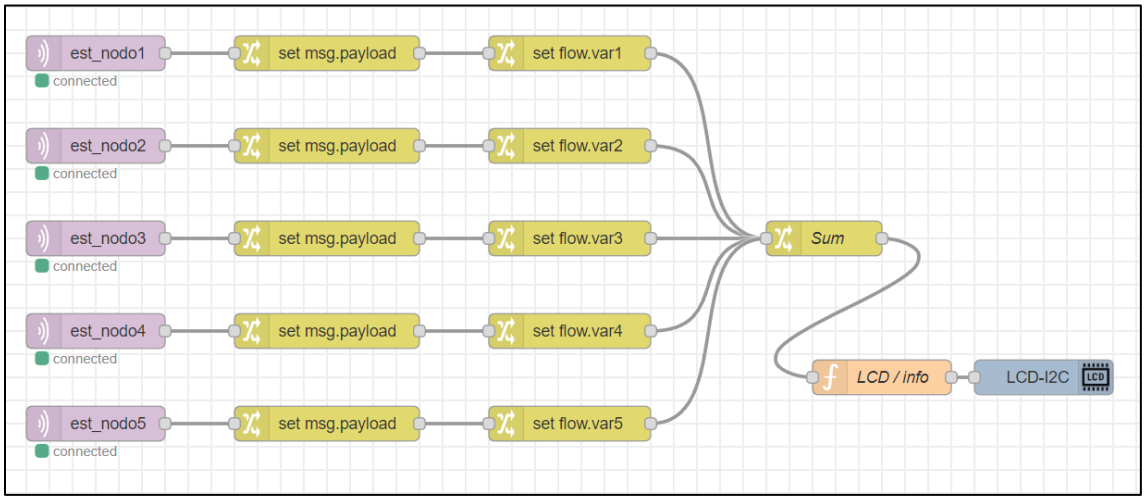
```
// Cargamos las librerias
#include <WiFi.h> //Conecta el ESP32 a redes Wi-Fi
#include <PubSubClient.h> //Conecta el ESP32 al servidor MQTT

// WiFi
const  char * ssid = "Gateway" ; // Ingrese su nombre de WiFi
const  char * password = "FiecEspol" ;  // Ingrese la contraseña de WiFi

// MQTT Broker
const char* mqtt_server = "192.168.0.100"; // IP del broker MQTT

// Pines sensor ultrasónico
int const TRIG1 = 2;
int const ECHO1 = 4;
int const TRIG2 = 5;
int const ECHO2 = 18;
int const TRIG3 = 19;
int const ECHO3 = 21;
int const TRIG4 = 22;
int const ECHO4 = 23;

// Variables de cálculo
long duracion;
float distancia;
float distancia1;
float distancia2;
float distancia3;
float distancia4;
//Estacionamientos
int est1;
int est2;
int est3;
int est4;
int ocupados;
float disponibles;

//Inicializamos los objetos el cliente y el publicador
WiFiClient espClient; //Crea un cliente que se puede conectar a una determinada dirección
IP
PubSubClient client(espClient); //Crea un cliente para publicar y suscribirse a través de un
//determinado cliente WiFi
```

```
long lastMsg = 0;
char msg[50];
int value = 0;
//  FUNCIONES
//Creamos la función que nos permitira realizar
//la conexión del wifi
void setup_wifi() {
   delay(10);
   Serial.println();
   Serial.print("Conectado a: ");
   Serial.println(ssid);
   WiFi.begin(ssid, password);
   while (WiFi.status() != WL_CONNECTED) {
   delay(500);
   Serial.print(".");
   }
   randomSeed(micros());
   Serial.println("");
   Serial.println("WiFi conectado");
   Serial.println("Direccion IP: ");
   Serial.println(WiFi.localIP());
}
//Creamos la función que permite realizar la conexión
//al servidor mosquitto
void reconnect() {
   //Validamos si existe una conexión al servidor,
   //de lo contrario intentamos conectarlo de nuevo
   while (!client.connected()) {
      Serial.print("Intentando conexión MQTT...");
      //Creamos un cliente ID ramdon
      String clientId = "ESP32Client-";
      clientId += String(random(0xffff), HEX);
      //Intentamos conectarnos
      if (client.connect(clientId.c_str())) {
         Serial.println("Conectado");
         //Enviamos un mensaje de prueba
         client.publish("outTopic", "hello world");
      } else {
         //Si la conexión fallo, intentamos de nuevo
         Serial.print("conexión fallida, rc=");
         Serial.print(client.state());
         Serial.println(" Se intentará de nuevo en unos segundos...");
         // Espera 1 segundo para volver a intentarlo
         delay(1000);
      }
   }
}
```

```
//Funciòn que lee la distancia del sensor
long lectura(int TRIG, int ECHO){
  /* Hacer el disparo */
  digitalWrite(TRIG, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG, HIGH);  // Flanco ascendente
  delayMicroseconds(10);       // Duracion del pulso
  digitalWrite(TRIG, LOW);  // Flanco descendente
  /* Recepcion del eco de respuesta */
  duracion = pulseIn(ECHO, HIGH);

  /* Calculo de la distancia efectiva */
  distancia = (duracion/2) / 29;
  return distancia;
}

void setup() {
   //Activación de pines como salida: para el pulso ultrasónico
   pinMode(TRIG1, OUTPUT);
   pinMode(TRIG2, OUTPUT);
   pinMode(TRIG3, OUTPUT);
   pinMode(TRIG4, OUTPUT);
   //Activación de pines como entrada: tiempo del rebote del ultrasonido
   pinMode(ECHO1, INPUT);
   pinMode(ECHO2, INPUT);
   pinMode(ECHO3, INPUT);
   pinMode(ECHO4, INPUT);
   Serial.begin(115200);
   Serial.println("setup");
   //Iniciamos la conexión del wifi
   setup_wifi();
   //Indicamos a que servidor nos conectamos
   client.setServer(mqtt_server, 1883);
}

void loop() {
   //Validamos la conexión si esta correcta
   if (!client.connected()) {
      reconnect();
   }
   //Dejamos el cliente el loop para que no cierre
   //la conexión actual
   client.loop();
```

```
  distancia1 = lectura(TRIG1, ECHO1);
  distancia2 = lectura(TRIG2, ECHO2);
  distancia3 = lectura(TRIG3, ECHO3);
  distancia4 = lectura(TRIG4, ECHO4);

  Serial.print("SENSOR1: ");
  Serial.println(distancia1);
  Serial.print("SENSOR2: ");
  Serial.println(distancia2);
  Serial.print("SENSOR3: ");
  Serial.println(distancia3);
  Serial.print("SENSOR4: ");
  Serial.println(distancia4);

  if(distancia1>=3 && distancia1<=7){ //Rango entre 3 y 7 cm
  est1=1; //Ocupado
  }else{
   est1=0; //Libre
   }

  if(distancia2>=3 && distancia2<=7){ //Rango entre 3 y 7 cm
  est2=1; //Ocupado
  }else{
   est2=0; //Libre
   }

  if(distancia3>=3 && distancia3<=7){ //Rango entre 3 y 7 cm
  est3=1; //Ocupado
  }else{
   est3=0; //Libre
   }

  if(distancia4>=3 && distancia4<=7){ //Rango entre 3 y 7 cm
  est4=1; //Ocupado
  }else{
   est4=0; //Libre
   }
  ocupados=est1+est2+est3+est4;
  disponibles=4-ocupados;

  snprintf (msg, 50, "%4.2f", disponibles);
  Serial.print("ESTACIONAMIENTOS DISPONIBLES EN NODO 1: ");
  Serial.println(msg);
  //Enviamos el mensaje
  client.publish("est_nodo1", msg);
  delay(2000);
}
```

Anexo 2. Programación node RED para recolección de datos y mostrarlos por pantalla LCD.



Anexo 3. Programación node RED para-Telegram.