



**UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE GUAYAQUIL
CARRERA DE INGENIERÍA ELECTRÓNICA**

**DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE
ESTANTERÍAS IOT PARA LA AUTOMATIZACIÓN DE PRECIOS
Y DESCRIPCIONES DE PRODUCTOS COMERCIALES
UTILIZANDO UNA RED ESP-WIFI-MESH Y HARDWARE DE
BAJO COSTO**

Trabajo de titulación previo a la obtención del
Título de Ingeniero Electrónico

**AUTORES: ANTHONY FABIÁN NAVARRETE MORÁN
RAMÓN GONZALO LUCIO SUÁREZ
TUTOR: ING. TEDDY JHENNSE NEGRETE PEÑA MSC.**

Guayaquil – Ecuador

2022

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Nosotros, Anthony Fabián Navarrete Morán con documento de identificación N° 0951324953 y Ramón Gonzalo Lucio Suárez con documento de identificación N° 1311963050; manifestamos que:

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

Guayaquil, 7 de julio de 2022

Atentamente,



Anthony Fabián Navarrete Morán
0951324953



Ramón Gonzalo Lucio Suárez
1311963050

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

Nosotros, Anthony Fabián Navarrete Moran con documento de identificación N° 0951324953 y Ramón Gonzalo Lucio Suárez con documento de identificación N° 1311963050, 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 de investigación: “Diseño e implementación de un prototipo de estanterías IoT para la automatización de precios y descripciones de productos comerciales utilizando una red ESP-WIFI-MESH y hardware de bajo costo”, el cual ha sido desarrollado para optar por el título de: Ingeniero Electrónico, 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.

Guayaquil, 7 de julio de 2022

Atentamente,



Anthony Fabián Navarrete Morán

0951324953



Ramón Gonzalo Lucio Suárez

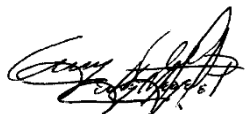
1311963050

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Teddy Jhennse Negrete Peña con documento de identificación N° 0912419611, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE ESTANTERÍAS IOT PARA LA AUTOMATIZACIÓN DE PRECIOS Y DESCRIPCIONES DE PRODUCTOS COMERCIALES UTILIZANDO UNA RED ESP-WIFI-MESH Y HARDWARE DE BAJO COSTO, realizado por Anthony Fabián Navarrete Moran con documento de identificación N°0951324953 y Ramón Gonzalo Lucio Suárez con documento de identificación N°1311963050, obteniendo como resultado final el trabajo de titulación bajo la opción tesis que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Guayaquil, 7 de julio de 2022

Atentamente,



Ing. Teddy Jhennse Negrete Peña, MSc

0912419611

AGRADECIMIENTO

Como primer punto estoy agradecido con Dios porque gracias al estoy donde estoy, y a si mismo a mis padres los cuales me ayudaron no el 100% en la parte económica, pero si en el apoyo de cumplir mis metas.

Recuerdo que cuando comencé la universidad solo tenía 17 años, y para poder cubrir los gastos de la institución trabajaba de oficial de un tío albañil ganándome 50 dólares semanales los cuales pagaba a una vecina quien me prestaba cada mes el valor de la mensualidad de la universidad y así pude estar dos semestres.

Luego entre a trabajar a una empresa en el área de sistemas, teniendo dos semanas laborando tuve un accidente cerca de la universidad, estuve a punto de dejar un semestre, pero gracias a los profesores y a mis compañeros quienes ayudaron en la parte económica para poder seguir estudiando junto con la universidad haciendo recolectas de dinero en cada curso pude seguir con mis estudios.

Y así estoy muy agradecido con todas las personas que fueron parte de este proceso y este logro que se viene va para Dios mis padres amigos y para mí.

ANTHONY FABIÁN NAVARRETE MORAN

AGRADECIMIENTO

El agradecer es una de las virtudes que como seres humanos jamás deberíamos olvidar; ante ello, siempre a Dios por el regalo de la vida, a mis familiares que, aunque físicamente ya no están conmigo me siguen guiando por el mejor camino y por todo lo que me enseñaron mientras estuvieron a mi lado, con una mención muy emotiva a mi papá.

A mi mamá, hermanas y dos sobrinos que han sido un motivo especial para continuar por este camino, que me han permitido perseverar a pesar de las dificultades y romper los límites mentales de lo que parece imposible.

A mi esposa y nuestro hijo por cada día ser el motor para seguir con un crecimiento exponencial y razón para cada amanecer siempre verlo con una sonrisa, esperanza y optimismo a la vida.

A nuestro tutor Teddy Negrete Peña Msc. por sus conocimientos y guía compartidas en nuestro proyecto final como estudiantes de pregrado.

Mi agradecimiento a la Universidad Politécnica Salesiana y a sus docentes por sus deseos de impartir sus conocimientos y por ser forjadores del lado humano de nuevos profesionales para nuestro país.

Finalmente, a mi compañero de tesis, que gracias a nuestro trabajo en equipo hemos conseguido obtener nuestro resultado final y plasmarlo en este texto.

RAMÓN GONZALO LUCIO SUÁREZ

RESUMEN DEL PROYECTO

Año	Alumnos	Tutor de Proyecto de titulación	Proyecto de titulación
2022	Anthony Fabián Navarrete Morán y Ramón Gonzalo Lucio Suárez.	Ing. Teddy Negrete MSc.	Diseño e implementación de un prototipo de estanterías IoT para la automatización de precios y descripciones de productos comerciales utilizando una red ESP-WIFI-MESH y hardware de bajo costo

Las tecnologías del internet de las cosas aplicada a negocios comerciales grandes o pequeños de cualquier índole es relativamente nueva, ya que no ha habido un despliegue de redes MESH aplicables a dispositivos IoT en el Ecuador.

Es común ver en las perchas de algunas farmacias o comercios grandes en Guayaquil, que los precios se encuentren erróneos o las descripciones equivocadas, esto es debido a que los precios y las descripciones de cada producto se colocan de manera manual ocasionando un aumento de error en la etiqueta de cada producto. De igual manera el etiquetado manual de productos y actualización de precios de perchas llega a generar un costo adicional en los negocios, ya que los empleados dedican algunas horas para realizar esta actividad, generando costo por cada hora por trabajador.

Este trabajo de investigación plantea una solución de diseño e implementación de un prototipo de estanterías IoT para la automatización de precios y descripciones de productos comerciales utilizando una red ESP-WIFI-MESH y hardware de bajo costo. El prototipo se basa en estanterías automatizadas en redes WIFI-MESH-ESP32 para la comunicación de las estanterías, la inteligencia en el etiquetado y la descripción de los productos que se exhiben en esas perchas o stands, mediante hardware de bajo costo como es el ESP32 y pantallas leds que se encargan de mostrar las

descripciones y los precios que son enviadas de forma remota desde un servidor web AWS, el cual administra el inventario con el precio respectivo de cada producto, para la automatización y sincronización de productos, promociones o precios en las perchas.

Como resultado del trabajo, los estudiantes de la carrera de ingeniería en electrónica y de la carrera de telecomunicaciones de la Universidad Politécnica Salesiana se ven beneficiados, puesto que se realiza un manual de práctica donde se detalla cómo realizar el prototipo a escala, y plantea sus conocimientos teóricos del internet de las cosas aplicadas a la ingeniería.

ABSTRACT

Year	Students	Degree Project Tutor	Technical Degree Project
2022	Anthony Fabián Navarrete Morán y Ramón Gonzalo Lucio Suárez.	Ing. Teddy Negrete MSc.	Design and implementation of a prototype of IoT shelving for price automation and commercial product descriptions using an ESP-WIFI-MESH network and low-cost hardware

Internet of Things technologies applied to commercial businesses large or small of any kind is relatively new, since there has been no deployment of MESH networks applicable to IoT devices in Ecuador.

It is common to see in the hangers of some pharmacies or large shops in Guayaquil, that the prices are wrong or the descriptions wrong, this is because the prices and descriptions of each product are placed manually causing an increase in error in the label of each product. In the same way, the manual labeling of products and updating of hanger prices comes to generate an additional cost in business, since employees dedicate a few hours to perform this activity, generating cost for each hour per worker.

This research paper proposes a solution for the design and implementation of an IoT shelf prototype for the automation of prices and descriptions of commercial products using an ESP-WIFI-MESH network and low-cost hardware. The prototype is based on automated shelves in WIFI-MESH-ESP32 networks for the communication of the shelves, the intelligence in the labeling and the description of the products that are exhibited in those hangers or stands, using low-cost hardware such as the ESP32 and LED screens that are responsible for showing the descriptions and prices that are sent remotely from an AWS web server, which manages the inventory with the respective price of each product, for the automation and synchronization of products, promotions or prices on the hangers.

As a result of the work, the students of the electronics engineering career and the telecommunications career of the Salesian Polytechnic University are benefited, since a practice manual is made where it is detailed how to make the prototype at scale, and raises their theoretical knowledge of the internet of things applied to engineering.

ÍNDICE GENERAL

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN	ii
CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA	iii
CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN.....	iv
AGRADECIMIENTO	v
AGRADECIMIENTO	vi
RESUMEN DEL PROYECTO	vii
ABSTRACT	ix
ÍNDICE GENERAL	xi
ÍNDICE DE FIGURAS.....	xiii
INTRODUCCIÓN.....	1
1. El problema.....	2
1.1. Descripción del problema	2
1.2. Antecedentes	3
1.3. Importancia y alcance.....	3
1.4. Delimitación.....	4
1.4.1. Delimitación temporal	4
1.4.2. Delimitación espacial.....	4
1.4.3. Delimitación académica.....	4
1.5. Beneficiarios de la propuesta.....	5
1.6. Propuesta de solución	5
1.7. Innovación e impacto del proyecto.....	6
1.8. Objetivos	6
1.8.1. Objetivo general	6
1.8.2. Objetivos específicos.....	6
2. Fundamentos teóricos.....	8
2.1. ESP32	8
2.1.1. Características de ESP32.....	9
2.1.2. Especificaciones ESP32.....	10
2.2. AWS	11
2.2.1. Ventajas de AWS	12

2.2.2.	Herramientas de AWS	13
2.3.	AWS IoT	15
2.4.	Raspberry Pi	16
2.4.1.	Ventajas Raspberry Pi	17
2.4.2.	Raspberry Pi 4	17
2.4.3.	Especificaciones de Raspberry Pi 4	18
2.5.	WIFI	19
2.6.	Red Mesh	20
2.7.	Red Wifi Mesh	21
2.8.	IoT	22
2.9.	Python	23
3.	Marco metodológico	25
3.1.	Tipo de investigación	25
3.2.	Diseño de investigación	25
3.3.	Enfoque de la investigación	25
3.4.	Metodología de investigación	25
3.5.	Proyectos de investigación vinculados	26
3.5.	Título de la propuesta	27
3.6.	Descripción de la propuesta	27
3.7.	Diseño electrónico del prototipo	28
3.8.	Maqueta de stands	33
3.9.	Programación del prototipo Script de Python	42
3.10.	Código de ESP32	44
3.11.	Comando de ejecución de código	51
3.12.	Configuración de ejecución automática al iniciar Raspberry Pi	51
3.13.	Configuración de AWS IoT	53
4.	Resultados	59
5.	Conclusiones	66
6.	Recomendaciones	67
	Bibliografía	68
	Anexos	70

ÍNDICE DE FIGURAS

Figura 2.1. Modulo ESP32 (PROMETEC, 2018).....	8
Figura 2.2 Diagrama en bloques del ESP32 (Pedro Bertoleti, 2019).....	11
Figura 2.3 Amazon Web Services (DIGITEOS, 2020).....	13
Figura 2.4 AWS IoT	16
Figura 2.5. Raspberry Pi 4 (Raspberry Pi, 2019).....	18
Figura 2.6 Wi-Fi Alliance logo	20
Figura 2.7. Red Mesh (Navas, 2017)	21
Figura 2.8 lot	23
Figura 2.9 Python	24
Figura 3.1 Prototipo de estantería comerciales IoT.....	27
Figura 3.2 Conexiones de ESP32.....	29
Figura 3.3 USB Uart Bridge	29
Figura 3.4 Power Supply.....	30
Figura 3.5 Display.....	30
Figura 3.6 CP1202.....	31
Figura 3.7 MP1584EN	31
Figura 3.8 Conexiones de ESP32.....	32
Figura 3.9 Diseño de placa PCB.....	32
Figura 3.10 Diseño de Pantalla IoT.....	33
Figura 3.11 Instalación de elementos del prototipo IoT	34
Figura 3.12 Stand físico	35
Figura 3.13 Stand IoT	35
Figura 3.14 Instalación de pantallas.....	36
Figura 3.15 Fijación de pantallas	36
Figura 3.16 Cableado en stand IoT.....	37
Figura 3.17 Elementos internos del prototipo IoT.....	38
Figura 3.18 Pantalla IoT.....	38
Figura 3.19 Base para pantalla	39
Figura 3.20 Raspberry Pi.....	39
Figura 3.21 Fuente de poder	40
Figura 3.22 Router WIFI	40
Figura 3.23 Elementos internos conectados	41
Figura 3.24 Vista frontal de prototipo IoT	41

Figura 3.25 Vista posterior de prototipo IoT	42
Figura 3.26 Configuraciones en crontab Raspberry Pi.....	52
Figura 3.27 Credenciales AWS en Raspberry Pi.....	52
Figura 3.28 Cuenta de AWS	53
Figura 3.29 Servicio de AWS IoT.....	54
Figura 3.30 Configuración de objeto en AWS IoT	54
Figura 3.31 ARN de objeto IoT	55
Figura 3.32 Certificado de objeto IoT	55
Figura 3.33 Punto de enlace de objeto	56
Figura 3.34 Política de objeto IoT	56
Figura 3.35 Política activa.....	57
Figura 3.36 Regla de objeto IoT.....	57
Figura 3.37 Acciones de regla	58
Figura 4.1 Envío de mensajes MQTT desde AWS.....	60
Figura 4.2 Stand-Pantalla 1	61
Figura 4.3 Stand-Pantalla 2	62
Figura 4.4 Stand-Pantalla 3	62
Figura 4.5 Stand con los productos	63

INTRODUCCIÓN

Los supermercados o farmacias tienen una característica común, que es la venta en perchas o estanterías. En los stands se pueden observar los precios y descripciones, así como las promociones de ese día. La actividad de colocación de precios, descripción y promociones se lo hace de forma manual, lo que ocasiona que haya errores y esto llega a confundir e incomodar al cliente.

Para el propietario de un comercio es fundamental que la automatización se implemente en alguna medida para minimizar precios y tiempos de trabajo, en especial si se habla de eludir el etiquetado de perchas cada día debido a las promociones de productos.

En este contexto, este trabajo de investigación permite implementar la automatización del etiquetado de perchas de negocios comerciales, el cual tiene como objetivo ser una solución de estanterías IoT conectados a una red Mesh-WIFI-ESP32 con salida a internet el cual envía la información de precios y promociones vía Web Services a través de plataformas de AWS.

En este proyecto se estudia e investiga el desarrollo de un prototipo de estantería IoT. Se realiza un manual de práctica donde se detalla cómo realizar el prototipo a escala, con la finalidad de que los estudiantes de la carrera de ingeniería en electrónica y de la carrera de telecomunicaciones de la UPS planteen sus conocimientos teóricos del internet de las cosas aplicadas a la ingeniería.

Este trabajo de titulación se clasifica de la siguiente manera, en el primer capítulo se explica el problema, antecedentes, importancia y alcance, delimitación, beneficiarios de la propuesta, propuesta de solución y objetivos. En el segundo capítulo se refiere a los fundamentos teóricos de un prototipo de stand comercial inteligente basado en redes WIFI-MESH-ESP32 y hardware como el ESP32 y pantallas leds, el capítulo tres se refiere al marco metodológico, tipo de investigación. El capítulo cuatro trata sobre los resultados de la investigación. El capítulo cinco trata sobre los análisis de resultados, conclusiones y recomendaciones.

1. El problema

1.1. Descripción del problema

Los negocios de comercio como farmacias, venta de víveres o tecnología tienen una característica común el cual es la tradicional venta en perchas, stands o estanterías donde el cliente se acerca a obtener un producto para luego pagarlo en la caja. En estos stands se puede observar los precios y descripciones, así como también las promociones que se encuentren en ese día.

La colocación de este etiquetado y cambio de precios o promociones se lo realiza de manera manual y la carga laboral para los trabajadores del comercio se extiende especialmente cuando hay días festivos o temporadas de venta alta.

Con el incremento de productos muchos de estos comercios pierden valioso tiempo en el etiquetado o cambio de precios que realizan en sus perchas, aproximadamente se toma dos horas al día realizar esta actividad que muchas veces tiene errores y confunden o molestan al cliente.

Para el dueño de un negocio es importante que la automatización se implemente en alguna medida para bajar costos y tiempos de trabajo especialmente si se trata de evitar el etiquetado de perchas todos los días debido a promociones de productos.

En este contexto, este trabajo de investigación implementa la automatización del etiquetado de perchas de negocios comerciales, el cual consiste en una solución de estanterías IoT conectados a una red Mesh-WIFI-ESP32 con salida a internet el cual envía la información de precios y promociones vía Web Services a través de plataformas de AWS.

El etiquetado, cambio de descripciones y promociones serán controlados mediante un servidor web el cual administrará el inventario y los precios que serán mostrados electrónicamente en las perchas inteligentes a través de displays adaptados a los ESP32.

1.2. Antecedentes

La necesidad de evitar la molestia de un cliente, minimizar los errores en un negocio automatizando el trabajo, incentiva al desarrollo de este prototipo IoT de bajo costo, el cual mediante una red MESH-WIFI-ESP32 se podrá automatizar la información necesaria para los estantes comerciales.

Con la realización de este proyecto no solo se logrará minimizar los errores en un negocio de comercio automatizando el trabajo, sino que también servirá como guía de aprendizaje para los futuros ingenieros electrónicos o de telecomunicaciones de la Universidad Politécnica Salesiana sede Guayaquil.

1.3. Importancia y alcance

Con el aporte de esta propuesta, la sociedad en general se beneficia, ya que se obtiene una mejor experiencia al momento de hacer sus compras, de igual manera se ven beneficiados los estudiantes de la carrera de ingeniería en electrónica de la Universidad Politécnica Salesiana sede Guayaquil los cuales podrán poner en práctica los conocimientos adquiridos en las diferentes materias aplicando el internet de las cosas.

En este contexto, se presenta este trabajo de investigación que permite la creación de un prototipo de stands comerciales mediante herramientas del internet de las cosas, hardware de bajo de costo y servidores web en la nube para el cambio automático de precios, descripciones de productos y promociones

1.4. Delimitación

1.4.1. Delimitación temporal

El tiempo estimado para el diseño, implementación y pruebas de funcionamiento del prototipo es de 12 meses.

1.4.2. Delimitación espacial

Debido a la emergencia sanitaria por motivo de la pandemia del COVID-19 y siguiendo las indicaciones de la dirección de carrera de Ingeniería Electrónica de la Universidad Politécnica Salesiana, sede Guayaquil, la implementación del sistema IoT de estanterías inteligentes se prueba en el domicilio del egresado Anthony Navarrete autor del proyecto de tesis, en la ciudad de Guayaquil.

El testeo y análisis de resultados se lo hace en tiempo real de acuerdo con el diseño del prototipo IoT toda esta información es administrada localmente por un servidor local y vía web mediante servidores web en la nube de AWS.

1.4.3. Delimitación académica

Este trabajo de investigación donde el resultado es un prototipo IoT funcional y operativo es una herramienta fundamental para el desarrollo de las capacidades y destrezas de los futuros ingenieros electrónicos e ingenieros de telecomunicaciones de la Universidad Politécnica Salesiana sede Guayaquil.

Con este trabajo se pone en práctica los conocimientos adquiridos en materias como redes de computadora, electrónica analógica y digital, medios de transmisión.

1.5. Beneficiarios de la propuesta

Este proyecto tiene como beneficiarios directos a los pequeños o grandes comerciantes de la ciudad de Guayaquil que venden diferentes tipos de productos y los muestran en sus perchas.

De esta manera se tecnifica el control de inventario, el marketing y el uso de horas hombre para esta actividad de actualización de información de precios, productos o promociones en cada percha que contiene productos.

Además, se ven beneficiados los estudiantes de la carrera de ingeniería electrónica y de la carrera de telecomunicaciones de la Universidad Politécnica Salesiana sede Guayaquil, puesto que se realiza un manual de práctica donde se detalla cómo realizar el prototipo a escala.

1.6. Propuesta de solución

La necesidad de automatizar procesos en los comercios incentiva a la investigación y realización de este prototipo IoT de perchas inteligentes mediante una red MESH con el ESP32 y servidores web en la nube con AWS.

Con la realización de este proyecto no solo se logra una tecnificación de los procesos de etiquetado y promociones de perchas inteligentes en pequeños o grandes negocios en Guayaquil - Ecuador, también sirve como guía de aprendizaje para los futuros ingenieros electrónicos o de telecomunicaciones de la Universidad Politécnica Salesiana sede Guayaquil que quieran poner en práctica sus conocimientos mediante esta maqueta del prototipo a desarrollar.

1.7. Innovación e impacto del proyecto

En este proyecto de investigación se plantea realizar el diseño e implementación de un prototipo de estanterías comerciales IoT para la automatización de precios, promociones y descripciones de productos comerciales utilizando una red ESP-WIFI-MESH y hardware de bajo costo.

El desarrollo de este trabajo de titulación tiene un nivel de innovación alto, debido a que la configuración, programación y recolección de datos se realiza con plataformas IoT diseñadas y adecuadas para la actualización de datos.

1.8. Objetivos

1.8.1. Objetivo general

Diseñar e implementar estantería IoT para la automatización de precios y descripciones de productos comerciales utilizando una red ESP-WIFI-MESH y hardware de bajo costo.

1.8.2. Objetivos específicos

- Diseñar una maqueta de tres stands IoT que se interconecten mediante una red ESP-WIFI-MESH y ESP32.
- Configurar servicios web, locales mediante Raspberry PI y servidor en la nube, para la administración del cambio remoto de precios, promociones y descripciones de los productos de los stands.

- Configurar paneles leds y ESP32 en las perchas inteligentes que visualicen precios, descripción y promociones en cada fila de la maqueta de los stands.
- Realizar manual de práctica del prototipo IoT de perchas inteligentes para el uso de estudiantes de la carrera de ingeniería electrónica y telecomunicaciones de la Universidad Politécnica Salesiana sede Guayaquil.

2. Fundamentos teóricos

2.1. ESP32

El ESP32 es un SoC, que ofrece conectividad Wifi y Bluetooth de 2,4 GHz de frecuencia, está diseñado para aplicaciones móviles, electrónica portar e IoT.

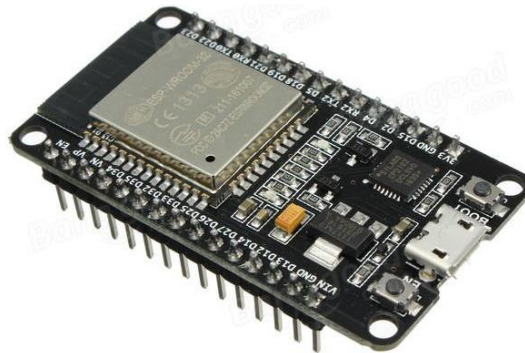


Figura 2.1. Modulo ESP32
(PROMETEC, 2018)

La integración de Bluetooth y Wi-Fi posibilita una vasta gama de aplicaciones. La utilización de Wi-Fi posibilita una comunicación de mediano alcance y conectarse a una red LAN por medio de un Router acceso al internet, mientras que el Bluetooth posibilita conectarse de manera directa a otro dispositivo como un celular.

Es ideal para aplicaciones de baja potencia, el SoC ESP32 cuenta con una marca de reloj de grano fino, diversos modos de potencia y escalado de potencia dinámico.

El periodo de trabajo bajo se usa para reducir la proporción de energía que gasta el SoC ESP32. La salida del amplificador de potencia es ajustable, lo cual ayuda a una compensación óptima entre el rango de comunicación, la rapidez de datos y el consumo de energía (Mouser Electronics., 2017).

2.1.1. Características de ESP32

- **Diseño robusto:** El ESP32 es capaz de funcionar de manera fiable en espacios industriales, con una temperatura de desempeño de -40°C a $+125^{\circ}\text{C}$. Alimentado por circuitos de calibración avanzados, ESP32 puede borrar dinámicamente las imperfecciones del circuito externo y ajustarse a los cambios en las condiciones externas.
- **Consumo de energía ultra bajo:** Pensado para dispositivos móviles, dispositivos electrónicos portátiles y aplicaciones de IoT, ESP32 consigue un consumo de energía ultra bajo con una conjunción de diversos tipos de programa patentado. El ESP32 además incluye propiedades de vanguardia, como la sincronización de reloj de grano fino, diversos modos de energía y escalado dinámico de energía.
- **Alto nivel de integración:** ESP32 está altamente integrado con interruptores de antena incorporados, balún de RF, amplificador de potencia, amplificador de recepción de bajo ruido, filtros y módulos de gestión de energía. ESP32 añade funcionalidad y versatilidad invaluable a sus aplicaciones con requisitos mínimos de placa de circuito impreso (ESPRESSIF, n.d.).
- **Chip híbrido Wi-Fi y Bluetooth:** ESP32 puede funcionar como un sistema independientemente completo o como un dispositivo esclavo de una MCU host, lo cual disminuye la sobrecarga de la pila de comunicación en el procesador de la aplicación primordial. ESP32 puede interactuar con otros sistemas para conceder funcionalidad Wi-Fi y Bluetooth por medio de sus interfaces SPI / SDIO o I2C / UART.

2.1.2. Especificaciones ESP32

El ESP32 es un procesador del fabricante chino Espressif que tiene las siguientes especificaciones:

- Procesador dual Core Xtensa® LX6 de 32 bits
- Velocidad de reloj: Entre 160 Mhz y 240 Mhz
- Wi-Fi integrado: Access point & Station
- Bluetooth 4.2 2.4 Ghz; BT 2.0 y 4.0 BLE
- 36 GPIO pins.
- 2 x Digital to Analog converter DAC de 8 bits.
- Pueden definirse hasta 16 canales de PWM.
- 2 x UART o puertas serie.
- 2 x I2C channels y 4 x SPI channels.

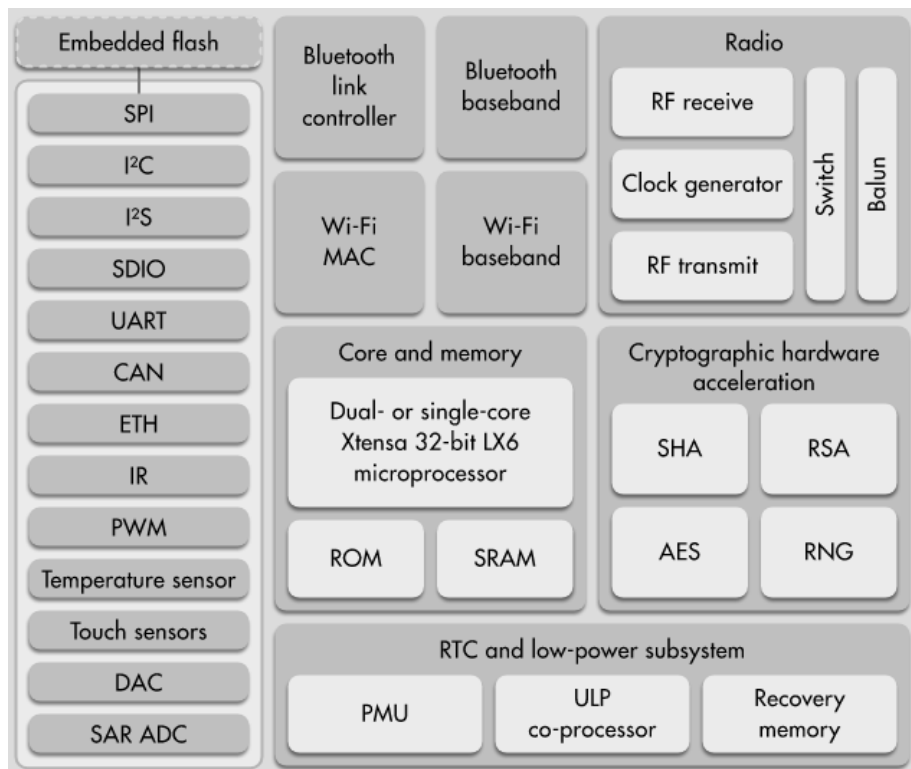


Figura 2.2 Diagrama en bloques del ESP32
(Pedro Bertoleti, 2019)

2.2. AWS

Amazon Web Services es un proveedor de servicios en la nube, que posibilita contar con almacenamiento, recursos de computación, aplicaciones móviles, bases de datos y un extensos servicios en modalidad de cloud computing.

AWS está elaborado para ser el entorno de informática en la nube más flexible y seguro disponible en la actualidad. Su infraestructura principal fue desarrollada para llevar a cabo los requisitos de seguridad del ejército, los bancos internacionales y otras empresas que deben cumplir cabo requisitos de confidencialidad rigurosos. Esto cuenta con el respaldo de un extenso grupo de herramientas de seguridad en la nube, con 230 servicios y características de seguridad, conformidad y gobernanza. AWS es compatible con 90 estándares de estabilidad y certificaciones de conformidad y los 117 servicios de AWS que almacenan datos de los consumidores ofrecen la capacidad de cifrar aquellos datos.

2.2.1. Ventajas de AWS

AWS tiene presencia en 190 territorios en torno al mundo y cuenta con más de 10.000 partners asociados. Sus centros de datos permanecen en USA, Japón, Australia, Brasil, Europa y Singapur y los consumidores gozan de ventajas tales como:

- Seguridad: AWS cuenta con certificaciones y auditorias, por ejemplo: PCI DSS grado 1, FISMA Moderate, HIPAA Y SOC 1, ISO 27001 y auditoria SOC 2, que hacen que sea 100% confiable.
- Bases de datos: AWS posibilita acceder a bases de datos como MySQL, Oracle, Aurora, PostgreSQL, SQL Server, MongoDB, etc.
- Bajo costo: Los negocios al conseguir una conexión con la nube no harán una inversión en infraestructura enorme, por esto, sus precios van a ser menores. Además, AWS ha logrado ser cada vez más eficiente, sacando beneficio de las economías de escala y logran minimizar sus precios, y transfieren aquellos ahorros a los costes de sus consumidores. (Mónica Gimenez, 2020)
- Accesibilidad: Uno de los puntos más importantes de AWS es lo simple que es su acceso y descubrir servicios que logren absorber la carga de trabajo que se le exige. Además, AWS se acomoda inmediatamente a las necesidades del mercado, por lo cual la llegada de nuevos servicios a AWS es continua.
- Resiliencia: Los CPDs de AWS fueron creados para seguir haciendo un trabajo en las peores contingencias. Tienen la posibilidad de diseñar infraestructuras con un 99.99999% de tiempo online.

2.2.2. Herramientas de AWS

Los servicios de AWS son adecuados tanto para autónomos como para empresas pequeñas, medianas o grandes, porque existen muchas formas de expandir instancias o almacenamiento a medida que la empresa crece (TicPortal, 2019).



Figura 2.3 Amazon Web Services
(DIGITEOS, 2020)

Amazon Web Services ofrece herramientas, y estos son los principales servicios a las siguientes características:

- Computación en la nube: Tiene todo lo necesario para crear y mantener una instancia. Amazon EC2 es el mejor entre los servicios de computación en la nube de Amazon.
- Base de datos: Varios tipos de bases de datos pueden ingresar a la nube a través del servicio Amazon RDS, incluidos diferentes tipos para elegir, como MySQL, PostgreSQL, Oracle, SQL Server y Amazon Aurora, o Amazon DynamoDB para NoSQL.

- Crear una red virtual: Con la ayuda del servicio Amazon VPC, se permite la creación de una red privada virtual a través de la nube.
- Aplicaciones comerciales: Amazon WorkMail es un servicio de correspondencia proporcionado por Amazon, que se puede agregar a otros servicios como Amazon WorkDocs y Amazon WorkSpaces.
- Administrador de contenido y almacenamiento: Diferentes tipos de almacenamiento para archivos a los que se accede regularmente, con poca frecuencia o incluso como archivos. Amazon S3 es el servicio principal, aunque otros servicios como Amazon Glacier o Amazon EBS lo complementan.
- Business Intelligence (BI): Un sistema que se utiliza para el análisis de datos empresariales a gran escala y otros servicios de gestión del flujo de datos.
- Administración de aplicaciones móviles: Herramientas como Amazon Mobile Hub permiten administrar, crear, probar y el mantenimiento de aplicaciones móviles a través de la nube.
- Internet de las cosas (IoT): Sirve para establecer una conexión y analizar todos los dispositivos conectados a Internet y los datos que recopilan.
- Herramientas para desarrolladores: Se utilizan para almacenar código, automatizar la implementación e incluso publicar software mediante sistemas de entrega continua.

- Seguridad y control de acceso: La verificación de identidad se puede establecer a través de varios pasos para proteger el acceso a los sistemas internos, ya sea que estén en la nube o instalados localmente en sus instalaciones.

2.3. AWS IoT

AWS IoT proporciona servicios en la nube que conectan dispositivos IoT a otros dispositivos y servicios en la nube de AWS. AWS IoT proporciona software de dispositivo para ayudarlo a integrar dispositivos IoT en soluciones basadas en AWS IoT. Si los dispositivos pueden conectarse a AWS IoT, AWS IoT puede conectarlos a los servicios en la nube proporcionados por AWS (Aws.amazon.com, 2021).

AWS IoT le permite elegir la tecnología más adecuada y actualizada para su solución. Para ayudarlo a administrar y respaldar sus dispositivos IoT en el campo, AWS IoT Korea admite los siguientes protocolos:

- MQTT (Mensaje en cola y transporte de telemetría)
- MQTT sobre WSS (Seguridad de Websockets)
- HTTPS (Protocolo de transferencia de hipertexto: seguro)
- LoRaWAN (Red de área amplia de largo alcance)

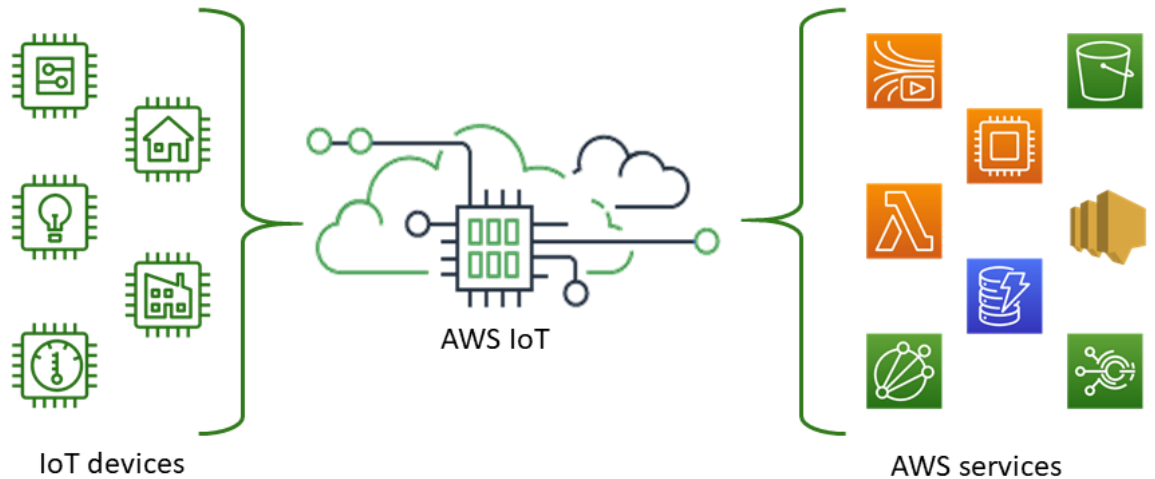


Figura 2.4 AWS IoT
(Aws.amazon.com, 2021)

2.4. Raspberry Pi

La Raspberry Pi es una computadora de bajo costo, de tamaño pequeño, del tamaño de una tarjeta de crédito, y puede conectarse a un monitor de computadora o TV y usarse con un mouse y un teclado estándar. Es una pequeña computadora que ejecuta el sistema operativo Linux que permite a personas de todas las edades explorar la informática y aprender lenguajes de programación como Scratch y Python. Puede completar la mayoría de las tareas típicas de una computadora de escritorio, desde navegar por Internet, reproducir videos de alta resolución, procesar documentos de oficina hasta jugar juegos (MCI Electronics., 2017).

Además, Raspberry Pi tiene la capacidad de interactuar con el mundo exterior y se puede utilizar para una variedad de proyectos digitales, desde reproductores de música y video, detectores parentales, estaciones meteorológicas hasta cajas de pájaros con cámaras infrarrojas.

2.4.1. Ventajas Raspberry Pi

Raspberry Pi es muy fácil de usar porque básicamente es una computadora muy pequeña. Cualquier solución que requiera hardware y un sistema operativo puede usar RPi (Raspberry Pi), que se usa ampliamente en soluciones autónomas para automatización del hogar, ayuda en sistema de red y muchas otras aplicaciones.

La ventaja más obvia de RPi es que es una computadora de bolsillo y es muy barata. Tiene un sistema Linux totalmente integrado y puede desarrollarlo internamente como lo haría en una computadora portátil o de escritorio tradicional. Es Fácil de usar, tienen un sistema operativo con todas las ventajas que conlleva, tiene un tamaño muy pequeño y también tienen pines de entrada/salida, por lo que se pueden conectar a otros componentes electrónicos, como sensores, LEDs, etc.

2.4.2. Raspberry Pi 4

La velocidad y el rendimiento de la nueva Raspberry Pi 4 son superiores a los de los modelos anteriores. Creó una experiencia de escritorio completa por primera vez. Descubrimos que esta experiencia es perfecta y altamente reconocible, pero en una forma más pequeña, más eficiente en términos de energía y más rentable. (Raspberry Pi, 2019)

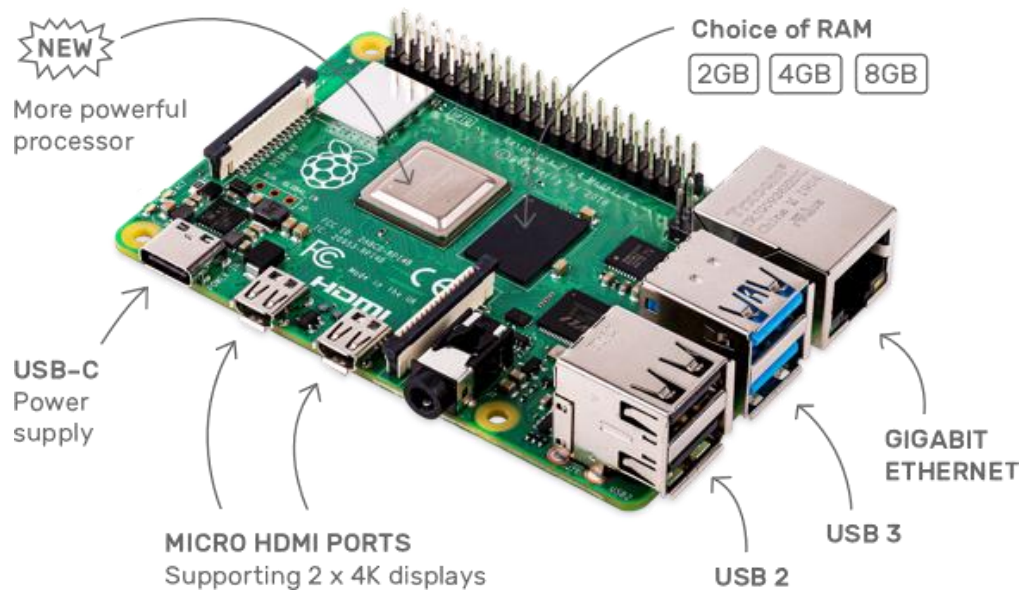


Figura 2.5. Raspberry Pi 4
(Raspberry Pi, 2019)

2.4.3. Especificaciones de Raspberry Pi 4

- Broadcom BCM2711, SoC de 64 bits Cortex-A72 de cuatro núcleos (ARM v8) 1,5 GHz
- SDRAM LPDDR4-3200 de 1 GB, 2 GB, 4 GB y 8 GB (según el modelo)
- 2.4 GHz y 5.0 GHz IEEE 802.11ac inalámbrico, Bluetooth 5.0, BLE
- Gigabit Ethernet
- 2 puertos USB 3.0; 2 puertos USB 2.0.
- Conector GPIO Raspberry Pi estándar de 40 pines (totalmente compatible con placas anteriores)
- 2 puertos micro HDMI (hasta 4k60 compatible)
- Puerto de pantalla MIPI DSI de 2 canales

- Puerto de cámara MIPI CSI de 2 canales
- Puerto de video compuesto y audio estéreo de 4 polos
- H.265 (decodificación 4kp60), H264 (decodificación 1080p60, codificación 1080p30)

2.5. WIFI

Wifi es una tecnología de red inalámbrica, la cual permite que dispositivos como, computadoras, dispositivos móviles y otros equipos, puedan interactuar con internet e intercambiar información entre sí, estableciendo una red.

La conectividad a Internet funciona por medio de un router inalámbrico. Cuando un dispositivo accede a wifi, se conecta por medio de un router inalámbrico, y este permite que los dispositivos interactúen con el internet.

Técnicamente, el estándar IEEE 802.11 define un protocolo para comunicarse con los dispositivos inalámbricos habilitados para Wi-Fi actuales, entre esos, los enrutadores y puntos de acceso inalámbricos. El punto de acceso inalámbrico es compatible con los diferentes estándares IEEE. Cada estándar es una modificación aprobada después de un cierto período de tiempo y estos se aplican a diferentes frecuencias, proporcionan diferentes anchos de banda y también admiten diferentes números de canales (Cisco, 2017.).



Figura 2.6 Wi-Fi Alliance logo
(Alliance, 2020)

2.6. Red Mesh

En una red de malla o red mesh, solo un nodo debe estar conectado físicamente a la conexión de red WAN. Este nodo cableado comparte de forma inalámbrica su conexión a Internet con todos los demás nodos cercanos. La red mesh dirige de la mejor manera el tráfico que pasa por una red para poder obtener siempre la mejor señal en la red.

Cuanto más nodos hay, más conexiones se extienden, creando una "nube de conexión" inalámbrica que puede servir a pequeñas oficinas o ciudades con millones de personas.

Solo un nodo de la red cableada debe estar conectado directamente a Internet. El nodo cableado comparte de forma inalámbrica la conexión a Internet con el grupo de nodos más cercano y luego comparte la conexión a Internet con el grupo de nodos más cercano, y así sucesivamente. Esto quiere decir que cada nodo individual no necesita estar conectado a nada. Si está al aire libre, solo necesita una fuente de alimentación, como un enchufe tradicional, una batería o un panel solar. Los nodos exteriores están encapsulados en una cubierta resistente a la intemperie y se pueden instalar en cualquier lugar, incluidas piscinas telefónicas, techos, etc. (Navas, 2017).

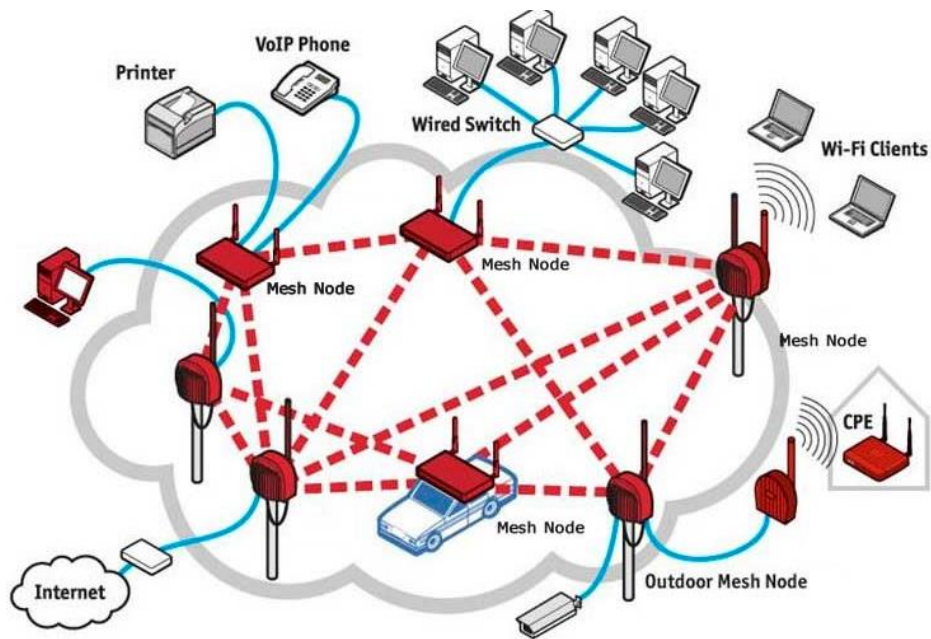


Figura 2.7. Red Mesh
(Navas, 2017)

2.7. Red Wifi Mesh

Básicamente, una red Wifi Mesh es una red compuesta por enrutadores / estaciones base y sus satélites o puntos de acceso. Estos satélites o puntos de acceso se comunican entre sí para formar una única red Wifi con el mismo SSID y contraseña para que los usen los usuarios.

En comparación con las redes inalámbricas fijas o alámbricas, la mayor ventaja de las redes de malla es que son verdaderas redes inalámbricas. La mayoría de los puntos de acceso inalámbricos tradicionales aún necesitan estar conectados a Internet para transmitir señales. Para grandes redes inalámbricas, los cables Ethernet deben enterrarse en techos y paredes y en áreas públicas.

En la red de Wifi en malla o Wifi Mesh, su fuerza radica precisamente en la gestión avanzada de los elementos de la red. La red de malla Wifi calcula el nodo o satélite más adecuado para la conexión en un momento dado según el estado de otros

nodos, dispositivos conectados, distancia de cada satélite, intensidad de la señal y muchos otros factores. Es completamente transparente ya que los usuarios no tienen que preocuparse de a que nodo se conectará.

2.8. IoT

El Internet de las cosas se trata sobre las conexiones de tecnologías integradas que tienen dentro objetos físicos y se usan para interactuar y comunicarse con los estados internos. El IoT hace énfasis en la comunicación de máquina a máquina. Cantidades masivas de datos incrementaron su complejidad en detección, comunicaciones, controlador, pero tendrá un crecimiento diario. Aunque el futuro de IoT será predecible para integrarse. Como sistema inteligente, los avances de IoT se pueden decidir por medio la cooperación de interoperabilidad, habilidad, trabajo en equipo, sostenibilidad energética, privacidad, confianza, confidencialidad y seguridad.

El término IoT se refiere a todos los sistemas de dispositivos físicos que reciben y transmiten datos a través de una red inalámbrica con una mínima intervención humana, gracias a la integración de dispositivos informáticos en varios objetos.

Por ejemplo, los termostatos inteligentes (es decir, que usan IoT) reciben datos de ubicación de su automóvil inteligente mientras conduce para ajustar la temperatura en su hogar antes de que llegue. Todo esto se logra sin su intervención e incluso puede brindar mejores resultados que hacerlo manualmente.

Los sistemas IoT tradicionales, como el hogar inteligente antes mencionado, funcionan enviando, recibiendo y analizando datos continuamente en un ciclo de retroalimentación. Según el tipo de tecnología IoT, los humanos o los sistemas de inteligencia artificial y aprendizaje automático (AI/ML) pueden analizar estos datos casi de inmediato o durante un período de tiempo (Red Hat, 2020).



Figura 2.8 lot
(Red Hat, 2020)

2.9. Python

En términos de tecnología, Python es un lenguaje de programación orientado a objetos de alto nivel con semántica dinámica incorporada, que se utiliza principalmente para el desarrollo de aplicaciones informáticas y web.

Es muy atractivo en el mundo del desarrollo rápido de aplicaciones (RAD) porque proporciona opciones dinámicas de escritura y enlace dinámico (Python, 2019).

Python es relativamente simple y, por lo tanto, fácil de aprender, lo que requiere una sintaxis única que se centre en la legibilidad. Los desarrolladores pueden leer y traducir el código de Python más fácilmente que otros lenguajes.

Por lo tanto, esto reduce el costo de mantenimiento y desarrollo del programa, ya que permite que los equipos trabajen en colaboración sin barreras significativas de lenguaje y experimentación.

Además, admite el uso de módulos y paquetes, lo que significa que los programas se pueden diseñar de forma modular y el código se puede reutilizar en varios

proyectos (Cursosaula21.com, 2022).



Figura 2.9 Python
(Python, 2019)

3. Marco metodológico

3.1. Tipo de investigación

El proyecto se basa en una investigación experimental, ya que se realiza un prototipo de stands automatizados, donde se puede controlar remotamente la información del producto, precio y promociones, y se monitoriza remotamente por medio de AWS.

Se realiza una investigación experimental porque las variables de estudio se manipulan de manera práctica por medio de la maqueta de stands automatizado.

3.2. Diseño de investigación

Se utiliza para investigaciones de tipo experimental porque según (Arias, 2012), esto permite manipular un objeto o grupo de personas, sometiéndolas a diferentes condiciones y estímulos para poder observar los resultados.

3.3. Enfoque de la investigación

En la investigación en este campo, las variables se consideran a partir de dos partes básicas, se adopta un método mixto de cualitativo y cuantitativo. Por ejemplo, revisión teórica se realiza a partir de varias fuentes bibliográfica, tales como: libros, trabajos relacionados, artículos científicos y otras publicaciones, en las que se analiza de manera cualitativa la información adquirida.

3.4. Metodología de investigación

La metodología utilizada es la experimental, ya que para este proyecto se realiza el control de información de precios, promociones y descripciones de productos de stands de una manera centralizada mediante la Raspberry PI y los ESP32 interconectados de manera inalámbrica y modificados remotamente a través de

AWS.

La metodología de investigación utilizada es la exploratoria, ya que se realiza la construcción de un prototipo de estanterías comerciales IoT que están conectadas mediante una red ESP32-WIFI-MESH.

El prototipo se lo administra de manera remota mediante AWS para hacer el cambio automático de precios, descripciones de productos y promociones, utilizando ESP32, Raspberry Pi para la administración local, y mostrado por medio de un display.

3.5. Proyectos de investigación vinculados

A continuación, se describe los títulos de varios proyectos de investigación vinculados a este trabajo.

Desarrollo y configuración de un sistema automatizado para el control y monitoreo de los stands publicitarios diseñados por la empresa Stand Design e implementación del proceso en un modelo a escala

Autores: T Jácome Ochoa, Paúl Daniel Verdezoto Jaramillo, Rommel Augusto
Universidad Politécnica Salesiana

Sistema de agentes para control de stock de almacén basado en identificación por radiofrecuencia.

Autores: Chuqui Nieto, Luis Alfonso
Universidad Politécnica Salesiana

Diseño e implementación de sistema de monitoreo Diseño y construcción de estantes inteligentes basados en RFID en banda UHF

Autores: Gutiérrez Arnal, Lourdes Monleón Torre, Victor

Universidad Politécnica de Cataluña

3.5. Título de la propuesta

Diseño e implementación de un prototipo de estanterías IoT para la automatización de precios y descripciones de productos comerciales utilizando una red ESP-WIFI-MESH y hardware de bajo costo.

3.6. Descripción de la propuesta

A continuación, se muestra la descripción de la propuesta planteada en este trabajo de investigación.

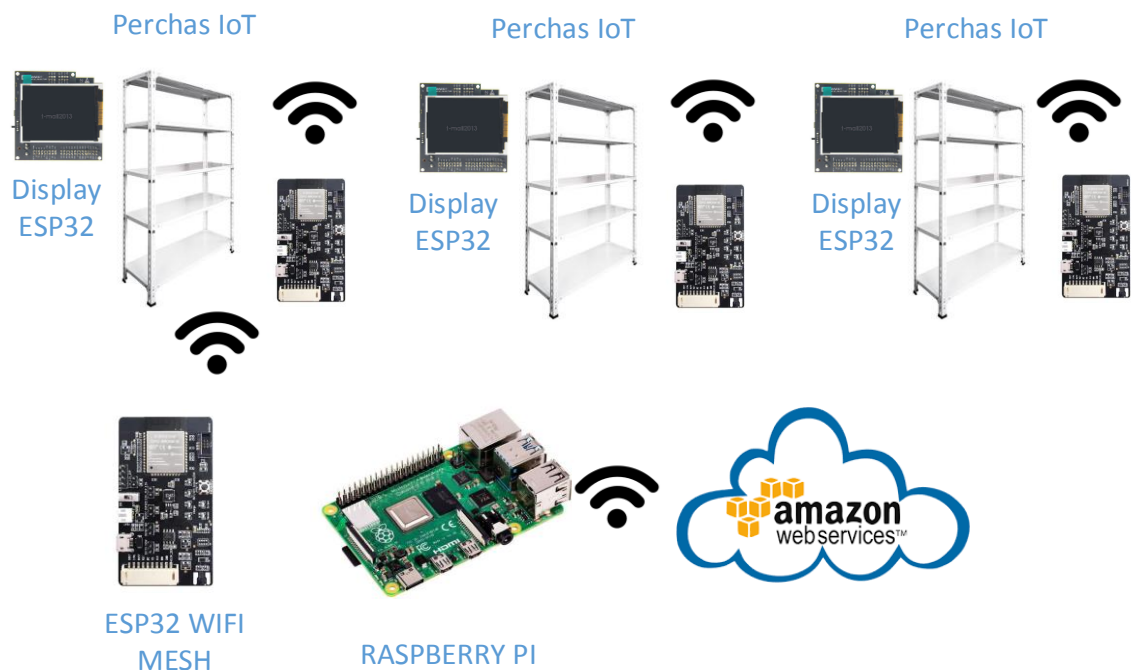


Figura 3.1 Prototipo de estantería comerciales IoT.

En la gráfica 3.1 se observa el prototipo de estanterías comerciales IoT conectadas mediante una red ESP32-WIFI-MESH y administradas remotamente mediante AWS para el cambio automáticos de precios, descripciones de productos y promociones, utilizando ESP32, displays, y Raspberry Pi para la administración local.

La propuesta planteada se la programa mediante el microcontrolador ESP32 el cual centralizará la red WIFI-MESH con los demás ESP32 conectados en cada stand y mediante displays de buena resolución mostrarán los valores de precios, descripciones de productos y promociones. La información es almacenada de manera local en la Raspberry PI y posteriormente cargado a servidores en la nube de Amazon.

El stand IoT está diseñado para ser administrado mediante plataforma IoT de AWS, para esto se realiza la configuración de una cuenta de AWS y se sincroniza con el objeto de que en este caso es una Raspberry Pi que conforma parte del prototipo IoT de Stand inteligente.

3.7. Diseño electrónico del prototipo

A continuación, se muestran los diseños electrónicos del prototipo IoT:

La figura 3.2 muestra la designación de pines del ESP32 utilizado en este proyecto.

ESP32 Wi-fi CORE

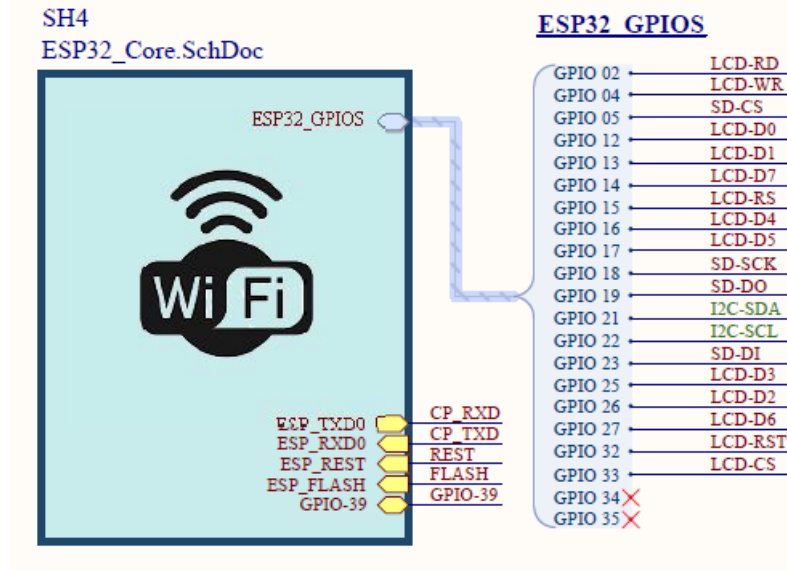


Figura 3.2 Conexiones de ESP32

La figura 3.3 muestra la conexión USB a UART del CP2102.

USB TO UART BRIDGE

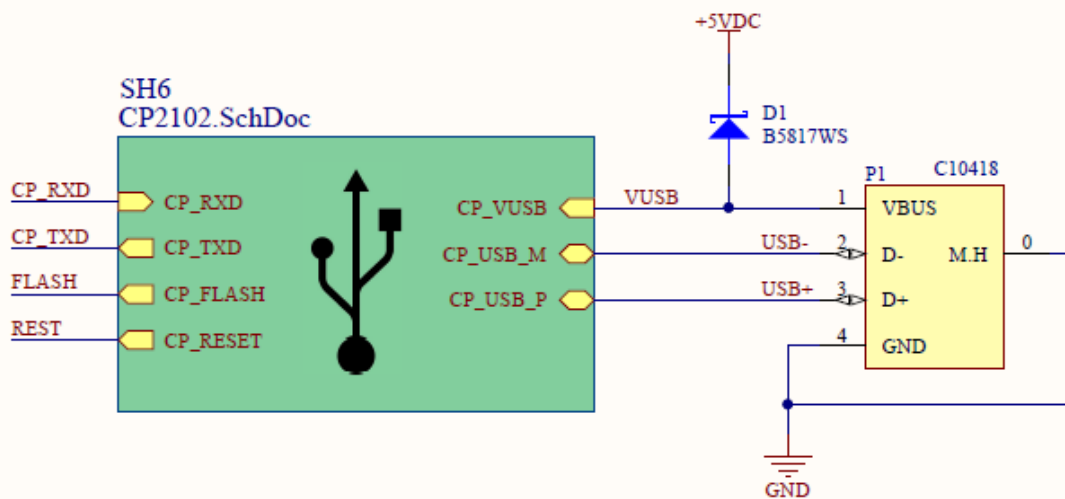


Figura 3.3 USB Uart Bridge

La figura 3.4 muestra el esquemático de la fuente de poder de 5VDC.

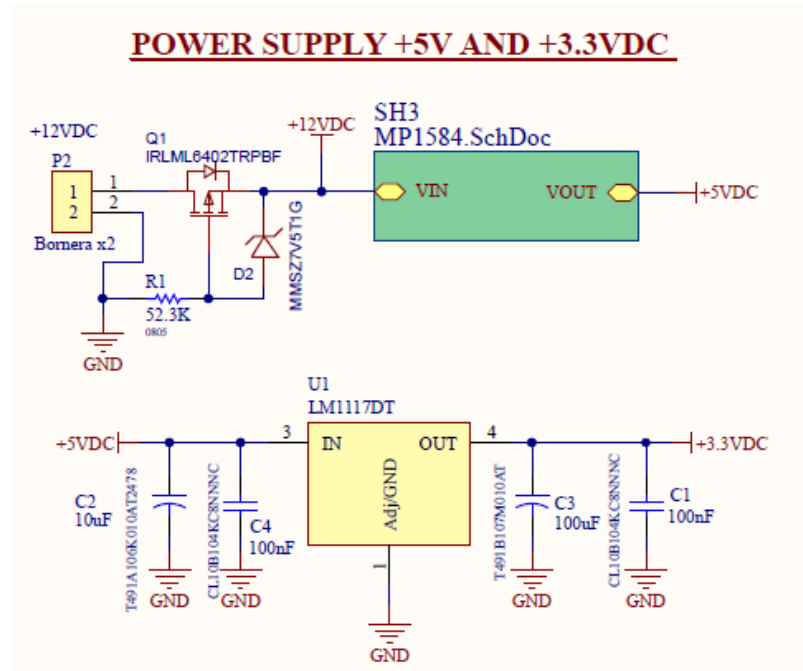


Figura 3.4 Power Supply

La figura 3.5 muestra el esquemático de las pantallas TFT de 3,5 pulgadas con resolución de 320 x 480

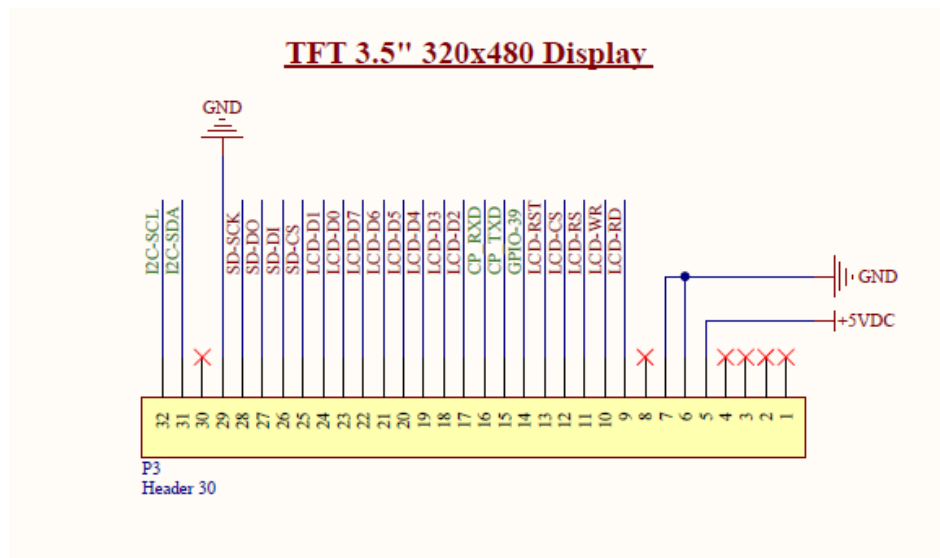


Figura 3.5 Display

La figura 3.6 muestra el esquemático de conexiones del CP1202.

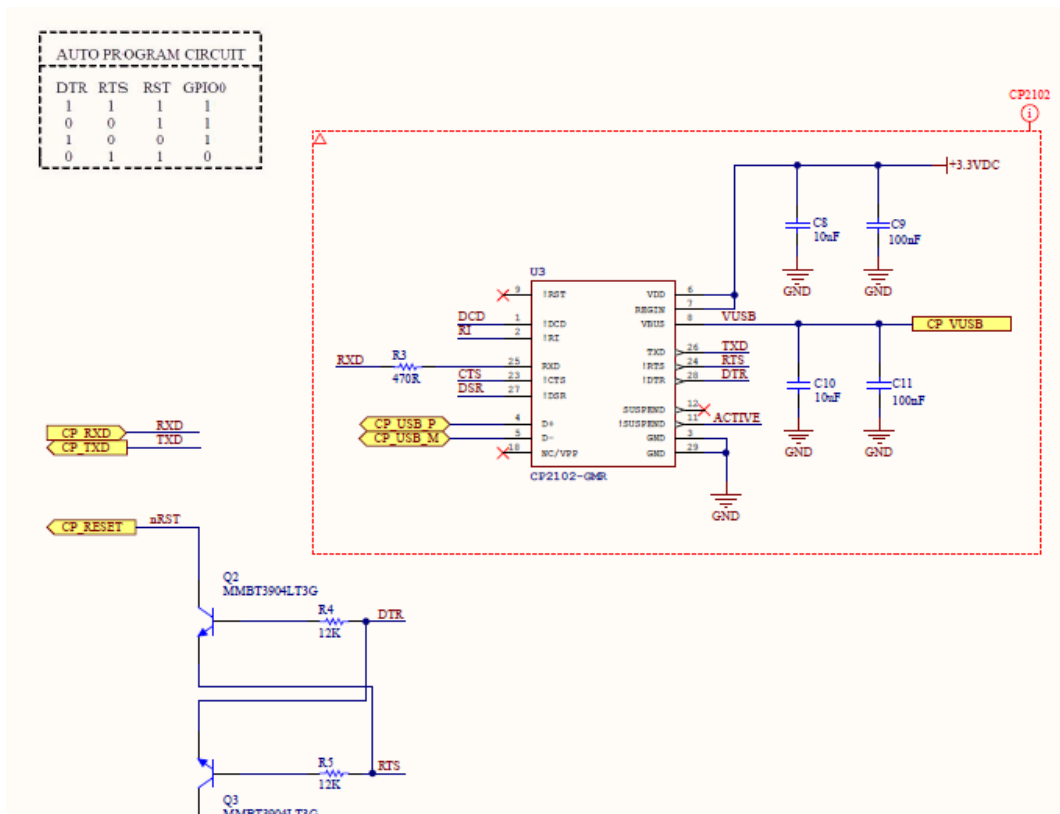


Figura 3.6 CP1202

La figura 3.7 muestra el esquemático de conexiones del MP1584EN

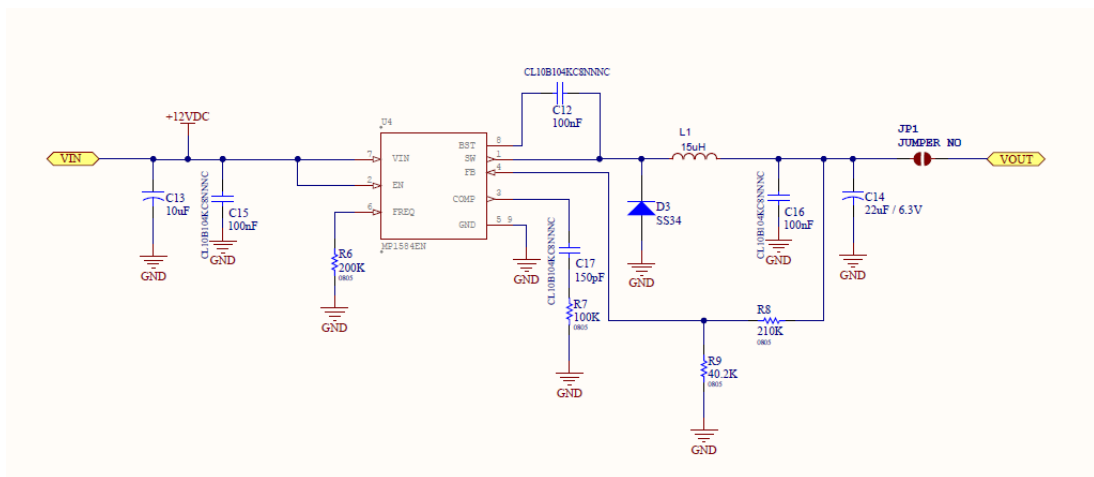


Figura 3.7 MP1584EN

La figura 3.8 muestra la designación de pines GPIO del ES32.

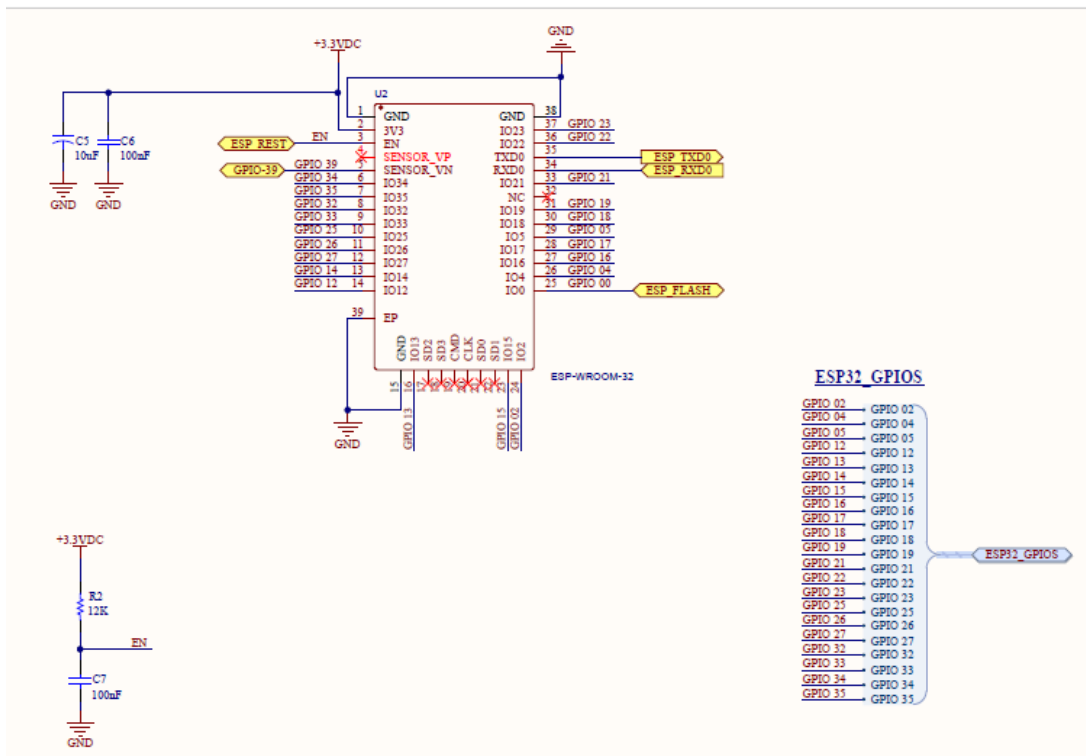


Figura 3.8 Conexiones de ESP32

La figura 3.9 muestra el diseño de la placa PCB.

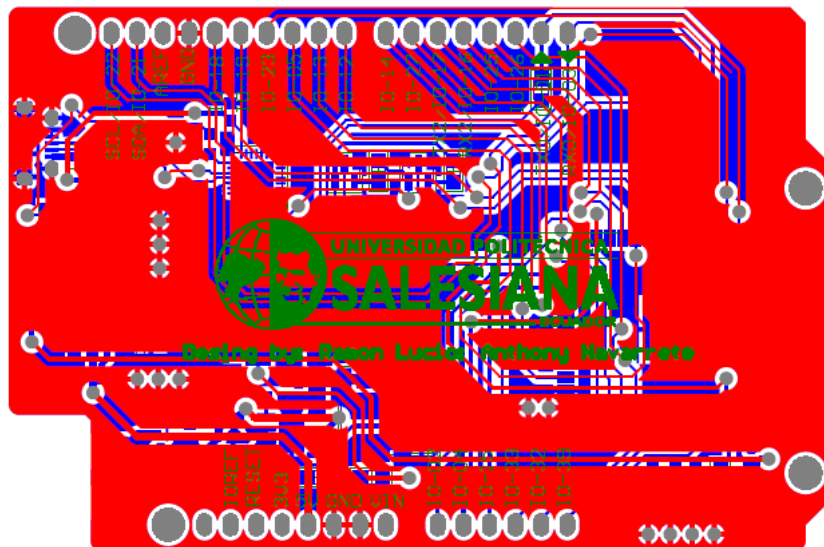


Figura 3.9 Diseño de placa PCB

La figura 3.10 muestra el diseño de la pantalla IoT.

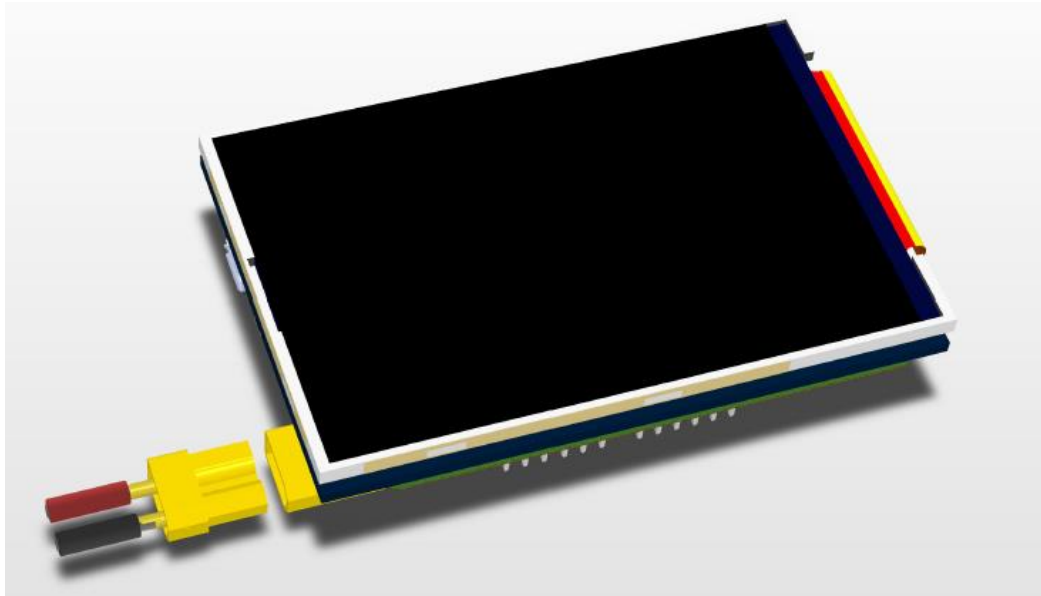


Figura 3.10 Diseño de Pantalla IoT

3.8. Maqueta de stands

Para el diseño de la maqueta del stand IoT se fabrica un stand de hierro con bases de porcelana para colocar los productos. En la figura 3.11 se puede observar el ajuste que se realiza al armar el stand IoT.



Figura 3.11 Instalación de elementos del prototipo IoT

El stand IoT se compone de 3 niveles mediante la estructura de hierro y las bases de porcelana. Ver figura 3.12.



Figura 3.12 Stand físico

En la figura 3.13 se muestra el stand con las pantallas instaladas.



Figura 3.13 Stand IoT

En la figura 3.14. se muestra la instalación del cableado del stand IoT.



Figura 3.14 Instalación de pantallas

La figura 3.15 muestra como se fijaron las pantallas en el stand IoT.



Figura 3.15 Fijación de pantallas

La figura 3.16 muestra los ajustes de la pantalla IoT con el cableado del prototipo.



Figura 3.16 Cableado en stand IoT

En la figura 3.17 se muestran los equipos que componen internamente la caja del stand IoT. Como se puede apreciar, está el Raspberry Pi, router WIFI, fuente de poder y las conexiones eléctricas y de red entre los elementos. Cabe indicar que la Raspberry Pi se conecta mediante WIFI. El router Wifi conecta también de manera inalámbrica al ESP32 master y este a su vez interconecta mediante una red mesh a los otros ESP32 esclavos.



Figura 3.17 Elementos internos del prototipo IoT

En la figura 3.18 se observa la caja de la pantalla, este dará protección física a la placa interna y a la pantalla donde se visualizarán los precios y descripciones de los productos.



Figura 3.18 Pantalla IoT

En la figura 3.19 se observa la base de fijación de la pantalla del prototipo de stand IoT.

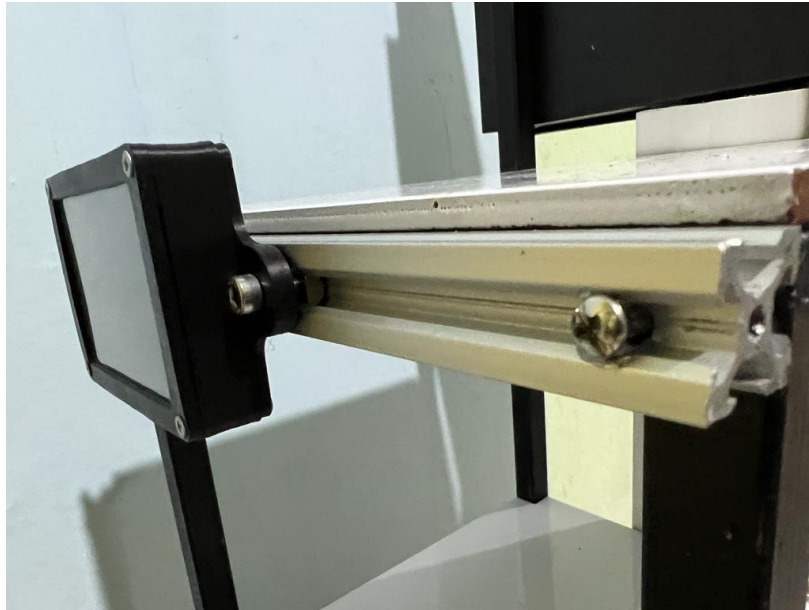


Figura 3.19 Base para pantalla

La figura 3.20 muestra a la Raspberry Pi utilizada en el proyecto, como se aprecia este tiene una carcasa de protección de la placa interna.



Figura 3.20 Raspberry Pi

La figura 3.21 muestra la fuente de poder que alimenta a los elementos internos del prototipo IoT.



Figura 3.21 Fuente de poder

La figura 3.22 muestra al router WIFI TP-Link, este elemento creará la red cableada e inalámbrica del prototipo.



Figura 3.22 Router WIFI

La figura 3.23 muestra los elementos internos del prototipo IoT. Estos elementos se encuentran enchufados a la red eléctrica y conectados por WIFI.



Figura 3.23 Elementos internos conectados

La figura 3.24 muestra la vista frontal del prototipo IoT.



Figura 3.24 Vista frontal de prototipo IoT

La figura 3.25 muestra la vista posterior del prototipo IoT.



Figura 3.25 Vista posterior de prototipo IoT

3.9. Programación del prototipo Script de Python

En esta sección se muestra el código o script de Python que se utilizó para las actualizaciones de las etiquetas IoT y el ESP32

```
import websocket  
import time  
import _thread
```

```
from AWSIoTPythonSDK.MQTTLib import AWSIoTMQTTClient
```



```

def on_message(ws, message):
    print(message)

def on_error(ws, error):
    print(error)

def on_close(ws, close_status_code, close_msg):
    print("### closed ###")

def stand(self, params, packet):
    print ('Mensaje recibido de AWS IoT Core')
    print ('Topic: '+ packet.topic)
    print (packet.payload)
    stand_send = packet.payload
    print (stand_send.decode("utf-8"))
    tag_data_string = stand_send.decode("utf-8")
    if __name__ == "__main__":
        print("Sending data...")
        ws = websocket.create_connection("ws://10.30.10.201:443")
        ws.send(tag_data_string)
        time.sleep(1)
        ws.close()
        print("Etiqueta Actualizada...")

myMQTTClient = AWSIoTMQTTClient("standID")
myMQTTClient.configureEndpoint("a5psath5j8lsu-ats.iot.us-east-1.amazonaws.com", 8883)

myMQTTClient.configureCredentials("/home/pi/AWSIoT/root-ca.pem",
"/home/pi/AWSIoT/private.pem.key", "/home/pi/AWSIoT/certificate.pem.crt")

```

```
myMQTTClient.configureOfflinePublishQueueing(-1)
myMQTTClient.configureDrainingFrequency(2)
myMQTTClient.configureConnectDisconnectTimeout(10)
myMQTTClient.configureMQTTOperationTimeout(5)
print ('Iniciando IoT Core Topic..')
myMQTTClient.connect()
```

```
myMQTTClient.subscribe("home/stand", 1, stand)
```

```
while True:
    time.sleep(2)
```

3.10. Código de ESP32.

A continuación, se muestra el código cargado en el ESP32 master

```
#Test de etiquetas
#include <Arduino.h>
#include <WiFi.h>
#include <TFT_eSPI.h> // Hardware-specific library
#include <SPI.h>
#include <FS.h>
#include "SPIFFS.h"

// ***** INTERNET LOCAL *****

const char* ssid = "SOL";
const char* password = "@Nth0n12017%$";
```

```

String wifi_ip = "10.30.10.201";

String wifi_mask = "255.255.255.0";

String wifi_gateway = "10.30.10.1";

String wifi_dns1 = "8.8.8.8";

String wifi_dns2 = "8.8.8.8";

String precio = "2.99";

String descripcion_part1 = "PLANTA DECORATIVA 11CM "; // No más de 23
Caracteres

String descripcion_part2 = "HOME CLUB TERRARIO"; // No más de 23 Caracteres

String codigo_producto = "0000000001"; // No más de 23 Caracteres

TFT_eSPI tft = TFT_eSPI();

int getIpBlock(int index, String str){
    char separator = '.';
    int found = 0;
    int strIndex[] = {0, -1};
    int maxIndex = str.length()-1;

    for(int i=0; i<=maxIndex && found<=index; i++){
        if(str.charAt(i)==separator || i==maxIndex){
            found++;
            strIndex[0] = strIndex[1]+1;
            strIndex[1] = (i == maxIndex) ? i+1 : i;
        }
    }
}

```

```

    }
}
return found > index ? str.substring(strIndex[0], strIndex[1]).toInt() : 0;
}

```

```

IPAddress str2IP(String str){
    IPAddress
    ret( getIpBlock(0,str),getIpBlock(1,str),getIpBlock(2,str),getIpBlock(3,str));
    return ret;
}

```

```

void setup(){
    Serial.begin(115200);
    delay(2000);
    WiFi.config(str2IP(wifi_ip),          str2IP(wifi_gateway),          str2IP(wifi_mask),
str2IP(wifi_dns1), str2IP(wifi_dns2));
    WiFi.mode(WIFI_STA);
    WiFi.begin(ssid, password);
    int wifi_cont = 0;
    while (WiFi.waitForConnectResult() != WL_CONNECTED){
        if(wifi_cont > 100){
            Serial.println("");
            Serial.println("Connection Failed! Rebooting...");
            Serial.println("Rebooting...");
            delay(500);
            ESP.restart();
        }
    }
}

```

```

    }
    Serial.print(".");
    wifi_cont++;
    delay(100);
}
if (!SPIFFS.begin(true)) {
    Serial.println(F("An Error has occurred while mounting SPIFFS"));
    return;
}
Serial.println(F("SPIFFS OK "));
tft.init();
tft.setRotation(1);

}

void loop(){
    tft.fillRect(0, 0, 480, 320, TFT_WHITE);
    drawBmp("/216573687891.bmp", 0, 0);
    tft.fillRect(230, 0, 250, 190, TFT_YELLOW);
    tft.setTextColor(TFT_BLACK, TFT_YELLOW);
    tft.setTextSize(2);
    tft.drawString("INCLUYE IVA", 330, 10, 1);
    tft.setTextSize(5);
    tft.drawString("$", 250, 70, 1);
    tft.setTextSize(8);
    tft.drawString(precio, 290, 70, 1);
}

```

```

tft.setTextSize(2);

tft.setTextColor(TFT_BLACK,TFT_WHITE);

tft.drawString(descripcion_part1, 200, 195, 1);
tft.drawString(descripcion_part2, 200, 215, 1);

tft.setTextSize(3);

tft.setTextColor(TFT_BLACK,TFT_WHITE);

tft.drawString(codigo_producto, 10, 200, 1);

delay (10000);
}

```

A continuación, se muestra el script para subir imágenes de código de barras.

```

// Bodmers BMP image rendering function

void drawBmp(const char *filename, int16_t x, int16_t y) {

  if ((x >= tft.width()) || (y >= tft.height())) return;

  fs::File bmpFS;

  // Open requested file on SD card
  bmpFS = SPIFFS.open(filename, "r");

  if (!bmpFS)
  {
    Serial.print("File not found");
    return;
  }
}

```

```

uint32_t seekOffset;
uint16_t w, h, row, col;
uint8_t r, g, b;

uint32_t startTime = millis();

if (read16 bmpFS == 0x4D42)
{
    read32 bmpFS;
    read32 bmpFS;
    seekOffset = read32 bmpFS;
    read32 bmpFS;
    w = read32 bmpFS;
    h = read32 bmpFS;

    if ((read16 bmpFS == 1) && (read16 bmpFS == 24) && (read32 bmpFS
== 0))
    {
        y += h - 1;

        bool oldSwapBytes = tft.getSwapBytes();
        tft.setSwapBytes(true);
        bmpFS.seek(seekOffset);

        uint16_t padding = (4 - ((w * 3) & 3)) & 3;
        uint8_t lineBuffer[w * 3 + padding];

        for (row = 0; row < h; row++) {

            bmpFS.read(lineBuffer, sizeof(lineBuffer));
            uint8_t* bptr = lineBuffer;
            uint16_t* tptr = (uint16_t*)lineBuffer;

```

```

// Convert 24 to 16 bit colours
for (uint16_t col = 0; col < w; col++)
{
  b = *bptr++;
  g = *bptr++;
  r = *bptr++;
  *tptr++ = ((r & 0xF8) << 8) | ((g & 0xFC) << 3) | (b >> 3);
}

// Push the pixel row to screen, pushImage will crop the line if needed
// y is decremented as the BMP image is drawn bottom up
tft.pushImage(x, y--, w, 1, (uint16_t*)lineBuffer);
}
tft.setSwapBytes(oldSwapBytes);
Serial.print("Loaded in "); Serial.print(millis() - startTime);
Serial.println(" ms");
}
else Serial.println("BMP format not recognized.");
}
bmpFS.close();
}

// These read 16- and 32-bit types from the SD card file.
// BMP data is stored little-endian, Arduino is little-endian too.
// May need to reverse subscript order if porting elsewhere.

uint16_t read16(fs::File &f) {
  uint16_t result;
  ((uint8_t *)&result)[0] = f.read(); // LSB
  ((uint8_t *)&result)[1] = f.read(); // MSB
  return result;
}

```



```

uint32_t read32(fs::File &f) {
    uint32_t result;
    ((uint8_t *)&result)[0] = f.read(); // LSB
    ((uint8_t *)&result)[1] = f.read();
    ((uint8_t *)&result)[2] = f.read();
    ((uint8_t *)&result)[3] = f.read(); // MSB
    return result;
}

```

3.11. Comando de ejecución de código

En la plataforma AWS IoT se ejecuta el siguiente código para el envío de actualización de las pantallas IoT.

```

{"id": "ESP-03", "Precio": "8.95", "Descripcion_3": "Shampoo Hombres",
"Descripcion_3": "180ml", "Codigo_barra": "/122323893997.bmp",
"Codigo_producto": "1234567890"}

```

3.12. Configuración de ejecución automática al iniciar Raspberry Pi

El stand IoT se compone en su parte principal de una Raspberry Pi que controla el envío de las actualizaciones a las pantallas IoT.

Para la ejecución del script de manera automática se configura con crontab y se agrega tarea para que cada vez que la Raspberry Pi se encienda, se pueda ejecutar el script de Python.

crontab -e

@reboot python /home/pi/myscript.py

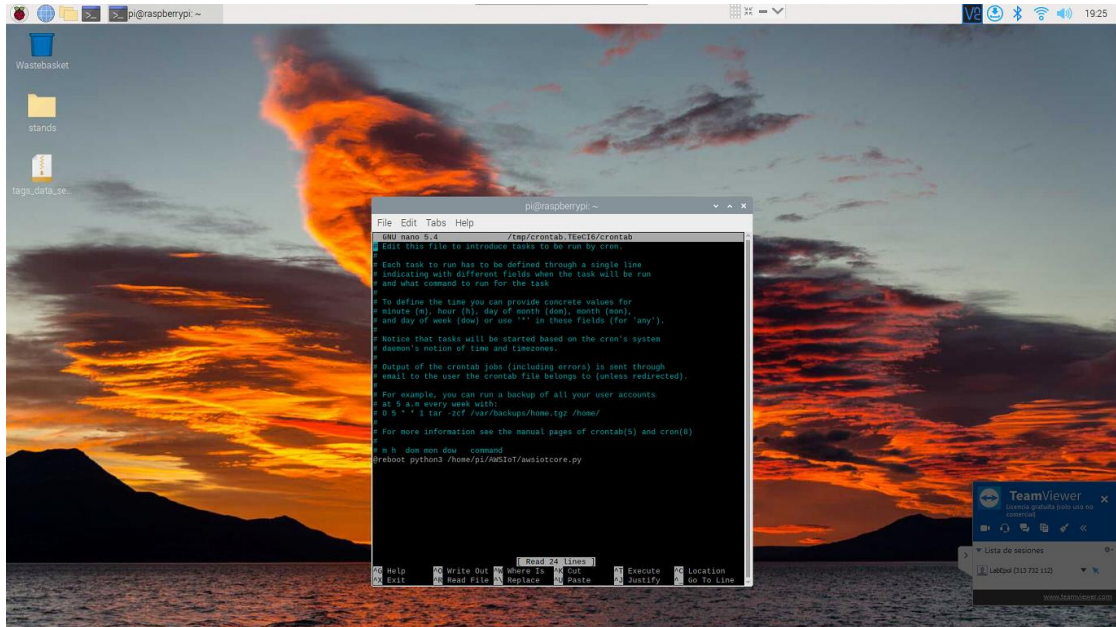


Figura 3.26 Configuraciones en crontab Raspberry Pi

En la figura 3.27 se observa las credenciales que fueron descargadas en la Raspberry Pi para la ejecución del script de Python.

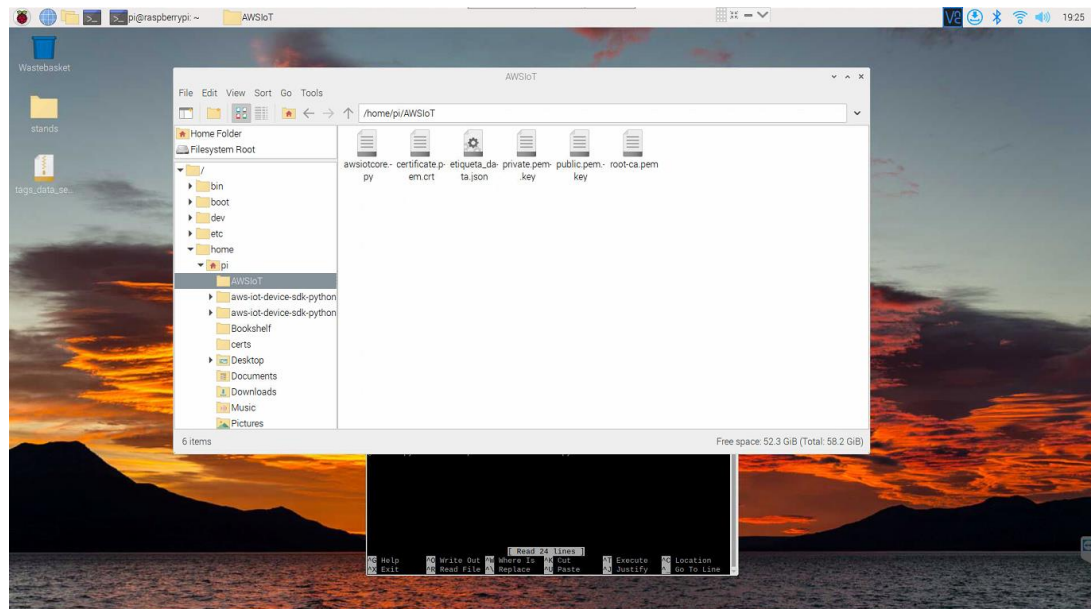


Figura 3.27 Credenciales AWS en Raspberry Pi

3.13. Configuración de AWS IoT.

En esta sección se describen las configuraciones que se realizaron en AWS IoT.

En primera instancia se debe tener una cuenta en AWS de Amazon, para esto se crea la cuenta con el usuario Anthony Navarrete. Ver figura 3.28.



The screenshot shows the AWS Billing console interface. At the top, the browser address bar displays 'console.aws.amazon.com/billing/home?#/account'. Below the address bar is a navigation bar with a 'Servicios' menu and a search bar containing the text 'Buscar servicios, características, blogs, documentos y mucho más' and a '[Alt+S]' shortcut. The main content area is titled 'Configuración de la cuenta' and is divided into two sections: 'Información de la cuenta' and 'Información de contacto'. The 'Información de la cuenta' section lists: ID de cuenta: 909256052835, Vendedor: AWS Inc., Nombre de la cuenta: anavarretem, and Contraseña: *****. The 'Información de contacto' section includes a warning: 'Tenga en cuenta que cuando modifique su información de contacto en esta página se generará un archivo PDF. Si desea actualizar la información de la dirección de facturación que aparece en su factura, ubíquese en [aquí](#).' Below this, the contact details are: Nombre completo: Anthony Navarrete, Dirección: Horizonte del Guerrero mz2074 sl 18, Ciudad: Guayaquil, Estado: Guayas, Código postal: 33166, País: EC, Número de teléfono: +593980416931, and Nombre de la empresa: (blank).

Consola de facturación

Servicios

Buscar servicios, características, blogs, documentos y mucho más [Alt+S]

Configuración de la cuenta

ID de cuenta: 909256052835
Vendedor: AWS Inc.
Nombre de la cuenta: anavarretem
Contraseña: *****

Información de contacto

Tenga en cuenta que cuando modifique su información de contacto en esta página se generará un archivo PDF. Si desea actualizar la información de la dirección de facturación que aparece en su factura, ubíquese en [aquí](#).

Nombre completo: Anthony Navarrete
Dirección: Horizonte del Guerrero mz2074 sl 18
Ciudad: Guayaquil
Estado: Guayas
Código postal: 33166
País: EC
Número de teléfono: +593980416931
Nombre de la empresa:

Figura 3.28 Cuenta de AWS

Dentro de la cuenta se debe buscar el servicio de AWS IoT para empezar con la configuración.



Figura 3.29 Servicio de AWS IoT

La figura 3.30 muestra la configuración del objeto en AWS IoT. En la sección administración seleccionar objeto. Se crea objeto con el nombre de stand.

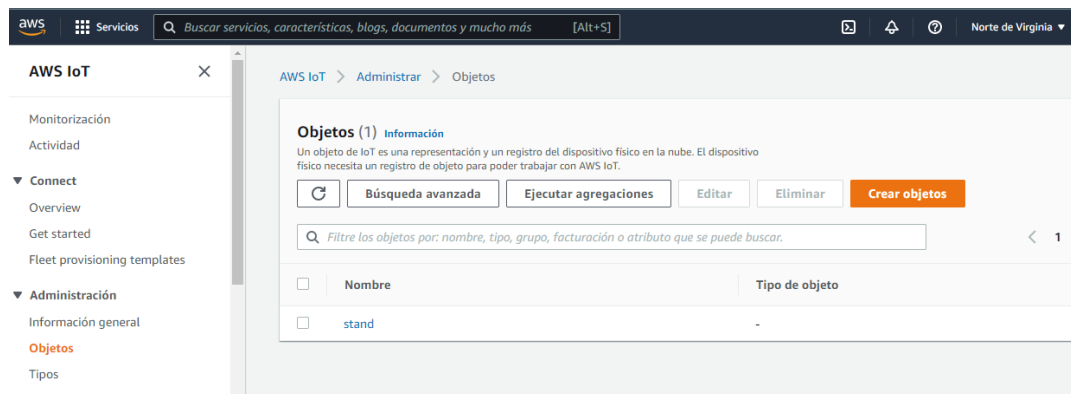


Figura 3.30 Configuración de objeto en AWS IoT

En la figura 3.31 se observa la configuración del ARN del objeto IoT, esto es único para cada objeto en internet.

The screenshot shows the 'stand' IoT object details page. At the top, there are 'Editar' and 'Eliminar' buttons. The main section is titled 'Detalles del objeto' and contains a table with the following information:

Nombre	stand	Tipo	-
ARN	arn:aws:iot:us-east-1:909256052835:thing/stand	Grupo de facturación	-

Below the table is a navigation bar with tabs: 'Atributos', 'Certificados', 'Grupos de objetos', 'Sombras de dispositivos', 'Interactuar', 'Actividad', and 'Trabajos'. The 'Atributos' tab is selected. Below the navigation bar, there is a section titled 'Atributos (0) Información' with a descriptive paragraph: 'Los atributos son pares clave-valor que permiten o no las búsquedas. Los atributos que se pueden buscar se pueden usar para filtrar listas de objetos sin usar la indexación de flotas. Los atributos que no permiten búsquedas se pueden usar para encontrar objetos, pero solo cuando la indexación de flotas está activada.'

Figura 3.31 ARN de objeto IoT

La figura 3.32 muestra el certificado del objeto que es asociado por seguridad en la cuenta de AWS.

The screenshot shows the 'Certificados' tab for the 'stand' IoT object. The 'Detalles del objeto' section at the top is identical to Figure 3.31. Below the navigation bar, the 'Certificados' tab is selected, showing 'Certificados (1) Información'. There are buttons for 'Recargar', 'Desasociar', and 'C'. A search bar contains the text 'Buscar certificados'. Below the search bar is a table with the following data:

<input type="checkbox"/>	ID de certificado	Estado
<input type="checkbox"/>	9b57119874315d28ce27a275a2a7eadf63ae1267fef95616c1235eba6c1a50...	Activo

Figura 3.32 Certificado de objeto IoT

En la figura 3.33 se observa la configuración del punto de enlace de objeto.

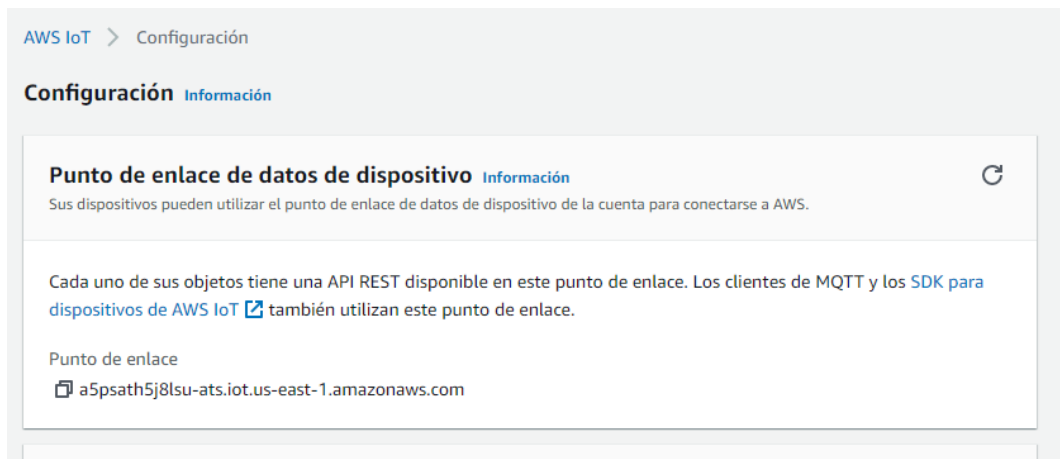


Figura 3.33 Punto de enlace de objeto

En la figura 3.34 se observa la política asociada al objeto IoT.

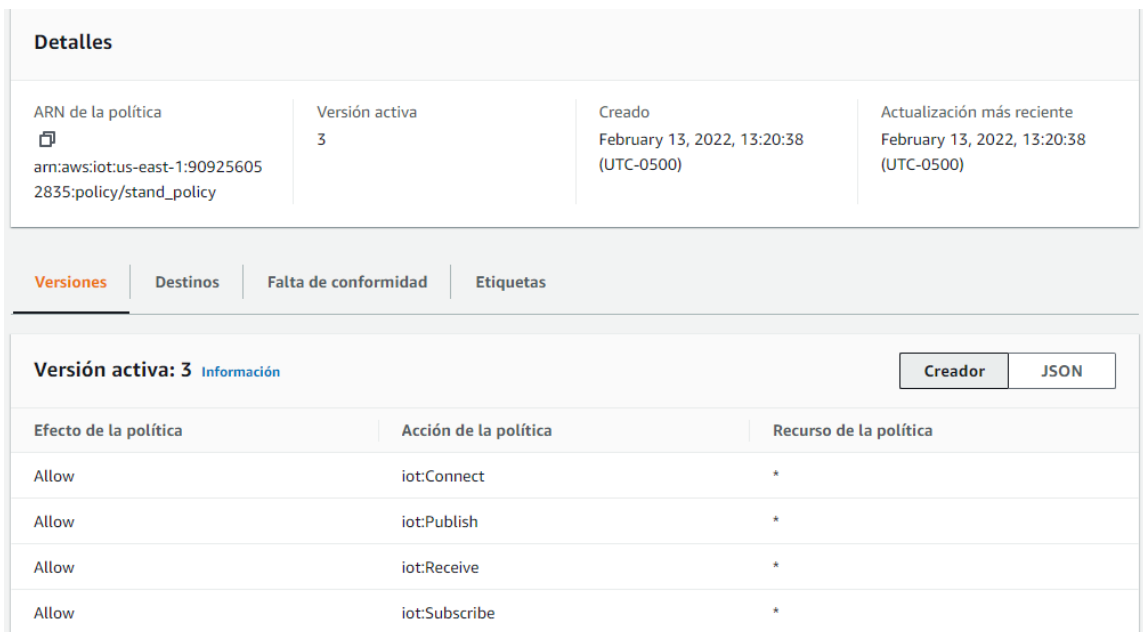


Figura 3.34 Política de objeto IoT

La figura 3.35 muestra la política activa y vigente.

Todas las versiones (3) [Información](#)

Las versiones activas y anteriores de esta política. Solo puede haber una versión activa. Una política no puede tener más de 5 versiones. Para actualizar una política con 5 versiones, primero debe eliminar una.

[🔄](#) [Eliminar](#) [Establecer como activa](#) [Editar versión](#) [Ver JSON](#)

<input type="checkbox"/>	Número de versión	Estado	Creado
<input type="checkbox"/>	3	🟢 Activa	February 14, 2022, 12:11:28 (UTC-0500)
<input type="checkbox"/>	2	⚪ Inactiva	February 13, 2022, 13:22:53 (UTC-0500)
<input type="checkbox"/>	1	⚪ Inactiva	February 13, 2022, 13:20:38 (UTC-0500)

Figura 3.35 Política activa

En la figura 3.36 se muestra la regla creada para el objeto IoT.

REGLA

stand_rule1

HABILITADAS Acciones ▾

Información general

Tags

Descripción Editar

Guardar información en base de datos de dynamoDB

Instrucción de consulta de regla Editar

El origen de los mensajes que desea procesar con esta regla.

```
select *, timestamp() AS ts from 'home/stand'
```

Uso de la versión de SQL 2016-03-23

Acciones

Las acciones son lo que ocurre cuando se activa una regla. [Más información](#)


 Dividir mensajes en varias columnas de una ... Eliminar Editar ▶
stand_table

Figura 3.36 Regla de objeto IoT

En la figura 3.37 se muestra la acción realizada en la regla.

Acciones

Las acciones son lo que ocurre cuando se activa una regla. [Más información](#)



Dividir mensajes en varias columnas de una ...
stand_table

Eliminar **Editar** ▼

Nombre de la tabla stand_table

Rol de IAM service-role/stand_rol

Figura 3.37 Acciones de regla

4. Resultados

En este capítulo se detallan los resultados obtenidos con el prototipo de stand inteligente IoT.

Luego de realizar el montaje y las configuraciones en la sección anterior, se realizan las pruebas de comunicación entre AWS y la Raspberry Pi, para esto se ejecuta en primera instancia el código de Python en la Raspberry Pi, con esto se puede enviar desde la plataforma de AWS las descripciones que se requieran cambiar en cada una de las pantallas.

En la sección de prueba se debe elegir cliente de prueba de MQTT y se selecciona la pestaña de publicar un tema.

En el tema o topic se escribe home/stand ya que el script de Python está habilitando este topic para la escritura de datos.

En la sección de carga del mensaje se especifica a que ESP32 se debe enviar la actualización de la descripción de las etiquetas.

Para el caso del ESP32 del primer nivel se escribe ESP-01, si es para el segundo nivel del stand se debe escribir ESP-02 y si es el último ESP32 se debe escribir ESP-03.

A continuación, se presentan los códigos para cada una de las descripciones de las etiquetas.

```
{"id": "ESP-01", "Precio": "2.49", "Descripcion_1": "Leche evaporada",  
"Descripcion_1": "500ml", "Codigo_barra": "/122323893997.bmp",  
"Codigo_producto": "123"}
```

```
{"id": "ESP-02", "Precio": "3.52", "Descripcion_2": "Durazno en almíbar",  
"Descripcion_2": "820G", "Codigo_barra": "/122323893997.bmp",  
"Codigo_producto": "456"}
```

```
{"id": "ESP-03", "Precio": "1.99", "Descripcion_3": "Gelatina Gelhada",  
"Descripcion_3": "400G", "Codigo_barra": "/122323893997.bmp",  
"Codigo_producto": "789"}
```

En la figura 4.1 se observa la carga del mensaje en AWS IoT sección publicar un tema.



Figura 4.1 Envío de mensajes MQTT desde AWS

En las siguientes figuras 4.2, 4.3, 4.4 y 4.5 se observa el stand con los productos y las descripciones en cada una de las pantallas.



Figura 4.2 Stand-Pantalla 1



Figura 4.3 Stand-Pantalla 2



Figura 4.4 Stand-Pantalla 3



Figura 4.5 Stand con los productos

Uno de los objetivos específicos es el desarrollo de manual de práctica de laboratorio para uso del stand IoT. La explicación y detalle de la práctica para su uso se encuentra en el anexo del libro.

Como resultado de la práctica se tiene un procedimiento para su uso el cual se resume de la siguiente manera:

- 1) Encender el prototipo IoT para la configuración de las descripciones, precios y detalles en cada pantalla.
- 2) Conectar a una red con internet el router Wifi del stand IoT.
- 3) Validar desde la Raspberry Pi que se tenga acceso a internet, para esto se debe acceder por VNC o por Putty a la ip de la Raspberry Pi, para conocer la ip de la Raspberry Pi se debe utilizar el software Advanced Ip Scanner. En caso de que no se tenga acceso por red, se debe ingresar a la Raspberry Pi con monitor, teclado y mouse y configurar la red manualmente. Por defecto la configuración está en DHCP.

- 4) Validar que desde la Raspberry Pi se tenga comunicación por red a la ESP32 master con ip: 10.30.10.201
- 5) Ejecutar código Python en la Raspberry Pi.
- 6) Validar que siempre esté en modo de ejecución el script, esperando recibir información de AWS IoT.

Configuración e ingreso a la consola de AWS.

- 1) Crear una cuenta en AWS y registrar los datos de usuario.
- 2) Dentro de la cuenta se debe buscar el servicio de AWS IoT Core para empezar con la configuración.
- 3) Realizar la configuración del objeto en AWS IoT. En la sección administración seleccionar objeto. Se crea objeto con el nombre de stand.
- 4) Realizar la configuración del ARN del objeto IoT, esto es único para cada objeto en internet.
- 5) Configurar el certificado del objeto que es asociado por seguridad en la cuenta de AWS.
- 6) Copiar la configuración del punto de enlace de objeto.
- 7) Realizar la configuración de la política asociada al objeto IoT.

- 8) Verificar la política activa y vigente.
- 9) Configurar la regla creada para el objeto IoT.
- 10) Configurar la acción realizada en la regla.
- 11) Enviar las descripciones a la Raspberry Pi mediante AWS IoT - prueba
- 12) Publicar un tema en la sección de pruebas.
- 13) Copiar la información enviada desde AWS IoT y gestionar una base de datos en DynamoDB de AWS. Para esto se debe crear una tabla en la base de datos.
- 14) Como pruebas finales se puede configurar más stands o prototipos en la misma red. Siempre y cuando se amplíen la cantidad de stands IoT.

5. Conclusiones

Las conclusiones de este trabajo de investigación son las siguientes, en base a los objetivos cumplidos:

- Se cumple con el diseño e implementación de estantería IoT para la automatización de precios y descripciones de productos comerciales utilizando una red ESP-WIFI-MESH y hardware de bajo costo. Se arma el prototipo y se prueba correcta funcionalidad.
- Se realiza el diseño de una maqueta de tres stands IoT que se interconecten mediante una red ESP-WIFI-MESH y ESP32.
- Se ejecuta la configuración de servicios web, locales mediante Raspberry PI y servidor en la nube AWS, para la administración del cambio remoto de precios, promociones y descripciones de los productos de los stands.
- Se realiza la configuración de paneles leds y ESP32 en las perchas inteligentes que visualicen precios, descripción y promociones en cada fila de la maqueta de los stands.
- Se efectuó manual de práctica del prototipo IoT de perchas inteligentes para el uso de estudiantes de la carrera de ingeniería electrónica y telecomunicaciones de la Universidad Politécnica Salesiana sede Guayaquil. Este manual está en la sección de anexos del libro.

6. Recomendaciones

Las recomendaciones de este trabajo de investigación son las siguientes:

- Se recomienda replicar el prototipo para más stands en perchas, con esto se puede cubrir más productos que tengan estas pantallas inteligentes.
- Se recomienda gestionar los productos, descripciones y precios mediante una base de datos. Para futuros desarrollos se puede integrar una base de datos como MySQL server para este fin.
- Se recomienda utilizar un router de mayor ganancia para cubrir mayores coberturas de red WIFI, en el caso de que se requiera conectar varios stands en la tienda o el comercio.
- Se recomienda utilizar servicios de AWS Greengrass para gestión de mayor cantidad de dispositivos IoT en una red local.

Bibliografía


- Arias, F. G. (2012). *El Proyecto de Investigación Introducción a la metodología científica* (Episteme, Ed.; sexta). <https://doi.org/980-07-8529-9>
- Cisco. (n.d.). *What is Wi-Fi?* Retrieved October 13, 2021, from <https://www.cisco.com/c/en/us/products/wireless/what-is-wifi.html#~resources>
- DIGITEOS. (2020). *Amazon Web Services | Cloud Computing, AWS, implementación*. <https://digiteos.com/cloud-computing/amazon-web-services/>
- ESPRESSIF. (n.d.). *ESP32 Wi-Fi & Bluetooth MCU | Espressif Systems*. Retrieved October 12, 2021, from <https://www.espressif.com/en/products/socs/esp32>
- MCI Electronics. (2017). *¿Qué es Raspberry Pi?* . <https://raspberrypi.cl/que-es-raspberry/>
- Mónica Gimenez. (2020, July 20). *Amazon Web Services (AWS): ¿qué es y qué ofrece?* - *Blog de Hiberus Tecnología*. <https://www.hiberus.com/crecemos-contigo/amazon-web-services-aws-que-es-y-que-ofrece/>
- Mouser Electronics. (2017, December 8). *Espressif Systems ESP32 Dual-Core Wi-Fi® Dual-Mode BLUETOOTH® SoC*. 2017. <https://www.mouser.ec/new/espressif/espressif-esp32-soc/>
- Navas, M. Á. (2017). *Qué es una Mesh Network o Red inalámbrica mallada*. <https://www.profesionalreview.com/2017/10/23/una-mesh-network-red-inalambrica-mallada/>
- Pedro Bertoleti. (2019). *Proyectos con ESP32 y LoRa* . Insituto NCB. <https://books.google.es/books?hl=es&lr=&id=Doi0DwAAQBAJ&oi=fnd&pg=PT3&dq=ESP32&ots=mBksCjmlkl&sig=XC0CSF3A0EEroXu7NIGX0RzLuL4#v=onepage&q=ESP32&f=false>
- PROMETEC. (2018, October). *Instalando el ESP32* . <https://www.prometec.net/instalando-esp32/>

Raspberry Pi. (2019). *Raspberry Pi 4 Modelo B* .

<https://www.raspberrypi.com/products/raspberry-pi-4-model-b/>

tic.PORTAL. (2019). *Amazon Web Services*. <https://www.ticportal.es/temas/cloud-computing/amazon-web-services#soluciones-aws>

Anexos

		GUÍA DE PRÁCTICA DE LABORATORIO
CARRERA: INGENIERÍA ELECTRÓNICA		ASIGNATURA:
NRO. PRÁCTICA:	1	TÍTULO PRÁCTICA: Configuración de prototipo IoT para estanterías automatizadas
<p>OBJETIVO GENERAL:</p> <p>Configurar dispositivo stand IoT para prácticas de estudiantes de telecomunicaciones de la UPS sede Guayaquil.</p> <p>OBJETIVOS ESPECÍFICOS:</p> <ul style="list-style-type: none"> Configurar cuenta de AWS para el envío de actualizaciones mediante la nube de internet. Configurar Raspberry Pi para la ejecución de código de Python. Probar diferentes descripciones y precios para el stand IoT. 		
INSTRUCCIONES		<ol style="list-style-type: none"> 1. Los estudiantes deben leer previamente el manual de práctica para el desarrollo. 2. Los estudiantes deben utilizar los equipos de las estanterías IoT de una responsable y calificada para evitar daños en los equipos. 3. Los estudiantes deben trabajar en grupo para el desarrollo de la práctica. 4. Se debe dejar en orden el sitio de práctica luego del desarrollo de esta.

ACTIVIDADES POR DESARROLLAR:

- 7) Encender el prototipo IoT para la configuración de las descripciones, precios y detalles en cada pantalla.



Figura 1. Stand IoT

- 8) Conectar a una red con internet el router Wifi del stand IoT.
- 9) Validar desde la Raspberry Pi que se tenga acceso a internet, para esto se debe acceder por VNC o por Putty a la ip de la Raspberry Pi, para conocer la ip de la Raspberry Pi se debe utilizar el software Advanced Ip Scanner. En caso de que no se tenga acceso por red, se debe ingresar a la

Raspberry Pi con monitor, teclado y mouse y configurar la red manualmente. Por defecto la configuración está en DHCP.

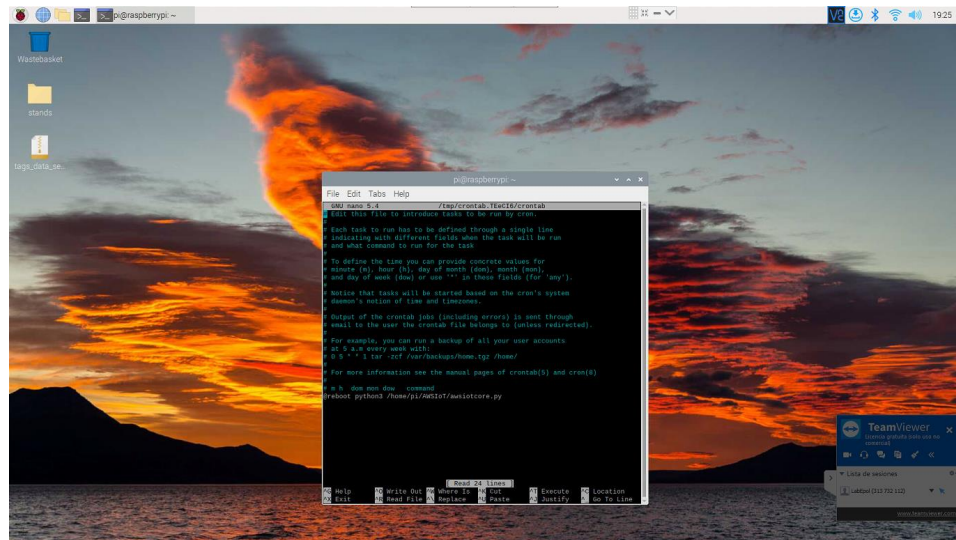


Figura 2. Acceso por VNC a la Raspberry Pi

10) Validar que desde la Raspberry Pi se tenga comunicación por red a la ESP32 master con ip: 10.30.10.201

11) Ejecutar código Python en la Raspberry Pi.

```
import websocket
```

```
import time
```

```
import _thread
```

```
from AWSIoTPythonSDK.MQTTLib import AWSIoTMQTTClient
```

```
def on_message(ws, message):  
    print(message)
```

```
def on_error(ws, error):  
    print(error)
```

```
def on_close(ws, close_status_code, close_msg):
```

```

print("### closed ###")

def stand(self, params, packet):
    print ('Mensaje recibido de AWS IoT Core')
    print ('Topic: '+ packet.topic)
    print (packet.payload)
    stand_send = packet.payload
    print (stand_send.decode("utf-8"))
    tag_data_string = stand_send.decode("utf-8")
    if __name__ == "__main__":
        print("Sending data...")
        ws = websocket.create_connection("ws://10.30.10.201:443")
        ws.send(tag_data_string)
        time.sleep(1)
        ws.close()
        print("Etiqueta Actualizada...")

myMQTTClient = AWSIoTMQTTClient("standID")
myMQTTClient.configureEndpoint("a5psath5j8lsu-ats.iot.us-east-1.amazonaws.com", 8883)

myMQTTClient.configureCredentials("/home/pi/AWSIoT/root-ca.pem",
"/home/pi/AWSIoT/private.pem.key", "/home/pi/AWSIoT/certificate.pem.crt")

myMQTTClient.configureOfflinePublishQueueing(-1)
myMQTTClient.configureDrainingFrequency(2)
myMQTTClient.configureConnectDisconnectTimeout(10)
myMQTTClient.configureMQTTOperationTimeout(5)
print ('Inicializando IoT Core Topic..')
myMQTTClient.connect()

myMQTTClient.subscribe("home/stand", 1, stand)

```

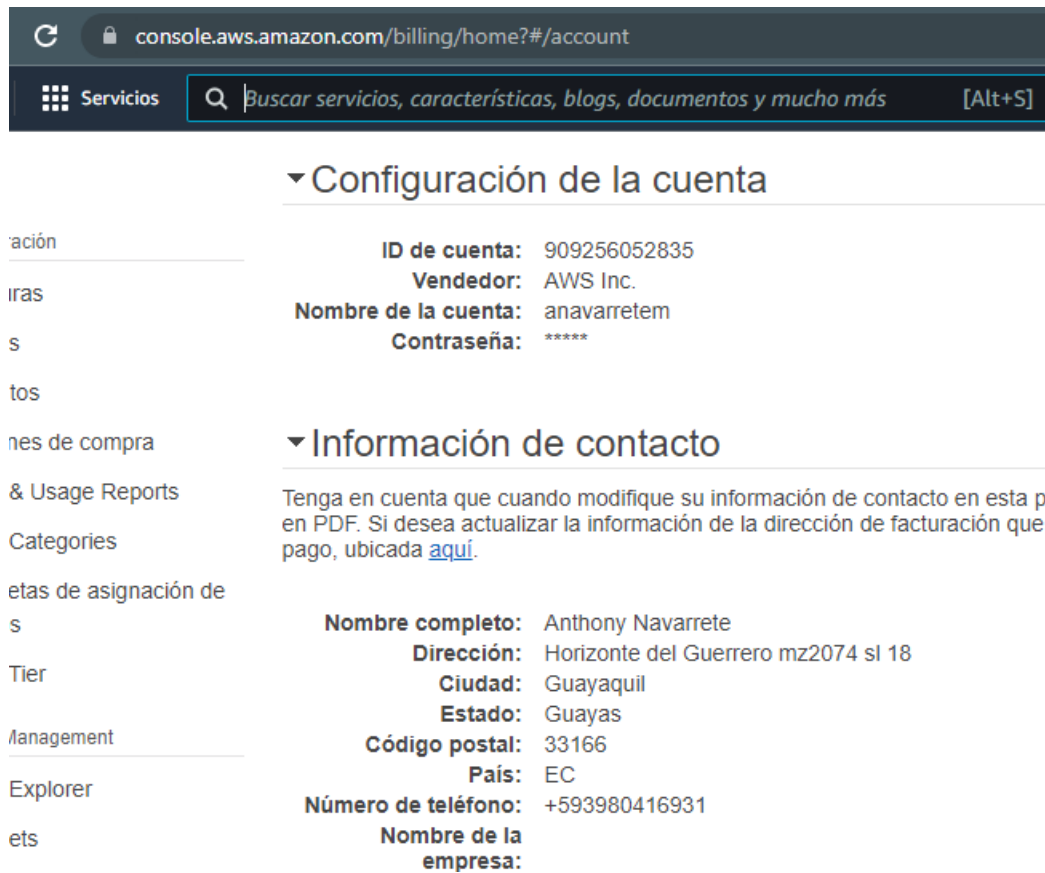
while True:

time.sleep(2)

12) Validar que siempre esté en modo de ejecución el script, esperando recibir información de AWS IoT.

Configuración e ingreso a la consola de AWS.

15) Crear una cuenta en AWS y registrar los datos de usuario.



The screenshot shows the AWS console account configuration page. The browser address bar displays 'console.aws.amazon.com/billing/home?#/account'. The page title is 'Configuración de la cuenta'. The account details are as follows:

ación	ID de cuenta: 909256052835
irras	Vendedor: AWS Inc.
s	Nombre de la cuenta: anavarretem
tos	Contraseña: *****

The page also includes a section for 'Información de contacto' with the following details:

nes de compra	Nombre completo: Anthony Navarrete
& Usage Reports	Dirección: Horizonte del Guerrero mz2074 sl 18
Categories	Ciudad: Guayaquil
etas de asignación de	Estado: Guayas
s	Código postal: 33166
Tier	País: EC
Management	Número de teléfono: +593980416931
Explorer	Nombre de la empresa:
ets	

Figura 3 Cuenta de AWS

16) Dentro de la cuenta se debe buscar el servicio de AWS IoT Core para empezar con la configuración.



Figura 4 Servicio de AWS IoT

17) Realizar la configuración del objeto en AWS IoT. En la sección administración seleccionar objeto. Se crea objeto con el nombre de stand.

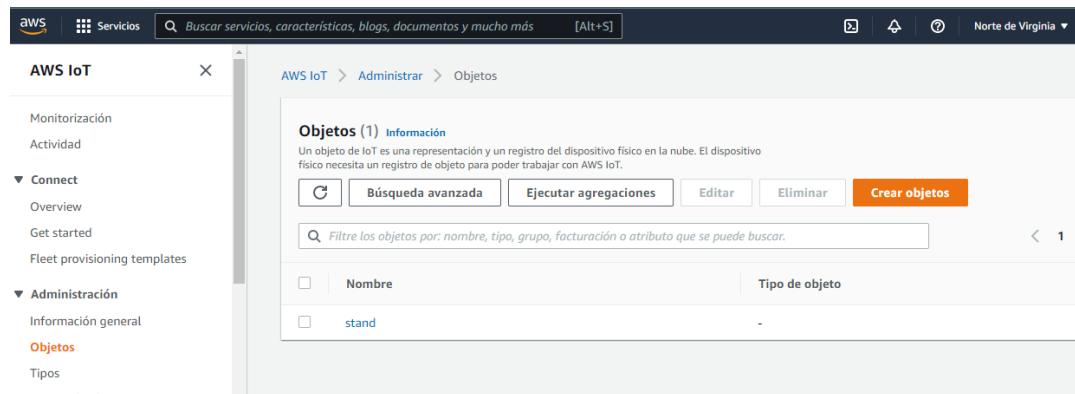


Figura 5 Configuración de objeto en AWS IoT

18) Realizar la configuración del ARN del objeto IoT, esto es único para cada objeto en internet.

stand Información Edit Eliminar

Detalles del objeto

Nombre stand	Tipo -
ARN arn:aws:iot:us-east-1:909256052835:thing/stand	Grupo de facturación -

< **Atributos** | Certificados | Grupos de objetos | Sombras de dispositivos | Interactuar | Actividad | Trabajos | >

Atributos (0) Información

Los atributos son pares clave-valor que permiten o no las búsquedas. Los atributos que se pueden buscar se pueden usar para filtrar listas de objetos sin usar la indexación de flotas. Los atributos que no permiten búsquedas se pueden usar para encontrar objetos, pero solo cuando la indexación de flotas está activada.

Figura 6 ARN de objeto IoT

19) Configurar el certificado del objeto que es asociado por seguridad en la cuenta de AWS.

Detalles del objeto

Nombre stand	Tipo -
ARN arn:aws:iot:us-east-1:909256052835:thing/stand	Grupo de facturación -

< **Atributos** | **Certificados** | Grupos de objetos | Sombras de dispositivos | Interactuar | Actividad | >

Certificados (1) Información Desasociar

Los certificados de dispositivo asociados a este recurso de objeto.

<input type="checkbox"/>	ID de certificado	Estado
<input type="checkbox"/>	9b57119874315d28ce27a275a2a7eadf63ae1267fef95616c1235eba6c1a50...	Activo

Figura 7 Certificado de objeto IoT

20) Copiar la configuración del punto de enlace de objeto.

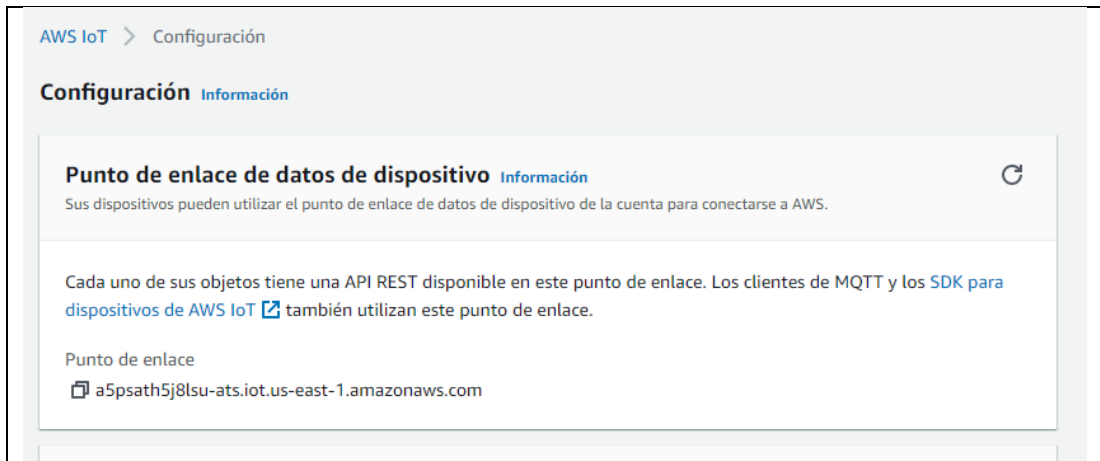


Figura 8 Punto de enlace de objeto

21) Realizar la configuración de la política asociada al objeto IoT.

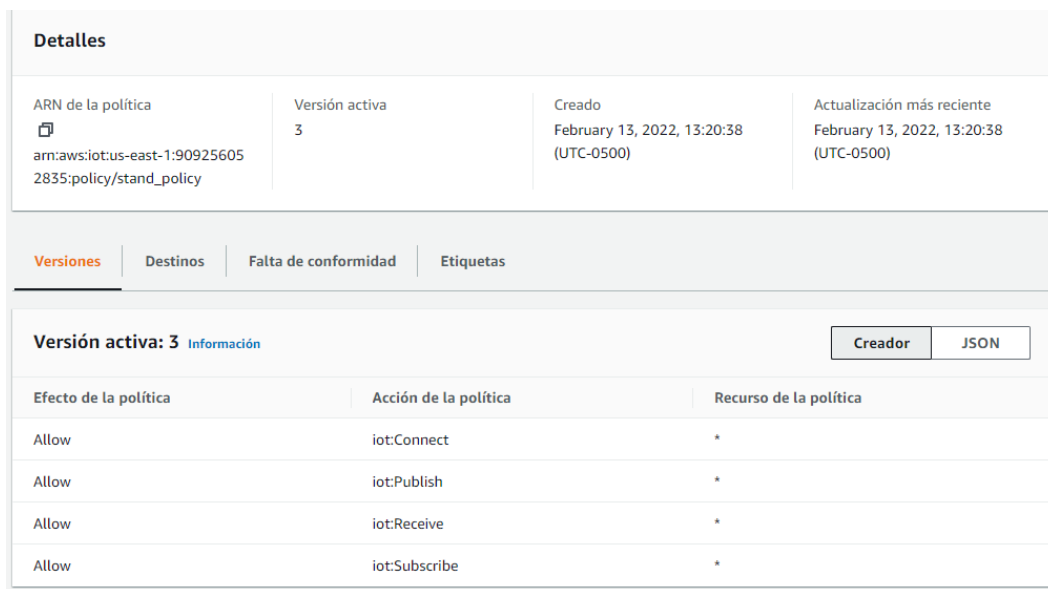


Figura 9 Política de objeto IoT

22) Verificar la política activa y vigente.

Todas las versiones (3) [Información](#)

Las versiones activas y anteriores de esta política. Solo puede haber una versión activa. Una política no puede tener más de 5 versiones. Para actualizar una política con 5 versiones, primero debe eliminar una.

<input type="checkbox"/>	Número de versión	Estado	Creado
<input type="checkbox"/>	3	Activa	February 14, 2022, 12:11:28 (UTC-0500)
<input type="checkbox"/>	2	Inactiva	February 13, 2022, 13:22:53 (UTC-0500)
<input type="checkbox"/>	1	Inactiva	February 13, 2022, 13:20:38 (UTC-0500)

Figura 10 Política activa

23) Configurar la regla creada para el objeto IoT.

REGLA

stand_rule1

HABILITADAS Acciones ▾

Información general

Tags

Descripción Editar

Guardar información en base de datos de dynamoDB

Instrucción de consulta de regla Editar

El origen de los mensajes que desea procesar con esta regla.

```
select *, timestamp() AS ts from 'home/stand'
```

Uso de la versión de SQL 2016-03-23

Acciones

Las acciones son lo que ocurre cuando se activa una regla. [Más información](#)

Dividir mensajes en varias columnas de una ...

stand_table

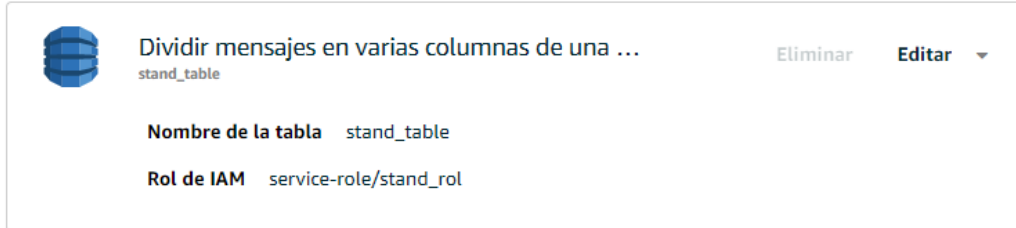
[Eliminar](#) [Editar](#) ▶

Figura 11 Regla de objeto IoT

10) Configurar la acción realizada en la regla.

Acciones

Las acciones son lo que ocurre cuando se activa una regla. [Más información](#)



The screenshot shows a configuration card for an action in the AWS IAM console. The card has a blue icon on the left and the title "Dividir mensajes en varias columnas de una ..." followed by "stand_table". On the right side, there are two buttons: "Eliminar" and "Editar" with a dropdown arrow. Below the title, there are two fields: "Nombre de la tabla" with the value "stand_table" and "Rol de IAM" with the value "service-role/stand_rol".

Figura 12 Acciones de regla

24) Enviar las descripciones a la Raspberry Pi mediante AWS IoT - prueba

```
{"id": "ESP-01", "Precio": "2.49", "Descripcion_1": "Leche evaporada",  
"Descripcion_1": "500ml", "Codigo_barra": "/122323893997.bmp",  
"Codigo_producto": "123"}
```

```
{"id": "ESP-02", "Precio": "3.52", "Descripcion_2": "Durazno en almíbar",  
"Descripcion_2": "820G", "Codigo_barra": "/122323893997.bmp",  
"Codigo_producto": "456"}
```

```
{"id": "ESP-03", "Precio": "1.99", "Descripcion_3": "Gelatina Gelhada",  
"Descripcion_3": "400G", "Codigo_barra": "/122323893997.bmp",  
"Codigo_producto": "789"}
```

25) Publicar un tema en la sección de pruebas.

The screenshot shows the AWS IoT MQTT test client interface. At the top, it says 'AWS IoT > MQTT test client'. Below that is the title 'Cliente de prueba de MQTT' with a link for 'Información'. A paragraph explains that the client is used to monitor MQTT messages and publish them. The main section is titled 'Publicar en un tema' and contains a form with the following fields:

- Nombre del tema:** A search box containing 'home/stand'.
- Carga del mensaje:** A text area containing a JSON payload: `{"id": "ESP-02", "Precio": "2034", "Descripcion_2": "ConfirmaGordo-Ahora", "Descripcion_3": "800G", "Codigo_barra": "/122323893997.bmp", "Codigo_producto": "1234567890"}`.
- Configuración adicional:** A link to expand additional settings.

Figura 13 Envío de mensajes MQTT desde AWS

26) Copiar la información enviada desde AWS IoT y gestionar una base de datos en DynamoDB de AWS. Para esto se debe crear una tabla en la base de datos.

The screenshot shows the AWS DynamoDB console interface. At the top, it says 'Tablas (3) Info'. There are buttons for 'Acciones', 'Eliminar', and 'Crear tabla'. A search box contains 'Buscar tablas por su nombre'. Below that is a table with the following columns: 'Nombre', 'Estado', 'Clave de partición', 'Clave de ordenación', 'Índices', and 'Modo de capacidad de lectura'. The table contains three rows:

Nombre	Estado	Clave de partición	Clave de ordenación	Índices	Modo de capacidad de lectura
Productos	Activo	id_stand (S)	-	0	Aprovisionado con Auto Sc
stand_tabla2	Activo	id (S)	ts (N)	0	Aprovisionado con Auto Sc
stand_table	Activo	mac_Id (S)	ts (N)	0	Aprovisionado con Auto Sc

Figura 14 Tablas de DynamoDB

Como pruebas finales se puede configurar más stands o prototipos en la misma red. Siempre y cuando se amplíen la cantidad de stands lot.

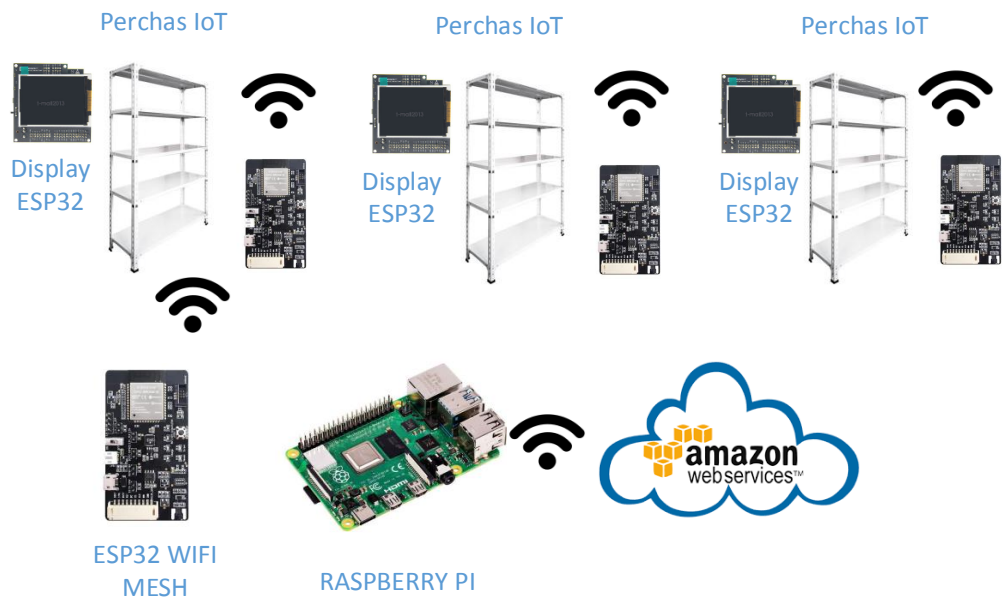


Figura 15 Stands IoT

RESULTADO(S) OBTENIDO(S):

El estudiante debe colocar los resultados obtenidos en la práctica con imágenes y descripciones de cada imagen.

CONCLUSIONES:

El estudiante debe colocar las conclusiones de las prácticas de acuerdo con los objetivos planteados.

RÚBRICA DE REVISIÓN DE PRÁCTICA:

Fecha de realización de la práctica:					
Integrantes:					
	Malo	Regular	Bueno	Excelente	Observaciones
Sustentación correcta de las prácticas. 35% del puntaje					
Eficacia, evidencia, ilustraciones y correcta organización del progreso de la práctica. 35% del puntaje					
Desenlaces de las prácticas. 30% del puntaje					
PUNTAJE:					/10