



INSTITUTO TECNOLÓGICO SUPERIOR
SUDAMERICANO
QUITO - ECUADOR

ESCUELA DE
DESARROLLO DE SOFTWARE

PROYECTO DE TITULACIÓN

TEMA:

**DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA PROTOTIPO DE
MONITOREO PARA LOS PARQUEADEROS DEL EDIFICIO MATRIZ
DEL INSTITUTO SUPERIOR TECNOLÓGICO SUDAMERICANO
QUITO**

AUTOR: REQUENA TONONY EMMANUEL ALEJANDRO

TUTOR: MSc. FABRIZIO VILLASÍS

San Francisco de Quito, noviembre del 2024

AUTORÍA

Yo, Emmanuel Alejandro Requena Tonony, portador de la cédula de identidad No. 0963189329, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional y que he consultado e investigado en base a las referencias bibliográficas que se incluyen en este documento. Esta investigación no contiene plagio alguno y es resultado de un trabajo serio desarrollado en su totalidad por mi persona.

Emmanuel Alejandro Requena Tonony

CERTIFICACIÓN

Una vez que se ha culminado la elaboración del proyecto de titulación cuyo tema es: “Diseño e implementación de un sistema prototipo de monitoreo para los parqueaderos del edificio matriz del Instituto Superior Tecnológico Sudamericano Quito”, certifico que el mismo se encuentra habilitado para su defensa pública.

MSc. Fabrizio Villasís
Coordinador de la Escuela de
Desarrollo de Software
Instituto Tecnológico Superior Sudamericano Quito

CERTIFICACIÓN

Por medio del presente certifico que el señor Emmanuel Alejandro Requena Tonony, ha realizado y concluido su trabajo de titulación, cuyo tema es: “Diseño e implementación de un sistema prototipo de monitoreo para los parqueaderos del edificio matriz del Instituto Superior Tecnológico Sudamericano Quito”, para obtener el título de Tecnólogo en Desarrollo de Software, bajo mi tutoría.

MSc. Fabrizio Villasís
Director del Proyecto de Titulación

AGRADECIMIENTOS

Este agradecimiento está dedicado, en primer lugar, a Dios, por darme la sabiduría e inteligencia de estudiar una carrera tan maravillosa, además de quien ha sido la guía y fuerza principal en mi vida. También quiero expresar mi gratitud a todas las personas que hicieron posible este proyecto. En especial, al Ingeniero Fabrizio Villasís, cuyas ideas y valiosos aportes facilitaron el progreso de este trabajo. Su paciencia y correcciones fueron fundamentales para completar con éxito el proyecto. Agradezco profundamente a mis padres por su inquebrantable apoyo y constancia durante todo el proceso, brindando no solo los recursos necesarios, sino también el tiempo para resolver dudas y atender detalles importantes en la presentación final del proyecto. Finalmente, extendiendo mi agradecimiento a toda la comunidad de Arduino y ESP8266, cuyo constante aporte ha proporcionado el material necesario y suficiente para superar cualquier desafío que surgiera durante el desarrollo del proyecto.

DEDICATORIA

Este proyecto de titulación va dedicado a mis padres principalmente por ser los pilares fundamentales dentro de mi desarrollo profesional y personal, resultado del esfuerzo y sacrificio de tanto tiempo invertido, demostrando que ha tenido sus frutos.

RESUMEN

El proyecto, titulado "Diseño e implementación de un sistema prototipo de monitoreo para los parqueaderos del edificio matriz del Instituto Superior Tecnológico Sudamericano Quito", propone la creación de un sistema automatizado de monitoreo para gestionar la disponibilidad de estacionamientos en la sede principal del Instituto Superior Tecnológico Sudamericano en Quito.

El sistema utiliza tecnologías de Internet de las Cosas (IoT), específicamente microcontroladores ESP8266 y sensores de proximidad, para monitorear en tiempo real la disponibilidad de los espacios de estacionamiento. La información recopilada se presenta a los usuarios mediante una página web personalizada, accesible desde dispositivos móviles, lo que permite una gestión más eficiente y optimizada del uso de los parqueaderos. El objetivo principal del proyecto es mejorar la eficiencia y seguridad del estacionamiento, reducir los tiempos de espera y minimizar el riesgo de accidentes al proporcionar datos precisos sobre la ocupación de los espacios.

Este trabajo destaca por su enfoque innovador, al aplicar tecnologías comúnmente utilizadas en entornos domésticos a un contexto institucional. Esto facilita una mayor personalización y flexibilidad en la gestión de los recursos del instituto. Además, el sistema automatizado no solo aborda la problemática de la limitada disponibilidad de estacionamientos, sino que también fomenta la adopción de soluciones tecnológicas avanzadas en entornos educativos.

ABSTRACT

The project, entitled "Design and implementation of a prototype monitoring system for the parking lots of the main building of the Instituto Superior Tecnológico Sudamericano Quito", proposes the creation of an automated monitoring system to manage the availability of parking spaces at the main headquarters of the Instituto Superior Tecnológico Sudamericano in Quito.

The system uses Internet of Things (IoT) technologies, specifically ESP8266 microcontrollers and proximity sensors, to monitor the availability of parking spaces in real time. The information collected is presented to users through a personalized web page, accessible from mobile devices, allowing for more efficient and optimized management of parking space use. The main objective of the project is to improve parking efficiency and safety, reduce waiting times, and minimize the risk of accidents by providing accurate data on space occupancy.

This work stands out for its innovative approach, by applying technologies commonly used in domestic environments to an institutional context. This facilitates greater customization and flexibility in the management of the institute's resources. Furthermore, the automated system not only addresses the issue of limited parking availability, but also encourages the adoption of advanced technological solutions in educational settings.

ÍNDICE

1.	Introducción	1
2.	Justificación.....	3
3.	Antecedentes	5
4.	Objetivos	7
4.1.	Objetivo General	7
4.2.	Objetivos Específicos.....	7
5.	Marco Teórico	8
5.1.	BackEnd	8
5.2.	Arduino.....	9
5.2.1.	Qué es Arduino.....	9
5.2.2.	Arduino UNO	11
5.2.3.	Arduino MEGA.....	12
5.2.4.	Arduino NANO	13
5.2.5.	Arduino IDE.....	14
5.2.6.	Módulos en Arduino.....	15
5.3.	Protocolo 802.11	17
5.3.1.	ESP8266.....	18
5.4.	Sensores de proximidad	21
5.4.1.	Sensores inductivos	22
5.4.2.	Sensores capacitivos.....	23
5.4.3.	Sensores fotoeléctricos	24
5.4.4.	Sensores magnéticos	25
5.4.5.	Sensores infrarrojos.....	26
5.4.6.	Sensores de ultrasonido	27
5.5.	Luces LED.....	29
5.6.	Módulo MB-102.....	31
5.7.	FrontEnd.....	32
5.7.1.	Página Web	32
5.7.2.	HTML.....	33

5.7.3.	CSS.....	36
5.7.4.	JavaScript	37
5.8.	Redes de computadoras	38
5.8.1.	Redes LAN.....	43
5.8.2.	Redes WAN.....	45
5.8.3.	Redes MAN.....	49
5.8.4.	La nube.....	50
5.8.5.	NGROK.....	53
5.9.	Simbología electrónica.....	55
5.9.1.	Cables Jumpers.....	56
6.	Desarrollo del Proyecto de Titulación.....	58
6.1.	Elección de las tecnologías	58
6.2.	Diseño del sistema prototipo y montaje de los módulos electrónicos.....	59
6.3.	Implementación del sistema prototipo basado en el ESP8266 en el IDE de Arduino .	91
6.4.	Desarrollo de la página web basado en HTML, CSS y JavaScript	113
6.5.	Aplicación de las distintas partes del proyecto dentro del ESP8266	126
6.6.	Pruebas de funcionamiento	141
7.	Conclusiones y Recomendaciones	147
7.1.	Conclusiones	147
7.2.	Recomendaciones.....	149
	Referencias.....	151
	ANEXOS.....	155

ÍNDICE DE TABLAS

Tabla 1. Costo de un módulo con baterías.	89
Tabla 2. Costo de un módulo con fuente de alimentación.	89
Tabla 3. Materiales y equipo necesarios para la instalación en el parqueadero del Instituto Superior Tecnológico Sudamericano.	90
Tabla 4. Costos adicionales.	90
Tabla 5. Costo total del proyecto dentro del parqueadero del Instituto Superior Tecnológico Sudamericano.	90

ÍNDICE DE FIGURAS

Figura 1. Página oficial de Arduino.	9
Figura 2. Entorno de Desarrollo Integrado de Arduino.	10
Figura 3. Imagen del chip ESP-12e impreso en el módulo NodeMCU.	21
Figura 4. Esquema de pines del sensor ultrasónico HC-SR04.	27
Figura 5. Principio de funcionamiento del sensor ultrasónico.	29
Figura 6. Ánodo y cátodo de un led.	30
Figura 7. Estructura de una red de computadoras.	40
Figura 8. Tipos de topología de red.	42
Figura 9. Capas del modelo OSI.	47
Figura 10. Imagen del funcionamiento de la nube.	51
Figura 11. Token de configuración para la herramienta de Ngrok.	54
Figura 12. Comando para realizar tunel de servicio local con el servidor de Ngrok.	54
Figura 13. Conexión establecida con el servicio local.	54
Figura 14. Jumpers macho-macho.	57
Figura 15. Jumpers macho-hembra.	57
Figura 16. Diagrama general de comportamiento del sistema prototipo.	59
Figura 17. Diagrama de comportamiento del sistema prototipo con detalle den entradas y salidas.	60
Figura 18. Diagrama de comportamiento de la entrada del sistema prototipo.	61
Figura 19. Diagrama de comportamiento del sistema prototipo modelo cliente-servidor.	63
Figura 20. Diagrama de comportamiento con la capacidad de respuesta en tiempo real de la página web con el usuario final.	65
Figura 21. Diagrama estructural del sistema prototipo.	67
Figura 22. Plano de planta junto con la distribución de vehículos dentro del parqueadero.	68
Figura 23. Distribución de los puestos dentro del parqueadero del 1 al 3 en sentido antihorario.	69
Figura 24. Distribución de los puestos dentro del parqueadero del 2 al 4 en sentido antihorario.	70
Figura 25. Distribución de los puestos 5 y 6 dentro del parqueadero en sentido antihorario. .	70
Figura 26. Distribución de los puestos 7y 8 dentro del parqueadero en sentido antihorario. ..	71
Figura 27. Ubicación del octavo puesto dentro del parqueadero junto con la entrada peatonal y caja eléctrica.	72

Figura 28. Plano de planta junto con el servidor y los módulos expresados con sensores ubicados encima de cada puesto.....	73
Figura 29. Plano de planta junto con la cometida eléctrica para el parqueadero.	74
Figura 30. Vista panorámica de todo el parqueadero.....	75
Figura 31. Diseño de plafón.	76
Figura 32. Ficha técnica del plafón.	76
Figura 33. Diseño de plafón alternativo.	77
Figura 34. Diseño de plafón moderno.	77
Figura 35. Esquema de conexión eléctrica del módulo cliente en Proteus.	79
Figura 36. Esquema de pines de energía del NodeMCU V3 ESP8266.....	80
Figura 37. Instalación del NodeMCU en el protoboard.	81
Figura 38. Esquema de conexión de leds y resistencias en el protoboard.	82
Figura 39. Conexión del módulo MB-102 en el protoboard.	83
Figura 40. Batería de 9 voltios y salida de 600 miliamperios.	85
Figura 41. Soporte con adaptador de batería de 9 voltios con salida de plug universal de 5.5 mm y 2.1 mm.....	85
Figura 42. Fuente AC DC de 9 voltios con salida de plug de 5.5 mm y 2.1mm.....	86
Figura 43. Montaje final de un módulo para parqueadero.	87
Figura 44. Caja de recubrimiento para los módulos.....	88
Figura 45. Definición de pines para el NodeMCU V3 ESP8266.....	91
Figura 46. Definición e inicialización de pines.....	92
Figura 47. Función para el funcionamiento del sensor en los pines configurados.....	93
Figura 48. Ruta para el anexo de gestor de tarjetas adicionales.....	94
Figura 49. Ubicación para adjuntar el enlace correspondiente a tarjetas adicionales distintas de Arduino.....	95
Figura 50. Ruta del gestor de tarjetas en Arduino IDE.	96
Figura 51. Instalación del gestor de tarjetas adecuado para ESP8266.	97
Figura 52. Librerías definidas para el módulo cliente.....	98
Figura 53. Librerías definidas para el servidor.	98
Figura 54. Ruta del Arduino IDE para incorporar una librería manualmente.....	99
Figura 55. Ruta para acceder al gestor de librerías de Arduino IDE.	100
Figura 56. Librería oficial de Wifi Manager.....	100
Figura 57. Página de configuración de una red Wifi por Wifi Manager.....	101
Figura 58. Reseteo de credenciales de la librería Wifi Manager.....	102

Figura 59. Página de inicio de Wifi Manager.	103
Figura 60. Programación de Access Point del servidor.	104
Figura 61. Función de conexión Wifi inicial.....	105
Figura 62. Reconexión del Wifi en caso de desconexión.	106
Figura 63. Función para el DNS local del servidor.....	107
Figura 64. Código para conexión con el servidor de forma estática.	108
Figura 65. Código de reconexión del cliente.....	108
Figura 66. Arreglos para definir la información que llevarán los paquetes UDP.	109
Figura 67. Función de envío de paquetes UDP al servidor.....	110
Figura 68. Línea que llama a la función en un tiempo definido constantemente.....	111
Figura 69. Función para leer los paquetes UDP entrantes.	112
Figura 70. Esquema de la red corporativa del INTESUD al servidor del sistema de parqueadero.	113
Figura 71. Esquema de redes Wifi que intervendrán con el servidor.....	114
Figura 72. Comunicación entre los equipos del sistema con el usuario final.	115
Figura 73. Elementos de la página web en HTML.	116
Figura 74. Identificadores dentro de las etiquetas en HTML.....	117
Figura 75. Página principal del administrador.	118
Figura 76. Código para anexar una hoja de estilos a una página HTML.....	119
Figura 77. Método para anexar una hoja de estilos a un documento HTML mediante @import	119
Figura 78. Código base de formateo de la hoja de estilos.....	121
Figura 79. Programación de estilo del botón dentro del portal web.	122
Figura 80. Programación de animaciones del botón.	123
Figura 81. Anexo de archivo externo de JavaScript.....	124
Figura 82. Variables usadas en JavaScript.	125
Figura 83. Asignación de DNS en el servidor.....	126
Figura 84. Acciones del servidor ante las solicitudes de HTTP GET/POST.....	127
Figura 85. Línea de lanzamiento de la página web del servidor.	128
Figura 86. Función que solicita la página web de inicio.	128
Figura 87. Función de la página web para solicitar los valores de los sensores al servidor. .	136
Figura 88. Código de verificación de la carga del portal web.....	136
Figura 89. Código de solicitud GET al servidor desde la página web.	137
Figura 90. Código de respuesta a la solicitud GET de la página web.....	138

Figura 91. Función que responde a la solicitud del sitio web para la distancia del sensor 2.	138
Figura 92. Página de error.	139
Figura 93. Funcionamiento del portal web en tiempo real.	140
Figura 94. Acceso a la página del servidor desde Internet.	141
Figura 95. Apartado de configuración de redes Wifi en sistema operativo Windows 10.	142
Figura 96. IP del servidor en la barra de navegación de Google Chrome.	142
Figura 97. Página de ingreso de credenciales de Wifi Manager.	143
Figura 98. Página de confirmación del servidor para las credenciales ingresadas.	143
Figura 99. Red levantada por el servidor después de establecer conexión.	143
Figura 100. Conexión del ESP8266 servidor al servicio de Ngrok para salida a Internet desde su terminal.	144
Figura 101. IP generada por el servicio de Ngrok para la salida a Internet.	144
Figura 102. Resultado y asignación del link por parte del software al realizar la conexión con el ESP8266.	145

LISTA DE ANEXOS

ANEXO 1: Comandos AT ESP-01.	156
ANEXO 2: Esquema de pines del ESP-01.....	157
ANEXO 3: Esquema de pines del ESP8266 NodeMCU V3 1.0 (ESP-12E).	158
ANEXO 4: Documentación oficial de configuración de red WIFI en el ESP8266.	160
ANEXO 5: Características físicas y comparativa entre ESP8266 y ESP32.....	161
ANEXO 6: Estructura básica de un documento HTML.....	163
ANEXO 7: Página oficial de la documentación guía del ESP8266.	164
ANEXO 8: Página web oficial del portal de GitHub para la librería Wifi Manager.	165
ANEXO 9: Sitio Web oficial de NGROK.....	166
ANEXO 10: Simbología electrónica de algunos componentes.....	167
ANEXO 11: Tabla del proceso de instalación del proyecto en el Parqueadero del Instituto	169

1. Introducción

El Instituto Superior Tecnológico Sudamericano tiene su edificio matriz ubicado en la ciudad de Quito, en la Av. 10 de Agosto N34-91 y Av. Atahualpa. Esta institución ofrece carreras profesionales de tercer nivel, como Gastronomía, Protección del Medio Ambiente, Administración de Empresas, Marketing y Desarrollo de Software.

Actualmente, el edificio matriz cuenta con un parqueadero en el subsuelo, destinado exclusivamente al personal de la institución. Sin embargo, el espacio disponible resulta limitado, ya que solo ofrece capacidad para 7 vehículos, lo que complica su organización y uso.

La principal problemática radica en la alta demanda de estacionamiento frente a la escasa disponibilidad de puestos durante la jornada laboral. Esta situación se agrava debido a que el personal administrativo, docentes e incluso algunos estudiantes utilizan medios de transporte particular. Además, el diseño del parqueadero permite el paso de un solo vehículo a la vez, utilizando la misma entrada y salida, lo que genera incomodidad, demoras y la necesidad de que los usuarios dependan de las indicaciones del guardia para conocer si hay espacios disponibles o si un vehículo está saliendo del lugar. Estas circunstancias no solo dificultan la gestión del parqueadero, sino que también incrementan el riesgo de accidentes.

Con el objetivo de solucionar esta problemática, el presente proyecto propone la implementación de un sistema automatizado para monitorear y gestionar la disponibilidad de los puestos de estacionamiento dentro del edificio. Este sistema proporcionará información en tiempo real a través de una página web personalizada, eliminando la necesidad de verificaciones manuales por parte del personal de vigilancia. De esta manera, se busca ofrecer un servicio eficiente, principalmente al personal administrativo y docente, permitiendo a los usuarios

estacionar de manera rápida y segura, reduciendo los tiempos de espera y minimizando el riesgo de accidentes dentro de las instalaciones.

El contenido del documento se organiza de manera lógica y estructurada para guiar al lector en la comprensión del proyecto. En primer lugar, se presentan la justificación, los antecedentes y los objetivos del trabajo, seguidos por un detallado marco teórico que abarca conceptos clave como las tecnologías IoT, el uso de microcontroladores ESP8266, sensores de proximidad y otros componentes electrónicos y de software. Posteriormente, se describe el desarrollo del sistema, incluyendo el diseño, la implementación y las pruebas realizadas para garantizar su funcionalidad. Finalmente, se ofrecen las conclusiones y recomendaciones, acompañadas de las referencias y anexos que complementan el estudio.

2. Justificación

El desarrollo de un sistema de monitoreo automatizado para el control de los parqueaderos disponibles en el Instituto Superior Tecnológico Sudamericano es esencial debido a la facilidad y eficacia que este tipo de solución puede ofrecer. Además, este proyecto promueve el avance tecnológico y la automatización, elementos clave en la sociedad actual, que permiten mejorar significativamente la calidad de vida al optimizar procesos cotidianos mediante el uso de tecnología avanzada.

Aunque en la actualidad existen sistemas eléctricos de estacionamiento implementados en lugares como centros comerciales, el sistema propuesto se distingue por su capacidad de personalización y flexibilidad. Diseñado específicamente para las necesidades de un grupo selecto de usuarios, este sistema puede ajustarse fácilmente gracias al uso de conexiones inalámbricas, permitiendo cambios en la ubicación u orientación de los parqueaderos sin mayores complicaciones. Asimismo, la implementación de dispositivos móviles como herramienta de acceso al sistema brinda a los usuarios la posibilidad de interactuar directamente con un portal web desde cualquier lugar con conexión a internet, mejorando la experiencia de uso y la accesibilidad.

Otra característica diferenciadora de este proyecto es la integración de conceptos de Internet de las Cosas (IoT), un campo en expansión utilizado principalmente en entornos domésticos, que aquí se traslada al ámbito de la gestión de estacionamientos. Este enfoque innovador permite que los usuarios interactúen con el sistema a través de tecnologías domóticas, aplicadas mediante software como Arduino y hardware como los microcontroladores ESP8266. Estos microcontroladores, seleccionados por ser de uso libre y ofrecer capacidades avanzadas de

conexión Wi-Fi, constituyen el núcleo del sistema, posibilitando una comunicación eficiente entre los sensores y los dispositivos móviles de los usuarios.

El desarrollo de este sistema requiere conocimientos en la creación de páginas web y su conexión con sensores mediante microcontroladores, utilizando el internet como medio para transmitir información. Este enfoque permite que los usuarios visualicen los datos generados por el sistema en tiempo real, a través de una interfaz web sencilla, intuitiva y funcional. De esta manera, se logra un entorno accesible y comprensible que facilita la interacción entre el usuario y el sistema, cumpliendo así con el objetivo de mejorar la gestión y disponibilidad de los parqueaderos en el instituto.

3. Antecedentes

La escasez de plazas de aparcamiento en zonas de alto tráfico es un problema recurrente en las grandes ciudades, incluyendo Quito. Esta situación contribuye a la congestión vehicular, una de las principales consecuencias de la insuficiencia de estacionamientos en áreas urbanas concurridas. La saturación de las vías complica el acceso a diversos sectores de la ciudad, agravando los múltiples desafíos de movilidad que enfrentan las metrópolis modernas.

Las ciudades contemporáneas funcionan como centros dinámicos donde convergen actividades comerciales, logísticas e industriales con aspectos residenciales de la vida urbana. Esta interacción continua de necesidades de movilidad y actividades económicas es crucial para mantener la competitividad de las iniciativas comerciales y garantizar una calidad de vida adecuada para los habitantes.

En las últimas décadas, el crecimiento poblacional ha incrementado significativamente los desplazamientos urbanos, tanto por el aumento del parque automovilístico como por la expansión de las actividades comerciales. Este fenómeno ha provocado una sobresaturación del tráfico en las zonas urbanas y, como consecuencia, un déficit de plazas de aparcamiento disponibles para los usuarios.

En la actualidad, existen empresas que ofrecen productos y servicios orientados a solucionar problemas de estacionamiento, como es el caso de LOGITEK, una corporación destacada en innovación tecnológica aplicada a la seguridad, movilidad vehicular y monitoreo. Uno de sus sistemas permite gestionar el acceso de vehículos a instalaciones específicas mediante un control de usuarios basado en códigos de barras únicos, registrando los horarios de entrada para calcular costos. Sin embargo, este sistema tiene limitaciones, ya que no proporciona

información sobre la disponibilidad de plazas dentro del parqueadero, lo que dificulta a los usuarios encontrar espacios y genera tráfico interno innecesario.

Por otro lado, empresas como HAÛSEN, con su producto LifeSmart, ofrecen soluciones domóticas orientadas principalmente a entornos domésticos. Este sistema utiliza asistentes de voz como Alexa, Google Home o Apple HomeKit, conectados mediante redes Wi-Fi, para controlar dispositivos en sectores específicos del hogar. No obstante, estos productos carecen de aplicaciones específicas para ámbitos como la gestión de estacionamientos en entornos institucionales o comerciales.

En el contexto urbano, tanto vehículos privados como transporte público se desplazan entre áreas periféricas y el centro financiero de la ciudad. En muchos casos, estos vehículos necesitan aparcamiento debido a las actividades que se desarrollan en esta zona, que concentra servicios administrativos, gubernamentales, bancarios, comerciales y corporativos.

Ante esta problemática, la integración de sistemas domóticos se presenta como una solución viable. Los equipos de sensores controlados desde una consola o dispositivo central permiten proporcionar información precisa sobre la disponibilidad de plazas de estacionamiento. Esto facilita a los usuarios movilizarse de manera ágil, reduciendo la incertidumbre y los tiempos perdidos en la búsqueda de espacios. Además, el uso de teléfonos móviles como herramienta de interacción con estos sistemas refuerza su utilidad en el contexto actual, donde los dispositivos móviles tienen un rol central en la vida cotidiana. Este enfoque tecnológico ya ha sido implementado exitosamente en centros comerciales de alta demanda vehicular, demostrando su potencial para mejorar la experiencia de estacionamiento en diversos entornos.

4. Objetivos

4.1. Objetivo General

Diseñar e implementar un sistema prototipo de monitoreo automatizado para identificar la disponibilidad de parqueaderos en el edificio matriz del Instituto Superior Tecnológico Sudamericano Quito.

4.2. Objetivos Específicos

1. Investigar y desarrollar el marco teórico que sustente la realización del proyecto.
2. Seleccionar y configurar los sensores y módulos electrónicos necesarios para el control del sistema, considerando las características y limitaciones del espacio disponible en el parqueadero.
3. Implementar el sistema prototipo utilizando microcontroladores ESP8266 como base tecnológica.
4. Desarrollar una página web funcional empleando HTML, CSS y JavaScript, que permita a los usuarios interactuar con el sistema.
5. Programar la integración entre los sensores, el microcontrolador ESP8266 y la página web para garantizar su funcionalidad.
6. Realizar pruebas de funcionamiento del sistema para validar su desempeño y efectividad.

5. Marco Teórico

5.1. BackEnd

El BackEnd es uno de los dos conceptos fundamentales en el mundo del desarrollo de software orientado a la web, que complementa aquellas aplicaciones y programas que facilitan el día a día. (García, 2021)

Esta área de la programación basada en web procesa la información que luego alimentará al FrontEnd de datos. Directamente es la capa de acceso a los datos de un software o de algún sistema, sensor o cualquier dispositivo físico. Hace referencia a la lógica tecnológica del funcionamiento en general de un conjunto de equipos o software de una página web quedando oculto a los ojos de los clientes. La programación de un buen BackEnd va adecuado a que tan bien este planteada la lógica, y que experiencia, ya sea positiva o negativa, tendrá el usuario. (García, 2021)

(García, 2021) afirmó lo siguiente:

El trabajo de un desarrollador BackEnd supone el dominio de términos más adecuados a lenguajes de programación que requieren una lógica, ya que es el área encargada de optimizar los recursos disponibles, seguridad del sitio y otros factores.

Se suelen utilizar frameworks de desarrollo que trabajan del lado del servidor que ayudan a la aceleración de procesos y reutilización de código obteniendo buenas prácticas y generación de código más dinámico.

Entre los lenguajes más utilizados para escribir códigos o desarrollar aplicaciones en términos funcionales con lógica de programación tenemos PHP, JavaScript, Python y Ruby. Pero existen muchos otros más como C++, Java, entre otros. Y las herramientas que se utilizan para accionar estos lenguajes son editores de código, compiladores, debuggers, gestores de bases de datos, entre otros. (“¿Qué es backend?”, párrafo 4-9)

5.2. Arduino

5.2.1. Qué es Arduino

(Yúbal, 2022) afirmó que Arduino es una plataforma de código abierto ampliamente utilizada para crear proyectos de electrónica interactiva. Desde su aparición, ha revolucionado el mundo de la electrónica y la programación, permitiendo a personas de todas las edades y niveles crear de todo, desde circuitos simples hasta complejos sistemas automatizados.

El Arduino actúa como un microcontrolador programable capaz de interactuar con su entorno a través de entradas y salidas digitales y analógicas. (Yúbal, 2022)

Ya que Arduino es una plataforma de código abierto, lo podemos encontrar en su página oficial en la sección de software, como lo muestra en la figura. (Yúbal, 2022)

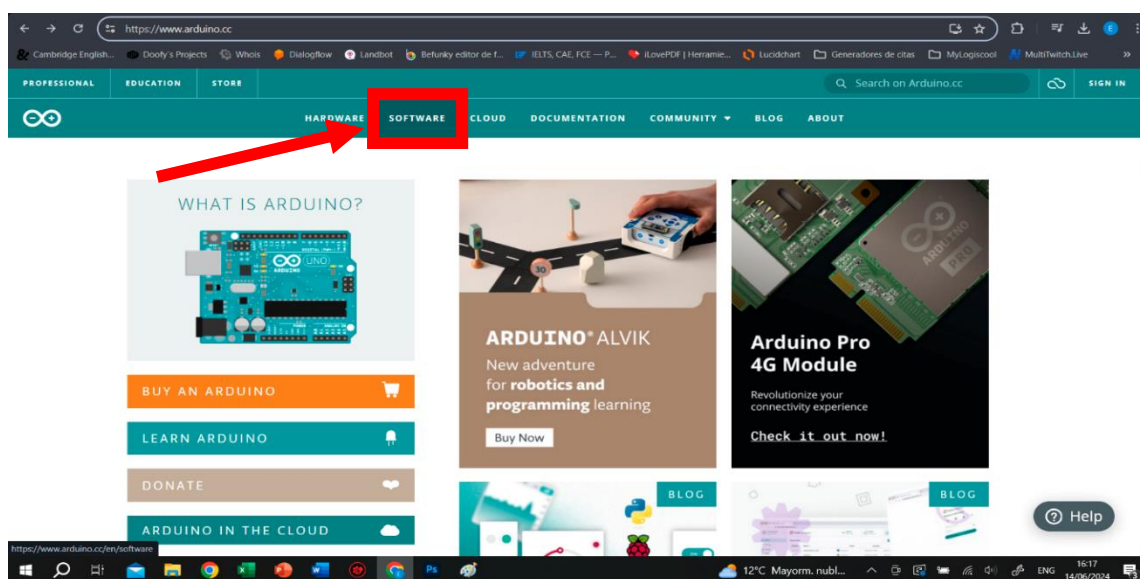
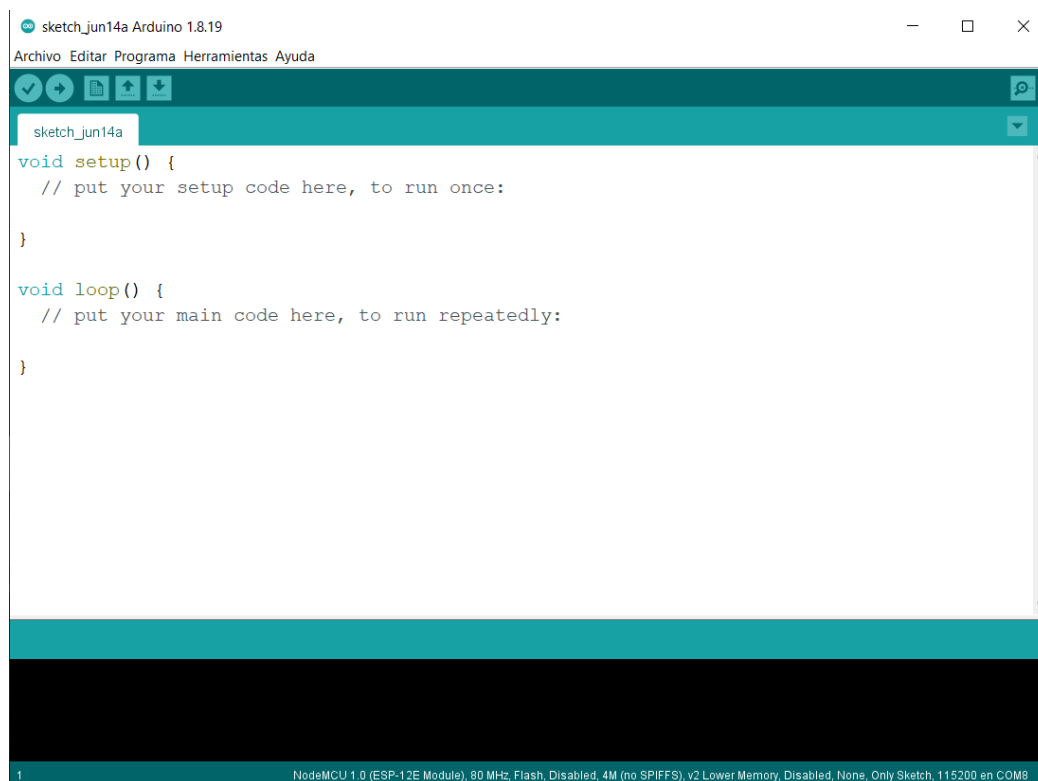


Figura 1. Página oficial de Arduino.
Fuente: El autor.

Para luego ser descargado e instalado sin otros requisitos de por medio para tenerlo disponible en cualquier ordenador. (Yúbal, 2022)

El código Arduino está escrito en un lenguaje de programación simplificado basado en C/C++. El código presentado controla la interacción del Arduino con los componentes conectados, respondiendo a diversas entradas y condiciones. La estructura básica de un

programa Arduino incluye una función de configuración (setup), en la cual se inicializan las entradas y salidas, y una función principal (loop), que ejecuta repetidamente el código principal. A continuación, se muestra un ejemplo de esta estructura en la figura adjunta. (Yúbal, 2022)

The image shows a screenshot of the Arduino IDE interface. The title bar reads 'sketch_jun14a Arduino 1.8.19'. The menu bar includes 'Archivo', 'Editar', 'Programa', 'Herramientas', and 'Ayuda'. The toolbar contains icons for opening, saving, and running. The main editor area shows the following code:

```
sketch_jun14a
void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

The status bar at the bottom indicates 'NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, Disabled, 4M (no SPIFFS), v2 Lower Memory, Disabled, None, Only Sketch, 115200 en COM8'.

Figura 2. Entorno de Desarrollo Integrado de Arduino.
Fuente: El autor.

La versión de Arduino IDE utilizada es la 1.8.19, en la actualidad pueden existir otras versiones

Donde el código escrito se ejecuta repetidamente mientras la placa esté encendida. Esto permite que Arduino realice tareas específicas automáticamente una vez que se carga el programa.

Una de las mayores ventajas de Arduino es su flexibilidad. Se puede utilizar en muchos proyectos ya que todos sus componentes, como los sensores, dependen del esquema estándar global para el desarrollo de sistemas micro controlados en Arduino, con los comandos de

programación correspondientes según el fabricante. Su versatilidad la convierte en una herramienta ideal para estudiantes, artistas, ingenieros y aficionados. (Yúbal, 2022)

5.2.2. Arduino UNO

La placa Arduino UNO consta de varios componentes importantes, incluido un microcontrolador ATmega328P, puertos digitales y analógicos y una memoria EEPROM que puede almacenar datos incluso cuando la placa está desconectada de la alimentación. Estos componentes forman la base de la funcionalidad y versatilidad de Arduino UNO en una amplia gama de aplicaciones. (Gonzalez, n.d.)

(Gonzalez, n.d.) afirmó que:

El ATmega328 tiene 32 KB (0,5 KB ocupados por el gestor de arranque, bootloader) disponibles para almacenar el programa. También tiene 2 KB de SRAM (volátil) y 1 KB de EEPROM (permanente), que se puede leer y escribir con la librería EEPROM. (“Memoria Flash, RAM y EEPROM”, párrafo 1)

(Wikimedia, Arduino Uno, 2019) afirmaron que:

Cada uno de los 14 pines digitales y 6 pines analógicos del Uno se puede usar como entrada o salida, utilizando las funciones `pinMode()`, `digitalWrite()` y `digitalRead()`. Operan a 5 voltios. Cada pin puede proporcionar o recibir 20 mA según las condiciones de funcionamiento recomendadas y tiene una resistencia de pull-up interna (desconectada por defecto) de 20-50 Kohm. Un máximo de 40 mA es el valor que no debe excederse en ningún pin de E/S para evitar daños permanentes al microcontrolador. El Uno tiene 6 entradas analógicas, etiquetadas de A0 a A5, cada una de las cuales proporciona 10 bits de resolución (es decir, 1024 valores diferentes). Por defecto, miden desde tierra hasta 5 voltios, aunque es posible cambiar el extremo superior de su rango utilizando el pin AREF y la función `analogReference()`. (“Funciones especiales de pin”, párrafo 1)

De los pines digitales, 6 de ellos brindan salida de PWM (modulación de ancho de pulso), el cual puede generar una señal analógica de salida partiendo de una fuente digital. La programación es realizada a través de un cable USB tipo B. (Wikimedia, Arduino Uno, 2019)

También posee un LED incorporado programable, entrada VIN para alimentación externa, pines de 5V y de 3.3V, pin de GND o tierra, pin de RESET para reiniciar la placa, acepta voltajes de entrada entre 7 y 20 voltios y para su funcionamiento requiere de 5 voltios, dispone de interruptores externos, Serie/UART, SPI o interfaz periférica en serie, I2C para comunicación de interfaz entre dos cables y el voltaje de referencia analógica. (Wikimedia, Arduino Uno, 2019)

Se le dio el nombre de Arduino UNO por su significado correspondiente a la primera de una serie de placas desarrolladas por conexión USB. (Wikimedia, Arduino Uno, 2019)

5.2.3. Arduino MEGA

Arduino Mega es una variante de la familia Arduino, conocida por su amplio conjunto de características y capacidades avanzadas. Esta placa se destaca por su abundancia de puertos de entrada/salida (I/O), lo que la hace ideal para proyectos que requieren numerosos sensores, actuadores o dispositivos periféricos. (MCI Electronics, 2015)

El Arduino Mega se utiliza en una amplia gama de aplicaciones, desde proyectos de hobby hasta aplicaciones industriales. Su capacidad para manejar múltiples entradas y salidas lo hace ideal para proyectos de domótica, automatización industrial, robótica y sistemas de control. (MCI Electronics, 2015)

El Arduino Mega está equipado con un microcontrolador ATmega2560, que ofrece una potencia de procesamiento significativamente mayor en comparación con otros modelos de Arduino. Con una velocidad de reloj de hasta 16 MHz, este microcontrolador es capaz de

manejar tareas complejas con facilidad. Disponiendo de 256 KB de memoria flash y 8 KB de RAM. (MCI Electronics, 2015)

Los desarrolladores tienen suficiente espacio para almacenar programas y datos, lo que permite la creación de proyectos más grandes y sofisticados. (MCI Electronics, 2015)

El Arduino Mega tiene 54 pines de entradas/salidas digitales (14 de las cuales pueden ser utilizadas como salidas PWM), 16 entradas análogas, 4 UART (puertos serial por hardware), cristal oscilador de 16MHz, conexión USB, Jack de alimentación, conector ICSP y botón de RESET. Arduino Mega incorpora todo lo necesario para que el microcontrolador trabaje; simplemente conéctalo a tu PC por medio de un cable USB o con una fuente de alimentación externa (9 hasta 12VDC). El Arduino Mega es compatible con la mayoría de los shields diseñados para Arduino Duemilanove, diecimila o UNO. (MCI Electronics, 2015, “Descripción”, párrafo 2)

5.2.4. Arduino NANO

El Arduino NANO es otra de las variantes del Arduino UNO que dispone de características útiles por su diseño compacto y versátil principalmente en proyectos donde el espacio es reducido, conservando la mayoría de posibilidades del Arduino UNO. (E. Ecda, 2023)

La placa Arduino Nano cuenta con el procesador ATmega328P, capaz de ejecutar tareas complejas a pesar de su tamaño. No te dejes engañar por su diminuto perfil; este pequeño dispositivo puede hacer maravillas. Con una memoria flash de 32 KB, 2 KB de SRAM y 1 KB de EEPROM, tiene espacio suficiente para todo tipo de proyectos, desde pequeños hasta grandes. (E. Ecda, 2023)

Cuando hablamos de conectividad, la Nano cuenta con un puerto mini-USB, pudiendo conectarla fácilmente a tu computadora para cargar programas o para alimentación, 14 pines digitales GPIO (6 de ellos pueden ser utilizados como salidas PWM) y 8 pines analógicos.

También tiene capacidades UART, SPI y I2C, lo que te permite conectarla a una multitud de sensores y dispositivos periféricos. La placa incluye un regulador de voltaje integrado que permite alimentarla con una fuente externa entre 7 y 12 voltios y una tensión promedio de 5 voltios. (E. Ecda, 2023)

Una de las ventajas de la Arduino Nano es su compatibilidad con diversas plataformas de desarrollo. Puedes usarlo con Arduino IDE o incluso con otras herramientas, lo que le da una flexibilidad sorprendente. (E. Ecda, 2023)

El software es sencillo de implementar y puede trabajar con otras placas similares debido a su software libre, bajo consumo de energía y adaptable a instalación con cualquier equipo. Una de sus desventajas más claras es la minoría de puertos de entrada y salida. (E. Ecda, 2023)

5.2.5. Arduino IDE

Arduino IDE es un entorno de desarrollo integrado para programar y cargar código para placas de microcontroladores Arduino. El software ofrece una interfaz intuitiva y fácil de usar diseñada para personas con diferentes niveles de experiencia, desde principiantes hasta desarrolladores experimentados. (Wikimedia, Arduino IDE, 2019)

Uno de los aspectos más destacables del IDE de Arduino es su interfaz sencilla y accesible. Está diseñado para ser amigable y accesible facilitando la forma de programar microcontroladores o sistemas microcontrolados, lo que permite a los usuarios escribir y probar código sin complicaciones innecesarias. “El IDE de Arduino admite los lenguajes C y C ++ utilizando reglas especiales de estructuración de códigos” (Wikimedia, Arduino IDE, 2019, p. párrafo 2), lo que proporciona una plataforma versátil para crear aplicaciones electrónicas o sistemas micro controlados.

El IDE de Arduino facilita este proceso al usuario mediante la compilación y carga de modelos. El código fuente, también conocido como "sketch", se escribe en la interfaz y se

compila antes de cargarlo en la placa Arduino mediante un cable USB. El software se encarga de configuraciones más técnicas, lo que permite a los usuarios centrarse en la lógica y la funcionalidad de sus proyectos. (Wikimedia, Arduino IDE, 2019)

La comunidad Arduino ha contribuido a la creación de una extensa biblioteca de recursos para Arduino IDE. Los usuarios tienen acceso a una variedad de ejemplos, bibliotecas y documentación para facilitar el aprendizaje y la creación de proyectos. Este ecosistema colaborativo fomenta la innovación y la experimentación, permitiendo a los usuarios compartir ideas y encontrar soluciones a problemas comunes. (Wikimedia, Arduino IDE, 2019)

5.2.6. Módulos en Arduino

Los módulos para Arduino son una herramienta extra que permite ampliar las capacidades de lo que ya podemos hacer con él. Arduino actúa como el cerebro y los distintos módulos o sensores se podrían reflejar como el resto de cuerpo y los sentidos de nuestro proyecto. Aunque eso no es todo, ya que no se puede comparar a una persona con un robot, también existen módulos en Arduino que cumplen con funciones más allá de los sentidos, como los módulos de comunicación, amplificadores, traductores y pantallas. (Solectroshop, n.d.)

Muchos módulos y placas de expansión permiten aumentar las posibilidades realizables con Arduino, pudiendo crear proyectos interesantes aplicando principalmente lógica, ya que gracias a la comunidad se han desarrollado diversidad de módulos los cuales poseen guías y manuales que ayudan al entendimiento de como conectar y el funcionamiento de los mismos. Lo mejor es que es de uso libre, así que todos los módulos tienen un estándar por el cual fueron fabricados para que sean utilizados por software y hardware de Arduino. ([GUÍA Arduino] Lista de módulos Arduino: resumen, conexión, n.d.)

Entre los módulos que existen tenemos:

- Módulos de diodo laser.

- Módulos Wifi.
- Pantallas LED y OLED.
- Módulos Bluetooth.
- Ethernet Shield.
- Módulo grabador de voz.
- Módulo lector RFID.
- Módulo reloj.
- Módulo micro SD.
- Módulo relé.
- Ring neopixel.
- Displays de 7 segmentos.
- Módulo player mini MP3.
- Módulo sensor táctil.
- Módulos sensores de proximidad.
- Mando a distancia.
- Joystick analógico.
- Motores DC.
- Servomotores.
- Módulos GSM.
- Módulo zumbador.
- Módulo sensor de humedad.
- Módulos de cámara VGA.
- Potenciómetros.
- Módulo de radiofrecuencia.
- Fuentes de alimentación DC.

- Amplificadores. ([GUÍA Arduino] Lista de módulos Arduino: resumen, conexión, n.d.)

5.3. Protocolo 802.11

El estándar 802.11 es una serie de normativas que especifican las redes inalámbricas creadas por la IEEE (Institute of Electrical and Electronics Engineers). Denominadas técnicas de modulación semidúplex. (Wikimedia, IEEE 802.11, 2004)

El primer estándar desarrollado fue el estándar 802.11-1997 y el primero adaptado con mayor presencia fue el 802.11b. Con los años y el avance de la tecnología se determinaron las versiones 802.11a, 802.11g, 802.11n, 802.11ac, entre otras. (Wikimedia, IEEE 802.11, 2004)

Los estándares 802.11b and 802.11g utilizan banda ISM (banda de radio industrial, científica y médica) pero uno de los mayores problemas es que estas bandas de frecuencia pueden sufrir de interferencias con electrodomésticos como el horno o el microondas a los dispositivos Bluetooth. Debido a esto es necesario que controlen las interferencias mediante métodos de señalización DSSS (espectro Ensanchado por Secuencia Directa) y OFDM (multiplexación por división de frecuencia ortogonal). (Wikimedia, IEEE 802.11, 2004)

En cuanto al estándar 802.11a utiliza la banda U-NII de 5GHz la cual ofrece 23 canales que no se superponen a diferencia de los estándares anteriormente mencionados que disponen de una banda de frecuencia de 2,4 GHz que solo dispone de 3 canales que no sufren de superposición. El protocolo 802.11n puede utilizar ambas bandas de frecuencia 2,4GHz y 5GHz, por otro lado, la normativa 802.11ac utiliza solo banda de 5GHz. Cabe resaltar que las radiofrecuencias empleadas por el espectro de la normativa 802.11 varía según el país. (Wikimedia, IEEE 802.11, 2004)

En resumen, los distintos estándares sirven para definir las tecnologías implementadas en equipos para las redes inalámbricas permitiendo la comunicación de ellos a través del aire.

Algunas de las características importantes que se puede mencionar es que la tecnología 802.11a no es compatible con otros protocolos de red. (Xfinity, 2024)

Los Gateway sin cables y los puntos de acceso soportan todos los protocolos requeridos por los distintos estándares (b/g/n/ac) para la conectividad de los clientes. Ya que los protocolos están disponibles dependiendo de la tecnología de los equipos que se estén conectando a estos enrutadores, proveen de la mejor banda que puedan soportar los clientes, quiere decir que poseen tanto bandas de 2,4GHz como de 5GHz. (Xfinity, 2024)

Durante el periodo de conexión, los equipos aprenden los protocolos que dispone el enrutador y seleccionan el estándar más óptimo para la comunicación de datos. Además, cuando el equipo pasa por la selección, este normalmente establece el mejor protocolo adecuado a la capacidad y velocidad. (Xfinity, 2024)

Mientras más actualizado sea el protocolo, mejor rendimiento tendrá. Como la tecnología 802.11ac dispone de una alta tasa de velocidad implementando la banda de 5GHz, es la ideal para la cobertura de señal en el lugar de residencia. Dispone de una funcionalidad llamada “beamforming” que identifica la localidad de un equipo y amplifica la señal en orientación al dispositivo. En la actualidad los equipos mayormente incorporan un conjunto de chips 802.11n/ac para disponer de un mejor desempeño. (Xfinity, 2024)

5.3.1. ESP8266

El ESP8266 es un microcontrolador para sistemas IoT con conectividad Wifi, perfecto para dispositivos portátiles y automatización del hogar. Aunque su arquitectura es simple, ofrece una serie de características destacables que lo hacen un componente atractivo para proyectos de electrónica y desarrollo de software. (Prometec, n.d.)

Este microcontrolador destacó por el motivo de ser programable como en Arduino usando su mismo IDE (Entorno de Desarrollo Integrado) pero con la posibilidad de incluir directamente

un módulo Wifi, lo que suponía una mejora ya que el Arduino clásico necesita de un Shield añadido lo que daba como resultado un aumento de costos y de dificultad, además que con el ESP8266 solo era necesario incluir la librería dentro del IDE de Arduino para poder utilizar todo su potencial al alcance del código. (Prometec, n.d.)

Entre algunas características podemos destacar:

- 30 metros de alcance teórico.
- Procesador interno de 32 bits a 80Mhz con un máximo de subida de 160Mhz
- 80kB de DRAM (El valor varía según los modelos de Esp8266).
- 35kB de IRAM, memoria rápida para el procesador (El valor varía según los modelos de Esp8266).
- Full stack TCP/IP WIFI a 2.4 GHz incluido.
- 802.11 protocol y WIFI Direct (P2P) Soft-AP (Funcionalidad de Access Point).
- Soporta antena externa para mayor rango. (Prometec, n.d.)

La familia ESP8266 se pueden encontrar diversos módulos distintos, que fueron desarrollados con el paso del tiempo según las necesidades. El ESP-01 es uno de los más populares debido a su bajo costo, aunque puede ser menos conveniente por su falta de pines GPIO además de dar la posibilidad a problemas al momento de programar que otros módulos más modernos no presentan. Otros módulos como el ESP-12 ofrecen más puertos GPIO y un puerto analógico con resolución de 10-bit. (Hernández, s.f.)

(Hernández, s.f.) afirmó que:

El ESP-201 es el favorito entre los Makers para prototipar. Pero cabe destacar que también existen módulos como el ESP-03, ESP-05, ESP-12E, entre otros.

El ESP8266 funciona entre 3V y 3,6V, aunque puede tolerar tensiones de entrada de hasta 5V. En cuanto al consumo, oscila entre 0,5 μ A cuando está apagado y 170 mA cuando está transmitiendo a máxima capacidad.

El ESP8266 tiene 17 puertos GPIO, pero solo 9 o 10 se pueden usar para propósitos generales. El GPIO16 es especial, ya que está conectado al RTC (Real Time Clock). Este microcontrolador soporta buses de comunicación como SPI, I2C y UART, y puede configurarse con resistencias pull-up o pull-down. En cuanto a los modos de operación, hay tres opciones: activo, sueño y sueño profundo. El modo activo consume más energía, pero está listo para funcionar, mientras que el modo de sueño y sueño profundo reducen el consumo de energía considerablemente. La diferencia radica principalmente en los sockets en los cuales son montados ya que pueden disponer de más o menos pines según el modelo. (Hernández, s.f.)

Para configurar un ESP-01 se puede utilizar el entorno de desarrollo integrado de Arduino. Primeramente, es necesario instalarlo desde el sitio web oficial de Arduino, y es completamente obligatorio instalar la librería particular del ESP8266, desde el buscador de librerías en Arduino es posible encontrarla, pero también existe la forma de añadirla por medio del repositorio oficial de GitHub para las distintas placas de ESP8266 y ESP32. Una vez instalado se puede configurar los puertos COM detectados por el IDE de Arduino para asignar el modelo de la placa ESP utilizada y diversas opciones como nivel de depuración, memoria flash disponible, frecuencia de la CPU, velocidad de carga de la información en el monitor serial, entre otros. (Hernández, s.f.)

Entre las tecnologías Wifi que posee esta placa tenemos el protocolo 802.11 b/g/n que operan en la misma frecuencia ya antes mencionada, y son tecnologías que fueron las últimas en desarrollarse antes de la llegada de la banda de frecuencia 5Ghz, por lo que son las más rápidas de la frecuencia anterior en orden respectivamente según su potencia, información que está inscrita encima del chip ESP-12e como lo muestra la figura. (Hernández, s.f.)



Figura 3. Imagen del chip ESP-12e impreso en el módulo NodeMCU.

Fuente: adaptado de Original file [fotografía], por Popolon, 2020, Wikimedia Commons

(https://upload.wikimedia.org/wikipedia/commons/thumb/f/f5/NodeMCU_Lua_Lol1n_V3.jpg/1200px-NodeMCU_Lua_Lol1n_V3.jpg?20201128004509). CC BY SA 4.0

Las capacidades que poseen estas placas para desarrollo son para tener en cuenta estos equipos dentro de proyectos de domótica. Es integrable con Home Smart que realiza un control de sensores midiendo presión, humedad, temperatura, asignación de colores por notas musicales, control de luces led en el hogar para ser activados vía internet. (Hernández, s.f.)

5.4. Sensores de proximidad

Un sensor de proximidad es un componente electrónico que identifica la presencia o ausencia de un objeto sin contacto físico y es capaz de detectar cuerpos, medidas, sustancias, distancias o movimientos. (2.1 Sensores industriales | Introducción a la Automatización Industrial, n.d.)

Estos detectores emiten campos electromagnéticos para capturar alteraciones o señales producidas por la presencia de objetos dentro de un rango específico. Luego, esta información

se convierte en señales eléctricas, lo que proporciona datos valiosos para los sistemas de control y automatización. (2.1 Sensores industriales | Introducción a la Automatización Industrial, n.d.)

La detección de un objeto se encuentra influenciado por una serie de factores, tales como la composición material del objeto en cuestión o la distancia a la que debe estar para su detección. Además, existen distintos tipos de sensores que se adecuan a los escenarios planteados. (2.1 Sensores industriales | Introducción a la Automatización Industrial, n.d.)

5.4.1. Sensores inductivos

Los detectores de proximidad inductivos identifican la presencia de elementos conductores (es decir, metálicos) sin necesidad de tener contacto físico y su alcance de detección varía según el tipo de metal detectado. (Pini, 2021)

Estos dispositivos operan con un campo magnético de alta frecuencia generado por una bobina en un circuito oscilante. Un objeto conductor que se aproxima al campo magnético experimenta una inducción o corriente de Foucault en él, generando así un campo magnético contrario que reduce la inductancia del sensor inductivo de manera efectiva. (Pini, 2021)

Los sensores inductivos de proximidad se basan en dos métodos de funcionamiento. En el primer método, a medida que el objetivo se aproxima al sensor, la corriente de inducción aumenta, lo que provoca un incremento en la carga en el circuito de oscilación y resulta en una disminución o detención de su oscilación. (Pini, 2021)

Una alternativa en el esquema de funcionamiento implica un cambio en la frecuencia en lugar de en la amplitud de la oscilación, debido a la presencia de un objetivo conductor. Un objeto de metal no ferroso, como el aluminio o el cobre, al acercarse al sensor, provoca un aumento en la frecuencia de oscilación, mientras que un objeto de metal ferroso, como el hierro o el acero, causa una disminución en la frecuencia de oscilación. La variación en la frecuencia

de oscilación respecto a una frecuencia de referencia conduce a un cambio en el estado de salida del sensor. (Pini, 2021)

El sensor inductivo se compone de un circuito oscilador LC, un rectificador para transformar la corriente alterna en continua, y un comparador que genera dos tensiones diferentes según la presencia o ausencia del objeto. El fabricante establece la distancia máxima teórica a la que el sensor puede detectar el objeto de referencia. Por lo general, la distancia de detección de estos sensores es limitada, oscilando entre 1 y 50 mm. La distancia a la que el objeto es detectable está influenciada por la permeabilidad del material; cuanto menor sea la permeabilidad, menor será la distancia de detección. Estos sensores presentan un rendimiento deficiente con materiales paramagnéticos (con permeabilidad magnética relativa superior a la unidad), e incluso pueden no detectar objetos si son diamagnéticos (con permeabilidad relativa inferior a la unidad). (2.1 Sensores industriales | Introducción a la Automatización Industrial, n.d.)

5.4.2. Sensores capacitivos

Los sensores de proximidad capacitivos pueden detectar objetivos metálicos y no metálicos en forma de polvo, granulados, líquidos y sólidos. El dispositivo es generalmente similar a un sensor inductivo. Se utilizan principalmente para detectar niveles de líquidos en tanques de almacenamiento. (2.1 Sensores industriales | Introducción a la Automatización Industrial, s.f.)

Estos sensores se basan en la interacción entre el objeto a detectar y el campo electrostático generado por el propio sensor. El funcionamiento es similar al caso del inductor, pero ahora el sensor es un condensador. La capacitancia del condensador depende de la distancia entre los electrodos, su área y la constante dieléctrica, que representa la capacidad del material para polarizarse en presencia de un campo eléctrico. (2.1 Sensores industriales | Introducción a la Automatización Industrial, s.f.)

La constante dieléctrica es igual al producto de la permitividad del vacío y la permitividad relativa del material dieléctrico que separa las placas del condensador. (2.1 Sensores industriales | Introducción a la Automatización Industrial, s.f.)

Cuando un objeto se acerca al sensor y entra en el campo electrostático de los electrodos, la capacidad del condensador aumenta. Los cambios de capacitancia son detectados por el circuito oscilador al que pertenecen. A medida que aumenta la capacidad, aumenta la amplificación del oscilador, lo que hace que el oscilador entre en un estado de oscilación. Cuando la amplitud de oscilación excede un cierto nivel en presencia de un objeto, el estado del sensor cambia. Si el objeto se aleja, la amplitud del oscilador disminuye hasta que cambia a su estado original. El sensor también consta de un circuito rectificador, un comparador y una etapa de salida. (2.1 Sensores industriales | Introducción a la Automatización Industrial, s.f.)

La posibilidad de detección dependerá de los valores de la constante dieléctrica, siendo directamente proporcionales con la facilidad y la detección, si su valor es más alto, mayor facilidad de detección tendrá y si su valor es más bajo, más difícil será de detectar ese objeto. Por ejemplo, el agua tiene una constante equivalente a 80, quiere decir que el sensor podrá detectar la presencia del elemento, a diferencia del aire que tiene una constante de 1, lo que indica que no será sensible a la detección del mismo. También es posible detectar materiales con altas constantes dieléctricas que estén a través de envases o paredes mientras su constante sea menor. (2.1 Sensores industriales | Introducción a la Automatización Industrial, s.f.)

5.4.3. Sensores fotoeléctricos

Los sensores fotoeléctricos operan detectando la presencia de un objeto mediante fenómenos asociados a la luz. Cada sensor cuenta con un emisor que emite un haz de luz, ya sea en el espectro visible o infrarrojo. (2.1 Sensores industriales | Introducción a la Automatización Industrial, n.d.)

Por lo general, el emisor envía una señal de luz modulada, lo que significa que la fuente de luz recibe pulsos de corriente que generan pulsos de luz. Este método permite aplicar una potencia instantánea mayor a la fuente de luz, lo que facilita la amplificación de la señal alterna en el receptor, resultando en un sensor con un alcance más extenso. Además, el emisor está equipado con una lente que ayuda a obtener una luz colimada en su salida. (2.1 Sensores industriales | Introducción a la Automatización Industrial, n.d.)

La fuente de iluminación es un diodo de infrarrojos hecho de arseniuro de galio, o de luz visible mediante Leds (Light Emitting Diode) de color verde o rojo. (2.1 Sensores industriales | Introducción a la Automatización Industrial, n.d.)

Este sensor también cuenta con un receptor que puede recibir el haz de luz emitido o no, dependiendo de la presencia del objeto, o recibirlo de manera modificada. Estos sensores se pueden clasificar según el método de detección utilizado y la disposición de los elementos emisor y receptor. (2.1 Sensores industriales | Introducción a la Automatización Industrial, n.d.)

El receptor generalmente incorpora una lente en su entrada para enfocar el haz de luz emitido en un componente fotosensible, como un fotodiodo o un fototransistor, cuyos parámetros se ven alterados por la incidencia de la luz. Además, puede filtrar y amplificar la señal en función de la frecuencia de los pulsos generados por el emisor para descartar la luz que no proviene de él. (2.1 Sensores industriales | Introducción a la Automatización Industrial, n.d.)

Estos sensores pueden activarse en presencia de luz o cuando no la reciben. (2.1 Sensores industriales | Introducción a la Automatización Industrial, n.d.)

5.4.4. Sensores magnéticos

Los sensores magnéticos se utilizan para medir la posición y velocidad de objetos metálicos en movimiento. Este tipo de sensores puede ser dispositivos activos, como los sensores de efecto Hall, o pasivos, como los sensores de reluctancia variable. El sensor de reluctancia mide

los cambios en la misma en donde el magnetismo es análogo a la resistencia eléctrica de un circuito y consta de un imán permanente, una pieza polar la cual recibe una polaridad y una bobina de detección que permanece cubierta dentro de una caja cilíndrica. (Pini, 2021)

Un cuerpo ferromagnético que pase cerca del polo provoca una variación en el campo magnético. Este cambio genera una tensión de señal en la bobina. La magnitud de la señal dependerá del tamaño del objetivo, de su velocidad, y la distancia entre la pieza polar y el objeto en cuestión. Los objetos deben de estar en movimiento para ser detectados por el sensor RV. (Pini, 2021)

Estos sensores RV son pasivos debido a que deben estar conectados a una fuente de alimentación, normalmente su aplicación está en la medición de máquinas que tienen rotación, como por ejemplo detectar el paso de los dientes de un engranaje dentado, también pueden ser vistos para la detección de cabezas de tornillos o cualquier otro elemento en un movimiento rápido. (Pini, 2021)

Son empleados por medio de tacómetros para medir la velocidad de rotación o también son aplicados en parejas para medir la excentricidad del eje giratorio. (Pini, 2021)

5.4.5. Sensores infrarrojos

Los sensores infrarrojos son capaces de medir la radiación electromagnética infrarroja que emiten los cuerpos dentro de su campo de visión. Son principalmente usados para medir temperatura de los objetos, además nos brindan la posibilidad de atravesar objetos opacos para la luz visible. (Securitas Direct, 2018)

Están diseñados para la detección de figuras, clasificación y posicionamiento de objetos, diferencias de superficie y colores, incluso sometido en condiciones ambientales extremas. (Securitas Direct, 2018)

Entre alguna de sus aplicaciones en la vida podemos encontrar la detección de fugas de gas, movimiento, inundaciones, en electrodomésticos como el microondas para distribuir el calor en el interior de forma uniforme, sirven para el control climático ahorrando energía y beneficiando a la ralentización del cambio climático. (Securitas Direct, 2018)

5.4.6. Sensores de ultrasonido

Los sensores de ultrasonido son los más utilizados para la detección de vehículos, detectando objetos de cualquier tipo a varios metros de distancia. (2.1 Sensores industriales | Introducción a la Automatización Industrial, s.f.)

Su funcionamiento está basado en el envío de señales sonoras que superan el rango audible por el ser humano (20KHz), estos sensores suelen operar en un rango de frecuencias de entre 20KHz - 250KHz. (2.1 Sensores industriales | Introducción a la Automatización Industrial, s.f.)

El ultrasónico está constituido por un emisor y un receptor de ultrasonido que utiliza como medio de transmisión el aire y su esquema de pines está representado como se muestra en la figura. (2.1 Sensores industriales | Introducción a la Automatización Industrial, s.f.)

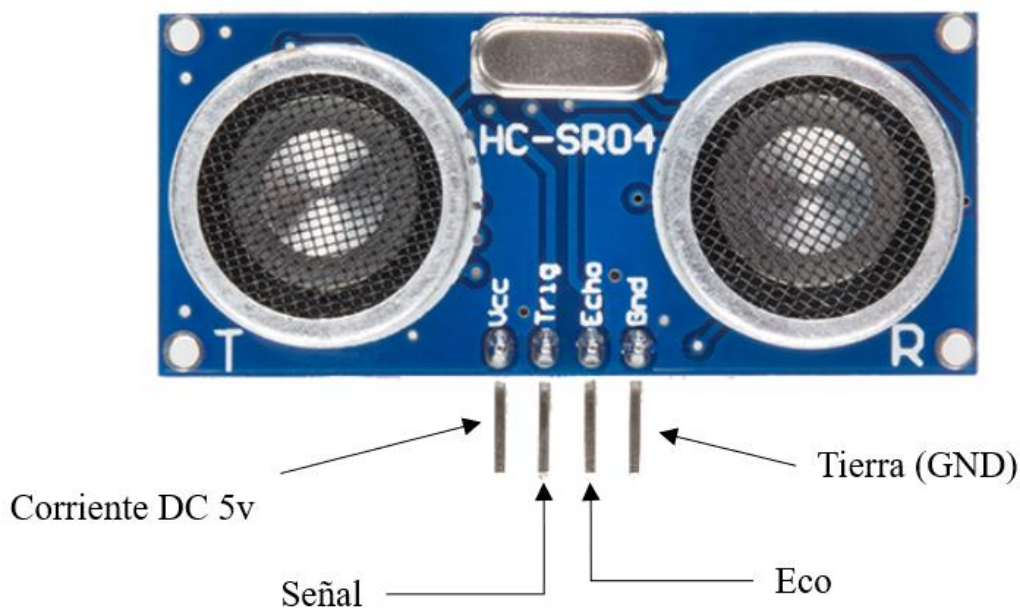


Figura 4. Esquema de pines del sensor ultrasónico HC-SR04.

Fuente: Adaptado de Ultrasonic Distance Sensor - HC-SR04 [fotografía], 2019, Flickr (<https://flic.kr/p/2gNszio>). CC BY 2.0

Utiliza el efecto piezoeléctrico realizando una presión sobre el material haciendo que se produzca un movimiento de cargas generando una diferencia de potencial entre las caras del material, el receptor está basado en la deformación que producen las señales de presión del aire.

(2.1 Sensores industriales | Introducción a la Automatización Industrial, s.f.)

Un tipo de sensores ultrasónicos son los detectores de eco. Para entender el principio de este sensor debemos tomar el tiempo que pasa desde el impulso transmitido por el emisor hasta el reflejo recibido por el receptor, representa el tiempo de vuelo que indica desde la posición del sensor hasta el objetivo y de regreso. Esto nos permite conocer la velocidad de propagación y el tiempo de vuelo dando como resultado la posibilidad de calcular la distancia entre el sensor y el objeto. (Pini, 2021)

Gracias a la alta capacidad de detectar cuerpos como líquidos, materiales de distintas formas, sólidos e incluso colores que tenga unas características mínimas de reflexión de ultrasonidos. Y por medio de un circuito electrónico se mide el tiempo del eco, por la conocida velocidad del sonido en el aire de unos 340m/s. (2.1 Sensores industriales | Introducción a la Automatización Industrial, s.f.)

Las limitaciones de este sensor son la dependencia del medio ambiente debido a la influencia ejercida en la velocidad del sonido y su carácter paraxial, quiere decir que, la trayectoria de las ondas ultrasónicas debe ser perpendiculares al sensor. (2.1 Sensores industriales | Introducción a la Automatización Industrial, s.f.)

Visto desde este punto el funcionamiento del sensor es relativamente sencillo según la aplicación que se persigue, tomando en cuenta las desventajas que tiene detectando objetos que estén perpendicularmente a él, como se muestra en la figura 1. (2.1 Sensores industriales | Introducción a la Automatización Industrial, s.f.)

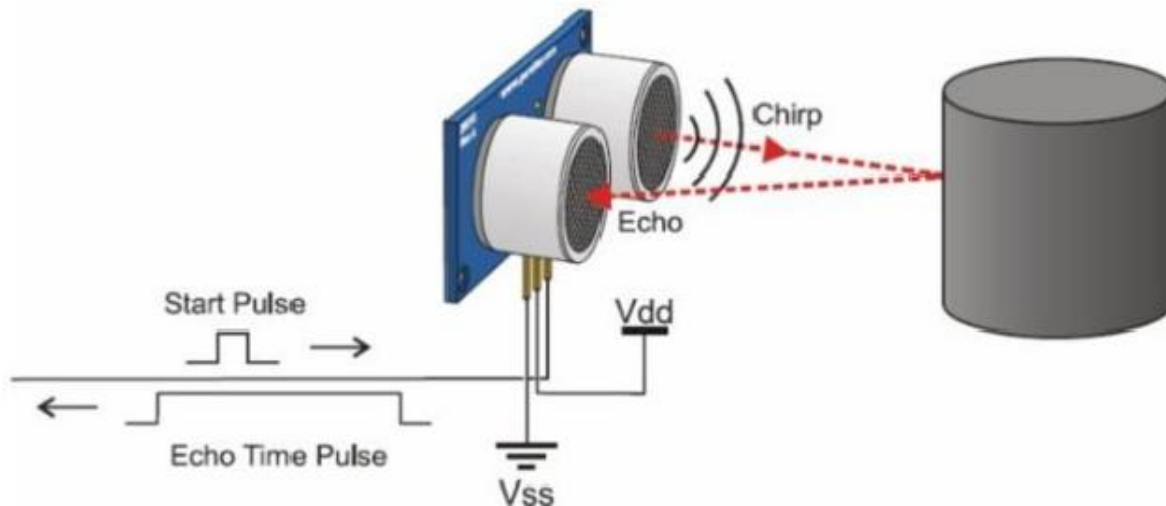


Figura 5. Principio de funcionamiento del sensor ultrasónico.

Fuente: adaptado de Working principle of ultrasonic sensor [dibujo], por Vando Gusti Al Hakim Nurhening Yuniart, 2024, ResearchGate

(<https://www.researchgate.net/publication/378534373/figure/fig4/AS:11431281226227599@1709126592377/Working-principle-of-ultrasonic-sensor-22.png>). CC BY NC ND 4.0

5.5. Luces LED

El acrónimo LED proviene de Light Emitting Diode o lo que se traduce como Diodo Emisor de Luz. Es un cuerpo semiconductor sólido resistente, cuando recibe una corriente eléctrica de muy baja intensidad, este emite luz con un muy alto rendimiento. (Villagran, 2023)

Esta tecnología de iluminación cambió totalmente la forma de ver el mundo, en comparación con su antecesor la bombilla incandescente creada por Edison, los LED poseen una vida útil hasta 30 veces más que las bombillas incandescentes con un promedio de funcionamiento de entre 30.000 a 50.000 horas. (Villagran, 2023)

La estructura básica de un LED se compone en un material que actúa como emisor semiconductor, montado en un chip reflector que especifica el color de la luz. Un ánodo y un cátodo que permiten la circulación de la corriente a través de él, dando la característica de polaridad al diodo como se muestra en la figura. (Villagran, 2023)

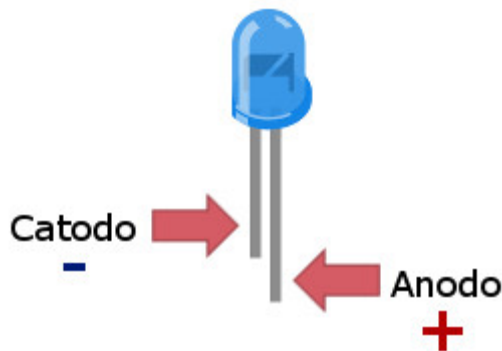


Figura 6. Ánodo y cátodo de un led.

Fuente: adaptado de Diodo_Led [dibujo], por DaviBosco04, 2020, Wikimedia Commons (https://upload.wikimedia.org/wikipedia/commons/1/16/Diodo_Led.png). CC BY SA 4.0

Un cable conductor que une los dos polos y, por último, un lente que protege al emisor del LED y determina el haz de la luz. (Villagran, 2023)

Este tipo de iluminación tiene diversas características que lo posicionan en lo mejor para crear luz, por ejemplo, estos LED no producen un rango de frecuencia ultravioleta ni infrarrojo, solo en un rango visible. Esto podría suponer un problema debido a que necesita mecanismo de conducción y convección para ser activado. La temperatura a la cual estén sometidos y la intensidad de corriente influyen ampliamente en su vida útil, en una temperatura ambiente de 30 grados centígrados puede provocar una disminución del 75% de su funcionalidad. (Villagran, 2023)

Aunque son semiconductores relativamente delicados, tienen los mejores valores referenciales de eficiencia energética en el mercado, entre 70 a 90 lumen/vatio en comparación con los 16 que tienen las lámparas halógenas. Esto es debido a su inexistencia de reflectores y direccionalidad que minimiza las pérdidas. Entre otras características, físicamente son de reducido tamaño y son fabricados de distintos colores, no emiten calor ni campos magnéticos. (Villagran, 2023)

El principio que siguen los LED para emitir luz es llamado electroluminiscencia, el cual establece que existen fotones que son emitidos para determinar el color de la luz que se observa.

El color de la luz irá acorde de los materiales que componen las capsulas que están en el interior de estos diodos. (Villagran, 2023)

La electroluminiscencia es el principio fotoeléctrico basado en la existencia de ciertos materiales que al ser alterados por una corriente eléctrica pueden generar luz. (Villagran, 2023)

Para que el proceso se cumpla es necesario energizar y otorgar una polaridad al diodo. Al desprenderse los electrones se produce el fotón que genera la luz que se percibe. (Villagran, 2023)

5.6. Módulo MB-102

Es un módulo que permite la alimentación de un protoboard de dos maneras distintas, la primera y más recomendable, mediante una fuente externa a través de un plug de conexión universal de 5.5 milímetros y 2.1 milímetros del cual puede recibir corriente directa (DC) entre 6.5 a 12 voltios, y la segunda forma por USB tipo A, este USB funciona tanto para salida como entrada de energía, como salida funciona en dado caso que se desee dar una fuente a un equipo electrónico externo del protoboard como lo puede ser un Arduino UNO, y de entrada igualmente para alimentar a los equipos conectados al protoboard, en ambos casos este conector USB entrega una alimentación de 5 voltios, en el caso donde el plug de 5.5 mm esté conectado y el USB también, el USB funcionara como salida únicamente. Posee una protección contra cortocircuito junto con un LED indicador que se enciende cuando es energizado correctamente, pero este se apaga en caso de cortocircuito para proteger los equipos y también dispone de un interruptor que permite o corta el flujo de corriente. (Módulo de Alimentación para Protoboard 5 V y 3.3 V MB102, n.d.)

Tiene dos salidas de voltaje fijas de 3.3V y otra de 5V junto con una intensidad de corriente máxima de 700 mA (miliamperios). Tiene dos rieles de salida donde se puede adecuar el voltaje

acorde a la necesidad, gracias a los jumpers se puede colocar la salida en 3.3V, 5V y 0V para deshabilitar dicho riel. (Módulo de Alimentación para Protoboard 5 V y 3.3 V MB102, n.d.)

5.7. FrontEnd

El FrontEnd es el área de desarrollo web que se centra en la parte delantera de una página web, en resumidas palabras, es el diseño de una página incluyendo estructurado de diversos elementos en la página, el diseño completo con los estilos respectivos, colores, tamaños, fondos y animaciones. (García, 2021)

Esta etapa de desarrollo es la que comunica al usuario con la aplicación y representa todo el código ejecutado en un navegador web del usuario, además, es lo que permite la interacción con el mismo. Todo lo que compone la experiencia de uso del individuo en una página web se conoce como experiencia del cliente. (García, 2021)

5.7.1. Página Web

Las páginas web son documentos electrónicos de los cuales se acceden mediante internet que contienen información organizada y presentada visualmente. Son utilizadas con diversos fines, como brindar información, promocionar productos o servicios, interactuar con los usuarios, ofrecer contenido multimedia, y facilitar la comunicación y colaboración en línea. La creación de una página web puede variar dependiendo de la complejidad de la misma. (GoDaddy, 2023)

Existen opciones sencillas, como los sistemas de gestión de contenidos (CMS), que disponen de plantillas predefinidas y requieren de pocos conocimientos técnicos para ser modificadas. Por otro lado, crear un sitio web desde cero puede ser más complejo, debido a que requiere de técnicas de programación y diseño. Además, es necesario ser consciente de los pasos a seguir para completar una página web; algunas cosas que se debe tomar en cuenta para la

culminación del sitio es que sea totalmente funcional, quiere decir que tenga un buen desarrollo de BackEnd, este bien organizado correctamente el contenido y sea presentado de una forma dinámica y creativa para el cliente, quiere decir que tenga un buen desarrollo de FrontEnd, y por último es que cumpla con los propósitos por lo cual fue diseñada la página, combinando FrontEnd y BackEnd para crear una aplicación sólida y robusta que mejore la calidad de vida del cliente durante su uso. (GoDaddy, 2023)

En la actualidad existen muchos tipos de páginas web que satisfacen necesidades distintas y particulares de diversos grupos de usuarios, entre algunas tenemos: (GoDaddy, 2023)

- Páginas web estáticas.
- Páginas web dinámicas.
- Páginas web de comercio electrónico.
- Páginas web de blogs.
- Páginas web de medios de comunicación.
- Páginas web de redes sociales.
- Páginas web corporativas. (GoDaddy, 2023)

5.7.2. HTML

El Lenguaje de Marcado de Hipertexto (HTML) es la programación basada en código que se utiliza para estructurar los contenidos de una página web como lo son párrafos, imágenes, tablas de datos, hipervínculos, listas con viñetas, etc. En realidad, HTML no es un lenguaje de programación; sino que es un lenguaje de marcado que define una estructura del contenido por medio de etiquetas. Estas etiquetas encierran distintas partes de la información para que sean presentadas de una manera específica. (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

Las etiquetas permiten darle algún propósito en particular al texto que estemos modificando, por ejemplo, modificar el tipo de letras o generar palabras en cursiva, agrandar, crear de una imagen un elemento con hipervínculo, entre otros. (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

El HTML también es conocido como el esqueleto de una página web por la misma razón de organizar la información por medio de las etiquetas, un punto importante de resaltar es que las partes principales de un HTML son: (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

La etiqueta de apertura la cual consiste en el nombre de un elemento encerrado dentro de paréntesis angulares (< >) de apertura y cierre. (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

La etiqueta de cierre es bastante similar a la etiqueta de apertura con la diferencia que incluye una barra inclinada (/) ubicada antes del nombre de la etiqueta que indica el fin de la misma. (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

El contenido que es sencillamente texto que va ubicado dentro de la etiqueta de apertura y cierre. (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

Y por último los elementos, además de la etiqueta de apertura, la de cierre y el contenido se pueden definir ciertos atributos que son ocultos al texto mostrado ayudando a identificar de forma más precisa cada sección de información, también ayudan a que la programación lógica y de estilos sea más cómoda aplicando buenas prácticas al momento de programar. Los atributos deben cumplir con ciertos parámetros como tener un espacio entre este y el nombre del elemento o de otros atributos en caso de poseer alguno anteriormente, nombre del atributo seguido de un signo de igual (=) y por último tener comillas de inicio y fin encerrando el valor del atributo colocado. (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

Existe la posibilidad de colocar etiquetas dentro de etiquetas lo que permite una personalización más específica a ciertas secciones de información dependiendo de los fines del desarrollador. Por otro lado, existen elementos vacíos y se les denomina así ya que no tienen ningún contenido dentro de su etiqueta de apertura y cierre, en este caso tenemos como ejemplo las etiquetas de tipo `` que incluyen internamente una imagen, pero esta es representada como un atributo de la etiqueta. (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

Ciertamente conocer los distintos elementos de HTML es necesario, pero también lo es aprender cómo son utilizados en conjunto ya que individualmente no son muy útiles. ¡Entre las etiquetas principales para el funcionamiento óptimo tenemos `<!DOCTYPE html>` que antiguamente los documentos actuaban vinculados a una serie de reglas que tenían que seguir, lo que significaba la verificación de problemas y otras cosas de utilidad. Actualmente lo consideran como antigüedad, pero debe igualmente ser incluido para el correcto funcionamiento. (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

La etiqueta `<html></html>` considerada como elemento raíz que encierra todo el contenido de la página. (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

La etiqueta `<head></head>` actuando como contenedor de aquello que fuese necesario incluir en la página como contenido no visible, principalmente usado para palabras clave, descripción breve de la página en resultados de búsqueda, llamado de código CSS o JavaScript, entre otros. (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

La etiqueta `<meta charset="utf-8">` es la que establece el código de caracteres que será empleado en el sitio web, entre comillas la codificación implementada que incluye la gran mayoría de caracteres de todos los idiomas humanos. (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

La etiqueta `<title></title>` establece el título de la página y este aparece en la ventana cuando es abierta por el navegador, también sirve para describir la página cuando es agregada a los marcadores. (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

Y por último la etiqueta `<body></body>` es textualmente el cuerpo del sitio web, aquí yace toda la información que el desarrollador desee mostrar a los clientes, como imágenes, juegos, videos, pistas de audio, entre otros. (contributors, Conceptos básicos de HTML - Aprende desarrollo web, 2023)

5.7.3. CSS

Las siglas para describir las hojas de estilo en cascada (Cascading Style Sheets) es un lenguaje que controla la presentación y el diseño de sitios web. CSS funciona en conjunto con HTML y se puede entender que CSS es comparable con los rasgos físicos de una persona. (Santos, 2023)

Se le denomina hojas en cascada debido a que puedes tener varias hojas de configuración de estilos con propiedades heredadas de otras. (Santos, 2023)

La programación de diseño permite crear reglas donde la información introducida por HTML es mostrada según esas reglas. También existe la posibilidad de crear formatos particulares que comunican tus ideas para producir mejoras de experiencias visuales para todos los clientes que visiten la página web. (Santos, 2023)

El funcionamiento es explicado cuando se accede a una web, el buscador debe ubicar la información dentro del HTML y traducirla a un DOM (modelo de objetos del documento). Estos objetos deberán ser solapados con las secciones de código correspondientes en CSS para que los estilos sean aplicados correctamente. (Santos, 2023)

CSS está en constante evolución, debido a que las nuevas tecnologías visuales demandan nuevas técnicas o estrategias para aplicar animaciones o estilos particulares basados en las

necesidades presentadas por los desarrolladores de ciertos navegadores web, o desarrolladores de páginas solicitan características que no han sido incluidas sin modificar lo que ya ha sido creado, ya que sitios web de años pasados deben poder seguir aplicando estilos con la capacidad de programación que había en ese momento, comparado con un sitio web actual con las mismas características. Este lenguaje resulta ser bastante extenso y en progresiva evolución, pero resulta bastante útil aprender el potencial que posee. (contributors, ¿Qué es el CSS? - Aprende desarrollo web | MDN, 2023)

5.7.4. JavaScript

Es un lenguaje de programación que resulta de las secuencias lógicas que podemos realizar a través de comandos los cuales nos permite implementar funciones complejas a sitios web, este lenguaje es lo que da vida a la información en lugar de mostrarla de forma estática, puede mostrar actualizaciones de contenido, mapas interactivos, desplazamiento de reproductores de video, animación de gráficos, entre otros. JavaScript es el tercer lenguaje que compone el tridente de desarrollo web junto con HTML y CSS, se puede representar a este lenguaje como los músculos del cuerpo, debido a que gracias a él podemos darles dinamismo y capacidades a nuestras páginas web por medio del conjunto de acciones que el desarrollador le indique. (contributors, ¿Qué es JavaScript? - Aprende desarrollo web, 2023)

JavaScript es aquello que nos permite como usuarios poder interactuar con las páginas, ingresar formularios, corroborar credenciales y muchas cosas más. Es posible almacenar información importante dentro de variables, operaciones sobre strings (cadenas de texto) y ejecutar código en base a ciertos eventos de los objetos que tengamos incluidos dentro de nuestra página web. (contributors, ¿Qué es JavaScript? - Aprende desarrollo web, 2023)

Cuando un sitio web es cargado ocurre lo mismo que con CSS, primero es leído el código HTML para ubicar el código JavaScript o el enlace que redirecciona al documento que lo

contenga, para posteriormente será ejecutado dentro de la pestaña del navegador. (contributors, ¿Qué es JavaScript? - Aprende desarrollo web, 2023)

Uno de los mayores usos de este lenguaje es la modificación dinámica de HTML y CSS para actualizaciones de interfaz de cliente. El código de los documentos web generalmente se cargan y ejecutan en el orden que aparece el sitio, quiere decir que la programación en JavaScript siempre debe ser lo último que se carga para tener una buena práctica y así evitar errores de asignar eventos a elementos que aún no han sido procesados por el navegador. (contributors, ¿Qué es JavaScript? - Aprende desarrollo web, 2023)

5.8. Redes de computadoras

Una red de computadoras, también conocida como una red de comunicaciones o informática es la interconexión de diversos equipos informáticos a través de un medio físico gracias a dispositivos de telecomunicaciones, ya sean alámbricos o inalámbricos. (Equipo editorial, Etecé, 2023)

El objetivo de crear este tipo de redes es la compartición de la información en paquetes de datos. Estos datos son transmitidos por medio de impulsos eléctricos, ondas electromagnéticas, entre otros, implementando protocolos de comunicación que funcionan como estándar para la codificación de la información para ser transmitida a través de los distintos medios. (Equipo editorial, Etecé, 2023)

Las redes de computadoras son idénticas en su lógica de comunicación ya que poseen un emisor, un receptor, un mensaje, un canal o medio por el cual transmitir la información y los protocolos que garantizan la correcta compartición de los paquetes según las tecnologías que aplican los dispositivos informáticos. Los equipos correspondientes de cada elemento para lo comunicación son: (Equipo editorial, Etecé, 2023)

- El servidor, encargado del procesamiento del flujo de datos en la red atendiendo a los clientes de acuerdo a las peticiones que estos soliciten, esto debido a que no en todos los casos las computadoras no desempeñan la misma jerarquía ni las mismas funciones. (Equipo editorial, Etecé, 2023)
- Clientes o estaciones, estas forman parte de la red y operan bajo la misma necesidad que cubre el servidor. (Equipo editorial, Etecé, 2023)
- Medios de transmisión, estos se refieren a cualquier equipo utilizado para la transmisión de datos en la red, ya sea alámbrico o inalámbrico. (Equipo editorial, Etecé, 2023)
- Elementos de hardware, estos elementos son aquellos que permiten la comunicación física de una red como las tarjetas de red, módems, antenas repetidoras y enrutadores. (Equipo editorial, Etecé, 2023) Elementos de software, de igual manera es necesario tener el software que pueda controlar los equipos físicos y poder comunicarse en el mismo lenguaje entre clientes y servidor. (Equipo editorial, Etecé, 2023)

Al tener equipos interconectados se abre la posibilidad de compartir puntos de acceso a internet o la administración de periféricos como las impresoras, además, lo que destaca es que gracias a este sistema de compartición en red el envío de datos resulta ser rápido sin la necesidad de ocupar medios de almacenamiento externos como memorias flash extraíbles o discos duros externos, que funcionen como intermediarios de las computadoras interconectadas. (Equipo editorial, Etecé, 2023)

Se puede determinar que existe una red de computadores cuando está presente un equipo de comunicación de datos como un módem, conmutador o un equipo terminal de datos, que también es denominado como un nodo, y es necesario que estén presentes 2 o más ordenadores y equipos periféricos, como se muestra en la figura. (AWS Inc. o sus filiales., n.d.)

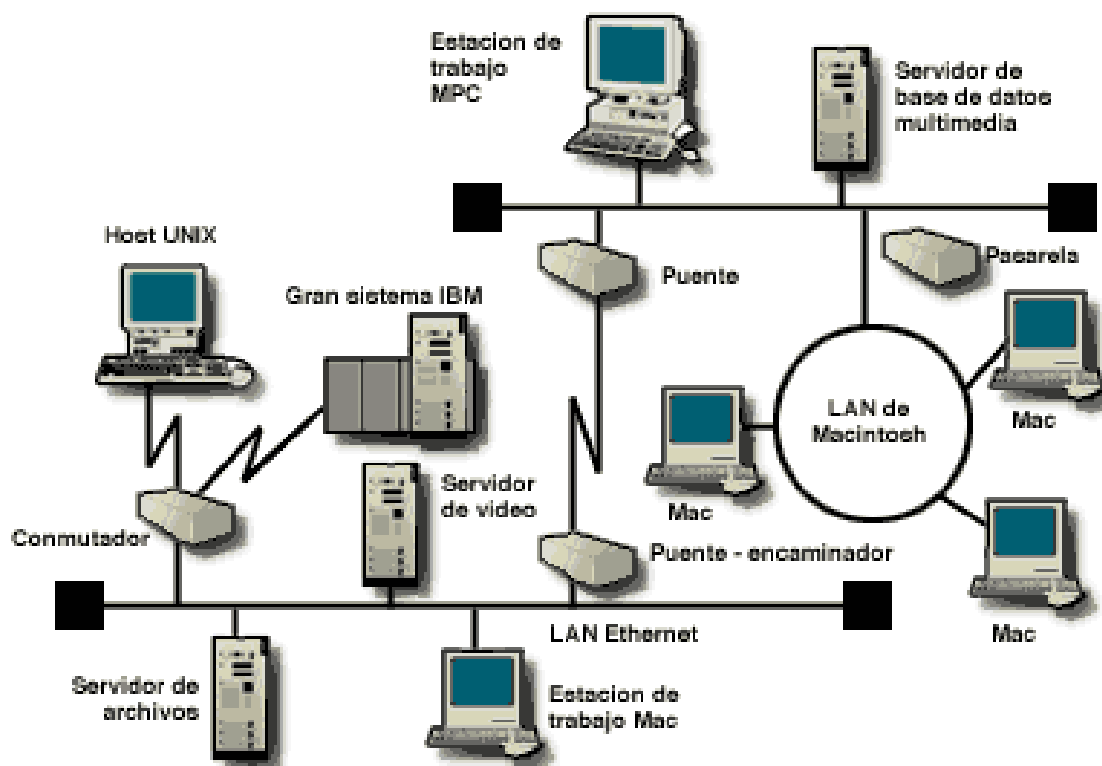


Figura 7. Estructura de una red de computadoras.

Fuente: adaptado de Las conexiones entre las computadoras [diagrama], por Damyy7, 2012, Wikimedia Commons (https://upload.wikimedia.org/wikipedia/commons/3/33/Redes_computacionales.gif). CC BY SA 3.0

En una red los nodos son los que manejan a que estaciones se envía la información. Estos componentes están definidos de forma física y lógica gracias a la arquitectura aplicada, esto proporciona especificaciones para los componentes físicos de red, procedimientos, organización funcional, entre otros. Las redes actuales pueden operar de manera virtual, esto quiere decir que la red física principal se puede dividir de forma lógica para la creación de múltiples subredes. En un esquema de subredes, los nodos están virtualmente conectados y los datos pueden ser enviados entre ellos a través de varias rutas físicas. Otra forma de operar es con la integración a gran escala, esto va relacionado a los servicios de red, permitiendo optimizar las funciones gracias a la supervisión y automatización para generar un alto rendimiento. También operan según las respuestas rápidas a las condiciones de cambio, la mayoría de redes de ordenadores están definidas por software, el tráfico de datos se puede

manejar de forma centralizada por medios digitales. Y, por último, proporcionando seguridad de datos, esto permite tanto un control en la información por medio de un cifrado de datos, como un control de acceso por medio de credenciales, también existen otras maneras de proteger la red como implementar software antimalware o firewalls. (AWS Inc. o sus filiales., n.d.)

Entre los tipos de arquitectura existe la arquitectura cliente servidor donde los nodos funcionan como servidores o clientes. Los que están establecidos como servidor proporcionan recursos como memoria, capacidad de procesamiento o datos a los clientes. Estos también administran el comportamiento de los clientes. Por otro lado, los clientes pueden comunicarse entre sí, pero no comparten recursos. Otra arquitectura que se maneja es la de punto a punto (P2P), esta arquitectura define que todos los dispositivos conectados tienen las mismas capacidades y privilegios, no existe un servidor central, lo que significa que cada equipo conectado puede operar como cliente o como servidor, también pueden compartir recursos como memoria y procesamiento de datos. (AWS Inc. o sus filiales., n.d.)

Algo que también es importante de las redes es su topología, esto significa la disposición de los nodos y enlaces configurándolos de distintas formas según la necesidad. Entre los tipos de topología existen: (AWS Inc. o sus filiales., n.d.)

- Bus, esta comunicación se realiza en una dirección y cada nodo está vinculado a otro. (AWS Inc. o sus filiales., n.d.)
- Anillo, cada nodo está conectado a otros dos nodos y en total todas las conexiones forman un anillo. Los datos se transmiten de forma bidireccional, pero el fallo de un solo equipo supone la caída de la red completa. (AWS Inc. o sus filiales., n.d.)
- Estrella, esta topología define un nodo central el cual se conecta con todo los demás, este es el tipo más implementado en la actualidad ya que existe mayor fiabilidad

debido a que los datos no tienen que pasar por cada estación. (AWS Inc. o sus filiales., n.d.)

- Malla, cada nodo esta interconectado a muchos otros nodos, y en una malla completa cada nodo está conectado a todos los otros equipos de la red. (AWS Inc. o sus filiales., n.d.)

Existen más tipos de topología que funcionan según la necesidad acorde a una distribución lógica como se muestra en la figura.

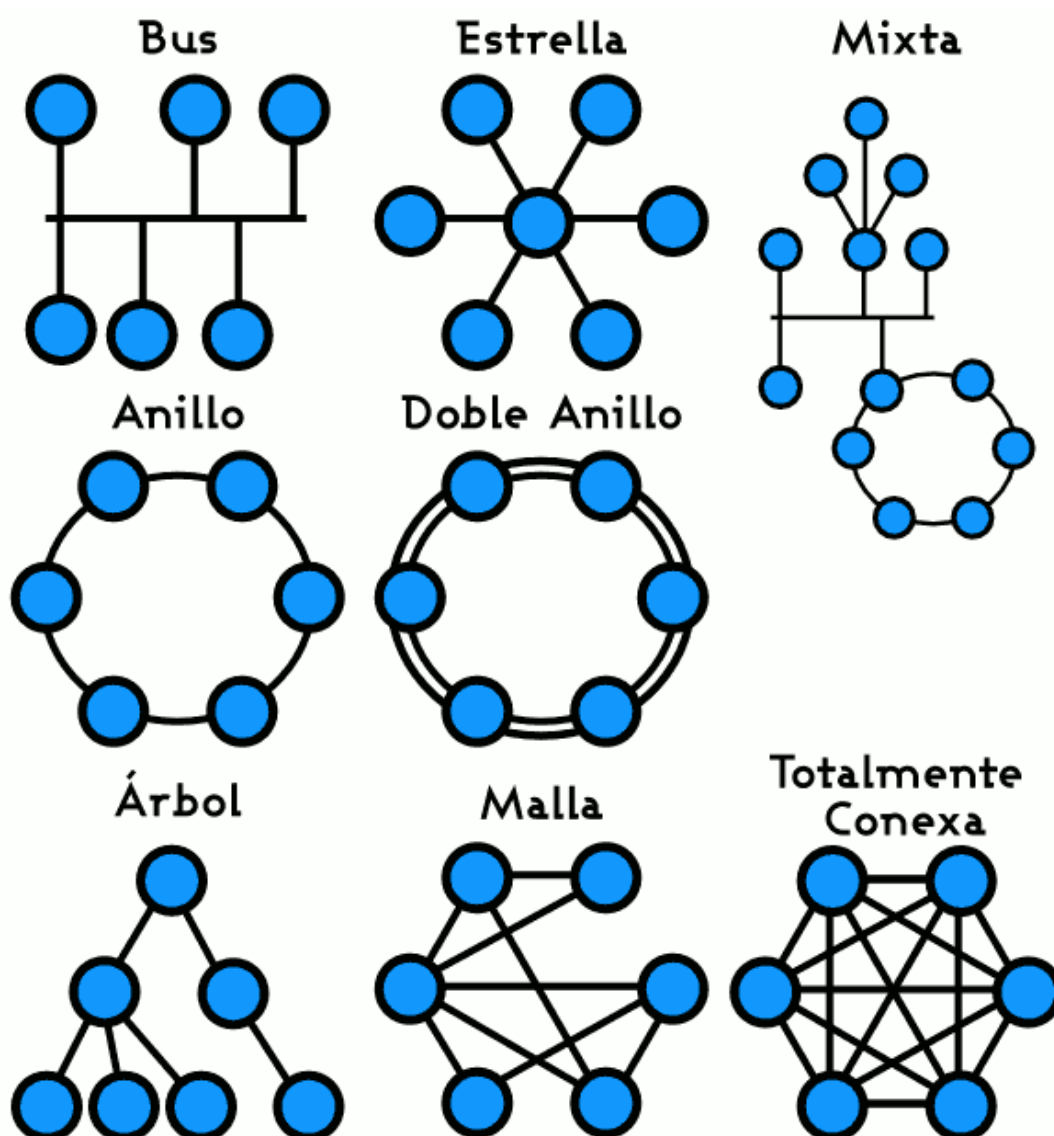


Figura 8. Tipos de topología de red.

Fuente: adaptado de Topología comunes de red [diagrama], por Kordas, 2004, Wikimedia Commons (https://upload.wikimedia.org/wikipedia/commons/4/4a/Topología_de_red.png). CC BY SA 3.0 migrated

Ciertamente las redes fueron la revolución de las comunicaciones, por eso entre las ventajas que existen esta la centralización y compartición de la información entre múltiples equipos, operaciones entre las computadoras rápidamente incluso a largas distancias como interacciones sociales, teleconferencias, videollamadas, compras electrónicas, movimiento de capital, correos electrónicos, compartición de recursos en tiempo real, streaming de contenido audiovisual, vigilancia y exploración satelital. (Equipo editorial, Etecé, 2023)

Lastimosamente también existen vulnerabilidades que generan desventajas ya que los datos son un bien muy preciado para todos en el mundo digital, por lo que muchos quieren controlarlos. Los ataques cibernéticos entran en acción ya que vulneran la confidencialidad de los datos comprometiéndolos en actividades ilícitas, el software malicioso, el ciberterrorismo, violaciones o daños al hardware y software. (Equipo editorial, Etecé, 2023)

5.8.1. Redes LAN

Una LAN o Red de Área Local por sus siglas en inglés (Local Area Network), es un tipo de red contenida en una zona geográfica pequeña, específicamente dentro del mismo espacio físico. Las más comunes que podemos encontrar actualmente son redes Wifi domesticas o de pequeñas empresas, pero también es posible encontrar LAN que sean más grandes, pero suelen igual denominarse de distinta forma. (Cloudflare Inc., n.d.)

El funcionamiento de estas redes es conectándose a Internet por medio de un nodo central, el enrutador o conmutadores de red dependiendo de la magnitud de la LAN. Las LAN utilizan Ethernet que es la conexión física de los equipos por medio de cables los cuales están categorizados por distintas velocidades de transferencia de información, Wifi que es el protocolo para la comunicación por ondas de radio, o ambas para interconectar sus dispositivos. (Cloudflare Inc., n.d.)

Las redes más sencillas de este tipo solo suelen utilizar un enrutador central que sirva para que los demás equipos dentro de la LAN se conecten a él ya sea por el medio que se disponga, ya sea Ethernet o Wifi. Las Redes de Área Local que no estén conectadas a internet necesitan un conmutador intermediario para poder intercambiar los datos. En el caso de LAN que sean más grandes igualmente será útil implementar conmutadores que permitan el paso de la información entre distintos grupos de LAN para agilizar y facilitar la comunicación a los equipos según las peticiones que estos tengan. Lo que requieren los equipos para poder formar parte de una LAN es configurar acorde a los protocolos de Internet incluso sino están conectados a la misma, para poder identificarse entre computadoras, como lo sería configurar la red que utilizara la LAN, las IP de cada estación dentro de la red, la máscara de subred, entre otros. (Cloudflare Inc., n.d.)

Dentro de las LAN también existe el termino de VLAN o LAN virtuales como lo denomina su nombre en inglés (Virtual Local Area Network). Esta práctica es la forma de dividir el tráfico de la red física en varias redes lógicas. Para poder crear este tipo de redes lógicas es necesario tener un enrutador con conexión a internet que disponga de la posibilidad de dividir la red física en redes lógicas. La capacidad de poder realizar esta práctica es la de distribuir los equipos conectados según para aquello que serán destinados, por ejemplo, la información que compete a un departamento se mantendrá exclusiva para ese departamento, y los demás que estén fuera de esa VLAN no podrán acceder a la información ya que es dedicada solo para los computadores que pertenezcan a ese grupo. Visto desde las instalaciones físicas es ver distintas redes que no se conocen entre sí, pero por medio de un enrutador compartir información acorde a su función con un servidor. Esta funcionalidad permite una mejor gestión de LAN que sean más grandes que el promedio sin necesidad de adquirir más equipos físicos para la separación de los dispositivos. (Cloudflare Inc., n.d.)

5.8.2. Redes WAN

La Red de Área Amplia (Wide Area Network) permite conectar redes más pequeñas en un área geográfica mayor, siendo este tipo de redes las que conectan a todo el mundo. En la actualidad, las organizaciones usan enlaces WAN privados para conectar distintas sucursales a su centro corporativo donde se manejan los datos. Las empresas lo único que alquilan son las líneas a los proveedores de servicios de Internet. Donde implementan tecnologías como la SD-WAN y la conmutación de etiquetas multiprotocolo. (Aruba Networks, s.f.)

Para este tipo de conexiones es necesario la presencia de un enrutador en cada LAN e incluso enrutadores intermediarios que permitan distribuir la información de forma eficiente. Existen las WAN públicas y privadas, en el caso de las privadas quiere decir que solo una organización opera sobre esa red, visto en la realidad es la forma en como una empresa se interconecta entre distintas sucursales de distintos países para que todos manejen la misma información acorde a lo que dicta la matriz de la organización, además en las WAN privadas existe mayor seguridad y fiabilidad de los datos. Por otro lado, una WAN pública es la red utilizada para interconectar cualquier dispositivo que tenga la capacidad de poder entrar en esas redes, el ejemplo más grande de una WAN pública es el Internet. (Aruba Networks, s.f.)

A lo largo de los años se han utilizado diversas tecnologías para la comunicación de este tipo de redes, como los satélites, los enlaces de microondas, líneas telefónicas, fibra óptica, entre otros. Los datos son transmitidos igualmente en paquetes que son divididos en distintos tamaños para evitar problemas de red como la pérdida de la información, la latencia, entre otros. Los paquetes siguen el método de empaquetado del stack de protocolos TCP/IP que define el modelo OSI de las capas que tienen que tener los paquetes para poder ser enviados al destino correcto, este mismo modelo se aplica a las redes LAN. (Aruba Networks, s.f.)

Un problema que pueden presentar las redes WAN es la distancia entre los dispositivos, por ello es propensa a sufrir de una latencia más elevada que las redes LAN afectando el

rendimiento de la red. Para poder solucionar estos efectos se puede mejorar la transmisión con técnicas de optimización WAN, que incluye la aceleración del protocolo TCP, la deduplicación o la compresión de los datos. (Aruba Networks, s.f.)

El objetivo de una conexión WAN es poder compartir recursos entre empleados y clientes, comunicarse mediante audio y voz, acceder al medio de almacenamiento de datos y crear copias de seguridad, conectar con aplicaciones, alojar aplicaciones internas. Para una mayor profundidad en la arquitectura del modelo de Interconexión de Sistemas Abiertos (OSI) el cual estandariza las telecomunicaciones en 7 capas de funcionamiento para realizar la comunicación como se muestra en la figura. (Amazon Web Services, n.d.)

LA PILA OSI



Figura 9. Capas del modelo OSI.

Fuente: adaptado de Pila OSI de ISO[dibujo], LordT, 2007, Wikimedia

Commons(<https://upload.wikimedia.org/wikipedia/commons/thumb/7/7d/Pila-osi-es.svg/636px-Pila-osi-es.svg.png?20080813021144>). CC BY SA 3.0 migrated

Entre las capas están:

- Capa 7, conocida como la capa de la aplicación, esta es la última capa de la jerarquía y por ende es la capa más cercana al usuario final. Contiene aquello referente a la lógica de la aplicación desconociendo la implementación de la red, por ejemplo, lo

que almacena esta capa son procesos como el envío de invitaciones a eventos, reservas, zonas horarias, entre otros. (Amazon Web Services, n.d.)

- Capa 6, la capa de la presentación, esta capa prepara los datos para su transmisión, por ejemplo, la incorporación de cifrado de datos para evitar el robo de información confidencial por parte de los delincuentes cibernéticos a las empresas en la WAN. (Amazon Web Services, n.d.)
- Capa 5, conocida como la capa de la sesión, esta capa administra los inicios de sesión en aplicaciones locales y remotas, permitiendo abrir, cerrar, o terminar la conexión entre dos equipos. (Amazon Web Services, n.d.)
- Capa 4, la capa de transporte define los procedimientos para la transmisión de los datos. Clasifica y envía la información, también tiene la capacidad de empaquetar y seccionar la información en distintos paquetes para su transmisión. Por ejemplo, un sitio de reservas donde el protocolo de transmisión TCP administra la comunicación clasificando los paquetes según la solicitud y la respuesta. (Amazon Web Services, n.d.)
- Capa 3, capa de red, esta capa administra el modo en que los paquetes viajan por la red, definiendo las reglas de enrutamiento, equilibrio de cargas y la pérdida de paquetes. (Amazon Web Services, n.d.)
- Capa 2, la capa de enlace de datos, esta capa se ocupa de las reglas de comunicación sobre las operaciones de capa física, decidiendo el inicio o el fin de una conexión. En este nivel del modelo la capa reenvía paquetes de un dispositivo a otro hasta llegar a destino. (Amazon Web Services, n.d.)
- Capa 1, y por último la capa física, en esta capa se administra la transferencia de información en forma de bits, señales ópticas u ondas de radio a través de los enrutadores conectados a la red. (Amazon Web Services, n.d.)

Aunque en la actualidad el protocolo más usado es el TCP/IP, no es el único protocolo implementado en WAN. El protocolo TCP/IP define la comunicación de extremo a extremo especificando de qué manera los datos deben ser empaquetados, direccionados, transmitidos, enrutados y recibidos. Actualmente la versión utilizada es el IPv6. (Amazon Web Services, n.d.)

Otros protocolos conocidos como la retransmisión de tramas, esta es una tecnología que empaqueta la información en tramas transmitiéndola en una línea privada a un nodo de retransmisión. Este protocolo agiliza la transferencia en las capas 1 y 2 del modelo OSI de una LAN a otra por medio de varios enrutadores. El modo de transferencia asíncrono (ATM) es una de las primeras tecnologías WAN la cual formatea los datos en celdas de 53 bytes. Los dispositivos en modo ATM aplican multiplexación por división de tiempo, lo que hace esto es convertir las señales digitales en celdas de tamaño fijo, para ser transmitidas y luego ser nuevamente ensambladas en el destino. El paquete sobre SONET/SDH se trata de un protocolo que expone la forma de comunicar los enlaces punto a punto implementando fibra óptica. (Amazon Web Services, n.d.)

5.8.3. Redes MAN

Una Red de Área Metropolitana (MAN) es un tipo de red que conecta los equipos que pueden ser de una gran ciudad, varias locaciones o pueblos. Este tipo de redes es el intermedio entre las redes LAN ya que son más grandes que estas, pero más pequeñas que una red WAN. El termino metropolitano es adecuado a la extensión de terreno que puede cubrir, no exactamente la zona a la que sirve. (Cloudflare, s.f.)

Una red MAN al igual que una WAN está formada por múltiples redes LAN interconectadas, la diferencia es que al ser más pequeñas que las WAN suelen ser más eficientes ya que la latencia se ve reducida por las distancias recorridas. La mayoría de estas redes utilizan

cables de fibra óptica entre LAN, específicamente con fibra oscura que son cables que no han sido utilizados anteriormente capaces de cargar con tráfico. (Cloudflare, s.f.)

Las redes MAN tienen muchas ventajas para poder ser implementadas en grandes organizaciones, permitiéndoles mejorar distintos aspectos de red como costos, problemas de seguridad, dependencia de proveedores o incluso dificultades en instalación. (Cloudflare, s.f.)

5.8.4. La nube

La nube es un término nuevo muy utilizado en espacios digitales. La nube es una red de servidores distribuidos por todo el mundo que están activos todo el día durante todos los días que operan como una única red donde se puede almacenar cualquier tipo de información, programas y cualquier otra cosa relacionada a lo digital, dando posibilidad de acceder al contenido desde cualquier parte del mundo sin ocupar espacio en tus dispositivos de almacenamiento local. Cuando se trabaja en la nube se dice que los procesos informáticos tienen lugar en el servidor remoto que ofrece el servicio y no en la computadora local. Por esto la nube es un término informático tan utilizado y tan importante para grandes empresas y el mundo en general, por presentar mejoras laborales al momento de productividad y eficacia. (Alonso, 2024)

La gran mayoría de aplicaciones ya se ejecutan en la nube, librando al usuario final de cuestiones de mantenimiento, almacenamiento, riesgo de pérdida de la información, entre otros. Incluso de las empresas de tener una infraestructura de servidores de alta capacidad dedicados a la misma. El almacenamiento en la nube es de los servicios más implementados y que ciertas empresas ofrecen como servicio a otras empresas ya que esto les permite guardar grandes volúmenes de datos e información a coste inferior, ofrece mantener tus datos mejor asegurados ya que suelen tener lo más avanzado en tecnología de ciberseguridad para evitar ser hackeados por los distintos medios que existen, también la capacidad de compartir la información

almacenada con otros usuarios además del trabajo en simultaneo con sincronización automática, junto con un acceso fácil e intuitivo por la web o aplicaciones para móviles o pc como se muestra en la próxima figura. (Alonso, 2024)



Figura 10. Imagen del funcionamiento de la nube.

Fuente: adaptado de Stock Computadoras 3D [dibujo], por Vectorportal.com, 2019, VectorPortal (<https://vectorportal.com/storage/computers-3d-vectorportal.jpg>). CC BY 4.0

Como los temas anteriormente mencionados, la nube también tiene una categorización:

- Nube privada, esto es una serie de servidores conectados entre sí, pero específicamente dedicados a una empresa. Lo que indica que ofrece servicios únicamente a los equipos conectados a esta red corporativa, visto en medios digitales de educación remota donde la plataforma funciona también como portafolio virtual para almacenar datos referentes de estudio. (Alonso, 2024)
- Nube pública, este tipo de nube es principalmente el que ofrecen servicios de almacenamiento como Google Cloud, Amazon Web Services o Microsoft Azure dando soporte a recursos virtuales para otras compañías o usuarios individuales.

Cada usuario paga por los servicios que ocupa y no existe un control de terceros con los que se comparten los servidores. (Alonso, 2024)

- Nube híbrida, este tipo de nube opera con las dos anteriores, pero con la diferencia de que puede ofrecer servicios públicos y privados, mayormente esta nube es operada en privado, pero utiliza la nube pública para la creación de copias de seguridad. (Alonso, 2024)
- Multi nube, en este caso la empresa cliente ocupa distintos servicios de diferentes proveedores de nubes públicas por lo que posee recursos virtuales de varias nubes públicas. (Alonso, 2024)

Los servicios de nube específicamente que pueden ser ofrecidos está el Software como Servicio (SaaS) que es el servicio de aplicaciones web o móviles con acceso a ellas por medio de Internet. También el servicio de Plataforma como Servicio (PaaS), este servicio consiste en una plataforma en la nube que permite a los clientes la creación de aplicaciones o entornos de forma rápida y ágil. Otro servicio de nube es el de Infraestructura como Servicio (IaaS), son de virtualización en la nube, quiere decir que quienes contratan este servicio son responsables de actualizaciones de software que gestionen por medio de la infraestructura de la nube. (Alonso, 2024)

El siguiente paso de la nube es el Cloud Computing que es el grupo de recursos que pueden ser ejecutados. Esta función de la segunda de mayor relevancia para corporaciones gracias poder almacenar y procesar servicios de software o procesamiento de datos que estén en la nube. De lo más utilizado hoy en día con la computación en la nube está el streaming de video, redes sociales, Google Drive, Netflix, Instagram o cualquier software adecuado a lo mencionado. Esto ha sido posible gracias a la mejora de la conectividad, la velocidad en la que la información viaja por Internet y la virtualización que es emular un entorno virtual como es

el funcionamiento de un ordenador físico. Estas máquinas virtuales permiten la instalación de sistemas operativos y acciones como se haría en un computador normalmente. (Alonso, 2024)

5.8.5. NGROK

Esta aplicación es uno de múltiples servicios que permite al usuario que lo maneja, poder crear un subdominio en un dispositivo local para poder ser visualizado fuera de la LAN por medio del Internet a través de un túnel donde se realizan las conexiones del local al servidor de la compañía, permitiendo la subida de aplicaciones o plataformas de forma online para cualquier tipo de revisiones de seguridad como intrusiones. (ACV, 2022)

La ventaja de utilizar este software multi plataforma es que es muy utilizado para el hacking debido a que se puede crear un subdominio para crear páginas phishing. Esto es gracias a que Ngrok no aplica restricciones por el uso de sus servidores. (ACV, 2022)

Para utilizar esta herramienta es necesario visitar su página web oficial, que se detalla en el anexo 9. Desde el portal web, existen dos formas de utilizar el software. El primero es descargar una versión de su ejecutable acorde al sistema operativo, y el segundo es aplicar una serie de comandos a través de la consola que realiza una búsqueda de librerías según las especificaciones del fabricante.

La diferencia entre ambas opciones radica en que el ejecutable debe estar presente en todo momento en el ordenador conectado a la misma red que el dominio local, mientras que el segundo método requiere la ejecución de los comandos una sola vez, permitiendo el acceso al programa en cualquier momento.

El funcionamiento de esta herramienta se realiza a través de la consola de comandos. El ejecutable descargado abre directamente la terminal con una serie de comandos que facilitan el manejo del software. Para configurarlo correctamente, es necesario ingresar un token en la terminal, el cual se obtiene en la página oficial, como se muestra en la figura.

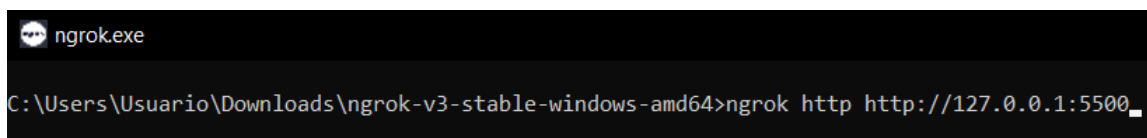


```
ngrok config add-authtoken 2iWf59ghBdjRLcU81y4Yi20p40F_48rgK2TSwnkxo9v5H8b4
```

Figura 11. Token de configuración para la herramienta de Ngrok.
Fuente: El Autor.

Este token es proporcionado por el desarrollador del software en su página oficial y debe ser ingresado desde la misma terminal donde se ejecuta el programa.

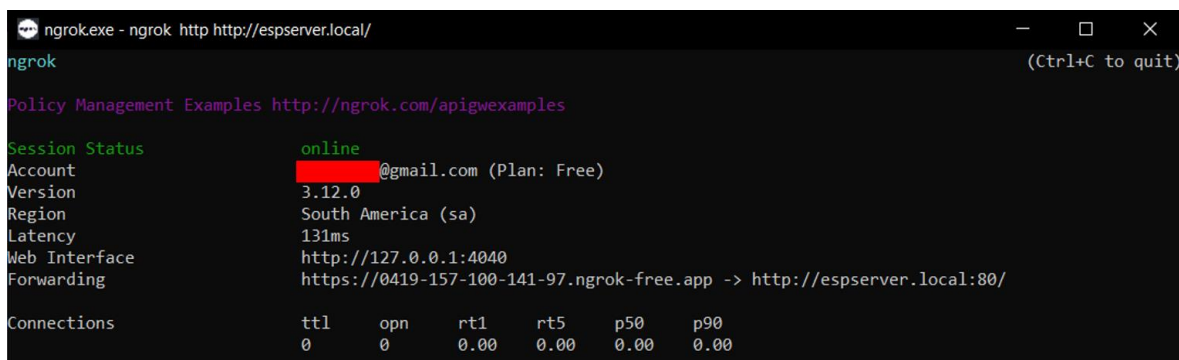
Una vez ingresado el token, la herramienta queda libre para asignar un servicio. Para configurar un servicio, es necesario que este esté activo de forma local, que se pueda acceder a ese servicio sin problemas y que exista un puerto de comunicación en uso para el servicio. Este puerto debe ser especificado en un orden dentro de la consola, como se muestra en la figura.



```
ngrok.exe
C:\Users\Usuario\Downloads\ngrok-v3-stable-windows-amd64>ngrok http http://127.0.0.1:5500_
```

Figura 12. Comando para realizar tunel de servicio local con el servidor de Ngrok.
Fuente: El Autor.

Si todo es correctamente aplicado el servicio de acceso desde Internet ya estará activo como se muestra en la figura.



```
ngrok.exe - ngrok http http://espserver.local/
ngrok (Ctrl+C to quit)
Policy Management Examples http://ngrok.com/apigwexamples
Session Status online
Account [redacted]@gmail.com (Plan: Free)
Version 3.12.0
Region South America (sa)
Latency 131ms
Web Interface http://127.0.0.1:4040
Forwarding https://0419-157-100-141-97.ngrok-free.app -> http://espserver.local:80/
Connections
  ttl   opn   rt1   rt5   p50   p90
  0     0     0.00 0.00 0.00 0.00
```

Figura 13. Conexión establecida con el servicio local.
Fuente: El Autor.

Como se puede observar en la descripción anterior, se presenta un enlace HTTPS que establece un túnel hacia el servicio HTTP de una página web alojada en el servidor local, utilizando el puerto 80 configurado para este propósito.

Existen dos tipos de cuentas disponibles: gratuita y de pago. La cuenta gratuita proporciona un dominio que cambia cada vez que se activa el servicio, limitado a un solo servicio activo a la vez. Por otro lado, la cuenta de pago permite gestionar varios dominios y personalizar los dominios según las necesidades del usuario.

5.9. Simbología electrónica

La simbología electrónica es constantemente implementada para identificar cada componente eléctrico de un circuito. Se identifican según sus características y utilidad para clasificar los componentes. (Anda, Simbología Electrónica, 2018)

Entre algunos tipos existen los cables conductores eléctricos los cuales poseen características de poca resistencia eléctrica, lo que los hace ideales para transmitir la corriente siendo los principales actuadores dentro de un circuito. (Anda, Simbología Electrónica, 2018)

Los interruptores, este grupo de componentes permiten la interrupción de la corriente, la idea con estos elementos es evitar el flujo de electrones debido a la diferencia de cargas, generando una detención del movimiento de la electricidad. (Anda, Simbología Electrónica, 2018)

Las resistencias, este grupo de componentes son utilizados para generar resistencia al paso de la corriente en un circuito, su nomenclatura de medición son los ohmios y es representado con la letra griega. Iluminación, estos elementos aprovechan la corriente eléctrica para generar energía lumínica. (Anda, Simbología Electrónica, 2018)

Transformadores, son componentes electrónicos que generan una disminución de la corriente por medio de una inducción magnética a través de dos o más bobinas. Motores, los

motores son el resultado de interacciones entre bobinas o imanes que intercambian energía eléctrica en energía mecánica y viceversa. (Anda, Simbología Electrónica, 2018)

Condensadores, también conocidos como capacitores, estos equipos tienen la característica de almacenar energía que es posteriormente liberada, son diseñados con dos o más superficies conductoras separadas por algún material de baja conductividad. (Anda, Simbología Electrónica, 2018)

Y, por último, las fuentes de energía, que brindan corriente eléctrica en un circuito. (Anda, Simbología Electrónica, 2018)

Cada tipo de estos componentes electrónicos es importante debido a su uso, así mismo tienen una simbología correspondiente, para más información revisar el anexo 10.

5.9.1. Cables Jumpers

Un jumper o puente eléctrico es un componente que permite cerrar el circuito eléctrico al conectar dos puntos específicos dentro del mismo. (Anda, Cables jumper macho-macho, 2018)

El cable con conectores macho-macho se utiliza frecuentemente en tableros protoboard, facilitando la interconexión de dos componentes insertados en el tablero. (Anda, Cables jumper macho-macho, 2018)

La denominación "macho-macho" proviene del hecho de que ambos extremos del cable poseen conectores sobresalientes, como se muestra en la figura. (Anda, Cables jumper macho-macho, 2018)

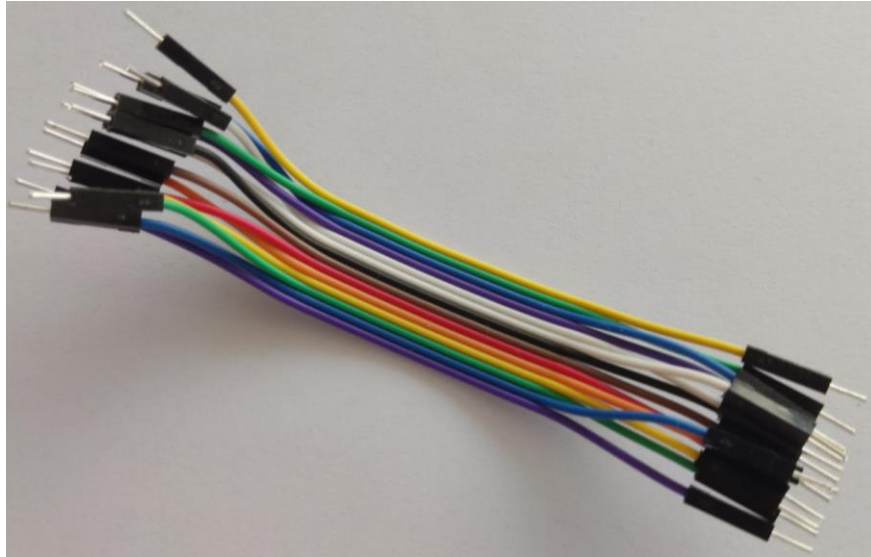


Figura 14. Jumpers macho-macho.
Fuente: El Autor.

Mientras que, la denominación "macho-hembra" proviene del hecho de que uno de los extremos del cable posee un conector sobresaliente, mientras que el otro extremo está diseñado para recibir un pin. (Anda, Cables jumper macho-hembra, 2018)

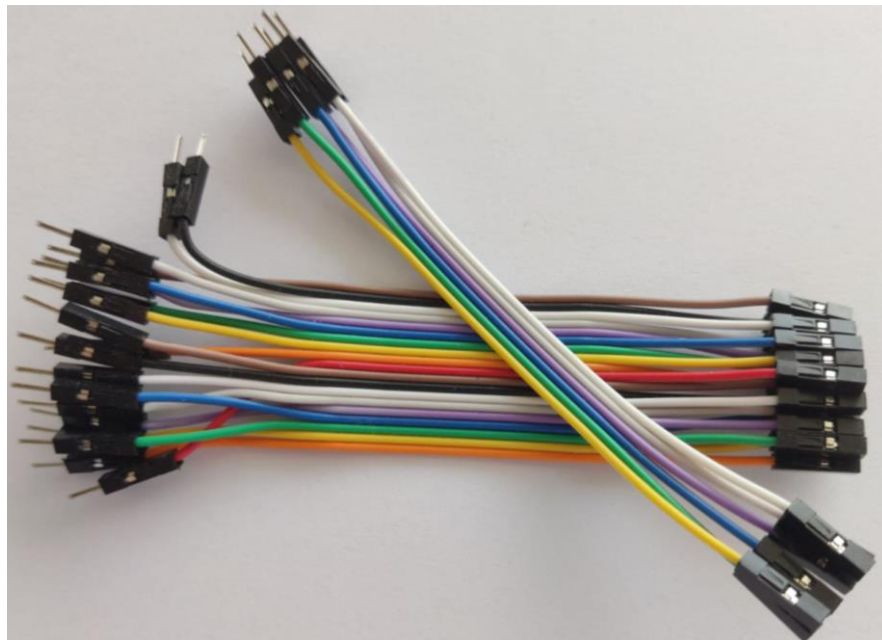


Figura 15. Jumpers macho-hembra.
Fuente: El Autor.

6. Desarrollo del Proyecto de Titulación

6.1. Elección de las tecnologías

En la etapa inicial del proyecto, se planteó la idea de utilizar Arduino como microcontrolador central junto con un dispositivo Bluetooth para la comunicación. Arduino se consideró por su capacidad para gestionar la información y compartirla entre dispositivos. Sin embargo, durante el análisis surgieron algunas limitaciones significativas, como la dificultad para conectar múltiples dispositivos Bluetooth de manera simultánea y el tamaño considerable del Arduino, que aumentaba aún más al incorporar cualquier shield adicional. Estas desventajas llevaron a redirigir el proyecto hacia la implementación de tecnologías de Internet de las Cosas (IoT).

La elección final recayó en el microcontrolador ESP8266, que integra un módulo Wi-Fi. Esta tecnología ofreció ventajas importantes, como una mayor capacidad de memoria y procesamiento, además de un diseño más compacto, lo que lo convirtió en una opción ideal para este proyecto.

Posteriormente, se investigaron las tecnologías disponibles para los sensores, ya que actualmente existen múltiples opciones diseñadas para adaptarse a diversas necesidades. El marco teórico permitió identificar los diferentes tipos de sensores y justificó la elección del sensor ultrasónico como el más adecuado para cumplir con los requerimientos del sistema, debido a su precisión y funcionalidad en este tipo de aplicaciones.

Además, fue fundamental distinguir entre la programación orientada al usuario final y la programación de los procesos subyacentes del sistema. Gracias al desarrollo teórico, se definieron claramente los aspectos que debían programarse en cada ámbito para garantizar una experiencia de usuario óptima y una operación eficiente del sistema.

Por último, se detallaron los lenguajes de programación y los entornos utilizados, con el propósito de proporcionar al lector una comprensión sólida de las bases técnicas sobre las cuales se desarrolló el proyecto, destacando cómo estas tecnologías contribuyeron al cumplimiento de los objetivos planteados.

6.2. Diseño del sistema prototipo y montaje de los módulos electrónicos

Para proceder con el ensamblaje físico, es necesario primero comprender el funcionamiento completo del sistema. Por ello, se diseñaron varios diagramas que ilustran todos los procesos necesarios para completar la serie de tareas, desde la detección de la presencia de un automóvil hasta la presentación de la información en la página web. Con el siguiente diagrama, se podrá visualizar el resultado que se busca obtener.

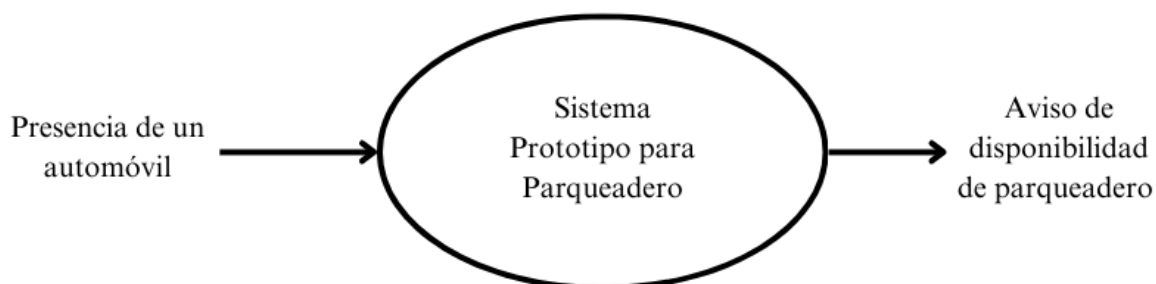


Figura 16. Diagrama general de comportamiento del sistema prototipo.
Fuente: El Autor.

Como se observa en la figura anterior, la entrada del sistema es la detección de la presencia de un automóvil, lo que permite determinar la disponibilidad del espacio a través de una notificación.

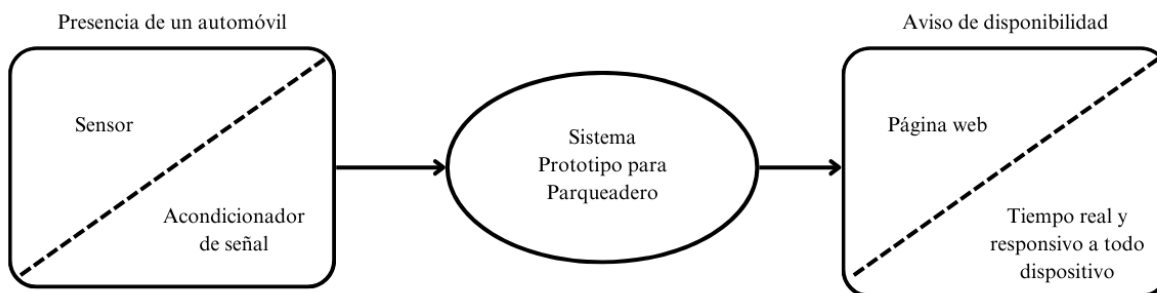


Figura 17. Diagrama de comportamiento del sistema prototipo con detalle de entradas y salidas.
Fuente: El Autor.

En la figura anterior se detalla el proceso a seguir. Para lograr el objetivo de la entrada, se implementa un sensor ultrasónico para medir la distancia de un objeto. Este sensor, junto con un acondicionador de señal, convierte la medida obtenida a centímetros. La información es posteriormente procesada por el sistema y finalmente mostrada en una página web en tiempo real, accesible desde cualquier dispositivo con acceso a Internet.

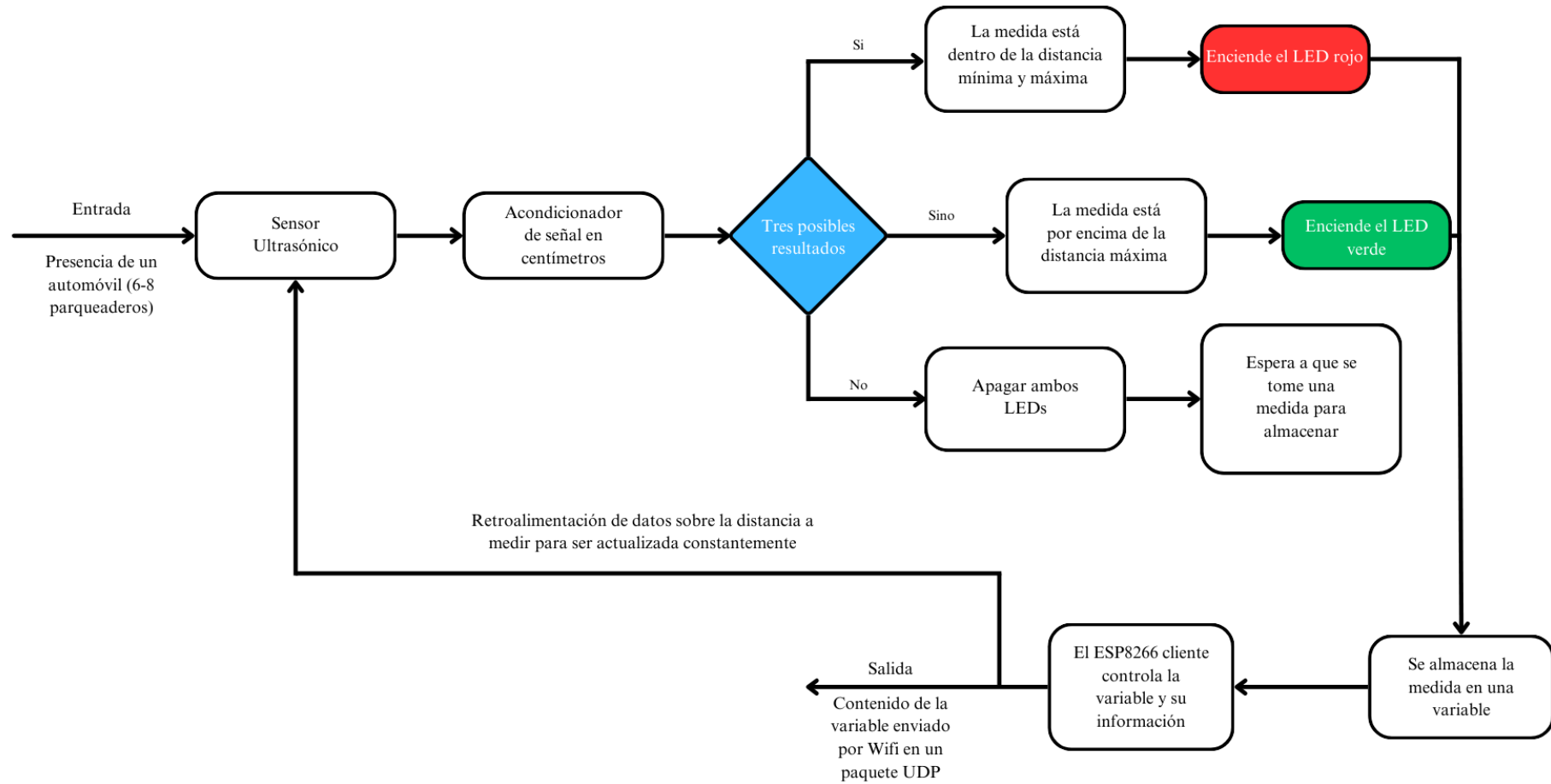


Figura 18. Diagrama de comportamiento de la entrada del sistema prototipo.

Fuente: El Autor.

Gracias al diagrama, se pueden identificar todas las acciones y decisiones que tienen lugar dentro del módulo, así como el resultado final que inicia otro conjunto de procesos.

En la figura anterior se muestra el proceso inicial que realiza el sistema, explicando el procedimiento lógico aplicado para determinar la presencia o ausencia de un vehículo. El primer paso es la detección realizada por el sensor ultrasónico. La información capturada pasa por el acondicionador de señal, que convierte el valor obtenido a la unidad de medida deseada. Posteriormente, dentro del cliente, se efectúa una operación de verificación basada en la distancia medida. Esta verificación se indica mediante LEDs, que señalan la presencia de un objeto. El resultado de esta operación se almacena en una variable para ser enviado al servidor.

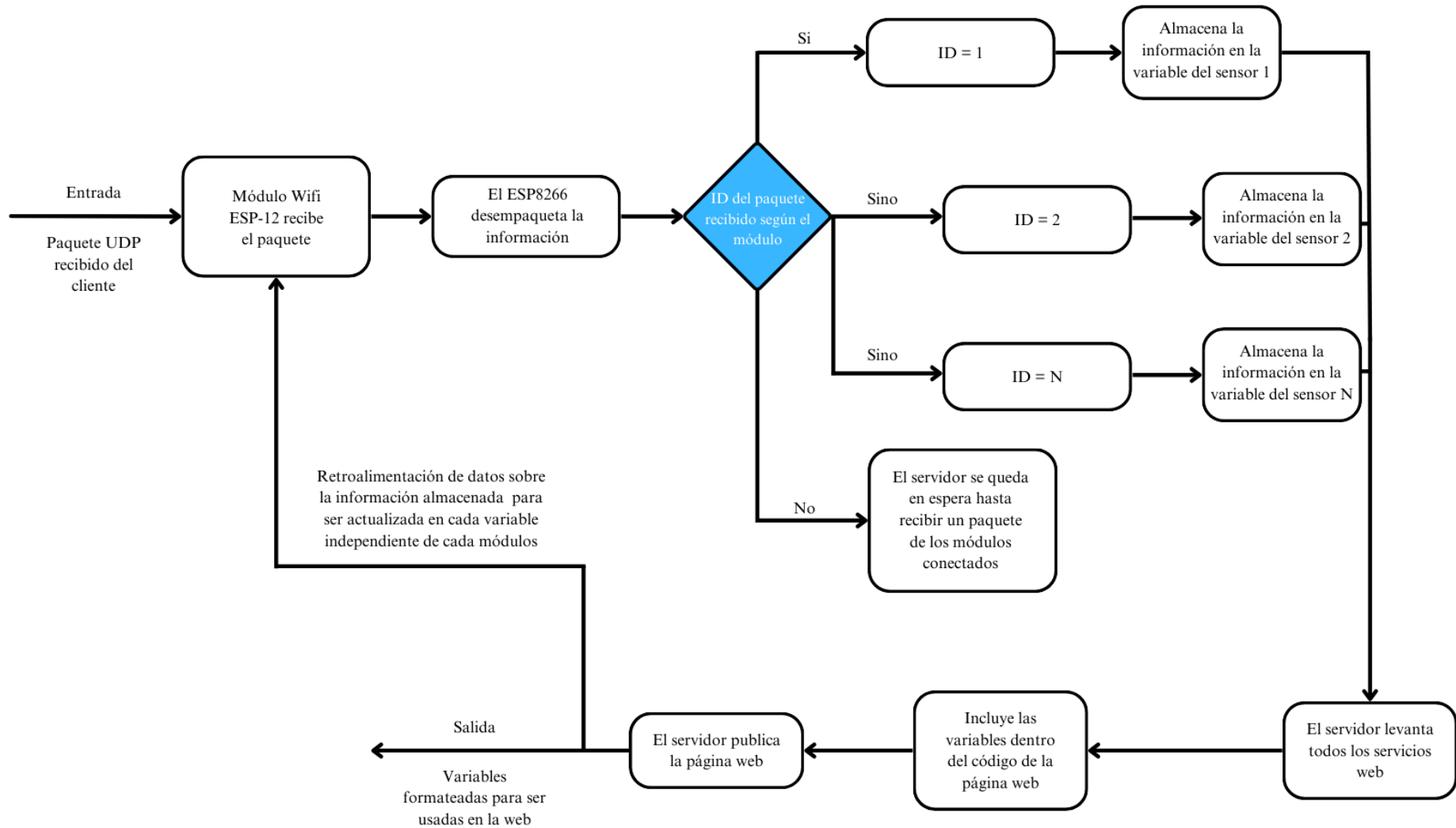


Figura 19. Diagrama de comportamiento del sistema prototipo modelo cliente-servidor.
Fuente: El Autor.

Como se indica en la figura anterior, estos son los procesos que realiza el servidor al recibir los paquetes de cada módulo cliente. El servidor analiza la información y la somete a un proceso sencillo para identificar de cuál cliente se ha recibido el paquete. Posteriormente, el servidor ejecuta todas las acciones necesarias para proporcionar los servicios web donde la información pueda ser mostrada. Además, incluye un túnel por el cual la información recibida de los clientes se presenta en el proceso final.

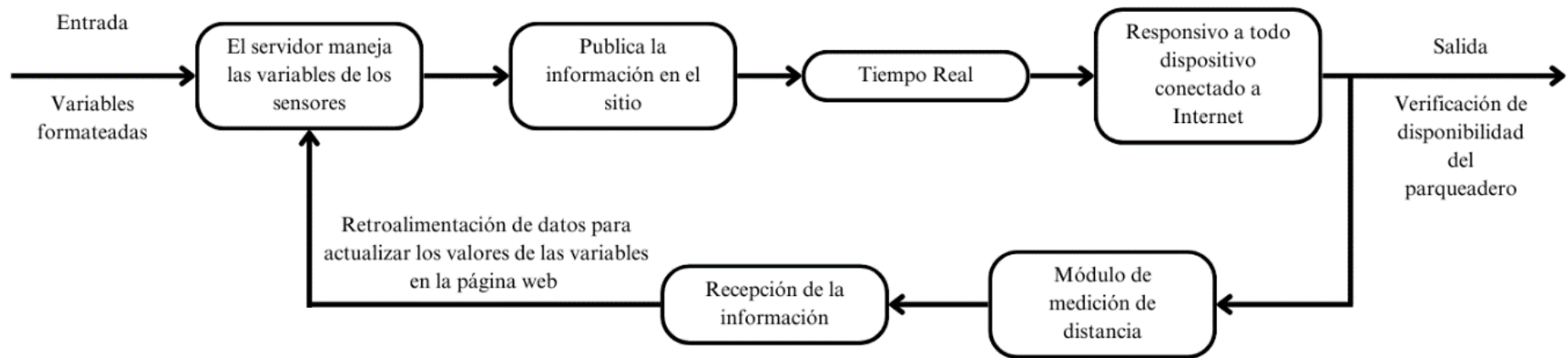


Figura 20. Diagrama de comportamiento con la capacidad de respuesta en tiempo real de la página web con el usuario final.
Fuente: El Autor.

Finalmente, como se muestra en la figura anterior, cuando el servidor identifica la información recibida de los clientes, este puede actualizar, mediante solicitudes constantes de datos a través del portal web, los campos correspondientes a las distancias de cada estacionamiento. Esto permite que el usuario final pueda verificar la disponibilidad en tiempo real.

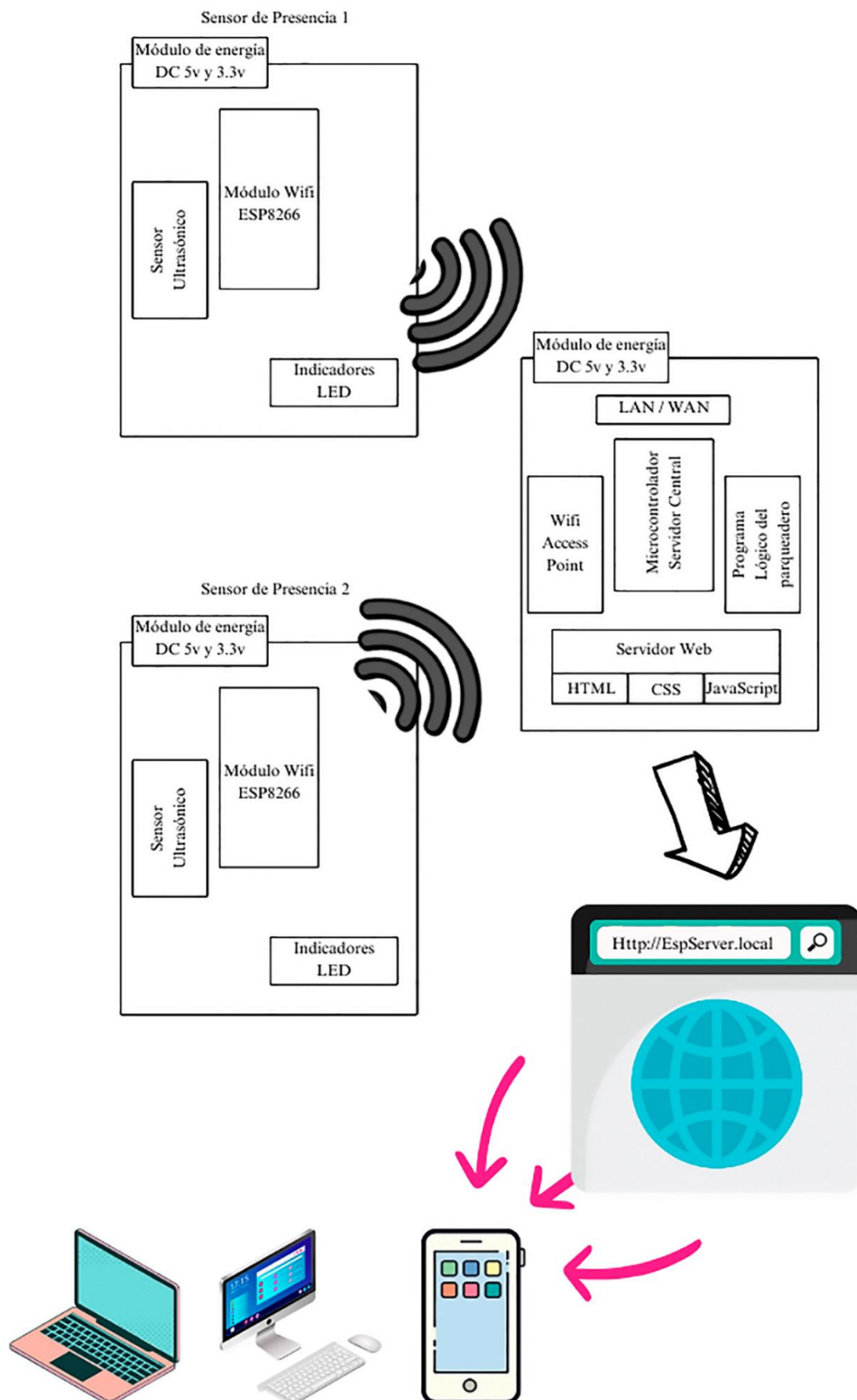


Figura 21. Diagrama estructural del sistema prototipo.
Fuente: El Autor.

Gracias a la determinación de los diagramas de comportamiento, se pudo definir un modelo estructural, expuesto en la figura anterior, que permite identificar las partes esenciales del sistema prototipo, según lo que se utilizará durante su operación.

Para finalizar el apartado del diseño fue necesario demostrar junto con los planos correspondientes como es el espacio el cual será montado el proyecto. Lo primero es reconocer y dimensionar el espacio como se muestra en la figura.

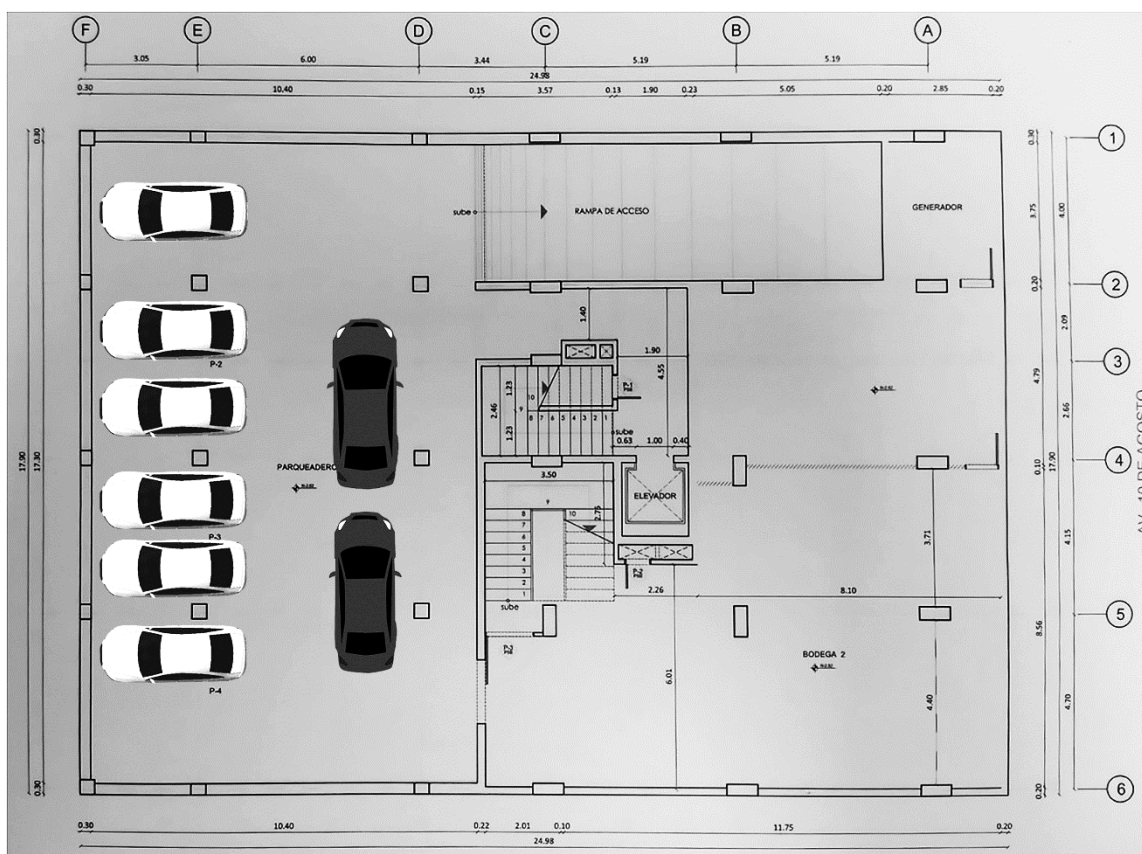


Figura 22. Plano de planta junto con la distribución de vehículos dentro del parqueadero.
Fuente: El Autor.

La distribución fue propuesta y adecuada por el Instituto para el presente proyecto el cual solo cubrirá los vehículos ubicados dentro de los espacios mostrados junto con sus medidas las cuales están representadas en metros. Como se observa en la figura, la disponibilidad máxima esta adecuada para 8 puestos que están identificados en sentido antihorario del 1 al 8 y específicamente para automóviles livianos, por lo que camionetas muy grandes, camiones

pequeños entre otros vehículos, no pueden acceder debido al espacio ya que puede entorpecer el tránsito dentro del estacionamiento. Para una mejor comprensión y visualización se mostrará como es el espacio visto frontalmente de cada área visualizadas en las siguientes figuras.



**Figura 23. Distribución de los puestos dentro del parqueadero del 1 al 3 en sentido antihorario.
Fuente: El Autor**

Los puestos mostrados en la figura anterior representan los que están ubicados justo en frente de la salida pegados a la pared.



Figura 24. Distribución de los puestos dentro del parqueadero del 2 al 4 en sentido antihorario.
Fuente: El Autor

La imagen anterior muestra los siguientes parqueaderos que están de derecha a izquierda de la entrada.



Figura 25. Distribución de los puestos 5 y 6 dentro del parqueadero en sentido antihorario.
Fuente: El Autor.

Los dos últimos puestos ilustrados en la figura previa corresponden a la ubicación de los vehículos que se encuentran más hacia el interior del parqueadero y están situados contra la pared.



**Figura 26. Distribución de los puestos 7y 8 dentro del parqueadero en sentido antihorario.
Fuente: El Autor.**

En última instancia, los puestos que se encuentran en la sección intermedia, tal como se ilustra en el plano del parqueadero, serán aquellos que se ubiquen en la parte media del mismo.

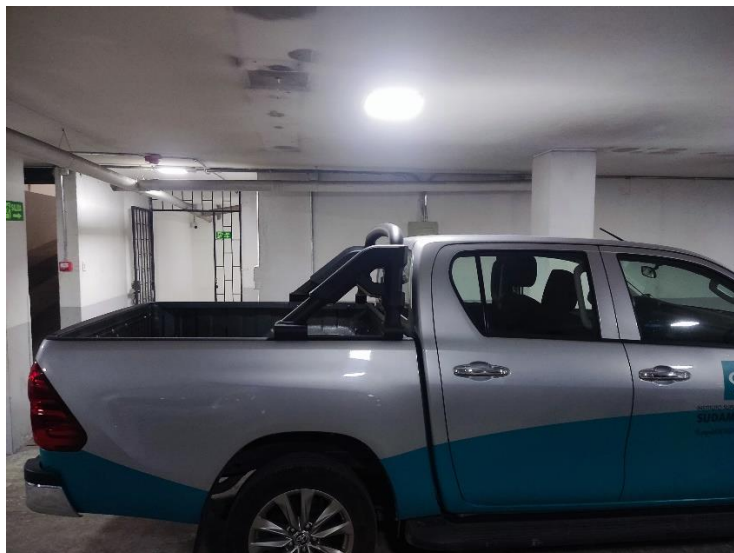


Figura 27. Ubicación del octavo puesto dentro del parqueadero junto con la entrada peatonal y caja eléctrica.

Fuente: El Autor.

Además, esta imagen representa, además de la ubicación del último parqueadero, la entrada peatonal y la caja eléctrica, que se utilizará en el futuro para realizar conexiones de suministro eléctrico.

Una vez realizada y proyectada una ubicación óptima se procede a determinar los espacios donde estarán ubicados los módulos del proyecto como se muestra en la figura.

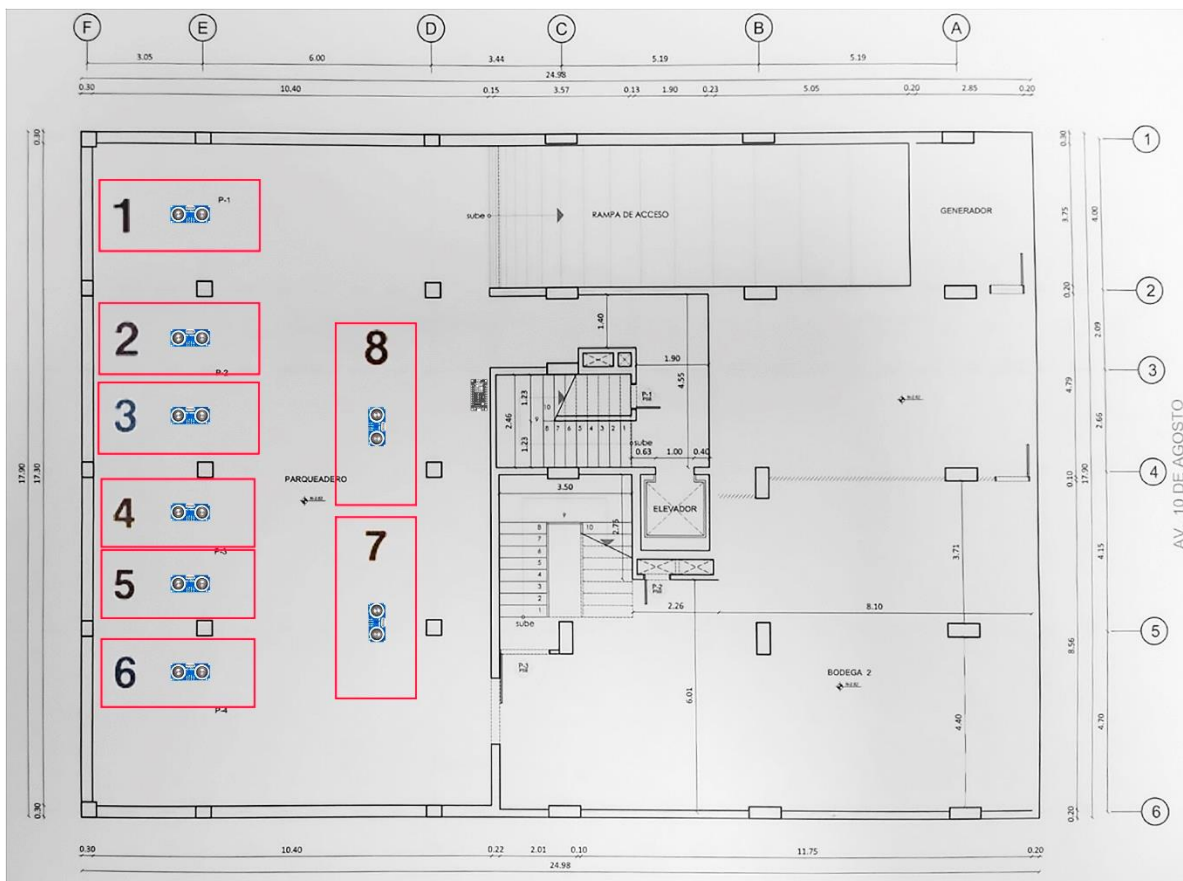


Figura 28. Plano de planta junto con el servidor y los módulos expresados con sensores ubicados encima de cada puesto
Fuente: El Autor.

En el modelo planteado los módulos estaban pensados para ser colocados frontal a los vehículos, pero debido al espacio reducido se prefirió optar por los sensores en la parte superior de cada puesto disponible ya que eso permitía tener uniformidad y mayor área para ser manipulados. El ancho de cada parqueadero está en aproximadamente 2.18 metros, por lo que el sensor se ubicó a la mitad de la distancia mencionada, quiere decir a 1.09 metros y de largo fue emparejado paralelamente con cada columna, en la siguiente figura se muestra una toma panorámica del parqueadero.

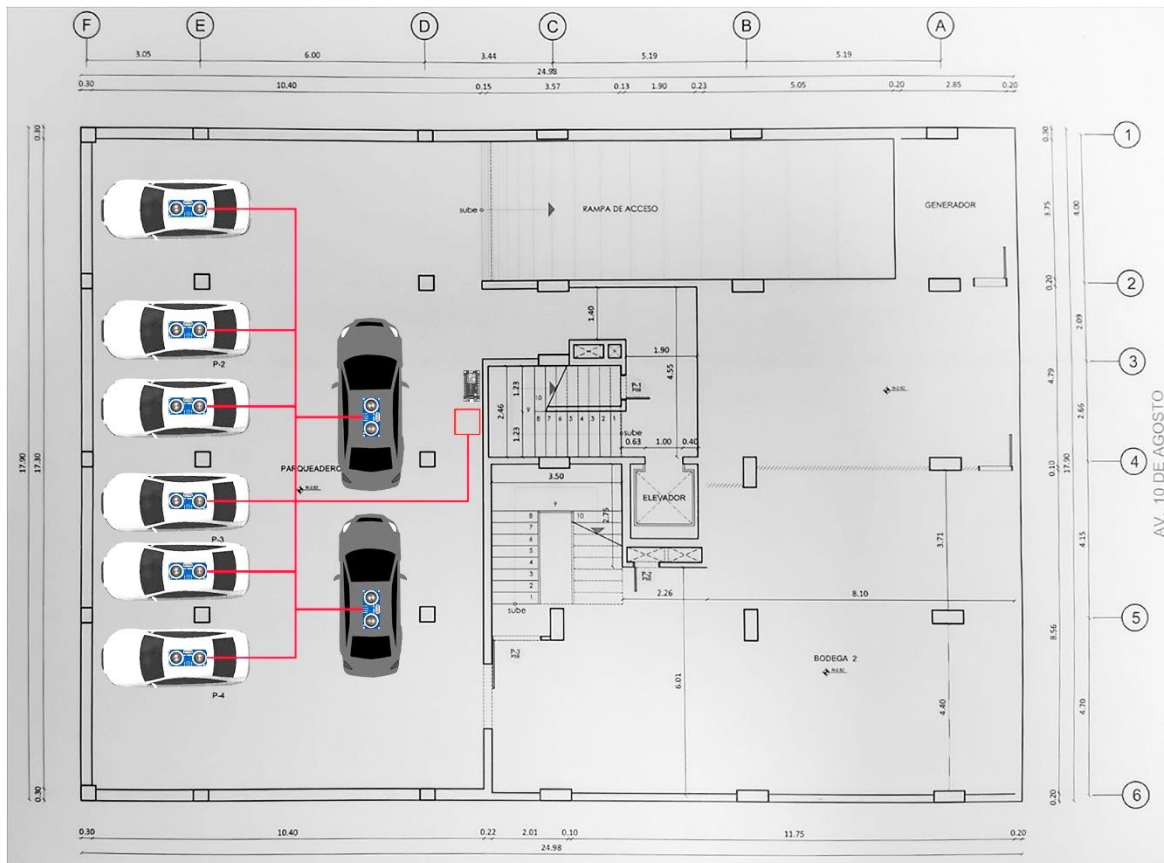


Figura 29. Plano de planta junto con la comitada eléctrica para el parqueadero.
Fuente: El Autor.

Para poder realizar esto es necesario colocar soportes metálicos que sirvan como canaletas para los cables que transmitirán la corriente junto con una toma de corriente tipo hembra para cada estación, y el cable debe ser resistente, adecuado para exteriores con un calibre de entre 8 a 12 AWG, de alrededor de unos 31 o 32 metros de largo para cubrir todas las áreas necesarias dentro del parqueadero de la Institución según las medidas del plano.

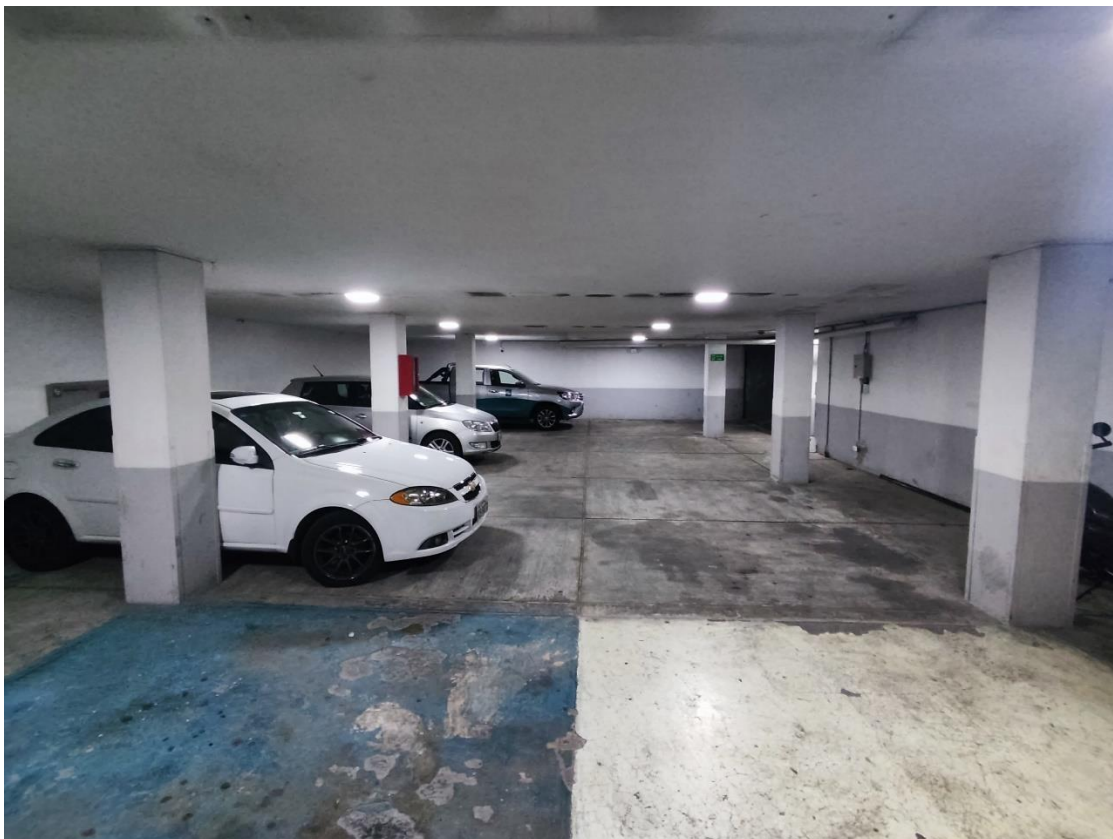


Figura 30. Vista panorámica de todo el parqueadero.

Fuente: El Autor.

La altura máxima que tiene el espacio mostrado en la figura anterior es de 2.45 metros.

La idea planteada fue de utilizar plafones planos que estén hechos en la parte frontal donde se ubicará el sensor de acrílico para poder ser maniobrado en caso de no disponer de un modelo que se adapte al diseño del módulo, y el resto de la cobertura de plástico o acero delgado para evitar interferencias y dimensiones que se adecuen al tamaño como se muestran en la figura.



Figura 31. Diseño de plafón.
Fuente: El Autor.

El objetivo del diseño es tener un plafón que tenga una buena altura para poder ser maniobrado el módulo, con especificaciones como las que se muestran en la siguiente figura.

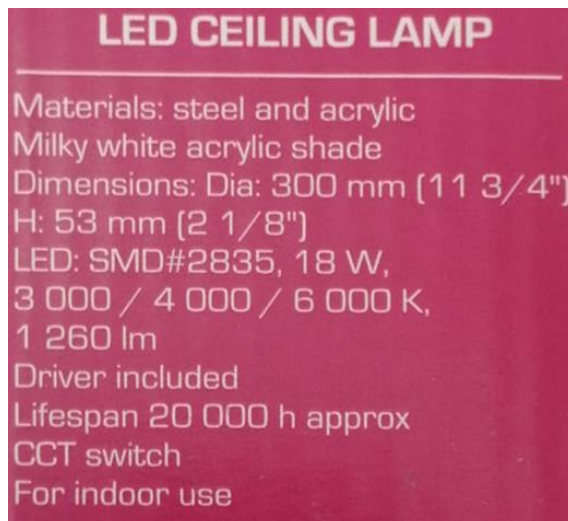


Figura 32. Ficha técnica del plafón.
Fuente: El Autor.

Las especificaciones pueden ir adecuadas a los gustos del comprador, lo más importante es disponer del espacio necesario para que el módulo pueda estar dentro, las dimensiones del módulo son 22.2 cm (largo) x 13.7 cm (ancho) x 7.5 cm (altura), también es posible aprovechar

las luces del plafón para ser configuradas de la misma forma que se encienden los leds dentro del proyecto para tener un diseño limpio y eficiente, además que junto con la toma puesta puede igualmente ser aprovechada. Existe diversidad de diseños de plafones que van según los distintos fabricantes de dichos productos como se muestra en la imagen.



Figura 33. Diseño de plafón alternativo.
Fuente: El Autor.

O incluso diseños con un estilo más moderno que disponen de otras luces distintas, como se muestra en la figura.



Figura 34. Diseño de plafón moderno.
Fuente: El Autor.

Con la característica en común de que son utilizados para interiores, de igual manera el diseño redondo es solo representativo, ya que es aceptable utilizar un diseño rectangular conforme a las dimensiones del módulo.

La ventaja de ofrecer el sensor con cableado es justamente la posición a la cual puede estar orientado el mismo ya que tiene libertad para poder ser movido, a diferencia del resto del módulo que si es recomendable de mantener en la posición original. Junto con una posición relativa del servidor que debe estar ubicado cerca o dentro de la caja eléctrica para poder recibir señal para conectar por wifi y también la alimentación particular que necesita.

Debido a que el estacionamiento no cuenta con tomas directas a la corriente, el proyecto inicialmente operara por el uso de baterías recargables de 9 voltios, pero en un futuro se espera poder cambiar a todo un cableado eléctrico. Para ello será necesario implementar una cometida eléctrica que permita identificar por donde es necesario montar cables para que cada puesto tuviera una salida de 110 voltios en corriente alterna con una frecuencia de 50-60 Hz que permitiese alimentar cada módulo como se muestra en la figura.

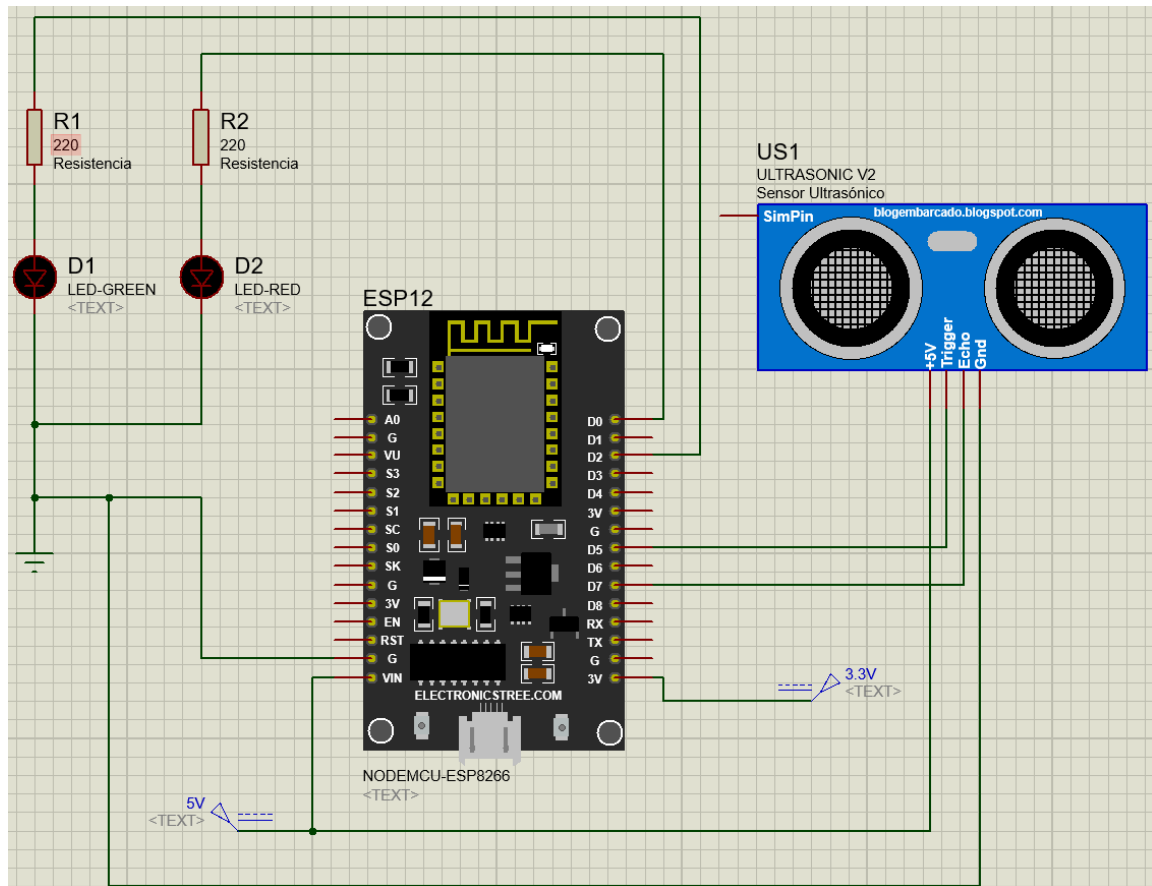


Figura 35. Esquema de conexión eléctrica del módulo cliente en Proteus.
Fuente: El Autor.

Como se indica en la figura anterior, fue necesario también explicar el sistema de conexiones eléctricas aplicado en el proyecto para cada módulo. Esto permite visualizar la importancia y distribución adecuada para evitar fallos o cortocircuitos que puedan dañar los equipos físicos. Cabe destacar que el servidor solo requiere energizar cualquiera de los pines de voltaje, ya sea el de 5V o el de 3.3V, pero para el presente proyecto fueron utilizados ambos pines para evitar fallas energéticas lo que pudieran provocar algún fallo.

Para el servidor, se utilizó el mismo módulo NodeMCU por su capacidad para escribir información en su memoria flash y su tecnología Wifi, que actúa como puente entre los sensores y el portal web. El servidor, al requerir únicamente conectividad Wifi, necesita una alimentación de 3.3V como mínimo. Esto puede lograrse conectando directamente al pin

correspondiente de 3.3V, o utilizando el pin VIN si se dispone únicamente de una corriente de 5V, utilizando este pin como filtro, como se muestra en la figura.

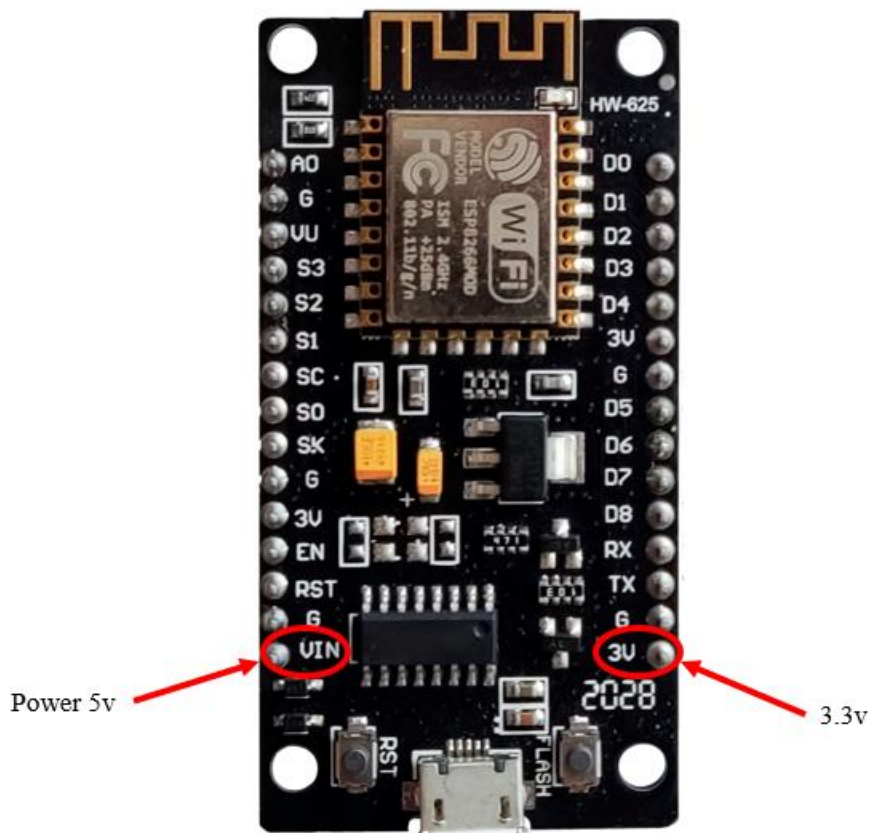


Figura 36. Esquema de pines de energía del NodeMCU V3 ESP8266.
Fuente: El Autor.

Para asegurar una correcta alimentación del módulo, se utilizaron ambos pines de energía para evitar inconvenientes. La antena del NodeMCU tiene un alcance teórico de 30 metros, aunque es posible mejorar su rendimiento incorporando antenas externas de transmisión más potentes o utilizando repetidores de señal.

Con esta estructura definida para el proyecto, se procede al ensamblaje de cada uno de los elementos necesarios para el sistema. Para la estructuración del módulo físico, se consideró el tamaño del NodeMCU, que requiere dos protoboards unidos, ya que la longitud del circuito impreso es mayor que el ancho de un solo protoboard, como se muestra en la figura.

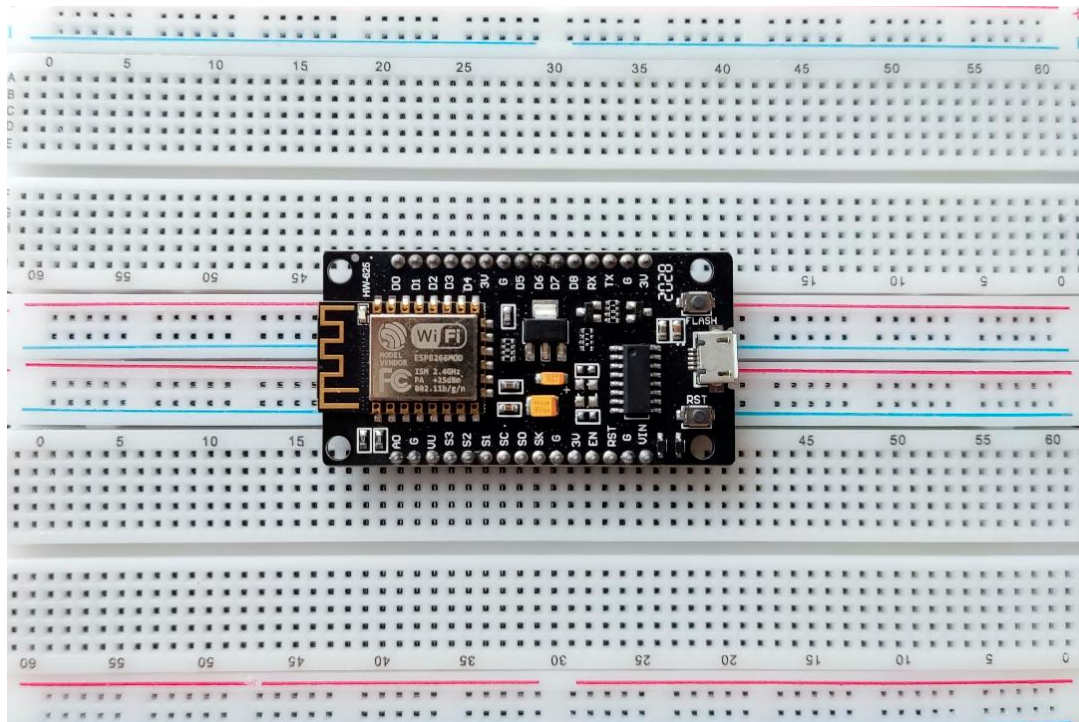


Figura 37. Instalación del NodeMCU en el protoboard.
Fuente: El Autor.

Los rieles de alimentación y tierra están divididos en 8 áreas claramente identificadas mediante colores y pines de conexión hembra para cada módulo individual.

Inicialmente, se utilizó Arduino como puente de comunicación RX-TX para el ESP-01, ya que este último solo dispone de estos pines y la placa Arduino permitía convertir la señal para la programación directa desde una computadora mediante un cable. Sin embargo, conforme el proyecto avanzó, se encontraron limitaciones significativas con el ESP-01 debido a problemas de capacidad de memoria y errores durante la sobrescritura del código. Por ello, se optó por una placa con un chip ESP más actualizado, permitiendo la escritura de un código más elaborado.

Aunque existen alternativas más potentes como el ESP32, capaz de soportar más dispositivos conectados simultáneamente y con mayor memoria, se determinó que el modelo elegido era ideal para cubrir las necesidades específicas del proyecto de gestión de espacios de

estacionamiento en la institución, ajustándose al presupuesto disponible. El ESP32, aunque más avanzado, resultaba más costoso y su capacidad no se utilizaría completamente en este contexto.

Para la medición de la distancia, se implementó un sensor de ultrasonido HC-SR04, que calcula el tiempo que tarda la señal de eco en rebotar del emisor al receptor. Este sensor fue elegido por su facilidad de uso, la documentación clara proporcionada por el fabricante para configurar correctamente los valores de funcionamiento, y su popularidad en proyectos similares. Existe suficiente material y apoyo de la comunidad de Arduino para interpretar y guiar el uso de este sensor de manera efectiva.

Además, se incorporaron dos LEDs, uno rojo y otro verde, con resistencias de 220 Ohm, para proporcionar indicadores visuales de las medidas tomadas por el sensor según la configuración establecida en la programación, como se ilustra en la figura correspondiente.

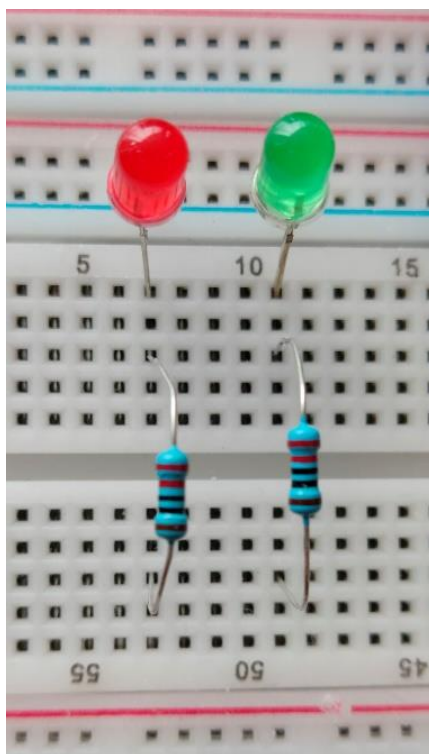


Figura 38. Esquema de conexión de leds y resistencias en el protoboard.
Fuente: El Autor.

El cátodo (-) del LED debe conectarse directamente a tierra, mientras que el ánodo (+) se conecta a través de una resistencia a la fuente de energía. Es importante destacar que las resistencias no tienen polaridad, por lo que su instalación no requiere considerar su orientación.

Para identificar físicamente el ánodo y el cátodo en un LED, existen dos métodos. El primero y más sencillo es observar que el pin del ánodo suele ser más largo que el del cátodo. En caso de que no sea así, el segundo método implica verificar dentro del LED, donde dos piezas separadas están presentes: una más corta (ánodo) y otra más larga (cátodo).

Además, se agregó un componente adicional, el módulo de alimentación MB102, como se muestra en la figura correspondiente.

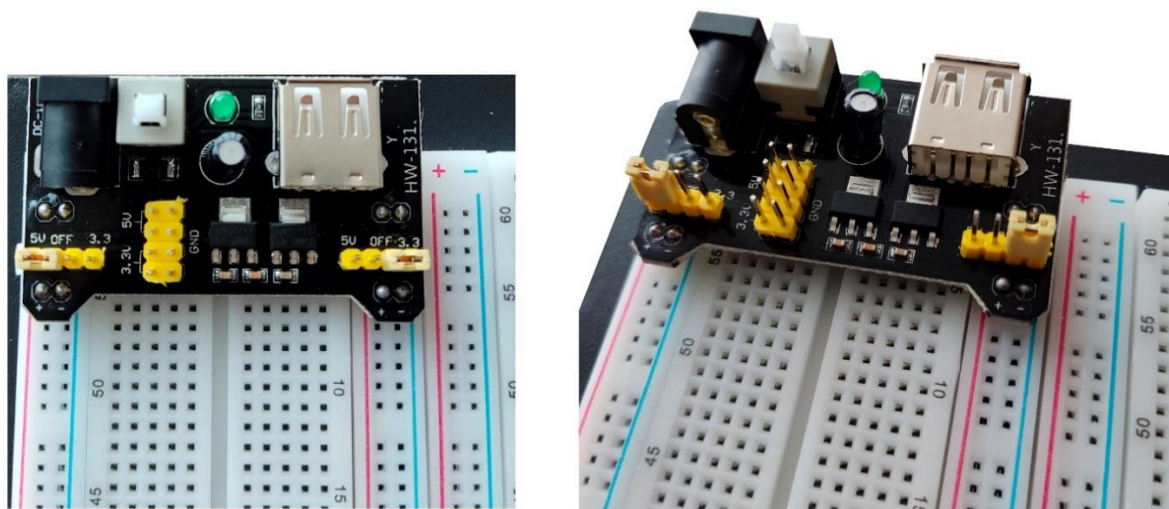


Figura 39. Conexión del módulo MB-102 en el protoboard.
Fuente: El Autor.

El módulo MB102 permitió la regulación de la corriente suministrada para adaptarse a las necesidades de los equipos conectados en cada módulo. Puede recibir una entrada de energía entre 6.5V y 12V, proporcionando salidas de voltajes configurables de 3.3V y 5V mediante una funda con una paleta metálica que facilita la configuración manual. La placa incluye especificaciones que permiten seleccionar el tipo de voltaje deseado para las líneas de alimentación conectadas a la protoboard, así como la opción de inhabilitarlas para permitir el flujo de corriente sin restricciones. Además, ofrece una corriente máxima de 700mA. En caso

de que los rieles de conexión no sean suficientes, el módulo también dispone de pines tipo macho que proporcionan salidas de 3.3V, 5V y tierra, diseñados para ser utilizados con cables conectados directamente al módulo de alimentación. Este módulo también cuenta con un LED indicador de estado de energización y un interruptor para controlar el flujo de corriente hacia la protoboard, lo que facilita realizar un corte energético en caso de fallas físicas.

Para las conexiones realizadas dentro de cada módulo, se utilizaron cables jumpers para protoboard de dos tipos. El más comúnmente utilizado fue el tipo macho-macho, mientras que también se empleó el tipo macho-hembra. Estos cables fueron especialmente útiles para permitir que el sensor ultrasónico no permanezca estático a la altura de la protoboard, adaptándolo así a un diseño específico según las necesidades del entorno.

Los colores utilizados para las diferentes conexiones fueron seleccionados conforme a estándares comúnmente empleados en fuentes de poder de computadoras: rojo para energía de 5V, naranja para energía de 3.3V, negro y marrón para tierra (GND). Además, se utilizaron azul y verde para las conexiones de los LEDs, blanco y gris para la señal del eco del sensor, y morado para la señal del sensor.

Una vez planteada la auto regulación del proyecto, se procede a incorporar un sistema de energía, inicialmente pensado en una batería de 9 voltios como la que se muestra en la figura.



**Figura 40. Batería de 9 voltios y salida de 600 miliamperios.
Fuente: El Autor.**

Esta batería permitirá energizar el módulo regulador de voltaje con funcionalidad autónoma junto con un adaptador que conectará la batería al módulo MB-102 como se muestra en la figura.



**Figura 41. Soporte con adaptador de batería de 9 voltios con salida de plug universal de 5.5 mm y 2.1 mm.
Fuente: El Autor.**

Este adaptador contará con el socket para meter la batería recargable junto con el plug de 5.5 mm que da la salida de voltaje.

Por último, cuando exista la posibilidad de cambiar el sistema de energía de las baterías a la corriente eléctrica, será necesario utilizar un adaptador de corriente de 9V que permitirá recibir la corriente alterna y transformarla en continua al voltaje que enviaba la batería anteriormente, como se muestra en la figura correspondiente.



**Figura 42. Fuente AC DC de 9 voltios con salida de plug de 5.5 mm y 2.1mm.
Fuente: El Autor.**

La fuente se conecta directamente al módulo MB102 y a la toma de corriente alterna, donde convierte la energía en corriente continua. Una vez completado el montaje de un módulo individual, debería tener el aspecto que se muestra en la figura correspondiente.

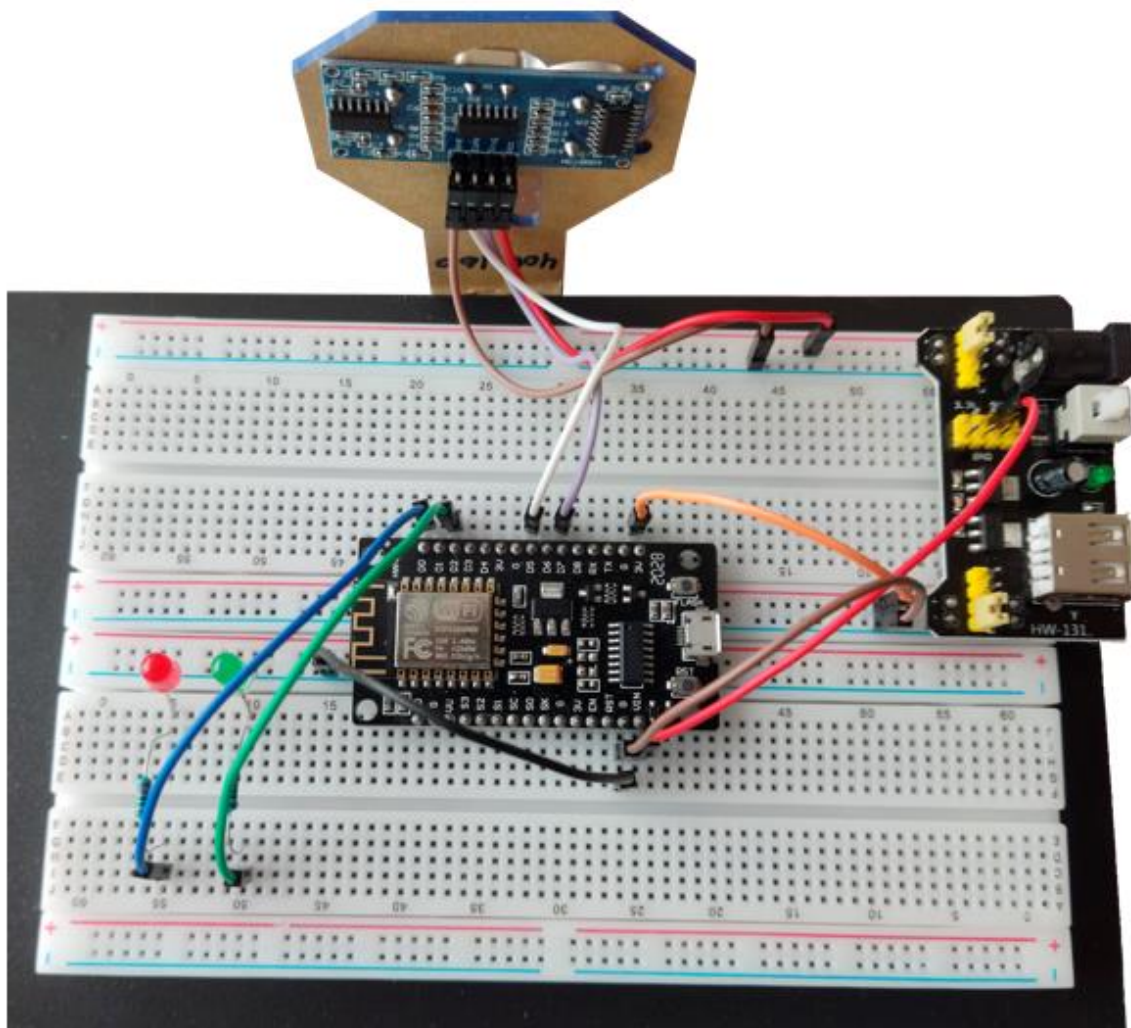


Figura 43. Montaje final de un módulo para parqueadero.
Fuente: El Autor.

Además de los pines de alimentación (VIN, 3.3V y GND), en el NodeMCU se utilizan también los pines D0, D2, D5 y D7, aunque cualquier otro pin disponible podría ser utilizado. Estos pines fueron seleccionados en el proyecto por su conveniencia y distribución dentro del módulo.

El NodeMCU deberá estar protegido por un recubrimiento para salvaguardar toda la circuitería contenida, como se muestra en la figura correspondiente.

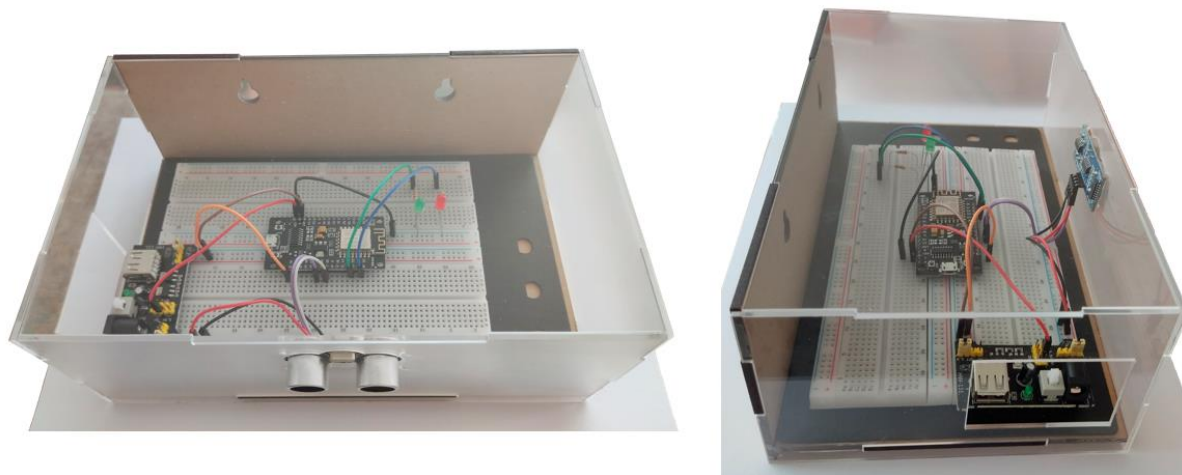


Figura 44. Caja de recubrimiento para los módulos.
Fuente: El Autor.

El refuerzo de la caja fue pensado por el tamaño de la base y la altura de las conexiones para evitar desgastes en el tiempo con un soporte especial para el sensor ultrasónico. Es importante destacar que las columnas que sostienen el edificio y el recubrimiento de todo el estacionamiento están hechas de un material inhibidor de ondas de radiofrecuencia, lo que puede afectar las comunicaciones Wifi dependiendo del posicionamiento de los módulos con respecto al servidor. Inicialmente, los módulos están orientados en una posición recta en relación a la forma de estacionar los vehículos, pero esto puede ser ajustado según la necesidad. Por ejemplo, en estacionamientos donde no todos los puestos disponen de una pared, podría ser mejor implementarlo desde el techo de cada uno.

Equipos	Precio
ESP8266 Node MCU V3	13,44 \$
Sensor Ultrasónico	6,00 \$
Diodos leds	0,30 \$
Resistencias	0,05 \$
Protoboard HD-BB-2T3D	12,30 \$
Cables jumper macho-macho	3,00 \$
Cables jumper macho-hembra	3,00 \$
Caja de protección y soporte	16,00 \$
Módulo MB-102	5,00 \$
Soporte de batería de 9V	1,50 \$
Batería de 9V	3,39 \$
Total	63,98 \$

Tabla 1. Costo de un módulo con baterías.
Fuente: El Autor.

La tabla anterior muestra el costo total que representa un módulo individual energizado por batería.

Equipos	Precio
ESP8266 Node MCU V3	13,44 \$
Sensor Ultrasónico	6,00 \$
Diodos leds	0,30 \$
Resistencias	0,05 \$
Protoboard	12,30 \$
Cables jumper macho-macho	3,00 \$
Cables jumper macho-hembra	3,00 \$
Caja de protección y soporte	16,00 \$
Módulo MB-102	5,00 \$
Fuente AC/DC	4,47 \$
Total	63,56 \$

Tabla 2. Costo de un módulo con fuente de alimentación.
Fuente: El Autor.

La tabla anterior muestra el costo total que representa un módulo individual energizado por adaptador AC/DC, por lo que a partir de esto se puede determinar el costo final incluyendo servidor y módulos adecuados al espacio.

Materiales y equipo		
Especificaciones	Cantidad	Precio total
Servidor	1	13,44 \$
Módulos con batería	8	511,80 \$
Módulos con fuente de poder	8	508,44 \$

Tabla 3. Materiales y equipo necesarios para la instalación en el parqueadero del Instituto Superior Tecnológico Sudamericano.
Fuente: El Autor.

En la tabla anterior se muestra el precio total únicamente de todos los módulos requeridos para el parqueadero del Instituto con su adaptación para baterías y con fuente de poder.

Costos Adicionales		
Especificaciones		Precio
Mano de obra	32	288,00 \$
Transporte		25,00 \$
Beneficio módulos con batería	30%	251,47 \$
Beneficio módulos con fuente de poder	30%	250,46 \$

Tabla 4. Costos adicionales.
Fuente: El Autor.

En la tabla anterior se contemplan costos adicionales que se tomarán en cuenta para el cálculo del presupuesto final.

Calculo Final	
Especificaciones	Precio total
Subtotal con baterías	838,24 \$
Subtotal con fuente de poder	834,88 \$
Costo total con batería	1089,71 \$
Costo total con fuente de poder	1085,34 \$

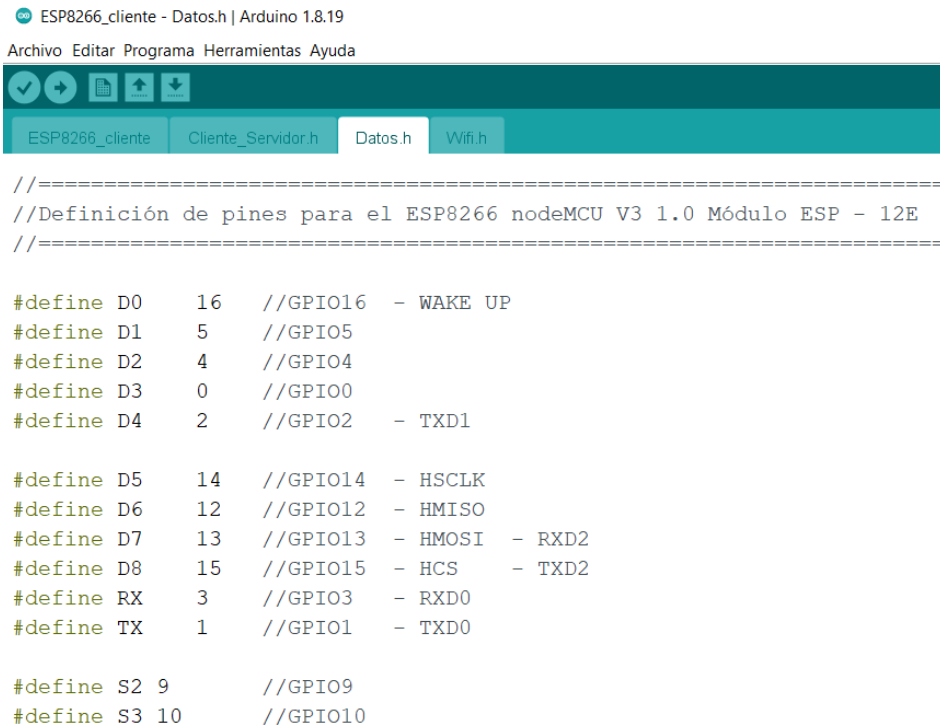
Tabla 5. Costo total del proyecto dentro del parqueadero del Instituto Superior Tecnológico Sudamericano.
Fuente: El Autor.

En la tabla anterior se muestra para el Instituto Superior Tecnológico Sudamericano el costo total para los módulos que se desean colocar según los 8 parqueaderos y del servidor.

Es importante tener en cuenta que los precios de los equipos pueden fluctuar con el tiempo, lo que puede afectar significativamente el presupuesto total del proyecto. Por lo tanto, es fundamental realizar una búsqueda actualizada de los equipos necesarios para asegurarse de que el presupuesto sea lo más preciso posible.

6.3. Implementación del sistema prototipo basado en el ESP8266 en el IDE de Arduino

Para la programación del funcionamiento del sensor, se definió la estructura de pines del módulo implementado en el entorno de desarrollo Arduino IDE, tal como se detalla en la figura correspondiente.



```

ESP8266_cliente - Datos.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_cliente Cliente_Servidor.h Datos.h Wifi.h

//=====
//Definición de pines para el ESP8266 nodeMCU V3 1.0 Módulo ESP - 12E
//=====

#define D0 16 //GPIO16 - WAKE UP
#define D1 5 //GPIO5
#define D2 4 //GPIO4
#define D3 0 //GPIO0
#define D4 2 //GPIO2 - TXD1

#define D5 14 //GPIO14 - HSCLK
#define D6 12 //GPIO12 - HMISO
#define D7 13 //GPIO13 - HMOSI - RXD2
#define D8 15 //GPIO15 - HCS - TXD2
#define RX 3 //GPIO3 - RXD0
#define TX 1 //GPIO1 - TXD0

#define S2 9 //GPIO9
#define S3 10 //GPIO10

```

Figura 45. Definición de pines para el NodeMCU V3 ESP8266.
Fuente: El Autor.

Estos pines fueron luego asignados a variables específicas, las cuales fueron programadas mediante una serie de funciones básicas en el entorno de desarrollo Arduino IDE para captar la señal de ultrasonido, como se ilustra en la figura correspondiente.



```

ESP8266_cliente - Datos.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_cliente Cliente_Servidor.h Datos.h Wifi.h

//=====
//Iniciación de pines para el funcionamiento del sensor
//=====

const uint8_t TRIG = D7; // Señal de transmisión
const uint8_t ECO = D5; // Señal de recepción
const uint8_t LED_ROJO = D0; // Led rojo
const uint8_t LED_VERDE = D2; // Led verde
int DURACION; // Variable para almacenar la información recibida del sensor
int DISTANCIA; // Variable utilizada para transformar a distancia según la información
// dada por el fabricante del sensor ultrasónico

uint8_t Dist_max = 40;
const uint8_t Dist_min = 2;

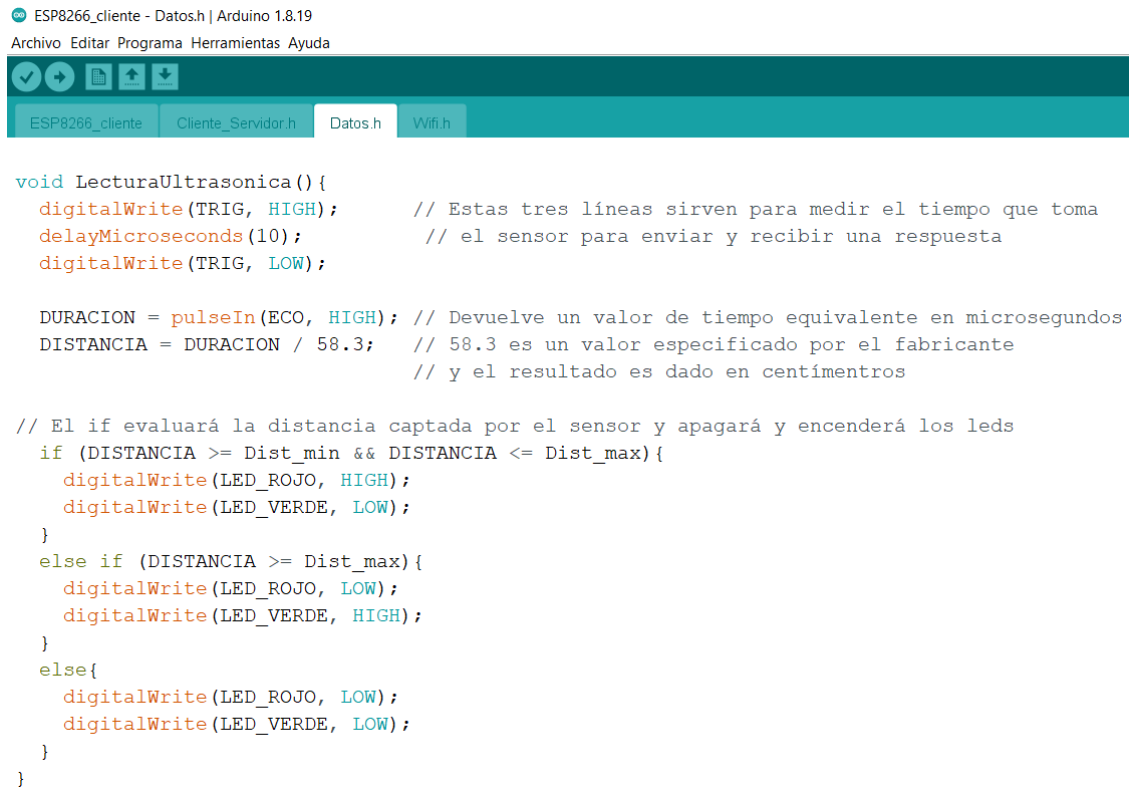
//=====
//Setting y funcionamiento del sensor
//=====

void SetPins(){
  pinMode(LED_ROJO, OUTPUT);
  pinMode(LED_VERDE, OUTPUT);
  pinMode(TRIG, OUTPUT);
  pinMode(ECO, INPUT);
}

```

Figura 46. Definición e inicialización de pines.
Fuente: El Autor.

Posteriormente, la información obtenida es convertida a centímetros de acuerdo con las especificaciones proporcionadas por el fabricante del sensor, sobre cómo transformar los valores capturados, como se representa en la figura.



```

ESP8266_cliente - Datos.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_cliente Cliente_Servidor.h Datos.h Wifi.h

void LecturaUltrasonica() {
  digitalWrite(TRIG, HIGH); // Estas tres líneas sirven para medir el tiempo que toma
  delayMicroseconds(10); // el sensor para enviar y recibir una respuesta
  digitalWrite(TRIG, LOW);

  DURACION = pulseIn(ECO, HIGH); // Devuelve un valor de tiempo equivalente en microsegundos
  DISTANCIA = DURACION / 58.3; // 58.3 es un valor especificado por el fabricante
  // y el resultado es dado en centímetros

  // El if evaluará la distancia captada por el sensor y apagará y encenderá los leds
  if (DISTANCIA >= Dist_min && DISTANCIA <= Dist_max) {
    digitalWrite(LED_ROJO, HIGH);
    digitalWrite(LED_VERDE, LOW);
  }
  else if (DISTANCIA >= Dist_max) {
    digitalWrite(LED_ROJO, LOW);
    digitalWrite(LED_VERDE, HIGH);
  }
  else {
    digitalWrite(LED_ROJO, LOW);
    digitalWrite(LED_VERDE, LOW);
  }
}

```

Figura 47. Función para el funcionamiento del sensor en los pines configurados.
Fuente: El Autor.

Después de esta etapa, fue necesario establecer toda la estructura de conexión Wifi para el cliente y el servidor. Inicialmente, se agregó la pila de comandos para los módulos ESP8266 y ESP32. Para obtener más detalles al respecto, se puede consultar el Anexo 7.

Para lograr esto, se debe copiar un enlace que apunta a un archivo .json del repositorio oficial de ESP8266 en GitHub. El enlace específico es https://arduino.esp8266.com/stable/package_esp8266com_index.json. Es crucial verificar que este enlace siga funcionando correctamente, ya que durante el desarrollo del proyecto se encontró que el enlace había sido modificado en varias actualizaciones. Para obtener la versión actualizada, se debe visitar el repositorio de ESP8266, donde el creador actualiza la información en la sección correspondiente. Este enlace permite la instalación y gestión de placas externas programables con el IDE de Arduino.

Una vez confirmado el enlace, debe ser pegado en la casilla de texto dentro del entorno de desarrollo, accediendo a la sección Archivo y luego Preferencias, como se muestra en la figura correspondiente.

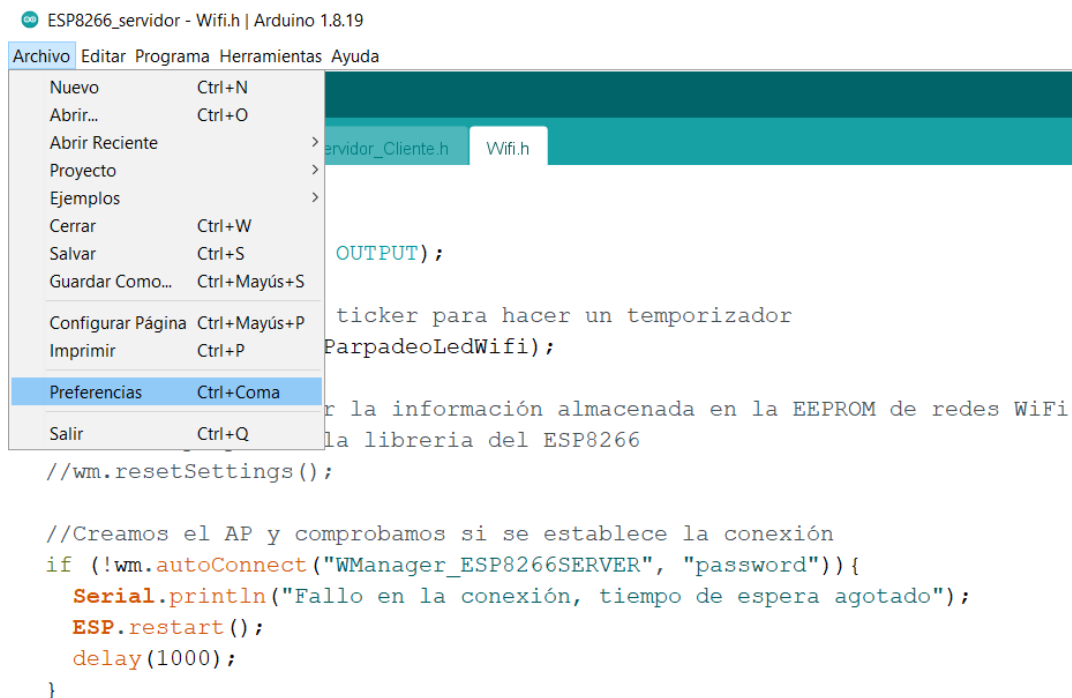


Figura 48. Ruta para el anexo de gestor de tarjetas adicionales.
Fuente: El Autor.

Una vez en la sección de Preferencias, se procede a pegar el enlace en la caja de texto destinada para el gestor de URL adicionales de tarjetas, tal como se indica en la figura correspondiente.

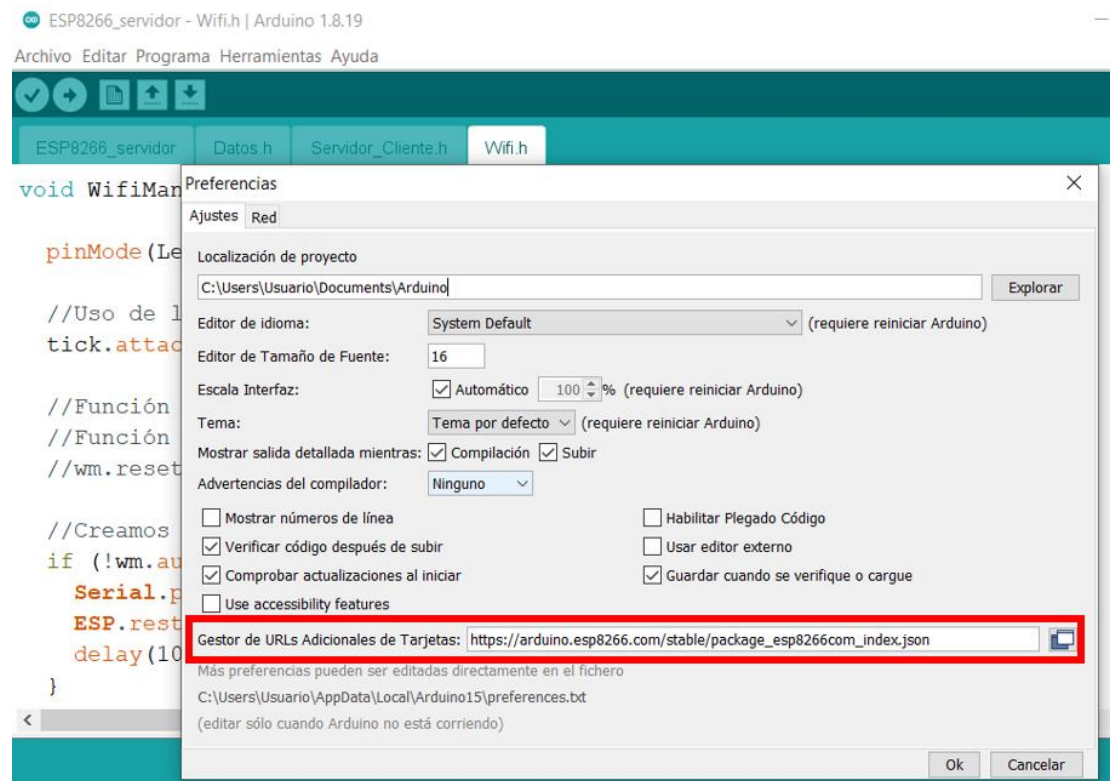


Figura 49. Ubicación para adjuntar el enlace correspondiente a tarjetas adicionales distintas de Arduino. Fuente: El Autor.

Una vez completado este paso, se puede proceder a descargar la configuración para las diferentes placas ESP8266 desde el gestor de tarjetas de Arduino. Esto se realiza accediendo a la sección de Herramientas en el entorno de Arduino, seleccionando la última tarjeta configurada, y eligiendo la opción de Gestor de Tarjetas, como se describe en la figura correspondiente.

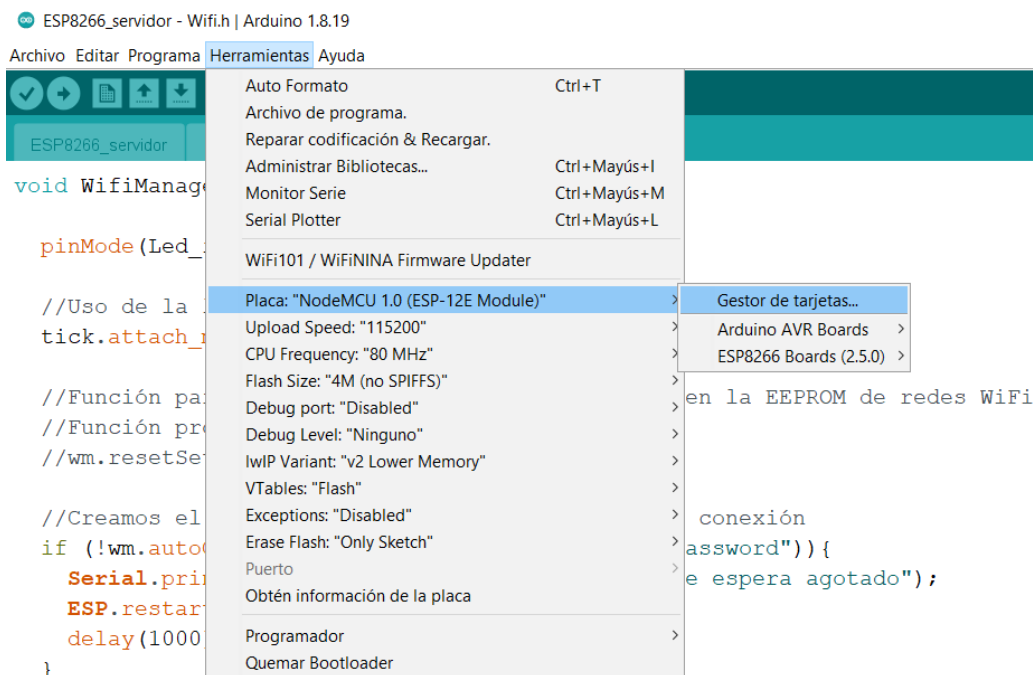


Figura 50. Ruta del gestor de tarjetas en Arduino IDE.
Fuente: El Autor.

Para encontrar la configuración necesaria, se utiliza el buscador dentro del gestor de tarjetas de Arduino y se introduce "ESP8266". Automáticamente aparecerá la opción que se debe instalar, como se muestra en la figura correspondiente.

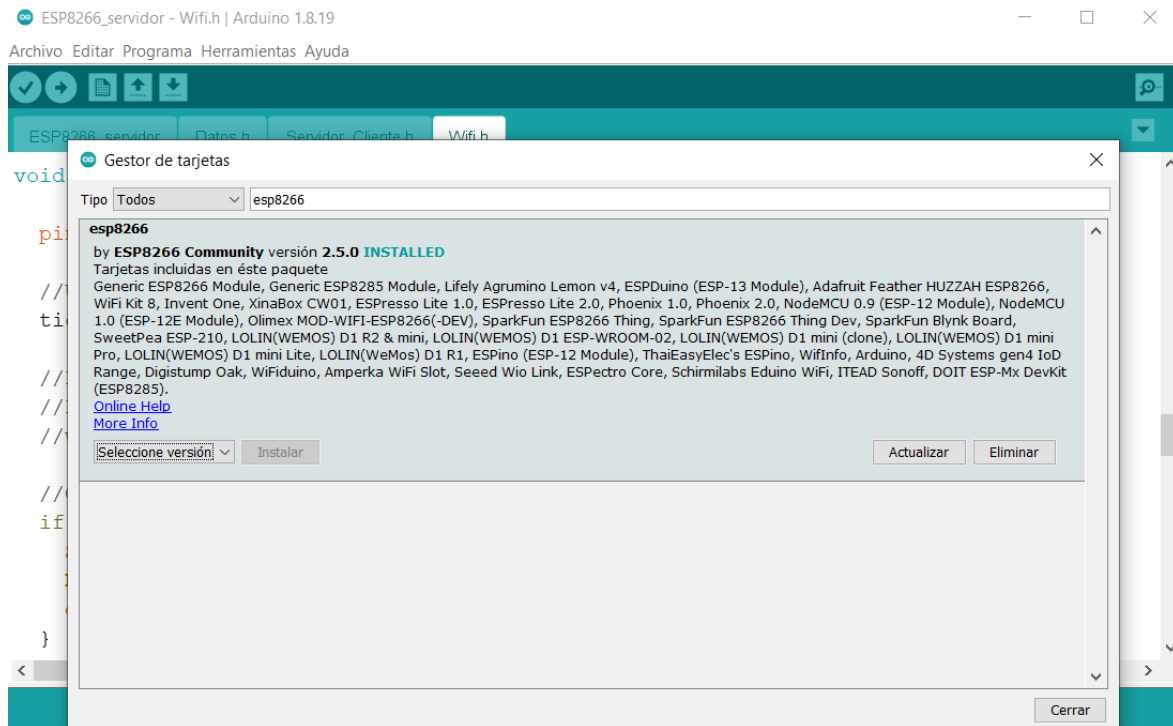


Figura 51. Instalación del gestor de tarjetas adecuado para ESP8266.
Fuente: El Autor.

También se incluye información relevante que puede ser útil para resolver problemas específicos, como la versión utilizada y las tarjetas compatibles con el gestor. Después de completar todo el proceso, se puede seleccionar la versión de la tarjeta deseada desde la sección de Herramientas.

Una vez finalizada la configuración de las placas, se procedió a definir las librerías necesarias para la conectividad. Esto incluyó la implementación de las librerías Wifi y Wifi multi, así como la librería UDP para el manejo de paquetes de información entre dispositivos, como se muestra en la figura correspondiente.



```

ESP8266_cliente - Wifi.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_cliente Cliente_Servidor.h Datos.h Wifi.h
#if defined(ESP32)

//Librerias para ESP32
#include <WiFi.h>
#include <HTTPClient.h>
#include <WiFiMulti.h>
WiFiMulti WifiMulti;

#elif defined(ESP8266)

//Librerias para ESP8266
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <ESP8266WiFiMulti.h>
ESP8266WiFiMulti WifiMulti;
#include <WiFiUDP.h>
WiFiUDP Udp_sending, Udp_receiving;

#endif

```

Figura 52. Librerías definidas para el módulo cliente.
Fuente: El Autor.

Finalmente, se incorporaron las librerías Wifi Manager y mDNS exclusivamente para el servidor, como se indica en la figura correspondiente.



```

ESP8266_servidor - Wifi.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_servidor Datos.h Servidor_Cliente.h Wifi.h
#elif defined(ESP8266)

//Librerias para ESP8266
#include <ESP8266WiFi.h>
#include <ESP8266mDNS.h>
#include <DNSServer.h>
#include <ESP8266WebServer.h>
#include <WiFiUDP.h>
//#include <ESPAsyncWebServer.h>

//Establecer el módulo ESP8266 como servidor
//AsyncWebServer server(80); //Puerto 80 (http)
ESP8266WebServer server(80);
WiFiUDP Udp_sending, Udp_receiving;

#endif

#include <WiFiManager.h>

//WiFiManager, creación de un objeto
WiFiManager wm;

```

Figura 53. Librerías definidas para el servidor.
Fuente: El Autor.

Para agregar la librería Wifi Manager, existen tres métodos disponibles. El primero consiste en descargar la librería desde el repositorio oficial de GitHub en un archivo .zip, como se muestra en la figura correspondiente.

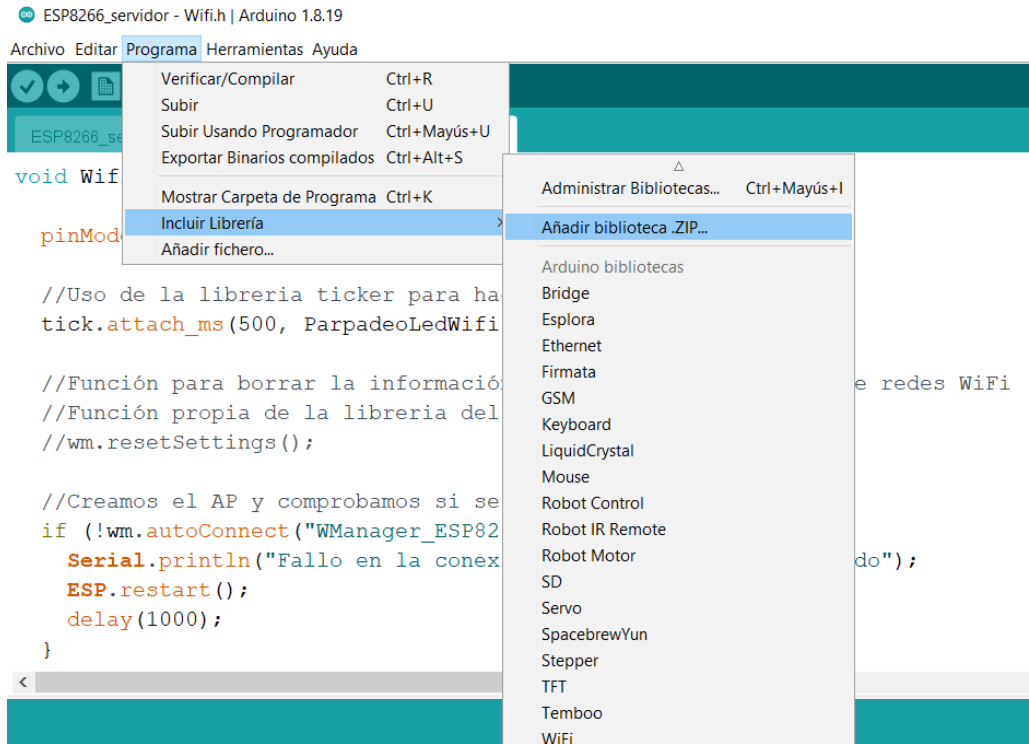


Figura 54. Ruta del Arduino IDE para incorporar una librería manualmente.
Fuente: El Autor.

Para obtener más información, se puede consultar el portal oficial de GitHub en el Anexo 8. Una vez que se haya descargado el archivo .zip en la computadora y accediendo a la ruta desde el Arduino IDE, solo será necesario seleccionarlo para que se cargue automáticamente en las librerías disponibles dentro del entorno. El segundo método consiste en buscar directamente la librería entre las disponibles para instalar en el mismo catálogo que ofrece el Arduino IDE, como se muestra en la figura.

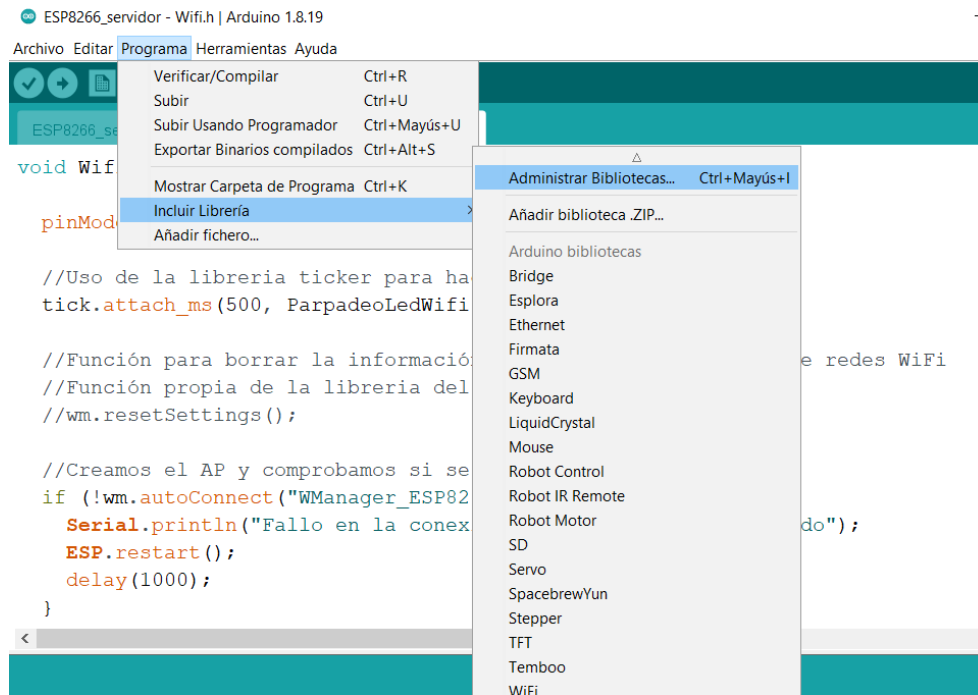


Figura 55. Ruta para acceder al gestor de librerías de Arduino IDE.
Fuente: El Autor.

Al ingresar directamente a la opción "Administrar bibliotecas" se abrirá la ventana que se muestra en la siguiente figura.

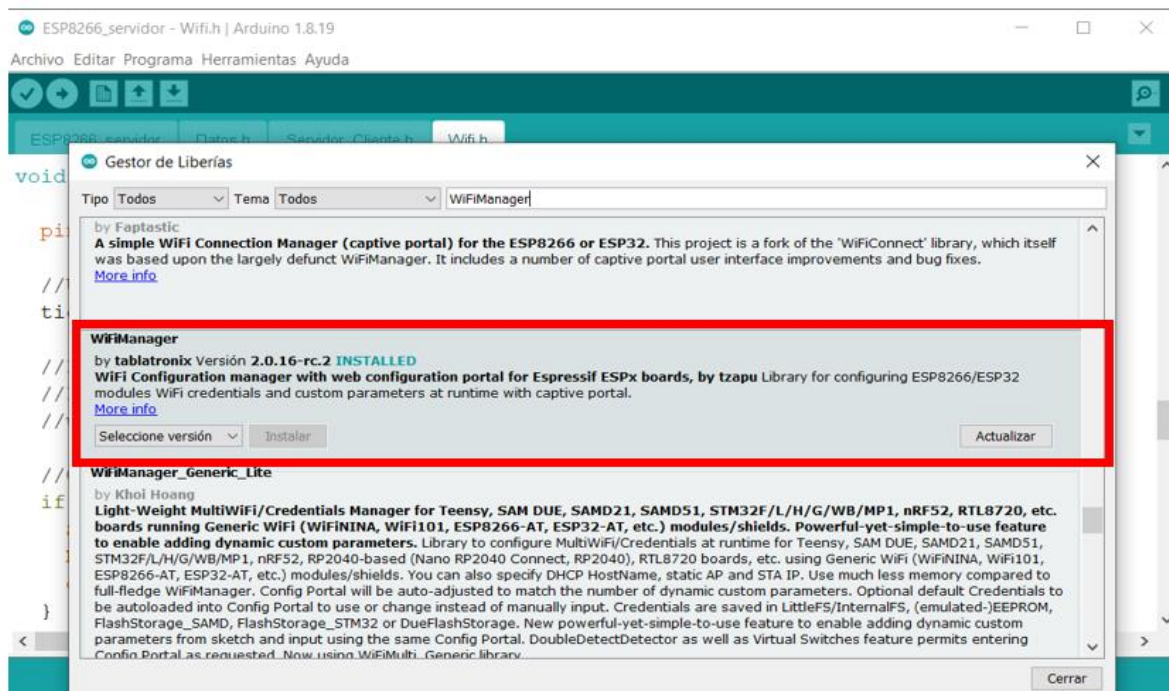


Figura 56. Librería oficial de Wifi Manager.
Fuente: El Autor.

Este método busca automáticamente la referencia del repositorio de GitHub donde está ubicada la librería, permitiendo descargarla o actualizarla a través del gestor de librerías. La tercera forma, aunque menos recomendable, implica copiar todo el contenido de la librería original en un archivo con la misma extensión que las librerías de Arduino, para luego incorporarlo manualmente al proyecto.

Wifi Manager se utilizó para configurar el servidor como un dispositivo cliente que se conecta dinámicamente a través de una página web, donde se ingresan las credenciales de una red Wifi, como se muestra en la figura correspondiente.

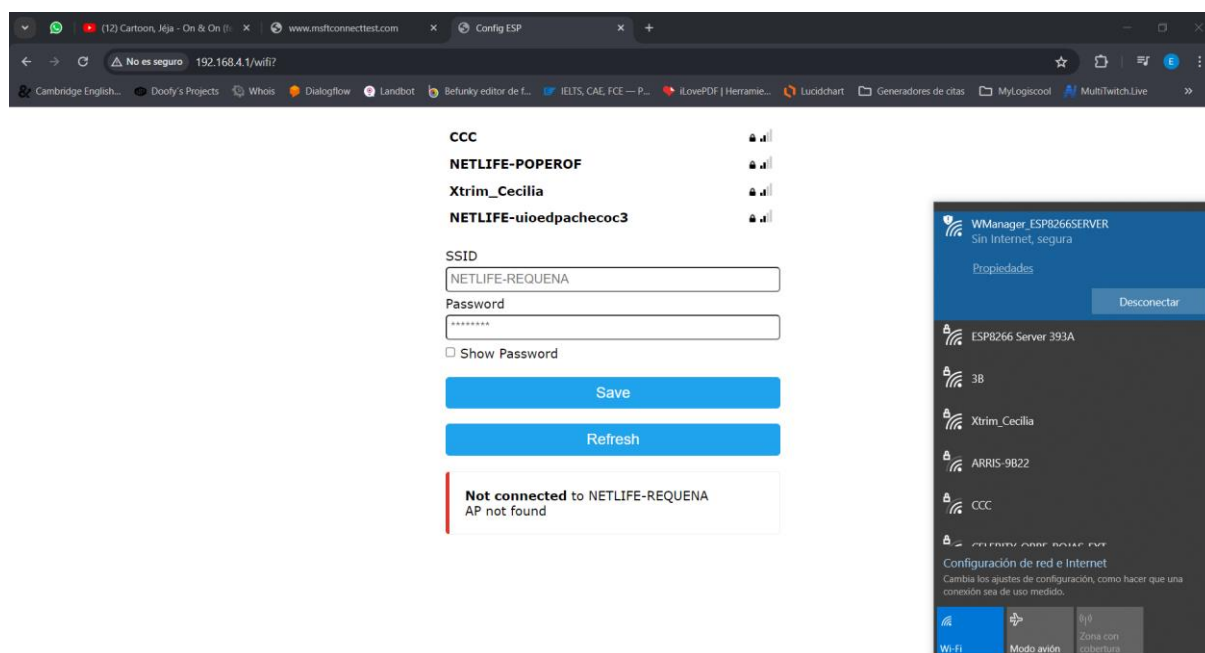
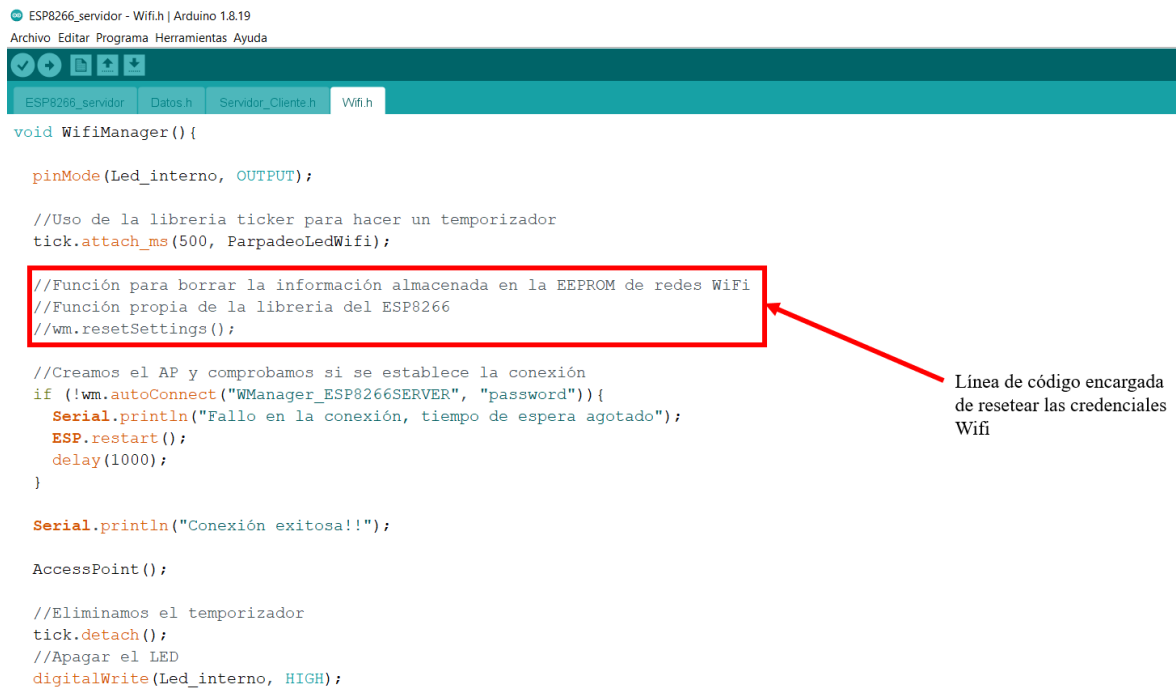


Figura 57. Página de configuración de una red Wifi por Wifi Manager.
Fuente: El Autor.

La información ingresada desde el portal web servirá para la próxima conexión, incluso en caso de desconexión o reinicio, a menos que se incluya una línea de código que borre la configuración guardada del Wifi Manager, como se muestra en la figura correspondiente.



```

ESP8266_servidor - Wifi.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_servidor Datos.h Servidor_Cliente.h Wifi.h
void WifiManager(){
    pinMode(Led_interno, OUTPUT);

    //Uso de la libreria ticker para hacer un temporizador
    tick.attach_ms(500, ParpadeoLedWifi);

    //Función para borrar la información almacenada en la EEPROM de redes WiFi
    //Función propia de la librería del ESP8266
    wm.resetSettings();

    //Creamos el AP y comprobamos si se establece la conexión
    if (!wm.autoConnect("WManager_ESP8266SERVER", "password")){
        Serial.println("Fallo en la conexión, tiempo de espera agotado");
        ESP.restart();
        delay(1000);
    }

    Serial.println("Conexión exitosa!!");

    AccessPoint();

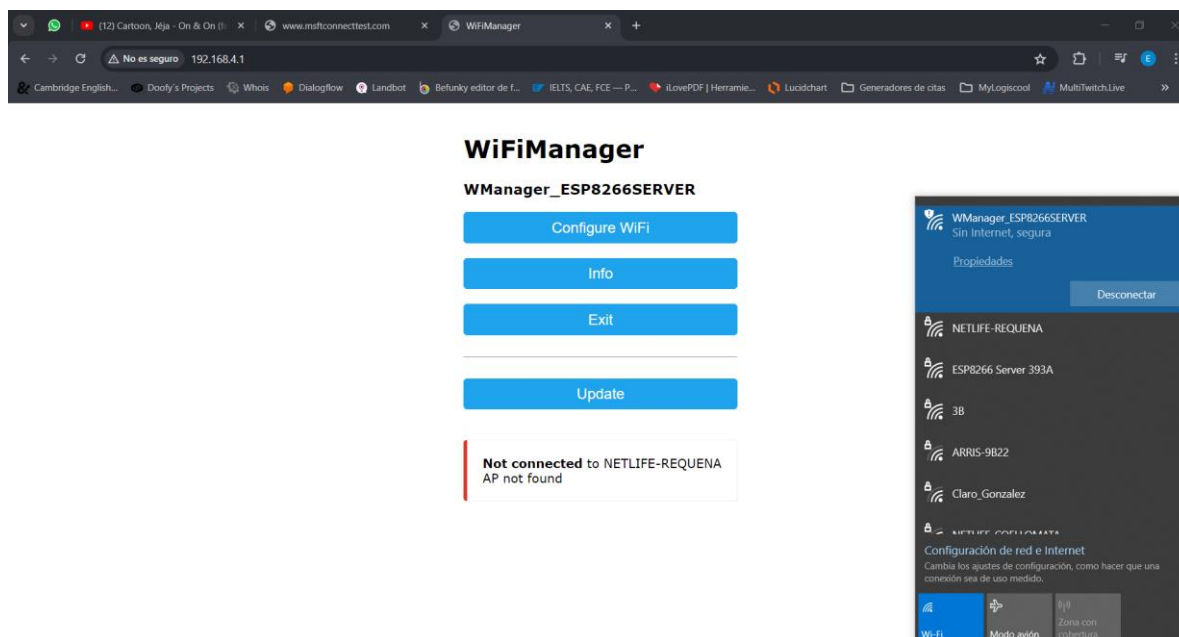
    //Eliminamos el temporizador
    tick.detach();
    //Apagar el LED
    digitalWrite(Led_interno, HIGH);
}

```

Línea de código encargada de resetear las credenciales Wifi

Figura 58. Reseteo de credenciales de la librería Wifi Manager.
Fuente: El Autor.

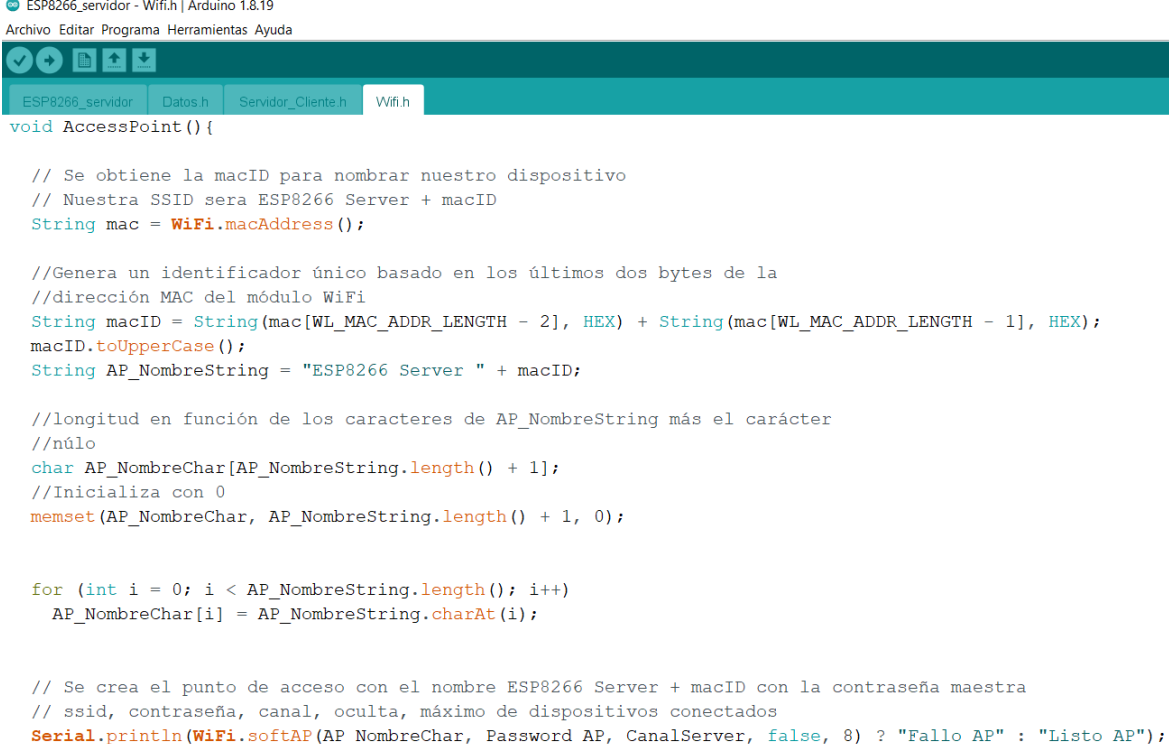
En la mayoría de los casos, esta línea de código no será necesaria, ya que se busca que el servidor pueda reconectarse automáticamente en caso de desconexión. La primera ejecución del servidor permite acceder a la página web desde cualquier dispositivo conectado a la misma red, donde el servidor se levanta para configurar una red Wifi disponible dentro de su rango, como se muestra en la figura correspondiente.



**Figura 59. Página de inicio de Wifi Manager.
Fuente: El Autor.**

Para acceder a esta página, es necesario conectarse a la red generada por Wifi Manager y luego ingresar la IP 192.168.4.1 en un navegador.

Una vez configurado el servidor como estación y Access Point (AP), este comienza a establecer una red configurable propia. Esta red está diseñada para permitir la comunicación exclusiva entre los dispositivos que enviarán la información recibida de los sensores, como se muestra en la figura.



```

ESP8266_servidor - Wifi.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_servidor Datos.h Servidor_Cliente.h Wifi.h
void AccessPoint() {

    // Se obtiene la macID para nombrar nuestro dispositivo
    // Nuestra SSID sera ESP8266 Server + macID
    String mac = WiFi.macAddress();

    //Genera un identificador único basado en los últimos dos bytes de la
    //dirección MAC del módulo WiFi
    String macID = String(mac[WLAN_MAC_ADDR_LENGTH - 2], HEX) + String(mac[WLAN_MAC_ADDR_LENGTH - 1], HEX);
    macID.toUpperCase();
    String AP_NombreString = "ESP8266 Server " + macID;

    //longitud en función de los caracteres de AP_NombreString más el carácter
    //nulo
    char AP_NombreChar[AP_NombreString.length() + 1];
    //Inicializa con 0
    memset(AP_NombreChar, AP_NombreString.length() + 1, 0);

    for (int i = 0; i < AP_NombreString.length(); i++)
        AP_NombreChar[i] = AP_NombreString.charAt(i);

    // Se crea el punto de acceso con el nombre ESP8266 Server + macID con la contraseña maestra
    // ssid, contraseña, canal, oculta, máximo de dispositivos conectados
    Serial.println(WiFi.softAP(AP_NombreChar, Password_AP, CanalServer, false, 8) ? "Fallo AP" : "Listo AP");
}

```

Figura 60. Programación de Access Point del servidor.
Fuente: El Autor.

La red del servidor tiene un nombre específico que incluye una numeración basada en los dos últimos bytes de su dirección MAC en su SSID. Esta red está protegida por una contraseña maestra conocida solo por los módulos clientes, y cuenta con parámetros de red como el canal, la opción de ocultar la red y el número máximo de dispositivos conectados, cada uno configurado en su respectivo orden. Para más detalles, consultar el Anexo 4.

Una vez completada la programación del Access Point (AP) y del Wifi Manager, es crucial establecer la conexión del servidor a una red local, ya sea doméstica o corporativa. Se ha creado una función específica para verificar esta conectividad y, en base al resultado obtenido, el servidor actuará según sea necesario, como se muestra en la figura correspondiente.



```

ESP8266_servidor - Wifi.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_servidor Datos.h Servidor_Cliente.h Wifi.h
//=====
//                               Código para estación
//=====

void ArranqueWifi() {

  Serial.begin(115200);
  delay(2000);
  WiFi.mode(WIFI_AP_STA);

  Serial.print("Conectando a Wifi...");
  if (WiFi.status() == WL_CONNECTED) {

    Serial << "\n" << "Conectado" << "\n";
    Serial << "SSID: " << WiFi.SSID() << "\n" << "IP: " << WiFi.localIP() << "\n";

    AccessPoint();

  }
  else{
    Serial.println("Iniciando WiFi manager..");
    WifiManager();
  }
}

```

Figura 61. Función de conexión Wifi inicial.
Fuente: El Autor.

Además, se ha desarrollado una función con programación recursiva para permitir que el servidor se reconecte automáticamente en caso de desconexión, o para utilizar la función Wifi Manager y configurar las credenciales de otra red, según se muestra en la figura correspondiente.



```

ESP8266_servidor - Wifi.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_servidor Datos.h Servidor_Cliente.h Wifi.h

//Función para refrescar el Wifi en caso de desconexión
void ActualizarWifi() {

  while(WiFi.status() != WL_CONNECTED) {
    if(WiFi.status() != WL_CONNECTED && cont == 5){
      cont = 0;
      break;
    }
    else{
      cont++;
      Serial.print(".");
      Serial.print(cont);
      delay(2000);
    }
  }

  WifiManager();

  Serial << "\n" << "Conectado" << "\n";
  Serial << "SSID: " << WiFi.SSID() << "\n" << "IP: " << WiFi.localIP() << "\n";
}

```

Figura 62. Reconexión del Wifi en caso de desconexión.
Fuente: El Autor.

Finalmente, en relación con las redes del servidor, se configuró un DNS local asignándole servicios específicos como HTTP y TCP en el puerto 80. Esto permite un acceso más conveniente al servidor para todos los usuarios al ingresar la URL ESPSERVER.LOCAL en el navegador de cualquier dispositivo conectado a la misma red donde se encuentra el servidor, como se muestra en la figura correspondiente.



```
ESP8266_servidor - Wifi.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_servidor Datos.h Servidor_Cliente.h Wifi.h

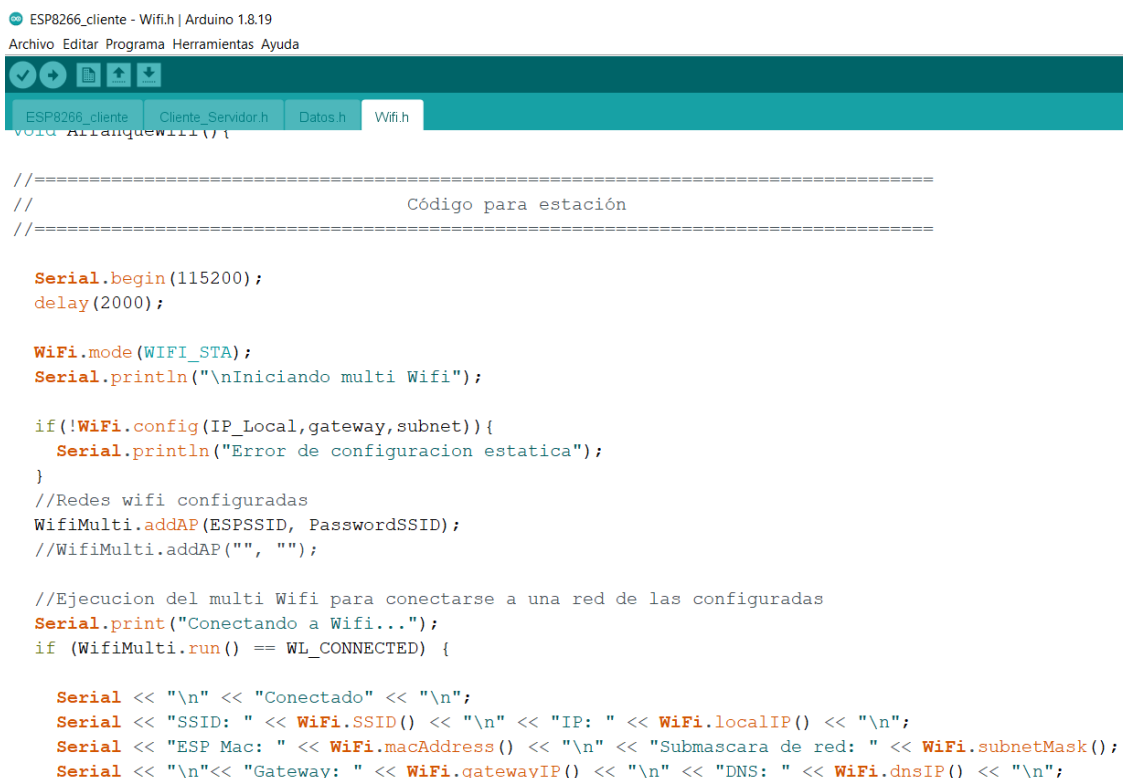
void mDNS () {

    //Configuración y uso de la librería mDNS para asignar nombre
    //para evitar usar la IP en el navegador
    if (!MDNS.begin("EspServer")) {
        Serial.println("Error configurando mDNS!");
        while (1) {
            delay(1000);
        }
    }
    Serial.println("mDNS configurado");

    //Servicios asignados al mDNS
    MDNS.addService("http", "tcp", 80);
}
```

Figura 63. Función para el DNS local del servidor.
Fuente: El Autor.

Para los clientes, se implementa la funcionalidad de conexión multi Wifi, lo cual les permite conectarse automáticamente al servidor ingresando las credenciales necesarias. Esto evita la necesidad de ingresar manualmente las credenciales de red. Además, se utiliza la librería Wifi para definir una IP y puerto específicos para cada estación, junto con otros datos necesarios como la IP particular del servidor. Este enfoque ayuda a evitar confusiones dentro de la red, como se detalla en la figura correspondiente.



```

ESP8266_cliente - Wifi.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_cliente Cliente_Servidor.h Datos.h Wifi.h
void ActualizarWifi() {

//=====
//                               Código para estación
//=====

Serial.begin(115200);
delay(2000);

WiFi.mode(WIFI_STA);
Serial.println("\nIniciando multi Wifi");

if(!WiFi.config(IP_Local,gateway,subnet)){
  Serial.println("Error de configuracion estatica");
}
//Redes wifi configuradas
WifiMulti.addAP(ESPSSID, PasswordSSID);
//WifiMulti.addAP("", "");

//Ejecucion del multi Wifi para conectarse a una red de las configuradas
Serial.print("Conectando a Wifi...");
if (WifiMulti.run() == WL_CONNECTED) {

  Serial << "\n" << "Conectado" << "\n";
  Serial << "SSID: " << WiFi.SSID() << "\n" << "IP: " << WiFi.localIP() << "\n";
  Serial << "ESP Mac: " << WiFi.macAddress() << "\n" << "Submascara de red: " << WiFi.subnetMask();
  Serial << "\n" << "Gateway: " << WiFi.gatewayIP() << "\n" << "DNS: " << WiFi.dnsIP() << "\n";
}
}

```

Figura 64. Código para conexión con el servidor de forma estática.
Fuente: El Autor.

De manera similar al servidor, el cliente utiliza un código de reconexión diseñado para restablecer automáticamente la comunicación con el servidor en caso de pérdida de conexión, tal como se muestra en la figura correspondiente.



```

ESP8266_cliente - Wifi.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_cliente Cliente_Servidor.h Datos.h Wifi.h

//Función para refrescar el Wifi en caso de desconexión
void ActualizarWifi() {

  while(WifiMulti.run() != WL_CONNECTED) {
    Serial.print(".");
    delay(5000);
  }
  yield();

  Serial << "\n" << "Conectado" << "\n";
  Serial << "SSID: " << WiFi.SSID() << "\n" << "IP: " << WiFi.localIP() << "\n";
  Serial << "ESP Mac: " << WiFi.macAddress() << "\n" << "Submascara de red: " << WiFi.subnetMask();
  Serial << "\n" << "Gateway: " << WiFi.gatewayIP() << "\n" << "DNS: " << WiFi.dnsIP() << "\n";
  Serial << "De: " << Udp_receiving.remoteIP() << ", puerto: " << Udp_receiving.remotePort() << "\n";
}
}

```

Figura 65. Código de reconexión del cliente.
Fuente: El Autor.

A diferencia del servidor, el cliente solo necesita establecer conexión con el servidor una vez y, al tener sus credenciales almacenadas, permanece en espera hasta que el servidor esté disponible para reanudar la comunicación.

La comunicación entre el cliente y el servidor se realiza mediante paquetes UDP, los cuales contienen la información necesaria identificada por estación mediante variables y programación que generan arreglos definidos por objetos, como se muestra en la figura correspondiente.



```

ESP8266_cliente - Datos.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda

ESP8266_cliente Cliente_Servidor.h Datos.h Wifi.h

//Se define una estructura para enviar la distancia al servidor por paquetes UDP
typedef struct udp{
    int id;
    float distancia;
    char sended[32];
} Paquete_UDP;

Paquete_UDP paquete = {1,0,"Enviado"};

typedef struct udpdos{
    int numEstaciones;
    int distanciaCambio;
    char sended[32];
} Paquete_UDP2;
Paquete_UDP2 paquete2;

String macID; //Variable para almacenar la macID
int NumStation = 1;

```

Figura 66. Arreglos para definir la información que llevarán los paquetes UDP.
Fuente: El Autor.

Luego, estos paquetes se utilizan dentro de los métodos reservados de la librería para empaquetar la información según el arreglo previamente definido con los valores de las variables correspondientes, y se envían al destino con la IP y puertos específicos para el servidor, tal como se ilustra en la figura.



```
ESP8266_cliente - Cliente_Servidor.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_cliente Cliente_Servidor.h Datos.h Wifi.h

void UDP_Enviado_Servidor() {

    paquete.id = NumStation;
    paquete.distancia = (int)DISTANCIA;

    Udp_sending.beginPacket(IP_Server, ServerPort);
    Udp_sending.write((byte *)&paquete, sizeof(Paquete_UDP));
    Udp_sending.endPacket();

    Serial << "Distancia: " << paquete.distancia << " cm" << "\n";
}
```

Figura 67. Función de envío de paquetes UDP al servidor.
Fuente: El Autor.

Además, se emplea un componente de la librería Ticker para invocar periódicamente esta función de envío de paquetes al servidor, con un intervalo definido de 1 segundo, como se detalla en la figura correspondiente.

```

ESP8266_cliente  Cliente_Servidor.h  Datos.h  Wifi.h
//Librerias internas del proyecto
#include "Datos.h"
#include "Wifi.h"
#include "Cliente_Servidor.h"

void setup() {

  ArranqueWifi();

  SetPins();

  Udp_sending.begin(ServerPort);
  Udp_receiving.begin(ClientPort);

  receiving.attach(1, UDP_Recibido_Servidor);
  sending.attach(1, UDP_Enviado_Servidor);
}

void loop() {

  //Condición que llama a la función de reconectar al wifi
  if (WifiMulti.run() != WL_CONNECTED){
    ActualizarWifi();
  }
}

```

Figura 68. Línea que llama a la función en un tiempo definido constantemente.
Fuente: El Autor.

La necesidad de emplear la librería Ticker surge de los problemas derivados del uso de la función delay en el proyecto. Esta función causa que el microcontrolador suspenda sus operaciones durante un período determinado por el usuario, lo cual puede ocasionar conflictos recurrentes con la página web y, en casos extremos, reinicios continuos del microcontrolador. Por esta razón, la librería Ticker se utiliza para esperar intervalos de tiempo sin afectar la ejecución de otras partes del código.

Una vez que los paquetes alcanzan su destino, estos son leídos únicamente si en el servidor se han definido arreglos que contengan los mismos parámetros que los paquetes enviados por el cliente. Con los arreglos adecuadamente configurados, se implementan funciones que utilizan las operaciones reservadas por la librería UDP para interpretar el contenido de los paquetes, como se detalla en la figura correspondiente.

```

ESP8266_servidor  Datos.h  Servidor_Cliente.h  Wifi.h
void LectorUDP(){
  //Si hay información disponible leerá el paquete
  int tamanoPaquete = Udp_receiving.parsePacket();

  if (tamanoPaquete){
    Serial << "\nPaquete de tamaño " << tamanoPaquete << " recibido\n";
    Serial << "De: " << Udp_receiving.remoteIP() << ", puerto: " << Udp_receiving.remotePort() << "\n";

    int len = Udp_receiving.read((byte *)&paquete, sizeof(Paquete_UDP));
    if (len > 0) {
      char* incomingPacket = new char[sizeof(Paquete_UDP)];
      incomingPacket[len] = 0;
    }

    Serial.println("Contenido: ");
    Serial << "ID: " << paquete.id << "\n" << "Estado: " << paquete.sended << "\n";

    if(paquete.id == 1){
      dist_sen1 = paquete.distancia;
      Serial << "Sensor 1: " << dist_sen1 << " cm" << "\n";

    }else if(paquete.id == 2){
      dist_sen2 = paquete.distancia;
      Serial << "Sensor 2: " << dist_sen2 << " cm" << "\n";
    }
  }
}

```

Figura 69. Función para leer los paquetes UDP entrantes.
Fuente: El Autor.

Al recibir un paquete, el servidor procede inicialmente a desempaquetar la información contenida y evaluar su contenido. Luego, se aplican condiciones para identificar el cliente emisor del paquete, permitiendo así la publicación independiente de la información recibida por cada módulo.

Este proceso se repite de manera constante utilizando la misma red Wifi previamente configurada, utilizando los IP y puertos establecidos por la librería Wifi para facilitar la transmisión de datos entre dispositivos. Es importante destacar que estos paquetes de datos permanecen ocultos al usuario final que accede al portal web, mostrándose únicamente la información formateada dentro de la página.

6.4. Desarrollo de la página web basado en HTML, CSS y JavaScript

Para el montaje del proyecto en el Instituto Superior Tecnológico Sudamericano, se deben considerar diversas secciones para su correcto funcionamiento. El primer paso consiste en determinar el esquema lógico de la red. Esto se hace con el fin de identificar el medio ideal y la disponibilidad para realizar las conexiones del sistema con la red corporativa.

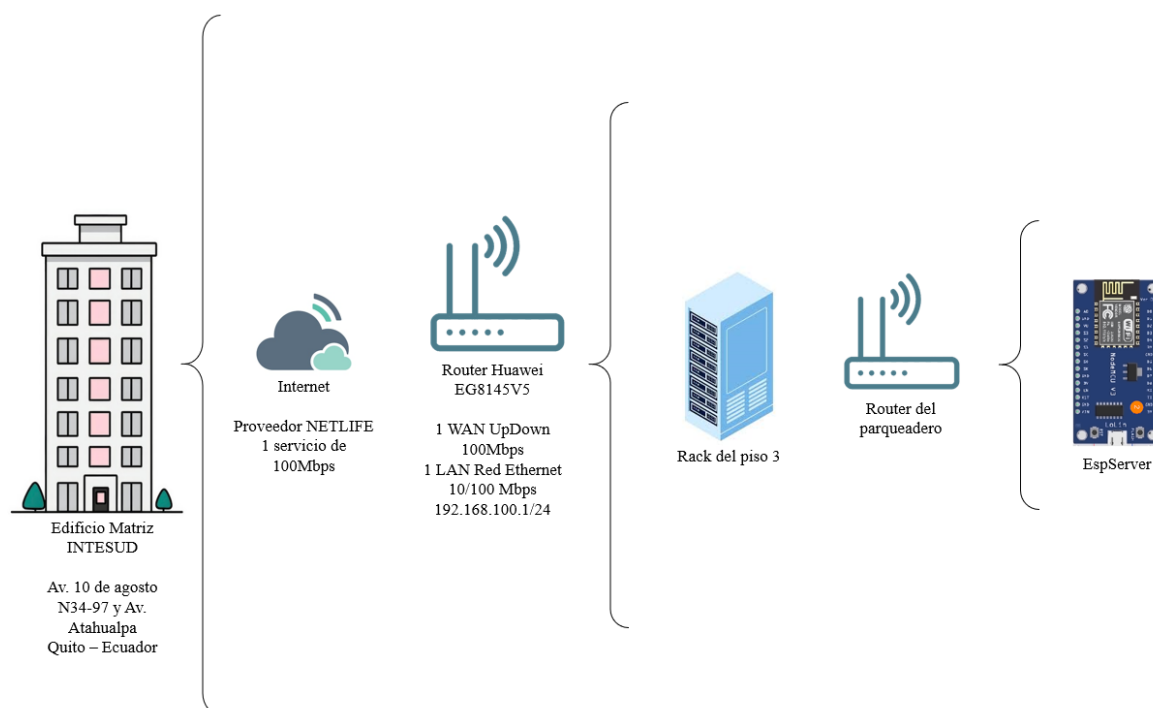


Figura 70. Esquema de la red corporativa del INTESUD al servidor del sistema de parqueadero.
Fuente: El Autor.

Dado que el servidor será el núcleo de todos los procesos, el siguiente paso es su configuración. Será necesario especificar las redes Wifi que estarán disponibles en el servidor, tal como se muestra en la figura anterior. La instalación de las redes variará según cada ubicación, dependiendo de los servicios y la disponibilidad que ofrezca cada sitio.

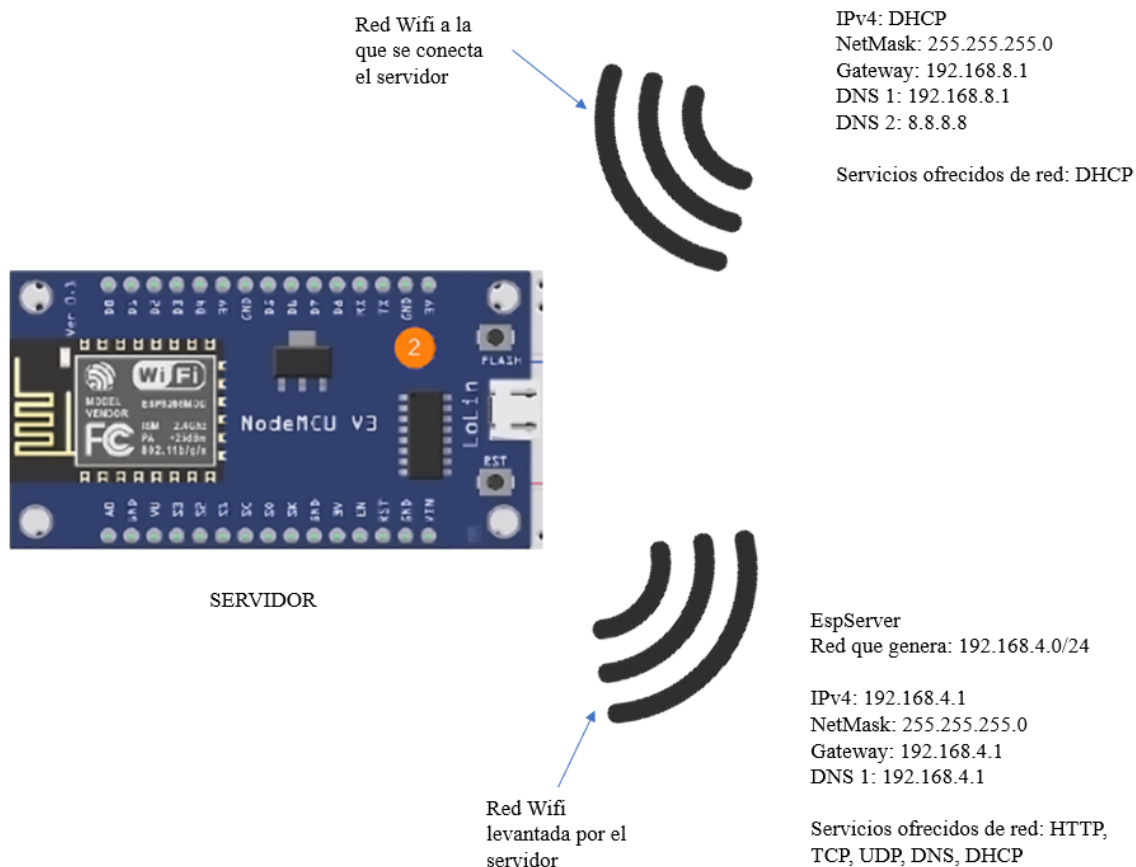


Figura 71. Esquema de redes Wifi que intervendrán con el servidor.
Fuente: El Autor.

El servidor se conecta a una red Wifi que puede variar, pero la red que el servidor crea es siempre la misma.

Una vez que se ha determinado la funcionalidad y la aplicación de la red de la empresa para integrar el sistema, es necesario considerar la ubicación óptima de los módulos según cada puesto dentro del establecimiento, así como la posición estratégica del servidor para cubrirlos adecuadamente.

Una vez que el servidor establece la conexión con los módulos, crea una página web que puede ser accedida a través de la red a la que está conectado el servidor. Esta información puede ser compartida con el resto de los usuarios dentro del edificio, como se ilustra en la figura.

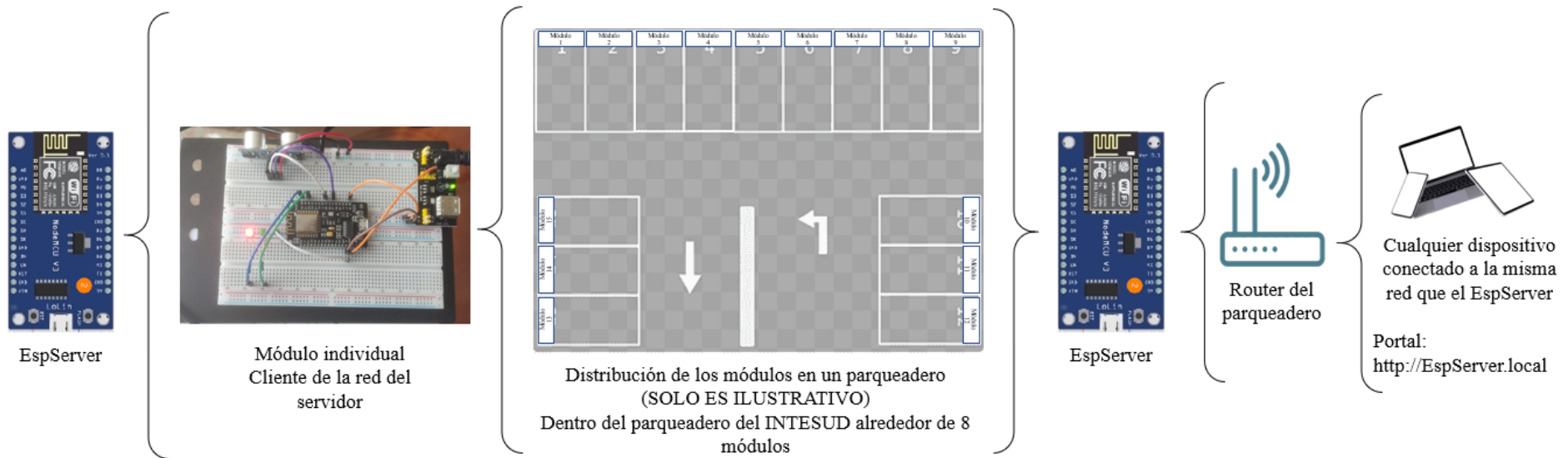


Figura 72. Comunicación entre los equipos del sistema con el usuario final.
Fuente: El Autor.

El nombre DNS asignado al servidor funciona únicamente de forma local. Para que el sistema funcione de manera óptima en Internet, es necesario utilizar un servicio que permita crear un túnel desde el sitio web del servidor, permitiendo así que sea accesible desde cualquier dispositivo conectado a Internet.

Para el cuarto objetivo del proyecto, se optó por implementar programación basada en HTML, CSS y JavaScript, considerados fundamentales para iniciarse en el desarrollo web. Esta elección se sustenta en la accesibilidad de recursos educativos y documentación disponible en línea, facilitando el cumplimiento de los objetivos propuestos.

HTML se utilizó como el esquema inicial de la página web. Esta tecnología proporcionó la estructura básica y los elementos esenciales del contenido, incluyendo etiquetas y títulos, fundamentales para la organización y presentación de la información dentro del proyecto.

```

Arduino Proyecto > <> FisgonParking.html > html > body
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>ESP8266 Server Checking</title>
7  >
146 </style>
147 </head>
148 <body>
149     <div class="contenedor">
150         <h1 class="titulo">Fisgon Parking Monitoring</h1>
151         <div class="espacio_contenedor">
152             <h4 id="dist1" class="display">Parqueadero 1: <span id="valor_distancia1" class="display">---</span> Cm</h4>
153             <output class="rango" id="rango_valP1">40</output><br>
154             <input id="rangoP1" type="range" value="40" min="2" max="450" oninput="rango_valP1.value=value">
155
156             <h4 id="dist2" class="display">Parqueadero 2: <span id="valor_distancia2" class="display">---</span> Cm</h4>
157             <output class="rango" id="rango_valP2">40</output><br>
158             <input id="rangoP2" type="range" value="40" min="2" max="450" oninput="rango_valP2.value=value">
159
160             <div>
161                 <button class="btn" onclick="Pagina2()">
162                     <span></span><span></span><span></span><span></span><span></span>
163                 </button>
164             </div>
165         </div>
166     </div>
167 </body>
168
169
170
171 >
281 </script>
282
283 </body>
284 </html>

```

Figura 73. Elementos de la página web en HTML.
Fuente: El Autor.

El contenido dentro de las etiquetas <body> constituye la información visible en la página inicial del proyecto. Esta estructura está organizada mediante diversas etiquetas <div>, cada una identificada para nombrar secciones específicas. Es crucial destacar los identificadores asignados a estas etiquetas, usualmente definidos como ID y clases, respectivamente en orden de prioridad. Estos identificadores facilitan la programación en CSS y JavaScript al permitir la identificación precisa de los elementos que se desean configurar y estilizar.

```

Arduino Proyecto > <> FisgonParking.html > html > body
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <title>ESP8266 Server Checking</title>
7 >
146 </style>
147 </head>
148 <body>
149 <div class="contenedor">
150 <h1 class="titulo">Fisgon Parking Monitoring</h1>
151 <div class="espacio contenedor">
152 <h4 id="dist1" class="display">Parqueadero 1: <span id="valor_distancia1" class="display">---</span> Cm</h4>
153 <output class="rango" id="rango_valP1">40</output><br>
154 <input id="rangoP1" type="range" value="40" min="2" max="450" oninput="rango_valP1.value=value">
155
156 <h4 id="dist2" class="display">Parqueadero 2: <span id="valor_distancia2" class="display">---</span> Cm</h4>
157 <output class="rango" id="rango_valP2">40</output><br>
158 <input id="rangoP2" type="range" value="40" min="2" max="450" oninput="rango_valP2.value=value">
159
160 <div>
161 <button class="btn" onclick="Pagina2()">
162 <span></span><span></span><span></span><span></span><span></span></span></div>
163
164 </div>
165 </div>
166 </div>
167
168
169
170
171 >
281 </script>
282
283 </body>
284 </html>

```

Figura 74. Identificadores dentro de las etiquetas en HTML.
Fuente: El Autor.

El CSS se aplicó integralmente para diseñar cada página web, abordando desde la selección de colores hasta el posicionamiento de elementos y la implementación de animaciones. Este proceso permitió ajustar todos los detalles visuales de los componentes previamente creados en HTML, con el objetivo de generar páginas web atractivas y dinámicas.

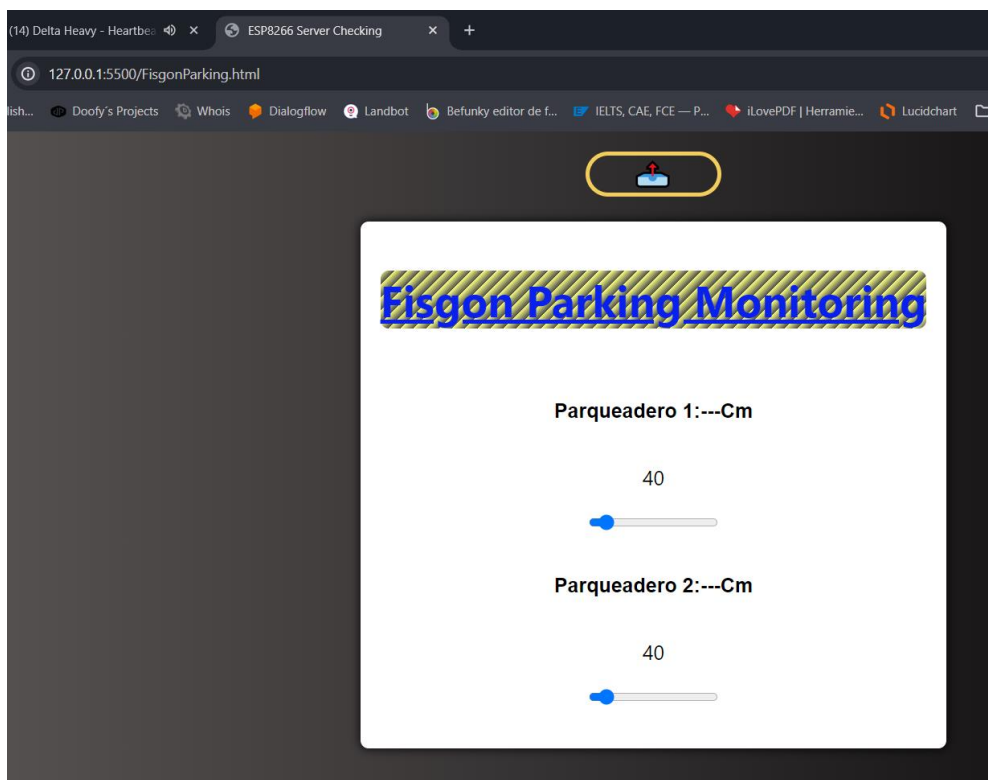


Figura 75. Página principal del administrador.
Fuente El Autor.

La primera forma de integrar CSS en un documento HTML es mediante el uso de la etiqueta `<style>`, donde se colocan las reglas de estilo directamente dentro del archivo HTML. Esto permite al editor de texto o IDE reconocer y aplicar las reglas de estilo al documento HTML de manera interna.

La segunda forma consiste en enlazar un archivo externo de hojas de estilo CSS al documento HTML utilizando la etiqueta `<link>`. Este método permite mantener el código CSS separado del HTML, facilitando la gestión y la reutilización de estilos en múltiples páginas HTML.

```

Arduino Proyecto > <> FisgonParkingMain.html > html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Fisgon Parking</title>
7      <link rel="stylesheet" type="text/css" href="styles.css">
8
9      <style>
10         @import url("style.css");
11     </style>
12 </head>

```

Figura 76. Código para anexar una hoja de estilos a una página HTML.
Fuente: El Autor.

La última modalidad consiste en una combinación de los métodos mencionados previamente, donde se incorpora una referencia a una hoja de estilos dentro de las etiquetas `<style>` de un documento HTML, como se ilustra en la figura.

```

Arduino Proyecto > <> FisgonParkingMain.html > html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Fisgon Parking</title>
7      <link rel="stylesheet" type="text/css" href="styles.css">
8
9      <style>
10         @import url("style.css");
11     </style>
12 </head>

```

Figura 77. Método para anexar una hoja de estilos a un documento HTML mediante @import
Fuente: El Autor

El método `@import` para llamar hojas de estilos puede variar en su sintaxis. Es crucial destacar que, sin importar qué método se utilice entre los tres mencionados anteriormente, las hojas de estilos siempre deben ser invocadas en la sección `<head>` de cualquier página web.

Esto se debe a que CSS determina la apariencia inicial del portal web, y al cargar el código HTML, este se formatea según las reglas establecidas en la hoja de estilos.

Para este proyecto en particular, se optó por el primer método debido a que todo el código se cargará posteriormente en el servidor, lo cual impide enlazar múltiples archivos. Por lo tanto, se incorpora directamente el código al documento HTML como una solución rápida.

La programación dentro de CSS puede ser compleja y extensa, ya que dicta la presentación visual de la página web. En este proyecto, se organizaron diferentes grupos según los identificadores establecidos en HTML para aplicar un formato inicial a todos los elementos, como se muestra en la figura correspondiente.

```

3 <head>
7 <style>
8   html{
9     font-family:'Segoe UI', tahoma, Geneva, Verdana, sans-serif;
10    background: ■ #000000;
11    background: linear-gradient(to right, ■ #5b5757, ■ #000000);
12    display: inline-block;
13    text-align: center;
14  }
15
16  body{
17    margin: 0 auto;
18    max-width: 600px;
19    padding-bottom: 25px;
20    display: flex;
21    align-items: center;
22    justify-content: center;
23    height: 100vh;
24  }
25
26  h1{
27    text-decoration: underline;
28    font-size: 2.7rem;
29    color: ■ #0a21ee;
30    border-radius: 8px;
31    background: repeating-linear-gradient(
32      -45deg,
33      ■ rgba(0,0,0,.7) 1px,
34      ■ rgba(246, 255, 0, 0.5) 10px
35    );
36  }
37  h4{
38    font-family: Arial, sans-serif;
39    font-size: 20px;
40    margin: 0;
41    padding: 0;
42    display: flex;
43    align-items: center;
44    justify-content: center;
45
46    height: 15vh;
47    color: ■ #000000;
48  }

```

Figura 78. Código base de formateo de la hoja de estilos.
Fuente: El Autor.

Luego, se agregaron detalles adicionales a ciertos elementos dentro de la página, como se observa en el código mostrado en la figura.

```
7 <style>
78 .btn{
79     position: relative;
80     top: -36em;
81     background-color: transparent;
82     --c: #F3CE5E;
83     color: var(--c);
84     font-size: 15px;
85     border: 0.3em solid var(--c);
86     border-radius: 1.9em;
87     width: 9em;
88     height: 3em;
89     text-transform: uppercase;
90     font-weight: bold;
91     font-family: sans-serif;
92     letter-spacing: 0.1em;
93     text-align: center;
94     line-height: 1em;
95     overflow: hidden;
96     z-index: 1;
97     transition: 0.5s;
98     margin: -4.0em;
99     content: ;
100 }
101
102 .btn span{
103     position: absolute;
104     width: 25%;
105     height: 100%;
106     background-color: var(--c);
107     transform: translateY(150%);
108     border-radius: 50%;
109     left: calc((var(--n) - 1) * 25%);
110     transition: 0.5s;
111     transition-delay: calc((var(--n) - 1) * 0.1s);
112     z-index: -1;
113 }
```

Figura 79. Programación de estilo del botón dentro del portal web.
Fuente: El Autor.

Luego, se añadieron efectos de colores y movimientos para dotar de dinamismo, tal como se evidencia en el código mostrado en la figura.

```

115     .btn:hover {
116     |     --c: #FDA8CF;
117     |     color: black;
118     | }
119     .btn::after{
120     |     content: "🔍";
121     |     font-size: 30px;
122     |     transition: 0.5s;
123     | }
124     .btn:hover::after{
125     |     font-size: medium;
126     |     content: "Actualizar";
127     | }
128
129     .btn:hover span {
130     |     transform: translateY(0) scale(2);
131     | }
132
133     .btn span:nth-child(1) {
134     |     --n: 1;
135     | }
136     .btn span:nth-child(2) {
137     |     --n: 2;
138     | }
139     .btn span:nth-child(3) {
140     |     --n: 3;
141     | }
142     .btn span:nth-child(4) {
143     |     --n: 4;
144     | }
145
146 </style>

```

Figura 80. Programación de animaciones del botón.
Fuente: El Autor.

Y, por último, JavaScript. El código implementado en este lenguaje es crucial, ya que permite manejar las interacciones del lado del cliente en la página web. En esta sección se explicará cómo interactúa con los elementos de la página y más adelante se abordará su integración con el servidor.

La programación en JavaScript proporcionó la capacidad de dar sentido a los elementos creados dentro del HTML, como botones o los valores a mostrar de las distancias capturadas por los sensores, mediante la creación de diversas tareas lógicas para el correcto funcionamiento del portal web. Para añadir código en JavaScript, existen dos métodos. El primero es incluir las instrucciones dentro de las etiquetas de apertura y cierre `<script>` para referenciar este lenguaje de programación. La segunda forma es mediante un enlace similar al utilizado en CSS para llamar a un archivo externo, el cual debe estar ubicado en el mismo directorio que el archivo HTML, como se muestra en la figura.

```
Arduino Proyecto > <> FisgonParkingMain.html > html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Fisgon Parking</title>
7      <link rel="stylesheet" type="text/css" href="styles.css">
8
9      <style>
10         @import url("style.css");
11     </style>
12 </head>
13 <body>
14
15 </body>
16
17 <script src="Directorio donde esta el .js y el .html"></script>
18
19 </html>
```

Figura 81. Anexo de archivo externo de JavaScript.
Fuente: El Autor.

Para implementar adecuadamente la funcionalidad en JavaScript, se optó por el segundo método debido a sus ventajas de reutilización en múltiples páginas, mejor legibilidad para la depuración de errores y el beneficio de la memoria caché para una carga más rápida de los sitios web. En el contexto de este proyecto, se desarrolló un código específico que se integró directamente en la página web. Este enfoque permitió actualizar de manera eficiente solo los valores de las distancias capturadas, sin necesidad de refrescar toda la página. En la etapa inicial, se definieron las variables que capturaron la información correspondiente de las etiquetas HTML, lo cual se muestra en la figura de referencia.

```
171     <script>
172         let ejecutar = false;
173         const tiempoAgotado = 5000;
174         const dist1 = document.getElementById('valor_distancia1');
175         const dist2 = document.getElementById('valor_distancia2');
176         const elementoRango1 = document.getElementById('rangoP1');
177         const elementoRango2 = document.getElementById('rangoP2');
178
179         elementoRango1.addEventListener('click', cambiarDistancia1);
180         elementoRango2.addEventListener('click', cambiarDistancia2);
```

Figura 82. Variables usadas en JavaScript.
Fuente: El Autor.

La función principal de JavaScript en la página web es dotar de funcionalidad a los botones, permitiendo acciones como cambios de página o actualización de datos específicos que afectan individualmente a los módulos. Además, mediante solicitudes al servidor y actualización dinámica de la información, se implementa un método que evita la necesidad de recargar constantemente toda la página, lo cual reduce los tiempos de espera y mejora la experiencia del usuario.

6.5. Aplicación de las distintas partes del proyecto dentro del ESP8266

Para este punto, se han desarrollado las páginas web necesarias y se ha establecido un esquema lógico de redes para facilitar la comunicación entre los módulos. La siguiente fase consistió en establecer un medio de comunicación bidireccional entre ambas partes, tal como se muestra en el diagrama que continúa el proceso inicialmente mencionado. Este diagrama ilustra cómo el servidor procesa la información enviada por cada módulo para luego mostrarla en el sitio web.

Para llevar a cabo las acciones delineadas en el diagrama, se han integrado las librerías de DNS para el servidor y Web Server, junto con la programación en JavaScript. El Domain Name System (DNS) ha sido implementado directamente en el servidor para configurar un nombre que facilite la resolución de la dirección IP correspondiente, como se detalla en la figura adjunta.

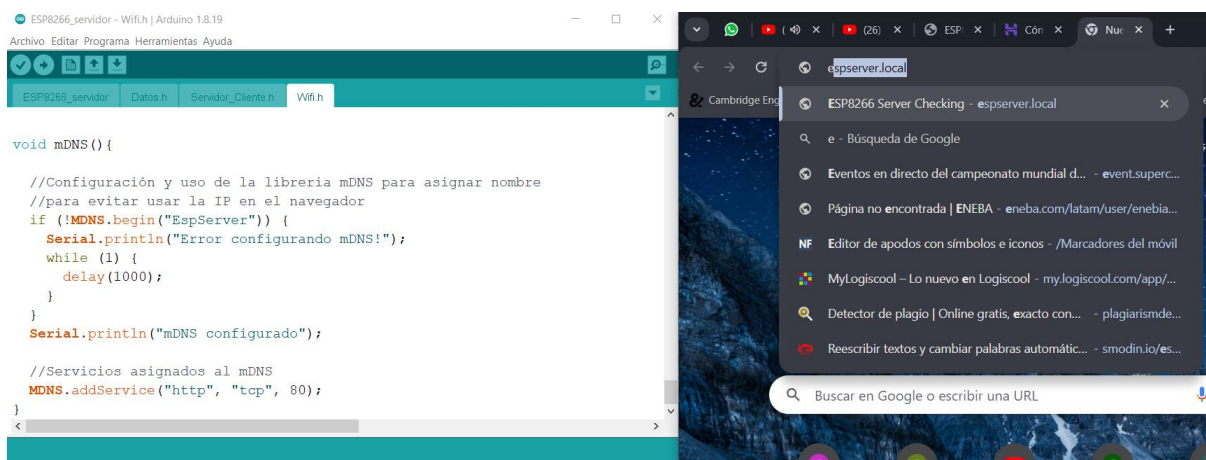
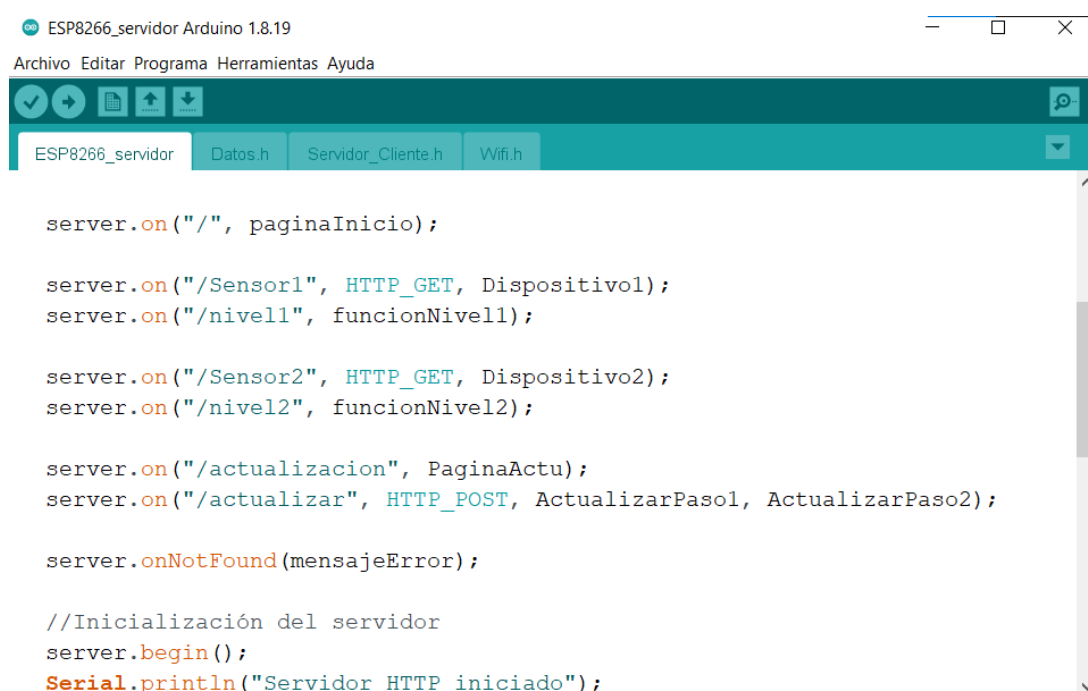


Figura 83. Asignación de DNS en el servidor.
Fuente: El Autor.

Al tratarse de un servicio local de DNS, la resolución de nombres de dominio está limitada únicamente al ámbito interno de la red a la cual está conectado el servidor. Esto facilita el acceso para los usuarios finales a las páginas web que contiene. Por otro lado, la librería Web Server proporcionó un manejo completo de todas las solicitudes dirigidas a la página web. Con esta

herramienta, fue posible integrar todo el código de las diversas páginas web junto con JavaScript para gestionar eficazmente las solicitudes de diferentes clientes que acceden al nombre de dominio establecido para visualizar su contenido.

Inicialmente, se configuraron respuestas para todas las solicitudes realizadas por los clientes, como la carga completa de la página. Para comprender este proceso, se introdujeron conceptos como HTTP GET/POST, los cuales JavaScript utilizó para realizar directamente a la dirección IP del servidor la consulta solicitada, tal como se ilustra en la figura.



```
ESP8266_servidor Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_servidor Datos.h Servidor_Cliente.h Wifi.h

server.on("/", paginaInicio);

server.on("/Sensor1", HTTP_GET, Dispositivo1);
server.on("/nivel1", funcionNivel1);

server.on("/Sensor2", HTTP_GET, Dispositivo2);
server.on("/nivel2", funcionNivel2);

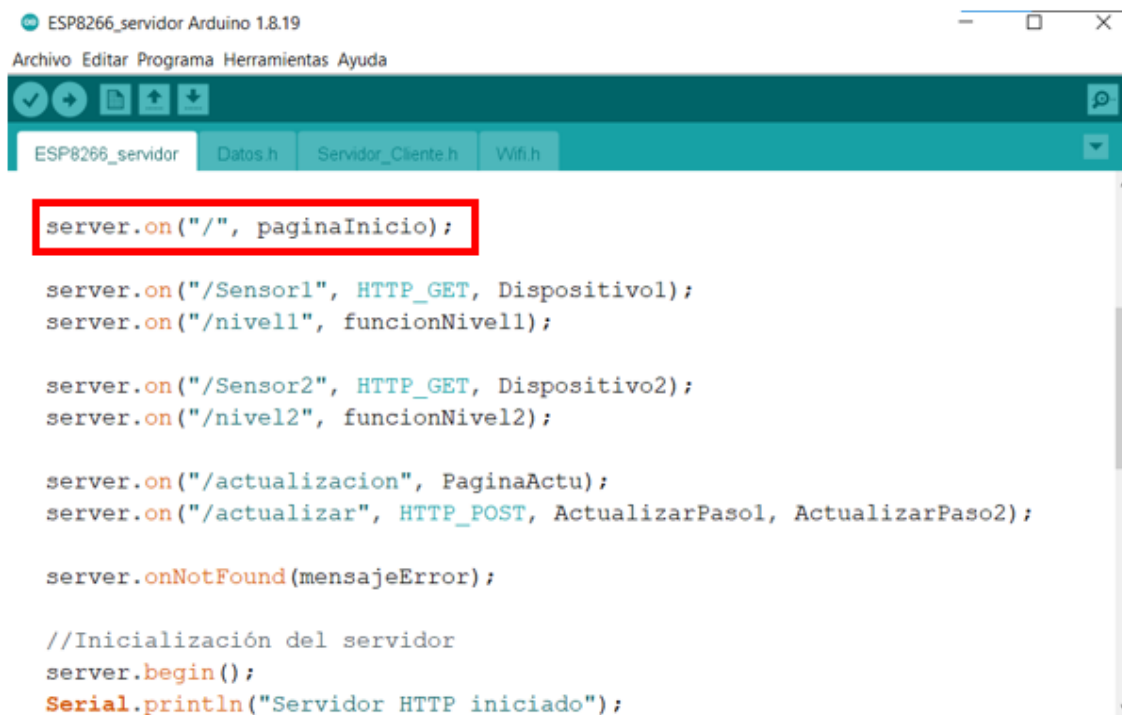
server.on("/actualizacion", PaginaActu);
server.on("/actualizar", HTTP_POST, ActualizarPaso1, ActualizarPaso2);

server.onNotFound(mensajeError);

//Inicialización del servidor
server.begin();
Serial.println("Servidor HTTP iniciado");
```

Figura 84. Acciones del servidor ante las solicitudes de HTTP GET/POST.
Fuente: El Autor.

En el ejemplo presentado, se muestra cómo se configuran acciones con el objeto creado a partir de la librería Web Server. Este objeto se encarga de responder a las solicitudes realizadas desde un navegador mediante JavaScript para obtener la información requerida a través de una URL específica. Por ejemplo, al encender el servidor y ejecutar el código de la página web, la primera acción consistirá en realizar una consulta a la dirección IP seguida de una barra inclinada (/), indicando la operación a ejecutar, como se ilustra en la figura.



```

ESP8266_servidor Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_servidor Datos.h Servidor_Cliente.h Wifi.h
server.on("/", paginaInicio);

server.on("/Sensor1", HTTP_GET, Dispositivo1);
server.on("/nivell", funcionNivell);

server.on("/Sensor2", HTTP_GET, Dispositivo2);
server.on("/nivel2", funcionNivel2);

server.on("/actualizacion", PaginaActu);
server.on("/actualizar", HTTP_POST, ActualizarPasol, ActualizarPaso2);

server.onNotFound(mensajeError);

//Inicialización del servidor
server.begin();
Serial.println("Servidor HTTP iniciado");

```

Figura 85. Línea de lanzamiento de la página web del servidor.
Fuente: El Autor.

En este contexto, al iniciar, se invoca la función “Pagina Inicio”, la cual incluye la llamada a nuestro sitio web, como se representa en la figura correspondiente.



```

ESP8266_servidor - Servidor_Cliente.h | Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_servidor Datos.h Servidor_Cliente.h Wifi.h
/*****
void paginaInicio() {
  server.setHeader("Access-Control-Allow-Origin", "*");
  server.send(200, "text/html", Pagina);
}

```

Figura 86. Función que solicita la página web de inicio.
Fuente: El Autor.

Dentro de la función mencionada, se utilizan sentencias específicas diseñadas para permitir que cualquier dispositivo acceda a la información contenida en ella. La primera línea se encarga de otorgar permisos de acceso, asegurando que no haya problemas de comunicación al intentar

acceder desde diferentes dispositivos. La segunda línea envía un estado de OK al sitio web y carga el contenido HTML almacenado en la función a la que apunta, en este caso, 'Pagina'.

Inicializar esta función es crucial para que una página de inicio se cargue automáticamente al acceder a la IP del servidor. En el caso de agregar una nueva página al servidor, es importante formatearla adecuadamente para evitar conflictos con el lenguaje de programación. A continuación, se muestra el código utilizado para la página web.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>ESP8266 Server Checking</title>
  <style>
    html{
      font-family:'Segoe UI', tahoma, Geneva, Verdana, sans-serif;
      background: #000000;
      background: linear-gradient(to right, #5b5757, #000000);
      display: inline-block;
      text-align: center;
    }

    body{
      margin: 0 auto;
      max-width: 600px;
      padding-bottom: 25px;
      display: flex;
      align-items: center;
      justify-content: center;
      height: 100vh;
    }

    h1{
      text-decoration: underline;
      font-size: 2.7rem;
      color: #0a21ee;
      border-radius: 8px;
      background: repeating-linear-gradient(
        -45deg,
        rgba(0,0,0,.7) 1px,
        rgba(246, 255, 0, 0.5) 10px
      );
    }
    h4{
```

```
    font-family: Arial, sans-serif;
    font-size: 20px;
    margin: 0;
    padding: 0;
    display: flex;
    align-items: center;
    justify-content: center;
    height: 15vh;
    color: #000000;
}

.contenedor{
    background-color: #fff;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 1.5);
}

.espacio_contenedor{
    position: relative;
    margin-top: 20px;
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    max-width: 600px;
}

#valor_distancia {
    margin-left: 10px;
    margin-right: 10px;
    height: 33px;
    font-weight: bold;
    color: #007bff;
}

.rango{
    font-size: 20px;
}

.btn{
    position: relative;
    top: -36em;
    background-color: transparent;
    --c: #F3CE5E;
    color: var(--c);
    font-size: 15px;
    border: 0.3em solid var(--c);
    border-radius: 1.9em;
```

```

width: 9em;
height: 3em;
text-transform: uppercase;
font-weight: bold;
font-family: sans-serif;
letter-spacing: 0.1em;
text-align: center;
line-height: 1em;
overflow: hidden;
z-index: 1;
transition: 0.5s;
margin: -4.0em;
content: 📄;
}

.btn span{
  position: absolute;
  width: 25%;
  height: 100%;
  background-color: var(--c);
  transform: translateY(150%);
  border-radius: 50%;
  left: calc((var(--n) - 1) * 25%);
  transition: 0.5s;
  transition-delay: calc((var(--n) - 1) * 0.1s);
  z-index: -1;
}

.btn:hover {
  --c: #FDA8CF;
  color: black;
}

.btn::after{
  content: "";
  font-size: 30px;
  transition: 0.5s;
}

.btn:hover::after{
  font-size: medium;
  content: "Actualizar";
}

.btn:hover span {
  transform: translateY(0) scale(2);
}

.btn span:nth-child(1) {
  --n: 1;
}

```

```

    .btn span:nth-child(2) {
        --n: 2;
    }
    .btn span:nth-child(3) {
        --n: 3;
    }
    .btn span:nth-child(4) {
        --n: 4;
    }

</style>
</head>
<body>
    <div class="contenedor">
        <h1 class="titulo">Fisgon Parking Monitoring</h1>
        <div class="espacio_contenedor">
            <h4 id="dist1" class="display">Parqueadero 1: <span id="valor_distancia1"
class="display">---</span> Cm</h4>
            <output class="rango" id="rango_valP1">40</output><br>
            <input id="rangoP1" type="range" value="40" min="2" max="450"
oninput="rango_valP1.value=value">

            <h4 id="dist2" class="display">Parqueadero 2: <span id="valor_distancia2"
class="display">---</span> Cm</h4>
            <output class="rango" id="rango_valP2">40</output><br>
            <input id="rangoP2" type="range" value="40" min="2" max="450"
oninput="rango_valP2.value=value">

            <div>
                <button class="btn" onclick="Pagina2()">

                    <span></span><span></span><span></span><span></span>
                </button>
            </div>
        </div>
    </div>
</div>

<script>
    let ejecutar = false;
    const tiempoAgotado = 5000;
    const dist1 = document.getElementById('valor_distancia1');
    const dist2 = document.getElementById('valor_distancia2');
    const elementoRango1 = document.getElementById('rangoP1');
    const elementoRango2 = document.getElementById('rangoP2');

    elementoRango1.addEventListener('click', cambiarDistancia1);
    elementoRango2.addEventListener('click', cambiarDistancia2);

```



```

function actualizarDistancia1(processTemp){
  if(ejecutar){
    console.log("La actualización del sensor 1 ya se está ejecutando. No
se permiten llamadas concurrentes.");
    return;
  }

  ejecutar = true;

  return new Promise((resolve, reject) => {
    let xhr = new XMLHttpRequest();

    xhr.onreadystatechange = () => {
      if(xhr.readyState === 4){ //Hecho = 4
        if(xhr.status === 200) {
          console.log("Respuesta del servidor para distancia 1:",
xhr.responseText);

          dist1.innerHTML = xhr.responseText;
          resolve();
        }else{
          reject("Error al obtener distancia 1:", xhr.statusText);
        }
        ejecutar = false;
      }
    }
    xhr.open('GET', "/Sensor1")
    xhr.send();
  });
}

function actualizarDistancia2(processTemp){
  if(ejecutar){
    console.log("La actualización del sensor 2 ya se está ejecutando. No
se permiten llamadas concurrentes.");
    return;
  }

  ejecutar = true;

  return new Promise((resolve, reject) => {
    let xhr = new XMLHttpRequest();

    xhr.onreadystatechange = () => {
      if (xhr.readyState === 4) { //Hecho = 4
        if(xhr.status === 200) {
          console.log("Respuesta del servidor para distancia 2:",
xhr.responseText);

```

```

        dist2.innerText = xhr.responseText;
        resolve();
    }else{
        reject("Error al obtener distancia 2:", xhr.statusText);
    }
    ejecutar = false;
}
}
xhr.open('GET', "/Sensor2")
xhr.send();
});
}

async function actualizarDistancias(){
    try {
        await actualizarDistancia1();
        await actualizarDistancia2();

    } catch (error) {
        console.error(error);
    }
}

function cambiarDistancia1(){
    const elementoRango1 = document.getElementById('rangoP1');
    console.log("Cambiando distancia del sensor a " + elementoRango1.value + "
para el parqueadero 1");
    consultaGET("http://%ip/nivel1?valor=" + elementoRango1.value);
}

function cambiarDistancia2(){
    const elementoRango2 = document.getElementById('rangoP2');
    console.log("Cambiando distancia del sensor a " + elementoRango2.value + "
para el parqueadero 2");
    consultaGET("http://%ip/nivel2?valor=" + elementoRango2.value);
}

function Pagina2(){
    window.location.href = "/actualizacion";
}

function consultaGET(consulta){
    const Http = new XMLHttpRequest();
    console.log(`Consultando ${consulta}`)
    Http.open("GET", consulta);
    Http.send();

    Http.onreadystatechange = (e) =>{
        console.log(Http.status);
    }
}

```

```

        console.log(Http.responseText);
    };
}

document.addEventListener('DOMContentLoaded', () => {
    console.log('Página cargada!');
    setInterval(actualizarDistancias, tiempoAgotado);
});
</script>

</body>
</html>

```

Para asegurar que el formato de la página web no afecte al resto del código en Arduino, se utiliza la función `R"====()===="` asignada a una variable de tipo String, donde se inserta el código del portal web entre los paréntesis.

Además, se utilizan herramientas para optimizar el espacio en microcontroladores como HTML MINIFIER, que comprime las secciones de CSS y HTML. Esta herramienta también puede comprimir otros elementos como los nombres de las variables, entre otros.

En cuanto a la actualización de los valores de las variables que almacenan la información de los sensores para su visualización en las páginas web, se utiliza una solicitud HTTP GET desde la página web ya cargada al servidor. Esto se logra con código JavaScript, como se ilustra en la figura del proyecto.

```

211     function actualizarDistancia2(processTemp){
212         if(ejecutar){
213             console.log("La actualización del sensor 2 ya se está ejecutando. No se permiten llamadas concurrentes.");
214             return;
215         }
216
217         ejecutar = true;
218
219         return new Promise((resolve, reject) => {
220             let xhr = new XMLHttpRequest();
221
222             xhr.onreadystatechange = () => {
223                 if (xhr.readyState === 4) { //Hecho = 4
224                     if(xhr.status === 200) {
225                         console.log("Respuesta del servidor para distancia 2:", xhr.responseText);
226                         dist2.innerHTML = xhr.responseText;
227                         resolve();
228                     }else{
229                         reject("Error al obtener distancia 2:", xhr.statusText);
230                     }
231                     ejecutar = false;
232                 }
233             }
234             xhr.open('GET', "/Sensor2")
235             xhr.send();
236         });
237     }
238
239     async function actualizarDistancias(){
240         try {
241             await actualizarDistancia1();
242             await actualizarDistancia2();
243         } catch (error) {
244             console.error(error);
245         }
246     }
247 }

```

Figura 87. Función de la página web para solicitar los valores de los sensores al servidor.
Fuente: El Autor.

La función en cuestión determina mediante una bandera si ya se está solicitando la información para evitar peticiones recurrentes que puedan ocasionar fallos. Una vez que la bandera pasa el control, se invoca una función asíncrona que se activa al cargar el sitio web, utilizando un bucle de ejecución regulado por una variable de tiempo, como se ilustra en la figura del proyecto.

```

277     document.addEventListener('DOMContentLoaded', () => {
278         console.log('Página cargada!');
279         setInterval(actualizarDistancias, tiempoAgotado);
280     });

```

Figura 88. Código de verificación de la carga del portal web.
Fuente: EL Autor.

El intervalo de tiempo mencionado se define como una variable numérica expresada en milisegundos, permitiendo su ajuste según las necesidades específicas del proyecto. Este enfoque se implementa para evitar la concurrencia de solicitudes entre los sensores, asegurando que cada petición se realice secuencialmente. El objetivo es prevenir posibles conflictos que podrían surgir debido a la superposición de datos, lo cual podría afectar el formateo correcto de la información en la página web. Cada solicitud espera su turno mediante una función asíncrona, donde se utiliza un objeto XMLHttpRequest para verificar el estado del servidor y recibir los datos correspondientes. Esta metodología incluye verificaciones exhaustivas para manejar posibles errores durante la recepción y procesamiento de la información del servidor, como se ilustra en la figura donde se muestra la solicitud GET para obtener datos del sensor 2.

```

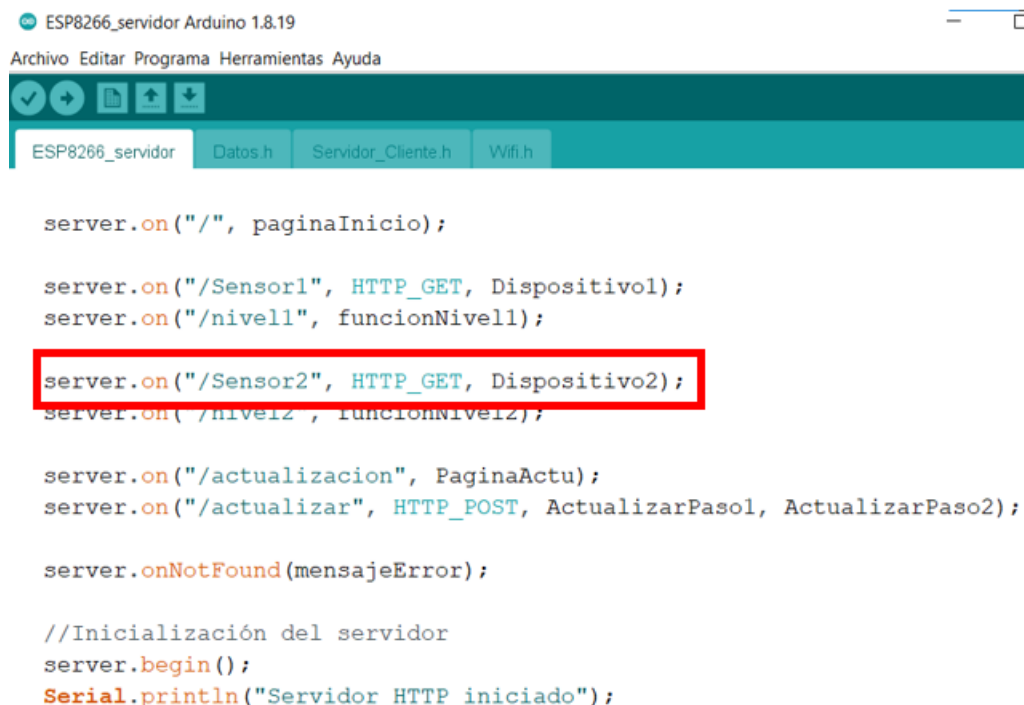
211     function actualizarDistancia2(processTemp){
212         if(ejecutar){
213             console.log("La actualización del sensor 2 ya se está ejecutando. No se permiten llamadas concurrentes.");
214             return;
215         }
216
217         ejecutar = true;
218
219         return new Promise((resolve, reject) => {
220             let xhr = new XMLHttpRequest();
221
222             xhr.onreadystatechange = () => {
223                 if (xhr.readyState === 4) { //Hecho = 4
224                     if(xhr.status === 200) {
225                         console.log("Respuesta del servidor para distancia 2:", xhr.responseText);
226                         dist2.innerHTML = xhr.responseText;
227                         resolve();
228                     }else{
229                         reject("Error al obtener distancia 2:", xhr.statusText);
230                     }
231                     ejecutar = false;
232                 }
233             }
234             xhr.open('GET', "/Sensor2")
235             xhr.send();
236         });
237     }
238
239     async function actualizarDistancias(){
240         try {
241             await actualizarDistancia1();
242             await actualizarDistancia2();
243         } catch (error) {
244             console.error(error);
245         }
246     }
247

```

Figura 89. Código de solicitud GET al servidor desde la página web.

Fuente: El Autor.

Una vez revisado desde la perspectiva del sitio web, es crucial comprender cómo responde el servidor a estas solicitudes. Para identificar la petición de datos del sensor 2, se utiliza el mismo método de llamada al servidor, como se ilustra en la figura.



```

ESP8266_servidor Arduino 1.8.19
Archivo Editar Programa Herramientas Ayuda
ESP8266_servidor Datos.h Servidor_Cliente.h Wifi.h

server.on("/", paginaInicio);

server.on("/Sensor1", HTTP_GET, Dispositivo1);
server.on("/nivel1", funcionNivel1);
server.on("/Sensor2", HTTP_GET, Dispositivo2);
server.on("/nivel2", funcionNivel2);

server.on("/actualizacion", PaginaActu);
server.on("/actualizar", HTTP_POST, ActualizarPaso1, ActualizarPaso2);

server.onNotFound(mensajeError);

//Inicialización del servidor
server.begin();
Serial.println("Servidor HTTP iniciado");

```

Figura 90. Código de respuesta a la solicitud GET de la página web.
Fuente: El Autor.

En el proceso, se realiza una llamada a la función "Dispositivo2", la cual responde cuando desde el portal se envía una solicitud con "/Sensor2" justo después de la IP, indicando al servidor que se trata de una solicitud GET.

La función "Dispositivo2" se encarga de enviar la información correspondiente del sensor 2, como se ilustra en la figura.

```

void Dispositivo2() {
  server.sendHeader("Access-Control-Allow-Origin", "*");
  server.send(200, "text/plain", String(dist_sen2));
}

```

Figura 91. Función que responde a la solicitud del sitio web para la distancia del sensor 2.
Fuente: El Autor.

En estos casos, las líneas de código primero permiten el acceso a la información desde cualquier dispositivo conectado al portal web. Segundo, el servidor responde con un texto plano que contiene la variable que almacena la distancia del sensor 2, convertida a una cadena de texto.

Además, se crea una página de respuesta del servidor relacionada con el error web 404. Este error indica la falta de información solicitada, por ejemplo, al realizar una petición GET desde la barra de búsqueda para un recurso que no existe o no está inicializado en el servidor. La página de error 404 es mostrada para comunicar al usuario que el recurso buscado no está disponible, como se ilustra en la figura.

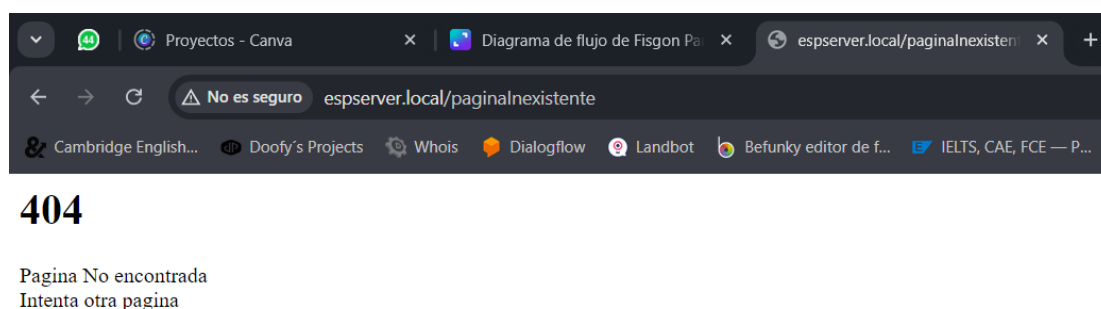


Figura 92. Página de error.
Fuente: El Autor.

Este apartado finaliza la descripción del proceso completo de comunicación, desde la recepción de datos de los sensores hasta su presentación al usuario final en el portal web.

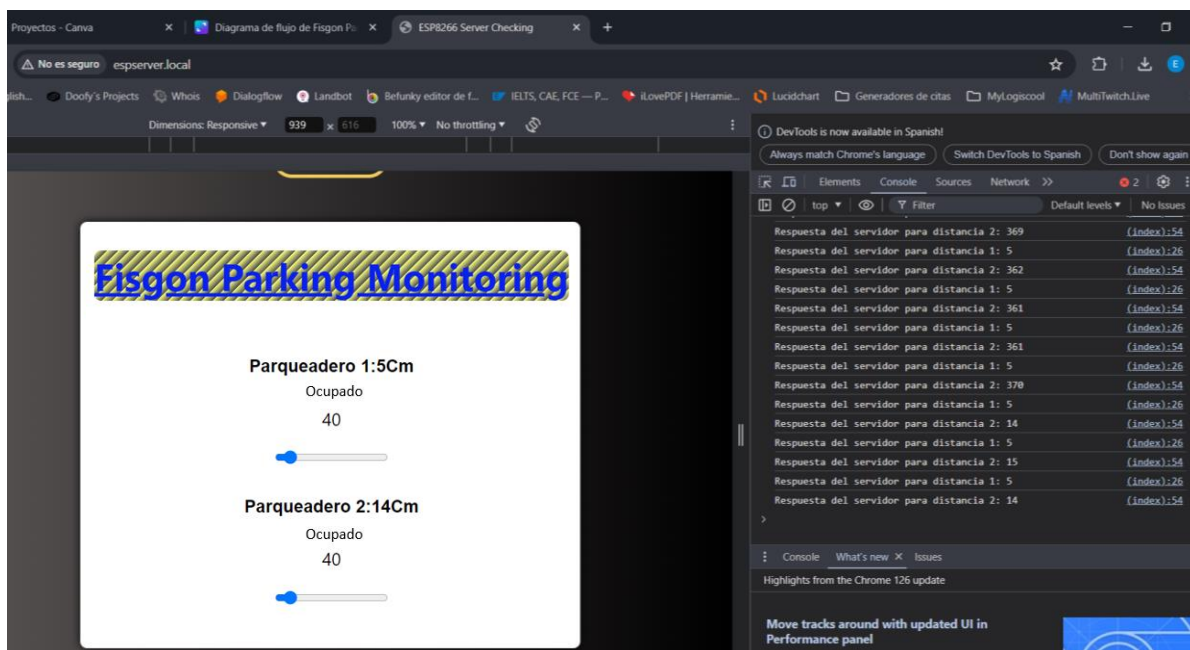


Figura 93. Funcionamiento del portal web en tiempo real.
Fuente: El Autor.

Finalmente, hasta este punto el proyecto habrá concluido de forma local. Ahora, lo importante es que pueda ser visto desde cualquier parte de Internet, ya que de otra manera sería complejo para las personas que no se encuentran en la zona donde está instalado el sistema prototipo, debido a que no podrán acceder al servicio que se ofrece si la página solo es visible de forma local. Para solventar esta problemática, se implementó un servicio de terceros gratuito que permite la resolución de la web ubicada en local, por medio de un túnel que accede al contenido que está disponible, generando una URL que permite la visualización desde cualquier parte del mundo con cualquier navegador y cualquier dispositivo, como se muestra en la figura.

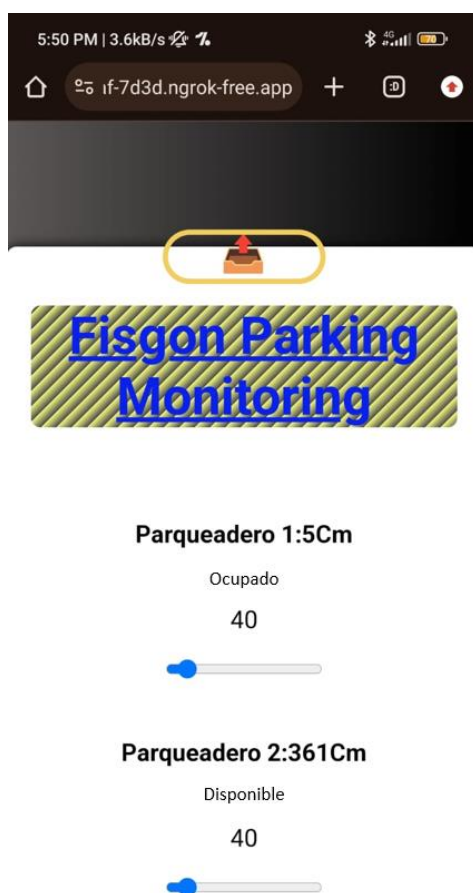


Figura 94. Acceso a la página del servidor desde Internet.
Fuente: El Autor.

En el resultado de la figura se muestra la página web vista desde Internet, utilizando la configuración del software de Ngrok para acceder a ella. Esto se logra aplicando la configuración mencionada en el marco teórico, completando así la implementación del sistema.

6.6. Pruebas de funcionamiento

En caso de cambios en la red Wifi que esté conectado el servidor, será necesario revisar en las redes Wifi disponibles ya que se mostrará nuevamente una red de configuración.



Figura 95. Apartado de configuración de redes Wifi en sistema operativo Windows 10.
Fuente: El Autor.

Posterior a la conexión con la red que maneja las credenciales de la red Wifi, será necesario ingresar en un navegador y colocar la IP del servidor, que siempre será la misma.

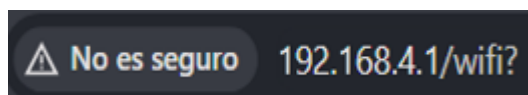


Figura 96. IP del servidor en la barra de navegación de Google Chrome.
Fuente: El Autor.

Una vez se ingresa a esta página, se procede a realizar los pasos anteriormente explicados en el punto 6.3 que habla de la implementación del sistema prototipo en el apartado de cómo fue implementado Wifi Manager.

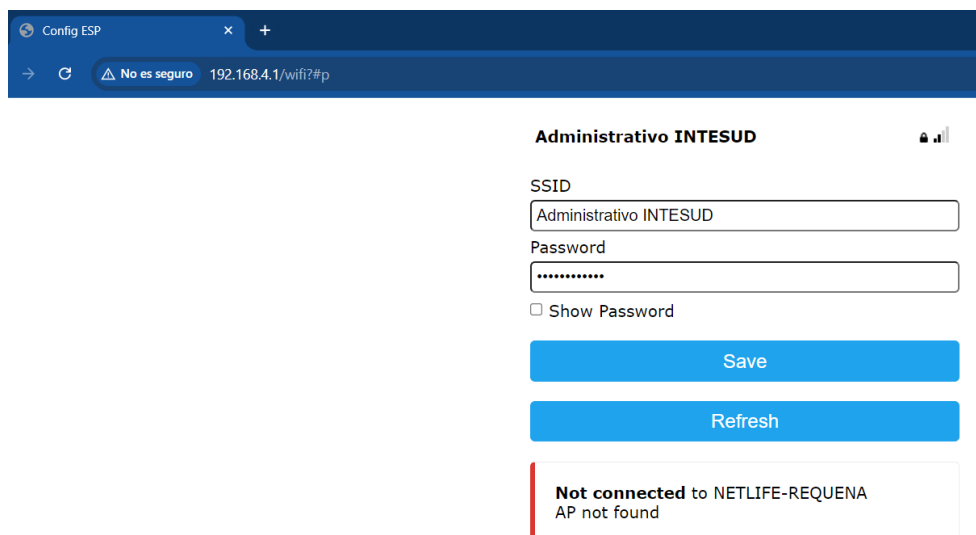


Figura 97. Página de ingreso de credenciales de Wifi Manager.
Fuente: El Autor.

Dentro del portal de configuración se ingresa la información correspondiente a la red wifi.

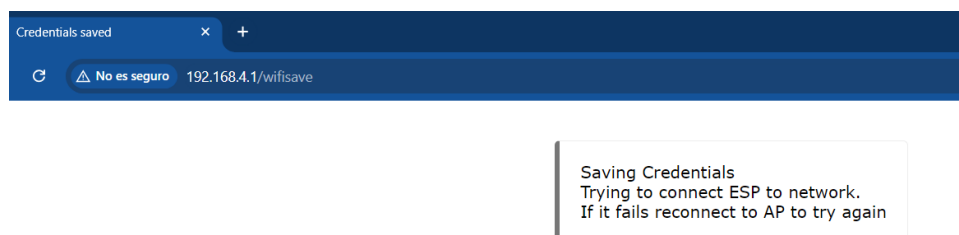


Figura 98. Página de confirmación del servidor para las credenciales ingresadas.
Fuente: El Autor.

Una vez ingresadas las credenciales aparecerá el mensaje indicado en la figura anterior que menciona haber salvado la información ingresada para la próxima conexión.



Figura 99. Red levantada por el servidor después de establecer conexión.
Fuente: El Autor.

El siguiente paso será verificar que se muestre la red Wifi que crea posteriormente el servidor que es la indicada en la figura anterior. El resto del proceso de conexión es automático pasados unos pocos segundos.

Es importante resaltar que en caso de desconexión de la computadora que ofrece servicio de túnel hacia el Internet con la herramienta de Ngrok, será necesario volver a configurar el portal desde su terminal de comandos solo en caso de contar con el servicio gratuito.

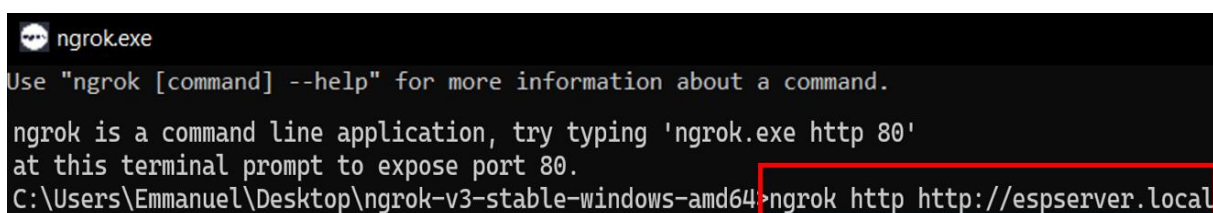
A screenshot of a terminal window titled 'ngrok.exe'. The terminal text reads: 'Use "ngrok [command] --help" for more information about a command. ngrok is a command line application, try typing 'ngrok.exe http 80' at this terminal prompt to expose port 80. C:\Users\Emmanuel\Desktop\ngrok-v3-stable-windows-amd64>ngrok http http://espserver.local'. The command 'ngrok http http://espserver.local' is highlighted with a red rectangular box.

Figura 100. Conexión del ESP8266 servidor al servicio de Ngrok para salida a Internet desde su terminal.
Fuente: El Autor.

Deberá ser configurado como se muestra en la figura anterior, de forma que se utiliza el comando para iniciar el servicio y se apunta a la dirección DNS configurada en el servidor que gestiona los parqueaderos, este también direcciona de la forma mostrada al puerto 80 que es la salida a Internet.

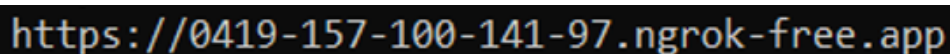
A screenshot of a black rectangular box containing the URL 'https://0419-157-100-141-97.ngrok-free.app' in white text.

Figura 101. IP generada por el servicio de Ngrok para la salida a Internet.
Fuente: El Autor.

El resultado genera una IP como la mostrada en la figura anterior. La configuración es debido a que la IP que ofrece la herramienta cambia con cada conexión realizada y solo dispone de una en su versión gratuita.

```

Ngrok - ngrok http http://espserver.local
ngrok
Try our new Traffic Inspector: https://ngrok.com/r/ti
Session Status      online
Account             intesudngrok@gmail.com (Plan: Free)
Version             3.14.0
Region              South America (sa)
Latency             130ms
Web Interface       http://127.0.0.1:4040
Forwarding           https://42af-186-4-226-54.ngrok-free.app -> http://espserver.local:80
Connections
  ttl   opn   rt1   rt5   p50   p90
   0    0    0.00 0.00 0.00 0.00

```

Figura 102. Resultado y asignación del link por parte del software al realizar la conexión con el ESP8266
Fuente: El Autor.

En la imagen anterior se muestra donde proporciona el enlace la Internet por parte del software para hacer túnel a la página web del ESP servidor.

Para finalizar, se realizaron pruebas de funcionamiento para verificar cada paso del proceso anteriormente expuesto y para depurar cualquier error presente a lo largo del proyecto, asegurando así el correcto funcionamiento del sistema. Las pruebas se llevaron a cabo con diversos objetos a distintas distancias y en varias direcciones a la misma altura para comprobar la precisión de los sensores y validar las hipótesis teóricas.

Se realizaron múltiples sesiones de pruebas con la configuración Wifi desde diferentes redes conectadas, para confirmar el funcionamiento del sistema. Posteriormente, se verificó la conexión de los clientes al servidor utilizando las credenciales de la red del servidor. También se realizaron diversas comprobaciones de la comunicación UDP del módulo cliente con el servidor, corroborando su correcta implementación y analizando las interacciones a través del monitor serial conectado al servidor, visualizado desde el Arduino IDE.

Para las páginas web, se utilizó el editor de texto Visual Studio Code con la extensión Live Server para simular en tiempo real las acciones y cambios realizados en la programación de HTML, CSS y JavaScript. Finalmente, se cargó todo el programa tanto en los clientes como en el servidor, y se accedió al servidor desde múltiples dispositivos, como teléfonos móviles y

laptops conectados a la misma red del servidor, confirmando la correcta programación de JavaScript para las solicitudes realizadas al servidor.

Por último, todo el sistema se puso en funcionamiento constante durante 3 días dentro del parqueadero del Instituto como prueba final, tomando resultados de aquel personal que decidieron colocar sus vehículos en los puestos con módulos instalados, donde se evaluó por medio de una encuesta la experiencia de los usuarios con el prototipo. Compartiendo la información receptada con el autor del proyecto para su posterior incorporación al documento, para más información revisar el anexo 11.

7. Conclusiones y Recomendaciones

7.1. Conclusiones

- Se investigó y desarrolló un marco teórico sólido, lo que permitió identificar las tecnologías y componentes más adecuados para el diseño e implementación del sistema. Este análisis teórico fue clave para comprender las necesidades específicas de los sensores, módulos electrónicos y su integración dentro del contexto de un parqueadero institucional.
- Se seleccionaron y configuraron los sensores y módulos electrónicos adecuados para el proyecto, tomando en cuenta las limitaciones del espacio disponible en el parqueadero. El diseño consideró factores como el posicionamiento óptimo de los sensores ultrasónicos y la cantidad de módulos necesarios, asegurando una cobertura eficiente y adaptable a las características físicas del lugar.
- El sistema prototipo basado en el microcontrolador ESP8266 fue implementado con éxito. Este microcontrolador, gracias a su módulo Wi-Fi integrado y capacidades de procesamiento, permitió la conexión eficiente entre los sensores, los módulos electrónicos y el sistema web, asegurando su funcionalidad.
- Se desarrolló una página web funcional utilizando HTML, CSS y JavaScript, lo que permitió a los usuarios acceder a información en tiempo real sobre la disponibilidad de los parqueaderos. La página web destaca por su diseño intuitivo y accesible, optimizado para dispositivos móviles, lo que mejora significativamente la experiencia del usuario.
- Se logró integrar la programación de los sensores y la funcionalidad de la página web con el microcontrolador ESP8266. Esta integración permitió que los datos capturados por los sensores fueran procesados y enviados en tiempo real al portal web, facilitando la interacción del usuario con el sistema y mejorando la eficiencia del monitoreo automatizado.

- Se realizaron pruebas exhaustivas para validar el funcionamiento del sistema en condiciones reales. Estas pruebas consideraron factores técnicos y ambientales, asegurando que el sistema cumpliera con las expectativas en términos de precisión, eficiencia y adaptabilidad a las necesidades del parqueadero institucional.
- El prototipo desarrollado logró cumplir con el objetivo general del proyecto, al proporcionar un sistema automatizado de monitoreo que comunica a los usuarios, mediante un portal web, la información en tiempo real sobre la disponibilidad de los parqueaderos. Este sistema representa una solución innovadora y eficiente, que mejora la gestión del estacionamiento en el Instituto Superior Tecnológico Sudamericano Quito.
- Finalmente, se concluye que las tecnologías utilizadas durante el desarrollo del proyecto, como el microcontrolador ESP8266 y las herramientas web, han demostrado ser efectivas y accesibles. Sin embargo, el avance constante de la tecnología puede introducir soluciones aún más eficientes en el futuro, simplificando tanto el desarrollo como la experiencia del usuario final.
- Como conclusión del objetivo general el prototipo generado del presente proyecto cumplió con su idea inicial de poder comunicar a los usuarios a través de un portal web la información del estado actual de los parqueaderos del local.
- Otra conclusión que se puede acotar es que las tecnologías usadas durante el desarrollo de este proyecto pudieron haber cambiado en el futuro, la tecnología no solo avanza a cosas más complejas, sino que también se persigue que sea más fácil de utilizar y no solo para el usuario final, sino para aquellos que la crean desde un principio y durante el proceso logrando que sea lo más accesible para cualquier persona.
- Para cables eléctricos es preferible usar un calibre de 8 AWG debido a que es la medida utilizada para cometidas eléctricas.

7.2. Recomendaciones

- Aunque Arduino cuenta con un *shield* Wi-Fi, se recomienda evaluar alternativas como el ESP8266, que ofrece ventajas significativas en términos de tamaño, capacidad y facilidad de integración, como se demostró en este proyecto.
- Es esencial mantener el sistema en un ambiente controlado, considerando factores que puedan afectar la precisión de los sensores ultrasónicos y la estabilidad de las comunicaciones Wi-Fi.
- Complementar la investigación teórica con recursos prácticos, como foros de Arduino, plataformas de preguntas relacionadas con programación, tutoriales en YouTube y herramientas de inteligencia artificial. Estos recursos pueden ser de gran ayuda para solucionar problemas y optimizar el desarrollo del proyecto.
- Verificar siempre la corriente suministrada al ESP8266, ya que utilizar una fuente distinta a 3.3V puede dañar el microcontrolador. Es recomendable incluir filtros de corriente para garantizar un suministro adecuado y estable.
- Realizar pruebas en un entorno real con todo el software instalado, además de simulaciones. Estas pruebas, incluso a pequeña escala, permiten identificar y resolver problemas que podrían no ser evidentes en simulaciones virtuales.
- Si el sistema requiere acceso desde cualquier parte del mundo, se recomienda utilizar servicios de nombres de dominio para expandir las utilidades del portal web, según las necesidades del caso.
- Tener en cuenta las limitaciones del protocolo UDP, que no establece una conexión directa y puede ocasionar problemas de comunicación debido a una configuración incorrecta de puertos o IP. Además, existe la posibilidad de pérdida de información, ya que este protocolo prioriza la velocidad sobre la fidelidad.

- Implementar mecanismos en JavaScript para evitar solicitudes simultáneas a los sensores en el portal web. Es recomendable programar funciones que gestionen las solicitudes de forma consecutiva, asegurando que cada proceso se complete antes de iniciar otro.
- Verificar que los módulos electrónicos estén correctamente energizados, ya que una alimentación inadecuada puede provocar reinicios constantes y fallos en el sistema.
- Si el código JavaScript está directamente incluido en el archivo HTML, es importante colocarlo al final de las etiquetas del <body> mediante las etiquetas <script>. Esto asegura que el código solo se ejecute una vez que todos los elementos de la página hayan sido cargados, evitando errores y conflictos.
- Utilizar herramientas de software como WireShark para capturar y analizar paquetes UDP. Esta herramienta de uso libre facilita la depuración y resolución de problemas relacionados con la comunicación de datos.
- Aunque en este proyecto se utilizó Ngrok para crear túneles locales, existen otras opciones como MQTT en ESP8266. Es recomendable evaluar estas alternativas para determinar cuál se adapta mejor a las necesidades específicas del sistema.

Referencias

- ACV, A. (. (14 de 11 de 2022). *El curso del Hacker*. Recuperado el 04 de 07 de 2024, de <https://www.elcursodelhacker.com/ngrok/>
- Alonso, M. (. (07 de 02 de 2024). *Asana*. Recuperado el 01 de 07 de 2024, de <https://asana.com/es/resources/what-is-the-cloud>
- Amazon Web Services. (s.f.). *Amazon Web Services*. Recuperado el 01 de 07 de 2024, de <https://aws.amazon.com/es/what-is/wan/>
- Anda, N. d. (24 de 10 de 2018). *Factor Evolución*. Recuperado el 10 de 07 de 2024, de <https://www.factor.mx/portal/base-de-conocimiento/cables-jumper-macho-macho/#:~:text=Un%20jumper%20o%20saltador%20es,elementos%20ingresados%20en%20dicho%20tablero.>
- Anda, N. d. (24 de 10 de 2018). *Factor Evolución*. Recuperado el 10 de 07 de 2024, de <https://www.factor.mx/portal/base-de-conocimiento/cables-jumper-macho-hembra/>
- Anda, N. d. (10 de 10 de 2018). *Factor Evolución*. Recuperado el 10 de 07 de 2024, de <https://www.factor.mx/portal/base-de-conocimiento/simbologia-electronica/>
- Arduino Spain*. (s.f.). Recuperado el 29 de 04 de 2024, de <https://sp.arduino-france.site/modulos/>
- Aruba Networks. (s.f.). *Aruba Networks*. Recuperado el 01 de 07 de 2024, de <https://www.arubanetworks.com/es/faq/que-es-una-wan/>
- AWS Inc. o sus filiales. (s.f.). *Amazon Web Services*. Recuperado el 30 de 06 de 2024, de <https://aws.amazon.com/es/what-is/computer-networking/>
- Carrod electronica*. (s.f.). Recuperado el 24 de 07 de 2024, de <https://www.carrod.mx/products/modulo-de-alimentacion-para-protoboard-5-v-y-3-3-v->

mb102#:~:text=Módulo%20que%20permite%20alimentar%20un,de%20electrónica%2C%20robótica%20y%20automatización.

Cloudflare. (s.f.). *Cloudflare*. Recuperado el 01 de 07 de 2024, de <https://www.cloudflare.com/es-es/learning/network-layer/what-is-a-metropolitan-area-network/>

Cloudflare Inc. (s.f.). *Cloudflare*. Recuperado el 01 de 07 de 2024, de <https://www.cloudflare.com/es-es/learning/network-layer/what-is-a-lan/>

colaboradores de Wikipedia. (13 de 06 de 2023). *Wikipedia, la enciclopedia libre*. Recuperado el 10 de 07 de 2024, de https://es.wikipedia.org/wiki/S%C3%ADmbolo_electr%C3%B3nico

contributors, M. (18 de 07 de 2023). *MDN Web Docs*. Recuperado el 16 de 05 de 2024, de https://developer.mozilla.org/es/docs/Learn/Getting_started_with_the_web/HTML_basics

contributors, M. (2 de 08 de 2023). *MDN Web Docs*. Recuperado el 16 de 05 de 2024, de https://developer.mozilla.org/es/docs/Learn/CSS/First_steps/What_is_CSS

contributors, M. (2 de 08 de 2023). *MDN Web Docs*. Recuperado el 16 de 05 de 2024, de https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/What_is_JavaScript

E. Ecda. (23 de 08 de 2023). *El Cajon de Ardu*. Recuperado el 29 de 04 de 2024, de <https://www.elcajondeardu.com/arduino-nano-todo-lo-que-necesitas-saber/>

Equipo editorial, Etecé. (19 de 11 de 2023). *Concepto*. Recuperado el 30 de 06 de 2024, de <https://concepto.de/red-de-computadoras/>

García, I. J. (30 de 03 de 2021). *Conectividad y Soluciones de TI / Servnet* . Recuperado el 15 de 05 de 2024, de <https://www.servnet.mx/blog/backend-y-frontend-partes-fundamentales-de-la-programacion-de-una-aplicacion-web>

GoDaddy, E. d. (28 de 06 de 2023). *GoDaddy Resources - LATAM*. Recuperado el 15 de 05 de 2024, de <https://www.godaddy.com/resources/latam/stories/pagina-web-que-es-tipos#:~:text=Una%20p%C3%A1gina%20web%20es%20un,y%20presentada%20de%20forma%20visual>.

Gonzalez, O. (s.f.). *Bricogeek.com*. Recuperado el 05 de 06 de 2024, de <https://lab.bricogeek.com/tutorial/guia-de-modelos-arduino-y-sus-caracteristicas/arduino-uno>

Hausen. (14 de 11 de 2010). *Domotica – hausen.ec*. Recuperado el 31 de 08 de 2023, de https://hausen.com.ec/domotica/?gclid=Cj0KCQjwuZGnBhD1ARIsACxbAViXC3k1rRDzVGpCteHnki4lnxNroIZoIqJZ9l8gfQ2-L_GtEQ0A9a0aApqoEALw_wcB

Hernández, L. d. (s.f.). *Programarfacil Arduino y Home Assistant*. Recuperado el 17 de 05 de 2024, de <https://programarfacil.com/podcast/esp8266-wifi-coste-arduino/>

Home / Bookdown. (s.f.). Recuperado el 13 de 05 de 2024, de https://bookdown.org/alberto_brunete/intro_automatica/sensores-industriales.html

Logitek. (18 de 06 de 2019). *logitek*. Recuperado el 31 de 08 de 2023, de <https://logitek.com.ec/>

MCI Electronics. (12 de 09 de 2015). *Arduino.cl - Compra tu Arduino en Línea*. Recuperado el 21 de 01 de 2024, de <https://arduino.cl/producto/arduino-mega-2560/#:~:text=Arduino%20Mega%20es%20una%20tarjeta,implementa%20el%20lenguaje%20Processing%2FWiring>.

Pini, A. (21 de 04 de 2021). *Digikey.com*. (Colaboración de Editores de DigiKey de América del Norte) Recuperado el 13 de 05 de 2024, de <https://www.digikey.com/es/articles/the-fundamentals-of-proximity-sensors-selection-and-use-industrial-automation>

Prometec. (s.f.). *comprar arduino con tutoriales para todos los niveles*. Recuperado el 09 de 05 de 2024, de <https://www.prometec.net/modelos-esp8266/>

- Santos, D. (25 de 07 de 2023). *Blog de HubSpot | Marketing, Ventas, Servicio al Cliente y Sitio Web* . Recuperado el 16 de 05 de 2024, de <https://blog.hubspot.es/website/que-es-css>
- Securitas Direct. (10 de 07 de 2018). *Protegiendo Personas*. Recuperado el 14 de 05 de 2024, de <https://protegiendopersonas.es/sensores-infrarrojos-que-son-y-para-que-se-utilizan/>
- Solectroshop. (s.f.). *Tu tienda de Arduino, Raspberry, Micro:Bit, Sparkfun*. Recuperado el 29 de 04 de 2024, de <https://solectroshop.com/es/1013-modulos-arduino>
- Villagran, V. (. (31 de 01 de 2023). *Mega Lámparas, Tienda de lámparas*. . Recuperado el 03 de 07 de 2024, de <https://megalamparas.com.gt/que-son-las-luces-led/>
- Wikimedia, C. d. (17 de 03 de 2004). *Wikipedia, la enciclopedia libre*. Recuperado el 16 de 05 de 2024, de https://es.wikipedia.org/wiki/IEEE_802.11
- Wikimedia, C. d. (27 de 07 de 2019). *Wikipedia, la enciclopedia libre*. Recuperado el 29 de 04 de 2024, de https://es.wikipedia.org/wiki/Arduino_IDE
- Wikimedia, C. d. (25 de 5 de 2019). *Wikipedia, la enciclopedia libre*. Recuperado el 21 de 10 de 2023, de https://es.wikipedia.org/wiki/Arduino_Uno
- Xfinity. (16 de 04 de 2024). Recuperado el 17 de 05 de 2024, de Xfinity Help & Support: <https://es.xfinity.com/support/articles/intro-wifi-home-network>
- Yúbal, F. (23 de 9 de 2022). *Xataka - Tecnología y gadgets, móviles, informática, electrónica*. (F. Yúbal, Productor) Recuperado el 20 de 10 de 2023, de <https://www.xataka.com/basics/que-arduino-como-funciona-que-puedes-hacer-uno>

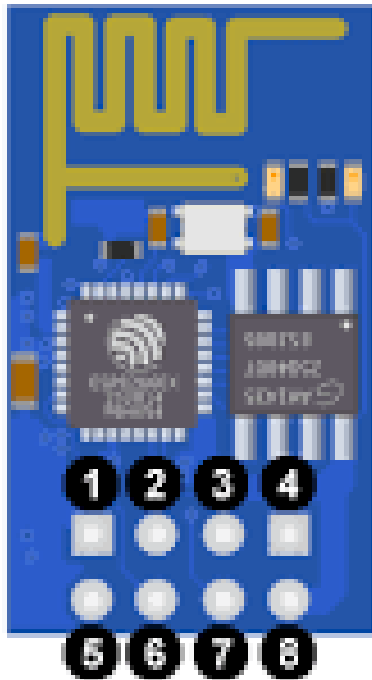
ANEXOS

ANEXO 1: Comandos AT ESP-01

Los comandos AT sirven como guía de funcionamiento y configuración en tiempo real durante la ejecución del ESP8266, aunque también es posible programarlo directamente con el código escrito. Este enfoque reemplazará la pila de comandos AT que están precargados.

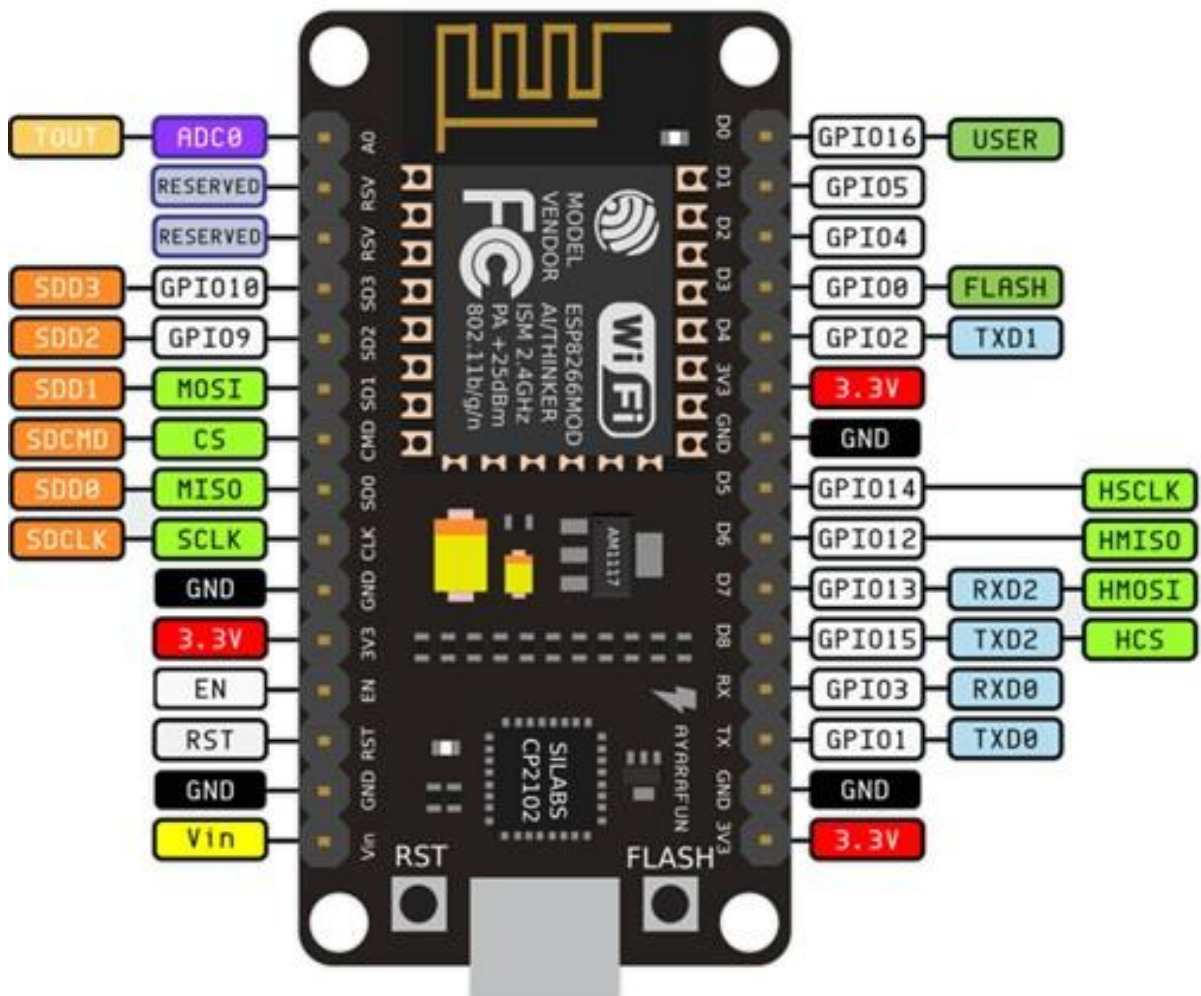
Commands	Description	Type	Set/Execute	Inquiry	test	Parameters	Examples
AT+RST	restart the module	basic	-	-	-	-	
AT+CWMODE	wifi mode	wifi	AT+CWMODE=<mode>	AT+CWMODE?	AT+CWMODE=?	1= Sta, 2= AP, 3=both	
AT+CWJAP	join the AP	wifi	AT+ CWJAP =<ssid>,<pwd >	AT+ CWJAP?	-	ssid = ssid, pwd = wifi password	
AT+CWLAP	list the AP	wifi	AT+CWLAP				
AT+CWQAP	quit the AP	wifi	AT+CWQAP	-	AT+CWQAP=?		
AT+ CWSAP	set the parameters of AP	wifi	AT+ CWSAP= <ssid>,<pwd>,<chl>,<ecn>	AT+ CWSAP?		ssid, pwd, chl = channel, ecn = encryption	Connect to your router: : AT+CWJAP="YOURSSID","helloworld"; and check if connected: AT+CWJAP?
AT+ CIPSTATUS	get the connection status	TCP/IP	AT+ CIPSTATUS				
AT+CIPSTART	set up TCP or UDP connection	TCP/IP	1)single connection (+CIPMUX=0) AT+CIPSTART= <type>,<addr>,<port>; 2) multiple connection (+CIPMUX=1) AT+CIPSTART= <id> <type>,<addr>,<port>	-	AT+CIPSTART=?	id = 0-4, type = TCP/UDP, addr = IP address, port= port	Connect to another TCP server, set multiple connection first: AT+CIPMUX=1; connect: AT+CIPSTART=4,"TCP","X1.X2.X3.X4",9999
AT+CIPSEND	send data	TCP/IP	1)single connection(+CIPMUX=0) AT+CIPSEND= <length>; 2) multiple connection (+CIPMUX=1) AT+CIPSEND= <id>,<length>		AT+CIPSEND=?		send data: AT+CIPSEND=4,15 and then enter the data
AT+CIPCLOSE	close TCP or UDP connection	TCP/IP	AT+CIPCLOSE=<id> or AT+CIPCLOSE		AT+CIPCLOSE=?		
AT+CIFSR	Get IP address	TCP/IP	AT+CIFSR		AT+ CIFSR=?		
AT+ CIPMUX	set mutiple connection	TCP/IP	AT+ CIPMUX=<mode>	AT+ CIPMUX?		0 for single connection 1 for mutiple connection	
AT+ CIPSERVER	set as server	TCP/IP	AT+ CIPSERVER= <mode>[,<port>]			mode 0 to close server mode, mode 1 to open; port = port	turn on as a TCP server: AT+CIPSERVER=1,8888, check the self server IP address: AT+CIFSR=?

ANEXO 2: Esquema de pines del ESP-01



- | | |
|---------|---------|
| ① GND | ⑤ TXD |
| ② GPIO2 | ⑥ CH_PD |
| ③ GPIO0 | ⑦ RESET |
| ④ RXD | ⑧ Vcc |

ANEXO 3: Esquema de pines del ESP8266 NodeMCU V3 1.0 (ESP-12E).

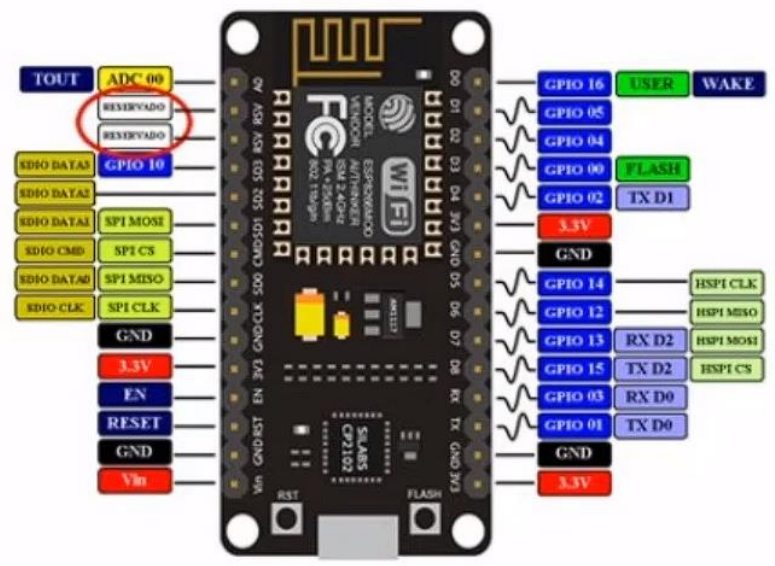


En la actualidad, existen varios modelos distintos del NodeMCU, y la distribución de los pines en cada módulo puede variar según el fabricante del circuito impreso. Por esta razón, se anexan algunas de las distribuciones más conocidas y comercializadas.

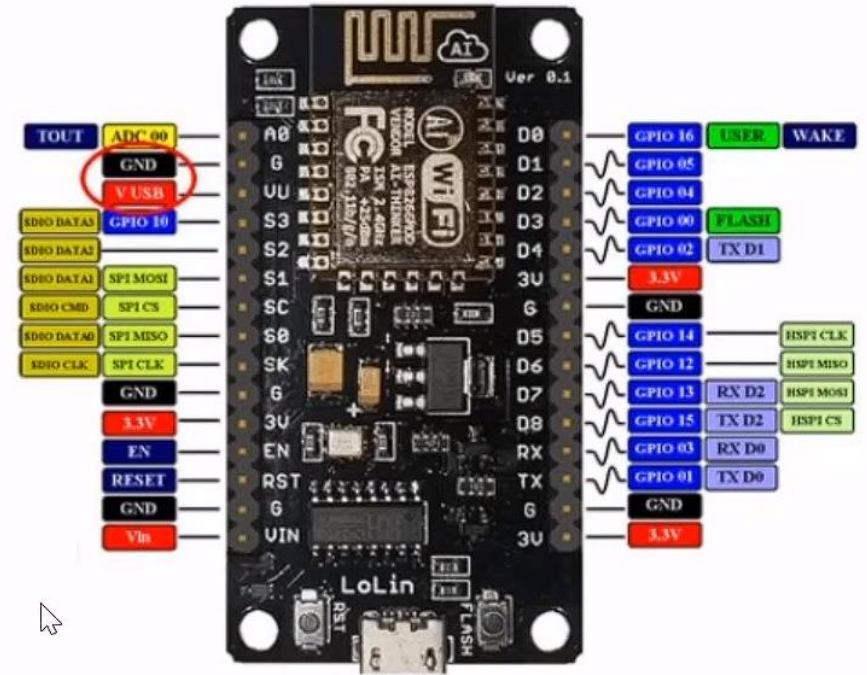
PINOUT NodeMCU 1.0



AMICA / DOIT (V2)



LOLIN (V3)



- Vih** ALIMENTACIÓN EXTERNA (de +5V a +10V).
- 3.3V** ALIMENTACIÓN INTERNA (desde la placa a dispositivos).
- VUSB** ALIMENTACIÓN INTERNA PROVENIENTE DEL MiniUSB (desde la placa a dispositivos +5V).
- GND** TIERRA (GND Ground).
- GPIO** PIN DE ENTRADA/SALIDA +3.3 V (GPIO *General Purpose Input/Output*).
Entrada digital —. Entrada analógica ~. (Todas las salidas son digitales).

- ADC** PIN DE SALIDA ANALÓGICA (el rango es entre +0V y +1V dividido en 1023 intervalos).
- SPI** BUS SPI (*Serial Peripheral Interface*).
- HSPI** BUS HSPI (*Hardware Serial Peripheral Interface*).
- SDIO** PINES PARA INICIO DEL ESP8266 DESDE UNA TARJETA SD.
Para activar el modo SDIO el pin GPIO 15 debe estar en tensión cuando se enciende la placa.
- TX/RX** COMUNICACIÓN SERIE TX/RX.
Los pines GPIO01 y GPIO02 están conectados al puerto USB a través del convertor UART.

ANEXO 4: Documentación oficial de configuración de red WIFI en el ESP8266

En este apartado se mencionan todas las capacidades del NodeMCU para identificar una red Wifi como Punto de Acceso (AP), así como las limitaciones y la información adicional que debe considerarse al momento de configurar dicha red.

The screenshot shows the documentation for the `WiFi.softAP(ssid, password, channel, hidden, max_connection)` function. The page is titled "ESP8266 Arduino Core" and "latest". The left sidebar contains a navigation menu with categories like "CONTENIDO:", "Instalación", "Referencia", "Librerías", "Sistema de ficheros", "ESP8266WiFi", "Actualizaciones OTA", "PROGMEM", "Utilizando GDB para depurar", "Tarjetas", "Preguntas - FAQ", "Errores fatales - Causas", "Depuración", "Volcado de pila", and "Uso con Eclipse".

The main content area shows the function signature: `WiFi.softAP(ssid, password, channel, hidden, max_connection)`. Below it, the text states: "El primer parámetro de esta función es obligatorio, los otros cuatro son opcionales." and "El significado de todos los parámetros es el siguiente:".

The parameters are listed as follows:

- `ssid` - cadena de caracteres que contiene el SSID de la red (máximo 31 caracteres)
- `password` - cadena de caracteres opcional con una contraseña. Para la red WPA2-PSK, debe tener al menos 8 caracteres de longitud (max. 63 caracteres). Si no se especifica, el punto de acceso estará abierto para que cualquiera pueda conectarse.
- `channel` - parámetro opcional para establecer el canal de WiFi, del 1 al 13. Canal predeterminado = 1.
- `hidden` - parámetro opcional, si se establece en `true` se ocultará el SSID
- `max_connection` - parámetro opcional para establecer el número máximo de estaciones conectadas simultáneamente, de 0 a 8. Por defecto 4. Una vez que el número máximo se ha alcanzado, cualquier otra estación que quiera conectarse se verá forzada a esperar hasta que alguna estación conectada se desconecte.

The text continues: "La función devolverá `true` o `false` según el resultado de configurar el soft-AP."

Notas:

- La red establecida por softAP tendrá una dirección IP predeterminada de 192.168.4.1. Esta dirección puede cambiarse usando `softAPConfig` (ver a continuación).
- A pesar de que ESP8266 puede operar en modo soft-AP + station, en realidad solo tiene un canal de hardware. Por lo tanto, en el modo soft-AP + station, el canal soft-AP se predeterminará al número utilizado por la estación. Para obtener más información sobre cómo puede esto afectar al funcionamiento de las estaciones conectadas al soft-AP de ESP8266, consulte [esta entrada de preguntas frecuentes](#) en el foro de Espressif.

ANEXO 5: Características físicas y comparativa entre ESP8266 y ESP32

Característica	ESP8266	ESP32
Procesador	Tensilica LX106 32 bit a 80 MHz (hasta 160 MHz)	Tensilica Xtensa LX6 32 bit Dual-Core a 160 MHz (hasta 240 MHz)
Memoria RAM	80 kB (40 kB disponibles)	520 kB
Memoria Flash	Hasta 4 MB	Hasta 16 MB
ROM	No	448 kB
Alimentación	3.0 a 3.6 V	2.2 a 3.6 V
Rango de temperaturas	-40°C a 125°C	-40°C a 125°C
Consumo de corriente	80 mA (promedio). 225 mA máximo	80 mA (promedio). 225 mA máximo
Consumo en modo sueño profundo	20 uA (RTC + memoria RTC)	2.5 uA (10 uA RTC + memoria RTC)
Coprocesador de bajo consumo	No	Sí. Consumo inferior a 150 uA
WiFi	802.11 b/g/n (hasta +20 dBm) WEP, WPA	802.11 b/g/n (hasta +20 dBm) WEP, WPA
Soft-AP	Sí	Sí

Bluetooth	No	v4.2 BR/EDR y BLE
UART	2* (En una de ellas solo puede utilizarse el pin Tx)	3
I2C	1	2
SPI	2	4
GPIO (utilizables)	32	11
PWM	8	16
ADC	1 (10 bit)	18 (12 bit)
ADC con preamplificador	No	Sí (Bajo ruido) Hasta 60 dB
DAC	No	2 (8 bit)
1-Wire	Implementado por software	Implementado por software
I2S	1	2
CAN bus	No	1 x 2.0
Ethernet	No	10/100 Mbps MAC
Sensor de temperatura	No	Sí
Sensor efecto HALL	No	Sí
IR	Sí	Sí
Temporizadores	3	4 (64 bits)
Encriptación por hardware	No (TLS 1.2 por software)	Sí (AES, SHA, RSA, ECC)
Gen. de núm. aleatorios	No	Sí
Encriptación de la flash	No	Sí
Arranque seguro	No	Sí

ANEXO 6: Estructura básica de un documento HTML

La forma mostrada en el presente anexo es solo representativa; existen muchas más formas de configurar el documento HTML, como cambiar el idioma de entrada o el código de caracteres implementado, entre otras opciones.

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Fisgon Parking</title>
7      <link rel="stylesheet" type="text/css" href="styles.css">
8
9
10 </head>
11 <body>
12
13 </body>
14
15 <script src="Directorio donde esta el .js y el .html"></script>
16
17 </html>
```


ANEXO 7: Página oficial de la documentación guía del ESP8266

En la documentación oficial se encuentra todo lo referente a las posibilidades de código que se pueden programar dentro del ESP8266 con diversas funcionalidades Wifi. Esto incluye el funcionamiento de diversas tarjetas disponibles dentro del paquete de gestor de tarjetas en el Arduino IDE, así como posibles errores que puedan presentarse y sus respectivas soluciones.

ESP8266 Arduino Core
latest

Search docs

CONTENIDO:

- Instalación
 - Gestor de Tarjetas
 - Usando la versión git
- Referencia
- Librerías
- Sistema de ficheros
- ESP8266WiFi
- Actualizaciones OTA
- PROGMEM
- Utilizando GDB para depurar
- Tarjetas
- Preguntas - FAQ
- Errores fatales - Causas
- Depuración
- Volcado de pila
- Uso con Eclipse

Read the Docs v: latest

Gestor de Tarjetas

Este es el método elegido para los usuarios finales.

Prerequisitos

- Arduino 1.6.8, descárgalo de la página [web de Arduino](#) :
- Conexión a internet

Instrucciones

- Inicia Arduino y abre la ventana Preferencias.
- Introduce `http://arduino.esp8266.com/stable/package_esp8266com_index.json` en la casilla *Gestor de URLs Adicionales de Tarjetas*. Puedes añadir múltiples URLs separándolas con comas.
- Abre en gestor de tarjetas desde Herramientas > «Última tarjeta seleccionada» > Gestor de tarjetas y busca la plataforma *esp8266*.
- Seleccione la versión que desee de la lista.
- Click en el botón *Instalar*.
- No olvides seleccionar tu tarjeta ESP8266 desde Herramientas > Menú de tarjetas tras la instalación.

Opcionalmente puedes usar el paquete *staging* desde el siguiente link:
http://arduino.esp8266.com/staging/package_esp8266com_index.json . Este contiene nuevas características, pero a la misma vez algo puede no funcionar.

Para mas información sobre el Gestor de Tarjetas, ver:

- <https://www.arduino.cc/es/guide/cores>

ANEXO 8: Página web oficial del portal de GitHub para la librería Wifi Manager.

The screenshot shows the GitHub repository page for 'tzapu/WifiManager'. The repository is public and has 1,298 commits, 19k forks, and 6.4k stars. The repository description is 'ESP8266 WiFi Connection manager with web captive portal'. The repository is linked to 'arduino' and 'esp8266' ecosystems, and is categorized as 'captive' and 'wifi manager configuration-portal'. The repository includes a README, MIT license, activity, 6.4k stars, 232 watchers, and 19k forks. The repository is a report repository and has 6 releases, with the latest being 'v2.1.17' from 3 months ago.

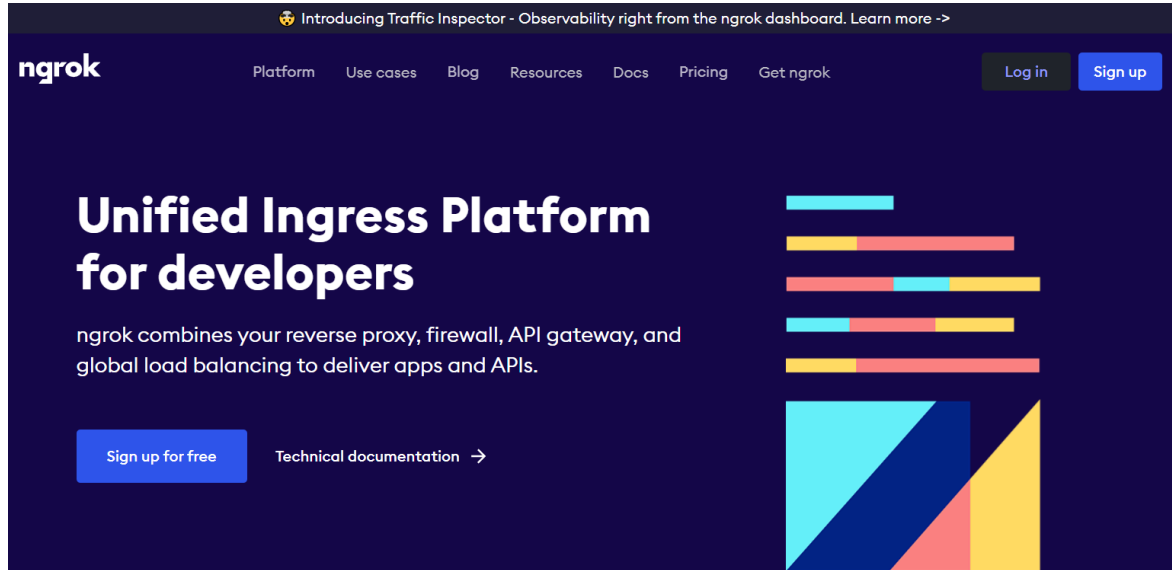
The repository structure is as follows:

- tablatronix Update compile_examples.yaml ✓ (e978bc0 · 3 months ago) 1,298 Commits
- .github 3 months ago
- examples 3 months ago
- extras last year
- travis 6 years ago
- .gitignore 2 years ago
- .travis.yml 3 years ago
- CMakeLists.txt (#1624) last year
- LICENSE 9 years ago
- README.md 5 months ago
- WifiManager.cpp Fix Debug Log format (#1722) 3 months ago

The repository also includes a 'Go to file' search bar and a 'Code' button. The repository is linked to 'arduino' and 'esp8266' ecosystems, and is categorized as 'captive' and 'wifi manager configuration-portal'. The repository includes a README, MIT license, activity, 6.4k stars, 232 watchers, and 19k forks. The repository is a report repository and has 6 releases, with the latest being 'v2.1.17' from 3 months ago.

ANEXO 9: Sitio Web oficial de NGROK

En esta página se puede acceder a un servicio de terceros para el alojamiento de sitios web desde Internet, que ofrece planes gratuitos de prueba y planes pagos con mejores funciones y mayor capacidad de tráfico en la red.



The image shows the homepage of the ngrok website. At the top, there is a navigation bar with the ngrok logo on the left and links for Platform, Use cases, Blog, Resources, Docs, Pricing, and Get ngrok in the center. On the right side of the navigation bar, there are buttons for Log in and Sign up. Below the navigation bar, the main heading reads "Unified Ingress Platform for developers". Underneath this heading, a sub-heading states "ngrok combines your reverse proxy, firewall, API gateway, and global load balancing to deliver apps and APIs." To the right of the text, there is a decorative graphic consisting of several horizontal bars of varying lengths and colors (cyan, yellow, red, blue) and a larger graphic below it composed of overlapping triangles in cyan, dark blue, red, and yellow. At the bottom left of the main content area, there is a button labeled "Sign up for free" and a link labeled "Technical documentation" with a right-pointing arrow.

ANEXO 10: Simbología electrónica de algunos componentes



Resistencia eléctrica
(norma americana)



Resistencia eléctrica
(norma europea)



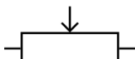
Inductancia propia
o bobina



Condensador



Potenciómetro
(symbole américain)



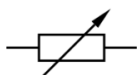
Potenciómetro
(symbole européen)



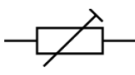
Condensador polarizado



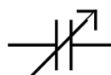
Condensador electrolítico



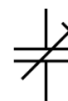
Resistencia variable
Reostato



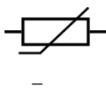
Resistencia ajustable
Trimmer



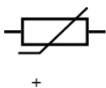
Condensador variable



Condensador ajustable
Trimmer



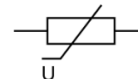
Termistancia
Termistancia NTC



Termistancia
Termistancia PTC



Fotorresistencia
LDR



Varistancia
VDR



Diodo



Diodo Zener



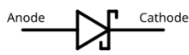
Diodo túnel



Diodo Varicap



Led
LED



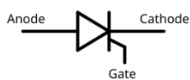
Diodo Schottky



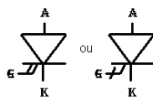
Fotodiodo



Diodo Transil



Tristor
SCR



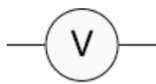
Tristor GTO



DIAC



Triac



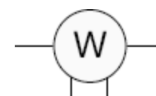
Voltímetro



Amperímetro



Óhmetro



Vatímetro



Cruce de hilos
no conectados



Cruce de hilos
conectados



Hilos conectados
Derivación



Masa



Protección Clase III
Muy baja tensión



Protección Clase II
Doble aislamiento



Protección Clase I
Puesta a tierra



Punto equipotencial
(tierra)



Puerta SI
Buffer



Puerta OR
OR



Puerta AND
AND



Puerta OR-exclusiva
XOR



Puerta NO
NOT



Puerta NO-O
NOR



Puerta NO-Y
NAND



Puerta NO-O exclusiva
XNOR

ANEXO 11: Tabla del proceso de instalación del proyecto en el Parqueadero del Instituto

La tabla siguiente presenta el proceso de instalación del prototipo del proyecto en el Instituto Superior Tecnológico Sudamericano. Se han considerado las observaciones detalladas en cada etapa, las cuales proporcionan información sobre el progreso alcanzado y los aspectos pendientes de completar.

Tabla de instalación y funcionamiento				
Fecha	Conexión física	Conexión local	Conexión Internet	Observaciones
5/8/2024	✓	X	X	El primer día de instalación se verificó el encendido con una red personal del autor pero no se logró conectar a la red del Instituto.
6/8/2024	✓	X	X	El segundo día se hicieron pruebas de conexión local sin éxito y la colocación del módulo prototipo en el puesto 3 del parqueadero.
7/8/2024	✓	✓	X	Se instaló el módulo en el puesto 2 del parqueadero y fuente de energía para alimentación y se estableció conexión con la red del Instituto, la conexión suele ser debil y presenta algunos retrasos de la información en la página.
13/8/2024	✓	✓	X	Selección de la PC adecuada.
14/8/2024	✓	✓	X	Instalación de Windows en la PC.
15/8/2024	✓	✓	X	Configuración de Windows para rendimiento.
16/8/2024	✓	✓	X	Instalación y configuración del software de Ngrok.
20/8/2024	✓	✓	✓	Se instaló todo el sistema en la infraestructura y se probó su funcionamiento, con la observación que la infraestructura tiene carencias para la instalación completa del sistema.