



UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

**Creación de un entorno de desarrollo integrado web de gestión de firmware
multilenguaje para un dispositivo IOT ESP32**

**VERA MACIAS OSCAR EDUARDO
INGENIERO EN TECNOLOGIAS DE LA INFORMACION**

**PALADINES CONDOY MIGUEL IGNACIO
INGENIERO EN TECNOLOGIAS DE LA INFORMACION**

**MACHALA
2023**



UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

**Creación de un entorno de desarrollo integrado web de gestión de
firmware multilenguaje para un dispositivo IOT ESP32**

**VERA MACIAS OSCAR EDUARDO
INGENIERO EN TECNOLOGIAS DE LA INFORMACION**

**PALADINES CONDOY MIGUEL IGNACIO
INGENIERO EN TECNOLOGIAS DE LA INFORMACION**

**MACHALA
2023**



UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

ENSAYOS O ARTÍCULOS ACADÉMICOS

**Creación de un entorno de desarrollo integrado web de gestión de
firmware multilenguaje para un dispositivo IOT ESP32**

**VERA MACIAS OSCAR EDUARDO
INGENIERO EN TECNOLOGIAS DE LA INFORMACION**

**PALADINES CONDOY MIGUEL IGNACIO
INGENIERO EN TECNOLOGIAS DE LA INFORMACION**

HERNANDEZ ROJAS DIXYS LEONARDO

**MACHALA
2023**

En São Paulo, Brasil, en 13 de Febrero del 2024

A quien pueda interesar,

En mi carácter de Director Editorial de la Revista Científica **Ibero-American Journal of Engineering & Technology Studies**, e-ISSN: 2764-6033, hago constancia que los investigadores:

Miguel Ignacio Paladines Condoy

Oscar Eduardo Vera Macias

Dixys Leonardo Hernández Rojas

Cuenta con la aprobación del Comité Científico Editorial mediante **evaluación ciega por pares**, al atender todos los lineamientos solicitados para la publicación de su artículo científico titulado:

Creacion de un IDE web de gestión de firmware multilenguaje para un dispositivo LoT ESPE32

El referido artículo será publicado en el **número 5 del volumen 4**, período **Enero-Julio del 2024**, y estará disponible en línea:

<https://tech.iberojournals.com/index.php/IBEROTECS>

Expido esta constancia a solicitud de los autores y permanezco a entera disposición para cualquier consulta o aclaratoria.

Atentamente,



PhD, Carlos Torres

Chief Editor – Sapienza Grupo Editorial
R. Santa Cruz, 2187 - Vila Mariana, São Paulo Brasil
editor@sapienzaeditorial.com

Verificación:



CLÁUSULA DE CESIÓN DE DERECHO DE PUBLICACIÓN EN EL REPOSITORIO DIGITAL INSTITUCIONAL

Los que suscriben, VERA MACIAS OSCAR EDUARDO y PALADINES CONDOY MIGUEL IGNACIO, en calidad de autores del siguiente trabajo escrito titulado Creación de un entorno de desarrollo integrado web de gestion de firmware multilenguaje para un dispositivo IOT ESP32, otorgan a la Universidad Técnica de Machala, de forma gratuita y no exclusiva, los derechos de reproducción, distribución y comunicación pública de la obra, que constituye un trabajo de autoría propia, sobre la cual tienen potestad para otorgar los derechos contenidos en esta licencia.

Los autores declaran que el contenido que se publicará es de carácter académico y se enmarca en las disposiciones definidas por la Universidad Técnica de Machala.

Se autoriza a transformar la obra, únicamente cuando sea necesario, y a realizar las adaptaciones pertinentes para permitir su preservación, distribución y publicación en el Repositorio Digital Institucional de la Universidad Técnica de Machala.

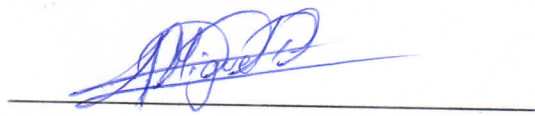
Los autores como garantes de la autoría de la obra y en relación a la misma, declaran que la universidad se encuentra libre de todo tipo de responsabilidad sobre el contenido de la obra y que asumen la responsabilidad frente a cualquier reclamo o demanda por parte de terceros de manera exclusiva.

Aceptando esta licencia, se cede a la Universidad Técnica de Machala el derecho exclusivo de archivar, reproducir, convertir, comunicar y/o distribuir la obra mundialmente en formato electrónico y digital a través de su Repositorio Digital Institucional, siempre y cuando no se lo haga para obtener beneficio económico.



VERA MACIAS OSCAR EDUARDO

0706287679



PALADINES CONDOY MIGUEL IGNACIO

0750907685

Creación de un IDE web de gestión de firmware multilinguaje para un dispositivo IOT ESP32

Miguel Ignacio Paladines Condoy¹
mpaladines5@utmachala.edu.ec
<https://orcid.org/0009-0007-8602-0668>
Universidad Técnica de Machala
El Oro, Ecuador

Dixys Leonardo Hernández Rojas¹
dhernandez@utmachala.edu.ec
<https://orcid.org/0000-0002-2116-6531>
Universidad Técnica de Machala
El Oro, Ecuador

Oscar Eduardo Vera Macias¹
overa2@utmachala.edu.ec
<https://orcid.org/0009-0008-6962-9102>
Universidad Técnica de Machala
El Oro, Ecuador

RESUMEN

La realización de este estudio tuvo como finalidad la flexibilidad al momento de programar microcontroladores embebidos utilizados en laboratorios remotos, existiendo la más común mediante un cable USB y programados en un único lenguaje, pero no es la manera más innovadora y eficiente que pueda existir. Por ello se creó un entorno de desarrollo integrado en la web, siendo accesible de manera local teniendo una interfaz intuitiva para que los usuarios puedan grabar el firmware al ESP32 en múltiples lenguajes (C++ y MicroPython), permitiendo que no solo varias personas interactúen de manera simultánea en diferentes instancias, sino que también puedan acceder desde distintos dispositivos que tengan acceso a internet (celular, computador de escritorio, laptops y tables). Además, el sistema implementa OTA (Over-The-Air) para la actualización remota de firmware. Esto significa que el código junto con la nueva versión será enviado y actualizado en el repositorio de GitHub Page dentro de la carpeta del usuario asignado. Para la evaluación del IDE se aplicó la norma ISO 25010, donde los aspectos a evaluar nos dieron los siguientes porcentajes: Funcionalidad 76.50%, Usabilidad 66.50%, Eficiencia 60.50%, Seguridad 78.50%, Mantenibilidad 67% y Portabilidad con un 65%.

Palabras claves: IoT; ESP32; IDE; multilinguaje; firmware.

Creating a multi-language firmware management web IDE for ESP32 IoT devices

ABSTRACT: The purpose of this study was to provide flexibility when programming embedded microcontrollers used in remote laboratories, the most common being through a USB cable and programmed in a single language, but it is not the most innovative and efficient way that could exist. For this reason, an integrated development environment was created on the web, being accessible locally, having an intuitive interface so that users can record the firmware to the ESP32 in multiple languages (C++ and MicroPython), allowing not only several people to interact simultaneously in different instances, but also that they can access from different devices that have Internet access (cell phone, desktop computer, laptops and tables). Additionally, the system implements OTA (Over-The-Air) for remote firmware update. This means that the code along with the new version will be pushed and updated in the GitHub Page repository within the assigned user's folder. The IDE was evaluated using the ISO 25010 standard, yielding percentages for different aspects: Functionality 76.50%, Usability 66.50%, Efficiency 60.50%, Security 78.50%, Maintainability 67%, and Portability 65%.

Keywords: IoT; ESP32; IDE; multilanguage; firmware.

INTRODUCCIÓN

El progreso en aplicaciones IoT se ha vuelto esencial para investigadores gracias a la omnipresencia de Internet (Novillo-Vicuña & Rojas, 2018). Para mejorar la facilidad de uso, las tecnologías web han cobrado relevancia en este campo innovador (Satyendra K Vishwakarma & Prashant Upadhyaya, 2019). Ante ello se busca la automatización de procesos, facilitando así la vida del ser humano, la comunicación simultánea entre usuarios y dispositivos agilizando la eficiencia al momento de resolver una interrogante o mejorar un proceso del diario a vivir (Al-Dabass & Institute of Electrical and Electronics Engineers, 2014). Algunos datos en su forma original provienen de dispositivos en redes de sensores inalámbricos (WSN) y se emplean para tomar decisiones de control en relación con los actuadores (Mazon-Olivo & Hernández-Rojas, 2018).

Un objetivo específico de estas nuevas tecnologías se enfoca en conectar dispositivos y máquinas a través de Internet con una arquitectura de IoT (Joel A. Cujilema Paguay & Dixys L. Hernandez Rojas, 2023), permitiendo que se intercambien datos sobre su funcionamiento automáticamente, sin necesidad de intervención humana (Omar Otoniel Flores-Cortez & Verónica Idalia Rosa, 2018). Los microcontroladores cumplen un rol fundamental al momento de querer implementar las soluciones que mejorarían el aprendizaje de IoT. (Díaz Ronceros, 2020) Sostiene que el microcontrolador es un chip que contiene una estructura parecida a la de un ordenador.

En la actualidad existen una variedad de formas de usar un microcontrolador, ya sea en laboratorios remotos que posibilitan a los usuarios llevar a cabo experimentos prácticos en electrónica a través de Internet como lo indica (S. López & A. Carpeño, 2014). O al realizar pequeños proyectos, uno de los microcontroladores que más resaltan son los ESP32 siendo de los más usados por su estructura de dos núcleos, una extensa variedad de periféricos, y su habilidad para realizar múltiples funciones (Marek & Pavel, 2019).

La programación de estos microcontroladores (ESP32) es esencial al momento de querer innovar en el mundo de la tecnología (Joni Welman Simatupang & Aida Mahdalena Lubis, 2022), por las múltiples funcionalidades que brindan y dentro del ámbito de estos, dicho microcontrolador (ESP32) es el que más destaca o en su defecto siendo el preferido por muchos desarrolladores al momento de buscar componentes IoT para sus trabajos, proyectos y/o investigaciones (Husam Karim & Dmitry Dunaev, 2021).

En la forma de programar un ESP32 el lenguaje más común para hacerlo es el C++ como opina (Prem Prakash Murmu & Harshit Paul, 2019), en tal caso dentro del medio académico la mayoría de prácticas de laboratorios son realizadas con dicho principio por tener un bajo costo al momento de realizarlas (Cyprian N. Oton & M. Tariq Iqbal, 2021), pero sin explorar nuevas soluciones que en muchos casos pueden ser más eficientes o con un menor tiempo de respuesta a la información solicitada.

Siendo esta una de las principales razones por la cual se aborda el desarrollo de este Artículo, ya que dentro del ámbito de la “Programación de Microcontroladores” existen otros lenguajes para poder hacer lo mismo, tal es el caso de Micropython (Cerón-Quiceno & Rodríguez-Chong, 2023). Al ser una versión adaptada de Python 3, comparte numerosas características con el estándar del lenguaje Python como señala (Valeriu Manuel Ionescu & Florentina Magda Enescu, 2020).

Por ello se busca demostrar que lenguaje tiene un mejor tiempo de respuesta, mejor latencia y facilidad al momento de grabar el firmware a un ESP32, en palabras de (Fengpei Yuan & Xiaolei Wang, 2019) el firmware es una categoría especializada de software diseñada para gestionar las operaciones de bajo nivel del hardware específico en un dispositivo (Pekez & Kaprocki, 2018). Mientras que el software de aplicación entrega funciones directas a los usuarios, el firmware sirve como un enlace entre el hardware del (Armijos-Reyes & Rojas, 2021)

Además de la comparación de los dos lenguajes existe la pregunta, ¿Cómo se podrá grabar dos diferentes lenguajes de programación a un ESP32?, por ende se implementó el desarrollo de un

IDE que permita la grabación de firmware a través de una red local mediante el protocolo Over-the-Air (OTA) (Sharf & Elhamied, 2021), ya que ha aportado muchos beneficios en lo que a nuevas tecnologías se refiere (WenXuan Wang & Guihe Qin, 2022).

Otra pregunta para abordar es, ¿Cómo se podrán conectar múltiples usuarios al IDE de manera simultánea?, para dar una pequeña idea previa se usará un repositorio, que es GitHub Page que ofrece la característica de solicitud de extracción, la cual habilita a los desarrolladores para duplicar un proyecto (Zhou & Sun, 2021), aplicar modificaciones al código y pedir a los dueños del proyecto que examinen e incorporen estas alteraciones del código (El Mezouar, 2019), de tal manera teniendo un IDE multifuncional para el usuario (Rojas & Villegas, 2015).

Dentro de este IDE diseñado en Visual Studio Code, hay una extensión importante para el desarrollo de la grabación hacia el ESP32 (Paul Schroeder, 2021), pero con lenguaje C++, la extensión en este caso es PlatformIO que permite unir las propuestas de tecnología física de los productores en un entorno creativo y eficiente para generar mayor innovación y rendimiento, para lenguaje C++ basado en Arduino (Miladinović, 2020).

Node.js permite la ejecución asíncrona de JavaScript, facilitando aplicaciones de red escalable (Rappl, 2023). A diferencia de modelos con hilos, gestiona múltiples conexiones sin bloqueo, y admite uso eficiente de múltiples núcleos. El FrontEnd fue diseñado en React ya que una de las bibliotecas más ampliamente utilizadas en el mundo de JavaScript para crear aplicaciones tanto móviles como web (Dabit, 2019). En conjunto con el lenguaje TypeScript debido a que proporciona respuestas a una variedad de desafíos en el ámbito de JavaScript (Da Costa, 2021).

Para la base de datos se usó la “Firestore DataBase” que es de FireBase, y por esta misma razón para la creación del Login se usó FireBase RealTime Database (Joni Welman Simatupang & Aida Mahdalena Lubis, 2022), toda la información de los usuarios se irá a la base de datos FireBase (Lakshmi Goswami, 2020) y para el desarrollo de lenguaje MicroPython se usó otra extensión de Visual Studio Code, en este caso es Pymark su propósito es asistir a los desarrolladores para comprender y contrastar la eficacia del código (Scott, 2023).

Por último punto al tratar el plan de evaluación se realizará basándose en la norma ISO-25010, que en palabras de (Rocha, 2020) es un estándar global que establece y clasifica los aspectos de calidad de los sistemas y productos de software. Forma parte de la serie ISO/IEC 25000, conocida también como SQuaRE (Requisitos y Evaluación de la Calidad del Producto de Software). Dicho estándar identifica ocho atributos clave de calidad en el software (Díaz-Muñoz, 2020).

METODOLOGÍA

Con este estudio se busca cumplir la hipótesis la cual consiste en, la implementación de un sistema de grabación de firmware multilenguaje, gestionado por un IDE web y que pueda ser accesible de manera local, mejorando significativamente la eficiencia y flexibilidad en el proceso de programación de microprocesadores y dispositivos electrónicos.

Se optó por una técnica de muestreo por convivencia, la cual, aunque no se adhiere a los principios del muestreo probabilístico o aleatorio, se fundamenta en la selección de sujetos basada en su accesibilidad y disponibilidad en un momento dado. En este caso, se seleccionaron estudiantes por cada nivel semestral mencionado, conformando así una muestra total de treinta individuos.

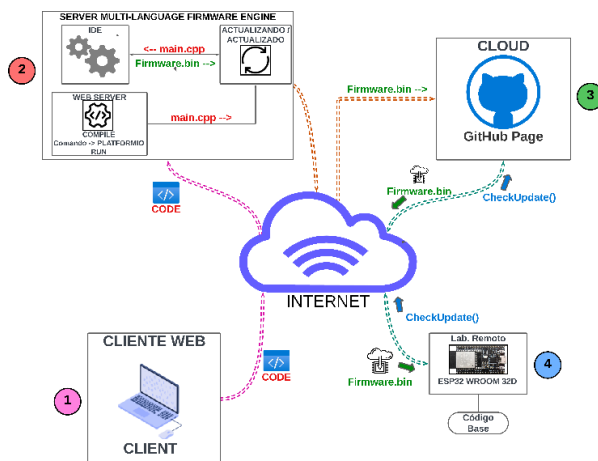
En el desarrollo de este IDE se implementó el uso del protocolo OTA (Over The Air) como se indica en la Figura 1, siendo el aporte principal en este estudio, para entender de mejor manera esto se lo separo en 4 secciones.

1. **Cliente Web:** El cliente o usuario accede al IDE ya desarrollado, selecciona el lenguaje donde quiere trabajar (C++ y MicroPython), y procede a tipiar el código dentro del área de trabajo para poder cargar, el código pasará vía internet hacia el servidor donde se procesará la solicitud con el código enviado
2. **Server Multi-Lenguage Firmware Engine:** Aquí se receipta el código y se procese a compilar dependiendo del lenguaje que se allá escogido si es un código en lenguaje C++ primero se reemplazara con el archivo main.cpp del usuario pertinente y se compilara con el comando Platformio Run, y si fuera del otro lenguaje seleccionado se reemplazara el

archivo main.py, para que pueda ser grabado el firmware si es la primera vez que se hace este proceso y/o actualizar si previamente ya se ha grabado.

3. **Cloud:** El firmware.bin accede a GitHub Page, donde se encuentra las carpetas y archivos para cada lenguaje, funciona como el repositorio del IDE.
4. **ESP32:** El esp32 estará ejecutando la actualización OTA cada 30 segundos, como último paso cuando se accede por primera vez al esp32 el firmware lo hace es grabar el código en el mismo, y se introduce una versión que luego servirá para ser comparada en este caso Versión 1.0, si el esp32 ya ha sido grabado previamente lo que procede a realizar es que compara las versiones en el repositorio, si las versiones son las mismas no sucede nada, si existe una nueva versión (Versión 1.1), se borra la anterior versión (Versión 1.0) y procede a grabarse la nueva versión.

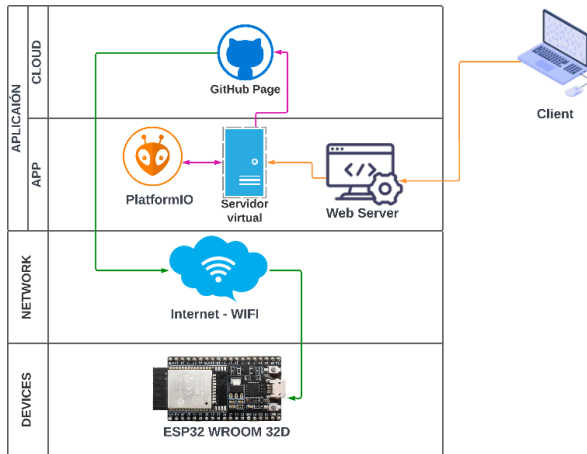
Figura 1: Mecanismos de actualización del firmware en el repositorio Github por parte del cliente y de forma automática vía OTA en el ESP32



Fuente 1: Elaboración propia (2024).

Se adopta la arquitectura de tres capas: Capa de dispositivo, Capa de Red y Capa de Aplicación (Chuquimarca & Maita, 2022). Esta arquitectura fue la base para el desarrollo, tal y como se representa a continuación en la Figura 2.

Figura 2: Esquema general del sistema propuesto en función de una Arquitectura IoT.

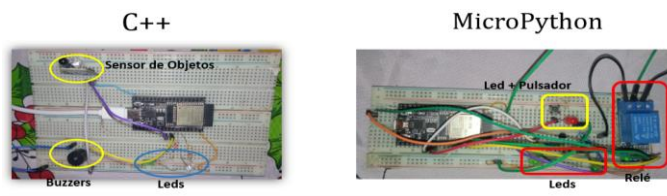


Fuente 2: Elaboración propia (2024).

Capa de Dispositivo

El dispositivo fundamental que se implementó es el ESP32 acompañado de otros dispositivos IOT, todo esto formando un circuito completo como se muestra en la Figura 3, donde se detalla cada uno de ellos. Los prototipos son conectados con el ESP32, que cumplen determinadas funciones según el usuario lo configure.

Figura 3: Prototipos usados en la capa de dispositivo para las pruebas de cada lenguaje.



Fuente 3: Elaboración propia (2024).

Capa de Red

La capa de red es la encargada de establecer y gestionar la conectividad entre los dispositivos IoT y los servidores asegurando que sus datos lleguen a sus destinos sin ningún problema.

Capa de Aplicación

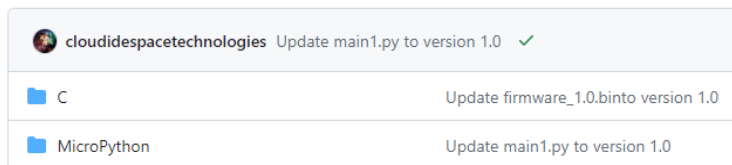
La capa de aplicación en la arquitectura de tres capas de IoT se divide en dos partes; la Cloud (GitHub Page) y la App.

GitHub Page

En GitHub Page se encuentran las carpetas de los lenguajes C y MicroPython con se visualiza en la Figura 4, cada una tendrá una denominación de "Usuario" con el número que le corresponda según haya ingresado al IDE tal y como lo representa la Figura 5.

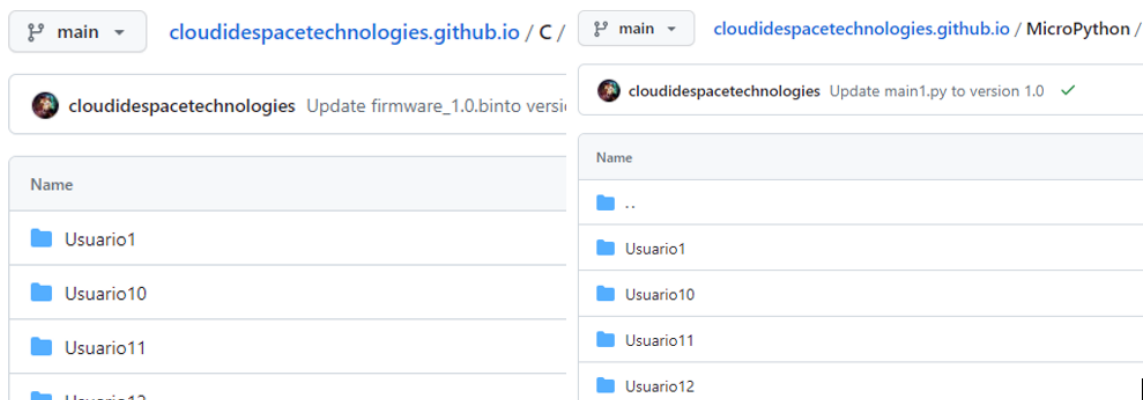
Cada nuevo usuario al iniciar sesión tiene su correo electrónico y carpeta de GitHub almacenados en Firestore. El último parámetro se asigna como "Usuario1". Si otro usuario de la misma red inicia sesión, se registran los mismos datos con el último parámetro incrementado (por ejemplo, "Usuario2"). Tras iniciar sesión, si la carpeta GitHub coincide con una carpeta en el repositorio específico, se muestra automáticamente al usuario. Si la carpeta no existe, ningún usuario se muestra en el sistema web IDE, explicando, así como funciona la representación de la Figura 6.

Figura 4: Carpetas de los lenguajes de programación del firmware dentro del repositorio GitHub.



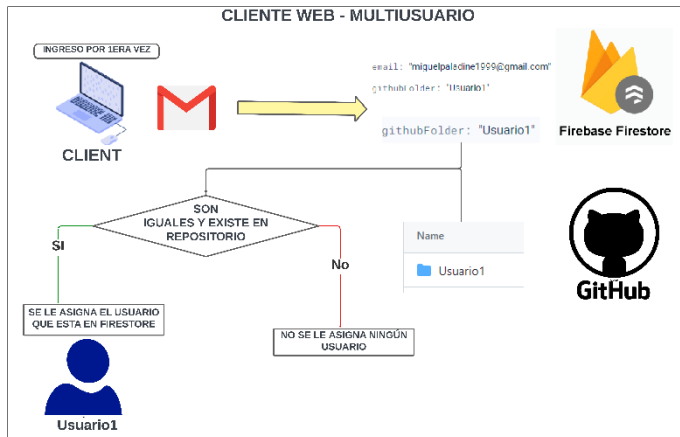
Fuente 4: Elaboración propia (2024).

Figura 5: Carpetas asignadas a cada Usuario (Cliente) dentro del repositorio GitHub.



Fuente 5: Elaboración propia (2024).

Figura 6: Flujo de ingreso (login) de un cliente web en el sistema propuesto.

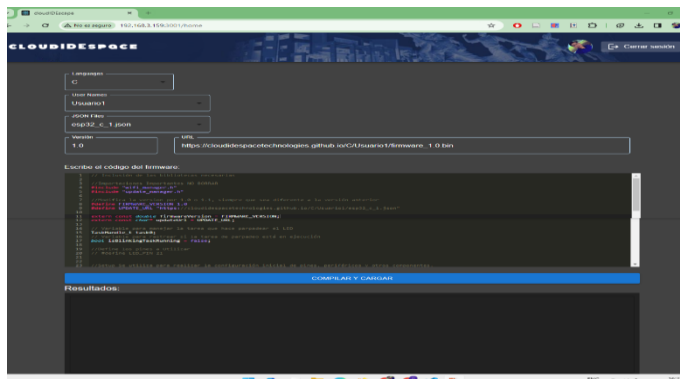


Fuente 6: Elaboración propia (2024).

App

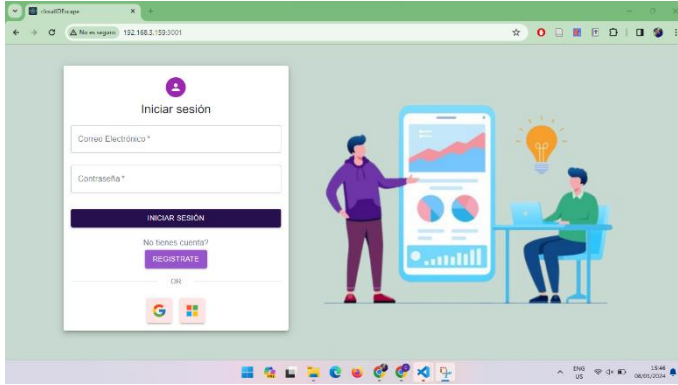
El sistema se basa en un IDE para programar el ESP32 en lenguajes como C y Micropython. Utiliza Node.js para el backend, permitiendo la ejecución asincrónica y la gestión eficiente de múltiples conexiones. El frontend se construye en React con TypeScript para aplicaciones sólidas y avanzadas como se representa en la Figura 7. La base de datos Firebase maneja el login Figura 8, asignando automáticamente tokens a los usuarios. GitHub se utiliza para crear carpetas específicas para cada usuario. PlatformIO facilita la programación del ESP32, mientras que Pymark divide el proceso para el funcionamiento de MicroPython, gestionando conexiones, código editable y actualizaciones del firmware. En resumen, el sistema integra tecnologías para permitir la programación y actualización eficiente del firmware del ESP32 desde un IDE accesible desde múltiples dispositivos como se visualiza en la Figura 9.

Figura 7: Página de inicio del IDE Web creado.



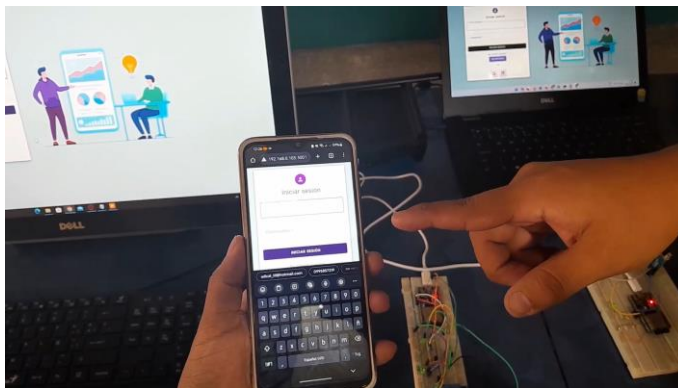
Fuente 7: Elaboración propia (2024).

Figura 8: Front-end del Login del Sistema propuesto.



Fuente 8: Elaboración propia (2024).

Figura 9: Ingreso al IDE Web desde varios dispositivos a la vez (Computadora, laptop y smartphone).



Fuente 9: Elaboración propia (2024).

RESULTADO Y DISCUSIONES

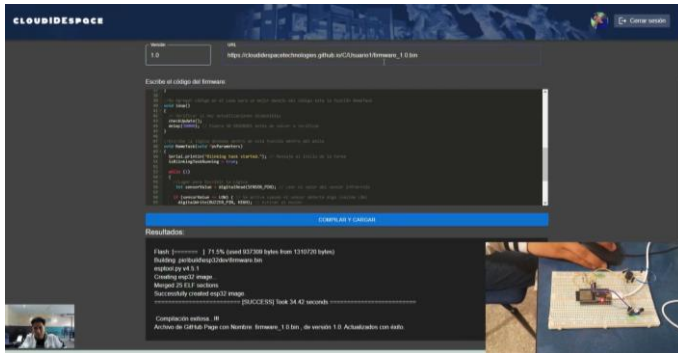
Se implemento la información de la Tabla 1, para poder evaluar el sistema de acuerdo con la Norma ISO-25010 y así obtener resultados de manera porcentual con respecto a las encuestas realizadas. Donde se probó el sistema en diferentes lenguajes (C++ como se visualiza en la Figura 10 y MicroPython como se aprecia en la Figura 11).

Tabla 1: Tabla de porcentajes de evaluación.

Muy aceptable	Aceptable	Poco aceptable	Nada aceptable
100% - 75%	74% - 51%	26% - 49%	0% - 25%

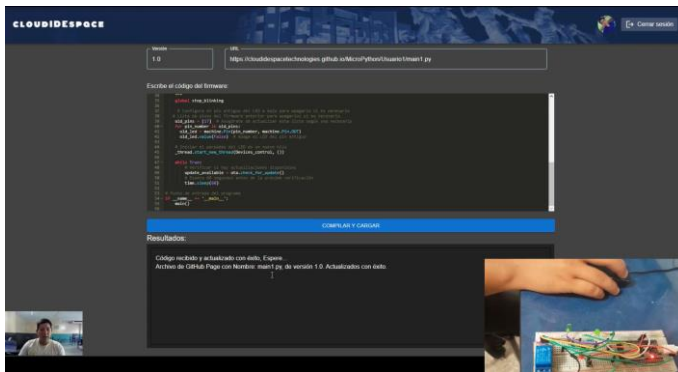
Fuente 10: Elaboración propia (2024).

Figura 10: Interfaz del IDE utilizando el lenguaje C++.



Fuente 11: Elaboración propia (2024).

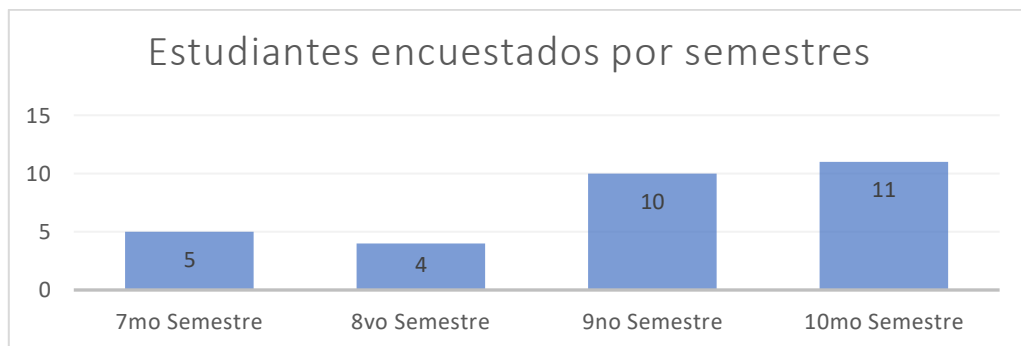
Figura 11: Interfaz del IDE utilizando el lenguaje MicroPython.



Fuente 12: Elaboración propia (2024).

A continuación, se representa una representación gráfica de los estudiantes que han sido encuestados como se muestra en la Figura 12.

Figura 12: Distribución de los estudiantes encuestados por cada semestre de la carrera de TI de la Utmach.



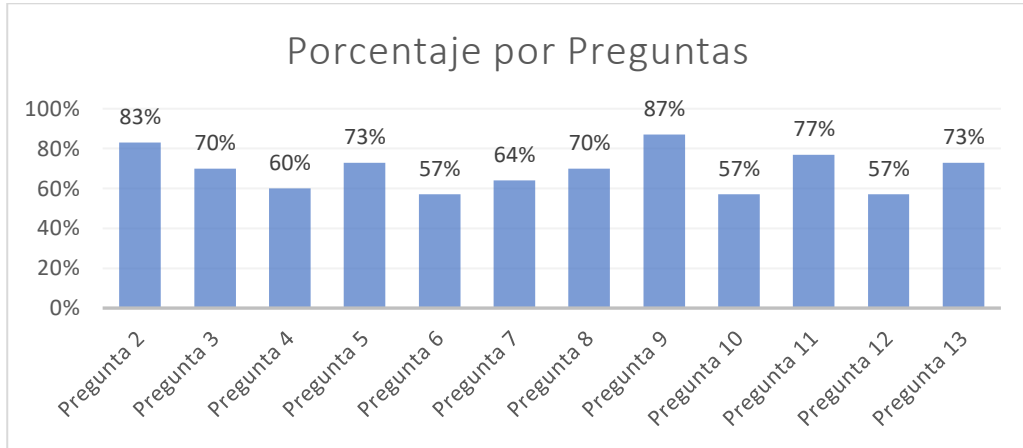
Fuente 13: Elaboración propia (2024).

Nota: Se encuestaron estudiantes de tecnologías de la información para evaluar nuestro IDE. Se seleccionó esta población por su conocimiento en el tema. De los 30 encuestados, el 37% son del décimo semestre, el 33% del noveno, el 17% del séptimo y el 13% del octavo. Estos datos serán

clave para evaluar la funcionalidad, usabilidad, eficiencia, mantenibilidad y portabilidad de nuestro IDE web para gestionar firmware multilenguaje en un dispositivo IoT ESP32.

Luego de realizar la encuesta y ponderar los resultados se obtuvieron los siguientes porcentajes por preguntas, como se aprecia en la Figura 13.

Figura 13: Resultados obtenidos por cada pregunta al momento de realizar la evaluación.



Fuente 14: Elaboración propia (2024).

Con las preguntas realizadas se utilizó dos por cada aspecto a evaluar de la norma, lo que dio un porcentaje ponderado y con ello obteniendo un resultado por grado de aceptación tal y como se visualiza en la Tabla 2.

Tabla 2: Evaluación del sistema con la Norma ISO 25010.

Norma ISO 25010			
Métricas	Preguntas correspondientes.	Porcentaje ponderado.	Grado de aceptación obtenido.
Funcionalidad	2. ¿El sistema permite grabar firmware en los lenguajes C++ y Micropython de manera efectiva? 3. ¿La programación inalámbrica (OTA) funciona correctamente?	76,50%	Muy Aceptable
Usabilidad	4. ¿Cómo calificarías la facilidad de uso de la interfaz web? 5. ¿El proceso para seleccionar lenguajes, escribir código y compilar/cargar es intuitivo?	66,50%	Aceptable
Eficiencia	6. ¿Cuál es la velocidad de compilación y carga del firmware?	60,50%	Aceptable

	7. ¿Cómo evalúas la gestión de recursos del sistema durante operaciones intensivas?		
Seguridad	8. ¿Te sientes seguro(a) al usar el sistema durante el login y la manipulación de firmware? 9. ¿El sistema parece implementar prácticas seguras en la gestión de archivos y comunicación?	78,50%	Muy Aceptable
Mantenibilidad	10. ¿Consideras que sería fácil realizar cambios o actualizaciones en el sistema? 11. ¿El código te parece bien documentado y estructurado?	67%	Aceptable
Portabilidad	12. ¿El sistema funciona bien en diferentes navegadores y dispositivos? 13. ¿Cómo calificarías la compatibilidad del sistema con distintas versiones de dispositivos IoT (Microprocesadores - ESP32)?	65%	Aceptable

Fuente 15: Elaboración propia (2024).

En términos de funcionalidad, la calificación del 76,5% señala una robusta capacidad del entorno de desarrollo integrado web para la gestión de firmware en múltiples lenguajes. Esto indica una implementación efectiva que permite abordar la heterogeneidad de sistemas presentes en el entorno del Internet de las Cosas.

Discusión: De acuerdo a mi trabajo realizado el 65% corresponde a la parte de funcionalidad dicha por (Valeriu Manuel Ionescu & Florentina Magda Enescu, 2020) referente a la programación a un microcontrolador con OTA en los lenguajes C++ y MicroPython. Con respecto a ello hemos obtenido un 76,50%, lo que lo hace un trabajo similar al nuestro, con la diferencia de que nuestro sistema permite grabar ambos lenguajes de manera más eficiente.

En cuanto a la usabilidad, a pesar de alcanzar un puntaje del 66,5%, se identifican áreas de mejora. Aspectos como interfaces más intuitivas y un flujo de trabajo más eficiente son esenciales, especialmente para usuarios con menos experiencia en la programación de firmware.

Discusión: En lo indagado con respecto a trabajos relacionados en lo que a Usabilidad de un sistema se refiere, un artículo realizado por (Maier, 2017) obtuvo un 62% en la programación de

lenguajes, escribir código y compilar/cargar de manera intuitiva. En mi trabajo con la información recaba obtuve un porcentaje de 66.5%, haciendo una pequeña diferencia, pero siendo un trabajo similar donde se aplica un estudio comparativo entre lenguajes.

La eficiencia, reflejada en una calificación del 60,50%, indica la necesidad de optimizar el sistema. Estrategias para la gestión óptima de recursos, reducción de tiempos de carga y mejora del rendimiento global serían fundamentales para una experiencia de usuario más fluida.

Discusión: La eficiencia del sistema diseñado nos refleja un 60,5% de porcentaje, ya que tiene un tiempo de carga y compilación moderadamente rápido, pero tiene un pequeño declive al momento de ser sometido a operaciones intensivas, si lo comparamos con trabajos como el de (González Álvarez, 2021) el cual tiene un 65%, siendo un porcentaje más elevado al de nuestro sistema, por el simple hecho de apoyarse de otros dispositivos IoT que lo hacen más estable al momento de ser sometido a tareas más intensas.

En términos de mantenibilidad, la calificación de 67% sugiere la necesidad de esfuerzos adicionales. Se requieren estrategias efectivas para la gestión del código, documentación detallada y herramientas para facilitar actualizaciones y correcciones futuras.

Discusión: En la Mantenibilidad se ha obtenido un porcentaje promedio de 67% donde se refleja que hacer cambios al sistema, sería un proceso difícil de realizar, comparado con otros trabajos en los que nos hemos basado, en los que compartimos el mismo inconveniente (Joni Welman Simatupang & Aida Mahdalena Lubis, 2022), pero nuestro trabajo se diferencia al ya mencionado por el simple hecho de nos manejamos con una correcta documentación del código que puede a futuro facilitar un poco el proceso de realizar cambios o mejoras.

La portabilidad, con una puntuación de 65%, destaca la capacidad del sistema para adaptarse a diversos entornos, permitiendo a los usuarios trabajar en diferentes dispositivos y plataformas, lo que amplía su versatilidad.

Discusión: El trabajo realizado por (Checa Agurto, 2022) nos da a conocer como su IDE es compatible con el navegador web de Google, en el cual se puede acceder a todo el front-end de

su sistema obteniendo un porcentaje de Portabilidad de 60% al no ser compatible con la amplia gama de navegadores existentes en la red. En cambio, las encuestas realizadas a los estudiantes no permitieron probar nuestro sistema en varios navegadores web, los más usados en la actualidad obteniendo un porcentaje de Portabilidad del 65%. Reflejando una gran diferencia a otros trabajos que contienen similitudes al nuestro.

Los datos de la encuesta demuestran un fuerte respaldo a los beneficios hipotetizados del sistema IDE web para la gestión de firmware multilenguaje en la programación de microprocesadores y dispositivos electrónicos. Con la mayoría de los participantes indicando una mejora en eficiencia y flexibilidad, se considera que la hipótesis ha sido corroborada.

CONCLUSIONES

Este estudio ha culminado con éxito la implementación de un IDE web de firmware multilenguaje para dispositivos IoT ESP32, cumpliendo con el propósito de mejorar la flexibilidad en la programación de microcontroladores embebidos en laboratorios remotos. El sistema, accesible localmente y con una interfaz intuitiva, ha logrado superar las limitaciones de los métodos tradicionales al permitir la grabación de firmware en diversos lenguajes (C++ y MicroPython) mediante programación inalámbrica (OTA). Además, destaca por la posibilidad de acceso simultáneo de múltiples usuarios desde distintos dispositivos. La aplicación de la norma ISO 25010 en la evaluación evidencia un desempeño sólido en funcionalidad, seguridad y eficiencia, aunque se identificaron áreas para mejorar en usabilidad y mantenibilidad. Este trabajo científico contribuye significativamente al campo al abordar eficazmente la problemática de la programación de microcontroladores, ofreciendo una solución integral que mejora la eficiencia de recursos en entornos educativos. Su aporte radica en la capacidad de grabar ESP32 con OTA en varios lenguajes y permitir acceso simultáneo, destacando así su relevancia para la comunidad académica y tecnológica.

FUTUROS DESARROLLOS

Se considera adaptar el IDE desarrollado para que pueda ser implementado en un servidor web, lo que posibilitará un acceso remoto y una mayor flexibilidad en su utilización en laboratorios a distancia. Esta evolución no solo ampliará su cobertura, sino que también brindará una solución más adaptable y escalable, simplificando su inserción en entornos educativos e investigativos. Así mismo, se investigará la viabilidad de agregar características adicionales, como la gestión centralizada de varios dispositivos y la integración con sistemas de supervisión y control, con el fin de ofrecer una herramienta aún más completa y adecuada a las necesidades en constante evolución de la comunidad académica y tecnológica.

REFERENCIAS BIBLIOGRAFICAS

- Song, J., Sierra, S. C., Rodriguez, J. C., Perandonos, J. M., Jimenez, G. D. C., Buján, J. O., ... & Galdón, A. S. (2014, November). Parameter-based mechanism for unifying user interaction, applications and communication protocols. In 2014 2nd International Conference on Artificial Intelligence, Modelling and Simulation (pp. 131-136). IEEE.
- Cerón-Quiceno, M., & Rodríguez-Chong, J. (2023). Design and Construction of a Prototype Radar Using Raspberry Pico and MicroPython. *2023 IEEE Central America and Panama Student Conference (CONESCAPAN)*, 13-18.
<https://doi.org/10.1109/CONESCAPAN60431.2023.10328434>
- Checa Agurto, L. A. (2022). *Sistema IoT con módulos ESP-32 para monitoreo de contenedores móviles refrigerados usando protocolo MQTT, plataforma IO Adafruit e IFTTT*.
- Chuquimarca, C. E. T., & Maita, S. S. (2022). Análisis comparativo entre arquitecturas de sistemas IoT. *Revista de Investigación en Tecnologías de la Información: RITI*, 10(21), 55-70.
- Cyprian N. Oton & M. Tariq Iqbal. (2021). *Low-Cost Open Source IoT-Based SCADA System for a BTS Site Using ESP32 and Arduino IoT Cloud*. The e Institute of Electrical and Electronics Engineers, Inc.

- Díaz Ronceros, E. (2020). Relevancia de la ejecución experimental de proyectos con microcontroladores en el aprendizaje de la ingeniería electrónica. *Educación*, 29(56), 48-72.
- Díaz-Muñoz, G. (2020). Metodología del estudio piloto. *Revista chilena de radiología*, 26(3), 100-104.
- El Mezouar, M. (2019). An empirical study on the teams structures in social coding using GitHub projects. *Empirical Software Engineering*, 24, 3790-3823.
- Fengpei Yuan & Xiaolei Wang. (2019). *The Method of Embedded Device Firmware Update Under Multi-layer Heterogeneous Network*. Conference Publishing Services, IEEE Computer Society.
- González Álvarez, J. (2021). *Posicionamiento de una cabina de ascensor mediante un ESP32 y sensores*.
- Husam Karim & Dmitry Dunaev. (2021). *The Working Principles of ESP32 and Analytical Comparison of using Low-Cost Microcontroller Modules in Embedded Systems Design*. IEEE.
- Joel A. Cujilema Paguay & Dixys L. Hernandez Rojas. (2023, julio). *Secure home automation system based on ESP-NOW mesh network, MQTT and Home Assistant platform | IEEE Journals & Magazine | IEEE Xplore*.
<https://ieeexplore.ieee.org/abstract/document/10244182>
- Joni Welman Simatupang & Aida Mahdalena Lubis. (2022). *IoT-Based Smart Parking Management System Using ESP32 Microcontroller*. IEEE.
- Lakshmi Goswami. (2020). *IOT based Diagnosing of Fault Detection in Power Line Transmission through GOOGLE Firebase database | IEEE Conference Publication | IEEE Xplore*. IEEE Xplore. <https://ieeexplore.ieee.org/document/9143007>
- Maier, A. (2017). *Comparative analysis and practical implementation of the ESP32 microcontroller module for the internet of things*. IEEE.

- Babiuch, M., Foltýnek, P., & Smutný, P. (2019, May). Using the ESP32 microcontroller for data processing. In 2019 20th International Carpathian Control Conference (ICCC) (pp. 1-6). IEEE.
- Mazon-Olivo, B., & Hernández-Rojas, D. (2018). Rules engine and complex event processor in the context of internet of things for precision agriculture. *Computers and Electronics in Agriculture*, 154, 347-360. <https://doi.org/10.1016/j.compag.2018.09.013>
- Miladinović, V. (2020). *Razvojna platforma PlatformIO: zaključno delo* [PhD Thesis]. Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko.
- Novillo-Vicuña, J., & Rojas, D. H. (2018). *Arduino y el Internet de las cosas*. 3Ciencias.
- Omar Otoniel Flores-Cortez & Verónica Idalia Rosa. (2018). *Monitoreo remoto usando internet de las cosas*. IEEE.
- Paul Schroeder. (2021). *Visual Studio 2019 Tricks and Techniques: A developer's guide to writing better code and maximizing productivity | Packt Publishing books | IEEE Xplore*. IEEE Xplore. <https://ieeexplore.ieee.org/document/10162553>
- Pekez, N., & Kaprocki, N. (2018). Firmware Update Procedure for Audio Systems based on CS4953xx DSP family. *2018 International Conference on Smart Systems and Technologies (SST)*, 29-34. <https://doi.org/10.1109/SST.2018.8564683>
- Prem Prakash Murmu & Harshit Paul. (2019). *A Novel modernistic techniques in women security system using ESP32 and Arduino Uno*. IEEE.
- Cruz, M. L. M. H., Álvarez, M. D. C. M., Chan, M. J. R. C., & Lao, M. F. J. B. (2020, June). Analysis of the Quality in Use and Greenability with the ISO/IEC 25010 Standard. In *2020 15th Iberian Conference on Information Systems and Technologies (CISTI)* (pp. 1-7). IEEE.
- S. López, & A. Carpeño (Eds.). (2014). *Laboratorio remoto eLab3D: Un mundo virtual inmersivo para el aprendizaje de la electrónica*. IEEE.
- Satyendra K Vishwakarma & Prashant Upadhyaya. (2019). *Smart Innovation and Usages (IoT-SIU)*. IEEE.

- Scott, M. (2023). Apples to Oranges: Using Python and the pymarc library to match bookstore ISBNs to locally held eBook ISBNs. *Code4Lib Journal*, 56.
- Sharf, S. H., & Elhamied, R. A. (2021). An Efficient OTA firmware updating Architecture based on LoRa suitable for agricultural IoT Applications. *2021 International Conference on Microelectronics (ICM)*, 262-265.
<https://doi.org/10.1109/ICM52667.2021.9664942>
- Valeriu Manuel Ionescu & Florentina Magda Enescu. (2020). *Investigating the performance of MicroPython and C on ESP32 and STM32 microcontrollers*. IEEE.
- Wang, W., Qin, G., Liang, Y., Yang, L., Wang, Y., & Feng, Y. (2023, May). An OTA-oriented Protocol for Security Protection. In *2023 3rd International Conference on Frontiers of Electronics, Information and Computation Technologies (ICFEICT)* (pp. 129-134). IEEE.
- Hernández-Rojas, D. L., Fernández-Caramés, T. M., Fraga-Lamas, P., & Escudero, C. J. (2017). *Design and practical evaluation of a family of lightweight protocols for heterogeneous sensing through BLE beacons in IoT telemetry applications*. *Sensors*, 18(1), 57.
- Hernández-Rojas, D. L., Fernández-Caramés, T. M., Fraga-Lamas, P., & Escudero, C. J. (2018). *A plug-and-play human-centered virtual TEDS architecture for the web of things*. *Sensors*, 18(7), 2052.
- Vicuña, J. N., Rojas, D. L. H., Olivo, B. M., & Elizaldes, K. D. C. (2018). *Monitoreo inalámbrico de señales eléctricas de voltaje 110/220V a través de Arduino*. *Alternativas*, 19(1), 55-62.
- Calva, J. J. C., Rojas, D. L. H., Román, R. F. M., & García, C. D. R. (2021). *Seguridad IoT: Principales amenazas en una taxonomía de activos*. *Hamut'ay*, 7(3), 51-59.
- Zhou, Y., & Sun, Y. (2021). GHTRec: A Personalized Service to Recommend GitHub Trending Repositories for Developers. *2021 IEEE International Conference on Web Services (ICWS)*, 314-323. <https://doi.org/10.1109/ICWS53863.2021.00049>