UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO

CARRERA DE COMPUTACIÓN

IMPLEMENTACIÓN DE UNA APLICACIÓN MÓVIL PARA UNA SOLUCIÓN DE SEGURIDAD CON CÁMARAS DE VIGILANCIA DIURNA Y NOCTURNA PARA PROTECCIÓN Y SUPERVISIÓN PERIMETRAL EXTERNA DEL HOGAR

Trabajo de titulación previo a la obtención del Título de Ingenieros en Ciencias de la Computación

AUTORES: JOSEPH ANGELO HERRERA GUERRA

RONNY ALEXANDER MICHILENA VILLACIS

TUTOR: WASHINGTON ARSENIO RAMÍREZ MONTALVAN

Quito - Ecuador

2024

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Nosotros, Joseph Angelo Herrera Guerra con documento de identificación N°1750183541 y Ronny Alexander Michilena Villacis con documento de identificación N°1751532084, manifestamos que:

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

Quito, 28 de febrero del 2024

Atentamente,

Joseph Angelo Herrera Guerra
1750183541

Ronny Alexander Michilena Villacis
1751532084

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

Nosotros, Joseph Angelo Herrera Guerra con documento de identificación No 1750183541 y Ronny Alexander Michilena Villacis con documento de identificación No 1751532084, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del Proyecto Técnico: "Implementación de una aplicación móvil para una solución de seguridad con cámaras de vigilancia diurna y nocturna para protección y supervisión perimetral externa del hogar", el cual ha sido desarrollado para optar por el título de: Ingenieros en Ciencias de la Computación, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

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

Quito, 28 de febrero del 2024

Atentamente,

Joseph Angelo Herrera Guerra
1750183541

Ronny Alexander Michilena Villacis
1751532084

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Washington Arsenio Ramírez Montalván con documento de identificación N°1710804681, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: IMPLEMENTACIÓN DE UNA APLICACIÓN MÓVIL PARA UNA SOLUCIÓN DE SEGURIDAD CON CÁMARAS DE VIGILANCIA DIURNA Y NOCTURNA PARA PROTECCIÓN Y SUPERVISIÓN PERIMETRAL EXTERNA DEL HOGAR. Realizado por Joseph Angelo Herrera Guerra con documento de identificación N° 1750183541 y Ronny Alexander Michilena Villacis con documento de identificación N° 1751532084, obteniendo como resultado final el trabajo de titulación bajo la opción de Proyecto Técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Quito, 28 de febrero del 2024

Atentamente,

Ing. Washington Arsenio Ramírez Montalván, MSc.

1710804681

DEDICATORIA

Con profundo agradecimiento, dedico esta tesis a mis queridos padres y a toda mi familia, cuyo apoyo incondicional ha sido el faro que ha guiado mi camino hacia la realización de este sueño. Su fe inquebrantable en mí y su amor incansable han sido las fuerzas motrices detrás de cada página escrita, cada desafío superado y cada logro alcanzado. Esta obra no es solo el fruto de mi esfuerzo, sino también el reflejo del amor, la dedicación y el sacrificio que cada uno de ustedes ha vertido en mi vida. Con todo mi corazón, les extiendo mi más sincero agradecimiento, esperando honrarles con los frutos de este logro.

Joseph Angelo Herrera Guerra

Dedico este trabajo a mis estimados padres y hermana, por ser las personas que me apoyaron desde mis inicios de forma incondicional. Les agradezco profundamente cada sacrificio que han realizado por mí. Todos sus esfuerzos me han hecho llegar a la realización del presente proyecto.

Ronny Alexander Michilena Villacis

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

Los sistemas de vigilancia destacan una serie de características propias y determinadas en sus servicios. Este equipamiento de visión en tiempo real valora otro elemento de protección externo para su vinculación. En contexto al estudio se integró un desarrollo de comunicación remoto para la cámara de vigilancia utilizando el protocolo RTSP y mecanismo de alerta en microcontrolador ESP32, sujeto a una placa adaptativa de tensión eléctrica.

Al mismo tiempo el procedimiento usado en distintas fases enfatiza la valoración de dispositivos, vinculación de placa, transmisión inmediata al usuario y almacenamiento local en MongoDB. Del mismo modo la conversión de datos concreta una ejecución automática con servicios de Node Js y App Inventor registrando los eventos generados en dispositivos de Android. Cada hallazgo estará demostrando la interconexión entre componentes físicos y digitales para el entorno local programado.

Palabras clave: Protocolo RTSP, Microcontrolador ESP32, MongoDB, Node Js, App Inventor

ABSTRACT

Surveillance systems highlight a series of specific characteristics in their services. This real-time vision equipment assesses another external protection element for its connection. In the context of the study, a remote communication development for the surveillance camera was integrated using the RTSP protocol and alert mechanism in an ESP32 microcontroller, subject to an electrical voltage adaptive board.

At the same time, the step procedure highlights different phases, device assessment, board linking, immediate transmission to the user and local storage in MongoDB. In the same way, the data conversion specifies an automatic execution with Node Js and App Inventor services, recording the events generated on Android devices. Each finding will be demonstrating the interconnection between physical and digital components for the local programmed environment.

Key words: RTSP Protocol, ESP32 Microcontroller,MongoDB,Node Js,App Inventor

INTRODUCCIÓN

La era digital actual y renovación tecnológica lleva a cabo una preponderancia con diferentes prototipos evaluando control de calidad y servicio, pero sobre todo innovadores métodos de infraestructura a un control domótico. Por otra parte, el contraste que se presenta al escenario técnico considera procesos de seguridad para respaldar una vigilancia constante y eficiente para la protección y supervisión externa del hogar. Asumiendo que solo el sistema de videovigilancia se limita a grabar eventos oportunos, es necesario ver un manejo adecuado a los equipos de hardware que entren en conjunto a elementos de software para operar en base al uso del beneficiario principal.

Los implementos tanto ya predeterminados en un sistema de cámara con una interfaz propia del servicio, son un sistema inteligente por defecto que aumenta la compresión de los datos. Relativamente este control de acceso real debe acrecentar otras interfaces de control de emergencia para emitir órdenes a los actuadores que lo ocupen y emitan la salida de datos cuando identifiquen un evento totalmente inusual, entran en relación con locaciones de almacenamiento registrados.

En continuo seguimiento se plantea la ubicación de varios elementos a un módulo que direccione un circuito de control auxiliar automatizado para mitigar vulnerabilidades inmediatas, por lo que "acción y respuesta” es la clave para estrechar un rol de operador entre cliente-servidor.

Antecedentes

El estudio (Flores Gutiérrez, M. A. 2022), desarrolló un sistema de videovigilancia integral en condominios, focalizando la seguridad interna y externa. Se seleccionan cámaras IP tipo 1 (Axis M1144-L) y 2 (Axis P1343) destacando un enfoque meticuloso a sus necesidades. Estas cámaras tienen un alcance visual de 20 metros que demuestran eficacia en la supervisión de distancias y conexión mediante banda ancha para asegurar una transmisión fluida de datos. A través de mediciones prácticas se respalda el cableado eléctrico y controladores lógicos programables para grabaciones de video en red (NVR).

En (Condor Silva, M. M. 2023), se desarrolló un enfoque integral para robustecer la seguridad residencial usando un sistema de videovigilancia basado en tecnología infrarroja y controlado por visión artificial. Se usó una cámara IP tipo domo de Hikvision con una resolución de 4MP, evidenciando su eficacia en el reconocimiento facial a una distancia óptima de 18 metros. Como identificación visual se usa bibliotecas de OpenCV para detectar variaciones de temperatura corporal en cualquier movimiento. Finalmente, la clave para superar restricciones técnicas y de programación inherentes a los dispositivos de videovigilancia destaca buscar la compatibilidad en aplicaciones personalizadas.

El estudio (Ávila Sánchez et al. (2023)), desarrolló un sistema domótico orientado a medias disuasorias simulando un entorno de vivienda vigilado y activo. Se utilizó diferentes componentes relacionados con ESP32 para su conectividad en WIFI y ESP32 CAM para la captura visual en tiempo real. Al mismo tiempo componentes de disuasión y alerta como módulos de relés, dimmers automatizados y buzzer están estratégicamente alineados al eje central de cámara en constante monitoreo bajo una infraestructura de IoT. Total, la

funcionalidad se gestiona a través de una interfaz web controlada, que ofrece al usuario la flexibilidad de ajustar los parámetros de grabación y revisar el contenido capturado.

(Babatunde et al. 2022), establece un contexto centrado en el desarrollo de una solución integral de seguridad para hogares a través de una aplicación móvil innovadora. La aplicación tiene como objetivo principal la implementación de cámaras de vigilancia diurna y nocturna, aprovechando tecnologías avanzadas como el geofencing y el reconocimiento facial para establecer zonas de seguridad virtuales y autenticación sin complicaciones. Además, se explora cómo estas tecnologías, originalmente diseñadas para entornos educativos, pueden adaptarse eficazmente para mejorar la seguridad en entornos residenciales redefiniendo la perspectiva de entornos domésticos.

(Qolomany, B et al. (2019)) priorizará la confiabilidad del sistema y la protección de la privacidad del usuario, con una interfaz intuitiva y amigable para un control efectivo del entorno. En consonancia con el proyecto, se plantea el enfoque para abordar el manejo de grandes flujos de datos en tiempo real, la aplicación usará tecnologías de manejo de datos en streaming y soluciones de big data, garantizando una respuesta ágil ante intrusiones. Además, que se utilizan técnicas avanzadas de aprendizaje automático y reconocimiento de imágenes para mejorar la detección de patrones anómalos en las imágenes de las cámaras de vigilancia, aumentando la precisión de las alertas y la seguridad del hogar.

Problema

La seguridad en propiedades residenciales se ha convertido en una preocupación creciente debido al aumento de incidentes de robos y vulnerabilidades. Actualmente, se han aplicado diversas soluciones para abordar estos incidentes. Sin embargo, aún existe la necesidad de desarrollar una solución más avanzada y completa que se centre en la seguridad perimetral y aproveche la tecnología móvil para proteger de manera integral las propiedades residenciales. En varios estudios el enfoque principal parte de crear una convergencia con capacidades móviles y seguridad perimetral, buscando una solución tecnológicamente avanzada para abordar los desafíos de seguridad residencial (Alam, F., Faulkner, N., & Parr, B., 2020). Sin embargo, la divergencia de diferentes componentes en hardware implementados en cámaras evalúa la detección de movimiento y alertas tempranas, lo que garantiza una protección integral tanto en el interior como en el exterior de la propiedad (Gómez, M. C., Echeverry, A. M. L., & Sánchez, P. A. V., 2022). En un aspecto esencial la seguridad del hogar es de suma importancia (Pinto-Archundia, R. (2016)).

A nivel global, la seguridad residencial enfrenta desafíos significativos y crecientes en diversas regiones. La Unión Europea reportó más de un millón de robos en viviendas en 2019, lo que demuestra la magnitud de este problema en una región con altos estándares de seguridad (Vistazo, 2022). Además, como lo señala Xiao et al. (2016), la video-vigilancia se ha convertido en una práctica bastante común con el auge mundial de ciudades inteligentes. La investigación criminal demanda tener calidad de datos de objetos sensibles, como peatones, rostros humanos, vehículos y placas de matrícula, pero la eficacia disminuye con la alta compresión de datos por parte de las instituciones de seguridad ciudadana. Esto ilustra cómo la seguridad residencial se ha convertido en una preocupación común en toda la región (Barragán et al., 2023).

En Latinoamérica, se han realizado sondeos en varios países, revelando niveles significativos de victimización que subrayan la problemática de robos en viviendas. Los países como Guatemala y Nicaragua lideran con tasas alarmantes de 1.357 y 1.089 robos, respectivamente. Además, Perú y Paraguay han experimentado un nivel medio de robos, mientras que Uruguay, Chile y Costa Rica exhiben tasas más bajas, con 149, 173 y 189 robos por cada 100,000 habitantes. Estas cifras ponen de manifiesto que la seguridad residencial es una preocupación compartida en la mayoría de los países de América Latina. En este contexto, la integración de soluciones tecnológicas en aplicaciones móviles de seguridad se presenta como una respuesta esencial para abordar este problema creciente (Vistazo, 2022).

En Ecuador, la creciente preocupación por la seguridad residencial se refleja en las estadísticas de robos a domicilio. Durante el período de 2019 a 2021, se reportaron fluctuaciones en el número de casos (El Universo, 2022). Además, la necesidad de reforzar la seguridad se extiende más allá de los hogares, ya que el aumento de atentados terroristas y el incremento del vandalismo en las ciudades han llevado a instituciones públicas y privadas de la provincia de Pichincha a implementar sistemas de video vigilancia IP como una medida indispensable para combatir la delincuencia y mejorar la seguridad en diversos entornos (Pavón Anrango, 2016).

La ciudad de Quito ha experimentado variaciones significativas en las cifras de robos a domicilio en los últimos años. En 2019, se registraron 2,114 casos de robo a domicilio, cifra que disminuyó a 1,385 en 2020. Sin embargo, en el año 2021, esta cifra aumentó nuevamente a 1,522 casos. Los datos más recientes hasta mayo de 2022 muestran un marcado descenso, con 623 casos reportados. Estas cifras resaltan la necesidad apremiante de abordar el problema de seguridad en la ciudad. Se busca investigar y analizar en profundidad los desafíos actuales y las oportunidades para mejorar la seguridad en el entorno residencial y urbano. (El Universo, 2022).

El problema central de este proyecto de tesis se deriva de la siguiente pregunta: ¿**Puede la combinación de tecnologías de vanguardia mejorar la seguridad en las residencias y urbanizaciones frente a los problemas de seguridad más frecuentes?** Esta cuestión subraya la problemática de la inseguridad en las viviendas y áreas urbanas, donde los robos e intrusiones son recurrentes, lo cual genera un alto nivel de preocupación y afecta negativamente la calidad de vida de los residentes.

A pesar de la disponibilidad de la tecnología de seguridad, como las cámaras de vigilancia, actualmente no existen soluciones efectivas y accesibles que aborden de una manera satisfactoria estos problemas. Esta falta de respuesta adecuada deja a muchas viviendas en una situación de vulnerabilidad ante los delitos, lo que hace necesario explorar si la implementación de una solución, la cual está basada en tecnología avanzada, como podría ser una aplicación móvil, que se encuentra respaldada por cámaras de vigilancia diurna y nocturna, podría proporcionar una mejora significativa en la seguridad de las residencias y urbanizaciones.

Partiendo de la investigación se pudo determinar las siguientes causas, i) Mayor posibilidad de vulnerabilidades en dispositivos móviles que podrían ser explotadas por ciberataques, ii) Riesgo de elección de aplicaciones de baja calidad o poco confiables debido a la amplia gama de opciones disponibles, iii) Posibilidad de una sobre dependencia en la tecnología de seguridad móvil, descuidando otras medidas de seguridad tradicionales.

Las consecuencias que se pudo identificar son las siguientes, i) Incremento de riesgo de exposición de datos personales debido a la explotación de vulnerabilidades en los dispositivos móviles, ii) Posibilidad de confiar en aplicaciones móviles de seguridad que no brindan protección adecuada, lo que podría llevar a una falsa sensación de seguridad, iii) Reducción de la atención a medidas de seguridad físicas, como cerraduras y sistemas de alarma tradicionales, lo que podría dejar vulnerables los aspectos no digitales de la seguridad.

En respuesta a esta problemática, nuestro principal aporte va a consistir en la implementación de una aplicación móvil, que va a integrar cámaras de vigilancia diurna y nocturna, junto con sensores que van a estar vinculados con cercos eléctricos de seguridad perimetral. Esta aplicación, va a garantizar una mejora en la protección en hogares y/o condominios, también dicha aplicación se va a destacar, por su capacidad para detectar intrusiones y actividades sospechosas, incluyendo la identificación de posibles cortes de cables y cualquier intento de violación de la seguridad del cerco eléctrico. Cuando se detecte una amenaza, la aplicación va a enviar alertas en tiempo real tanto a las cámaras de vigilancia como a las luces de dichas cámaras, actuando como un disuasivo efectivo y a su vez brindando una respuesta inmediata.

En base al desarrollo de una tecnología de compresión de datos con criterios analíticos específicos, como la grabación selectiva de movimientos sospechosos, se convierte en un elemento crucial para abordar eficazmente el desafío del almacenamiento de información de vigilancia, lo cual permitirá a los usuarios de la app gestionar y supervisar la seguridad de sus hogares o condominios de manera sencilla, a través de una interfaz de usuario diseñada para una experiencia fluida tanto en el backend como en el frontend.

Justificación

La relevancia de este proyecto de investigación se fundamenta en la creciente necesidad de abordar la problemática de la seguridad residencial de manera efectiva en la ciudad de Quito. La seguridad del hogar es una preocupación primordial para los residentes, y para las tasas crecientes de robos y delitos que tienen que ver con domicilios, los han generado una creciente inseguridad en la comunidad. La integración de tecnologías avanzadas en una aplicación móvil que incluye cámaras de vigilancia diurna y nocturna, cerco eléctrico y un sistema de seguridad perimetral.

La pertinencia de esta investigación se deriva de la necesidad urgente, de medidas tanto en el ámbito digital como en el físico para proteger a las personas, a sus bienes y a los entornos residenciales en general. La implementación de la aplicación móvil propuesta beneficiará directamente a las familias y hogares, brindando un mayor control y seguridad, permitiendo así una vigilancia más efectiva. Además, también impactará de manera indirecta a otros beneficiarios, como podrían ser los guardias de seguridad, las empresas de seguridad y el sector inmobiliario, al momento de elevar los estándares de seguridad y de esta manera ofrecer soluciones más avanzadas. La incorporación de tecnologías de vanguardia, como el reconocimiento facial y de objetos, demuestra la innovación en la seguridad residencial y su capacidad para abordar los desafíos actuales de manera integral.

Objetivos

Objetivo general:

Implementar una solución de seguridad en un condominio mediante la integración de cámaras de vigilancia y sistemas de seguridad complementarios, para una aplicación móvil intuitiva con el propósito de garantizar la protección efectiva de los residentes.

Objetivos específicos:

- Explorar diferentes cámaras de vigilancia multifuncional aprovechando tecnologías avanzadas como la visión nocturna en color detección de movimiento y almacenamiento local.
- Integrar sistemas que combinen sensores y cámaras en áreas críticas del hogar, aprovechando la integración con un cerco eléctrico para detectar intrusiones en situaciones de corte de energía.

- Evaluar la idoneidad de estructura consolidando pruebas exhaustivas en una aplicación móvil con funciones adicionales para el software de cámara en tiempo real

Metodología

Metodología experimental

Para el desarrollo del proyecto se utilizó la metodología experimental ya que se empleó un sistema controlado por microcontrolador, simulando situaciones realistas de seguridad mediante activación y desactivación. De eso se desprende la variable de alertas para controlar la grabación de mensajes, aunado a esto, se centra en la adaptabilidad del reproductor multimedia y la recolección de datos en los eventos. Así pues, este proceso al cambio de respuesta es dinámico basado en una base de dato ajusta su comportamiento según la información configurada en la base de datos, para su gestión en cualquier evento.

Por otra parte, se planifican pruebas con adaptabilidad de reproductor multimedia, y se recopilan datos para evaluar la eficacia del sistema en condiciones diversas. Además, se contempla el lanzamiento de otros cambios de imagen que pueden resultar de estudios manuales y automáticos, permitiendo una mejora continua del sistema.

Este enfoque experimental asegura una evaluación rigurosa y objetiva de la aplicación móvil con el sistema de cámara infrarroja controlada por microcontrolador, permitiendo ajustes y mejoras. La metodología se adapta a las necesidades específicas de un proyecto que involucra seguridad perimetral y tecnología de grabación mediante una cámara infrarroja

CAPÍTULO I

MARCO TEÓRICO

1.1.1 Estado de Arte

En (Velasco Llano, 2018), el objetivo principal es implementar un prototipo de cerco eléctrico autosustentable que permita de manera remota conocer si se produce una vulneración al perímetro establecido. La metodología que se aplica consta de 3 partes: i) conceptualización, ii) implementación y iii) diagnóstico de resultados. El prototipo permite mantener una visualización del estado del cerco eléctrico en tiempo real de manera remota mediante el envío de datos a través de radio frecuencia. El sistema mantiene un consumo de corriente de 245 [mA] generando mediante un panel solar de 50[Wp] una autonomía completa para su operación, finalmente el tiempo de respuesta frente a una vulneración es de 3 segundos. Concluyendo, la implantación de un sistema inalámbrico para detectar fallos en un cerco eléctrico representa una inversión segura frente a un sistema autosustentable y de acción rápida.

En (Cruz Cáceres, 2020), el autor desarrolla un sistema de seguridad basado en IOT (Internet of things) el cual se aplica a cualquier vivienda utilizando componentes básicos con el objetivo de brindar un ambiente seguro y que pueda monitorear de forma remota. La metodología utilizada se enfoca en el método inductivo donde: i) Conceptualiza, ii) Implementa, iii) Somete a pruebas de funcionamiento. El sistema de seguridad se valida con un estudio de casos para probar el funcionamiento de los componentes, las alertas y monitoreo remoto del sistema, el tiempo tarda en enviar la alerta activando el sensor PIR unos 60 segundos y el tiempo máximo de actualizar los datos es de 4 segundos. Se concluye que el sistema mantiene un funcionamiento efectivo con tiempos de respuestas cortos.

En (Ojeda Crespo & Cabrera Mejía, 2020), el objetivo de la investigación es mostrar la importancia del manejo de datos en conceptos como IoT o domótica. El documento realza el

análisis del protocolo MQTT como una solución flexible y sencilla para establecer una conexión entre varios dispositivos. Como resultado del trabajo investigativo, se concluye que la domótica y el internet brindan ventajas, pero una buena práctica de seguridad en los datos que se manejan evitaría que un sistema de seguridad se torne un sistema de expiación.

El propósito de este estudio (Medina Espinosa, 2017)es diseñar un sistema automático de videovigilancia, detección de intrusos, sistema de acceso, alarmas técnicas, control de iluminación, control de persianas para una empresa. Los puntos centrales del trabajo de investigación son: i) sensores y actuadores, ii) Interconexión entre los dispositivos que conforman el sistema, iii) Monitoreo remoto, iv) Alertas. Se utiliza varios protocolos de comunicación como, Zelio, Simon, ZigBee, KNX, BUSing, las cámaras mantienen un protocolo IP, con lo cual se conforma el sistema automatizado. Concluyó que el sistema es robusto, frente al fallo de uno de los dispositivos del sistema, el conjunto como tal sigue su funcionamiento sin verse afectado.

En (Ribero Corzo & Prieto Guerrero, 2020), tiene como objetivo diseñar una guía que muestre los riesgos a los cuales se encuentran expuestas las cámaras de seguridad IP. La metodología que utilizaron fue: i) Recolección de información, ii) Análisis de información, iii) Ejecución, iv) Desarrollo de entregables. El documento resalta la importancia de la seguridad tanto a nivel de hardware, como a nivel de software, brindando un enfoque en la prevención de riesgos ante ciber ataques. Finalmente concluyó que, en la actualidad las aplicaciones a gran escala de los dispositivos IoT comprometen una amplia atención en la prevención de ataques cibernéticos, consiguiendo una guía donde se muestran los riesgos actuales acompañadas con una serie de recomendaciones para prevenir dichos riesgos.

En (Oñate Miranda, 2020), el enfoque del estudio es el diseño y construcción de nodos inteligentes para la detección de armas dentro de una red de video-vigilancia utilizando visión artificial. La investigación siguió una metodología estructurada en tres fases: i) recopilación de imágenes para el entrenamiento del clasificador, ii) diseño y construcción del prototipo, y iii) validación del prototipo mediante pruebas de precisión y sensibilidad, y pruebas de tiempo de procesamiento y respuesta. El sistema diseñado hace uso de tecnologías de captura de imágenes y análisis mediante algoritmos de visión artificial, como los clasificadores HAAR, ejecutándose sobre una plataforma Linux en hardware de bajo costo. Este envía alertas GSM, almacenas imágenes detectadas en una base de datos de Gmail para análisis y reconocimiento facial. Las pruebas demostraron que, a pesar de los desafíos de detección en diferentes entornos, el sistema puede enviar alertas en un tiempo promedio de 6.5 segundos y almacenar imágenes en Gmail en 10 segundos, proporcionando una solución efectiva y en tiempo real para la detección de armas.

En (Monteros Mejía, 2015), el autor propone el diseño de un sistema de video-vigilancia inalámbrico para la ciudad de Cayambe con el fin de contribuir a la reducción del índice delictivo en la zona. La metodología del proyecto abarca desde la conceptualización del sistema hasta su implementación y evaluación, enfocándose principalmente en el diseño sobre una red inalámbrica de datos en malla (WMN), la cual permite el uso de la banda de frecuencia libre de 5 GHz para la transmisión de datos. La investigación incluye un análisis detallado del entorno, una encuesta de mercado para determinar la aceptación del sistema por parte de la población, y el diseño detallado del sistema de video-vigilancia considerando la ubicación de las cámaras, el tráfico de red generado, y la infraestructura necesaria para el almacenamiento y gestión de video. El trabajo culmina con la implementación de un prototipo para demostrar las capacidades funcionales del sistema y su viabilidad técnica y económica.

En (Azurin Villanque y Núñez Neyra, 2022), los autores diseñan un innovador sistema de videovigilancia para el Condominio La Isla Asia, utilizando tecnología IP inalámbrica y capacidades de IVS (Intelligent Video System) para cercos virtuales. La metodología adoptada para este proyecto se divide en tres fases críticas: i) la identificación y análisis de las necesidades de seguridad del condominio, ii) el diseño y la implementación del sistema de videovigilancia que incluye la selección de cámaras IP con capacidad IVS y la configuración de enlaces inalámbricos para garantizar una cobertura completa, y iii) la validación del sistema mediante pruebas de campo que demostraron la eficacia de la videovigilancia en la detección de intrusiones y actividades sospechosas en tiempo real. El sistema propuesto aprovecha la tecnología inalámbrica para superar las limitaciones físicas de la instalación y utiliza cámaras de alta definición para monitorear áreas críticas del condominio, ofreciendo una solución segura, escalable y de bajo mantenimiento. Los resultados obtenidos destacan la mejora significativa en la seguridad y la gestión de incidentes, demostrando la viabilidad del sistema de videovigilancia para proteger eficazmente los bienes y la integridad de los residentes del Condominio La Isla.

En (Contreras Clavijo y Rojano Mueses, 2023), se diseña un sistema inteligente de monitoreo y control para la planta de tratamiento de agua potable "El Carrizal-Salcedo", enfocado en mejorar la eficiencia y seguridad del tratamiento del agua mediante el uso de IoT e inteligencia artificial. La metodología implementada consta de varias etapas: i) la recolección y análisis de datos necesarios para el desarrollo del sistema, ii) el diseño y construcción del prototipo utilizando microcontroladores Arduino y ESP8266 para el procesamiento y recolección de datos de los sensores, y iii) la implementación y pruebas del sistema para validar su funcionamiento. Este enfoque permitió el monitoreo en tiempo real de variables críticas como temperatura, caudal, pH, conductancia, y turbidez, así como el control automático de bombas y electroválvulas, mejorando significativamente la gestión del tratamiento del agua.

La validación del sistema demostró su capacidad para operar de forma autónoma, con una interfaz de usuario amigable para el monitoreo y control remoto, proponiendo una solución viable y eficaz para la automatización de procesos en plantas de tratamiento de agua bajo el concepto de Industria 4.0.

A continuación, se presenta el resumen del estado del arte en la **Tabla 1** y **Tabla 2**.

Tabla 1

Resumen de estado de arte de dispositivos eléctricos

Cita	Componente	Alimentación	Fabricante	Memoria	Velocidad de reloj	Salida	Dimensiones	Características
[1]	Sensor de corriente	5V	Allegro	N/A	N/A	1.5 hasta 3,5 V	57 mm x 32 mm x 21 mm	Voltaje de salida de bajo ruido, error de ±1,5%, uso de módulo con placa para mayor facilidad de usa
[2]	ESP-WROOM-32	3,3V	Espressif	520 KB RAM 448 KB ROM	40 MHz	Digital, PWM, DAC	18 mm x 25.5 mm x 3.1 mm	Periféricos: Tarjeta SD, UART, SPI, SDIO, I2C, LED PWM, I2S, IR, Contador de pulsos, GPIO, ADC, DAC, Two-Wire Automotive Interface (TWAI®), compatible Con ISO11898-1 (CAN Specification 2.0), Wifi 802.11 b/g/n, Bluethoot
[3]	ESP8266	3,3V	Espressif	<50 KB RAM	52 MHz	Digital, PWM, DAC	48 mm x 26 mm x 3.1 mm	Periféricos: Tarjeta SD, UART, SPI, SDIO, I2C, LED PWM, I2S, IR, Contador de pulsos, GPIO, ADC, DAC, Two-Wire Automotive Interface (TWAI®), compatible Con ISO11898-1 (CAN Specification 2.0), Wifi 802.11 b/g/n, Bluethoot
[4]	Router	9 V	TP-Link	NA	300 Mbps	WIFI	182mm x 35mm x 128mm	Router wifi de 300 Mbps, cuatro puertos LAN RJ45, estándar IEEE 802.3, 802.3u, configuración de DHCP, VPNs, VLANs,
[5]	Xbee	3,4V	Xbee	NA	250 Kbps	Protocolo Xbee	56,4×40,6×15,7 mm	Estándar IEEE 802.15.4, Entradas y salidas digitales, entradas análogas

Nota. Características de dispositivos que utilizan otros autores en el estado de arte. Elaborado por: Los Autores

Tabla 2

Resumen de estado de arte de dispositivos para cámaras

Cita	Cámara	Resolución	Ángulo de Visión	Comunicación Bidireccional	Resistencia al Agua	Otras **Características**
[1]	Ezviz C3W Pro	N/A	104° horizontal, 87° vertical	Si	IP67	Detección inteligente, visión nocturna
[2]	Samsung Hanwha Techwin SN0-L6083R	2 Megapíxeles (FullHD)	N/A	N/A	IP66, IK10	Resistente a la intemperie y al vandalismo
[3]	See3CAM_ CU135	Alta resolución	N/A	N/A	N/A	Enfoque automático, Full HD a 60 fps
[4]	Dahua DH-SD6582	2 Megapíxeles	N/A	NA	N/A	Detección de movimiento
[5]	Hikvision DS-2DE4220-AE	2 Megapíxeles	N/A	NA	N/A	Sensibilidad lumínica mejorada, zoom digital más amplio

Nota**.** Características de dispositivos que utilizan otros autores en el estado de arte Elaborado por: Los Autores

En la **Tabla 3**, se comparan las características de dos modelos distintos de cámaras. Se ha identificado una dificultad en el modelo REOLINK Duo 2, específicamente en lo que respecta al acceso al video en tiempo real, atribuida a sus protocolos de comunicación. Esto se debe a las complejidades robustas predeterminadas del equipo, no ofrecen la flexibilidad necesaria para sistemas que requieren una interacción directa y personalizada con los flujos de video, limitando su interoperabilidad.

Por otro lado, la cámara TP-Link Tapo C100, es compatible al entorno de conexión, aunque tiene recursos limitados, mantiene similitudes en características esenciales facilitando la interacción domótica de hardware con obtención del video en vivo. Esta cualidad resulta ser de gran ventaja para el sistema de seguridad propuesto, asegurando una integración sin contratiempos y funcionamiento óptimo de implementación.

Tabla 3

Resumen de modelos de cámaras de vigilancia

N	M	DSN	CVQ	CNV	TVS	ASM	MD	PRC	IMS	FPS	PCM	EDT	ACN	DM	IAA	VA	DA
[1]	TP-Link Tapo C100 V1		2MP (1080p)	Wi-Fi 2.4 Ghz	Nocturna 3 LED hasta 30 pies	Tarjeta microSD hasta 128 GB/Cloud (opcional)	Vista de 105°	Hi3518E de 1 núcleo	1/3,2” CMOS	15 fps	HTTPS RTSP TCP/IP DNS SSL/TLS	AES-128	DC 9.0V/0.6A,	SI	Buenas capacidades de integración	Instalación y configuración sencillas, notificaciones en tiempo real	Restricciones de integración y sensibilidad en detección de movimiento
[2]	REOLINK Duo 2 – Sistema de 2 Cámaras		4K ultra HD	Conexión Wi-Fi doble banda (2.4 GHz y 5 GHz)	Nocturna 6 piezas de LEDs hasta 80 pies	Tarjeta microSD hasta 256 GB/Cloud (opcional)	Vista de 360	SOC de 4 núcleos	1/2,7” CMOS	20fps	HTTPS RTSP TCP/IP DNS SSL/TLS	AES-256	DC 12,0 V/2 A, 15 W	SI	Potencialmente limitada por bloqueos de interfaz	Calidad superior de imagen, detección de movimiento más sofisticada	Interfaz y manejo más complejos, limitaciones de exportación de video

Nota. M= Modelos, DSN= Diseño, CVQ= Calidad de video, CNV= Conectividad, TVS= Tipo de visión, ASM= Almacenamiento MD= Detección de movimiento, PRC= Tipo de procesador, IMS= Sensor de imagen, FPS= Fotogramas por segundo, PCM= Protocolos de comunicación, EDT= Encriptación de datos, ACN= Alimentación/Consumo, DM= Detecta Movimiento, IAA= Integración con Aplicaciones, VA= Ventajas Adicionales, DA= Desventajas Adicionales. Elaborado por: Los autores

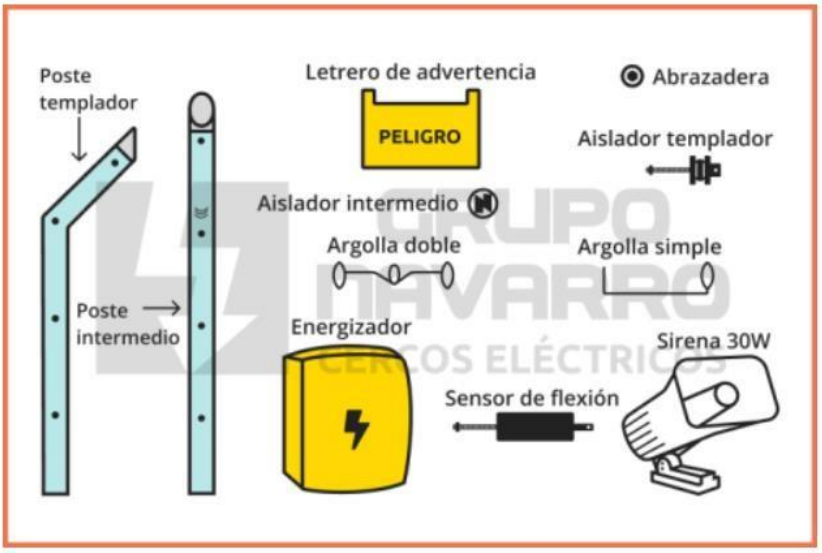
1.1.2 Sistema de seguridad perimetral con cerco eléctrico

Un cerco eléctrico es un mecanismo de seguridad perimetral, tiene como propósito disuadir, detectar y retrasar intentos de acceso no autorizado, los componentes que constituyen un cerco eléctrico son: **cables o alambres electrificados, fuente de energía, postes y aisladores, sistema de alarma.**

Para el diseño de un cero eléctrico es importante considerar factores que garanticen cumplir con los requisitos específicos de seguridad, considerando para el caso de propiedad residencial, en la **Figura 1** se puede observar los componentes que se utilizan para instalar un cerco eléctrico perimetral estándar, los materiales necesarios son:

Figura 1

Materiales para la instalación de un cerco eléctrico



Nota. Cerco eléctrico perimetral estándar. Fuente: Navarro (2021)

Uno de los elementos importantes para el desarrollo de la aplicación es el dispositivo energizador, ya que permite conocer el estado del cerco eléctrico mediante un contacto seco, al presentarse una vulneración se genera una alerta que es transmitida hacia el servidor, el cual se encarga de iniciar la grabación mediante la cámara de videovigilancia, generar alertas hacia

la aplicación móvil y finalmente generar datos históricos con el evento sucedido. Todo el fujo de datos se lo realiza mediante la red local.

1.1.3 Dispositivos electrónicos

Para la implementación del proyecto se consideran los siguientes dispositivos electrónicos presentados en la **Tabla 4**:

Tabla 4

Dispositivos a utilizar en el proyecto

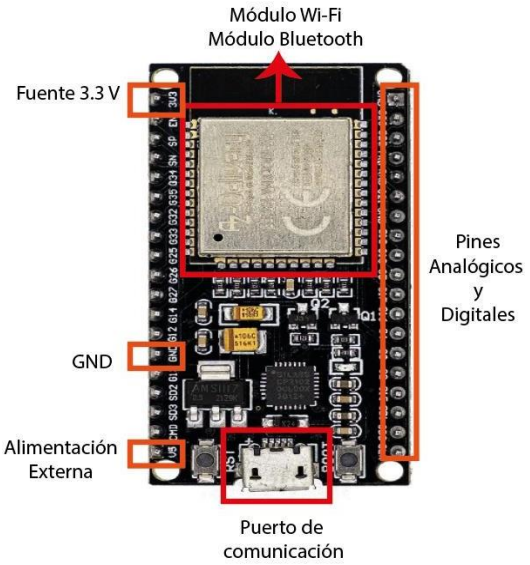
	Dispositivos a utilizar para el desarrollo del proyecto
Unidad	Componentes
1	Módulo ESP32 WROOM-32
1	Cámara de vigilancia IP Tapo C 100
1	Resistencia 1KΩ
1	Fuente regulada 3.3 a 12 [V]
1	Router Wifi
1	Computador portátil
1	Placa fenólica de baquelita con cobre
1	Teléfono móvil con sistema operativo Android

Nota. Dispositivos electrónicos que tendrá el sistema. Elaborado por: Los autores

Módulo ESP32. Un microcontrolador altamente valorado debido a su combinación de costo accesible y eficiencia energética, contribuyendo a su creciente popularidad en el ámbito de la electrónica y la Internet de las cosas (IoT) (Systems Espressif, 2023). Este microcontrolador cuenta con módulos Bluetooth y Wifi, 18 canales de convertidor analógico digital, 12 canales de salida PWM, 10 GPIO de detección capacitiva (8 entrada pull-up interno), en la **Figura 2** se muestra una imagen referencial del módulo ESP32.

Figura 2

Placa ESP32



Nota**.** Placa referencial del módulo ESP32. Fuente: Systems Espressif, 2023

Cámara de vigilancia IP Tapo C 100. La función principal de la cámara de videovigilancia en el sistema de seguridad es proporcionar una visión en tiempo real de un área designada. En la **Figura 3** se presenta la cámara de vigilancia IP Tapo C 100 del fabricante tp-link brinda videos de alta definición con imágenes nítidas de 1080p Full HD. Equipado con un sensor CMOS de 1/3,2 pulgadas, garantiza una captura efectiva de luz para obtener imágenes claras y detalladas. Además, su campo de visión en 105°, cuenta con visión nocturna avanzada de 850 nm que alcanza hasta 30pies, brindando detección de movimiento con notificaciones, alarma de luz y sonido para ahuyentar visitantes no deseados, además cuenta con un protocolo de comunicación IP lo que permite acceder a la imagen de video, así como, habilitar el inicio y final de la grabación (PINCOMPUTERS, 2024).

Figura 3

Cámara de seguridad

Nota. Imagen referencial de la cámara de vigilancia IP Tapo C100. Fuente: tp-link, 2024.

En la **Tabla 5** se presenta una recopilación de las características técnicas de la cámara de vigilancia IP Tapo C100:

Tabla 5

Características técnicas de la cámara de vigilancia IP Tapo C100.

Cámara de seguridad Wifi Tapo C 100	
Definición de cámara	1080 p Full HD
Visión nocturna	Distancia 30 pies
Detección de movimiento	Envía notificaciones, activa alarma de luz y sonido
Almacenamiento	128 GB (384 horas-16 días)
Wireless Rate	11 Mbps
Wireless Security	WPA/WPA2-PSK
Extensión de video	.avi

Nota. Especificaciones técnicas de cámara. Elaborado por: Los autores

Resistencia 1KΩ. Proporciona una ruta de baja resistencia hacia GND, con el fin de que no haya una señal de entrada activa (como un botón presionado o una señal de otro circuito) y la entrada se mantenga en un nivel bajo debido a la conexión a tierra.

Fuente regulada 3.3 a 12 [V]. Es un dispositivo que proporciona una salida de voltaje ajustable y se conecta a la red doméstica 110 [V].

Router Wifi. Dispositivo de red que se utiliza para proporcionar acceso inalámbrico a

Internet o a una red privada. Este tipo de router permite a múltiples dispositivos como smartphones, tabletas, computadoras y otros dispositivos inteligentes conectarse a Internet o comunicarse entre sí sin necesidad de cables físicos.

Computador portátil. Es un dispositivo electrónico diseñado para ser transportado y utilizado fácilmente en diferentes ubicaciones. Combina todas las funcionalidades de un computador de escritorio en un dispositivo compacto y portátil. Sus características son: DELL Inspiron, memoria RAM 8GB, Core i7.

Placa fenólica de baquelita con cobre. Es utilizada para montar los elementos necesarios para la lectura del estado del cerco, sus dimensiones son 5x6 cm.

Teléfono móvil con sistema operativo Android. Es un dispositivo inteligente que utiliza un sistema operativo desarrollado por Google como su plataforma principal para funcionar. Sus características son: Xiaomi Redmi 9s, versión Android 12.

Tabla 6

Función de los dispositivos electrónicos en el sistema.

Dispositivo	Uso
Módulo ESP32 WROOM-32	Se utiliza para detectar vulneraciones en un cerco eléctrico a través de la lectura de un pin digital en configuración PULL-DOWN y obtener imágenes en tiempo real del perímetro de la vivienda
Cámara de vigilancia IP Tapo C 100	Se usa para detectar violaciones en un cerco eléctrico con un pin digital PULL-DOWN y capturar imágenes del perímetro de la vivienda en tiempo real
Resistencia 1KΩ	Se emplea para configurar PULL-DOWN y permitir la lectura digital cuando el energizador del cerco eléctrico se cierra tras una vulneración de seguridad.
Fuente regulada 3.3 a 12 [V]	Es utilizada para energizar el módulo ESP32 como una fuente externa, se conecta a la red 110[V], y entrega un voltaje regulado DC.
Router Wifi	El Router se configura como un punto de acceso WiFi local que permite la interconexión de todos los elementos del sistema y asegura un alcance adecuado en toda la vivienda.
Computador portátil	El ordenador actúa como servidor intermediario entre el módulo ESP32, la base de datos y el teléfono móvil.

Placa fenólica de baquelita con cobre	Es utilizado para montar el módulo ESP32 junto a borneras que permiten la conexión con el energizador del cerco eléctrico
Teléfono móvil con sistema operativo Android	Se utiliza para visualizar el video en vivo del perímetro de la vivienda, así como videos históricos de eventos registrados.

Nota. Descripción y uso de dispositivos. Elaborado por: Los autores

1.1.4 Lenguajes de programación

JavaScript (JS). es un lenguaje de programación que se utiliza principalmente para crear páginas web interactivas. Es un lenguaje interpretado, lo que significa que los scripts se ejecutan sin necesidad de ser compilados. JS es un componente esencial de la mayoría de los sitios web y funciona junto con HTML y CSS para mejorar la experiencia del usuario. Se puede utilizar tanto en el cliente (navegador web) como en el servidor (Node.js) (Flanagan, 2020).

MicroPython. es una implementación eficiente y optimizada del lenguaje de programación Python 3, creada para ejecutarse en microcontroladores y en ambientes de hardware limitado. Es una herramienta poderosa para programar dispositivos pequeños, especialmente en el campo de la Internet de las Cosas (IoT) (Damien, 2023).

1.1.5 Aplicaciones

Proteus Design Suite. creada por Labcenter Electronics Ltd. en 1990, es un software diseñado para simular y desarrollar sistemas eléctricos y diseños con componentes electrónicos. Esta herramienta permite verificar el funcionamiento de circuitos electrónicos, identificando y corrigiendo errores antes de su implementación definitiva, asegurando así un desempeño eficiente. Posee una interfaz amigable y ofrece funciones como visualización en 2D y 3D, diseño modular, opciones de configuración de red, enrutamiento automático y diversos sistemas para diseño y ensamblaje, facilitando así un aprendizaje rápido y efectivo (Labcenter Electronics Ltd., 2017).

Visual Studio Code. Comúnmente conocido como VS Code, es un editor de código fuente gratuito y de código abierto desarrollado por Microsoft. Disponible para Windows, macOS y Linux, es ampliamente utilizado para la programación en una variedad de lenguajes de programación. Ofrece características como soporte para depuración, control de versiones Git integrado, resaltado de sintaxis, autocompletado inteligente, fragmentos de código y refactorización de código. Además, su funcionalidad se puede ampliar a través de extensiones, lo que permite a los usuarios personalizarlo para adaptarse a diferentes lenguajes de programación y herramientas de desarrollo (Flores, 2022).

Thonny. Es un entorno de desarrollo integrado (IDE) específicamente diseñado para la programación en Python, enfocado principalmente en principiantes. Desarrollado por la Universidad de Tartu en Estonia, se destaca por su simplicidad y facilidad de uso, ofreciendo una plataforma amigable para aquellos que están aprendiendo a programar. Cuenta con características como un depurador simple y efectivo, visualización de la ejecución de código y la capacidad de mostrar la estructura interna de Python al ejecutar programas (Juanweb, 2023).

APP inventor. Es una plataforma de desarrollo de aplicaciones basada en la web, creada por el Instituto Tecnológico de Massachusetts (MIT). Está diseñada para permitir a usuarios de todos los niveles, especialmente a principiantes y estudiantes, desarrollar aplicaciones para dispositivos Android de manera sencilla e intuitiva. Utiliza una interfaz de programación visual basada en bloques, lo que elimina la necesidad de escribir código, facilitando así el proceso de aprendizaje y desarrollo (Posada Prieto, 2019).

Node.js: Representa un paradigma de "JavaScript en todas partes", unificando el desarrollo de aplicaciones web alrededor de un único lenguaje de programación, en lugar de diferentes lenguajes para scripts del lado del servidor y del lado del cliente (Kinsta, 2023).

Jmeter: Para la prueba de carga y el análisis de rendimiento de aplicaciones web. Aunque inicialmente se creó para probar aplicaciones web, con el tiempo se ha expandido para soportar otros tipos de pruebas. JMeter puede ser utilizado para simular una carga pesada en un servidor, grupo de servidores, red o aplicación para probar su resistencia o para analizar el rendimiento general bajo diferentes tipos de carga (CloudAPPi, 2024).

MongoDB: Es un programa popular de base de datos NoSQL de código abierto, reconocido por su flexibilidad, escalabilidad y facilidad de uso. Pertenece a la categoría de bases de datos orientadas a documentos, almacenando datos en documentos flexibles similares a JSON. MongoDB utiliza un esquema flexible llamado BSON (Binary JSON), que permite estructuras de datos dinámicas y jerárquicas dentro de los documentos (MongoDB, Inc., 2024).

Flatcam: Es ideal para hobbistas, pequeños talleres de fabricación, y educadores que necesitan una solución económica y flexible para la producción de PCBs mediante fresado CNC. Permite a los usuarios llevar sus diseños de PCB desde el concepto hasta la producción física sin necesidad de costosas herramientas de fabricación o servicios de producción externos (Fernández Alazte, 2023).

Tp-Link: TP-Link es conocido por su compromiso con la innovación, ofreciendo productos que incorporan las últimas tecnologías de red para mejorar la experiencia del usuario y satisfacer las crecientes demandas de conectividad. La empresa ha sido reconocida en diversas ocasiones por su liderazgo en el mercado de dispositivos de red, destacando su fiabilidad, rendimiento y facilidad de uso (TP-Link Corporation Limited, 2017).

Tabla 7

Lenguajes de programación y aplicaciones utilizados.

Lenguajes de programación	
Lenguajes	Uso
JavaScript	Se realizó la configuración del servidor para interconectar el módulo ESP32, generar la página web con los datos obtenidos, y guardar la información en la base de datos
MicroPython	Se realizó la programación del módulo ESP32 para detectar el estado del cerco eléctrico
Aplicaciones	
Aplicación	Uso
Proteus Design Suite	Se realizó el diseño PCB de la placa donde se colocan el módulo ESP32 y varias borneras
Visual Studio Code	Se realizó la instalación de librerías necesarias para programar el servidor.
Thonny	Se realizó la instalación de librerías necesarias para programar el módulo ESP32
APP inventor	Se utilizó para el diseño de la aplicación móvil, tanto en el entorno gráfico como de programación.
Node.js	Se utilizó como entorno de ejecución del código del servidor.
Jmeter	Se utiliza para levantar el servidor.
FlatCam	Se realizó el archivo requerido para la impresión de la placa PCB
TP-Link	Se realizó la configuración de la cámara de seguridad con la red Wi-Fi
MongoDB	Se utilizó como base de datos.

Nota. Herramientas de lenguaje y aplicación utilizados. Elaborado por: Los autores

1.1.6 Librerías

Las librerías utilizadas para el desarrollo del proyecto se describen en la **Tabla 8**. Estas herramientas fueron escogidas para facilitar la implementación del proyecto, enfocándose en funcionalidades clave como la interconexión de módulos, la gestión de protocolos, la capacidad para establecer conexiones a distancia, y la compatibilidad con una variedad de componentes.

Tabla 8

Librerías de aplicaciones utilizadas

LIBRERIAS DE APLICACIONES **Proteus Desing Suite**	
Librería	Uso
ESP32 Library for Proteus	Proporciona una variedad de dispositivos ESP32 entre ellos Wroom-32
Connectors Library	Proporciona elementos conectores como borneras
Thonny	
Machine:	Esta biblioteca en MicroPython controla el hardware del dispositivo, incluyendo pines de E/S y ADC (Convertidores Analógico a Digital).
network	También parte de MicroPython, se usa para manejar conexiones de red, como Wi-Fi
socket	Proporciona acceso a la interfaz de sockets BSD, permitiendo comunicaciones en red, como la creación de un servidor web básico.
time y utime	Estas bibliotecas gestionan funciones temporales como retrasos (sleeps) y marcas de tiempo. utime es la versión de MicroPython de la biblioteca estándar de Python llamada time.
uio	Una versión de MicroPython de la biblioteca estándar de Python io, utilizada para manejar operaciones de entrada/salida con archivos.
urequests	Es el equivalente en MicroPython de requests en Python estándar. Se utiliza para realizar peticiones HTTP, como enviar datos a un servidor.
json	Se usa para codificar y decodificar datos en formato JSON, un formato estándar para el intercambio de datos.
uasyncio:	Versión de MicroPython de la biblioteca asyncio de Python, que se utiliza para escribir código concurrente usando la sintaxis async/await.
Visual Studio code	
express	Es un marco de trabajo para Node.js que facilita la creación de aplicaciones web y API. Se utiliza para manejar las solicitudes HTTP y el enrutamiento
body-parser	Middleware para Express que se utiliza para analizar el cuerpo de las solicitudes entrantes, particularmente en formatos JSON y URL-encoded
mongodb:	Proporciona una API para interactuar con bases de datos MongoDB desde Node.js. Se utiliza para realizar operaciones como insertar y recuperar datos de la base de datos
vlc-simple-player	Una biblioteca para interactuar con VLC Media Player, usada en este contexto para controlar la reproducción de video

dotenv	Se utiliza para cargar variables de entorno desde un archivo. env. Esto es útil para mantener configuraciones sensibles, como cadenas de conexión a bases de datos, fuera del código fuente
node-rtsp-stream	Esta biblioteca permite transmitir video en tiempo real (RTSP) a un navegador web, convirtiendo flujos RTSP a WebSockets
avi2mp4.js	Permite realizar una conversión de formato de video de ".avi" a "MP4"
APP Inventor	
Notifier	Se usa para cargar variables de entorno desde un archivo "env", protegiendo configuraciones sensibles como conexiones a bases de datos.
Sound	Se usa para controlar el sonido, incluyendo silenciar, activar o ajustar el volumen, y generalmente se representa con un icono de altavoz u ondas de sonido.
Sharing	Permite a los usuarios compartir contenido a través de diferentes medios o en varias plataformas. A menudo parece una flecha que se ramifica o un símbolo de re
Texting	Es una función de mensajería para enviar mensajes entre usuarios, generalmente representada por un globo de diálogo o un sobre

Nota. Librerías de programación usadas. Elaborado por: Los autores

1.1.7 Protocolos de red

Los protocolos de comunicación de Internet son un conjunto de reglas y estándares que permiten la transmisión de datos entre dispositivos en una red. Estos protocolos son esenciales para que distintos dispositivos puedan comunicarse entre sí, independientemente de sus diferencias en software y hardware (IONOS, 2019).

IP (Protocolo de Internet): Este protocolo define cómo los dispositivos conectados a Internet deben funcionar, enfocándose en el direccionamiento y el enrutamiento de paquetes de datos. La aplicación utiliza el protocolo IP para conectar los dispositivos a la red, en el caso particular del módulo ESP32 a través de su librería Network.WLAN() establece una conexión con el router Wifi (Delgado, 2022).

HTTP (Protocolo de Transferencia de Hipertexto): Funciona como un lenguaje que permite la comunicación entre un navegador y un servidor web. Es fundamental para mostrar páginas web y opera en un formato de solicitud y respuesta. HTTP es crucial para la navegación web y trabaja en conjunto con otros protocolos como DNS (MDN Web Docs, 2023).

DNS traduce nombres de dominio en direcciones IP. Los nombres de dominio son los nombres fáciles de recordar que usamos para acceder a sitios web y otros recursos en Internet, como www.google.com, se utiliza para acceder a las páginas web generadas para acceder al estado del cero y las diferentes opciones de video de vigilancia (Cloudflare, 2023.).

RTSP (Real Time Stream Protocol): es un protocolo de red de nivel de aplicación utilizado para controlar la transmisión de datos multimedia en tiempo real. RTSP proporciona un marco extensible para permitir el control de la reproducción de medios en servidores multimedia. Este protocolo es utilizado comúnmente para la transmisión de video y audio en aplicaciones de streaming como la vigilancia por vídeo, la televisión por Internet y las conferencias de vídeo (Dacast, 2022).

RTSP establece y controla uno o varios flujos de tiempo real entre un cliente y un servidor. Permite acciones como reproducir, pausar y detener la transmisión de medios, así como moverse adelante o atrás dentro del flujo de medios. A diferencia de otros protocolos de transmisión que envían datos directamente, RTSP actúa más bien como una "señal de control" para establecer y controlar la sesión de streaming, mientras que el transporte real de los datos se realiza típicamente mediante otros protocolos, como RTP (Real-time Transport Protocol) para la entrega de los datos y RTCP (Real-time Control Protocol) para el monitoreo de la entrega de datos en tiempo real (Dacast, 2022).

CAPÍTULO II

METODOLOGÍA DEL SISTEMA

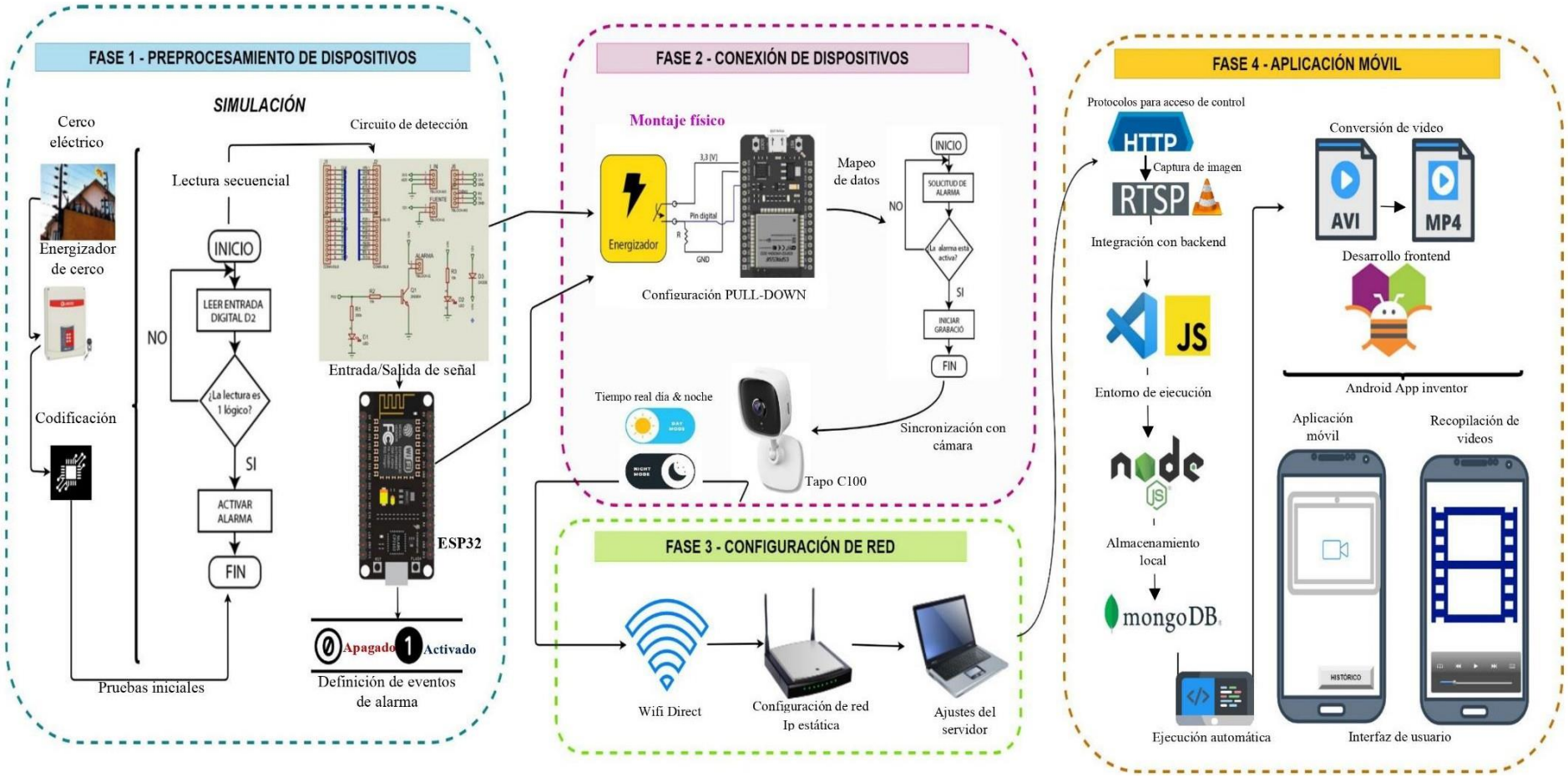
2.1.1. Sistema de detección de fallos en el perímetro de vivienda

El sistema de detección de fallos en el perímetro de vivienda se apoya en una variedad de dispositivos electrónicos, protocolos de comunicación y aplicaciones que permiten un funcionamiento confiable. La arquitectura del sistema utiliza de manera conjunta un elemento del cerco eléctrico y una cámara de video vigilancia para monitorear constantemente su estado. Se apoya también, la conexión de los dispositivos, tales como un circuito PULL-DOWN utilizando el contacto seco del energizador del cerco eléctrico, donde, una lectura de 3.3 voltios equivalente a "uno lógico" activa la alarma detectando una vulneración, por otro lado, si la lectura es cero voltios que equivale a "cero lógico" la alarma permanece en la espera de un cambio en la lectura. De manera similar el módulo envía el estado del cerco a través de una solicitud HTTP POST.

Por otra parte, el estado de la alarma inicia la grabación de la cámara de seguridad, generando un archivo con la extensión ".avi" la cual mediante un proceso de conversión se genera un archivo "MP4", todos los eventos generan datos históricos que se almacenan la base de datos MongoDB incluyendo los archivos de las grabaciones. En la **Figura 4** se puede observar la estructura del sistema de detección de vulneración del perímetro de seguridad.

Figura 4

Esquema del sistema de detección de fallos en el perímetro de vivienda



Nota. Esquema general de proyecto. Elaborado por: Los autores

2.1.2. Autenticación del sistema de detección de fallos en el perímetro de vivienda

En el marco de este proyecto, se ha introducido una metodología robusta que facilita la implementación de cada etapa del sistema de detección de fallo en cerco eléctrico. Estas etapas se categorizan en cuatro procesos clave: **Configuración de los dispositivos, Conexión de los dispositivos, Configuración de la Red y el Desarrollo de una Aplicación Móvil**. Cada una de estas fases contribuye al logro de un objetivo concreto, posibilitando así un sistema de detección de fallo que incluye el seguimiento del estado del cerco eléctrico en tiempo real y mediante datos históricos, tal como se ilustra en la **Figura 4**.

En la etapa inicial denominada. **Configuración de dispositivos**, se llevó a cabo la programación de los componentes electrónicos, definiendo su operación mediante el establecimiento de casos o eventos. En esta fase se realizó la programación estableciendo el estado de la alarma dependiendo de la lectura del pin digital correspondiente al estado del cerco eléctrico.

En la segunda fase denominada. **Conexión de los dispositivos**, mediante el modelo establecido se realiza la conexión física de los dispositivos involucrados en el sistema.

En la tercera fase. **Configuración de red**, se realizó la configuración del módulo ESP32 para conectarse a la red Wifi y al servidor para establecerse como intermediario entre la lectura del estado del cerco y la cámara de video vigilancia, con la base de datos y la aplicación móvil.

Finalmente, en la cuarta fase. **Aplicación móvil**, se realizó la configuración de la aplicación utilizando el protocolo HTTP, el cual permite a la aplicación acceder al estado del cerco eléctrico, así como al video de la cámara de seguridad.

En la elaboración de este proyecto, se describirán minuciosamente todas las etapas, explicando los procedimientos llevados a cabo para alcanzar los objetivos de cada una. Esto

con el fin de desarrollar un programa efectivo para sistema de detección de seguridad del perímetro de vivienda.

2.1.3. Desarrollo Fase 1: Configuración de dispositivos

Se creó un prototipo fundamental a base de los componentes previstos para la validación experimental de hipótesis, sosteniendo un diseño eficiente y efectivo para una aplicación doméstica inteligente. Los detalles visuales presentados se comprenden en la implementación práctica del diseño y la metodología subsecuente de operación.

Entorno para el sistema. Para el entorno del sistema se implementa un accionamiento utilizando un interruptor que emulará el funcionamiento del contacto seco del energizador de un cerco eléctrico. En la **Figura 5** se muestra una ilustración con el entorno de prueba utilizado para la validación del sistema

Figura 5

Entorno del sistema

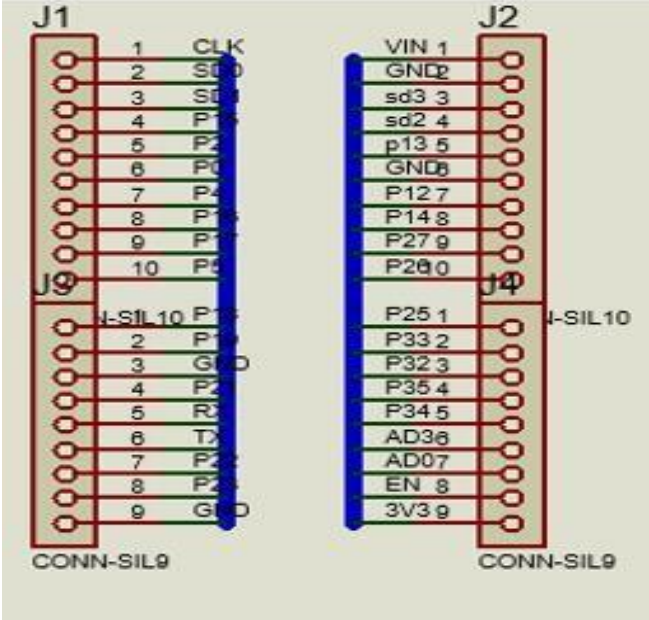
Nota. Cerco Eléctrico. Fuente: Lead. 2019.

Esquema general en la ESP32. Con la ayuda del software PROTEUS se realiza la determinación de los elementos electrónicos que conforman el sistema de detección de fallo, que serán montados a una baquelita para facilitar la fijación y distribución de estos, en la **Figura 6** se puede observar la distribución del circuito, y su diseño PCB.

La ilustración presenta el esquemático electrónico del sistema diseñado. En el corazón del diseño se encuentra el módulo ESP32, el cual actúa como el componente central de procesamiento y control. Este módulo está flanqueado por dos conectores de pines, J1 y J2, que facilitan la interconexión con otros dispositivos o módulos.

Figura 6

Placa para módulo ESP32

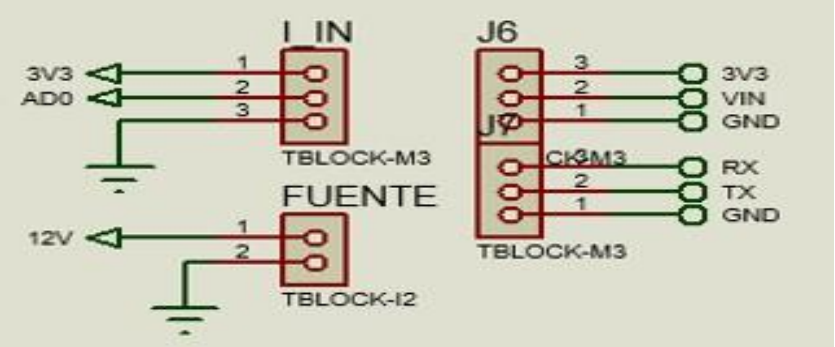


Nota. **J1 y J2:** Conector de pines Elaborado por: Los autores

Bloques de terminales. Se observan dos bloques de terminales, etiquetados como J6 y J7. El terminal J6 está destinado a la conexión de una fuente de alimentación externa, mientras que J7 es una interfaz para la comunicación de datos en el módulo ESP32, como se muestra en la **Figura 7**.

Figura 7

Terminal de bloques



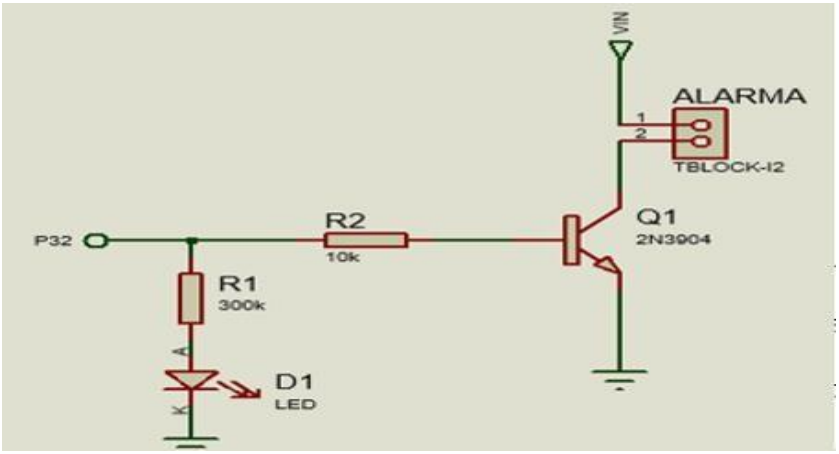
Nota. **J6 y J7:** Conexión a una fuente de alimentación externa

Elaborado por: Los autores

Control de cargas. Al lado del módulo ESP32, el transistor Q1, es un componente clave para la amplificación de señal y actuación del interruptor, la parte inferior del diagrama sitúa un LED D1, que está en serie con una resistencia R1. Este conjunto sirve como un indicador adicional dentro del circuito, para notificar que el sistema está energizado, permitiendo al control de cargas requerir más potencia de la que el ESP32 puede manejar directamente, como se muestra en la **Figura 8**.

Figura 8

Amplificación de señal



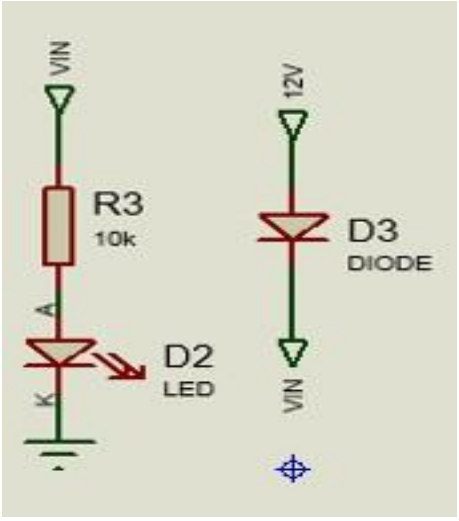
Nota. **R1:** Resistencias, **Q1:** Control de cargas, **D1:** Led.

Elaborado por*:* Los autores

Indicador de estado. Debajo del transistor de la **Figura 9**, se realiza un conjunto compuesto por una resistencia R3 y un diodo emisor de luz D2, funcionando como indicador de estado e iluminándose para señalar la activación del transistor, por consiguiente, el estado de alarma.

Figura 9

Estado de alarma



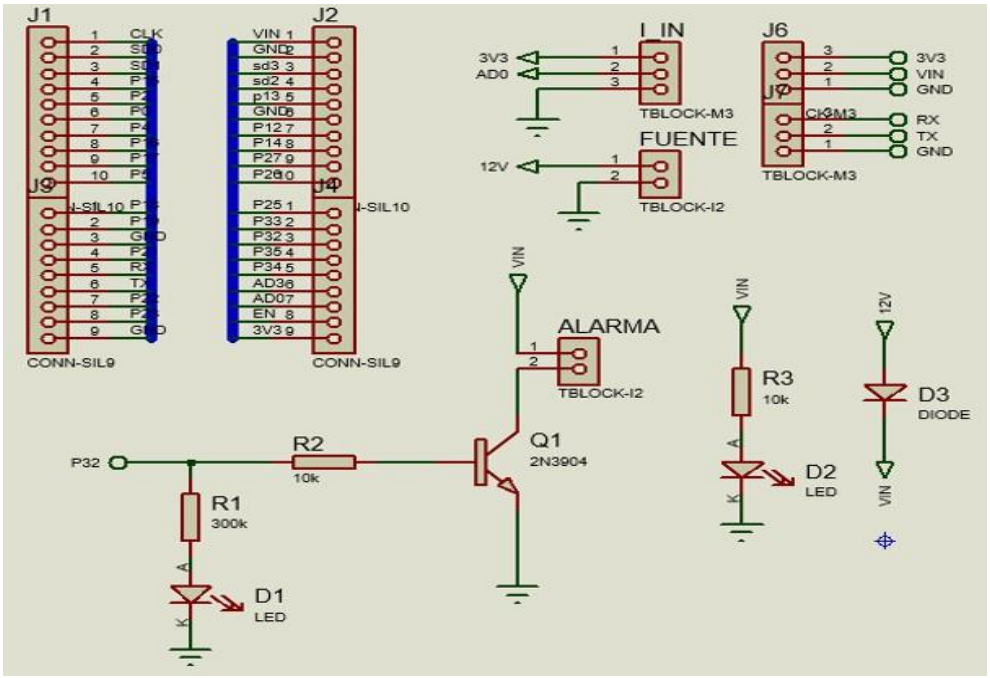
Nota. **R3:** Resistencia, **D2 y D3:** Leds

Elaborado por: Los autores

Esquema general. Los trazos que interconectan los componentes representan las conexiones eléctricas, y la orientación de los símbolos indica la dirección de flujo de la corriente. En la esquina inferior derecha del esquemático, se encuentra un símbolo de tierra, que indica un punto de referencia común para el retorno de corriente. Cada componente está meticulosamente etiquetado con su designación y valor, lo que proporciona información vital sobre su propósito y características dentro del circuito. Las conexiones ilustradas y los terminales de bornera facilitan la interacción con componentes externos, brindando un método sistemático para integrar y probar diferentes funcionalidades del diseño en la **Figura 10**.

Figura 10

Esquema del circuito para el sistema de detección de vulnerabilidad del cerco eléctrico



Nota. Diseño del sistema. Elaborado por: Los autores

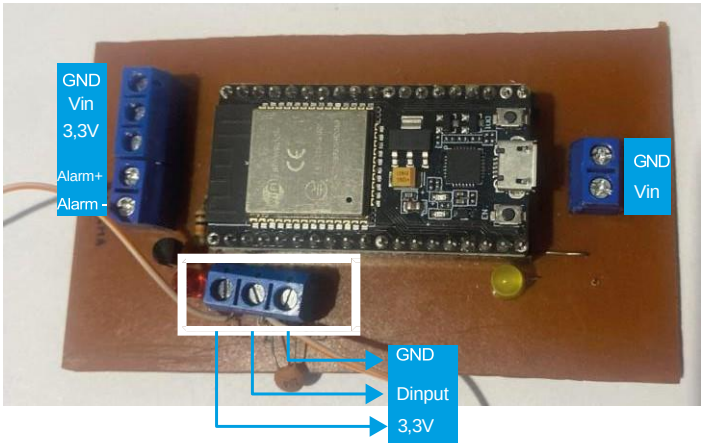
Diseño y elaboración de placa para módulo. La **Figura 11** muestra el ensamble físico del prototipo de circuito diseñado para la investigación. En el centro de la placa de baquelita se encuentra el módulo ESP32, que actúa como el cerebro del sistema permitiendo la conectividad Wi-Fi, crucial para las aplicaciones de IoT que son el foco de este proyecto. A la izquierda, se visualiza un bloque de borneras que facilitan la conexión de alimentación y señales externas, detalladamente etiquetadas para indicar su propósito: GND para tierra, Vin para la entrada de voltaje y una salida regulada de 3,3V. Un LED amarillo, situado en la esquina inferior derecha, sirve como indicador visual del estado operativo del sistema. En la esquina superior derecha, otro bloque de borneras proporciona una interfaz para la alimentación directa del módulo ESP32.

La inserción ampliada muestra una resistencia, parte de la red de componentes pasivos que apoyan la operación del microcontrolador. Las líneas azules representan las conexiones propuestas del prototipo, ilustrando cómo se integra el GND, la entrada digital (Dinput) y la alimentación de 3,3V con el resto del sistema.

Este prototipo es fundamental para la validación experimental de la hipótesis de esta tesis, que sostiene que un diseño eficiente y de bajo costo puede ser efectivo en aplicaciones domésticas inteligentes. Los detalles visuales de esta figura son cruciales para comprender la implementación práctica del diseño y la metodología de pruebas subsecuentes.

Figura 11

Circuito conectado



Nota. Conexión del sistema. Elaborado por: Los autores

2.1.4. Desarrollo fase 2: Desarrollo e implementación del sistema

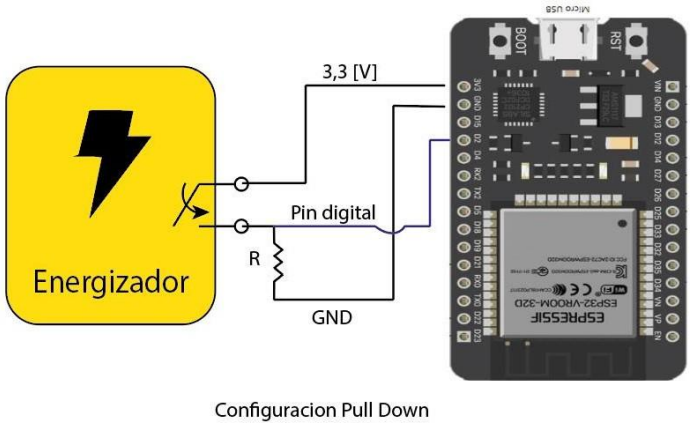
Dentro de este marco se configura tanto hardware y software para detectar los cambios de estado en un cerco eléctrico y responda automáticamente a este cambio.

Módulo ESP32. La conexión para la tarjeta ESP32 incluye el pin digital (D36) que se encuentra bajo una lectura en configuración PULL-DOWN, el circuito detecta un estado de encendido si el contacto seco del energizador se cierra, y en estado apagado cuando el contacto

seco se encuentra abierto. En la **Figura 12** se puede observar la disposición del circuito PULL-DOWN para la lectura del pin digital D2.

Figura 12

Esquema de detección del estado del cerco



Nota. Este esquema permite conocer estado del cerco.

Elaborado por: Los autores.

A continuación, en el **Pseudocódigo 1** se configura inicialmente una entrada digital en configuración Pull Down para conocer el estado del cerco eléctrico, con fines didácticos se considera una salida digital (LED) que se lo utilizará más adelante, esta configuración se puede observar.

Pseudocódigo 1

Configuración de entradas y salidas del ESP32

Pseudocódigo 1. Configuración de entradas y salidas del ESP32

Input: Lectura del estado del cerco eléctrico mediante el sensor de estado (Cerco_state)
Output: Estado de la alarma asociada al estado del cerco (estadoAlarma)

```
01: OUT_alar_aux = Pin(2, Pin.OUT)
02: OUT_alar_aux.off()

03: Cerco_state = Pin(36,Pin.IN)
04: Cerco_state.PULL_DOWN
```

Donde:
Input: La entrada es la señal del estado del cerco eléctrico, que se detecta mediante el pin digital configurado como entrada.
Output: La salida es el estado de la alarma, que puede ser 'activada' o 'desactivada'. Esto se representa mediante una variable 'estadoAlarma' que indica si la alarma está activa = '1' o desactivada = '0'.
OUT_alat_aux: define el pin 2 como un pin de salida
OUT_alar_aux.off(): Inicializa el pin 2 en estado “apagado”
Cerco_state: Define el pin 36 como un pin de entrada
Cerco_state.PULL_DOWN: Configura la lectura del pin digital como cero lógico a cero voltios, y uno lógico a la lectura de 3.3 voltios

Nota. Este esquema permite conocer estado del cerco

Elaborado por: Los autores.

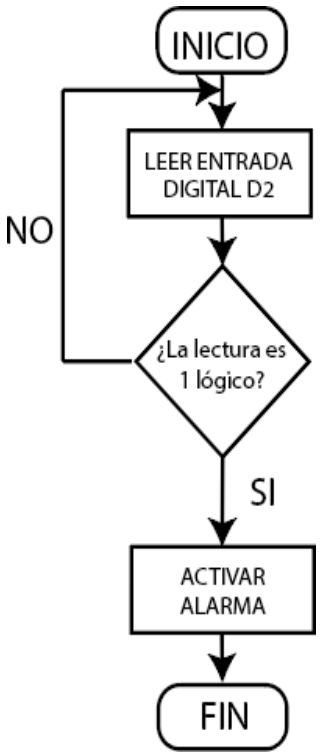
Detección de vulnerabilidades. La **Figura 13** describe el procedimiento automatizado implementado para la activación de una alarma en respuesta a la señal de entrada digital D2. El proceso se inicia con la puesta en marcha del sistema, seguido de la fase de monitoreo en la que se lee continuamente la señal de la entrada D2. Si la señal es reconocida como '1 lógico', indicando una condición predefinida, el sistema procede a activar la alarma.

Este mecanismo de decisión y respuesta es esencial para asegurar una reacción inmediata ante condiciones específicas, siendo un elemento crítico en sistemas de control y seguridad automatizados. La simplicidad del proceso no resta importancia a su funcionalidad, reflejando la eficiencia del diseño del sistema. Este diagrama ilustra una parte fundamental de la

investigación llevada a cabo, demostrando la aplicación práctica de los conceptos de automatización y control en un entorno real.

Figura 13.

Diagrama de flujo del sistema de detección de vulneración del cerco eléctrico



Nota. Diagrama general del proceso de lectura mediante un pin digital del estado del cerco eléctrico. Elaborado por: Los autores

Cámara de seguridad Tapo C100. La **Figura 14** detalla el protocolo establecido para iniciar un proceso de grabación en respuesta a una solicitud de activación de alarma. El flujo comienza con la etapa inicial de la solicitud de alarma, seguido por la evaluación continua del estado de la alarma. Si se determina que la alarma está activa, el sistema procede a iniciar la grabación. Este procedimiento es esencial para documentar los eventos que ocurren durante y después de la activación de una alarma, lo que es crucial para los análisis de seguridad.

La cámara de seguridad inicialmente se encontrará en estado de espera únicamente enviando la imagen en tiempo real, en caso de presentarse una vulneración de seguridad, el estado de alarma enviado al servidor genera un requerimiento hacia la cámara de seguridad para que se inicie la grabación. Al finalizar la secuencia de acciones, el proceso llega a su fin. Este diagrama remarca la importancia de la automatización en la seguridad y recopilación de evidencia del mecanismo de respuesta implementado en el sistema de seguridad.

Figura 14

Diagrama de flujo para la grabación de video de la cámara de seguridad

Nota. Diagrama de flujo correspondiente al accionar de la cámara de seguridad. Elaborado por: Los autores

2.1.5. Desarrollo fase 3: Configuración de red

Esta etapa crea un entorno de red adecuado para la comunicación efectiva entre los componentes del sistema tanto para dispositivos locales como para aquellos conectados a través de Wi-Fi.

Configuración de la red local mediante Router Wi-Fi. En la **Figura 15**, se muestra el router, para gestionar el ancho de banda y priorizar el tráfico de la cámara con el servidor

Figura 15

Router Tp-Link

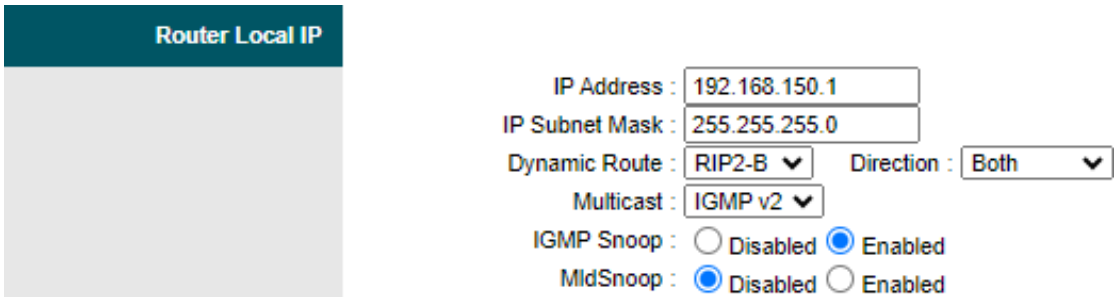
Nota. Router Tp Link. Fuente: Wireless. 2019.

Configuración de IP del router. La **Figura 16** ilustra la configuración del router que forma parte de la infraestructura de red necesaria para el proyecto. La red seleccionada para conectar los dispositivos es la 192.168.150.0/24, en donde se ha configurado intencionalmente el tercer octeto como 150 para diferenciar esta red de otras posibles subredes que podrían coexistir en un entorno con múltiples puntos de acceso, como sería el caso de un área residencial.

La dirección IP del router se ha establecido en 192.168.150.1 con una máscara de subred de 255.255.255.0, confirmando su operación dentro de la subred escogida. Además, se ha habilitado el servidor DHCP, comenzando la asignación de direcciones IP en 192.168.150.100 y limitando el pool a 50 direcciones IP, lo que permite una distribución automática y gestionada de las direcciones IP a los dispositivos conectados.

Figura 16

Detalles de IP local

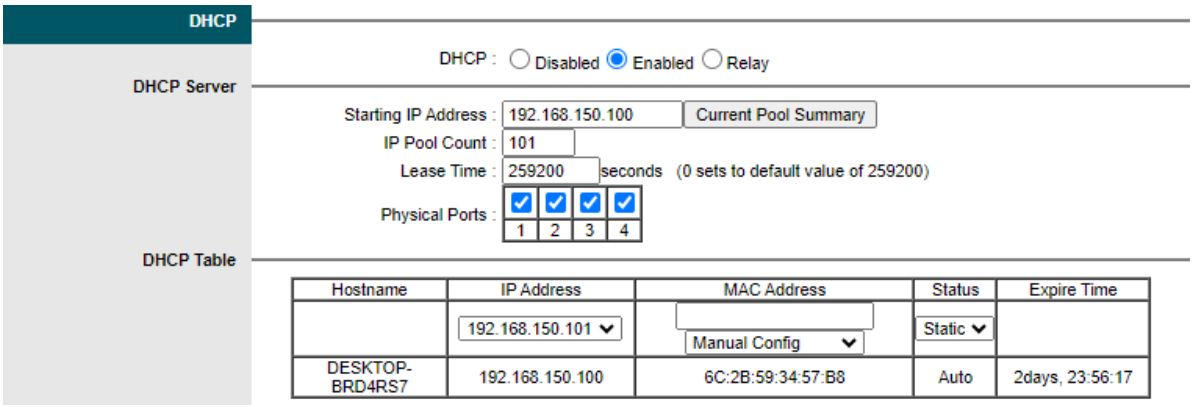


Nota. Planificación de red. Elaborado por: Los autores

Configuración del servidor DHCP. La **Figura 17** muestra el tiempo de concesión por defecto para las direcciones IP asignadas por DHCP es de 259200 segundos, o 3 días, tras los cuales los dispositivos necesitan renovar su dirección IP. La tabla DHCP expone dos dispositivos que ya están conectados a la red: uno con una dirección IP estática, 192.168.150.101, y otro con una dirección IP asignada dinámicamente, 192.168.150.100. Para cada dispositivo conectado, se proporciona la dirección MAC, el estado de la asignación de IP (estática o dinámica), y el tiempo restante para la expiración de la concesión.

Figura 17

Gestión de dirección IP del DHCP

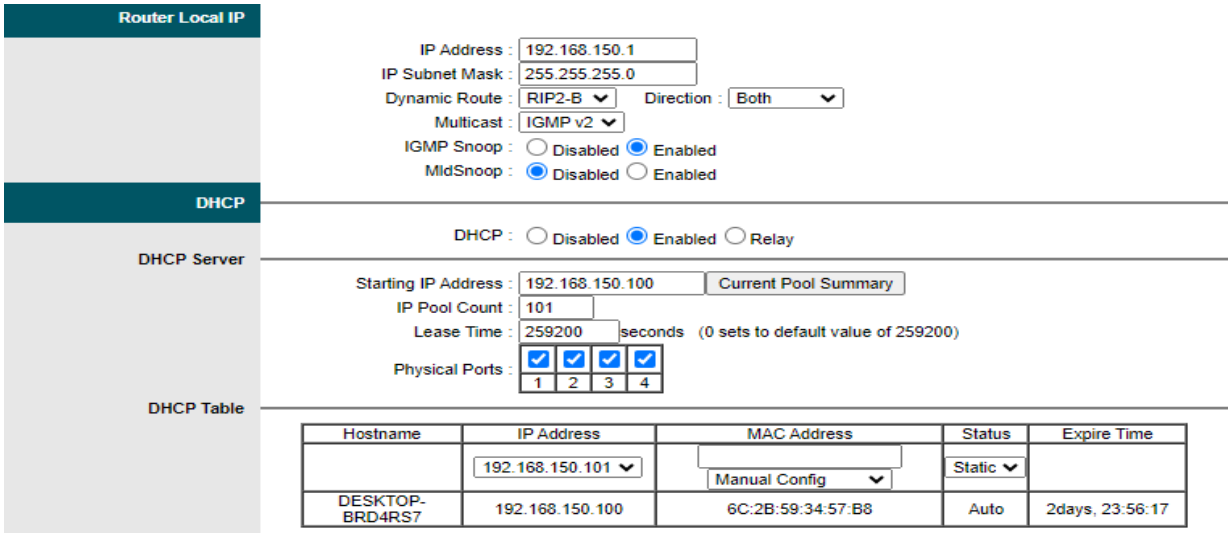


Nota. Resumen de configuración DHCP. Elaborado por: Los autores

Configuración LAN. En la **Figura 18** se muestra la configuración general para toda la red.

Figura 18

Configuración de la red LAN



Nota. Configuración del router y gestión de red. Elaborado por: Los autores

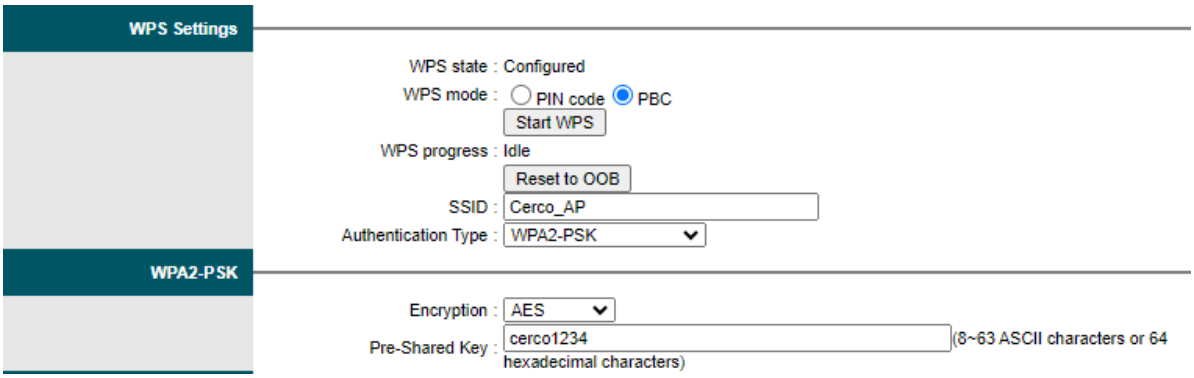
Configuración de la red WIFI. La **Figura 19** muestra la sección de configuración de seguridad dentro de la interfaz de administración de un router, específicamente en la sección de Wi-Fi Protected Setup (WPS). Se puede observar que el estado de WPS está configurado y se ha optado por el modo PBC (Push Button Configuration), lo cual es una de las formas de conectar dispositivos a la red Wi-Fi de manera segura con solo presionar un botón, facilitando la conexión sin necesidad de ingresar manualmente las credenciales.

La red Wi-Fi que se está configurando llevará el nombre de SSID (Service Set Identifier) "Cerco_AP". Este identificador es el nombre de la red que verán los dispositivos al escanear las redes Wi-Fi disponibles. Para la seguridad de la red, se ha seleccionado el tipo de autenticación WPA2-PSK, que es actualmente uno de los estándares de seguridad más robustos para redes Wi-Fi domésticas y pequeñas oficinas, proporcionando una buena protección contra el acceso no autorizado.

Para la encriptación, se ha elegido AES (Advanced Encryption Standard), que es recomendado por su fortaleza y eficiencia en proteger la comunicación inalámbrica. El Pre-Shared Key (PSK) o clave de acceso compartida, que es esencialmente la contraseña de la red Wi-Fi, se ha establecido como "cerco1234". Aunque se ha optado por una contraseña relativamente simple por razones prácticas, en un entorno real se recomendaría una clave más compleja para incrementar la seguridad de la red. Esta configuración, mostrada en la **Figura 19**, es esencial para garantizar que el router y la red Wi-Fi asociada sean seguros y estén protegidos contra vulneraciones, limitando el acceso a usuarios autorizados y asegurando la integridad de la comunicación dentro de la red.

Figura 19

Configuración de la red Wifi

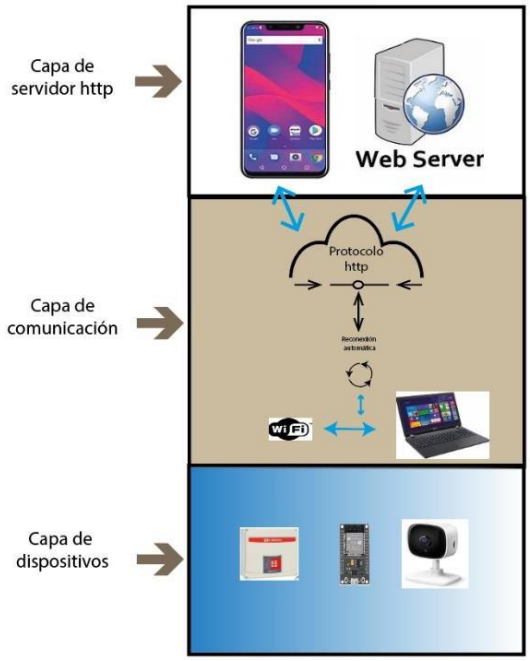


Nota. Configuración del router y gestión de red. Elaborador por: Los autores

Topología IoT. La conexión se establece en topología de estrella como se muestra en la **Figura 20** representa una estructura cliente-servidor, donde el punto central es el servidor levantado en el computador local. El servidor maneja el flujo de datos desde el sistema de detección de vulneración del cerco, así como las acciones de la cámara de video vigilancia, todo esto mediante el protocolo http.

Figura 20

Topología IoT



Nota. Conexión en topología del cliente-servidor. Elaborador por: Los autores

Conexión a internet. Ya que el dispositivo pretende conectarse tanto en una red LAN sin acceso a internet o con internet, este se lo deja en la configuración por defecto es decir como IP dinámica, en el caso de necesitar conectarse a internet, se debe conectar al puerto WAN el servicio de internet.

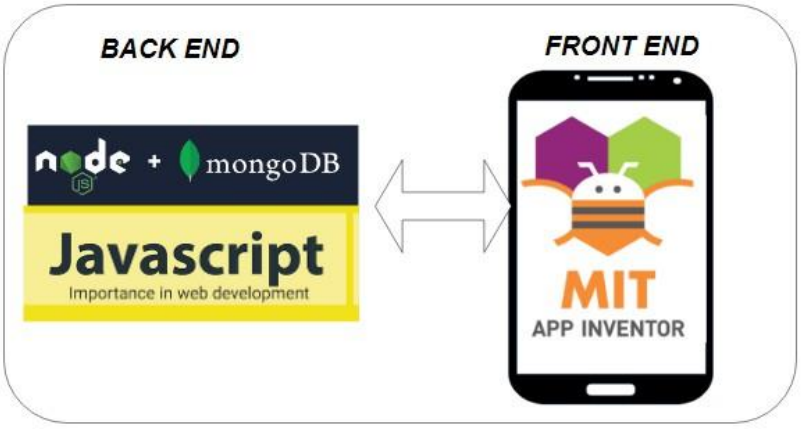
2.1.6. Desarrollo fase 4: Aplicativo

La elaboración de software, se desarrolló utilizando Node.js para el manejo conexiones asíncronas y procesos en tiempo real. A través de la **Figura 21**, con JavaScript se programaron funciones específicas para la gestión de formatos de vídeo, facilitando la grabación y reproducción de contenido de vigilancia. Además, se desarrollaron funcionalidades personalizadas para la conversión y manejo de videos, permitiendo una integración fluida con sistemas de almacenamiento.

Para Frontend, se seleccionó MIT App Inventor debido a su metodología intuitiva basada en bloques visuales. Este entorno de desarrollo permitió crear una aplicación móvil para dispositivos Android. La aplicación facilita la visualización en tiempo real de la cámara, el acceso a grabaciones históricas y recepción de alertas inmediatas ante cualquier incidencia.

Figura 21

Arquitectura global



Nota. Integración de Node.js y JavaScript con APP Inventor. Elaborado por: Los autores

En la **Figura 22** se representa un diagrama de flujo de la arquitectura del sistema de manejo de datos utilizado por un servidor. Agregando a lo anterior, se puede apreciar cómo el servidor etiquetado como SERVER NODE, actúa como el núcleo central de procesamiento y enrutamiento de la información. Este servidor gestiona las solicitudes de datos que se transmiten desde un sistema doméstico hacia una aplicación móvil, permitiendo la interacción y monitoreo remoto.

Funcionalidades del servidor. En relación al SERVER NODE las múltiples funciones dentro del sistema es la conversión de archivos de video de formato .avi a .mp4, lo que sugiere una funcionalidad de adaptación de medios para garantizar la compatibilidad con diferentes dispositivos y plataformas. Además, la gestión de páginas web, facilita la entrega de contenido

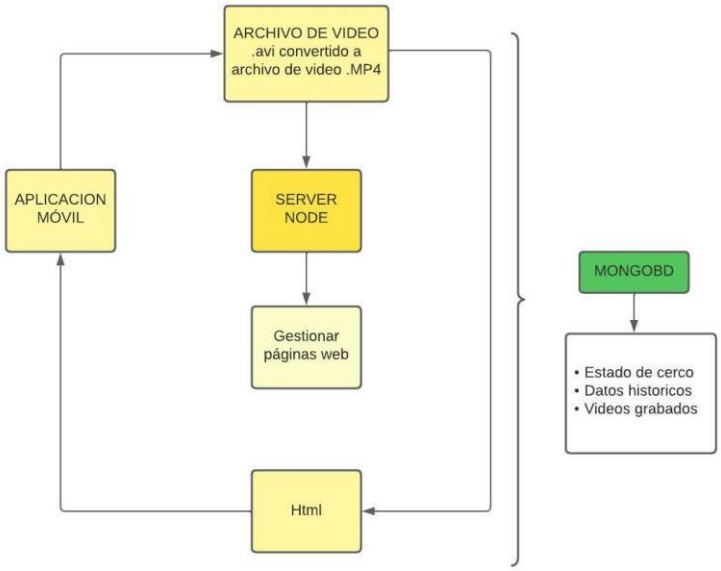
por HTML a los clientes, permitiendo a los usuarios acceder a interfaces web para visualizar y gestionar el estado de su sistema de seguridad en casa.

Registro de actividades. De manera similar el servidor también se comunica con una base de datos MongoDB, conocida por su capacidad para gestionar grandes volúmenes de datos de manera estructurada. En este caso, la base de datos almacena información como el estado del cerco de la vivienda, datos históricos y videos grabados. Esto indica que el sistema no solo realiza un seguimiento en tiempo real, sino que también mantiene registros para consultas y análisis posteriores.

Finalmente, la aplicación móvil recibe los datos procesados y convertidos en el servidor, lo que permite al usuario final tener control y visibilidad del sistema de seguridad de su hogar en cualquier lugar y momento a través de su dispositivo móvil.

Figura 22

Manejo de datos del servidor



Nota. Manejo de datos para la aplicación móvil. Elaborado por: Los autores

Gestor de solicitudes HTTP tipo POST con MongoDB. Para la solicitud, en el **Pseudocódigo 2** se recogen datos del cliente mediante una solicitud POST en configuración de enlaces web, incluyendo detalles como el estado del cerco, alarma, y una fecha específica. Estos se transmiten a MongoDB tras establecer una conexión. La información se inserta a la base de datos con una marca de tiempo actual, asegurando una actualización precisa del estado y el seguimiento eficiente.

Pseudocódigo 2

Código en JavaScript para gestión de solicitudes tipo POST

Pseudocódigo 2. Código en JavaScript para gestión de solicitudes tipo POST

Input: Datos de estado de cerco y alarma recibidos en POST
Output: Confirmación en la inserción de datos en MongoDB y respuesta HTTP al cliente

```
1. app.post('/put', async (req, res) => {
2.     const { cerco_id, Estado_cerco, Estado_alarma, Fecha } = req.body;
3.     alarma_st = Estado_alarma;
4.     cerco_st = Estado_cerco;
5.     Fecha_evento = fecha_actual.fecha_string();
6.     const client = new MongoClient(url);
7.
8.     try {
9.         await client.connect();
10.        const db = client.db(dbName);
11.        const collection = db.collection(collectionName);
12.        //envio de datos de cerco
13.            const result = await collection.insertOne({ cerco_id, Estado_cerco, Estado_alarma, Fecha_evento });
14.
15.        const data = await collection.findOne({Fecha_evento: Fecha_evento})
16.        console.log("Mongo Id: "+data._id.toString())
17.        //envio de datos CPU
18.        const cpu_var = require('./controllers/CPU_perf.js').cpu_perf();
19.        const Memoria_Ram_MB = cpu_var[0];
20.        const Tiempo_user_mode_ms = cpu_var[1];
21.                                                const cpucollection = db.collection(process.env.MONGO_SECOND_COLLECTION_NAME);
22.
23.                    const resultcpu = await cpucollection.insertOne({Memoria_Ram_MB, Tiempo_user_mode_ms})
24.        res.status(200).send('Datos insertados correctamente');
25.    } catch (error) {
26.        console.error('Error al conectar con la base de datos:', error);
27.        res.status(500).send('Error al procesar la solicitud');
28.    }});
```

Donde,
1. Definición de ruta '/put' para manejar solicitudes POST asincrónicas.
2. Extracción de datos del cuerpo de la solicitud POST, incluyendo identificación, estados del cerco y alarma actualizados en variables globales.
3. Se establece una conexión con MongoDB para insertar los datos recibidos.
4. Se insertan los datos del cerco y alarma en una colección de MongoDB, incluyendo una marca de tiempo del evento.
5. Se recopilan datos de rendimiento del CPU y se insertan en otra colección de MongoDB destinada a este fin.
6. Envío de respuesta HTTP al cliente indicando el éxito o fallo de la operación.

Nota. Gestión de Datos y Monitoreo de Sistemas. Elaborado por: Los autores

Conversión de video. El **Pseudocódigo 3** muestra la automatización del proceso de conversión de videos .avi a mp4 desde el entorno Node.js. Se incorpora la configuración de variables de entorno y una función para generar nombres de archivo basados en la fecha y hora actuales. La conversión se maneja con eventos de progreso y errores, optimizando la gestión de los archivos convertidos.

Pseudocódigo 3

Conversión de video .avi a mp4

Pseudocódigo 3. Conversión de video .avi a mp4

```
1  const hbjs = require('handbrake-js')
2  const dotenv = require('dotenv')
3  const fs = require('fs')
4  dotenv.config()
5
6  const fecha_actual ={
7    fecha_string(){
8      const now = new Date();
9      const anio = now.getFullYear().toString();
10     const mes = now.getMonth()+1;
11     const mes2 = mes.toString();
12     const dia = now.getDate().toString();
13     const hora = now.getHours().toString();
14     const min = now.getMinutes().toString();
15     const fecha = anio+'-'+mes2+'-'+dia+'T'+hora+'h'+min+'min';
16     return fecha
17   }
18 }
19
```

```
exports.convert_video=function(archivo_video,callback){
  const video_path_output = './public/eventVideo'+fecha_actual.fecha_string()+'.mp4';
  console.log(video_path_output)
  hbjs.spawn({ input: process.env.VIDEO_PATH+'\\'+archivo_video, output: './public/video.mp4' })
    .on('error', err => {
      console.log("invalid user input, no video found etc");
    })
    .on('progress', progress => {
      console.log('Conversión de video al: '+progress.percentComplete+'%')
      if(progress.percentComplete === 100){
        console.log('Conversión 1 completada')
      }
    })
  setTimeout((tiempo)=>{},1000)
        hbjs.spawn({ input: process.env.VIDEO_PATH+'\\'+archivo_video, output: video_path_output})
      .on('error', err => {
        console.log("invalid user input, no video found etc",err);
        callback(1,null)
      })
      .on('progress', progress => {
        console.log('Conversión de video al: '+progress.percentComplete+'%')
        if(progress.percentComplete === 100){
           console.log('Conversiones completadas!')
           callback(null,video_path_output)
        }
      })
}
```

Donde,

Input: Requisición de módulos y manejo de variables de entorno.

Output: Se exporta y ejecuta la función de conversión de video, se manejan eventos de progreso y errores, y se retorna la ruta del video convertido.

1. Se importa módulos 'handbrake-js', 'dotenv', y 'fs' para conversión de videos, gestión de variables de entorno y manipulación de archivos.
2. Se ejecuta la configuración de 'dotenv' para acceder a variables de entorno desde archivo. env.
3. Se declara 'fecha_actual' con método 'fecha_string' para generar fecha y hora actuales.
4. La función 'convert_video' inicia proceso de conversión de video con parámetros y callback.
5. Se establece ruta de salida para video convertido incluyendo fecha y hora en el nombre.
6. Se utiliza 'handbrake-js' para convertir video, manejando eventos de error y progreso.
7. La creación en el tiempo de espera es de 1000 milisegundos entre conversiones de video.
8. Se gestiona errores y muestra progreso de conversión en consola.
9. Finaliza conversión de video, indicando su completitud con mensaje en consola.

Nota. Conversión de video .avi a mp4. Elaborado por: Los autores

Solicitudes Tipo GET para ingresar a las páginas HTML. En el **Pseudocódigo 4** se muestra el uso de solitudes de tipo GET es para mostrar las páginas HTML, en ellas se presenta los datos necesarios para el control e información del estado del sistema de seguridad.

Pseudocódigo 4

Código en JavaScript para gestión de solicitudes tipo GET

Pseudocódigo 4. **Código en JavaScript para gestión de solicitudes tipo POST**

```
app.get('/last_video',(req,res)=>{
  res.render('index.ejs')
});

app.get('/',(req,res)=>{
  let fecha = fecha_actual.fecha_string();
  res.render('main.ejs',{alarma: alarma_st, estado: cerco_st, fecha: fecha})
});

app.get('/video',(req,res)=>{
  res.render('video_stream.ejs',{estado:  alarma_st});
})
app.get('/video_hist',async  (req,res)=>{
    const datos = await require('./controllers/read_mongo.js').read_hist_video();
                                    res.render('video_hist.ejs',{data:  datos,  Path:
process.env.VIDEO_PATH+'\\event_videos',estado: alarma_st})
})
app.get('/show_video_hist.ejs',  (req,res)=>{
  const datos = req.query.video_name
  res.render('show_video_hist.ejs',{data:  datos})
})
```

Donde,

Input: Establecer solicitudes HTTP GET a diferentes rutas de una aplicación Express.js.

Output: Conexión a páginas HTML renderizadas del lado del servidor enviadas al cliente.

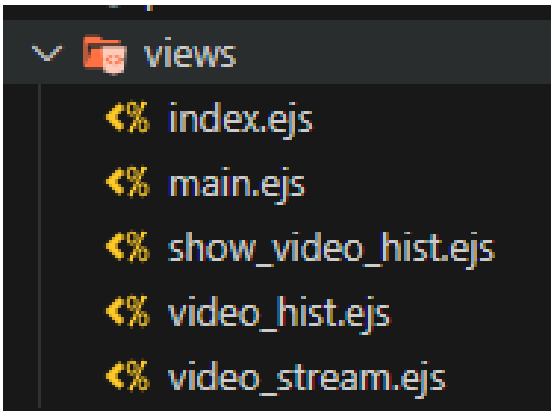
1. Al acceder a '/last_video', se renderiza la última plantilla 'index.ejs' con el último video publicado.
2. La ruta raíz '/' muestra la plantilla 'main.ejs' con información general de la fecha actual y cambio de evento.
3. A través de la ruta '/video' se muestra 'video_stream.ejs' con el estado de alarma actual, para ver transmisiones de video en vivo o recientes.
4. La ruta '/video_hist' obtiene datos históricos de video de MongoDB y muestra 'video_hist.ejs' con sus datos y ruta específica
5. Se define la ruta '/video' en el manejo de solicitudes para mostrar videos históricos seleccionados, mostrando 'show_video_hist.ejs' con la información correspondiente al video seleccionado.

Nota. Gestión de contenido de video en aplicación web. Elaborado por: Los autores

Plantillas EJS. Dentro de este contexto, las páginas HTML ocupados para el proyecto NodeJS que usan “express” son archivos de tipo. ejs que permiten el uso de lenguaje JavaScript dentro de una sintaxis propia de express. (<% “código en JavaScript” %>), con la intención de enviar datos o realizar paginas dinámicas. Estos archivos deben estar guardados en una carpeta llamada views dentro del proyecto como se muestra en la **Figura 23**.

Figura 23

Organización de archivos de un proyecto en NodeJS con express



Nota. Implementación de plantillas ejs para páginas dinámicas. Elaborador por: Los autores

Gestión de controladores. Dentro del manejo de video, se utiliza códigos en JavaScript adicionales para guardarlos en una carpeta llamada controllers ya que estos llevan la lógica de control.

Considerando que no se puede saber exactamente cuál sería el nombre del video que se guardó una vez el software VLC media Player haya terminado la grabación, es necesario implementar un código que me permita obtener el último video grabado en el lugar donde este los guarda, el **Pseudocódigo 5** realiza dicha acción y devuelve la dirección completa del video.

Pseudocódigo 5

Inicio de la grabación en VLC Media Player

Pseudocódigo 5. Inicio de la grabación en VLC Media Player

```
1  const robot = require('robotjs');
2
3  exports.press_key = function(){
4      robot.keyToggle('shift', 'down');
5      robot.keyTap('r');
6      robot.setKeyboardDelay(1000);
7      robot.keyToggle('shift', 'up');
8  }
```

Donde,

Input: Ninguno explícito, la función se invoca para realizar una acción específica.

Output: La simulación de la presión de teclas en el teclado.

1. Importa el módulo 'robotjs' que permite controlar el mouse y el teclado.
2. Define y exporta la función 'press_key', la cual ejecuta una secuencia de acciones en el teclado.
3. Presiona y mantiene la tecla 'shift' hacia abajo.
4. Simula la pulsación de la tecla 'r' mientras 'shift' está presionado.
5. Establece un retraso de 1000 milisegundos (1 segundo) para la siguiente acción del teclado.
6. Libera la tecla 'shift', volviéndola a su estado original (no presionado).

Nota. Control automatizado de VLC Media Player con RobotJS. Elaborado por: Los autores

Base de datos en MongoDB. El **Pseudocódigo 6** muestra la colección de los archivos de video guardados, para el registro histórico de los videos capturados cuando ocurre un evento dentro del sistema de seguridad. En este caso se pretende escribir el video en la base de datos, usando una librería propia de MongoDB llamada GridFSBucket, que crea dos colecciones una donde se guarda el vídeo y el segundo que guarda las características del video.

Pseudocódigo 6

Lectura de base de datos en MongoDB

Pseudocódigo 6. **Lectura de base de datos en MongoDB**

```
1  const { MongoClient } = require('mongodb');
2  const dotenv = require('dotenv'); 3
4  dotenv.config();
5  let fecha = new Array();
6  let file_name = new Array();
7  exports.read_hist_video = async function(){
8      const client = new MongoClient(process.env.MONGO_URL);
9      await client.connect();
10      const db = client.db(process.env.MONGO_DB_NAME);
11                                                    const          collection          =
db.collection(process.env.MONGO_COLLECTON_VIDEO_NAME);
12      try {
13          const docs =  await collection.find({}).sort({uploadDate : -1 }).toArray();
14          docs.forEach(doc => {
15              fecha.push(doc.uploadDate);
16              file_name.push(doc.filename);
17          });
18      } finally {
19          await client.close(); }
20      return [fecha, file_name];}
```

Donde,
Input: El nombre del último archivo de video.
Output: Conexión con la base de datos MongoDB y escritura del archivo de video en la base de datos utilizando GridFS si el archivo existe.

1: Importa el módulo 'fs' para operaciones del sistema de archivos.
2: Importa 'MongoClient' y 'GridFSBucket' de 'mongodb' para interactuar con MongoDB.
3: Importa 'dotenv' para cargar variables de entorno desde un archivo. env.
4: Ejecuta la configuración de 'dotenv'.
5: Recupera la URL de conexión a MongoDB desde las variables de entorno.
6: Recupera el nombre de la base de datos de MongoDB desde las variables de entorno.
7: Define y exporta la función principal 'main'.
8: Crea una instancia de 'MongoClient' con la URL de la base de datos.
12: Intenta conectar al cliente de MongoDB.
13: Registra en consola la conexión exitosa a la base de datos.
15: Obtiene la referencia a la base de datos específica.
16: Crea una instancia de 'GridFSBucket' para almacenar archivos grandes.
18: Muestra en consola el nombre del último archivo.
20-29: Si el último archivo no está vacío, lee el archivo de video y lo carga a MongoDB usando GridFS, luego cierra la conexión al finalizar.
33-37: Captura y maneja cualquier error durante el proceso de conexión o carga y cierra la conexión con la base de datos.

Nota. Control automatizado de VLC Media Player con RobotJS. Elaborado por: Los autores

Localización de videos. El **Pseudocódigo 7**, utiliza rutas en el sistema de archivos para filtrar y ordenar los archivos de video por fecha de modificación devolviendo el último evento generado. Este script es esencial para sistemas de seguridad que requieren un acceso rápido y confiable a las grabaciones más actuales para la revisión de eventos

Pseudocódigo 7

Obtención de la dirección del video grabado en el evento

Pseudocódigo 7 Obtención de la dirección del video grabado en el evento

```
1  const fs = require('fs');
2  const path = require('path');
3  const dotenv = require('dotenv');
4
5  dotenv.config();
6  exports.get_last_video = function(directorioVideos, callback){
7      fs.readdir(directorioVideos, (err, archivos) => {
8          if (err) {
9              console.error('Error al leer el directorio:', err);
10             callback(err, "");
11         }
12
13         const archivosDeVideo = archivos.filter(archivo => {
14             return archivo.endsWith('.avi');
15         });
16
17         const archivosOrdenados = archivosDeVideo.map(archivo => {
18             const rutaCompleta = path.join(directorioVideos, archivo);
19             return {
20                 nombre: archivo,
21                 fechaModificacion: fs.statSync(rutaCompleta).mtime.getTime()
22             };
23         }).sort((a, b) => b.fechaModificacion - a.fechaModificacion);
24
25         if (archivosOrdenados.length > 0) {
26             let ultimoArchivo = archivosOrdenados[0].nombre;
27             console.log('El archivo más reciente es:', ultimoArchivo);
28             callback(null, ultimoArchivo);
29         } else {
30             console.log('No se encontraron archivos de video en la carpeta proporcionada.');
31                     callback(new Error('No se encontraron archivos de video en la carpeta
proporcionada.'), "");
32         }
33     });
34 }
```

Donde,
Input: Un directorio especificado que contiene archivos de video.
Output: Se identifica y devuelve el nombre del archivo de video más reciente en el directorio, utilizando una función de callback para comunicar el resultado o un error.
1: Importa el módulo 'fs' para operaciones del sistema de archivos.
2: Importa el módulo 'path' para manejar y transformar rutas de archivos.
3: Importa 'dotenv' para cargar y usar variables de entorno.
4: Inicializa las variables de entorno configuradas en un archivo .env.
5: Define y exporta la función 'get_last_video', que busca el último archivo de video en un directorio.
6: Lee el contenido del directorio de videos y maneja errores en caso de que no pueda leerse.
7: Filtra los archivos leídos para encontrar solo aquellos con la extensión '.avi'. 17-23: Mapea cada archivo de video a un objeto que incluye su nombre y fecha de modificación, luego los ordena de más reciente a más antiguo.
8: Si hay archivos de video encontrados, selecciona el más reciente, lo imprime y lo devuelve a través de la función de callback.
9: Si no se encuentran archivos de video, informa a través de un mensaje en consola y devuelve un error mediante la función de callback.

Nota. Control automatizado de VLC Media Player con RobotJS. Elaborado por: Los autores

La **Figura 24** ilustra el diagrama de flujo general que maneja el sistema de seguridad de manera general. El proceso comienza con la inicialización, donde se establece la interconexión entre todos los dispositivos que conforman el sistema, asegurando una red integrada para la supervisión de seguridad.

Este diseño refleja una integración cohesiva entre las interfaces de usuario, el procesamiento del servidor y la persistencia de datos, clave para la eficiencia y eficacia del sistema de monitoreo remoto.

Tras la fase de inicio, el sistema procede a verificar el estado del cerco perimetral. En caso de detectar un posible evento de seguridad o una vulneración del cerco, el sistema activa una serie de respuestas: envía una alerta al teléfono celular vinculado, inicia la grabación de la cámara de seguridad para documentar el evento, y espera la notificación del dispositivo móvil para posiblemente desactivar la alarma o escalar la respuesta.

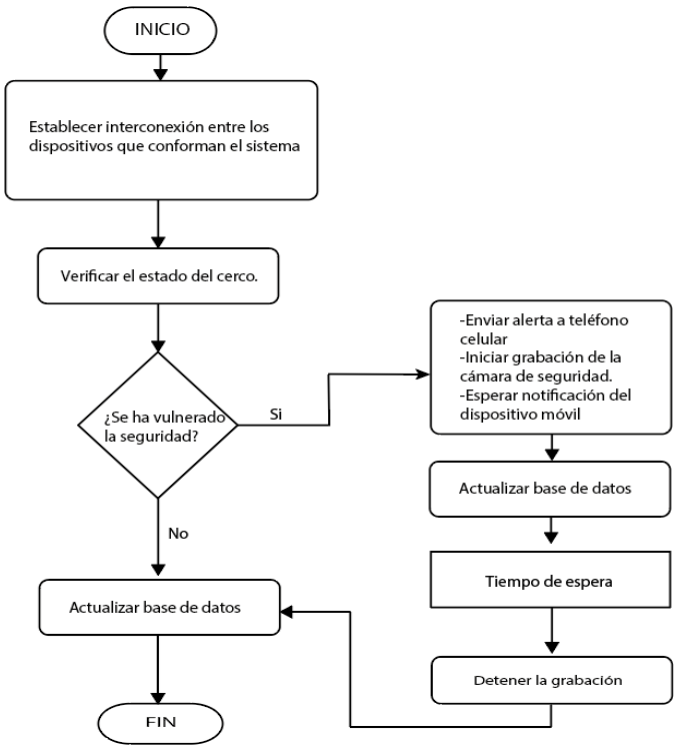
Independientemente de si se detecta una vulneración, el sistema actualiza la base de datos con la información más reciente. Si no se detecta una vulneración, el flujo se dirige directamente a esta actualización y luego concluye el ciclo.

En caso de que se haya emitido una alerta, después de actualizar la base de datos, el sistema entra en un estado de tiempo de espera. Este estado de espera es necesario para brindar el tiempo suficiente al usuario para verificar y responder a la alerta. Finalmente, el proceso termina con la detención de la grabación una vez que se resuelva la situación o después de que transcurra un intervalo de tiempo preestablecido.

Este diagrama de flujo es central para comprender la lógica operativa y la secuencia de respuestas del sistema de seguridad diseñado, demostrando su capacidad para una respuesta automatizada y documentada ante incidentes de seguridad.

Figura 24

Diagrama de flujo general del Sistema de seguridad



Nota. Flujo operativo y respuesta del sistema de seguridad. Elaborado por: Los autores

Interfaz principal de seguridad. La aplicación cuenta con dos pantallas que permiten visualizar diferentes funciones, en la pantalla principal se muestra el video en tiempo real de la cámara de seguridad, en caso de presentarse una alerta de vulnerabilidad de la seguridad en el perímetro de la vivienda, en la parte inferior se puede visualizar el estado del cerco, así como, una opción para acceder a los datos históricos, la pantalla principal se muestra en la **Figura 25**.

Figura 25

Primera pantalla de la aplicación del dispositivo móvil

Nota. Funcionalidades de monitoreo en tiempo real y acceso histórico en sistema de vigilancia.
Elaborado por: Los autores

Visualización de datos históricos. La segunda pantalla se expone en la **Figura 26** y esta muestra los datos históricos con la facilidad de acceder al video grabado correspondiente a un evento, mediante la opción “Read video” en área de visualización se muestra el ultimo video grabado por la cámara de videovigilancia.

Figura 26

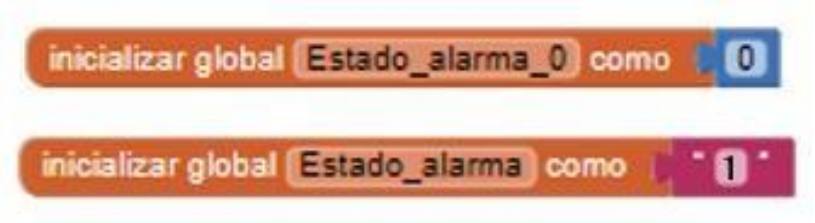
Segunda pantalla de la aplicación del dispositivo móvil

Nota. Acceso a videos grabados en el sistema de seguridad. Elaborado por: Los autores

Configuración de aplicación. Inicialmente se establecen variables globales (Estado_alarma_0, Estado_alarma 1), como se muestra en la **Figura 27**.

Figura 27

Definición de variables globales

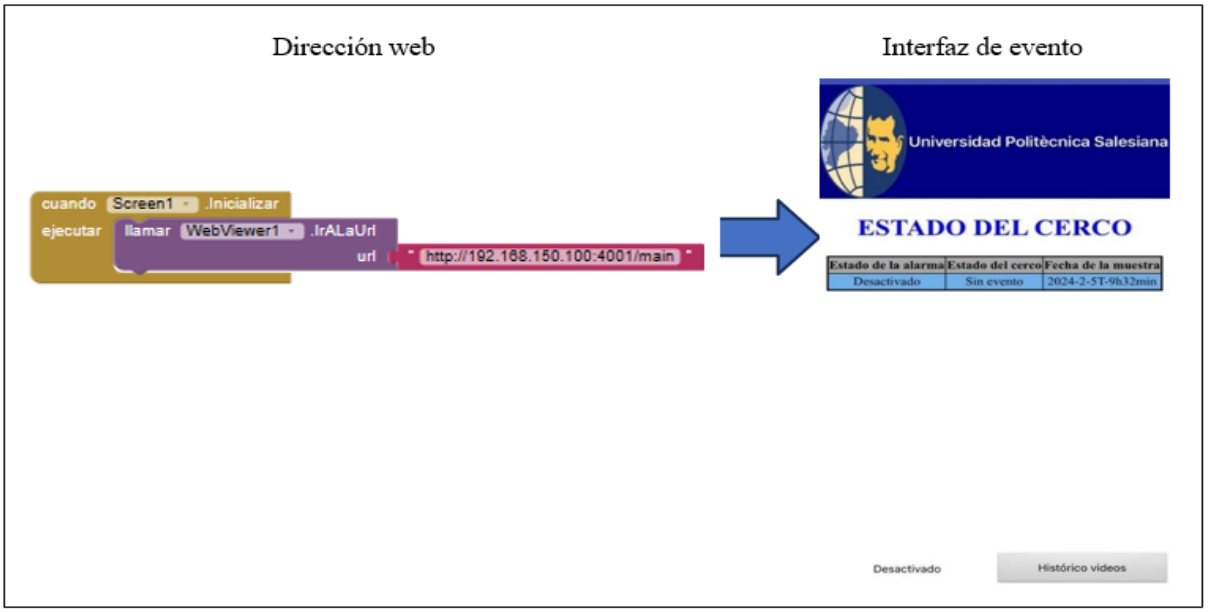


Nota. Definición en variables de alerta. Elaborado por: Los autores

Arranque y exploración web. Al inicializar la aplicación el evento "cuando Screen1 – Inicializar", el bloque mediante el componente "WebViewr1" navegue a la URL "http://192.168.150.100:4001/main" como se muestra en la **Figura 28**.

Figura 28

Evento designado para direccionar hacia una página web

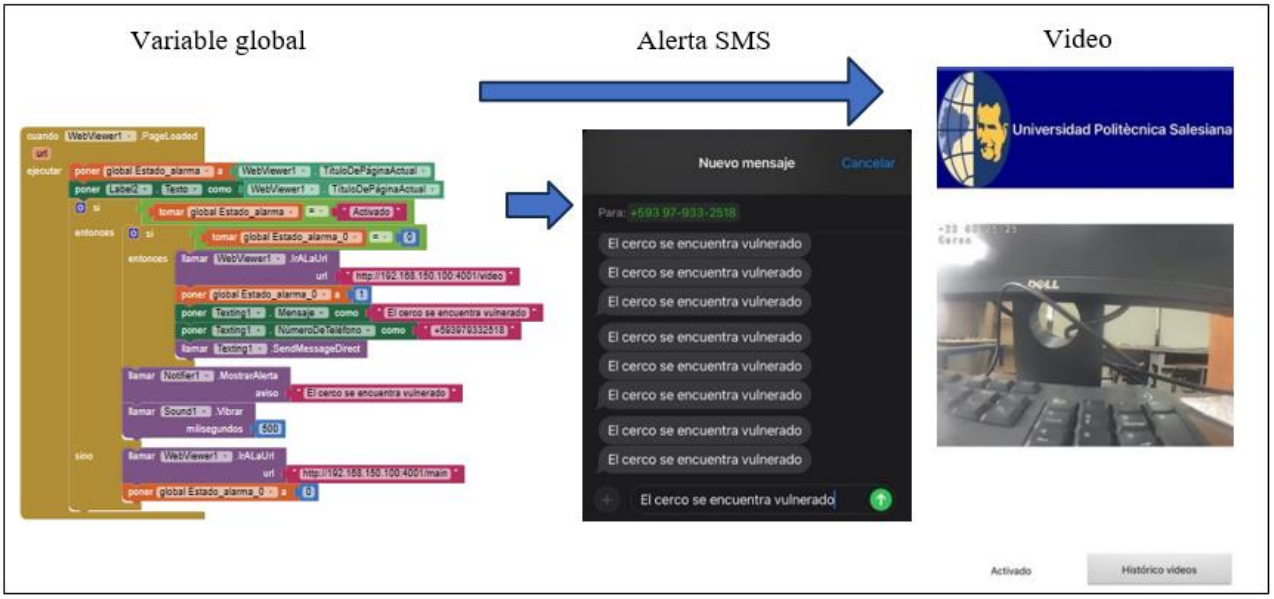


Nota. Inicialización de evento de la aplicación mediante webview. Elaborado por: Los autores

Gestión de eventos. En la **Figura 29** se muestra un conjunto de eventos anidados que permiten de manera inicial tomar el valor del estado del cerco y actualizar la variable global "Estado_alarma". Si el estado de la alarma se encuentra activado (se ha presentado una vulneración de seguridad), se verifica el estado anterior de la alarma mediante la variable "Estado_alarma_0", en el caso de detectar un cambio del estado de la alarma la aplicación genera una alerta mediante un mensaje de texto a un número de teléfono móvil, se muestra en la pantalla el mensaje "El cerco se encuentra vulnerado", así como, el video en tiempo real de la cámara de vigilancia. Para el caso de no presentarse un cambio del estado de la alarma, la pantalla muestra únicamente el estado del cerco sin texto.

Figura 29

Evento designado para el direccionamiento de páginas web, notificaciones de alarma y visualización de datos históricos.

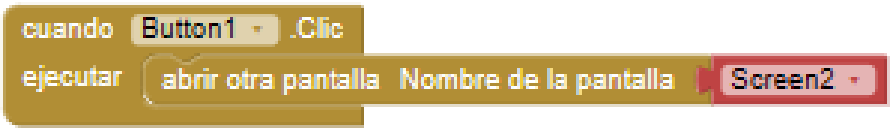


Nota. Manejo de eventos para detección y alerta de vulnerabilidad de seguridad en sistema de monitoreo. Elaborado por: Los autores

Al presionar el botón correspondiente a “Histórico videos”, se realiza el cambio de pantalla de la principal hacia la segunda pantalla, como se muestra en la **Figura 30**.

Figura 30

Evento designado para el cambio de pantalla

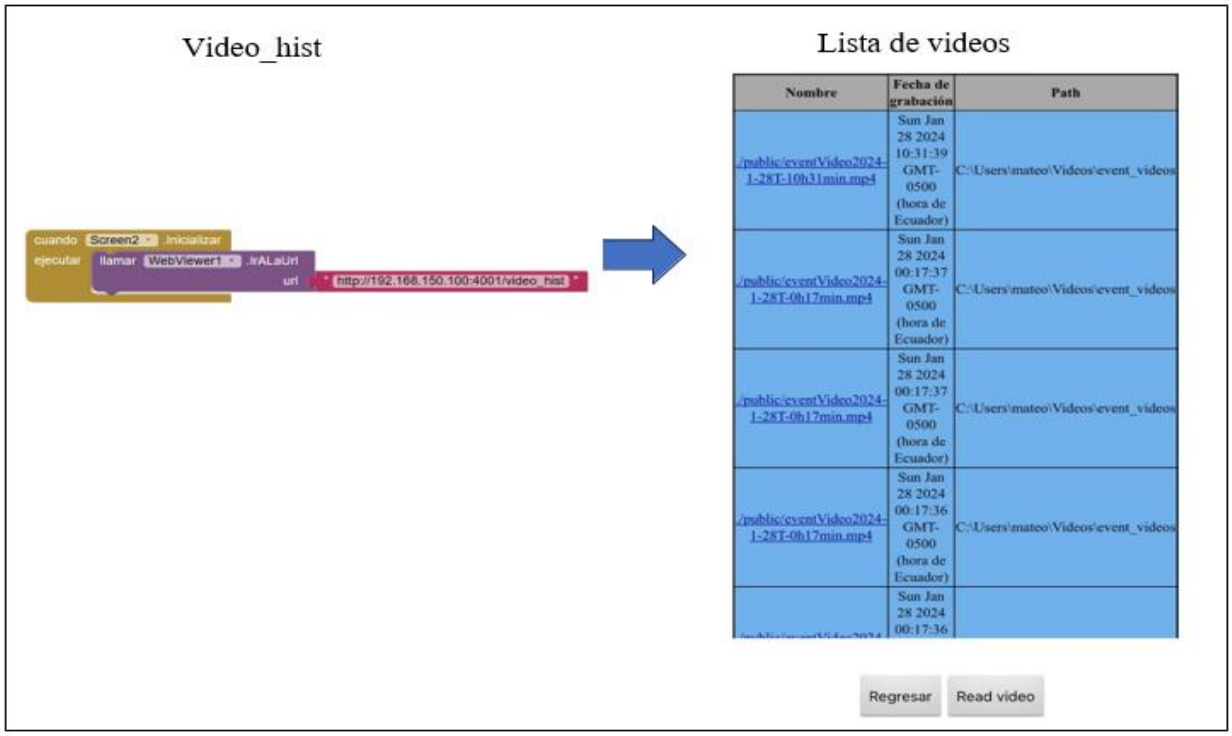


Nota. Evento de cambio de pantalla al accionar el botón 'Histórico de Videos’. Elaborado por: Los autores

Al realizarse el cambio de pantalla, se direcciona la página web “http//192.168.150.100:4001/video_hist”, que muestra los datos históricos, el evento se puede observar en la **Figura 31.**

Figura 31

Evento designado para mostrar los datos históricos.



Nota. Redireccionamiento a la página de datos históricos al cambiar de pantalla Elaborado por: Los autores

En la **Figura 32** se muestra el evento designado para el botón “Read Video”, el cual muestra el ultimo video grabado por la cámara de seguridad mediante la URL “http://192.168.150.100:4001/main”.

Figura 32

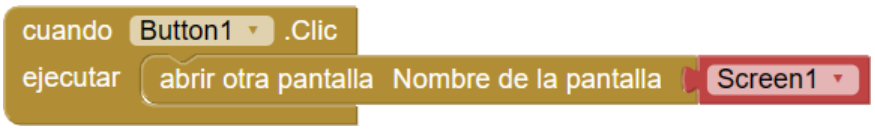
Evento designado para mostrar el último video grabado por la cámara de seguridad.

Nota. Evento para mostrar el último video grabado. Elaborado por: Los autores

Finalmente, en la **Figura 33** se presenta la función para el botón “Regresar”, el evento asignado permite el cambio de la pantalla segundaria a la pantalla principal, el evento se muestra en la siguiente figura

Figura 33

Evento designado para el cambio hacia la pantalla principal



Nota. Función de retorno a la pantalla principal. Elaborado por: Los autores

CAPÍTULO III

RESULTADOS

En este capítulo se presentan dos pruebas, una para obtener la buena práctica de programación y rendimiento de carga para verificar la tolerancia del aplicativo.

3.1.1 *Análisis en la calidad de código*

El estudio realizado en la **Tabla 9,** está orientado por bloques de código hacia detección de fallas y transferencia de datos en la aplicación móvil. En el **primer bloque** se ocupa el procesamiento de señales recibidas a la cámara captando las condiciones diurnas y nocturnas de ambiente. El **segundo bloque** se centra en la comunicación entre el hardware del sistema de vigilancia y la interfaz de la aplicación móvil. Esto incluye la transmisión de datos en tiempo real, la gestión de alertas y el acceso seguro a las grabaciones almacenadas.

El conjunto central en la comunicación de protocolos es ampliamente adaptable en intercambiar datos con Node.js. En continuación a los estándares de codificación en nuestros scripts a un formato analizable, se utilizó la herramienta SonarCloud, para apreciar los errores en el código y rectificarlos.

Tabla 9

Métricas de SonarCloud

Bloque 1: Señal de cámara diurna y nocturna	
Medidas	**Resultados**
Releasability	QS = Passed
Reliability	BG= 0, RAT = A, RE = 0
Security	VL= 0, RAT = A, RE = 0
Security Review	NSH = 0, RAT = A, RW = 100%
Maintainability	CS = 2, DB= 3H, DBR = 0%
Duplications Size	DS = 0,0%, DL = 0, DB = 0, NL = 196, LOC = 124, L = 200, ST = 100, FU = 2, CL = 0, FL = 1, COL = 20, COM = 10,8%
Complexity Issues	CC= 0, CGC = 0, ISS = 0
Bloque 2: Comunicación entre el ESP32 y la aplicación móvil	
Medidas	**Resultados**
Releasability	QS = Passed
Reliability	BG= 0, RAT = A, RE = 0
Security	VL= 0, RAT = A, RE = 0
Security Review	NSH = 0, RAT = A, RW = 100%
Maintainability	CS = 0, DB= 3H, DBR = 0%
Duplications Size	DS = 0,0%, DL = 0, DB = 0, NL = 196, LOC = 164, L = 224, ST = 100, FU = 2, CL = 0, FL = 1, COL = 24, COM = 12,8%
Complexity Issues	CC= 0, CGC = 0, ISS = 0

Nota. QS - Quality Gate Status, BG - Bugs, RAT - Rating, RE - Remediation Effort, VL - Vulnerabilities, NSH - New Security Hotspot, RW - Reviewed, CS - Code Smells, DB - Debt & Duplicated Blocks, DBR - Debt Ratio, DS - Density, DL - Duplicated Lines, NL - New Lines, LOC - Lines of Code, L - Lines, ST - Statements, FU - Functions, CL - Classes, FL - Files, COL - Comment Lines, COM - Comments, CC - Cyclomatic Complexity, CGC - Cognitive Complexity, ISS – Issue. Elaborado por: Los autores

En la **Tabla 9**, podemos contemplar que ambos bloques de código presentan métricas con excelencia en los aspectos críticos de software. El Bloque 1 encargado de captar las señales de cámara diurna y nocturna, indica una puerta de calidad sobrepasada, con una confiabilidad y seguridad excepcional, no se detectaron bugs ni vulnerabilidades, lo cual es fundamental para un sistema de seguridad que requiere una operatividad impecable en todo momento. El 0% duplicación es un indicativo de código bien estructurado, que favorece la mantenibilidad y ampliaciones del sistema.

Es notable la presencia de 100 statements, lo que sugiere que, aunque el código es confiable, su complejidad podría revisarse. Además, la presencia de un 10,8% en líneas de comentarios proporciona una documentación adecuada, vital para el mantenimiento y comprensión del sistema para otros desarrolladores.

En lo que respecta al Bloque 2, también ha superado con éxito la prueba de calidad, demostrando fiabilidad y seguridad sin errores ni vulnerabilidades, aspectos cruciales para la integridad de comunicación en la aplicación móvil. Aunque no se detectaron bloques de código duplicados, lo cual es excelente, hay un porcentaje ligeramente superior en líneas de comentarios (12,8%) en comparación con el Bloque 1, se indica una posible sobre documentación que podría ser optimizada cambiando las variables para mantener la claridad de código sin redundancia.

3.1.2 *Concurrencia*

Para el análisis de concurrencia se realizaron tres escenarios basados en el tiempo de cada escenario realizando pruebas en conexión wifi y ethernet. Las pruebas se formaron con un grupo de hilos para generar 6 peticiones HTTP tipo GET y POST, las cuales se detallan en la **Tabla 10**.

Para comprobar el funcionamiento y desempeño del servidor se utilizará Apache JMeter determinando el número de usuarios que la computadora puede soportar en el servidor NodeJS.

3.1.3 *Peticiones HTTP*

Para comprobar el funcionamiento y el desempeño del servidor se utilizará Apache JMeter para pruebas de concurrencia para determinar el número de usuarios que la computadora puede soportar corriendo el servidor en NodeJS.

Para esto se realizaron tres escenarios basados en el tiempo de la concurrencia y dentro de cada escenario se realizaron dos pruebas basadas en el medio de conexión, wifi o ethernet.

Para las pruebas de concurrencia, se creó un grupo de hilos para generar seis peticiones HTTP tipo GET y POST, las cuales se detallan en la **Tabla 10.**

La **Tabla 10** presenta un desglose de las solicitudes web efectuadas en la aplicación, delineando así la interfaz de comunicación que conecta al servidor Node.js con el cliente. Estas peticiones se probaron exhaustivamente para evaluar la capacidad de respuesta del servidor bajo condiciones de alta carga, simulando el uso real del sistema en un entorno de producción.

Tabla 10.

Peticiones HTTP para prueba de concurrencia

Petición	Tipo	URL	Funcionamiento
M	GET	http://localhost:4001	Carga de la interfaz principal del sistema y visualización en el estado actual de cerco con alarma.
UV	GET	http://localhost:4001/last_video	Revisión del último video capturado.
VS	GET	http://localhost:4001/video	Monitoreo activo inmediato en tiempo real.
HV	GET	http://localhost:4001/video_hist	Accesos simultáneos en históricos de video mp4.

VH	GET	http://localhost:4001 /show_video_hist.ejs?video_name=./public/eventVideo2024- 1-28T-0h17min.mp4	Gestión de solicitudes para recuperación y transmisión de grandes archivos multimedia.
EDE	POST	http://localhost:4001 /put	Interacción bidireccional del ESP32 al servidor principal.

Nota. **M**: Main, **UV**: Último video, **VS**: Video Stream, **HV**: Histórico de videos, **VH**: Video del histórico, **EDE**: envío de datos desde la esp32. Elaborado por: Los autores

3.1.4 *Primer escenario (30 minutos)*

En este escenario se realiza la prueba de concurrencia con un total de usuarios de 300 en un tiempo de 30 minutos, los resultados de las pruebas se presentan en la **Tabla 11**.

Tabla 11. Primer escenario 300 usuarios en 30 minutos

Peticion	Wifi	Eth	Wifi	Eth	Wifi	Eth	Wifi	Eth	Wifi	Eth	Wifi	Eth
	M		UV		VS		HV		VH		EDE	
Muestras	300	300	300	300	300	300	300	300	300	300	300	300
Media	0,27	0,37	0,48	0,83	0,48	0,83	0,93	1,28	0,48	0,83	1,64	2,90
Min	0,001	0,002	0,001	0,001	0,001	0,001	0,016	0,017	0,001	0,002	0,029	0,046
Max	**3,70**	**3,91**	**3,75**	**4,41**	**3,75**	**4,39**	**6,84**	**7,84**	**3,75**	**4,41**	**8,54**	**9,32**
Error	0,0%	0,0%	0,3%	0,2%	0,4%	0,2%	0,1%	0,1%	0,4%	0,2%	1,1%	1,5%
Rendimiento	**1,1**	**1,1**	**0,1**	**0,1**	**0,1**	**0,1**	**0,9**	**0,9**	**0,1**	**0,1**	**0,3**	**0,3**
Recepción	**1,7**	**1,7**	**7,7**	**7,3**	**5,7**	**5,4**	**15,3**	**14,9**	**5,6**	**5,3**	**5,2**	**4,5**
Envío	**0,1**	**0,1**	**1,0**	**0,9**	**0,9**	**0,9**	**0,1**	**0,1**	**1,4**	**1,4**	**5,8**	**5,0**

Nota. **Media**: Valor promedio del total de muestras de tiempo de respuesta en segundos, **Min**: Tiempo mínimo de respuesta en segundo, **Max**: Tiempo máximo de respuesta en segundo, **Rendimiento**: Número de solicitudes por segundo, **Recepción**: Velocidad de recepción en Kb/sec, **Envío**: Velocidad de envío en Kb/sec. Elaborado por: Los autores

Las métricas que se consideran para el análisis son, el tiempo máximo, el rendimiento y la velocidad de envío y recepción, en la **Figura 34**, **Figura 35** y **Figura 36** se presentan las métricas subrayadas de la **Tabla 11.**

En la **Figura 34.** Gráfica de valores Tiempo Max, escenario se observa que la petición que toma mayor tiempo es la petición tipo POST de la escritura a la base de datos con un valor de 9,32 segundos, esto puede ser debido a que, al aumentar el número de usuarios, MongoDB demora en realizar la escritura. Así mismo también se puede observar que la petición **HV** también presenta valores de tiempo de respuesta altos ya que al aumentar el número de videos guardados en la base de datos, el archivo en HTML que el servidor envía al cliente es de mayor tamaño aumentando el cálculo computacional y por ende aumentando el tiempo de respuesta al tener varios usuarios solicitando dicha información. La petición que genera una menor cantidad de tiempo es la solitud **M**, ya que esta, solo muestra en el documento HTML tres datos que se encuentran de forma global en el entorno de NodeJS.

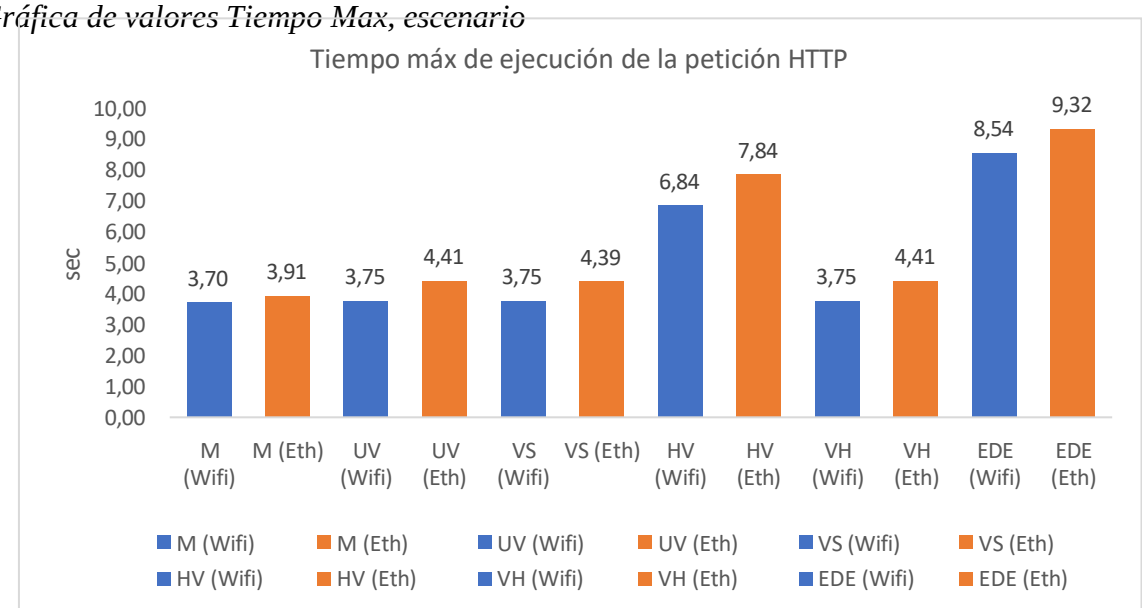
- Para **'M'** en Wi-Fi, el tiempo máximo de ejecución de la petición HTTP es de 3,70 segundos. Para 'M' en Ethernet, el tiempo es ligeramente superior con 3,91 segundos.
- La página **'UV'** en Wi-Fi tiene un tiempo de ejecución de 3,75 segundos. En Ethernet, el tiempo de ejecución es de 4,41 segundos.
- **'VS'** en Wi-Fi muestra un tiempo de ejecución de 3,75 segundos. Para 'VS' en Ethernet, el tiempo es casi el mismo con 4,39 segundos.
- En **'HV'** con Wi-Fi, el tiempo de ejecución es de 6,84 segundos. Para 'HV' en Ethernet, se reduce a 7,84 segundos.
- **'VH'** en Wi-Fi, tiene un tiempo de ejecución de 7,84 segundos. En Ethernet, el tiempo de ejecución es de 4,41 segundos.
- Finalmente, para **'EDE'** en Wi-Fi, el tiempo es de 8,54 segundos, y en Ethernet, es de

9,32 segundos, siendo este el tiempo más largo registrado en el gráfico.

Los datos muestran que los tiempos de ejecución de las peticiones HTTP varían entre Wi-Fi y Ethernet para diferentes páginas o categorías evaluadas. En general, los tiempos de ejecución en Wi-Fi tienden a ser más bajos para algunas categorías, como 'M', 'UV', y 'VS', donde los tiempos son de 3,70 segundos, 3,75 segundos, y 3,75 segundos respectivamente. Sin embargo, en otras categorías como 'HV', 'VH', y 'EDE', los tiempos de ejecución en Ethernet son más altos, con 7,84 segundos, 4,41 segundos, y 9,32 segundos respectivamente, mostrando una variabilidad en la eficiencia de ambas conexiones dependiendo de la categoría específica.

En conclusión, los tiempos de ejecución para completar peticiones HTTP no muestran una ventaja consistente de Wi-Fi sobre Ethernet o viceversa. La eficiencia en términos de tiempo de ejecución varía según la categoría específica, lo que sugiere que la elección entre Wi-Fi y Ethernet podría basarse en otros factores además del tiempo de ejecución solo, dada la cercanía en el rendimiento entre ambas tecnologías para las tareas evaluadas.

Figura 34.
Gráfica de valores Tiempo Max, escenario



Nota. Tiempo máx de ejecución. Elaborado por: Los autores

En la **Figura 35** se observa que la petición HV (Histórico de video), tiene la mayor tasa de recepción de datos, esto es debido a que se debe enviar una tabla en formato HTML y esta es una tabla dinámica es decir que el tamaño de esta depende del número de videos guardados por lo tanto el número de fila no es un número estático.

- Para '**M**', la velocidad de envío es igual en ambas conexiones, Wi-Fi y Ethernet, con 0.1 KB/s. La velocidad de recepción también es la misma en ambos casos, siendo 1.7 KB/s.
- En '**UV**', la velocidad de envío difiere ligeramente entre Wi-Fi (1.0 KB/s) y Ethernet (0.9 KB/s), mientras que la velocidad de recepción es casi idéntica, siendo 7.7 KB/s en Wi-Fi y 7.3 KB/s en Ethernet.
- Para '**VS**', ambas conexiones presentan una velocidad de envío de 0.9 KB/s. La recepción alcanza los 5.7 KB/s en Wi-Fi y los 5.4 KB/s en Ethernet, mostrando una pequeña diferencia.
- En '**HV**', tanto en Wi-Fi como en Ethernet, la velocidad de envío es de 0.1 KB/s. La recepción muestra una mayor velocidad en Wi-Fi con 15.3 KB/s, comparado con 14.9 KB/s en Ethernet.
- Para '**VH**', la velocidad de envío es idéntica en Wi-Fi y Ethernet, con 1.4 KB/s. En la recepción, Wi-Fi ofrece una velocidad de 5.6 KB/s, ligeramente superior a los 5.3 KB/s de Ethernet.
- En '**EDE**', se observa una mayor velocidad de envío en Wi-Fi con 5.8 KB/s, en comparación con 5.0 KB/s en Ethernet. En cuanto a la recepción, Wi-Fi presenta 5.2 KB/s frente a los 4.5 KB/s de Ethernet.

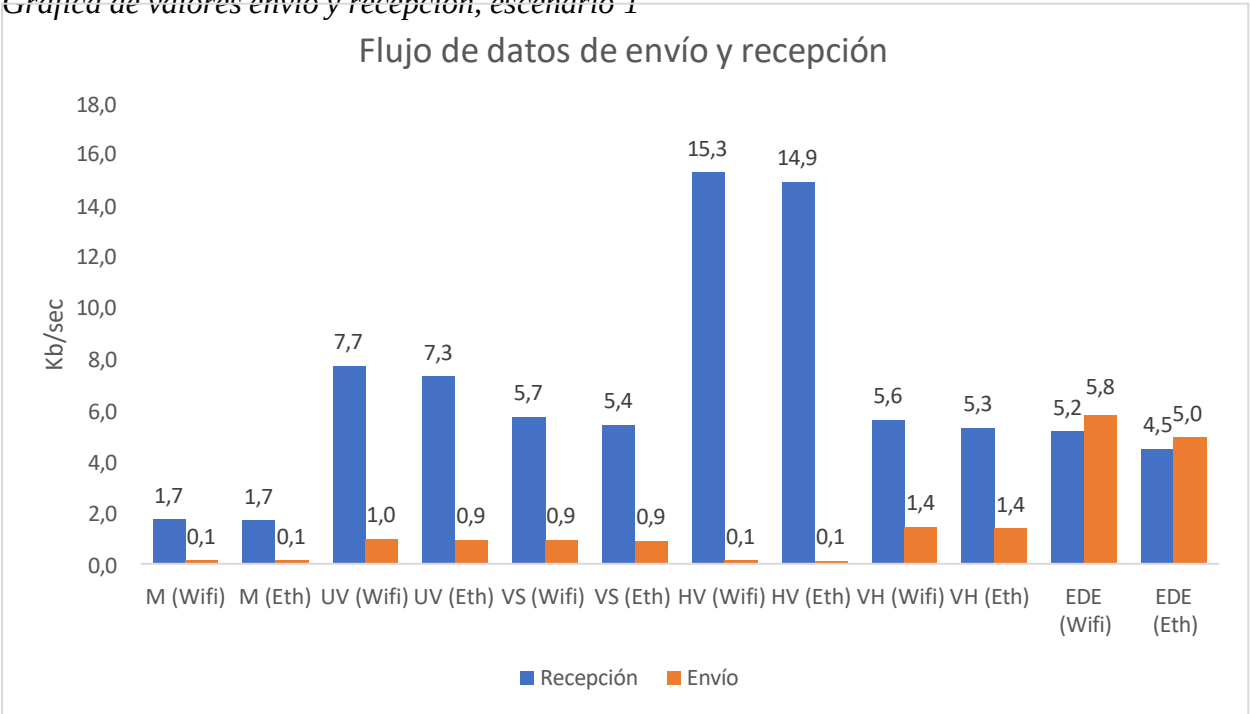
Los datos muestran que las velocidades de envío y recepción entre Wi-Fi y Ethernet son relativamente similares para las categorías evaluadas ('M', 'UV', 'VS', 'HV', 'VH', 'EDE'). En 'M', ambas conexiones presentan velocidades idénticas tanto en envío (0.1 KB/s) como en

recepción (1.7 KB/s). Se observan diferencias ligeras en 'UV', 'VS', 'HV', 'VH', y 'EDE', pero ninguna es significativamente amplia. Por ejemplo, en 'UV', la velocidad de envío es ligeramente mayor en Wi-Fi (1.0 KB/s) comparado con Ethernet (0.9 KB/s), y un patrón similar se observa en las demás categorías, donde Wi-Fi muestra en general una ligera ventaja en la velocidad de recepción, como en 'HV' con 15.3 KB/s en Wi-Fi frente a 14.9 KB/s en Ethernet.

Por lo tanto, mientras que existen diferencias mínimas en las velocidades de envío y recepción entre Wi-Fi y Ethernet en varias categorías, estas no son lo suficientemente pronunciadas como para declarar una superioridad clara de un tipo de conexión sobre el otro en términos generales. La elección entre Wi-Fi y Ethernet puede, por tanto, basarse más en consideraciones de conveniencia, disponibilidad o preferencia personal, dada la similitud en el rendimiento de velocidad entre ambas tecnologías para las tareas evaluadas.

Figura 35

Gráfica de valores envió y recepción, escenario 1



Nota. Flujo de datos envío y recepción. Elaborado por: Los autores

La **Figura 36** muestra que la petición **M** tiene la mayor tasa de solicitudes (1,1 req/sec), esto concuerda con las gráficas anteriores ya que al ser una petición que devuelve un documento de menor tamaño, adicionalmente no presenta un proceso antes de enviar el documento en HTML no existe calculo computacional adicional. Adicionalmente se nota que la petición **HV** está como segundo en cuanto a tasa de solicitudes, esto concuerda con la **Figura 35**, ya que, al tener una alta tasa de envío y recepción de datos, debería poder realizar una mayor cantidad de solicitudes que las que tienen una tasa baja de flujo de datos.

A continuación, presentamos una comparación acerca de cómo el rendimiento indica las solicitudes que puede dar por segundo.

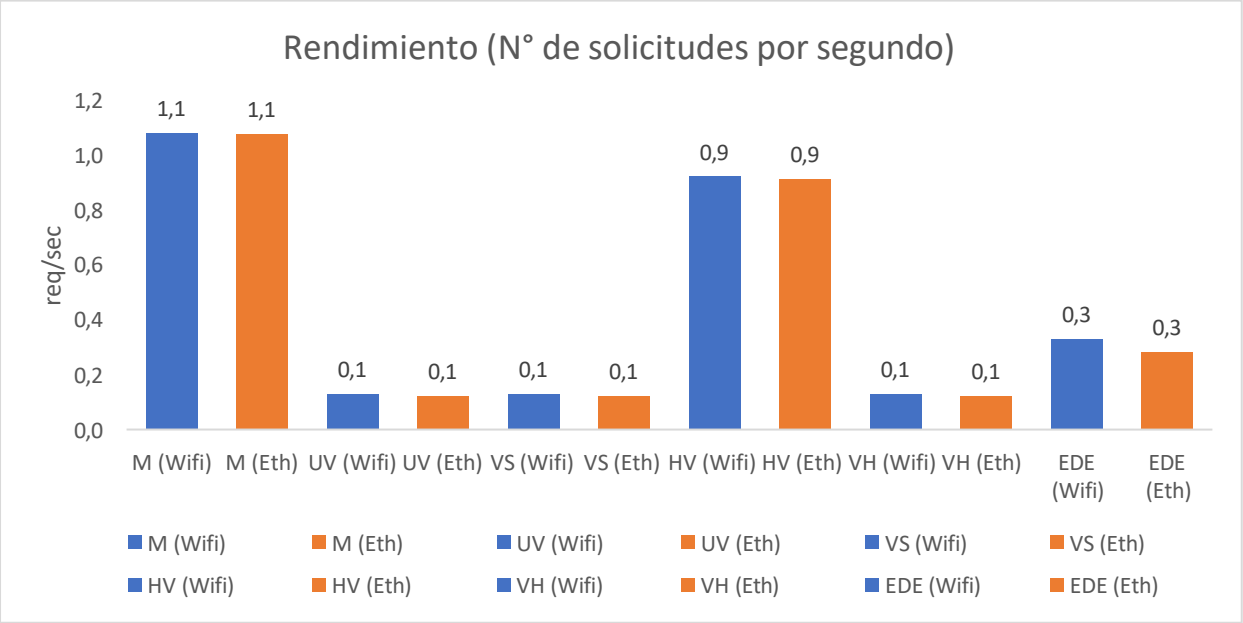
- Para **'M'**, se mantiene un rendimiento constante con 1.1 solicitudes por segundo tanto en Wi-Fi como en Ethernet.
- **'UV'** muestra un rendimiento idéntico de 0.1 solicitudes por segundo en ambas Wi-Fi y Ethernet.
- **'VS'** también exhibe un rendimiento uniforme en ambas conexiones con 0.1 solicitudes por segundo.
- **'HV'** alcanza 0.9 solicitudes por segundo en ambos Wi-Fi y Ethernet, indicando una capacidad de respuesta similar en ambas modalidades de conexión.
- **'VH'** registra 0.1 solicitudes por segundo tanto en Wi-Fi como en Ethernet, sin diferencias notables entre los dos tipos de conexión.
- **'EDE'** igualmente tiene un rendimiento de 0.3 solicitudes por segundo en Wi-Fi y Ethernet, lo cual sugiere que la eficiencia es comparable independientemente de la conexión utilizada.

Los datos indican que el rendimiento, medido en solicitudes por segundo, es consistentemente similar entre las conexiones Wi-Fi y Ethernet para todos los servicios evaluados ('M', 'UV', 'VS', 'HV', 'VH', 'EDE'). Cada uno de estos servicios muestra un

rendimiento idéntico en ambas modalidades de conexión, con 'M' manteniendo 1.1 solicitudes por segundo, 'UV', 'VS', y 'VH' con 0.1 solicitudes por segundo, 'HV' alcanzando 0.9 solicitudes por segundo, y 'EDE' con 0.3 solicitudes por segundo. Esto sugiere que, para los servicios evaluados, la elección entre Wi-Fi y Ethernet no influye en la capacidad de procesar solicitudes por segundo, manteniendo una eficiencia comparable en ambos tipos de conexión.

Figura 36

Gráfica de valores de rendimiento, escenario 1



Nota. Rendimiento. Elaborado por: Los autores

Con base a la **Figura 34.** Gráfica de valores Tiempo Max, escenario **,** Los datos muestran que las velocidades de envío y recepción entre Wi-Fi y Ethernet son relativamente similares para las categorías evaluadas ('M', 'UV', 'VS', 'HV', 'VH', 'EDE'). En 'M', ambas conexiones presentan velocidades idénticas tanto en envío (0.1 KB/s) como en recepción (1.7 KB/s). Se observan diferencias ligeras en 'UV', 'VS', 'HV', 'VH', y 'EDE', pero ninguna es significativamente amplia. Por ejemplo, en 'UV', la velocidad de envío es ligeramente mayor en Wi-Fi (1.0 KB/s) comparado con Ethernet (0.9 KB/s), y un patrón similar se observa en las

demás categorías, donde Wi-Fi muestra en general una ligera ventaja en la velocidad de recepción, como en 'HV' con 15.3 KB/s en Wi-Fi frente a 14.9 KB/s en Ethernet.

Por lo tanto, mientras que existen diferencias mínimas en las velocidades de envío y recepción entre Wi-Fi y Ethernet en varias categorías, estas no son lo suficientemente pronunciadas como para declarar una superioridad clara de un tipo de conexión sobre el otro en términos generales. La elección entre Wi-Fi y Ethernet puede, por tanto, basarse más en consideraciones de conveniencia, disponibilidad o preferencia personal, dada la similitud en el rendimiento de velocidad entre ambas tecnologías para las tareas evaluadas.

Figura 35. Gráfica de valores envió y recepción, escenario y **Figura 36**. Gráfica de valores de rendimiento, escenario se puede notar que las peticiones de EDE son las que mayor tiempo tardan en terminar la concurrencia, con tiempo de 8,54 y 9,32 segundos para wifi y cable ethernet respectivamente. Mientras que en el caso de la petición M son las que mejor rendimiento presentan con valor de 1,08 peticiones por segundo. Para el caso de envío y recepción de datos se puede notar que la petición HV es la que mayor cantidad de datos de recepción tiene y para el envío destaca la petición EDE, esto da una idea de cual petición es la que más datos envía al cliente y cuál es la que más datos recibe del cliente.

3.1.5 *Segundo escenario (60 minutos)*

En este escenario se realiza la prueba de concurrencia con un total de usuarios de 600 en un tiempo de 60 minutos, los resultados de las pruebas se presentan en la **Tabla 12**.

Tabla 12
Segundo escenario 600 usuarios en 60 minutos

Petición	Wifi	Eth	Wifi	Eth	Wifi	Eth	Wifi	Eth	Wifi	Eth	Wifi	Eth
	M		UV		VS		HV		VH		EDE	
Muestras	600	580	600	580	600	580	600	580	600	580	600	580
Media	1,103	2,172	0,759	1,358	0,759	1,353	6,441	6,807	0,758	1,358	7,465	8,839
Min	0,002	0	0,001	0	0,001	0	0,03	0	0,001	0	0,028	0
Max	**4,93**	**11,335**	**5,171**	**11,649**	**4,804**	**11,648**	**13,648**	**20,363**	**5,169**	**11,65**	**17,205**	**25,705**
Error	0,00%	28,04%	0,04%	6,24%	0,04%	6,25%	0,00%	29,76%	0,04%	6,19%	5,79%	39,88%
Rendimiento	**1,04**	**3,68**	**2,22**	**0,63**	**2,22**	**0,63**	**1,24**	**3,52**	**2,22**	**0,63**	**6,82**	**17,50**
Recepción	**1,66**	**6,79**	**131,52**	**41,34**	**97,57**	**32,23**	**38,15**	**24,36**	**95,97**	**31,73**	**1,79**	**19,76**
Envío	**0,12**	**0,31**	**16,76**	**4,48**	**16,11**	**4,30**	**0,16**	**0,31**	**24,93**	**6,66**	**2,00**	**3,27**

Nota. **Media**: Valor promedio del total de muestras de tiempo de respuesta en segundos, **Min**: Tiempo mínimo de respuesta en segundo, **Max**: Tiempo máximo de respuesta en segundo, **Rendimiento**: Número de solicitudes por segundo, **Recepción**: Velocidad de recepción en Kb/sec, **Envío**: Velocidad de envío en Kb/sec, Elaborado por: Los autores

Se consideran las mismas métricas para el análisis de la sección anterior, en la **Figura 37**, **Figura 38** y **Figura 39** se presentan los gráficos de la **Tabla 12**.

En la **Figura 37** se observa que la petición que toma mayor tiempo es la de EDE (escritura a la base de datos) nuevamente, esto es debido a que la cantidad de peticiones adicionales, el servidor tarda en guardar los datos en la base de datos y mostrar los valores solicitados. Se puede observar también que las peticiones VS y VH son las peticiones que se mantienen en un valor medio, ya que Jmeter al ser un entorno que mide el tiempo de respuesta del servidor, no recibe el video en sí, solo recibe la página HTML que el servidor envía como parte de su respuesta, teniendo un documento de tamaño fijo. Mientras que la petición que tiene menor tasa flujo de datos es M ya que como se mencionó antes es un documento HTML de bajo tamaño.

- En la categoría **'M'**, la petición se completa considerablemente más rápido en Wi-Fi, tomando solo 4.93 segundos, comparado con Ethernet que toma más del doble de tiempo con 11.335 segundos.

- Para **'UV'**, la eficiencia es mayor con Wi-Fi completando la petición en 5.171 segundos, mientras que Ethernet muestra un tiempo de ejecución mayor de 11.649 segundos.
- **'VS'** demuestra una notable eficiencia en Wi-Fi, ejecutando la petición en 4.804 segundos; en contraste, Ethernet muestra un tiempo mucho mayor con 11.648 segundos.
- **'HV'** exhibe una diferencia significativa, con Wi-Fi completando la petición en un tiempo sustancialmente menor de 13.648 segundos, mientras que Ethernet se demora mucho más, alcanzando los 20.363 segundos.
- En el caso de **'VH'**, Wi-Fi muestra una mayor rapidez al completar la petición en 5.146 segundos, en comparación con Ethernet que tiene un tiempo de ejecución de 11.65 segundos.
- Finalmente, para **'EDE'**, Wi-Fi muestra un tiempo de respuesta de 17.205 segundos, siendo más eficiente que Ethernet que registra el tiempo más extenso en el gráfico con 25.705 segundos.

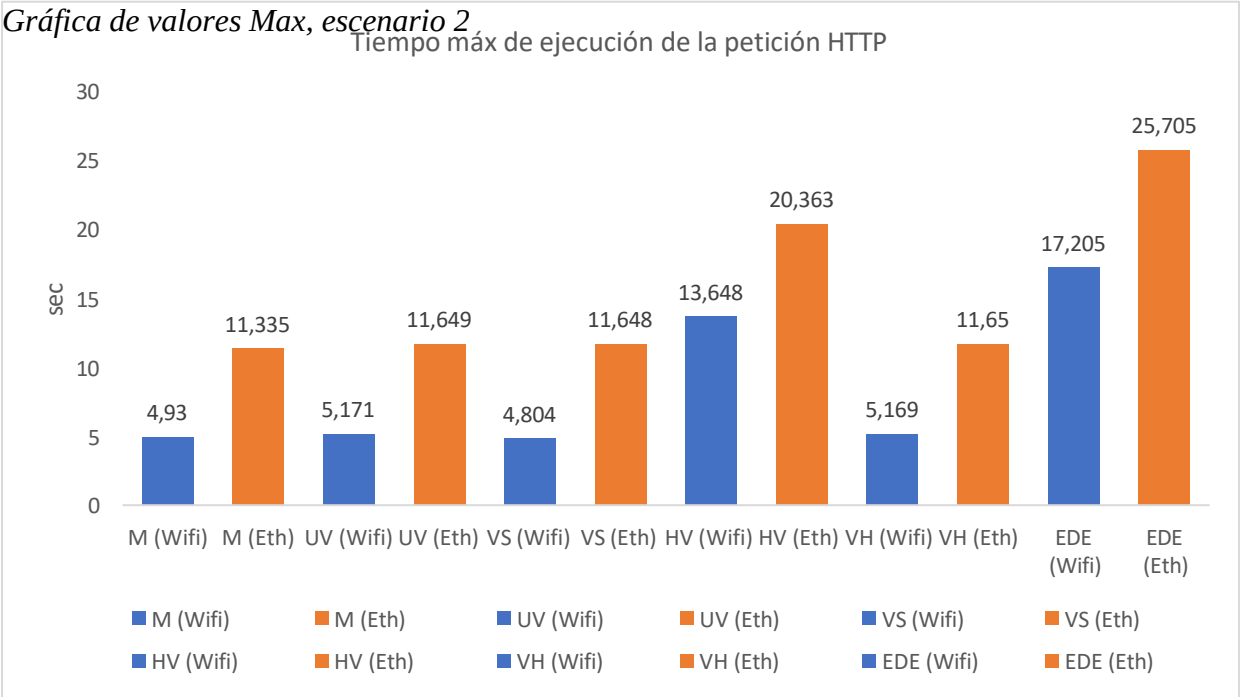
Los datos presentados muestran una diferencia significativa en los tiempos de ejecución de peticiones entre conexiones Wi-Fi y Ethernet para varias categorías ('M', 'UV', 'VS', 'HV', 'VH', 'EDE'). En todos los casos, Wi-Fi completó las peticiones considerablemente más rápido que Ethernet. Por ejemplo, en la categoría 'M', Wi-Fi tomó solo 4.93 segundos, mientras que Ethernet necesitó más del doble de tiempo, con 11.335 segundos. Similarmente, para 'EDE', Wi-Fi completó la petición en 17.205 segundos, comparado con los 25.705 segundos que tomó Ethernet, marcando este como el tiempo más extenso registrado en el análisis.

Estos datos indican una tendencia clara donde Wi-Fi supera a Ethernet en términos de velocidad de ejecución de peticiones en las categorías evaluadas, con tiempos de respuesta en Wi-Fi siendo consistentemente menores a los registrados en Ethernet. La eficiencia en la ejecución de peticiones es notablemente superior en Wi-Fi a través de todas las categorías

analizadas, sugiriendo que, para estas pruebas específicas, la conexión inalámbrica proporcionó un rendimiento más rápido en comparación con la conexión cableada.

Figura 37

Gráfica de valores Max, escenario 2



Nota. Elaborado por: Los autores

En la **Figura 38** se aprecia que las peticiones UV y VH son las que mayor volumen de datos manejan tanto en la recepción como en envío, lo cual concuerda con operaciones intensivas como carga y visualización de videos en Wifi. Por otro lado, la petición EDE sobresale por Ethernet, se evidencia un envío significativo de datos hacia el servidor, mientras que las peticiones M, VS y HV reflejan un intercambio de datos más moderado.

- En '**M**', la velocidad de envío con Wi-Fi es de 0.12 KB/s y aumenta a 0.31 KB/s cuando se utiliza Ethernet. Para la recepción, Wi-Fi logra una velocidad de 1.66 KB/s, la cual es significativamente superada por los 6.79 KB/s alcanzados con Ethernet.

- Para **'UV'**, la velocidad de envío es considerablemente mayor con Wi-Fi a 16.76 KB/s, en comparación con Ethernet que es de 4.48 KB/s. En recepción, Wi-Fi destaca con una velocidad de 131.52 KB/s, frente a los 41.34 KB/s de Ethernet.
- **'VS'** envía datos a 16.11 KB/s a través de Wi-Fi y a 4.30 KB/s con Ethernet. La recepción con Wi-Fi es de 97.57 KB/s, mientras que con Ethernet es de 32.23 KB/s.
- **'HV'** tiene una velocidad de envío menor con Wi-Fi de 0.16 KB/s, ligeramente inferior a la velocidad de Ethernet de 0.31 KB/s. No obstante, la recepción con Wi-Fi es mucho más eficiente a 38.15 KB/s en comparación con los 24.36 KB/s de Ethernet.
- Para **'VH',** el envío de datos es más alto con Wi-Fi a 24.93 KB/s frente a los 6.66 KB/s de Ethernet. La recepción también es mejor con Wi-Fi, alcanzando velocidades de 95.97 KB/s contra 31.73 KB/s con Ethernet.
- Finalmente, en la categoría **'EDE'**, la velocidad de envío con Wi-Fi es de 2.00 KB/s, mientras que con Ethernet es ligeramente superior a 3.27 KB/s. La recepción presenta una gran diferencia, siendo 1.79 KB/s para Wi-Fi y notablemente más alta para Ethernet con 19.76 KB/s.

Los datos indican que, en general, Wi-Fi proporciona velocidades de recepción superiores en comparación con Ethernet, excepto en 'EDE' donde Ethernet supera a Wi-Fi tanto en envío como en recepción.

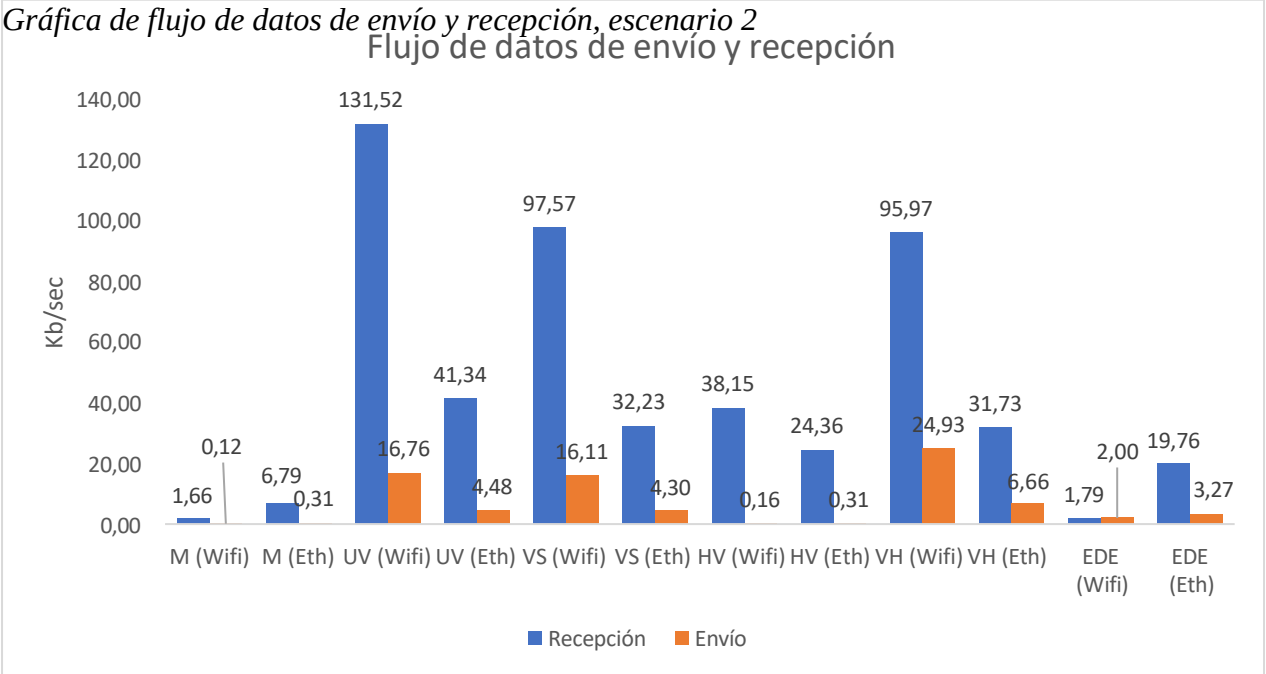
Los datos muestran variaciones en las velocidades de envío y recepción entre conexiones Wi-Fi y Ethernet para diferentes categorías ('M', 'UV', 'VS', 'HV', 'VH', 'EDE'). En general, Ethernet supera a Wi-Fi en la velocidad de recepción en categorías como 'M' y 'EDE', con velocidades de 6.79 KB/s y 19.76 KB/s respectivamente, frente a las velocidades de Wi-Fi de 1.66 KB/s y 1.79 KB/s. Sin embargo, en otras categorías como 'UV', 'VS', y 'VH', Wi-Fi muestra una mayor velocidad tanto en envío como en recepción, con 'UV' destacando en Wi-Fi con

16.76 KB/s en envío y 131.52 KB/s en recepción, comparado con 4.48 KB/s y 41.34 KB/s de Ethernet.

Este análisis indica que la eficiencia de Wi-Fi y Ethernet puede variar significativamente dependiendo de la tarea específica, con Wi-Fi mostrando superioridad en la velocidad de envío y recepción en ciertas categorías, mientras que Ethernet destaca en otras, especialmente en la velocidad de recepción en las categorías 'M' y 'EDE'. La elección entre Wi-Fi y Ethernet podría, por lo tanto, depender de las necesidades específicas de velocidad de envío y recepción de datos de la tarea en cuestión.

Figura 38

Gráfica de flujo de datos de envío y recepción, escenario 2



Nota. Elaborado por: Los autores

La **Figura 39** ilustra que el rendimiento más alto lo refleja la escritura a la base de datos **EDE**, esto puede dar una idea del porque en la prueba de Ethernet el servidor se detiene, ya que el servidor intenta responder a todos los usuarios que están enviando la solicitud de tipo POST consumiendo una mayor cantidad de RAM y por ende ejerciendo una carga mayor en la

computadora. De igual manera el rendimiento del resto de peticiones se puede notar que no ha

existido gran cambio en comparación al escenario anterior, sin embargo, hay que tomar en cuenta en la prueba de Ethernet, el servidor se detiene cerca del minuto 30 por lo tanto en la prueba de 60 minutos con ethernet los valores pueden provocar confusiones.

- '**M**' demuestra un rendimiento en Wi-Fi de 1.04 solicitudes por segundo, que se incrementa significativamente a 3.68 solicitudes por segundo con Ethernet, mostrando que la conexión cableada es más eficiente para esta categoría.
- Para '**UV**', se observa un rendimiento en Wi-Fi de 2.22 solicitudes por segundo, mientras que Ethernet muestra un rendimiento reducido a 0.63 solicitudes por segundo. Esto podría indicar una mayor eficacia de la conexión inalámbrica sobre la cableada para esta categoría en particular.
- En **'VS'**, el rendimiento con Wi-Fi es de 2.22 solicitudes por segundo, equivalente al rendimiento con Ethernet, que también es de 0.63 solicitudes por segundo. Esto sugiere una consistencia en la capacidad de respuesta entre ambas formas de conexión para 'VS'.
- **'HV'** muestra un rendimiento de 1.24 solicitudes por segundo en Wi-Fi, que se ve superado por un rendimiento de 3.52 solicitudes por segundo en Ethernet, indicando que Ethernet proporciona una mejor respuesta en esta categoría.
- **'VH'** presenta un rendimiento de 2.22 solicitudes por segundo en Wi-Fi, que se iguala al rendimiento de Ethernet de 0.63 solicitudes por segundo.
- **'EDE'** se destaca con un rendimiento en Wi-Fi de 6.82 solicitudes por segundo, que es mucho menor en comparación con un rendimiento extraordinariamente alto de 17.50 solicitudes por segundo en Ethernet, lo que resalta la superioridad de la red cableada para esta categoría.

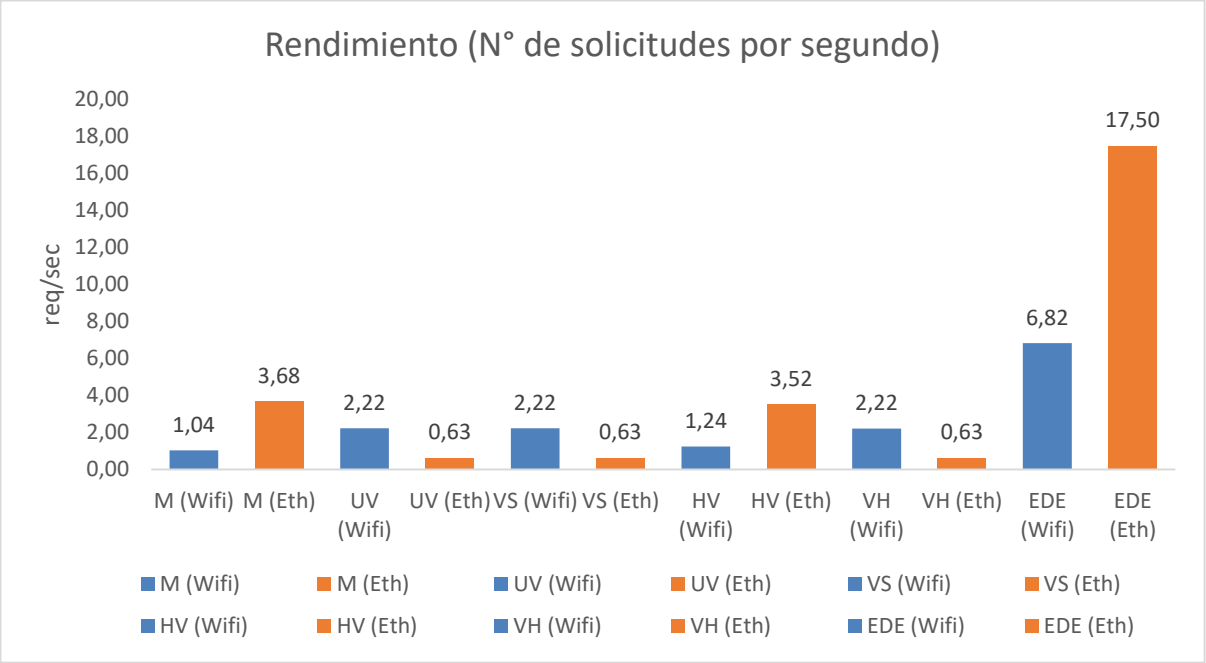
Los datos muestran variaciones en el rendimiento entre Wi-Fi y Ethernet al manejar solicitudes HTTP en diferentes categorías. Para 'M' y 'HV', Ethernet muestra un rendimiento

superior con 3.68 y 3.52 solicitudes por segundo respectivamente, comparado con Wi-Fi. Sin embargo, en 'UV', Wi-Fi supera a Ethernet con 2.22 solicitudes por segundo frente a 0.63 solicitudes por segundo. 'VS' y 'VH' muestran un rendimiento equivalente en Wi-Fi y Ethernet a 2.22 y 0.63 solicitudes por segundo respectivamente, indicando una consistencia en la capacidad de respuesta para 'VS', pero una discrepancia para 'VH' donde ambos modos igualan en un rendimiento más bajo. 'EDE' destaca por mostrar una diferencia significativa, con Ethernet manejando 17.50 solicitudes por segundo frente a 6.82 en Wi-Fi, subrayando una eficacia considerablemente mayor de la conexión cableada.

En conclusión, los resultados sugieren que la eficiencia de Wi-Fi y Ethernet varía según la categoría específica, con Ethernet mostrando una ventaja significativa en ciertas áreas ('M', 'HV', 'EDE'), mientras que Wi-Fi es preferible en otras ('UV'). Esto indica que la elección entre Wi-Fi y Ethernet debería basarse en el tipo específico de uso y las necesidades de rendimiento asociadas a cada categoría.

Figura 39

Gráfica de valores de rendimiento, escenario 2



Nota. Elaborado por: Los autores

Con base a la **Figura 37**, **Figura 38** y **Figura 39** se puede notar que la petición de EDE con cable ethernet es la que mayor tiempo tarda en terminar la concurrencia, con tiempo de 25,75 segundos, sin embargo, es la que mayor rendimiento presenta con un valor de 17,50 solicitudes por segundo. Para el caso de envío y recepción de datos se puede notar que la petición UV con wifi es la que mayor cantidad de datos de recepción tiene y para el envío destaca la petición VH con wifi, esto da una idea de cuál es la que mayor cantidad de datos que recibe del cliente.

Cabe notar que para la prueba con cable ethernet el servidor termina de forma repentina en la concurrencia número 480, es por esto que en la tabla 12 se puede apreciar un valor de error mayor al 20% para las solicitudes M, HV y EDE.

3.1.6 *Tercer escenario (120 minutos)*

En este escenario se realiza la prueba de concurrencia con un total de usuarios de 1000 en un tiempo de 120 minutos. Los resultados de las pruebas se presentan en la **Tabla 13** se aprecia que el número de muestras no coincide con los 1000 usuarios, esto debido a que el servidor termina su funcionamiento de forma inesperada a partir de la concurrencia número 820 para wifi y 850 para ethernet, las peticiones que provocaron que el servidor termine fueron las peticiones M, VS y EDE para la prueba de wifi y para ethernet fueron las peticiones M, VS, HV y EDE por el error mayor al 20%.

Tabla 13. Tercer escenario 1000 usuarios en 120 minutos

Petición	Wifi	Eth	Wifi	Eth	Wifi	Eth	Wifi	Eth	Wifi	Eth	Wifi	Eth
	M		UV		VS		HV		VH		EDE	
Muestras	820	850	820	850	820	850	820	850	820	850	820	850
Media	0,69	1,27	0,62	1,10	0,62	1,09	3,69	4,04	0,62	1,10	4,55	5,87
Min	0,00	0,00	0,00	0,00	0,00	0,00	0,02	0,01	0,00	0,00	0,03	0,02
Max	4,31	7,62	4,46	8,03	4,28	8,02	10,24	14,10	4,46	8,03	12,87	17,51
Error	48,40%	38,54%	16,04%	6,24%	48,40%	38,54%	16,04%	29,76%	16,04%	6,19%	48,40%	38,54%
Rendimiento	1,06	2,38	1,17	0,38	1,17	0,38	1,08	2,22	1,17	0,38	13,26	17,21
recepción	1,70	4,25	69,63	24,33	51,65	18,82	26,71	19,63	50,79	18,52	3,47	12,12
Envío	0,13	0,22	8,87	2,71	8,53	2,60	0,14	0,21	13,19	4,02	3,89	4,12

Nota. **Media**: Valor promedio del total de muestras de tiempo de respuesta en segundos, **Min**: Tiempo mínimo de respuesta en segundo, **Max**: Tiempo máximo de respuesta en segundo, **Rendimiento**: Número de solicitudes por segundo, **Recepción**: Velocidad de recepción en Kb/sec, **Envío**: Velocidad de envío en Kb/sec. Elaborado por: Los autores

Se consideran las mismas métricas para el análisis, en la **Figura 40**, **Figura 41** y **Figura 42** que se presentan con la **Tabla 13**.

En la **Figura 40.** Gráfica de valores Max, escenario 3 se puede notar nuevamente que el que posee el mayor tiempo de respuesta es la petición de escritura a la base de datos, indicando que cuando existe un mayor número de usuarios, la escritura de datos toma más tiempo. En este caso las pruebas, Wifi y ethernet, se detuvieron a los 30 y 39 minutos respectivamente debido a que el servidor se detuvo, por lo tanto, los datos adquiridos pueden dar una idea de cuál fue la causa de que el servidor se detenga.

- Para la categoría '**M**', utilizando Wi-Fi se completa una petición HTTP en 4.31 segundos, mientras que con Ethernet se hace en 4.46 segundos, mostrando una ligera diferencia con Ethernet siendo un poco más lento.
- En la categoría '**UV**', el tiempo de respuesta para completar una petición HTTP es casi idéntico en ambas conexiones, con Wi-Fi a 8.03 segundos y Ethernet a 8.02 segundos.

- La categoría **'VS'** muestra tiempos de respuesta iguales para completar una petición HTTP en ambas conexiones: 10.24 segundos tanto para Wi-Fi como para Ethernet.

- Se observa en **'HV'** una diferencia significativa, donde completar una petición HTTP en Wi-Fi toma 14.10 segundos, pero en Ethernet se reduce a 8.03 segundos, lo que indica una mejora notable con el uso de Ethernet.
- Para **'VH'**, Wi-Fi toma 12.87 segundos para completar una petición HTTP, y sorprendentemente, Ethernet toma aún más tiempo con 17.51 segundos.
- Finalmente, en la operación **'EDE'**, el tiempo para completar una petición HTTP en Wi-Fi es de 8.03 segundos, y en Ethernet aumenta considerablemente a 17.51 segundos, marcando este último como el tiempo más extenso registrado en la gráfica para ambas conexiones.

Los datos presentados muestran variaciones en los tiempos de respuesta para completar peticiones HTTP entre las conexiones Wi-Fi y Ethernet, dependiendo de la categoría específica analizada. En 'M', la diferencia es mínima, con Wi-Fi siendo ligeramente más rápido (4.31 segundos) en comparación con Ethernet (4.46 segundos). Para 'UV' y 'VS', los tiempos de respuesta son prácticamente idénticos entre las dos conexiones, destacando una uniformidad en el rendimiento con tiempos de 8.03 segundos para Wi-Fi y 8.02 segundos para Ethernet en 'UV', y 10.24 segundos para ambos en 'VS'. Sin embargo, se observa una variación significativa en 'HV', donde Ethernet supera a Wi-Fi reduciendo el tiempo de respuesta a 8.03 segundos frente a los 14.10 segundos de Wi-Fi. Contrariamente, en 'VH' y 'EDE', Wi-Fi es más rápido con tiempos de 12.87 segundos y 8.03 segundos respectivamente, mientras que Ethernet registra tiempos más prolongados de 17.51 segundos en ambas categorías.

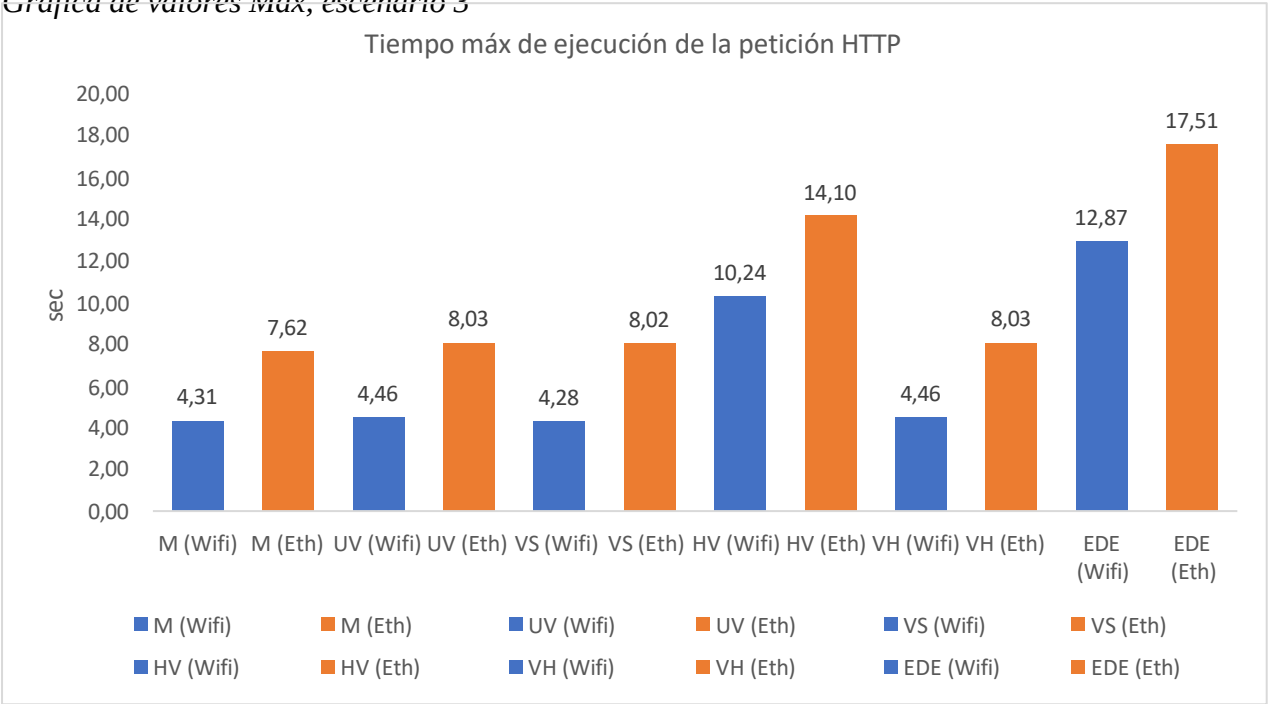
Los tiempos de respuesta varían significativamente entre Wi-Fi y Ethernet según la categoría, sin que un tipo de conexión demuestre ser uniformemente superior en todas las situaciones. Ethernet ofrece ventajas en 'HV', mientras que Wi-Fi es preferible en 'VH' y 'EDE' por sus tiempos de respuesta más cortos. Estos resultados sugieren que la selección entre Wi-

Fi y Ethernet debería considerar las características específicas de la tarea y las preferencias de eficiencia temporal.

En conclusión, el peor tiempo de escenario como respuesta fue en EDE mostrando un tiempo máximo de ejecución con 17.51 segundos, sugiriendo que la petición estaba captando varias tareas que no pudieron responder al cliente.

Figura 40

Gráfica de valores Max, escenario 3



Nota. Elaborado por: Los autores

Analizando la **Figura 40** y **Figura 41** se puede notar que la petición de EDE es la causante de que el servidor detenga su proceso ya que presentó un alto rendimiento, indicando a que petición, el servidor intentó atender a todos los usuarios. Con un tiempo máximo de 17,51 y una tasa de datos de 12,12 Kb/sec indica cuanto de carga el computador presentaba. A pesar de ser una tasa de flujo bajo en comparación a la petición **UV** el tiempo de solicitud máximo es considerablemente alto.

La **Figura 41**, muestra un alto volumen en recepción para UV (Ethernet). La configuración UV (Eth) muestra una recepción de datos excepcionalmente alta con 69,63 kB/sec, lo que puede indicar que las operaciones asociadas a esta configuración, como el acceso al último video, requieren una cantidad sustancial de datos entrantes.

- '**M**' muestra que la petición HTTP se completa más rápidamente en Wi-Fi con un tiempo de 4.31 segundos, en comparación con Ethernet que toma más tiempo con 7.62 segundos. Esto sugiere que Wi-Fi es más eficiente para esta categoría.
- Para '**UV**', se observa un tiempo de respuesta de 4.46 segundos para completar una petición HTTP en Wi-Fi, mientras que Ethernet muestra un tiempo mayor de 8.03 segundos, lo que indica una ventaja en la velocidad de respuesta para Wi-Fi.
- En '**VS**', la petición HTTP se completa en un tiempo de 4.28 segundos en Wi-Fi y en 8.02 segundos en Ethernet, resaltando nuevamente la mayor rapidez de la conexión inalámbrica sobre la cableada.
- **'HV'** presenta una diferencia notable con Wi-Fi completando la petición HTTP en 10.24 segundos y Ethernet en 14.10 segundos, lo que evidencia una ventaja considerable para Wi-Fi en términos de velocidad.
- **'VH'** tiene un tiempo de ejecución de petición HTTP en Wi-Fi de 4.46 segundos, mientras que con Ethernet se incrementa a 8.03 segundos, confirmando la tendencia de que Wi-Fi ofrece una mejor velocidad de respuesta.

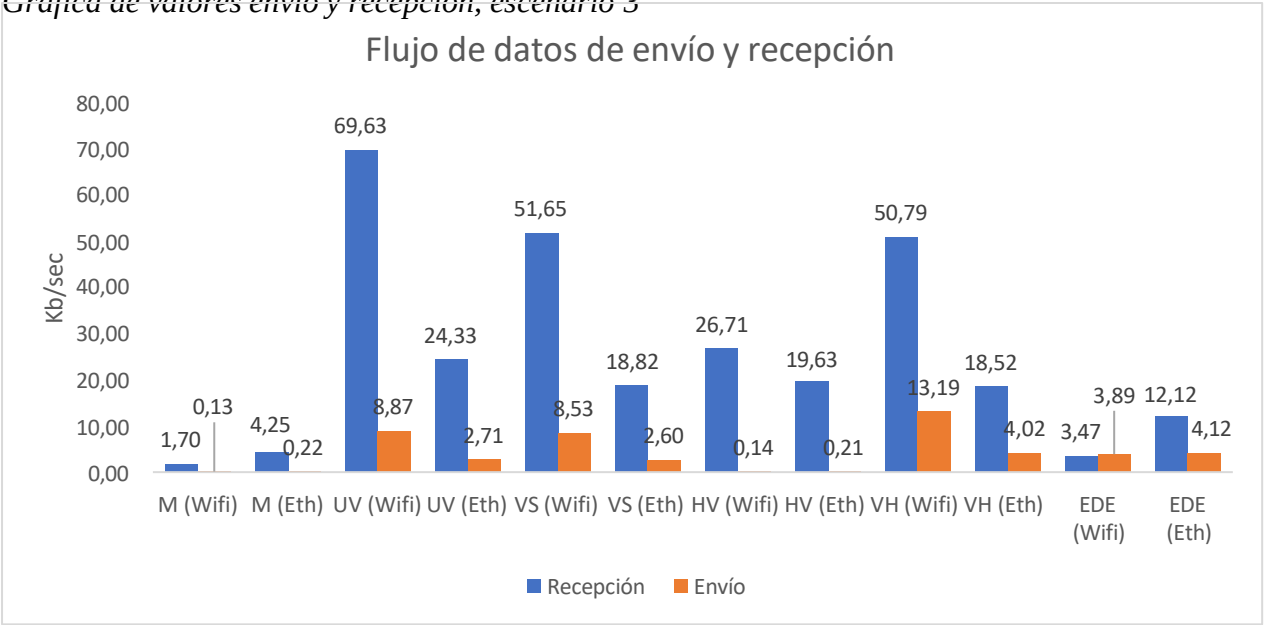
- Finalmente, '**EDE**' muestra un tiempo de ejecución para la petición HTTP en Wi-Fi de 12.87 segundos frente a un tiempo más elevado en Ethernet de 17.51 segundos, marcando este último como el tiempo más largo registrado en la gráfica para completar una petición HTTP en ambas conexiones.

Los datos indican que, para todas las categorías evaluadas ('M', 'UV', 'VS', 'HV', 'VH', 'EDE'), las peticiones HTTP se completan más rápidamente usando Wi-Fi en comparación con Ethernet. Los tiempos de ejecución en Wi-Fi varían desde los 4.28 segundos para 'VS' hasta los 12.87 segundos para 'EDE', mientras que en Ethernet, estos tiempos son más largos, desde 7.62 segundos para 'M' hasta 17.51 segundos para 'EDE'. Esta consistencia en la velocidad de Wi-Fi sobre Ethernet sugiere que, para las peticiones HTTP evaluadas, la conexión inalámbrica ofrece una eficiencia en términos de tiempo de respuesta.

Por lo tanto, los datos indican que en la mayoría de las categorías evaluadas, Wi-Fi ofrece un tiempo de respuesta más rápido para completar una petición HTTP en comparación con Ethernet.

Figura 41.

Gráfica de valores envío y recepción, escenario 3



Nota. Elaborado por: Los autores

En la **Figura 42** se confirma que el causante de que el servidor se detenga es por la petición **EDE** con una taza de 17,21 solicitudes por minuto, indicando cuanta carga representa esta solicitud para el computador, en comparación el resto de peticiones esta tiende ser hasta 20 veces más alto.

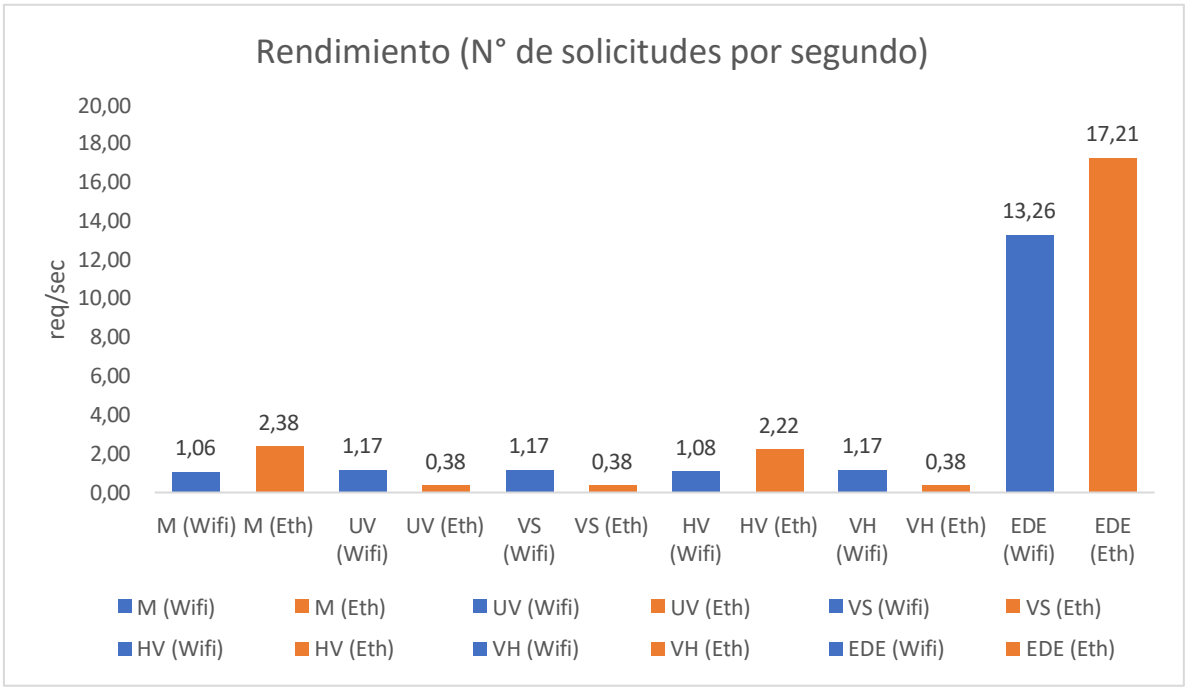
- Para la solicitud tipo 'M', Wi-Fi maneja 1.06 solicitudes por segundo, y bajo Ethernet, este número aumenta a 2.38 solicitudes por segundo, mostrando mejor rendimiento en Ethernet.
- 'UV' registra 1.17 solicitudes por segundo en Wi-Fi, mientras que Ethernet muestra una disminución significativa con 0.38 solicitudes por segundo.
- 'VS' mantiene un rendimiento constante en Wi-Fi con 1.17 solicitudes por segundo, que disminuye a 0.38 solicitudes por segundo en Ethernet.
- En el caso de 'HV', Wi-Fi logra 1.08 solicitudes por segundo frente a un rendimiento más alto en Ethernet con 2.22 solicitudes por segundo.
- Por último, 'EDE' sobresale con 13.26 solicitudes por segundo en Wi-Fi y supera esta

cifra en Ethernet, alcanzando 17.21 solicitudes por segundo, lo que indica un rendimiento superior bajo Ethernet cuando hay una alta demanda de usuarios.

Los datos revelan diferencias notables en el número de solicitudes manejadas por segundo entre conexiones Wi-Fi y Ethernet para varios tipos de solicitudes ('M', 'UV', 'VS', 'HV', 'EDE'). Para 'M' y 'HV', Ethernet muestra un rendimiento superior, con 2.38 y 2.22 solicitudes por segundo respectivamente, en comparación con Wi-Fi. Sin embargo, para 'UV' y 'VS', Wi-Fi sobresale con 1.17 solicitudes por segundo, mientras que Ethernet muestra una disminución significativa a 0.38 solicitudes por segundo. En el caso de 'EDE', tanto Wi-Fi como Ethernet manejan un alto número de solicitudes, con Ethernet mostrando una capacidad superior (17.21 solicitudes por segundo) en comparación con Wi-Fi (13.26 solicitudes por segundo).

Estos resultados sugieren que la eficacia de Wi-Fi y Ethernet puede variar ampliamente dependiendo del tipo específico de solicitud, con Ethernet ofreciendo un rendimiento especialmente alto en situaciones de alta demanda de usuarios, como se ve en 'EDE'. La elección entre Wi-Fi y Ethernet podría, por lo tanto, considerarse en función del tipo específico de carga de trabajo y el volumen de solicitudes esperado.

Figura 42.
Gráfica de valores de rendimiento, escenario 3



Nota. Elaborado por: Los autores

Con base a la **Figura 40**, **Figura 41** y **Figura 42** se puede notar que la petición de EDE con cable ethernet es la que mayor tiempo tarda en terminar la concurrencia, con tiempo de 17,51 segundos, sin embargo, es la que mayor rendimiento presenta con un valor de 17,21 solicitudes por segundo. La petición UV con wifi recibe la mayor cantidad de datos, mientras que la petición VH con wifi envía la mayor cantidad de datos.

3.1.7 *Resumen de pruebas de concurrencia*

A continuación, en la Tabla 14. Resumen de concurrencialos valores promedio de cada prueba y escenario tomando en cuenta las métricas de mayor impacto, que son rendimiento y tráfico de datos (envío).

Tabla 14

Resumen de concurrencia

Métrica	Wifi			Ethernet		
	30 min	60 min	120 minutos	30 min	60 min	120 minutos
Rendimiento	0,45	2,63	3,15	0,44	4,43	3,82
Trafico	1,57	10,01	5,79	1,40	3,22	2,31

Nota. Elaborado por: Los autores

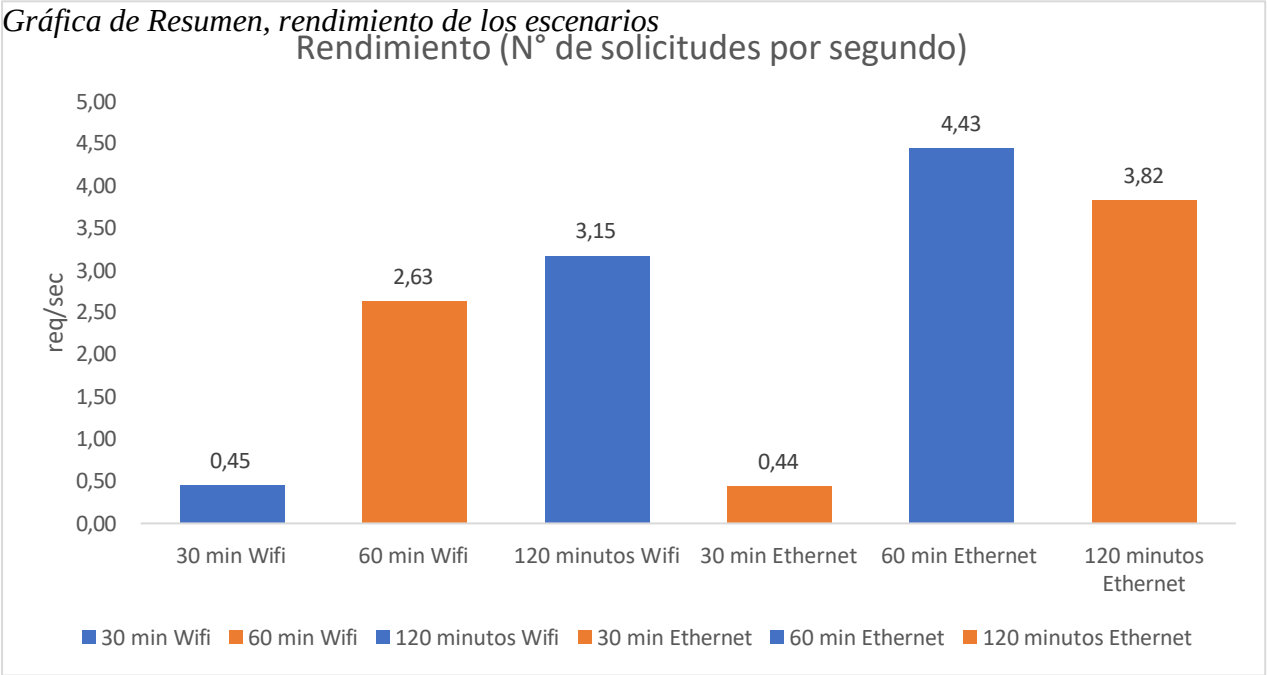
En la **Figura 43.** Gráfica de Resumen, rendimiento de los escenarios los escenarios de ethernet son lo que presentan un mayor rendimiento de solicitudes, llegando a la conclusión que el uso de cable ethernet es la mejor opción, pero tener el más alto desenvolvimiento del servidor.

La **Figura 43** muestra el rendimiento en términos de número de solicitudes por segundo en diferentes intervalos de tiempo para conexiones Wi-Fi y Ethernet. En el caso de Wi-Fi, se observa un rendimiento inicial de 0.45 solicitudes por segundo a los 30 minutos, que se incrementa a 2.63 solicitudes por segundo al pasar una hora y alcanza las 3.15 solicitudes por segundo a los 120 minutos. Para Ethernet, el rendimiento comienza con 0.44 solicitudes por segundo a los 30 minutos, aumenta significativamente a 4.43 solicitudes por segundo a los 60 minutos y luego disminuye a 3.82 solicitudes por segundo a los 120 minutos. Estos datos reflejan cómo el rendimiento de la red puede variar no solo con el tipo de conexión utilizada sino también con el tiempo.

Finalmente, a lo largo del tiempo, Ethernet muestra un rendimiento generalmente más alto en la gestión de solicitudes en comparación con Wi-Fi, con un pico más alto a los 60 minutos.

Figura 43

Gráfica de Resumen, rendimiento de los escenarios



Nota. Elaborado por: Los autores

En cuanto a tráfico de datos se puede notar en la **¡Error! No se encuentra el origen de la referencia.** que el uso de wifi como medio de comunicación es el que más demanda de envío de datos tiene.

La **Figura 44** destaca cómo el uso de datos varía según el tipo de conexión y la duración del uso. En Wi-Fi, el tráfico comienza relativamente bajo a 1,57 kB/s tras 30 minutos de uso. Este valor se incrementa notablemente a 10,01 kB/s después de una hora, señalando un período de alta demanda de datos. Sin embargo, al continuar el uso hasta las dos horas, el tráfico disminuye a 5,79 kB/s. Por otro lado, Ethernet inicia con un tráfico similarmente bajo de 1,40 kB/s a los 30 minutos, pero su aumento es menos pronunciado, alcanzando 3,22 kB/s a la hora

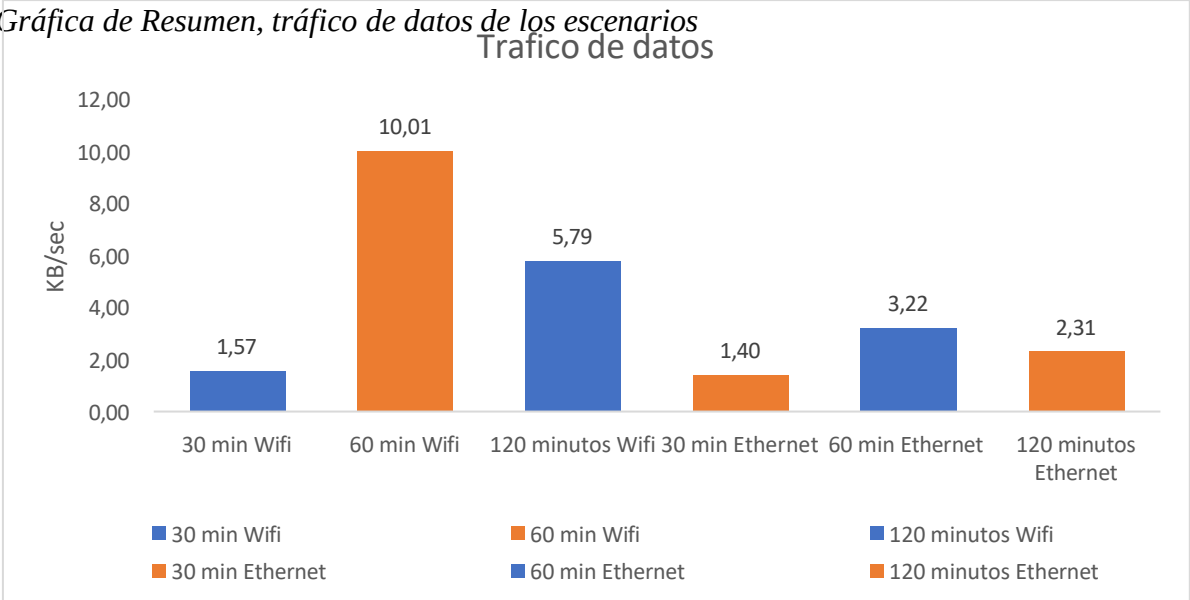
y luego reduciéndose a 2,31 kB/s a las dos horas. Esta información subraya un patrón de mayor consumo de datos en Wi-Fi, especialmente notable después de una hora de uso, en comparación con Ethernet, que presenta un incremento más moderado en la demanda de datos a lo largo del tiempo.

En base a los datos proporcionados, se puede concluir que después de una hora de uso, Wi-Fi experimenta un aumento notable en el consumo de datos, alcanzando un tráfico de 10,01 kB/s, mientras que Ethernet muestra un aumento más moderado con 3,22 kB/s. Sin embargo, a medida que pasa el tiempo, el tráfico en Wi-Fi disminuye a 5,79 kB/s después de dos horas, mientras que en Ethernet se reduce a 2,31 kB/s en el mismo período.

Esto indica que, aunque Wi-Fi inicialmente muestra un mayor consumo de datos con un pico de 10,01 kB/s, su tráfico tiende a disminuir a medida que pasa el tiempo. Por otro lado, Ethernet presenta un incremento más gradual en la demanda de datos, llegando a 3,22 kB/s después de una hora y luego reduciéndose a 2,31 kB/s después de dos horas.

Figura 44

Gráfica de Resumen, tráfico de datos de los escenarios



Nota. Elaborado por: Los autores

3.1.8 *Análisis de financiero*

En el marco del presente proyecto, se realizará una gestión exhaustiva de todos sus componentes con el objetivo primordial de maximizar la rentabilidad integral del sistema. Dicho enfoque nos permitirá cuantificar los resultados de ganancia de una manera más efectiva, haciendo uso de equipos que, a pesar de contar con recursos limitados, no comprometen su desempeño funcional. Para alcanzar este fin, aplicaremos un análisis de costo-beneficio, fundamentado en metodologías probadas y en la aplicación de una fórmula específica para el cálculo de la ratio costo-beneficio, conforme a los estudios realizados por (Nancy Rodríguez, 2023).

Donde:

$$EC1. \quad \frac{Costo}{Benefico} = \frac{VAI}{VAC}$$

Este enfoque nos permitirá identificar las opciones más rentables, asegurando así la viabilidad económica del proyecto y la optimización de los recursos disponibles.

Costo en equipos y materiales. En la **Tabla 15** se muestra que la inversión en hardware y software es crítica en el desglose de precios siendo un enfoque meticuloso para la gestión de recursos, asegurando que cada elemento contribuya de manera efectiva al funcionamiento de cámaras externas en zonas residenciales.

Tabla 15

Tabla de costos de los materiales

Dispositivos a utilizar para el desarrollo del proyecto		
Unidad	**Componentes**	**Costo**
1	Módulo ESP32 WROOM-32	12$
1	Cámara de vigilancia	30$
1	Resistencia 1KΩ	0.10$
1	Fuente regulada 3.3 a 12 [V]	3,5$
1	Router Wifi	25$
1	Computador portátil	800$
1	Placa fenólica de baquelita con cobre	0,75$
1	Teléfono móvil con sistema operativo Android	340$
	Total	1211,35$

Nota. Costo total del proyecto - Elaborado por: Los autores

En la Tabla 16. Tabla de costos de los materiale se ha calculado el costo de la mano de obra, basado en el tiempo empleado por dos trabajadores, ambos dedicando 90 horas al proyecto a un precio de $2.87 por hora. Esto resulta en un costo total de mano de obra de $516.6.

Tabla 16

Mano de Obra

Trabajadores	**Tiempo Empleado**	**Precio por hora**	**Total**
Trabajador 1	90	2,87	258,3
Trabajador 2	90	2,87	258,3
		TOTAL	516,6

Nota. Duración y costo del desarrollo del proyecto - Elaborado por: Los autores

3.1.9 *Estimación de gastos*

Tal como se notó con anterioridad en la Tabla 15, consiguiendo la suma de todos los elementos ocupados. El costo de mano de obra en la Tabla 16 se agrupa en un margen de rentabilidad del

25% para estimar gastos.

EC2.	1211,35+516,6	=	$1727,95	**Gastos**
EC3.	1727,95 * 25%	=	$431,99	**Rentabilidad**
EC4.	**1727,95+431,99**	**=**	**$2159,94**	**Precio por unidad**

Cálculo del beneficio. Con la proyección de ventas realizada, se espera obtener ingresos totales de $2159,94. Este cálculo se fundamenta en la expectativa de que cada unidad producida generará un retorno financiero acorde con el precio establecido por unidad.

Venta de unidades	150
EC5.	431,99*150 = $64798,5

Cálculo de relación costo beneficio. El cálculo se efectúa comparando el Valor Actual de Ingresos (VAI) con el Valor Actual de Costos (VAC), ambos ajustados por el número de unidades (Nancy Rodríguez, 2023). El precio de venta proyectado establece $2,159.94 por cada unidad y se anticipa la venta de 150 unidades en un año. La inversión inicial, que engloba tanto los dispositivos como la mano de obra, se calcula en $1,727.95. Aplicando la fórmula se obtiene.

EC6. $\frac{Costo}{Beneficio} = \frac{VAI}{VAC}$

EC7. $\frac{Costo}{Beneficio} = \frac{2159{,}94*150}{1727{,}95*150}$

EC8. $\frac{Costo}{Beneficio} = 1{,}25$

Tras realizar los cálculos necesarios, obtenemos una relación costo-beneficio de 1.25. Esta cifra, al ser superior a 1, indica que, por cada dólar invertido, se espera obtener un retorno de $1.25 favorable para el proyecto.

CONCLUSIONES

La implementación de un servidor en NodeJS para procesar video en tiempo real y almacenar videos muestra la capacidad del sistema de aprovechar tecnologías avanzadas en cámaras de vigilancia. La conversión de video de protocolo RTSP a HTTP permite la integración de cámaras con funcionalidades como visión nocturna en color y detección de movimiento, cumpliendo con el objetivo de explorar y utilizar cámaras de vigilancia multifuncionales.

El uso del ESP32 para monitorear sensores y enviar datos a un servidor central refleja una estrategia eficaz para seleccionar e integrar sensores adecuados en el sistema de seguridad perimetral. Esto subraya la importancia de elegir hardware y tecnología adecuados para fortalecer la seguridad, especialmente en áreas críticas como los cercos eléctricos, alineándose con el objetivo de mejorar la efectividad del sistema de seguridad perimetral.

Los resultados de las pruebas de calidad de código y las pruebas de concurrencia utilizando SonarCloud y Apache JMeter respectivamente, demuestran la fiabilidad y el alto rendimiento del sistema. La corrección de errores y la optimización del código han resultado en una plataforma robusta y segura, capaz de manejar múltiples solicitudes simultáneamente sin comprometer la seguridad o la funcionalidad. Esto es crucial para mantener la integridad del sistema de seguridad perimetral y asegurar una supervisión efectiva a través de la aplicación móvil propuesta.

La identificación y corrección de errores, la optimización del rendimiento y la garantía de la seguridad de los datos son aspectos que contribuyen directamente a la confiabilidad y eficacia del sistema de seguridad perimetral. Este enfoque proactivo hacia la calidad del código y la capacidad de respuesta del servidor no solo mejora la experiencia del usuario final al utilizar la aplicación móvil para la supervisión de seguridad, sino que también asegura que el sistema esté preparado para responder de manera efectiva ante intentos de intrusión, cumpliendo así con el objetivo de desarrollar una aplicación móvil para una gestión eficiente de la seguridad residencial

RECOMENDACIONES

Se recomienda para futuros trabajos implementar sistemas de vigilancia, adicionando lógicas de reconocimiento de movimiento, para fortalecer el sistema de seguridad. Usando herramientas como Python para el reconocimiento de movimiento y de imágenes, permitiendo así tener un sistema de vigilancia más inteligente.

Se sugiere implementar una interfaz gráfica dentro del mismo servidor para tener una interacción con el usuario, en cuanto a información del video, información de todos los dispositivos IoT que se utilicen, usando plataformas como Angular para mejorar las páginas web y separar el Backend del Frontend, ya que en este proyecto de unieron los dos en uno solo entorno.

Se recomienda evaluar cuidadosamente las cámaras de seguridad antes de su implementación, considerando tanto sus capacidades técnicas como posibles restricciones de software impuestas por el fabricante. La selección debe enfocarse en dispositivos que ofrezcan funcionalidades avanzadas, como visión nocturna y detección de movimiento, sin sacrificar la flexibilidad para futuras actualizaciones o integraciones con sistemas de gestión de seguridad más completos.

BIBLIOGRAFÍA

Arduino. (2024). Arduino Education. https://www.arduino.cc/education

Azurin Villanque, D. A., & Núñez Neyra, L. G. (2022). *Diseño de un sistema de videovigilancia utilizando IP inalámbrico y capacidad IVS de cercos virtuales para el condominio La Isla Asia*, 2022. [Tesis de pregrado, Universidad Ricardo Palma, Lima, Perú]. https://hdl.handle.net/20.500.14138/6156

Alasdair, A. (2022). Thermokon Home of Sensor Technology.

Contreras Clavijo, S. A., & Rojano Mueses, X. G. (2023). *Sistema inteligente de monitoreo y control para la planta de tratamiento de agua potable "El Carrizal-Salcedo" basado en IOT e inteligencia artificial.* [Tesis de pregrado, Universidad Técnica de Ambato. Facultad de Ingeniería en Sistemas, Electrónica e Industrial. Carrera de Telecomunicaciones, Ambato, Ecuador]. https://repositorio.uta.edu.ec/jspui/handle/123456789/38419

Cruz Cáceres, L. (2020). *Sistema de seguridad basado en el internet de las cosas para viviendas urbanas*. [Tesis de pregrado, Universidad Mayor de San Andrés, La Paz, Bolivia]. http://repositorio.umsa.bo/xmlui/handle/123456789/27706

Cruz Lovato, J. J. (2018). *Análisis y diseño de una red de video vigilancia por medio de wifi en la ciudadela "Portete de Tarqui*". [Tesis de pregrado, Universidad de Guayaquil, Guayaquil]. http://repositorio.ug.edu.ec/handle/redug/32566

Damien, G. (2023). *MicroPython*. https://micropython.org/

Dereniak, E., & Boreman, G. (2006). *Infrared detectors and systems*. John Wiley & Sons Inc. https://www.wiley.com/en-us/Infrared+Detectors+and+Systems-p-9780471122098

Farahani, S. (2008). *ZigBee Wireless Networks and Transceivers.* Oxford, Inglaterra. ELSEVIER. https://www.scirp.org/reference/referencespapers?referenceid=2210601

Flanagan, D. (2020). JavaScript: The Definitive Guide. Estados Unidos: O'Reilly Media Inc. https://www.oreilly.com/library/view/javascript-the-definitive/9781449393854/

Gourley, D., & Totty, B. (2002). *HTTP: The Definitive Guide.* Sebastopol, Rusia. O'Reilly Media Inc. https://www.oreilly.com/library/view/http-the-definitive/1565925092/index.html

Heydon, R. (2013). *Bluetooth Low Energy.* Estados Unidos: Pearson Education. https://search.library.wisc.edu/catalog/9913039541102121

Hillar, G. (2017). *MQTT Essentials - A Lightweight IoT Protocol.* Birmingham, Reino Unido: Packt Publishing Ltd. https://es.everand.com/book/382268152/MQTT-Essentials-A-Lightweight-IoT-Protocol

Knapp, E., & Langill, J. T. (2015). *Industrial network security: Securing critical infrastructure networks for smart grid, SCADA, and other industrial control systems.* Waltham, Estados Unidos: ELSEVIER. https://es.everand.com/book/287529286/Industrial-Network-Security-Securing-Critical-Infrastructure-Networks-for-Smart-Grid-SCADA-and-Other-Industrial-Control-Systems

Kovatsch, M. (2013). *CoAP for the Web of Things*: *From tiny resource-constrained devices to the web browser.* [Institute for Pervasive, Zurich, Suiza] https://dl.acm.org/doi/10.1145/2494091.2497583

Lead. (2019, December 11). *Cercos eléctricos*. Puertas La Campana. http://www.puertaslacampana.com/productos/cercos-electricos/

Medina Espinosa, C. A. (2017). Diseño de un sistema domótico de seguridad, video vigilancia, control de acceso y confort para las oficinas de Grupo Mega. [Tesis de pregrado, Universidad de las Américas, Quito, Ecuador]. http://dspace.udla.edu.ec/handle/33000/7017

Monteros Mejía, J. L. (2015). *Diseño de un sistema de video-vigilancia inalámbrico para la ciudad de Cayambe.* [Tesis de pregrado, Escuela Politécnica Nacional, Quito, Ecuador].http://bibdigital.epn.edu.ec/handle/15000/10648

Navarro. (2021). *Cómo instalar un cerco eléctrico paso a paso*. https://cercoselectricos.pe/blog/como-instalar-un-cerco-electrico-paso-a-paso/

Oñate Miranda, F. P. (2020). *Diseño y construcción de nodos inteligentes para detección de armas*

dentro de una red de video-vigilancia utilizando visión artificial. [Tesis de pregrado, Escuela Superior Politécnica de Chimborazo, Chimborazo, Ecuador]. http://dspace.espoch.edu.ec/handle/123456789/13783

Ojeda Crespo, L. G., & Cabrera Mejía, J. B. (2020). *Análisis de las estrategias aplicadas en el desarrollo de sistemas domóticos de seguridad.* Dominio de las Ciencias. *6*(3), 342-363. https://dialnet.unirioja.es/servlet/articulo?codigo=7539682

Paetz, C. (2018). *Z-Wave Essentials. Zwickau*, Alemania: Interoperability in Smart Homes. https://www.digitalhomesystems.com.au/orders/all-products/zwavebasicsbook145-detail

Raspberry Pi. (n.d.). More from Raspberry Pi. https://www.raspberrypi.com/

Reolink Duo 2 | Cámara con batería WiFi de doble lente 2K+ panorámica de 180 grados | Sitio web oficial de Reolink. (2023, April 3). https://reolink.com/es/product/reolink-duo/

Ribero Corzo, S., & Prieto Guerrero, Y. (2020). *Identificación de riesgos en la seguridad de la información de cámaras de vigilancia domésticas en entornos IOT.* [Tesis de pregrado, Instituto de Tecnología de Massachusetts, Massachusetts, Estados Unidos]. https://hdl.handle.net/10983/26242

Sarver, W. (2017). *The 5G Deployment Plan Handbook: Volume 1*. Estados Unidos: USA. Independently. https://www.buscalibre.ec/libro-the-5g-deployment-plan-handbook-volume-1-5g-technical-deployment-and-history-around-building-5g-and-iot-businesses-5g-deployment-handbook-libro-en-ingles/9781520485577/p/53892913

Scherz, P., & Monk, S. (2016). *Practical Electronics For Inventors*. Estados Unidos: McGraw Hill. Education. https://www.buscalibre.ec/libro-practical-electronics-for-inventors-fourth-edition/9781259587542/p/46769745

Sinclair, I. (2001). *Sensors and Transducers*. Great Britain: ELSEVIER.

Systems Espressif. (2023). *ESP32 Series Datasheet*. https://www.iberlibro.com/9780750649322/Sensors-Transducers-Sinclair-Ian-0750649321/plp

Tollervey, N. (2017). *Programming with MicroPython*. O'Reilly Media Inc.

https://www.abebooks.com/9781491972731/Programming-MicroPython-Embedded-Microcontrollers-Python-1491972734/plp

tp-link. (2024). *Tapo C100 Cámara de Seguridad Wi-Fi.* TP-Link Corporation Limitited: https://www.tp-link.com/ec/home-networking/cloud-camera/tapo-c100/

Velasco Llano, W. D. (2018). *Implementación de un prototipo de un cerco eléctrico para protección de ganado utilizando energía solar y envío de mensajes cuando exista una violación del sistema*. [Tesis de pregrado, Escuela Superior Politécnica de Chimborazo, Riobamba, Ecuador]. http://dspace.espoch.edu.ec/handle/123456789/9250

ANEXOS

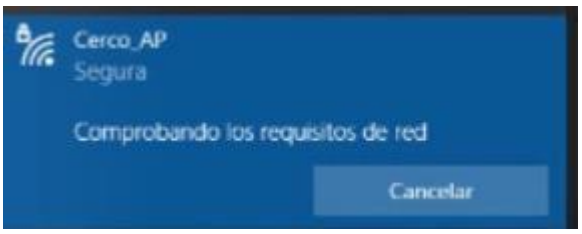
Figura 45

Desarrollo del Script Python para el módulo ESP32 en Thonny IDE

*Nota***.** Entorno de desarrollo con código Python para manejo de datos y comunicación web en un dispositivo IoT. Elaborado por: Los autores.

Figura 46

Verificación de configuración de red

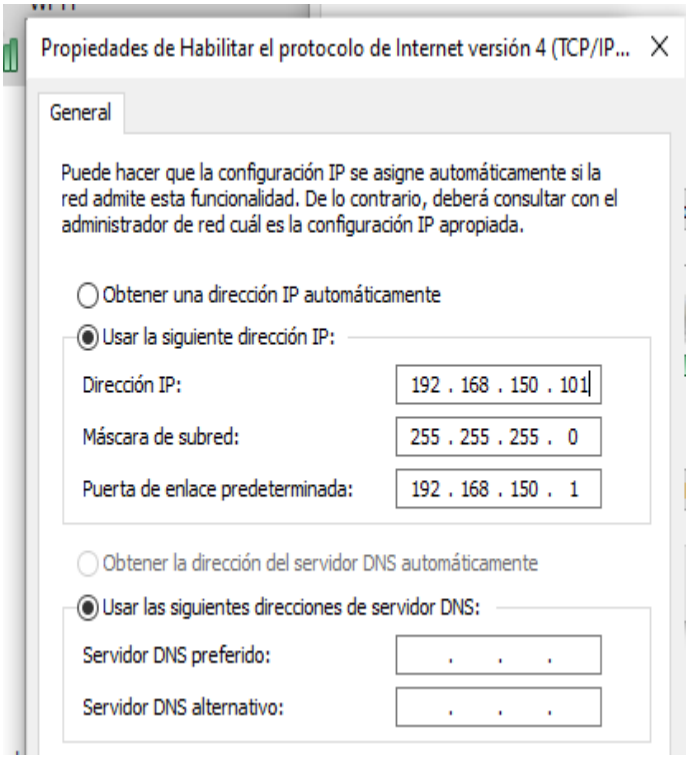


Nota. Interfaz mostrando la verificación de configuración de red.

Elaborado por: Los autores

Figura 47

Configuración de IP estática para la conexión Wi-Fi en Windows

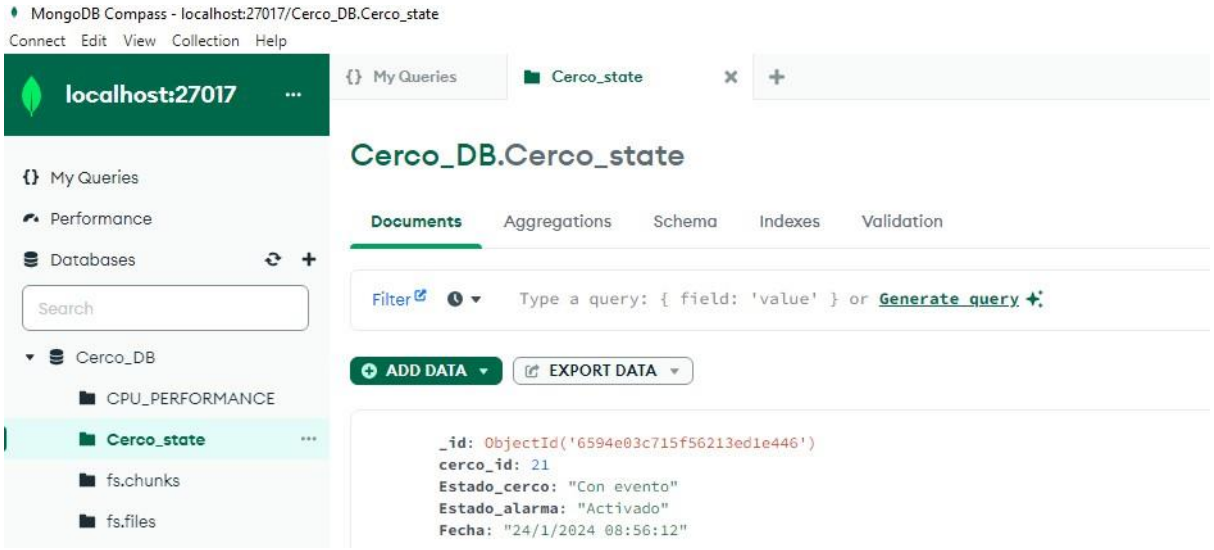


Nota. Configuración de IP estática para la conexión Wi-Fi en Windows, ajustada para el servidor del proyecto.

Elaborado por: Los autores

Figura 48

Interfaz de MongoDB Compass



Nota. Interfaz de MongoDB Compass mostrando registros de estados de eventos en la base de datos local para el seguimiento del proyecto.

Elaborado por: Los autores

Figura 49.

Ejecución del sistema con el servidor Node.js

Nota. Ejecución del servidor Node.js, para la monitorización de recursos del sistema, parte de las operaciones de backend para nuestro proyecto.

Elaborado por: Los autores

Figura 50

Captura de video con cámara Tapo C100 en el día

Nota. Imagen capturada por la cámara de seguridad Tapo C100, mostrando un entorno interno bien iluminado, como parte de las pruebas de monitoreo de nuestro proyecto.

Elaborado por: Los autores

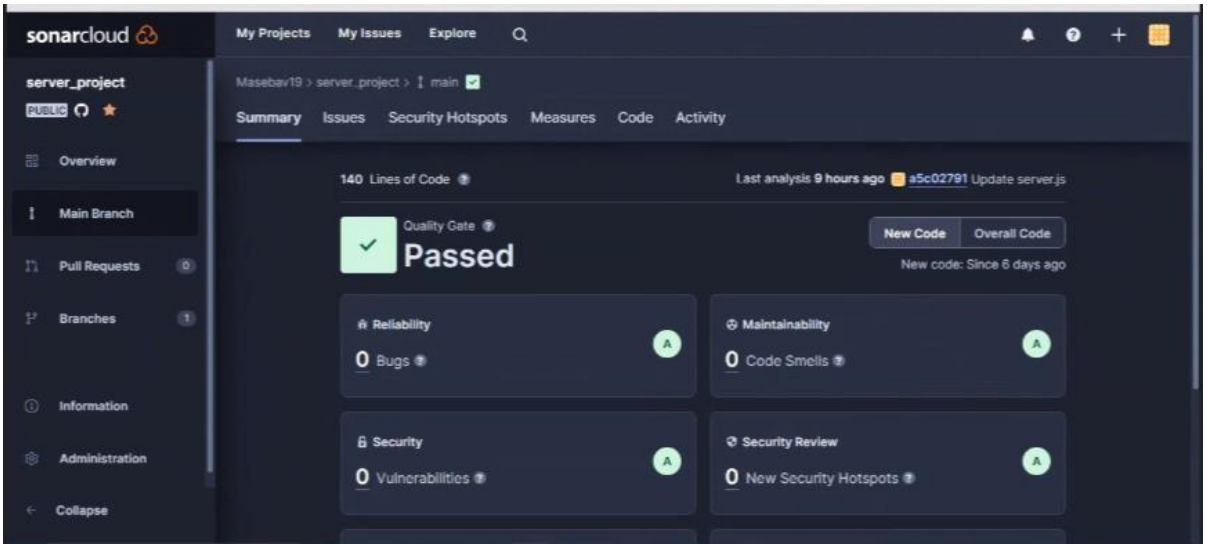
Figura 51

Captura de video con cámara Tapo C100 en la noche

Nota. Grabación nocturna de la cámara Tapo C100, demostrando su capacidad de visión nocturna en un entorno con iluminación limitada, para la fase de pruebas de seguridad.

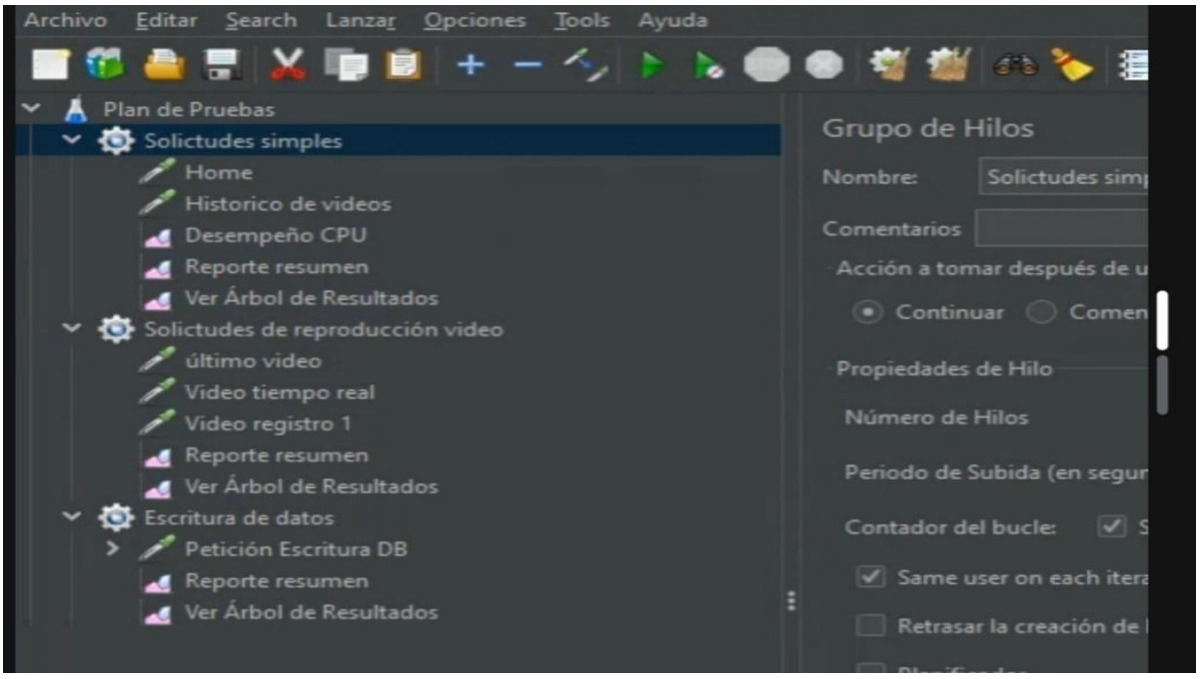
Elaborado por: Los autores

Figura 52.
Resultados de las pruebas de SonarCloud



Nota. Resultados de las pruebas de SonarCloud aplicadas al código fuente, mostrando un análisis exitoso sin errores ni vulnerabilidades, para garantizar la calidad del software de nuestro proyecto.
Elaborado por: Los autores

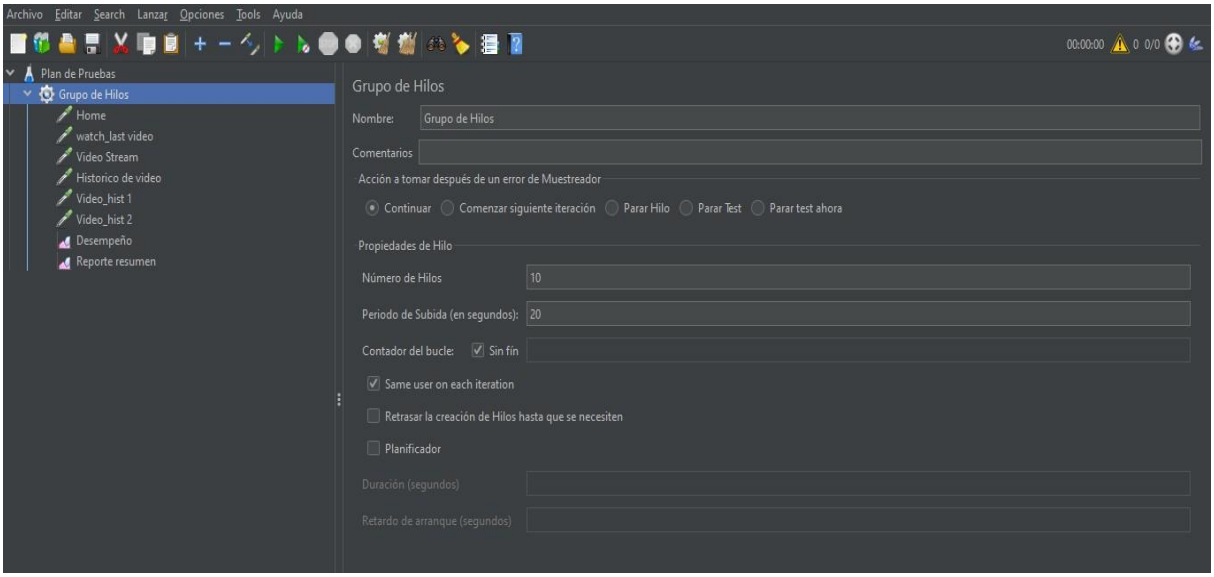
Figura 53.
Configuración de pruebas en JMeter



Nota. Configuración de pruebas en JMeter para evaluar el rendimiento y la funcionalidad de las páginas web de nuestro proyecto.
Elaborado por: Los autores

Figura 54

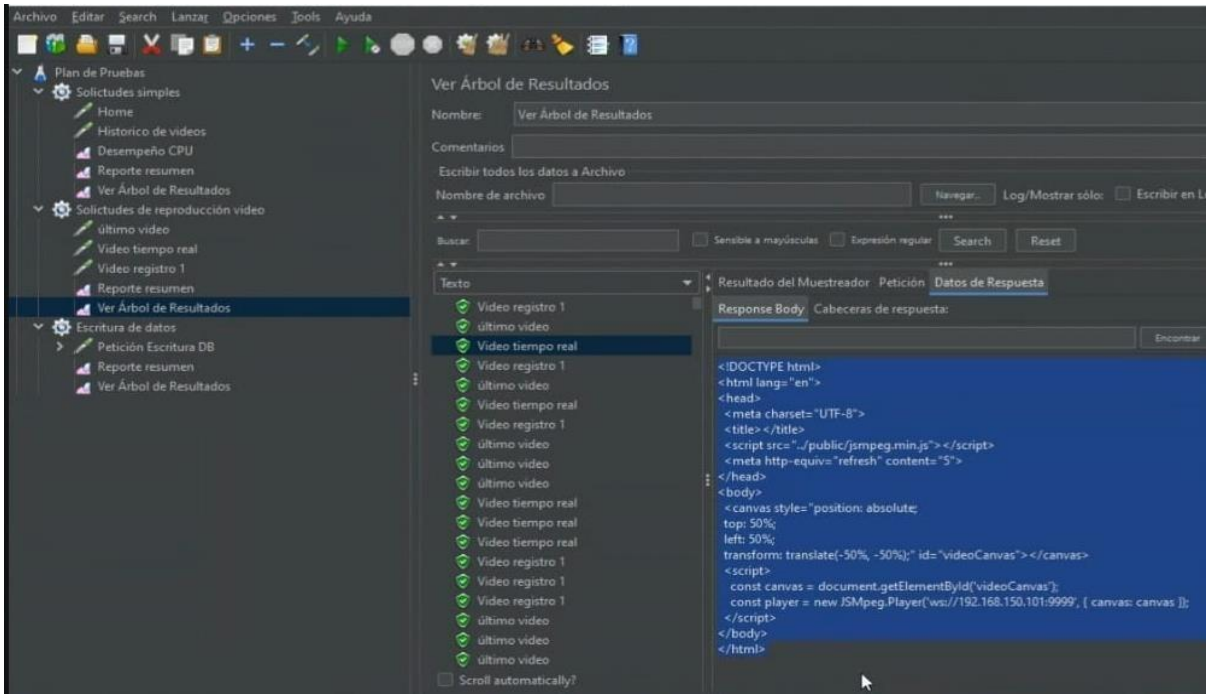
Configuración del grupo de Hilo para prueba de carga en JMETER



Nota. Vista de la configuración de un Grupo de Hilos en JMeter, utilizado para simular múltiples usuarios y probar la carga y el rendimiento de nuestras páginas web dentro del proyecto. Elaborado por: Los autores

Figura 55

Resultados de carga



Nota. Interfaz de JMeter mostrando el árbol de resultados de pruebas de carga, con un enfoque en la reproducción de vídeo para evaluar la respuesta del sistema en el proyecto. Elaborado por: Los autores