REPÚBLICA DEL ECUADOR

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE POSGRADO

MAESTRÍA EN COMPUTACIÓN CON MENCIÓN EN SEGURIDAD INFORMÁTICA

Evaluación de la seguridad de software mediante el uso de las herramientas SonarQube y OWASP ZAP: Estudio de caso en la plataforma tecnológica GastroEc.

Trabajo de Titulación previo a la obtención del Título de Magíster en

Computación con mención en seguridad informática

AUTOR

Ing. Anthony Patricio Valverde Quispe

DIRECTOR

Msc. Daisy Elizabeth Imbaquingo Esparza

Ibarra, junio 2024

DEDICATORIA

Dedico este trabajo a todas las personas que me han acompañado durante el desarrollo de este proyecto a mi madre Bety Valverde por su apoyo, a mi tío Javier Valverde por su guía y compartir conmigo su sabiduría y la Licenciada Andrea Córdova por acompañarme y apoyarme durante todo este proceso académico

Anthony Valverde

AGRADECIMIENTOS

Agradezco a mi familia y amigos que me han acompañado en todo momento brindándome su apoyo y consejos.

Agradezco a mi directora de proyecto, Msc. Daisy Imbaquingo quien ha sabido guiarme y aconsejarme durante la realización de este proyecto

Anthony Valverde

UNIVERSIDAD TÉCNICA DEL NORTE

DIRECCIÓN DE BIBLIOTECA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	1004131916		
APELLIDOS Y NOMBRES:	Valverde Quispe Anthony Patricio		
DIRECCIÓN:	Pastora alomia y Venezuela		
EMAIL:	apvalverdeq@utn.edu.ec		
TELÉFONO FIJO:		**TELÉFONO MÓVIL:**	0967591457

DATOS DE LA OBRA	
TÍTULO:	Evaluación de la seguridad de software mediante el uso de las herramientas SonarQube y OWASP ZAP: Estudio de caso en la plataforma tecnológica GastroEc.
AUTOR (ES):	Valverde Quispe Anthony Patricio
FECHA:	18/06/2024
PROGRAMA:	**( ) PREGRADO (x) POSGRADO**
TITULO POR EL QUE OPTA:	Magíster en Computación con mención en seguridad informática
DIRECTOR/ASESOR	Msc. Daisy Imbaquingo, Msc José Jácome

2. CONSTANCIAS

El autor (es) manifiesta (n) que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es (son) el (los) titular (es) de los derechos patrimoniales, por lo que asume (n) la responsabilidad sobre el contenido de la misma y saldrá (n) en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 18 días del mes de junio de 2024.

EL AUTOR:

Anthony Patricio Valverde Quispe

APROBACIÓN DEL TUTOR

Yo Dra. Daisy Elizabeth Imbaquingo Esparza en calidad de director de la tesis titulada: **"Evaluación de la seguridad de software mediante el uso de las herramientas SonarQube y OWASP ZAP: Estudio de caso en la plataforma tecnológica GastroEc.**" de la autoría del Ing. Anthony Patricio Valverde Quispe, para optar por el grado de Magister en Computación con Mención en Seguridad Informática, doy fé de que dicho trabajo reúne los requisitos y méritos suficientes para ser sometidos a presentación privada y evaluación por parte del jurado examinador que se designe.

En la ciudad de Ibarra, a los 18 días del mes de junio del 2024.

Lo certifico

1002873048 DAISY ELIZABETH IMBAQUINGO ESPARZA

Firmado digitalmente por 1002873048 DAISY ELIZABETH IMBAQUINGO ESPARZA
Fecha: 2024.06.18 12:57:54 -05'00'

Dra. Daisy Elizabeth Imbaquingo Esparza

DIRECTOR DE TESIS

Ibarra, 28 de septiembre del 2023

AUTORIZACIÓN

Yo, **Ivan Bravo**, titular de la cédula de identidad número **1002379327**, en mi calidad de **CEO** de **Ikono.Ec**, por el presente documento autorizo expresamente la realización de un análisis forense informático del código fuente y de la **Plataforma Web de Gestión de Restaurantes Gastro-EC** con el propósito de identificar posibles vulnerabilidades en la seguridad.

El análisis será llevado a cabo por el **Ing. Anthony Patricio Valverde** titular de la cédula número **1004131916** como parte del trabajo final de la **Tesis de Maestría en "Computación mención Seguridad Informática" de la Universidad Técnica del Norte**, en su calidad de maestrante.

Este análisis tiene como objetivo principal garantizar la integridad, confidencialidad y disponibilidad de la información manejada por la aplicación mencionada. Además, busca identificar y corregir posibles vulnerabilidades que puedan comprometer la seguridad del sistema.

La autorización abarca el acceso al código fuente, así como a la infraestructura de la aplicación web en entorno de prueba para llevar a cabo una evaluación completa. El auditor se compromete a mantener la confidencialidad de cualquier información sensible a la que tenga acceso durante el proceso de análisis.

Esta autorización tiene validez desde la fecha de emisión hasta el 27 de octubre del 2023, a menos que sea revocada previamente por escrito.

IVAN FERNANDO BRAVO MARINO

Iván Bravo Mariño
Director Ejecutivo
Telf. 0984-264-034
Email: ibravo@ikono.ec
www.ikono.ec

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El presente trabajo se centra en una evaluación de seguridad aplicada a la plataforma web, enfocada a la protección de uno de los activos más importante de una empresa la información, la cual se vuelve un objetivo para los ciberdelincuentes, por tal razón en la actualidad existe mayor énfasis en proteger tal activo y minimizar el riesgo de que esta sea vulnerada.

Ahora podemos encontrar varias herramientas que nos ayudan a evaluar software y nos brindan ayuda para minimizar estos riesgos, como las utilizadas en este proyecto que son Owasp Zap y SonarQube, herramientas de uso libre que se encarga de encontrar vulnerabilidades en sistema a nivel de código como de producción.

Se tomó como estudio de caso para aplicar la evaluación de seguridad a la plataforma tecnológica de restaurantes GastroEc, empezando con un estudio de las herramientas Owasp Zap y SonarQube para proceder a realizar la evaluación de vulnerabilidades, en las que se encontraron ciertas alertas a tomar en cuenta, una vez localizadas las vulnerabilidades se recomendó ciertas medidas a tomar, las que se aplicaron y resultaron ser eficientes, volviendo a la plataforma más segura respecto a cómo era en un principio y demostrando que las herramientas resultaron de gran utilidad para este proyecto.

ABSTRACT

This work focuses on a security evaluation applied to the web platform, focused on the protection of one of the most important assets of a company, information, which becomes a target for cybercriminals, for this reason there is currently greater emphasis on protecting such an asset and minimizing the risk of it being violated.

Now we can find several tools that help us evaluate software and help us minimize these risks, such as those used in this project, which are Owasp Zap and SonarQube, free-to-use tools that are responsible for finding vulnerabilities in the system at the code level such as of production.

It was taken as a case study to apply the security evaluation to the technological platform of GastroEc restaurants, starting with a study of the Owasp Zap and SonarQube tools to proceed to carry out the vulnerability evaluation, in which certain alerts were found to be taken into account. Account, once the vulnerabilities were located, certain measures were recommended to be taken, which were applied and turned out to be efficient, making the platform more secure compared to how it was initially and demonstrating that the tools were very useful for this project.

CAPÍTULO I
EL PROBLEMA

1.1. Problema de investigación

Toda empresa en la actualidad tiene un objetivo principal en común que es de proteger su activo más importante que es la información que maneja su organización, esto implica en tomar en cuenta cualquier forma o manera que la empresa use sus datos (sistemas informáticos, plataformas web, transacciones etc.), en cualquier empresa es posible que se generen amenazas que afecten a sus activos provocando vulnerabilidades, estas vulnerabilidades pueden centrarse en afectar los datos, el hardware o software de la empresa para obtener un objetivo en específico o afectar el funcionamientos de sus procesos (Palacios, 2020)

Las plataformas tecnológicas hacen uso de varias herramientas para poder ofrecer un mejor servicio como es el de las aplicaciones informáticas que en la actualidad abarcan un gran ámbito de actividades adaptándose a cualquier idea de negocio para la mejora de procesos y servicios, siendo así que manejan una gran cantidad de datos e información, además de que en la actualidad la mayoría de los sistemas que podemos encontrar están enfocando su uso hacia plataformas web, por lo que la seguridad en estas aplicaciones es una prioridad para las empresas y un objetivo a vulnerar para hackers o ciberdelincuentes.

Un ataque puede centrarse en intentar aprovechar una debilidad de un sistema para atentar con la confidencialidad, integridad y disponibilidad de sus datos, siendo estos el resultado de un mal diseño en el desarrollo de software, malas prácticas, fallos en el funcionamiento del software o resultado de propias limitaciones tecnológicas. (Cordovilla, Ordoñez Sigcho, Peñaherrera-Larenas, & Suárez-Matamoros, 2020)

Las aplicaciones web que las empresas utilizan en producción como parte de su plataforma tecnológica suelen ser uno de los blancos más frecuentes para vulnerar, debido a que estos son utilizados desde cualquier dispositivo que tenga acceso a cualquier navegador de internet, mediante los cuales pueden ser interceptados el intercambio de datos e información entre un cliente y un servidor por las peticiones que el sistema realiza para su interacción con los clientes. (Yeison Molina Marin, 2020)

El uso de plataformas tecnológicas aplicadas a ideas de negocio ha ido evolucionando con el tiempo, es por eso que ahora se opta por diseñar plataformas orientadas a la web para que así cualquier dispositivo con acceso a internet y un navegador web puedan hacer uso de ellas, es así como tenemos plataformas orientadas a diferentes ideas de negocio, una de ellas es la plataforma tecnológica de gestión de restaurantes GastroEc la cual se dedica a "Brindar soluciones tecnológicas y de digitalización integral de procesos y atención al cliente especializada en restaurantes" (GastroEc, 2024). La cual será tomada como estudio de caso para esta investigación

Ahora podemos encontrar varias herramientas que nos pueden ayudar a evaluar la seguridad de una plataforma web, como es el caso del framework Owasp ZAP, que es un proyecto open-source siendo esta una herramienta dedicada a realizar pruebas de penetración en aplicaciones web, con la finalidad de mitigar estas amenazas (ZAP, s.f.). Tenemos otra herramienta que nos puede ayudar en la lucha contra la mitigación de vulnerabilidades y a mejorar la calidad de código en los diseños de sistemas web, que es SonarQube, esta herramienta nos ayuda en la evaluación del código, brindándonos soluciones enfocadas en buenas prácticas para la obtención de un código seguro, confiable, limpio, modular y mantenible dándonos como resultado un sistema más eficiente y confiable además de fácil de escalar y mantener (SonarQube, s.f.)

1.2. Interrogantes de la investigación

¿Existen vulnerabilidades en la plataforma Tecnológica de restaurantes GastroEc?

¿Qué herramientas se puede utilizar para realizar evaluaciones de vulnerabilidades a la plataforma tecnológica de restaurantes GastroEc?

¿Qué correcciones de vulnerabilidades se pueden aplicar en la plataforma tecnológica de restaurantes GastroEc?

1.3. Objetivos de la investigación

1.3.1. Objetivo general

Evaluar la seguridad de la plataforma tecnológica de Restaurantes GastroEc mediante el uso de las herramientas SonarQube y OWASP ZAP, para reforzar la calidad del código fuente de la plataforma.

1.3.2. Objetivos específicos

1. Identificar vulnerabilidades de la plataforma mediante el uso de las herramientas SonarQube y OWASP ZAP.
2. Desarrollar una prueba de concepto aplicada a la plataforma tecnológica GastroEc aplicando las recomendaciones y correcciones obtenidas mediante el uso de las herramientas SonarQube y OWASP ZAP
3. Evaluar la efectividad de las medidas implementadas para corregir las vulnerabilidades de seguridad de la plataforma tecnológica.

1.4. Justificación

Esta investigación propone una solución a un problema que existe en la actualidad abarcando diferentes conocimientos, técnicas y herramientas relaciones con la seguridad informática, primero tenemos la evaluación de la seguridad del software que en esta investigación se aplicará a la plataforma tecnológica de restaurantes GastroEc que será tomada como estudio de caso para esta investigación, ya que su plataforma maneja datos que pueden ser considerados sensibles para sus clientes y posibles objetivos para ciberataques, evaluación que será realizada mediante herramientas como Owasp Zap que es la que permitirá realizar una evaluación de seguridad mediante pruebas de penetración hacia la plataforma para saber qué tipo de vulnerabilidades podremos encontrar y posibles soluciones a aplicar

Muchas de las vulnerabilidades que se pueden encontrar dentro de una plataforma tecnológica pueden originarse en sus aplicaciones web, ya que por medio de estas se accede a los datos que las organizaciones manejan, razón por la cual una de las soluciones más viables para actuar y mitigar esas vulnerabilidades es partir del mismo código fuente

de la plataforma web (Candel, 2019), para esto se puede utilizar la herramienta SonarQube, esta herramienta es la que permitirá evaluar el código fuente de la plataforma web, basando sus evaluaciones en patrones de diseño, calidad de código, esta herramienta es la que permitirá mejorar el código fuente dando como resultado una aplicación que contenga código limpio, seguro, mantenible y escalable, mejorando así la calidad del software (SonarQube, s.f.).

Este proyecto de investigación está alineado hacia los objetivos de desarrollo sostenible (ODS). Específicamente este proyecto se encuentra relacionado con el siguiente objetivo:

ODS 8- TRABAJO DECENTE Y CRECIMIENTO ECONÓMICO

Metas del objetivo	Justificación
8.2 Lograr niveles más elevados de productividad económica mediante la diversificación, la modernización tecnológica y la innovación, entre otras cosas centrándose en los sectores con gran valor añadido y un uso intensivo de la mano de obra. (Unidas, 2023)	El proyecto mejorará la calidad de la plataforma web de gestión de restaurantes, lo que resultará en la mejora de productividad en los procesos que se realicen dentro de los restaurantes, mejorando e innovando la calidad de servicio que estos brinden

Tabla 1 Explicación ODS 8

ODS 9- INDUSTRIA, INNOVACIÓN E INFRAESTRUCTURA

Metas del objetivo	Justificación
9.b Apoyar el desarrollo de tecnologías, la investigación y la innovación nacionales en los países en desarrollo, incluso garantizando un entorno normativo propicio a la diversificación industrial y la adición de valor a los productos básicos, entre otras cosas. (Unidas, 2023)	Este proyecto apoya la investigación de tecnologías relacionadas hacia la seguridad informática aplicada hacia un entorno de productividad que es la gestión de Restaurantes

Tabla 2 Explicación ODS 9

Este proyecto contribuye a la línea de investigación de “Desarrollo, aplicación de software y cyber security (seguridad cibernética)” de la Universidad Técnica del Norte.

CAPÍTULO II
MARCO REFERENCIAL

2.1. Antecedentes

En la actualidad estamos en una sociedad digital donde el uso de herramientas o dispositivos tecnológicos se ha vuelto cotidiano en personas u organizaciones para realizar tareas u actividades, que antes se lo realizaba de manera física o manual ahora se han actualizado para llevarlo a cabo de manera más fácil y rápido mediante el uso de nuevas tecnológicas, esto ha influido en la manera que las organizaciones llevan a cabo sus servicios y procesos cambiando el modo en que realizan sus actividades para adaptarse a la era digital mediante el uso de estas herramientas tecnológicas, innovando sus procesos para ofrecer mejores servicios, es por eso que con estos cambios incluso aparecen nuevos riesgos que están ligados al uso de las nuevas tecnologías como son los ciberataques, ciberamenazas, problemas que pueden afectar a plataformas tecnologías y de las cuales hay que estar actualizados para evitar que estos riesgos sean explotados (David Arroyo Guardeño, 2020).

Siendo así, que la seguridad informática es mucho más relevante en estos últimos años, pues es la encargada de afrontar estos nuevos retos con respecto a la seguridad, encontrando cada día nuevos casos de ciberataque a plataformas tecnológicas como aplicaciones webs, aplicaciones móviles , infraestructura de red o conexiones a sistemas; estudiándolos y buscando manera de poder mitigar estos riesgos y vulnerabilidades a las que todos los que hacen uso de plataformas o herramientas digitales están expuestos (Suárez J. L., 2020).

Ecuador no es una excepción ante el uso de nuevas tecnologías y al estar expuestos a estos riesgos cibernéticos, con el uso de internet el cual ahora más que un privilegio es una necesidad, servicio al cual el 43% de ecuatorianos tiene acceso, aun no se tiene mucho conocimiento acerca de lo que conlleva la seguridad informática, dejando de lado medidas de protección contra las amenazas y riesgos que conlleva su uso. Lo que ha resultado en ataques cibernéticos a varias plataformas en el país, siendo estas de uso público o privado, tenemos por ejemplo el ataque con el virus Wannacry en el 2017 que afectó a América Latina siendo Ecuador el tercer país más afectado por este virus o en el 2019 donde la información de 17 millones de ecuatorianos se vio expuesta en la que tuvo que intervenir

la empresa de seguridad VpnMentor encontrando tal falla, siendo esta la más grande filtración de información en América Latina (Alvarado, 2020).

Con estos antecedentes, la seguridad informática en el país toma relevancia que lleva a la creación de planes a favor de la ciberseguridad y seguridad Informática planteando retos y oportunidades que puedan poner en práctica todas las empresas apoyando en el avance de un Ecuador moderno, poniendo en conocimiento información acerca de los riesgos cibernéticos a que cualquier tipo de persona u organización pueda llegar a enfrentarse, como es el caso del documento "Estrategia Nacional de Ciberseguridad en Ecuador" un documento en el que se detalla cómo es la realidad en el país respecto a la seguridad informática y plantea pilares para mantener una seguridad informática competente, reflejando lo que plasma su visión "Ecuador es una sociedad inclusiva y competitiva en el futuro digital con capacidades nacionales para gestionar los riesgos de ciberseguridad" (Maino, 2022)

2.2. Marco teórico

2.2.1 Seguridad informática

La seguridad informática abarca todo lo relacionado con la protección de la información siendo esta personal, organizacional o gubernamental y se enfoca a todo dispositivo de uso cotidiano, laptops, computadores, celulares, entre otros dispositivos, buscando amenazadas que puedan alterar la integridad de la información guardada y manejada por estos dispositivos (Suárez J. L., 2021).

Dentro de cualquier organización es posible que existan sucesos que atenten contra los activos de la empresa, a estos problemas se los conoce como amenazas y a la probabilidad de que estos se materialicen se lo conoce como vulnerabilidad. Estas consecuencias pueden desencadenar en impactos de diferentes niveles dentro de la organización, al tipo de impacto que estas vulnerabilidades puedan causar en la organización se lo conoce como riesgo, lo que busca la seguridad informática es minimizar esto riesgos para que las vulnerabilidades que existan no puedan ser explotadas. (Palacios, 2020)

De acuerdo a Eduardo Samaniego la seguridad informática es toda medida que se lleve a cabo para impedir la ejecución de operaciones no autorizadas sobre sistemas informáticos que puedan comprometer su integridad, autenticidad y confidencialidad. Con el objetivo de minimizar riesgos y garantizar que los recursos del sistema sean utilizados de manera (Mena, 2021)
De acuerdo a José Gamboa la seguridad informática se refiere a la protección de la información de toda índole que puede encontrarse alojado no solo en la red sino también en dispositivos tecnológicos (celulares, computadoras, tablets entre otras), protegerá de toda amenaza que pueda poner en riesgo la información que se encuentra almacenada, Destaca que una buena seguridad no solo se debe basar en la prevención de ataques, sino también en la detección y corrección de los mismos (Suárez J. L., 2021)
De acuerdo a Javier Guaña la seguridad informática es fundamental en la protección de la información, esta debe garantizar la confidencialidad, integridad y disponibilidad, incluso lo enfoca a un tema escolar. Destaca la importancia de manejar buenas prácticas y aplicar normas internacionales para controlar el manejo de la información en el ámbito educacional de acuerdo a la dependencia de las herramientas digitales dentro de un entorno educativo. (Moya, 2023)

Tabla 3 Definición Seguridad Informática

Abarcando estas definiciones podemos decir que la seguridad informática es todo medida que se tome con el objetivo de salvaguardar la información que se encuentra alojada en la red o dispositivos y garantizar el acceso a ella solo para personas autorizadas, las medidas que se tomen deben garantizar que la información no se verá comprometida en su Integridad, Autenticidad y Confidencialidad, las acciones que se lleven a cabo deben minimizar el riesgo de comprometer la información y saber cómo actuar en caso de que ocurra algún tipo de ataque.

Existen tres conceptos principales que abarca la seguridad informática que son: la integridad, la confidencialidad y la disponibilidad. A estos se los ha denominado como la Triada de la seguridad de la información, enfocándose principalmente en la protección de

los datos, estas propiedades son las que ayudan a mantener un sistema robusto, minimizando el riesgo de sufrir eventos que puedan comprometer la seguridad de la información (Tejada, 2023), en la figura 1 se detalla el concepto de la triada de la seguridad de la información.

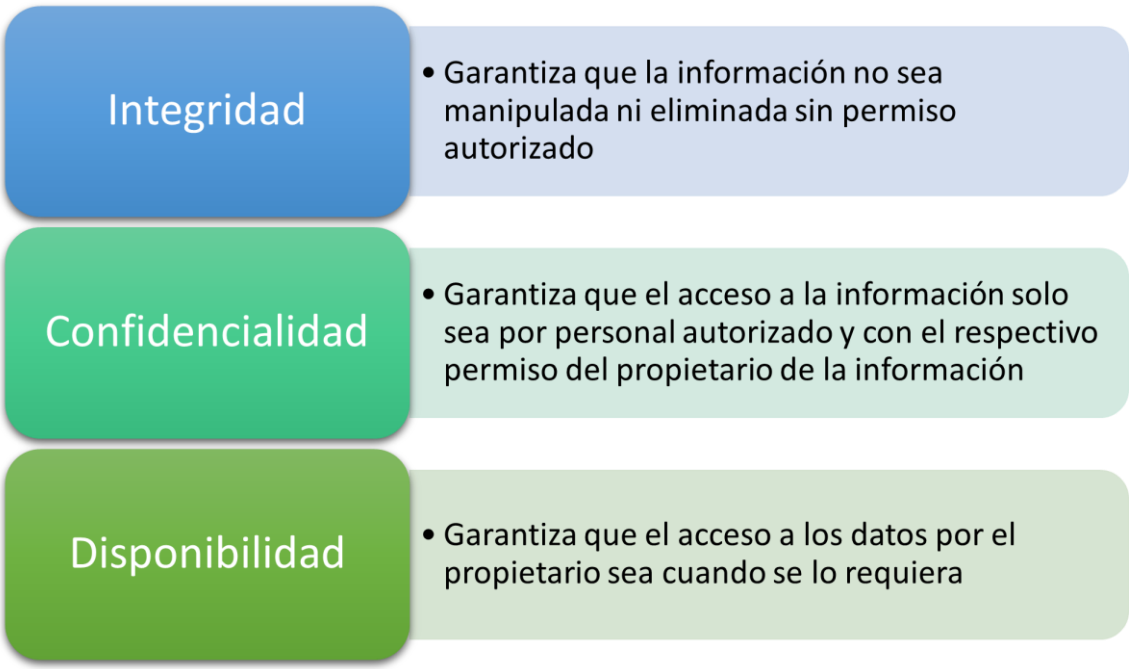

Figura 1 Triada de la seguridad de la información

En la actualidad se ha discutido mucho acerca de si los conceptos vistos abarcan todo lo que demanda tomar en cuenta respecto a la seguridad informática, siendo así que para algunos no resulta ser suficientes, todo esto debido a la evolución que ha tenido la tecnología y la forma en donde se encuentra y maneja la información en la actualidad, siendo así que nacen nuevos conceptos que complementan a la triada de la seguridad de la información los cuales son: El control, La autenticidad y Utilidad (Vega, 2021), en la figura 2 se detalla las definiciones de los nuevos conceptos.

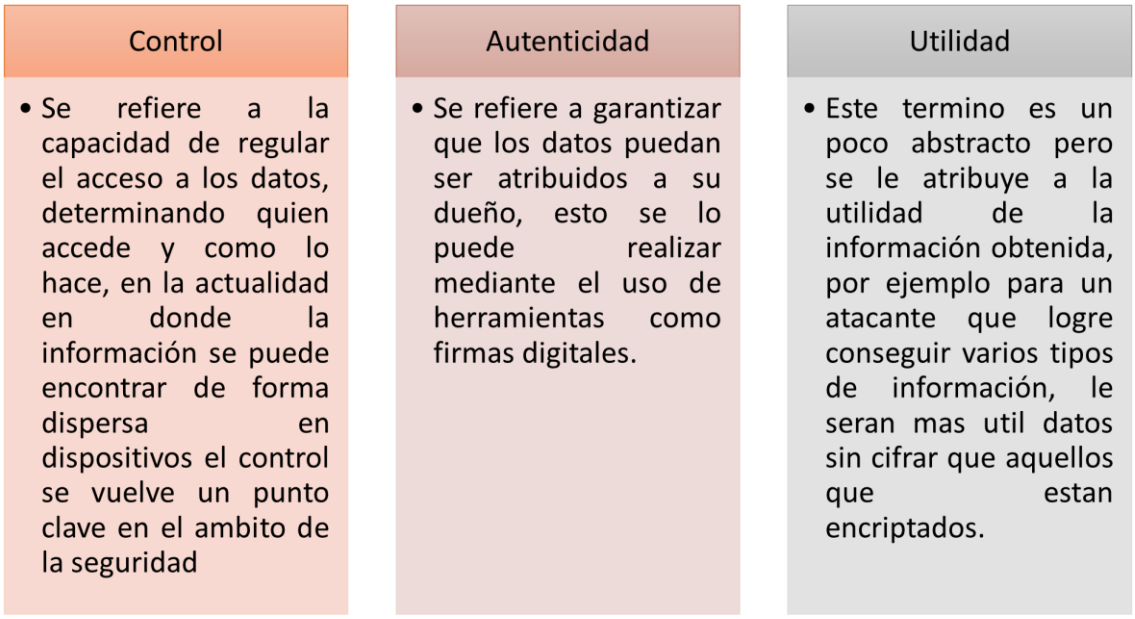

Figura 2 Conceptos complementarios de la seguridad informática

Aunque la triada de la seguridad de la información son considerados como los pilares en la seguridad informática con el tiempo aparecerán nuevos términos que tomaran como base estos conceptos y se adecuaran a la realidad tecnológica del momento para cumplir con el objetivo de proteger y salvaguardar la información de la mejor manera.

2.2.2 Incidentes de seguridad

Un incidente de seguridad se lo puede definir como un evento no esperado o no contemplado que afecta directamente a las características principales de la seguridad informática (integridad, confidencialidad y disponibilidad) comprometiendo el funcionamiento normal de operaciones o servicios y poniendo en peligro la información (Tejada, 2023).

Dentro de un sistema existen diferentes tipos de incidentes de seguridad que podrían llegar a darse debido a una mala gestión de seguridad como, por ejemplo:

- Acceso no autorizado
- Malware
- Denegación de servicio
- Intentos de obtención de información

- Mal uso de recursos

En la actualidad ahora existen formas en las que podemos clasificar estos incidentes de seguridad, de acuerdo con su nivel de peligro o impacto que estos tienen en el sistema se los puede calificar como incidentes de nivel bajos o alto, basándose en parámetros de calificación que pueden ser propios adoptando algún sistema de calificación que permita gestionar incidentes de seguridad (Moreno García, 2022).

2.2.3 Evaluación de la seguridad en sistemas.

Existen varias técnicas para la evaluación de seguridad en sistemas, cada una con diferentes métodos de aplicabilidad, pero con el mismo objetivo buscar vulnerabilidades algunas de estas técnicas sacadas de (Briceño, 2020) serán descritas a continuación.

Técnicas de revisión	Este tipo de técnica busca encontrar vulnerabilidades mediante la evaluación de aplicaciones, sistemas, redes o procedimientos, estas suelen ser realizadas de manera manual buscando en configuraciones de archivos o programas, bitácoras o documentación
Técnicas de identificación	Esta técnica se enfoca en la evaluación hacia los sistemas, buscando fallas en servicios, puertos, la red o el sistema en sí mismo, esta puede ser realizada de manera manual igualmente, aunque ahora ya existen herramientas que ayudan en la exploración de vulnerabilidades en sistemas.
Técnicas de validación de vulnerabilidades	Este tipo de técnica está basado en el descubrimiento de contraseñas, ingeniería social, pruebas de intrusión o de seguridad en aplicaciones, permitiendo conocer y ejecutar las vulnerabilidades encontradas ya sea de manera manual o con el uso de herramientas.

Tabla 4 Técnicas de evaluación de seguridad

Cada una de estas técnicas usadas de manera individual quizás nos puedan proporcionar una visión completa de toda la seguridad de una infraestructura, es por eso que se puede optar por la combinación de varias técnicas para obtener una evaluación de seguridad más completa y confiable.

2.2.4 Herramientas de evaluación de seguridad informática

Al momento de realizar una evaluación de seguridad es importante saber que herramientas se va a utilizar, el proceso que se realiza para la evaluación a la seguridad de una infraestructura es el de atacar al sistema en si para encontrar las vulnerabilidades, por eso existen varias herramientas que funcionan de esa manera teniendo en su lógica los ataques más comunes y actuales que puedan encontrarse en los sistemas. Terminando con un diagnóstico de las vulnerabilidades encontradas y posibles recomendaciones a llevar a cabo (Romero et al, 2018).

En la actualidad existen varios programas que se dedican a realizar búsqueda de vulnerabilidades, estos pueden ser de tipo software libre como de paga, el campo que abarcan este tipo de software es tan amplio que podemos encontrar programas que analicen tanto aplicaciones móviles como sistemas web, enfocándose en peticiones http, funcionalidades en la vista, código estático, toda forma y proceso por la cual el usuario interactúe con la aplicación o sistema (Zafra, 2017). Es así como encontramos programas como los siguientes:

Owasp Zap. Es un proyecto enfocado en la seguridad de aplicaciones web, siendo una comunidad dedicada a desarrollar, comprar y mantener aplicaciones seguras. La complejidad en el desarrollo de software ha aumentado por lo que se ha dificultado el lograr un software seguro, siendo así que los riesgos más comunes son los esenciales a descubrir y resolver de manera rápida. Es así como el proyecto Owasp genera un informe centrado en los 10 principales riesgos de seguridad (Zambrano et al, 2022).

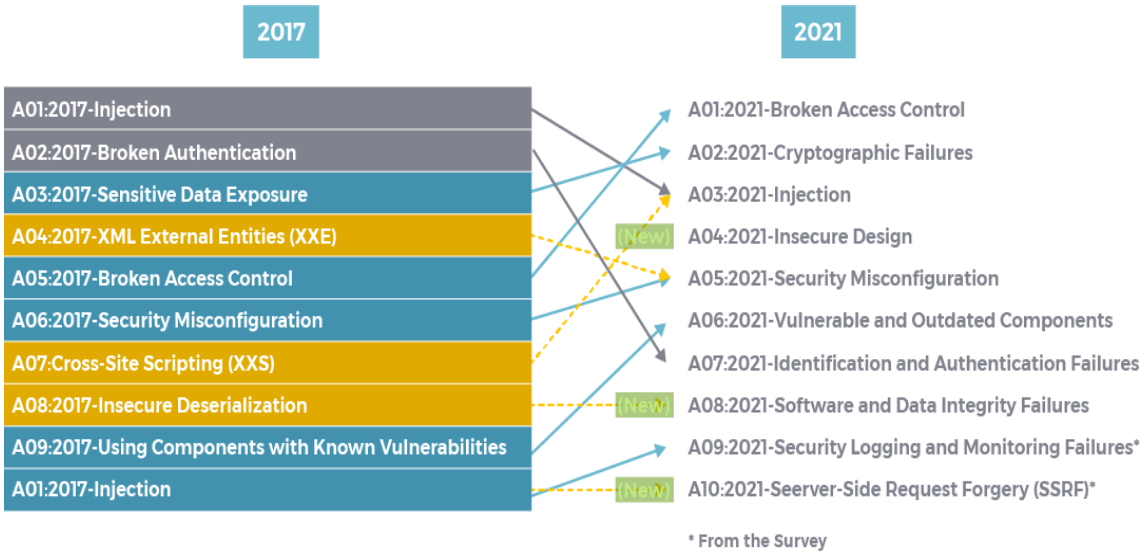

Figura 3 Owasp Top 10 de vulnerabilidades de seguridad

Fuente: (Zambrano & Willians Eduardo Basurto Vidal, 2022)

Owasp Zap es una herramienta desarrollada para realizar pruebas de penetración orientadas a aplicaciones web gratuita, lo que quiere decir que es una aplicación de código abierto razón por la cual puede ser adaptados a las necesidades de la evaluación requerida, siendo así la aplicación de escaneo de vulnerabilidades más descargada y utilizada, su funcionamiento es como un proxy, siendo intermediario entre el navegador y la aplicación web, interceptando e inspeccionando solicitudes y mensajes de la aplicación web. (Cero, 2023). De esta manera clasifica las vulnerabilidades encontradas dándoles una calificación dependiendo de cuan riesgosas sean estas para la plataforma.

Owasp Zap cuenta con una metodología de evaluación de riesgo que puede ser aplicada a las vulnerabilidades encontradas en un sistema, estas se adaptarían a la realidad de la plataforma en la que se aplica con la cual se puede observar que tan riesgoso puede ser que una vulnerabilidad sea explotada (Foundation)

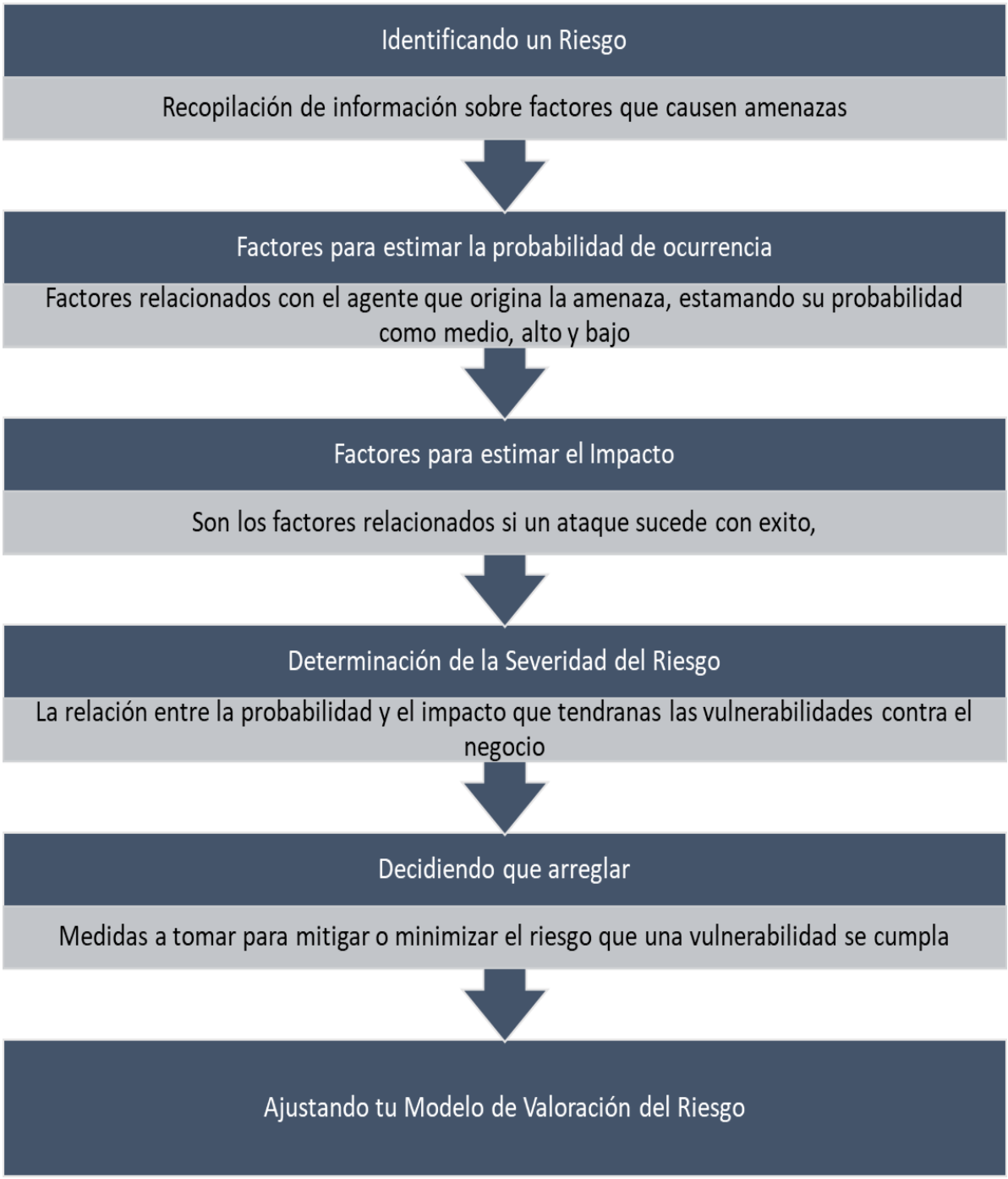

Figura 4 Modelo de evaluación de Riesgo Owasp

El riesgo evaluado resulta de la relación entre en la probabilidad de ocurrencia de una amenaza y el impacto que puedan generar las vulnerabilidades al negocio, estas pueden ser clasificadas en tres niveles (bajo, medio y alto) y su puntuación dependerá del criterio de calificación de la persona a cargo de llevar la metodología en marcha.

Likelihood and Impact Levels	
0 to <3	LOW
3 to <6	MEDIUM
6 to 9	HIGH

Figura 5 Nivel de probabilidad e impacto

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	edium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

Figura 6 Cálculo de severidad del riesgo

A pesar de que Owasp Zap es una herramienta libre, cuenta con una comunidad que la mantiene al tanto de las nuevas vulnerabilidades que puedan aparecer, actualizando su código constantemente y volviéndolo una herramienta eficiente contra la búsqueda de vulnerabilidades en sistemas web, fortaleciendo así la seguridad de plataformas que se encuentren en producción.

SonarQube. Es una herramienta Open Source que nos permite analizar el código estático de un proyecto, revisando la calidad de código y encontrando vulnerabilidades en ella, brinda soluciones basadas en buenas prácticas de programación y soporta varios lenguajes de programación como php, javaScript, C++, C, Python entre otros. Tomando en cuenta parámetros como confiabilidad, seguridad o Smell Code que se refiere a código redundante o código duplicado o malas prácticas de codificación. La importancia en el uso de este software es que mejora la calidad del código fuente al momento de desarrollar un proyecto, mitigando vulnerabilidades que nacen de malas prácticas de codificación,

mejorando la seguridad en las aplicaciones desarrolladas, dando como resultado un proyecto con una codificación más limpia y confiable (Onyenweaku, 2021).

Estructura de SonarQube

Existen cuatro componentes en los que se encuentra dividido SonarQube.

- Servidores de SonarQube
- Base de datos
- Sonar Scanner

Los servidores de SonarQube son los encargados de levantar tanto el servidor web el cual contiene la interfaz de SonarQube, y el servidor de búsqueda basado en ElasticSearch, además tenemos motor informático que se encarga de procesar el resultado de los análisis y guardarlos en la base de datos de SonarQube. La base de datos es la encargada de almacenar todo lo relacionado a los parámetros que aplica SonarQube (métricas, problemas de seguridad y calidad de código) que se generan durante el proceso de escaneo. Sonar scanner es el encargado de realizar uno o varios escaneos a los proyectos requeridos (SonarQube, s.f.).

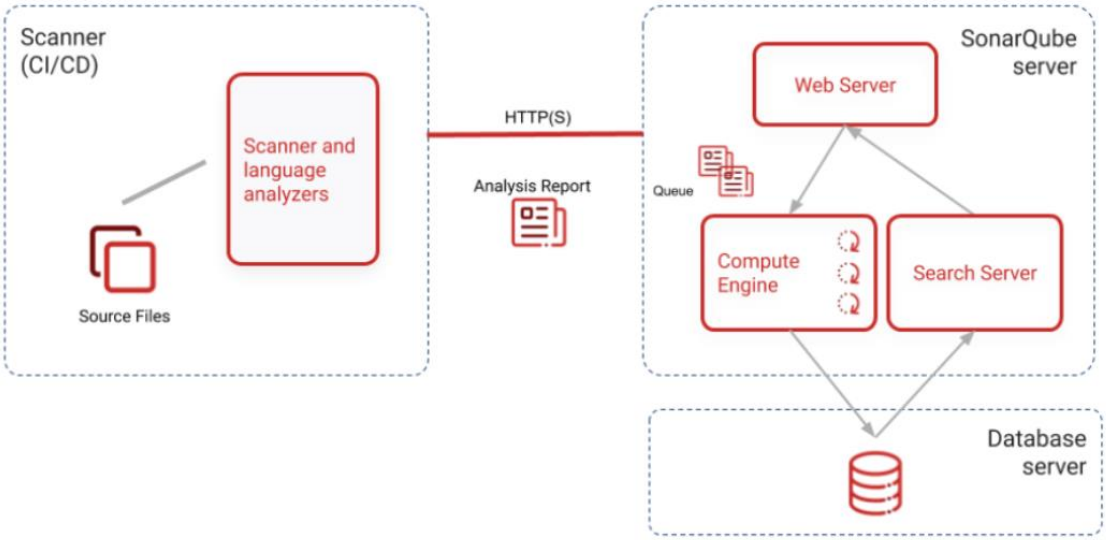

Figura 7 Análisis de vulnerabilidades SonarQube

Fuente: (SonarQube, 2023).

SonarQube es una plataforma de código abierto utilizada para evaluar y analizar la calidad del código fuente. Ofrece una amplia gama de herramientas para identificar problemas, vulnerabilidades, y áreas de mejora en el código de un proyecto. La herramienta es altamente reconocida en entornos de desarrollo de software debido a su capacidad para mejorar la calidad del código y facilitar la entrega de software más confiable y seguro.

En la Figura 8 se muestran las funcionalidades principales en las que está basado SonarQube

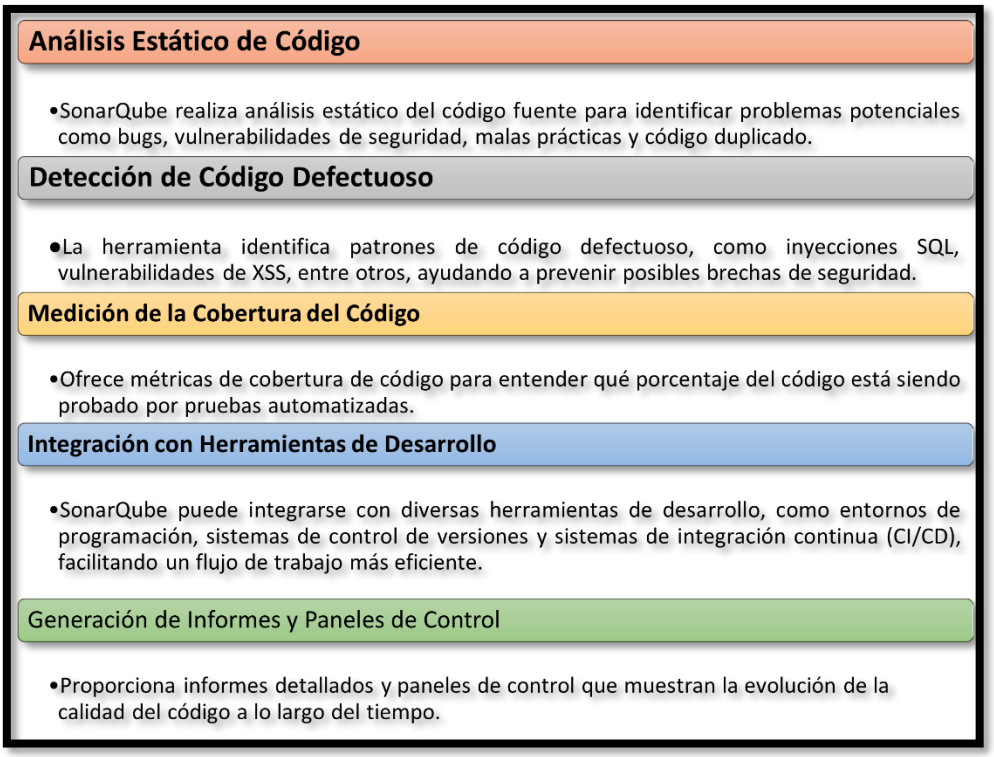

Figura 8 Funciones principales de SonarQube

SonarQube es una herramienta valiosa para el análisis de calidad del código que ofrece beneficios significativos en términos de mejora de la calidad, seguridad y eficiencia en el desarrollo de software. Si bien tiene limitaciones, su capacidad para identificar problemas comunes en el código y proporcionar recomendaciones para mejorarlos lo convierte en una herramienta fundamental para cualquier equipo de desarrollo.

2.3. Marco legal

La evaluación de la seguridad en plataformas tecnológicas busca la protección de la información, las organizaciones manejan una gran cantidad de datos que pueden ser información personal de clientes o empleados y es un riesgo que esta se vea filtrada por terceros para usos no éticos, al aplicar una evaluación de seguridad se mitigaran esas vulnerabilidades para proteger la información especialmente la información personal de clientes apegándose así a los lineamientos con la "Ley Orgánica de Protección de Datos Personales del Ecuador" como lo es el artículo 37 que dice lo siguiente

"Art. 37.-Seguridad de datos personales. -El responsable o encargado del tratamiento de datos personales según sea el caso, deberá sujetarse al principio de seguridad de datos personales, para lo cual deberá tomar en cuenta las categorías y volumen de datos personales, el estado de la técnica, mejores prácticas de seguridad integral y los costos de aplicación de acuerdo con la naturaleza, alcance, contexto y los fines del tratamiento, así como identificar la probabilidad de riesgos.

El responsable o encargado del tratamiento de datos personales, deberá implementar un proceso de verificación, evaluación y valoración continua y permanente de la eficiencia, eficacia y efectividad de las medidas de carácter técnico, organizativo y de cualquier otra índole, implementadas con el objeto de garantizar y mejorar la seguridad del tratamiento de datos personales" (LOPDP, 2021).

Con artículos como el citado dentro las leyes que rigen a un país, podemos darnos cuenta de la relevancia de la tecnología en la actualidad, obligando a los países a crear artículos que incluyen derechos y obligaciones que las personas o empresas deben cumplir, para proteger su integridad dentro del mundo cibernético.

CAPÍTULO III
MARCO METODOLÓGICO

3.1. Descripción del área de estudio / Descripción del grupo de estudio

La investigación se desarrolló en la ciudad de Ibarra en la empresa Ikono.ec en la cual se realizó el análisis de vulnerabilidades sobre la plataforma Tecnológica GastroEc, que es un software de gestión de servicios de restaurantes que está ejecutándose en todo el país. Al tratarse de un proyecto enfocado al análisis de seguridad de la aplicación el equipo de trabajo estuvo conformado por el responsable de la empresa, jefe de desarrollo de la plataforma y el estudiante investigador.

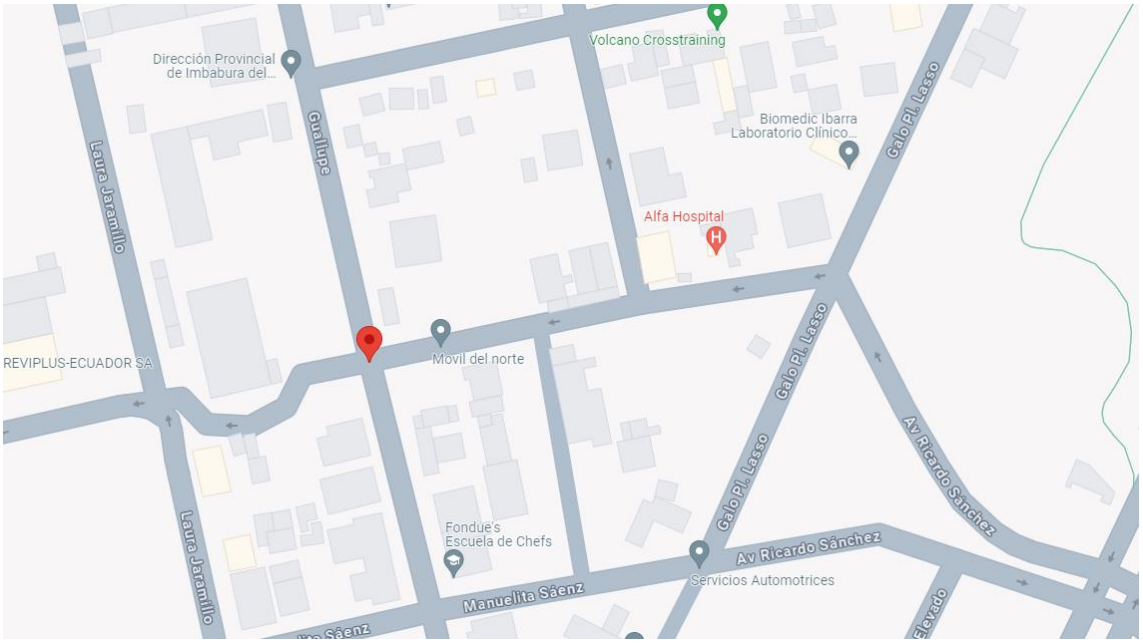

Figura 9 Ubicación referencial de la oficina de desarrollo de GastroEc

3.2. Enfoque y tipo de investigación

El tipo de investigación fue el de ingeniería aplicada con un enfoque cuantitativo, la cual se centró en el análisis y recopilación de datos e información esto con el fin de comprobar teorías, caracterizado por ser considerado un enfoque secuencial, por lo que no se puede saltar una etapa sin terminar otra (Hernández, 2014).

Este tipo de investigación fue la más adecuada pues permitió una evaluación a la seguridad de una infraestructura tecnológica por lo cual se aplicó todo un proceso

secuencial basado en las practicas recomendadas y metodología Owasp, que conlleva un análisis de la actualidad de la infraestructura, para detectar vulnerabilidades y encontrar soluciones que posteriormente serán aplicadas y evaluadas nuevamente para obtener resultados.

3.3. Procedimiento de investigación

El proceso de investigación se definió mediante la especificación del entorno de desarrollo y despliegue de la plataforma GastroEc. El desarrollo de la investigación es el proceso en sí de la prueba de concepto.

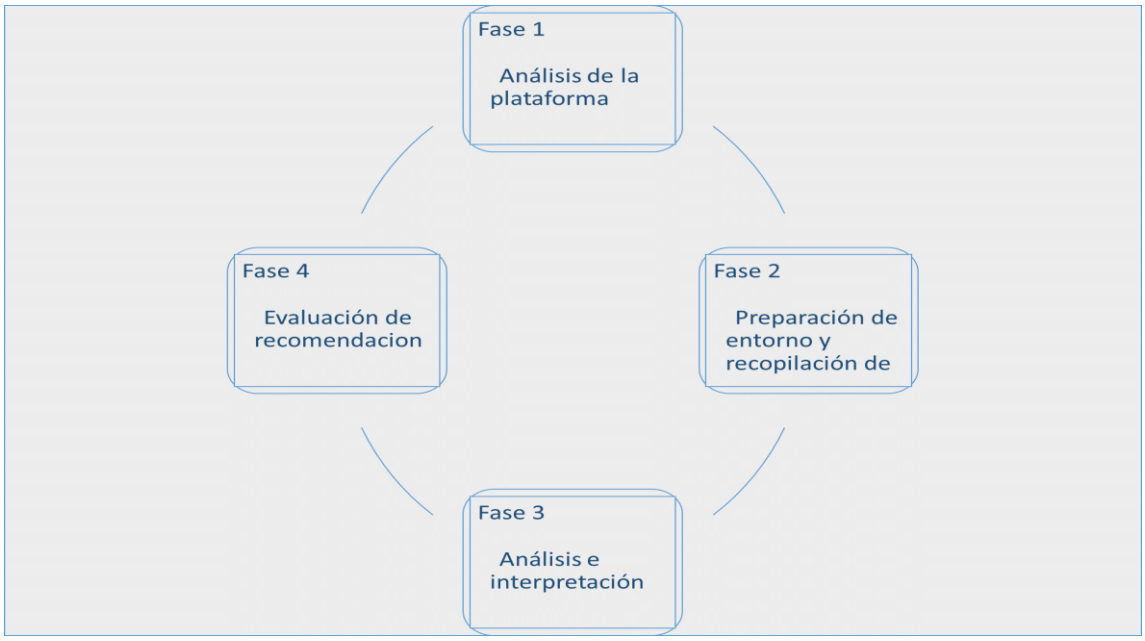

Figura 10 Proceso de investigación

Fase 1: Análisis de la plataforma tecnológica GastroEc. En esta parte de la investigación se recopiló la información, investigando a profundidad las herramientas que se utilizaran que son Owasp Zap y SonarQube y la plataforma tecnológica GastroEc. Con esto obtendremos información de cómo evaluar la plataforma tecnológica, escogiendo las pruebas y métodos que más se adapten a la realidad de la plataforma

Fase 2: Preparación de entorno y recopilación de datos. En esta fase se realizó una evaluación exhaustiva de la seguridad a la plataforma tecnológica para encontrar vulnerabilidades mediante el uso de la herramienta Owasp Zap para realizar pruebas de penetración al sistema y SonarQube para revisar el tipo de vulnerabilidades que puedan existir a nivel de código fuente.

Fase 3: Análisis e interpretación de datos. Una vez realizada la evaluación de la seguridad se procedió a generar un informe con los resultados encontrados y las acciones que se podrían llegar a tomar, en el cual se detalla el nivel de riesgo de tales vulnerabilidades y acciones a seguir, las herramientas que se han utilizado para la evaluación pueden ayudar en la elaboración de este informe, las mismas ofrecen posibles correcciones que pueden ser aplicadas al sistema.

Fase 4: Evaluación de recomendaciones. En esta fase se realizó una evaluación de seguridad aplicada a la plataforma tecnológica para comprobar que tan eficiente son las recomendaciones aplicadas, esto se basará en una comparación de resultados del primer informe de resultado de la evaluación y el nuevo que se generará.

3.4 Evaluación de efectividad

La efectividad de la prueba de concepto aplicada a la plataforma tecnológica GastroEc se dio mediante la evaluación del riesgo real que propone la metodología OWASP realizando este proceso tanto al inicio del proyecto y al final para observar que tan efectivas son las recomendaciones aplicadas a la plataforma.

3.5. Consideraciones bioéticas

La investigación se desarrolló considerando los principios bioéticos de beneficencia, no maleficencia y autonomía. Esta investigación contó con la autorización del gerente de la plataforma tecnológica GastroEc se garantizó confidencialidad y anonimato en los procesos y actividades que intervenga la investigación, además de que se solicitara los permisos necesarios que sean requeridos por la investigación.

Se garantizó que todo lo que conllevo esta investigación no fue en contra de los principios de la plataforma, y que los cambios que esta requiera no afecten con la misión y visión de la plataforma.

CAPÍTULO IV
RESULTADOS Y DISCUSIÓN

En este capítulo se detalla el proceso de evaluación de seguridad aplicada a la plataforma GastroEc, usando las herramientas Owasp Zap y SonarQube, ejecutando el análisis de vulnerabilidades que las herramientas nos provee y discutiremos los resultados y las posibles recomendaciones que se pueden aplicar.

4.1 Ambiente general de la plataforma GastroEc

El entorno de desarrollo de la plataforma GastroEc se detalla a continuación:

Lenguaje de programación	Php 7.4
Framework	CodeIgniter 3.1.13
Base de datos	MariaDB 10.2
Patrón de diseño	MVC (modelo, vista, controlador)
Metodología de desarrollo	XP

Tabla 5 Tecnologías de desarrollo GastroEc

4.2 Análisis de resultados Owasp Zap

En este apartado se detalla el proceso de ejecución del análisis de las vulnerabilidades del sistema, con el objetivo de identificar las debilidades que comprometen la seguridad y estabilidad de la plataforma tecnológica mediante la herramienta Owasp Zap. El entorno de desarrollo de la plataforma GastroEc se detalla a continuación:

Herramienta	**Owasp Zap**
Método	Ataques de penetración al sistema
Versión	2.13.0

Tabla 6 Descripción herramienta Owasp Zap.

Se llevaron a cabo pruebas exhaustivas utilizando OWASP ZAP, y se identificaron múltiples áreas de preocupación en la seguridad de la aplicación. A continuación, se presentan los hallazgos agrupados por categorías.

4.2.1. Recuentos de alertas por riesgo y confianza

Del análisis de vulnerabilidades con la herramienta OWASP ZAP se identificó un número de alertas para cada nivel de riesgo y confianza incluido en el informe. Los porcentajes entre paréntesis representan el recuento como un porcentaje del número total de alertas incluidas en el informe, redondeado a un decimal.

		Confidence				
		Confirmado por Usuario	Alta	Media	Baja	Total
Risk	Alto	0 (0,0 %)	0 (0,0 %)	2 (7,7 %)	0 (0,0 %)	2 (7,7 %)
	Medio	0 (0,0 %)	2 (7,7 %)	4 (15,4 %)	1 (3,8 %)	7 (26,9 %)
	Bajo	0 (0,0 %)	2 (7,7 %)	5 (19,2 %)	1 (3,8 %)	8 (30,8 %)
	Informativo	0 (0,0 %)	1 (3,8 %)	6 (23,1 %)	2 (7,7 %)	9 (34,6 %)
	Total	0 (0,0 %)	5 (19,2 %)	17 (65,4 %)	4 (15,4 %)	26 (100%)

Figura 11 Análisis de vulnerabilidades OWASP

4.2.2. Tipo de alertas de seguridad detectadas

La tabla 6 nos muestra el detalle del tipo de alerta específico que se encontró durante la evaluación del sistema, junto con el nivel de riesgo del tipo de alerta y la cantidad que se encontró del mismo.

Tipo de alerta	Riesgo	Cantidad
Inyección SQL	Alto	2
Inyección SQL – MySQL	Alto	1
Ausencia de fichas (tokens) Anti-CSRF	Medio	4
Cabecera Content Security Policy (CSP) no configurada	Medio	12
Desconfiguración de Dominio cruzado	Medio	5
Falta de cabecera Anti-Clickjacking	Medio	6
Filtrado de información en .htaccess	Medio	1
Hidden File Found (Archivo Oculto Encontrado)	Medio	1
Librería JS Vulnerable	Medio	3
Cookie No HttpOnly Flag	Bajo	1
Cookie Without Secure Flag	Bajo	1
Cookie sin el atributo SameSite	Bajo	1
Cross-Domain JavaScript Source File Inclusion	Bajo	9
Divulgación de la marca de hora – Unix	Bajo	33
Server Leaks Version Information via "Server" HTTP Response Header Field	Bajo	53
Strict-Transport-Security Header Not Set	Bajo	75
X-Content-Type-Options Header Missing	Bajo	51
Authentication Request Identified	Informativo	2
Content-Type Header Missing	Informativo	2
Divulgación de información - Comentarios sospechosos	Informativo	60
GET para POST	Informativo	1
Modern Web Application	Informativo	2
Re-examine Cache-control Directives	Informativo	10
Retrieved from Cache	Informativo	8
Session Management Response Identified	Informativo	18
User Agent Fuzzer	Informativo	24
Total	26	

Tabla 7 Tipos de alertas de seguridad identificadas

Del análisis de vulnerabilidad se observaron otros problemas de seguridad que, aunque no se clasifican como críticos o importantes, requieren atención para fortalecer la seguridad general de la aplicación.

4.2.3. Evaluación del riesgo

En la tabla 8 podemos revisar el riesgo-evaluado para cada una de las vulnerabilidades encontradas por Owasp Zap estas se encuentran agrupadas por el tipo de riesgo que representa la vulnerabilidad. Las tablas que indican la calificación de impacto y probabilidad se encuentran en anexos

Tipo de alerta	Factor de impacto	Calif	Probabilidad	Calif	Riesgo
Inyección SQL	6.5	**Alto**	2.63	Bajo	Medio
Inyección SQL – MySQL					
Ausencia de fichas (tokens) Anti-CSRF	3.36	**Medio**	1.38	Bajo	Bajo
Cabecera Content Security Policy (CSP) no configurada					
Desconfiguración de Dominio cruzado					
Falta de cabecera Anti-Clickjacking					
Filtrado de información en .htaccess					
Hidden File Found (Archivo Oculto Encontrado)					
Librería JS Vulnerable					
Cookie No HttpOnly Flag	2.75	Bajo	1.38	Bajo	Nota
Cookie Without Secure Flag					
Cookie sin el atributo SameSite					
Cross-Domain JavaScript Source File Inclusion					
Divulgación de la marca de hora – Unix					
Server Leaks Version Information via "Server" HTTP Response Header Field					
Strict-Transport-Security Header Not Set					
X-Content-Type-Options Header Missing					
Authentication Request Identified	2.75	Bajo	0.88	Bajo	Nota
Content-Type Header Missing					
Divulgación de información - Comentarios sospechosos					
GET para POST					
Modern Web Application					
Re-examine Cache-control Directives					
Retrieved from Cache					
Session Management Response Identified					
User Agent Fuzzer					

Tabla 8 Calculo del riesgo por vulnerabilidad

4.2.4. Prototipo de mejora para puntos de acceso de seguridad

Una vez visto los tipos de vulnerabilidades encontrados en el sistema se procede a contrarrestar o mitigar estas vulnerabilidades, tomando como prioridad aquellas que vulneren la seguridad del sistema catalogadas como riesgo alto.

Se recomienda hacer las siguientes acciones de manera inmediata para disminuir el riesgo en los niveles: Alto y Medio.

<table>
<tr><td>Problemas</td><td>Inyección SQL

Inyección SQL – MySQL</td></tr>
<tr><td>Solución propuesta</td><td>Actualmente los modelos programados dentro del software en gran medida permiten el “Inyección SQL”, es necesario que se realicen los siguientes ajustes donde sea necesario para que el framework haga un escapado de los parámetros de consulta.

Cuando se usan Active Record:

Original

<pre>
1111 $this->db->select('*');
1112 $this->db->from('tbl_empleado');
1113 $this->db->where("identificacion_empleado='$identificacion_empleado' and id_negocio='$id_negocio'
1114 $this->db->limit(1);
1115 $query = $this->db->get();
1116 $temporal = $query->result_array();
1117 return $temporal[0];
</pre>
Modificada

<pre>
1114 public function EmpleadoByIdentificacion($identificacion_empleado, $id_negoci
1115 {
1116 $this->db->select('*');
1117 $this->db->from('tbl_empleado');
1118 $this->db->where("identificacion_empleado",$identificacion_empleado);
1119 $this->db->where("id_negocio",$id_negocio);
1120 $this->db->limit(1);
1121 $query = $this->db->get();
1122 $temporal = $query->result_array();
1123   return $temporal[0];
1124 }
</pre>
Cuando no se usan Active Record:

Original

<pre>
213 public function GetFacturaById($id_factura){
214      $query = $this->db->query("SELECT * FROM $this->table_factura WHERE id_factura = '$id_factura'
215      $temporal = $query->result_array();
216   return $temporal[0];
217 }
</pre>
Modificada</td></tr>
</table>

	208 public function GetFacturaById($id_factura){ 209 $query = $this->db->query("SELECT * FROM $this->table_factura WHERE id_factura = ?",[$id_factura]); 210 $temporal = $query->result_array(); 211 return $temporal[0]; 212 }
Beneficios	Al usar las primitivas de QueryBuilder de Codeigniter al agregar como parámetros y no adjuntos a una cadena se permite una sanitización de la consulta lo que permite que los datos de consulta sean seguros.
Recomendaciones	Se debe cambiar en todos los modelos las consultas basadas en cadenas concatenadas a consultas con parámetros.

Tabla 9 Solución - Inyección SQL

Problema	**Ausencia de fichas (tokens) Anti-CSRF**
Solución propuesta	Los formularios no tienen un método para evitar ataques de tipo CSRF. La solución es incrementar el token CSRF en los formularios dentro de la aplicación en las vistas y validación del token dentro de los controladores. Vista Formulario: Original 26 <!-- Login Form --> 27 <form action="<?= site_url('administrador/validausuario') ?>" method="post" id="form-login" class="form-horizontal form-bordered form- 28 29 30 31 32 <input type="email" id="login-email" name="login-email" class="form-control input-lg" placeholder="Email"> 33 34 35 36 37 38 39 40 <input type="password" id="login-password" name="login-password" class="form-control input-lg" placeholder="Password"> 41 42 43 Vista Formulario: Modificada

	<!-- Login Form -->
	<form autocomplete="off" action="<?= site_url('administrador/validausuario') ?>" method="post">
	<?php
	$_SESSION['admtoken'] = md5(uniqid(mt_rand(), true));
	?>
	<input type="hidden" name="token" id='csrf_token' value="<?php echo $_SESSION['admtoken'];
	
	<input type="email" id="login-email" name="login-email" class="form-control i
	
	<input type="password" id="login-password" name="login-password" class="form-
	
	Validación en el Controlador
	
	$token = filter_input(INPUT_POST, 'token', FILTER_SANITIZE_STRING);
	
	if (!$token || $token !== $_SESSION['admtoken']) {
	// return 405 http status code
	//header($_SERVER['SERVER_PROTOCOL'] . ' 405 Method Not Allowed'
	header('Content-Type: application/json;charset=utf-8');
	echo json_encode([
	'status' => 'ERROR',
	'message' => 'Acceso Denegado'
	]);
	exit;
	} else {

Beneficios	Principalmente en la Seguridad de Sesiones de Usuario ya que se garantiza que las solicitudes provengan de usuarios legítimos y no de fuentes maliciosas evitando que los atacantes engañen a usuarios autenticados para realizar acciones no autorizadas en aplicaciones web.

Tabla 10 Solución - Ausencia de fichas Anti-CSRF

Problemas	**Cabecera Content Security Policy (CSP) no configurada** **Desconfiguración de Dominio cruzado**
Solución propuesta	Para solucionar este problema se recomienda agregar en la cabecera de la plantilla la Política de Seguridad de Contenido; para todas las fuentes externas ya sean scripts o estilos. Al inicio del archivo: application/views/templates/directorio/inc/template_end.php incrementar las siguientes líneas. Antes

	<?php
	/**
	* template_start.php
	*
	* Author: pixelcave
	*
	* The first block of code used in every page of the template
	*
	*/
	?>
	<!DOCTYPE html>
	<!--[if IE 9]> <html class="no-js lt-ie10" lang="en"> <![endif]-->
	<!--[if gt IE 9]><!--> <html class="no-js" lang="en"> <!--<![endif]-->
	<head>
	<meta charset="utf-8">
	
	Despues
	<?php
	
	/**
	* template_start.php
	*
	* Author: pixelcave
	*
	* The first block of code used in every page of the template
	*
	*/
	$CSP = "";
	$CSP .= "default-src 'self' https://firebase.google.com https://maps.googleapis.
	$CSP .= "script-src 'self' https://cdn.jsdelivr.net https://fonts.googleapis.com
	$CSP .= "style-src 'self' cdn.jsdelivr.net https://cdn.jsdelivr.net https://font
	$CSP .= "font-src 'self' https://fonts.gstatic.com https://cdn.jsdelivr.net data
	$CSP .= "connect-src 'self' wss://ws-us2.pusher.com https://maps.googleapis.com;
	$CSP .= "worker-src 'self' blob:;";
	$CSP .= "frame-ancestors 'self';";
	$CSP .= "form-action 'self';";
	
	header("Content-Security-Policy: $CSP");
	header("Cache-Control: no-cache, no-store, must-revalidate"); // HTTP 1.1.
	header("Pragma: no-cache"); // HTTP 1.0.
	header("Expires: 0"); // Proxies.
	header('X-Frame-Options: SAMEORIGIN');

Beneficios	Ayuda a prevenir ataques XSS al permitir a los desarrolladores especificar qué fuentes de contenido (scripts, estilos, imágenes, etc.) son consideradas seguras. Esto reduce la probabilidad de que scripts maliciosos se ejecuten en el navegador del usuario. Permite especificar qué dominios tienen permiso para cargar recursos en la página. Esto ayuda a prevenir la carga de contenido no autorizado y protege contra ataques como la inyección de contenido no deseado.
Recomendación	Para que el prototipo funcione se incluyó la cláusula 'unsafe-inline' debido a que dentro del software existe una gran cantidad de script's incrustados. Se recomienda en lo posible agregar los scripts en archivos independientes.

Tabla 11 Solución - Cabecera Content Security Policy

Problemas	**Falta de cabecera Anti-Clickjacking**

Solución propuesta	Para solucionar o prevenir el clickjacking y mejorar la seguridad de tu sitio web se debe implementar la siguiente medida.
	
	Al inicio del archivo: application/views/templates/backend/inc/template_end.php incrementar las siguientes líneas.
	
	Antes
	
	`<?php`
	`/**`
	`* template_start.php`
	`*`
	`* Author: pixelcave`
	`*`
	`* The first block of code used in every page of the template`
	`*`
	`*/`
	`?>`
	`<!DOCTYPE html>`
	`<!--[if IE 9]>  <html class="no-js lt-ie10" lang="en"> <![endif]-->`
	`<!--[if gt IE 9]><!--> <html class="no-js" lang="en"> <!--<![endif]-->`
	`<head>`
	`<meta charset="utf-8">`
	
	Despues
	
	`<?php`
	
	`/**`
	`* template_start.php`
	`*`
	`* Author: pixelcave`
	`*`
	`* The first block of code used in every page of the template`
	`*`
	`*/`
	`$CSP = "";`
	`$CSP .= "default-src 'self' https://firebase.google.com https://maps.googleapis.`
	`va $CSP .= "script-src 'self' https://cdn.jsdelivr.net https://fonts.googleapis.com`
	`$CSP .= "style-src 'self' cdn.jsdelivr.net https://cdn.jsdelivr.net https://font`
	`$CSP .= "font-src 'self' https://fonts.gstatic.com https://cdn.jsdelivr.net data`
	`$CSP .= "connect-src 'self' wss://ws-us2.pusher.com https://maps.googleapis.com;`
	`$CSP .= "worker-src 'self' blob:;";`
	`$CSP .= "frame-ancestors 'self';";`
	`$CSP .= "form-action 'self';";`
	
	`header("Content-Security-Policy: $CSP");`
	`header("Cache-Control: no-cache, no-store, must-revalidate"); // HTTP 1.1.`
	`header("Pragma: no-cache"); // HTTP 1.0.`
	`header("Expires: 0"); // Proxies.`
	`header('X-Frame-Options: SAMEORIGIN');`

Beneficios	Principalmente se evita que la aplicación sea abierta desde un frame o iframe y por ende evitar que los atacantes puedan superponer contenido malicioso sobre la página cargada en el marco, engañando a los usuarios para que realicen acciones no deseadas, robo de contenido, fuga de información confidencial.

Tabla 12 Solución - Falta de cabecera Anti-Clickjacking

4.2.5. Análisis de resultados del prototipo de Owasp Zap

Una vez aplicada las correcciones a ciertos módulos del sistema se realiza una nueva evaluación con la que se obtienen los siguientes resultados

En esta tabla se muestra el número de alertas de cada tipo de alerta, junto con el nivel de riesgo del tipo de alerta.

		Confidence				
		Confirmado por Usuario	Alta	Media	Baja	Total
Risk	Alto	0 (0,0 %)	0 (0,0 %)	0 (0,0 %)	0 (0,0 %)	0 (0,0 %)
	Medio	0 (0,0 %)	2 (14,3 %)	1 (7,1 %)	0 (0,0 %)	3 (21,4 %)
	Bajo	0 (0,0 %)	2 (14,3 %)	4 (28,6 %)	0 (0,0 %)	6 (42,9 %)
	Informativo	0 (0,0 %)	0 (0,0 %)	4 (28,6 %)	1 (7,1 %)	5 (35,7 %)
	Total	0 (0,0 %)	4 (28,6 %)	9 (64,3 %)	1 (7,1 %)	14 (100%)

Tabla 13 Análisis de vulnerabilidades OWASZap

La tabla 14 nos muestra el detalle del tipo de alerta específico que se encontró durante la evaluación del sistema, junto con el nivel de riesgo del tipo de alerta y la cantidad que se encontró del mismo.

Tipo de alerta	Riesgo	Cantidad
CSP: script-src unsafe-inline	Medio	1
CSP: style-src unsafe-inline	Medio	1
Librería JS Vulnerable	Medio	3
Cookie No HttpOnly Flag	Bajo	1
Cookie Without Secure Flag	Bajo	1
Cookie sin el atributo SameSite	Bajo	1
Server Leaks Version Information via "Server" HTTP Response Header Field	Bajo	40

Strict-Transport-Security Header Not Set	Bajo	40
X-Content-Type-Options Header Missing	Bajo	38
Content-Type Header Missing	Informativo	1
Divulgación de información - Comentarios sospechosos	Informativo	41
Modern Web Application	Informativo	1
Petición de Autenticación Identificada	Informativo	1
Respuesta de Gestión de Sesión Identificada	Informativo	15
Total	14	

Tabla 14 Nuevos tipos de alertas de seguridad identificadas

4.2.6. Evaluación del Riesgo

En la tabla número 15 podemos revisar el riesgo-evaluado para cada una de las vulnerabilidades encontradas por Owasp Zap, estas se encuentran agrupadas por el tipo de riesgo que representa la vulnerabilidad. Las tablas que indican la calificación de impacto y probabilidad final se encuentran en anexos.

Tipo de alerta	Factor de impacto	Calif	Probabilidad	Calif	Riesgo
Inyección SQL	0.00	Bajo	0.00	Bajo	Nota
Inyección SQL – MySQL					
Ausencia de fichas (tokens) Anti-CSRF	0.39	Bajo	0.20	Bajo	Nota
Cabecera Content Security Policy (CSP) no configurada					
Desconfiguración de Dominio cruzado					
Falta de cabecera Anti-Clickjacking					
Filtrado de información en .htaccess					
Hidden File Found (Archivo Oculto Encontrado)					
Librería JS Vulnerable					
Cookie No HttpOnly Flag	2.06	Bajo	1.03	Bajo	Nota
Cookie Without Secure Flag					
Cookie sin el atributo SameSite					
Cross-Domain JavaScript Source File Inclusion					
Divulgación de la marca de hora – Unix					
Server Leaks Version Information via "Server" HTTP Response Header Field					
Strict-Transport-Security Header Not Set					
X-Content-Type-Options Header Missing					

Authentication Request Identified	1.22	Bajo	0.39	Bajo	Nota
Content-Type Header Missing					
Divulgación de información - Comentarios sospechosos					
GET para POST					
Modern Web Application					
Re-examine Cache-control Directives					
Retrieved from Cache					
Session Management Response Identified					
User Agent Fuzzer					

Tabla 15 Evaluación del riesgo, aplicada recomendaciones

4.2.7. Conclusiones análisis con OWASP ZAP

El análisis con OWASP ZAP reveló vulnerabilidades críticas e importantes que deben abordarse de inmediato. La implementación de las recomendaciones proporcionadas fortalecerá la seguridad de la aplicación y reducirá significativamente el riesgo de exposición a amenazas. Para evitar en lo posible el manejo de "script" y "style" incrustados en las páginas con ello poder deshabilitar "style-src includes unsafe-inline" desde la directiva "Content-Security-Policy". Se recomienda actualizar las librerías de Bootstrap 3.3.6 a una superior y actualizar jQuery y plugins relacionados con esas librerías a versiones más actuales.

4.3 Análisis de resultados SonarQube

El análisis de vulnerabilidades realizado con la herramienta SonarQube hacia el código fuente de la plataforma GastroEc, proporciono una visión detallada de la calidad y seguridad del código el cual permitió identificar las áreas clave en las que el software tiene debilidades.

Herramienta	SonarQube
Método	Análisis de código estático
Versión	10.2.1.78527

Tabla 16 Descripción herramienta SonarQube.

El análisis se enfoca a las siguientes carpetas Su ámbito de análisis abarca varios aspectos críticos para el desarrollo de software, proporcionando una visión integral de la salud y la seguridad del código. A continuación, se detallan los principales ámbitos de análisis de SonarQube

- Application/controllers
- Application/models
- Application/views
- Application/helpers

Tipo de Análisis:

- SonarPHP
- SonarHtml

4.3.1. Tipos de vulnerabilidades detectadas con SonarQube

Del análisis de vulnerabilidades detectadas en el análisis de vulnerabilidades de seguridad con la herramienta SonarQube, se detallan a continuación:

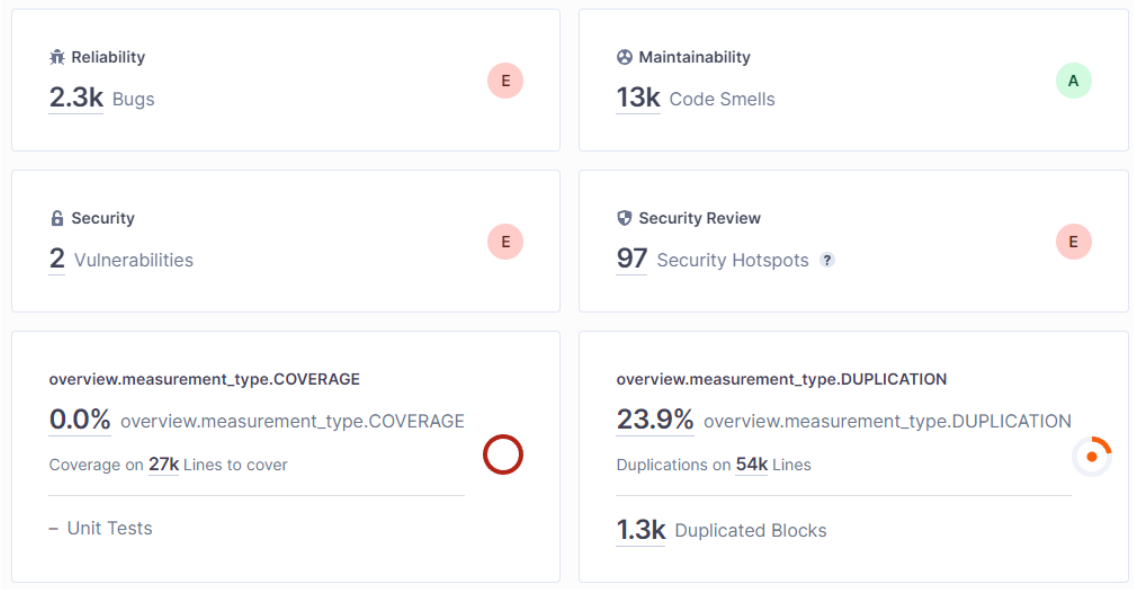

Figura 12 Vulnerabilidades de seguridad de SonarQube

Del análisis de vulnerabilidades de seguridad la figura 12, se muestra el resumen del tipo de tipo de fallo de seguridad detectado.

Tamaño del Proyecto	
Líneas de Código	54.152
Archivos	540
Funciones	1.516
Declaraciones	27.874
Comentarios	
Líneas con comentarios	3.028
Densidad de líneas comentadas	5,3%
Duplicaciones	
Bloques de código duplicados	1.261
Líneas de código duplicadas	24.937
Densidad de líneas duplicadas	23,9%
Archivos con código duplicado	203
Problemas por gravedad	
TOTAL, PROBLEMAS	15.786
Problemas con los bloqueadores	32
Cuestiones críticas	997
Principales problemas	7.652
Problemas menores	7.093
Problemas de información	12

Tabla 17 Detalle de escaneo de SonarQube

A continuación en la tabla 18 se observan los problemas encontrados por SonarQube donde se detalla la severidad, a qué tipo de regla pertenece, el tipo de problema, el lenguaje de programación y el número de problemas encontrados.

Desglose de problemas

Severidad	Reglas	Tipo	Lenguaje	Problemas
BLOCKER	Las propiedades CSS deben ser válidas	BUG	CSS	5
BLOCKER	Las credenciales no deben estar codificadas de forma rígida	VULNERABILIDAD	PHP	2
CRÍTICO	Los literales de cadena no deben duplicarse	CODE_SMELL	PHP	367
CRÍTICO	No se deben usar paréntesis innecesarios para las construcciones	CODE_SMELL	PHP	148
CRÍTICO	Las estructuras de control deben usar llaves	CODE_SMELL	PHP	139
CRÍTICO	La complejidad cognitiva de las funciones no debe ser demasiado alta	CODE_SMELL	PHP	133
CRÍTICO	Las sentencias "switch" deben tener cláusulas "default"	CODE_SMELL	PHP	71
CRÍTICO	Las referencias utilizadas en los bucles "foreach" deben ser "unset"	BUG	PHP	25
CRÍTICO	Una sola línea ejecutada condicionalmente debe denotarse con sangría	CODE_SMELL	PHP	5
CRÍTICO	Los métodos no deben estar vacíos	CODE_SMELL	PHP	4
CRÍTICO	Los condicionales deben comenzar en nuevas líneas	CODE_SMELL	PHP	1
IMPORTANTE	Las filas no deben ser demasiado largas	CODE_SMELL	PHP	3.673
IMPORTANTE	Las etiquetas "<th>" deben tener atributos "id" o "scope"	BUG	HTML	1.295
IMPORTANTE	Los atributos obsoletos en HTML5 no deben usarse	CODE_SMELL	HTML	623
IMPORTANTE	Las secciones de código no deben ser comentadas	CODE_SMELL	PHP	601
IMPORTANTE	Los métodos "privados" no utilizados deben eliminarse	CODE_SMELL	PHP	429
IMPORTANTE	Se debe preferir el uso de espacios de nombres a las funciones "include" o "require"	CODE_SMELL	PHP	250

IMPORTANTE	Las tablas deben tener encabezados	BUG	HTML	156
IMPORTANTE	El elemento "<html>" debe tener un atributo de idioma	BUG	HTML	112
IMPORTANTE	Las secciones de código no deben ser comentadas	CODE_SMELL	HTML	84
IMPORTANTE	"<! Las declaraciones DOCTYPE>" deben aparecer antes de las etiquetas "<html>"	BUG	HTML	49
IMPORTANTE	Los bloques de código anidados no deben dejarse vacíos	CODE_SMELL	PHP	41
IMPORTANTE	HTML "<table>" no debe utilizarse con fines de diseño	CODE_SMELL	HTML	37
IMPORTANTE	Las asignaciones no utilizadas deben eliminarse	CODE_SMELL	PHP	34
IMPORTANTE	Los parámetros de función no utilizados deben eliminarse	CODE_SMELL	PHP	21
IMPORTANTE	Dos bifurcaciones en una estructura condicional no deben tener exactamente la misma implementación	CODE_SMELL	PHP	19
IMPORTANTE	Las variables deben inicializarse antes de su uso	BUG	PHP	18
IMPORTANTE	Las instrucciones "if" contraíbles deben combinarse	CODE_SMELL	PHP	18
IMPORTANTE	Las propiedades no deben duplicarse	BUG	CSS	14
IMPORTANTE	Los pares redundantes de paréntesis deben eliminarse	CODE_SMELL	PHP	14
IMPORTANTE	Los selectores no deben duplicarse	CODE_SMELL	CSS	14
IMPORTANTE	Las funciones no deben tener demasiadas líneas de código	CODE_SMELL	PHP	13
IMPORTANTE	Los campos "privados" no utilizados deben eliminarse	CODE_SMELL	PHP	11
IMPORTANTE	Los bloques de varias líneas deben estar encerrados entre llaves	CODE_SMELL	PHP	11
IMPORTANTE	Los métodos no deben tener implementaciones idénticas	CODE_SMELL	PHP	10
IMPORTANTE	Las clases no deben tener demasiados métodos	CODE_SMELL	PHP	10
IMPORTANTE	Las funciones no deben contener demasiadas instrucciones return	CODE_SMELL	PHP	6

IMPORTANTE	No se debe usar la salida de funciones que no devuelven nada	BUG	PHP	4
IMPORTANTE	Las declaraciones de fuentes deben contener al menos una familia de fuentes genéricas	BUG	CSS	3
IMPORTANTE	Las variables no deben ser autoasignadas	BUG	PHP	3
IMPORTANTE	Los bloques inútiles "if(true) {...}" y "if(false){...}" deben eliminarse	BUG	PHP	3
IMPORTANTE	Los operadores ternarios no deben estar anidados	CODE_SMELL	PHP	2
IMPORTANTE	Las funciones deben usar "return" de forma coherente	CODE_SMELL	PHP	2
IMPORTANTE	Los valores de matriz no deben reemplazarse incondicionalmente	BUG	PHP	2
IMPORTANTE	Las comparaciones de recuento de matrices o objetos contables deberían tener sentido	BUG	PHP	2
IMPORTANTE	Todas las ramas de una estructura condicional no deben tener exactamente la misma implementación	BUG	PHP	1
IMPORTANTE	Los bloques vacíos deben eliminarse	CODE_SMELL	CSS	1
IMPORTANTE	Todo el código debe ser accesible	BUG	PHP	1
MENOR	Las líneas no deben terminar con espacios en blanco al final	CODE_SMELL	PHP	3.457
MENOR	Las palabras clave y constantes de PHP "verdadero", "falso", "nulo" deben estar en minúsculas	CODE_SMELL	PHP	509
MENOR	Las etiquetas "<table>" deben tener una descripción	BUG	HTML	392
MENOR	Los archivos deben contener una nueva línea vacía al final	CODE_SMELL	PHP	382
MENOR	Los nombres de función deben cumplir con una convención de nomenclatura	CODE_SMELL	PHP	340
MENOR	No se deben utilizar caracteres de tabulación	CODE_SMELL	PHP	334
MENOR	Los literales booleanos no deben ser redundantes	CODE_SMELL	PHP	265
MENOR	"empty()" debe usarse para probar el vacío	CODE_SMELL	PHP	222

MENOR	Las variables locales no utilizadas deben eliminarse	CODE_SMELL	PHP	219
MENOR	Los nombres de variables locales y parámetros de función deben cumplir con una convención de nomenclatura	CODE_SMELL	PHP	200
MENOR	El retorno de expresiones booleanas no debe incluirse en una instrucción "if-then-else"	CODE_SMELL	PHP	128
MENOR	Se debe usar "require_once" y "include_once" en lugar de "require" e "include"	BUG	PHP	126
MENOR	Los nombres de campo deben cumplir con una convención de nomenclatura	CODE_SMELL	PHP	102
MENOR	Se deben usar "&&" y "||"	CODE_SMELL	PHP	94
MENOR	Los nombres de clase deben cumplir con una convención de nomenclatura	CODE_SMELL	PHP	63
MENOR	La etiqueta de cierre "?>" debe omitirse en los archivos que contienen solo PHP	CODE_SMELL	PHP	49
MENOR	La imagen, el área y el botón con etiquetas de imagen deben tener un atributo "alt"	BUG	HTML	45
MENOR	La visibilidad del método debe declararse explícitamente	BUG	PHP	44
MENOR	Las etiquetas "<fieldset>" deben contener una "<leyenda>"	BUG	HTML	36
MENOR	Las sentencias "switch" deben tener al menos 3 cláusulas "case"	CODE_SMELL	PHP	27
MENOR	Las variables locales no deben declararse y luego devolverse o lanzarse inmediatamente	CODE_SMELL	PHP	12
MENOR	La palabra clave "elseif" debe usarse en lugar de las palabras clave "else if"	CODE_SMELL	PHP	6
MENOR	No se deben ignorar los valores iniciales de los parámetros de función y método	BUG	PHP	2
MENOR	Una llave cerrada debe ubicarse al principio de una línea	CODE_SMELL	PHP	2
MENOR	Las declaraciones vacías deben eliminarse	CODE_SMELL	PHP	2
MENOR	No se debe utilizar la palabra clave "var"	CODE_SMELL	PHP	2
MENOR	Las instrucciones de salto no deben ser redundantes	CODE_SMELL	PHP	1

INFO	Seguimiento de los usos de las etiquetas "TODO"	CODE_SMELL	PHP	12

Tabla 18 Desglose de problemas

Del análisis sobre los puntos de acceso y seguridad en análisis de vulnerabilidades de SonarQube, ha generado los siguientes problemas que se detallan en la tabla 19.

Puntos de acceso de seguridad			
Tipo	**Problema**	**Prioridad**	**Cantidad**
auth	Se ha detectado 'contraseña' en el nombre de esta variable, revise esta credencial potencialmente codificada	Alta	2
Weak-cryptography	Asegúrese de que el uso de este generador de números pseudoaleatorios sea seguro aquí	Media	17
Log-injection	Asegúrese de que la configuración de este registrador sea segura.	Baja	15
Others	Asegúrese de que no usar la función de integridad de recursos sea seguro aquí	Baja	58

Tabla 19 Puntos de acceso de seguridad

En la tabla 20 encontramos una calificación general del proyecto basados en los parámetros de confiabilidad, seguridad, mantenibilidad y la deuda técnica.

Calificaciones generales de calidad del código	
Calificación de confiabilidad	E
Calificación de seguridad	E
Calificación de mantenibilidad	A
Deuda Técnica	85d 2h 41min

Tabla 20 Calificación de SonarQube

Recomendaciones:

- Realizar pruebas unitarias para aumentar la cobertura del código.
- Corregir problemas críticos identificados en el código.

4.3.2. Prototipo de mejora para puntos de acceso de seguridad

El prototipo de mejora para puntos de acceso de seguridad de la plataforma tecnológica GastroEc, tiene como objetivo fortalecer la protección contra las amenazas y vulnerabilidades detectadas. Se centra en proporcionar una capa adicional de seguridad sin comprometer la usabilidad y la accesibilidad para los usuarios autorizados.

Problema	**Las credenciales no deben estar codificadas de forma rígida**

Solución propuesta	EL problema ha sido identificado en las credenciales de pusher dentro del archivo rjxtools_helper.php en las líneas 846 y 863, para solucionar este problema se debe seguir este procedimiento:
	
	1. Crear de la de carpeta 'maincoregec\application\config' un archivo llamado pusher.php con el siguiente código:
	
	`<?php`
	`$config['pusher']['app_key'] = "[redacted]";`
	`$config['pusher']['app_secret'] = "[redacted]";`
	`$config['pusher']['app_id'] = "[redacted]";`
	
	2. Modificar 'maincoregec\application\config\autoload.php':
	
	Original
	
	`$autoload['config'] = array();`
	Modificada
	`$autoload['config'] = array('aws','pusher','firebase');`
	
	3. Modificar 'maincoregec\ci_core\application\helpers\rjxtools_helper.php' para llamar la configuración de claves utilizar dentro de las llamadas de pusher:
	
	Original

```
842         $options = array(
843             'cluster' => 'us2',
844             'useTLS' => true
845         );
846         $pusher = new Pusher\Pusher(
847             '                    ',
848             '                    ',
849             '      ',
850             $options
851         );
```

Modificada

```
841         $CI = &get_instance();
842         $pusher_config = $CI->config->item( 'pusher' );
843         $options = array(
844             'cluster' => 'us2',
845             'useTLS' => true
846         );
847         $pusher = new Pusher\Pusher(
848             $pusher_config['mi_app_key'],
849             $pusher_config['mi_app_secret'],
850             $pusher_config['mi_app_id'],
851             $options
852         );
```

Beneficios	Mantener las claves de servicios como Pusher en la configuración en lugar de en el código mejora la seguridad, facilita la gestión, promueve la mantenibilidad del código y sigue las mejores prácticas de desarrollo seguro.
Recomendaciones	Cambiar las credenciales por si estuvieran comprometidas.

Tabla 21 Solución - Codificación de credenciales

Problema	**Se ha detectado 'contraseña' en el nombre de esta variable, revise esta credencial potencialmente codificada.**
Solución propuesta	EL problema ha sido identificado en las credenciales de firebase dentro del archivo Firebase_model.php en la línea 20, para solucionar este problema se debe seguir este procedimiento: 1. Crear de la de carpeta ‘maincoregec\application\config’ un archivo llamado firebase.php con el siguiente código:

```php
<?php
$config['pvd']['email'] = "                    ";
$config['pvd']['password'] = "                    F";
```

2. Modificar ‘maincoregec\application\config\autoload.php’:

Original

```php
$autoload['config'] = array();
```

Modificada

```php
$autoload['config'] = array('aws','pusher','firebase');
```

3. Modificar ‘maincoregec\ci_core\application\models\Firebase_model.php’ inicializar los valores de las propiedades en el constructor:

Original

```php
public $pvdEmail = "                    ";
public $pvdPassword = "                    ";

public function __construct()
{
    parent::__construct();
    $this->load->database();

    $factory = (new Factory)
```

Recomendación

```php
public $pvdEmail = null;
public $pvdPassword = null;

public function __construct()
{
    parent::__construct();
    $this->load->database();
    $CI = &get_instance();
    $pvd_config = $CI->config->item( 'pvd' );
    $this->pvdEmail = $pvd_config['email'];
    $this->pvdPassword = $pvd_config['password'];
```

Beneficios	Mantener las claves de servicios como Firebase en la configuración en lugar de en el código mejora la seguridad, facilita la gestión, promueve la mantenibilidad del código y sigue las mejores prácticas de desarrollo seguro.
Recomendaciones	Cambiar las credenciales por si estuvieran comprometidas.

Tabla 22 Solución - Nombre de variable

Problema	**Asegúrese de que el uso de este generador de números pseudoaleatorios sea seguro aquí**
Solución propuesta	En lugar de usar la función rand(); utilizar mt_rand().
Beneficios	La función rand() retornar los mismo valores cada determinado tiempo en caso de que sea usado para claves daría problemas de identidad por eso la opción es usar mt_rand() que tiene algoritmos de generación de número aleatorios mejorado.

Tabla 23 Solución - Función rand()

Problema	**Asegúrese de que este algoritmo hash débil no se utilice en un contexto sensible aquí.**
Solución propuesta	Se recomienda usar algoritmos alternos de encriptación; en lugar de usar md5 o sha1 usar sha512 siempre y cuando sea aplicable y no afecte la funcionalidad del sistema. Original $hash = hash(‘md5’, $data); $hash = hash(‘sha1’, $data); Recomendación $hash = hash(‘sha512’, $data);
Beneficios	Previene ataques de fuerza bruta y diccionario.

Tabla 24 Solución - Función hash

Problema	**Asegúrese de que la configuración de este registrador sea segura**
Solución propuesta	Verifique en el código de producción solo exista un solo lugar donde se establezca el registro de errores. Se han detectado 15 lugares donde se activa mostrar los errores: 'ini_set('display_errors', 1)'; debe eliminar esas líneas.
Beneficios	Al usar la línea 'ini_set('display_errors', 1)' se podría mostrar información relevante a usuario no registrados; al quitar esta línea de los archivos que no sea de configuración garantiza un mejor uso del despliegue de errores y evita mostrar información confidencial.

Tabla 25 Solución - Registro de errores

Problema	**Asegúrese de que no usar la función de integridad de recursos sea seguro aquí**
Solución propuesta	Al utilizar recurso de javascript remotos sean seguros a través de la cláusula 'integrity'. Por ejemplo: Original <script src="https://cdn.jsdelivr.net/npm/summernote@0.8.18/dist/summernote.min.js"></script>
Beneficios	Proporciona un mecanismo para verificar la integridad de los recursos externos. Se utiliza una función hash criptográfica para generar un valor hash único para el archivo externo.

Tabla 26 Solución - Integridad de recursos

4.3.3. Análisis después de aplicar cambios de SonarQube

Una vez analizado las vulnerabilidades de la plataforma GastroEc se procedió a corregir las vulnerabilidades críticas por varias razones, y estas se relacionan directamente con la seguridad, la integridad y la confidencialidad de la información.

Figura 13 Pos Escaneo SonarQube

Las vulnerabilidades y fallos detectados con el sistema SonarQube, han sido identificados y mitigados, esto ha permitido garantizar la seguridad del sistema, la adopción de estrategias de pruebas de penetración, implementación de buenas prácticas de seguridad y actualizaciones regulares para reducir el riesgo de explotación de la plataforma GastroEC.

Nuevas clasificaciones de calidad de código	
Calificación de confiabilidad en el nuevo código	A
Calificación de seguridad en el nuevo código	A
Calificación de mantenibilidad en código nuevo	B
Deuda técnica en el nuevo código	2h 36min

Tabla 27 Nuevas clasificaciones de calidad de código

4.4 Discusión de resultados

Del análisis de vulnerabilidades de la plataforma de software GastroEc, con las herramientas OWASP ZAP y SonarQube, se logró detectar debilidades que comprometían la seguridad de la aplicación. Las vulnerabilidades detectadas fueron errores de programación, configuraciones inseguras y falta de actualizaciones, entre otras.

4.4.1. Discusión de resultados OWASP ZAP

Una vez realizado el análisis de las vulnerabilidades mediante la implementación de buenas prácticas de desarrollo seguro, pruebas exhaustivas del software y la modificación a nivel de código fuente, los desarrolladores de la empresa han tomado medidas proactivas para proteger la plataforma GastroEC, así como los datos reduciendo así el riesgo de incidentes de seguridad.

Tipo de alerta	Impacto		Probabilidad		Riesgo	
	antes	después	antes	después	antes	después
Inyección SQL	6.50	0.00	2.63	0.00	Medio	Nota
Inyección SQL – MySQL						
Ausencia de fichas (tokens) Anti-CSRF	3.36	0.39	1.38	0.00	Bajo	Nota
Cabecera Content Security Policy (CSP) no configurada						
Desconfiguración de Dominio cruzado						
Falta de cabecera Anti-Clickjacking						
Filtrado de información en .htaccess						
Hidden File Found (Archivo Oculto Encontrado)						
Librería JS Vulnerable						
Cookie No HttpOnly Flag	2.75	2.06	1.38	1.03	Nota	Nota
Cookie Without Secure Flag						
Cookie sin el atributo SameSite						
Cross-Domain JavaScript Source File Inclusion						
Divulgación de la marca de hora – Unix						
Server Leaks Version Information via "Server" HTTP Response Header Field						
Strict-Transport-Security Header Not Set						
X-Content-Type-Options Header Missing						
Authentication Request Identified	2.75	1.22	0.88	0.39	Nota	Nota
Content-Type Header Missing						
Divulgación de información - Comentarios sospechosos						
GET para POST						
Modern Web Application						
Re-examine Cache-control Directives						
Retrieved from Cache						
Session Management Response Identified						
User Agent Fuzzer						

Tabla 28 Comparativas de resultados de vulnerabilidades OWAS ZAP

De los resultados se determina que se han reducido significativamente las vulnerabilidades detectadas especialmente aquellas que fueron consideradas amenazas graves contra la seguridad, producto de la modificación y reparación a nivel de tipo y grado de vulnerabilidad a la que estaba expuesto la plataforma GastroEC.

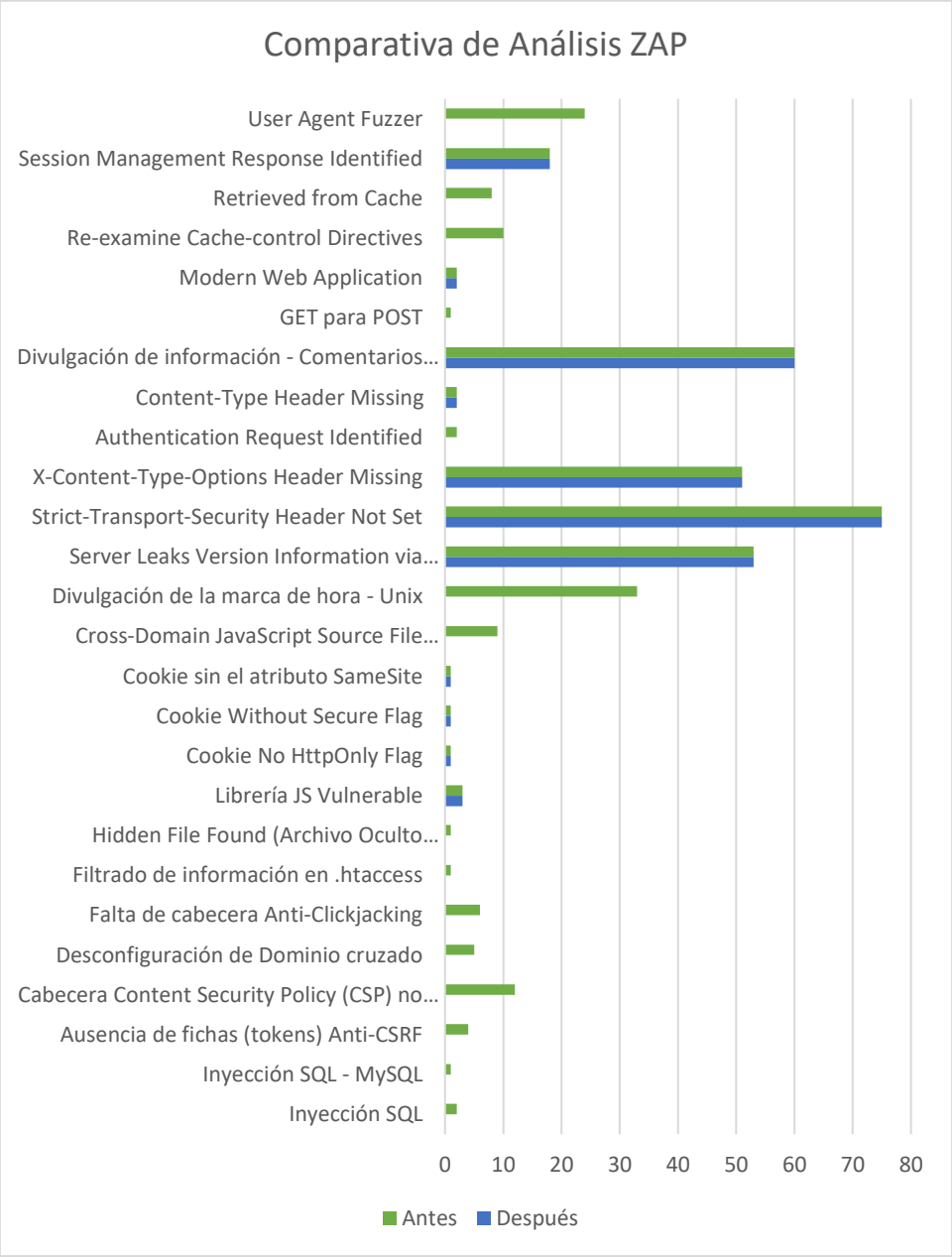

Figura 14 Comparación de resultados Owasp

4.2.2. Discusión de resultados SonarQube

De resultados de un análisis de vulnerabilidades de SonarQube destaca los hallazgos significativos y las acciones recomendadas para actualizar la seguridad del código modificando las vulnerabilidades, errores y malas prácticas de programación en el software. La discusión de resultados se centra en comprender la gravedad de estas vulnerabilidades, priorizarlas y proponer soluciones.

Tipo de Problemas	Antes	Después
Problemas con los bloqueadores	32	0
Cuestiones críticas	997	893
Principales problemas	7.652	7.559
Problemas menores	7.093	6837
Problemas de información	12	12

Tabla 29 Análisis comparativo de vulnerabilidades SonarQube

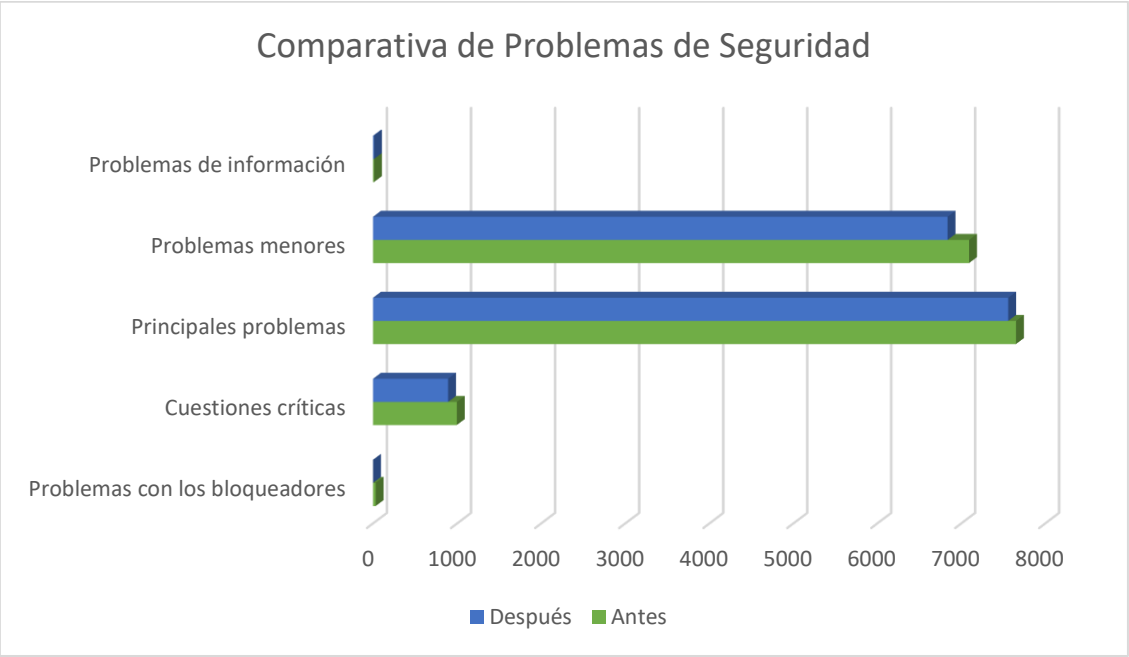

Figura 15 Comparativa de Problemas de Seguridad

En la figura 16 observaremos una comparativa respecto al inicio de la evaluación y al final, una vez aplicadas ciertas recomendaciones notaremos como se elevó el parámetro de seguridad, los cuales eran prioridad para esta investigación.

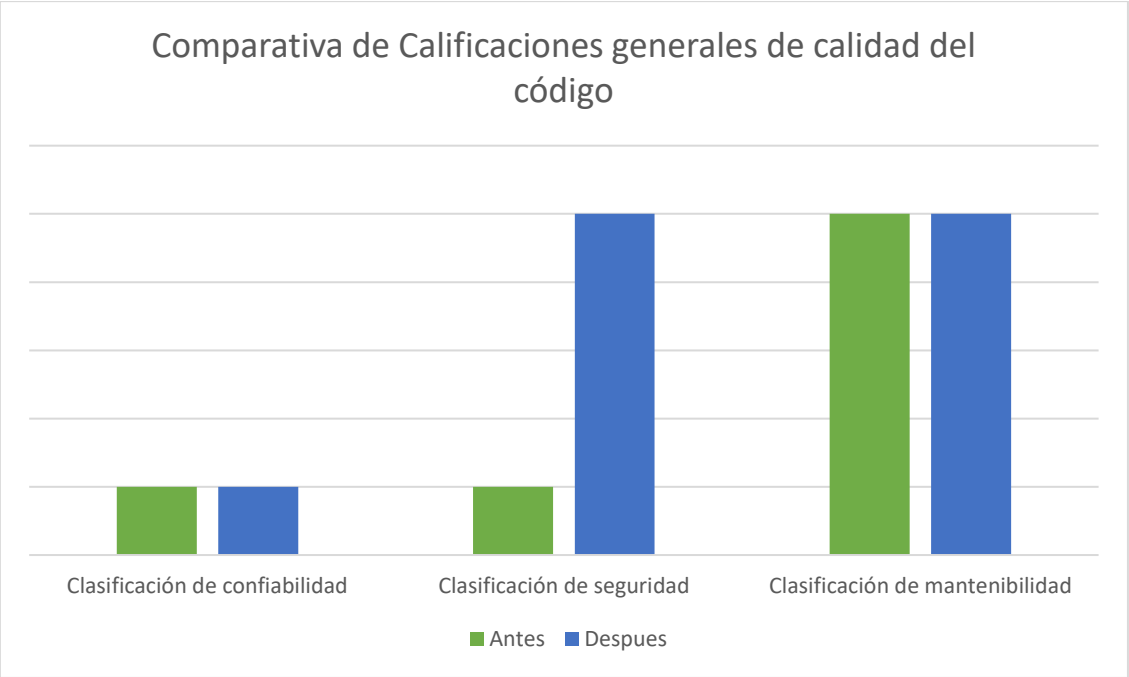

Figura 16 Comparativa de Calificaciones generales de calidad del código

El análisis de resultados con SonarQube nos muestra que hubo una mejora considerable en aspectos de seguridad respecto a la codificación, lo que nos indica que las recomendaciones aplicadas en los diferentes módulos del sistema pueden ser tomadas en cuenta para ser ejecutadas de manera general en todo el proyecto y así tener como resultado un sistemas más robusto a nivel de código.

CONCLUSIONES

- La evaluación de seguridad realizada hacia la plataforma tecnológica GastroEc mediante las herramientas Owasp Zap, SonarQube demostraron que el sistema cuenta con varias vulnerabilidades tanto a nivel de código fuente como en su versión de producción, evidenciando debilidades en áreas críticas como lo es el ingreso de usuarios y manejo de funcionalidades del cliente en donde un ataque pondría en riesgo la confidencialidad, integridad y disponibilidad del sistema.

- La prueba de concepto realizada a la plataforma GastroEc mediante el uso y guías de las herramientas SonarQube y Owasp Zap y tomando en cuenta consejos brindados para la mitigación de vulnerabilidades y correcciones a nivel de código fuente resultaron ser exitosas y beneficiosas para la plataforma. Destacando así el uso de estas herramientas que trabajando en conjunto no solamente pueden ayudar a fortalecer de un sistema, sino que también pueden resultar en mejores prácticas de desarrollo dando como resultado un sistema más seguro y robusto. reducir

- La evaluación de seguridad realizada a la plataforma GastroEc destaco la presencia de vulnerabilidades relacionadas al ambiente de desarrollo, esto por el hecho de utilizar versiones desactualizadas de las herramientas de trabajo (librerías, framework, lenguaje de programación) las mismas que suelen ser mitigadas y corregidas en versiones más recientes.

- La evaluación de seguridad realizada posteriormente a la plataforma GastroEc, tomando en cuenta las recomendaciones y aplicando las medidas preventivas aplicadas obtuvo buenos resultados, mitigando vulnerabilidades consideradas con un riesgo alto tanto a nivel de código fuente como en el sistema de producción, mostrando un impacto positivo en las correcciones recomendadas dando como resultado un sistema más robusto y seguro

RECOMENDACIONES

- Realizar análisis de seguridad a la plataforma GastroEc de manera periódica, se podrían utilizar las mismas herramientas (SonarQube, Owasp Zap) generando informes en los que quede detalle todo lo encontrado para que en actualizaciones posteriores sean tomadas en cuenta

- Al momento de realizar alguna actualización al sistema se podría ejecutar el escaneo de código estático que brinda SonarQube para verificar la calidad y seguridad del código que se va a implementar y en caso de detectar algún problema de los que encuentra la herramienta corregirlo en el momento.

- Actualizar las herramientas en las que está desarrollada la plataforma principalmente migrar a la última versión estable al framework core Codeigniter, esto derivaría en actualizaciones tanto de librerías como del lenguaje de programación.

REFERENCIAS

Alvarado, C. (2020). ANÁLISIS DE ATAQUES CIBERNÉTICOS HACIA EL ECUADOR. *Revista Científica Aristas*. Obtenido de https://revistacientificaistjba.edu.ec/images/home/documentos/Mayo_2020/2.pdf

Candel, J. M. (2019). *Seguridad en aplicaciones web Java.* Ediciones de la U.

Cero, S. (2023). *Escaneo de vulnerabilidades automático con OWASP ZAP*. Recuperado el 2023, de https://academy.seguridadcero.com.pe/blog/escaneo-vulnerabilidades-autom%C3%A1tico-OWASP-ZAP

Cordovilla, F. E., Ordoñez Sigcho, I. B., Peñaherrera-Larenas, M. F., & Suárez-Matamoros, V. J. (2020). Importancia de la seguridad de los sistemas de información frente el abuso, error. *Dominio de las ciencias*.

David Arroyo Guardeño, V. M. (2020). *Ciberseguridad.* CSIC y Catarata.

Foundation, O. (s.f.). GUÍA DE PRUEBAS OWASP. Recuperado el 2024, de https://owasp.org/www-pdf-archive/Gu%C3%ADa_de_pruebas_de_OWASP_ver_3.0.pdf

GastroEc. (2024). *GastroEc*. Recuperado el 2023, de https://gastro-ec.com/

Maino, V. (2022). ESTRATEGIA NACIONAL DE CIBERSEGURIDAD DEL ECUADOR. Obtenido de https://asobanca.org.ec/wp-content/uploads/2022/08/ESTRATEGIA-NACIONAL-DE-CIBERSEGURIDAD-DEL-ECUADOR-2022481.pdf

Mena, E. A. (2021). *Fundamentos de seguridad informática.* COMPAS.

Moreno García, M. (2022). *Gestión de incidentes de ciberseguridad.* RA_MA.

Moya, J. G. (2023). La importancia de la seguridad informática en la educación digital: retos y soluciones. *ReciMundo*, 8.

Onyenweaku, I. R. (2021). A SonarQube Static Analysis of the Spectral Workbench. *International Journal of Natural Science and Reviews*, 15.

Palacios, A. P. (2020). *Seguridad Informática.* Paraninfo.

SonarQube. (s.f.). *SonarQube*. Obtenido de https://docs.sonarqube.org/latest/

Suárez, J. L. (2020). *IMPORTANCIA DE LA SEGURIDAD INFORMÁTICA Y CIBERSEGURIDAD EN EL MUNDO ACTUAL.* Universidad Piloto de Colombia. Obtenido de http://repository.unipiloto.edu.co/bitstream/handle/20.500.12277/8668/IMPORTANCIA%20DE%20LA%20SEGURIDAD%20INFORM%C3%81TICA%20Y%

20CIBERSEGURIDAD%20EN%20EL%20MUNDO%20ACTUAL.pdf?sequence=1&isAllowed=y

Suárez, J. L. (2021). *IMPORTANCIA DE LA SEGURIDAD INFORMÁTICA Y CIBERSEGURIDAD EN EL MUNDO ACTUAL.* Colombia. Obtenido de http://repository.unipiloto.edu.co/bitstream/handle/20.500.12277/8668/IMPORTANCIA%20DE%20LA%20SEGURIDAD%20INFORM%C3%81TICA%20Y%20CIBERSEGURIDAD%20EN%20EL%20MUNDO%20ACTUAL.pdf?sequence=1&isAllowed=y

Tejada, E. C. (2023). *Gestión de incidentes de seguridad informática.* IC Editorial.

Vega, E. (03 de 2021). *SEGURIDAD DE LA INFORMACIÓN.* Obtenido de https://3ciencias.com/: https://3ciencias.com/wp-content/uploads/2021/03/LIBRO-SEGURIDAD-INFORMACIO%CC%81N.pdf

Yeison Molina Marin, L. G. (2020). Vulnerabilidades de los Sistemas de Información: una revisión.

Zafra, G. A. (2017). *HERRAMIENTAS DE PRUEBA DE SEGURIDAD DE APLICACIONES.*

Zambrano, K. B., & Willians Eduardo Basurto Vidal, R. R. (2022). VULNERABILIDADES EN LOS SISTEMAS INFORMÁTICOS OWASP TOP 10: REVISIÓN BIBLIOGRÁFICA. *Journal Business Science*.

ZAP. (s.f.). Obtenido de https://www.zaproxy.org/pdf/ZAPGettingStartedGuide-2.11.pdf

Anexos

Pérdida de confidencialidad	¿Cuanta información podría ser revelada y cuán delicada es? Revelación mínima de datos no sensibles (2), revelación mínima de datos críticos (6), amplia revelación de datos no sensibles (6), amplia revelación de datos críticos, todos los datos revelados (9)
Pérdida de integridad	¿Cuantos datos se podrían corromper y cuanto sería el daño sufrido? Mínimo, datos ligeramente corruptos (1), mínimos datos seriamente dañados (3), gran cantidad de datos ligeramente dañados (5), gran cantidad de datos seriamente dañados, todos los datos totalmente corruptos (9)
Pérdida de disponibilidad	¿Cuantos servicios se pueden ver interrumpidos y cuán vitales son? Mínimo número de servicios secundarios interrumpidos (1), mínimo número de servicios primarios interrumpidos (5), gran número de servicios secundarios interrumpidos (5), gran número de servicios primarios interrumpidos (7), todos los servicios perdidos (9)
Pérdida de responsabilidad	¿Se pueden trazar las acciones de los atacantes hasta llegar a un individuo? Totalmente trazable (1), es posible que se pueda trazar (7), completamente anónimo (9)

Tabla 30 Guía para calificación de Impacto

Facilidad de descubrimiento	¿Es fácil descubrir esta vulnerabilidad? Prácticamente imposible (1), difícil (3), fácil (7), existen herramientas automatizadas disponibles (9)
Facilidad de explotación	¿Hasta qué punto es fácil para estos atacantes explotar esta vulnerabilidad? En teoría es posible explotarla (1), difícil (3), fácil (5), existen herramientas automatizadas disponibles (9)
Conocimiento de la vulnerabilidad	¿Se trata de una vulnerabilidad muy conocida? Desconocida (1), oculta (4), obvia (6), se conoce de forma pública (9)
Detección de la intrusión	¿Con que frecuencia se detecta un exploit? Detección activa en la aplicación (1), registrada y revisada (3), registrada pero no revisada (8), no registrada (9).

Tabla 31Guía para calificación de Vulnerabilidad

Tipo de alerta	Riesgo	Cantidad	perdida de Confidencialidad	integridad	disponibilidad	responsabilidad	Factor de impacto	Promedio
Inyección SQL	Alto	2	9	5	5	7	6.5	6.5
Inyección SQL – MySQL	Alto	1	9	5	5	7	6.5	
Ausencia de fichas (tokens) Anti-CSRF	Medio	4	2	1	5	7	3.75	3.36
Cabecera Content Security Policy (CSP) no configurada	Medio	12	2	1	5	7	3.75	
Desconfiguración de Dominio cruzado	Medio	5	2	2	1	7	3	
Falta de cabecera Anti-Clickjacking	Medio	6	2	3	5	7	4.25	
Filtrado de información en .htaccess	Medio	1	2	3	1	7	3.25	
Hidden File Found (Archivo Oculto Encontrado)	Medio	1	2	1	1	7	2.75	
Librería JS Vulnerable	Medio	3	2	1	1	7	2.75	
Cookie No HttpOnly Flag	Bajo	1	2	1	1	7	2.75	2.75
Cookie Without Secure Flag	Bajo	1	2	1	1	7	2.75	
Cookie sin el atributo SameSite	Bajo	1	2	1	1	7	2.75	
Cross-Domain JavaScript Source File Inclusion	Bajo	9	2	1	1	7	2.75	
Divulgación de la marca de hora – Unix	Bajo	33	2	1	1	7	2.75	
Server Leaks Version Information via "Server" HTTP Response Header Field	Bajo	53	2	1	1	7	2.75	
Strict-Transport-Security Header Not Set	Bajo	75	2	1	1	7	2.75	
X-Content-Type-Options Header Missing	Bajo	51	2	1	1	7	2.75	
Authentication Request Identified	Informativo	2	2	1	1	7	2.75	2.75
Content-Type Header Missing	Informativo	2	2	1	1	7	2.75	
Divulgación de información - Comentarios sospechosos	Informativo	60	2	1	1	7	2.75	
GET para POST	Informativo	1	2	1	1	7	2.75	
Modern Web Application	Informativo	2	2	1	1	7	2.75	

Re-examine Cache-control Directives	Informativo	10	2	1	1	7	2.75	
Retrieved from Cache	Informativo	8	2	1	1	7	2.75	
Session Management Response Identified	Informativo	18	2	1	1	7	2.75	
User Agent Fuzzer	Informativo	24	2	1	1	7	2.75	

Tabla 32 Cálculo de Impacto-Vulnerabilidades

Tipo de alerta	Riesgo	Facilidad de descubrimiento	Facilidad de explotación	Conocimiento de la vulnerabilidad	Detección de la intrusión	Probabilidad	Promedio
Inyección SQL	Alto	3	9	4	5	2.63	2.63
Inyección SQL – MySQL	Alto	3	9	4	5	2.63	
Ausencia de fichas (tokens) Anti-CSRF	Medio	3	3	2	3	1.38	1.38
Cabecera Content Security Policy (CSP) no configurada	Medio	3	3	2	3	1.38	
Desconfiguración de Dominio cruzado	Medio	3	3	2	3	1.38	
Falta de cabecera Anti-Clickjacking	Medio	3	3	2	3	1.38	
Filtrado de información en .htaccess	Medio	3	3	2	3	1.38	
Hidden File Found (Archivo Oculto Encontrado)	Medio	3	3	2	3	1.38	
Librería JS Vulnerable	Medio	3	3	2	3	1.38	
Cookie No HttpOnly Flag	Bajo	3	3	2	3	1.38	1.38
Cookie Without Secure Flag	Bajo	3	3	2	3	1.38	
Cookie sin el atributo SameSite	Bajo	3	3	2	3	1.38	
Cross-Domain JavaScript Source File Inclusion	Bajo	3	3	2	3	1.38	

Divulgación de la marca de hora – Unix	Bajo	3	3	2	3	1.38	
Server Leaks Version Information via "Server" HTTP Response Header Field	Bajo	3	3	2	3	1.38	
Strict-Transport-Security Header Not Set	Bajo	3	3	2	3	1.38	
X-Content-Type-Options Header Missing	Bajo	3	3	2	3	1.38	
Authentication Request Identified	Informativo	1	1	2	3	0.88	
Content-Type Header Missing	Informativo	1	1	2	3	0.88	
Divulgación de información - Comentarios sospechosos	Informativo	1	1	2	3	0.88	
GET para POST	Informativo	1	1	2	3	0.88	
Modern Web Application	Informativo	1	1	2	3	0.88	0.88
Re-examine Cache-control Directives	Informativo	1	1	2	3	0.88	
Retrieved from Cache	Informativo	1	1	2	3	0.88	
Session Management Response Identified	Informativo	1	1	2	3	0.88	
User Agent Fuzzer	Informativo	1	1	2	3	0.88	

Tabla 33Cálculo de Probabilidad-Vulnerabilidad

Tipo de alerta	Riesgo	Cantidad	Perdida de Confidencialidad	integridad	disponibilidad	responsabilidad	Factor de impacto	Promedio
Inyección SQL	Alto	0	0	0	0	0	0	0.00
Inyección SQL – MySQL	Alto	0	0	0	0	0	0	
Ausencia de fichas (tokens) Anti-CSRF	Medio	0	0	0	0	0	0	0.39
Cabecera Content Security Policy (CSP) no configurada	Medio	0	0	0	0	0	0	
Desconfiguración de Dominio cruzado	Medio	0	0	0	0	0	0	
Falta de cabecera Anti-Clickjacking	Medio	0	0	0	0	0	0	
Filtrado de información en .htaccess	Medio	0	0	0	0	0	0	
Hidden File Found (Archivo Oculto Encontrado)	Medio	0	0	0	0	0	0	
Librería JS Vulnerable	Medio	3	2	1	1	7	2.75	
Cookie No HttpOnly Flag	Bajo	1	2	1	1	7	2.75	2.06
Cookie Without Secure Flag	Bajo	1	2	1	1	7	2.75	
Cookie sin el atributo SameSite	Bajo	1	2	1	1	7	2.75	
Cross-Domain JavaScript Source File Inclusion	Bajo	0	0	0	0	0	0	
Divulgación de la marca de hora – Unix	Bajo	0	0	0	0	0	0	
Server Leaks Version Information via "Server" HTTP Response Header Field	Bajo	53	2	1	1	7	2.75	
Strict-Transport-Security Header Not Set	Bajo	75	2	1	1	7	2.75	
X-Content-Type-Options Header Missing	Bajo	51	2	1	1	7	2.75	
Authentication Request Identified	Informativo	0	0	0	0	0	0	1.22
Content-Type Header Missing	Informativo	2	2	1	1	7	2.75	
Divulgación de información - Comentarios sospechosos	Informativo	60	2	1	1	7	2.75	
GET para POST	Informativo	0	0	0	0	0	0	
Modern Web Application	Informativo	2	2	1	1	7	2.75	

Re-examine Cache-control Directives	Informativo	0	0	0	0	0	0	
Retrieved from Cache	Informativo	0	0	0	0	0	0	
Session Management Response Identified	Informativo	18	2	1	1	7	2.75	
User Agent Fuzzer	Informativo	0	0	0	0	0	0	

Tabla 34 Cálculo de Impacto-Vulnerabilidades-Final

Tipo de alerta	Riesgo	Cantidad	Facilidad de descubrimiento	Facilidad de explotación	Conocimiento de la vulnerabilidad	Detección de la intrusión	Probabilidad	Promedio
Inyección SQL	Alto	0					0	0.00
Inyección SQL – MySQL	Alto	0					0	
Ausencia de fichas (tokens) Anti-CSRF	Medio	0	0	0	0	0	0	0.20
Cabecera Content Security Policy (CSP) no configurada	Medio	0	0	0	0	0	0	
Desconfiguración de Dominio cruzado	Medio	0	0	0	0	0	0	
Falta de cabecera Anti-Clickjacking	Medio	0	0	0	0	0	0	
Filtrado de información en .htaccess	Medio	0	0	0	0	0	0	
Hidden File Found (Archivo Oculto Encontrado)	Medio	0	0	0	0	0	0	
Librería JS Vulnerable	Medio	3	3	3	2	3	1.375	
Cookie No HttpOnly Flag	Bajo	1	3	3	2	3	1.375	1.03
Cookie Without Secure Flag	Bajo	1	3	3	2	3	1.375	
Cookie sin el atributo SameSite	Bajo	1	3	3	2	3	1.375	
Cross-Domain JavaScript Source File Inclusion	Bajo	0	0	0	0	0	0	
Divulgación de la marca de hora – Unix	Bajo	0	0	0	0	0	0	

Server Leaks Version Information via "Server" HTTP Response Header Field	Bajo	53	3	3	2	3	1.375	
Strict-Transport-Security Header Not Set	Bajo	75	3	3	2	3	1.375	
X-Content-Type-Options Header Missing	Bajo	51	3	3	2	3	1.375	
Authentication Request Identified	Informativo	0	0	0	0	0	0	
Content-Type Header Missing	Informativo	2	1	1	2	3	0.875	
Divulgación de información - Comentarios sospechosos	Informativo	60	1	1	2	3	0.875	
GET para POST	Informativo	0	0	0	0	0	0	
Modern Web Application	Informativo	2	1	1	2	3	0.875	0.39
Re-examine Cache-control Directives	Informativo	0	0	0	0	0	0	
Retrieved from Cache	Informativo	0	0	0	0	0	0	
Session Management Response Identified	Informativo	18	1	1	2	3	0.875	
User Agent Fuzzer	Informativo	0	0	0	0	0	0	

Tabla 35 Cálculo de Probabilidad-Vulnerabilidades-Final