ESCUELA SUPERIOR POLITECNICA DE CHIMBORAZO

Desarrollo de una interfaz web, que permita mejorar la seguridad en la transferencia de estados de servicios web, basado en autenticación y autorización mediante el estándar Json Web Token

LUIS RODRIGO TAYUPANDA TACURI

Trabajo de Titulación modalidad Proyectos de Investigación y Desarrollo, presentado ante el Instituto de Posgrado y Educación Continua de la ESPOCH como requisito parcial para la obtención del grado de:

MAGÍSTER EN SEGURIDAD TELEMÁTICA

Riobamba-Ecuador

Septiembre 2022

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

CERTIFICACIÓN:

EL TRIBUNAL DE TRABAJO DE TITULACIÓN CERTIFICA QUE:

El Trabajo de Titulación modalidad Proyectos de Investigación y Desarrollo, titulado "Desarrollo de una interfaz web, que permita mejorar la seguridad en la transferencia de estados de servicios web, basado en autenticación y autorización mediante el estándar json web token", de responsabilidad del Ing. Luis Rodrigo Tayupanda Tacuri ha sido prolijamente revisado y se autoriza su presentación.

Ing. Luis Eduardo Hidalgo Almeida; Ph. D.
PRESIDENTE

Ing. Washington Gilberto Luna Encalada, Ph. D
DIRECTOR

Ing. Lady Marieliza Espinoza Tinoco; Mag
MIEMBRO

Lic. Carlos Volter Buenaño Pesantez; Mag.
MIEMBRO

Riobamba Septiembre 2022

DERECHOS INTELECTUALES

Yo, Ing. Luis Rodrigo Tayupanda Tacuri declaro que soy responsable de las ideas, doctrinas y resultados expuestos en el Trabajo de Titulación modalidad Proyectos de Investigación y Desarrollo, y que el patrimonio intelectual generado por la misma pertenece exclusivamente a la Escuela Superior Politécnica de Chimborazo.

Luis Rodrigo Tayupanda Tacuri
N° Cedula 0604789693

DECLARACIÓN DE AUTENTICIDAD

Yo, Ing. Luis Rodrigo Tayupanda Tacuri, declaro que el presente proyecto de investigación, es de mi autoría y que los resultados del mismo son auténticos y originales, y el patrimonio intelectual de la misma pertenecen a la Escuela Superior Politécnica de Chimborazo.

Firmado electrónicamente por:
LUIS RODRIGO
TAYUPANDA
TACURI

Luis Rodrigo Tayupanda Tacuri
N° 0604789693

DEDICATORIA

Este trabajo de Investigación va dedicado en primer lugar a Dios, por darme la oportunidad de ver la luz de cada amanecer, la salud, la inteligencia y los conocimientos necesarios, para poder concluir con éxitos unas de mis metas trazadas en mi vida.

A mi padre en el cielo por brindarme el apoyo incondicional cuando estuvo con vida, a mi madre por el apoyo y consejos verbales, ya que sin su apoyo no hubiera sido posible tan anheloso logro.

A mi esposa y a mi hija que con su paciencias y comprensiones han sabido apoyarme en las buenas y las malas.

A mis hermanas y hermano, por darme el aliento y sustento para seguir estudiando día a día, elevándome él autoestima para seguir, a pesar de los tropiezos que he tenido durante mi maestría.

En fin, a toda mi familia y amigos que aportaron con sus buenos deseos para animarme a seguir adelante, en los malos momentos que se me presento en la vida, de esta forma todos fueron un pilar importante para que se pueda lograr este triunfo.

Luis Tayupanda

AGRADECIMIENTO

Agradezco infinitamente a Dios en primera instancia quien es el dador de la vida y salud, y considero que gracias el he tenido la oportunidad de realizar mi maestría.

A mi padre desde el cielo y a mi madre desde la tierra por ser un gran pilar para mi vida quienes me han animado a continuar mis estudios, a formarme profesionalmente y moralmente.

A mi esposa y a mi hija que con su paciencias y comprensiones han sabido apoyarme en las buenas y las malas.

A mis hermanos de quienes aprendo mucho, me han motivado a cumplir este objetivo en mi vida, espero ser un ejemplo para ellos.

A mis docentes, que con todo su esfuerzo no solo me han enseñado aspectos académicos, sino también valores morales y éticos para la vida,

Luis Tayupanda

CONTENIDO

Paginas

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

RESUMEN

El presente trabajo de investigación se realizó con la finalidad de indagar y mitigar las vulnerabilidades que se presentan a diario en los servicios orientados a la web. La seguridad informática es uno de los pilares fundamentales que se debe tomar en cuenta al momento de implementar los servicios web de tipo REST, para mantener la integridad de la información en las transferencias de estado o consumo de los servicios. Por tal razón se ha desarrollado los servicios web de tipo REST conjuntamente con el estándar de seguridad Json Web Token en el sistema académico de la Escuela Superior Politécnica de Chimborazo (ESPOCH). En la presente investigación se realizó los siguientes escenarios, en el escenario 1 se implementó los servicios web de tipo REST del sistema académico sin el uso del estándar de seguridad Json Web Token (JWT), y el escenario 2 se implementó los servicios web de tipo REST del sistema académico con autenticación y el estándar de seguridad Json Web Token, para la transferencia de estados con cada uno de los métodos GET, POST, PUT, y DELETE. La siguiente hipótesis planteada, la implementación de una interfaz Web con el estándar de seguridad Json Web Token garantiza el acceso y autorización segura a los servicios web del sistema académico de la Escuela Superior Politécnica de Chimborazo, aplicando la observación en base a los parámetros evaluados se obtuvo un 92.5% de optimización en el nivel de satisfacción, y a su vez aplicando la herramienta de pentesting Vooki se obtuvo un 80% de optimización de números de vulnerabilidades detectadas en las transferencia de estados, se concluye que el estándar propuesto optimiza el nivel de seguridad en los servicios web que tipo REST y se recomienda la configuración adecuada para la generación del token de seguridad.

Palabras Claves: SEGURIDAD INFORMÁTICA, JSON WEB TOKEN (JWT), PROTOCOLO SEGURO DE TRANSFERENCIA DE HIPERTEXTO (HTTP), SERVICIO WEB REST, VOOKI, VULNERABILIDAD>

0055-DBRA-UPT-IPEC-2022

SUMARY

The present research work was carried out with the purpose of investigating and mitigating the vulnerabilities that occur daily in web-oriented services. Computer security is one of the fundamental pillars that must be taken into account when implementing REST-type web services, in order to maintain the integrity of the information in the state transfers or consumption of the services. For this reason, REST web services have been developed together with the Json Web Token security standard in the academic system of the Escuela Superior Politécnica de Chimborazo (ESPOCH). In the present investigation the following scenarios were carried out, in scenario 1 the REST type web services of the academic system were implemented without the use of the Json Web Token (JWT) security standard, and scenario 2 the REST type web services of the academic system were implemented with authentication and the Json Web Token security standard, for the transfer of states with each of the GET, POST, PUT, and DELETE methods. The following hypothesis, the implementation of a Web interface with the Json Web Token security standard guarantees the access and secure authorization to the web services of the academic system of the Escuela Superior Politécnica de Chimborazo, applying the observation based on the evaluated parameters, 92.5% of optimization in the level of satisfaction was obtained. 5% of optimization in the level of satisfaction, and in turn applying the Vooki pentesting tool was obtained an 80% optimization of vulnerability numbers detected in the state transfer, it is concluded that the proposed standard optimizes the level of security in web services that REST type and the proper configuration for the generation of the security token is recommended.

Keywords: <JWT>, <HTTP>, <REST>, <VOOKI>, <VULNERABILITY>**,** <JSON>**,** <WEB>**,** <TOKEN><ESPOCH>

CAPITULO I

1. INTRODUCION

Hoy en día la seguridad informática y la necesidad de garantizar una alta disponibilidad de la información ha pasado a ser una faceta importantísima para la mayoría de las empresas, e instituciones, independientemente del sector al que pertenezcan, debido a la gran dependencia que éstas tienen con el uso de herramientas informáticas: software de gestión, contabilidad, e información etc. Se debe implementar diferentes posibilidades de mejora de nuestra infraestructura informática y servicios web que podrían prevenir la aparición de posibles problemas de seguridad a posterior. (Romero, 2006)
Con el avance de la tecnología que día a día va creciendo y evolucionado la capacidad de seguridad y escalabilidad de los servicios web REST que nos permite distribuir por medio del internet.

La arquitectura orientados a servicios (SOA) se basa en el desarrollo de servicios altamente reutilizable, los servicios web de tipo REST deben tener una interfaz estándar bien definidas y seguras para que se pueda integrar con cualquier aplicación ya sea web, escritorio o móvil.

Los servicios web de tipo REST significa transferencia de estados representacionales que hoy en día grandes empresas hacen uso por su estándar lógico, eficiente y habitual de creación de APIs para servicios de internet, a su vez tienen un estándar web tales como URIs, HTTP, XML, JSON. El formato de la información que se intercambia lo decidirá el desarrollador de servicios en el back end.

Para la distribución de servicios web de tipo REST es imprescindible hacer uso de tecnologías de seguridad para garantizar la confidencialidad e integridad de los datos, por esta razón los servicios web de tipo REST deberían tener una autenticación, y autorización para poder acceder a la información de manera segura.

Este trabajo de investigación tiene como finalidad de crear, configurar, e implementar la seguridad de los servicios web de tipo REST, implementando una interfaz web, para la transferencia de estados, basada en autenticación y autorización mediante el estándar Json Web Token.

El estándar Json Web Token se define como un mecanismo de poder propagar entre dos partes y de forma segura, la identidad del usuario, además con una serie de privilegios. Estos privilegios están codificados en objetos de tipo Json que se impregna dentro del cuerpo del mensaje que va firmado digitalmente.

1.1 Planteamiento del Problema

1.1.1 Situación Problemática

Actualmente la tecnología y la automatización de procesos, mediante el desarrollo de servicios web de tipo REST han sido herramientas de gran ayuda para todos los negocios, empresas, instituciones financieras, instituciones educativas, etc. Pero a su vez están expuestos a ser vulneradas si no se implementa algún estándar de seguridad que impidan a mantener la integridad de la información.

En la presente investigación se propone desarrollar los servicios web con la arquitectura orientada a servicios REST para la automatización de procesos del sistema académico de la Escuela Superior Politécnica de Chimborazo, dicha tecnología hoy en día es muy flexible, eficiente y eficaz para el consumo de servicios web desde todas los aplicativos y dispositivos. Pero a su vez los servicios web de tipo REST carecen de una cabecera de seguridad para la transferencia de estados representacionales, por tal razón pueden ser vulnerables bajo las amenazas en la red.

Al fin de mitigar dicho inconveniente se considera que los servicios de tipo REST puedan ser menos vulnerables si se implementa un estándar de seguridad, que permita verificar la autenticación y la autorización bajo la creación de un token de seguridad para el consumo de los servicios web, Por tal razón se desea implementar la arquitectura orientada a servicios REST conjuntamente con el estándar de seguridad Json Web Token que servirá para la transferencia de datos seguros entre aplicaciones informáticas.

1.1.2 Formulación del Problema

¿Cómo se puede mejorar la seguridad de la arquitectura orientada a servicios REST para reducir los riesgos de vulnerabilidad bajo las amenazas que se encuentran presentes en la red, debido que dichas tecnologías por naturalidad no son seguras?

1.1.3 Preguntas

¿Existen métodos o estándares de seguridad para el consumo de servicios web de tipo REST?
Hoy en día existen varios métodos o estándares que nos ayuda cubrir esas deficiencias de seguridad que tiene los servicios de tipo REST. A continuación, detallamos los métodos que pueden ser utilizados para precautelar la seguridad como Autenticación, Basic HTTP Authentication, HTTP Digest Access Authentication, OAuth 1.0a, OAuth 2.0 y OpenID Connect (OIDC), y el estándar Json Web Token. En la presente investigación haremos uso del estándar de seguridad Json Web Token para la creación el token de acceso.

¿Cuáles son los riesgos y vulnerabilidades de la arquitectura orientada a servicio web REST?

Los riesgos que pueden tener los servicios web de tipo REST son varios a continuación mencionamos las vulnerabilidad en la red, inyeccion de SQL, perdida de autenticacion y administracion de sesion, cross-site scriting (XSS), referencias inseguras a objetos directos, configuracion erronea de seguridad,exposicion de datos sencibles, falta de control de acceso de nivel de funciones, cross-site request forgery (CSRF), uso de componentes con vulnerabilidades conocidas redireccionamiento y renvio no valido. (Corredor, 2017)

1.2 Justificación de da Investigación

1.2.1 Justificación Teórica

En la actualidad las empresas e instituciones requieren de la asistencia y apoyo de las TICs, en la cual se incluye la arquitectura orientada a servicios, los servicios web de tipo REST como principal herramienta tecnológica para la transferencia, y almacenamiento de la información. Los servicios web de tipo REST brindan grandes ventajas en las aplicaciones informáticas al ser procesos informáticos que se encuentran en el servidor y están disponibles para los terminales o clientes de una red.

La arquitectura orientada a servicios REST brinda mayor flexibilidad, escalabilidad, y reusabilidad debido al uso de protocolos HTTP, también ofrece una variedad de soluciones de software basado en estándares para integrar aplicaciones y automatizar procesos de transferencia de información confidencial, por tal razón la seguridad de los servicios web se debe considerar una característica muy importante para las entidades que tiene como objetivo ofrecer un mejor servicio al usuario, proporcionando una infraestructura completa que permita el intercambio de información de manera segura.

Los inconvenientes o desventajas que tiene este tipo de tecnología es que pueden ser muy vulnerables a ataques que están presente en la red, pudiendo ocasionar suplantación de identidad, inyección SQL, causando la pérdida o modificación de la información, por tal razón es de vital importancia utilizar diferentes tipos de métodos o estándares para precautelar la seguridad a los servicios web de tipo REST.

A fin de resolver los problemas antes planteados se propone el diseño e implementación de una arquitectura orientada a servicio web REST con autenticación y autorización mediante el estándar de seguridad Json Web Token, para precautelar el acceso y seguridad de consumos de los servicios web. Json Web Token es un estándar de seguridad para la creación de token de acceso que permite la propagación de identidad y privilegios, los Json Web Token está compuesto de tres partes un encabezado o header, un contenido o payload, y una firma o signature. El

encabezado identifica que algoritmo se emplea para generar la firma digital, en esta investigación se utilizara el algoritmo HMAC-SHA256, El contenido contiene la información de los privilegios del token, y la firma se calcula codificando el encabezado y el contenido en base64.

1.2.2 Justificación Metodológica

Basados en las metodologías existentes como Json Web Token es un conjunto de medios de seguridad que permiten gestionar la seguridad de los servicios web de tipo REST. Pese a ser un elemento fundamental a la hora de desarrollar un servicio web, no todas las empresas e instituciones lo aplican debido a desconocimiento o adaptaciones a nuevos estándares de seguridad. El objetivo de la implementación de este estándar es con la finalidad de tratar de mitigar o reducir los riesgos de vulnerabilidad de los servicios web de tipo REST.

Los JWT están formados por tres partes:

Encabezado (header): identifica el algoritmo utilizado para generar, normalmente HS256

Contenido (payload): Contiene la información sobre la identidad del usuario y sus privilegios.

Firma (signature): se calcula codificando el encabezado y el contenido en base64url, concadenándose ambas partes con un punto.

1.2.3 Justificación Practica

Con el desarrollo e implementación de una interfaz web para la trasferencia de estados de servicios web con la autenticación y autorización con el estándar Json Web Token se pretende aumentar el grado de seguridad de los servicios web de tipo REST del sistema académico de la ESPOCH utilizando las herramientas y estándares que se realizara en dos escenarios o ambiente, el primero escenario se realizara sin la implementación del estándar de seguridad Json Web Token y el segundo escenario se realizara con la implementación, del estándar de seguridad Json Web Token y de esta manera verificar ambos escenarios para determinar el nivel de seguridad de cada uno de ellos.
Existen varias metodologías o estándares de seguridad que son adaptables para los servicios web de tipo REST en la presente investigación se hará uso del estándar de seguridad Json Web Token con su debida autenticación y autorización para cada uno de los servicios web.

Los JSON Web Token (JWT) es un estándar abierto (RFC 7519) que define una forma compacta y autónoma de transmitir información de forma segura entre las partes como un objeto JSON. Esta información puede ser verificada y confiable porque está firmada digitalmente. Los JWT se pueden firmar usando un secreto (con el algoritmo HMAC) o un par de claves pública y privada usando RSA o ECDSA. (Wikipedia, 2021)

Aunque los JWT se pueden cifrar para proporcionar también secreto entre las partes, nos centraremos en los tokens firmados. Los tokens firmados pueden verificar la integridad de los reclamos contenidos en él, mientras que los tokens encriptados ocultan esos reclamos a otras partes. Cuando los tokens se firman utilizando pares de claves públicas y privadas, la firma también certifica que solo la parte que posee la clave privada es la que la firmó.

Las afirmaciones en un JWT son codificadas como un objeto JSON que está firmado digitalmente utilizando una firma web JSON (JWS) y, opcionalmente, cifrado mediante JSON Web Encryption (JWE). Esta especificación fue desarrollada en colaboración sobre la base de las aportaciones de una serie de precursores desarrollados independientemente de especificaciones de cifrado, firma y token JSON. Ya existen varias implementaciones independientes e interoperables de JWT. (Jones, 2011)

1.3 Objetivos de la Investigación

A continuación, se detallan los objetivos generales y específicos de la presente investigación.

1.3.1 Objetivo General

- ✓ Analizar e implementar una interfaz web, para la transferencia de estados, basada en autenticación y autorización mediante el estándar de seguridad Json Web Token.

1.3.2 Objetivos Específicos

- ✓ Investigar el estándar de seguridad Json Web Token para la autenticación y acceso seguro a los servicios del sistema académico de la Escuela superior Politécnica de Chimborazo.
- ✓ Analizar los servicios web del sistema académico de la Escuela Superior Politécnica de Chimborazo, con la tecnología tipo REST.
- ✓ Desarrollar el token de seguridad con el algoritmo de cifrado RSA256 para la creación de la firma digital.
- ✓ Implementar un ambiente de pruebas para la arquitectura orientada a servicios REST, y poder evaluar la seguridad.

1.4 Hipótesis

Al implementar una interfaz Web con el estándar de seguridad Json Web Token garantiza el acceso y autorización segura a los servicios web del sistema académico de la Escuela Superior Politécnica de Chimborazo.

CAPITULO II

2. MARCO TEORICO

2.1 Json Web Token

JSON Web Token (JWT) es un estándar abierto basado en JSON propuesto por IETF (RFC 7519) para la creación de tokens de acceso que permiten la propagación de identidad y privilegios. Un servidor podría generar un token indicando que el usuario tiene privilegios de administrador y proporcionarlo a un cliente. El cliente entonces podría utilizar el token para probar que está actuando como un administrador en el cliente o en otro sistema. El token está firmado por la clave del servidor, así que el cliente y el servidor son ambos capaz de verificar que el token es legítimo. Los JSON Web Tokens están diseñados para ser compactos, poder ser enviados en las URLs -URL-safe- y ser utilizados en escenarios de Single Sign-On (SSO). Los privilegios de los JSON Web Tokens puede ser utilizados para propagar la identidad de usuarios como parte del proceso de autenticación entre un proveedor de identidad y un proveedor de servicio, o cualquiera otro tipo de privilegios requeridos por procesos empresariales. (Jones B. , 2015)

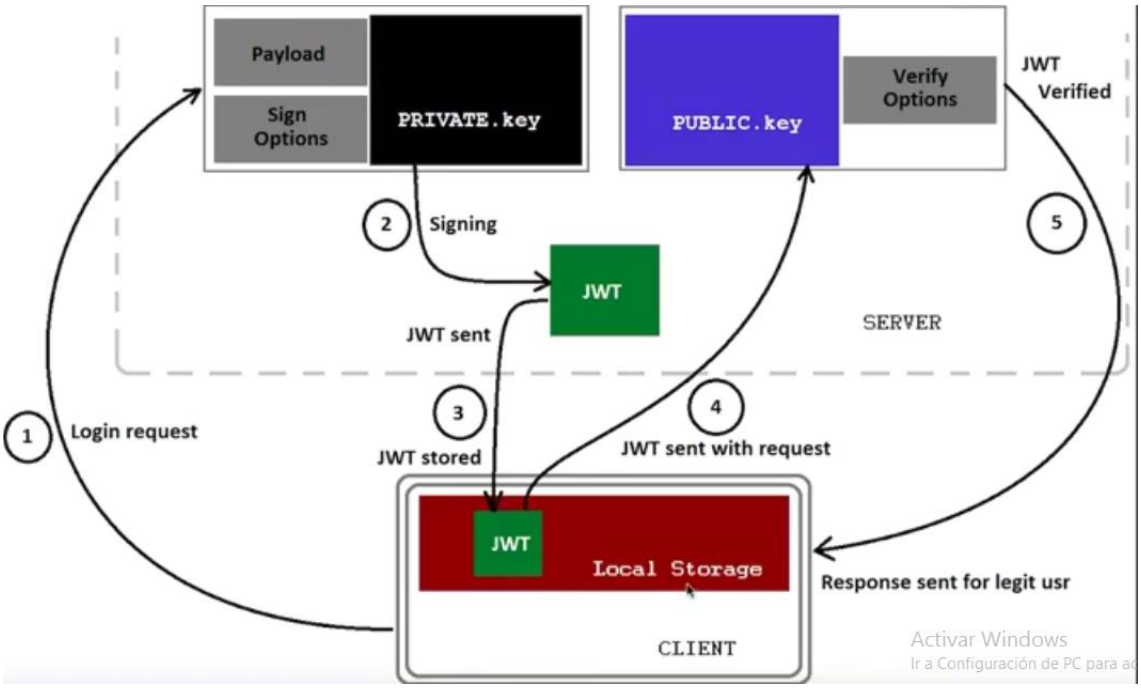


Figura 1-2: Json Web Token
Realizado por. Tayupanda Luis, 2021.

2.2 Algoritmo SHA256

En el algoritmo SHA-256 es la función de compresión que opera sobre bloques de mensaje de 512 bits y con valores de hash intermedios de 256 bits. Esencialmente se trata de un algoritmo de cifrado el cual cifra un valor de hash intermedio utilizando el bloque del mensaje como clave. Para comenzar, el mensaje sobre el que se aplicará la función resumen debe ser extendido hasta

formar un mensaje cuya longitud sea múltiplo de 512 bits. Posteriormente, dicho mensaje será dividido en bloques de 512 bits que se irán proporcionando a la función hash de uno en uno. (Zone, 2021)

2.3 Aplicaciones Web

En la Ingeniería de software se dice aplicación web, aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet, mediante un navegador. En otras palabras, es una aplicación (Software) que se codifica en un lenguaje soportado por los navegadores web en la que se confía la ejecución al navegador.

Las aplicaciones orientados a la web son muy populares debido a lo práctico del navegador web como cliente ligero, a la independencia del Sistema Operativo, así como a la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software a miles de usuarios potenciales.

Es significativo mencionar que una Página Web puede contener elementos que permiten una comunicación activa entre el usuario y la información. Esto permite que el usuario acceda a los datos de modo interactivo, gracias a que la página responderá a cada una de sus acciones, como por ejemplo rellenar y enviar formularios, participar en juegos diversos, y acceder a gestores de base de datos. (EcuRed, 2016)

2.4 Servicio web

Normalmente nos referimos con Servicio Web a una colección de procedimientos, métodos a los que podemos llamar desde cualquier lugar de Internet o de nuestra intranet, siendo este mecanismo de invocación totalmente independiente de la plataforma que utilicemos y del lenguaje de programación en el que se haya implementado internamente el servicio. Los servicios Web son componentes de aplicaciones distribuidas que están disponibles de forma externa. Se pueden utilizar para integrar aplicaciones escritas en diferentes lenguajes y que se ejecutan en plataformas diferentes. (Alicante, 2014)

Los servicios web son métodos que están publicados en un servidor web y pueden ser invocados por medio del internet usando mensajería XML basado en estándares como SOAP, WSDL y UDDI. (Quispe, 2017)

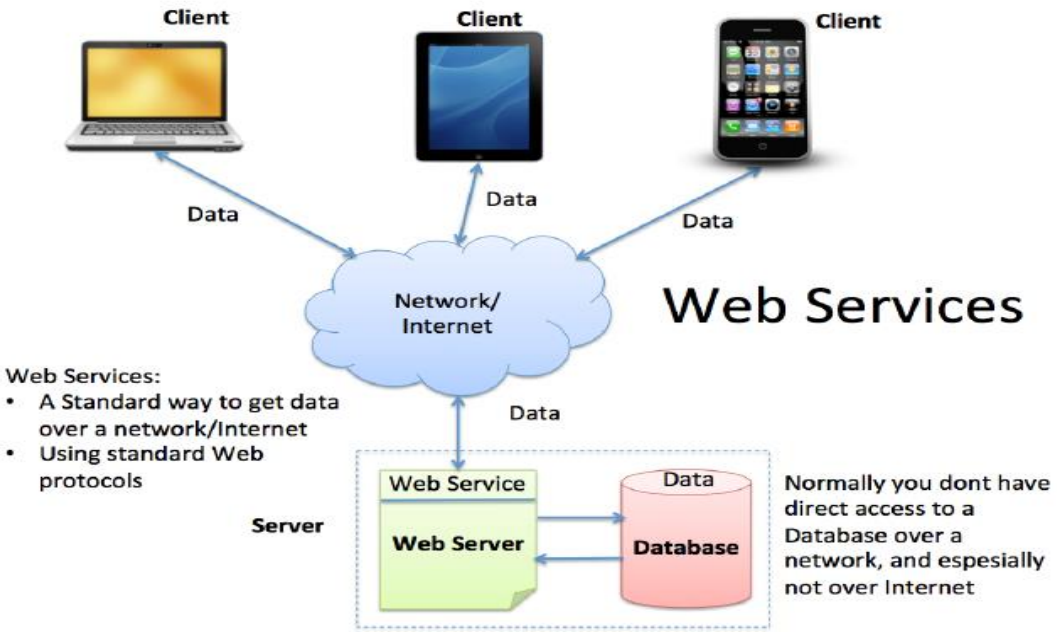


Figura 2-2: Servicio Web
Realizado por. Tayupanda Luis, 2021.

2.5 Servicios web SOAP

SOAP se define como un protocolo estándar de comunicación (conjunto de reglas), un intercambio de mensajes basado en la especificación de XML. SOAP utiliza diferentes protocolos de transporte, tales como HTTP y SMTP. El protocolo HTTP estándar hace que sea más fácil para el modelo de SOAP para túnel a través de cortafuegos y proxis sin ninguna modificación en el protocolo SOAP. SOAP a veces puede ser más lenta que las tecnologías de middleware como CORBA o ICE debido a su formato XML detallado.

A continuación, visualizaremos las capas de los servicios web SOAP.

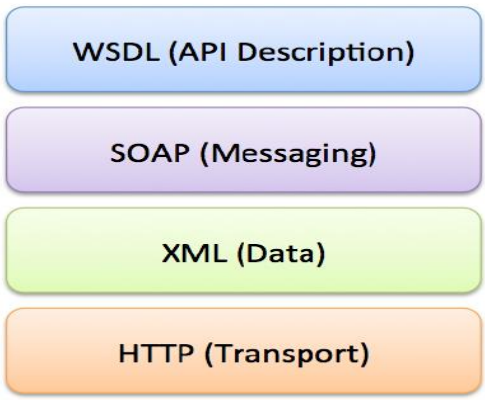


Figura 3-2: Servicio web SOAP
Realizado por: Tayupanda Luis, 2021

2.6 Servicios Web REST

Se define como una tecnología de servicios orientados a la manipulación de recursos de servidor por medio del protocolo HTTP. En un servicio REST típico se tiene que cada recurso expuesto se gestiona de acuerdo con el método utilizado en el protocolo HTTP (GET, POST, PUT, DELETE). (Cruz & Loaiza, 2017)

REST describe un conjunto de principios de la arquitectura por el cual los datos se pueden transmitir a través de una interfaz estandarizada (como HTTP). REST no contiene una capa adicional de mensajería y se centra en las reglas de diseño para la creación de servicios sin estado. Un cliente puede acceder al recurso mediante el único URI se devuelve y una representación del recurso. Con cada nuevo recurso de la representación, se dice que el cliente para transferir estado. Si bien el acceso a los recursos REST con el protocolo HTTP, la URL del recurso sirve como el identificador de recursos y GET, PUT, DELETE, POST y HEAD son las operaciones HTTP estándar que se deben realizar en ese recurso.

El Servicio Web REST, lucha constantemente con el proceso de autenticación de los usuarios. Sin embargo, hay problemas de autenticación debido a que REST no utiliza sesiones entre el servidor y el cliente. Por lo tanto, el proceso de autenticación del cliente es necesario para REST cuando los servidores reciben las solicitudes de los clientes (Villa, 2019)
A continuación, visualizaremos las capas de los servicios web REST.

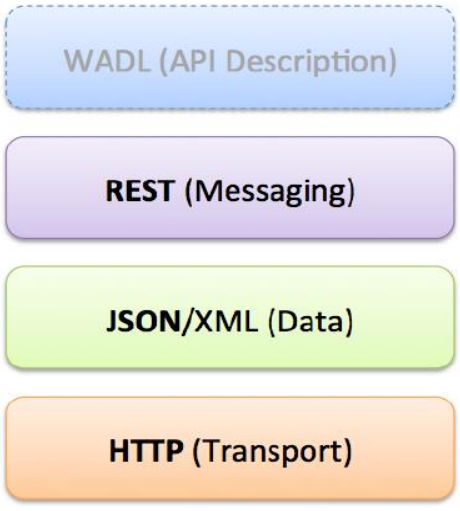


Figura 4-2: Servicios web REST
Realizado por: Tayupanda Luis, 2021.

2.7 Protocolo HTTP

Hyper Text Transfer Protocol es el protocolo que emplea WWW. Define como se tiene que crear y enviar los mensajes y que acciones se debe de tomar el servidor y el navegador en respuesta a un comando. Es un protocolo sin estatless (sin estado), porque cada comando se ejecuta independiente mente de los anteriores y posteriores, actualmente la mayoría de los servidores soportan HTTP. (Lujan Mora, 2002)

2.8 Protocolo HTTPS

El protocolo de transferencia de Hiper-Texto (HTPS) es una versión segura del http, este protocolo nos permite hacer transacciones de forma segura, de esta manera codifica la sesión con un certificado digital. (Rivera, 2011)

2.9 SSL

Es la capa de conexión segura que sirve como un estándar de seguridad global que permite la transferencia de datos cifrados entre un navegador y un servidor web. Es utilizado por millones de empresas e individuos en línea a fin de disminuir el riesgo de robo y manipulación de información confidencial (como números de tarjetas de crédito, nombres de usuario, contraseñas, correos electrónicos, etc.) por parte de hackers y ladrones de identidades. Básicamente, la capa SSL permite que dos partes tengan una "conversación" privada.

Para establecer esta conexión segura, se instala en un servidor web un certificado SSL (también llamado "certificado digital") que cumple dos funciones:

- ✓ Autenticar la identidad del sitio web, garantizando a los visitantes que no están en un sitio falso.
- ✓ Cifrar la información transmitida. (VERISIGN, 2018)

2.10 Arquitectura Orientada a Servicios SOA

Es un marco de trabajo conceptual que establece una estructura de diseño para la integración de aplicaciones, que permite a las organizaciones unir los objetivos de negocio, en cuanto a flexibilidad de integración con sistemas legados y alineación directa a los procesos de negocio, con la infraestructura de TI.

Esto permite la reducción de costos de implementación, innovación de servicios a clientes, adaptación ágil ante cambios y reacción temprana ante la competitividad, ya que, combinan fácilmente las nuevas tecnologías con aplicaciones independientes, permitiendo que los componentes del proceso se integren y coordinen de manera efectiva y rápida. (Sandobal, 2009)

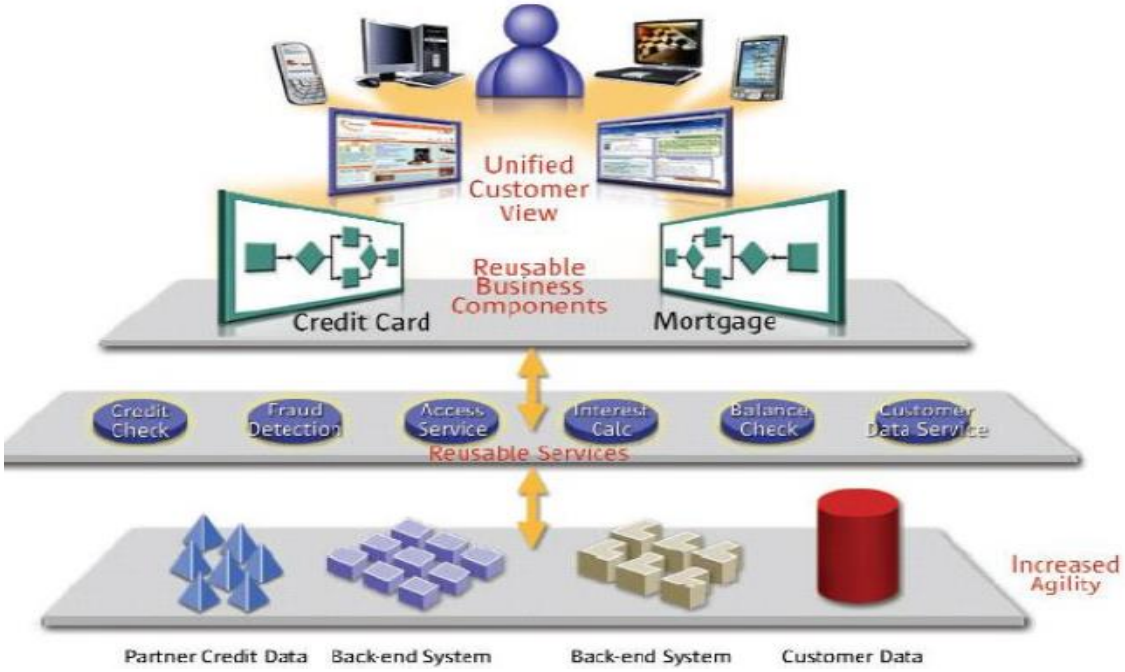


Figura 5-2: Arquitectura orientada a servicios
Fuente: (Aransay, 2009).

2.11 Autenticación

Se conoce como autenticación al proceso de confirmación que un remitente es quien dice ser a la hora de intentar acceder a un sistema o servicios web. En este proceso interviene dos partes.

- ✓ **Remitente:** Se refiere al usuario, servicio web o máquina que quiere conectarse a un servicio o sistema que requiere identificación.
- ✓ **Verificador:** Es la parte que verifica la identidad digital del remitente que quiere conectarse a los servicios web o sistema.

2.12 Autorización

Se conoce como autorización al proceso por el cual los servicios web o sistemas informáticos autorizan al usuario autenticado a acceder a ciertos recursos protegidos sobre los cuales se le haya concedido autorización previa. Estos recursos pueden ser ficheros, datos, dispositivos, funciones u otros tipos de información que ofrece los servicios web. La autorización debe asegurar la integridad y confidencialidad de los datos ofreciendo o denegando el acceso de lectura, creación, modificación, o borrado. (Méndez & Garcia, 2019)

CAPITULO III

3. METODOLOGÍA DE LA INVESTIGACIÓN

3.1 Tipo y Diseño de la Investigación

Por la naturaleza investigativa y experimental del proceso se define a este estudio como un compendio de diferentes estándares, métodos y técnicas a través de las cuales se conseguirán tanto las bases teóricas y fundamentales, así como las métricas de resultados.

Dentro de los estudios a utilizar tenemos:

- ✓ **Estudios exploratorios**

"Los estudios exploratorios sirven para familiarizarse con fenómenos relativamente desconocidos, obtener información sobre la posibilidad de llevar a cabo una investigación más completa sobre un contexto particular, investigar problemas de comportamiento humano que consideren cruciales los profesionales de determinada área, identificar conceptos o variables promisorias, establecer prioridad para investigaciones futuras, o sugerir afirmaciones o postulados." (Sampieri, 2007) Este tipo de estudio será fundamental en las fases iniciales de la investigación, específicamente en el proceso de recolección de la información y decisión.

- ✓ **Estudios descriptivos**

"Así como los estudios exploratorios se interesan fundamentalmente en descubrir y prefigurar, los descriptivos se centran en recolectar datos que muestren un evento, una comunidad, un fenómeno, hecho, contexto o situación que ocurre (para los investigadores cuantitativos medir con la mayor precisión posible)." (Sampieri, 2007) Este estudio se utilizará específicamente en la fase de medición de características y resultados para la definición del prototipo.

- ✓ **Estudio Experimental**

Se basa en pruebas realizadas en escenarios de laboratorio, en las que se observa los elementos más importantes del objeto de estudio que se investiga para obtener una captación de los fenómenos a primera vista.

3.2. Métodos de Investigación

Los métodos de investigación científica a utilizar siguen los siguientes pasos:

- ✓ Consulta en base a documentos (Registros, Internet, bibliografía científica, investigaciones realizadas en el país y estadísticas oficiales).
- ✓ Experimentación: Se recrearán distintas circunstancias en un ambiente controlado para la ejecución de pruebas, las cuales proveerán los resultados para la toma de decisiones y la definición del prototipo.

- ✓ Análisis de la información.
- ✓ Observación de campo: se harán distintas mediciones bajo métricas para la toma de decisiones.

3.2.1 El método hipotético – deductivo

Se emplea para la presente investigación puesto que, a partir de lo observado en diferentes experimentos, se formulan las correspondientes hipótesis, posteriormente aplicaríamos algunos conocimientos previos acerca del tema para obtener conclusiones que serán verificadas mediante la experiencia.

3.2.2 Método de Análisis y Síntesis

Éste método será utilizado para la revisión de la seguridad de acceso a los servicios web del sistema académico, además para la toma de decisiones, así como la obtención de información sobre los datos medidos.

3.3 Fuentes de Información

Dentro de las fuentes de recopilación de la información utilizadas en la presente investigación se mencionan:

3.3.1 *Primaria*

- ✓ Pruebas
- ✓ Observación de Resultado

3.3.2 *Secundaria*

- ✓ Artículos publicados en revistas científicas.
- ✓ Trabajos de investigación publicados a nivel nacional e internacional con temas afines al investigado.
- ✓ Páginas de internet que brinden información confiable y especializada.
- ✓ Libros especializados en la biblioteca y electrónicos.
- ✓ Revistas electrónicas.

3.4 Planteamiento de la Hipótesis

3.4.1 Hipótesis General

Al implementar una interfaz Web con el estándar de seguridad Json Web Token garantiza el acceso y autorización segura a los servicios web del sistema académico de la Escuela Superior Politécnica de Chimborazo.

3.4.2 Identificación de variables

Variables Independientes

Implementación de los servicios web de tipo REST con el estándar de seguridad Json Web Token para el acceso seguro a los servicios del sistema académico de la Escuela Superior Politécnica de Chimborazo.

Variables Dependiente

Seguridad y rendimiento en la transferencia de estado representacional de los servicios webde tipo REST del sistema académico de la Escuela Superior Politécnica de Chimborazo.

3.4.5 Operacionalización de variables

Tabla 1-3: Operacionalización conceptual de variables

VARIABLE	**TIPO**	**CONCEPTO**
Implementación de los servicios web de tipo REST con el estándar de seguridad Json Web Token para el acceso seguro a los servicios del sistema académico de la Escuela Superior Politécnica de Chimborazo.	Independiente	Estándar de acceso seguro a los servicios web.
Seguridad y rendimiento en la transferencia de estado representacional de los servicios web del sistema académico de la Escuela	Dependiente	Métricas esenciales en cualquier red para la garantía de la integridad de los datos y la optimización de recursos.

Superior Politécnica de Chimborazo.		

Realizado por. Tayupanda Luis, 2021.

Tabla 2-3: Operacionalización metodológica de variables

VARIABLES	INDICADORES	TECNICAS	INSTRUMENTOS
V.I. Implementación de los servicios web de tipo REST con el estándar de seguridad Json Web Token para el acceso seguro a los servicios del sistema académico de la Escuela Superior Politécnica de Chimborazo.	Frecuencia de peticiones solicitadas y atendidos	Observación	Ambiente de pruebas
V.D. Seguridad y rendimiento en la transferencia de estado representacional de los servicios web del sistema académico de la Escuela Superior Politécnica de Chimborazo.	Tiempo de respuesta invertido en el manejo de peticiones	Observación	Ambiente de pruebas

Realizado por. Tayupanda Luis, 2021.

3.5 Enfoque de la Investigación

EL presente estudio por su concepción es de tipo cualitativo.

3.6 Alcance Investigativo

El alcance de la presente investigación y propuesta es de tipo descriptivo.

3.7 Población de Estudio

La población de la investigación son todas las vulnerabilidades que ponen en riesgo a los servicios Web REST del sistema académico dentro del ambiente de pruebas.

3.8 Unidad de Análisis

En la Dirección de Tecnologías de la investigación y comunicación de la Escuela Superior Politécnica de Chimborazo.

3.9 Selección de la Muestra

Para la selección de la muestra se consideró los aspectos definidos para la seguridad de transferencia de estados o peticiones a los servicios web de tipo REST de OWASP, tomados de forma no probabilística, partiendo desde la más transcendental en cuanto a seguridad y que ayudaron a determinar un método óptimo en el desarrollo de los servicios web de tipo REST y que finalmente fueron probados al aplicar la propuesta en uno de los ambientes implementados como se define en la (**Tabla 3-3)**

Tabla 3-1: Parámetros a Evaluar

Nº	**Parámetro de seguridad**
1	Access Control
2	Uso HTTPS
3	Códigos de Error
4	JWS
5	Auditoría de logs
6	CORS
7	Restricciones de Métodos
8	Validación de Tipos de Contenido y Entradas
9	Documentación

Realizado por. Tayupanda Luis, 2021.

3.10 Instrumentos de Colección de Datos

Para la recolección de los datos se realizará la implementación de los escenarios de pruebas, las herramientas utilizadas son de código abierto y licenciado para su debida implementación y además permiten utilizar sus características principales sin ningún tipo de limitación.

Para implementar el token de seguridad se utilizó el IDE de desarrollo Visual Studio con unas claves privadas y claves públicas.

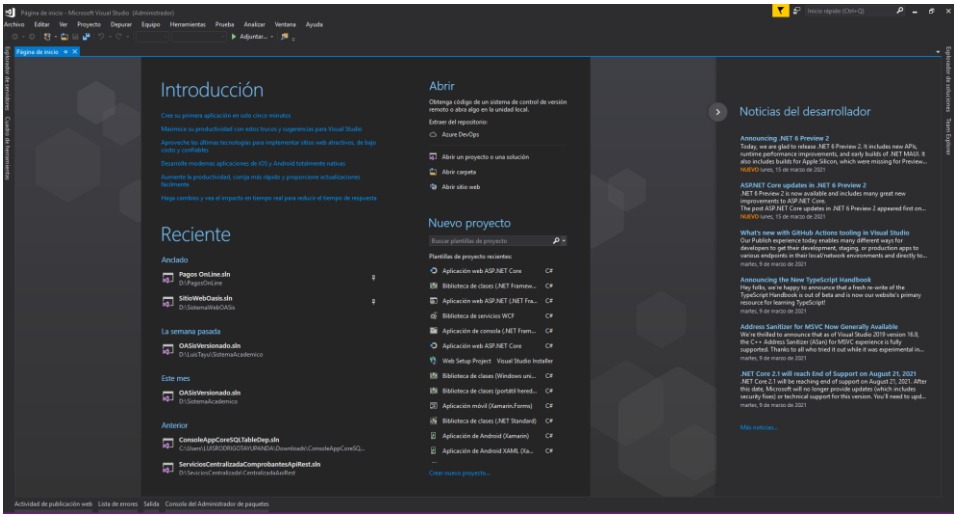


Figura 1-3: IDE de desarrollo Visual Studio
Fuente: Visual Studio, 2021.

Para realizar el testing del consumo de los servicios web se utilizó la herramienta Posman, que admite varias funcionalidades de testing de los servicios de tipo REST.

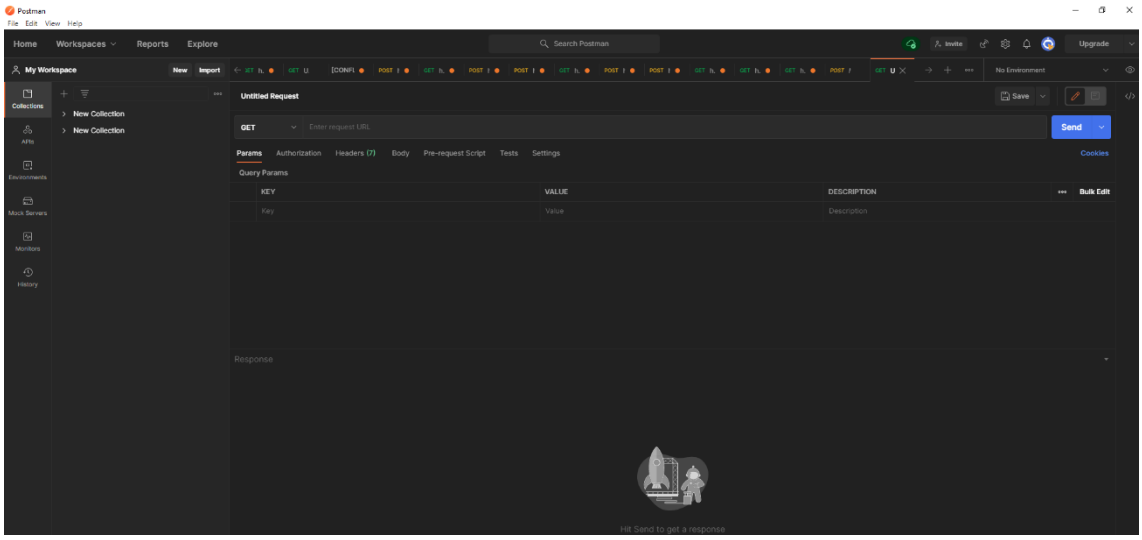


Figura 2-3: Testing Posman
Fuente: Postman, 2021.

Para alojar los servicios web se utilizó el servidor IIS, que nos permite utilizar un conjunto de servicios para brindar información tanto FTP y SMTP entre otro.

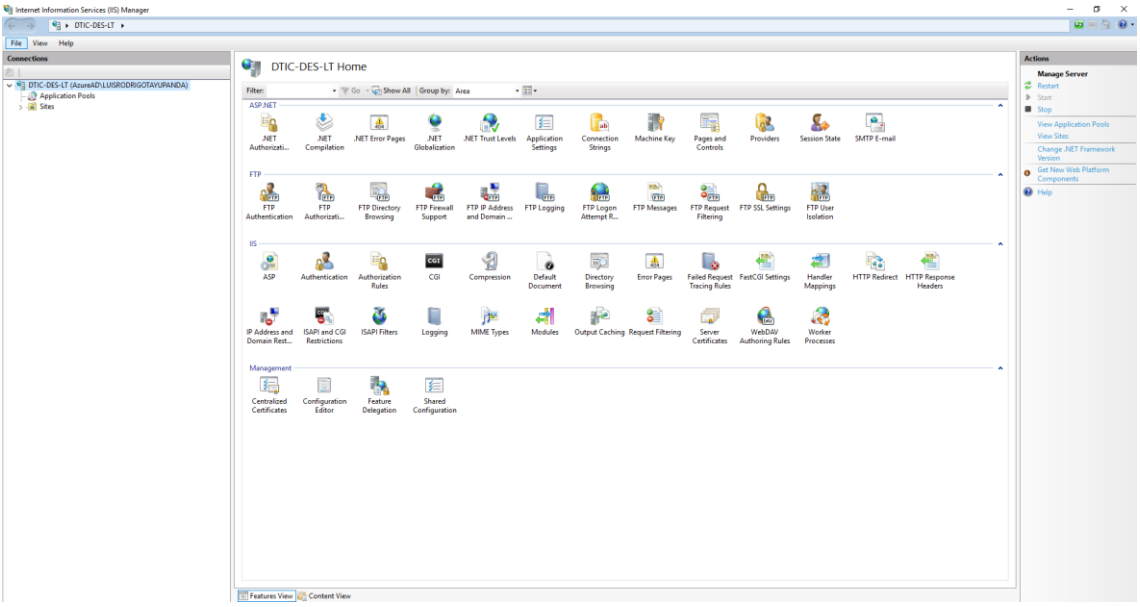


Figura 3-3: Servidor IIS
Fuente: Internet Information Services, 2021.

Para el almacenamiento y gestión de la información se utiliza el motor de base de datos SQL Server

Figura 4-3: Motor de Base de datos
Fuente: (SQL server, 2021).

Para implementar la funcionalidad del token en los servicios web de tipo REST del sistema académico, se valoró de acuerdo al cumplimiento de las características que posee cada uno de ellas.

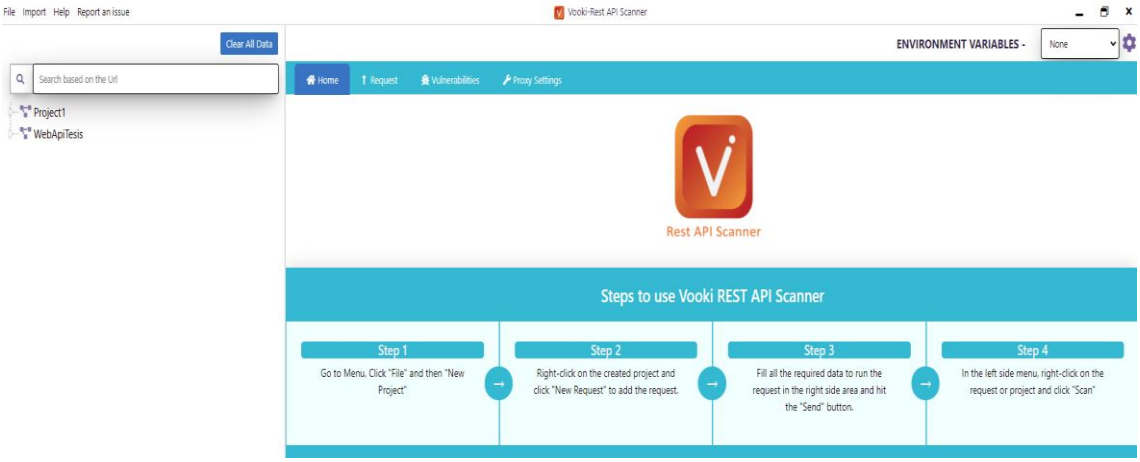


Figura 5-3: Scanner de Vulnerabilidades
Fuente: (Vooki, 2021).

Para el escaneo de las vulnerabilidades de los servicios web REST, se utilizó Vooki que permite scanear y dar un reporte de vulnerabilidades.

3.11 ESCENARIO DE PRUEBAS

3.12.1 Descripción del escenario

Los escenarios de pruebas consisten en la simulación de transferencia de estados o peticiones HTTP entre el cliente y el servidor.

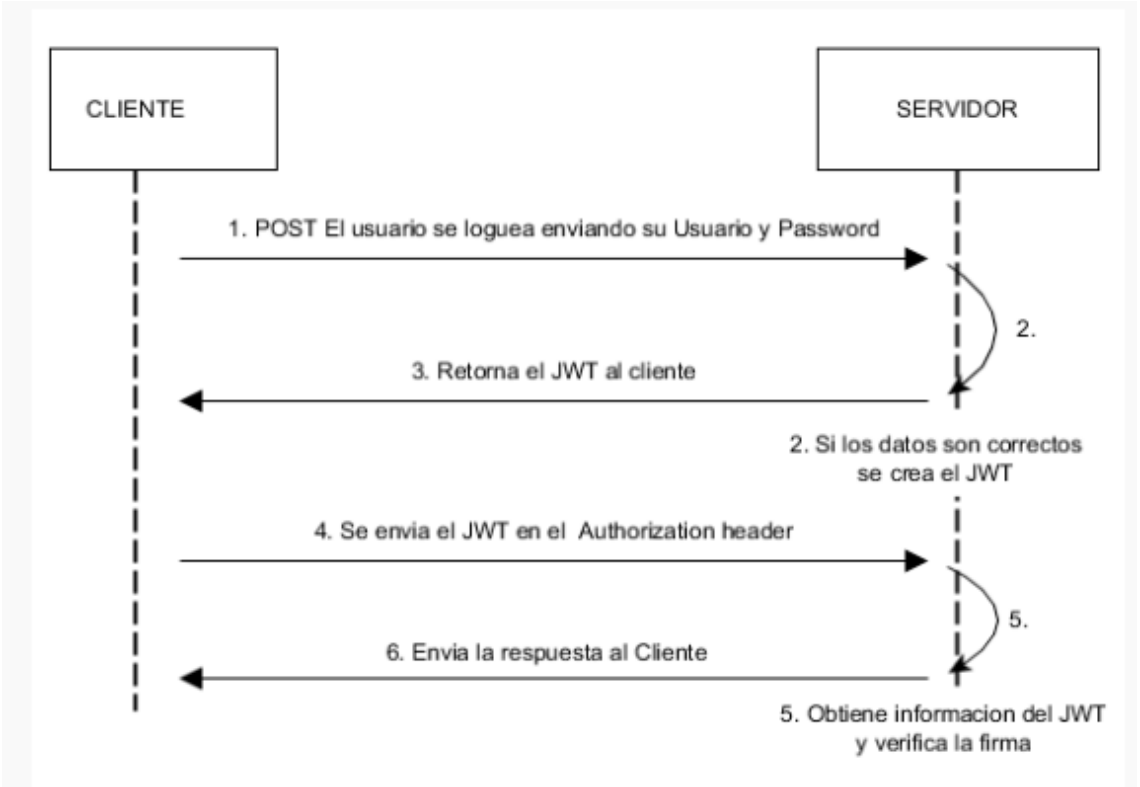


Figura 6-3: Diseño de transferencia de estados con JWT
Realizado por: Tayupanda Luis, 2021

Para la presente investigación se realizó dos escenarios, el primer escenario no se utilizó el estándar de seguridad Json Web Token, y en el segundo escenario si se utilizó el estándar de seguridad Json Web Token.

Los servicios web REST implementados en cada uno de los escenarios contemplan la lógica de transferencia de estados o peticiones HTTP.

Escenario 1

Para el escenario 1 se desarrolló los servicios web de tipo REST, sin utilizar el estándar de seguridad Json Web token en los servicios del sistema académico de la Escuela Superior Politécnica de Chimborazo con los métodos tradicionales.

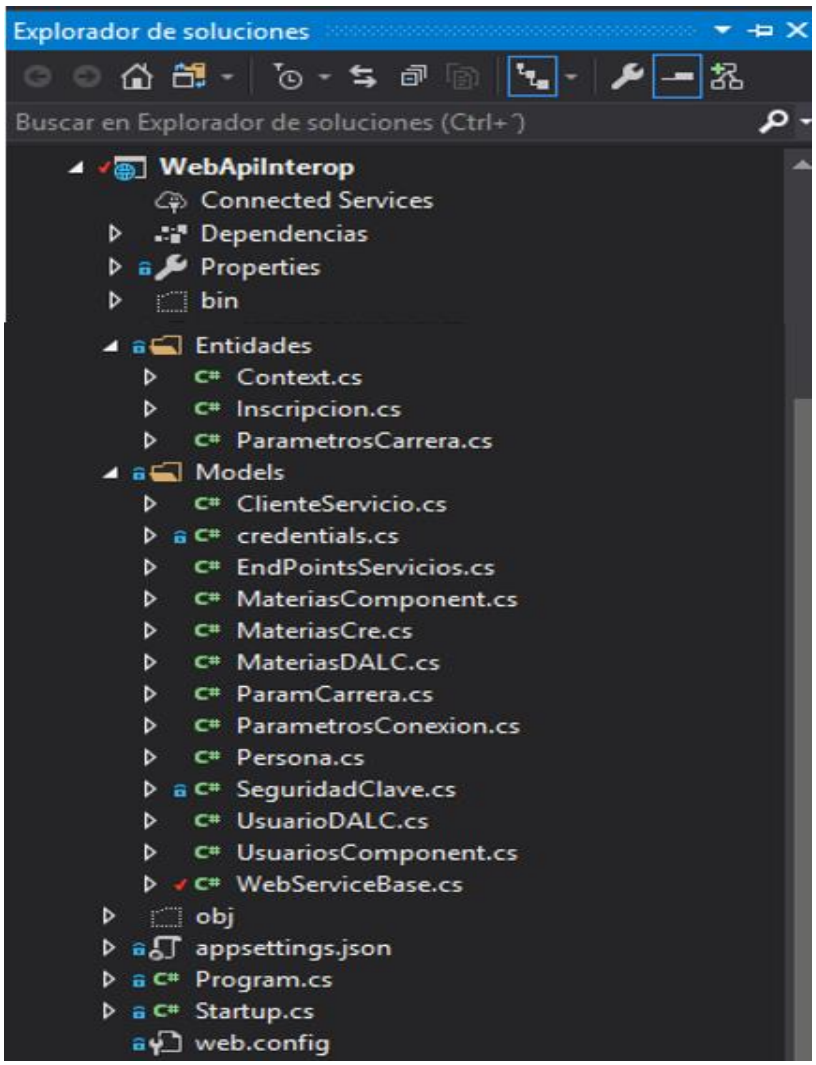


Figura 7-3: Estructura del código de los servicios web del escenario 1
Realizado por: Tayupanda Luis ,2021

En este escenario 1 no se utilizó ninguna autenticación para poder acceder a los métodos, para realizar el consumo de la información, se utilizó la herramienta de testing Posman que nos ayuda a visualizar la información en un formato Json, previa mente al tener el path o la url del servicio.

Escenario 2

Para el escenario 2 se desarrolló los servicios web de tipo REST, del sistema académico de la Escuela Superior Politécnica de Chimborazo con una autenticación, y con el estándar de seguridad Json Web Token, para su validación y verificación del token para cada petición o transferencia de estados entre el cliente y el servidor.

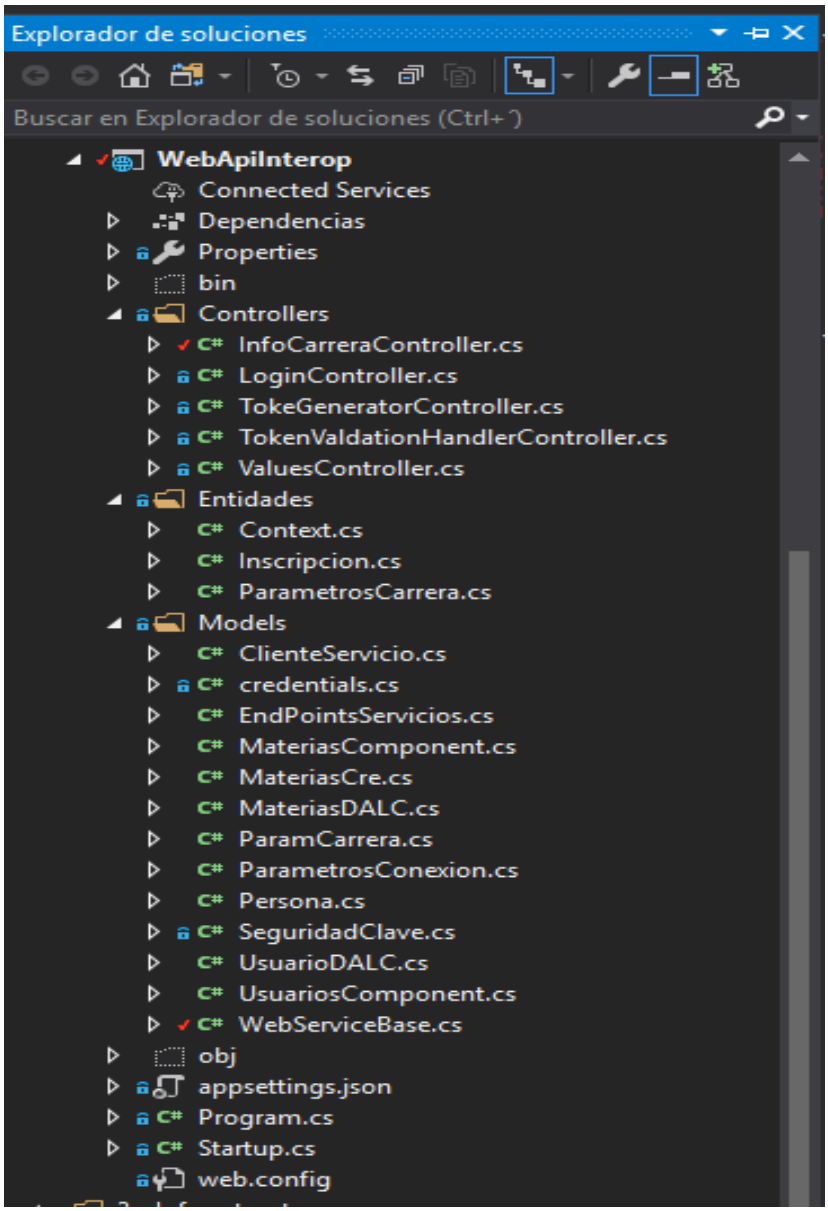


Figura 8-3: Estructura del código de los servicios web del escenario 2
Realizado por: Tayupanda Luis, 2021

Para este escenario 2 se desarrolló una autenticación, en el cual el servidor verificaba y validaba la existencia del usuario y la contraseña, si todo estaba correcto el servidor generaba un token de seguridad con la identidad del usuario, el token de seguridad Json Web Token se genera codificado en un objeto JSON que esta incrustado dentro del payload o cuerpo de un mensaje que va firmado digitalmente.

El token de seguridad se trata de una cadena de texto que tiene tres partes codificadas en Base 64, cada una de ellas separadas por un punto.

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9.TJVA95OrM7E2cBab30RMHrHDcEfxjoYZgeFONFh7HgQ

Figura 9-3: Json Web Token
Realizado por: Tayupanda Luis, 2021

Para realizar las peticiones a los métodos generados, en cada transferencia de estado se debe envía el token de seguridad Json Web Token para que sea verificado y validado su autenticación.

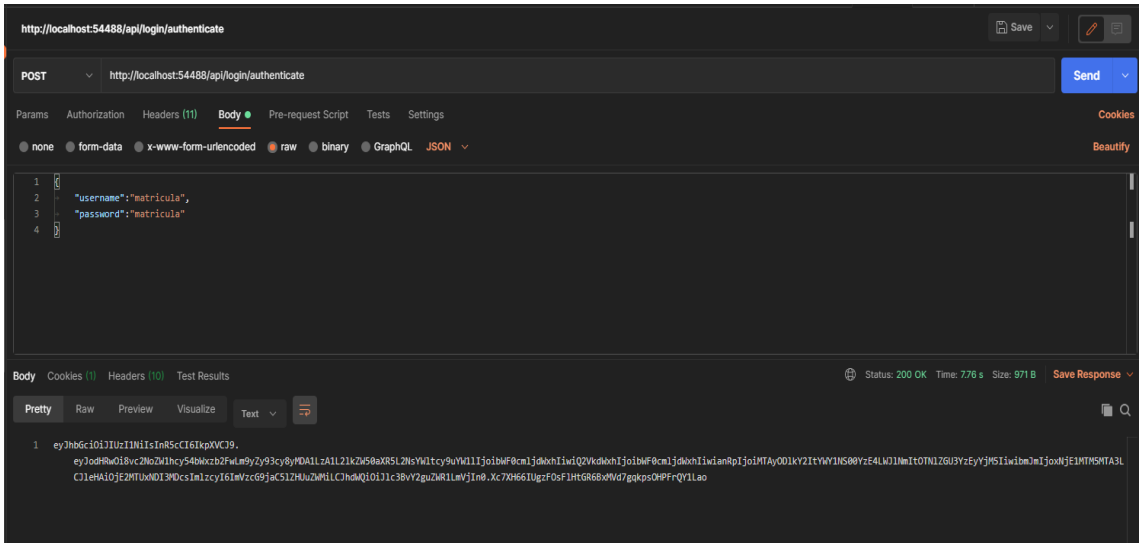


Figura 10-3: Json Web Token Generado
Realizado por: Tayupanda Luis, 2021

3.12.2 Resultados

Para conseguir los resultados para esta investigación, se establecieron las siguientes pruebas para cado uno de los escenarios: Una vez desarrollados los escenarios, se procedió a realizar la verificación del cumplimiento de los parámetros definidos en la (**Tabla 3-3**) usando la herramienta POSTMAN, ejecutando desde el cliente las transferencias de estado HTTP, y en base a las respuestas que se obtuvieron con la herramienta de pentesting se registraron los datos en una tabla comparativa de los dos escenarios. Para la obtención de los datos para verificar el nivel de

seguridad se utilizó la herramienta VOOKI que permitió realizar el pentesting y obtener las vulnerabilidades en cada uno de los escenarios. Posterior a la aplicación de las pruebas definidas para los dos escenarios, se recolectaron los datos numéricos necesarios para aplicar la distribución, con el objetivo de demostrar que el método propuesto permite tener el acceso seguro hacia los servicios web de tipo REST.

CAPITULO IV

4. RESULTADOS Y DISCUCIÓN

En este capítulo se analizará los datos obtenidos en los distintos escenarios de pruebas elaboradas y se contrastarán sus resultados, además se realizarán la comparación de la hipótesis definida para la presente investigación.

4.1 DESARROLLO DE PRUEBAS

Los resultados han sido divididos en dos grupos, en primer lugar, se presentan aquellos resultados relacionados exclusivamente sin el uso del estándar de seguridad Json Web Token, luego se dan a conocer aquellos resultados implementado con el estándar de seguridad Json Web Token para que sea más seguro los servicios para las vulnerabilidades.

4.1.1 Escenario de prueba sin el estándar de seguridad

El primer escenario de pruebas consiste en la implementación del servicio web REST, sin el estándar de seguridad a continuación se presente el diagrama lógico.

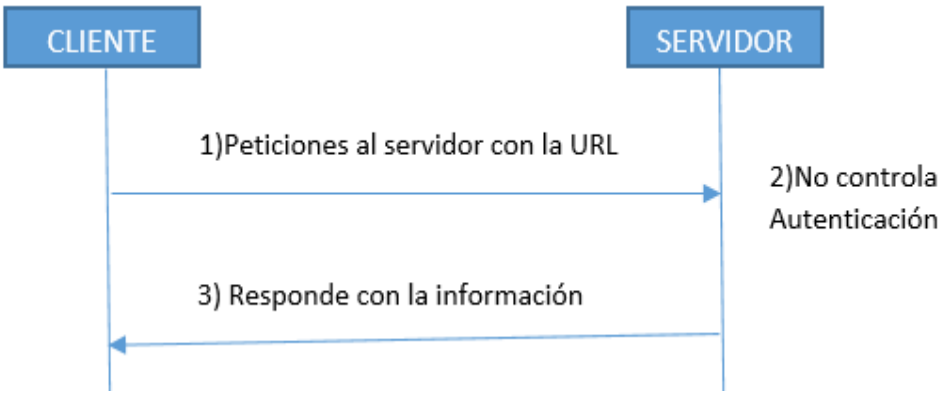


Figura 11-4: Diseño del primer escenario de pruebas
Realizado por: Tayupanda Luis, 2021.

El primer escenario de pruebas se realiza peticiones o transferencia de estado de un servicio web REST, sin tener el token de seguridad, con una autenticación básica para las diferentes peticiones, en el cual se puede acceder a todos los métodos del servicio esto a su vez viene hacer muy vulnerable al no tener ningún tipo de seguridad.

4.1.2 Escenario de prueba con el estándar de seguridad

El segundo escenario de prueba consiste en la implementación del estándar de seguridad acceso Json Web Token.

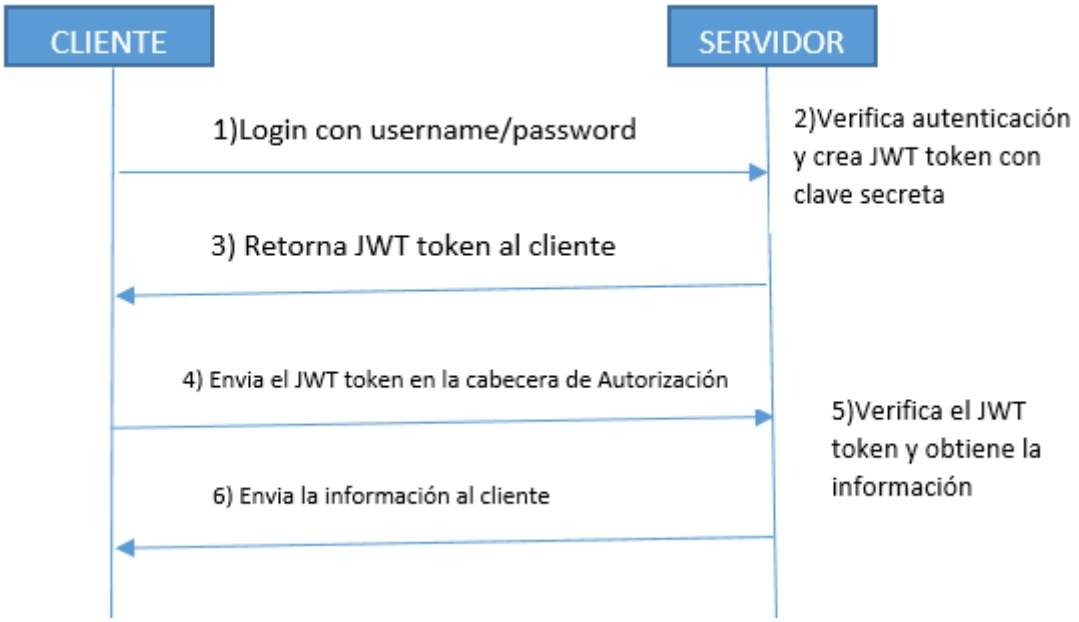


Figura 12-4: Diseño del primer escenario de pruebas
Realizado por: Tayupanda Luis, 2021

El segundo escenario de prueba consiste en la autenticación y creación del token de seguridad para las peticiones HTTP que consiste en el uso de los métodos GET, POST, PUT, DELETE. En este escenario consiste por cada petición se debe enviar el token de seguridad para que sea validada y verificada en el servidor, de esta manera garantice el acceso a la información correspondiente de acuerdo a los privilegios de acceso.

4.2 Análisis e interpretación de resultados

Para obtener los resultados de los dos escenarios para la investigación, se verifico el cumplimiento de la aplicabilidad en cada uno de los parámetros, utilizando la herramienta POSTMAN, ejecutando la peticiones o transferencias de estado hacia el servidor.

Para cada uno de los escenarios se realizó varias pruebas de pentesting para observar las vulnerabilidades de cada uno de los escenarios correspondientes.

Para la obtención de los datos para verificar el nivel de seguridad se utilizó la herramienta VOOKI que permitió realizar el pentesting y obtener las vulnerabilidades de cada uno de los escenarios.

Luego de haber ejecutado varias pruebas definidas para los dos escenarios, se recolectaron los datos numéricos necesarios para aplicar la distribución, con el objetivo de demostrar que el estándar de seguridad Json Web Token permite incrementar la seguridad de los servicios web de tipo REST.

4.3. Valoración de la Variable Independiente

4.3.1. Variable Independiente:

El estándar de seguridad Json Web Token para el acceso seguro a los servicios del sistema académico de la Escuela Superior Politécnica de Chimborazo, se procedió a realizar una observación y así validar el cumplimiento del uso de los parámetros planteados para la evaluación en cada escenario.

4.3.2. Indicador

4.3.2.1 Nivel de satisfacción en la aplicación del método: Para la medición de este indicador se utilizó la escala de Likert (Netquest, 2014) que permitió tener una valoración del nivel de satisfacción de la aplicación del estándar de seguridad Json Web Token (**Tabla 1-4**).

Tabla 1-4: Escala de Linker

Escala	**Siempre**	**Casi Siempre**	**Alguna Veces**	**Muy poca veces**	**Nunca**
Valoración	5	4	3	2	1

Realizado por: Tayupanda Luis, 2021

En base a la observación realizada sobre los escenarios definidos, la utilización de la herramienta POSTMAN hemos obtenido los siguientes resultados.

Tabla 2-4: Parámetros evaluados de los escenarios

N°	**Parámetro**	**Escenario 1 (Sin el estándar de seguridad)**	**Escenario 2 (Con el estándar de seguridad)**
1	Uso de certificado digital	5	5
2	Control de acceso a recursos	1	5
3	Uso de Json Web Token	1	5
4	Restricciones de métodos HTTP no disponibles	1	5
5	Uso de CORS	1	4
6	Presentación correcta de código de error	2	4
7	Validación de tipos de contenidos en sus entradas	2	4
8	Auditoria de registros de logs de acceso y error	2	5
	Total	**15**	**37**

Realizado por: Tayupanda Luis, 2021

En base a esta información el porcentaje total de aplicabilidad se calcula como valor de 40 que equivale al 100% esto indicaría que siempre se aplica los ocho (8) parámetros planteados a evaluar.

Los valores de aplicabilidad obtenidos en la observación corresponden al 37,5% en el Escenario 1, y el 92.5% en el Escenario 2, al aplicar el estándar propuesto se puede apreciar una mejora en el desarrollo seguro de estos servicios web de tipo REST.

4.3 Valoración de la variable Dependiente

Nivel de Seguridad. Para su valoración se utilizó la herramienta de pentesting Vooki aplicada a los dos escenarios implementados.

- ✓ **Identificación de vulnerabilidades sobre el escenario 1**

En este escenario no se utilizó el estándar de seguridad Json Web Token.

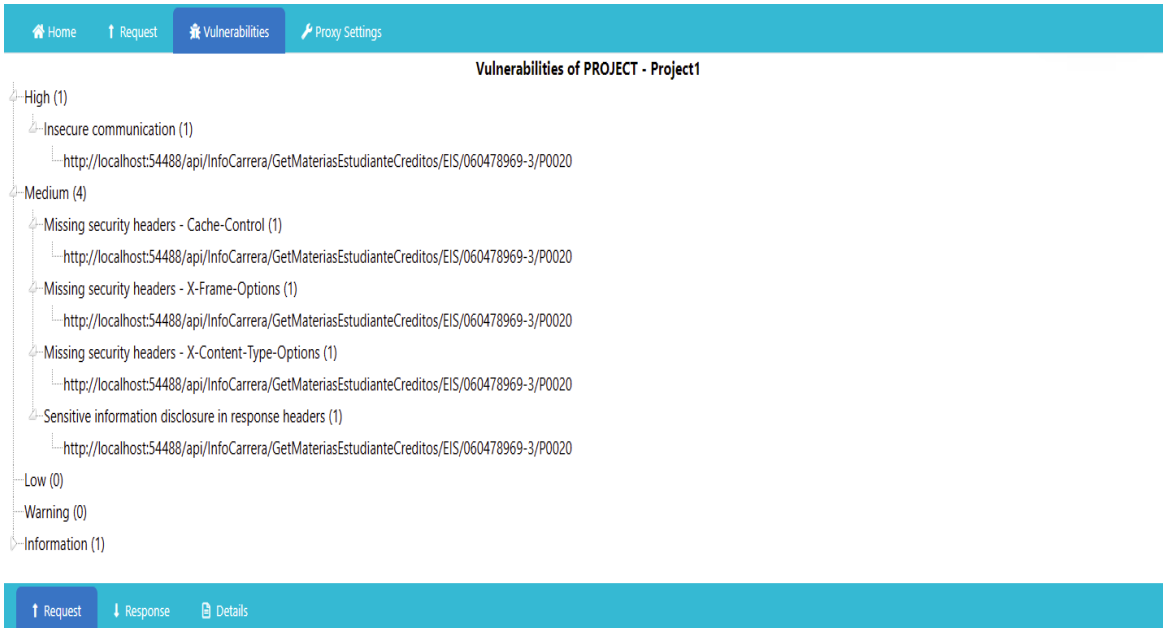


Figura 13-4: Captura de pentesting escenario 1
Realizado por: Tayupanda Luis, 2021

Al realizar el escaneo de las vulnerabilidades la herramienta vooki, detecto 6 tipos de vulnerabilidades que a continuación se detalla.

Rest Api Vulnerability Scanner

SUMMARY OF FINDINGS (Scanned Node: Project1)

Risk	Count
High	1
Medium	4
Low	0
Warning	0
Information	1
Total	6

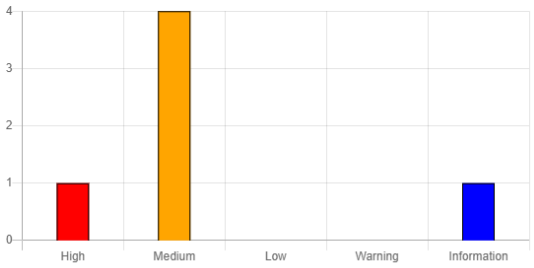


Figura 14-4: Captura de la clasificación de riesgo escenario 1
Realizado por: Tayupanda Luis, 2021

Como se puede observar en la imagen (**Figura4-4**) la cantidad de vulnerabilidades clasificados por los tipos de riesgos.

No	Vulnerability Name	Risk	Severity	Cvss score	Occurrences
1	Insecure communication	High	High	8.1	1
2	Sensitive information disclosure in response headers	Medium	Medium	5.0	1
3	Missing security headers - X-Content-Type-Options	Medium	Medium	5.0	1
4	Missing security headers - X-Frame-Options	Medium	Medium	5.0	1
5	Missing security headers - Cache-Control	Medium	Medium	5.0	1
6	Missing security headers - X-XSS-Protection	Information	Information		1

Figura 15-4: Captura del detalle del pentesting escenario 1
Realizado por: Tayupanda Luis, 2021

Como se puede observar en la imagen (**Figura5-4**) los valores de las métricas de vulnerabilidades son uno alto y cuatro medios, y uno informativo.

✓ **Identificación de vulnerabilidades sobre el escenario 2**

En este escenario se utilizó el uso del estándar de seguridad Json Web Token.

Figura 16-4: Captura de pentesting escenario 2
Realizado por: Tayupanda Luis, 2021

Al realizar el escaneo de las vulnerabilidades la herramienta vooki, detecto 1 tipos de vulnerabilidad que a continuación se detalla.

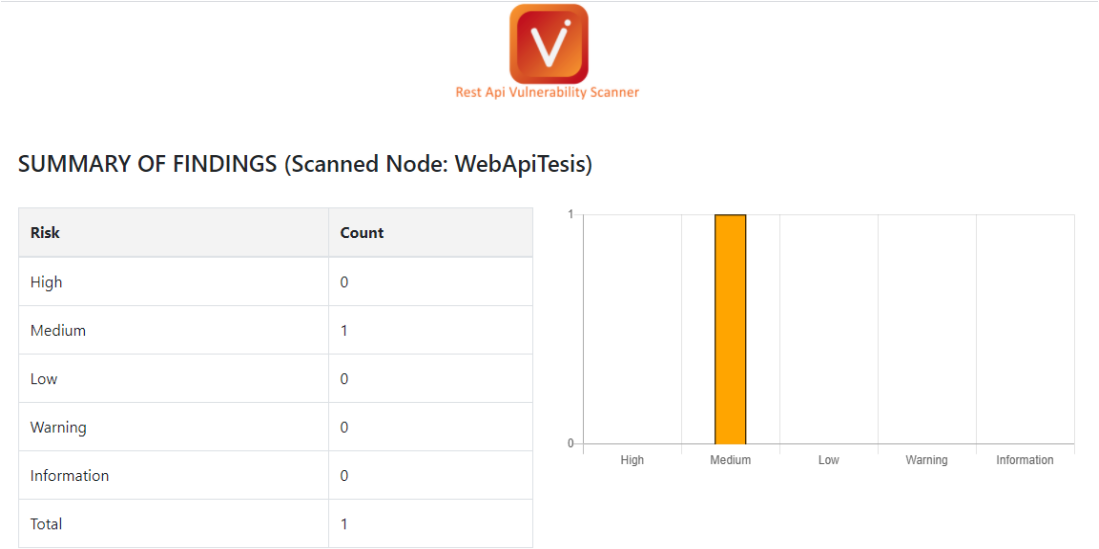


Figura 17-4: Captura de la clasificación de riesgo escenario 2
Realizado por: Tayupanda Luis, 2021

Como se puede observar en la imagen (**Figura 7-4**) la cantidad de vulnerabilidades clasificados por los tipos de riesgos.

No	Vulnerability Name	Risk	Severity	Cvss score	Occurrences
1	Sensitive information disclosure in response headers	Medium	Medium	5.0	1

Figura 18-4: Captura del detalle del pentesting escenario 2
Realizado por: Tayupanda Luis, 2021

Tabla 3-4: Resumen de resultados obtenidos en los pentesting

Vulnerabilidades Encontradas	**Frecuencia**		**Porcentajes de Reducción**
	Escenario 1 (Sin estándar de seguridad)	**Escenario 2 (Con estándar de seguridad)**	
Altas	1	0	100%
Medias	4	1	75%
Bajas	0	0	0%
Total	**5**	**1**	**80%**

Realizado por: Tayupanda Luis, 2021

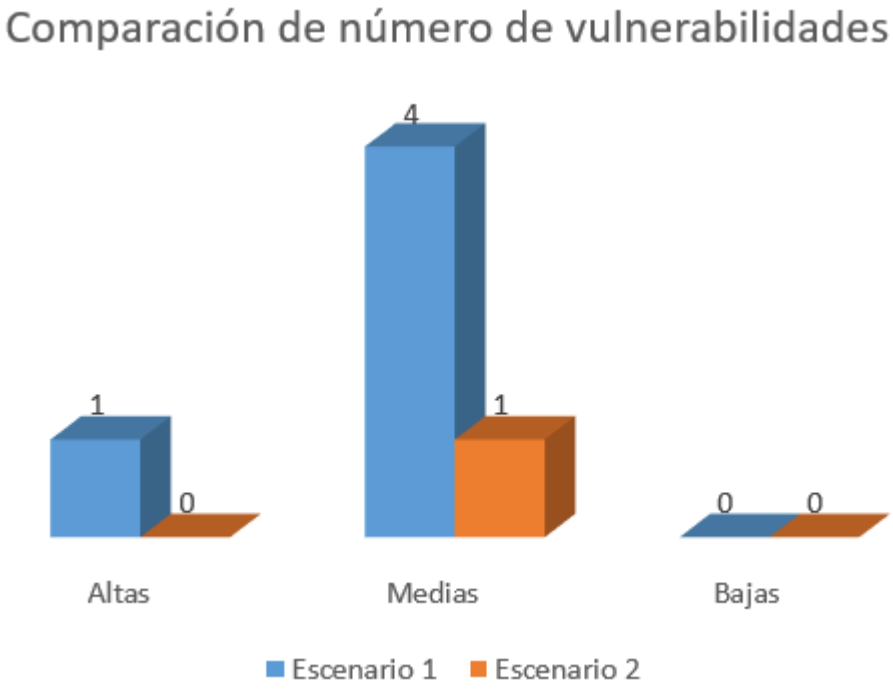


Figura 19-4: Comparación número de vulnerabilidades encontradas
Realizado por: Tayupanda Luis, 2021

Análisis e interpretación de Resultados

Para la reducción de vulnerabilidades en un servicio web de tipo REST, se debe utilizar el método de validación y verificación del estándar de seguridad Json Web Token, que ha reducido en un 100% de vulnerabilidades de Nivel Alto, en un 75% en un Nivel Medio, dando como resultado total un 80% de reducción con el estándar de seguridad desarrollado.

4.4 Comparación estadística de la hipótesis

Para la demostración de la hipótesis general "Al implementar una interfaz Web con el estándar de seguridad Json Web Token garantiza el acceso y autorización segura a los servicios web del sistema académico de la Escuela Superior Politécnica de Chimborazo", se utilizó estadística referencial aplicando la prueba Chi-Cuadrado(X^2).

Luego de realizar los diferentes análisis, y con la información obtenida se procede a definir la hipótesis de investigación Ha y la Hipótesis Nula H0 a ser consideradas.

Ho: "Al implementar una interfaz Web con el estándar de seguridad Json Web Token **no** garantiza el acceso y autorización segura a los servicios web del sistema académico de la Escuela Superior Politécnica de Chimborazo"

Ha: "Al implementar una interfaz Web con el estándar de seguridad Json Web Token **si** garantiza el acceso y autorización segura a los servicios web del sistema académico de la Escuela Superior Politécnica de Chimborazo"

Tabla 4-1: Frecuencia de valores encontrado

	Escenario 1 (Sin estándar de seguridad)	**Escenario 2 (Con estándar de seguridad)**	**Total**
Vulnerabilidades encontrada	5	1	6
Vulnerabilidades solventadas	0	4	4
Total	5	5	10

Realizado por: Tayupanda Luis, 2021

Para la obtención de la tabla de frecuencia esperada se aplica la siguiente formula de cada valor de tabla.

$$Fe = \frac{\text{total columna} * \text{total fila}}{\text{suma total}}$$

Tras la aplicación de la formula en cada valor de la tabla anterior obtendremos la siguiente tabla **5-4** de frecuencia esperadas.

Tabla 5-4: Frecuencia Esperada

	Escenario 1 (Sin estándar de seguridad)	**Escenario 2 (Con estándar de seguridad)**	**Total**
Vulnerabilidades encontrada	3	3	6
Vulnerabilidades solventadas	2	2	4
Total	5	5	10

Realizado por: Tayupanda Luis, 2021

A continuación, se calcula el valor de X2 mediante la siguiente fórmula

$$X^2 = \sum \frac{(FO - FE)^2}{FE}$$

Donde:

FO: Frecuencia Observada por celda

FE: Frecuencia Esperada por celda

$$X^2 = \frac{(5-3)^2}{3} + \frac{(0-2)^2}{2} + \frac{(1-3)^2}{3} + \frac{(4-2)^2}{2}$$

X^2=1.33+2+1.33+2

X^2 =6.66

El siguiente paso a seguir es el cálculo de los grados de libertad

$$v = (r-1)x(k-1)$$

Donde:

r: número de filas

k: número de Columnas

$$v = (2-1)x(2-1)$$

v=1

En base a la tabla de la distribución de Chi-Cuadrado (Figura 5-4), y determinando el valor de significancia de 0.05% obtenemos el punto crítico con 1 como valor de grados de libertad.

TABLA 3-Distribución Chi Cuadrado χ^2

P = Probabilidad de encontrar un valor mayor o igual que el chi cuadrado tabulado, ν = Grados de Libertad

ν/p	0,001	0,0025	0,005	0,01	0,025	0,05	0,1	0,15	0,2	0,25	0,3	0,35	0,4	0,45	0,5
1	10,8274	9,1404	7,8794	6,6349	5,0239	3,8415	2,7055	2,0722	1,6424	1,3233	1,0742	0,8735	0,7083	0,5707	0,4549
2	13,8150	11,9827	10,5965	9,2104	7,3778	5,9915	4,6052	3,7942	3,2189	2,7726	2,4079	2,0996	1,8326	1,5970	1,3863
3	16,2660	14,3202	12,8381	11,3449	9,3484	7,8147	6,2514	5,3170	4,6416	4,1083	3,6649	3,2831	2,9462	2,6430	2,3660
4	18,4662	16,4238	14,8602	13,2767	11,1433	9,4877	7,7794	6,7449	5,9886	5,3853	4,8784	4,4377	4,0446	3,6871	3,3567
5	20,5147	18,3854	16,7496	15,0863	12,8325	11,0705	9,2363	8,1152	7,2893	6,6257	6,0644	5,5731	5,1319	4,7278	4,3515
6	22,4575	20,2491	18,5475	16,8119	14,4494	12,5916	10,6446	9,4461	8,5581	7,8408	7,2311	6,6948	6,2108	5,7652	5,3481
7	24,3213	22,0402	20,2777	18,4753	16,0128	14,0671	12,0170	10,7479	9,8032	9,0371	8,3834	7,8061	7,2832	6,8000	6,3458
8	26,1239	23,7742	21,9549	20,0902	17,5345	15,5073	13,3616	12,0271	11,0301	10,2189	9,5245	8,9094	8,3505	7,8325	7,3441
9	27,8767	25,4625	23,5893	21,6660	19,0228	16,9190	14,6837	13,2880	12,2421	11,3887	10,6564	10,0060	9,4136	8,8632	8,3428
10	29,5879	27,1119	25,1881	23,2093	20,4832	18,3070	15,9872	14,5339	13,4420	12,5489	11,7807	11,0971	10,4732	9,8922	9,3418
11	31,2635	28,7291	26,7569	24,7250	21,9200	19,6752	17,2750	15,7671	14,6314	13,7007	12,8987	12,1836	11,5298	10,9199	10,3410
12	32,9092	30,3182	28,2997	26,2170	23,3367	21,0261	18,5493	16,9893	15,8120	14,8454	14,0111	13,2661	12,5838	11,9463	11,3403
13	34,5274	31,8830	29,8193	27,6882	24,7356	22,3620	19,8119	18,2020	16,9848	15,9839	15,1187	14,3451	13,6356	12,9717	12,3398
14	36,1239	33,4262	31,3194	29,1412	26,1189	23,6848	21,0641	19,4062	18,1508	17,1169	16,2221	15,4209	14,6853	13,9961	13,3393
15	37,6978	34,9494	32,8015	30,5780	27,4884	24,9958	22,3071	20,6030	19,3107	18,2451	17,3217	16,4940	15,7332	15,0197	14,3389
16	39,2518	36,4555	34,2671	31,9999	28,8453	26,2962	23,5418	21,7931	20,4651	19,3689	18,4179	17,5646	16,7795	16,0425	15,3385
17	40,7911	37,9462	35,7184	33,4087	30,1910	27,5871	24,7690	22,9770	21,6146	20,4887	19,5110	18,6330	17,8244	17,0646	16,3382
18	42,3119	39,4220	37,1564	34,8052	31,5264	28,8693	25,9894	24,1555	22,7595	21,6049	20,6014	19,6993	18,8679	18,0860	17,3379
19	43,8194	40,8847	38,5821	36,1908	32,8523	30,1435	27,2036	25,3289	23,9004	22,7178	21,6891	20,7638	19,9102	19,1069	18,3376
20	45,3142	42,3358	39,9969	37,5663	34,1696	31,4104	28,4120	26,4976	25,0375	23,8277	22,7745	21,8265	20,9514	20,1272	19,3374
21	46,7963	43,7749	41,4009	38,9322	35,4789	32,6706	29,6151	27,6620	26,1711	24,9348	23,8578	22,8876	21,9915	21,1470	20,3372
22	48,2676	45,2041	42,7957	40,2894	36,7807	33,9245	30,8133	28,8224	27,3015	26,0393	24,9390	23,9473	23,0307	22,1663	21,3370
23	49,7276	46,6231	44,1814	41,6383	38,0756	35,1725	32,0069	29,9792	28,4288	27,1413	26,0184	25,0055	24,0689	23,1852	22,3369
24	51,1790	48,0336	45,5584	42,9798	39,3641	36,4150	33,1962	31,1325	29,5533	28,2412	27,0960	26,0625	25,1064	24,2037	23,3367
25	52,6187	49,4351	46,9280	44,3140	40,6465	37,6525	34,3816	32,2825	30,6752	29,3388	28,1719	27,1183	26,1430	25,2218	24,3366

Figura 20-4: Tabla de distribución de Chi Cuadrado
Fuente: (Universidad Carlos III de Madrid - Departamento de Estadística, 2021)

$$x^2 critico = 3{,}8415$$

Dado los datos anteriores Ho debe ser aceptada si sucede el siguiente condicionante

$$x^2\ Calculado \leq x^2\ \ critico$$

Caso contrario se rechaza Ho y se Acepta Hi

Con los datos obtenidos anteriormente donde X2 = 6.66 y X2 critico = 3.8415 se puede aplicar el criterio de decisión y obtenemos que (Grafica)

$$x^2\ 6.66\ \geq x^2\ \ critico\ 3.8415$$

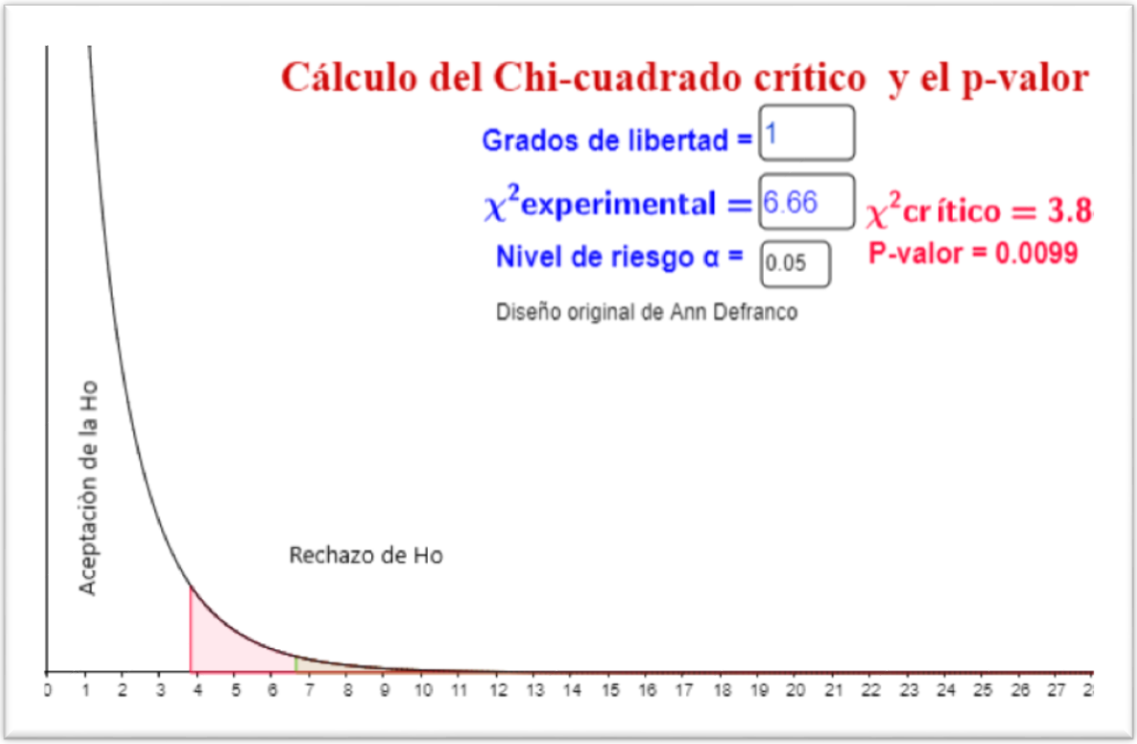


Figura 211-4: Distribución de Chi Cuadrado
Fuente: Tayupanda Luis, 2021

Interpretación

En consecuencia, con los datos obtenidos y como se aprecia en la gráfica se concluye que se rechaza la Hipótesis nula Ho y se acepta la Hipótesis alternativa Ha con un nivel de confianza del 95% y un nivel de significancia de 5%.

Quedando demostrado que:

Ha: "Al implementar una interfaz Web con el método de seguridad Json Web Token **si** garantiza el acceso y autorización segura a los servicios web del sistema académico de la Escuela Superior Politécnica de Chimborazo"

CAPITULO V

5. METODOLOGÍA

En el presente capitulo se describe la metodología para garantizar el acceso seguro a los servicios web de tipo REST de la Escuela Superior Politécnica de Chimborazo, con la generación del token JWT.

5.1 Descripción de la metodología

El desarrollo del estándar de seguridad Json Web Token ayudara a mitigar algunas vulnerabilidades presentadas en la red, en la transferencia de estados o peticiones de los servicios web REST realizadas desde cualquier cliente como puede ser móvil, escritorio o web. Este estándar deberá ser aplicada en los servicios web de tipo REST.

5.2 Fases de la metodología

Para reducir las vulnerabilidades en la trasferencia de estados o peticiones de los servicios web de tipo REST se presenta el siguiente esquema.

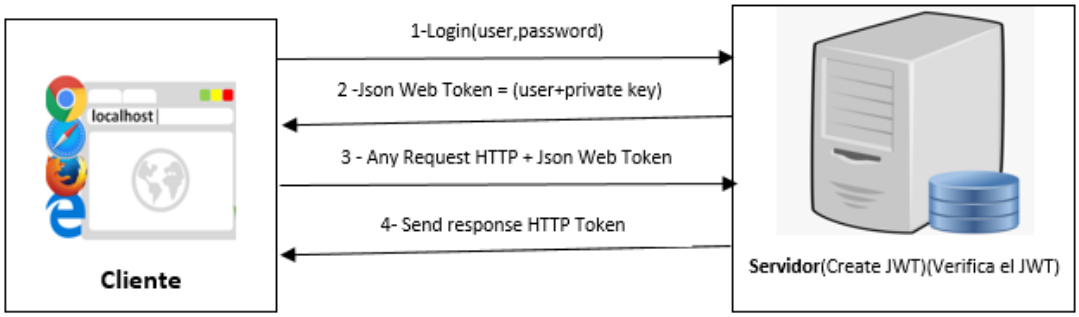


Figura 1-5: Fases de JWT
Fuente: Tayupanda Luis, 2021

Como se puede apreciar en la (Figura 1-5) la secuencia que se debe seguir para una transferencia de estados de una manera segura.

1. El cliente por medio de una petición POST se autentica con el usuario y contraseña.
2. El servidor verifica y crea el token de seguridad con las propiedades del usuario.
3. El cliente para realizar cualquier petición HTTP al servidor debe adjuntar el token de seguridad.
4. El servidor verifica y valida el token de seguridad, y si todo esta correcto devuelve la petición.

A continuación, se detalla cada una de ellas.

5.3 Autenticación y generación del Token

Por medio de la herramienta POSTMAN y el método POST enviamos el usuario y la contraseña para poder generar el token de seguridad en el servidor. El servidor una vez que verifica que el usuario y contraseña es el correcto nos genera un token de seguridad con los privilegios correspondientes que en el cuerpo del token va firmado digitalmente.

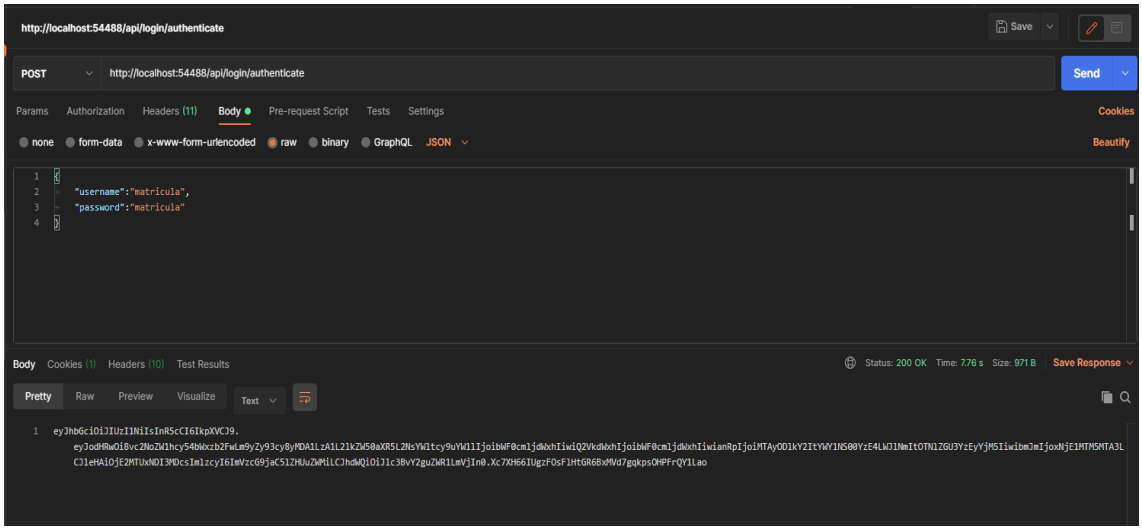


Figura 22-5: Autenticación y generación del JWT
Fuente: Tayupanda Luis, 2021

De esta forma obtenemos el token de seguridad que viene en tres partes separados por un punto.

5.3 Configuración de la cabecera JWT

Una vez obtenido el token, para poder hacer las peticiones debemos configúralo en nuestro cliente POSTMAN para enviarlo en cada petición HTTP. Para ello debemos seleccionar el menú autorización, luego seleccionar el tipo de seguridad en nuestro caso seleccionamos “Bearer Token” posterior a eso ingresamos el token de seguridad.

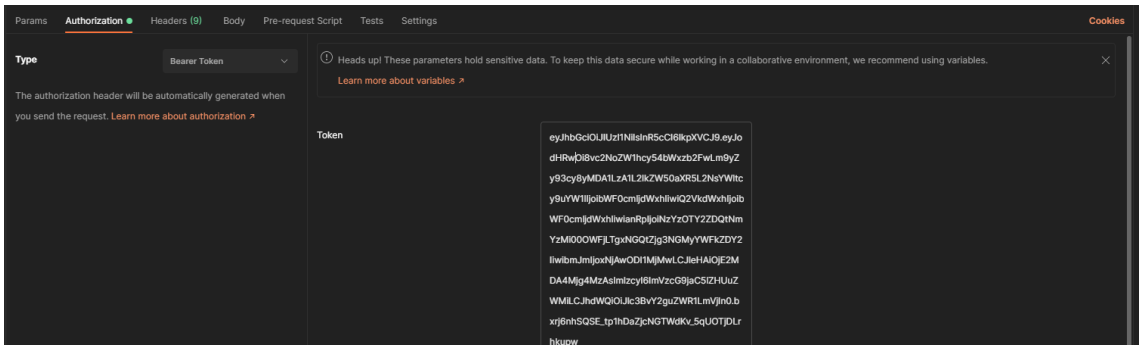


Figura 3-5: Configuración de la cabecera JWT
Fuente: Tayupanda Luis, 2021

5.4 Transferencias de estados

Una vez configurado la cabecera de seguridad, realizamos la petición HTTP al servidor con la url correspondiente, el servidor verifica y valida el token de seguridad y devuelve la información correspondiente en un formato JSON.

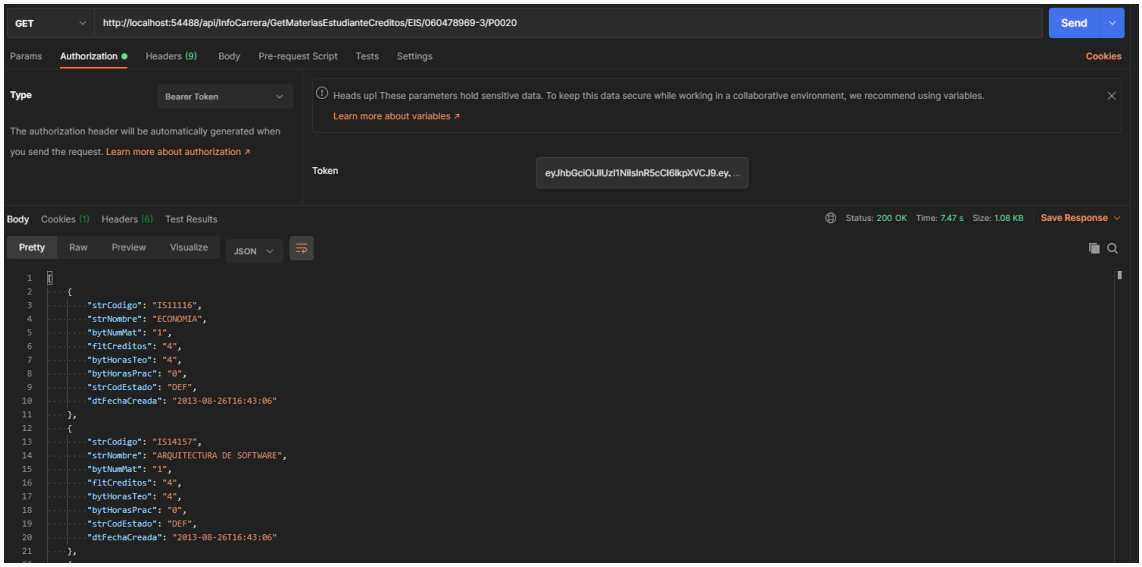


Figura 4-5: Petición GET
Fuente: Tayupanda Luis, 2021

A continuación, en la **(Figura 5-5)** se puede observar que el servidor nos devuelve el estado de la petición 401, que corresponde a no autorizado, esto puede ser porque el token ya caduco o esta incorrecto.

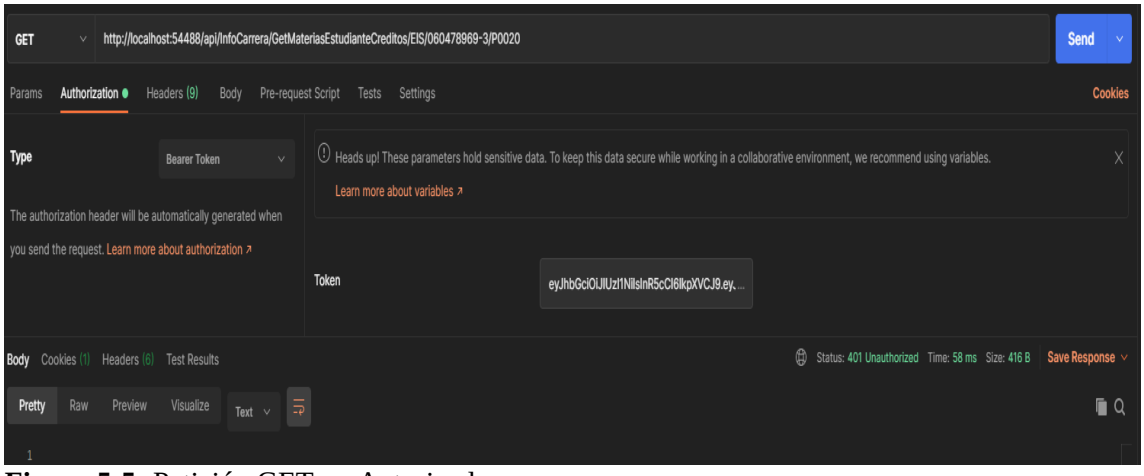


Figura 5-5: Petición GET no Autorizado
Fuente: Tayupanda Luis, 2021

5.5 Métodos de Seguridad de los Servicios Web

Los servicios web de tipo REST pueden utilizar varios métodos de seguridad para la transferencia de estados, a continuación, se detalla cada uno de ellos.

Tabla 1-5: Métodos y estándares de seguridad

Métodos	**Descripción**
Basic HTTP Authentication	En la autenticación básica, el usuario debe enviar el ID del usuario y la contraseña en el formato userid: password codificado en formato base64. Este método se puede seleccionar sobre el protocolo https. No se aconseja utilizarlo sobre el protocolo HTTP, ya que las credenciales se transferencia en formato plano.
HTTP Digest Access Authentication	es uno de los métodos usados en servidores web para negociar credenciales, tales como nombre de usuario y contraseña, desde el navegador web. El segundo mecanismo es Autenticación de acceso básica. El método Digest access authentication es usado para confirmar la identidad de un usuario antes de servir información sensible, como el historial de transacciones de un banco.
Json Web Token	La autenticación de token del portador también se conoce como autenticación basada en Token. Cuando el usuario inicia sesión en una aplicación utilizando las credenciales, el servidor de Autorización genera un token criptográfico para identificar al usuario de forma exclusiva
OAuth2.0	Es un framework de autorización que permite a los usuarios otorgar un sitio web o aplicación de terceros para acceder a los recursos protegidos del usuario sin revelar sus credenciales o identidad. Para ese fin, un servidor OAuth 2.0 emite tokens de acceso que las aplicaciones cliente pueden usar para acceder a recursos protegidos en nombre del propietario del recurso.

Realizado por: Tayupanda Luis, 2021

En la presente investigación se hizo el uso de Json Web Token debido a su amplia seguridad de los servicios web de tipo REST antes de las evaluaciones de los frameworks del lado del cliente.

5.6 Consideraciones adicionales

Al desarrollar servicios web de tipo REST hay que tener en cuenta un modelo de mejores prácticas para poder abarcar todos los puntos clave de las vulnerabilidades que se presenta en la red.

Unas veces desplegado los servicios web de tipo REST en un servidor de producción es recomendable utilizar el protocolo de seguridad para la comunicación entre aplicaciones HTTPS.

CONCLUCIONES

- ✓ Se concluye que el estándar de seguridad Json Web Token en los servicios web de tipo REST ayuda a disminuir las vulnerabilidades de la red, permitiendo tener privilegios especiales para poder acceder a la información de manera segura.

- ✓ La tecnología de servicios web de tipo REST permite tener varios front end con un unico back end, a su vez esta tecnología admite escalabilidad, y flexibilidad al momento de la transferencia de estados.

- ✓ Se desarrolló e implemento el estándar de seguridad Json Web Token para el escenario de pruebas del pentesting con la herramienta Vooki, obteniendo un 80 % de optimización de números de vulnerabilidades, de esta manera se podría precautelar la integridad de la información.

RECOMENDACIONES

- ✓ Implementar un servicio web de tipo REST seguro, no solo depende de tener estándares y técnicas seguras de desarrollo, también implica tener una adecuada configuración del ambiente donde se los aloje.

- ✓ Al desarrollar servicios web de tipo REST hay que tener en cuenta un modelo de mejores prácticas para poder abarcar todos los puntos clave de las vulnerabilidades que se presenta en la red.
- ✓ Al desplegar los servicios web de tipo REST en un servidor de producción es recomendable utilizar los protocolos y estándares de seguridad para la comunicación segura entre aplicaciones.

GLOSARIO

ESPOCH: Escuela superior Politécnica de Chimborazo.

GET: Método para obtener un listado o un recurso en concreto.

POST: Método para crear recursos nuevos.

PUT: Método para modificar.

DELETE: Método para borrar un recurso, un dato de la base de datos.

JWT: JSON Web Token es un estándar qué está dentro del documento RFC 7519. En el mismo se define un mecanismo para poder propagar entre dos partes, y de forma segura, la identidad de un determinado usuario, además con una serie de claims o privilegios.

HTTP: Es el protocolo que se usa para comunicarse con el servidor web con el fin de acceder a un navegador web o página web.

REST: Es aquél servicio web que está basado en la arquitectura REST. Los servicios Web RESTful se basan en recursos. Un recurso es una entidad, la cual se almacena principalmente en un servidor y el cliente solicita el recurso utilizando servicios Web RESTful.

VOOKI: Es una herramienta de pentesting, que ayuda a encontrar vulnerabilidades en los servicios web.

VULNERABILIDAD: La vulnerabilidad es la incapacidad de resistencia cuando se presenta un fenómeno amenazante, o la incapacidad para reponerse después de que ha ocurrido un desastre. Por ejemplo, las personas que viven en la planicie son más vulnerables ante las inundaciones que los que viven en lugares más altos.

JSON: Es un formato ligero de intercambio de datos, que resulta sencillo de leer y escribir para los programadores y simple de interpretar y generar para las máquinas.

WEB: Se designa como ‘la web’ al sistema de gestión de información más popular para la trasmisión de datos a través de internet.

TOKEN: Un token es un objeto digital que tiene valor en cierto contexto o para determinada transferencia de estados, de los servicios web.

API: El término API es una abreviatura de Application Programming Interfaces, que en español significa *interfaz de programación de aplicaciones*.

BIBLIOGRAFÍA

Aguilar, S. (05 de 08 de 2005). *Formulas para el calculo de muestra*. Obtenido de Formulas para el calculo de muestra: https://www.redalyc.org/pdf/487/48711206.pdf

Alicante, U. d. (2014). *Universidad de Alicante*. Recuperado el 30 de Marzo de 2018, de http://www.jtech.ua.es/j2ee/publico/servc-web-2012-13/sesion01-apuntes.html

Alvarez, M. (2016). *Desarrollo Agil con SCRUM*. Obtenido de http://cic.puj.edu.co/wiki/lib/exe/fetch.php?media=materias:sg07.p02.scrum.pdf

Aransay, A. (01 de Marzo de 2009). *Metodologías para el desarrollo de servicios en la web.* Recuperado el 03 de Abril de 2018, de http://www.albertolsa.com/wp-content/uploads/2009/07/mdsw-revision-de-los-servicios-web-soap_rest-alberto-los-santos.pdf

Corredor, E. (Diciembre de 2017). *Estado del arte revision sistematica de la seguridad orientada a REST*. Recuperado el 05 de Abril de 2018, de http://repository.ucatolica.edu.co/jspui/bitstream/10983/15230/1/Revisi%C3%B3n%20sistem%C3%A1tica%20de%20la%20seguridad%20orientada%20a%20REST.pdf

Cruz, J., & Loaiza, L. (01 de Abril de 2017). *Consumo de servicio web.* Recuperado el 01 de Abril de 2018, de http://repository.udistrital.edu.co/bitstream/11349/6781/1/Sebasti%C3%A1n%20Cruz%20Mora%202017.pdf

EcuRed. (05 de Marzo de 2016). *Desarrollo Web*. Recuperado el 03 de Marzo de 2018, de http://www.ecured.cu/Aplicaci%C3%B3n_web

Franklind. (2016). *Proyectos Agiles.* Obtenido de https://proyectosagiles.org/que-es-scrum/

INTECO. (2009). INGENIERÍA DEL SOFTWARE: METODOLOGÍAS Y CICLOS DE VIDA. En INTECO, *INGENIERÍA DEL SOFTWARE* (págs. 44,45). España.

Jones, M. (27 de 04 de 2011). *El conjunto emergente de protocolos de identidad basados en JSON*. Obtenido de El conjunto emergente de protocolos de identidad basados en JSON: https://www.w3.org/2011/identity-ws/papers/idbrowser2011_submission_30.pdf

Lujan Mora, S. (31 de Octubre de 2002). *Programacion de plicaiones web.* Recuperado el 01 de Abril de 2018, de https://gplsi.dlsi.ua.es/~slujan/materiales/pi-cliente2-muestra.pdf

Luján Mora, S. (2013). *Programacion de Aplicaciones Web.* España: Altaria.

Méndez, V., & Garcia, V. (31 de 12 de 2019). *Análisis de sistemas de autenticacion y autorizacion para entornos web*. Obtenido de Análisis de sistemas de autenticacion y autorizacion para entornos web: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/107926/6/oparrabTFM0120memoria.pdf

Netquest. (12 de 12 de 2014). *Netquest*. Obtenido de Netquest: https://www.netquest.com/blog/es/la-escala-de-likert-que-es-y-como-utilizarla

Nuñez, B., & Gaibor, J. (2015). *Determinacion del Cumplimiento de las Metodologías XP y SCRUM con relacion al estandar IEEE 12207 Aplicado al Sistema de Control de Proveeduria en la CACECH.* Riobamba: Dspace Espoch.

Quispe, L. (04 de Abril de 2017). *Atacando servicios wen en el mundo real.* Recuperado el 01 de Abril de 2018, de https://www.owasp.org/images/6/63/LatamTour2017_LuisQuispe_AtacandoServiciosWeb.pdf

Rivera, H. (02 de Septiembre de 2011). *Navegacion segura en internet.* Recuperado el 01 de Abril de 2018, de http://www.itsteziutlan.edu.mx/site2010/pdfs/2011/09/http-o-https.pdf

Romero, A. M. (2006). *Seguridad Informatica y alta disponibilidad*. Obtenido de Seguridad Informatica y alta disponibilidad: http://migasfree.educa.aragon.es/presentaciones/curso-administracion_linux-gerencia_dga-materiales/PDFs-Documentacion/libro_seguridad_informatica.v14.5.baja_resol.pdf

Sandobal, F. (2009). *Intelligence to Bussiness*. Recuperado el 01 de Abril de 2018, de http://www.i2btech.com/blog-i2b/tech-deployment/que-se-entiende-por-soa-y-cuales-son-sus-beneficios/

Tahuiton Mora, J. (2011). *Arquitectura de software para sistema web.* Obtenido de http://delta.cs.cinvestav.mx/~pmalvarez/tesis-tahuiton.pdf

VERISIGN. (2018). *VERISIGN*. Recuperado el 01 de Abril de 2018, de https://www.verisign.com/es_LA/website-presence/website-optimization/ssl-certificates/index.xhtml

Villa, F. (01 de 10 de 2019). *Propuesta de mejores practicas de seguridad*. Obtenido de Propuesta de mejores practicas de seguridad: http://dspace.espoch.edu.ec/bitstream/123456789/13028/1/20T01267.pdf

ANEXO A: DESARROLLO DE LA PROGRAMACIÓN

1. Desarrollo del método de autenticación a los servicios web REST

```
[HttpPost]
        [Route("authenticate")]
        public IActionResult  Authenticate(credentials login)
        {
            try
            {
                if (new WebServiceBase().ValidarCredenciales(login))
                {

                        HttpContext.Session.SetString("nombre", login.username);
                        HttpContext.Session.SetString("pw", login.password);
                    return new
ObjectResult(TokeGeneratorController.GenerateTokenJwt(login));
                }
                else {
                    return BadRequest();
                }

            }
            catch (Exception ex)
            {
                new ApplicationException(string.Format("Existe un error en : {0}
to {1}", "Get", "TokenController"), ex);
                return Unauthorized();
            }
            }
```

2. Método de generación del token de seguridad Json Web Token

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Security.Claims;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.IdentityModel.Protocols;
using Microsoft.IdentityModel.Tokens;
using System.Configuration;
using WebApiInterop.Models;
using System.IdentityModel.Tokens.Jwt;
using System.Text;

namespace WebApiInterop.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class TokeGeneratorController : ControllerBase
    {
        private const string SECRET_KEY = "QWERTYUIOPASDFGHJKLÑZXCVBNM";
        public static readonly SymmetricSecurityKey SIGNING_KEY = new
                 SymmetricSecurityKey(Encoding.UTF8.GetBytes(SECRET_KEY));
        public static string GenerateTokenJwt(credentials login)
```

```
        {
        var token = new JwtSecurityToken(
              issuer: "espoch.edu.ec",
              audience: "espoch.edu.ec",
              claims: new Claim[] { new Claim(ClaimTypes.Name, login.username),
                                    new Claim("Cedula",login.password),
                                    new Claim(JwtRegisteredClaimNames.Jti,
Guid.NewGuid().ToString())
                                  },
              notBefore: new DateTimeOffset(DateTime.Now).DateTime,
              expires: new DateTimeOffset(DateTime.Now.AddMinutes(60)).DateTime,
              signingCredentials: new SigningCredentials(SIGNING_KEY,
              SecurityAlgorithms.HmacSha256)
              );
            var tokenUsuario = new JwtSecurityTokenHandler().WriteToken(token);
            return tokenUsuario;
        }
    }
}
```

3. Controlador de los métodos de generación de Token

```
using Microsoft.AspNetCore.Mvc;
using WebApiInterop.Models;
using System.Threading;
using Microsoft.AspNetCore.Identity;
using Microsoft.Extensions.Configuration;
using Microsoft.IdentityModel.Tokens;
using System.Text;
using System;
using Microsoft.AspNetCore.Http;


namespace WebApiInterop.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class LoginController : Controller
    {
        private const string SECRET_KEY = "QWERTYUIOPASDFGHJKLÑZXCVBNM";
        public static readonly SymmetricSecurityKey SIGNING_KEY = new
                        SymmetricSecurityKey(Encoding.UTF8.GetBytes(SECRET_KEY));
        [HttpGet]
        [Route("echoping")]
        public IActionResult EchoPing()
        {
            return Ok(true);
        }

        [HttpGet]
        [Route("echouser")]
        public IActionResult EchoUser()
        {
            var identity = Thread.CurrentPrincipal.Identity;
            return Ok($" IPrincipal-user: {identity.Name} - IsAuthenticated:
{identity.IsAuthenticated}");
        }

        [HttpPost]
        [Route("authenticate")]
```

```
        public IActionResult  Authenticate(credentials login)
        {
            try
            {
                if (new WebServiceBase().ValidarCredenciales(login))
                {

                        HttpContext.Session.SetString("nombre", login.username);
                        HttpContext.Session.SetString("pw", login.password);
                    return new
ObjectResult(TokeGeneratorController.GenerateTokenJwt(login));
                }
                else {
                    return BadRequest();
                }

            }
            catch (Exception ex)
            {
                new ApplicationException(string.Format("Existe un error en : {0}
to {1}", "Get", "TokenController"), ex);
                return Unauthorized();
            }
        }
    }
}
```

4. Métodos de validación y acceso a la información

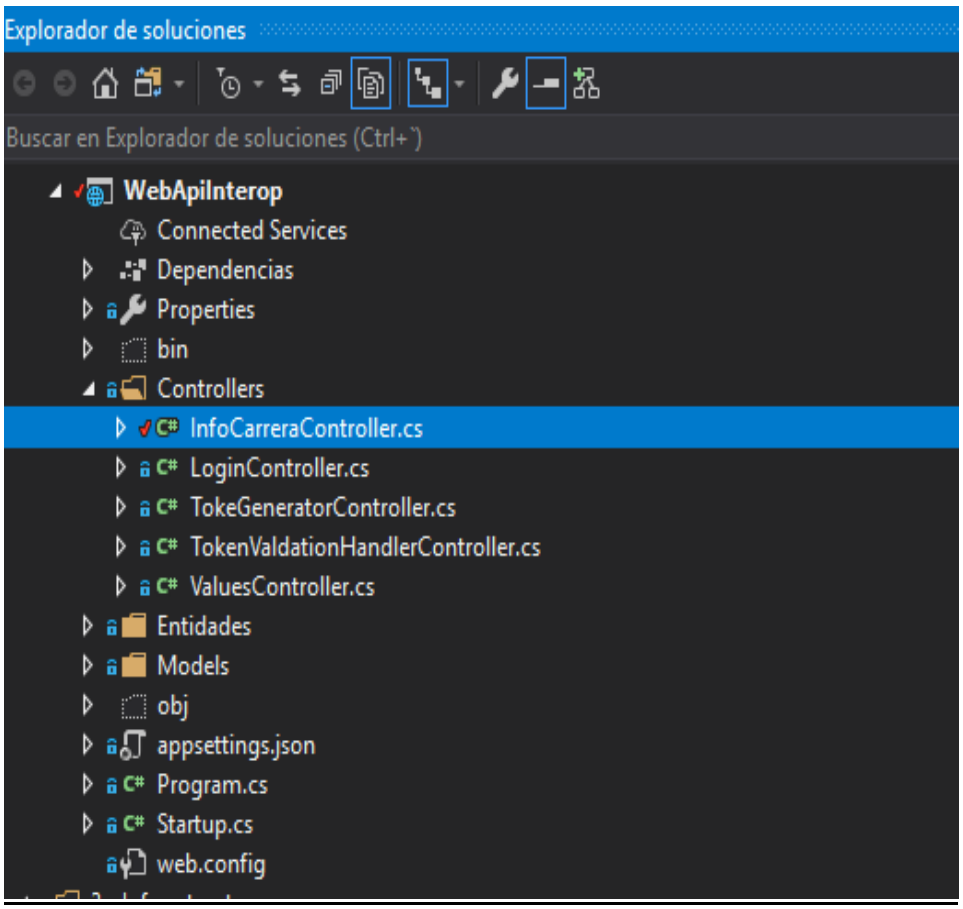
```
using System;
using System.Collections.Generic;
using Microsoft.AspNetCore.Mvc;



using OAS_InteropEntComp;
using GestorErrores;
using Microsoft.AspNetCore.Authorization;
using WebApiInterop.Models;

using System.Web.Services;
using Microsoft.AspNetCore.Http;
using WebApiInterop.Entidades;

namespace WebApiInterop.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    [Authorize]
    public class InfoCarreraController : Controller
    {
        /*
        [HttpPost]
        [Route("authenticate")]*/


[HttpGet("GetMateriasEstudianteCreditos/{CodCarrera}/{Cedula}/{CodPeriodo}")]
        public List<MateriasCre> GetMateriasEstudianteCreditos(string CodCarrera,
string Cedula, string CodPeriodo)
        {
```

```
            List<MateriasCre> rstMateriasEstudianteCreditos = new
List<MateriasCre>();
            Errores err = new Errores();
            try
            {
                rstMateriasEstudianteCreditos =
Models.MateriasComponent.GetMateriasEstudianteCre(CodCarrera, Cedula,
CodPeriodo);

            }
            catch (Exception ex)
            {
                err.SetError(ex, "InterOP - GetMateriasEstudianteCreditos");
            }

            return rstMateriasEstudianteCreditos;
        }

        // GET: api/InfoCarrera/5
        [HttpGet]
        [Route("GetInscripcionesEstudiante/{strCedula}")]
        public List<Inscripcion> GetInscripcionesEstudiante(string strCedula)
        {
            List<Inscripcion> objInscriciones = new List<Inscripcion>();
            try
            {
                if (!(string.IsNullOrEmpty(strCedula)))
                {
                    objInscriciones =
Models.UsuariosComponent.GetInscripcionesEstudiante(strCedula);
                }
                else {
                    return objInscriciones;
                }
            }
            catch (Exception ex)
            {
                new ApplicationException(string.Format("Existe un error en : {0}
to {1}", "Get", "GetInscripcionesEstudiante"), ex);
            }
            return objInscriciones;
        }

        [HttpGet]
        [Route("GetSemanasCarrera/{strCodCarrera}")]
        public int GetSemanasCarrera(string strCodCarrera)
        {
            try
            { //REVISAR EN EL ENDPOIT NO COGE LA RUTA QUE DEBE SER
                return GeneralComponent.getSemanasCarrera(strCodCarrera);
            }
            catch (Exception ex)
            {
                new ApplicationException(string.Format("Existe un error en : {0}
to {1}", "Get", "GetSemanasCarrera"), ex);
                return 0;
            }
        }

         [HttpGet]
         [Route("GetParametrosCarrera/{strCodCarrera}")]
```

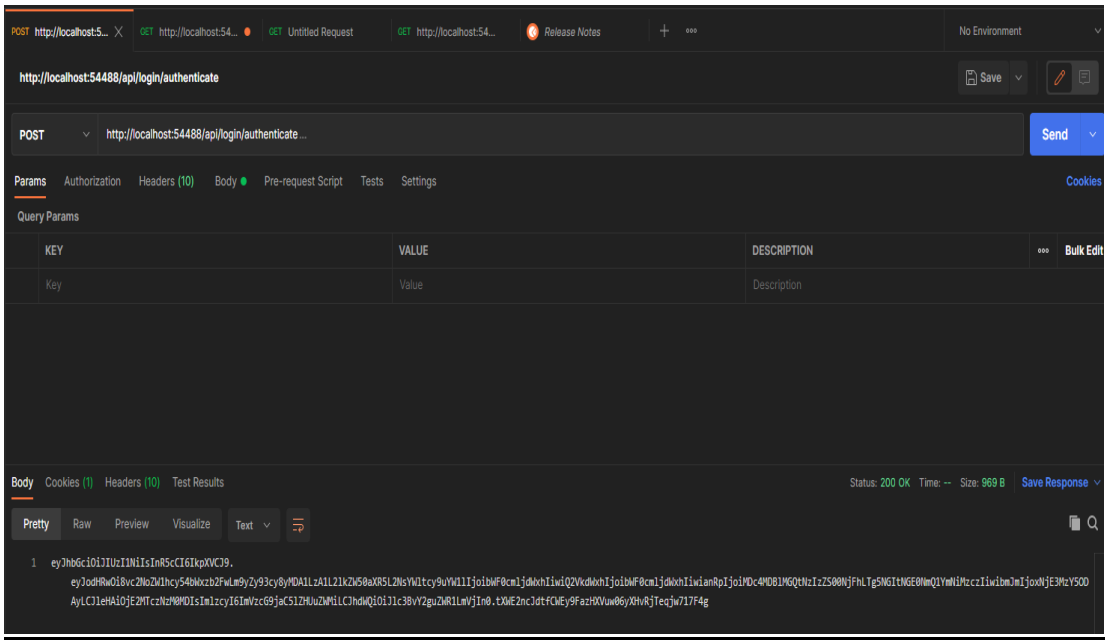
```
         //[WebService(Description = "Common Server Variables", Namespace =
"http://www.contoso.com/")]
         public ParamCarrera GetParametrosCarrera(string strCodCarrera)
         {
             return Models.MateriasComponent.GetParamCarrera(strCodCarrera);

         }

        [HttpGet]
        [Route("GetDatosUsuarioCarrera/{strCodCarrera}/{strCedula}")]
        public Persona GetDatosUsuarioCarrera(string strCodCarrera, string
strCedula)
        {
             return
Models.UsuariosComponent.GetDatosUsuarioCarrera(strCodCarrera, strCedula);
        }

    }
}
```

ANEXO B: HERRAMIENTAS Y ESTRUCTURA DE DESARROLLO

1. Estructura del desarrollo de la investigación



2. Herramienta POSTMAN para la transferencia de estados de los servicios web



3. Herramienta VOOKY para el escaneo de vulnerabilidades