INFORME DE ATRIBUTOS DE CALIDAD APLICATIVO CAFSI

DEIVY RODRIGUEZ  
CARLOS ANDRES RAMIREZ  
MICHAEL DUQUE  
MANUEL RUIZ

SERVICIO NACIONAL DE APRENDIZAJE “S.E.N.A”  
ANALISIS Y DESARROLLO DE SISTEMAS DE INFORMACIÓN  
BOGOTA D.C.  
2021

Contenido

[INTRODUCCIÓN 3](#_Toc83580836)

[DEFINICIÓN 4](#_Toc83580837)

[DISPONIBILIDAD 5](#_Toc83580838)

[DESPLEGABILIDAD 6](#_Toc83580839)

[ESCALABILIDAD 7](#_Toc83580840)

[INTEROPERATIVIDAD 8](#_Toc83580841)

[MODIFICABILIDAD. 9](#_Toc83580842)

[RENDIMIENTO 10](#_Toc83580843)

[SEGURIDAD 11](#_Toc83580844)

[TESTEABILIDAD (FACILIDAD DE PROBAR EL SISTEMA). 20](#_Toc83580845)

[USABILIDAD 21](#_Toc83580846)

[CONCLUSIÓN 22](#_Toc83580847)

# INTRODUCCIÓN

Este documento se explica los atributos de calidad de software que conllevaran a que nuestro sistema funcione de manera óptima, segura y confiable.

# DEFINICIÓN

Se entienden como atributos de calidad de software las propiedades que debe tener un sistema de software para satisfacer las necesidades requerida por el usuario que lo operara, de manera segura y eficiente.

# DISPONIBILIDAD

La disponibilidad se refiere a la capacidad de un sistema para estar correctamente operativo para su uso cuando se requiere. Entendiendo este concepto aplicándolo a nuestro sistema de software, el aplicativo actualmente se encuentra desplegado en un ambiente de pruebas con el proveedor web de **HEROKU** el cual se caracteriza por manejar sus procesos en nube a través de contendores en Linux los cuales se administran a través de la plataforma creada por heroku denominada heroku runtime. Ya que es la responsable de administrar y orquestar los contendores donde se ejecutan las aplicaciones, también se les conoce como “**Dynos”**.

Esto significa que la ejecución y proceso de administración del servidor donde esta alojada actualmente la aplicación es administrada a través de estos “**Dynos**”, los cuales como se explicó anteriormente son contenedores en Linux aislados y virtualizados. Se caracterizan por tener un escalamiento basado en los recursos que demanden cada aplicación.

Ya entendiendo básicamente la arquitectura de funcionamiento del servidor donde se encuentra actualmente nuestra aplicación, y centrándonos en este atributo podríamos decir que en cuanto a disponibilidad se refiere heroku proporciona un asilamiento por cada aplicación es decir que si algún “**dyno”** donde se encuentra actualmente funcionando la aplicación falla este levantara inmediatamente otro contendedor (“**dyno**”), para que esta puede seguir operativa sin ver afectación en su servicio garantizando que esta este aproximadamente 90% disponible para su uso 24 horas 7 días a la semana, también cabe aclarar que esta ofrece bastante estabilidad en su funcionamiento, ya que está apoyada sobre AWS (Amazon Web Service).

Para el despliegue en ambiente de producción contamos con el proveedor DONGEE, el cual se caracteriza por estar gestionado también bajo el servicio de AWS. Brindándonos una alta disponibilidad y capacidad de almacenamiento de 10 GB en estos contenedores, también donde estará alojada nuestra aplicación cuenta con aceleración de conectividad CDN que es una red de distribución de contenidos la cual nos garantiza que la aplicación este óptimamente disponible a través de un amplio ancho de banda brindado por esta configuración de RED administrado por este proveedor.

De igual manera garantiza que el aplicativo esté disponible las 24 horas 7 días a la semana presentando un porcentaje disponibilidad del 90% anual.

# DESPLEGABILIDAD

Se conoce este atributo como la facilidad en la que un sistema puede ser publicado en el servidor web donde este alojado. Para nuestro sistema actualmente se encentra desplegado en ambiente de pruebas a través del proveedor **HEROKU,** este nos brinda una desplegabilidad bastante sencilla e intuitiva a través de su página, ya que básicamente se centra en tener el sistema o aplicación subido en GitHub, el cual cuenta con el complemento de vinculo del sistema a desplegar a través de conexión con GitHub, y este se puede configurar para que una vez esté vinculado el repositorio en Git con heroku y cuenta con sincronización en tiempo real para que cada vez que se suba o realice un cambio en el repositorio el cal afecte al sistema este se pueda evidenciar en el despliegue realizado en heroku.

En ambiente de producción cuenta con casi la misma particularidad de despliegue continuo con GitHub ya que este proveedor cuenta con un panel de administración simple e intuitivo el cual nos facilita el despliegue de nuestra aplicación. También cuenta con asesoría y soporte las 24 horas para cualquier tema de soporte o mantenimiento del servidor donde se encontrará alojada nuestra aplicación evitando fallos por mala mantenibilidad o administración del servidor.

# ESCALABILIDAD

Se entiende como escalabilidad, la propiedad atributo de n software para funcionar de manera óptima cuando se presentan cambios en la demanda de peticiones o carga de trabajo en el mismo. Va muy de la mano con el atributo de rendimiento.

En nuestro escenario de ambiente de Pruebas donde actualmente se encuentra nuestro aplicativo proporcionado por el servicio del proveedor **HEROKU**, este nos brinda la capacidad de extender el número de **“dynos**” asignados a nuestra aplicación para que esta trabaje de optima manera, configurándolos a través del “dashboard” que proporciona este proveedor. Aunque nuestro aplicativo no maneja una alta transaccionalidad, lo que brinda que al ser configurada adecuadamente no sufra problemas de funcionamiento por este motivo.

Ya en ambiente de producción el plan que tenemos cotizado para que permita funcionar nuestro aplicativo de manera correcta nos asegura que el servidor donde estará alojado nuestro sistema trabaje de manera óptima soportando las cargas que tendrá este sistema, ya que como se mencionó anteriormente. Este sistema no exige una carga de trabajo tal alta por ser un aplicativo administrado solo por 4 personas aproximadamente.

Si llegase a ocurrir un aumento significativo el cual requiera una mayor escalabilidad solo seria aumentar el plan actual cotizado por uno que se adapte mejor a la necesidad de procesamiento/funcionamiento que permita funcionar de mejor manera el sistema.

# INTEROPERATIVIDAD

El sistema no contara con este atributo ya que su funcionamiento se centra en un funcionamiento monolítico esto quiere decir que no se comunicara con servicios externos. Y en la especificación y/o requerimientos funcionales no se especificó, algún alcance el cual determine o requiera el desarrollo de esta funcionalidad.

# MODIFICABILIDAD.

Se entiende como la capacidad que tiene un sistema para admitir cambios que pueden ser necesarios tanto por un cambio de requerimiento como por la detención de algún error que debe ser corregido.

En nuestro sistema permite efectuar esta propiedad de manera correcta brindando la capacidad de realizar modificaciones tanto a procesos que conlleven la modificación en código fuente como a las vistas que requieran algún complemento adicional para el correcto manejo del sistema.

De igual manera cualquier modificación que se requiera hacer y que se quiera realizar una vez el proyecto sea entregado al cliente, este se puede apoyar con el manual de instalación y/o operación para efectuar dicha configuración del código fuente para realizar alguna modificación que se requiera por parte ajena a los desarrolladores del sistema.

# RENDIMIENTO

Se refiere a la capacidad de un sistema de funcionar de manera optima ante cualquier escenario donde se requiera utilizar este mismo. Mas técnicamente también mide la eficiencia del sistema ante el consumo de recursos de hardware donde el sistema estará alojado.

En ambiente de pruebas donde esta actualmente nuestro sistema a través del proveedor **HEROKU,** este garantiza un optimo manejo de recursos de hardware ya que nuestro sistema no soporta tanto stress transaccional por lo que su operatividad se centra básicamente en 4 usuarios máximo “logeados” en este sistema lo que garantiza que el rendimiento funcione de manera óptima a lo esperado. El procesamiento con 4 usuarios ingresados simultáneamente en el sistema consume aproximadamente 530 MB de memoria RAM, con una subida de proceso en hilo de menos de 1 CPU y en almacenamiento la base de datos pesa aproximadamente 300 MB y la transaccionalidad del sistema hacia la base de datos no es alta, garantizando su optimo funcionamiento.

En ambiente de producción con el proveedor de servicio en nube para despliegue de aplicativo y base de datos **Dongee,** nos brinda un almacenamiento en nube de 10 GB, una capacidad de procesamiento de 1 GB RAM con 1 CPU, pero el servidor donde estará alojado contiene “LiteSpeed” el cual nos garantiza una mayor eficiencia de procesamiento y distribución de recursos, garantizando el correcto funcionamiento de este sistema.

# SEGURIDAD

Es la capacidad que tiene un sistema para proteger la integridad de los datos y de restringir el acceso a usuarios no autorizados un sistema debe asegurar principalmente: confidencialidad, integridad, autenticidad para decir que se encuentra seguro.

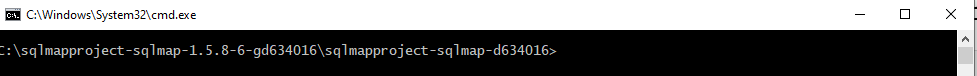
El sistema en ambiente de pruebas no cuenta con certificado SSL, sin embargo, este si permite autenticar de manera correcta restringiendo el acceso a personas sin permisos. También asegura la correcta integridad de los datos a través del adecuado manejo de roles y acceso a ciertas funcionalidades, brindando la capacidad de restringir a través del rol o personalización de acceso por usuario.

Este sistema fue testeado en este ambiente a través de las principales vulnerabilidades que la OWASP nos brinda para saber si un sistema trabaja de manera segura. Estos casos de uso de pruebas de vulnerabilidades se exponen a continuación:

**INYECCION DE CODIGO SQL**

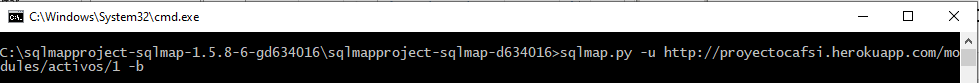
Para esta prueba se realizó con la ayuda de la herramienta de inyección SQLMap :

Se accede a la ruta local donde está ubicado la herramienta SQLMap .

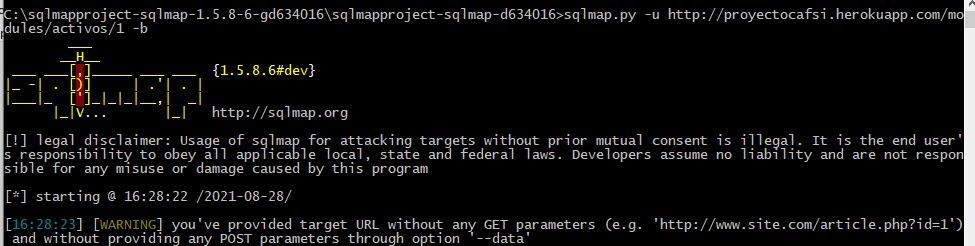


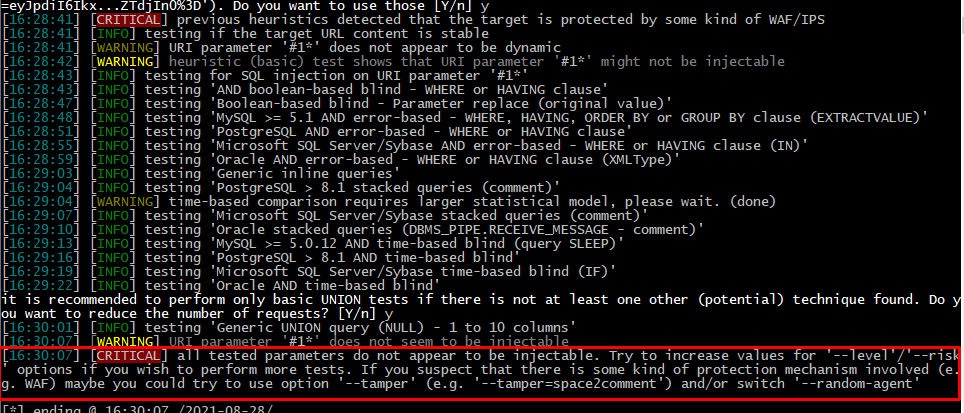
Una vez accedido a la ubicación local de la herramienta, se procede con pegar el siguiente comando:

sqlmap.py -u http://proyectocafsi.herokuapp.com/modules/activos/1 -b



Este comando ejecutará una consulta sql en la cual se podrá visualizar la versión del servidor y motor de base de datos donde este alojado el sistema.

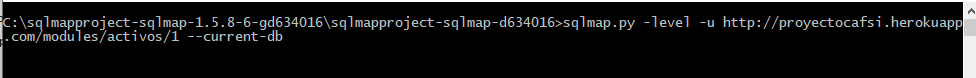


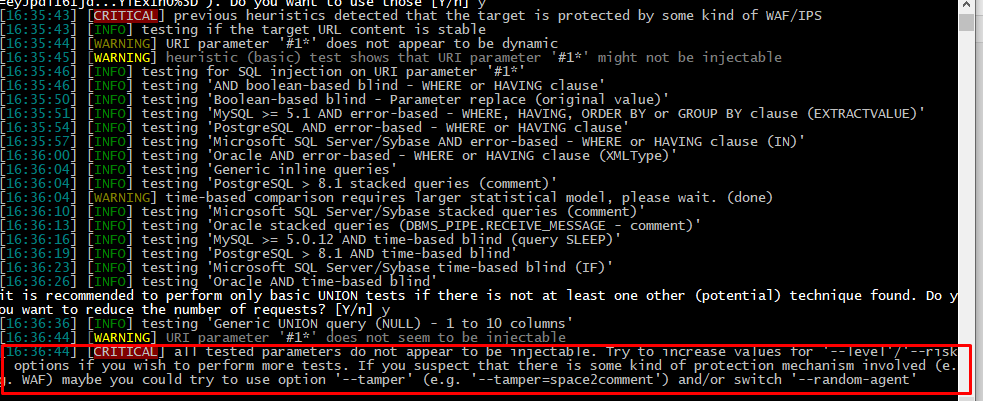


Donde se evidencia que a pesar de intentar realizar varios procesos para lograr se efectiva dicha inyección, el WAF donde esta alojada nuestra aplicación no permite hacer efectiva dicha petición.

Ahora se ejecutará el siguiente comando para evidencia si es posible la inyección, en la cual con los parámetros establecidos que se observan a continuación nos permitirá ver el nombre de la base de datos del sitio.

sqlmap.py -level -u http://proyectocafsi.herokuapp.com/modules/activos/1 --current-db





**Respuesta solicitud**

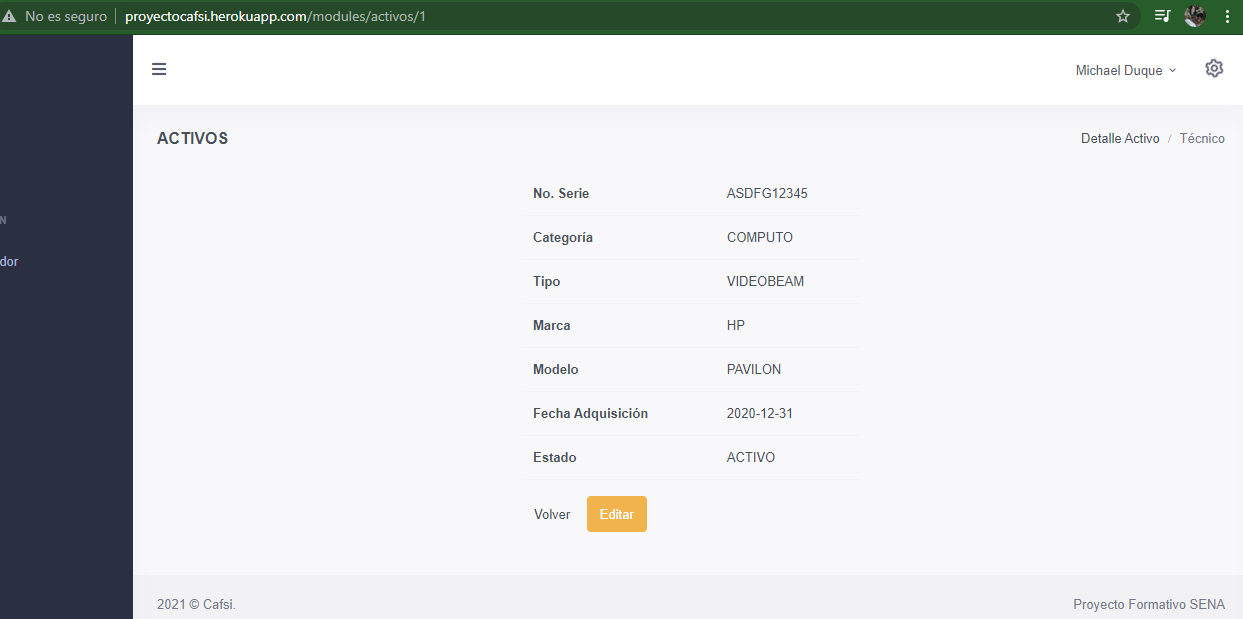
El parámetro URI '#1\*' no parece ser inyectable

[16:16:19] [CRITICAL] todos los parámetros probados no parecen ser inyectables. Intente aumentar los valores de las opciones '--level'/'--risk' si desea realizar más pruebas. Si sospecha que hay algún tipo de mecanismo de protección involucrado (por ejemplo, WAF) tal vez podría intentar usar la opción '--tamper' (por ejemplo, '--tamper=space2comment') y/o cambiar '--random-agent

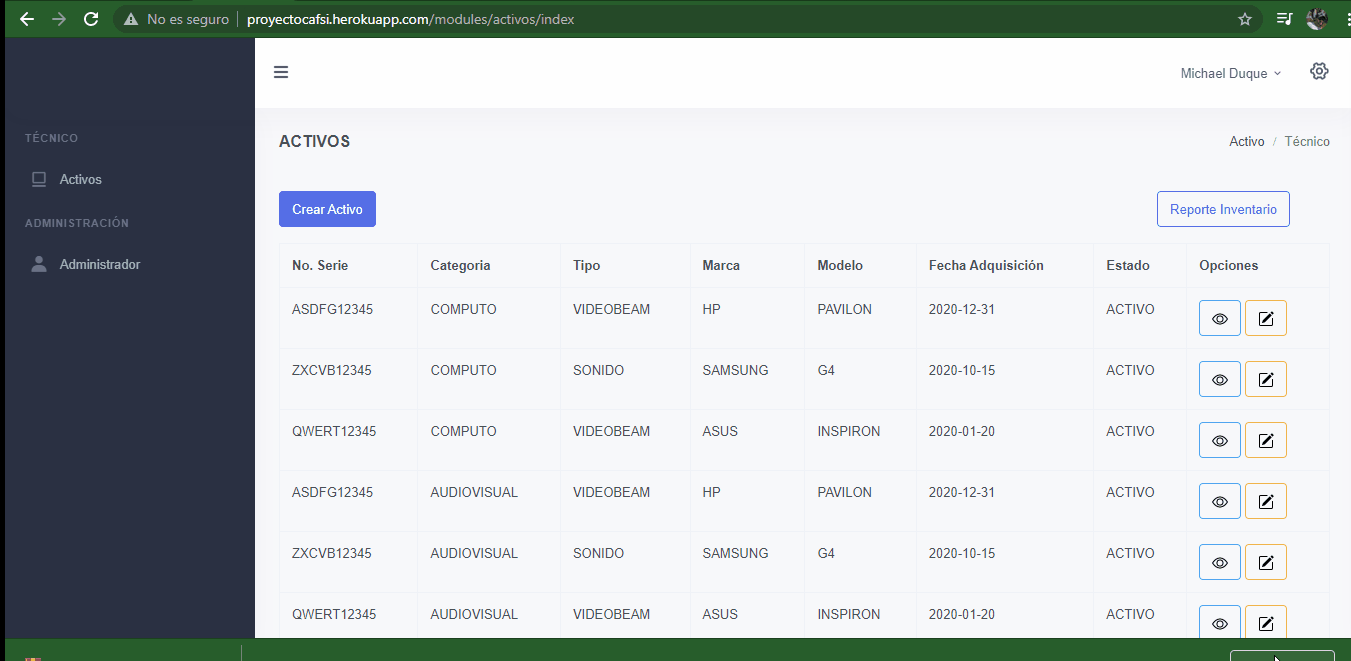
Donde se evidencia que como en la otra solicitud, no permite hacer efectiva la petición enviada a través de la url al sitio.

**Configuración de seguridad incorrecta:**

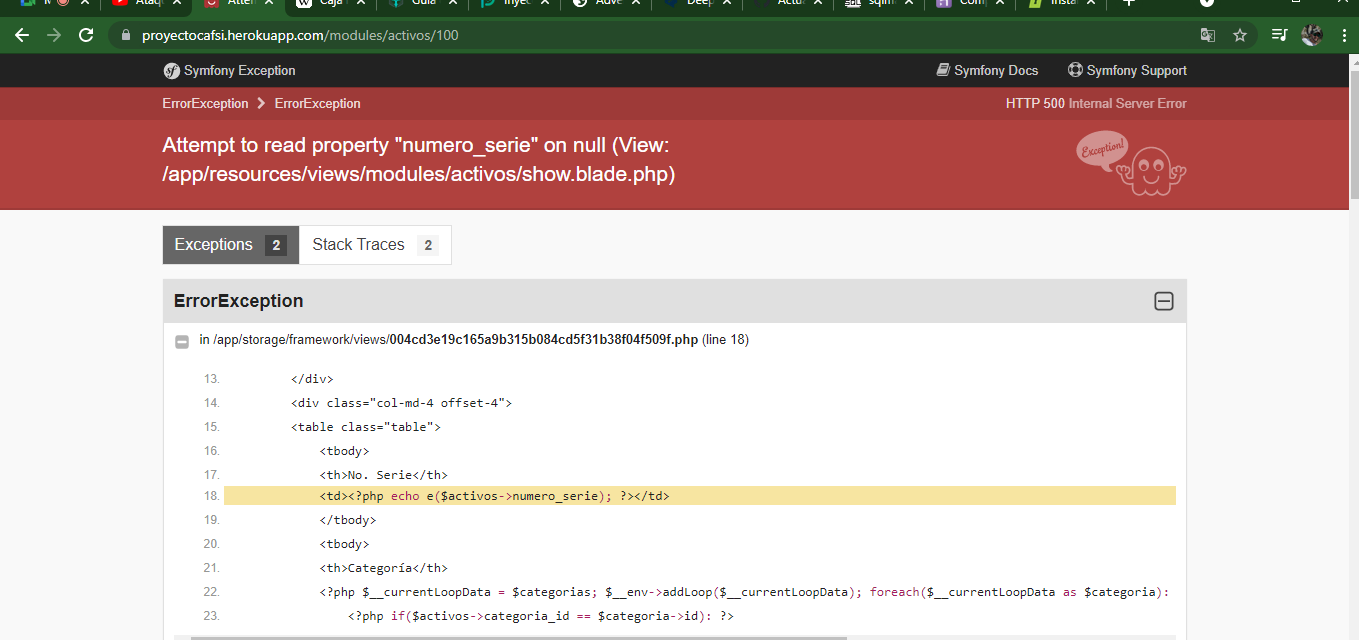
Para este escenario de prueba se accede al sitio web a través de la siguiente opción:



Donde se realiza el cambio de parámetro directamente desde el código tal y como se observa a continuación:



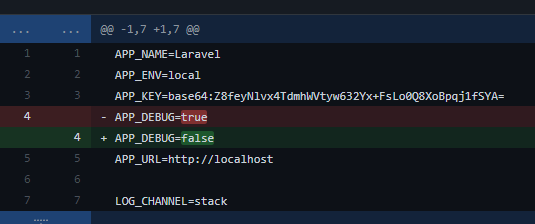
El cual genera el siguiente error:



Este error le da al atacante una visualización del código fuente que se esta explotando por la ejecución realizada.

Para prevenir esta vulnerabilidad se debe aplicar el siguiente cambio en el archivo de configuración del sitio. env

Donde se actualiza el siguiente parámetro

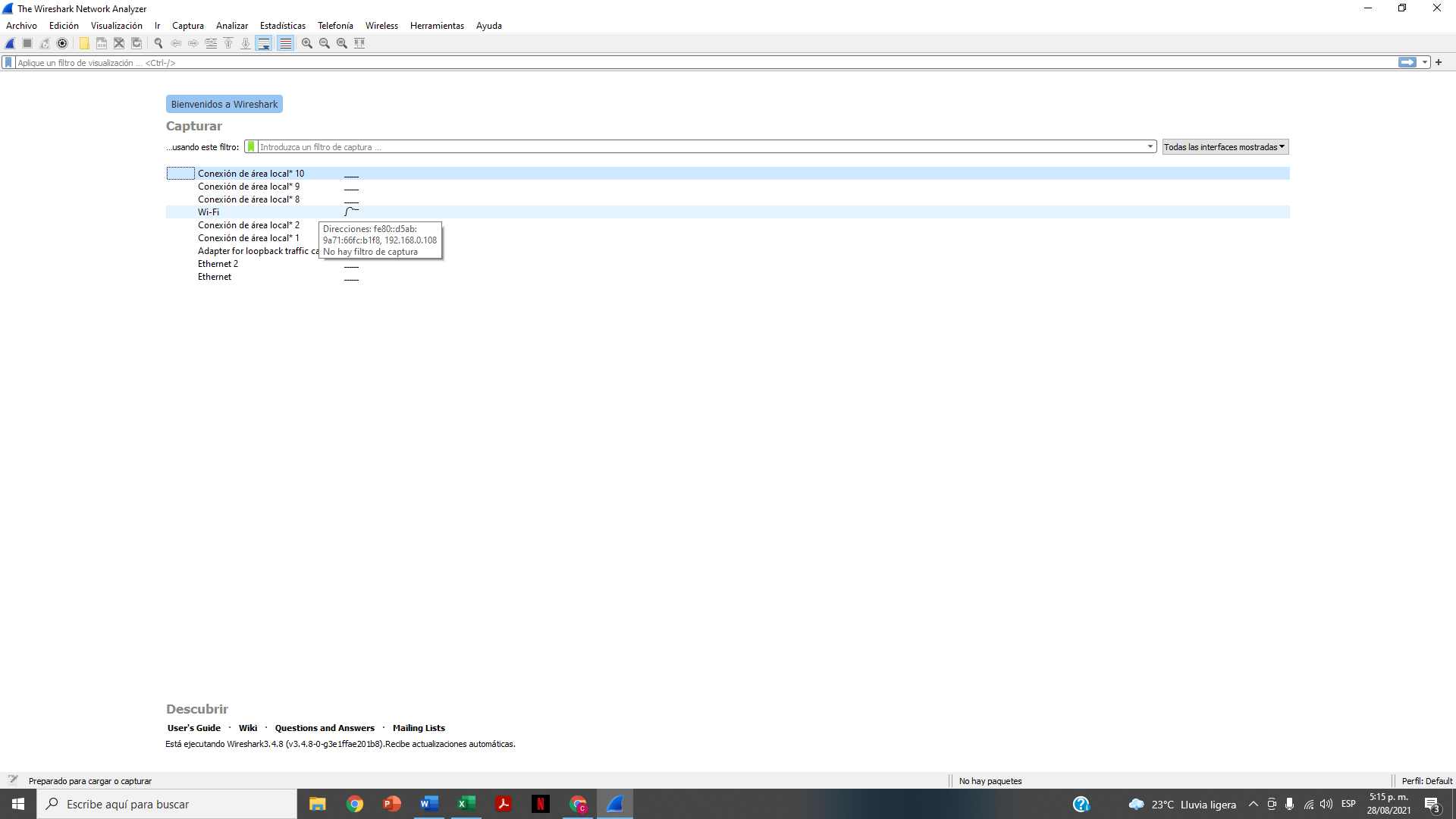


Y una vez aplicado este cambio, se evita que el sitio retorne errores de códigos en la publicación actual del ambiente de producción. Evitando la exposición de datos que le den al atacante información del código fuente ejecutado.

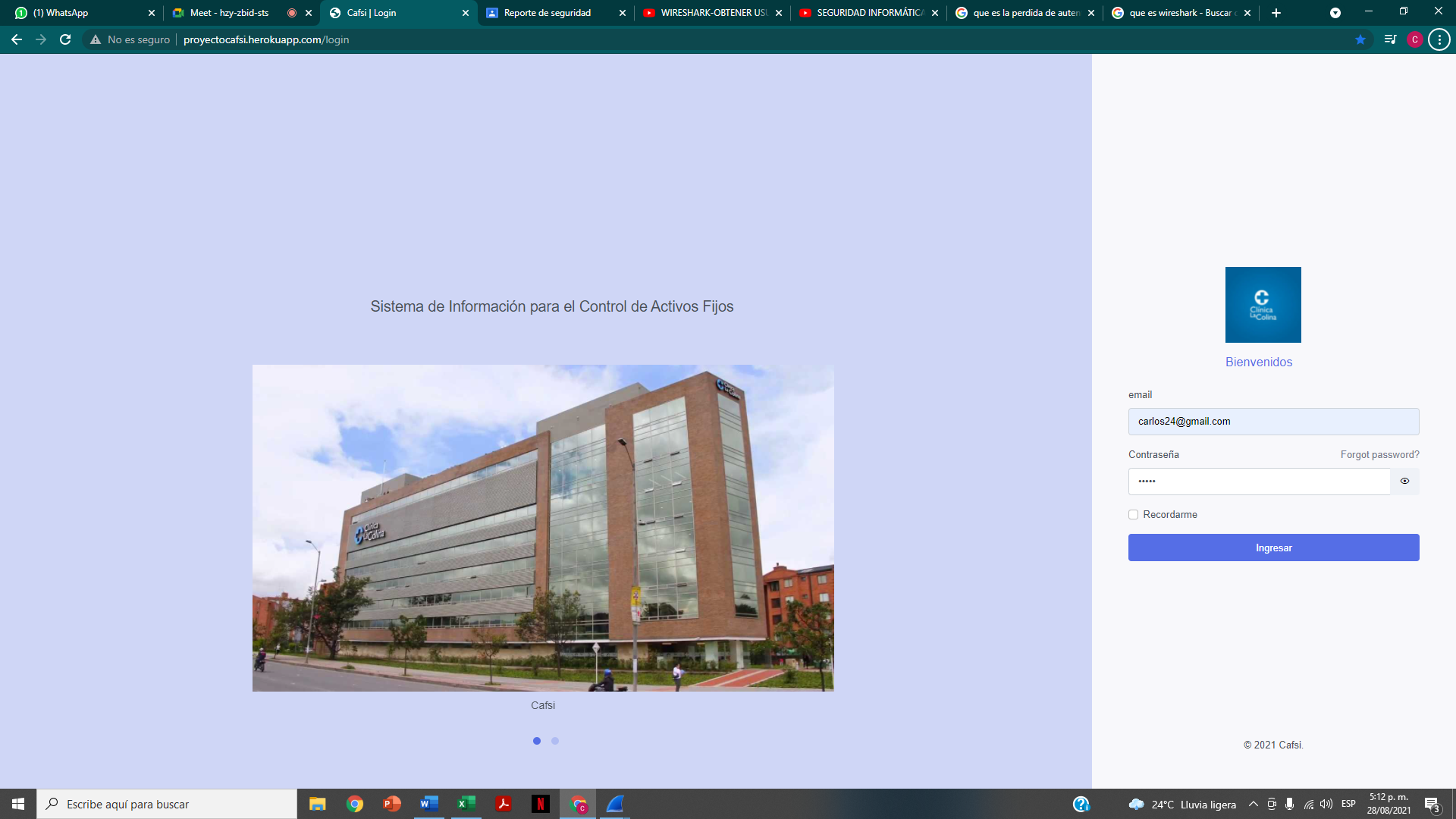
**Exposición de datos sensibles:**

1. descargar Wireshark y ejecutarlo.

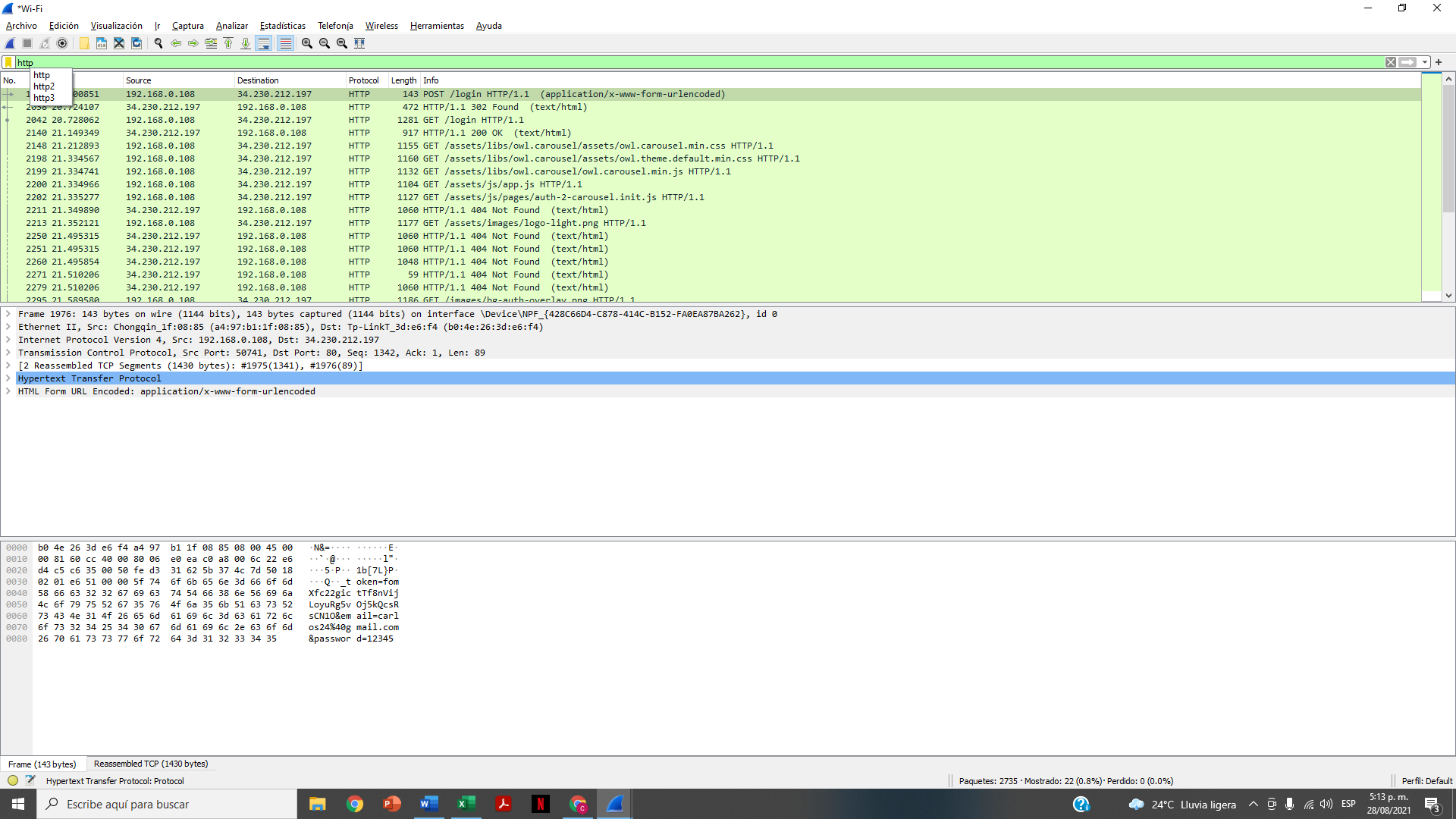
2. Como estamos realizando la red inalámbrica, elegimos la red WI-FI para que capture toda la información de la tarjeta de red inalámbrica.



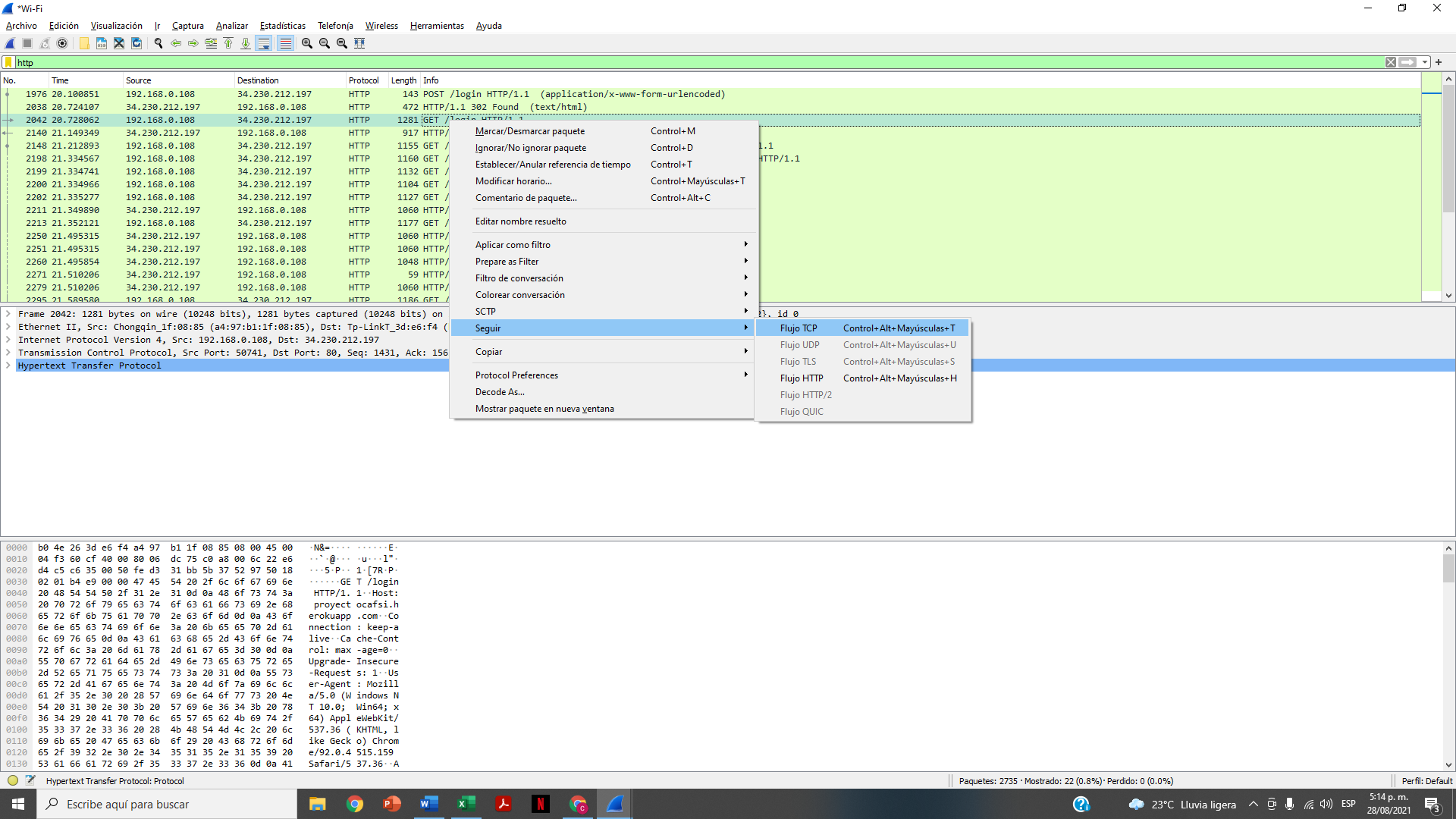
3. Ingresamos al sitio web y digitamos un usuario y una contraseña([carlos24@gmail.com-12345](mailto:carlos24@gmail.com-12345)) e ingresamos.



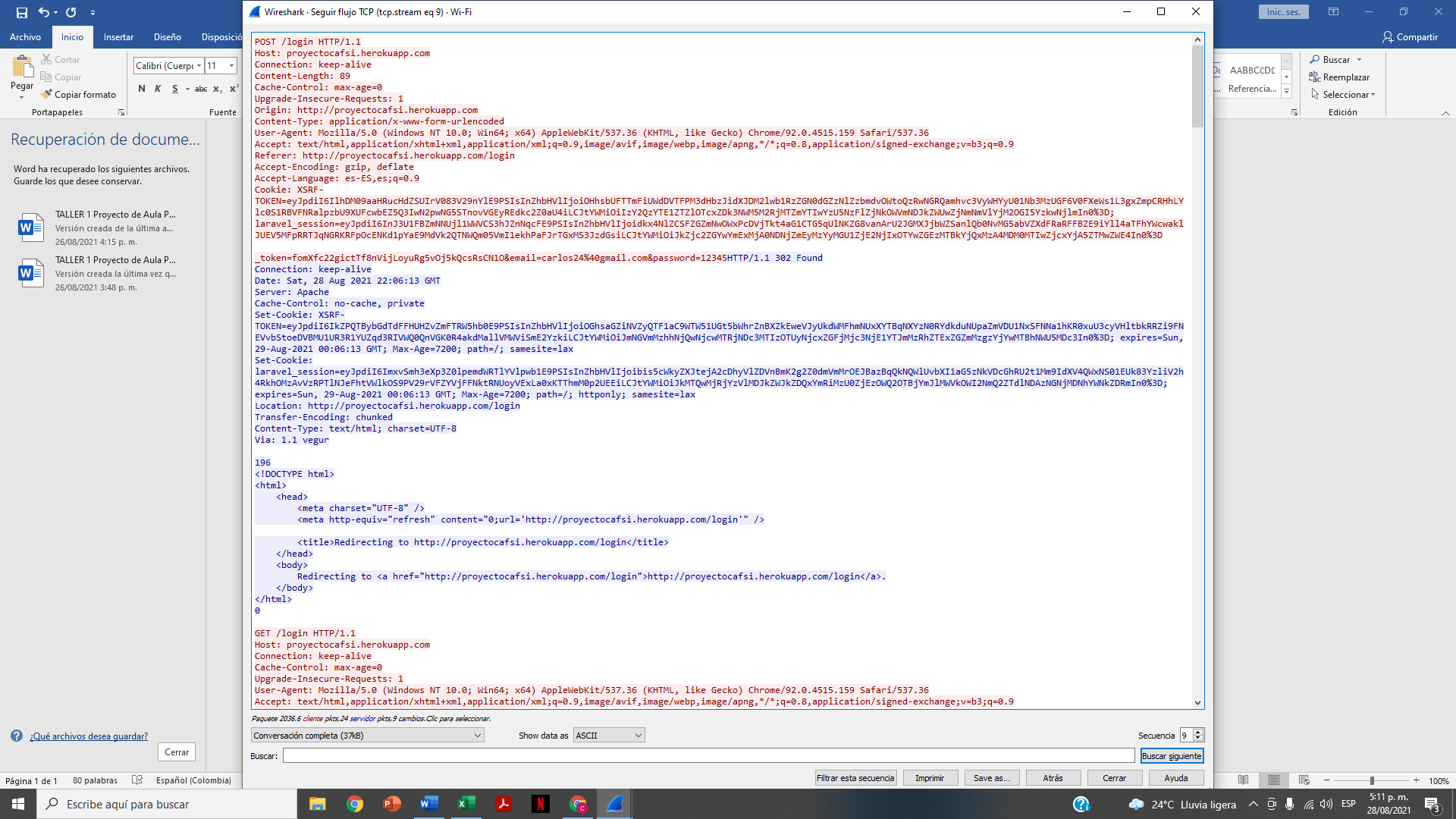
4. Volvemos a Wireshark paramos el tráfico, en el filtro escribimos http para que nos muestre solo estos paquetes.

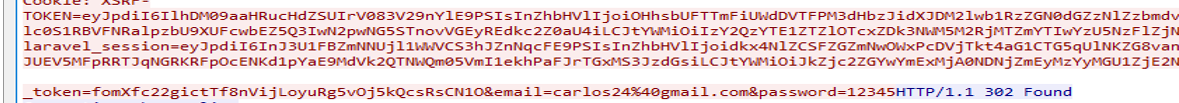


5. elegimos el sitio web que corresponde, le damos clic derecho, seguir-flujo TCP



6. Podremos visualizar la información digitada([carlos24@gmail.com-12345](mailto:carlos24@gmail.com-12345)).

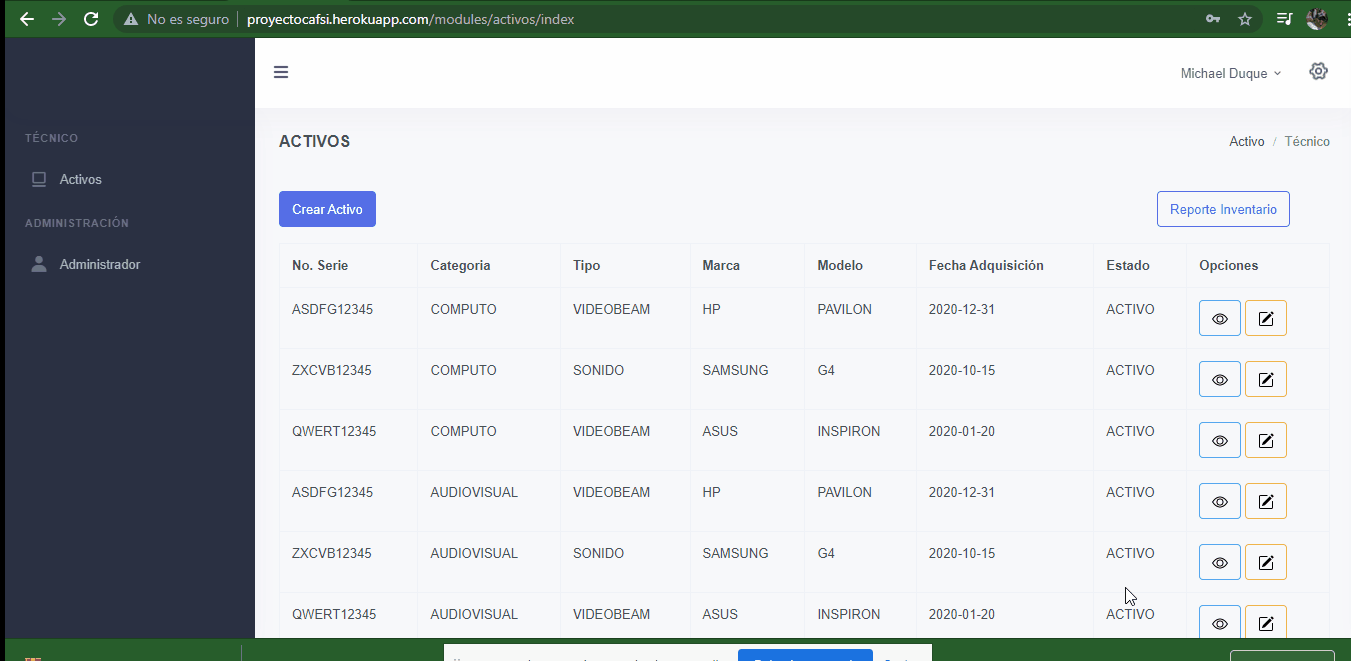




Para evitar este tipo de vulnerabilidad se debe instalar el certificado SSL en el servidor web donde esta alojada la aplicación y así enviar la información encriptada.

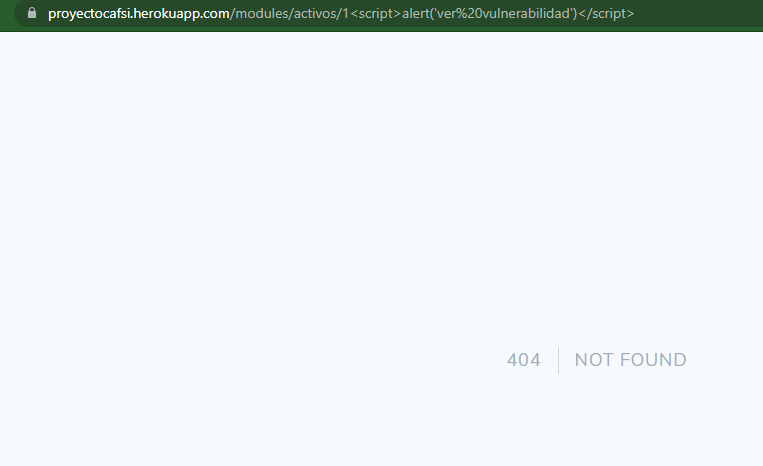
**Pérdida de autenticación**

Se valida la variable de cierre de sesión por inactividad y se encontró que el sistema tiene una habilitación de sesión de 120 minutos lo que daba una vulnerabilidad de la autenticación del usuario al sistema con lo cual se disminuye el tiempo de inactividad para cierre de sesión a 3 minutos así previniendo esta inseguridad.



**Secuencia de comandos en sitios cruzados**

Se valida a través de la consulta la ejecución de un código en JS para ver si el sistema permite ejecutar script desde el envió de petición de consulta, tal como se observa a continuación:



Y se observa que al querer ejecutar el comando de la siguiente manera no permite ejecutar el código en script JS lo que evidencia que el sistema está seguro para la ejecución de script Js desde él envió de petición hacia el servidor donde está alojado el aplicativo.

Una vez realizadas todas estas pruebas y ajustes garantizamos que el sistema es adecuadamente seguro para trabajar en un ambiente de producción.

En ambiente de producción el proveedor **Dongee,** nos garantiza que el aplicativo contara con certificado SSL para no exponer datos desencriptados y que las peticiones se envíen de manera segura, de igual manera el servidor donde se desplegara el aplicativo cuenta con antivirus y “liteSpeed”, el cual también garantiza que este óptimamente seguro ante cualquier pentesting o ataque cibernético que quieran hacer a este sistema.

# TESTEABILIDAD (FACILIDAD DE PROBAR EL SISTEMA).

Es la capacidad o facilidad que tiene un sistema para realizarle pruebas tanto de caja negra (pruebas de código fuente), como de caja blanca prueba a funcionalidad del sistema.

En nuestro sistema aseguramos que las pruebas de usuario se puedan realizar de manera bastante sencilla ya que este no contiene tanta funcionalidad que produzca complejidad en su “testeabilidad”, apoyado con el manual de usuario permite garantizar una correcta ejecución de pruebas para el usuario final

La testeabilidad realizada a este aplicativo técnicamente se centró en la marcha de la ejecución del código y en los escenarios de pruebas digitadas por dos usuarios donde se plasmo cada ítem a probar según las acciones que permitiera realizar cada entidad o modelo que se visualiza en el sistema funcionalmente.

# USABILIDAD

Es la capacidad que tiene un sistema para satisfacer la necesidad planteada por el usuario a utilizar el sistema. Es decir, debe garantizar la adecuada experiencia de usuario en su manejo.

En el aplicativo se mostro ya a usuarios finales los cuales realizaron un acceso tanto con el rol administrador como con el rol auxiliar apoyados del manual de usuario donde evidenciaron su sencillo e intuitivo manejo del aplicativo y se centra en lo requerido por el usuario, garantizando así su correcto funcionamiento

También, se aplicó la técnica de seguimiento ocular y clasificación; el cual arrojo un resultado satisfactorio, dado que los participantes manifestaron su agrado con la interfaz.

# CONCLUSIÓN

Los atributos de calidad de software me brindan la capacidad de comprender que también desarrollado y diseñado esta un software y da la capacidad de analizar y evaluar que condiciones o falencias tiene el sistema para ser ajustadas y así permita garantizar un óptimo manejo del sistema.

En este aplicativo se aplicaron principalmente estos atributos para así brindarle al usuario, una interfaz sencilla e intuitiva la cual satisface la necesidad exigida por el cliente.