Análisis dinámico de seguridad de la aplicación web “Buscador Rick y Morty”

Versión 1.0.0

01/2025

Pedro Garvich

Proyecto: Mini SSDLC aplicado a sistemas web

Índice

Pedro Garvich

1. Resumen Ejecutivo

El presente informe documenta el análisis dinámico de seguridad (DAST) realizado sobre la aplicación web “Buscador Rick y Morty”, en el marco de un ejercicio práctico de Mini SSDLC aplicado a sistemas web. El análisis se complementó con la correlación de hallazgos previamente identificados mediante análisis estático (SAST).

El alcance incluyó la evaluación de endpoints públicos y autenticados, con foco en autenticación, manejo de sesiones, validación de entradas y exposición de información sensible. Las pruebas se realizaron utilizando Burp Suite como herramienta principal, junto con validaciones manuales para confirmar o descartar hallazgos automáticos.

Como resultado, se identificó una vulnerabilidad de alta severidad relacionada con la transmisión de credenciales por canales no cifrados, confirmada dinámicamente. Asimismo, se detectaron debilidades de diseño y configuración, algunas de las cuales no resultaron explotables en tiempo de ejecución, pero fueron validadas conceptualmente mediante SAST.

El análisis cruzado SAST–DAST permitió distinguir entre vulnerabilidades explotables, riesgos teóricos y falsos positivos, mejorando la priorización y trazabilidad de los hallazgos. En conjunto, los resultados evidencian la importancia de integrar análisis estático y dinámico dentro de un proceso SSDLC, incluso en aplicaciones de alcance reducido y con fines educativos.

2. Tabla de vulnerabilidades

ID	Tipo	Descripción	Severidad	Confianza	Validación DAST	Estado
DAST-01	Vulnerabilidad	Transmisión de credenciales por canal no cifrado (HTTP)	Alta	Alta	Confirmada	Abierta
DAST-02	Posible vuln.	Client-side Desync detectado por scanner automático	Alta	Baja	No confirmada (FP probable)	Mitigable por arquitectura
SEC-01	Configuración	Exposición de SECRET_KEY de Django	Crítica	Alta	No aplicable	Abierta
SH-01	Diseño	Endpoints de escritura aceptan GET y POST	Media	Media	Confirmada (Mitigada por middleware)	Cerrada - Riesgo aceptado
SH-02	Configuración	DEBUG habilitado en entorno no productivo	Baja	Media	No observada en DAST	Corregida
SG-01	Código	Uso de datos de request.POST sin sanitización explícita	Media	Media	No explotable	Abierta (preventiva)

3. Mapeo de la aplicación y metodología del análisis desde la perspectiva DAST

3.1 Alcance del análisis dinámico

El análisis dinámico de seguridad (DAST) se realizó sobre una aplicación web de alcance reducido, basada en Django, orientada a la gestión y visualización de contenido relacionado con la serie *Rick and Morty*.

El objetivo de esta fase fue identificar vulnerabilidades explotables en tiempo de ejecución, validando el comportamiento real de la aplicación frente a entradas controladas por el usuario y escenarios de ataque típicos a nivel HTTP.

El análisis se llevó a cabo utilizando **Burp Suite** como proxy de interceptación y scanner activo, complementado con validaciones manuales para confirmar o descartar los hallazgos automáticos.

Las pruebas se realizaron bajo un enfoque black-box, priorizando endpoints y flujos previamente identificados como relevantes a partir de los hallazgos del análisis estático (SAST).

3.2 Superficie de ataque identificada

A partir de la enumeración de endpoints y del análisis de flujos de navegación, se identificaron los siguientes puntos de exposición:

Endpoints accesibles **sin autenticación:**

- /
- /home/
- /members/
- /buscar/ (GET / POST)
- /accounts/login/ (GET / POST)
- /register/ (GET / POST)

Estos endpoints procesan:

- credenciales de usuario,
- datos ingresados por el cliente (formularios),
- parámetros de búsqueda.

Por lo tanto, constituyen la principal superficie de ataque externa.

Endpoints que **requieren autenticación:**

- /exit/
- /favourites/
- /favourites/add/ (GET / POST)
- /favourites/delete/ (GET / POST)

Estos endpoints permiten acciones sobre datos asociados a la cuenta del usuario autenticado, por lo que fueron considerados relevantes para la evaluación de:

- control de sesión,
- autorización,
- manipulación de estados.

Recursos estáticos:

- /static/Audio/rickandmorty.mp3

Este recurso fue identificado como archivo estático sin lógica de negocio asociada, por lo que su análisis se limitó a verificar su exposición directa y no fue considerado crítico desde el punto de vista de seguridad.

3.3 Consideraciones de entorno

La aplicación se encontraba desplegada en un entorno de desarrollo, utilizando el servidor integrado de Django (WSGIServer/0.2 CPython/3.14.0).
Este contexto es relevante para la interpretación de ciertos hallazgos automáticos, ya que dicho servidor no está diseñado para escenarios de producción ni para el manejo robusto de conexiones persistentes ante solicitudes malformadas.

Por este motivo, los hallazgos detectados fueron:

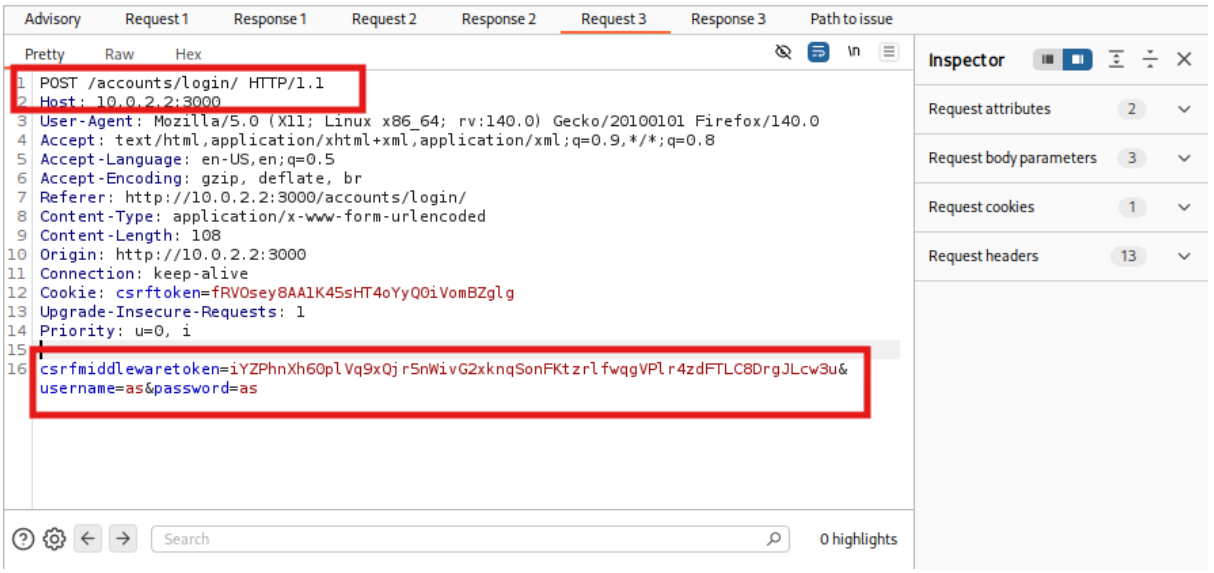
- validados manualmente cuando fue posible,
- contextualizados según el entorno,
- clasificados cuidadosamente para evitar falsos positivos o conclusiones exageradas.

4. Detalle de hallazgos

DAST-01: Transmisión de credenciales por canal no cifrado (Cleartext Submission)

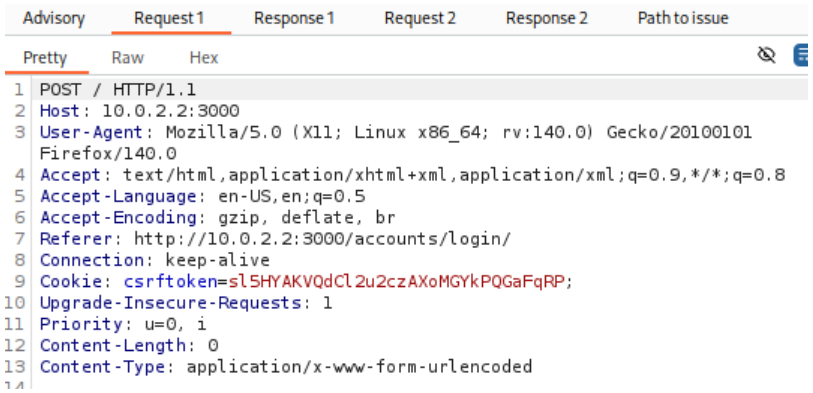
- **Severidad:** Alta
- **Confianza:** Alta
- **Endpoints afectados:** /accounts/login/, /register/, / (form de inicio).
- **Descripción:** La aplicación permite el envío de formularios de autenticación a través de HTTP simple. Esto expone las credenciales a ataques de interceptación (*Man-in-the-Middle*) y habilita escenarios de *Information Disclosure* y *Spoofing*, permitiendo el secuestro de credenciales y la suplantación de identidad de usuarios legítimos.
- **Validación:** Confirmado mediante interceptación de tráfico con Burp Proxy, donde los campos password, password1 y password2 son legibles en el cuerpo de la solicitud POST.
- **Remediación recomendada:**
    - 1. Implementar **TLS/SSL** en el servidor.
    - 2. Configurar en Django SECURE_SSL_REDIRECT = True para forzar HTTPS.
    - 3. Configurar SESSION_COOKIE_SECURE = True y CSRF_COOKIE_SECURE = True.

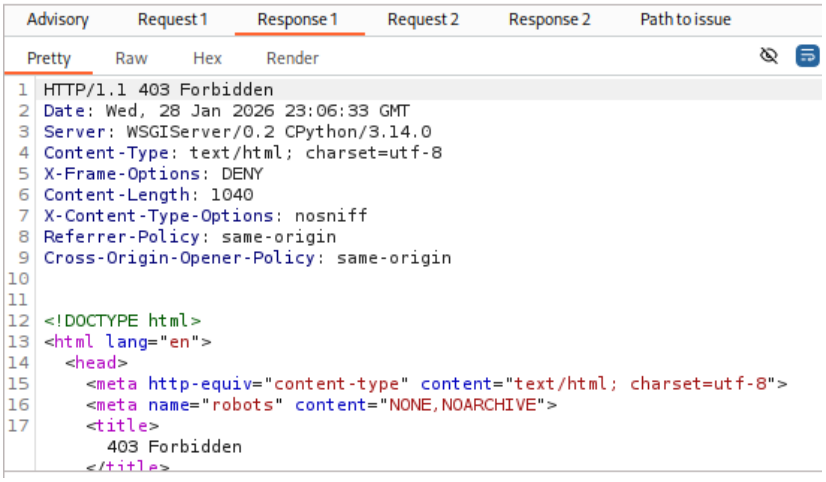
- **Evidencia del hallazgo:**

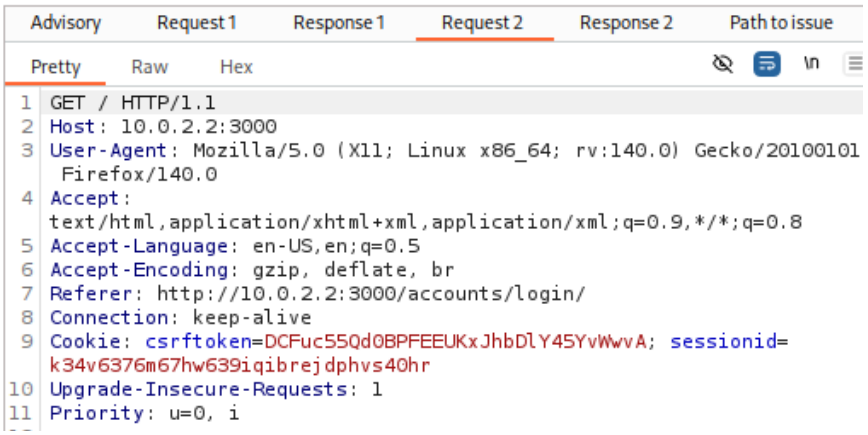


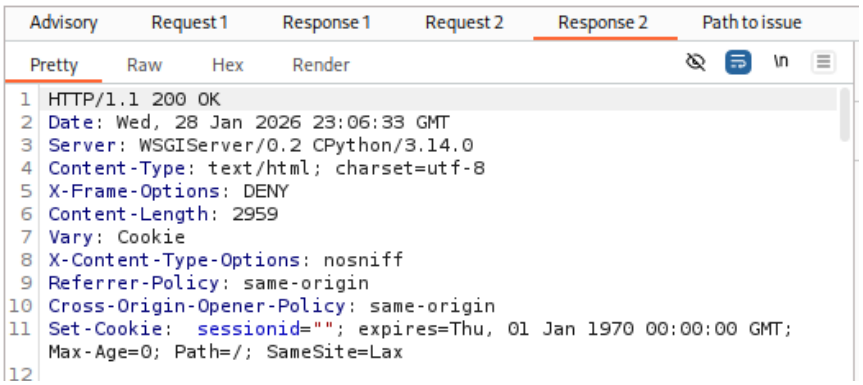
DAST-02: Posible Client-side Desync (CSD)

- **Severidad:** Alta
- **Confianza:** Baja (posible falso positivo)
- **Descripción:** El scanner automático detectó una discrepancia en cómo el servidor procesa el Content-Length en los endpoints / y /favourites/.
- **Validación:** Se sospecha que este hallazgo es un **Falso Positivo** derivado del comportamiento del servidor de desarrollo WSGIServer/0.2 CPython/3.14.0. Este tipo de servidores no están optimizados para el manejo de conexiones persistentes ante solicitudes malformadas, lo que genera una respuesta prematura que Burp interpreta como desincronización.
- **Recomendación:** En caso de pasar a producción, se debe utilizar un servidor de aplicaciones de grado industrial (Gunicorn/Uvicorn) detrás de un proxy inverso (Nginx) correctamente configurado para evitar ataques de HTTP Request Smuggling.
- **Evidencia:**




Advisory
Request 1
Response 1
Request 2
Response 2
Path to issue
Pretty
Raw
Hex
Render
HTTP/1.1 403 Forbidden
Date: Wed, 28 Jan 2026 23:06:33 GMT
Server: WSGIServer/0.2 CPython/3.14.0
Content-Type: text/html; charset=utf-8
X-Frame-Options: DENY
Content-Length: 1040
X-Content-Type-Options: nosniff
Referrer-Policy: same-origin
Cross-Origin-Opener-Policy: same-origin
<!DOCTYPE html>
<html lang="en">
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8">
<meta name="robots" content="NONE,NOARCHIVE">
<title>
403 Forbidden


Advisory
Request 1
Response 1
Request 2
Response 2
Path to issue
Pretty
Raw
Hex
GET / HTTP/1.1
Host: 10.0.2.2:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:140.0) Gecko/20100101 Firefox/140.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://10.0.2.2:3000/accounts/login/
Connection: keep-alive
Cookie: csrftoken=DCFuc55Qd0BPFEEUKxJhbDlY45YvWwvA; sessionid=k34v6376m67hw639iqibrejdphvs40hr
Upgrade-Insecure-Requests: 1
Priority: u=0, i


Advisory
Request 1
Response 1
Request 2
Response 2
Path to issue
Pretty
Raw
Hex
Render
HTTP/1.1 200 OK
Date: Wed, 28 Jan 2026 23:06:33 GMT
Server: WSGIServer/0.2 CPython/3.14.0
Content-Type: text/html; charset=utf-8
X-Frame-Options: DENY
Content-Length: 2959
Vary: Cookie
X-Content-Type-Options: nosniff
Referrer-Policy: same-origin
Cross-Origin-Opener-Policy: same-origin
Set-Cookie: sessionid=""; expires=Thu, 01 Jan 1970 00:00:00 GMT; Max-Age=0; Path=/; SameSite=Lax

5. Correlación y validación de hallazgos SAST mediante DAST

Esta sección tiene como objetivo contrastar los hallazgos identificados durante el análisis estático (SAST) con el comportamiento observado durante el análisis dinámico (DAST), validando su impacto real en tiempo de ejecución y su explotabilidad práctica.

La correlación SAST–DAST permite:

- confirmar vulnerabilidades efectivamente explotables,
- identificar falsos positivos o riesgos teóricos,
- y reforzar la trazabilidad dentro del enfoque SSDLC adoptado.

SEC-01 – Exposición de clave secreta de Django (SECRET_KEY)

Resultado SAST:
Vulnerabilidad crítica confirmada (Blocker).

Validación DAST:
No directamente validable mediante DAST.

Análisis:

La exposición de la SECRET_KEY constituye una vulnerabilidad de configuración y gestión de secretos, cuyo impacto se manifiesta a nivel criptográfico interno del framework (firmado de cookies, tokens CSRF, etc.).

Este tipo de vulnerabilidad **no es detectable ni explotable de forma directa mediante pruebas dinámicas**, ya que no depende de un endpoint específico ni de entradas controladas por el usuario.

No obstante, el impacto potencial identificado en SAST es consistente con riesgos reales:

- forja de cookies de sesión,
- bypass de protecciones CSRF,
- compromiso de mecanismos de autenticación.

Conclusión:

Hallazgo **válido, crítico y correctamente identificado por SAST**.

DAST no aplica como mecanismo de detección, lo cual refuerza la necesidad de análisis estático en etapas tempranas del SSDLC.

SH-01 – Métodos HTTP seguros y no seguros en vistas Django

Resultado SAST: Security Hotspot – severidad media.

Validación DAST: Confirmada y mitigada.

Pedro Garvich

Análisis:

Se realizó una prueba de validación manual sobre el endpoint /favourites/add/ utilizando Burp Repeater para verificar la efectividad de la protección contra falsificación de solicitudes entre sitios (CSRF). Se ejecutaron tres escenarios de ataque:

1. Eliminación completa del parámetro csrfmiddlewaretoken.
2. Envío del parámetro con valor nulo.
3. Envío de un token alterado/inválido.

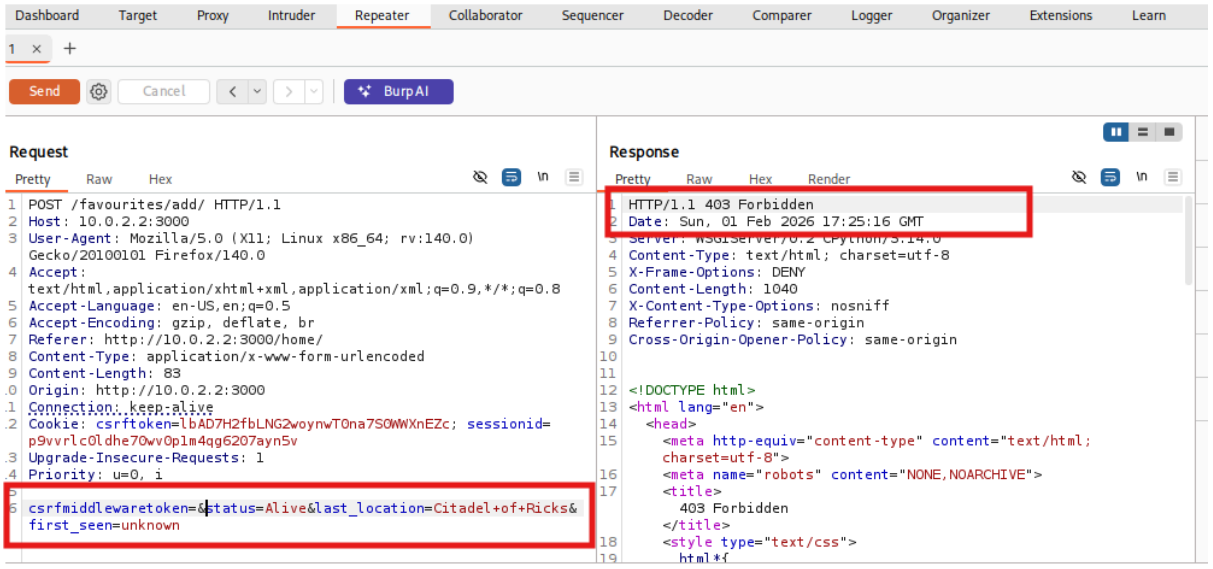
Resultado del Test:

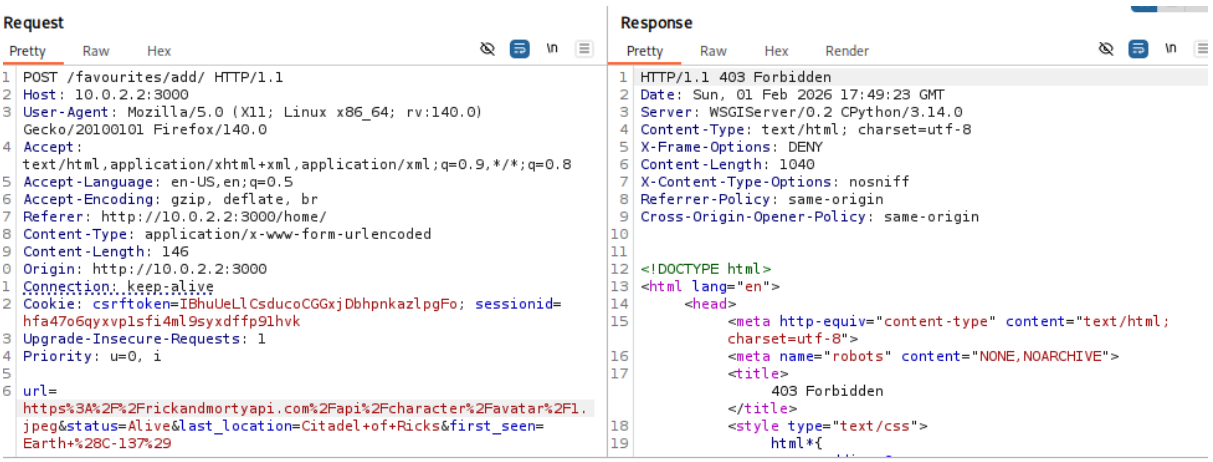
En los tres casos, el servidor respondió con un código **HTTP 403 Forbidden**, bloqueando la acción de escritura.

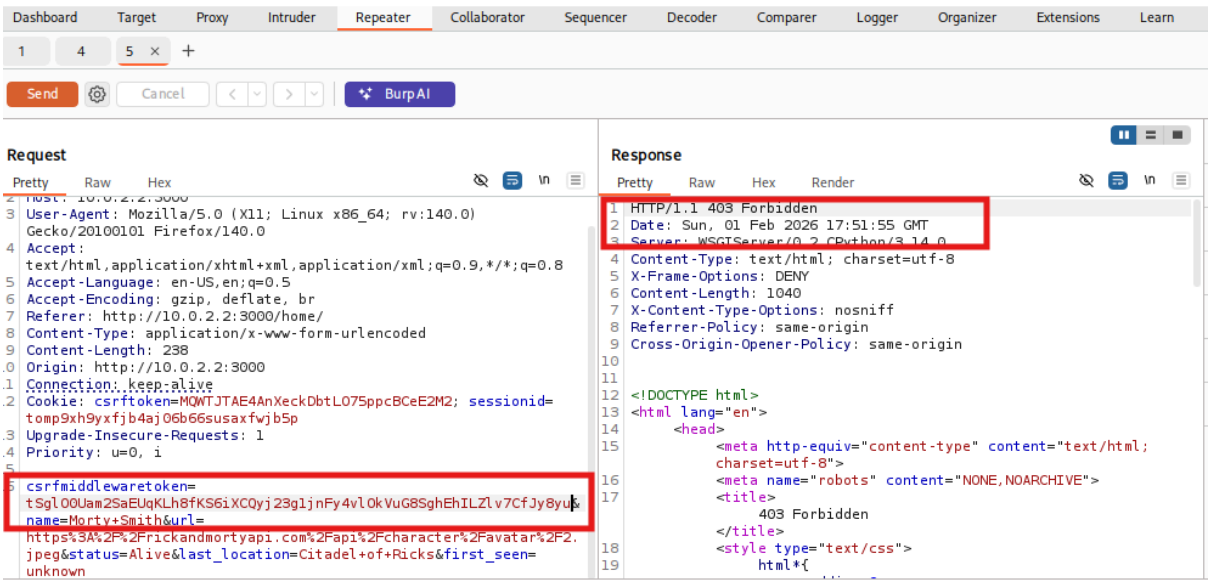
Conclusión:

Si bien el SAST identifica correctamente la debilidad de diseño (aceptar GET en rutas de escritura), la explotación práctica de CSRF está **mitigada con éxito** por el middleware global de Django. El hallazgo se reclasifica como una "Mejora de Código" (Hardening) y no como una vulnerabilidad crítica explotable.

Evidencia del test:







En este último caso se reemplazó el csrfmiddlewaretoken por el de la sesión de otro usuario.

SH-02 – Configuración DEBUG habilitada

Resultado SAST: Security Hotspot – severidad baja.

Validación DAST: No validado dinámicamente.

Análisis: Durante la fase DAST, la configuración DEBUG = True fue desactivada previamente por motivos operativos, con el fin de:

- evitar ruido en las respuestas,
- simular un entorno más cercano a producción,
- y facilitar el análisis de comportamiento real de la aplicación.

Por este motivo, no se observaron:

- trazas de error detalladas,
- exposición de información interna en respuestas HTTP.

Conclusión:
El hallazgo es **válido desde el punto de vista de configuración**, pero su impacto no se manifestó durante el análisis dinámico debido a la corrección previa aplicada.
El riesgo se considera **aceptado en el contexto educativo**, con recomendación clara de desactivación antes de cualquier despliegue productivo.

(Este caso ilustra cómo DAST puede confirmar la efectividad de una remediación aplicada entre fases del SSDLC.)

SG-01 – Uso de datos sin sanitizar desde request.POST

Resultado SAST: Hallazgo de severidad media confirmado por Semgrep.

Validación DAST: No explotable en el contexto actual.

Análisis:
Durante pruebas dinámicas con entradas maliciosas controladas:

- no se observaron inyecciones SQL,
- no se evidenció XSS reflejado ni persistente,
- ni comportamientos inesperados en el procesamiento de formularios.

Esto se debe a:

- validaciones estrictas del UserCreationForm,
- uso de Django ORM,
- escape automático del motor de templates.

Conclusión:
El hallazgo representa una **mala práctica real**, correctamente identificada por SAST, pero **no se traduce en una vulnerabilidad explotable en runtime** en el estado actual de la aplicación.
Se mantiene la recomendación de remediación para prevenir riesgos futuros ante cambios de contexto.

Relación general SAST–DAST

El análisis cruzado evidencia que:

- SAST es clave para detectar **riesgos estructurales y de configuración** no observables en runtime.
- DAST permite confirmar **impacto real y explotabilidad práctica** de debilidades identificadas.

- La combinación de ambos enfoques reduce falsos positivos y mejora la priorización de riesgos.

Este ejercicio refuerza la importancia de integrar análisis estático y dinámico dentro de un proceso SSDLC, incluso en proyectos de alcance reducido y con fines educativos.

6. Conclusión General del Ciclo SSDLC

El análisis integral de la aplicación "Buscador Rick y Morty" demuestra que la seguridad no es un evento único, sino un proceso de capas. La implementación de un **Mini SSDLC** permitió identificar riesgos que habrían pasado inadvertidos si se hubiera optado por un único método de evaluación.

- **Complementariedad Técnica:** Mientras que el análisis **SAST** fue fundamental para detectar malas prácticas de configuración y gestión de secretos (como la exposición de la `SECRET_KEY`), el análisis **DAST** permitió descartar falsos positivos de lógica (CSRF) y confirmar vulnerabilidades críticas de infraestructura (transmisión en texto claro).
- **Criterio Analítico vs. Automatización:** Este ejercicio pone de relieve que las herramientas automáticas son puntos de partida, no de llegada. La capacidad de discernir entre un hallazgo real y una limitación del entorno de desarrollo (como en el caso del *Client-side Desync*) es lo que garantiza un uso eficiente de los recursos de remediación.
- **Impacto en el Negocio/Proyecto:** Aún en una aplicación de escala reducida, los hallazgos confirman que la configuración por defecto de los frameworks (Django) provee una base sólida, pero la intervención humana es indispensable para asegurar el manejo de datos sensibles y el hardening del entorno productivo.

En conclusión, este proyecto cumple con el objetivo de profesionalizar el desarrollo, transformando un código funcional en una aplicación consciente de su postura de seguridad, lista para ser robustecida y desplegada bajo estándares modernos de la industria.