



---

# INFORME DE AUDITORIA DE SEGURIDAD WEB

---

Golden Player – Evaluación de Seguridad



13 DE ENERO DEL 2026  
CARLOS FELIPE VALBUENA CORREDOR  
Auditor

## Contenido

1. Introducción .....	2
Objetivo general .....	2
Objetivos específicos .....	2
2. Resumen .....	2
3. Alcance .....	3
4. Descripción del entorno .....	3
5. Metodología de auditoría .....	3
6. Herramientas utilizadas .....	3
7. Vulnerabilidades identificadas .....	4
7.1 Broken Access Control .....	4
7.2 Inyección SQL – MySQL.....	4
7.3 XSS Almacenado en gestión de campeonatos .....	5
7.4 Cross-Site Scripting (XSS) almacenado en múltiples funcionalidades .....	6
7.5 Ausencia de Tokens Anti-CSRF .....	6
7.6 Manipulación de Parámetros (Parameter Tampering) .....	7
7.7 Cabeceras de Seguridad Inseguras .....	7
7.8 Cookies Inseguras .....	8
8. Evidencias.....	9
8.1 Evidencias de reconocimiento .....	9
8.2 Evidencias de Control de acceso roto .....	10
8.3 Evidencias de Inyeccion SQL.....	11
8.4 Evidencias de XSS almacenado en gestión de campeonatos .....	12
8.5 Evidencias Cross-Site Scripting (XSS) almacenado en múltiples funcionalidades .....	13
8.6 Evidencias Ausencia de Tokens Anti-CSRF.....	13
8.7 Evidencias Manipulación de Parámetros (Parameter Tampering).....	15
8.8 Evidencias Cabeceras de Seguridad Inseguras.....	16
8.9 Evidencias Cookies Inseguras.....	17
9. Recomendaciones generales .....	17
10. Conclusiones .....	18
11. Anexos .....	18

## 1. Introducción

El presente informe documenta la auditoría de seguridad realizada a la aplicación web *Golden Player*, desarrollada en PHP y MySQL y desplegada mediante Docker. El objetivo es identificar posibles vulnerabilidades, evaluar los controles existentes y proponer mejoras que fortalezcan la seguridad de la aplicación.

### Objetivo general

Evaluar el nivel de seguridad de la aplicación web **Golden Player**, identificando vulnerabilidades y debilidades técnicas, con el fin de proponer medidas de mitigación que fortalezcan la protección de la información y el correcto funcionamiento del sistema.

### Objetivos específicos

1. Identificar vulnerabilidades comunes en la aplicación mediante pruebas de seguridad basadas en OWASP.
2. Analizar los mecanismos de autenticación, gestión de sesiones y control de accesos implementados.
3. Proponer recomendaciones técnicas para mitigar los riesgos detectados y mejorar la postura de seguridad de la aplicación.

## 2. Resumen

Durante la auditoría de seguridad realizada a la aplicación web, se identificaron múltiples vulnerabilidades que podrían comprometer la confidencialidad, integridad y disponibilidad de la información. Entre los hallazgos más relevantes se encuentran vulnerabilidades de Inyección SQL, ausencia de protección contra CSRF y configuraciones inseguras en cabeceras HTTP.

Estas debilidades podrían permitir accesos no autorizados, manipulación de datos y ataques contra usuarios autenticados.

### **3. Alcance**

- **URL Evaluada:** http://localhost:8080/
- **Módulos:**
  - Login
  - Perfil de Usuario
  - Home
  - Crear cuenta
  - Torneo
  - Perfil de Administrador
- **Tipos de prueba:**
  - Enumeración de directorios
  - Escaneo automático
  - Escaneo pasivo
- **Exclusiones:**
  - Infraestructura externa
  - Ataques de denegación de servicio (DoS)

### **4. Descripción del entorno**

Describe el entorno evaluado:

- Sistema operativo
- Tecnologías utilizadas
- Arquitectura Docker
- Puerto y método de acceso

### **5. Metodología de auditoría**

1. Enumeración de rutas
2. Escaneo pasivo
3. Escaneo activo controlado
4. Análisis manual de hallazgos

### **6. Herramientas utilizadas**

- Gobuster
- OWASP ZAP
- WFUZZ
- Burp Suite
- Nmap

## **7. Vulnerabilidades identificadas**

### **7.1 Broken Access Control**

**Severidad:** Alta

**Descripción:**

La aplicación permite que usuarios con rol estándar accedan directamente a rutas administrativas mediante la manipulación de la URL, sin validar el rol del usuario en el backend.

**Impacto:**

- Escalada de privilegios
- Acceso a información sensible
- Modificación de datos críticos

**Recomendación:**

- Validar roles y permisos en cada endpoint
- Implementar control de acceso basado en sesión
- Restringir rutas administrativas en backend

### **7.2 Inyección SQL – MySQL**

**Severidad:** Alta

**Herramienta:** OWASP ZAP

**Descripción:**

Se identificaron parámetros vulnerables a inyección SQL que podrían permitir a un atacante manipular consultas a la base de datos. La expresión regular del mensaje de error que aparece [You have an error in your SQL syntax] corresponde con los resultados HTML. la vulnerabilidad fue detectada por la manipulación del parámetro para causar un mensaje de error de base de datos a ser retornado y reconocido.

**Impacto:**

- Acceso no autorizado
- Robo o modificación de información
- Compromiso total de la base de datos

**Evidencia:**

- Link: <http://localhost:8080/controlador/usuarios/regarUuario.php>
- Link: <http://localhost:8080/controlador/usuarios/cambiarUuario.php>
- Parámetros marcados como vulnerables

### **Recomendación:**

- No confíe en los datos de entrada del lado del cliente, incluso si existe una validación del lado del cliente.
- Como norma general, escriba la verificación de los datos en el lado del servidor.
- Escape todos los datos recibidos del cliente.
- Aplique una 'lista de permitidos' para caracteres permitidos o una 'lista de denegados' para caracteres no permitidos en la entrada del usuario.
- Aplique el principio de privilegio mínimo utilizando el usuario de base de datos con el menor privilegio posible.
- En particular, evite utilizar usuarios de bases de datos 'sa' o 'db-owner'. Esto no elimina la inyección SQL, pero minimiza su impacto.
- Otorgue el acceso mínimo a la base de datos que sea necesario para la aplicación.

## **7.3 XSS Almacenado en gestión de campeonatos**

**Severidad:** Alta

### **Descripción:**

Se identificó una vulnerabilidad de Cross-Site Scripting (XSS) almacenado en el campo de nombre de equipo dentro de la sección de campeonatos. La aplicación permite almacenar código JavaScript sin sanitización adecuada, el cual se ejecuta cuando un usuario con rol administrador accede a la vista correspondiente.

### **Impacto:**

- Ejecución de código JavaScript en el navegador del administrador
- Robo de sesión administrativa
- Escalada de privilegios
- Compromiso total de la aplicación

### **Prueba de Concepto:**

1. Acceder a la funcionalidad de creación/edición de equipos
2. Ingresar el siguiente payload en el nombre del equipo:
3. `<script>alert(1)</script>`
4. Guardar el registro
5. Acceder a la vista de campeonatos como administrador
6. Se ejecuta el script en el navegador

**Recomendación:**

- Sanitizar y validar todas las entradas en el backend
- Escapar correctamente la salida (output encoding)
- Implementar Content-Security-Policy (CSP)
- Validar campos por tipo y longitud

**7.4 Cross-Site Scripting (XSS) almacenado en múltiples funcionalidades****Descripción:**

Se identificó una vulnerabilidad de Cross-Site Scripting (XSS) almacenado en distintos formularios de la aplicación, incluyendo la creación de equipos y el registro de reservas. La aplicación permite almacenar código JavaScript sin sanitización adecuada, el cual se ejecuta cuando los datos son renderizados en vistas administrativas.

**Impacto:**

- Ejecución de código JavaScript arbitrario
- Robo de sesión
- Escalada de privilegios
- Compromiso del panel administrativo

**Evidencia:**

- Figura X: XSS en nombre de equipo
- Figura Y: XSS en formulario de reserva

**7.5 Ausencia de Tokens Anti-CSRF**

**Severidad:** Media-Alta

**Descripción:**

No se encontraron tokens Anti-CSRF en un formulario de envío HTML.

Una solicitud falsa entre sitios en un ataque que compromete y obliga a una víctima a enviar su solicitud HTTP a un destino objetivo sin su conocimiento o intención para poder realizar una acción como víctima. La causa oculta es la funcionalidad de la aplicación utilizando acciones de URL/formulario que pueden ser adivinados de forma repetible. La naturaleza del ataque es que CSRG explota la confianza que un sitio

url: <http://localhost:8080/crearCuenta.php>

url: <http://localhost:8080/login.php>

**Impacto:**

- Ejecución de acciones no autorizadas
- Riesgo para usuarios autenticados

**Recomendación:**

- Implementar tokens CSRF únicos por sesión
- Validar el token en cada solicitud POST

**7.6 Manipulación de Parámetros (Parameter Tampering)**

**Severidad:** Media

**Descripción:**

Se detectó la posibilidad de modificar parámetros enviados desde el cliente sin validación adecuada en el servidor. En este caso permite el registro del usuario sin validar la información, cualquier atacante puede manipularlo antes de que se envíe al servidor

**Impacto:**

- Acceso a recursos de otros usuarios (IDOR)
- Manipulación de información

**Recomendación:**

- Validar permisos en backend
- No confiar en datos del cliente

**7.7 Cabeceras de Seguridad Inseguras**

**Severidad:** Media

**Hallazgos:**

La Política de seguridad de contenido (CSP) es una capa adicional de seguridad que ayuda a detectar y mitigar ciertos tipos de ataques, incluidos Cross Site Scripting (XSS) y ataques de inyección de datos. Estos ataques se utilizan para todo, desde el robo de datos hasta la desfiguración del sitio o la distribución de malware. CSP proporciona un conjunto de encabezados HTTP estándar que permiten a los propietarios de sitios web declarar fuentes de contenido aprobadas que los navegadores deberían poder cargar en esa página; los tipos cubiertos son JavaScript, CSS, marcos HTML, fuentes, imágenes y objetos incrustados como applets de Java, ActiveX, archivos de audio y video.



- CSP no configurada
- Falta de X-Frame-Options
- X-Content-Type-Options ausente

**Impacto:**

- Riesgo de XSS
- Clickjacking
- Ataques de contenido mixto

**Recomendación:**

Configurar cabeceras HTTP seguras.

## **7.8 Cookies Inseguras**

**Severidad:** Media

**Descripción:**

Las cookies de sesión no cuentan con los atributos HttpOnly y SameSite.

**Impacto:**

- Robo de sesión
- CSRF

**Recomendación:**

Configurar cookies con atributos de seguridad.

## 8. Evidencias

### 8.1 Evidencias de reconocimiento

Se observa la enumeración de rutas accesibles en la aplicación, identificando directorios potencialmente expuestos que sirvieron como punto de partida para el análisis de seguridad además de un reconociendo de puertos con servicios y versiones.

```
> gobuster dir -w /usr/share/SecLists/Discovery/Web-Content/DirBuster-2007_directory-list-2.3-small.txt -u http://localhost:8080/modules/ -x php,txt,html -t 20 | grep -v "200"
```

Gobuster v3.6  
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url: http://localhost:8080/modules/  
[+] Method: GET  
[+] Threads: 20  
[+] Negative Status codes: 404  
[+] User Agent: gobuster/3.6  
[+] Extensions: php,txt,html  
[+] Timeout: 10s

Starting gobuster in directory enumeration mode

/.html (Status: 403) [Size: 276]  
/admin.php (Status: 302) [Size: 8115] [--> ../login.php]  
/edit (Status: 301) [Size: 320] [--> http://localhost:8080/modules/edit/]  
/html (Status: 403) [Size: 276]  
Progress: 350668 / 350672 (100.00%)

Finished

```
> gobuster dir -w /usr/share/SecLists/Discovery/Web-Content/DirBuster-2007_directory-list-2.3-small.txt -u http://localhost:8080/user/ -x php,txt,html -t 20 | grep -v "200"
```

Gobuster v3.6  
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url: http://localhost:8080/user/  
[+] Method: GET  
[+] Threads: 20  
[+] Negative Status codes: 404  
[+] User Agent: gobuster/3.6  
[+] Extensions: php,txt,html  
[+] Timeout: 10s

Starting gobuster in directory enumeration mode

/.html (Status: 403) [Size: 276]  
/home.php (Status: 302) [Size: 11372] [--> ../login.php]  
/profile.php (Status: 302) [Size: 6137] [--> ../login.php]  
/contacto.php (Status: 302) [Size: 3202] [--> ../login.php]  
/html (Status: 403) [Size: 276]  
Progress: 268005 / 350668 (74.38%)

```
> wfuzz --hc 404 -c -t 20 -w /usr/share/SecLists/Discovery/Web-Content/DirBuster-2007_directory-list-2.3-small.txt -u http://localhost:8080/FUZZ
```

/usr/lib/python3/dist-packages/wfuzz/\_init\_.py:34: UserWarning:Pycurl is not compiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documenta  
tion for more information.

\* Wfuzz 3.1.0 - The Web Fuzzer \*

Target: http://localhost:8080/FUZZ  
Total requests: 87667

ID	Response	Lines	Word	Chars	Payload
000000001	302	0 L	0 W	0 Ch	"# directory-list-2.3-small.txt"
000000003	302	0 L	0 W	0 Ch	"# Copyright 2007 James Fisher"
000000011	302	0 L	0 W	0 Ch	"# Priority-ordered case-sensitive list, where entries were found"
000000066	302	0 L	0 W	0 Ch	"# Attribution:Share Alike 3.0 License. To view a copy of this"
000000014	302	0 L	0 W	0 Ch	"http://localhost:8080/"
000000004	302	0 L	0 W	0 Ch	"#"
000000022	302	0 L	0 W	0 Ch	"#"
000000125	301	9 L	28 W	312 Ch	"user"
000000549	301	9 L	28 W	311 Ch	"css"
000001268	301	9 L	28 W	315 Ch	"modules"
000002254	301	9 L	28 W	315 Ch	"modules"
000003193	301	9 L	28 W	319 Ch	"controlador"
000004508	302	0 L	0 W	0 Ch	"http://localhost:8080/"
000000954	301	9 L	28 W	310 Ch	"js"
000000007	302	0 L	0 W	0 Ch	"# license, visit http://creativecommons.org/licenses/by-sa/3.0/"
000000039	301	9 L	28 W	311 Ch	"log"
000000012	302	0 L	0 W	0 Ch	"# on at least 3 different hosts"
000000005	302	0 L	0 W	0 Ch	"# This work is licensed under the Creative Commons"
000000000	302	0 L	0 W	0 Ch	"# or send a letter to Creative Commons, 171 Second Street,"
000000009	302	0 L	0 W	0 Ch	"# Suite 300, San Francisco, California, 94105, USA."
000000018	302	0 L	0 W	0 Ch	"#"
000000013	302	0 L	0 W	0 Ch	"#"

Total time: 0  
Processed Requests: 87667  
Filtered Requests: 87645  
Requests/sec.: 0

```
> nmap -p 8080 --open -sS -sVC -Pn 127.0.0.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2026-01-10 20:24 -05
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00020s latency).

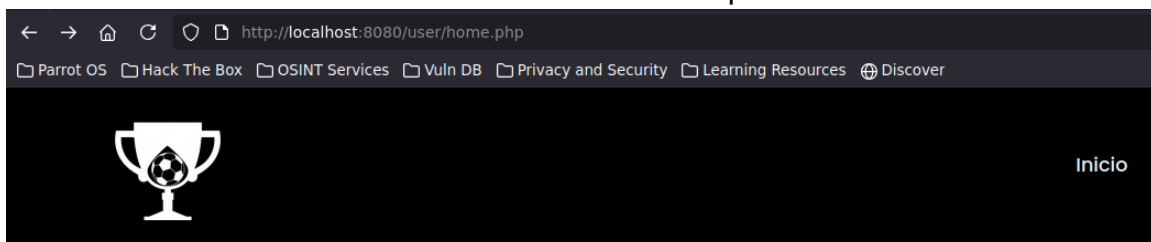
PORT      STATE SERVICE VERSION
8080/tcp  open  http    Apache httpd 2.4.65 ((Debian))
|_ http-server-header: Apache/2.4.65 (Debian)
|_ http-cookie-flags:
|   /:
|       PHPSESSID:
|       httponly flag not set
|_ http-open-proxy: Proxy might be redirecting requests
|_ http-title: Golden Player
|_ Requested resource was login.php

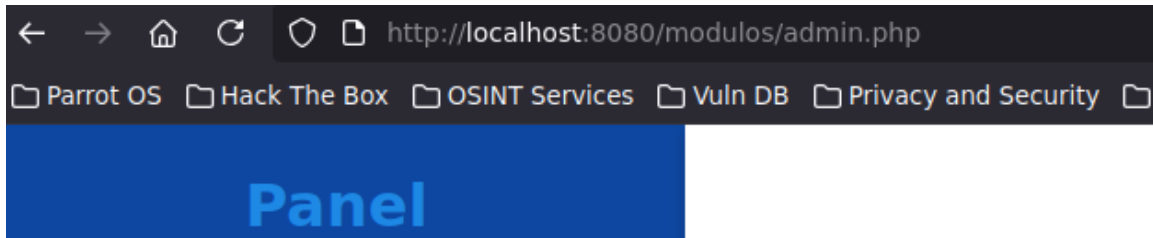
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.92 seconds
```

```
/home/felipe/Golden-player > on P main > nmap -p- --open -sS -Pn 127.0.0.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2026-01-13 09:19 -05
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000030s latency).
Not shown: 65495 closed tcp ports (reset)
PORT      STATE SERVICE
1795/tcp  open  dpi-proxy
2364/tcp  open  oi-2000
3717/tcp  open  wv-csp-udp-cir
5783/tcp  open  3par-mgmt-ssl
8080/tcp  open  http-proxy
8081/tcp  open  blackice-icecap
8682/tcp  open  unknown
9701/tcp  open  unknown
10290/tcp open  unknown
14881/tcp open  unknown
16394/tcp open  unknown
19137/tcp open  unknown
19986/tcp open  unknown
21082/tcp open  unknown
21964/tcp open  unknown
22654/tcp open  unknown
26579/tcp open  unknown
26681/tcp open  unknown
27249/tcp open  unknown
29121/tcp open  unknown
30250/tcp open  unknown
31059/tcp open  unknown
31264/tcp open  unknown
31710/tcp open  unknown
32771/tcp open  sometimes-rpc5
```

## 8.2 Evidencias de Control de acceso roto

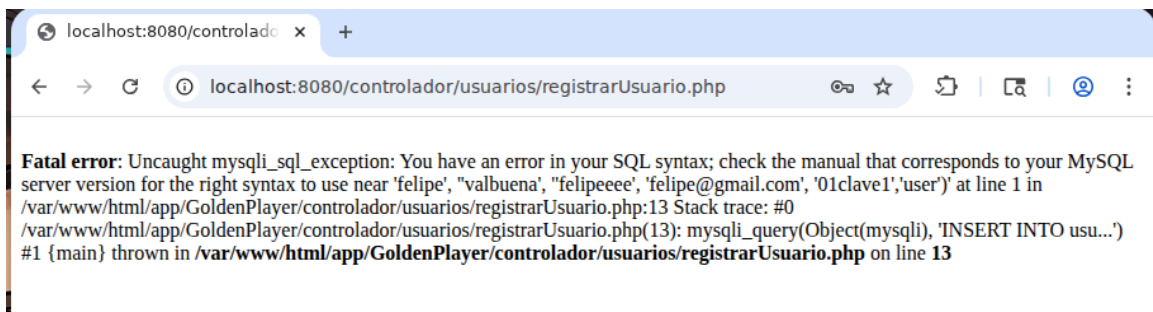
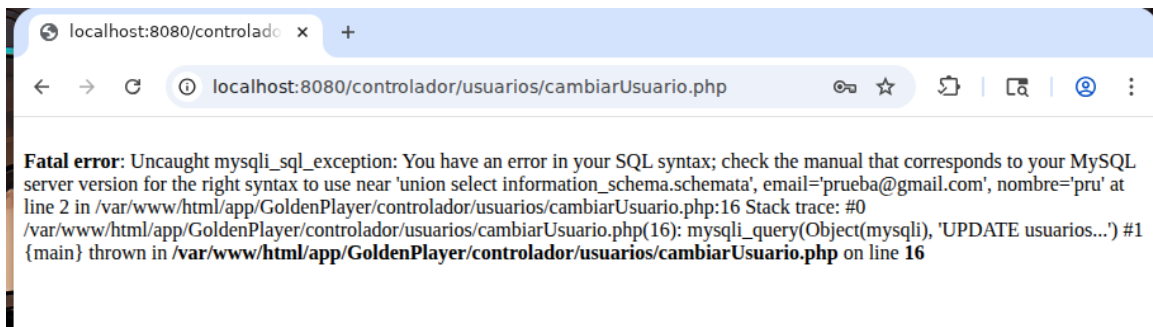
La imagen evidencia que un usuario con rol estándar accede directamente a funcionalidades de administrador mediante la manipulación de la URL.





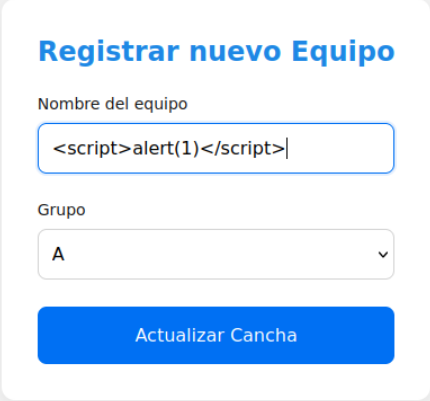
### 8.3 Evidencias de Inyeccion SQL

Se muestra el uso de payloads de inyección SQL que alteran el comportamiento esperado de la consulta, confirmando la falta de validación en el backend.



## 8.4 Evidencias de XSS almacenado en gestión de campeonatos

Se evidencia la posibilidad de almacenar código JavaScript en el campo de nombre, el cual se ejecuta posteriormente al ser visualizado por un usuario con privilegios elevados.



**Registrar nuevo Equipo**

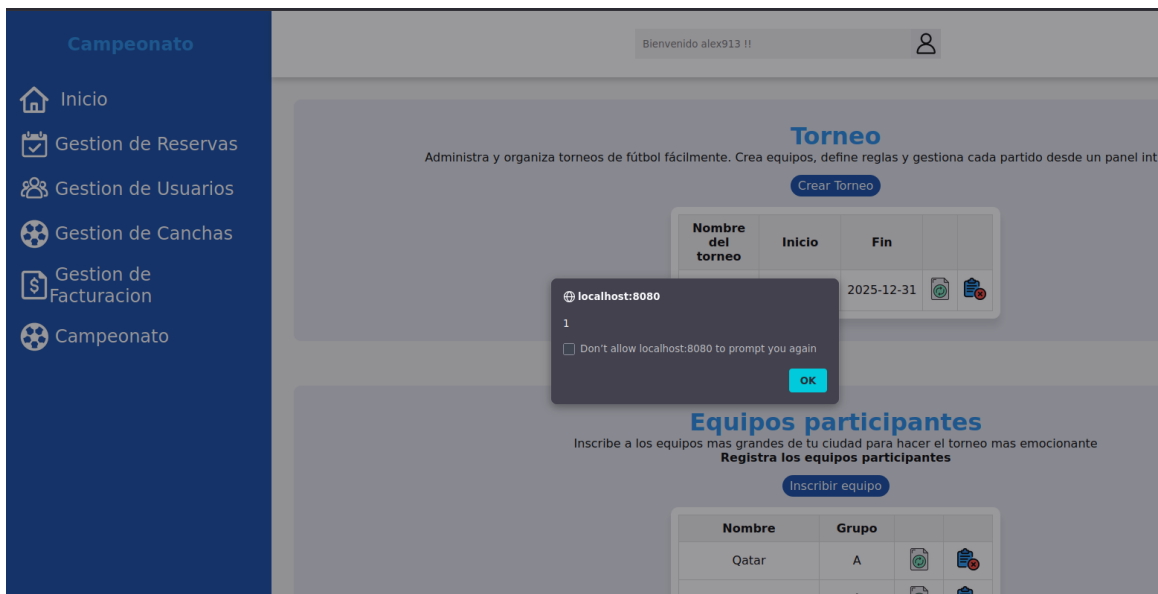
Nombre del equipo

`<script>alert(1)</script>`

Grupo

A

Actualizar Cancha



Campeonato

- Inicio
- Gestion de Reservas
- Gestion de Usuarios
- Gestion de Canchas
- Gestion de Facturacion
- Campeonato

Bienvenido alex913 !!

### Torneo

Administra y organiza torneos de fútbol fácilmente. Crea equipos, define reglas y gestiona cada partido desde un panel int

Crear Torneo

Nombre del torneo	Inicio	Fin
		2025-12-31

localhost:8080

1

☐ Don't allow localhost:8080 to prompt you again

OK

### Equipos participantes

Inscribe a los equipos mas grandes de tu ciudad para hacer el torneo mas emocionante

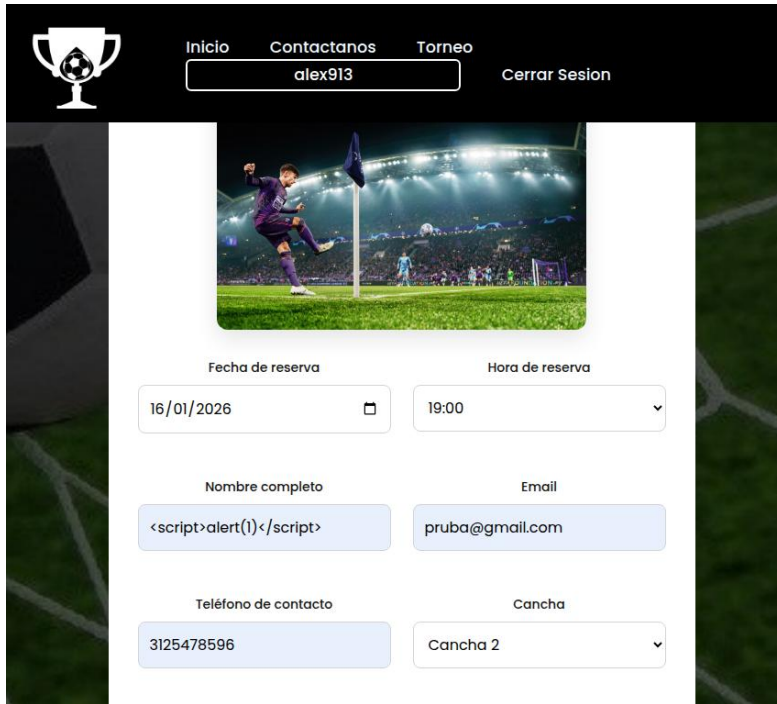
Registra los equipos participantes

Inscribir equipo

Nombre	Grupo
Qatar	A
	A

## 8.5 Evidencias Cross-Site Scripting (XSS) almacenado en múltiples funcionalidades

Se evidencia la posibilidad de almacenar código JavaScript en el campo de nombre, el cual se ejecuta posteriormente al ser visualizado por un usuario con privilegios elevados.



The screenshot shows a web application interface with a dark header. The header contains a trophy icon, navigation links for 'Inicio', 'Contactanos', and 'Torneo', a user profile dropdown showing 'alex913', and a 'Cerrar Sesion' button. The main content area features a soccer field background image. Below the image is a registration form with the following fields:

- Fecha de reserva:** 16/01/2026
- Hora de reserva:** 19:00
- Nombre completo:** <script>alert(1)</script>
- Email:** pruba@gmail.com
- Teléfono de contacto:** 3125478596
- Cancha:** Cancha 2

## 8.6 Evidencias Ausencia de Tokens Anti-CSRF

Se observa que las peticiones sensibles no incluyen tokens de protección contra CSRF, permitiendo la ejecución de acciones sin validación de origen.

```
<div class="form-box login">
  <h2>Iniciar sesion</h2>
  <form method="POST" action="">
    <div class="input-box">
      <span class="icon"><ion-icon name="mail-outline"></ion-icon></span>
      <input type="email" name="email" required>
      <label>Correo Electronico</label>
    </div>
    <div class="input-box">
      <span class="icon"><ion-icon name="lock-closed-outline"></ion-icon></span>
      <input type="password" name="password" required>
      <label>Contraseña</label>
    </div>
    <input type="submit" name="btningresar" class="btn" value="Login">
    <div class="login-register">
      <p><a href="#">No tienes una cuenta?</a> <a href="crearCuenta.php" class="register-link"> Crear cuenta</a></p>
    </div>
  </form>
</div>
```

Spider(Araña) AJAX Spider Escaneo Activo +

**Ausencia de Tokens Anti-CSRF**  
URL: http://localhost:8080  
Riesgo: Medium  
Confianza: Low  
Parámetro:  
Ataque:  
Evidencia: <form method="POST" action="">  
CWE ID: 352  
WASC ID: 9  
Origen: Pasivo (10202 - Ausencia de Tokens Anti-CSRF)  
Vector de Entrada:



```


## Crear nueva cuenta


<form method="POST" action="controlador/usuarios/regarstrarUsuario.php">


  <div class="input-box">
    <input type="text" name="nombre" required>
    <label>Nombre</label>
  </div>
  <div class="input-box">
    <input type="text" name="apellido" required>
    <label>Apellido</label>
  </div>
  <div class="input-box">
    <input type="text" name="username" required>
    <label>Nombre de usuario</label>
  </div>
  <div class="input-box">
    <input type="email" name="email" required>
    <label>Correo Electronico</label>
  </div>
  <div class="input-box">
    <input type="password" name="password_hash" required>
    <label>Contraseña</label>
  </div>


```

Spider(Araña) AJAX Spider Escaneo Activo +

**Ausencia de Ttokens Anti-CSRF**

URL: http://localhost:8080/crearCuenta.php

Riesgo:  Medium

Confianza: Low

Parámetro:

Ataque:

Evidencia: <form method="POST" action="controlador/usuarios/regarstrarUsuario.php">

CWE ID: 352

WASC ID: 9

Origen: Pasivo (10202 - Ausencia de Ttokens Anti-CSRF)

Vector de Entrada:

```


## Iniciar sesion



<form method="POST" action="">
  <div class="input-box">
    <span class="icon"><ion-icon name="mail-outline"></ion-icon></span>
    <input type="email" name="email" required>
    <label>Correo Electronico</label>
  </div>
  <div class="input-box">
    <span class="icon"><ion-icon name="lock-closed-outline"></ion-icon></span>
    <input type="password" name="password" required>
    <label>Contraseña</label>
  </div>
  <input type="submit" name="btningresar" class="btn" value="Login">
  <div class="login-register">
    <p>¿No tienes una cuenta?<a href="crearCuenta.php" class="register-link"> Crear cuenta</a></p>
  </div>
</form>
</div>


```

Spider(Araña) AJAX Spider Escaneo Activo +

**Ausencia de Ttokens Anti-CSRF**

URL: http://localhost:8080/login.php

Riesgo:  Medium

Confianza: Low

Parámetro:

Ataque:

Evidencia: <form method="POST" action="">

CWE ID: 352

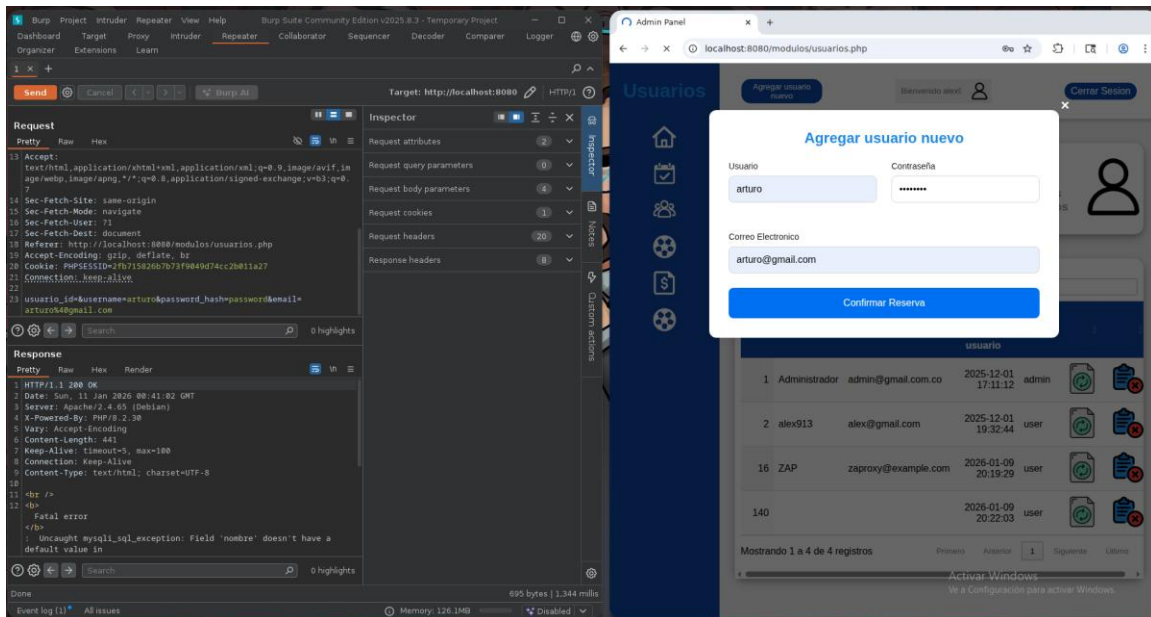
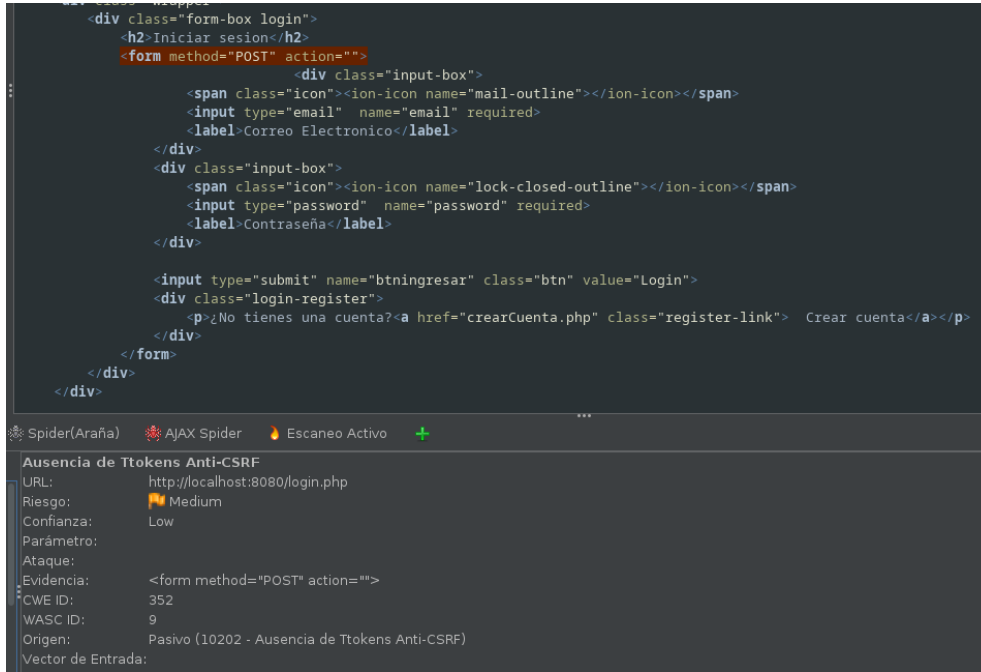
WASC ID: 9

Origen: Pasivo (10202 - Ausencia de Ttokens Anti-CSRF)

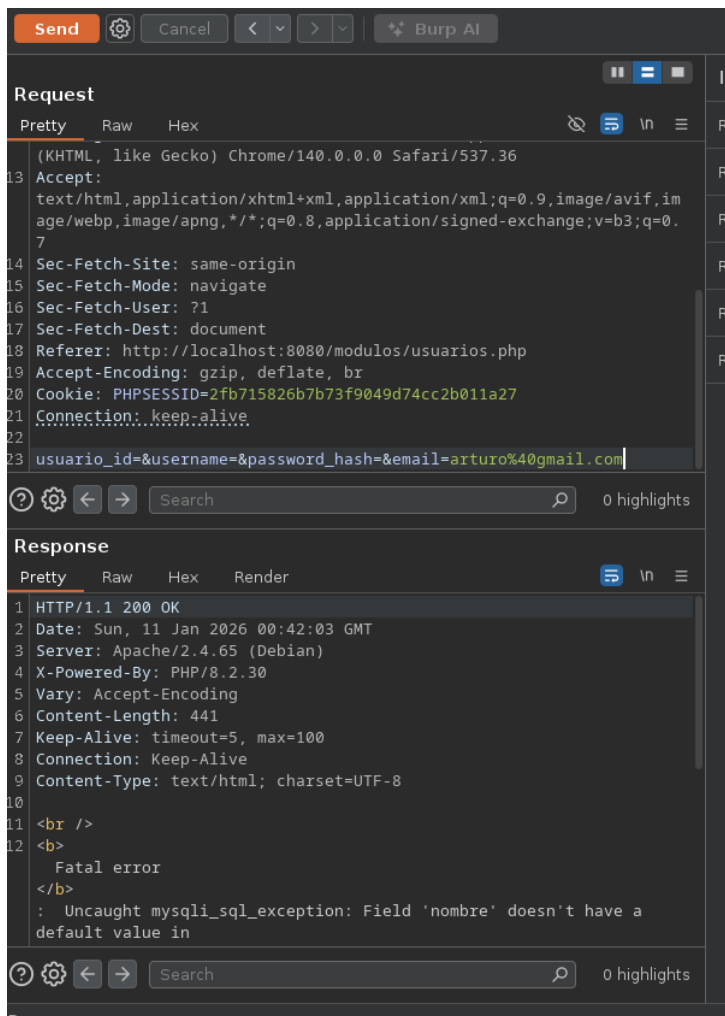
Vector de Entrada:

## 8.7 Evidencias Manipulación de Parámetros (Parameter Tampering)

La imagen muestra la alteración manual de parámetros enviados desde el cliente, lo que permite modificar el comportamiento de la aplicación sin validación del lado servidor.







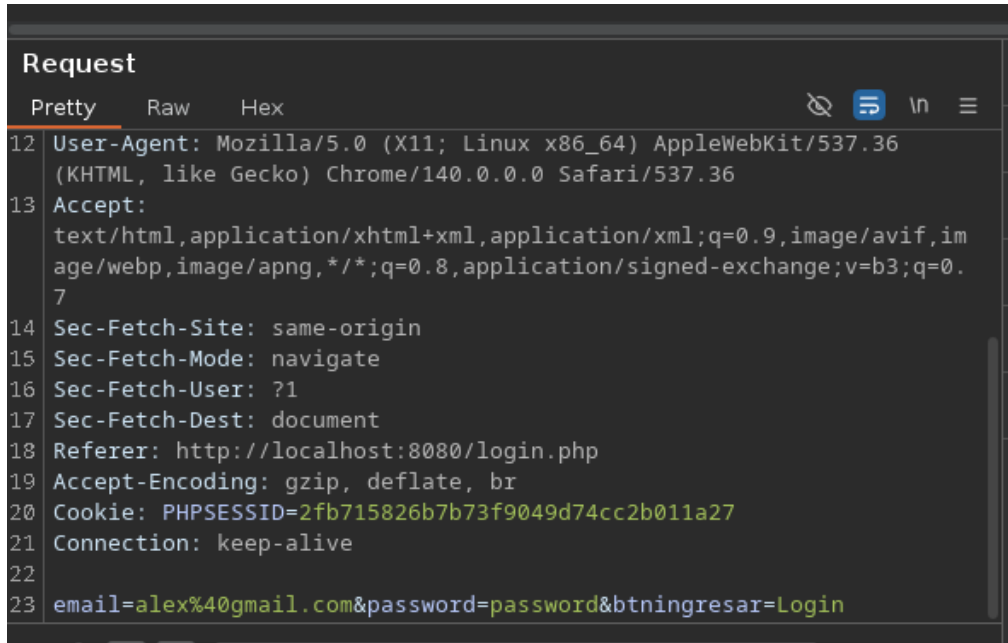
## 8.8 Evidencias Cabeceras de Seguridad Inseguras

Se evidencia la ausencia de cabeceras como CSP, X-Frame-Options y X-Content-Type-Options en la respuesta del servidor.

Cabecera Content Security Policy (CSP) no configurada	
URL:	http://localhost:8080
Riesgo:	🟡 Medium
Confianza:	High
Parámetro:	
Ataque:	
Evidencia:	
CWE ID:	693
WASC ID:	15
Origen:	Pasivo (10038 - Cabecera Content Security Policy (CSP) no configurada)
Referencia de Alerta:	10038-1
Vector de Entrada:	

## 8.9 Evidencias Cookies Inseguras

Se observa que la cookie de sesión carece de los atributos HttpOnly y SameSite, aumentando el riesgo de robo de sesión y ataques CSRF.



```
Request
Pretty Raw Hex
12 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/140.0.0.0 Safari/537.36
13 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
14 Sec-Fetch-Site: same-origin
15 Sec-Fetch-Mode: navigate
16 Sec-Fetch-User: ?1
17 Sec-Fetch-Dest: document
18 Referer: http://localhost:8080/login.php
19 Accept-Encoding: gzip, deflate, br
20 Cookie: PHPSESSID=2fb715826b7b73f9049d74cc2b011a27
21 Connection: keep-alive
22
23 email=alex%40gmail.com&password=password&btningresar>Login
```

## 9. Recomendaciones generales

A partir de los hallazgos identificados durante la auditoría de seguridad, se recomienda implementar las siguientes medidas generales con el fin de fortalecer la seguridad de la aplicación:

- Implementar validaciones estrictas en el backend para todos los datos enviados por el cliente, evitando confiar en controles del frontend.
- Aplicar sanitización y escape de salida (output encoding) en todos los campos que se renderizan en vistas HTML para prevenir ataques XSS.
- Corregir los controles de acceso y validar los roles de usuario en cada endpoint, asegurando que solo usuarios autorizados accedan a funcionalidades administrativas.
- Configurar correctamente las cabeceras de seguridad HTTP, incluyendo Content-Security-Policy (CSP), X-Frame-Options y X-Content-Type-Options.
- Establecer cookies de sesión con los atributos de seguridad HttpOnly, SameSite y Secure.
- Restringir la exposición de servicios innecesarios, manteniendo bases de datos y servicios internos aislados de accesos externos.
- Realizar pruebas de seguridad periódicas y revisiones de código para detectar vulnerabilidades antes de pasar a producción.

## 10. Conclusiones

La auditoría de seguridad realizada permitió identificar diversas vulnerabilidades relacionadas principalmente con controles de acceso, validación de entradas y configuraciones de seguridad a nivel de servidor. Entre los hallazgos más relevantes se destacan fallos de control de acceso que permiten la escalada de privilegios, así como vulnerabilidades de Cross-Site Scripting (XSS) almacenado que podrían comprometer cuentas administrativas.

Si bien la aplicación cuenta con mecanismos básicos de autenticación y control de sesiones, se evidencian debilidades en la implementación de buenas prácticas de seguridad recomendadas por OWASP. La ausencia de cabeceras de seguridad y el uso de cookies sin atributos de protección incrementan el riesgo ante ataques comunes en aplicaciones web.

En conclusión, el sistema presenta un nivel de seguridad mejorable. La implementación de las recomendaciones propuestas permitirá reducir significativamente la superficie de ataque y fortalecer la protección de la información y de los usuarios finales.

## 11. Anexos

- OWASP Top 10 2021  
<https://owasp.org/www-project-top-ten/>
- Documentación oficial de OWASP ZAP  
<https://www.zaproxy.org/documentation/>
- Guía de cabeceras de seguridad HTTP  
<https://owasp.org/www-project-secure-headers/>