Web Applica on Vulnerability Assessment Report

Future Interns – Cyber Security Internship

Name of Candidate: Aditya Gunjal
Internship Provider: Future Interns
Domain: Cyber Security
Date of Execution: 27 December 2025
Target Applica on: OWASP Juice Shop (Localhost)
Target URL: http://localhost:3000
Primary Tool Used: OWASP ZAP

---

1. Introduction

Web applications are frequent targets of cyberattacks due to improper input validation, weak authentication mechanisms, and insecure configurations. This project focuses on identifying real-world web application vulnerabilities using industry-standard security tools and OWASP guidelines.

The objective of this assessment was to analyse a deliberately vulnerable web application, identify common security flaws, and recommend appropriate remediation strategies. The findings of this assessment simulate real-world penetration testing tasks performed by cybersecurity professionals.

---

2. Scope of Testing

The security assessment was conducted only on a locally hosted instance of the OWASP Juice Shop application running on http://localhost:3000.

- The testing was performed strictly for educational and internship purposes
- No public, production, or third-party systems were tested
- Both automated and manual testing techniques were applied

This ensured that the assessment remained legal, ethical, and controlled.

---

3. Tools & Methodology

Tools Used

- OWASP ZAP – Automated vulnerability scanning and traffic interception
- Web Browser (Proxy configured with ZAP)
- Local OWASP Juice Shop environment

Methodology

1. Ini al reconnaissance and application exploration
2. Automated vulnerability scanning using OWASP ZAP
3. Manual validation of critical vulnerabilities
4. Mapping findings to OWASP Top 10
5. Documenting impact and remediation steps

---

4. Overview of OWASP

What is OWASP?

OWASP (Open Web Applica on Security Project) is a non-profit organization dedicated to improving the security of software applications. It provides free tools, documentation, and frameworks that help organizations identify and mi gate application security risks.

Purpose of OWASP

- To educate developers and security professionals about web security risks
- To provide open-source tools such as OWASP ZAP
- To publish security standards like the OWASP Top 10
- To promote secure coding and defensive development practices

---

Importance of OWASP in Cybersecurity

- Standardized Framework: OWASP Top 10 is globally accepted for web security assessment
- Risk Reduction: Helps prevent vulnerabilities like SQL Injection on and XSS
- Compliance Support: Assists organizations in meting security standards (PCI-DSS, GDPR)
- Career Relevance: OWASP knowledge is essential for cybersecurity roles

---

5. Automated Vulnerability Scan Results

An automated vulnerability scan was performed using OWASP ZAP on the target application.

ZAP Scan Report (HTML format).html

Scan Summary:

- Total Alerts Identified: 22
- High Risk: 1
- Medium Risk: 5
- Low Risk: 9
- Informational: 7

Key Vulnerabilities Identified

Sr. No	Vulnerability	Risk Level	OWASP Category	Fix Summary
1	SQL Injection	High	A03:2021 – Injection	Use parameterized queries
2	Content Security Policy Not Set	Medium	A05 – Security Misconfiguration	Implement CSP headers
3	Cross-Domain Script Inclusion	Medium	A05 – Security Misconfiguration	Use trusted CDNs & SRI
4	Missing An -Clickjacking Header	Medium	A05 – Security Misconfiguration	Add X-Frame-Options
5	Session ID in URL	Medium	A07 – Auth Failures	Use secure cookies
6	Vulnerable JavaScript Library	Medium	A06 – Vulnerable Components	Update libraries
7	HSTS Header Missing	Low	A05 – Security Misconfiguration	Enable HSTS
8	X-Content-Type-Options Missing	Low	A05 – Security Misconfiguration	Add no-sniff header
9	Private IP Disclosure	Low	A01 – Broken Access Control	Remove internal details
10	Informa on Disclosure via Comments	Informational	A01 – Broken Access Control	Remove debug comments

6. Manual Vulnerability Findings

In addition to automated scanning, manual testing was performed to validate high-impact vulnerabilities.

6.1 SQL Injection (Authentication Bypass)

OWASP Category: A03 – Injection
Risk Level: High

The login functionality was vulnerable to SQL Injection, allowing authentication bypass without valid credentials.

Impact:
An attacker can gain unauthorized access to user accounts, including privileged users.

Remediation:

- Use parameterized queries
- Validate and sanitize user input
- Avoid dynamic SQL construction

6.2 Broken Authentication (Admin Login)

OWASP Category: A07 – Identification & Authentication Failures
Risk Level: High

Using crafted input, the attacker was able to log in as the admin user, gaining administrative privileges.

Impact:
Complete compromise of the application, including sensitive data access and administrative control.

Remediation:

- Secure authentication logic
- Enforce strict credential validation
- Implement account lockout mechanisms

6.3 Reflected Cross-Site Scrip ng (XSS)

OWASP Category: A03 – Injection
Risk Level: Medium

A reflected XSS vulnerability was identified in the search functionality using the payload:

<img src=x onerror=alert('XSS')>
Impact:
Allows execution of malicious JavaScript in the victim’s browser, leading to session hijacking or phishing attacks.

Remediation:

- Output encoding

- Input sanitization
- Implement Content Security Policy (CSP)

6.4 Broken Access Control (Confidential Resource Access)

OWASP Category: A01 – Broken Access Control
Risk Level: Medium

Unauthorized access to restricted resources was possible without proper access validation.

Impact:
Sensitive data exposure and privilege escalation.

Remediation:

- Enforce role-based access control
- Validate authorization on server side

---

7. OWASP Top 10 Mapping Summary

Vulnerability	OWASP Category
SQL Injection	A03
Admin Login Bypass	A07
Reflected XSS	A03
Security Misconfiguration	A05
Broken Access Control	A01

---

8. Learning Outcomes

- Practical understanding of OWASP Top 10 vulnerabilities
- Hands-on experience with OWASP ZAP
- Improved knowledge of secure coding practices
- Understanding of real-world attack vectors and defences

---

9. Challenges Faced

During the assessment, several challenges were encountered, including configuring OWASP ZAP to properly intercept browser traffic and resolving initial environment setup issues with OWASP Juice

Shop. Multiple attempts and troubleshooting were required to ensure successful scanning and accurate vulnerability detection.

10. Conclusion & Recommendations

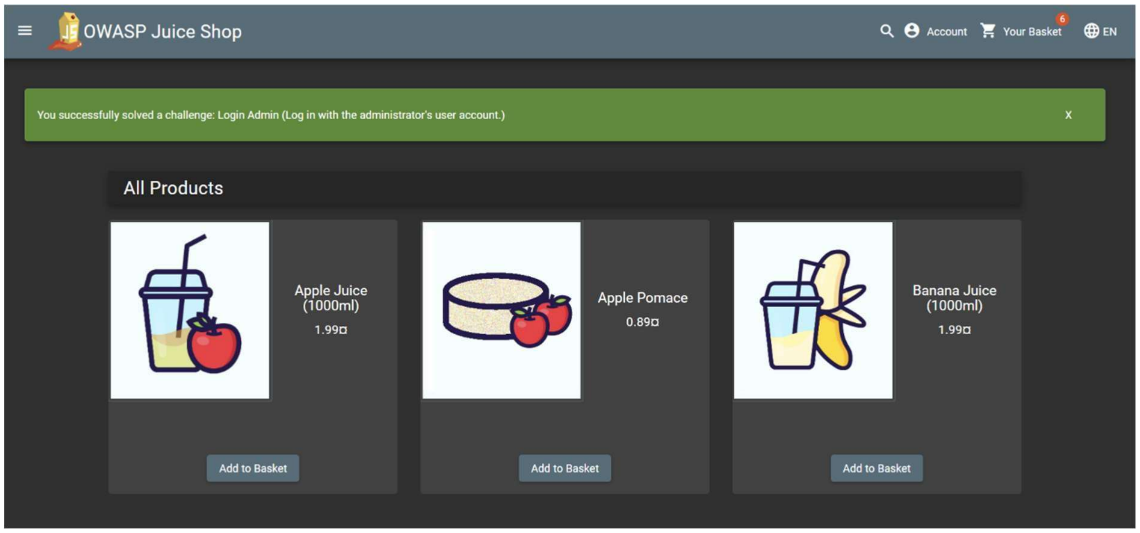
The security assessment revealed multiple critical and medium-risk vulnerabilities that could lead to unauthorized access, data leakage, and client-side attacks. Addressing these vulnerabilities through secure coding practices, proper configuration, and regular security testing will significantly enhance the application's security posture.

It is recommended that organizations conduct periodic vulnerability assessments and follow OWASP guidelines to reduce the risk of web-based attacks.

Screenshots & Evidence

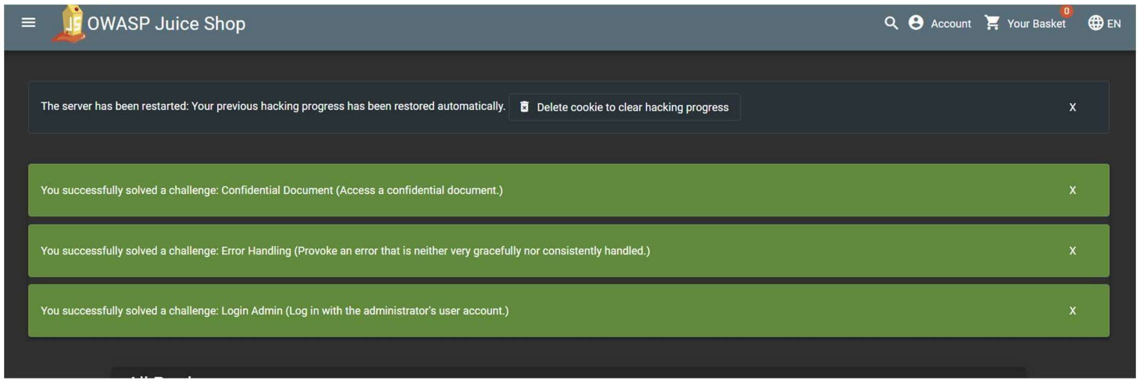
Screenshots of vulnerability exploitation, OWASP ZAP alerts, SQL Injection payloads, XSS execution, and administrative login were captured during testing and are attached as evidence to support the findings in this report.

1.



You successfully solved a challenge: Login Admin (Log in with the administrator's user account.)

2.



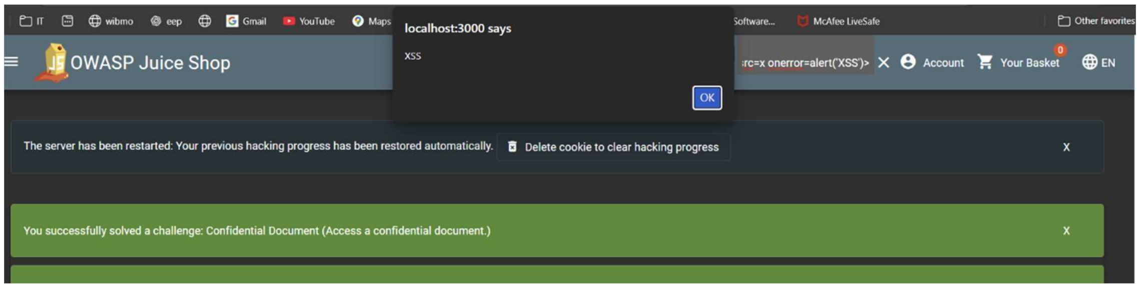
The server has been restarted: Your previous hacking progress has been restored automatically.

You successfully solved a challenge: Confidential Document (Access a confidential document.)

You successfully solved a challenge: Error Handling (Provoke an error that is neither very gracefully nor consistently handled.)

You successfully solved a challenge: Login Admin (Log in with the administrator's user account.)

3.



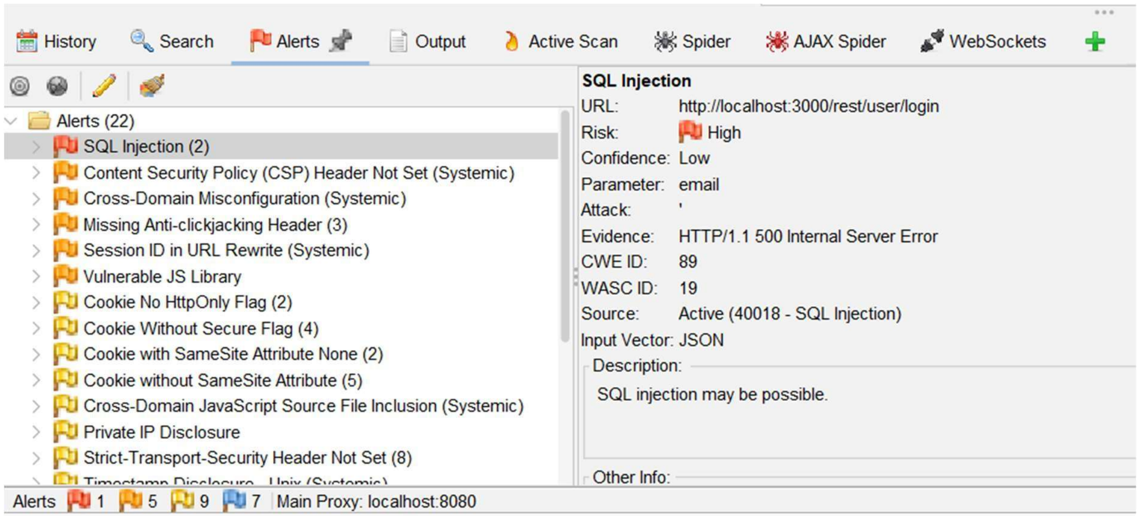
localhost:3000 says

xss

The server has been restarted: Your previous hacking progress has been restored automatically.

You successfully solved a challenge: Confidential Document (Access a confidential document.)

4.



Alerts (22)
- SQL Injection (2)
- Content Security Policy (CSP) Header Not Set (Systemic)
- Cross-Domain Misconfiguration (Systemic)
- Missing Anti-clickjacking Header (3)
- Session ID in URL Rewrite (Systemic)
- Vulnerable JS Library
- Cookie No HttpOnly Flag (2)
- Cookie Without Secure Flag (4)
- Cookie with SameSite Attribute None (2)
- Cookie without SameSite Attribute (5)
- Cross-Domain JavaScript Source File Inclusion (Systemic)
- Private IP Disclosure
- Strict-Transport-Security Header Not Set (8)
- Timestamp Disclosure - Unix (Systemic)

SQL Injection

URL:	http://localhost:3000/rest/user/login
Risk:	High
Confidence:	Low
Parameter:	email
Attack:	'
Evidence:	HTTP/1.1 500 Internal Server Error
CWE ID:	89
WASC ID:	19
Source:	Active (40018 - SQL Injection)
Input Vector:	JSON

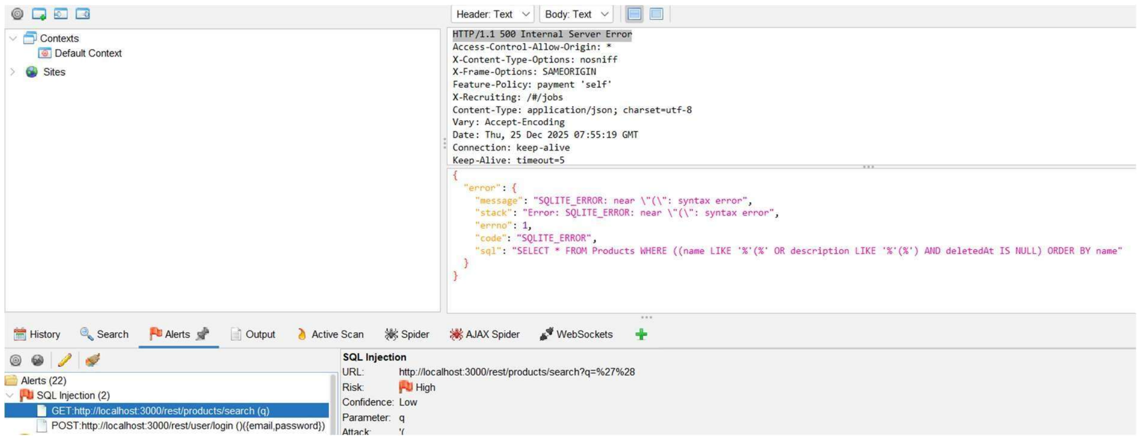
Description:

SQL injection may be possible.

Other Info:

Alerts 1 5 9 7 Main Proxy: localhost:8080

5.



```
HTTP/1.1 500 Internal Server Error
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: application/json; charset=utf-8
Vary: Accept-Encoding
Date: Thu, 25 Dec 2025 07:55:19 GMT
Connection: keep-alive
Keep-Alive: timeout=5
```

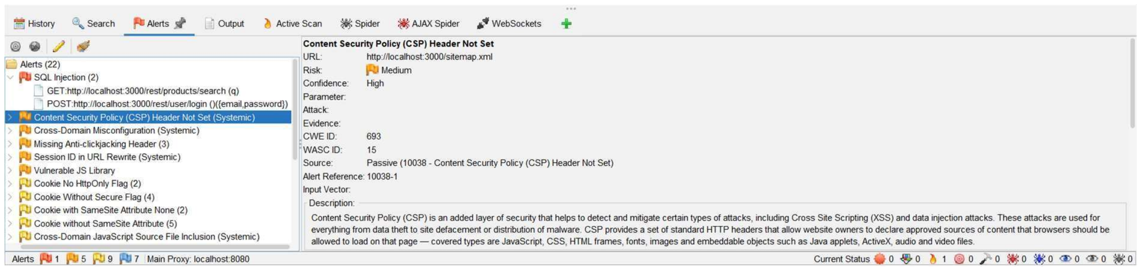
```
{
  "error": {
    "message": "SQLITE_ERROR: near \"(\": syntax error",
    "stack": "Error: SQLITE_ERROR: near \"(\": syntax error",
    "errno": 1,
    "code": "SQLITE_ERROR",
    "sql": "SELECT * FROM Products WHERE ((name LIKE '%'(%' OR description LIKE '%'(%') AND deletedAt IS NULL) ORDER BY name"
  }
}
```

```
Alerts (22)
  SQL Injection (2)
    GET:http://localhost:3000/rest/products/search (q)
    POST:http://localhost:3000/rest/user/login ()({email,password})
```

SQL Injection

URL: http://localhost:3000/rest/products/search?q=%27%28
Risk: High
Confidence: Low
Parameter: q
Attack: '(

6.



```
Alerts (22)
  SQL Injection (2)
    GET:http://localhost:3000/rest/products/search (q)
    POST:http://localhost:3000/rest/user/login ()({email,password})
  Content Security Policy (CSP) Header Not Set (Systemic)
  Cross-Domain Misconfiguration (Systemic)
  Missing Anti-clickjacking Header (3)
  Session ID in URL Rewrite (Systemic)
  Vulnerable JS Library
  Cookie No HttpOnly Flag (2)
  Cookie Without Secure Flag (4)
  Cookie with SameSite Attribute None (2)
  Cookie without SameSite Attribute (5)
  Cross-Domain JavaScript Source File Inclusion (Systemic)
```

Content Security Policy (CSP) Header Not Set

URL: http://localhost:3000/sitemap.xml
Risk: Medium
Confidence: High
Parameter:
Attack:
Evidence:
CWE ID: 693
WASC ID: 15
Source: Passive (10038 - Content Security Policy (CSP) Header Not Set)
Alert Reference: 10038-1
Input Vector:

Description:
Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Alerts 1 5 9 7 Main Proxy: localhost:8080

Current Status 0 0 1 0 0 0 0 0 0 0