

OWASP Juice Shop — Web Application Security Assessment

Prepared by: Sanskar
Project: Future Interns — Web App Vulnerability Assessment (Internship)
Target: OWASP Juice Shop (local) — <http://localhost:3000>
Date: 2025-11-01

Executive Summary

This assessment was performed against a local instance of OWASP Juice Shop as part of the Future Interns internship project. Testing focused on common web application vulnerabilities (XSS, credential handling, access control, information disclosure). Multiple findings were identified and validated with proof-of-concept evidence. The overall risk rating is **High**. Immediate remediation is recommended for high-severity issues (credentials exposure, exposed confidential files, DOM XSS).

Scope & Methodology

In-scope: OWASP Juice Shop running locally at <http://localhost:3000> (training instance).

Out-of-scope: External or third-party services not reachable from the lab.

Methodology: Reconnaissance (dirb), automated scanning (optional), proxy capture (Burp Suite), manual DOM testing, proof-of-concept exploitation, and evidence collection. Findings are mapped to OWASP Top 10 categories.

Environment & Tools

OS: Kali Linux (testing VM).

Juice Shop: Local instance (<http://localhost:3000>).

Tools used: Burp Suite Community, dirb v2.22, browser devtools, report screenshots.

Findings Summary (High-level)

Finding	OWASP Category	Severity	Status
Credentials in Transit (login POST visible)	A02: Cryptographic Failures	High	Open
DOM-based XSS (search/feedback, iframe payloads)	A03: Injection (XSS)	High	Open
Exposed Confidential Files via /ftp (acquisitions.md, incident-support.kdbx)	A01: Broken Access Control	High	Open
Chatbot coupon leakage (bully chatbot)	A01/A05: Business Logic / Misconfiguration	Medium	Open
Missing encoding / photo retrieval (photo wall)	A01: Broken Access Control	Medium	Open
Error handling info leak / verbose responses	A05: Security Misconfiguration	Medium	Open

Finding 1 — Credentials in Transit

Affected URL: POST /rest/user/login (<http://localhost:3000/rest/user/login>)

Severity: High

Summary: Login credentials were observed in a captured POST request body via Burp Suite. The JSON body contains email and password fields, visible in proxy logs (redact before external sharing).

Evidence: Burp HTTP history screenshot showing the POST request (attach screenshot: burp_login_post.png).

Reproduction Steps: Configure browser to use Burp proxy and navigate to Juice Shop. Submit login form with test credentials. Observe the POST request in Burp → Proxy → HTTP history containing {"email": "...", "password": "..."} **Impact:** Network or proxy observers can capture credentials, leading to account takeover or credential reuse attacks.

Remediation: Enforce HTTPS, enable HSTS, ensure cookies use Secure/HttpOnly/SameSite, avoid logging plaintext credentials, rotate any test passwords.

Finding 2 — DOM-based Cross-Site Scripting (DOM XSS)

Affected URL(s): /#/search?q=, feedback form and other client-side inputs

Severity: High

Summary: User-supplied input is reflected into the DOM without proper encoding. Confirmed via iframe/javascript payloads (e.g., `iframe src=javascript:alert('xss')` and remote SoundCloud iframe).

Evidence: Screenshots show the injected iframe and the green challenge banners confirming exploit (attach: dom_xss_payload.png, dom_xss_soundcloud.png).

Reproduction Steps: Navigate to <http://localhost:3000/#/search?q=> Append payload: `<iframe src="javascript:alert('xss')">` and press Enter Observe JavaScript execution or embedded remote content **Impact:** Executes arbitrary JavaScript in victims' browsers — session theft, CSRF, UI spoofing.

Remediation: Sanitize and encode user input before inserting into DOM; use `textContent` instead of `innerHTML`; implement CSP; server-side validation.

Finding 3 — Exposed Confidential Files via /ftp

Affected URL: <http://localhost:3000/ftp> and specific files like [/ftp/acquisitions.md](#), [/ftp/incident-support.kdbx](#)

Severity: High

Summary: Sensitive/confidential documents are publicly accessible via a directory listing. Example file 'acquisitions.md' explicitly states 'This document is confidential'.

Evidence: FTP directory listing screenshot and acquisitions.md content (attach: ftp_listing.png, acquisitions_md.png).

Reproduction Steps: Browse to <http://localhost:3000/ftp> Click acquisitions.md to view its content

Impact: Unauthorized disclosure of corporate plans and potentially sensitive key files (.kdbx) leading to information leakage and compliance issues.

Remediation: Remove sensitive files from web root, restrict access with authentication and authorization, store secrets in secure vaults, and remove backup .bak files from public directories.

Finding 4 — Chatbot Coupon Leakage / Business Logic Abuse

Affected URL: <http://localhost:3000/#/chatbot>

Severity: Medium

Summary: The chatbot can be interacted with to reveal coupon codes via crafted inputs (bully chatbot challenge). This demonstrates business logic weakness allowing unauthorized benefit retrieval.

Evidence: Chat UI screenshot showing coupon requests and challenge banner (attach: chatbot_coupon.png).

Reproduction Steps: Open the chatbot at [#/chatbot](#) Send 'coupon' or crafted messages as performed in test Observe the chatbot responses / coupon code retrieval **Impact:** Unauthorized coupon/code disclosure, monetary loss, or abuse of promotional systems.

Remediation: Add server-side checks to prevent unauthorized coupon issuance, rate-limiting, and validation of user entitlement.

Finding 5 — Missing Encoding / Unauthorized Image Retrieval

Affected URL: Photo wall [#/photo-wall](#) and related image endpoints

Severity: Medium

Summary: Photo retrieval is possible without authorization and encoding protections; challenge 'Missing Encoding' solved by retrieving cat photo. Images may be accessible via predictable paths.

Evidence: Photo wall screenshot showing images (attach: photo_wall.png).

Reproduction Steps: Navigate to [#/photo-wall](#) Access image resources directly or via page

Impact: Insecure direct object references may expose user media; moderate data disclosure risk.

Remediation: Control access to media, use signed URLs for private content, avoid predictable file paths.

Finding 6 — Error Handling & Information Disclosure

Affected URL: Observed error endpoints ([/rest/erro](#)) and verbose responses captured in proxy

Severity: Medium

Summary: Error endpoints return verbose responses or stack traces that may reveal internal implementation details. Such disclosures help attackers craft targeted exploits.

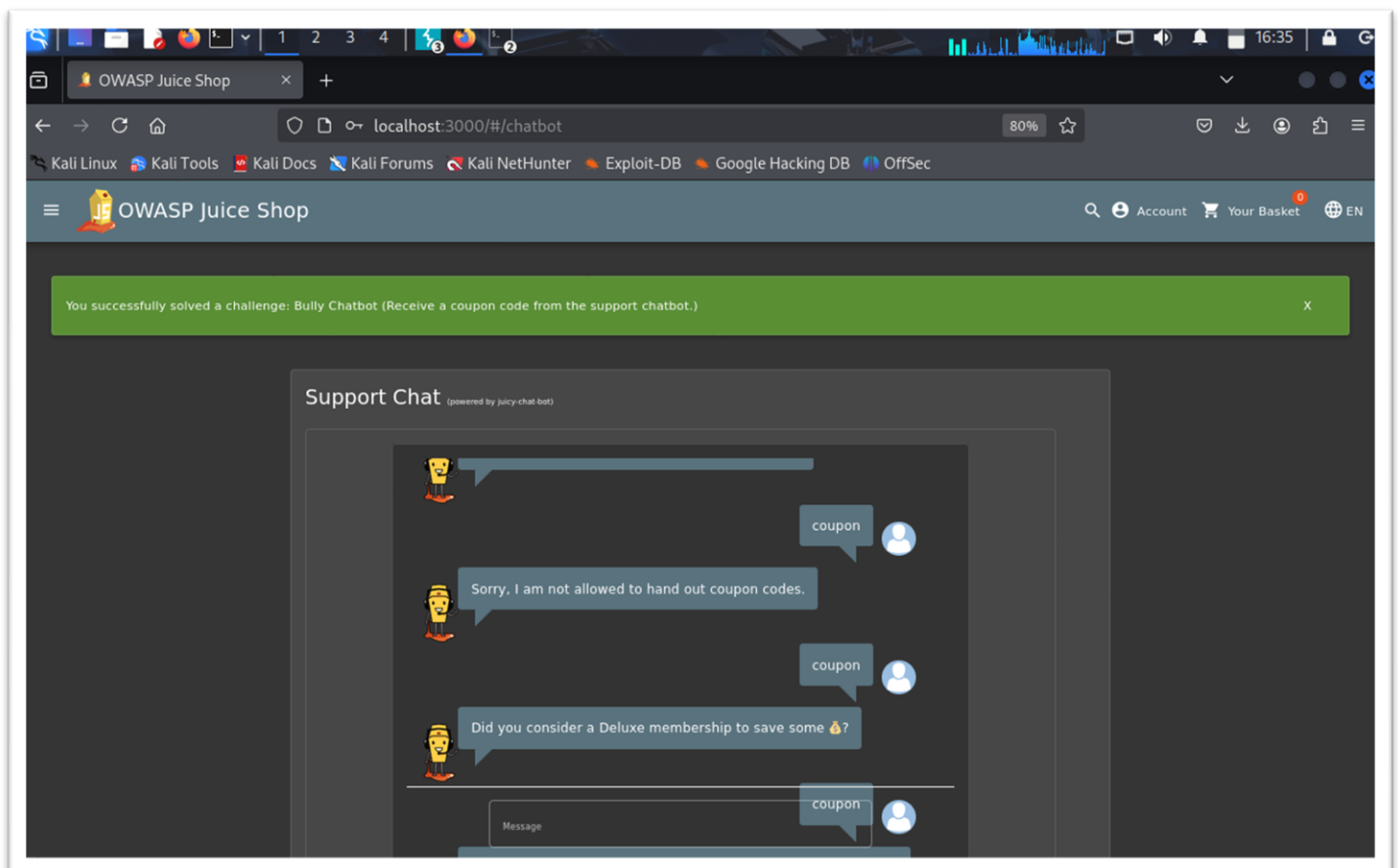
Evidence: Burp screenshot showing error-related requests and responses (attach: burp_error_response.png).

Remediation: Return generic error messages to users; log details securely server-side; remove stack traces from production responses.

Appendix A — Payloads & PoC Summary

PoC Name	Payload / URL	Observed Effect
DOM XSS (alert iframe)	/#/search?q=<iframe src="javascript:alert('xss')">	Executed JS alert / challenge
DOM XSS (SoundCloud iframe)	SoundCloud iframe embed payload	Remote content loaded and
Login POST capture	POST /rest/user/login JSON {email, password}	Credentials visible in Burp
Directory enumeration	dirb http://localhost:3000/	Found /ftp, /profile, /promotion
FTP file access	/ftp/acquisitions.md	Accessed confidential docum

Appendix B — Evidence Attachment Instructions



Final Recommendations & Next Steps

1. Fix high severity issues within 72 hours: credentials exposure, exposed FTP files, DOM XSS entry points.
2. Enforce HTTPS/HSTS and cookie flags.
3. Remove sensitive files from public directories and implement authentication/authorization for file access.
4. Apply input validation and output encoding across client and server.
5. Implement CSP headers and harden error handling.
6. Re-run scans after fixes and provide evidence of remediation.