Security Assessment Report

Target Application	OWASP Juice Shop
Report Author	Pranav Poornachandra S
Date of Assessment	01.11.2025
Date of Report	01.11.2025

1. Executive Summary

This report details the findings of a web application security assessment conducted on the OWASP Juice Shop application as part of the Future Interns Cybersecurity Intern.

The primary objective was to identify common security flaws, specifically focusing on the OWASP Top 10 categories. The assessment identified five (5) critical and high-risk vulnerabilities, which demonstrate significant weaknesses in input validation and access control enforcement.

The most severe findings include SQL Injection and Broken Access Control, both allowing for unauthorized access to sensitive application components and data. Immediate remediation of these issues is strongly recommended to simulate real-world mitigation practices.

Overall Risk Rating: High

2. Assessment Details

2.1 Objectives

The assessment was conducted with the following goals:

- Identify and successfully exploit a minimum of five vulnerabilities.
- Propose practical and specific mitigation strategies suitable for developers.

2.2 Scope and Methodology

Detail	Description
Target Platform	OWASP Juice Shop
Assessment Type	Gray Box
Tools Used	Burp Suite, Manual browser testing
Testing Approach	Manual fuzzing, forced browsing

3. Detailed Findings

The vulnerabilities are prioritized using a standard risk rating system (Critical, High, Medium, Low). The risk is calculated based on Likelihood (How easy is it to exploit?) and Impact (What is the resulting damage?).

3.1 SQL Injection

OWASP Category	Location	Likelihood	Impact
SQL Injection	User Login Functionality	High	Critical

The application constructs database queries using non-sanitized user input from the login form. This vulnerability allowed the injection of malicious SQL commands, resulting in a successful authentication bypass and unauthorized login as the system administrator.

Remediation: Implement Prepared Statements (or parameterized queries) for all database interactions. This fundamentally separates the SQL code from the user-supplied data, ensuring the input is always treated as data, not executable code.

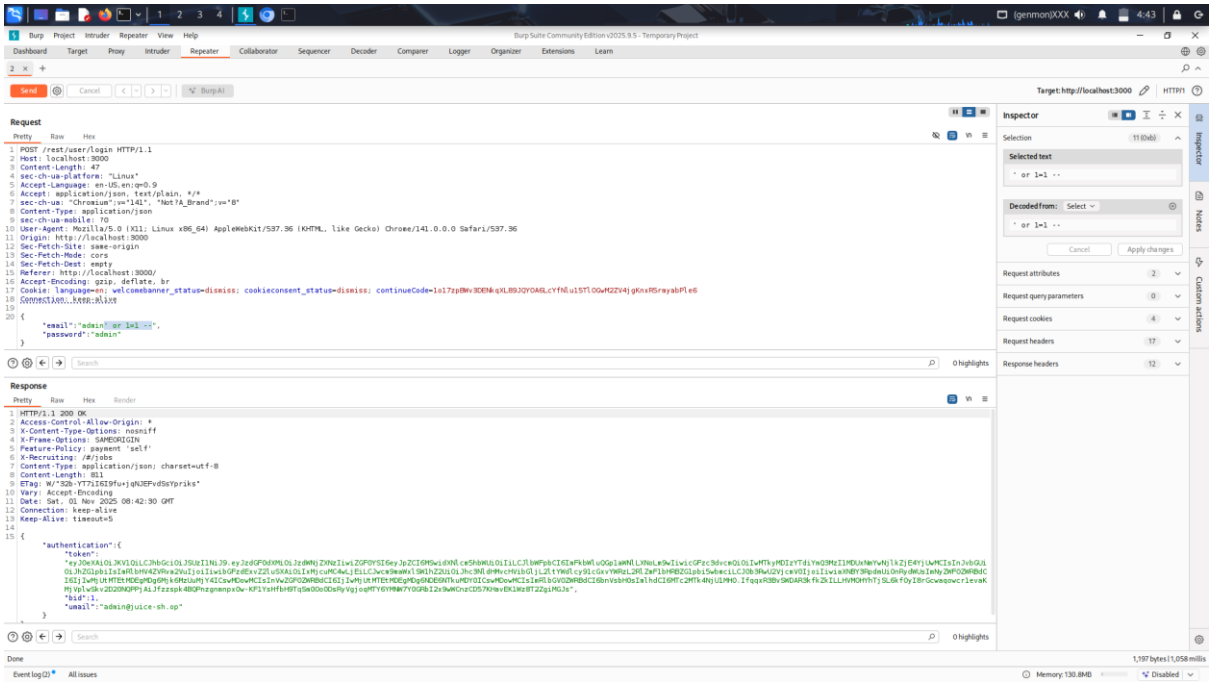


Figure 1 SQL Injection

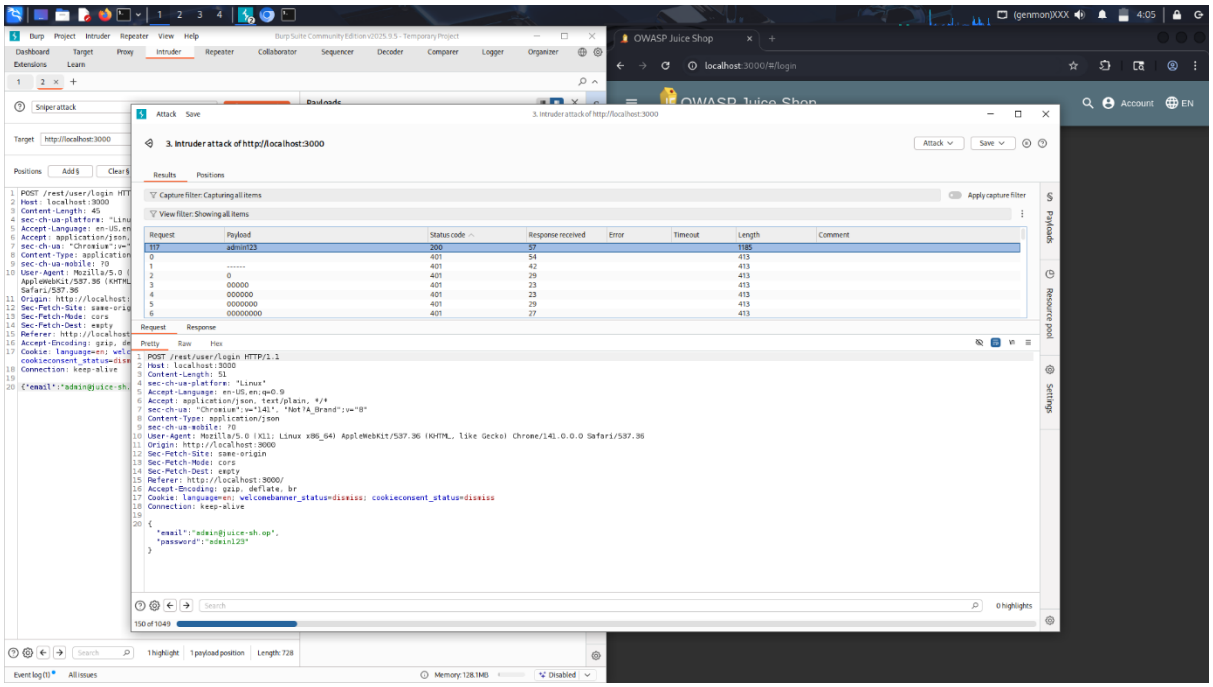
3.2 Broken Access Control

OWASP Category	Location	Likelihood	Impact
Broken Access Control	User Login Functionality	High	Critical

The application fails to enforce granular authorization checks on restricted API endpoints. By simply navigating to or requesting specific administrative URLs, a low-privileged or

unauthenticated user could access and potentially modify resources reserved for the administrator.

Remediation: Adopt a "Deny by Default" authorization policy. On the server side, before processing any request for a restricted resource, explicitly check the user's session token and verify their assigned role or permissions against a predefined access control list (ACL).



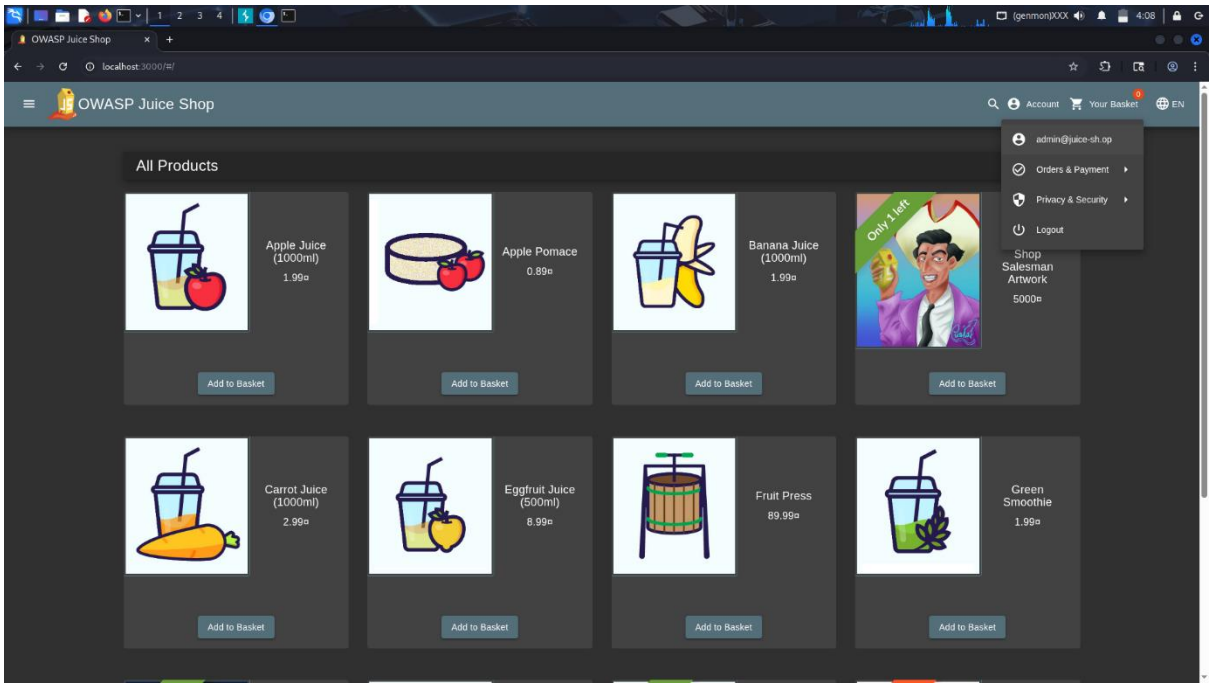


Figure 2 Broken Access Control Provided Admin Privileges

3.3 DOM Cross-Site Scripting (DOM XSS)

OWASP Category	Location	Likelihood	Impact
Injection (XSS)	Search Functionality	Medium	High

The application takes user supplied input (e.g., from the URL hash or search parameter) and directly writes it to the Document Object Model (DOM) without proper validation or encoding. This allows an attacker to execute arbitrary client-side JavaScript in the context of the victim's browser session.

Remediation: Ensure that any user-controlled input used to modify the DOM is properly encoded before rendering. Prefer safe JavaScript APIs (like textContent) over unsafe APIs (like innerHTML).

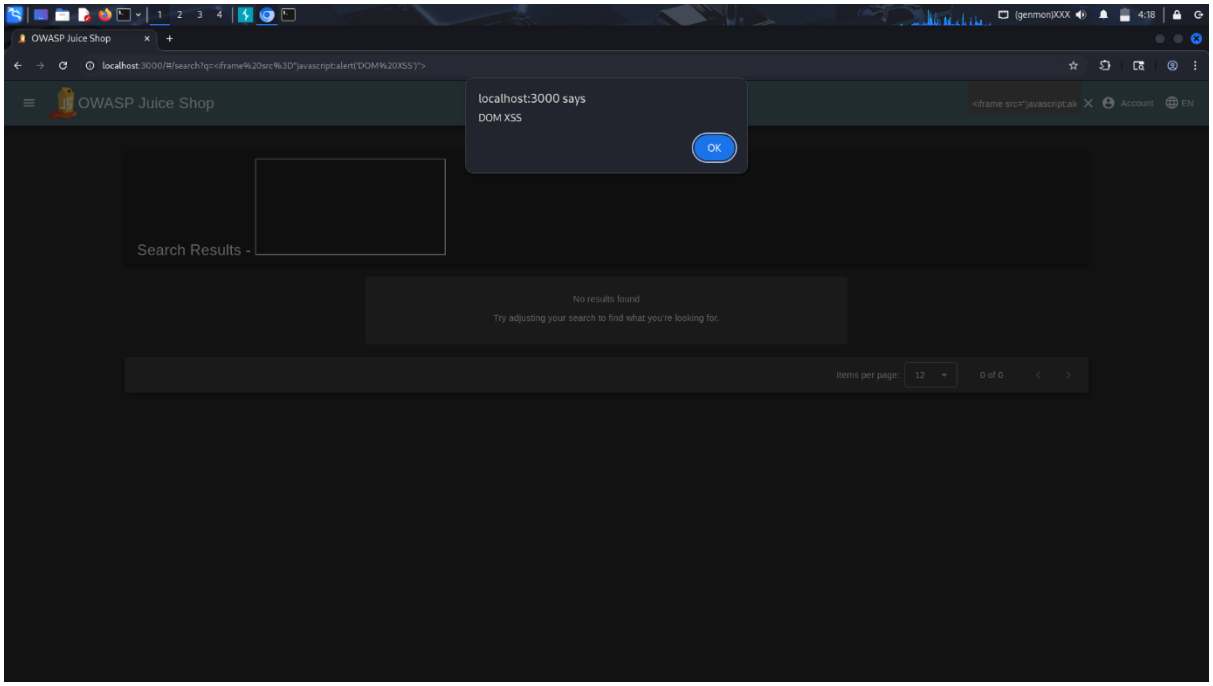


Figure 3 DOM XSS

3.4 Sensitive Data Exposure

OWASP Category	Location	Likelihood	Impact
Sensitive Data Exposure	Redirect from the about section	High	High

A sensitive internal file (legal.md) containing confidential corporate information was publicly accessible via a direct file path. This flaw in file system configuration allows unauthorized access to critical internal documents without any authentication or authorization.

Remediation: Store sensitive internal files outside of the web root directory (where public files are stored). Implement a controlled service endpoint that requires authentication and authorization checks before retrieving and serving the file's content.

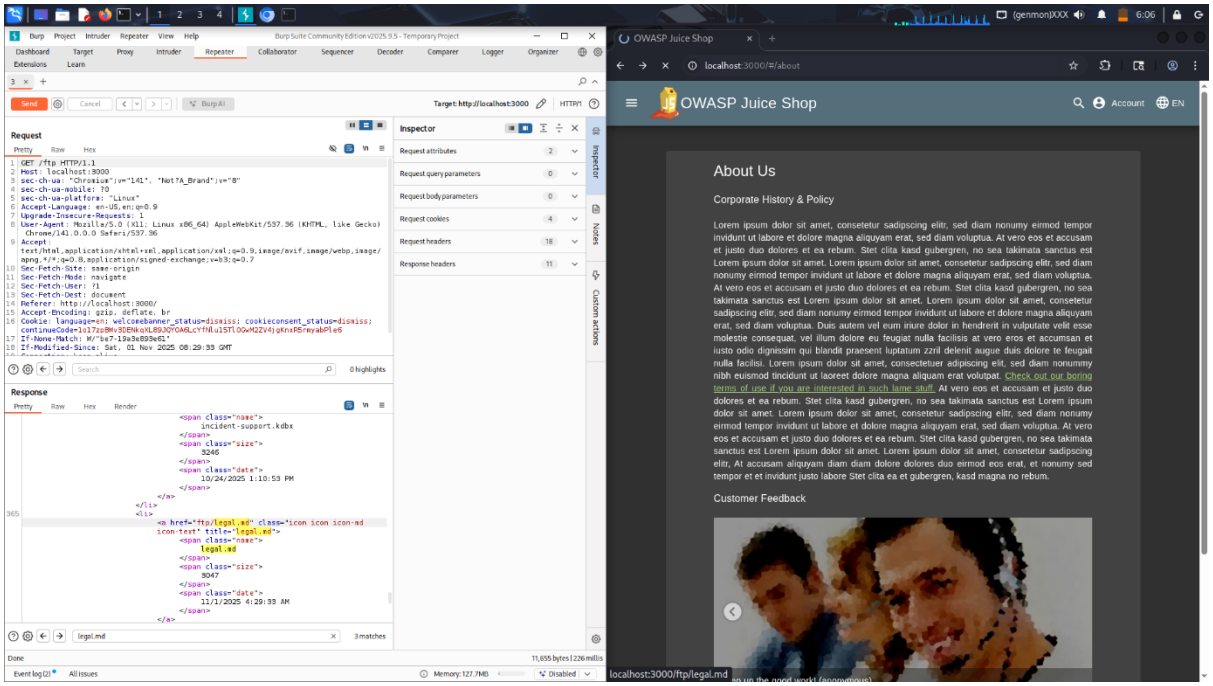


Figure 4 Sensitive Data Exposure

3.5 Information Disclosure (Admin Email)

OWASP Category	Location	Likelihood	Impact
Sensitive Data Exposure	Products' Reviews	High	Medium

The application's server provided verbose error messages in response to certain requests. These responses inadvertently disclosed sensitive information, specifically the email address of the system administrator, which could be leveraged for future phishing or brute-force attacks.

Remediation: Configure the web server and application to display generic, non-informative pages in a production environment. Suppress technical details, stack traces and internal contact information from all client-facing error messages.

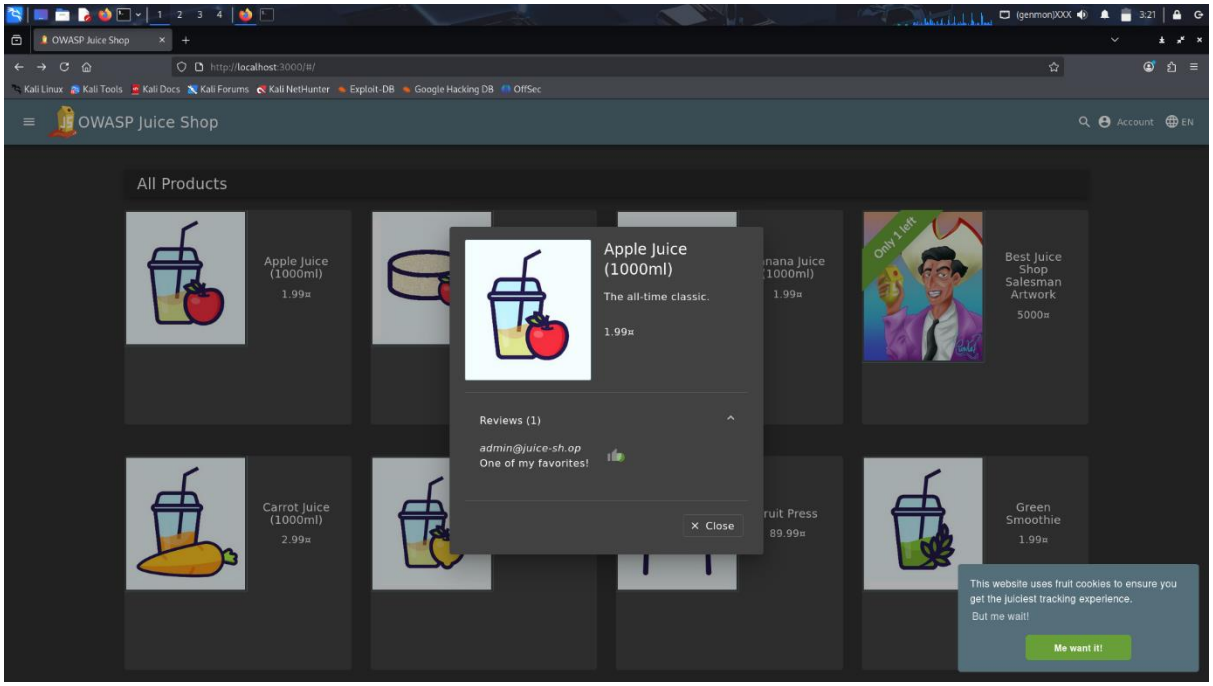


Figure 5 Information Disclosure of Admin Email

4. Conclusion and Recommendations

The OWASP Juice Shop assessment successfully fulfilled the requirements of Future Interns task 1 by identifying and exploiting five critical and high-risk vulnerabilities. The findings underscore the critical necessity of implementing security throughout the entire software development lifecycle.

4.1 Key Recommendations

- **Immediate Fix:** Fix the SQL Injection and Broken Access Control flaws immediately. These represent the highest risk to the application's integrity.
- **Input Handling:** Implement a comprehensive, centralized input validation and encoding library to handle both server-side and client-side data, mitigating XSS and Injection risks.
- **Security Configurations:** Review server configuration to ensure sensitive files are not exposed and that detailed error messages are disabled for end users.