Web Application Security Assessment Report

Project Name: Cyber Security Task 1

Target Application: OWASP Juice Shop

Date: December 9, 2025

Tester Name: Rathnam S.

1. Executive Summary - The Takeaway

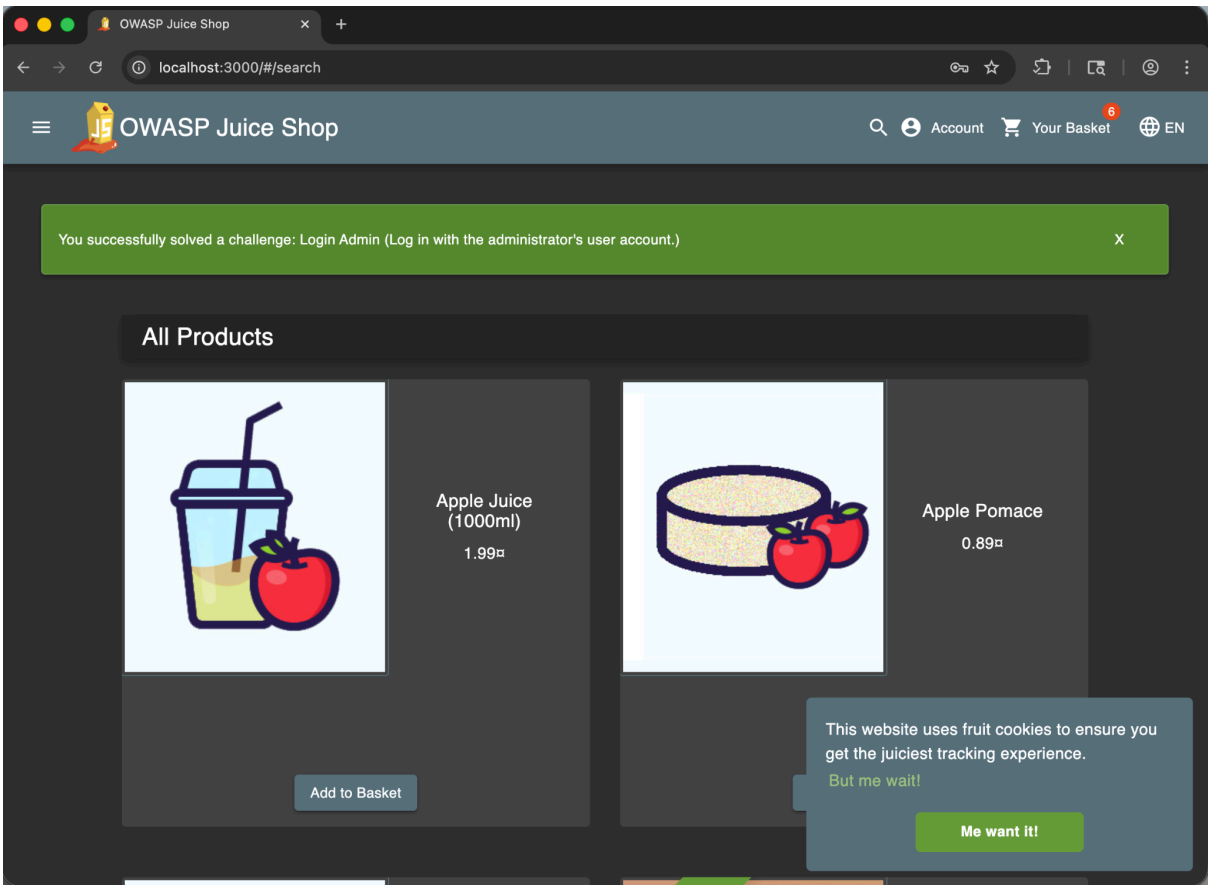
This report is about a security test I ran on the OWASP Juice Shop application. My main goal was to find real-world weaknesses, especially those covered by the infamous OWASP Top 10 list

I'm happy to report I successfully got in and proved three major flaws:

- **I became the Admin (SQL Injection):** I managed to log in as the system administrator without knowing the password.
- **I injected my own code (Cross-Site Scripting):** I made the application run custom JavaScript code in a user's browser.
- **I found the secret map (Sensitive Data Exposure):** I found a hidden page with sensitive internal data just by guessing the right address.

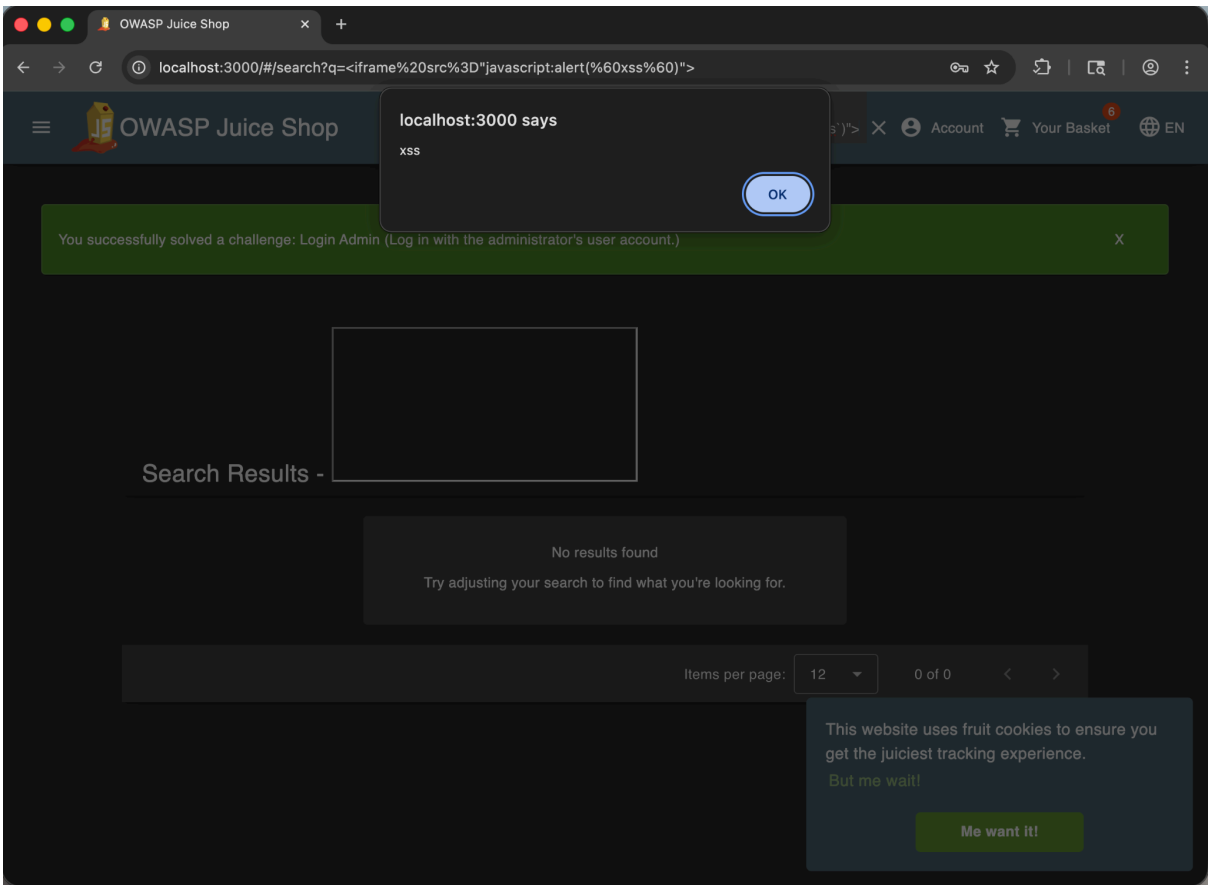
2. Technical Findings - Where It BrokeFinding 1: Sneaky SQL Injection (I'm the Admin now)

- **Vulnerability Type:** Injection (OWASP A03:2021)
- **Risk Level: CRITICAL - Game Over**
- **What happened:** The login page is a total pushover. By typing a simple trick phrase into the email box, I tricked the database into thinking my password check was already successful. It essentially said, "Hey, I don't care about the password, just log me in."
- **The Cheat Code:** `' OR 1=1 --`
- **Proof of Concept:**



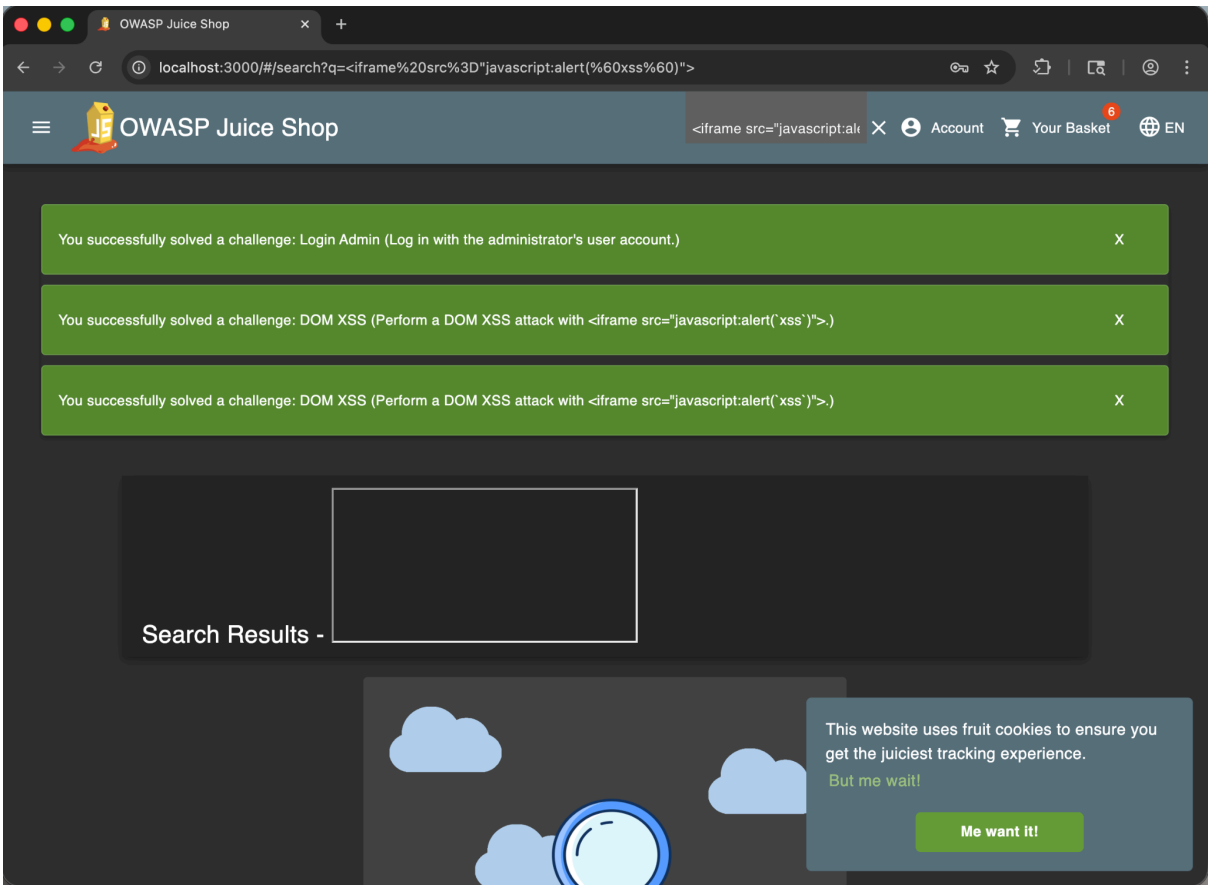
Finding 2: DOM-Based Cross-Site Scripting (The Pop-up Prank)

- **Vulnerability Type:** Injection / Insecure Design (OWASP A03:2021)
- **Risk Level: High - Major Pain**
- **What happened:** The search bar is too trusting. Whatever you type, it displays right back on the page without cleaning it up. I exploited this by inserting a tiny piece of HTML that forces the browser to run a script, showing how an attacker could steal cookies or redirect users.
- **The Cheat Code:** `<iframe src="javascript:alert('xss')">`
- **Proof of Concept:**



Finding 3: Sensitive Data Exposure (The Score Board Secret)

- **Vulnerability Type:** Broken Access Control (OWASP A01:2021)
- **Risk Level: Medium - Easily Avoidable**
- **What happened:** The developers tried to hide the sensitive "Score Board" page by just not putting a link to it (a practice called "Security through Obscurity"). However, the page has zero protection. If you know the address, you're in, revealing all the application's solved challenges to anyone.
- **The Secret Address:** `http://localhost:3000/#/score-board`
- **Proof of Concept:**



3. Recommendations - How to Fix It

Here’s the simple path to a more secure application:

1. **Stop SQL Injection:** Always use **Prepared Statements** (parameterized queries). This physically separates the user's input data from the actual database command, making injection impossible.
2. **Stop XSS:** Don't trust user input! You must **sanitize and encode** all user-provided data before showing it on the page so the browser treats it as plain text, not executable code.
3. **Fix Access Control:** Every critical page (like the Score Board) needs a **server-side bouncer**. The application must check the user's permissions *before* loading the page content, regardless of whether the user guessed the URL or not.