WEB APPLICATION VULNERABILITY ASSESSMENT REPORT

Internship Program: Future Interns – Cyber Security
Task: Task 1 – Web Application Security Testing
Candidate Name: Shruti Mate
Target Application: OWASP Juice Shop
Target URL: http://localhost:3000
Testing Environment: Localhost
Primary Tool: OWASP ZAP
Testing Date: December 2025

---

1. Introduction

Web applications are frequent targets of cyberattacks due to improper input validation, weak authentication mechanisms, and insecure access control implementations.

As part of the Cyber Security Internship at Future Interns, this security assessment was conducted to identify common web application vulnerabilities using industry-standard tools and OWASP guidelines.

The objective of this task was to simulate a real-world vulnerability assessment by performing both automated and manual security testing on **OWASP Juice Shop**, an intentionally vulnerable application designed for learning and training purposes.

---

2. Testing Environment

- **Application:** OWASP Juice Shop
- **Deployment:** Localhost (Node.js)
- **URL:** http://localhost:3000
- **Testing Approach:** Black-box testing

---

3. Methodology

The assessment followed a hybrid testing methodology commonly used in professional penetration testing engagements:

1. **Automated Scanning:**
   OWASP ZAP was used to perform an automated vulnerability scan to identify potential security weaknesses such as injection flaws, misconfigurations, and information disclosure.
2. **Manual Testing & Validation:**
   Critical findings were manually tested using crafted payloads to confirm exploitability and assess real-world impact, reducing false positives.

This approach ensures accurate and reliable security findings.

---

4. Tools Used

- **OWASP ZAP** – Automated vulnerability scanning and alert analysis
- **Web Browser** – Manual exploitation and payload testing
- **OWASP Juice Shop Challenges** – Validation of exploitation impact

---

5. Vulnerability Findings

5.1 Authentication Bypass (SQL Injection)

OWASP Category: A03 – Injection
Severity: High

Description:
The login mechanism is vulnerable to SQL Injection, allowing authentication bypass using crafted input. This flaw enabled access to administrative functionality without valid credentials.

Payload Example:

```
' OR 1=1--
```

Impact:

- Unauthorized login as admin
- Direct access to privileged functionality

Remediation:

- Use parameterized queries and prepared statements
- Avoid dynamic SQL query construction
- Apply server-side input validation

5.2 Broken Access Control (Admin Function Abuse)

OWASP Category: A01 – Broken Access Control
Severity: High

Description:
After bypassing authentication, administrative functionalities are accessible without proper role validation. This allows abuse of admin features such as deletion of feedback and viewing sensitive user data.

Steps to Reproduce:

1. Bypass admin login using authentication flaw
2. Log in as administrative user
3. Navigate to:
4. http://localhost:3000/#/administration
5. Observe unrestricted access to admin features

Impact Observed:

- Deletion of customer feedback (Five-Star Feedback challenge)
- Viewing list of administrator accounts
- Execution of privileged operations

Remediation:

- Implement strict role-based authorization checks
- Validate user privileges on each request
- Apply defense-in-depth beyond authentication

5.3 Cross-Site Scripting (XSS)

OWASP Category: A03 – Injection / A07 – Cross-Site Scripting
Severity: Medium

Description:
User input is reflected without proper sanitization, allowing execution of malicious JavaScript.

Payload Used:

```
<img src=x onerror=alert('SHRUTI')>
```

Impact:

- Arbitrary JavaScript execution
- Session hijacking
- Client-side attacks such as phishing

Remediation:

- Sanitize and validate input
- Encode output before rendering
- Implement Content Security Policy (CSP)

5.4 Sensitive Information Disclosure

OWASP Category: A02 – Cryptographic Failures
Severity: Medium

Description:
Administrative user information is exposed without encryption or masking, accessible after authentication bypass.

Impact:

- User enumeration
- Targeted attacks on privileged accounts
- Privilege escalation opportunities

Remediation:

- Mask sensitive data
- Enforce strict authentication and authorization checks
- Limit exposure of sensitive resources

5.5 Security Misconfiguration

OWASP Category: A05 – Security Misconfiguration
Severity: Medium

Description:
Missing security headers and exposed unnecessary routes increase the attack surface.

Impact:

- Exploitation of other vulnerabilities easier
- Increased system exposure

Remediation:

- Configure security headers (CSP, HSTS, X-Frame-Options)
- Disable unused endpoints
- Conduct regular security audits

6. OWASP Top 10 (2021) Mapping Summary

OWASP ID	Category	Status	Evidence
A01	Broken Access Control	Identified	Admin functions accessible after bypassing authentication
A02	Cryptographic Failures	Identified	Exposure of admin user information
A03	Injection	Identified	SQL Injection login bypass & XSS payload
A04	Insecure Design	Identified	Missing authorization logic for admin features
A05	Security Misconfiguration	Identified	Missing headers & unnecessary routes
A06	Vulnerable & Outdated Components	Identified	Vulnerable JS libraries detected
A07	Identification & Authentication Failures	Identified	Authentication bypass vulnerability
A08	Software & Data Integrity Failures	Identified	Lack of integrity validation for client-side scripts
A09	Security Logging & Monitoring Failures	Identified	No alerting or monitoring for unauthorized admin access
A10	Server-Side Request Forgery (SSRF)	Identified	Potential uncontrolled URL access patterns observed

Note: Some categories were identified via design weakness and configuration observations, which aligns with real-world pen testing methodology.

---

7. Risk Summary

Severity	Count
High	2
Medium	3
Low / Informational	Remaining ZAP scan results

---

8. Learning Outcomes

- Practical understanding of OWASP Top 10 vulnerabilities
- Hands-on experience with OWASP ZAP
- Improved knowledge of attack vectors and secure coding practices
- Ability to document vulnerabilities professionally

---

9. Challenges Faced

- Configuring OWASP ZAP for correct proxy interception
- Initial setup of Juice Shop environment
- Differentiating true positives from false alerts

---

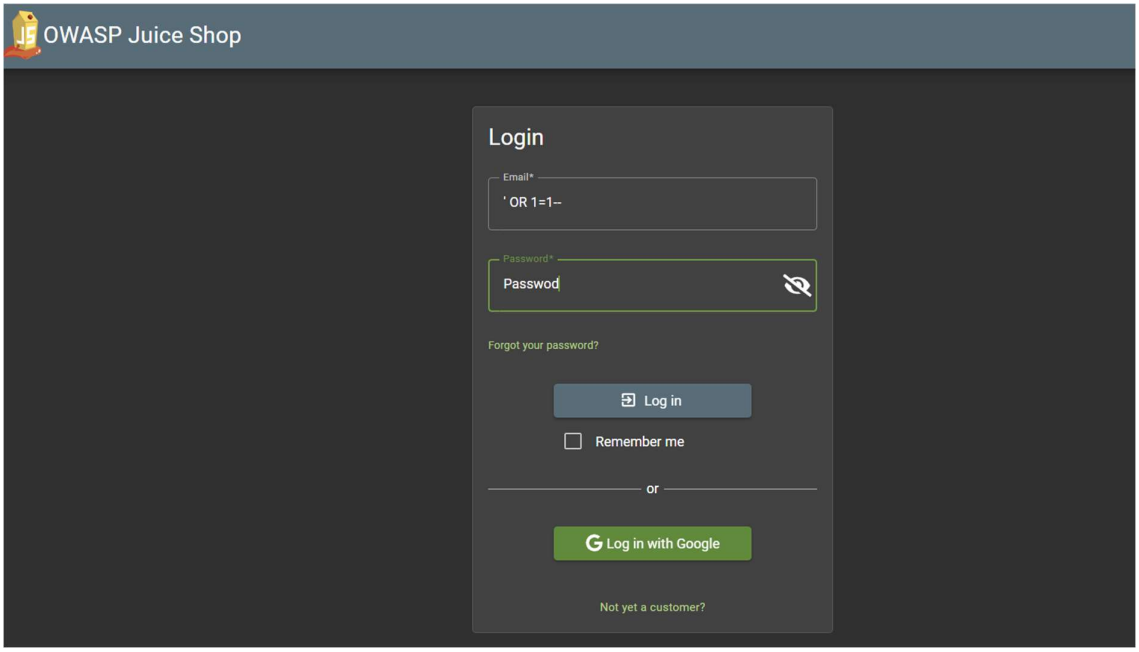
10. Conclusion

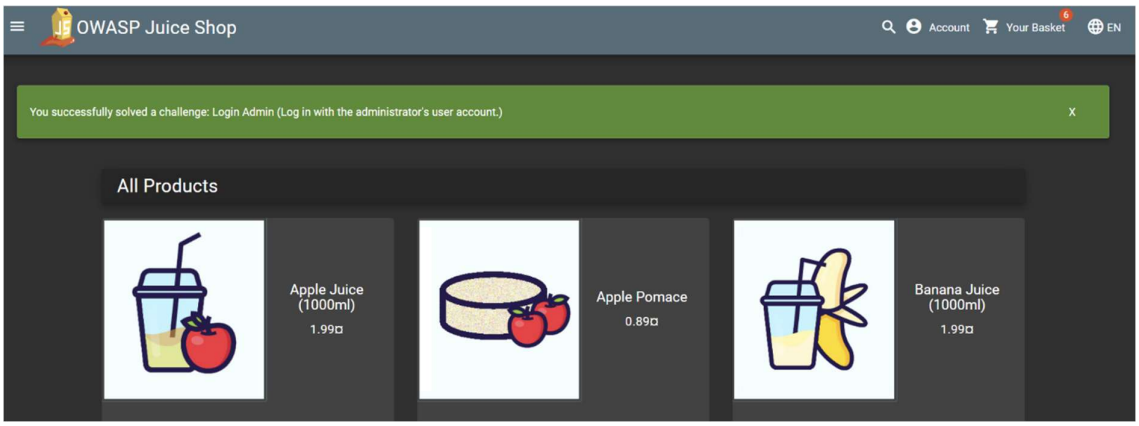
The assessment demonstrated several high- and medium-risk vulnerabilities, including authentication bypass, broken access control, XSS, and sensitive data exposure. Addressing these issues via secure coding, proper access controls, and regular security testing will improve application security posture significantly.

---

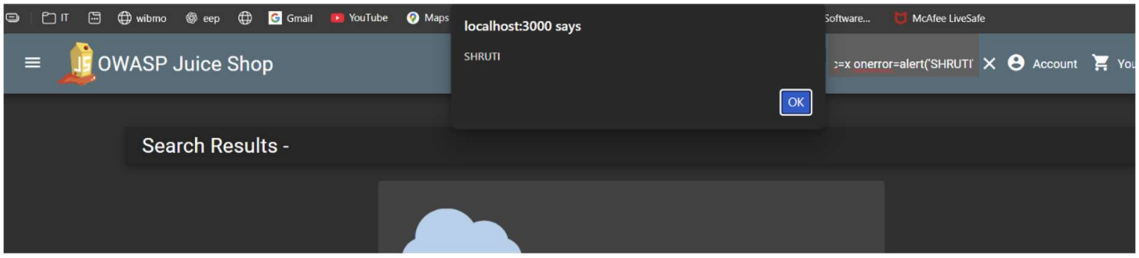
11. Evidences

1. SQL Injection

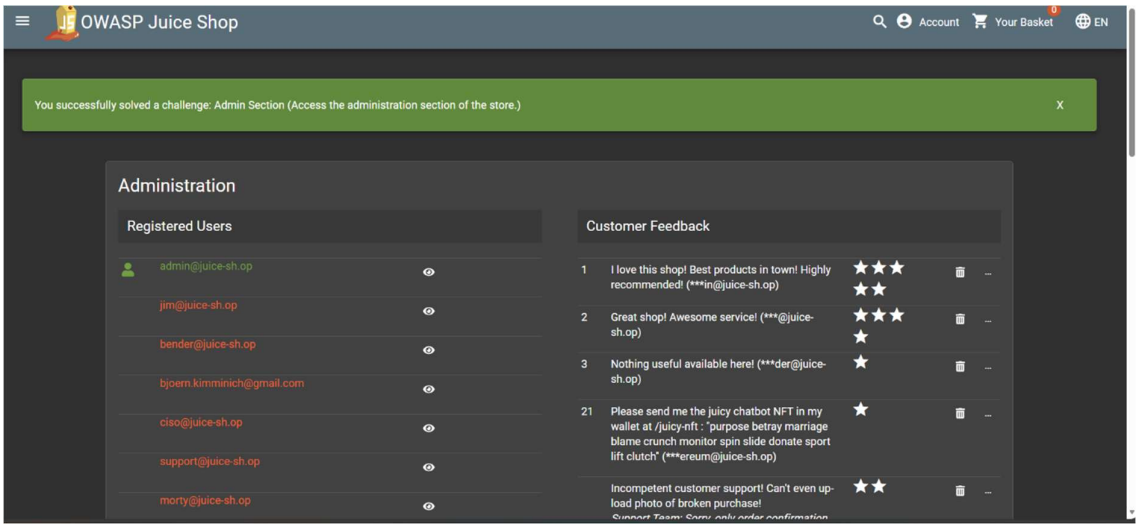




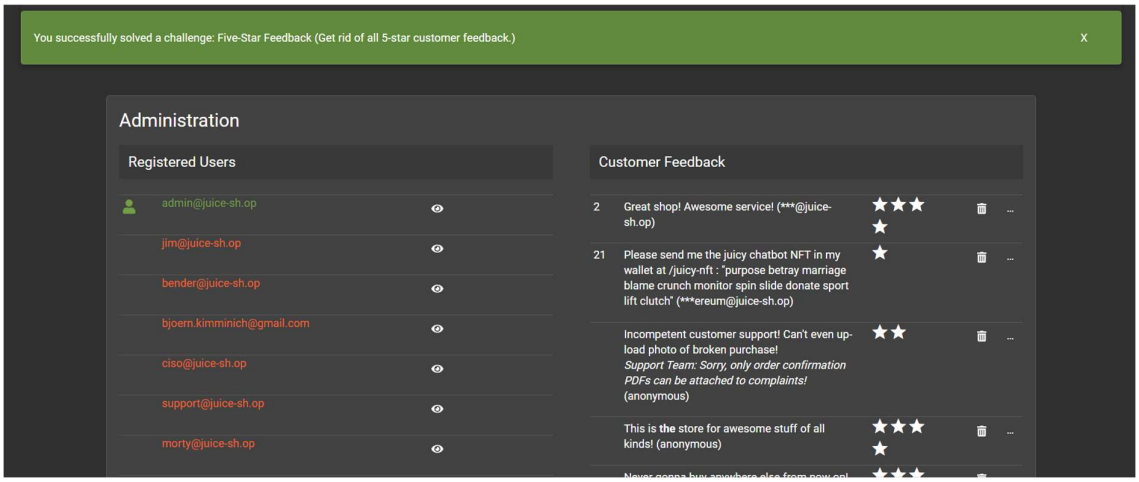
2. Cross-Site Scripting



3. Broken Authentication



4. Administration login and performing feedback delete operation



5.