WEB APPLICATION SECURITY ASSESSMENT REPORT

NAME: IBE KINGSLEY

TASK 1 : WEB APPLICATION SECURITY TESTING

PROGRAM : FUTURE INTERNS - CYBERSECURITY INTERNSHIP

DATE: SEPTEMBER 2025

TARGET APPLICATION : OWASP JUICE SHOP (An Intentional Vulnerable App)

Task Summary

This report documents the findings of a manual security assessment conducted on OWASP Juice Shop. The vulnerabilities identified include SQL Injection, JWT Credential Exposure, Insecure Direct Object Reference (IDOR), Cross-Site Scripting (XSS), and Replay Vulnerability. Each issue is mapped to the OWASP Top 10 (2021) and supported with technical evidence and mitigation strategies.

Tools Utilized

- *Kali Linux – Penetration testing environment*
- *OWASP Juice Shop – Intentionally vulnerable web application*
- *Burp Suite – Web vulnerability scanner and proxy tool*

Procedure Overview

- *The OWASP Juice Shop was deployed locally on a Linux virtual machine, with all dependencies (Node.js and npm) installed via the command line.*
- *The local instance served as the target for both automated and manual security testing.*
- *Burp Suite was configured to intercept traffic and perform active scanning to identify input vectors and potential vulnerabilities.*
- *All discovered issues were manually validated and categorized according to the OWASP Top 10 (2021) framework.*
- *Remediation strategies were proposed for each identified vulnerability to enhance the application's security posture.*

5 IDENTIFIED VULNERABILITIES IN OWASP JUICE SHOP

1. SQL Injection in Login Function

During the assessment of the login feature on OWASP Juice Shop, it was discovered that the application is vulnerable to SQL injection. By entering a crafted input (' OR 1=1--) into the email field, the system bypassed authentication and granted access to a privileged account without verifying credentials.
This behavior indicates that the backend fails to properly sanitize user input before executing SQL queries, allowing attackers to manipulate the logic of the database.

Security Impact

- *Unauthorized access to sensitive user accounts, including administrative privileges.*
- *Potential exposure of customer data and internal application logic.*
- *Risk of full system compromise if chained with other vulnerabilities.*

***Risk Rating** : High*

Evidence Collected

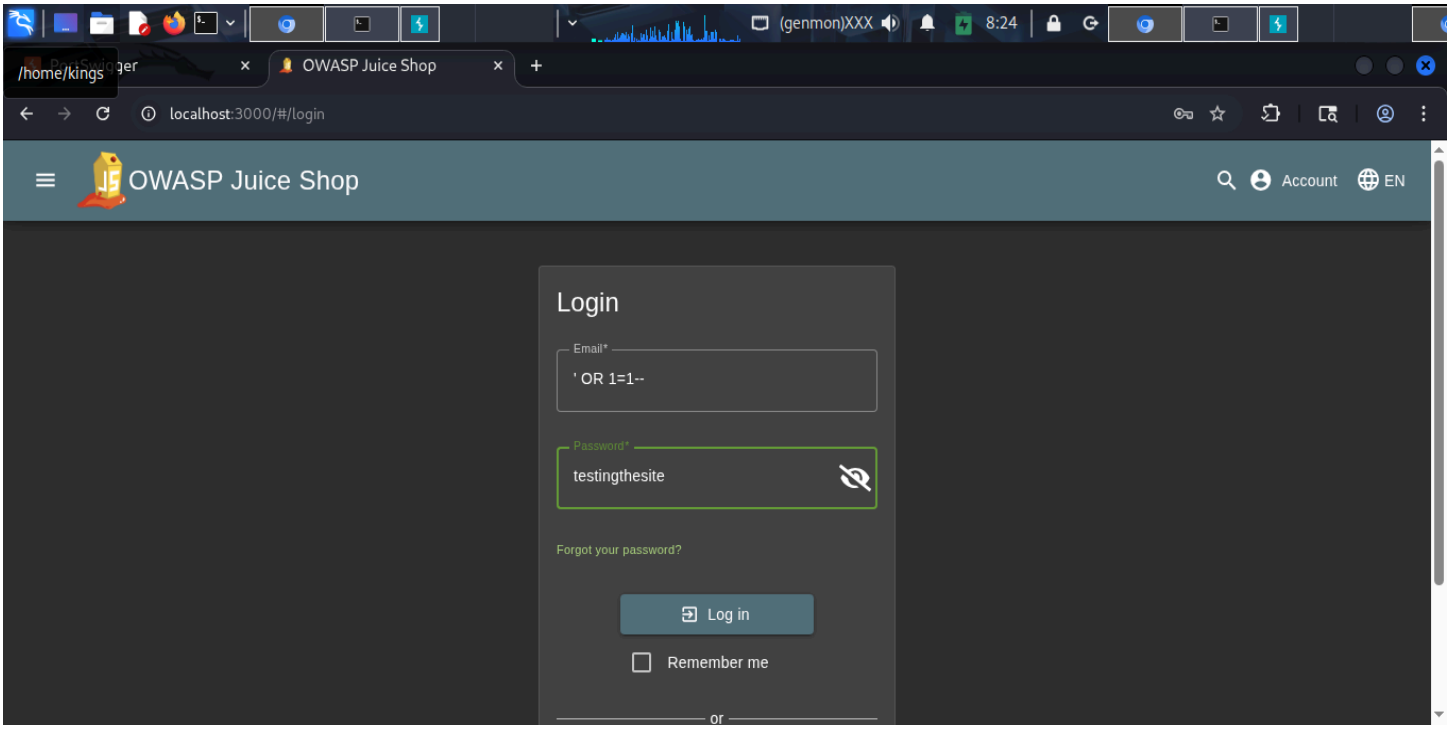

***Figure 1** SQLi Payload in Login Form*

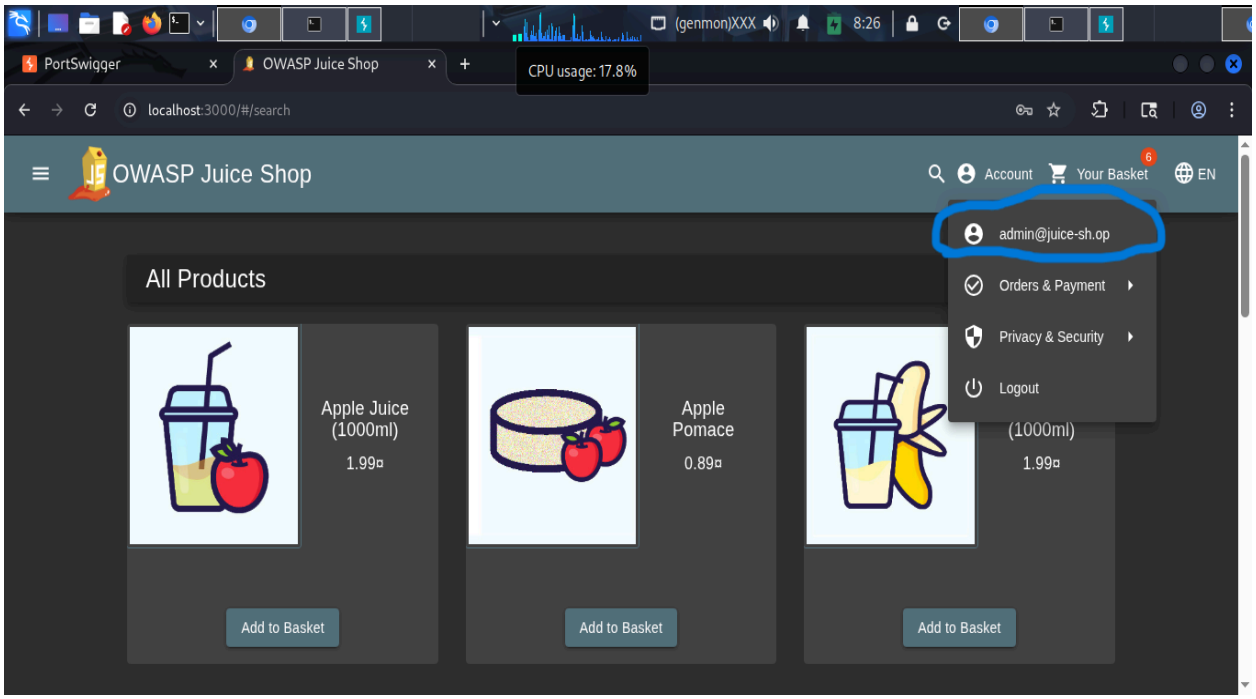

Figure 2 *: Post-login view showing access to the admin account.*

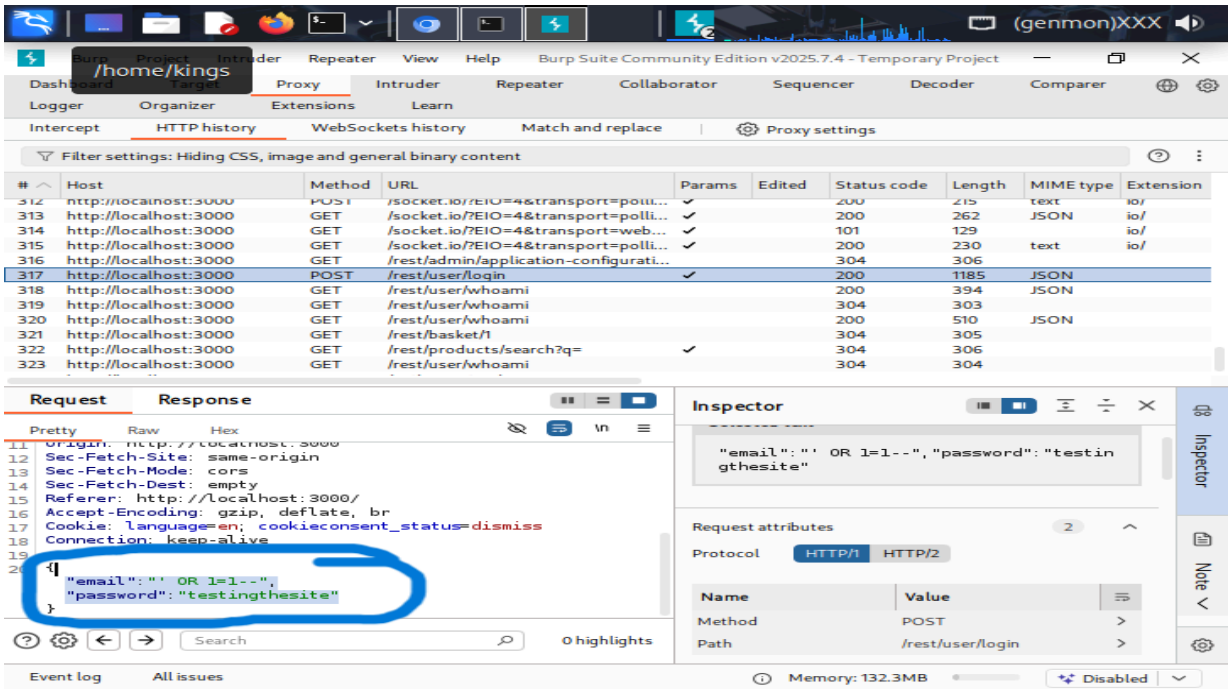

Figure 3*: Burp Suite intercepted request containing the malicious payload.*

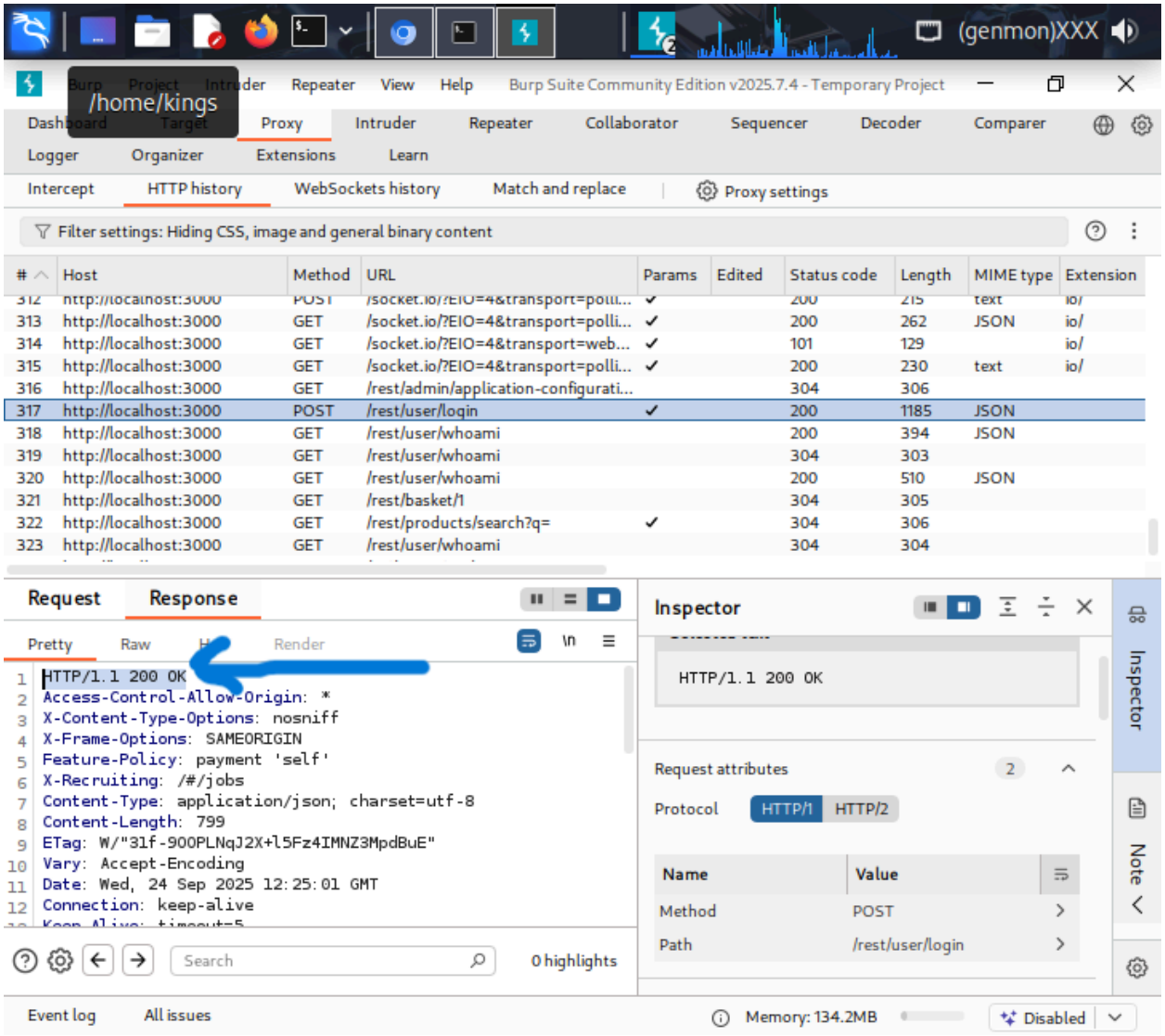

***Figure 4**: Burp Suite response showing HTTP status 200 OK, confirming successful login.*

Mitigation Strategies

- *Implement parameterized queries to prevent direct injection into SQL statements.*
- *Apply strict input validation and sanitization on all user-supplied data.*
- *Limit database privileges for application accounts to reduce impact if compromised*

2. *Insecure Token Design – Credential Exposure in JWT*

The application embeds user credentials inside JWTs using MD5 hashing. MD5 is a weak and reversible algorithm, making it possible for attackers to crack the hash and recover plaintext passwords. This exposes users to account takeover and undermines the integrity of the authentication system.

Security Impact

- *Reveals hashed credentials to attackers*
- *Enables password cracking and unauthorized access*
- *Violates secure token design principles*

***Risk Rating**: High*

Evidence Collected

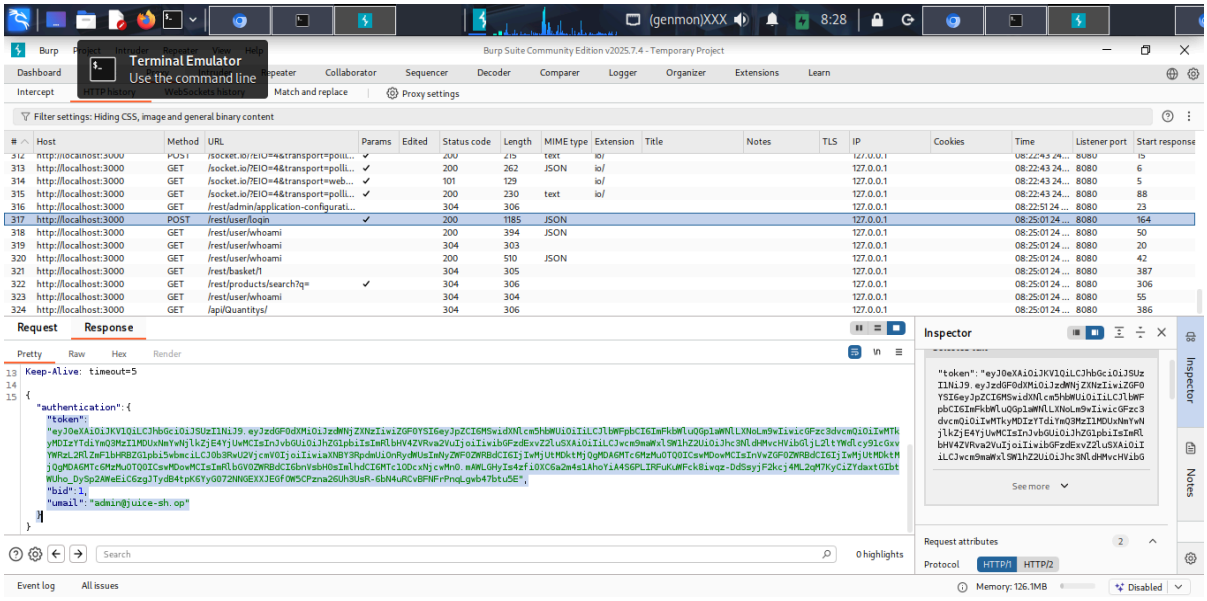

***Figure 1**; JWT token captured in Burp Suite*

Mitigation Strategies

- *Never store credentials in tokens*
- *Replace MD5 with secure hashing algorithms like bcrypt or Argon2*
- *Ensure JWTs only contain non-sensitive claims and use strong signing methods*

3. *Insecure Direct Object Reference (IDOR)*

The application exposes internal object identifiers (e.g., userId) in client-side requests. Without proper authorization checks, these values can be manipulated to access or modify data belonging to other users. This flaw allows attackers to bypass access controls and compromise user privacy.

Security Impact

- *Unauthorized access to other users' data*
- *Potential data modification or deletion*
- *Violation of user confidentiality*

***Risk Rating**: High*

Evidence Collected.

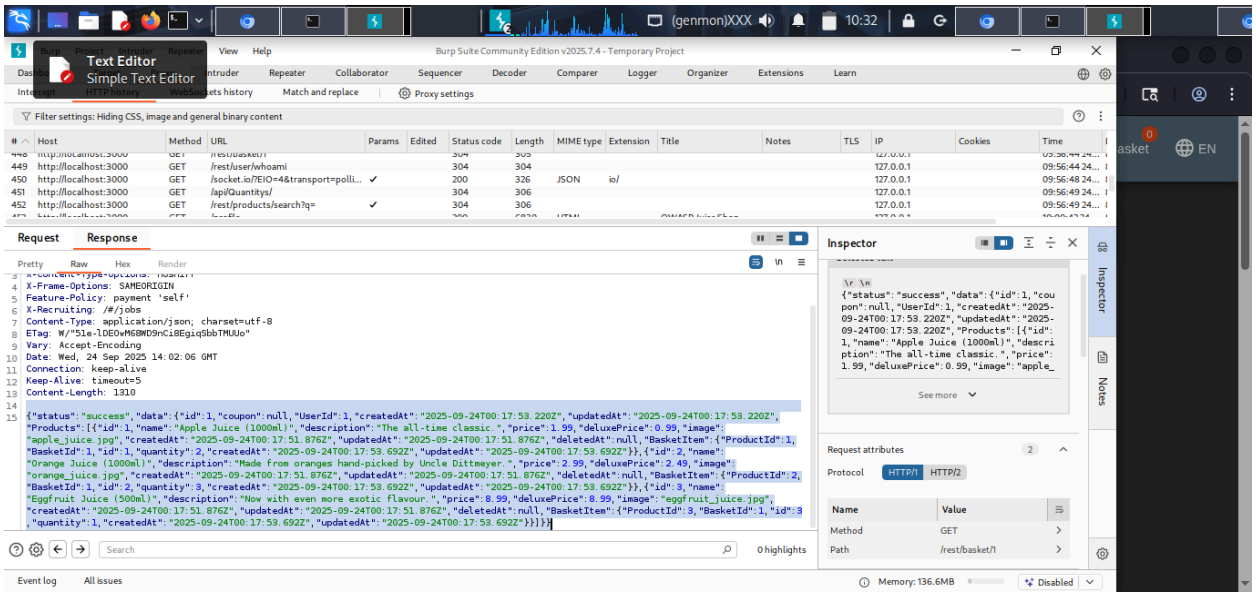

***Figure 1**: Screenshot of Burp Suite showing the manipulated request and successful response*

Mitigation Strategies

- *Enforce strict authorization checks on every object access*
- *Avoid exposing predictable identifiers in client-side requests*
- *Use indirect references or access tokens instead of raw IDs*

4. *Cross-Site Scripting (XSS) – DOM-Based Execution via Feedback Form*

The application fails to properly sanitize user input submitted through the Customer Feedback form. By injecting a malicious script into the comment field, it was possible to execute arbitrary JavaScript in the browser. This confirms a DOM-based XSS vulnerability, which could be exploited to hijack sessions, deface content, or redirect users to malicious sites.

Security Impact

- *Execution of unauthorized scripts in the user's browser*
- *Potential for session hijacking, phishing, or UI manipulation*
- *Violation of client-side security controls*

***Risk Rating**: High*

Evidence Collected

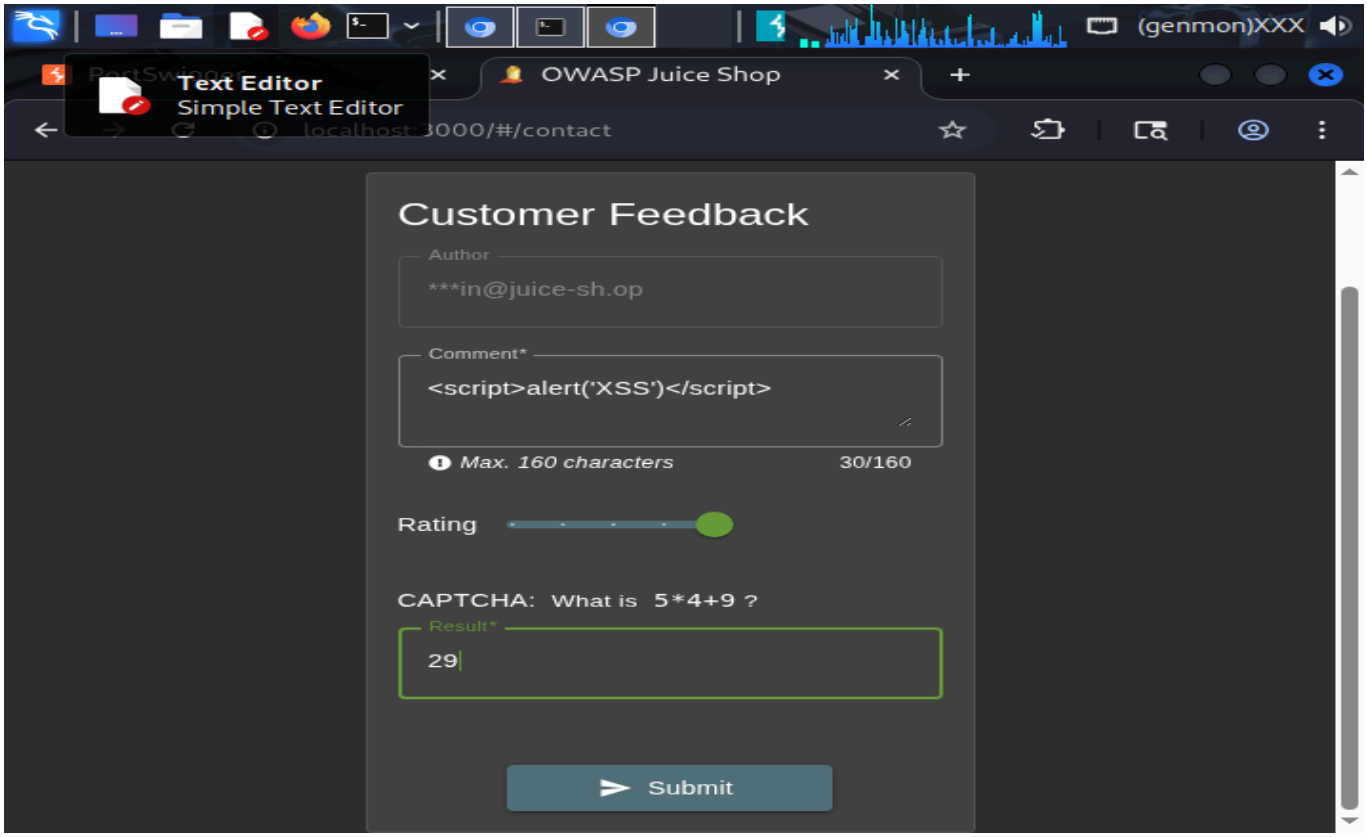

***Figure 1**: Feedback form with XSS payload (<script>alert('XSS')</script>) entered in the comment field*

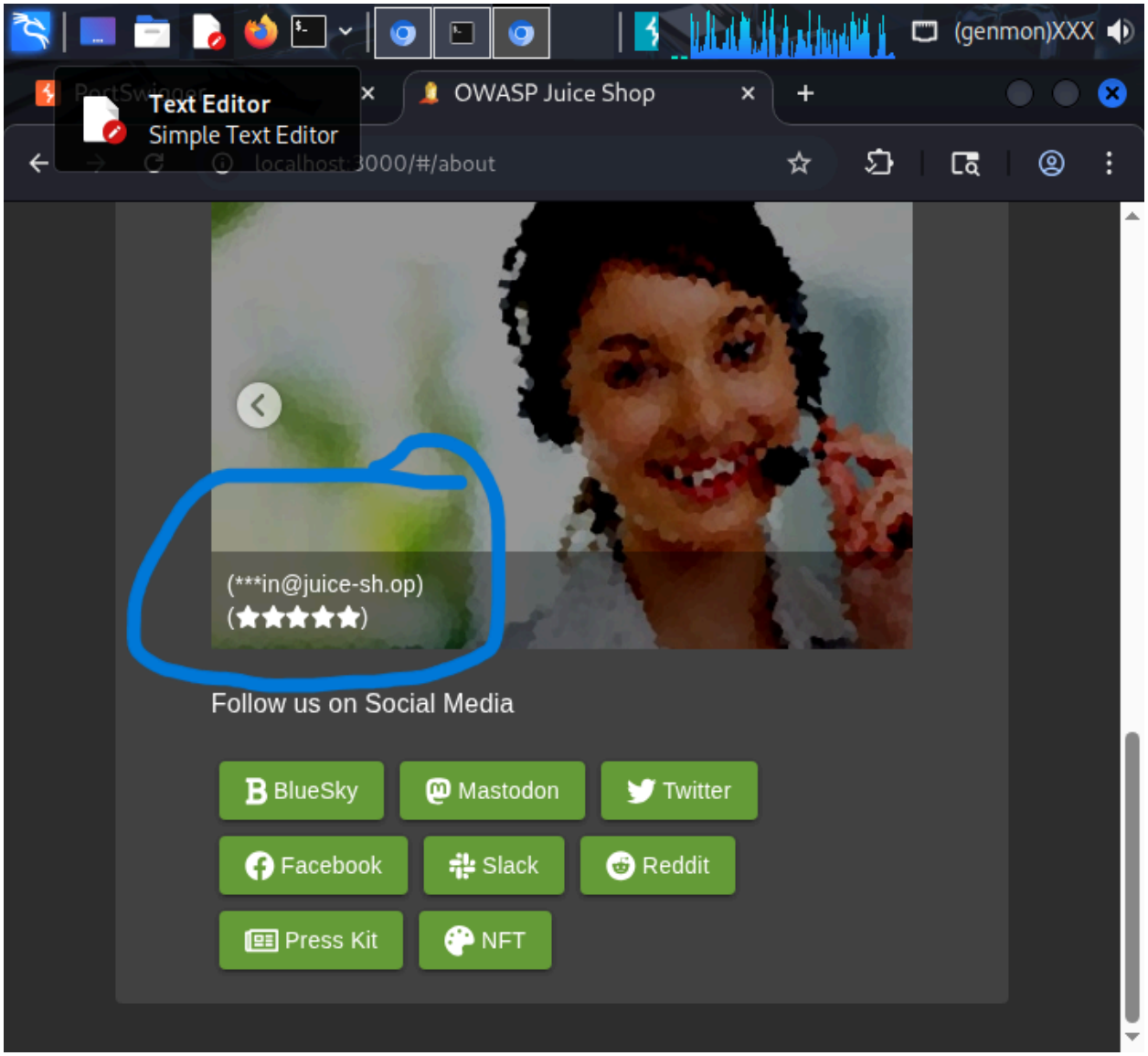

***Figure 2**: Alert box triggered after visiting the About Us page, confirming script execution*

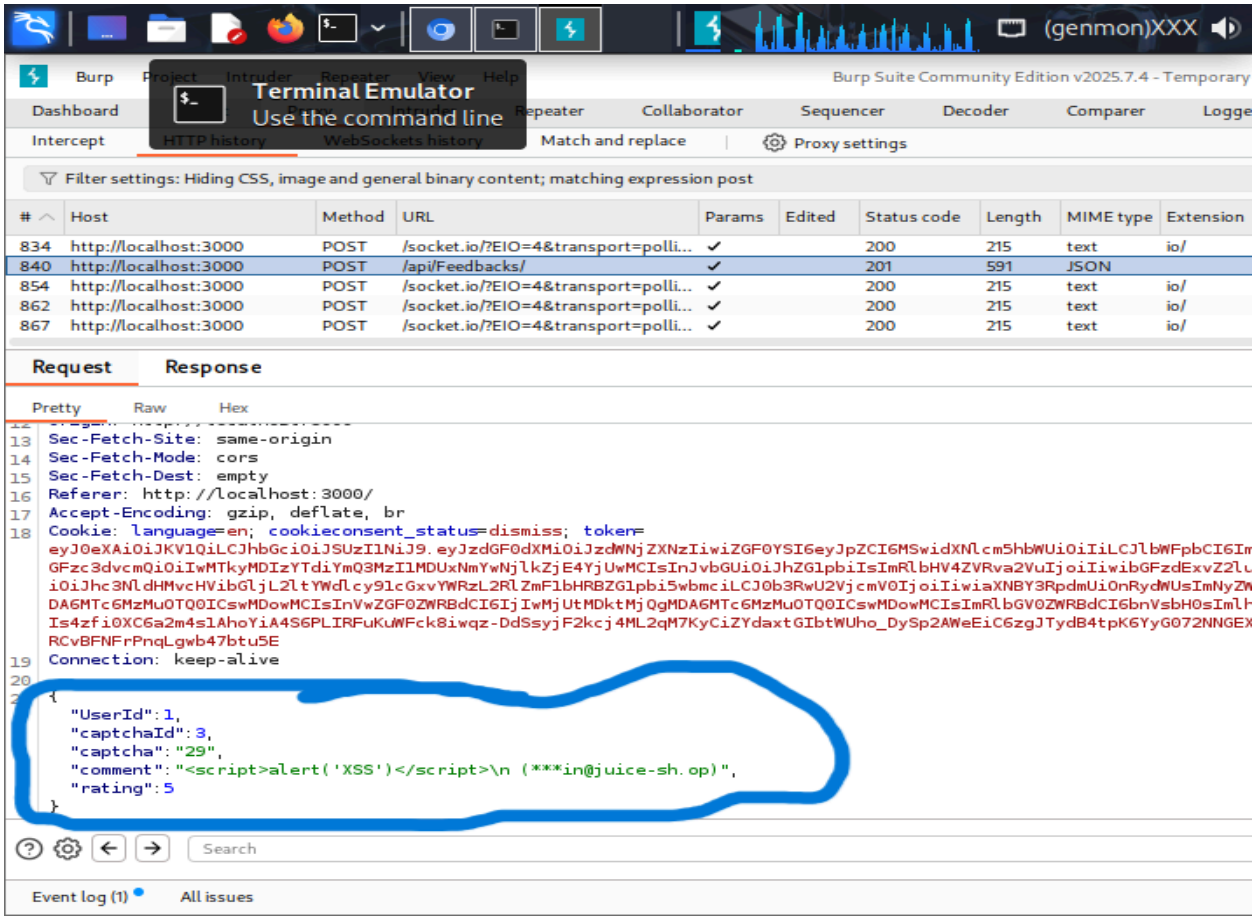

***Figure 3**: Burp Suite request showing the injected payload in the POST /api/Feedback request*

Mitigation Strategies for the Cross-Site Scripting (XSS)

- *Sanitize and encode all user input before rendering in the DOM*
- *Use libraries like DOMPurify to clean HTML content*
- *Implement a strict Content Security Policy (CSP) to block inline scripts*

5. *Replay Vulnerability in Feedback Submission*

The feedback feature in OWASP Juice Shop accepts repeated submissions of identical requests without enforcing uniqueness or freshness. This behavior confirms a replay vulnerability, where previously captured requests can be resent and accepted multiple times. Although attempts to impersonate other users by modifying the userId field were unsuccessful, the lack of anti-replay protections exposes the system to abuse.

Security Impact

- *Enables attackers to flood the system with duplicate feedback*
- *May lead to spam, denial-of-service conditions, or manipulation of feedback metrics*
- *Demonstrates weak request validation and broken authentication logic*

***Risk Rating**: Medium*

Evidence Collected

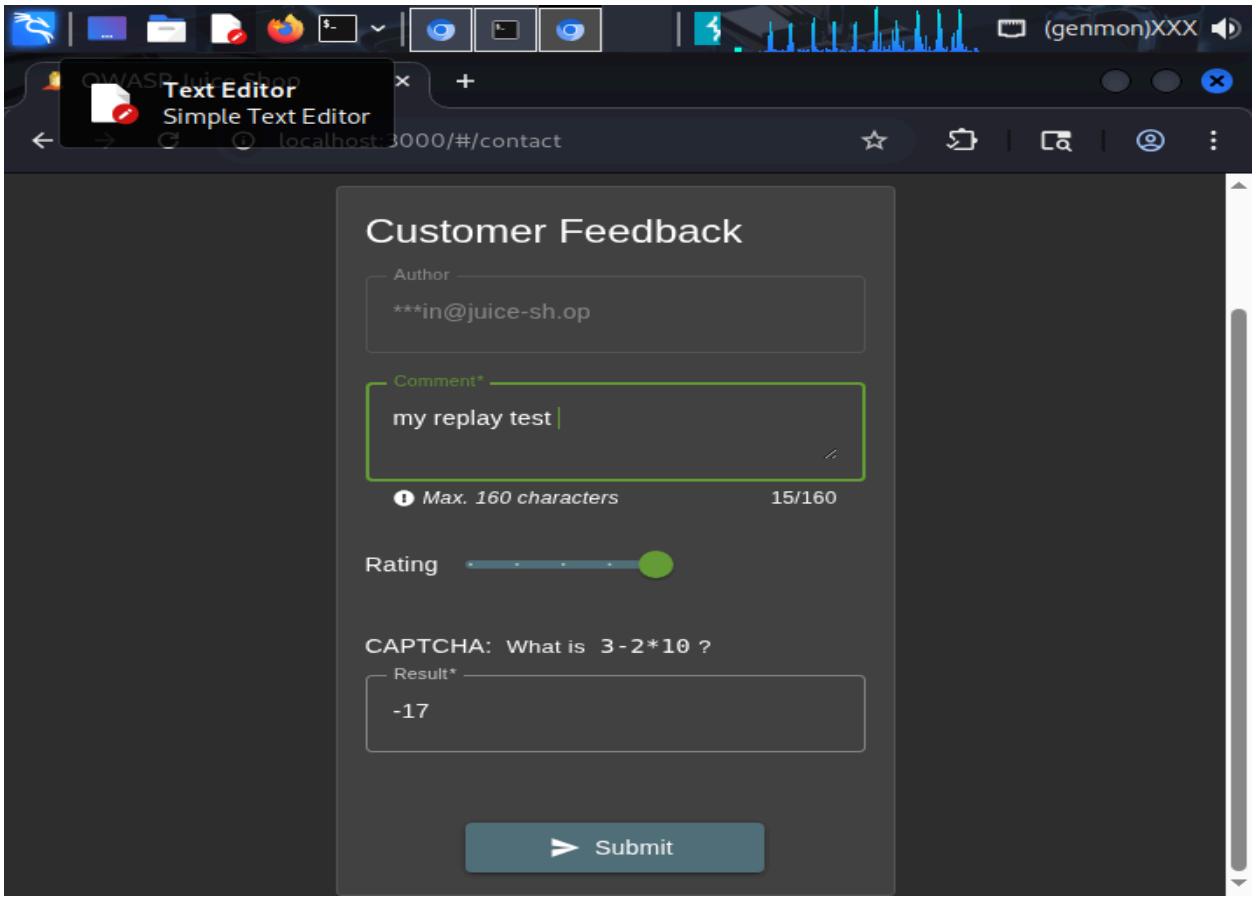

***Figure 1**: Original feedback request submitted through the Customer Feedback form*

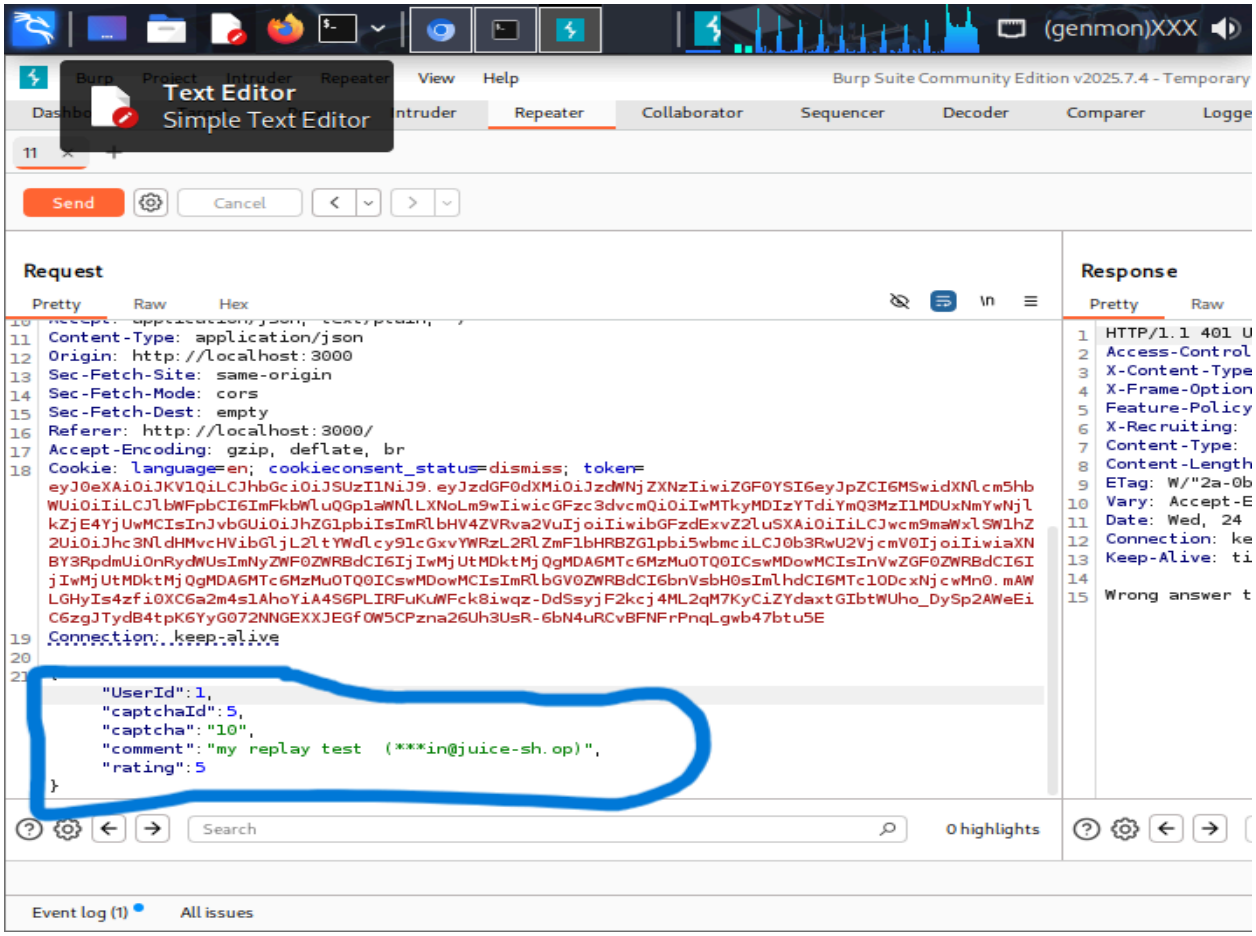

***Figure 2**: Replayed request sent via Burp Suite Repeater and accepted multiple times*

Mitigation Strategies

- *Implement anti-replay mechanisms such as timestamps, nonces, or one-time tokens*
- *Validate session identity server-side for all feedback submissions*
- *Enforce rate limiting and feedback uniqueness to prevent abuse*

Vulnerability Summary Table Mapping to OWASP Top 10

Vulnerability	***OWASP Category (2021)***	***Impact***	***Risk Rating***
SQL Injection	*A03:2021 – Injection*	*Authentication bypass and unauthorized access*	***High***
JWT Credential Exposure	*A02:2021 – Cryptographic Failures*	*Password cracking and account takeover*	***High***
Insecure Direct Object Reference (IDOR)	*A01:2021 – Broken Access Control*	*Unauthorized access to other users' data*	***High***
Cross-Site Scripting (XSS)	*A07:2021 – Identification & Authentication Failures*	*Arbitrary script execution in browser*	***High***
Replay Vulnerability	*A01:2021 – Broken Access Control*	*Duplicate feedback accepted without validation*	***Medium***

Conclusion

*This assessment of the **OWASP Juice Shop** application uncovered five critical vulnerabilities that align with the **OWASP Top 10 (2021) framework**. These include SQL Injection, JWT Credential Exposure, Insecure Direct Object Reference (IDOR), Cross-Site Scripting (XSS), and Replay Vulnerability in the feedback system. Each issue was validated through manual testing using Burp Suite and supported with clear evidence and mitigation strategies.*

The vulnerabilities demonstrate weaknesses in input validation, token design, access control, and client-side security. If exploited, they could lead to unauthorized access, account takeover, data leakage, and abuse of application functionality.

***To improve the security posture of the application, it is recommended to**:*

- *Apply secure coding practices and robust input validation*
- *Use modern cryptographic standards for token handling*
- *Enforce strict identity and session validation*
- *Implement anti-replay mechanisms and rate limiting*
- *Apply access control checks on all sensitive operations*

This report highlights the importance of proactive vulnerability management and secure development practices in modern web applications.