Security Analysis of a Web Application Using OWASP ZAP”

Project Title:

Web Application Security Testing using OWASP ZAP

Intern Name:

Kanishka Mishra

Internship Program:

Cyber Security Intern – Future Interns

Tool Used:

OWASP ZAP 2.17.0

Target Application:

OWASP Juice Shop (http://localhost:3000)

Date:

20th December

1. INTRODUCTION

With the rapid growth of web-based applications, security has become a critical concern in today's digital era. Web applications are widely used to store, process, and transmit sensitive information, making them attractive targets for cyber-attacks. Vulnerabilities such as cross-site scripting (XSS), security misconfigurations, and improper data handling can lead to serious security breaches if not identified and mitigated in time.

Web application security testing plays a vital role in identifying potential weaknesses before they can be exploited by attackers. One of the most widely used tools for this purpose is **OWASP Zed Attack Proxy (ZAP)**. OWASP ZAP is an open-source security testing tool designed to help developers and security testers automatically find vulnerabilities in web applications during development and testing phases.

In this project, an automated vulnerability assessment was performed on a web application using OWASP ZAP. The tool was used to scan the application, analyze HTTP requests and responses, and detect common security issues based on industry standards such as the **OWASP Top 10**. The objective of this assessment is to understand common web application vulnerabilities, evaluate the security posture of the application, and generate a detailed security report.

This study helps in gaining practical exposure to web application security testing and highlights the importance of secure coding practices and proactive vulnerability management in modern software development.

2. OBJECTIVES OF THE PROJECT

The primary objective of this project is to perform a security assessment of a web application in order to identify common vulnerabilities and understand their potential impact. This assessment was carried out using automated security testing techniques and industry-standard tools.

The specific objectives of this project are:

- To understand the importance of web application security in modern software systems
- To identify security vulnerabilities using OWASP Zed Attack Proxy (ZAP)
- To perform automated vulnerability scanning on a test web application
- To analyze detected vulnerabilities based on risk severity
- To map identified vulnerabilities to the OWASP Top 10 security risks

- To document findings with evidence and recommend appropriate mitigation strategies

3. TOOLS & TECHNOLOGIES USED

The following tools and technologies were used to perform the web application security assessment:

- **OWASP Zed Attack Proxy (ZAP):**
  An open-source web application security testing tool used to identify vulnerabilities such as misconfigurations and information disclosure issues through automated scanning.
- **OWASP Juice Shop:**
  An intentionally vulnerable web application used as a testing environment to practice and understand real-world web security flaws.

- **Docker:**
  A containerization platform used to deploy and run the OWASP Juice Shop application locally in a secure and isolated environment.
- **Web Browser (Google Chrome / Microsoft Edge):**
  Used to access the target web application and verify application behaviour during testing.
- **Operating System:**
  Microsoft Windows was used as the host operating system for conducting the security assessment.

4. METHODOLOGY

The web application security assessment was carried out using an automated testing approach. Initially, OWASP Juice Shop was

deployed locally using Docker to create a controlled testing environment. The application was accessed through a web browser to ensure successful deployment.

After confirming the availability of the application, OWASP Zed Attack Proxy (ZAP) was launched and configured for automated scanning. The target URL of the application was provided to OWASP ZAP, and both traditional and AJAX spider options were enabled. An automated vulnerability scan was then initiated to identify potential security issues.

Once the scanning process was completed, the generated alerts were reviewed and analyzed. The identified vulnerabilities were categorized based on their risk level and mapped to the OWASP Top 10 security risks. Relevant screenshots were captured as evidence, and appropriate remediation measures were suggested for each identified vulnerability.

5. VULNERABILITY ASSESSMENT & FINDINGS

During the automated security scan performed using OWASP ZAP, multiple vulnerabilities were identified in the target web application. The following section describes the key vulnerabilities, their impact, and recommended remediation steps.

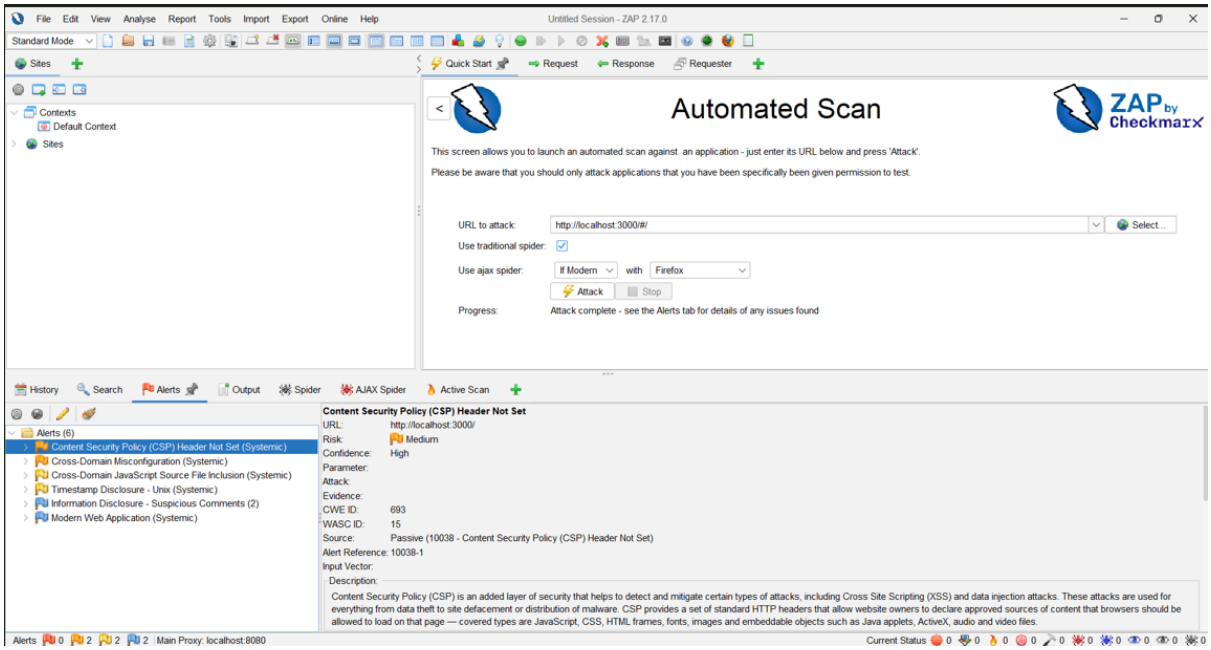
Vulnerability 1: Content Security Policy (CSP)

Risk Level: Medium
OWASP Top 10 Category: A05 – Security Misconfiguration

Description:
The application does not implement a Content Security Policy (CSP) header. CSP helps prevent various types of attacks, including Cross-Site Scripting (XSS), by restricting the sources from which content can be loaded.

Impact:

Without a CSP, attackers may inject malicious scripts into the application, potentially leading to data theft or unauthorized actions.



Remediation:

Implement a strong Content Security Policy header to allow content only from trusted sources.

Vulnerability

2: Cross-Domain Misconfiguration

Risk Level: Medium

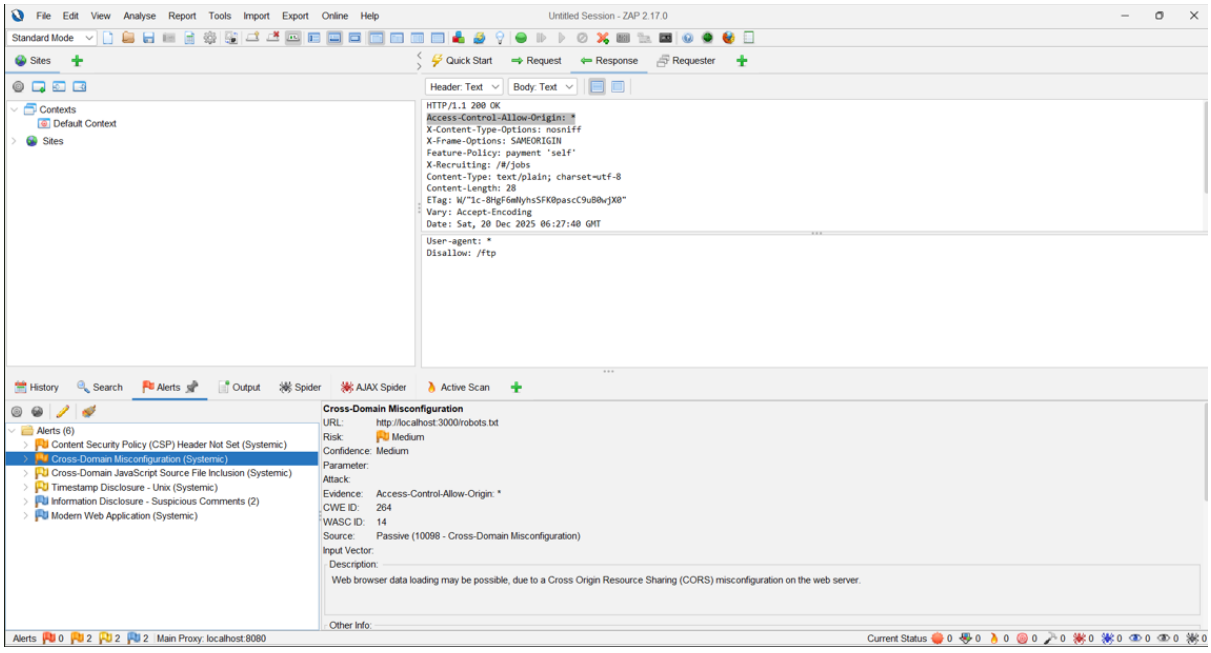
OWASP Top 10 Category: A05 – Security Misconfiguration

Description:

The application allows cross-domain requests without proper restrictions, which may expose sensitive resources to untrusted domains.

Impact:

An attacker could exploit this misconfiguration to access or manipulate application data from unauthorized domains.



Remediation:
Configure Cross-Origin Resource Sharing (CORS) policies to allow requests only from trusted domains.

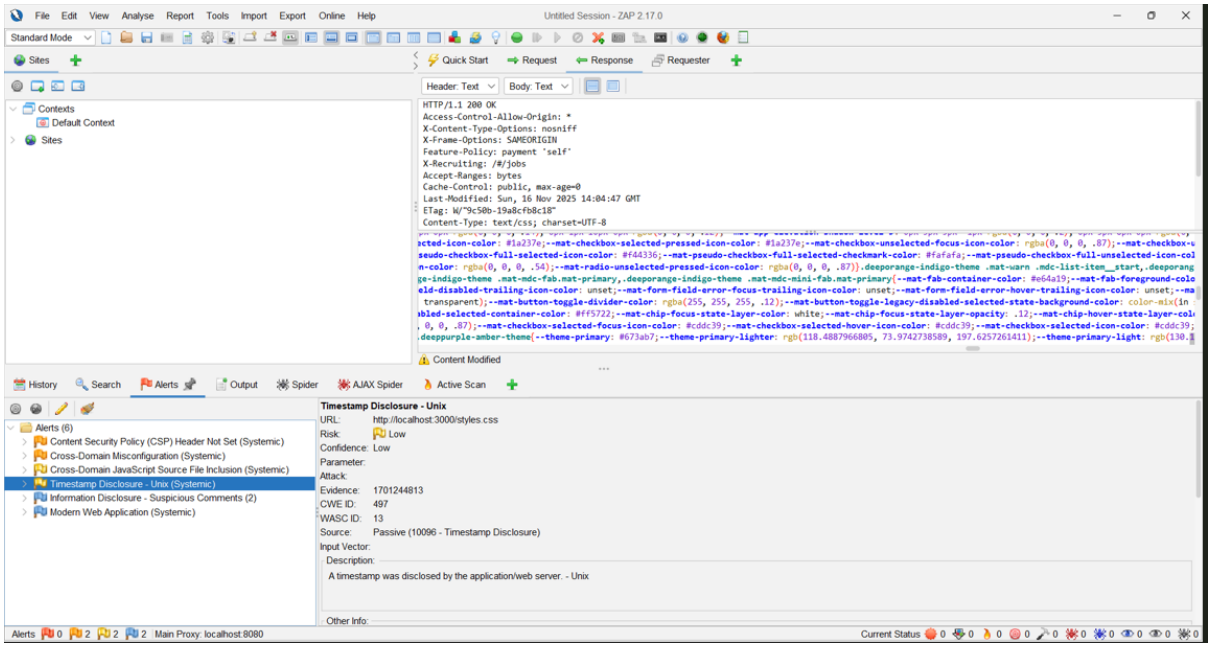
Vulnerability 3: Timestamp Disclosure – Unix

Risk Level: Low

OWASP Top 10 Category: A02 – Cryptographic Failures

Description:
The application exposes Unix timestamps in responses, which may reveal internal system information.

Impact:
This information could assist attackers in understanding system behavior or timing attacks.



Remediation:

Avoid exposing system timestamps in client-side responses unless necessary.

Vulnerability 4: Suspicious Comments Disclosure

Risk Level: Low

OWASP Top 10 Category: A01 – Broken Access Control

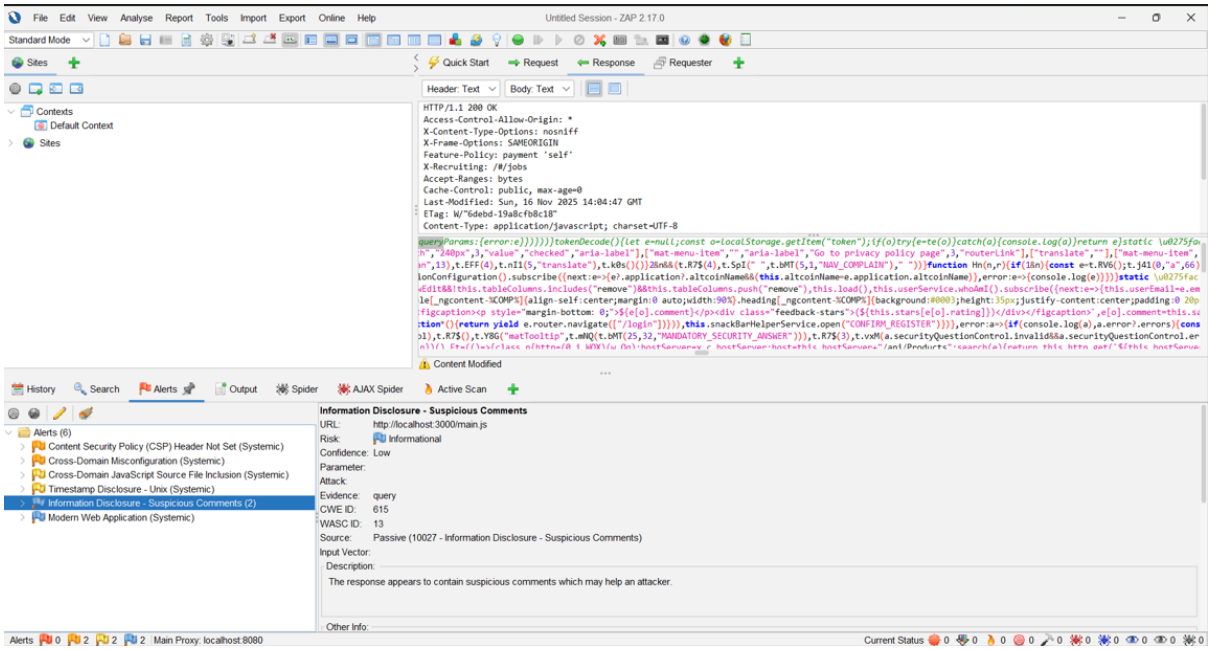
Description:

Suspicious comments were found in the application source code, which may disclose

internal logic or sensitive implementation details.

Impact:

Such information may help attackers understand the application structure and plan attacks.



Remediation:

Remove unnecessary comments and sensitive information from production code.

6. OWASP Top 10 Mapping

Identified Vulnerability	OWASP Top 10 Category
Content Security Policy Header Not Set	A05 – Security Misconfiguration
Cross-Domain Misconfiguration	A05 – Security Misconfiguration
Timestamp Disclosure	A02 – Cryptographic Failures
Suspicious Comments Disclosure	A01 – Broken Access Control

7. CONCLUSION

The web application security assessment conducted using OWASP Zed Attack Proxy (ZAP) successfully identified multiple vulnerabilities within the target application. The findings primarily included security misconfigurations and information disclosure issues, which are common risks in web applications if not properly addressed.

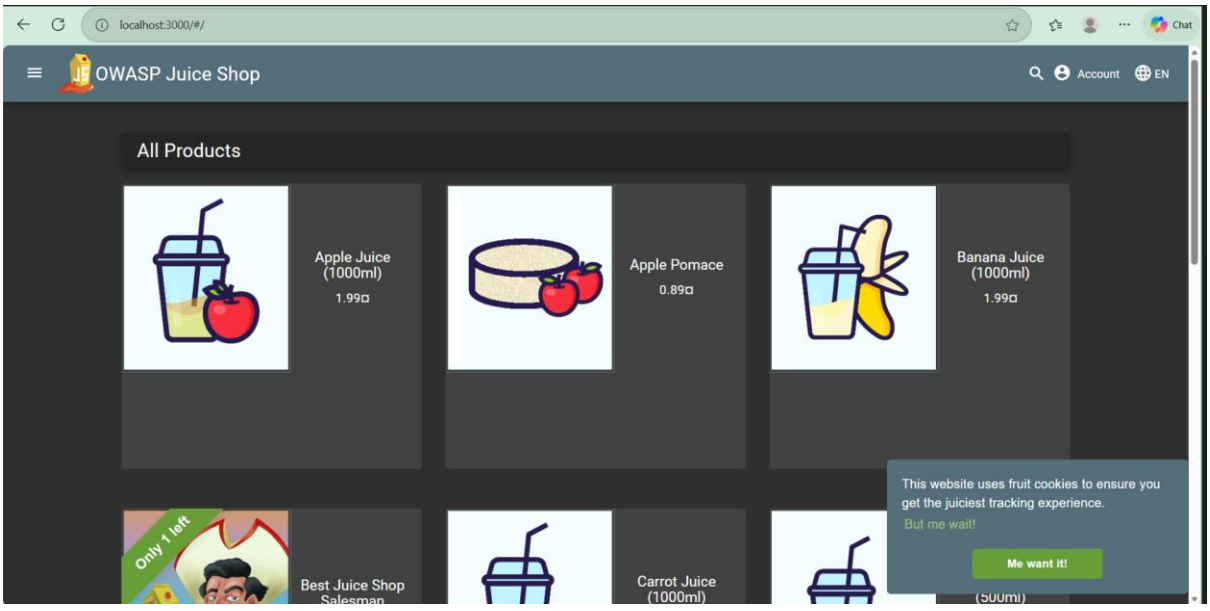
This project provided practical exposure to automated web application security testing and enhanced understanding of identifying, analyzing, and documenting security vulnerabilities based on the OWASP Top 10 framework. The assessment highlights the importance of secure configuration, regular vulnerability scanning, and proactive security practices in strengthening the overall security posture of web applications.

By implementing the recommended remediation measures, organizations can significantly reduce the risk of potential cyber attacks and ensure a more secure web environment.

8. APPENDIX ADDITIONAL SCREENSHOTS

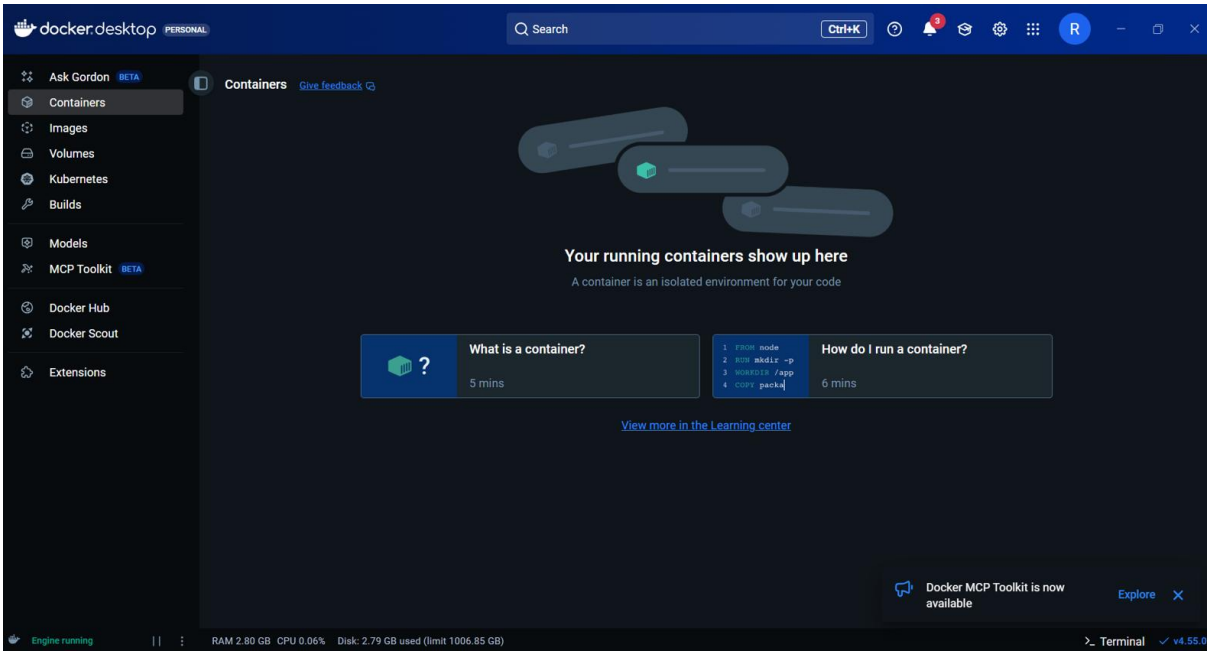
Screenshot 1: OWASP Juice Shop Home Page

This screenshot shows the successful deployment of the OWASP Juice Shop web application running locally in the browser using the URL http://localhost:3000.



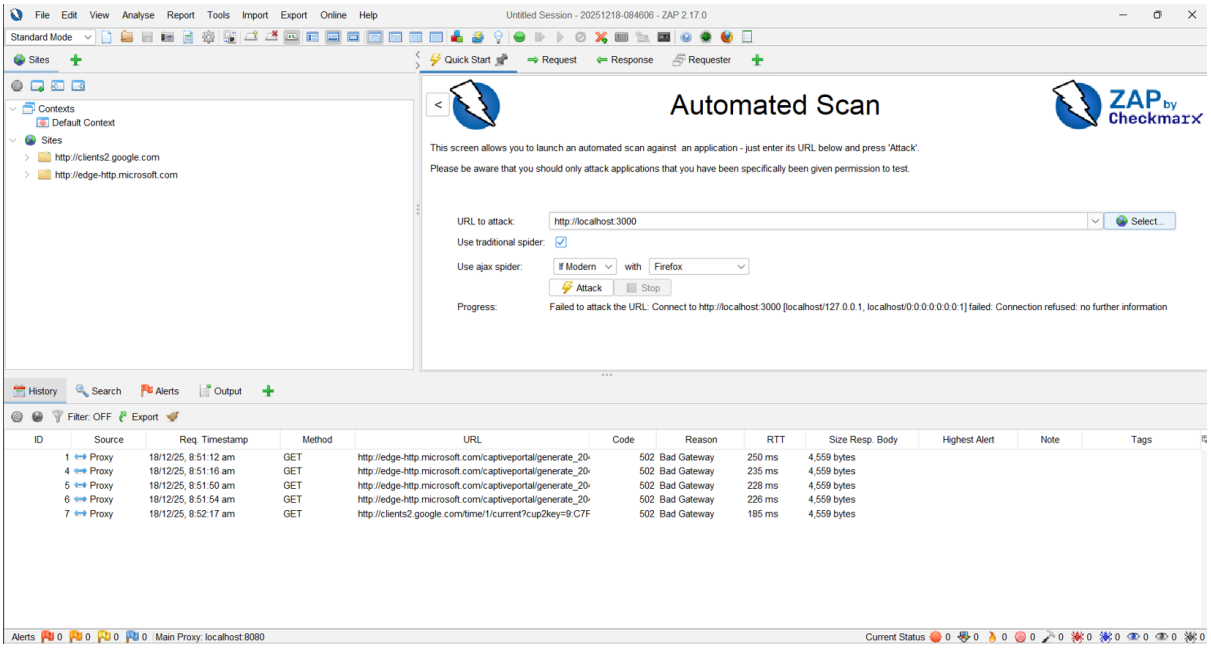
Screenshot 2: Docker Container Execution

This screenshot displays the Docker environment where the OWASP Juice Shop container is running successfully, confirming that the application was deployed using Docker.



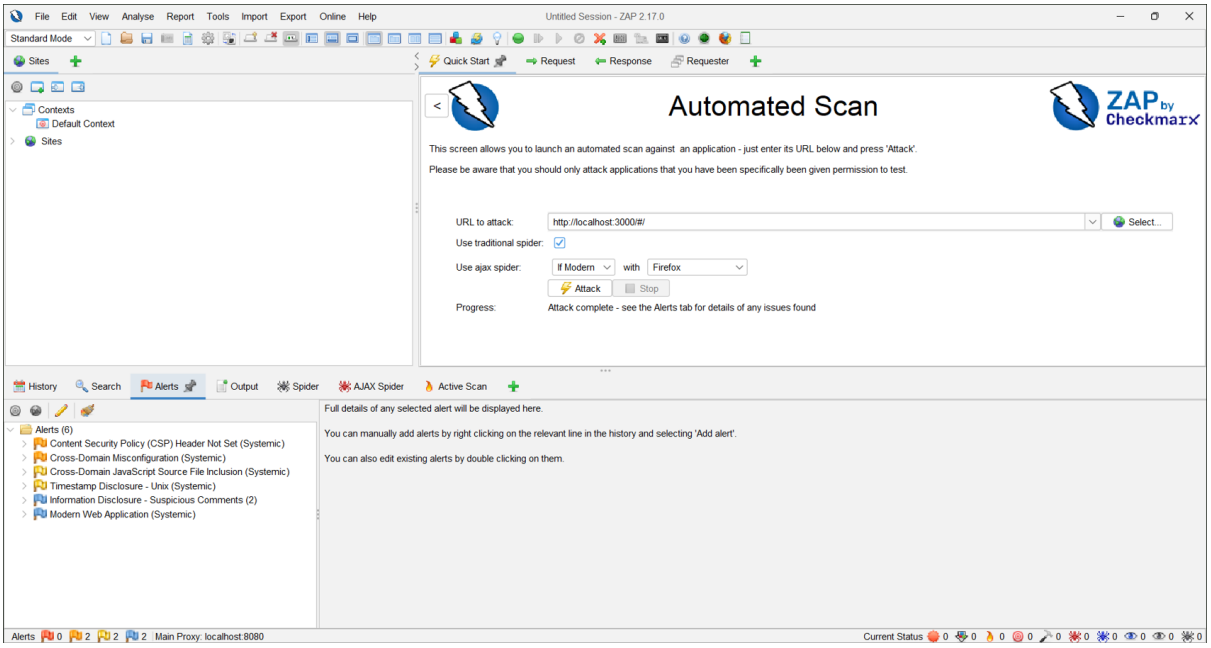
Screenshot 3: OWASP ZAP Dashboard

This screenshot shows the OWASP ZAP tool interface after loading the target application URL for automated security testing.



Screenshot 4: Automated Scan Results

This screenshot presents the vulnerability scan results generated by OWASP ZAP, highlighting detected security issues along with their risk levels.



9. References

- OWASP Foundation. OWASP Juice Shop.
- OWASP Zed Attack Proxy (ZAP) Documentation.
- OWASP Top 10 Web Application Security Risks.