



# **Security Assessment Report**

Web Application Penetration Testing – Week 1

**DeveloperHub.co**

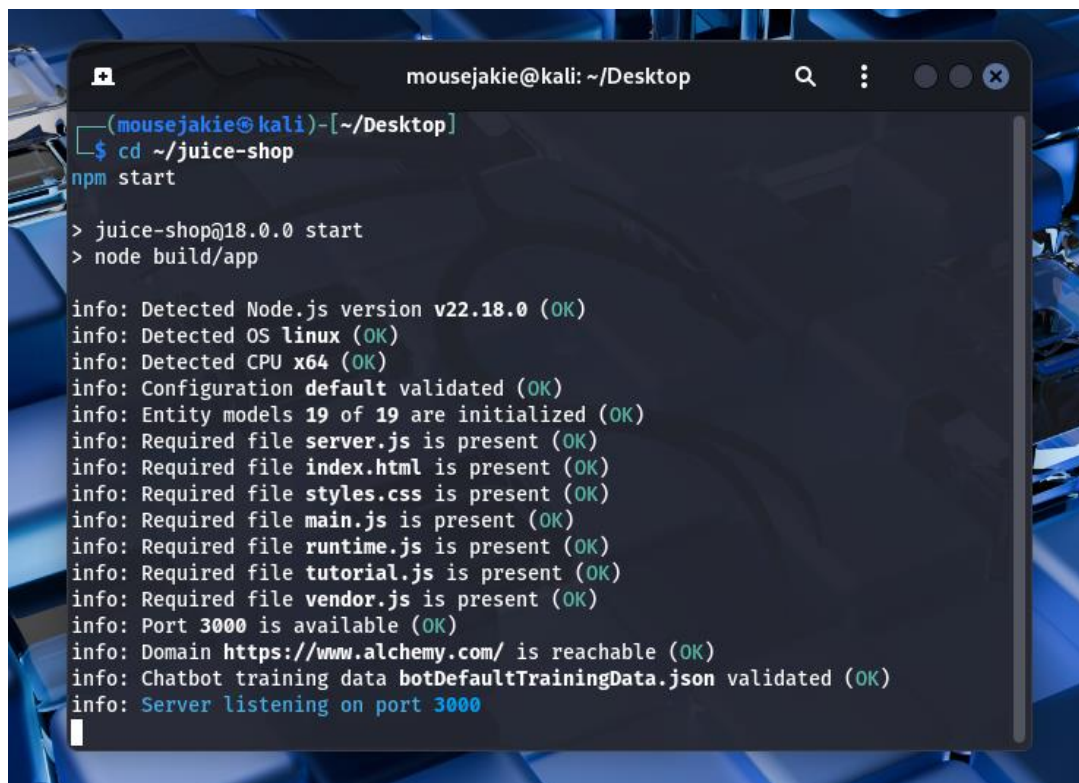
Rimsha Bibi

**DHC-504**

22-Aug-2025

## ➤ Introduction:

This report provides the findings of a web application security assessment conducted using **OWASP ZAP** against a test application running at **http://localhost:3000**. The primary objective was to identify potential vulnerabilities that could be exploited by attackers.

A terminal window titled 'mousejakie@kali: ~/Desktop' showing the execution of 'npm start' for the 'juice-shop' project. The output displays various system information checks (Node.js version, OS, CPU, configuration) and file presence validations, all marked as 'OK'. It concludes with 'Server listening on port 3000'.

```
(mousejakie@kali)-[~/Desktop]
$ cd ~/juice-shop
npm start

> juice-shop@18.0.0 start
> node build/app

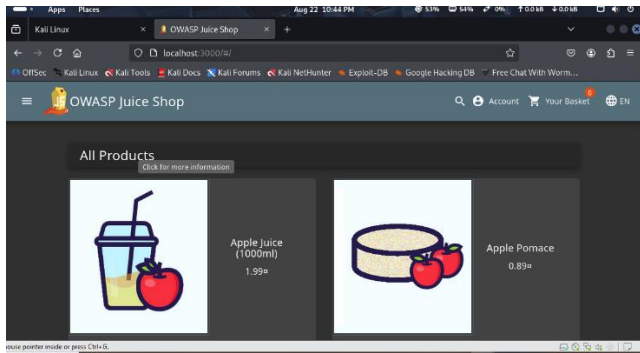
info: Detected Node.js version v22.18.0 (OK)
info: Detected OS linux (OK)
info: Detected CPU x64 (OK)
info: Configuration default validated (OK)
info: Entity models 19 of 19 are initialized (OK)
info: Required file server.js is present (OK)
info: Required file index.html is present (OK)
info: Required file styles.css is present (OK)
info: Required file main.js is present (OK)
info: Required file runtime.js is present (OK)
info: Required file tutorial.js is present (OK)
info: Required file vendor.js is present (OK)
info: Port 3000 is available (OK)
info: Domain https://www.alchemy.com/ is reachable (OK)
info: Chatbot training data botDefaultTrainingData.json validated (OK)
info: Server listening on port 3000
```

## ➤ Tool Used:

The following tool was used for this assessment: **OWASP ZAP** (Zed Attack Proxy): An open-source tool for finding security vulnerabilities in web application.

## ➤ Methodology:

The assessment followed these steps:



1. **Application Setup:** The test web application was deployed and accessed at **http://localhost:3000**.
2. **Authentication:** A test account was created using email 'hahuu@hi.ji' and password '123456'.
3. **Automated Scan:** OWASP ZAP's Automated Scan feature was used to crawl and test the application.
4. **Reporting:** The scan results were analyzed to identify vulnerabilities.

### ➤ Findings Overview:

The scan identified several findings with the following severity distribution:

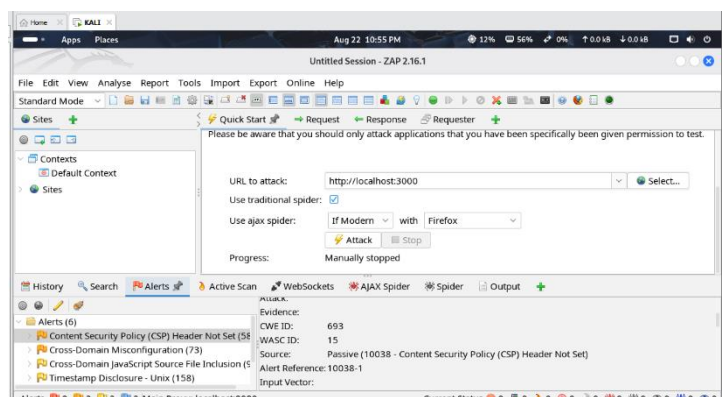
- High Severity: 0
- Medium Severity (Orange): 2
- Low Severity (Yellow): 2
- Informational (Blue): 2

### ➤ Example of Vulnerability:

One of the identified issues was the missing Content Security Policy (CSP).

**Description:** CSP is a security feature that helps prevent attacks like Cross-Site Scripting (XSS) and data injection. The lack of a CSP header increases the risk of such attacks.

**Solution:** Configure the web server or application server to set the Content-Security-Policy header with appropriate directives to mitigate these risks.



### ➤ Recommendation:

- Review all identified vulnerabilities (Medium, Low, and Informational. Implement missing security headers such as Content-Security-Policy. Apply secure coding practices to reduce exposure to XSS and injection attacks.
- Regularly test the application using automated and manual security testing methods. Ensure that authentication and authorization mechanisms are robust and resistant to attacks.

➤ **Conclusion:**

The assessment identified multiple areas where the application could be improved in terms of security. Although no high-severity issues were found, addressing the medium and low-level issues will significantly strengthen the application's security posture.