# Security Assessment Report -- OWASP Juice Shop (Local Lab)

**NAME: Oderinde Toluwanimi**

**TASK 1: Web Application Vulnerability Assessment -- OWASP Juice Shop (Docker)**

**PROGRAM: Future Interns Cybersecurity Internship**

**Date: September 2025**

**Target / Scope:** OWASP Juice Shop running locally at http://localhost:3000 (Docker image: bkimminich/juice-shop:latest unless otherwise noted). Testing performed using Burp Suite (bundled Chromium), SecLists, and browser DevTools.

---

## Executive summary

During a focused security assessment of a lab-hosted OWASP Juice Shop instance, multiple high- and medium-severity vulnerabilities were identified that allow account takeover, unauthorized data access, and sensitive data disclosure. Key risks include SQL injection-based authentication bypasses, exposed confidential files via anonymous FTP and backup downloads, insecure object references (IDOR) enabling access to other users' baskets, weak account recovery and leaked/guessable credentials, and a reflected XSS vulnerability in the search input. These issues combined allow an attacker to escalate privileges, exfiltrate sensitive documents, and manipulate user data. Immediate remediation of the high-severity findings is recommended.

---

## Scope & methodology

- **Scope:** Local Juice Shop instance (`http://localhost:3000`) and any resources discovered from within the application (FTP links, file directories). Testing was limited to intentionally vulnerable lab resources.
- **Methodology:** Manual testing supported by Burp Suite (intercept, repeater, intruder for brute-force), SecLists for password brute-force, and static inspection of client-side assets (DevTools). All actions were non-destructive and performed in a controlled lab environment.
- **Tools used:** Docker, Burp Suite (bundled Chromium), SecLists, browser DevTools, FTP client, terminal utilities (`sha256sum`).

---

## Top 5 prioritized findings

These are the highest-priority items to address first (concise):

1. **F-01 / F-02 -- SQL injection allowing authentication bypass (admin + specific users)** -- *High*: enables immediate account takeover and admin access. Fix: parameterized queries and input validation.
2. **F-04 -- Confidential document exposure via anonymous FTP (acquisition.md)** -- *High*: sensitive file accessible without auth. Fix: remove public access, enforce auth, rotate secrets.
3. **F-06 -- Backup file disclosure via null byte poisoning (`package.json.bak`)** -- *High*: reveals internal config/backups. Fix: remove backups from webroot, sanitize file-serving logic.

4. **F-07 / F-08 -- Hidden admin route + IDOR (access other users' baskets)** -- *High*: exposed endpoints + broken access control. Fix: remove client-side secrets and enforce server-side RBAC/authorization.

5. **F-09 -- Reflected XSS in search input** -- *Medium→High*: allows JS execution in victims' browsers. Fix: output encoding + CSP.

---

# Detailed findings

(Each finding includes: title, severity, affected endpoint(s), proof summary, impact, root cause, remediation. Evidence placeholders included; you will embed images locally.)

### F-01 -- Authentication bypass via SQL injection (admin)

- **Severity:** High
- **Affected endpoint(s):** `/#/login` (POST)
- **Proof summary:** Authentication bypass achieved using payloads such as `' OR '1'='1' --`, enabling login as `admin@juice-sh.op` without valid password. Evidence: Future1-JuiceShopLoginInjectionpng, Future1-JuiceShopLoggedIn, Future1-JuiceShopLoggedInBenderUser.
- **Impact:** Full admin account takeover -- ability to view/modify application data and perform privileged actions.
- **Root cause:** Unparameterized SQL queries; user-supplied input concatenated into SQL statements.
- **Remediation:** Use parameterized queries/prepared statements or ORM; server-side input validation; least-privilege DB account; add auth rate-limiting and monitoring.

---

### F-02 -- Authentication bypass via SQL injection (targeted user: benda)

- **Severity:** High
- **Affected endpoint(s):** `/#/login` (POST)
- **Proof summary:** Similar injection technique allowed impersonation of `benda@juice-sh.op`. Evidence: Future1-JuiceShopLoginInjectionpng,

Future1-JuiceShopLoggedIn, Future1-JuiceShopLoggedInBenderUser.

- **Impact & Remediation:** Same as F-01.

---

## F-03 -- Account takeover via weak account-recovery (jim)

- **Severity:** Medium → High
- **Affected endpoint(s):** Password recovery flow (`/forgot-password`)
- **Proof summary:** Recovery question answered using publicly discoverable info (comments showing Star Trek fandom). Evidence: Future1-JuiceShopResetJimPass1, Future1-JuiceShopResetJimPass2, Future1-JuiceShopResetJimPass3, Future1-JuiceShopResetJimPass4.
- **Impact:** Account takeover for users with weak knowledge-based recovery answers.
- **Remediation:** Replace knowledge-based questions with secure, time-limited email reset links or OTPs; enforce rate-limiting; support MFA.

---

## F-04 -- Confidential document exposure via publicly-accessible FTP (acquisition.md)

- **Severity:** High
- **Affected resource:** FTP link discovered in About page → `acquisition.md`
- **Proof summary:** Followed FTP link from About page and downloaded `acquisition.md`. Evidence: Future1-JuiceShopConfidentialDocx3, Future1-JuiceShopConfidentialDocx2, Future1-JuiceShopConfidentialDocx.
- **Impact:** Sensitive corporate documents exfiltratable without auth.
- **Remediation:** Immediately remove public access; require authenticated SFTP/FTPS; move files to access-controlled storage; rotate secrets found in files; enable logging/alerting for downloads.

---

### F-05 -- Account takeover via publicly-disclosed weak password (McSafeSearch)

- **Severity:** Medium
- **Affected endpoint(s):** Login
- **Proof summary:** User disclosed password in public post/lyrics; successful login with `MrN00dle`. Evidence: Future1-JuiceShopMcSafe.
- **Impact:** Account compromise due to credential disclosure.
- **Remediation:** Enforce password policy, MFA, credential leak detection, user education.

---

### F-06 -- Backup file disclosure via null byte poisoning

- **Severity:** High
- **Affected resource:** `.../package.json.bak` (downloadable via `package.json.bak%2500.md`)
- **Proof summary:** File extension filter bypassed using encoded null byte trick allowing download of `package.json.bak`. Evidence: Future1-JuiceShopBackUpFile3, Future1-JuiceShopBackUpFile2, Future1-JuiceShopBackUpFile.
- **Impact:** Disclosure of config/backups which may include secrets.
- **Remediation:** Remove backups from webroot, enforce server-side file validation, allowlist download types, disable path traversal/null byte tricks, set correct file permissions.

---

### F-07 -- Hidden administration endpoint disclosed in client-side JS (main.js)

- **Severity:** High
- **Affected files/endpoint(s):** `main.js` (client bundle) contains admin route
- **Proof summary:** Found admin path in `main.js`; direct access after authenticating enabled admin UI. Evidence: Future1-JuiceShopAdminBruteForce, Future1-JuiceShopAdminBruteForce2, Future1-JuiceShopAdminBruteForce3, Future1-JuiceShopAdminBruteForce4, Future1-JuiceShopAdminPage2,

Future1-JuiceShopAdminPage.

- **Impact:** Knowledge of sensitive endpoints simplifies targeted attacks; combined with compromised credentials, enables full admin control.

- **Remediation:** Strip internal routes and secrets from production bundles; ensure server-side authorization for admin routes; use environment configuration that excludes debug/internal info.

---

## F-08 -- Insecure Direct Object Reference (IDOR) -- admin can access other users' baskets

- **Severity:** High

- **Affected endpoint(s):** `GET /rest/basket/{id}`

- **Proof summary:** Intercepted GET `/rest/basket/1` and changed to `/rest/basket/2` to retrieve another user's basket. Evidence: Future1-JuiceShopShoppingBasket2, Future1-JuiceShopShoppingBasket.

- **Impact:** Unauthorized access to other users' data; potential for data tampering if write operations are possible.

- **Remediation:** Implement server-side per-resource authorization checks; use indirect references (UUIDs) where appropriate; log and monitor admin accesses.

---

## F-09 -- Reflected Cross-Site Scripting (XSS) via search input

- **Severity:** Medium → High

- **Affected endpoint(s):** Search rendering (e.g., `/#/search?term=`)

- **Proof summary:** Reflected payload `<iframe src="javascript:alert('xss')">` executed in browser, showing an alert. Evidence: Future1-JuiceShopSearchXXS.

- **Impact:** Execution of arbitrary JS; risk of session token theft, CSRF escalation, and user-targeted phishing.

- **Remediation:** Apply proper output encoding, input validation, CSP, and set `HttpOnly`/`Secure` cookie flags.

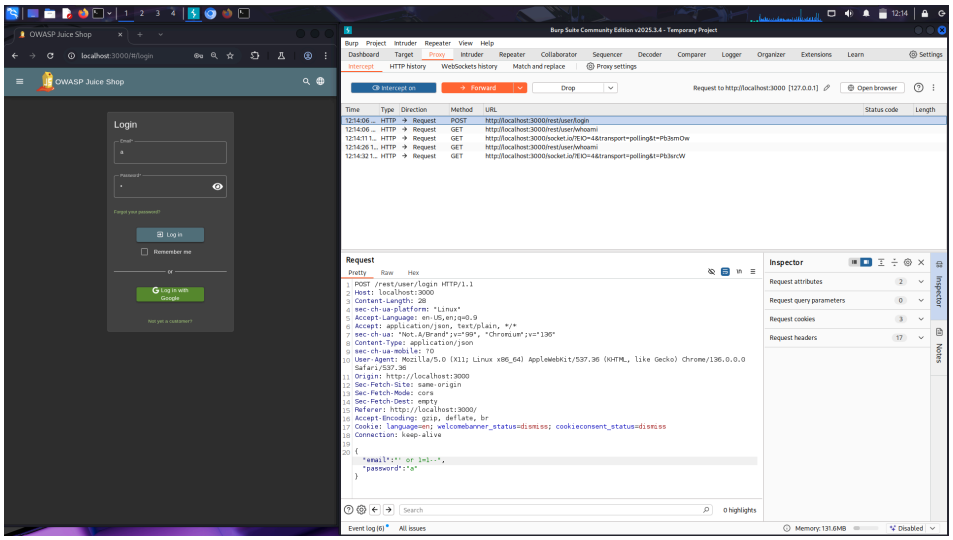
## OWASP Top-10 Compliance checklist (mapped)

| Finding ID | Vulnerability | OWASP Top-10 (2021) | Priority |
|---|---|---|---|
| F-01 | SQLi auth bypass (admin) | A03 -- Injection | High |
| F-02 | SQLi auth bypass (benda) | A03 -- Injection | High |
| F-03 | Weak account recovery (jim) | A07 -- Identification & Authentication Failures | Medium → High |
| F-04 | Confidential doc via FTP | A05 -- Security Misconfiguration / A01 -- Broken Access Control | High |
| F-05 | Publicly disclosed password (McSafeSearch) | A07 -- Identification & Authentication Failures | Medium |
| F-06 | Backup file via null byte poisoning | A05 -- Security Misconfiguration / A01 -- Broken Access Control | High |
| F-07 | Admin path in JS / exposed endpoint | A05 -- Security Misconfiguration / A01 -- Broken Access Control | High |
| F-08 | IDOR -- basket access by ID | A01 -- Broken Access Control | High |

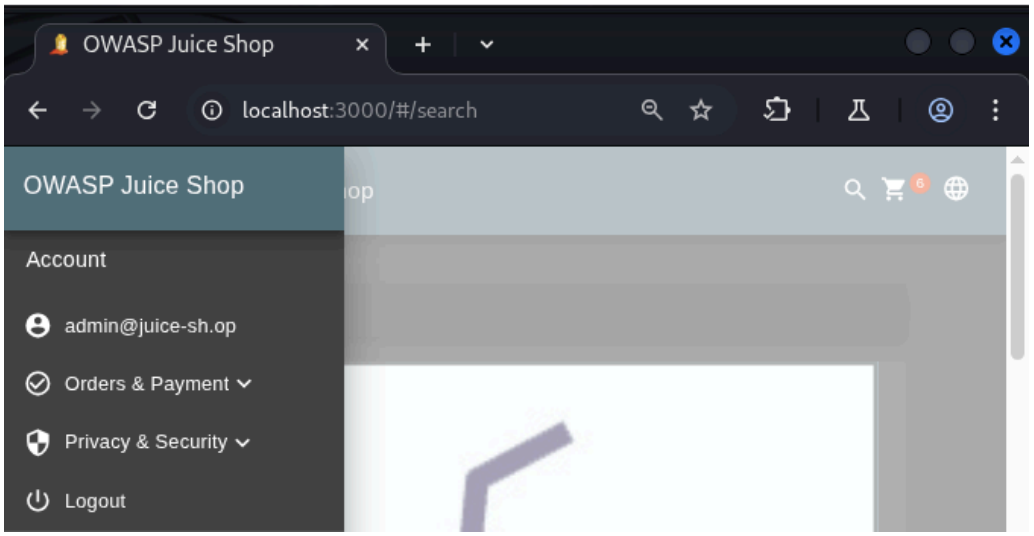
| | | | |
|---|---|---|---|
| F-09 | Reflected XSS | A03 -- Injection / A04 -- Insecure Design | Medium → High |

# Evidence appendix

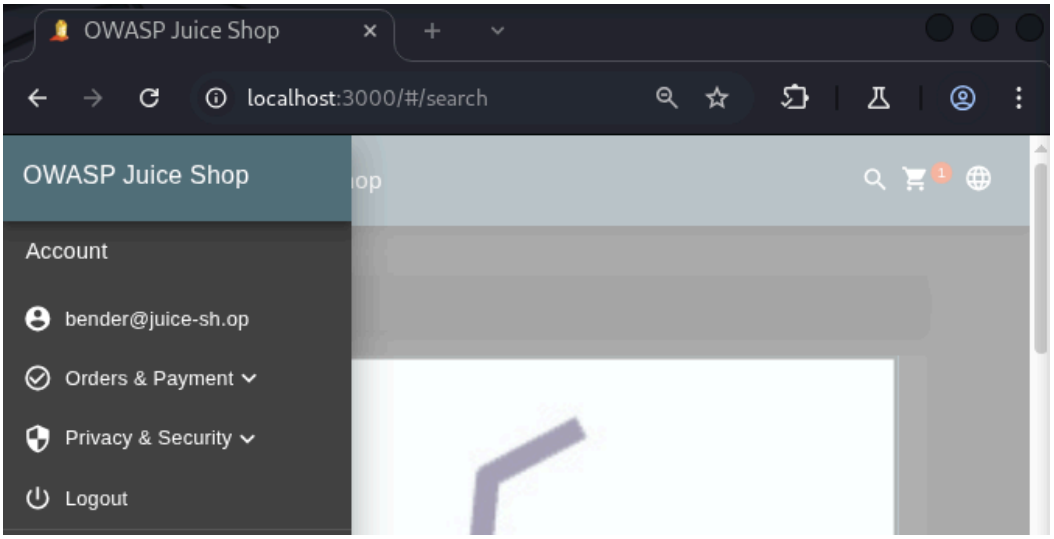
- **F-01 / F-02 -- SQLi Authentication Bypass**
  - Future1-JuiceShopLoginInjection



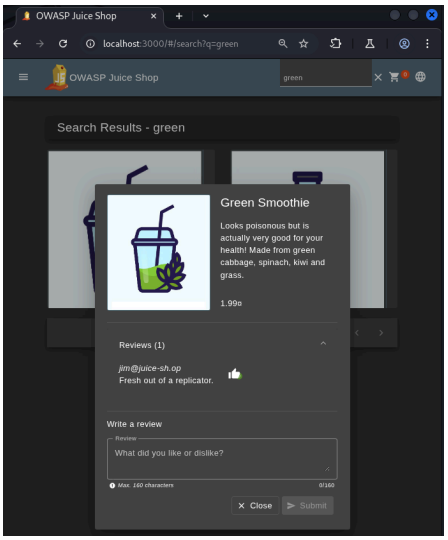

  - Future1-JuiceShopLoggedIn



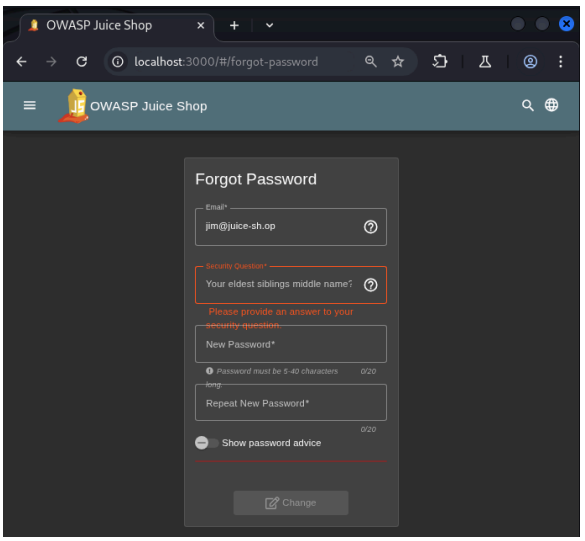

  - Future1-JuiceShopLoggedInBenderUser




- **F-03 -- Weak Account Recovery (Jim)**
    - Future1-JuiceShopResetJimPass




    - Future1-JuiceShopResetJimPass2



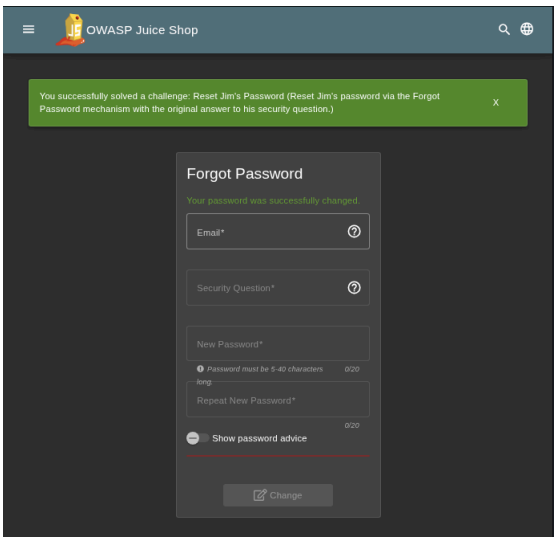

- Future1-JuiceShopResetJimPass3

## Biography [ edit ]

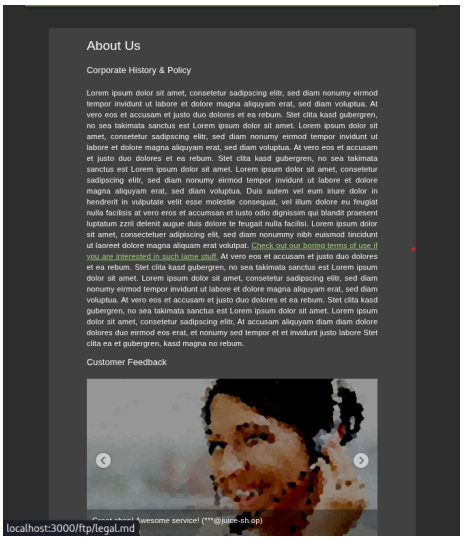
James Tiberius Kirk was born in Riverside, Iowa, on March 22, 2233,[2] where he was raised by his parents, George and Winona Kirk.[3] Although born on Earth, Kirk lived for a time on Tarsus IV, where he was one of nine surviving witnesses to the massacre of 4,000 colonists by Kodos the Executioner.

| | |
|---|---|
| | Starfleet |
| **Family** | George Kirk (father)<br>Winona Kirk (mother)<br>George Samuel Kirk (brother)<br>Tiberius Kirk (grandfather)<br>James (maternal grandfather)<br>Aurelan Kirk (sister-in-law)<br>Peter Kirk (nephew)<br>2 other nephews |

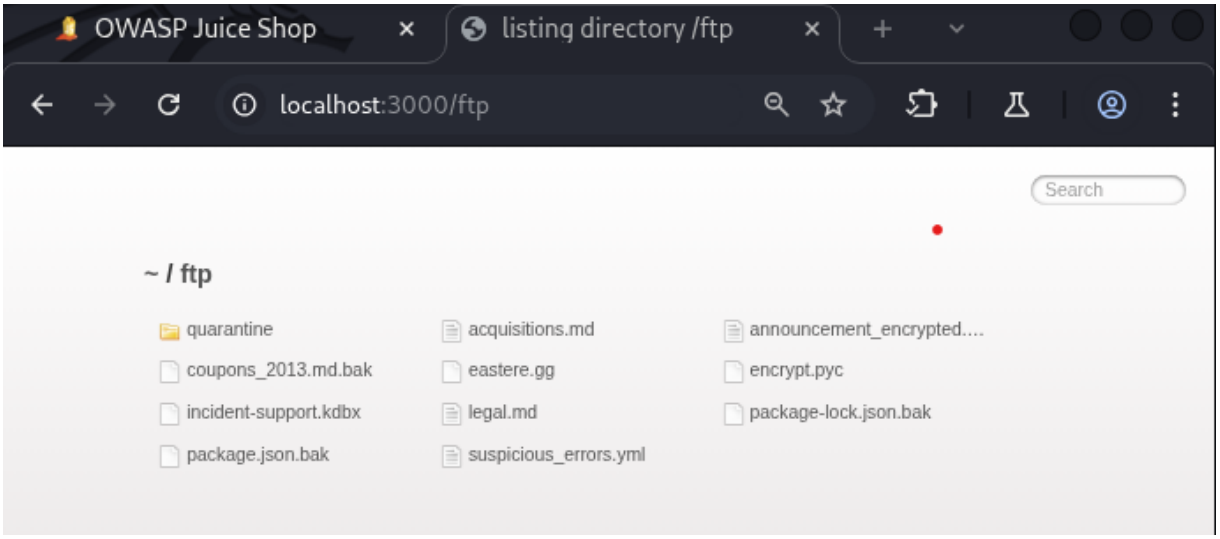
- Future1-JuiceShopResetJimPass4



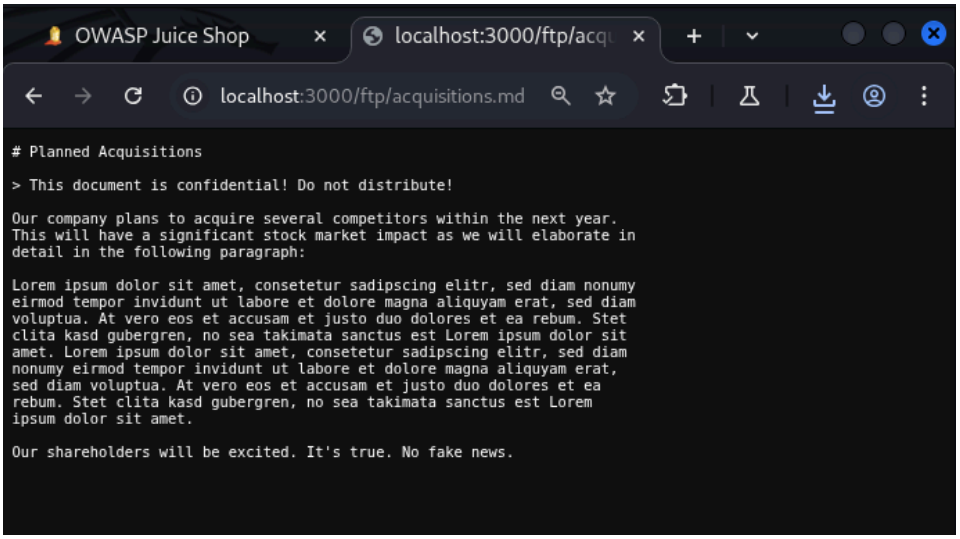

- **F-04 -- Confidential document via FTP**
    - Future1-JuiceShopConfidentialDocx



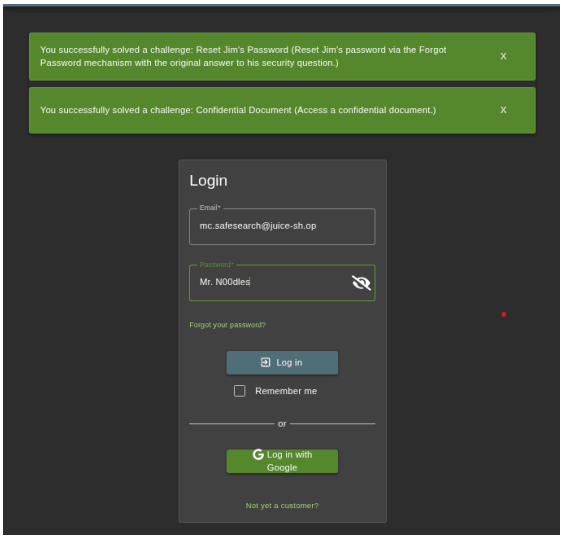

    - Future1-JuiceShopConfidentialDocx2



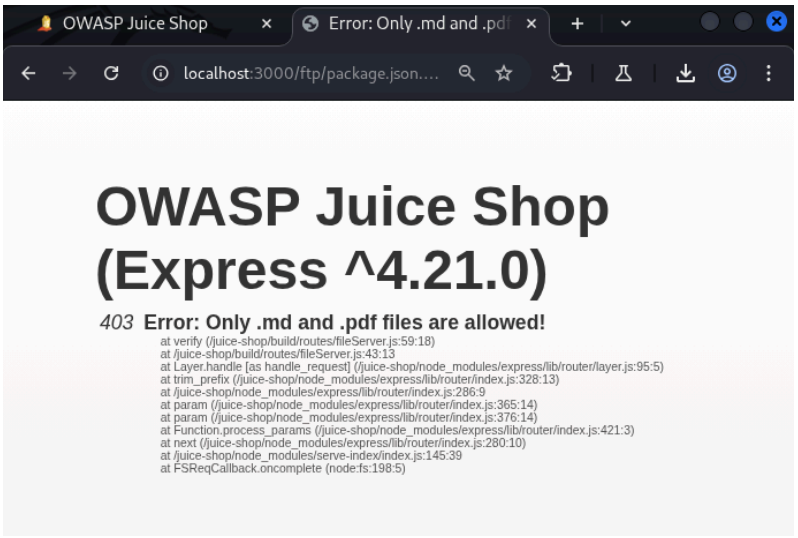

- Future1-JuiceShopConfidentialDocx3



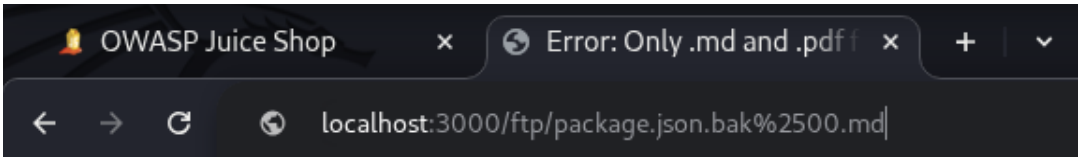

- **F-05 -- Weak Password Disclosure (McSafeSearch)**
  - Future1-JuiceShopMcSafe



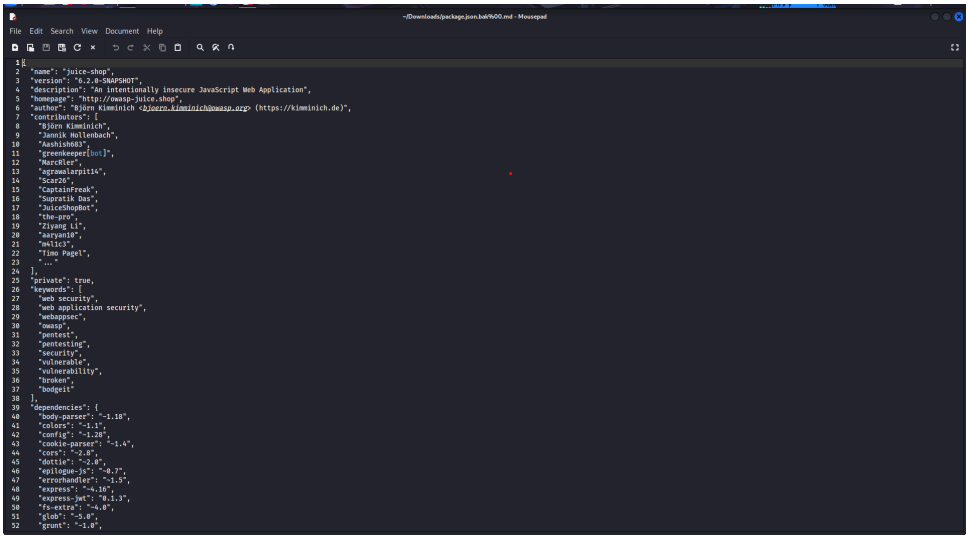

- **F-06 -- Backup file disclosure (Null byte)**
    - Future1-JuiceShopBackUpFile



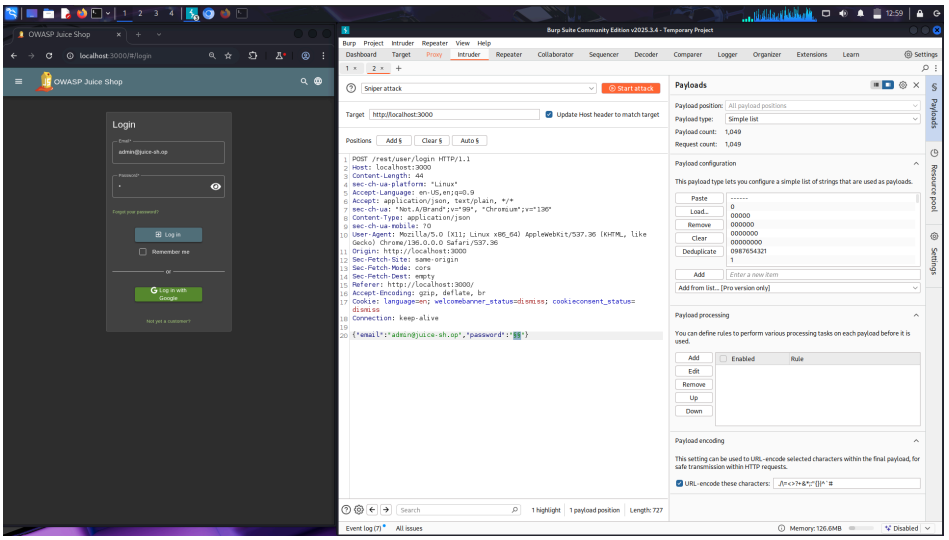

    - Future1-JuiceShopBackUpFile2



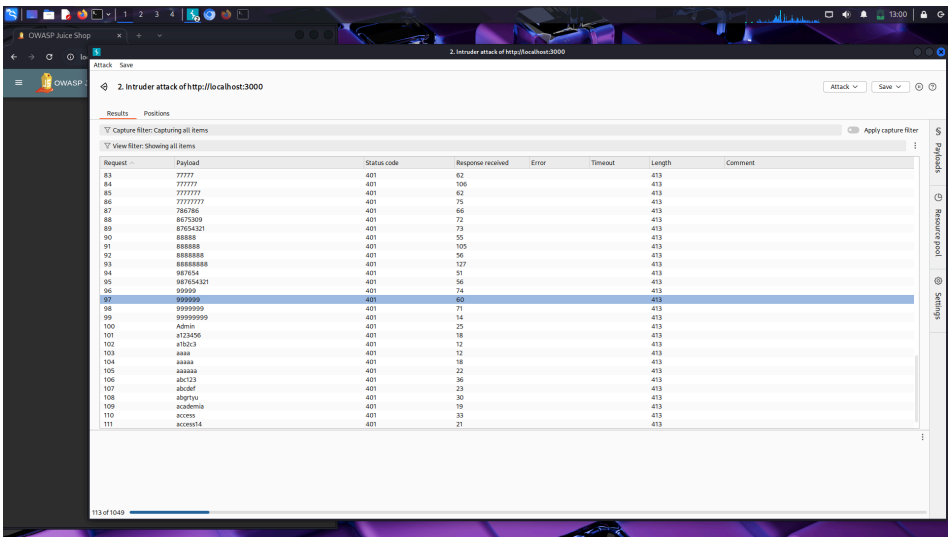

    - Future1-JuiceShopBackUpFile3



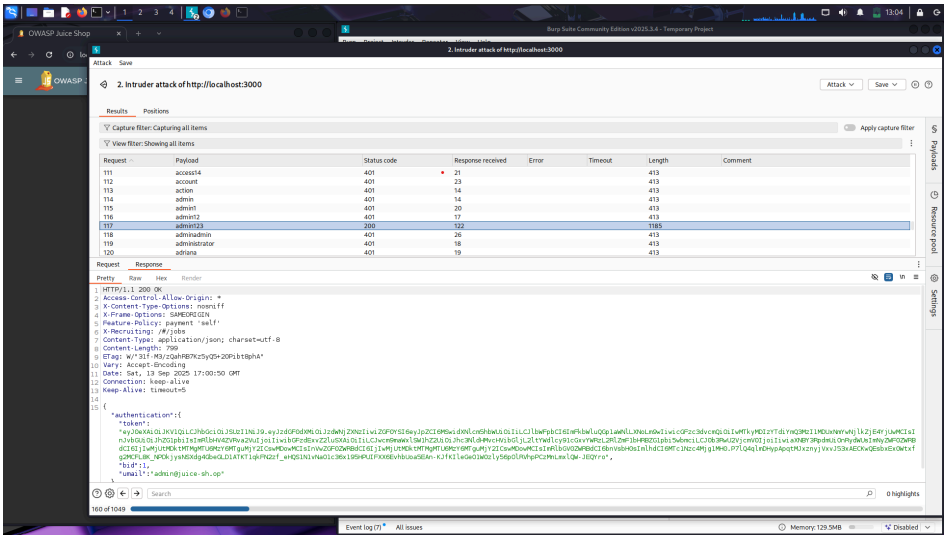

- **F-07 -- Hidden Admin Route**
  - Future1-JuiceShopAdminBruteForce



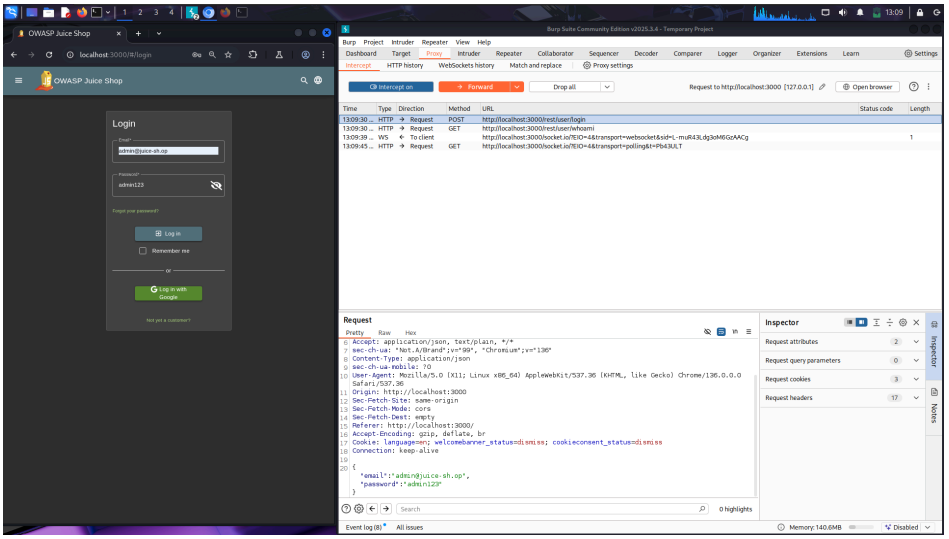

  - Future1-JuiceShopAdminBruteForce2



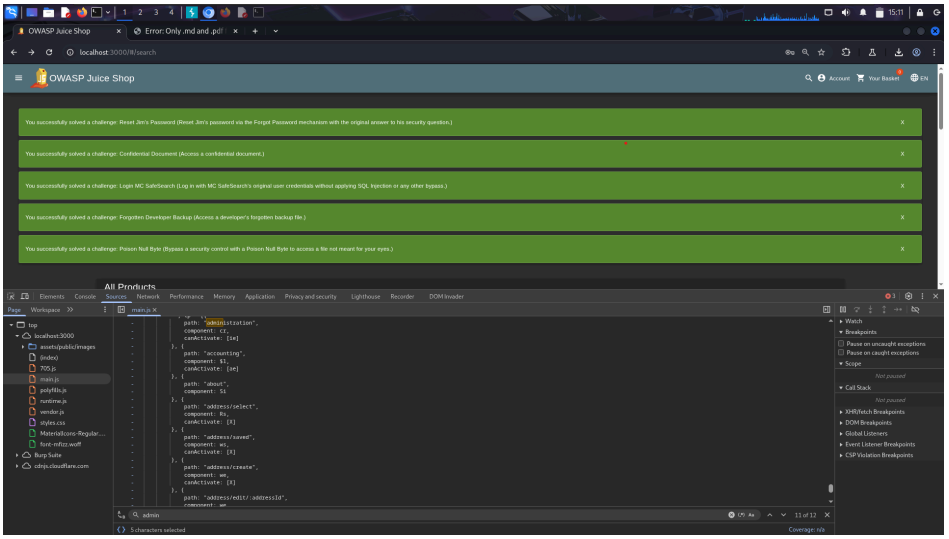

- Future1-JuiceShopAdminBruteForce3



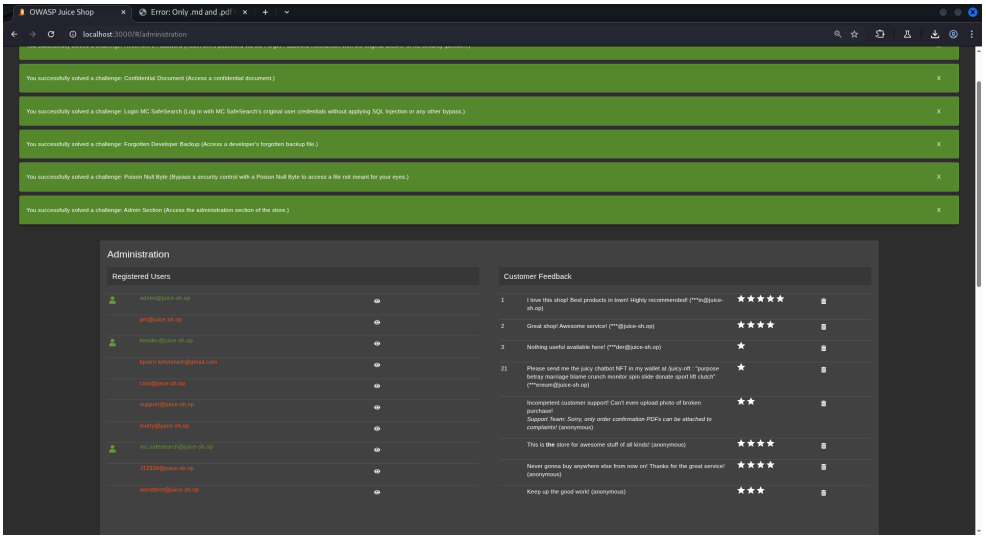

- Future1-JuiceShopAdminBruteForce4



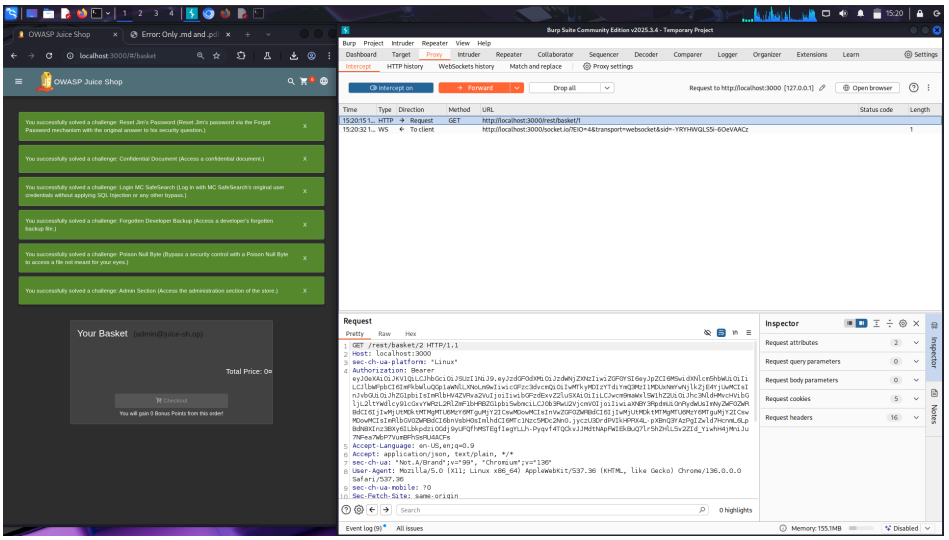
- Future1-JuiceShopAdminPage



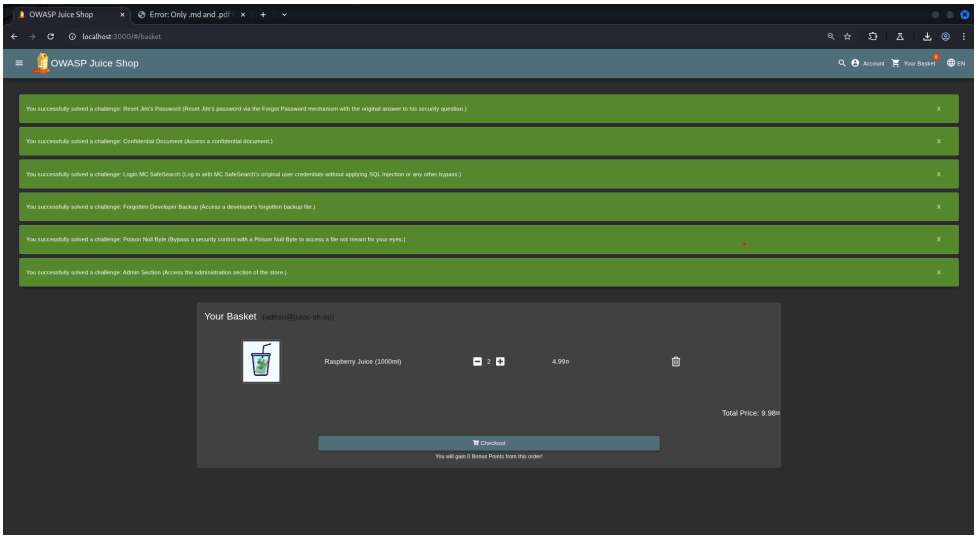
- Future1-JuiceShopAdminPage2



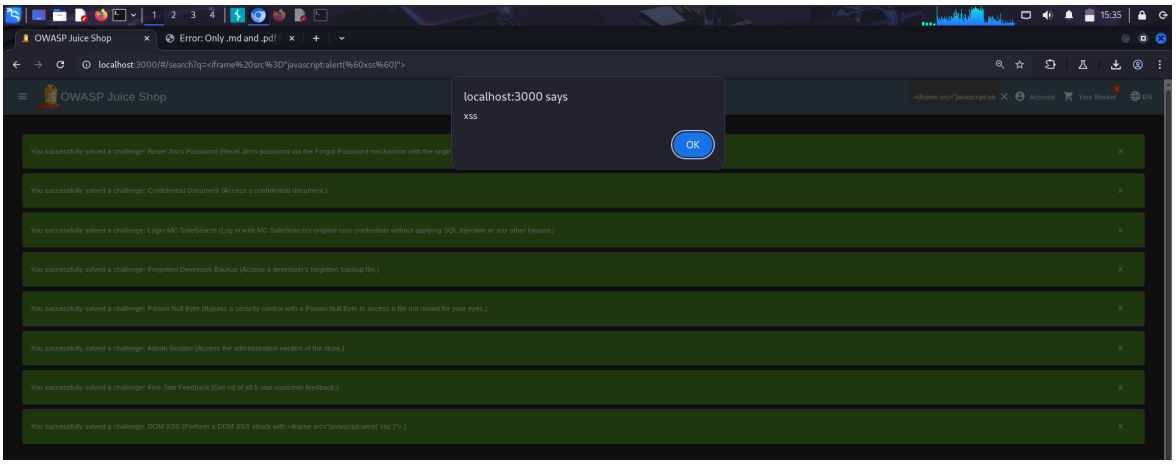

- **F-08 -- IDOR on Basket**
  - Future1-JuiceShopShoppingBasket




  - Future1-JuiceShopShoppingBasket2




- **F-09 -- Reflected XSS**
    - Future1-JuiceShopSearchXXS




---

# Risk matrix and remediation roadmap

**Immediate (0–7 days):**

- Fix SQL injection on login and all DB access points (parameterized queries).
- Remove public FTP/backups and restrict access to sensitive files; rotate secrets.
- Enforce strong admin account protections (MFA, session regeneration, secure cookies).

**Short-term (1–4 weeks):**

- Enforce server-side authorization and RBAC for all admin/user resources.
- Implement password policies, rate-limiting, and account lockouts; disable knowledge-based recovery.
- Harden file-serving logic (allowlist), fix null byte/path traversal issues.

**Medium-term (1–3 months):**

- Add automated security tests in CI (SQLi/XSS fuzzing, IDOR checks).
- Implement monitoring/alerting for unusual downloads or admin access.
- Conduct a retest after fixes and produce an updated report.

---

# Notes & legal

All testing described was executed against a local, intentionally vulnerable application (OWASP Juice Shop) running in a controlled lab. Do not run these tests against third-party systems without explicit written permission. Include this legal disclaimer in the final client report cover page.