VAPT on OWASP Juice Shop using Kali Linux

This project demonstrates a complete **Vulnerability Assessment and Penetration Testing (VAPT)** workflow using **Kali Linux** tools on the **OWASP Juice Shop**, a purposely vulnerable web application designed for security training. The goal is to simulate a real-world ethical hacking engagement, uncover security flaws, and document findings in a professional format.

🔧 What You'll Be Doing

You'll walk through each phase of a typical VAPT process:

1. **Lab Setup**: Deploy Juice Shop locally using Docker on Kali Linux.
2. **Reconnaissance**: Use tools like Nmap and WhatWeb to gather information about the target.
3. **Vulnerability Scanning**: Run Nikto and Burp Suite to identify weaknesses.
4. **Exploitation**: Perform manual attacks like SQL injection and explore automated options with Metasploit.
5. **Post-Exploitation**: Access sensitive data and understand the impact of successful attacks.
6. **Reporting**: Create a detailed report with screenshots, findings, and remediation suggestions.
7. **GitHub Deployment**: Push your project to GitHub to showcase your work to recruiters and peers.

🧠 Why This Matters

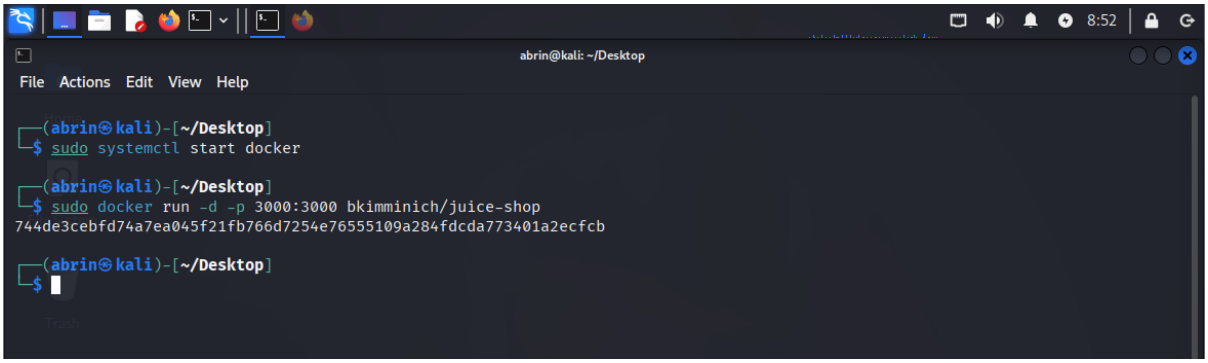
This hands-on project helps you:

- Understand how attackers think and operate
- Learn to use industry-standard tools
- Build a cybersecurity portfolio with real evidence
- Prepare for bug bounty programs or security certification

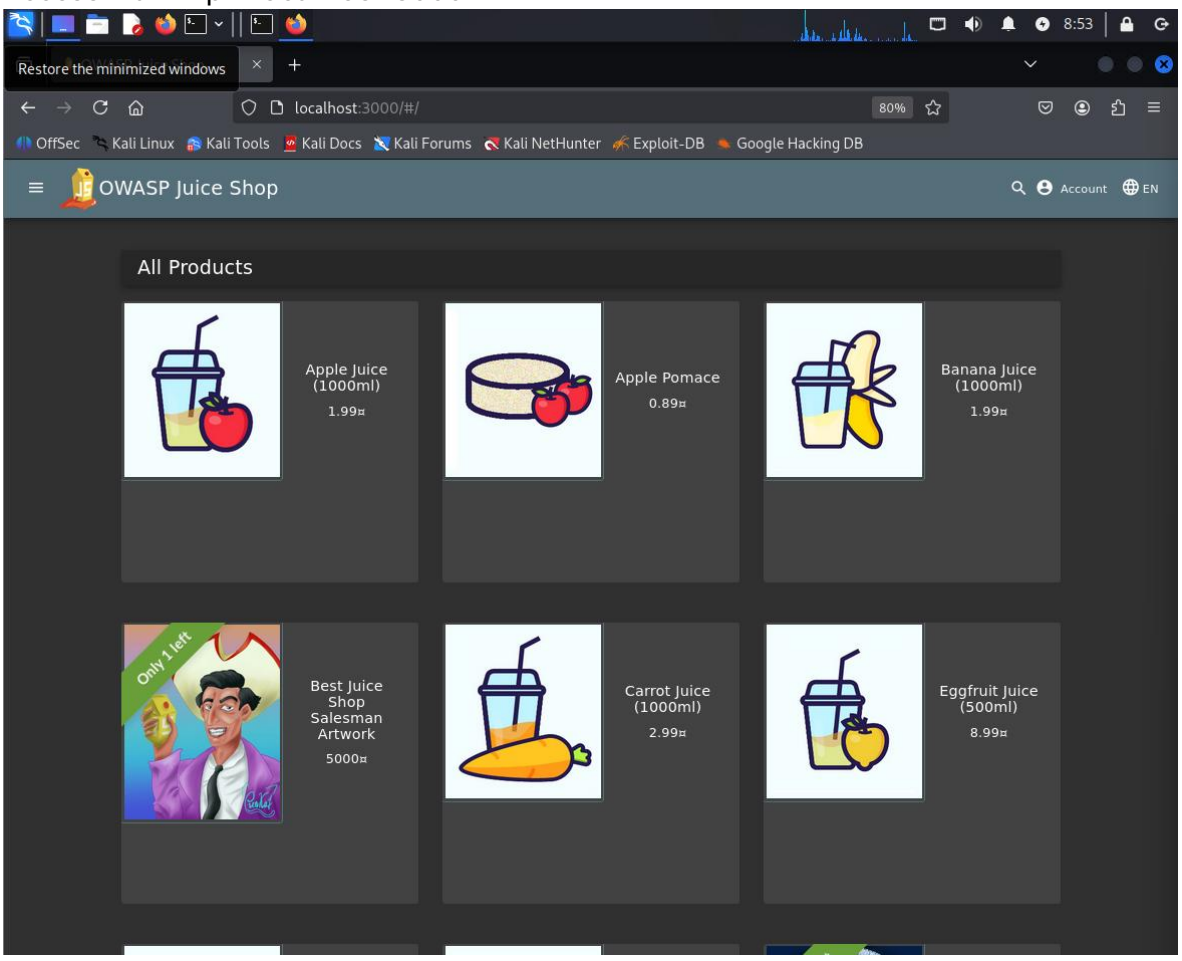
1. Step 1: Lab Setup

Install OWASP Juice Shop

```
sudo apt update
sudo apt install docker.io -y
sudo systemctl start docker
sudo docker pull bkimminich/juice-shop
sudo docker run -d -p 3000:3000 bkimminich/juice-shop
```



Access it at http://localhost:3000



Step 2: Reconnaissance

Reconnaissance is the **first active phase** in penetration testing where you gather information about the target system. It helps you understand:

- What services are running
- Which ports are open
- What operating system and software versions are in use
- Potential entry points for exploitation

There are two types:

- **Passive Recon**: Observing without interacting (e.g., Google dorking, WHOIS lookup)
- **Active Recon**: Direct interaction with the target (e.g., Nmap scans)

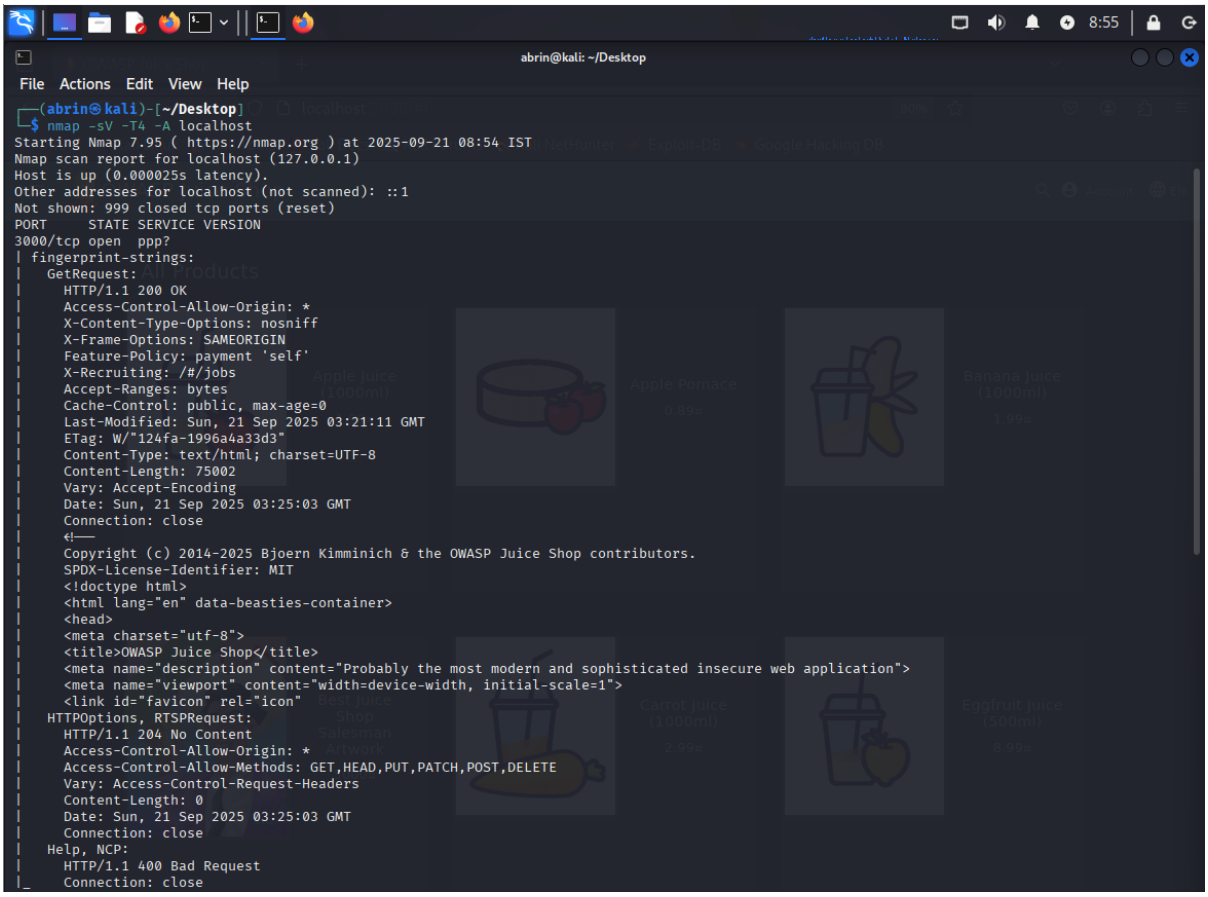
In your Juice Shop project, you're doing **active recon** using tools like Nmap and WhatWeb.

🔎 Scan with Nmap

🔎 What is Nmap?

Nmap (Network Mapper) is a powerful open-source tool used to:

- Discover hosts and services on a network
- Identify open ports
- Detect service versions and operating systems
- Perform vulnerability scans



🔍 Command Breakdown:

```
nmap -sV -T4 -A localhost
```

-sV	Detects **service versions** (e.g., Apache 2.4.41)
-T4	Sets **aggressive timing** for faster scanning (T0 to T5; T4 is fast but still stable)
-A	Enables **OS detection**, **version detection**, **script scanning**, and **traceroute**
localhost	Target is your own machine (where Juice Shop is running)

🧠 What You'll Learn from This Scan:

- Which ports are open (e.g., 3000 for Juice Shop)
- What services are running (e.g., Node.js, Express)
- OS fingerprinting results

- Potential vulnerabilities via Nmap scripts

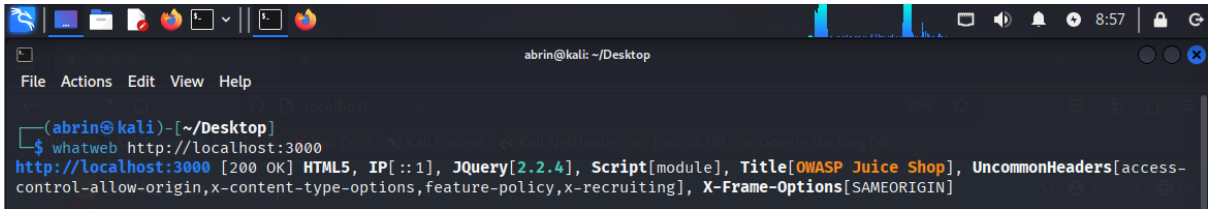
🔎 Fingerprint with WhatWeb

🔎 What is WhatWeb?

WhatWeb is a web scanner used for fingerprinting websites. It identifies technologies used by a web application, such as:

- Web servers (e.g., Express, Apache)
- Frameworks (e.g., Angular, React)
- CMS (e.g., WordPress, Joomla)
- Programming languages (e.g., PHP, Node.js)
- Analytics tools, cookies, and more

It's fast, lightweight, and perfect for reconnaissance.



🧪 What Happens When You Run whatweb http://localhost:3000

You're scanning the Juice Shop running on your local machine. Here's what each part does:

whatweb	Launches the WhatWeb tool
http://localhost:3000	Targets the Juice Shop app running on port 3000

🧠 What You'll Discover:

- **Server type**: Likely Node.js with Express
- **Cookies**: Session or tracking cookies
- **JavaScript libraries**: AngularJS, Bootstrap, etc.
- **Security headers**: CSP, X-Frame-Options, etc.
- **Other metadata**: HTML comments, analytics, etc.

Step 3: Vulnerability Assessment

🔎 Run Nikto

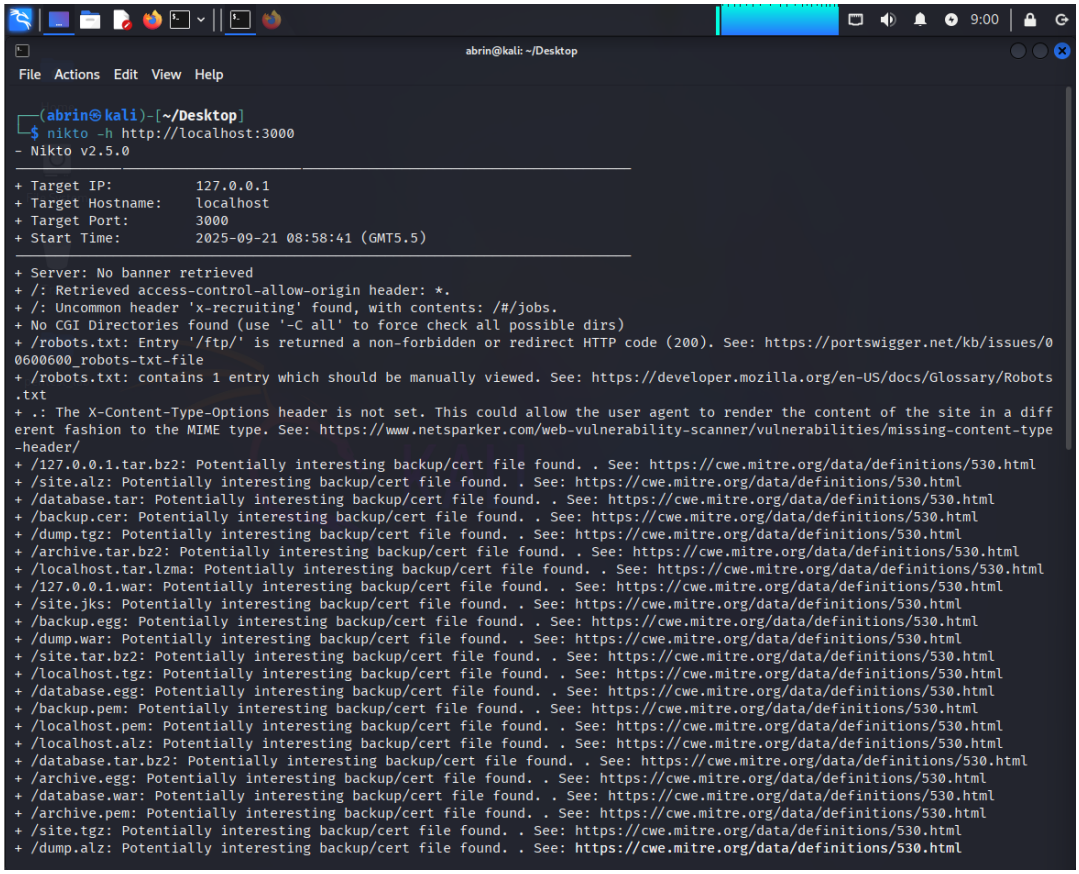
📌 What is Nikto?

Nikto is a web server vulnerability scanner that checks for:

- Dangerous files and directories
- Outdated server software
- Misconfigurations
- Security headers
- Known vulnerabilities

🧪 What Happens in This Scan:

- Nikto sends multiple HTTP requests to Juice Shop
- It analyzes the server response for signs of weakness
- You'll get a report showing:
- HTTP methods allowed (e.g., PUT, DELETE)
- Missing security headers (e.g., X-Frame-Options)
- Known vulnerabilities in the server software



🔎 Use Burp Suite

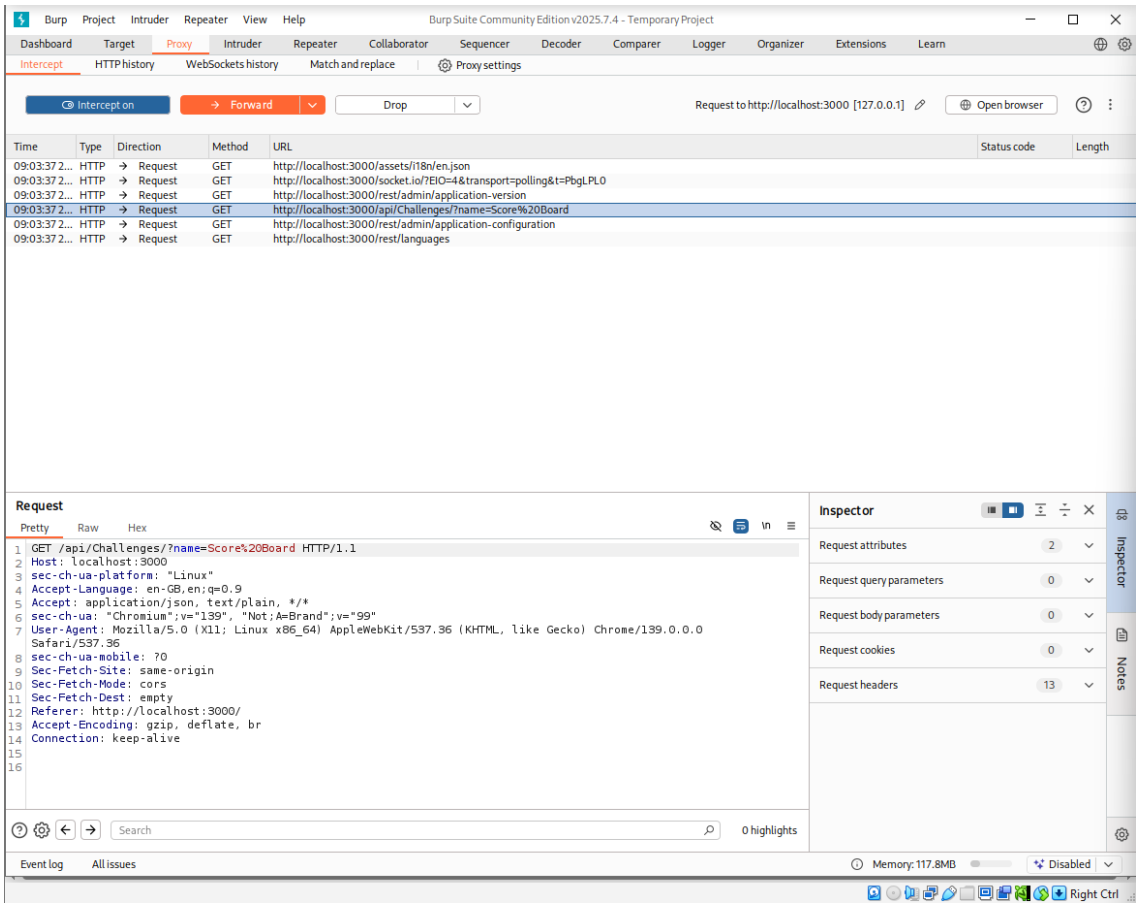
🛠 Steps:

1. **Open Burp Suite** in Kali Linux
2. **Configure Browser Proxy**:
    - Set your browser's proxy to 127.0.0.1:8080
    - This routes traffic through Burp Suite
3. **Intercept Login Requests**:

- Go to Juice Shop login page
- Enter dummy credentials
- Burp Suite will intercept the HTTP POST request
- You can inspect:
- Request headers
- Parameters (e.g., username, password)
- Cookies
- Response from the server

🧠 Why This Matters:

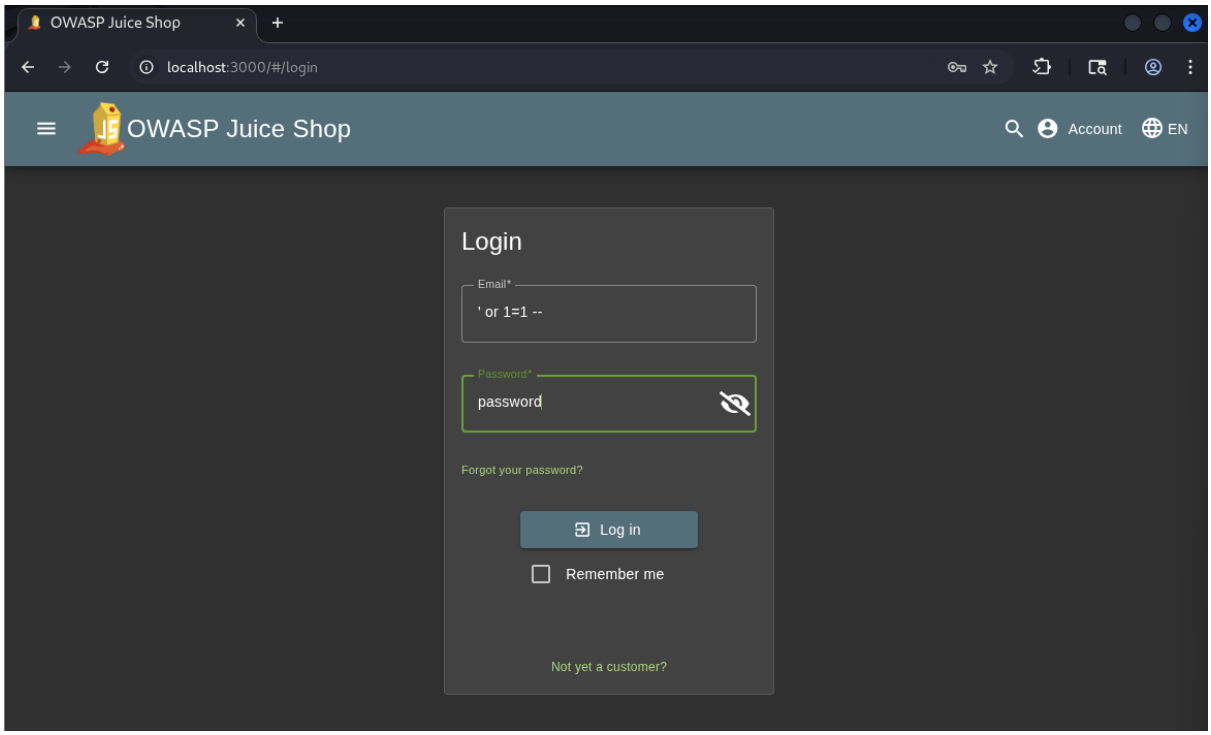
- You can analyze how login data is sent
- Modify requests to test for SQL injection, XSS, etc.
- Understand session handling and authentication flow



Step 4: Exploitation

🧨 SQL Injection (Manual)

SQL Injection is a web security vulnerability that allows an attacker to interfere with the queries an application makes to its database. It happens when user input **is not properly sanitized**, allowing malicious SQL code to be executed.



🔍 Breakdown of the Payload:

'	Closes the original string in the SQL query
OR 1=1	A condition that is always **true**
--	SQL comment syntax — **ignores the rest** of the query (like password check)

🧠 What Happens Behind the Scenes:

Assume the original SQL query looks like this:

```
SELECT * FROM users WHERE username = '[input]' AND password =
'[input]';
```

After injection, it becomes:

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

- OR 1=1 makes the condition always true
- -- comments out the rest of the query
- The database returns **all users**, and the app logs you in as the first on

⚠️ **Why This Is Dangerous**

- Bypasses authentication
- Grants unauthorized access
- Can lead to **data theft**, **account takeover**, or **full system compromise**

Step 5: Post Exploitation

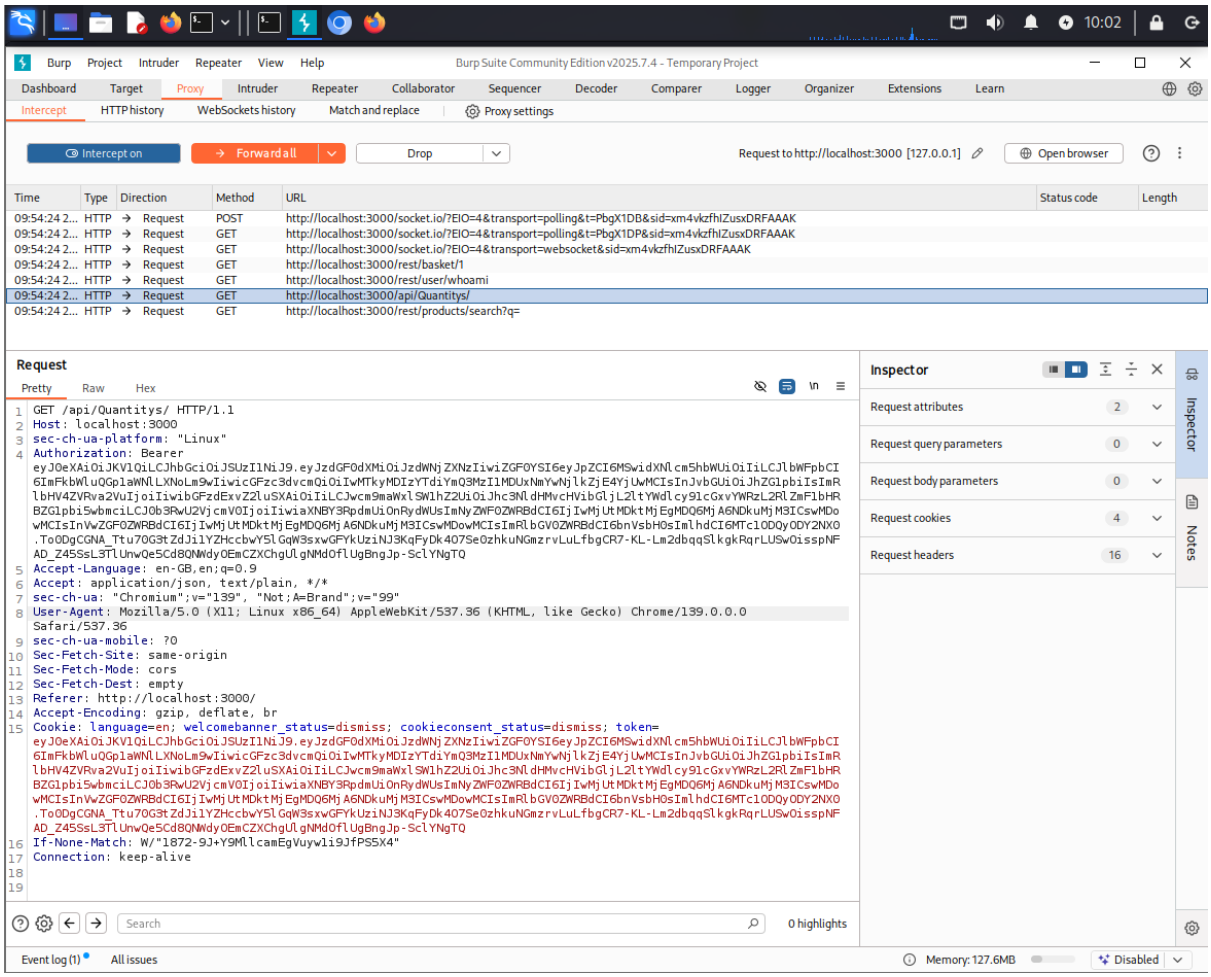
🗃️ Access Sensitive Data

After bypassing login via SQL injection, you're technically logged in — but to interact with protected endpoints like **/rest/user/whoami**, you need to prove your identity to the server. That's where the session token comes in.

🧠 What Is a Session Token?

When you log in, the server creates a **session token** and sends it to your browser as a **cookie**. This token:

- Identifies you as an authenticated user
- Is stored in your browser
- Is sent with every request to protected resources



This confirms you're authenticated and can now:

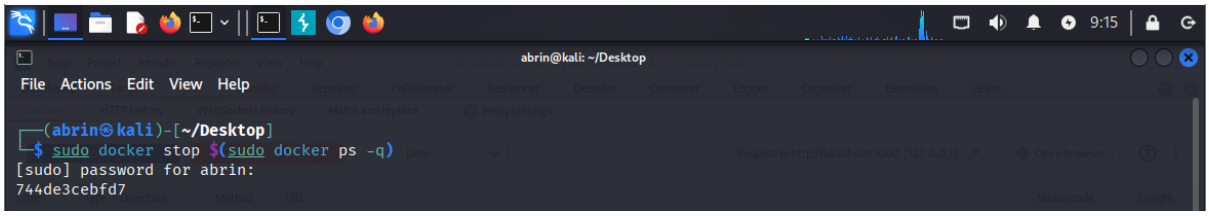
- Access user-specific data
- Explore other protected endpoints
- Simulate privilege escalation

🛡️ Remediation

- Implement input validation and parameterized queries
- Secure API endpoints with authentication and authorization
- Add proper security headers to HTTP responses
- Regularly update and patch server software

Step 6: Cleanup

Stop Docker container:



Step 7: Reporting

report.md with the following sections:

📌 Scope

- Perform a VAPT on OWASP Juice Shop using Kali Linux
- Identify and exploit vulnerabilities in a controlled environment

🛠 Tools Used

- Kali Linux
- Docker
- Nmap
- WhatWeb
- Nikto
- Burp Suite
- Metasploit
- Curl

📸 Screenshots

- Juice Shop homepage
- Nmap scan results
- WhatWeb fingerprinting
- Nikto vulnerability scan
- Burp Suite intercepted login
- SQL injection success
- Sensitive data or order history via API