OUR TEAM

OMAR ATEF MAHMOUD 2305099

OSAMA MAHMOUD MOHAMED 2305572

HOPE YOU LIKE IT

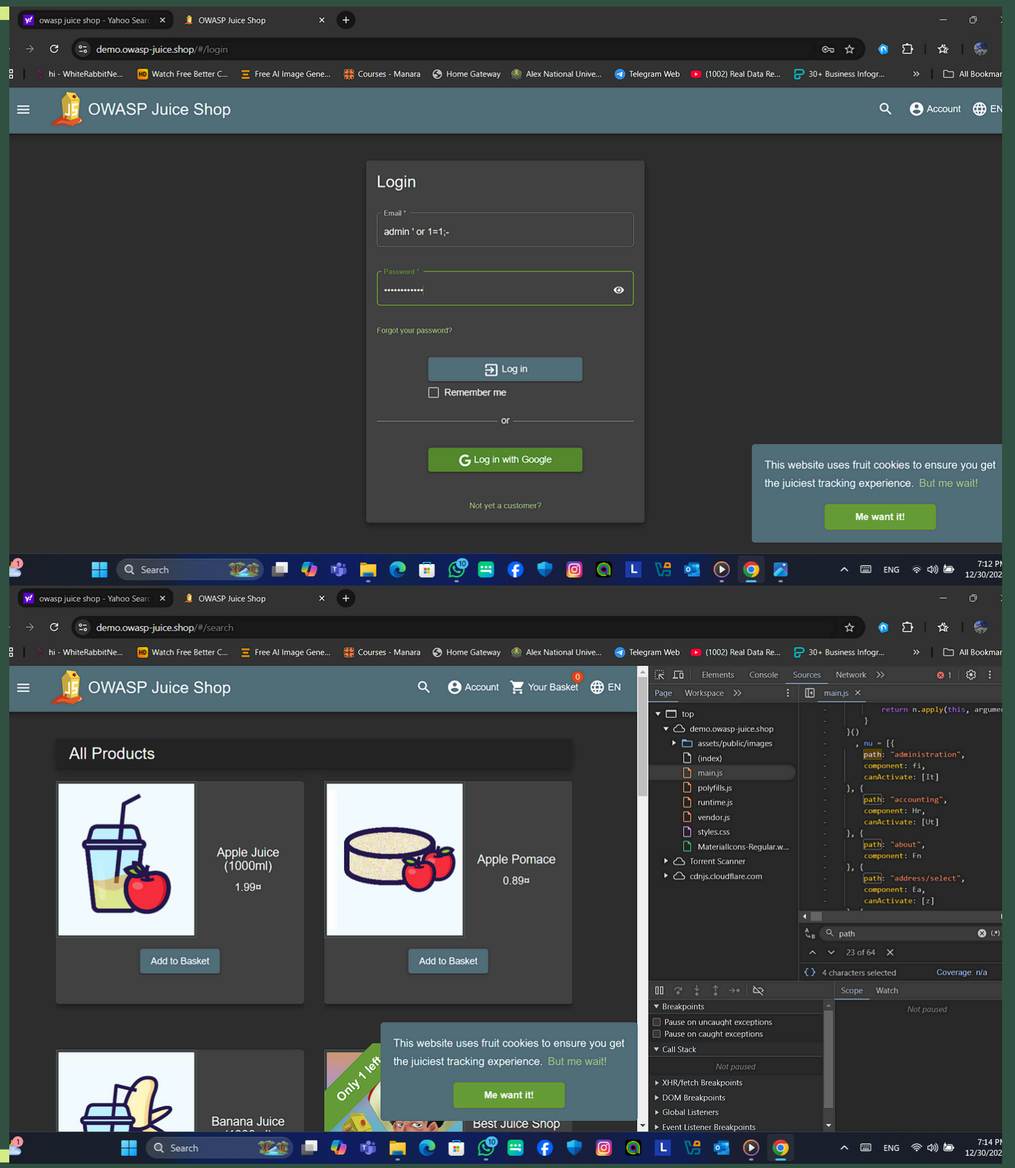
SQL INJECTION

1- used user input field to insert malicious SQL payload ”admin’ or 1=1=;–”

2- became able to Gain an administrative privileges “Authentication Bypass”

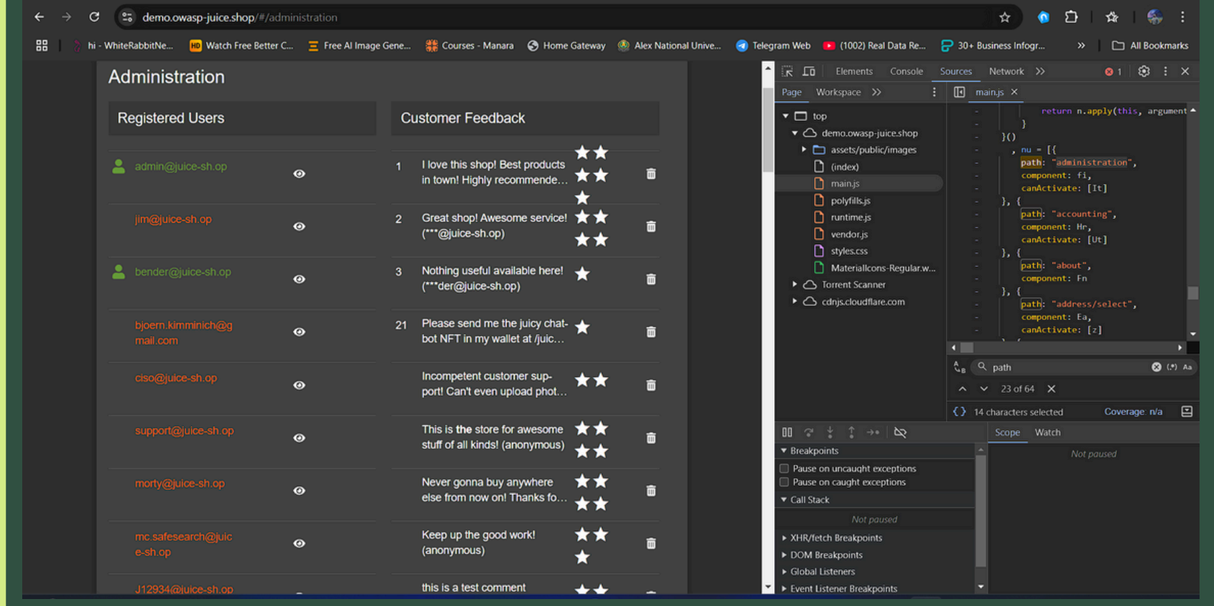
3- Unauthorized access to sensitive information

4- gained access to the admin website.



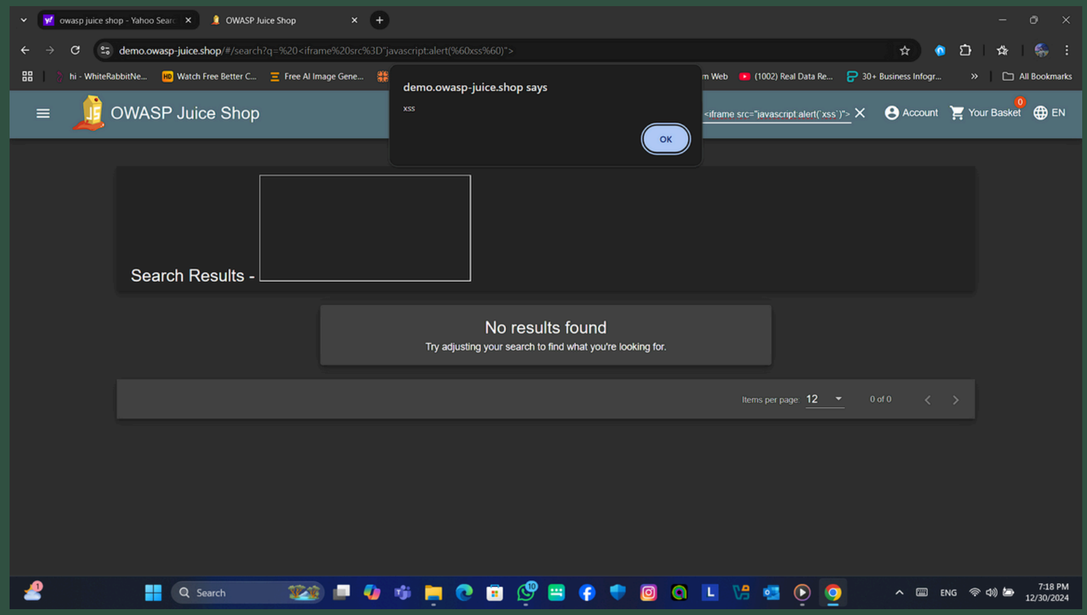
SQL INJECTION

5 - gained access to the usernames of users which is a real sensetive info due to the the data breach



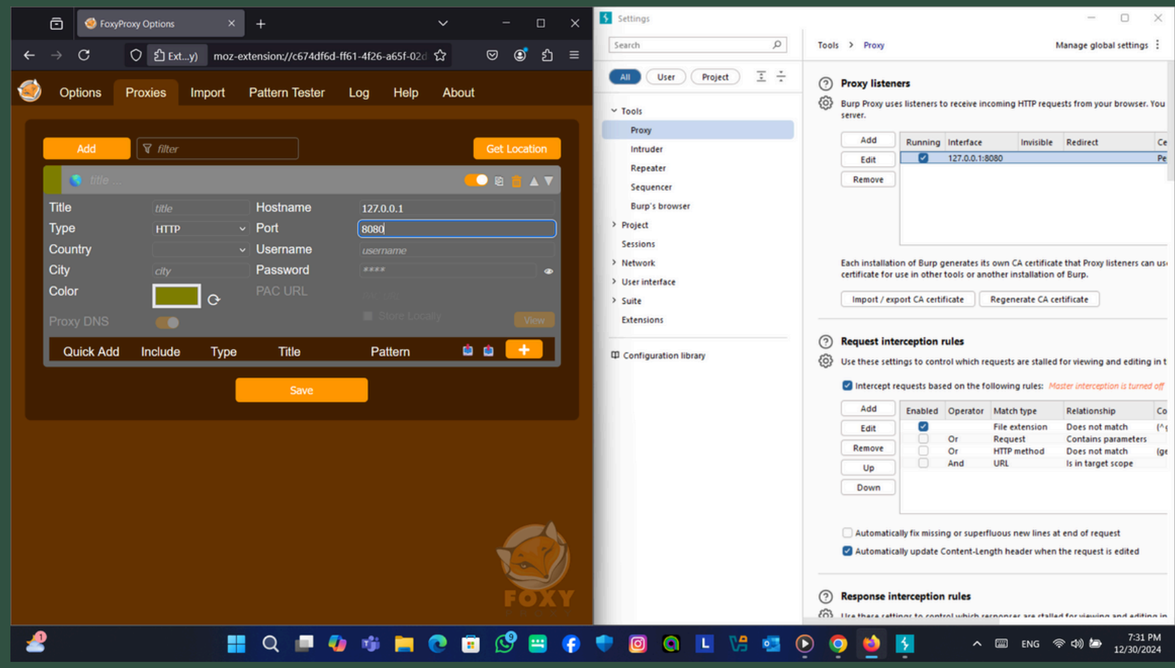
XSS ATTACK

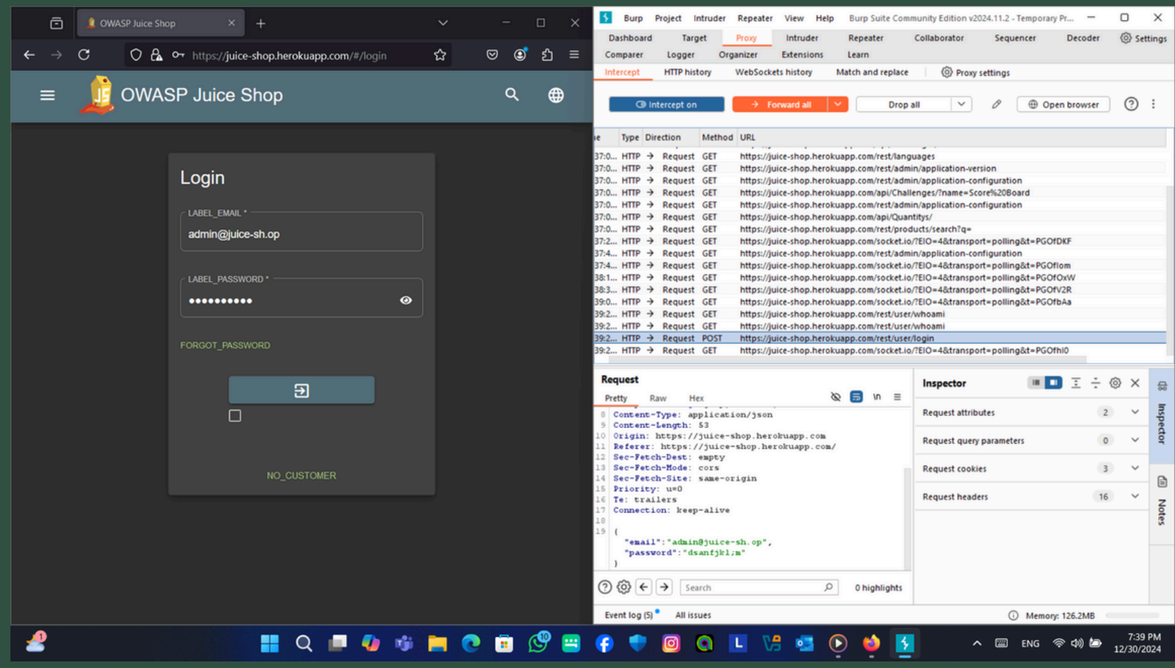
Example of a Basic XSS Attack by submitting this script into the search box



BRUTE FORCE ON ADMIN CREDENTIALS

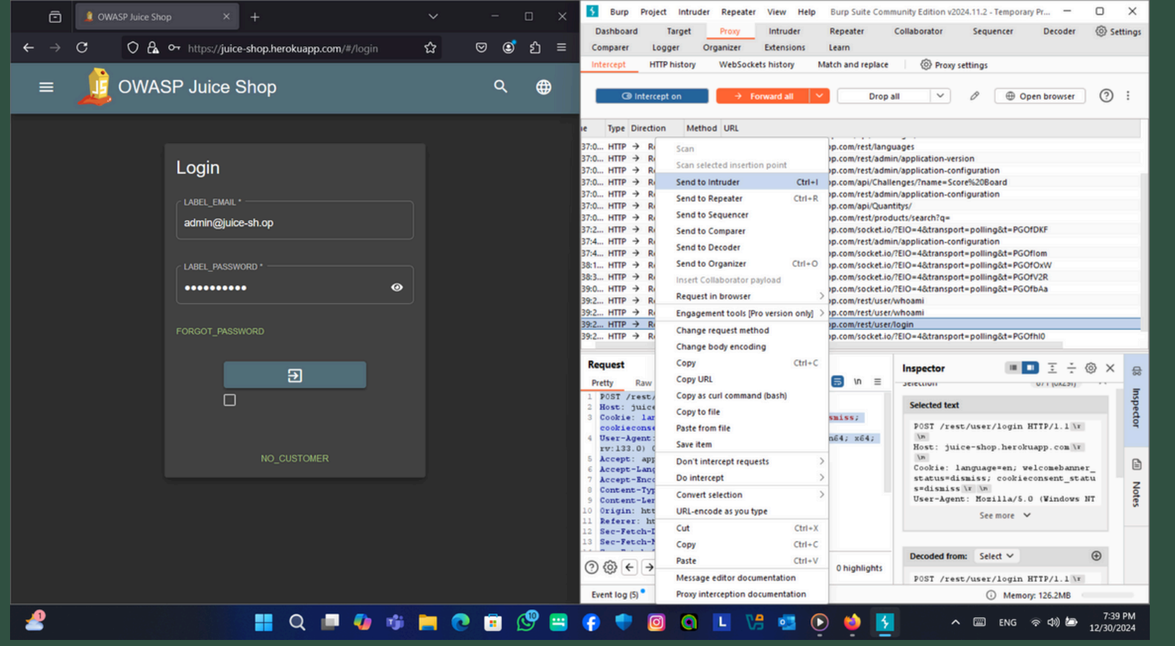
1. sets up a proxy on Firefox using the FoxyProxy extension.
2. They intercept a login request using Burp Suite.
3. The POST request containing the login credentials is sent to Hydra, a password-cracking tool.

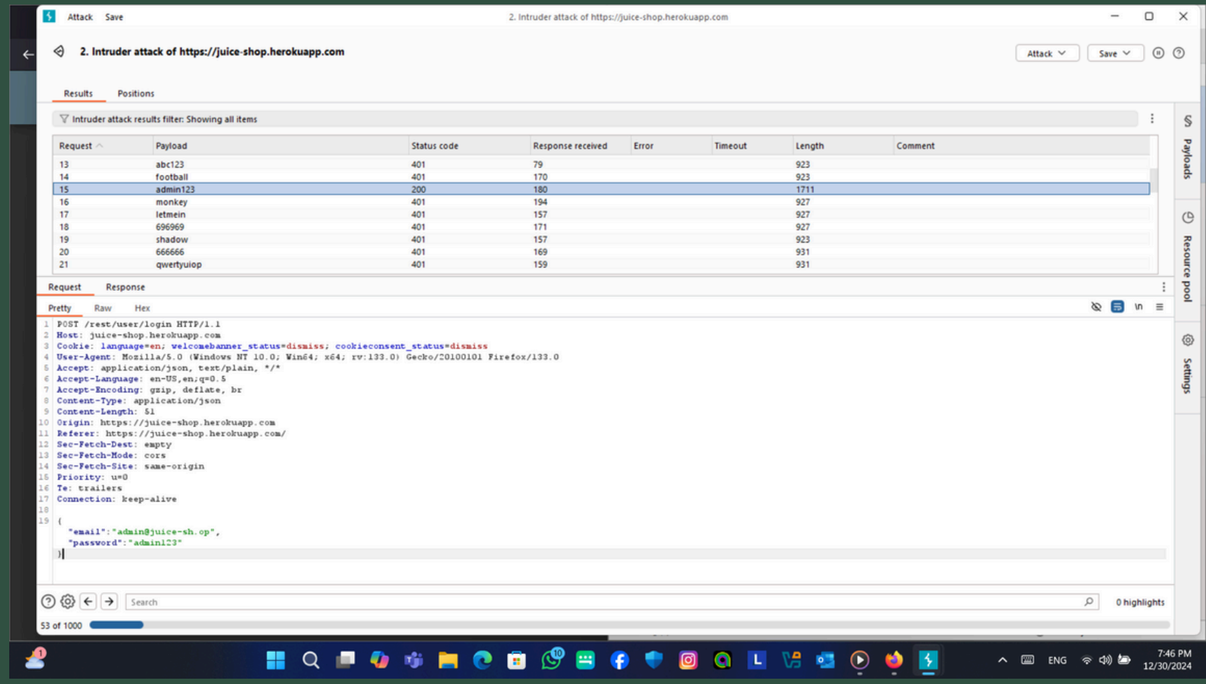




BRUTE FORCE ON ADMIN CREDENTIALS

1. Hydra uses a wordlist to guess the correct password for the admin account.
2. Once the password is found, the attacker logs in as the admin.





FINALLY HOW TO DEFENCE AGAINST THESE ATTACKS

Validate All User Inputs: Always check and sanitize anything users type into forms, search bars, or URLs.

Use Prepared Statements: Prevent SQL injection by using prepared or parameterized queries instead of directly inserting user input into database commands.

Escape Output and Sanitize Content: For XSS, encode user-provided content before displaying it to prevent harmful scripts from running.

Add CAPTCHA and Rate Limits: Use these on login forms to prevent automated attacks like Hydra.

Implement HTTPS Everywhere: Encrypt all data exchanges between users and the website to prevent interception by tools like Burp Suite.

Monitor and Block Malicious Activity: Regularly review logs, detect suspicious patterns, and block problematic IPs.

Enforce Strong Password Policies: Require complex passwords, use account lockouts, and encourage users to update their passwords regularly.

Secure Your APIs: Protect APIs with authentication and authorization mechanisms to prevent unauthorized access.

Install Security Tools: Use Web Application Firewalls (WAFs) and regular vulnerability scanning tools to keep your website safe.

Educate Developers and Users: Ensure developers know secure coding practices and teach users about strong passwords and phishing awareness.