Developer’s Hub (Cyber Security Internship)
Submitted by Hasan Raza

<u>Week 1: Security Assessment</u>

1. Understand the Application

Setting up the application:

Made a separate directory:

```
┌──(hasanraza㉿kali)-[~]
└─$ mkdir ~/webgoat

┌──(hasanraza㉿kali)-[~]
└─$ cd ~/webgoat
```

Downloaded the setup:

```
┌──(hasanraza㉿kali)-[~/webgoat]
└─$ wget https://github.com/WebGoat/WebGoat/releases/download/v2025.3/webgoat-2025.3.jar

--2025-07-30 16:20:59--  https://github.com/WebGoat/WebGoat/releases/download/v2025.3/webgoat-2025.3.jar
Resolving github.com (github.com)... 20.207.73.82
Connecting to github.com (github.com)|20.207.73.82|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://release-assets.githubusercontent.com/github-production-release-asset/31771754/08d798b2-f725-4765-9062-0f104bc5d341?sp=r&sv=2018-11-09&sr=b&spr=https&se=2025-07-30T11%3A33%3A30Z&rscd=attachment
%3B+filename%3Dwebgoat-2025.3.jar&rsct=application%2Foctet-stream&skoid=96c2d410-5711-43a1-aedd-ab1947aa7ab0&sktid=398a6654-997b-47e9-b12b-9515b896b4de&skt=2025-07-30T10%3A33%3A01Z&ske=2025-07-30T11%3A33%3A30Z&
sks=b&skv=2018-11-09&sig=n3f1HaGgEw8csfE2r3DGZ5E8VQN%2BOdDUGYJ4vAuy%2Fc%3D&jwt=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJnaXRodWIuY29tIiwiYXVkIjoicmVsZWFzZS1hc3NldHMuZ2l0aHVidXNlcmNvbnRlbnQuY29tIiwia2V5I
joia2V5MSIsImV4cCI6MTc1Mzg3Mjk2MSwibmJmIjoxNzUzODcyNjYxLCJwYXRoIjoicmVsZWFzZWFzc2V0cHJvZHVjdGlvbi5ibG9iLmNvcmUud2luZG93cy5uZXQifQ.p9Jlwc8Jn4JprCd0HOgBHr1uMeQEN_4DQVOI5Eg0eC8&response-content-disposition=attachm
ent%3B%20filename%3Dwebgoat-2025.3.jar&response-content-type=application%2Foctet-stream [following]
--2025-07-30 16:20:59--  https://release-assets.githubusercontent.com/github-production-release-asset/31771754/08d798b2-f725-4765-9062-0f104bc5d341?sp=r&sv=2018-11-09&sr=b&spr=https&se=2025-07-30T11%3A33%3A30Z&
rscd=attachment%3B+filename%3Dwebgoat-2025.3.jar&rsct=application%2Foctet-stream&skoid=96c2d410-5711-43a1-aedd-ab1947aa7ab0&sktid=398a6654-997b-47e9-b12b-9515b896b4de&skt=2025-07-30T10%3A33%3A01Z&ske=2025-07-30
T11%3A33%3A30Z&sks=b&skv=2018-11-09&sig=n3f1HaGgEw8csfE2r3DGZ5E8VQN%2BOdDUGYJ4vAuy%2Fc%3D&jwt=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJnaXRodWIuY29tIiwiYXVkIjoicmVsZWFzZS1hc3NldHMuZ2l0aHVidXNlcmNvbnRlbn
QuY29tIiwia2V5Ijoia2V5MSIsImV4cCI6MTc1Mzg3Mjk2MSwibmJmIjoxNzUzODcyNjYxLCJwYXRoIjoicmVsZWFzZWFzc2V0cHJvZHVjdGlvbi5ibG9iLmNvcmUud2luZG93cy5uZXQifQ.p9Jlwc8Jn4JprCd0HOgBHr1uMeQEN_4DQVOI5Eg0eC8&response-content-disp
osition=attachment%3B%20filename%3Dwebgoat-2025.3.jar&response-content-type=application%2Foctet-stream
Resolving release-assets.githubusercontent.com (release-assets.githubusercontent.com)... 185.199.110.133, 185.199.108.133, 185.199.109.133, ...
Connecting to release-assets.githubusercontent.com (release-assets.githubusercontent.com)|185.199.110.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 149122725 (142M) [application/octet-stream]
Saving to: ‘webgoat-2025.3.jar’

webgoat-2025.3.jar                  100%[===================================================================================>] 142.21M  1.34MB/s    in 2m 49s

2025-07-30 16:23:48 (864 KB/s) - ‘webgoat-2025.3.jar’ saved [149122725/149122725]
```

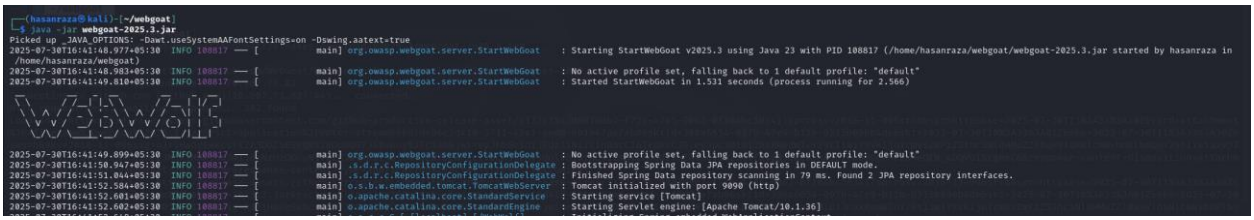
Then installed the java version 23 that is compatible with the WebGoat:

```
┌──(hasanraza㉿kali)-[~]
└─$ sudo update-alternatives --remove-all java

[sudo] password for hasanraza:

┌──(hasanraza㉿kali)-[~]
└─$ curl -s "https://get.sdkman.io" | bash
source "$HOME/.sdkman/bin/sdkman-init.sh"
```

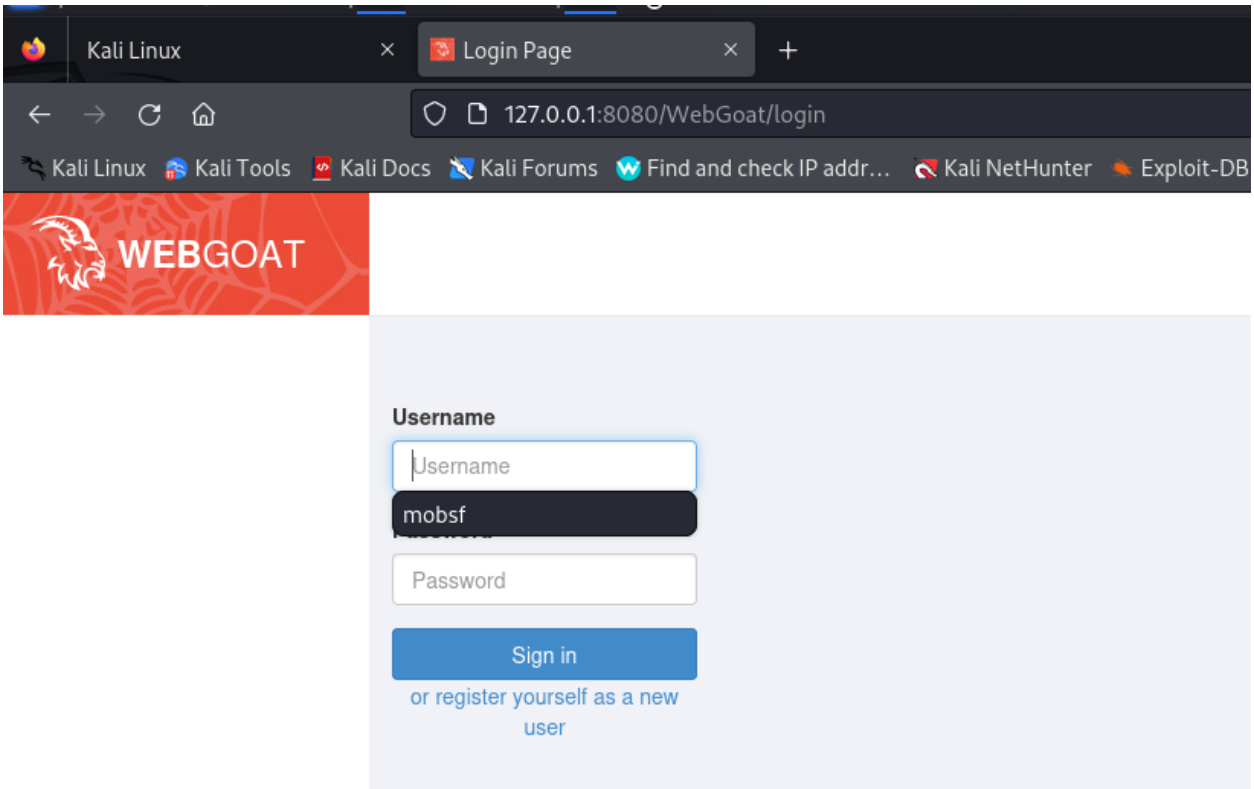
```
All done!

You are subscribed to the STABLE channel.

Please open a new terminal, or run the following in the existing one:

    source "/home/hasanraza/.sdkman/bin/sdkman-init.sh"

Then issue the following command:

    sdk help

Enjoy!!!

┌──(hasanraza㉿kali)-[~]
└─$ sdk install java 23-tem

Downloading: java 23-tem

In progress...

################################################################################################################################################################################################################################################################ 100.0%

Repackaging Java 23-tem...

Done repackaging...

Installing: java 23-tem
Done installing!

Setting java 23-tem as default.

┌──(hasanraza㉿kali)-[~]
└─$ java -version

Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
openjdk version "23" 2024-09-17
OpenJDK Runtime Environment Temurin-23+37 (build 23+37)
OpenJDK 64-Bit Server VM Temurin-23+37 (build 23+37, mixed mode, sharing)
```

Now ran this command: java -jar webgoat-2025.3.jar

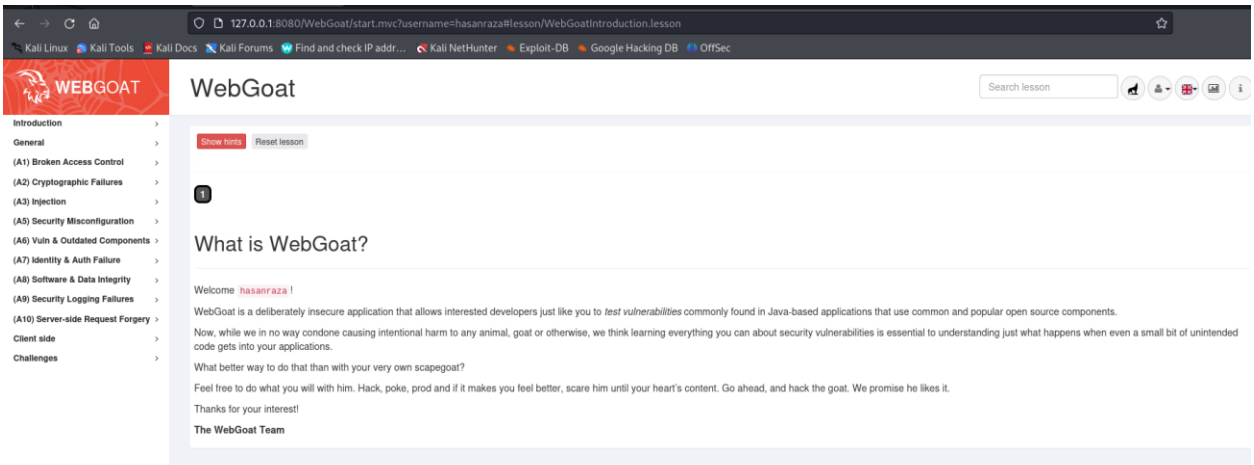
- To start a local web server on `localhost:8080`
- Show logs indicating it's running
- Make WebGoat available in your browser



Then accessed it using Fire Fox:



Signed up as a new user:



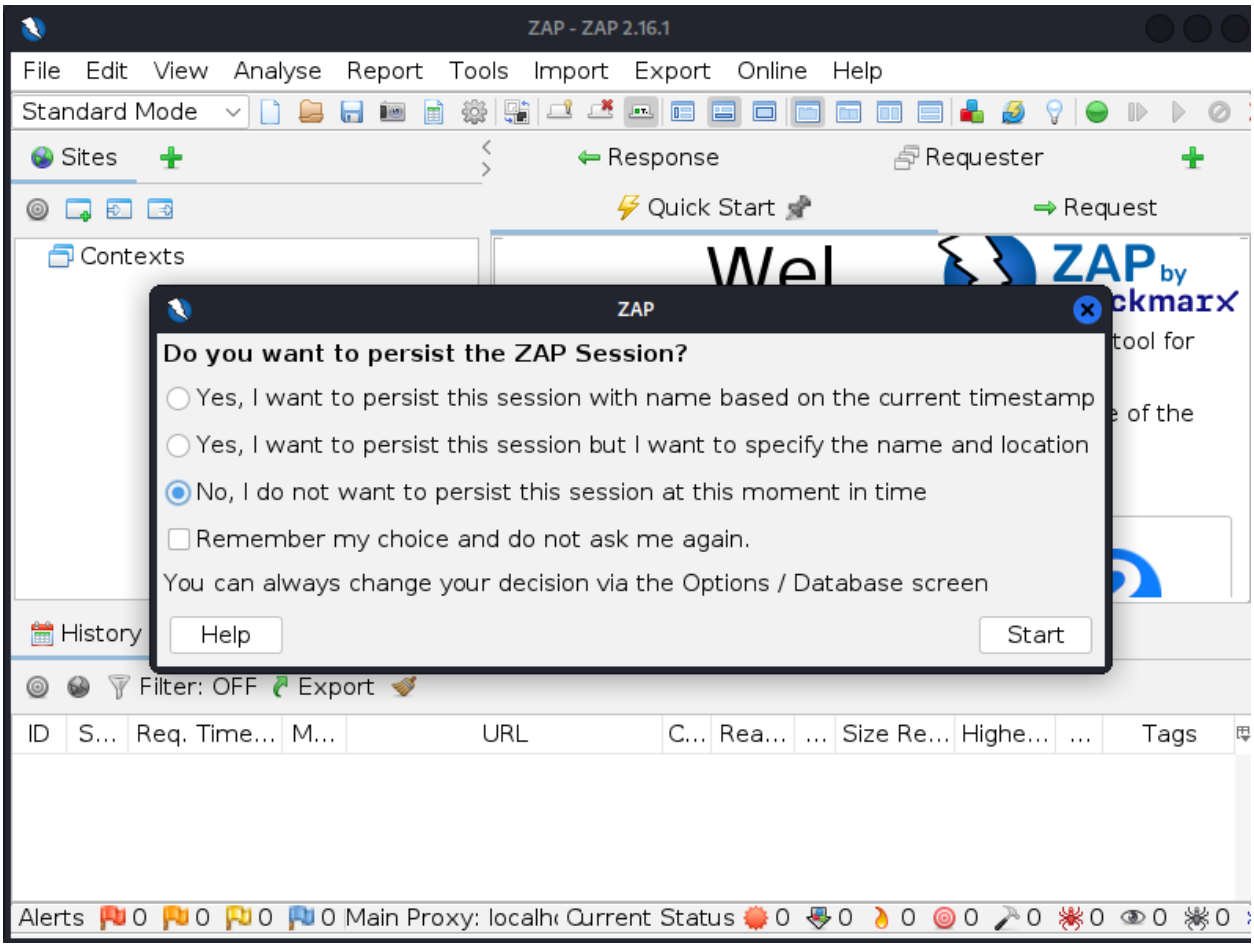
2. Performing Basic Vulnerability Assessment

Use simple tools to identify vulnerabilities:

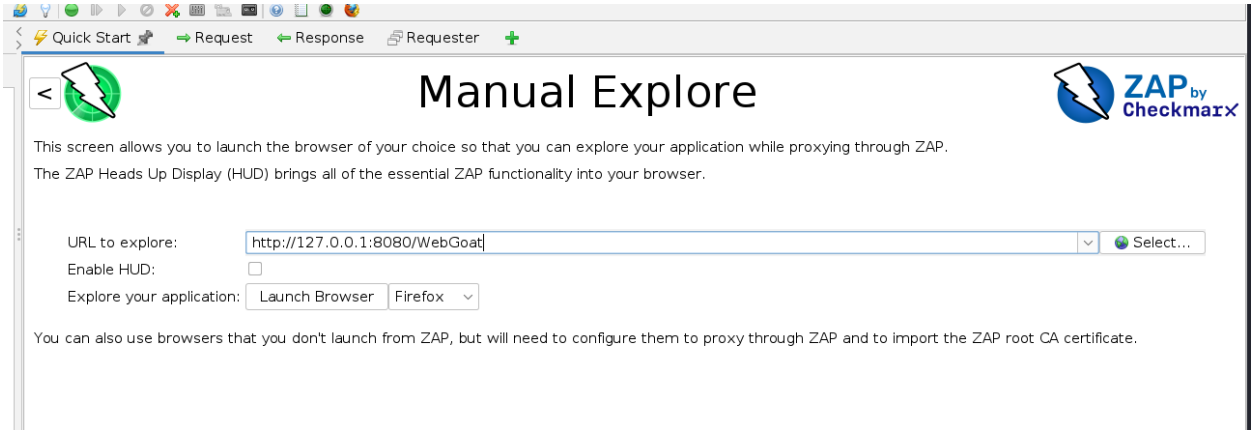
<u>OWASP ZAP</u>: Automated scanner for web app vulnerabilities:

Installed ZAP:

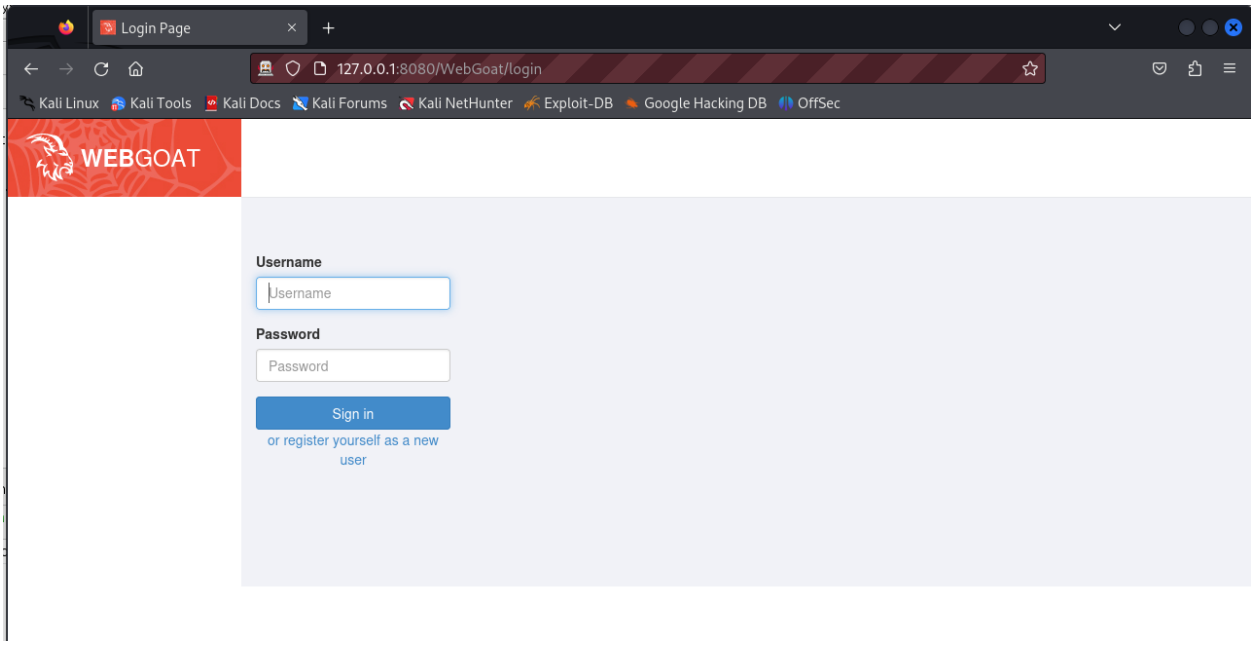
```
┌──(hasanraza㉿kali)-[~]
└─$ sudo apt install zaproxy -y

[sudo] password for hasanraza:
Installing:
  zaproxy

Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 2133
  Download size: 214 MB
  Space needed: 271 MB / 4244 MB available

Get:1 http://kali.download/kali kali-rolling/main amd64 zaproxy all 2.16.1-0kali1 [214 MB]
Fetched 214 MB in 1min 15s (2876 kB/s)
Selecting previously unselected package zaproxy.
(Reading database ... 446813 files and directories currently installed.)
Preparing to unpack .../zaproxy_2.16.1-0kali1_all.deb ...
Unpacking zaproxy (2.16.1-0kali1) ...
Setting up zaproxy (2.16.1-0kali1) ...
Processing triggers for kali-menu (2024.3.1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.

┌──(hasanraza㉿kali)-[~]
└─$ zaproxy &

[2] 116870

┌──(hasanraza㉿kali)-[~]
└─$ Found Java version 23
Available memory: 3922 MB
Using JVM args: -Xmx980m
```



Went to manual explore and typed in the URL :

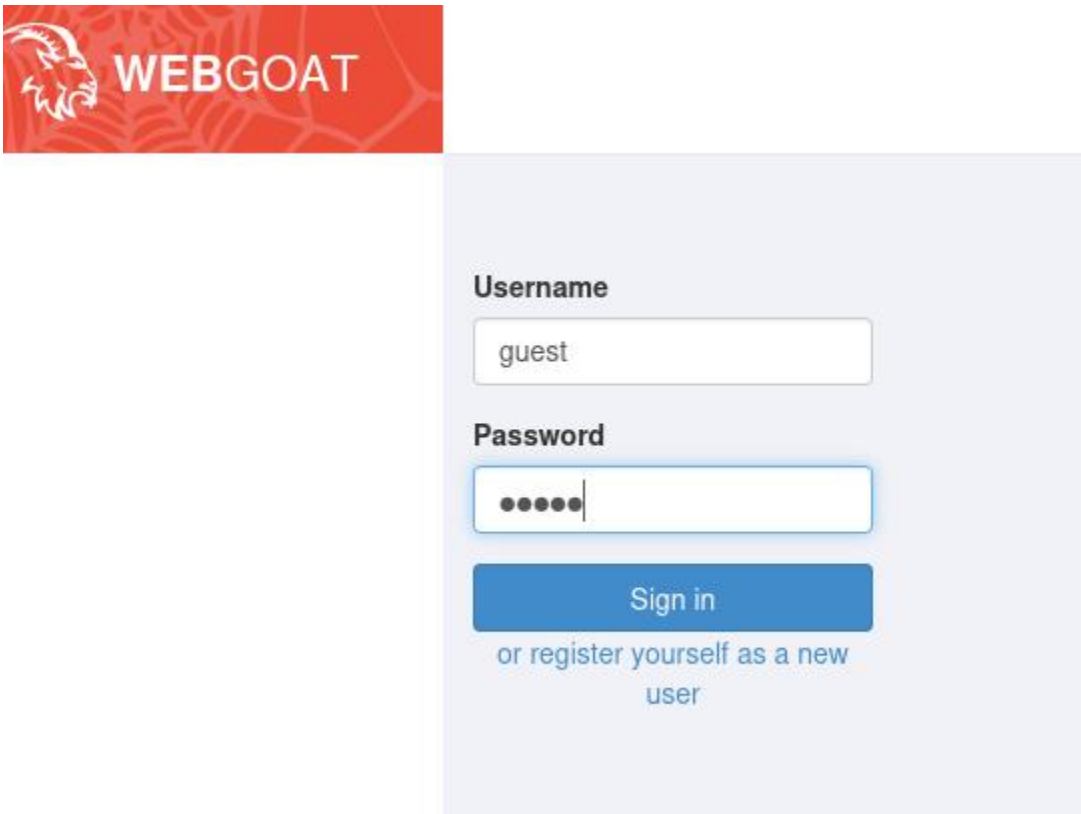


When clicked on launch browser this is prompted:



EXPLORING OPTIONS:

Tried logging in with guest credentials:



It showed invalid username and password

Then login with already created user:

1

What is WebGoat?

Welcome `hasanraza`!

WebGoat is a deliberately insecure application that allows interested developers just like you to *test vulnerabilities* commonly found in Java-based applications that use common and popular open source components.

Now, while we in no way condone causing intentional harm to any animal, goat or otherwise, we think learning everything you can about security vulnerabilities is essential to understanding just what happens when even a small bit of unintended code gets into your applications.

What better way to do that than with your very own scapegoat?

Feel free to do what you will with him. Hack, poke, prod and if it makes you feel better, scare him until your heart's content. Go ahead, and hack the goat. We promise he likes it.

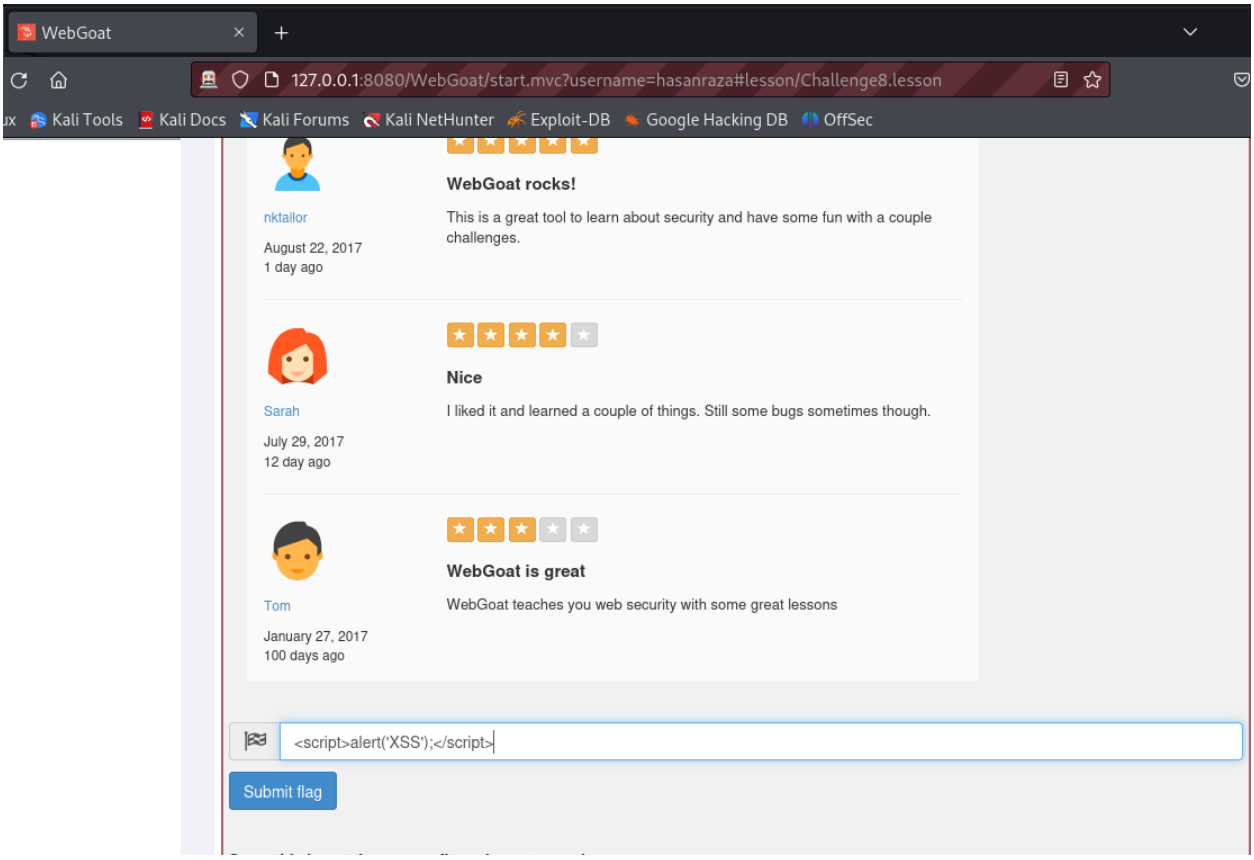
Thanks for your interest!

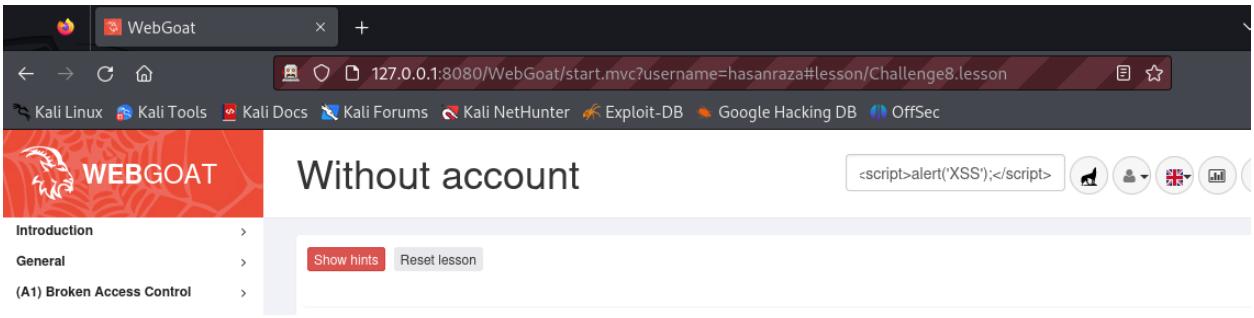
The WebGoat Team

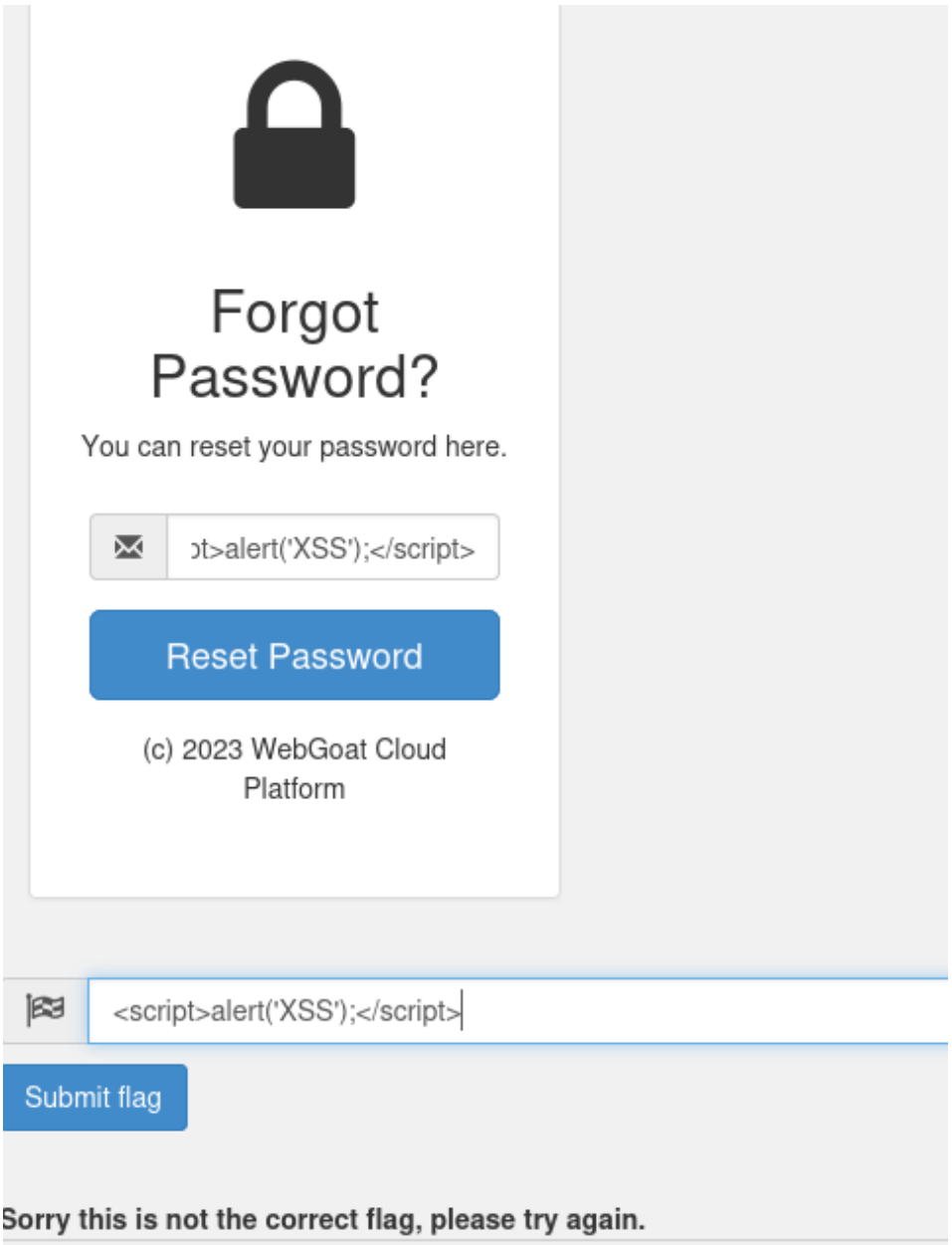
Testing for Cross-Site Scripting (XSS):

In all the possible input fields i typed:

<script>alert('XSS');</script>




WebGoat
127.0.0.1:8080/WebGoat/start.mvc?username=hasanraza#lesson/Challenge8.lesson
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
WEBGOAT
Without account
<script>alert('XSS');</script>
Introduction
General
(A1) Broken Access Control
Show hints
Reset lesson


Forgot Password?
You can reset your password here.
ɔt>alert('XSS');</script>
Reset Password
(c) 2023 WebGoat Cloud Platform
<script>alert('XSS');</script>
Submit flag
Sorry this is not the correct flag, please try again.

Try It! Reflected XSS

The assignment's goal is to identify which field is susceptible to XSS.

It is always a good practice to validate all input on the server side. XSS can occur when unvalidated user input gets used in an HTTP response. In a reflected XSS attack, an attacker can craft a URL with the attack script and post it to another website, email it, or otherwise get a victim to click on it.

An easy way to find out if a field is vulnerable to an XSS attack is to use the `alert()` or `console.log()` methods. Use one of them to find out which field is vulnerable.

Shopping Cart

Shopping Cart Items -- To Buy Now	Price	Quantity	Total
Studio RTA - Laptop/Reading Cart with Tilting Surface - Cherry	69.99	1	$0.00
Dynex - Traditional Notebook Case	27.99	1	$0.00
Hewlett-Packard - Pavilion Notebook with Intel Centrino	1599.99	1	$0.00
3 - Year Performance Service Plan $1000 and Over	299.99	1	$0.00

Enter your credit card number: <script>alert(1)</script>

Enter your three digit access code: <script>alert(1)</script>

Purchase

Seems like you tried to compromise our shop with an reflected XSS attack.
We do our... "best"... to prevent such attacks. Try again!

Try It! Reflected XSS

The assignment's goal is to identify which field is susceptible to XSS.

It is always a good practice to validate all input on the server side. XSS can occur when unvalidated user input gets used in an HTTP response. In a reflected XSS attack, an attacker can craft a URL with the attack script and post it to another website, email it, or otherwise get a victim to click on it.

An easy way to find out if a field is vulnerable to an XSS attack is to use the `alert()` or `console.log()` methods. Use one of them to find out which field is vulnerable.

Shopping Cart

127.0.0.1:8080

1

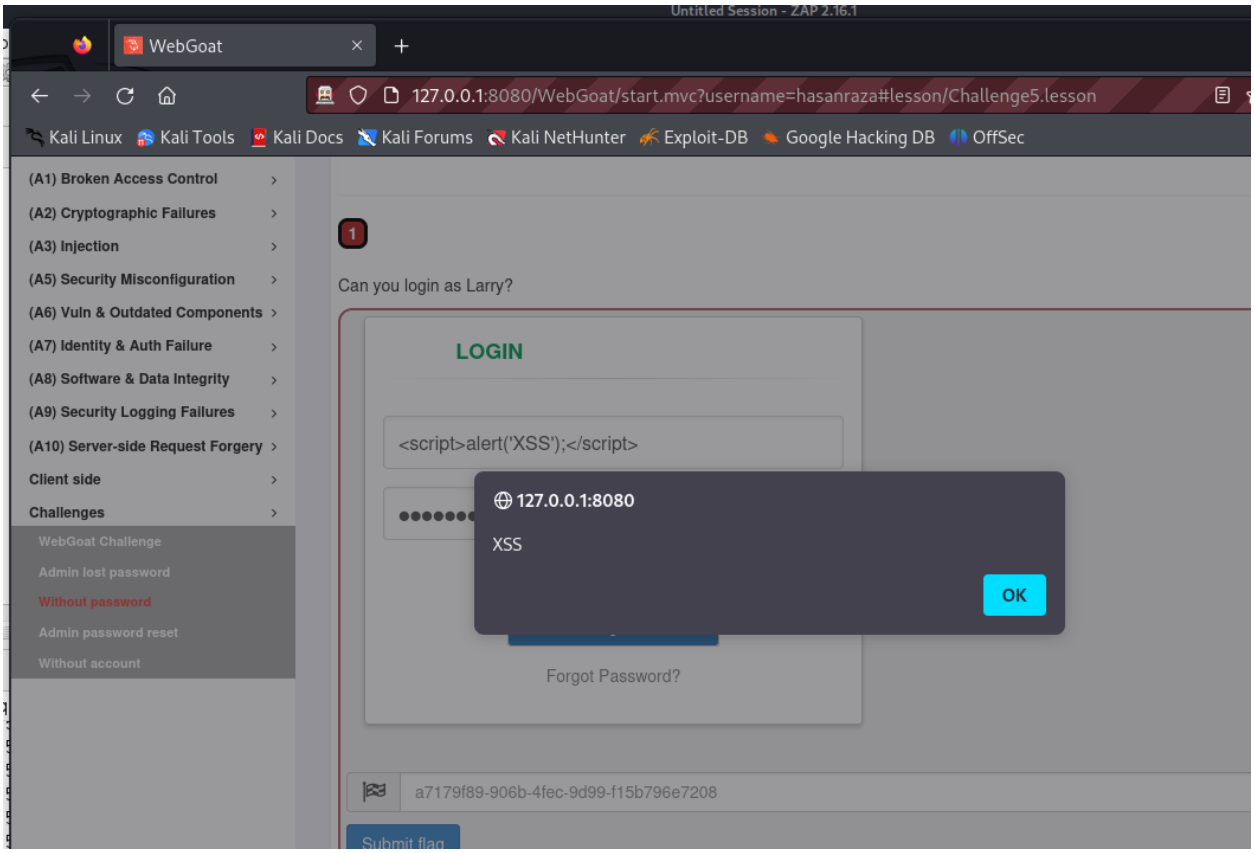
OK

Shopping	Price	Quantity	Total
Studio RT	69.99	1	$0.00
Dynex - T	27.99	1	$0.00
Hewlett-P	1599.99	1	$0.00
3 - Year Performance Service Plan $1000 and Over	299.99	1	$0.00

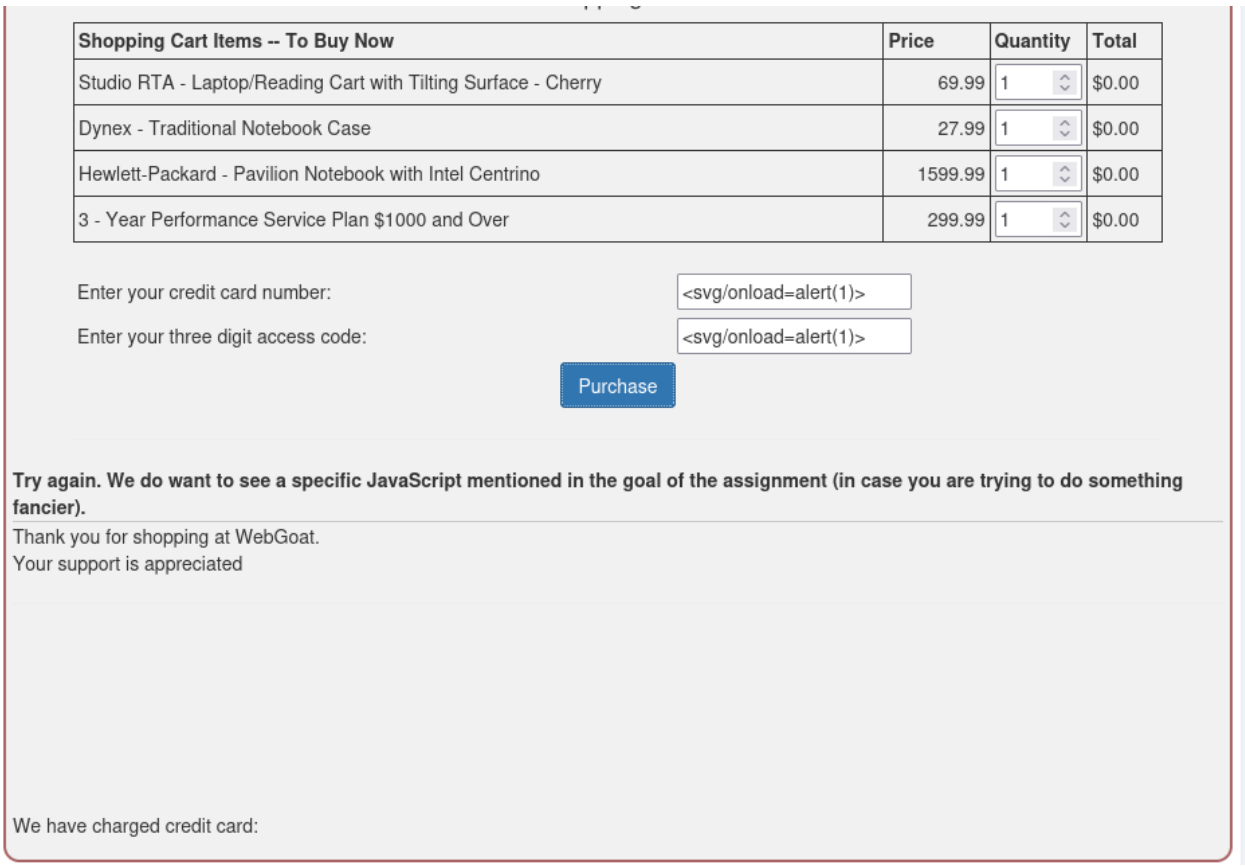
Enter your credit card number: mg src=x onerror=alert(1)>

Enter your three digit access code: mg src=x onerror=alert(1)>

Purchase


Untitled Session - ZAP 2.16.1
WebGoat
127.0.0.1:8080/WebGoat/start.mvc?username=hasanraza#lesson/Challenge5.lesson
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
OffSec
(A1) Broken Access Control
(A2) Cryptographic Failures
(A3) Injection
(A5) Security Misconfiguration
(A6) Vuln & Outdated Components
(A7) Identity & Auth Failure
(A8) Software & Data Integrity
(A9) Security Logging Failures
(A10) Server-side Request Forgery
Client side
Challenges
WebGoat Challenge
Admin lost password
Without password
Admin password reset
Without account
1
Can you login as Larry?
LOGIN
<script>alert('XSS');</script>
127.0.0.1:8080
XSS
OK
Forgot Password?
a7179f89-906b-4fec-9d99-f15b796e7208
Submit flag

But when i tried it with this command: <svg/onload=alert(1)>
it showed success



Shopping Cart Items -- To Buy Now	Price	Quantity	Total
Studio RTA - Laptop/Reading Cart with Tilting Surface - Cherry	69.99	1	$0.00
Dynex - Traditional Notebook Case	27.99	1	$0.00
Hewlett-Packard - Pavilion Notebook with Intel Centrino	1599.99	1	$0.00
3 - Year Performance Service Plan $1000 and Over	299.99	1	$0.00

Enter your credit card number: <svg/onload=alert(1)>

Enter your three digit access code: <svg/onload=alert(1)>

Purchase

Try again. We do want to see a specific JavaScript mentioned in the goal of the assignment (in case you are trying to do something fancier).

Thank you for shopping at WebGoat.
Your support is appreciated

We have charged credit card:

Vulnerability: Reflected XSS (Bypass with image tag)

- **Location**: WebGoat → Cross-Site Scripting → Reflected XSS lesson
- **Payload Used**:

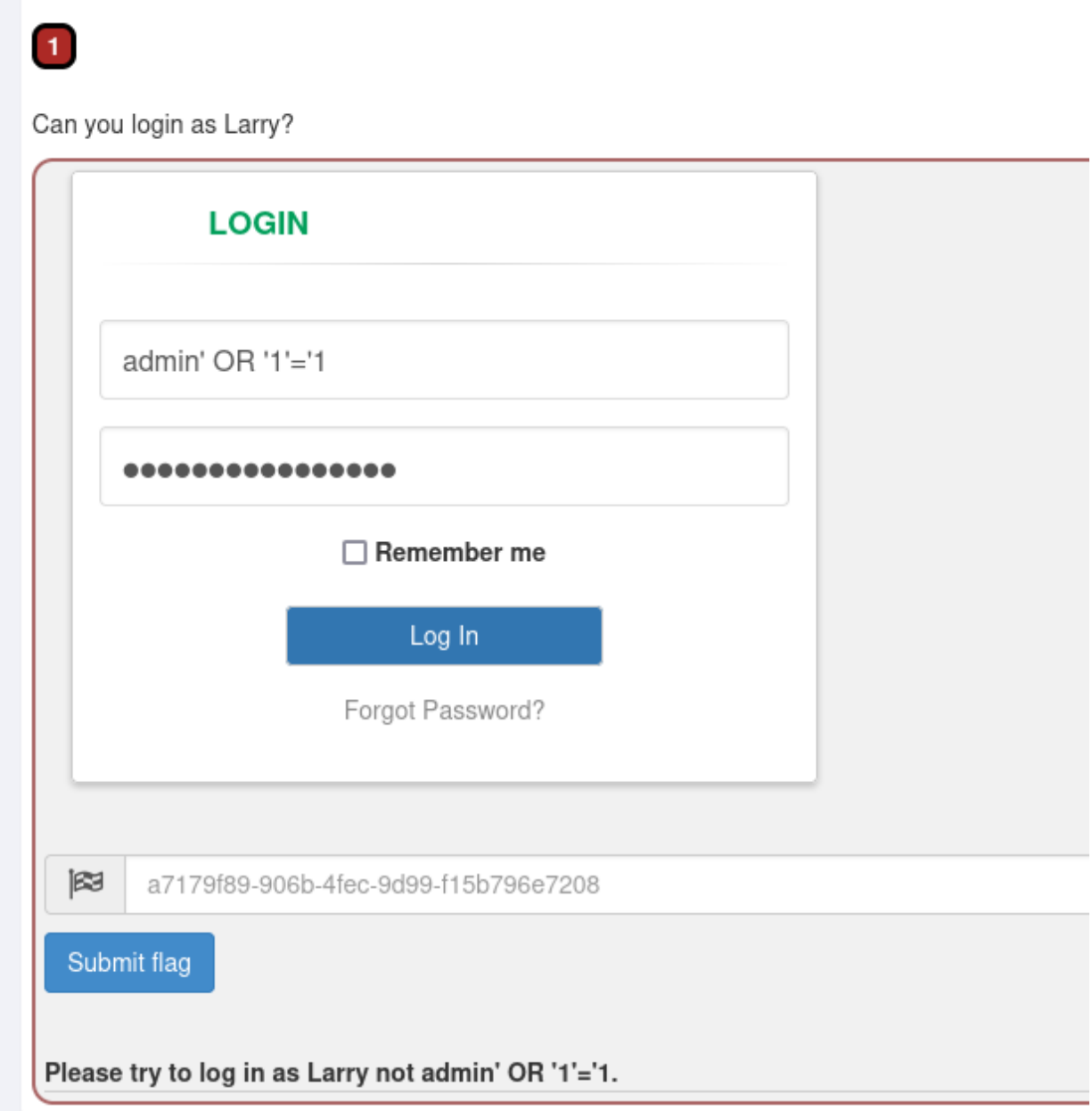
<img src=x onerror=alert(1)>

- **Observation**: Alert popup was triggered despite WebGoat claiming basic protections.
- **Impact**: JavaScript code execution through input field.
- **Recommendation**: Use context-aware encoding and stricter input validation to prevent bypasses like this.

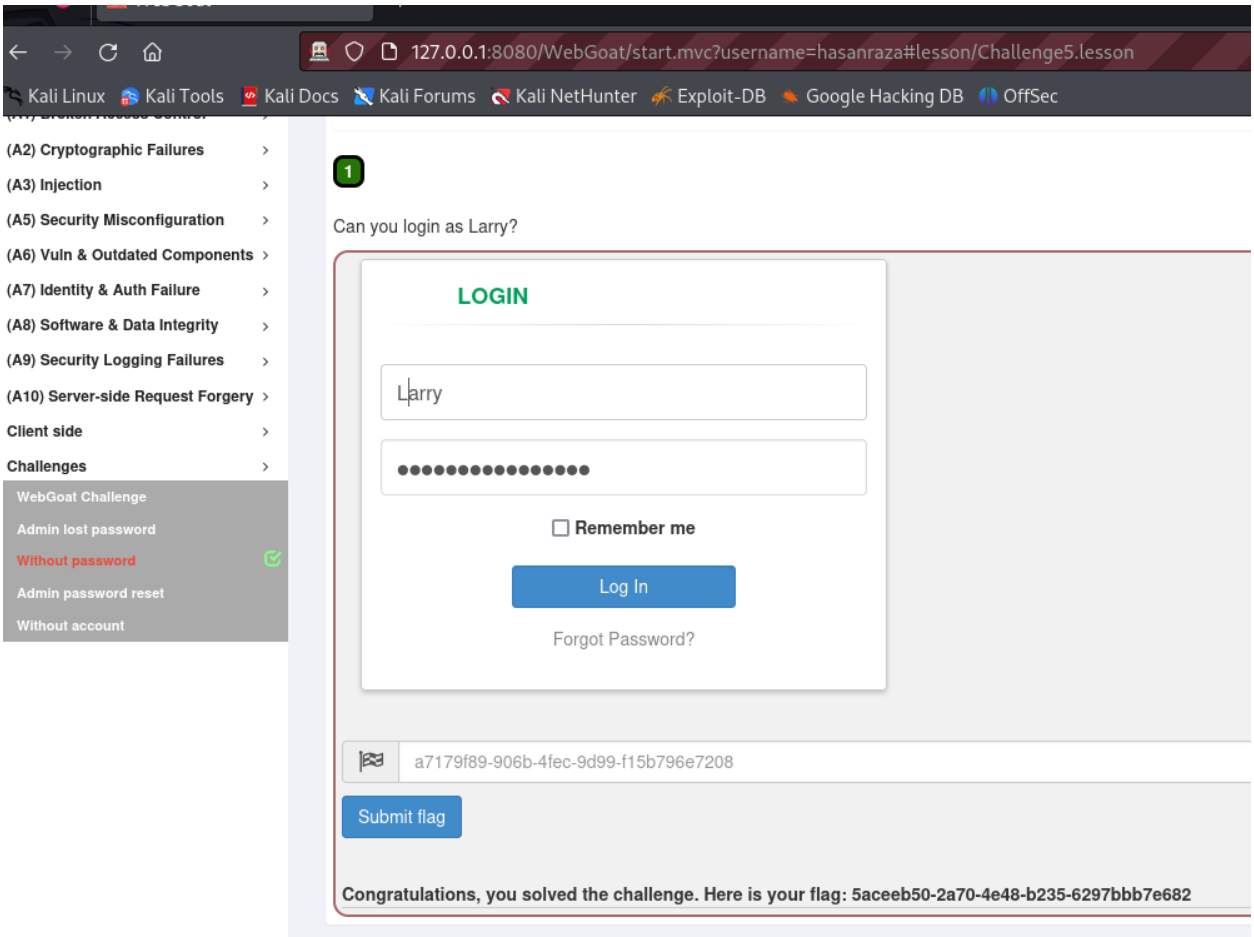
Testing for SQL Injection:

Tried entering admin' OR '1'='1

In both username and password fields



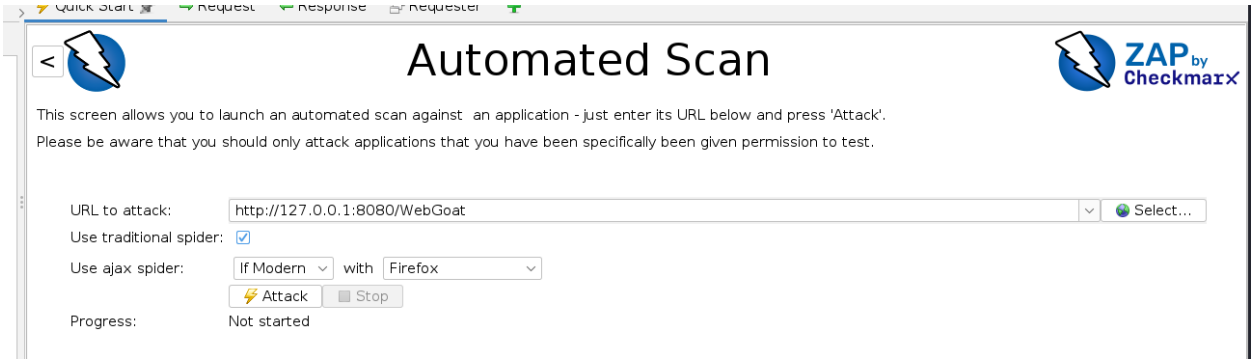
Then tried :
Larry as username and for password: larry' OR '1'='1



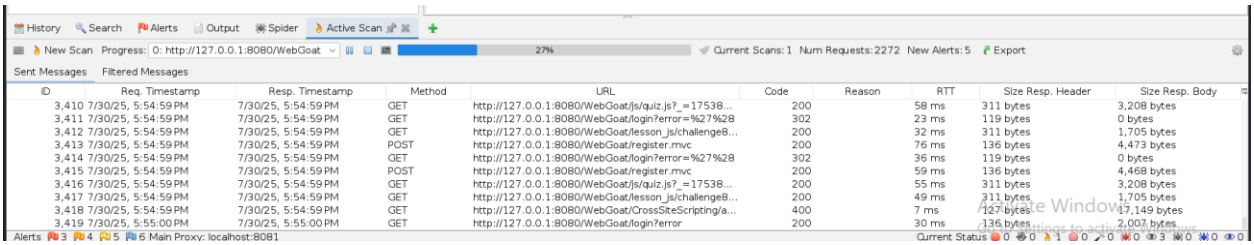
Vulnerability: SQL Injection

- **Location**: Login page of WebGoat
- **Input Used**:
  - **Username**: `Larry`
  - **Password**: `larry' OR '1'='1`
- **Result**: Logged in without correct password; received success flag.
- **Impact**: Unauthorized access to user accounts.
- **Recommendation**: Use parameterized queries (prepared statements) to prevent SQL code injection.

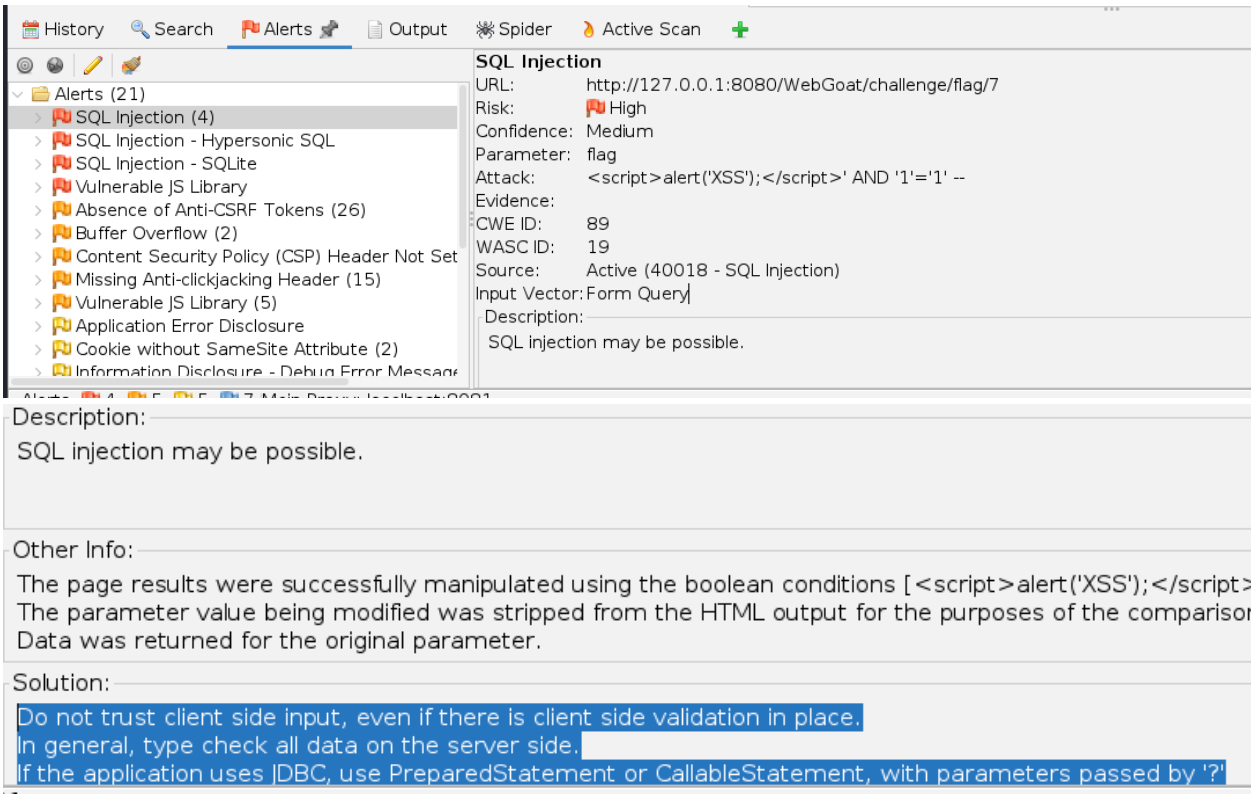
Automated Scan in OWASP ZAP:



Clicked ATTACK: scanning in process



After the successful scan these are the findings:



Note: The automated scan performed using OWASP ZAP identified multiple vulnerabilities across the WebGoat application, as visible in the Alerts tab (screenshot above). Unfortunately, due to a system freeze after the scan, I was only able to capture one screenshot before the Kali VM and VirtualBox became unresponsive.