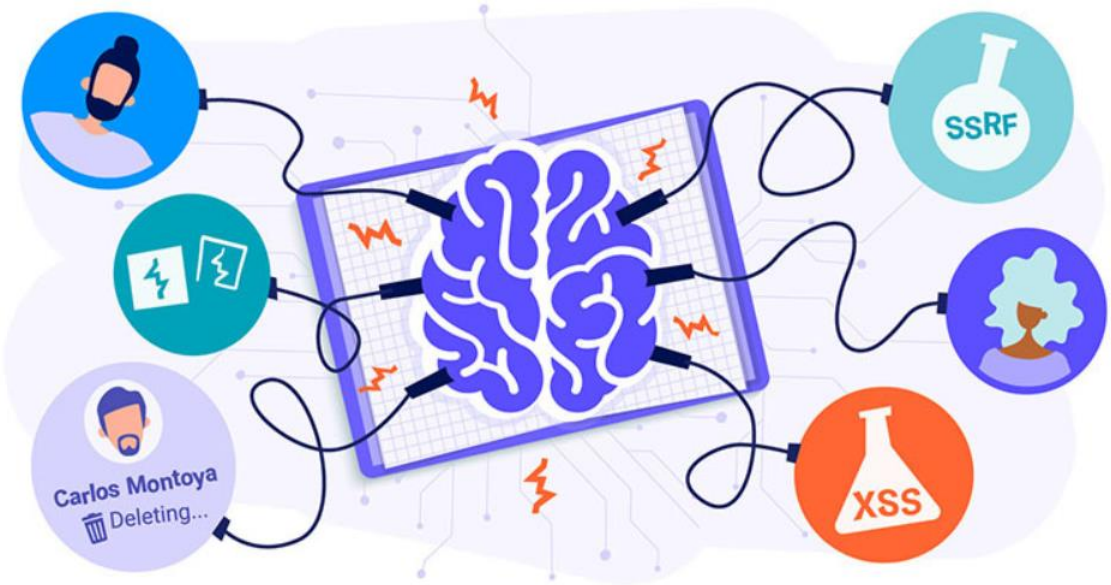
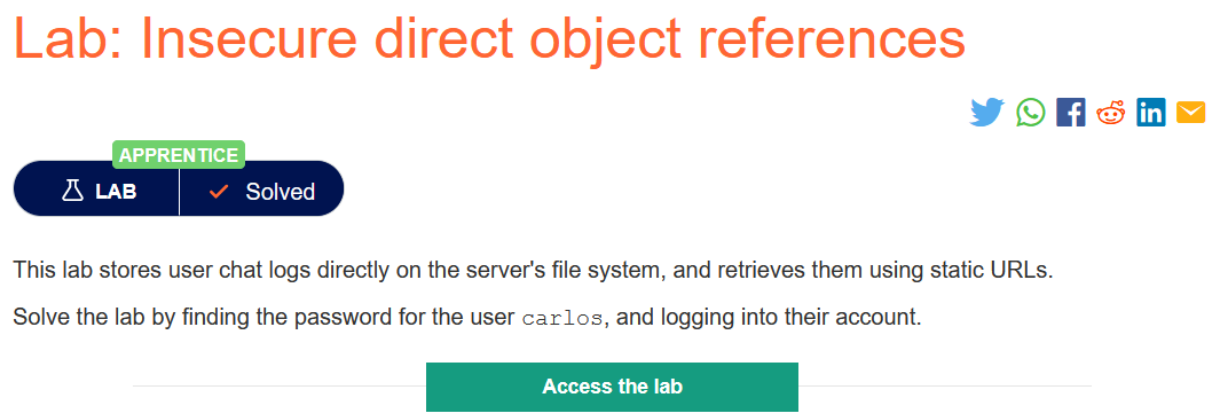


Author: Kharim Mchatta

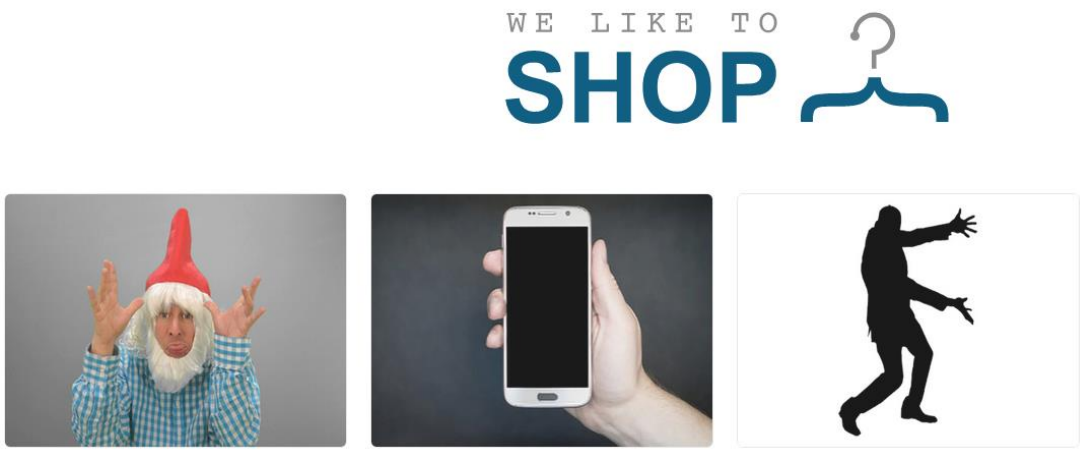
Topic: Access Control Vulnerability

Date: 7/6/2022

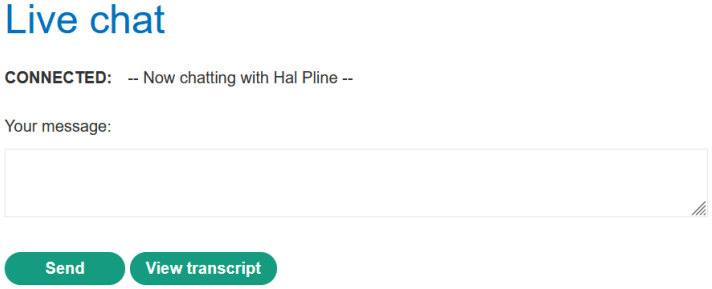
The first challenge that I had began solving was all about **Insecure direct object references** and here is how I solved it.



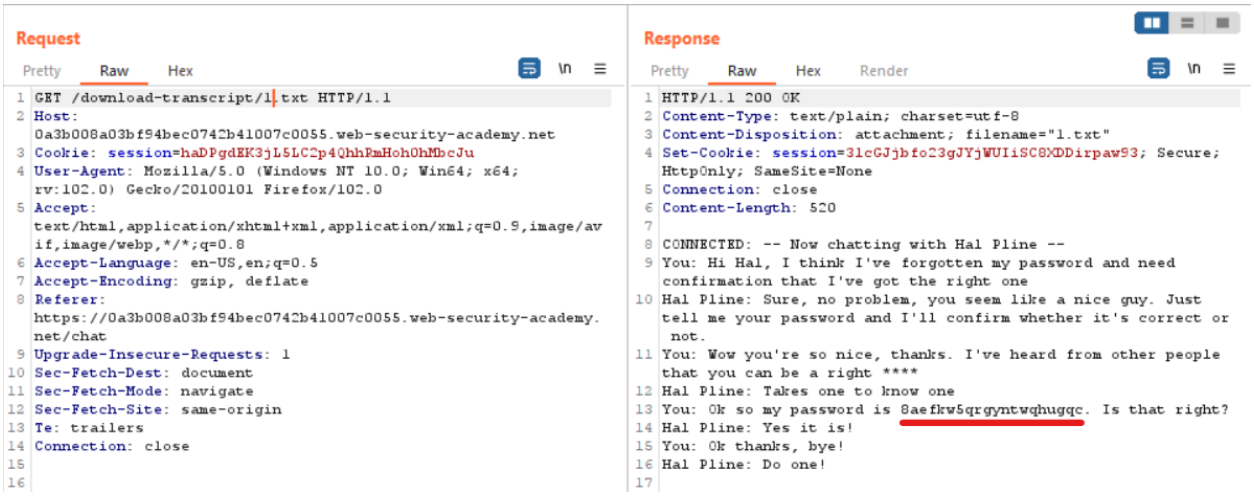
After reading the scenario I accessed the lab and found that it was an ecommerce website



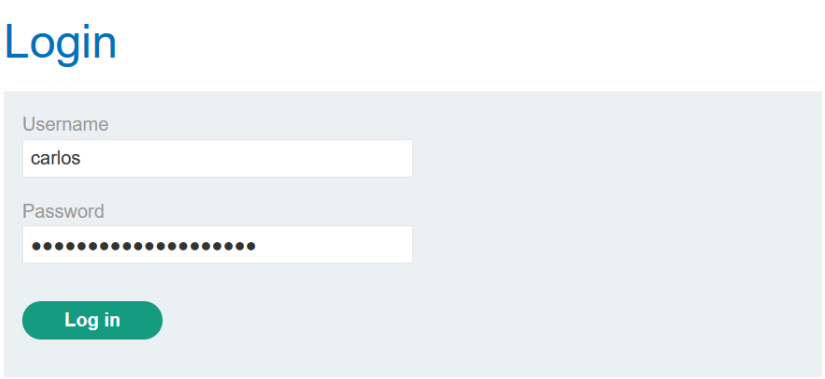
After going through the page, I saw there was a chat bot in the web application. I decided to interact with the bot, then after sending few requests I noticed that there was a view transcript button, clicking on it, it showed all the conversation we were having with the bot.



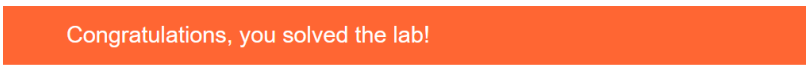
Next, I decided to send the view transcript traffic to burp suite. Looking at the request it had a file called 2.txt, then I started playing around the numbers, I started with 0.txt, it had nothing, I changed it to 1.txt and found some information about the interaction the previous user had with the bot and found the password for carlos.

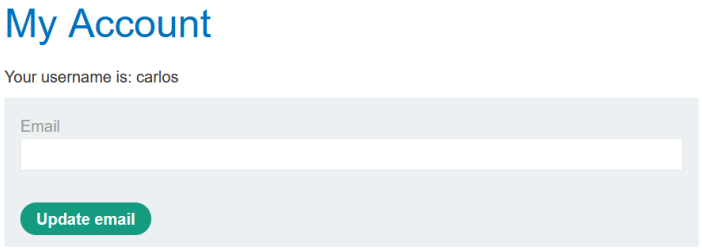


Inserting the credentials on the login page and we managed to successfully login the page

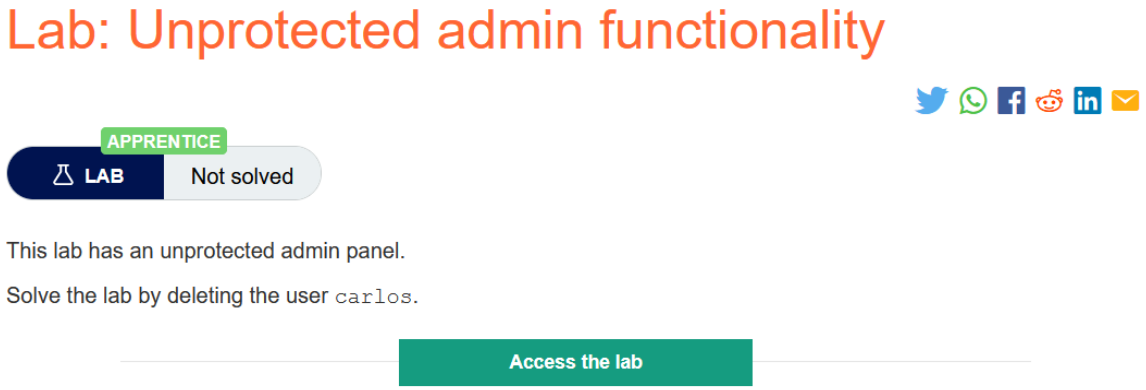


Logging in I saw a banner that said congratulations, you solved the lab.

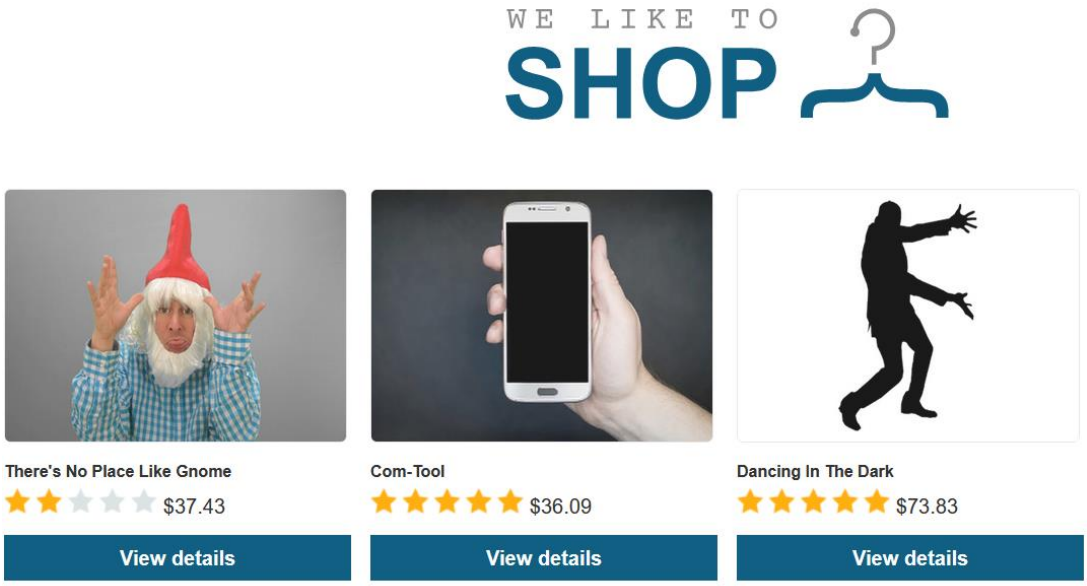




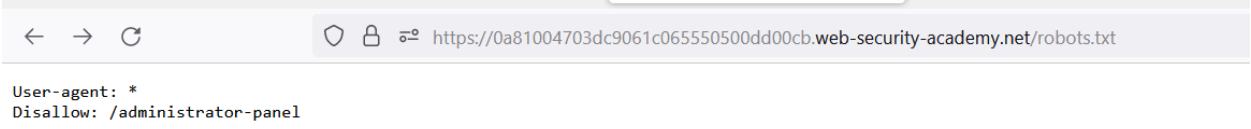
The second challenge to be solved was all about **Unprotected Admin Functionality** and here is how I solved it.



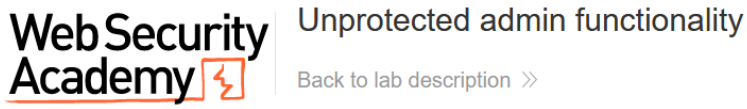
Opening the lab, it was a ecommerce website, going through the website there was nothing interesting of use.

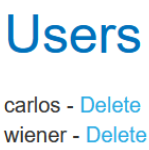


Then I check the robots.txt file and found the administrative login page path

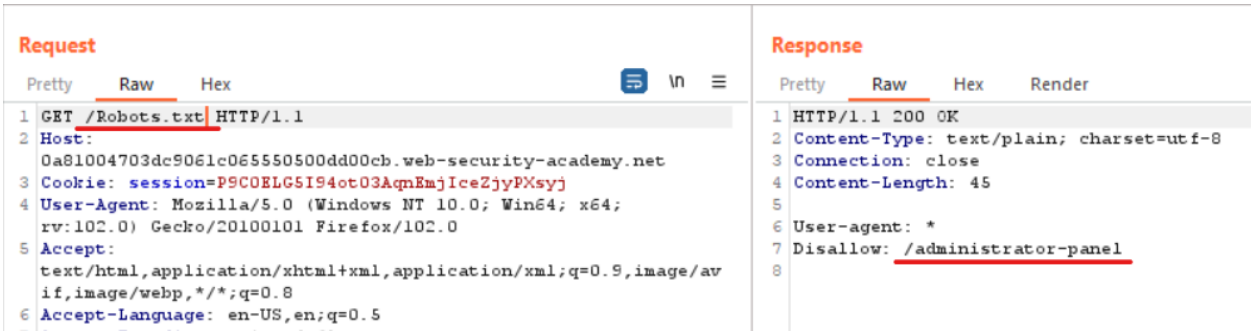


Accessing the administrative login page, I was able to delete the user carlos as the challenge demanded.

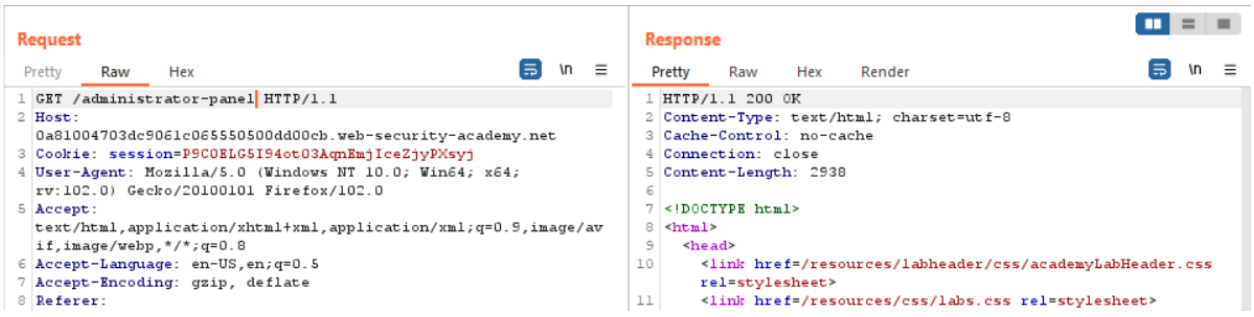




You can achieve the same goal with using burp suite on interpreter, you can send the request to burp and then add /robots.txt after the http method as shown on the pic below



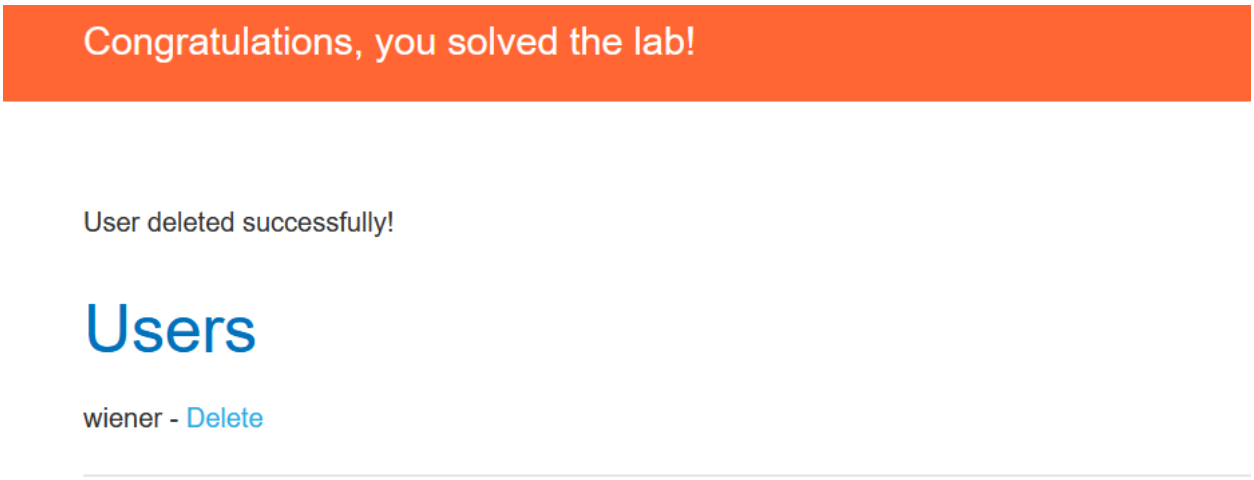
Then copy the administrative login url and paste it in the http method and then send the request



To delete the user just add the path **/administrator-panel/delete?username=carlos** and send the request back to the web application.

```
GET /administrator-panel/delete?username=carlos HTTP/1.1
Host:
0a81004703dc9061c065550500dd00cb.web-security-academy.net
Cookie: session=P9C0ELG5I94ot03AqnEmjIceZjyPXsyj
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
rv:102.0) Gecko/20100101 Firefox/102.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/av
if,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
```

Once the user is deleted is when the challenge is completed.



The third challenge to be solved was all about **Unprotected admin functionality with unpredictable URL** and here is how I solved it.

Lab: Unprotected admin functionality with unpredictable URL

APPRENTICE

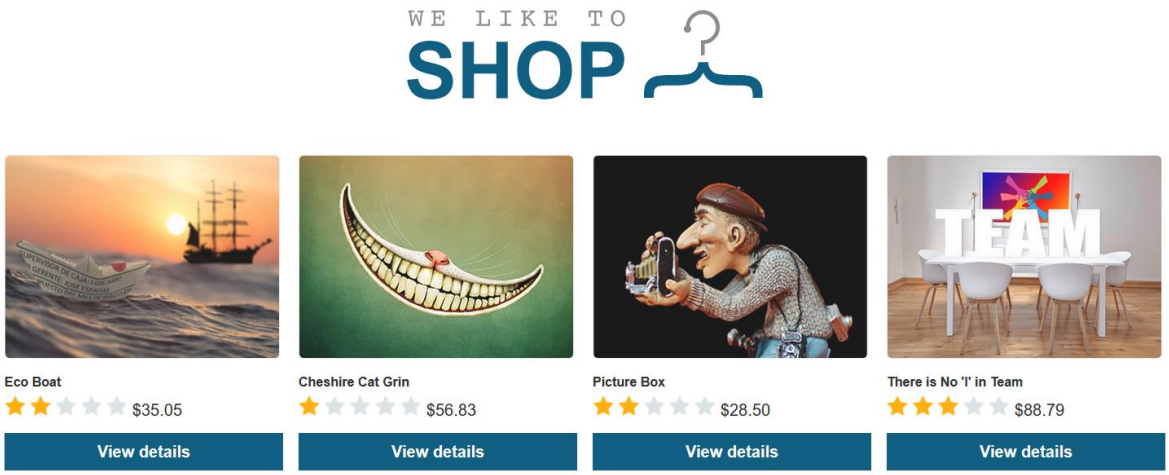
LAB Not solved

This lab has an unprotected admin panel. It's located at an unpredictable location, but the location is disclosed somewhere in the application.

Solve the lab by accessing the admin panel, and using it to delete the user `carlos`.

Access the lab

Accessing the lab, it opened up an ecommerce website and going through the page and there was nothing interesting, I tried accessing robots.txt but unfortunately it was blocked,

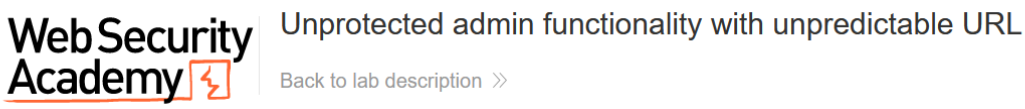


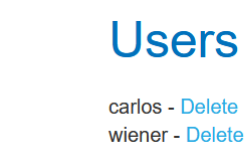
then I decided to try and examine the source code and saw this piece of interesting script.

```
                        <script>
var isAdmin = false;
if (isAdmin) {
   var topLinksTag = document.getElementsByClassName("top-links")[0];
   var adminPanelTag = document.createElement('a');
   adminPanelTag.setAttribute('href', '/admin-sy2lko');
   adminPanelTag.innerText = 'Admin panel';
   topLinksTag.append(adminPanelTag);
   var pTag = document.createElement('p');
   pTag.innerText = '|';
   topLinksTag.appendChild(pTag);
}
</script>
```

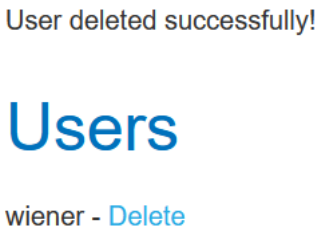
Carefully reading the script I saw the URL to the administrative page which was **/admin-sy2lko**

Pasting it into the URL and I got the administrative web page

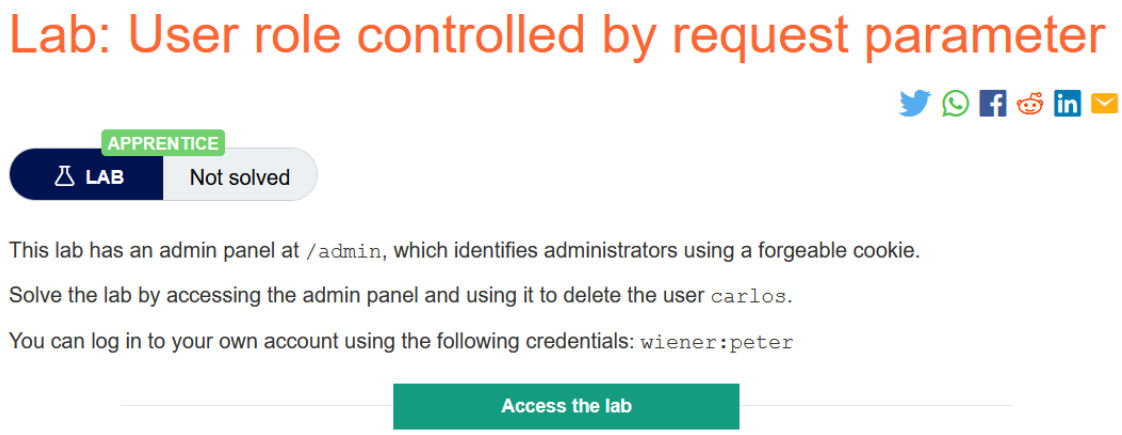




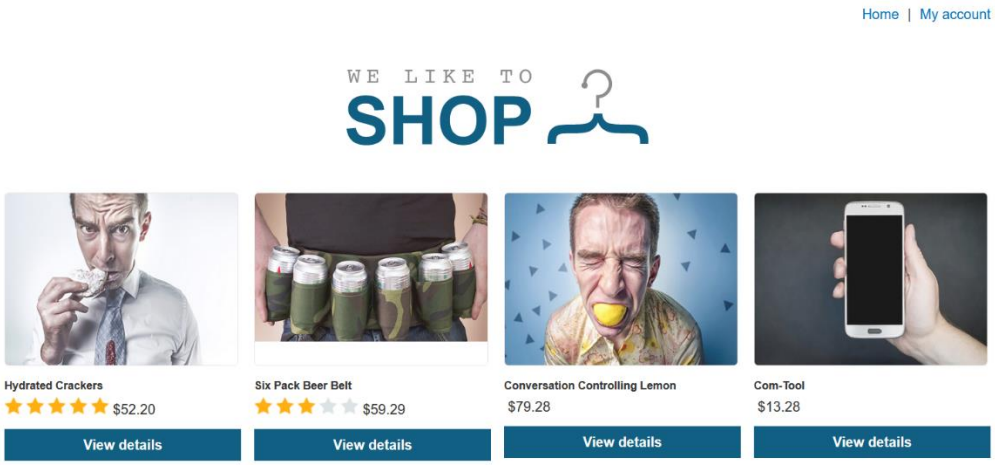
I deleted the user carlos and the challenge ended there.



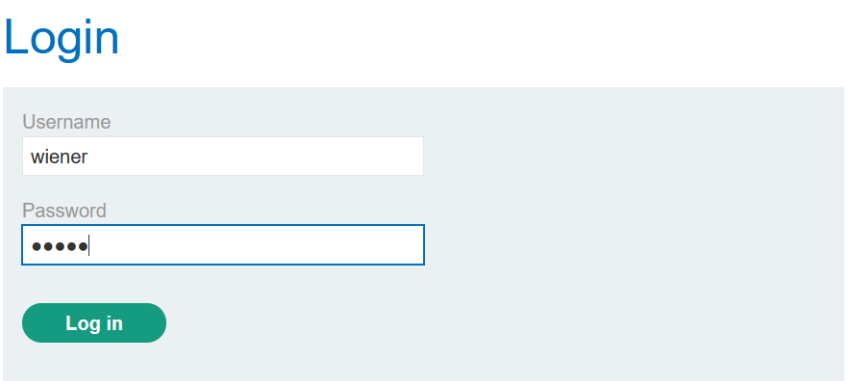
The fourth challenge to be solved was all about **User Role Controlled by Request Parameter** and here is how I solved it.



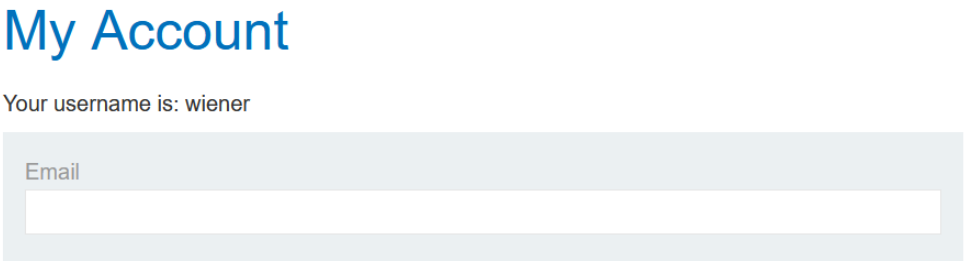
I accessed the lab and I was introduced to a ecommerce web application.



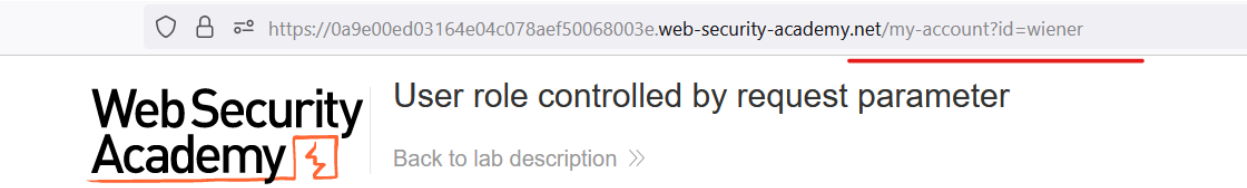
I went directly to the login page using the provided credentials



Accessing the login page there was nothing interesting



Looking at the URL and saw that on the URL it showed the name of the account that logged in, trying to change the parameter from wiener to carlos it didn't work

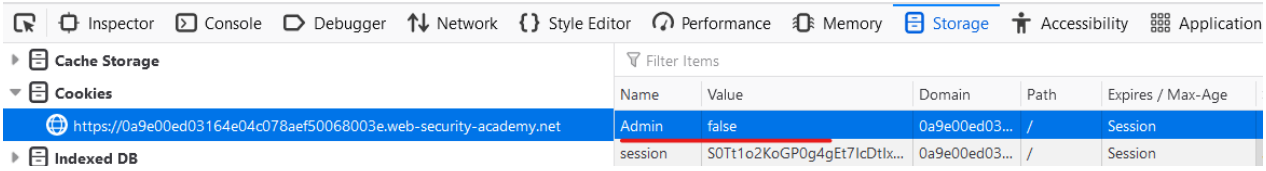


Then I had sent the request to burp to examine the traffic. I sawa on the cookie that the admin was set to false

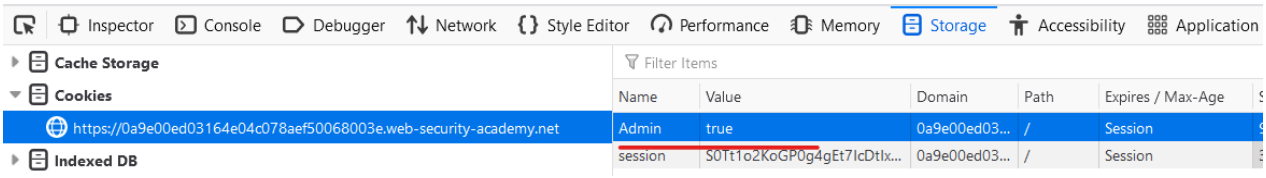
```
1 GET /my-account?id=wiener HTTP/1.1
2 Host:
  0a9e00ed03164e04c078aef50068003e.web-security-academy.net
3 Cookie: session=SOTtlo2KoGP0g4gEt7IcDtIxQbPC9Zl6; Admin=false
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
  rv:102.0) Gecko/20100101 Firefox/102.0
5 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/av
  if,image/webp,*/*;q=0.8
```

Then I changed from false to true and sent the traffic and it was a success

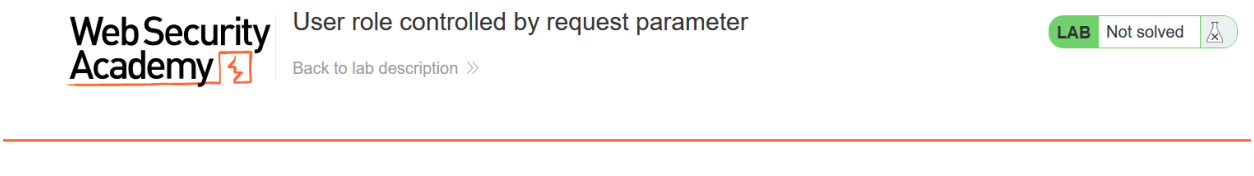
In ethical hacking there is no single way on doing things to achieve the same goal. Another simple method is to use the inspect element feature on our browsers

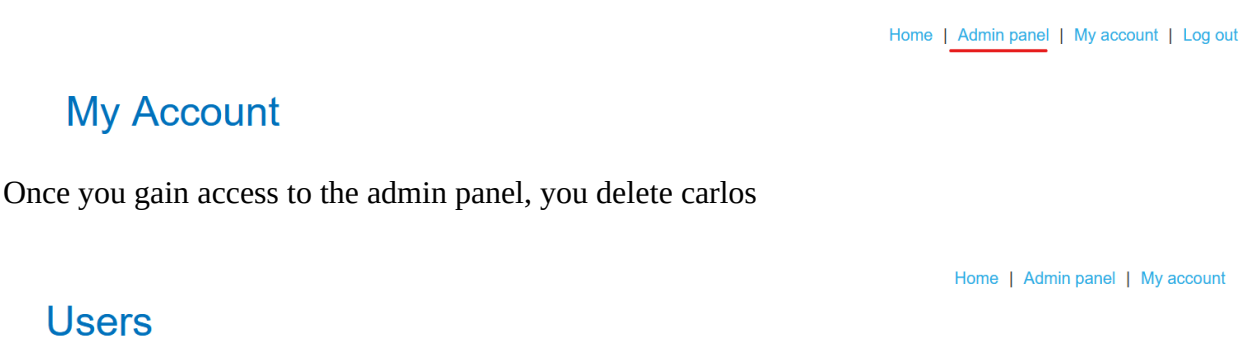


When you open it you can get to see the cookie then you can change the value from false to true



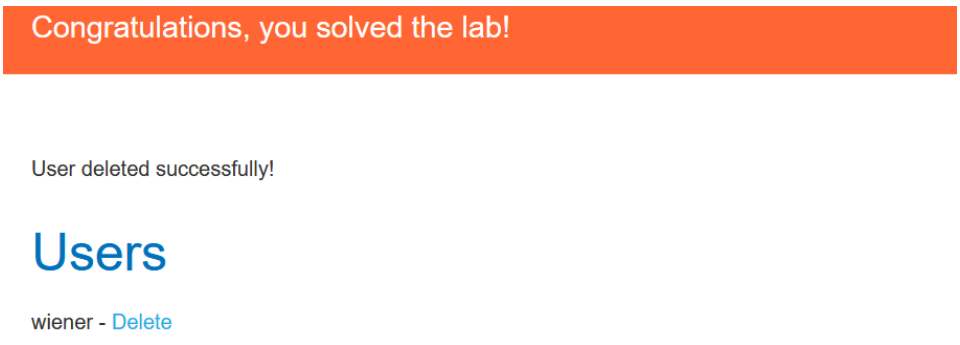
Once you send the request you can see that an admin panel tab that appears as shown below



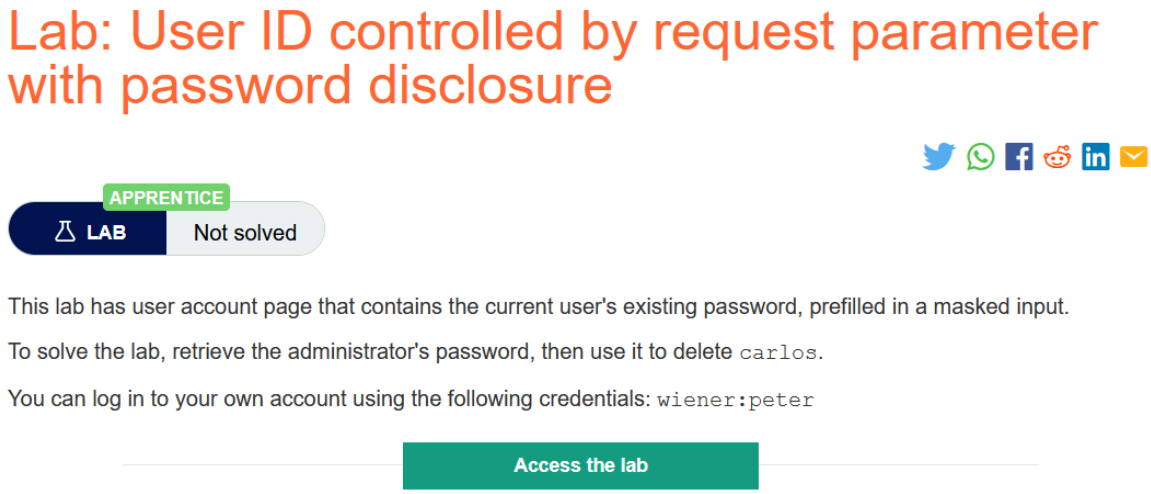


Once you gain access to the admin panel, you delete carlos

And the challenge ended there after deleting the account of carlos

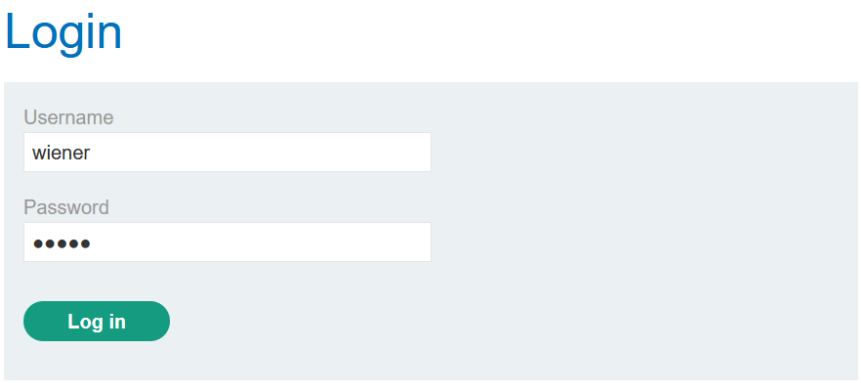


The fifth challenge to be solved was all about **User ID controlled by request parameter with password disclosure** and here is how I solved it.

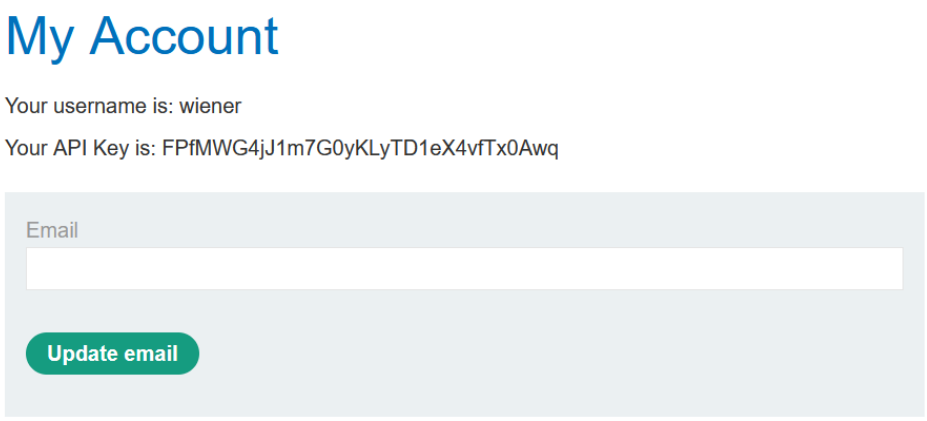


I accessed the lab and I was introduced to an ecommerce site as shown below

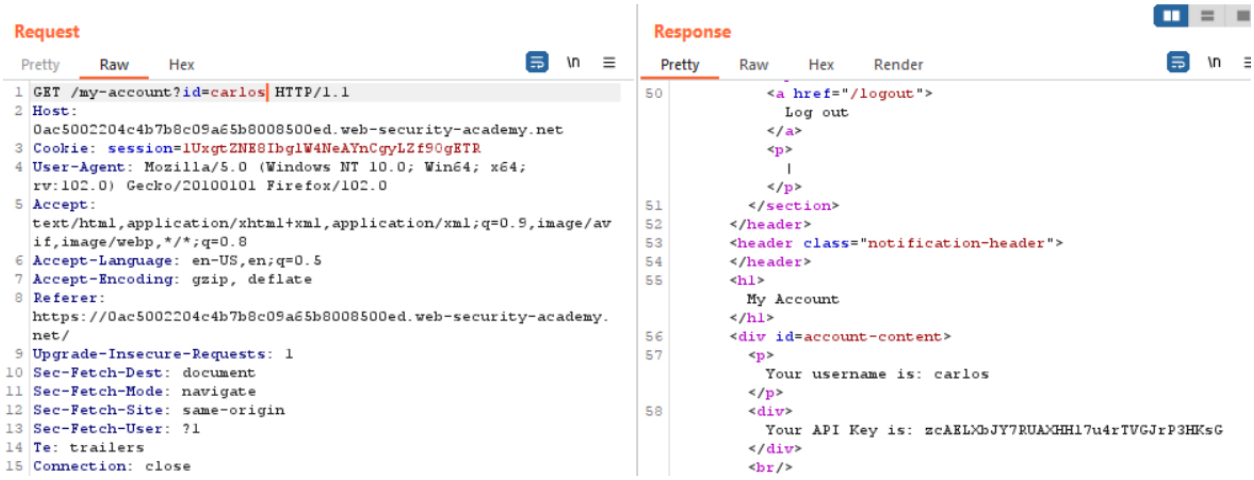
Then I went to the login page and logged in as wiener



Gaining access to the account I had found the API key of wiener but the task was to get the API key of carlos



I then sent the request to burp and changed the id name in the URL from weiner to carlos and sent the request back to the web application



and I managed to get the API key of carlos and the challenge ended there

The sixth challenge to be solved was all about **User ID Controlled by request parameter, with unpredictable user ID** and here is how I solved it.

Lab: User ID controlled by request parameter, with unpredictable user IDs

APPRENTICE

LAB Not solved

This lab has a horizontal privilege escalation vulnerability on the user account page, but identifies users with GUIDs.

To solve the lab, find the GUID for `carlos`, then submit his API key as the solution.

You can log in to your own account using the following credentials: `wiener:peter`

Access the lab

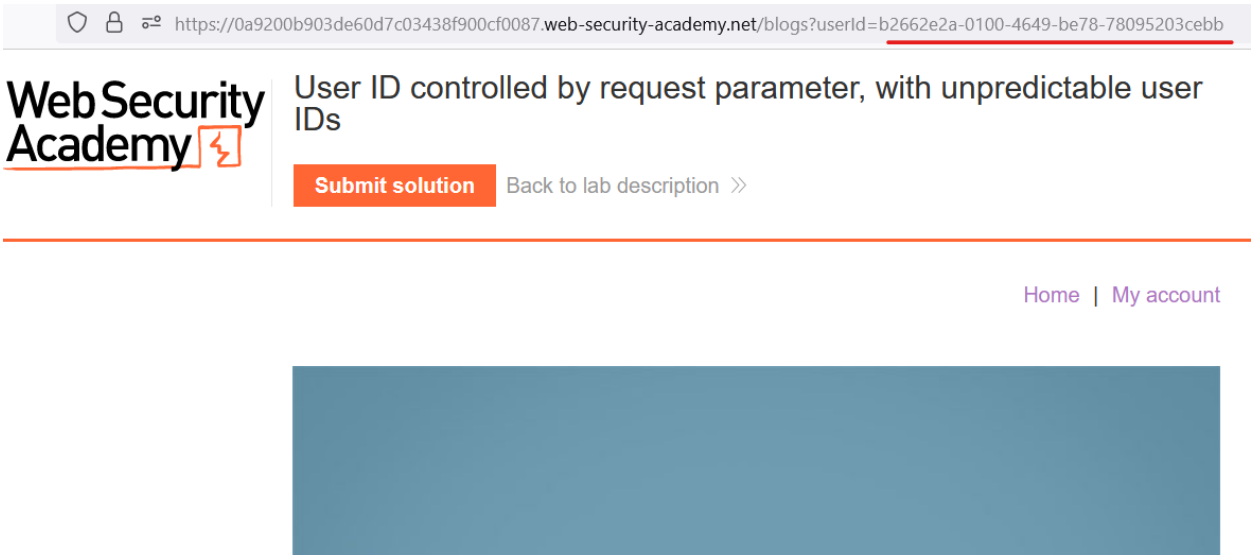
I accessed the lab and the web site appeared, initially I logged in and had seen that GUID was being used as stated In the scenario, now we had to find the GUID of carlos in order to privilege escalate to his account. Exploring the website I had found a blog written by carlos

Home | My account

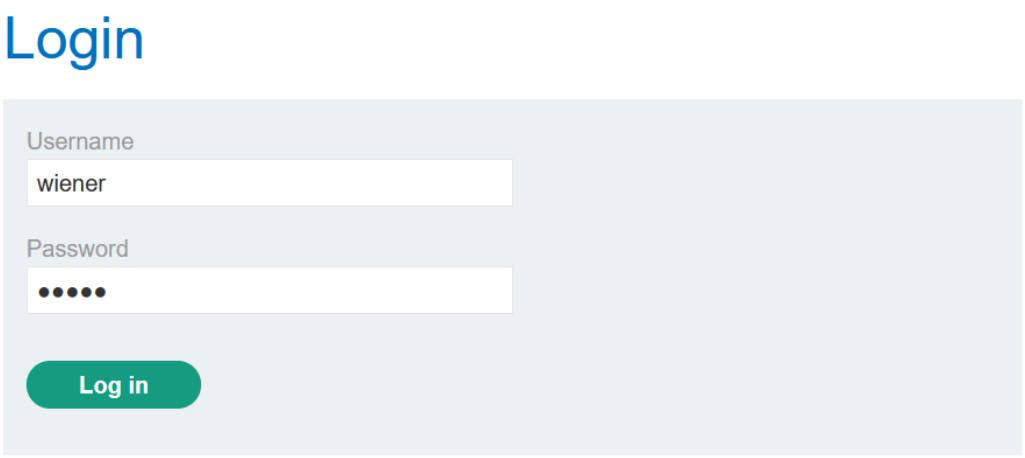
New Year - New Friends

carlos | 10 June 2022

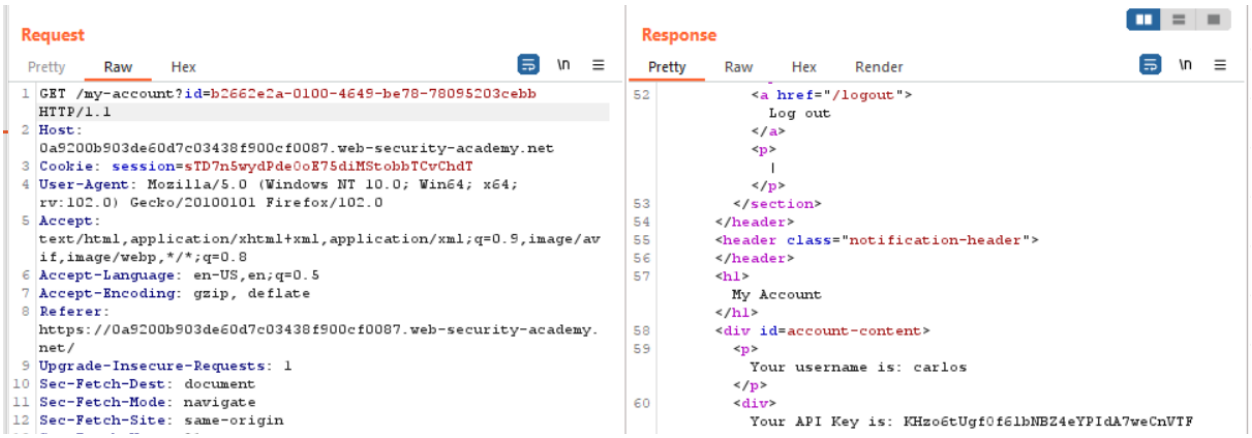
Checking the url I had seen the GUID of carlos



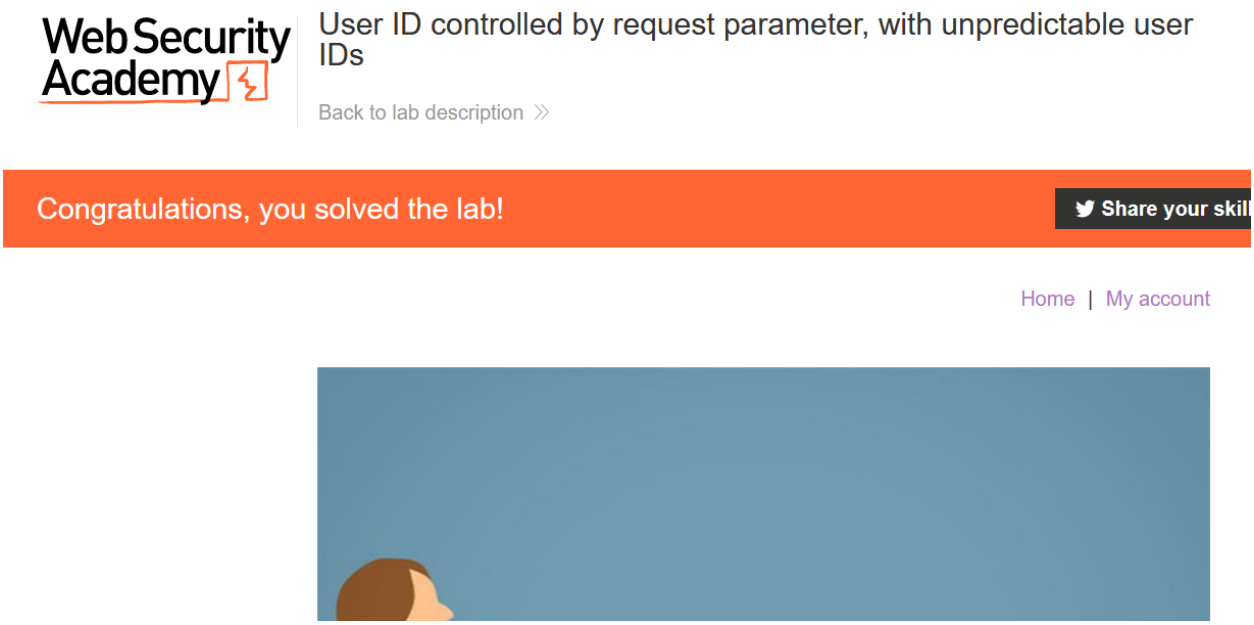
I went back and logged in the account of wiener



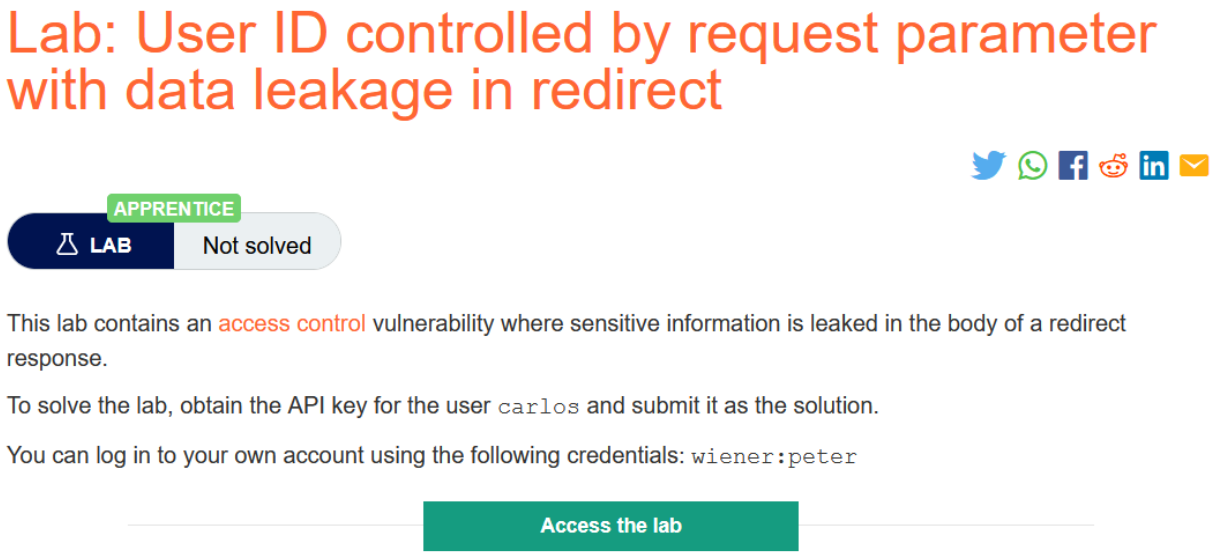
Then I had sent the request to burp and then I pasted the GUID of carlos that I had found from the blog and replaced it with the one of weiner and sent the request



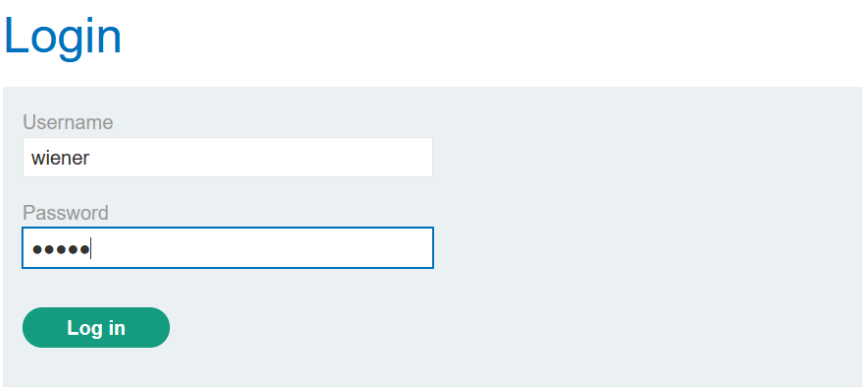
And I managed to get API key of Carlos and the challenge had finished



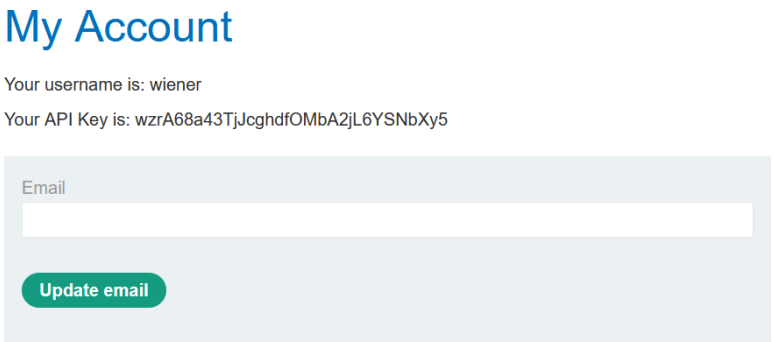
The seventh challenge to be solved was all about **User ID controlled by request parameter with data leakage in redirect** and here is how I solved it.



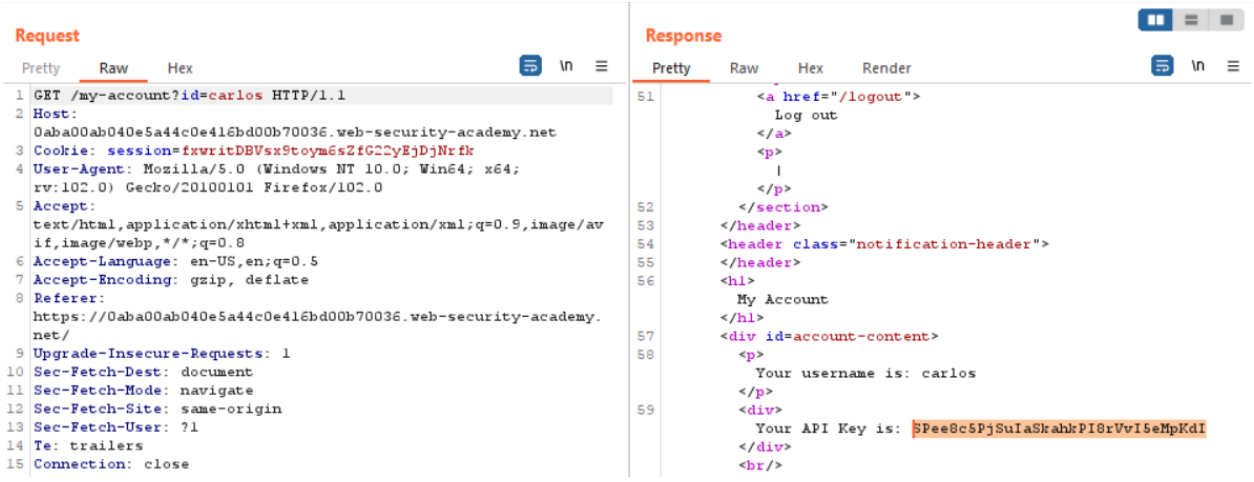
I accessed the lab and there was an ecommerce site, then I clicked on the login page



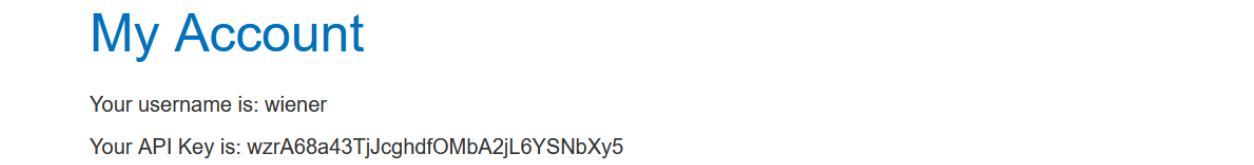
Once logged in, I had send the traffic to burpsuite



Once intercepted the traffic I changed the id from wiener to carlos and sent the traffic to the web application



And I had obtained the API key of carlos and submitted it and the challenge ended there.



The eighth challenge to be solved was all about **User ID controlled by request parameter with password disclosure** and here is how I solved it.

Lab: User ID controlled by request parameter with password disclosure

APPRENTICE

LAB Not solved

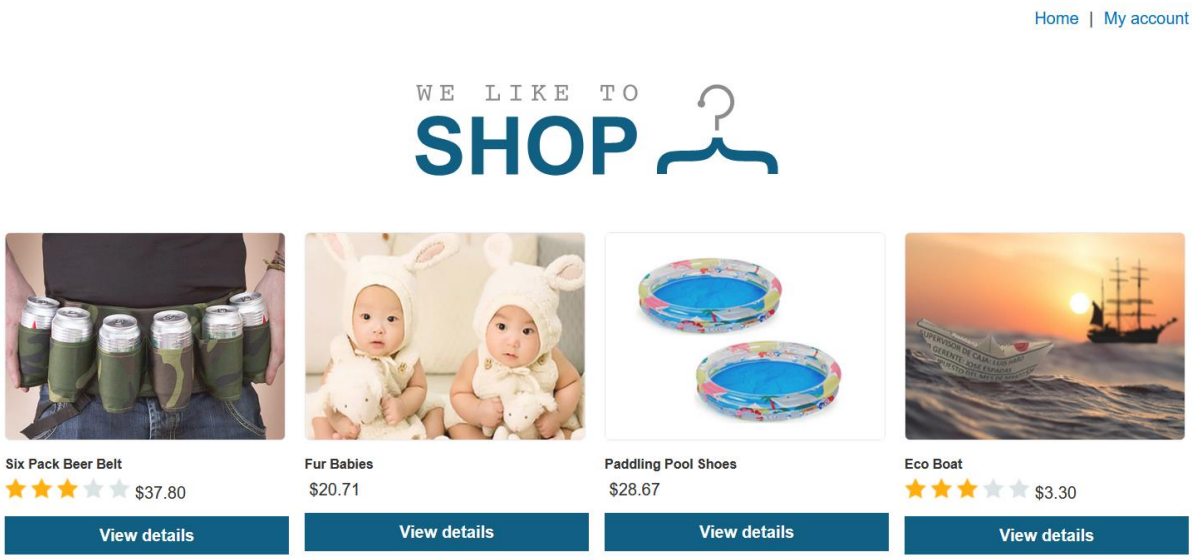
This lab has user account page that contains the current user's existing password, prefilled in a masked input.

To solve the lab, retrieve the administrator's password, then use it to delete `carlos`.

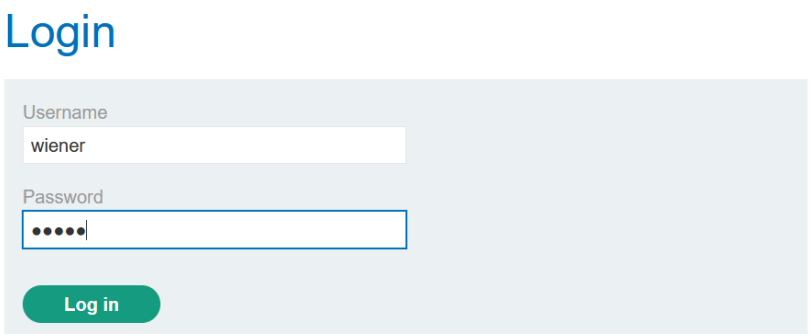
You can log in to your own account using the following credentials: `wiener:peter`

Access the lab

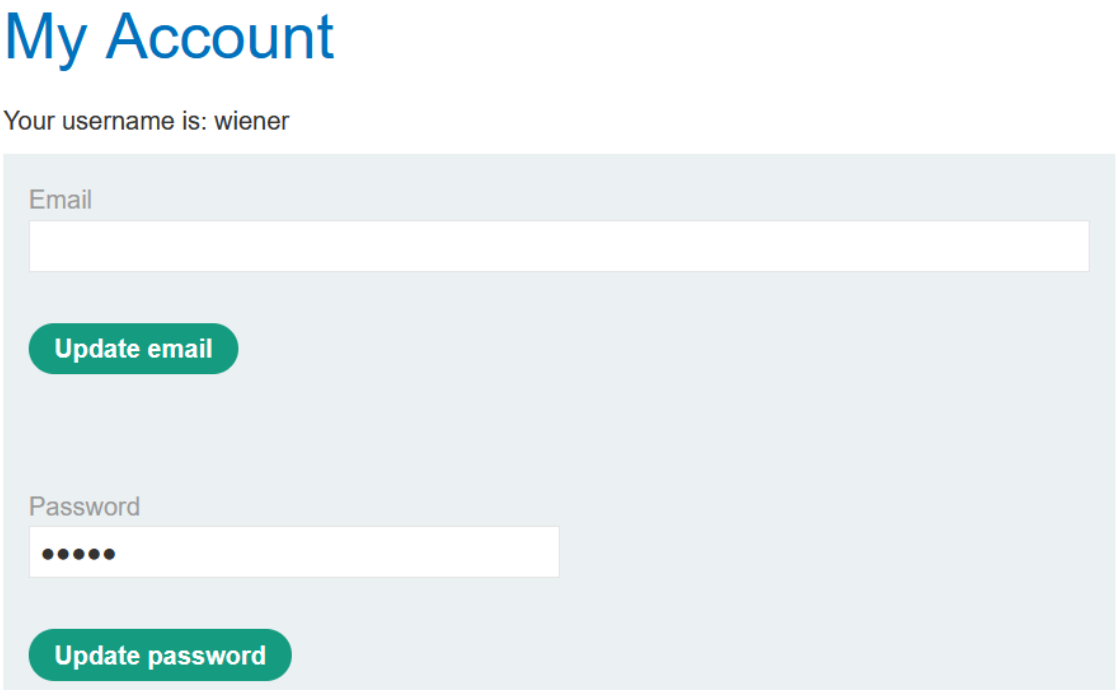
I accessed the lab and I was presented with an ecommerce website, I clicked on my account so that I could go and login



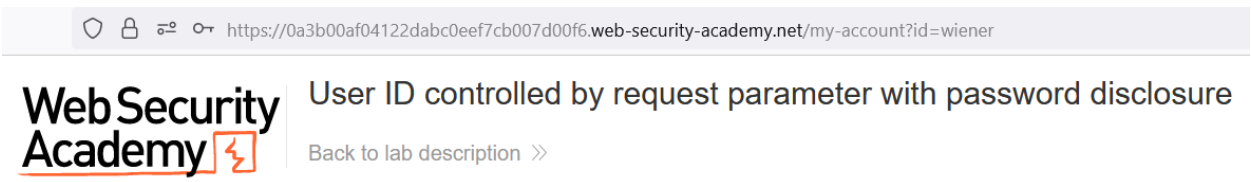
I entered the login credentials and logged in the website.



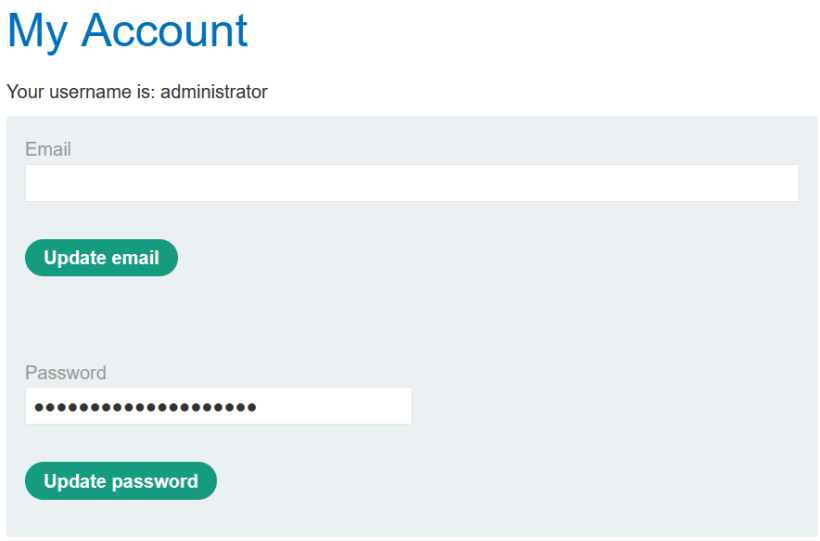
Once successfully logged in, on my account page I had seen that the password of wiener was on the password field for updating the password, but it was hidden.



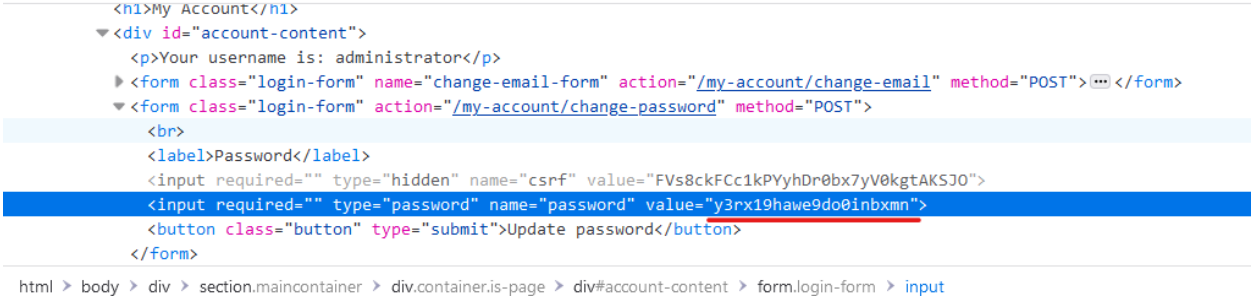
Looking at the URL we can see that there is an ID parameter and its value which is the account name, so I changed it from wiener to administrator



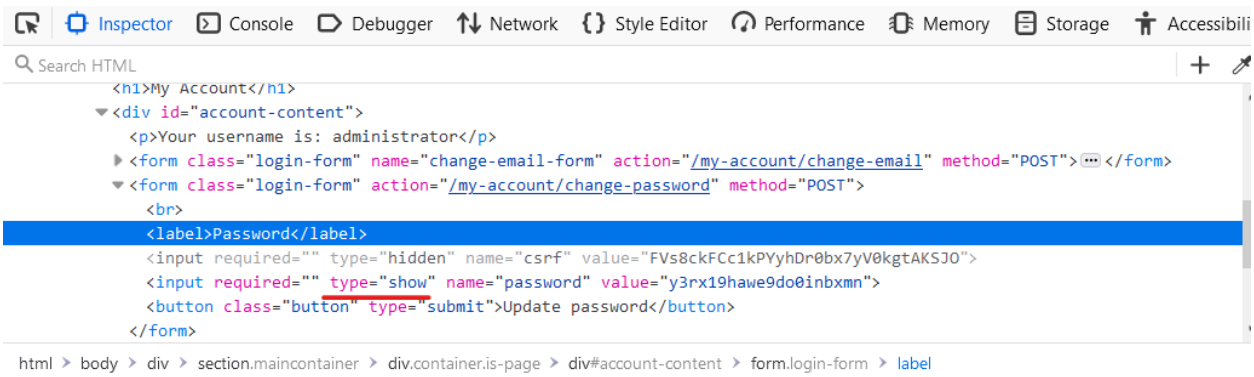
I noticed that the password of the administrator appeared but it was hidden as shown below



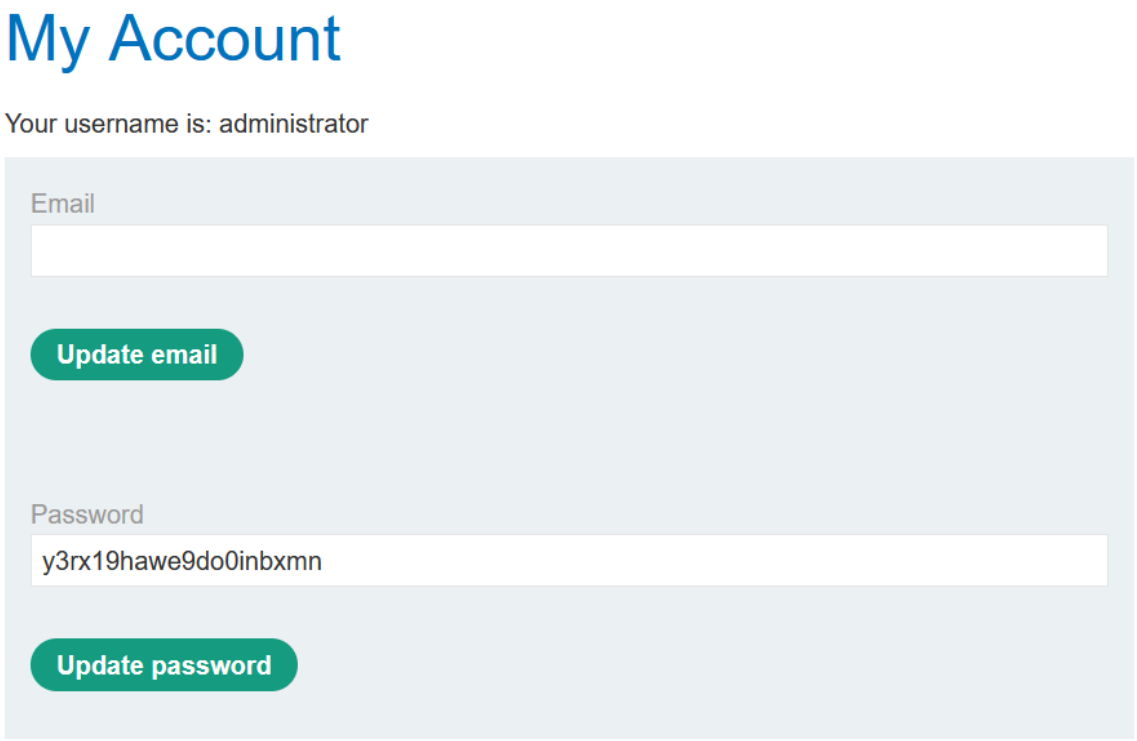
So I inspected element as show below and we could see the password of the administrator



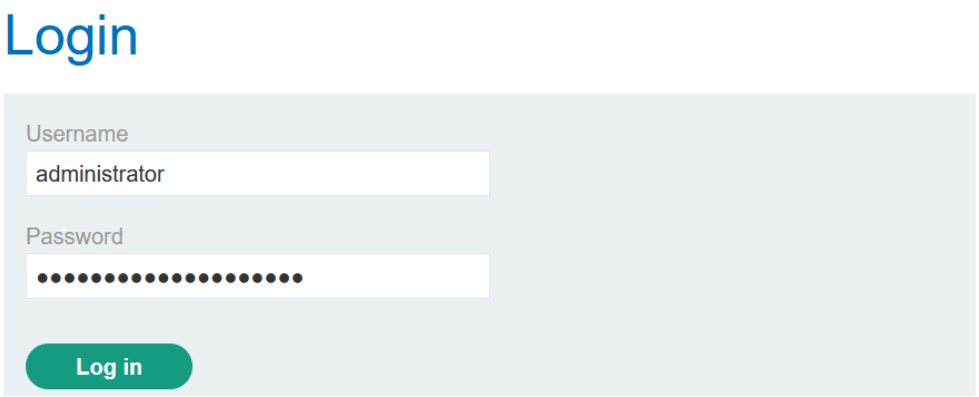
I added a parameter of type with its value show to reveal the password of the administrator as shown below



And I had obtained the password of the administrator

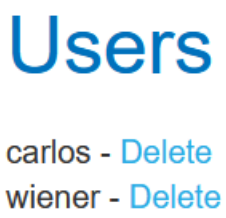


I used the credentials to login in the administrator's account

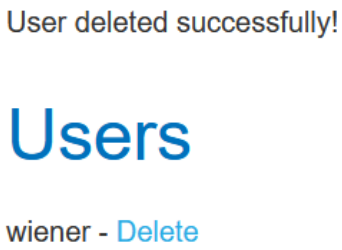


I then clicked on the admin panel

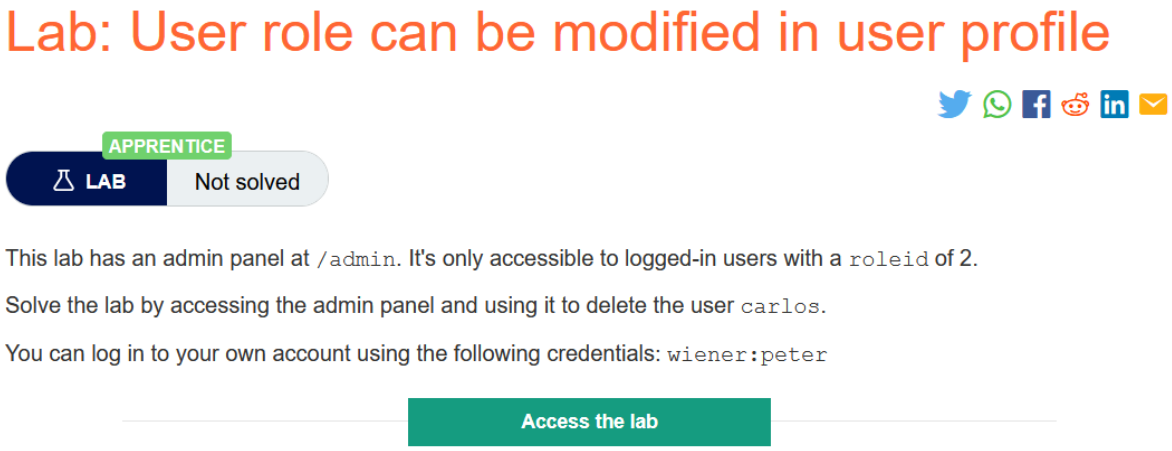
And accessed the admin panel



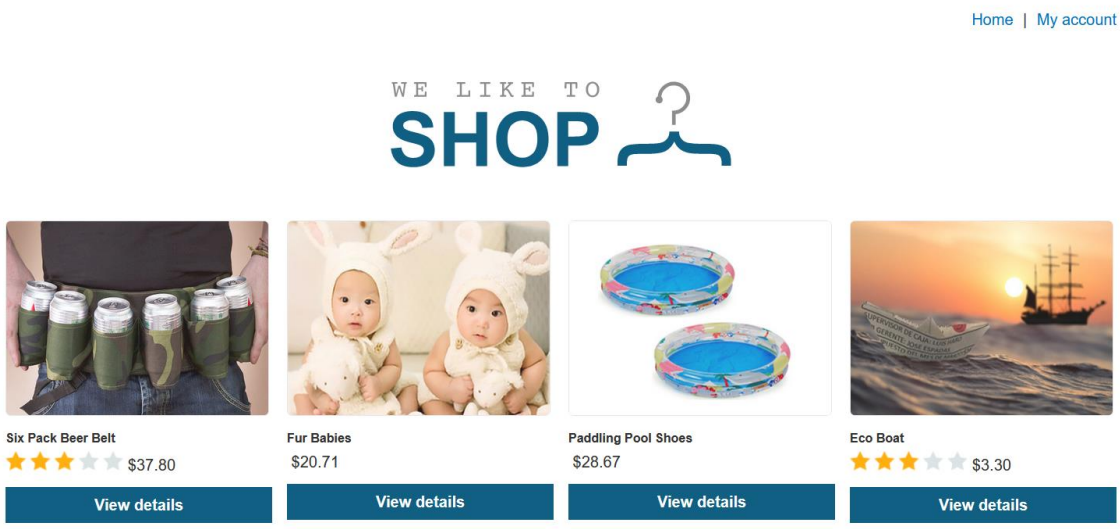
Then I deleted the user carlos and the challenged ended there.



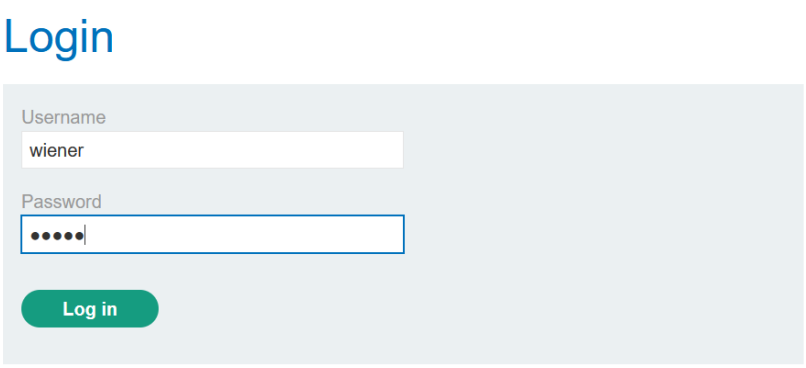
The nineth challenge to be solved was all about **User role can be modified in user profile** and here is how I solved it.



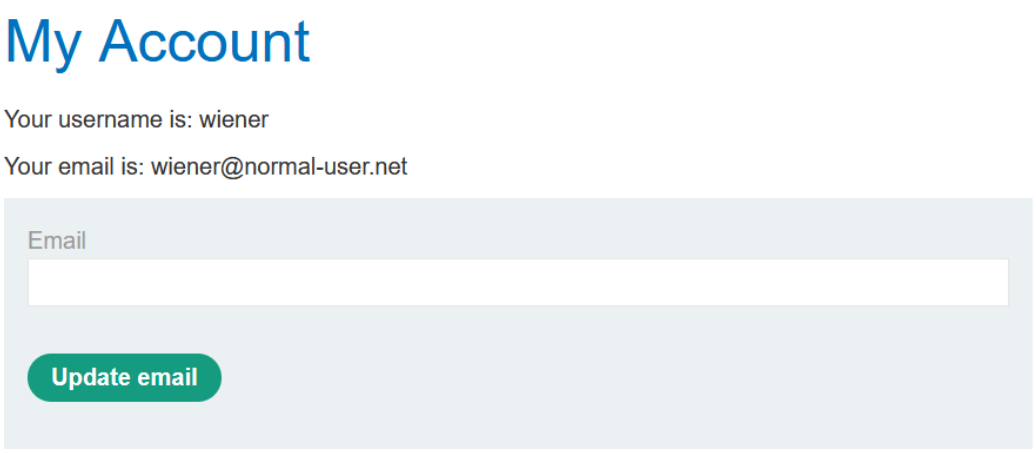
I accessed the lab and a web page was loaded and went to my account



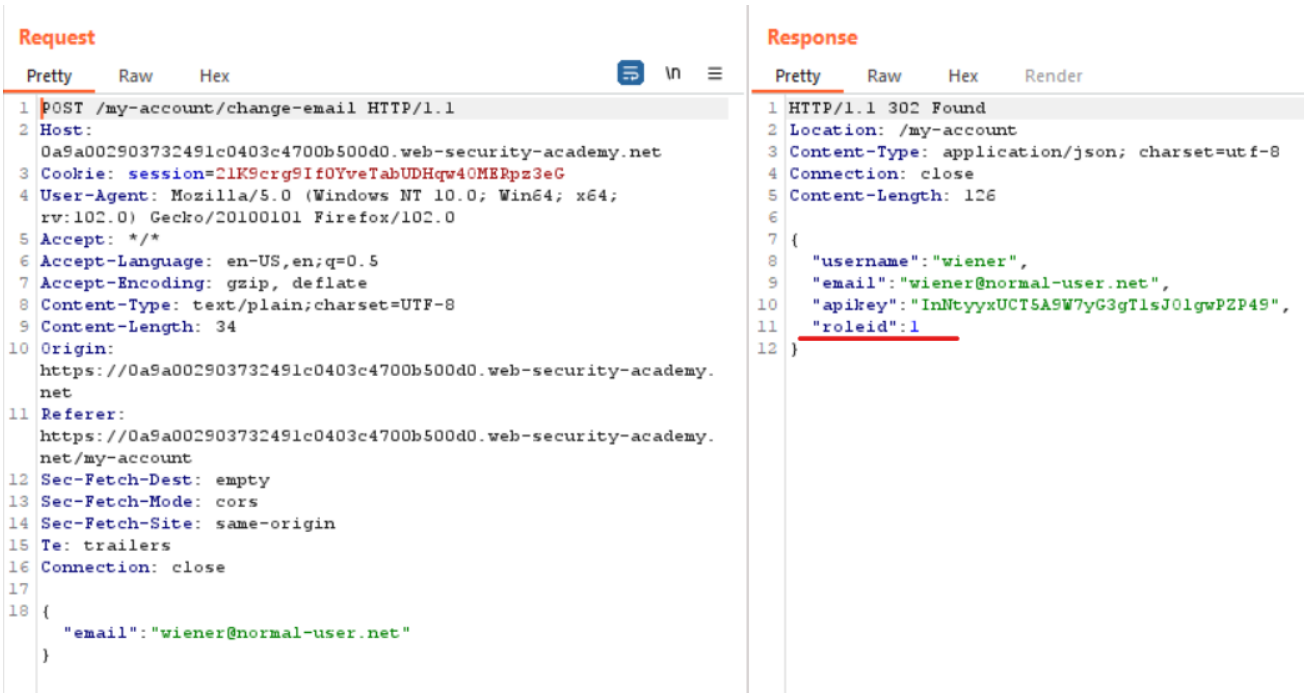
Went and inserted the credentials and logged in



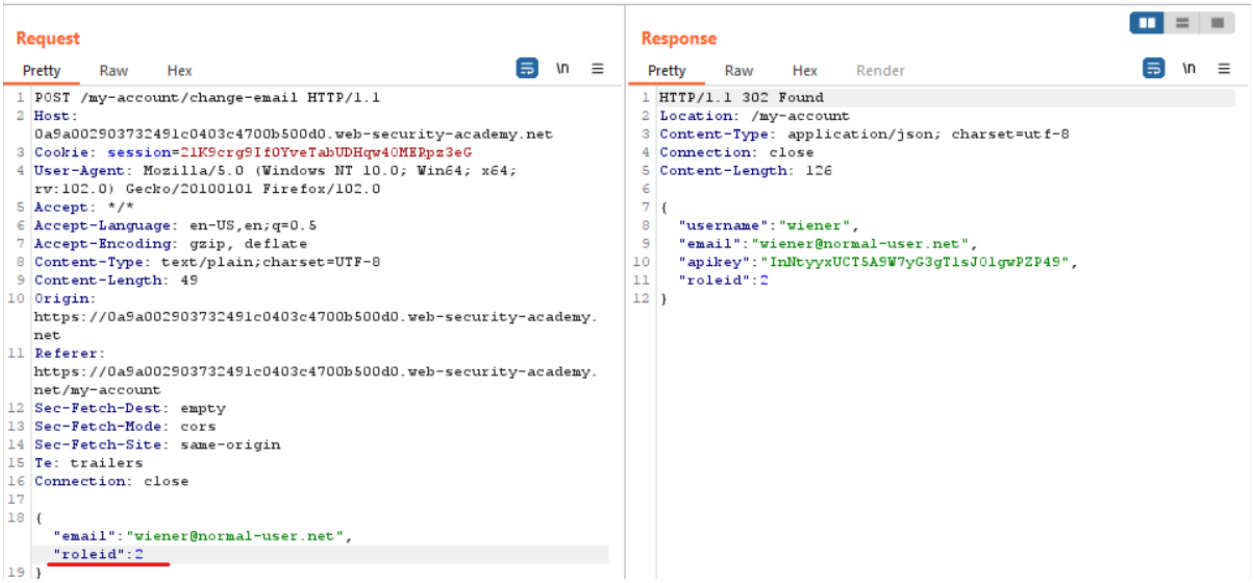
Login in there was a section of updating your email address, I tested the functionality and when it updated the email I showed up at the bottom of your username as shown below.



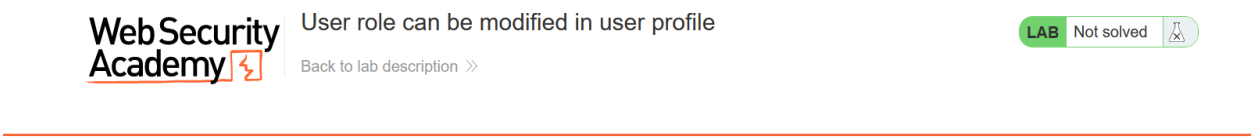
I had sent the request to burp suite and had seen that whenever you send the request you will be given the role id number of 2



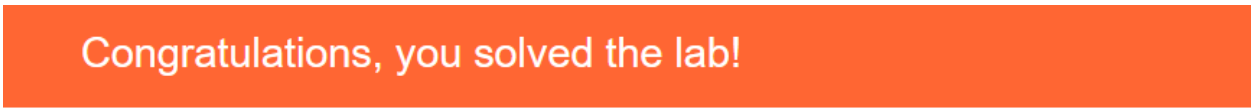
I went to the request and added the role id to 2 and sent the request and it went successfully



Looking at the dashboard I saw the admin panel tab appear, I clicked on it and gained access to the admin panel



Then I deleted the user carlos and the challenge was over.



The tenth challenge to be solved was all about **URL-based access control can be circumvented** and here is how I solved it.

Lab: URL-based access control can be circumvented

PRACTITIONER

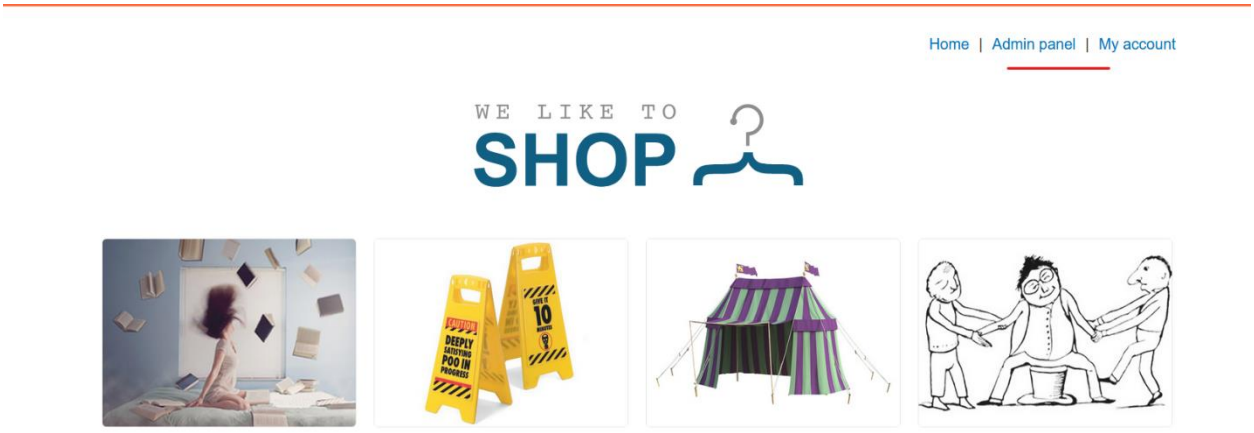
LAB Not solved

This website has an unauthenticated admin panel at `/admin`, but a front-end system has been configured to block external access to that path. However, the back-end application is built on a framework that supports the `X-Original-URL` header.

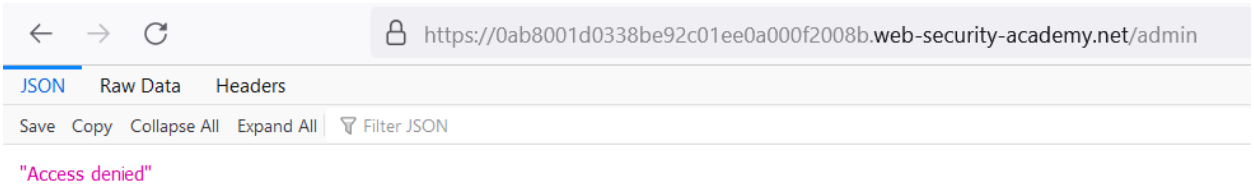
To solve the lab, access the admin panel and delete the user `carlos`.

Access the lab

This was an interesting challenge, I accessed the lab and an ecommerce appeared, straight away I noticed there was an admin panel tab on the home page.



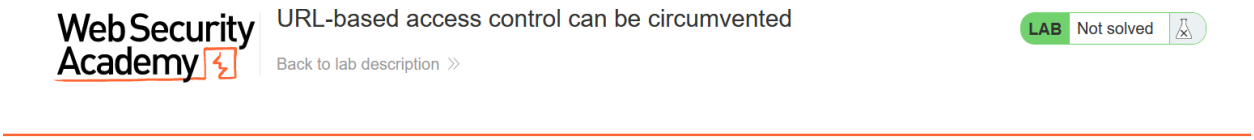
Clicking on the admin panel tab I was not able to access the page



Some application frameworks support various non-standard HTTP headers that can be used to override the URL in the original request, the header **X-Forwarded-URL** and **X-Rewrite-URL** this can be used to access restricted areas in the web application.

If a web site uses rigorous front-end controls to restrict access based on URL, but the application allows the URL to be overridden via a request header, then it might be possible to bypass the access controls using the request **X-Forwarded-URL** and **X-Rewrite-URL**.

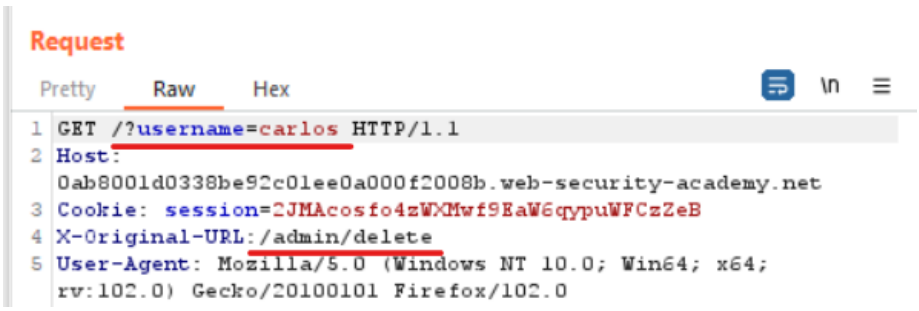
So I added the header **X-Forwarded-URL** and sent the request to the web application and I managed to gain access to the admin panel.



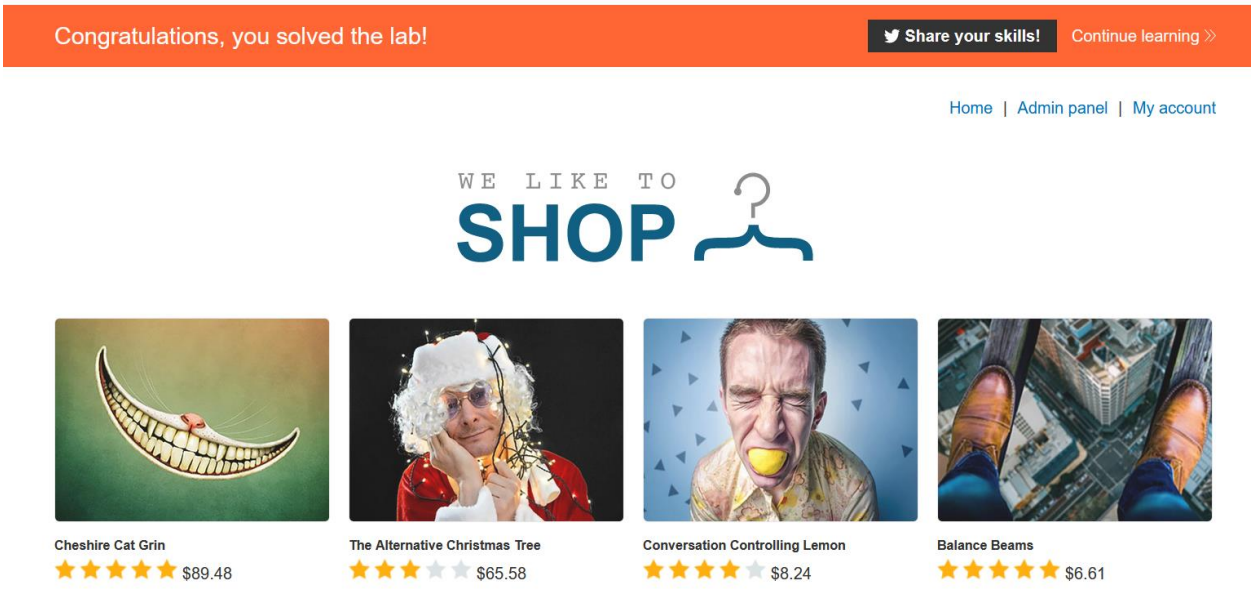
I then clicked on delete carlos and sent the traffic to burp, on the original request was

GET /admin/delete/?username=carlos

I edited the request slightly as show below to make the request bypass the controls



Once the request was sent, it was successful, and the challenge ended there



These challenges had taught me several things and some of them include

1. IDORS are not complicated to exploit
2. robots.txt can contain useful information
3. Access controls can be bypassed
4. check the source code can contain hard coded messages, comments, or even useful piece of scripts to progress with your attack
5. Cookies can be tampered with to give you privileged access
6. id's can be tampered with