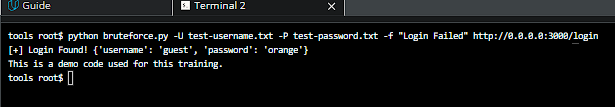
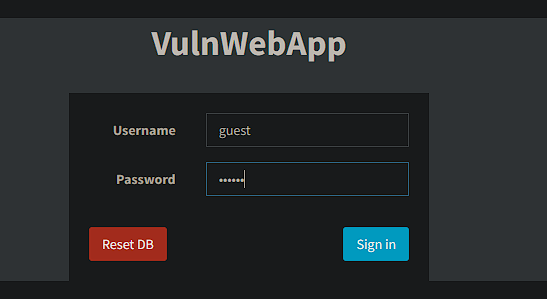
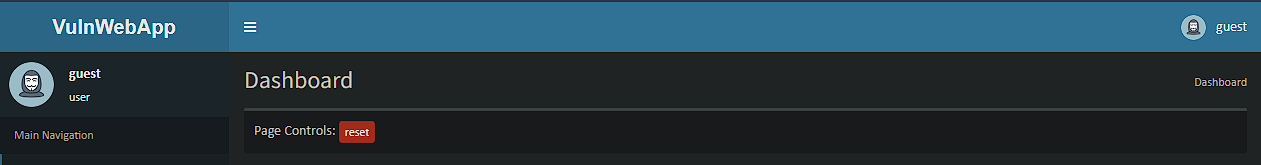
After Breaking into the app There’s a login form to continue   
  
first Vulnerability is

1. **Broken AUTH:**

I used the “Bruteforce.py” script against the login form





I was able to login as “guest” 

Checking the profile page   
there’s input field to send message to admin

I was able to find Stored-XSS vulnerability

1. **XSS-Vulnerability**

Graphical user interface, rectangle

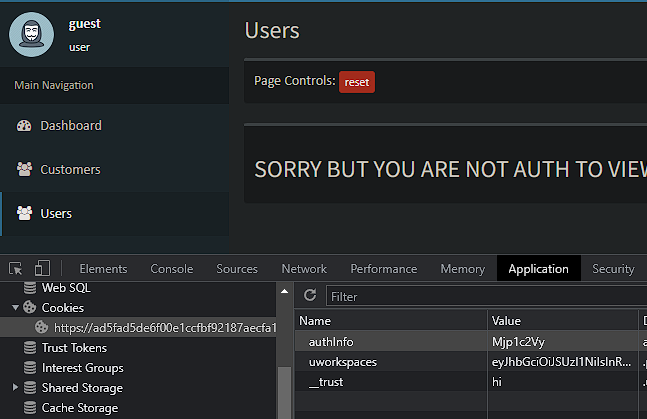
Description automatically generated with medium confidence

Text

Description automatically generated

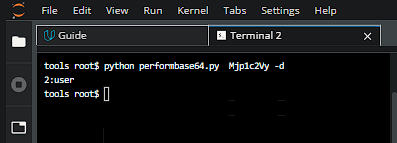
----------------------------------------------------  
Checked the users and customers page but the guest user not auth to view

1. **Insecure-Deserialization**

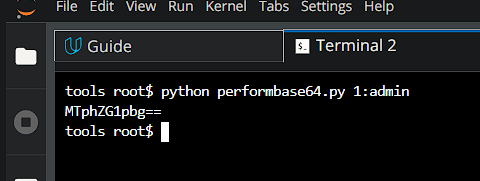


There’s cookie “authinfo” with value=”Mjp1c2Vy”

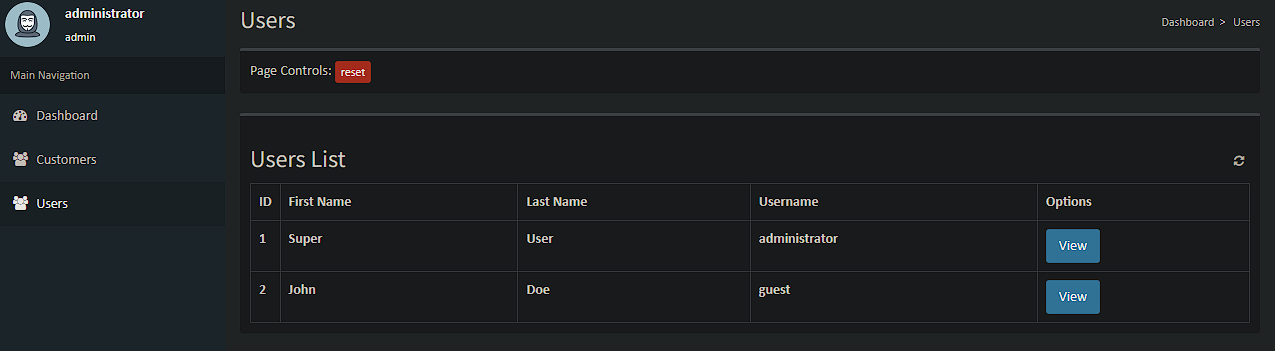
It’s base64 encoded

After decoded it with performbase64.py 

Let’s change the value to 1:admin  
then encode it



1. **Broken-Access**

Let’s modify the cookie value from Developer options MTphZG1pbg== and reload the page 

We login as administrator now

1. **Broken-Access**

A screenshot of a computer

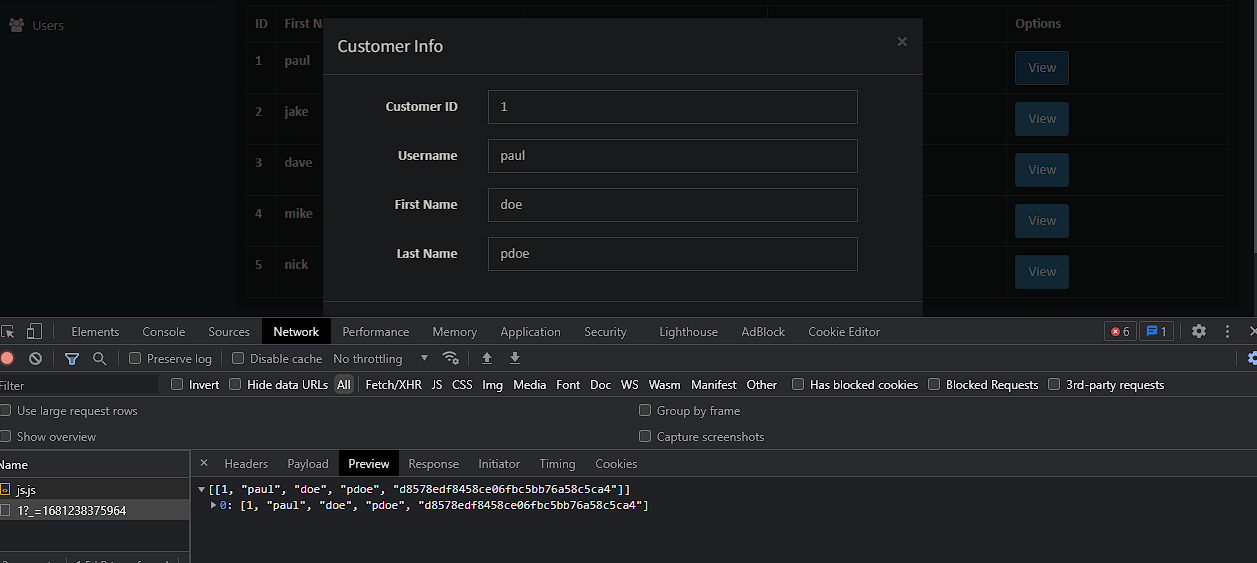
Description automatically generated with medium confidence

After we login as admin we able to see users list and customers list A screenshot of a computer

Description automatically generated with medium confidence

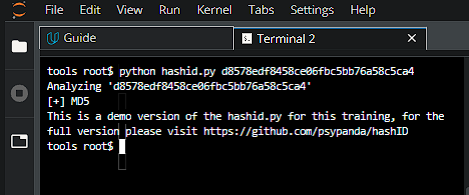
1. **Sensitive Data Exposure**

Let’s check the customer’s page and investigate the columns.

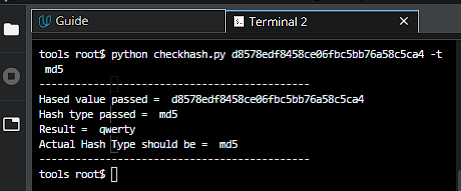
There’s a HASH value 

1. **Security misconfiguration**

Using md5 to hash user’s password

Let’s use the “hashid.py” to identify the hash 

It’s md5 hash now let’s check it with “checkhash.py”.



So the password for the user ‘paul’ is “qwerty”

1. **Security misconfiguration**

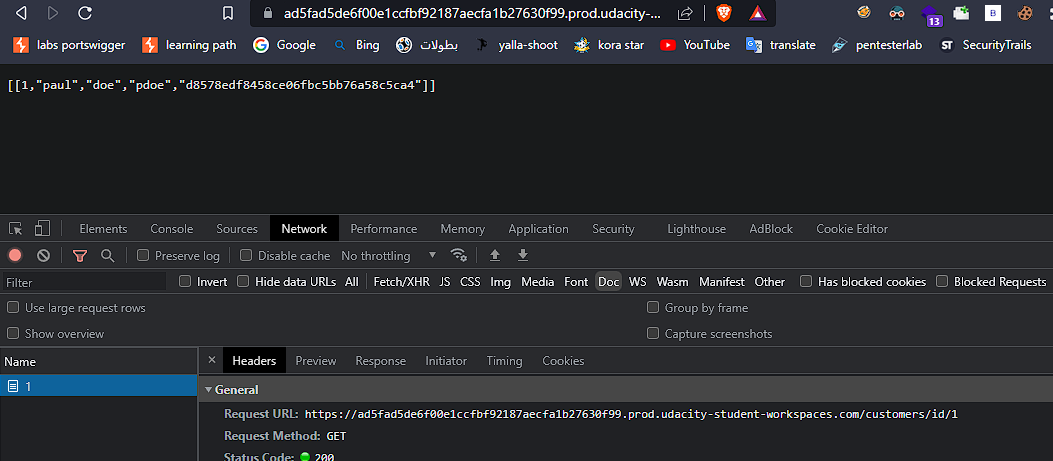
A screenshot of a computer

Description automatically generated with medium confidence

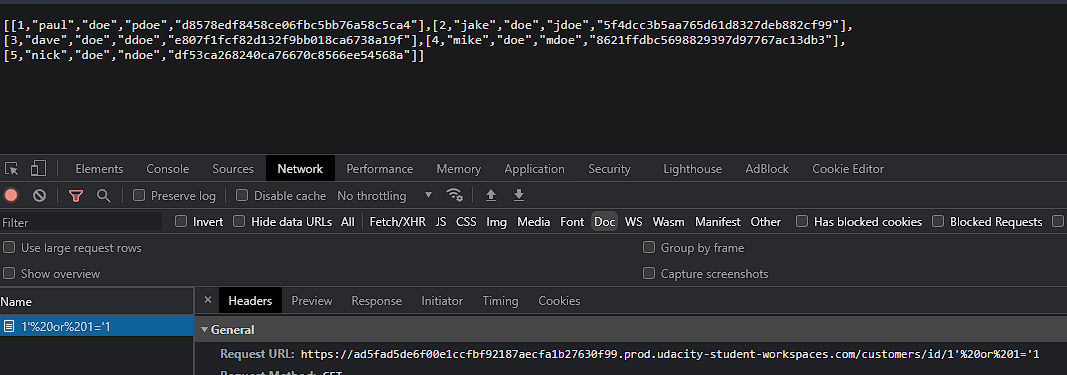
Using base64 to encode cookies.

1. **SQL-Injection**

Let’s navigate the first user in customers list.

From this link: [https://ad5fad5de6f00e1ccfbf92187aecfa1b27630f99.prod.udacity-student-workspaces.com/customers/id/1](https://ad5fad5de6f00e1ccfbf92187aecfa1b27630f99.prod.udacity-student-workspaces.com/customers/id/1)

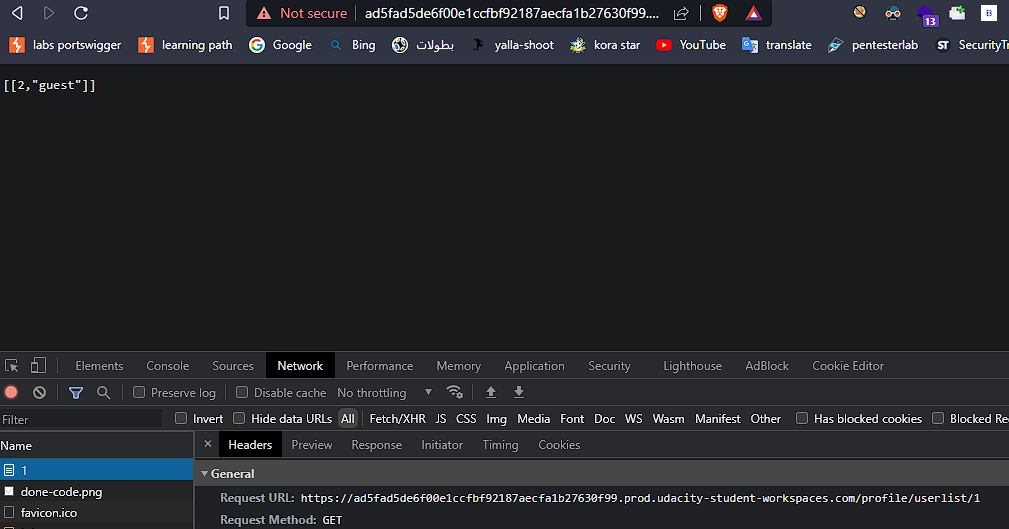
Now in the url add this payload   
<https://ad5fad5de6f00e1ccfbf92187aecfa1b27630f99.prod.udacity-student-workspaces.com/customers/id/1’> or 1=’1

It retrieve the whole customer’s list data

1. **SQL-INJECTION**

Also there’s sql injection at this endpoint

https://ad5fad5de6f00e1ccfbf92187aecfa1b27630f99.prod.udacity-student-workspaces.com/profile/userlist/1



<https://ad5fad5de6f00e1ccfbf92187aecfa1b27630f99.prod.udacity-student-workspaces.com/profile/userlist/1>’ or 1=’1

