Practicals

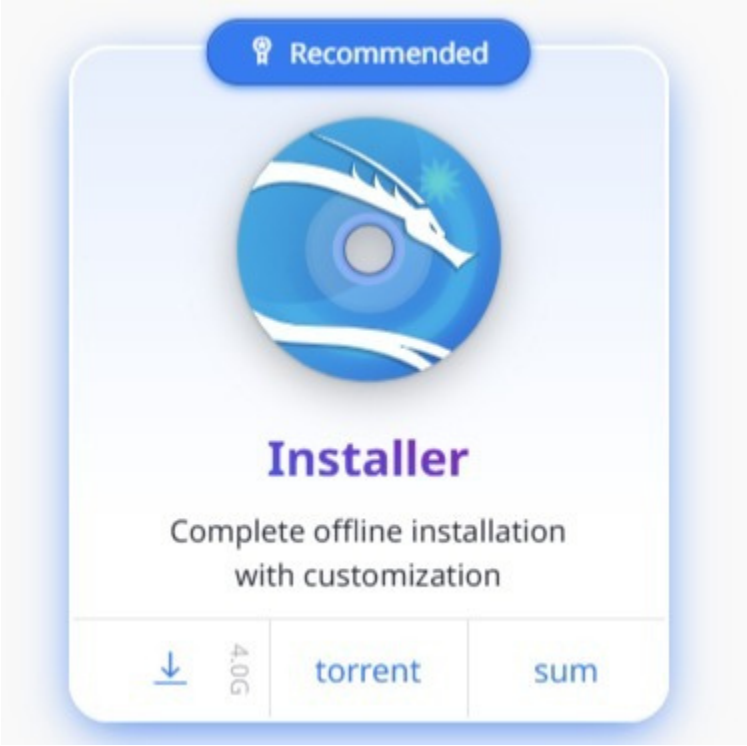
Q1) Introduction to Vulnerabilities and OWASP Top 10:

Set up a local installation of OWASP Juice Shop or DVWA (Damn Vulnerable Web Application) on a virtual machine.Identify vulnerabilities such as SQL Injection or Cross-Site Scripting (XSS). Tools: VirtualBox/VMware, OWASP Juice Shop/DVWA.

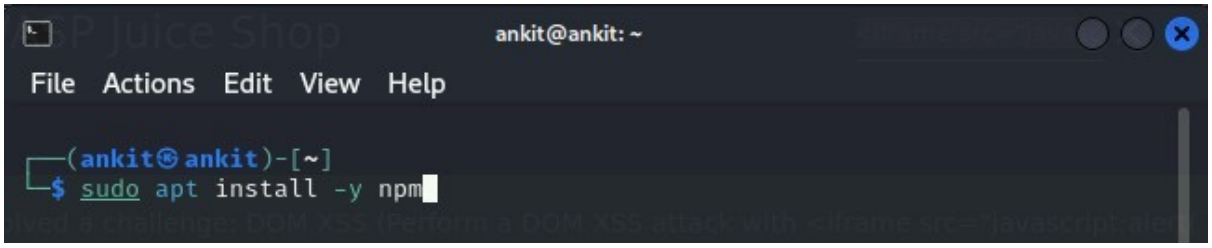
Practical:-

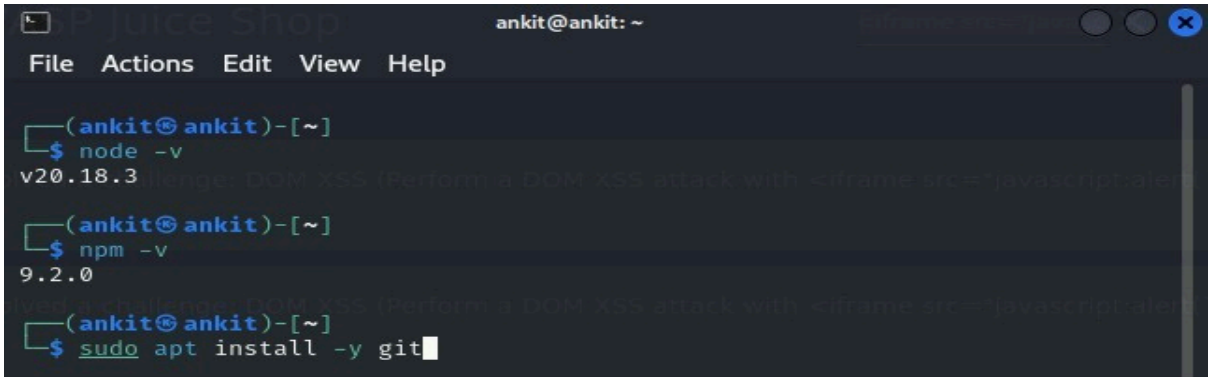
Install Kali Linux ISO from- ~~https://www.kali.org/get-kali/#kali-installer-images~~

Click on 4.0G install button

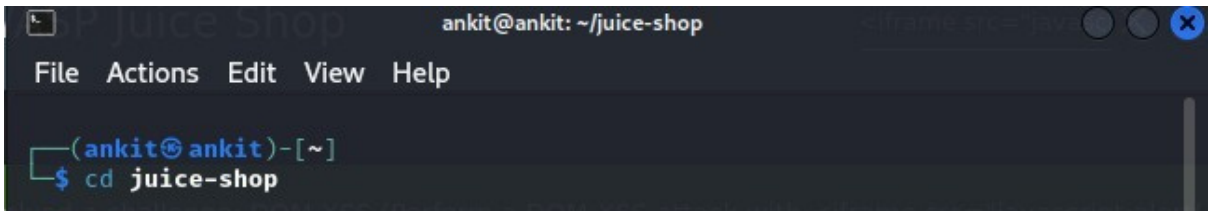


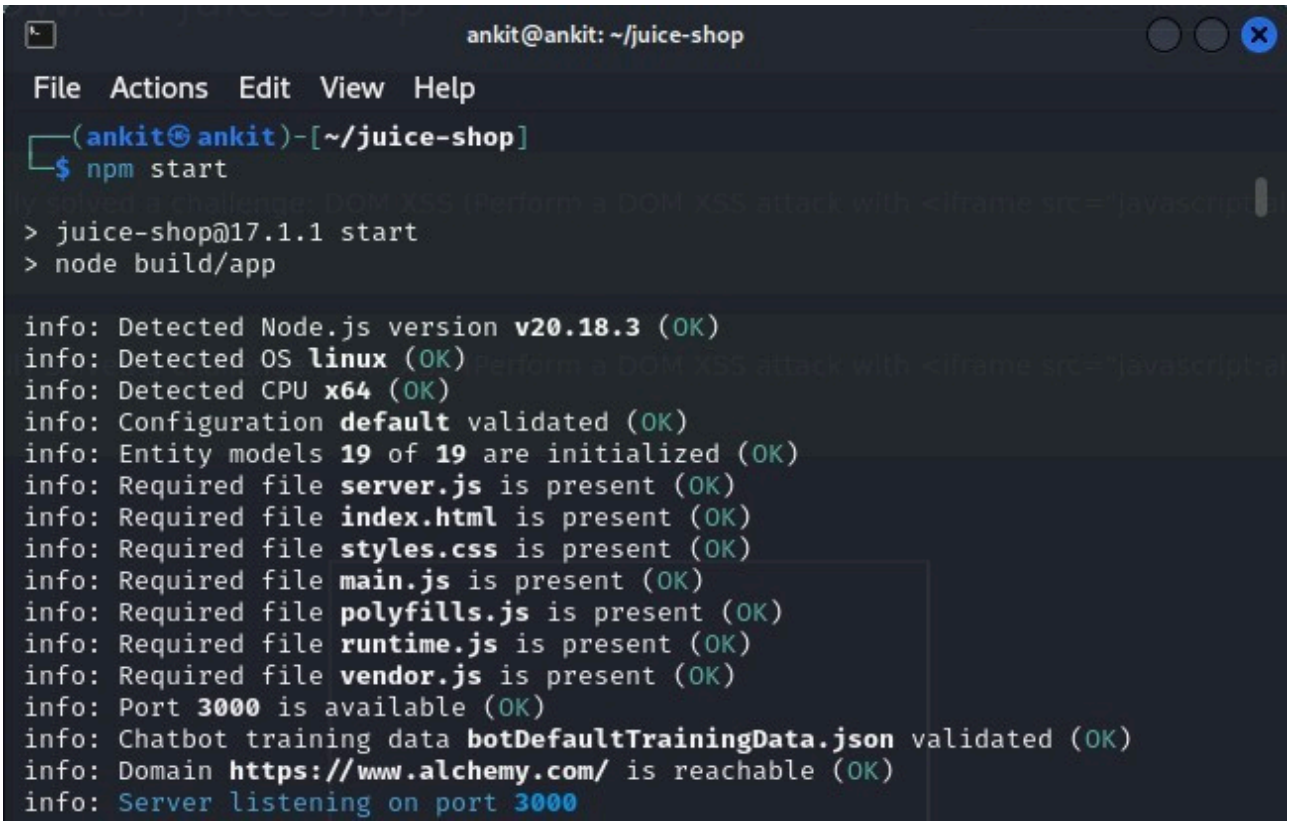
Setup Kali Linux on Virtual Box as a New Machine then follow the following steps:-


ankit@ankit: ~
File Actions Edit View Help
(ankit㉿ankit)-[~]
$ sudo apt install -y npm

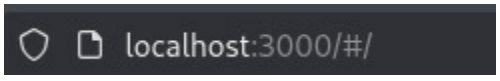

ankit@ankit: ~
File Actions Edit View Help
(ankit㉿ankit)-[~]
$ node -v
v20.18.3
(ankit㉿ankit)-[~]
$ npm -v
9.2.0
(ankit㉿ankit)-[~]
$ sudo apt install -y git

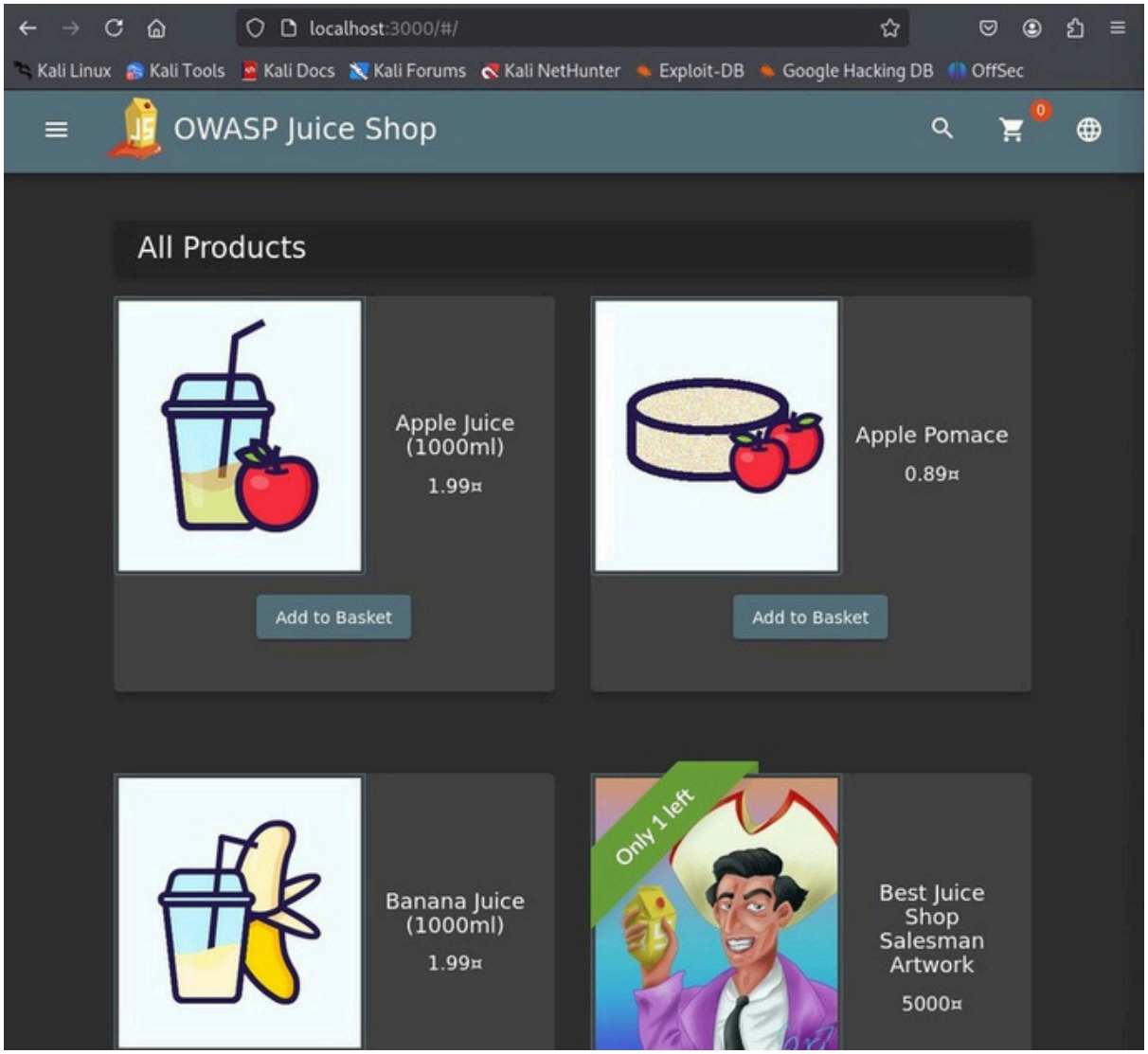

ankit@ankit: ~
File Actions Edit View Help
(ankit㉿ankit)-[~]
$ node -v
v20.18.3
(ankit㉿ankit)-[~]
$ npm -v
9.2.0
(ankit㉿ankit)-[~]
$ git config --global http.postBuffer 104857600
(ankit㉿ankit)-[~]
$ git clone --depth 1 https://github.com/juice-shop/juice-shop.git


ankit@ankit: ~/juice-shop
File Actions Edit View Help
(ankit㉿ankit)-[~]
$ cd juice-shop



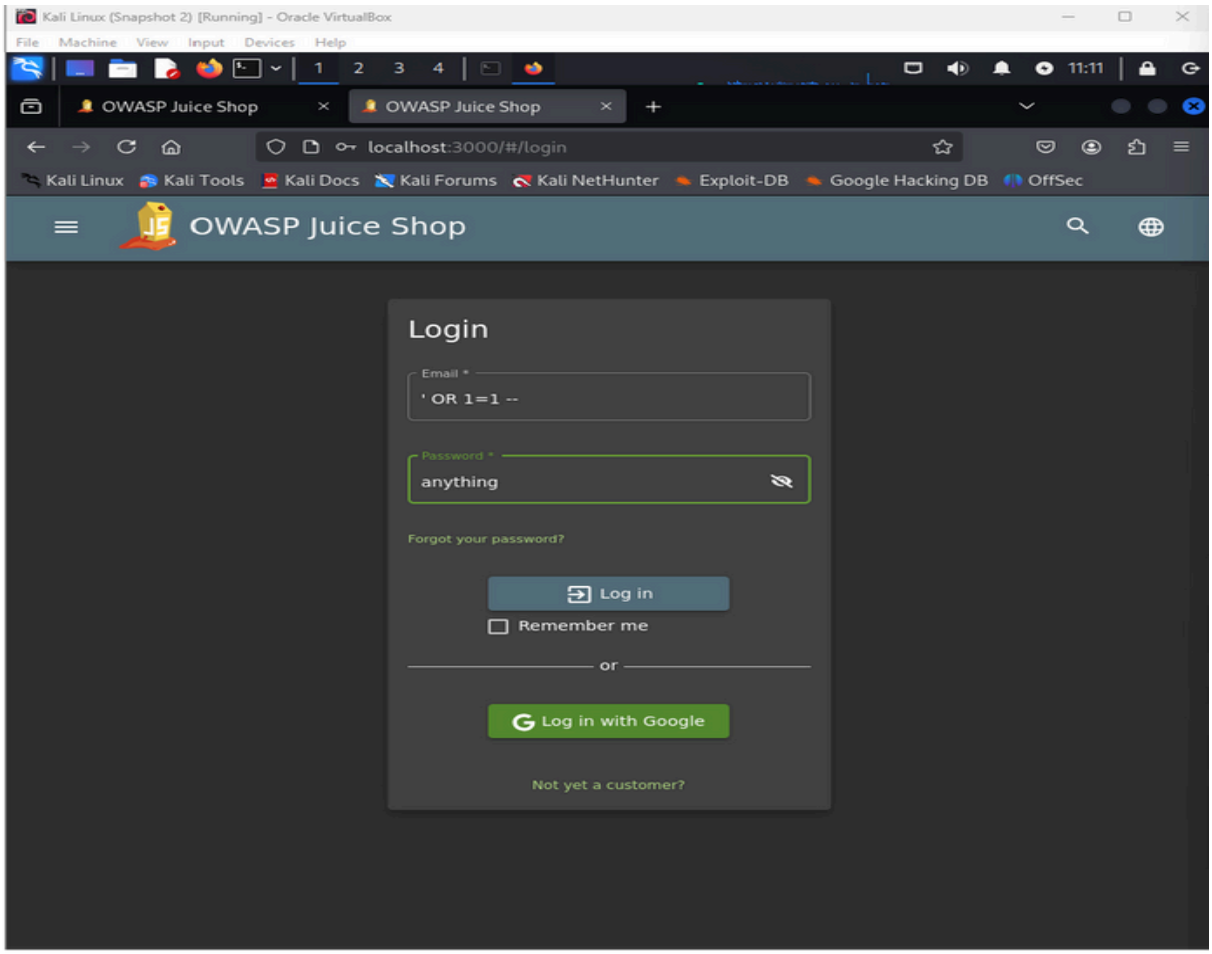
Now visit the following link using the Mozilla Firefox(or any other available) search engine:-

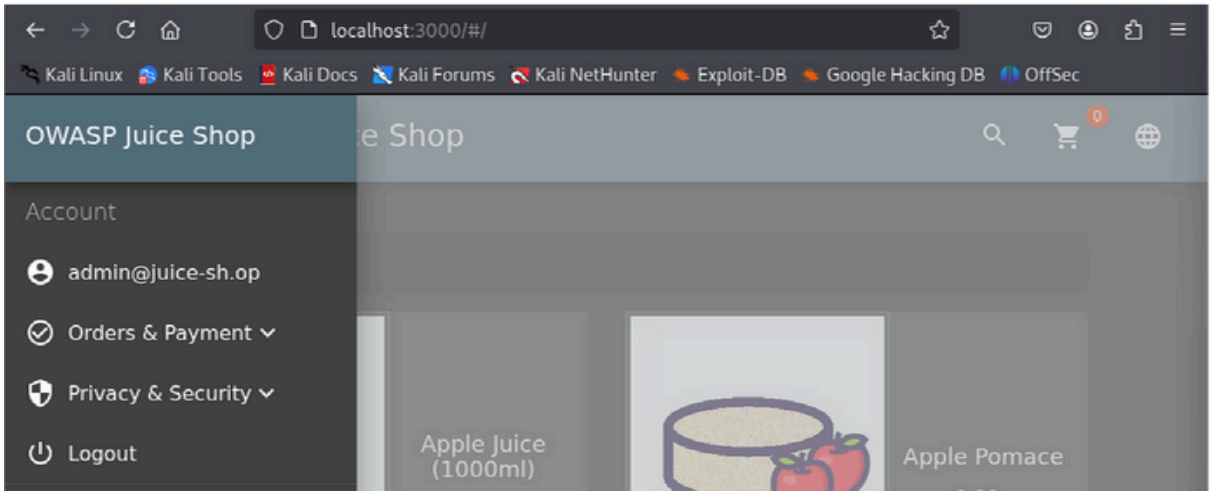




This page shows that owasp juice-shop is running successfully.

Now click on Account button on top right corner of the website and click on Login. Then on the login page type the following text on the Email textbox and type anything you want in the password textbox.



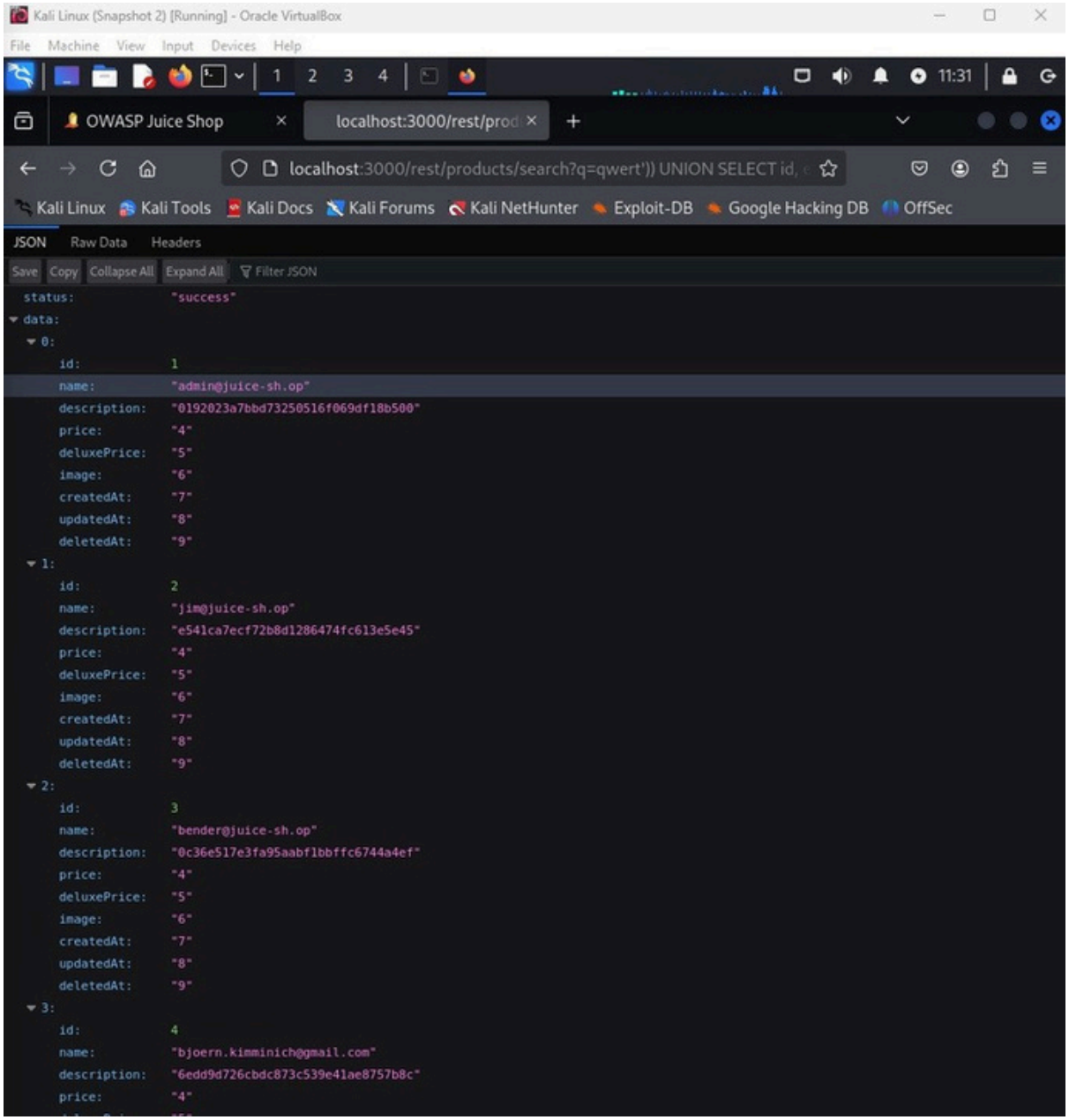


If you see this page after clicking Log in then you have successfully used sql injection to get admin access in owasp.

Now write the following in the search bar of Juice Shop:- UNION SELECT id, email, password, '4', '5', '6', '7', '8', '9' FROM Users–

OR type the following link in the search engine:-

http://localhost:3000/rest/products/search?q=qwert%27))%20UNION%20SELECT%20id,%20email,%20password,%20%274%27,%20%275%27,%20%276%27,%20%277%27,%20%278%27,%20%279%27%20FROM%20Users--

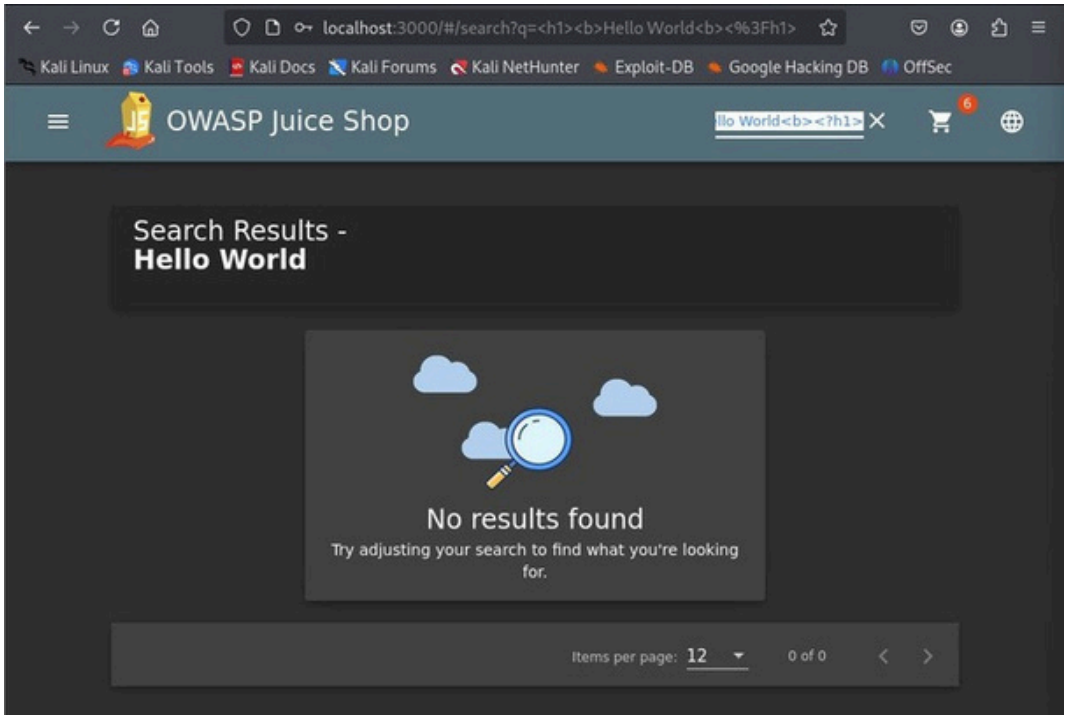


If this page shows up then you have successfully used SQL Injection to get the database containing all user details.

Now to check XSS vulnerabilities.

Write the following command in the search bar of Juice Shop:-

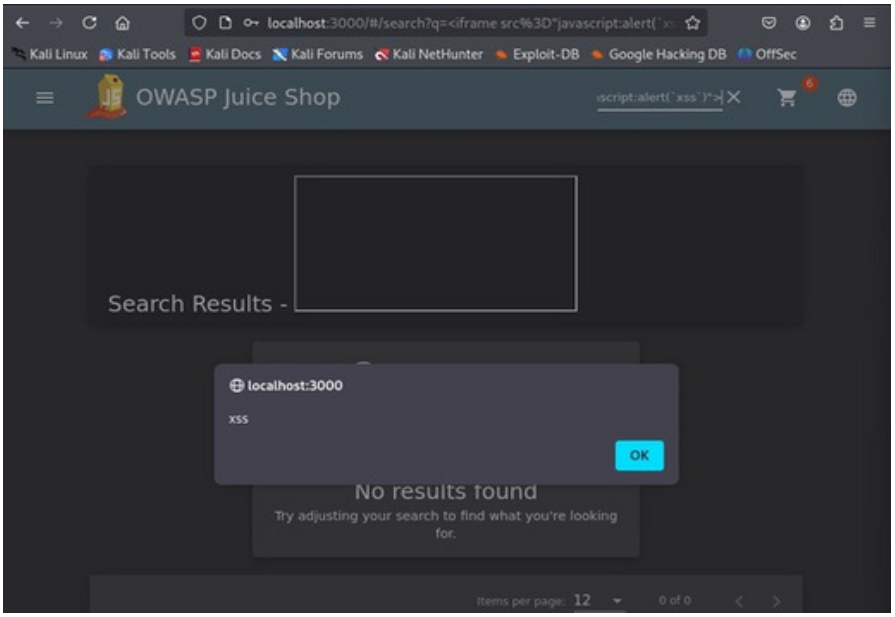
<h1><b>Hello World<b></h1>



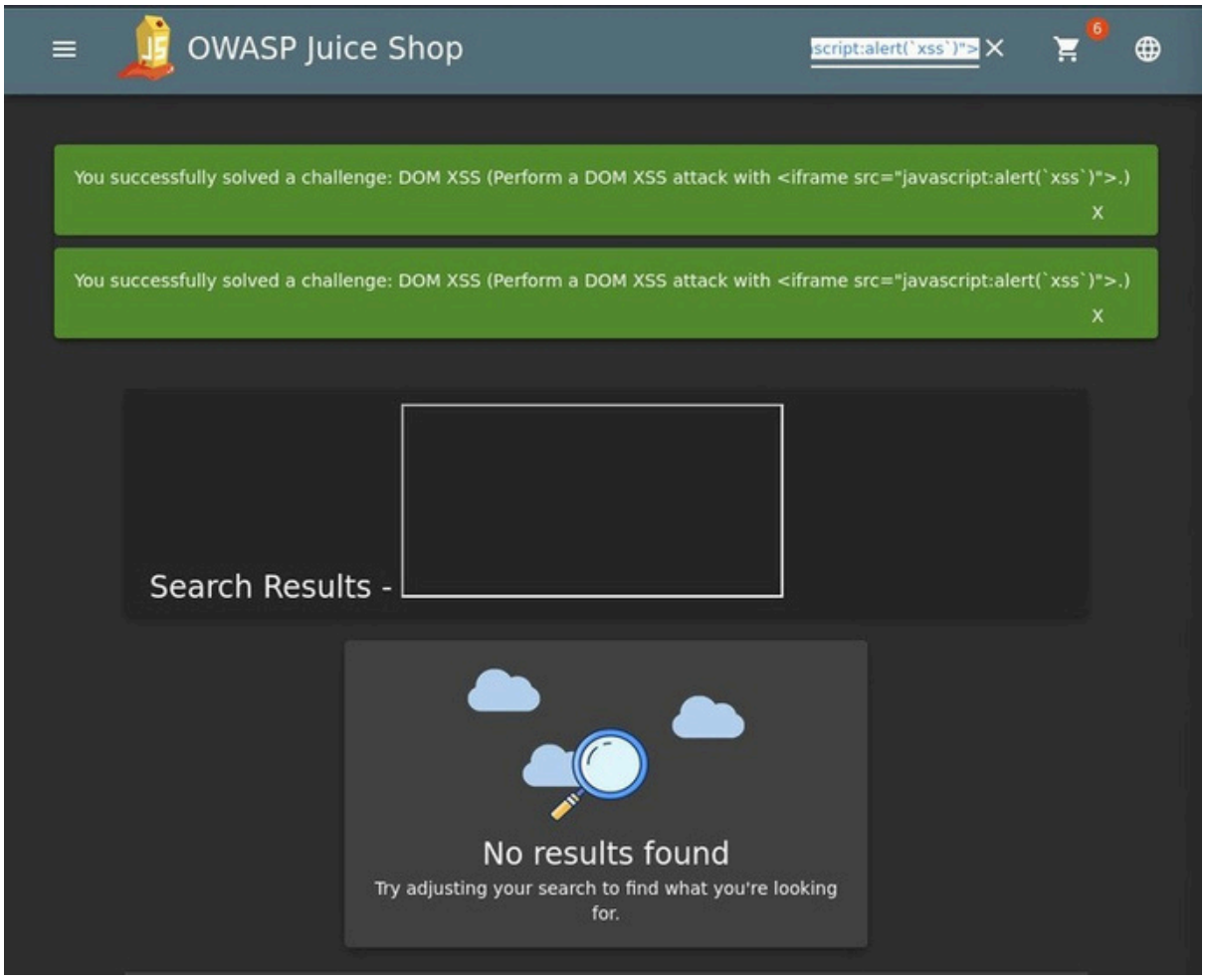
This shows that we can inject HTML in Juice shop.

Next write the following command in the search bar:-

<iframe src="javascript:alert(`xss`)">



The following popup will show that DOM XSS attack can also be done on Juice Shop.



This page after clicking Ok on the popup shows a successful DOM XSS vulnerability.

Q2) Use tools like ipconfig, ifconfig, arp, and ping to explore network configurations on a test VM.Perform network scanning using Nmap on a local network (ensure permission is obtained).
You may use Nmap, Command-line tools, Virtual Machines.

Practical:-

1) Install iputils and nmap

```
[achauhan@vbox ~]$ sudo dnf install net-tools iputils -y
```

```
[achauhan@vbox ~]$ sudo dnf install nmap -y
```

2) Run the following commands

```
[achauhan@vbox ~]$ ifconfig
```

```
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fd00::a00:27ff:fe3b:f21b  prefixlen 64  scopeid 0x0<global>
        inet6 fe80::a00:27ff:fe3b:f21b  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:3b:f2:1b  txqueuelen 1000  (Ethernet)
        RX packets 1411930  bytes 1486351562 (1.3 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 343909  bytes 23403275 (22.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
[achauhan@vbox ~]$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.2.3                 ether   52:55:0a:00:02:03   C                     enp0s3
10.0.2.2                 ether   52:55:0a:00:02:02   C                     enp0s3
```

```
[achauhan@vbox ~]$ ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=255 time=10.2 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=255 time=4.33 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=255 time=8.91 ms
64 bytes from 192.168.1.1: icmp_seq=4 ttl=255 time=8.23 ms
64 bytes from 192.168.1.1: icmp_seq=5 ttl=255 time=4.79 ms
64 bytes from 192.168.1.1: icmp_seq=6 ttl=255 time=7.24 ms
64 bytes from 192.168.1.1: icmp_seq=7 ttl=255 time=9.10 ms
64 bytes from 192.168.1.1: icmp_seq=8 ttl=255 time=4.92 ms
64 bytes from 192.168.1.1: icmp_seq=9 ttl=255 time=4.63 ms
64 bytes from 192.168.1.1: icmp_seq=10 ttl=255 time=5.99 ms
64 bytes from 192.168.1.1: icmp_seq=11 ttl=255 time=7.91 ms
64 bytes from 192.168.1.1: icmp_seq=12 ttl=255 time=7.48 ms
64 bytes from 192.168.1.1: icmp_seq=13 ttl=255 time=5.20 ms
64 bytes from 192.168.1.1: icmp_seq=14 ttl=255 time=7.86 ms
64 bytes from 192.168.1.1: icmp_seq=15 ttl=255 time=9.39 ms
64 bytes from 192.168.1.1: icmp_seq=16 ttl=255 time=9.23 ms
64 bytes from 192.168.1.1: icmp_seq=17 ttl=255 time=8.86 ms
64 bytes from 192.168.1.1: icmp_seq=18 ttl=255 time=8.52 ms
```

```
[achauhan@vbox ~]$ ip stat
1: lo: group offload subgroup hw_stats_info
    l3_stats off used off
1: lo: group xstats_slave subgroup bond suite 802.3ad
1: lo: group xstats_slave subgroup bridge suite mcast
1: lo: group xstats_slave subgroup bridge suite stp
1: lo: group xstats subgroup bond suite 802.3ad
1: lo: group xstats subgroup bridge suite mcast
1: lo: group xstats subgroup bridge suite stp
1: lo: group afstats subgroup mpls
1: lo: group offload subgroup l3_stats off used off
1: lo: group offload subgroup cpu_hit

1: lo: group link
    RX:  bytes packets errors dropped  missed   mcast
               0       0      0       0       0       0
    TX:  bytes packets errors dropped carrier collsns
               0       0      0       0       0       0

2: enp0s3: group offload subgroup hw_stats_info
    l3_stats off used off
2: enp0s3: group xstats_slave subgroup bond suite 802.3ad
2: enp0s3: group xstats_slave subgroup bridge suite mcast
2: enp0s3: group xstats_slave subgroup bridge suite stp
2: enp0s3: group xstats subgroup bond suite 802.3ad
2: enp0s3: group xstats subgroup bridge suite mcast
2: enp0s3: group xstats subgroup bridge suite stp
2: enp0s3: group afstats subgroup mpls
2: enp0s3: group offload subgroup l3_stats off used off
2: enp0s3: group offload subgroup cpu_hit

2: enp0s3: group link
    RX:  bytes packets errors dropped  missed   mcast
    1486367620 1412100      0       0       0      19
    TX:  bytes packets errors dropped carrier collsns
      23420021  344087      0       0       0       0
```

```
[achauhan@vbox ~]$ ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
    link/ether 08:00:27:3b:f2:1b brd ff:ff:ff:ff:ff:ff
```

```
[achauhan@vbox ~]$ nmap 192.168.1.1
Starting Nmap 7.92 ( https://nmap.org ) at 2025-02-17 14:06 IST
Note: Host seems down. If it is really up, but blocking our ping probes, tr
Nmap done: 1 IP address (0 hosts up) scanned in 3.21 seconds
```

Q3) Use John the Ripper or Hashcat to crack simple password hashes generated using online tools.

Practical:-

Step1) Download John the Ripper from google from the following website

https://www.openwall.com/john/

John the Ripper is free and Open Source software, distributed primarily in source code form. If you wo
meant to be easier to install and use while delivering optimal performance.

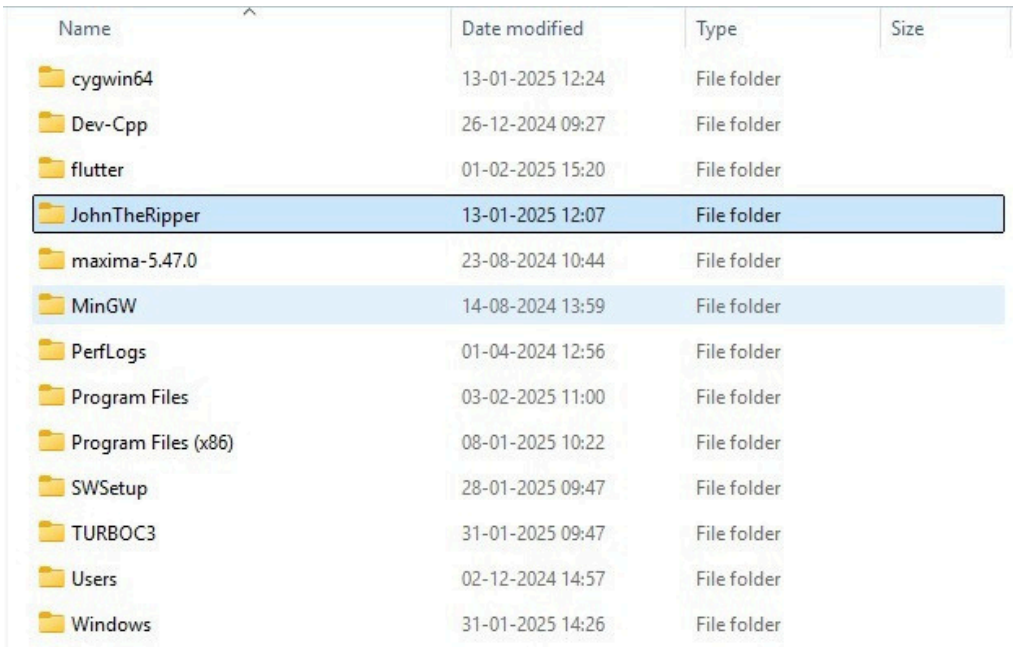
Proceed to **John the Ripper *Pro*** homepage for your OS:

- John the Ripper *Pro* for Linux
- John the Ripper *Pro* for macOS
- **On Windows, consider Hash Suite** (developed by a contributor to John the Ripper)
- On Android, consider Hash Suite Droid

Download the latest John the Ripper jumbo release (release notes) or development snapshot:

- 1.9.0-jumbo-1 sources in tar.xz, 33 MB (signature) or tar.gz, 43 MB (signature)
- **1.9.0-jumbo-1 64-bit Windows binaries in 7z, 22 MB (signature) or zip, 63 MB (signature)**
- **1.9.0-jumbo-1 32-bit Windows binaries in 7z, 21 MB (signature) or zip, 61 MB (signature)**
- Development source code in GitHub repository (download as tar.gz or zip)

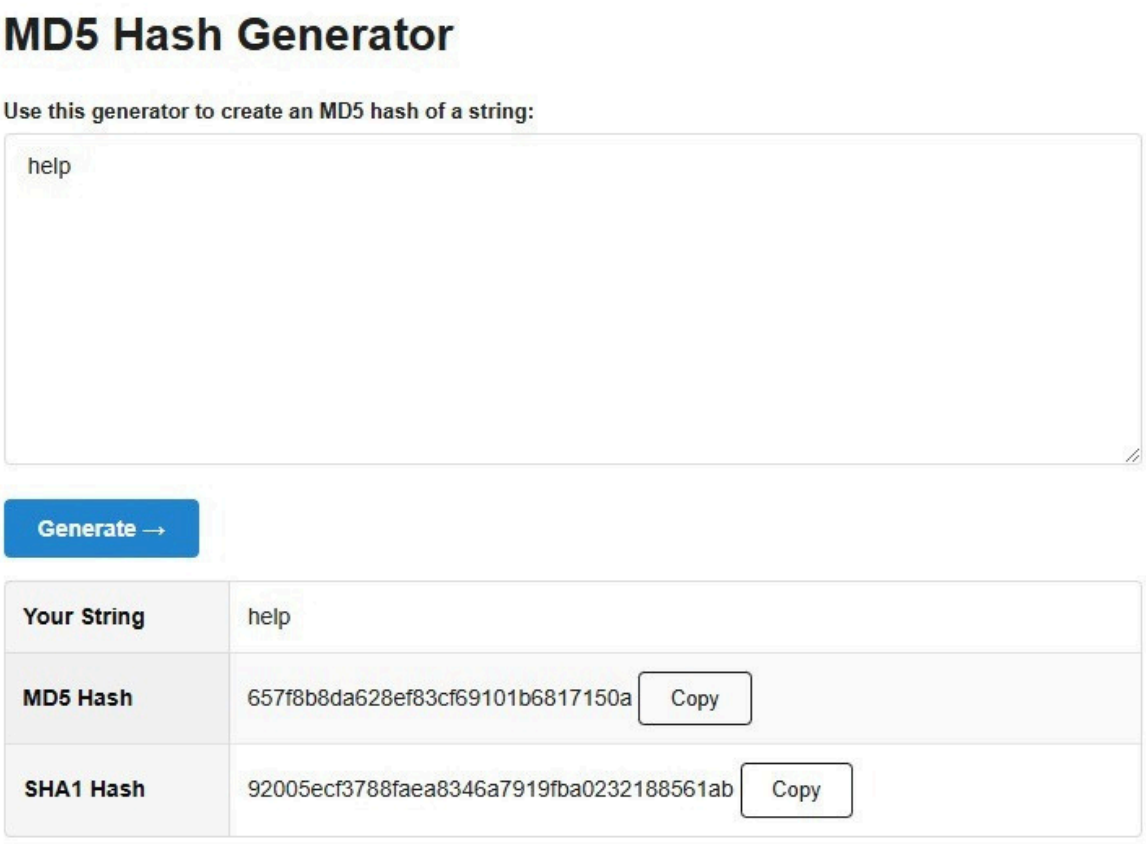
Step2) Extract the downloaded file and save the file in C drive, then change the name of the file in C drive to JohnTheRipper



Name	Date modified	Type	Size
cygwin64	13-01-2025 12:24	File folder	
Dev-Cpp	26-12-2024 09:27	File folder	
flutter	01-02-2025 15:20	File folder	
JohnTheRipper	13-01-2025 12:07	File folder	
maxima-5.47.0	23-08-2024 10:44	File folder	
MinGW	14-08-2024 13:59	File folder	
PerfLogs	01-04-2024 12:56	File folder	
Program Files	03-02-2025 11:00	File folder	
Program Files (x86)	08-01-2025 10:22	File folder	
SWSetup	28-01-2025 09:47	File folder	
TURBOC3	31-01-2025 09:47	File folder	
Users	02-12-2024 14:57	File folder	
Windows	31-01-2025 14:26	File folder	

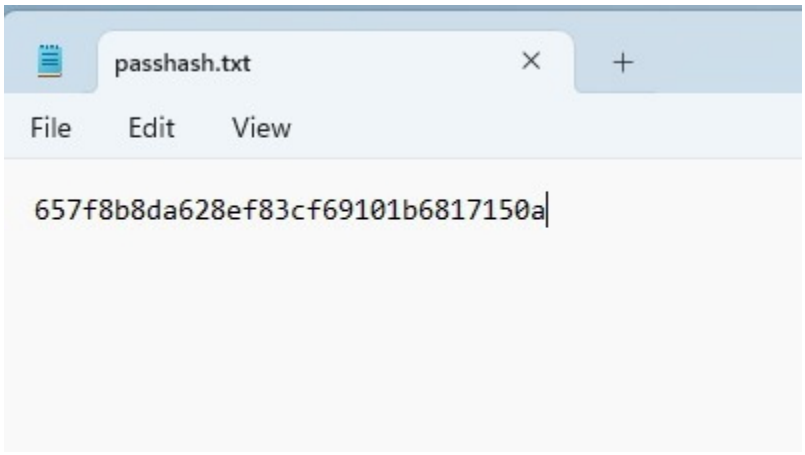
Step3) Now open an MD5 hash generator on google and generate hash value for any password of length of upto 4 letters.

https://www.md5hashgenerator.com/

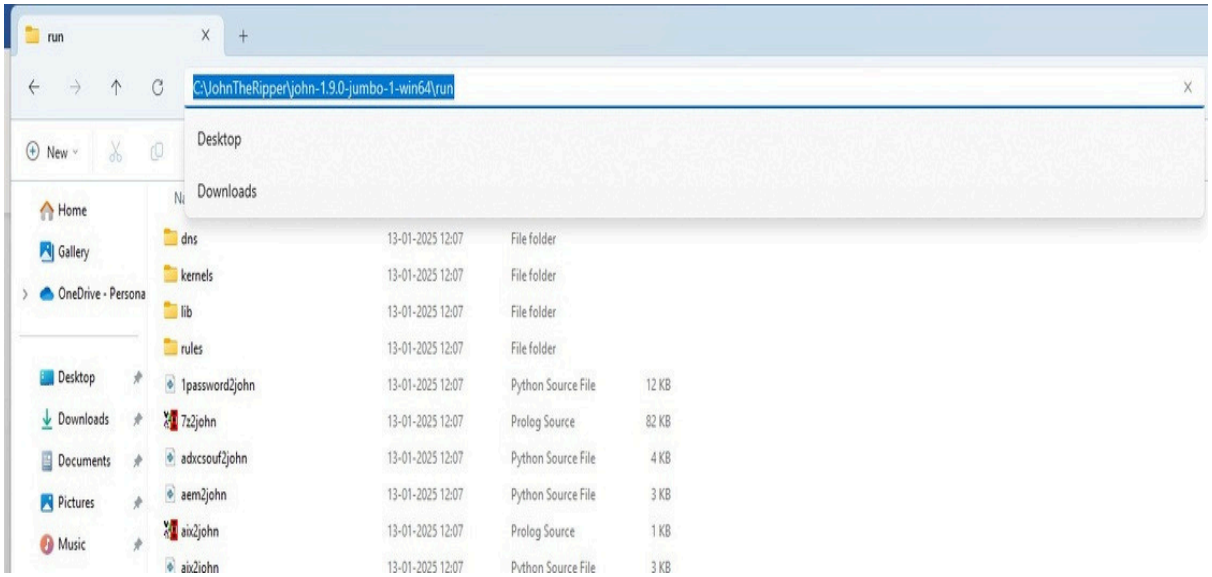


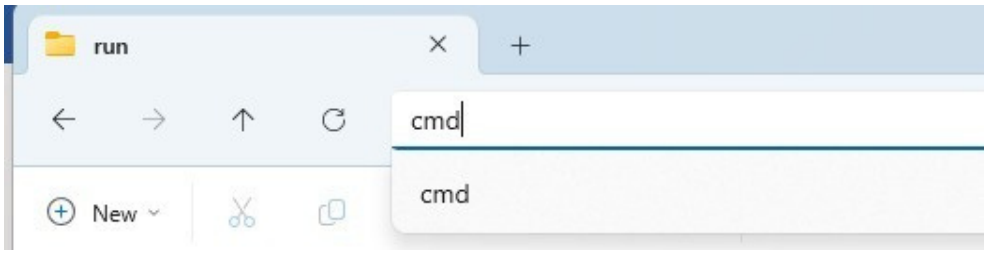
Step4) Now copy the MD5 hash value

Step5) Open a new Notepad file and save the copied MD5 hash value in this file and save the file with any name.



Step6) Now open C:\JohnTheRipper\john-1.9.0-jumbo-1-win64\run folder and open a CMD by typing cmd on the top in place of the file path name and click Enter.





Step7) In CMD write the following command

john --format=raw-MD5 “hash_file_path”

In my case the command is:- john --format=raw-MD5 C:\Users\HP\Desktop\passhash.txt

```
C:\JohnTheRipper\john-1.9.0-jumbo-1-win64\run>john --format=raw-MD5 C:\Users\HP\Desktop\passhash.txt
```

Step8) Click Enter

```
C:\JohnTheRipper\john-1.9.0-jumbo-1-win64\run>john --format=raw-MD5 C:\Users\HP\Desktop\passhash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 256/256 AVX2 8x3])
Warning: no OpenMP support for this hash type, consider --fork=20
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:password.lst, rules:Wordlist
help             (?)
1g 0:00:00:00 DONE 2/3 (2025-02-03 11:27) 55.55g/s 149333p/s 149333c/s 149333C/s chacha..nermal
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed
```

The cracked password will be visible in the CMD.

Q5) Alternate Data Streams (ADS): Practice creating ADS on a Windows VM to hide files. Explore how to detect and counter such techniques.

• Tools: Windows VM, Command Prompt

Practical:-

Create a file using cmd with some text in it using the following command:

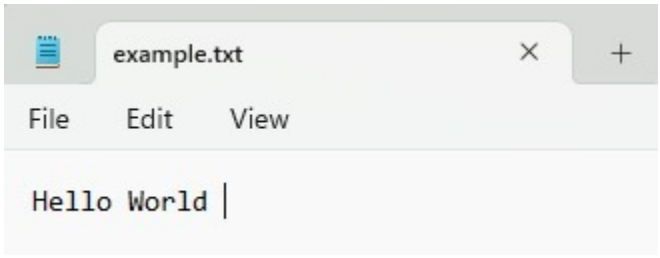
```
C:\Users\HP>cd Desktop

C:\Users\HP\Desktop>echo Hello World > example.txt
```

Now add some hidden text in the file using the following command:

```
C:\Users\HP\Desktop>echo This is hidden data > example.txt:hidden.txt
```

The file looks like this:



Now write the following command to list alternate data streams (ADS) of the file with the following output:

```
C:\Users\HP\Desktop>dir /R example.txt
 Volume in drive C is Windows
 Volume Serial Number is 9A65-D026

 Directory of C:\Users\HP\Desktop

17-03-2025  11:28                14 example.txt
                                 22 example.txt:hidden.txt:$DATA
               1 File(s)             14 bytes
               0 Dir(s)  315,102,126,080 bytes free
```

To read the ADS text use the following command:

```
C:\Users\HP\Desktop>more < example.txt:hidden.txt
This is hidden data
```

To detect ADS using CMD write the following command:

```
C:\Users\HP\Desktop>dir /R C:\Users\HP\Desktop
 Volume in drive C is Windows
 Volume Serial Number is 9A65-D026

 Directory of C:\Users\HP\Desktop

17-03-2025  11:27    <DIR>          .
17-03-2025  11:27    <DIR>          ..
24-10-2024  10:20    <DIR>          24sep
04-03-2025  11:03           125,900 4feb.docx
                                  0 4feb.docx:SandBoxSafeFile:$DATA
                                 50 4feb.docx:Zone.Identifier:$DATA
09-12-2024  15:18            16,682 9-12-24.docx
```

```
09-12-2024  15:18            16,682 9-12-24.docx
21-02-2025  14:44    <DIR>          Ankit CS
20-02-2025  14:23               793 binary_search.py
14-12-2024  09:54           447,966 caf.webp
                                119 caf.webp:Zone.Identifier:$DATA
14-12-2024  09:53            21,208 cafe1
                                147 cafe1:Zone.Identifier:$DATA
04-02-2025  14:00       312,690,998 clagav64.exe
10-09-2024  14:52         7,682,810 CPUSim3.9.0.zip
                                147 CPUSim3.9.0.zip:Zone.Identifier:$DATA
14-02-2025  12:39             1,072 Dev-C++.lnk
14-02-2025  12:38        50,433,966 Dev-Cpp 5.11 TDM-GCC 4.9.2 Setup.exe
21-12-2024  11:11                98 dff.py
12-09-2024  16:47           494,447 digital india.pdf
27-08-2024  14:31    <DIR>          DSA
17-03-2025  11:28                14 example.txt
                                 22 example.txt:hidden.txt:$DATA
20-02-2025  14:14               230 fabonacci.py
20-02-2025  14:32               261 factorial(recursive).py
18-02-2025  12:59               281 factorial.cpp
18-02-2025  12:59         1,921,671 factorial.exe
20-02-2025  14:08               347 factorial.py
16-11-2024  10:30               224 hey.html
22-02-2025  10:23    <DIR>          himanshu
16-11-2024  10:26               203 html.txt
```

To restrict permissions for non-administrative users to prevent them from creating or modifying ADS write the following command:

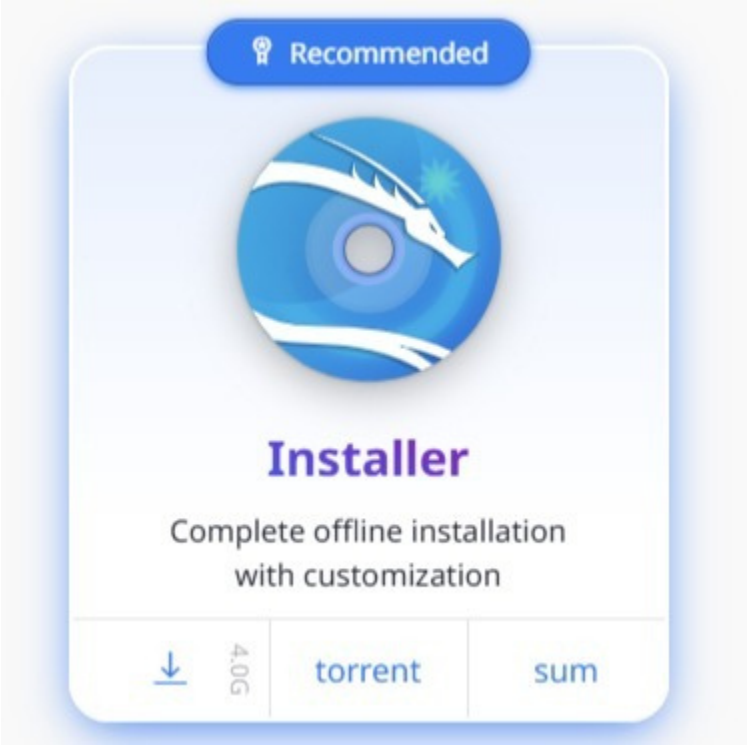
```
C:\Users\HP\Desktop>icacls "C:\Users\HP\Desktop" /deny "User:(M)"
```

Q6) SQL Injection (Test Environment): Perform SQL Injection on DVWA or Juice Shop to extract sample data. Understand how to mitigate such vulnerabilities.
• Tools: DVWA, OWASP Juice Shop.

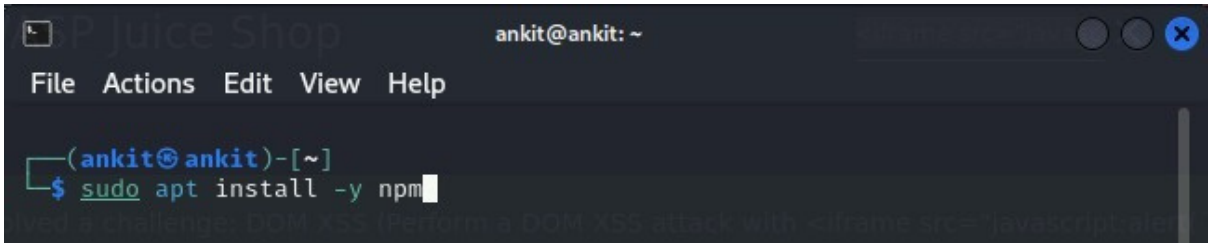
Practical:-

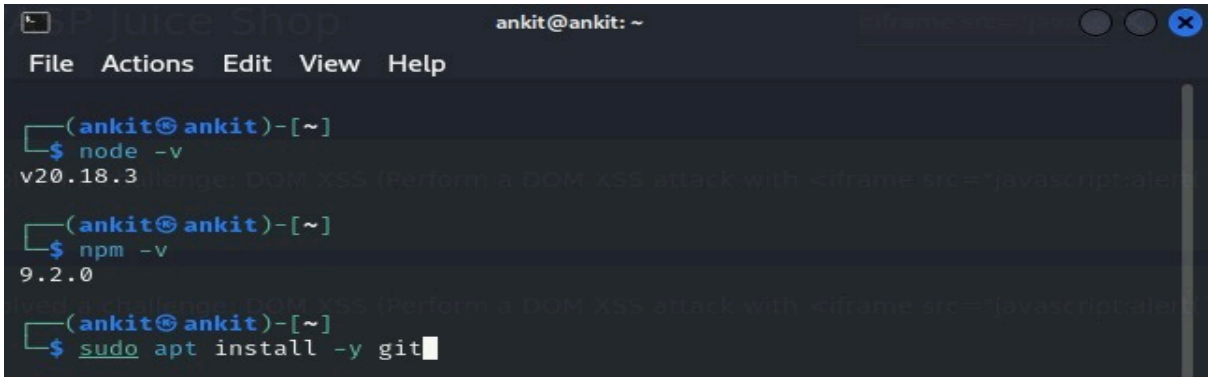
Install Kali Linux ISO from- ~~https://www.kali.org/get-kali/#kali-installer-images~~

Click on 4.0G install button

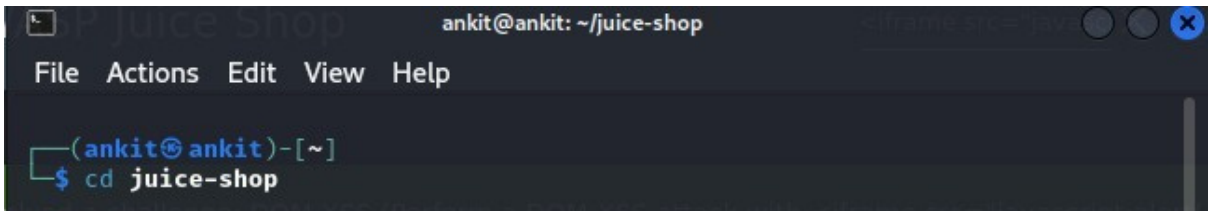


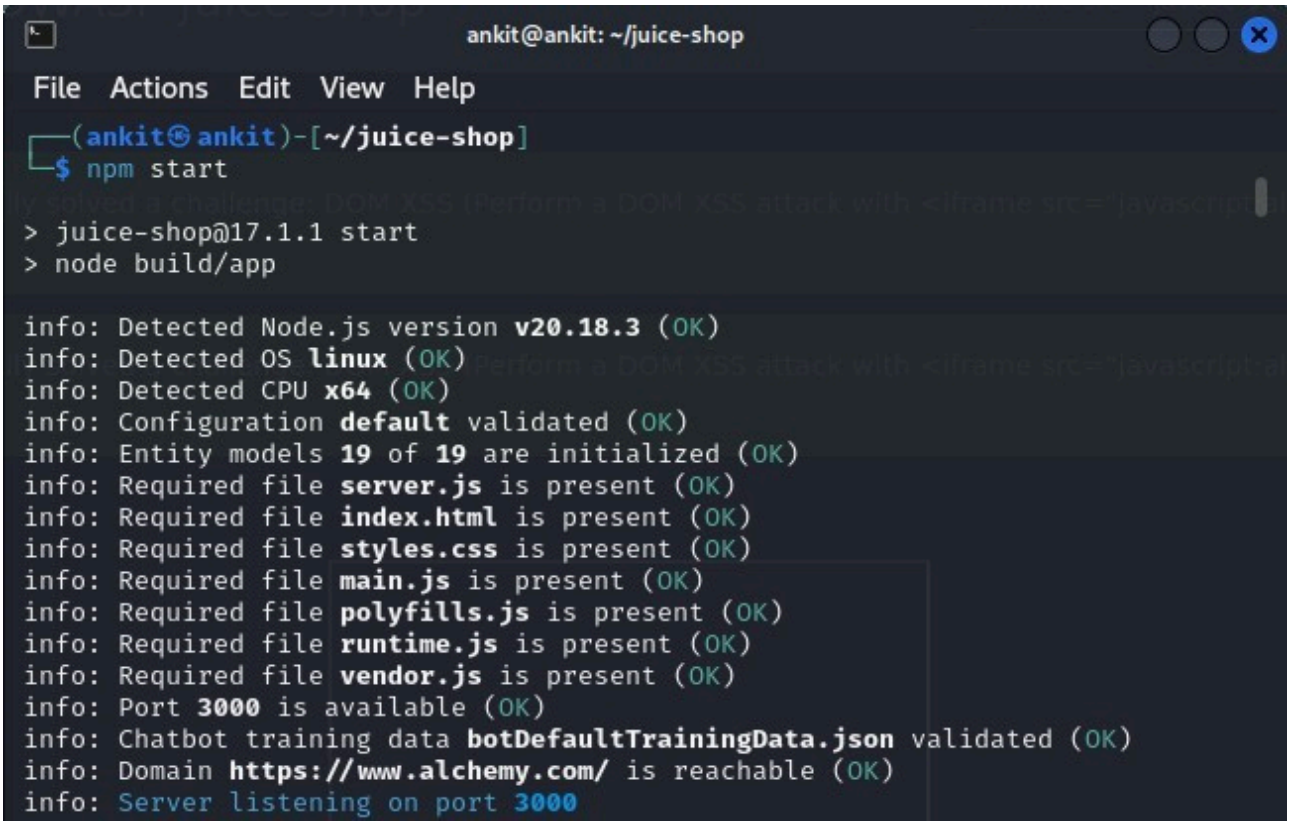
Setup Kali Linux on Virtual Box as a New Machine then follow the following steps:-



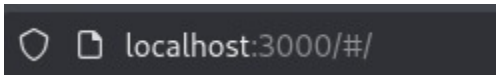

ankit@ankit: ~
File Actions Edit View Help
(ankit㉿ankit)-[~]
$ node -v
v20.18.3
(ankit㉿ankit)-[~]
$ npm -v
9.2.0
(ankit㉿ankit)-[~]
$ sudo apt install -y git

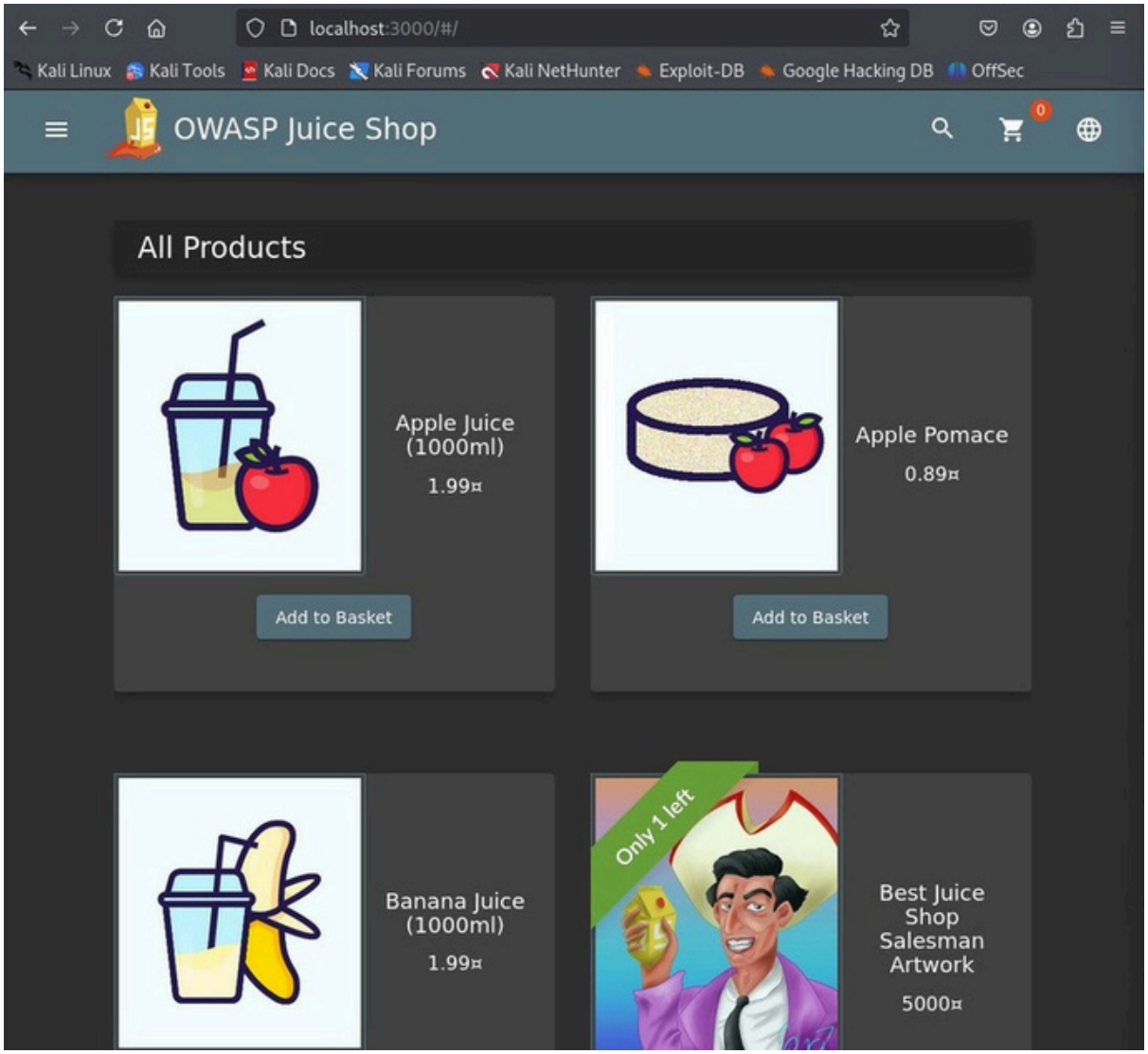

ankit@ankit: ~
File Actions Edit View Help
(ankit㉿ankit)-[~]
$ node -v
v20.18.3
(ankit㉿ankit)-[~]
$ npm -v
9.2.0
(ankit㉿ankit)-[~]
$ git config --global http.postBuffer 104857600
(ankit㉿ankit)-[~]
$ git clone --depth 1 https://github.com/juice-shop/juice-shop.git


ankit@ankit: ~/juice-shop
File Actions Edit View Help
(ankit㉿ankit)-[~]
$ cd juice-shop



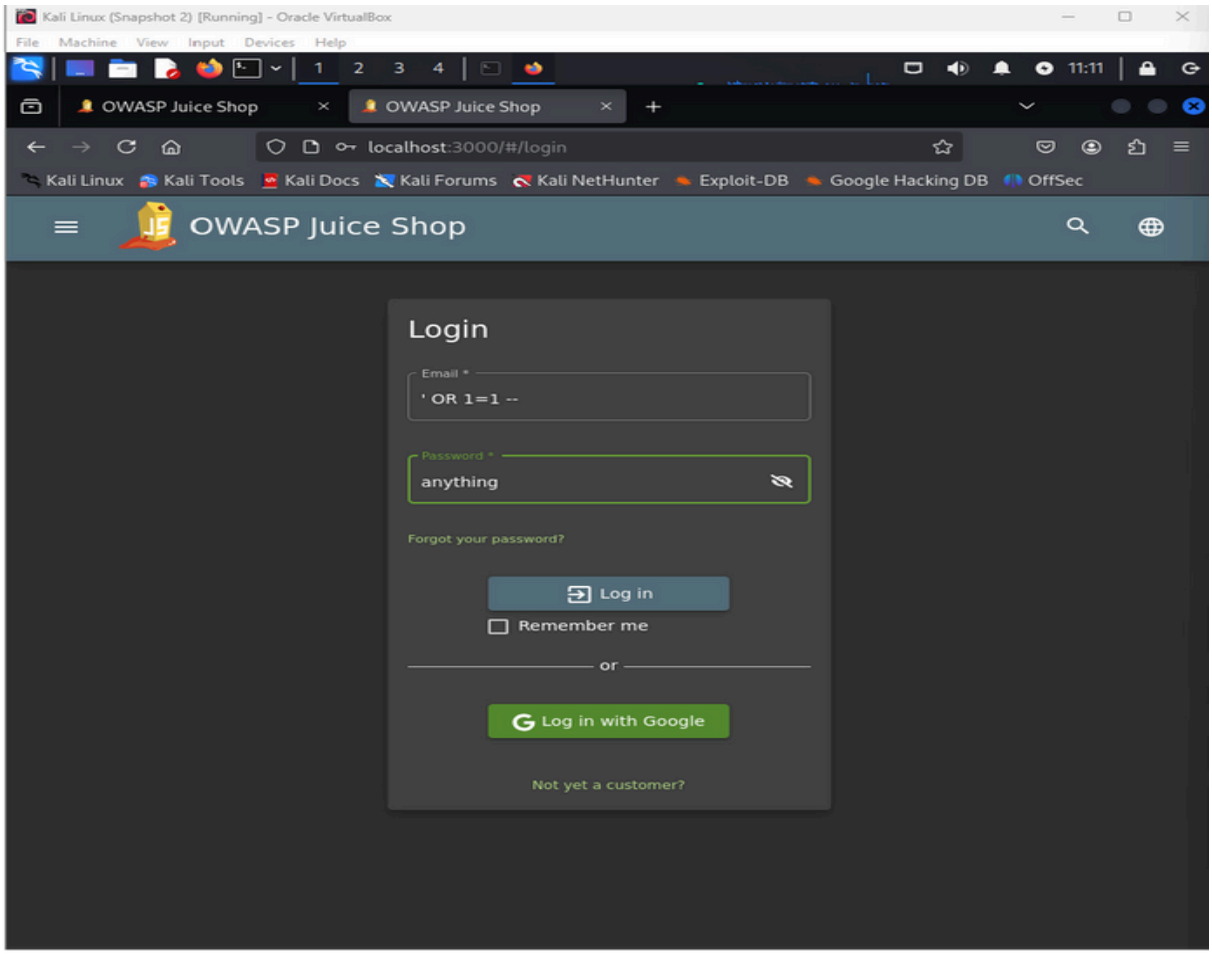
Now visit the following link using the Mozilla Firefox(or any other available) search engine:-

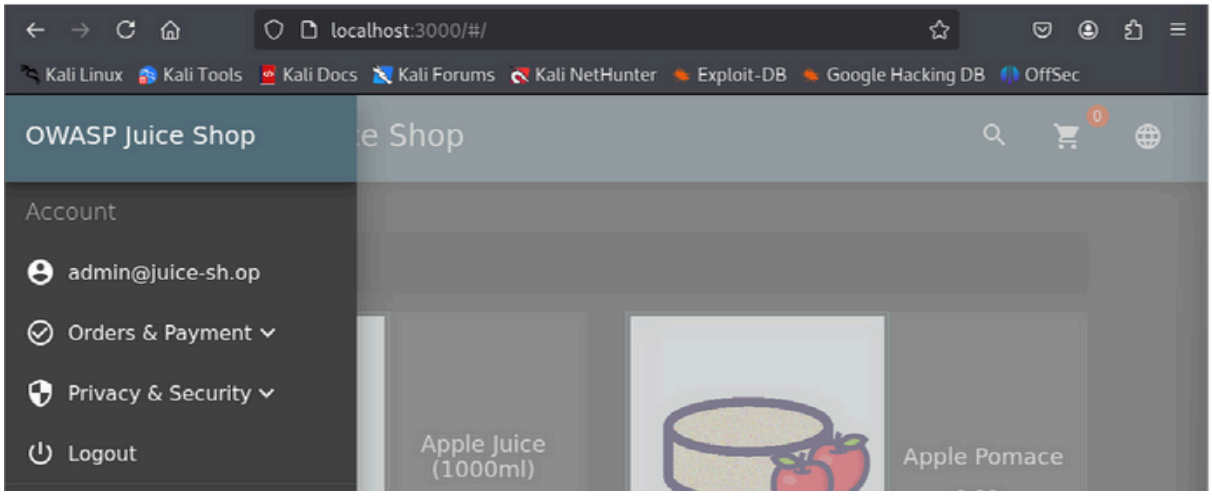




This page shows that owasp juice-shop is running successfully.

Now click on Account button on top right corner of the website and click on Login. Then on the login page type the following text on the Email textbox and type anything you want in the password textbox.



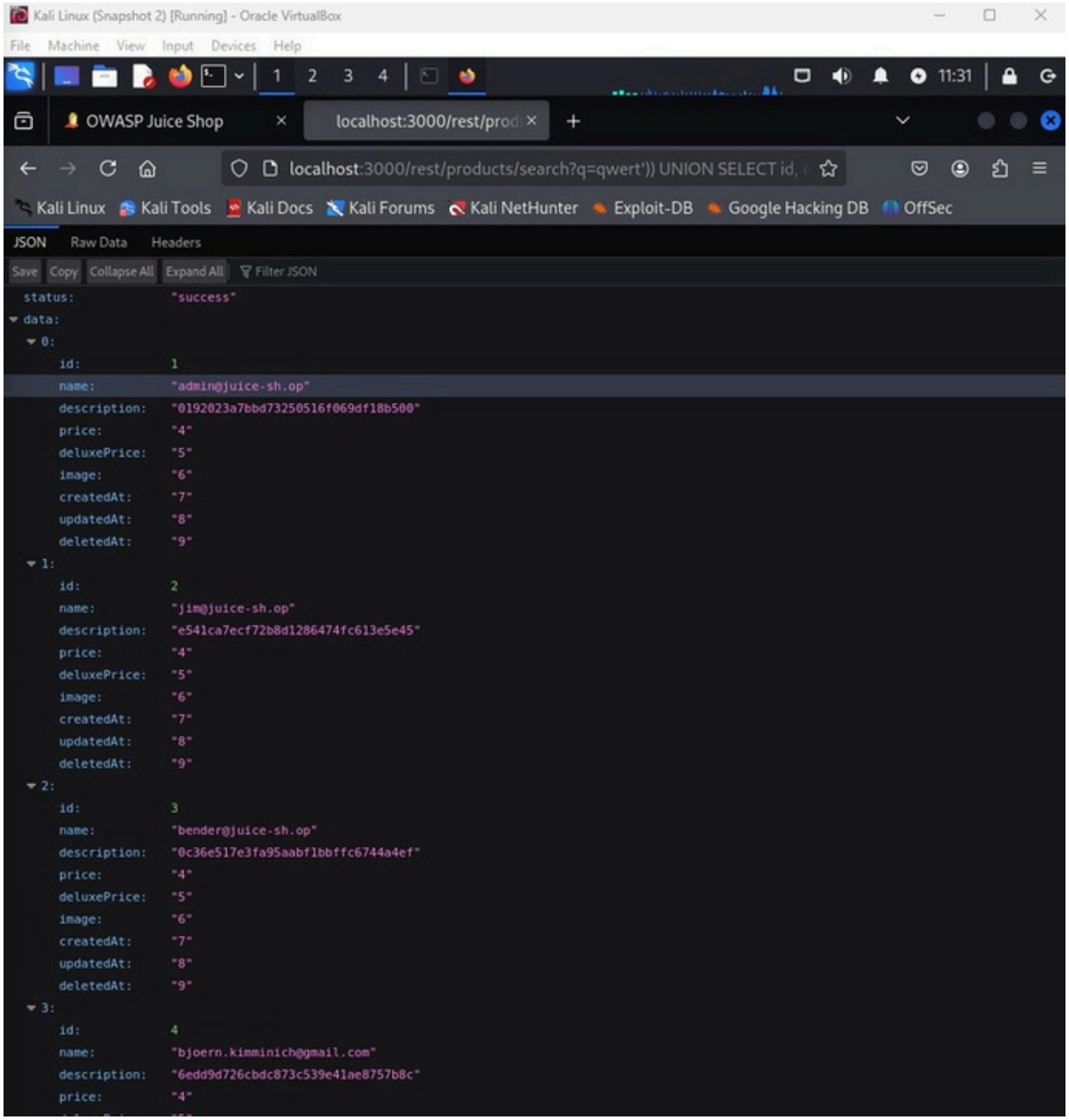


If you see this page after clicking Log in then you have successfully used sql injection to get admin access in owasp.

Now write the following in the search bar of Juice Shop:- UNION SELECT id, email, password, '4', '5', '6', '7', '8', '9' FROM Users–

OR type the following link in the search engine:-

http://localhost:3000/rest/products/search?q=qwert%27))%20UNION%20SELECT%20id,%20email,%20password,%20%274%27,%20%275%27,%20%276%27,%20%277%27,%20%278%27,%20%279%27%20FROM%20Users--



If this page shows up then you have successfully used SQL Injection to get the database containing all user details.

Mitigating SQL Injection Attacks

OWASP lists the following four methods to mitigate SQL Injection Attacks-

1) Use of Prepared Statements (with Parameterized Queries)-Prepared statements with parameterized queries are one of the most effective ways to prevent SQL injection. This technique involves creating a SQL query template where placeholders (such as ? or named parameters) are used for user inputs, rather than directly inserting user-supplied data into the query. The database engine then treats the user input as data, not executable code, preventing malicious input from altering the query's structure. This ensures that user input cannot inject arbitrary SQL code.

2) Use of Properly Constructed Stored Procedures- Stored procedures are precompiled SQL statements stored within the database. By using stored procedures, you can centralize logic on the server side, and avoid directly embedding user inputs into dynamic SQL queries. When constructed properly, stored procedures can protect against SQL injection by using parameters instead of concatenating user input directly into queries. However, it's important to avoid dynamic SQL within stored procedures, as it can still be vulnerable to injection if not handled carefully.

3) Allow-list Input Validation- Allow-list input validation (also known as whitelisting) involves checking user input against a set of predefined, acceptable values (such as alphanumeric characters or specific formats). This reduces the risk of malicious input being passed into SQL queries by ensuring that only expected, safe data is accepted. This technique doesn't entirely eliminate the risk of SQL injection but helps mitigate it by reducing the possibility of dangerous input.

4) STRONGLY DISCOURAGED: Escaping All User Supplied Input- Escaping user input involves adding escape characters to potentially dangerous characters (like single quotes or semicolons) to neutralize their threat. This was once a common method to prevent SQL injection but is now considered ineffective and error-prone. If you improperly escape input or fail to escape every possible special character, you leave your application vulnerable to injection. Modern database systems and frameworks support safer alternatives like prepared statements, so relying on escaping input is discouraged.

While it can be effective when done carefully, escaping input does not provide the same level of safety as other methods like parameterized queries, and it’s much harder to get right. Therefore, relying on it alone can be risky.

Q7) Framework Mapping: Map a given security incident to the NIST Cybersecurity Framework or Cyber Kill Chain. Prepare a simple report outlining the stages of the attack.

Practical:-

Cybersecurity Incident Overview- Domino's India Data Breach of 2021

On March 24, 2021, one of the most popular pizza chains of the world **Domino's Pizza** suffered a massive cyber attack in India where 18 crore order details were stolen and put up for sale on the dark web by hackers. The leaked data included sensitive information like name, mobile number, email address, and GPS location of users.

Mapping to NIST Cybersecurity Framework

1) Identify (Risk Management, Asset Management)- Lack of strong data security policies for stored customer data, inadequate third-party security audits.

Evidence: Domino's India acknowledged a security incident but did not provide specifics about their data security policies or audit practices.

2) Protect (Data Security, Access Control)- Possible weak database security (misconfigurations, unencrypted data), inadequate API security controls.

Evidence: While the breach involved unauthorized access to customer data, the exact vulnerabilities exploited (e.g., database misconfigurations or unsecured APIs) were not publicly disclosed.

3) Detect (Security Monitoring, Anomalies & Events)- The breach was discovered externally (on the dark web), indicating lack of real-time breach detection.

Evidence: The breach became public when hackers advertised the stolen data on the dark web, suggesting that Domino's India may not have detected the intrusion internally.

4) Respond Function (Incident Analysis, Mitigation)- Engaged global forensic experts post-breach, lodged cybercrime complaints, public disclosure was delayed.

Evidence: Domino's India stated they were investigating the incident with the help of experts and had reported the matter to authorities.

5) Recover Function (Recovery Planning, Security Improvements)- Strengthened security infrastructure, enhanced data access controls, and collaborated with cybersecurity professionals.

Evidence: Specific post-breach security enhancements were not detailed in public statements.

Conclusion

In conclusion, while certain aspects of the breach were documented, specific technical details about the vulnerabilities exploited and the organization's internal detection capabilities have not been publicly disclosed.