YACHA VENKATA RAKESH
B180427CS
CSE - A Batch

NETWORKS LAB EXP - 7

1. Update system repositories
   Firstly, we need to update the local package index of Ubuntu repositories.
   $ sudo apt-get update

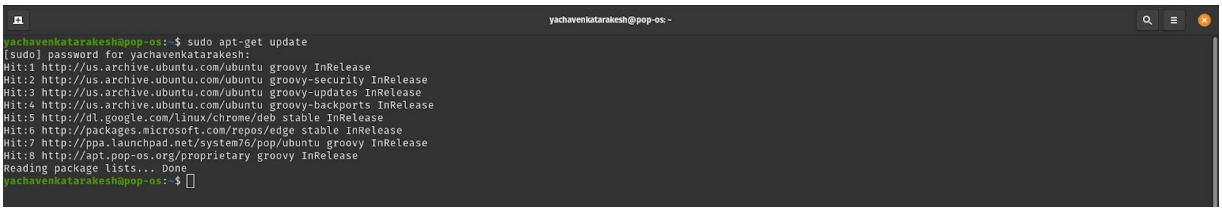

FIG 1. Updating system repositories

2. Install Apache2
   Install apache2 and its required dependencies.
   $ sudo apt-get purge apache2
   This is to remove configuration files of the package if you have related to that package and so to have fresh installation of the package
   $ sudo apt-get install apache2

```
sudo apt-get purge apache2
```

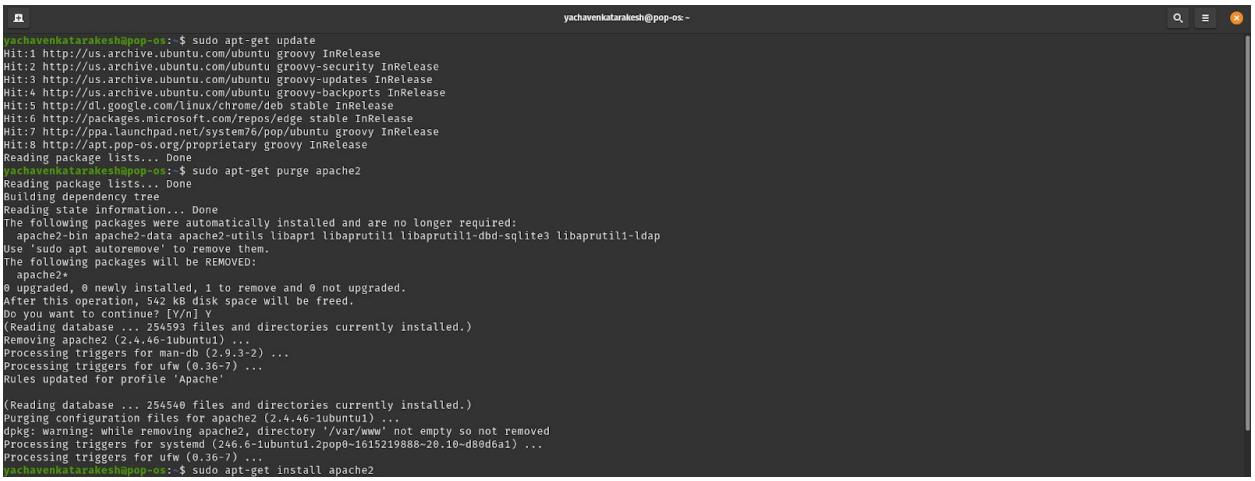

FIG 2. Removing any previous traces of Apache server

```
yachavenkatarakesh@pop-os: ~
Processing triggers for ufw (0.36-7) ...
yachavenkatarakesh@pop-os:~$ sudo apt-get install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom
The following NEW packages will be installed:
  apache2
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 0 B/95.6 kB of archives.
After this operation, 542 kB of additional disk space will be used.
Selecting previously unselected package apache2.
(Reading database ... 254375 files and directories currently installed.)
Preparing to unpack .../apache2_2.4.46-1ubuntu1_amd64.deb ...
Unpacking apache2 (2.4.46-1ubuntu1) ...
Setting up apache2 (2.4.46-1ubuntu1) ...
Enabling module mpm_event.
Enabling module authz_core.
Enabling module authz_host.
Enabling module authn_core.
Enabling module auth_basic.
Enabling module access_compat.
Enabling module authn_file.
Enabling module authz_user.
Enabling module alias.
Enabling module dir.
Enabling module autoindex.
Enabling module env.
Enabling module mime.
Enabling module negotiation.
Enabling module setenvif.
Enabling module filter.
Enabling module deflate.
Enabling module status.
Enabling module reqtimeout.
Enabling conf charset.
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service → /lib/systemd/system/apache-htcacheclean.service.
Processing triggers for systemd (246.6-1ubuntu1.2pop0~1615219888~20.10~d80d6a1) ...
Processing triggers for man-db (2.9.3-2) ...
Processing triggers for ufw (0.36-7) ...
Rules updated for profile 'Apache'

yachavenkatarakesh@pop-os:~$
```

FIG3. Installing Apache server

3. Verify the Apache installation
   We can check the version number of the apache2 and thus verify if it's installed correctly
   $ apache2 -version

```
yachavenkatarakesh@pop-os:~$ apache2 -version
Server version: Apache/2.4.46 (Ubuntu)
Server built:   2020-08-25T12:13:38
```

FIG4. Checking apache version

This can also be checked by typing http://localhost and if it results in Apache2 default page

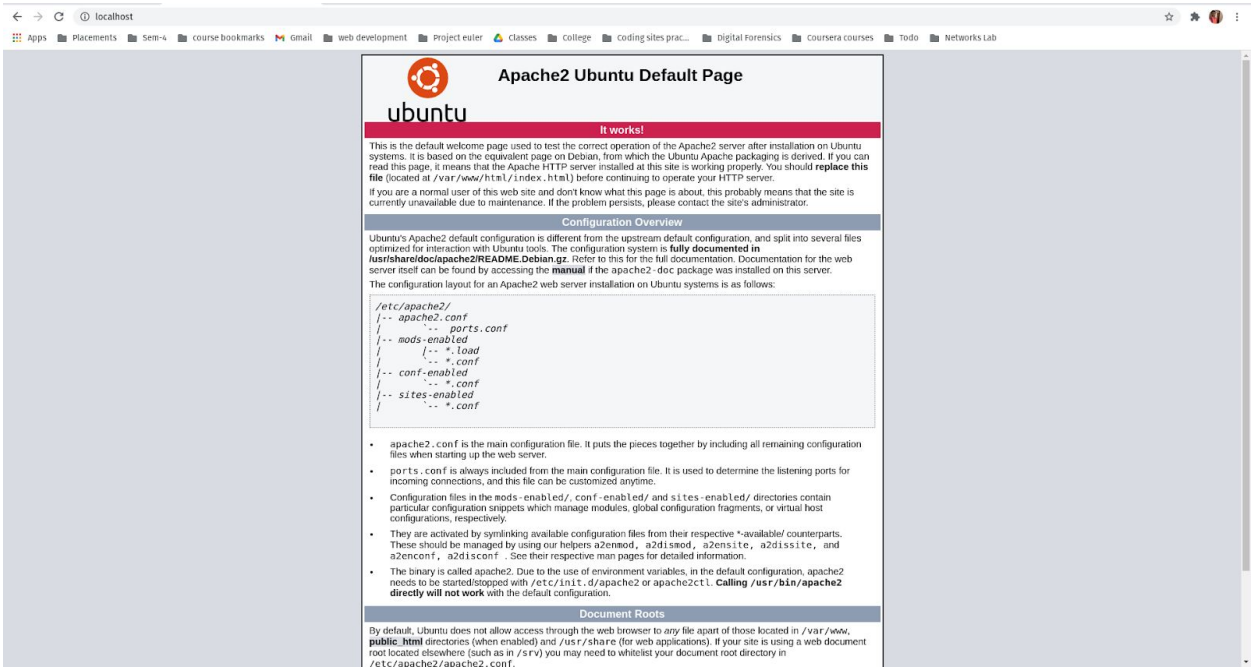

FIG5. Apache running on localhost

4. List the UFW application profiles
   So we now have to configure the firewall settings to allow outside access to certain web ports of our system. Now first list the application profiles we will need to enable access to Apache.
   $ sudo ufw app list

```
yachavenkatarakesh@pop-os:~$ sudo ufw app list
Available applications:
  Apache
  Apache Full
  Apache Secure
  CUPS
```

FIG6. UFW application profiles list showing different apache profiles

   Here we could see three apache profiles all providing different levels of security and Apache is one of those that provides maximum restriction with port 80 still open

5. Allow Apache on UFW and verify its status
   Allowing Apache on UFW will open port 80 for network traffic, while providing maximum security for the server.
   $ sudo ufw allow ‘Apache’

```
yachavenkatarakesh@pop-os:~$ sudo ufw allow 'Apache'
Skipping adding existing rule
Skipping adding existing rule (v6)
yachavenkatarakesh@pop-os:~$ sudo ufw status
Status: inactive
yachavenkatarakesh@pop-os:~$ sudo ufw enable
Firewall is active and enabled on system startup
yachavenkatarakesh@pop-os:~$ sudo ufw default deny
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
yachavenkatarakesh@pop-os:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-logging-input  all  --  anywhere             anywhere
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere             anywhere
ufw-after-logging-input  all  --  anywhere             anywhere
ufw-reject-input  all  --  anywhere             anywhere
ufw-track-input  all  --  anywhere             anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination
ufw-before-logging-forward  all  --  anywhere             anywhere
ufw-before-forward  all  --  anywhere             anywhere
ufw-after-forward  all  --  anywhere             anywhere
ufw-after-logging-forward  all  --  anywhere             anywhere
ufw-reject-forward  all  --  anywhere             anywhere
ufw-track-forward  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ufw-before-logging-output  all  --  anywhere             anywhere
ufw-before-output  all  --  anywhere             anywhere
ufw-after-output  all  --  anywhere             anywhere
ufw-after-logging-output  all  --  anywhere             anywhere
ufw-reject-output  all  --  anywhere             anywhere
ufw-track-output  all  --  anywhere             anywhere
```

FIG7. Enabling firewall and entering apache as a whitelist

The status of UFW will now display Apache enabled on the firewall
$ sudo ufw status
If it displays status as inactive then it means that firewall is not enabled. So first enable the firewall and change it to default and check iptables list
$ sudo ufw enable
$ sudo ufw default deny
$ sudo iptables -L
Now $ sudo ufw allow 'Apache'
$ sudo ufw status

```
yachavenkatarakesh@pop-os:~$ sudo ufw allow 'Apache'
Skipping adding existing rule
Skipping adding existing rule (v6)
yachavenkatarakesh@pop-os:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
Apache                     ALLOW       Anywhere
Apache (v6)                ALLOW       Anywhere (v6)
```

FIG8. Firewall status and allowance for apache

6. Verify if the Apache service is running
   $ sudo systemctl status apache2

```
yachavenkatarakesh@pop-os:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2021-03-13 19:50:46 IST; 10min ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 13040 (apache2)
      Tasks: 55 (limit: 4571)
     Memory: 7.8M
     CGroup: /system.slice/apache2.service
             ├─13040 /usr/sbin/apache2 -k start
             ├─13041 /usr/sbin/apache2 -k start
             └─13042 /usr/sbin/apache2 -k start

Mar 13 19:50:45 pop-os systemd[1]: Starting The Apache HTTP Server...
Mar 13 19:50:46 pop-os systemd[1]: Started The Apache HTTP Server.
```

FIG9. Verify the status of apache

   The Active : active (running) verifies that the apache2 service is running.
7. Verify if the Apache is running properly and listening on your IP Address.
   We can verify by requesting a page from the Apache server. For that we can use server's IP and it returns default webpage if successful.
   $ hostname -I
   To know the IP address in which server's IP address.
   http://server_IP

FIG11. Apache running on the system's IP address

8. Set up a domain name
   Now it's time to set up virtual hosts in Apache. As per the assignment given we need to configure two virtual hosts. Let's name it virtualhost1.com, virtualhost2.com. So we need to first set up these domain names.
   $ sudo mkdir -p /var/www/virtualhost1.com/html
   $ sudo mkdir -p /var/www/virtualhost2.com/html

   Now assign the ownership of the directory to the present user

   $ sudo chown -R $USER:$USER /var/www/virtualhost1.com/html
   $ sudo chown -R $USER:$USER /var/www/virtualhost2.com/html

   We should also modify our permissions a little bit to ensure that read access is permitted by the general web directory so that required files can be served.

   $ sudo chmod -R 755 /var/www/

9. Create a web page for each of the virtual hosts
   $ sudo nano /var/www/virtualhost1.com/html/index.html

   For virtual host 1

```
<html>
        <head>
                <title>Networks Lab Experiment 7</title>
        </head>
        <body>
                <h1>Welcome to Networks Lab!</h1>
                <h2>You have accessed virtual host 1</h2>
        </body>
</html>
```

$ sudo nano /var/www/virtualhost2.com/html/index.html

For virtual host2

```
<html>
        <head>
                <title>Networks Lab Experiment 7</title>
        </head>
        <body>
                <h1>Welcome to Networks Lab!</h1>
                <h2>You have accessed virtual host 2</h2>
        </body>
</html>
```

10. Create New virtual host file configurations
    By binding IP_address and Port_No. Along with name server and other server details.

    $ sudo nano /etc/apache2/sites-available/virtualhost1.com.conf

    ```
    <VirtualHost 127.0.1.2:80>
    ServerAdmin admin@virtualhost1.com
    ServerName virtualhost1.com
    ServerAlias www.virtualhost1.com
    DocumentRoot /var/www/virtualhost1.com/html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
    </VirtualHost>
    ```

    $ sudo nano /etc/apache2/sites-available/virtualhost2.com.conf

    ```
    <VirtualHost 127.0.1.2:80>
    ServerAdmin admin@virtualhost2.com
    ServerName virtualhost2.com
    ServerAlias www.virtualhost2.com
    ```

DocumentRoot /var/www/virtualhost2.com/html
ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>

11. Enable the domain configuration file
    $ sudo a2ensite virtualhost1.com.conf
    $ sudo a2ensite virtualhost1.com.conf
    $ sudo a2dissite 000-default.conf
    $ sudo systemctl restart apache2

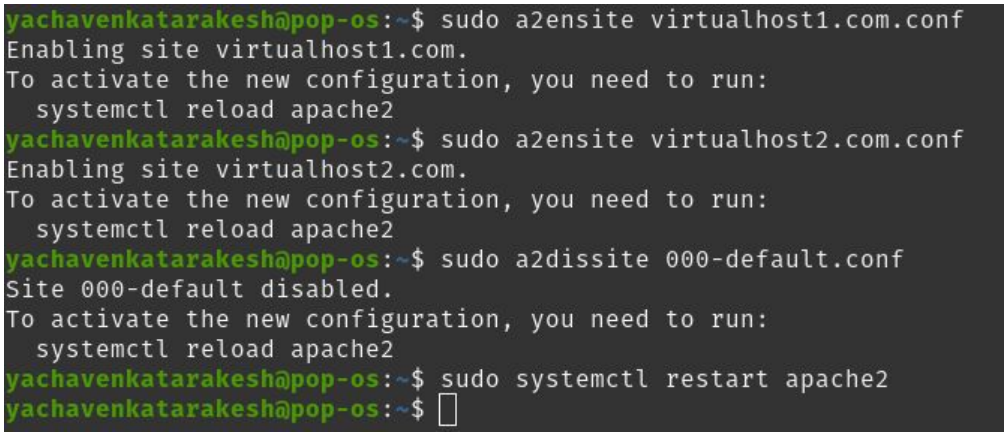

FIG12. Enabling domain configuration files

12. Set up local host file
    $ sudo nano /etc/hosts
    Enter the IP address and domain names of virtual hosts and save it using ctrl+x

FIG13. Updating hosts with virtual host's IP address

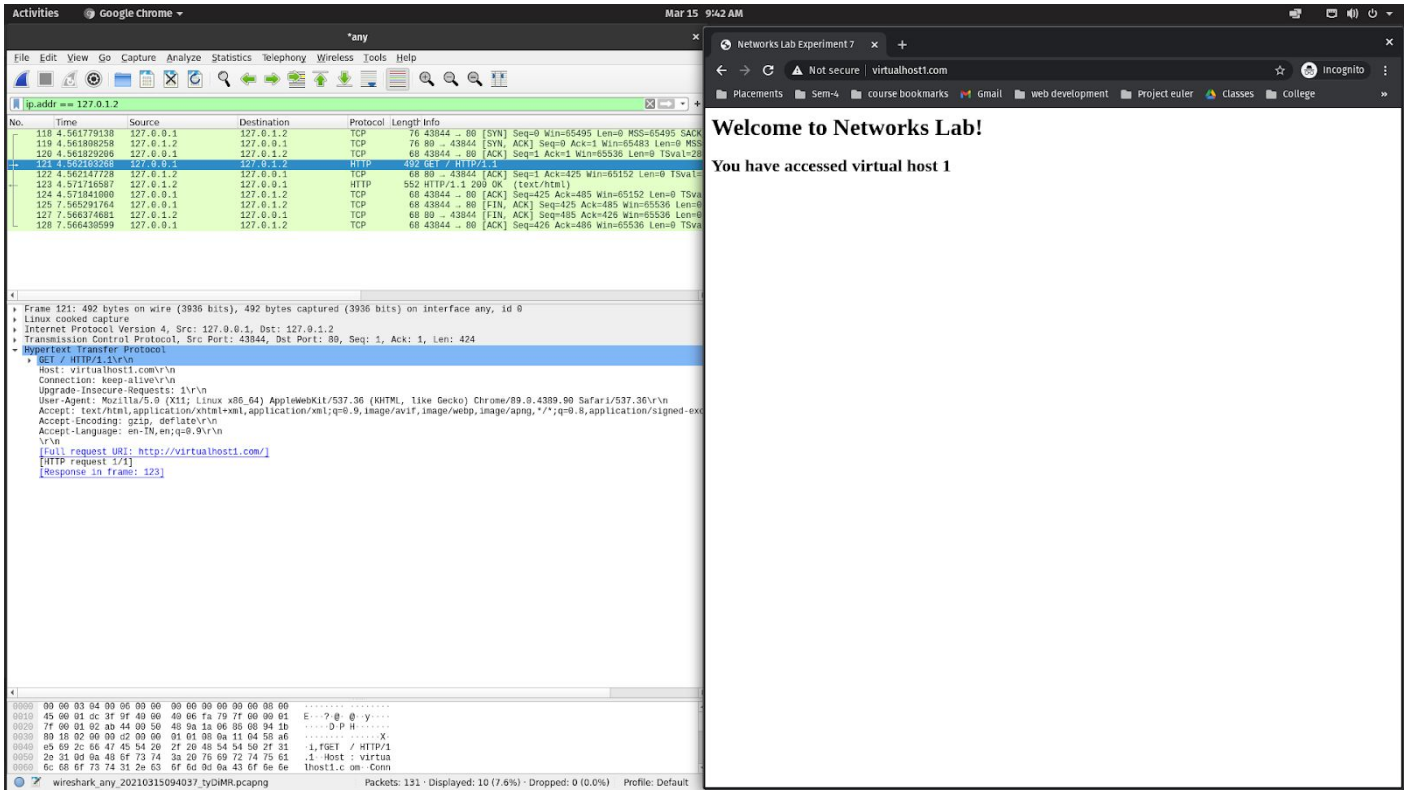

FIG14. HTTP request message captured using wireshark for virtual host 1(can be seen under host)

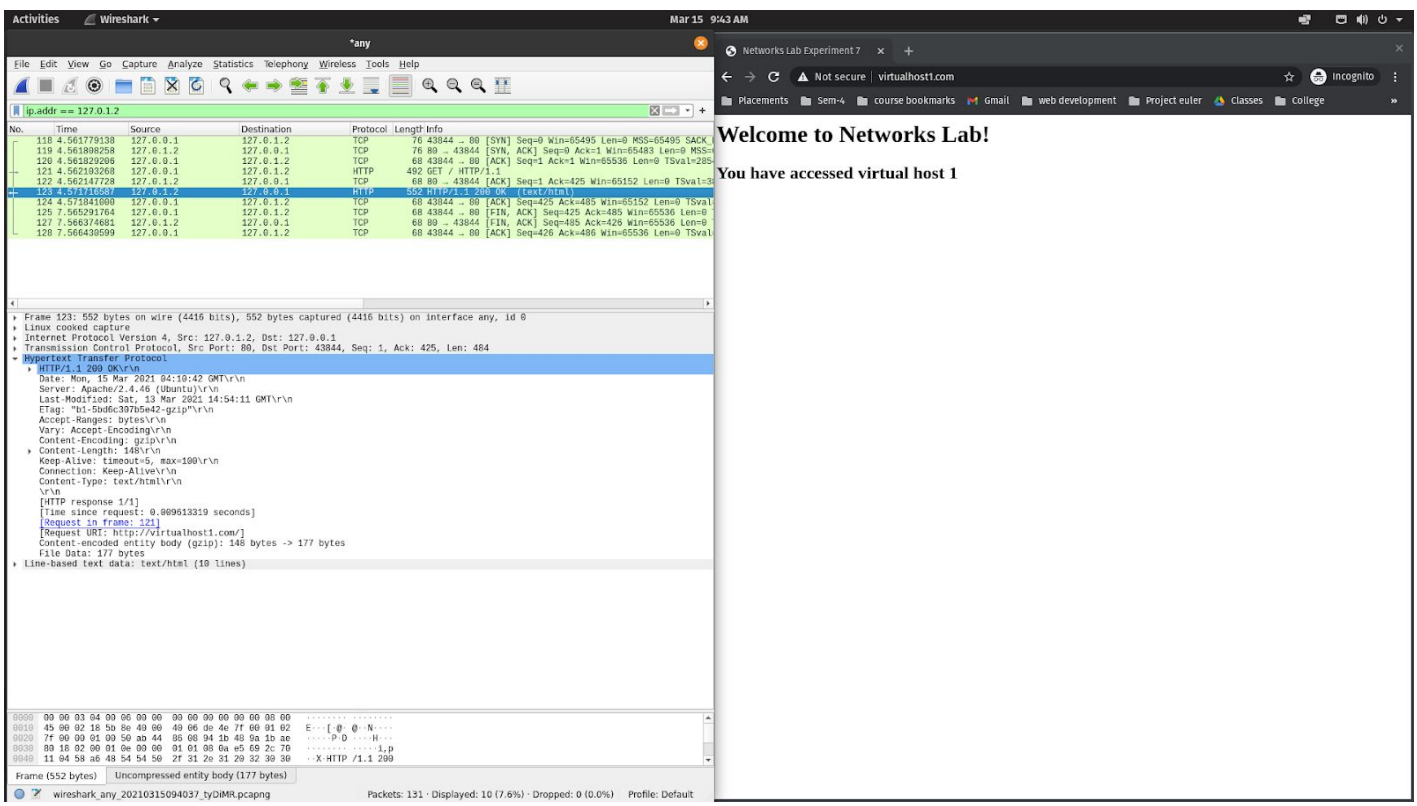

FIG15. HTTP response message captured using wireshark for virtual host1(can be seen under host)

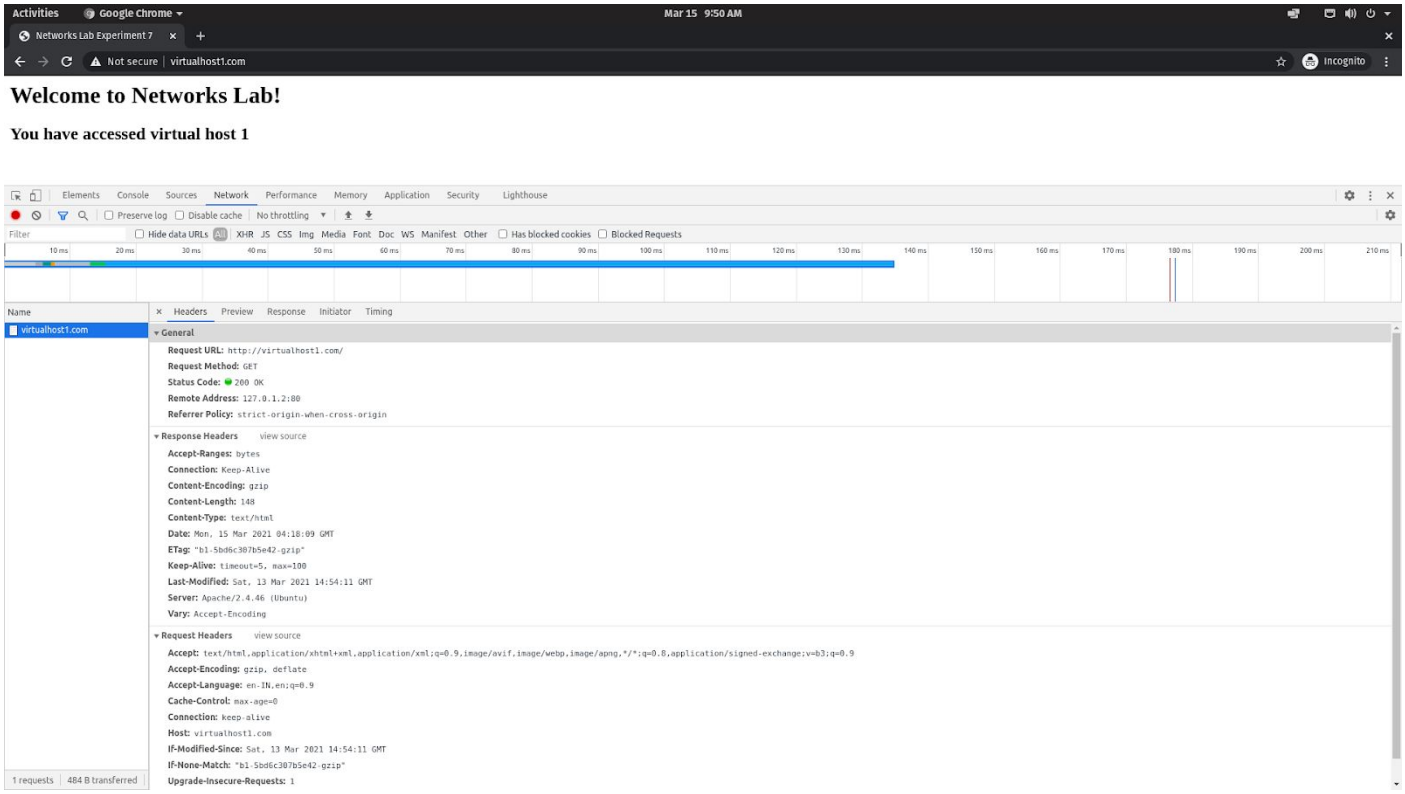

FIG16. HTTP headers captured using chrome developer tools for virtualhost1

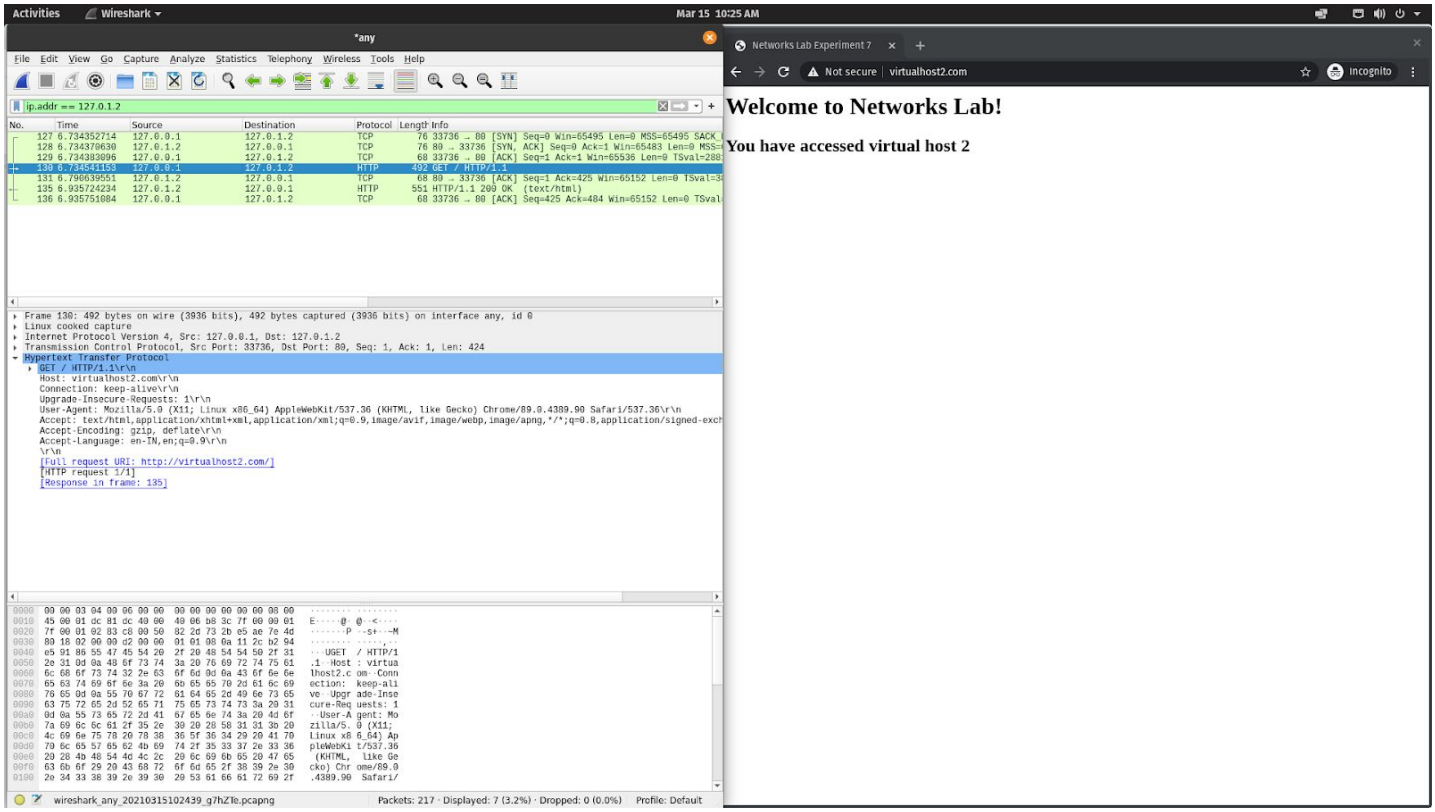

FIG17. HTTP request message captured using wireshark for virtual host 2(can be seen under host)

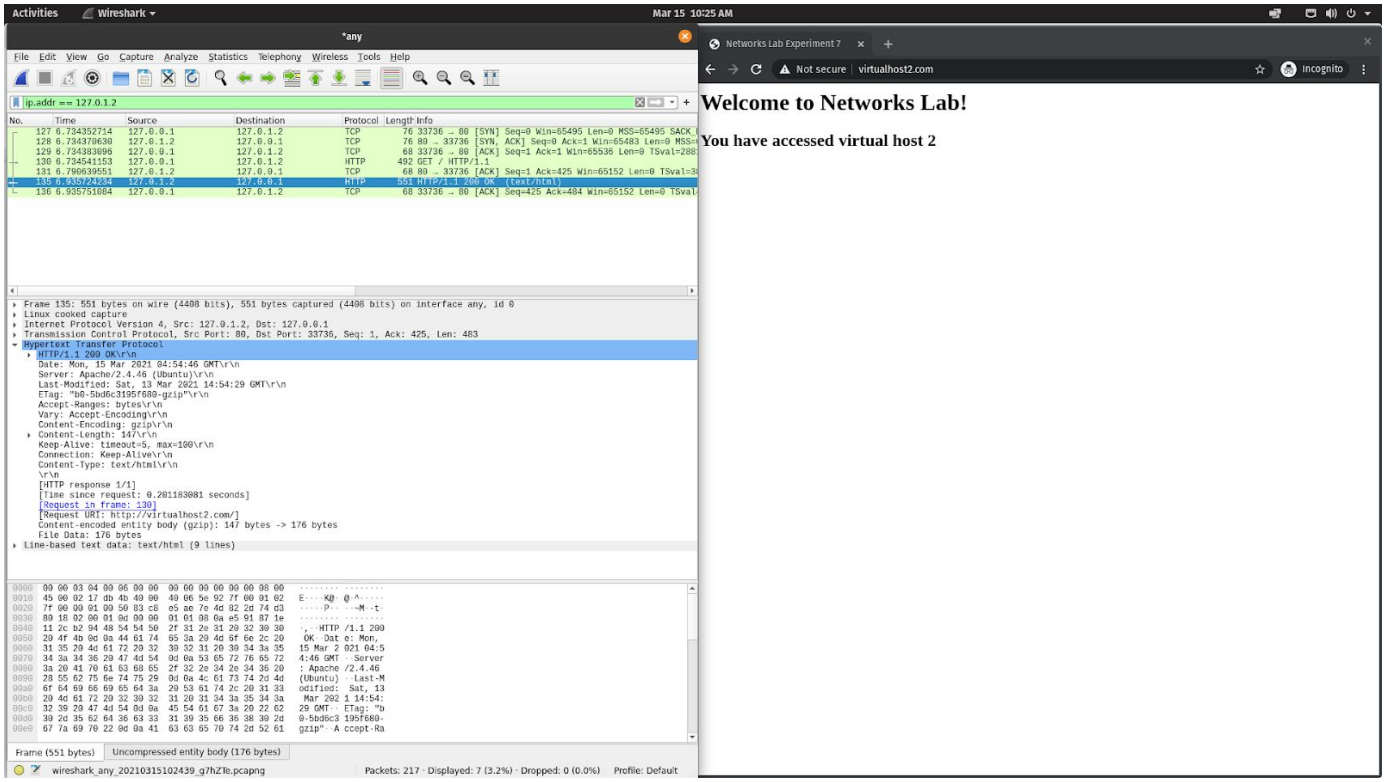

FIG18. HTTP response message captured using wireshark for virtual host2(can be seen under host)

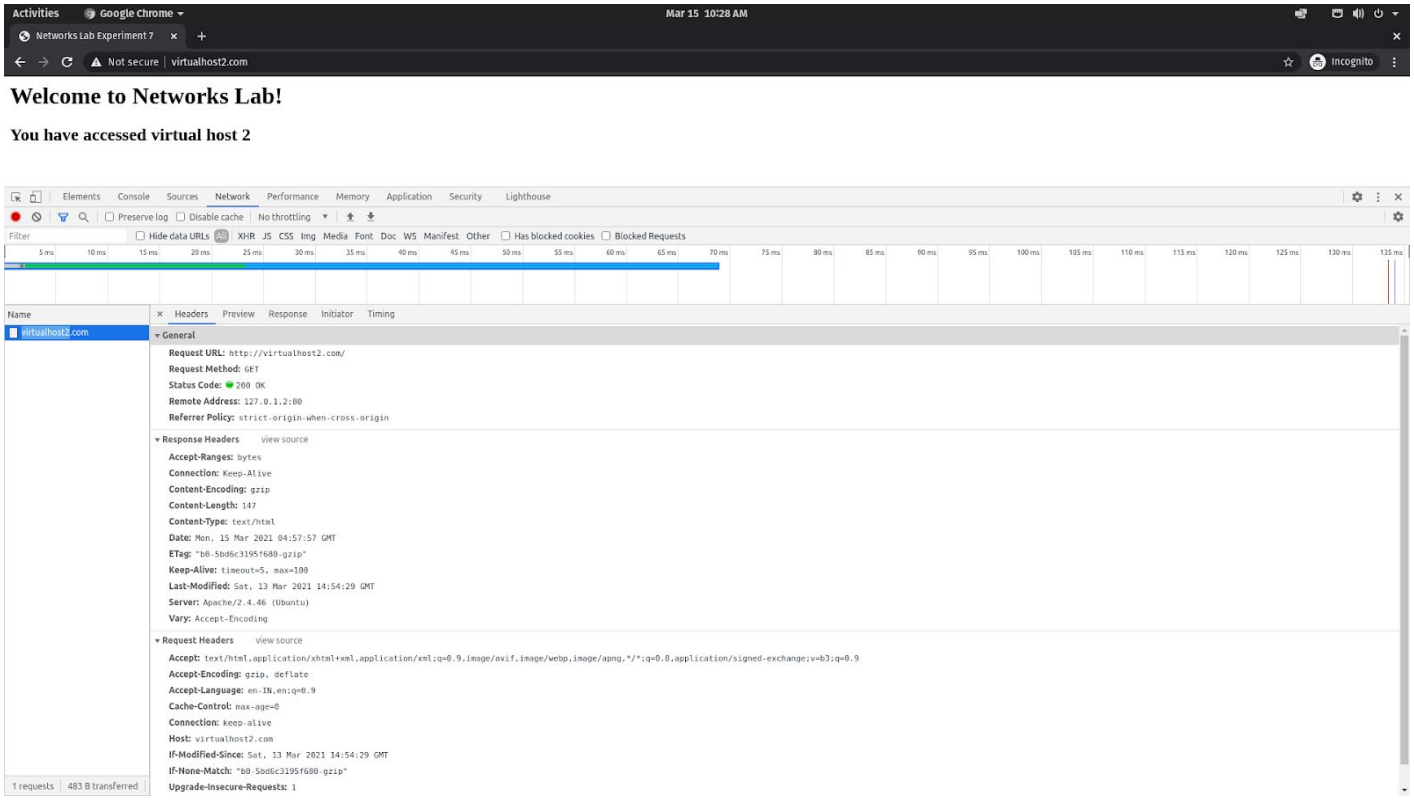

FIG19. HTTP headers captured using chrome developer tools for virtualhost2

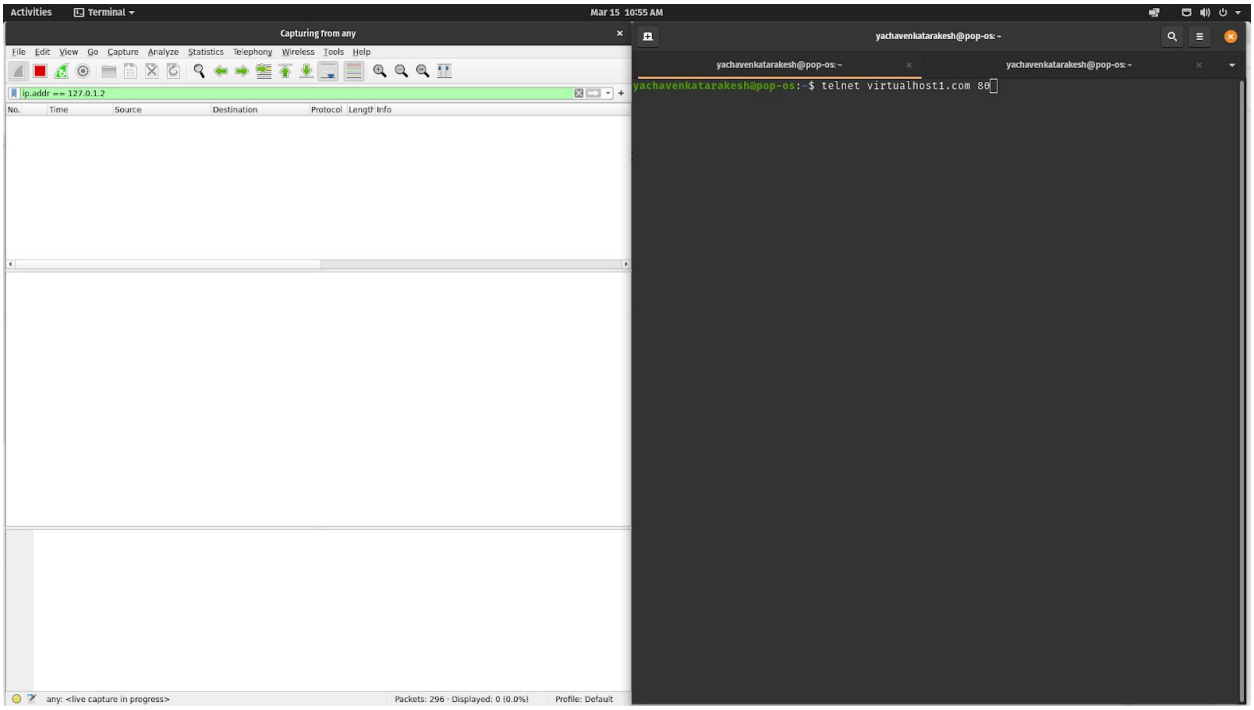

FIG20. Initially wireshark interface and terminal

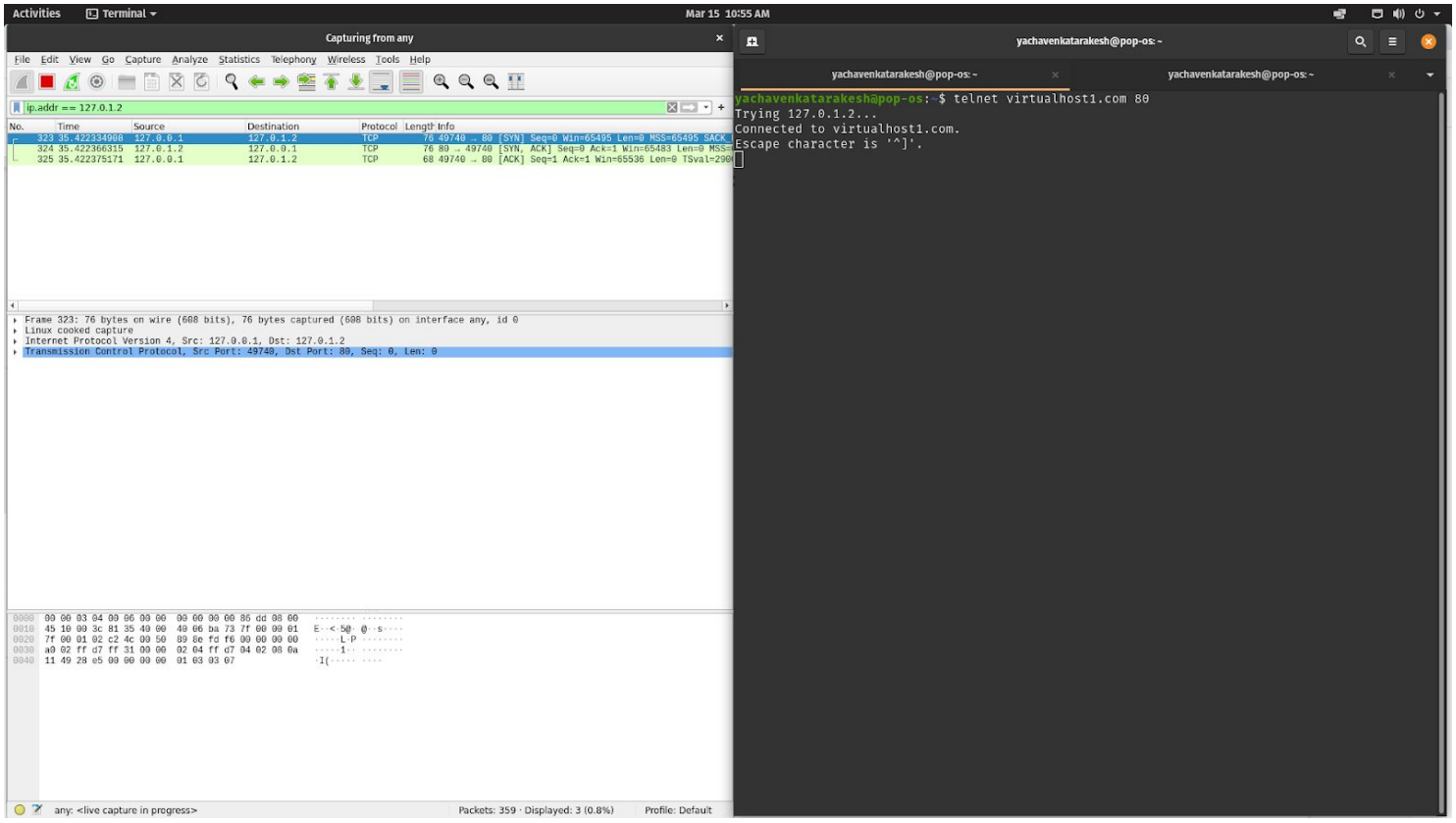

FIG21. Connection established by TCP 3-way handshake for virtualhost1

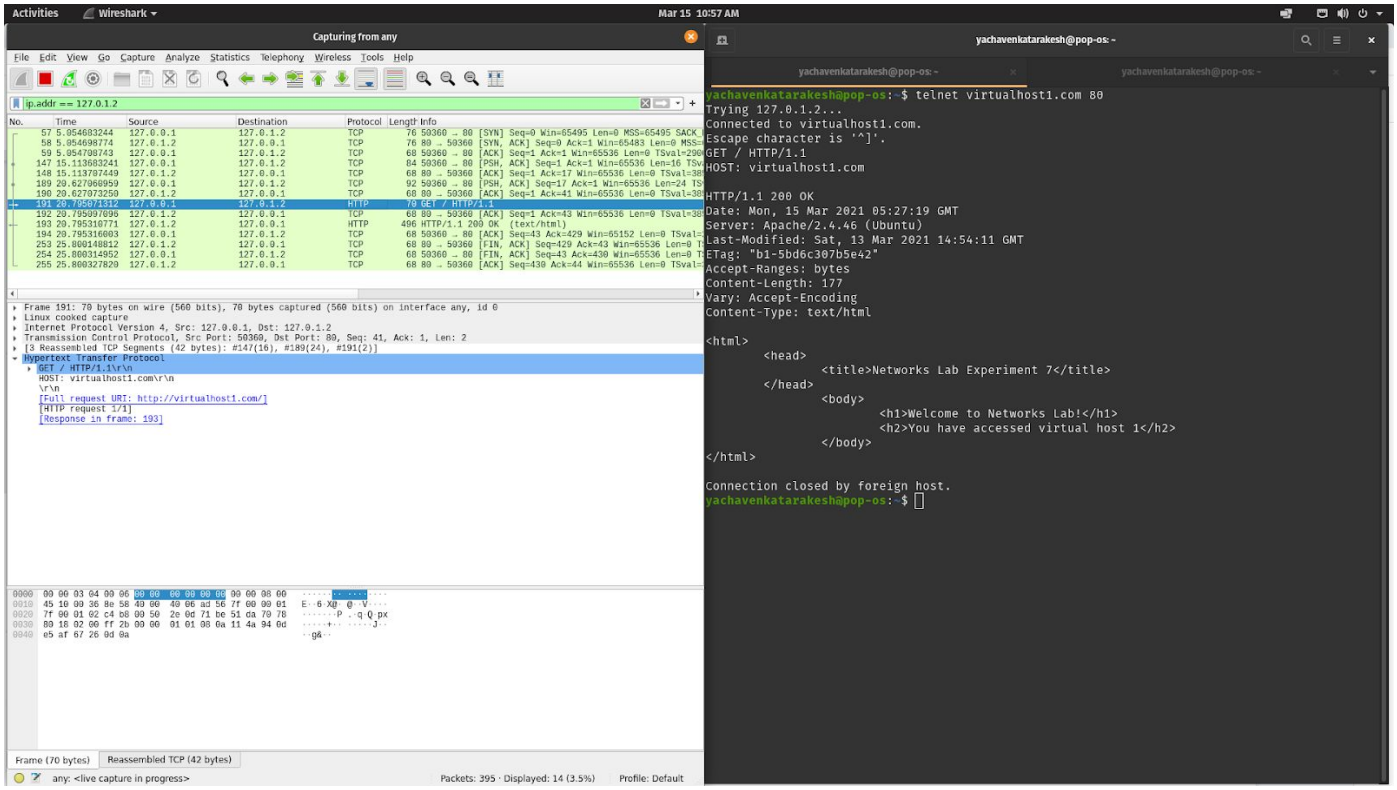

FIG22. HTTP Request for virtualhost1 using telnet and wireshark

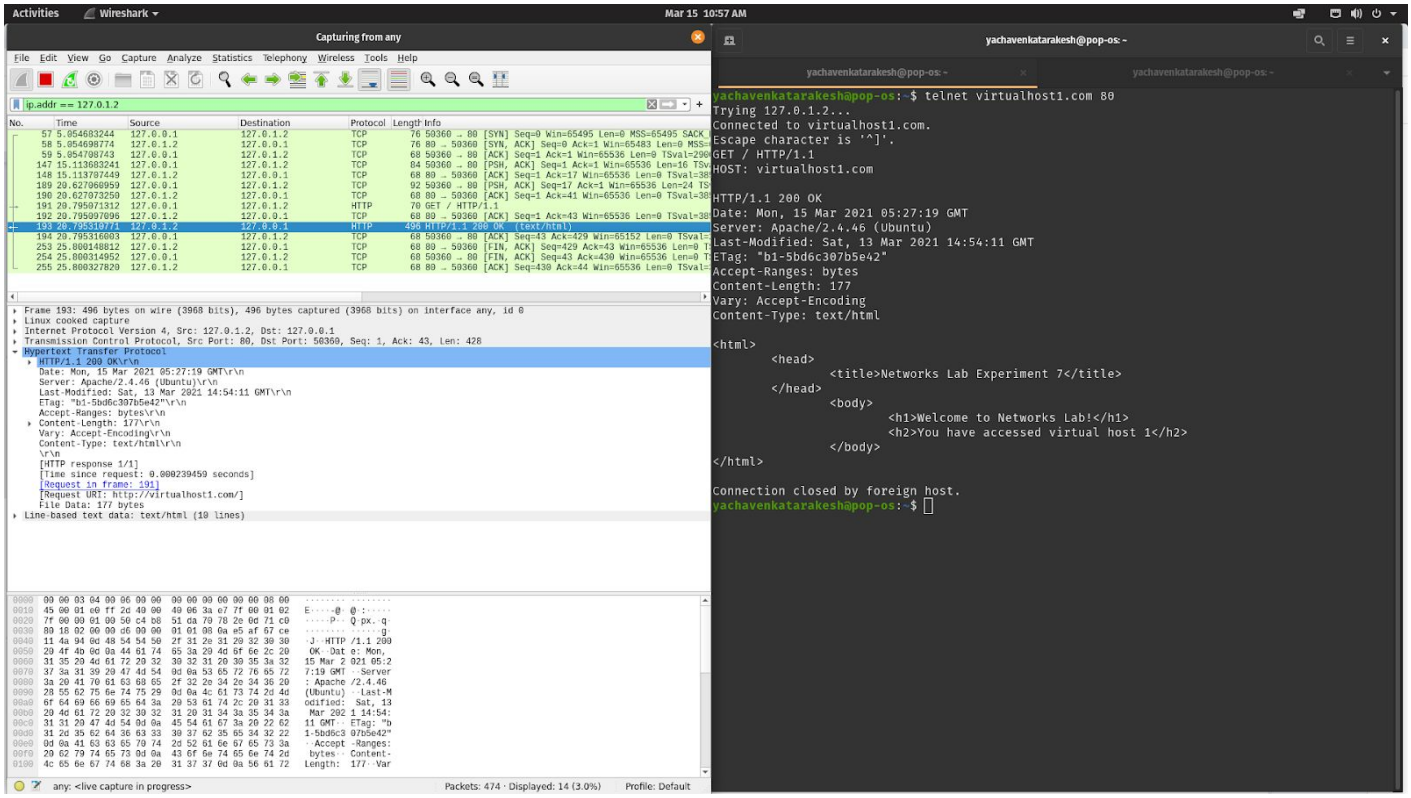

FIG23. HTTP response for virtualhost1 using telnet and wireshark

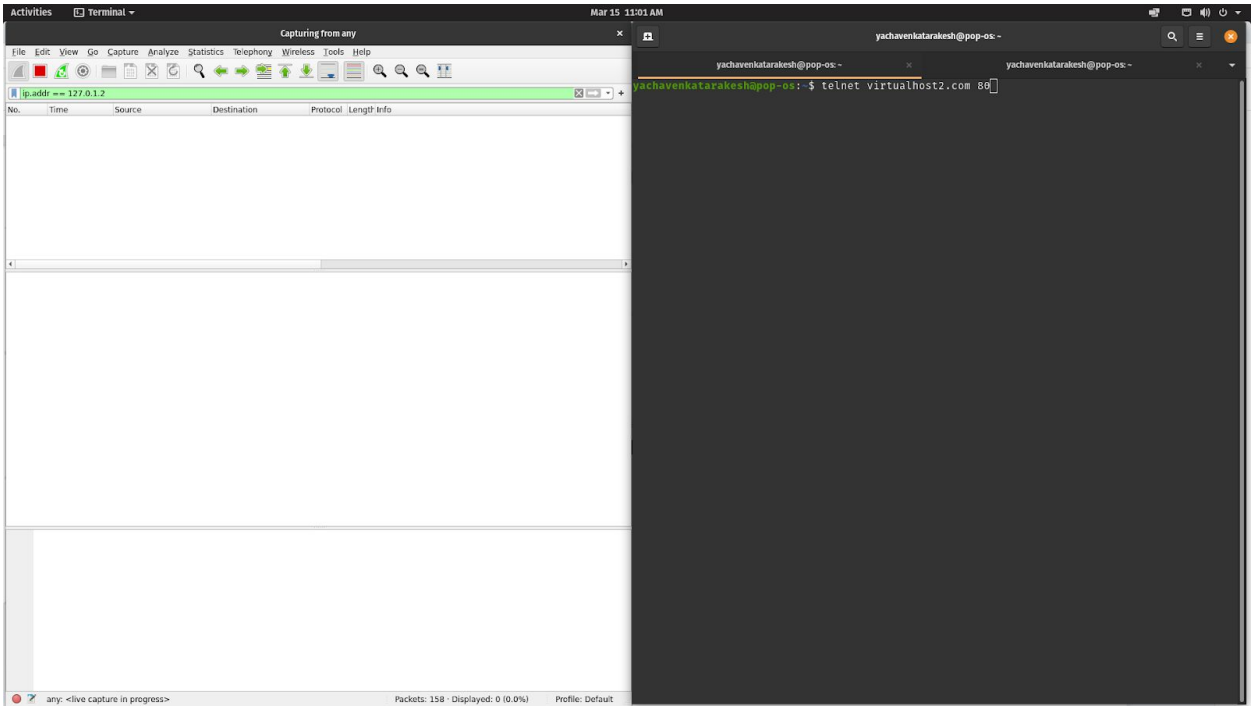

FIG24. Initially wireshark interface and terminal

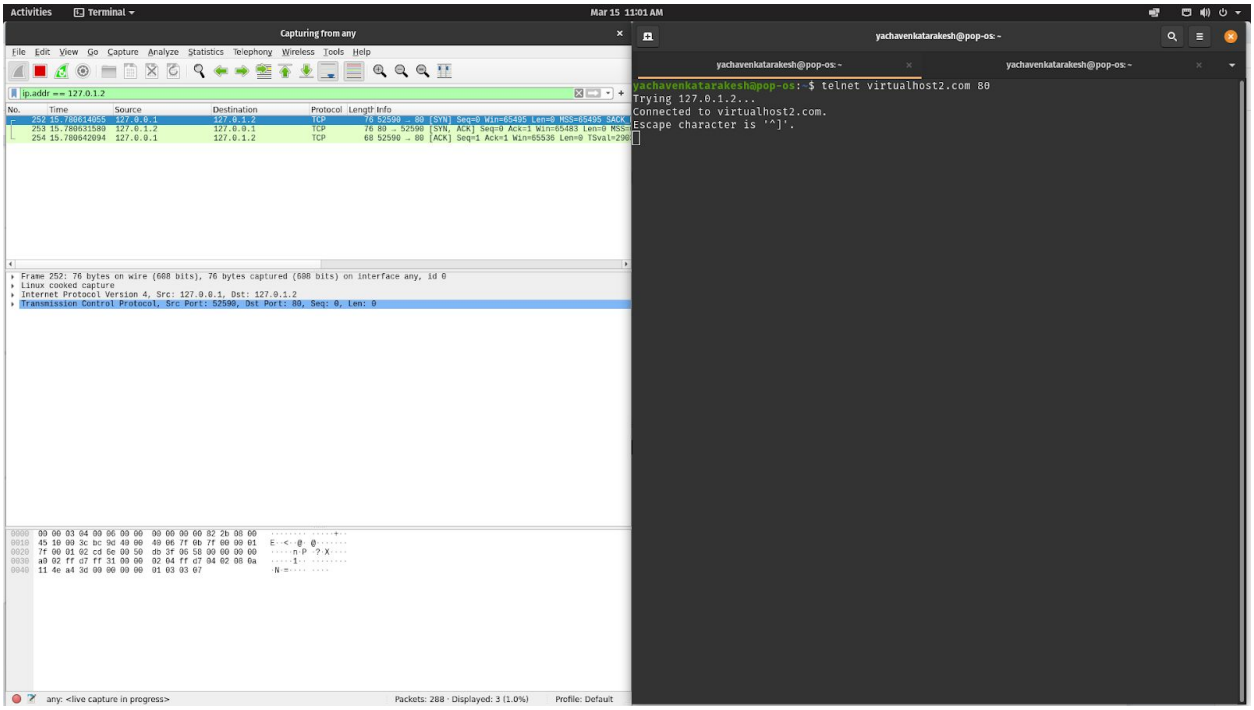

FIG25. Connection established by TCP 3-way handshake for virtualhost2

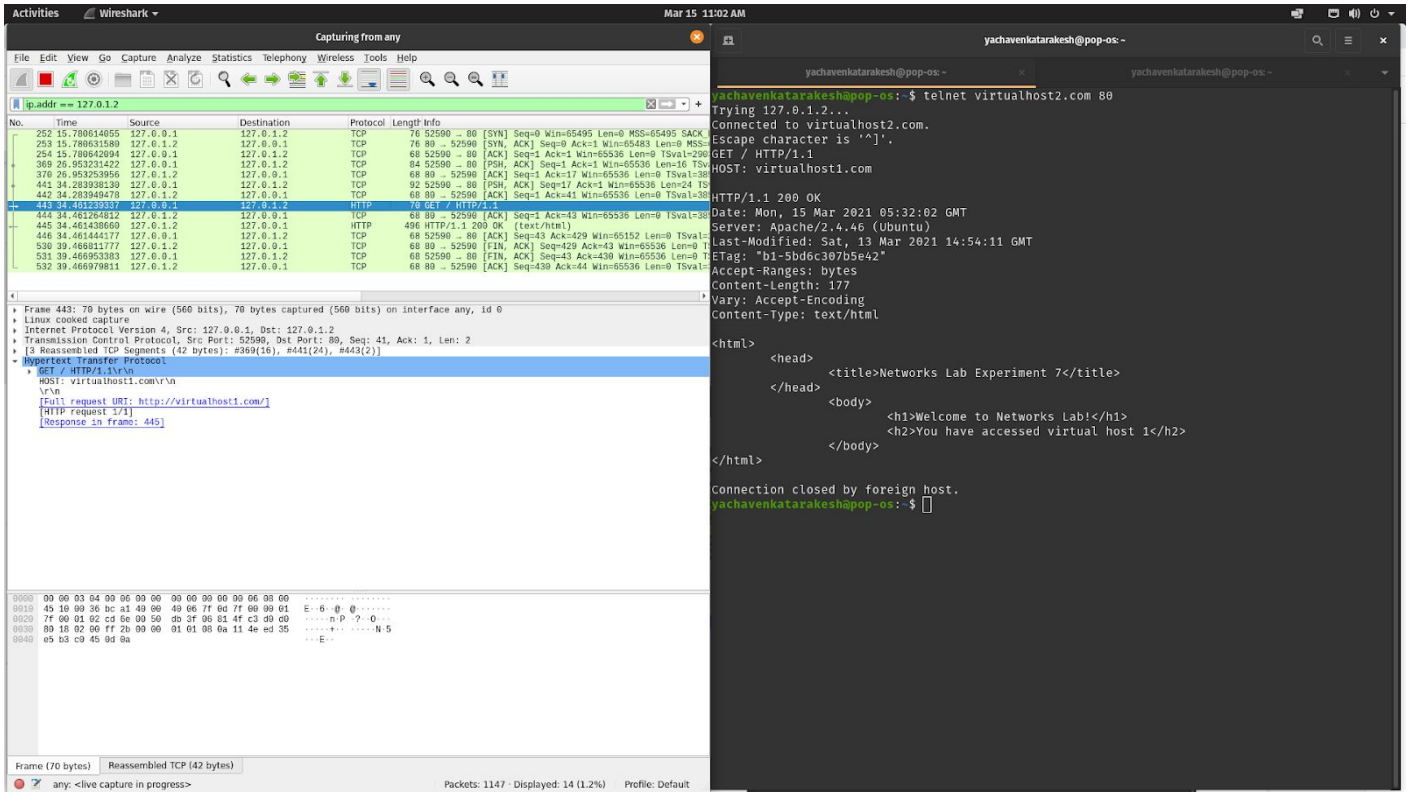

FIG26. HTTP request for virtualhost2 using telnet and wireshark

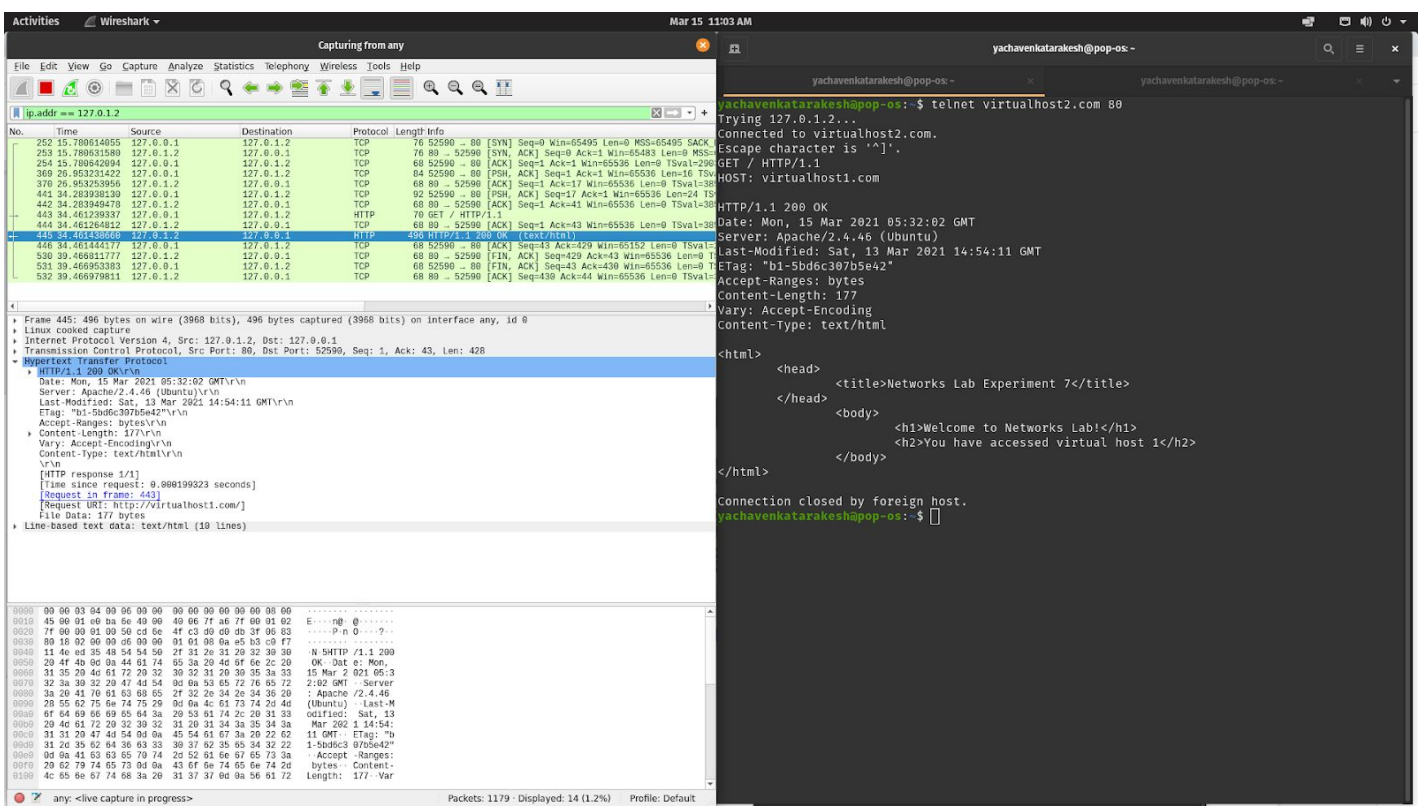

FIG27. HTTP response for virtualhost2 using telnet and wireshark

QUESTION 2

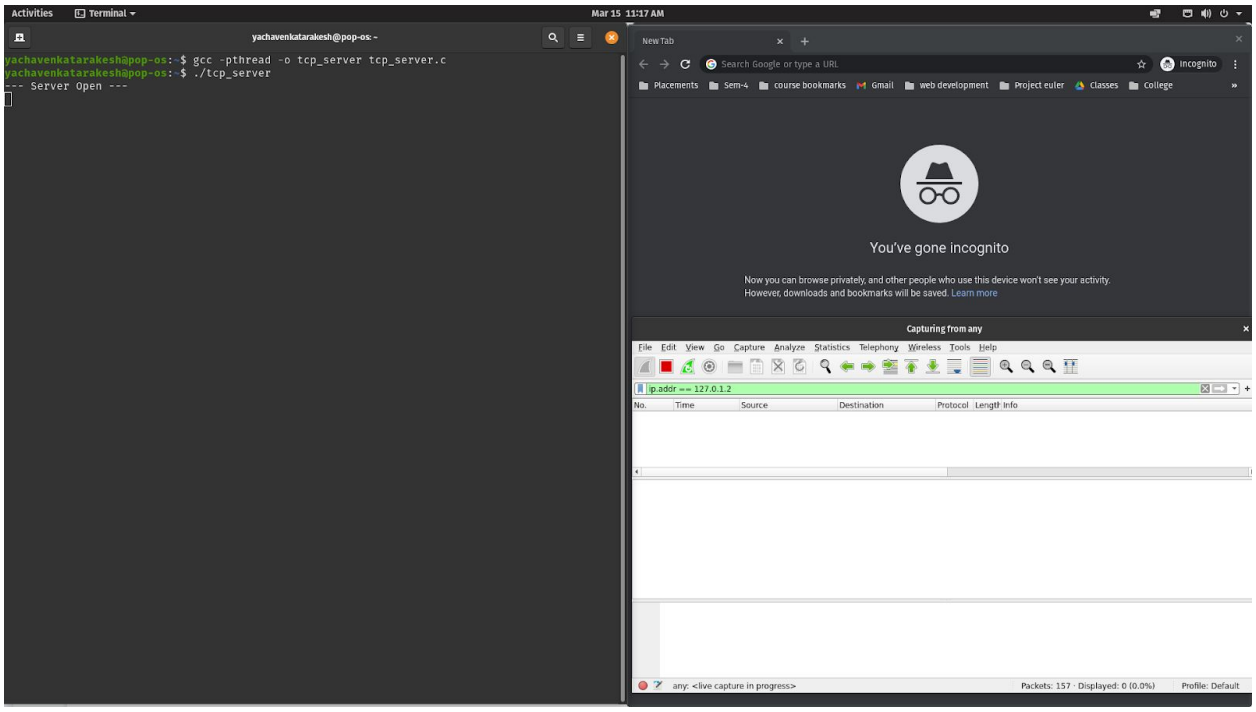

FIG28. TCP web server started

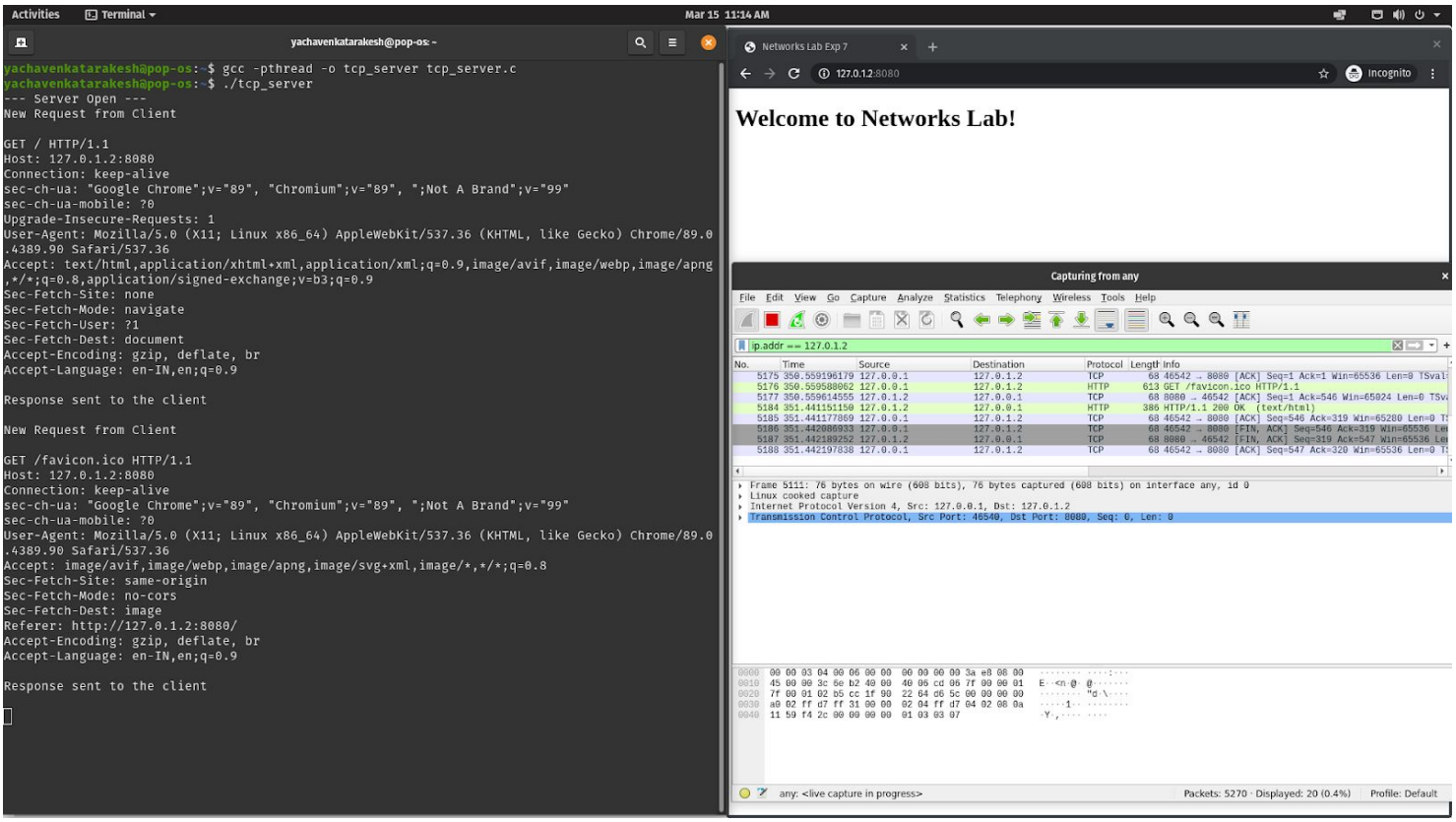

FIG29. On Entering the webpage 127.0.1.2:8080

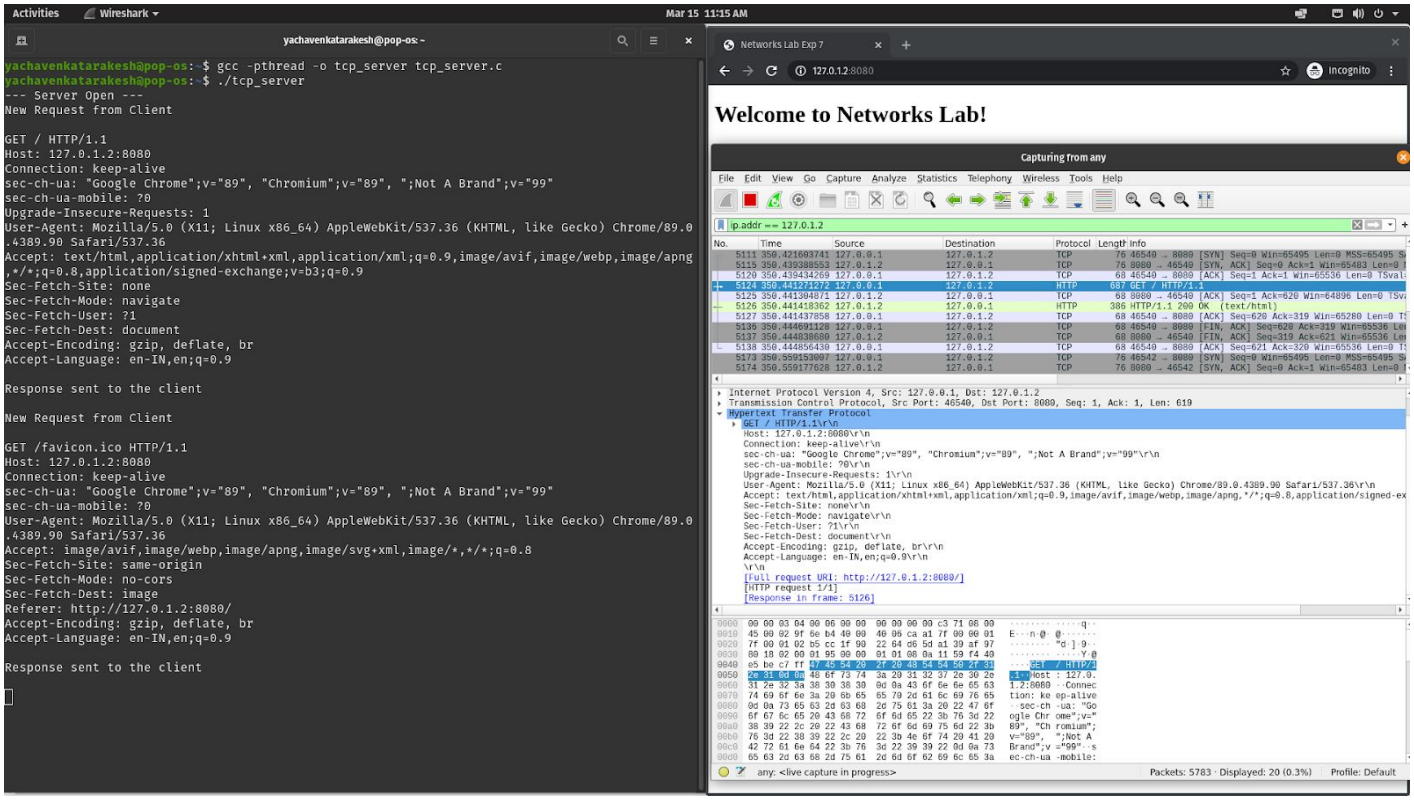

FIG30. HTTP Request captured using wireshark and received message display in terminal

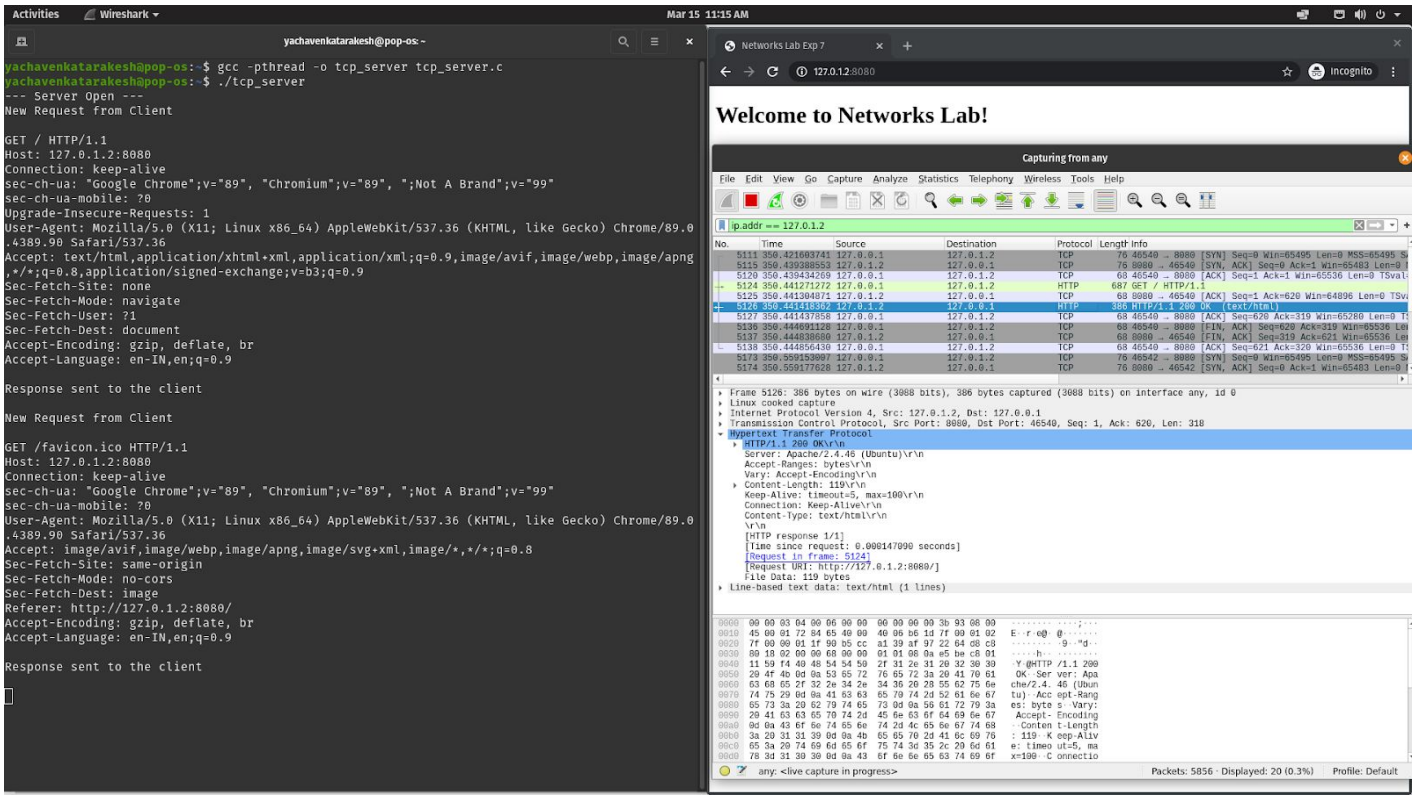

FIG31. HTTP Response captured using wireshark and response sent message in terminal