**Networking and Servers**

<https://github.com/AdarshIITDH/devops_class_assignment.git>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Question 1**

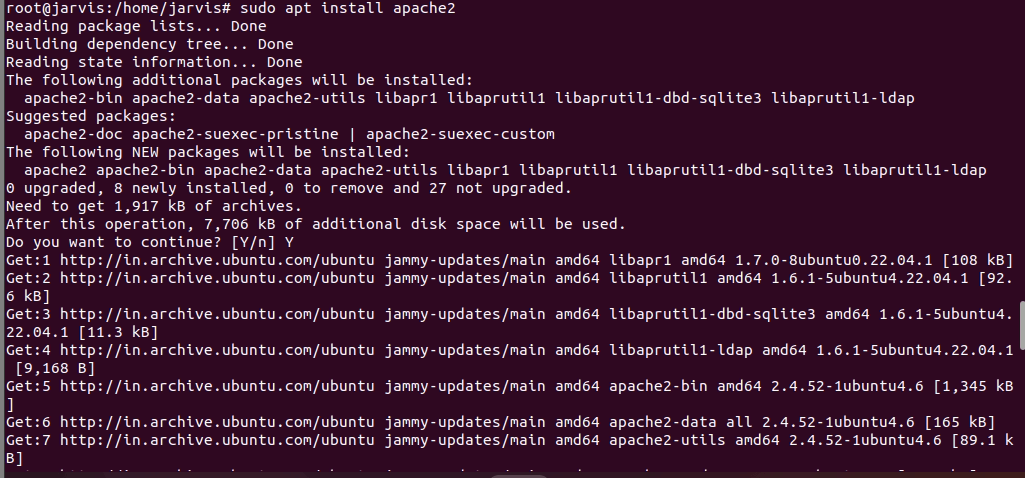
Deploy a website on localhost using either apache2 or nginx. Create a DNS name for this website as ‘awesomeweb’. You can use any web template you want or can write your own simple html code. Write a detailed documentation with steps involved.

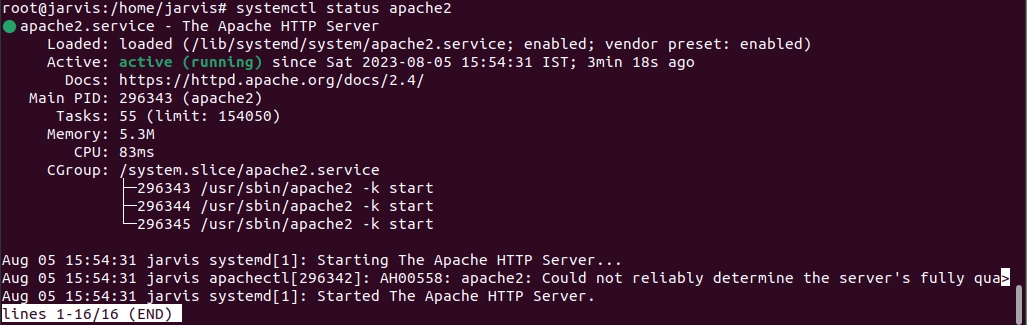
**Solution:**

**For Ubuntu with Apache**

Step 1: Install Apache2

* sudo apt update
* sudo apt install apache2
* systemctl status apache2





Step 2: Configure Virtual Host

1. Create a configuration file for the virtual host.
   * + sudo nano /etc/apache2/sites-available/awesomeweb.conf

<VirtualHost \*:80>

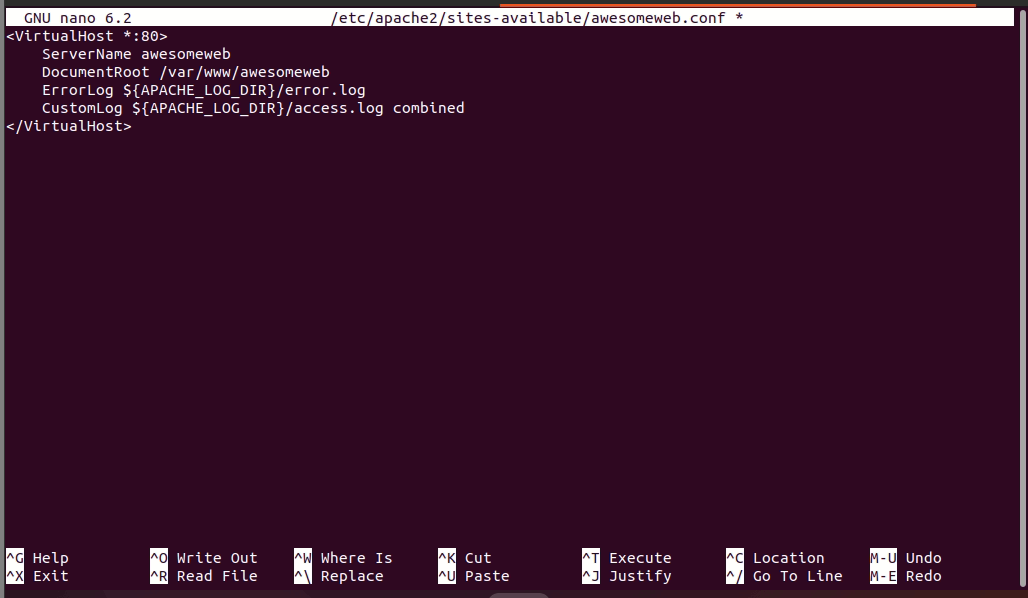
ServerName awesomeweb

DocumentRoot /var/www/awesomeweb

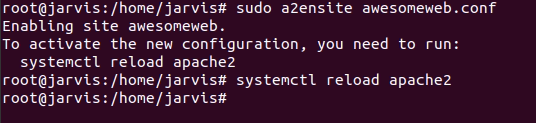
ErrorLog ${APACHE\_LOG\_DIR}/error.log

CustomLog ${APACHE\_LOG\_DIR}/access.log combined

</VirtualHost>

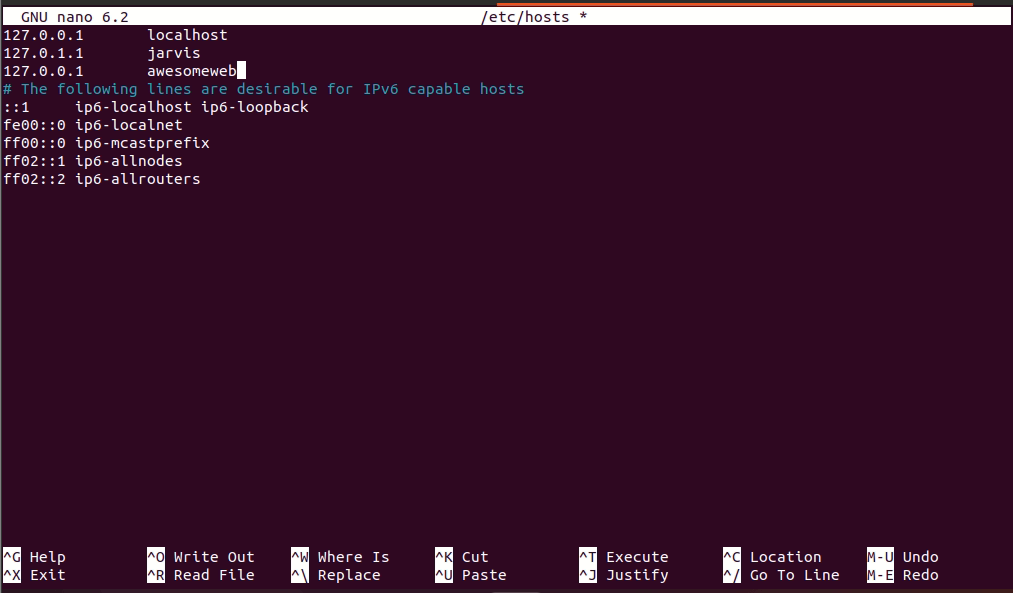


1. Enable the virtual host:
   * + sudo a2ensite awesomeweb.conf



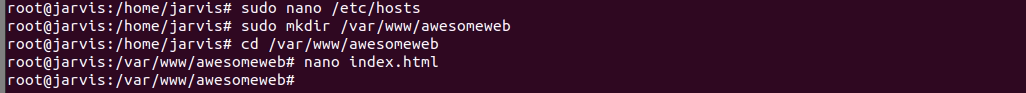
Step 3: Create DNS entry (Edit Hosts File)

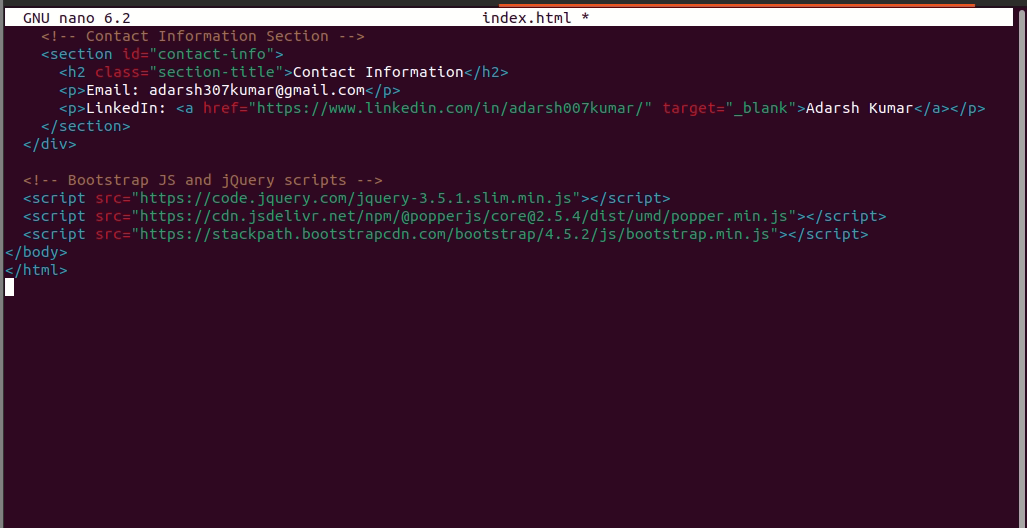
1. Open the hosts file in the editor:
   * + sudo nano /etc/hosts
2. Add the following line:
   * + 127.0.0.1 awesomeweb



Step 4: Prepare Website Files

1. Create the website directory:
   * + sudo mkdir /var/www/awesomeweb
2. Place your website files (HTML) in the /var/www/awesomeweb directory.



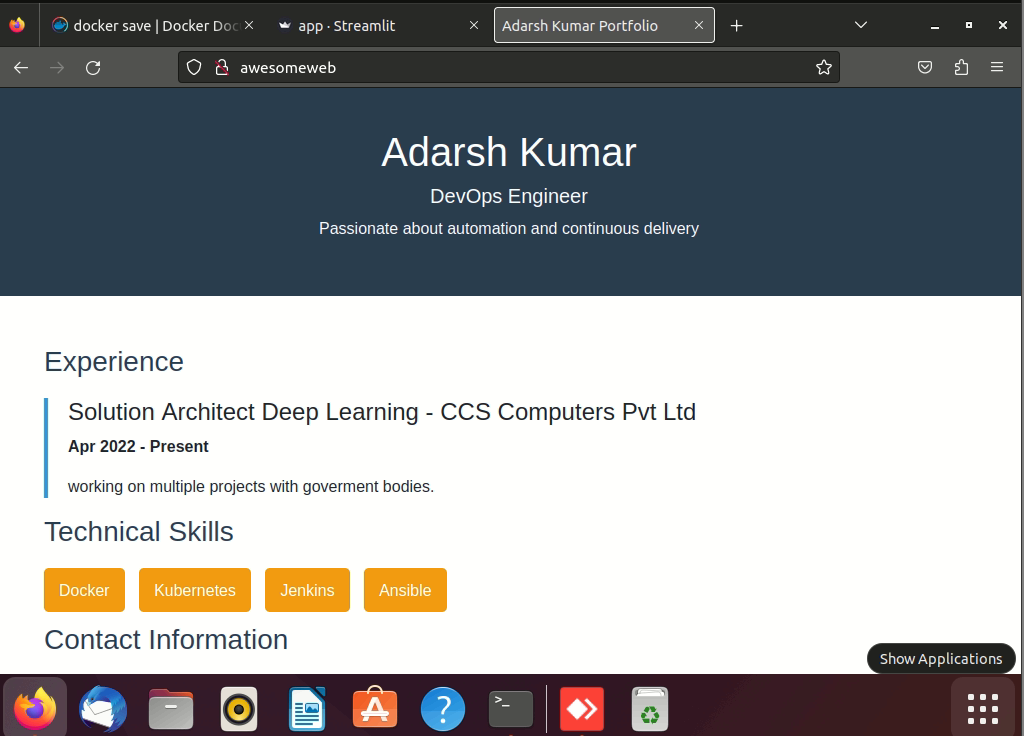


Step 5: Restart Apache2

* + sudo service apache2 restart

Step 6: Open <http://awesomeweb> in web browser

* + firefox http://awesomeweb



**For Ubuntu with Nginx**

Step 1: Install Nginx

* sudo apt update
* sudo apt install nginx
* systemctl status ngnix

Step 2: Configure Virtual Host

1. Create a configuration file for the virtual host.
   * + sudo nano /etc/nginx/sites-available/awesomeweb

server {

listen 80;

server\_name awesomeweb;

root /var/www/awesomeweb;

access\_log /var/log/nginx/awesomeweb.access.log;

error\_log /var/log/nginx/awesomeweb.error.log;

location / {

index index.html;

}

}

1. Enable the virtual host:
   * + sudo ln -s /etc/nginx/sites-available/awesomeweb /etc/nginx/sites-enabled/

Step 3: Create DNS entry (Edit Hosts File)

1. Open the hosts file in the editor:
   * + sudo nano /etc/hosts
2. Add the following line:
   * + 127.0.0.1 awesomeweb

Step 4: Prepare Website Files

1. Create the website directory:
   * + sudo mkdir /var/www/awesomeweb
2. Place your website files (HTML) in the /var/www/awesomeweb directory.

Step 5: Restart Nginx

* + sudo service nginx restart

Step 6: Open <http://awesomeweb> in web browser

* + firefox http://awesomeweb

--------------------------------------------------------------------------------------------------------------------------------------

**Question 2**

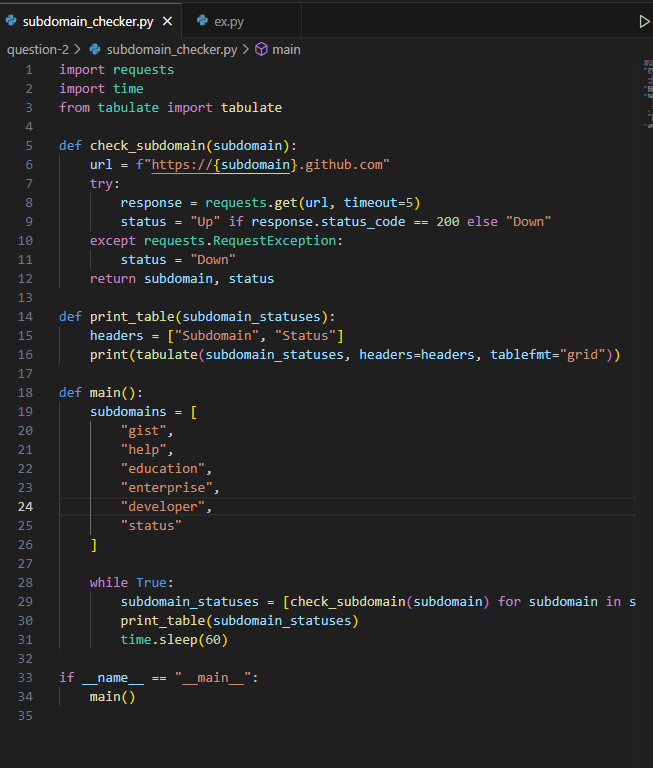
A website can have many subdomains and different services are running on them. Write a Python script to check the status of the subdomains which are up or down. The script should automatically check the status every min and should update it in tabular format on the screen. Write a detailed documentation of it.

**Solution:**

Requirements:

* requests library: To make HTTP requests and check the status of subdomains.
* tabulate library: To present the results in a tabular format.
* time library: to check in 60sec
* pip install requests tabulate

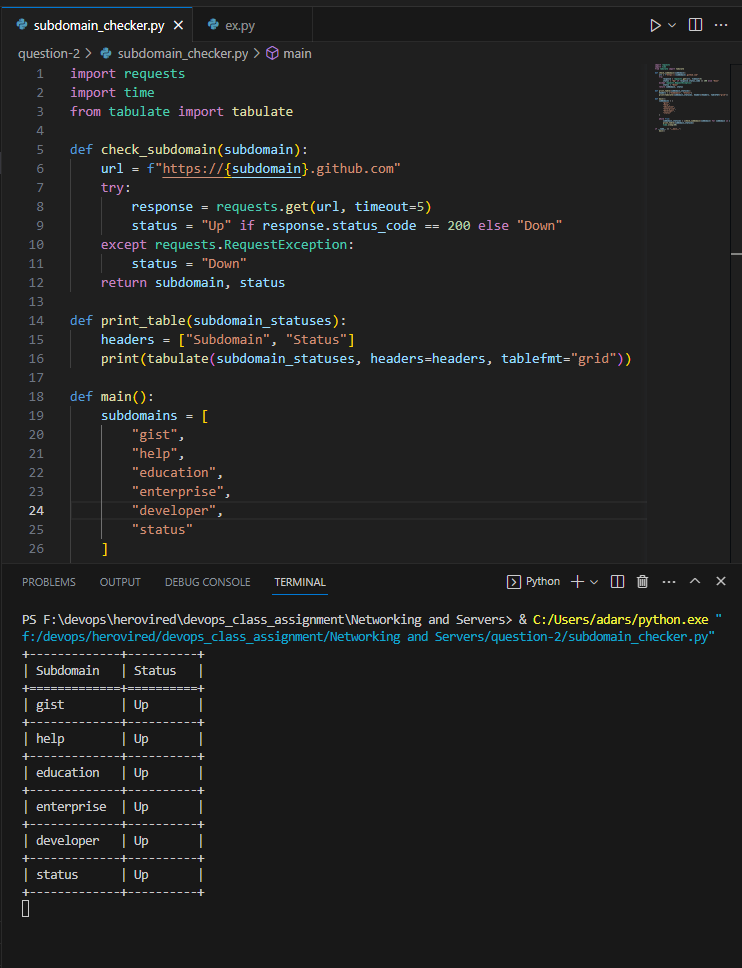
Code:



Functionality:

* The script uses the requests library to make HTTP GET requests to the URLs of the subdomains to check their status.
* The tabulate library is employed to display the results in a grid-based tabular format on the screen.
* The script continuously checks the status of all subdomains every minute and updates the status table accordingly.
* If a subdomain's service is up and responds with a status code of 200, the status will be marked as 'Up.' Otherwise, it will be marked as 'Down.'
* To stop the script, simply press Ctrl+C.

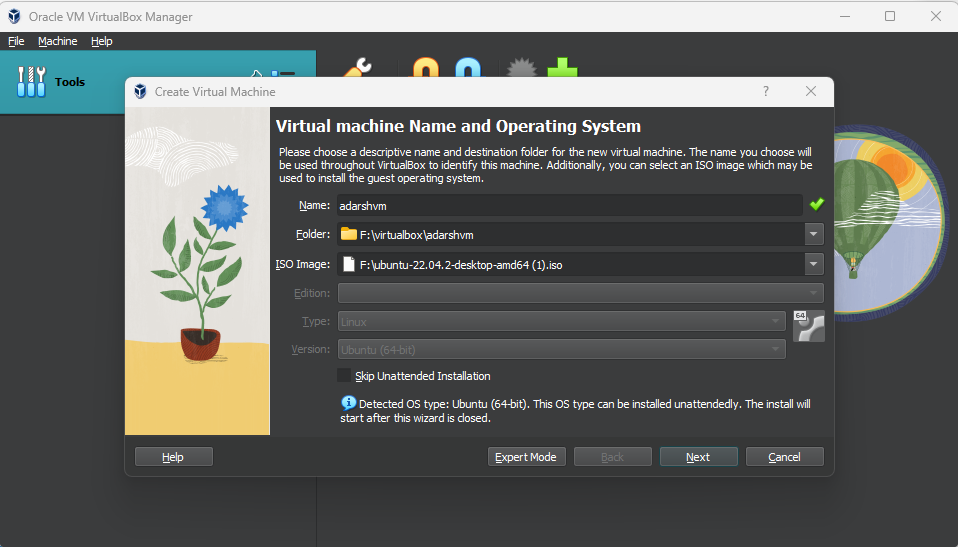
Output:



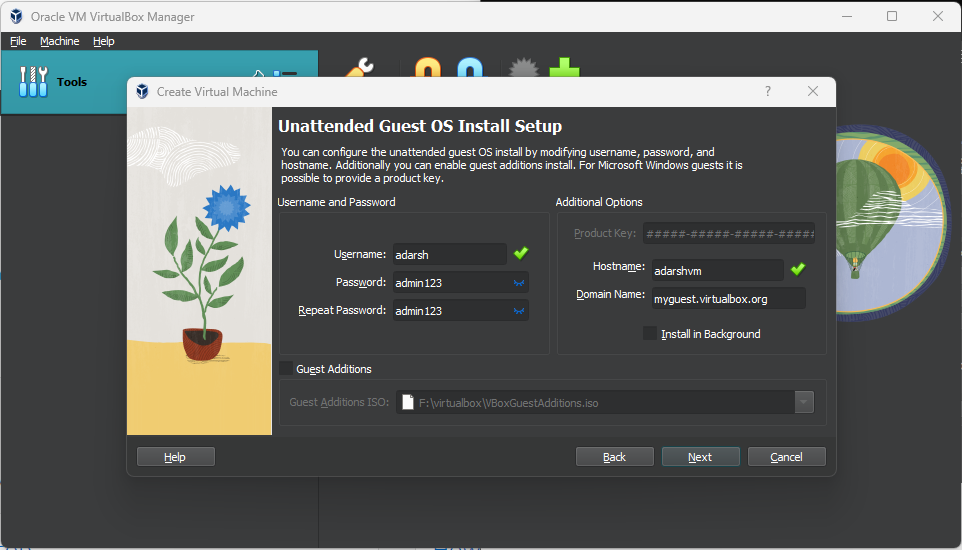
**Question 3: Hosting and Scanning a website on Virtual Machine**

Install VirtualBox

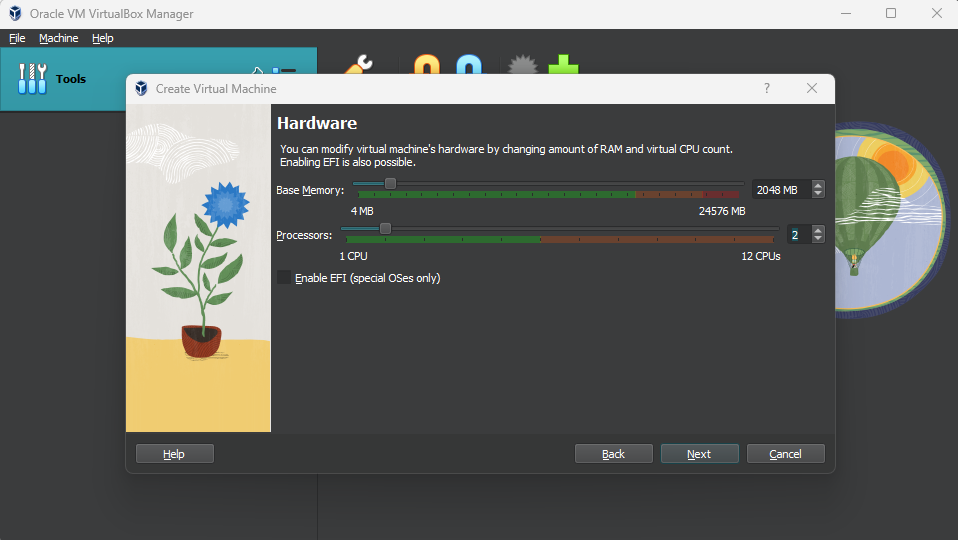
1. For Windows: Host machine is windows installing a VirtualBox from <https://www.virtualbox.org/> and creating a ubuntu VM inside it.



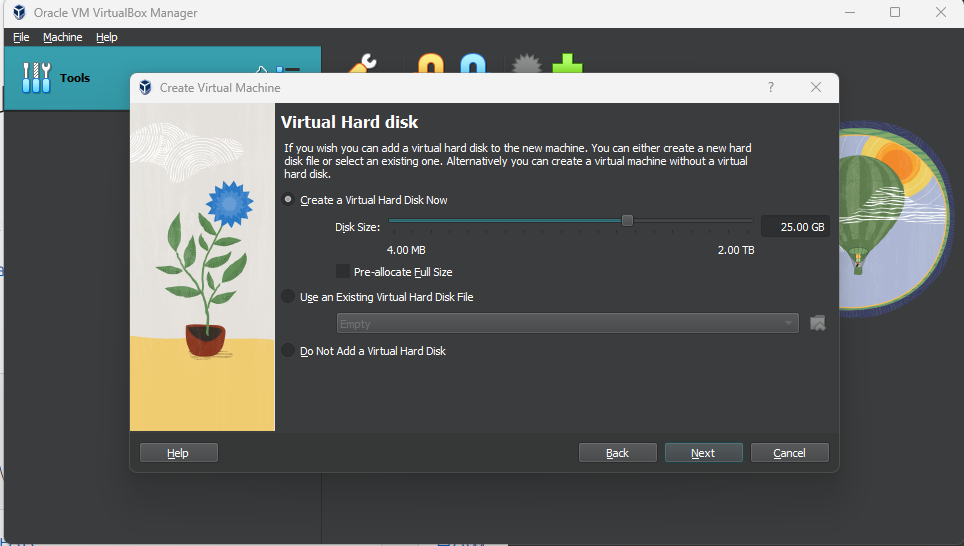
Added a user “adarsh” with password “admin123”



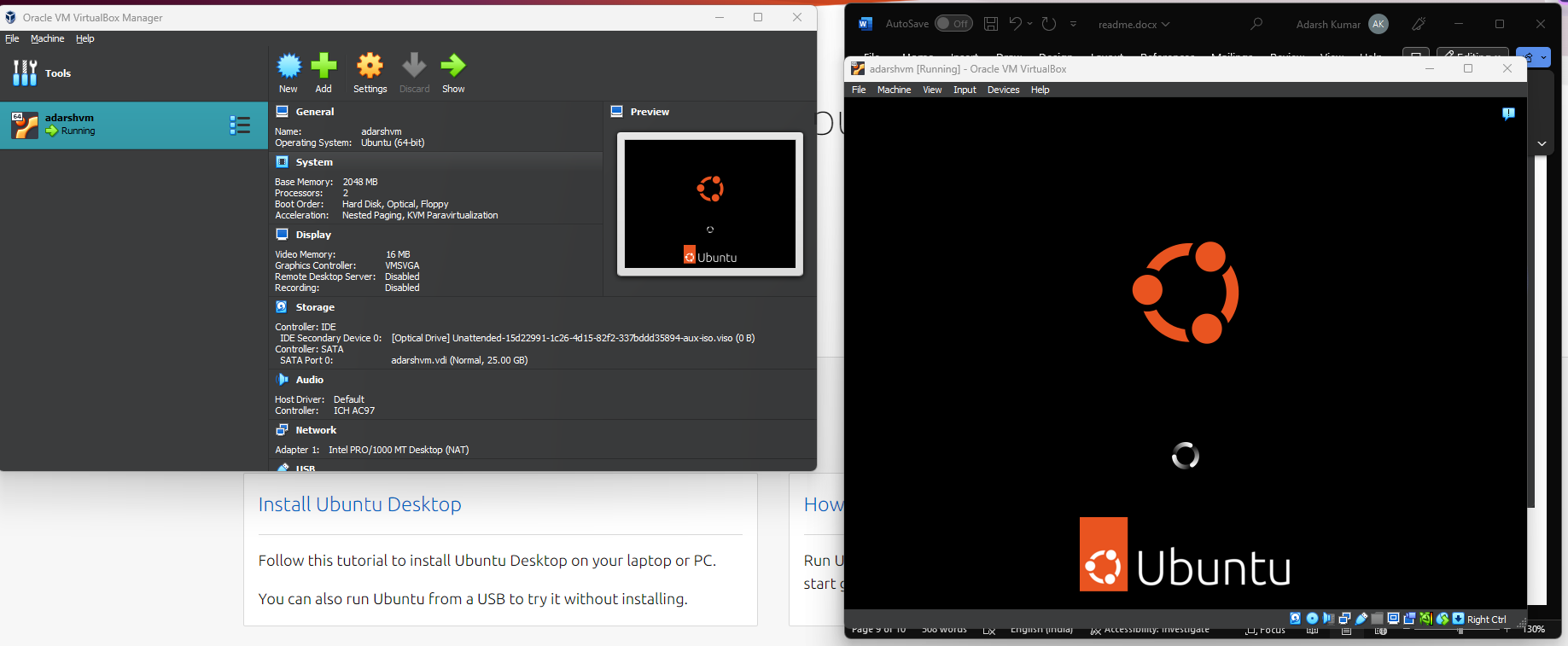
Assigning the Ram to the VM



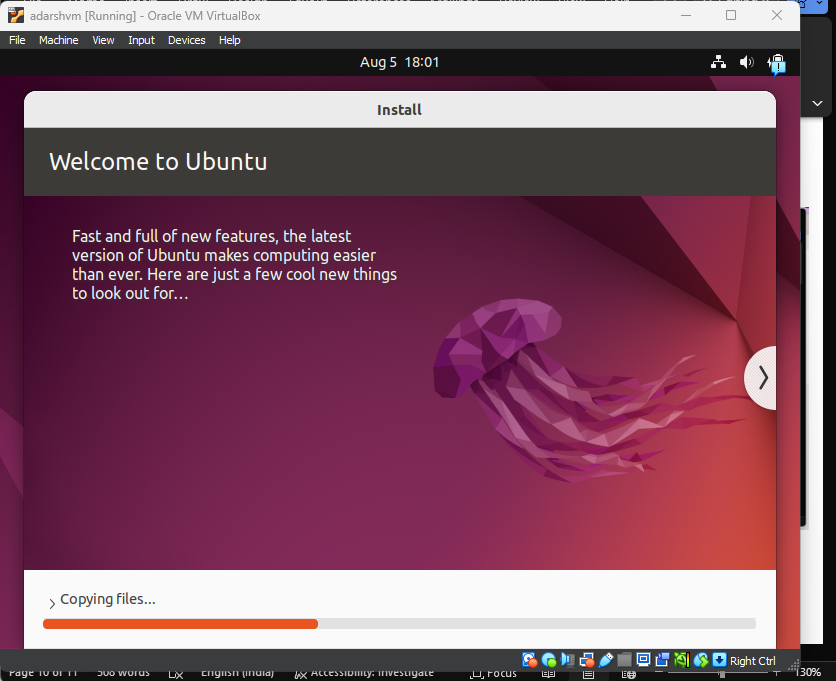
Assigning the hard disk to the VM



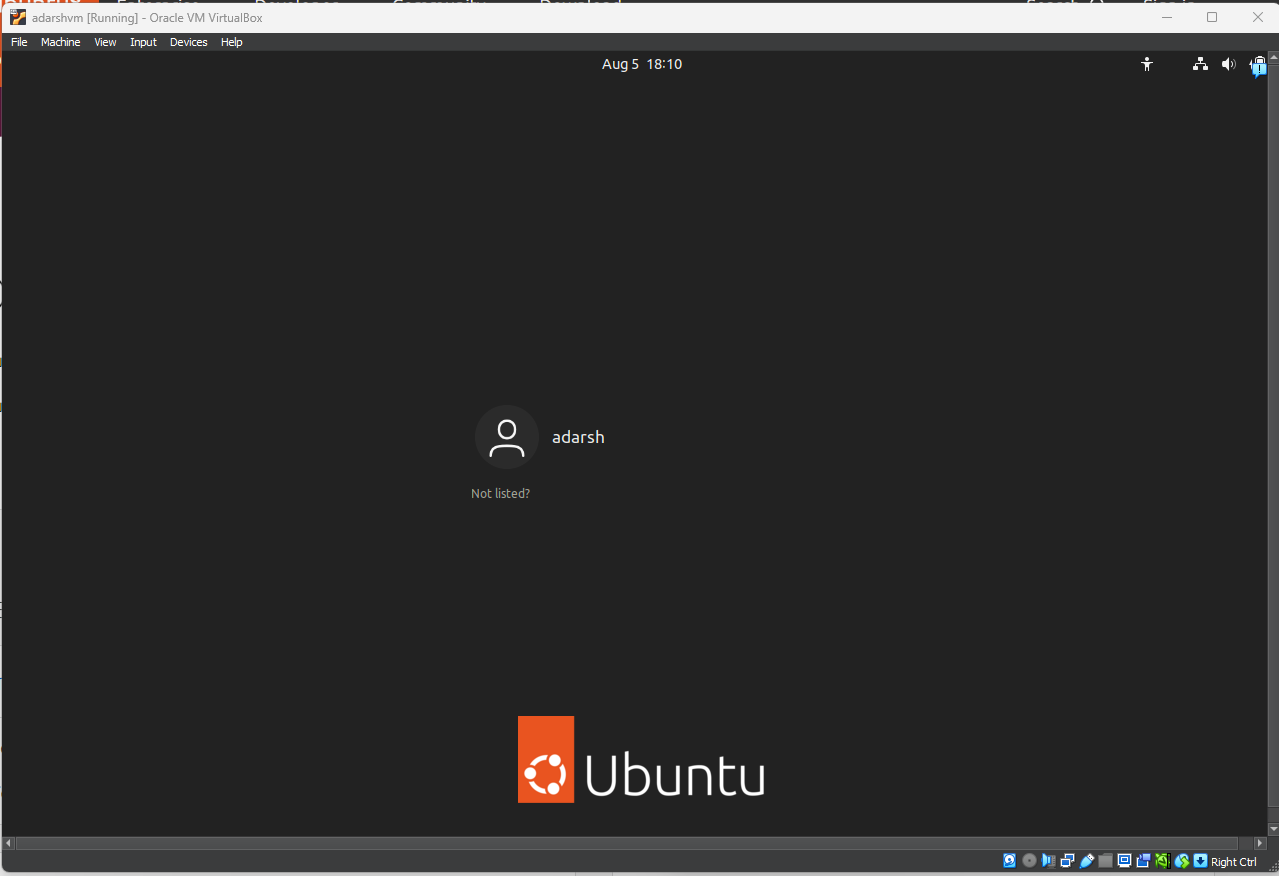
Ubuntu VM is ready.



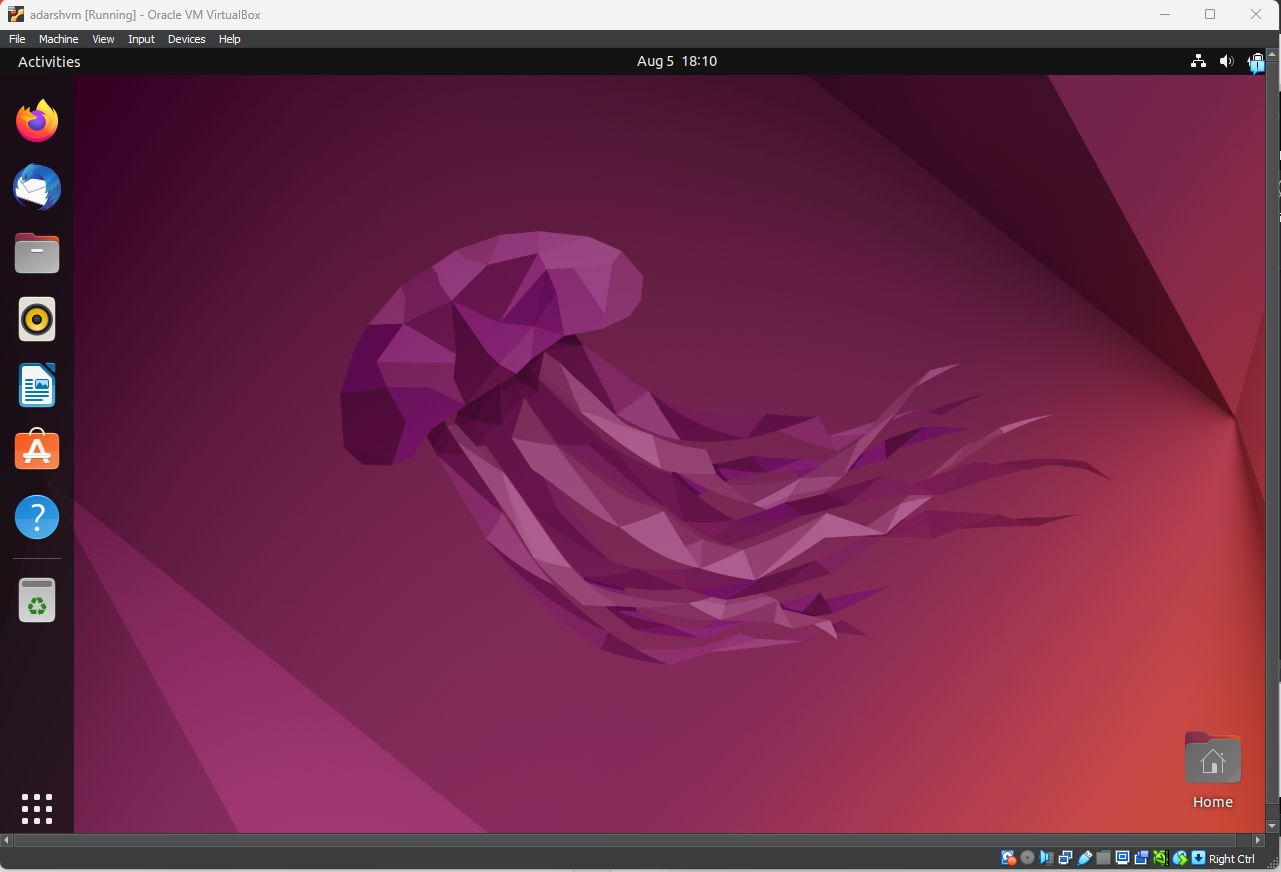
Ubuntu is being installed in the virtualbox



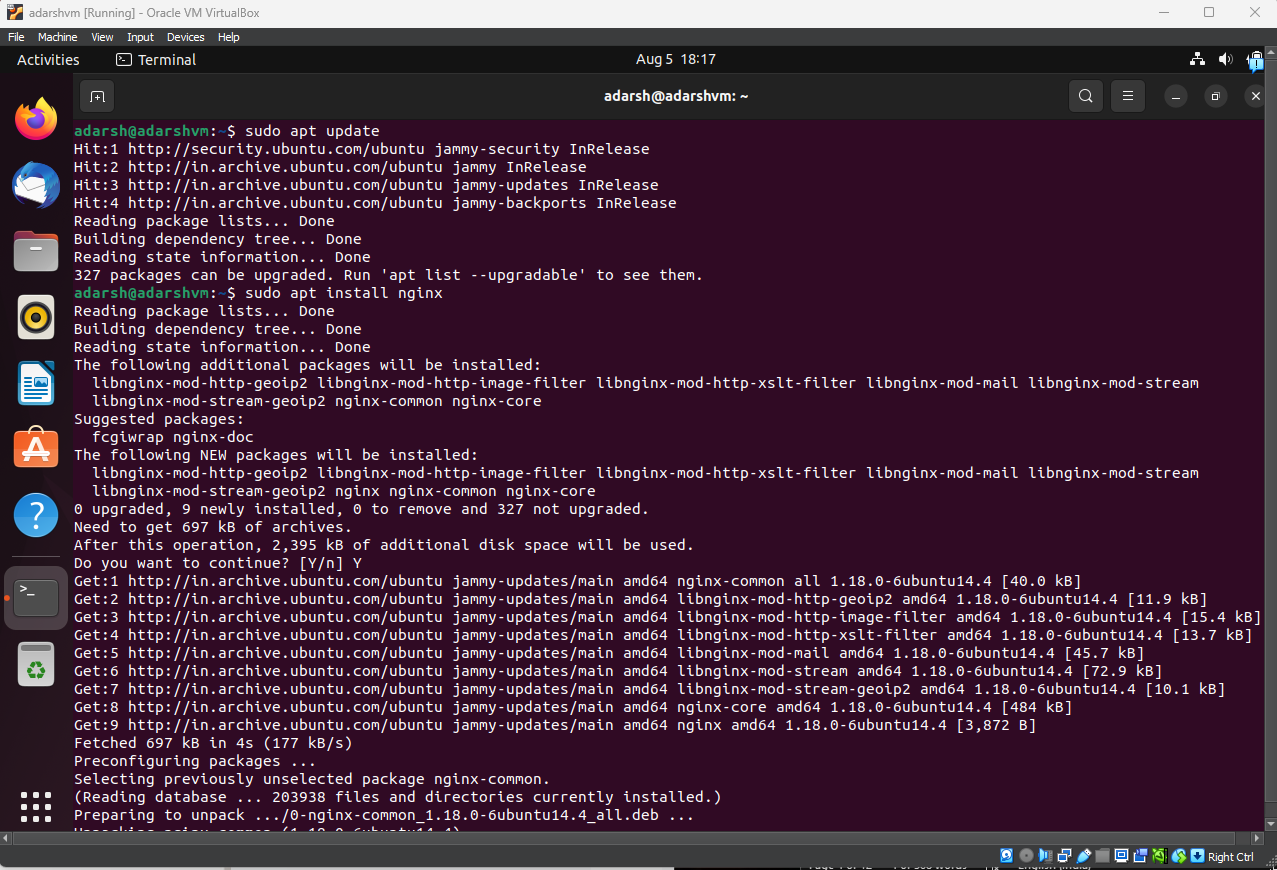
Logging the VM via user “adarsh” and password “admin123”



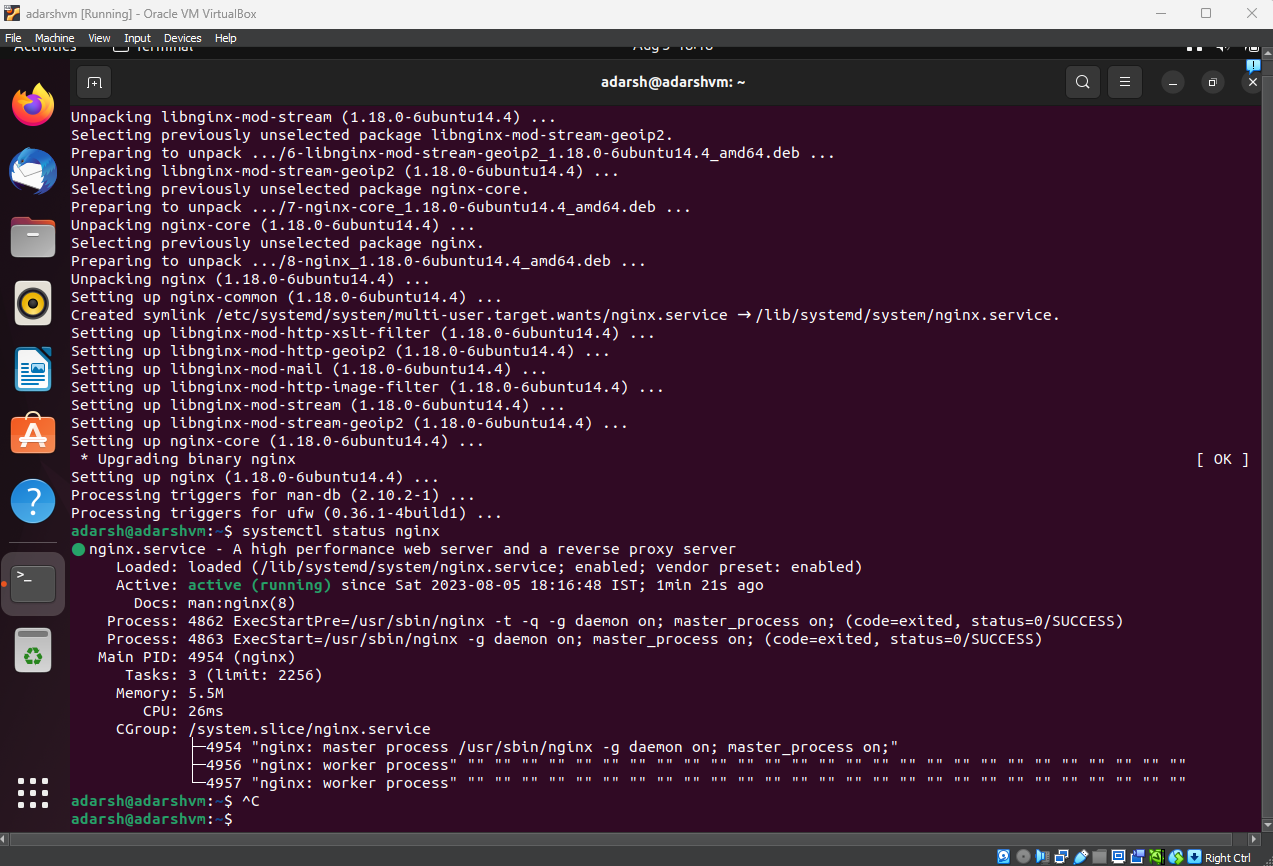
We are inside the ubuntu VM



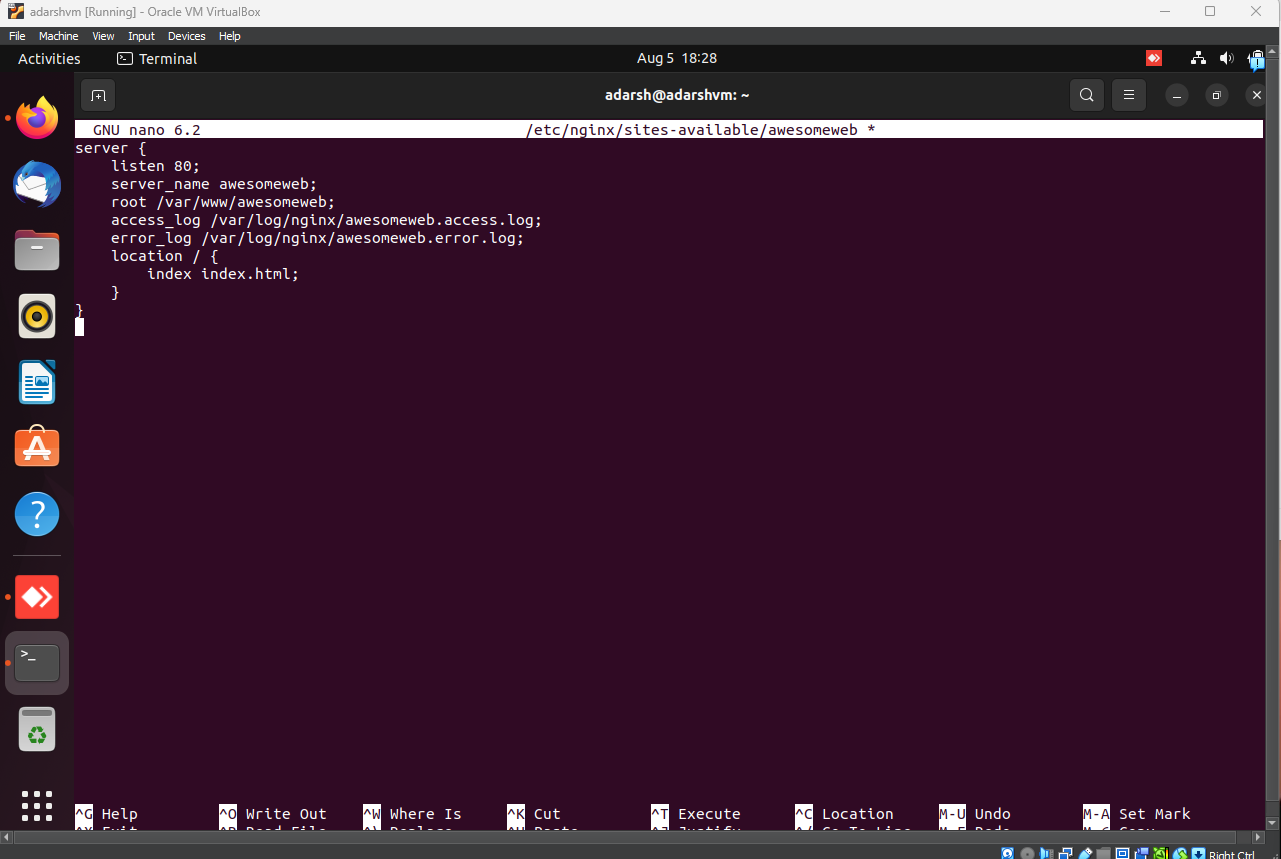
Task 1: Install Nginx inside the Ubuntu machine and host a website.



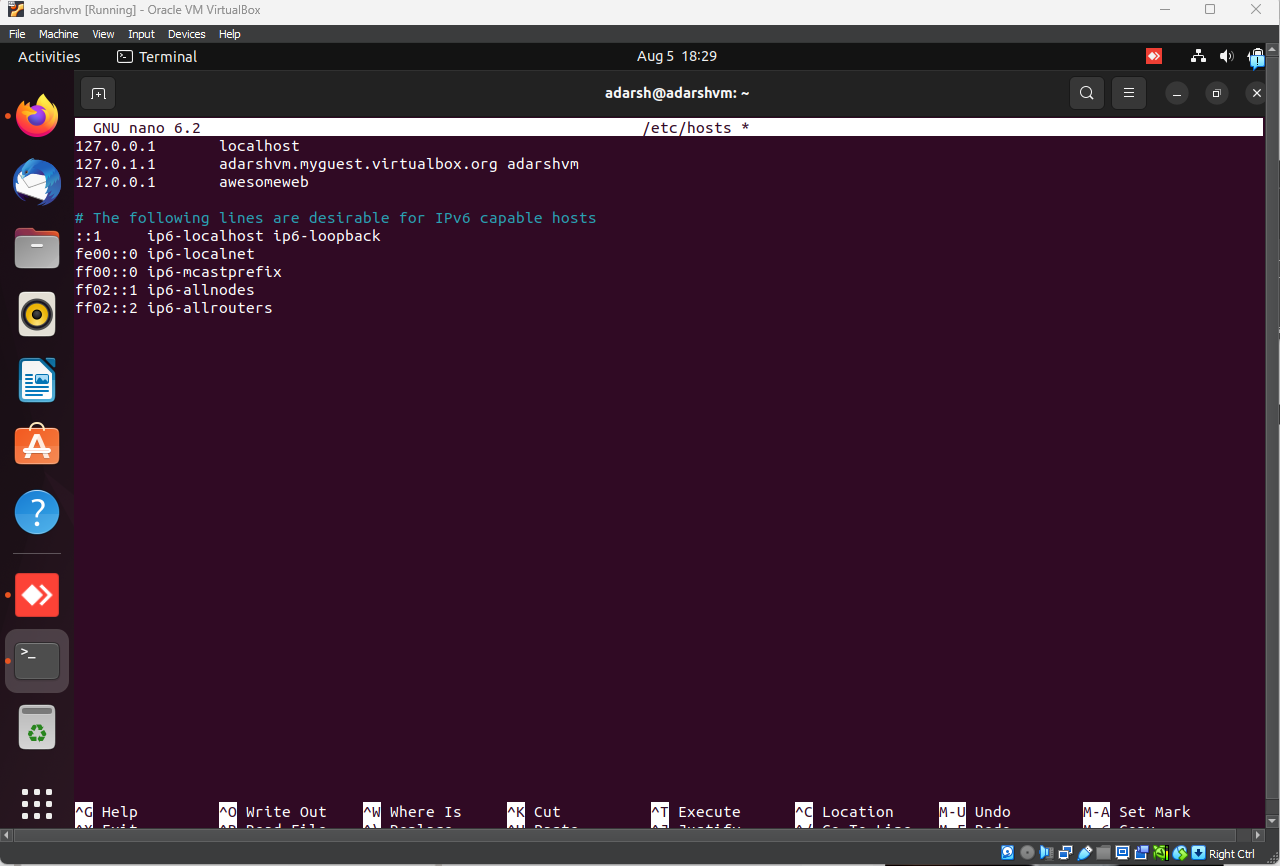
Nginx server is Active



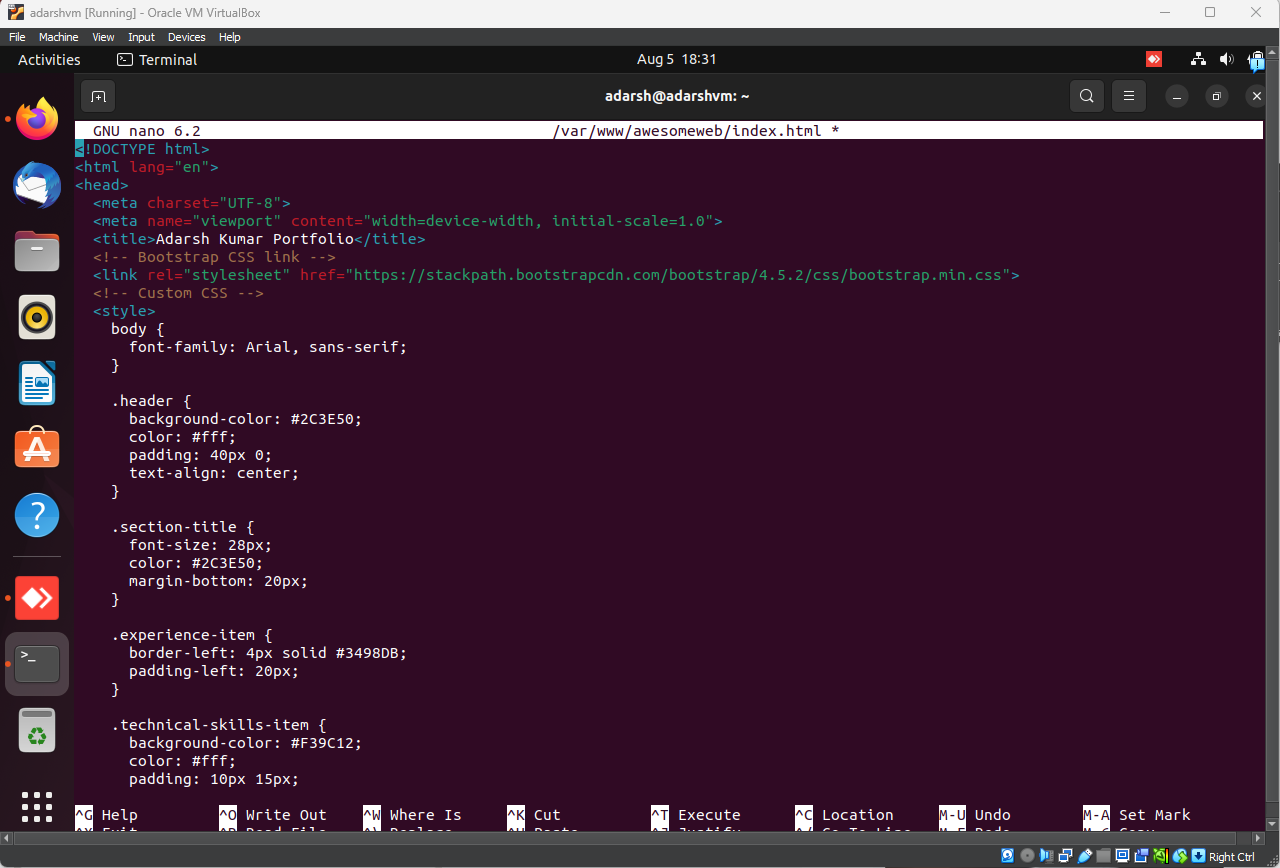
Hosting a webpage in the nginix server with DNS as awesomeweb



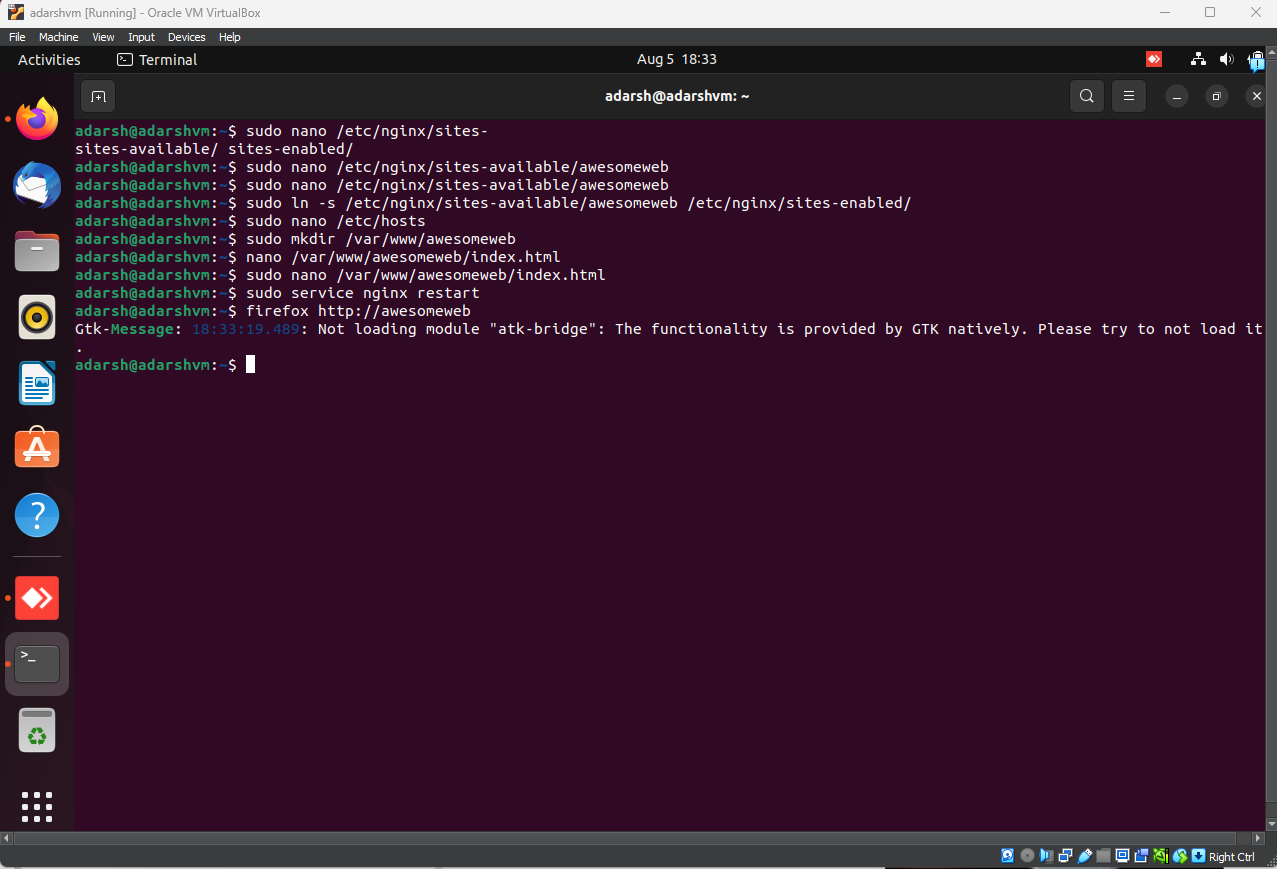
Adding the DNS in the hosts file



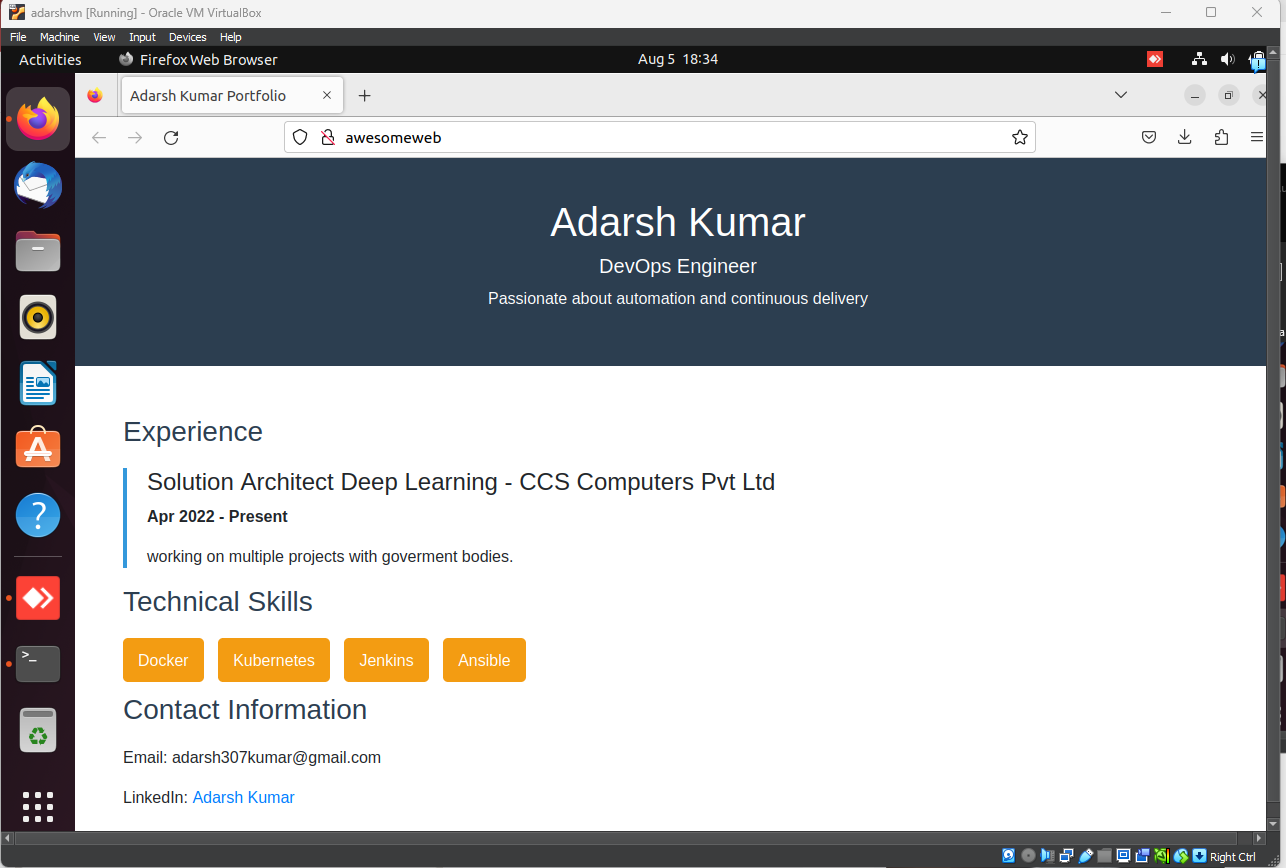
Placing the static html page in /var/www/awesomeweb/



Restarting the server to update the changes and launching the webpage in firefox

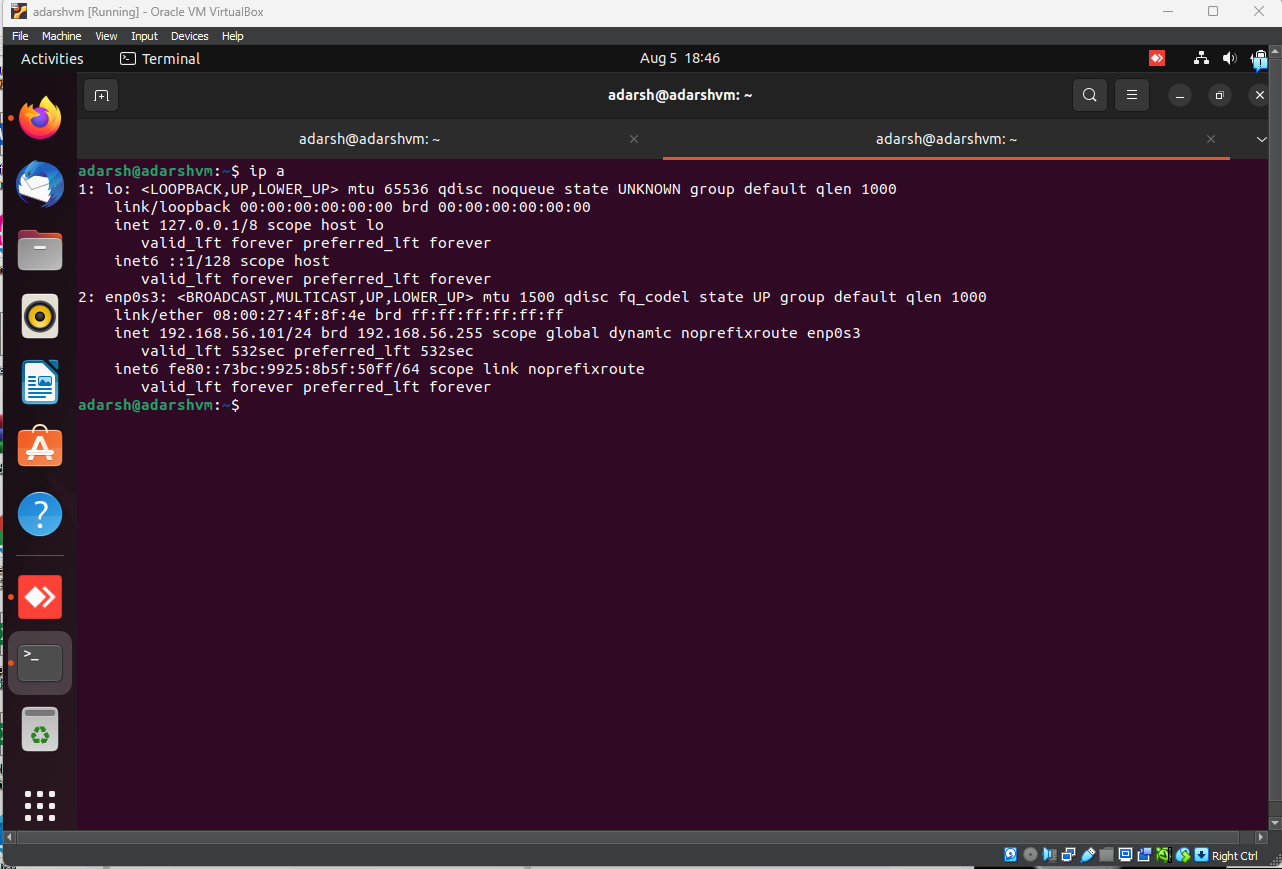


Webpage is hosted locally at <http://awesomeweb>

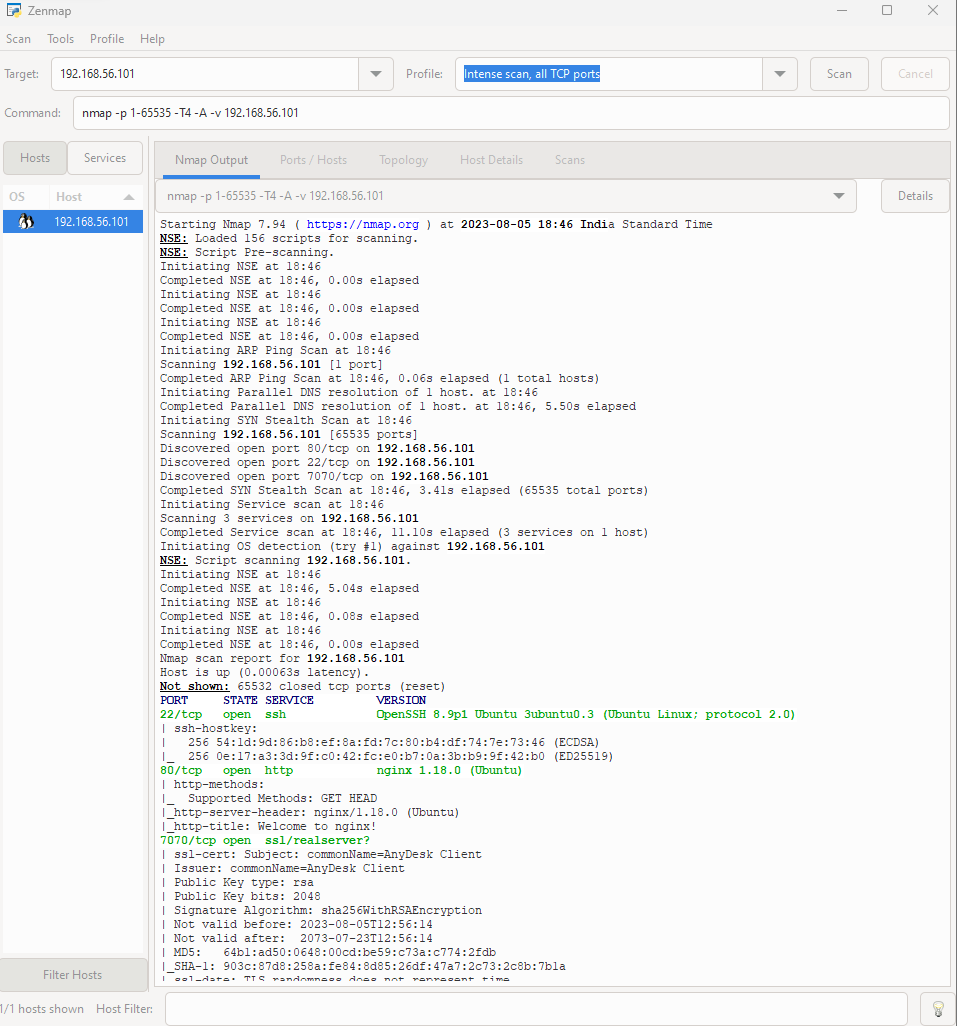


Task 2: Come back to your host machine (windows/Linux/mac) and scan the virtual machine using Nmap.

Checking the IP of Ubuntu VM. IP is 192.168.56.101



Scanning the VM using NMAP on host windows machine.



We can see port 22, 80 and 7070 is open on Ubuntu VM. It means ssh facility is enable at Ubuntu VM. Port 80 is telling that nginx server is being hosted.

