Name: Charu C A

USN: 20BTRCM003

Course: Cloud Technology and Mobile Application

Subject: Linux Administration Lab

Experiment - 1

Aim:-

Installing RHEL and understanding the services of Linux.

Theory:-

Red Hat Enterprise Linux (RHEL) is a distribution of the Linux operating system that is developed and maintained by Red Hat. RHEL is known for its stability, security, and reliability, and provides a stable platform for deploying applications and services. The operating system is also compatible with a wide range of hardware platforms, making it suitable for use in a variety of environments.

A "service" in Linux Operating System refers to a background process that runs on a system, providing a specific function or set of functions to other parts of the system or to users.

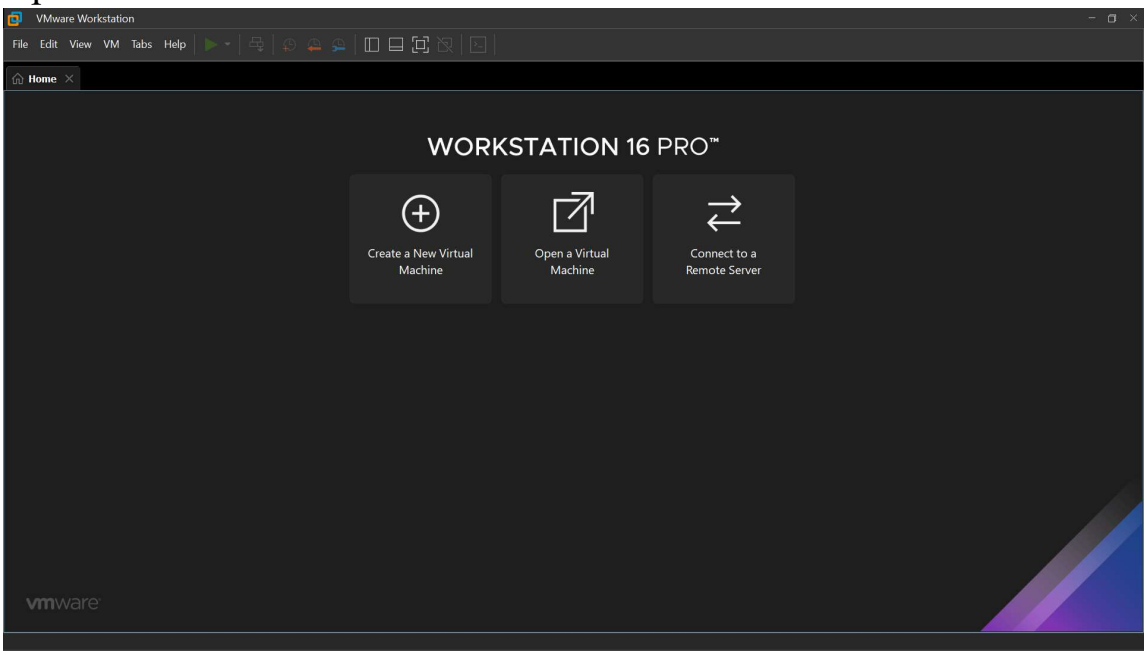
Services of Linux:-

- Apache Web Server - a popular open-source web server for hosting websites and applications.
- SSH (Secure Shell) - a secure network protocol for remote access to a Linux system.
- FTP (File Transfer Protocol) - a standard network protocol used to transfer files between computers over a network.
- NTP (Network Time Protocol) - a protocol used to synchronize the clocks of computers in a network.
- DNS (Domain Name System) - a system that translates domain names into IP addresses, enabling users to access websites and other network resources by name.

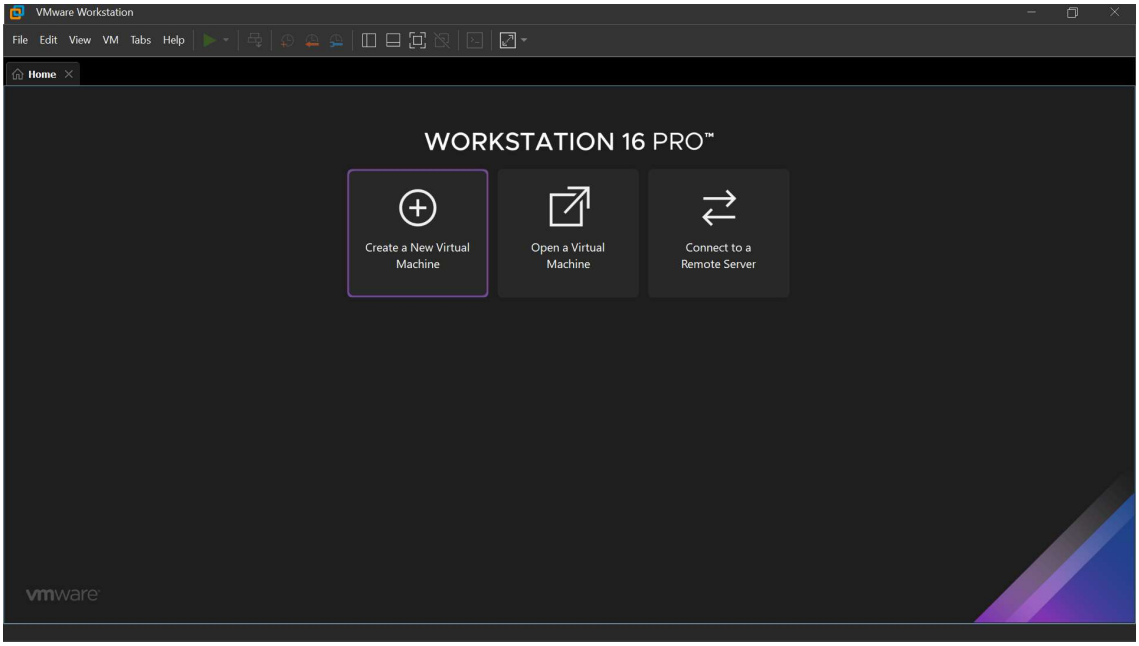
› DHCP (Dynamic Host Configuration Protocol) - a protocol used to automatically assign IP addresses to devices on a network.
› Samba - a service for sharing files and printers between Linux and Windows systems.
› CUPS (Common Unix Printing System) - a printing system for Unix-like operating systems.
› MySQL - a popular open-source relational database management system.

Procedure:-

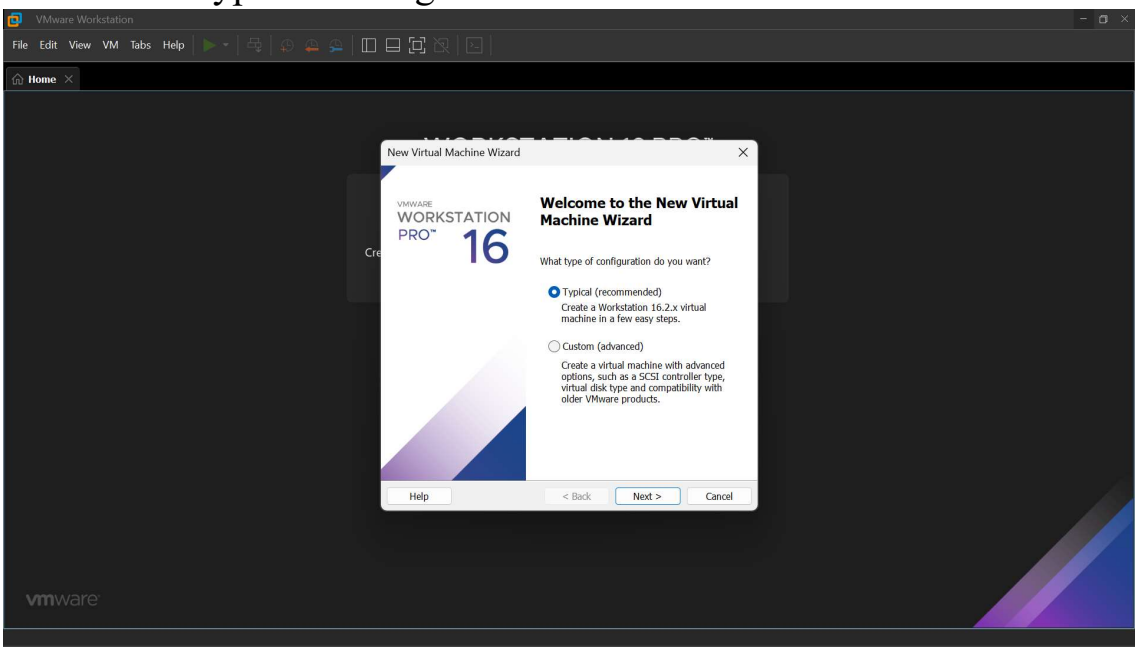
- Open VMware Workstation.



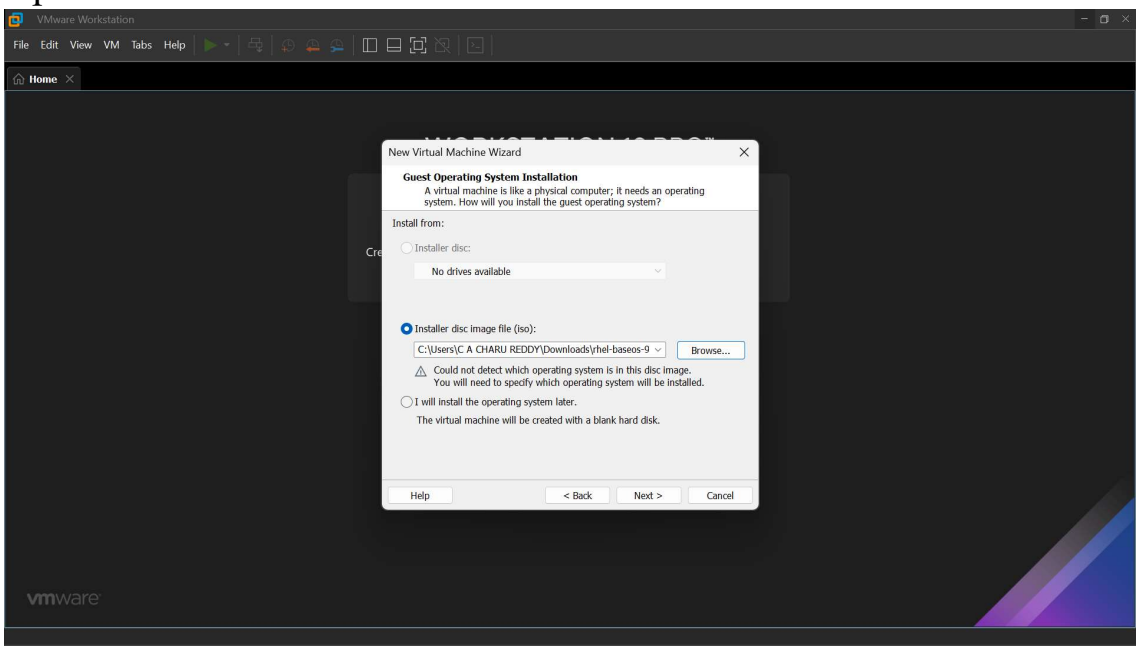
- Create a New Virtual Machine.



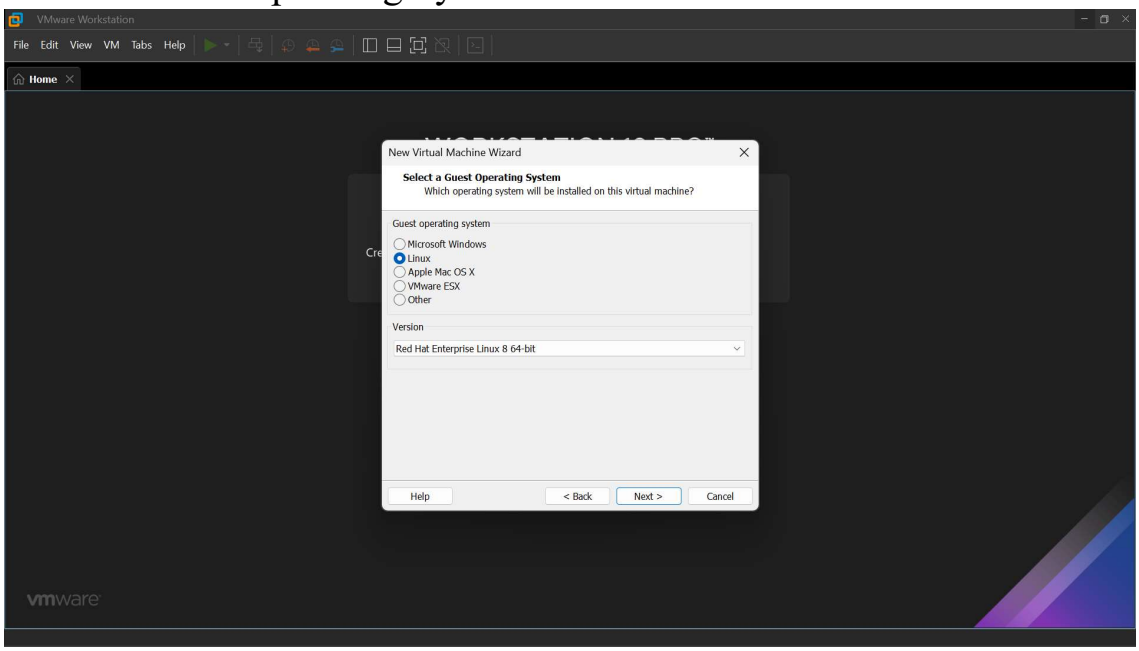
- Choose the Type of Configuration.



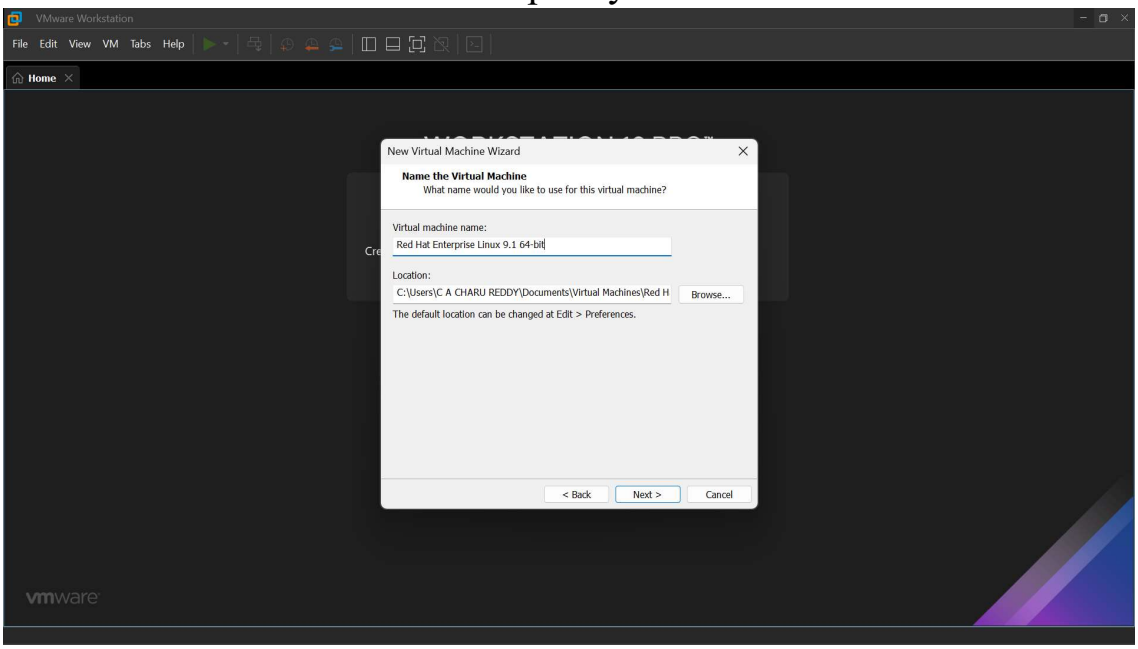
- Upload the rhel.iso file.



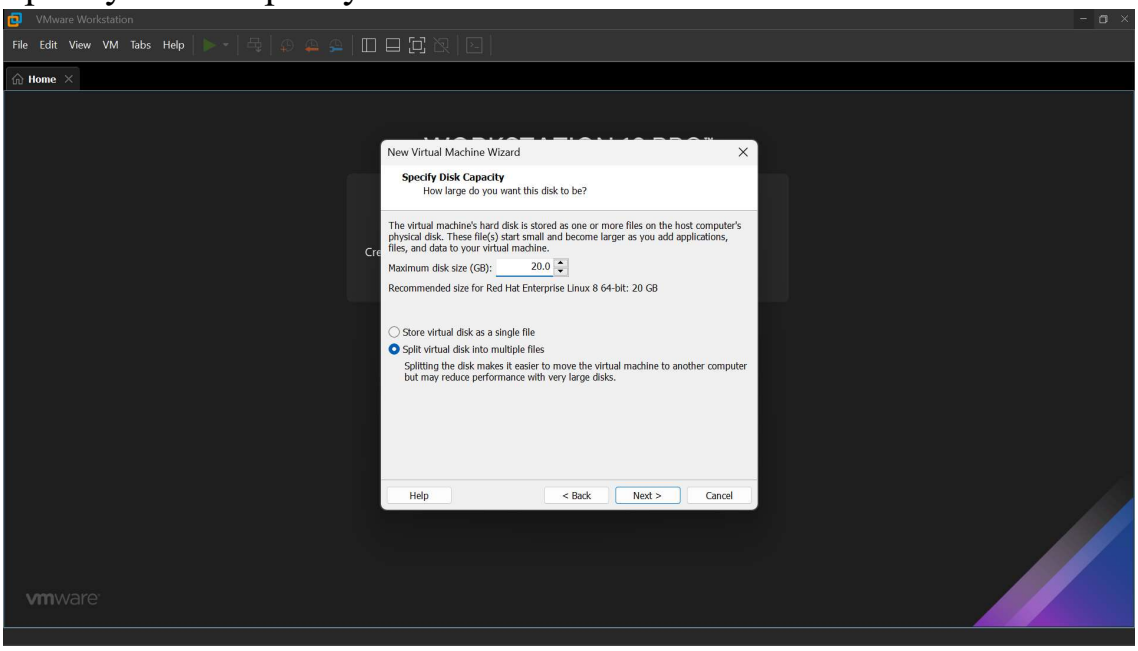
- Select a Guest Operating System and its Version.



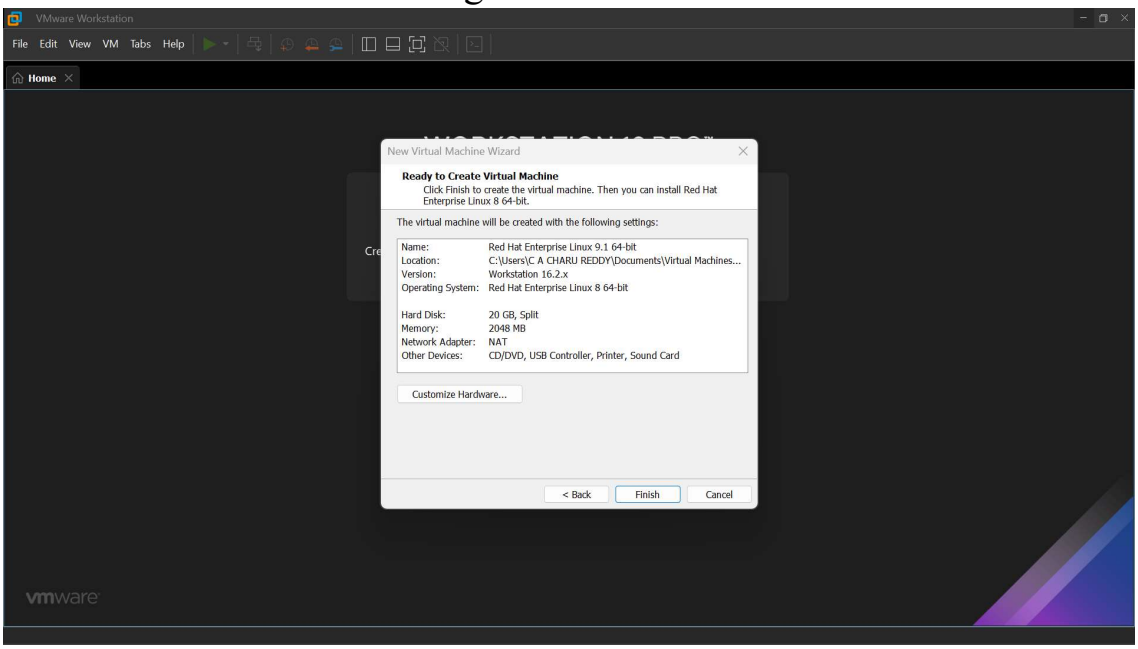
- Name the Virtual Machine and Specify the Location.



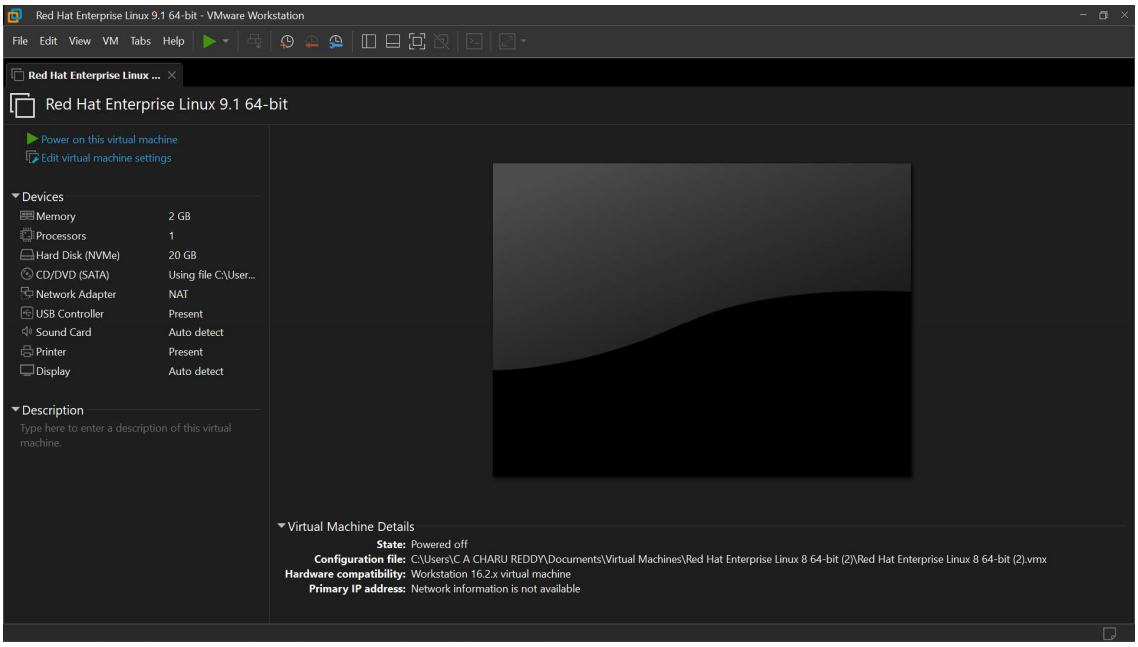
- Specify Disk Capacity.

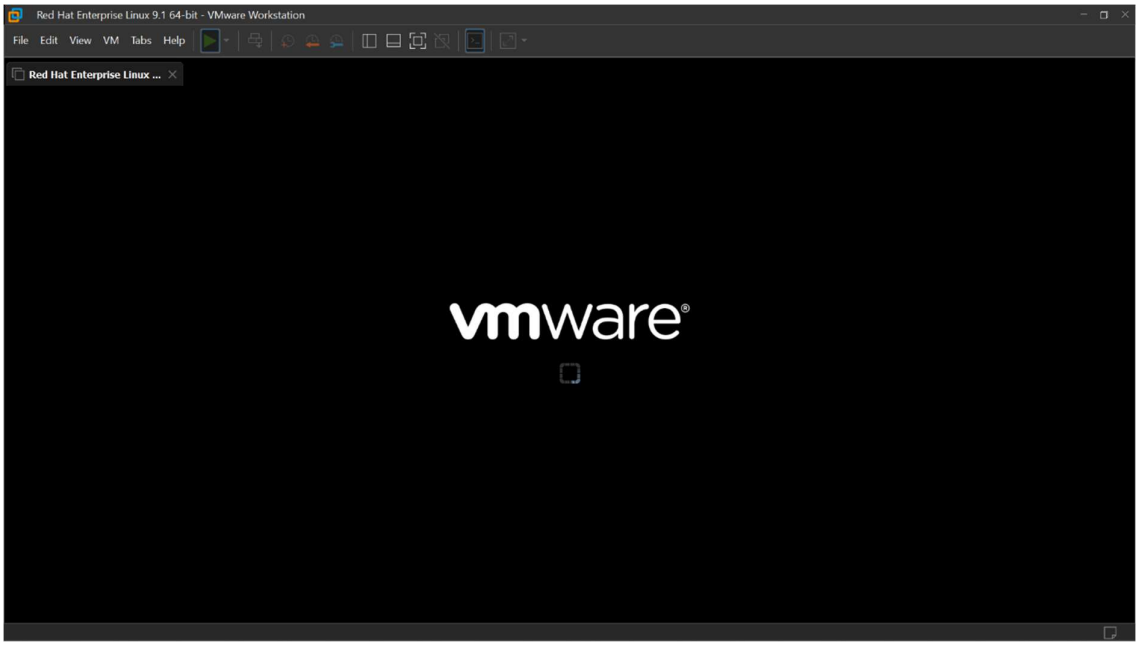


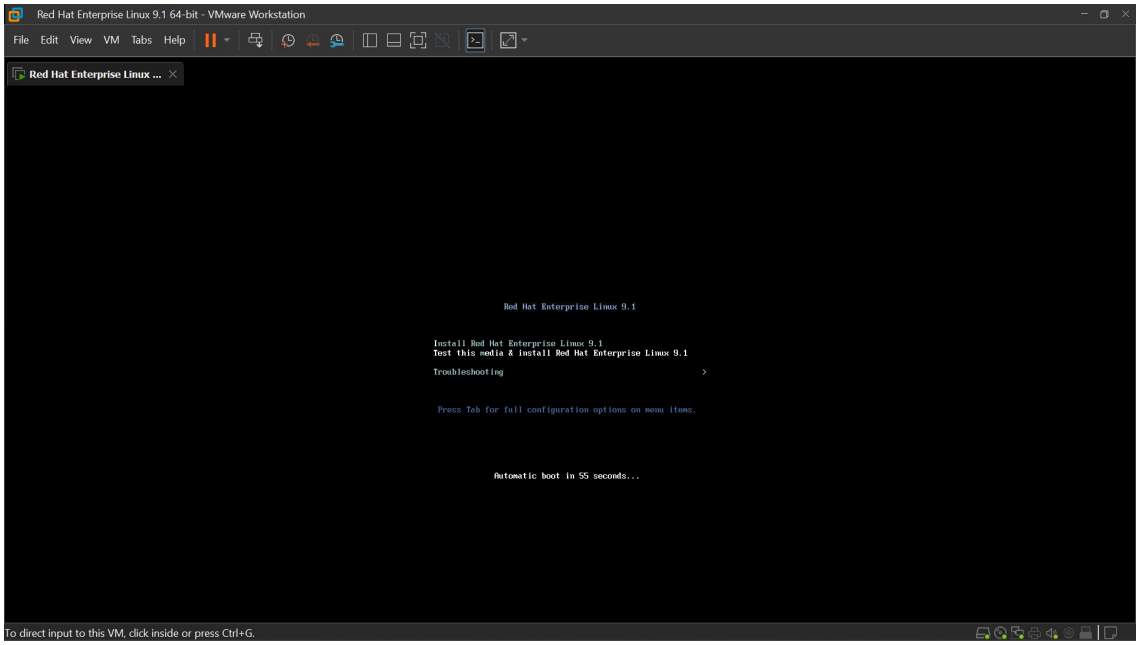
- Preview the Hardware Settings and click on Finish.

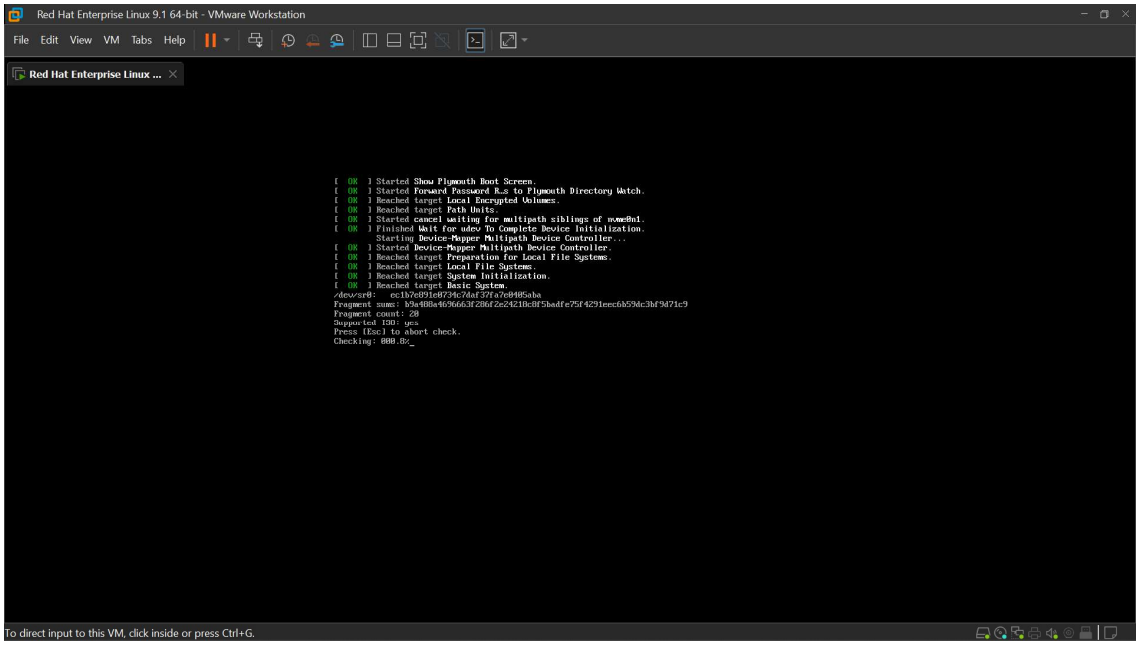


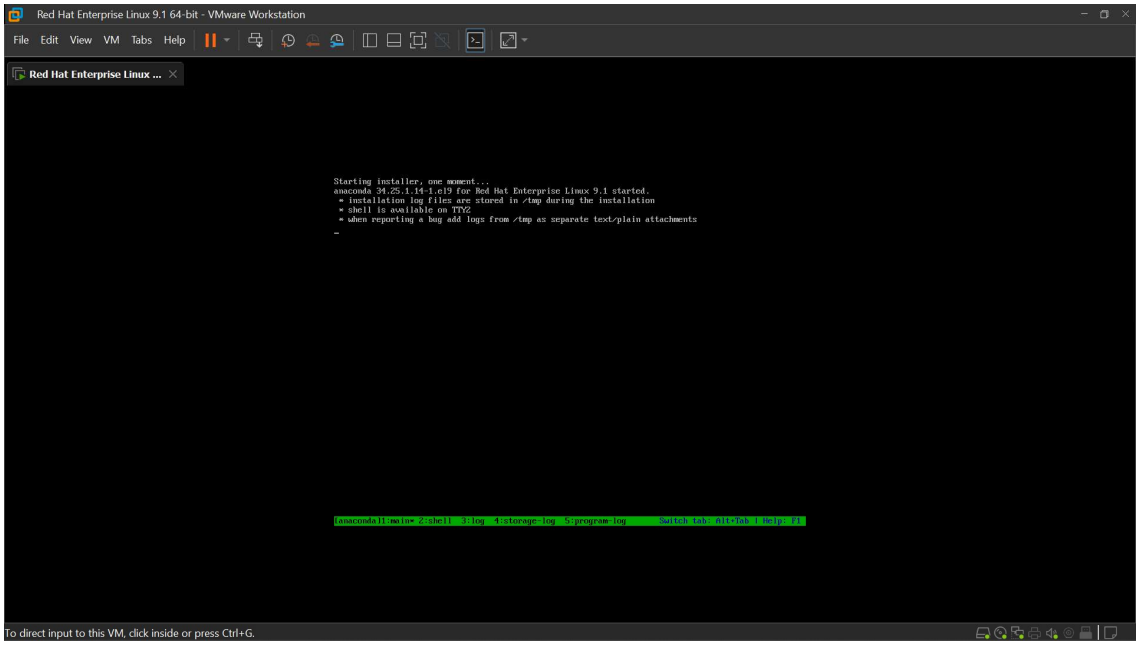
- Power on this virtual machine to install.



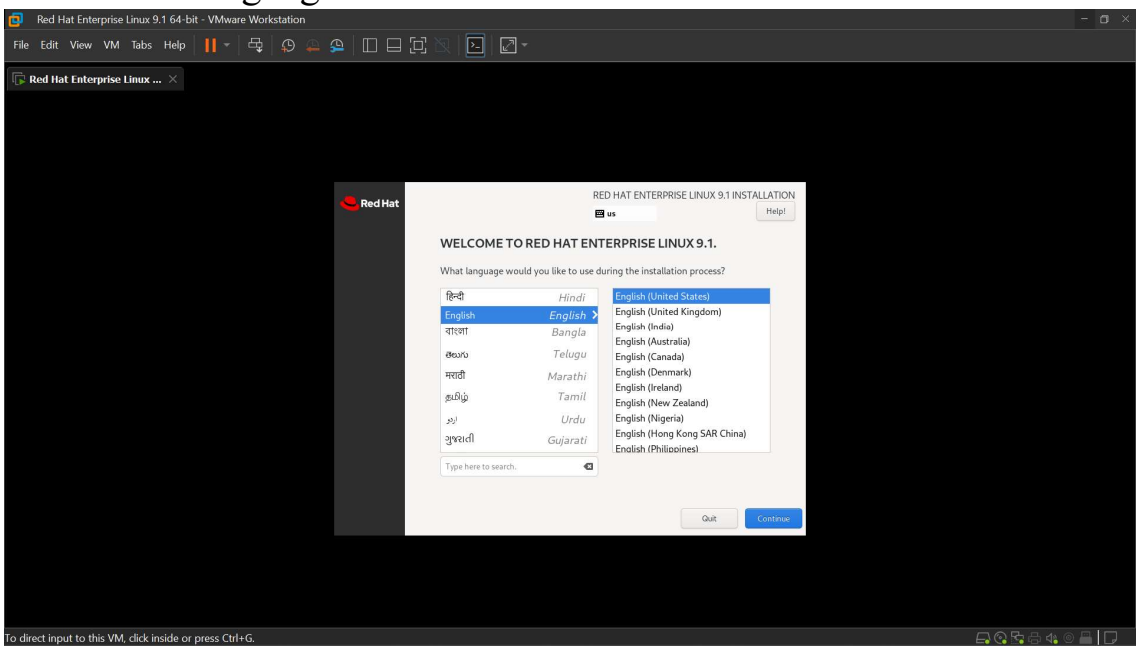

Red Hat Enterprise Linux 9.1 64-bit - VMware Workstation
File Edit View VM Tabs Help
Red Hat Enterprise Linux ...
vmware


Red Hat Enterprise Linux 9.1 64-bit - VMware Workstation
File Edit View VM Tabs Help
Red Hat Enterprise Linux ...
Red Hat Enterprise Linux 9.1
Install Red Hat Enterprise Linux 9.1
Test this media & install Red Hat Enterprise Linux 9.1
Troubleshooting
Press Tab for full configuration options on menu items.
Automatic boot in 55 seconds...
To direct input to this VM, click inside or press Ctrl+G.

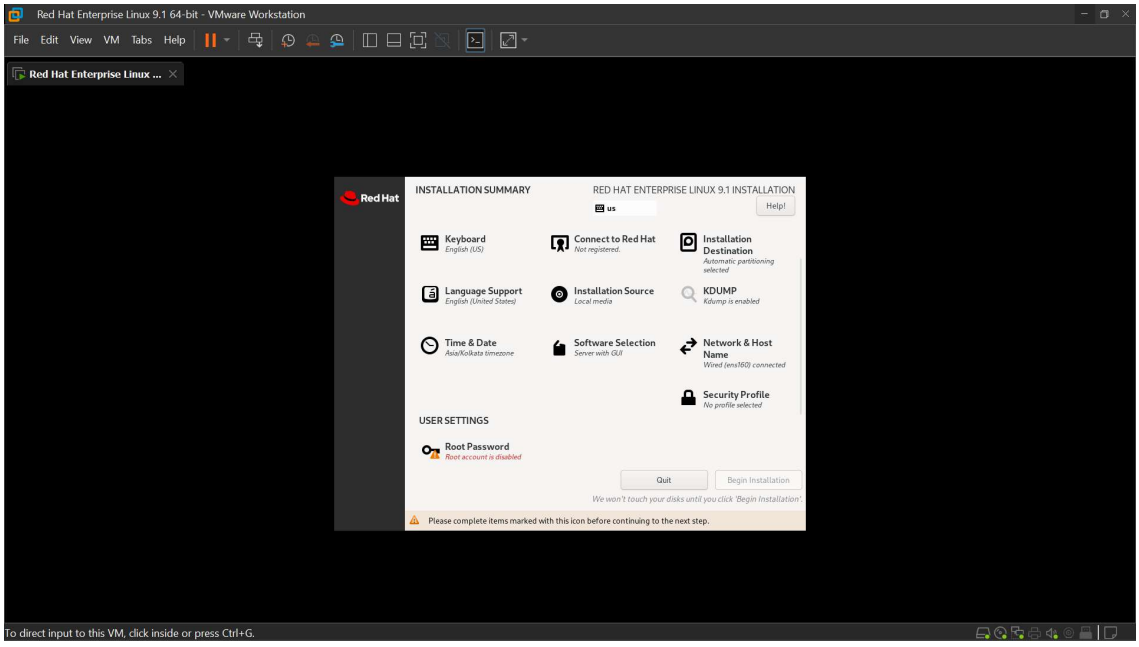

Red Hat Enterprise Linux 9.1 64-bit - VMware Workstation
File Edit View VM Tabs Help
Red Hat Enterprise Linux ...
[OK] Started Show Plymouth Boot Screen.
[OK] Started Forward Password R…s to Plymouth Directory Watch.
[OK] Reached target Local Encrypted Volumes.
[OK] Reached target Path Units.
[OK] Started cancel waiting for multipath siblings of nvme0n1.
[OK] Finished Wait for udev To Complete Device Initialization.
Starting Device-Mapper Multipath Device Controller...
[OK] Started Device-Mapper Multipath Device Controller.
[OK] Reached target Preparation for Local File Systems.
[OK] Reached target Local File Systems.
[OK] Reached target System Initialization.
[OK] Reached target Basic System.
/dev/sr0: ecb7e091e0734c7daf37fa7e0405aba
Fragment sums: b9e40ba469663f286f2e2421bc8f5badfe75f4291eecc6b594c3bf9471e9
Fragment count: 20
Supported ISO: yes
Press [Esc] to abort check.
Checking: 000.0%
To direct input to this VM, click inside or press Ctrl+G.

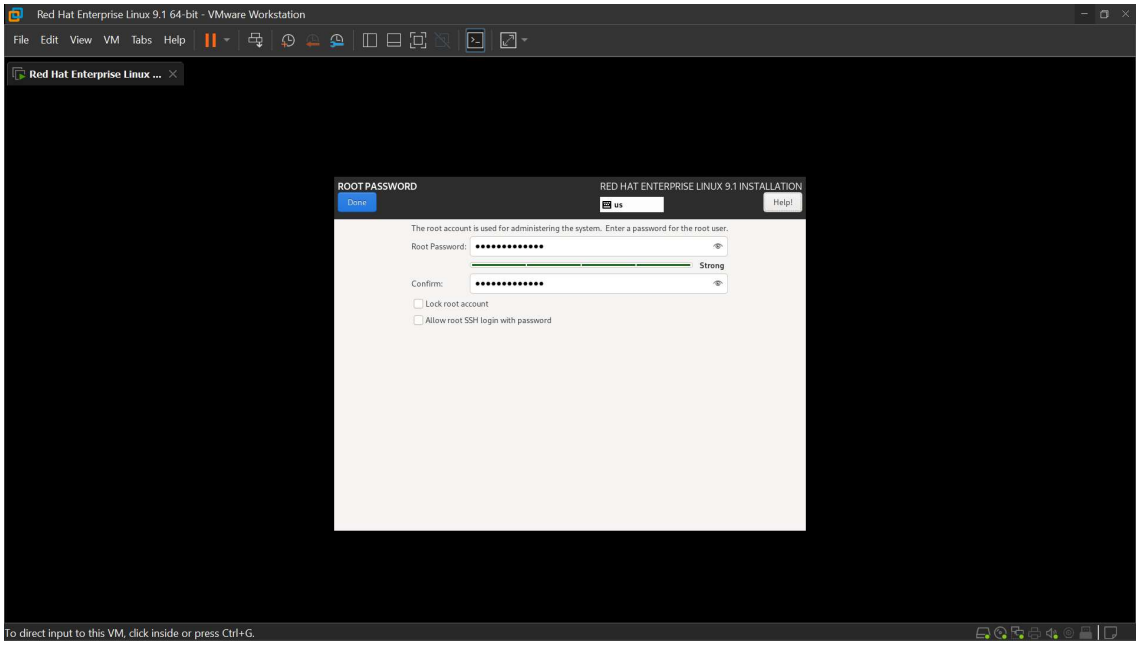

Red Hat Enterprise Linux 9.1 64-bit - VMware Workstation
File Edit View VM Tabs Help
Red Hat Enterprise Linux ...
Starting installer, one moment...
anaconda 34.25.1.14-1.el9 for Red Hat Enterprise Linux 9.1 started.
* installation log files are stored in /tmp during the installation
* shell is available on TTY2
* when reporting a bug add logs from /tmp as separate text/plain attachments
anaconda11:main* 2:shell 3:log 4:storage-log 5:program-log Switch tab: Alt+Tab | Help: F1
To direct input to this VM, click inside or press Ctrl+G.

- Choose the language.

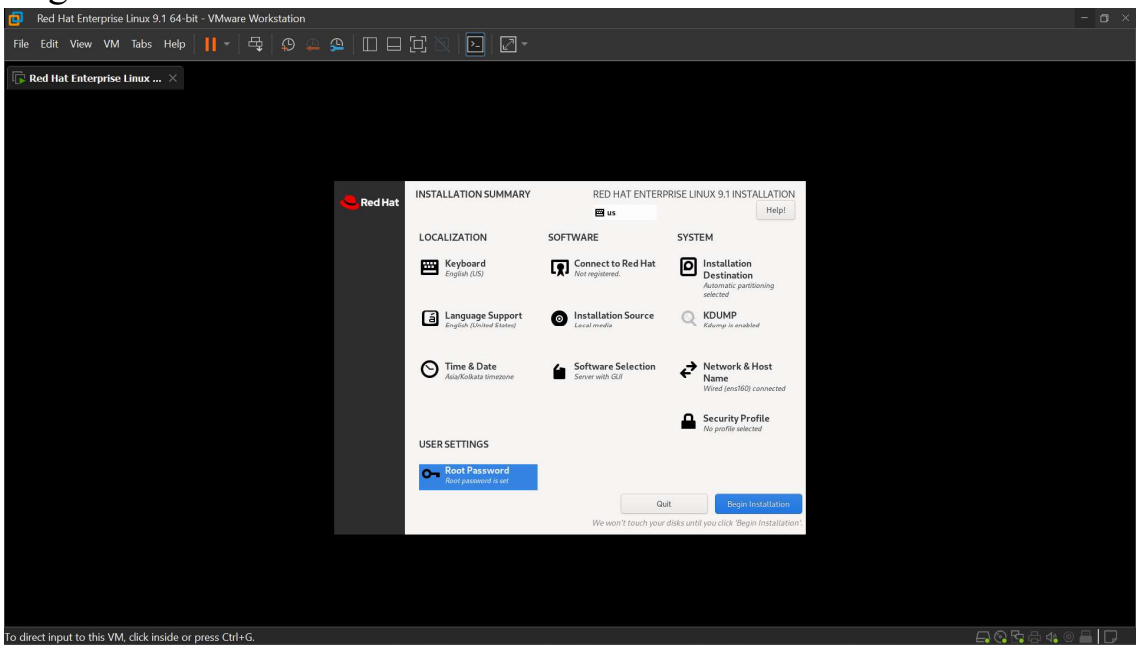


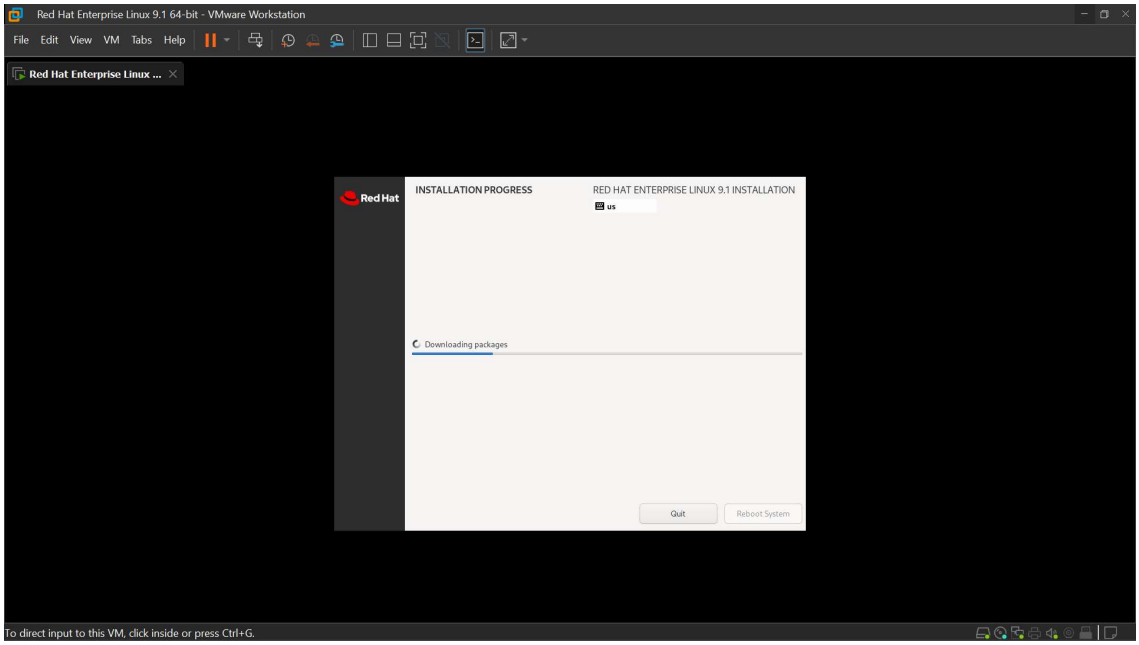
- Set the root password.

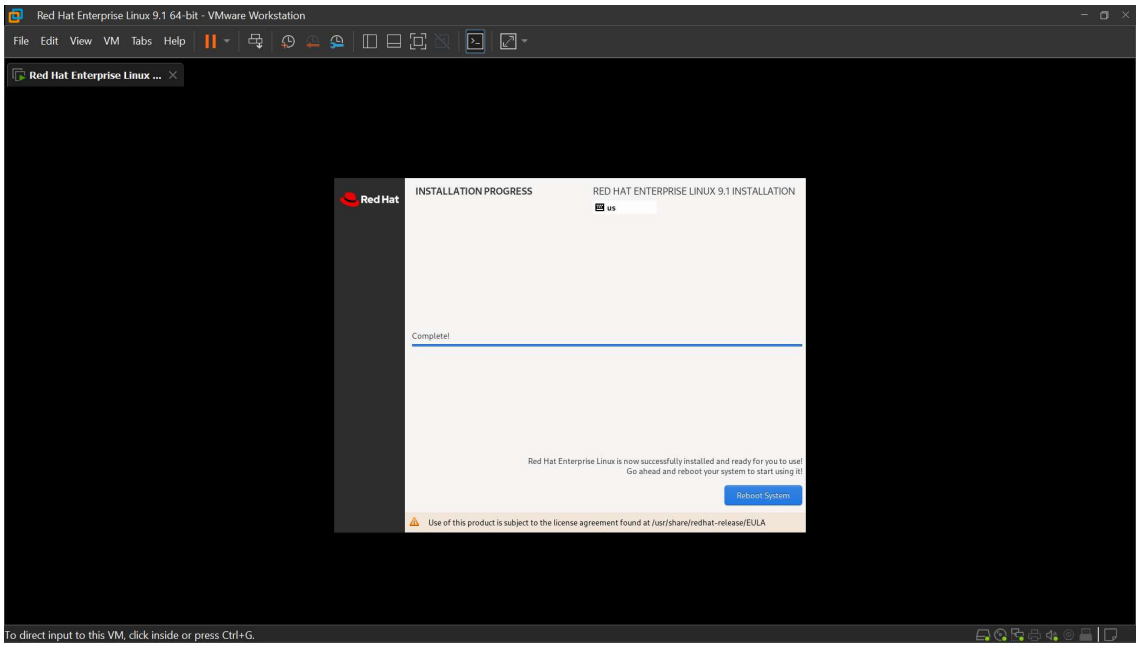


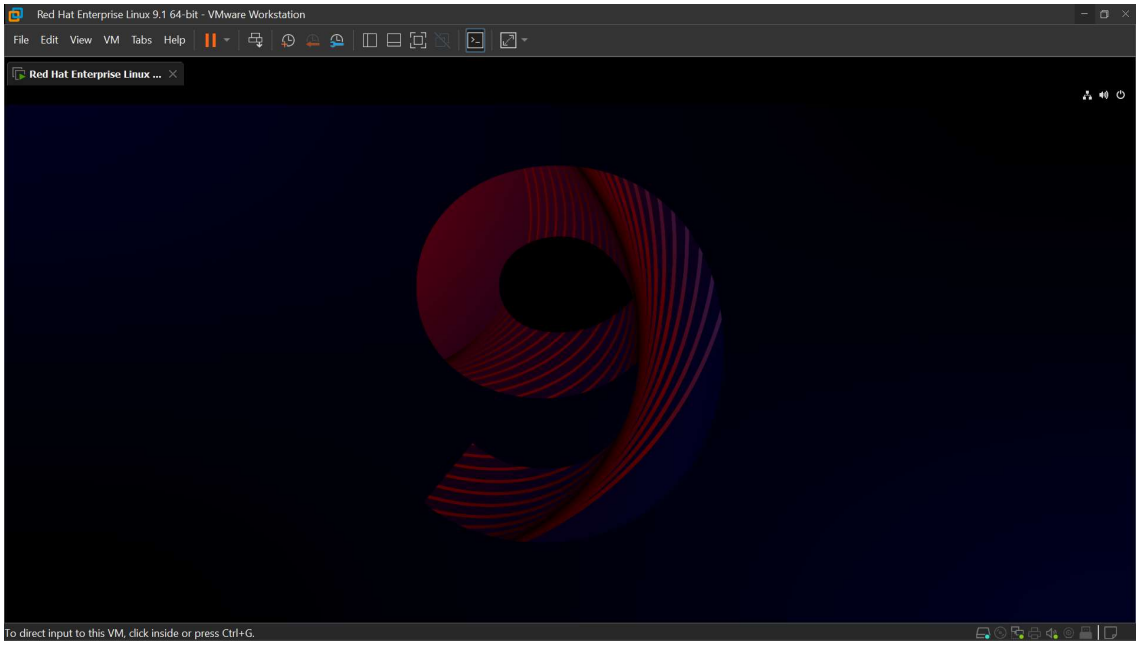


- Begin Installation.

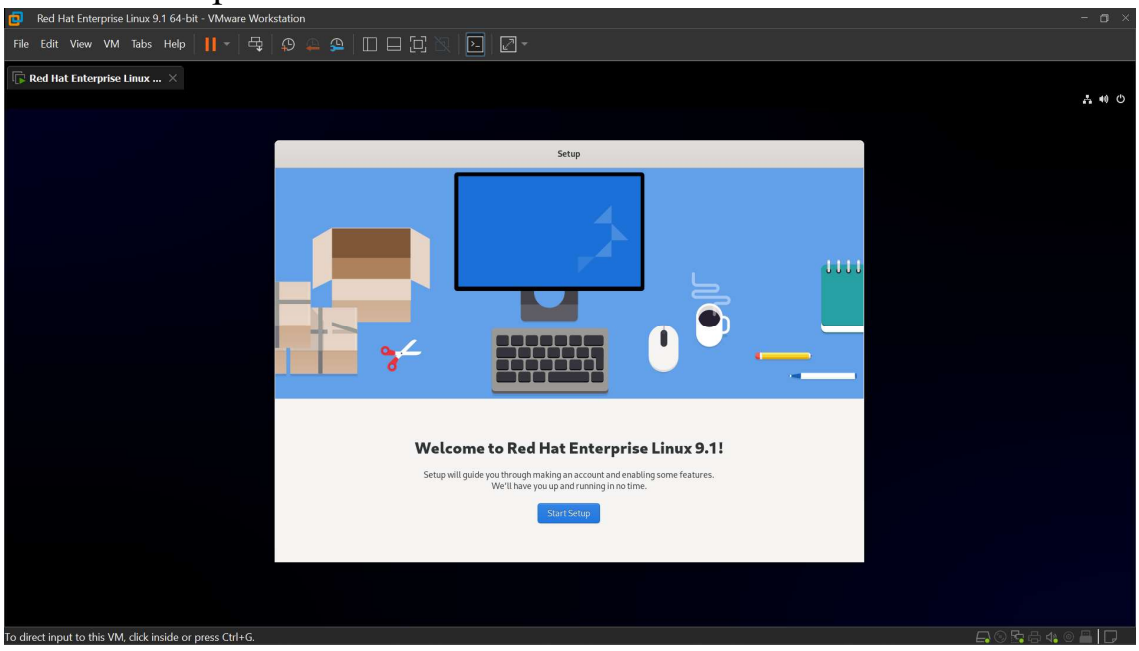


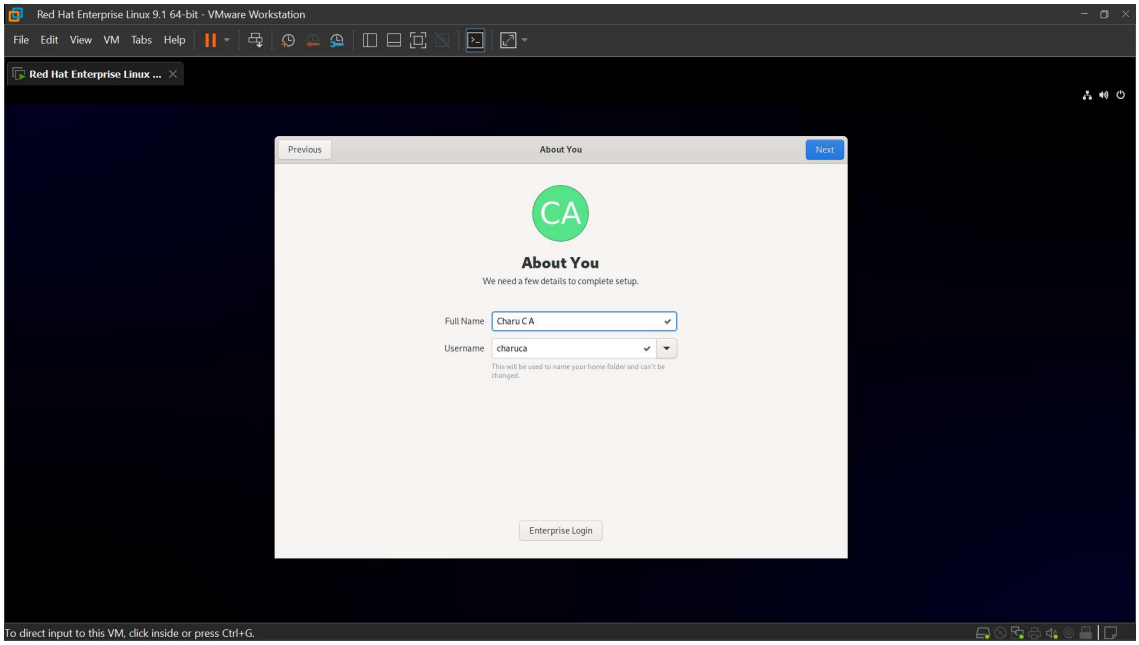

Red Hat Enterprise Linux 9.1 64-bit - VMware Workstation
File Edit View VM Tabs Help
Red Hat Enterprise Linux ...
Red Hat
INSTALLATION PROGRESS
RED HAT ENTERPRISE LINUX 9.1 INSTALLATION
us
Downloading packages
Quit
Reboot System
To direct input to this VM, click inside or press Ctrl+G.

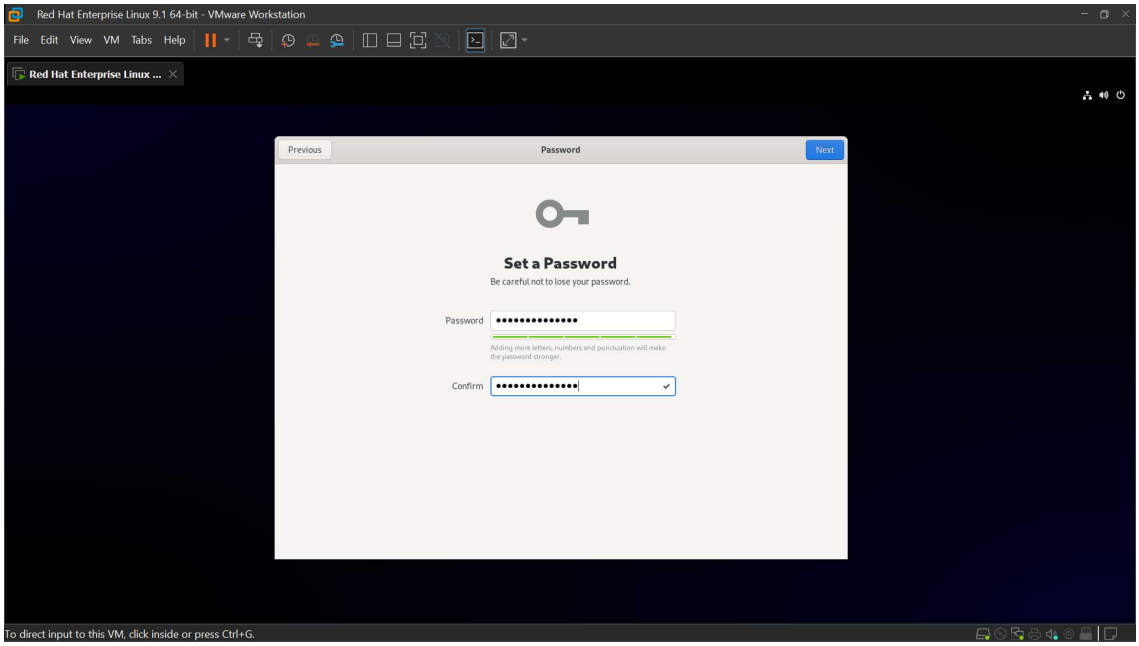

Red Hat Enterprise Linux 9.1 64-bit - VMware Workstation
File Edit View VM Tabs Help
Red Hat Enterprise Linux ...
Red Hat
INSTALLATION PROGRESS
RED HAT ENTERPRISE LINUX 9.1 INSTALLATION
us
Complete!
Red Hat Enterprise Linux is now successfully installed and ready for you to use!
Go ahead and reboot your system to start using it!
Reboot System
Use of this product is subject to the license agreement found at /usr/share/redhat-release/EULA
To direct input to this VM, click inside or press Ctrl+G.

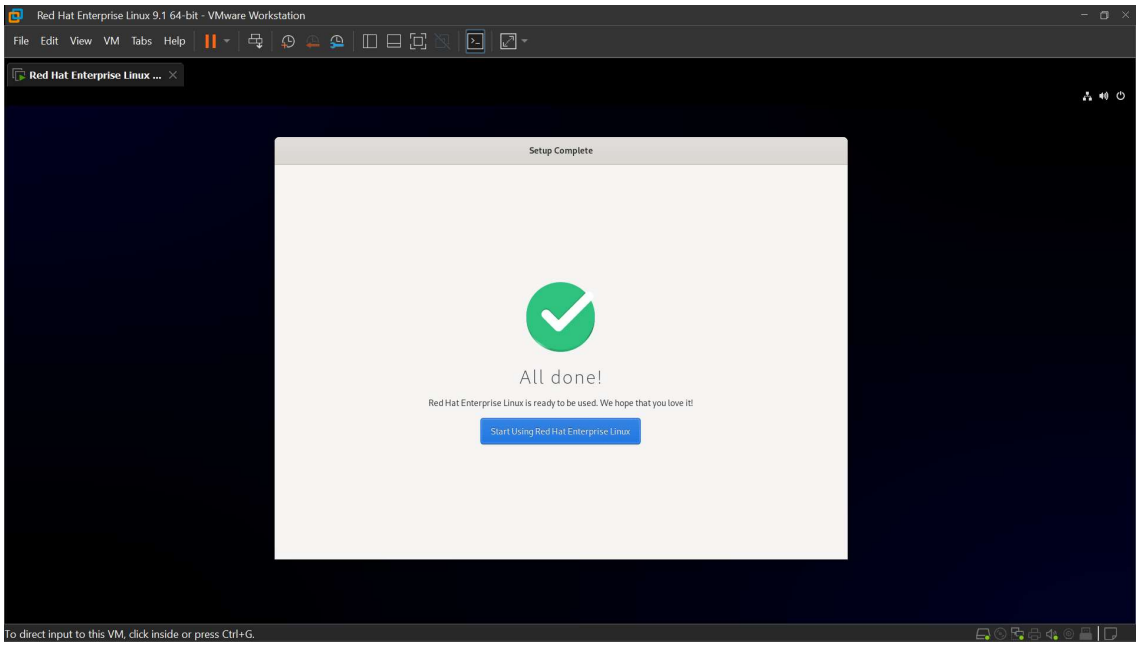


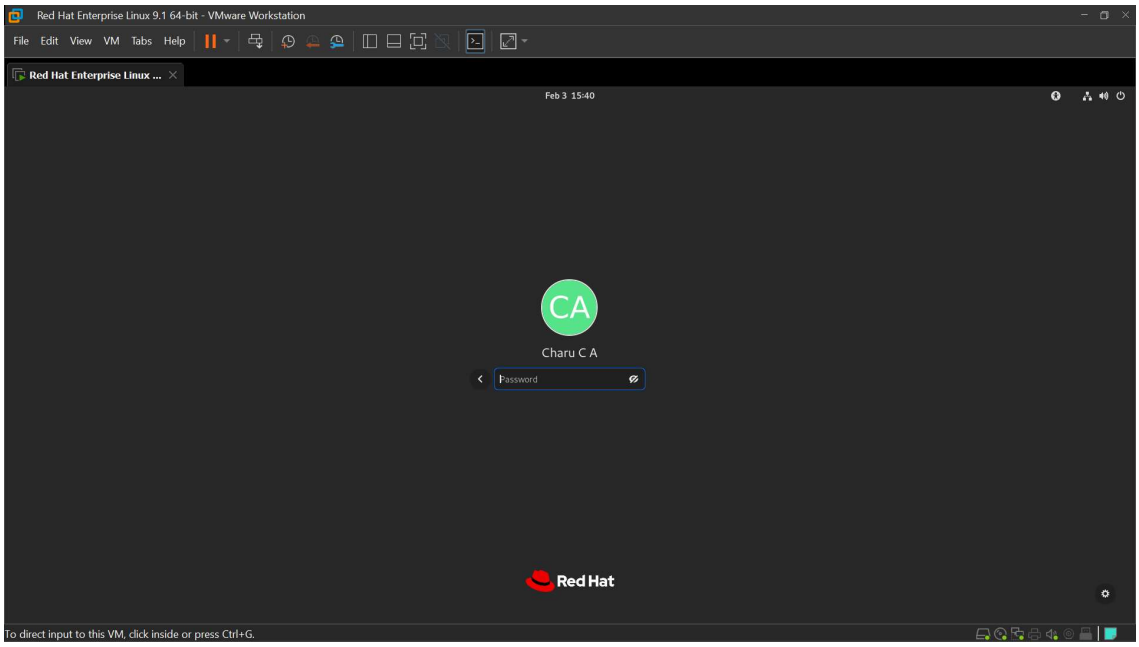
- Start the setup.

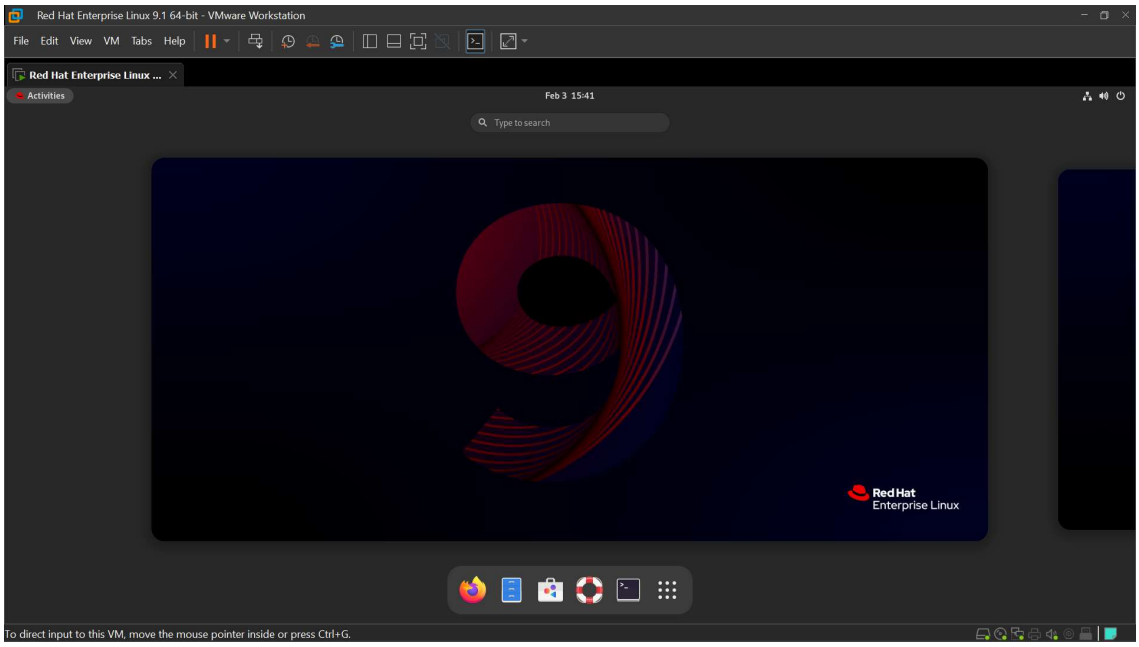



Red Hat Enterprise Linux 9.1 64-bit - VMware Workstation
File Edit View VM Tabs Help
Red Hat Enterprise Linux ...
Previous
About You
Next
CA
About You
We need a few details to complete setup.
Full Name
Charu C A
Username
charuca
This will be used to name your home folder and can't be changed.
Enterprise Login
To direct input to this VM, click inside or press Ctrl+G.


Red Hat Enterprise Linux 9.1 64-bit - VMware Workstation
File Edit View VM Tabs Help
Red Hat Enterprise Linux ...
Previous
Password
Next
Set a Password
Be careful not to lose your password.
Password
Adding more letters, numbers and punctuation will make the password stronger.
Confirm
To direct input to this VM, click inside or press Ctrl+G.


Red Hat Enterprise Linux 9.1 64-bit - VMware Workstation
File Edit View VM Tabs Help
Red Hat Enterprise Linux ...
Setup Complete
All done!
Red Hat Enterprise Linux is ready to be used. We hope that you love it!
Start Using Red Hat Enterprise Linux
To direct input to this VM, click inside or press Ctrl+G.


Red Hat Enterprise Linux 9.1 64-bit - VMware Workstation
File Edit View VM Tabs Help
Red Hat Enterprise Linux ...
Feb 3 15:40
CA
Charu C A
Password
Red Hat
To direct input to this VM, click inside or press Ctrl+G.



- Red Hat Enterprise Linux 9.1 64-bit is successfully installed.

Result:-

Red Hat Enterprise Linux 9.1 is successfully installed.

Experiment - 2

Aim:-

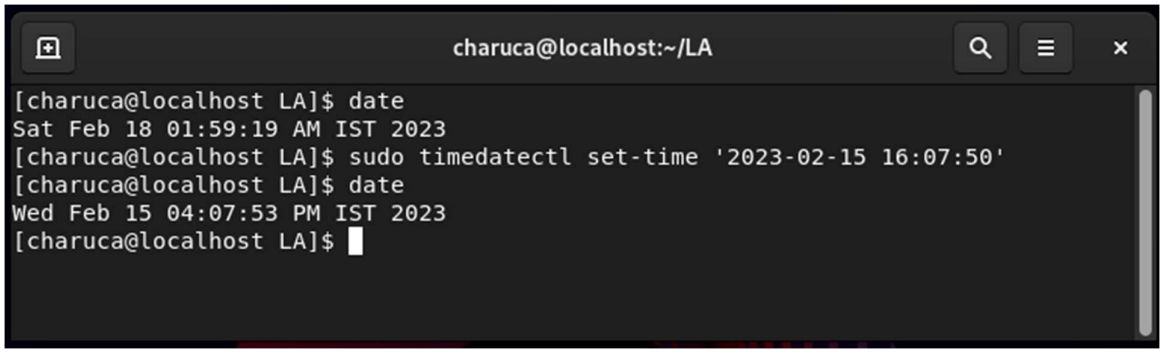
Configure date and time in Linux.

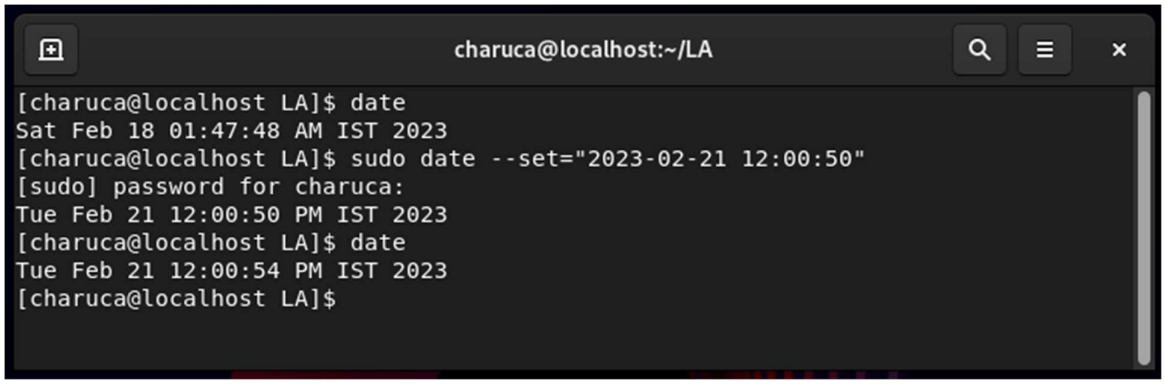
Source Code:-

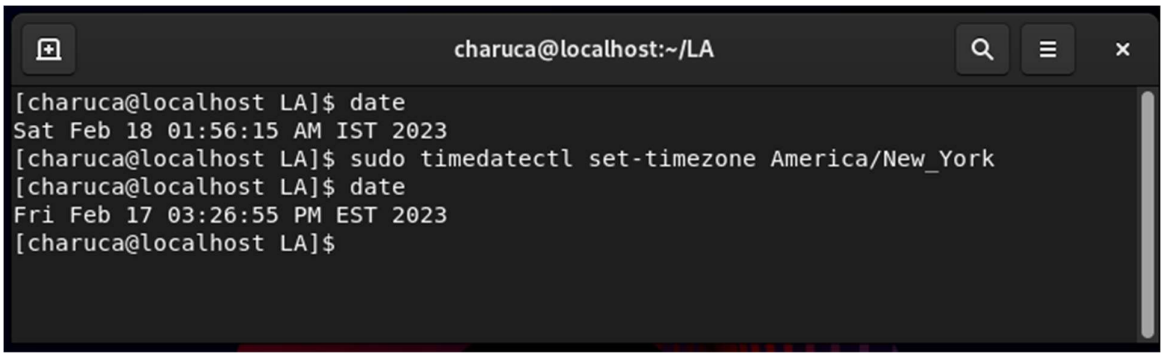
Date and Time can be configured in multiple ways:-

- `sudo timedatectl set-time 'YYYY-MM-DD HH:MM:SS'`
- `sudo date --set="YYYY-MM-DD HH:MM:SS"`
- `sudo timedatectl set-timezone "ZONE/TIMEZONE"`

Output:-







Result:-

Date and time are configured successfully.

Experiment - 3

Aim:-

Create, copy, move, link, unlink and remove the types of files such as

a. Regular files
b. Directory files
c. Link files
d. Block files

Source Code and Output:-

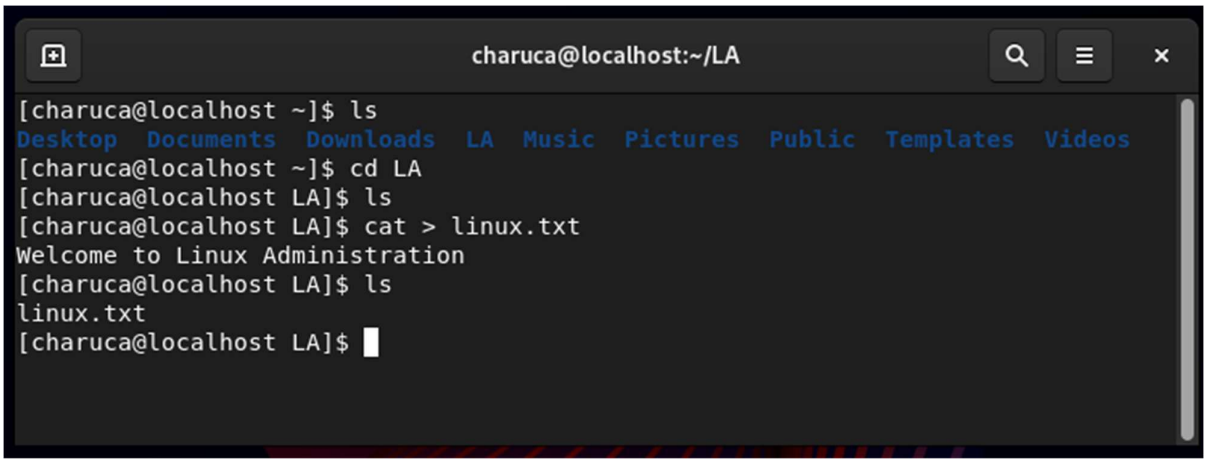
a. Regular files

A "regular file" refers to a file that contains data or text in any format, such as plain text, binary data, images, videos, or any other type of file that is not a directory, device file, or symbolic link.

Creating a file:-

Syntax: cat > filename

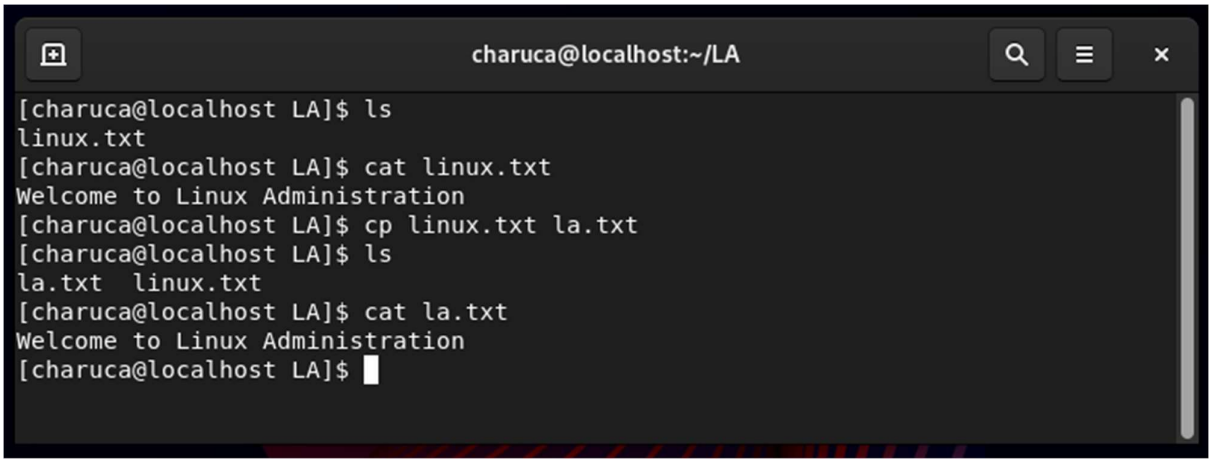
```
cat > linux.txt
Welcome to Linux Administration
^D
```



Copying a file:-

Syntax: cp [options] source_file destination_file

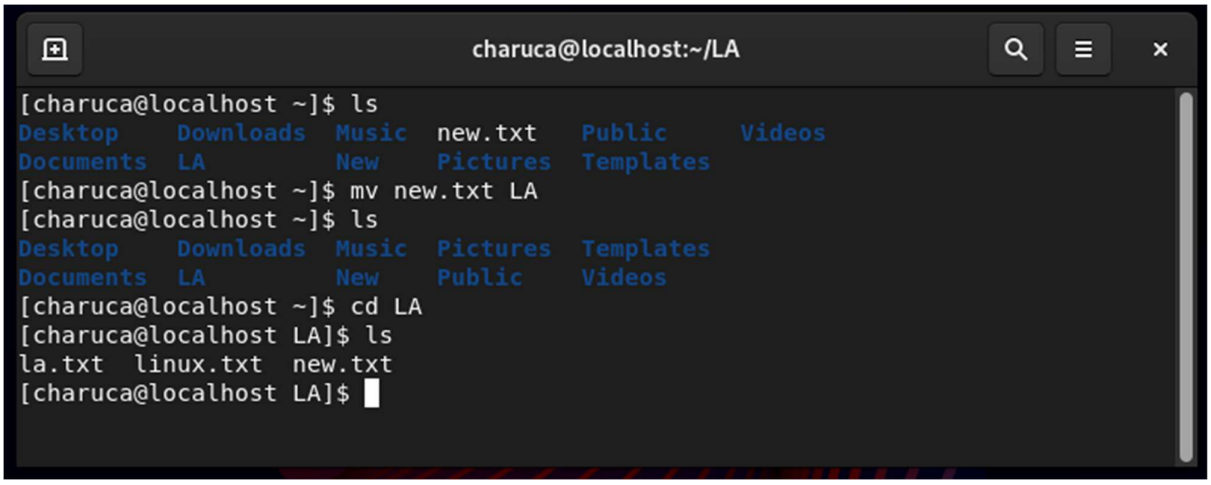
```
cp linux.txt la.txt
```



Moving a file:-

Syntax: mv [options] filename destination

```
mv new.txt LA
```

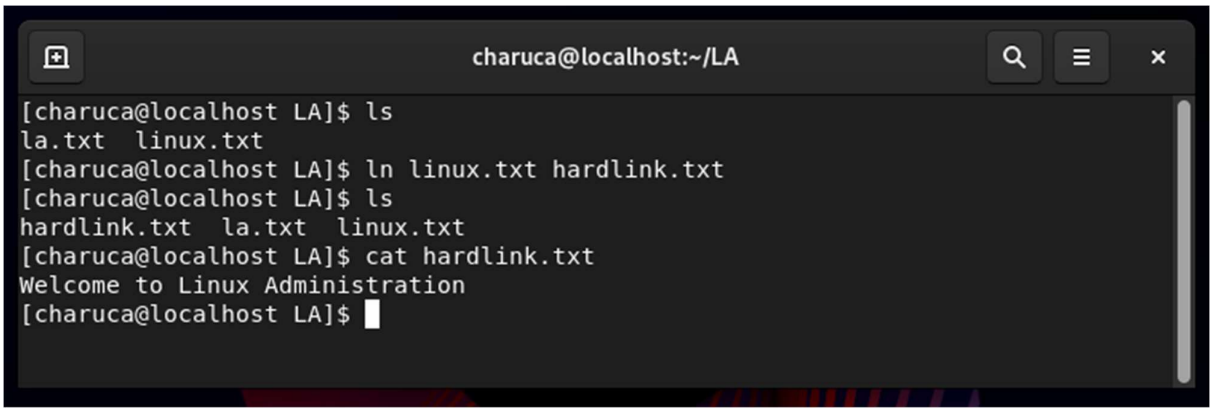


Linking a file:-

Hard Link:-

Syntax: ln source_file link_file

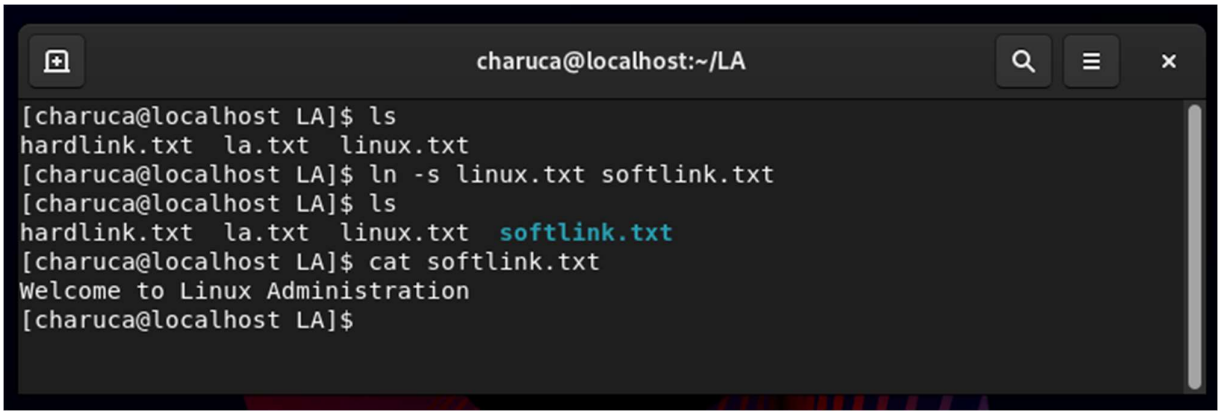
```
ln linux.txt hardlink.txt
```



Soft Link:-

Syntax: ln -s source_file link_file

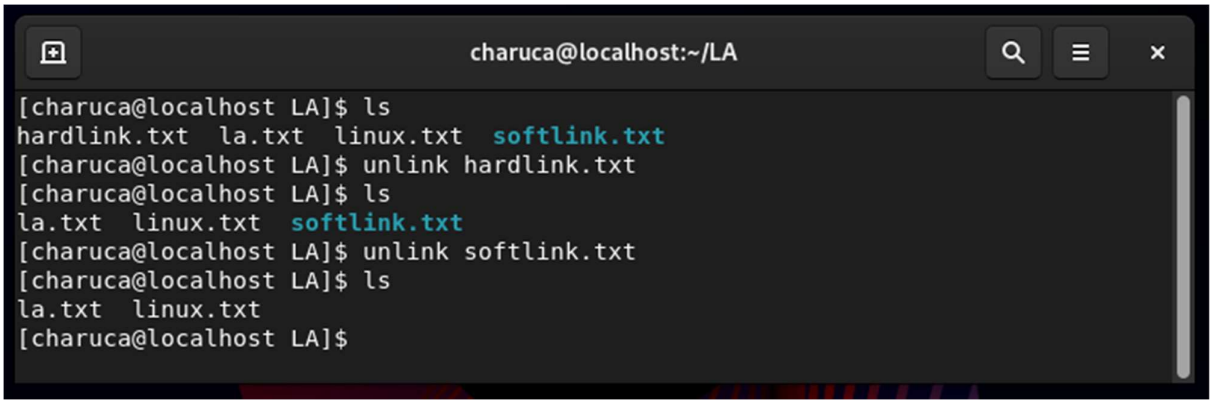
```
ln -s linux.txt softlink.txt
```



Unlinking a file:-

Syntax: unlink filename

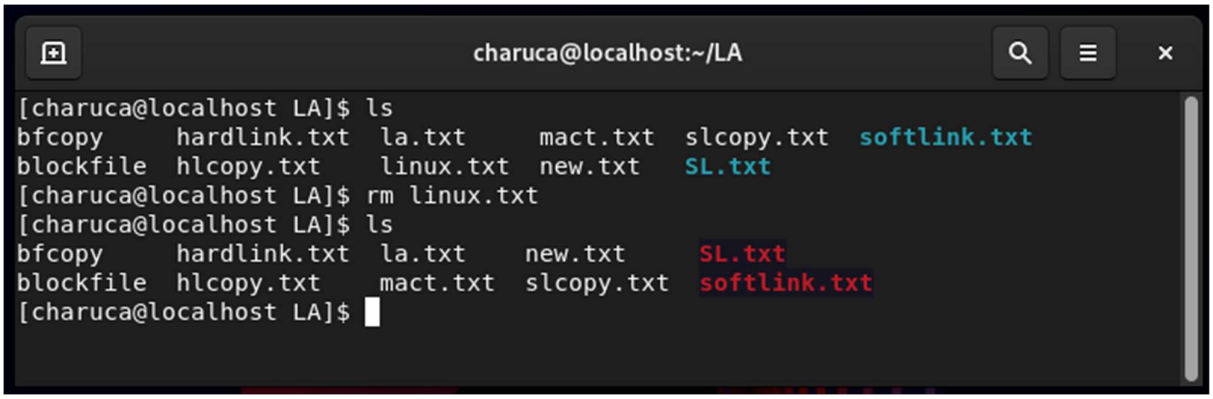
```
unlink hardlink.txt
unlink softlink.txt
```



Removing a file:-

Syntax: rm [options] filename

```
rm linux.txt
```



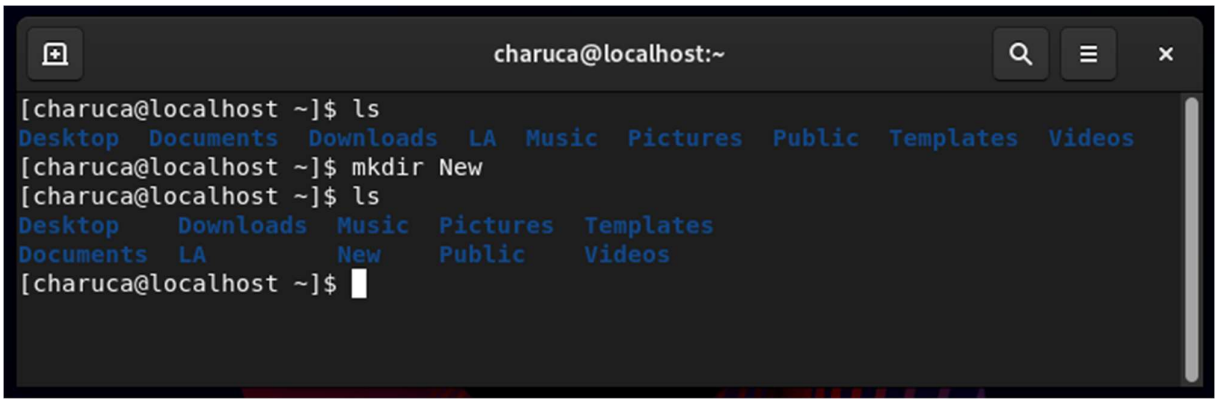
b. Directory files

A directory is a special type of file that contains a list of files and subdirectories within it. Directories are organized in a hierarchical structure, with the root directory ("/") at the top, followed by various subdirectories.

Creating a directory:-

Syntax: mkdir directory

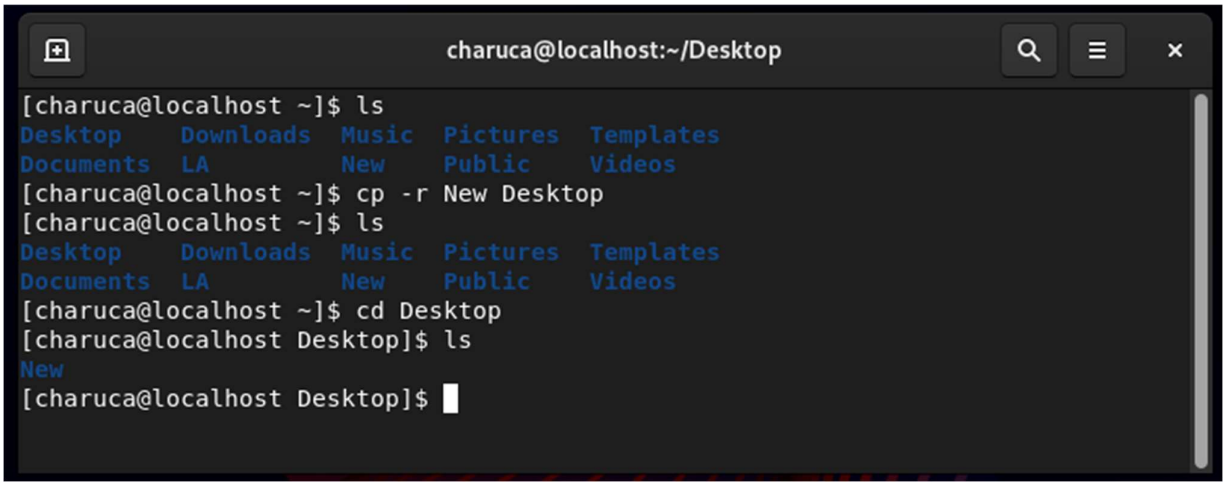
```
mkdir New
```



Copying a directory:-

Syntax: cp [options] source_directory destination_directory

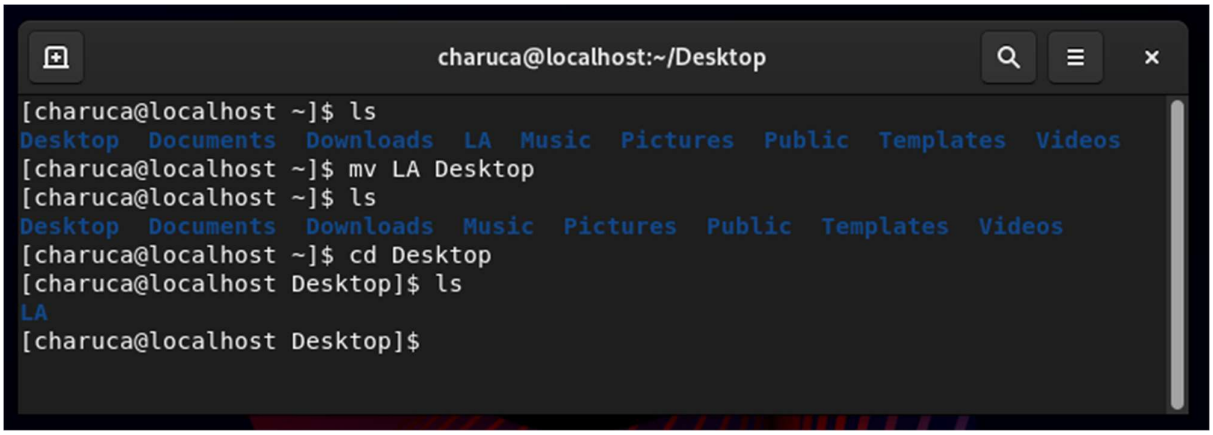
```
cp -r New Desktop
```



Moving a directory:-

Syntax: mv [options] directory destination

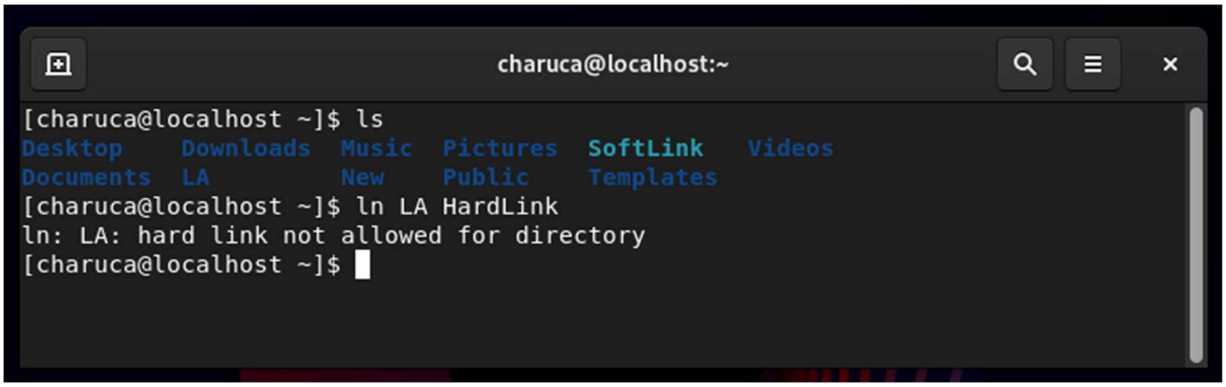
```
mv LA Desktop
```



Linking a directory:-

Hard Link:-

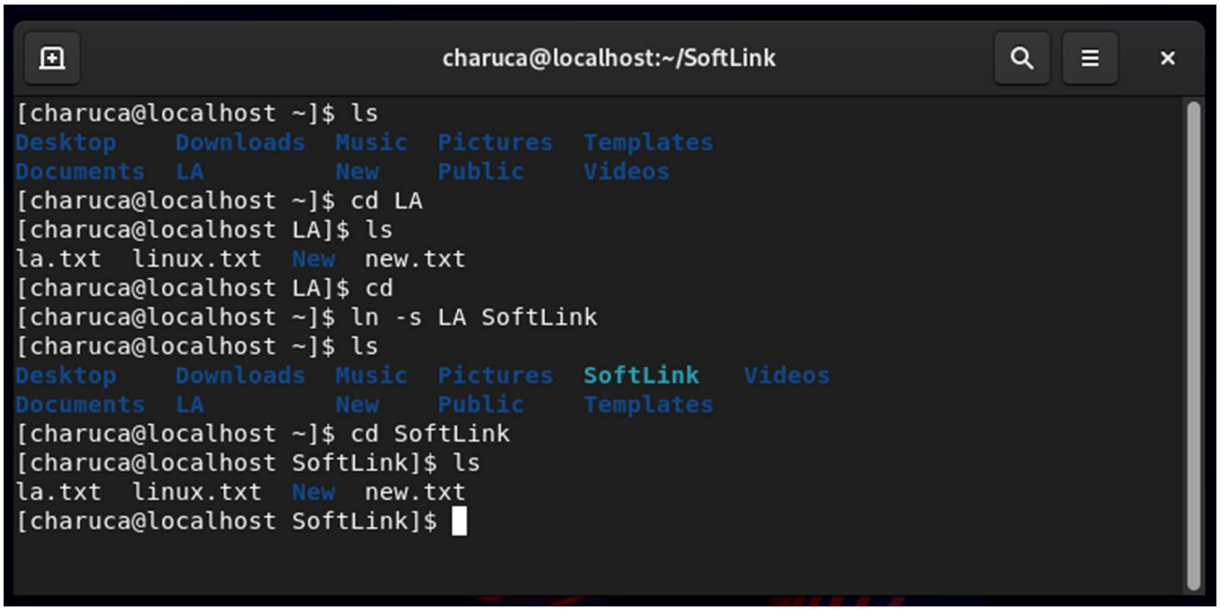
A hard link is not allowed for a directory.



Soft Link:-

Syntax: ln -s source_directory link_directory

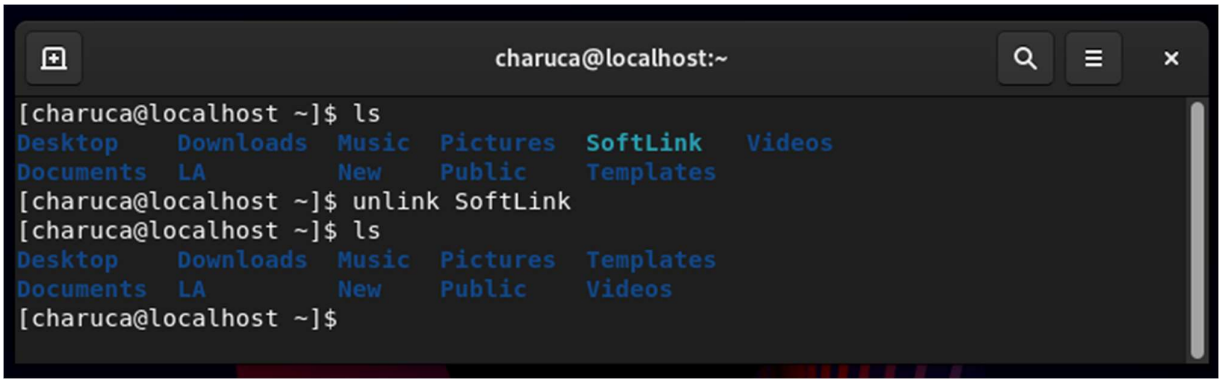
```
ln -s LA SoftLink
```



Unlinking a directory:-

Syntax: unlink directory

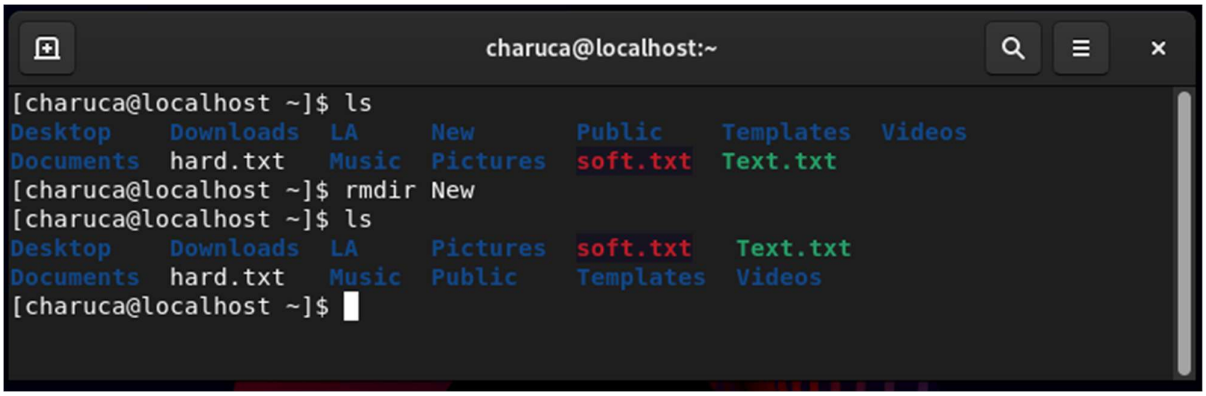
```
unlink SoftLink
```



Removing a directory:-

Syntax: rmdir directory

```
rmdir New
```



c. Link files

A link is a way to create a shortcut to a file or directory. There are two types of links: hard links and soft links.

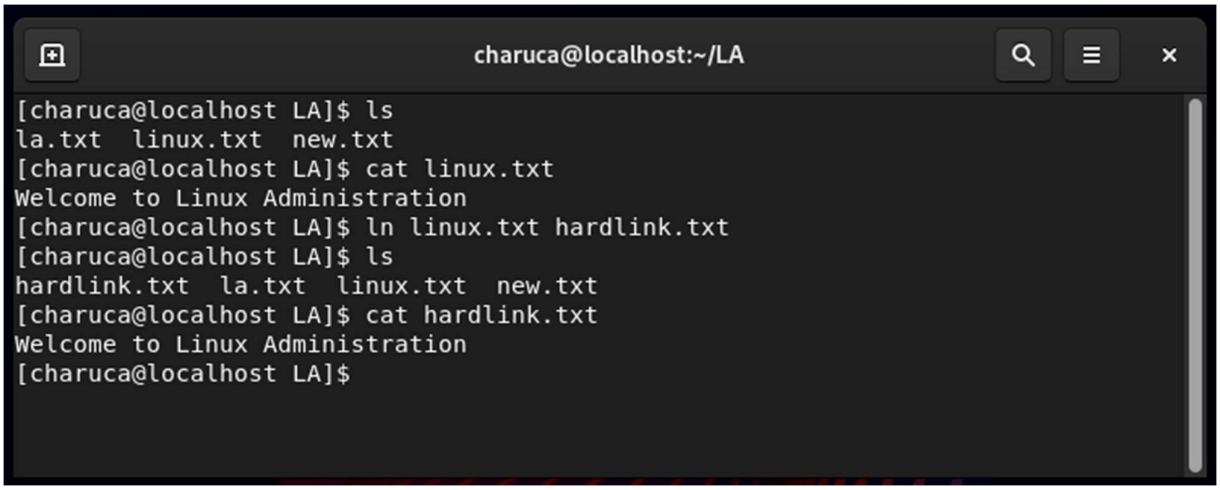
- Hard links: A hard link is a direct reference to a file's inode (i.e. the data structure that stores information about the file).
- Soft links: A soft link (or "symbolic link") is a reference to the path of a file or directory. It is essentially a special type of file that contains the path to the original file or directory.

Creating a link file:-

Hard link:-

Syntax: ln source_file link_file

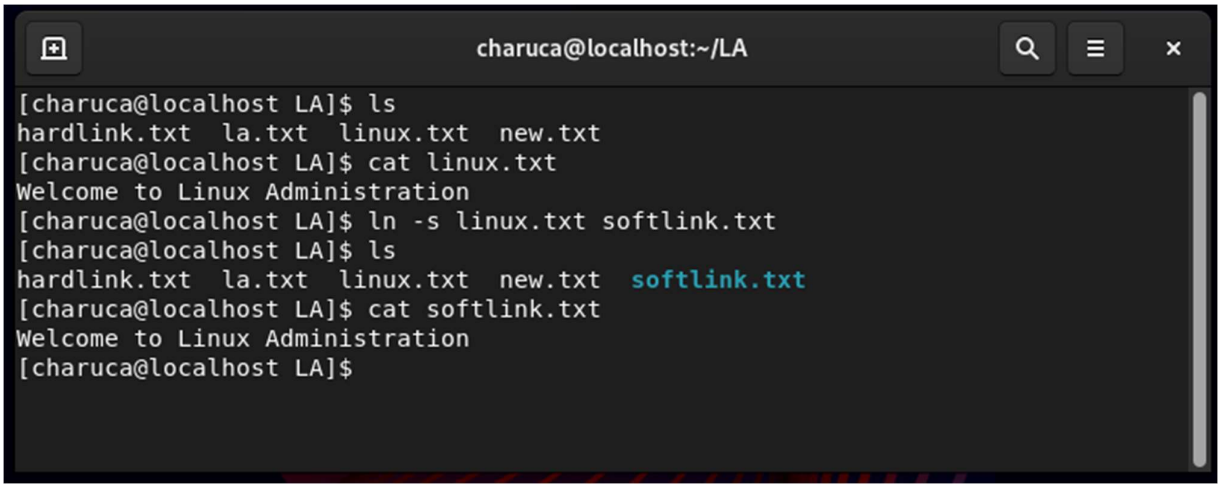
```
ln linux.txt hardlink.txt
```



Soft link:-

Syntax: ln -s source_file link_file

```
ln -s linux.txt softlink.txt
```

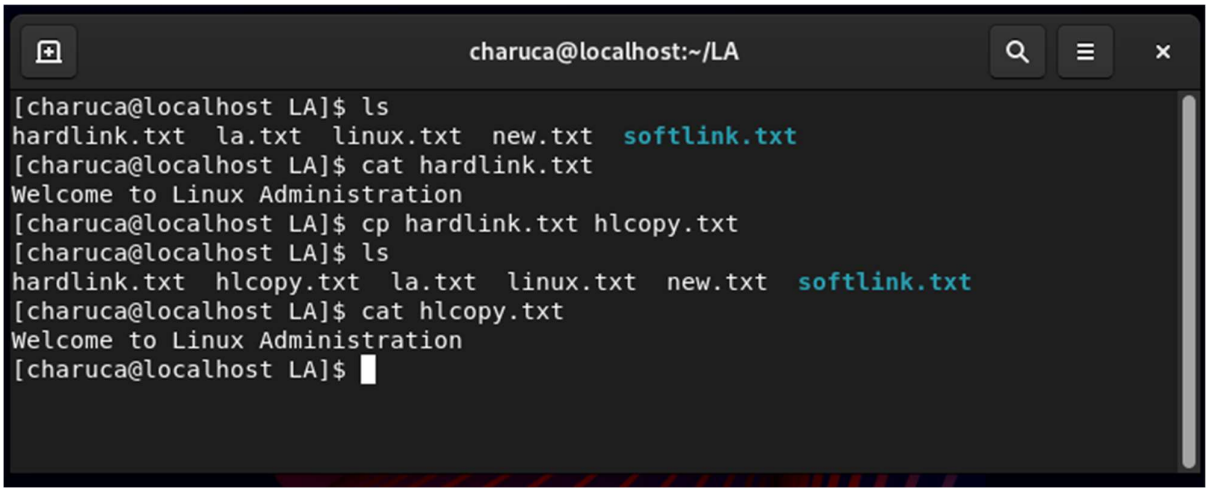


Copying a link file:-

Syntax: cp [options] source_linkfile destination_linkfile

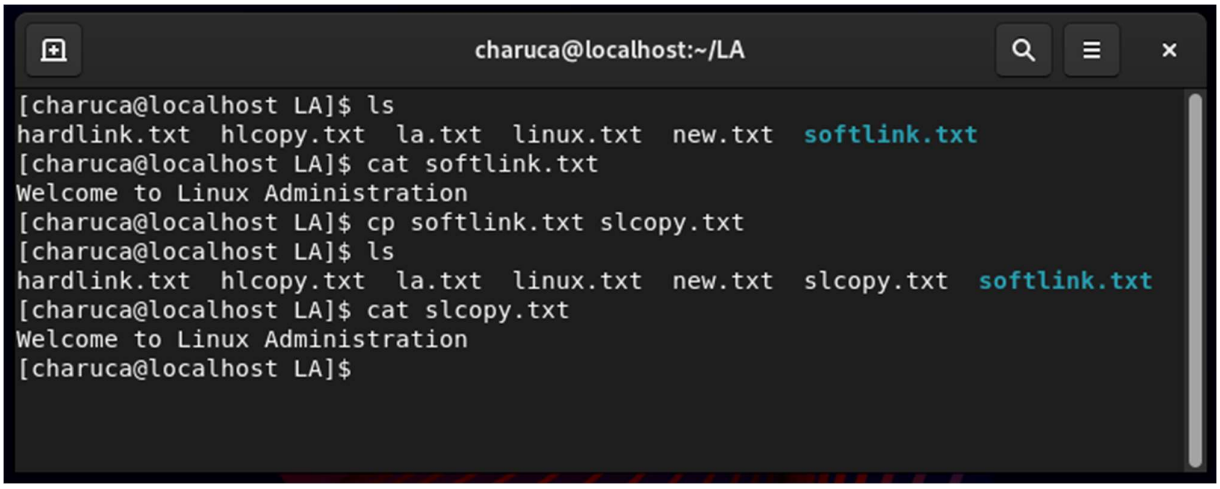
Hard link:-

```
cp hardlink.txt hlcopy.txt
```



Soft link:-

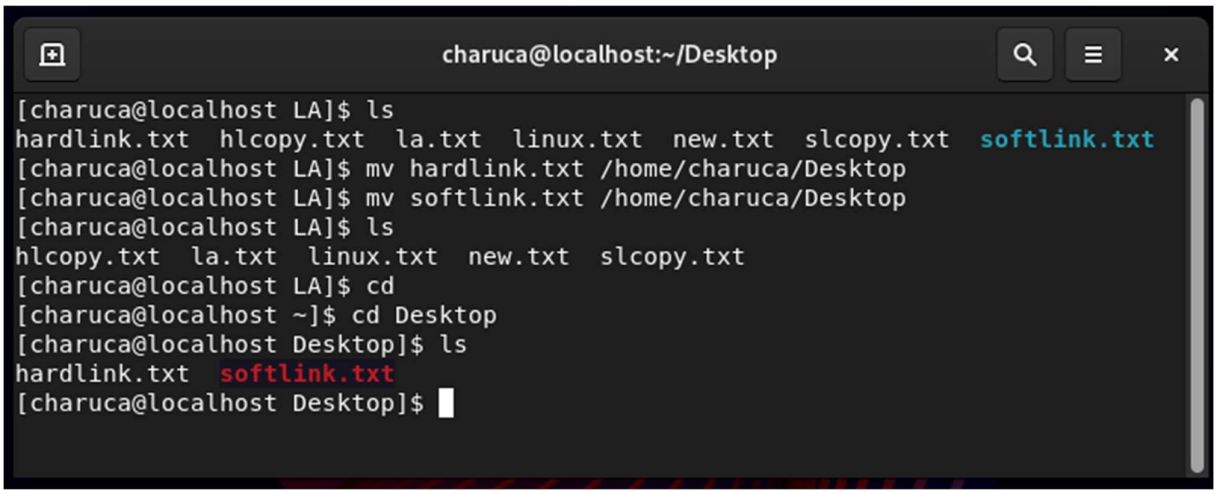
```
cp softlink.txt slcopy.txt
```



Moving a link file:-

Syntax: mv [options] linkfile destination

```
mv hardlink.txt /home/charuca/Desktop
mv softlink.txt /home/charuca/Desktop
```



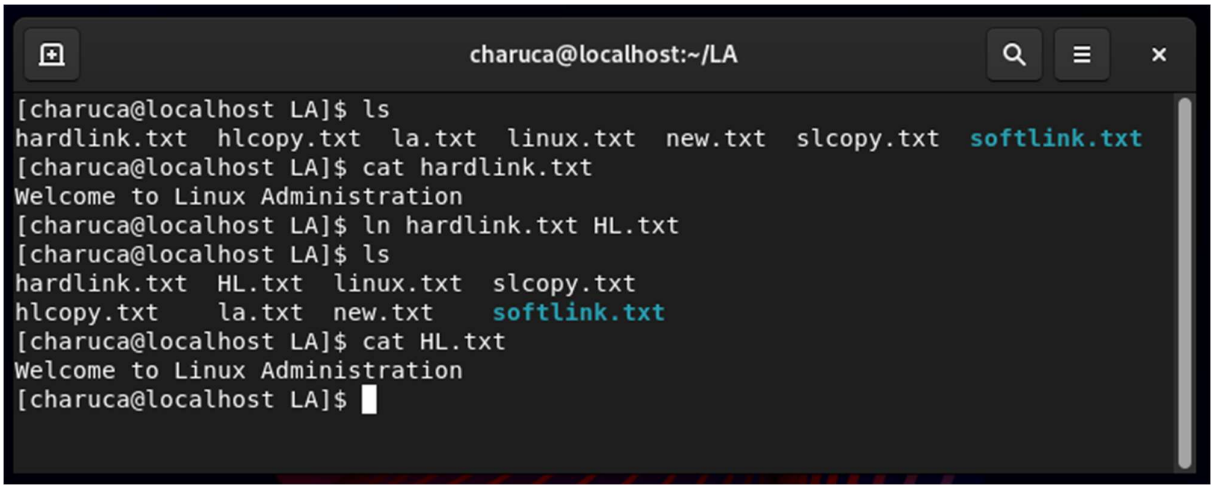
A soft link cannot be used after it is moved because its target file doesn't exist.

Linking a link file:-

Hard Link:-

Syntax: ln source_linkfile link_linkfile

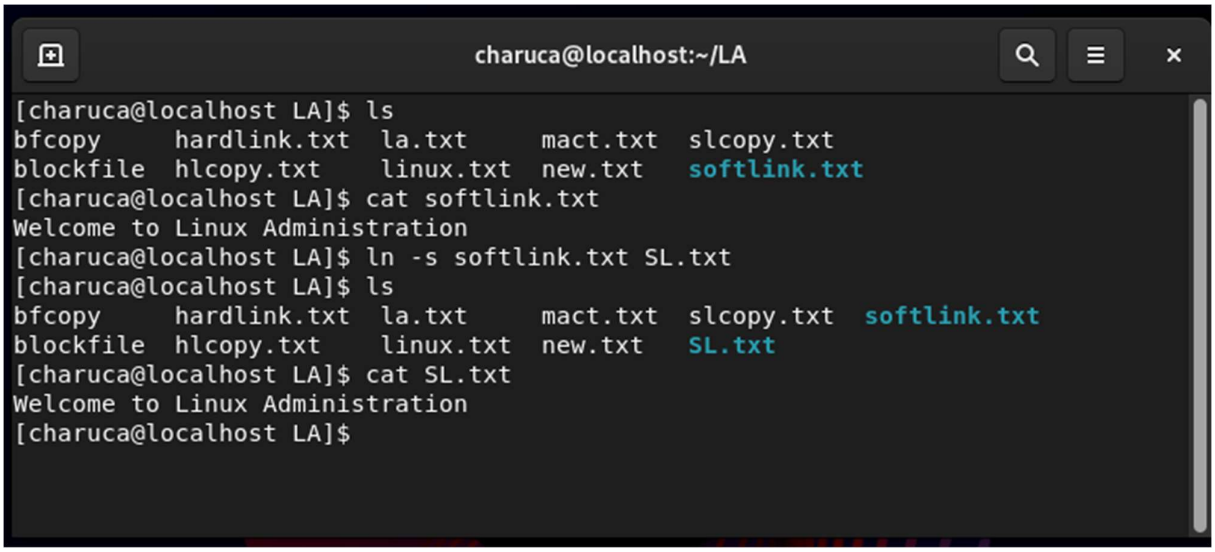
```
ln hardlink.txt HL.txt
```



Soft Link:-

Syntax: ln -s source_linkfile link_linkfile

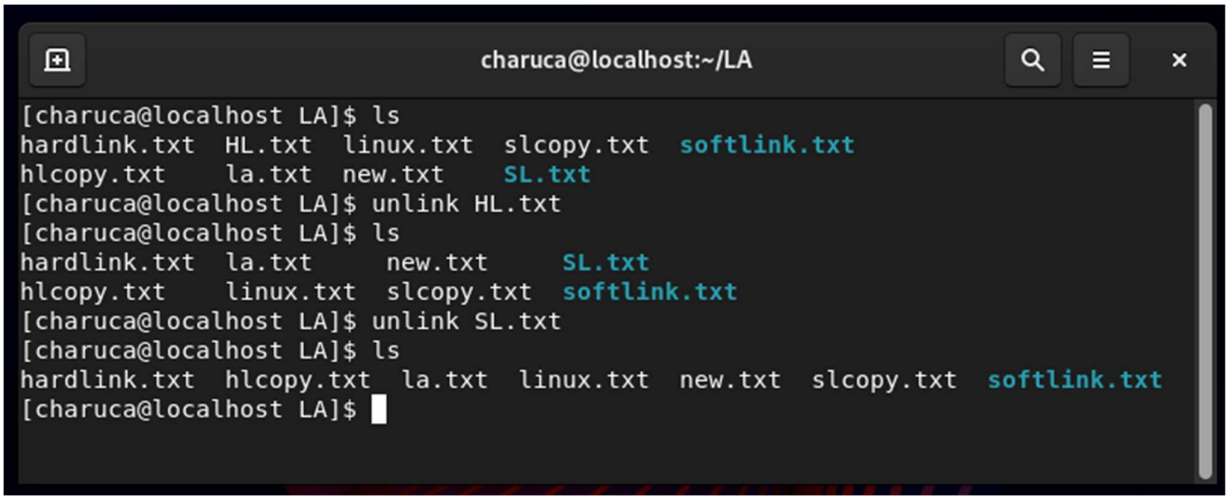
```
ln -s softlink.txt SL.txt
```



Unlinking a link file:-

Syntax: unlink linkfile

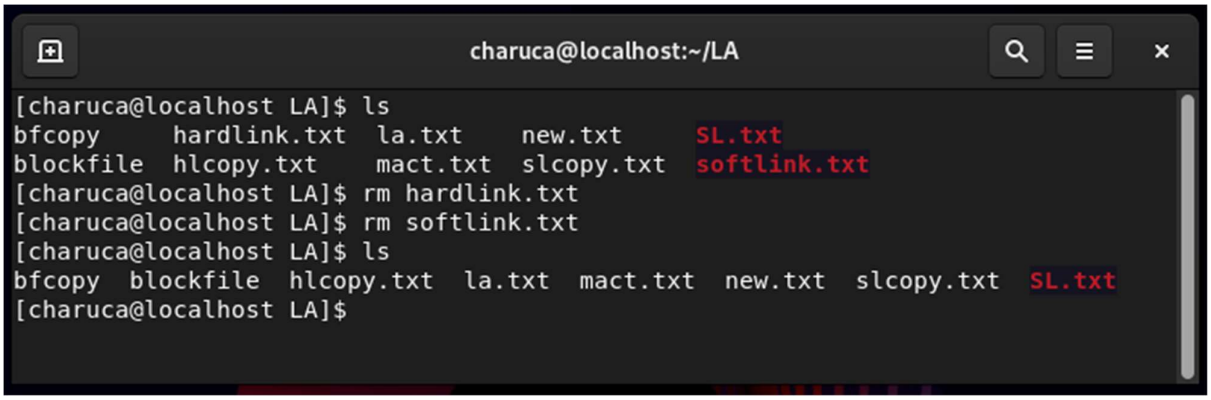
```
unlink HL.txt
unlink SL.txt
```



Removing a link file:-

Syntax: rm [options] linkfile

```
rm hardlink.txt
rm softlink.txt
```



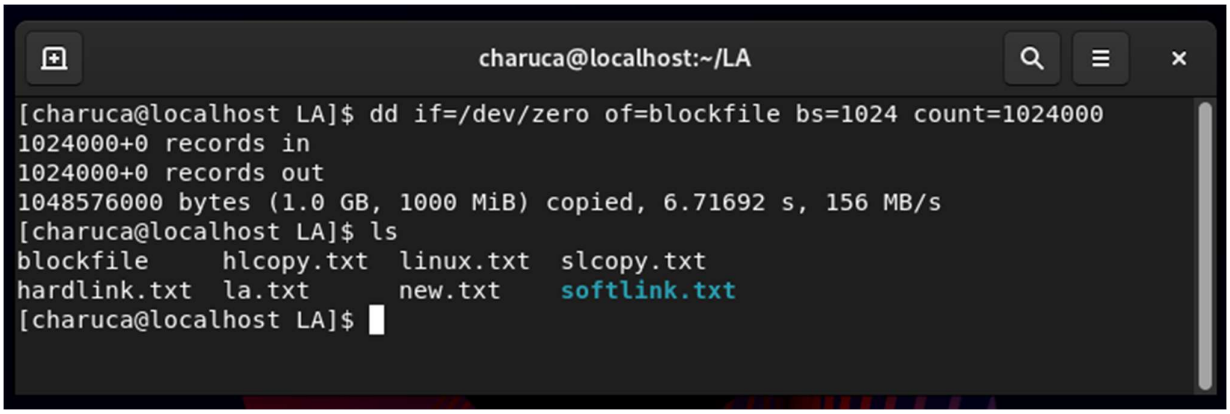
d. Block files

A block file is a special file that provides a way to access a block device (such as a hard drive or USB drive) through the file system.

Creating a block file:-

Syntax: dd if=/dev/zero of=blockfilename bs=1024 count=1024000

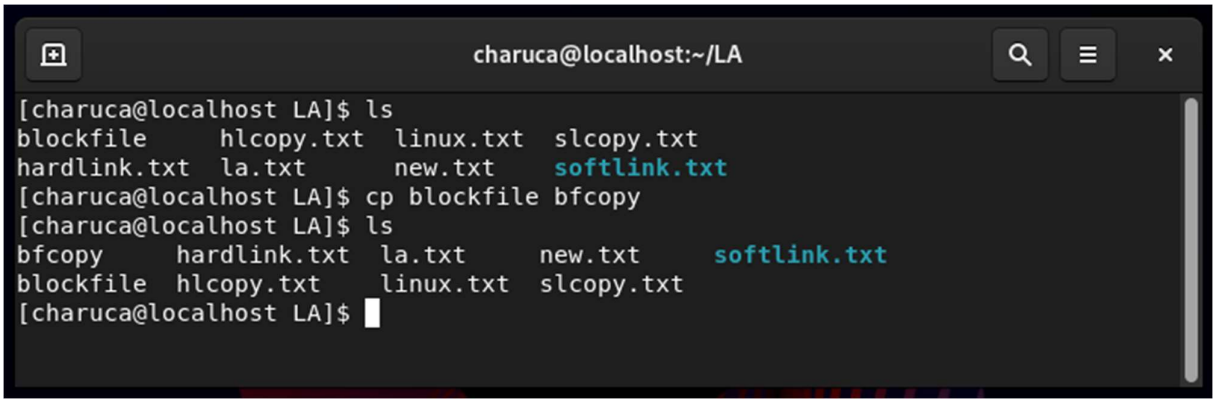
```
dd if=/dev/zero of=blockfile bs=1024 count=1024000
```



Copying a block file:-

Syntax: cp [options] source_blockfile destination_blockfile

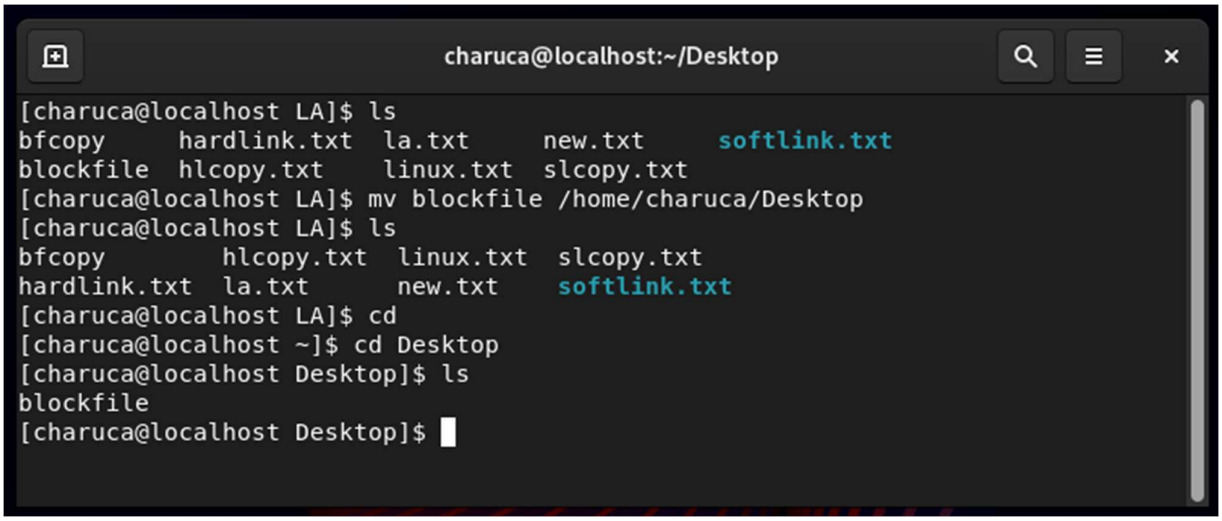
```
cp blockfile bfcopy
```



Moving a block file:-

Syntax: mv [options] blockfile destination

```
mv blockfile /home/charuca/Desktop
```

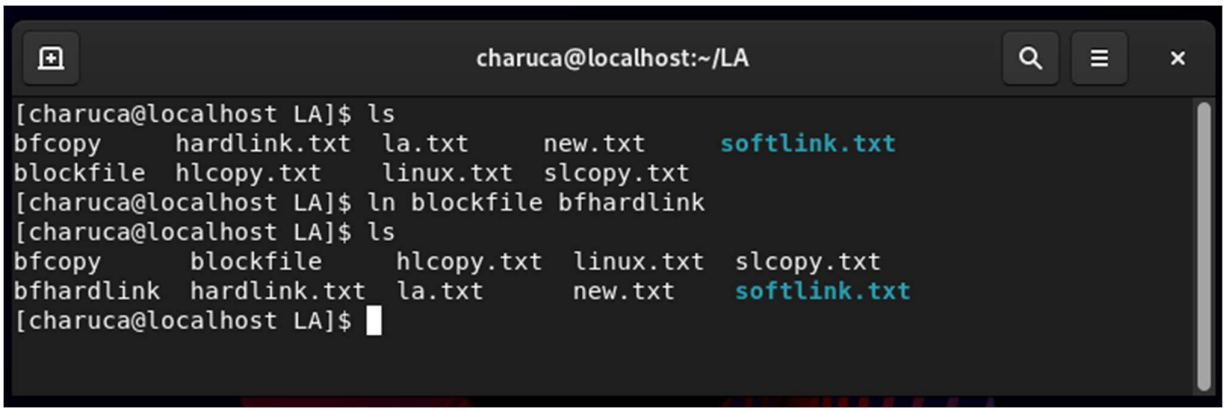


Linking a block file:-

Hard Link:-

Syntax: ln blockfile block_linkfile

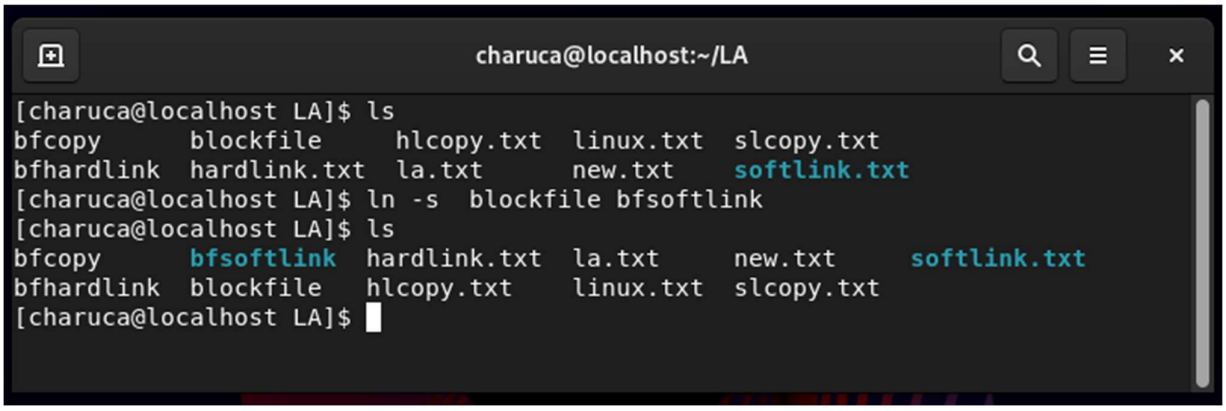
```
ln blockfile bfhardlink
```



Soft Link:-

Syntax: ln -s blockfile block_linkfile

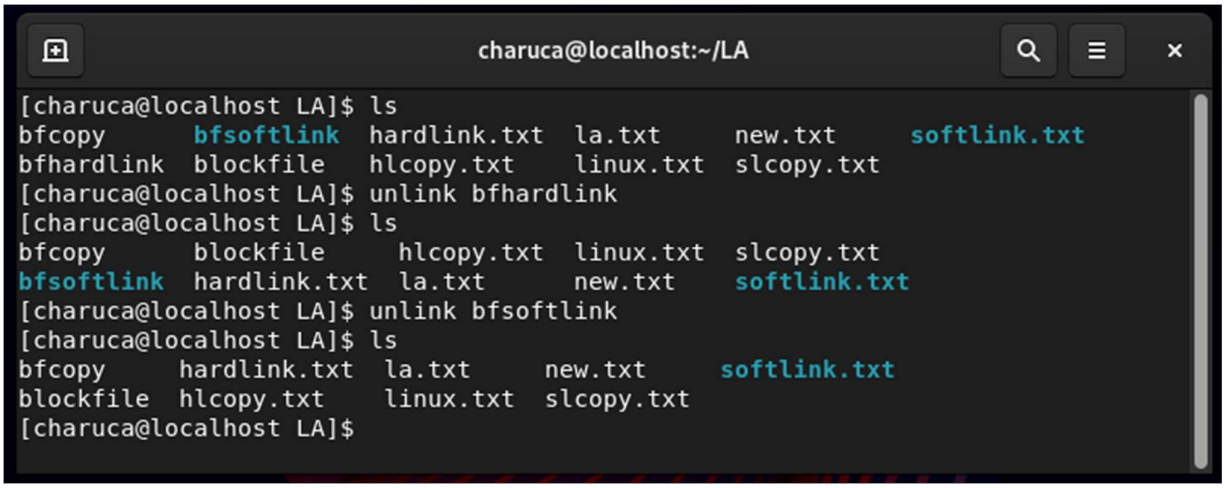
```
ln -s blockfile bfsoftlink
```



Unlinking a block file:-

Syntax: unlink blockfile

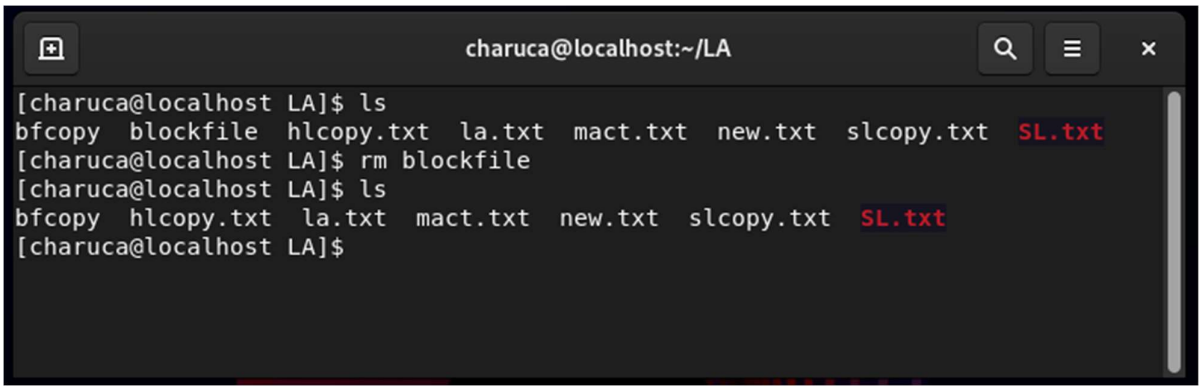
```
unlink bfhardlink
unlink bfsoftlink
```



Removing a block file:-

Syntax: rm [options] blockfile

```
rm blockfile
```



Result:-

Regular files, directory files, link files and block files created, copied, moved, linked, unlinked and removed successfully.

Experiment - 4

Aim:-

List, understand and change the file permission using chmod (numeric method and absolute method) and file ownerships chown.

chmod command

Theory:-

The chmod command is used in Linux to change the permissions of a file or directory. File permissions determine who can access a file and what actions they can perform on it (e.g. read, write, execute).

There are two ways to specify file permissions using chmod: absolute method and numeric method.

› Absolute Method

In absolute method, you use a combination of letters and operators to specify the permissions. The letters rwx represent the read, write, and execute permissions, respectively. The operator + is used to add a permission, - is used to remove a permission, and = is used to set a specific set of permissions.

› Numeric Method

In numeric mode, you use a three-digit number to specify the permissions, where each digit represents the permissions for the owner, group, and others, respectively.

0 = ---	4 = r-
1 = --x	5 = r-x
2 = -w-	6 = rw-
3 = -wx	7 = rwx

Source Code:-

› Absolute Method:-

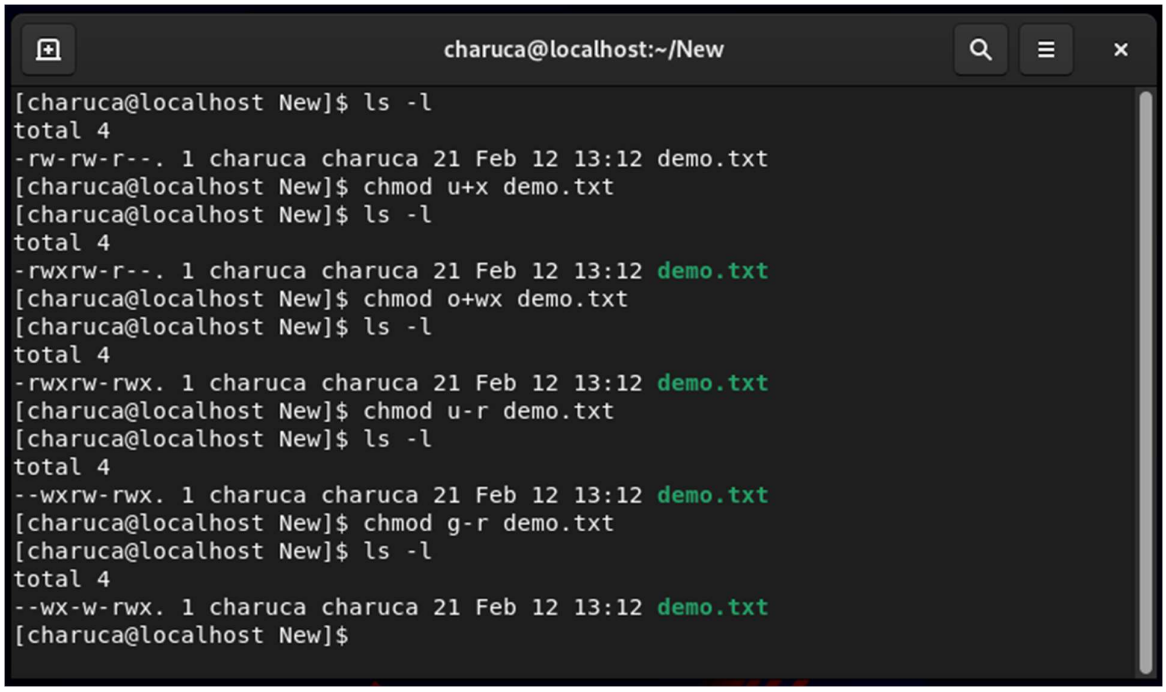
```
chmod u+x demo.txt
chmod o+wx demo.txt
chmod u-r demo.txt
chmod g-r demo.txt
```

› Numeric Method:-

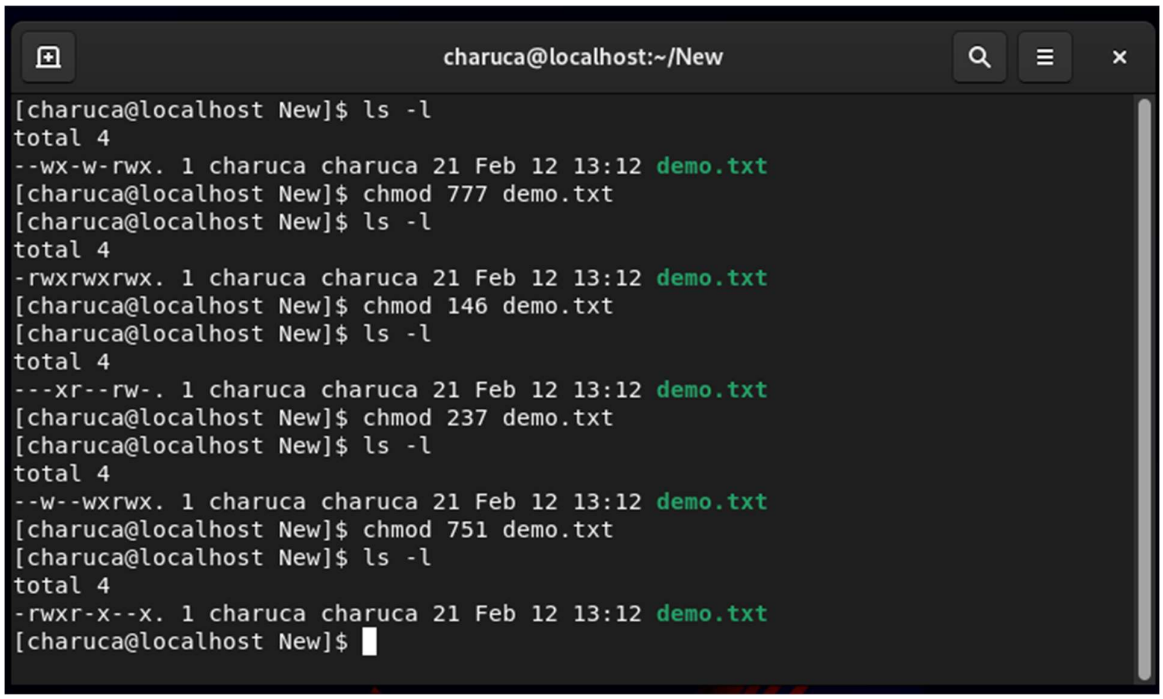
```
chmod 777 demo.txt
chmod 146 demo.txt
chmod 237 demo.txt
chmod 751 demo.txt
```

Output:-

› Absolute Method:-



› Numeric Method:-



chown command

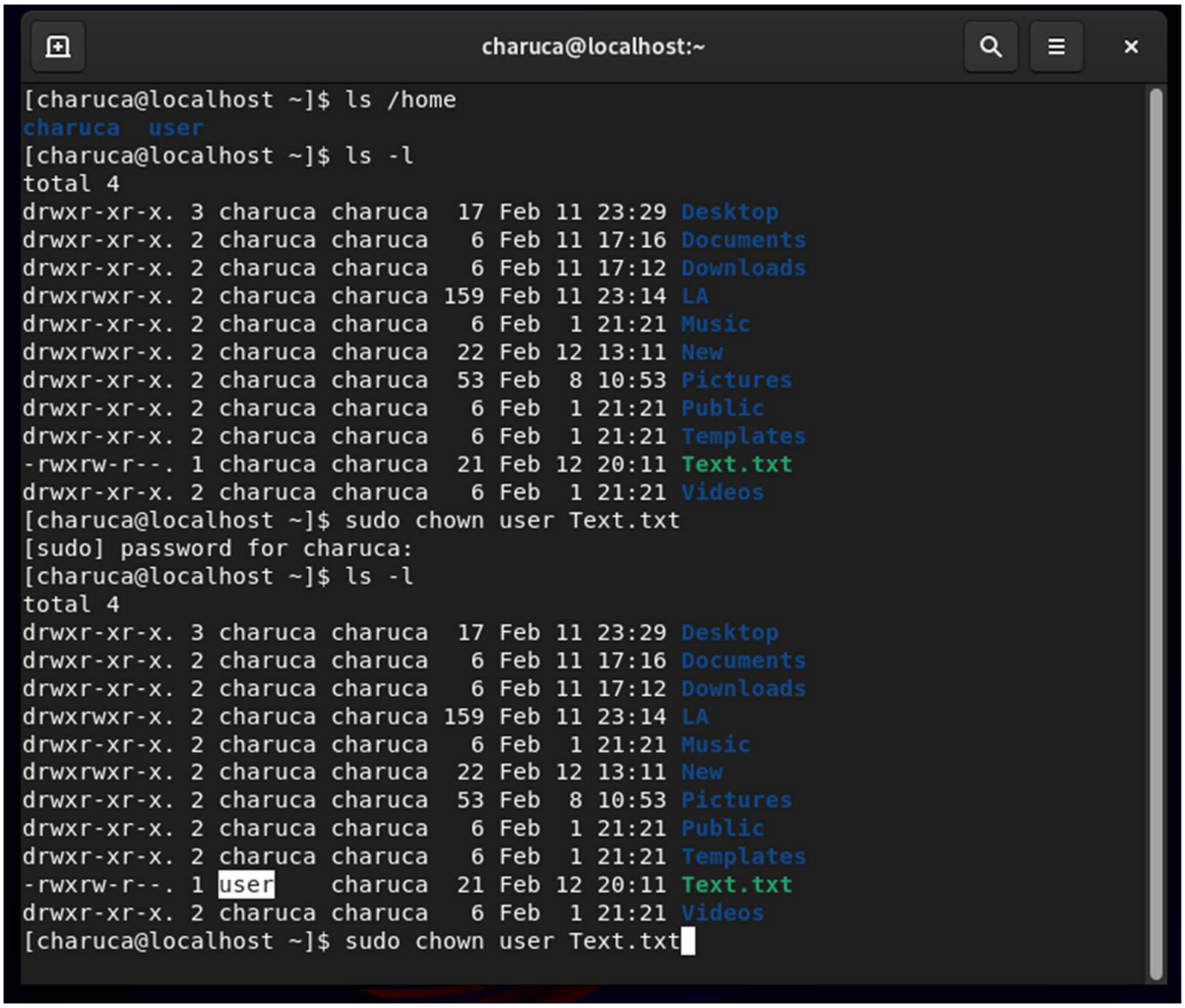
Theory:-

The chown command is used in Linux operating systems to change the owner and group ownership of files and directories. In Linux, every file and directory is associated with an owner and a group, and the chown command allows you to change these associations.

Source Code:-

```
sudo chown user Text.txt
```

Output:-



Result:-

File permissions changed and file ownership changed successfully.

Experiment - 5

Aim:-

List the processes that are running in fore ground and back ground and in zombie state.

Theory:-

In Linux operating system, there are three states that a process can be in:

- Foreground process: A foreground process is a process that is currently running and accepting input from the user. When a user starts a command or program in the terminal, it runs in the foreground by default until it finishes execution or is suspended using a specific key combination (e.g. Ctrl + Z).

- Background process: A background process is a process that runs in the background, meaning that it does not receive input from the user and does not occupy the terminal. Users can start a program in the background by appending an ampersand (&) to the end of the command.

- Zombie process: A zombie process is a process that has completed execution but still has an entry in the process table. This can happen when the parent process has not yet collected the exit status of the child process.

Source Code:-

To list the processes that are currently running, you can use the '`ps command`' with different options:

- Foreground process:-

```
ps - f
```

- Background process:-

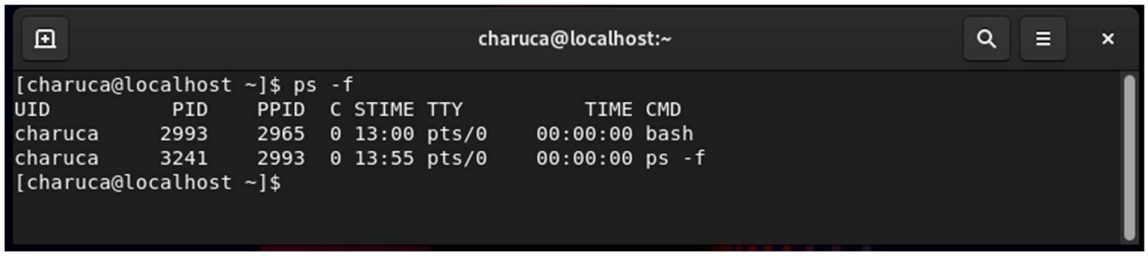
```
ps – ef
```

- Zombie process:-

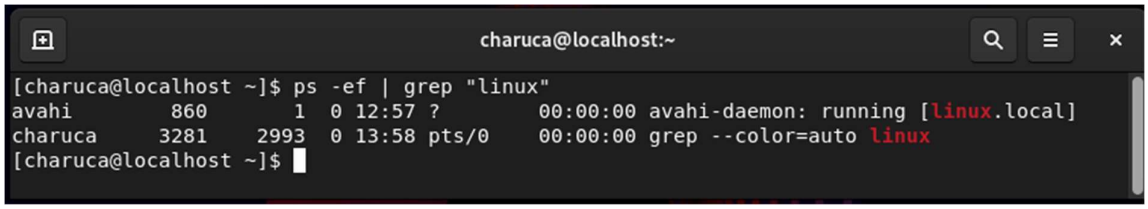
```
ps aux | grep -w Z
```

Output:-

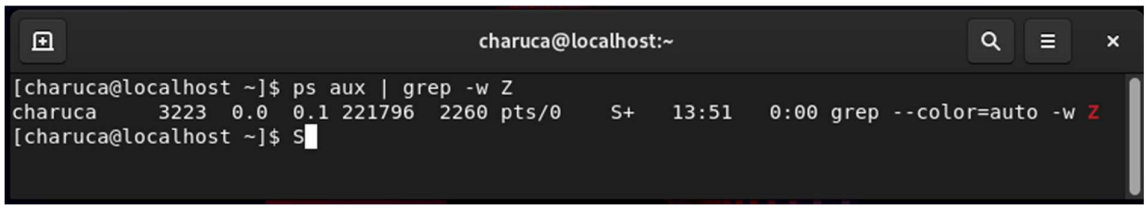
- Foreground process:-



- Background process:-



- Zombie process:-



Result:-

The processes that are running in fore ground and back ground and in zombie state are listed successfully.

Experiment - 6

Aim:-

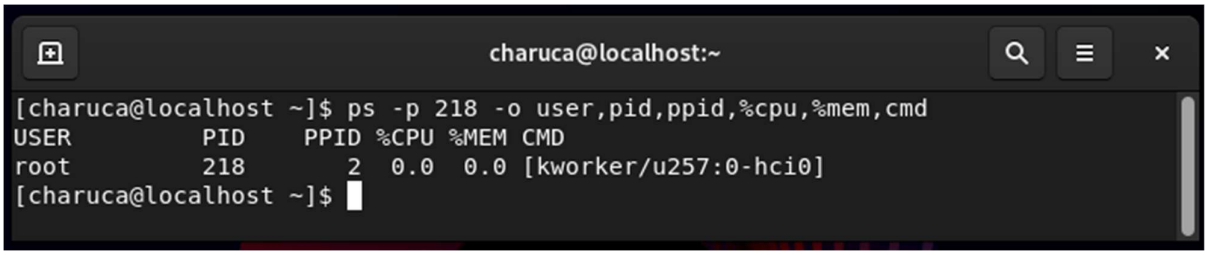
List and display the processes and usage of resources with process id, stop a process temporarily using pid and suspend, kill the process using PID.

Source Code and Output:-

- List the processes and usage of resources with process id.

Syntax : `ps -p <pid> -o pid,ppid,%cpu,%mem,cmd`

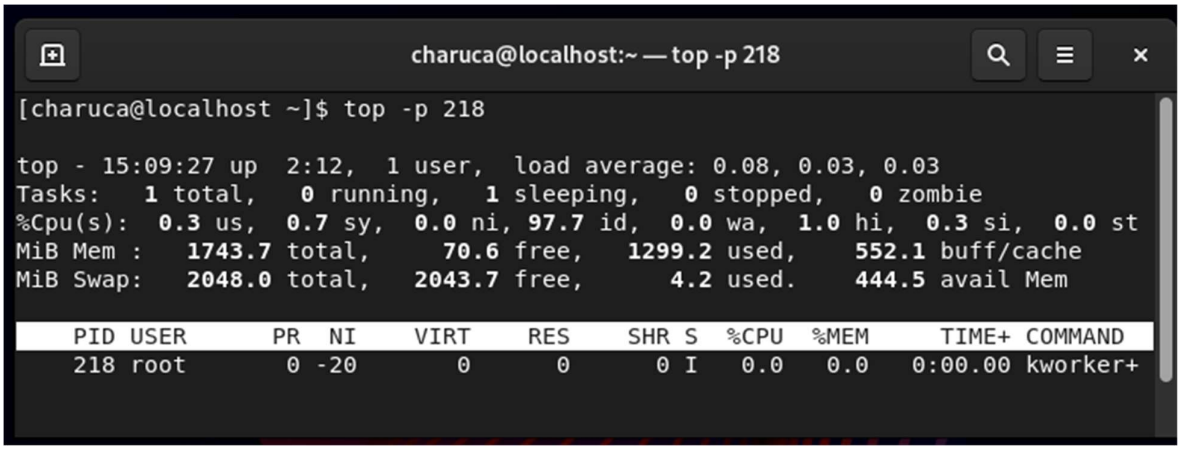
```
ps -p 218 -o pid,ppid,%cpu,%mem,cmd
```



- Display the processes and usage of resources with process id.

Syntax : `top -p <pid>`

```
top -p 218
```



- Stop a process temporarily or suspend a process using pid.

  Syntax :

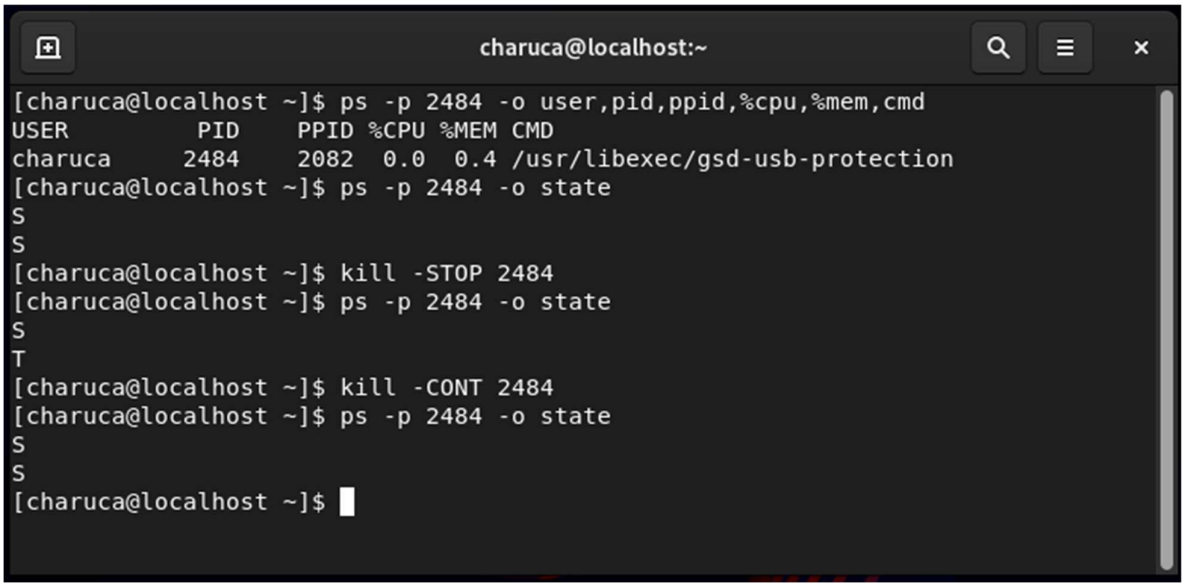
  To suspend the process

  `kill -STOP <pid>`

  To resume the process

  `kill -CONT <pid>`

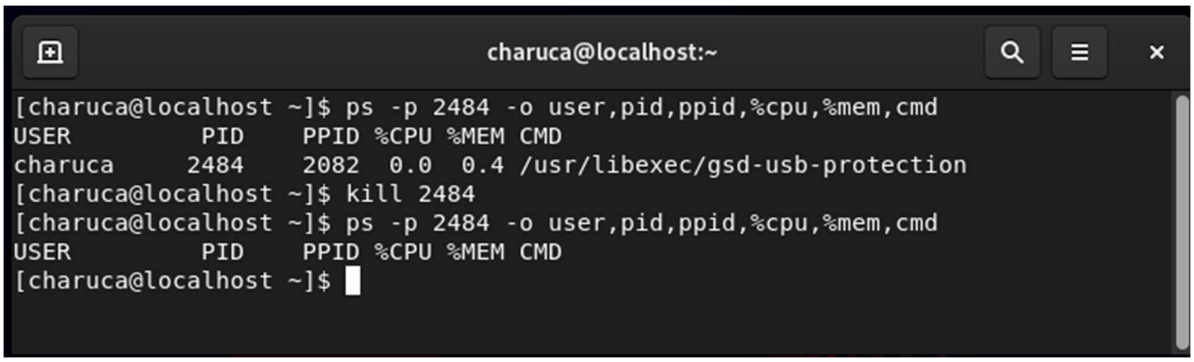
```
kill -STOP 2484
kill -CONT 2484
```



- Kill the process using pid.

Syntax : `kill <pid>`

```
kill 2484
```



Result:-

The processes are listed, displayed, suspended and killed using pid successfully.

Experiment - 7

Aim:-

Install, remove, list and update the packages in linux using rpm and yum.

Theory:-

Linux using RPM (Red Hat Package Manager) is a popular way to install and manage software packages on Linux operating systems, particularly those derived from or based on Red Hat Enterprise Linux (RHEL).

Linux using YUM (Yellowdog Updater, Modified) is a command-line package management utility used on Red Hat-based Linux operating systems, including CentOS and Fedora. YUM is designed to manage RPM packages, making it easy to install, update, and remove software packages.

Source Code:-

Install, remove, list and update packages in Linux using rpm.

- **Install a package.**

  Syntax:-
  rpm -ivh package.rpm
  rpm -i package.rpm

  Example:-
  rpm -ivh htop.rpm
  rpm -i htop.rpm

- **Remove a package.**

  Syntax:-
  rpm -e package
  rpm -evh package

  Example:-
  rpm -e htop
  rpm -evh htop

- **List packages.**

  Syntax: rpm -qa

- **Update a package.**

  Syntax: rpm -U package_name.rpm
  Example: rpm -U htop.rpm

Install, remove, list and update packages in Linux using yum.

- **Install a package.**

  Syntax: yum install package_name
  Example: yum install htop

- **Remove a package.**

  Syntax: yum remove package_name
  Example: yum remove htop

- **List packages.**

  Syntax: yum list

- **Update a package.**

  Syntax: yum update package_name
  Example: yum update htop

Result:-

The packages are installed, removed, listed and updated in Linux using rpm and yum.

Experiment - 8

Aim:-

Create a local yum repository to install the package own cloud from a public network and verify the yum repository

Theory and Source Code:-

To create a local yum repository to install the package own cloud from a public network and verify the yum repository, you can follow these steps:-

1. Download the ownCloud package and its dependencies from a public network, such as the official ownCloud website or a mirror site.

2. Install the "createrepo" package, which is used to create the local repository. You can install it using the following command:

   ```
   sudo yum install createrepo
   ```

3. Create a directory to hold the repository, such as "/var/www/html/owncloud-repo". You can create it using the following comand:

   ```
   sudo mkdir -p /var/www/html/owncloud-repo
   ```

4. Copy the ownCloud package and its dependencies to the repository directory.

5. Run the "createrepo" command in the repository directory to create the repository metadata:

   ```
   sudo createrepo /var/www/html/owncloud-repo/
   ```

6. Create a new file called "owncloud.repo" in the "/etc/yum.repos.d/" directory with the following contents:

```
[owncloud-repo] name=ownCloud Local Repo
baseurl=file:///var/www/html/owncloud-repo/ enabled=1
gpgcheck=0
```

7. Save the file and exit.

8. Run the following command to verify that the repository is configured correctly:

```
sudo yum --disablerepo=* --enablerepo=owncloud-repo list
available
```

This command should display a list of available packages in the local ownCloud repository. You can then install ownCloud and its dependencies using the following command:

```
sudo yum install owncloud
```

Note: Make sure to update the local repository regularly to ensure that you have the latest versions of ownCloud and its dependencies.

Result:-

A local yum repository is created to install the package own cloud from a public network and verified successfully.

Experiment – 9

Aim:-

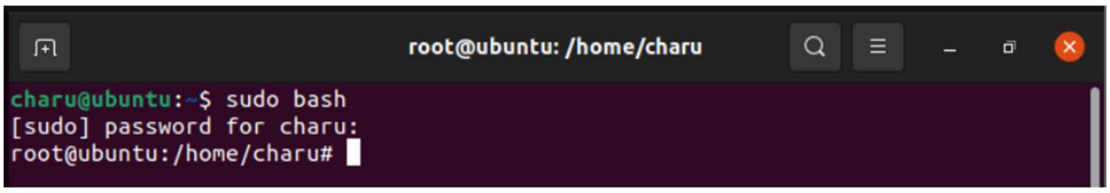
Create a raw partitions sdb1, sdb2, sdc1, sdc2 using fdisk utility

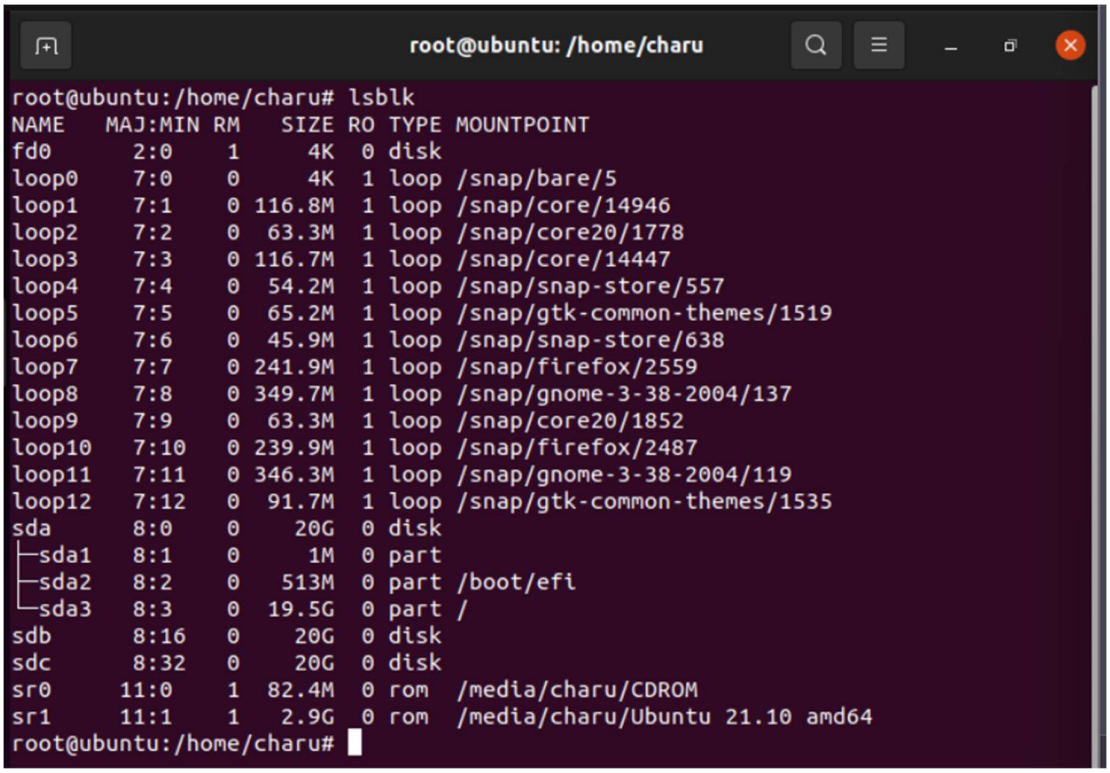
a. Create a partition table
b. Create a swap space of 10G in sdc2
c. Assign the sdb1 as bootable flag
d. Create various file systems such as ext2, ext3, ext4 in those raw partitions

Source Code and Output:-

To go to the root user,

```
sudo bash
```

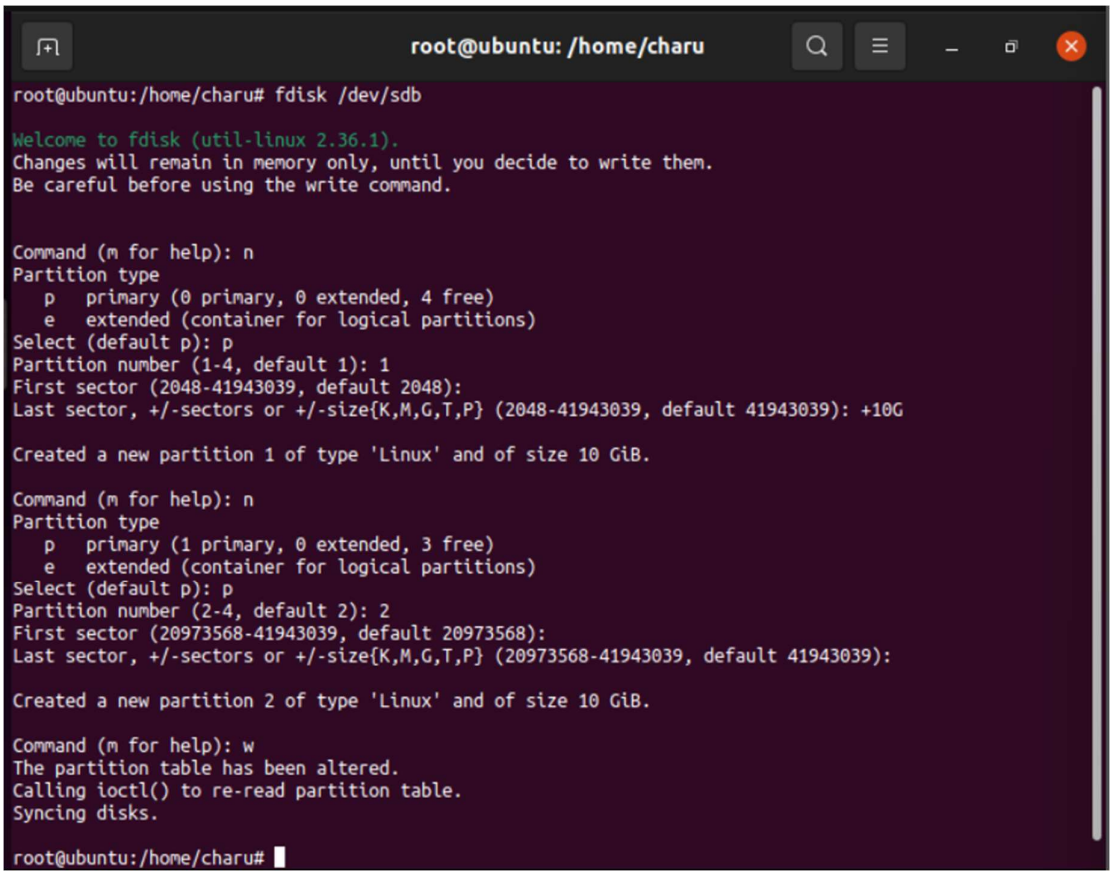


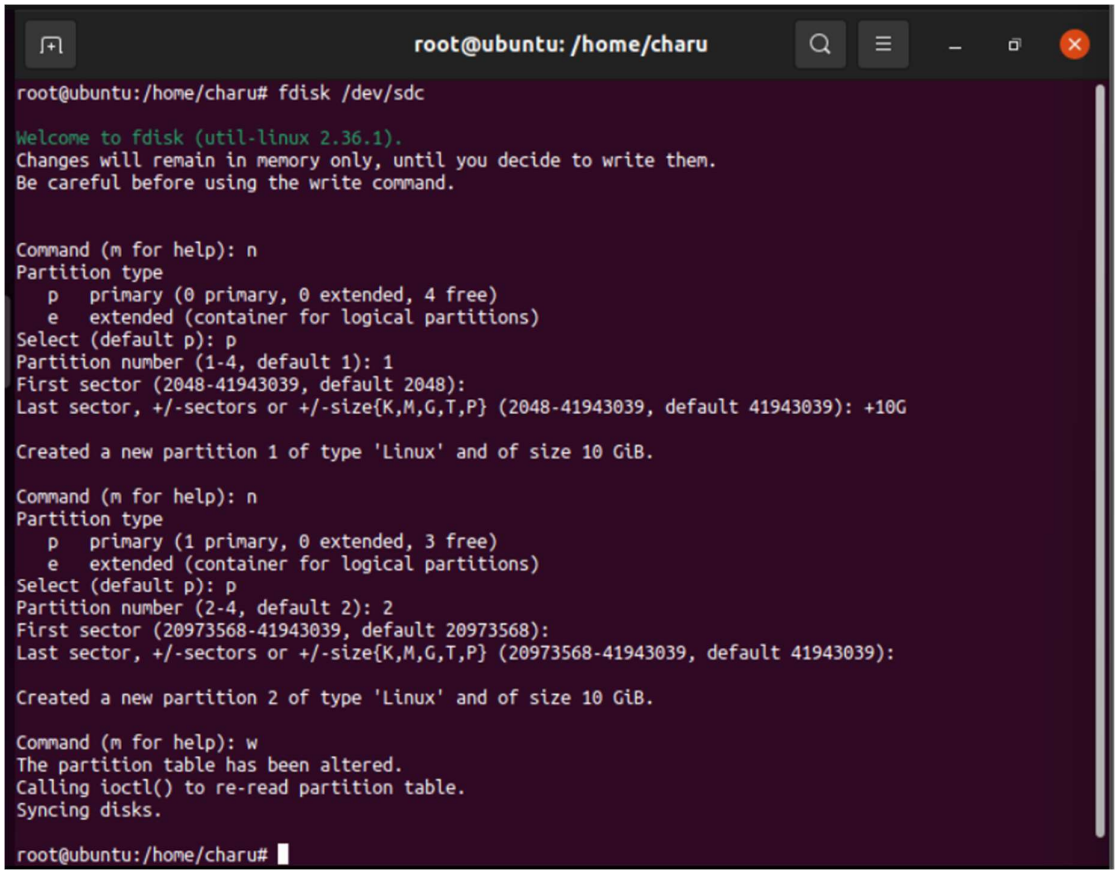


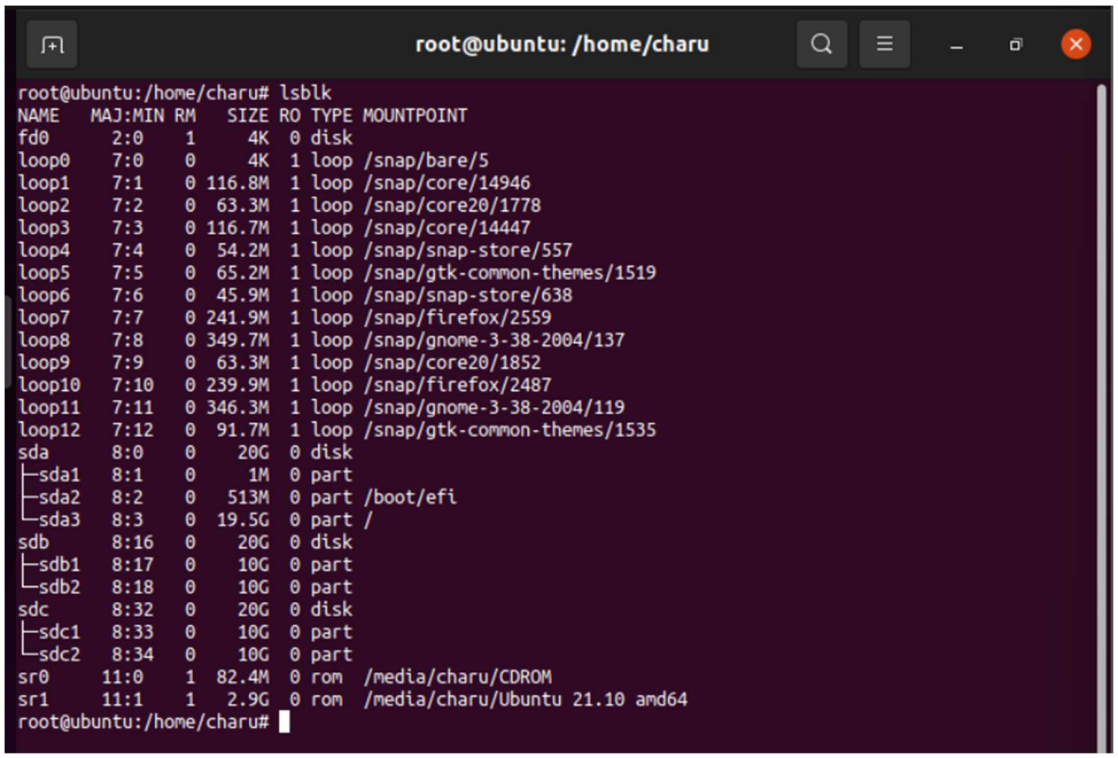
a. Create a partition table:-

```
#fdisk /dev/sdb
Command (m for help): o

Command (m for help): n
Partition type
   p   primary (1 primary, 0 extended, 3 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 2): 1
First sector (20973568-41943039, default 20973568):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (20973568-
41943039, default 41943039): +10G

Command (m for help): n
Partition type
   p   primary (1 primary, 0 extended, 3 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (2-4, default 2): 2
First sector (20973568-41943039, default 20973568):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (20973568-
41943039, default 41943039)


#fdisk /dev/sdc
Command (m for help): o

Command (m for help): n
Partition type
   p   primary (1 primary, 0 extended, 3 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 2): 1
First sector (20973568-41943039, default 20973568):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (20973568-
41943039, default 41943039): +10G
```

```
Command (m for help): n
Partition type
   p   primary (1 primary, 0 extended, 3 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (2-4, default 2): 2
First sector (20973568-41943039, default 20973568):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (20973568-
41943039, default 41943039):
```

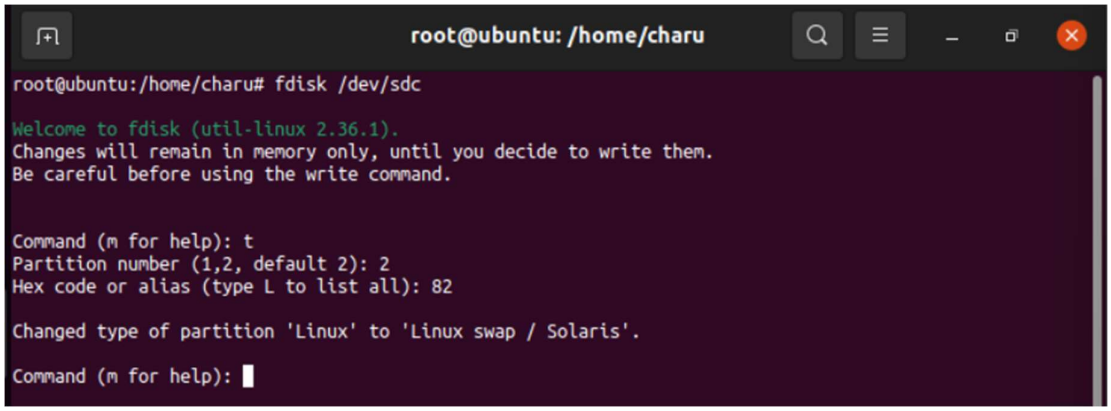



root@ubuntu: /home/charu
root@ubuntu:/home/charu# fdisk /dev/sdc
Welcome to fdisk (util-linux 2.36.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
Command (m for help): n
Partition type
p primary (0 primary, 0 extended, 4 free)
e extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-41943039, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-41943039, default 41943039): +10G
Created a new partition 1 of type 'Linux' and of size 10 GiB.
Command (m for help): n
Partition type
p primary (1 primary, 0 extended, 3 free)
e extended (container for logical partitions)
Select (default p): p
Partition number (2-4, default 2): 2
First sector (20973568-41943039, default 20973568):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (20973568-41943039, default 41943039):
Created a new partition 2 of type 'Linux' and of size 10 GiB.
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
root@ubuntu:/home/charu#


root@ubuntu: /home/charu
root@ubuntu:/home/charu# lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
fd0 2:0 1 4K 0 disk
loop0 7:0 0 4K 1 loop /snap/bare/5
loop1 7:1 0 116.8M 1 loop /snap/core/14946
loop2 7:2 0 63.3M 1 loop /snap/core20/1778
loop3 7:3 0 116.7M 1 loop /snap/core/14447
loop4 7:4 0 54.2M 1 loop /snap/snap-store/557
loop5 7:5 0 65.2M 1 loop /snap/gtk-common-themes/1519
loop6 7:6 0 45.9M 1 loop /snap/snap-store/638
loop7 7:7 0 241.9M 1 loop /snap/firefox/2559
loop8 7:8 0 349.7M 1 loop /snap/gnome-3-38-2004/137
loop9 7:9 0 63.3M 1 loop /snap/core20/1852
loop10 7:10 0 239.9M 1 loop /snap/firefox/2487
loop11 7:11 0 346.3M 1 loop /snap/gnome-3-38-2004/119
loop12 7:12 0 91.7M 1 loop /snap/gtk-common-themes/1535
sda 8:0 0 20G 0 disk
├─sda1 8:1 0 1M 0 part
├─sda2 8:2 0 513M 0 part /boot/efi
└─sda3 8:3 0 19.5G 0 part /
sdb 8:16 0 20G 0 disk
├─sdb1 8:17 0 10G 0 part
└─sdb2 8:18 0 10G 0 part
sdc 8:32 0 20G 0 disk
├─sdc1 8:33 0 10G 0 part
└─sdc2 8:34 0 10G 0 part
sr0 11:0 1 82.4M 0 rom /media/charu/CDROM
sr1 11:1 1 2.9G 0 rom /media/charu/Ubuntu 21.10 amd64
root@ubuntu:/home/charu#

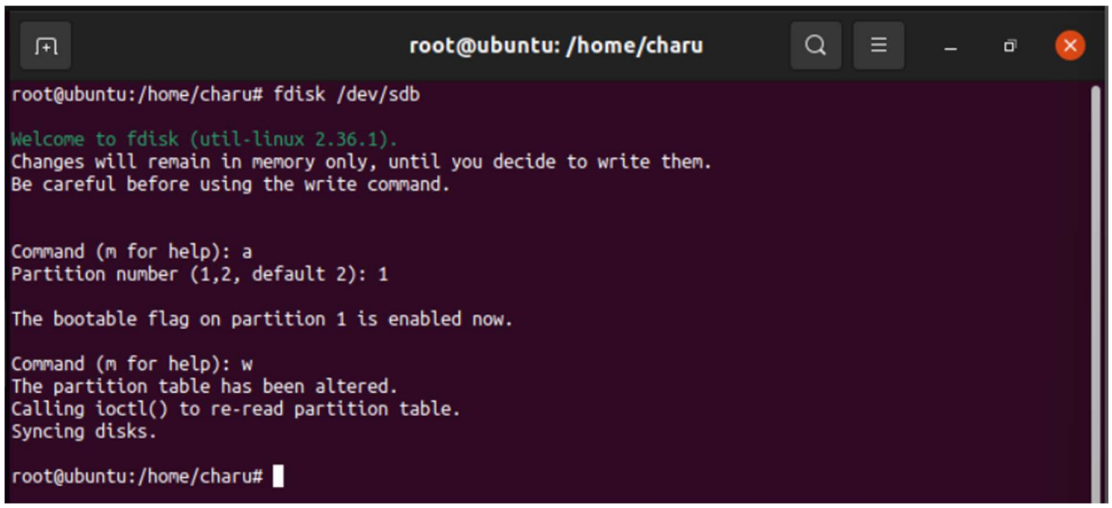
b. Create a swap space of 10G in sdc2

```
#fdisk /dev/sdb
Command (m for help): t
Partition number (1,2, default 2): 2
Hex code or alias (type L to list all): 82
```



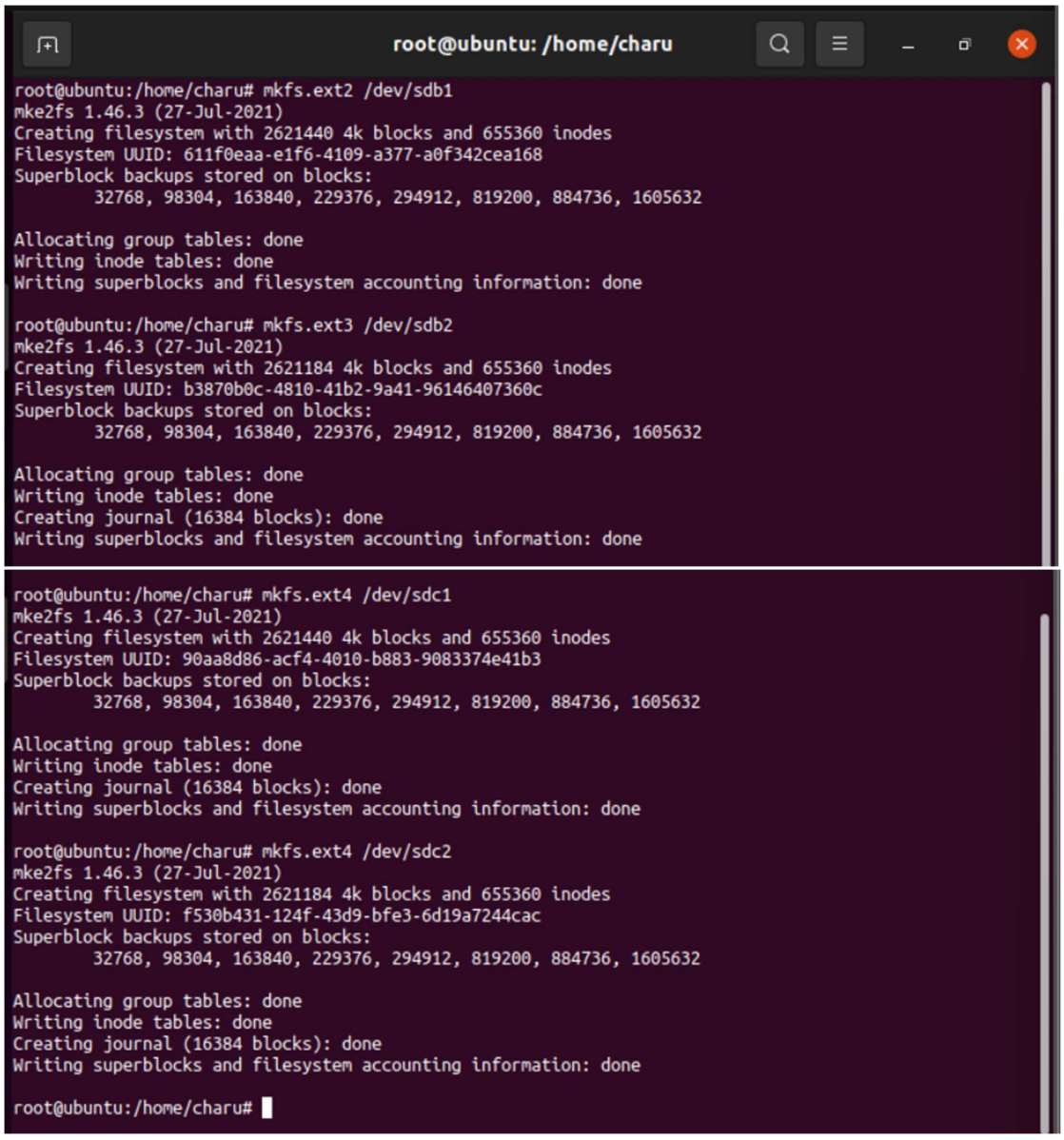
c. Assign the sdb1 as bootable flag

```
# fdisk /dev/sdb
Command (m for help): a
Partition number (1,2, default 2): 1
```



d. Create various file systems such as ext2, ext3, ext4 in those raw partitions

```
#mkfs.ext2 /dev/sdb1
#mkfs.ext3 /dev/sdb2
#mkfs.ext4 /dev/sdc1
#mkfs.ext4 /dev/sdc2
```



Result:-

The experiment is verified.

Experiment – 10

Aim:-

Configure LVM by creating the physical volumes PV1 and PV2 from sdb2 and sdc1 and Logical volume group LVG and logical volumes LV1, LV2 and LV3

Source Code and Output:-

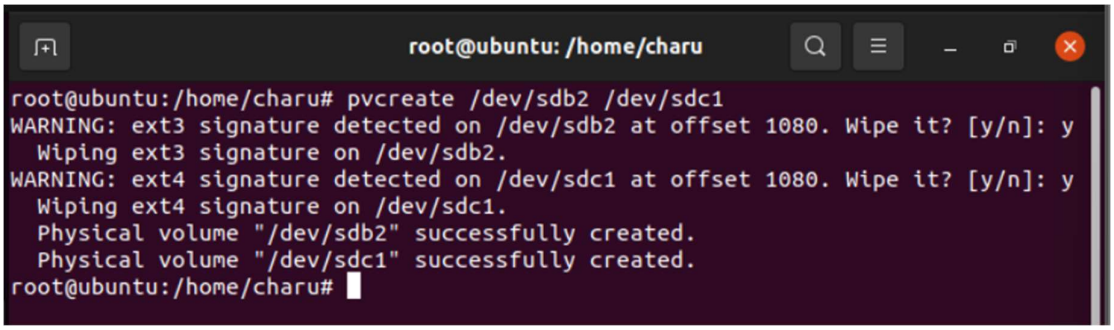
The steps to configure LVM by creating the physical volumes PV1 and PV2 from sdb2 and sdc1, and Logical volume group LVG and logical volumes LV1, LV2, and LV3:

1. Make sure that sdb2 and sdc1 are not mounted or being used by any system process. If they are, unmount them using the following command:

```
umount /dev/sdb2
umount /dev/sdc1
```

2. Initialize the physical volumes using the following command:

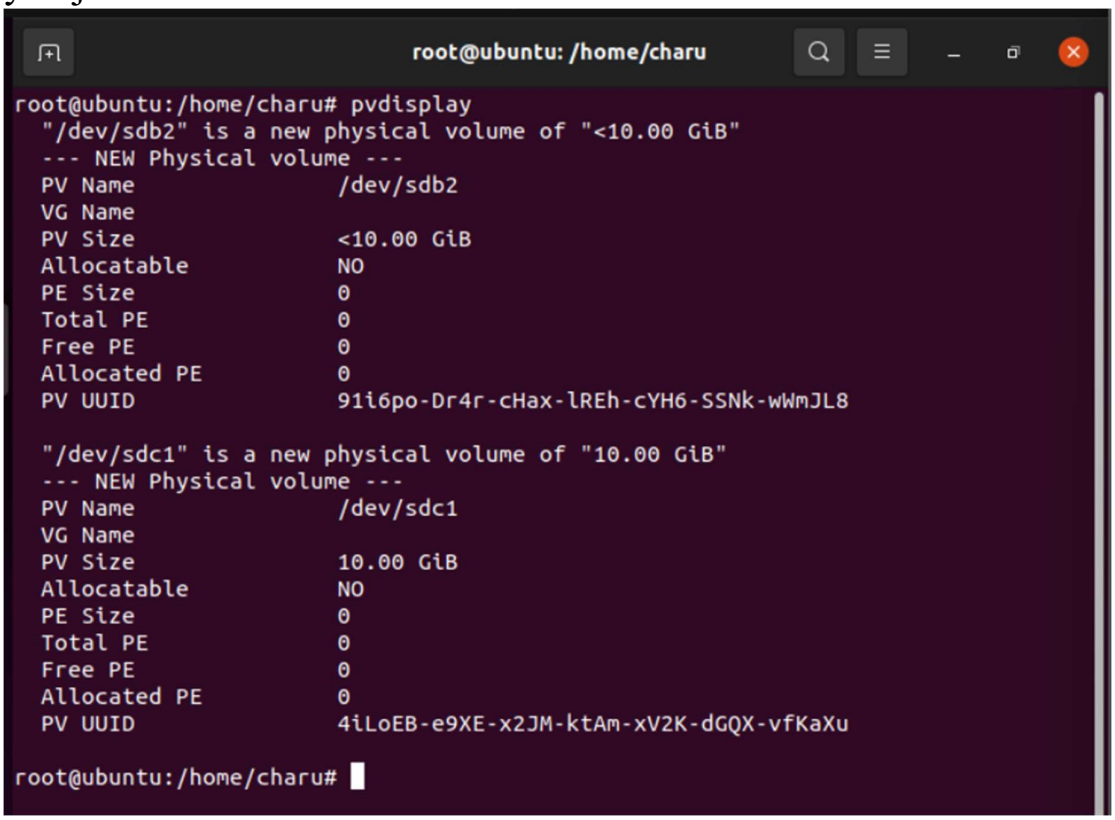
```
pvcreate /dev/sdb2 /dev/sdc1
```



3. Verify the physical volumes using the following command:

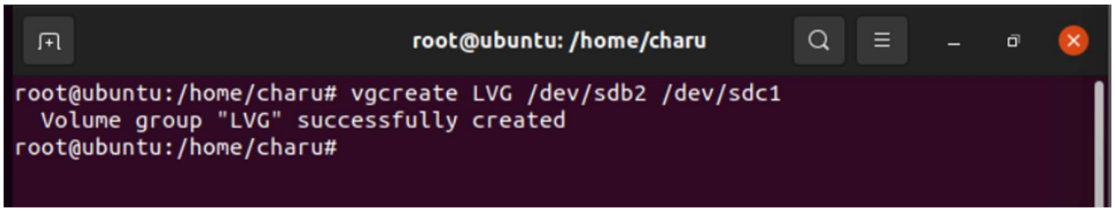
   `pvdisplay`

   This should show you the details of the physical volumes that you just created.



4. Create the logical volume group using the following command:

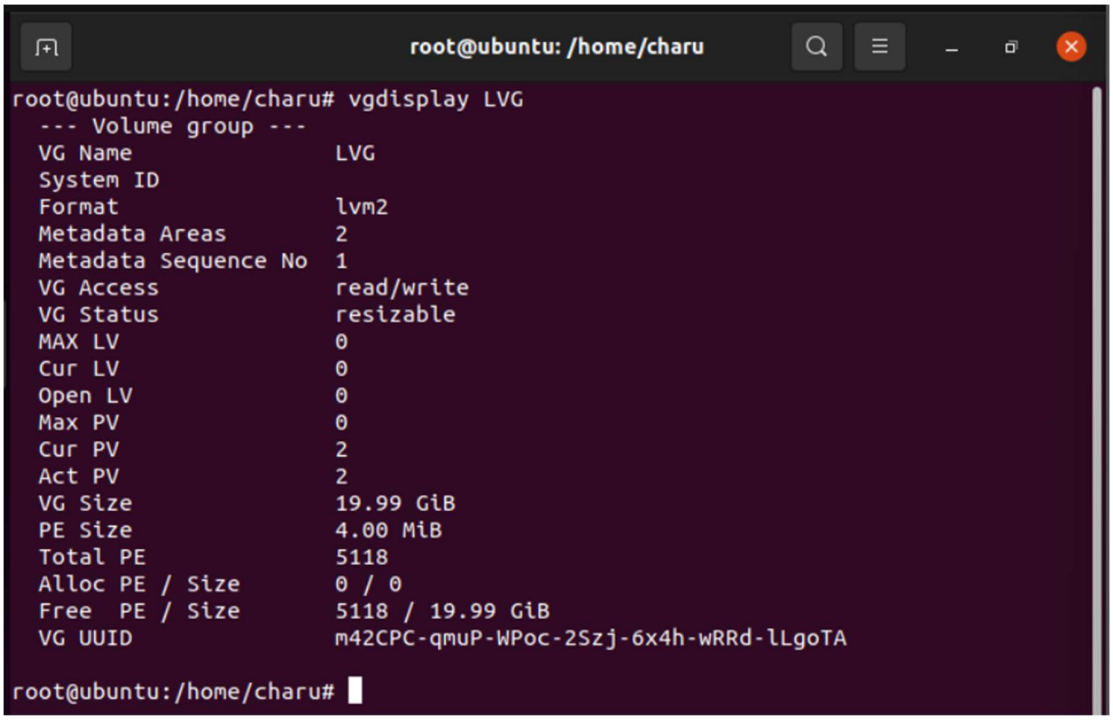
   `vgcreate LVG /dev/sdb2 /dev/sdc1`



5. Verify the logical volume group using the following command:

   `vgdisplay LVG`

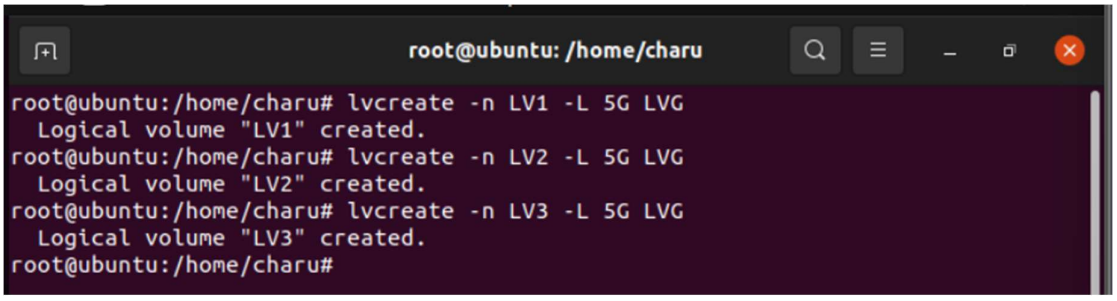
   This should show you the details of the logical volume group that you just created.



6. Create the logical volumes using the following commands:

   ```
   lvcreate -n LV1 -L 5G LVG
   lvcreate -n LV2 -L 5G LVG
   lvcreate -n LV3 -L 5G LVG
   ```

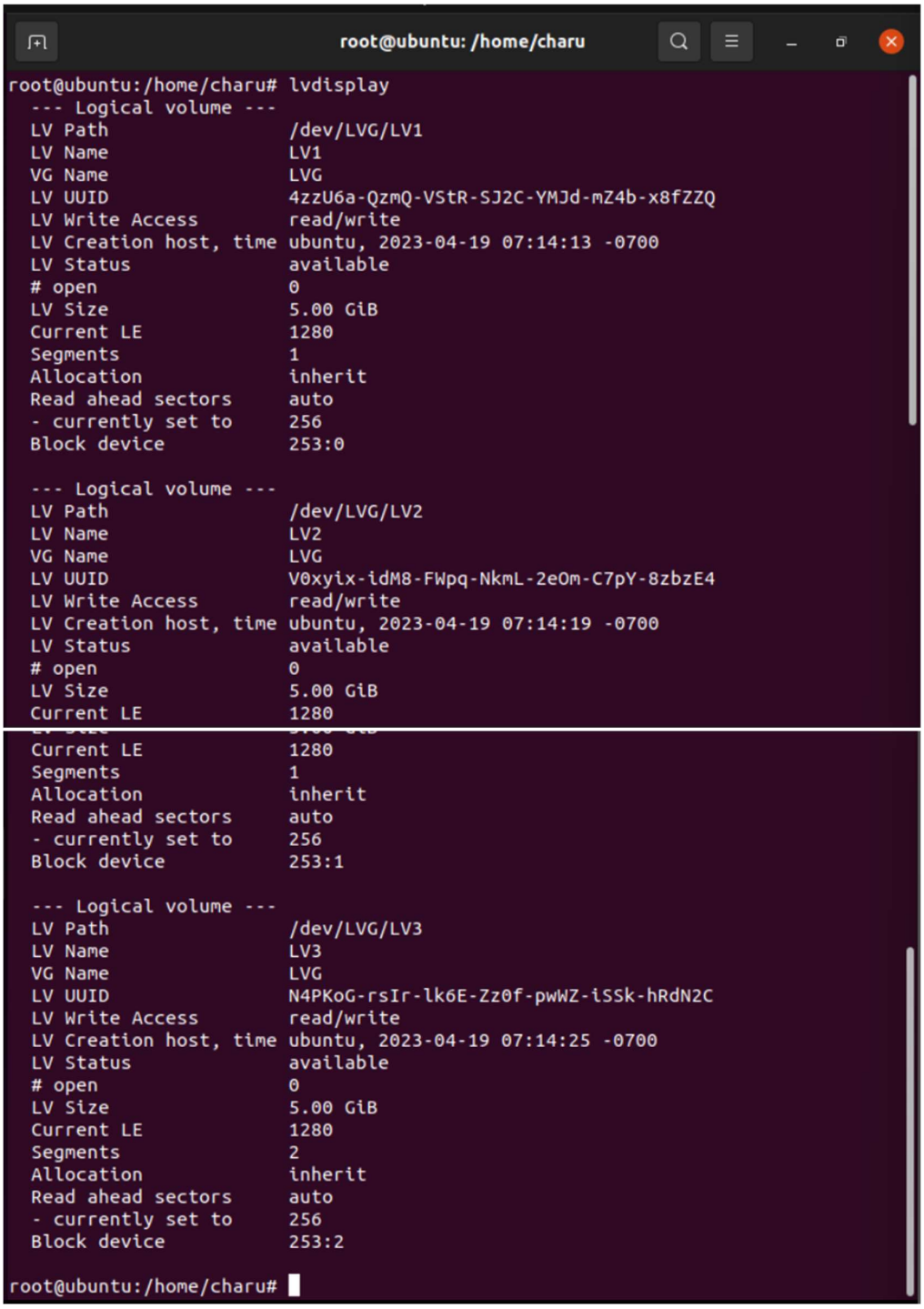
   This will create three logical volumes named LV1, LV2, and LV3 with sizes of 5GB, 5GB, and 5GB, respectively.



7. Verify the logical volumes using the following command:

   `lvdisplay`

   This should show you the details of the logical volumes that you just created.



Result:-

Configured LVM by creating physical volumes PV1 and PV2 from sdb2 and sdc1 and logical volume group LVG and logical volumes LV1, LV2, and LV3 successfully.

Experiment – 11

Aim:-

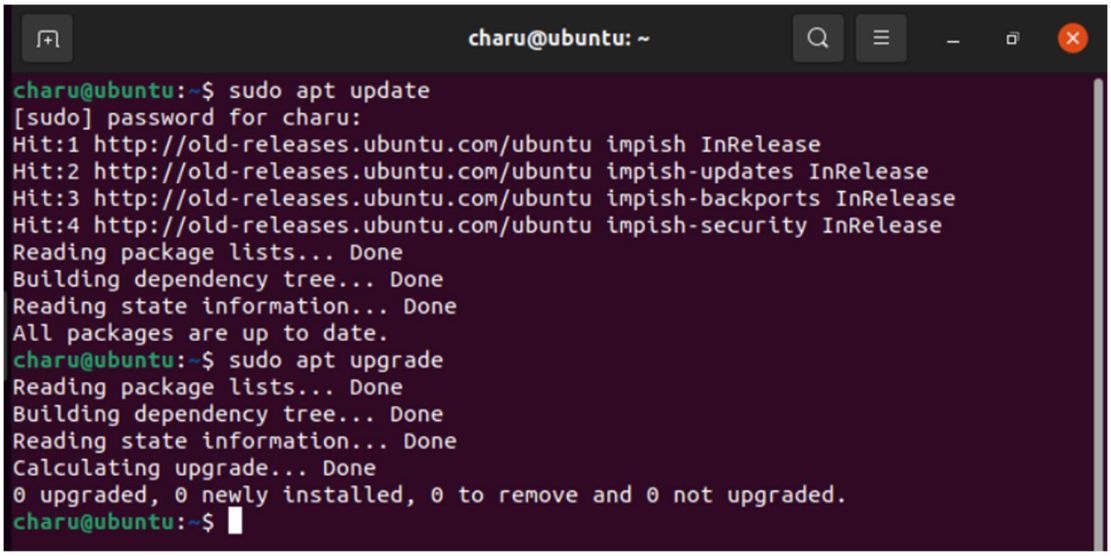
Configure apache webserver using httpd with port number 8081 and install open ssl certificate to deploy the webserver in https.

Source Code and Output:-

The steps to configure the Apache web server using `httpd` with port number 8081 and install an SSL certificate to enable HTTPS deployment

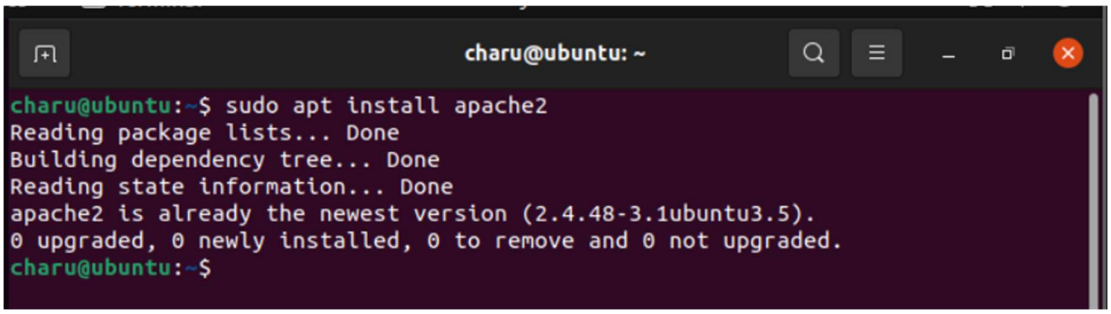
Step 1: Update System Packages

```
sudo apt update
sudo apt upgrade
```



Step 2: Install Apache HTTP Server (`httpd`)

```
sudo apt install apache2
```



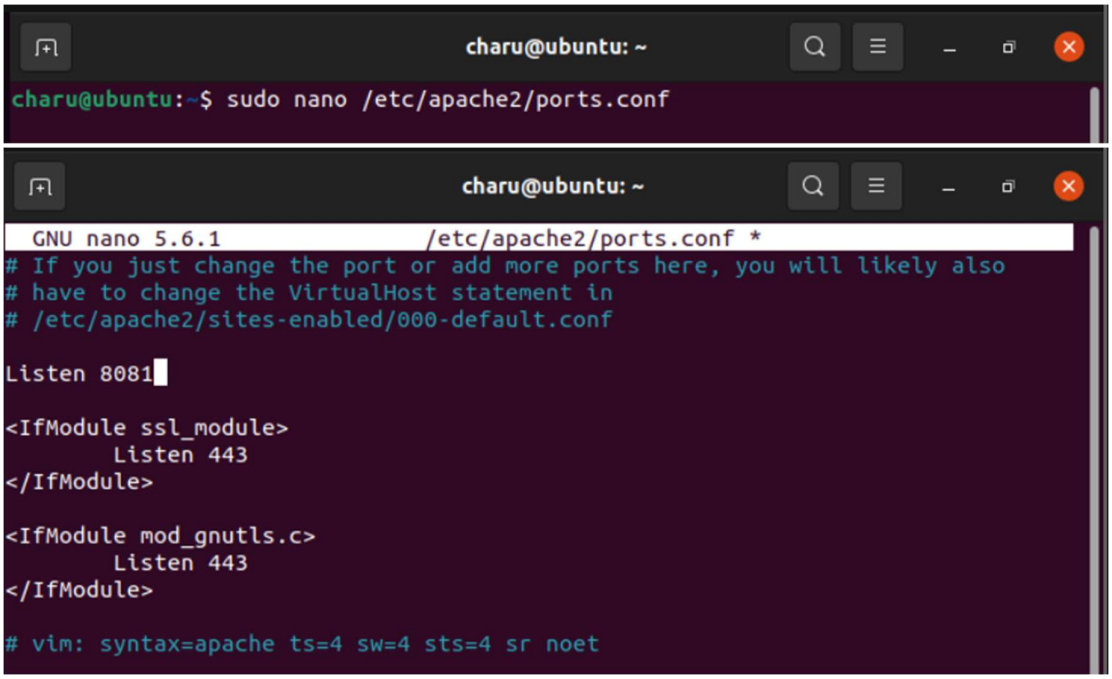
Step 3: Configure Apache to Use Port 8081

```
sudo nano /etc/apache2/ports.conf
```

Inside the `ports.conf` file, change the `Listen` directive to:

```
Listen 8081
```

Save the file and exit.



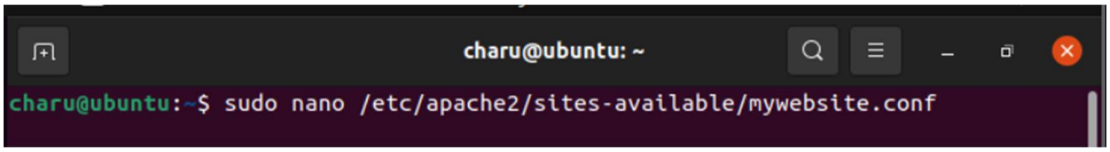
Step 4: Create a New Virtual Host Configuration

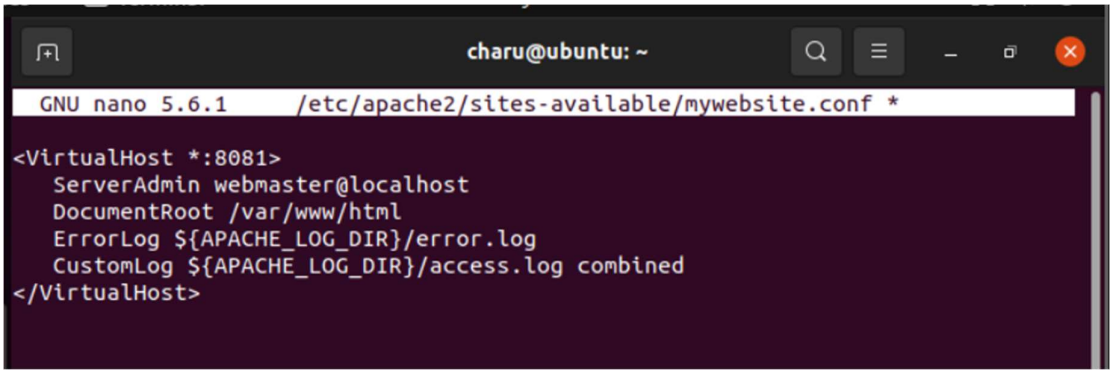
```
sudo nano /etc/apache2/sites-
available/mywebsite.conf
```

Add the following content to the file:

```
<VirtualHost *:8081>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log
combined
</VirtualHost>
```

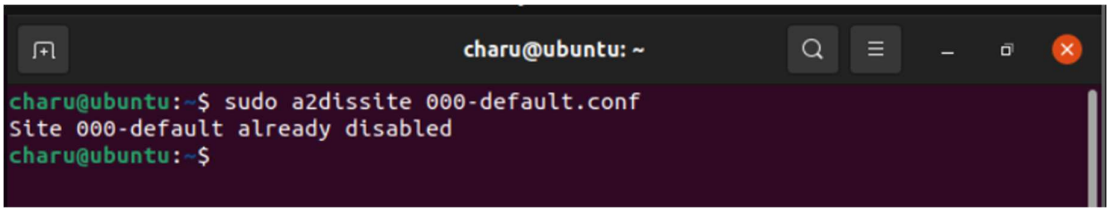
Save the file and exit.





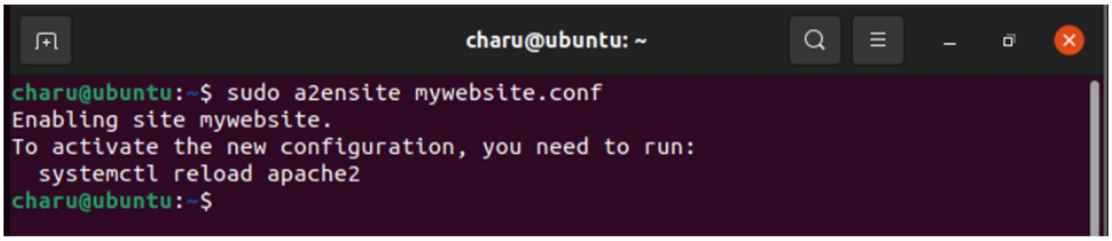
Step 5: Disable the Default Virtual Host Configuration

```
sudo a2dissite 000-default.conf
```



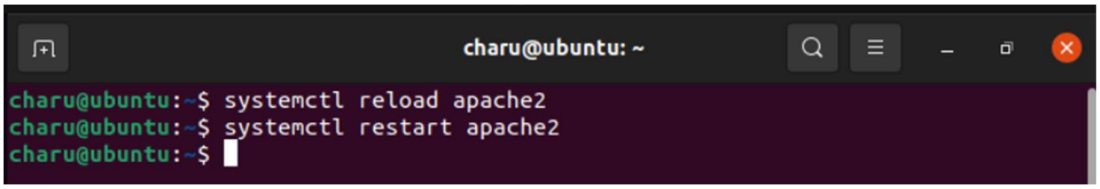
Step 6: Enable the New Virtual Host Configuration

```
sudo a2ensite mywebsite.conf
```



Step 7: Restart Apache Service

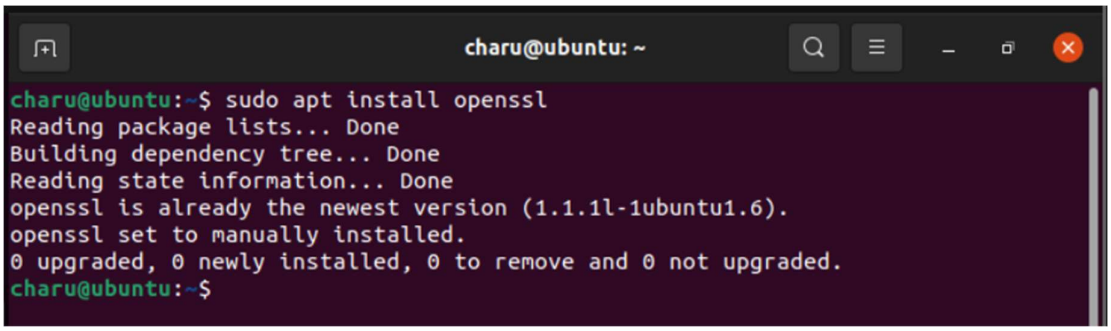
```
sudo service apache2 restart
```



At this point, Apache should be configured to use port 8081.

Step 8: Install OpenSSL

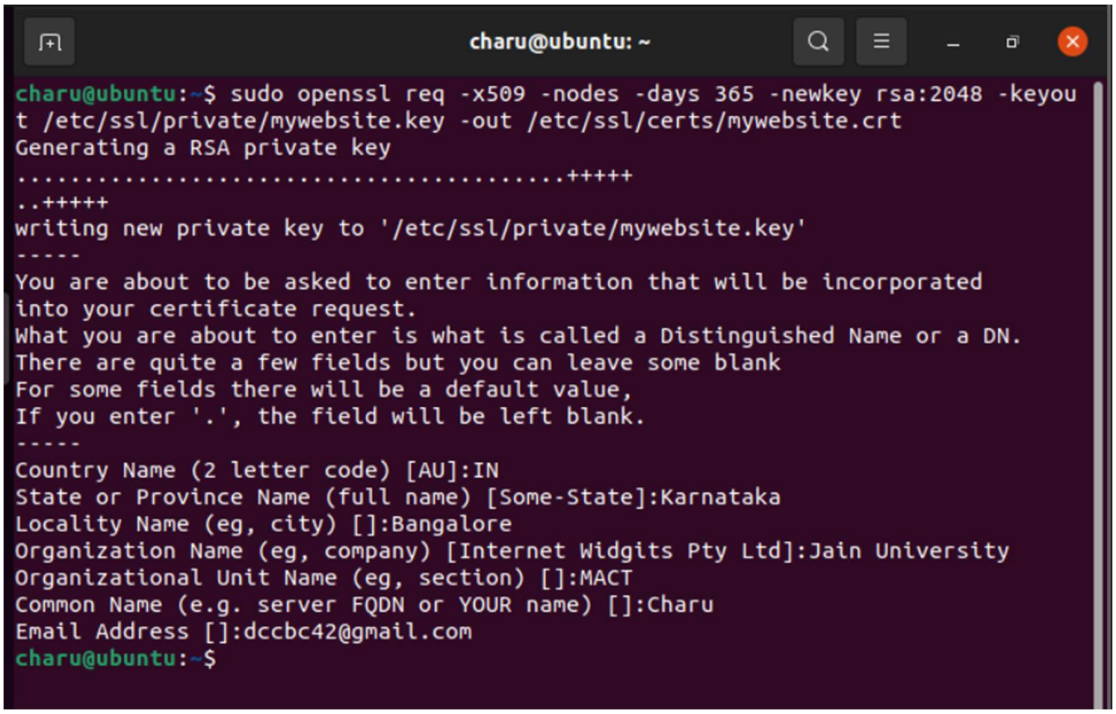
```
sudo apt install openssl
```



Step 9: Generate a Self-Signed SSL Certificate

```
sudo openssl req -x509 -nodes -days 365 -newkey
rsa:2048 -keyout /etc/ssl/private/mywebsite.key -
out /etc/ssl/certs/mywebsite.crt
```

Follow the prompts to enter the necessary details for your SSL certificate.



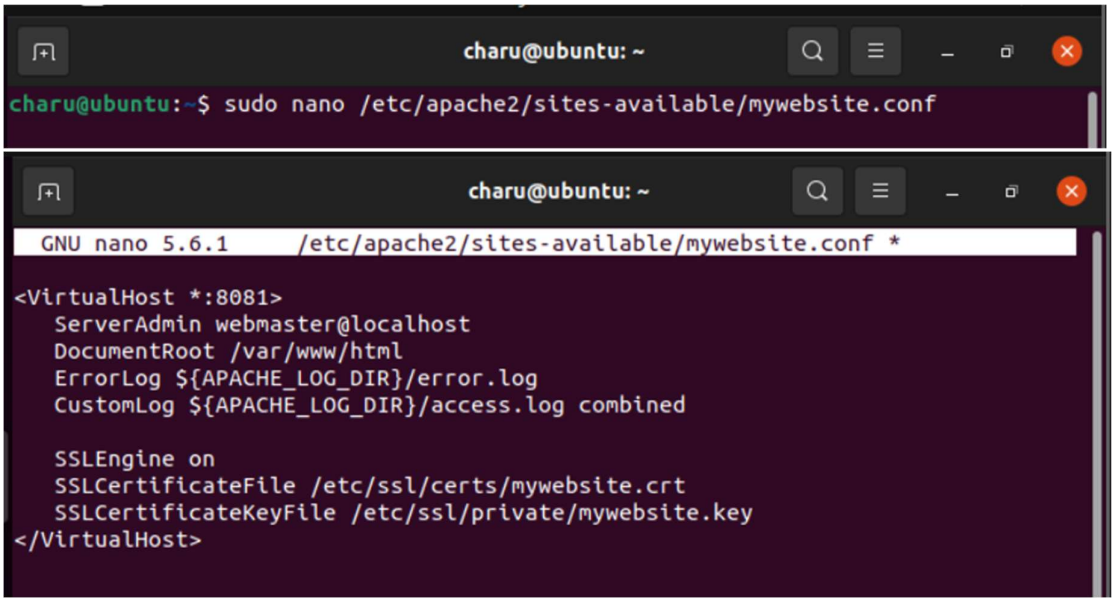
Step 10: Configure Apache to Use SSL

```
sudo nano /etc/apache2/sites-available/mywebsite.conf
```

Add the following lines inside the `<VirtualHost>` block:

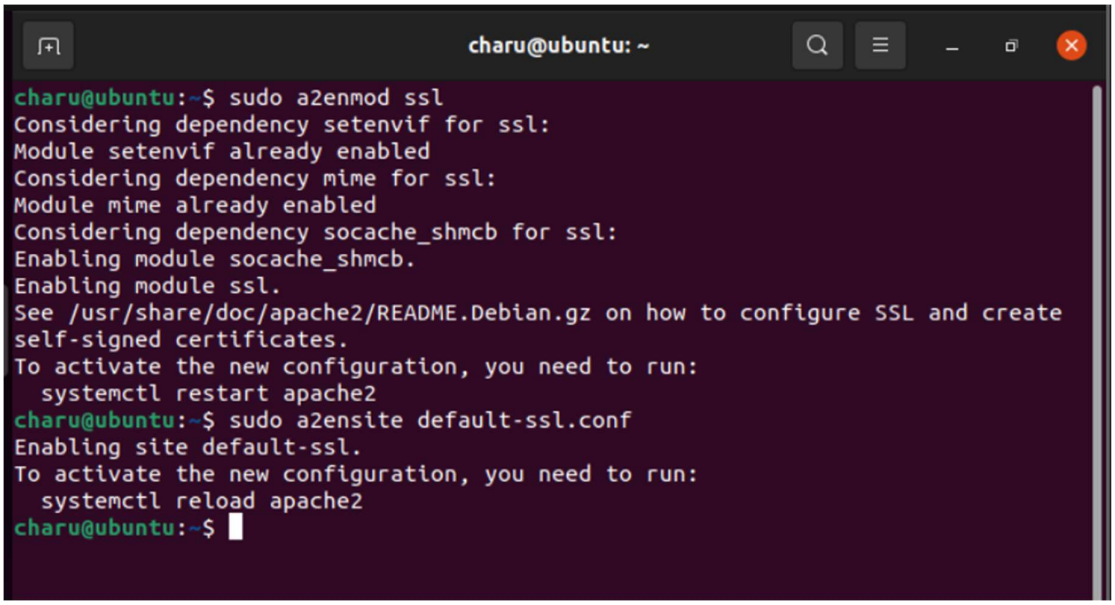
```
SSLEngine on
SSLCertificateFile /etc/ssl/certs/mywebsite.crt
SSLCertificateKeyFile /etc/ssl/private/mywebsite.key
```

Save the file and exit.



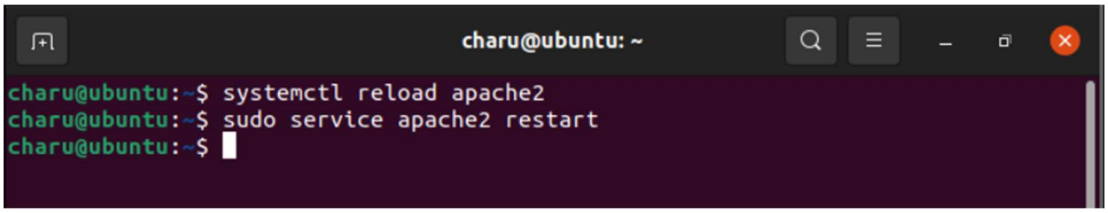
Step 11: Enable SSL Module and Default SSL Site

```
sudo a2enmod ssl
sudo a2ensite default-ssl.conf
```



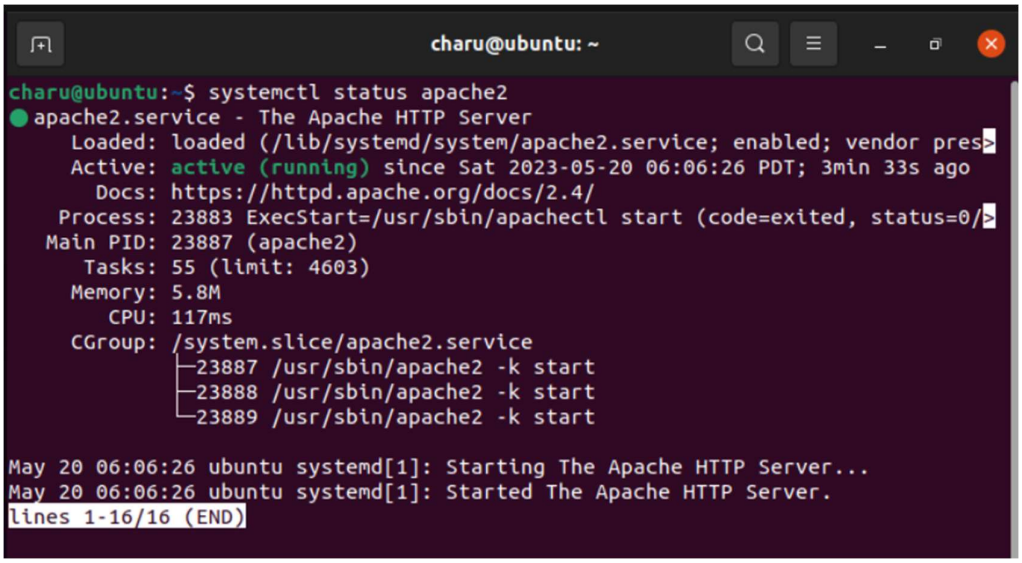
Step 12: Restart Apache Service

```
sudo service apache2 restart
```



Step 13: Check the status of Apache status

```
systemctl status apache2
```

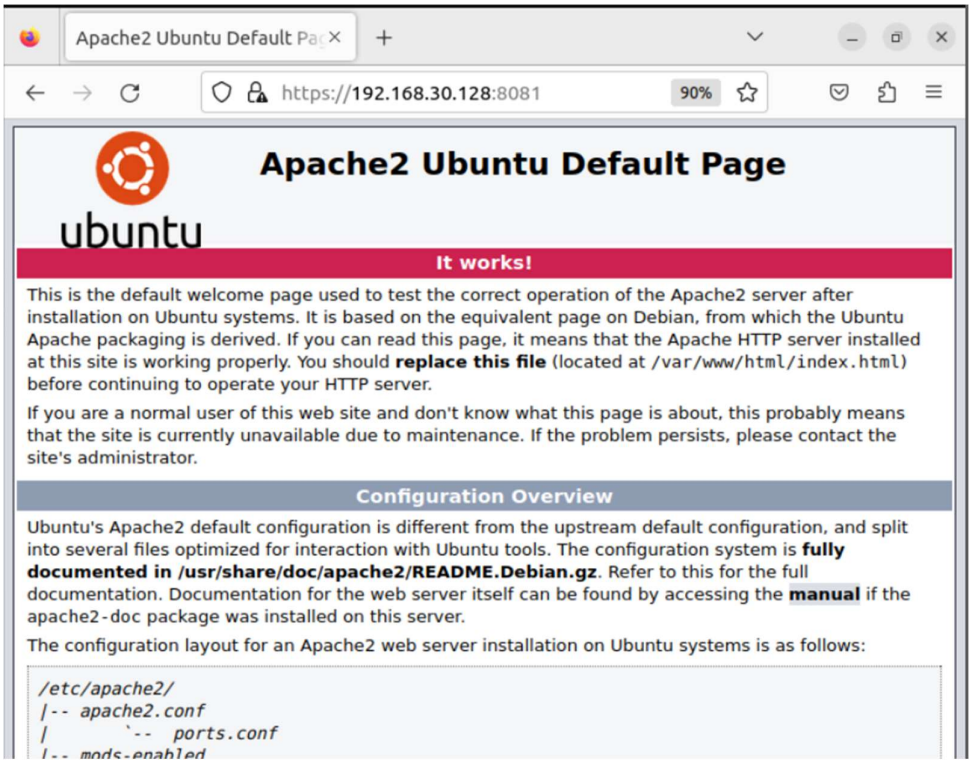


Apache is now configured with port 8081 and an SSL certificate to enable HTTPS deployment.

Step 14: Website can be accessed using,

https://<your-server-ip>:8081

(hostname -I for ip)



Experiment – 12

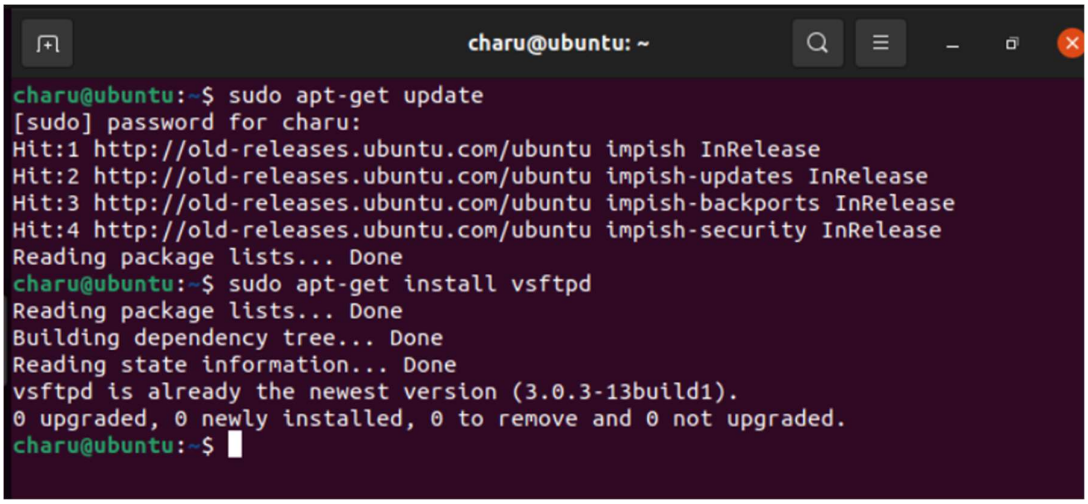
Aim:-

Configure the FTP server and enable the FTP to be accessible from webserver.

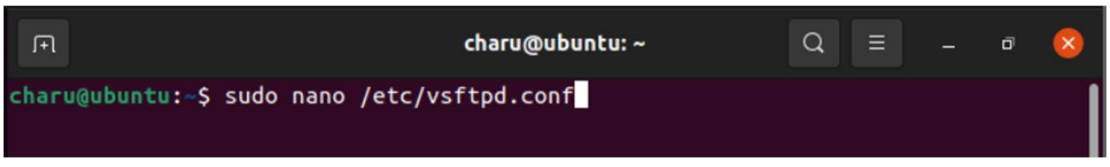
Source Code and Output:-

To configure an FTP server and enable access from a web server in Ubuntu, you can follow these steps:

1. Install the vsftpd package by running the following command:
   ```
   sudo apt-get update
   sudo apt-get install vsftpd
   ```



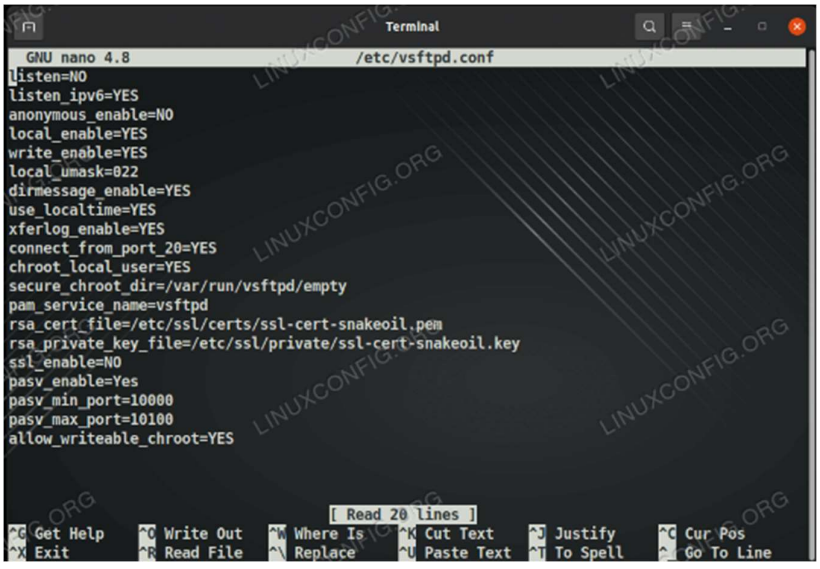
2. Once the installation is complete, open the configuration file for vsftpd using a text editor. For example:
   ```
   sudo nano /etc/vsftpd.conf
   ```



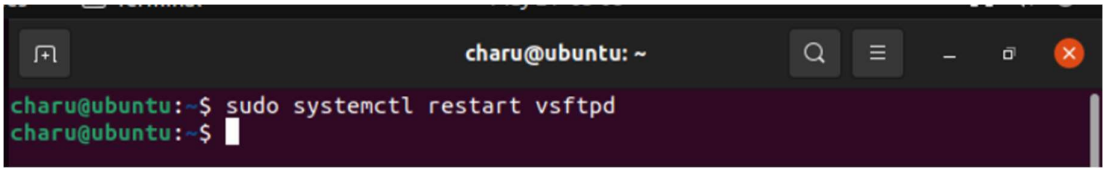
3. Make the following changes in the configuration file:

```
listen=NO
listen_ipv6=YES
anonymous_enable=NO
local_enable=YES
write_enable=YES
local_umask=022
dirmessage_enable=YES
use_localtime=YES
xferlog_enable=YES
connect_from_port_20=YES
chroot_local_user=YES
secure_chroot_dir=/var/run/vsftpd/empty
pam_service_name=vsftpd
rsa_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
rsa_private_key_file=/etc/ssl/private/ssl-cert-
snakeoil.key
ssl_enable=NO
pasv_enable=Yes
pasv_min_port=10000
pasv_max_port=10100
allow_writeable_chroot=YES
```

Save the changes and exit the text editor.

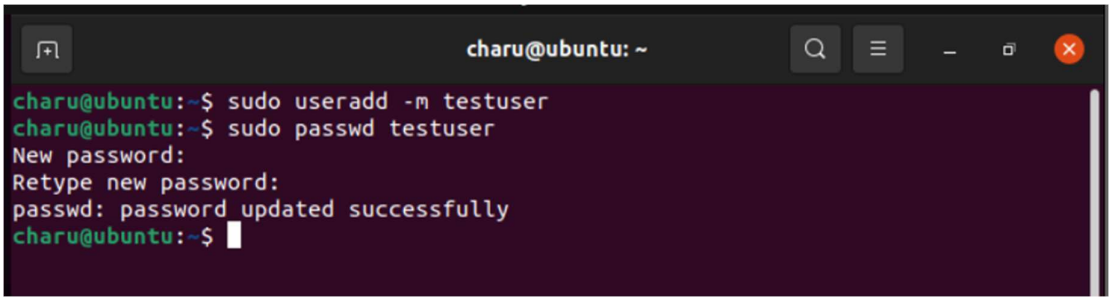


4. Restart the vsftpd service to apply the configuration changes:
   `sudo systemctl restart vsftpd`



5. Use this first command to create a new account called testuser, and the second command to set a password for the account:

   ```
   sudo useradd -m testuser
   sudo passwd testuser
   ```

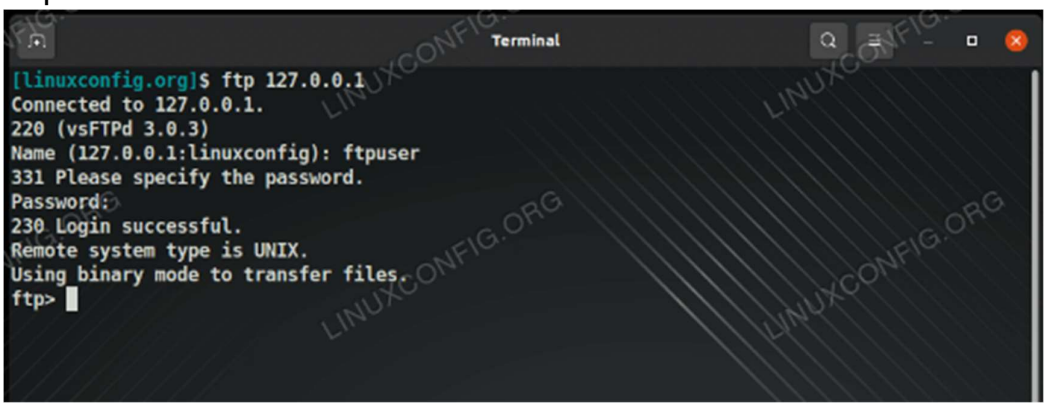


6. In order to verify that everything's working properly, you should store at least one file in ftpuser's home directory. This file should be visible when we login to FTP in the next steps.

   ```
   sudo bash -c "echo FTP TESTING > /home/ftpuser/FTP-TEST"
   ```

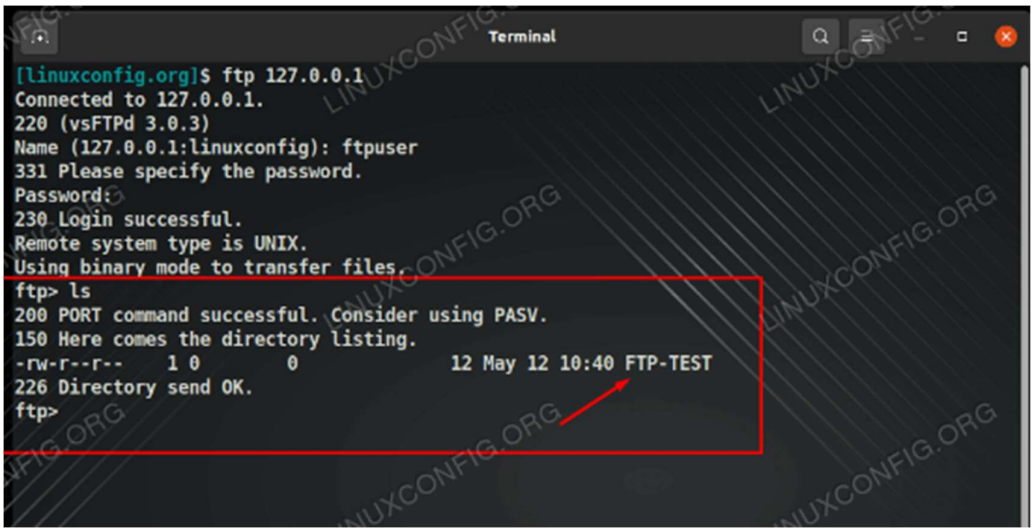
7. Ensure that the default FTP client utility is installed on your system by running the following command. It will either install the software or tell you that it already exists on the system.
   `sudo apt install ftp`

```
ftp 127.0.0.1
```



ls command, which will list the test file that we created in previous steps.

```
ftp> ls
```



8. By default, the FTP protocol listens on port 21 for user authentication and port 20 for data transfer. However, we can change this behavior by making a small edit to the /etc/vsftpd.conf file. At the bottom of the file, use the listen_port directive to specify a different port for vsftpd to use.

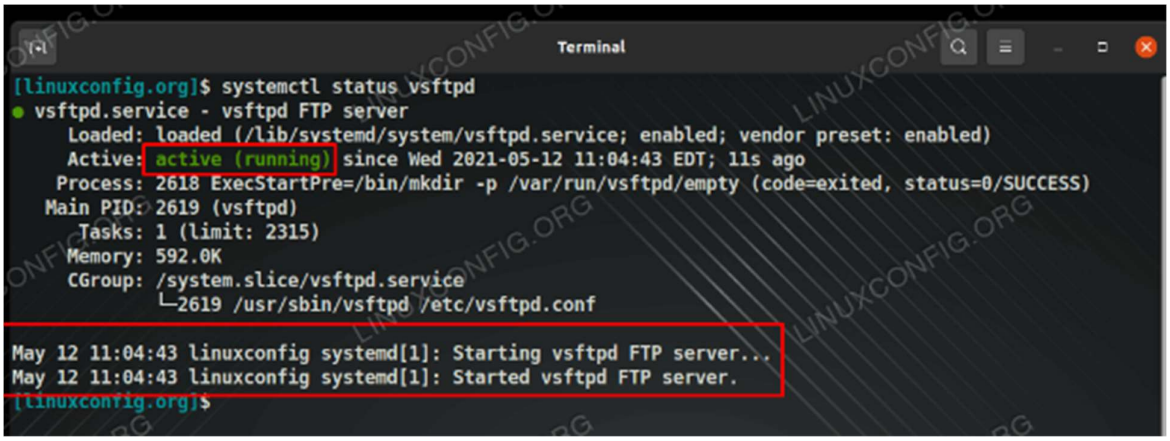
```
listen_port=2121
```

9. Ubuntu's built-in firewall will block FTP traffic by default, but the following command will create an exception in UFW to allow the traffic. Make sure that the FTP port is not blocked by your firewall by executing this command:

```
sudo ufw allow from any to any port 20,21,10000:10100
proto tcp
```

10. Check the status of the vsftpd service to ensure that it's running and has not encountered any startup errors.

```
systemctl status vsftpd
```



Start or restart the server for configuring.

```
sudo systemctl start vsftpd
```

OR

```
sudo systemctl restart vsftpd
```

Experiment – 13

Aim:-

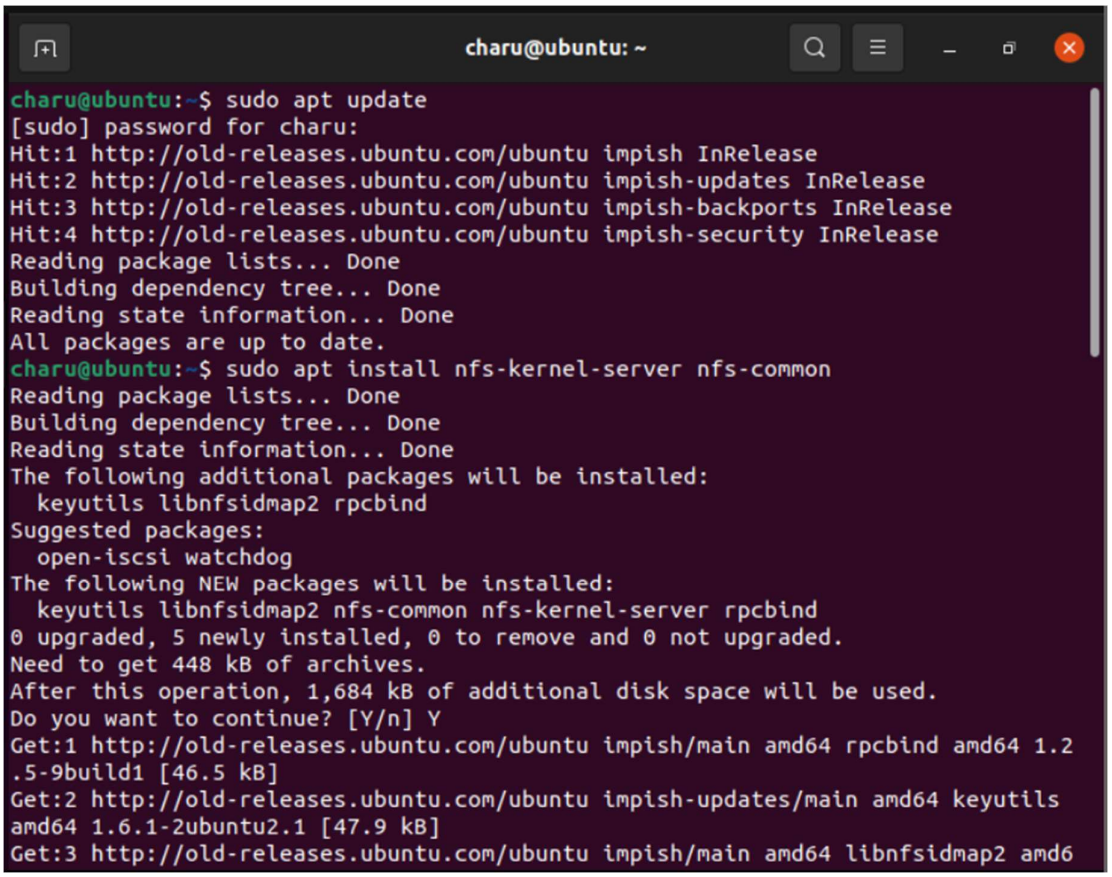
Configure NFS in a network of 3 hosts.

Source Code and Output:-

To configure NFS (Network File System) in a network of three hosts running Ubuntu, you'll need to perform the following steps on each host:

1. Install NFS packages

```
sudo apt update
sudo apt install nfs-kernel-server nfs-common
```



2. Creating a directory that will be shared among client systems. This is also referred to as the export directory and it's in this directory that we shall later create files that will be accessible by client systems. Run the command below by specifying the NFS mount directory name.

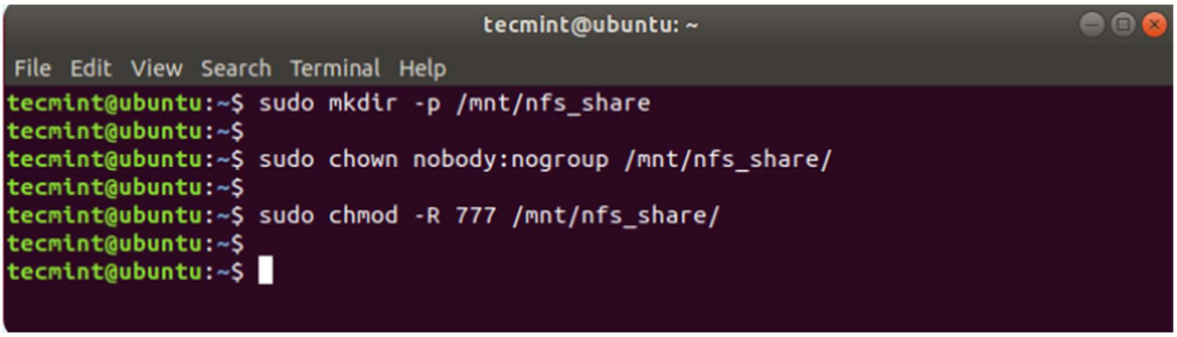
```
sudo mkdir -p /mnt/nfs_share
```

We want all the client machines to access the shared directory, remove any restrictions in the directory permissions.

```
sudo chown -R nobody:nogroup /mnt/nfs_share/
```

You can also tweak the file permissions to your preference.

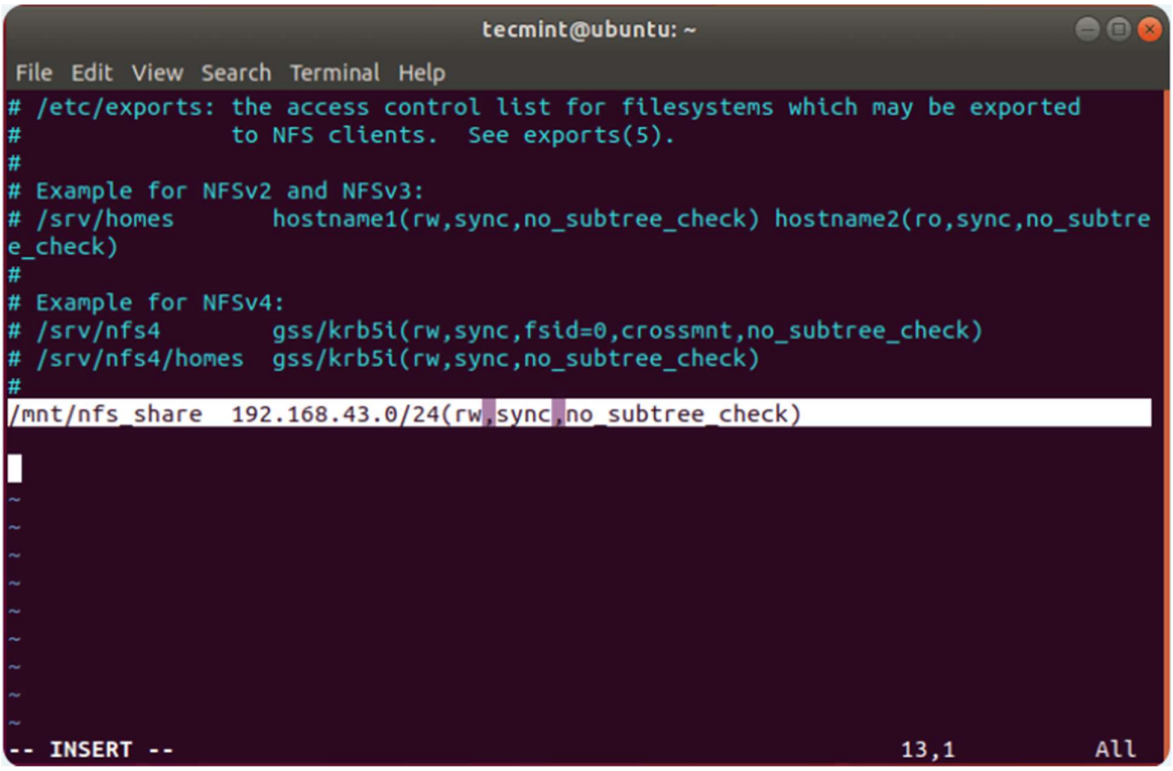
```
sudo chmod 777 /mnt/nfs_share/
```



3. Permissions for accessing the NFS server are defined in the /etc/exports file. So open the file using your favorite text editor:

```
sudo vim /etc/exports
```

```
/mnt/nfs_share 192.168.43.0/24(rw,sync,no_subtree_check)
```

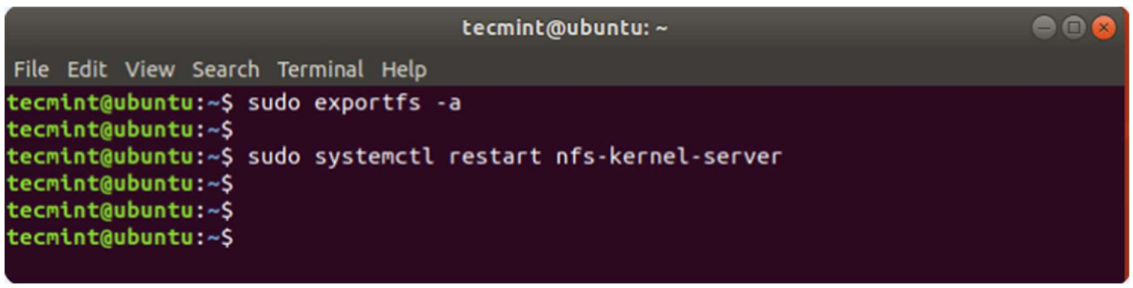


For multiple clients, specify each client on a separate file:

```
/mnt/nfs_share  client_IP_1 (re,sync,no_subtree_check)  client_IP_2
(re,sync,no_subtree_check)  client_IP_3 (re,sync,no_subtree_check)
```

4. After granting access to the preferred client systems, export the NFS share directory and restart the NFS kernel server for the changes to come into effect.

```
sudo exportfs -a
sudo systemctl restart nfs-kernel-server
```

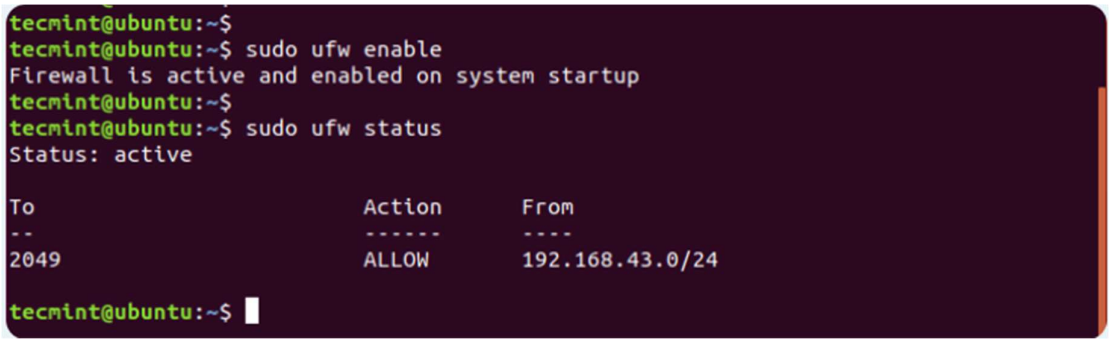


5. For the client to access the NFS share, you need to allow access through the firewall otherwise, accessing and mounting the shared directory will be impossible. To achieve this run the command:

```
sudo ufw allow from 192.168.43.0/24 to any port nfs
```

Reload or enable the firewall (if it was turned off) and check the status of the firewall.

```
sudo ufw enable
sudo ufw status
```



Experiment – 14

Aim:-

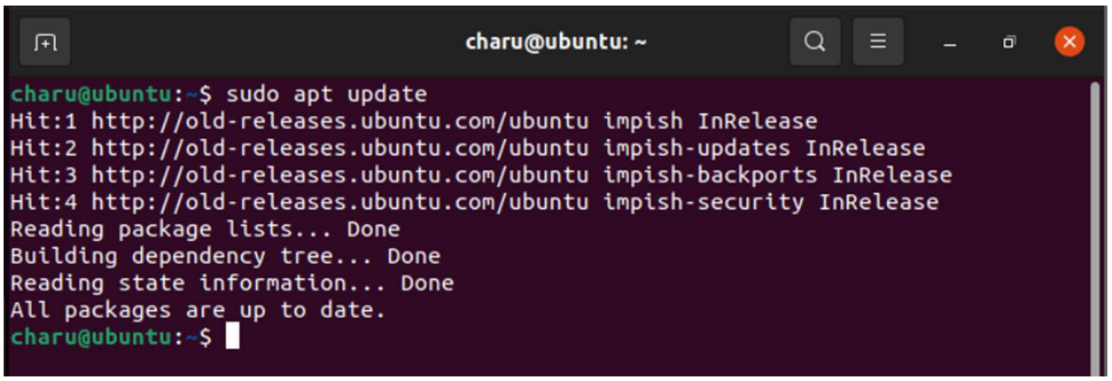
Configure samba server in a LAN.

Source Code and Output:-

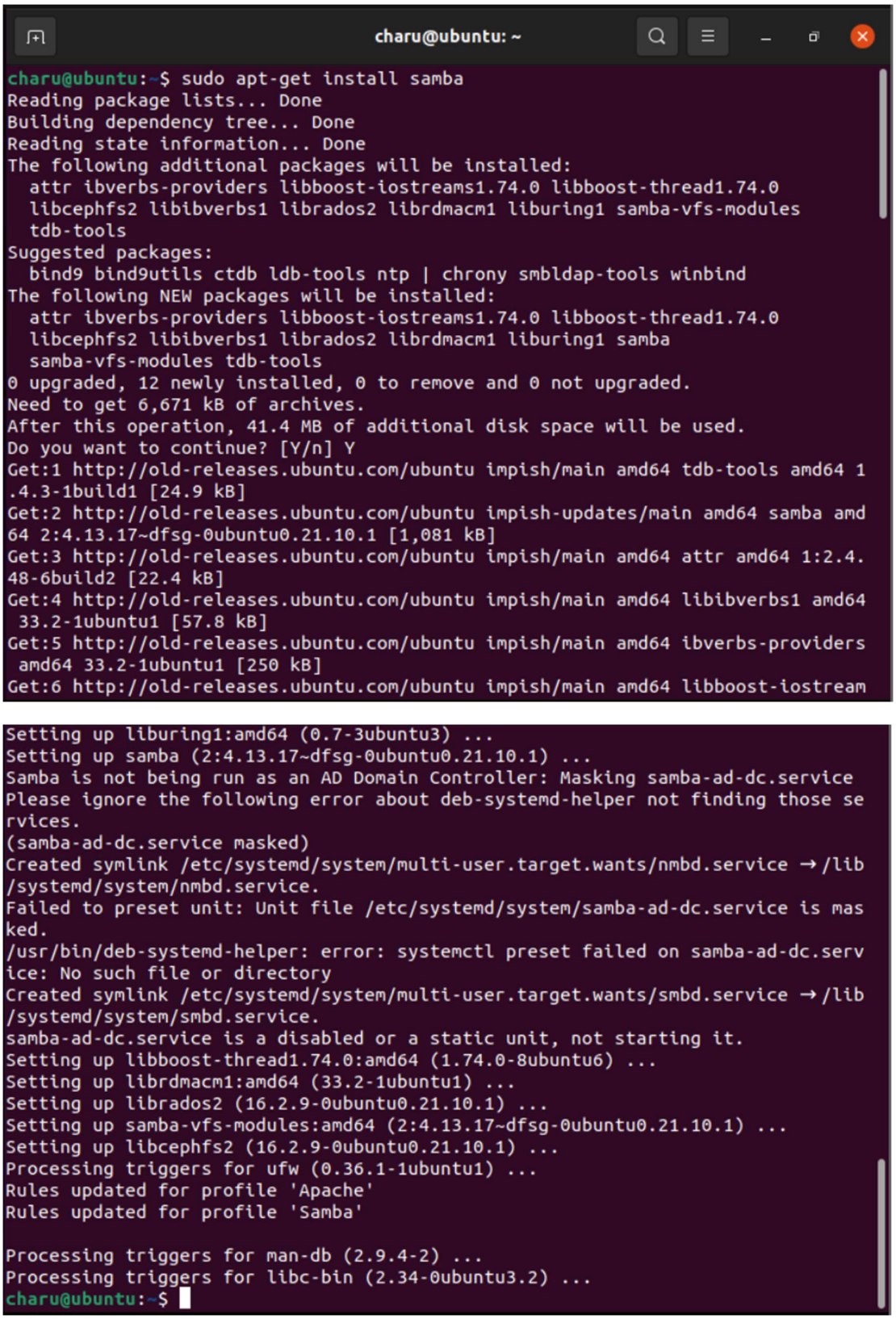
Steps for installation of Samba server:-

1. To install Samba, we run,

   ```
   sudo apt update
   ```



sudo apt-get install samba



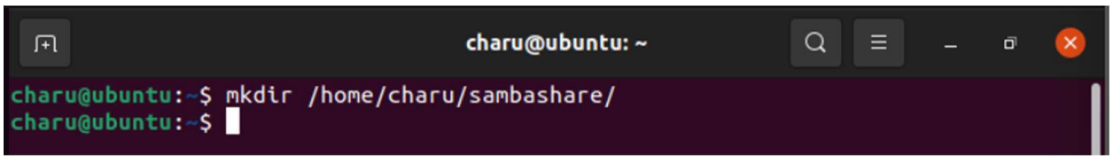
2. We can check if the installation was successful by running,

   `whereis samba`

3. Setting up Samba

   Now that samba is installed, we need to create a directory for it to share.

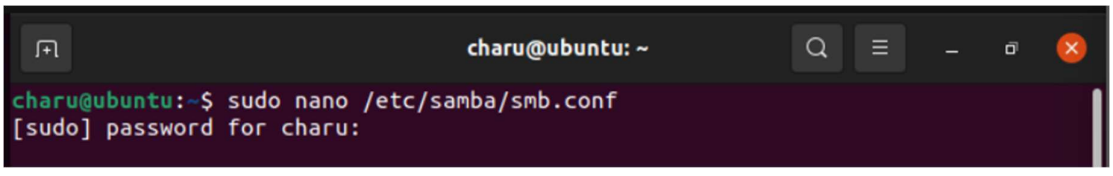
   `mkdir /home/username/sambashare/`



   The command above creates a new folder sambashare in our home directory which we will share later.

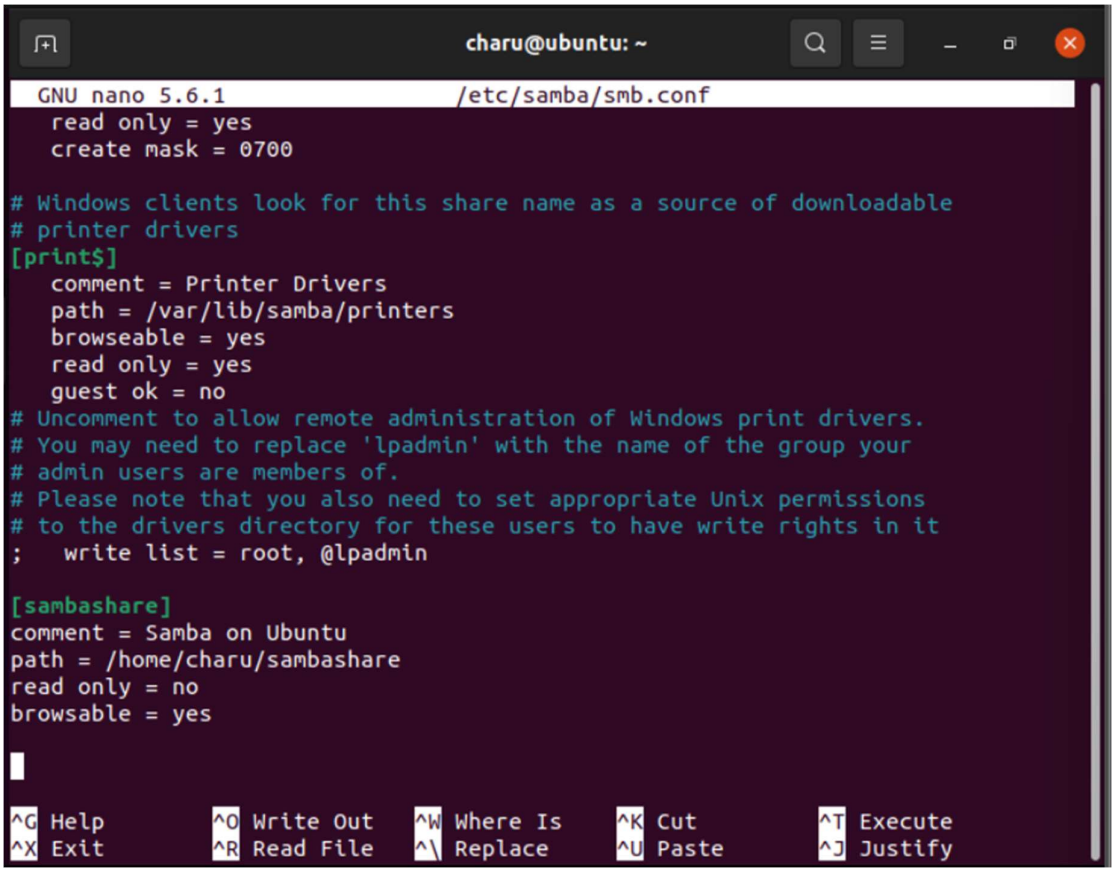
4. The configuration file for Samba is located at /etc/samba/smb.conf. To add the new directory as a share, we edit the file by running:

   `sudo nano /etc/samba/smb.conf`



At the bottom of the file,

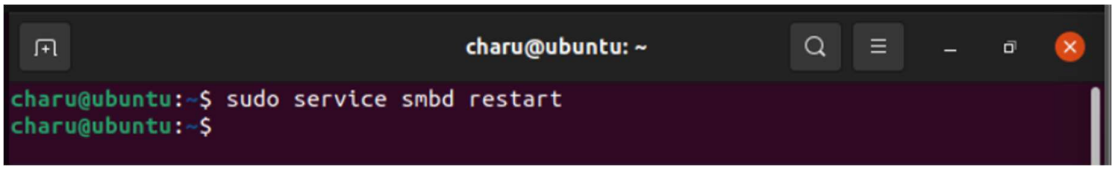
```
[sambashare]
comment = Samba on Ubuntu
path = /home/username/sambashare
read only = no
browsable = yes
```



Then press Ctrl-O to save and Ctrl-X to exit from the nano text editor.

5. Now that we have our new share configured, save it and restart Samba for it to take effect:

```
sudo service smbd restart
```



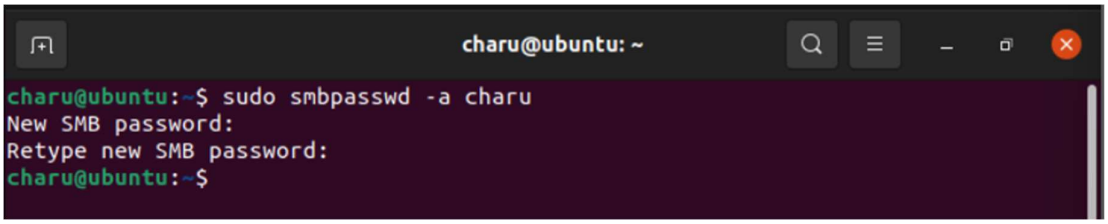
6. Update the firewall rules to allow Samba traffic:

   `sudo ufw allow samba`

7. Setting up User Accounts and Connecting to Share

   Since Samba doesn't use the system account password, we need to set up a Samba password for our user account:

   `sudo smbpasswd -a username`



8. On Windows, open up File Manager and edit the file path to:

   \\ip-address\sambashare

Experiment – 15

Aim:-

Configure DNS server with primary and secondary nodes.

Source Code and Output:-

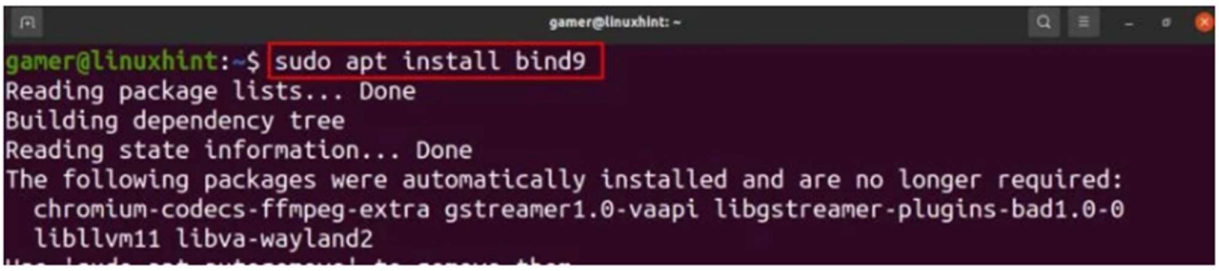
To configure a DNS server with primary and secondary nodes in Ubuntu, you can use the BIND (Berkeley Internet Name Domain) software. Here's a step-by-step guide on how to set it up:

1. Before the installation process we will update our repository:

```
sudo apt update
```

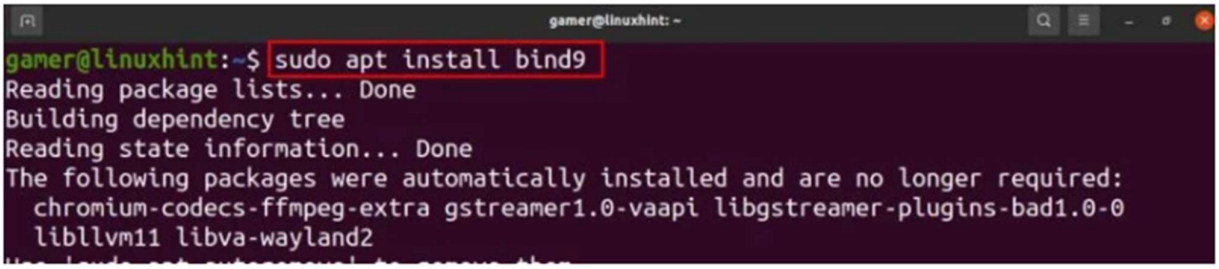
2. Install the DNS server by using the command bind9:

```
sudo apt install bind9
```



Install the utilities of the DNS by using the command of “dnsutils”:

```
sudo apt install dnsutils
```



3. To configure the DNS, we will first go to the address /etc/bind/named.conf.options and add the Google DNS for just understanding. We will add the following text by opening the address in the nano editor.

```
sudo nano /etc/bind/named.conf.options
```

Replace the following text in the editor, 8.8.8.8 is Google’s DNS:

```
forwarders {
8.8.8.8;
};
```

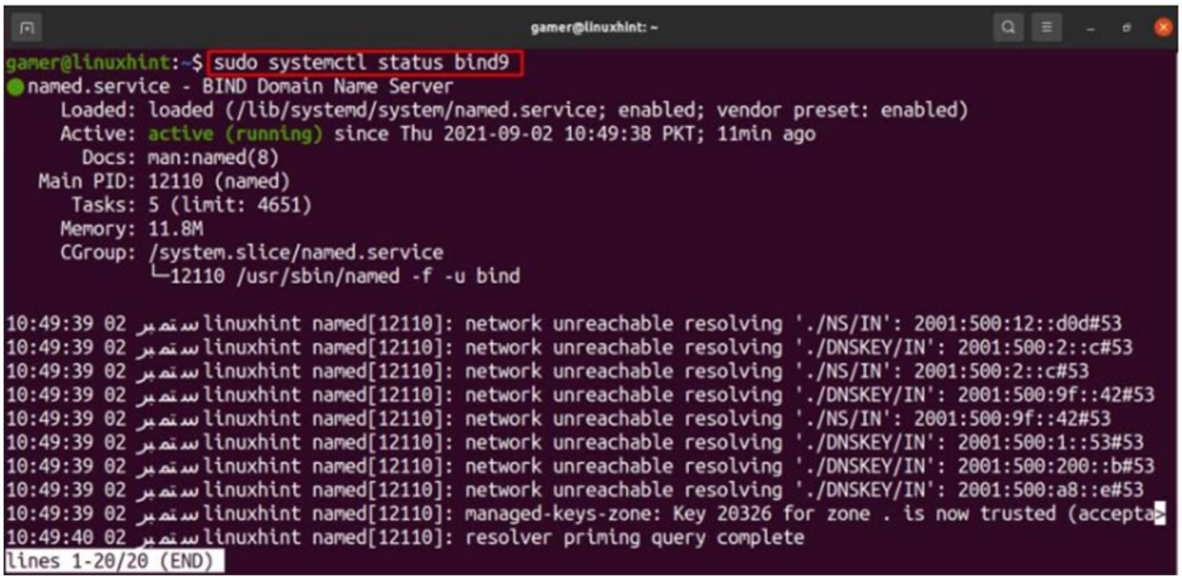
Now quit after saving it

4. Enable the new configuration by restarting the DNS using the systemctl command.

```
sudo systemctl restart bind9
```

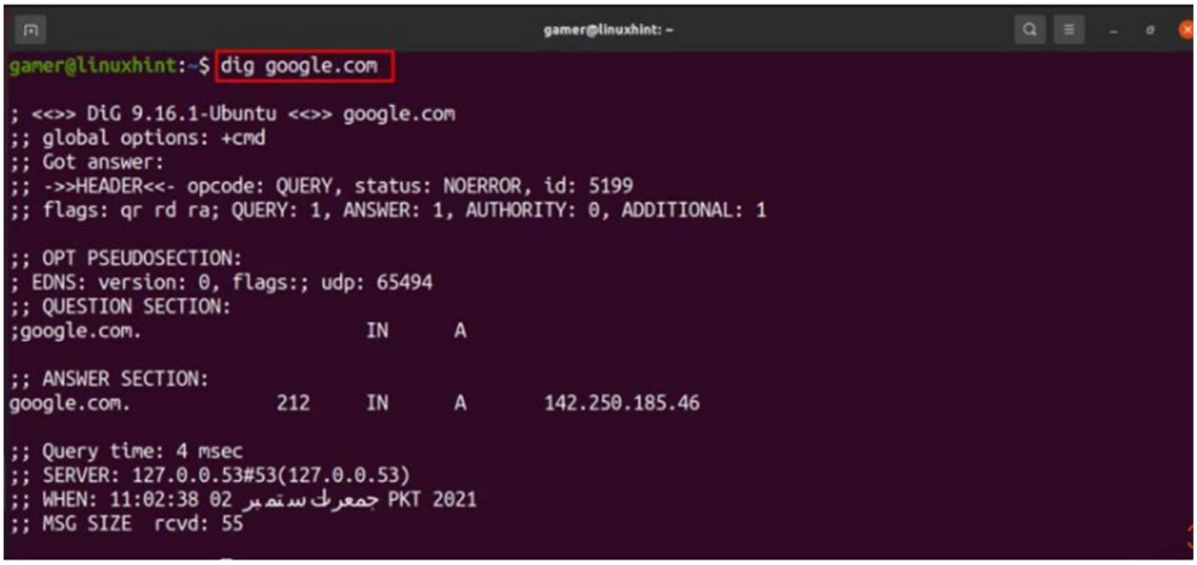
5. Check the status of bind9

```
sudo systemctl status bind9
```



6. As the bind9 is running now we will test the domain which we edit in the configuration file as:

```
dig google.com
```



The output is showing it is the domain of Google and it runs successfully.

Experiment – 16

Aim:-

Configure SSH server and telnet server to access linux hosts remotely.

a. Copy the files from source to destination
b. Restrict the access of ssh at logical address level

Source Code and Output:-

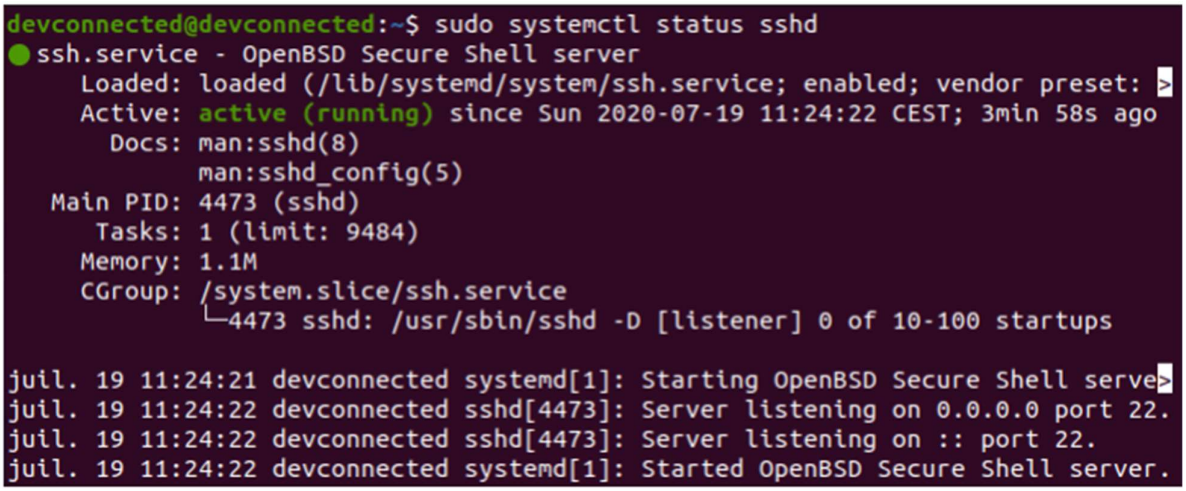
1. Make sure that your current packages are up to date for security purposes.
   sudo apt-get update

```
devconnected@devconnected:~$ sudo apt-get update
[sudo] password for devconnected:
Hit:1 http://fr.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://fr.archive.ubuntu.com/ubuntu focal-updates InRelease [111 kB]
Get:3 http://fr.archive.ubuntu.com/ubuntu focal-backports InRelease [98,3 kB]
Get:4 http://security.ubuntu.com/ubuntu focal-security InRelease [107 kB]
Get:5 http://fr.archive.ubuntu.com/ubuntu focal/main DEP-11 48x48 Icons [98,4 k
B]
Get:6 http://fr.archive.ubuntu.com/ubuntu focal/main DEP-11 64x64 Icons [163 kB
]
Get:7 http://fr.archive.ubuntu.com/ubuntu focal/main DEP-11 64x64@2 Icons [15,8
 kB]
Get:8 http://fr.archive.ubuntu.com/ubuntu focal/universe DEP-11 48x48 Icons [3
016 kB]
Get:9 http://fr.archive.ubuntu.com/ubuntu focal/universe DEP-11 64x64 Icons [7
794 kB]
Get:10 http://security.ubuntu.com/ubuntu focal-security/main i386 Packages [46,
6 kB]
```

Now that all packages are up-to-date, run the “apt-get install” command in order to install OpenSSH.

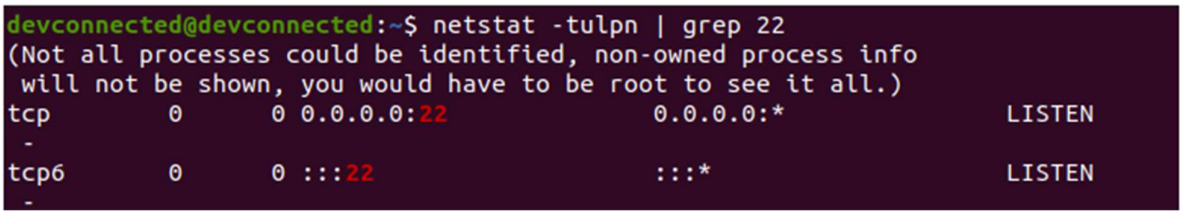
```
sudo apt-get install openssh-server
```

2. A SSH service was created and you can check that it is actually up and running.
   `sudo systemctl status sshd`

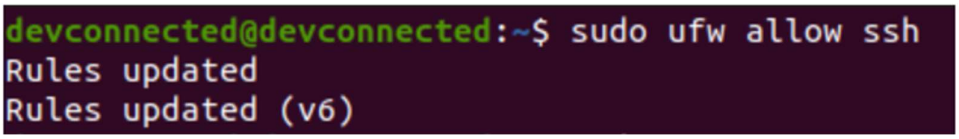


3. By default, your SSH server is listening on port 22 (which is the default SSH port).
   If you want to go into further details, you can actually check that the SSH server is listening on port 22 with the netstat command.

   `netstat -tulpn | grep 22^`



4. To enable SSH connections on your host, run the following command
   `sudo ufw allow ssh`



If you are not sure if you are actively using the UFW firewall, you can run the “ufw status” command.

```
sudo ufw status
```

```
devconnected@devconnected:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
```

5. To check whether your service is enable or not, you can run the following command

```
sudo systemctl list-unit-files | grep enabled | grep ssh
```

If you have no results on your terminal, you should “enable” the service in order for it to be launched at boot time.

```
sudo systemctl enable ssh
```

```
devconnected@devconnected:~$ sudo systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /lib/systemd/s
ystemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable ssh
devconnected@devconnected:~$ sudo systemctl enable sshd
Failed to enable unit: Refusing to operate on alias name or linked unit file: s
shd.service
```

6. By default, SSH configuration files are located in the /etc/ssh folder.

```
devconnected@devconnected:~$ ll /etc/ssh/
total 584
drwxr-xr-x   4 root root   4096 juil. 19 11:24 ./
drwxr-xr-x 131 root root  12288 juil. 19 11:24 ../
-rw-r--r--   1 root root 535195 mai   29 09:37 moduli
-rw-r--r--   1 root root   1603 févr. 26 11:55 ssh_config
drwxr-xr-x   2 root root   4096 févr. 26 11:55 ssh_config.d/
-rw-r--r--   1 root root   3289 mai   29 09:37 sshd_config
drwxr-xr-x   2 root root   4096 mai   29 09:37 sshd_config.d/
-rw-------   1 root root    513 juil. 19 11:24 ssh_host_ecdsa_key
-rw-r--r--   1 root root    179 juil. 19 11:24 ssh_host_ecdsa_key.pub
-rw-------   1 root root    411 juil. 19 11:24 ssh_host_ed25519_key
-rw-r--r--   1 root root     99 juil. 19 11:24 ssh_host_ed25519_key.pub
-rw-------   1 root root   2602 juil. 19 11:24 ssh_host_rsa_key
-rw-r--r--   1 root root    571 juil. 19 11:24 ssh_host_rsa_key.pub
-rw-r--r--   1 root root    342 juil. 19 11:24 ssh_import_id
```

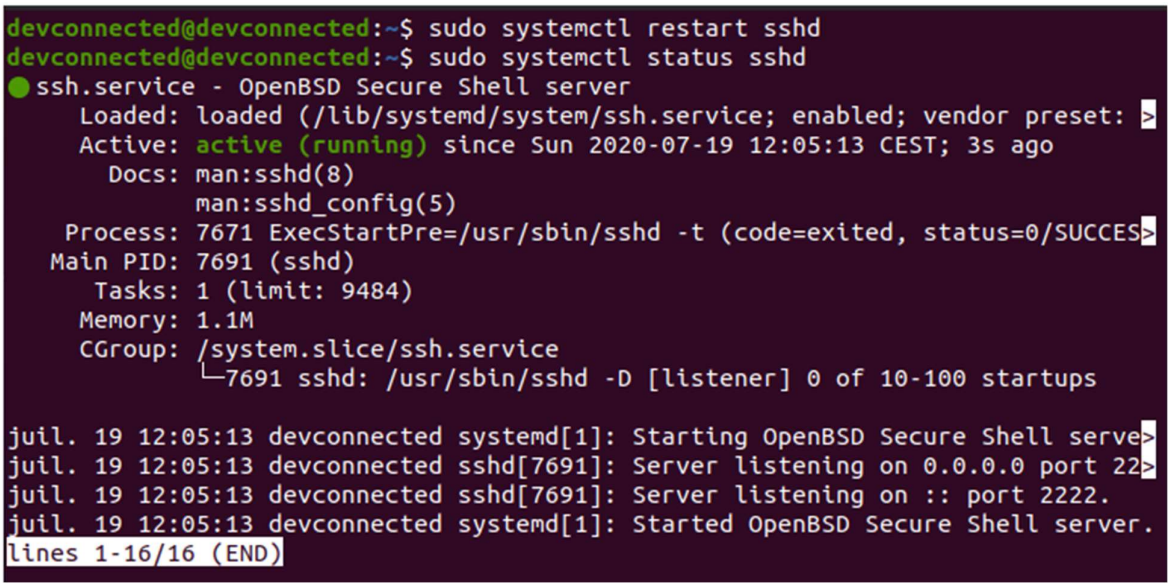
7. Edit your sshd_config configuration file and look for the following line.

   ```
   sudo nano /etc/ssh/sshd_config
   ```

   Change your port to one that is not reserved for other protocols. I will choose 2222 in this case.

8. In order for the changes to be applied, you need to restart your SSH server.

   ```
   sudo systemctl restart sshd
   sudo systemctl status sshd
   ```



9. In order to connect to your SSH server, you are going to use the ssh command with the following syntax
   `ssh -p <port> <username>@<ip_address>`

```
devconnected@devconnected:~$ ssh -p 2222 devconnected@127.0.0.1
The authenticity of host '[127.0.0.1]:2222 ([127.0.0.1]:2222)' can't be establi
shed.
ECDSA key fingerprint is SHA256:cTQt0Z1YbOxq6tV8ewdyPzEkOllX8RpKI3iNgK4bBc8.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[127.0.0.1]:2222' (ECDSA) to the list of known host
s.
devconnected@127.0.0.1's password:
Welcome to Ubuntu 20.04 LTS (GNU/Linux 5.4.0-40-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage


48 updates can be installed immediately.
7 of these updates are security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

Experiment – 17

Aim:-

Configure the firewall services with various zones and add the http, FTP, NFS and SSH services permanently using protocol and corresponding port number.

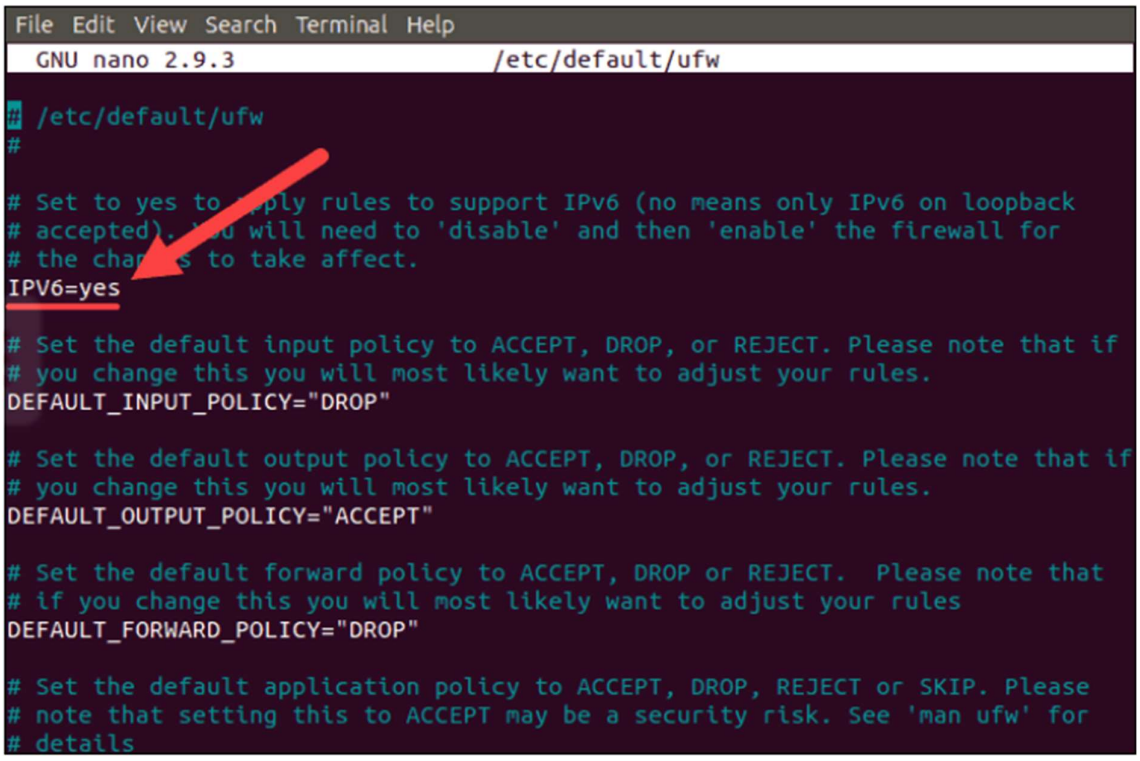
Source Code and Output:-

1. UFW comes pre-installed on Ubuntu 20.04 and Ubuntu 22.04. In case you do not have UFW, run the following command to install it:

```
sudo apt install ufw
```

2. Open the default settings file using nano or any other text editor:

```
sudo nano /etc/default/ufw
```



If the IPv6 value is set as no, change the value to yes to enable IPv6 use. Save and close the file.

3. If you changed the default settings and want to return to the default behavior, run the following command to deny incoming connections:

   ```
   sudo ufw default deny incoming
   ```

   Allow outgoing connections by running:

   ```
   sudo ufw default allow outgoing
   ```

```
kb@phoenixNAP:~$ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
kb@phoenixNAP:~$ sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
```

4. Configure UFW to allow SSH connections with the command:
   ```
   sudo ufw allow ssh
   ```

```
sofija@sofija-VirtualBox:~$ sudo ufw allow ssh
[sudo] password for sofija:
Rules updated
Rules updated (v6)
```

5. After configuring the settings, disable and enable the UFW firewall for the changes to take effect. Disable UFW by entering:
   ```
   sudo ufw disable
   ```

   Enable the firewall again with the following command:
   ```
   sudo ufw enable
   ```

```
kb@phoenixNAP:~$ sudo ufw disable
Firewall stopped and disabled on system startup
kb@phoenixNAP:~$ sudo ufw enable
Firewall is active and enabled on system startup
```

6. To check UFW status and show detailed information, run the following command:
   `sudo ufw status verbose`

```
kb@phoenixNAP:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW IN    Anywhere
22/tcp (v6)                ALLOW IN    Anywhere (v6)
```

7. Set your server to listen to HTTP by running:
   `sudo ufw allow http`

   Alternatively, use port number 80 for HTTP connections:
   sudo ufw allow 80

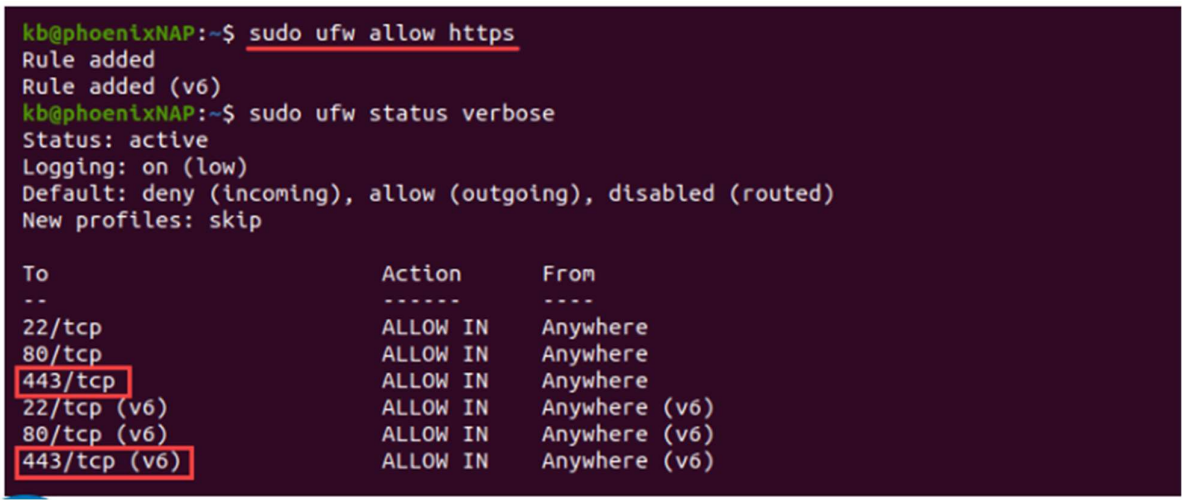
   The rule is visible in the UFW status:
   sudo ufw status verbose

```
kb@phoenixNAP:~$ sudo ufw allow http
[sudo] password for kb:
Rule added
Rule added (v6)
kb@phoenixNAP:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW IN    Anywhere
80/tcp                     ALLOW IN    Anywhere
22/tcp (v6)                ALLOW IN    Anywhere (v6)
80/tcp (v6)                ALLOW IN    Anywhere (v6)
```

8. To enable HTTPS connections, use the following command:
   `sudo ufw allow https`

   Alternatively, use port number 443 for HTTPS connections:
   `sudo ufw allow 443`

   Check the UFW status to confirm the new rule is visible:
   `sudo ufw status verbose`



The enabled HTTPS connections on port 443 are visible for IPv4 and IPv6.

9. To set a rule that allows access to all ports from a specific IP address, run:
   `sudo ufw allow from <IP address>`

   To allow access from a particular machine to a specific port, run the command:
   `sudo ufw allow from <IP address> to any port <port number>`

10. To allow access to a range of ports, specify the range values and the protocol type (TCP or UDP).
    ```
    sudo ufw allow 2000:2004/tcp
    ```