

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.

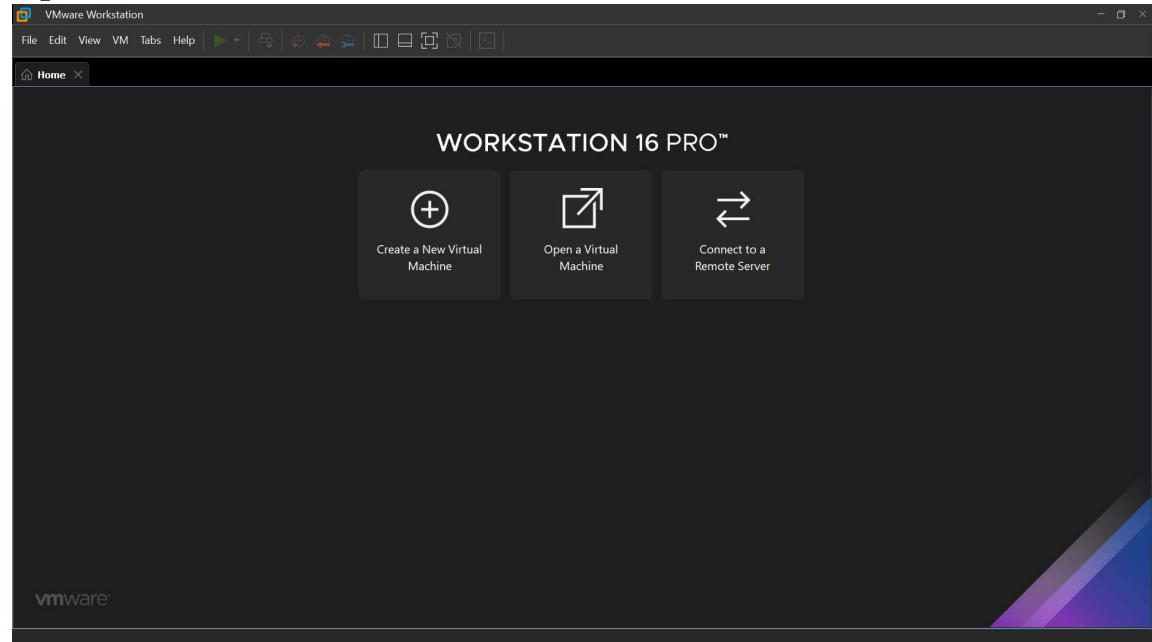
Services of Linux:-

- › 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.

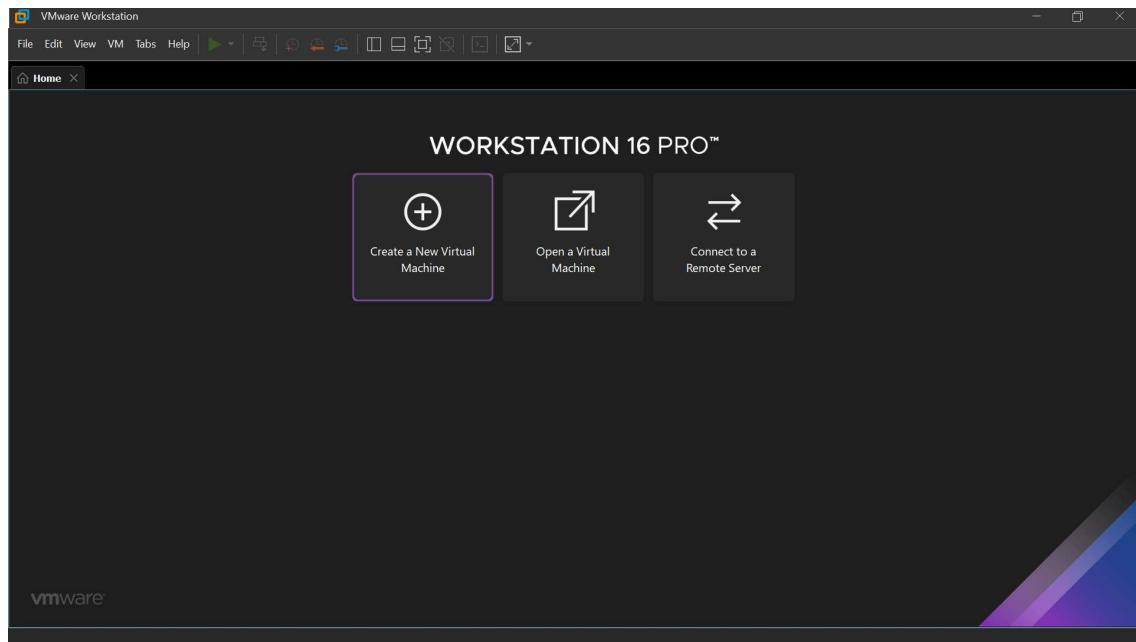
- › DHCP (Dynamic Host Configuration Protocol) - a protocol used to automatically assign IP addresses to devices on a network.
- › Samba - a service for sharing files and printers between Linux and Windows systems.
- › CUPS (Common Unix Printing System) - a printing system for Unix-like operating systems.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

Procedure:-

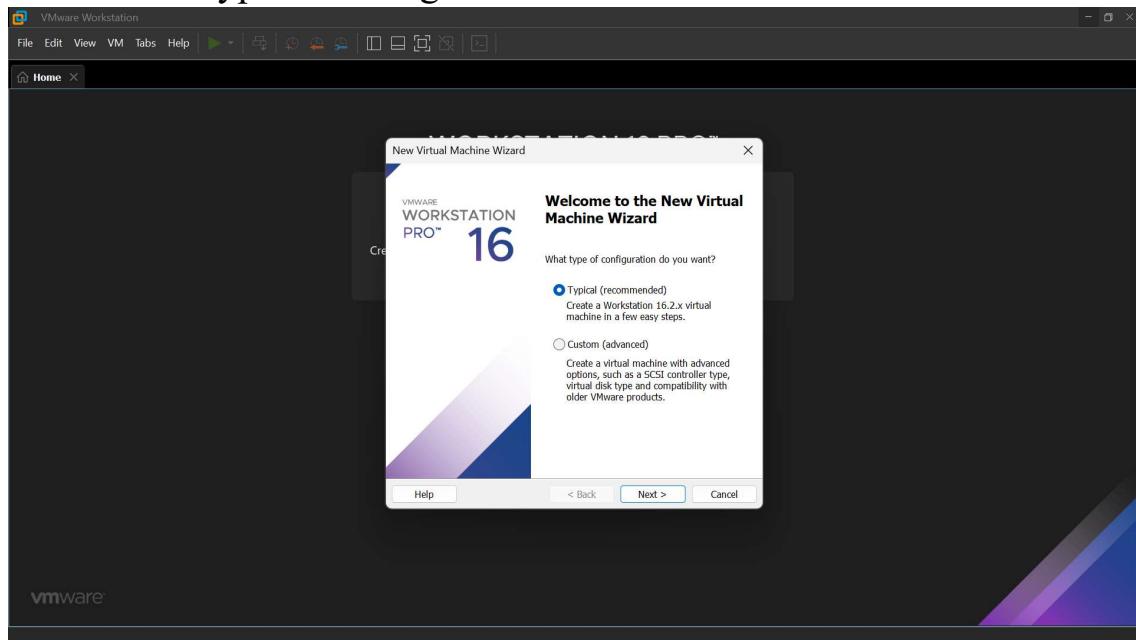
- Open VMware Workstation.



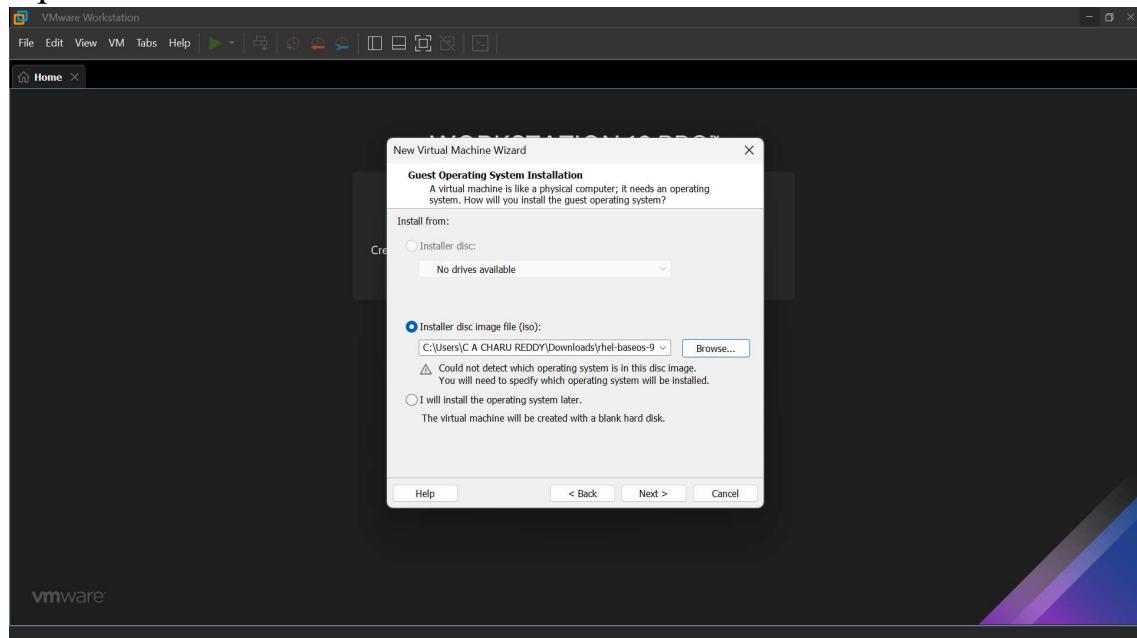
- Create a New Virtual Machine.



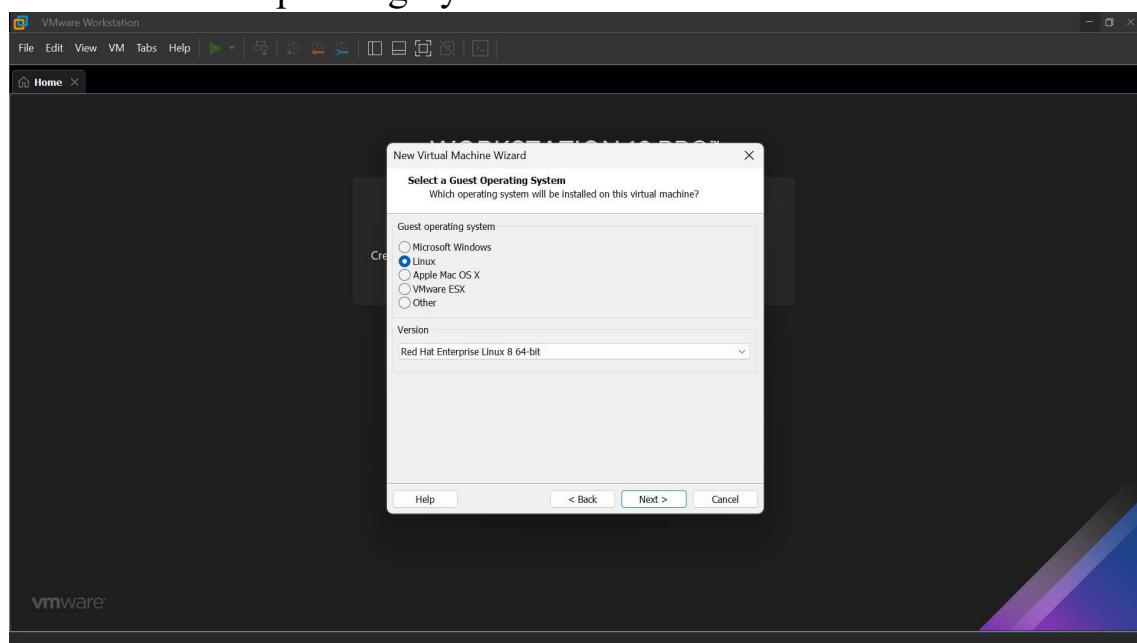
- Choose the Type of Configuration.



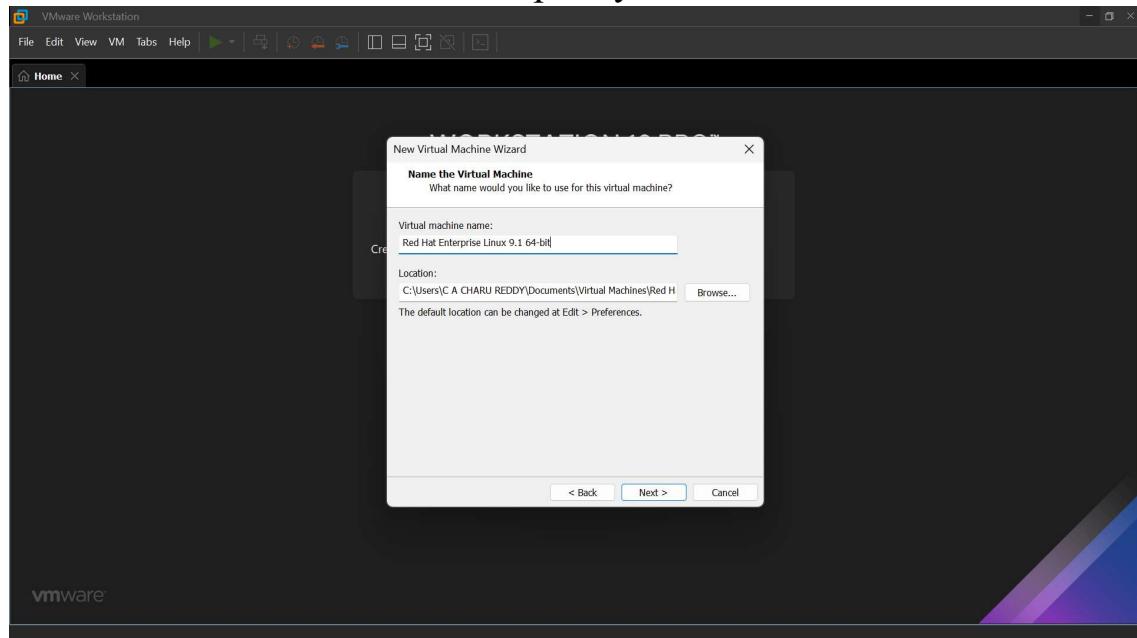
- Upload the rhel.iso file.



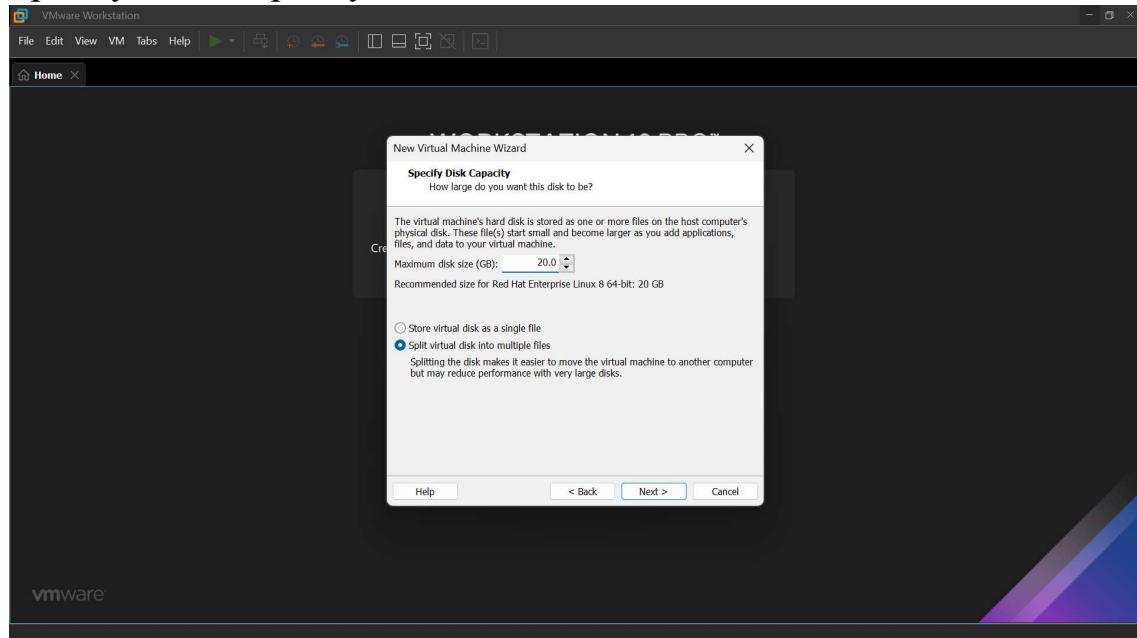
- Select a Guest Operating System and its Version.



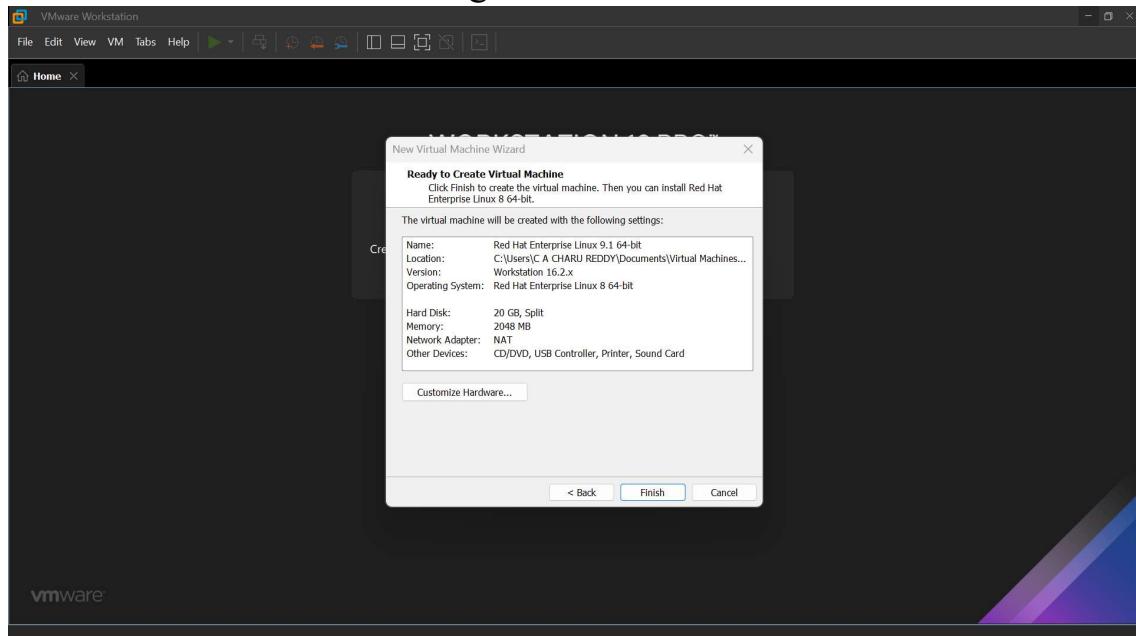
- Name the Virtual Machine and Specify the Location.



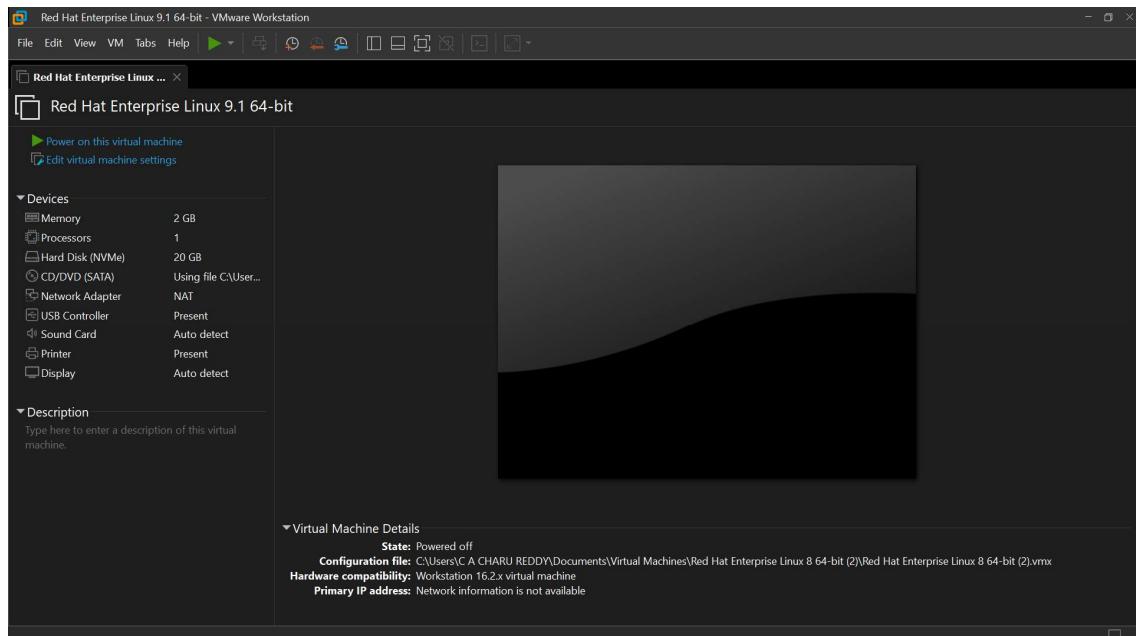
- Specify Disk Capacity.

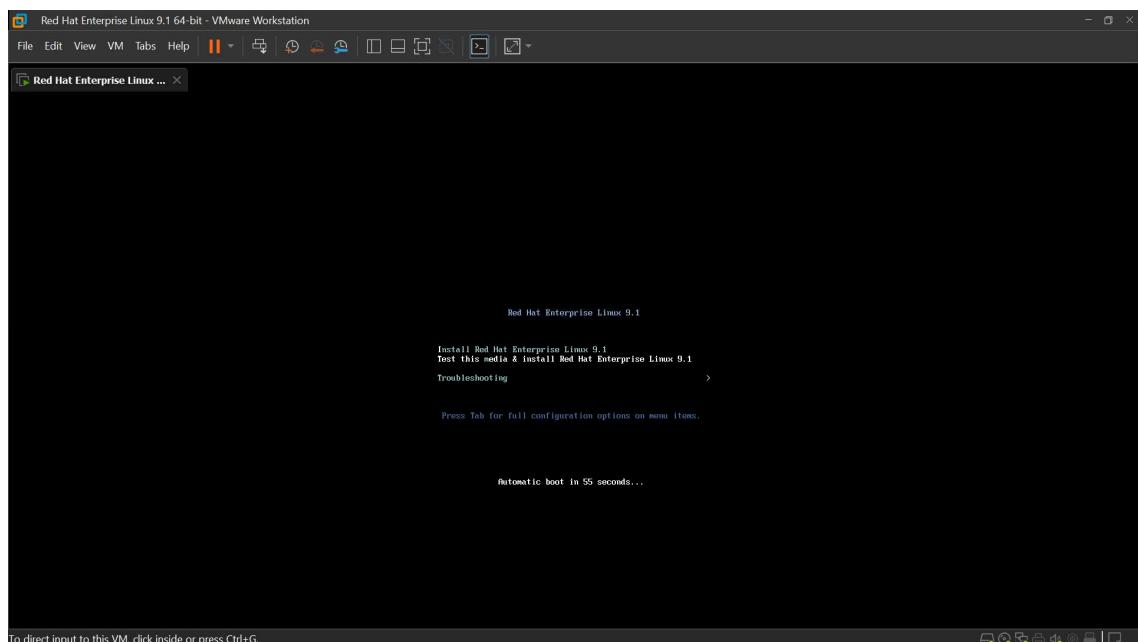
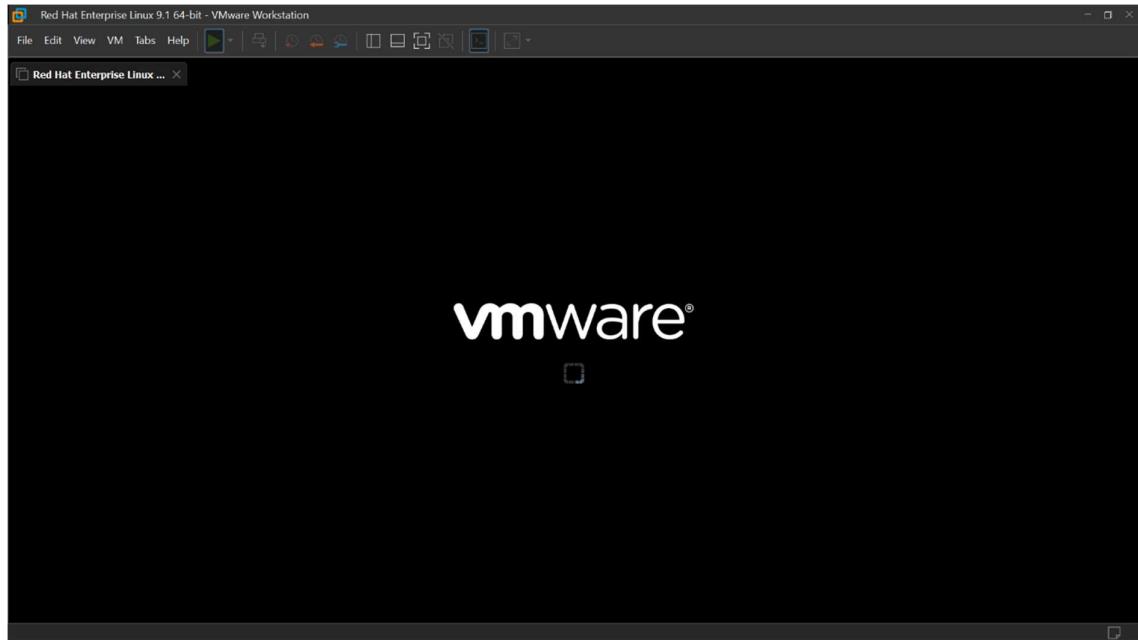


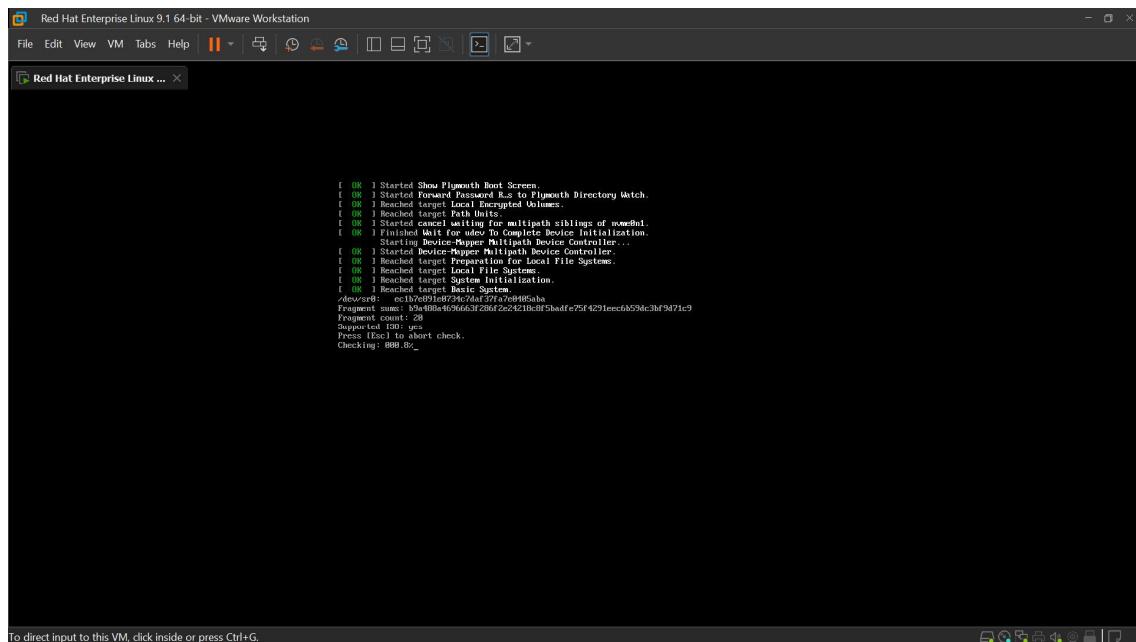
- Preview the Hardware Settings and click on Finish.



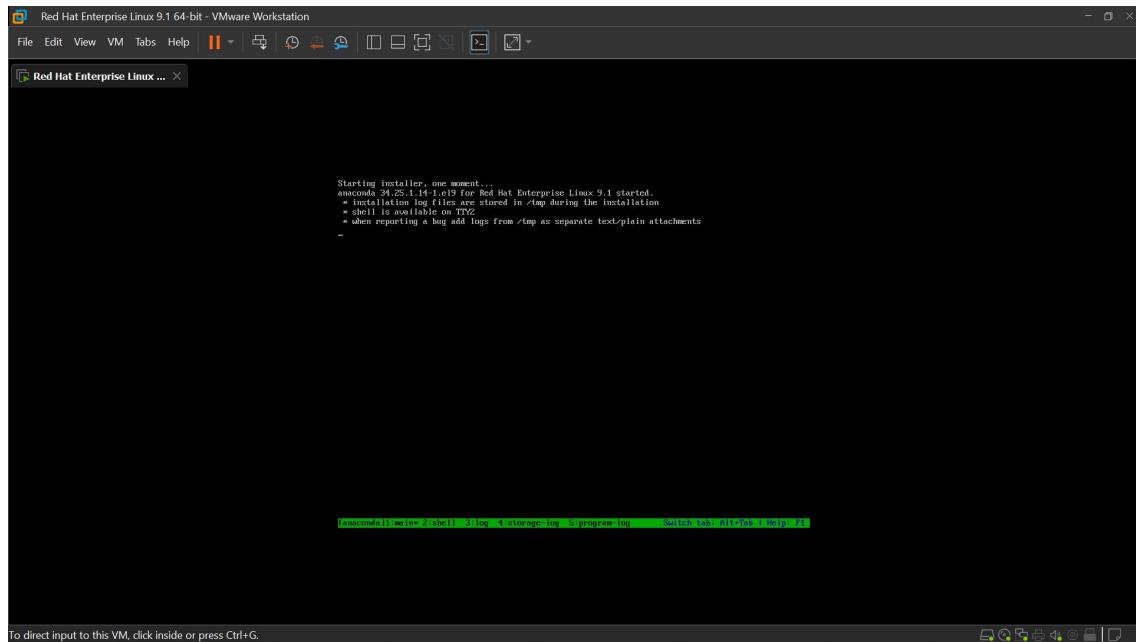
- Power on this virtual machine to install.





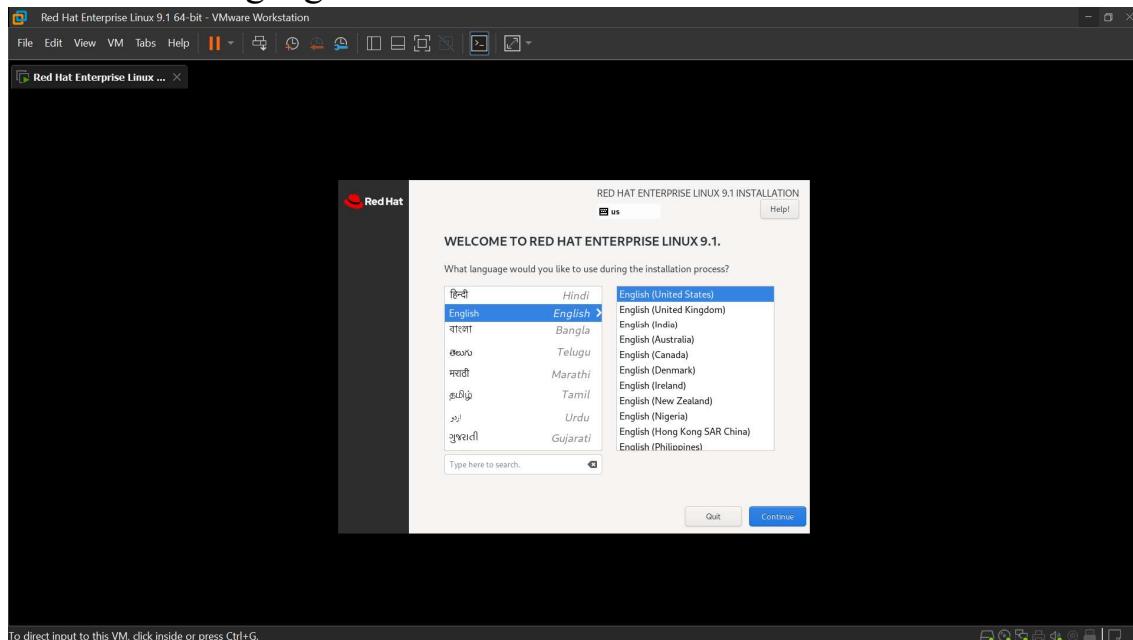


To direct input to this VM, click inside or press Ctrl+G.

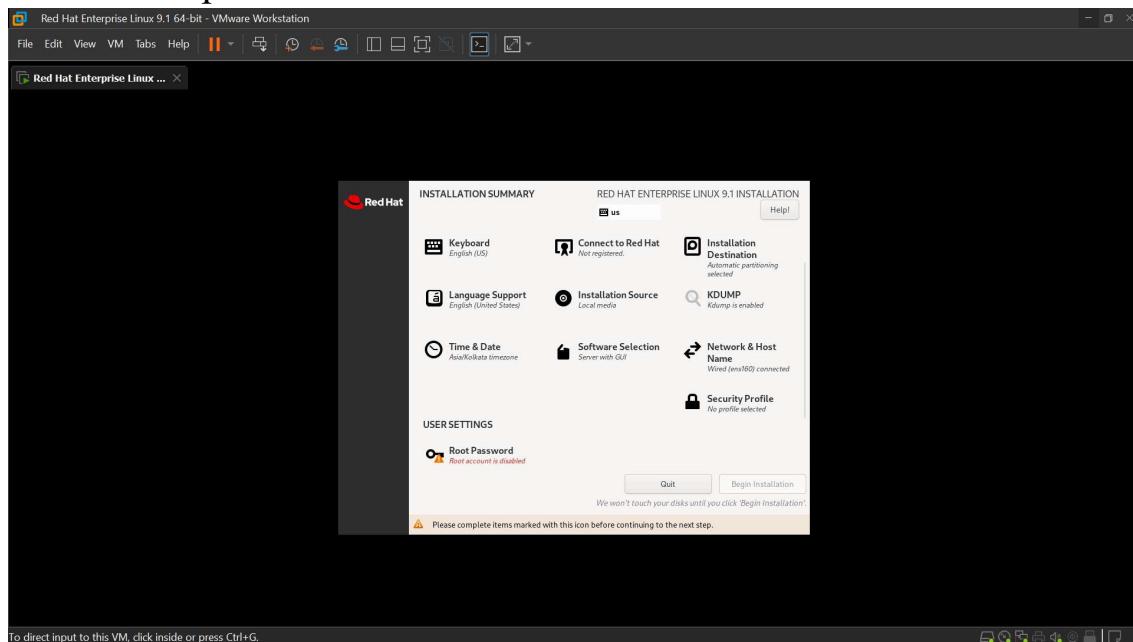


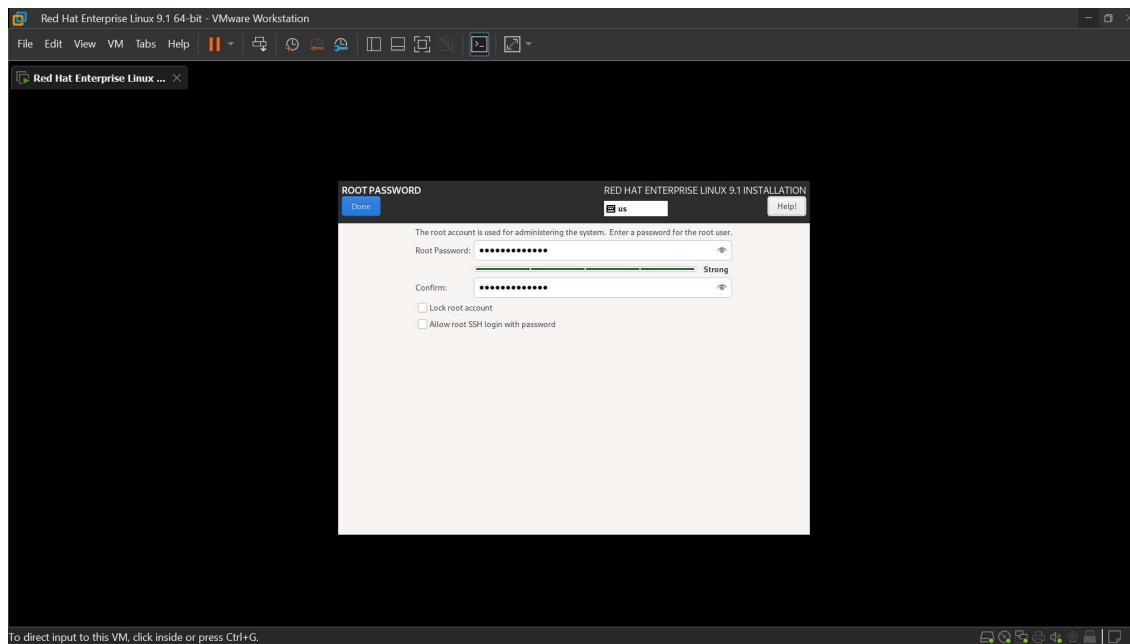
To direct input to this VM, click inside or press Ctrl+G.

- Choose the language.

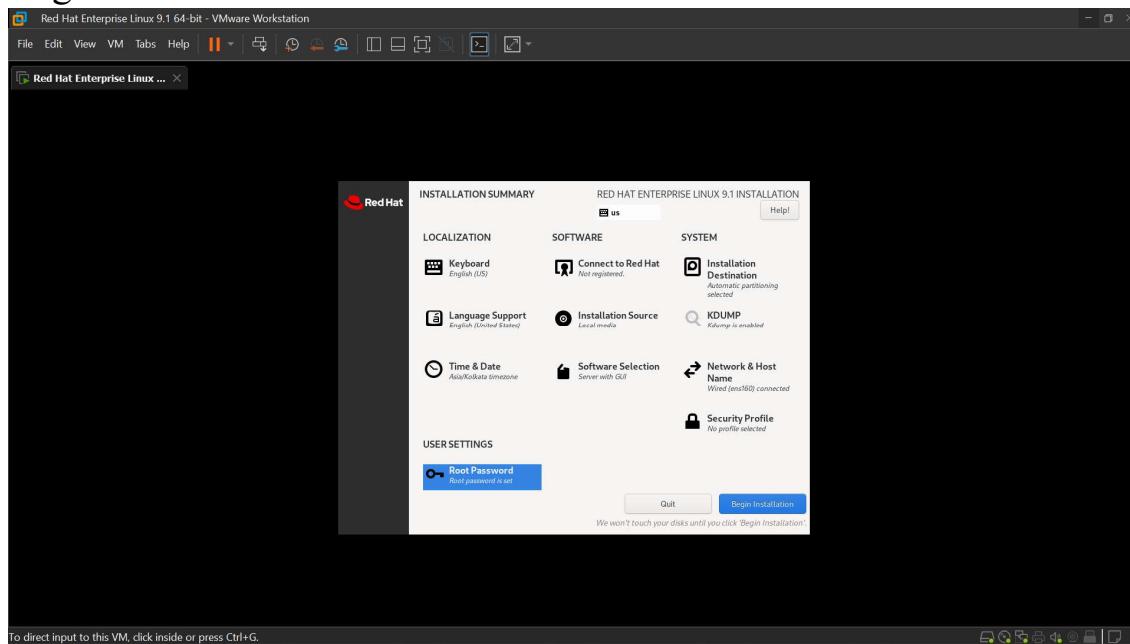


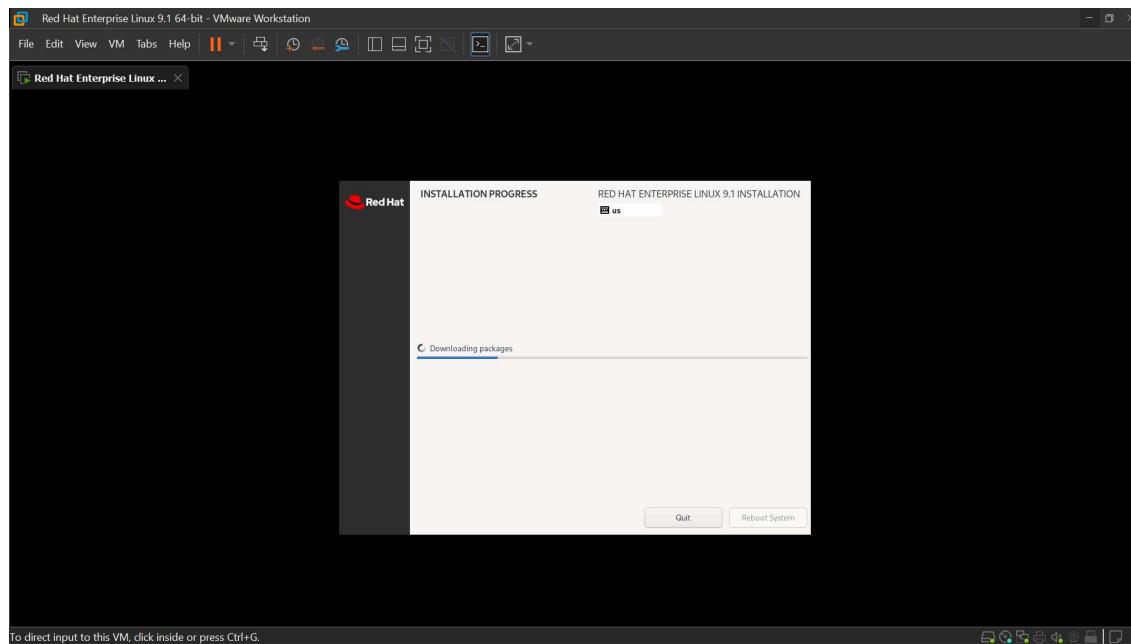
- Set the root password.



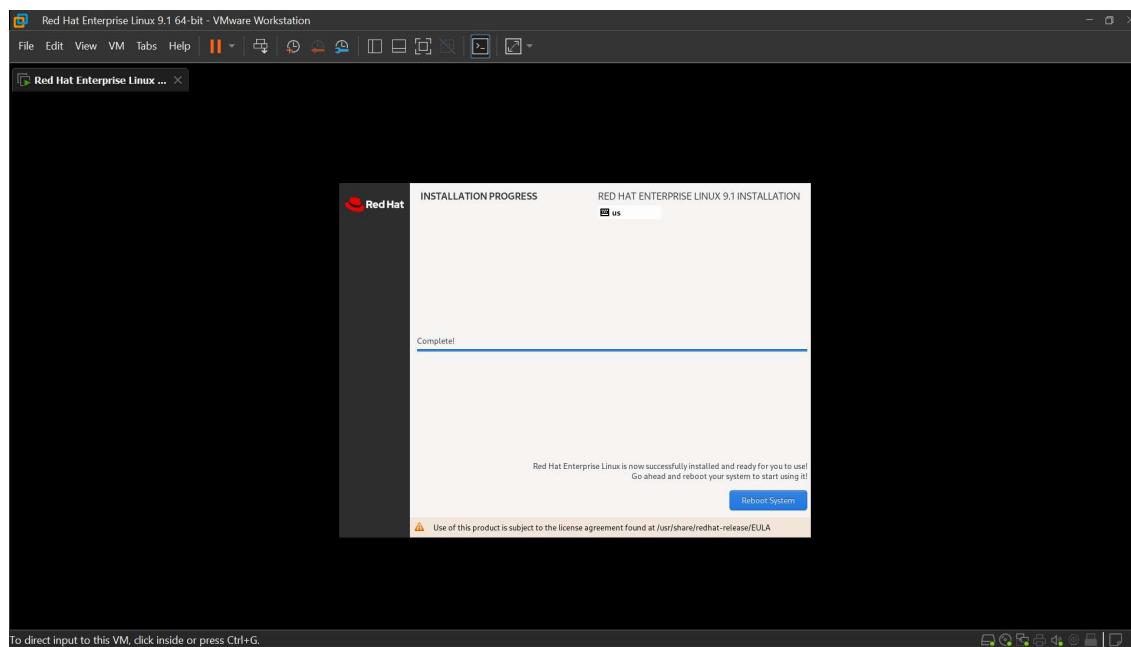


- Begin Installation.

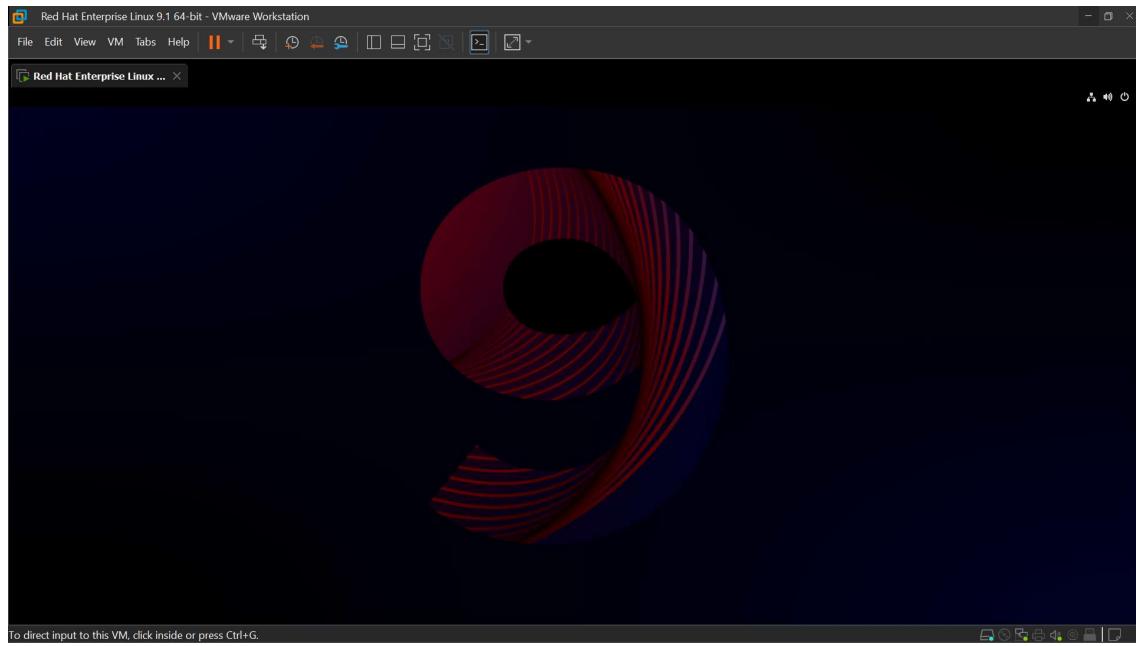




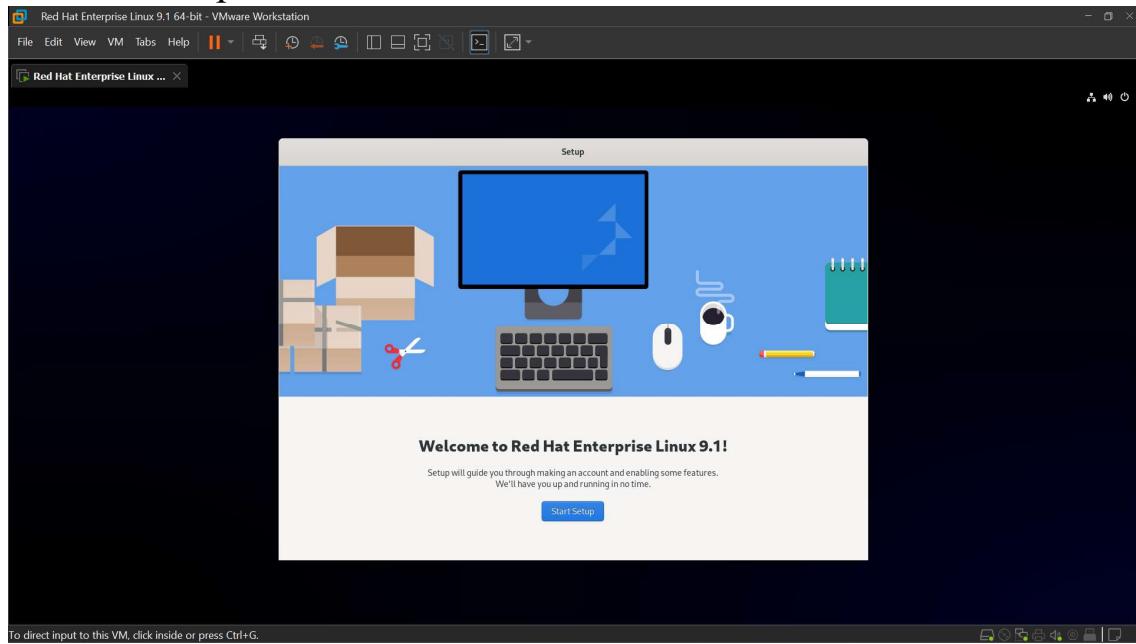
To direct input to this VM, click inside or press Ctrl+G.

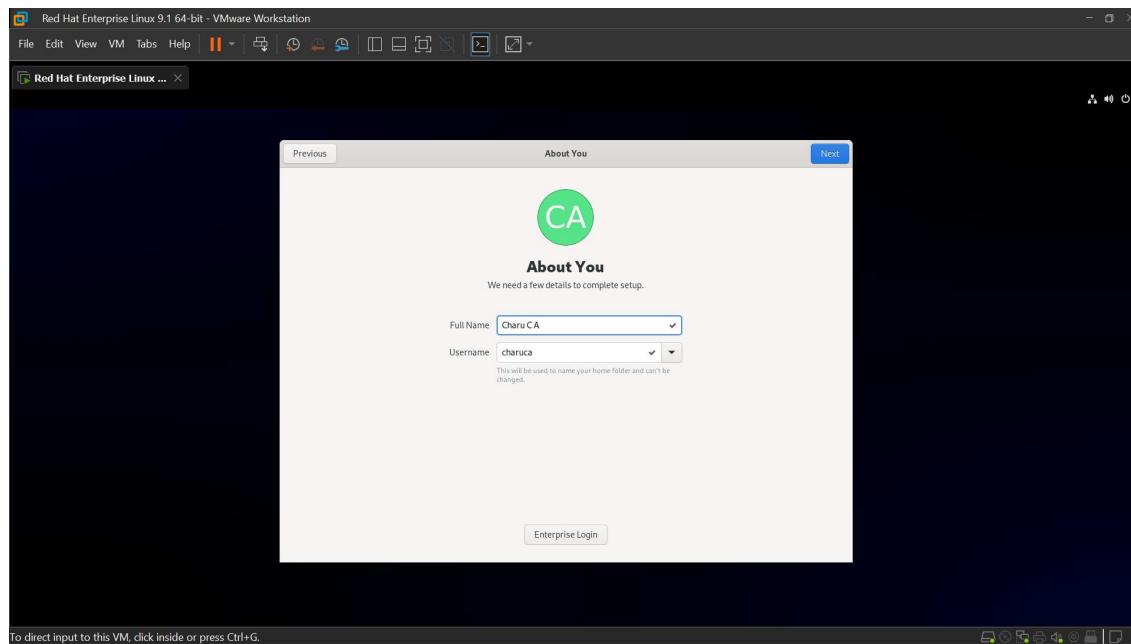


To direct input to this VM, click inside or press Ctrl+G.

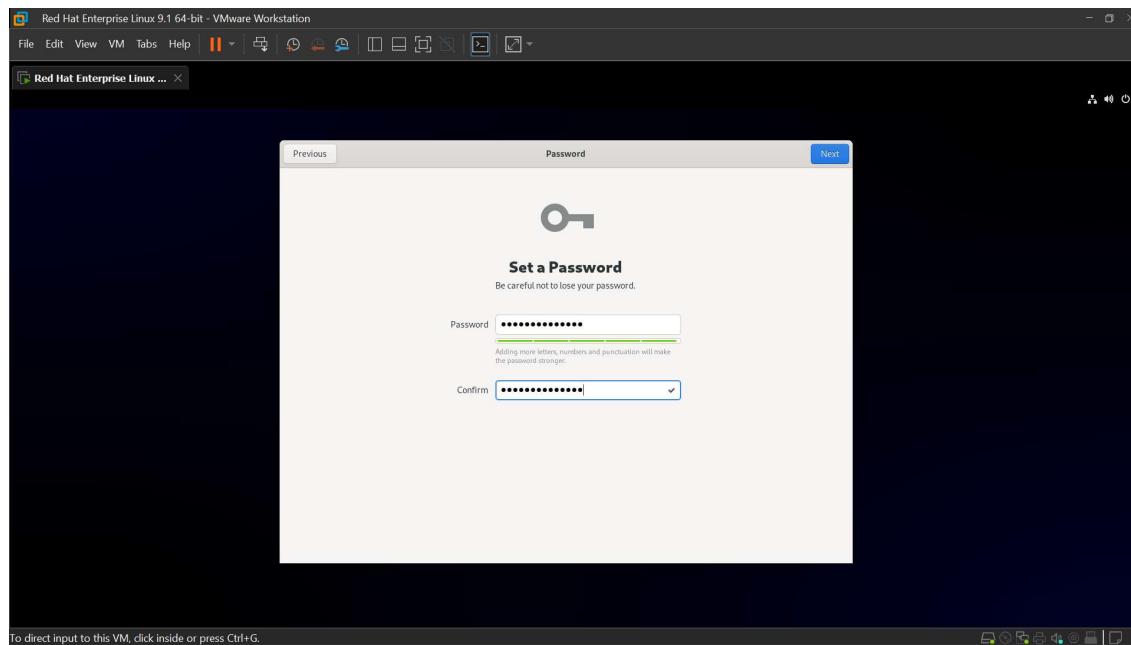


- Start the setup.

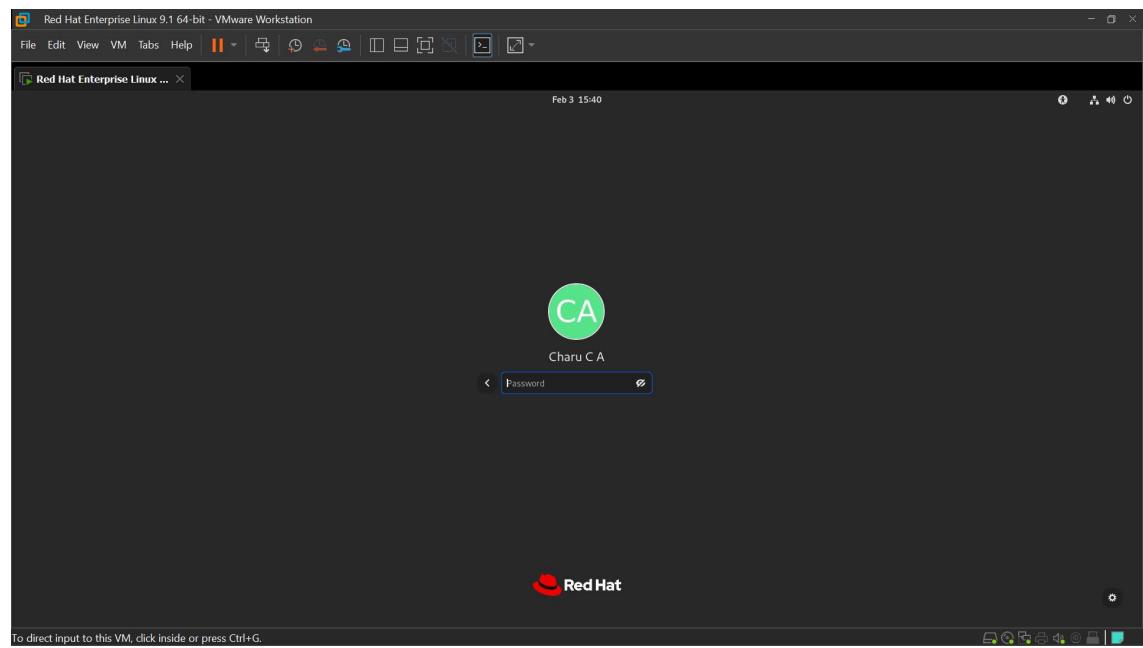
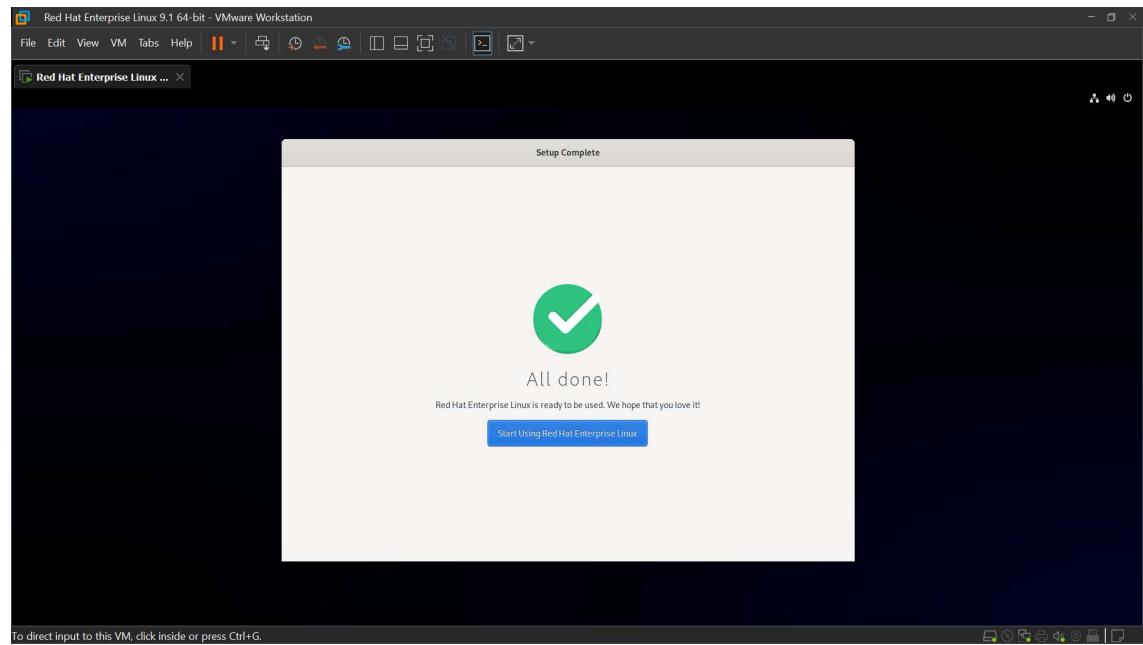


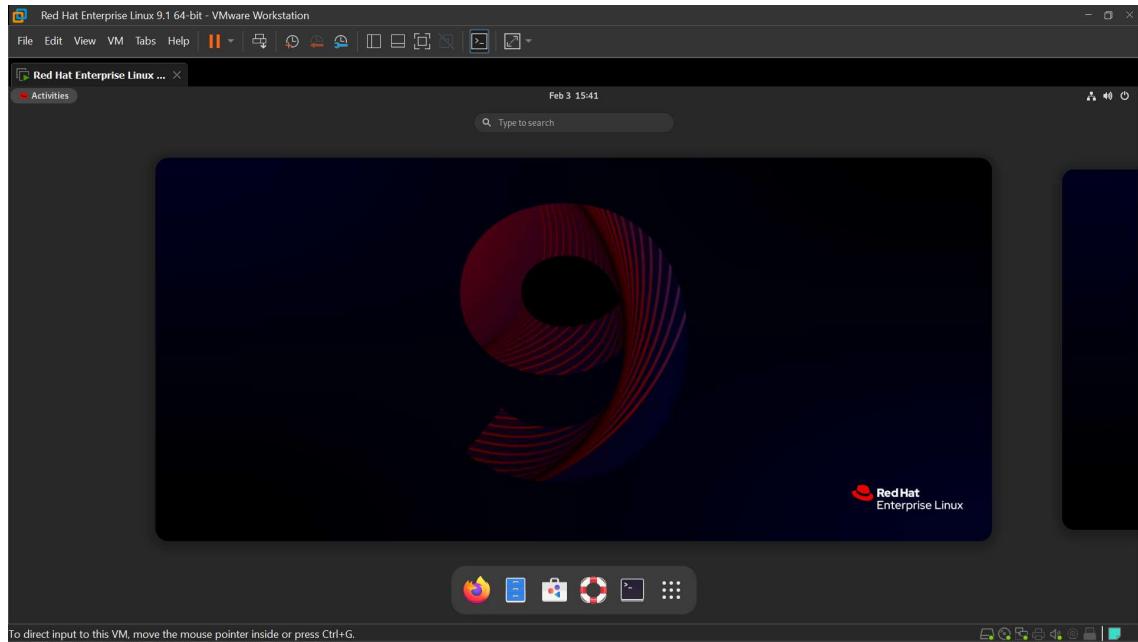


To direct input to this VM, click inside or press Ctrl+G.



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:-

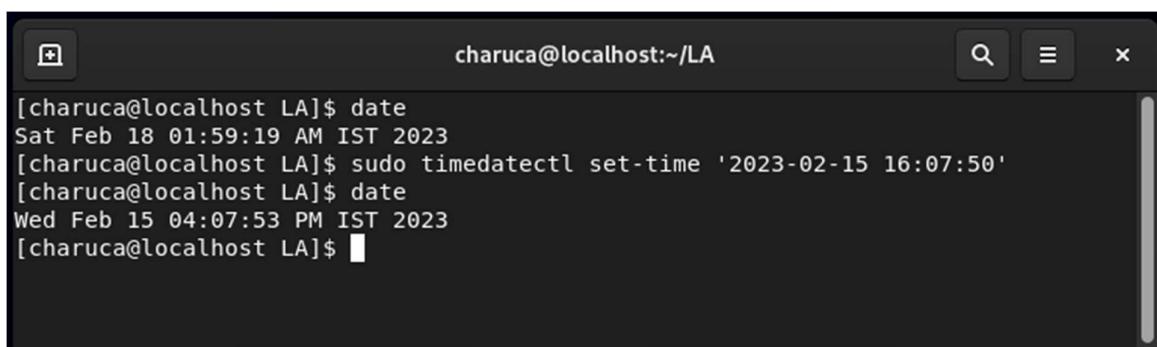
Configure date and time in Linux.

⊕ Source Code:-

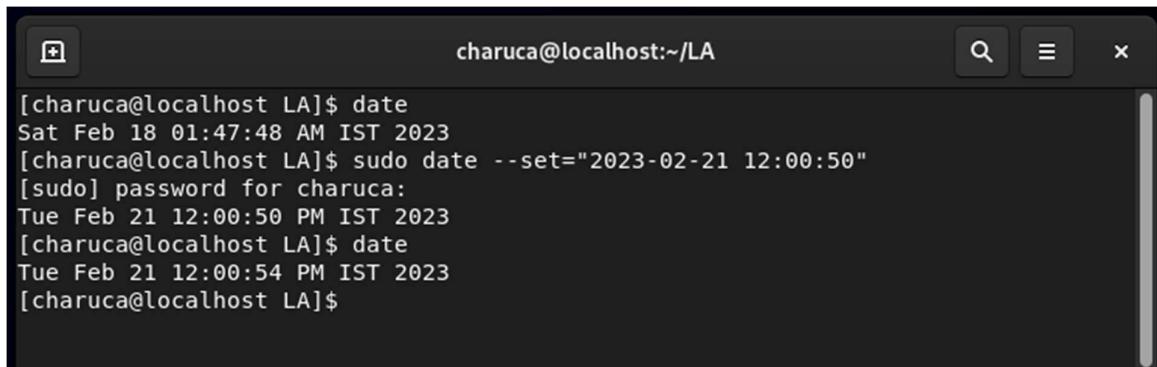
Date and Time can be configured in multiple ways:-

- `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"`

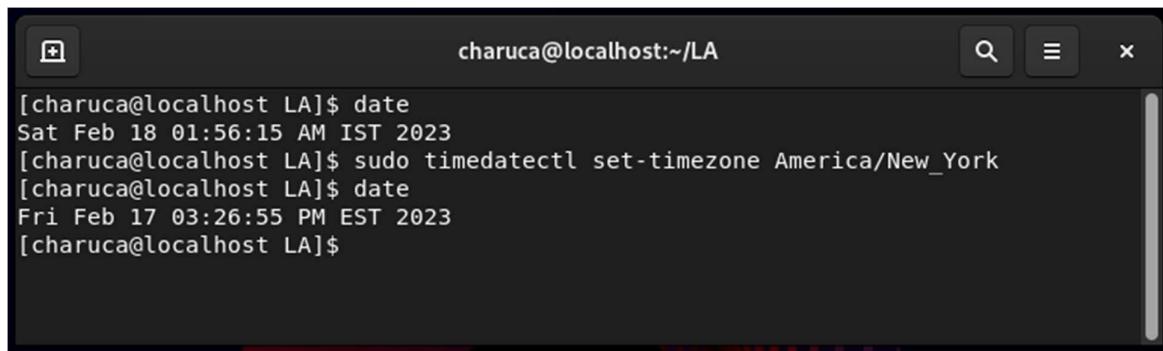
⊕ Output:-



```
charuca@localhost:~/LA
[charuca@localhost LA]$ date
Sat Feb 18 01:59:19 AM IST 2023
[charuca@localhost LA]$ sudo timedatectl set-time '2023-02-15 16:07:50'
[charuca@localhost LA]$ date
Wed Feb 15 04:07:53 PM IST 2023
[charuca@localhost LA]$
```



```
charuca@localhost:~/LA
[charuca@localhost LA]$ date
Sat Feb 18 01:47:48 AM IST 2023
[charuca@localhost LA]$ sudo date --set="2023-02-21 12:00:50"
[sudo] password for charuca:
Tue Feb 21 12:00:50 PM IST 2023
[charuca@localhost LA]$ date
Tue Feb 21 12:00:54 PM IST 2023
[charuca@localhost LA]$
```



A screenshot of a terminal window titled "charuca@localhost:~/LA". The window contains the following text:

```
[charuca@localhost LA]$ date  
Sat Feb 18 01:56:15 AM IST 2023  
[charuca@localhost LA]$ sudo timedatectl set-timezone America/New_York  
[charuca@localhost LA]$ date  
Fri Feb 17 03:26:55 PM EST 2023  
[charuca@localhost LA]$
```

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:-

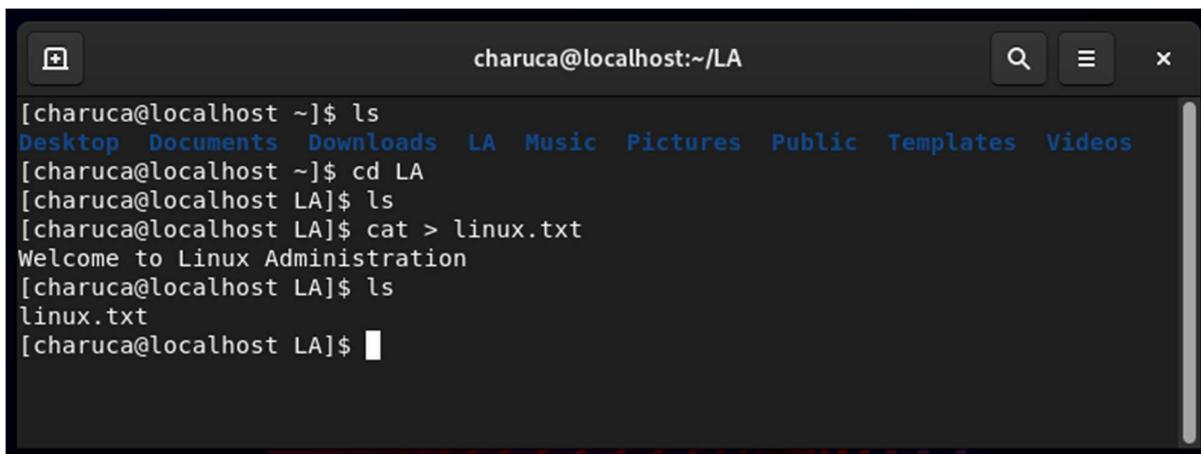
a. Regular files

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
```



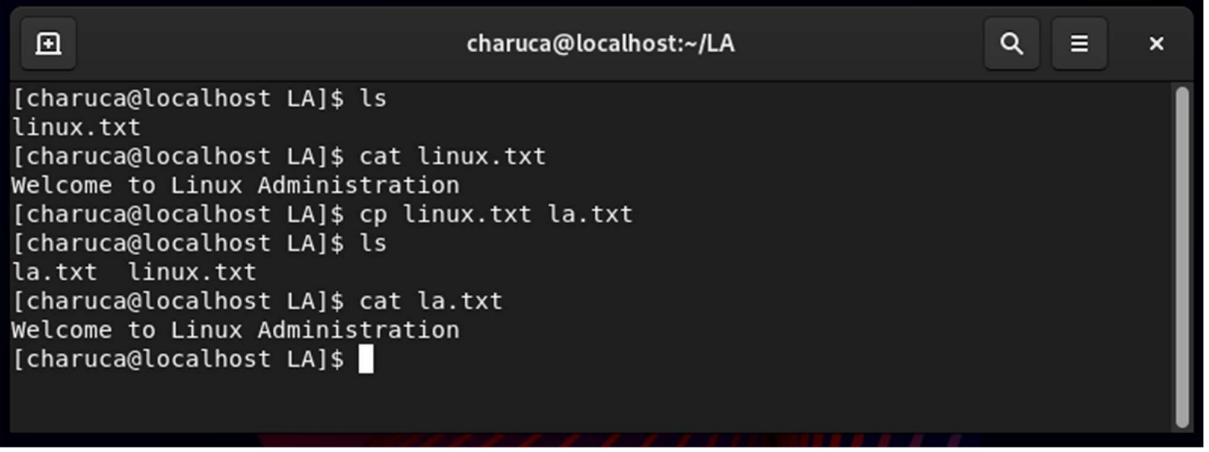
The screenshot shows a terminal window with a dark theme. The title bar reads "charuca@localhost:~/LA". The terminal content is as follows:

```
[charuca@localhost ~]$ ls
Desktop Documents Downloads LA Music Pictures Public Templates Videos
[charuca@localhost ~]$ cd LA
[charuca@localhost LA]$ ls
[charuca@localhost LA]$ cat > linux.txt
Welcome to Linux Administration
[charuca@localhost LA]$ ls
linux.txt
[charuca@localhost LA]$
```

Copying a file:-

Syntax: **cp [options] source_file destination_file**

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

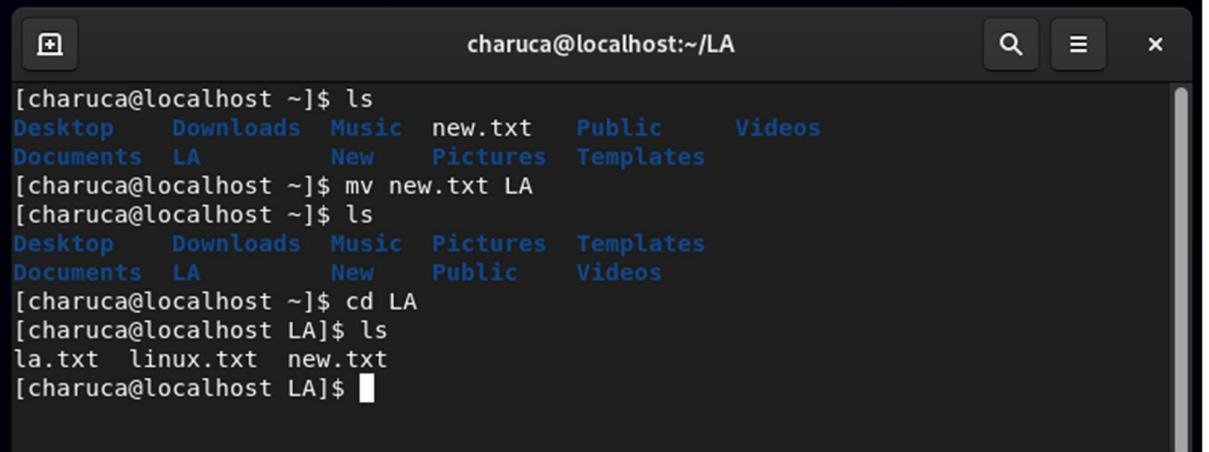


```
[charuca@localhost LA]$ ls  
linux.txt  
[charuca@localhost LA]$ cat linux.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$ cp linux.txt la.txt  
[charuca@localhost LA]$ ls  
la.txt linux.txt  
[charuca@localhost LA]$ cat la.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$
```

Moving a file:-

Syntax: **mv [options] filename destination**

```
mv new.txt LA
```



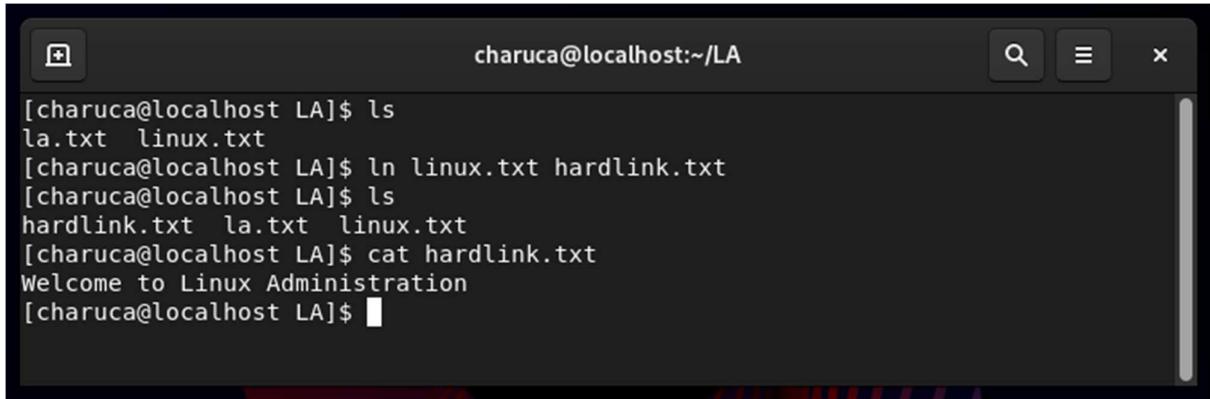
```
[charuca@localhost ~]$ ls  
Desktop Downloads Music new.txt Public Videos  
Documents LA New Pictures Templates  
[charuca@localhost ~]$ mv new.txt LA  
[charuca@localhost ~]$ ls  
Desktop Downloads Music Pictures Templates  
Documents LA New Public Videos  
[charuca@localhost ~]$ cd LA  
[charuca@localhost LA]$ ls  
la.txt linux.txt new.txt  
[charuca@localhost LA]$
```

Linking a file:-

Hard Link:-

Syntax: **ln source_file link_file**

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



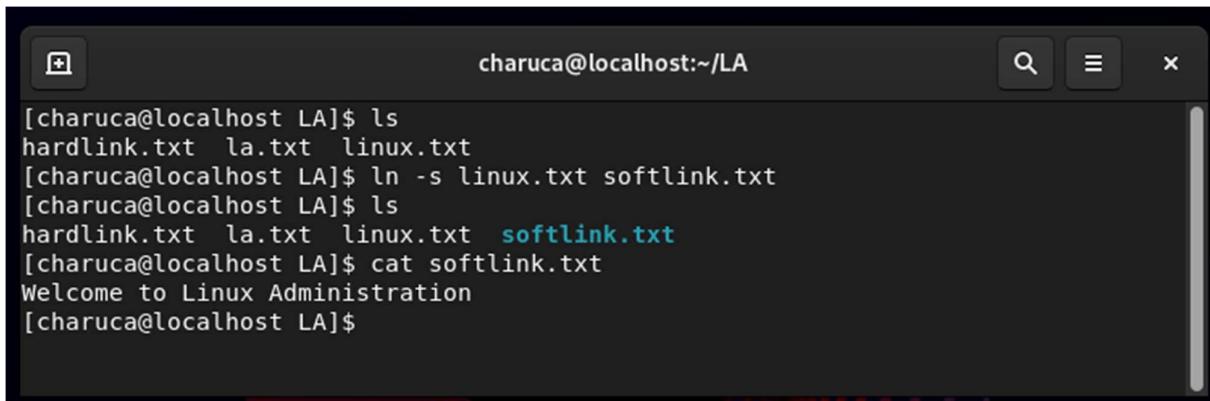
A terminal window titled "charuca@localhost:~/LA". The session shows the creation of a hard link named "hardlink.txt" to the file "linux.txt". The user then lists the files in the directory, which now includes both "linux.txt" and "hardlink.txt". Finally, the user reads the content of "hardlink.txt", which is identical to "linux.txt".

```
[charuca@localhost LA]$ ls  
la.txt linux.txt  
[charuca@localhost LA]$ ln linux.txt hardlink.txt  
[charuca@localhost LA]$ ls  
hardlink.txt la.txt linux.txt  
[charuca@localhost LA]$ cat hardlink.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$
```

Soft Link:-

Syntax: **ln -s source_file link_file**

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



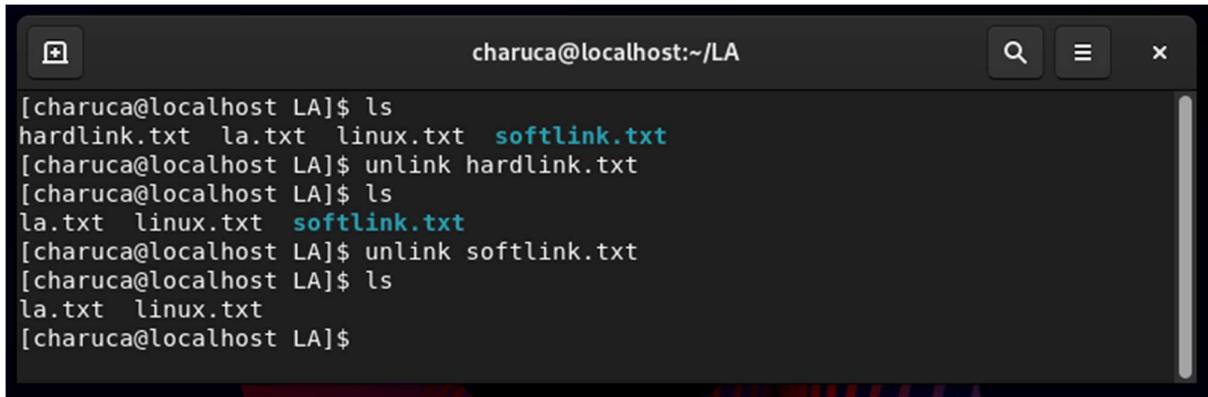
A terminal window titled "charuca@localhost:~/LA". The session shows the creation of a soft link named "softlink.txt" to the file "linux.txt". The user then lists the files in the directory, which now includes "hardlink.txt", "la.txt", "linux.txt", and "softlink.txt". The user reads the content of "softlink.txt", which is identical to "linux.txt".

```
[charuca@localhost LA]$ ls  
hardlink.txt la.txt linux.txt  
[charuca@localhost LA]$ ln -s linux.txt softlink.txt  
[charuca@localhost LA]$ ls  
hardlink.txt la.txt linux.txt softlink.txt  
[charuca@localhost LA]$ cat softlink.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$
```

Unlinking a file:-

Syntax: unlink filename

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



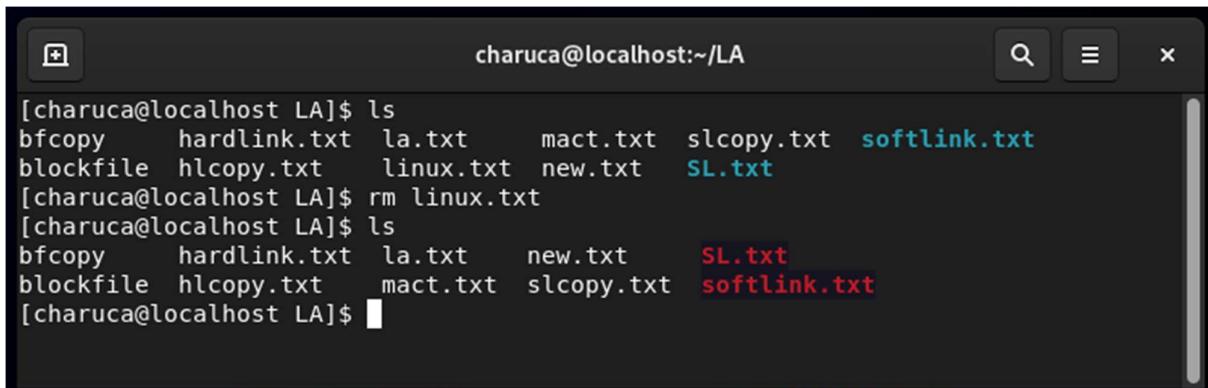
The screenshot shows a terminal window titled "charuca@localhost:~/LA". It displays the following command sequence:

```
[charuca@localhost LA]$ ls  
hardlink.txt la.txt linux.txt softlink.txt  
[charuca@localhost LA]$ unlink hardlink.txt  
[charuca@localhost LA]$ ls  
la.txt linux.txt softlink.txt  
[charuca@localhost LA]$ unlink softlink.txt  
[charuca@localhost LA]$ ls  
la.txt linux.txt  
[charuca@localhost LA]$
```

Removing a file:-

Syntax: rm [options] filename

```
rm linux.txt
```



The screenshot shows a terminal window titled "charuca@localhost:~/LA". It displays the following command sequence:

```
[charuca@localhost LA]$ ls  
bfcopy hardlink.txt la.txt mact.txt slcopy.txt softlink.txt  
blockfile hlcopyst.txt linux.txt new.txt SL.txt  
[charuca@localhost LA]$ rm linux.txt  
[charuca@localhost LA]$ ls  
bfcopy hardlink.txt la.txt new.txt SL.txt  
blockfile hlcopyst.txt mact.txt slcopy.txt softlink.txt  
[charuca@localhost LA]$
```

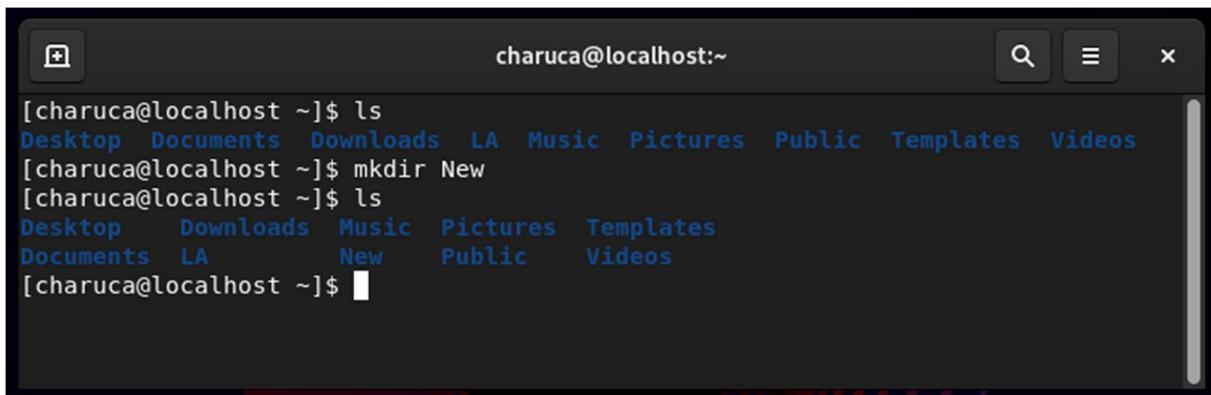
b. Directory files

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
```



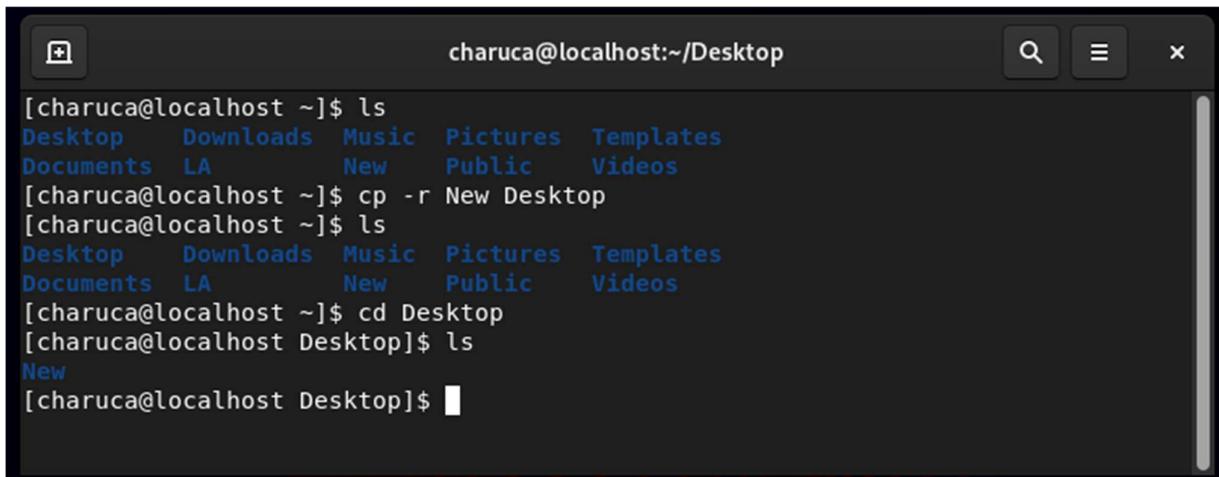
The screenshot shows a terminal window with a dark background and light-colored text. The title bar says "charuca@localhost:~". The command entered was "mkdir New", which creates a new directory named "New" in the current working directory. The terminal then lists the contents of the directory again, showing the newly created "New" folder.

```
[charuca@localhost ~]$ ls
Desktop Documents Downloads LA Music Pictures Public Templates Videos
[charuca@localhost ~]$ mkdir New
[charuca@localhost ~]$ ls
Desktop Downloads Music Pictures Templates
Documents LA New Public Videos
[charuca@localhost ~]$
```

Copying a directory:-

Syntax: cp [options] source_directory destination_directory

```
cp -r New Desktop
```



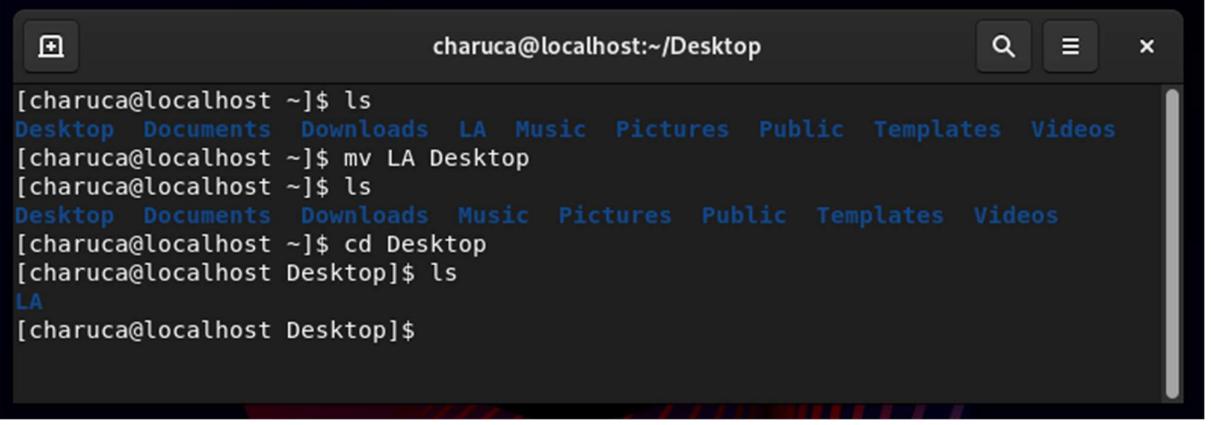
The screenshot shows a terminal window with a dark background and light-colored text. The title bar says "charuca@localhost:~/Desktop". The command entered was "cp -r New Desktop", which copies the "New" directory from the current working directory to the "Desktop" directory. The terminal then lists the contents of the "Desktop" directory, showing the copied "New" folder.

```
[charuca@localhost ~]$ ls
Desktop Downloads Music Pictures Templates
Documents LA New Public Videos
[charuca@localhost ~]$ cp -r New Desktop
[charuca@localhost ~]$ ls
Desktop Downloads Music Pictures Templates
Documents LA New Public Videos
[charuca@localhost ~]$ cd Desktop
[charuca@localhost Desktop]$ ls
New
[charuca@localhost Desktop]$
```

Moving a directory:-

Syntax: mv [options] directory destination

```
mv LA Desktop
```

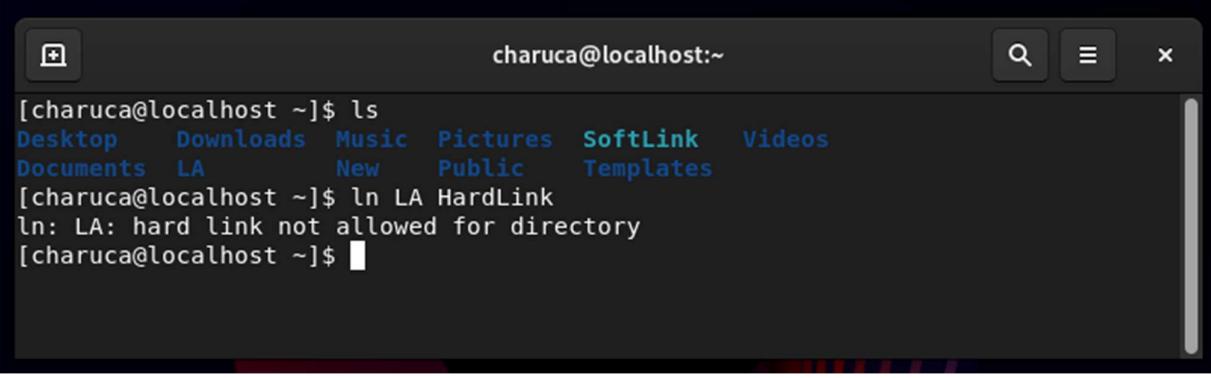


```
charuca@localhost:~/Desktop
[charuca@localhost ~]$ ls
Desktop Documents Downloads LA Music Pictures Public Templates Videos
[charuca@localhost ~]$ mv LA Desktop
[charuca@localhost ~]$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos
[charuca@localhost ~]$ cd Desktop
[charuca@localhost Desktop]$ ls
LA
[charuca@localhost Desktop]$
```

Linking a directory:-

Hard Link:-

A hard link is not allowed for a directory.

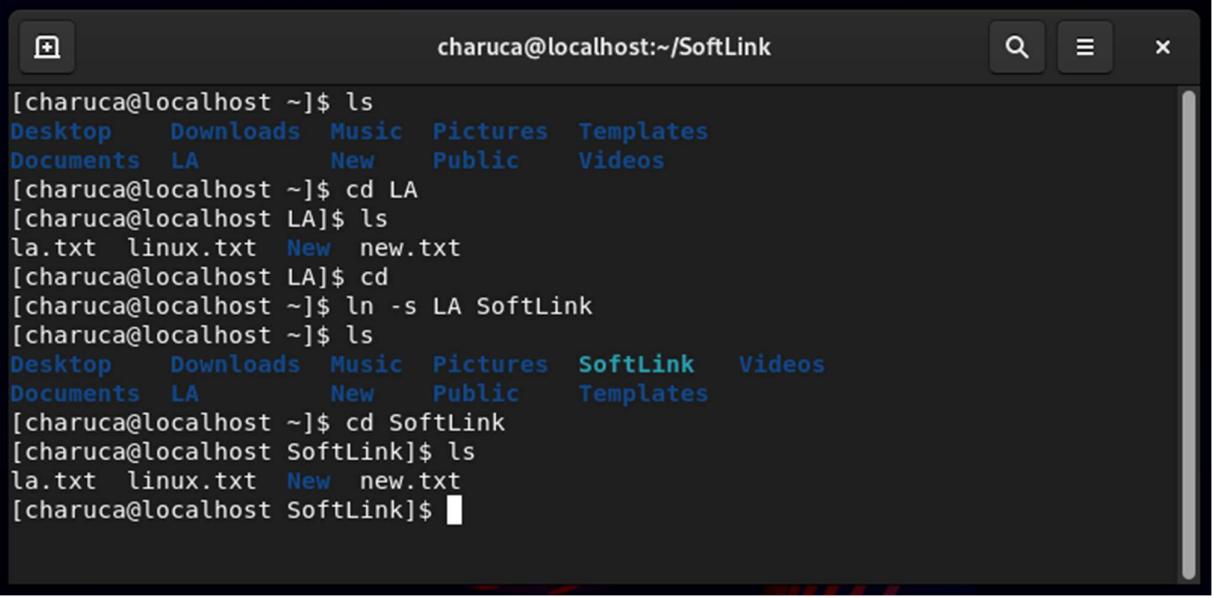


```
charuca@localhost:~
[charuca@localhost ~]$ ls
Desktop Downloads Music Pictures SoftLink Videos
Documents LA New Public Templates
[charuca@localhost ~]$ ln LA HardLink
ln: LA: hard link not allowed for directory
[charuca@localhost ~]$
```

Soft Link:-

Syntax: **ln -s source_directory link_directory**

ln -s LA SoftLink

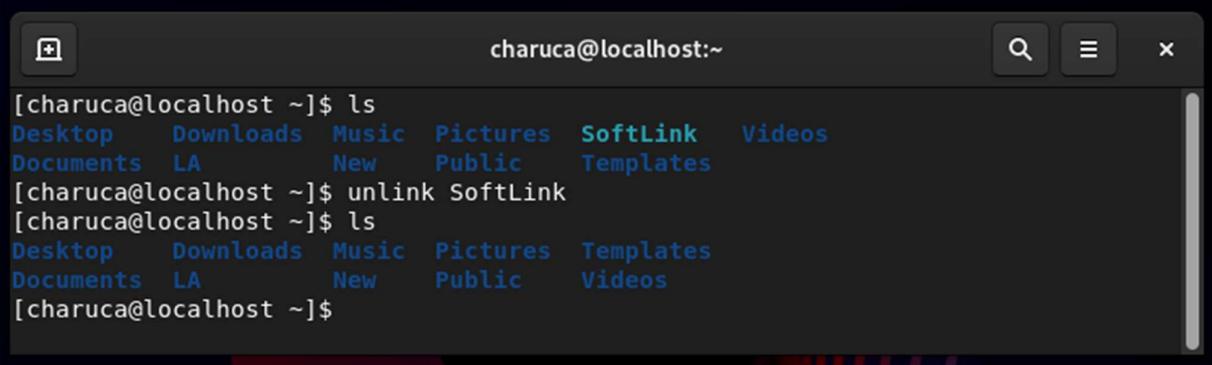


```
charuca@localhost:~/SoftLink
[charuca@localhost ~]$ ls
Desktop  Downloads  Music  Pictures  Templates
Documents  LA      New    Public    Videos
[charuca@localhost ~]$ cd LA
[charuca@localhost LA]$ ls
la.txt  linux.txt  New  new.txt
[charuca@localhost LA]$ cd
[charuca@localhost ~]$ ln -s LA SoftLink
[charuca@localhost ~]$ ls
Desktop  Downloads  Music  Pictures  SoftLink  Videos
Documents  LA      New    Public    Templates
[charuca@localhost ~]$ cd SoftLink
[charuca@localhost SoftLink]$ ls
la.txt  linux.txt  New  new.txt
[charuca@localhost SoftLink]$
```

Unlinking a directory:-

Syntax: **unlink directory**

unlink SoftLink

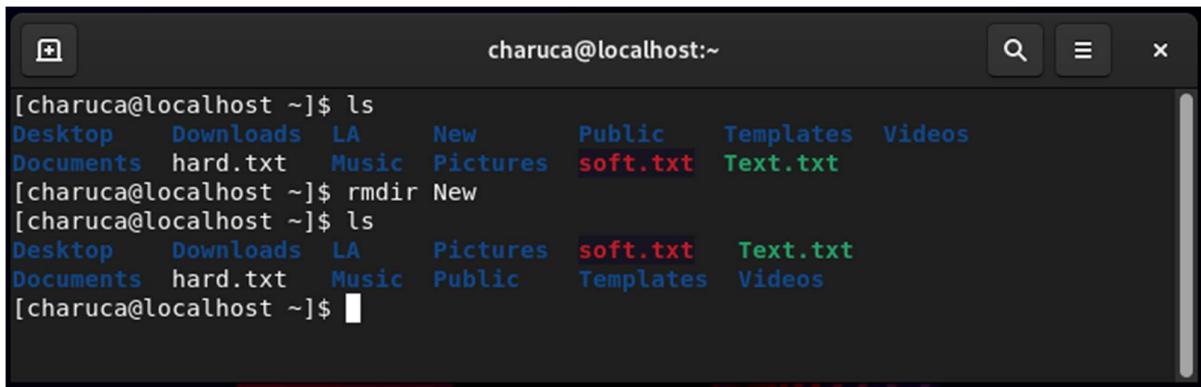


```
charuca@localhost:~
[charuca@localhost ~]$ ls
Desktop  Downloads  Music  Pictures  SoftLink  Videos
Documents  LA      New    Public    Templates
[charuca@localhost ~]$ unlink SoftLink
[charuca@localhost ~]$ ls
Desktop  Downloads  Music  Pictures  Templates
Documents  LA      New    Public    Videos
[charuca@localhost ~]$
```

Removing a directory:-

Syntax: rmdir directory

rmdir New



The screenshot shows a terminal window titled "charuca@localhost:~". The user runs the command "ls" to list files, showing "Desktop", "Downloads", "LA", "New", "Public", "Templates", and "Videos". Then, the user runs "rmdir New", which removes the directory "New". Finally, the user runs "ls" again, and the directory "New" is no longer listed, while the other files remain.

```
[charuca@localhost ~]$ ls
Desktop  Downloads  LA      New      Public   Templates  Videos
Documents hard.txt  Music   Pictures soft.txt  Text.txt
[charuca@localhost ~]$ rmdir New
[charuca@localhost ~]$ ls
Desktop  Downloads  LA      Pictures  soft.txt  Text.txt
Documents hard.txt  Music   Public    Templates  Videos
[charuca@localhost ~]$
```

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.

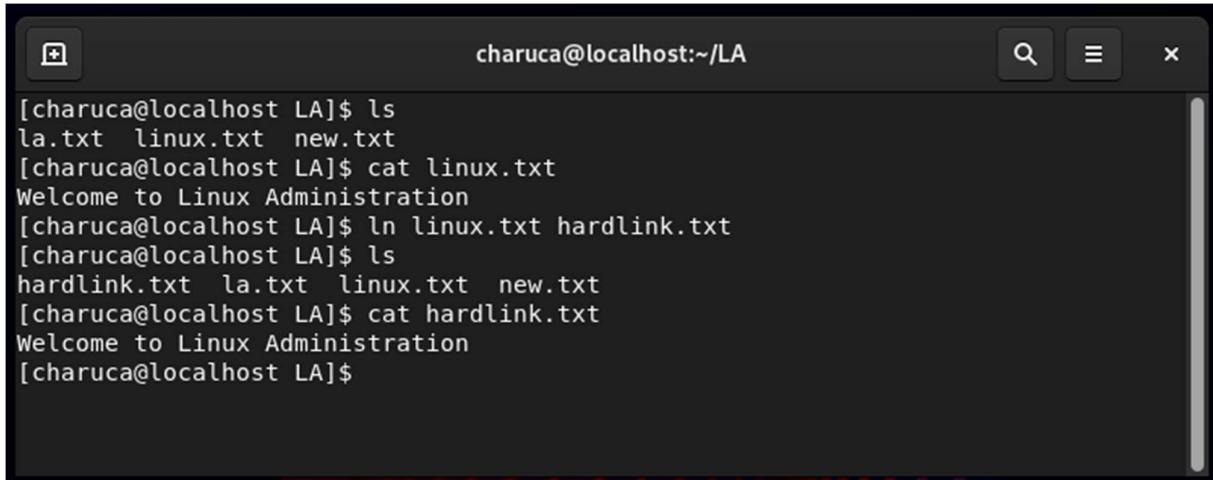
- Hard links: A hard link is a direct reference to a file's inode (i.e. the data structure that stores information about the file).
- Soft links: A soft link (or "symbolic link") is a reference to the path of a file or directory. It is essentially a special type of file that contains the path to the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

Creating a link file:-

Hard link:-

Syntax: **ln source_file link_file**

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



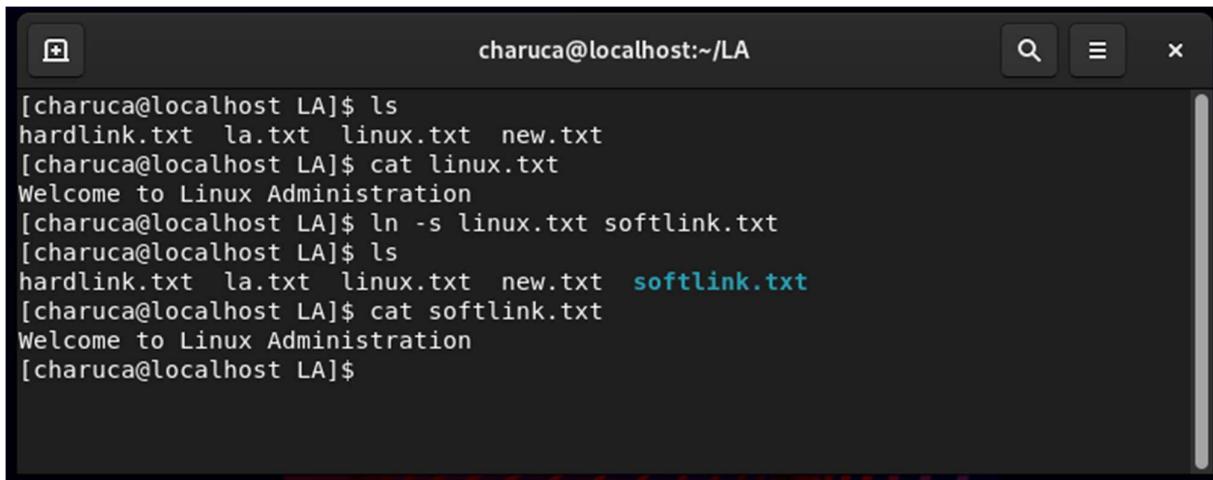
A terminal window titled "charuca@localhost:~/LA". The session shows the creation of a hard link named "hardlink.txt" pointing to the file "linux.txt". The terminal output is as follows:

```
[charuca@localhost LA]$ ls  
la.txt linux.txt new.txt  
[charuca@localhost LA]$ cat linux.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$ ln linux.txt hardlink.txt  
[charuca@localhost LA]$ ls  
hardlink.txt la.txt linux.txt new.txt  
[charuca@localhost LA]$ cat hardlink.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$
```

Soft link:-

Syntax: **ln -s source_file link_file**

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



A terminal window titled "charuca@localhost:~/LA". The session shows the creation of a soft link named "softlink.txt" pointing to the file "linux.txt". The terminal output is as follows:

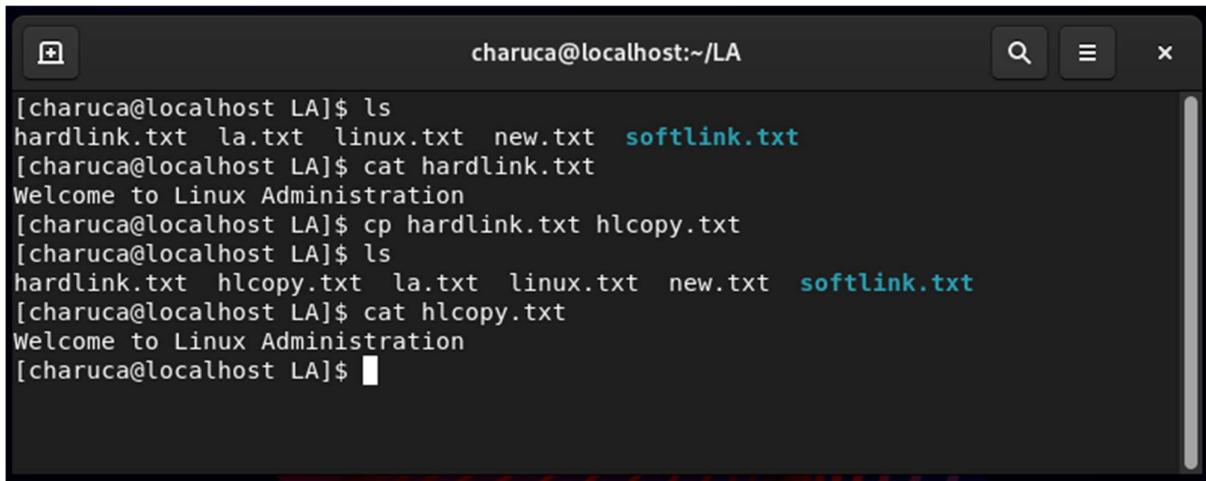
```
[charuca@localhost LA]$ ls  
hardlink.txt la.txt linux.txt new.txt  
[charuca@localhost LA]$ cat linux.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$ ln -s linux.txt softlink.txt  
[charuca@localhost LA]$ ls  
hardlink.txt la.txt linux.txt new.txt softlink.txt  
[charuca@localhost LA]$ cat softlink.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$
```

Copying a link file:-

Syntax: cp [options] source_linkfile destination_linkfile

Hard link:-

```
cp hardlink.txt hlcop.y.txt
```

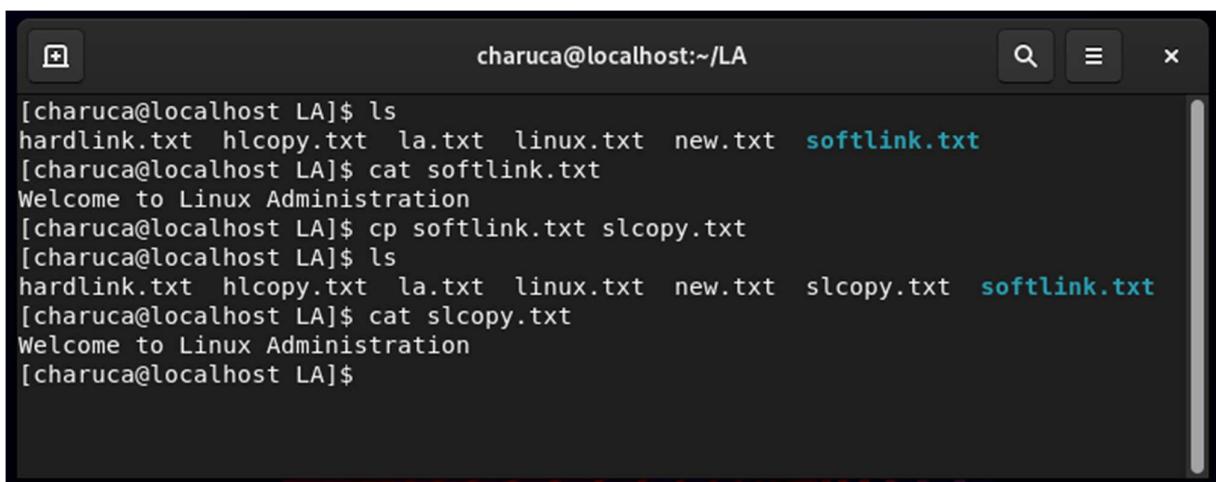


A terminal window titled "charuca@localhost:~/LA". The session shows the creation of a hard link named "hlcop.y.txt" from the original file "hardlink.txt". The terminal output is as follows:

```
[charuca@localhost LA]$ ls  
hardlink.txt la.txt linux.txt new.txt softlink.txt  
[charuca@localhost LA]$ cat hardlink.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$ cp hardlink.txt hlcop.y.txt  
[charuca@localhost LA]$ ls  
hardlink.txt hlcop.y.txt la.txt linux.txt new.txt softlink.txt  
[charuca@localhost LA]$ cat hlcop.y.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$
```

Soft link:-

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



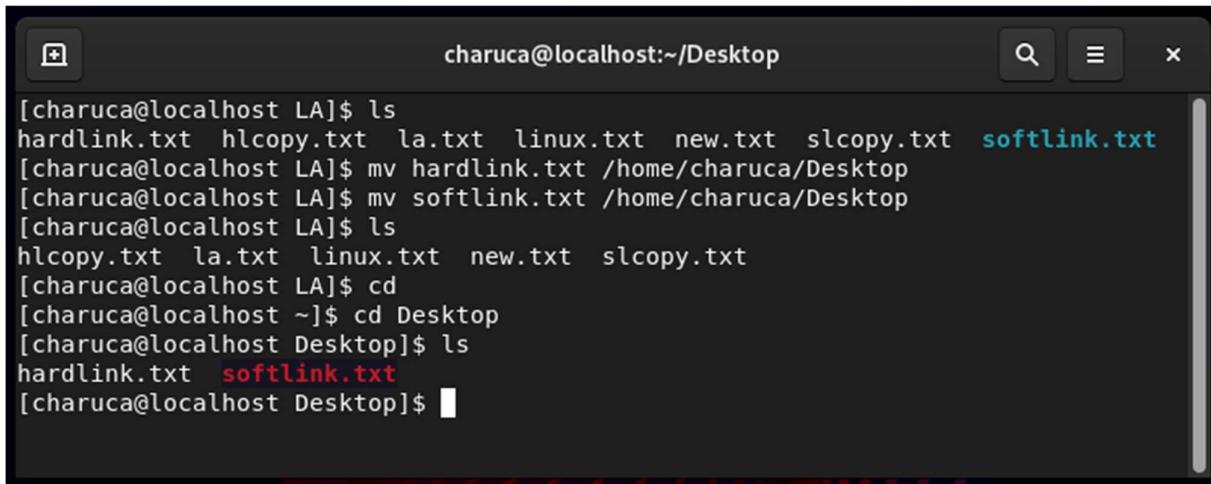
A terminal window titled "charuca@localhost:~/LA". The session shows the creation of a soft link named "slcopy.txt" from the original file "softlink.txt". The terminal output is as follows:

```
[charuca@localhost LA]$ ls  
hardlink.txt hlcop.y.txt la.txt linux.txt new.txt softlink.txt  
[charuca@localhost LA]$ cat softlink.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$ cp softlink.txt slcopy.txt  
[charuca@localhost LA]$ ls  
hardlink.txt hlcop.y.txt la.txt linux.txt new.txt slcopy.txt softlink.txt  
[charuca@localhost LA]$ cat slcopy.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$
```

Moving a link file:-

Syntax: mv [options] linkfile destination

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



A terminal window titled "charuca@localhost:~/Desktop". The session shows the user moving two files: "hardlink.txt" and "softlink.txt" from the current directory to the "/home/charuca/Desktop" directory. After the moves, the user runs "ls" to list the contents of the directory, which now only contains "hlcopy.txt", "la.txt", "linux.txt", "new.txt", and "slcopy.txt". The "softlink.txt" file is shown in red text, indicating it is no longer a valid link.

```
[charuca@localhost LA]$ ls  
hardlink.txt hlcopy.txt la.txt linux.txt new.txt slcopy.txt softlink.txt  
[charuca@localhost LA]$ mv hardlink.txt /home/charuca/Desktop  
[charuca@localhost LA]$ mv softlink.txt /home/charuca/Desktop  
[charuca@localhost LA]$ ls  
hlcopy.txt la.txt linux.txt new.txt slcopy.txt  
[charuca@localhost LA]$ cd  
[charuca@localhost ~]$ cd Desktop  
[charuca@localhost Desktop]$ ls  
hardlink.txt softlink.txt  
[charuca@localhost Desktop]$
```

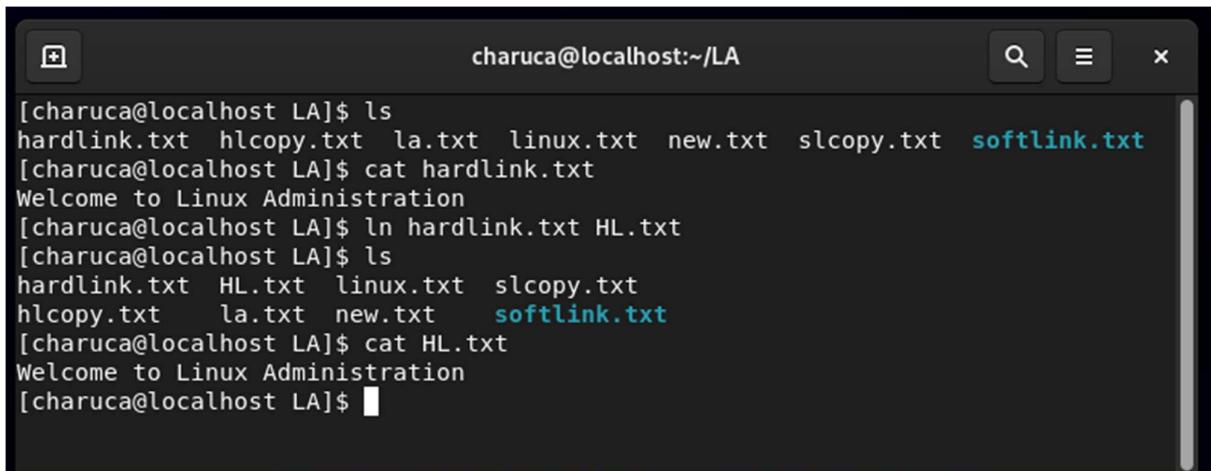
A soft link cannot be used after it is moved because its target file doesn't exist.

Linking a link file:-

Hard Link:-

Syntax: ln source_linkfile link_linkfile

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



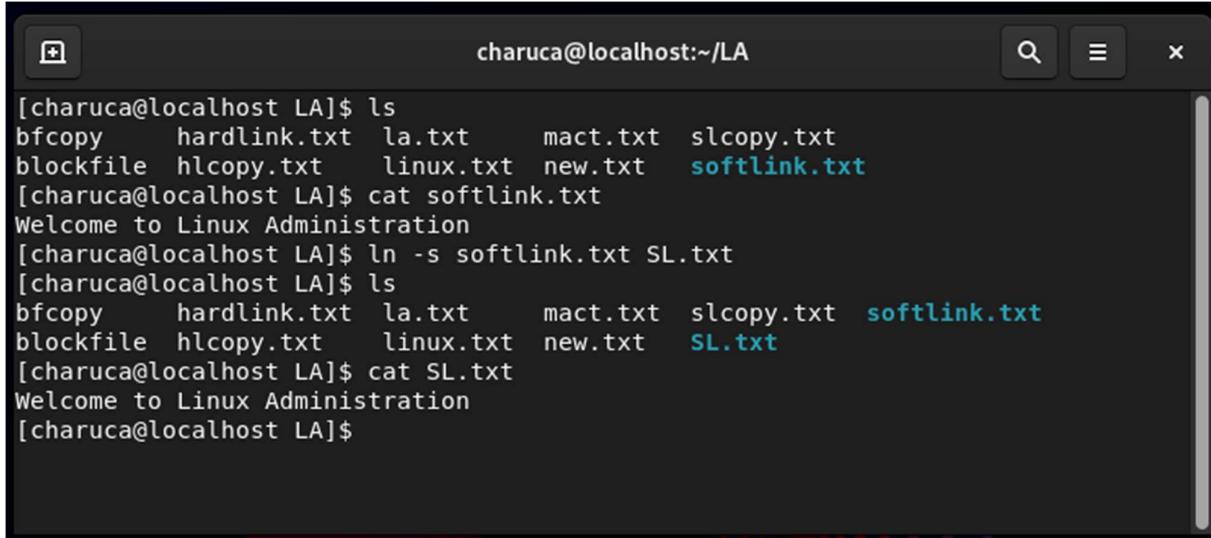
A terminal window titled "charuca@localhost:~/LA". The session shows the user creating a hard link named "HL.txt" to the existing "hardlink.txt" file. The user then lists the contents of the directory, showing both "hardlink.txt" and "HL.txt". The user also runs "cat" on "HL.txt" to verify its content, which matches the original "hardlink.txt" file.

```
[charuca@localhost LA]$ ls  
hardlink.txt hlcopy.txt la.txt linux.txt new.txt slcopy.txt softlink.txt  
[charuca@localhost LA]$ cat hardlink.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$ ln hardlink.txt HL.txt  
[charuca@localhost LA]$ ls  
hardlink.txt HL.txt linux.txt slcopy.txt  
hlcopy.txt la.txt new.txt softlink.txt  
[charuca@localhost LA]$ cat HL.txt  
Welcome to Linux Administration  
[charuca@localhost LA]$
```

Soft Link:-

Syntax: ln -s source_linkfile link_linkfile

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



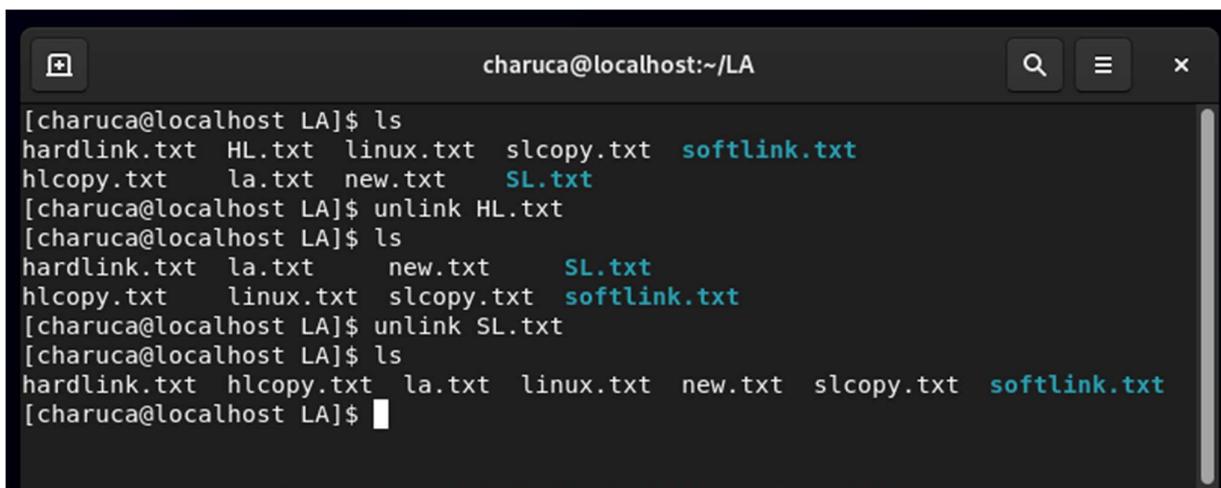
A terminal window titled "charuca@localhost:~/LA". The session shows the creation of a soft link named "SL.txt" pointing to "softlink.txt". The terminal output is as follows:

```
[charuca@localhost LA]$ ls
bfcopy    hardlink.txt  la.txt    mact.txt  slcopy.txt
blockfile  hlcopystxt  linux.txt new.txt   softlink.txt
[charuca@localhost LA]$ cat softlink.txt
Welcome to Linux Administration
[charuca@localhost LA]$ ln -s softlink.txt SL.txt
[charuca@localhost LA]$ ls
bfcopy    hardlink.txt  la.txt    mact.txt  slcopy.txt  softlink.txt
blockfile  hlcopystxt  linux.txt new.txt   SL.txt
[charuca@localhost LA]$ cat SL.txt
Welcome to Linux Administration
[charuca@localhost LA]$
```

Unlinking a link file:-

Syntax: unlink linkfile

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



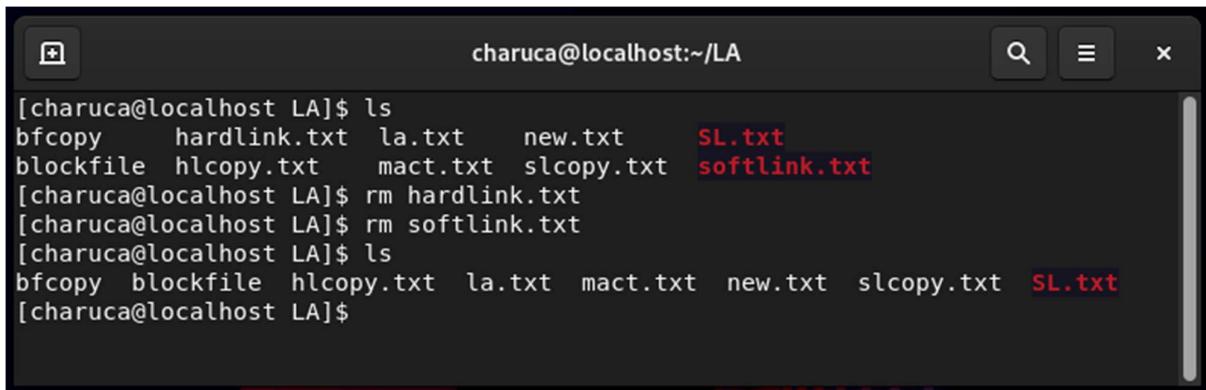
A terminal window titled "charuca@localhost:~/LA". The session shows the unlinking of two symbolic links, "HL.txt" and "SL.txt", which were previously created. The terminal output is as follows:

```
[charuca@localhost LA]$ ls
hardlink.txt  HL.txt  linux.txt  slcopy.txt  softlink.txt
hlcopystxt  la.txt  new.txt   SL.txt
[charuca@localhost LA]$ unlink HL.txt
[charuca@localhost LA]$ ls
hardlink.txt  la.txt  new.txt   SL.txt
hlcopystxt  linux.txt  slcopy.txt  softlink.txt
[charuca@localhost LA]$ unlink SL.txt
[charuca@localhost LA]$ ls
hardlink.txt  hlcopystxt  la.txt  linux.txt  new.txt  slcopy.txt  softlink.txt
[charuca@localhost LA]$
```

Removing a link file:-

Syntax: **rm [options] linkfile**

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



A terminal window titled "charuca@localhost:~/LA". The session shows the user listing files, removing "hardlink.txt" and "softlink.txt", and then listing files again to show they are gone.

```
[charuca@localhost LA]$ ls  
bfcopy  hardlink.txt  la.txt  new.txt  SL.txt  
blockfile  hlcopystyle="text-decoration: underline;">y.txt  mact.txt  slcopy.txt  softlink.txt  
[charuca@localhost LA]$ rm hardlink.txt  
[charuca@localhost LA]$ rm softlink.txt  
[charuca@localhost LA]$ ls  
bfcopy  blockfile  hlcopystyle="text-decoration: underline;">y.txt  la.txt  mact.txt  new.txt  slcopy.txt  SL.txt  
[charuca@localhost LA]$
```

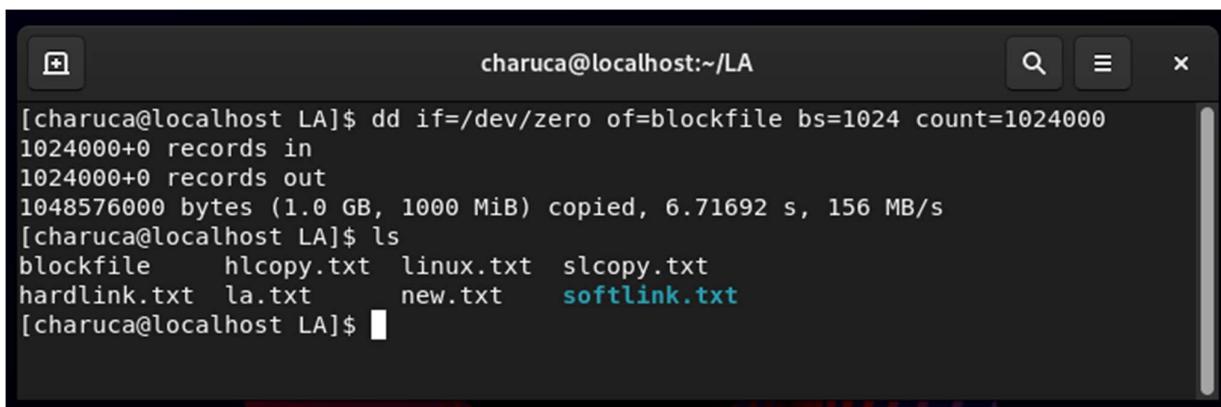
d. Block files

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
```



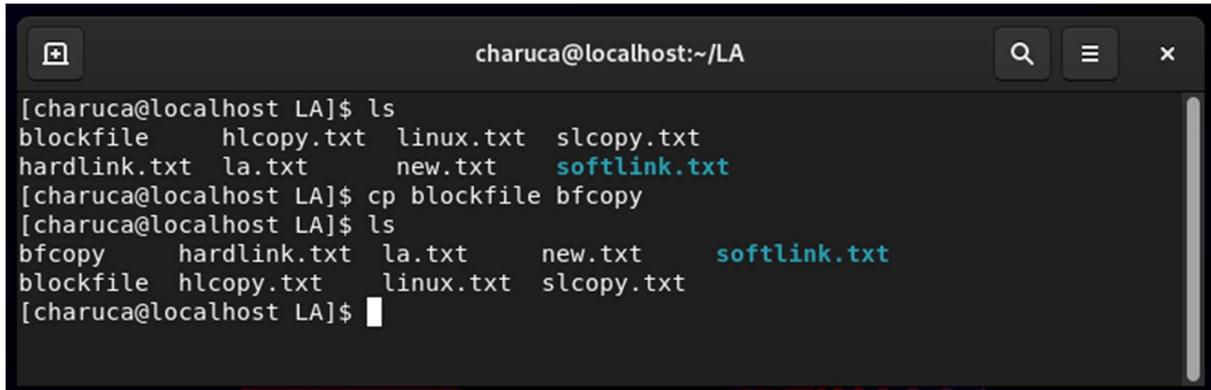
A terminal window titled "charuca@localhost:~/LA". The session shows the user running the dd command to create a block file, followed by a file listing.

```
[charuca@localhost LA]$ dd if=/dev/zero of=blockfile bs=1024 count=1024000  
1024000+0 records in  
1024000+0 records out  
1048576000 bytes (1.0 GB, 1000 MiB) copied, 6.71692 s, 156 MB/s  
[charuca@localhost LA]$ ls  
blockfile  hlcopystyle="text-decoration: underline;">y.txt  linux.txt  slcopy.txt  
hardlink.txt  la.txt  new.txt  softlink.txt  
[charuca@localhost LA]$
```

Copying a block file:-

Syntax: cp [options] source_blockfile destination_blockfile

```
cp blockfile bfcopy
```



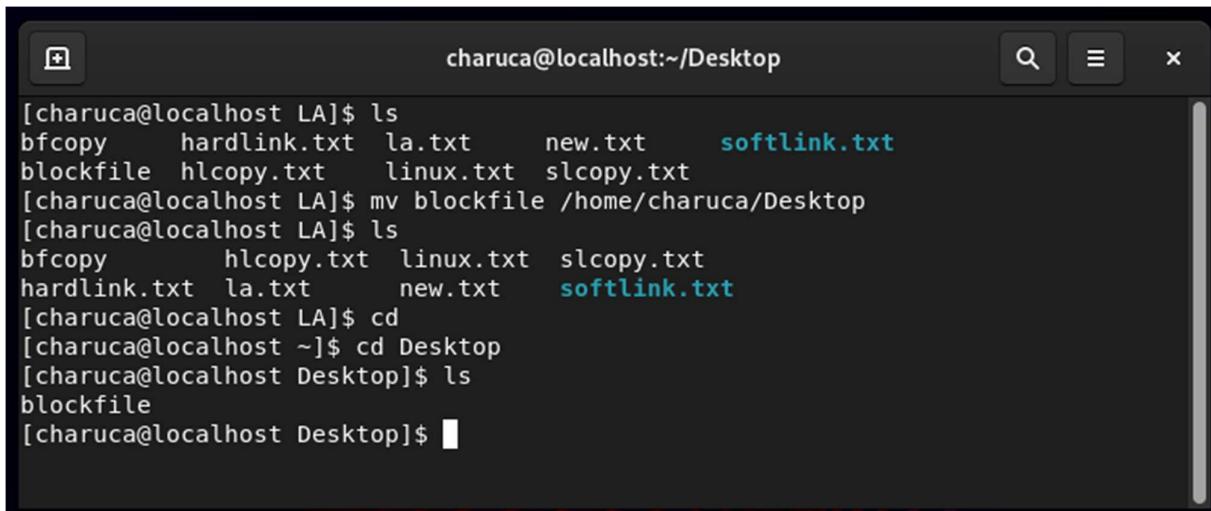
A terminal window titled "charuca@localhost:~/LA". The user runs the command "ls" to show files: blockfile, hlcop... (redacted), linux.txt, slcopy.txt, hardlink.txt, la.txt, new.txt, softlink.txt. Then, "cp blockfile bfcopy" is run. Finally, "ls" is run again, showing bfcopy, hardlink.txt, la.txt, new.txt, softlink.txt, blockfile, hlcop... (redacted), linux.txt, slcopy.txt. The "softlink.txt" file appears twice in the output.

```
[charuca@localhost LA]$ ls
blockfile    hlcop... (redacted)  linux.txt  slcopy.txt
hardlink.txt  la.txt      new.txt    softlink.txt
[charuca@localhost LA]$ cp blockfile bfcopy
[charuca@localhost LA]$ ls
bfcopy      hardlink.txt  la.txt      new.txt    softlink.txt
blockfile   hlcop... (redacted)  linux.txt  slcopy.txt
[charuca@localhost LA]$
```

Moving a block file:-

Syntax: mv [options] blockfile destination

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



A terminal window titled "charuca@localhost:~/Desktop". The user runs "ls" to show files: bfcopy, hardlink.txt, la.txt, new.txt, softlink.txt, blockfile, hlcop... (redacted), linux.txt, slcopy.txt. Then, "mv blockfile /home/charuca/Desktop" is run. Finally, "ls" is run again, showing bfcopy, hlcop... (redacted), linux.txt, slcopy.txt, hardlink.txt, la.txt, new.txt, softlink.txt. The "blockfile" file is no longer present in the current directory.

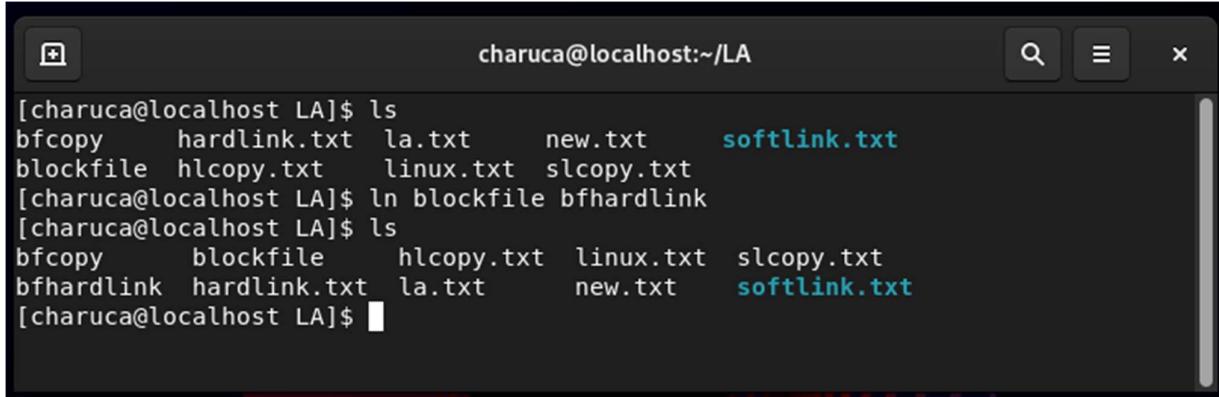
```
[charuca@localhost LA]$ ls
bfcopy      hardlink.txt  la.txt      new.txt    softlink.txt
blockfile   hlcop... (redacted)  linux.txt  slcopy.txt
[charuca@localhost LA]$ mv blockfile /home/charuca/Desktop
[charuca@localhost LA]$ ls
bfcopy      hlcop... (redacted)  linux.txt  slcopy.txt
hardlink.txt  la.txt      new.txt    softlink.txt
[charuca@localhost LA]$ cd
[charuca@localhost ~]$ cd Desktop
[charuca@localhost Desktop]$ ls
blockfile
[charuca@localhost Desktop]$
```

Linking a block file:-

Hard Link:-

Syntax: **ln blockfile block_linkfile**

```
ln blockfile bfhardlink
```

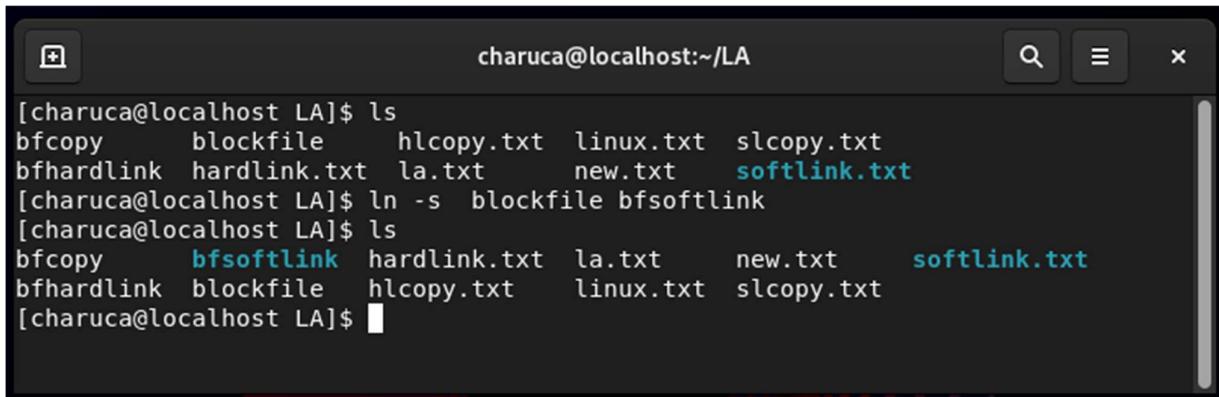


```
[charuca@localhost LA]$ ls  
bfcopy  hardlink.txt  la.txt  new.txt  softlink.txt  
blockfile  hlcopystxt  linux.txt  slcopy.txt  
[charuca@localhost LA]$ ln blockfile bfhardlink  
[charuca@localhost LA]$ ls  
bfcopy  blockfile  hlcopystxt  linux.txt  slcopy.txt  
bfhardlink  hardlink.txt  la.txt  new.txt  softlink.txt  
[charuca@localhost LA]$
```

Soft Link:-

Syntax: **ln -s blockfile block_linkfile**

```
ln -s blockfile bfsoftlink
```

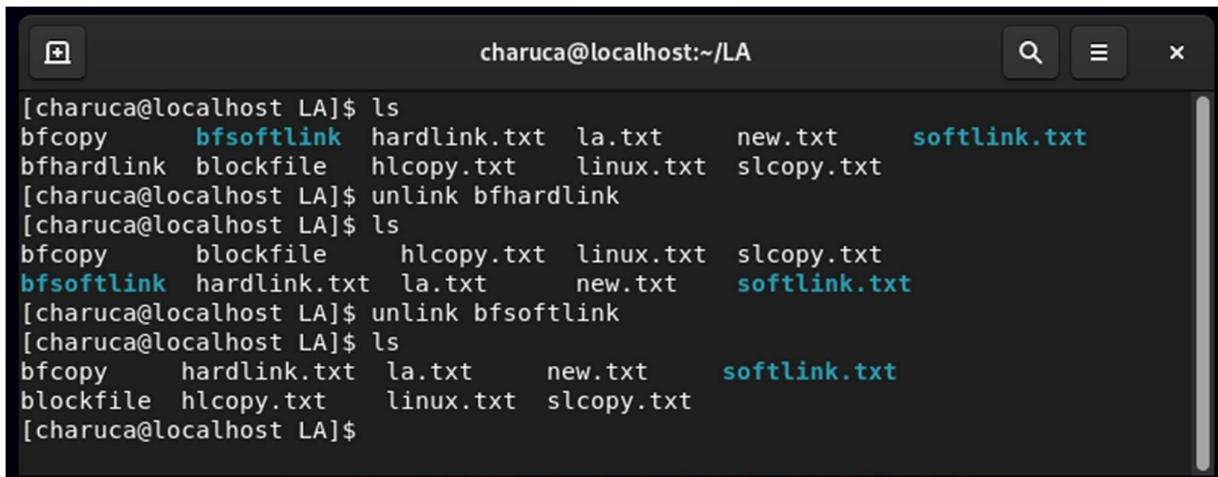


```
[charuca@localhost LA]$ ls  
bfcopy  blockfile  hlcopystxt  linux.txt  slcopy.txt  
bfhardlink  hardlink.txt  la.txt  new.txt  softlink.txt  
[charuca@localhost LA]$ ln -s blockfile bfsoftlink  
[charuca@localhost LA]$ ls  
bfcopy  bfsoftlink  hardlink.txt  la.txt  new.txt  softlink.txt  
bfhardlink  blockfile  hlcopystxt  linux.txt  slcopy.txt  
[charuca@localhost LA]$
```

unlinking a block file:-

Syntax: unlink blockfile

```
unlink bfhardlink  
unlink bfsoftlink
```



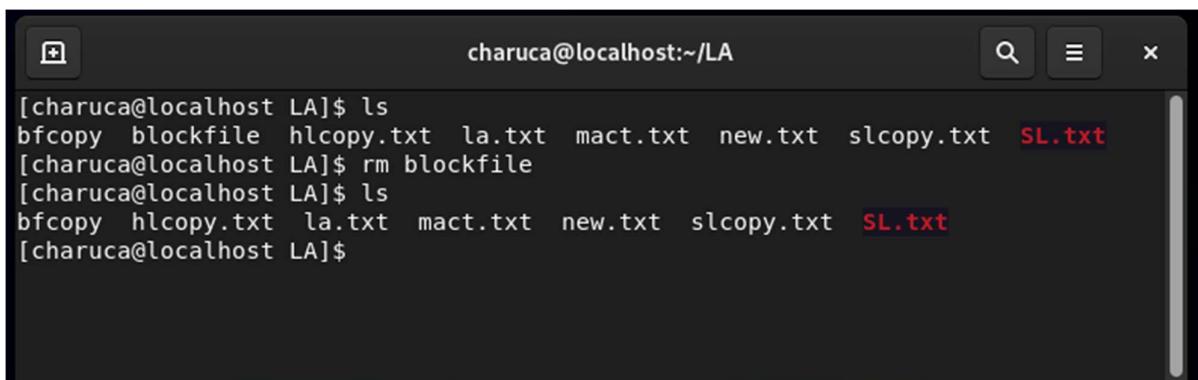
The terminal window shows the user's session on a local host. It starts with listing files in the current directory, then uses the 'unlink' command to remove a hard link ('bfhardlink'). After removing it, the user lists the files again to verify. Then, the user removes a soft link ('bfsoftlink'), lists the files again, and finally removes the remaining hard link ('bfhardlink') to show that all links have been removed.

```
[charuca@localhost LA]$ ls  
bfcopy      bfsoftlink  hardlink.txt  la.txt    new.txt    softlink.txt  
bfhardlink  blockfile   hlcopystxt   linux.txt  slcopy.txt  
[charuca@localhost LA]$ unlink bfhardlink  
[charuca@localhost LA]$ ls  
bfcopy      blockfile   hlcopystxt  linux.txt  slcopy.txt  
bfsoftlink  hardlink.txt  la.txt    new.txt    softlink.txt  
[charuca@localhost LA]$ unlink bfsoftlink  
[charuca@localhost LA]$ ls  
bfcopy      hardlink.txt  la.txt    new.txt    softlink.txt  
blockfile   hlcopystxt  linux.txt  slcopy.txt  
[charuca@localhost LA]$
```

Removing a block file:-

Syntax: rm [options] blockfile

```
rm blockfile
```



The terminal window shows the user's session on a local host. It starts by listing files, then uses the 'rm' command to remove a specific file ('blockfile'). After removing it, the user lists the files again to verify that the file is no longer present.

```
[charuca@localhost LA]$ ls  
bfcopy  blockfile  hlcopystxt  la.txt  mact.txt  new.txt  slcopy.txt  SL.txt  
[charuca@localhost LA]$ rm blockfile  
[charuca@localhost LA]$ ls  
bfcopy  hlcopystxt  la.txt  mact.txt  new.txt  slcopy.txt  SL.txt  
[charuca@localhost LA]$
```

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:-

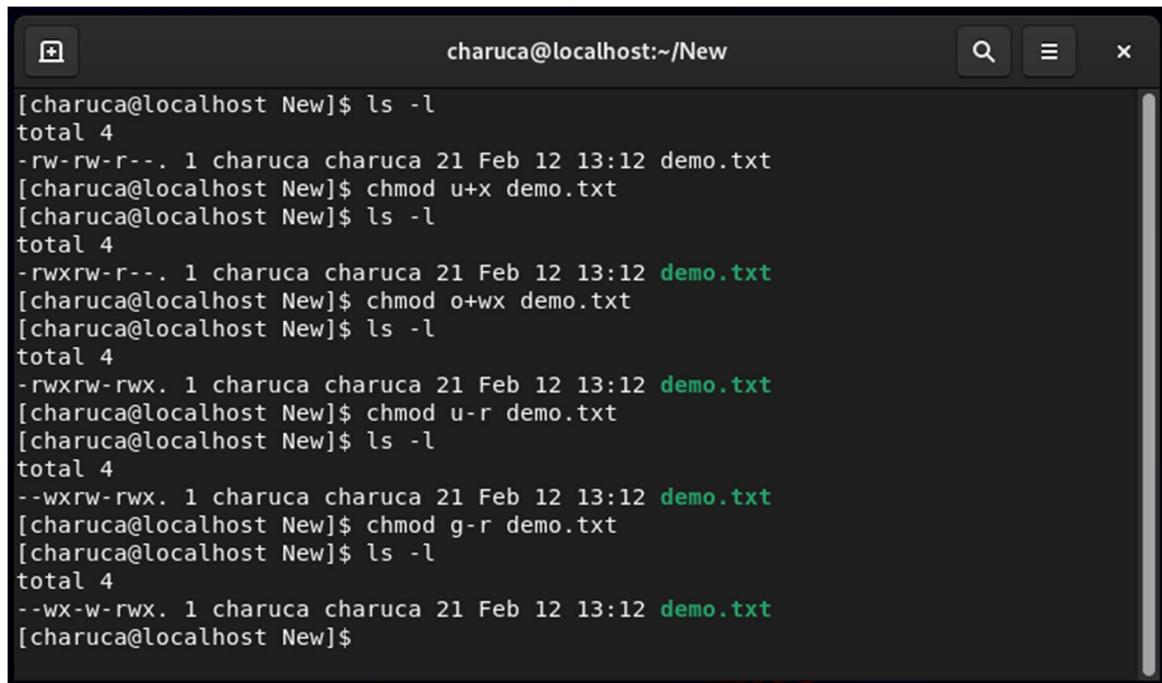
```
chmod u+x demo.txt  
chmod o+wx demo.txt  
chmod u-r demo.txt  
chmod g-r demo.txt
```

› Numeric Method:-

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

Output:-

› Absolute Method:-

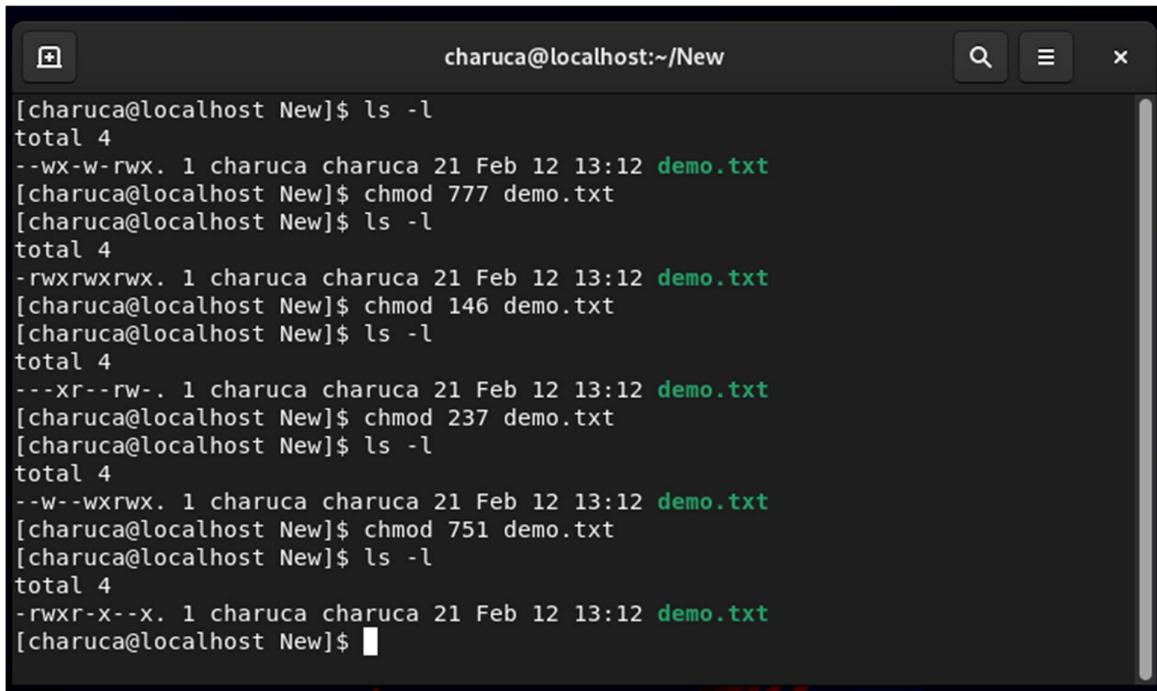


A terminal window titled "charuca@localhost:~/New" showing the execution of chmod commands on a file named "demo.txt". The terminal output is as follows:

```
[charuca@localhost New]$ ls -l  
total 4  
-rw-rw-r--. 1 charuca charuca 21 Feb 12 13:12 demo.txt  
[charuca@localhost New]$ chmod u+x demo.txt  
[charuca@localhost New]$ ls -l  
total 4  
-rwxrw-r--. 1 charuca charuca 21 Feb 12 13:12 demo.txt  
[charuca@localhost New]$ chmod o+wx demo.txt  
[charuca@localhost New]$ ls -l  
total 4  
-rwxrw-rwx. 1 charuca charuca 21 Feb 12 13:12 demo.txt  
[charuca@localhost New]$ chmod u-r demo.txt  
[charuca@localhost New]$ ls -l  
total 4  
--wxrw-rwx. 1 charuca charuca 21 Feb 12 13:12 demo.txt  
[charuca@localhost New]$ chmod g-r demo.txt  
[charuca@localhost New]$ ls -l  
total 4  
--wx-w-rwx. 1 charuca charuca 21 Feb 12 13:12 demo.txt  

```

› Numeric Method:-



The screenshot shows a terminal window titled "charuca@localhost:~/New". It displays a series of commands and their outputs related to file permissions:

```
[charuca@localhost New]$ ls -l
total 4
--wx-w-rwx. 1 charuca charuca 21 Feb 12 13:12 demo.txt
[charuca@localhost New]$ chmod 777 demo.txt
[charuca@localhost New]$ ls -l
total 4
-rwxrwxrwx. 1 charuca charuca 21 Feb 12 13:12 demo.txt
[charuca@localhost New]$ chmod 146 demo.txt
[charuca@localhost New]$ ls -l
total 4
---xr--rw-. 1 charuca charuca 21 Feb 12 13:12 demo.txt
[charuca@localhost New]$ chmod 237 demo.txt
[charuca@localhost New]$ ls -l
total 4
--w--wxrwx. 1 charuca charuca 21 Feb 12 13:12 demo.txt
[charuca@localhost New]$ chmod 751 demo.txt
[charuca@localhost New]$ ls -l
total 4
-rwxr-x---x. 1 charuca charuca 21 Feb 12 13:12 demo.txt
[charuca@localhost New]$
```

chown command

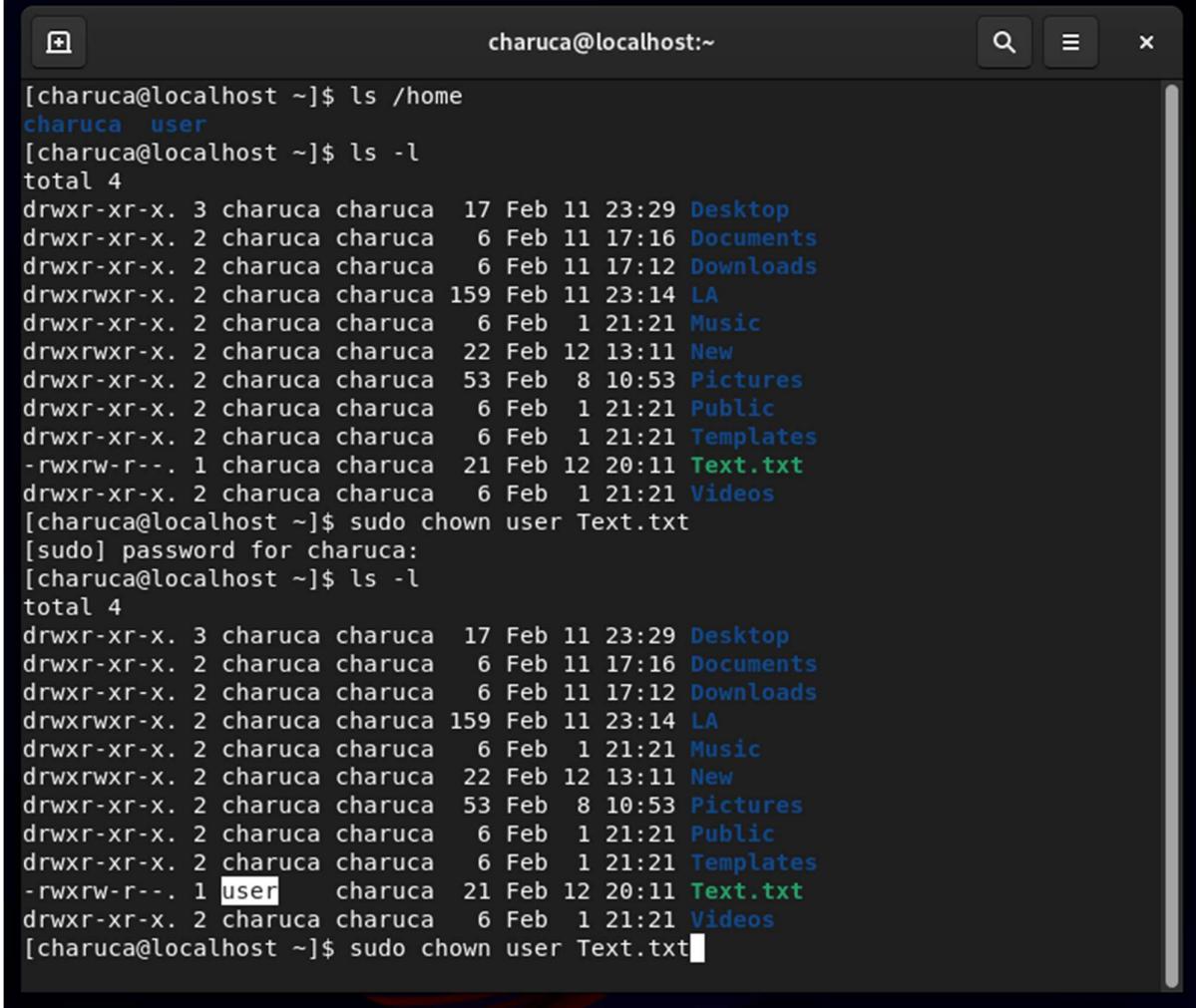
❖ **Theory:-**

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:-



The screenshot shows a terminal window titled "charuca@localhost:~". The terminal displays the following session:

```
[charuca@localhost ~]$ ls /home  
charuca user  
[charuca@localhost ~]$ ls -l  
total 4  
drwxr-xr-x. 3 charuca charuca 17 Feb 11 23:29 Desktop  
drwxr-xr-x. 2 charuca charuca 6 Feb 11 17:16 Documents  
drwxr-xr-x. 2 charuca charuca 6 Feb 11 17:12 Downloads  
drwxrwxr-x. 2 charuca charuca 159 Feb 11 23:14 LA  
drwxr-xr-x. 2 charuca charuca 6 Feb 1 21:21 Music  
drwxrwxr-x. 2 charuca charuca 22 Feb 12 13:11 New  
drwxr-xr-x. 2 charuca charuca 53 Feb 8 10:53 Pictures  
drwxr-xr-x. 2 charuca charuca 6 Feb 1 21:21 Public  
drwxr-xr-x. 2 charuca charuca 6 Feb 1 21:21 Templates  
-rwxrw-r--. 1 charuca charuca 21 Feb 12 20:11 Text.txt  
drwxr-xr-x. 2 charuca charuca 6 Feb 1 21:21 Videos  
[charuca@localhost ~]$ sudo chown user Text.txt  
[sudo] password for charuca:  
[charuca@localhost ~]$ ls -l  
total 4  
drwxr-xr-x. 3 charuca charuca 17 Feb 11 23:29 Desktop  
drwxr-xr-x. 2 charuca charuca 6 Feb 11 17:16 Documents  
drwxr-xr-x. 2 charuca charuca 6 Feb 11 17:12 Downloads  
drwxrwxr-x. 2 charuca charuca 159 Feb 11 23:14 LA  
drwxr-xr-x. 2 charuca charuca 6 Feb 1 21:21 Music  
drwxrwxr-x. 2 charuca charuca 22 Feb 12 13:11 New  
drwxr-xr-x. 2 charuca charuca 53 Feb 8 10:53 Pictures  
drwxr-xr-x. 2 charuca charuca 6 Feb 1 21:21 Public  
drwxr-xr-x. 2 charuca charuca 6 Feb 1 21:21 Templates  
-rwxrw-r--. 1 user charuca 21 Feb 12 20:11 Text.txt  
drwxr-xr-x. 2 charuca charuca 6 Feb 1 21:21 Videos  
[charuca@localhost ~]$ sudo chown user Text.txt
```

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:-

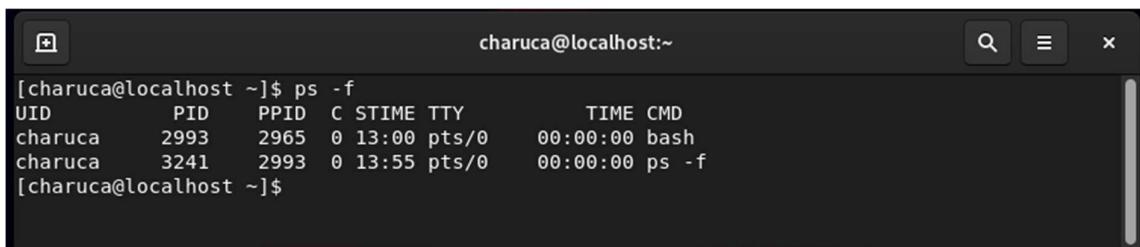
```
ps - ef
```

- Zombie process:-

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

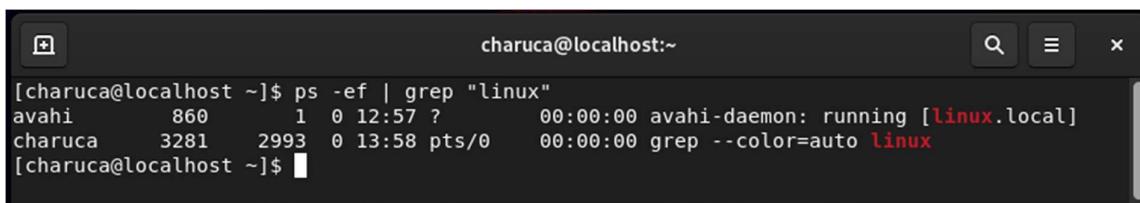
Output:-

- Foreground process:-



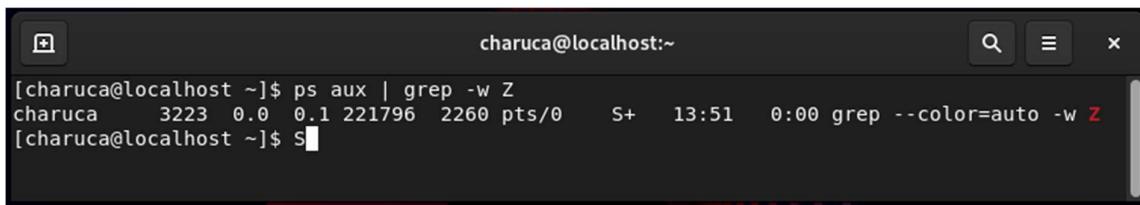
```
[charuca@localhost ~]$ ps -f
UID      PID  PPID  C STIME TTY      TIME CMD
charuca  2993  2965  0 13:00 pts/0    00:00:00 bash
charuca  3241  2993  0 13:55 pts/0    00:00:00 ps -f
[charuca@localhost ~]$
```

- Background process:-



```
[charuca@localhost ~]$ ps -ef | grep "linux"
avahi      860      1  0 12:57 ?    00:00:00 avahi-daemon: running [linux.local]
charuca   3281  2993  0 13:58 pts/0    00:00:00 grep --color=auto linux
[charuca@localhost ~]$
```

- Zombie process:-



```
charuca@localhost:~$ ps aux | grep -w Z
charuca    3223  0.0  0.1 221796  2260 pts/0    S+   13:51   0:00 grep --color=auto -w Z
[charuca@localhost ~]$ S
```

Result:-

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

Experiment - 6

Aim:-

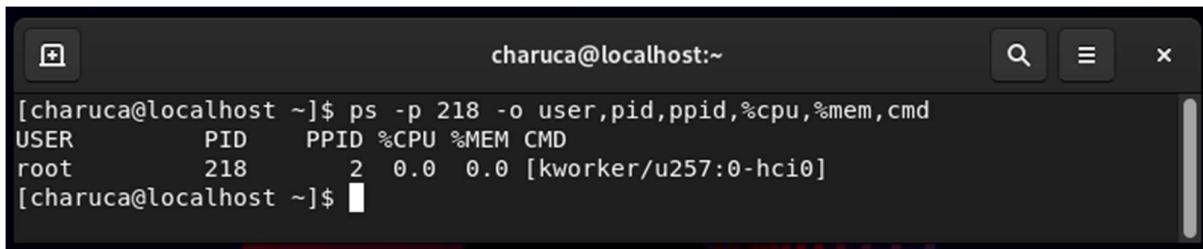
List and display the processes and usage of resources with process id, stop a process temporarily using pid and suspend, kill the process using PID.

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
```

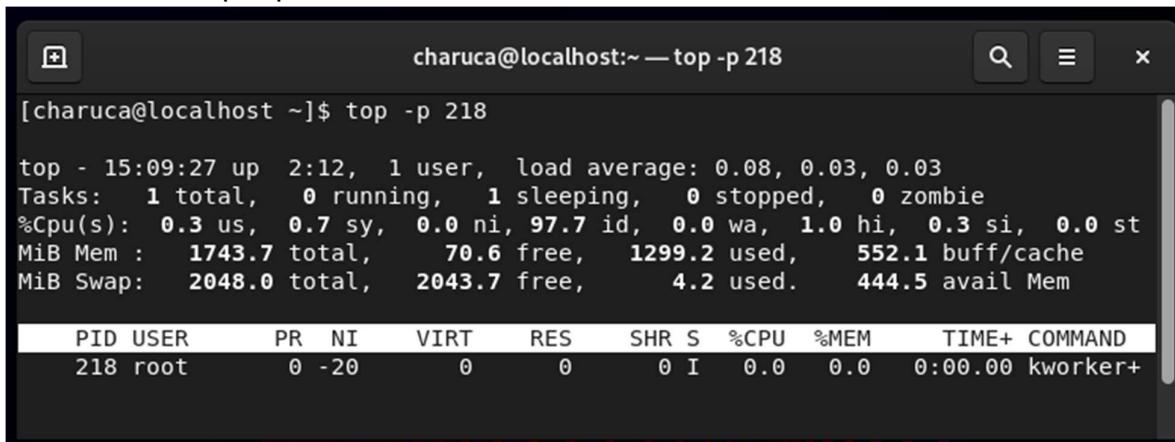


```
[charuca@localhost ~]$ ps -p 218 -o user,pid,ppid,%cpu,%mem,cmd
USER      PID      PPID    %CPU    %MEM CMD
root      218       2     0.0     0.0 [kworker/u257:0-hci0]
[charuca@localhost ~]$
```

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

Syntax : top -p <pid>

```
top -p 218
```



```
[charuca@localhost ~]$ top -p 218
top - 15:09:27 up 2:12, 1 user, load average: 0.08, 0.03, 0.03
Tasks: 1 total, 0 running, 1 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.3 us, 0.7 sy, 0.0 ni, 97.7 id, 0.0 wa, 1.0 hi, 0.3 si, 0.0 st
MiB Mem : 1743.7 total, 70.6 free, 1299.2 used, 552.1 buff/cache
MiB Swap: 2048.0 total, 2043.7 free, 4.2 used. 444.5 avail Mem

PID USER      PR  NI      VIRT      RES      SHR S %CPU %MEM TIME+ COMMAND
218 root      0 -20          0          0          0 I 0.0 0.0 0:00.00 kworker+
```

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

Syntax :

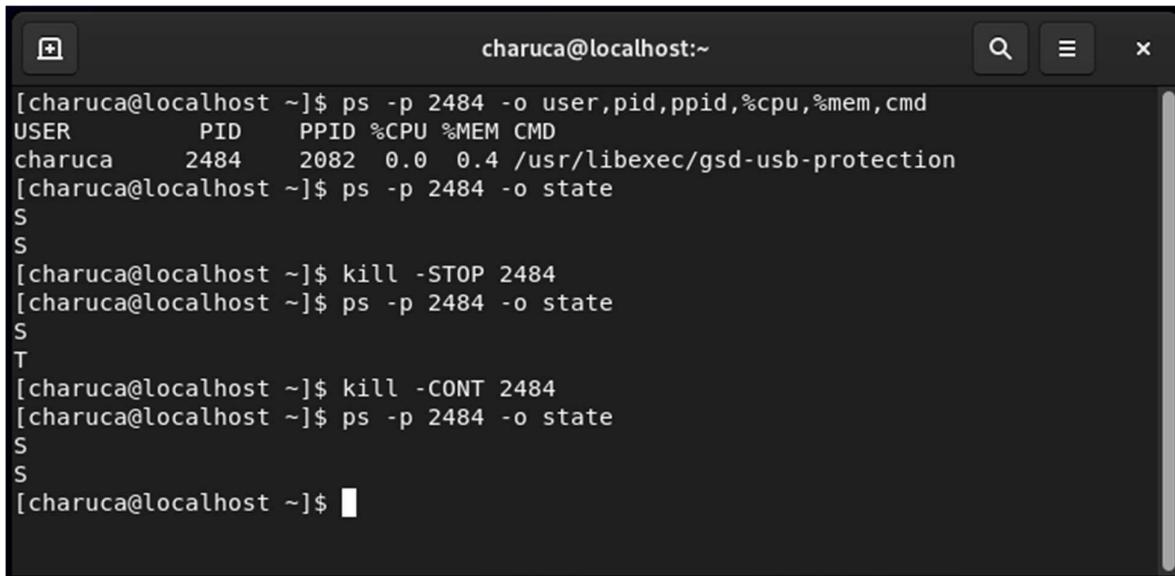
To suspend the process

```
kill -STOP <pid>
```

To resume the process

```
kill -CONT <pid>
```

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



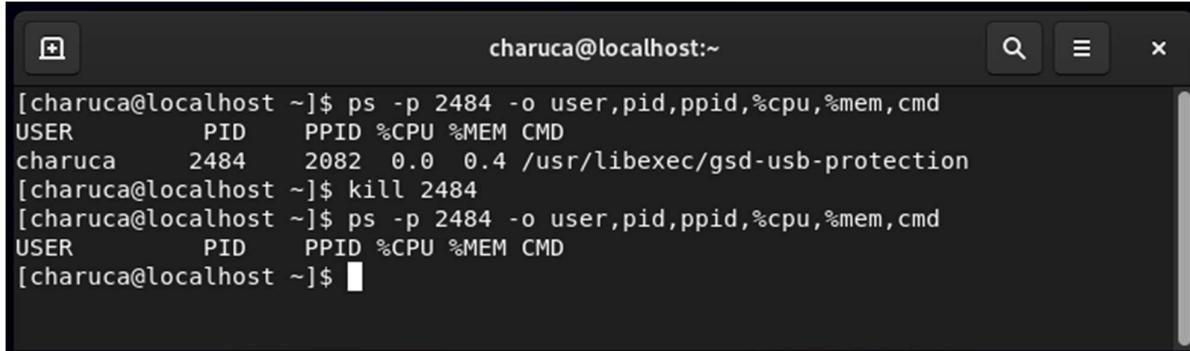
The screenshot shows a terminal window titled "charuca@localhost:~". The user runs "ps -p 2484 -o user,pid,ppid,%cpu,%mem,cmd" to view process details, showing a process for "charuca" with PID 2484. Then, the user runs "kill -STOP 2484", which changes the process state to "S" (Suspending). Finally, the user runs "kill -CONT 2484", which changes the process state back to "T" (Runnable).

```
[charuca@localhost ~]$ ps -p 2484 -o user,pid,ppid,%cpu,%mem,cmd  
USER      PID      PPID %CPU %MEM CMD  
charuca    2484    2082  0.0  0.4 /usr/libexec/gsd-usb-protection  
[charuca@localhost ~]$ ps -p 2484 -o state  
S  
S  
[charuca@localhost ~]$ kill -STOP 2484  
[charuca@localhost ~]$ ps -p 2484 -o state  
S  
T  
[charuca@localhost ~]$ kill -CONT 2484  
[charuca@localhost ~]$ ps -p 2484 -o state  
S  
S  
[charuca@localhost ~]$
```

- Kill the process using pid.

Syntax : kill <pid>

kill 2484



The screenshot shows a terminal window titled "charuca@localhost:~". The user runs the command "ps -p 2484 -o user,pid,ppid,%cpu,%mem,cmd" which lists a process owned by "charuca" with PID 2484. The user then runs "kill 2484" to terminate the process. Finally, they run "ps -p 2484 -o user,pid,ppid,%cpu,%mem,cmd" again, showing that the process has been successfully killed.

```
[charuca@localhost ~]$ ps -p 2484 -o user,pid,ppid,%cpu,%mem,cmd
USER      PID      PPID %CPU %MEM CMD
charuca    2484    2082  0.0  0.4 /usr/libexec/gsd-usb-protection
[charuca@localhost ~]$ kill 2484
[charuca@localhost ~]$ ps -p 2484 -o user,pid,ppid,%cpu,%mem,cmd
USER      PID      PPID %CPU %MEM CMD
[charuca@localhost ~]$
```

 **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 command:
`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

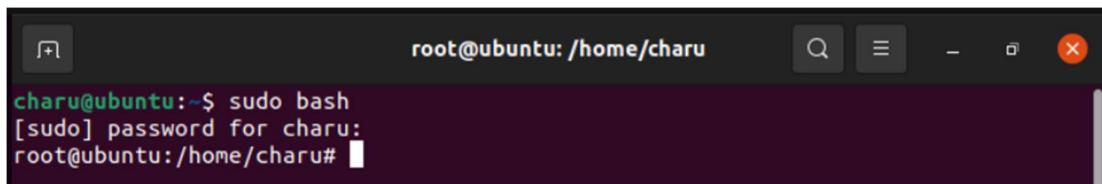
Aim:-

- Create a raw partitions sdb1, sdb2, sdc1, sdc2 using fdisk utility
- Create a partition table
 - Create a swap space of 10G in sdc2
 - Assign the sdb1 as bootable flag
 - Create various file systems such as ext2, ext3, ext4 in those raw partitions

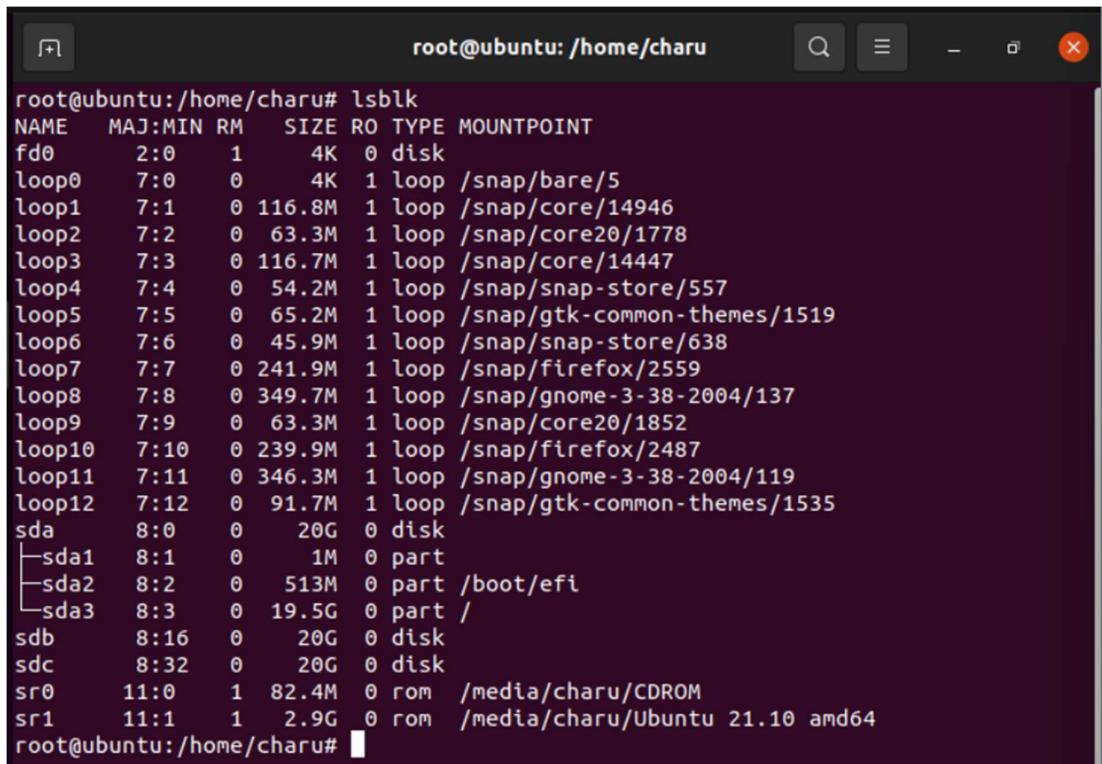
Source Code and Output:-

To go to the root user,

```
sudo bash
```



```
charu@ubuntu:~$ sudo bash
[sudo] password for 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 
sdc     8:32     0   20G  0 disk 
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#
```

a. Create a partition table:-

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

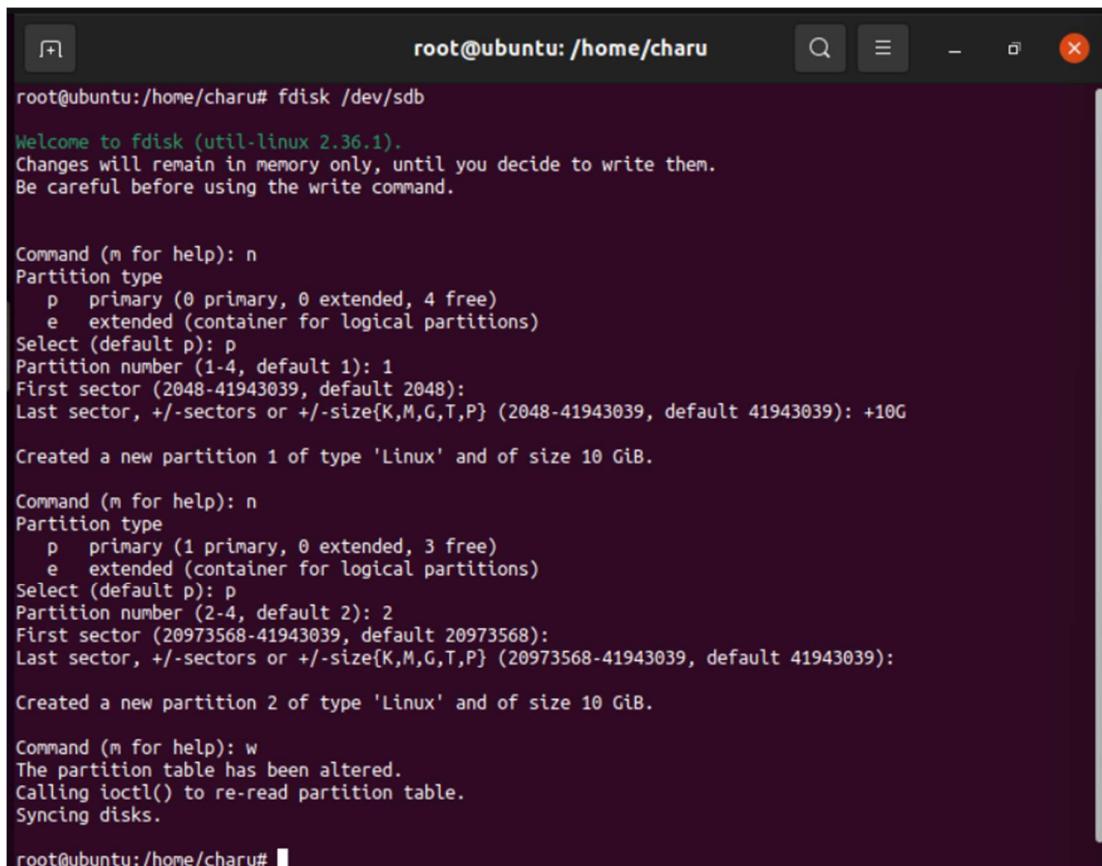
Command (m for help): n
Partition type
    p    primary (1 primary, 0 extended, 3 free)
    e    extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 2): 1
First sector (20973568-41943039, default 20973568):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (20973568-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



The screenshot shows a terminal window titled "root@ubuntu:/home/charu". The terminal displays the output of the fdisk command on the /dev/sdb device. The user is creating two new partitions, both of which are set to be of type 'Linux' (83). The first partition is created from sector 2048 to 41943039, and the second partition follows immediately after it. Both partitions are 10 GiB in size. After creating the partitions, the user writes the changes back to the disk table.

```
root@ubuntu:/home/charu# fdisk /dev/sdb

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): +10G

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# 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.

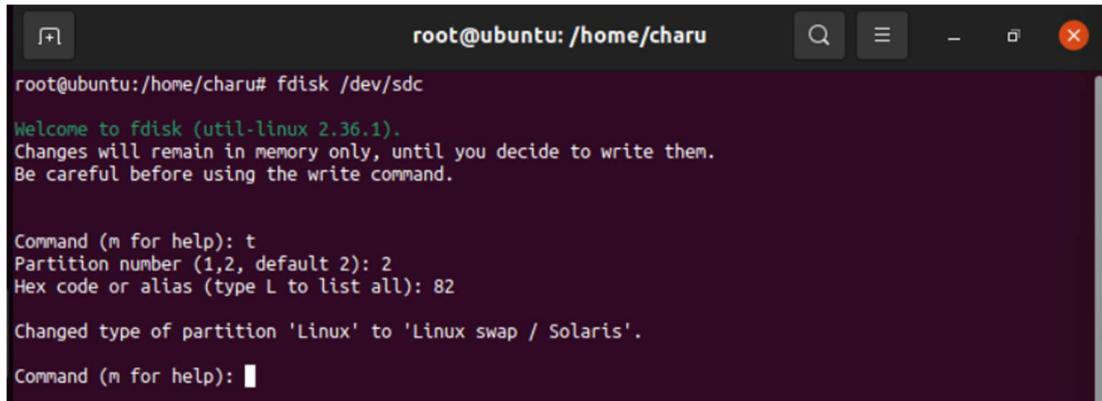
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

root@ubuntu:/home/charu#
```

```
root@ubuntu:/home/charu# 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
```



The screenshot shows a terminal window titled "root@ubuntu:/home/charu". The command "fdisk /dev/sdc" is being run. The output shows the user changing the type of partition 2 from "Linux" to "Linux swap / Solaris" using the "t" command and hex code 82.

```
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.

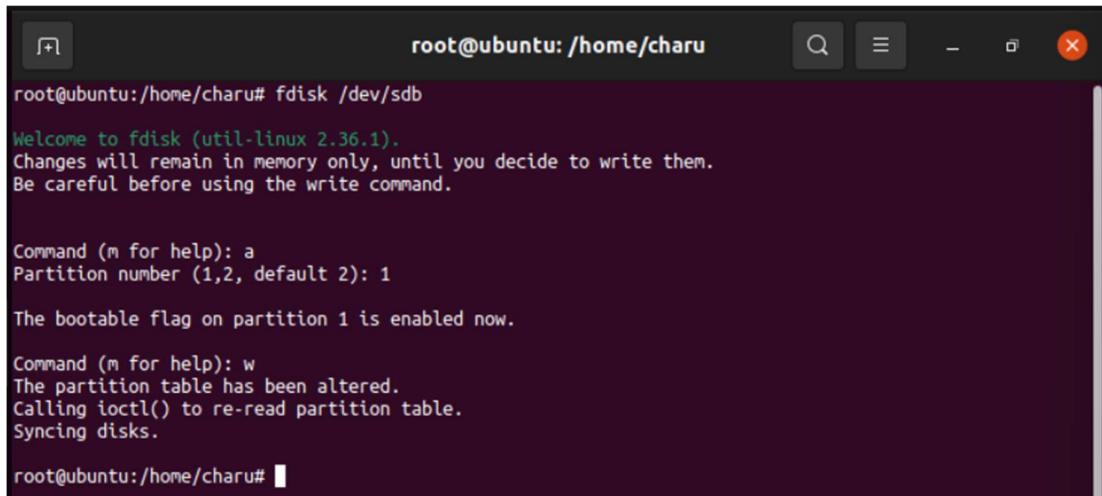
Command (m for help): t
Partition number (1,2, default 2): 2
Hex code or alias (type L to list all): 82

Changed type of partition 'Linux' to 'Linux swap / Solaris'.

Command (m for help):
```

c. Assign the sdb1 as bootable flag

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



The screenshot shows a terminal window titled "root@ubuntu:/home/charu". The command "fdisk /dev/sdb" is being run. The output shows the user enabling the bootable flag for partition 1 using the "a" command. It also shows the command "w" being run to write the changes to the partition table.

```
root@ubuntu:/home/charu# fdisk /dev/sdb
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): a
Partition number (1,2, default 2): 1

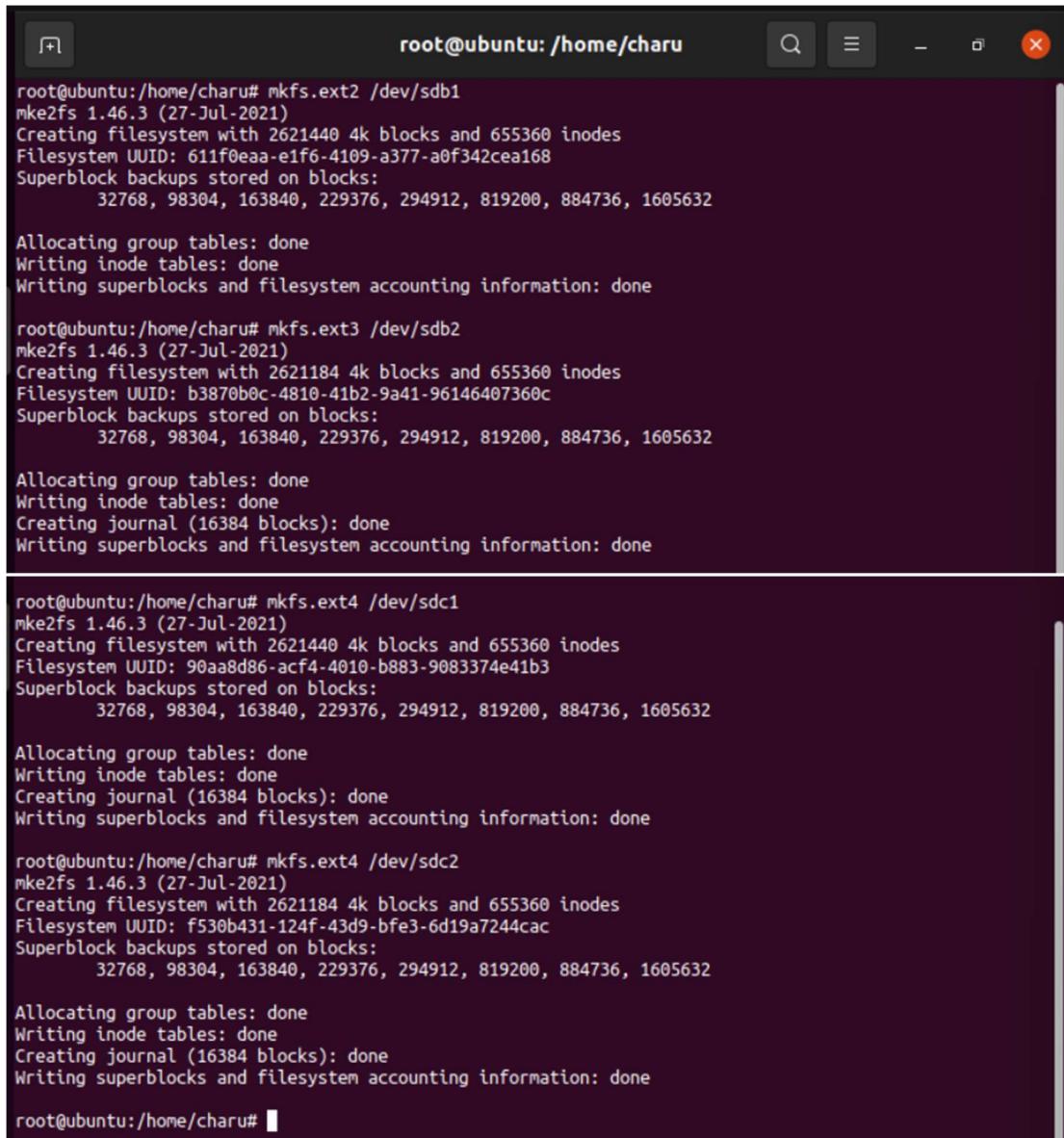
The bootable flag on partition 1 is enabled now.

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

root@ubuntu:/home/charu#
```

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
```



A terminal window titled "root@ubuntu:/home/charu" showing the execution of four mkfs commands. The first command creates an ext2 filesystem on /dev/sdb1. The second creates an ext3 filesystem on /dev/sdb2. The third creates an ext4 filesystem on /dev/sdc1. The fourth creates an ext4 filesystem on /dev/sdc2. Each command displays the mke2fs version, creation date, file system parameters, UUID, superblock backups, and the progress of group tables, inode tables, journal, and superblocks.

```
root@ubuntu:/home/charu# mkfs.ext2 /dev/sdb1  
mke2fs 1.46.3 (27-Jul-2021)  
Creating filesystem with 2621440 4k blocks and 655360 inodes  
Filesystem UUID: 611f0eaa-e1f6-4109-a377-a0f342cea168  
Superblock backups stored on blocks:  
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632  
  
Allocating group tables: done  
Writing inode tables: done  
Writing superblocks and filesystem accounting information: done  
  
root@ubuntu:/home/charu# mkfs.ext3 /dev/sdb2  
mke2fs 1.46.3 (27-Jul-2021)  
Creating filesystem with 2621184 4k blocks and 655360 inodes  
Filesystem UUID: b3870b0c-4810-41b2-9a41-96146407360c  
Superblock backups stored on blocks:  
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632  
  
Allocating group tables: done  
Writing inode tables: done  
Creating journal (16384 blocks): done  
Writing superblocks and filesystem accounting information: done  
  
root@ubuntu:/home/charu# mkfs.ext4 /dev/sdc1  
mke2fs 1.46.3 (27-Jul-2021)  
Creating filesystem with 2621440 4k blocks and 655360 inodes  
Filesystem UUID: 90aa8d86-acf4-4010-b883-9083374e41b3  
Superblock backups stored on blocks:  
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632  
  
Allocating group tables: done  
Writing inode tables: done  
Creating journal (16384 blocks): done  
Writing superblocks and filesystem accounting information: done  
  
root@ubuntu:/home/charu# mkfs.ext4 /dev/sdc2  
mke2fs 1.46.3 (27-Jul-2021)  
Creating filesystem with 2621184 4k blocks and 655360 inodes  
Filesystem UUID: f530b431-124f-43d9-bfe3-6d19a7244cac  
Superblock backups stored on blocks:  
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632  
  
Allocating group tables: done  
Writing inode tables: done  
Creating journal (16384 blocks): done  
Writing superblocks and filesystem accounting information: done  
  
root@ubuntu:/home/charu#
```

 **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:-

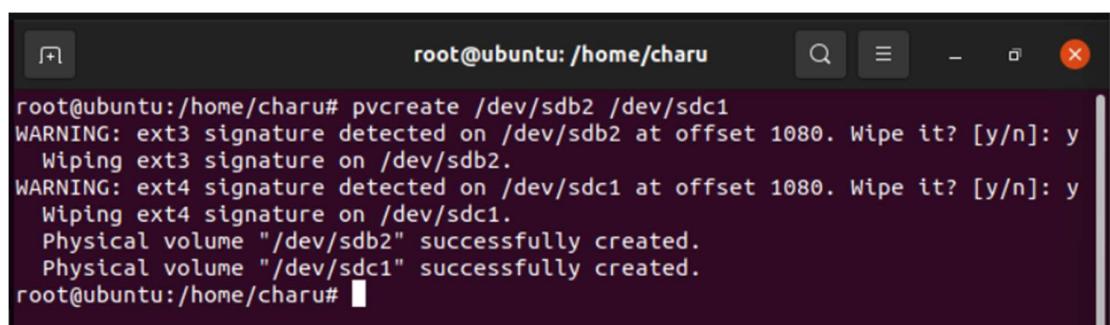
The steps to configure LVM by creating the physical volumes PV1 and PV2 from sdb2 and sdc1, and Logical volume group LVG and logical volumes LV1, LV2, and LV3:

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
```



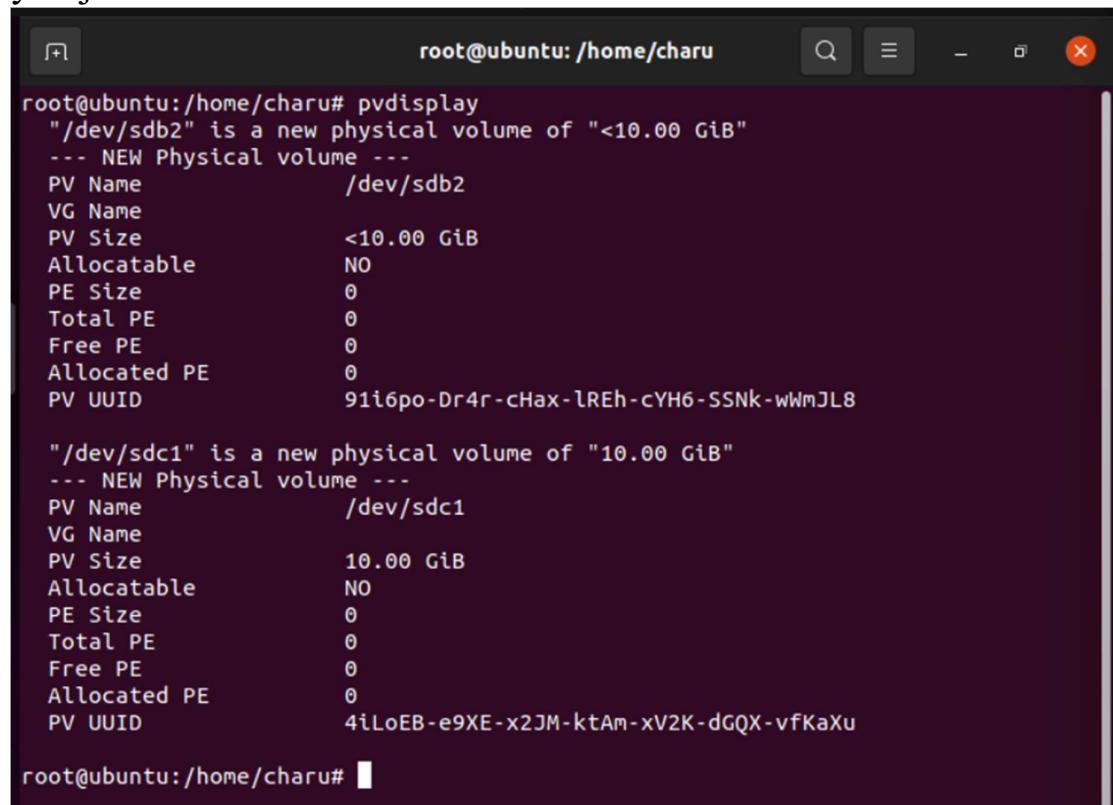
A screenshot of a terminal window titled "root@ubuntu:/home/charu". The terminal shows the command "pvcreate /dev/sdb2 /dev/sdc1" being run. The output includes several "WARNING" messages about ext3 and ext4 signatures and asks if the user wants to wipe them. The user responds with "y". The terminal then displays that two physical volumes have been successfully created.

```
root@ubuntu:/home/charu# pvcreate /dev/sdb2 /dev/sdc1  
WARNING: ext3 signature detected on /dev/sdb2 at offset 1080. Wipe it? [y/n]: y  
      Wiping ext3 signature on /dev/sdb2.  
WARNING: ext4 signature detected on /dev/sdc1 at offset 1080. Wipe it? [y/n]: y  
      Wiping ext4 signature on /dev/sdc1.  
Physical volume "/dev/sdb2" successfully created.  
Physical volume "/dev/sdc1" successfully created.  
root@ubuntu:/home/charu#
```

3. Verify the physical volumes using the following command:

```
pvdisplay
```

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



A terminal window titled "root@ubuntu:/home/charu". The command "pvdisplay" is run, showing two physical volumes: "/dev/sdb2" and "/dev/sdc1". The output includes the PV Name, VG Name, PV Size, Allocatable, PE Size, Total PE, Free PE, Allocated PE, and PV UUID for each volume.

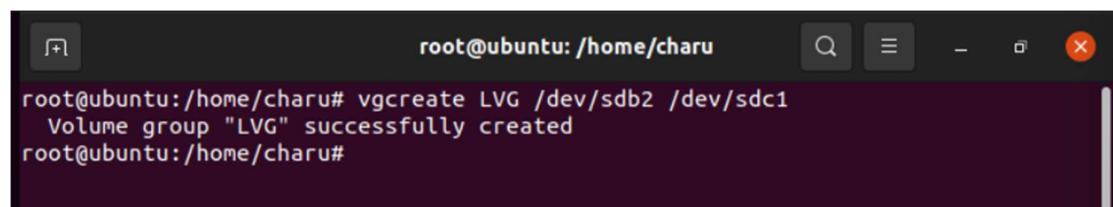
```
root@ubuntu:/home/charu# pvdisplay
"/dev/sdb2" is a new physical volume of "<10.00 GiB"
--- NEW Physical volume ---
PV Name            /dev/sdb2
VG Name
PV Size           <10.00 GiB
Allocatable       NO
PE Size           0
Total PE          0
Free PE           0
Allocated PE      0
PV UUID           91i6po-Dr4r-cHax-lREh-cYH6-SSNK-wWmJL8

"/dev/sdc1" is a new physical volume of "10.00 GiB"
--- NEW Physical volume ---
PV Name            /dev/sdc1
VG Name
PV Size           10.00 GiB
Allocatable       NO
PE Size           0
Total PE          0
Free PE           0
Allocated PE      0
PV UUID           4iLoEB-e9XE-x2JM-ktAm-xV2K-dGQX-vfKaXu

root@ubuntu:/home/charu#
```

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

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



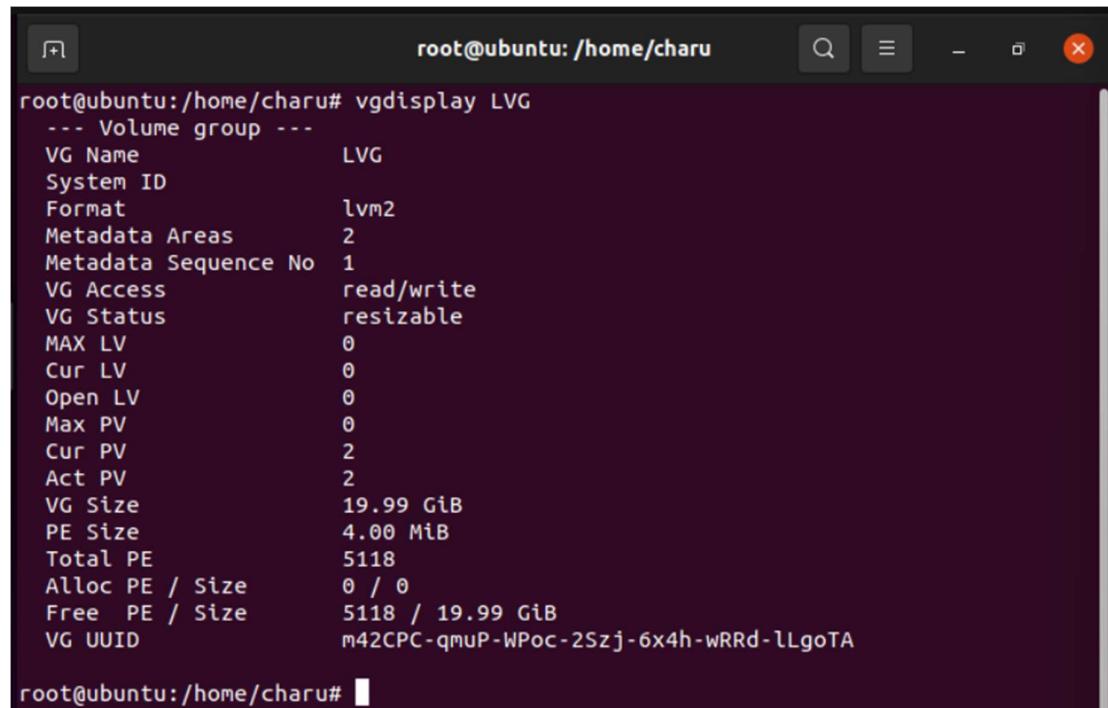
A terminal window titled "root@ubuntu:/home/charu". The command "vgcreate LVG /dev/sdb2 /dev/sdc1" is run, creating a volume group named "LVG". The output shows the message "Volume group "LVG" successfully created".

```
root@ubuntu:/home/charu# vgcreate LVG /dev/sdb2 /dev/sdc1
Volume group "LVG" successfully created
root@ubuntu:/home/charu#
```

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.



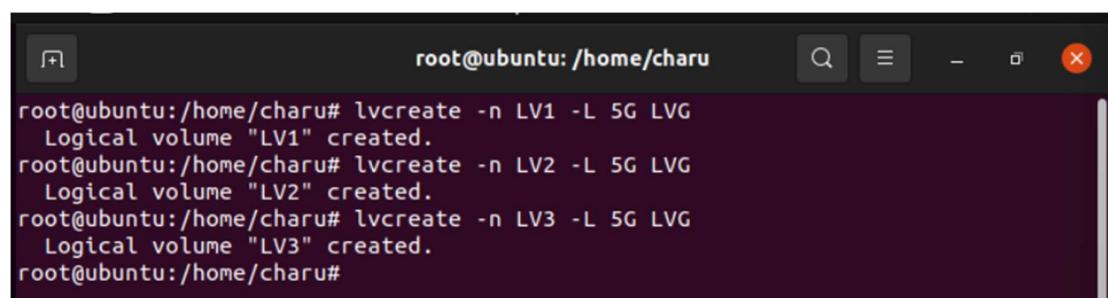
```
root@ubuntu:/home/charu# vgdisplay LVG
--- Volume group ---
VG Name          LVG
System ID
Format          lvm2
Metadata Areas   2
Metadata Sequence No  1
VG Access        read/write
VG Status        resizable
MAX LV
Cur LV
Open LV
Max PV
Cur PV
Act PV
VG Size         19.99 GiB
PE Size          4.00 MiB
Total PE        5118
Alloc PE / Size  0 / 0
Free  PE / Size  5118 / 19.99 GiB
VG UUID          m42CPC-qmuP-WPoc-2Szj-6x4h-wRRd-lLgoTA

root@ubuntu:/home/charu#
```

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.

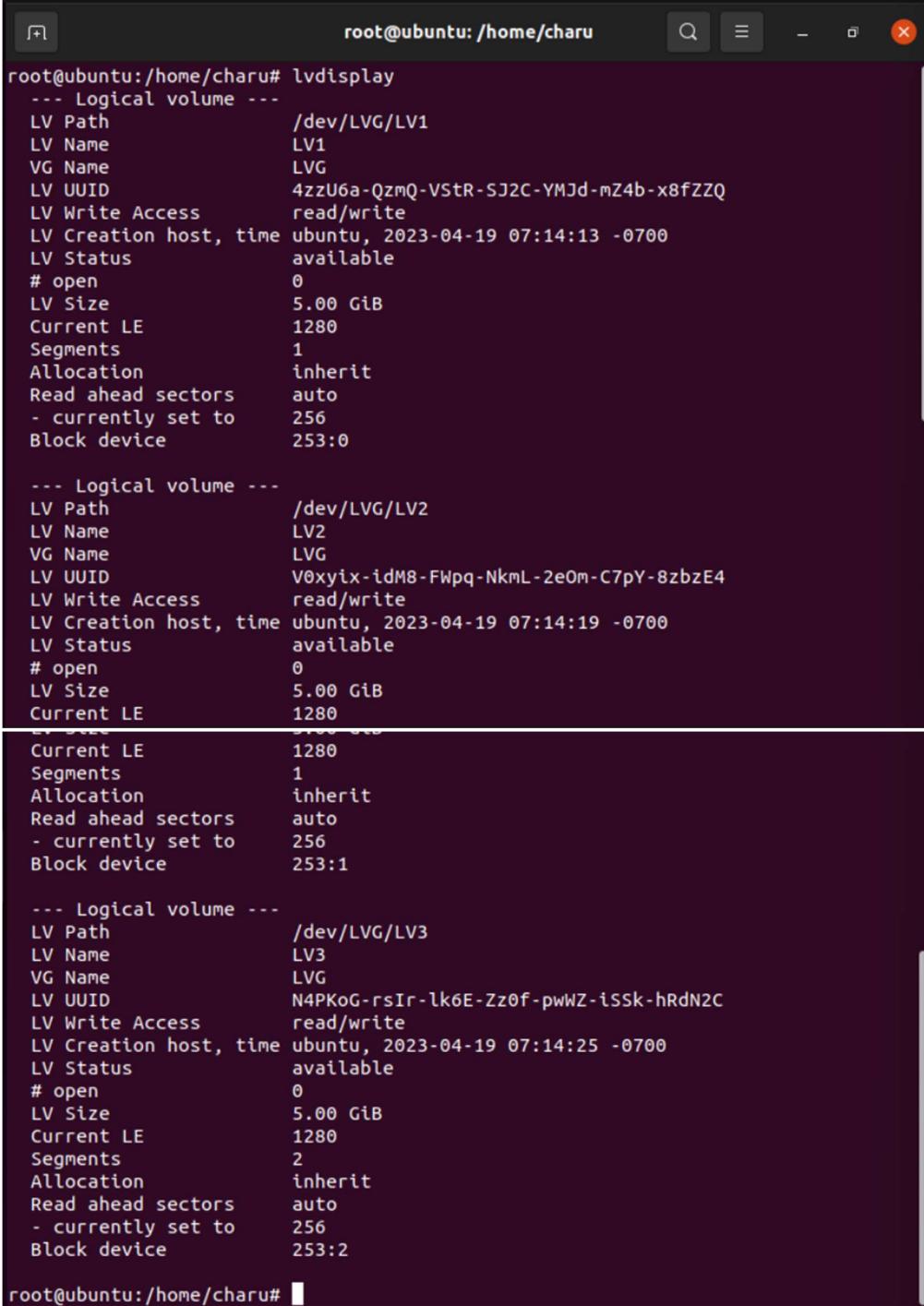


```
root@ubuntu:/home/charu# lvcreate -n LV1 -L 5G LVG
Logical volume "LV1" created.
root@ubuntu:/home/charu# lvcreate -n LV2 -L 5G LVG
Logical volume "LV2" created.
root@ubuntu:/home/charu# lvcreate -n LV3 -L 5G LVG
Logical volume "LV3" created.
root@ubuntu:/home/charu#
```

7. Verify the logical volumes using the following command:

```
lvdisplay
```

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



The screenshot shows a terminal window titled "root@ubuntu:/home/charu". The command "lvdisplay" is run, displaying detailed information about three logical volumes (LV1, LV2, and LV3) within a volume group (VG Name: LVG). The output includes fields such as LV Path, LV Name, VG Name, LV UUID, LV Write Access, LV Creation host, time, LV Status, # open, LV Size, Current LE, Segments, Allocation, Read ahead sectors, and Block device. The logical volumes are defined with specific parameters like size (5.00 GiB), allocation (inherit), and read-ahead sectors (auto).

```
root@ubuntu:/home/charu# lvdisplay
  --- Logical volume ---
  LV Path      /dev/LVG/LV1
  LV Name     LV1
  VG Name     LVG
  LV UUID    4zzU6a-QzmQ-VStR-SJ2C-YMJD-mZ4b-x8fZZQ
  LV Write Access  read/write
  LV Creation host, time ubuntu, 2023-04-19 07:14:13 -0700
  LV Status    available
  # open       0
  LV Size     5.00 GiB
  Current LE   1280
  Segments     1
  Allocation   inherit
  Read ahead sectors auto
  - currently set to 256
  Block device 253:0

  --- Logical volume ---
  LV Path      /dev/LVG/LV2
  LV Name     LV2
  VG Name     LVG
  LV UUID    V0xyix-idM8-FWpq-NkmL-2e0m-C7pY-8zbzE4
  LV Write Access  read/write
  LV Creation host, time ubuntu, 2023-04-19 07:14:19 -0700
  LV Status    available
  # open       0
  LV Size     5.00 GiB
  Current LE   1280
  LV SECs      1
  Current LE   1280
  Segments     1
  Allocation   inherit
  Read ahead sectors auto
  - currently set to 256
  Block device 253:1

  --- Logical volume ---
  LV Path      /dev/LVG/LV3
  LV Name     LV3
  VG Name     LVG
  LV UUID    N4PKoG-rsIr-lk6E-Zz0f-pwWZ-iSSk-hRdN2C
  LV Write Access  read/write
  LV Creation host, time ubuntu, 2023-04-19 07:14:25 -0700
  LV Status    available
  # open       0
  LV Size     5.00 GiB
  Current LE   1280
  Segments     2
  Allocation   inherit
  Read ahead sectors auto
  - currently set to 256
  Block device 253:2

root@ubuntu:/home/charu#
```

 **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:-

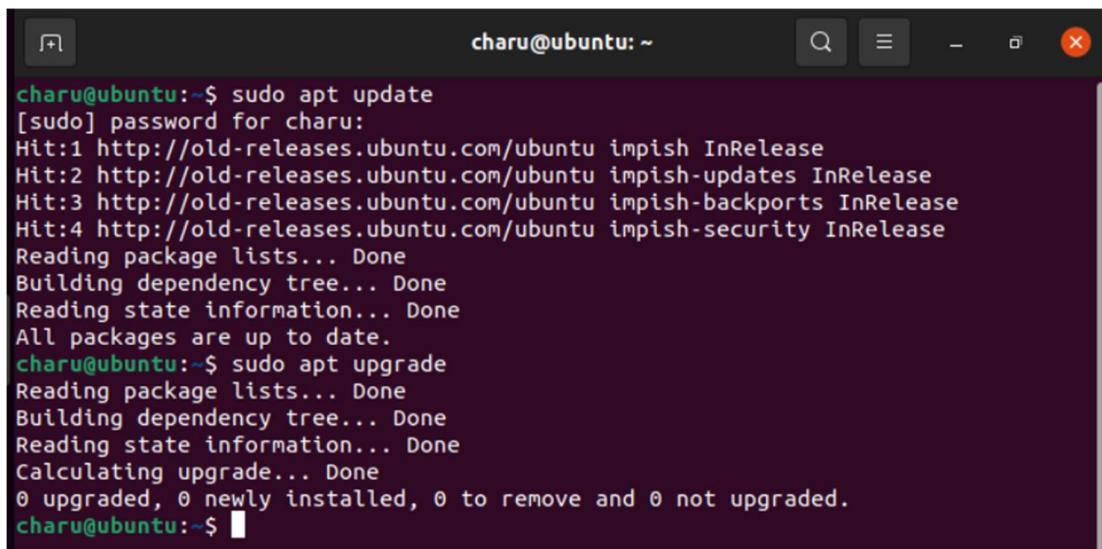
Configure apache webserver using httpd with port number 8081 and install open ssl certificate to deploy the webserver in https.

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
```

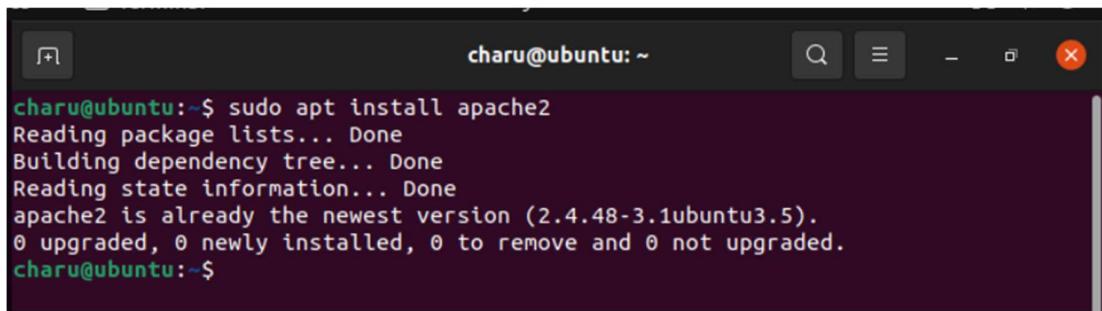


A screenshot of a terminal window titled "charu@ubuntu: ~". The window shows the command "sudo apt update" followed by its output, which includes hits from various Ubuntu mirrors and a message stating all packages are up to date. Below that, the command "sudo apt upgrade" is run, showing a similar process and concluding with a message that 0 packages were upgraded.

```
charu@ubuntu:~$ sudo apt update  
[sudo] password for charu:  
Hit:1 http://old-releases.ubuntu.com/ubuntu impish InRelease  
Hit:2 http://old-releases.ubuntu.com/ubuntu impish-updates InRelease  
Hit:3 http://old-releases.ubuntu.com/ubuntu impish-backports InRelease  
Hit:4 http://old-releases.ubuntu.com/ubuntu impish-security InRelease  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
All packages are up to date.  
charu@ubuntu:~$ sudo apt upgrade  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
Calculating upgrade... Done  
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.  
charu@ubuntu:~$
```

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

```
sudo apt install apache2
```



```
charu@ubuntu:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apache2 is already the newest version (2.4.48-3.1ubuntu3.5).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
charu@ubuntu:~$
```

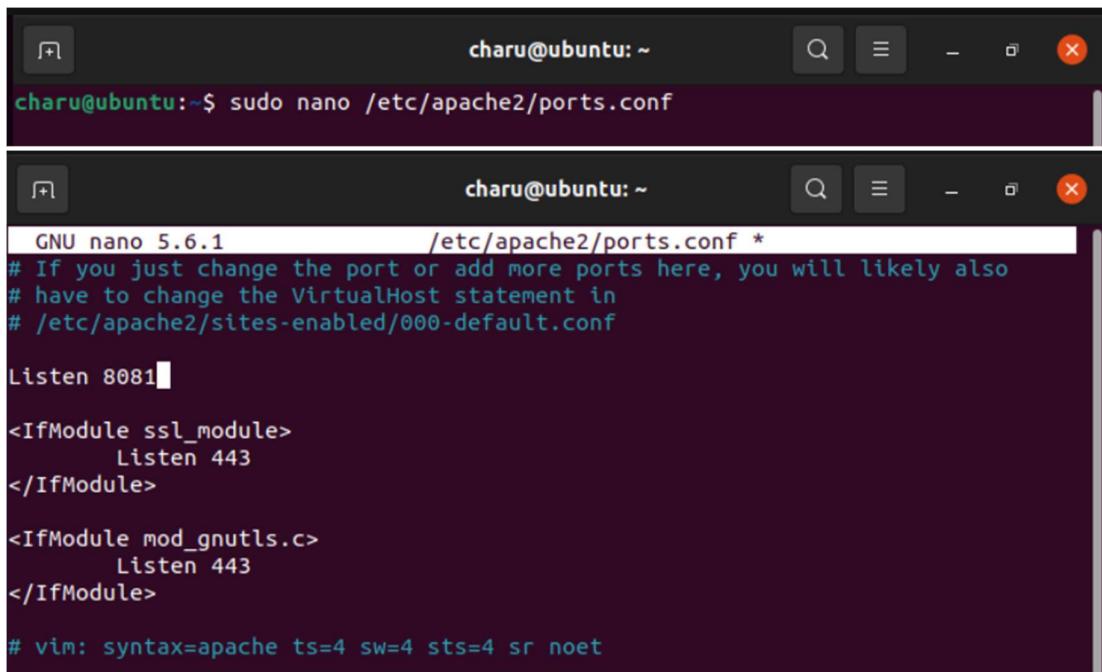
Step 3: Configure Apache to Use Port 8081

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

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

```
Listen 8081
```

Save the file and exit.



```
charu@ubuntu:~$ sudo nano /etc/apache2/ports.conf
```



```
GNU nano 5.6.1          /etc/apache2/ports.conf *
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 8081

<IfModule ssl_module>
    Listen 443
</IfModule>

<IfModule mod_gnutls.c>
    Listen 443
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Step 4: Create a New Virtual Host Configuration

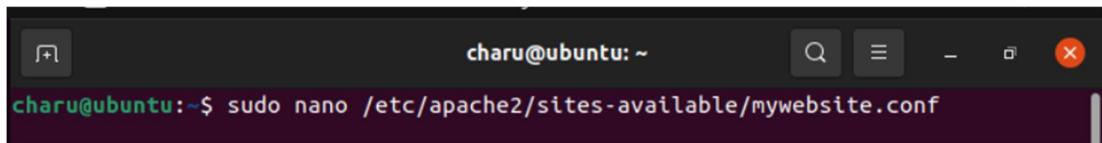
```
sudo nano /etc/apache2/sites-available/mywebsite.conf
```

Add the following content to the file:

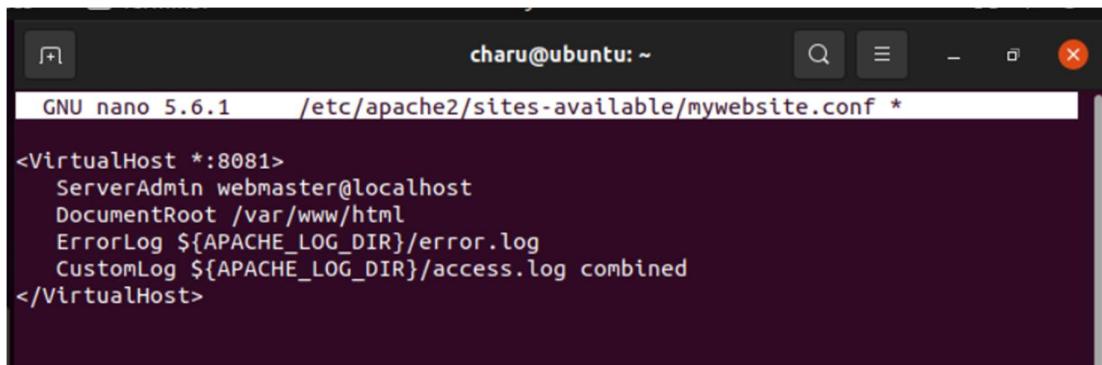
```
<VirtualHost *:8081>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

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

Save the file and exit.



```
charu@ubuntu:~$ sudo nano /etc/apache2/sites-available/mywebsite.conf
```

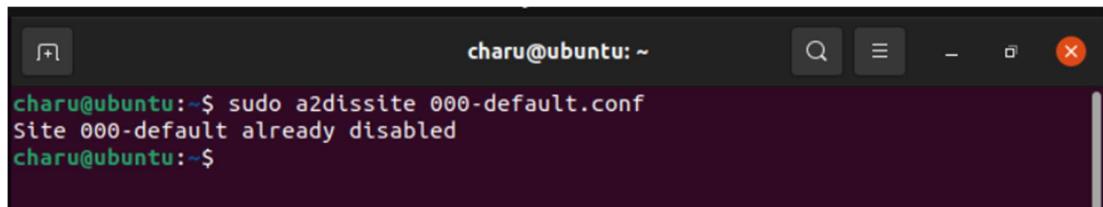


```
GNU nano 5.6.1      /etc/apache2/sites-available/mywebsite.conf *

<VirtualHost *:8081>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Step 5: Disable the Default Virtual Host Configuration

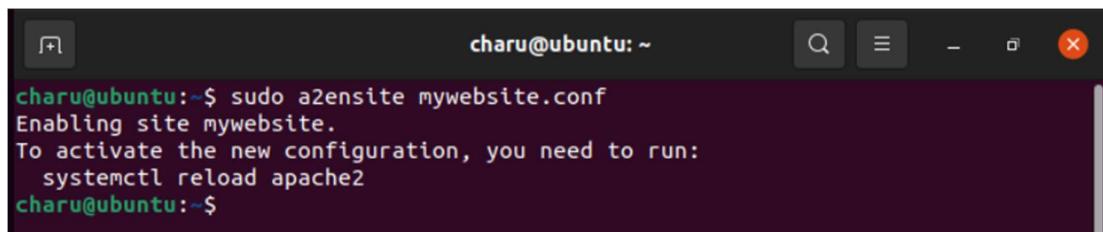
```
sudo a2dissite 000-default.conf
```



```
charu@ubuntu:~$ sudo a2dissite 000-default.conf
Site 000-default already disabled
charu@ubuntu:~$
```

Step 6: Enable the New Virtual Host Configuration

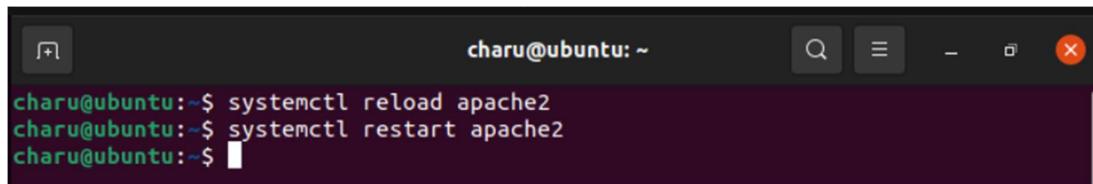
```
sudo a2ensite mywebsite.conf
```



```
charu@ubuntu:~$ sudo a2ensite mywebsite.conf
Enabling site mywebsite.
To activate the new configuration, you need to run:
  systemctl reload apache2
charu@ubuntu:~$
```

Step 7: Restart Apache Service

```
sudo service apache2 restart
```

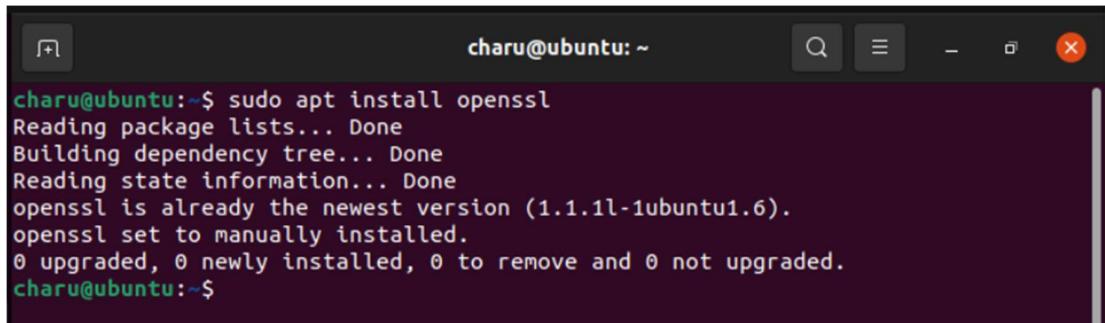


```
charu@ubuntu:~$ systemctl reload apache2
charu@ubuntu:~$ systemctl restart apache2
charu@ubuntu:~$
```

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

Step 8: Install OpenSSL

```
sudo apt install openssl
```

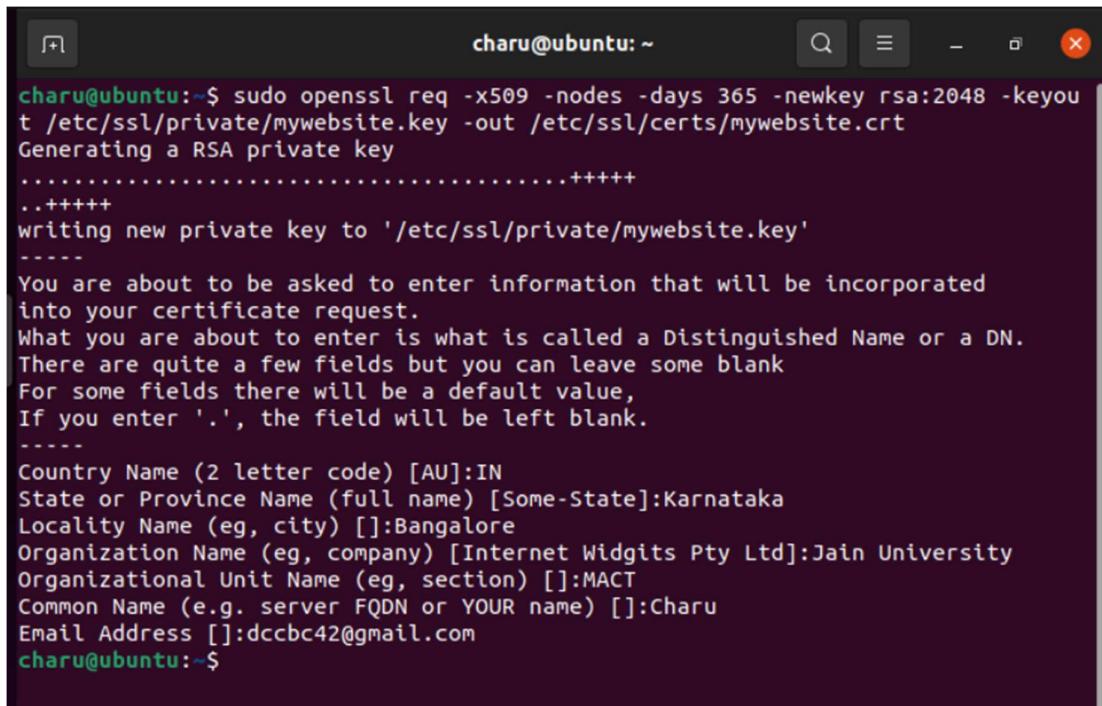


```
charu@ubuntu:~$ sudo apt install openssl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
openssl is already the newest version (1.1.1l-1ubuntu1.6).
openssl set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
charu@ubuntu:~$
```

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.



```
charu@ubuntu:~$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyou
t /etc/ssl/private/mywebsite.key -out /etc/ssl/certs/mywebsite.crt
Generating a RSA private key
.....+++++
.....+++++
writing new private key to '/etc/ssl/private/mywebsite.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:IN
State or Province Name (full name) [Some-State]:Karnataka
Locality Name (eg, city) []:Bangalore
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Jain University
Organizational Unit Name (eg, section) []:MACT
Common Name (e.g. server FQDN or YOUR name) []:Charu
Email Address []:dccbc42@gmail.com
charu@ubuntu:~$
```

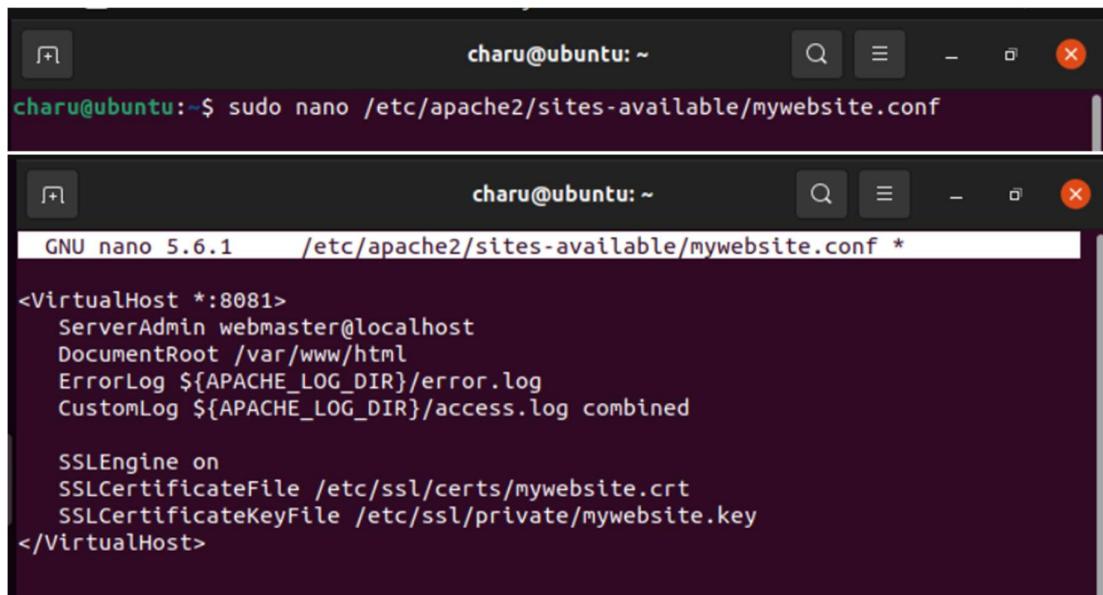
Step 10: Configure Apache to Use SSL

```
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.

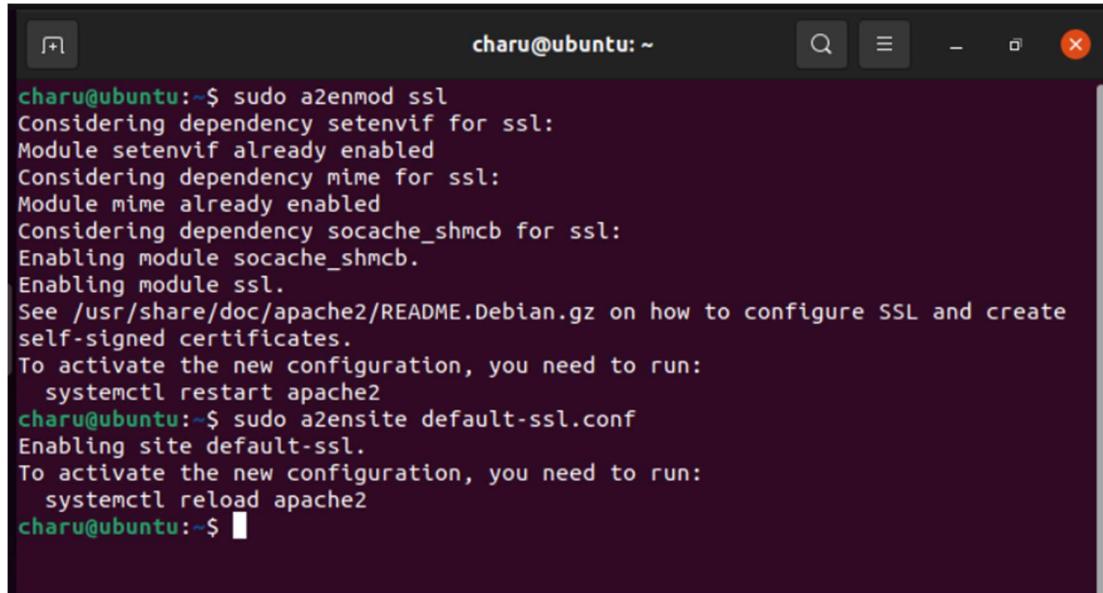


The screenshot shows a terminal window with two tabs. The top tab is titled "charu@ubuntu: ~" and contains the command: "charu@ubuntu:~\$ sudo nano /etc/apache2/sites-available/mywebsite.conf". The bottom tab is also titled "charu@ubuntu: ~" and shows the contents of the file being edited:

```
GNU nano 5.6.1      /etc/apache2/sites-available/mywebsite.conf *  
  
<VirtualHost *:8081>  
    ServerAdmin webmaster@localhost  
    DocumentRoot /var/www/html  
    ErrorLog ${APACHE_LOG_DIR}/error.log  
    CustomLog ${APACHE_LOG_DIR}/access.log combined  
  
    SSLEngine on  
    SSLCertificateFile /etc/ssl/certs/mywebsite.crt  
    SSLCertificateKeyFile /etc/ssl/private/mywebsite.key  
</VirtualHost>
```

Step 11: Enable SSL Module and Default SSL Site

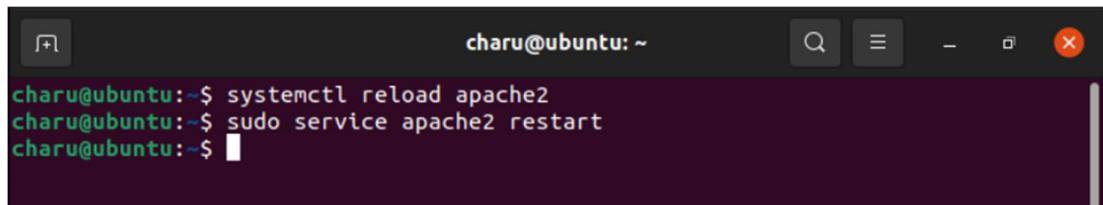
```
sudo a2enmod ssl  
sudo a2ensite default-ssl.conf
```



```
charu@ubuntu:~$ sudo a2enmod ssl  
Considering dependency setenvif for ssl:  
Module setenvif already enabled  
Considering dependency mime for ssl:  
Module mime already enabled  
Considering dependency socache_shmcb for ssl:  
Enabling module socache_shmcb.  
Enabling module ssl.  
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create  
self-signed certificates.  
To activate the new configuration, you need to run:  
    systemctl restart apache2  
charu@ubuntu:~$ sudo a2ensite default-ssl.conf  
Enabling site default-ssl.  
To activate the new configuration, you need to run:  
    systemctl reload apache2  
charu@ubuntu:~$
```

Step 12: Restart Apache Service

```
sudo service apache2 restart
```



```
charu@ubuntu:~$ systemctl reload apache2  
charu@ubuntu:~$ sudo service apache2 restart  
charu@ubuntu:~$
```

Step 13: Check the status of Apache status

```
systemctl status apache2
```

```
charu@ubuntu:~$ systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor pres>
   Active: active (running) since Sat 2023-05-20 06:06:26 PDT; 3min 33s ago
     Docs: https://httpd.apache.org/docs/2.4/
 Process: 23883 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/>
 Main PID: 23887 (apache2)
    Tasks: 55 (limit: 4603)
   Memory: 5.8M
      CPU: 117ms
     CGroup: /system.slice/apache2.service
             ├─23887 /usr/sbin/apache2 -k start
             ├─23888 /usr/sbin/apache2 -k start
             └─23889 /usr/sbin/apache2 -k start

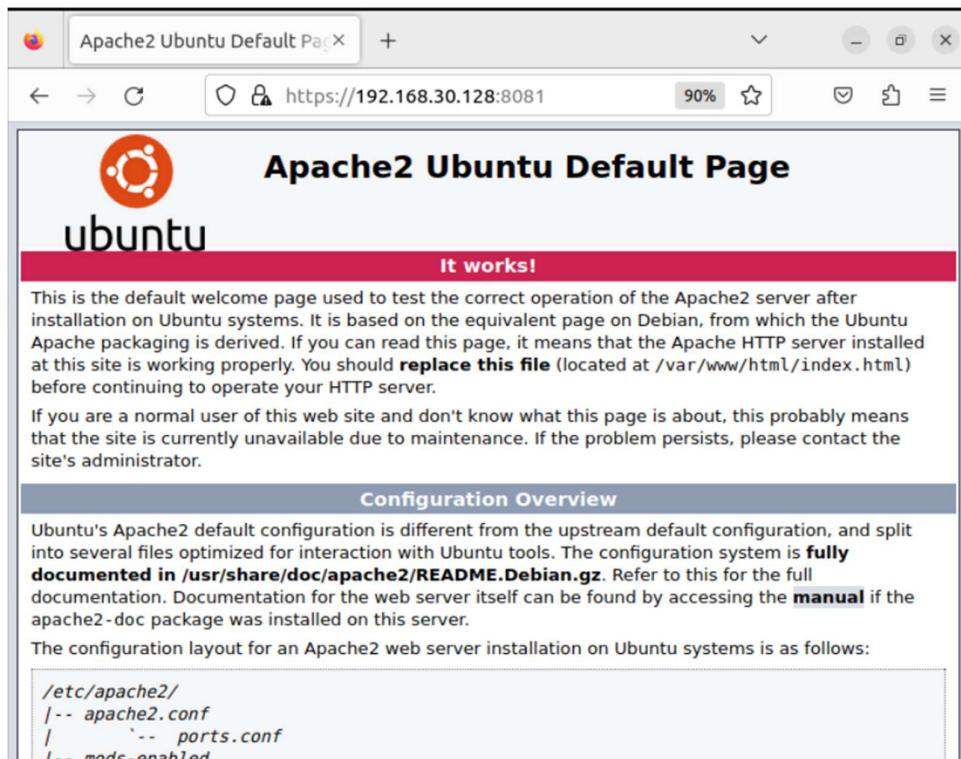
May 20 06:06:26 ubuntu systemd[1]: Starting The Apache HTTP Server...
May 20 06:06:26 ubuntu systemd[1]: Started The Apache HTTP Server.
lines 1-16/16 (END)
```

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:-

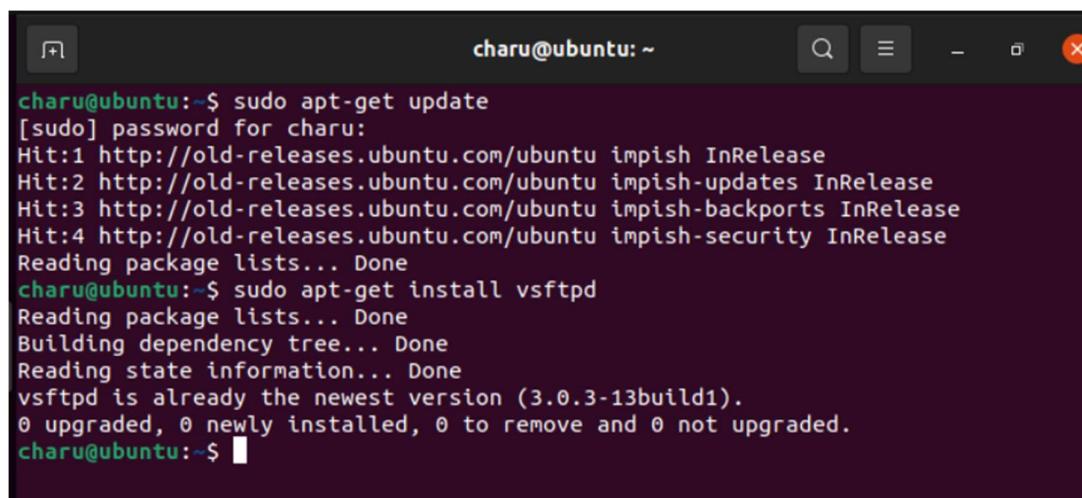
Configure the FTP server and enable the FTP to be accessible from webserver.

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:

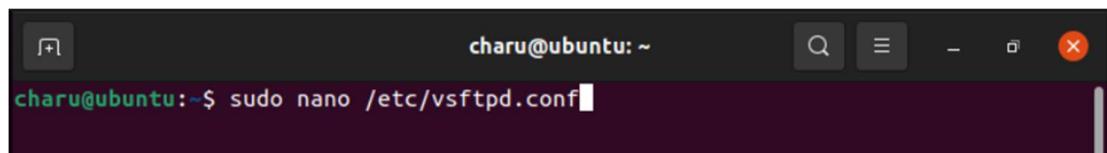
```
sudo apt-get update  
sudo apt-get install vsftpd
```



```
charu@ubuntu:~$ sudo apt-get update  
[sudo] password for charu:  
Hit:1 http://old-releases.ubuntu.com/ubuntu impish InRelease  
Hit:2 http://old-releases.ubuntu.com/ubuntu impish-updates InRelease  
Hit:3 http://old-releases.ubuntu.com/ubuntu impish-backports InRelease  
Hit:4 http://old-releases.ubuntu.com/ubuntu impish-security InRelease  
Reading package lists... Done  
charu@ubuntu:~$ sudo apt-get install vsftpd  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
vsftpd is already the newest version (3.0.3-13build1).  
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.  
charu@ubuntu:~$
```

2. Once the installation is complete, open the configuration file for vsftpd using a text editor. For example:

```
sudo nano /etc/vsftpd.conf
```

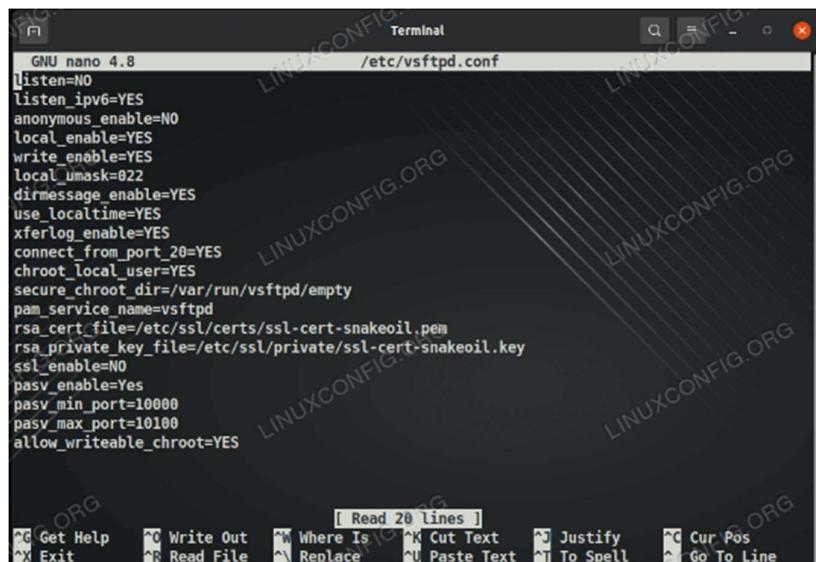


```
charu@ubuntu:~$ sudo nano /etc/vsftpd.conf
```

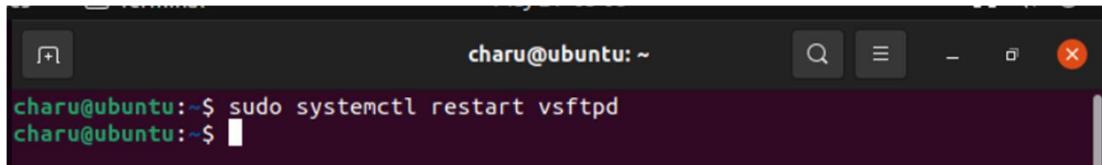
3. Make the following changes in the configuration file:

```
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.



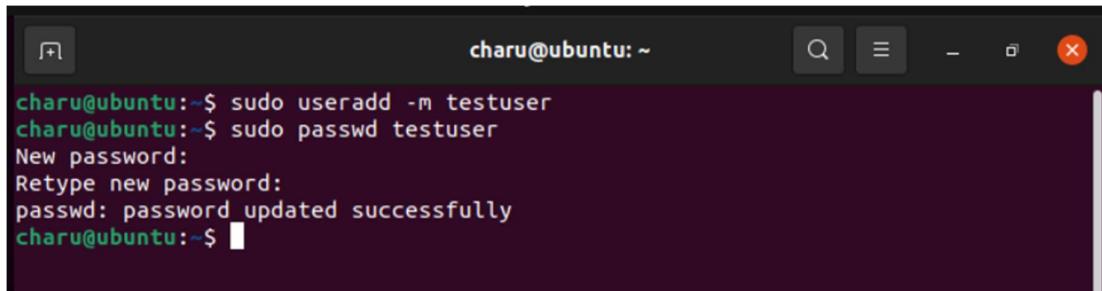
4. Restart the vsftpd service to apply the configuration changes:
`sudo systemctl restart vsftpd`



A screenshot of a terminal window titled "charu@ubuntu: ~". The window shows the command `sudo systemctl restart vsftpd` being typed at the prompt. The terminal has a dark background with white text and standard Linux-style icons.

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
```



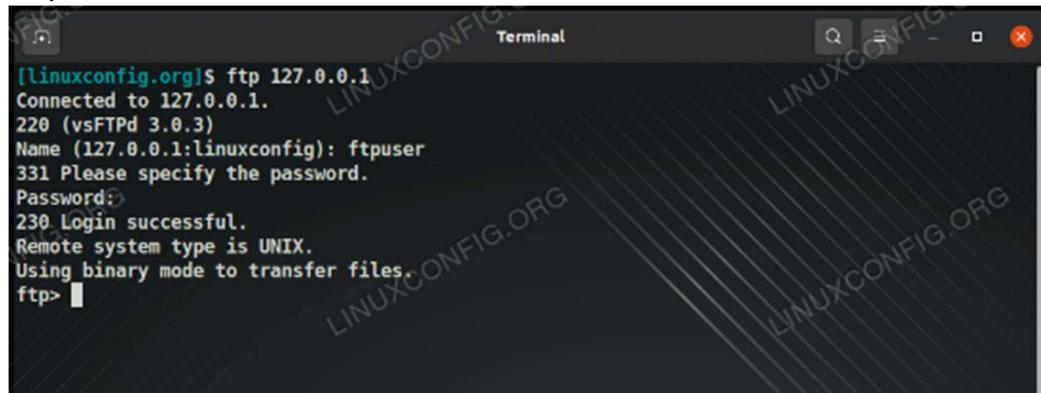
A screenshot of a terminal window titled "charu@ubuntu: ~". The window shows two commands being run: `sudo useradd -m testuser` and `sudo passwd testuser`. The user is prompted for a new password and asked to retype it. The message "passwd: password updated successfully" is displayed. The terminal has a dark background with white text and standard Linux-style icons.

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
```



```
[linuxconfig.org]$ ftp 127.0.0.1
Connected to 127.0.0.1.
220 (vsFTPd 3.0.3)
Name (127.0.0.1:linuxconfig): ftpuser
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> 
```

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

```
ftp> ls
```



```
[linuxconfig.org]$ ftp 127.0.0.1
Connected to 127.0.0.1.
220 (vsFTPd 3.0.3)
Name (127.0.0.1:linuxconfig): ftpuser
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-r--r-- 1 0 0 12 May 12 10:40 FTP-TEST
226 Directory send OK.
ftp>
```

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
```

```
[linuxconfig.org]$ systemctl status vsftpd  
● vsftpd.service - vsftpd FTP server  
   Loaded: loaded (/lib/systemd/system/vsftpd.service; enabled; vendor preset: enabled)  
   Active: active (running) since Wed 2021-05-12 11:04:43 EDT; 11s ago  
     Process: 2618 ExecStartPre=/bin/mkdir -p /var/run/vsftpd/empty (code=exited, status=0/SUCCESS)  
    Main PID: 2619 (vsftpd)  
      Tasks: 1 (limit: 2315)  
     Memory: 592.0K  
       CGroup: /system.slice/vsftpd.service  
               └─2619 /usr/sbin/vsftpd /etc/vsftpd.conf  
  
May 12 11:04:43 linuxconfig systemd[1]: Starting vsftpd FTP server...  
May 12 11:04:43 linuxconfig systemd[1]: Started vsftpd FTP server.  
[linuxconfig.org]$
```

Start or restart the server for configuring.

```
sudo systemctl start vsftpd
```

OR

```
sudo systemctl restart vsftpd
```

Experiment – 13

Aim:-

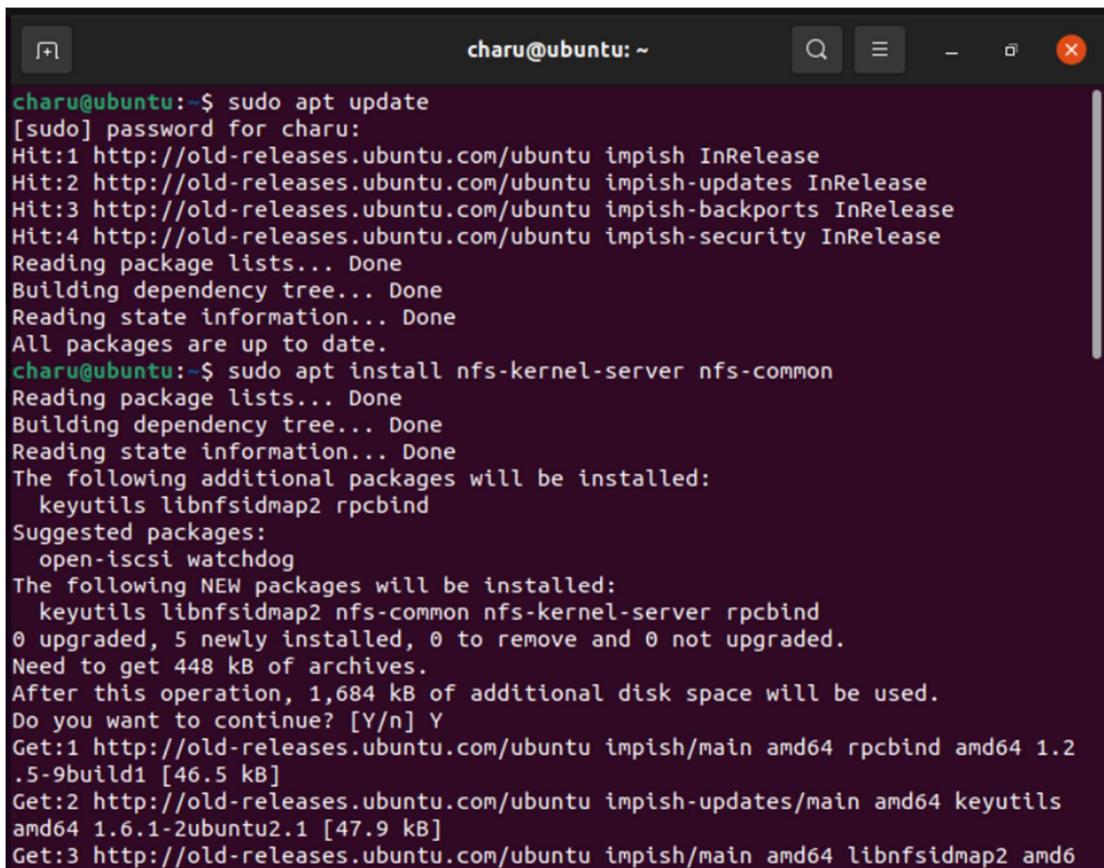
Configure NFS in a network of 3 hosts.

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
```



The screenshot shows a terminal window titled "charu@ubuntu: ~". The terminal displays the following command and its output:

```
charu@ubuntu:~$ sudo apt update  
[sudo] password for charu:  
Hit:1 http://old-releases.ubuntu.com/ubuntu impish InRelease  
Hit:2 http://old-releases.ubuntu.com/ubuntu impish-updates InRelease  
Hit:3 http://old-releases.ubuntu.com/ubuntu impish-backports InRelease  
Hit:4 http://old-releases.ubuntu.com/ubuntu impish-security InRelease  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
All packages are up to date.  
charu@ubuntu:~$ sudo apt install nfs-kernel-server nfs-common  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following additional packages will be installed:  
  keyutils libnfsidmap2 rpcbind  
Suggested packages:  
  open-iscsi watchdog  
The following NEW packages will be installed:  
  keyutils libnfsidmap2 nfs-common nfs-kernel-server rpcbind  
0 upgraded, 5 newly installed, 0 to remove and 0 not upgraded.  
Need to get 448 kB of archives.  
After this operation, 1,684 kB of additional disk space will be used.  
Do you want to continue? [Y/n] Y  
Get:1 http://old-releases.ubuntu.com/ubuntu impish/main amd64 rpcbind amd64 1.2  
.5-9build1 [46.5 kB]  
Get:2 http://old-releases.ubuntu.com/ubuntu impish-updates/main amd64 keyutils  
amd64 1.6.1-2ubuntu2.1 [47.9 kB]  
Get:3 http://old-releases.ubuntu.com/ubuntu impish/main amd64 libnfsidmap2 amd6
```

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.

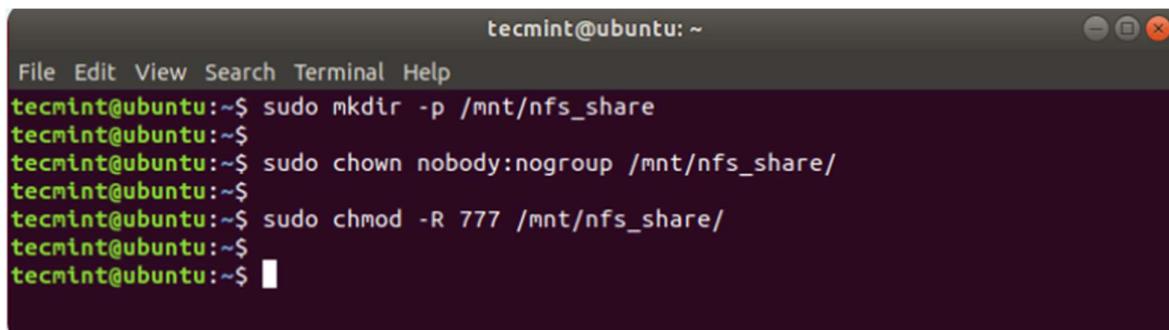
```
sudo mkdir -p /mnt/nfs_share
```

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/
```



```
tecmin@ubuntu:~$ sudo mkdir -p /mnt/nfs_share
tecmin@ubuntu:~$ sudo chown nobody:nogroup /mnt/nfs_share/
tecmin@ubuntu:~$ sudo chmod -R 777 /mnt/nfs_share/
tecmin@ubuntu:~$
```

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)
```

```
tecmint@ubuntu: ~
File Edit View Search Terminal Help
# /etc/exports: the access control list for filesystems which may be exported
#           to NFS clients. See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes      hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4      gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#
/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
```

```
File Edit View Search Terminal Help  
tecmint@ubuntu:~$ sudo exportfs -a  
tecmint@ubuntu:~$ sudo systemctl restart nfs-kernel-server  
tecmint@ubuntu:~$  
tecmint@ubuntu:~$  
tecmint@ubuntu:~$
```

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
```

```
tecmint@ubuntu:~$  
tecmint@ubuntu:~$ sudo ufw enable  
Firewall is active and enabled on system startup  
tecmint@ubuntu:~$  
tecmint@ubuntu:~$ sudo ufw status  
Status: active  
  
To                         Action      From  
--                         ----      --  
2049                       ALLOW      192.168.43.0/24  
  
tecmint@ubuntu:~$
```

Experiment – 14

■ Aim:-

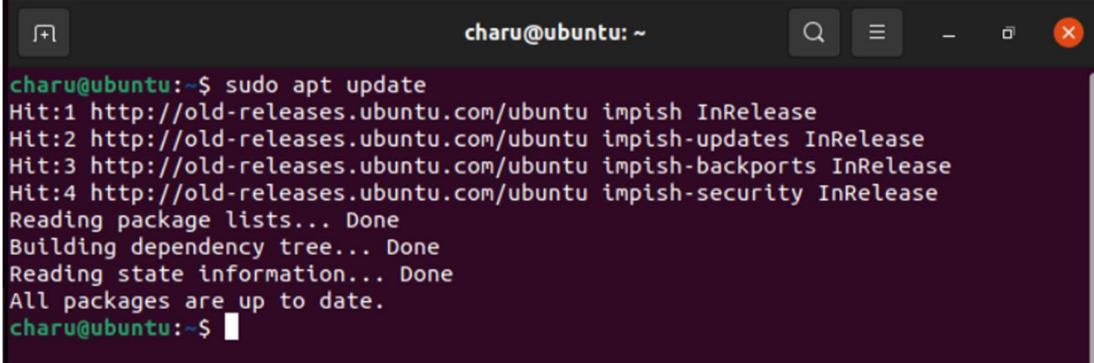
Configure samba server in a LAN.

■ Source Code and Output:-

Steps for installation of Samba server:-

1. To install Samba, we run,

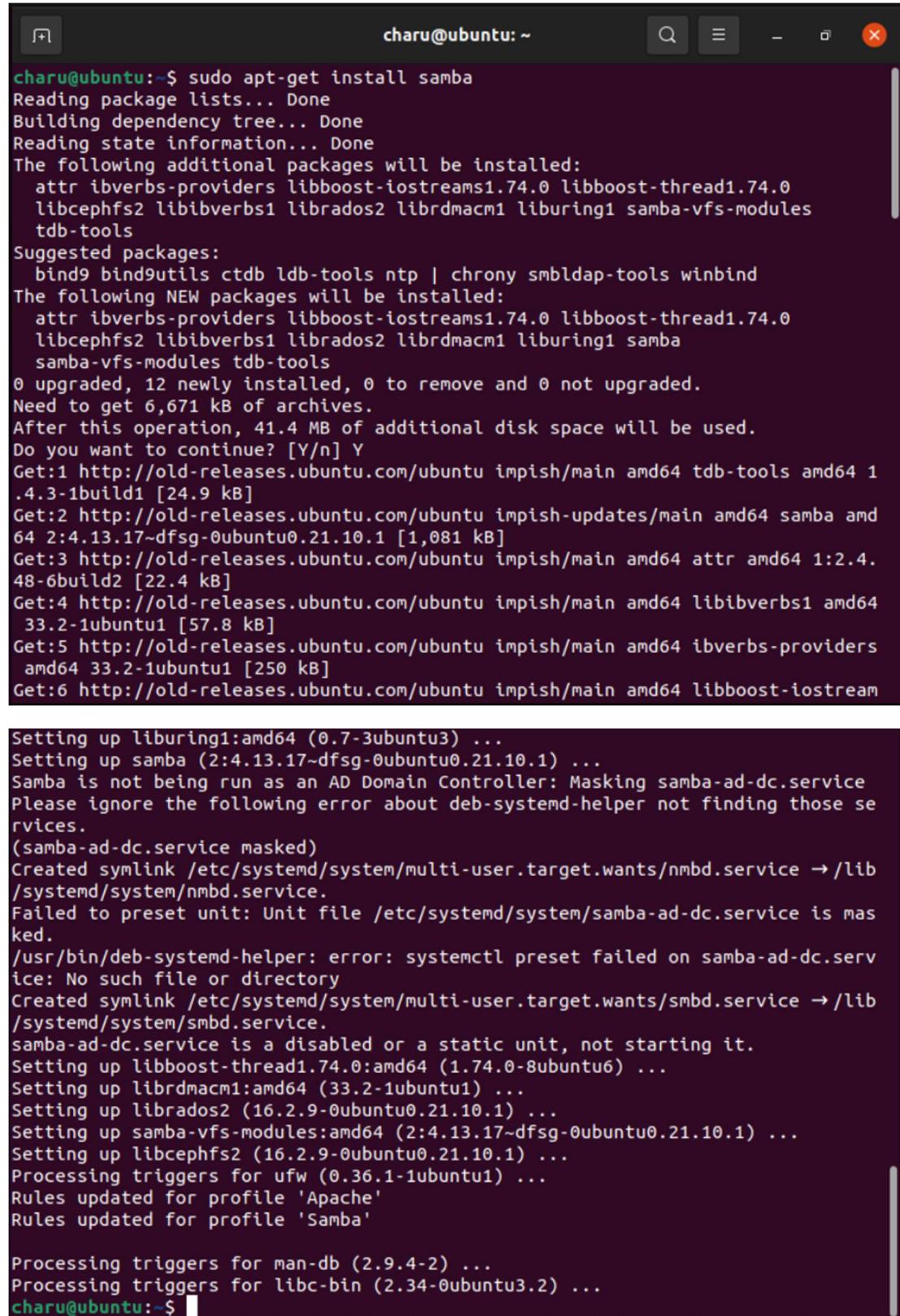
```
sudo apt update
```



The screenshot shows a terminal window titled "charu@ubuntu: ~". The command "sudo apt update" is run, and the output is displayed. The output shows hits from various repositories and indicates that all packages are up to date.

```
charu@ubuntu:~$ sudo apt update
Hit:1 http://old-releases.ubuntu.com/ubuntu impish InRelease
Hit:2 http://old-releases.ubuntu.com/ubuntu impish-updates InRelease
Hit:3 http://old-releases.ubuntu.com/ubuntu impish-backports InRelease
Hit:4 http://old-releases.ubuntu.com/ubuntu impish-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
charu@ubuntu:~$
```

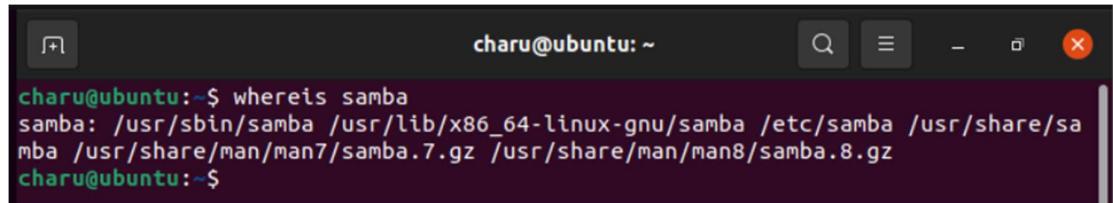
```
sudo apt-get install samba
```



```
charu@ubuntu:~$ sudo apt-get install samba
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  attr ibverbs-providers libboost-iostreams1.74.0 libboost-thread1.74.0
  libcephfs2 libibverbs1 librados2 librdmacm1 liburing1 samba-vfs-modules
  tdb-tools
Suggested packages:
  bind9 bind9utils ctdb ldb-tools ntp | chrony smbdap-tools winbind
The following NEW packages will be installed:
  attr ibverbs-providers libboost-iostreams1.74.0 libboost-thread1.74.0
  libcephfs2 libibverbs1 librados2 librdmacm1 liburing1 samba
  samba-vfs-modules tdb-tools
0 upgraded, 12 newly installed, 0 to remove and 0 not upgraded.
Need to get 6,671 kB of archives.
After this operation, 41.4 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://old-releases.ubuntu.com/ubuntu impish/main amd64 tdb-tools amd64 1
.4.3-1build1 [24.9 kB]
Get:2 http://old-releases.ubuntu.com/ubuntu impish-updates/main amd64 samba amd
64 2:4.13.17~dfsg-0ubuntu0.21.10.1 [1,081 kB]
Get:3 http://old-releases.ubuntu.com/ubuntu impish/main amd64 attr amd64 1:2.4.
48-6build2 [22.4 kB]
Get:4 http://old-releases.ubuntu.com/ubuntu impish/main amd64 libibverbs1 amd64
33.2-1ubuntu1 [57.8 kB]
Get:5 http://old-releases.ubuntu.com/ubuntu impish/main amd64 ibverbs-providers
amd64 33.2-1ubuntu1 [250 kB]
Get:6 http://old-releases.ubuntu.com/ubuntu impish/main amd64 libboost-iostream
Setting up liburing1:amd64 (0.7-3ubuntu3) ...
Setting up samba (2:4.13.17~dfsg-0ubuntu0.21.10.1) ...
Samba is not being run as an AD Domain Controller: Masking samba-ad-dc.service
Please ignore the following error about deb-systemd-helper not finding those se
rvices.
(samba-ad-dc.service masked)
Created symlink /etc/systemd/system/multi-user.target.wants/nmbd.service → /lib
/systemd/system/nmbd.service.
Failed to preset unit: Unit file /etc/systemd/system/samba-ad-dc.service is mas
ked.
/usr/bin/deb-systemd-helper: error: systemctl preset failed on samba-ad-dc.serv
ice: No such file or directory
Created symlink /etc/systemd/system/multi-user.target.wants/smbd.service → /lib
/systemd/system/smbd.service.
samba-ad-dc.service is a disabled or a static unit, not starting it.
Setting up libboost-thread1.74.0:amd64 (1.74.0-8ubuntu6) ...
Setting up librdmacm1:amd64 (33.2-1ubuntu1) ...
Setting up librados2 (16.2.9-0ubuntu0.21.10.1) ...
Setting up samba-vfs-modules:amd64 (2:4.13.17~dfsg-0ubuntu0.21.10.1) ...
Setting up libcephfs2 (16.2.9-0ubuntu0.21.10.1) ...
Processing triggers for ufw (0.36.1-1ubuntu1) ...
Rules updated for profile 'Apache'
Rules updated for profile 'Samba'

Processing triggers for man-db (2.9.4-2) ...
Processing triggers for libc-bin (2.34-0ubuntu3.2) ...
charu@ubuntu:~$
```

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

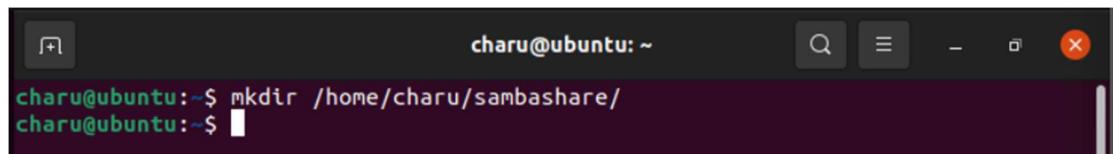


```
charu@ubuntu:~$ whereis samba
samba: /usr/sbin/samba /usr/lib/x86_64-linux-gnu/samba /etc/samba /usr/share/sa
mba /usr/share/man/man7/samba.7.gz /usr/share/man/man8/samba.8.gz
charu@ubuntu:~$
```

3. Setting up Samba

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

```
mkdir /home/username/sambashare/
```

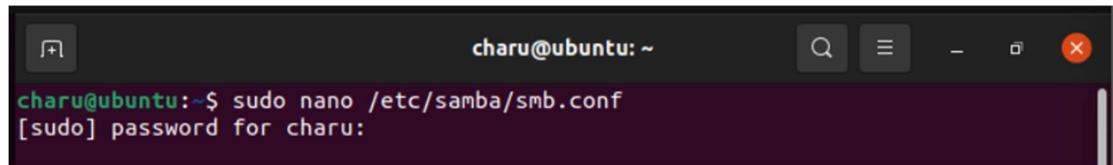


```
charu@ubuntu:~$ mkdir /home/charu/sambashare/
charu@ubuntu:~$
```

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:

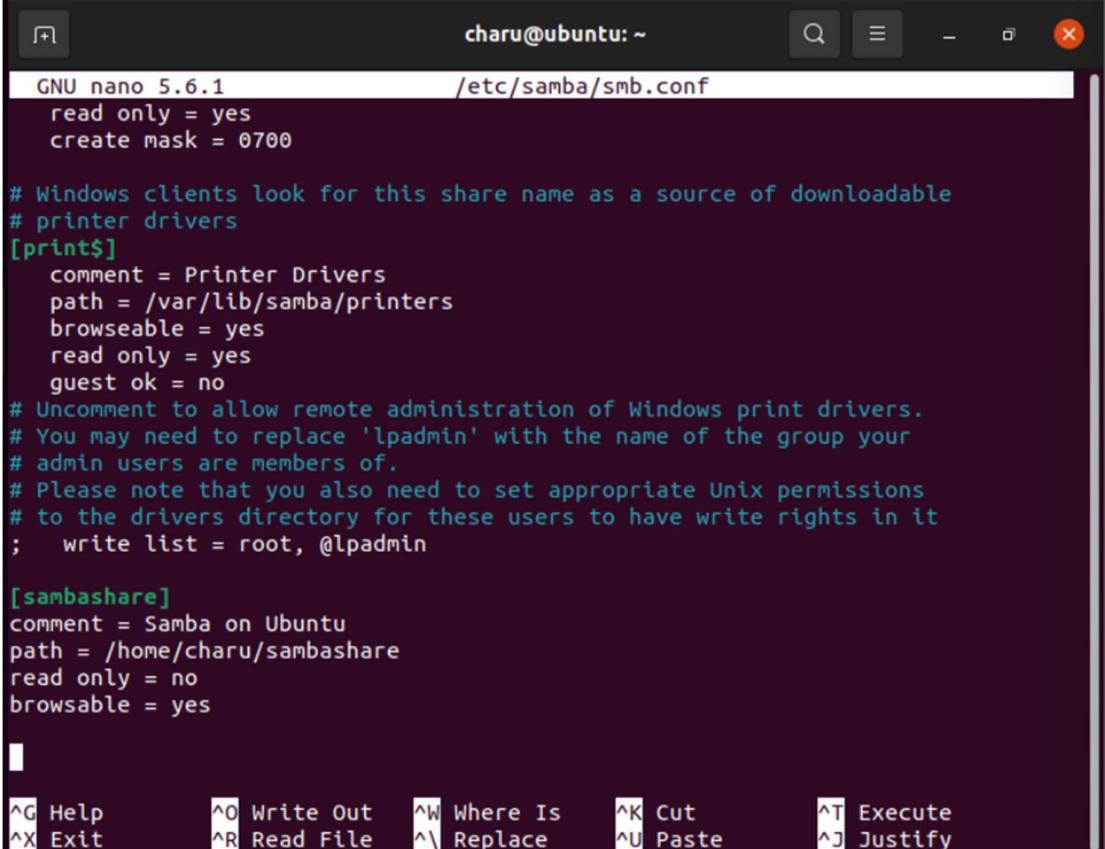
```
sudo nano /etc/samba/smb.conf
```



```
charu@ubuntu:~$ sudo nano /etc/samba/smb.conf
[sudo] password for charu:
```

At the bottom of the file,

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



The screenshot shows a terminal window titled "charu@ubuntu: ~". The window title bar also displays "GNU nano 5.6.1" and the file path "/etc/samba/smb.conf". The main area of the terminal shows the configuration file content. It includes sections for "[print\$]", "[sambashare]", and "[sambashare]" again. The configuration parameters shown include "comment", "path", "read only", "browsable", and "create mask". The file ends with a semi-colon and a "write list" entry. At the bottom of the terminal window, there is a menu bar with options like Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, and Justify.

```
GNU nano 5.6.1          /etc/samba/smb.conf
read only = yes
create mask = 0700

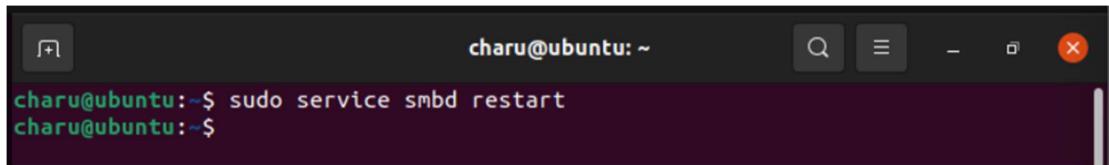
# Windows clients look for this share name as a source of downloadable
# printer drivers
[print$]
comment = Printer Drivers
path = /var/lib/samba/printers
browseable = yes
read only = yes
guest ok = no
# Uncomment to allow remote administration of Windows print drivers.
# You may need to replace 'lpadmin' with the name of the group your
# admin users are members of.
# Please note that you also need to set appropriate Unix permissions
# to the drivers directory for these users to have write rights in it
;   write list = root, @lpadmin

[sambashare]
comment = Samba on Ubuntu
path = /home/charu/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:

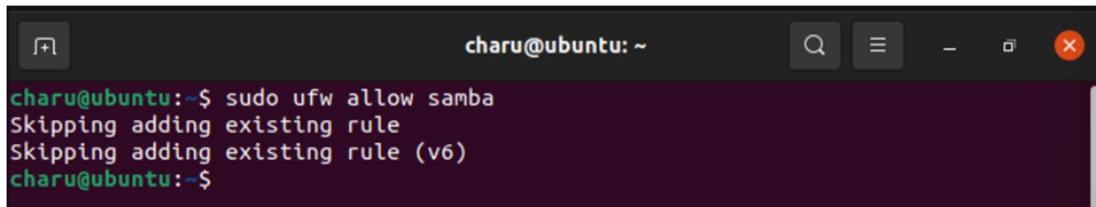
```
sudo service smbd restart
```



```
charu@ubuntu:~$ sudo service smbd restart
charu@ubuntu:~$
```

6. Update the firewall rules to allow Samba traffic:

```
sudo ufw allow samba
```

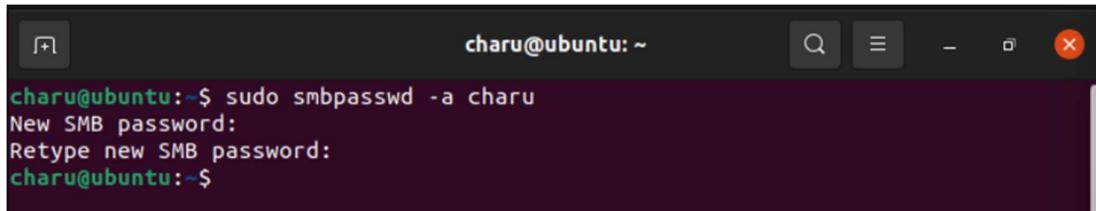


```
charu@ubuntu:~$ sudo ufw allow samba
Skipping adding existing rule
Skipping adding existing rule (v6)
charu@ubuntu:~$
```

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:

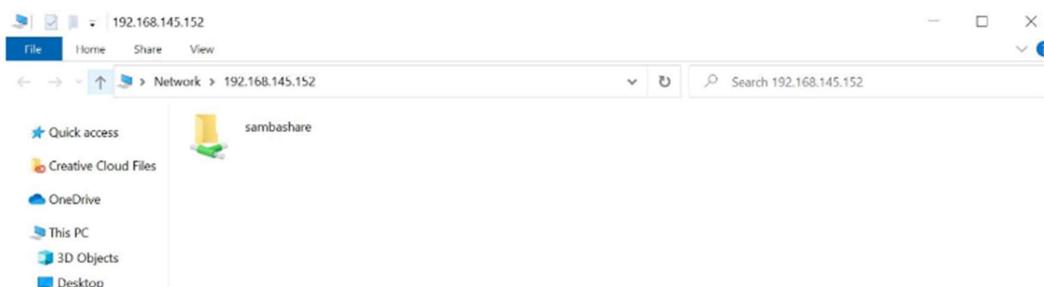
```
sudo smbpasswd -a username
```



```
charu@ubuntu:~$ sudo smbpasswd -a charu
New SMB password:
Retype new SMB password:
charu@ubuntu:~$
```

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
```

```
gamer@linuxhint:~$ sudo apt update
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Ign:2 http://ppa.launchpad.net/mc3man/trusty-media/ubuntu focal InRelease
Hit:3 http://pk.archive.ubuntu.com/ubuntu focal InRelease
Get:4 http://pk.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
```

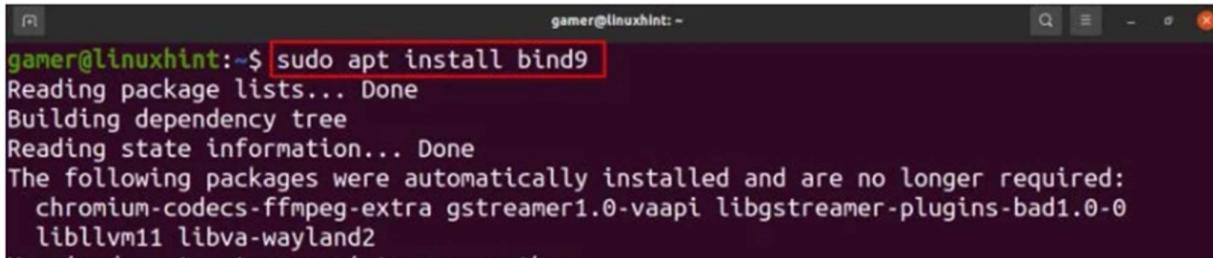
2. Install the DNS server by using the command bind9:

```
sudo apt install bind9
```

```
gamer@linuxhint:~$ sudo apt install bind9
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi libgstreamer-plugins-bad1.0-0
  libllvm11 libva-wayland2
```

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

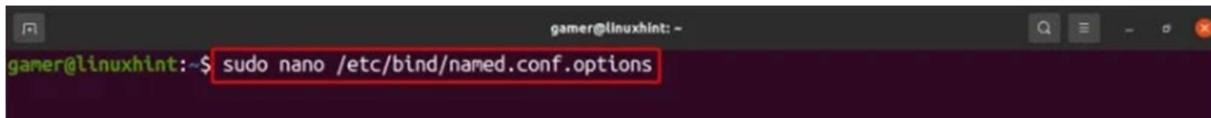
```
sudo apt install dnsutils
```



```
gamer@linuxhint:~$ sudo apt install bind9
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi libgstreamer-plugins-bad1.0-0
  libl LLVM11 libva-wayland2
Use 'apt autoremove' to remove them
```

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
```



```
gamer@linuxhint:~$ sudo nano /etc/bind/named.conf.options
```

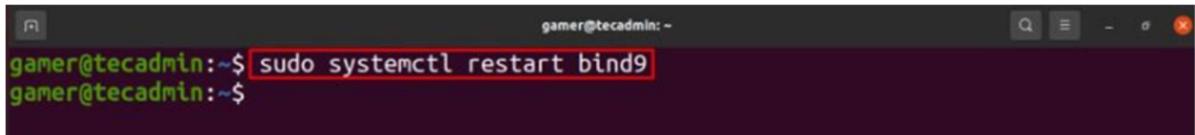
Replace the following text in the editor, 8.8.8.8 is Google's DNS:

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

Now quit after saving it

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

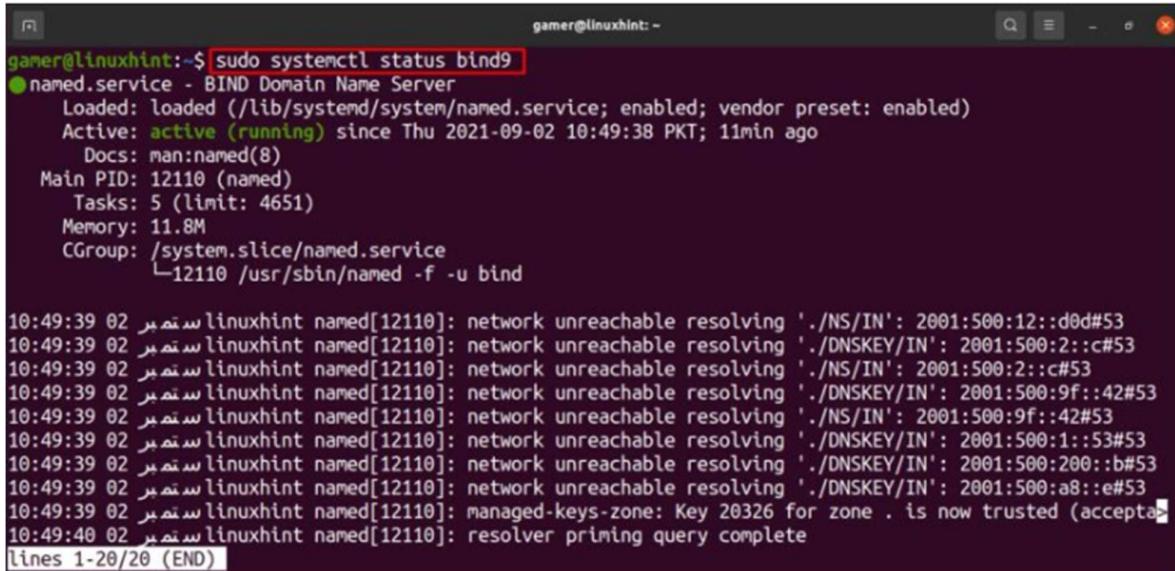
```
sudo systemctl restart bind9
```



```
gamer@tecadmin:~$ sudo systemctl restart bind9
gamer@tecadmin:~$
```

5. Check the status of bind9

```
sudo systemctl status bind9
```

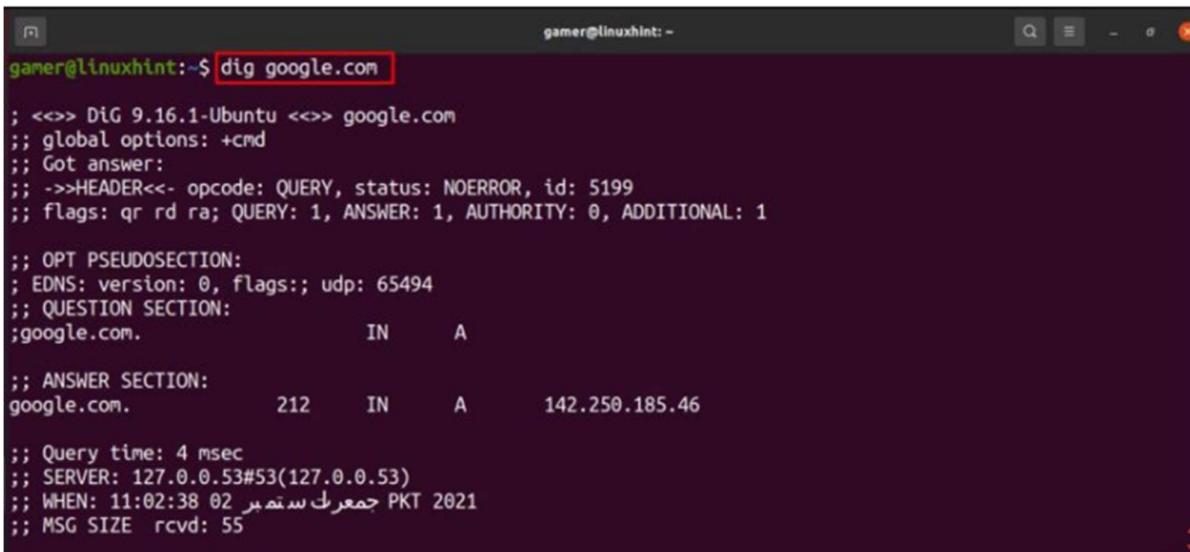


```
gamer@linuxhint:~$ sudo systemctl status bind9
● named.service - BIND Domain Name Server
  Loaded: loaded (/lib/systemd/system/named.service; enabled; vendor preset: enabled)
  Active: active (running) since Thu 2021-09-02 10:49:38 PKT; 11min ago
    Docs: man:named(8)
   Main PID: 12110 (named)
      Tasks: 5 (limit: 4651)
     Memory: 11.8M
        CPU: 0.000 CPU(s) since start
       CGroup: /system.slice/named.service
               └─12110 /usr/sbin/named -f -u bind

10:49:39 02 سنتپر linuxhint named[12110]: network unreachable resolving './NS/IN': 2001:500:12::d0d#53
10:49:39 02 سنتپر linuxhint named[12110]: network unreachable resolving './DNSKEY/IN': 2001:500:2::c#53
10:49:39 02 سنتپر linuxhint named[12110]: network unreachable resolving './NS/IN': 2001:500:2::c#53
10:49:39 02 سنتپر linuxhint named[12110]: network unreachable resolving './DNSKEY/IN': 2001:500:9f::42#53
10:49:39 02 سنتپر linuxhint named[12110]: network unreachable resolving './NS/IN': 2001:500:9f::42#53
10:49:39 02 سنتپر linuxhint named[12110]: network unreachable resolving './DNSKEY/IN': 2001:500:1::53#53
10:49:39 02 سنتپر linuxhint named[12110]: network unreachable resolving './DNSKEY/IN': 2001:500:200::b#53
10:49:39 02 سنتپر linuxhint named[12110]: network unreachable resolving './DNSKEY/IN': 2001:500:a8::e#53
10:49:39 02 سنتپر linuxhint named[12110]: managed-keys-zone: Key 20326 for zone . is now trusted (accepted)
10:49:40 02 سنتپر linuxhint named[12110]: resolver priming query complete
lines 1-28/20 (END)
```

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

```
dig google.com
```



```
gamer@linuxhint:~$ dig google.com

; <>> DiG 9.16.1-Ubuntu <>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5199
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;google.com.           IN      A

;; ANSWER SECTION:
google.com.        212     IN      A      142.250.185.46

;; Query time: 4 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: 11:02:38 02 سنتپر PKT 2021
;; MSG SIZE rcvd: 55
```

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 kB]
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
```

```
devconnected@devconnected:~$ sudo systemctl status sshd
● ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: >
  Active: active (running) since Sun 2020-07-19 11:24:22 CEST; 3min 58s ago
    Docs: man:sshd(8)
          man:sshd_config(5)
   Main PID: 4473 (sshd)
     Tasks: 1 (limit: 9484)
    Memory: 1.1M
      CGroup: /system.slice/ssh.service
              └─4473 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

juil. 19 11:24:21 devconnected systemd[1]: Starting OpenBSD Secure Shell serve>
juil. 19 11:24:22 devconnected sshd[4473]: Server listening on 0.0.0.0 port 22.
juil. 19 11:24:22 devconnected sshd[4473]: Server listening on :: port 22.
juil. 19 11:24:22 devconnected systemd[1]: Started OpenBSD Secure Shell server.
```

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^
```

```
devconnected@devconnected:~$ netstat -tulpn | grep 22
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
tcp        0      0 0.0.0.0:22                    0.0.0.0:*          LISTEN
-
tcp6       0      0 :::22                         ::::*             LISTEN
-
```

4. To enable SSH connections on your host, run the following command

```
sudo ufw allow ssh
```

```
devconnected@devconnected:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
```

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/systemd-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: sshd.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.

```
GNU nano 4.8          /etc/ssh/sshd_config      Modified
# $OpenBSD: sshd_config,v 1.103 2018/04/09 20:41:22 tj Exp $

# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/bin:/bin:/usr/sbin:/sbin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

Port 2222
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
```

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

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

```
devconnected@devconnected:~$ sudo systemctl restart sshd
devconnected@devconnected:~$ sudo systemctl status sshd
● ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: >
  Active: active (running) since Sun 2020-07-19 12:05:13 CEST; 3s ago
    Docs: man:sshd(8)
          man:sshd_config(5)
   Process: 7671 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
 Main PID: 7691 (sshd)
   Tasks: 1 (limit: 9484)
  Memory: 1.1M
    CGroup: /system.slice/ssh.service
            └─7691 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

juil. 19 12:05:13 devconnected systemd[1]: Starting OpenBSD Secure Shell server>
juil. 19 12:05:13 devconnected sshd[7691]: Server listening on 0.0.0.0 port 22>
juil. 19 12:05:13 devconnected sshd[7691]: Server listening on :: port 2222.
juil. 19 12:05:13 devconnected systemd[1]: Started OpenBSD Secure Shell server.
lines 1-16/16 (END)
```

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 established.
ECDSA key fingerprint is SHA256:cTQt0Z1Yb0xq6tV8ewdyPzEk0llX8RpKI3iNgK4bBc8.
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 hosts.
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.

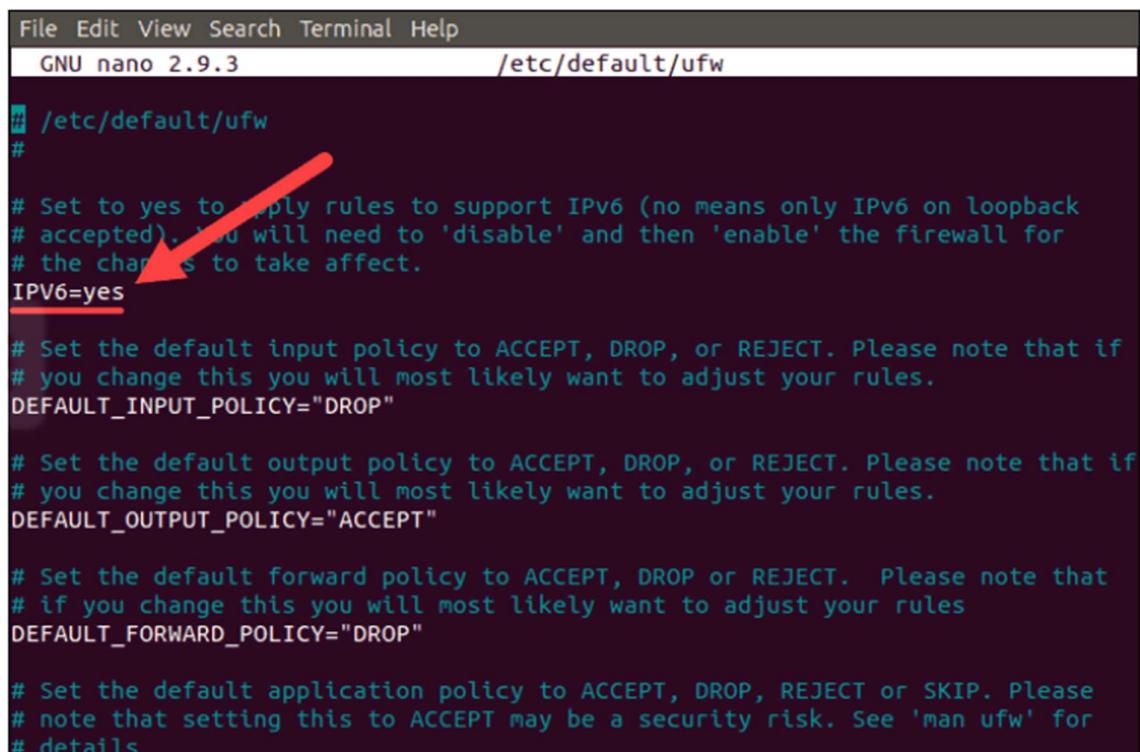
Source Code and Output:-

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
```



```
File Edit View Search Terminal Help
GNU nano 2.9.3                               /etc/default/ufw

# /etc/default/ufw
#
# Set to yes to apply rules to support IPv6 (no means only IPv6 on loopback
# accepted). You will need to 'disable' and then 'enable' the firewall for
# the changes to take affect.
IPV6=yes

# Set the default input policy to ACCEPT, DROP, or REJECT. Please note that if
# you change this you will most likely want to adjust your rules.
DEFAULT_INPUT_POLICY="DROP"

# Set the default output policy to ACCEPT, DROP, or REJECT. Please note that if
# you change this you will most likely want to adjust your rules.
DEFAULT_OUTPUT_POLICY="ACCEPT"

# Set the default forward policy to ACCEPT, DROP or REJECT. Please note that
# if you change this you will most likely want to adjust your rules
DEFAULT_FORWARD_POLICY="DROP"

# Set the default application policy to ACCEPT, DROP, REJECT or SKIP. Please
# note that setting this to ACCEPT may be a security risk. See 'man ufw' for
# details
```

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
```

```
kb@phoenixNAP:~$ sudo ufw allow https
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
443/tcp                     ALLOW IN    Anywhere
22/tcp (v6)                 ALLOW IN    Anywhere (v6)
80/tcp (v6)                 ALLOW IN    Anywhere (v6)
443/tcp (v6)                ALLOW IN    Anywhere (v6)
```

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
```