**Docker Installation**

1. for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker containerd runc; do sudo apt-get remove $pkg; done
2. sudo apt-get update
3. sudo apt-get install ca-certificates curl gnupg
4. sudo install -m 0755 -d /etc/apt/keyrings
5. curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
6. sudo chmod a+r /etc/apt/keyrings/docker.gpg
7. echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \

$(. /etc/os-release && echo "$VERSION\_CODENAME") stable" | \

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

1. sudo apt-get update
2. sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

Verify that the Docker Engine installation is successful by running the hello-world image.

1. sudo docker run hello-world

**Create Docker image:**  
1. Go the the directory where the application files exists.

2. Create docker file

gedit Dockerfile

3. Write this in Dockerfile

FROM ubuntu

COPY . .

CMD ["echo","Hello,there"]

It means that the ubuntu is used as a VM to create the docker image.

COPY . . means it copies the entire content of the current directory to the current directory of the docker images. So, this copies everything present in our directory to the VM

CMD is used to give any instructions in the docker file

4. To build Docker image

Sudo docker build -t mydocker .

5. To run docker image

Sudo docker run mydocker

6. To see all docker images

Sudo docker images

7. To open terminal of the docker image

Sudo docker run -it mydocker bash

It means interactive mode

**If we want to build our dockerfile on a base images eg. nginx**

1. Write this in dockerfile

It shows the location in the images where to copy

1. Then build Docker image

Sudo docker build -t mydocker .

1. Run docker container from docker image

Sudo docker run -d -p 8080:80 mydocker

8080 is the host machine’s port from where we can access our application using web browser. It maps port 8080 in host to port 80 in container.

1. We can access our application from the web browser

localhost:8080

And download our file by localhost:8080/1.py

**Docker Experiment example**

**Create a web application with simple web page containing login details and create a docker image of the application.(Use Apache Web server) Run the Docker container from recently created image and run the container at port number 80 in host system.**

1. Create HTML Form
2. Create docker file

FROM httpd:latest

COPY index.html /usr/local/apache2/htdocs/

1. Build docker image

docker build -t simple-web-app .

1. Run docker container

docker run -d -p 80:80 simple-web-app

1. Access the form from

localhost:80

**FTP**

1. sudo apt update
2. sudo apt install vsftpd
3. Sudo service vsftpd status
4. Sudo nano /etc/vsftpd.conf

Uncomment write\_enable=YES

ADD

user\_sub\_token=$USER

local\_root=/home/$USER/ftp

pasv\_min\_port=10000

pasv\_max\_port=10100

userlist\_enable=YES

userlist\_file=/etc/vsftpd.userlist

userlist\_deny=NO

1. sudo ufw allow from any to any port 20,21,10000:10100 proto tcp
2. sudo adduser ftpuser1

password : abcd

1. sudo mkdir /home/ftpuser1/ftp
2. sudo chown nobody:nogroup /home/ftpuser1/ftp
3. sudo chmod a-w /home/ftpuser1/ftp
4. sudo mkdir /home/ftpuser1/ftp/upload
5. sudo chown ftpuser1:ftpuser1 /home/ftpuser1/ftp/upload
6. echo "My FTP Server" | sudo tee /home/ftpuser1/ftp/upload/demo.txt
7. sudo ls -la /home/ftpuser1/ftp
8. echo "Adwait" | sudo tee -a /etc/vsftpd.userlist
9. sudo systemctl restart vsftpd
10. ifconfig

Take first ip address (inet ke baju ka)

1. Go to Other Locations on the PC and write ftp://your-ip-address
2. Login from the created ftpuser1
3. We see the files of the ftpuser1

**Telnet**

1. In one machine/terminal, configure the server for telnet

sudo apt install telnetd xinetd

1. Check if it is running

sudo systemctl status xinetd.service

1. If is not active/running

sudo systemctl start xinetd.service

1. Create Telnet file

sudo nano /etc/xinetd.d/telnet

Write below in the file

service telnet

{

disable = no

flags = REUSE

socket\_type = stream

wait = no

user = root

server = /usr/sbin/in.telnetd

log\_on\_failure += USERID

}

1. Then save and close the file and restart xinetd.service as follows:

sudo systemctl restart xinetd.service

1. Telnet server uses port 23 for listening to the incoming connections. Therefore, you will need to open this port in your firewall. Run the command below to do so :

sudo ufw allow 23

1. Note the ip address ->10.10.13.226 //in my case
2. Open new terminal which would be the client

Now you can connect to your Telnet server from another machine (where the Telnet client is installed). On your client machine, use the following command syntax to connect to the Telnet server:

telnet 10.10.13.226

**NFS**

Letters a,b,c,d tell us the order of executing steps

Open 2 separate terminals for client and server

**A Server**

1.sudo apt update

2.sudo apt install nfs-kernel-server

3.sudo mkdir -p /mnt/nfs\_share

4.sudo chown -R nobody:nogroup /mnt/nfs\_share/

5.sudo chmod 777 /mnt/nfs\_share/

6.sudo nano /etc/exports

7.sudo exportfs -a

8.sudo systemctl restart nfs-kernel-server

9.sudo ufw allow from 10.10.13.133 to any port nfs

10.udo ufw enable

11.sudo ufw status

**B client**

1.sudo apt update

2.sudo apt install nfs-common

3.sudo mkdir -p /mnt/nfs\_clientshare

4.sudo mount 10.10.13.133:/mnt/nfs\_share /mnt/nfs\_clientshare

**C server**

1.cd /mnt/nfs\_share

2.touch file1.txt file2.txt file3.txt

**D client**

1.ls -l /mnt/nfs\_clientshare/

**SVN**

SVN stands for Subversion. It is an open-source centralized version control system written in Java, licensed under Apache. Software developers use Subversion to maintain current and historical versions of files such as source code.

# **Step 1: Install Apache2**

* sudo apt update
* sudo apt install apache2 apache2-utils

**We have installed Apache2 now let’s start and enable it.**

* sudo systemctl start apache2.service
* sudo systemctl enable apache2.service

**We have successfully set-up and enable the HTTP web server. Let’s install SVN now.**

# **Step 2: Install SVN**

* sudo apt-get install subversion libapache2-mod-svn subversion-tools libsvn-dev

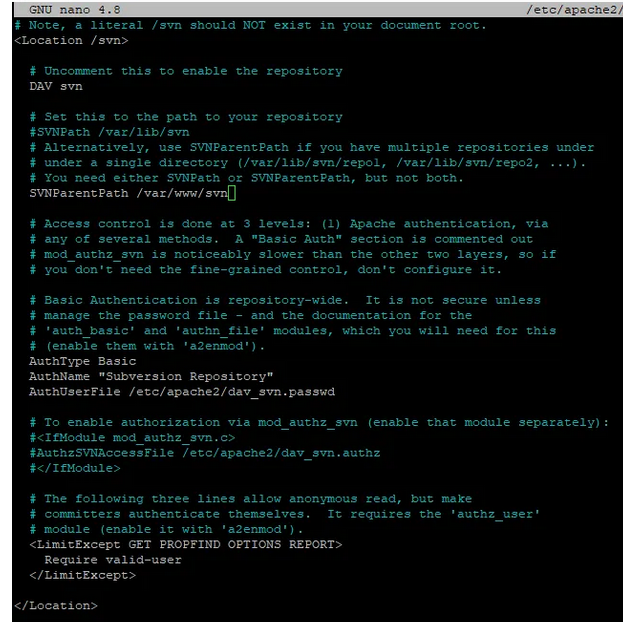
SVN and all dependencies are installed. Now enable Apache2 modules to run SVN to function.

* sudo a2enmod dav
* sudo a2enmod dav\_svn
* sudo service apache2 restart

# **Step 3: Configure Apache2 with SVN**

* sudo nano /etc/apache2/mods-enabled/dav\_svn.conf

Make mentioned Changes/un-comment lines in the file.



**Let’s Create Repository Now**

* sudo mkdir /var/www/svn
* sudo svnadmin create /var/www/svn/project
* sudo chown -R www-data:www-data /var/www/svn
* sudo chmod -R 775 /var/www/svn

# **Step 4: Create SVN User Accounts**

Use the below command to create a new SVN user(admin).

* sudo htpasswd -cm /etc/apache2/dav\_svn.passwd admi**n**

If you wish to create more users then use the below command

* sudo htpasswd -m /etc/apache2/dav\_svn.passwd awais

We have successfully Installed and configure SVN let’s restart the Apache2 server and Test it. Restart Apache2 server with the below command.

* sudo systemctl restart apache2.service

# **Let’s Test It**

Open your browser and write the following in your URL bar.

* localhost/svn/project

{to remove anything from www folder

**sudo rm -R** /var/www/wordpress/wp-content/themes/myFolder/\*

-R to recursively remove anything inside it (and deeper).

This also removes files (not just directories).

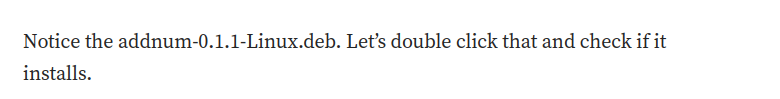
ex.sudo rm -R /var/www/svn/project

}

**Debian Package**

<https://karthikkalyanaraman.medium.com/creating-debian-packages-cmake-e519a0186e87>

Note: build the directory structure with extreme care

**after double clicking extract the data.tar.gz file and u’ll be able to see home folder inside that folder there will bw app**

**Wordpress**