

## Task 1

Firstly to install KVM, I need to check if it was supported in my PC or not so I had to install cpu-checker and run the kvm-ok command

```
debjyotisarkar@sumonta-22341019:~$ sudo apt install cpu-checker
[sudo] password for debjyotisarkar:
Reading package lists... Done
Building dependency tree... Done
```

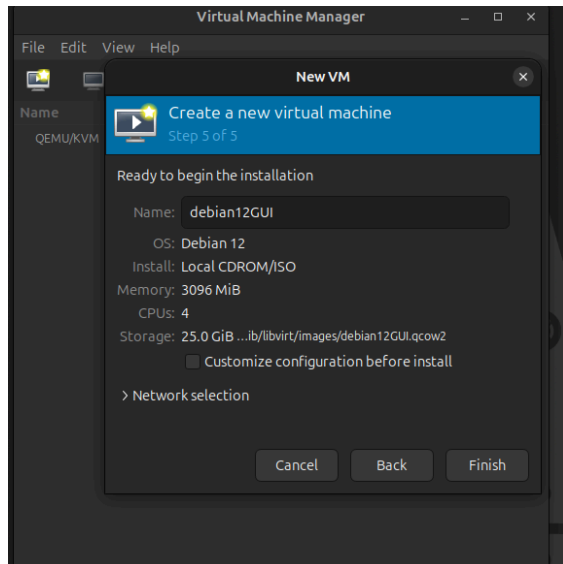
```
debjyotisarkar@sumonta-22341019:~$ kvm-ok
INFO: /dev/kvm exists
KVM acceleration can be used
debjyotisarkar@sumonta-22341019:~$
```

As KVM is supported, we need to install these packages for KVM and other things needed to run virtual machines

```
debjyotisarkar@sumonta-22341019:~$ sudo apt install qemu-kvm libvirt-daemon l
libvirt-clients bridge-utils virt-manager
Reading package lists... Done
Building dependency tree... Done
```

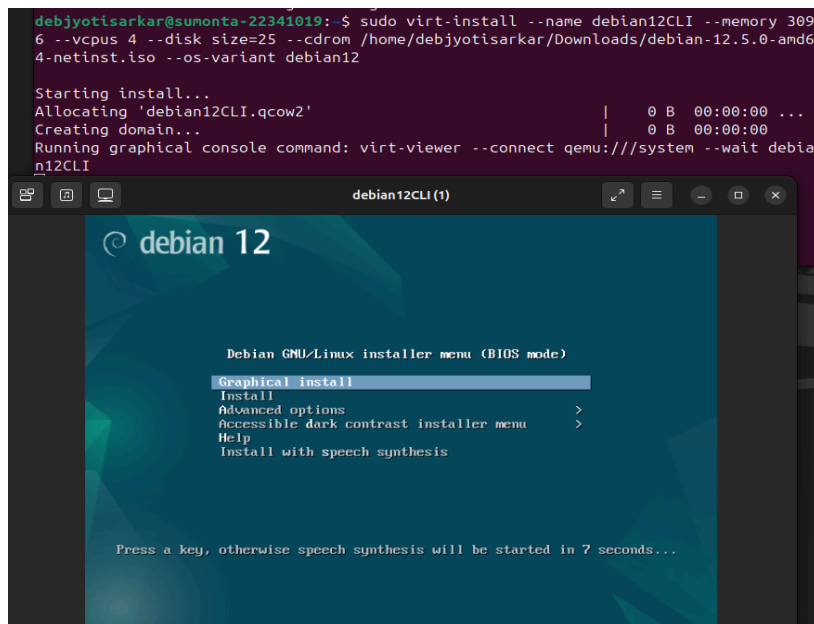
## Task 2

To create a new virtual machine using GUI, we need to open Virtual Machine Manager and add a new virtual machine and enter the name and the iso file. Following that, we just need to go through the steps to have a running VM. Here, we can assign however much CPU cores, RAM, storage to the VM as we wish.



### Task 3

To set up a VM using CLI, we need to do it using `virt-install`. The component of this command do the following: `name` - sets the name for the VM, `memory` - allocates the defined number of RAM in MB, `vcpu` - defines the CPU cores, `disk-size` - allocates the storage of the VM in GBs, `cdrom` - points to the directory where the iso file for the installation is stored, `os-variant` - defines the OS that is being used in the VM. Similar to the GUI method, after running this command we just need to go through the steps to install the VM.



## Task 5

To share a folder between guest and host, firstly I needed to install virtiofsd then edit the XML file of the VM

```
root@sumonta-22341019:/home/debjyotisarkar# sudo apt install virtiofsd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
```

```
debjyotisarkar@sumonta-22341019:~$ virsh edit debian12CLI
```

In the XML file added this part of code for allowing access to the shared folder.

```
<filesystem type='mount' accessmode='passthrough'>
  <driver type='virtiofs' />
  <source dir='/home/debjyotisarkar/Desktop/shared' />
  <target dir='shared' />
  <address type='pci' domain='0x0000' bus='0x07' slot='0x00' function='0x0' />
</filesystem>
```

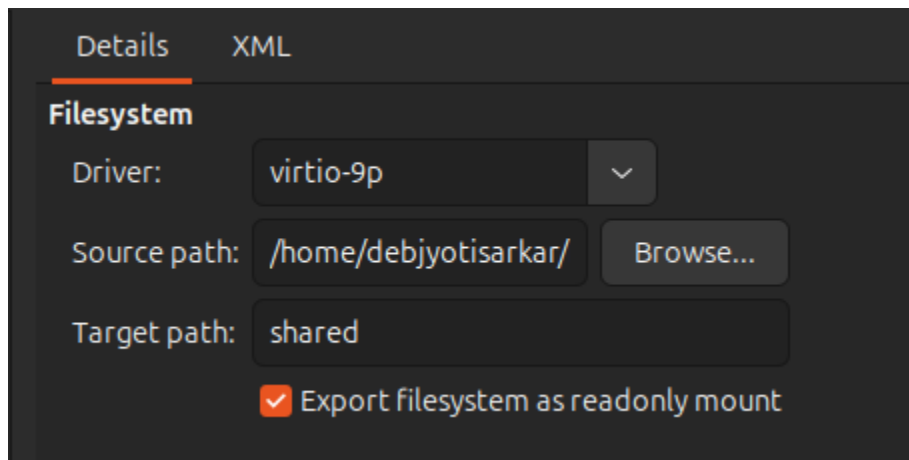
I created a new directory in the host machine and enabled read-write permission to that directory.

```
debjyotisarkar@sumonta-22341019:~$ sudo mkdir Desktop/shared
[sudo] password for debjyotisarkar:
debjyotisarkar@sumonta-22341019:~$ sudo chmod 777 Desktop/shared
debjyotisarkar@sumonta-22341019:~$
```

I started a virtiofs Daemon with the correct path to the shared folder, it will act as a socket for communication.

```
debjyotisarkar@sumonta-22341019:~$ sudo /usr/lib/qemu/virtiofsd --socket-path=/tmp/vhostqemu -o source=/home/debjyotisarkar/Desktop/shared -o cache=always -o no_posix_lock
[2024-06-29T09:55:51Z WARN virtiofsd] Use of deprecated option format '-o': Please specify options without it (e.g., '--cache auto' instead of '-o cache=auto')
[2024-06-29T09:55:51Z INFO virtiofsd] Waiting for vhost-user socket connection.
..
█
```

In the details -> add hardware -> select file system of the VM, I added both the source files and the target path.



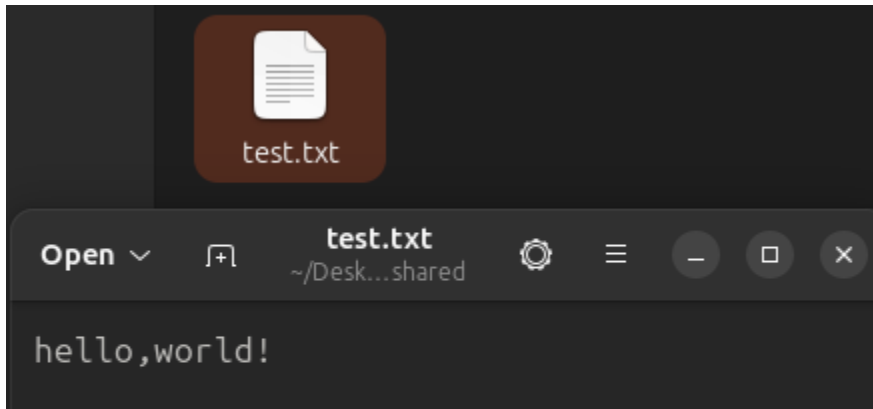
The screenshot shows the 'Details' tab of a VM configuration interface. Under the 'Filesystem' section, the 'Driver' is set to 'virtio-9p'. The 'Source path' is '/home/debjyotisarkar/' with a 'Browse...' button next to it. The 'Target path' is 'shared'. At the bottom, the checkbox 'Export filesystem as readonly mount' is checked.

Now, in the VM, I got root access using `sudo -s` and made a directory in the VM and mounted it and the shared folder is ready and working.

```
debjyotisarkar@sumonta-22341019:~$ sudo -s
[sudo] password for debjyotisarkar:
root@sumonta-22341019:/home/debjyotisarkar# sudo mkdir -p /mnt/shared
root@sumonta-22341019:/home/debjyotisarkar# sudo mount -t virtiofs shared /mnt/shared
```

To check if the folder is actually working, I opened a text file in the VM and it appeared the same in the host machine as well.

```
root@sumonta-22341019:/# echo "hello,world!" > /mnt/shared/test.txt
root@sumonta-22341019:/#
```

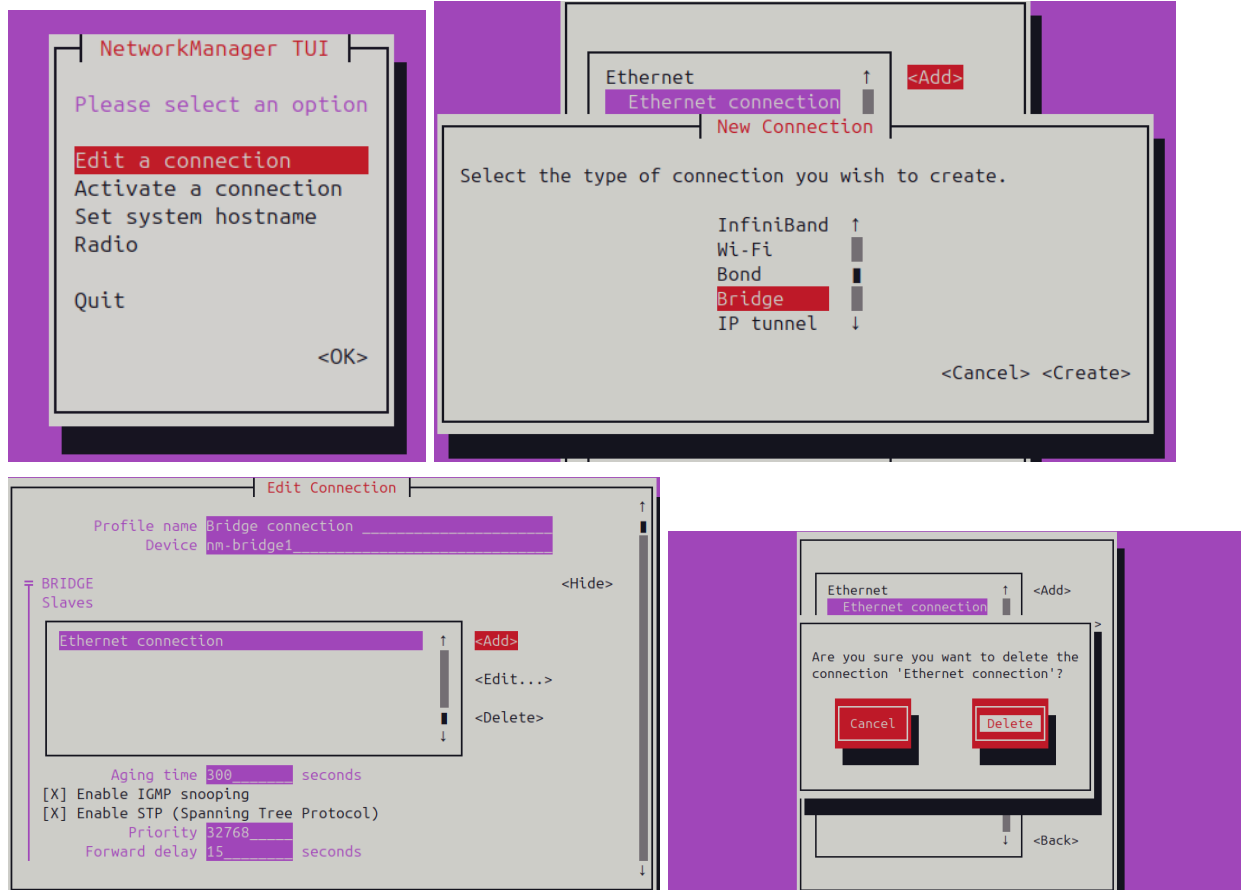


## Task 6

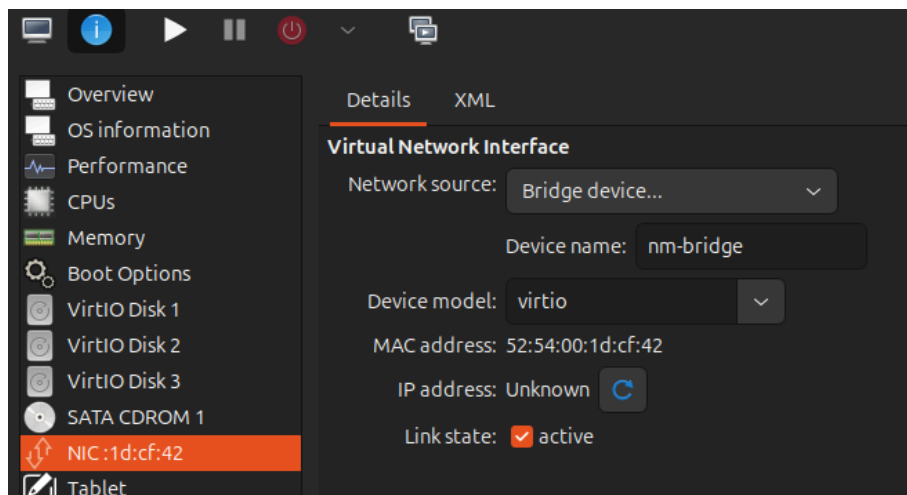
To access the phone's files in the guest OS, I used KDE connect. As we know, whenever we create a VM the host OS allocates a virtual IP using NAT for the guest OS and in order for the guest OS to access the files of the phone using the local network the network type had to be changed from NAT to bridge. In order to operate a bridge, we need to open nmtui first.

```
debjyotisarkar@sumonta-22341019:~$ sudo nmtui
[sudo] password for debjyotisarkar:
```

In nmtui, we need to create a bridge and add the current ethernet connection to the bridge. Following that, we also need to delete/deactivate the ethernet connection from the server in order to activate the bridge.



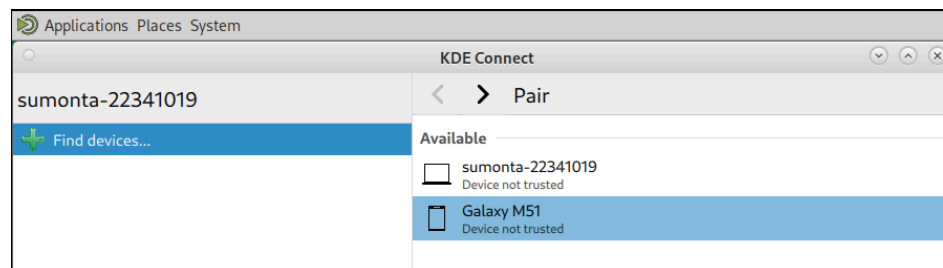
In the details of the VM, we need to change the network source to the bridge device.



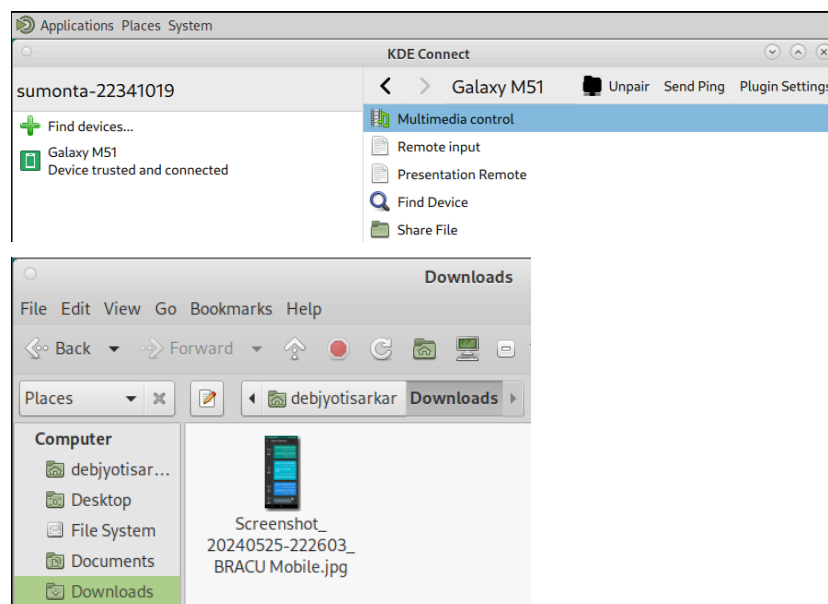
In the guest OS, we need to install KDE connect, it also needs to be installed on the phone from the app store.

```
debjyotisarkar@sumonta-22341019:~$ sudo apt install kdeconnect
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

Inside the KDE connect app, it will show all the available devices the current network. In my case, my host machine and my phone were the two devices available.

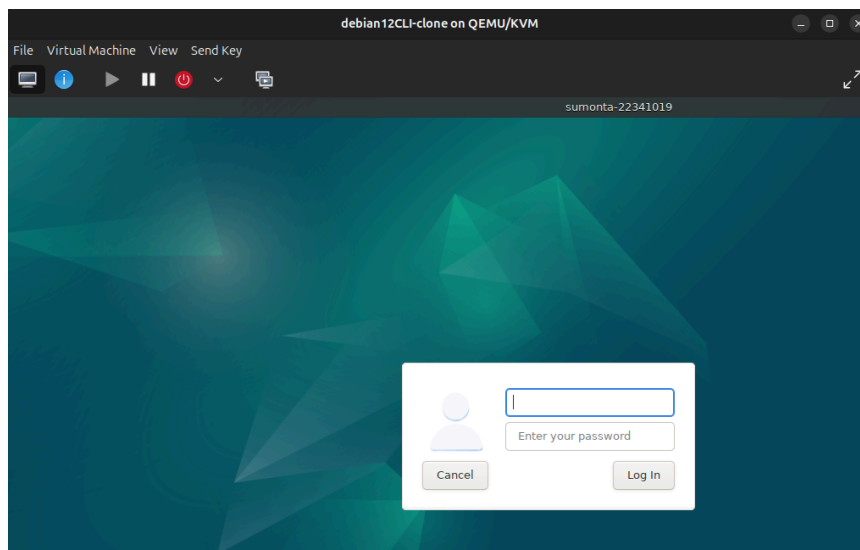
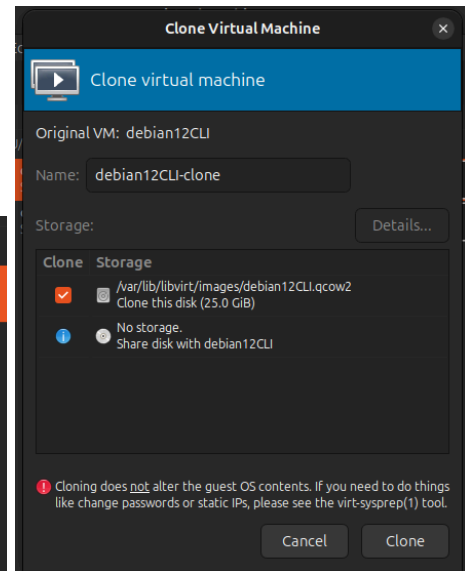
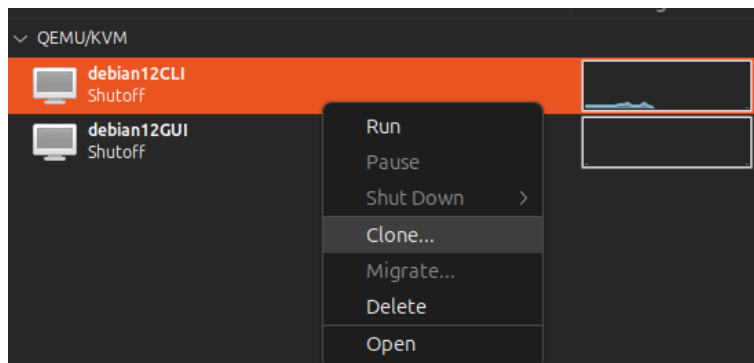


In the app, through the share file option, I can share any file from the phone to the VM or vice versa.



## Task 7

To clone the VM using GUI, it's just as simple as right clicking the VM and clicking clone and naming the new VM.



To clone the VM using CLI, we first need to install virtinst

```
debjyotisarkar@sumonta-22341019:~$ sudo apt-get install virtinst
[sudo] password for debjyotisarkar:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

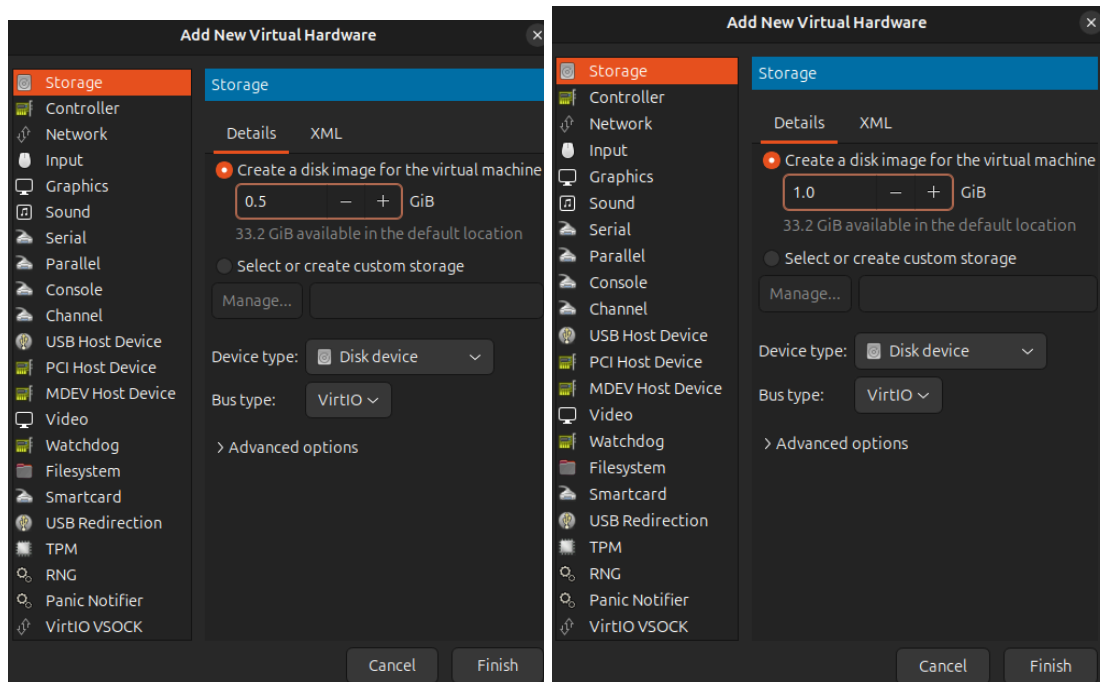
We can clone the system using virt-clone, the component of this command do the following:  
Original: name of the original VM, name - name of the new VM, file - the directory of the original VM is stored.

```
debjyotisarkar@sumonta-22341019:~$ sudo virt-clone --original debian12CLI --name
  debian12CLI_clone2 --file /var/lib/libvirt/images/ubuntu_vm_clone.img
Allocating 'ubuntu_vm_clone.img' | 7.0 GB 00:42 ...
Clone 'debian12CLI_clone2' created successfully.
```

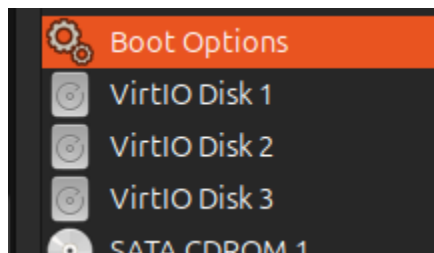


## Task 8

To create new disks using GUI, we need to go to the details sections of the VM and go to the add hardware option. From there, in the storage option we can create new disks and allocate the designated storage capacity that we want.



It shows all the disk options.



## Task 9

In order to add two hard disks using CLI, we first need to create two disks using qemu-img create a disk and following that, the directory of the new file, mentioning the size of the new disk, 1G in my case.

```
debjyotisarkar@sumonta-22341019: ~  
debjyotisarkar@sumonta-22341019:~$ sudo qemu-img create -f qcow2 /var/lib/libvirt/images/first-disk.img 1G  
Formatting '/var/lib/libvirt/images/first-disk.img', fmt=qcow2 cluster_size=65536 extended_l2=off compression_type=zlib size=1073741824 lazy_refcounts=off refcount_bits=16  
debjyotisarkar@sumonta-22341019:~$ sudo qemu-img create -f qcow2 /var/lib/libvirt/images/second-disk.img 1G  
Formatting '/var/lib/libvirt/images/second-disk.img', fmt=qcow2 cluster_size=65536 extended_l2=off compression_type=zlib size=1073741824 lazy_refcounts=off refcount_bits=16  
debjyotisarkar@sumonta-22341019:~$
```

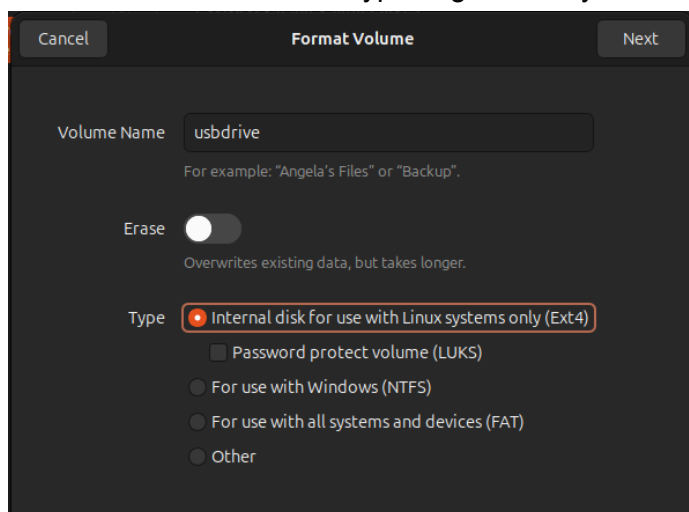
Using virsh attach-disk we can attach the newly created disks to the VM.

```
ount_bits=16  
debjyotisarkar@sumonta-22341019:~$ virsh attach-disk debian12CLI_clone2 /home/debjyotisarkar/vm-amges/first-disk.img vdd --targetbus virtio --persistent  
Disk attached successfully
```

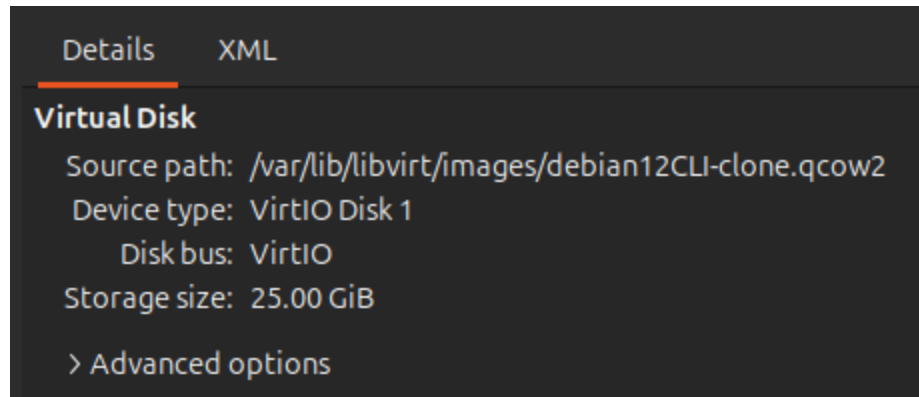
```
debjyotisarkar@sumonta-22341019:~$ virsh attach-disk debian12CLI_clone2 /home/debjyotisarkar/vm-amges/second-disk.img vde --targetbus virtio --persistent  
Disk attached successfully
```

## Task 10

To migrate a VM to another host, we first need a way to transfer the file. I am using a USB flash drive, I formatted it to Ext4 type to get it ready.



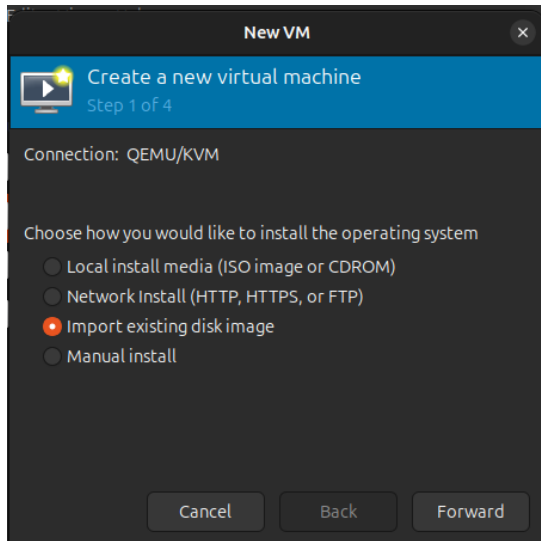
From the details section of the VM I want to migrate, I copied its original directory.



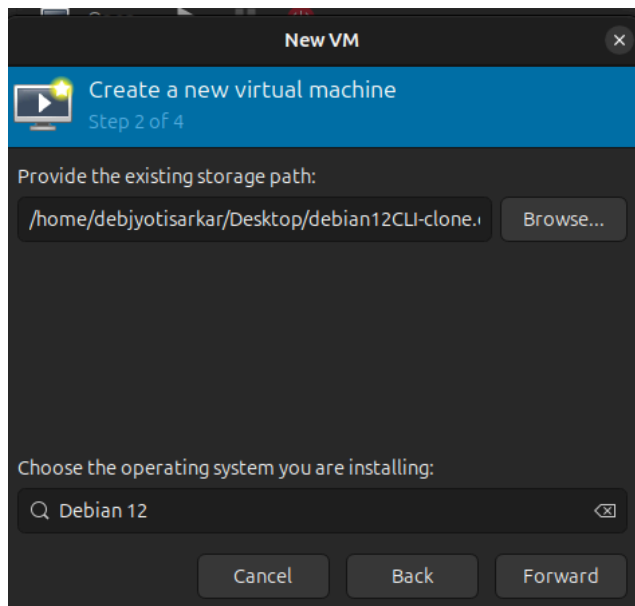
Firstly, I gave read-write permission using `sudo chmod 777`. Following that, I copied the file to the flash drive.

```
debjyotisarkar@sumonta-22341019:/media/debjyotisarkar/usbdrive$ sudo chmod 777 /var/lib/libvirt/images/debian12CLI-clone.qcow2
[sudo] password for debjyotisarkar:
debjyotisarkar@sumonta-22341019:/media/debjyotisarkar/usbdrive$ cp /var/lib/libvirt/images/debian12CLI-clone.qcow2 .
```

Now, in the new PC we need to copy the file from the flash drive to the local storage and click on the add a new VM option. From here, we need to choose the import existing disk image.



From there, I need to put the directory of where I stored the file in the local system and select the OS.



After allocating proper RAM, CPU and storage and naming my VM, the migration is complete and the VM is ready to be used.

