Cloud Computing - Assignment 1

Virtualization and Cloud Infrastructure Management using QEMU/KVM

Submitted By

Group 29

Mallidi Akhil Reddy – 2024TM93056

Ajinkya Thorawat – 2024TM93054

Naveen Kumar R – 2024TM93055

Pranjal Rastogi – 2024TM93053

Objective:

The objective of this cloud lab experiment is to gain practical experience in virtualization by setting up and working with QEMU/KVM hypervisor. The experiment involves tasks such as virtual machine (VM) creation, snapshot management, resource allocation, and network configuration, all using QEMU/KVM on a Linux host.

Background:

Theory/Concepts:

The concept of virtualization allows a physical computer to host multiple operating systems (OS). QEMU/KVM is a hypervisor that enables the creation and management of virtual machines (VMs) that mimic hardware, running independently.

Context:

In this experiment, the QEMU/KVM hypervisor is used on a Linux machine for virtualization. Management of the VMs is done via virt-manager and virsh, tools that facilitate the command-line and graphical management of VMs.

Tools and Services:

Cloud Services:

- QEMU/KVM hypervisor for virtual machine management.
- Linux host for running the hypervisor.

Software/Tools:

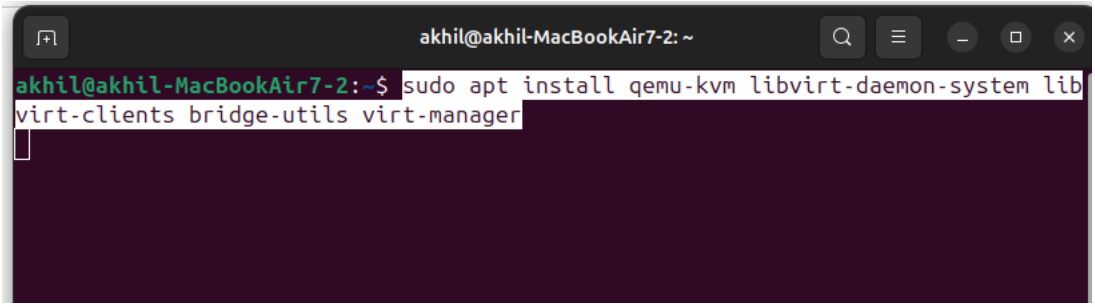
- **virt-manager:** GUI-based tool for managing KVM virtual machines.
- **virsh:** Command-line interface for managing VMs.
- **virt-top:** Monitoring tool for performance metrics.

Experiment Setup:

Task 1: QEMU/KVM Installation

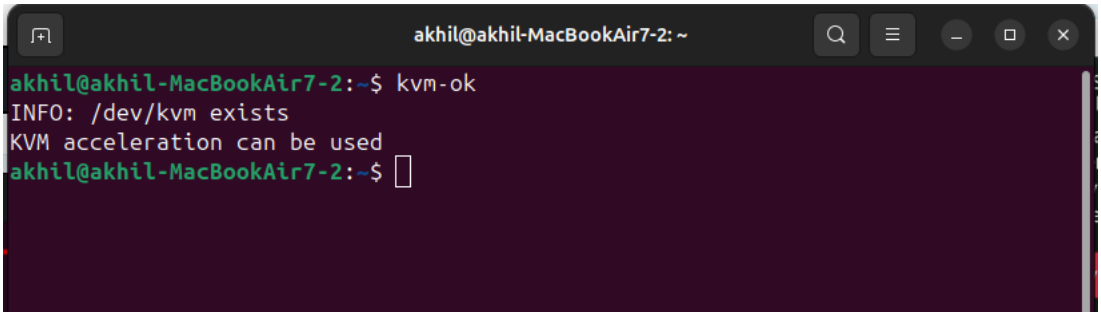
Install QEMU/KVM hypervisor and command and GUI tools for managing VM using the package manager in Linux machine using the below command.

```
sudo apt install qemu-kvm libvirt-daemon-system libvirt-clients
bridge-utils
```



Verify the installation by checking the status of KVM modules.

```
kvm-ok
```

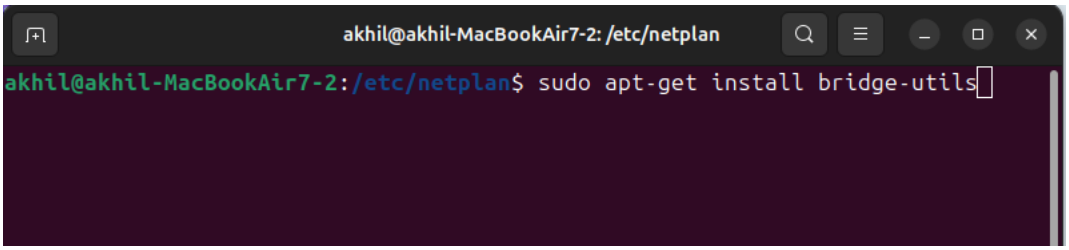


Networking Setup:

Configuring a bridge network, open the network configuration file and edit it with the following bridge network configuration.

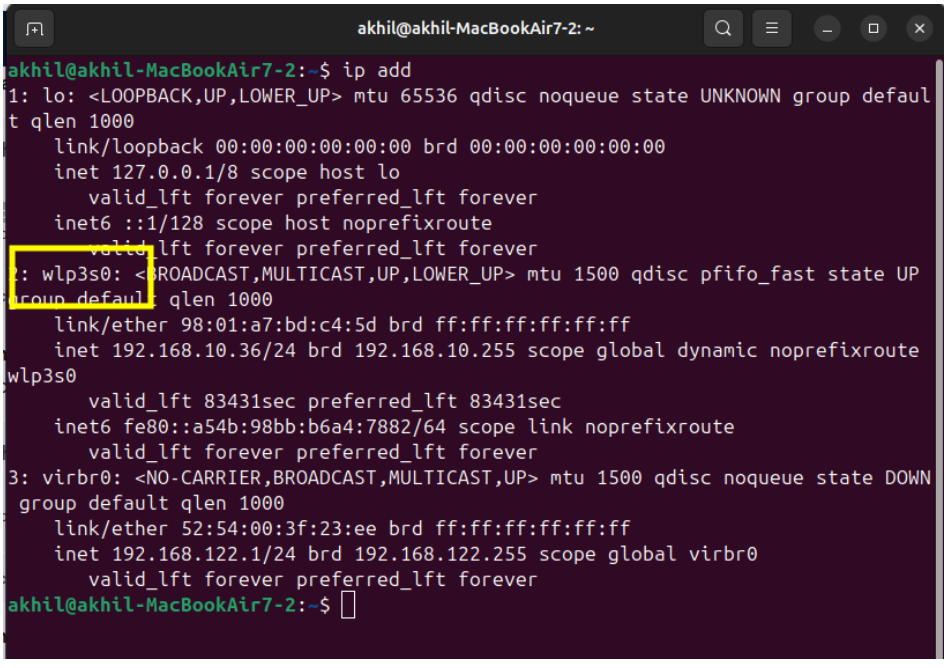
Install bridge-utils package and get the interface where we are trying to add bridge configuration

```
sudo apt-get install bridge-utils
```



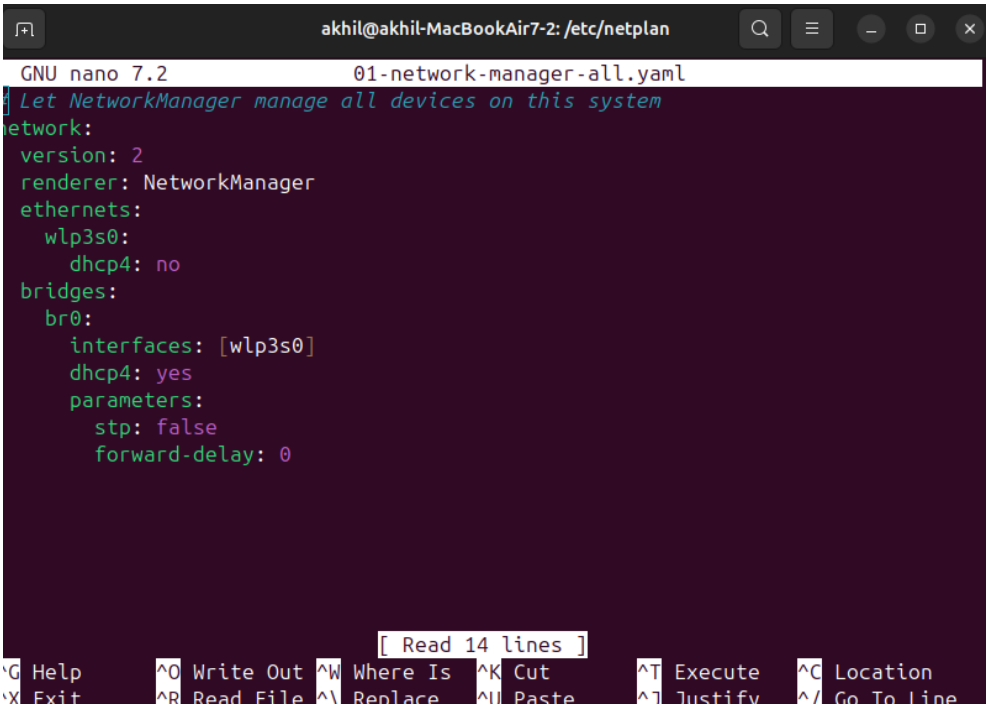
Using the below command, we can get the network interface keyname

```
ip add
```



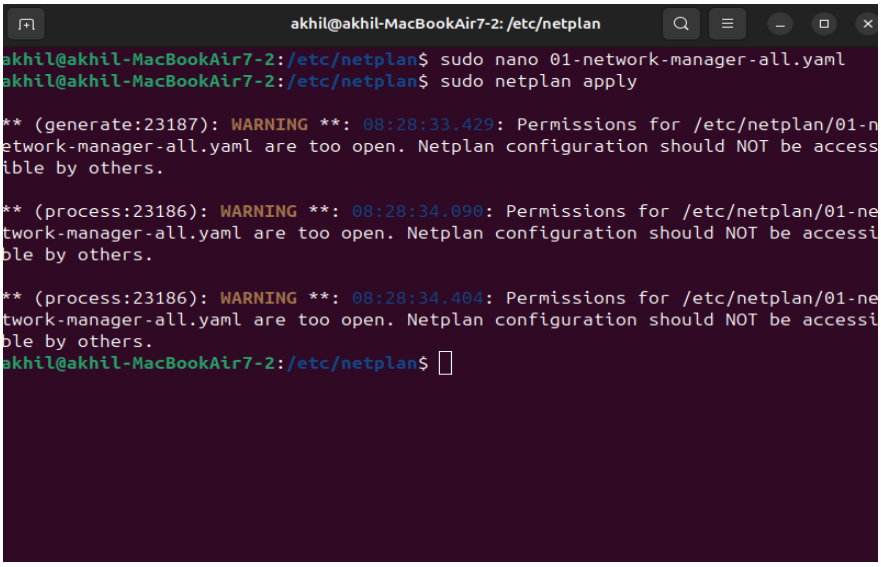
Open the network configuration file (*ex: 01-network-manager-all.yaml)* with nano edit to add the bridge configuration as below

```
sudo nano /etc/netplan/01-network-manager-all.yaml
```



Save the network configuration file and apply the bridge configuration with the below command

```
sudo netplan apply
```

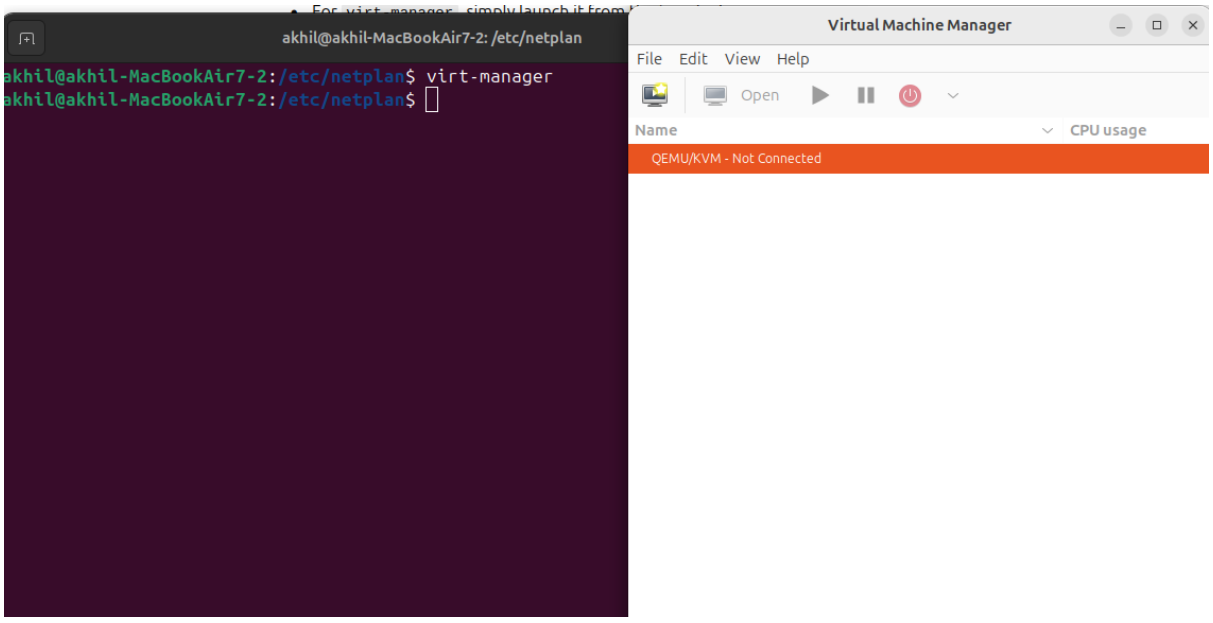


Management Tools:

We have already installed virt-manager (a graphical tool) or we can use virsh (a command-line tool).

We can run the virt-manager (GUI) using the below command

```
virt-manager
```

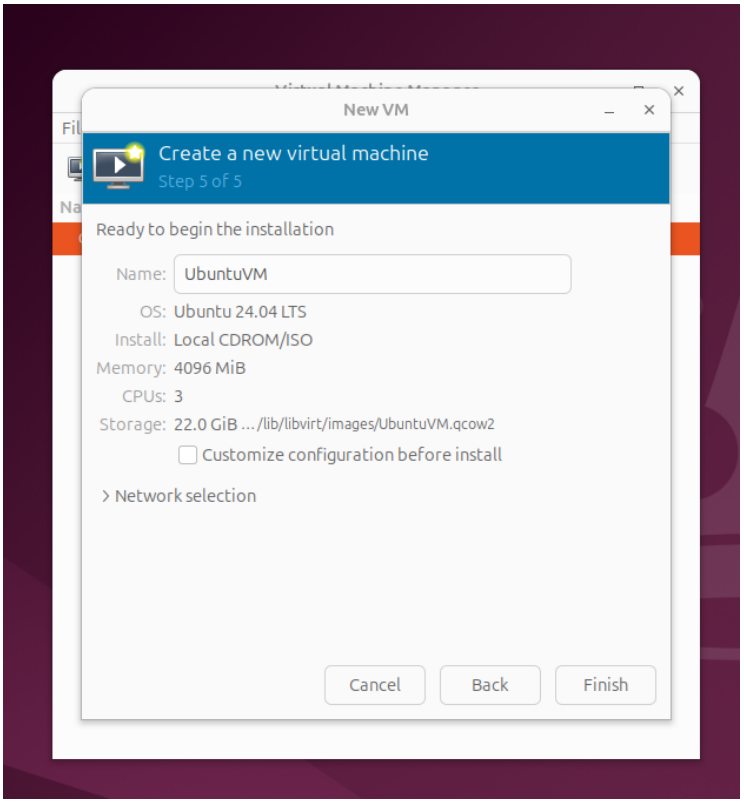


Task 2: Virtual Machine Creation and Management

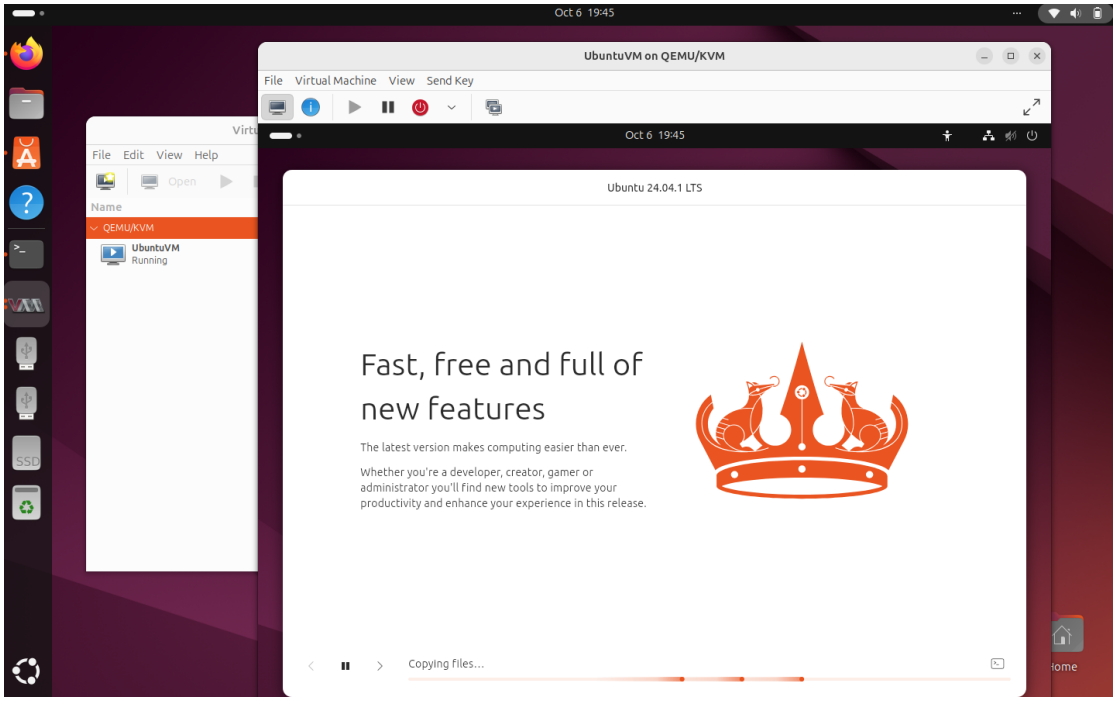
Create a Virtual Machine:

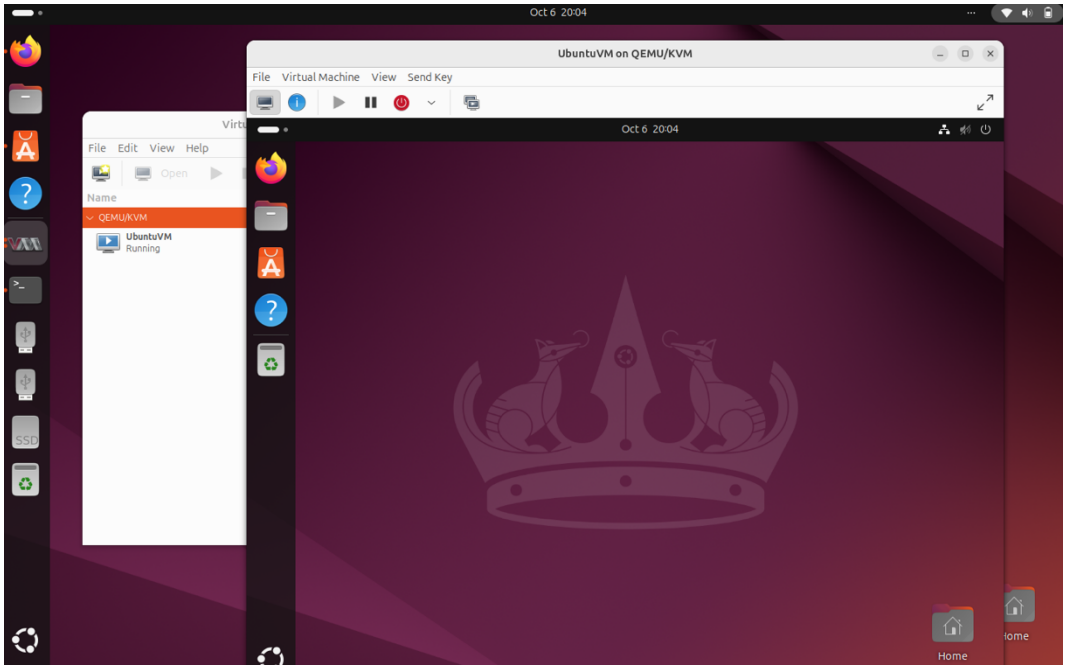
Download the Ubuntu ISO image from the official website.

Using virt-manager GUI, create a VM and select the configuration options like Local ISO Media, and the OS image we have downloaded, also memory, no of CPU's and disk space based on system capability and proceed with the VM creation.



Follow the installation steps for the Linux OS with disk partition, dependencies etc with the VM created

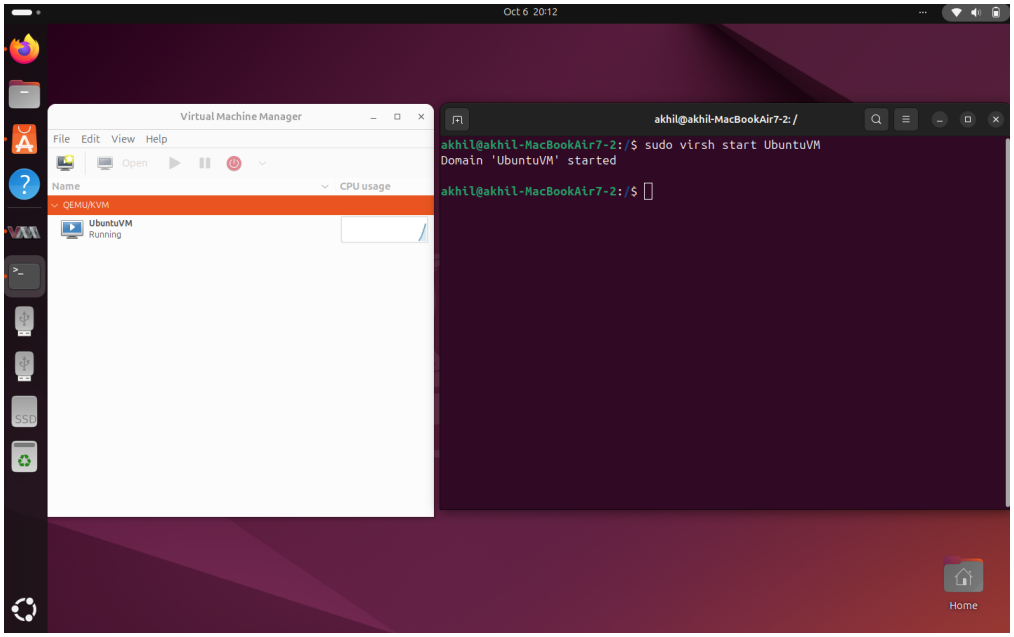




Basic VM Operations:

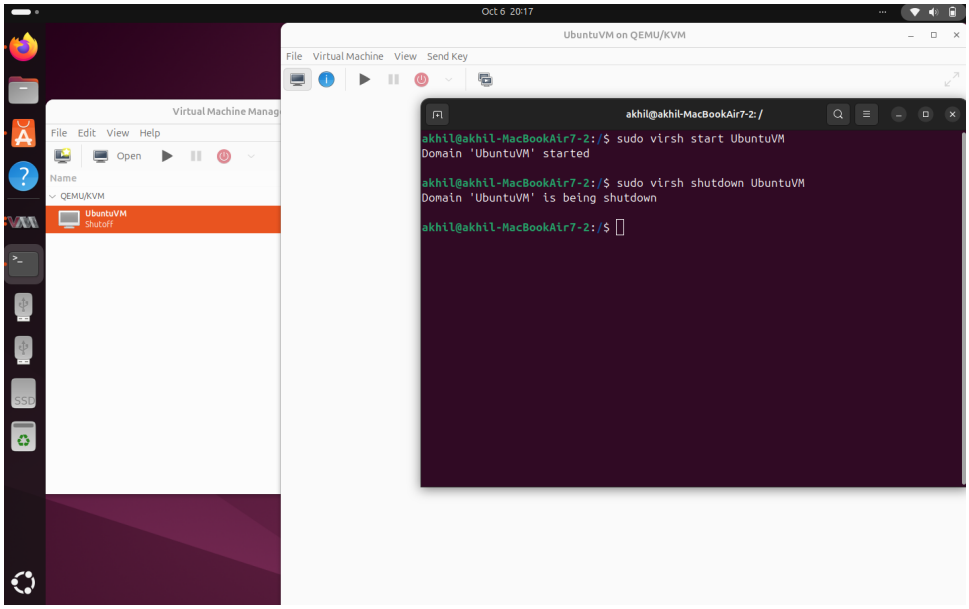
To start a VM using virsh:

```
virsh start UbuntuVM
```



For stopping:

```
virsh shutdown UbuntuVM
```



For suspending VM:

```
virsh suspend UbuntuVM
```

For resuming VM:

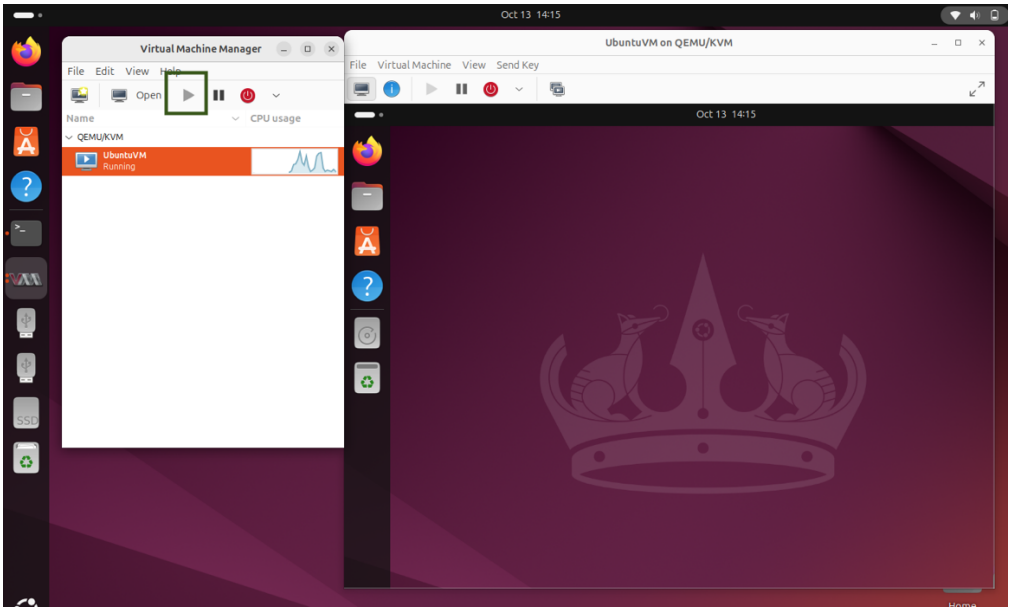
```
virsh resume UbuntuVM
```

To start a VM using virt-manager:

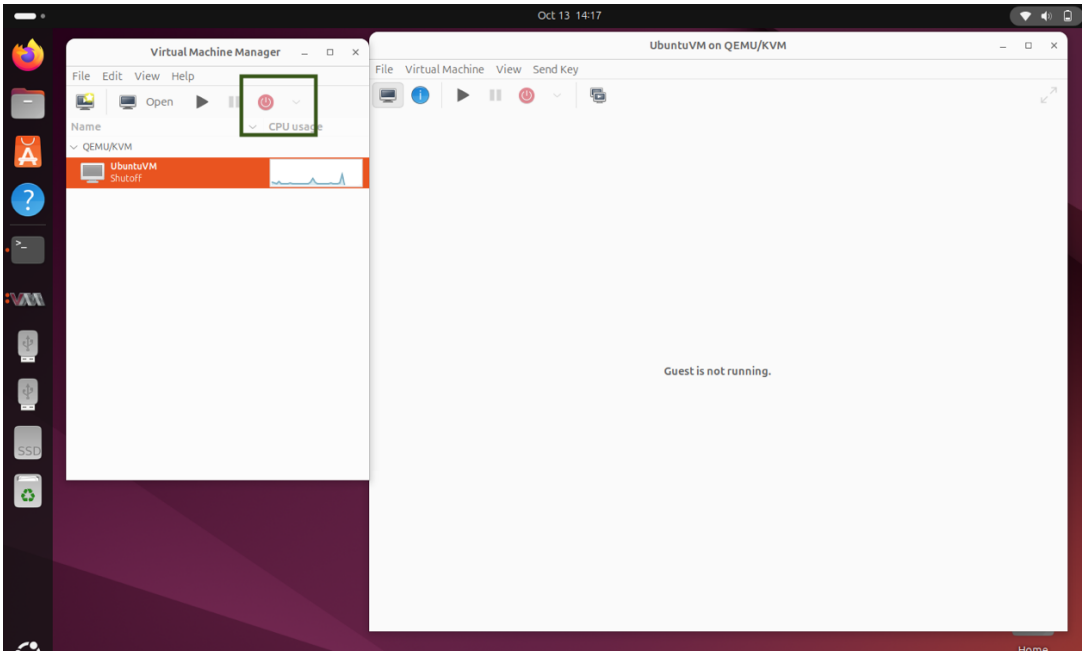
Start virt-manager GUI using the below command

```
virt-manager
```

In virt-manager GUI, we have start, reboot, shutdown etc buttons to manage the VM



For stopping VM using virt-manager:



Task 3: Snapshot and Cloning

Process of Snapshot:

A snapshot in virtualization captures the state of a virtual machine (VM) at a specific point in time. This state includes the VM's disk, memory, and configuration. Snapshots allow users to revert to a previous state of the VM, which is helpful for testing, troubleshooting, or experimenting.

Steps involved are:

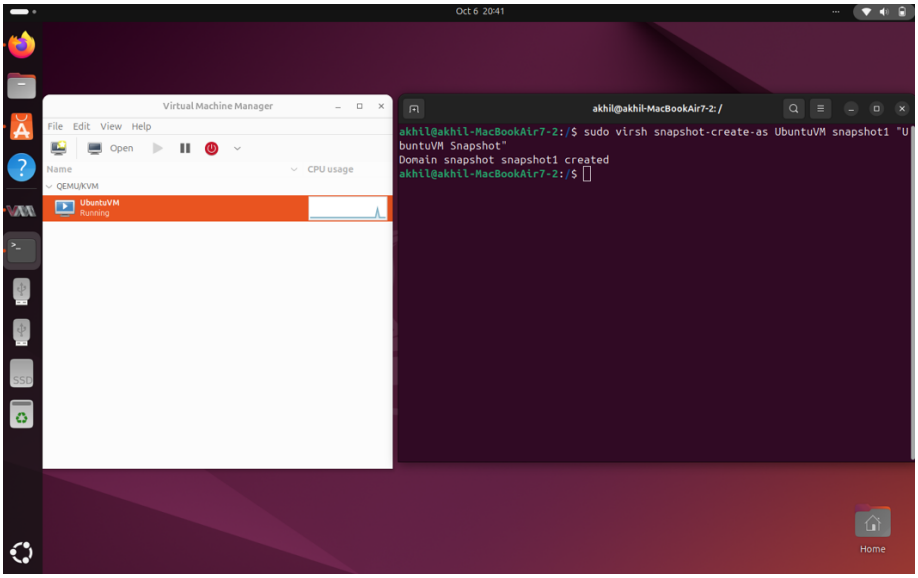
- Creating a Snapshot
- Modify the VM
- Reverting to previous state with Snapshot
- Deleting Snapshot

Use Cases:

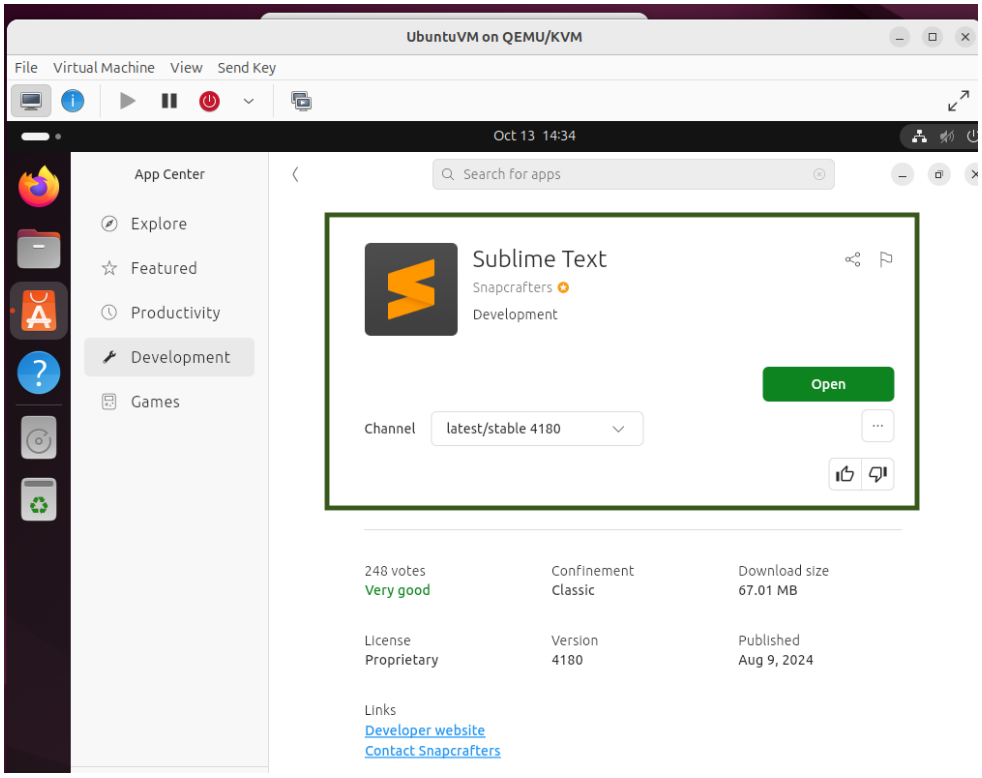
- Testing and Development
- Backup and Recovery
- System Migration
- Security Testing

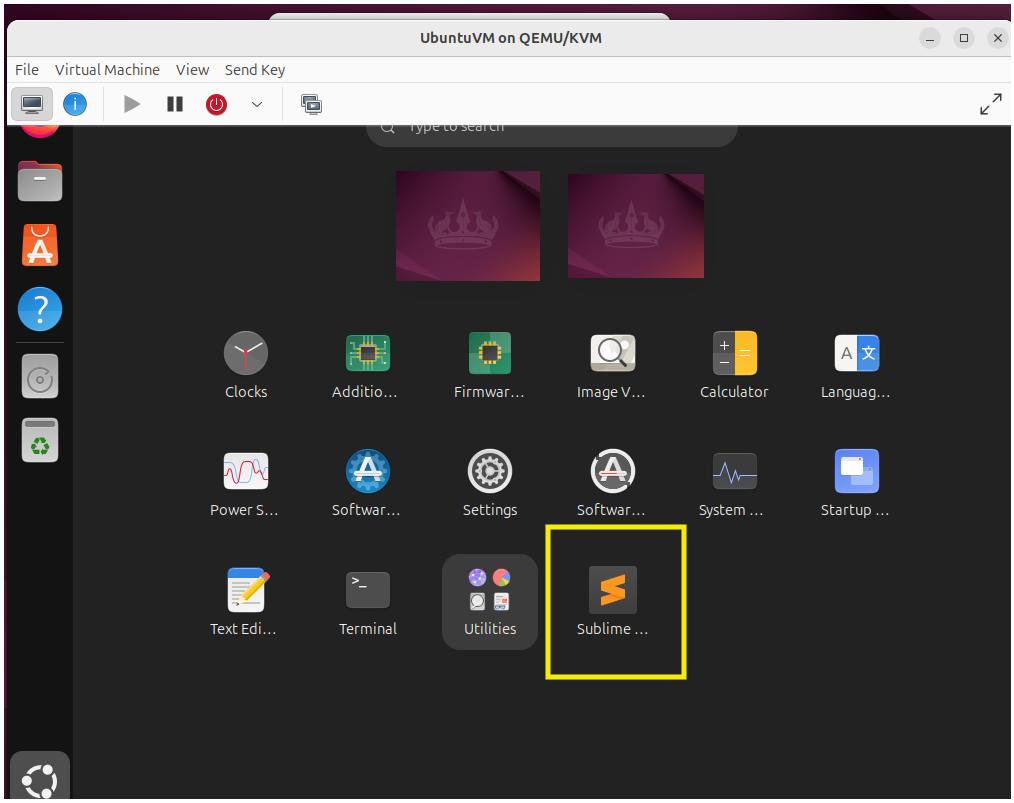
Creating a snapshot of a running VM using the command below

```
virsh snapshot-create-as UbuntuVM snapshot1 "UbuntuVM Snapshot"
```



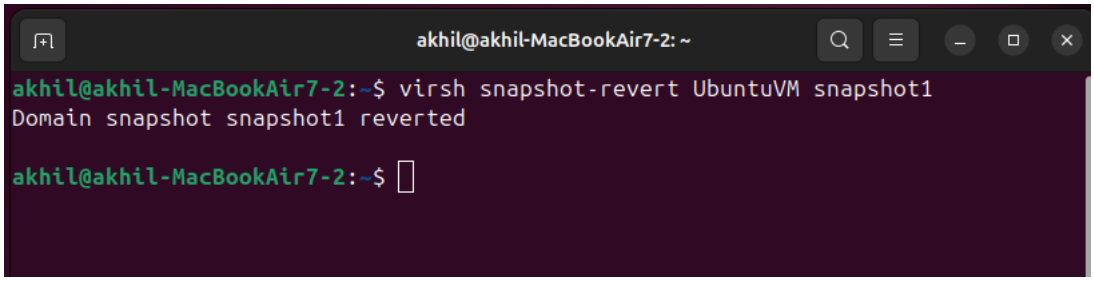
By installing or making changes in the VM, we can revert back to the previous state using the command below



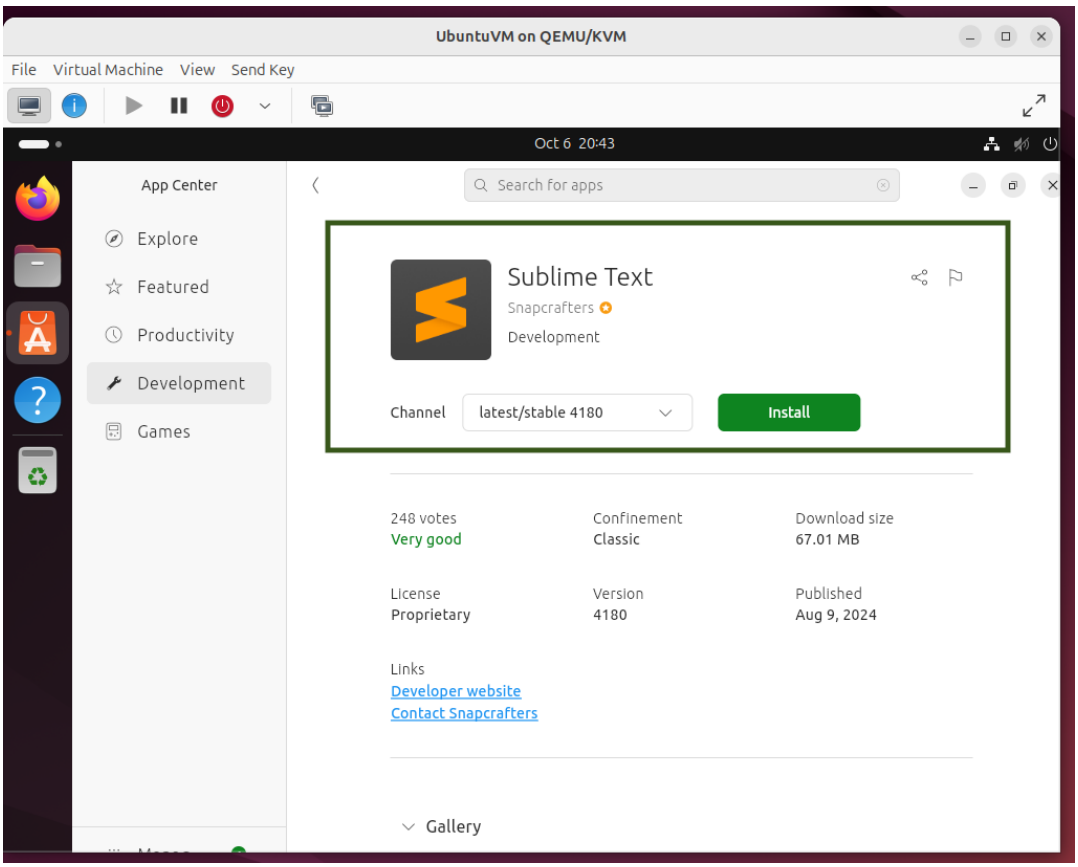


Reverting the VM to the previous snapshot using the below command

```
virsh snapshot-revert UbuntuVM snapshot1
```



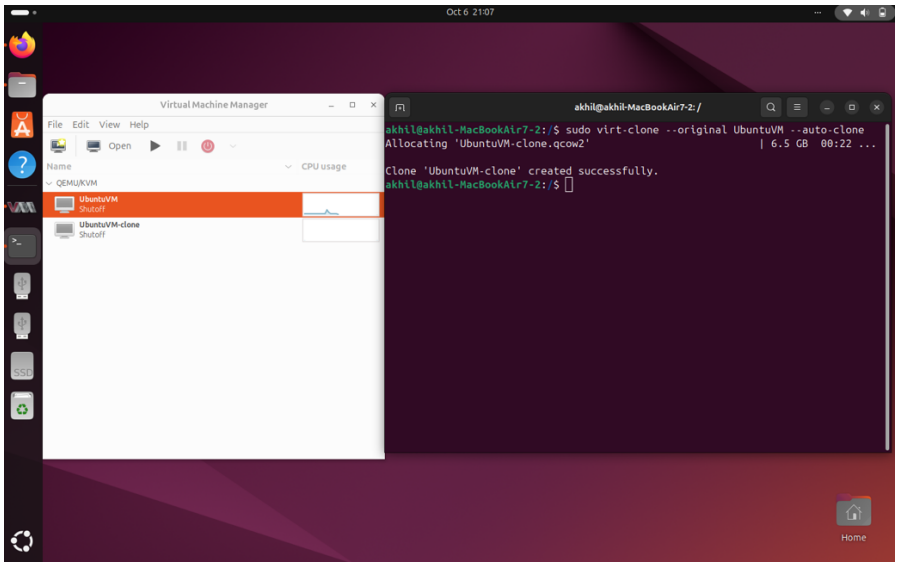
To verify reverting back to previous state using snapshot, the application we have installed will be uninstalled.



VM Cloning:

We can clone the existing VM using the following virsh command

```
virt-clone  --original  UbuntuVM  --auto-clone
```



Change the hostname and IP address of the cloned VM to avoid conflicts on the network.

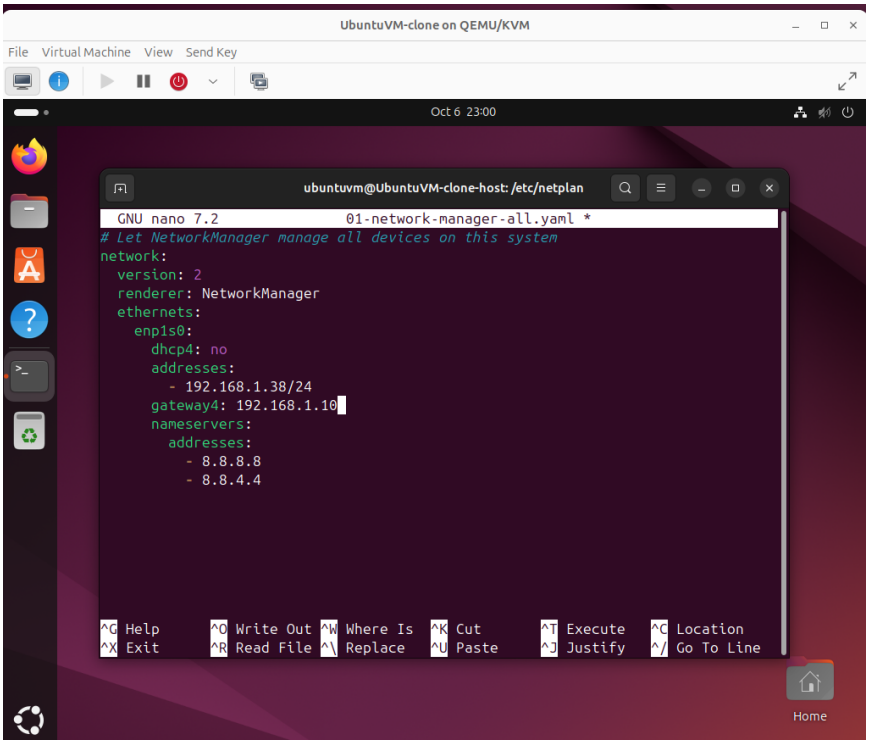
We can change the hostname of Cloned VM, by the following command in the Cloned VM terminal.

```
sudo hostnamectl set -hostname UbuntuVM-clone-host
```

Verify the hostname in the */etc/hosts,* and then reboot the system to apply the changes

```
sudo reboot
```

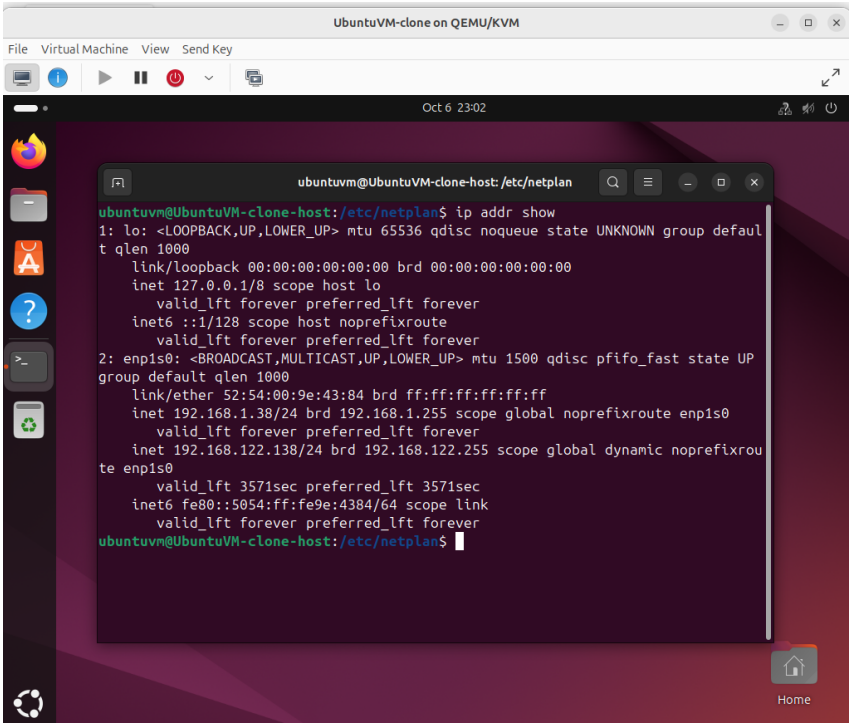
In *01-network-manager-all.yaml*, update the IP's near address and gateway to avoid the conflicts.



Save the network configuration file and apply the changes to reflect

```
sudo netplan apply
```

Verifying the IP changes

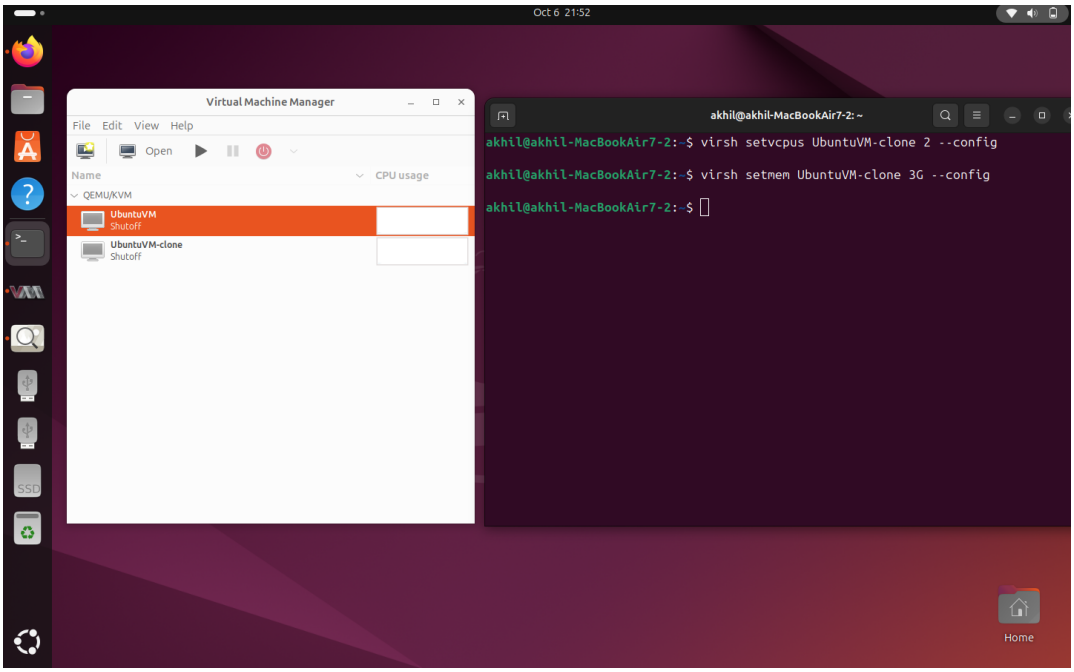


Task 4: Resource Allocation and Monitoring

We can modify the CPU and memory allocation of VM using virsh commands

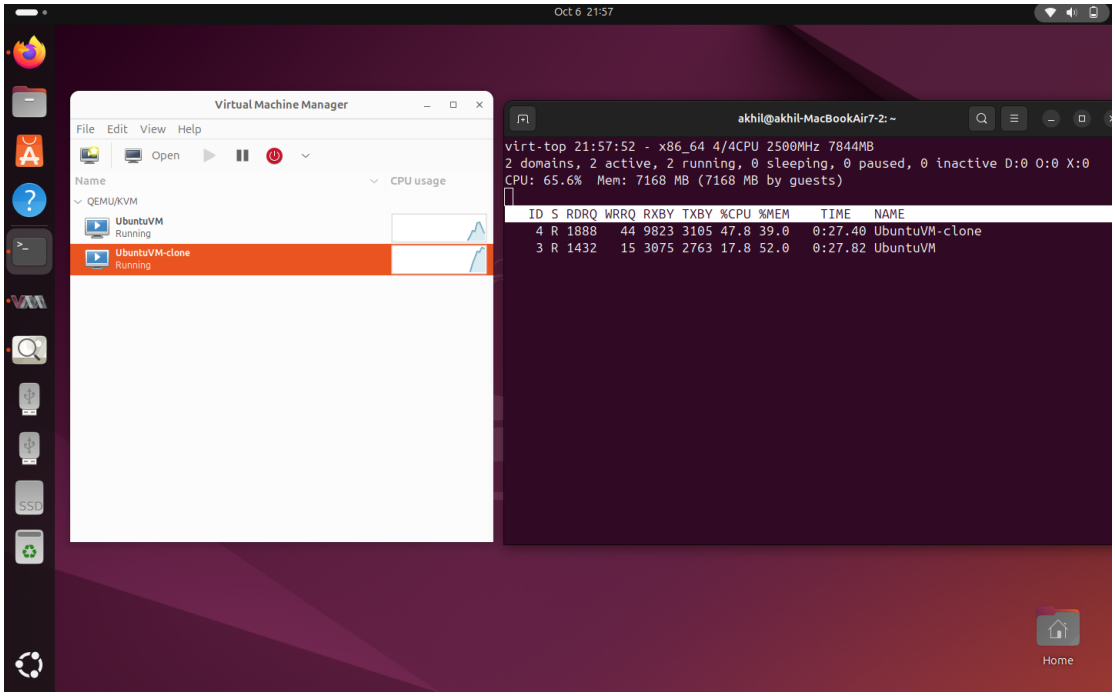
In our case we are modifying the CPU and memory allocation of cloned VM.

```
virsh setvcpus UbuntuVM-clone 2 - - config
virsh setmem UbuntuVM-clone 3G - - config
```



Installing virt-top package to monitor resource of usage of VM's created and the virt-top command

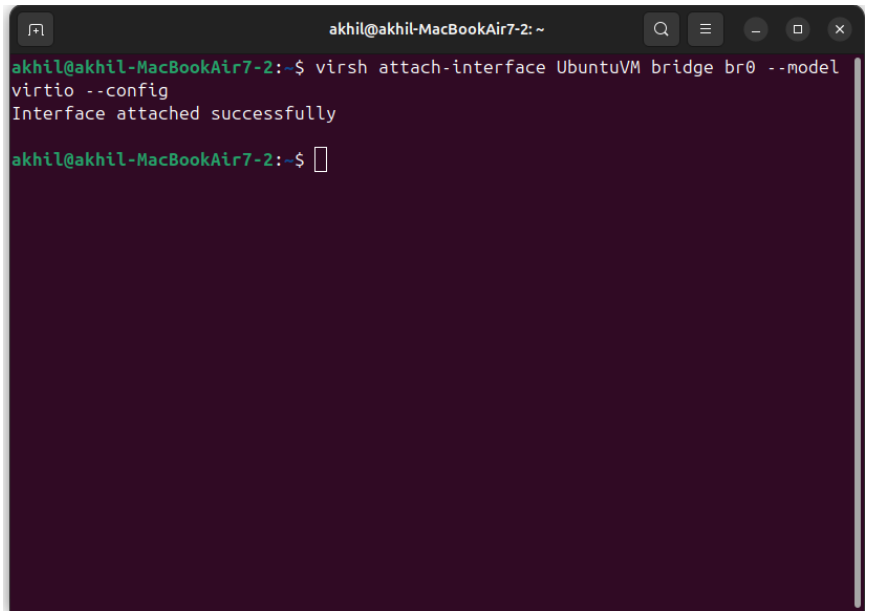
```
sudo apt install virt-top
```



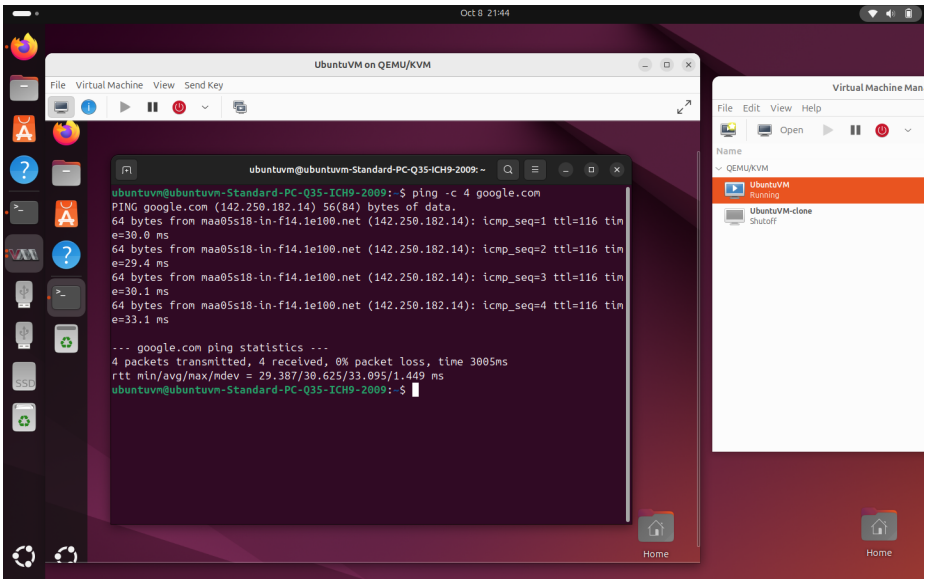
Task 5: Networking and Storage Management

Attaching a new virtual network interface that is br0 bridge network that we created in the Task 1 to the VM network configuration using the below command.

```
virsh attach-interface UbuntuM bridge br0 - - model virtio - -config
```



Verifying the bridge network connectivity in VM, by pinging google.com



Storage Management

Create a new virtual disk with 30GB storage, which can be used to attach and mount on the VM using the below command.

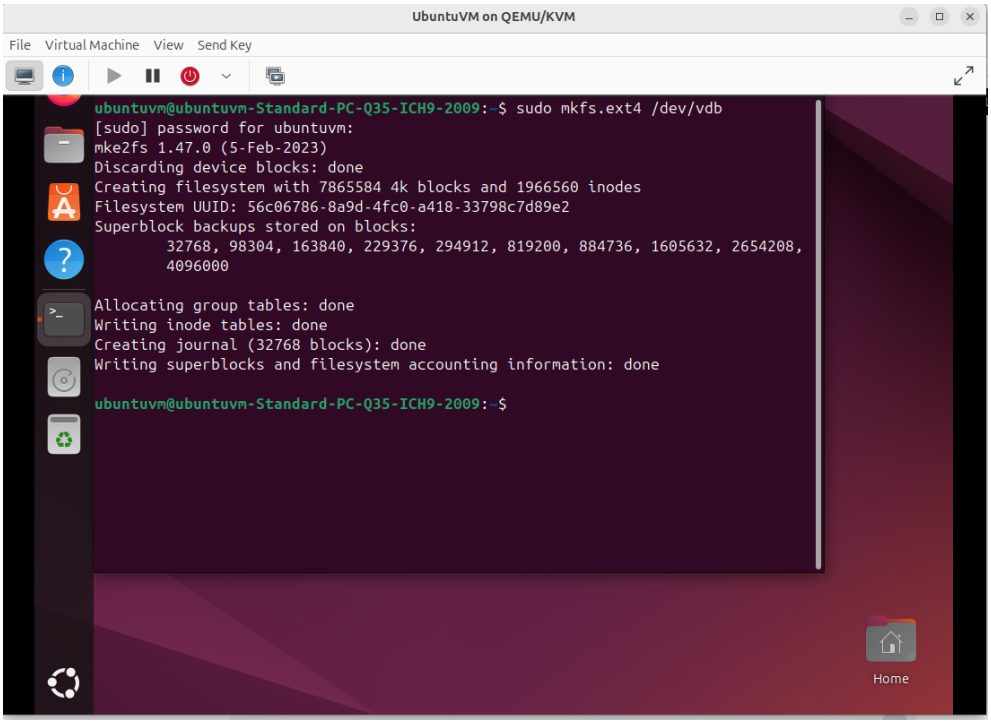
```
qemu-img create -f qcow2 -o preallocation=metadata
/var/lib/libvirt/images/vm_disk.img  30G
```

Attach the created new virtual disk to the VM

```
virsh attach-disk UbuntuVM  var/lib/libvirt/images/vm_disk.img  vdb
--config
```

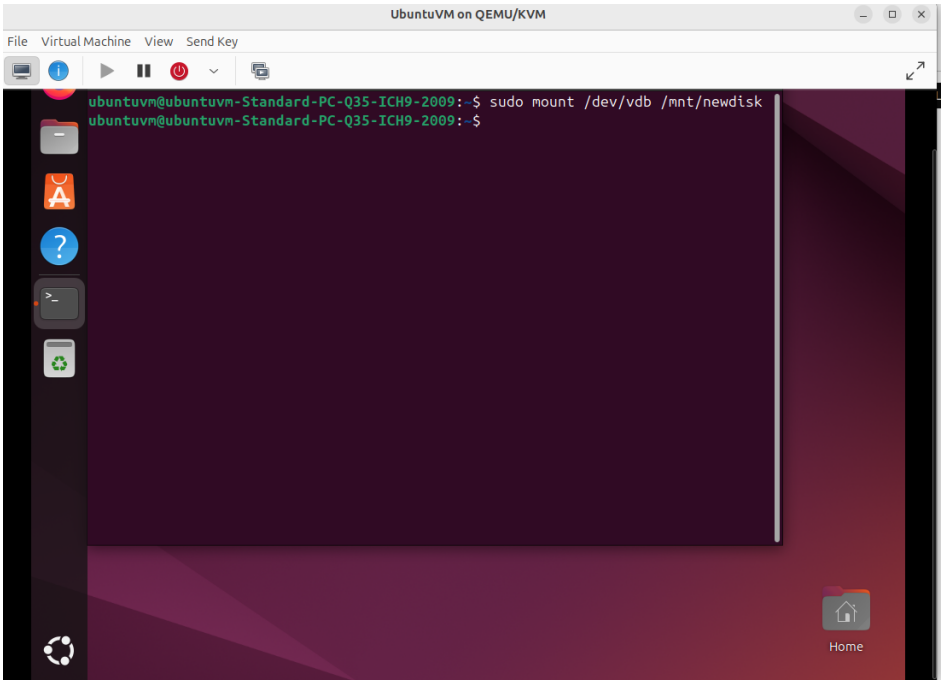
In the VM terminal, format the disk using the below command

```
sudo mkfs.ext4 /dev/vdb
```

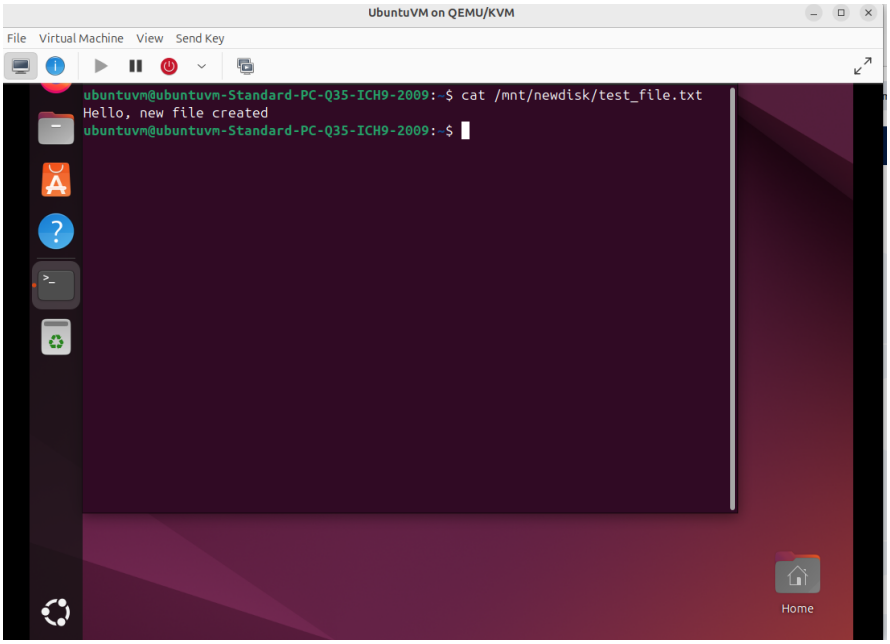


Mount the new virtual disk inside the VM

```
sudo mount /dev/vdb /mnt/newdisk
```

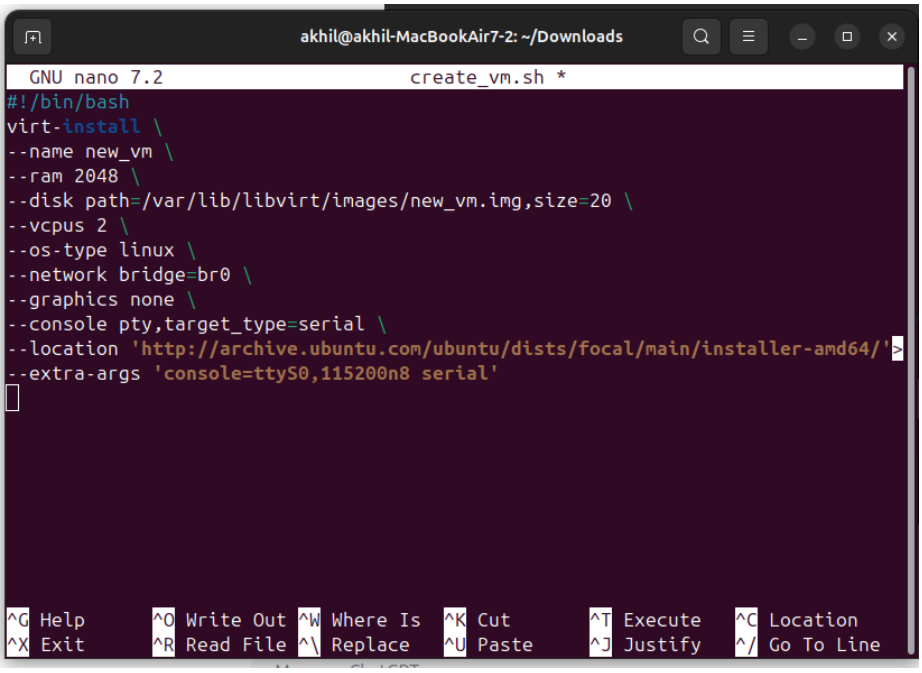


Creating a file and access that text file from the new virtual disk, to verify disk access using terminal.



Task 6: Automation with Scripts

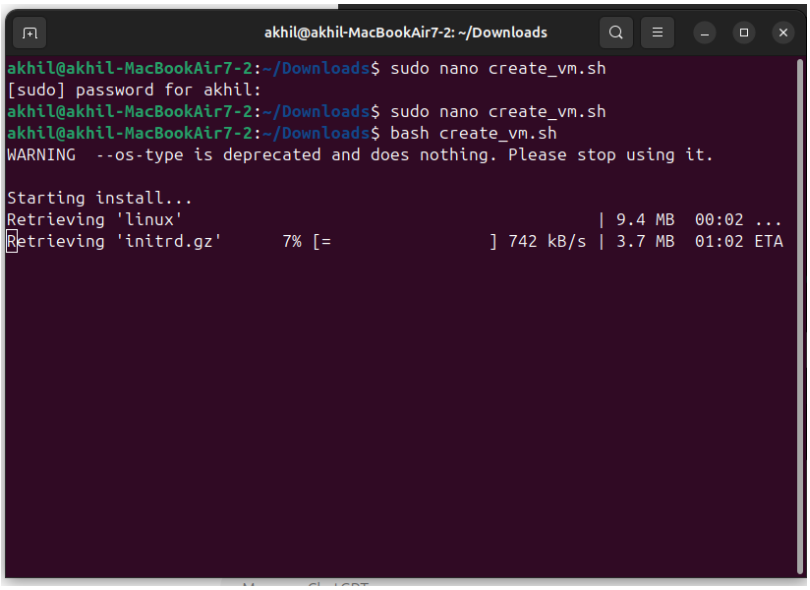
Create a Bash script as below to automate VM creation

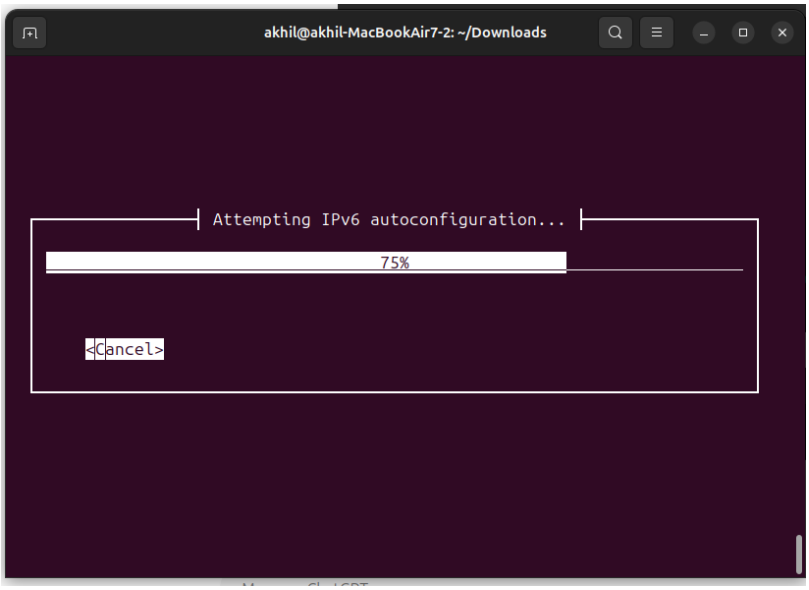


Running the bash create to create a new VM

```
bash create_vm.sh
```

Verifying the installation steps of new VM creation





Challenges faced:

A bridge network setup that is incorrect can prevent VMs from accessing the network or communicating with one another. We hade to revert back to previous snapshot, and ensuring bridge interface is configured correctly using netplan.

Incorrect resource allocation (CPU, memory, or disk) might degrade the performance VM. We have reconfigured the VM with sufficient resources.

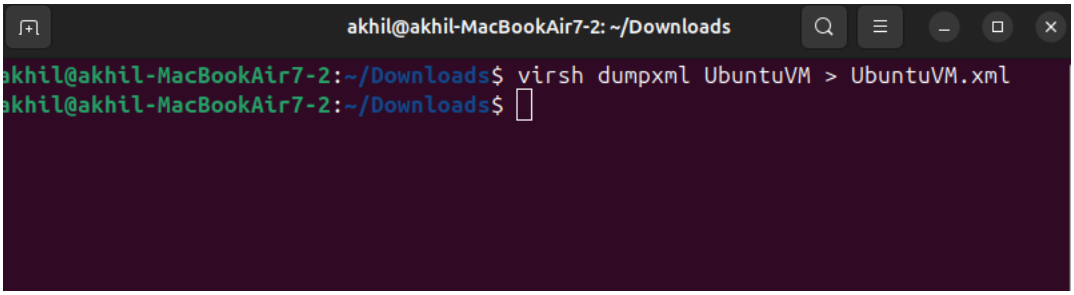
Ensure newly attached disk is formatted properly using a supported filesystem, or we might run into issues while mounting the new disk.

Results:

The QEMU/KVM hypervisor and VMs functioned as expected, allowing for VM creation, management, and snapshot restoration.

VM Images are exported as disk images using the following command

```
virsh dumpxml UbuntuVM > UbuntuVM.xml
```



UbuntuVM Image Link – https://drive.google.com/file/d/1EHo3IONBf-ULUh7zP8LhgOU2tKqsguLZ/view?usp=sharing

UbuntuVM-clone Image Link - https://drive.google.com/file/d/1B8qm3x97wHF6giwO1wRxIejmM2EITnJM/view?usp=sharing

Contribution Table:

Name	BITS ID	Contribution
Mallidi Akhil Reddy	2024TM93056	100%
Ajinkya Thorawat	2024TM93054	100%
Naveen Kumar R	2024TM93055	100%
Pranjal Rastogi	2024TM93053	100%

Key Learnings:

1. Mallidi Akhil Reddy

I gained a deep understanding of virtualization while working with QEMU/KVM. I learned how to create, manage, and configure virtual machines, handling operations such as starting, stopping, and suspending VMs efficiently. A major takeaway for me was mastering snapshot management, which allowed me to capture the VM's state and revert to it for testing or troubleshooting. I also dealt with challenges related to resource allocation and network configurations, ensuring the system ran smoothly by troubleshooting and reconfiguring both CPU and network settings for optimal VM performance.

2. Ajinkya Thorawat

Through this project, I learned how to set up and configure bridge networks for virtual machines, ensuring effective communication between them. I also gained hands-on experience with virt-manager and virsh for managing VMs via both GUI and command-line tools. While working on network configurations, I encountered issues with incorrect setups, but I resolved these by mastering tools like netplan. This experience has enhanced my ability to troubleshoot network issues and fine-tune configurations, which was critical to ensuring the smooth functioning of the VMs and their communication.

3. Naveen Kumar R

I focused on automating the virtual machine creation process using Bash scripting, which greatly improved efficiency. I learned how to optimize resource allocation, adjusting CPU and memory settings to enhance VM performance. Using tools like virt-top, I monitored the VMs, identified bottlenecks, and made necessary adjustments to ensure stability. This project taught me how crucial proper resource management is to virtual environments, and it also helped me understand the importance of automation in saving time and minimizing manual errors.

4. Pranjal Rastogi

In this project, I learned to manage storage in virtual environments by creating, attaching, and mounting virtual disks. I formatted disks and ensured proper file system management, troubleshooting any issues related to disk access. Additionally, I configured virtual network interfaces, making sure the VMs had seamless access to both network and storage resources. This hands-on experience with managing storage and networking has given me a strong foundation in virtualization, and I now feel confident in handling virtual environments in a cloud infrastructure setup.