CSE 3035

PRINCIPLES OF CLOUD COMPUTING

Lab Assessment – 1

L15+L16 | SJT501
Dr. Sivaprakash S

FALL SEMESTER 2022-23

by

SHARADINDU ADHIKARI
19BCE2105

Assessment 1 .

Experiment 1

Title: VM config (Local & Cloud).

Aim: To Create, Configure, and access a Virtual machine (VM) on my local machine and Cloud with the basic resources. Allocate CPU, memory, and storage space as per a specified requirement. Install the Guest OS image in that instance, launch the same and confirm the successful installation of the OS by performing a few OS commands in the Terminal.

Background Theory:

Virtualization:

- Uses software to create an abstraction layer over computer hardware that allows the hardware elements of a single computer—processors, memory, storage and more—to be divided into multiple virtual computers, commonly called virtual machines (VMs).

- Each VM runs its own operating system (OS) and behaves like an independent computer, even though it is running on just a portion of the actual underlying computer hardware.

Virtual Machine:

- Virtual machines (VMs) are virtual environments that simulate physical computing in software form.

- They normally comprise several files containing the VM's configuration, the storage for the virtual hard drive, and some snapshots of the VM that preserve its state at a particular point in time.

Hypervisors:

- A hypervisor is the software layer that coordinates VMs.

- It serves as an interface between the VM and the underlying physical hardware, ensuring that each has access to the physical resources it needs to execute.

- It also ensures that the VMs don't interfere with each other by impinging on each other's memory space or compute cycles.

Hypervisor Types:

- Type 1 or "bare-metal" hypervisors interact with the underlying physical resources, replacing the traditional operating system altogether. They most commonly appear in virtual server scenarios.

- Type 2 hypervisors run as an application on an existing OS. Most commonly used on endpoint devices to run alternative operating systems, they carry a performance overhead because they must use the host OS to access and coordinate the underlying hardware resources.

Requirements for conducting the experiment:

Local Instance:

- Oracle VM VirtualBox (or)
  VMware Workstation Player (or)
  VMware Workstation Pro (or)
  Any other Virtual Machine Setup

- Any OS ISO Image (preferably Ubuntu)

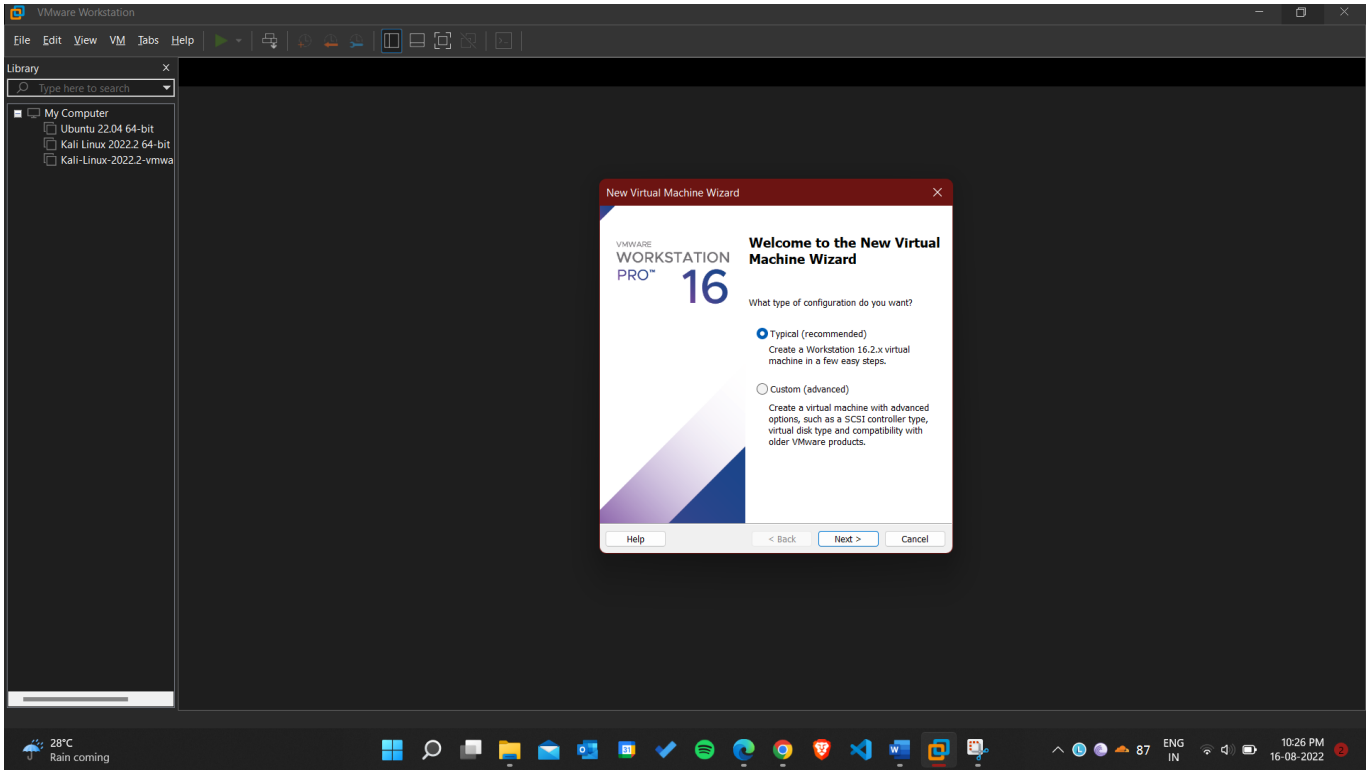
Cloud:

- AWS Licence

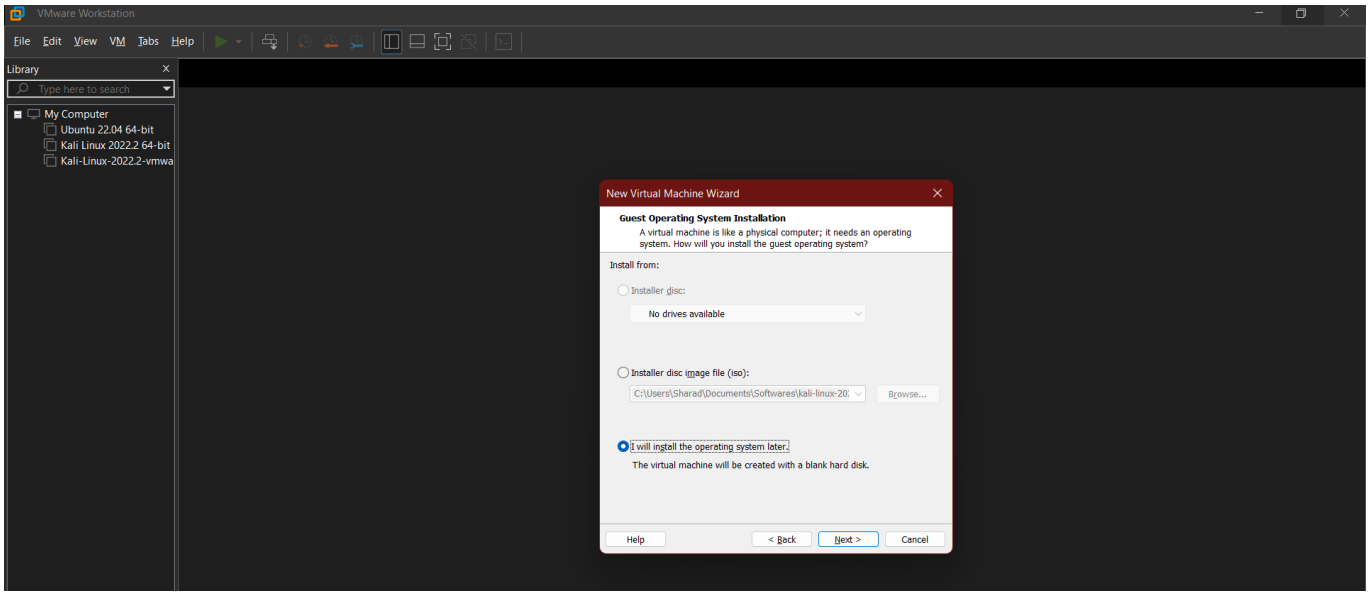
Procedure & Screenshots:

(1) Local Instance:

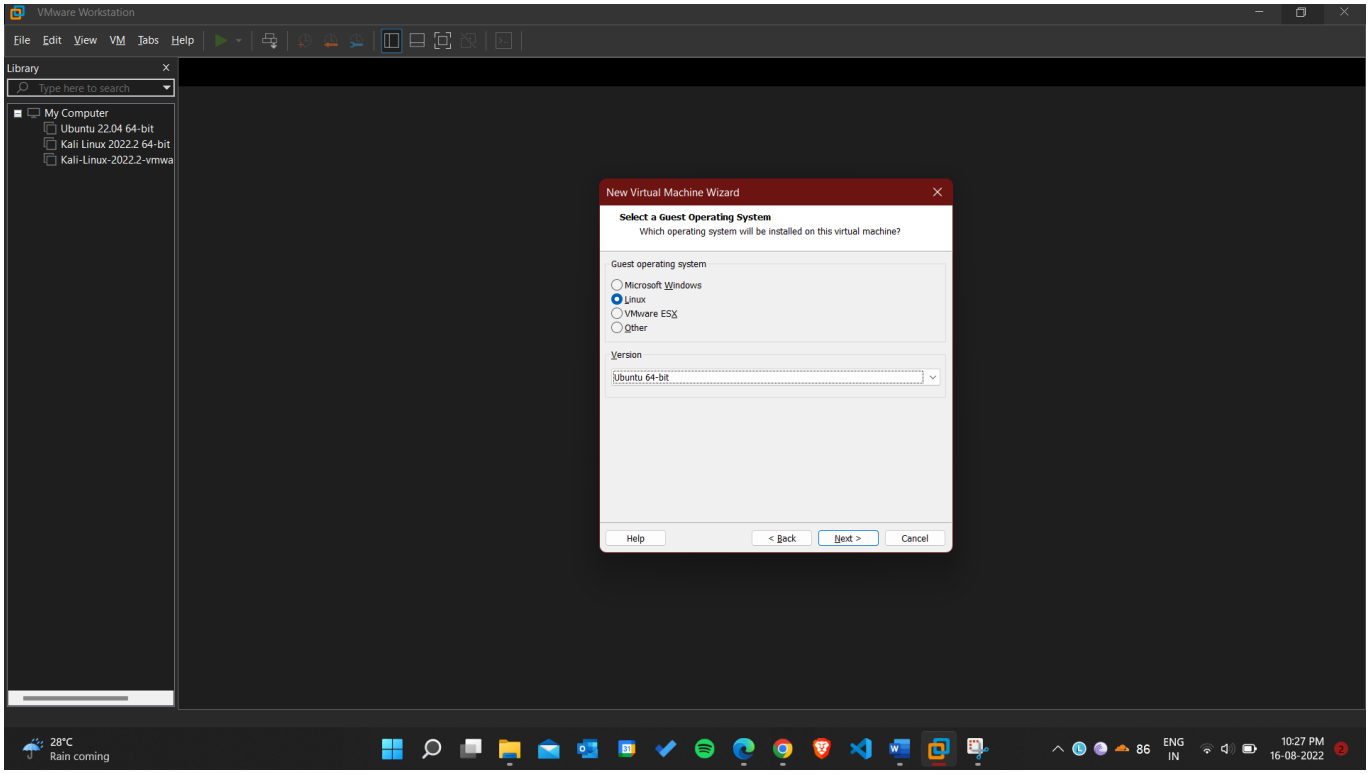
Step 1: Type of Installation → Typical, where we do not go into the SCSI Controller type, Virtual Disk Type etc.



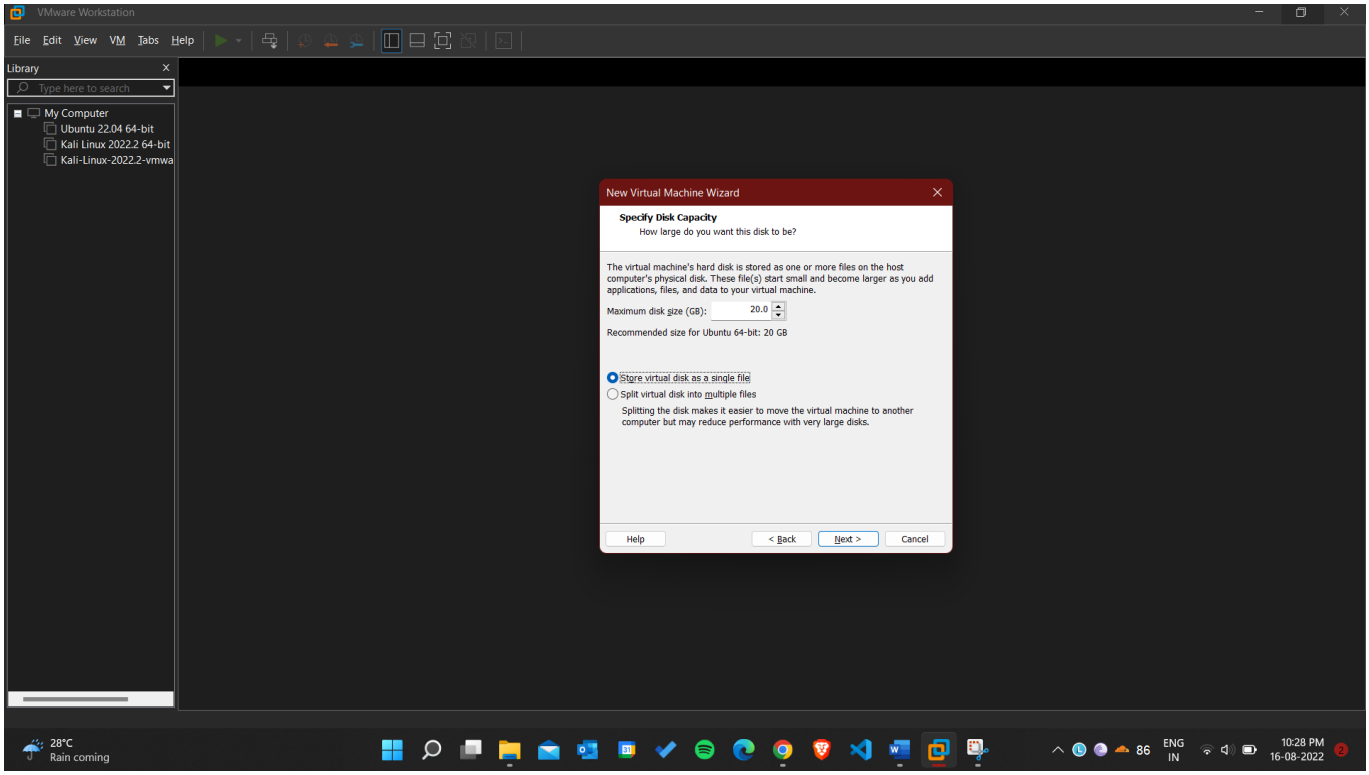
Step 2: We skip adding the iso file, for now, just creating the virtual machine setup.



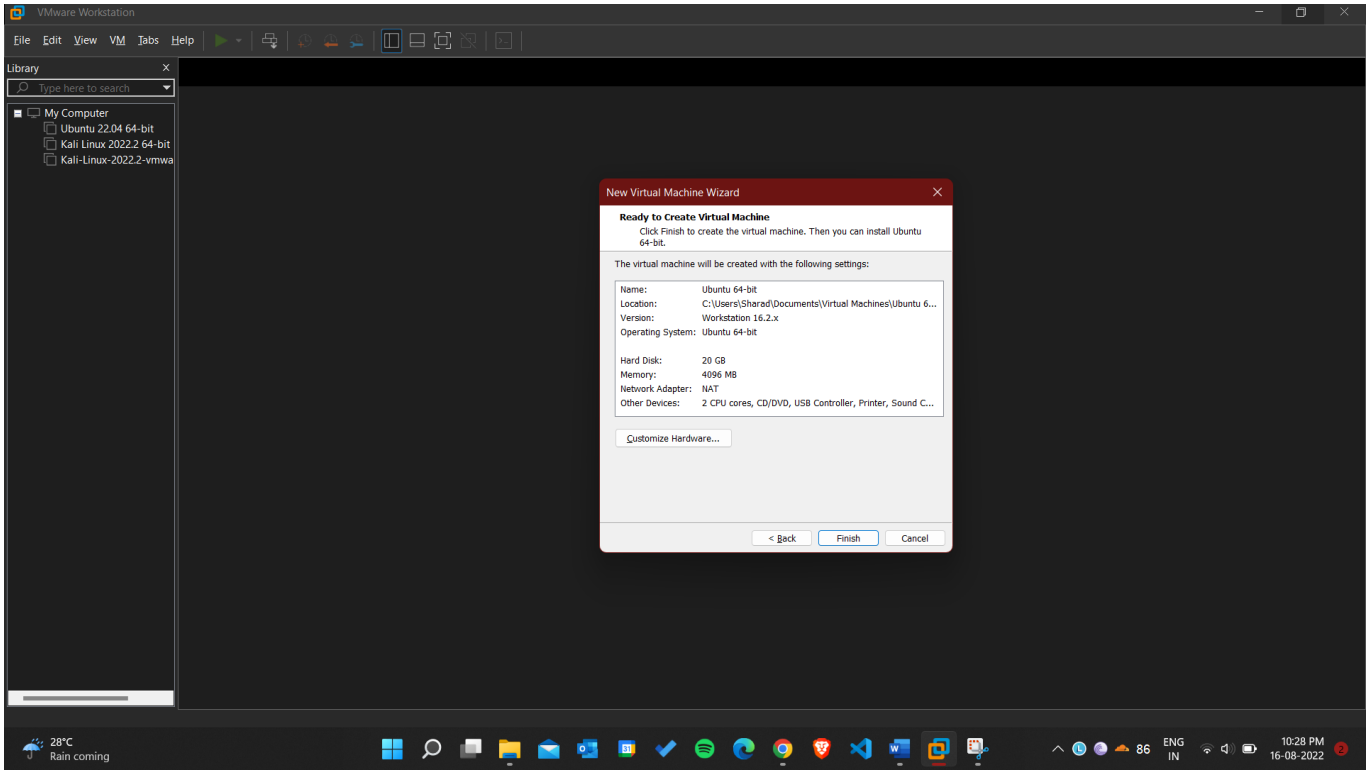
Step 3: Select the type of OS we're planning on installing.

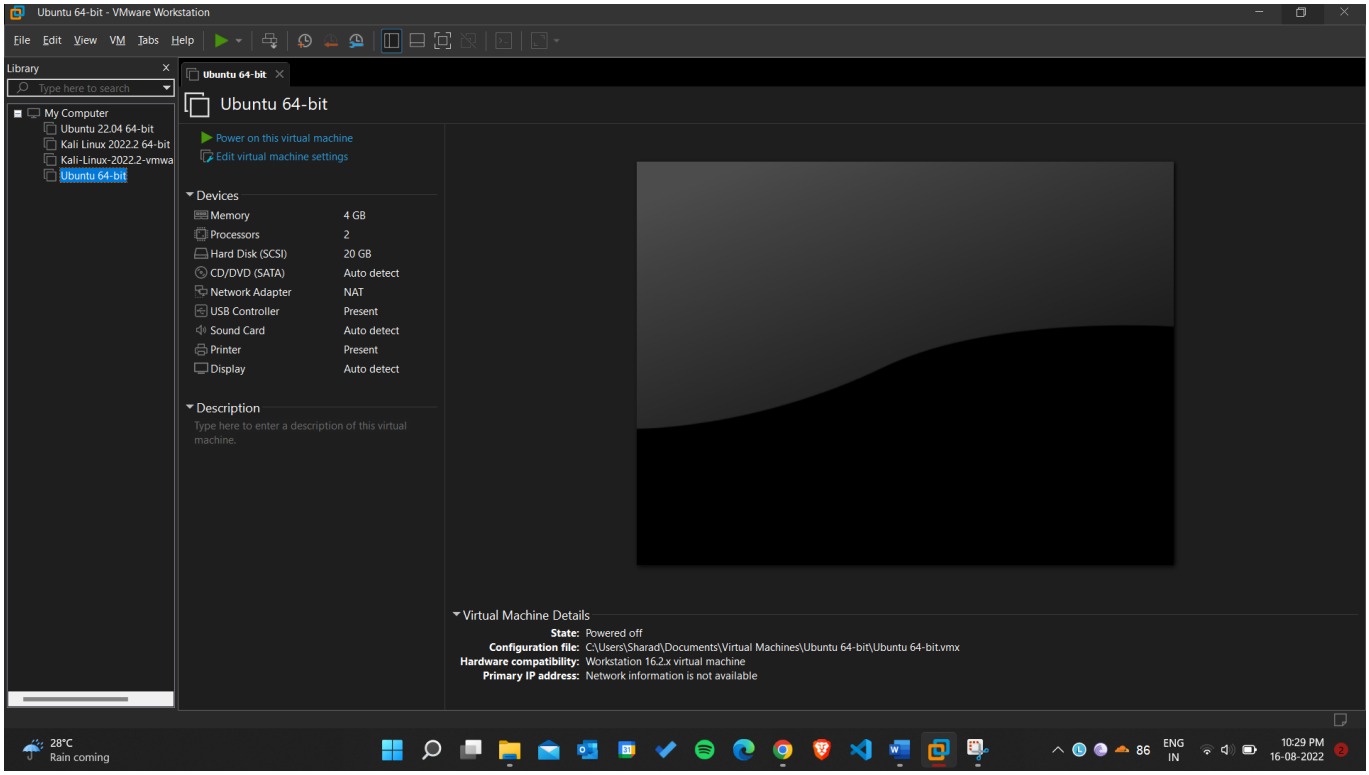


Step 4: Select the VM disk storage type.



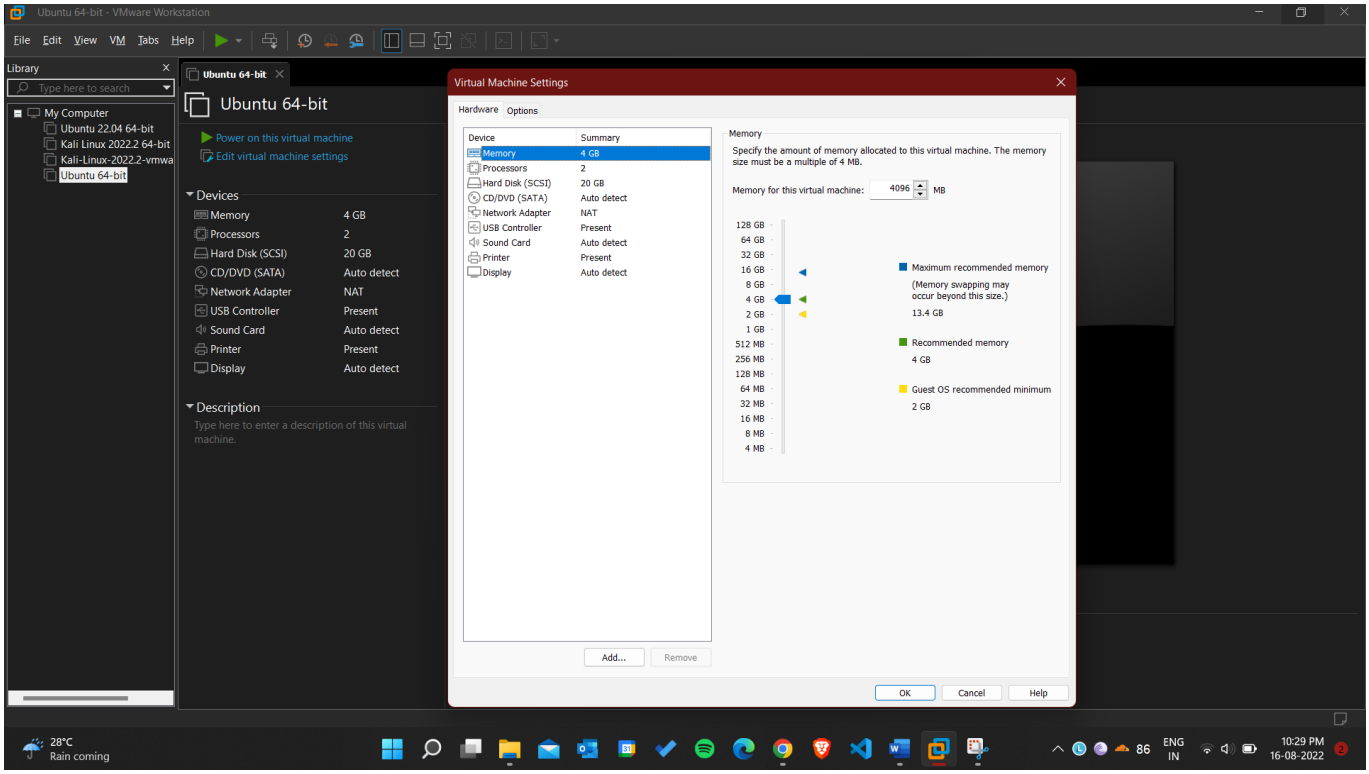
Step 5: Finish setting up the VM (for now).



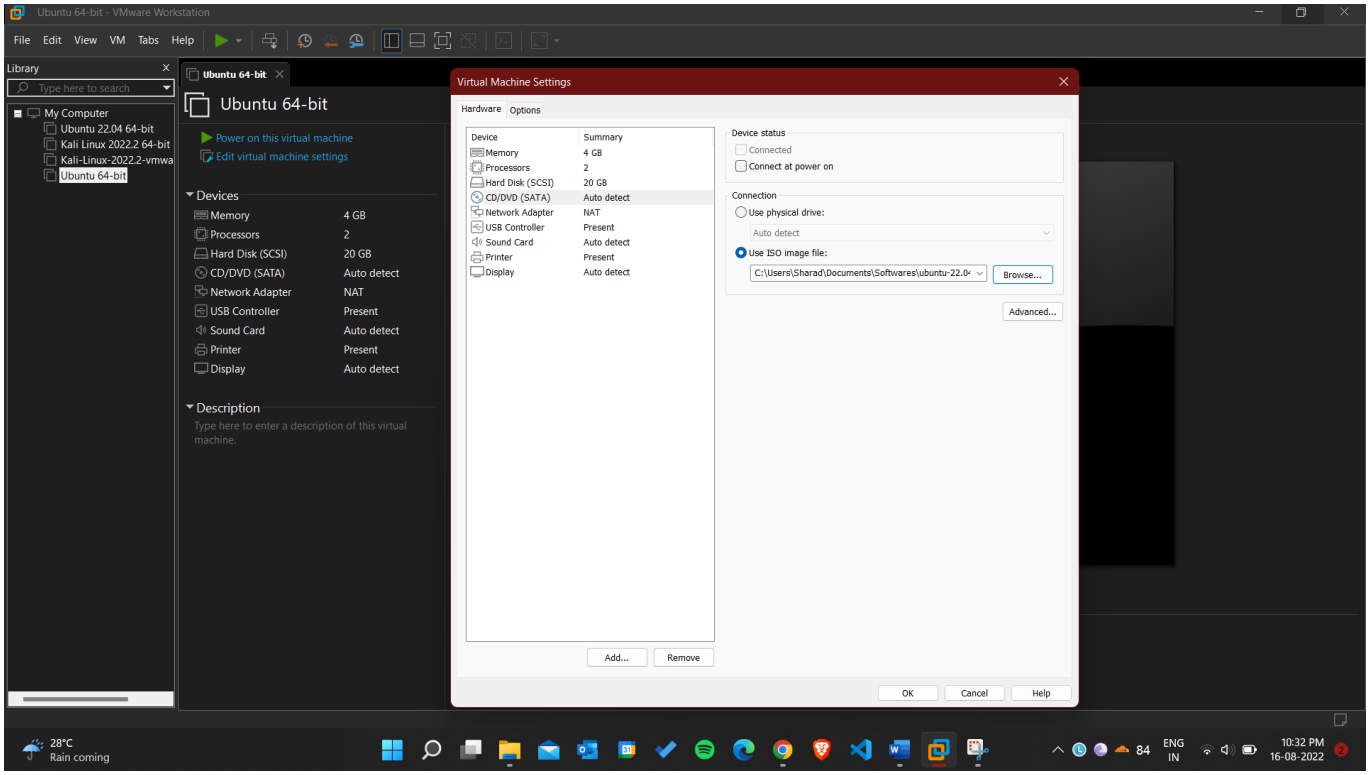


Step 6: Editing the VM configurations

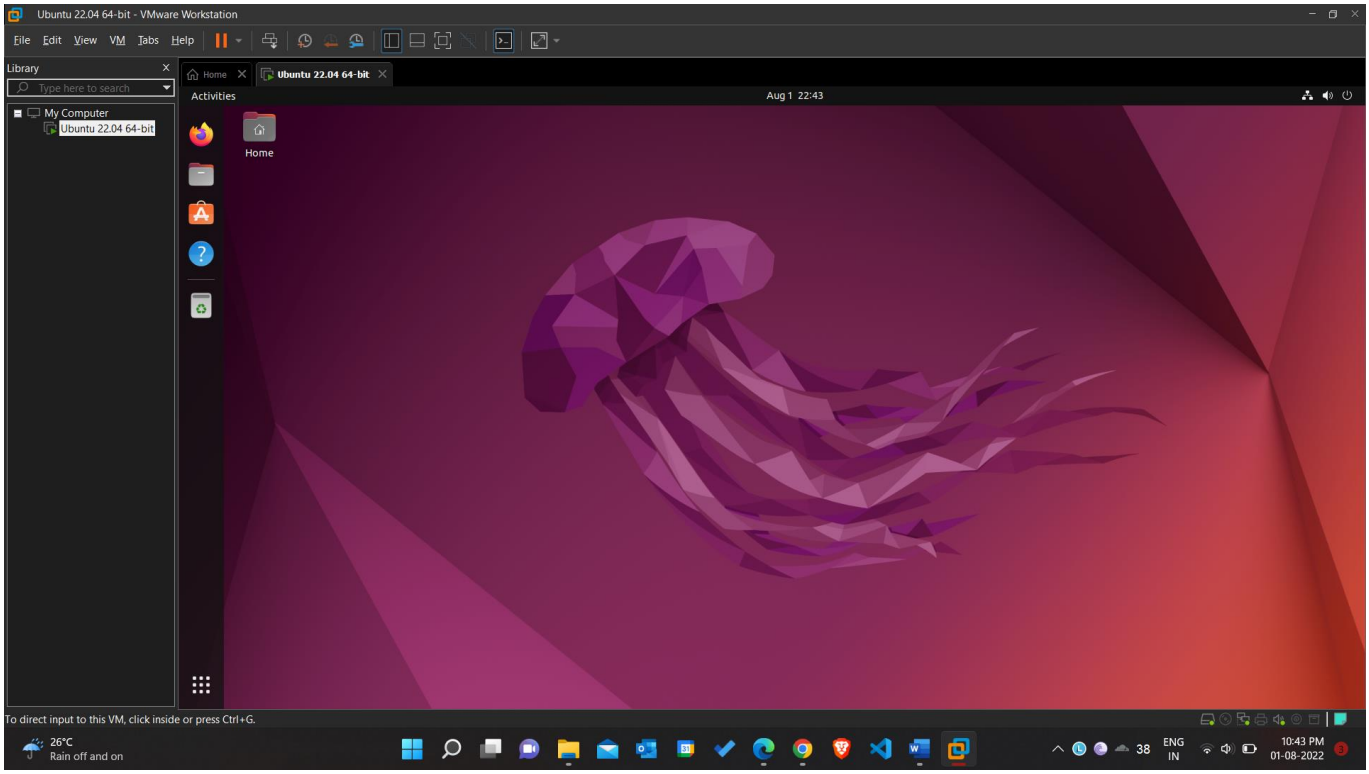
sharadindu.adhikari2019@vitstudent.ac.in



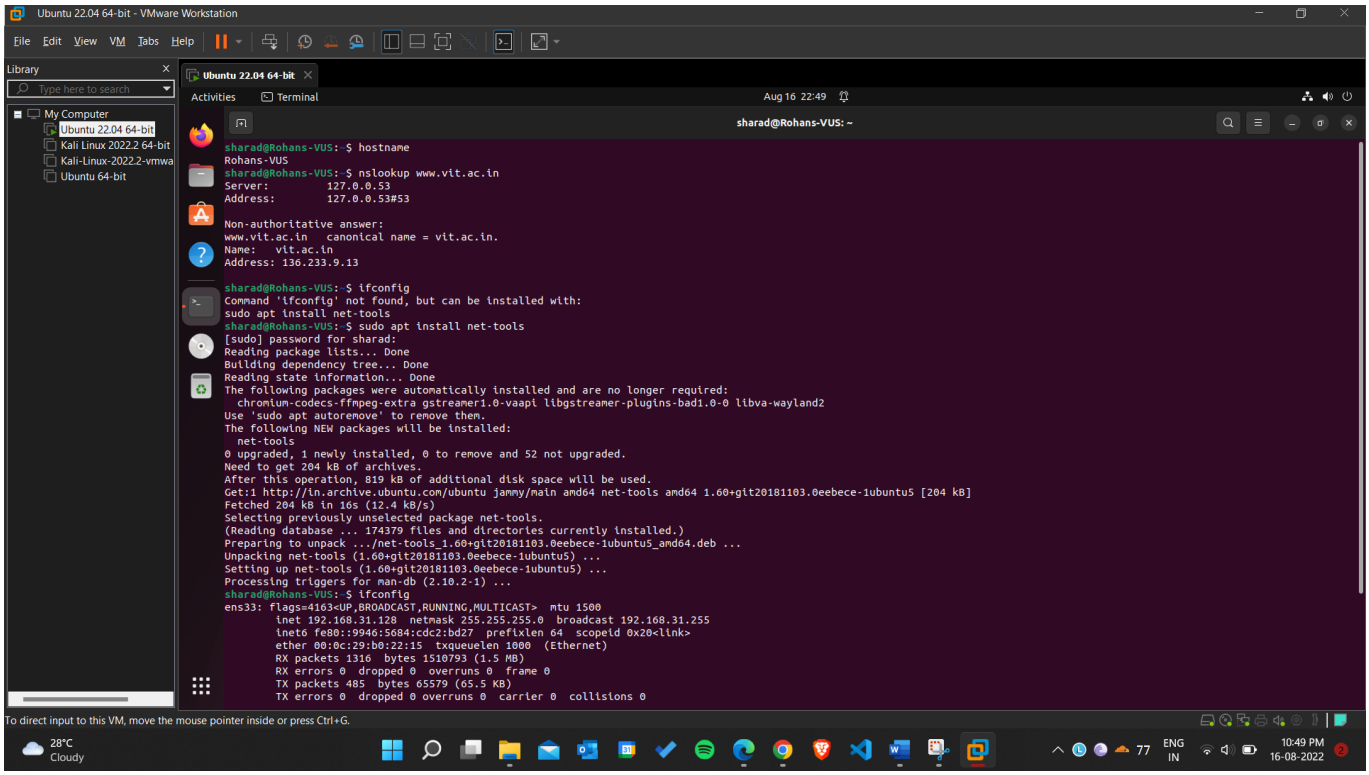
Step 7: Adding the iso file to install the OS.



Step 8: Install the OS (Ubuntu here). Then boot it up.

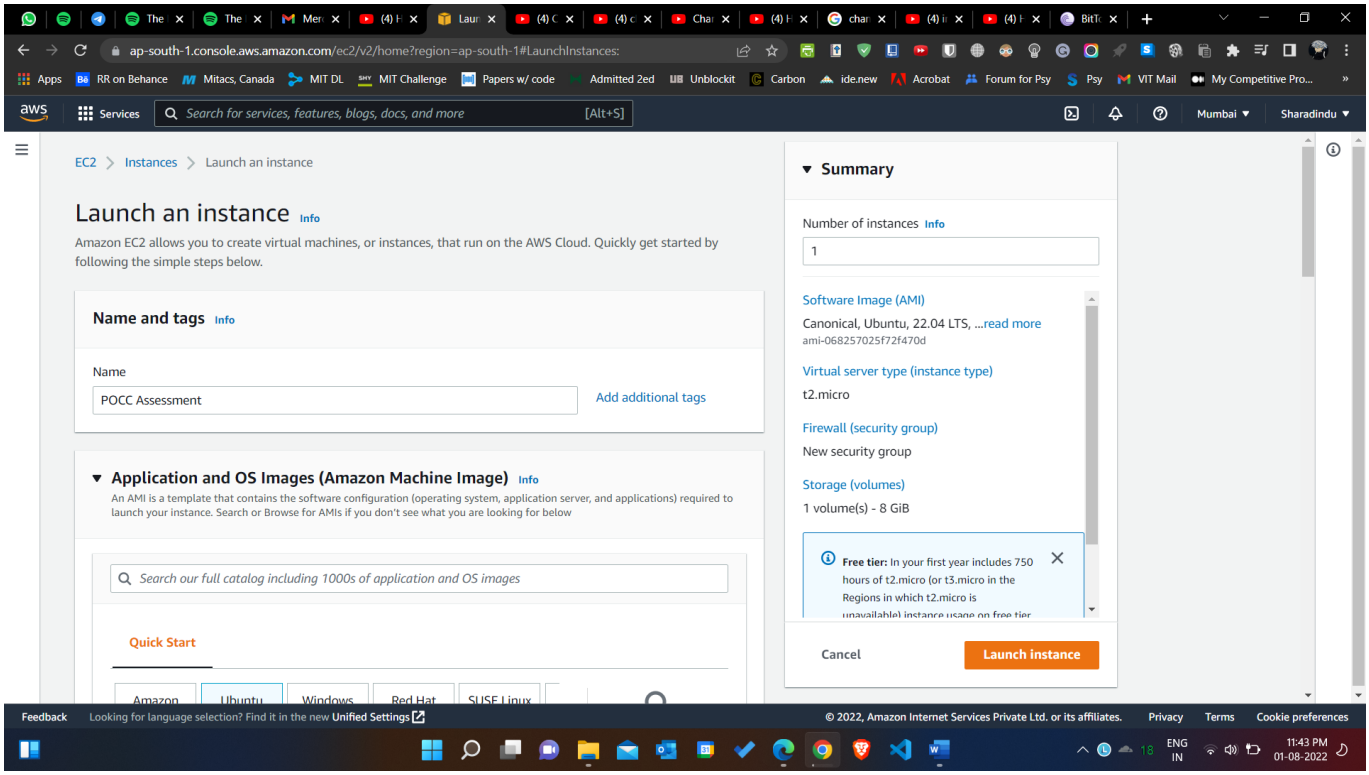


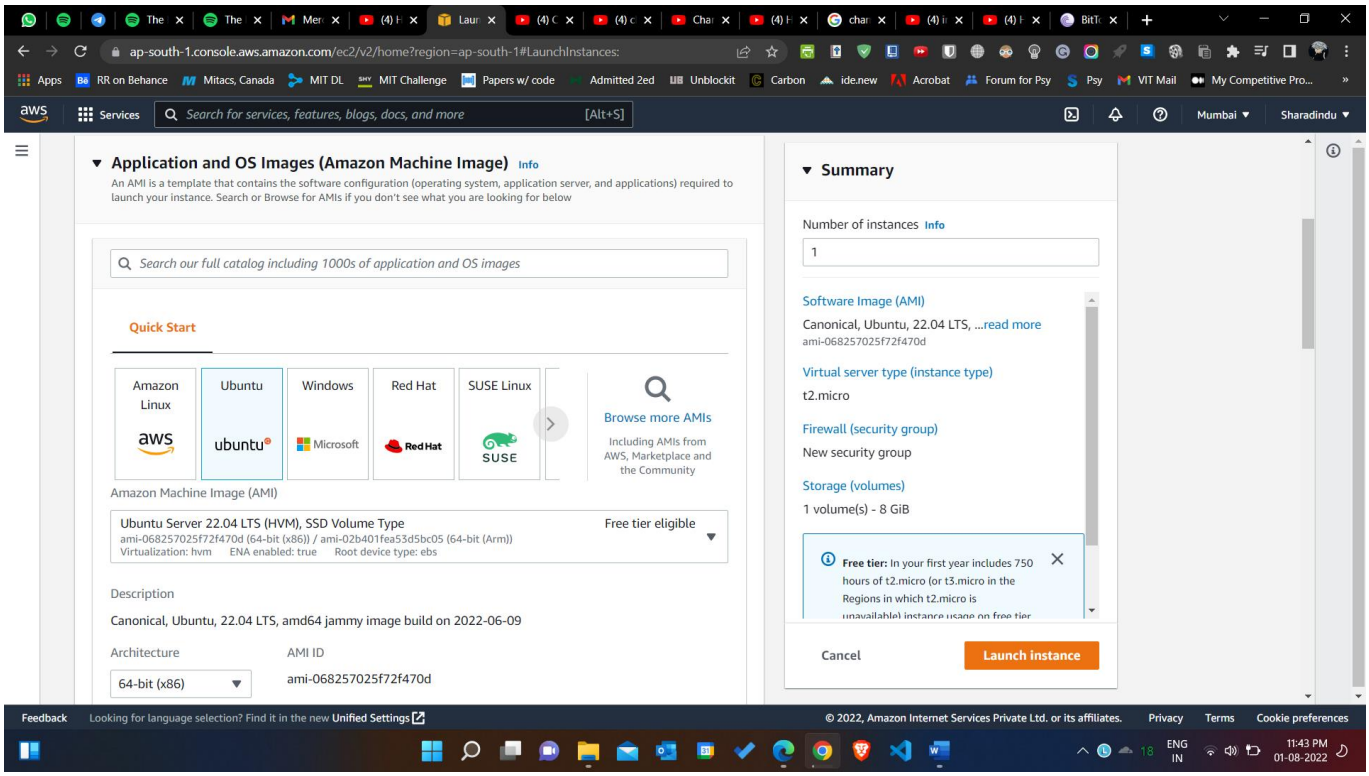
Step 9: Running basic Linux commands in the Terminal.

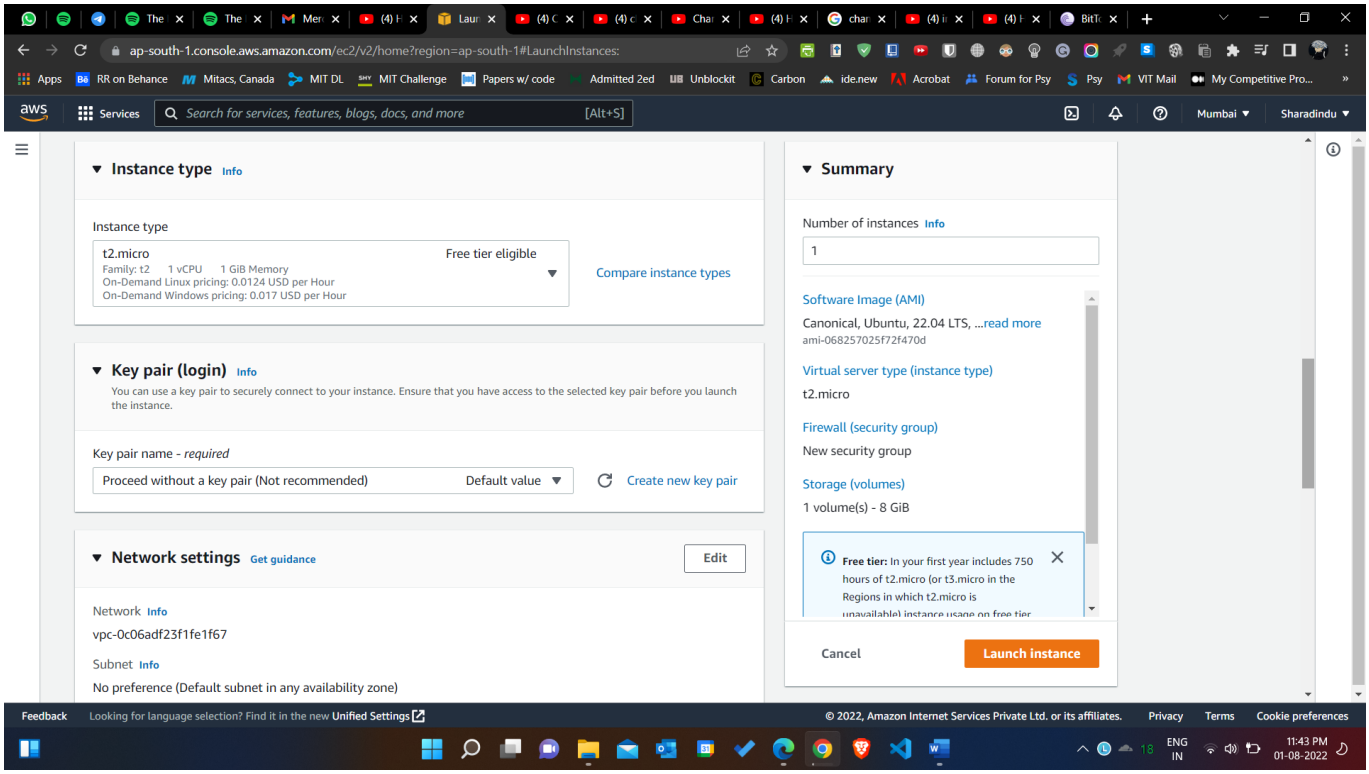


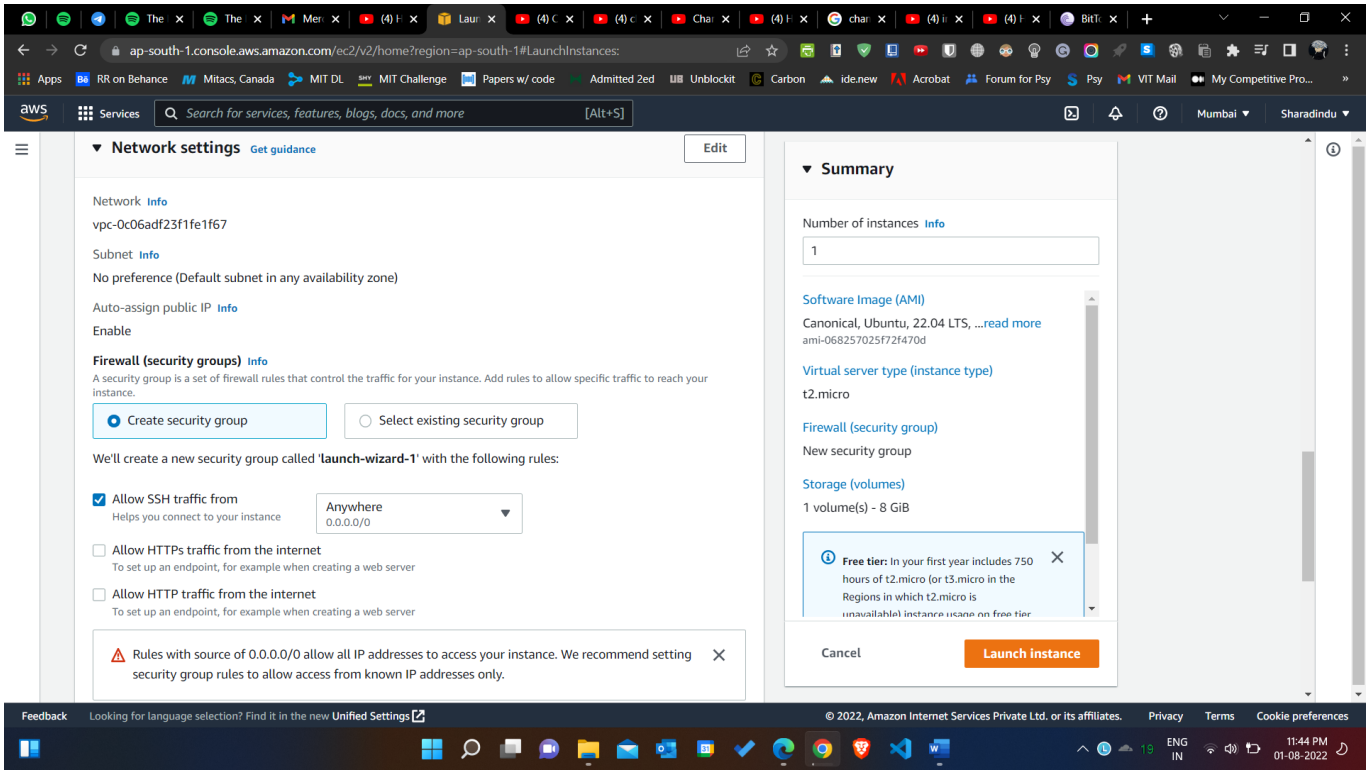
(2) Cloud:

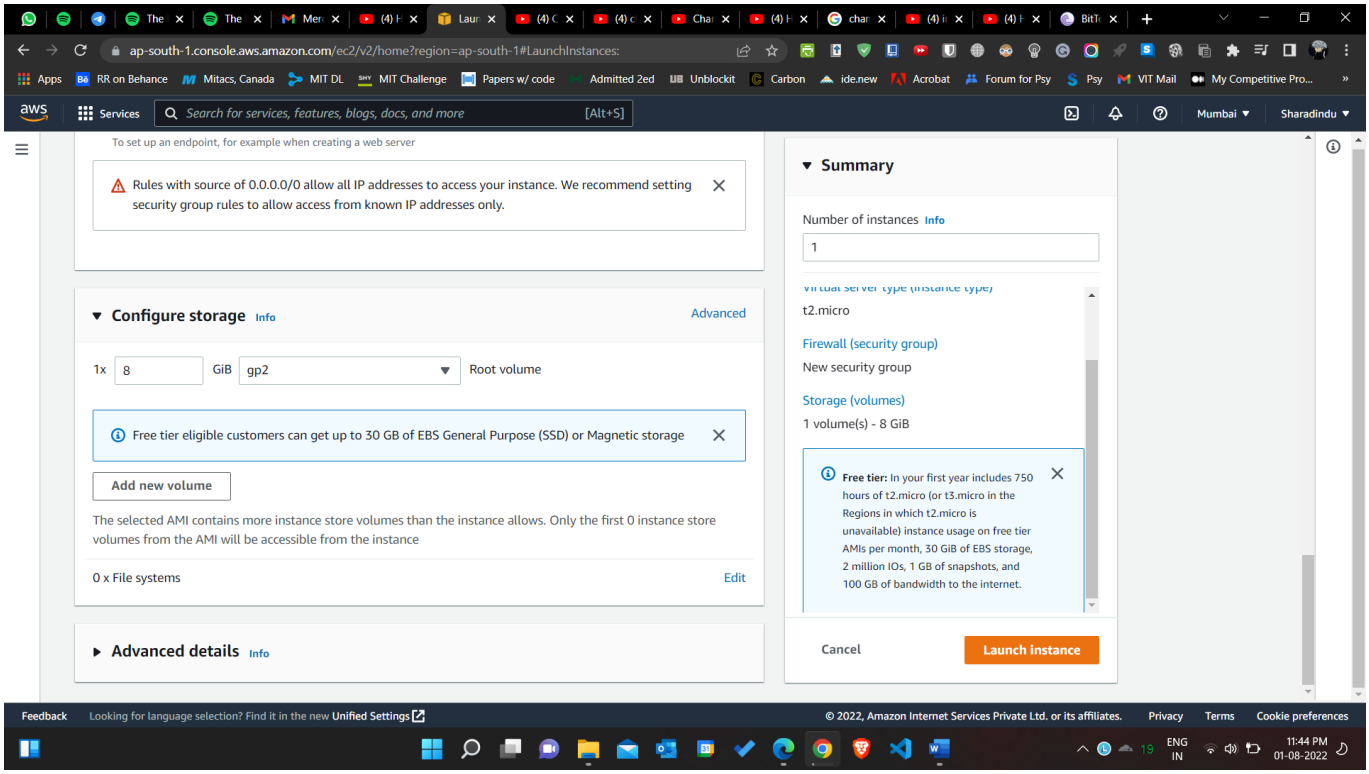
Step 1: Create an Instance with the following configuration: (a) ubuntu 64-bit architecture, (b) type: t2.micro, (c) 8 gib, gpu2 root volume



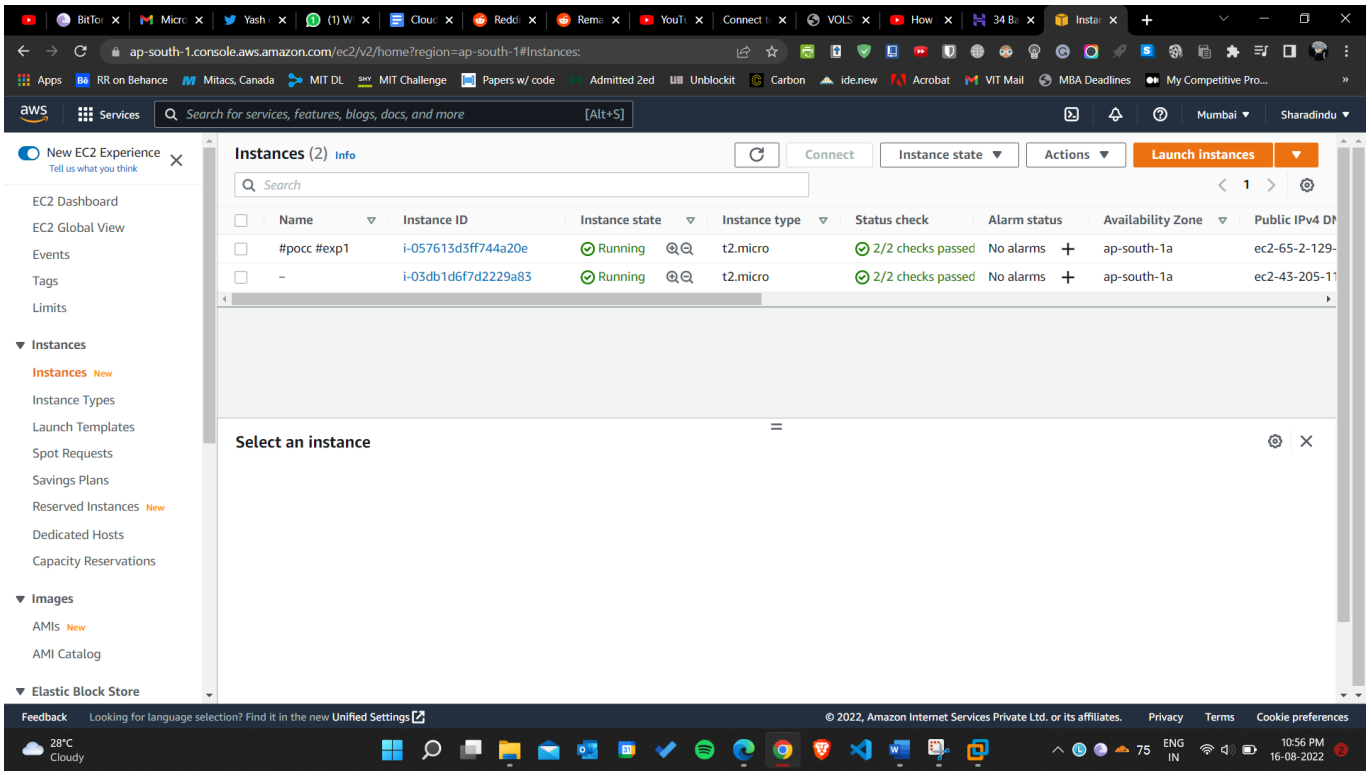


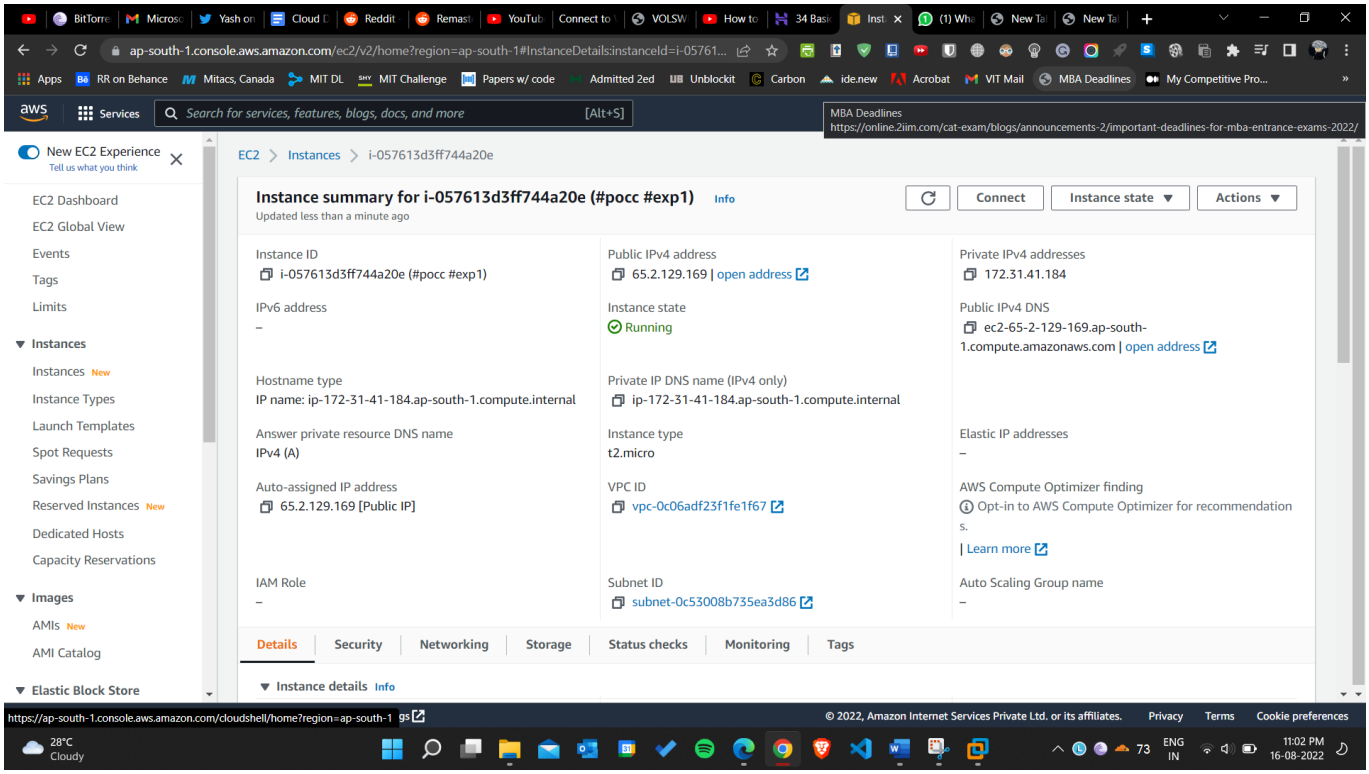

ap-south-1.console.aws.amazon.com/ec2/v2/home?region=ap-south-1#LaunchInstances:
Services
Search for services, features, blogs, docs, and more [Alt+S]
Mumbai
Sharadindu
Instance type Info
Instance type
t2.micro Free tier eligible
Family: t2 1 vCPU 1 GiB Memory
On-Demand Linux pricing: 0.0124 USD per Hour
On-Demand Windows pricing: 0.017 USD per Hour
Compare instance types
Key pair (login) Info
You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.
Key pair name - required
Proceed without a key pair (Not recommended) Default value
Create new key pair
Network settings Get guidance
Edit
Network Info
vpc-0c06adf23f1fe1f67
Subnet Info
No preference (Default subnet in any availability zone)
Summary
Number of instances Info
1
Software Image (AMI)
Canonical, Ubuntu, 22.04 LTS, ...read more
ami-068257025f72f470d
Virtual server type (instance type)
t2.micro
Firewall (security group)
New security group
Storage (volumes)
1 volume(s) - 8 GiB
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is
Cancel
Launch instance
Feedback
Looking for language selection? Find it in the new Unified Settings
© 2022, Amazon Internet Services Private Ltd. or its affiliates.
Privacy
Terms
Cookie preferences
ENG IN
11:43 PM
01-08-2022


ap-south-1.console.aws.amazon.com/ec2/v2/home?region=ap-south-1#LaunchInstances:
Services
Search for services, features, blogs, docs, and more [Alt+S]
Mumbai
Sharadindu
Network settings Get guidance
Edit
Network Info
vpc-0c06adf23f1fe1f67
Subnet Info
No preference (Default subnet in any availability zone)
Auto-assign public IP Info
Enable
Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
Create security group
Select existing security group
We'll create a new security group called 'launch-wizard-1' with the following rules:
Allow SSH traffic from
Helps you connect to your instance
Anywhere
0.0.0.0/0
Allow HTTPs traffic from the internet
To set up an endpoint, for example when creating a web server
Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.
Summary
Number of instances Info
1
Software Image (AMI)
Canonical, Ubuntu, 22.04 LTS, ...read more
ami-068257025f72f470d
Virtual server type (instance type)
t2.micro
Firewall (security group)
New security group
Storage (volumes)
1 volume(s) - 8 GiB
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is
Cancel
Launch instance
Feedback
Looking for language selection? Find it in the new Unified Settings
© 2022, Amazon Internet Services Private Ltd. or its affiliates.
Privacy
Terms
Cookie preferences
ENG IN
11:44 PM
01-08-2022



Step 2: Launch cloud instance





Conclusion:

I've created a virtual machine setup on my local host computer, and another on the cloud (in AWS). All of the steps are chronologically depicted, and screenshots at each milestone are added.

Experiment 2

Title: Nested Virtual Machines

Aim: To install the KVM emulator (VM Manager) and Configure a Nested Virtual Machine (VM under another VM) in the cloud and local machine. To Install OS images and work with a few OS commands.

Background Theory:

Virtualization Technology:

- Virtualization relies on software to simulate hardware functionality and create a virtual computer system. This enables IT organizations to run more than one virtual system – and multiple operating systems and applications – on a single server. The resulting benefits include economies of scale and greater efficiency.

Intel® VT:

- Intel® Virtualization Technology (Intel® VT) represents a growing portfolio of technologies and features that make virtualization practical by eliminating performance overheads and improving security. Intel® Virtualization Technology (Intel® VT) provides hardware assist to the virtualization software, reducing its size, cost, and complexity. Special attention is also given to reduce the virtualization overheads occurring in cache, I/O, and memory.

- Over the last decade or so, a significant number of hypervisor vendors, solution developers, and users have been enabled with Intel® Virtualization Technology (Intel® VT), which is now serving a broad range of customers in the consumer, enterprise, cloud, communication, technical computing, and many more sectors.

AMD-V®:

- AMD virtualization (AMD-V®) is a virtualization technology developed by Advanced Micro Devices. AMD-V technology takes some of the tasks that virtual machine managers perform through software emulation and simplifies those tasks through enhancements in the processor's instruction set.

- AMD-V® was first announced in 2004 and added to AMD's Pacifica 64-bit x86 processor designs. By 2006, AMD's Athlon 64 X2 and Athlon 64 FX processors appeared with AMD-V technology, and today, the technology is available on Turion 64 X2, second- and third-generation Opteron, Phenom, and Phenom II processors.

KVM Tool:

- KVM (for Kernel-based Virtual Machine) is a full virtualization solution for Linux on hardware containing virtualization extensions. It consists of a loadable kernel module that exposes virtualization APIs to userspace for use by applications such as QEMU.

- Specifically, KVM lets you turn Linux into a hypervisor that allows a host machine to run multiple, isolated virtual environments called guests or virtual machines (VMs).

Requirements for conducting the experiment:

Local Instance:

- Oracle VM VirtualBox (or)
  VMware Workstation Player (or)
  VMware Workstation Pro (or)
  Any other Virtual Machine Setup

- Any OS ISO Image (preferably Ubuntu)

- Virtual Machine with Guest Operating System

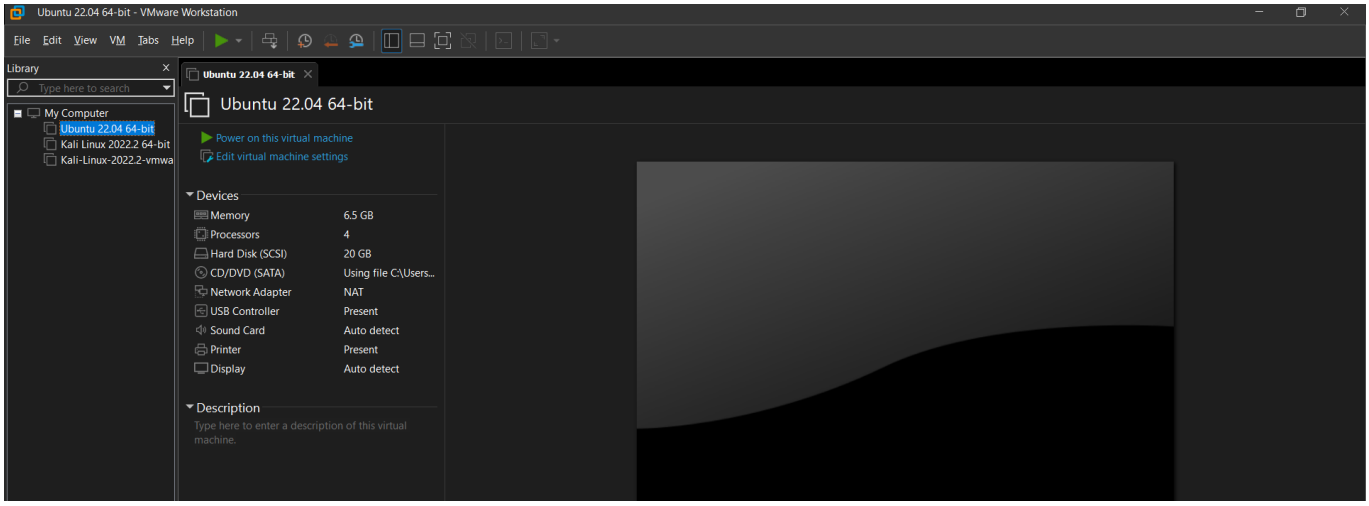
Cloud:

- AWS Licence

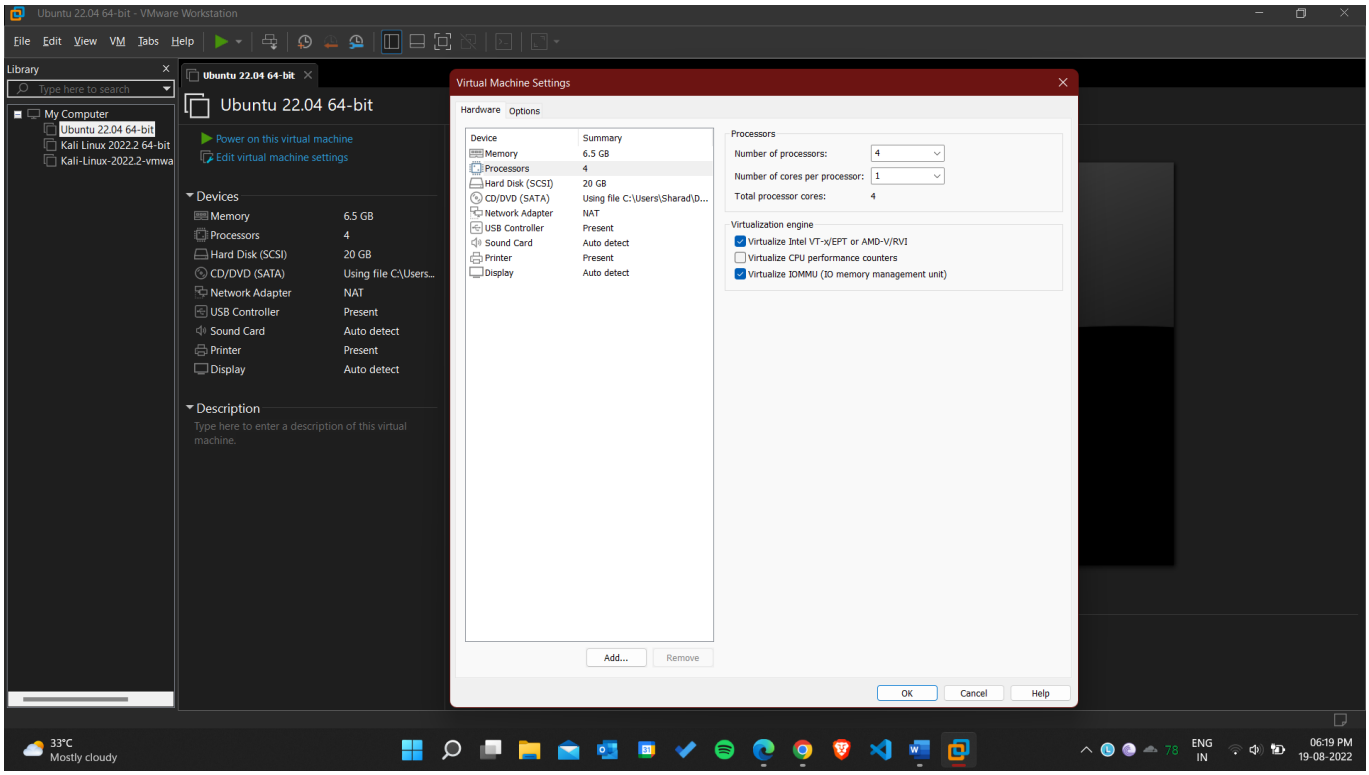
Procedure & Screenshots:

(1) Local Instance:

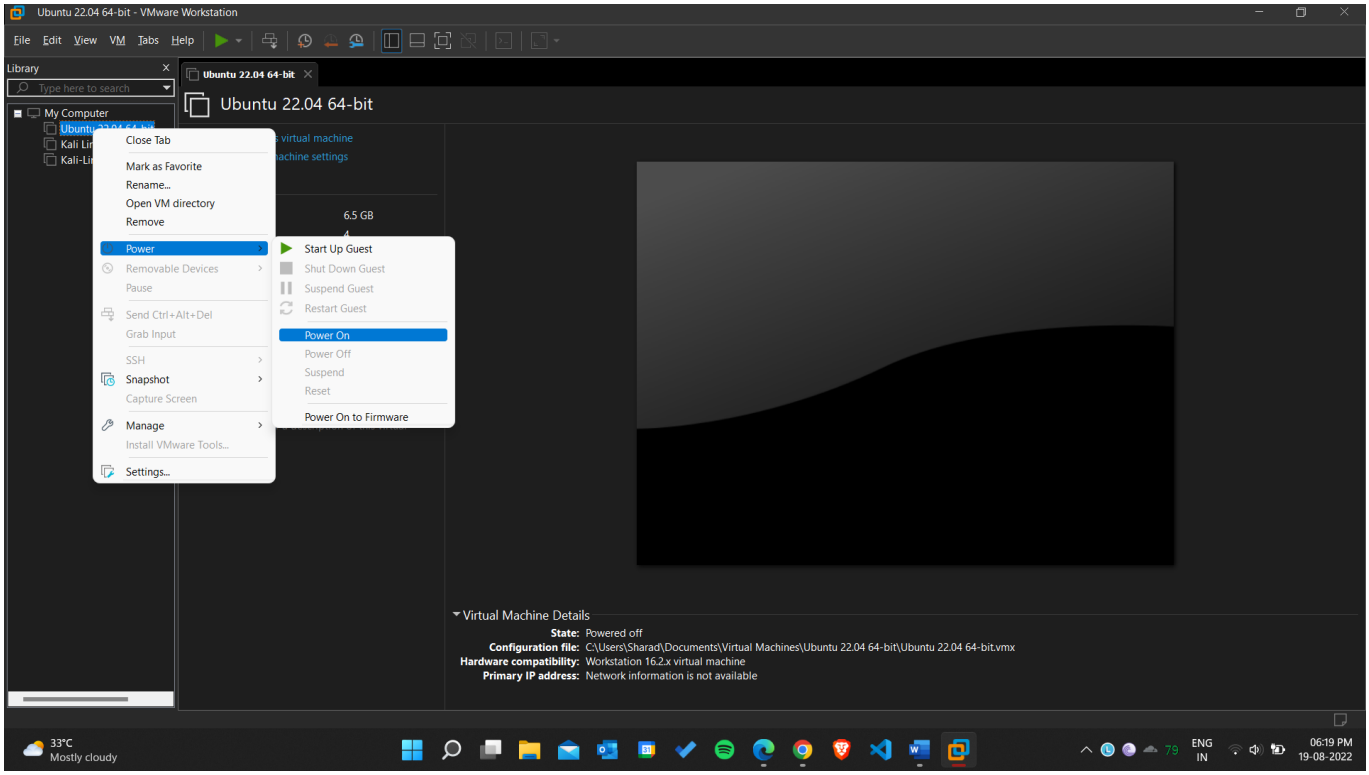
Step 1: Open VMWare Workstation Pro and select the Virtual Machine (here, Ubuntu 22.04 64-bit which I'd installed in Experiment 1)



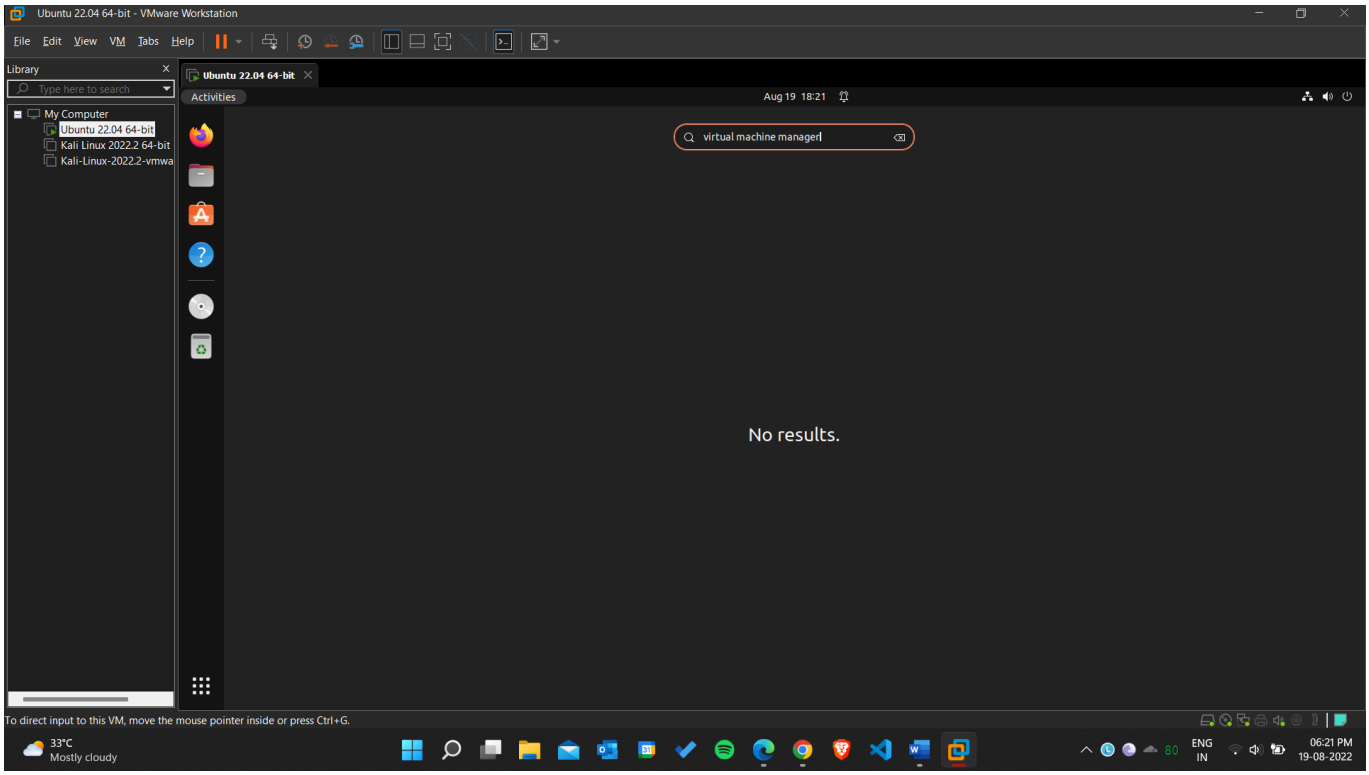
Step 2: Then go to VM (at the top bar) → Settings → Processors, and enable the options that are under Virtualization engine 'Virtualize Intel VT-x/EPT or AMD-V/RVI' and 'Virtualize IOMMU' and then click on 'OK'.



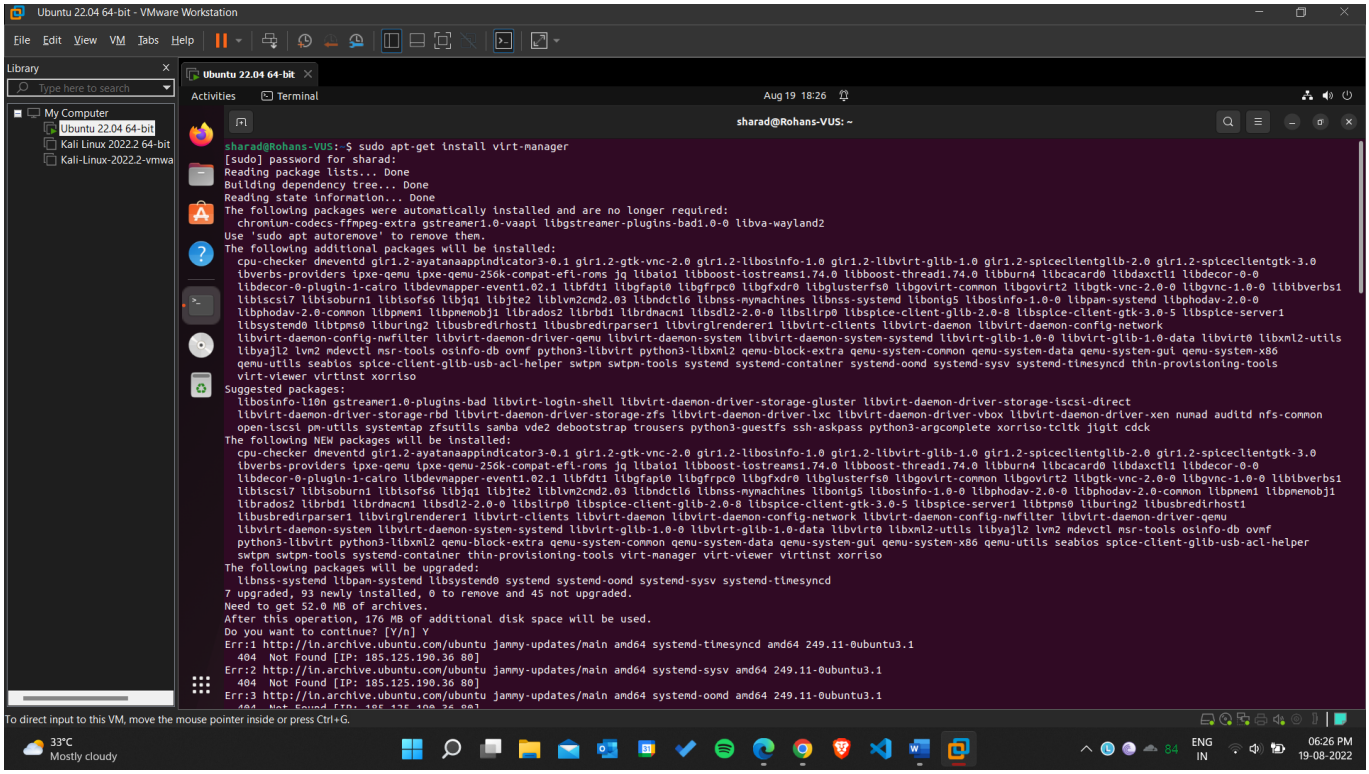
Step 3: Now after enabling the options ‘Power On’ the OS.



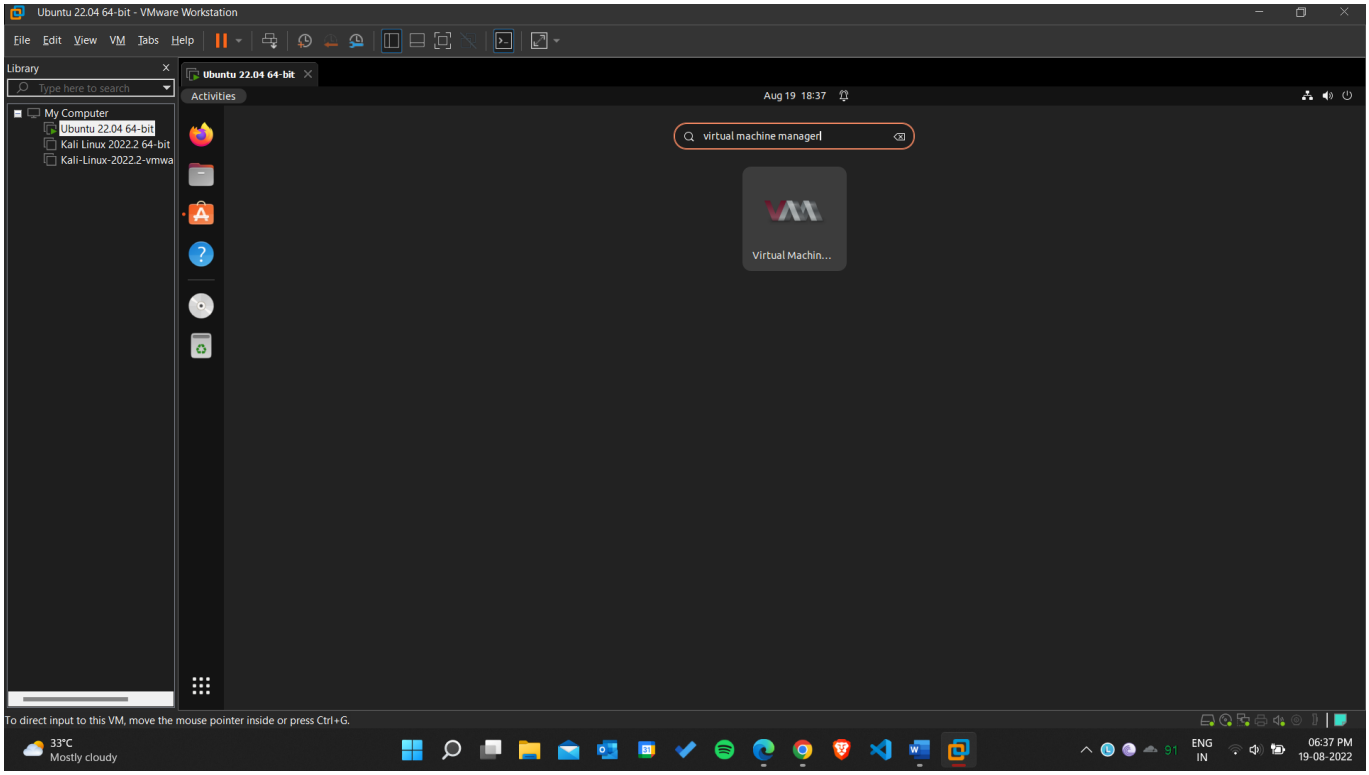
Step 4: Search for ‘virtual machine manager’. If it is not installed go to the terminal.



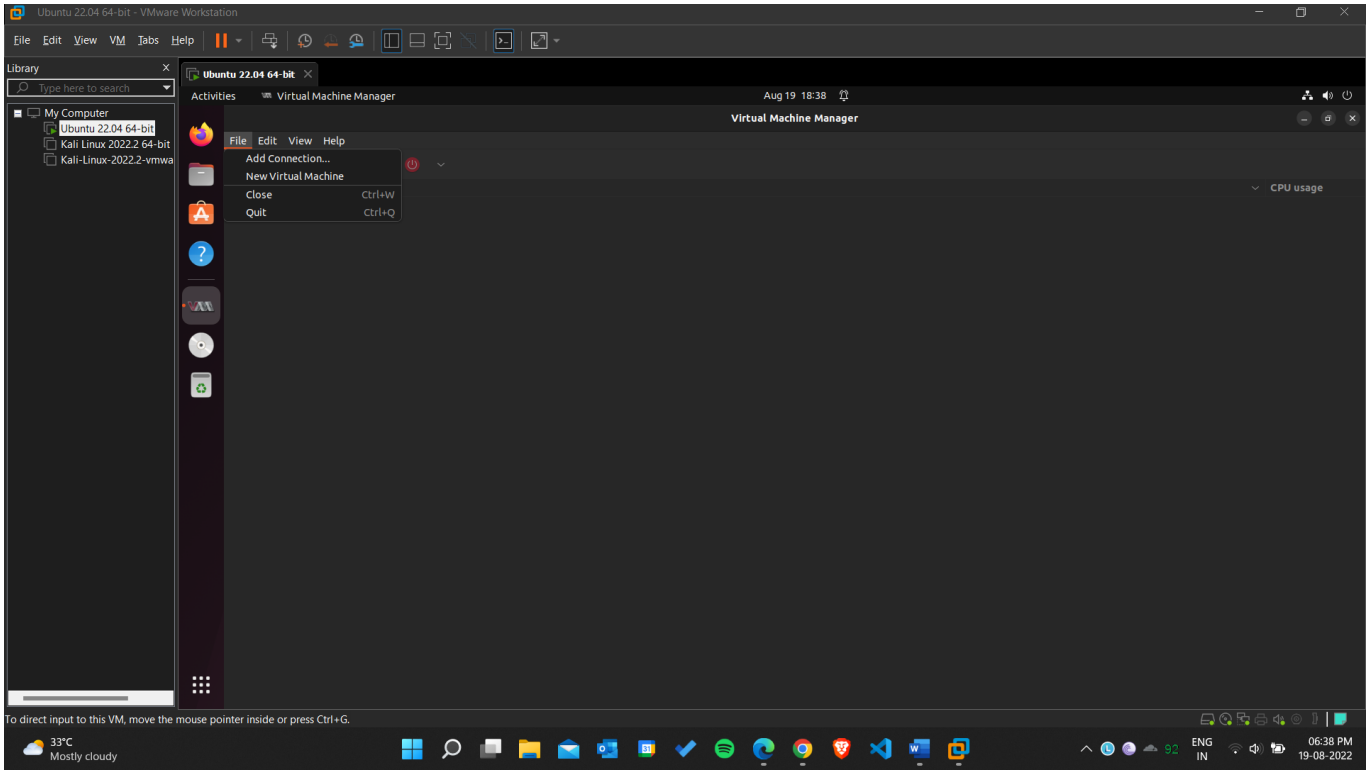
Step 5: Type in ‘sudo apt-get install virt-manager’ to install the virtual machine manager. If this way, it shows any error, go to Applications → Ubuntu Software Center → Search for "virt-manager" & install.



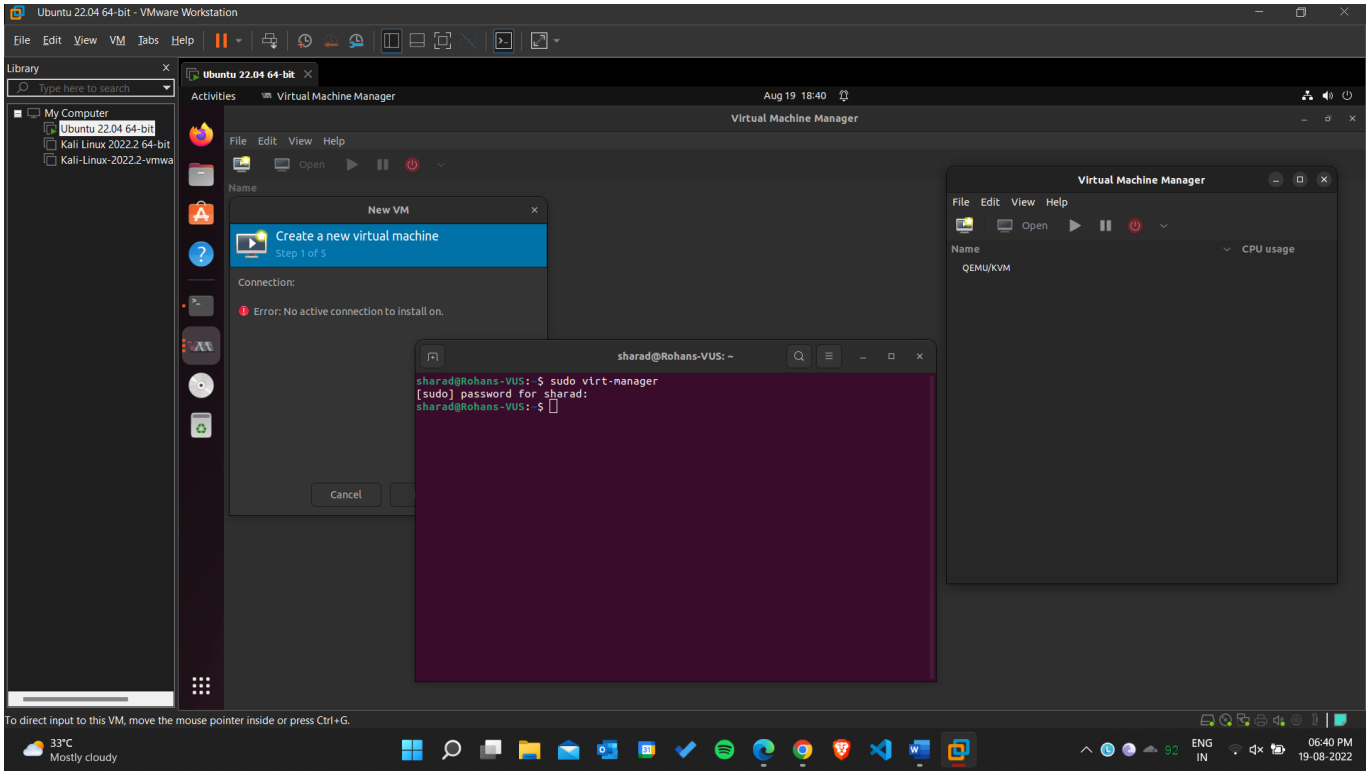
Step 6: Looks like our manager is here now.



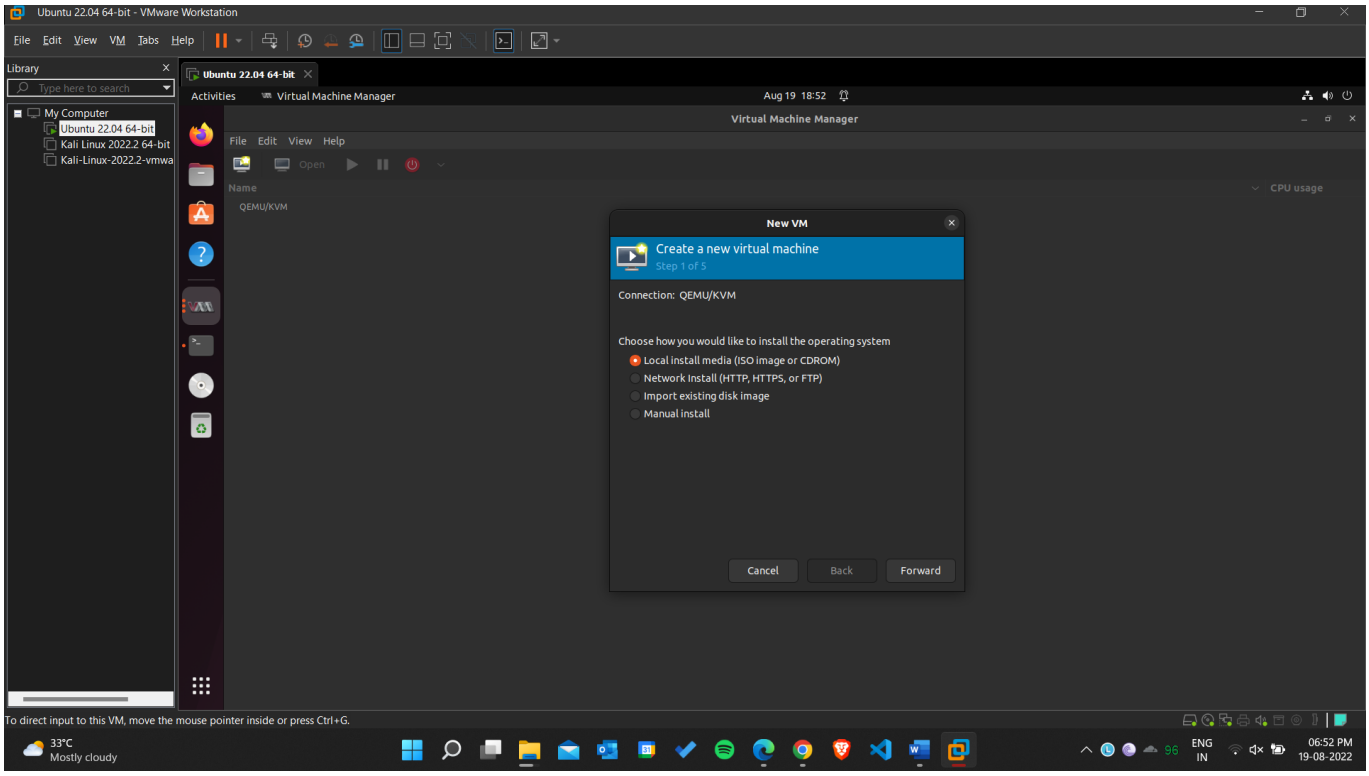
Step 7: Open the virtual machine manager, Click on ‘File’ and then click on ‘New Virtual Machine’.



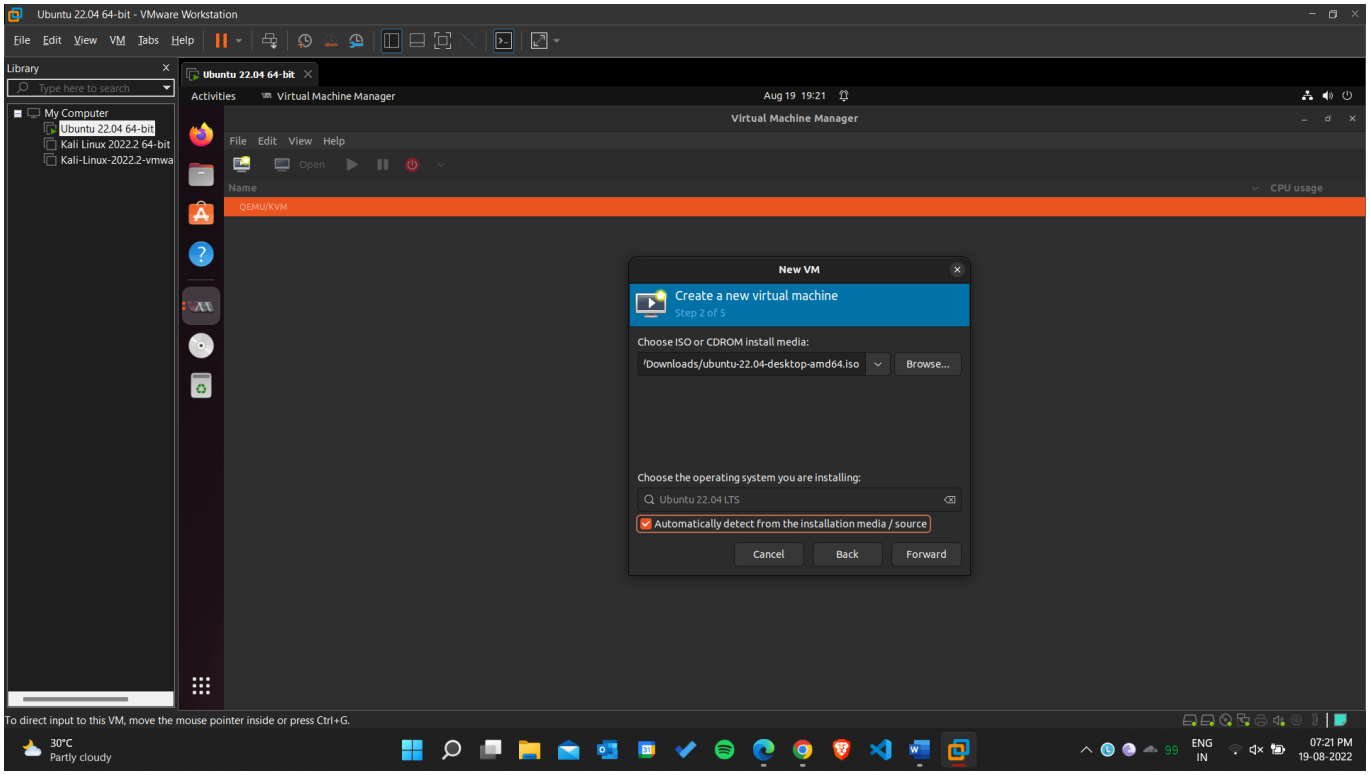
Step 8: Sometimes after doing so, it might show that there isn't any active connection. To mitigate this, just open the Terminal, and type in 'sudo virt-manager'. It'll open the manager again, in another window, which has the connection established.



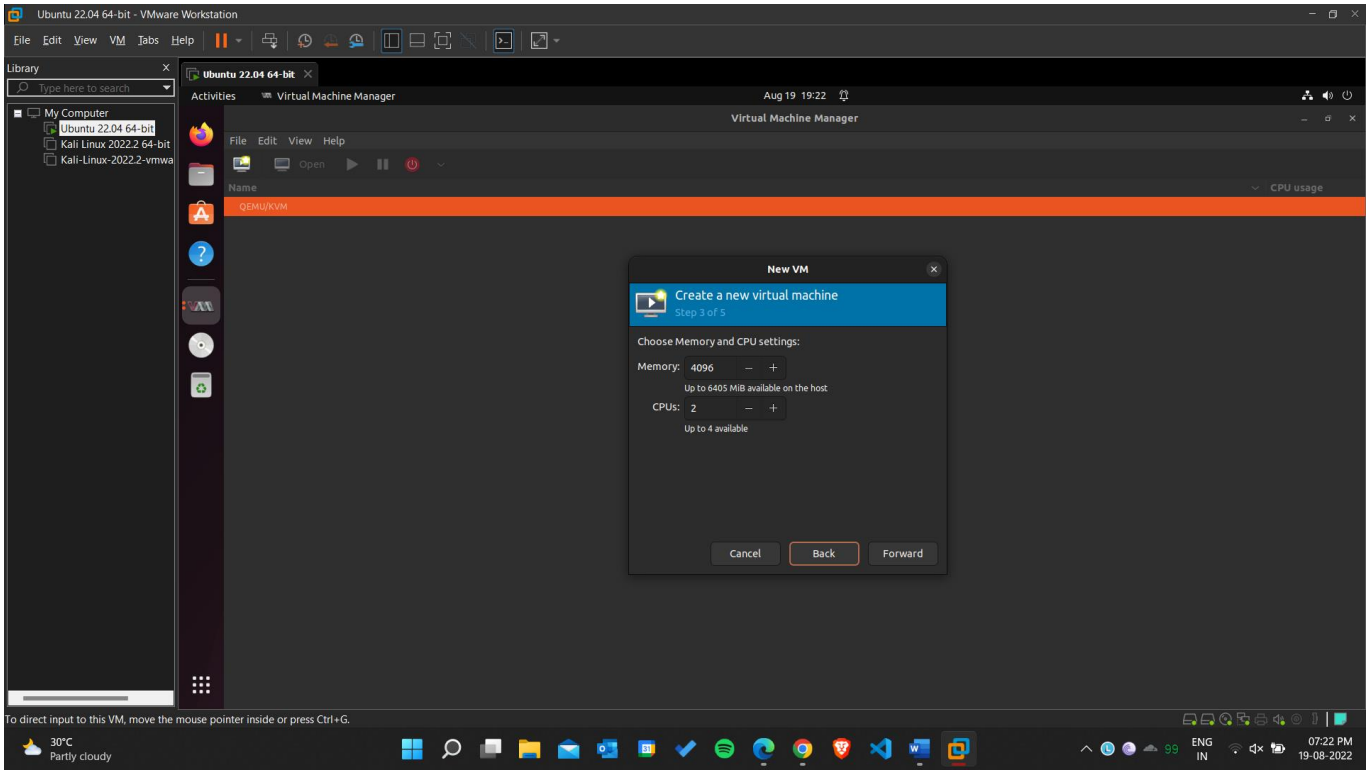
Step 9: In this new window, click on File → New Virtual Machine. Then Select ‘Local install media’ and click on ‘Forward’.



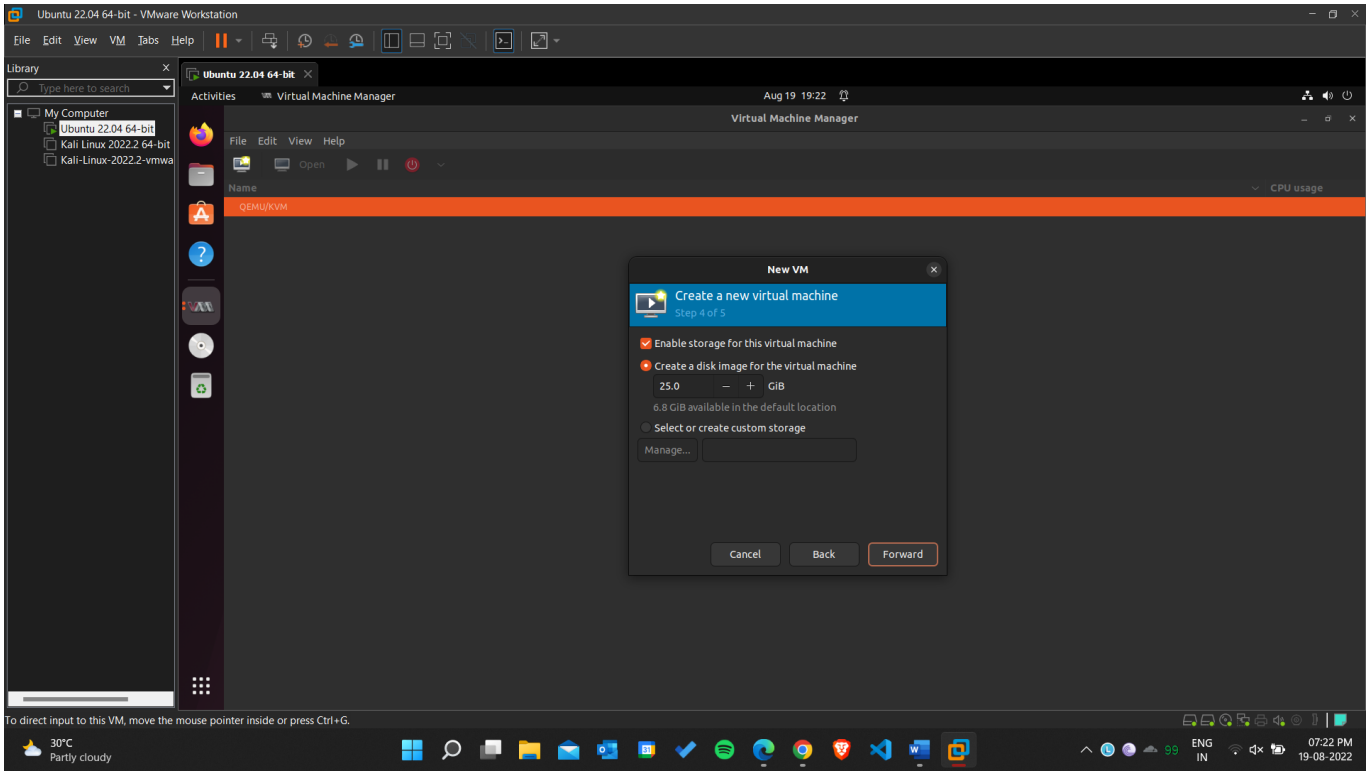
Step 10: Select the iso file.



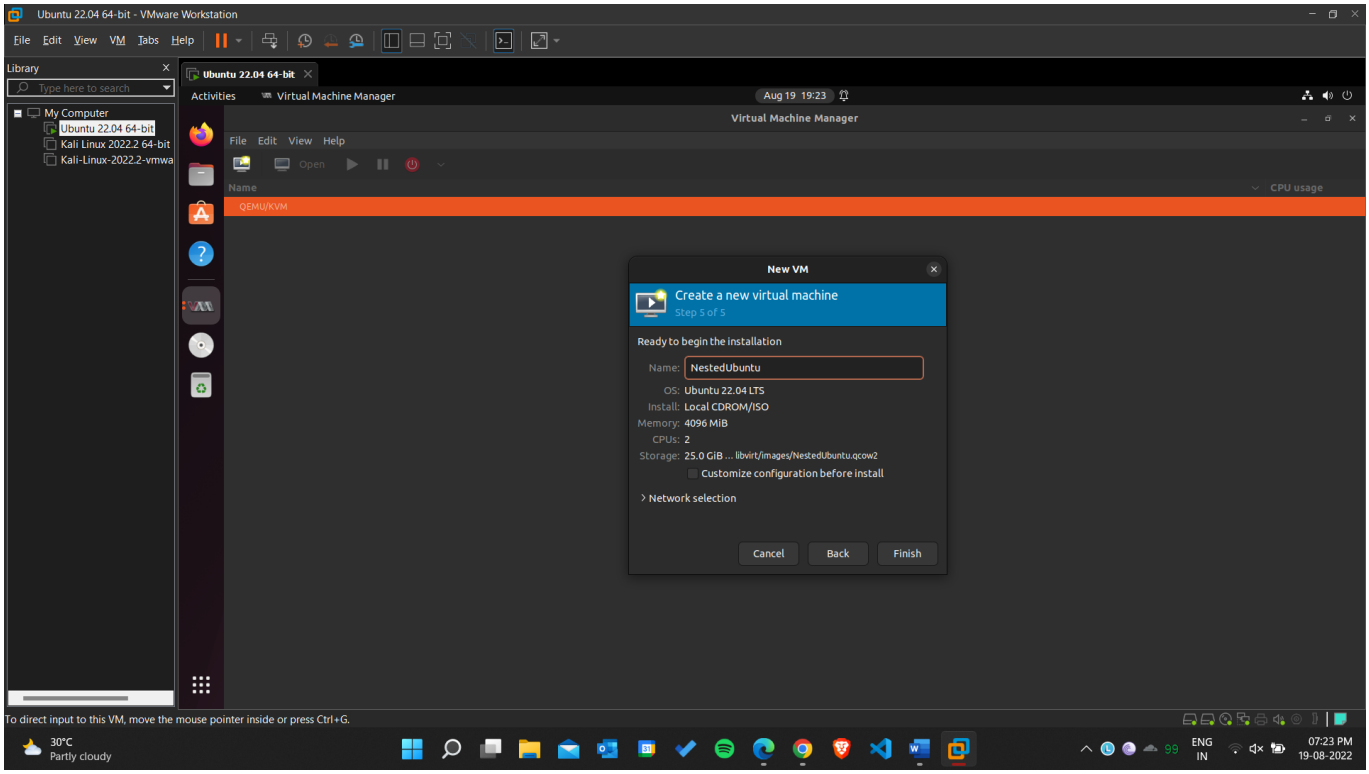
Step 11: Allocate the memory and CPU to the Virtual Machine Manager



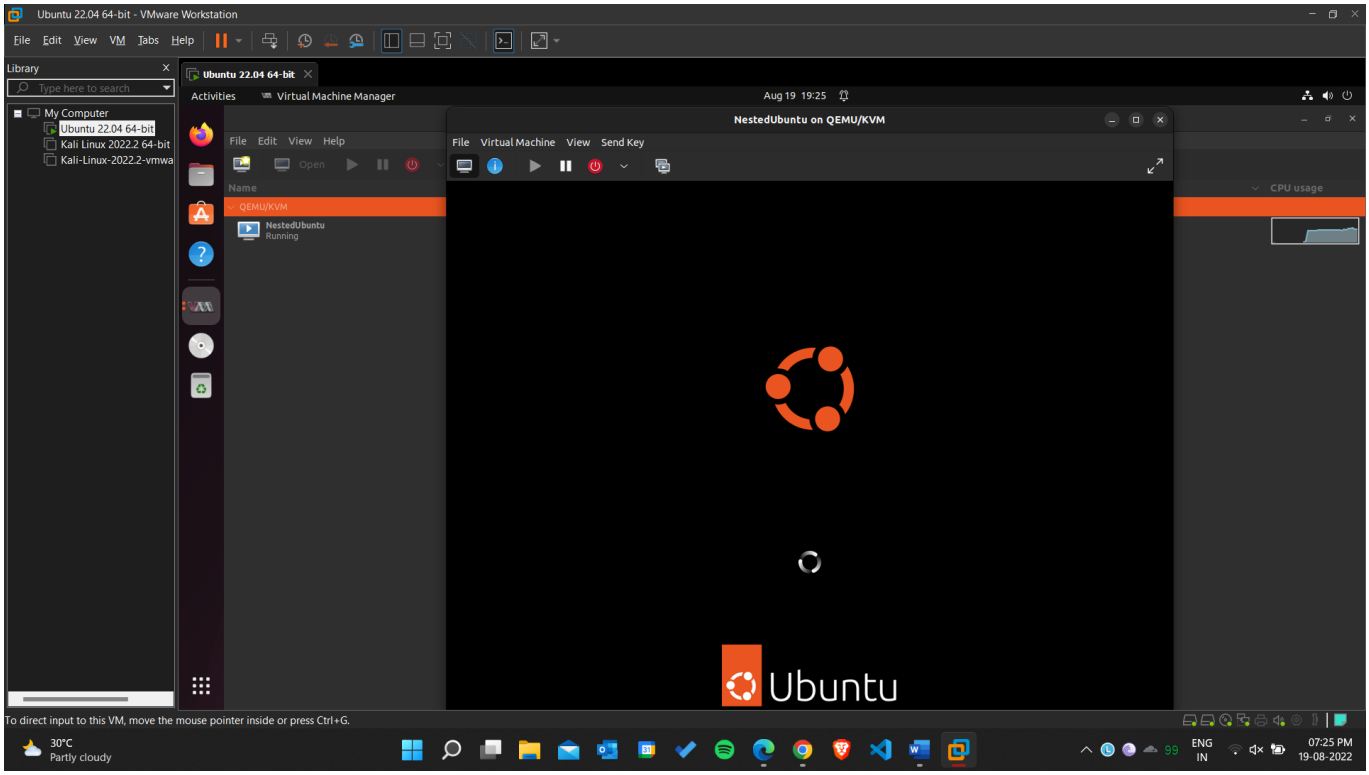
Step 12: Allocate disk size to the new VM



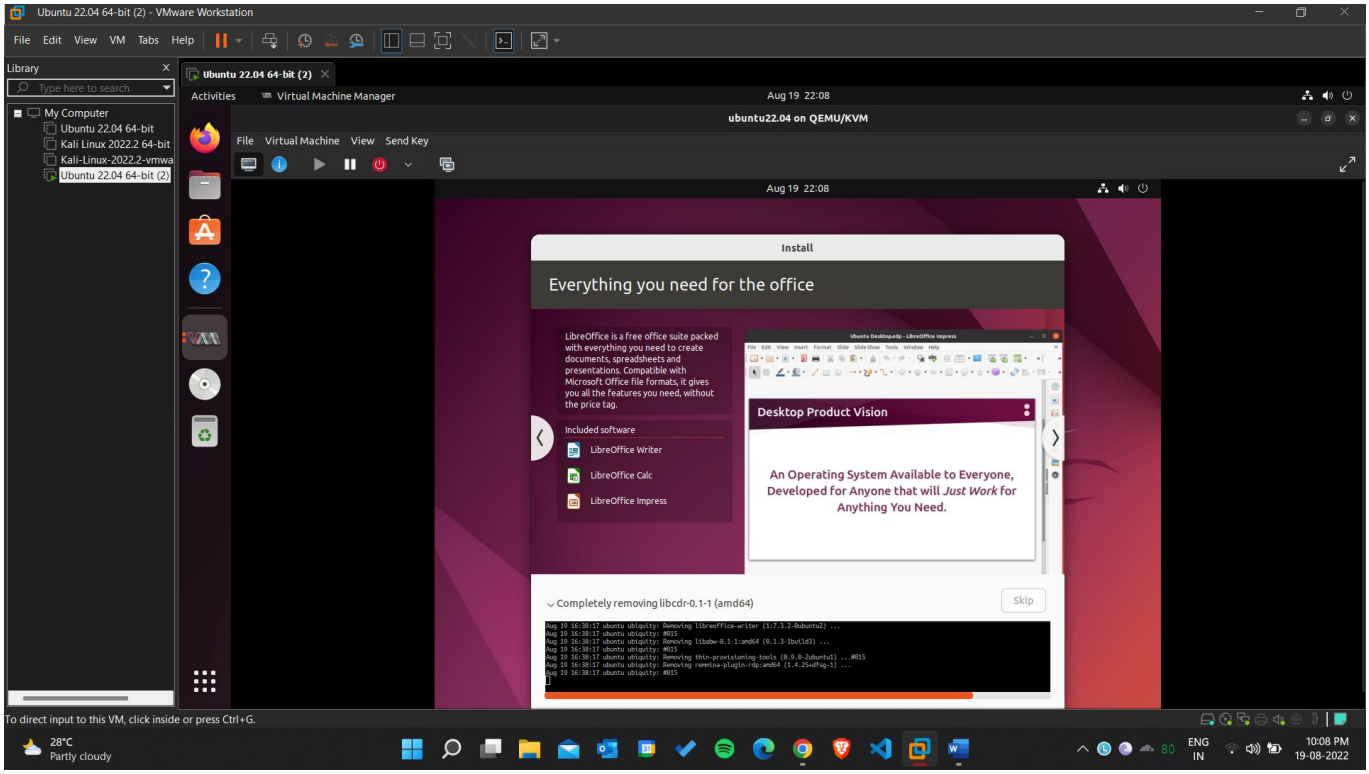
Step 13: Give a name to the nested VM.



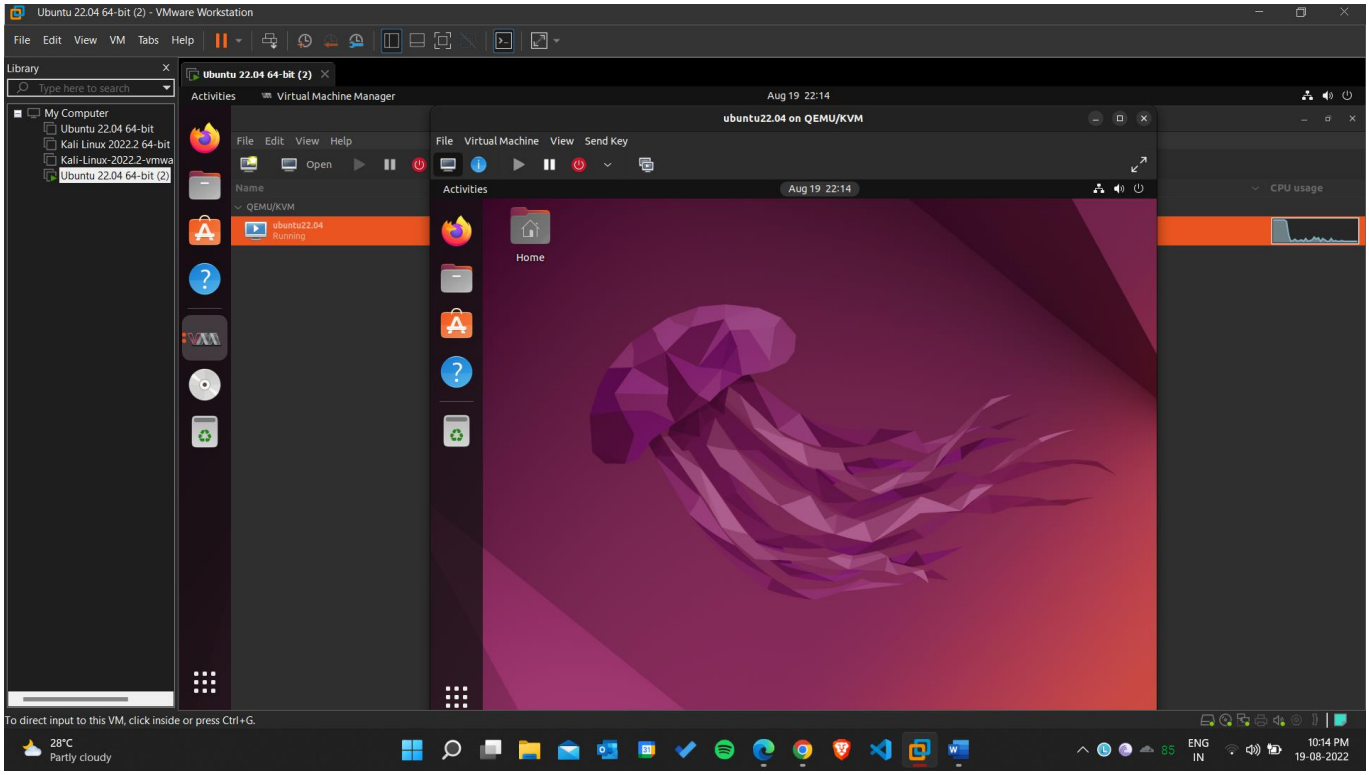
Step 14: After clicking Finish, OS in Nested VM starts installing



Step 15: Getting through the installation.

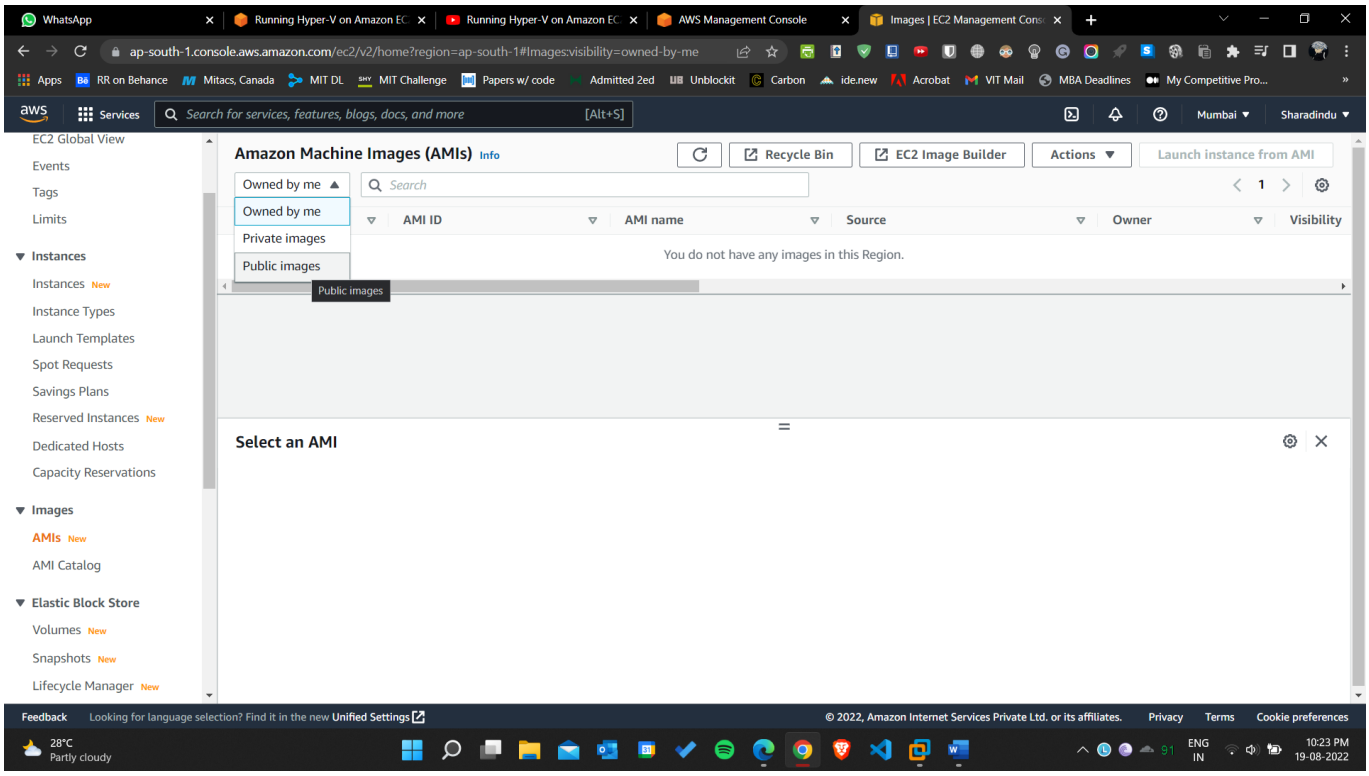


Step 16: OS in Nested VM has been successfully installed.

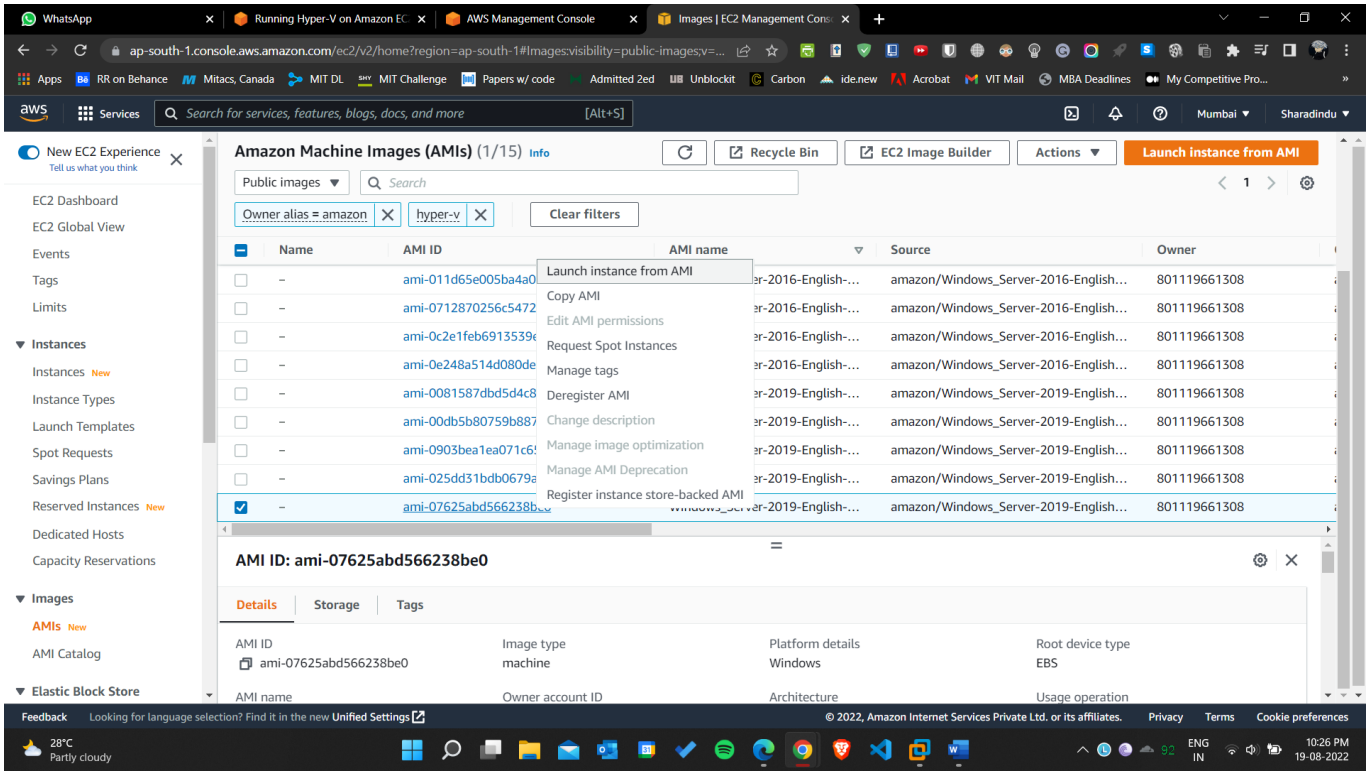


(2) Cloud:

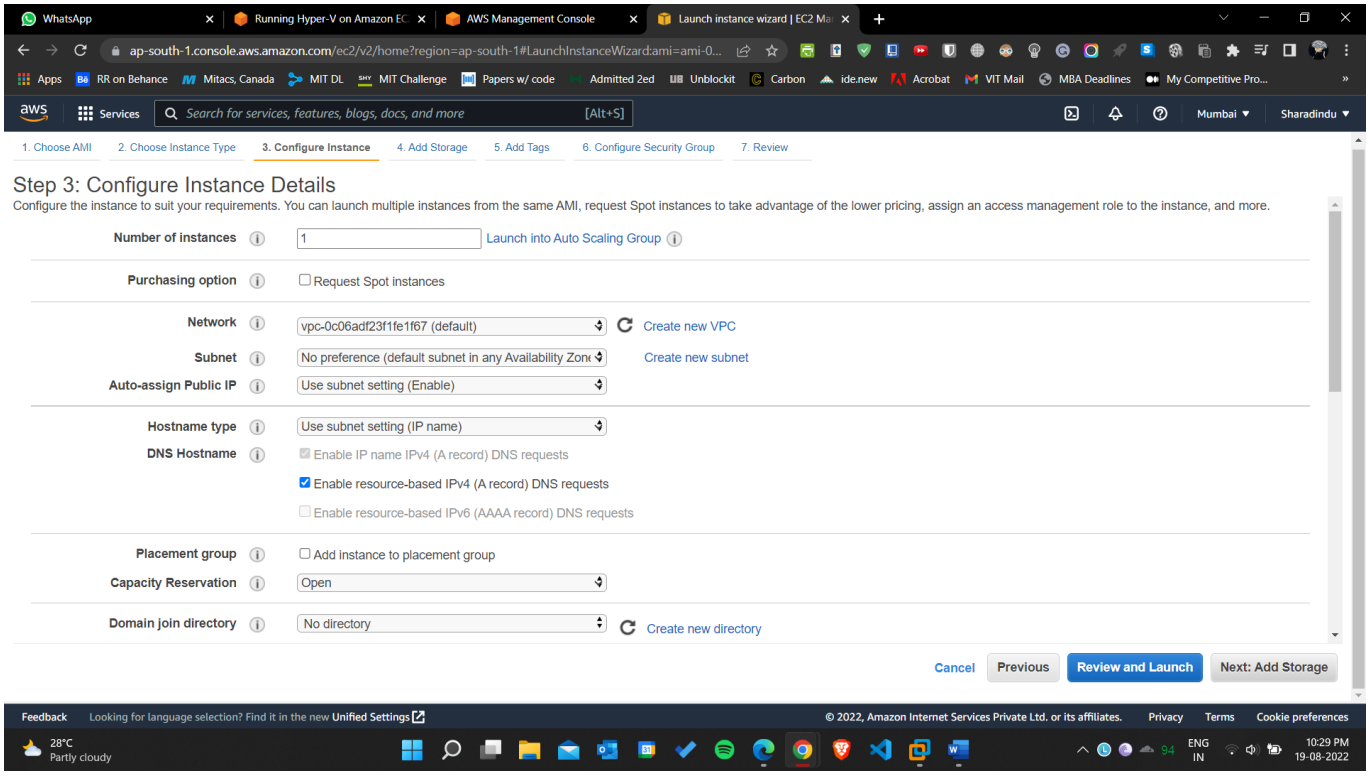
Step 1: In the dashboard, go to the Images, and click on AMIs. Click on ‘Public images’.



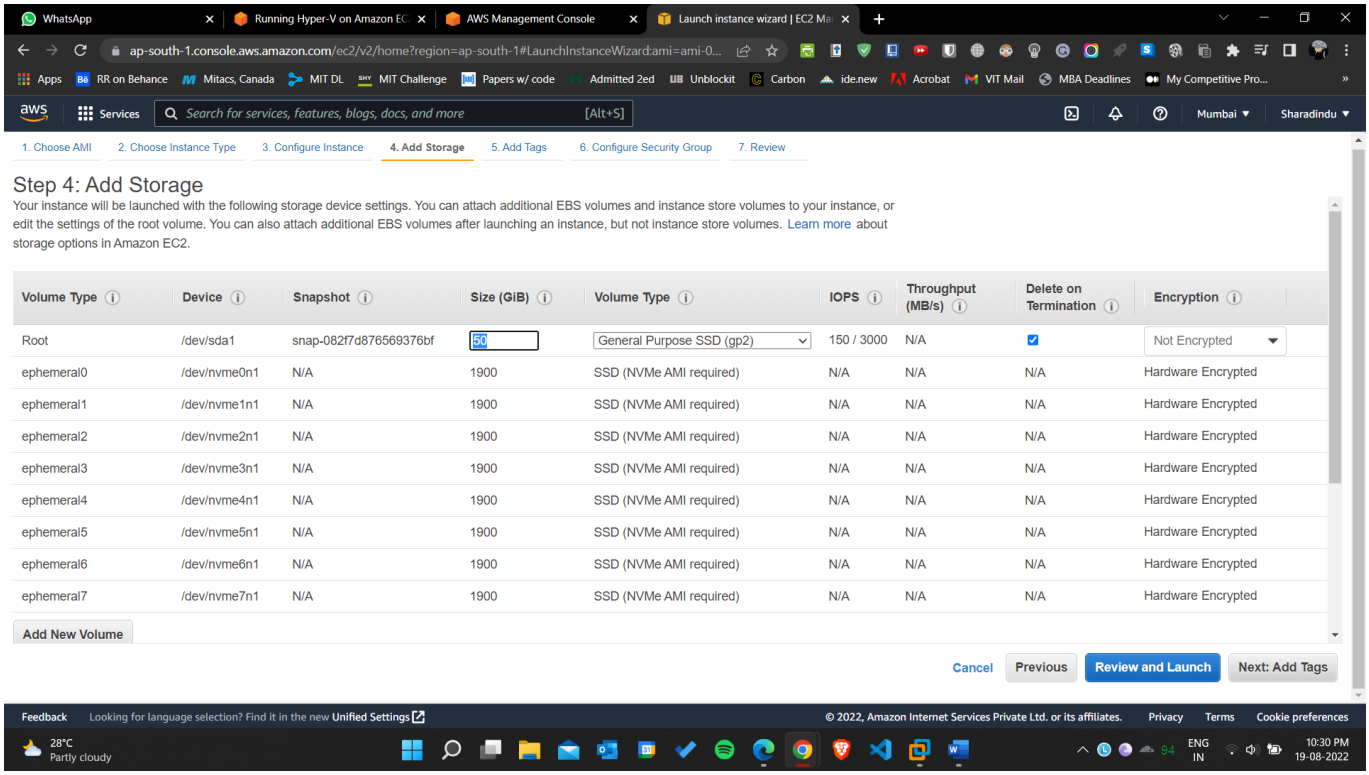
Step 2: Then, in the search bar, add ‘Owner alias = amazon’, and ‘hyper-v’. Select the most recent AMI from there.



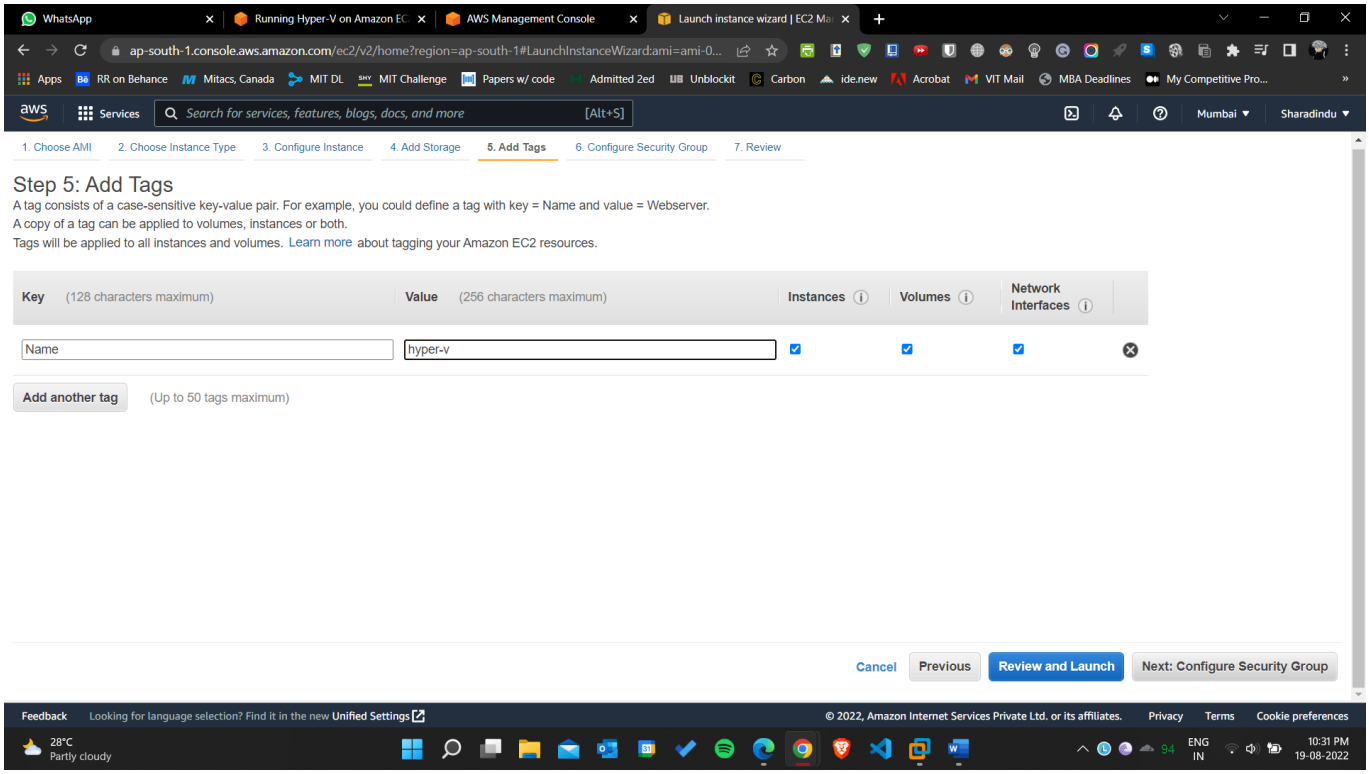
Step 3: Launch the instance from the AMI and Configure it.



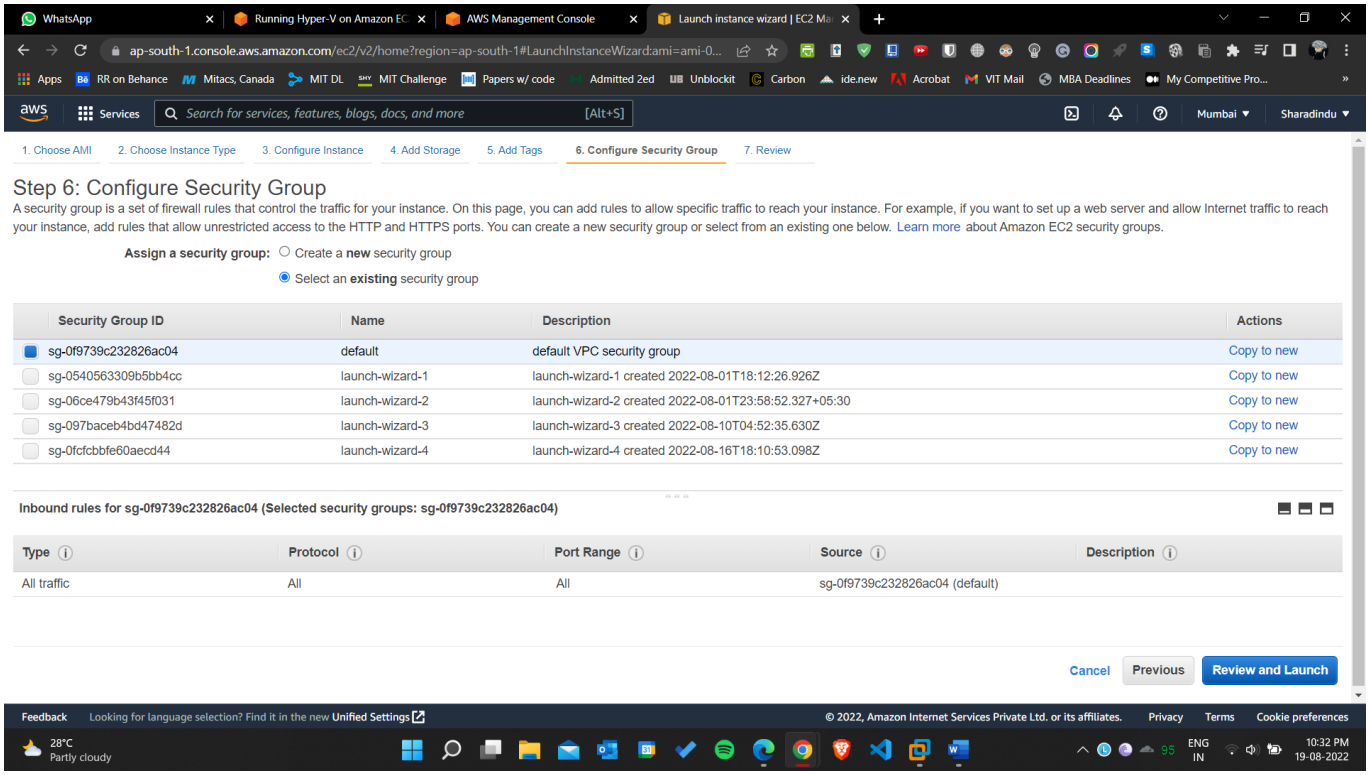
Step 4: Add storage. Increase it from 30 to 50 GiB



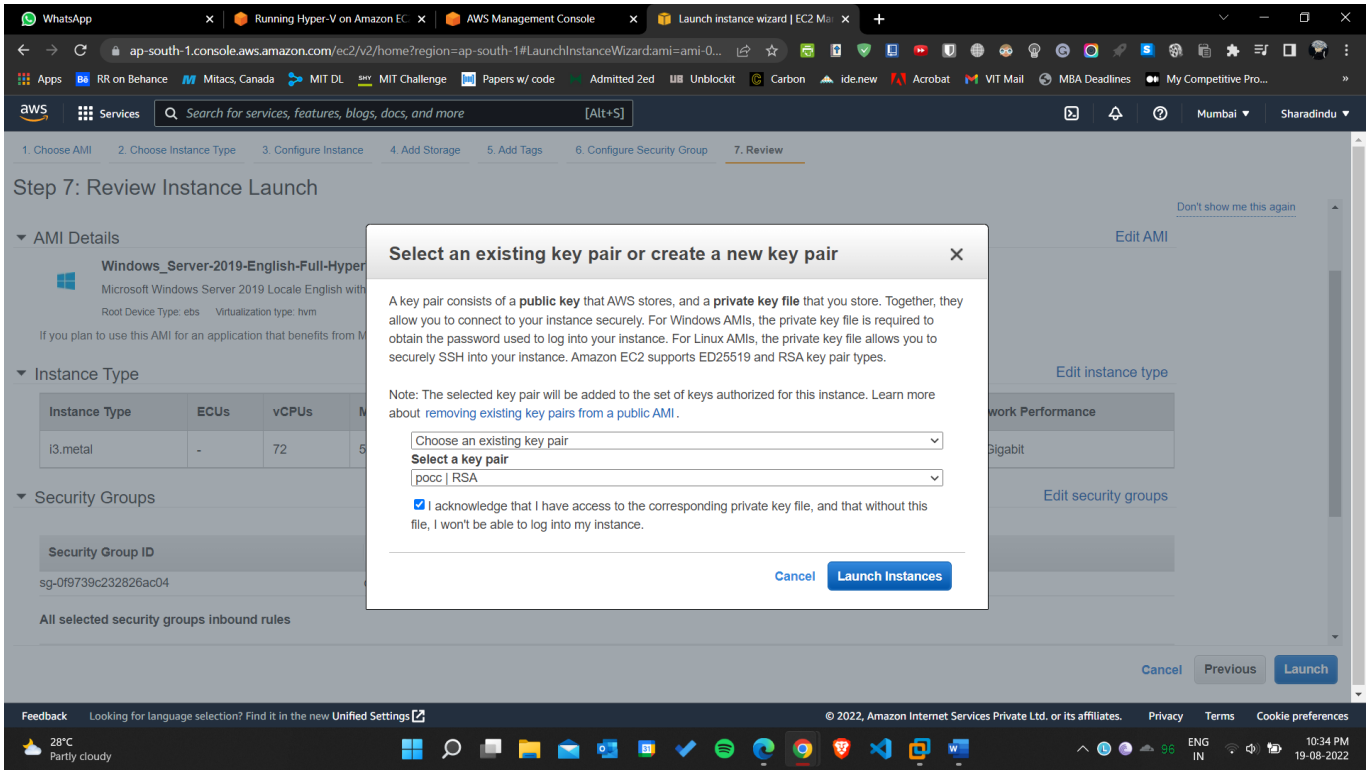
Step 5: Add tag, ‘hyper-v’.



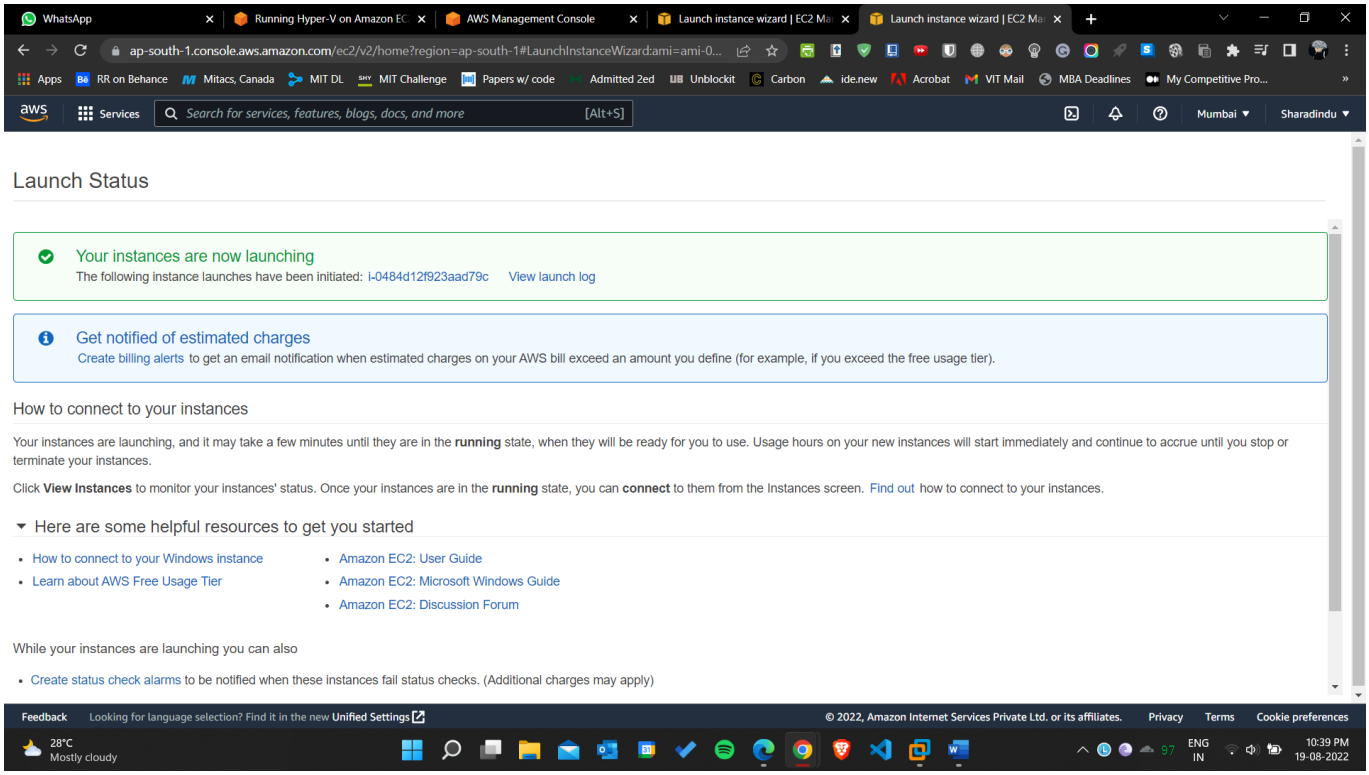
Step 6: Configure the Security Group. Better to just go with the default option.



Step 7: Select the existing key pair (which was made in Experiment 1).



Step 8: Launch the instance.



Thereafter, we need Hyper-V (which is only available on Windows 10 Pro versions) and use that to create a Nested VM in AWS, which is out of the scope of my laptop. So I'm abandoning this part.

Conclusion: I've successfully created a nested VM in my local machine (inside VMWare), and partially done the process of the nest VM on the Cloud.

Experiment 3

Title: SSH Tunnelling

Aim: To Create an SSH tunnel between our server in the local machine and remote clients in EC2 instances and test the connections.

Background Theory:

SSH Tunnelling:

- SSH tunnelling, or SSH port forwarding, is a method of transporting arbitrary data over an encrypted SSH connection. SSH tunnels allow connections made to a local port (that is, to a port on your own desktop) to be forwarded to a remote machine via a secure channel.

- To protect our network services, not all of them are reachable directly from outside the ENCS network. If you are offsite and need to access a resource that is protected in this way, you can use ssh to tunnel through an accessible resource to reach the protected resource.

Usage of SSH:

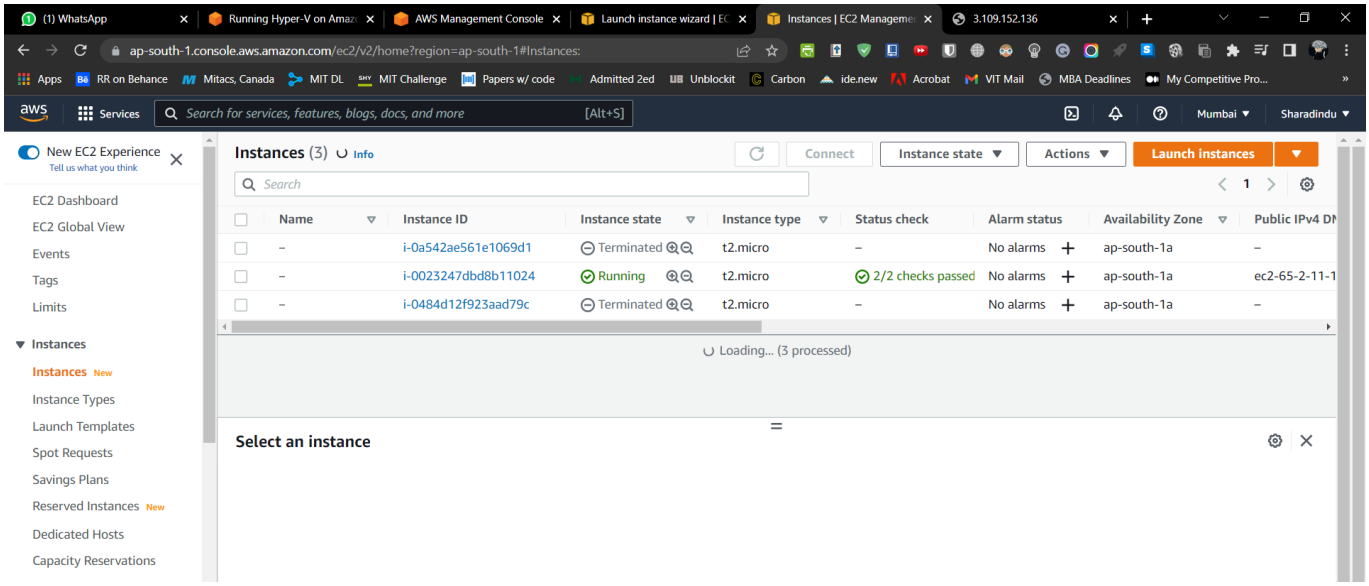
- To connect to a remote host using terminal (On Linux): ssh [username]@[server_ip_or_hostname]
- For Windows-based systems use the tool called 'PuTTY'.
- PuTTY is an SSH client with a GUI for SSH and Telnet.

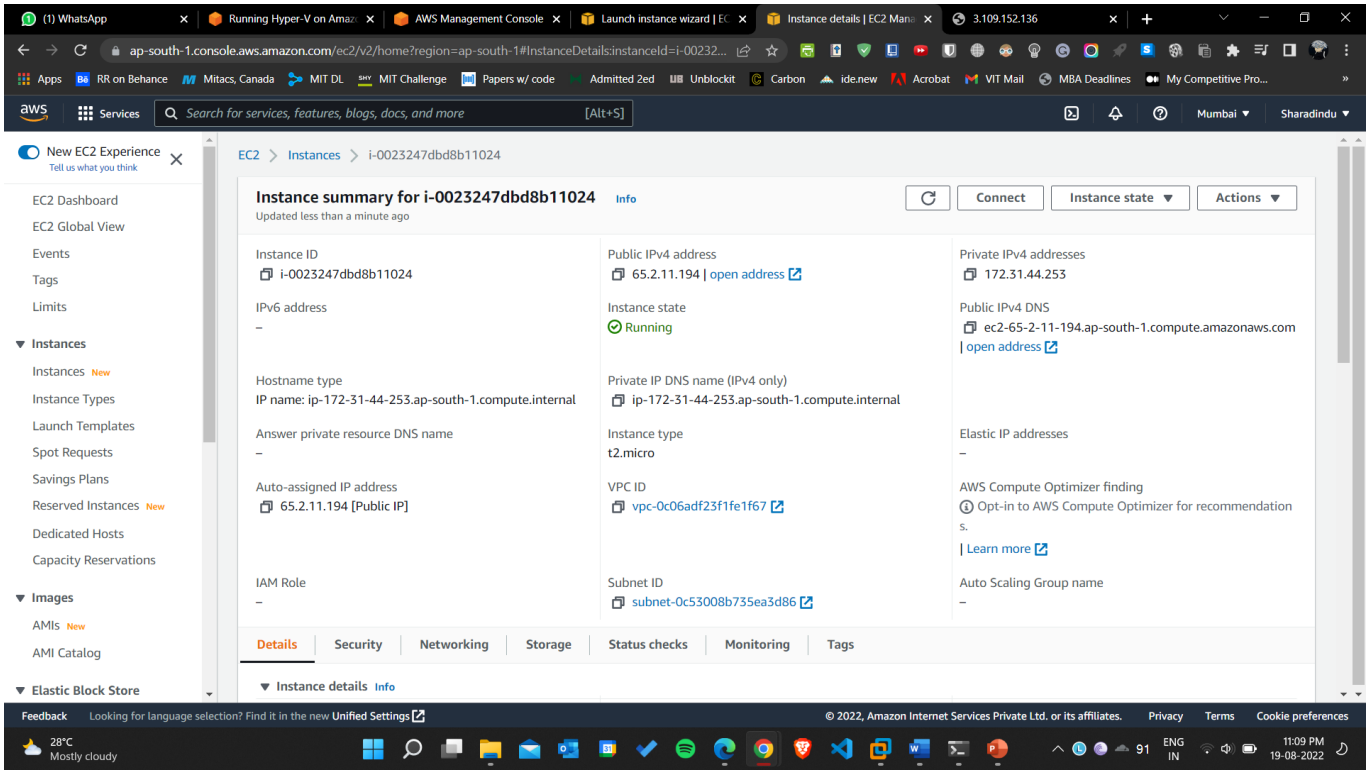
Requirements for conducting the experiment:

- VM instances running on local machines and cloud
- PuTTY for Windows-based system
- Terminal for Linux
- SSH clients and server

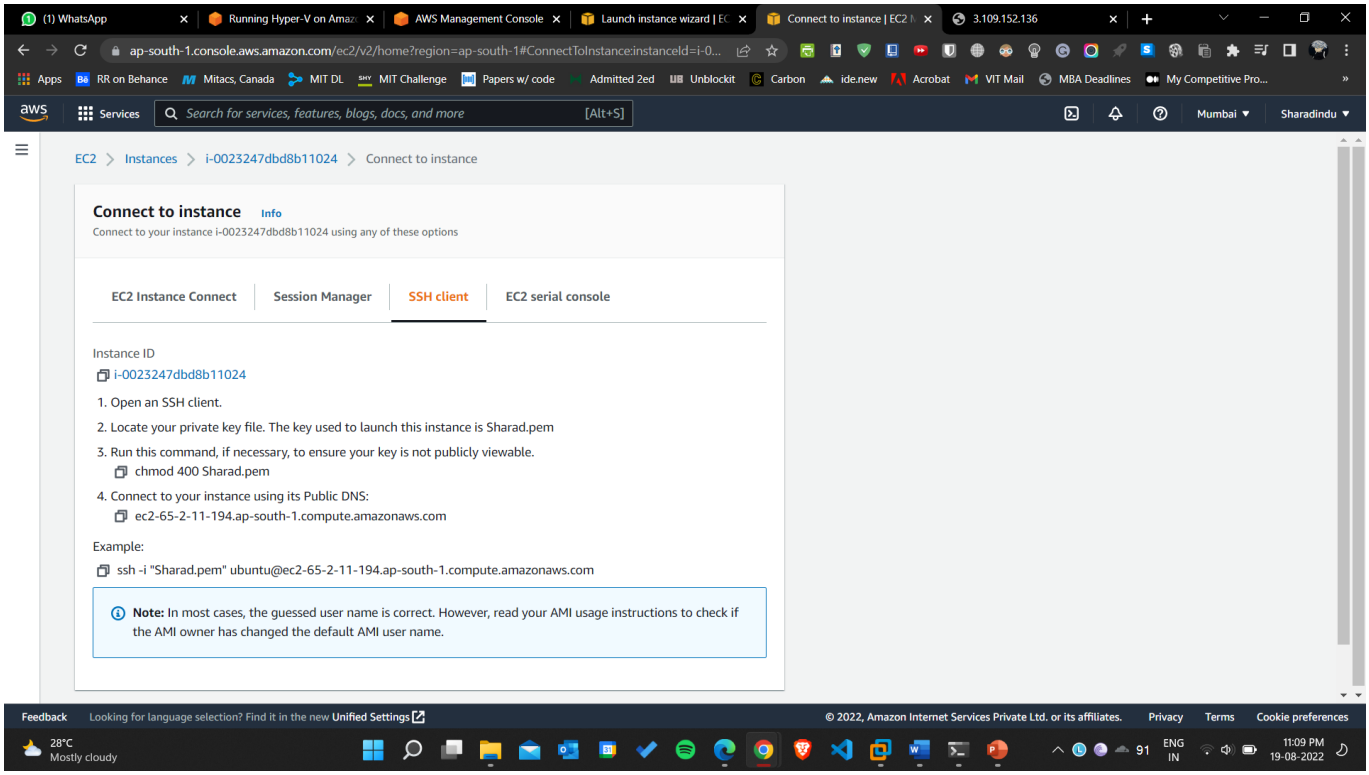
Procedure & Screenshots:

Step 1: Create a new instance in AWS, after terminating the existing ones.



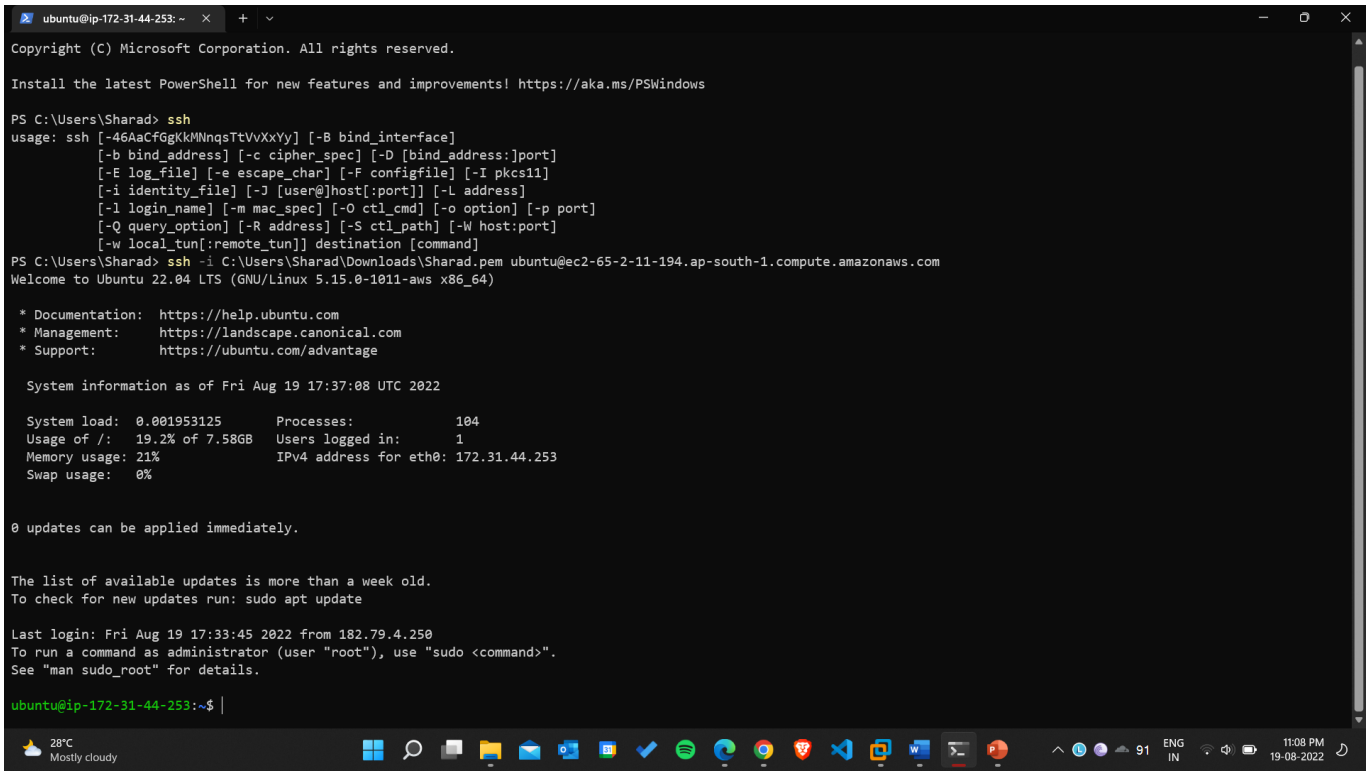


Step 2: Copy the Public DNS from the following page.



Step 3: Open PowerShell. Type in ‘SSH’, to see if SSH is already configured on the laptop or not. In my system it is. So no additional steps are necessary at this point.

Step 4: Use the .pem file downloaded earlier (in experiment 1), and the public dns copied to type in the following command: “ssh -i C:\Users\Sharad\Downloads\Sharad.pem ubuntu@ec2-65-2-11-194.ap-south-1.compute.amazonaws.com”



Conclusion: I’ve successfully created an SSH tunnel between our server in the local machine and remote clients in an EC2 instance of AWS.