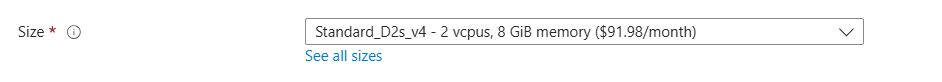
**Assignment 7 - Virtualization**

**Machine specs on Azure:**

1. **For use of virtualization in Linux its better to use a VM with higher performance.**
2. **So we create a VM with high performance,**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/1.png)**

**Checking virtualization support on machine:**

1. **Once the VM is created we can check the virtualization support by cpuinfo, cat /proc/cpuinfo**

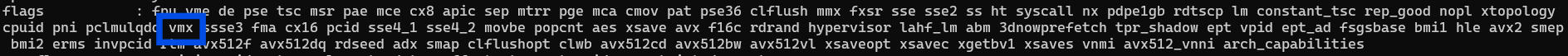
**[A screenshot of a computer

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/2.png)**

1. **Am I using 64 bit CPU/system [x86\_64/AMD64/Intel64]? IN flags check for lm**

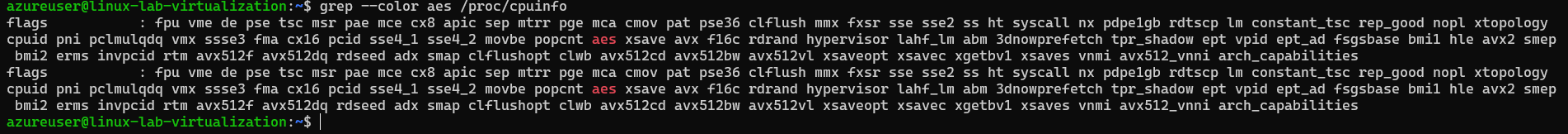
**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/3.png)**

1. **Do I have hardware virtualization support?**
   * **vmx – Intel VT-x, virtualization support enabled in BIOS.**
   * **svm – AMD SVM, virtualization enabled in BIOS.**
   * **Since we are using intel based system we only have VMX**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/4.png)**

**4.Do I have hardware AES/AES-NI advanced encryption support?**

**aes – Applications performing encryption and decryption using the Advanced Encryption Standard on Intel and AMD cpus.**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/5.png)**

**# Install kvm-ok on a Debian/Ubuntu**

1. **sudo apt install cpu-checker**

**[A screen shot of a computer program

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/6.png)**

1. **sudo kvm-ok**

**[A black screen with green and blue text

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/7.png)**

**Linux Virtualization Exercise**

**All exercises should be attempted on new Ubuntu 24.04 Linux installation, that supports nested virtualization.**

**Part 1: Introduction to virtualization concepts (30 minutes)**

**Research: Using sources (Sources linked at the end of the task), learn about the following concepts:**

**Virtualization**

**​Virtualization is a technology that allows multiple operating systems or applications to run concurrently on a single physical machine by abstracting the underlying hardware resources. This is achieved through the use of hypervisors, which are software layers that create and manage virtual machines (VMs).​**

**Hypervisor**

**A hypervisor is software or firmware that creates and manages virtual machines (VMs) on a physical host. It allows multiple VMs to run on a single physical machine by abstracting and allocating hardware resources like CPU, memory, and storage.**

**Virtual machines (VM)**

**A virtual machine (VM) is a software-based emulation of a physical computer that runs an operating system (OS) and applications just like a real machine. VMs are created and managed by a hypervisor, which allocates CPU, memory, storage, and other resources from the physical hardware to the VM.**

**Containers**

**A container is a lightweight, portable software unit that packages an application and its dependencies together so it can run consistently across different environments. Unlike virtual machines (VMs), containers share the host operating system’s kernel, making them more efficient and faster.**

**The main differences between VMs and tanks**

**Summary: Summary in a text file briefly summarize the core differences between VMs and containers. Focus on their architecture, resource utilization, and insulation levels.**

| **Feature Virtual** | **Machines (VMs)** | **Containers** |
| --- | --- | --- |
| **Architecture** | **Each VM has a full OS, including a guest OS and virtualized hardware.** | **Containers share the host OS kernel and run as isolated processes.** |
| **Resource Usage** | **More resource-intensive due to separate OS instances.** | **Lightweight, as they share the OS kernel.** |
| **Startup Time** | **Slower, since the OS needs to boot.** | **Faster, since they don’t require OS booting.** |
| **Isolation** | **Stronger isolation (separate OS for each VM).** | **Weaker isolation (process-level separation).** |
| **Portability** | **Less portable, depends on the hypervisor.** | **Highly portable across different environments.** |
| **Scalability** | **Requires more resources to scale.** | **Easily scalable due to lightweight nature.** |
| **Use Case** | **Best for running multiple OS environments or legacy applications.** | **Best for microservices, cloud applications, and fast deployments.** |

**Part 2: Working with Multipass (1-2 hours)**

**Multipass is a command-line tool that allows you to quickly create and manage Ubuntu virtual machines.**

**Installation: Follow the instructions on the Multipass website to install Multipass on your system. More information can be found in the source.**

1. **Start VM and install Multipass.**

**sudo snap install multipass**

1. **Verify the installation**

**multipass version**

**[A screen shot of a computer code

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/1.png)**

**Basic commands: Use the multipass command-line interface and practice the following:**

1. **multipass boot: Launch the default Ubuntu instance.**

**multipass launch --name my-instance**

**multipass launch --name multi-VM (Name of the instance is multi-VM)**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/2.png)**

1. **multipass list: Lists all running instances.**

**multipass list**

**[A black screen with white text and green and blue letters

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/3.png)**

1. **multipass info: View details about a specific instance.**

**[A screenshot of a computer

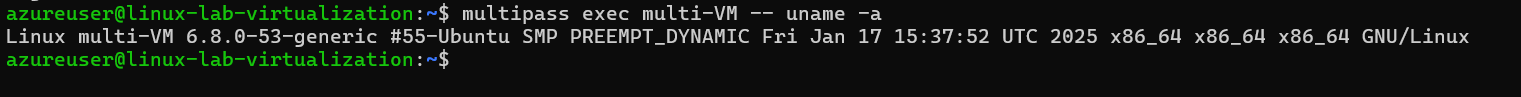
AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/4.png)**

1. **multipass shell: Access to the shell of a running instance.**

**[A computer screen with white text

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/5.png)**

1. **multipass exec: Run the command on the instance.**

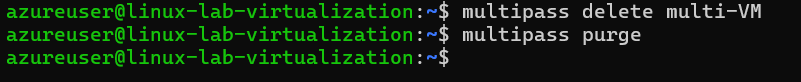
**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/6.png)**

1. **multipass stop: Stop the running instance.**

**[A screenshot of a computer program

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/7.png)**

1. **multipass delete: Delete the instance.**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/8.png)**

**Cloud-init:Study: Learn about cloud-initi and how it can be used to configure virtual machines**

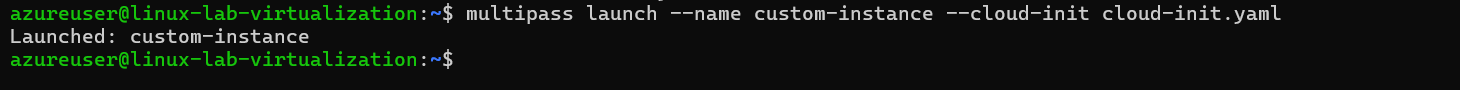
**Experiment: Create a cloud-init configuration file to customize the installation of a new instance.**

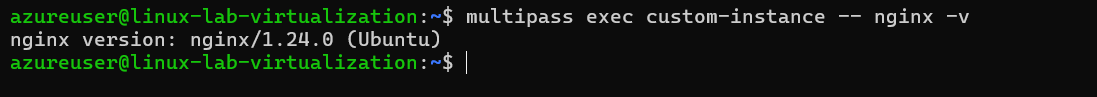
* **For example, you can install specific packages or assign users.**
* **Start a new instance of Multipass using this configuration.**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/9.png)**

**[A screenshot of a computer program

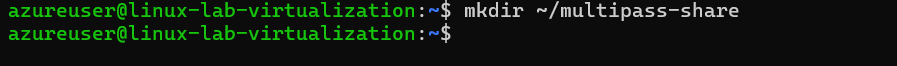
AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/10.png)**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/11.png)**

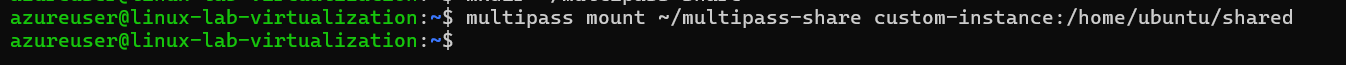
**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/12.png)**

**File sharing: Find out: Explore how to share files and folders between your host computer and Multipass instances**

1. **Create a shared folder on your Linux VM**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/13.png)**

1. **Mount the folder inside the Multipass instance**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/14.png)**

1. **Check the shared folder inside the instance**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/15.png)**

**Policy: Create a shared folder and access it from both your host and your Multipass instance.**

1. **Create sample.txt file inside VM and check availability inside custom-instance VM,**

**[A screenshot of a computer screen

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/16.png)**

**Part 3: Exploring LXD (1 hour)**

**LXD is another container and virtual machine management built on LXC, the so-called "Virtual Machine Management" of the Linux container. Runtime.**

**Study: Read about LXD and its features on LinuxContainers website Setup: Install LXD on your system, enable LXD Basic commands: Experiment with basic LXD commands to create, manage, and interact with containers. You can find the starting point in the source**

**Installing and managing LXD / LXC system**

1. **update your system by using apt: apt update && apt upgrade -y**

**[A screen shot of a computer

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/8.png) [A screenshot of a computer program

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/9.png) [A screen shot of a computer

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/10.png)**

1. **check if you just updated your kernel or other systems needing a complete system reboot, and if so, reboot: sudo reboot**

**[A screen shot of a computer

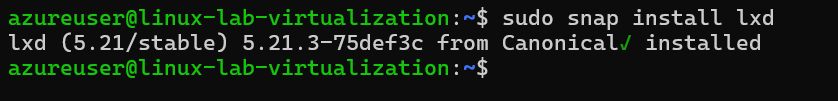
AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/11.png)**

1. **Install Snap: sudo apt install snap -y**

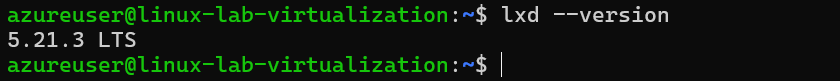
**[A computer screen with white text

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/12.png)**

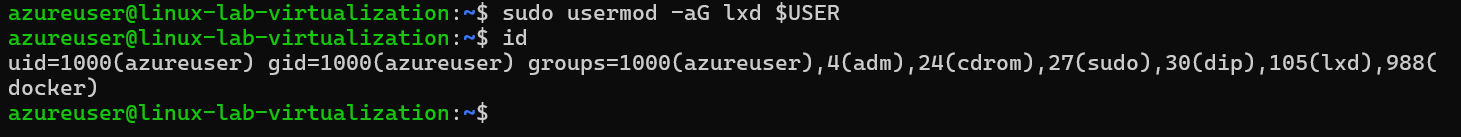
1. **install lxd using snap: sudo snap install lxd**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/13.png)**

1. **check lxd version and installation: lxd --version**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/14.png)**

1. **check that your user belongs to LXD group: id, and look for LXD. If you do not find lxd group, add user to it: sudo usermod -aG lxd $USER**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/15.png)**

1. **check lxc system for listing of machines and containers: lxc list**

**[A screen shot of a computer

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/16.png)**

1. **initialize xld, to configure system to your environment: lxd init. Make sure you run this as basic user, without root / sudo privileges it will ask series of questions, so based on your requirement answer them.**
   * **When it asks about clustering, choose ‘no’ (unless you’re setting up a cluster)**
   * **For storage, I recommend saying ‘yes’ to a new storage pool**
   * **The ‘dir’ backend is fine for beginners**
   * **Say ‘yes’ to a network bridge**
   * **Choose ‘no’ to make LXD sever available over the network.**

**[A computer screen shot of white text

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/17.png)**

1. **Once lxd is initialized successfully, we can verify the information using following set of commands:**

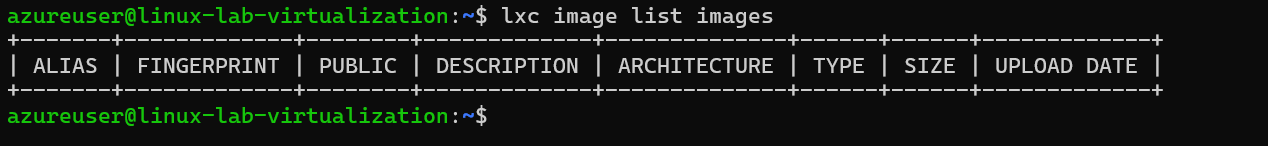
**$lxc profile list $lxc network list $lxc storage list**

**[A screenshot of a computer screen

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/18.png)**

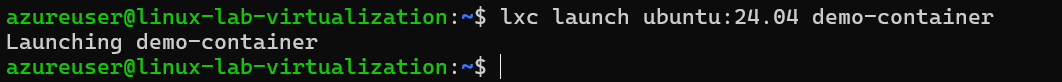
1. **In order to list all available images, run:**

**$lxc image list images:**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/19.png)**

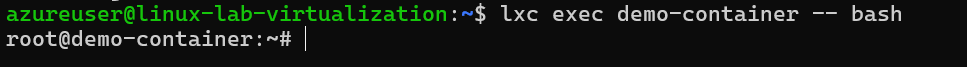
1. **Create your first container:**

**$lxc launch ubuntu:24.04 demo-container**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/20.png)**

1. **Access the console of container. Run:**

**$ lxc exec demo-container -- bash**

**[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/21.png)**

**Part 4: How to Stick Apps with Docker (1 hour)**

**Docker is a platform for building, deploying and managing container applications**

**Installation: Install Docker Desktop on your system if you want to familiarize yourself with this section, You can find installation information for Docker Engine in the sources.**

**Basic concepts: Research: Explore core Docker concepts such as images, containers, and Dockerfiles. Source.**

**Experiment: Follow the "Docker Workshop" to get hands-on experience with Docker.**

**Installing and managing Docker engine based system**

**Basic steps:**

**Follow good instructions from Docker, at**[**https://docs.docker.com/**](https://docs.docker.com/)

**[A screenshot of a computer

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/22.png)**

**More specifically from Ubuntu engine install instructions for Ubuntu system, and using convenience script:**

[**https://docs.docker.com/engine/install/ubuntu/#install-using-the-convenience-script**](https://docs.docker.com/engine/install/ubuntu/#install-using-the-convenience-script)

**curl -fsSL**[**https://get.docker.com**](https://get.docker.com/)**-o get-docker.sh**

**sudo sh get-docker.sh**

**Executing docker install script, commit: 7cae5f8b0decc17d6571f9f52eb840fbc13b2737**

**<...>**

**[A screen shot of a computer program

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/23.png)**

**post installation guide tells us to make user part of Docker group on Linux machine:**

**sudo groupadd docker**

**sudo usermod -aG docker $USER**

**newgrp docker**

**[A computer screen with white text

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/24.png)**

**check docker version**

**[A screenshot of a computer

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/25.png)**

**Run Nginx on docker:**

1. **Get the latest Nginx**

**[A screen shot of a computer

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/26.png)**

1. **Start docker nginx image**
   * **docker run -p 80:80 nginx**
   * **docker run -d -p 80:80 nginx (To run on background use -d tag)**

**[A screen shot of a computer program

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/27.png)**

1. **If we open the IP of our virtual machine we can see Nginx is running,**

**[A screenshot of a computer

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/28.png)**

**Part 5: Snaps for Self-Contained Applications (30 minutes)**

**Snaps provides a way to package applications with their dependencies for consistent execution on Linux distributions.**

* **Research: Using sources, explore Snapcraft and the concept of Snaps.**
* **Experiment: Choose a simple app and compress it into Snap using Snapcraft. For more information, you can find "Snapcraft" - from the source.**

**Snapcraft is a tool and ecosystem that helps developers package their applications into snaps, which are self-contained software packages that include the application and all of its dependencies. This makes it easy to distribute and run applications across different Linux distributions without worrying about compatibility issues or missing dependencies. Snaps are designed to work on most modern Linux distributions, and they are confined, meaning they run in isolated environments to ensure security.**

1. **Install snapcraft**

**sudo apt update sudo apt install snapcraft**

**[A screen shot of a computer

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/1.png)**

1. **Create the python file**

**[A screenshot of a computer

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/2.png)**

1. **Create the Snapcraft Configuration**

**[A screen shot of a computer

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/3.png)**

1. **Fix the error by adding base: core20**
2. **Create the snapcraft**

**[A screenshot of a computer

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/4.png)**

1. **Remove python-version to fix the error,**

**[A black screen with white text

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/5.png)**

**[A screen shot of a computer

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/6.png)**

1. **Check for the file hello-world and create the missing file. Re run the snap**

**[A screen shot of a computer program

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/7.png)**

**Method 2**

1. **Build snap ENV.**

**sudo snap install lxd**

**[A screen shot of a computer screen

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/8.png)**

1. **Create YAML template**

**[A screenshot of a computer program

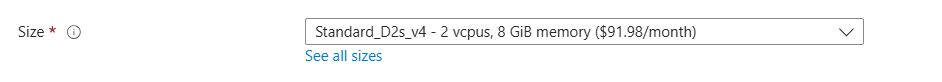
AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/9.png)**

1. **Build template**

**Assignment 7 - Virtualization**

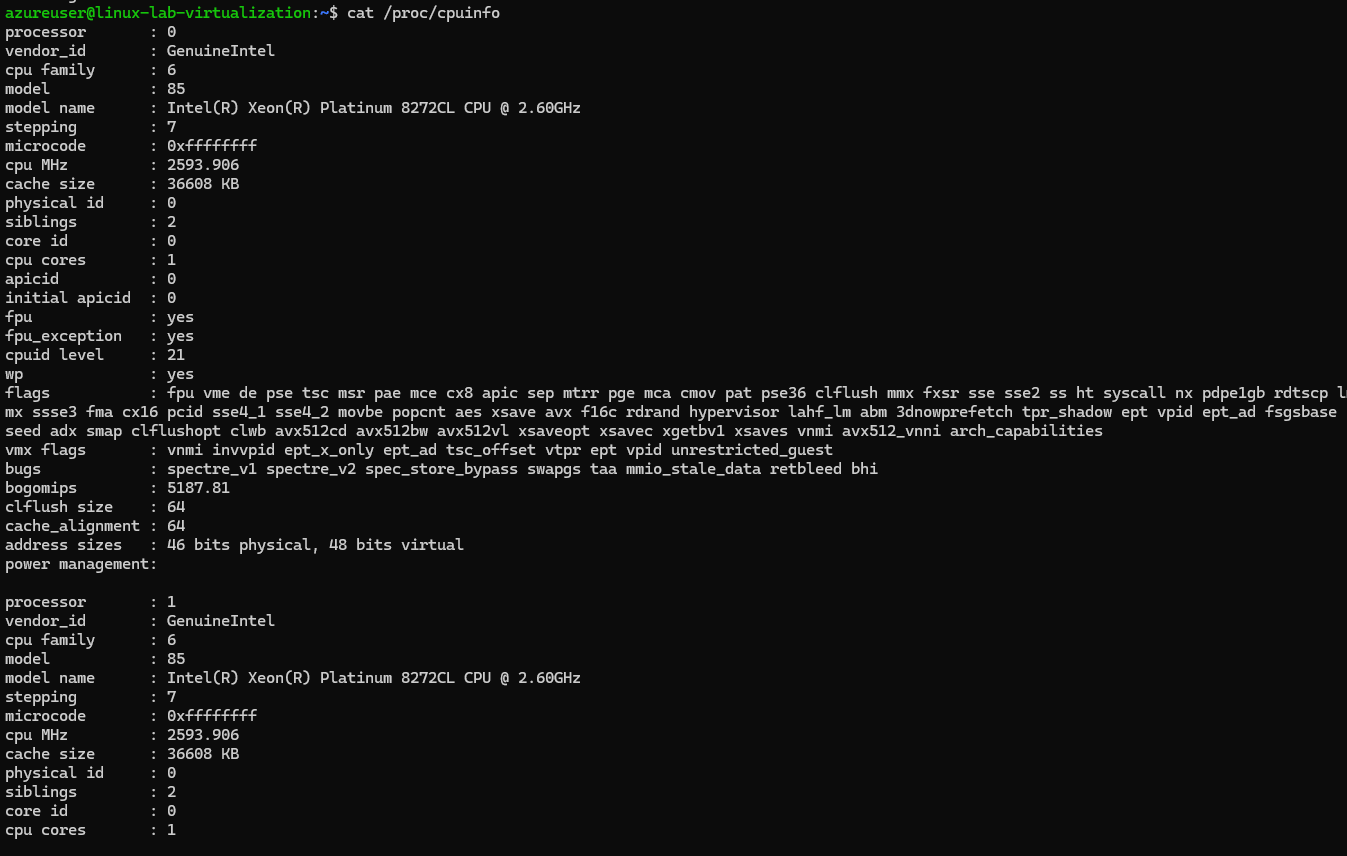
**Machine specs on Azure:**

1. For use of virtualization in Linux its better to use a VM with higher performance.
2. So we create a VM with high performance,

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/1.png)

**Checking virtualization support on machine:**

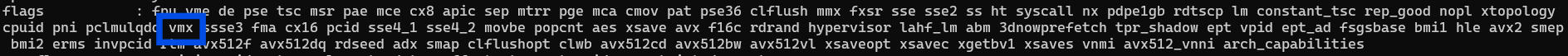
1. Once the VM is created we can check the virtualization support by cpuinfo, **cat /proc/cpuinfo**

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/2.png)

1. Am I using 64 bit CPU/system [x86\_64/AMD64/Intel64]? IN flags check for **lm**

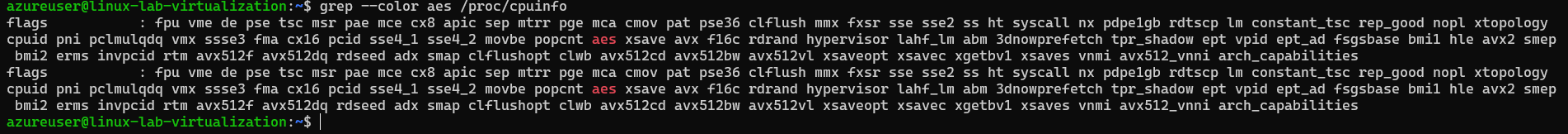
[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/3.png)

1. Do I have hardware virtualization support?
   * vmx – Intel VT-x, virtualization support enabled in BIOS.
   * svm – AMD SVM, virtualization enabled in BIOS.
   * Since we are using intel based system we only have VMX

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/4.png)

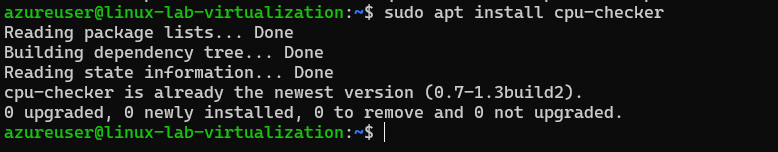
4.Do I have hardware AES/AES-NI advanced encryption support?

aes – Applications performing encryption and decryption using the Advanced Encryption Standard on Intel and AMD cpus.

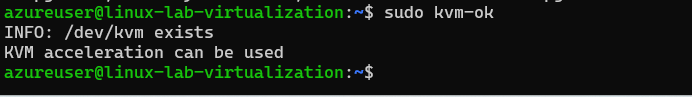
[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/5.png)

**# Install kvm-ok on a Debian/Ubuntu**

1. sudo apt install cpu-checker

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/6.png)

1. sudo kvm-ok

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/7.png)

**Linux Virtualization Exercise**

All exercises should be attempted on new Ubuntu 24.04 Linux installation, that supports nested virtualization.

**Part 1: Introduction to virtualization concepts (30 minutes)**

Research: Using sources (Sources linked at the end of the task), learn about the following concepts:

**Virtualization**

​Virtualization is a technology that allows multiple operating systems or applications to run concurrently on a single physical machine by abstracting the underlying hardware resources. This is achieved through the use of hypervisors, which are software layers that create and manage virtual machines (VMs).​

**Hypervisor**

A hypervisor is software or firmware that creates and manages virtual machines (VMs) on a physical host. It allows multiple VMs to run on a single physical machine by abstracting and allocating hardware resources like CPU, memory, and storage.

**Virtual machines (VM)**

A virtual machine (VM) is a software-based emulation of a physical computer that runs an operating system (OS) and applications just like a real machine. VMs are created and managed by a hypervisor, which allocates CPU, memory, storage, and other resources from the physical hardware to the VM.

**Containers**

A container is a lightweight, portable software unit that packages an application and its dependencies together so it can run consistently across different environments. Unlike virtual machines (VMs), containers share the host operating system’s kernel, making them more efficient and faster.

**The main differences between VMs and tanks**

**Summary: Summary in a text file briefly summarize the core differences between VMs and containers. Focus on their architecture, resource utilization, and insulation levels.**

| **Feature Virtual** | **Machines (VMs)** | **Containers** |
| --- | --- | --- |
| Architecture | Each VM has a full OS, including a guest OS and virtualized hardware. | Containers share the host OS kernel and run as isolated processes. |
| Resource Usage | More resource-intensive due to separate OS instances. | Lightweight, as they share the OS kernel. |
| Startup Time | Slower, since the OS needs to boot. | Faster, since they don’t require OS booting. |
| Isolation | Stronger isolation (separate OS for each VM). | Weaker isolation (process-level separation). |
| Portability | Less portable, depends on the hypervisor. | Highly portable across different environments. |
| Scalability | Requires more resources to scale. | Easily scalable due to lightweight nature. |
| Use Case | Best for running multiple OS environments or legacy applications. | Best for microservices, cloud applications, and fast deployments. |

**Part 2: Working with Multipass (1-2 hours)**

Multipass is a command-line tool that allows you to quickly create and manage Ubuntu virtual machines.

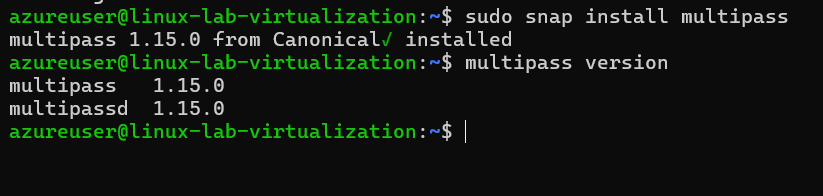
**Installation: Follow the instructions on the Multipass website to install Multipass on your system. More information can be found in the source.**

1. Start VM and install Multipass.

sudo snap install multipass

1. Verify the installation

multipass version

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/1.png)

**Basic commands: Use the multipass command-line interface and practice the following:**

1. **multipass boot: Launch the default Ubuntu instance.**

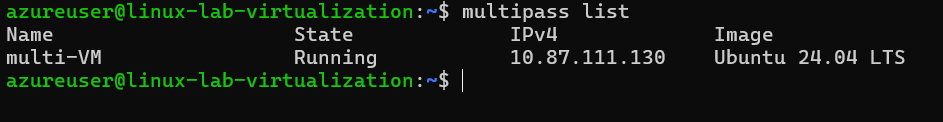
multipass launch --name my-instance

multipass launch --name multi-VM (Name of the instance is multi-VM)

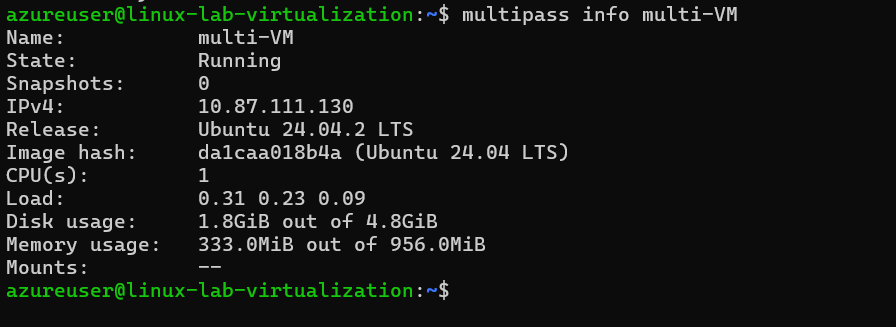
[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/2.png)

1. **multipass list: Lists all running instances.**

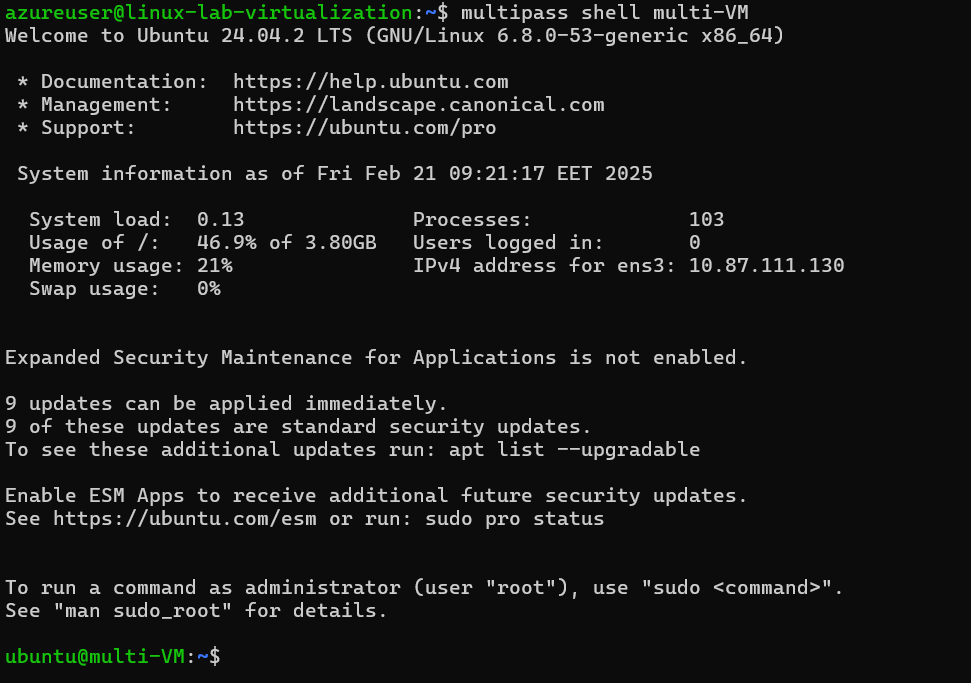
multipass list

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/3.png)

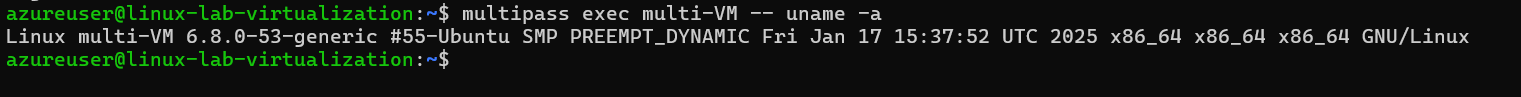
1. **multipass info: View details about a specific instance.**

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/4.png)

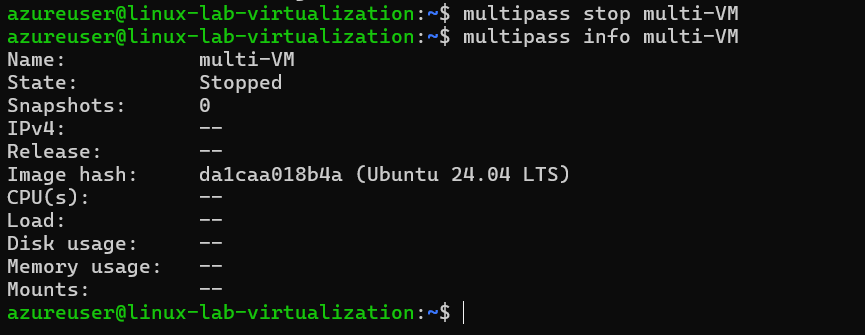
1. **multipass shell: Access to the shell of a running instance.**

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/5.png)

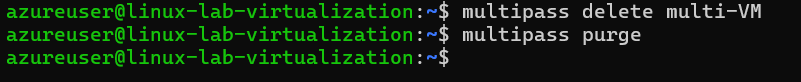
1. **multipass exec: Run the command on the instance.**

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/6.png)

1. **multipass stop: Stop the running instance.**

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/7.png)

1. **multipass delete: Delete the instance.**

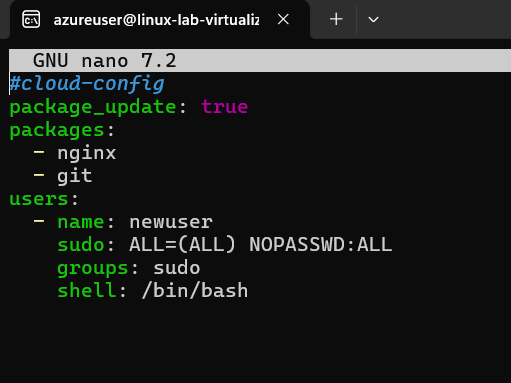
[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/8.png)

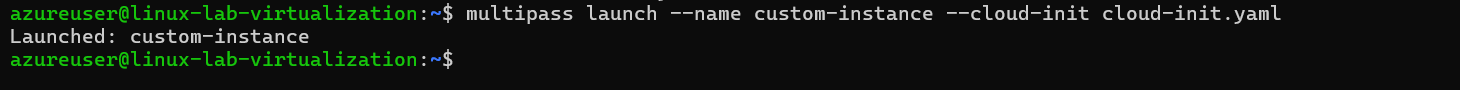
**Cloud-init:Study: Learn about cloud-initi and how it can be used to configure virtual machines**

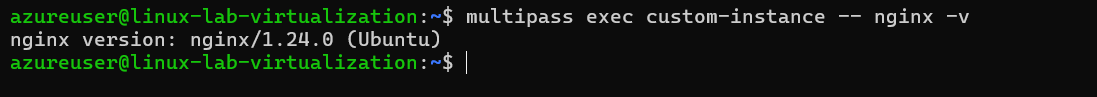
**Experiment: Create a cloud-init configuration file to customize the installation of a new instance.**

* For example, you can install specific packages or assign users.
* Start a new instance of Multipass using this configuration.

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/9.png)

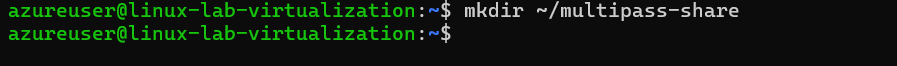
[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/10.png)

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/11.png)

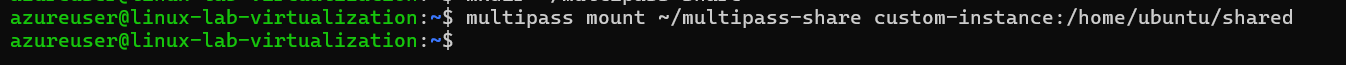
[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/12.png)

**File sharing: Find out: Explore how to share files and folders between your host computer and Multipass instances**

1. Create a shared folder on your Linux VM

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/13.png)

1. Mount the folder inside the Multipass instance

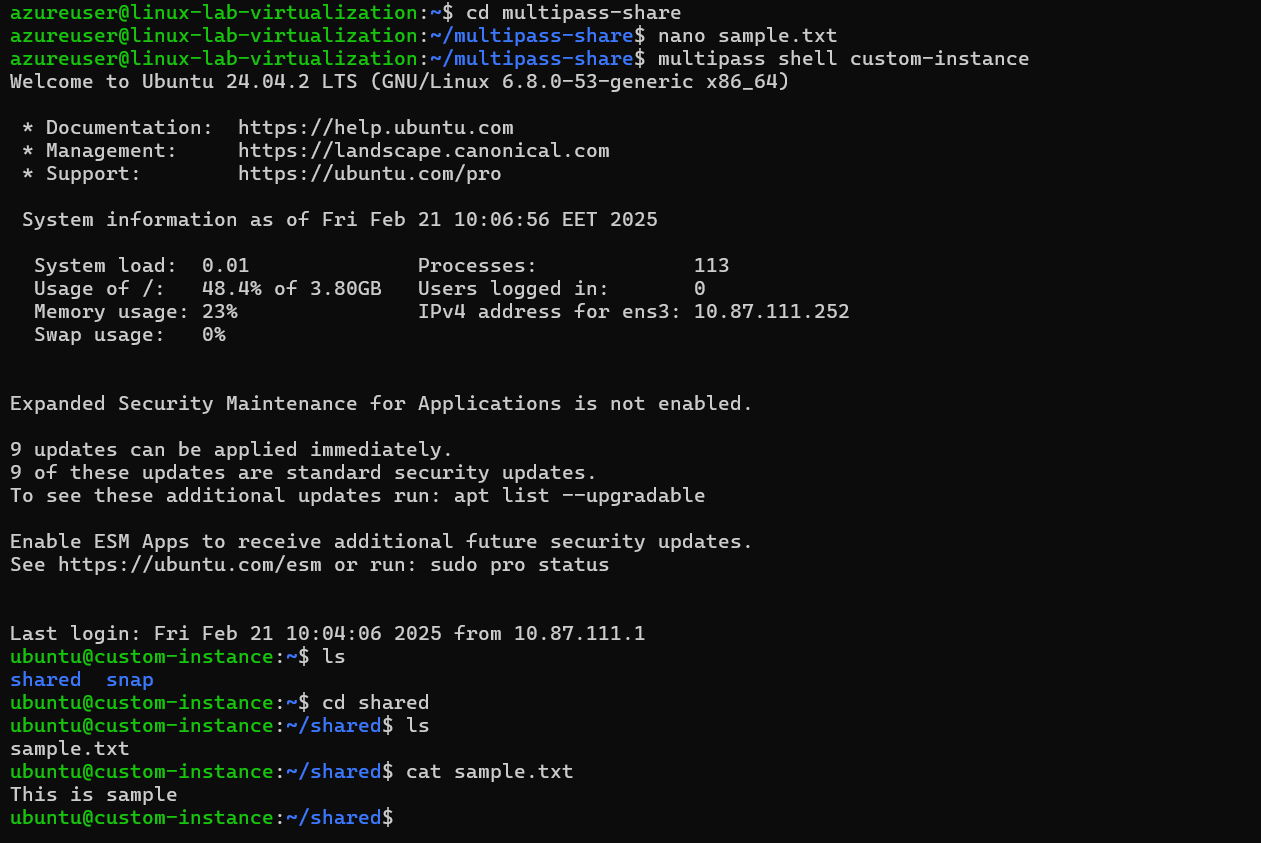
[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/14.png)

1. Check the shared folder inside the instance

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/15.png)

**Policy: Create a shared folder and access it from both your host and your Multipass instance.**

1. Create sample.txt file inside VM and check availability inside custom-instance VM,

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Multipass/16.png)

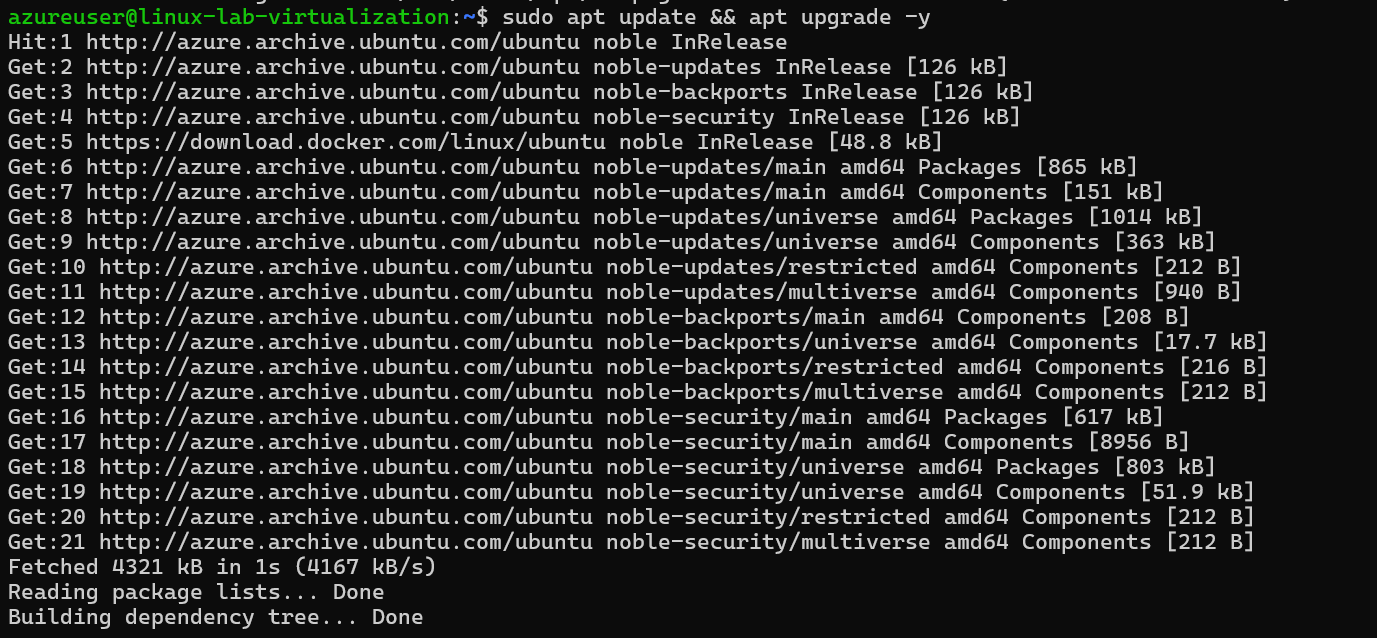
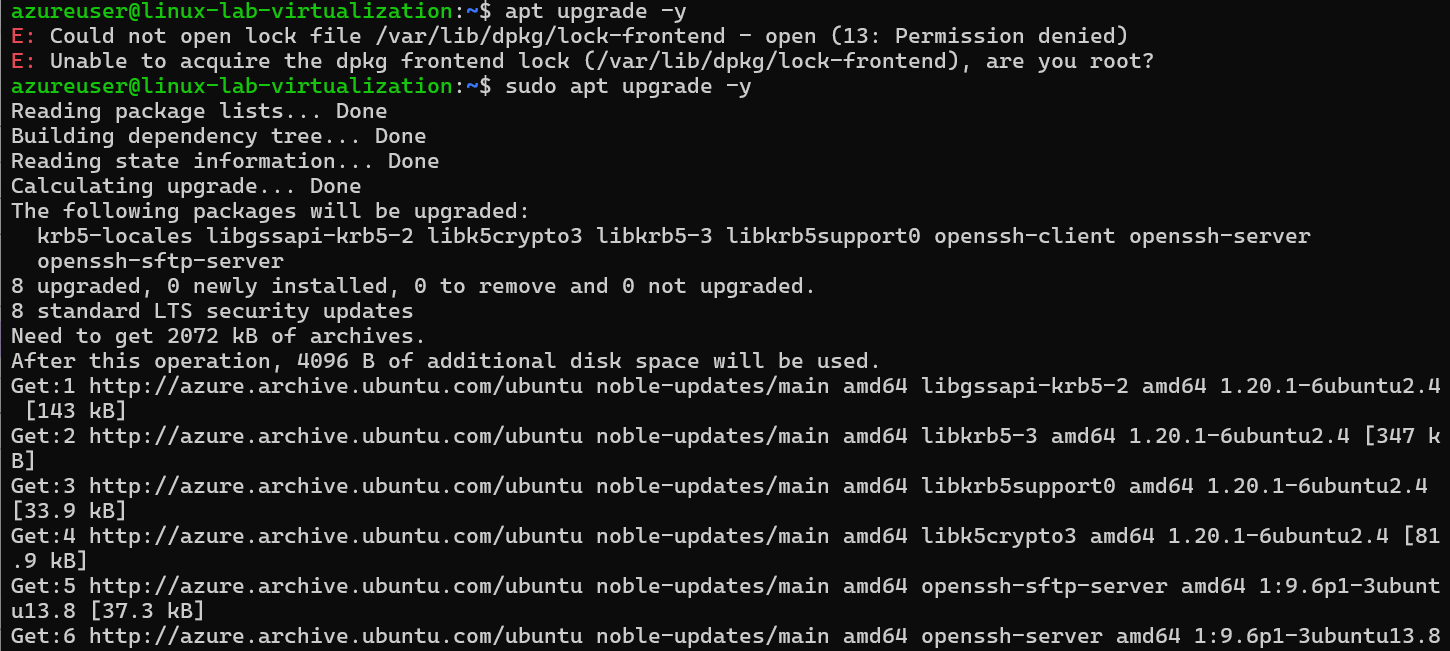
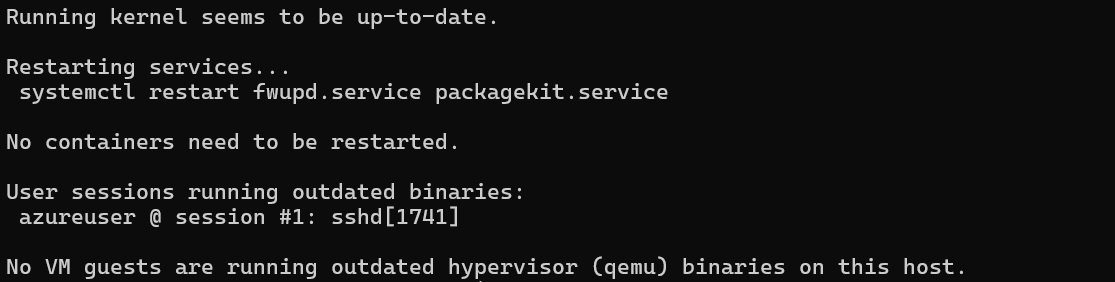
**Part 3: Exploring LXD (1 hour)**

LXD is another container and virtual machine management built on LXC, the so-called "Virtual Machine Management" of the Linux container. Runtime.

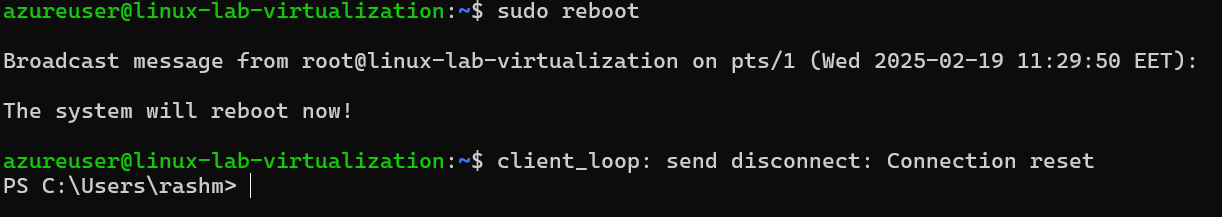
Study: Read about LXD and its features on LinuxContainers website Setup: Install LXD on your system, enable LXD Basic commands: Experiment with basic LXD commands to create, manage, and interact with containers. You can find the starting point in the source

**Installing and managing LXD / LXC system**

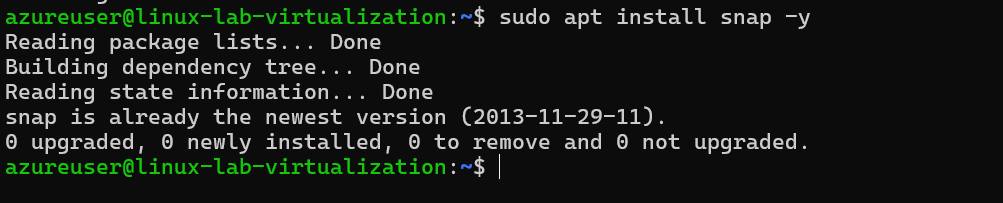
1. update your system by using apt: apt update && apt upgrade -y

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/8.png) [](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/9.png) [](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/10.png)

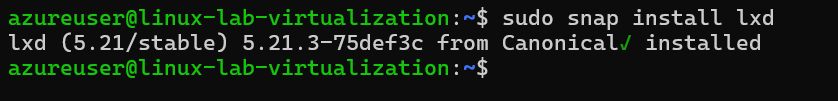
1. check if you just updated your kernel or other systems needing a complete system reboot, and if so, reboot: sudo reboot

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/11.png)

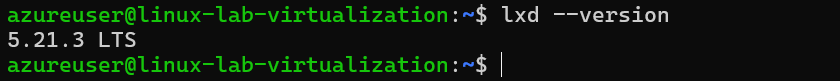
1. Install Snap: sudo apt install snap -y

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/12.png)

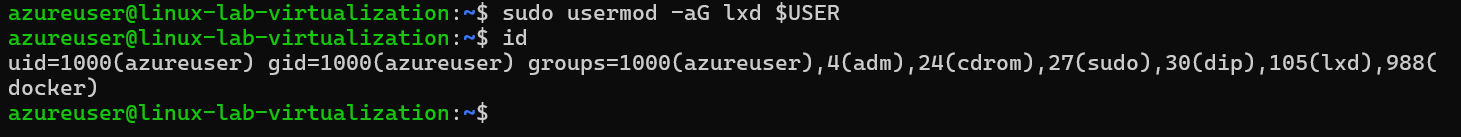
1. install lxd using snap: sudo snap install lxd

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/13.png)

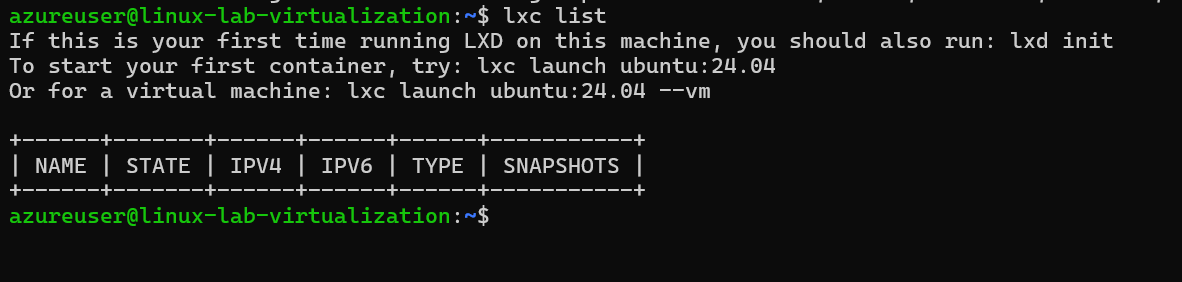
1. check lxd version and installation: lxd --version

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/14.png)

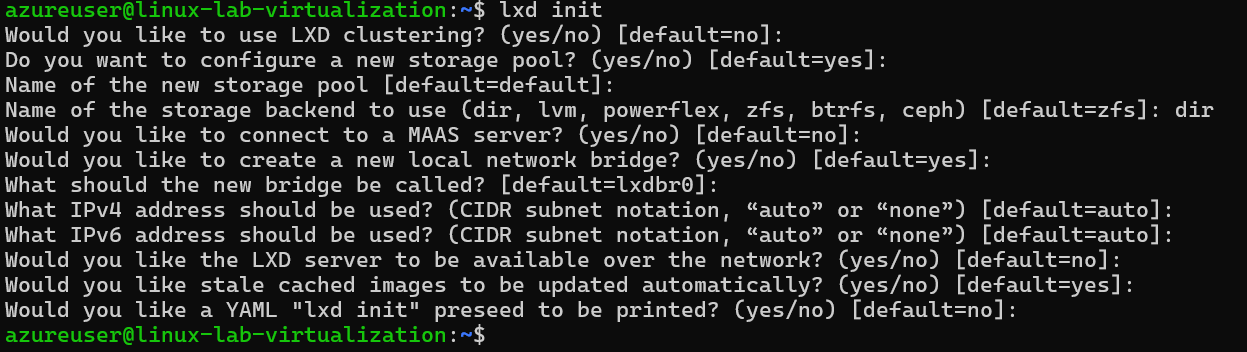
1. check that your user belongs to LXD group: id, and look for LXD. If you do not find lxd group, add user to it: sudo usermod -aG lxd $USER

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/15.png)

1. check lxc system for listing of machines and containers: lxc list

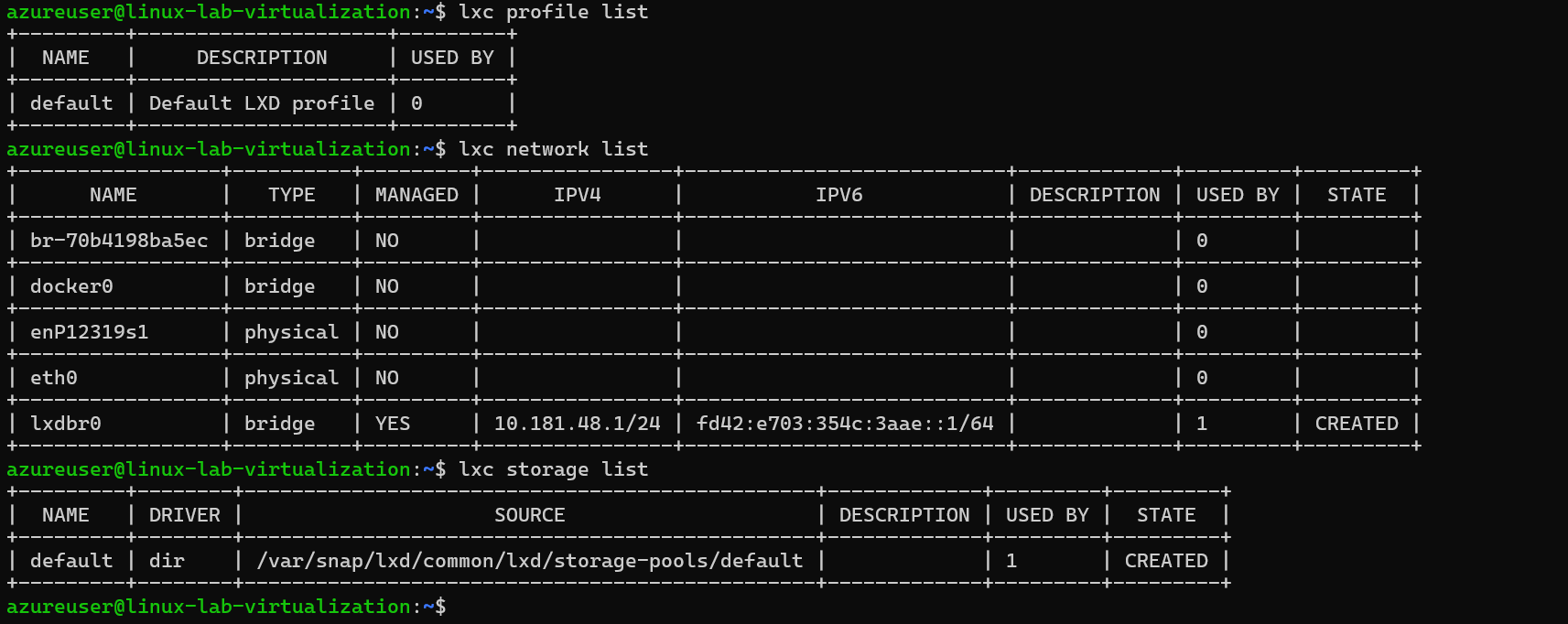
[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/16.png)

1. initialize xld, to configure system to your environment: lxd init. Make sure you run this as basic user, without root / sudo privileges it will ask series of questions, so based on your requirement answer them.
   * When it asks about clustering, choose ‘no’ (unless you’re setting up a cluster)
   * For storage, I recommend saying ‘yes’ to a new storage pool
   * The ‘dir’ backend is fine for beginners
   * Say ‘yes’ to a network bridge
   * Choose ‘no’ to make LXD sever available over the network.

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/17.png)

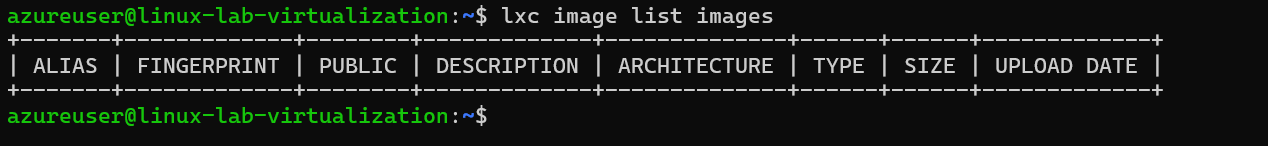
1. Once lxd is initialized successfully, we can verify the information using following set of commands:

$lxc profile list $lxc network list $lxc storage list

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/18.png)

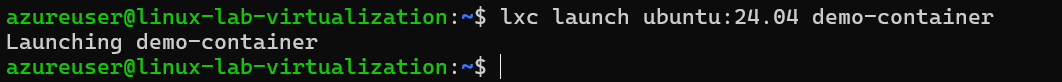
1. In order to list all available images, run:

$lxc image list images:

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/19.png)

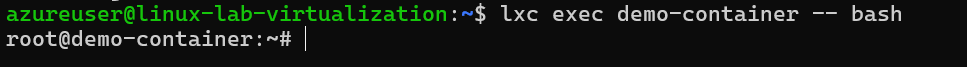
1. Create your first container:

$lxc launch ubuntu:24.04 demo-container

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/20.png)

1. Access the console of container. Run:

$ lxc exec demo-container -- bash

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/21.png)

**Part 4: How to Stick Apps with Docker (1 hour)**

Docker is a platform for building, deploying and managing container applications

Installation: Install Docker Desktop on your system if you want to familiarize yourself with this section, You can find installation information for Docker Engine in the sources.

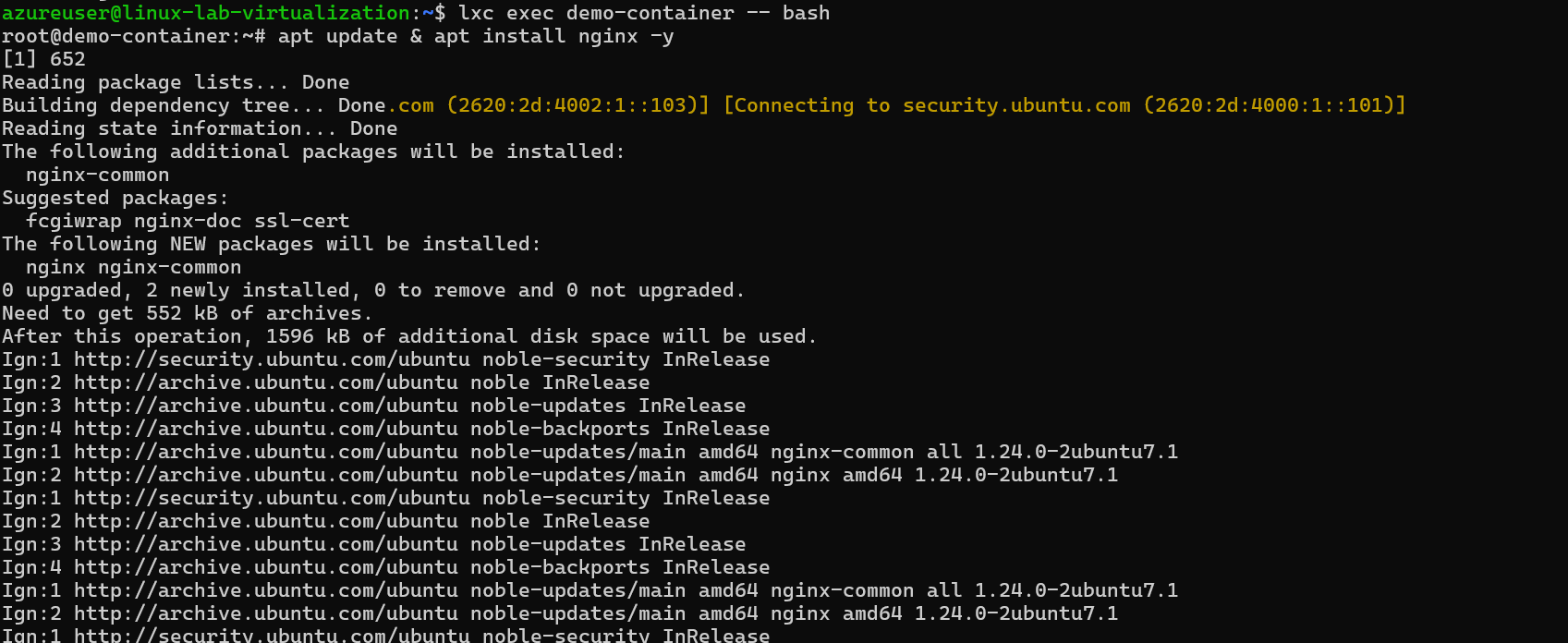
Basic concepts: Research: Explore core Docker concepts such as images, containers, and Dockerfiles. Source.

Experiment: Follow the "Docker Workshop" to get hands-on experience with Docker.

**Installing and managing Docker engine based system**

**Basic steps:**

Follow good instructions from Docker, at <https://docs.docker.com/>

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/22.png)

More specifically from Ubuntu engine install instructions for Ubuntu system, and using convenience script:

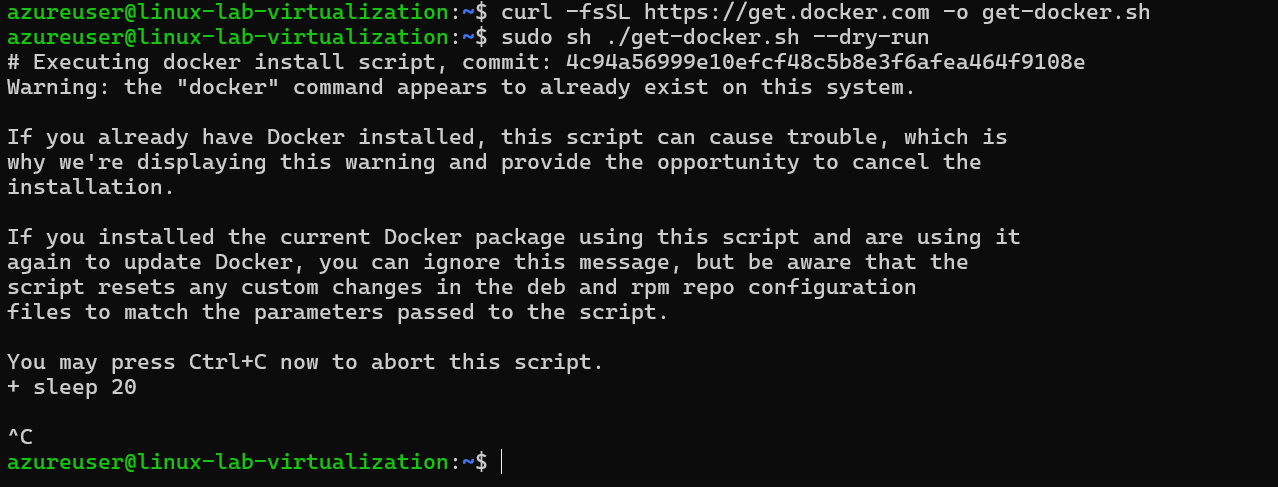
<https://docs.docker.com/engine/install/ubuntu/#install-using-the-convenience-script>

curl -fsSL [https://get.docker.com](https://get.docker.com/) -o get-docker.sh

sudo sh get-docker.sh

Executing docker install script, commit: 7cae5f8b0decc17d6571f9f52eb840fbc13b2737

<...>

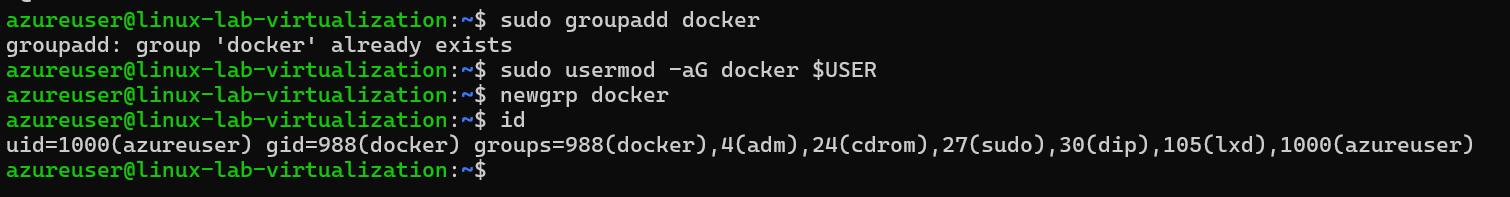
[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/23.png)

post installation guide tells us to make user part of Docker group on Linux machine:

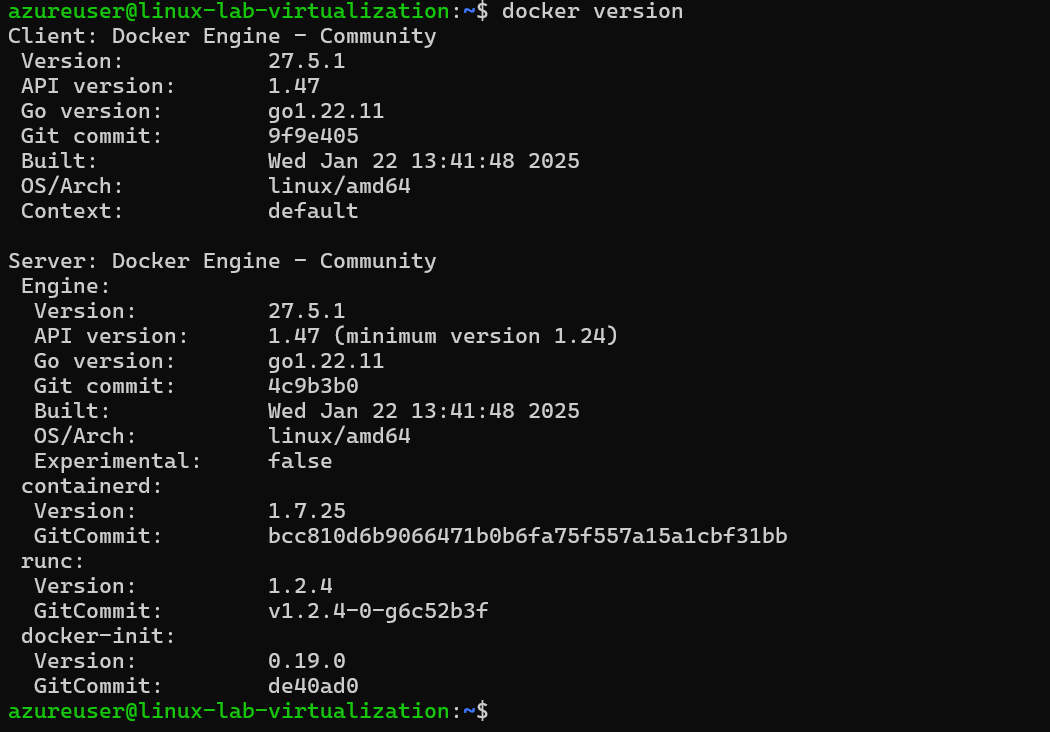
sudo groupadd docker

sudo usermod -aG docker $USER

newgrp docker

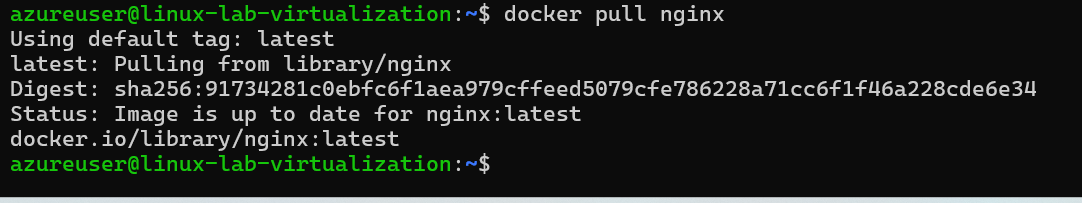
[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/24.png)

check docker version

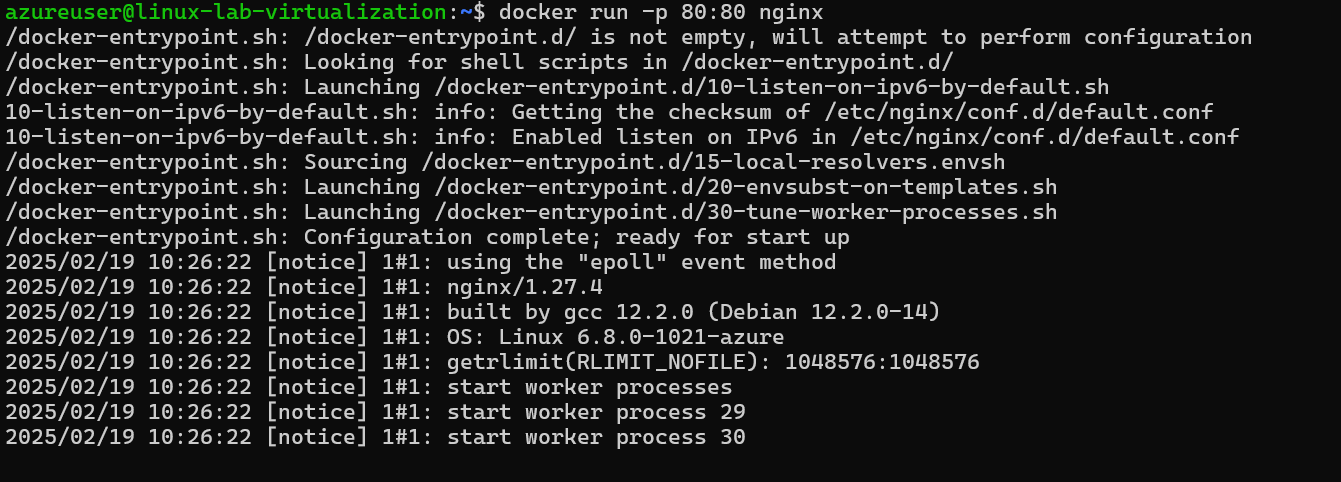
[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/25.png)

**Run Nginx on docker:**

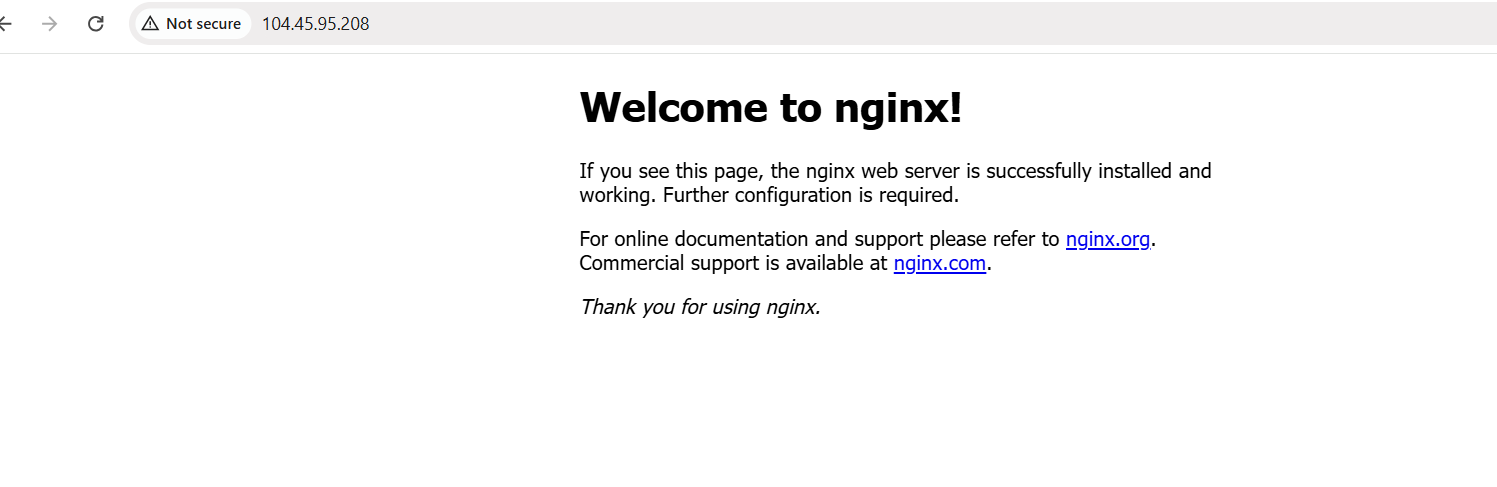
1. Get the latest Nginx

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/26.png)

1. Start docker nginx image
   * docker run -p 80:80 nginx
   * docker run -d -p 80:80 nginx (To run on background use -d tag)

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/27.png)

1. If we open the IP of our virtual machine we can see Nginx is running,

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/28.png)

**Part 5: Snaps for Self-Contained Applications (30 minutes)**

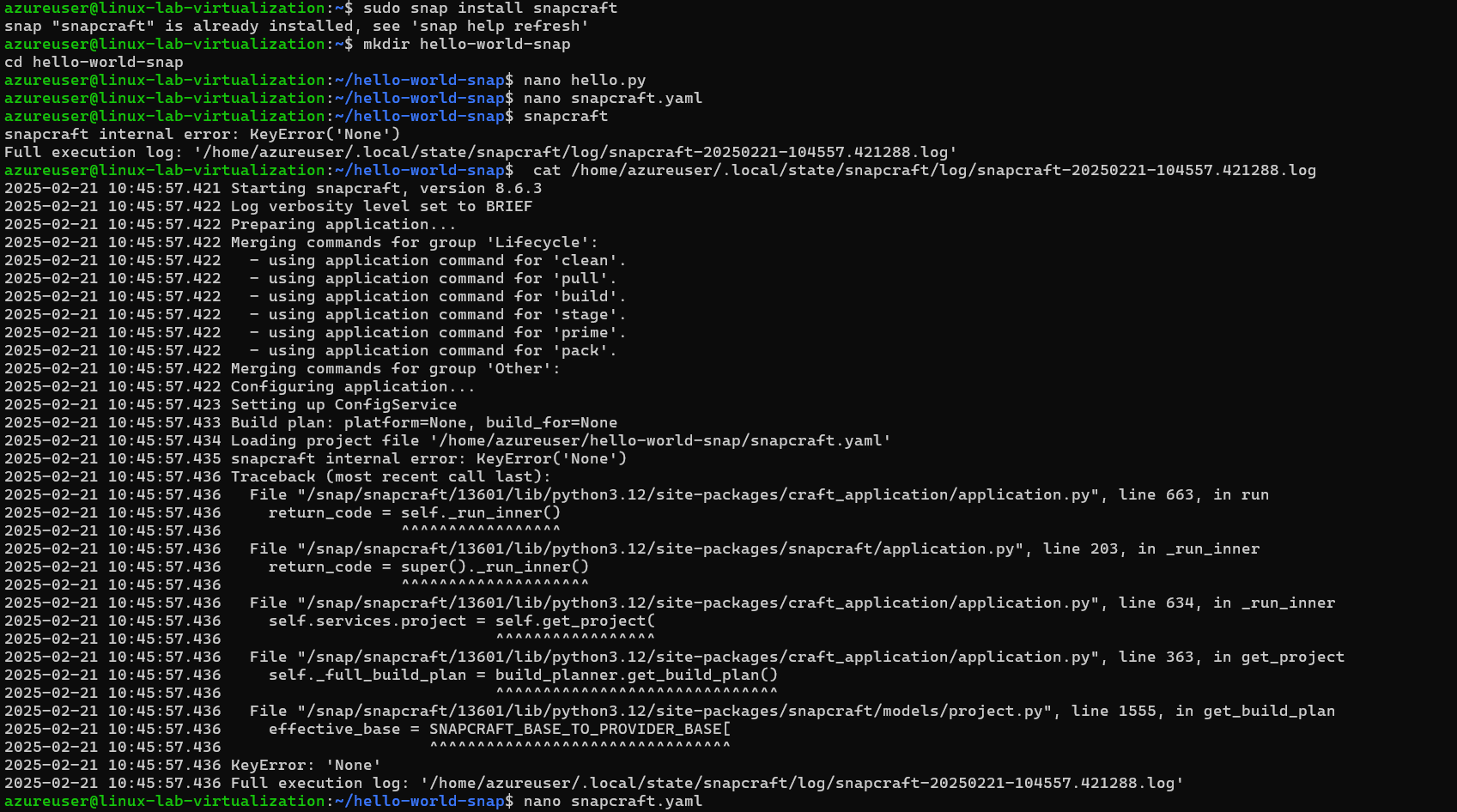
Snaps provides a way to package applications with their dependencies for consistent execution on Linux distributions.

* Research: Using sources, explore Snapcraft and the concept of Snaps.
* Experiment: Choose a simple app and compress it into Snap using Snapcraft. For more information, you can find "Snapcraft" - from the source.

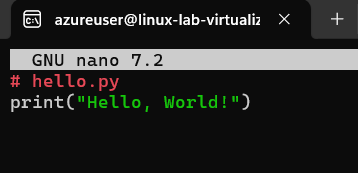
Snapcraft is a tool and ecosystem that helps developers package their applications into snaps, which are self-contained software packages that include the application and all of its dependencies. This makes it easy to distribute and run applications across different Linux distributions without worrying about compatibility issues or missing dependencies. Snaps are designed to work on most modern Linux distributions, and they are confined, meaning they run in isolated environments to ensure security.

1. Install snapcraft

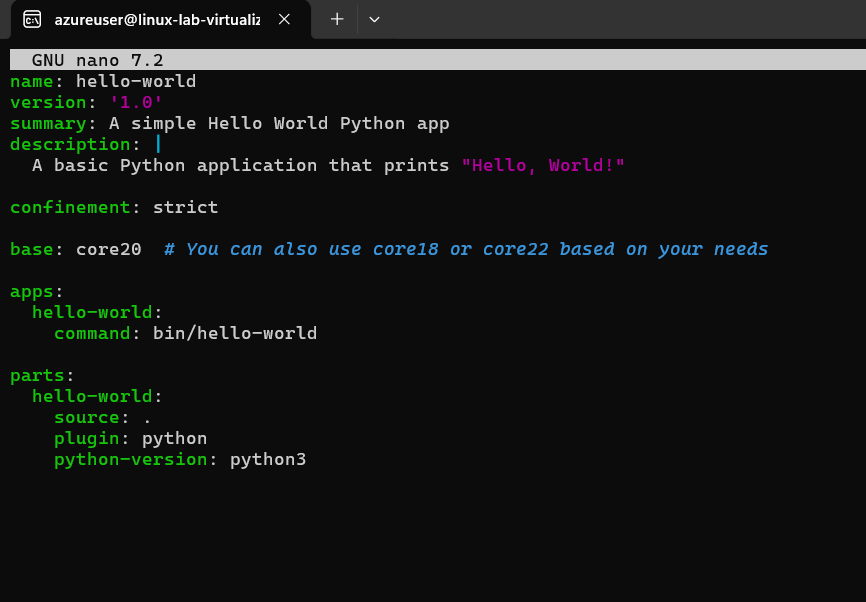
sudo apt update sudo apt install snapcraft

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/1.png)

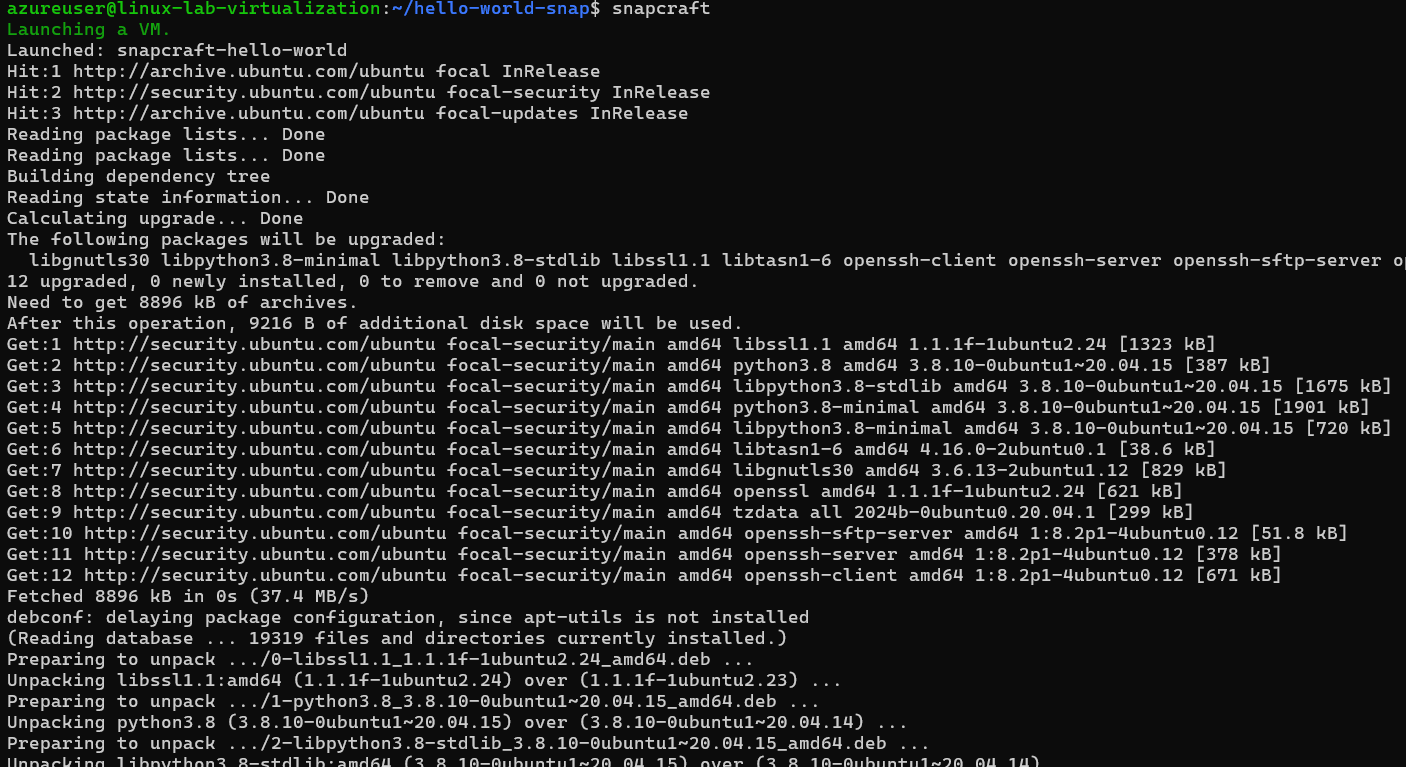
1. Create the python file

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/2.png)

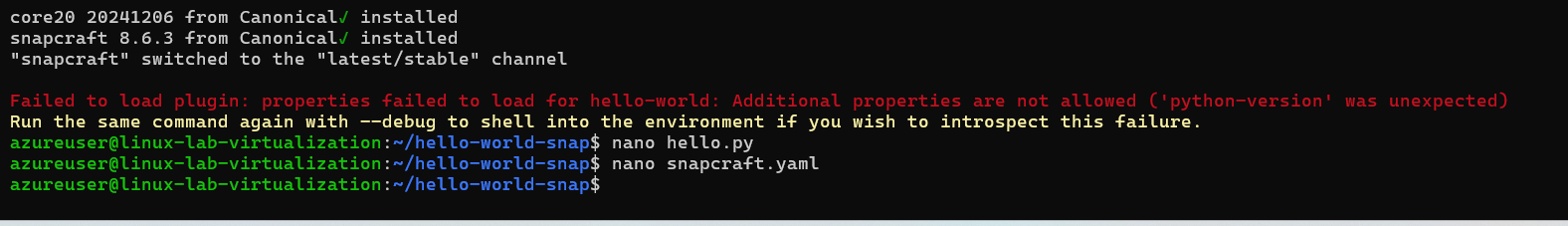
1. Create the Snapcraft Configuration

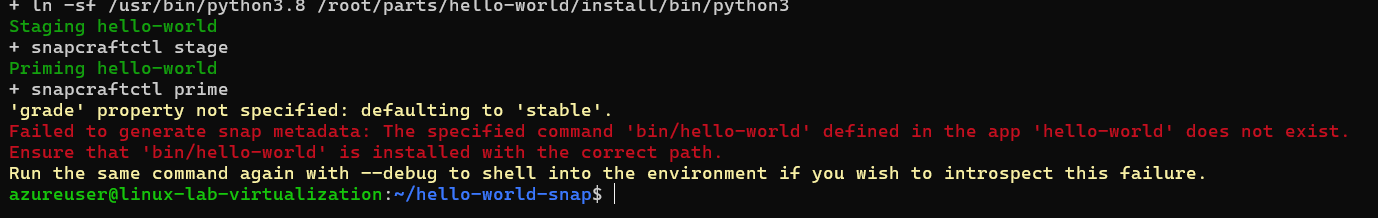
[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/3.png)

1. Fix the error by adding base: core20
2. Create the snapcraft

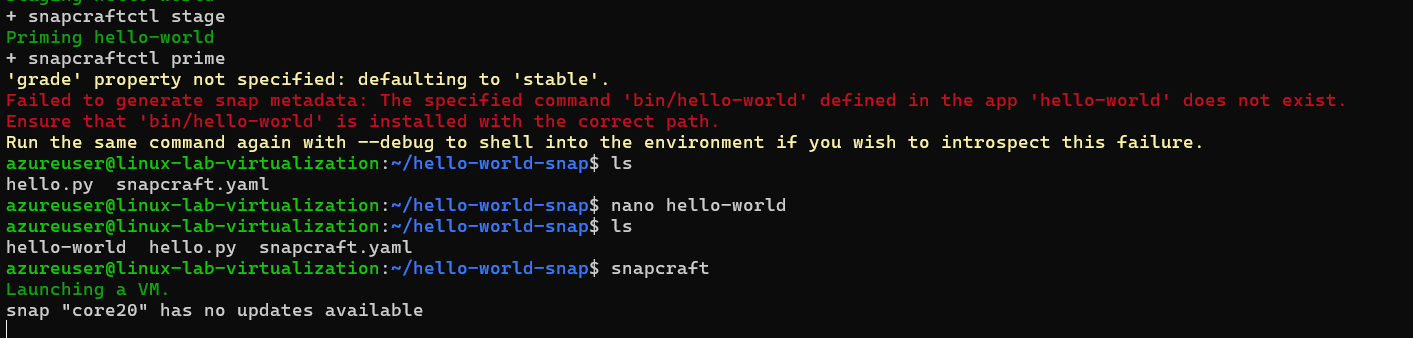
[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/4.png)

1. Remove python-version to fix the error,

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/5.png)

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/6.png)

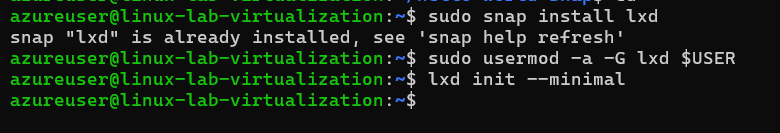
1. Check for the file hello-world and create the missing file. Re run the snap

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/7.png)

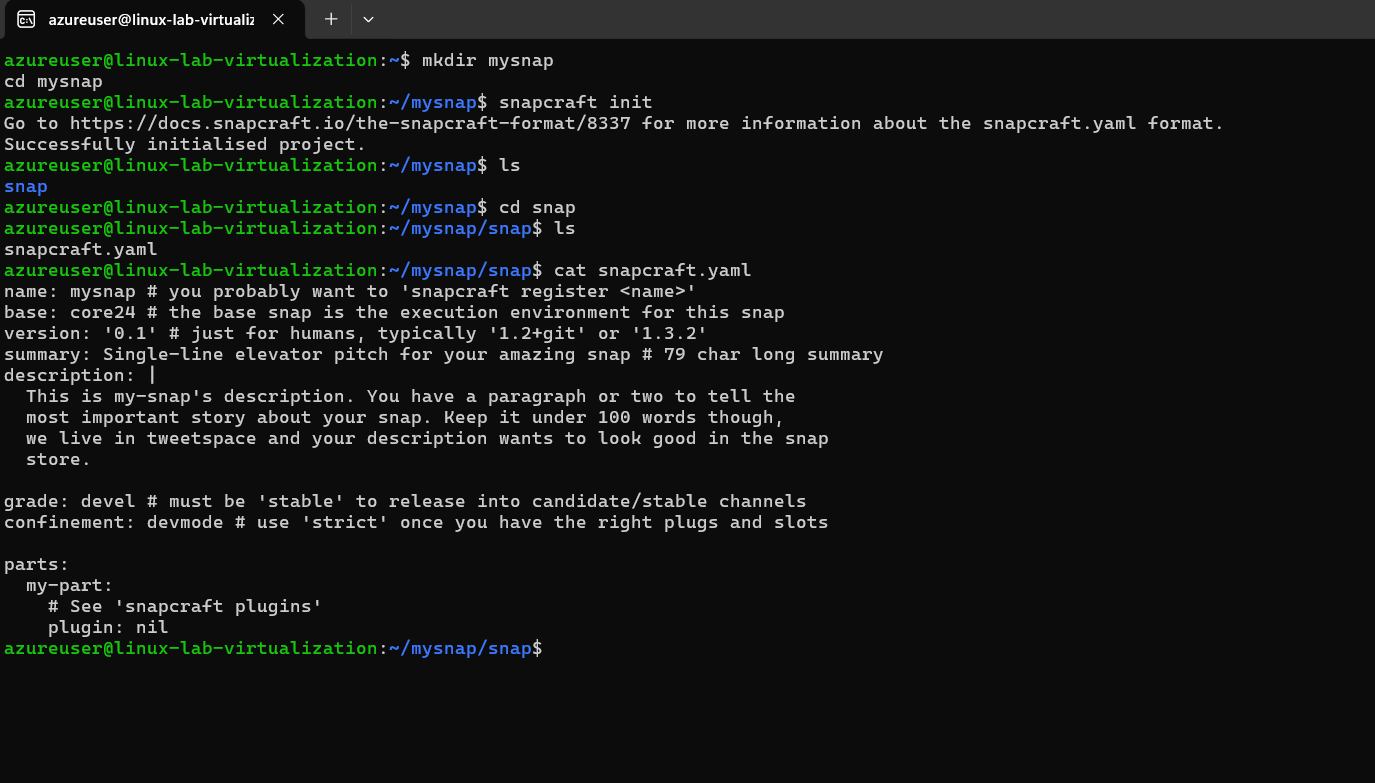
**Method 2**

1. Build snap ENV.

sudo snap install lxd

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/8.png)

1. Create YAML template

[](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/9.png)

1. Build template

[A screenshot of a computer program

AI-generated content may be incorrect.](https://github.com/Rashmika-Dineth/Linux/blob/main/Assignment%207/Images/Snap/10.png)