Name: Lujain Zia
Roll no: 2023-BSE-034
Date: October 24, 2025

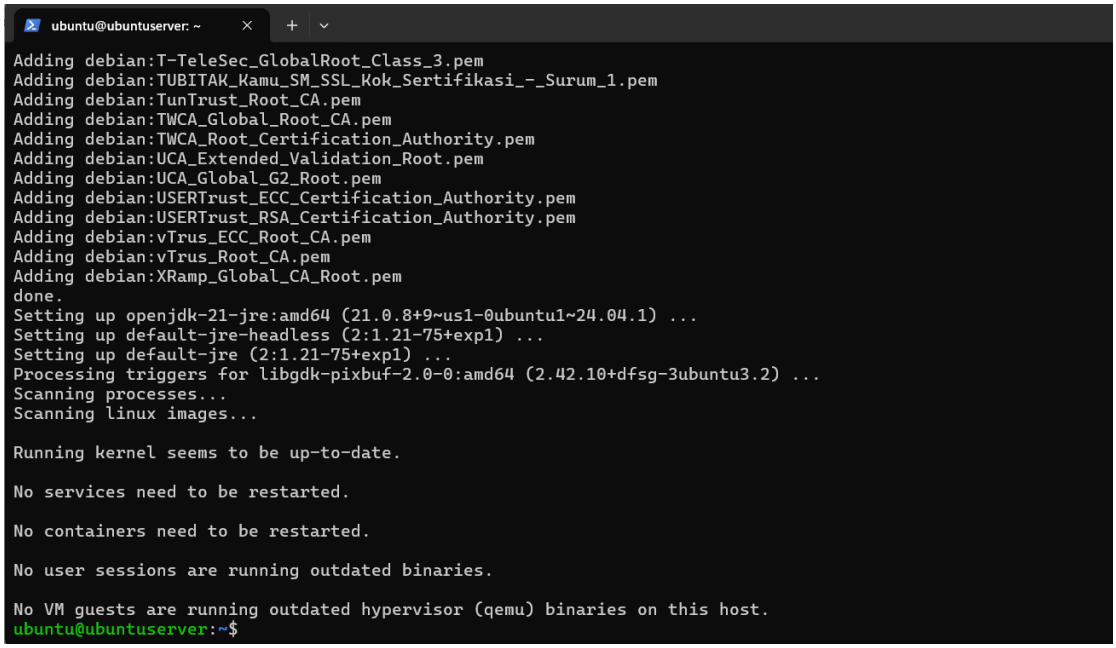
Lab 5

Task 1:
Discover missing command & install Java using apt suggestion

1. Run the java command to see what the system suggests:
java
Expected: message similar to "The program 'java' can be found in the following packages: ... Try: sudo apt install "
Save screenshot as: task1_java_suggestion.png

```
Last login: Mon Oct 20 13:48:10 2025 from 192.168.85.1
ubuntu@ubuntuserver:~$ java
Command 'java' not found, but can be installed with:
sudo apt install openjdk-17-jre-headless  # version 17.0.16+8~us1-0ubuntu1~24.04.1, or
sudo apt install openjdk-21-jre-headless  # version 21.0.8+9~us1-0ubuntu1~24.04.1
sudo apt install default-jre              # version 2:1.17-75
sudo apt install openjdk-11-jre-headless  # version 11.0.28+6-1ubuntu1~24.04.1
sudo apt install openjdk-25-jre-headless  # version 25+36-1~24.04.2
sudo apt install openjdk-8-jre-headless   # version 8u462-ga~us1-0ubuntu2~24.04.2
sudo apt install openjdk-19-jre-headless  # version 19.0.2+7-4
sudo apt install openjdk-20-jre-headless  # version 20.0.2+9-1
sudo apt install openjdk-22-jre-headless  # version 22~22ea-1
ubuntu@ubuntuserver:~$
```

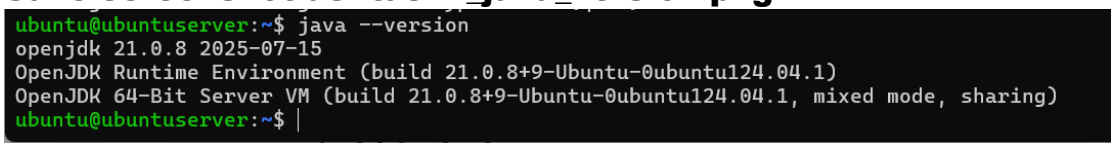
2. Use the suggested apt command (copy the exact package name suggested by the system). Example (replace with the name shown on your VM):
sudo apt install <suggested-package> -y
Save screenshot of the apt install progress (or final lines) as: task1_java_install.png



3. Verify Java is installed and check version:

java --version

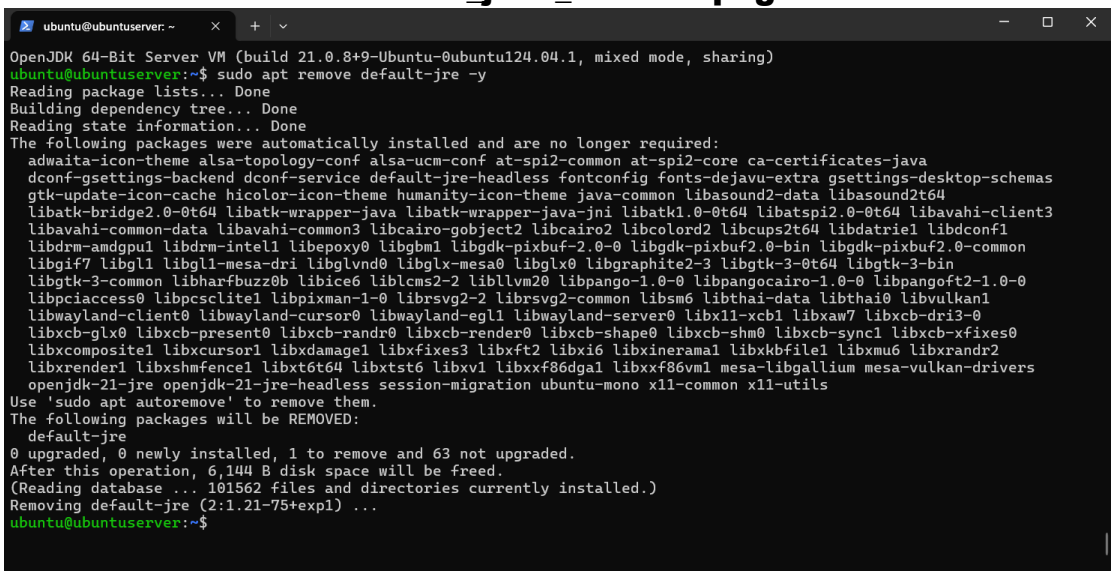
Save screenshot as: task1_java_version.png



4. Remove the Java package using apt remove (use the same package name you installed):

sudo apt remove <suggested-package> -y

Save screenshot as: task1_java_remove.png



5. Confirm java is no longer available (run java again) — it should again indicate "not found" or suggest installation:

java

Save screenshot as: task1_java_not_found.png

```
ubuntu@ubuntuserver:~$ java
Command 'java' not found, but can be installed with:
sudo apt install openjdk-17-jre-headless  # version 17.0.16+8~us1-0ubuntu1~24.04.1, or
sudo apt install openjdk-21-jre-headless  # version 21.0.8+9~us1-0ubuntu1~24.04.1
sudo apt install default-jre              # version 2:1.17-75
sudo apt install openjdk-11-jre-headless  # version 11.0.28+6-1ubuntu1~24.04.1
sudo apt install openjdk-25-jre-headless  # version 25+36-1~24.04.2
sudo apt install openjdk-8-jre-headless   # version 8u462-ga~us1-0ubuntu2~24.04.2
sudo apt install openjdk-19-jre-headless  # version 19.0.2+7-4
sudo apt install openjdk-20-jre-headless  # version 20.0.2+9-1
sudo apt install openjdk-22-jre-headless  # version 22~22ea-1
ubuntu@ubuntuserver:~$
```

6. Clear the shell's command hash cache so the shell forgets cached command locations (run as your regular user — no sudo required):

hash -r

java

Save screenshot showing hash -r followed by java result as: task1_hash_clear.png

```
ubuntu@ubuntuserver:~$ hash -r
ubuntu@ubuntuserver:~$ java
Command 'java' not found, but can be installed with:
sudo apt install openjdk-17-jre-headless  # version 17.0.16+8~us1-0ubuntu1~24.04.1, or
sudo apt install openjdk-21-jre-headless  # version 21.0.8+9~us1-0ubuntu1~24.04.1
sudo apt install default-jre              # version 2:1.17-75
sudo apt install openjdk-11-jre-headless  # version 11.0.28+6-1ubuntu1~24.04.1
sudo apt install openjdk-25-jre-headless  # version 25+36-1~24.04.2
sudo apt install openjdk-8-jre-headless   # version 8u462-ga~us1-0ubuntu2~24.04.2
sudo apt install openjdk-19-jre-headless  # version 19.0.2+7-4
sudo apt install openjdk-20-jre-headless  # version 20.0.2+9-1
sudo apt install openjdk-22-jre-headless  # version 22~22ea-1
ubuntu@ubuntuserver:~$
```

Task 2:

1. Install Java using apt-get (choose a common package, e.g., default-jre — or the same package you used in Task 1):

sudo apt-get update

sudo apt-get install default-jre -y

Save screenshot(s) as: task2_aptget_install.png

```
Preparing to unpack .../default-jre_2%3a1.21-75+exp1_amd64.deb ...
Unpacking default-jre (2:1.21-75+exp1) ...
Setting up openjdk-21-jre-headless:amd64 (21.0.8+9~us1-0ubuntu1~24.04.1) ...
update-alternatives: using /usr/lib/jvm/java-21-openjdk-amd64/bin/java to provide /usr/bin/java (java) in auto mode
update-alternatives: using /usr/lib/jvm/java-21-openjdk-amd64/bin/jpackage to provide /usr/bin/jpackage (jpackage) in au
to mode
update-alternatives: using /usr/lib/jvm/java-21-openjdk-amd64/bin/keytool to provide /usr/bin/keytool (keytool) in auto
mode
update-alternatives: using /usr/lib/jvm/java-21-openjdk-amd64/bin/rmiregistry to provide /usr/bin/rmiregistry (rmiregist
ry) in auto mode
update-alternatives: using /usr/lib/jvm/java-21-openjdk-amd64/lib/jexec to provide /usr/bin/jexec (jexec) in auto mode
Processing triggers for hicolor-icon-theme (0.17-2) ...
Processing triggers for ca-certificates-java (20240118) ...
done.
Setting up openjdk-21-jre:amd64 (21.0.8+9~us1-0ubuntu1~24.04.1) ...
Setting up default-jre-headless (2:1.21-75+exp1) ...
Setting up default-jre (2:1.21-75+exp1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ubuntuserver:~$
```

2. Verify Java version again:

java --version

Save screenshot as: task2_java_version_after_aptget.png

```
ubuntu@ubuntuserver:~$ java --version
openjdk 21.0.8 2025-07-15
OpenJDK Runtime Environment (build 21.0.8+9-Ubuntu-0ubuntu124.04.1)
OpenJDK 64-Bit Server VM (build 21.0.8+9-Ubuntu-0ubuntu124.04.1, mixed mode, sharing)
ubuntu@ubuntuserver:~$
```

3. Remove Java using apt-get remove:

sudo apt-get remove default-jre -y

Save screenshot as: task2_aptget_remove.png

```
  dconf-gsettings-backend dconf-service fontconfig fonts-dejavu-extra gsettings-desktop-schemas gtk-update-icon-cache
  hicolor-icon-theme humanity-icon-theme java-common libasound2-data libasound2t64 libatk-bridge2.0-0t64
  libatk-wrapper-java libatk-wrapper-java-jni libatk1.0-0t64 libatspi2.0-0t64 libavahi-client3 libavahi-common-data
  libavahi-common3 libcairo-gobject2 libcairo2 libcolord2 libcups2t64 libdatrie1 libdconf1 libdrm-amdgpu1
  libdrm-intel1 libepoxy0 libgbm1 libgdk-pixbuf-2.0-0 libgdk-pixbuf2.0-bin libgdk-pixbuf2.0-common libgif7 libgl1
  libgl1-mesa-dri libglvnd0 libglx-mesa0 libglx0 libgraphite2-3 libgtk-3-0t64 libgtk-3-bin libgtk-3-common
  libharfbuzz0b libice6 liblcms2-2 libllvm20 libpango-1.0-0 libpangocairo-1.0-0 libpangoft2-1.0-0 libpciaccess0
  libpcsclite1 libpixman-1-0 librsvg2-2 librsvg2-common libsm6 libthai-data libthai0 libvulkan1 libwayland-client0
  libwayland-cursor0 libwayland-egl1 libwayland-server0 libx11-xcb1 libxaw7 libxcb-dri3-0 libxcb-glx0 libxcb-present0
  libxcb-randr0 libxcb-render0 libxcb-shape0 libxcb-shm0 libxcb-sync1 libxcb-xfixes0 libxcomposite1 libxcursor1
  libxdamage1 libxfixes3 libxft2 libxi6 libxinerama1 libxkbfile1 libxmu6 libxrandr2 libxrender1 libxshmfence1
  libxt6t64 libxtst6 libxv1 libxxf86dga1 libxxf86vm1 mesa-libgallium mesa-vulkan-drivers session-migration ubuntu-mono
  x11-common x11-utils
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 64 not upgraded.
```

4. Clear the terminal hash cache and confirm java is missing:

hash -r

java

Save screenshot as: task2_hash_after_remove.png

```
ubuntu@ubuntuserver:~$ hash -r
ubuntu@ubuntuserver:~$ java
Command 'java' not found, but can be installed with:
sudo apt install openjdk-17-jre-headless  # version 17.0.16+8~us1-0ubuntu1~24.04.1, or
sudo apt install openjdk-21-jre-headless  # version 21.0.8+9~us1-0ubuntu1~24.04.1
sudo apt install default-jre              # version 2:1.17-75
sudo apt install openjdk-11-jre-headless  # version 11.0.28+6-1ubuntu1~24.04.1
sudo apt install openjdk-25-jre-headless  # version 25+36-1~24.04.2
sudo apt install openjdk-8-jre-headless   # version 8u462-ga~us1-0ubuntu2~24.04.2
sudo apt install openjdk-19-jre-headless  # version 19.0.2+7-4
sudo apt install openjdk-20-jre-headless  # version 20.0.2+9-1
sudo apt install openjdk-22-jre-headless  # version 22~22ea-1
ubuntu@ubuntuserver:~$
```

Task 3 - apt update vs apt upgrade - run & explain

1. Update the package index (this downloads the latest lists of available packages):

sudo apt update

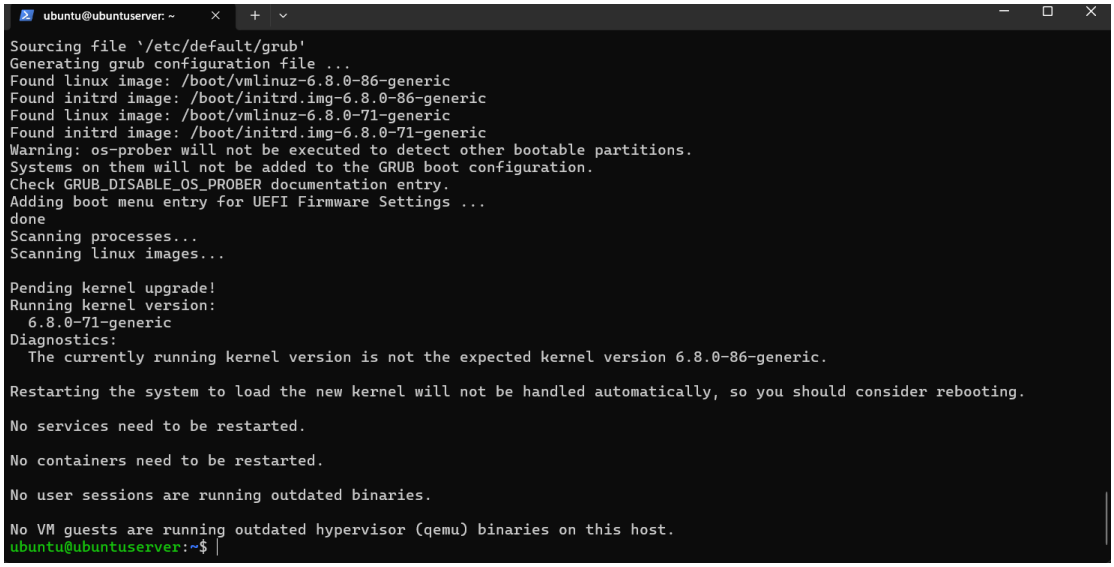
Save screenshot as: task3_apt_update.png

```
ubuntu@ubuntuserver:~$ sudo apt update
Ign:1 http://archive.ubuntu.com/ubuntu noble InRelease
Ign:2 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:2 http://security.ubuntu.com/ubuntu noble-security InRelease
Ign:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:1 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
64 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

2. Upgrade installed packages (this installs available updates for currently installed packages):

sudo apt upgrade

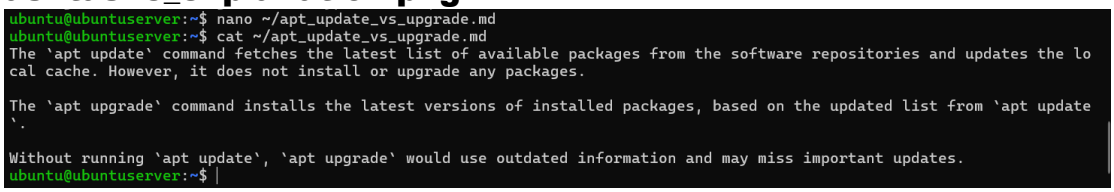
Save screenshot as: task3_apt_upgrade.png



3. Write a short 3–5 sentence explanation describing the difference between apt update and apt upgrade. Put your text into a small file and capture it as a screenshot (do not upload the text file; provide the screenshot):

nano ~/apt_update_vs_upgrade.md# write 3-5 sentences, save and exit

Open the file or show it on screen and save screenshot as: task3_explanation.png

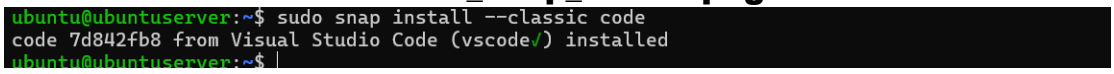


Task 4 - Install Visual Studio Code via snap on CLI and verify

1. Install VS Code via snap (snap may require sudo):

sudo snap install --classic code

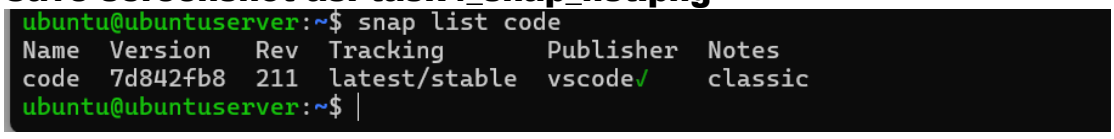
Save screenshot as: task4_snap_install.png



2. Verify snap shows the package is installed:

snap list code

Save screenshot as: task4_snap_list.png



3. Check the installed application's version. On some systems code --version is available; also check snap info:

code --version

Save screenshot(s) as: task4_code_version_or_info.png

```
ubuntu@ubuntuserver:~$ snap info code
name:      code
summary:   Code editing. Redefined.
publisher: Visual Studio Code (vscode✓)
store-url: https://snapcraft.io/code
contact:   https://twitter.com/code
license:   unset
description: |
  Visual Studio Code is a new choice of tool that combines the
  simplicity of a code editor with what developers need for the core
  edit-build-debug cycle.
commands:
  - code
  - code.url-handler
snap-id:      Ht0aUHi7ofh9Fbwh6m7jUN2pAy6kzBiu
tracking:     latest/stable
refresh-date: today at 08:59 UTC
channels:
  latest/stable:    7d842fb8 2025-10-15 (211) 346MB classic
  latest/candidate: ↑
  latest/beta:      ↑
  latest/edge:      ↑
installed:          7d842fb8            (211) 346MB classic
ubuntu@ubuntuserver:~$
```

4. If the code binary is not in PATH, show where the snap placed it:

ls -l /snap/bin | grep code

Save screenshot as: task4_snap_bin_location.png

```
ubuntu@ubuntuserver:~$ ls -l /snap/bin | grep code
lrwxrwxrwx 1 root root 13 Oct 24 08:59 code -> /usr/bin/snap
lrwxrwxrwx 1 root root 13 Oct 24 08:59 code.url-handler -> /usr/bin/snap
ubuntu@ubuntuserver:~$
```

5. IMPORTANT: Do NOT remove VS Code at the end of this task — keep it installed. It will be launched later in Task 5 and Task 6.

Task 5 - Install XFCE GUI + XRDP - minimal desktop and remote access (GUI) and launch VS Code

1. From your host, open your preferred terminal (for example: Windows Command Prompt, PowerShell, macOS Terminal, or Linux Terminal) and connect to the VM using SSH. Example: ssh student@<vm-ip-address>

2. Update the server (download package lists and apply upgrades):

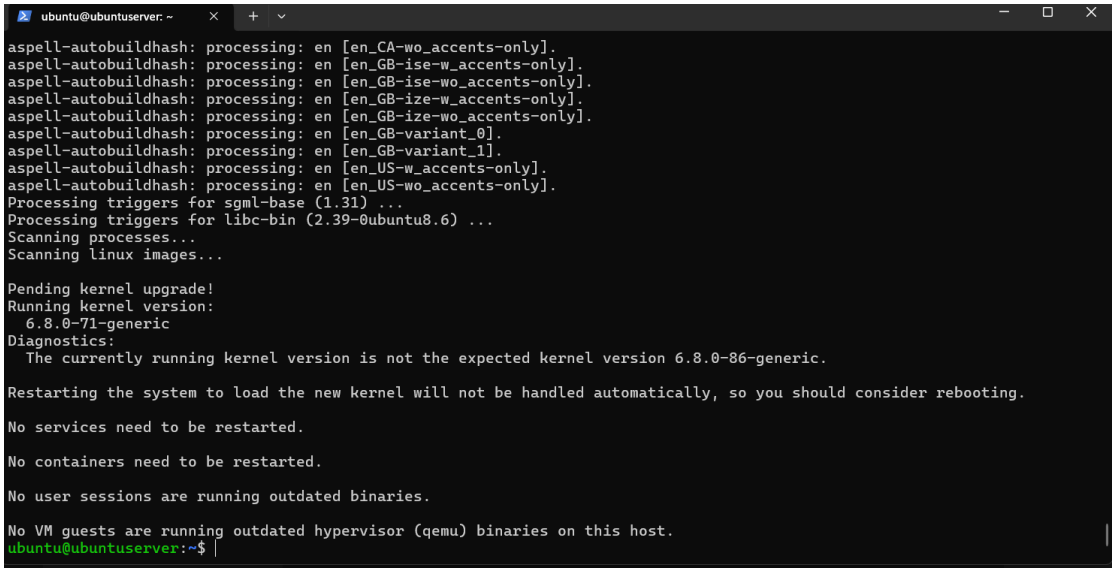
sudo apt update && sudo apt upgrade -y

Save screenshot as: task5_update.png

3. Install XFCE and XFCE goodies (lightweight desktop):

sudo apt install xfce4 xfce4-goodies -y

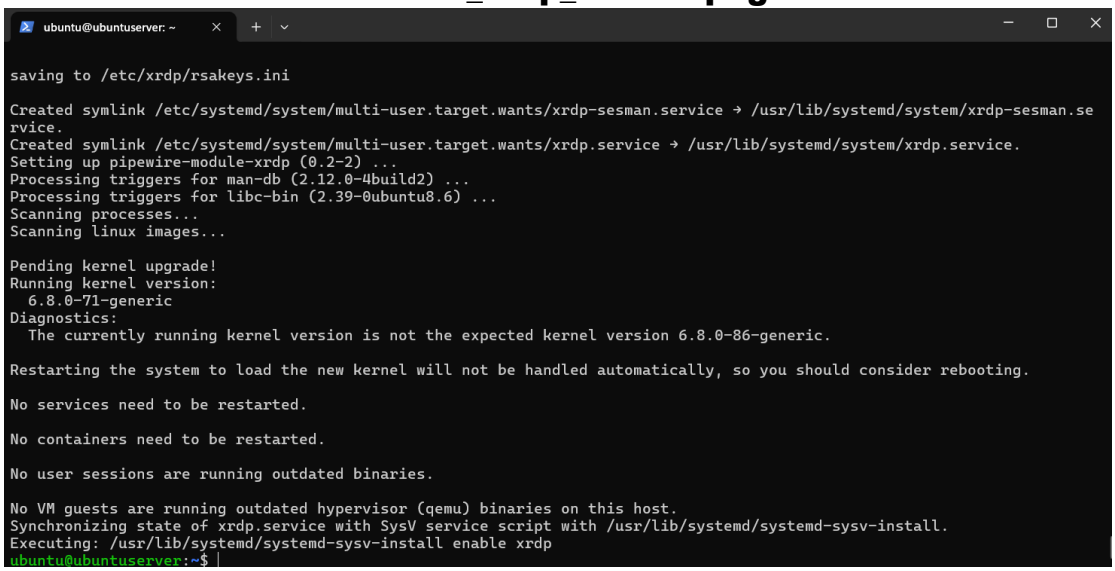
Save screenshot as: task5_xfce_install.png



4. Install and enable XRDP (Remote Desktop Protocol server):

sudo apt install xrdp -y
sudo systemctl enable --now xrdp

Save screenshot as: task5_xrdp_enable.png



5. Verify XRDP status:

sudo systemctl status xrdp

Save screenshot as: task5_xrdp_status.png

```
ubuntu@ubuntuserver:~$ sudo systemctl status xrdp
● xrdp.service - xrdp daemon
     Loaded: loaded (/usr/lib/systemd/system/xrdp.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-10-24 09:10:15 UTC; 1min 13s ago
       Docs: man:xrdp(8)
             man:xrdp.ini(5)
   Main PID: 31087 (xrdp)
      Tasks: 1 (limit: 4548)
     Memory: 864.0K (peak: 1.6M)
        CPU: 25ms
     CGroup: /system.slice/xrdp.service
             └─31087 /usr/sbin/xrdp

Oct 24 09:10:14 ubuntuserver systemd[1]: Starting xrdp.service - xrdp daemon...
Oct 24 09:10:14 ubuntuserver xrdp[31085]: [INFO ] address [0.0.0.0] port [3389] mode 1
Oct 24 09:10:14 ubuntuserver xrdp[31085]: [INFO ] listening to port 3389 on 0.0.0.0
Oct 24 09:10:14 ubuntuserver xrdp[31085]: [INFO ] xrdp_listen_pp done
Oct 24 09:10:14 ubuntuserver systemd[1]: xrdp.service: Can't open PID file /run/xrdp/xrdp.pid (yet?) after start: No su>
Oct 24 09:10:15 ubuntuserver systemd[1]: Started xrdp.service - xrdp daemon.
Oct 24 09:10:16 ubuntuserver xrdp[31087]: [INFO ] starting xrdp with pid 31087
Oct 24 09:10:16 ubuntuserver xrdp[31087]: [INFO ] address [0.0.0.0] port [3389] mode 1
Oct 24 09:10:16 ubuntuserver xrdp[31087]: [INFO ] listening to port 3389 on 0.0.0.0
Oct 24 09:10:16 ubuntuserver xrdp[31087]: [INFO ] xrdp_listen_pp done
 ESCOC

nabled; preset: enabled)
```

6. Configure XRDP to use XFCE session:

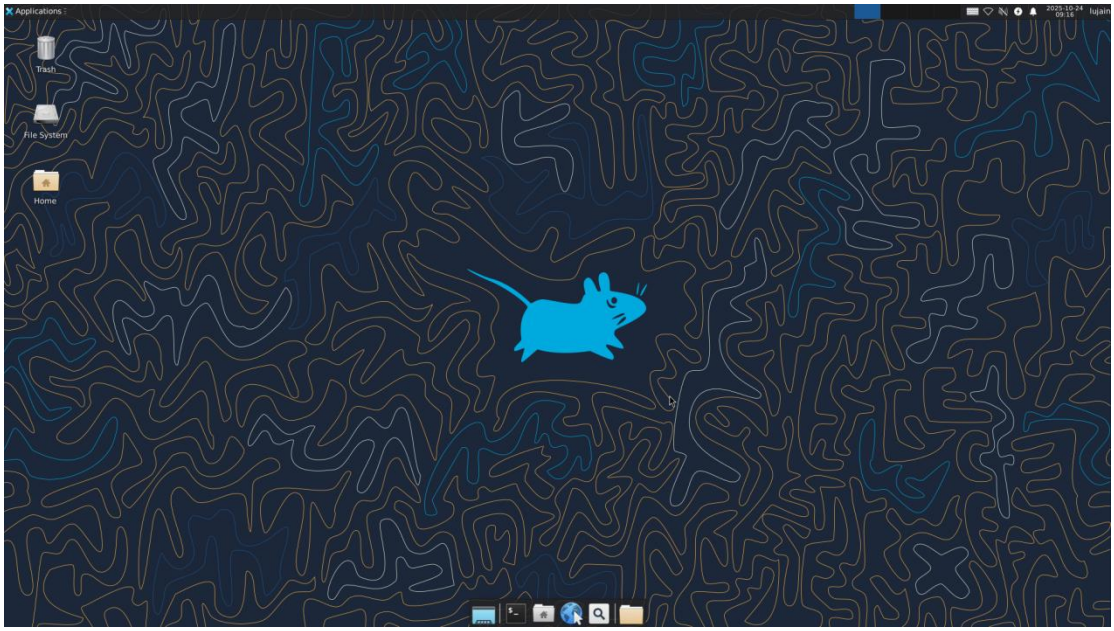
echo xfce4-session > ~/.xsession

Save screenshot as: task5_xsession.png

```
ubuntu@ubuntuserver:~$ echo xfce4-session > ~/.xsession
ubuntu@ubuntuserver:~$
```

7. From a Windows host or RDP client, connect with Remote Desktop (mstsc) to your server IP and login using your Ubuntu username/password. Capture a screenshot of the remote desktop or the RDP session window (if allowed by your environment) and save it as: task5_rdp_connect.png

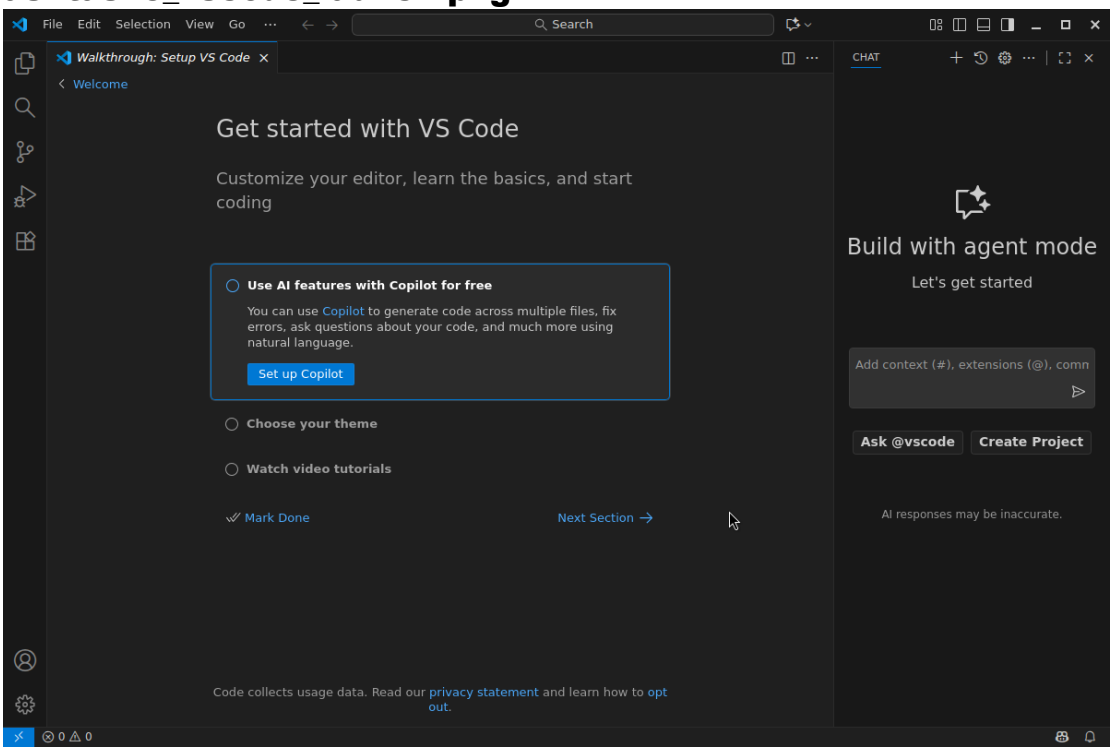
If you cannot capture a screenshot from the client, show ss -ltnp or ps evidence that the session was established and save as an alternative.



8. After you are in the GUI (local console or RDP session), launch Visual Studio Code (installed in Task 4) from the GUI menu or a terminal inside the GUI. Example command from a GUI terminal:

code

Save a screenshot of VS Code running in the GUI as: task5_vscode_launch.png



(Optional) From inside VS Code you may open a file or explore the UI — but DO NOT need to create or save files here for Task Log out of the Ubuntu server, close the session, and exit the remote desktop program.

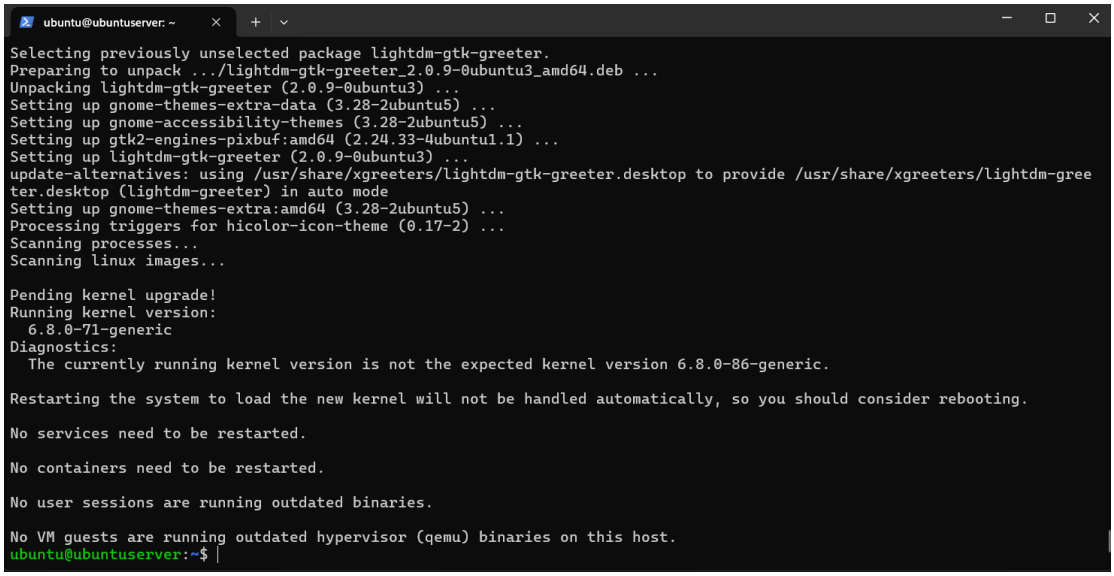
Task 6 - Install lightdm-gtk-greeter and GUI verification - start GUI, open VS Code, take snapshot, then end (GUI)

1. Fix GUI login screen issues (if lightdm / greeter problems appear)

Install LightDM and greeter using Host Terminal:

sudo apt install lightdm lightdm-gtk-greeter -y

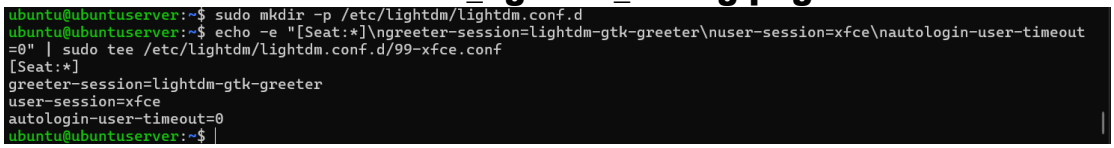
Save screenshot as: task6_lightdm_install.png



Create LightDM config to use XFCE:

sudo mkdir -p /etc/lightdm/lightdm.conf.decho -e "[Seat:*]\ngreeter-session=lightdm-gtk-greeter\nuser-session=xfce\nautologin-user-timeout=0" | sudo tee /etc/lightdm/lightdm.conf.d/99-xfce.conf

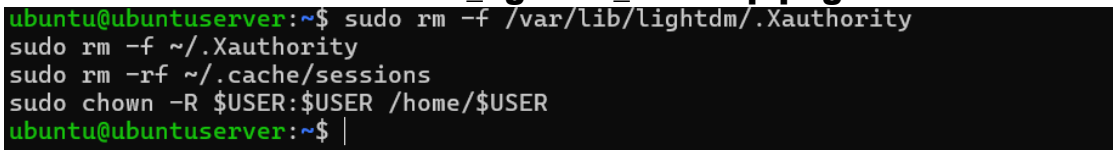
Save screenshot as: task6_lightdm_config.png



Clean up problematic session files and permissions:

sudo rm -f /var/lib/lightdm/.Xauthority
sudo rm -f ~/.Xauthority
sudo rm -rf ~/.cache/sessions
sudo chown -R $USER:$USER /home/$USER

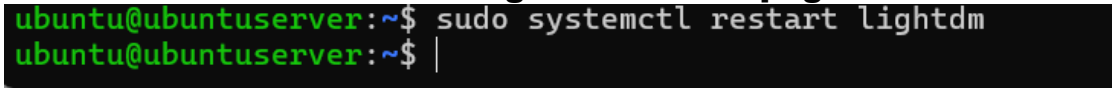
Save screenshot as: task6_lightdm_cleanup.png



- **Restart LightDM:**

sudo systemctl restart lightdm

Save screenshot as: task6_lightdm_restart.png



2. Control GUI login at boot — ENABLE first, then DISABLE (observe and understand terminal/GUI behavior after each reboot)

Important: students MUST perform the reboot after each target change to observe the boot-time behavior. The sequence below has been adjusted so you ENABLE the GUI boot target first, reboot and observe GUI, then DISABLE the GUI boot target, reboot and observe the CLI.

Enable GUI Login Screen (Boot to GUI)

Re-enable LightDM and set the graphical target as default:

sudo systemctl enable lightdm
sudo systemctl set-default graphical.target

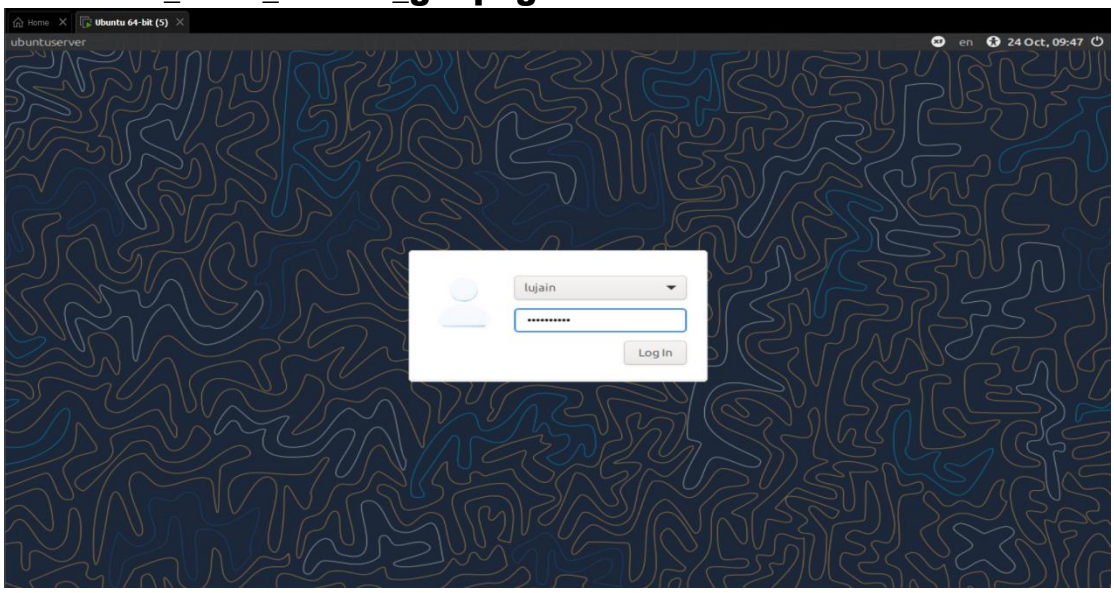
Save a screenshot immediately after running the commands as: task6_gui_enable_boot.png

```
ubuntu@ubuntuserver:~$ sudo systemctl restart lightdm
ubuntu@ubuntuserver:~$ sudo systemctl enable lightdm
sudo systemctl set-default graphical.target
Synchronizing state of lightdm.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable lightdm
The unit files have no installation config (WantedBy=, RequiredBy=, UpheldBy=,
Also=, or Alias= settings in the [Install] section, and DefaultInstance= for
template units). This means they are not meant to be enabled or disabled using systemctl.

Possible reasons for having these kinds of units are:
• A unit may be statically enabled by being symlinked from another unit's
  .wants/, .requires/, or .upholds/ directory.
• A unit's purpose may be to act as a helper for some other unit which has
  a requirement dependency on it.
• A unit may be started when needed via activation (socket, path, timer,
  D-Bus, udev, scripted systemctl call, ...).
• In case of template units, the unit is meant to be enabled with some
  instance name specified.
Created symlink /etc/systemd/system/default.target → /usr/lib/systemd/system/graphical.target.
ubuntu@ubuntuserver:~$
```

Reboot the VM to observe that it boots to the GUI login screen:

sudo reboot

After the VM boots, capture a screenshot of the GUI login screen (or remote desktop showing the greeter) and save it as: task6_after_reboot_gui.png



Disable GUI Login Screen (Boot to CLI)

Set the default boot target to multi-user (text mode) and disable LightDM so the system boots to the terminal:

sudo systemctl set-default multi-user.target
sudo systemctl disable lightdm

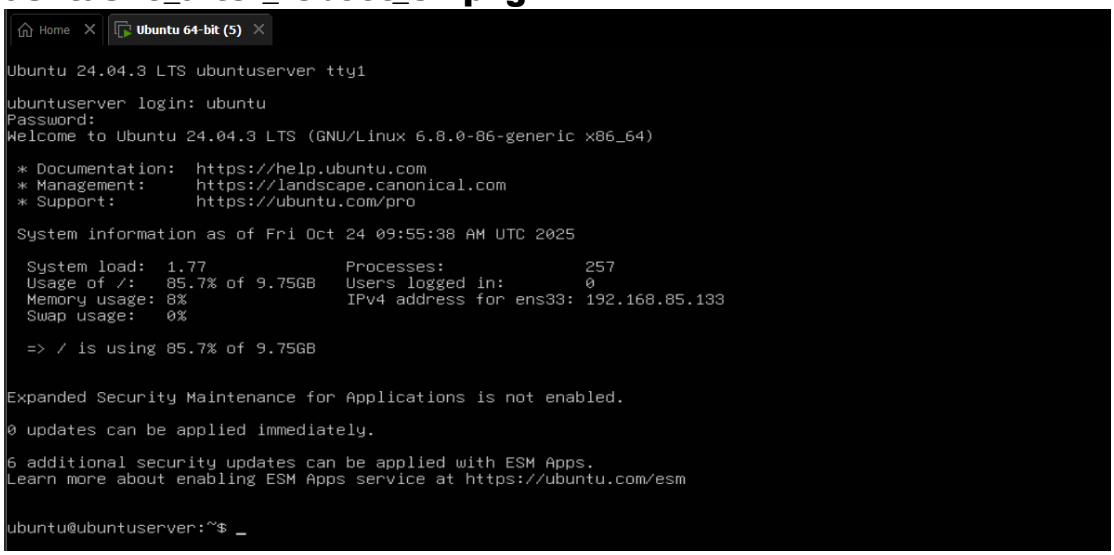
Save a screenshot immediately after running the commands as: task6_gui_disable_boot.png

```
ubuntu@ubuntuserver:~$ sudo systemctl set-default multi-user.target
sudo systemctl disable lightdm
[sudo] password for ubuntu:
Removed "/etc/systemd/system/default.target".
Created symlink /etc/systemd/system/default.target → /usr/lib/systemd/system/multi-user.target.
Synchronizing state of lightdm.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install disable lightdm
Removed "/etc/systemd/system/display-manager.service".
ubuntu@ubuntuserver:~$
```

Reboot the VM to observe that it boots to the terminal (CLI):

sudo reboot

After the VM boots, capture a screenshot of the login prompt or terminal session showing CLI-only behavior and save it as: task6_after_reboot_cli.png



Start/Stop GUI manually (no reboot)

You can start the GUI session without changing the boot target. This is useful if you want to keep the boot target as CLI but run GUI temporarily:

Save screenshot(s) showing the start commands and any immediate status output as: task6_gui_start.png

sudo systemctl start lightdm # start GUI now

```
Last login: Fri Oct 24 09:53:26 2025 from 192.168.85.1
ubuntu@ubuntuserver:~$ sudo systemctl start lightdm
[sudo] password for ubuntu:
ubuntu@ubuntuserver:~$
```

Press Ctrl + Alt + F3 to switch back to TTY. You can stop the GUI session without changing the boot target.

sudo systemctl stop lightdm # stop GUI now

Save screenshot(s) showing the start/stop commands and any immediate status output as: task6_gui_stop.png

```
ubuntu@ubuntuserver:~$ sudo systemctl stop lightdm
ubuntu@ubuntuserver:~$
```

3. Start the GUI (if the system is currently set to CLI or the GUI is not running):

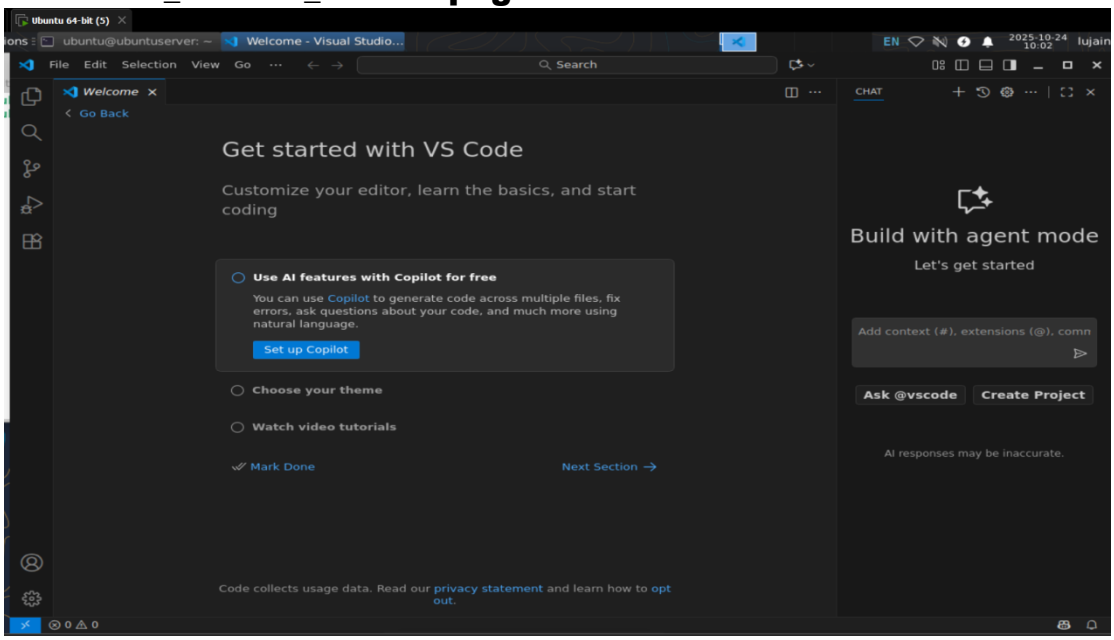
sudo systemctl start lightdm

Save a screenshot immediately after running the command as: task6_gui_start_command.png

```
ubuntu@ubuntuserver:~$ sudo systemctl start lightdm
ubuntu@ubuntuserver:~$
```

4. In the GUI session launch Visual Studio Code (installed earlier in Task 4). From a GUI terminal inside the desktop, run: code

Save a screenshot of VS Code running in the GUI as: task6_vscode_launch.png



5. Close VS Code using the GUI window controls. Task 6 ends here.

Task 7 - Install Google Chrome by adding its apt source & key (Chrome)

1. (Learning step — first command must be the install attempt) Attempt to install Google Chrome directly to see the failure when the repo/key are missing:

sudo apt install google-chrome-stable -y

Expected: this will typically fail with "Unable to locate package google-chrome-stable" or similar because the Google repo/key are not yet configured.

Save a screenshot of the error output as: task7_install_chrome_error.png

```
ubuntu@ubuntuserver:~$ sudo apt install google-chrome-stable -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package google-chrome-stable
ubuntu@ubuntuserver:~$
```

2. Inspect apt configuration so you understand why install failed. List the /etc/apt directory:

ls -la /etc/apt

Save screenshot as: task7_ls_etc_apt.png

```
ubuntu@ubuntuserver:~$ ls -la /etc/apt
total 48
drwxr-xr-x   9 root root  4096 Sep 27 01:39 .
drwxr-xr-x 142 root root 12288 Oct 24 09:15 ..
drwxr-xr-x   2 root root  4096 Oct 24 08:45 apt.conf.d
drwxr-xr-x   2 root root  4096 Mar 31  2024 auth.conf.d
drwxr-xr-x   2 root root  4096 Mar 31  2024 keyrings
drwxr-xr-x   2 root root  4096 Aug  5 17:14 preferences.d
drwxr-xr-x   2 root root  4096 Aug  5 17:14 preferences.d.save
-rw-r--r--   1 root root    70 Sep 27 01:43 sources.list
drwxr-xr-x   2 root root  4096 Sep 27 01:43 sources.list.d
drwxr-xr-x   2 root root  4096 Aug  5 17:01 trusted.gpg.d
ubuntu@ubuntuserver:~$
```

3. View the main /etc/apt/sources.list:

cat /etc/apt/sources.list

Save screenshot as: task7_cat_sources_list.png

```
ubuntu@ubuntuserver:~$ cat /etc/apt/sources.list
# Ubuntu sources have moved to /etc/apt/sources.list.d/ubuntu.sources
ubuntu@ubuntuserver:~$
```

4. List files under /etc/apt/sources.list.d:

ls -la /etc/apt/sources.list.d/

Save screenshot as: task7_ls_sources_list_d.png

```
ubuntu@ubuntuserver:~$ ls -la /etc/apt/sources.list.d/
total 16
drwxr-xr-x 2 root root 4096 Sep 27 01:43 .
drwxr-xr-x 9 root root 4096 Sep 27 01:39 ..
-rw-r--r-- 1 root root  383 Sep 27 01:43 ubuntu.sources
-rw-r--r-- 1 root root 2552 Aug  5 17:02 ubuntu.sources.curtin.orig
ubuntu@ubuntuserver:~$
```

5. If there is a file named ubuntu.sources (or similarly named source file), display it to see whether Chrome's repo is present:

cat /etc/apt/sources.list.d/ubuntu.sources

Save screenshot as: task7_cat_ubuntu_sources.png

```
ubuntu@ubuntuserver:~$ cat /etc/apt/sources.list.d/ubuntu.sources
Types: deb
URIs: http://archive.ubuntu.com/ubuntu/
Suites: noble noble-updates noble-backports
Components: main restricted universe multiverse
Signed-By: /usr/share/keyrings/ubuntu-archive-keyring.gpg

Types: deb
URIs: http://security.ubuntu.com/ubuntu/
Suites: noble-security
Components: main restricted universe multiverse
Signed-By: /usr/share/keyrings/ubuntu-archive-keyring.gpg
ubuntu@ubuntuserver:~$
```

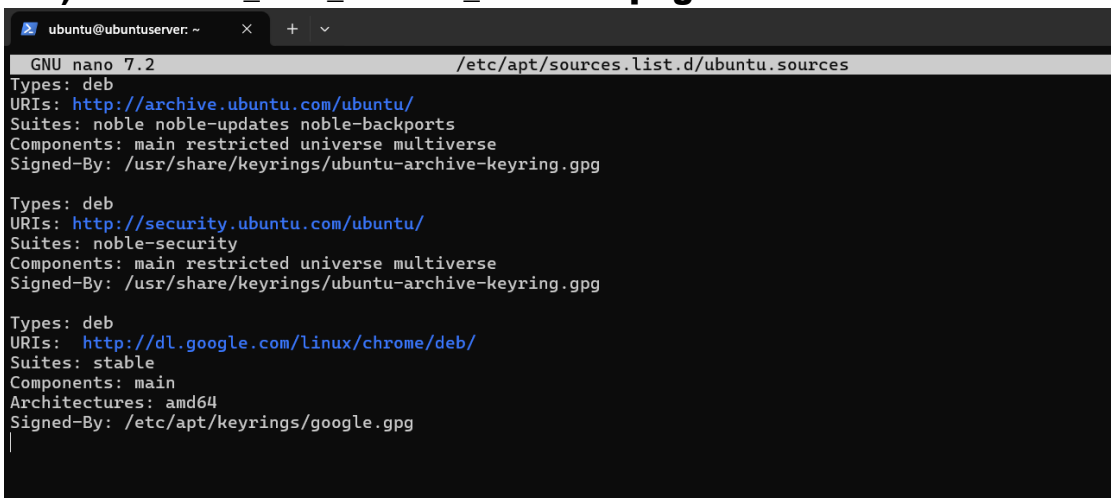
6. Add Chrome repository metadata to a sources file (method A — using ubuntu.sources). Open or create the file and append the stanza (you can alternatively use the preferred one-line method in step 11):

sudo nano /etc/apt/sources.list.d/ubuntu.sources

Append these exact lines at the end of the file:

Types: deb
URIs: http://dl.google.com/linux/chrome/deb/
Suites: stable
Components: main
Architectures: amd64
Signed-By: /etc/apt/keyrings/google.gpg
Save the file (Ctrl+O → Enter) and exit (Ctrl+X).

Save a screenshot after editing (or show the file contents with cat) as: task7_edit_ubuntu_sources.png



7. Ensure the keyrings directory exists and import Google's signing key:

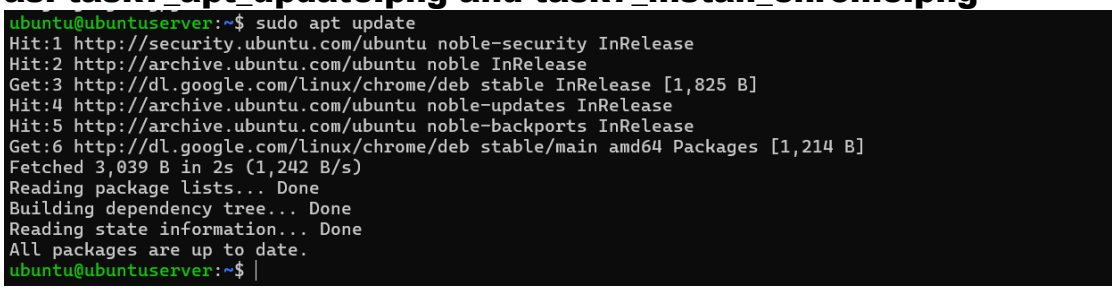
curl -fsSL https://dl.google.com/linux/linux_signing_key.pub | sudo gpg --dearmor -o /etc/apt/keyrings/google.gpg

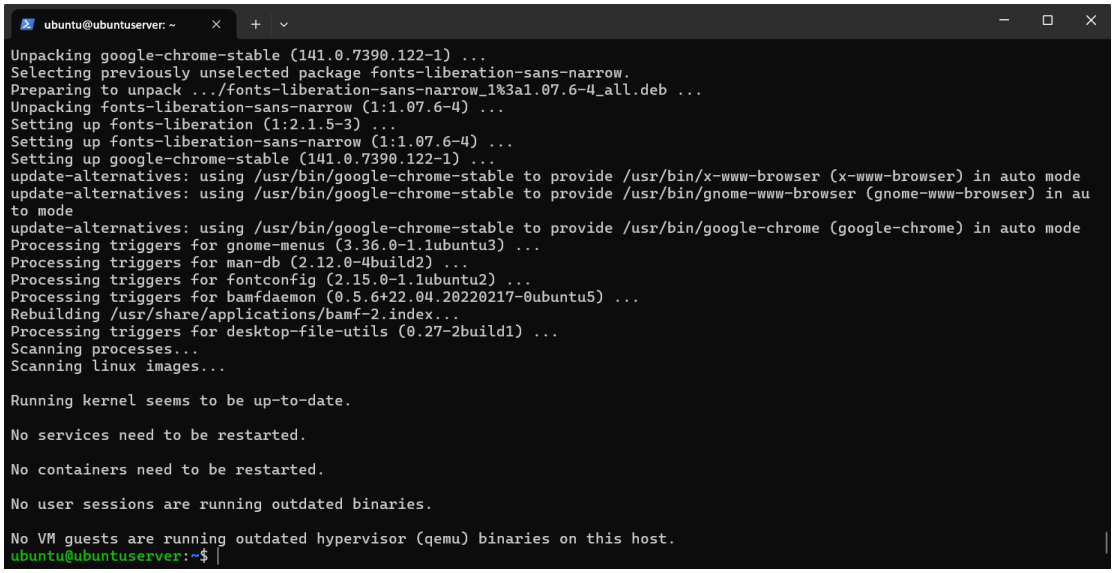
Save screenshot as: task7_add_key.png

8. Update apt and attempt to install Chrome again (now that repo + key are added):

sudo apt update
sudo apt install google-chrome-stable -y

Save screenshots as: task7_apt_update.png and task7_install_chrome.png



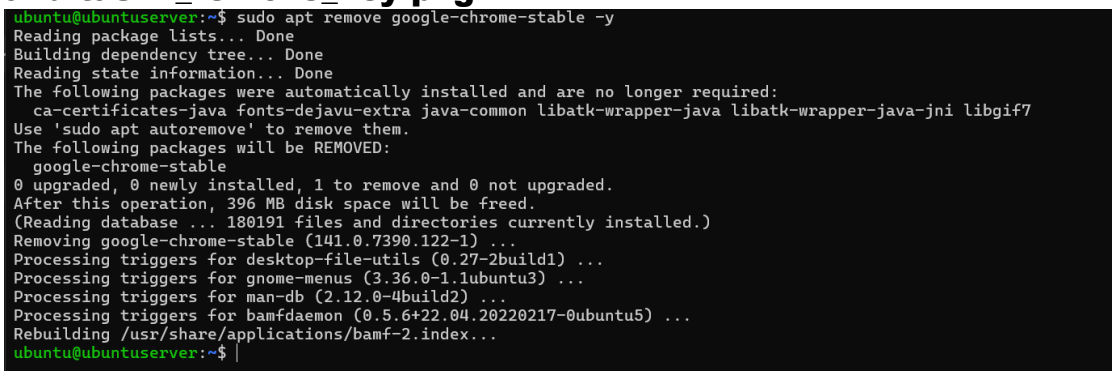


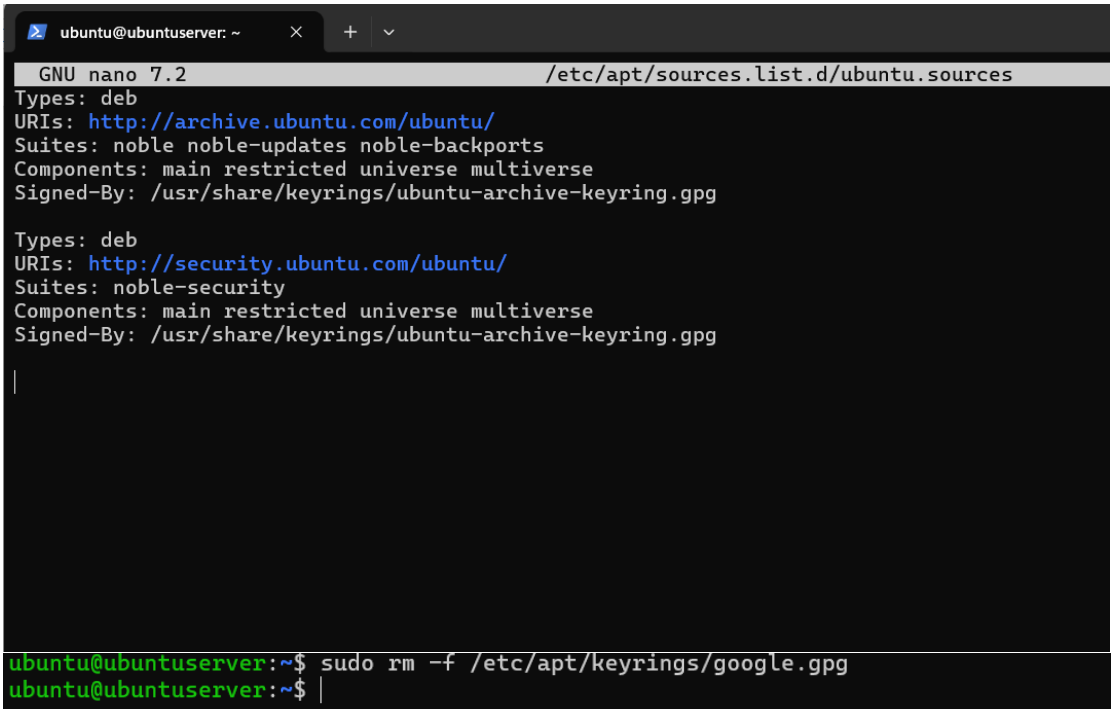
9. Alternate (preferred, cleaner) method — create a single google-chrome.list entry

10. Cleanup before alternate method you added the chrome earlier and want to switch to the preferred method:

sudo apt remove google-chrome-stable -y
sudo nano /etc/apt/sources.list.d/ubuntu.sources # remove the chrome stanza you added earlier, save and exit
sudo rm -f /etc/apt/keyrings/google.gpg

Save screenshots as: task7_alternate_remove.png, task7_alternate_edit.png, and task7_remove_key.png





11. Create a dedicated one-line list file for Google Chrome (preferred):

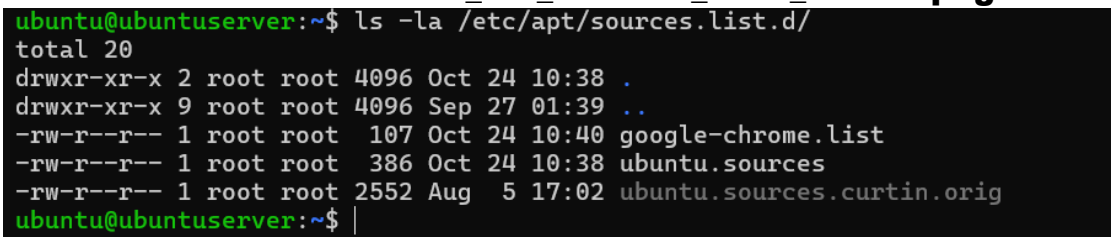
echo "deb [arch=amd64 signed-by=/etc/apt/keyrings/google.gpg] http://dl.google.com/linux/chrome/deb/ stable main" | sudo tee /etc/apt/sources.list.d/google-chrome.list > /dev/null

Save screenshot as: task7_create_google_chrome_list.png

12. Verify the new file exists:

ls -la /etc/apt/sources.list.d/

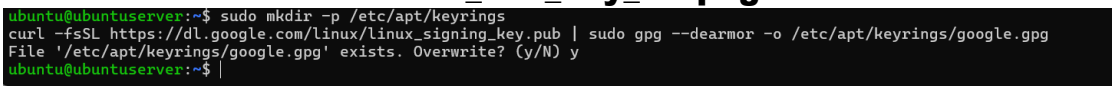
Save screenshot as: task7_list_sources_after_create.png



13. Re-add the Google signing key (if removed previously or not present):

sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://dl.google.com/linux/linux_signing_key.pub | sudo gpg --dearmor -o /etc/apt/keyrings/google.gpg

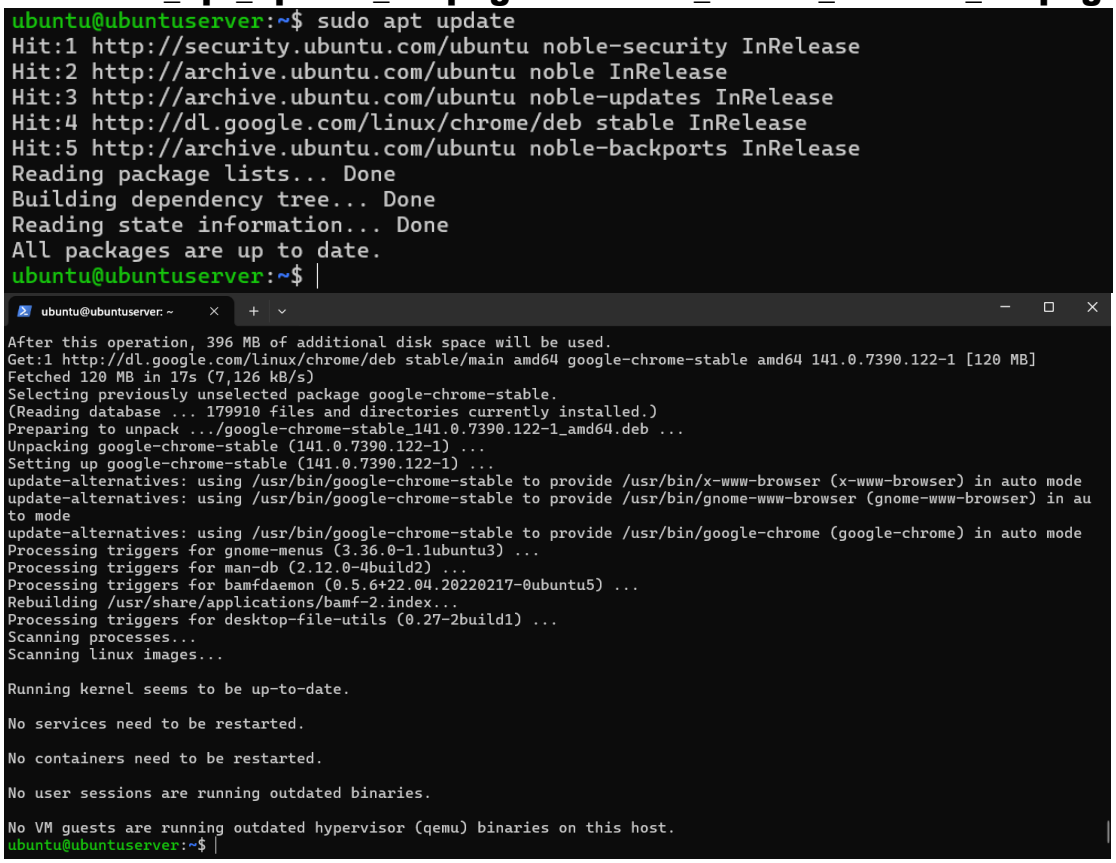
Save screenshot as: task7_add_key_alt.png



14. Update apt and install Chrome (preferred flow):

sudo apt update
sudo apt install google-chrome-stable -y

Save screenshots
as: task7_apt_update_alt.png and task7_install_chrome_alt.png

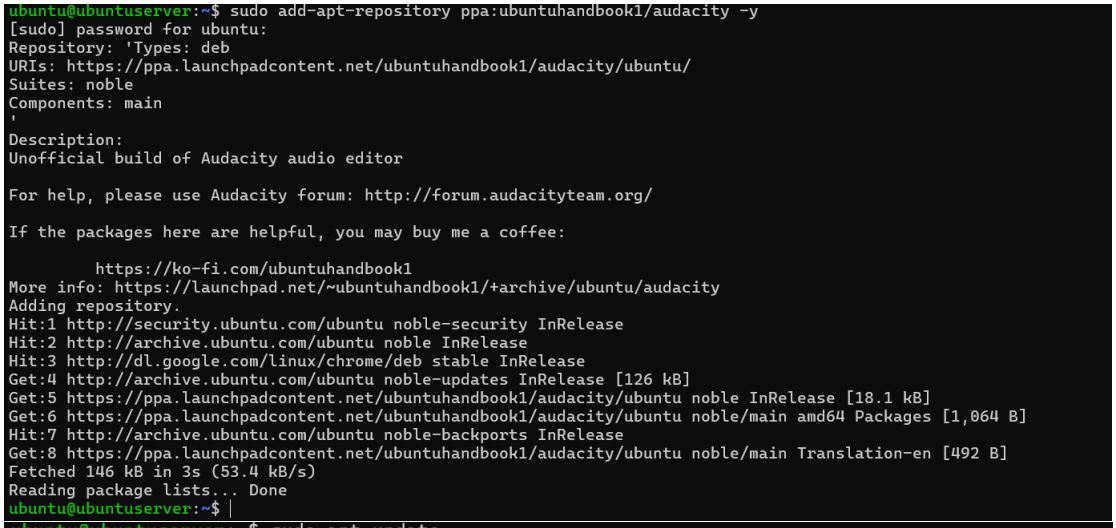


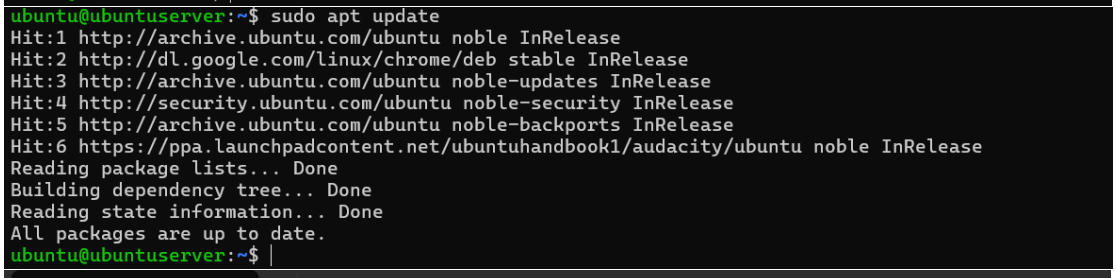
Task 8 - Install applications via PPA (Audacity & OBS) and launch

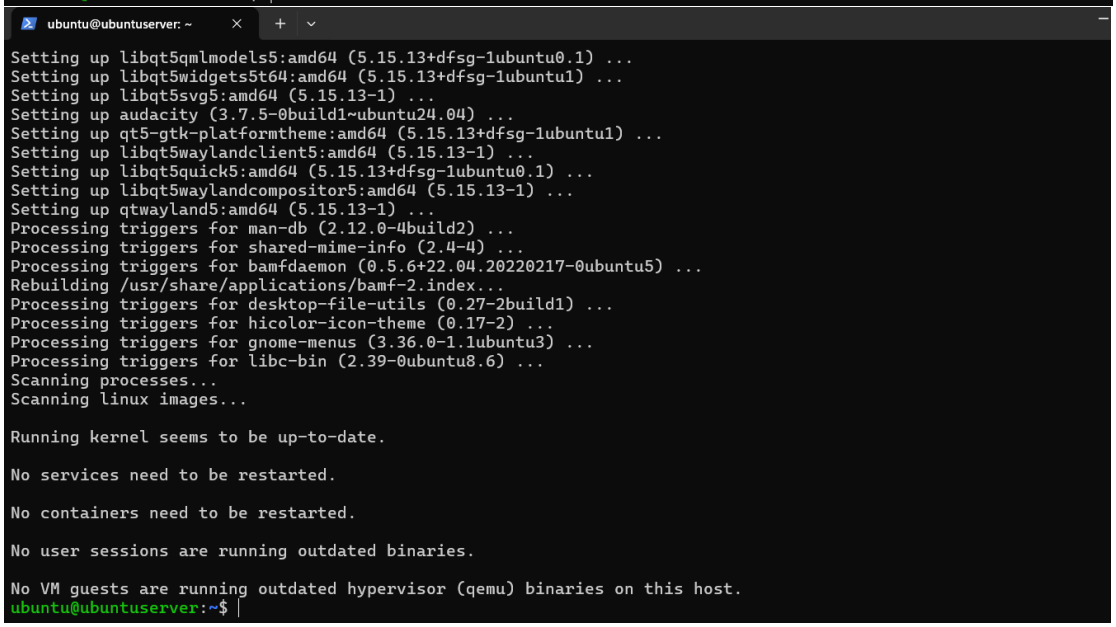
1. Add the Audacity PPA, update apt and install audacity:

sudo add-apt-repository ppa:ubuntuhandbook1/audacity -y
sudo apt update
sudo apt install audacity -y

Save screenshots as:
task8_add_ppa_audacity.png (output of add-apt-repository)
task8_apt_update_audacity.png (apt update after adding PPA)
task8_install_audacity.png (apt install output)







2. Launch Audacity (from GUI or CLI). On a headless server you may not get a GUI window — if you are using the XFCE GUI session, launch from a GUI terminal or run check for binary:

audacity --version

audacity

Save screenshots as:

task8_audacity_launch.png (GUI launch screenshot if possible) or task8_audacity_version.png (CLI verification)

```
ubuntu@ubuntuserver:~$ apt show audacity
Package: audacity
Version: 3.7.5-0build1~ubuntu24.04
Priority: optional
Section: sound
Maintainer: Jhonny Oliveira <jhonny.oliveira@gmail.com>
Original-Maintainer: Debian Multimedia Maintainers <debian-multimedia@lists.debi
an.org>
Installed-Size: 26.2 MB
Depends: audacity-data (= 3.7.5-0build1~ubuntu24.04), libasound2t64 (>= 1.0.16),
 libc6 (>= 2.38), libexpat1 (>= 2.0.1), libflac++10 (>= 1.3.1), libflac12t64 (>=
 1.3.0), libgcc-s1 (>= 4.0), libgdk-pixbuf-2.0-0 (>= 2.22.0), libglib2.0-0t64 (>
= 2.12.0), libgtk-3-0t64 (>= 3.9.10), libid3tag0 (>= 0.15.1b), liblilv-0-0 (>= 0
.24.0~dfsg0), libmpg123-0t64 (>= 1.28.0), libogg0 (>= 1.0rc3), libopus0 (>= 1.1)
, libopusfile0 (>= 0.5), libportaudio2 (>= 19+svn20101113), libportmidi0, libpor
tsmf0t64 (>= 0.1~svn20101010), libsbsms10 (>= 2.3.0), libsndfile1 (>= 1.0.20), l
ibsoundtouch1 (>= 2.0.0), libsoxr0 (>= 0.1.0), libsqlite3-0 (>= 3.20.0), libstdc
++6 (>= 13.1), libsuil-0-0 (>= 0.8.0~dfsg0), libtwolame0 (>= 0.3.6), libuuid1 (>
= 2.16), libvamp-hostsdk3v5 (>= 2.10.0), libvorbis0a (>= 1.1.2), libvorbisenc2 (
>= 1.1.2), libvorbisfile3 (>= 1.1.2), libwavpack1 (>= 5.2.0), libwxbase3.2-1t64
```

3. Add the OBS Studio PPA, update apt and install obs-studio:

sudo add-apt-repository ppa:obsproject/obs-studio -y
sudo apt update
sudo apt install obs-studio -y

Save screenshots as:

task8_add_ppa_obs.png (output of add-apt-repository)

task8_apt_update_obs.png (apt update after adding PPA)

task8_install_obs.png (apt install output)

```
ubuntu@ubuntuserver:~$ sudo add-apt-repository ppa:obsproject/obs-studio -y
Repository: 'Types: deb
URIs: https://ppa.launchpadcontent.net/obsproject/obs-studio/ubuntu/
Suites: noble
Components: main
'
Description:
Latest stable release of OBS Studio
More info: https://launchpad.net/~obsproject/+archive/ubuntu/obs-studio
Adding repository.
Hit:1 http://dl.google.com/linux/chrome/deb stable InRelease
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:3 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:4 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Get:5 https://ppa.launchpadcontent.net/obsproject/obs-studio/ubuntu noble InRelease [17.8 kB]
Hit:6 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:7 https://ppa.launchpadcontent.net/ubuntuhandbook1/audacity/ubuntu noble InRelease
Get:8 https://ppa.launchpadcontent.net/obsproject/obs-studio/ubuntu noble/main amd64 Packages [1,172 B]
Get:9 https://ppa.launchpadcontent.net/obsproject/obs-studio/ubuntu noble/main Translation-en [160 B]
Fetched 19.1 kB in 3s (7,143 B/s)
Reading package lists... Done
ubuntu@ubuntuserver:~$
```

```
ubuntu@ubuntuserver:~$ sudo apt update
Hit:1 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:2 http://dl.google.com/linux/chrome/deb stable InRelease [1,825 B]
Hit:3 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:4 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:5 https://ppa.launchpadcontent.net/obsproject/obs-studio/ubuntu noble InRelease
Hit:6 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:7 https://ppa.launchpadcontent.net/ubuntuhandbook1/audacity/ubuntu noble InRelease
Get:8 http://dl.google.com/linux/chrome/deb stable/main amd64 Packages [1,214 B]
Fetched 3,039 B in 3s (1,034 B/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
ubuntu@ubuntuserver:~$
```

```
ubuntu@ubuntuserver: ~
Setting up libqt6waylandcompositor6:amd64 (6.4.2-5build3) ...
Setting up libqt6waylandeglcompositorhwintegration6:amd64 (6.4.2-5build3) ...
Setting up libqt6waylandeglclienthwintegration6:amd64 (6.4.2-5build3) ...
Setting up libpocketsphinx3:amd64 (0.8.0+real5prealpha+1-15ubuntu5) ...
Setting up qt6-wayland:amd64 (6.4.2-5build3) ...
Setting up libavfilter9:amd64 (7:6.1.1-3ubuntu5) ...
Setting up libavdevice60:amd64 (7:6.1.1-3ubuntu5) ...
Setting up obs-studio (32.0.0-0obsproject1~noble) ...
Processing triggers for bamfdaemon (0.5.6+22.04.20220217-0ubuntu5) ...
Rebuilding /usr/share/applications/bamf-2.index...
Processing triggers for desktop-file-utils (0.27-2build1) ...
Processing triggers for hicolor-icon-theme (0.17-2) ...
Processing triggers for gnome-menus (3.36.0-1.1ubuntu3) ...
Processing triggers for libc-bin (2.39-0ubuntu8.6) ...
Processing triggers for man-db (2.12.0-4build2) ...
/usr/bin/mandb: can't write to /var/cache/man/9456: No space left on device
/usr/bin/mandb: can't create index cache /var/cache/man/9456: No space left on device
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ubuntuserver:~$
```

4. Launch OBS Studio (from GUI or verify binary presence):

obs --version

obs

Save screenshots as:

task8_obs_launch.png (GUI launch screenshot if possible)

or task8_obs_version.png (CLI verification)

```
ubuntu@ubuntuserver:~$ obs --version
OBS Studio - 32.0.0
ubuntu@ubuntuserver:~$
```

Task 9 - Create a Kubernetes sample YAML using vim

1. Check whether vim is installed by running:

vim

If vim opens, you will see the vim interface; if not installed you will see a command-not-found error.

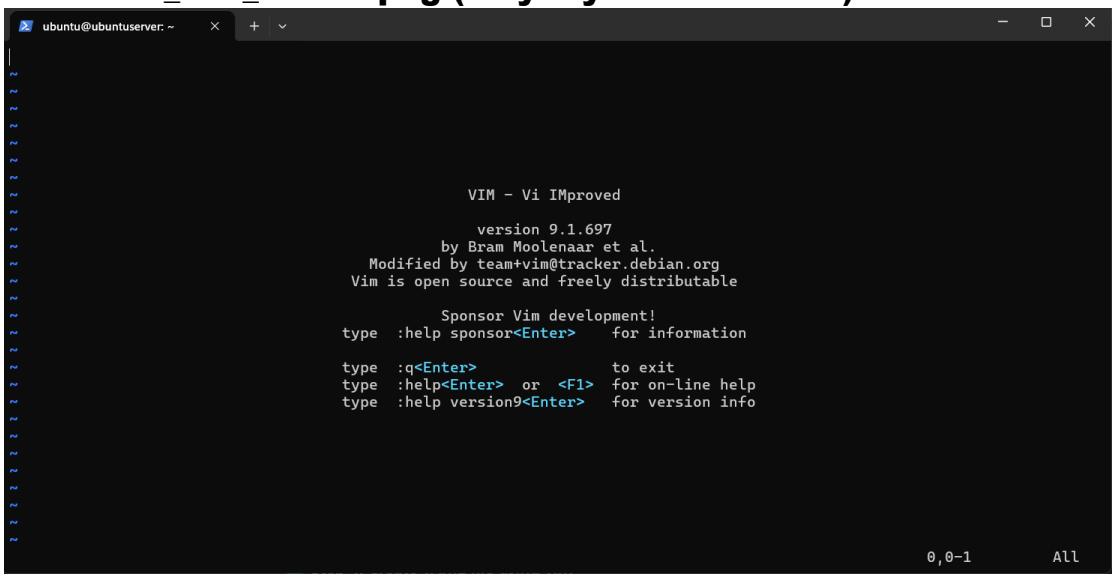
Save a screenshot of the result (either the vim splash screen or the error) as: task9_vim_check.png

If vim is not installed, install it:

sudo apt update

sudo apt install vim -y

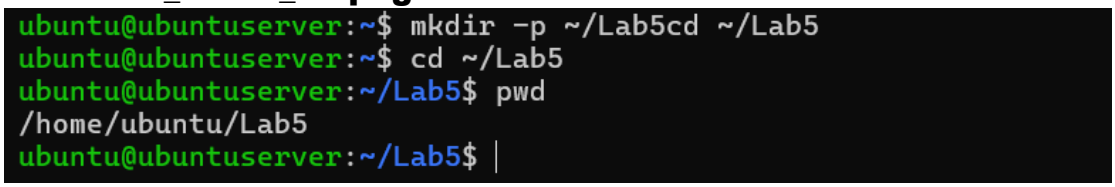
Save screenshot of the installation
as: task9_vim_install.png (only if you installed it).



2. Create the Lab5 working directory in your home and change into it:

mkdir -p ~/Lab5cd ~/Lab5

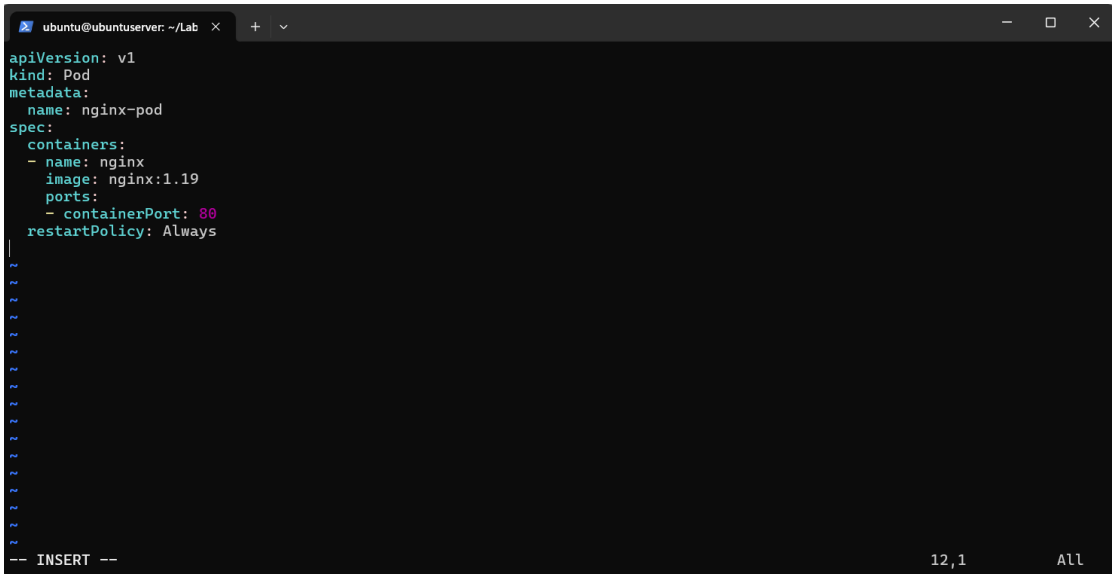
Save screenshot showing mkdir and cd or pwd output
as: task9_mkdir_cd.png



3. Create the Kubernetes sample file using vim:
vim k8s-sample.yaml
Once vim opens, enable insert mode by pressing i, then paste the following YAML exactly:

apiVersion: v1kind: Podmetadata:
  name: nginx-podspec:
  containers:
  - name: nginx
    image: nginx:1.19
    ports:
    - containerPort: 80
  restartPolicy: Always

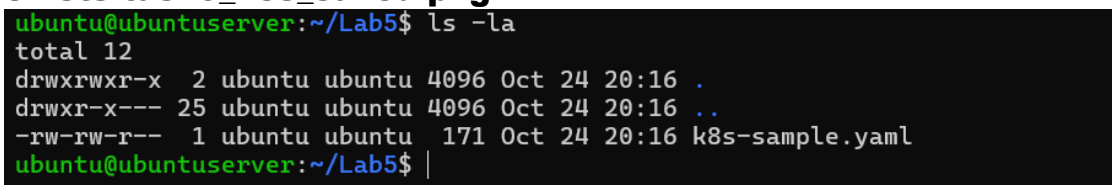
Save a screenshot of the vim editor showing the file contents before saving as task9_vim_edit.png



4. Exit insert mode by pressing Esc, then save and quit vim with:

:wq

Save a screenshot of the ls -la output showing the file exists task9_k8s_saved.png



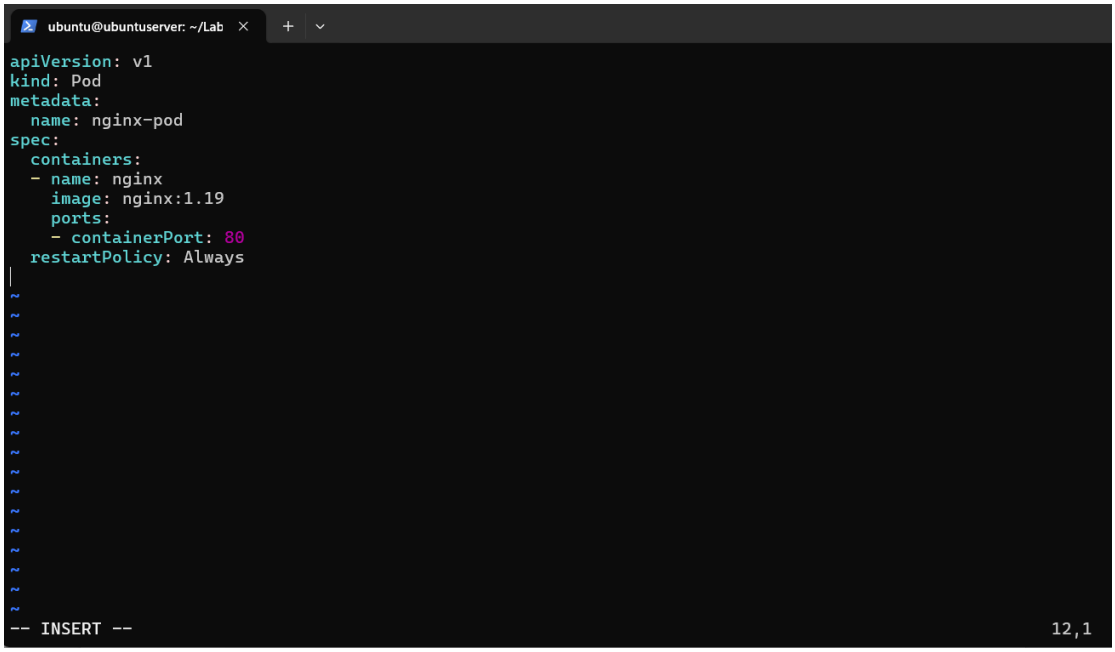
Task 10 - Edit the Kubernetes YAML - add annotation, verify, then discard temporary change

1. Open the manifest with vim:

cd ~/Lab5

vim k8s-sample.yaml

When vim opens, ensure you are in command mode (press Esc) then press i to enter insert mode.



2. Add the annotation under the metadata section (indentation must match YAML). Example (insert these lines under metadata:):

annotations:
  lab: lesson11

After adding the lines, press Esc to return to command mode and save & quit:

:wq

Save a screenshot showing the saved file contents (using cat) as: task10_verify_annotation.png

Example verify command:

cat k8s-sample.yaml

3. Discard changes (practice: make a temporary edit and exit without saving)

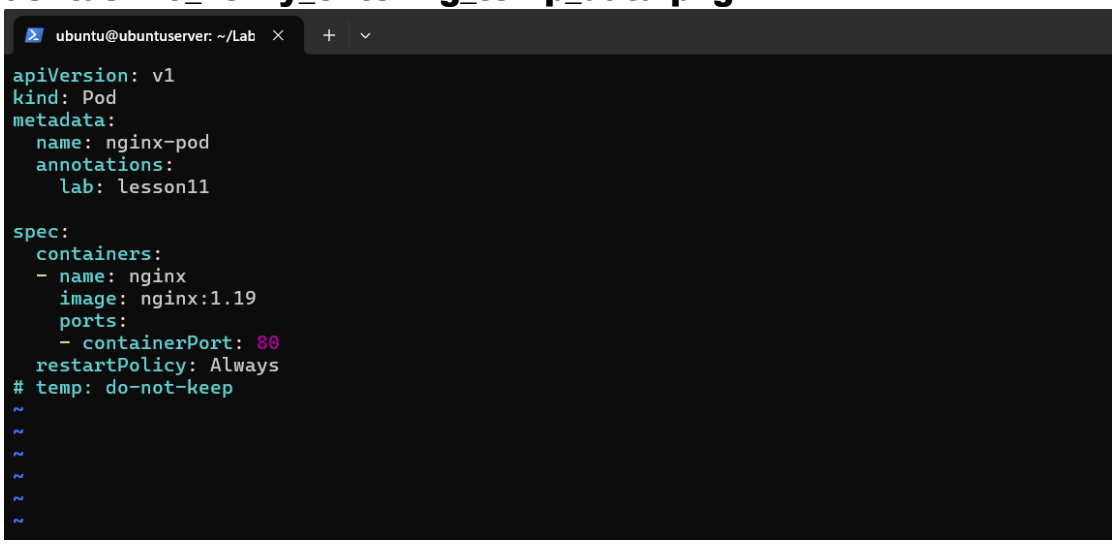
Open the file again:

vim k8s-sample.yaml

Enter insert mode (i) and add a temporary comment line anywhere, for example:

temp: do-not-keep

Save a screenshot showing vim editor having temp data as: task10_verify_entering_temp_data.png

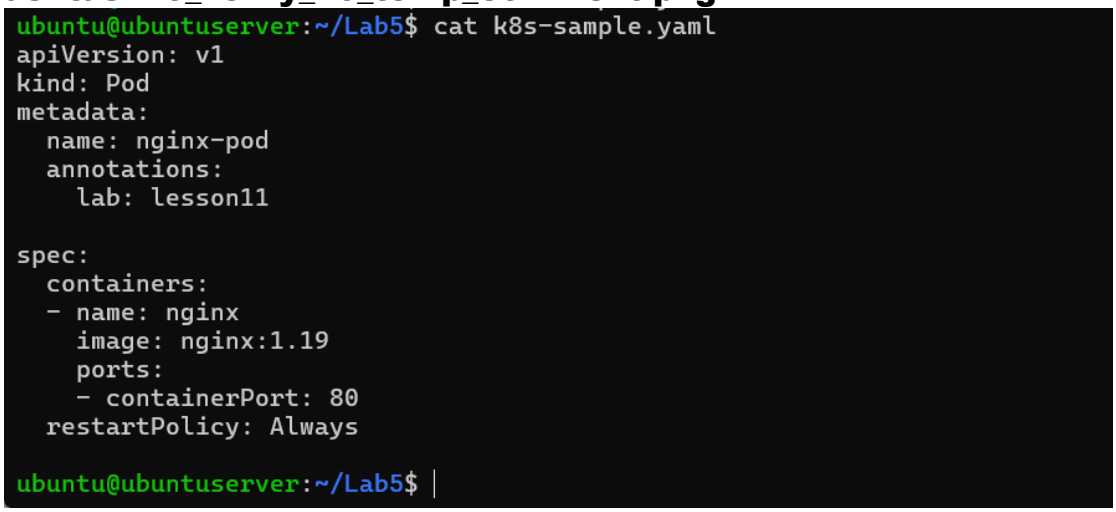


Do NOT save. Press Esc to go back to command mode, then force quit without saving:

:q!

Verify the file does NOT contain the temporary comment: cat k8s-sample.yaml

Save a screenshot of the cat output proving the temporary comment is not present as: task10_verify_no_temp_comment.png



Task 11 - Vim editing practice - delete, undo, numeric deletes, navigation

1. Open the file with vim:

cd ~/Lab5

vim k8s-sample.yaml

2. Delete the line containing image: nginx:1.19 with a single command:

Ensure you are in command mode (press Esc), move the cursor to the line with image: nginx:1.19, then delete that line with:

dd

Immediately undo the deletion with:

u

Capture a screenshot showing the file after the undo (or a cat output after saving) as: task11_dd_delete_and_undo.png

```
spec:
  containers:
  - name: nginx
    image: nginx:1.19
    ports:
    - containerPort: 80
  restartPolicy: Always

~
~
~
~
~
~
~
~
~
~
~
~
~
~
1 more line; before #1  1 second ago
```

3. Delete 3 lines at once using the numeric delete command:

In command mode, position the cursor on the first line of the three you want to delete (for example start at the image: line again), then run:

d3d

(or 3dd or d3<enter> — either numeric prefix form is acceptable; document which you used)

Immediately undo the deletion with:

u

Capture a screenshot showing the file after the undo (or a cat output) as: task11_delete3_and_undo.png

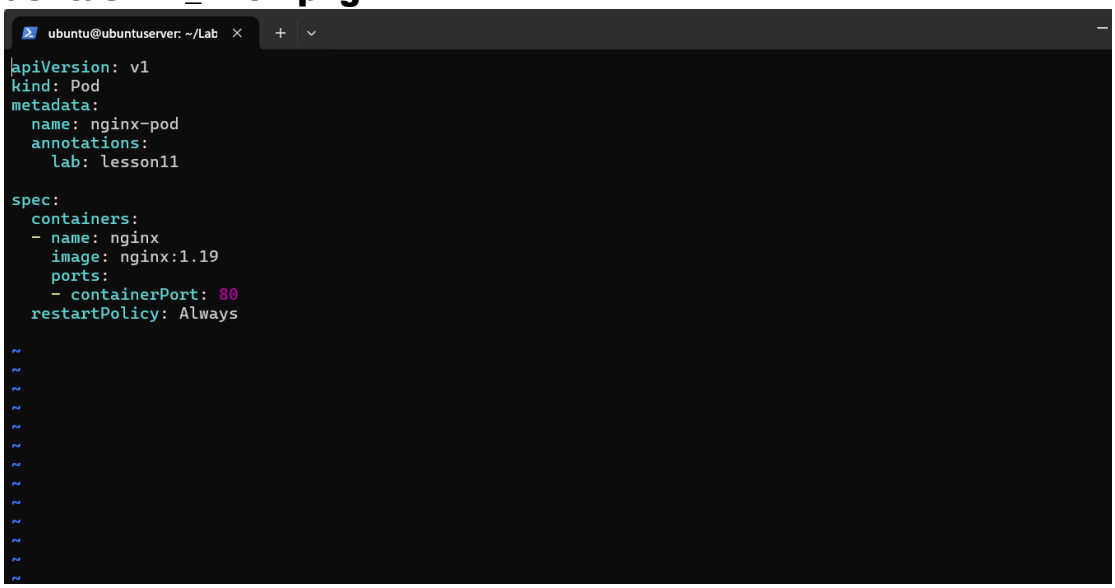
```
spec:
  containers:
  - name: nginx
    image: nginx:1.19
    ports:
    - containerPort: 80
  restartPolicy: Always

~
~
~
~
~
~
~
~
~
~
~
~
~
~
3 more lines; before #2  1 second ago
```

4. Navigation practice (from command mode)

Jump to the first line:

1G

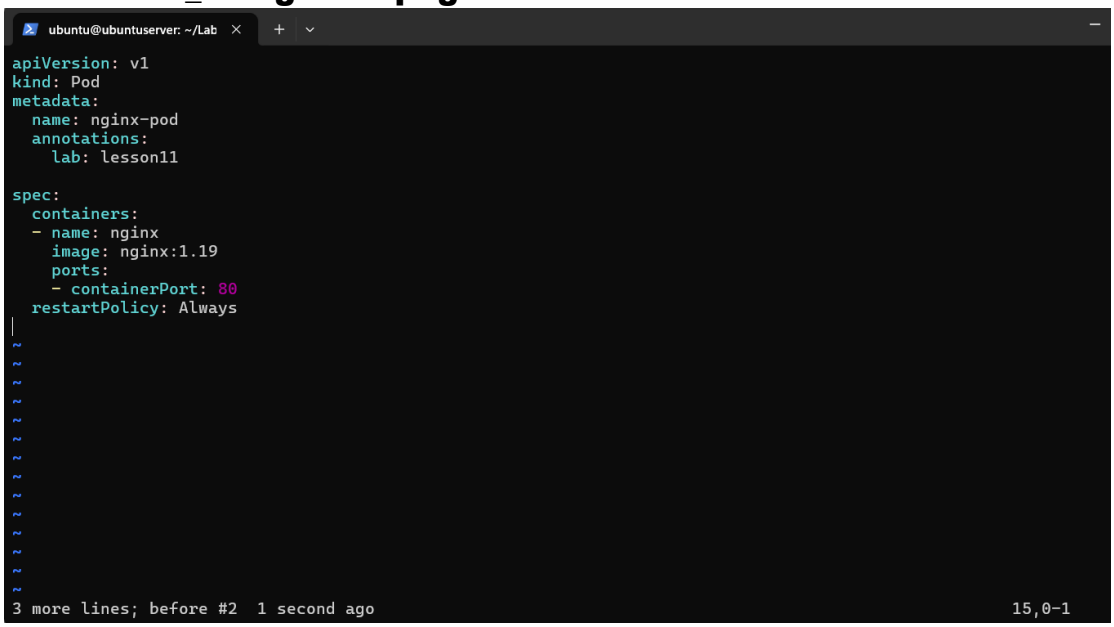
Note the line contents capture a screenshot saved as: task11_line1.png



Jump to the last line:

G

On the line containing containerPort: 80 press $ to move to end of line, then 0 to move back to the start of the line. Capture a brief terminal screenshot showing you on the line and the commands capture the screenshot as: task11_navigation.png

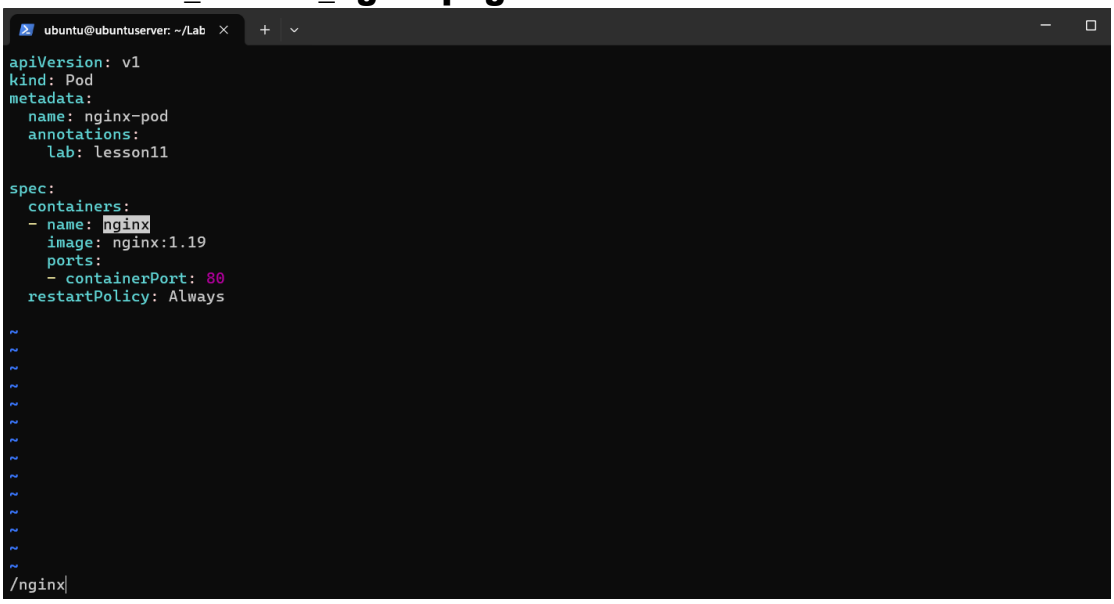


5. Exit vim (no changes should remain if you undid properly):

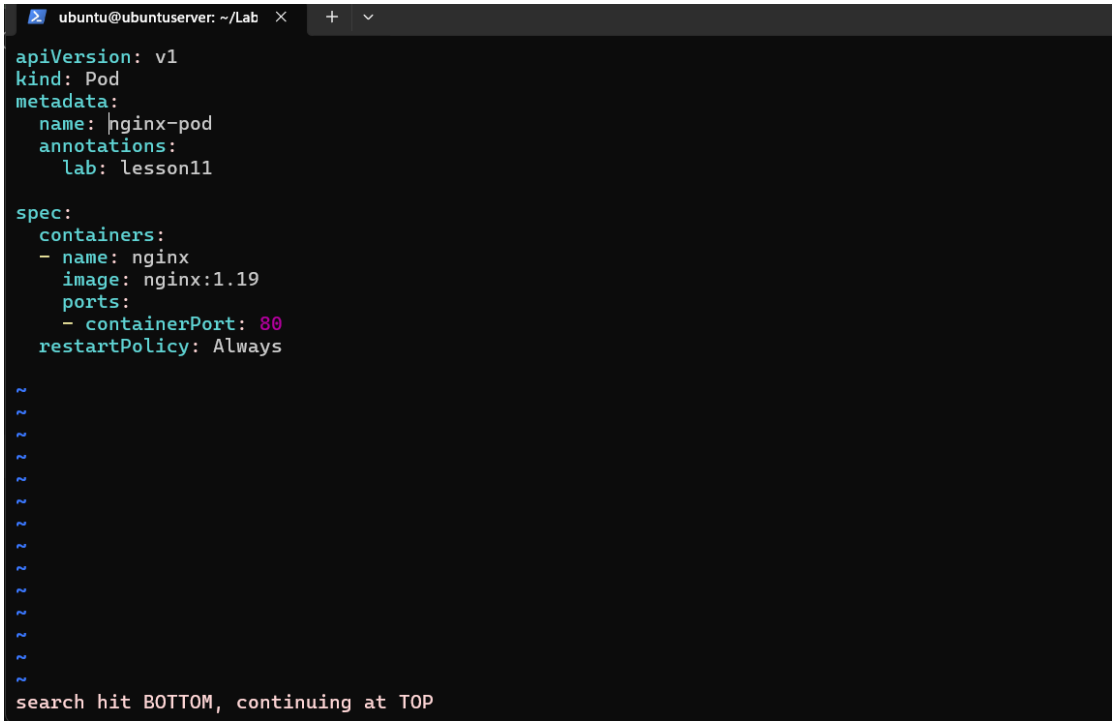
:q

Task 12 - Vim search, add matches, substitute, undo

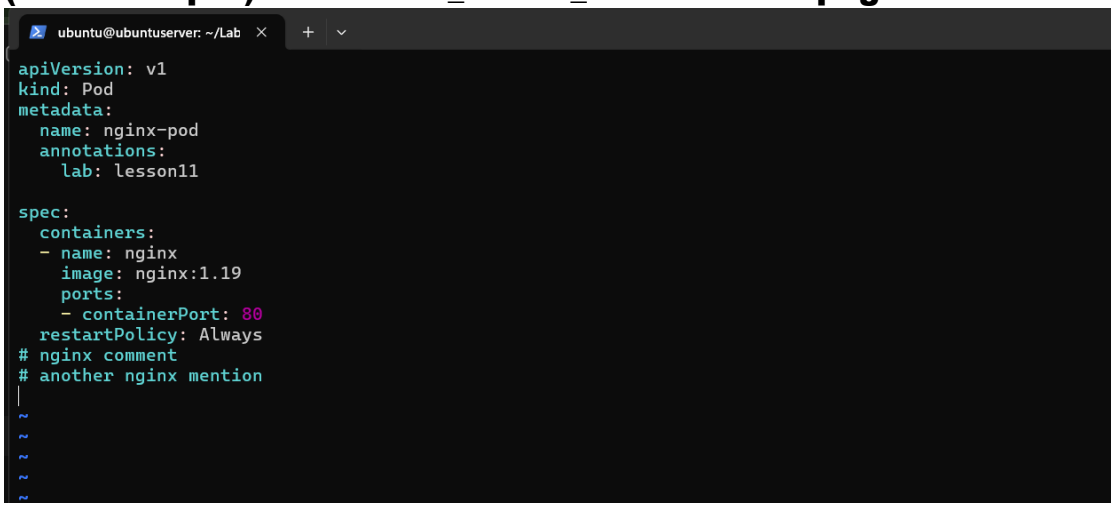
1. Open the file with vim:
cd ~/Lab5
vim k8s-sample.yaml
2. Search for the string nginx using the forward search command:
From command mode type:
/nginx
and press Enter.
Note the first match is highlighted and the cursor is placed on it. Save a screenshot showing the first match in vim as: task12_search_nginx.png



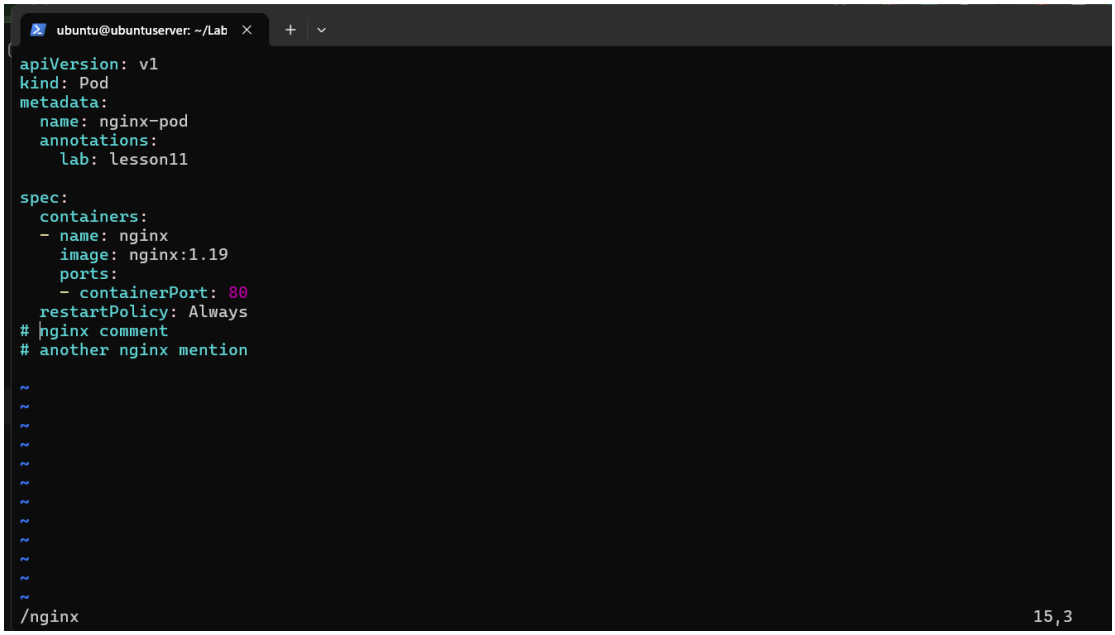
3. Move to the next match and previous match:
Press n to jump to the next match (capture screenshot if desired) and press N to jump back to the previous match. Save a screenshot showing navigation between matches as: task12_n_and_N_navigation.png



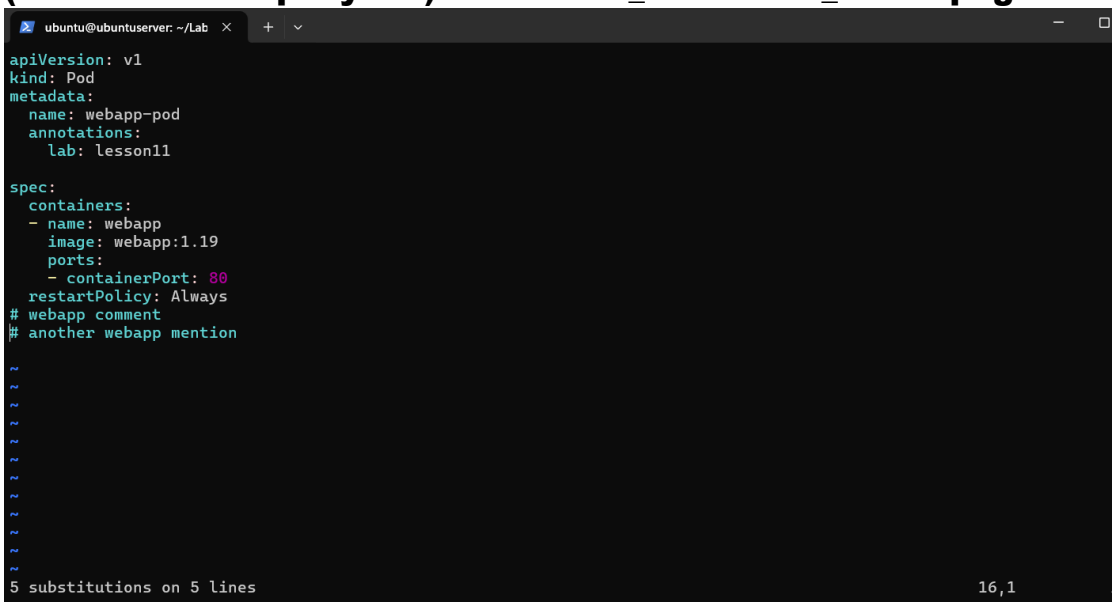
4. Add two more occurrences of the word nginx in the file:
Enter insert mode (i) at an appropriate place (for example add two new comment lines or add them under metadata as comments), type two new occurrences of nginx, then press Esc to return to command mode and save the file with :w.
Save a screenshot showing the added occurrences (or cat output) as: task12_added_occurrences.png



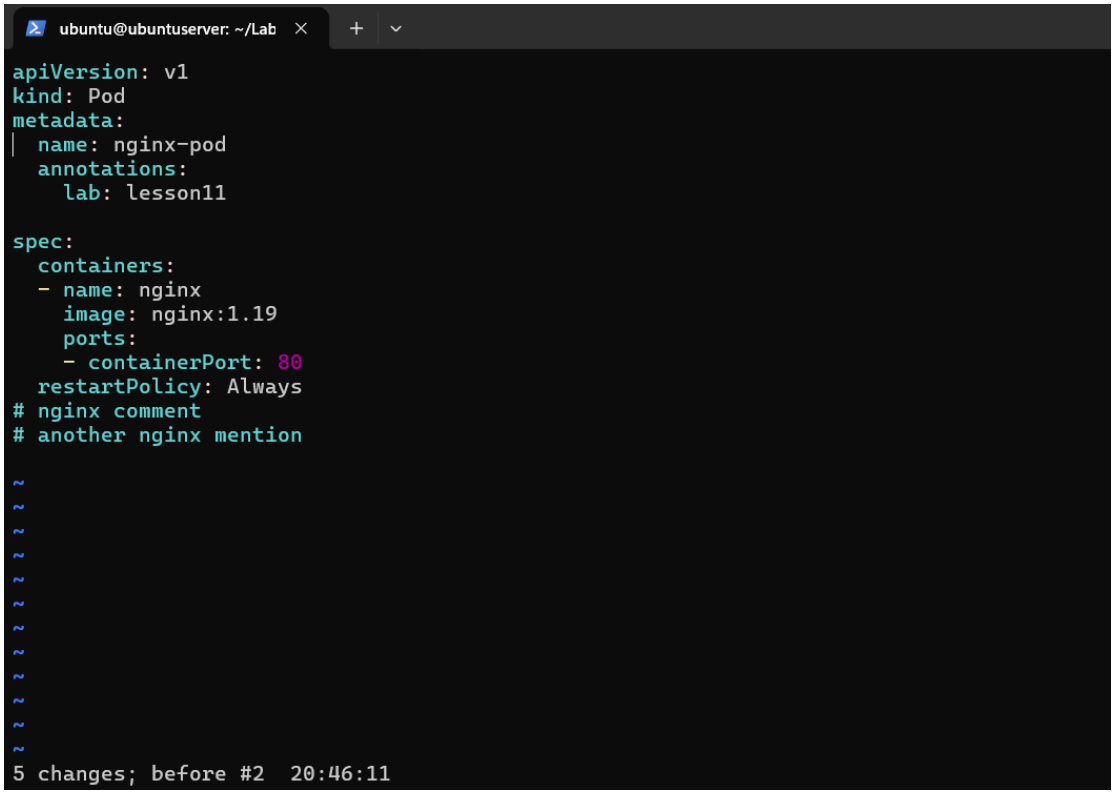
5. Demonstrate that n cycles forward through all matches:
In vim command mode press /nginx Enter, then repeatedly press n to cycle forward through each match. Capture a short sequence (or a terminal screenshot showing the cursor on a later match) as: task12_cycle_matches.png



6. Substitute all occurrences of nginx with webapp:
From command mode execute the substitute command:
:%s/nginx/webapp
This will replace all occurrences in the file (note: this changes the buffer). Save a screenshot showing the substitution result (or cat k8s-sample.yaml) as: task12_substitute_result.png



7. Immediately undo the substitution using u:
Press u in command mode to undo the last change. Save a screenshot showing the file restored to the previous state as: task12_undo_and_quit.png



Quit vim without saving any accidental changes (if you didn't already save after substitution and want to discard):
If you want to ensure no changes were written, exit with:
:q!
If you intentionally saved the substitution and then undid it and wish to quit normally, use :q.

Exam evaluation questions:

Goal: This is an exam-style evaluation prompt. Students are asked to install Docker Desktop as part of the evaluation exercise on VM. No commands, solutions, hints, or step-by-step instructions are provided here — install Docker Desktop using your own knowledge and research.
Steps:

1. Install Docker Engine

```
ubuntu@ubuntuserver:~$ sudo apt update
sudo apt install ca-certificates curl gnupg lsb-release -y
Hit:1 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://dl.google.com/linux/chrome/deb stable InRelease
Hit:5 https://ppa.launchpadcontent.net/obsproject/obs-studio/ubuntu noble InRelease
Hit:6 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:7 https://ppa.launchpadcontent.net/ubuntuhandbook1/audacity/ubuntu noble InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203).
ca-certificates set to manually installed.
curl is already the newest version (8.5.0-2ubuntu10.6).
curl set to manually installed.
gnupg is already the newest version (2.4.4-2ubuntu17.3).
gnupg set to manually installed.
lsb-release is already the newest version (12.0-2).
lsb-release set to manually installed.
The following packages were automatically installed and are no longer required:
  ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java libatk-wrapper-java-jni libgif7
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

2. Add Docker's official GPG key

```
ubuntu@ubuntuserver:~$ sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | \
  sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

3. Add Docker repo

```
ubuntu@ubuntuserver:~$ echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] \
  https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

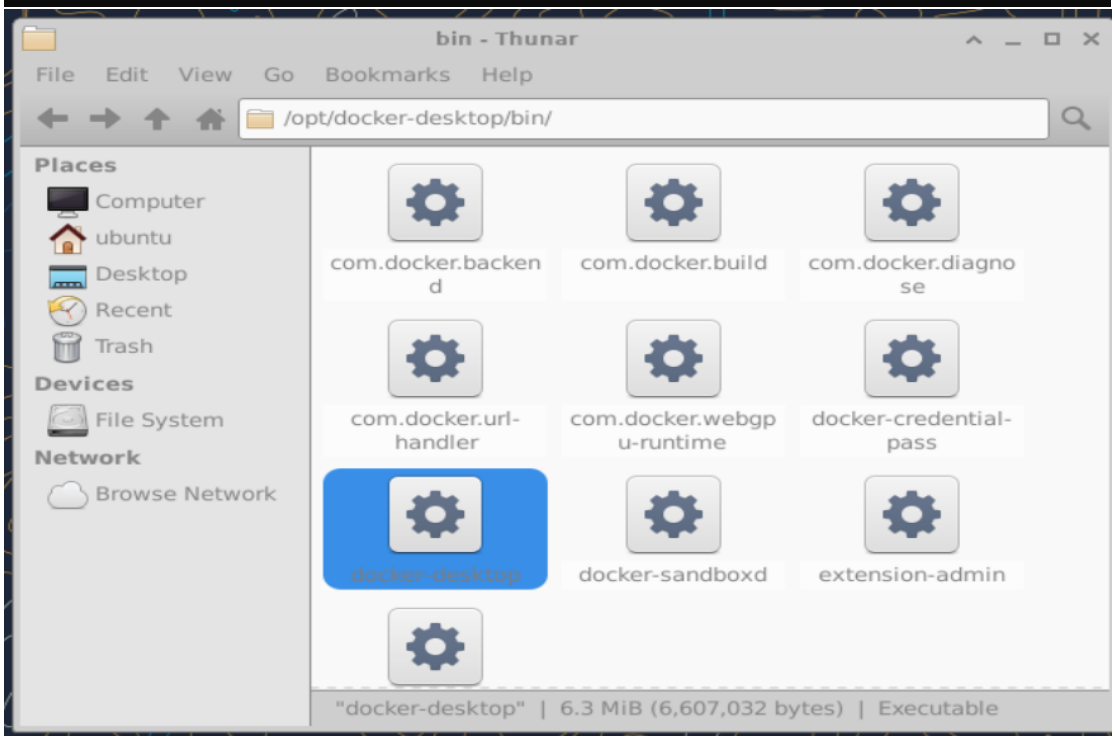
4. Install Docker Engine

```
ubuntu@ubuntuserver:~$ sudo apt update
sudo apt install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y
Hit:1 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:2 https://download.docker.com/linux/ubuntu noble InRelease [48.5 kB]
Get:3 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [33.3 kB]
Hit:4 http://dl.google.com/linux/chrome/deb stable InRelease
Hit:5 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:6 https://ppa.launchpadcontent.net/obsproject/obs-studio/ubuntu noble InRelease
Hit:7 https://ppa.launchpadcontent.net/ubuntuhandbook1/audacity/ubuntu noble InRelease
Hit:8 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:9 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Fetched 81.7 kB in 2s (45.1 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java libatk-wrapper-java-jni libgif7
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  docker-ce-rootless-extras libslirp0 pigz slirp4netns
Suggested packages:
```

```
ubuntu@ubuntuserver:~$ sudo apt-get install ./docker-desktop-amd64.deb
Reading package lists... Done
E: Unsupported file ./docker-desktop-amd64.deb given on commandline
ubuntu@ubuntuserver:~$  sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-10-24 21:10:41 UTC; 10min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 11037 (dockerd)
      Tasks: 11
     Memory: 26.0M (peak: 29.8M)
        CPU: 1.068s
     CGroup: /system.slice/docker.service
             └─11037 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Oct 24 21:10:41 ubuntuserver dockerd[11037]: time="2025-10-24T21:10:41.079463794Z" level=info msg="Creating a container>
Oct 24 21:10:41 ubuntuserver dockerd[11037]: time="2025-10-24T21:10:41.131258582Z" level=info msg="Loading containers: >
Oct 24 21:10:41 ubuntuserver dockerd[11037]: time="2025-10-24T21:10:41.764146167Z" level=info msg="Loading containers: >
Oct 24 21:10:41 ubuntuserver dockerd[11037]: time="2025-10-24T21:10:41.789181041Z" level=info msg="Docker daemon" commi>
Oct 24 21:10:41 ubuntuserver dockerd[11037]: time="2025-10-24T21:10:41.789351267Z" level=info msg="Initializing buildki>
Oct 24 21:10:41 ubuntuserver dockerd[11037]: time="2025-10-24T21:10:41.827268946Z" level=info msg="Completed buildkit i>
Oct 24 21:10:41 ubuntuserver dockerd[11037]: time="2025-10-24T21:10:41.833783029Z" level=info msg="Daemon has completed>
Oct 24 21:10:41 ubuntuserver dockerd[11037]: time="2025-10-24T21:10:41.833870621Z" level=info msg="API listen on /run/d>
Oct 24 21:10:41 ubuntuserver systemd[1]: Started docker.service - Docker Application Container Engine.
Oct 24 21:11:04 ubuntuserver dockerd[11037]: time="2025-10-24T21:11:04.659097930Z" level=info msg="ignoring event" cont>
```

To see if its working:

```
ubuntu@ubuntuserver:~$ sudo docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```



Installation done: