Week 5 – Operating Systems

Student number: 585732

Assignment 5.1: Unix-like

a) Find out what the difference is between UNIX and unix-like operating systems?

= UNIX and unix-like systems are related, but they're not the same thing.

- UNIX (in capital letters) is a **brand name and a certified standard**.
The name is owned by The Open Group, and only operating systems that pass their official tests are allowed to call themselves "UNIX".

Examples:
- modern macOS
- IBM AIX
- HP-UX

- unix-like systems are operating systems that work in a similar way to UNIX, follow similar standards, and feel like UNIX to the user, but they are not officially certified as UNIX.

Examples:
- Linux distributions (Ubuntu, Debian, Fedora, etc.)
- FreeBSD, OpenBSD, NetBSD
- Android and ChromeOS (built on the Linux kernel)

In short:
UNIX = officially certified and allowed to use the name.
unix-like = similar in design and behaviour, but not officially UNIX.

b) Study the image above named UNIX timeline. Find out who Ken Thompson, Dennis Ritchie, Bill Joy, Richard Stallman, and Linus Torvalds are and what they have contributed to the development of UNIX or unix-like systems and to IT in general. **TIP!** English-language sources often contain more detailed information about these individuals.

= **Ken Thompson** – Co-creator of UNIX at Bell Labs. Wrote the first UNIX kernel and several core tools; also created the B language and introduced key UNIX ideas like simple programs connected with pipes.

Dennis Ritchie – Co-creator of UNIX and inventor of the C programming language. Rewriting UNIX in C made it portable and hugely influenced modern operating systems and software.

Bill Joy – Major developer of BSD UNIX at UC Berkeley. Wrote the vi editor and the C shell, and later co-founded Sun Microsystems. His BSD work strongly influenced networking and later unix-like systems.

Richard Stallman – Founder of the GNU Project and the Free Software Foundation. Created tools like GNU Emacs and GCC and wrote the GPL license. He launched the free software movement and aimed to build a free UNIX-like system.

Linus Torvalds – Creator of the Linux kernel, started in 1991. Linux, combined with GNU tools, powers many modern systems (servers, phones, supercomputers). He also created Git, widely used in software development.

c) What is the philosophy of the GNU movement?

= The philosophy of the GNU movement is centered on software freedom – the idea that users should have control over the software they use rather than being restricted by proprietary licenses.

In this philosophy, "free software" means that the software grants these four essential freedoms:

1. Freedom to run the program for any purpose.
2. Freedom to study how the program works and modify it (access to source code is required).
3. Freedom to redistribute copies of the program.
4. Freedom to distribute modified versions so that others can benefit from the changes.

To protect these freedoms, the GNU project uses copyleft licenses such as the GNU General Public License (GPL), which ensure that modified versions that are shared must also remain free software. The overall goal is a completely free operating system and software ecosystem that respects these user freedoms.

d) Does Ubuntu as a Linux operating system conform to the philosophy of the GNU movement? Please explain your answer.

= Ubuntu follows the philosophy of the GNU movement only partially.

On one hand, Ubuntu is built on GNU/Linux and includes a large amount of free software: the Linux kernel, GNU tools (like coreutils, GCC, bash), and many applications under free licenses such as GPL, MIT, and BSD. These components respect the GNU idea of software freedom: users can study, modify, and share much of the system's code.

On the other hand, Ubuntu also includes and promotes non-free (proprietary) software. This applies in particular to some hardware drivers and firmware (e.g. for graphics cards or Wi-Fi), as well as certain codecs and optional closed-source applications. Because of this, the Free Software Foundation does not classify Ubuntu as a fully free distribution and instead recommends distros that remove all proprietary components (such as Trisquel).

In conclusion, Ubuntu supports many goals of the GNU movement but does not fully conform to its strict philosophy, because it still allows and distributes non-free software for practical and compatibility reasons.

e) Find out what is the Windows Subsystem for Linux?

= The Windows Subsystem for Linux (WSL) is a feature in Windows 10 and Windows 11 that allows Linux to run directly on Windows.

It provides a Linux environment inside Windows, so Linux distributions such as Ubuntu or Debian can be installed from the Microsoft Store. Through WSL, Linux command-line tools and applications run alongside normal Windows programs, without needing a separate virtual machine or dual-boot setup.

There are two main versions:

- WSL 1: uses a compatibility layer that translates Linux system calls into Windows system calls.
- WSL 2: runs a real Linux kernel in a lightweight virtual machine, offering better compatibility and performance.

f) Find out, which operating system family belongs to Android, iOS and ChromeOS?

= **Android**
Android is based on the Linux kernel, so it belongs to the Linux / Unix-like operating system family.

iOS
iOS is built on Darwin, which uses the XNU kernel and includes code from BSD UNIX. It belongs to the Unix / Unix-like family.

ChromeOS
ChromeOS is also built on the Linux kernel (with Gentoo Linux as an early base), so it belongs to the Linux / Unix-like family.

Android, iOS and ChromeOS all belong to the broad Unix-like family of operating systems. Android and ChromeOS are Linux-based, while iOS is based on a BSD/UNIX core.

Assignment 5.2: Supercomputers and gameconsoles

a) Research on this site what supercomputers are used for and write a short summary of it: https://www.computerhistory.org/timeline/search/?q=Supercomputer

= Supercomputers are extremely powerful machines built to handle problems that are too big and complex for ordinary computers. They are mainly used for tasks that involve huge amounts of calculations and data, often in science, engineering, and national security.

Typical uses include things like weather forecasting and climate modelling, where supercomputers simulate the atmosphere and oceans to predict storms or study climate change. They are also used in physics and engineering to model things such as airflow over aircraft wings, car crash simulations, nuclear reactions, and the behaviour of materials under extreme conditions. Government and military organizations use them for code-breaking and for simulating nuclear weapons tests instead of carrying out real explosions. In addition, supercomputers are used in areas like astronomy (for modelling galaxies and black holes), biology and medicine (for protein folding, genomics, and drug discovery), and more recently for large-scale data analysis and advanced AI and machine learning.

Overall, the main point is that supercomputers let researchers and engineers run extremely detailed simulations and calculations that would be impossible or take far too long on normal computers.

b) IBM is a company that has already built a number of supercomputers. One of them is IBM's Roadrunner. The CPU developed for this supercomputer was further developed at a later stage as the CPU for the PlayStation 3 console. Find out what a **PlayStation 3 cluster** is and what it was used for?

= A PlayStation 3 cluster is basically a bunch of PS3 consoles linked together and used as one big, powerful computer.

The reason this works is that the PS3 uses the Cell processor, which is related to the one used in IBM's Roadrunner supercomputer. For a while, you could install Linux on a PS3, so people connected many PS3s on a network, ran Linux on them, and treated them like a cheap supercomputer.

What it was used for:

Scientific research
Universities and research labs used PS3 clusters to run heavy calculations, for example in physics and astronomy (like simulating black holes or other complex models), because the Cell processor was very good at number-crunching.

Military and government projects
The U.S. Air Force, for example, built a large PS3 cluster (often called the Condor Cluster) with over a thousand PS3s. They used it for things like processing high-resolution images and video, pattern recognition, and other tasks that need a lot of parallel computing power.

Overall, a PlayStation 3 cluster is just many PS3s working together as a low-cost supercomputer for serious scientific and technical work.

c) You can build a supercomputer by putting a few computers together in a cluster. Here's what Oracle did with a collection of Raspberry Pi's, for example:
https://blogs.oracle.com/developers/post/building-the-worlds-largest-raspberry-pi-cluster
What specific operating system is running on this cluster?

= They're running Oracle Linux 7 for ARM on the Raspberry Pi cluster.

d) Does Oracle's Raspberry Pi supercomputer appear in the list of the 500 fastest supercomputers in the world? Make a logical decision for this, without going through the entire list.
https://www.top500.org/lists/top500/list/2023/06/

= No, Oracle's Raspberry Pi "supercomputer" won't be in the TOP500 list.

The TOP500 only includes the fastest supercomputers in the world, and today those machines reach petaflop performance (and above) – that's millions of billions of calculations per second.

A Raspberry Pi is a low-power mini computer meant for learning and small projects, not raw speed. Even if you connect a lot of them together, their combined performance is still far below what TOP500 systems achieve.

On top of that, Oracle's Raspberry Pi cluster is clearly more of a demonstration and educational project than a serious attempt to build a top-ranked supercomputer.

So, logically, it's not on the list of the 500 fastest supercomputers.

e) What CPU architecture is used for the PlayStation 5 and Xbox Series X?
What operating systems run on these consoles?
What conclusion can you draw from the answer to the previous question?

= 1. CPU architecture in PS5 and Xbox Series X

- **PlayStation 5:** Uses a custom AMD Zen 2 processor, 8 cores, based on 64-bit x86-64 architecture.

- **Xbox Series X:** Also uses a custom AMD Zen 2 processor, 8 cores, also 64-bit x86-64.

So both consoles use very similar PC-style 64-bit x86 CPUs from AMD.

2. Operating systems on these consoles

- **PlayStation 5:** Runs a proprietary Sony OS that is based on FreeBSD, which is a Unix-like operating system.
- **Xbox Series X:** Runs a customized version of Microsoft's Windows NT (basically a special Xbox-tuned version of Windows).

3. Conclusion

Both modern consoles:

- Use PC-like x86-64 CPUs, and
- Run OSes closely related to desktop/server systems (Unix-like for PS5, Windows NT for Xbox).

The clear conclusion is that today's consoles are essentially specialized, locked-down PCs, which makes it much easier to develop and port games between PC, PlayStation, and Xbox.

Assignment 5.3: Working with Windows
Take relevant screenshots of the assignments below

a) Practice for about 10 minutes with the ⊞ keyboard shortcuts combinations, skip the general shortcuts in this exercise. Take a look at which screens are opened.

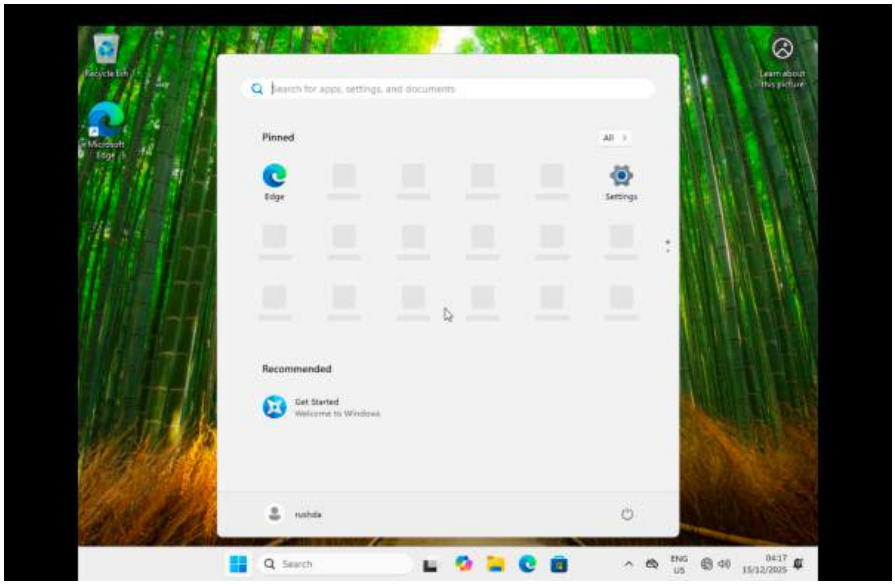


Fig 1: Command = Start Menu

Fig 2: Command + B

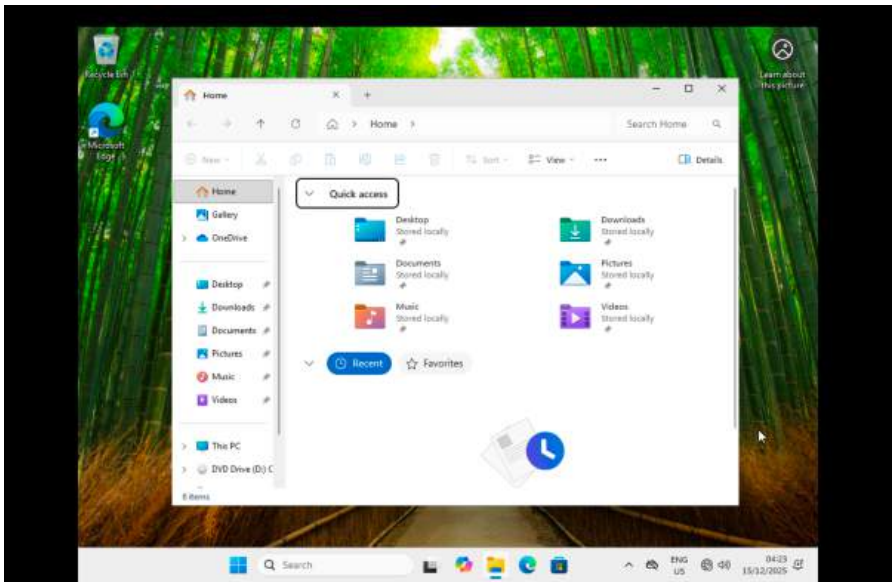


Fig 3: Command + E

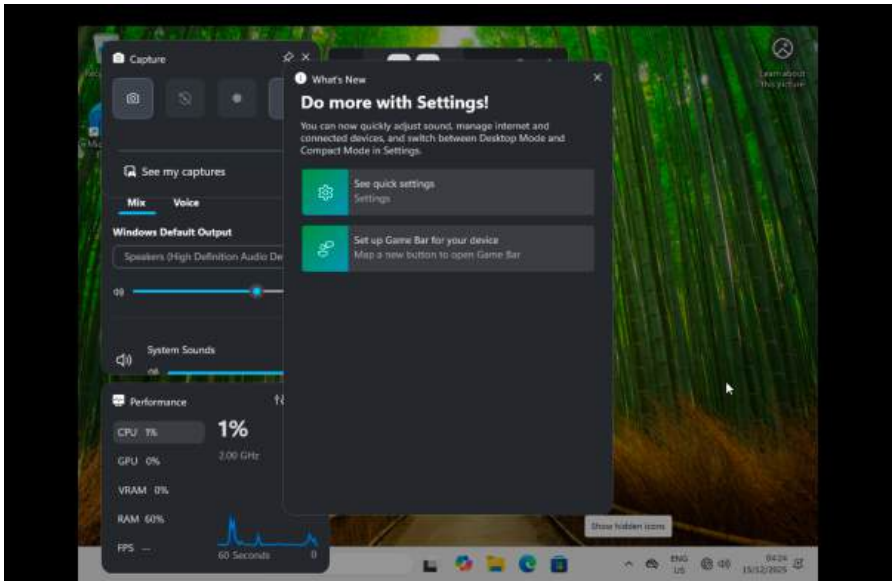


Fig 4: Command + G

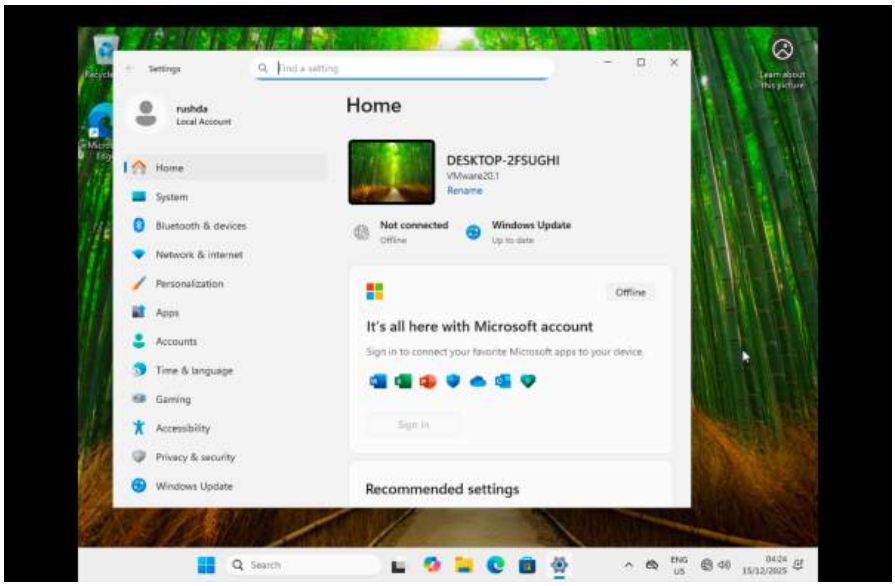


Fig 5: Command + I

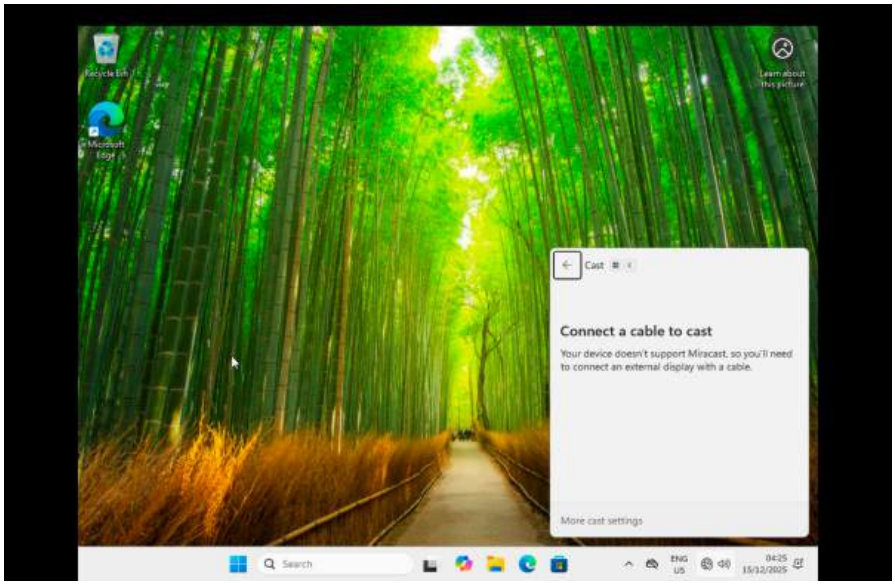


Fig 6: Command + K

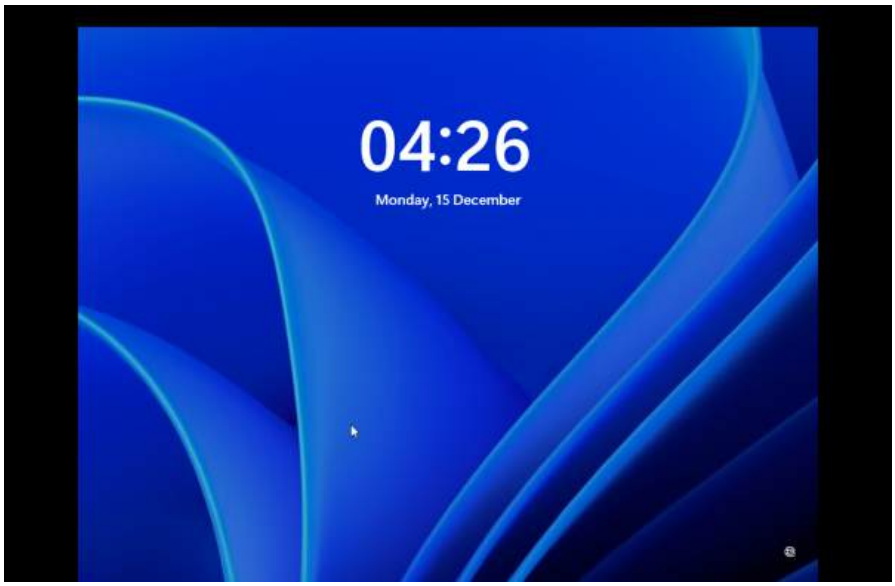


Fig 7: Command + L

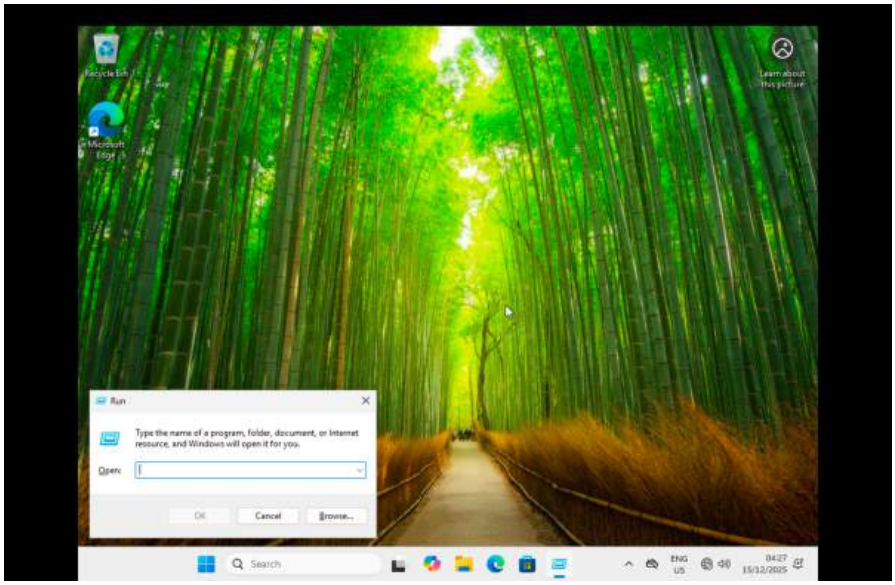
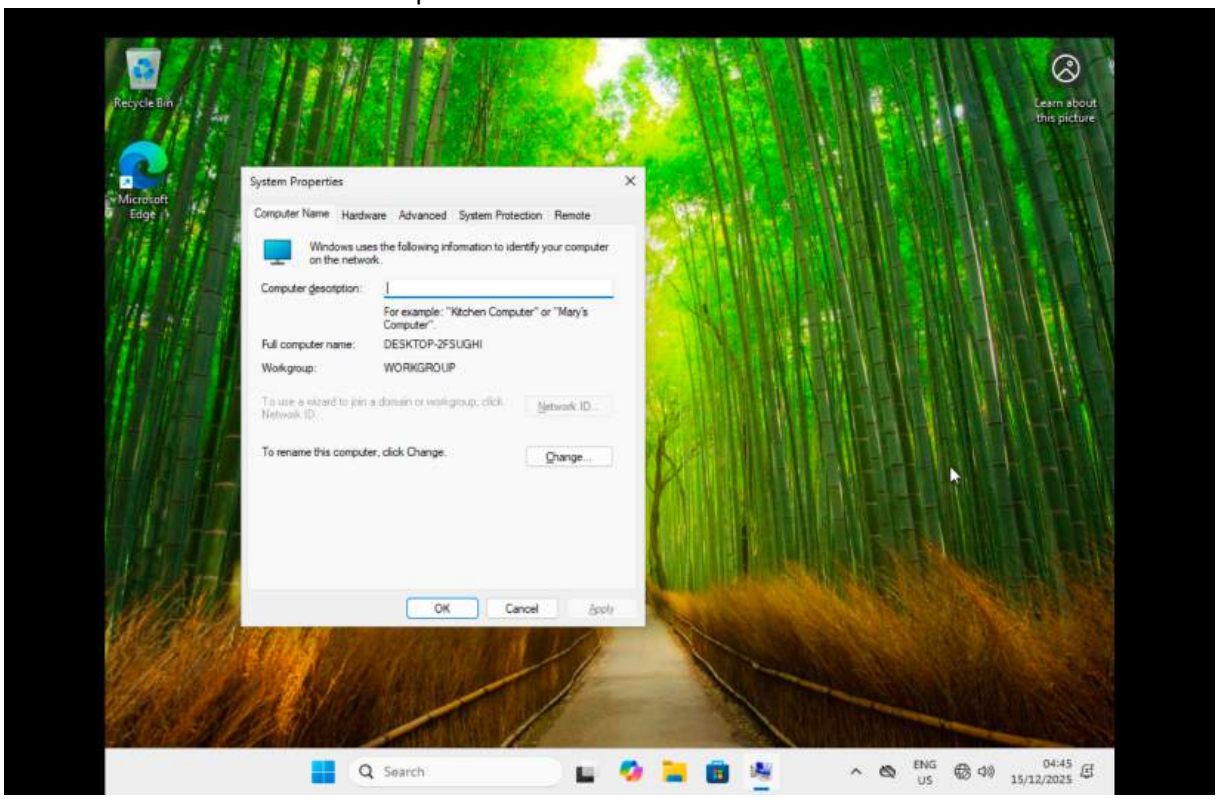


Fig 8: Command + R

b) The file explorer can be opened with ⊞ + E, Which key combination could you also use?
=Command+X+E (All these 3 keys at the same time).

c) Open the system properties with a ⊞ key combination, take a screenshot of the open screen. Paste this screenshot into this template.



There is no pause button in my mac, so I pressed Command + R and then typed sysdm.cpl and pressed enter.

d) Open task manager with a key combination. Take screenshots of the tabs: processes (shows active processes), performance, and users. Place these three screenshots in this template.

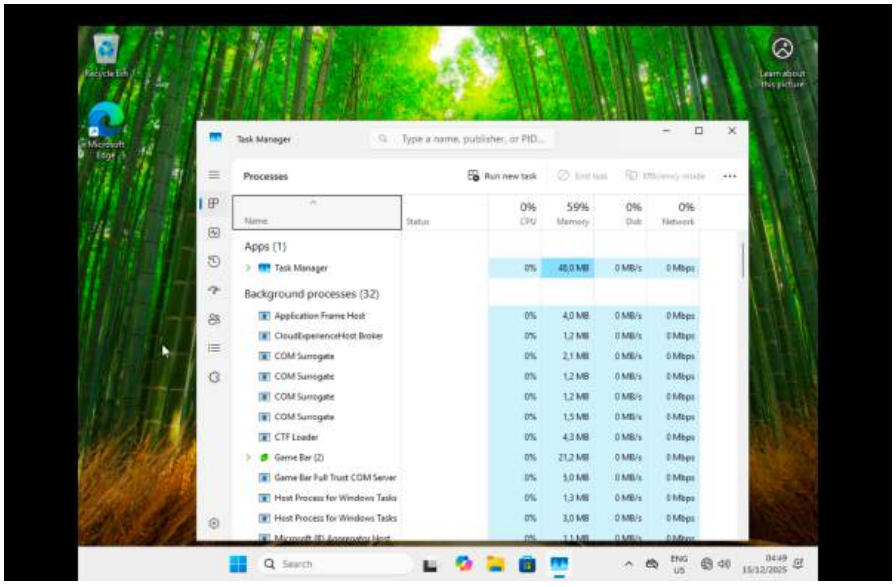


Fig 1: Processes

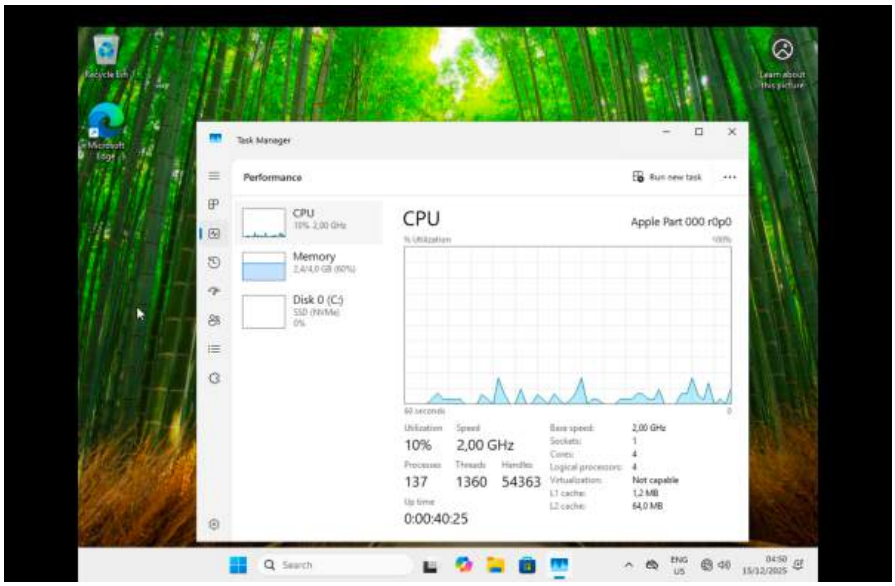


Fig 2: Performance

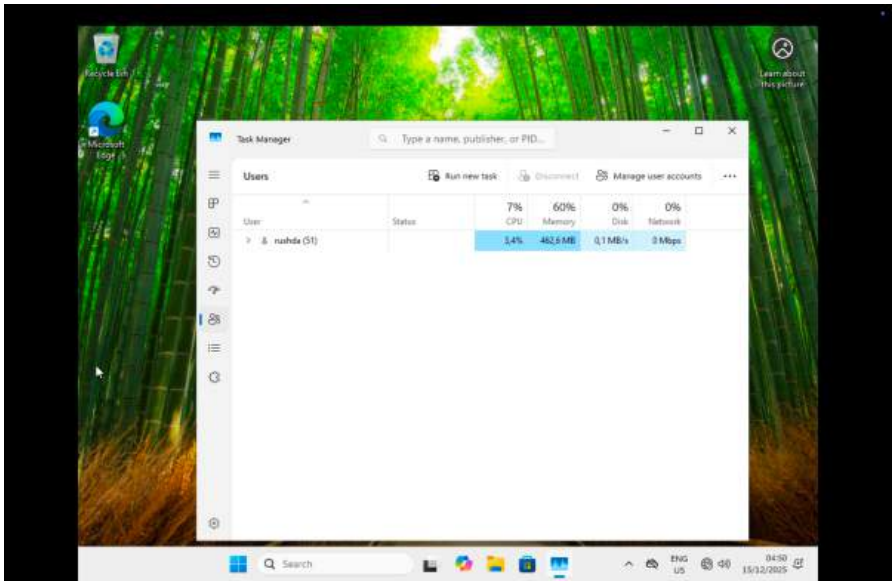


Fig 3: Users

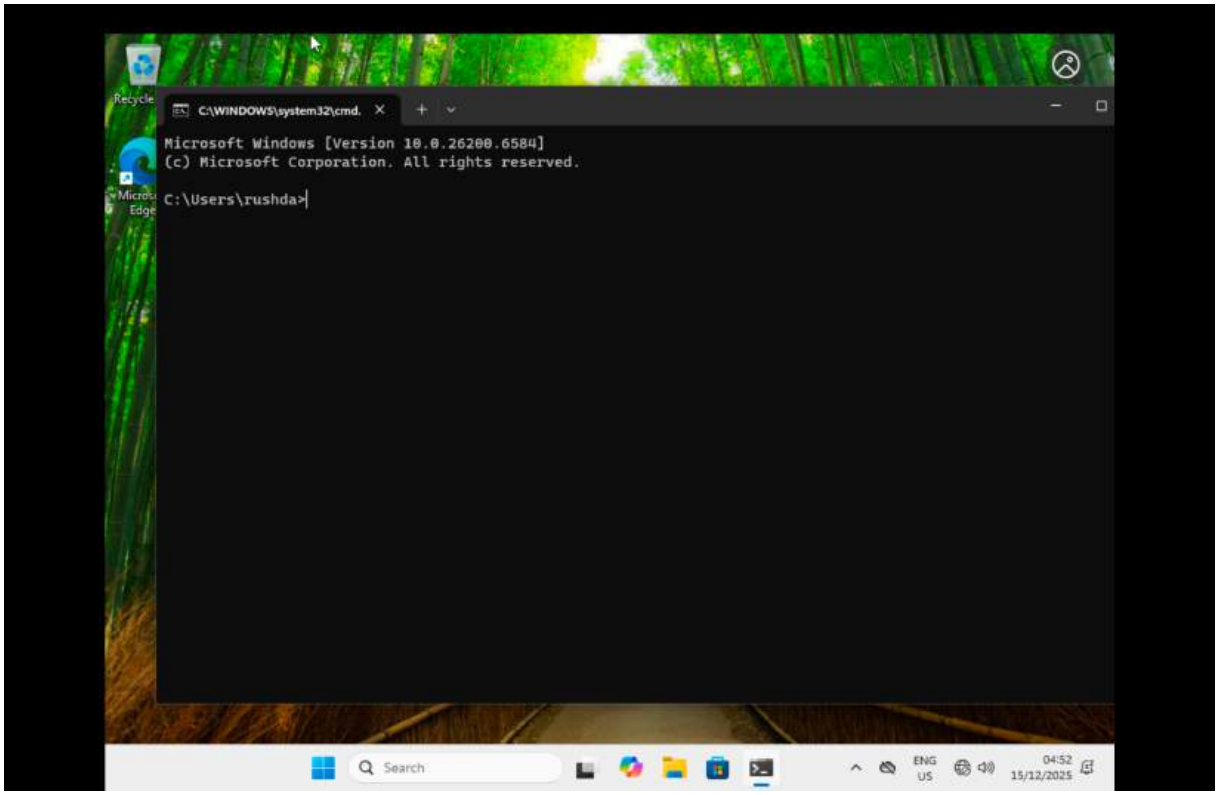
e) If you're giving a PowerPoint presentation and you connect your laptop to a projector, Windows can use the projector as a second screen. For example, you may have Outlook open on your first screen that you don't show over the projector, while the PowerPoint presentation is displayed on the projector, or the second screen. Which key combination should you use for this?

=Command + P

f) If you leave the classroom for a while and you leave your laptop behind, it is wise to lock the screen. Your Apps will continue to run in the background. So, for example, if you're waiting for a download that takes a while, lock the screen and get a cup of coffee. Which key combination do you use for this?

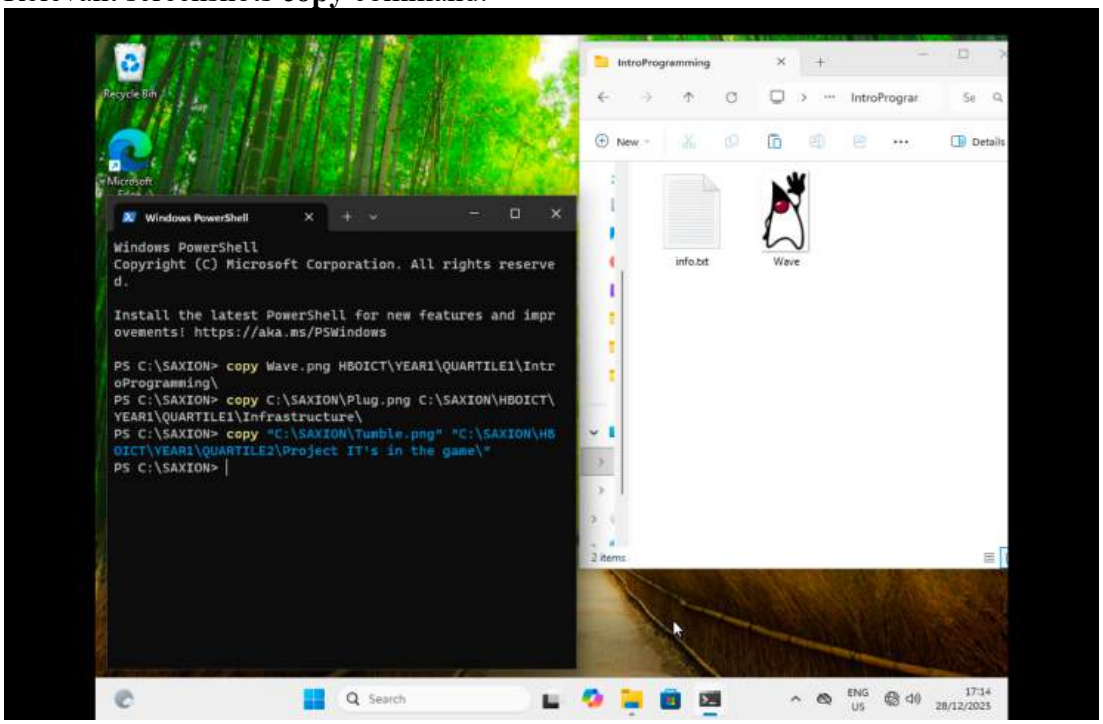
= Command + L

g) Open the Run screen with a key combination. On this screen, type CMD and press <enter>. Take a screenshot of this result and paste it into this template.

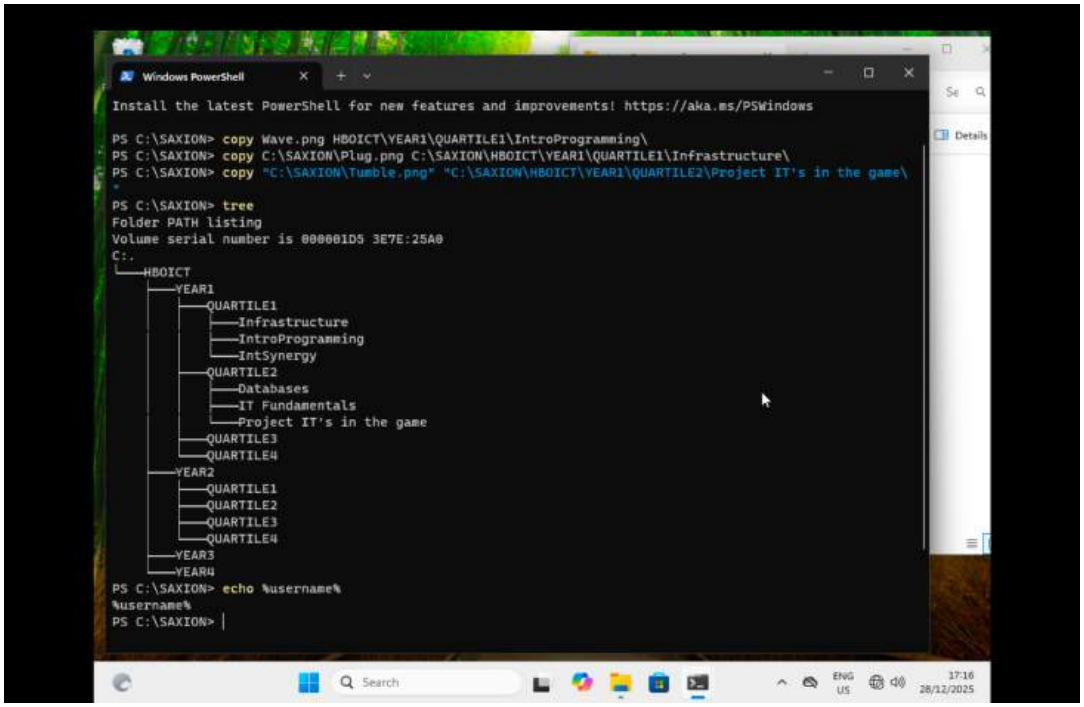


Working in the File Explorer

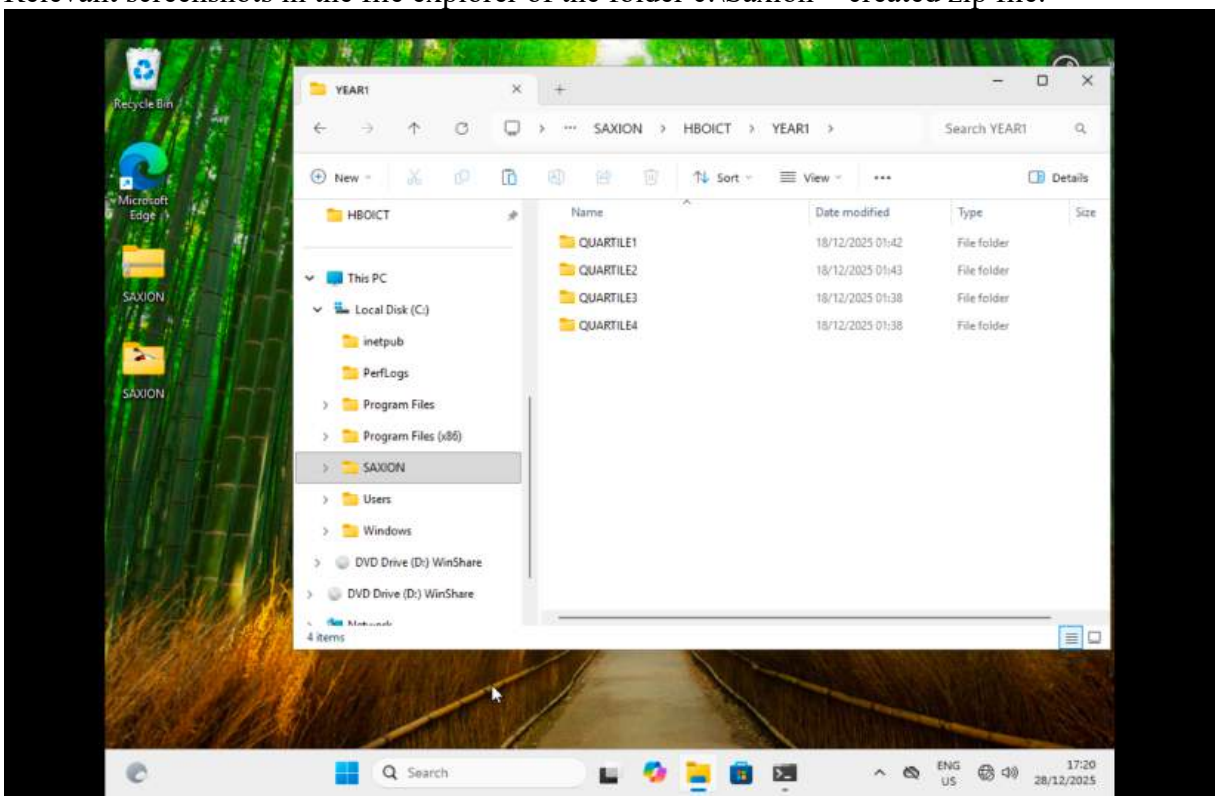
Relevant screenshots **copy** command:



Relevant screenshots **tree** command:

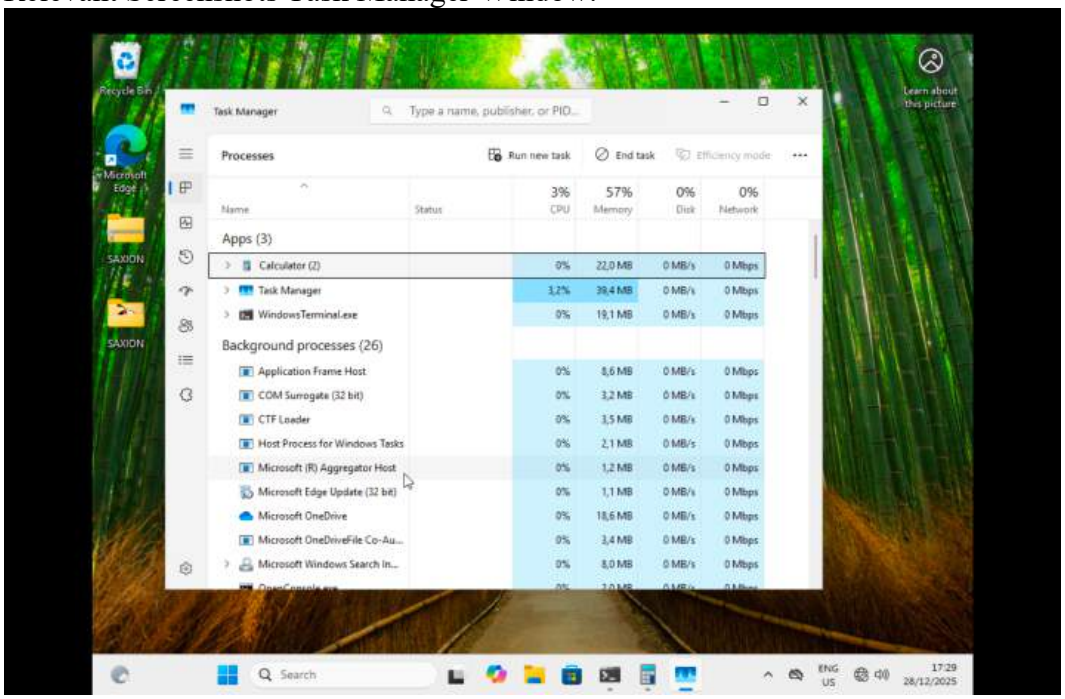


Relevant screenshots in the file explorer of the folder c:\Saxion + created zip file.



Terminating Processes

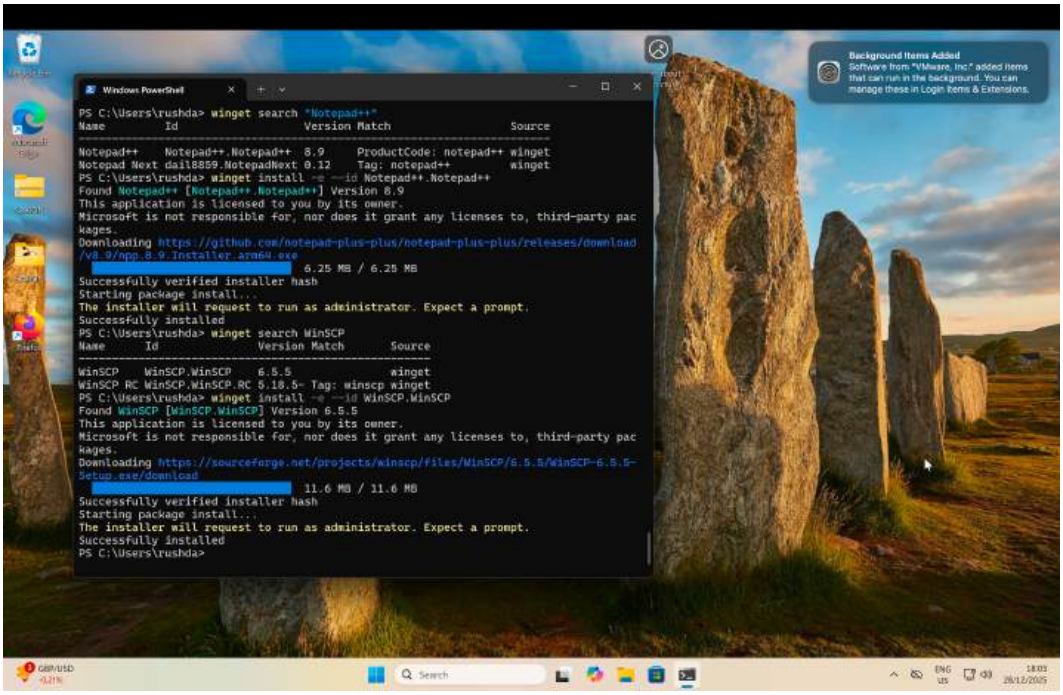
Relevant Screenshots Task Manager Window:



Install Software

Relevant screenshots that the following software is installed with winget:

- WinSCP



- Notepad++

- 7zip

Assignment 5.4: Working with Linux
Relevant screenshots + motivation

This screenshot shows the text file testfile.txt that I created in my home folder using the Ubuntu file manager.

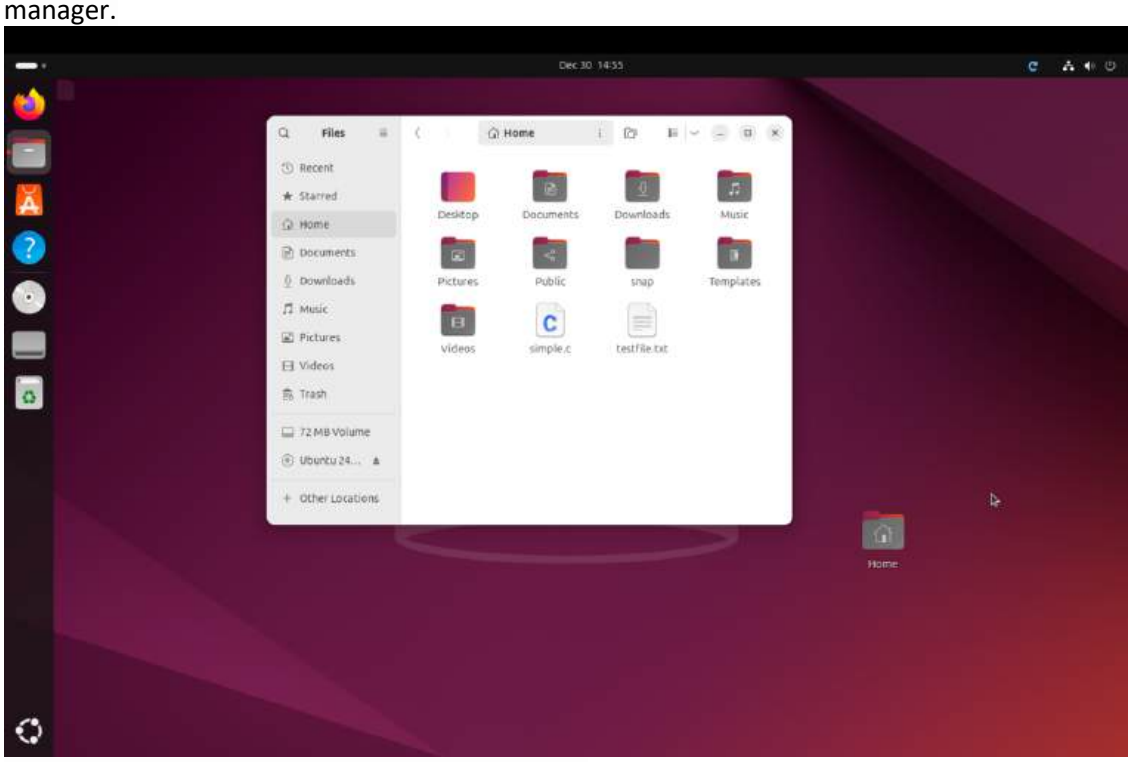


Figure 5.4-1 – Home folder with new text file

I copied testfile.txt from Home to the Documents folder using right-click Copy/Paste.

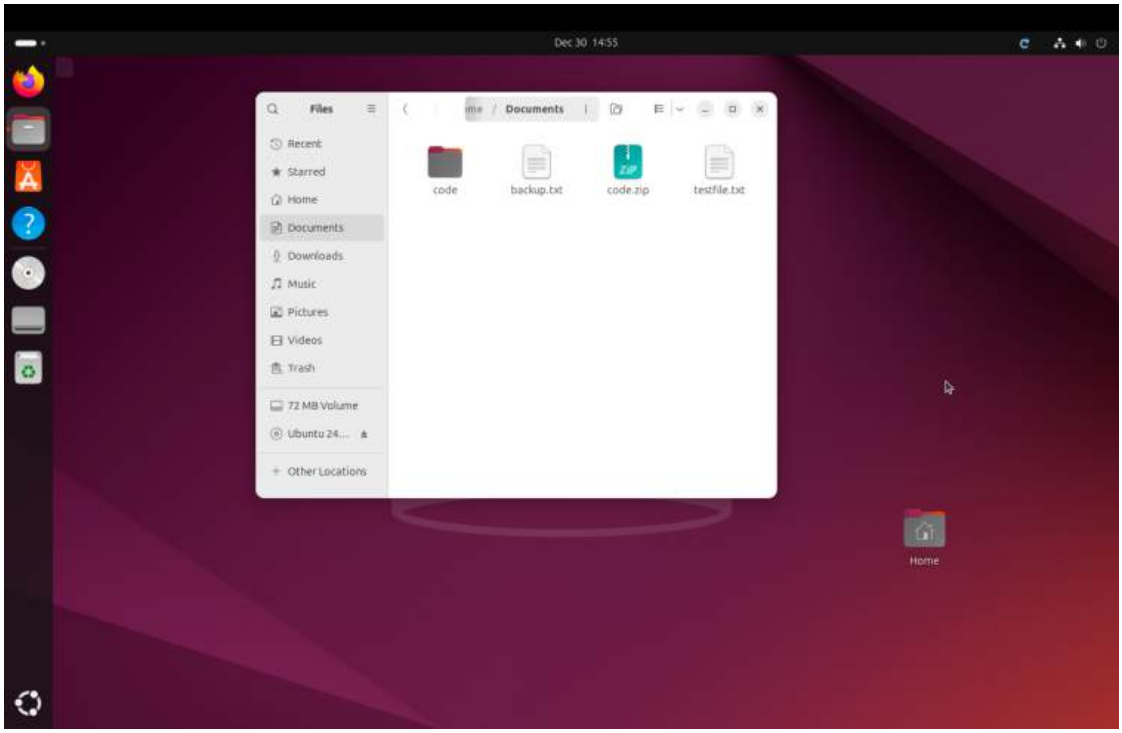


Figure 5.4-2 – testfile.txt folder after GUI copy

This screenshot shows the same copy operation on the command line. I used cp ~/testfile.txt ~/Documents/ and ls ~/Documents to check that the file is present. This shows that file operations can be done both via GUI and terminal.

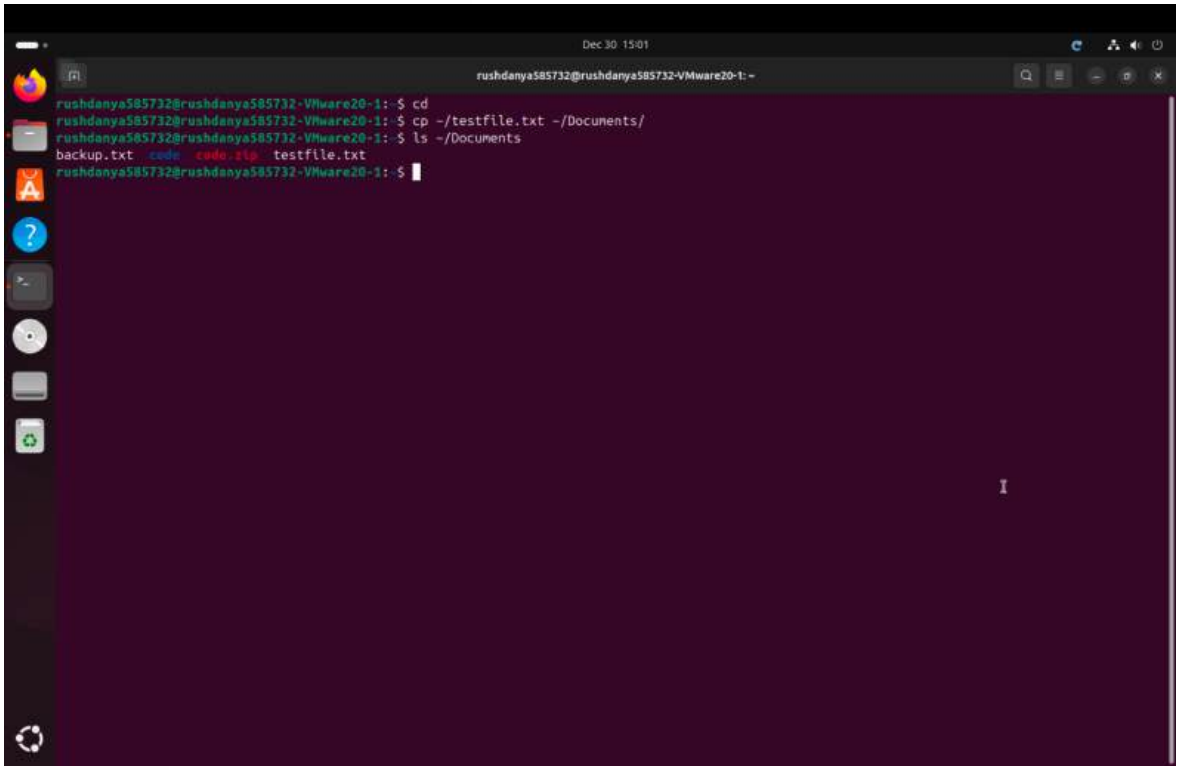


Figure 5.4-3 – Terminal copy of lab5.txt to Documents

Here I navigated to the /etc directory using the file manager (Other Locations → Computer → etc). This folder contains system-wide configuration files.

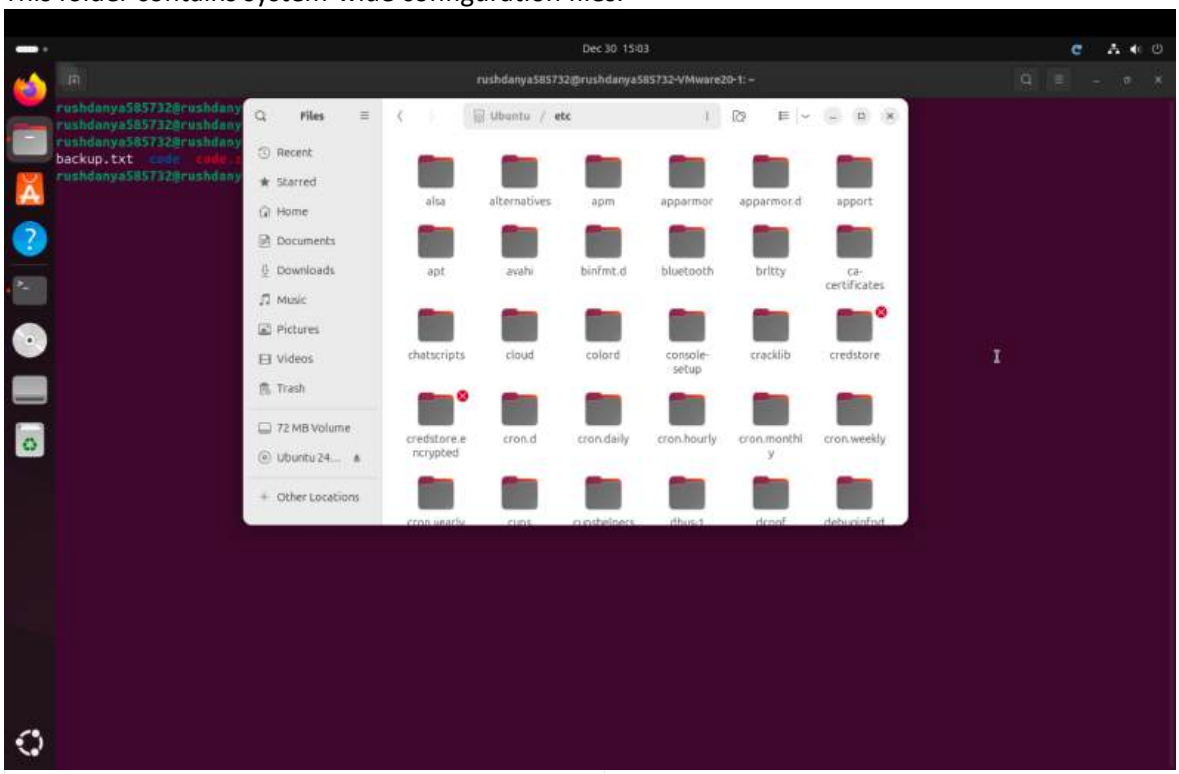


Figure 5.4-4 – /etc opened in the file manager

With cd /etc , pwd and cd plus pwd I showed how to move between system and home directories from the command line.

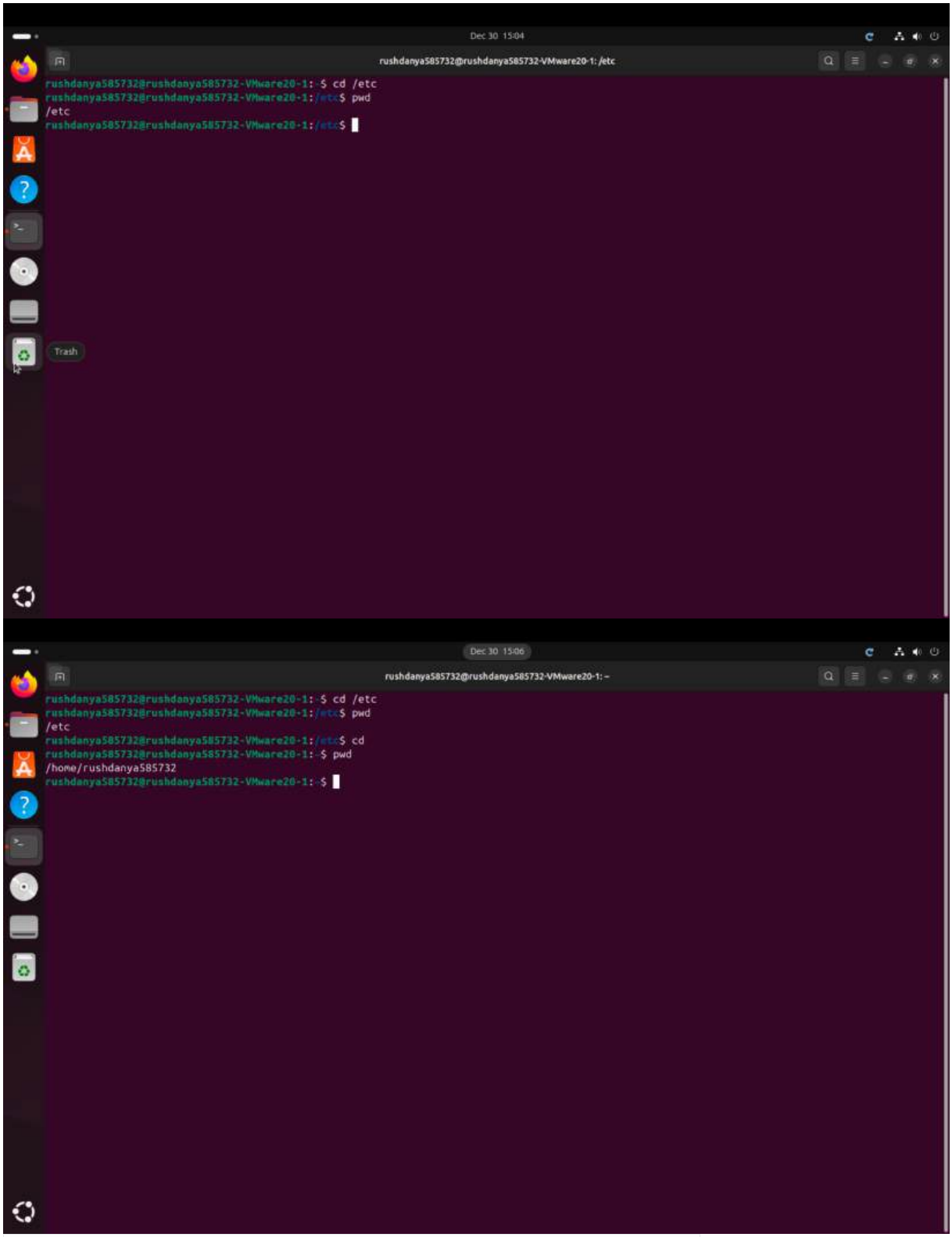


Figure 5.4-5,6 – Navigating to /etc and back to home in the terminal

Linux has a single root directory / and everything is under it, while Windows uses separate drive letters like C:\ , D:\ , etc. Linux also treats file names as case-sensitive.

/etc contains system-wide configuration files fort the operating system and services.

In this screenshot I created a tar archive of testfile.txt using tar -cvf compress.tar compress.txt and then compressed it with gzip compress.tar, which produced compress.tar.gz.

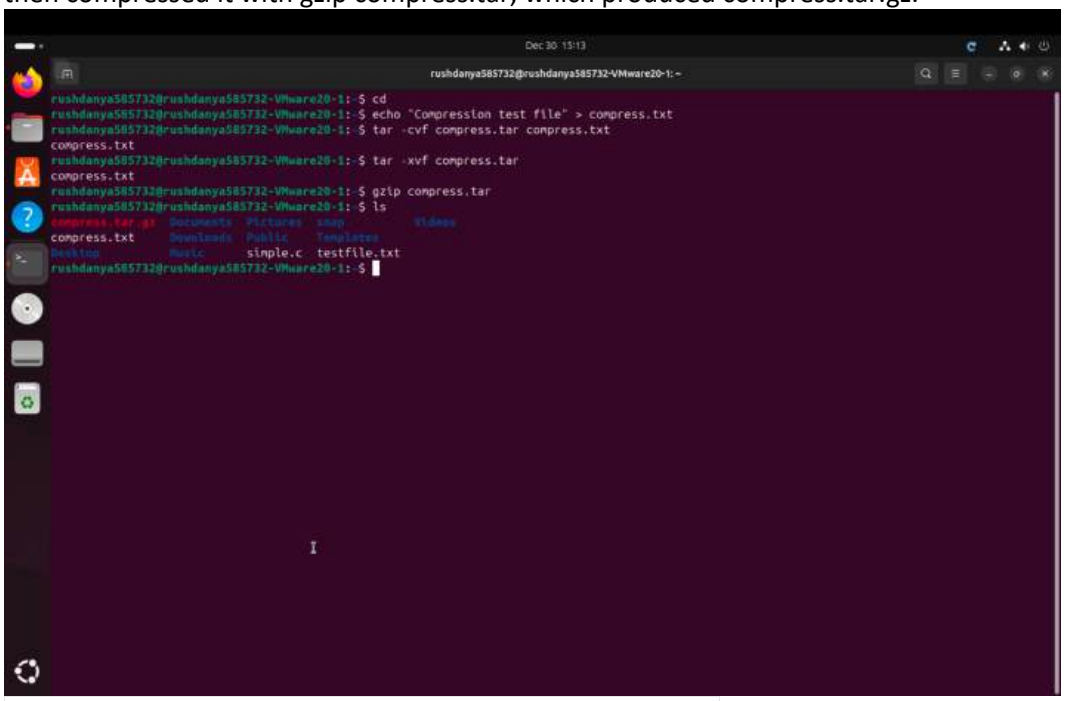


Figure 5.4-6 – Creating and compressing a tar archive

htop is an interactive process viewer that shows real-time CPU, memory and swap usage and lists all running processes. It lets you scroll, sort and kill processes.

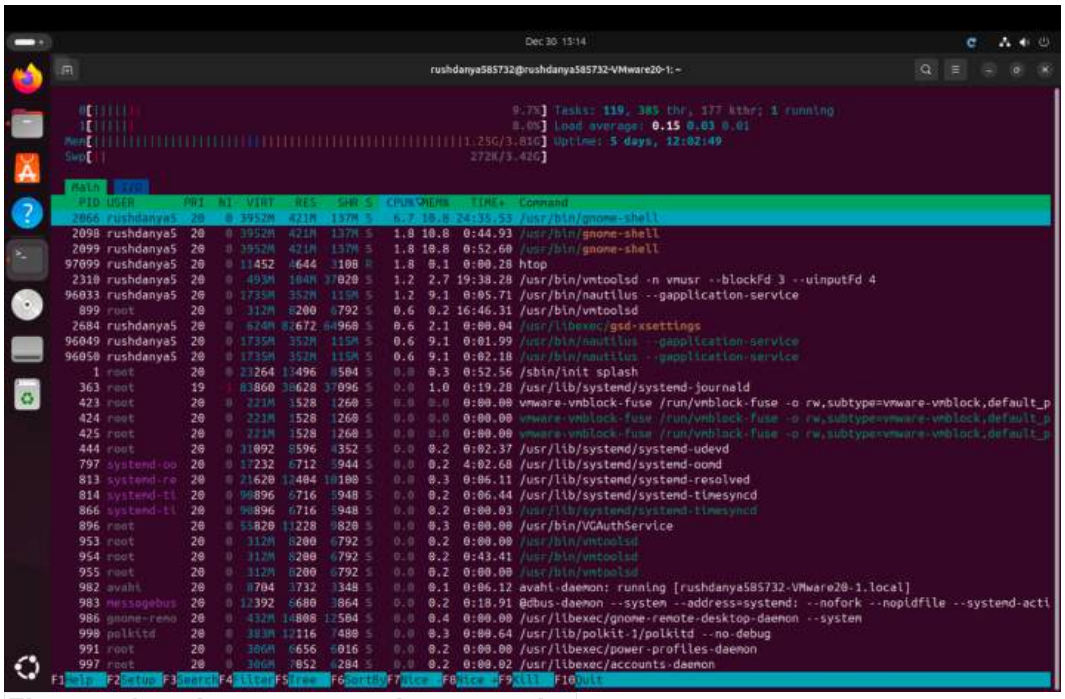


Figure 5.4-7 – htop process viewer running

Here I used the Ubuntu Software Center to search for and install the Sublime Text editor. This demonstrates how to install applications using the graphical package manager.

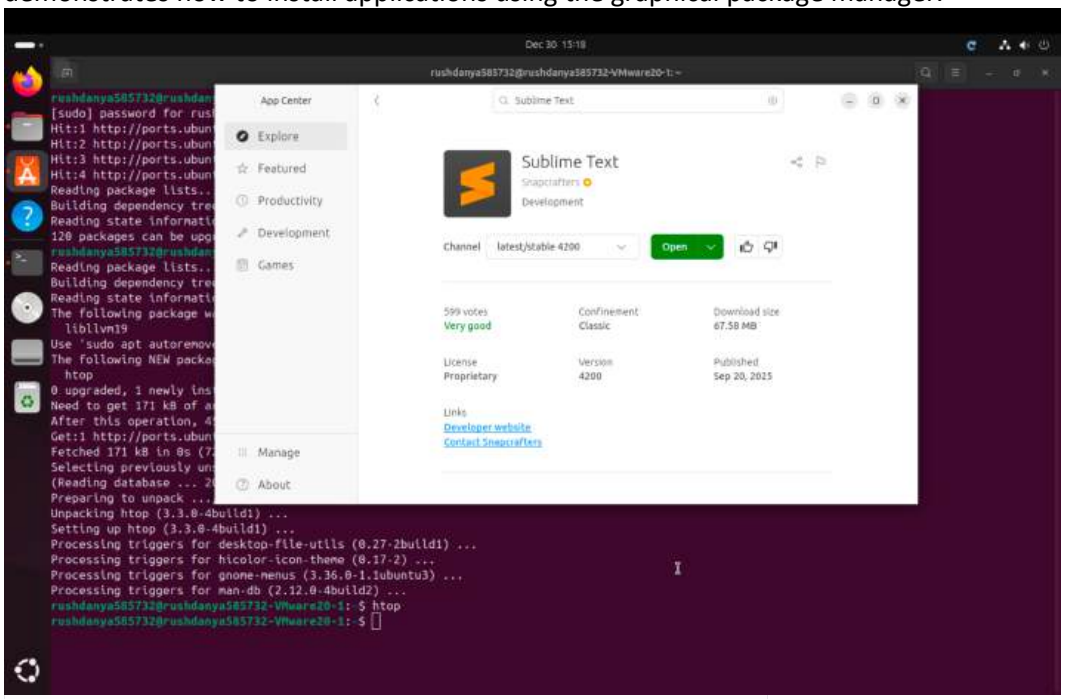


Figure 5.4-8 – Sublime Text installed via Ubuntu Software

In this terminal screenshot I installed neofetch with sudo apt install neofetch and ran it. neofetch prints system information (OS version, kernel, CPU, RAM, etc.) together with an ASCII logo. This shows that I can install packages from the command line and verify that they work.

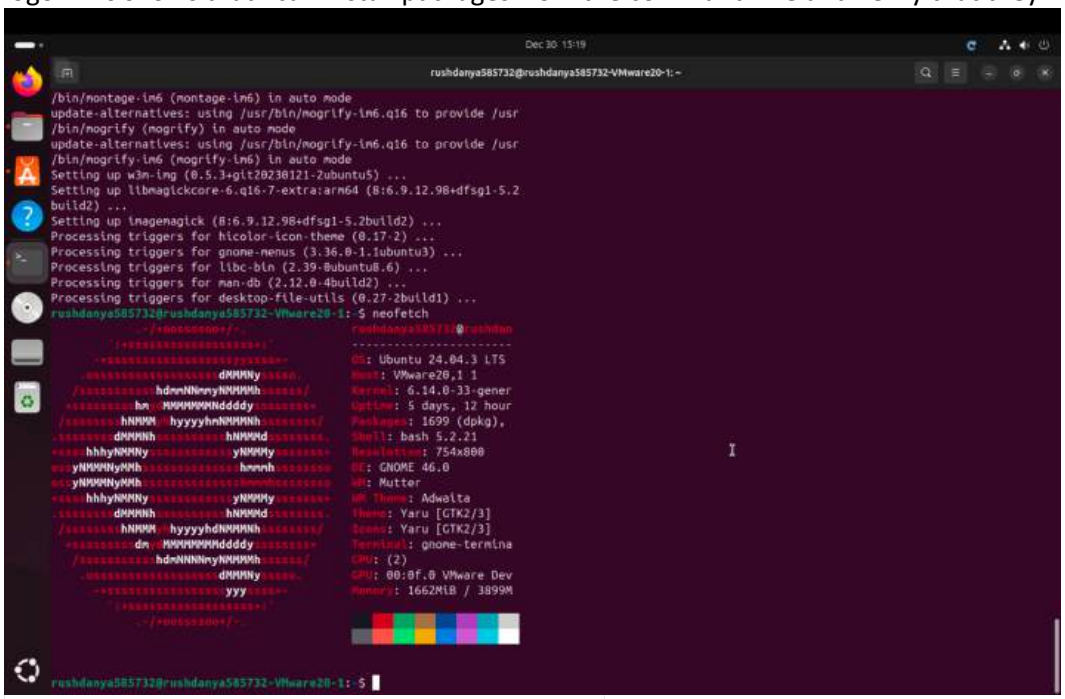


Figure 5.4-9 – neofetch output in the terminal

Assignment 5.5: Users and permissions on Linux
Relevant screenshots + motivation

cat file: prints the whole contents of a file to the terminal.
wc file: shows how many lines, words and bytes a file has.
wc -m file: shows how many characters a file has.
less file: opens the file in a viewer so you can scroll; q closses it.
head file: shows the first part of a file (by default the first 10 lines).
tail file: shows the last part of a file (by default the last 10 lines).
grep “text” file: searches the file and prints the lines that contain the given text.
grep -n “text” file: same as above, but also shows line numbers.

In this figure I create a new directory called hello in my home folder with mkdir ~/hello and move into it with cd ~/hello. This prepares a separate place for the shell script, just like you would organise scripts in their own folder on a real system.

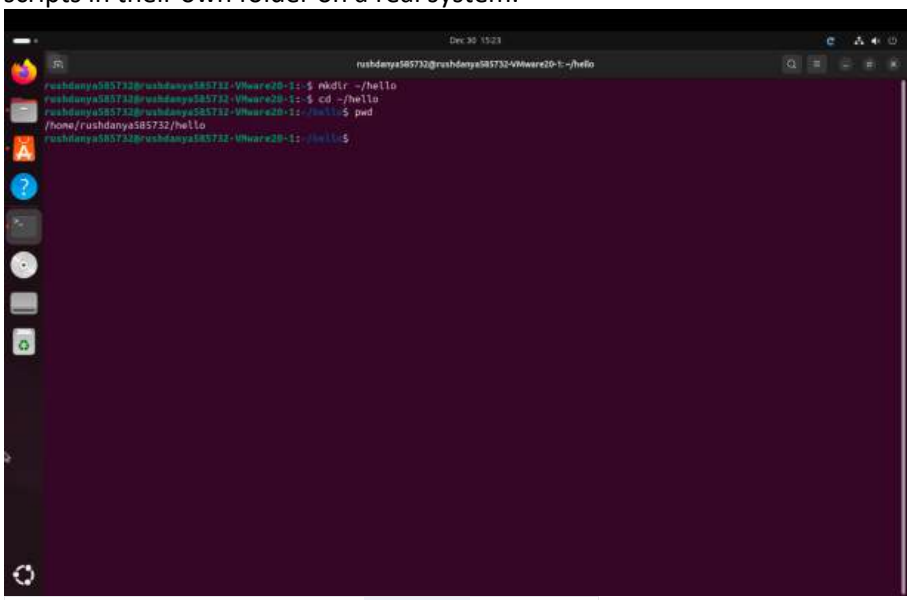


Figure 5.5-1 – Creating the `~/hello` directory

The hello.sh file contains the required bash shebang and an echo command with my name and student number.

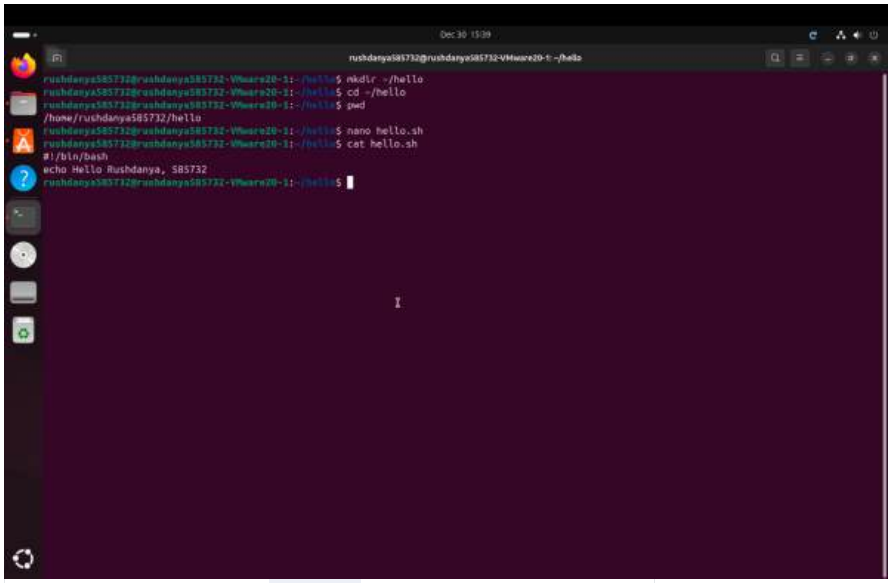


Figure 5.5-2 – Editing `hello.sh` with the script contents

After running chmod +x hello.sh, ls -l shows that the file has execute permission (rwxr-xr-x).

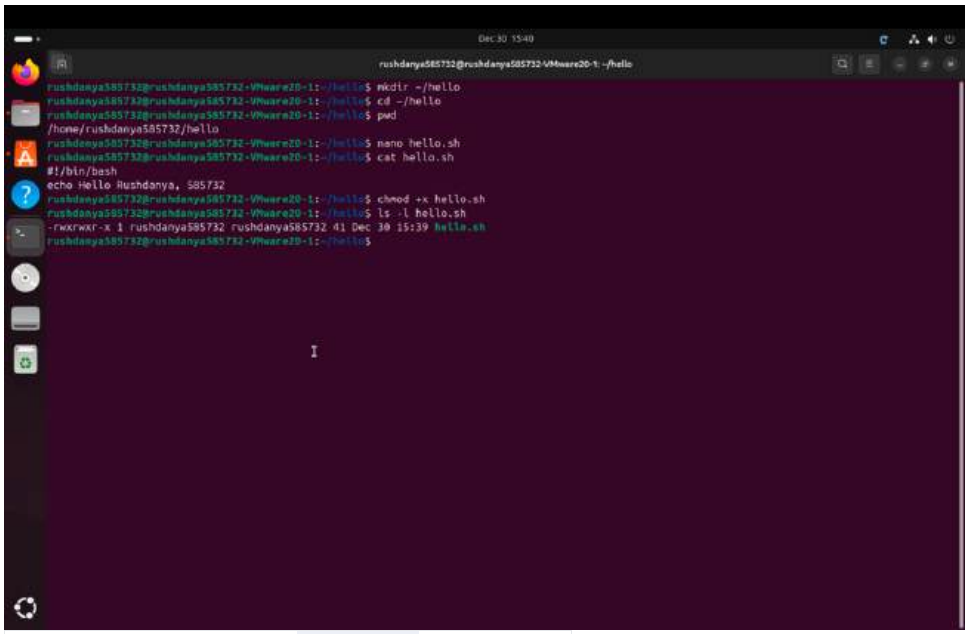


Figure 5.5-3 – Making `hello.sh` executable

Executing ./hello.sh prints the personalised “Hello …” message, proving the script runs correctly.

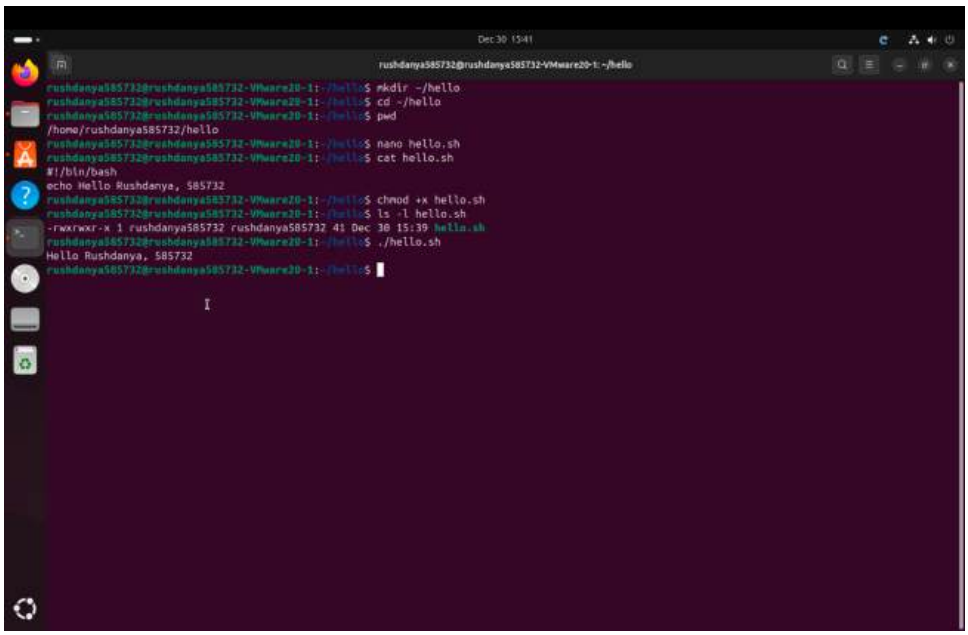


Figure 5.5-4 – Running the script with `./hello.sh`

With chmod 744 hello.sh, only the owner can execute the script while others only have read access (rwxr--r--).

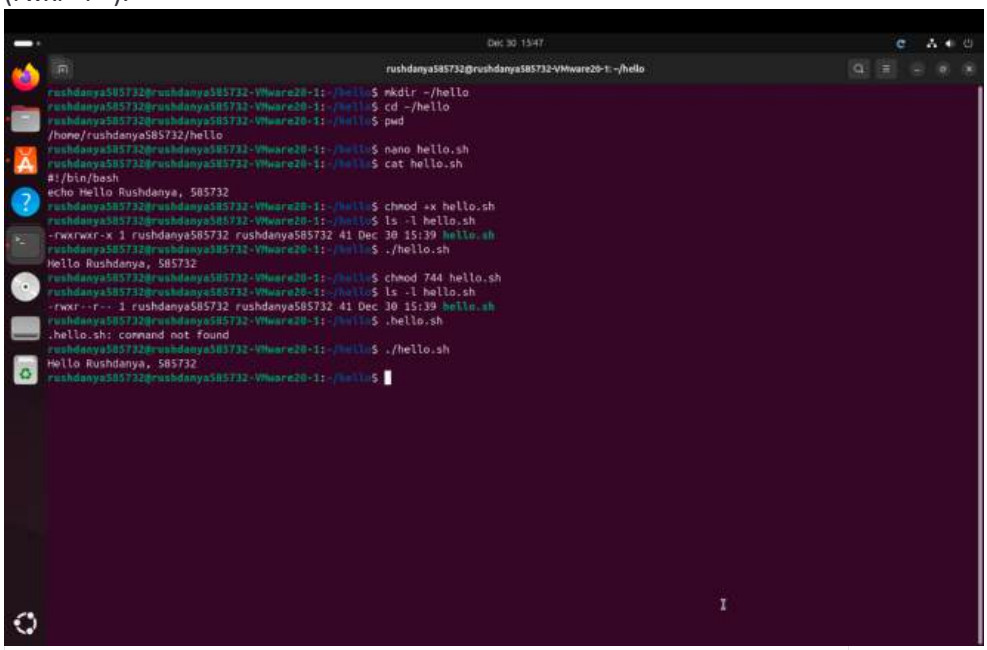


Figure 5.5-5 – Restricting execute permission to the owner only

Assignment 5.6: View the contents of files
Relevant screenshots + motivation

I created the week5_6 folder and downloaded the Sherlock Holmes book as sherlock.txt using wget.

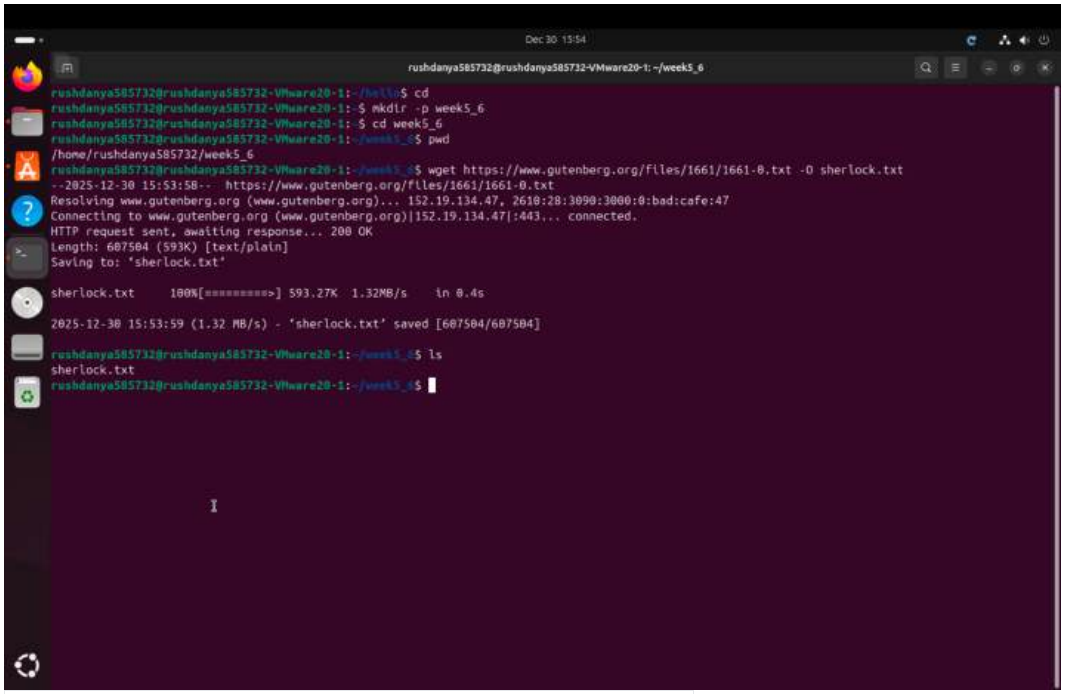


Figure 5.6-1 – Downloading the Sherlock Holmes text file

Using head/cat | head I displayed the first lines of the Sherlock Holmes text file

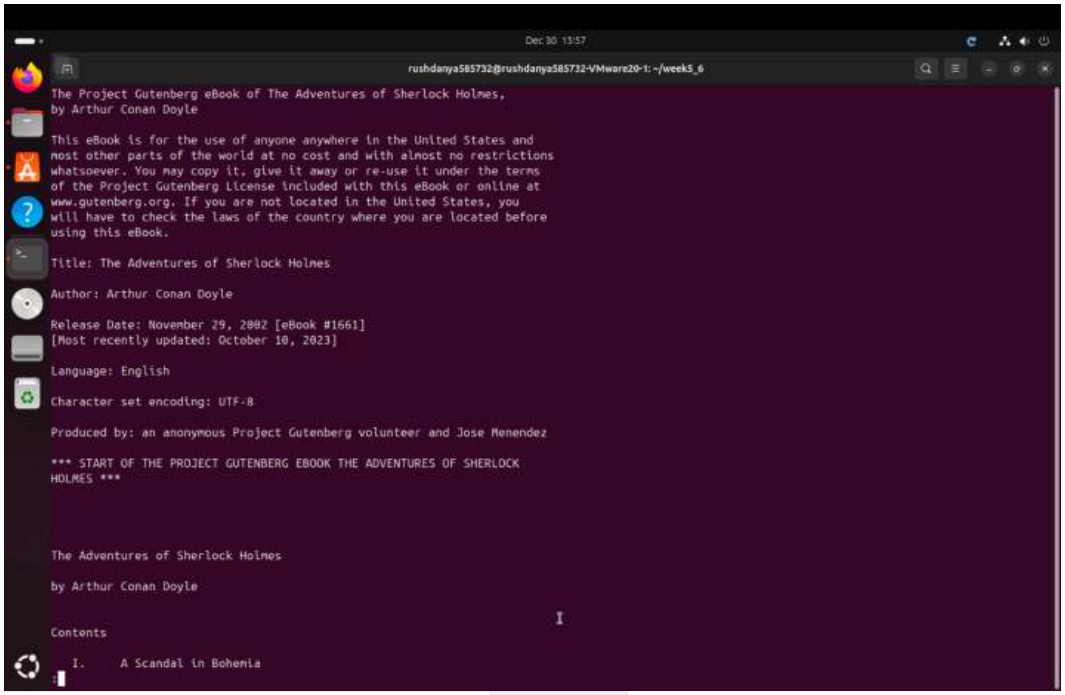


Figure 5.6-2 – Viewing the beginning of sherlock.txt

The wc and wc -m commands show how many lines, words and characters the file contains.The file has 12306 lines, 107562 words and 593731 characters.

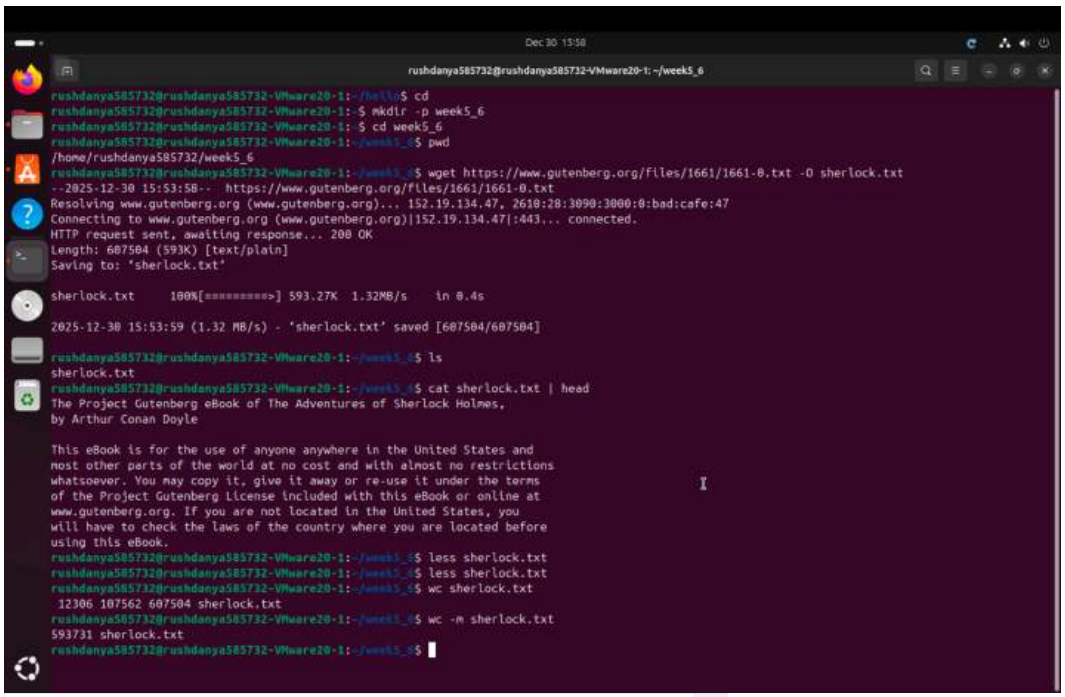


Figure 5.6-3 – Counting lines, words and characters with wc

grep -n "kingdom" sherlock.txt lists every line containing “kingdom” together with its line number.The word ‘kingdom’ occurs on lines 490 and 1124.

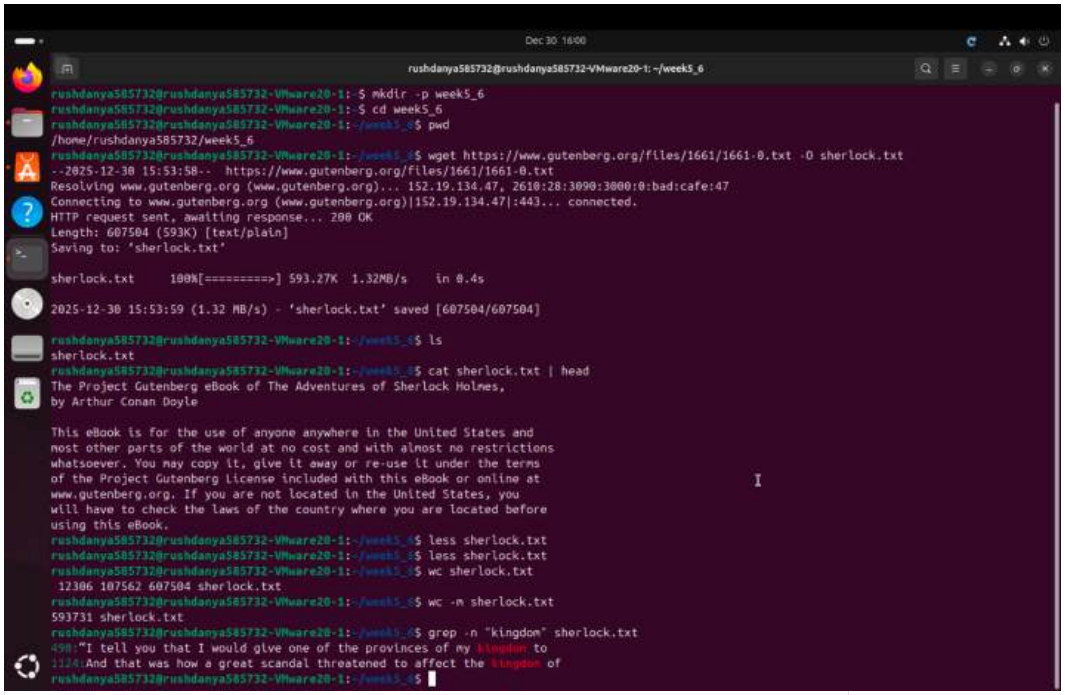


Figure 5.6-4 – Finding all lines that contain the word “kingdom”

Combining head -n … | tail -n 41 displays 20 lines before and after the first line that contains “kingdom”.

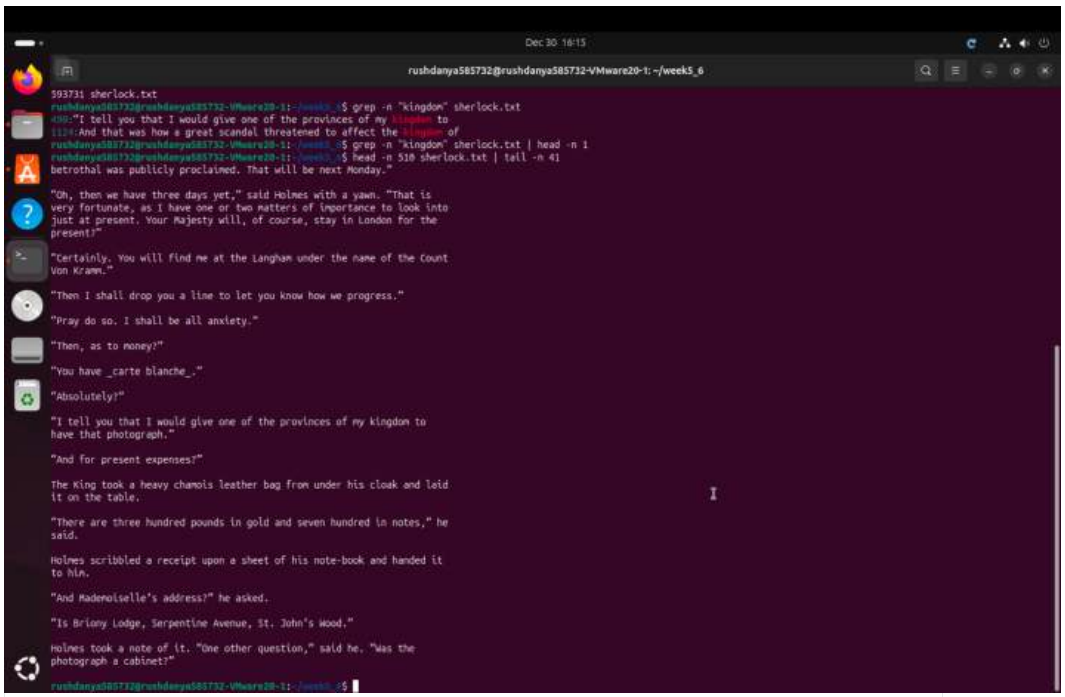


Figure 5.6-5 – Showing context around the first “kingdom” occurrence

Assignment 5.7: Digital forensics
Relevant screenshots + motivation

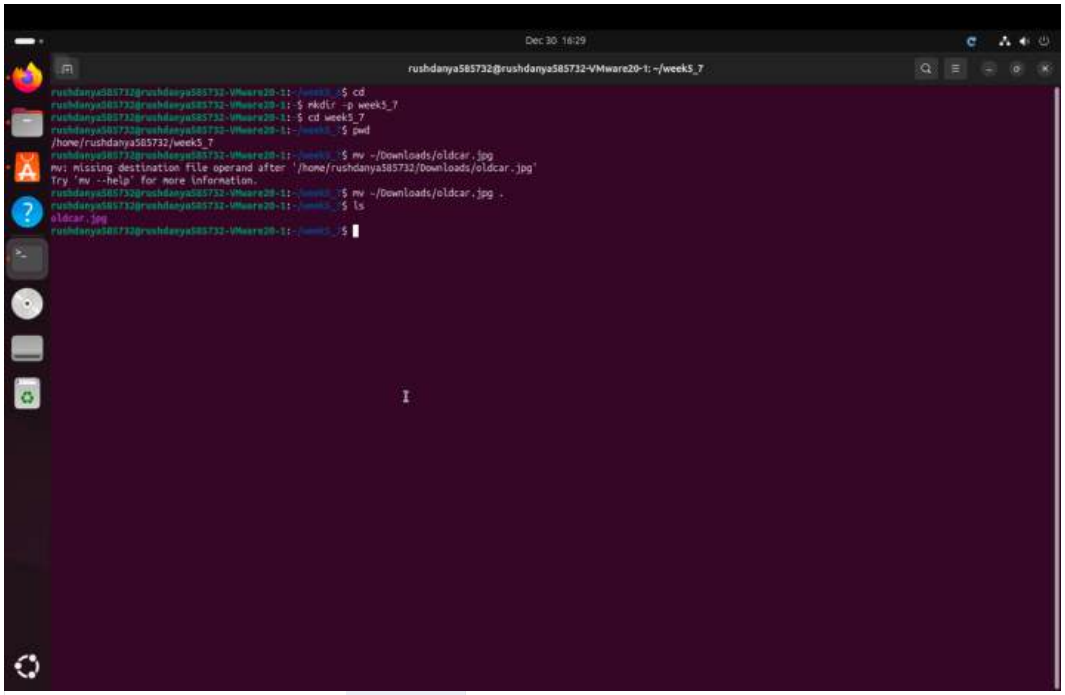


Figure 5.7-1 – Downloading oldcar.jpg

exiftool oldcar.jpg reveals camera information, including Make: motorola and Camera Model Name: moto g(6) play.

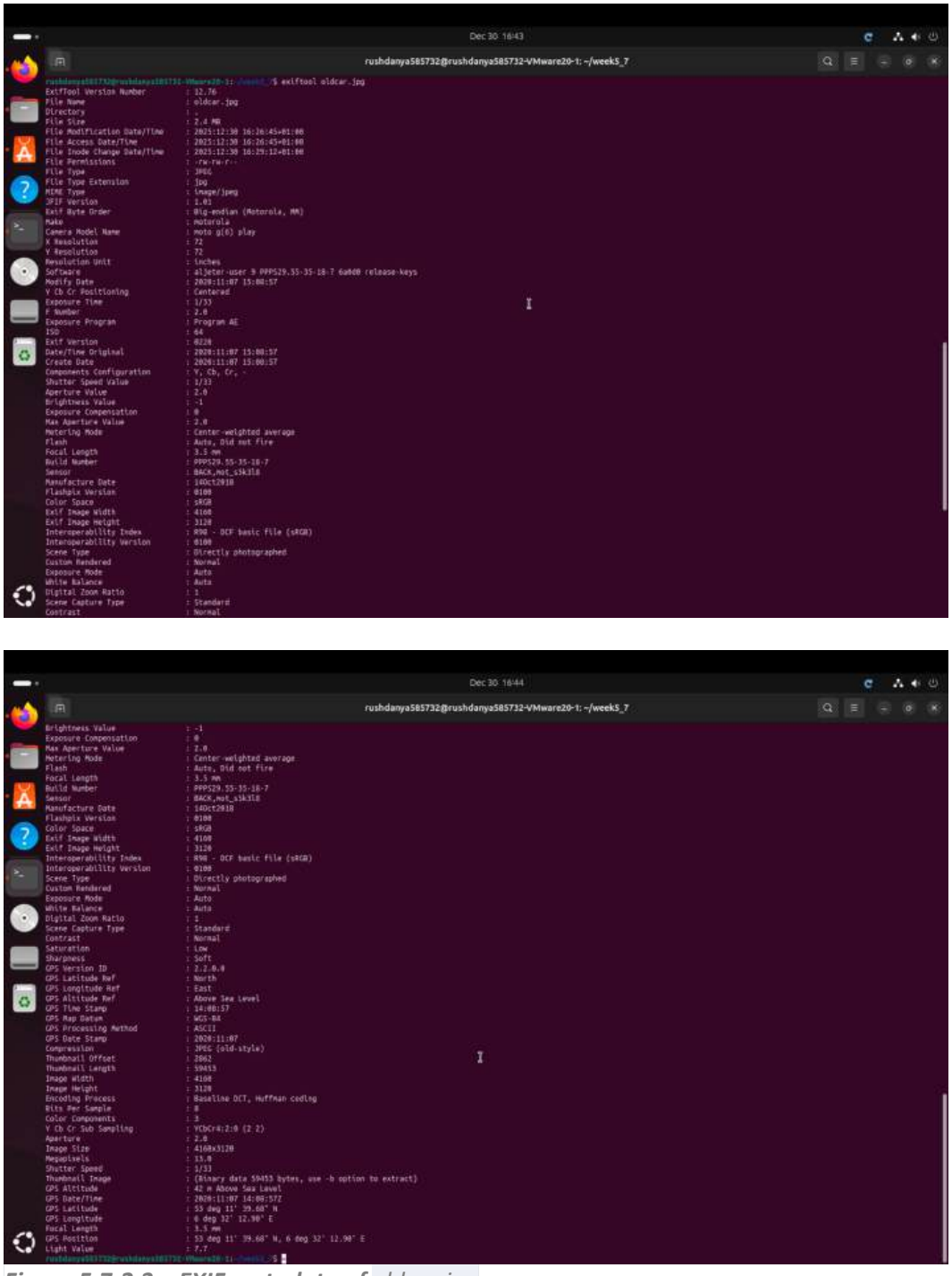


***Figure 5.7-2,3 – EXIF metadata of** oldcar.jpg*

I analysed the photo file with the command exiftool oldcar.jpg on my Ubuntu VM.
The EXIF data shows that the picture was taken with a **Motorola moto g(6) play** smartphone.
The GPS information in the file gives the coordinates **53°19’39.68”N, 6°32’12.90”E** at an altitude of **42 m**.
I entered these coordinates into Google Maps and checked the surroundings with Street View.
The coordinates point to a **rural area in the Netherlands, in the province of Groningen, near the village of Harnem in the municipality of Eemsdelta.**

I removed the .jpg extension from the filename using mv oldcar.jpg oldcar.

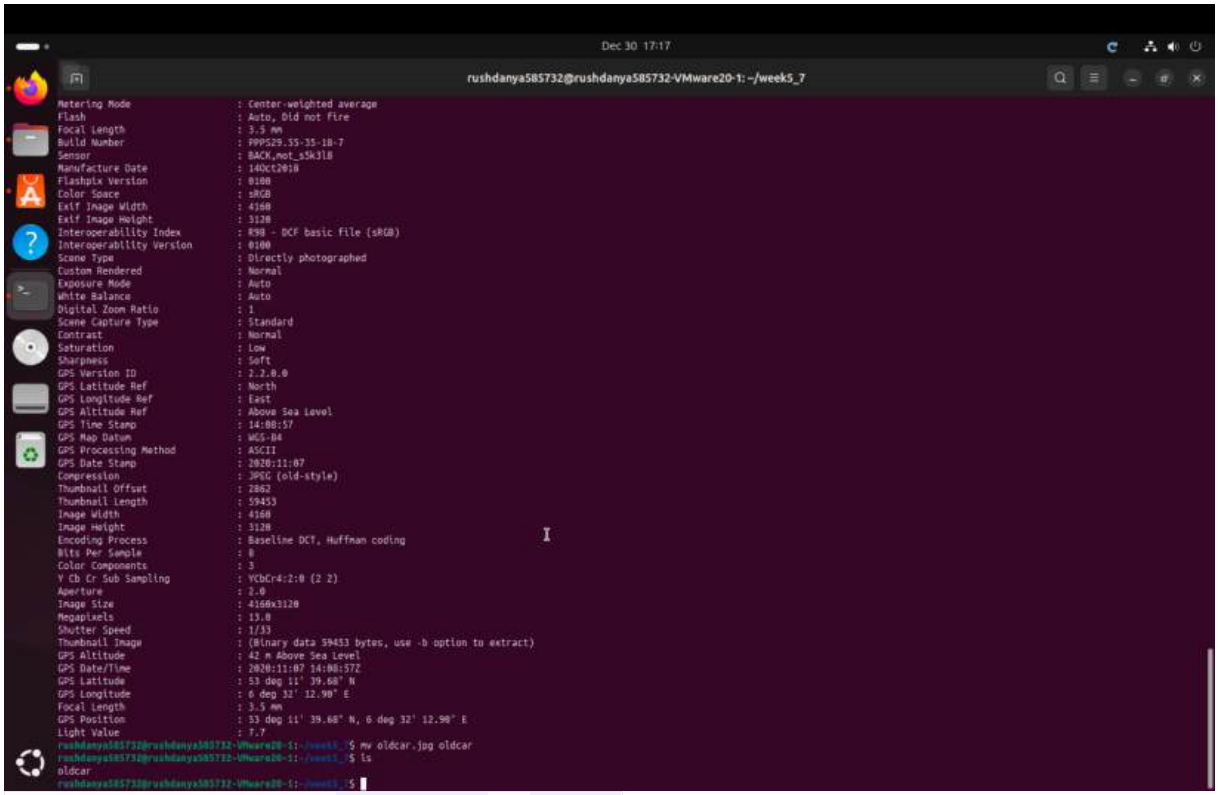


Figure 5.7-4 – Renaming `oldcar.jpg` to `oldcar`

After removing the .jpg extension, Ubuntu still recognises the file oldcar as JPEG image data. So yes, Ubuntu still considers it to be a JPG file, because it determines the type from the file's contents, not from the extension.

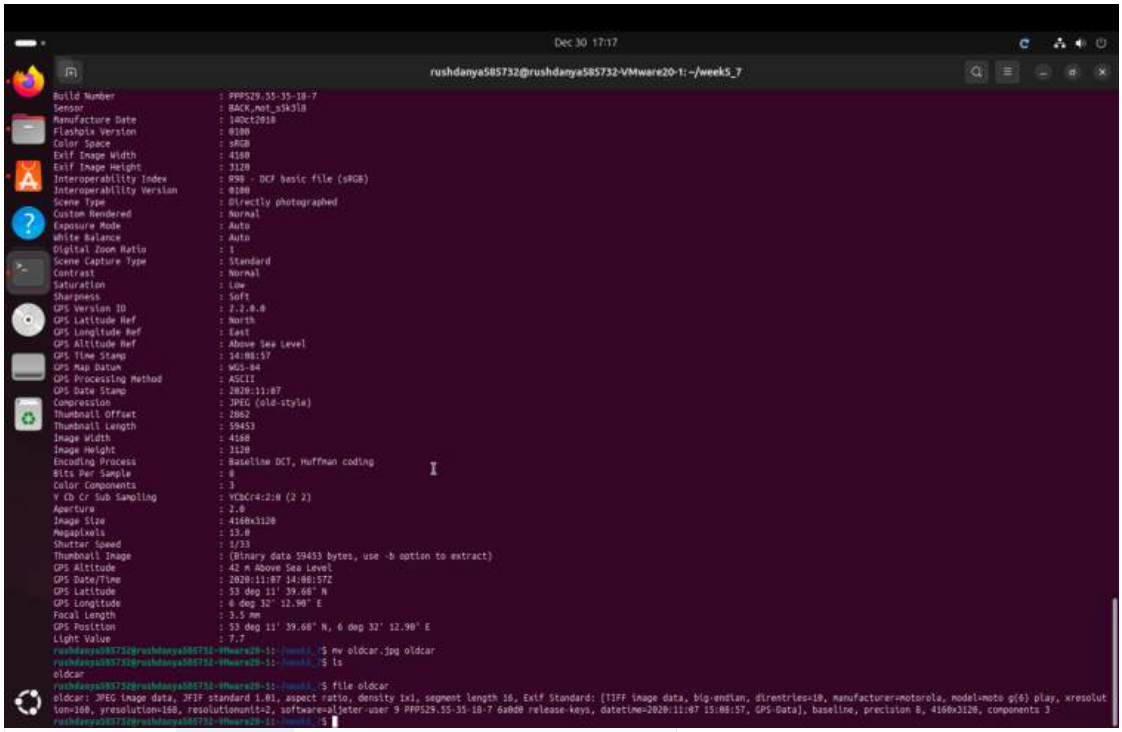


Figure 5.7-5 – file oldcar still reporting JPEG image data

I created `email-base64.txt` and pasted the long BASE64-encoded text from the assignment into it.

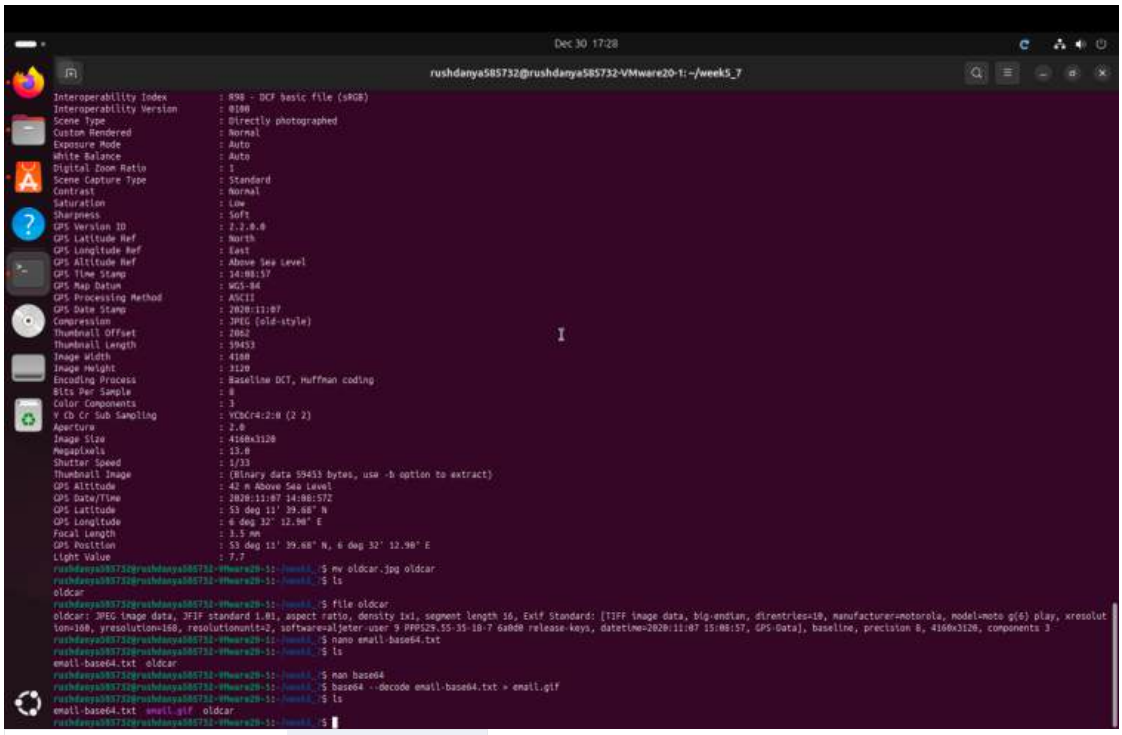


Figure 5.7-6 – Creating email-base64.txt

Using base64 --decode email-base64.txt > email.gif I reconstructed the original binary GIF file.

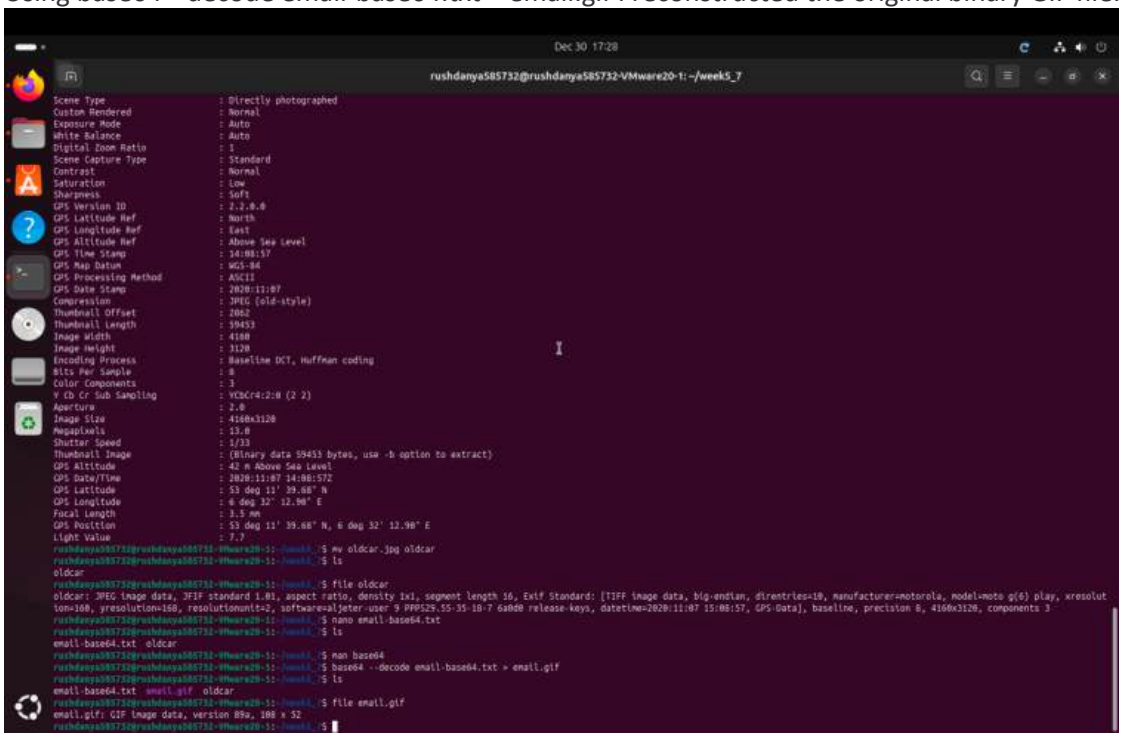


Figure 5.7-7 – Decoding the BASE64 string into email.gif

Opening email.gif in an image viewer shows the Saxion logo, confirming that the BASE64 text decoded correctly.

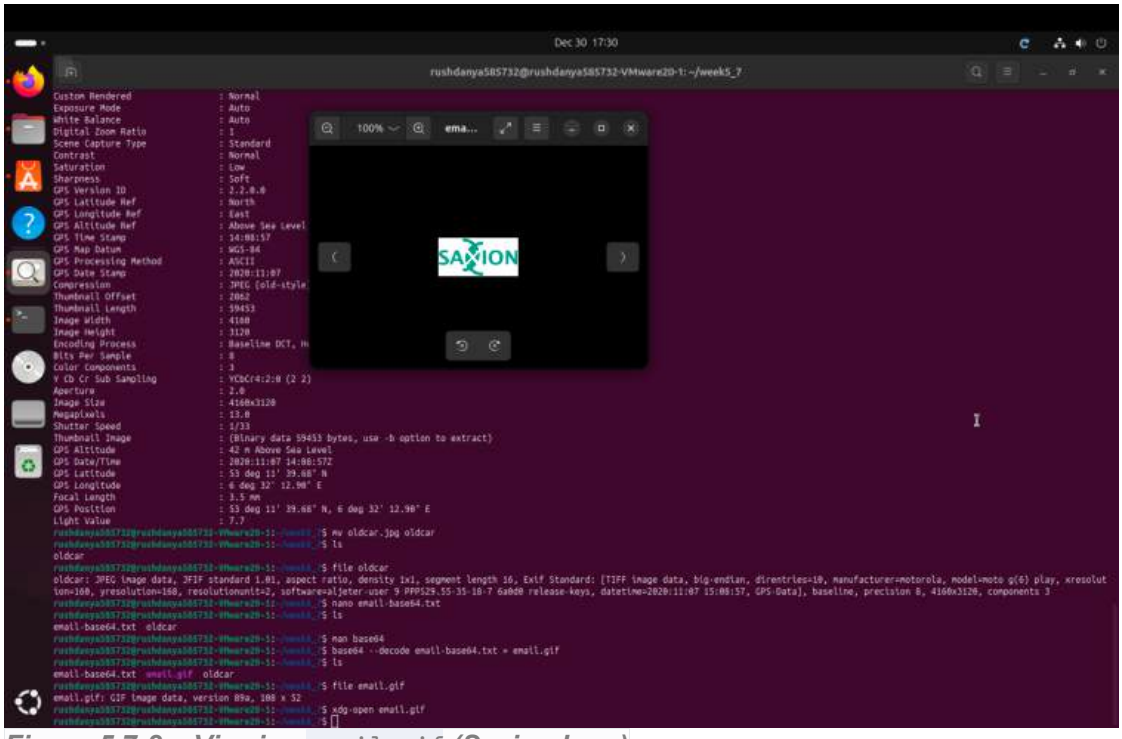


Figure 5.7-8 – Viewing email.gif (Saxion logo)

Assignment 5.8: Steganography
Relevant screenshots + motivation

This screenshot shows the downloaded apple2.jpg file, which contains hidden data in its least significant bits.

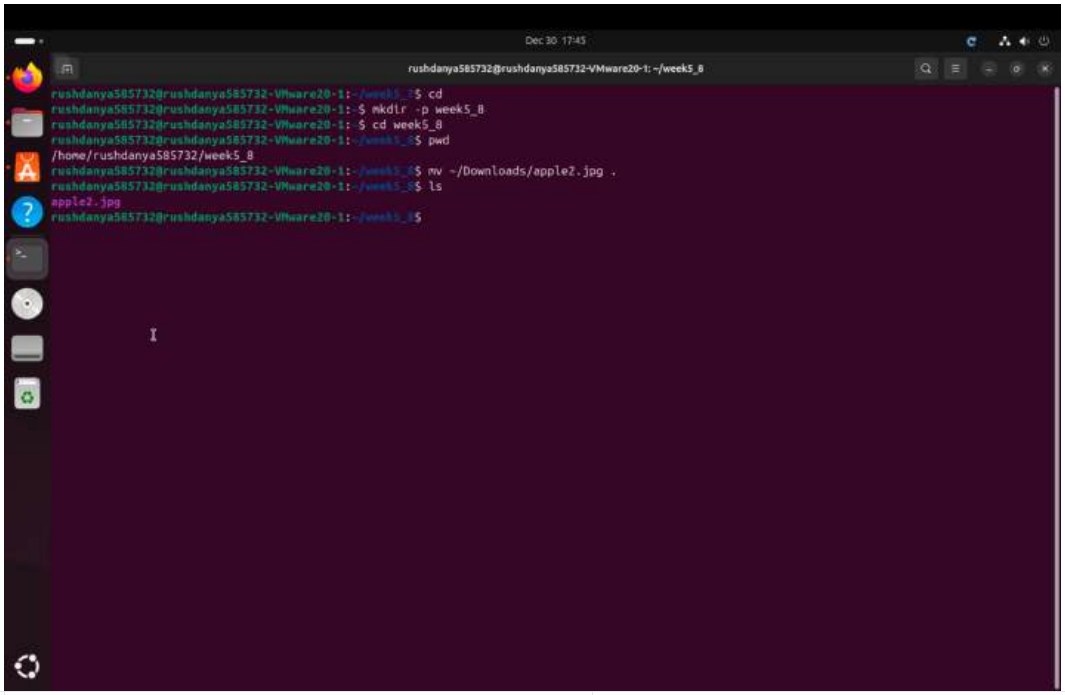


Figure 5.8-1 – apple2.jpg on the Ubuntu desktop

steghide --help confirms that the tool is installed and shows the extract option used to retrieve the hidden file.

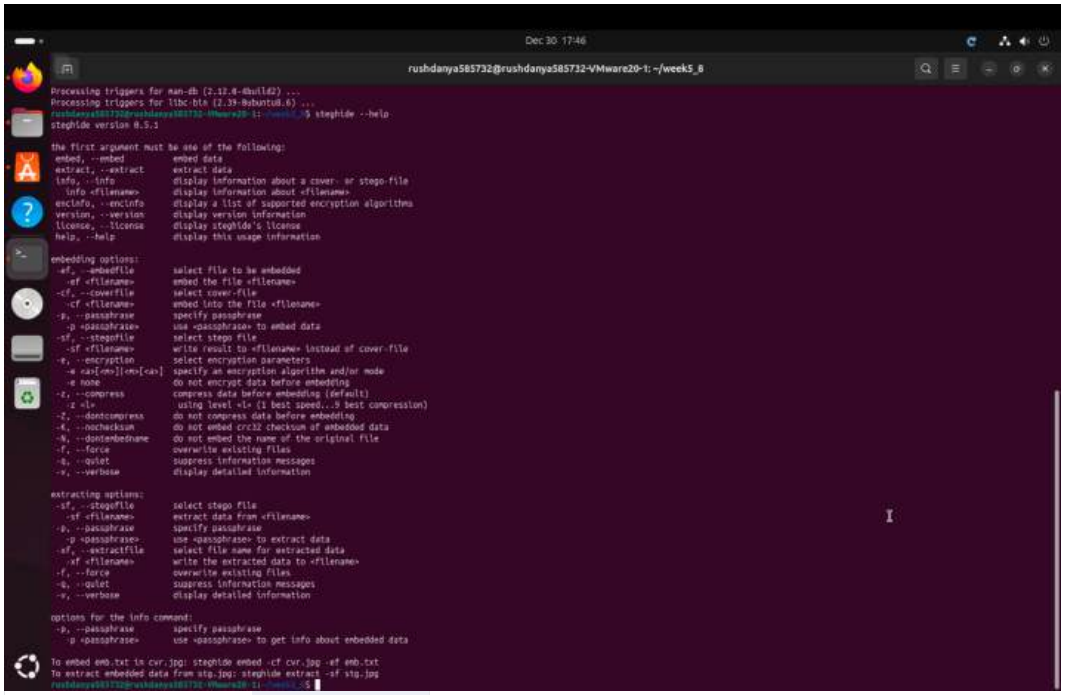


Figure 5.8-2 – Using steghide --help

With steghide extract -sf apple2.jpg and the passphrase apple2 I extracted the concealed text file to disk.

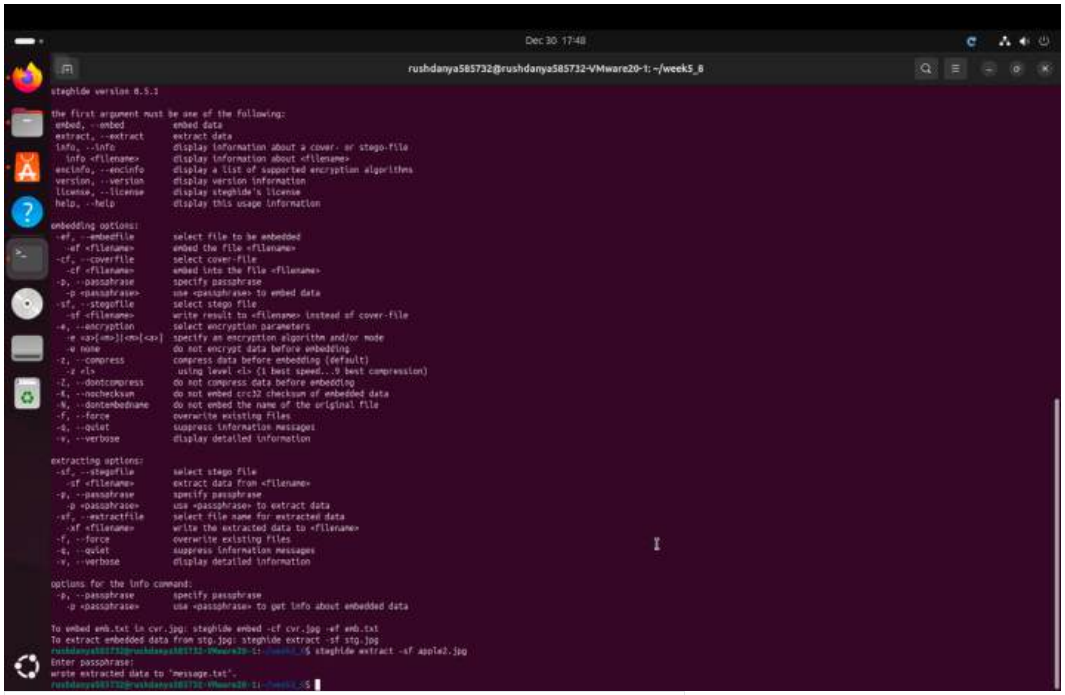


Figure 5.8-3 – Extracting the hidden file from apple2.jpg

Using cat (or a text editor) I show the message stored inside the hidden text file that was embedded in the image.

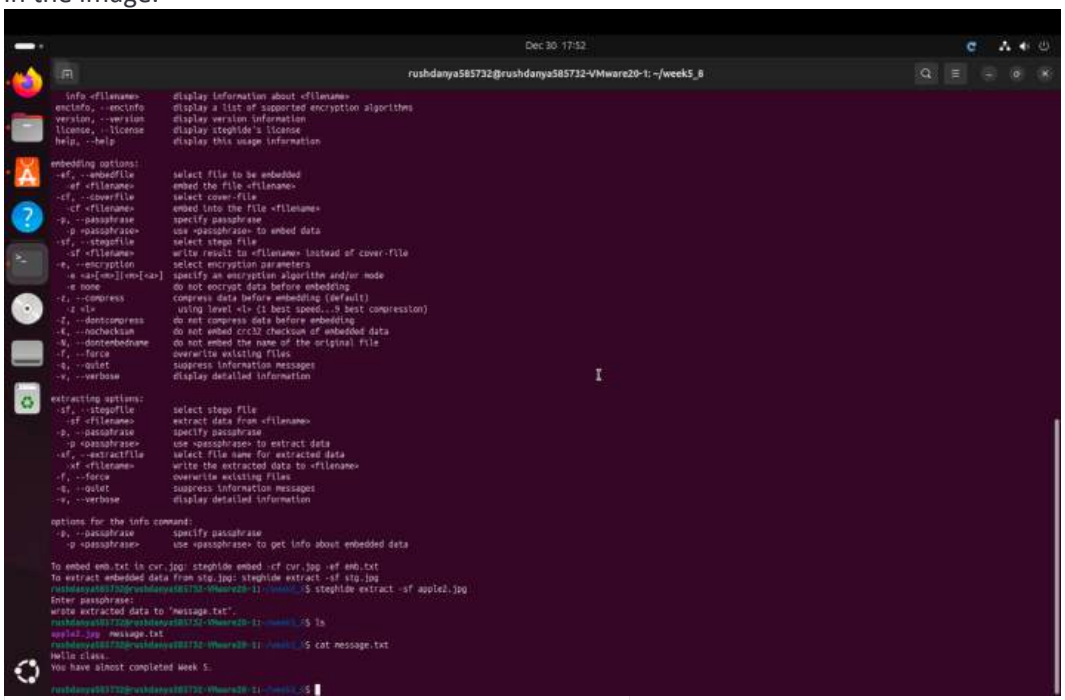


Figure 5.8-4 – Viewing the extracted secret text

Assignment 5.9: Capture disk images

Make relevant screenshots + motivation:

- Proof that the Debian 13 server stored a back-up image of the Ubuntu 24.04 Desktop VM.

On the Debian server I enabled ssh and created /srv/images as the storage location for disk image files.

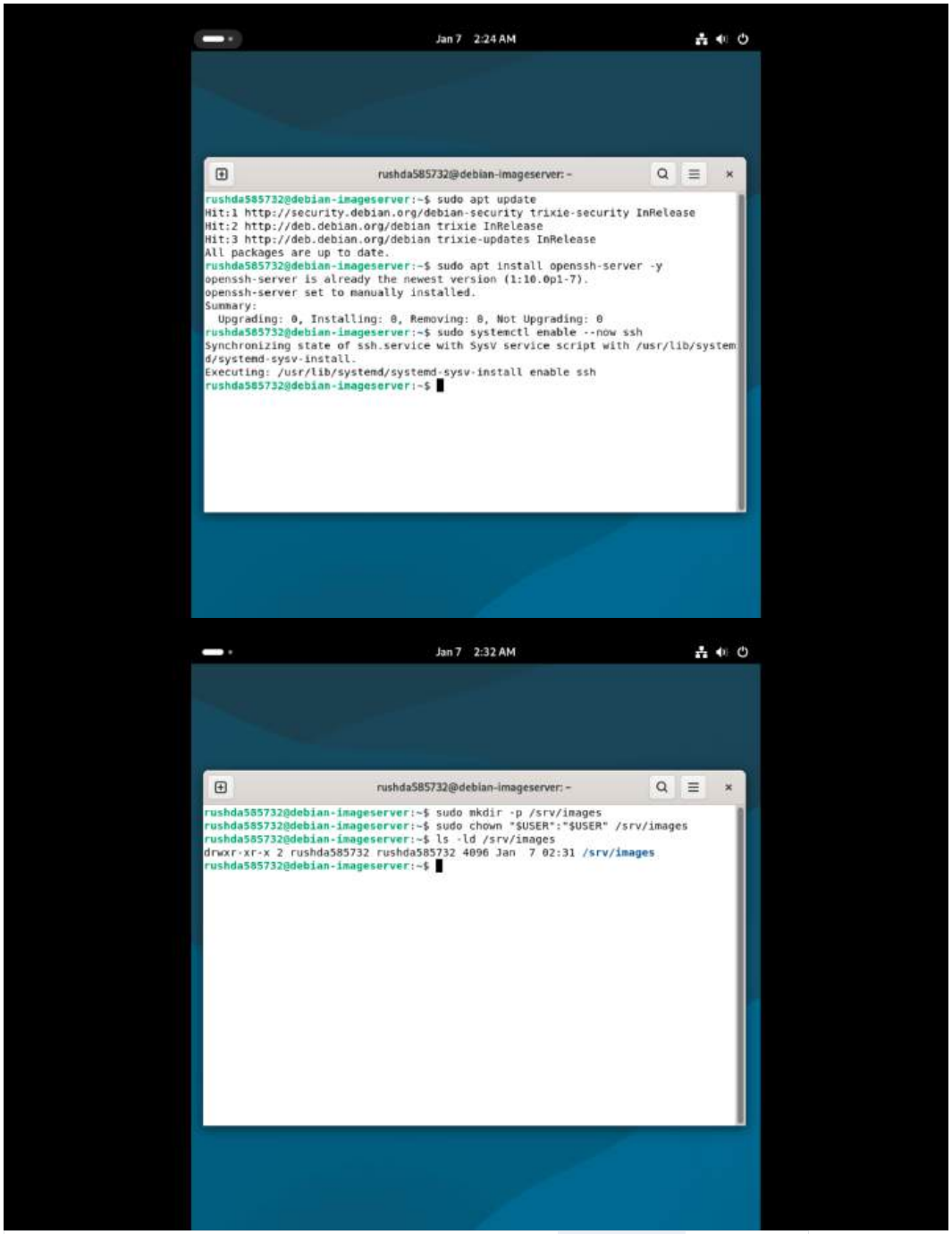


Figure 5.9-1,2 – Debian image server with SSH and /srv/images configured

The ip a output shows the server's IP address, which I use in all ssh commands from the Ubuntu VMs.

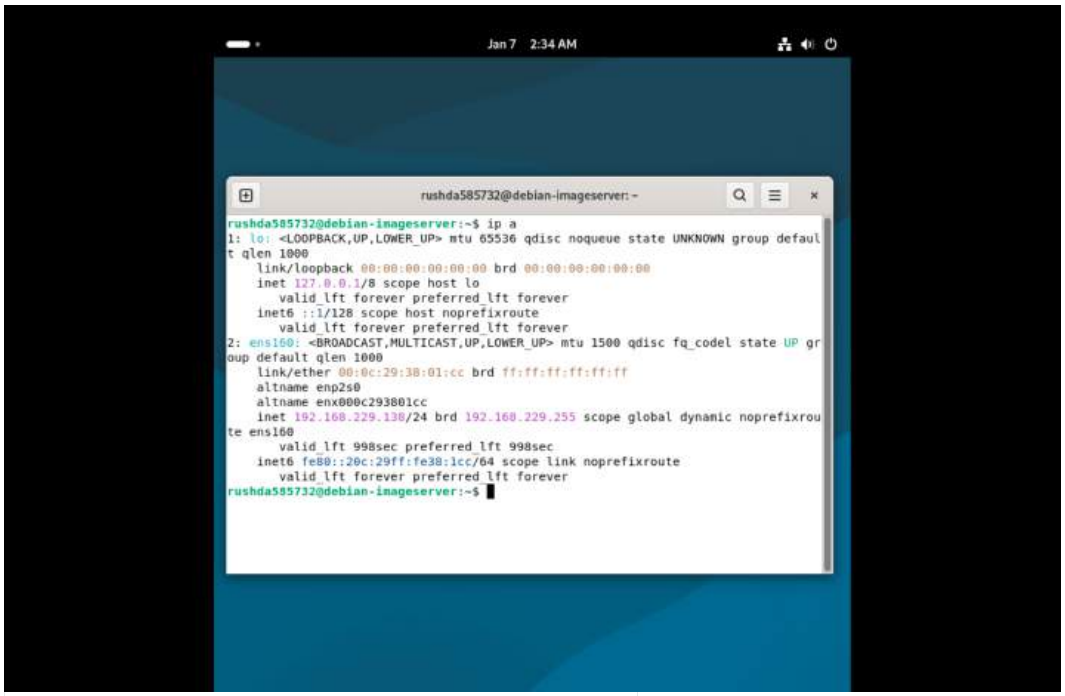


Figure 5.9-3 – IP address of the Debian image server

In the original Ubuntu VM booted as "Try Ubuntu", lsblk identifies the full disk device (e.g. /dev/nvme0n1) to be captured.

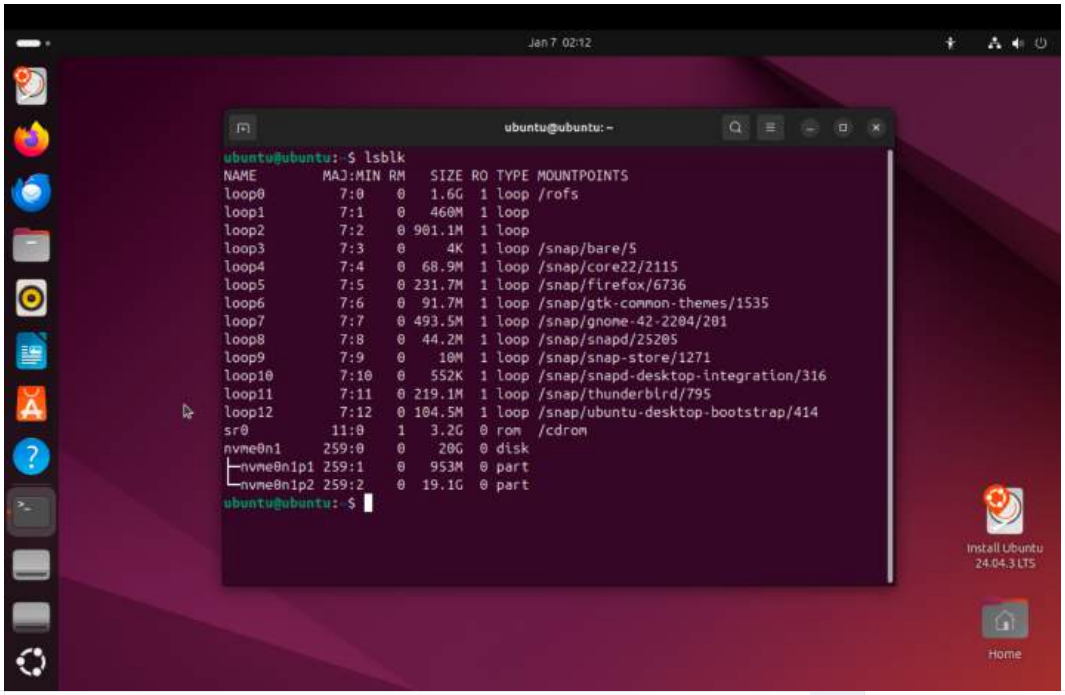


Figure 5.9-4 – Ubuntu client VM booted into Live session with disk lsblk

The command sudo dd if=/dev/… bs=4M status=progress | gzip | ssh user@debian 'cat > /srv/images/ubuntu2404_vm.img.gz' copies, compresses and streams the entire disk to the Debian server.

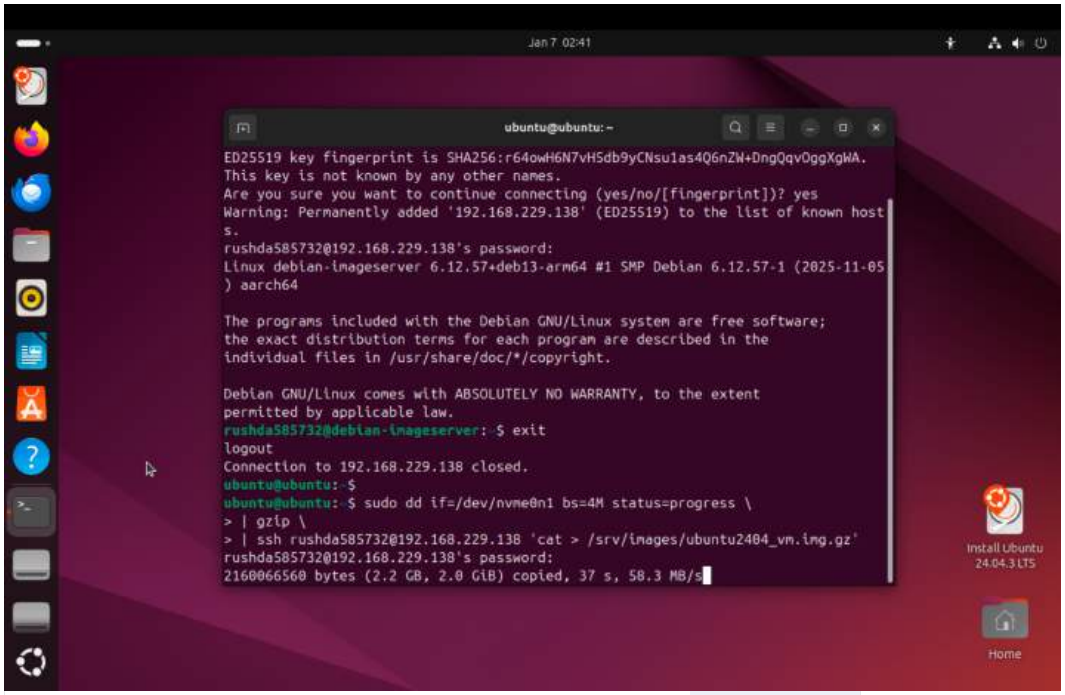


Figure 5.9-5 – Capturing the Ubuntu disk to Debian with dd | gzip | ssh

ls -lh /srv/images on Debian shows the large ubuntu2404_vm.img.gz file, proving the capture completed successfully.

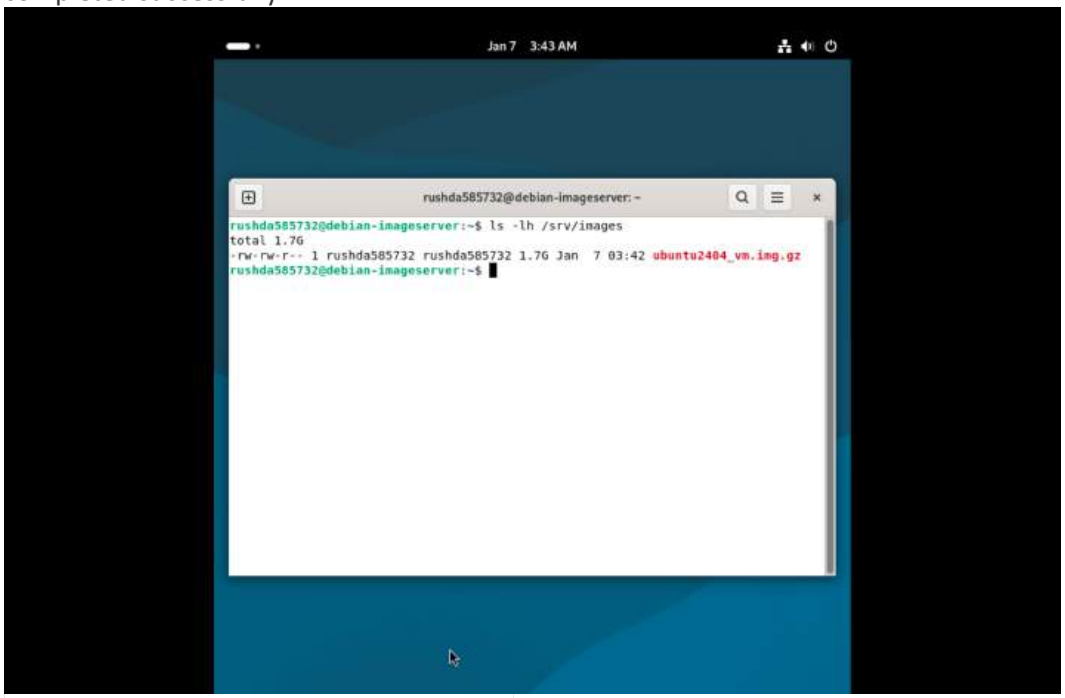


Figure 5.9-6 – Image file stored on Debian

- Proof that you can restore the back-up image into an empty VM.

On the new empty VM in “Try Ubuntu” mode, lsblk shows the target disk (same size as the original) that will receive the restored image.

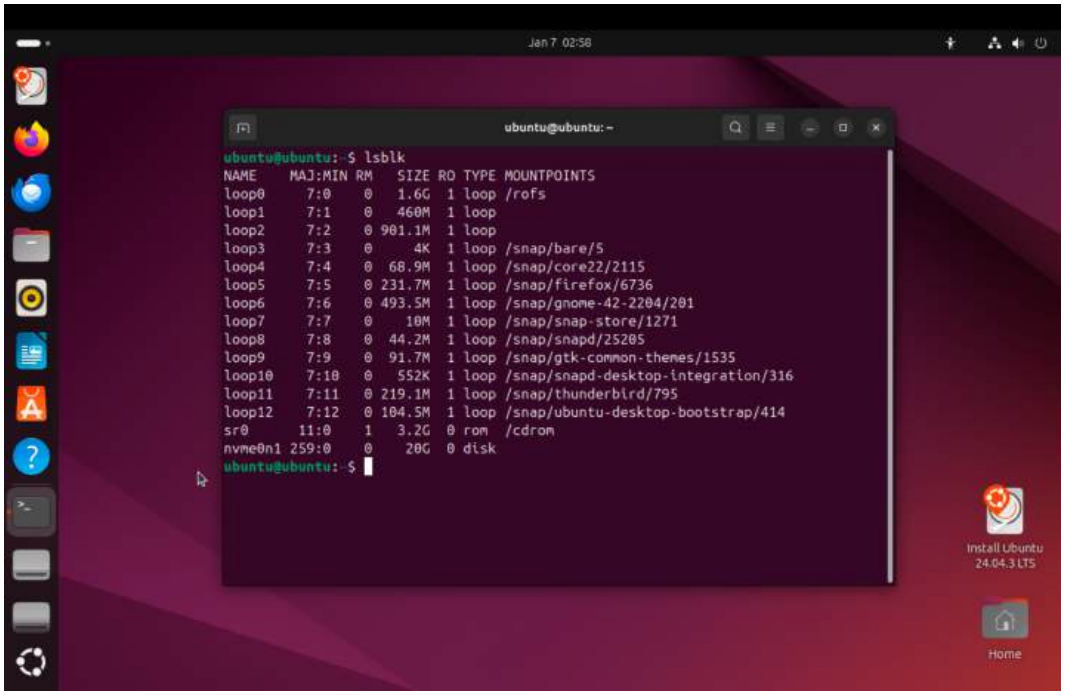
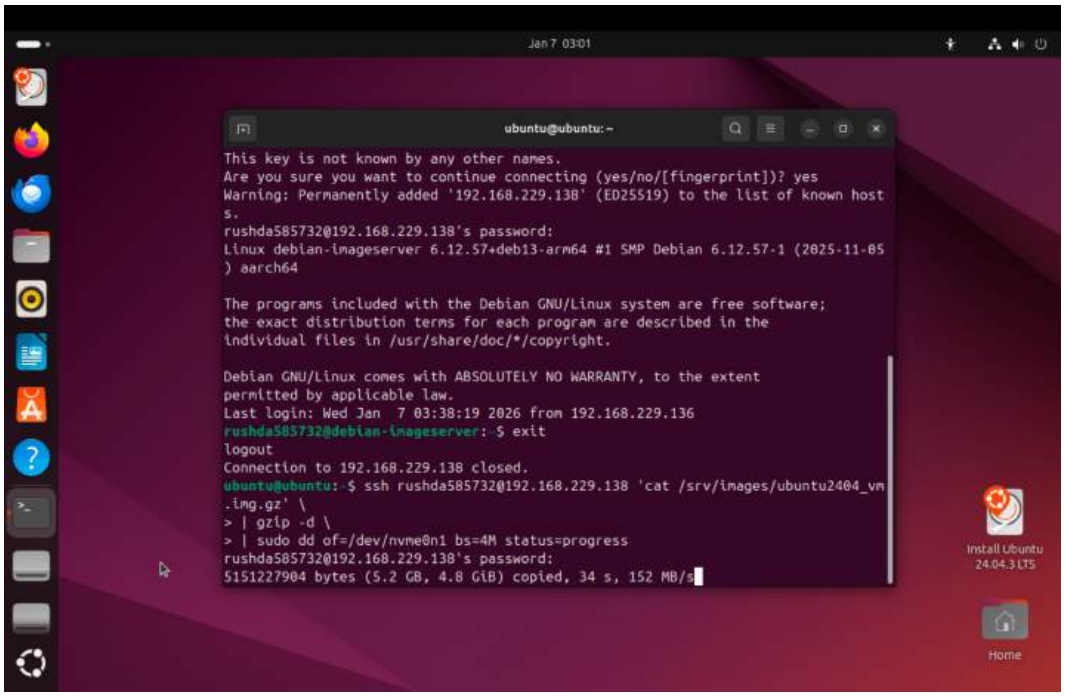


Figure 5.9-7 – Ubuntu-Restored VM Live session showing target disk with lsblk

The restore pipeline ssh user@debian 'cat /srv/images/ubuntu2404_vm.img.gz' | gzip -d | sudo dd of=/dev/… bs=4M status=progress writes the captured image back to the new VM’s disk.



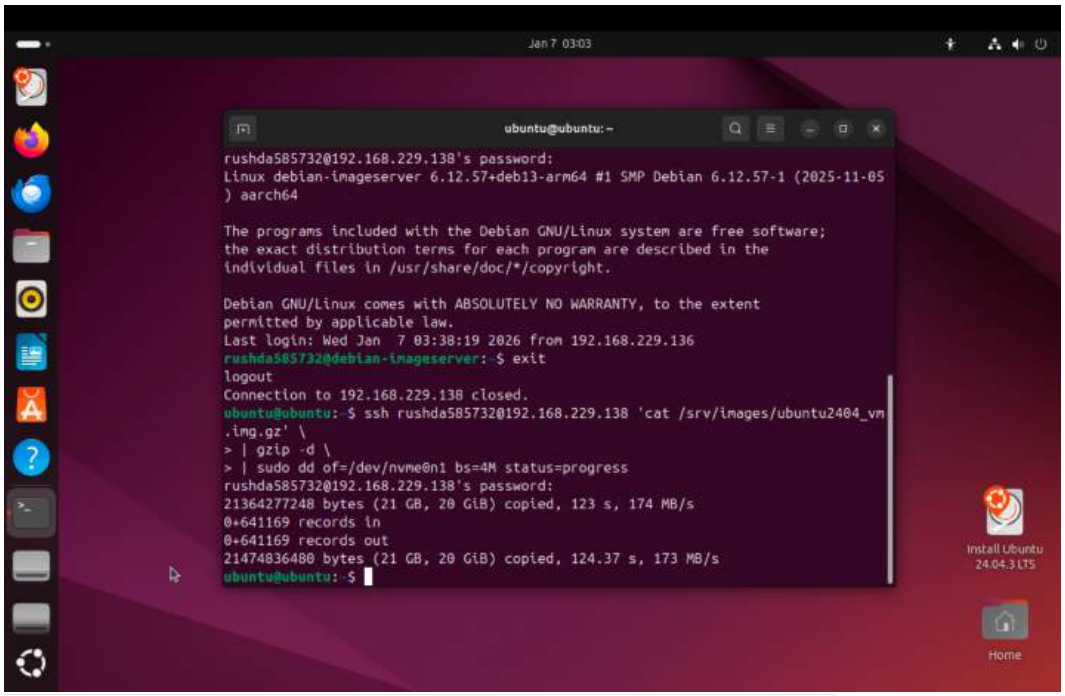


Figure 5.9-8, 9 – Restoring the image from Debian onto the new disk

On a new empty Ubuntu VM I booted into a live session and restored the captured image from the Debian server with:
ssh rushda585732@192.168.229.138 'cat /srv/images/ubuntu2404_vm.img.gz' | gzip -d | sudo dd of=/dev/nvme0n1 bs=4M status=progress.
After shutting down the live session, disconnecting the ISO and booting from the disk, the VM started normally with the restored Ubuntu system.

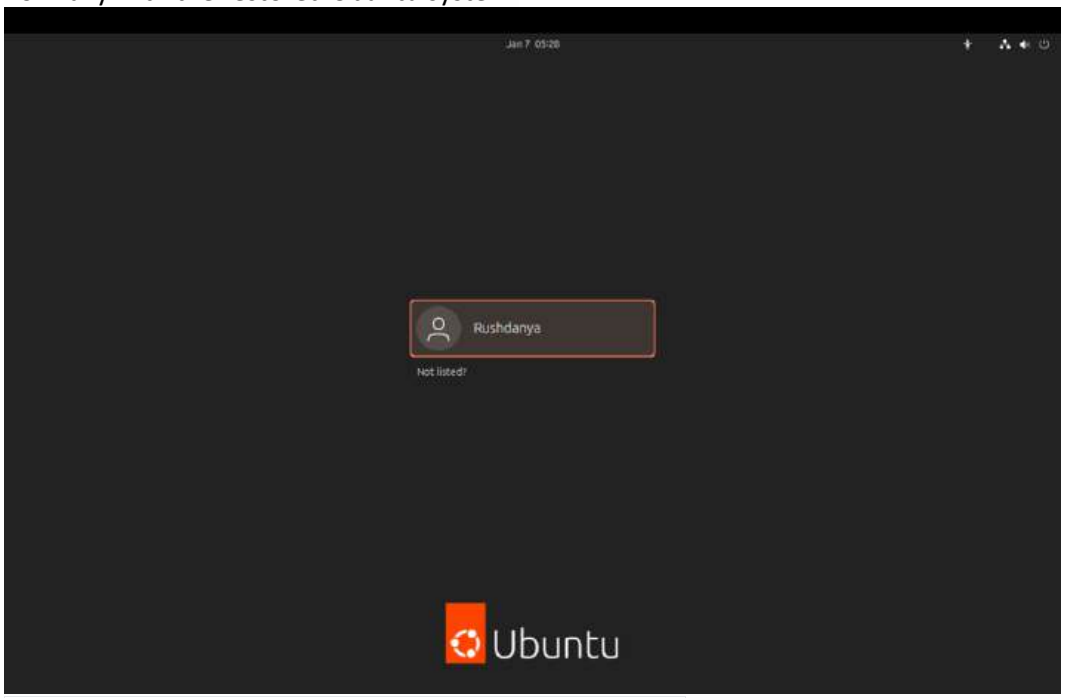


Figure 5.9-10 – Successfully booted restored Ubuntu VM