

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.

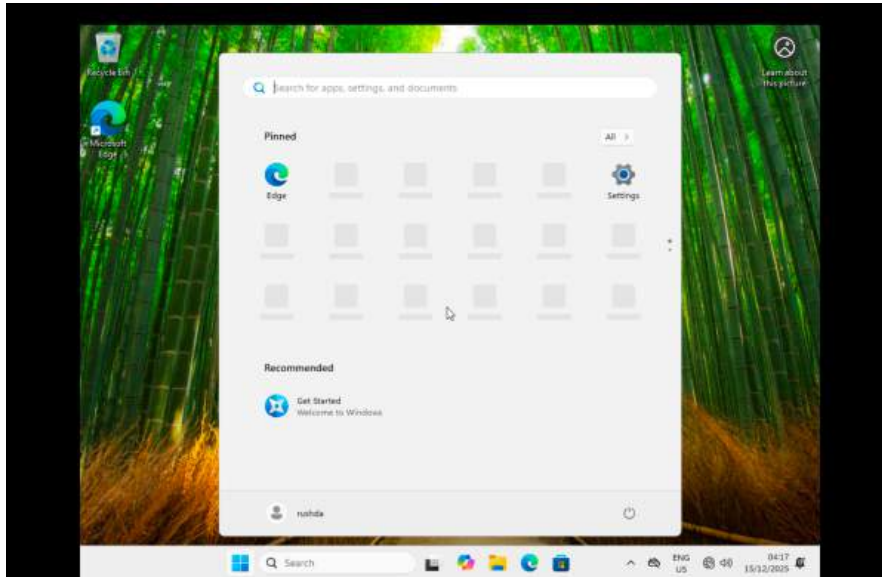


Fig 1: Command = Start Menu

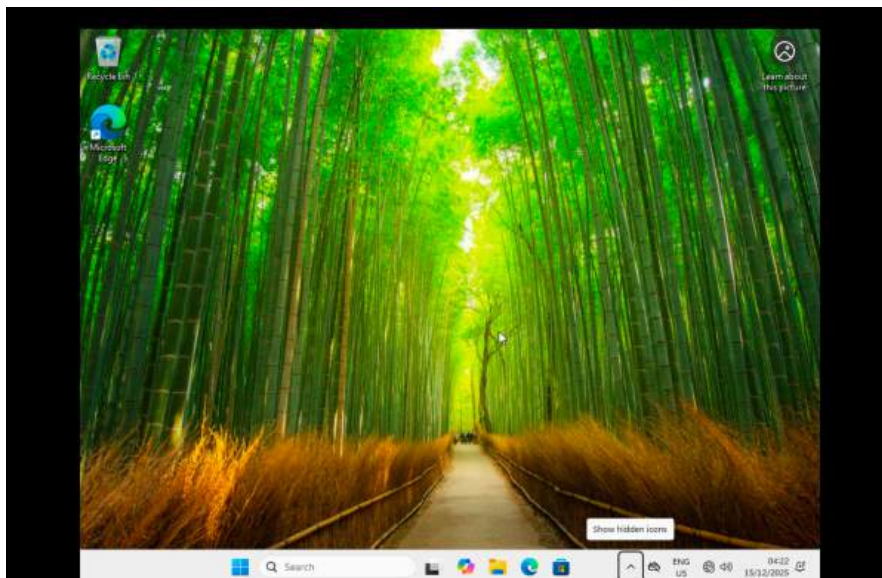


Fig 2: Command + B

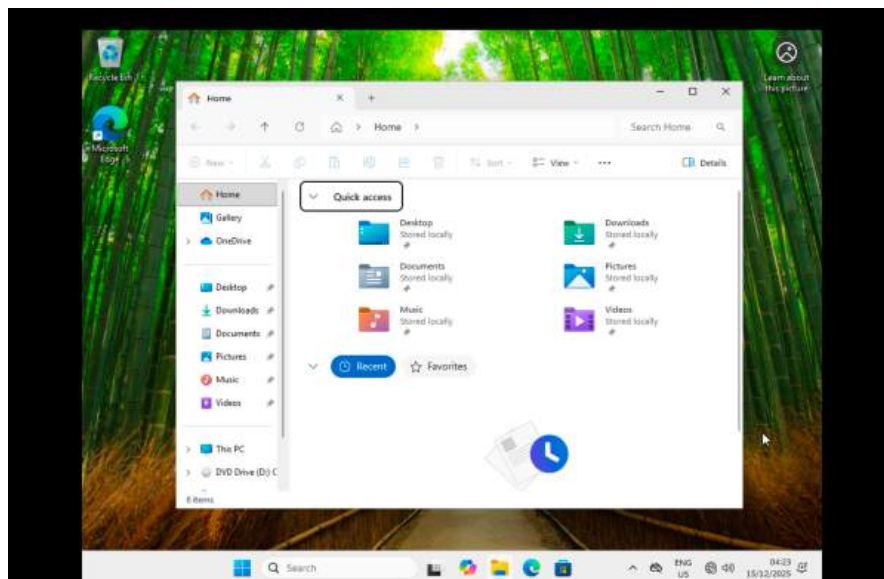


Fig 3: Command + E

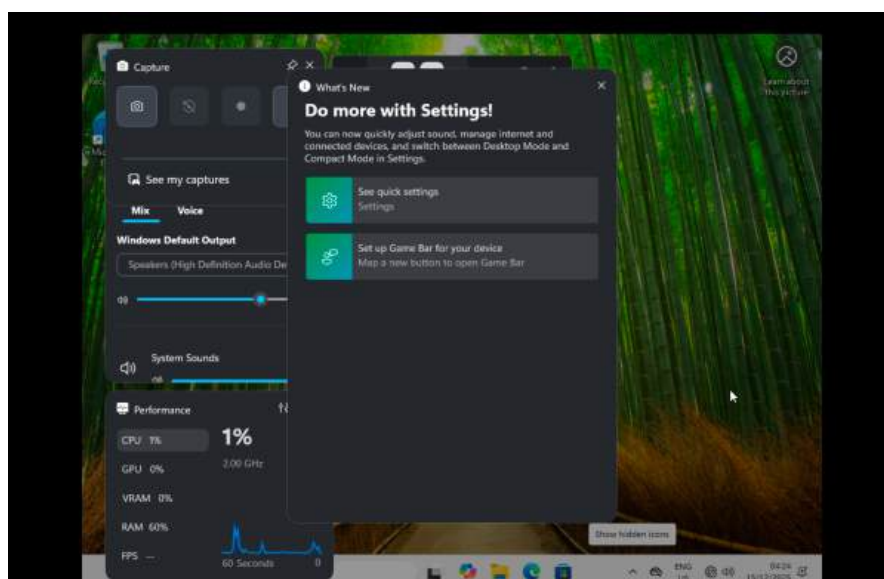


Fig 4: Command + G

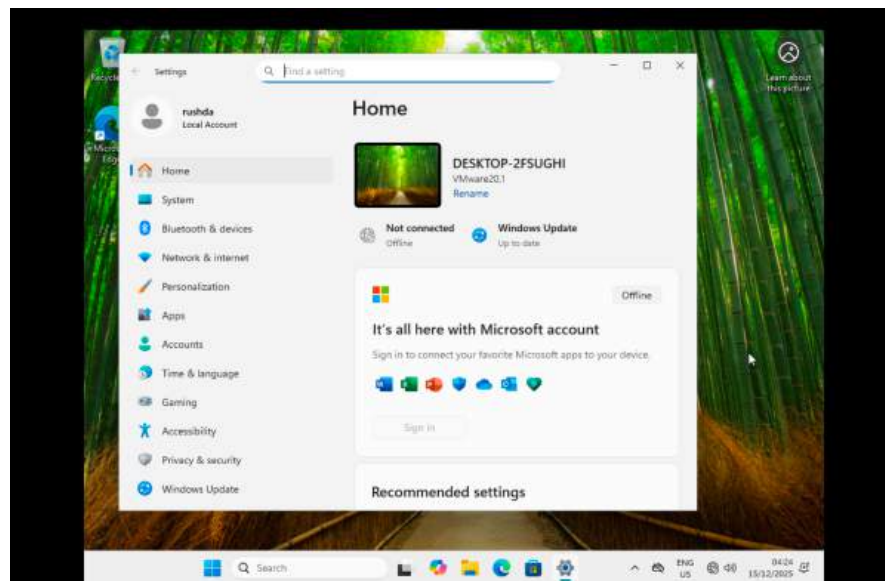


Fig 5: Command + I

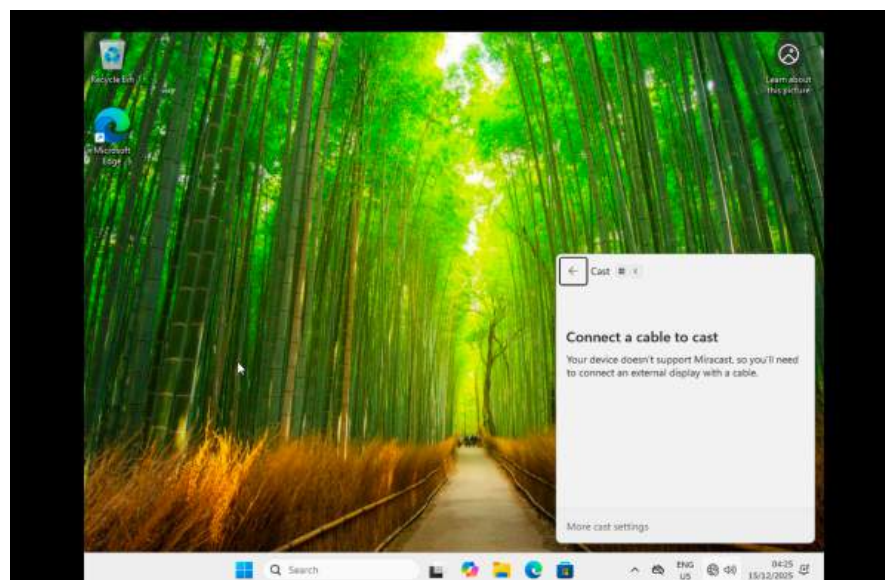


Fig 6: Command + K

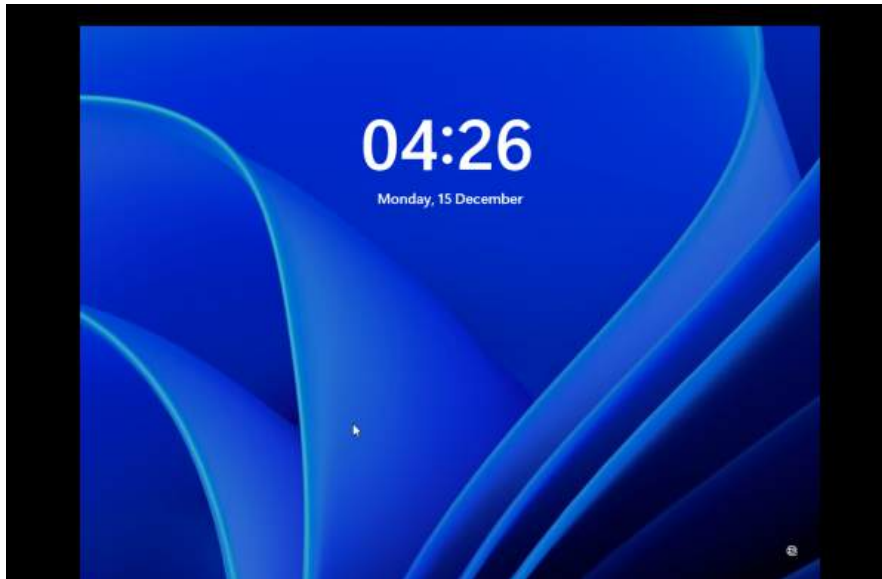


Fig 7: Command + L

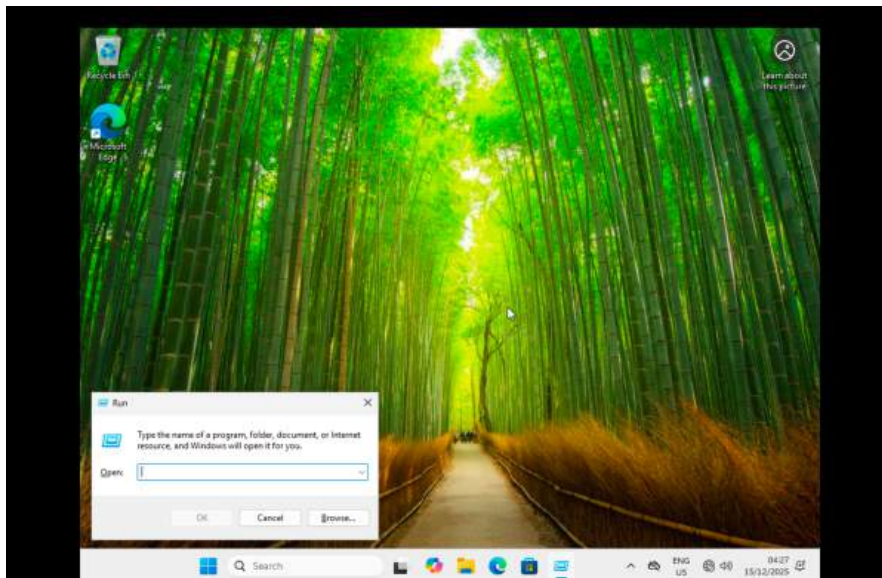

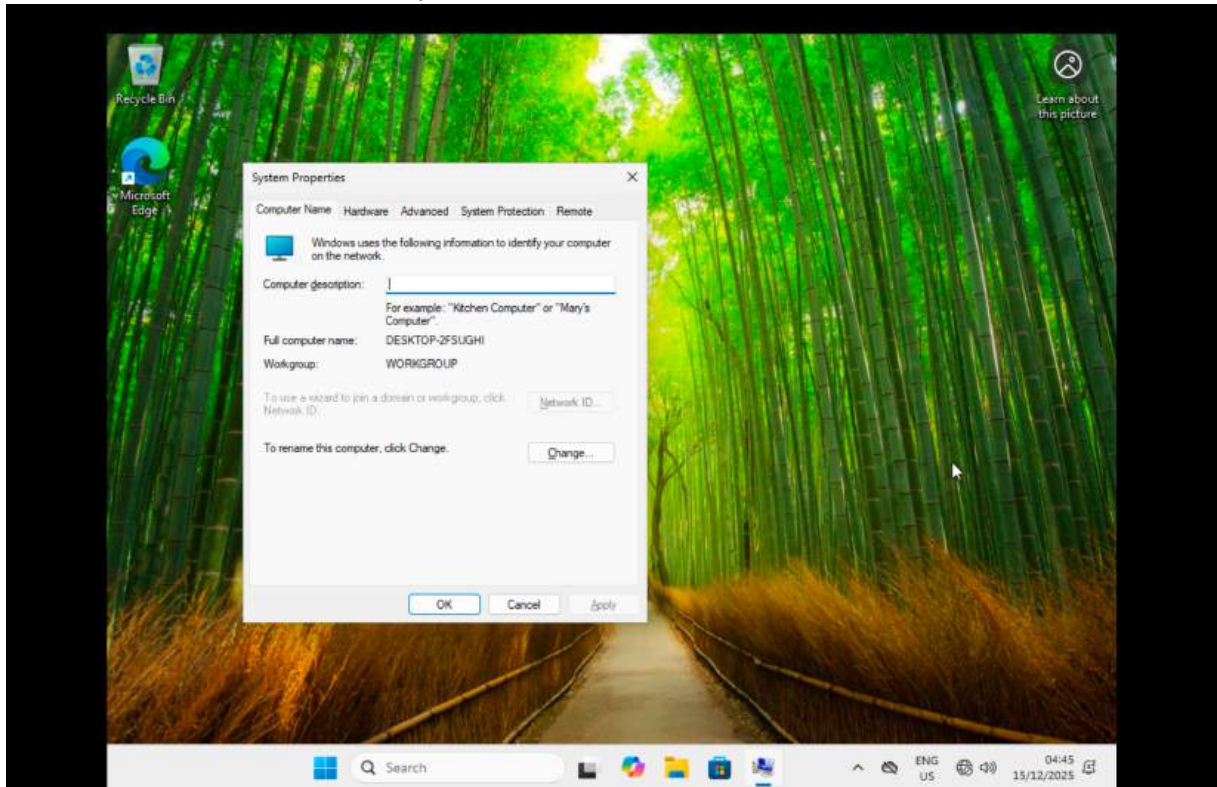


Fig 8: Command + R

- b) The file explorer can be opened with **Windows + 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.

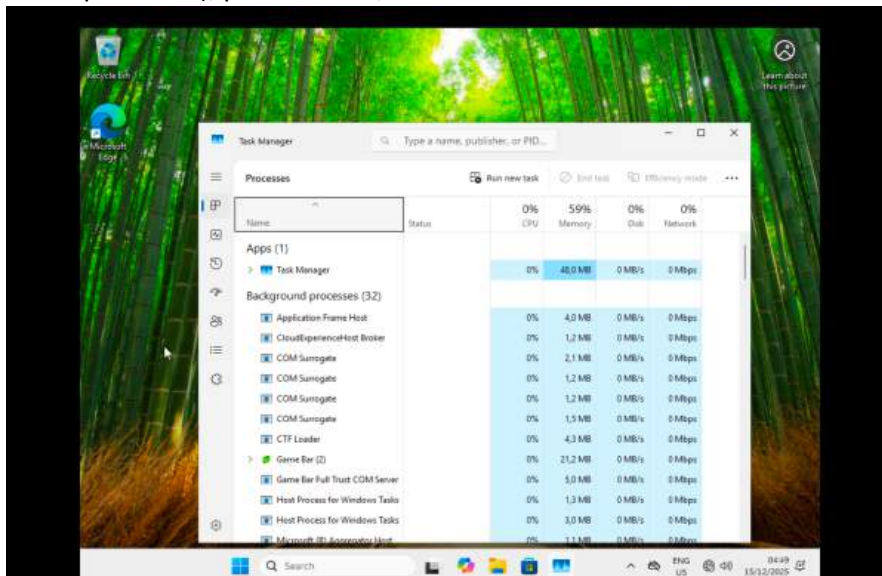


Fig 1: Processes

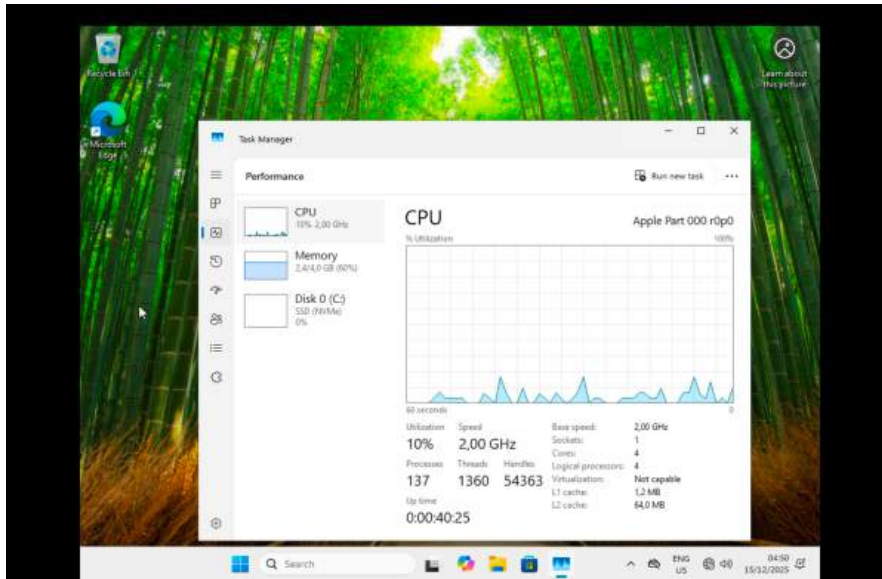


Fig 2: Performance

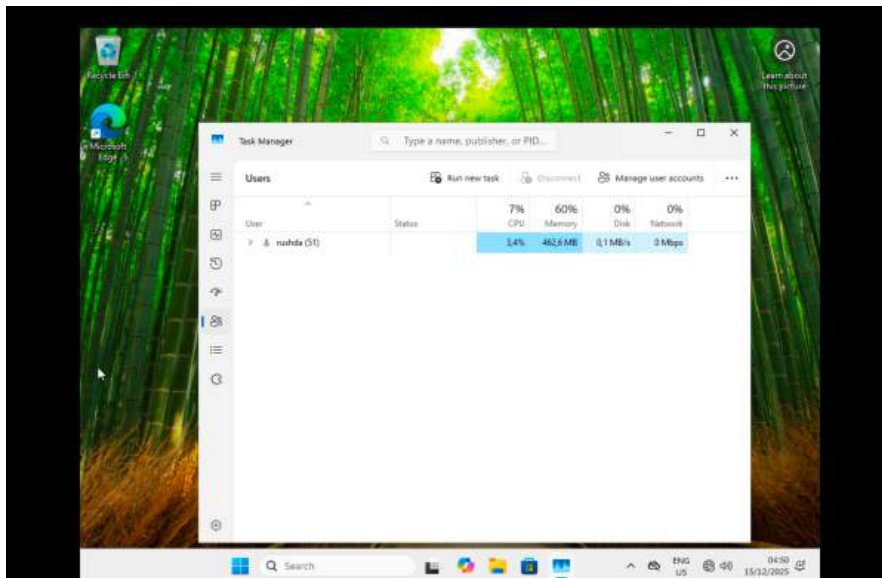


Fig 3: Users

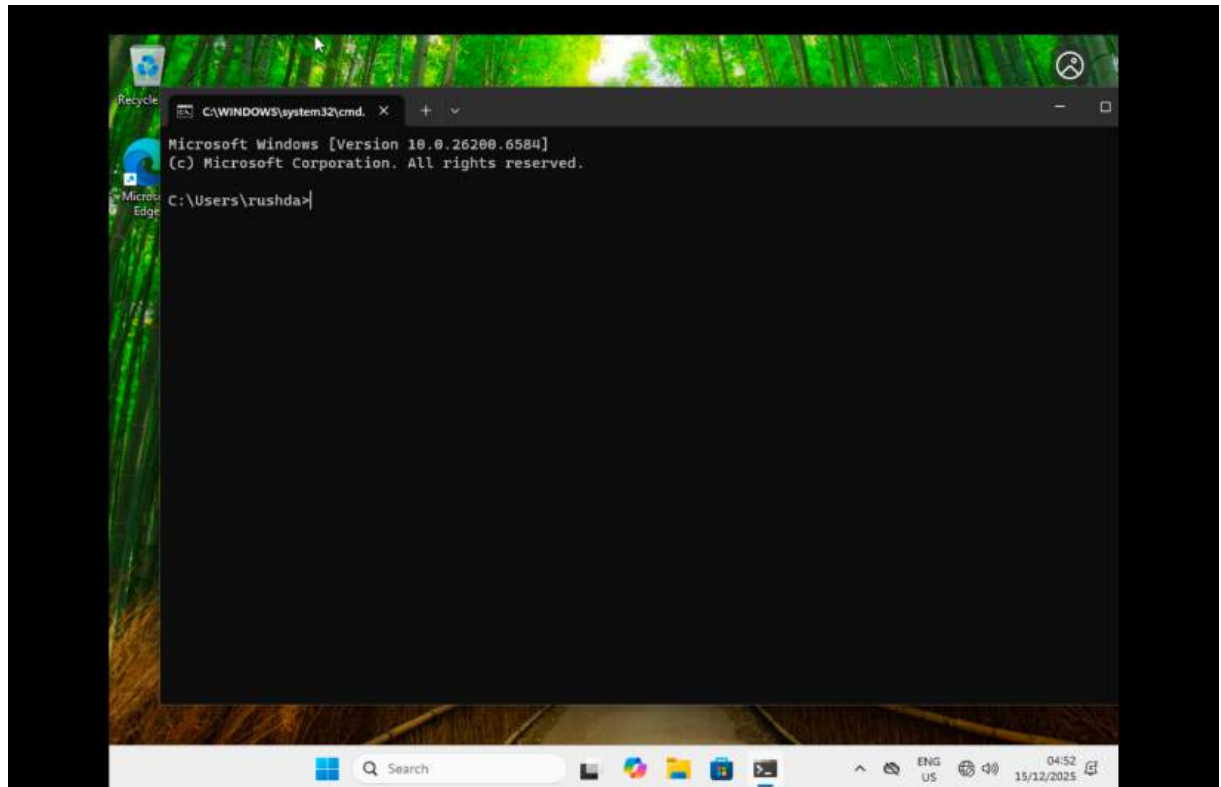
- e) If you're giving a PowerPoint presentation and you connect your laptop to a projector, Windows can use the projector as a second screen. For example, you may have Outlook open on your first screen that you don't show over the projector, while the PowerPoint presentation is displayed on the projector,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

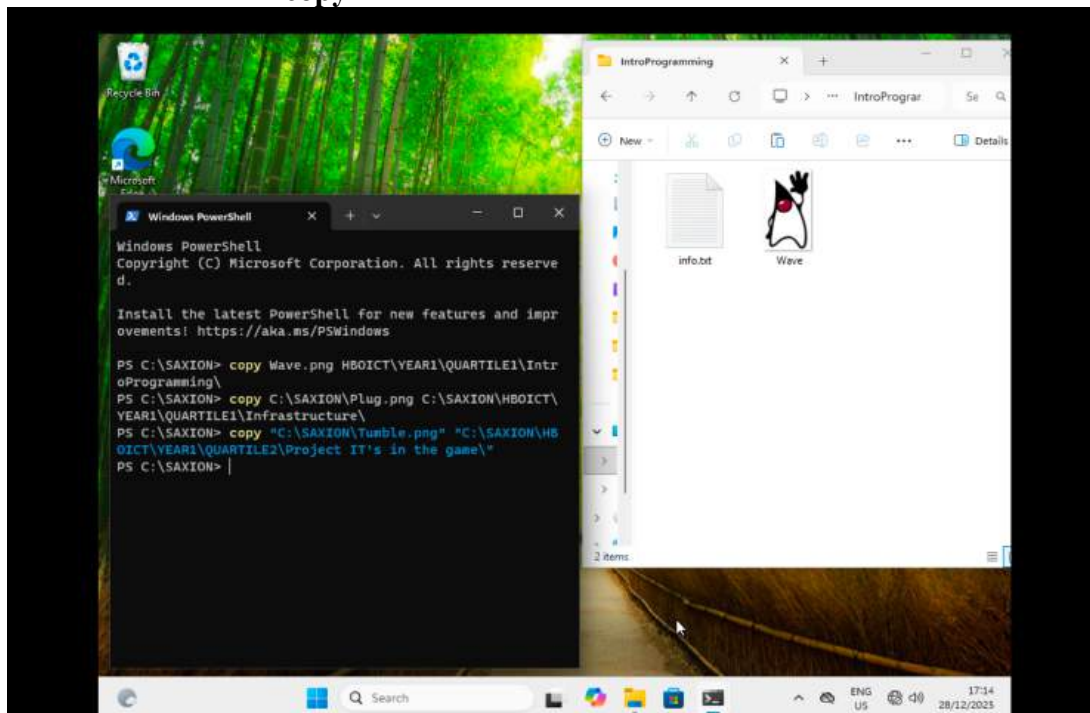
= Command + L

- 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.

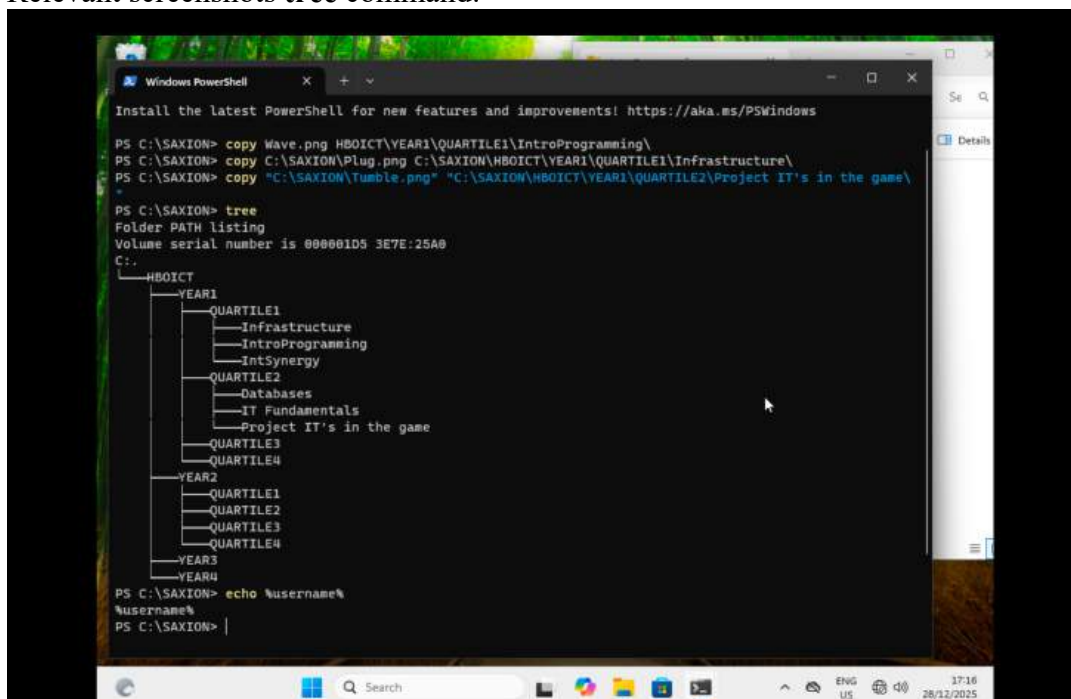


Working in the File Explorer

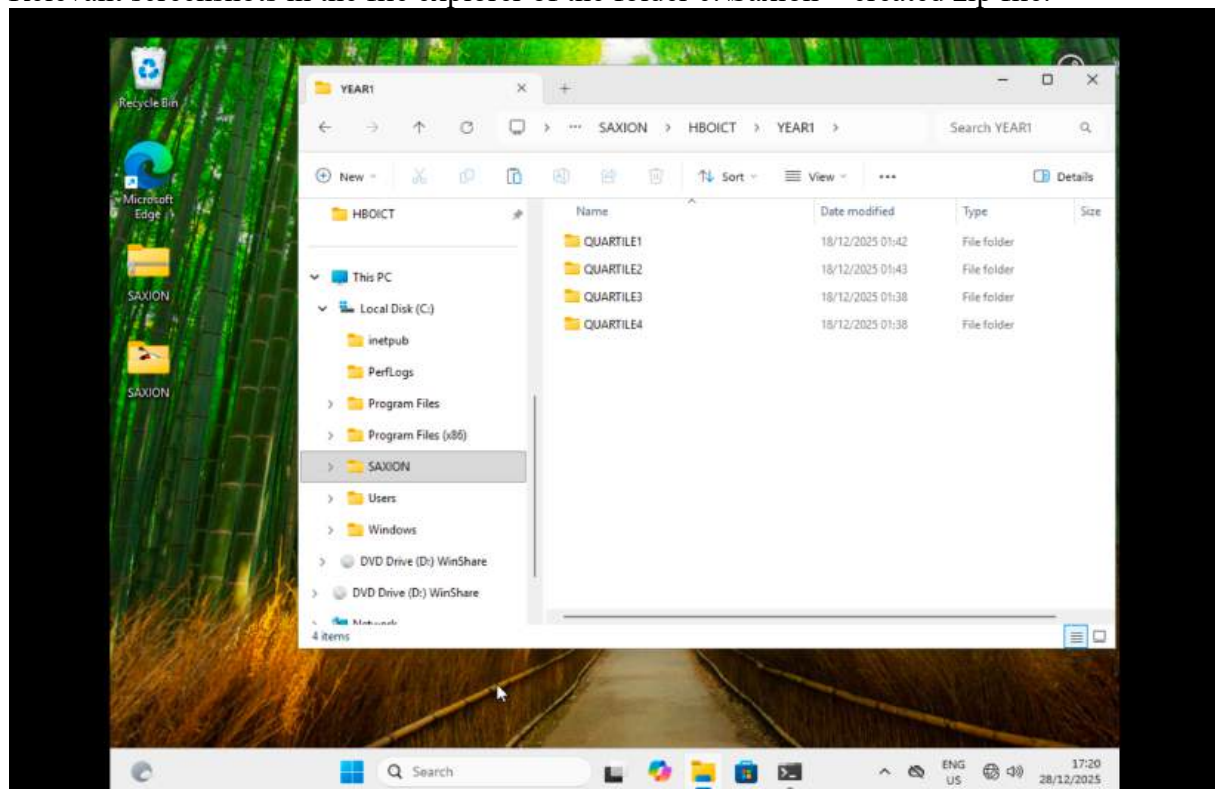
Relevant screenshots **copy** command:



Relevant screenshots **tree** command:

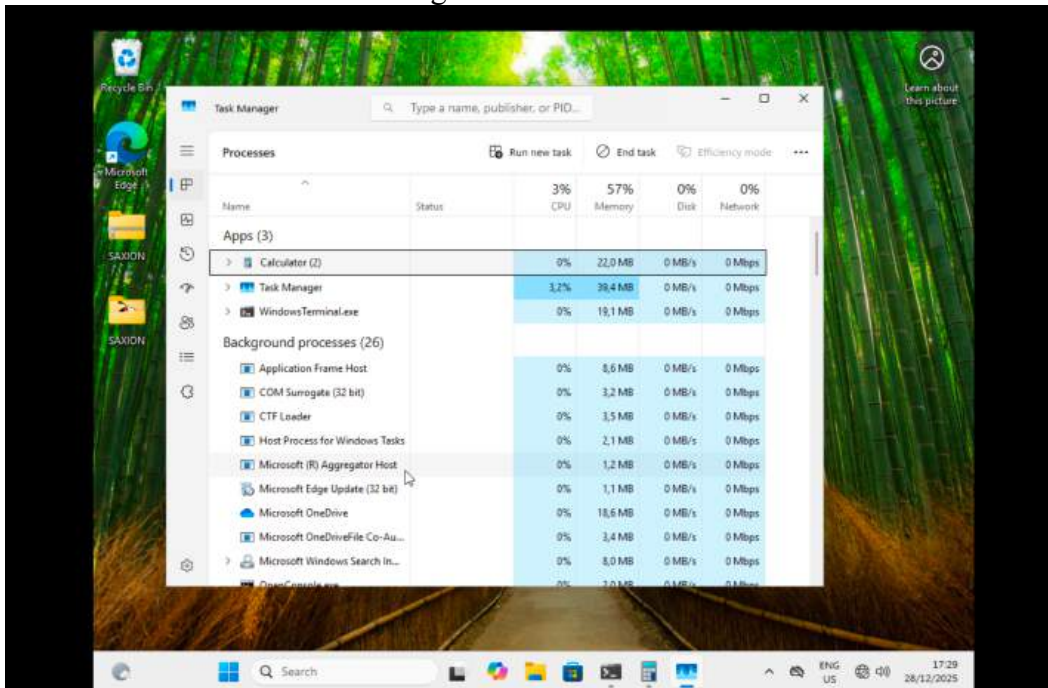


Relevant screenshots in the file explorer of the folder c:\Saxion + created zip file.



Terminating Processes

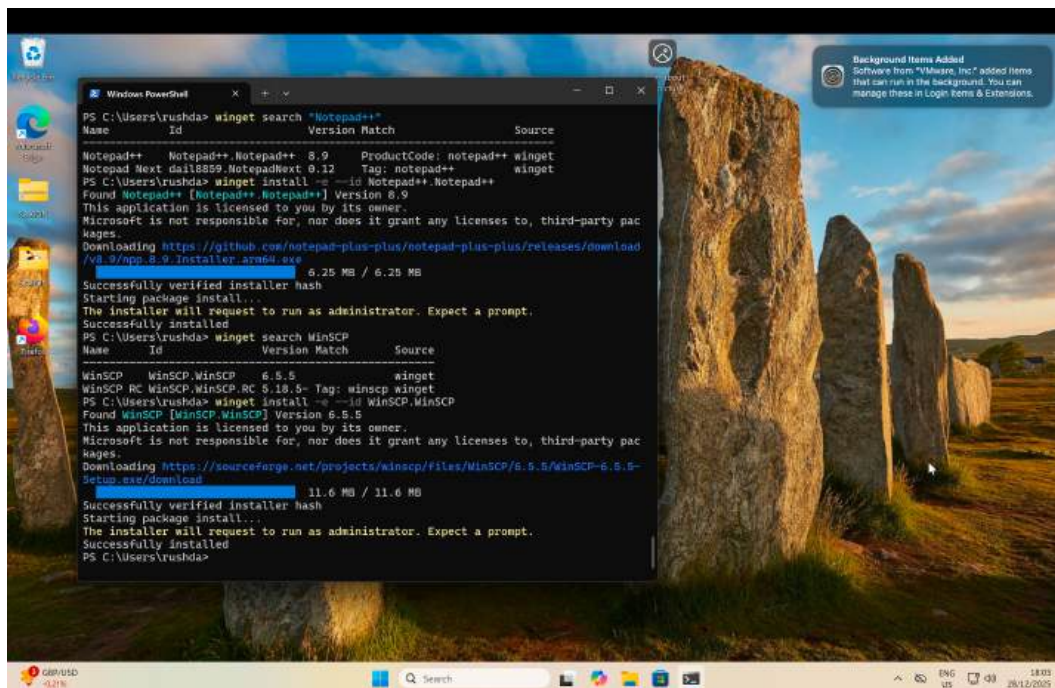
Relevant Screenshots Task Manager Window:



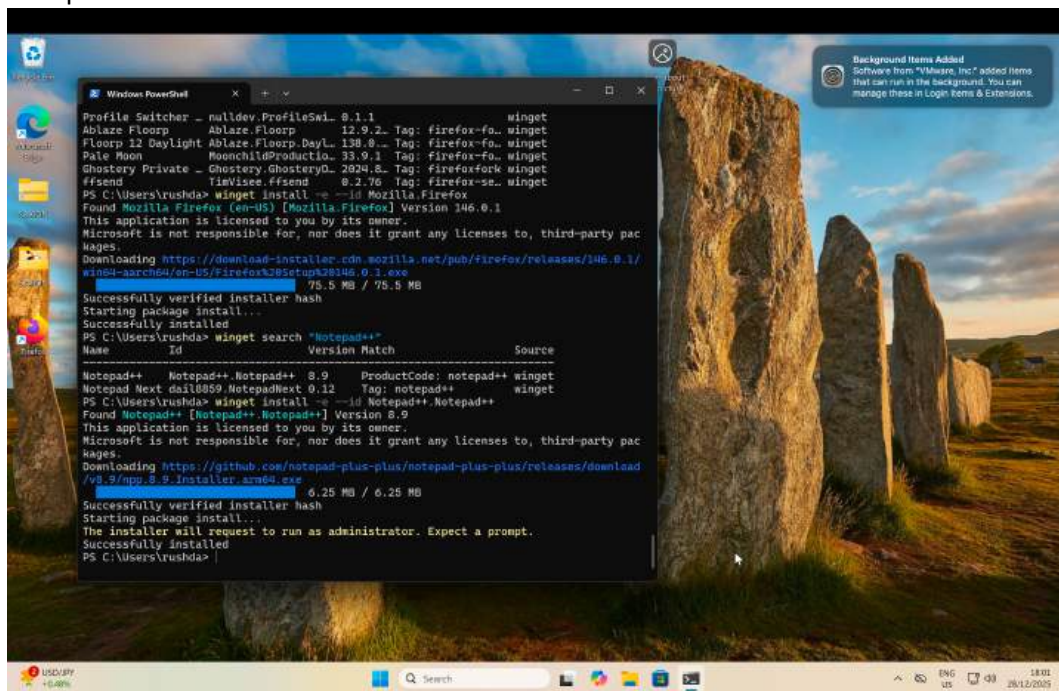
Install Software

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.

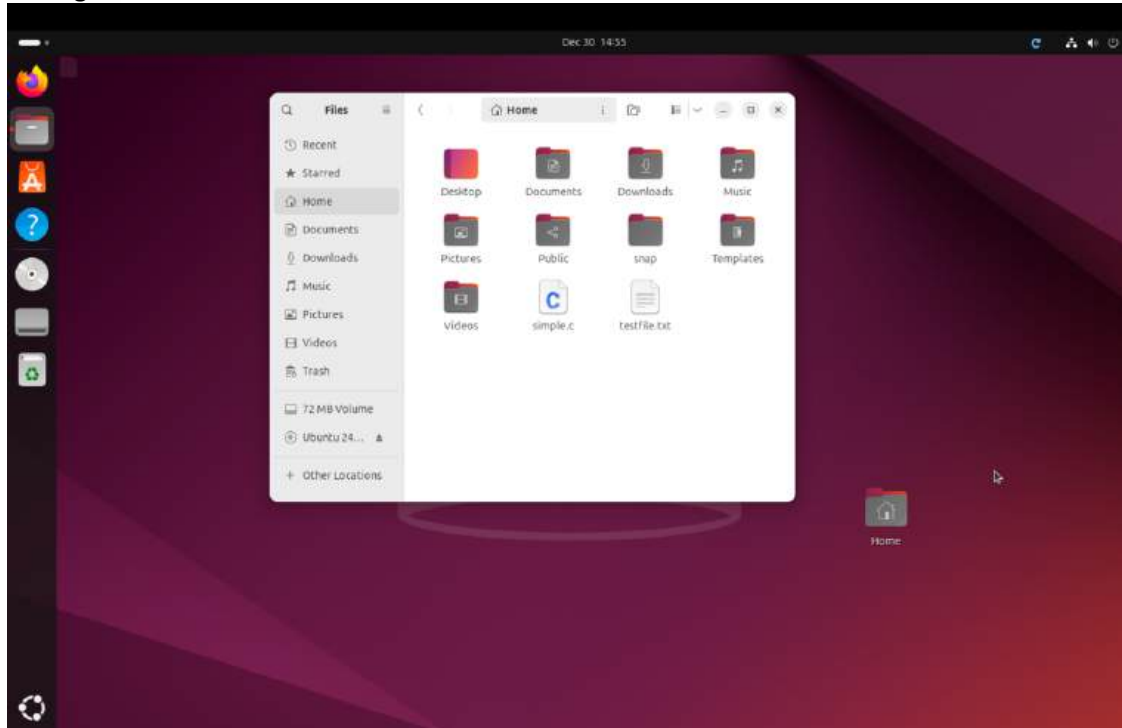


Figure 5.4-1 – Home folder with new text file

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

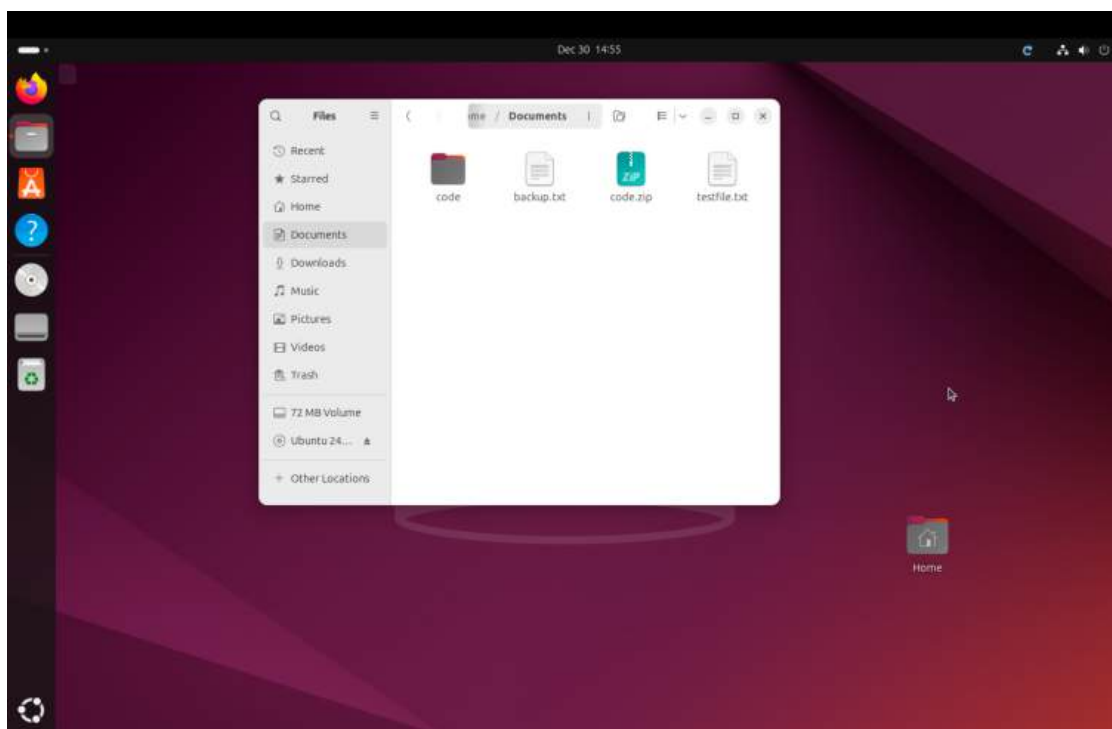


Figure 5.4-2 – `testfile.txt` folder after GUI copy

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.

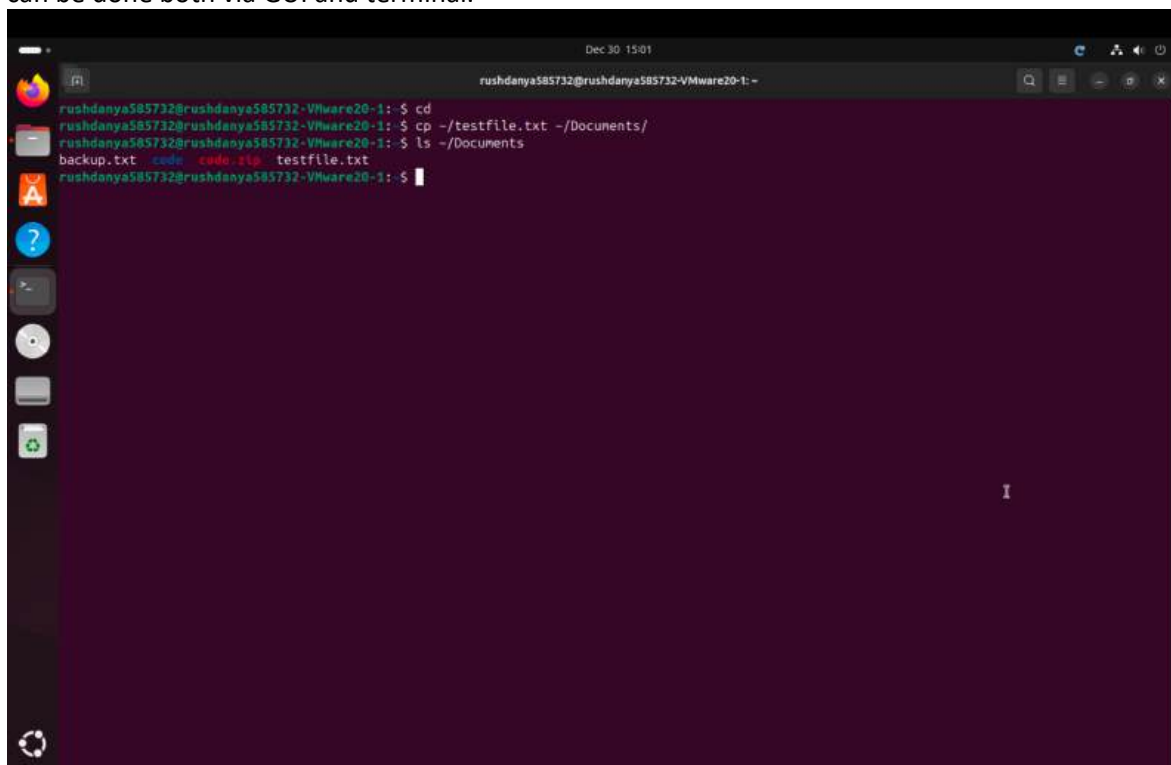


Figure 5.4-3 – Terminal copy of lab5.txt to Documents

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

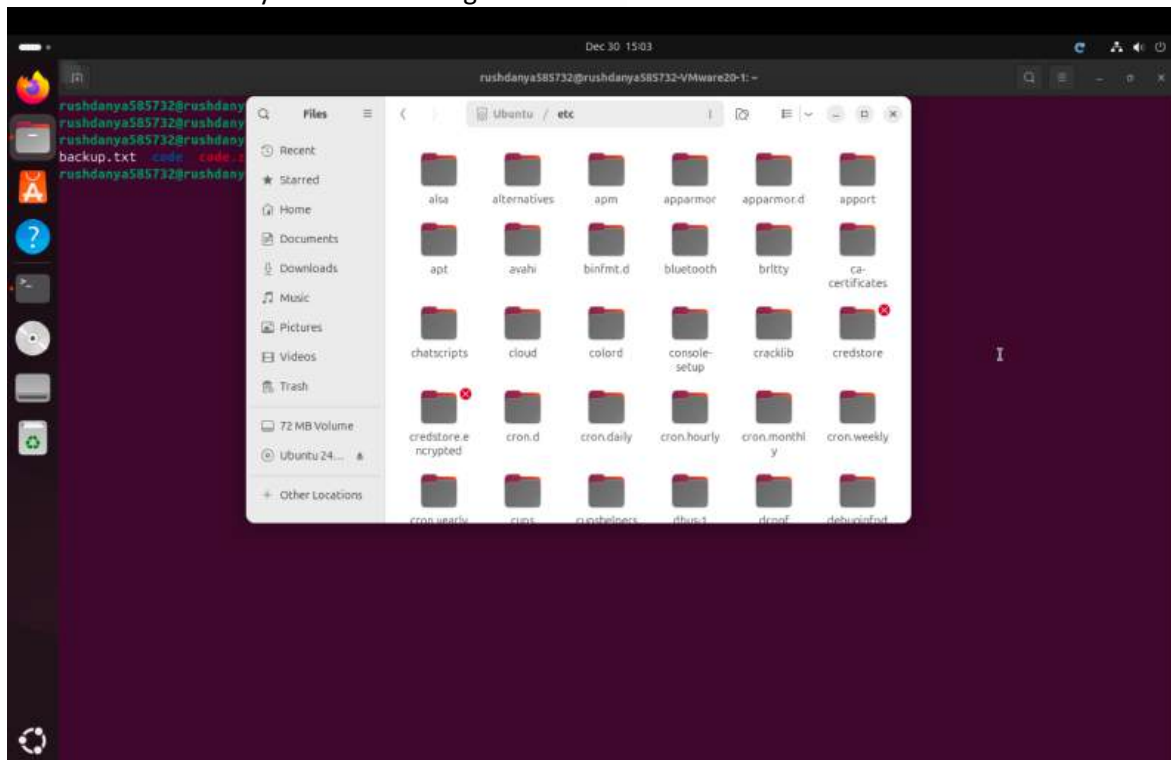
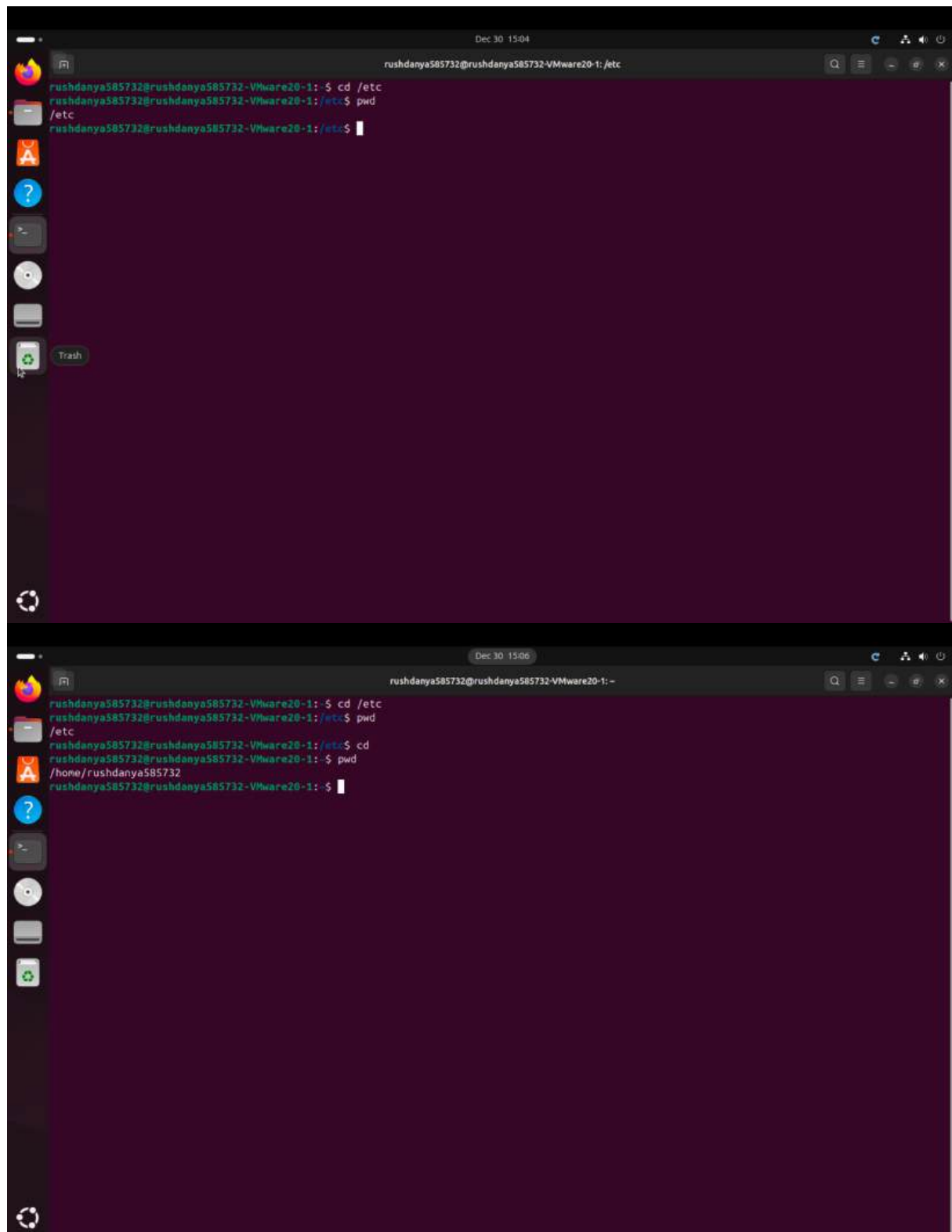


Figure 5.4-4 – `/etc` opened in the file manager

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



The image consists of two screenshots of a Linux terminal window, stacked vertically. Both screenshots show a terminal session for a user named 'rushdanya585732' on a system named 'rushdanya585732-VMware20-1'. The terminal has a dark purple background and a light-colored prompt.

The top screenshot shows the following commands and output:

```
rushdanya585732@rushdanya585732-VMware20-1: ~$ cd /etc
rushdanya585732@rushdanya585732-VMware20-1: /etc$ pwd
/etc
rushdanya585732@rushdanya585732-VMware20-1: /etc$
```

The bottom screenshot shows the following commands and output:

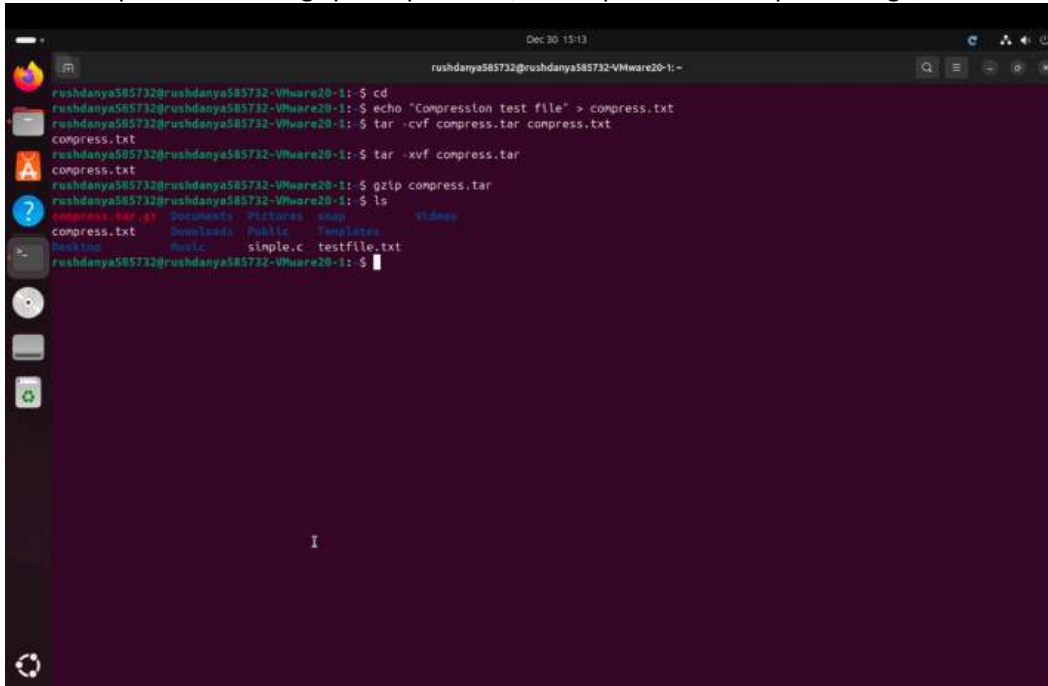
```
rushdanya585732@rushdanya585732-VMware20-1: ~$ cd /etc
rushdanya585732@rushdanya585732-VMware20-1: /etc$ pwd
/etc
rushdanya585732@rushdanya585732-VMware20-1: /etc$ cd
rushdanya585732@rushdanya585732-VMware20-1: ~$ pwd
/home/rushdanya585732
rushdanya585732@rushdanya585732-VMware20-1: ~$
```

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 for the operating system and services.

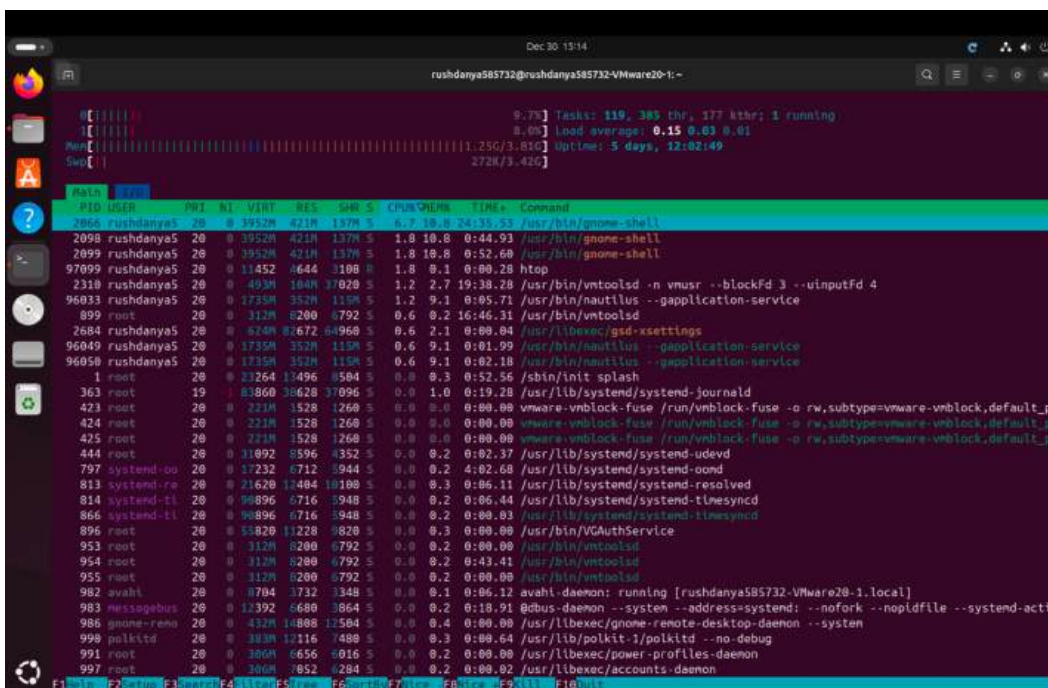
In this screenshot I created a tar archive of testfile.txt using tar -cvf compress.tar compress.txt and then compressed it with gzip compress.tar, which produced compress.tar.gz.



```
Dec 30 15:13
rushdanya585732@rushdanya585732-Vmware20-1:~$ cd
rushdanya585732@rushdanya585732-Vmware20-1:~$ echo "Compression test file" > compress.txt
rushdanya585732@rushdanya585732-Vmware20-1:~$ tar -cvf compress.tar compress.txt
compress.txt
rushdanya585732@rushdanya585732-Vmware20-1:~$ tar -xvf compress.tar
compress.txt
rushdanya585732@rushdanya585732-Vmware20-1:~$ gzip compress.tar
rushdanya585732@rushdanya585732-Vmware20-1:~$ ls
compress.tar.gz  Documents  Pictures  snap          Videos
compress.txt     Downloads  Public    Templates
Desktop          Music      simple.c  testfile.txt
rushdanya585732@rushdanya585732-Vmware20-1:~$
```

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.



```
Dec 30 15:14
rushdanya585732@rushdanya585732-Vmware20-1:~$ htop

9.7% Tasks: 119, 385 khr, 177 khr; 1 running
0.0% load average: 0.15 0.03 0.01
Mem: 1.25G/3.81G
Swap: 272K/3.42G

  PID USER   PRI NI   VIRT   RES   SHR S CPU% MEM%   TIME+  Command
2066 rushdanya5 20  0 3952M 421M 137M S  6.7 19.8 24:35.53 /usr/bin/gnome-shell
2098 rushdanya5 20  0 3952M 421M 137M S  1.8 10.8  0:44.93 /usr/bin/gnome-shell
2099 rushdanya5 20  0 3952M 421M 137M S  1.8 10.8  0:52.60 /usr/bin/gnome-shell
97099 rushdanya5 20  0 11452 4644 3108 R  1.8  0.1  0:00.28 htop
2310 rushdanya5 20  0 493M 104M 37020 S  1.2 2.7 19:38.28 /usr/bin/vmtoolsd -n vmusr --blockFd 3 --inputFd 4
96033 rushdanya5 20  0 1735M 352M 115M S  1.2 9.1  0:05.71 /usr/bin/nautilus --gapplication-service
899 root      20  0 312M  8200 6792 S  0.6 0.2 16:46.31 /usr/bin/vmtoolsd
2684 rushdanya5 20  0 624M 82672 64960 S  0.6 2.1  0:00.04 /usr/libexec/gsd-xsettings
96049 rushdanya5 20  0 1735M 352M 115M S  0.6 9.1  0:01.99 /usr/bin/nautilus --gapplication-service
96050 rushdanya5 20  0 1735M 352M 115M S  0.6 9.1  0:02.18 /usr/bin/nautilus --gapplication-service
1 root      20  0 21264 11496 8504 S  0.0 0.3  0:52.56 /sbin/init splash
363 root     19  1 8360 38628 37096 S  0.0 1.0  0:19.28 /usr/lib/systemd/systemd-journald
423 root     20  0 221M  1528 1260 S  0.0 0.0  0:00.00 vmware-vmblock-fuse /run/vmblock-fuse -o rw,subtype=vmware-vmblock,default_p
424 root     20  0 221M  1528 1260 S  0.0 0.0  0:00.00 vmware-vmblock-fuse /run/vmblock-fuse -o rw,subtype=vmware-vmblock,default_p
425 root     20  0 221M  1528 1260 S  0.0 0.0  0:00.00 vmware-vmblock-fuse /run/vmblock-fuse -o rw,subtype=vmware-vmblock,default_p
444 root     20  0 31092 8596 4352 S  0.0 0.2  0:02.37 /usr/lib/systemd/systemd-udev
797 systemd-co 20  0 17232 6712 3944 S  0.0 0.2  4:02.68 /usr/lib/systemd/systemd-oomd
813 systemd-ra 20  0 21620 11484 10100 S  0.0 0.3  0:06.11 /usr/lib/systemd/systemd-resolved
814 systemd-tl 20  0 90896 6716 3948 S  0.0 0.2  0:06.44 /usr/lib/systemd/systemd-timesyncd
866 systemd-tl 20  0 90896 6716 3948 S  0.0 0.2  0:00.03 /usr/lib/systemd/systemd-timesyncd
896 root     20  0 55820 11228 9820 S  0.0 0.3  0:00.00 /usr/bin/VGAuthService
953 root     20  0 312M  8200 6792 S  0.0 0.2  0:00.00 /usr/bin/vmtoolsd
954 root     20  0 312M  8200 6792 S  0.0 0.2  0:43.41 /usr/bin/vmtoolsd
955 root     20  0 312M  8200 6792 S  0.0 0.2  0:00.00 /usr/bin/vmtoolsd
982 avahi     20  0 8704 3732 1348 S  0.0 0.1  0:06.12 avahi-daemon: running [rushdanya585732-Vmware20-1.local]
983 messagebus 20  0 12392 6680 3864 S  0.0 0.2  0:18.91 @dbus-daemon --system --address=systemd: --nofork --nopidfile --systemd-acti
986 gnome-remo 20  0 432M 10808 12504 S  0.0 0.4  0:00.00 /usr/libexec/gnome-remote-desktop-daemon --system
998 polkitd   20  0 383M 12116 7480 S  0.0 0.3  0:00.64 /usr/lib/polkit-1/polkitd --no-debug
991 root     20  0 306M 6656 6016 S  0.0 0.2  0:00.00 /usr/libexec/power-profiles-daemon
997 root     20  0 306M 3852 6284 S  0.0 0.2  0:00.02 /usr/libexec/accounts-daemon
```

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.

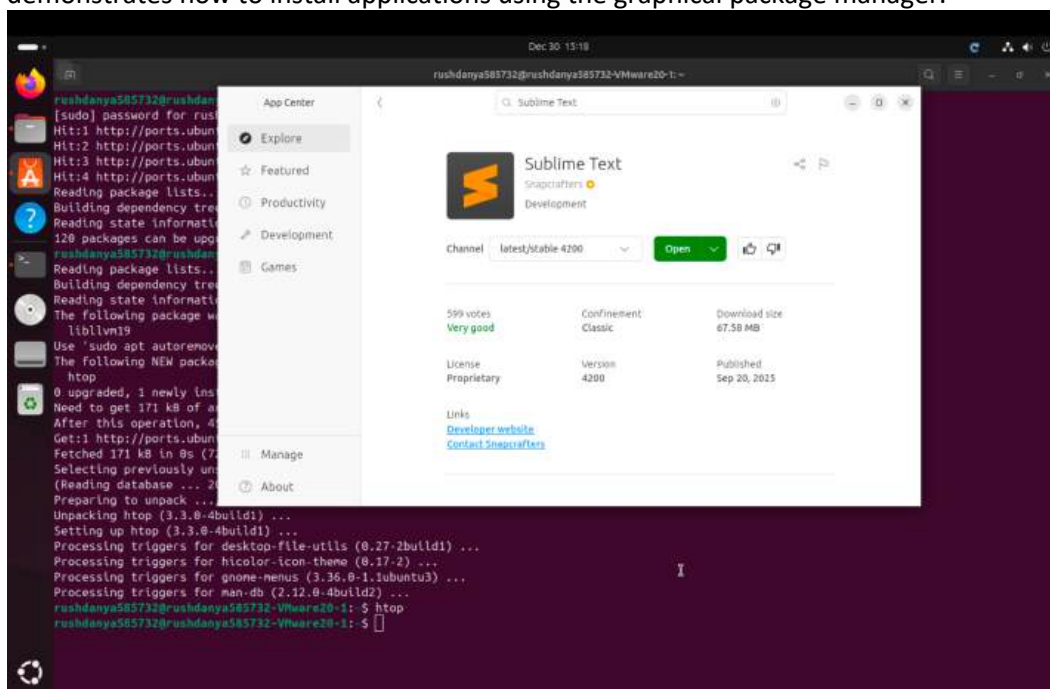


Figure 5.4-8 – Sublime Text installed via Ubuntu Software

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.

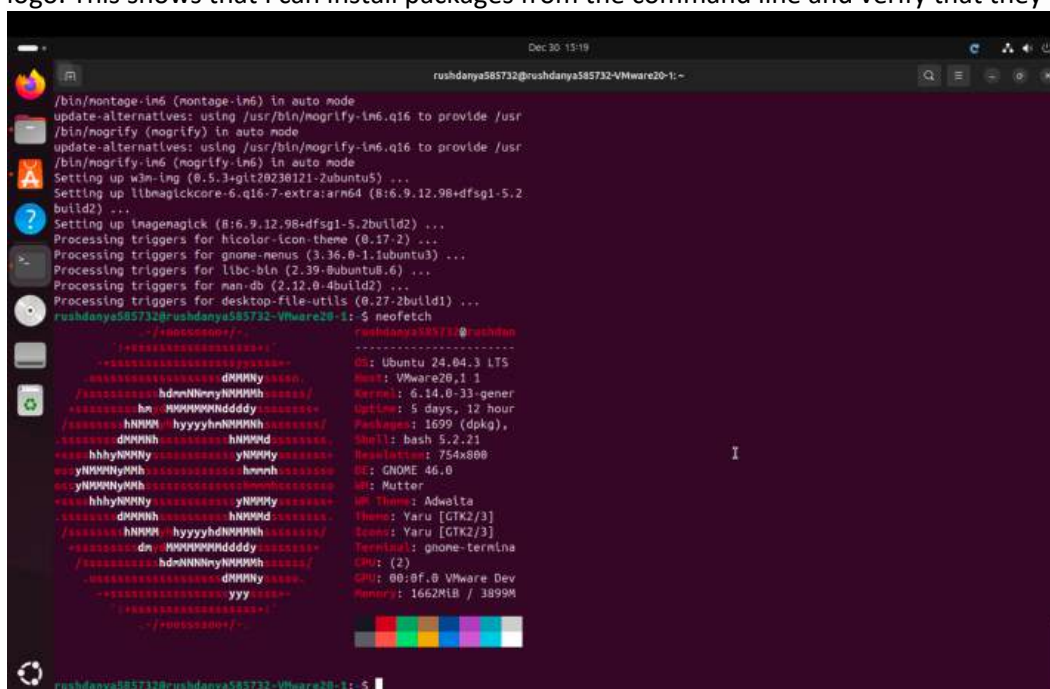


Figure 5.4-9 – neofetch output in the terminal

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` closes it.

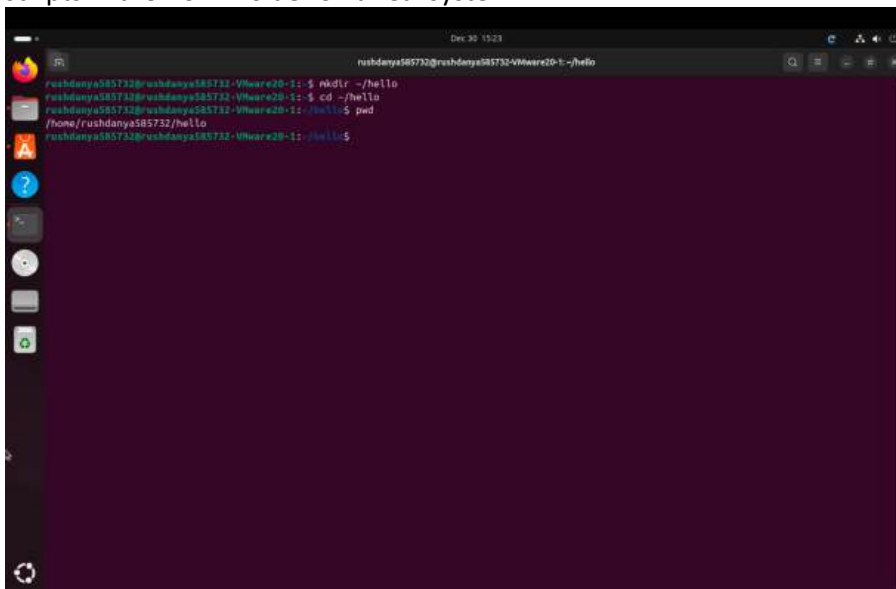
`head file`: shows the first part of a file (by default the first 10 lines).

`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.

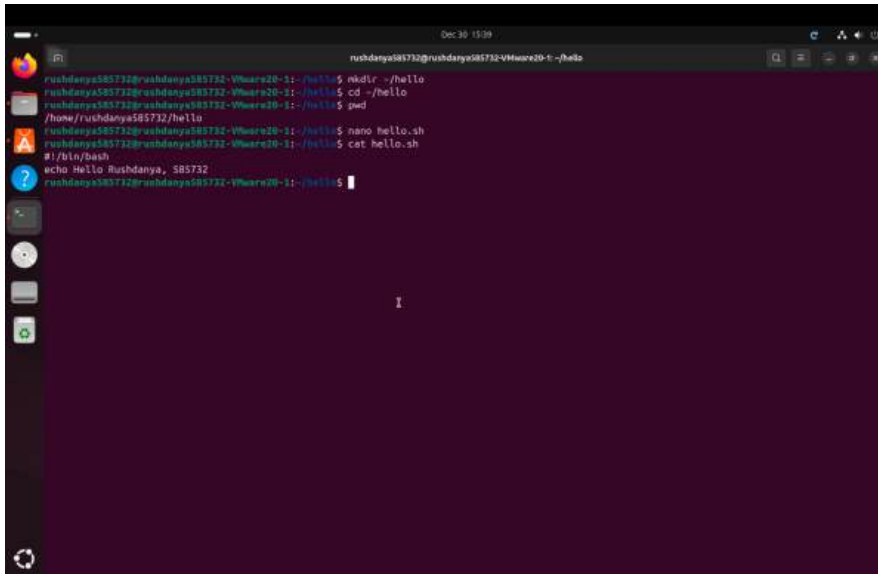
In this figure I create a new directory called `hello` in my home folder with `mkdir ~/hello` and move into it with `cd ~/hello`. This prepares a separate place for the shell script, just like you would organise scripts in their own folder on a real system.

A terminal window with a dark purple background and white text. The window title is "rushi@rushi: ~/hello". The terminal shows the following commands and output:

```
rushi@rushi:~$ mkdir ~/hello
rushi@rushi:~$ cd ~/hello
rushi@rushi:~/hello$ pwd
/home/rushi/hello
rushi@rushi:~/hello$
```

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

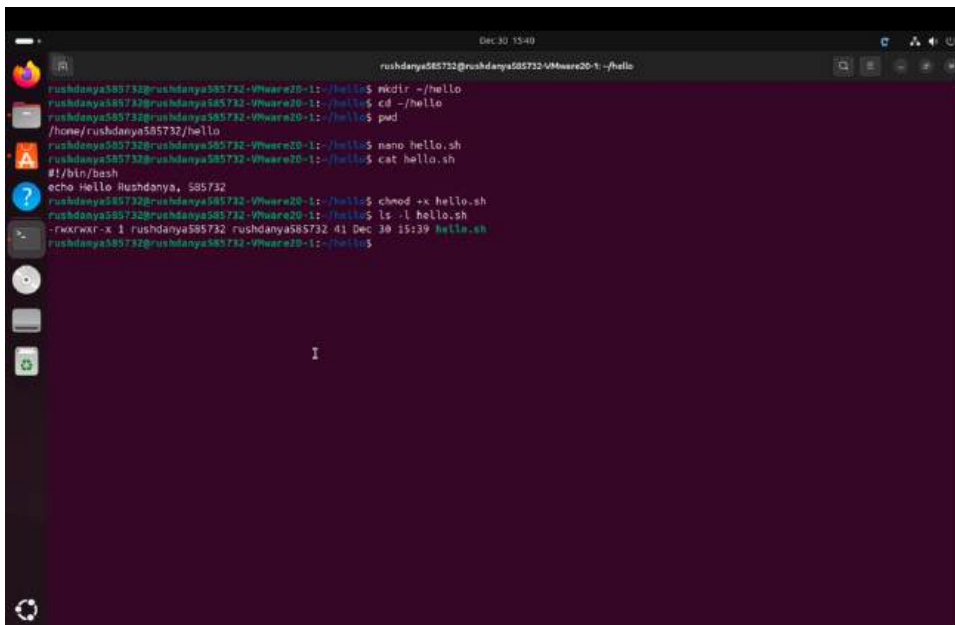
The hello.sh file contains the required bash shebang and an echo command with my name and student number.

A terminal window with a dark purple background. The prompt is 'rushdanya585732@rushdanya585732-VirtualBox: ~'. The user enters 'mkdir ~/hello', 'cd ~/hello', 'pwd' (output: '/home/rushdanya585732/hello'), 'nano hello.sh', and 'cat hello.sh'. The output of 'cat' shows the file's contents: '#!/bin/bash' followed by 'echo Hello Rushdanya, 585732' on the next line. The cursor is at the end of the second line.

```
rushdanya585732@rushdanya585732-VirtualBox: ~$ mkdir ~/hello
rushdanya585732@rushdanya585732-VirtualBox: ~$ cd ~/hello
rushdanya585732@rushdanya585732-VirtualBox: ~$ pwd
/home/rushdanya585732/hello
rushdanya585732@rushdanya585732-VirtualBox: ~$ nano hello.sh
rushdanya585732@rushdanya585732-VirtualBox: ~$ cat hello.sh
#!/bin/bash
echo Hello Rushdanya, 585732
rushdanya585732@rushdanya585732-VirtualBox: ~$
```

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 (`rw-r-xr-x`).

A terminal window showing the same sequence of commands as Figure 5.5-2, plus 'chmod +x hello.sh' and 'ls -l hello.sh'. The output of 'ls' shows the file with permissions '-rwxr-xr-x', owner 'rushdanya585732', group 'rushdanya585732', and date '41 Dec 30 15:39'. The file name is highlighted in green.

```
rushdanya585732@rushdanya585732-VirtualBox: ~$ mkdir ~/hello
rushdanya585732@rushdanya585732-VirtualBox: ~$ cd ~/hello
rushdanya585732@rushdanya585732-VirtualBox: ~$ pwd
/home/rushdanya585732/hello
rushdanya585732@rushdanya585732-VirtualBox: ~$ nano hello.sh
rushdanya585732@rushdanya585732-VirtualBox: ~$ cat hello.sh
#!/bin/bash
echo Hello Rushdanya, 585732
rushdanya585732@rushdanya585732-VirtualBox: ~$ chmod +x hello.sh
rushdanya585732@rushdanya585732-VirtualBox: ~$ ls -l hello.sh
-rwxr-xr-x 1 rushdanya585732 rushdanya585732 41 Dec 30 15:39 hello.sh
rushdanya585732@rushdanya585732-VirtualBox: ~$
```

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

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

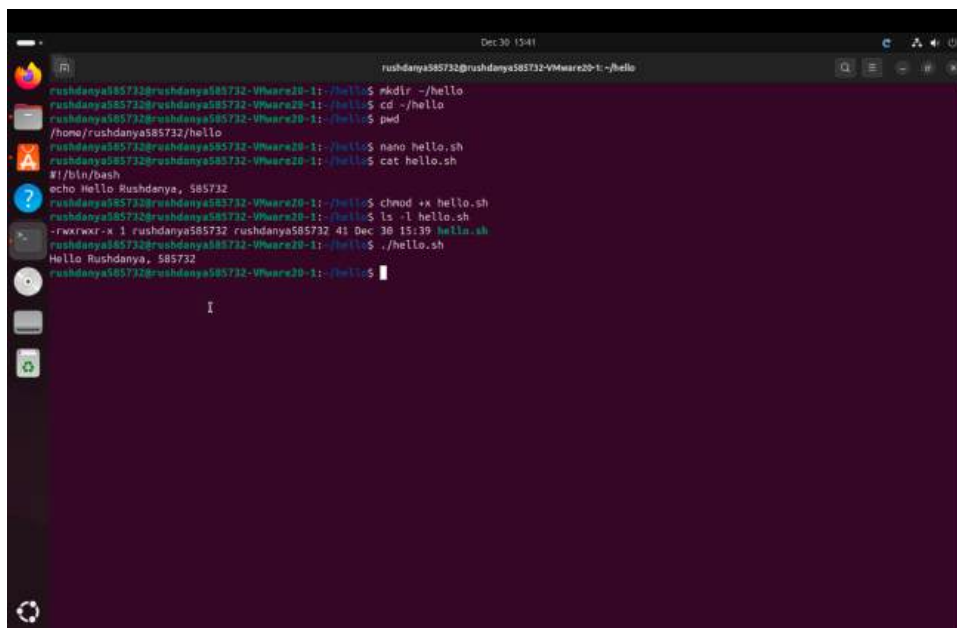
A terminal window with a dark background and light text. The prompt is `rushdanya585732@rushdanya585732-Vmware20-1: ~/hello`. The user enters `mkdir ~/hello`, `cd ~/hello`, and `pwd`, which outputs `/home/rushdanya585732/hello`. Then, `nano hello.sh` is used to create a script containing `#!/bin/bash`, `echo Hello Rushdanya, 585732`, and `chmod +x hello.sh`. The user runs `ls -l hello.sh`, showing `-rwxrwxr-x 1 rushdanya585732 rushdanya585732 41 Dec 30 15:39 hello.sh`. Finally, `./hello.sh` is executed, resulting in the output `Hello Rushdanya, 585732`.

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 (`rw-r--r--`).

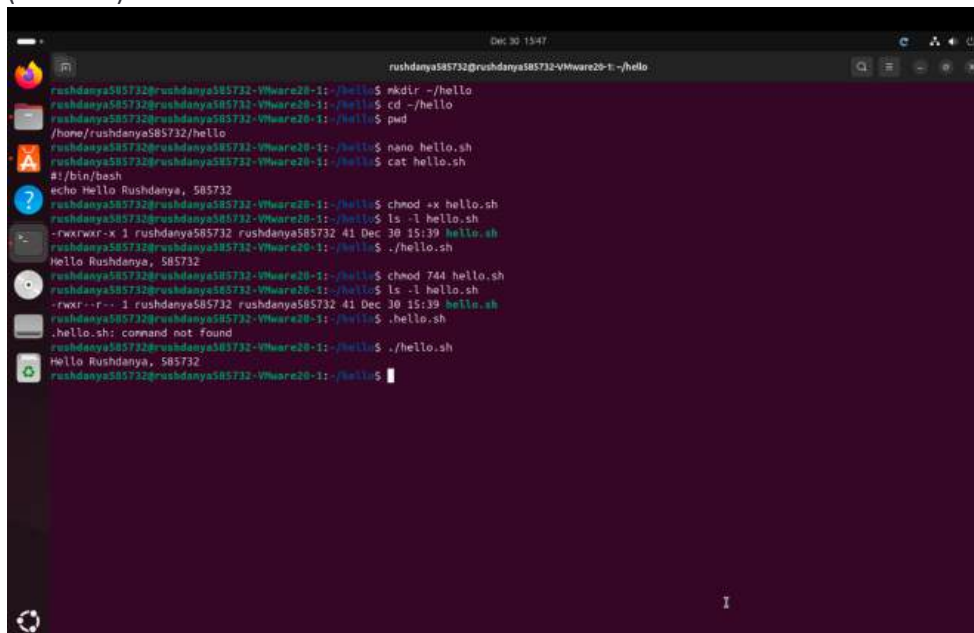
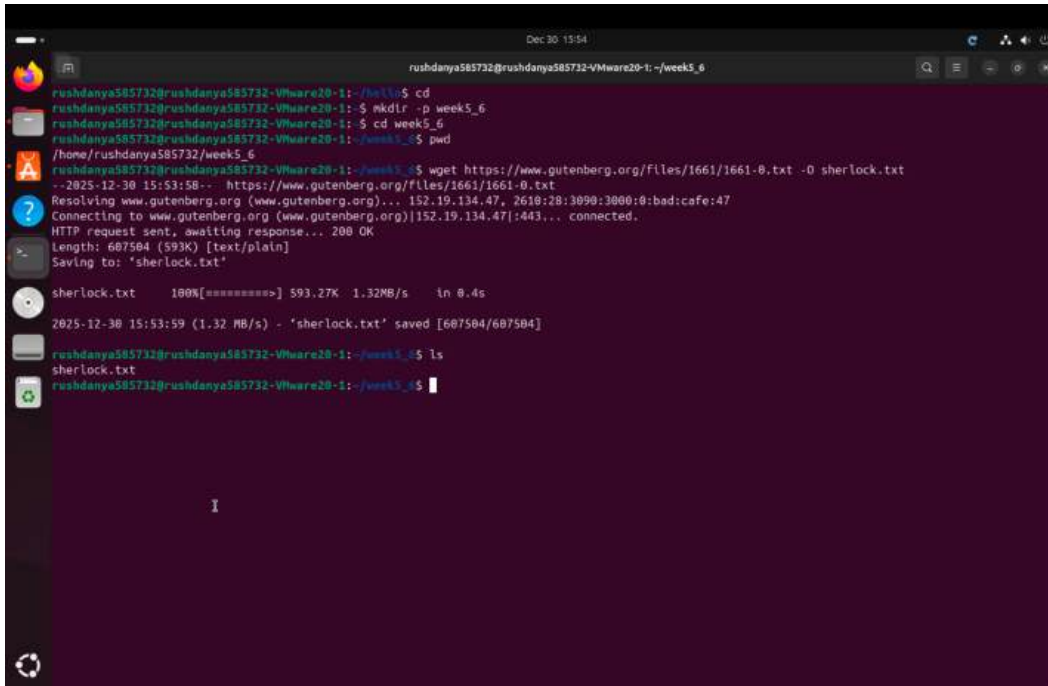
A terminal window showing the same sequence of commands as Figure 5.5-4, but with an additional step. After running `./hello.sh`, the user enters `chmod 744 hello.sh`. The subsequent `ls -l hello.sh` command now shows `-rwxr--r-- 1 rushdanya585732 rushdanya585732 41 Dec 30 15:39 hello.sh`. When `./hello.sh` is run again, it still outputs `Hello Rushdanya, 585732`.

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.



```
Dec 30 15:54
rushi@rushi:~/week5_6$ cd
rushi@rushi:~/week5_6$ mkdir -p week5_6
rushi@rushi:~/week5_6$ cd week5_6
rushi@rushi:~/week5_6$ pwd
/home/rushi/week5_6
rushi@rushi:~/week5_6$ wget https://www.gutenberg.org/files/1661/1661-0.txt -O sherlock.txt
--2025-12-30 15:53:58-- https://www.gutenberg.org/files/1661/1661-0.txt
Resolving www.gutenberg.org (www.gutenberg.org)... 152.19.134.47, 2610:28:3090:0:bad:cafe:47
Connecting to www.gutenberg.org (www.gutenberg.org)|152.19.134.47|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 607504 (593K) [text/plain]
Saving to: 'sherlock.txt'

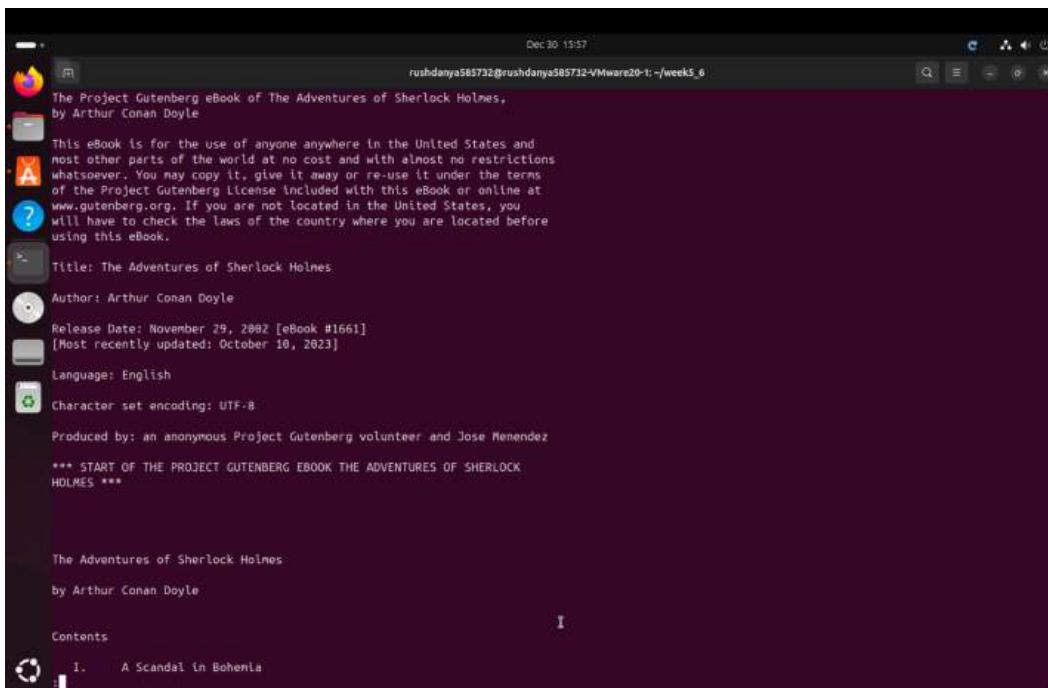
sherlock.txt      100%[=====] 593.27K  1.32MB/s   in 0.4s

2025-12-30 15:53:59 (1.32 MB/s) - 'sherlock.txt' saved [607504/607504]

rushi@rushi:~/week5_6$ ls
sherlock.txt
rushi@rushi:~/week5_6$
```

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



```
Dec 30 15:57
rushi@rushi:~/week5_6$ head sherlock.txt
The Project Gutenberg eBook of The Adventures of Sherlock Holmes,
by Arthur Conan Doyle

This eBook is for the use of anyone anywhere in the United States and
most other parts of the world at no cost and with almost no restrictions
whatsoever. You may copy it, give it away or re-use it under the terms
of the Project Gutenberg License included with this eBook or online at
www.gutenberg.org. If you are not located in the United States, you
will have to check the laws of the country where you are located before
using this eBook.

Title: The Adventures of Sherlock Holmes
Author: Arthur Conan Doyle
Release Date: November 29, 2002 [eBook #1661]
[Most recently updated: October 10, 2023]
Language: English
Character set encoding: UTF-8

Produced by: an anonymous Project Gutenberg volunteer and Jose Menendez

*** START OF THE PROJECT GUTENBERG EBOOK THE ADVENTURES OF SHERLOCK
HOLMES ***

The Adventures of Sherlock Holmes

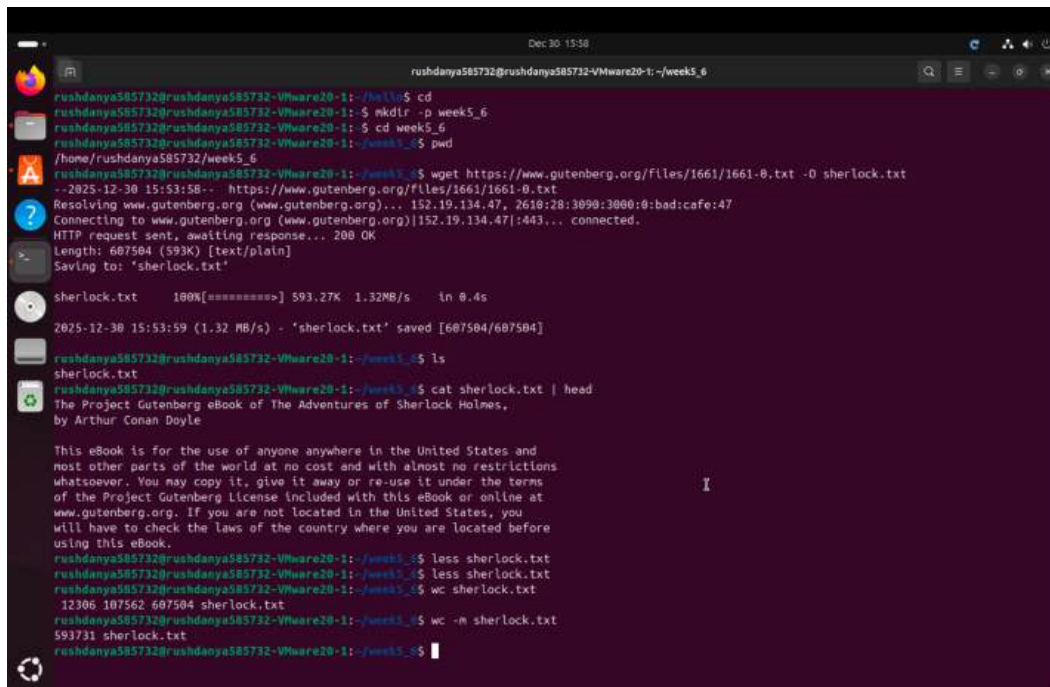
by Arthur Conan Doyle

Contents

I. A Scandal in Bohemia
```

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.

A terminal window showing the process of downloading a file from the Gutenberg website and then counting its lines, words, and characters using the `wc` command. The user creates a directory `week5_6`, changes to it, and uses `wget` to download `sherlock.txt`. After the download is complete, the user runs `ls` to confirm the file is there, then `cat sherlock.txt | head` to view the first few lines. Finally, the user runs `wc sherlock.txt` and `wc -m sherlock.txt` to get the statistics.

```
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ cd
rushdanya585732@rushdanya585732-VWware20-1: $ mkdir -p week5_6
rushdanya585732@rushdanya585732-VWware20-1: $ cd week5_6
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ pwd
/home/rushdanya585732/week5_6
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ wget https://www.gutenberg.org/files/1661/1661-0.txt -O sherlock.txt
--2025-12-30 15:53:58-- https://www.gutenberg.org/files/1661/1661-0.txt
Resolving www.gutenberg.org (www.gutenberg.org)... 152.19.134.47, 2610:28:3090:0:bad:cafe:47
Connecting to www.gutenberg.org (www.gutenberg.org)|152.19.134.47|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 607504 (593K) [text/plain]
Saving to: 'sherlock.txt'

sherlock.txt  100%[=====] 593.27K  1.32MB/s   in 0.4s

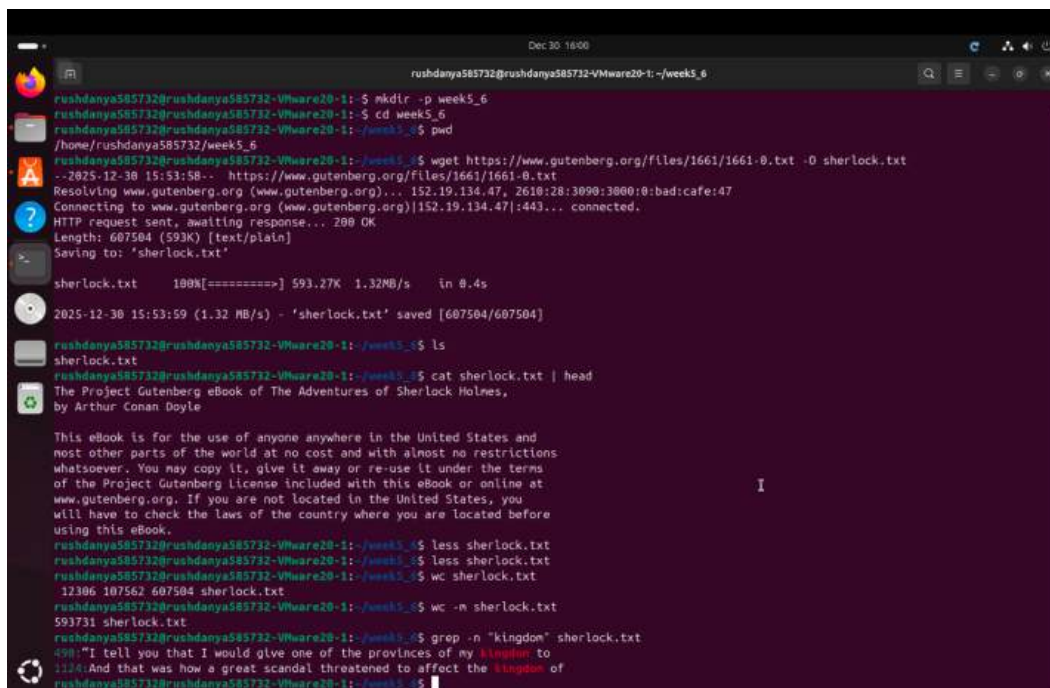
2025-12-30 15:53:59 (1.32 MB/s) - 'sherlock.txt' saved [607504/607504]

rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ ls
sherlock.txt
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ cat sherlock.txt | head
The Project Gutenberg eBook of The Adventures of Sherlock Holmes,
by Arthur Conan Doyle

This eBook is for the use of anyone anywhere in the United States and
most other parts of the world at no cost and with almost no restrictions
whatsoever. You may copy it, give it away or re-use it under the terms
of the Project Gutenberg License included with this eBook or online at
www.gutenberg.org. If you are not located in the United States, you
will have to check the laws of the country where you are located before
using this eBook.
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ less sherlock.txt
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ less sherlock.txt
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ wc sherlock.txt
12306 107562 607504 sherlock.txt
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ wc -m sherlock.txt
593731 sherlock.txt
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$
```

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.

This terminal window shows the same sequence of commands as Figure 5.6-3, but with an additional `grep` command at the end. The `grep -n "kingdom" sherlock.txt` command successfully finds two occurrences of the word 'kingdom' in the file, displaying the line numbers 490 and 1124 along with the corresponding text from the file.

```
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ mkdir -p week5_6
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ cd week5_6
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ pwd
/home/rushdanya585732/week5_6
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ wget https://www.gutenberg.org/files/1661/1661-0.txt -O sherlock.txt
--2025-12-30 15:53:58-- https://www.gutenberg.org/files/1661/1661-0.txt
Resolving www.gutenberg.org (www.gutenberg.org)... 152.19.134.47, 2610:28:3090:0:bad:cafe:47
Connecting to www.gutenberg.org (www.gutenberg.org)|152.19.134.47|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 607504 (593K) [text/plain]
Saving to: 'sherlock.txt'

sherlock.txt  100%[=====] 593.27K  1.32MB/s   in 0.4s

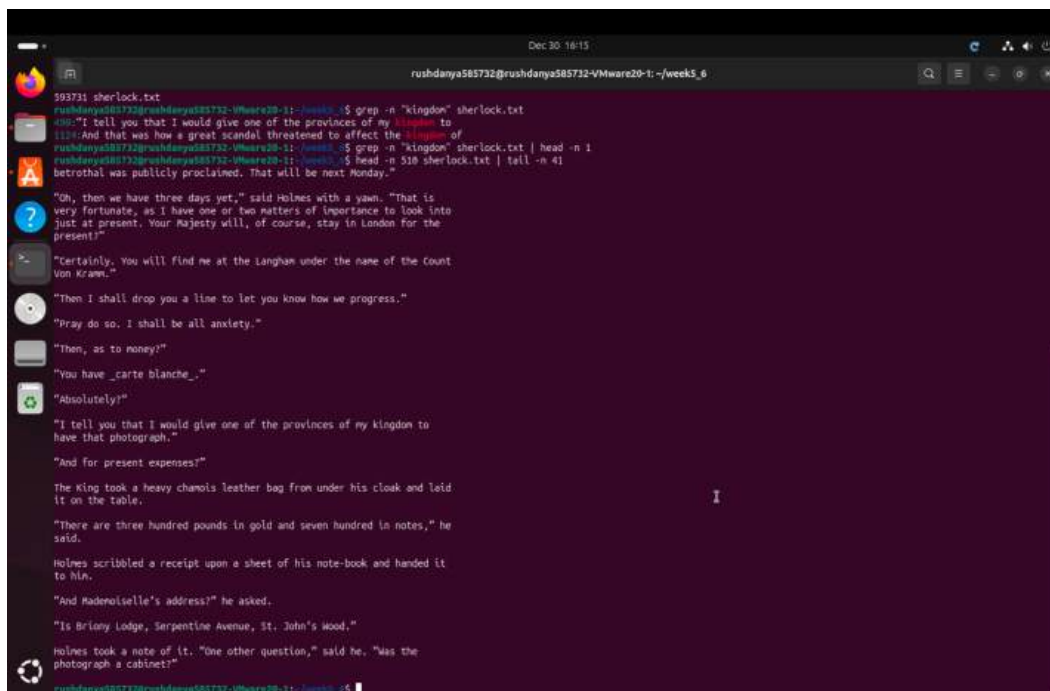
2025-12-30 15:53:59 (1.32 MB/s) - 'sherlock.txt' saved [607504/607504]

rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ ls
sherlock.txt
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ cat sherlock.txt | head
The Project Gutenberg eBook of The Adventures of Sherlock Holmes,
by Arthur Conan Doyle

This eBook is for the use of anyone anywhere in the United States and
most other parts of the world at no cost and with almost no restrictions
whatsoever. You may copy it, give it away or re-use it under the terms
of the Project Gutenberg License included with this eBook or online at
www.gutenberg.org. If you are not located in the United States, you
will have to check the laws of the country where you are located before
using this eBook.
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ less sherlock.txt
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ less sherlock.txt
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ wc sherlock.txt
12306 107562 607504 sherlock.txt
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ wc -m sherlock.txt
593731 sherlock.txt
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$ grep -n "kingdom" sherlock.txt
490:"I tell you that I would give one of the provinces of my kingdom to
1124:And that was how a great scandal threatened to affect the kingdom of
rushdanya585732@rushdanya585732-VWware20-1: ~/week5_6$
```

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”.



```
Dec 30 16:15
rushdanya585732@rushdanya585732-VMware20-1: ~/week5_6
593731 sherlock.txt
rushdanya585732@rushdanya585732-VMware20-1:~/week5_6$ grep -n "kingdom" sherlock.txt
1195:"I tell you that I would give one of the provinces of my kingdom to
1196:And that was how a great scandal threatened to affect the kingdom of
rushdanya585732@rushdanya585732-VMware20-1:~/week5_6$ grep -n "kingdom" sherlock.txt | head -n 1
1195:"I tell you that I would give one of the provinces of my kingdom to
rushdanya585732@rushdanya585732-VMware20-1:~/week5_6$ head -n 510 sherlock.txt | tail -n 41
betrothal was publicly proclaimed. That will be next Monday."

"Oh, then we have three days yet," said Holmes with a yawn. "That is
very fortunate, as I have one or two matters of importance to look into
just at present. Your Majesty will, of course, stay in London for the
present!"

"Certainly. You will find me at the Langham under the name of the Count
von Kramm."

"Then I shall drop you a line to let you know how we progress."

"Pray do so. I shall be all anxiety."

"Then, as to money?"

"You have _carte blanche_."

"Absolutely?"

"I tell you that I would give one of the provinces of my kingdom to
have that photograph."

"And for present expenses?"

The King took a heavy chamol leather bag from under his cloak and laid
it on the table.

"There are three hundred pounds in gold and seven hundred in notes," he
said.

Holmes scribbled a receipt upon a sheet of his note-book and handed it
to him.

"And Mademoiselle's address?" he asked.

"Is Briery Lodge, Serpentine Avenue, St. John's Wood."

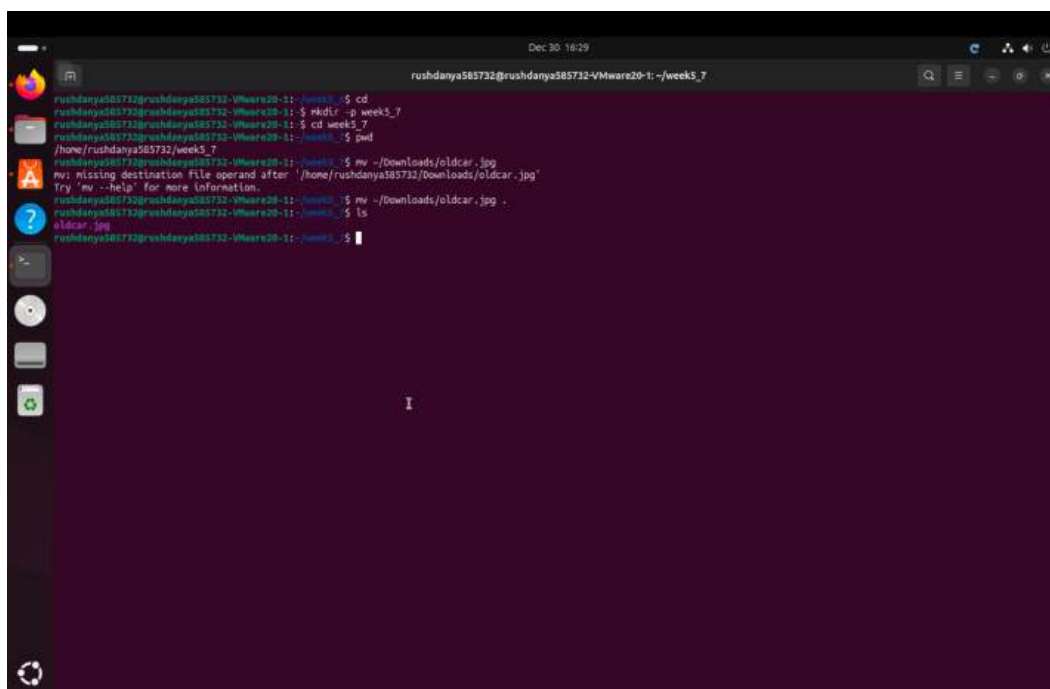
Holmes took a note of it. "One other question," said he. "Was the
photograph a cabinet?"

rushdanya585732@rushdanya585732-VMware20-1:~/week5_6$
```

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

Assignment 5.7: Digital forensics

Relevant screenshots + motivation



```
Dec 30 16:29
rushdanya585732@rushdanya585732-VMware20-1: ~/week5_7
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ cd
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ mkdir -p week5_7
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ cd week5_7
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ pwd
/home/rushdanya585732/week5_7
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ mv ~/Downloads/oldcar.jpg
mv: missing destination file operand after '/home/rushdanya585732/Downloads/oldcar.jpg'
Try 'mv --help' for more information.
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ mv ~/Downloads/oldcar.jpg .
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ ls
oldcar.jpg
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$
```

Figure 5.7-1 – Downloading oldcar.jpg

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

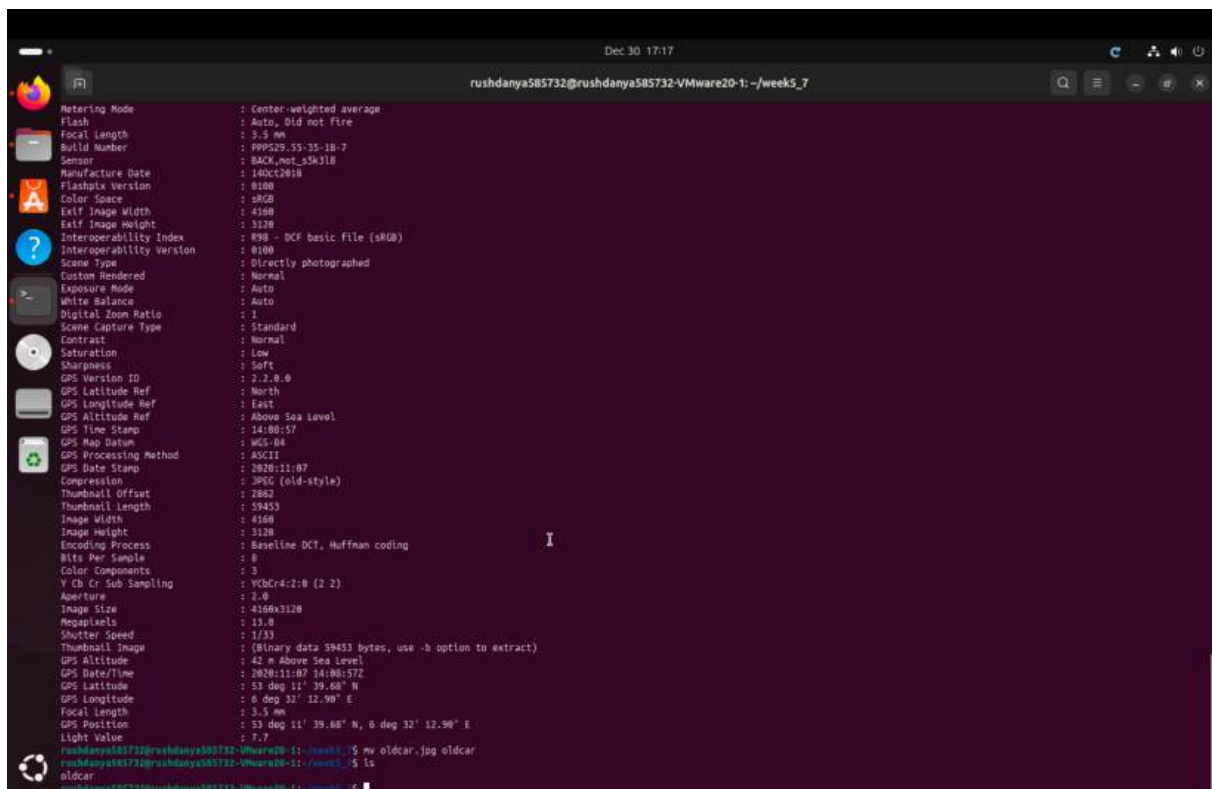
```
Dec 30 16:43
rushdanya585732@rushdanya585732-Vmware20-1: ~/week5_7
exiftool oldcar.jpg
ExifTool Version Number      : 12.76
File Name                    : oldcar.jpg
Directory                     : .
File Size                     : 2.4 Mb
File Modification Date/Time   : 2021:12:30 16:36:45+01:00
File Access Date/Time        : 2021:12:30 16:26:45+01:00
File Inode Change Date/Time   : 2021:12:30 16:25:12+01:00
File Permissions              : -rw-rw-r--
File Type                     : JPEG
File Type Extension           : jpg
MIME Type                     : image/jpeg
EXIF Version                  : 2.2
Exif Byte Order                : Big-endian (Motorola, MM)
Make                           : Motorola
Camera Model Name              : Moto g(6) play
X Resolution                   : 72
Y Resolution                   : 72
Resolution Unit                : Inches
Software                       : aljeter-user 3 PPP529-33-35-18-7 6a60 release-keys
Modify Date                    : 2020:11:07 15:00:57
YCbCr Positioning             : Centered
Exposure Time                  : 1/33
F Number                       : 2.0
Exposure Program               : Program AE
ISO                             : 64
Exif Version                   : 0220
Date/Time Original             : 2020:11:07 15:00:57
Create Date                    : 2020:11:07 15:00:57
Components Configuration      : Y, Cb, Cr, -
Shutter Speed Value            : 1/33
Aperture Value                 : 2.0
Brightness Value               : -1
Exposure Compensation         : 0
Max Aperture Value             : 2.0
Metering Mode                  : Center-weighted Average
Flash                           : Auto, did not fire
Focal Length                   : 3.3 mm
Build Number                   : PPP529-33-35-18-7
Sensor                         : 0A6C6A01_0A310A
Manufacture Date               : 16Oct2018
Flashpix Version               : 0100
Color Space                    : sRGB
Exif Image Width               : 4100
Exif Image Height              : 3128
Interoperability Index         : R98 - DCF basic file (sRGB)
Interoperability Version       : 0100
Scene Type                     : Directly photographed
Custom Rendered                : Normal
Exposure Mode                  : Auto
White Balance                  : Auto
Digital Zoom Ratio              : 1
Scene Capture Type              : Standard
Contrast                       : Normal
```

```
Dec 30 16:44
rushdanya585732@rushdanya585732-Vmware20-1: ~/week5_7
exiftool oldcar.jpg
Brightness Value               : -1
Exposure Compensation         : 0
Max Aperture Value             : 2.0
Metering Mode                  : Center-weighted Average
Flash                           : Auto, did not fire
Focal Length                   : 3.3 mm
Build Number                   : PPP529-33-35-18-7
Sensor                         : 0A6C6A01_0A310A
Manufacture Date               : 16Oct2018
Flashpix Version               : 0100
Color Space                    : sRGB
Exif Image Width               : 4100
Exif Image Height              : 3128
Interoperability Index         : R98 - DCF basic file (sRGB)
Interoperability Version       : 0100
Scene Type                     : Directly photographed
Custom Rendered                : Normal
Exposure Mode                  : Auto
White Balance                  : Auto
Digital Zoom Ratio              : 1
Scene Capture Type              : Standard
Contrast                       : Normal
Saturation                     : Low
Sharpness                      : Soft
GPS Version ID                 : 2.2.0.0
GPS Latitude Ref                : North
GPS Longitude Ref               : East
GPS Altitude Ref               : Above Sea Level
GPS Time Stamp                 : 2020:11:07
GPS Map Datum                  : WGS 84
GPS Processing Method           : ASCII
GPS Date Stamp                 : 2020:11:07
Compression                    : JPEG (old-style)
Thumbnail Offset               : 3861
Thumbnail Length               : 59415
Image Width                    : 4100
Image Height                   : 3128
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
YCbCr Sub Sampling             : YCbCr4:2:0 (2 2)
Aperture                       : 2.0
Image Size                     : 4100x3128
Megapixels                     : 13.0
Shutter Speed                  : 1/33
Thumbnail Image                 : (Binary data 59415 bytes, use -b option to extract)
GPS Altitude                   : 42 m Above Sea Level
GPS Date/Time                  : 2020:11:07 14:00:57Z
GPS Latitude                   : 53 deg 11' 39.60" N, 6 deg 32' 12.90" E
GPS Longitude                  : 6 deg 32' 12.90" E
Focal Length                   : 3.3 mm
GPS Position                    : 53 deg 11' 39.60" N, 6 deg 32' 12.90" E
Light Value                     : 7.7
```

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`.



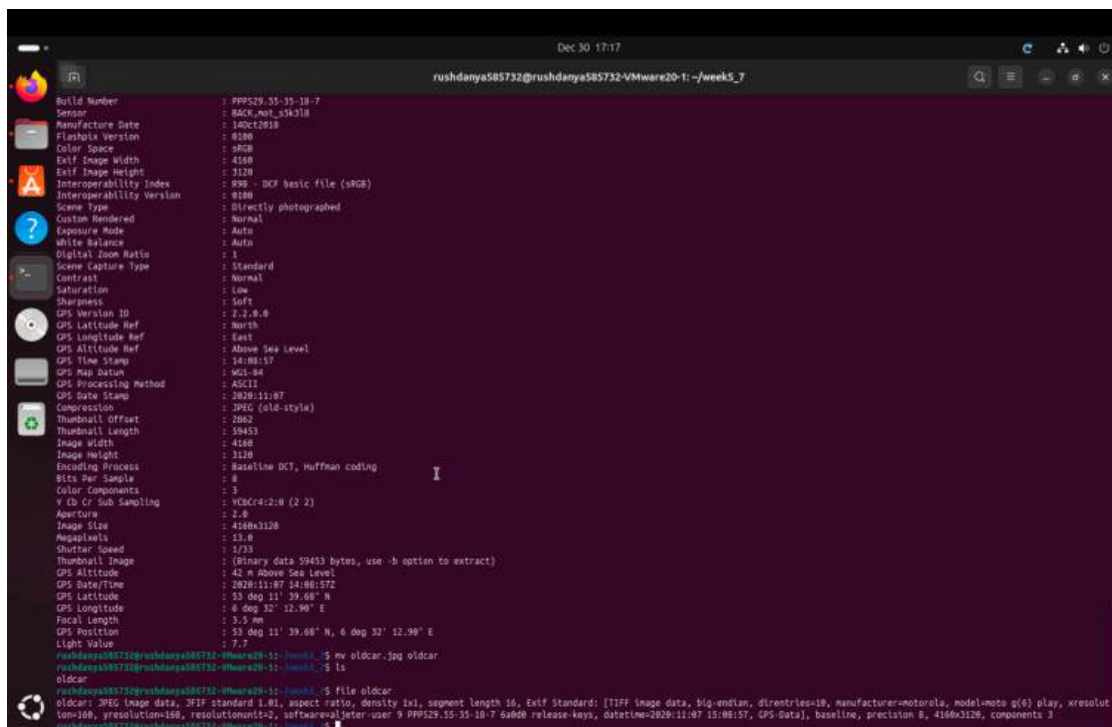
```
Dec 30 17:17
rushdanya585732@rushdanya585732-VMware20-1: ~/week_7

Metering Mode      : Center-weighted average
Flash             : Auto, did not fire
Focal Length       : 3.5 mm
Build Number       : PPP529.35-35-18-7
Sensor            : BACK_mot_s5k318
Manufacture Date   : 14Oct2018
Flashpix Version   : 0100
Color Space        : sRGB
Exif Image Width   : 4160
Exif Image Height  : 3120
Interoperability Index : 898 - DCF basic file (sRGB)
Interoperability Version : 0100
Scene Type         : Directly photographed
Custom Rendered    : Normal
Exposure Mode      : Auto
White Balance      : Auto
Digital Zoom Ratio  : 1
Scene Capture Type  : Standard
Contrast           : Normal
Saturation         : Low
Sharpness         : Soft
GPS Version ID     : 2.2.0-0
GPS Latitude Ref   : North
GPS Longitude Ref  : East
GPS Altitude Ref   : Above Sea Level
GPS Time Stamp     : 14:08:57
GPS Map Datum      : WGS-84
GPS Processing Method : ASCII
GPS Date Stamp     : 2020:11:07
Compression        : JPEG (old-style)
Thumbnail Offset   : 2800
Thumbnail Length   : 59453
Image Width        : 4160
Image Height       : 3120
Encoding Process   : Baseline DCF, Huffman coding
Bits Per Sample    : 8
Color Components   : 3
Y Cb Cr Sub Sampling : YCbCr4:2:0 (2 2)
Aperture           : 2.0
Image Size         : 4160x3120
Megapixels         : 13.0
Shutter Speed      : 1/33
Thumbnail Image     : (Binary data 59453 bytes, use -b option to extract)
GPS Altitude       : 42 m Above Sea Level
GPS Date/Time      : 2020:11:07 14:08:57Z
GPS Latitude       : 53 deg 19' 39.68" N
GPS Longitude      : 6 deg 32' 12.90" E
Focal Length       : 3.5 mm
GPS Position       : 53 deg 19' 39.68" N, 6 deg 32' 12.90" E
Light Value        : 7.7

rushdanya585732@rushdanya585732-VMware20-1: ~/week_7 $ mv oldcar.jpg oldcar
rushdanya585732@rushdanya585732-VMware20-1: ~/week_7 $ ls
oldcar
rushdanya585732@rushdanya585732-VMware20-1: ~/week_7 $
```

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.



```

Dec 30 17:17
rushdanya585732@rushdanya585732-VMware20-1: ~/week5_7

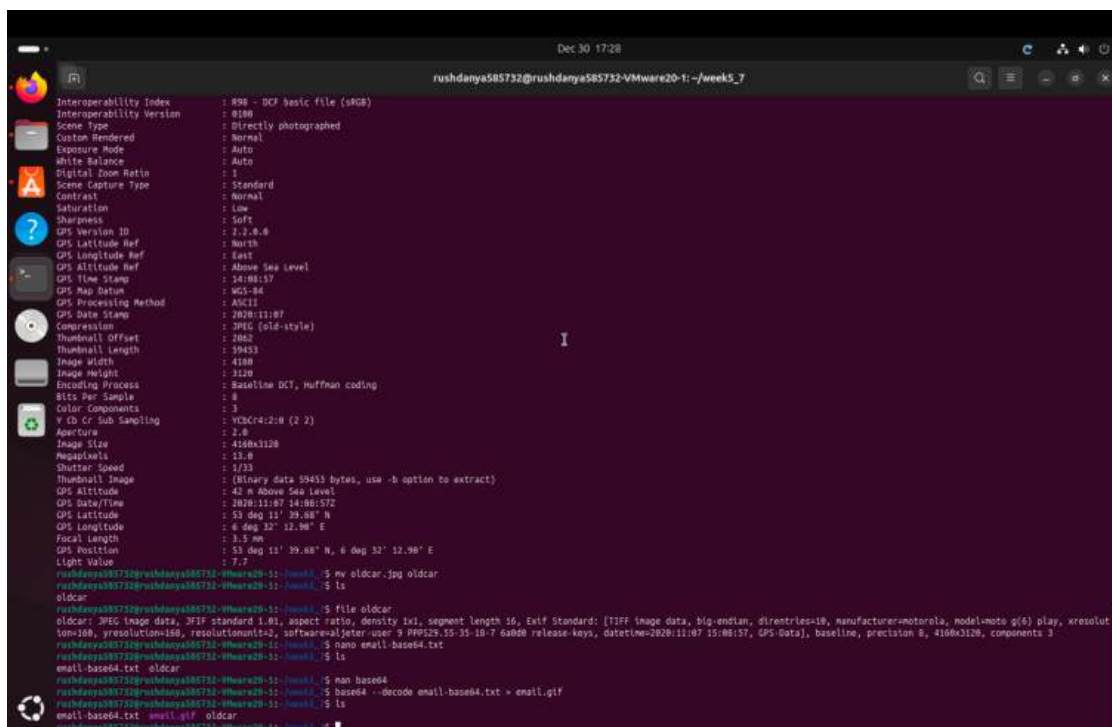
Build Number      : PPS29-35-35-18-7
Sensor            : BACK_not_skl3l8
Manufacture Date  : 14Oct2018
Flashcode version : 0100
Color Space       : sRGB
Exif Image Width  : 4160
Exif Image Height : 3120
Interoperability Index : 898 - DCF basic file (sRGB)
Interoperability Version : 0100
Scene Type        : Directly photographed
Custom Rendered   : Normal
Exposure Mode     : Auto
White Balance     : Auto
Digital Zoom Ratio : 1
Scene Capture Type : Standard
Contrast          : Normal
Saturation        : Low
Sharpness        : Soft
GPS Version ID    : 2.2.0.0
GPS Latitude Ref  : North
GPS Longitude Ref : East
GPS Altitude Ref  : Above Sea Level
GPS Time Stamp    : 2020:11:07 15:08:17
GPS Map Datum     : WGS-84
GPS Processing Method : ASCII
GPS Date Stamp    : 2020:11:07
Compression       : JPEG (old-style)
Thumbnail Offset  : 2062
Thumbnail Length  : 59453
Image Width       : 4160
Image Height      : 3120
Encoding Process  : Baseline DCT, Huffman coding
Bits Per Sample   : 8
Color Components   : 3
Y Cb Cr Sub Sampling : YCbCr4:2:0 (2 2)
Aperture          : 2.8
Image Size        : 4160x3120
Megapixels        : 13.0
Shutter Speed     : 1/33
Thumbnail Image    : (Binary data 59453 bytes, use -b option to extract)
GPS Altitude      : 42 m Above Sea Level
GPS Date/Time     : 2020:11:07 14:06:57Z
GPS Latitude      : 53 deg 11' 39.68" N
GPS Longitude     : 6 deg 32' 12.98" E
Focal Length      : 3.5 mm
GPS Position      : 53 deg 11' 39.68" N, 6 deg 32' 12.98" E
Light Value       : 7.7

rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ mv oldcar.jpg oldcar
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ ls
oldcar
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ file oldcar
oldcar: JPEG image data, JFIF standard 1.01, aspect ratio, density 1x1, segment length 16, Exif Standard: [TIFF image data, big-endian, dirdentries:18, manufacturer:motorola, model:moto g(6) play, xresolut
ion:160, yresolution:160, resolutionunit:2, software:mjpeg-user 9 PPS29-35-35-18-7 6a00 release keys, datetm=2020:11:07 15:08:17, GPS-Gata], baseline, precision 8, 4160x3120, components 3
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$

```

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.



```

Dec 30 17:28
rushdanya585732@rushdanya585732-VMware20-1: ~/week5_7

Interoperability Index : 898 - DCF basic file (sRGB)
Interoperability Version : 0100
Scene Type        : Directly photographed
Custom Rendered   : Normal
Exposure Mode     : Auto
White Balance     : Auto
Digital Zoom Ratio : 1
Scene Capture Type : Standard
Contrast          : Normal
Saturation        : Low
Sharpness        : Soft
GPS Version ID    : 2.2.0.0
GPS Latitude Ref  : North
GPS Longitude Ref : East
GPS Altitude Ref  : Above Sea Level
GPS Time Stamp    : 2020:11:07 15:08:17
GPS Map Datum     : WGS-84
GPS Processing Method : ASCII
GPS Date Stamp    : 2020:11:07
Compression       : JPEG (old-style)
Thumbnail Offset  : 2062
Thumbnail Length  : 59453
Image Width       : 4160
Image Height      : 3120
Encoding Process  : Baseline DCT, Huffman coding
Bits Per Sample   : 8
Color Components   : 3
Y Cb Cr Sub Sampling : YCbCr4:2:0 (2 2)
Aperture          : 2.8
Image Size        : 4160x3120
Megapixels        : 13.0
Shutter Speed     : 1/33
Thumbnail Image    : (Binary data 59453 bytes, use -b option to extract)
GPS Altitude      : 42 m Above Sea Level
GPS Date/Time     : 2020:11:07 14:06:57Z
GPS Latitude      : 53 deg 11' 39.68" N
GPS Longitude     : 6 deg 32' 12.98" E
Focal Length      : 3.5 mm
GPS Position      : 53 deg 11' 39.68" N, 6 deg 32' 12.98" E
Light Value       : 7.7

rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ mv oldcar.jpg oldcar
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ ls
oldcar
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ file oldcar
oldcar: JPEG image data, JFIF standard 1.01, aspect ratio, density 1x1, segment length 16, Exif Standard: [TIFF image data, big-endian, dirdentries:18, manufacturer:motorola, model:moto g(6) play, xresolut
ion:160, yresolution:160, resolutionunit:2, software:mjpeg-user 9 PPS29-35-35-18-7 6a00 release keys, datetm=2020:11:07 15:08:17, GPS-Gata], baseline, precision 8, 4160x3120, components 3
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ nano email-base64.txt
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ ls
email-base64.txt oldcar
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ nano base64
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ base64 -decode email-base64.txt > email.gif
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$ ls
email-base64.txt email.gif oldcar
rushdanya585732@rushdanya585732-VMware20-1:~/week5_7$

```

Figure 5.7-6 – Creating email-base64.txt

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

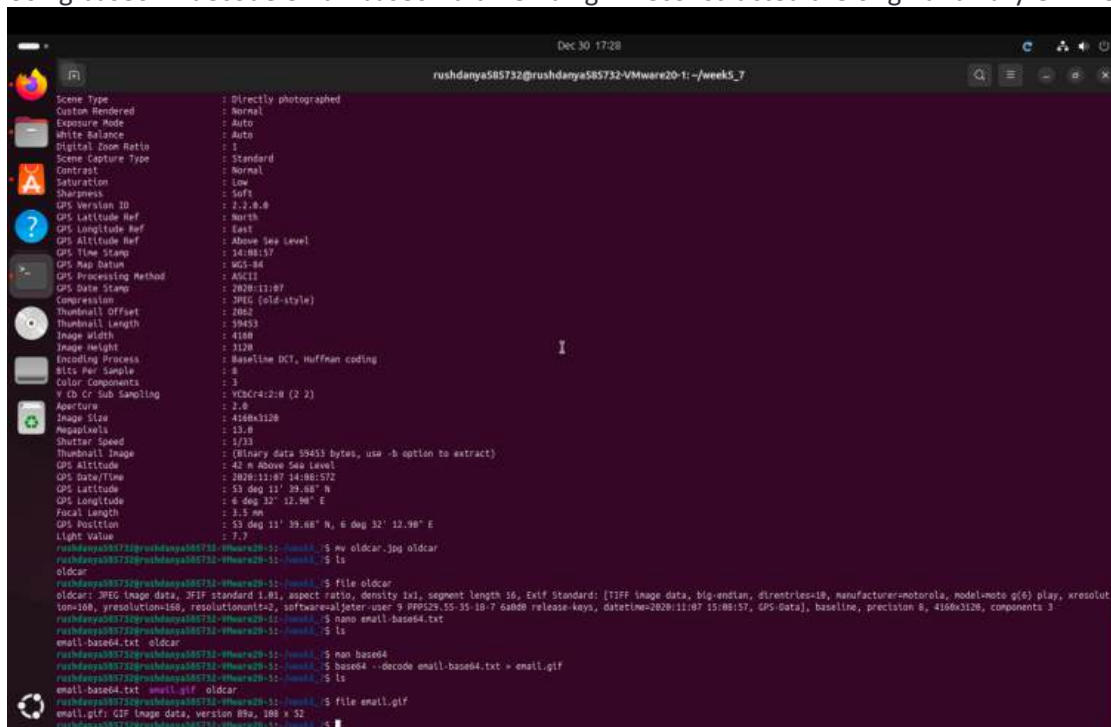


Figure 5.7-7 – Decoding the BASE64 string into email.gif

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

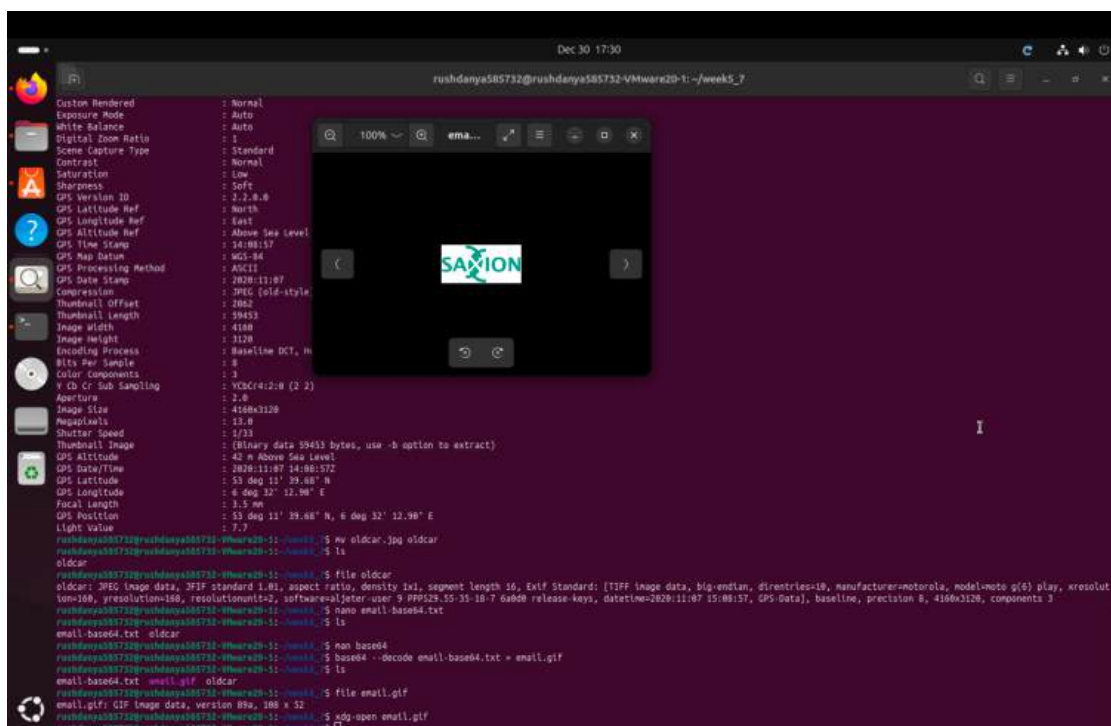


Figure 5.7-8 – Viewing email.gif (Saxion logo)

Assignment 5.8: Steganography

Relevant screenshots + motivation

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

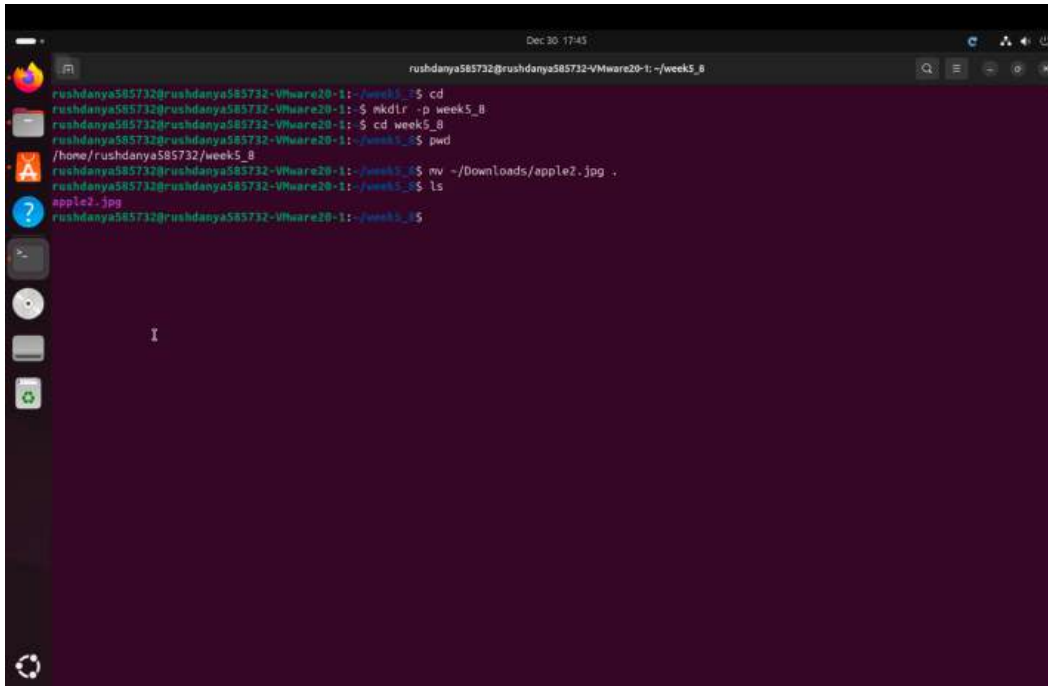


Figure 5.8-1 – apple2.jpg on the Ubuntu desktop

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

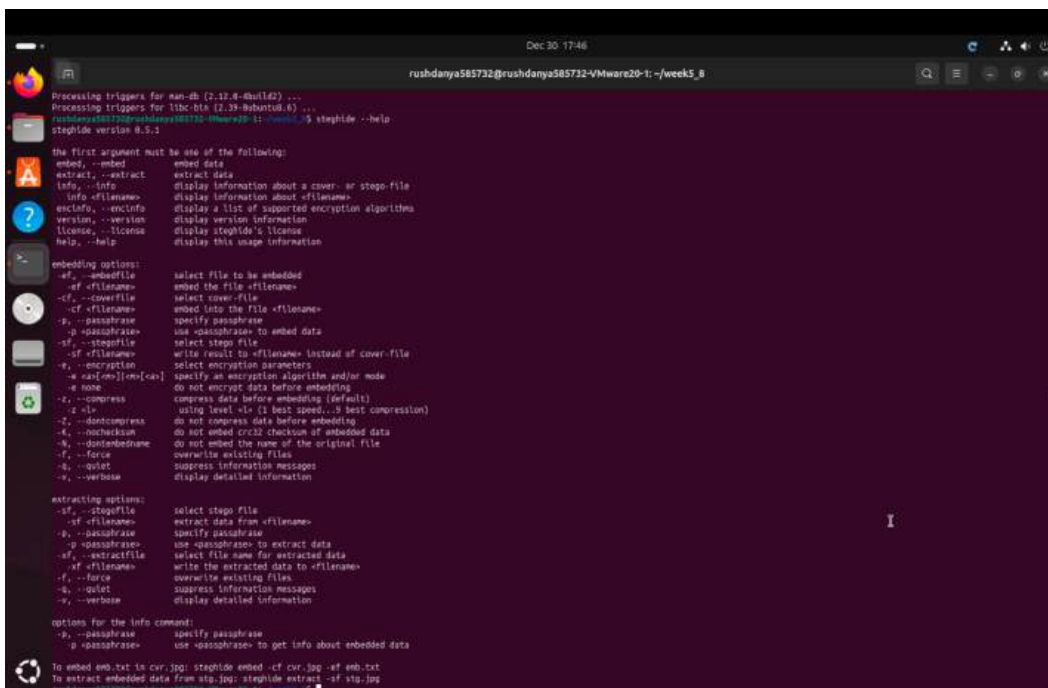


Figure 5.8-2 – Using steghide --help

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

```

Dec 30, 17:48
rushdanya585732@rushdanya585732-VMware20-1: ~/week5_8

steghide version 0.5.1

the first argument must be one of the following:
embed, --embed          embed data
extract, --extract      extract data
info, --info            display information about a cover, or stego-file
info <filename>        display information about <filename>
encinfo, --encinfo      display a list of supported encryption algorithms
version, --version       display version information
license, --license       display steghide's license
help, --help            display this usage information

embedding options:
-f, --embedfile          select file to be embedded
-sf, --stegofile         embed the file <filename>
-cf, --coverfile         select cover file
-Cf, --coverfile <filename> embed into the file <filename>
-p, --passphrase         specify passphrase
-P, --passphrase <passphrase> use <passphrase> to embed data
-sf, --stegofile         select stego file
-Sf, --stegofile <filename> write result to <filename> instead of cover-file
-a, --encryption         select encryption parameters
-A, --encryption <algorithm> specify an encryption algorithm and/or mode
-m <mode>[aes|cam]       do not encrypt data before embedding
-w, --noencrypt          do not encrypt data before embedding
-z, --compress           compress data before embedding (default)
-Z, --nocompress        using level 6 (fastest speed) - 9 best compression
-d, --dontcompress       do not compress data before embedding
-c, --crc32              do not embed crc32 checksum of embedded data
-C, --nochecksum         do not embed the name of the original file
-f, --force              overwrite existing files
-q, --quiet              suppress information messages
-v, --verbose            display detailed information

extracting options:
-sf, --stegofile         select stego file
-Sf, --stegofile <filename> extract data from <filename>
-p, --passphrase         specify passphrase
-P, --passphrase <passphrase> use <passphrase> to extract data
-sf, --extractfile       select file name for extracted data
-Sf, --extractfile <filename> write the extracted data to <filename>
-f, --force              overwrite existing files
-q, --quiet              suppress information messages
-v, --verbose            display detailed information

options for the info command:
-p, --passphrase         specify passphrase
-P, --passphrase <passphrase> use <passphrase> to get info about embedded data

To embed amb.txt in cvr.jpg: steghide embed -cf cvr.jpg -ef amb.txt
To extract embedded data from stp.jpg: steghide extract -sf stp.jpg
rushdanya585732@rushdanya585732-VMware20-1: ~/week5_8
$ steghide extract -sf apple.jpg
Enter passphrase:
write extracted data to "message.txt".

```

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.

```

Dec 30 17:52
rushdanya585732@rushdanya585732-VMware20-1: ~/week5_8

info: «filename»      display information about «filename»
encinfo              display a list of supported encryption algorithms
version              display version information
license              display steghide's license
help, --help         display this usage information

embedding options:
-sf, --embedfile      select file to be embedded
-f, «filename»        embed the file «filename»
-cf, --coverfile      select cover-file
-CF, «filename»       embed into the file «filename»
-p, --passphrase       specify passphrase
-P, «passphrase»       use «passphrase» to embed data
-sf, --stegofile       select stego file
-F, «filename»         write result to «filename» instead of cover-file
-e, --encryption       select encryption parameters
-s «a|c|m|l|mc|ca»    specify an encryption algorithm and/or mode
-s none               do not encrypt data before embedding
-t, --compress         compress data before embedding (default)
-z, «1-9»             using level «z» (1 best speed, 9 best compression)
-T, --dontcompress     do not compress data before embedding
-v, --nochecksum       do not embed CRC32 checksum of embedded data
-d, --dontembedname    do not embed the name of the (original) file
-f, --force            overwrite existing files
-q, --quiet            suppress information messages
-v, --verbose          display detailed information

extracting options:
-sf, --stegofile       select stego file
-f, «filename»         extract data from «filename»
-p, --passphrase       specify passphrase
-P, «passphrase»       use «passphrase» to extract data
-sf, --extractfile     select file name for extracted data
-F, «filename»         write the extracted data to «filename»
-f, --force            overwrite existing files
-q, --quiet            suppress information messages
-v, --verbose          display detailed information

options for the info command:
-p, --passphrase       specify passphrase
-P, «passphrase»       use «passphrase» to get info about embedded data

To embed emb.txt in cvr.jpg: steghide embed -cf cvr.jpg -ef emb.txt
To extract embedded data from stg.jpg: steghide extract -sf stg.jpg
rushdanya585732@rushdanya585732-VMware20-1: ~/week5_8 $ steghide embed -cf applied.jpg
Enter passphrase:
write extracted data to "message.txt".
rushdanya585732@rushdanya585732-VMware20-1: ~/week5_8 $ ls
applied.jpg  message.txt
rushdanya585732@rushdanya585732-VMware20-1: ~/week5_8 $ cat message.txt
hello class.

You have almost completed Week 5.

```

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.

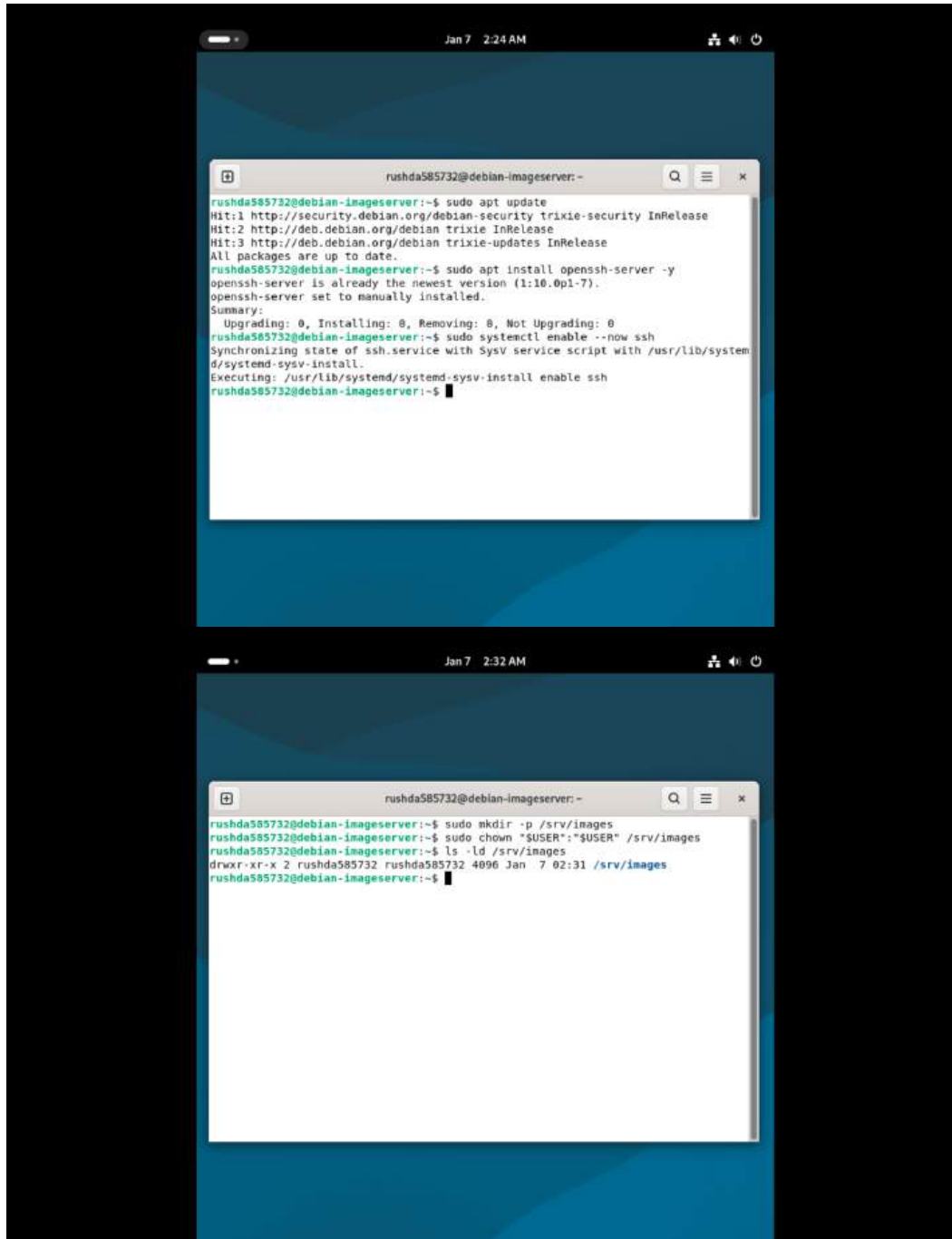


Figure 5.9-1,2 – Debian image server with SSH and /srv/images configured

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

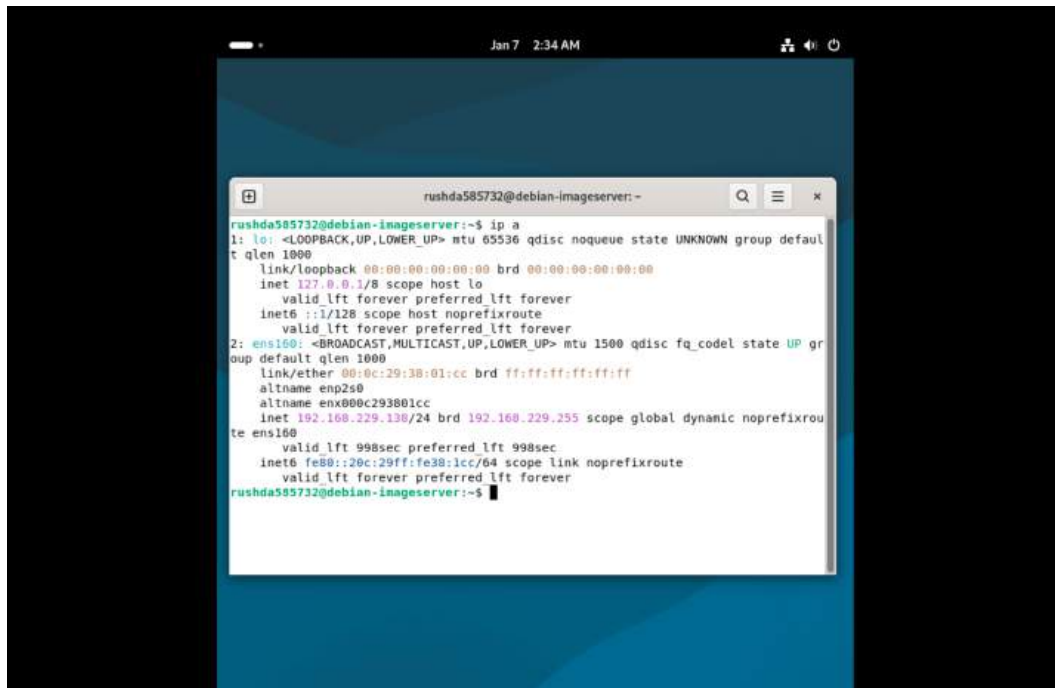


Figure 5.9-3 – IP address of the Debian image server

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

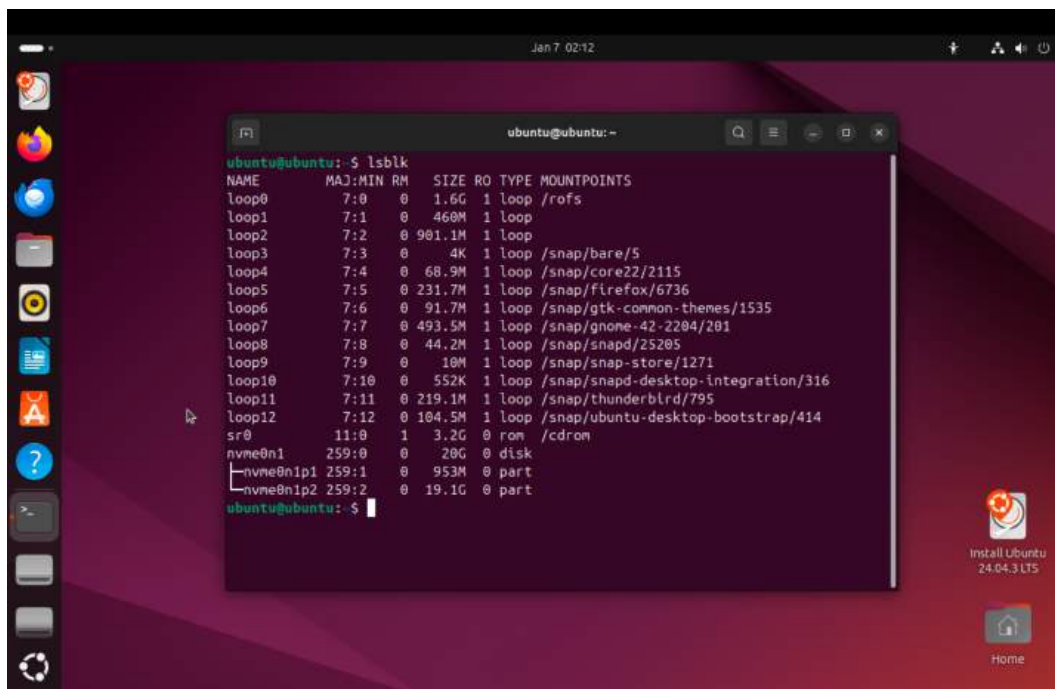


Figure 5.9-4 – Ubuntu client VM booted into Live session with disk lsblk

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.

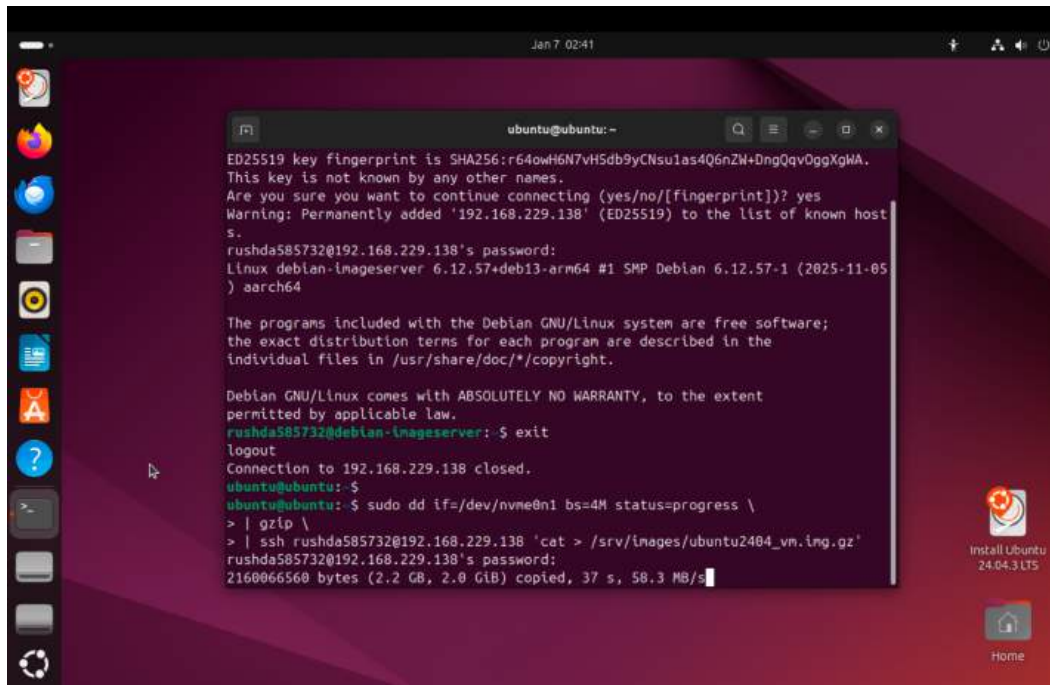


Figure 5.9-5 – Capturing the Ubuntu disk to Debian with `dd | gzip | ssh`

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

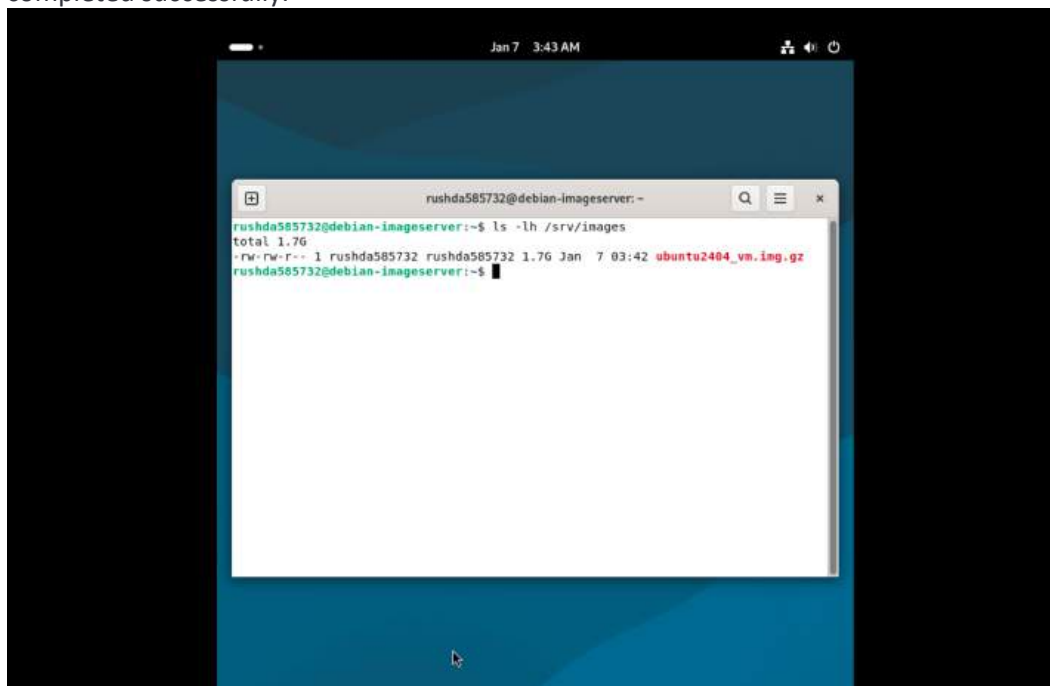


Figure 5.9-6 – Image file stored on Debian

- 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.

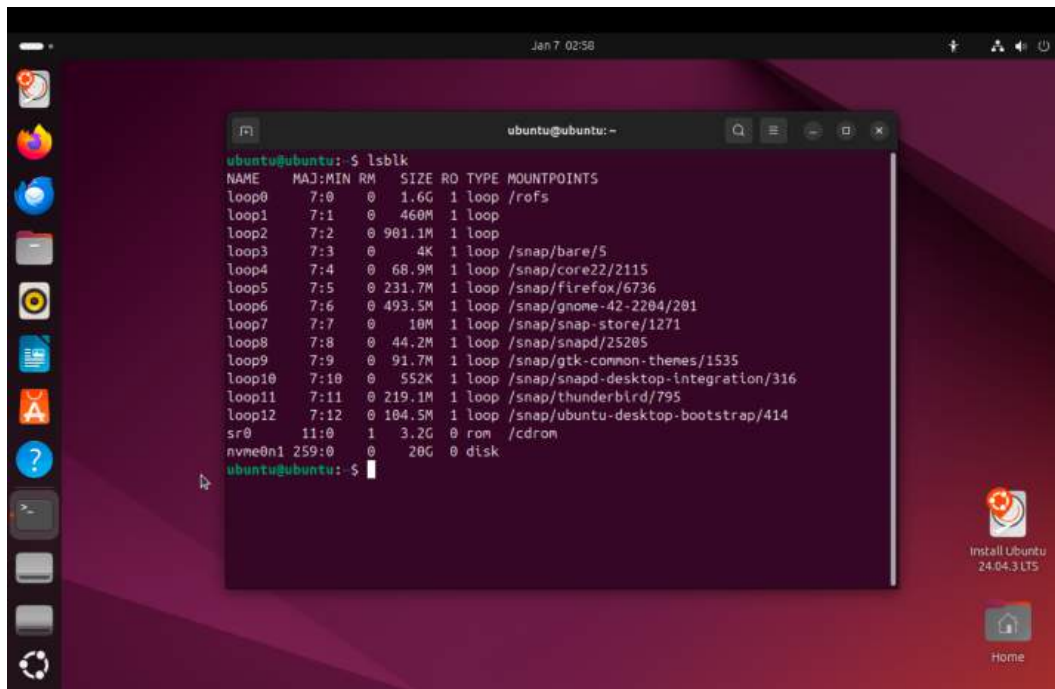
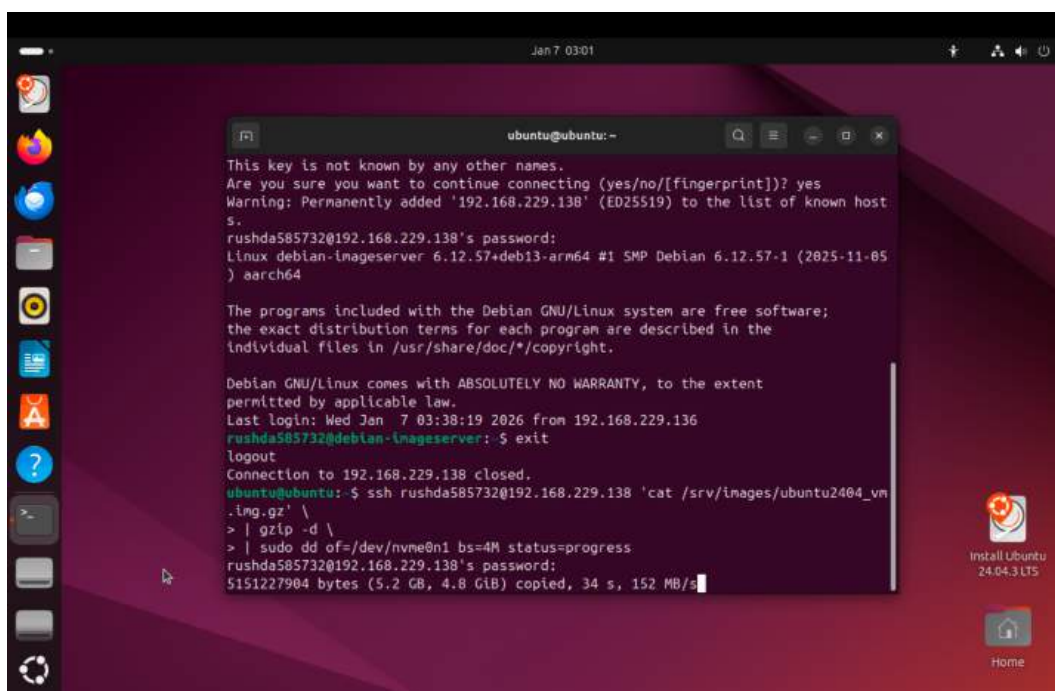


Figure 5.9-7 – Ubuntu-Restored VM Live session showing target disk with lsblk

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.



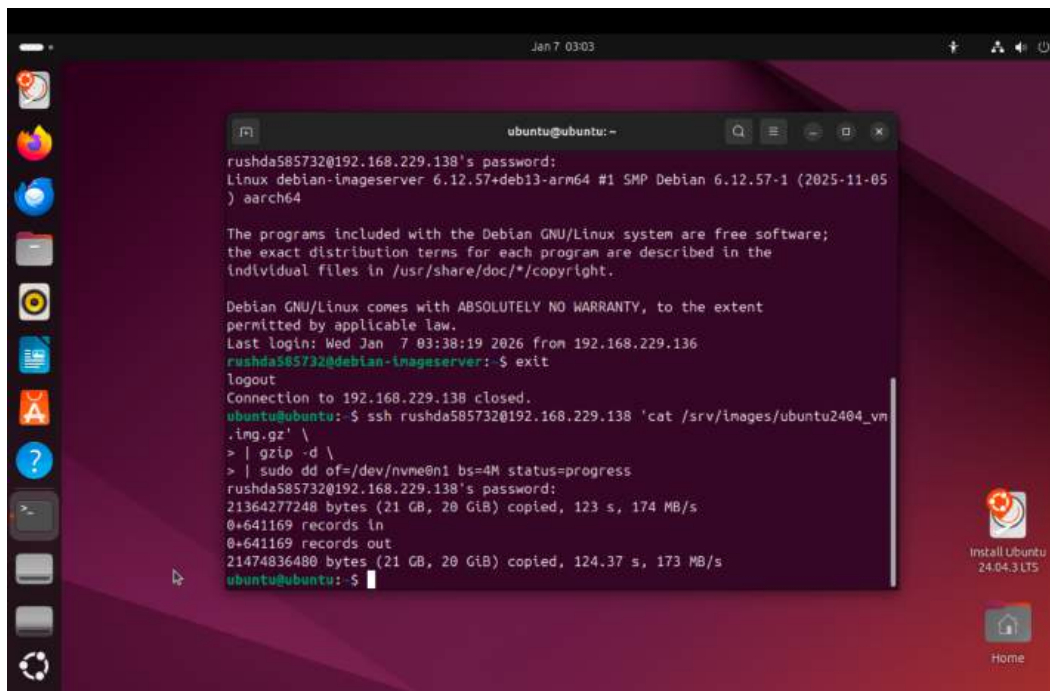


Figure 5.9-8, 9 – Restoring the image from Debian onto the new disk

On a new empty Ubuntu VM I booted into a live session and restored the captured image from the Debian server with:

```
ssh rshda585732@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.

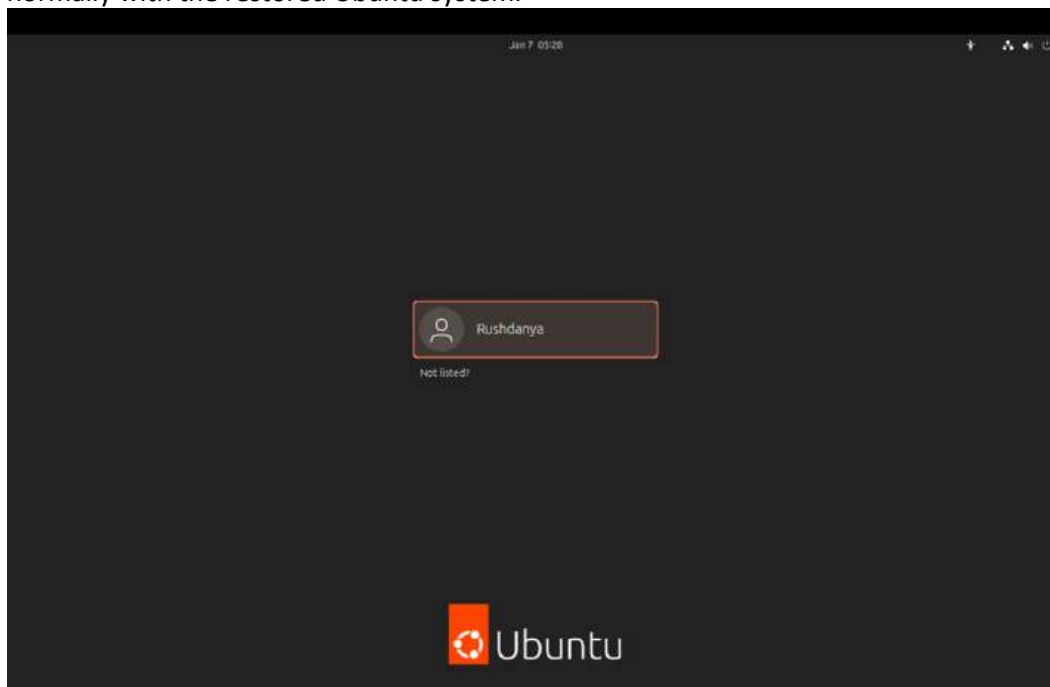


Figure 5.9-10 – Successfully booted restored Ubuntu VM