FSDS
fullstackdatascience

fullstackdatascience.com

Linux

A FREE, STABLE, AND SECURE OPERATING SYSTEM (OS).

FSWE **Lesson 1**

Created by Tien Nguyen
tiennk@fullstackdatascience.com

Learn About

01/ **GNU/LINUX**

02/ **COMMAND LINE BASICS**

03/ **BASH SCRIPTING**

04/ **CRON JOB**

fullstackdatascience.com

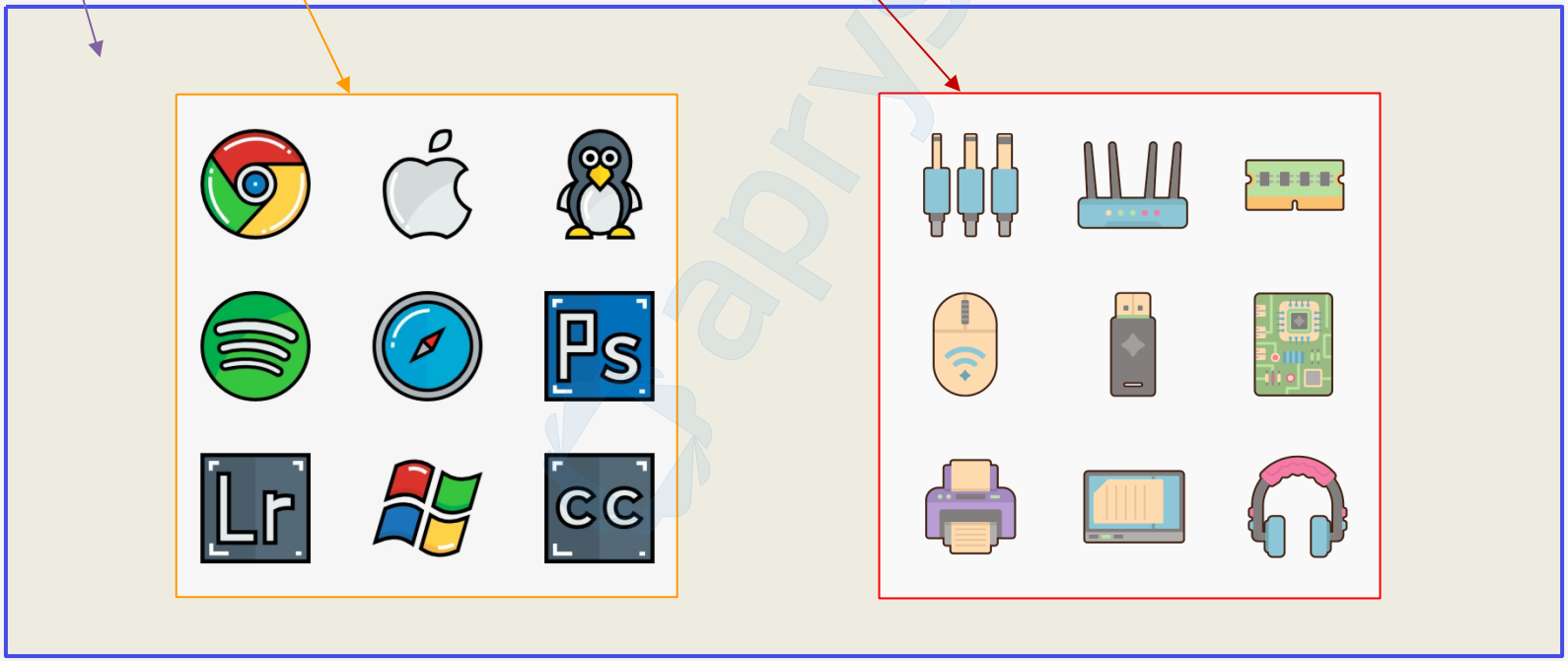
GNU/ Linux

Introduction

- **Open-source** operating system
- Free to use and modify
- Community-developed
- Secure and flexible

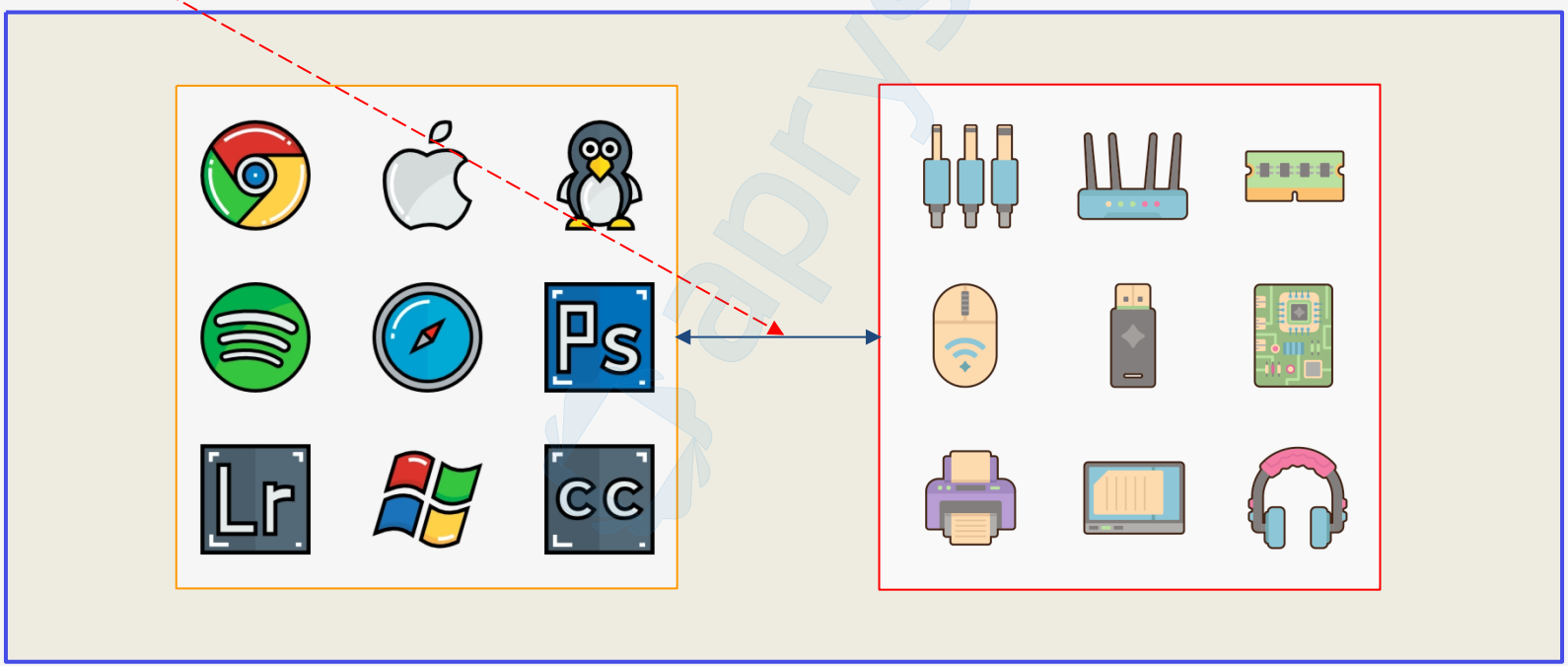
Operating system (OS)

The **OS** is the **software** that manages all the **hardware** and **software on your computer/phone**



Kernel

The **kernel** is the *core part* of the OS



Kernel is like the **"bridge"** between your apps and the hardware

GNU

FSDS
fullstackdatascience

An old OS

GNU: A free software project, provides most tools of a Unix-like OS

GNU project made everything: the wheels, seats, steering, engine parts, etc

But they didn't finish the **engine** (the kernel)

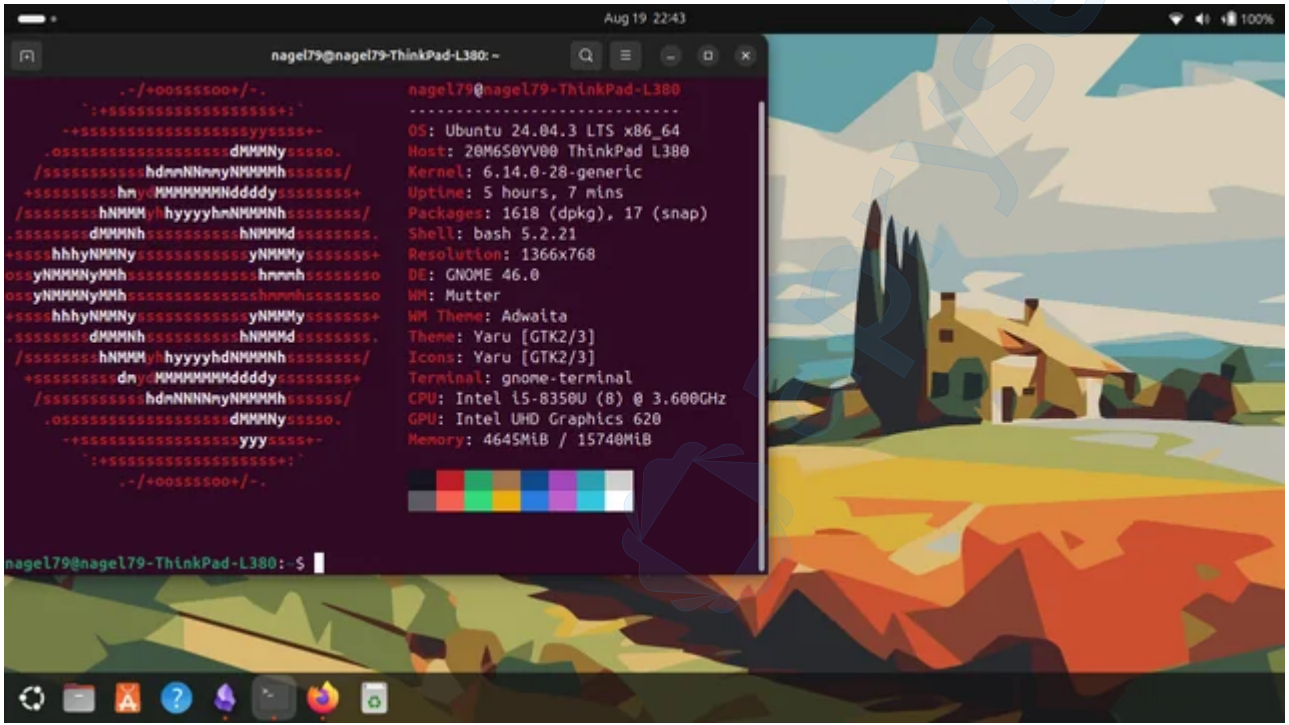
Linus Torvalds came along and built a working **engine (Linux kernel)**

Together, **GNU + Linux kernel = a complete working car (GNU/Linux OS)**

GNU/Linux distros

These are some common Linux distros

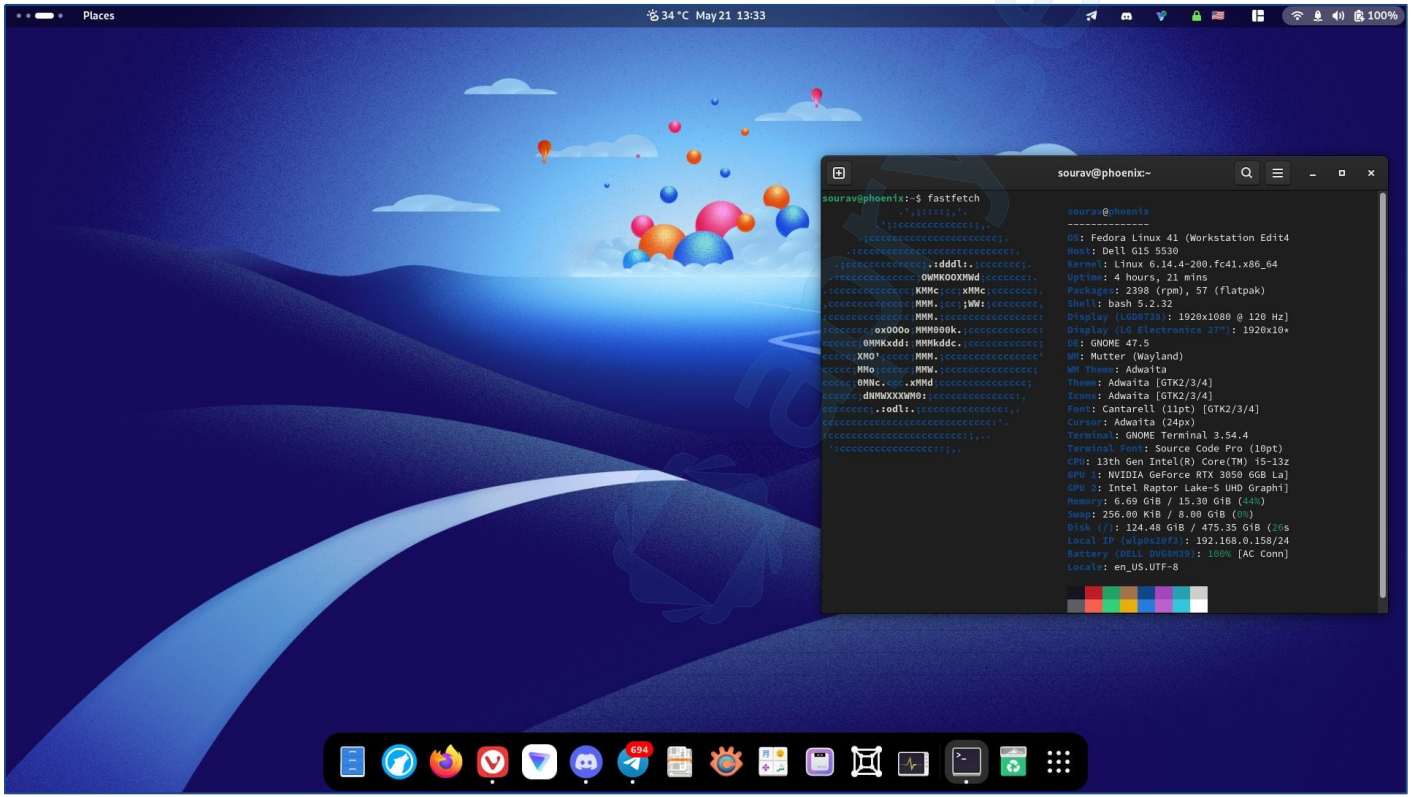
Linux Distro: Ubuntu



- Ubuntu is the easiest for new users
- Large community + lots of tutorials
- Stable, secure, and user-friendly
- Great choice for people new to Linux

We will use this distro in this course

Linux Distro: Fedora



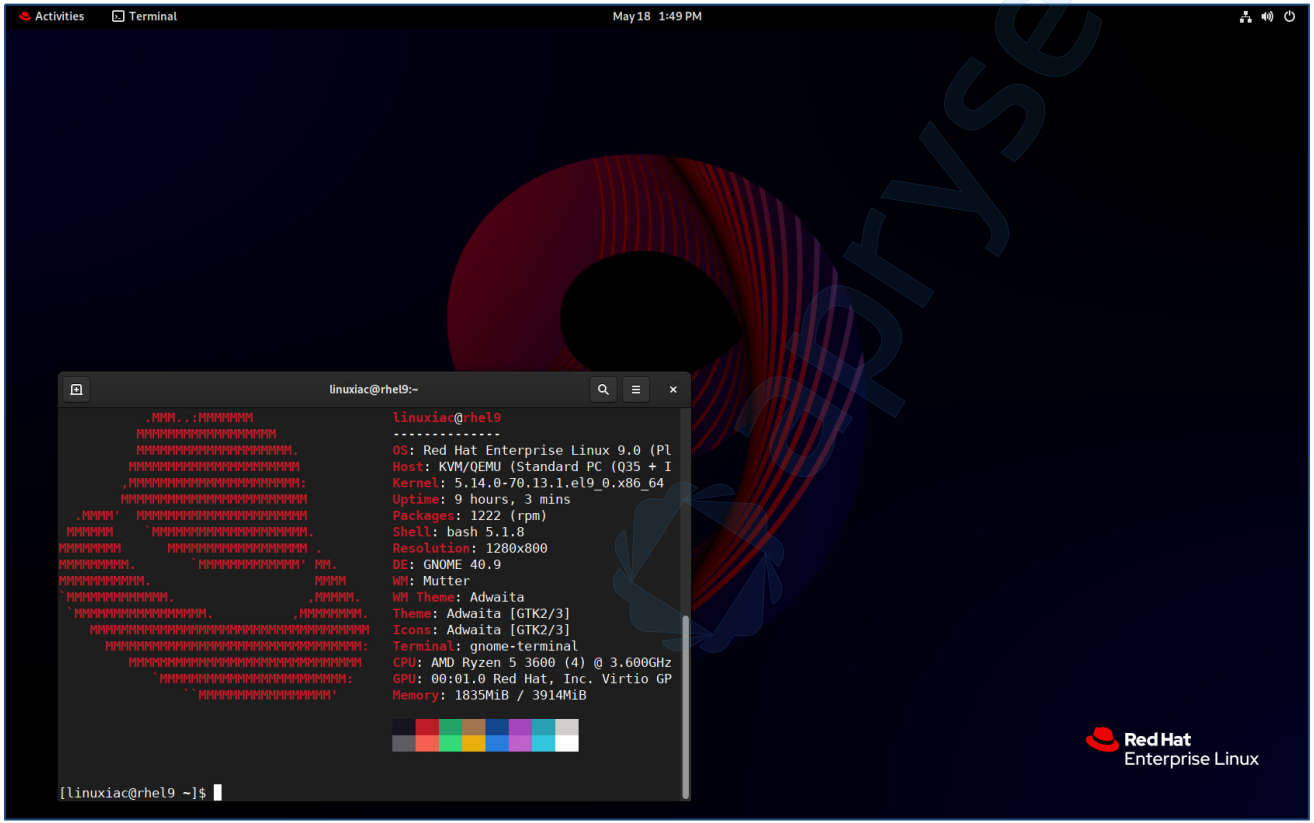
Sponsored by Red Hat

Cutting-edge software, frequent updates

Default GNOME desktop environment

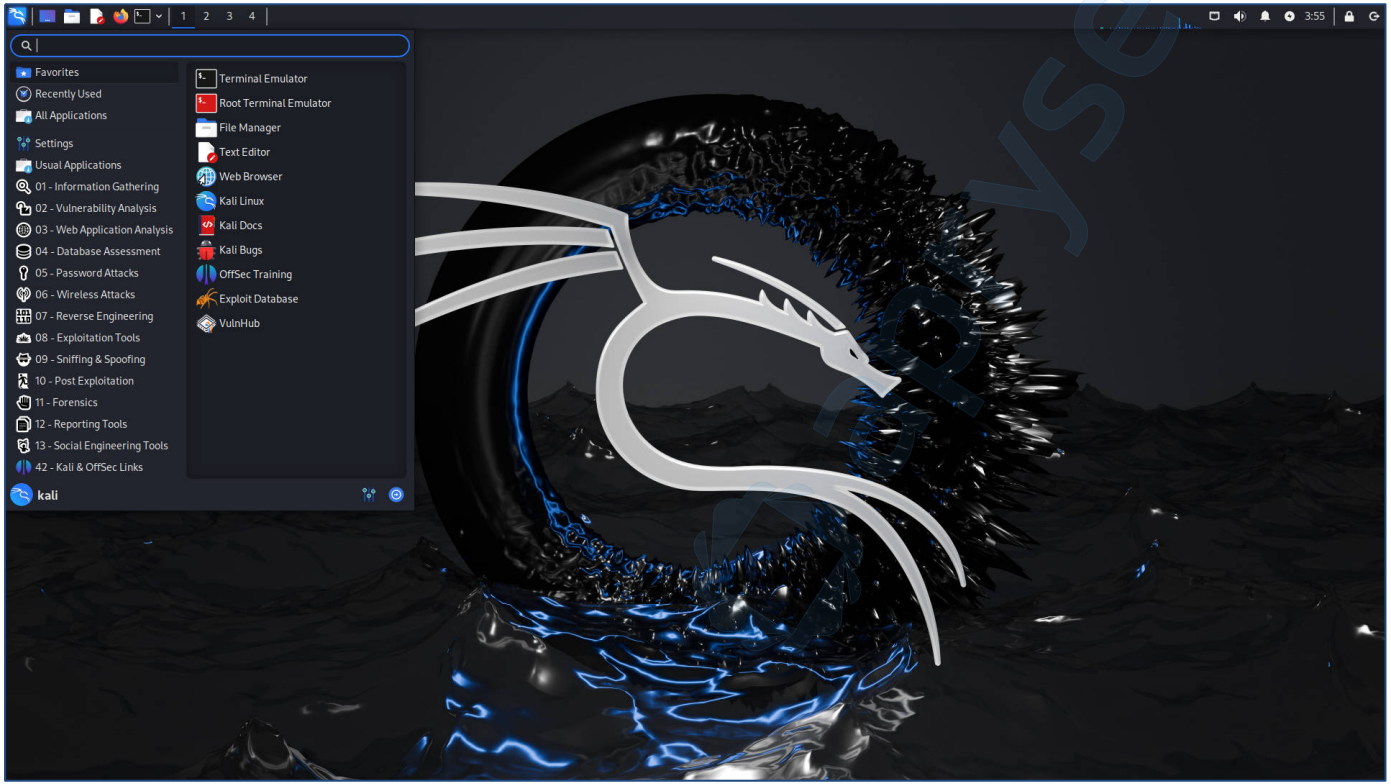
Used as a testbed for Red Hat Enterprise Linux (RHEL)

Linux Distro: Red Hat Linux



- Commercial Linux distribution
- Widely used in enterprises and servers
- Strong support and stability
- Subscription-based model

Linux Distro: Kali Linux



- Based on Debian
- Designed for **penetration testing, ethical hacking, and cybersecurity**
- Comes with hundreds of pre-installed security tools
- Used by security professionals and researchers
- Not recommended for beginners or daily desktop use

Linux Distro: Linux Mint

- Very beginner-friendly (similar to Windows UI)
- Comes with multimedia codecs pre-installed
- Focus on ease of use and stability
- Popular among users switching from Windows

Linux Distro: Arch Linux

FSDS
fullstackdatascience

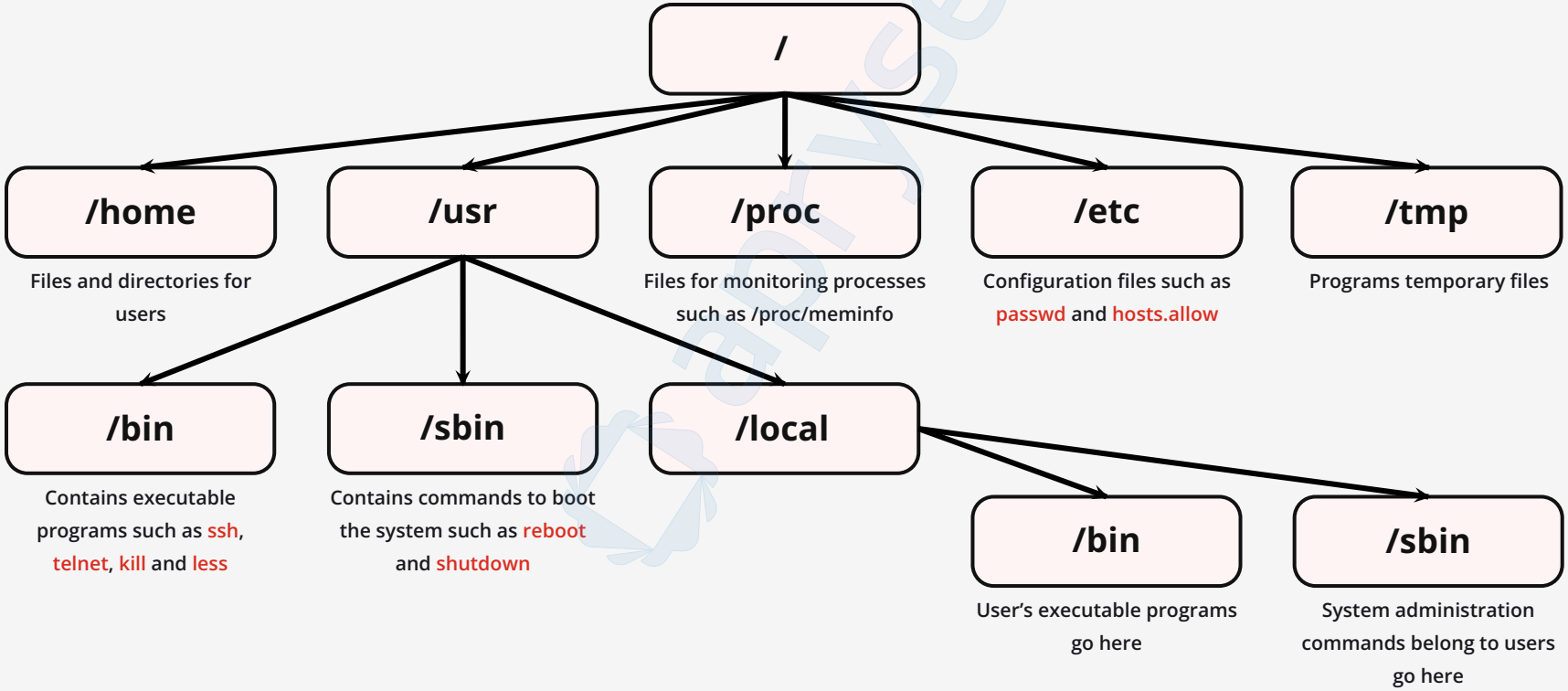
```
Arch Linux 5.8.12-arch1-1 (tty1)

archiso login: root (automatic login)

To install Arch Linux follow the installation guide:
https://wiki.archlinux.org/index.php/Installation_guide

For Wi-Fi, authenticate to the wireless network using the iwctl utility.
Ethernet and Wi-Fi connections using DHCP should work automatically.

After connecting to the internet, the installation guide can be accessed
via the convenience script Installation_guide.

root@archiso ~ # _
```

archlinux™

- Lightweight
- "Do-it-yourself" philosophy → minimal by default
- Requires manual setup and configuration
- Excellent documentation (Arch Wiki)
- Suited for advanced users

You can turn from this dark screen to this crazy UI if you have enough skills!

Linux Directory Structure



TL;DR- GNU/Linux

FSDS
fullstackdatascience

GNU = Free software project, provides most tools of a Unix-like OS.

Linux = A kernel that, together with GNU, forms a full OS.

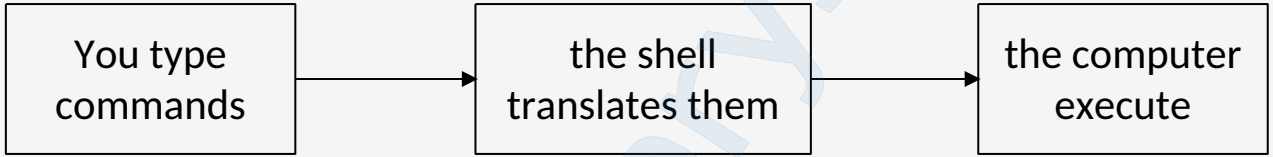
That's why many people say **"GNU/Linux"** instead of just "Linux".

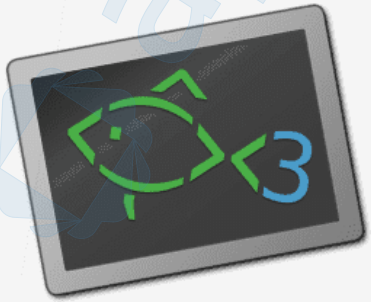
Command Line Basics

Typing commands differs from navigating a graphical user interface (GUI) !

Shell

The **shell is the translator** between you and the computer





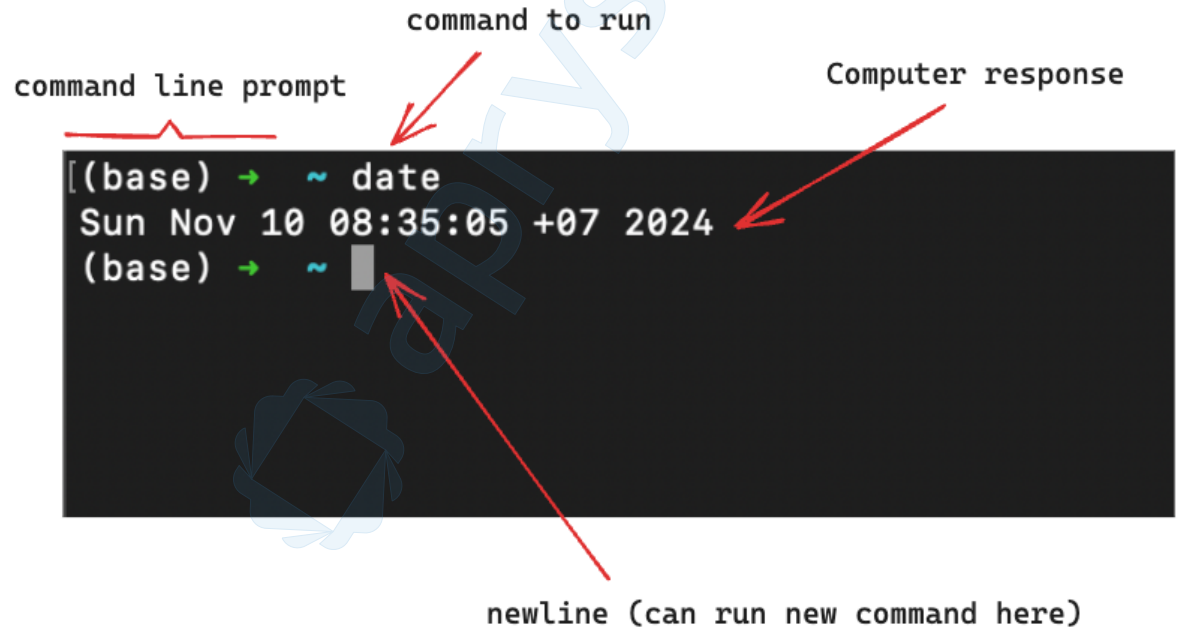
Bash is the default shell for most Linux distributions.

What is Command Line?

FSDS
fullstackdatascience

A text-based interface to interact directly with the computer's operating system

You type commands and get text output instead of clicking buttons, using menus, and dragging things around



Basic Commands

Functions	Popular commands
Navigate the file system	cd, pwd, ls
Manipulate files	cp, touch, mkdir, mv, rm
View files	cat, head, tail
User manual of commands	man
Search	find . -name “*.log” -size +1G
Others	history, clear, shutdown, reboot, uptime

Fun fact:

A directory is just a file containing the names of other files (refer to here)

Navigate the file system

ls (list): Lists files and directories in the current directory (or a specified one)

```
magnusdtd@magnusdtd-VivoBook ~/D/test> ls
folder1/  folder2/
magnusdtd@magnusdtd-VivoBook ~/D/test>
```

Use **ls** to list files inside in the current directory

Use **ls** to list files inside the **folder1** directory

```
magnusdtd@magnusdtd-VivoBook ~/D/test> ls folder1
file.txt  folder1.1/  folder1.2/
magnusdtd@magnusdtd-VivoBook ~/D/test>
```

Navigate the file system

cd (change directory): Moves you from your current working directory to another one

```
magnusdtd@magnusdtd-VivoBook ~/D/test> ls
folder1/  folder2/
magnusdtd@magnusdtd-VivoBook ~/D/test> cd folder1/
magnusdtd@magnusdtd-VivoBook ~/D/t/folder1> ls
folder1.1/  folder1.2/
magnusdtd@magnusdtd-VivoBook ~/D/t/folder1>
```

The **test** has 2 folder: **folder1** and **folder2**.
The **folder1** has 2 subfolder: **folder1.1** and **folder1.2**

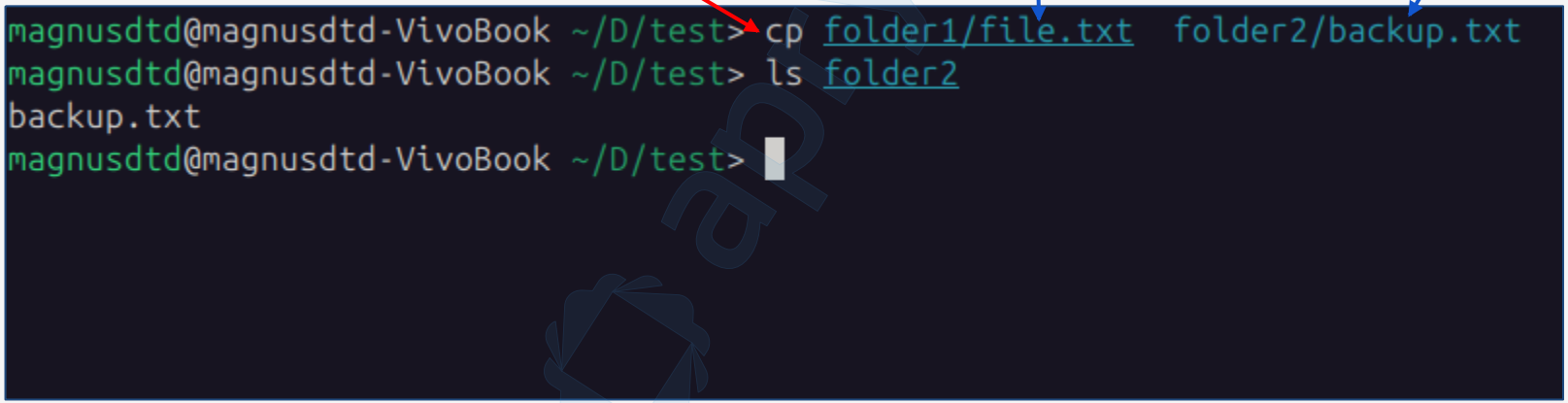
Navigate the file system

pwd (print working directory): Shows the full path of the directory you're currently in

Manipulate files

cp (copy): Makes a copy of a file

New file name

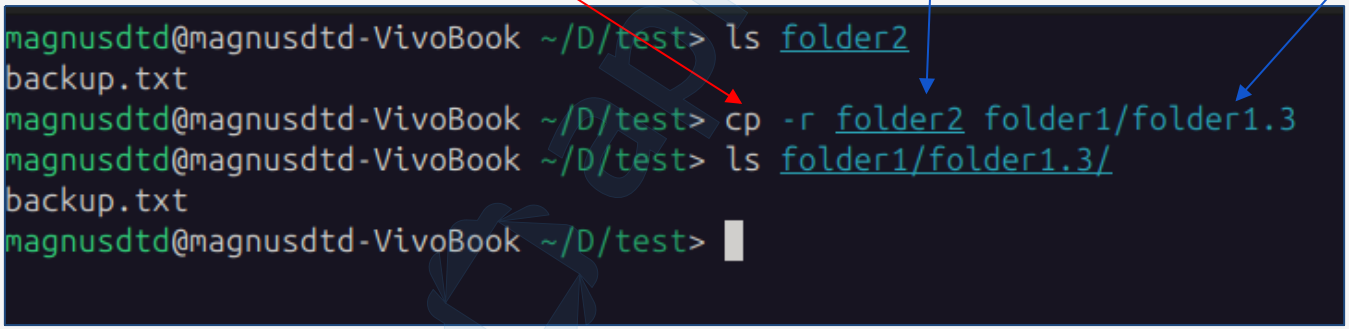


Copy the **file.txt** from the **folder1** to the **folder2**

Manipulate files

cp (copy): Makes a copy of a folder

New folder name



Copy all the content of the **folder2** to the new destination **folder1/folder1.3**

Manipulate files

touch: Creates a new empty file

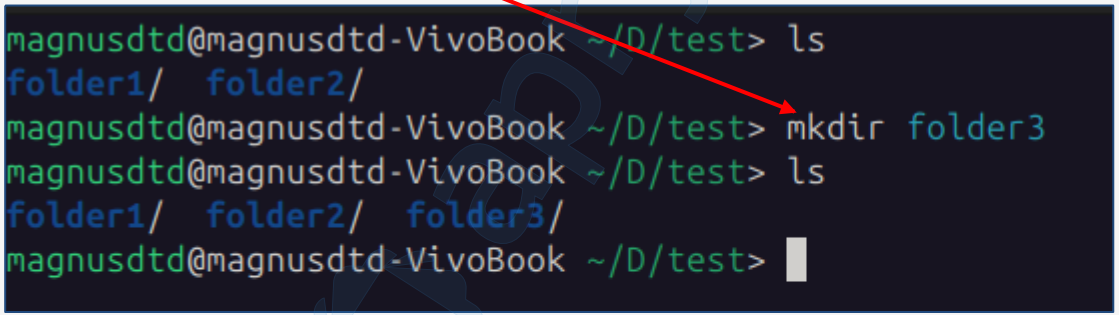
if it doesn't exist

```
magnusdtd@magnusdtd-VivoBook ~/D/test> ls folder1
file.txt  folder1.1/  folder1.2/  folder1.3/
magnusdtd@magnusdtd-VivoBook ~/D/test> touch folder1/newfile.txt
magnusdtd@magnusdtd-VivoBook ~/D/test> ls folder1
file.txt  folder1.1/  folder1.2/  folder1.3/  newfile.txt
magnusdtd@magnusdtd-VivoBook ~/D/test>
```

Create a new file named **newfile.txt**

Manipulate files

mkdir **(make directory)**: Creates a new folder



Create a new folder named **folder3**

Manipulate files

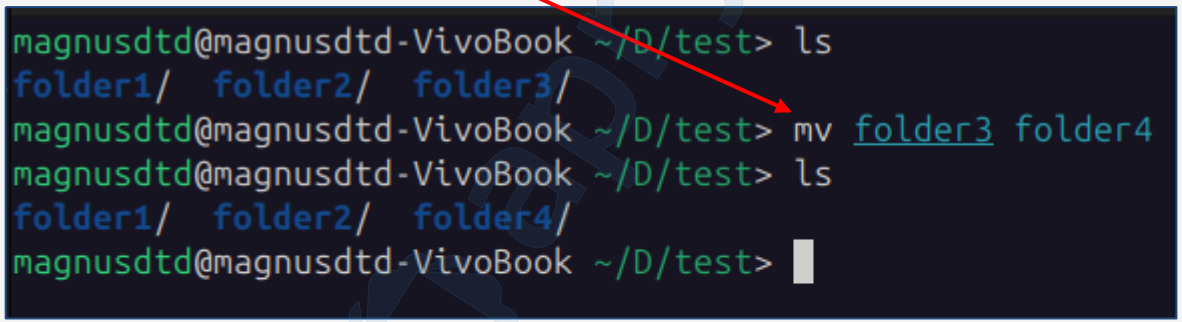
mv **(move)**: Moves files/directories

```
magnusdtd@magnusdtd-VivoBook ~/D/test> ls folder1
file.txt  folder1.1/  folder1.2/  folder1.3/  newfile.txt
magnusdtd@magnusdtd-VivoBook ~/D/test> mv folder1/newfile.txt  folder1/file2.txt
magnusdtd@magnusdtd-VivoBook ~/D/test> ls folder1
file2.txt  file.txt  folder1.1/  folder1.2/  folder1.3/
magnusdtd@magnusdtd-VivoBook ~/D/test>
```

Create a new folder named **folder3**

Manipulate files

mv **(move)**: Renames files/directories



Rename from **folder3** to **folder4**

Manipulate files

rm **(remove)**: Deletes files (or directories with -r)

```
magnusdtd@magnusdtd-VivoBook ~/D/test> ls folder1
file2.txt  file.txt  folder1.1/  folder1.2/  folder1.3/
magnusdtd@magnusdtd-VivoBook ~/D/test> rm folder1/file2.txt
magnusdtd@magnusdtd-VivoBook ~/D/test> ls folder1
file.txt  folder1.1/  folder1.2/  folder1.3/
magnusdtd@magnusdtd-VivoBook ~/D/test>
```

Delete **file2.txt**

Delete **folder4**

```
magnusdtd@magnusdtd-VivoBook ~/D/test> ls
folder1/  folder2/  folder4/
magnusdtd@magnusdtd-VivoBook ~/D/test> rm -r folder4
magnusdtd@magnusdtd-VivoBook ~/D/test> ls
folder1/  folder2/
magnusdtd@magnusdtd-VivoBook ~/D/test>
```

View files

cat **(concatenate)**: reads files and writes their contents to the terminal

```
magnusdtd@magnusdtd-VivoBook ~/D/test> cat folder1/file.txt
This is the first lesson of the FSWE course!
magnusdtd@magnusdtd-VivoBook ~/D/test>
```

Show the content of the **file.txt**

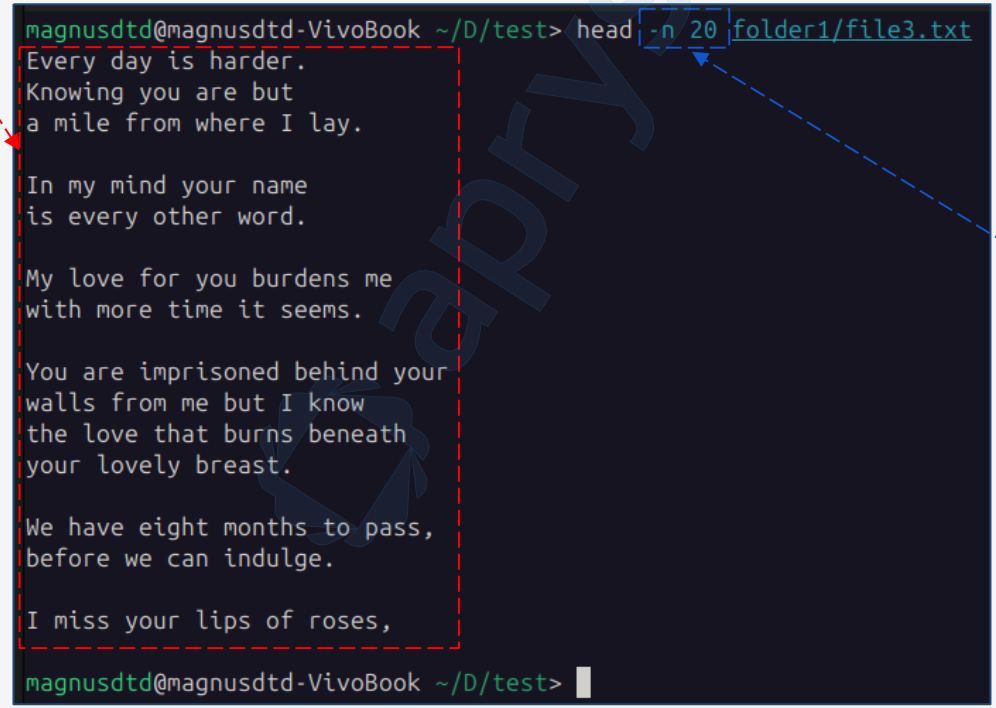
Merge the contents of **file.txt** and **file2.txt** into **merge.txt.**

```
magnusdtd@magnusdtd-VivoBook ~/D/test> cat folder1/file.txt folder1/file2.txt \
                                       > folder1/merge.txt
magnusdtd@magnusdtd-VivoBook ~/D/test> cat folder1/merge.txt
This is the first lesson of the FSWE course!
Enjoy the lesson!
magnusdtd@magnusdtd-VivoBook ~/D/test>
```

View files

head **(head of the file)**: prints the first part of a file

Show the content of the **file.txt**



default: 10 lines

shows the first 20 lines

View files

tail **(tail of the file)**: prints the first part of a file

Show the content of the **file.txt**

```
magnusdtd@magnusdtd-VivoBook ~/D/test> tail -n 20 folder1/file3.txt
I'm an over powered anxiety complex

in a box as big as a dime.


My love for you will never cease,

Death is the only ending for this.


Come to me because

I am yet a mere feather

You are my match my lover

My sweet and beautiful Heather.
magnusdtd@magnusdtd-VivoBook ~/D/test>
```

default: 10 lines

Others

man **(manual)**: Displays the documentation for a command.

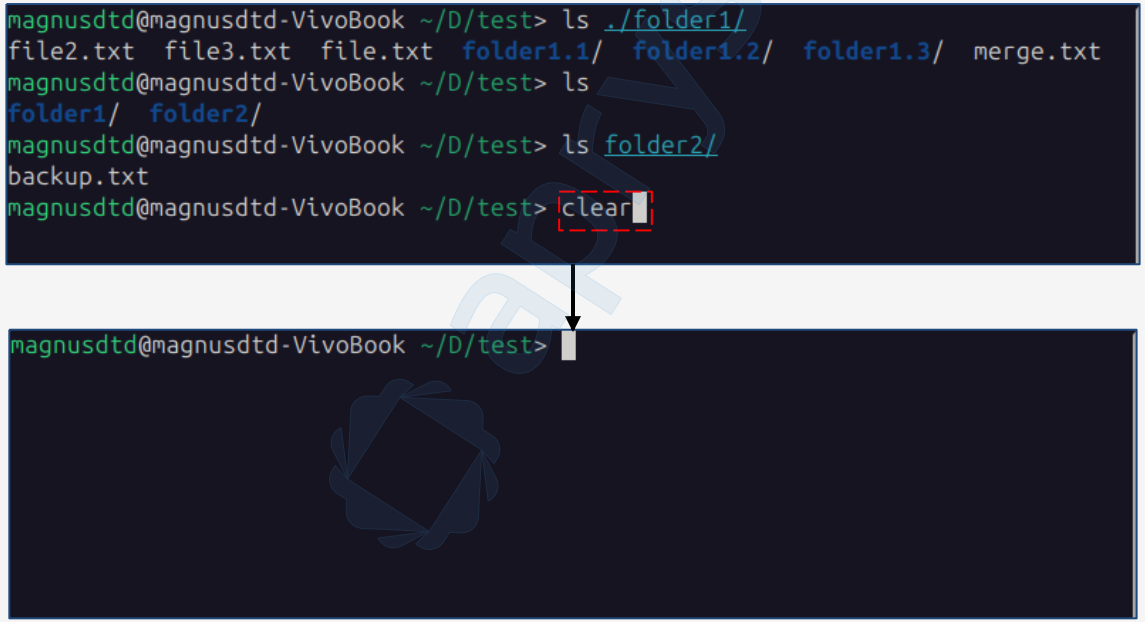
```
magnusdtd@magnusdtd-VivoBook ~/D/test> man ls
magnusdtd@magnusdtd-VivoBook ~/D/test>
```

```
LS(1)                          User Commands                          LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about  the  FILEs (the current directory by default).
       Sort entries alphabetically if none of -cftuvSUX nor  --sort  is  speci-
       fied.

       Mandatory arguments to long options are mandatory for short options too.

       -a, --all
              do not ignore entries starting with .

       -A, --almost-all
              do not list implied . and ..

       --author
              with -l, print the author of each file
Manual page ls(1) line 1 (press h for help or q to quit)
```

Show the documentation for the **ls** command

Others

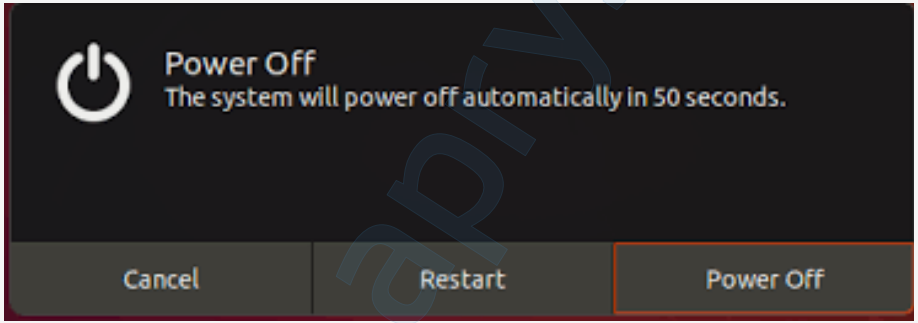
clear: Clears the terminal screen



The screen has been cleaned up after using the **clear** command

Others

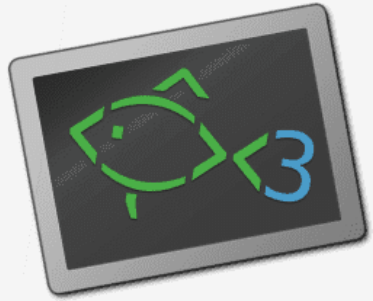
shutdown/reboot: Powers off/restarts your computer



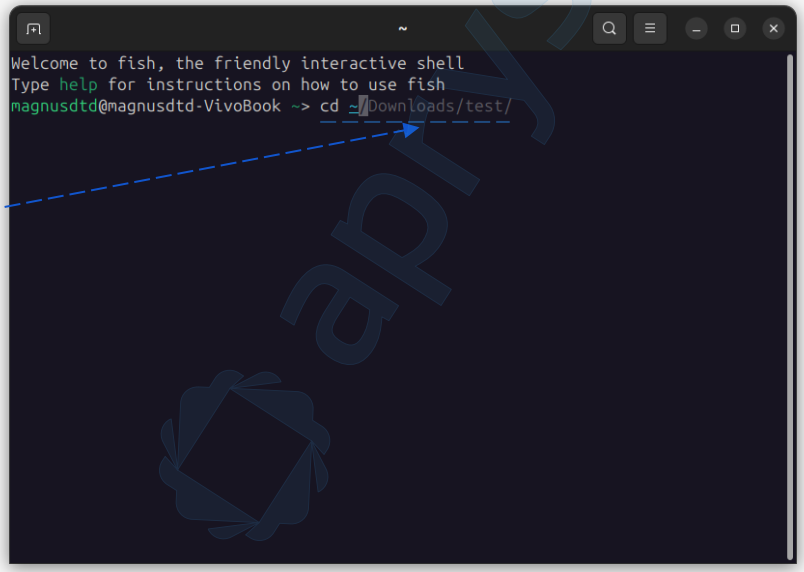
! DO NOT TRY THESE COMMANDS RIGHT NOW !

Fish shell

Wow, that's a lot of stuff. I can't remember all of them ... You can try Fill Shell



Autocomplete feature



Warning: Some commands in Fish may differ the Bash (Linux default shell)

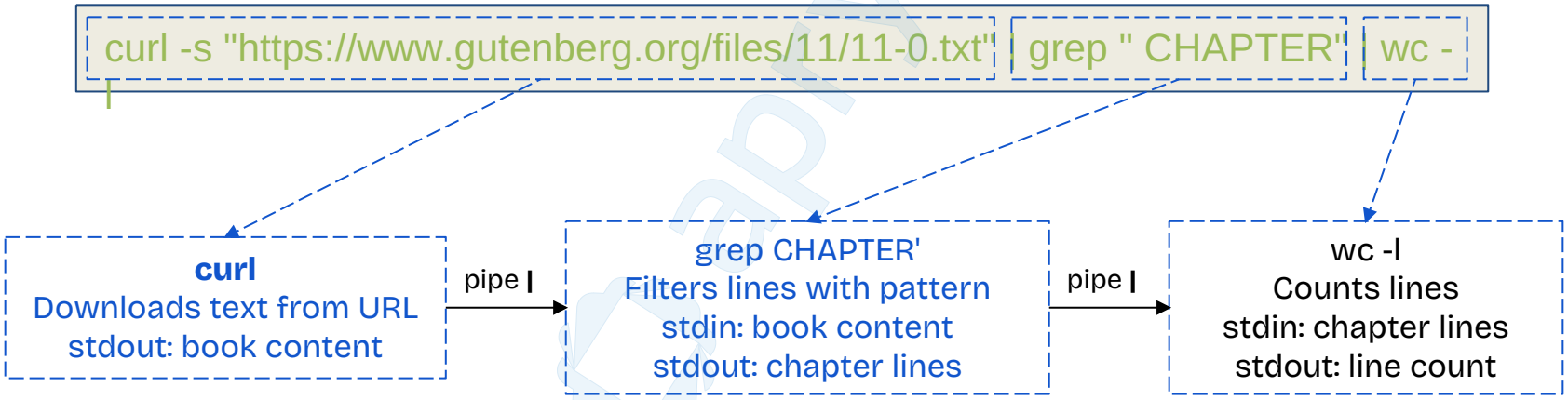
Interesting commands

FSDS
fullstackdatascience

Replace all unix words with linux	sed 's/unix/linux/g' geekfile.txt
List all containers' names containing minio, skipping the row number 0, and stop them	docker ps -a | grep "minio" | awk "NR>1 {print $1}" | xargs docker stop
Find large files or folders	find / -type f -size +1G OR find / -type d -size +1G
Summary disk usage with	du -sh *

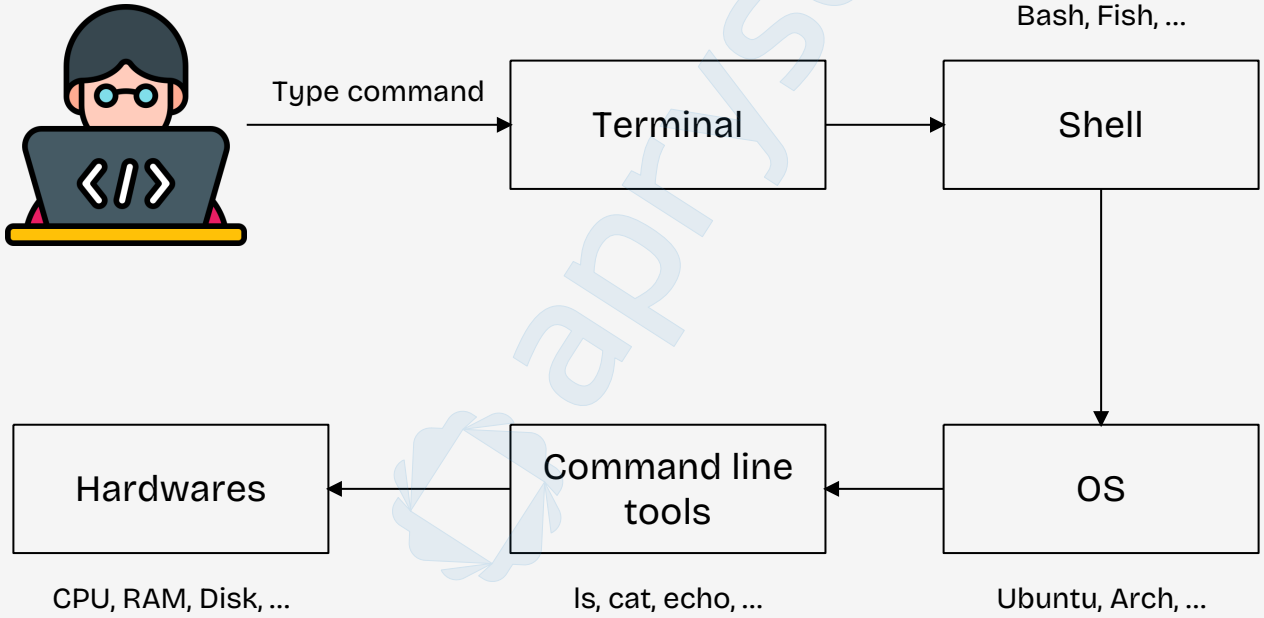
Combining command-line tools

Command-line tools are designed to do one thing and do it well. We can combine these tools to create powerful workflows by chaining them together using pipes and redirects



The command prints the **number of chapters** in *Alice's Adventures in Wonderland*

Command line environment



Command line is useful

FSDS
fullstackdatascience

Fast & Interactive

Instant command execution
— type, hit Enter, go!

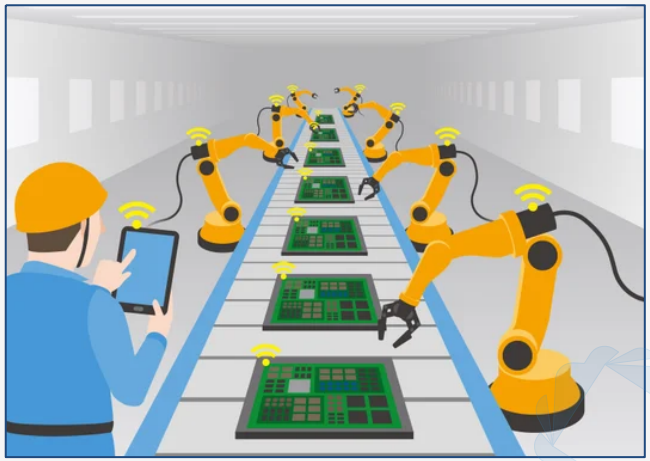
Seamless Integration

Connects tools like Python, R, Spark
—outputs flow as inputs

Command line is useful

FSDS
fullstackdatascience

Automated



Scriptable for recurring tasks—runs on servers anytime

Ubiquitous

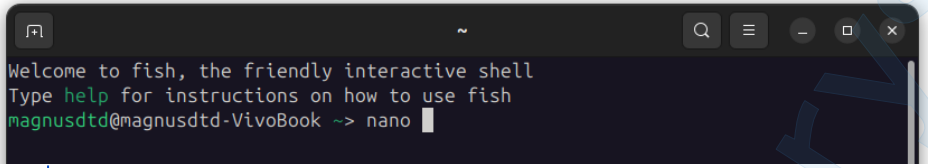
CLI tools present on all platforms—vital for cloud and servers.

Bash Scripting

Transform commands into automation !

Nano

Nano is a simple text editor that runs inside the terminal



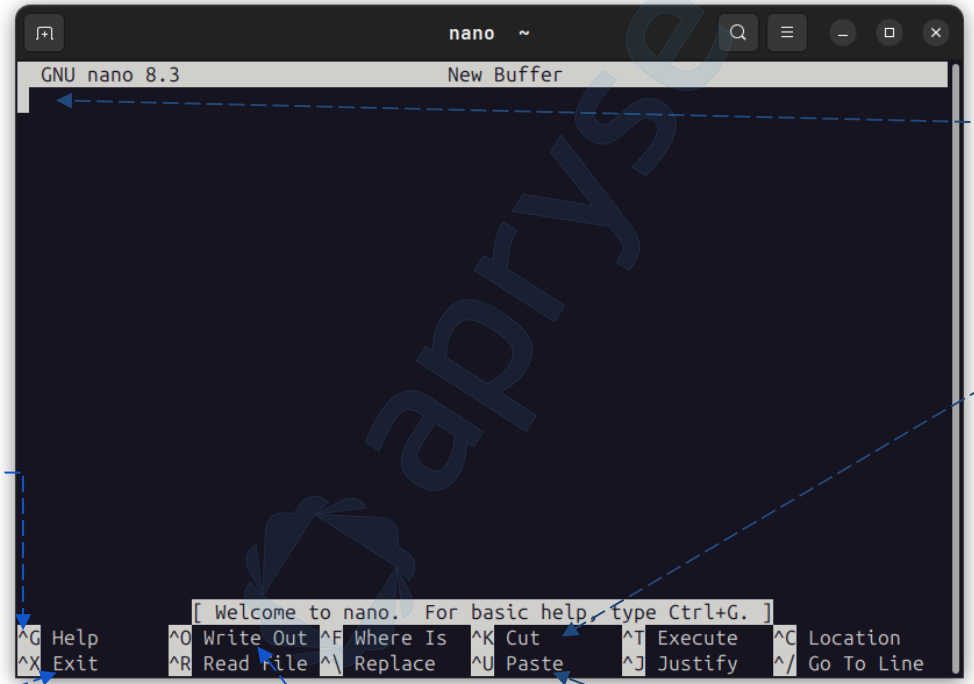
Type **nano** to open nano

It lets you create, open, and edit text files without leaving the command line

Comes pre-installed in most Linux systems, so it's always available.

Nano: Shortcut

FSDS
fullstackdatascience



The "^" means **CTRL**

When typing, you text will appear here

CTRL + K → Cut a line

CTRL + X → Exit

CTRL + O → Save (Write Out)

CTRL + U → Paste

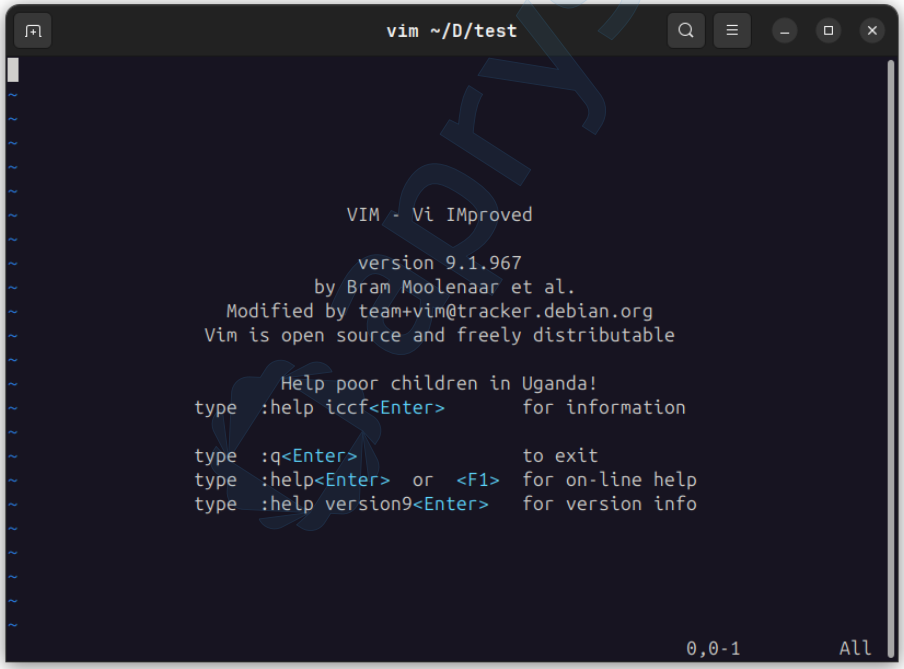
Vim

Vim is a simple text editor like **nano** that runs inside the terminal

Normal mode
(default)
→ move around,
delete, copy, paste

Insert mode
→ actually type text

Command mode
→ give commands
(like save or quit)



Fast
Lightweight
Ubiquitous

Not beginner-friendly
Takes time to master
Lacks GUI features

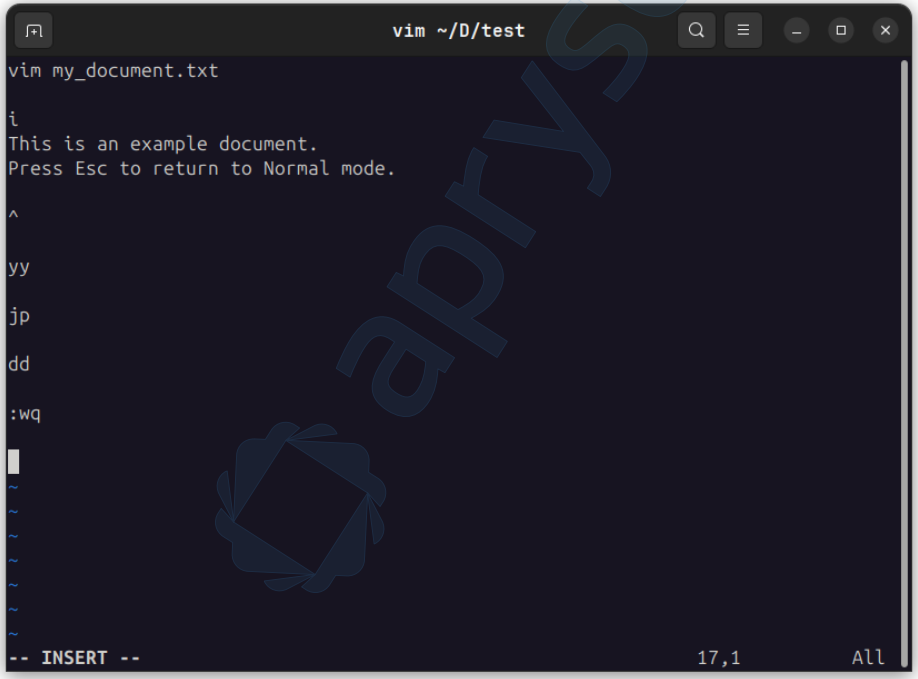
Vim

FSDS
fullstackdatascience

i → go to **Insert mode** (start typing)

Esc → go back to **Normal mode**

:w → save file



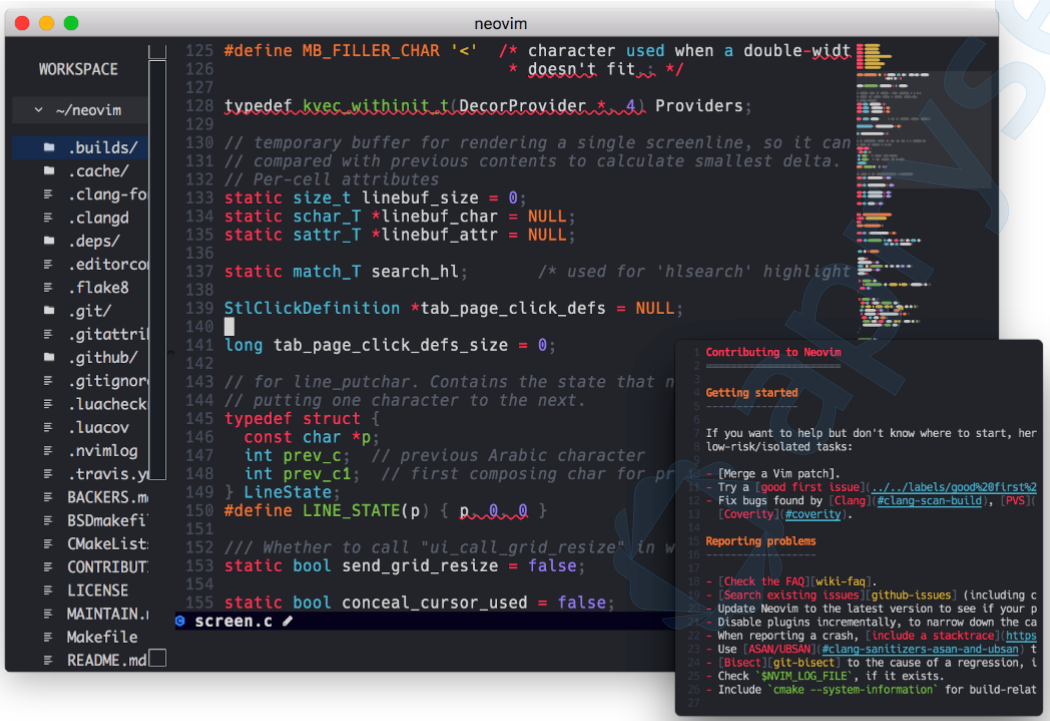
:q → quit vim

:wq → save **and** quit

:q! → quit **without saving**

Neovim

FSDS
fullstackdatascience



NeoVim is a fork (copy) of the Vim

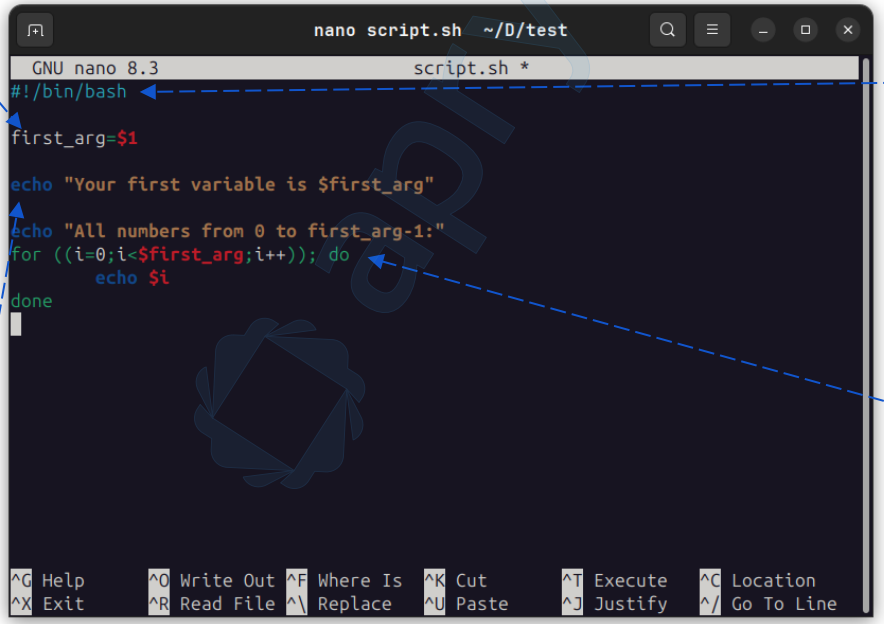
Highly popular among developers for deep **customization**

Bash Script

FSDS
fullstackdatascience

A bash script is like a recipe made of Linux commands

Declare a variable first arg to be your input argument



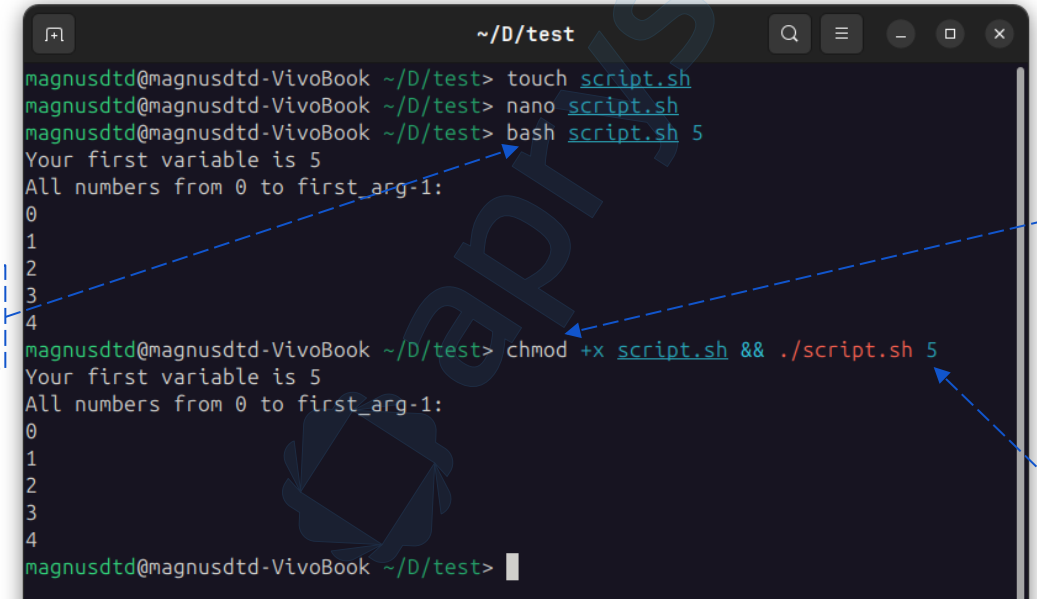
This line means to use the Bash shell to interpret

Print your first ever variable

Print all numbers from 0 to first arg-1

Bash Script: How to run

Save the script after editing it by **Ctrl + O** and **Ctrl + X**



Run with bash

Give execute permission

First argument

Bash Script: Syntax

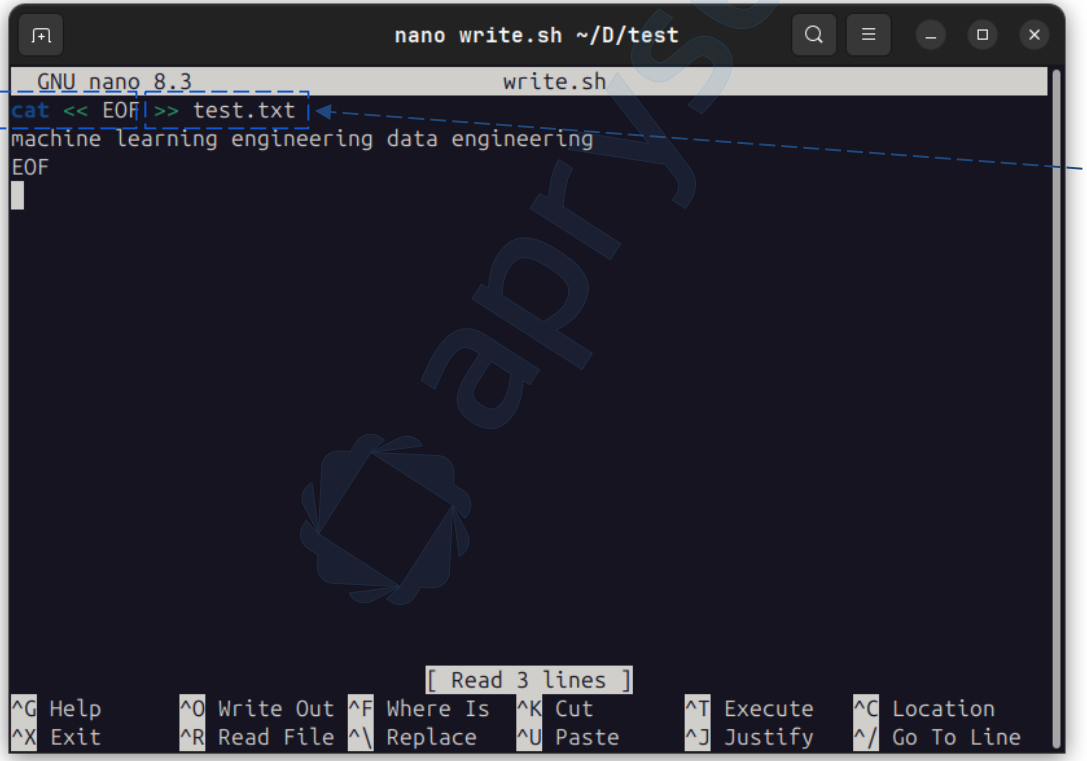
FSDS
fullstackdatascience

String

Bash Script: Syntax

FSDS
fullstackdatascience

Write to a file



starts a **here document**, letting you write multiple lines into a file.

appends the content to **test.txt**

If you use > test.txt, it **overwrites** the file instead of appending

Bash Script: Syntax

FSDS
fullstackdatascience

Array and condition

creates an array with 3 elements

gets the **first element**

gets the **last element**

gets the **last element**

string comparison

```
nano array.sh ~/D/test
 GNU nano 8.3                    array.sh
my_array=(10 2 300)
echo "First element: ${my_array[0]}"
echo "Last element: ${my_array[-1]}"
echo "Number of elements: ${#my array[@]}"
```

```
nano compare.sh ~/D/test
 GNU nano 8.3                    compare.sh *
if [[ 10 > 9 ]] then
        echo "10 > 9 is true"
else
        echo "10 > 9 is false"
fi
```

Don't use **[[10 > 9]]** but use **((10 > 9))** for numerical operations

Bash Script: Syntax

FSDS
fullstackdatascience

For loop

```
for ((i = 0; i < 5; i++)) do
        echo $i
done
```

While loop

```
i = 5
while (( $i > 0 )); do
        echo $i
        ((i = i - 1))
done
```

If else

```
#Demo if-else
if [[ $first_arg > 0 ]]; then
        echo "First argument is positive"
elif [[ $first_arg == 0 ]]; then
        echo "First argument is zero"
else
        echo "First argument is negative"
fi
```

Comment

Bash Script: Syntax

FSDS
fullstackdatascience

Function

Defining a Bash Function

Calling a Bash Function

```bash
add() {
    sum=$(($1 + $2))
    echo $sum
}
result=$(add 5 3)
echo "The sum is: $result" # Output: The sum is: 8
```

Another way to define a Bash function

```bash
function add {
    sum=$(($1 + $2))
    echo $sum
}
result=$(add 5 3)
echo "The sum is: $result" # Output: The sum is: 8
```

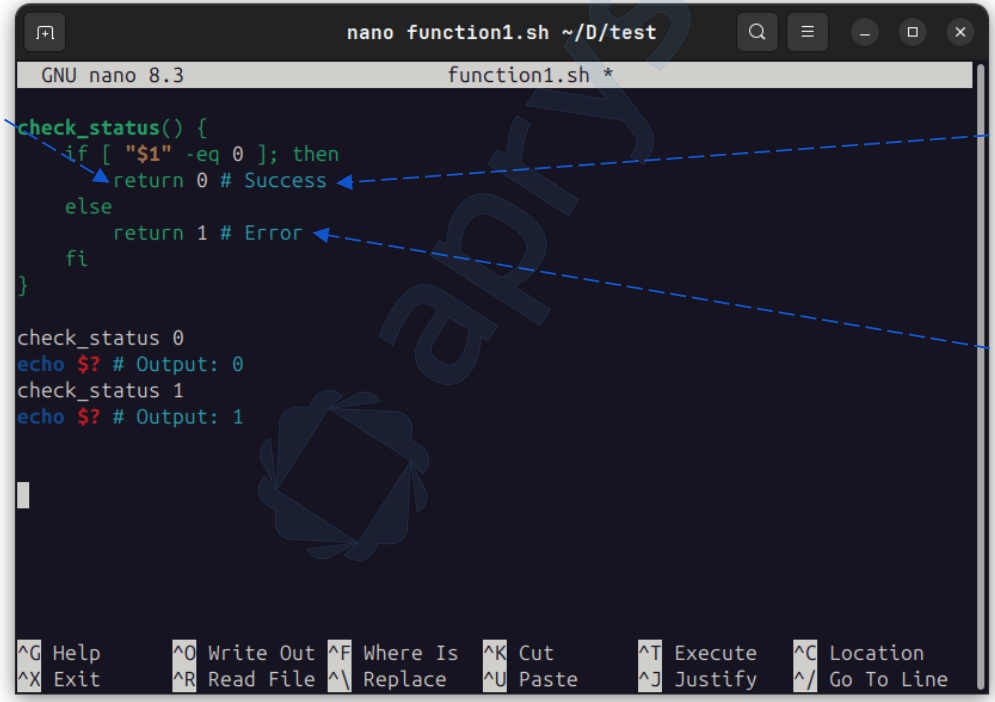
Bash Script: Syntax

FSDS
fullstackdatascience

Function

Using the **return** statement

This statement sets the exit status of the function (a numerical value between 0 and 255)



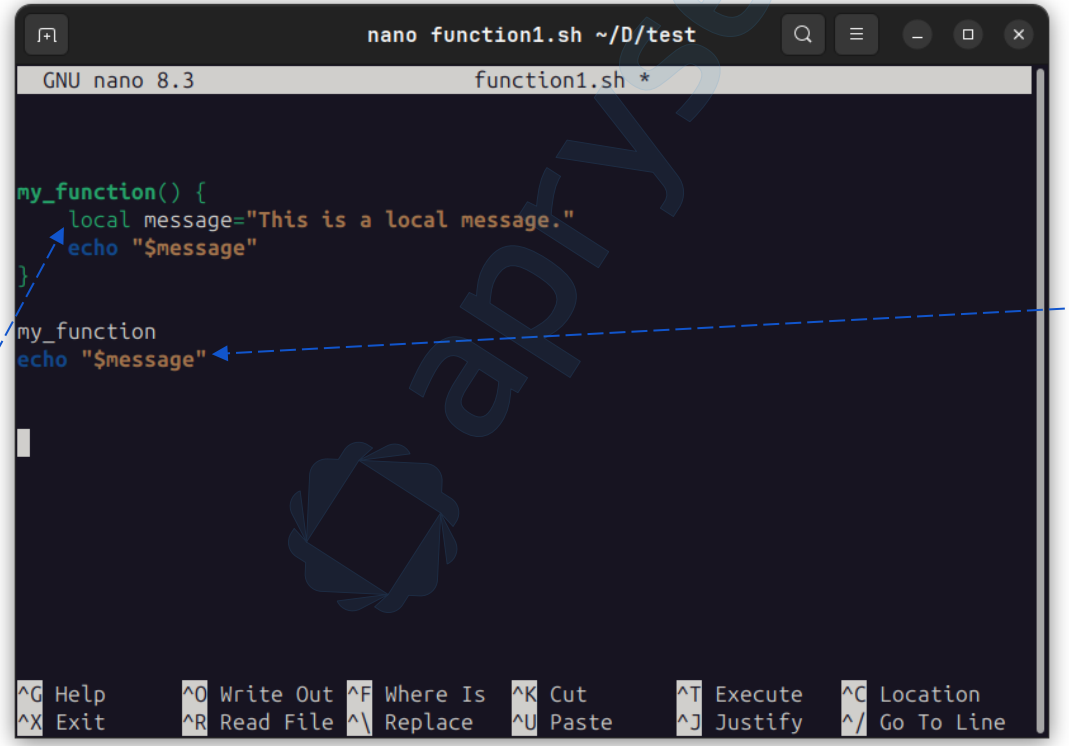
0 typically indicates success

non-zero values indicate an error

Bash Script: Syntax

FSDS
fullstackdatascience

Local variable

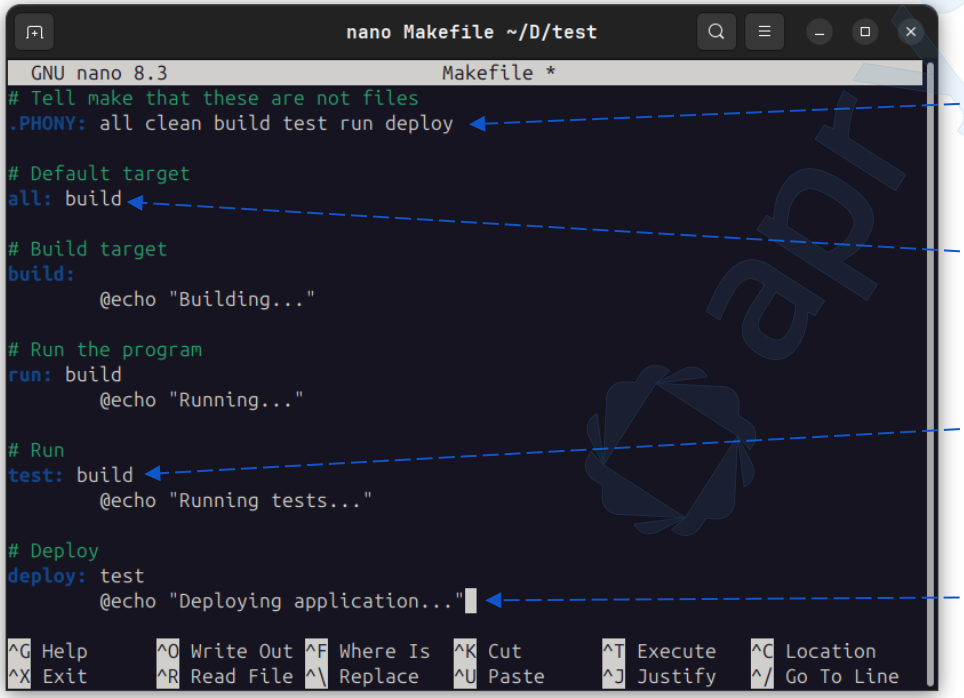


Defining a local variable

This will be empty or refer to a global variable if it exists

Makefile

A **Makefile** is like a *to-do list* that automates tasks such as compiling code, running tests, or deploying an application



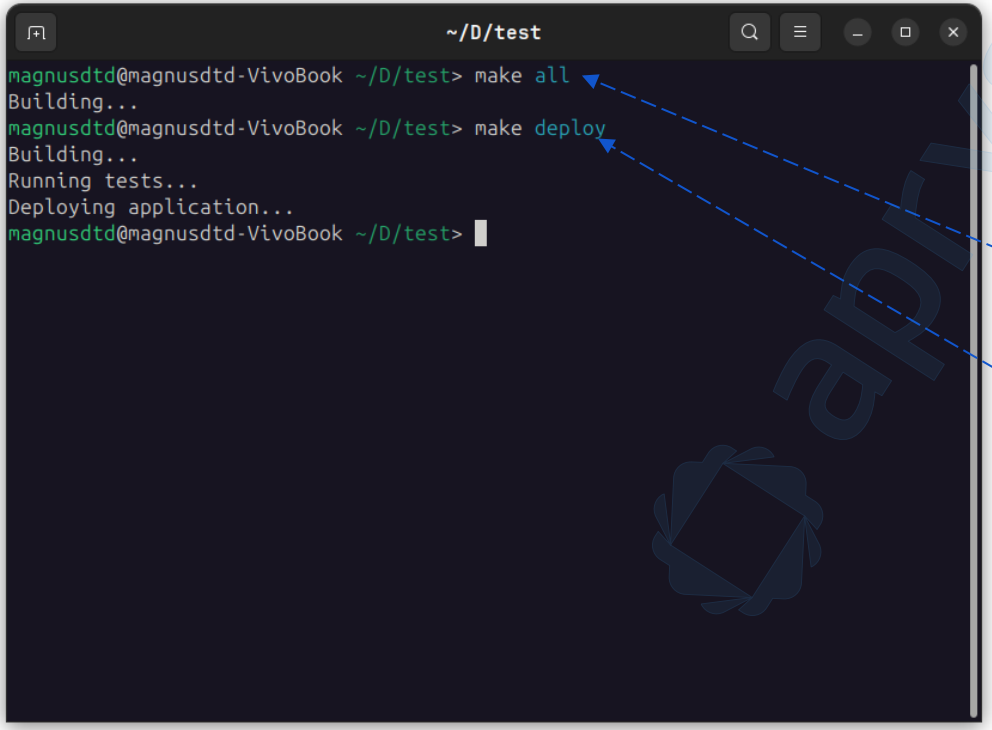
.PHONY
→ marks targets that are not files

Target → a task name

Dependencies
→ what must be done before the task

Recipe
→ commands to execute the task

Makefile: How to run



Use the command:

make + <target>

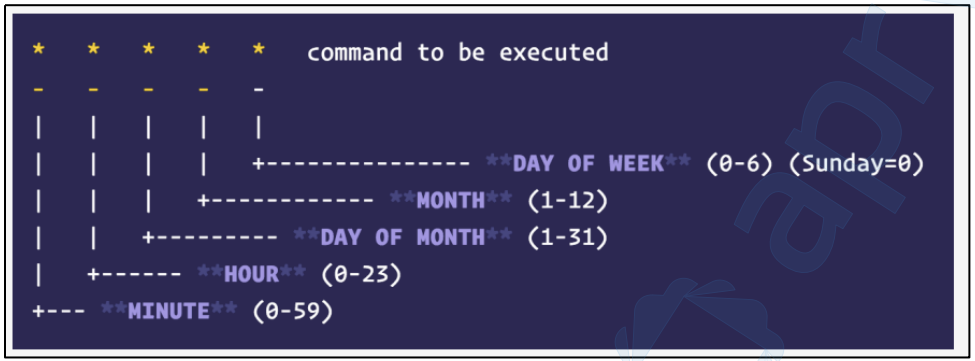
Run the **default** target

Run the **deploy** target

→ Saves time by running repetitive tasks automatically

CronJob

A **cronjob is like an alarm clock** for your Linux tasks.



Source: https://ahmadawais.com/setup-cron-in-unix-basic-understanding/

Examples:

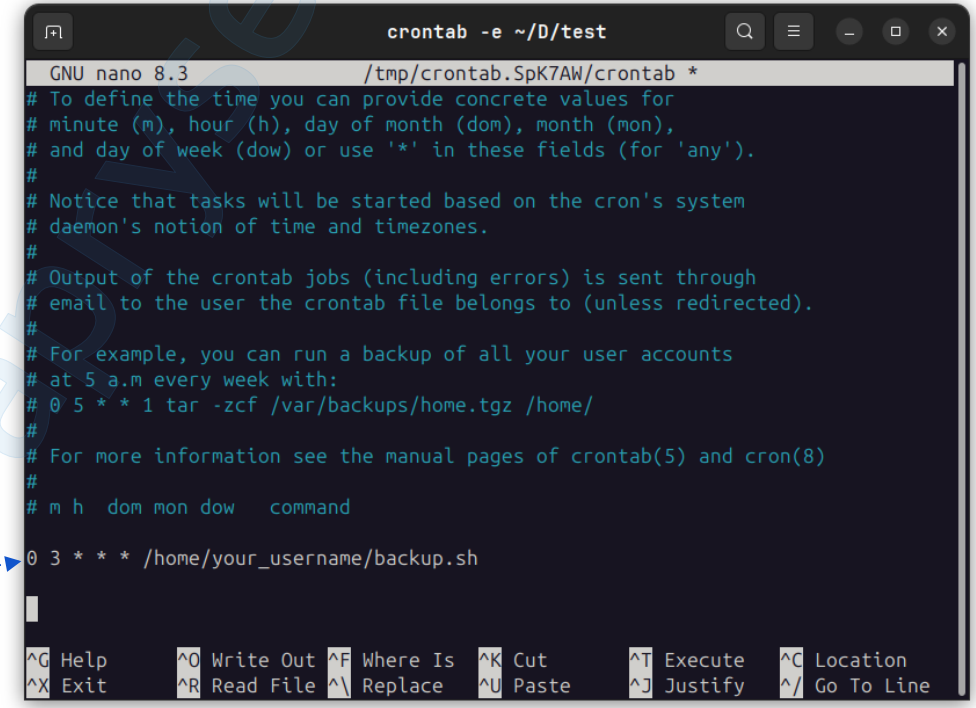
- **5 4 * * *** → At 04:05 everyday
- ***** → Every minute
- **00 03 25 09 *** → At 03:00 on Sep 25th

CronJob: How to run

List all crontabs: **crontab -l**

Add a crontab: **crontab -e**

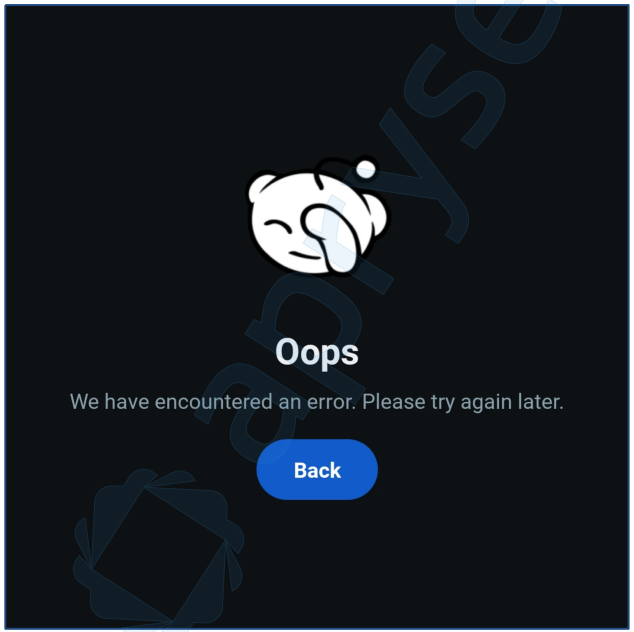
Run a script named **backup.sh** located in your home directory **every day at 3:00 AM**



CronJob: Limitations

FSDS
fullstackdatascience

No auto-retry



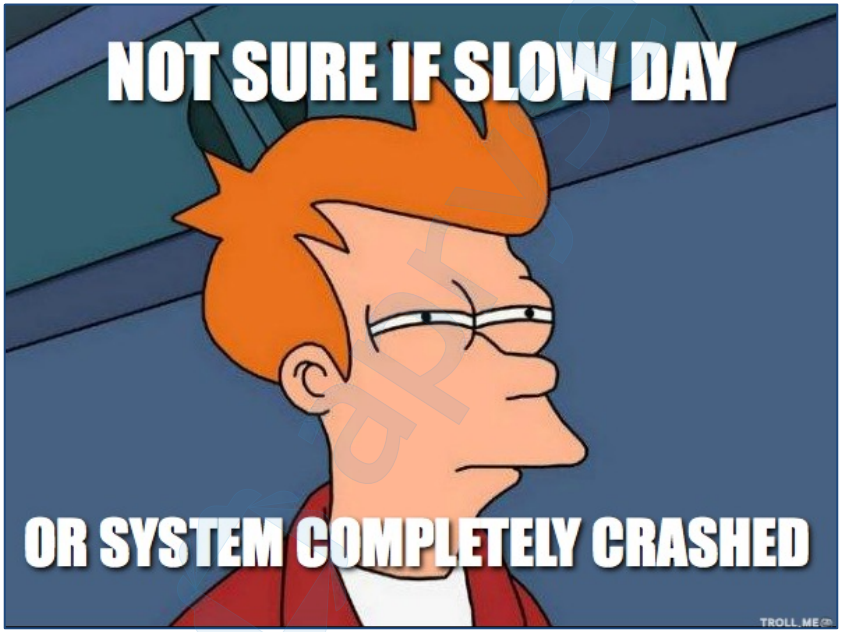
If a job fails, cron will not automatically retry the task

It will only run again at the next scheduled time

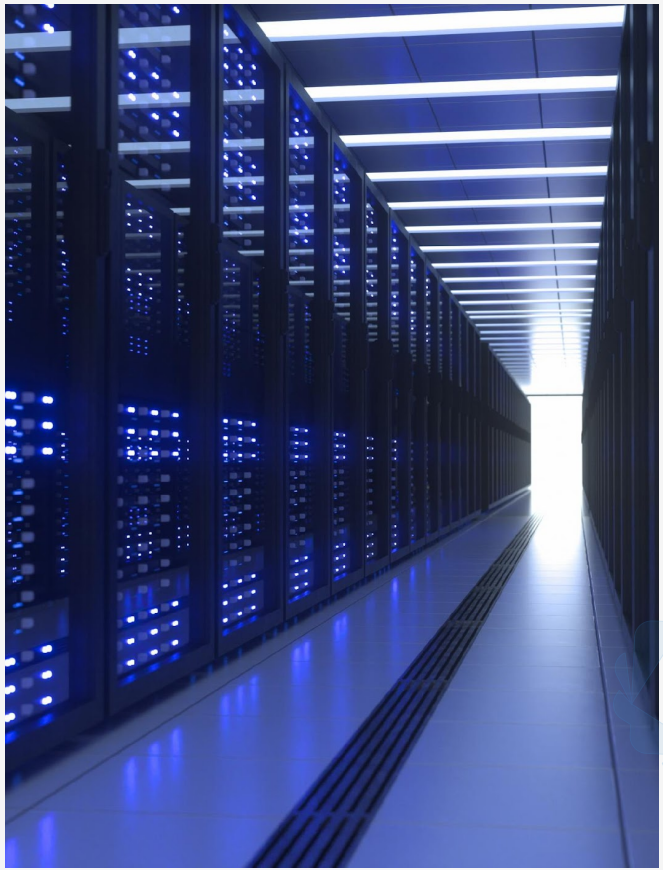
CronJob: Limitations

FSDS
fullstackdatascience

Missed jobs



If the system is powered off or the cron daemon crashes, any jobs that were scheduled during that time will not run and must be reset manually



FSDS
fullstackdatascience

fullstackdatascience.com

References

- https://www.cis.rit.edu/class/simg211/unixintro/Access_Permissions.html
- https://linuxize.com/post/chmod-command-in-linux/
- https://labs.iximiuz.com/playgrounds/ubuntu
- https://linuxhandbook.com/linux-directory-structure/
- https://www.geeksforgeeks.org/sed-command-in-linux-unix-with-examples/

FSDS
fullstackdatascience

Thank you