Linux Architecture & File Systems

Session 3

Linux Architecture & File Systems

Session 3

Objectives:

• Identify Linux system components (kernel, shell, userland)

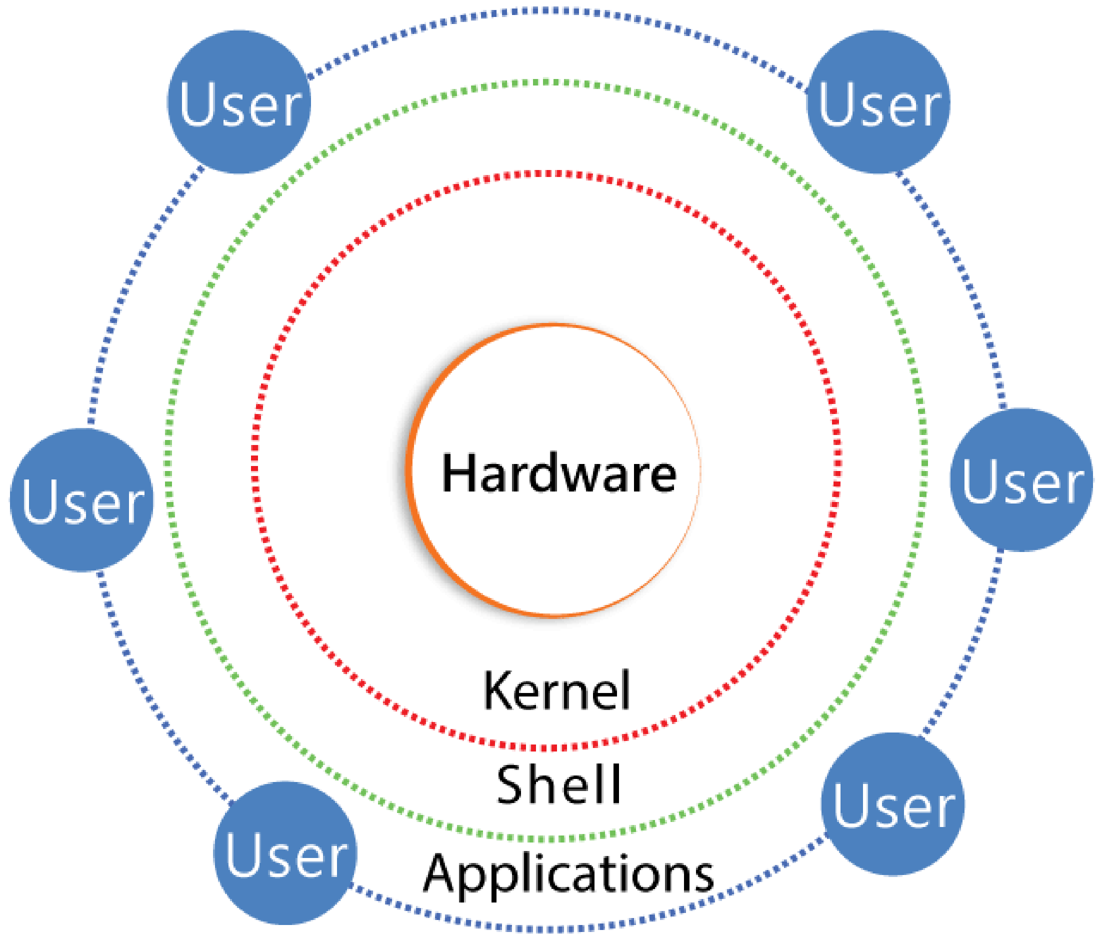
•Understand basic Linux boot process

•Explore key directories and their purposes

•Practice navigation and observation in Linux filesystem

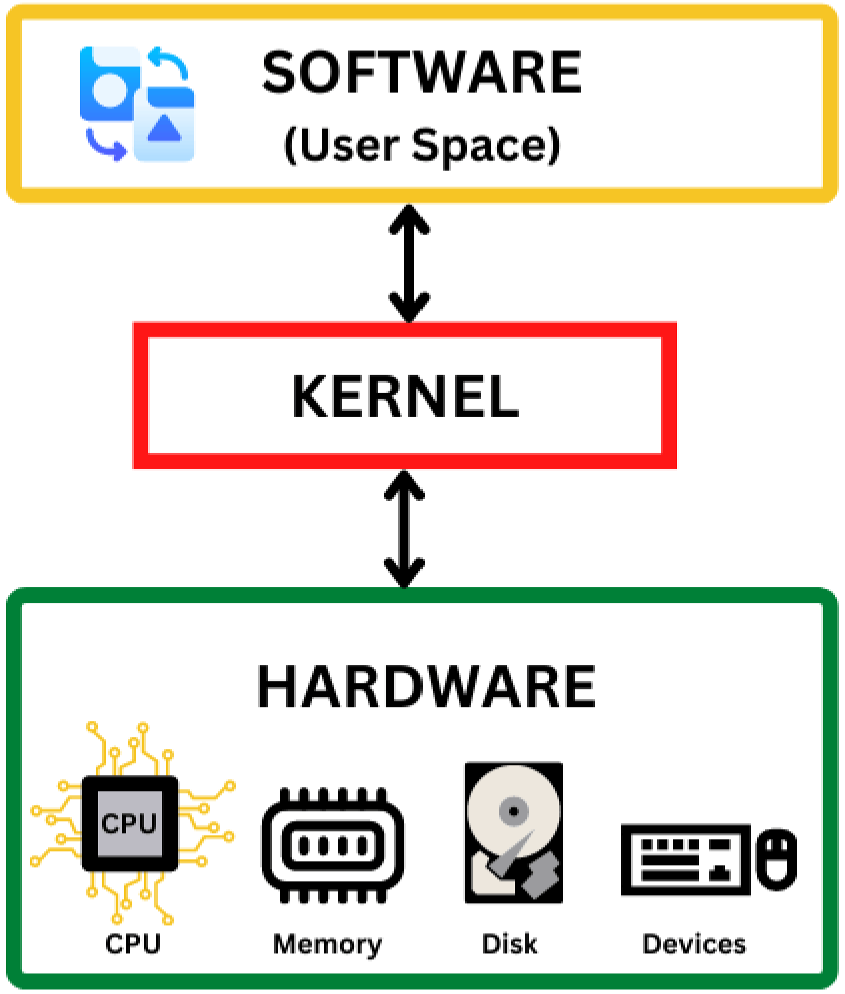
System structure

- **Hardware:**
- **Kernel:**
- **Shell:**
- **User Applications:**



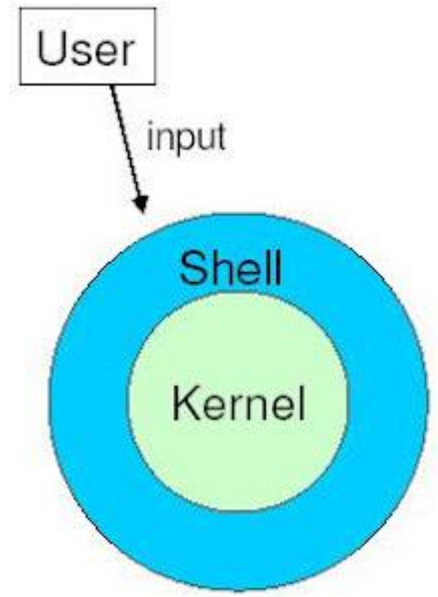
Kernel: core of Linux

- Operating system "kernel" is the core software used to "talk" to computer hardware
- It's a core and modular system of drivers used to create a standardized environment for interfacing with hardware
- Resource manager for allocating memory and time to system and user processes as well as interacting with files (I/O)
- Kernel operates in its own memory or "kernel-space"



Shell: user interface

- On user log-in, the system runs a shell
- A shell is the environment within which you will interface with the kernel via commands
- It determines the syntax for complex command-line operations and shell scripting.
- The shell you're using is called "bash," the successor to the venerable "Bourne Shell" called "sh"
- BASH: "Bourne Again SHell"



More Shell

- What shell am I in?

Typing “echo $SHELL” will show you!

- You should see ‘/bin/bash’
- Typing “echo $0” will also show your shell
- $SHELL and $0 are shell variables…more about variables later
- List of available shells on the system can be displayed by typing “chsh --list-shells”

The chsh command can be used to change your default shell as well

Shell Preference

- When you login, startup scripts are run to setup your environment

- For bash, you can customize your environment by adding or modifying environment variables and aliases in the **.bashrc** file in your home directory.
- Examples:

-alias ls=‘ls -rtl’ alias

-bwulf='ssh $USER@epita.fr’

-PATH=$PATH:/data/myusername

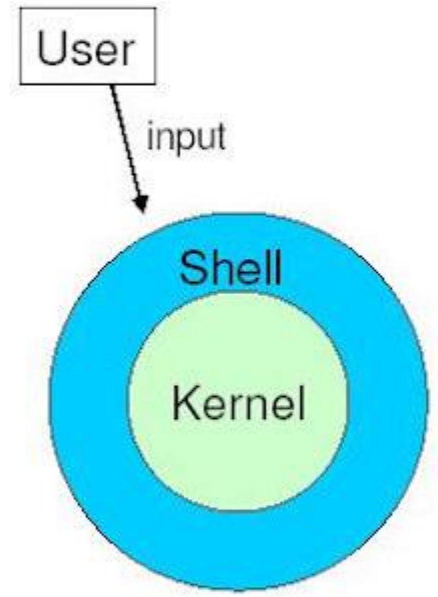
-EDITOR=/usr/bin/vim

Different Shells

- sh – the original UNIX shell (Bourne shell)
- bash – written as a replacement/extension of sh
- csh – C shell based on the C programming language developed in the late 1970s
- tcsh – enhanced version of C shell
- ksh – Korn shell developed in the early 1980’s, backward compatible with sh, but with some features of csh
- zsh – extended version of sh, developed in 1990
- dash – developed as replacement for ash in Debian

Applications

- Applications: browsers, text editors, games
- Services: web servers, mail servers
- Everything running on top of shell and kernel



How Linux boots?

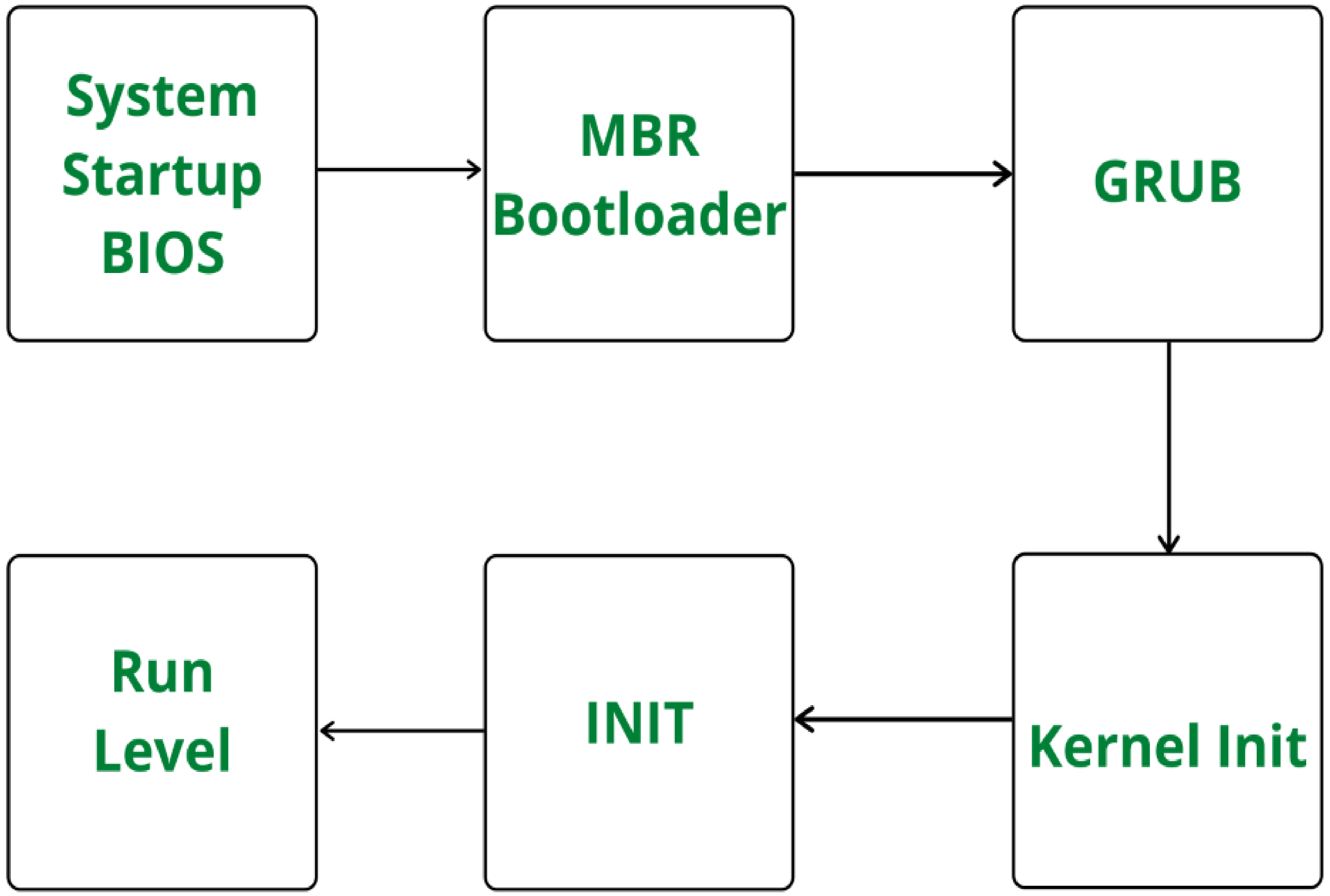
Basic Boot Process

1. BIOS/UEFI loads the bootloader

2. Bootloader loads the kernel

3. Kernel starts init/system

4. Systemd launches services and login



Booting Process

BIOS/UEFI Initialization

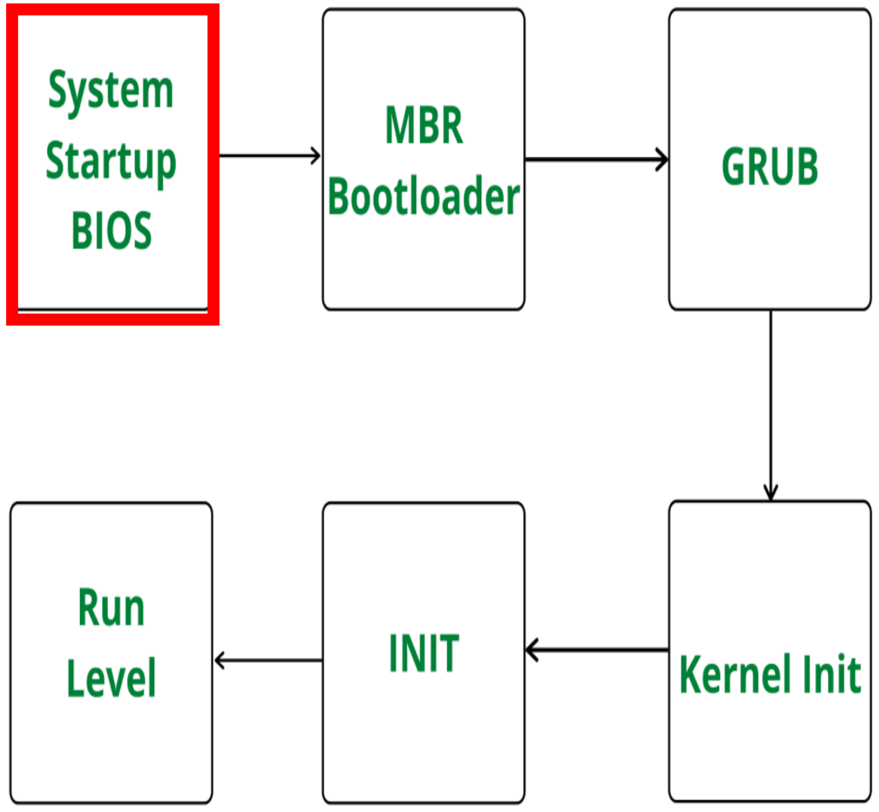
BIOS or UEFI

•The machine's BIOS (Basic Input/Output System) or UEFI initializes hardware and runs a Power-On Self-Test (POST) or a boot loader

•**Power-On Self-Test (POST):** basic device check on the most important hardware components which includes the RAM, CPU, and storage devices (HDD, SSD, and USB)

•**Storage Initialization:** find and set up connected storage devices while looking for media files that can be used to boot the computer.

•**Bootloader Search:** BIOS will look into the MBR on the first storage device to find the bootloader (like GRUB)



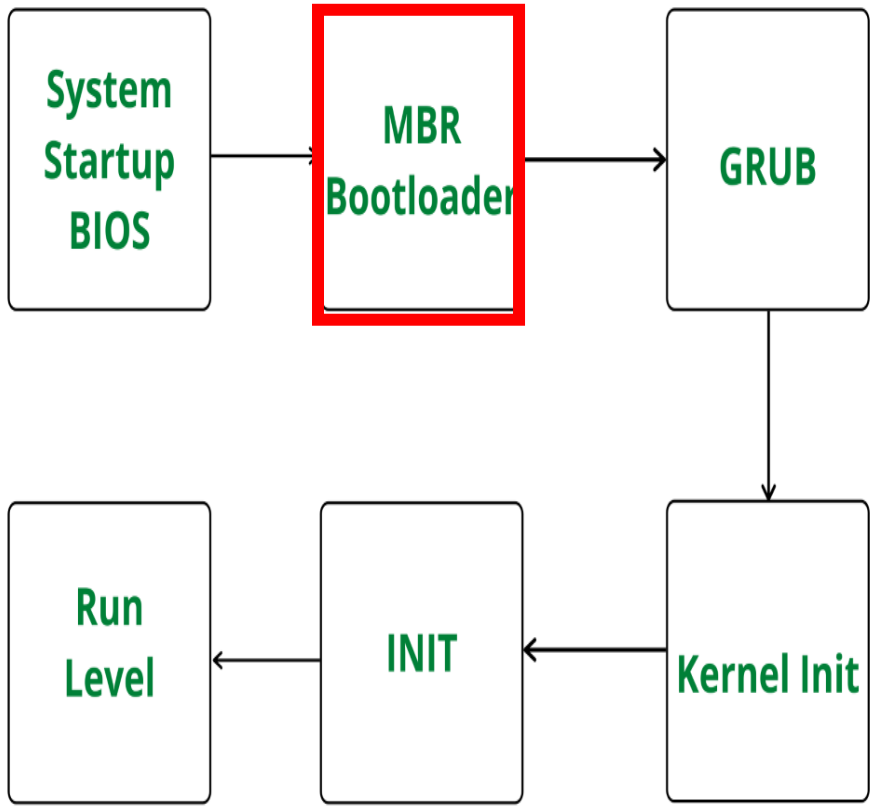
Booting Process

MBR Bootloader

- MBR stands for **Master Boot Record**, and is responsible for loading and executing the GRUB boot loader.

- Only 512 bytes!!

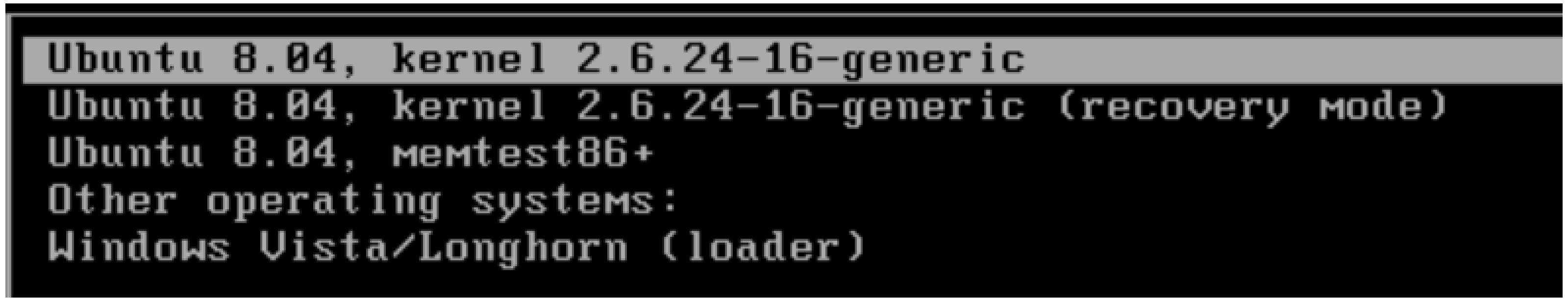
- located in the 1st sector of the bootable disk, which is typically **/dev/hda, or /dev/sda**. The MBR also contains information about GRUB



Booting Process

GRUB: GRand Unified Bootloader

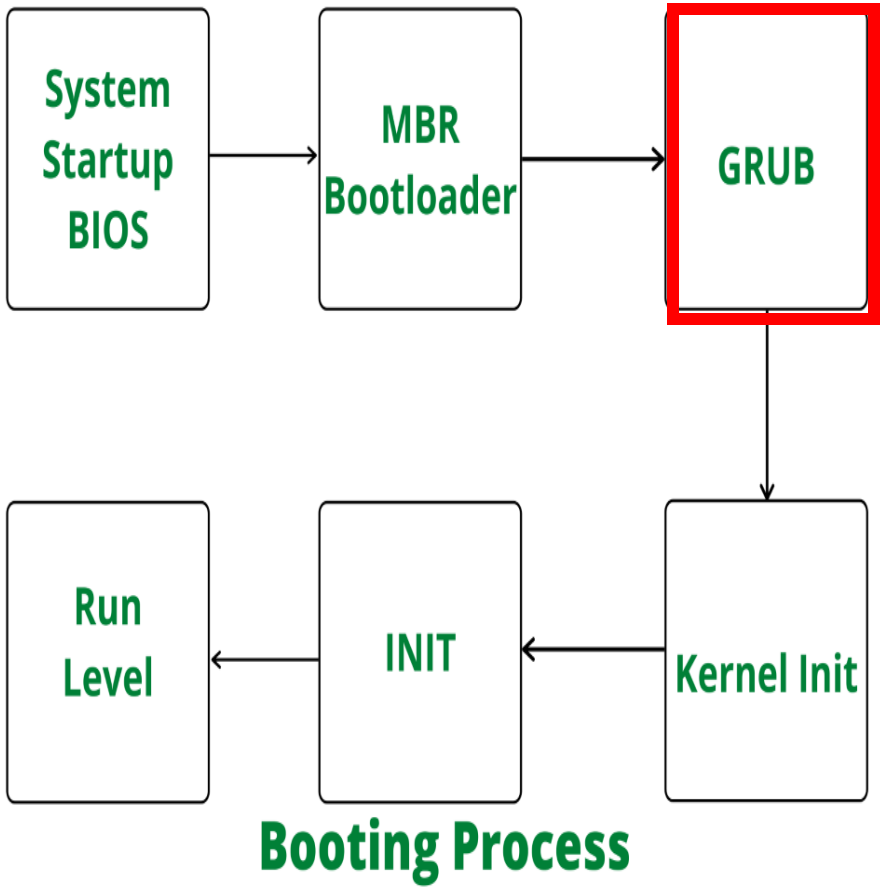
- GRUB is the most common (LILO, syslinux, etc..) and it is the first thing you see when you boot your computer. It has a simple menu where you can select some options (e.g. kernel).



```
Ubuntu 8.04, kernel 2.6.24-16-generic
Ubuntu 8.04, kernel 2.6.24-16-generic (recovery mode)
Ubuntu 8.04, memtest86+
Other operating systems:
Windows Vista/Longhorn (loader)
```

- GRUB configuration file at /boot/grub/grub.conf or /etc/grub.conf

```
#boot=/dev/sda
default=0
timeout=5
splashimage=(hd0,0)/boot/grub/splash.xpm.gz
hiddenmenu
title CentOS (2.6.18-194.el5PAE)
      root (hd0,0)
      kernel /boot/vmlinuz-2.6.18-194.el5PAE ro root=LABEL=/
      initrd /boot/initrd-2.6.18-194.el5PAE.img
```



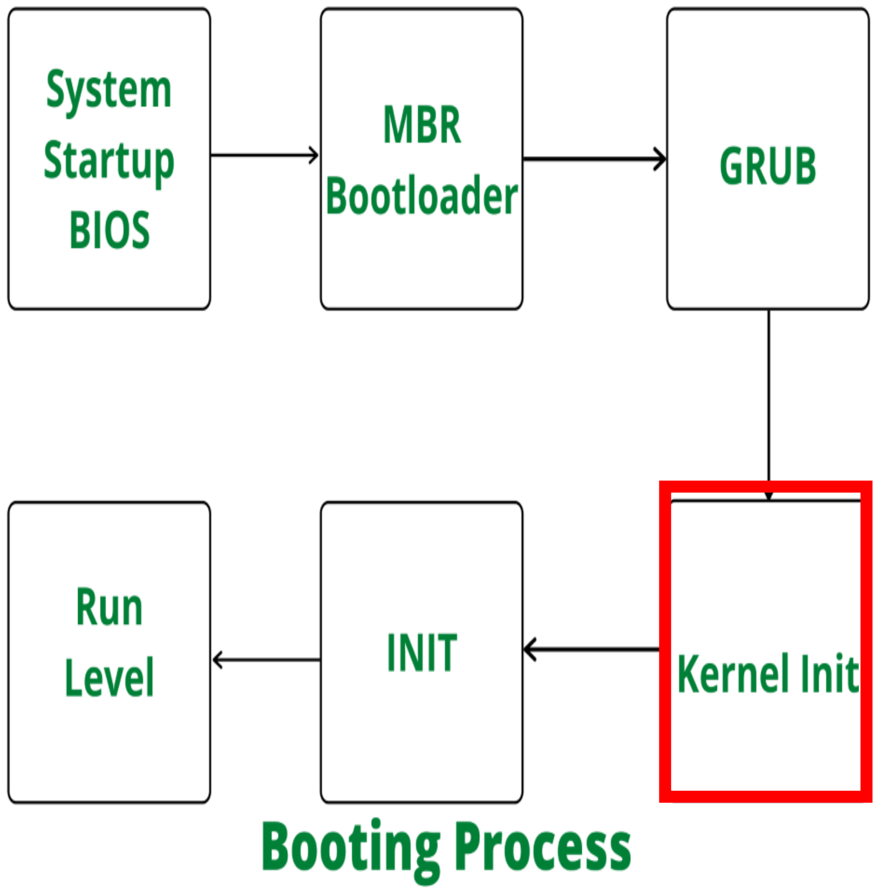
Kernel init

•It has complete control over everything in your system

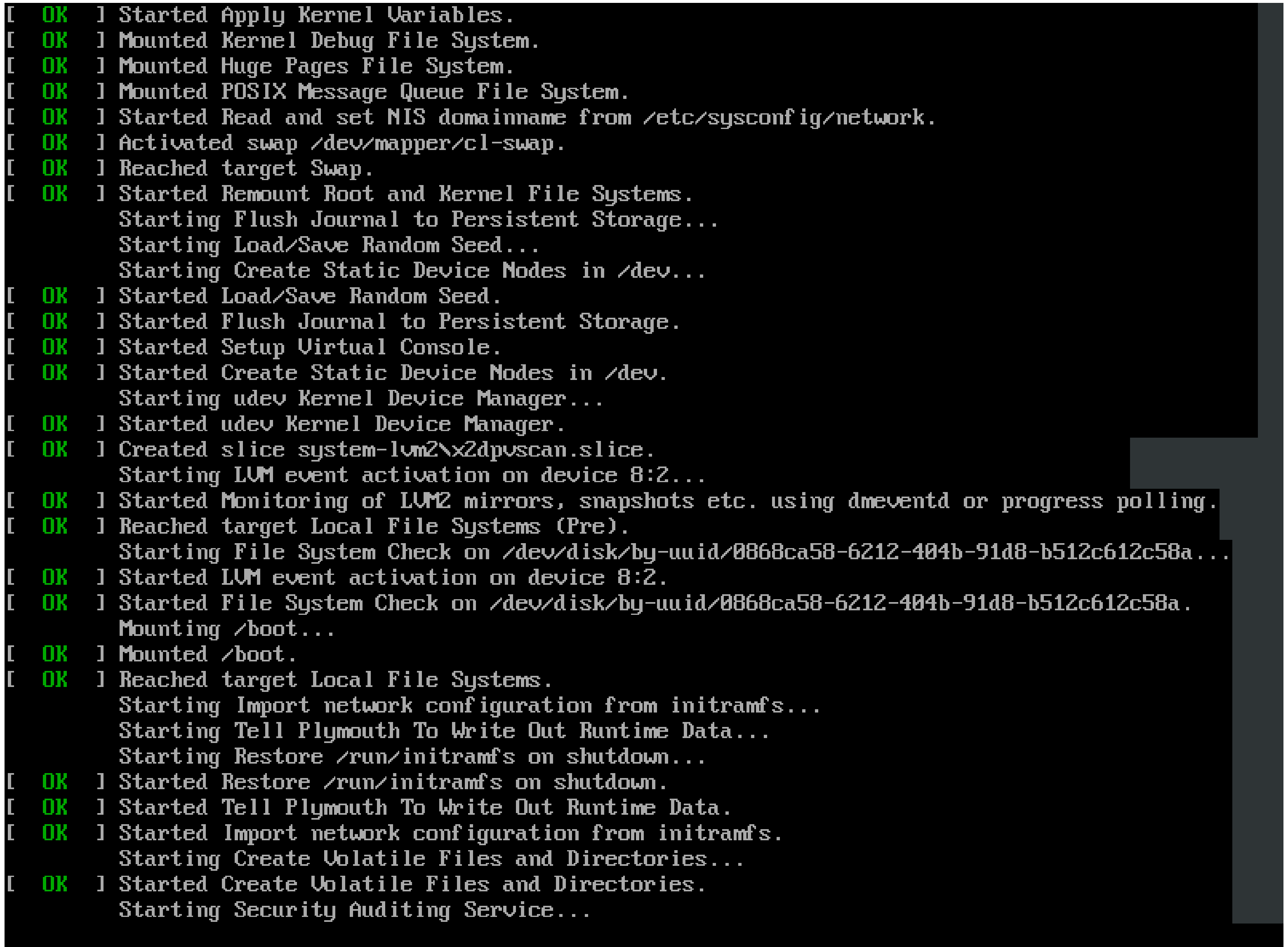
•it executes the /sbin/init program, which is always the first program to be executed. You can confirm this with its process id (PID), which should always be 1 (ps -ef | grep init)

•The kernel then establishes a temporary root file system using Initial RAM Disk (initrd) until the real file system is mounted.

Initrd: A compressed **block-based filesystem** that includes essential **drivers and utilities** needed for booting.



Kernel init

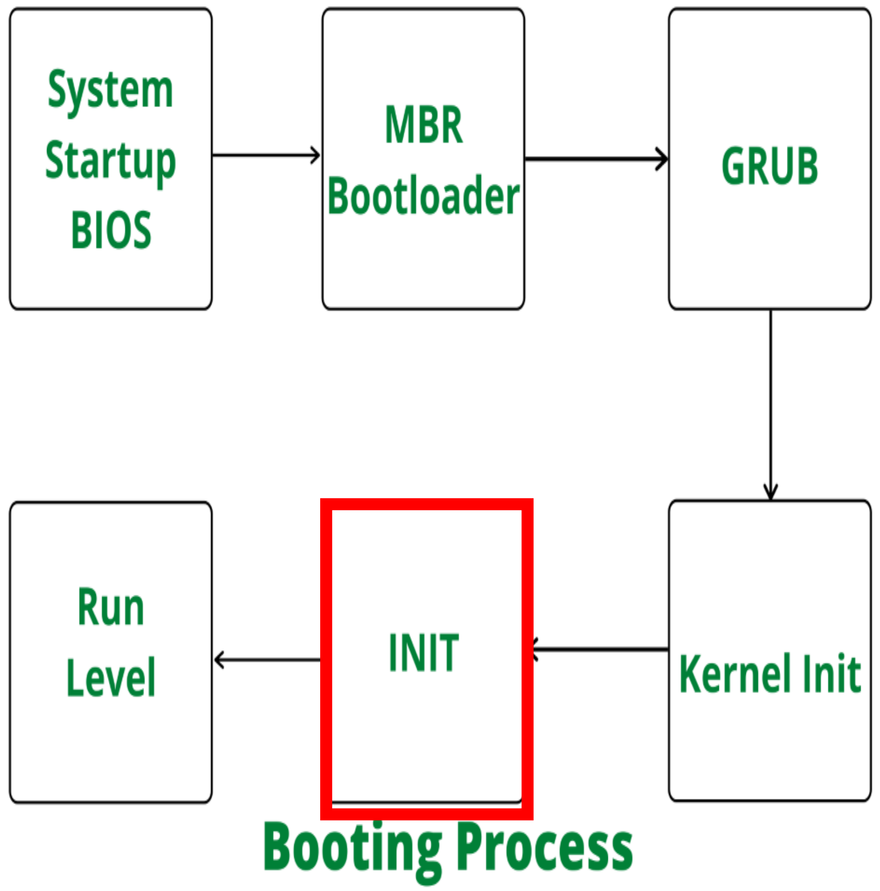


```
[  OK  ] Started Apply Kernel Variables.
[  OK  ] Mounted Kernel Debug File System.
[  OK  ] Mounted Huge Pages File System.
[  OK  ] Mounted POSIX Message Queue File System.
[  OK  ] Started Read and set NIS domainname from /etc/sysconfig/network.
[  OK  ] Activated swap /dev/mapper/cl-swap.
[  OK  ] Reached target Swap.
[  OK  ] Started Remount Root and Kernel File Systems.
         Starting Flush Journal to Persistent Storage...
         Starting Load/Save Random Seed...
         Starting Create Static Device Nodes in /dev...
[  OK  ] Started Load/Save Random Seed.
[  OK  ] Started Flush Journal to Persistent Storage.
[  OK  ] Started Setup Virtual Console.
[  OK  ] Started Create Static Device Nodes in /dev.
         Starting udev Kernel Device Manager...
[  OK  ] Started udev Kernel Device Manager.
[  OK  ] Created slice system-lvm2\x2dpvscan.slice.
         Starting LVM event activation on device 8:2...
[  OK  ] Started Monitoring of LVM2 mirrors, snapshots etc. using dmeventd or progress polling.
[  OK  ] Reached target Local File Systems (Pre).
         Starting File System Check on /dev/disk/by-uuid/0868ca58-6212-404b-91d8-b512c612c58a...
[  OK  ] Started LVM event activation on device 8:2.
[  OK  ] Started File System Check on /dev/disk/by-uuid/0868ca58-6212-404b-91d8-b512c612c58a.
         Mounting /boot...
[  OK  ] Mounted /boot.
[  OK  ] Reached target Local File Systems.
         Starting Import network configuration from initramfs...
         Starting Tell Plymouth To Write Out Runtime Data...
         Starting Restore /run/initramfs on shutdown...
[  OK  ] Started Restore /run/initramfs on shutdown.
[  OK  ] Started Tell Plymouth To Write Out Runtime Data.
[  OK  ] Started Import network configuration from initramfs.
         Starting Create Volatile Files and Directories...
[  OK  ] Started Create Volatile Files and Directories.
         Starting Security Auditing Service...
```

INIT

•**init system**: handles system services, processes, and sessions. The init system takes care of provisioning all required background services such as networking, logging.

•At one point it would look for an init file, usually found at /etc/inittab to decide the Linux run level. *grep initdefault /etc/inittab*



Run Level

•When the Linux system is booting up, you might see various services getting started. For example, it might say "starting sendmail .... OK". Those are the runlevel programs, executed from the run level directory as defined by your run level.

Systemd targets (replacing runlevels):

•0 → poweroff.target

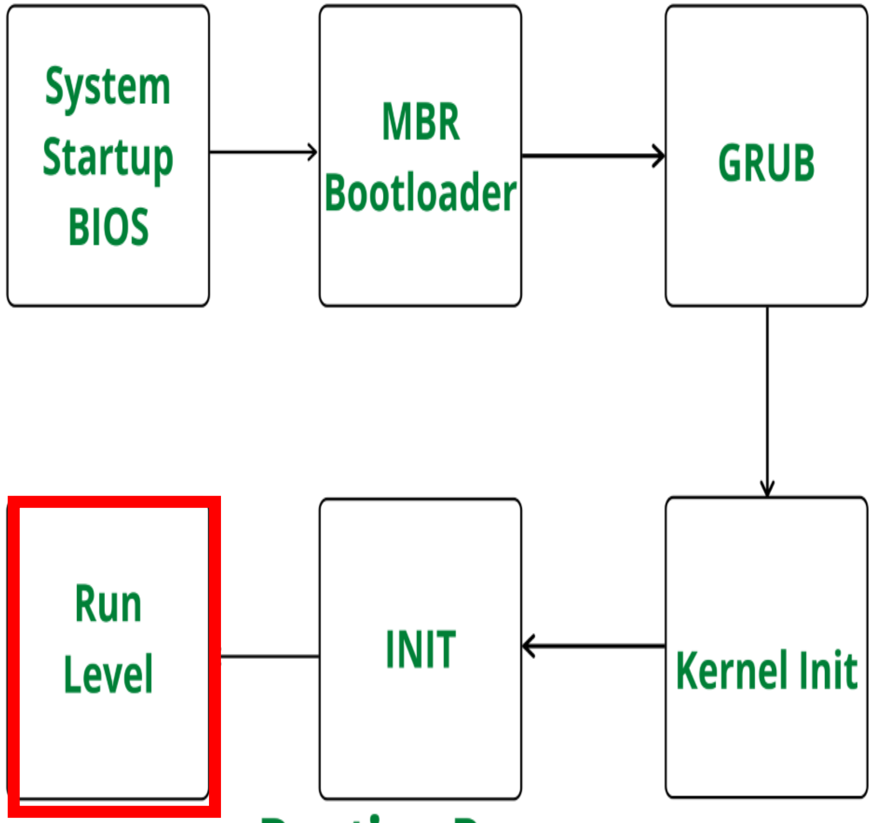
•1 → rescue.target

•3 → multi-user.target

•5 → graphical.target

•6 → reboot.target

•Emergency → emergency.target



Booting Process

Linux file system structure

What is a file system?

Linux file system structure

What is a file system?

•Organizes data on disk

•Stores files and directories

•Defines access permissions

Unix File System is a logical method of organizing and storing large amounts of information in a way that makes it easy to manage

Linux file system structure

Popular File Systems

- **FAT** (File Allocation Table)
    - Old file system used by MS-DOS, early Windows
- **NTFS** (New Technology File System)
    - Modern Windows file system (permissions, encryption)
- **ext (ext2/ext3/ext4)** (Extended Filesystem)
    - Common file system for Linux
- **HFS+** (Hierarchical File System Plus)
    - Older macOS file system
- **APFS** (Apple File System)
    - New file system for modern macOS and iOS

Linux file system structure

What is a file?

Linux file system structure

What is a file?

- **Ordinary files (-):** text, data, or program information. (avoid space or special character in name)
- **Directories (d):** containers or folders that hold files, and other directories. They do not contain files, only entries that point to them.
- **Devices:** Hardware accessed like files, like hard disks or printers — represented as files under /dev . e.g. /dev/tty for terminal
- **Links (l):** Shortcuts or references to files (hard/soft). A **hard link** is a second name for a file. A **symbolic (soft) link** is more like a shortcut pointing to another file or folder.

EVERYTHING IS A FILE

(even directories, devices, and links)

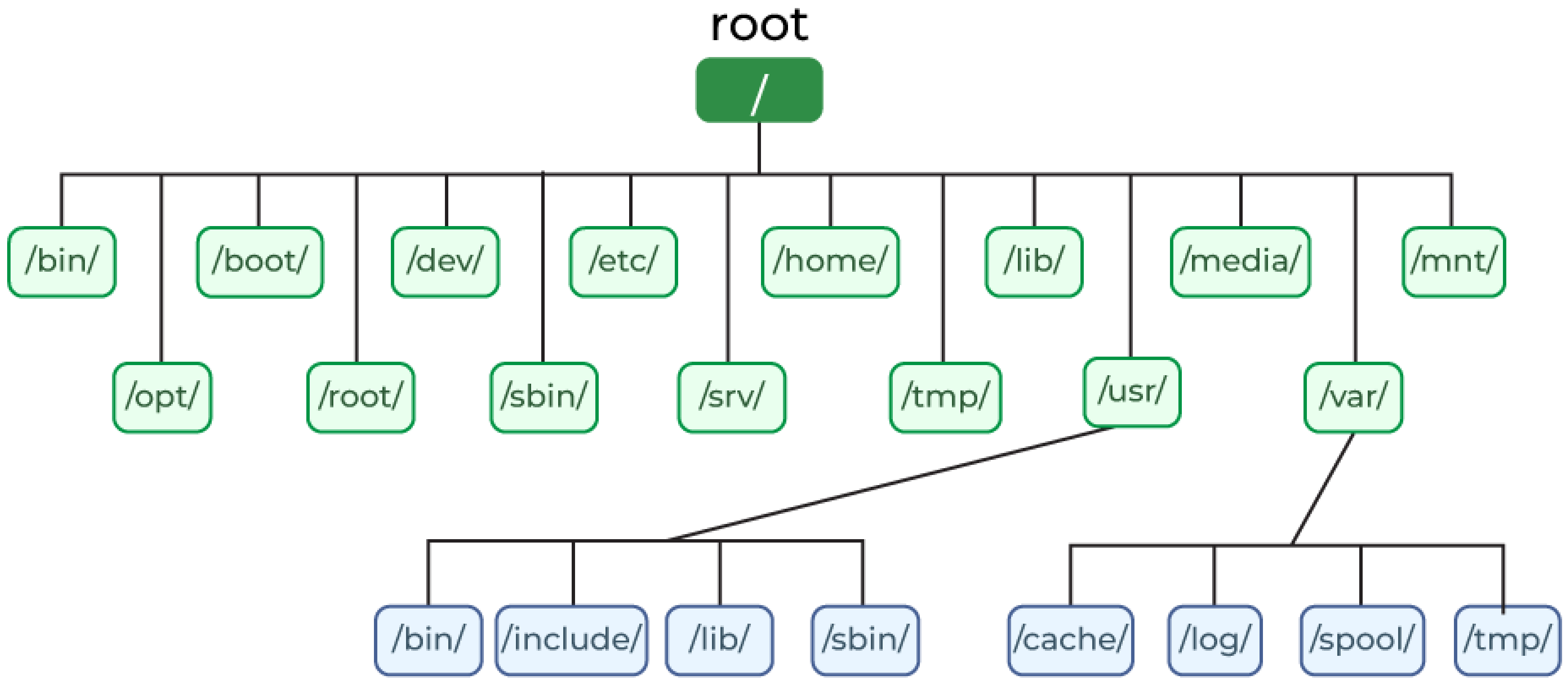
Linux file system structure

File convention?

- Filenames are **case-sensitive**. For example, **‘File.txt’** and **‘file.txt’** are considered two distinct files
- Special characters like **/, *, ?, :, $, %,** are not typically allowed in filenames.

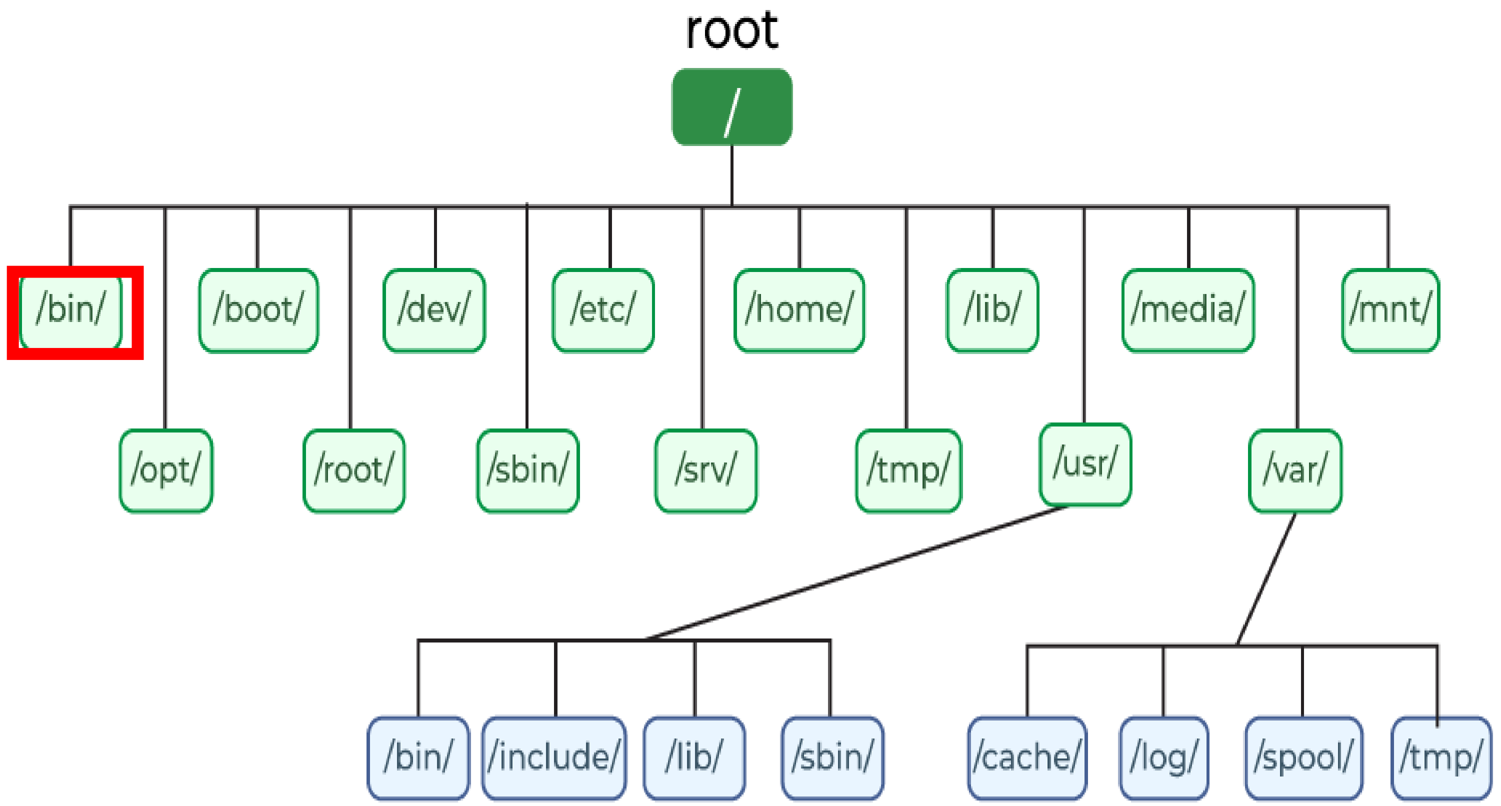
Directory structure

hierarchical tree structure which is anchored at a special top-level directory known as the root (designated by a slash '/').



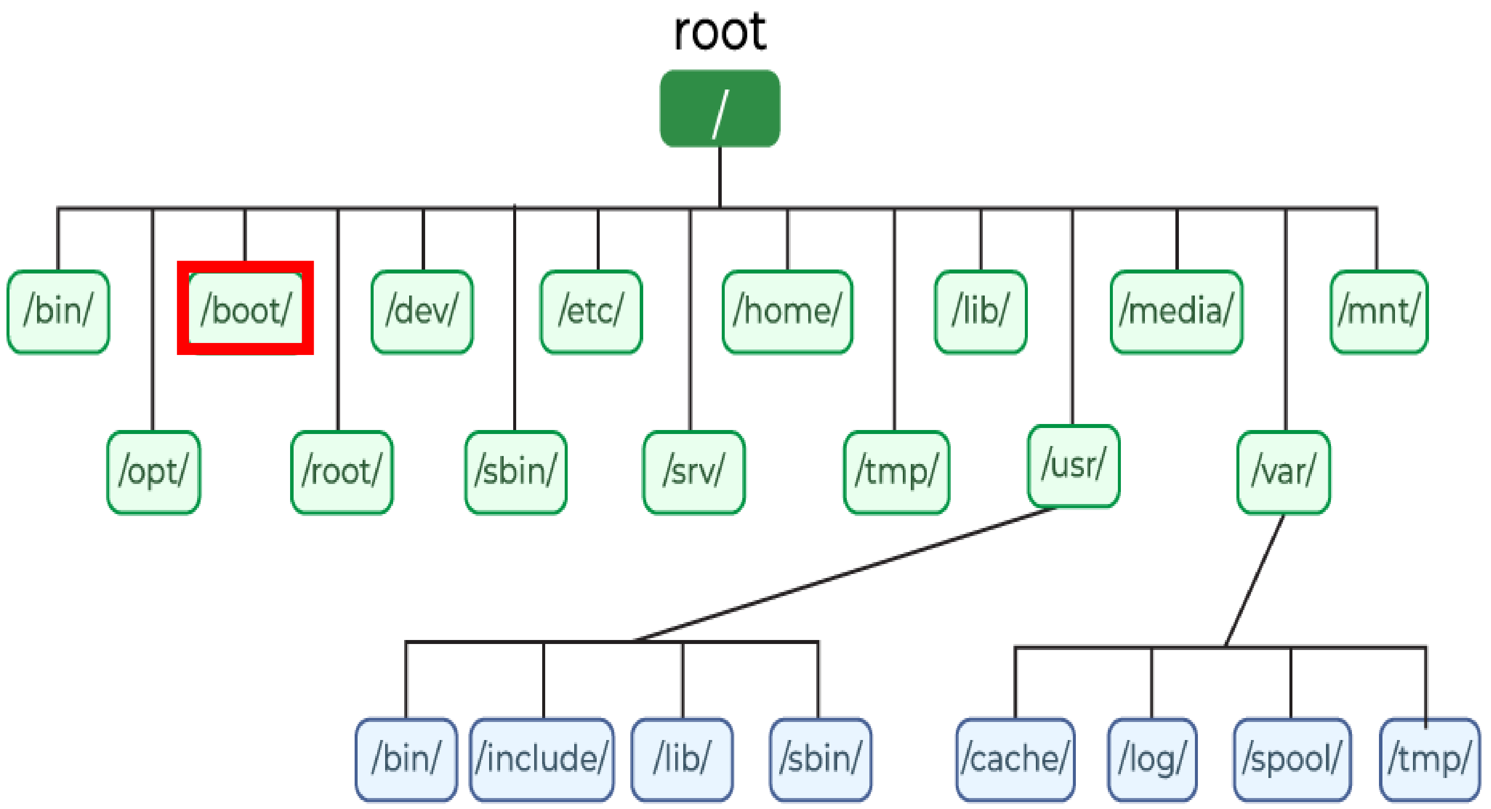
Directory structure

- /bin: Essential command binaries that need to be available in single user mode; for all users, e.g., cat, ls, cp.



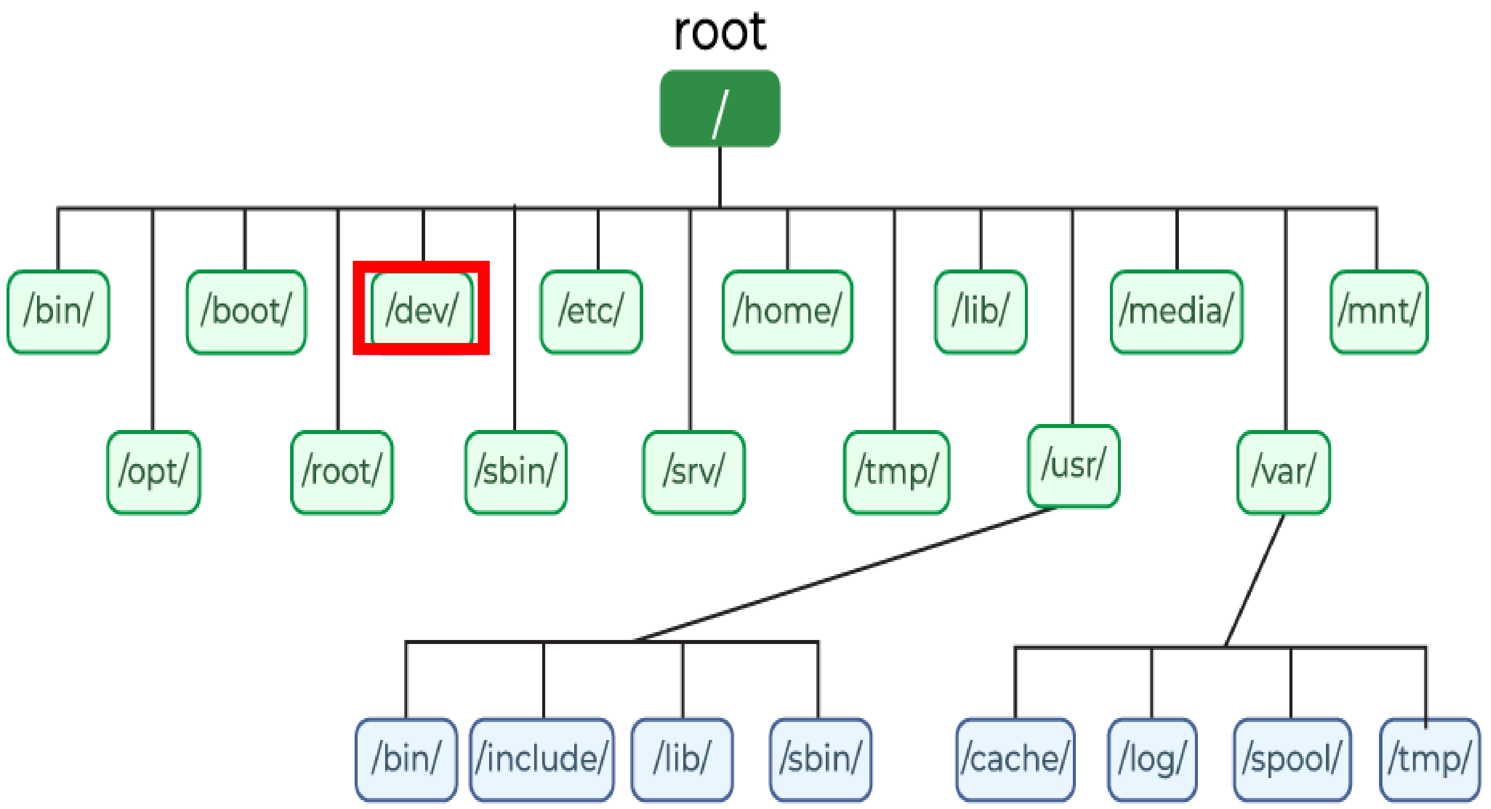
Directory structure

- /boot: Boot loader files, e.g., kernels, initrd.



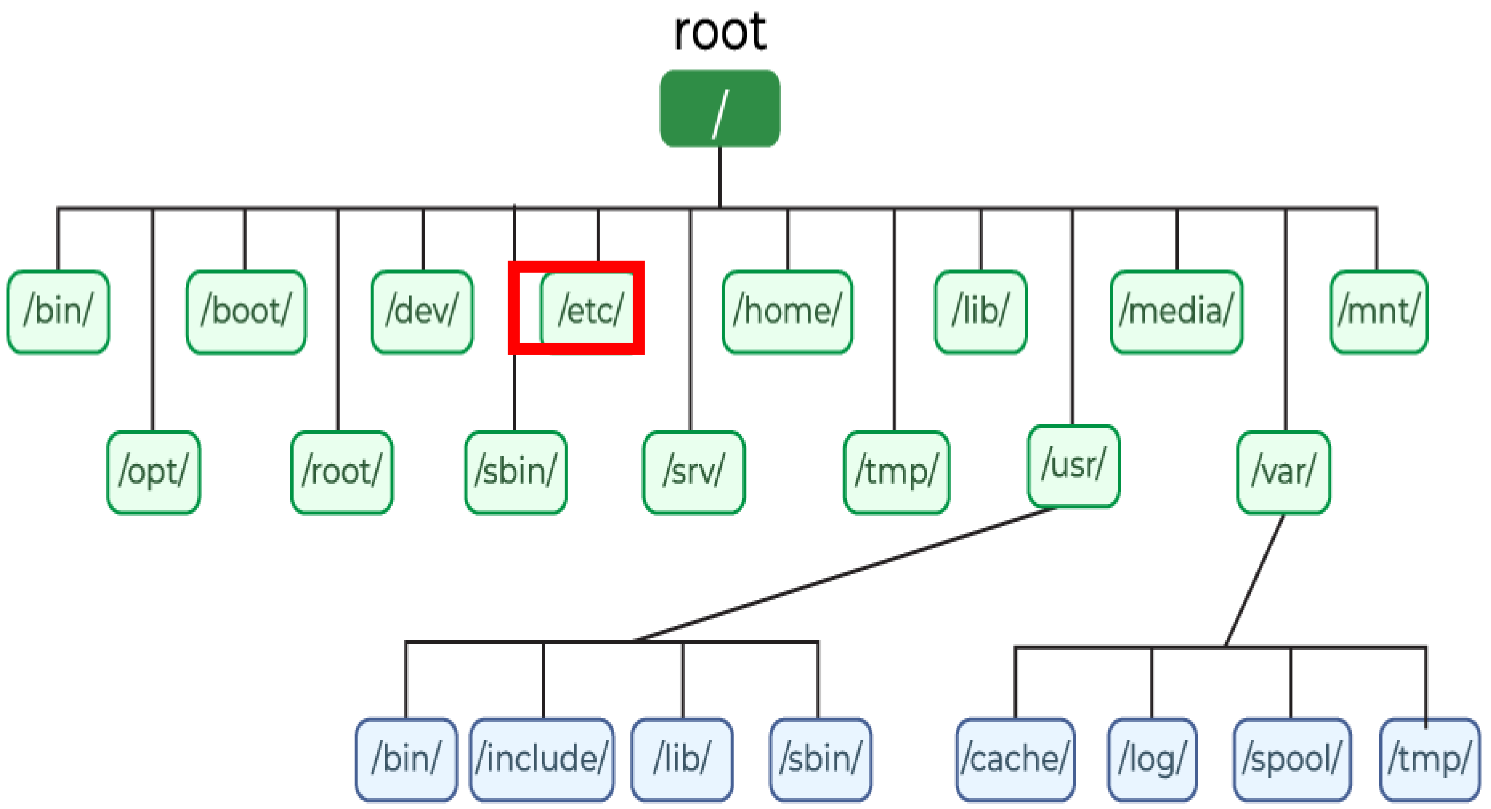
Directory structure

- /dev: system device files



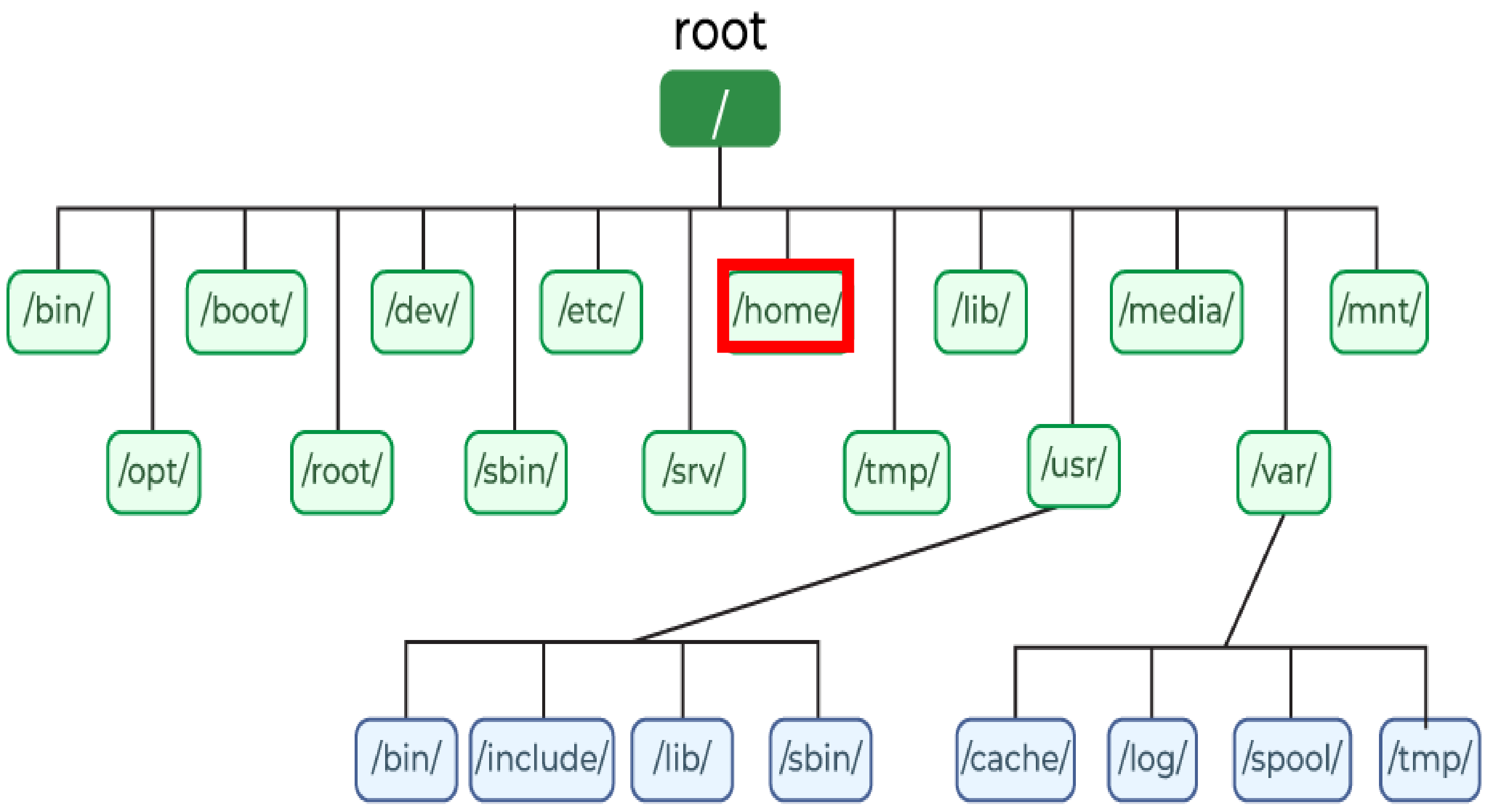
Directory structure

- /etc: Contains system-wide configuration files and system databases. Originally also contained “dangerous maintenance utilities” such as init,but these have typically been moved to /sbin or elsewhere.



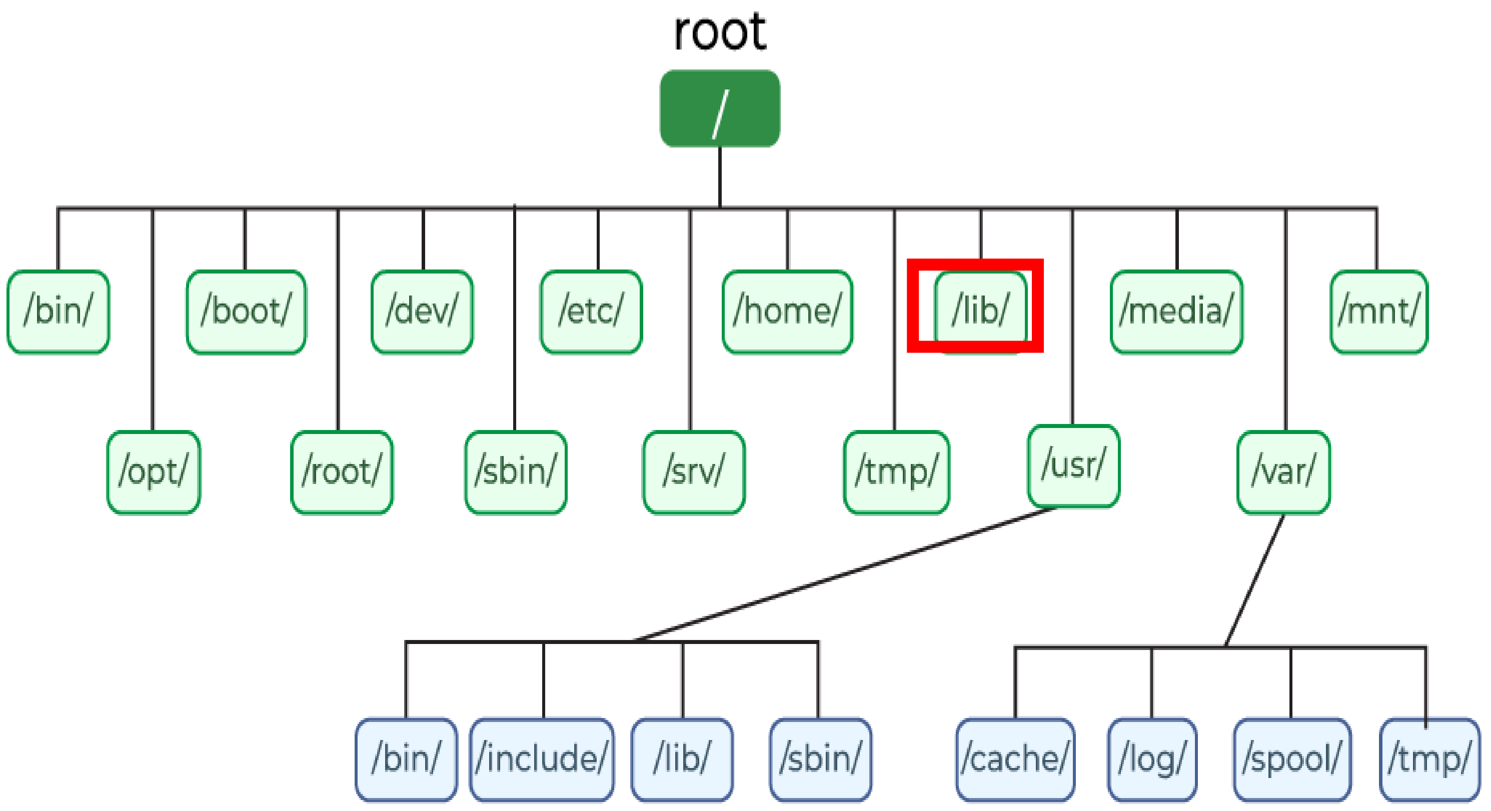
Directory structure

- /home: Contains the home directories for the users.



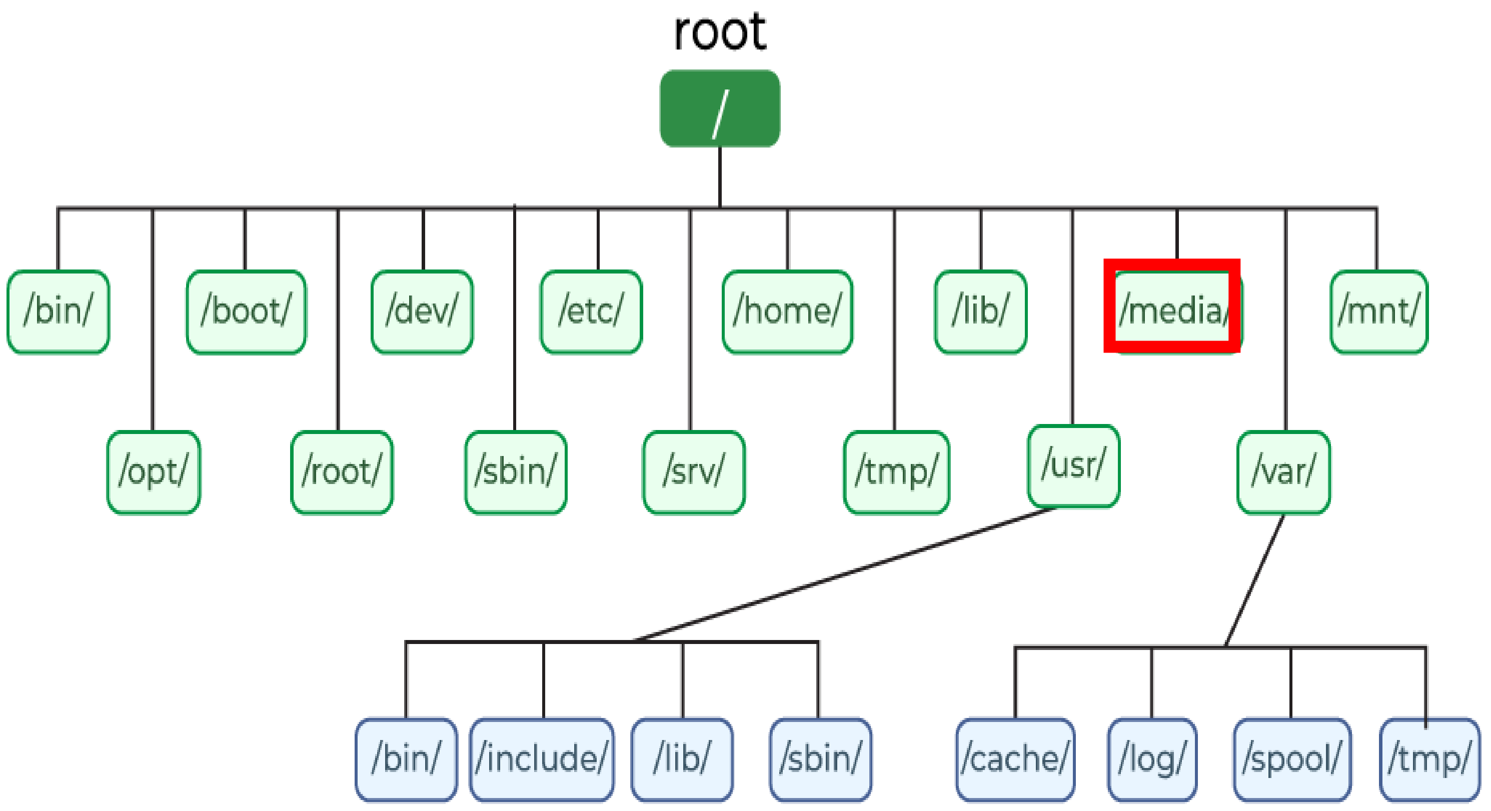
Directory structure

- /lib: Contains system libraries, and some critical files such as kernel modules or device drivers.



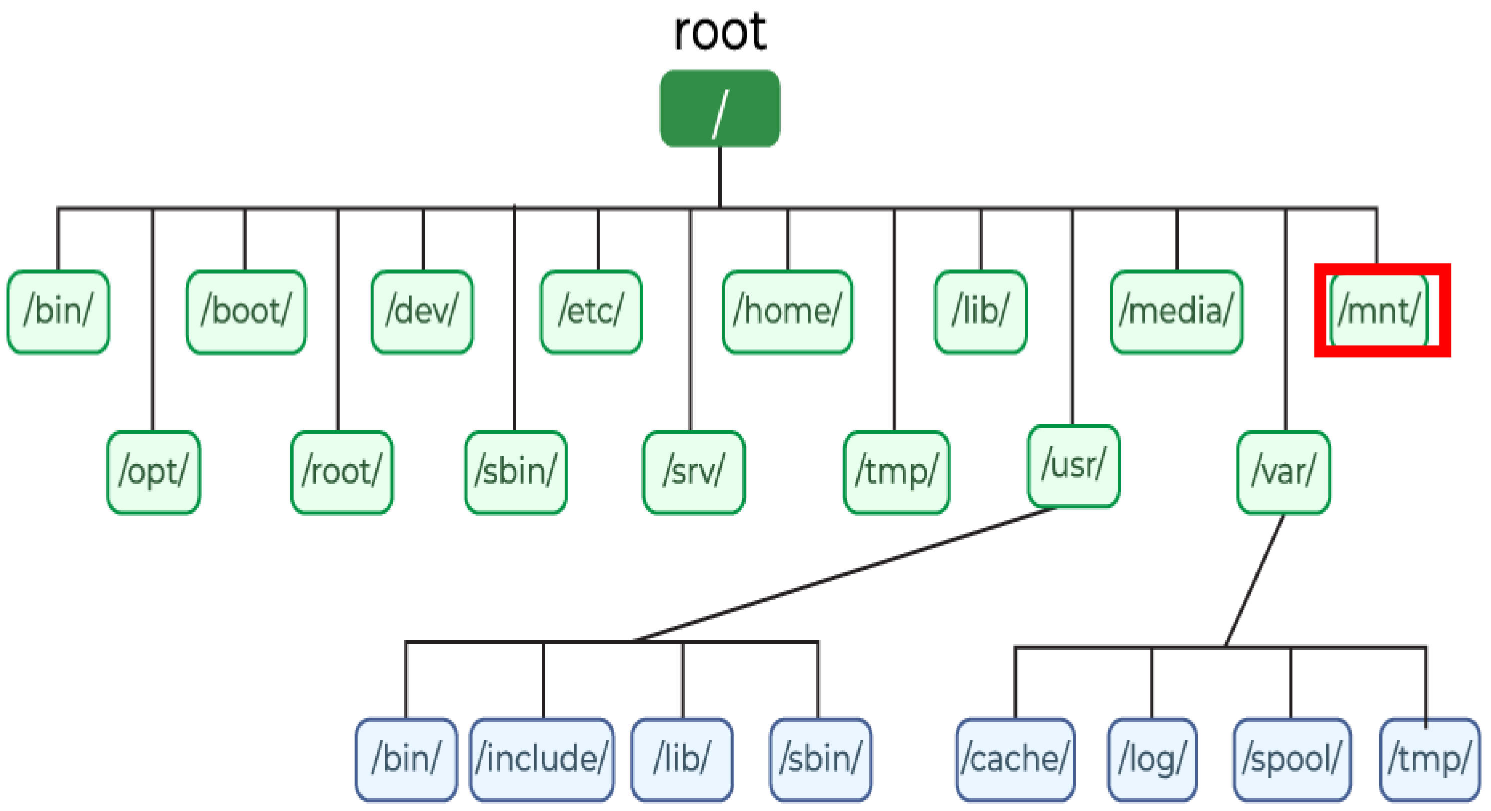
Directory structure

- /media: Default mount point for removable devices, such as USB sticks, media players, etc.



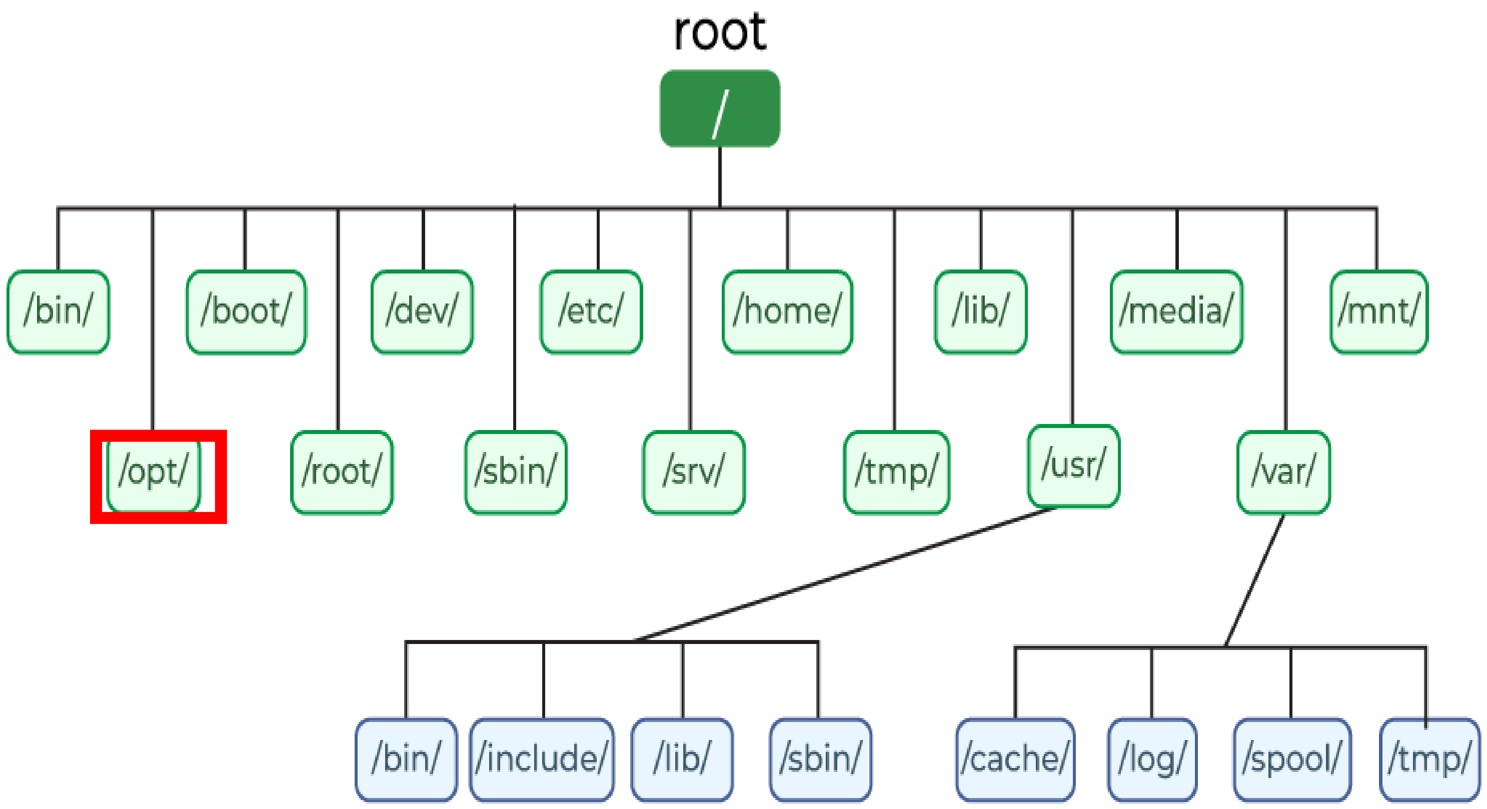
Directory structure

- /mnt: “mount” Contains filesystem mount points. These are used, for example, if the system uses multiple hard disks or hard disk partitions. It is also often used for remote (network) filesystems, CD-ROM/DVD drives, and so on.



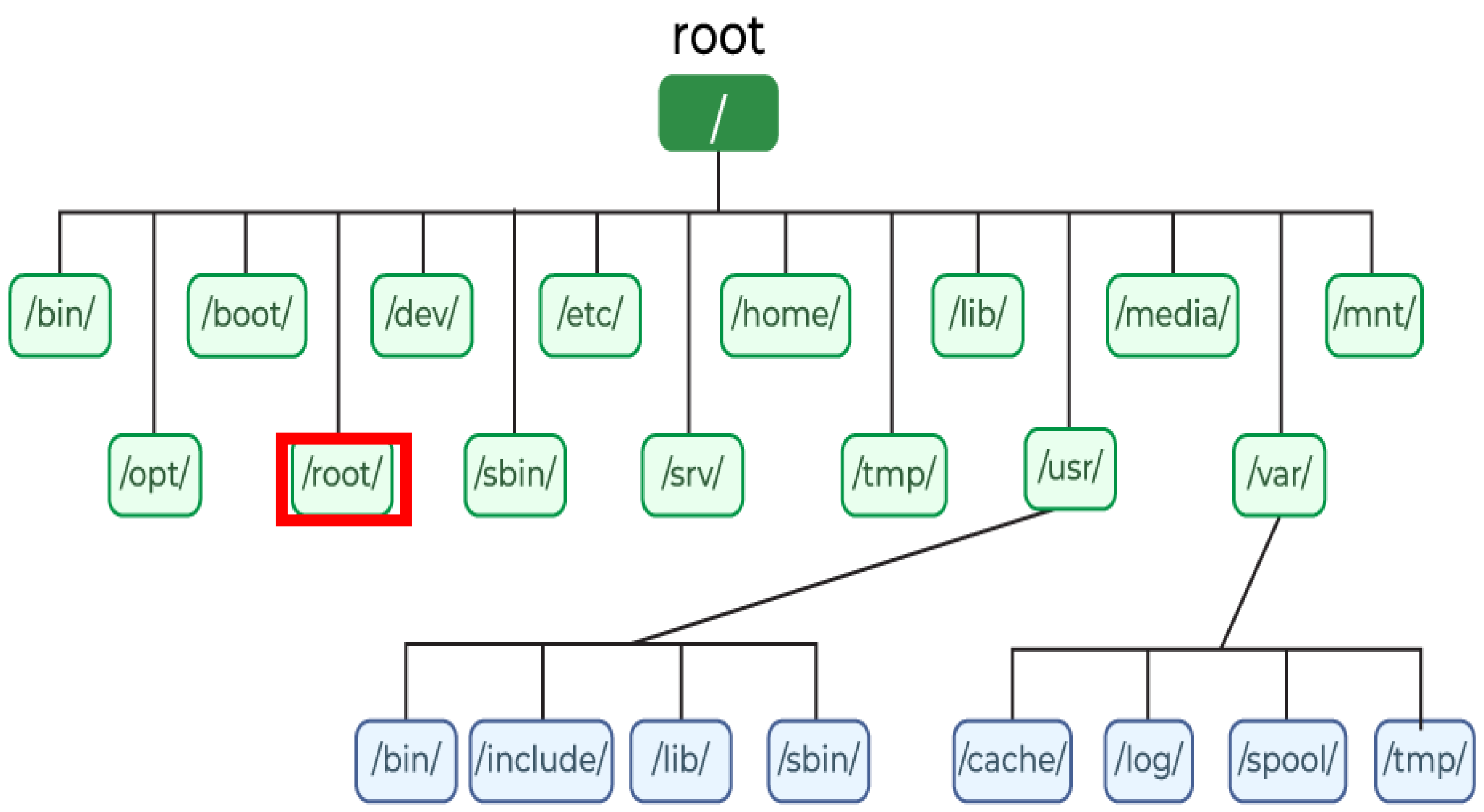
Directory structure

- /opt: optional application software packages that are not part of the standard Linux distribution.



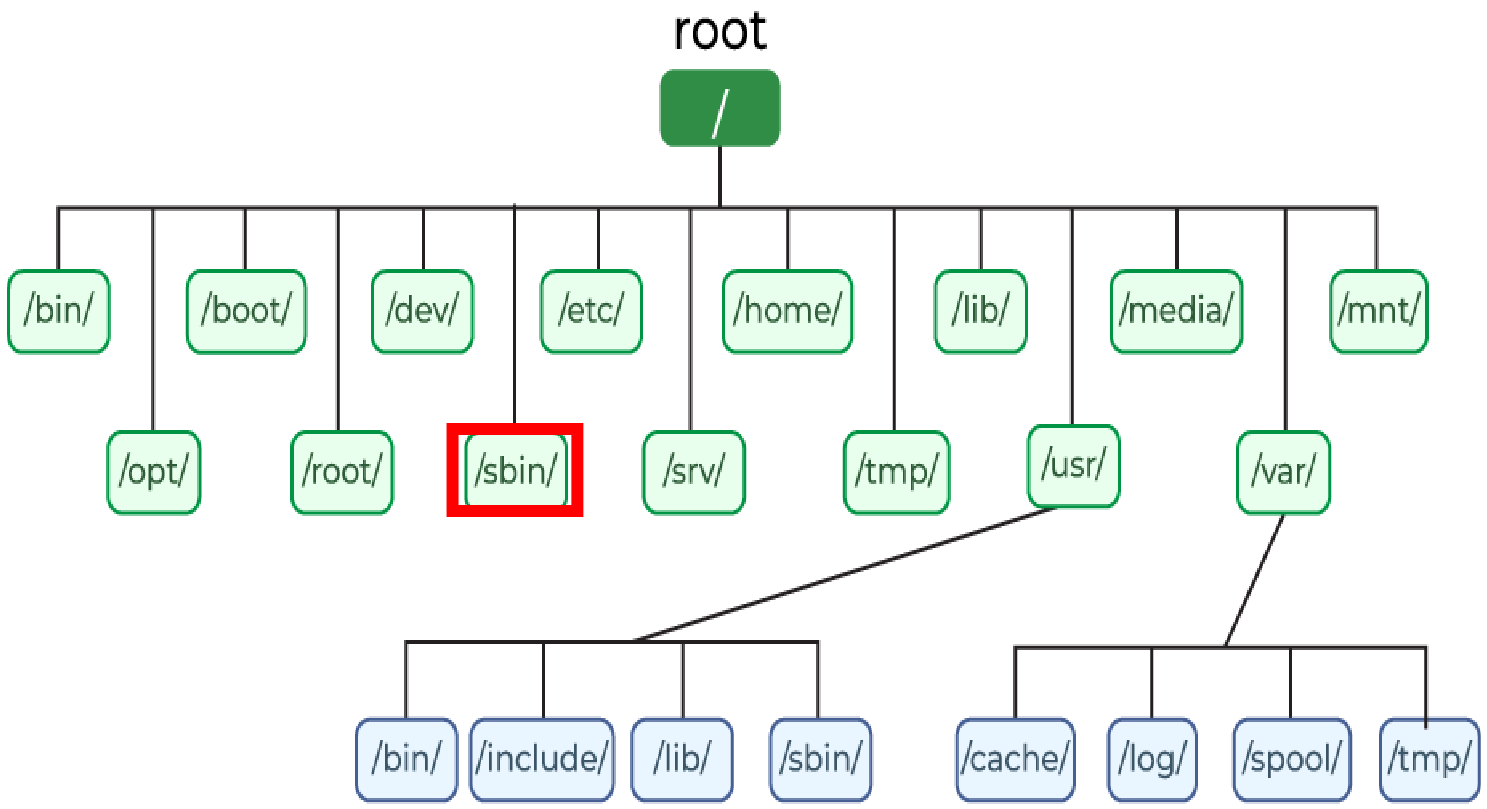
Directory structure

- /root: Home directory for the root user.
- Always available even if /home/ fails



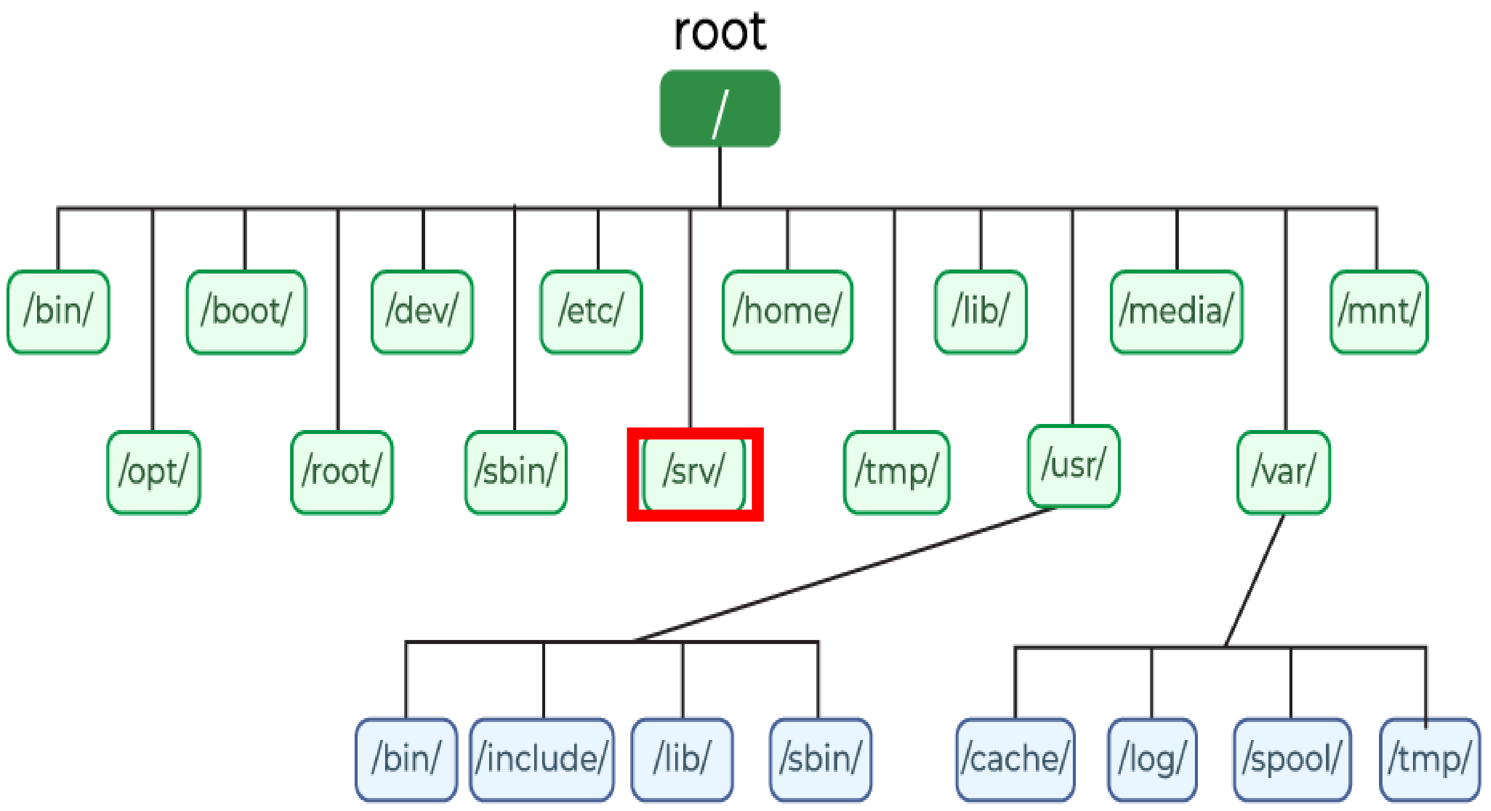
Directory structure

- /sbin: Superuser system utilities (for performing system administration tasks).



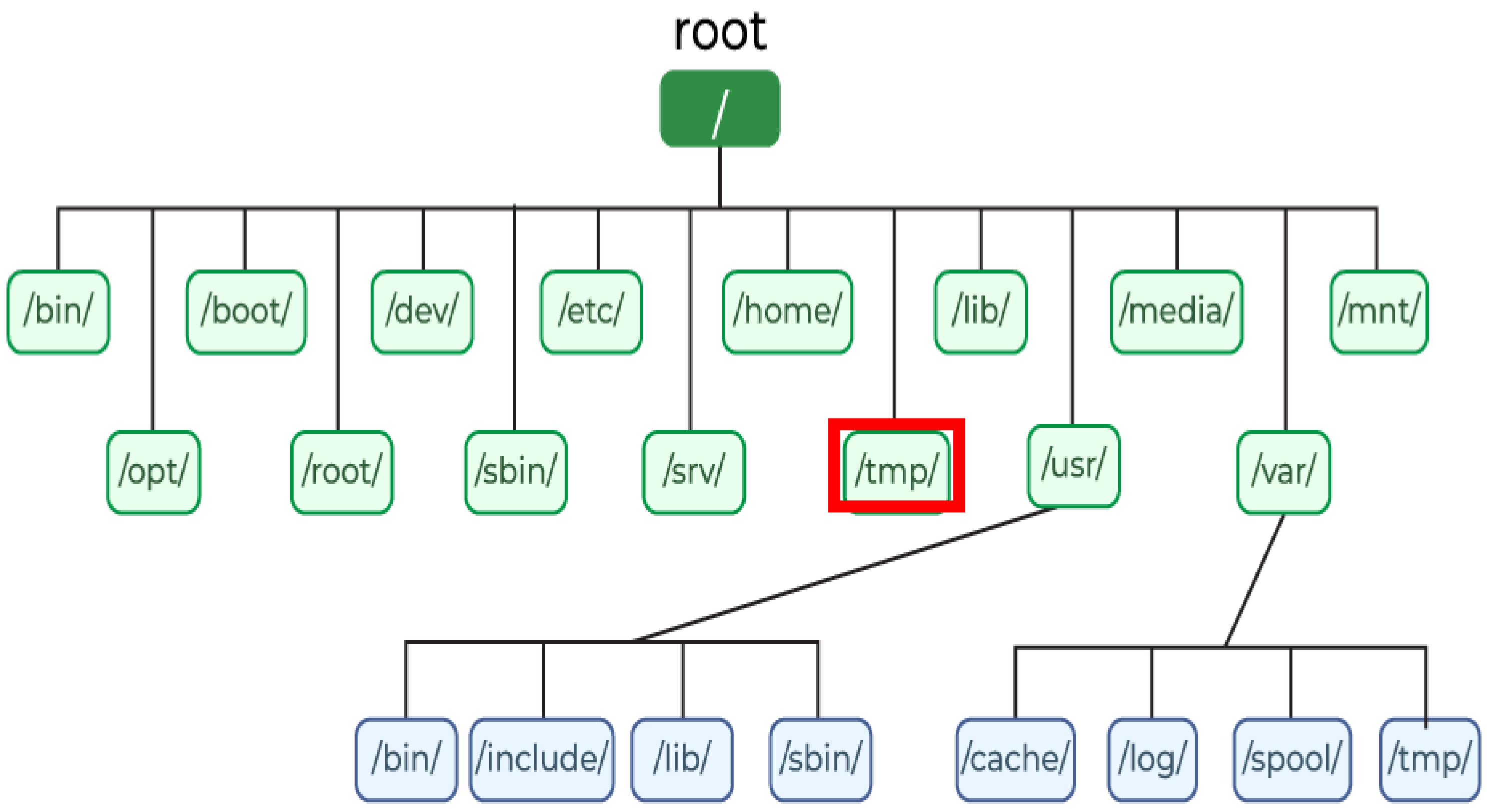
Directory structure

- /srv: “service” Contain files that are served to other systems (webserver, database).



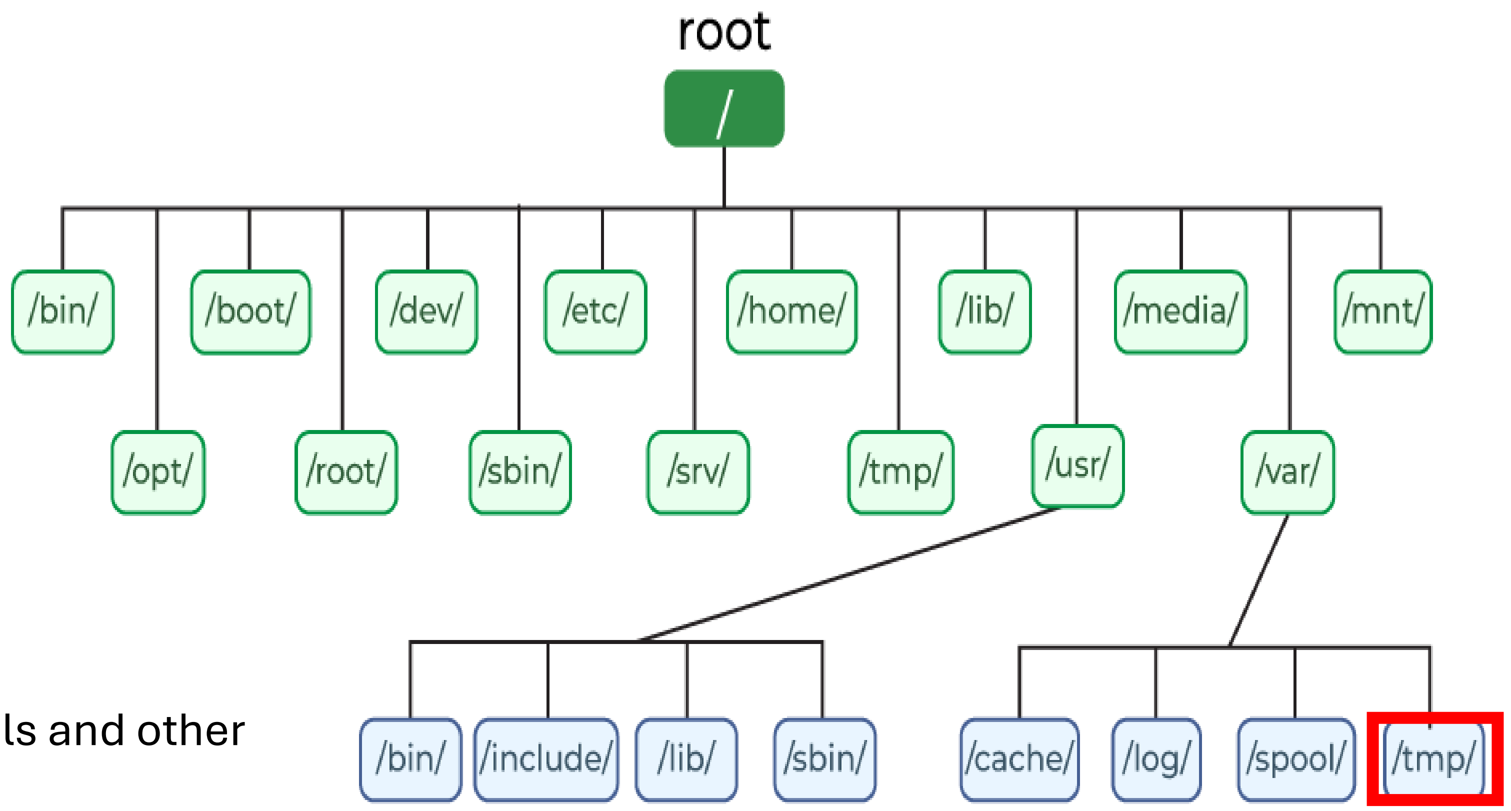
Directory structure

- /tmp: temporary files.



Directory structure

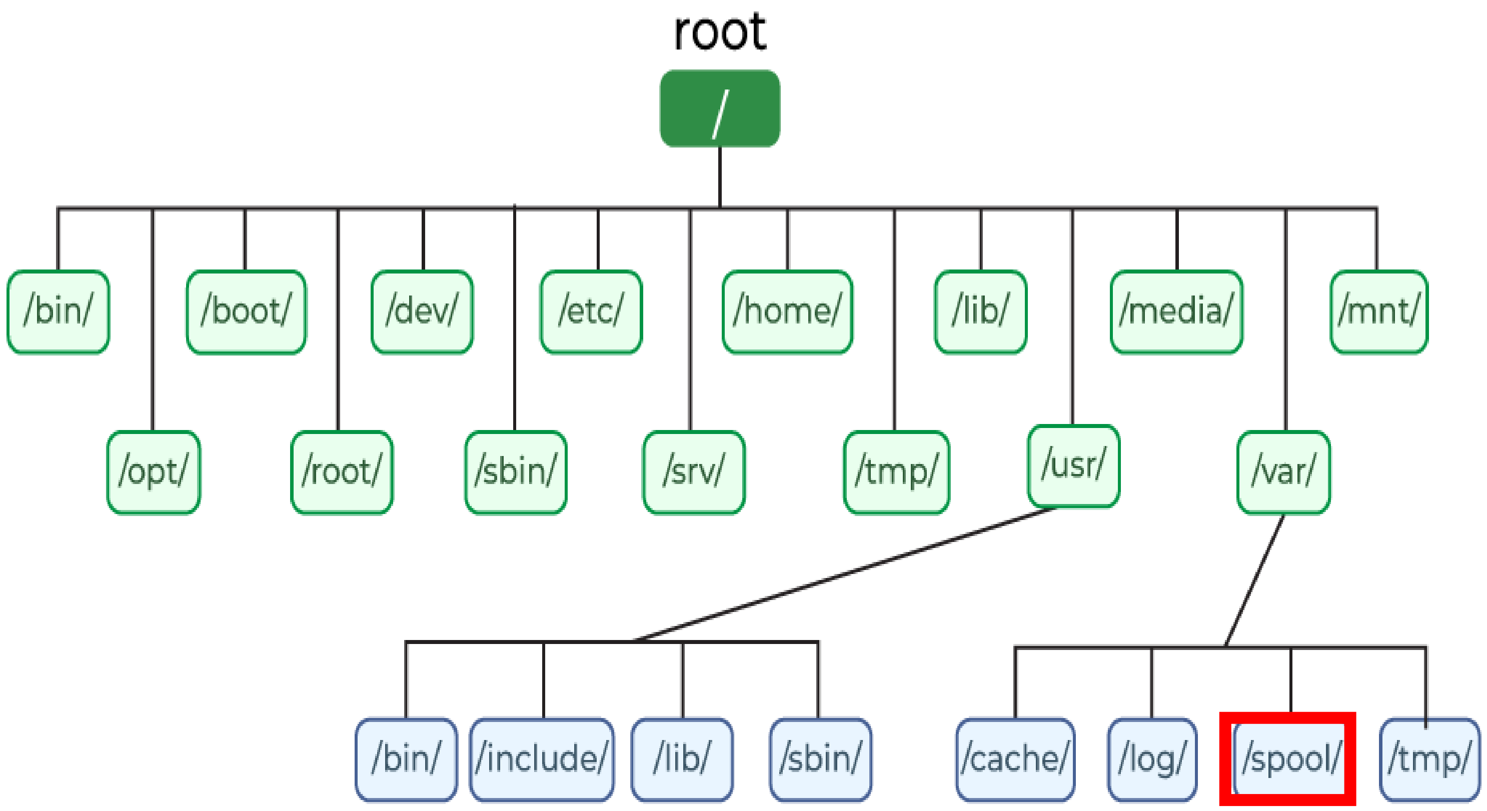
- /tmp/var: Temporary files meant to survive across reboots, but still not permanent.



Spool directory. Contains print jobs, mail spools and other queued tasks.

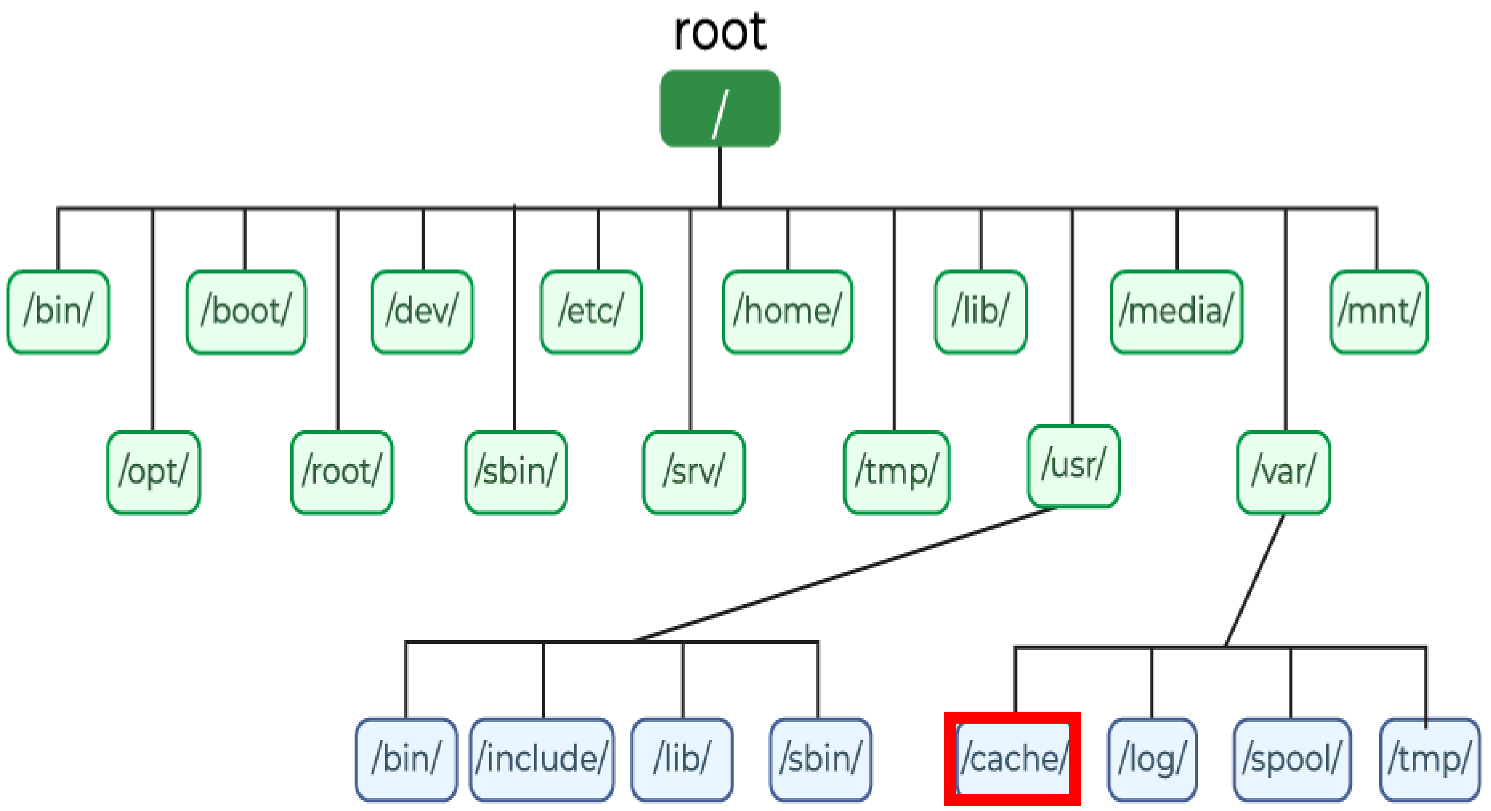
Directory structure

- /tmp/spool: Contains print jobs, mail spool and other queued tasks



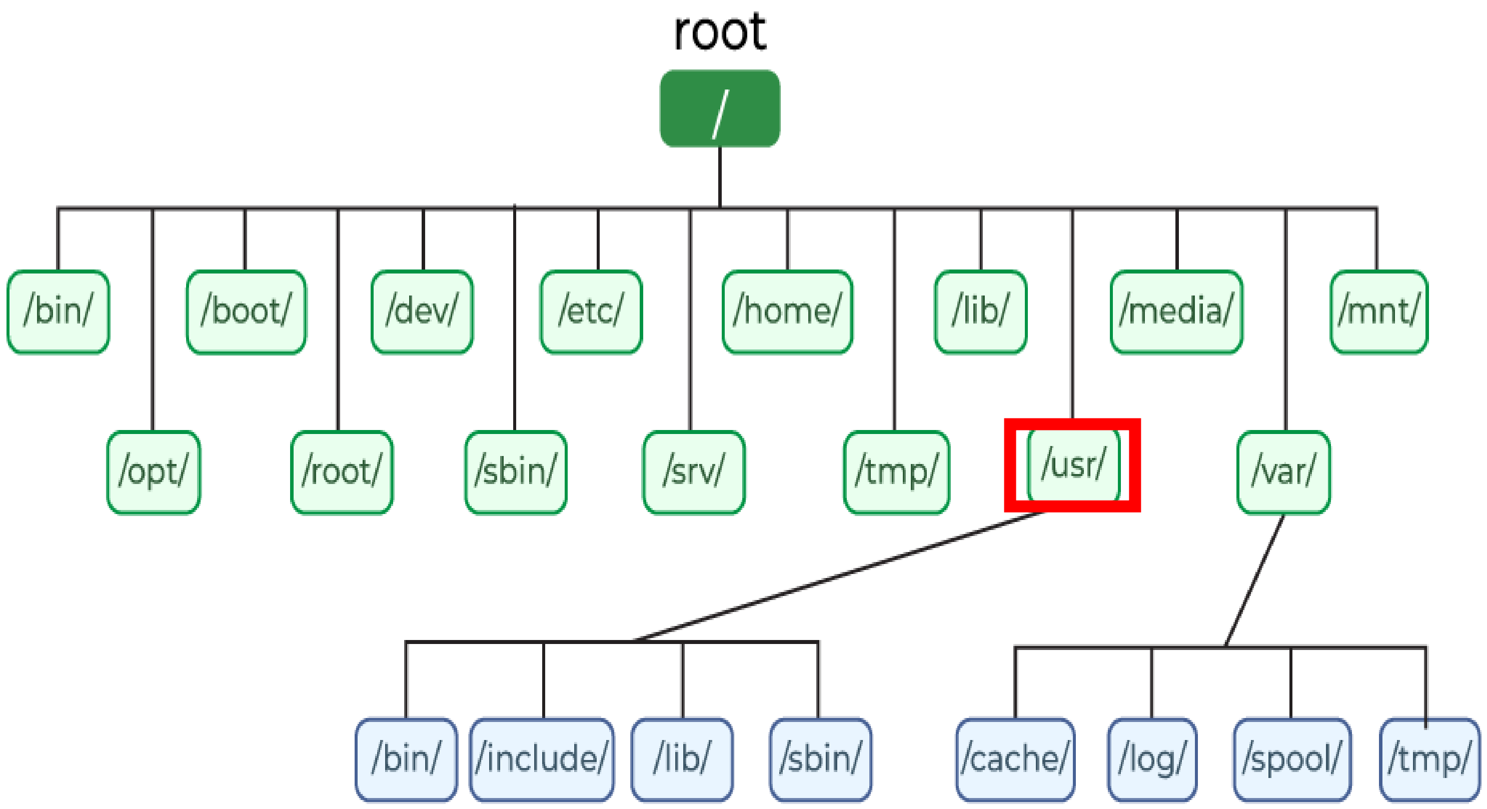
Directory structure

- /tmp/cache/: e.g. web browsers cache website to load faster



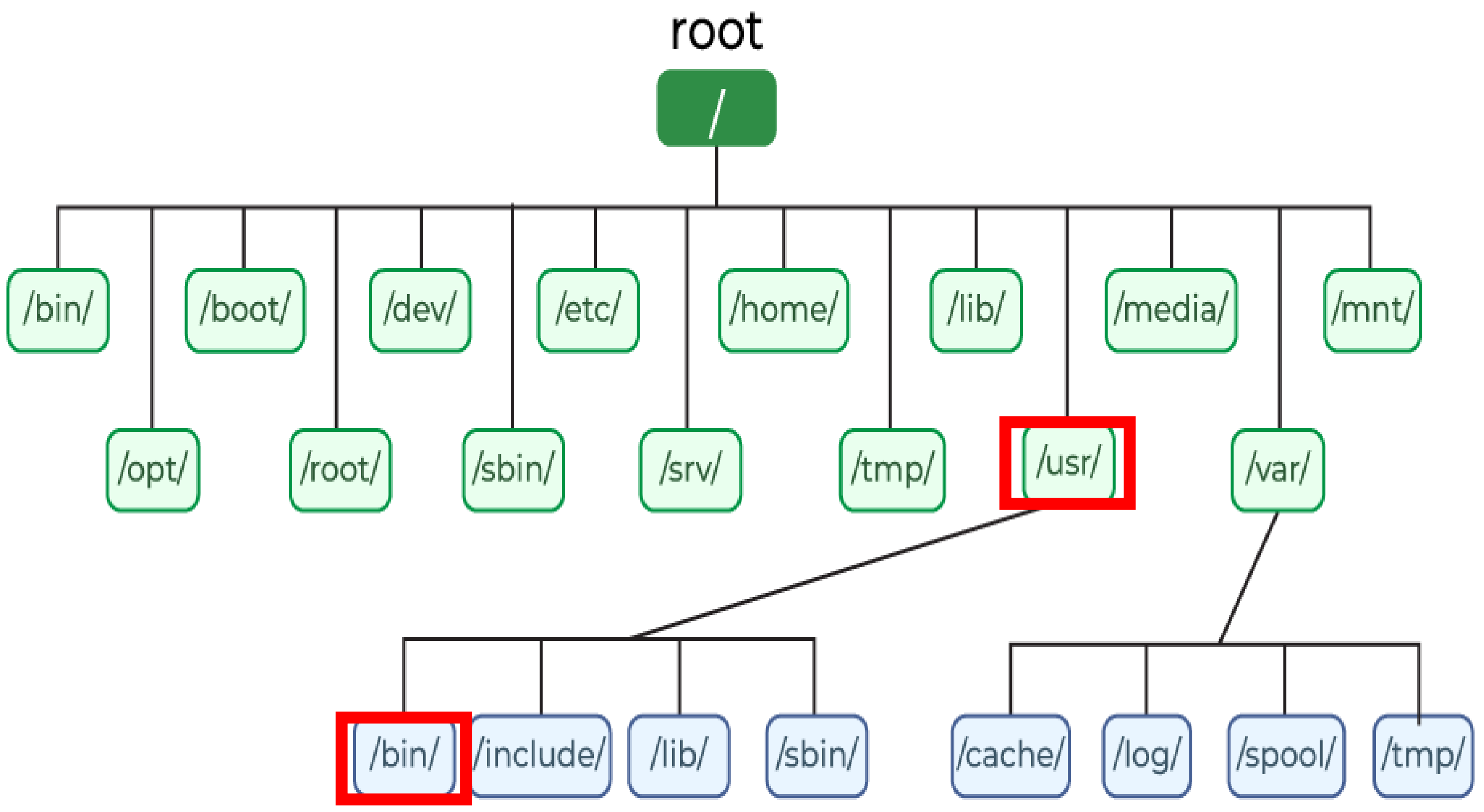
Directory structure

- /usr: applications and files that are mostly available for all users to access.



Directory structure

- /usr/bin: This directory stores all binary programs distributed with the operating system not residing in /bin, /sbin or (rarely) /etc.



Paths

What is a path?

- A path is a unique location to a file or folder in the filesystem.
- Paths use / and alphanumeric characters to describe locations

Paths

What is a path?

- A path is a unique location to a file or folder in the filesystem.
- Paths use / and alphanumeric characters to describe locations

Two types of paths:

- **Absolute Path**: Complete path starting from /.
    - /home/base/myData/test.config
- **Relative Path**: Path relative to your current directory (pwd)
    - myData/

Basic commands

🖥️ **Commands to use:**

- pwd ➔ Show where you are (Print Working Directory)
- ls ➔ List files and directories
- cd ➔ Change directory

Basic commands

🛠 **Quick Tasks:**

•Move to the root directory:

```
$ cd /
```

•List what's inside /home/:

```
$cd /home
ls
```

•Find what’s inside /etc/:

```
cd /etc
ls
```

•Come back to home directory:

```
cd ~
```

Basic commands

🛠️ **More Tasks:**

- Continue to explore the filesystem tree using cd, ls, pwd. Look in /bin, /usr/bin, /sbin, /tmp and /boot. What do you see? What are the permissions on /etc/passwd
- Explore /dev
- Run echo $SHELL, echo $0, and ps -p $$.
- Explore /dev/: Identify your terminal device.
- Use who am i and tty.
- Can you find your USB or disk partitions?
- Inspect /boot/ — what kernel version is installed?
- Open /etc/fstab — what filesystems are auto-mounted?
- Use systemctl list-units to view systemd targets.
- Why cd /.. does not give an error message
- Give 2 different way to go to your home directory

Basic commands

🛠️ **More Tasks (sol):**

- /bin Essential system programs like ls, cp, mkdir, bash
- /usr/bin Most user applications like python3, gcc, vim, nano
- /sbin System administration programs like ifconfig, reboot, fdisk
- /tmp Temporary files (might be empty if recently rebooted)
- /boot Boot files, including vmlinuz (kernel image) and grub configuration
- echo $SHELL → e.g., /bin/bash; echo $0 → current shell.
- ps -p $$ → shows the shell process info.
- tty or who am i → shows current terminal device (e.g., /dev/pts/0).
- Use lsblk or df -h to list block devices and mounted partitions.
- Character devices /dev/tty0, /dev/tty1, /dev/console, /dev/random
- Block devices /dev/sda, /dev/sdb, /dev/loop0 (disks and partitions)
- ls /boot/ → look for files like vmlinuz-5.15.0-XX for kernel version.
- cat /etc/fstab → shows device mount rules, e.g., /dev/sda1 / ext4 defaults 0 1.
- systemctl list-units --type=target → shows active systemd targets.
- Because /.. exists and it points back to / itself.
- cd , cd ~, cd $home