**What is an Operating System (OS)?**

Think of an operating system as a bridge that connects the software (apps) with the hardware (CPU, RAM, disk, etc.). It manages all interactions between the two.

Example: Let’s say you install Jenkins. Jenkins itself cannot directly communicate with the CPU or RAM. Instead, it tells the OS, and the OS handles all requests to the hardware.

**How it Works (Simple Flow):**

• You install an application (e.g., Jenkins).

• The application requests resources (like CPU power or RAM).

• The OS talks to the hardware and gets those resources.

• The OS sends them back to the application.

• The application then functions as expected.

So, the OS is the middleman that makes everything work smoothly.

**Why is Linux So Popular?**

Linux is widely used in DevOps, cloud computing, and server management. Here’s why:

It's Free & Open Source

• Unlike Windows (which you need to buy), Linux is free.

• It’s also open-source, meaning anyone can modify or improve it.

**Security**

• Linux is inherently more secure than Windows.

• You don’t need an antivirus because Linux doesn’t allow unauthorized access easily.

**Many Distributions (Flavors)**

• Linux has different versions called distributions (distros). Some **popular ones include:**

• Ubuntu (most user-friendly, great for beginners)

• Red Hat (RHEL) (popular in enterprises)

• CentOS (was popular in servers, now replaced by AlmaLinux/Rocky Linux)

• Debian, Alpine, Fedora, Arch Linux (for different purposes)

**Speed & Stability**

• Linux is fast and doesn’t slow down over time, unlike Windows.

• It's stable, meaning it doesn’t crash often—perfect for running servers and applications.

Used in Software Development

• Almost every application in DevOps runs on Linux.

• It’s used in production, testing, and development environments.

**Basic Architecture of Linux**

Linux has different components that work together:

**Kernel (The Heart of Linux)**

• The kernel is the core of Linux.

• It directly communicates with hardware (CPU, memory, devices).

**It manages:**

• Device management (handles hardware like USB, keyboard, etc.)

• Memory management (allocates RAM)

• Process management (controls running apps)

* + System calls (allows apps to talk to the kernel)

**System Libraries**

* These are pre-written functions that help applications communicate with the kernel.
* Example: libc (C library) helps run common tasks like file handling.

**User Space**

* This is where users interact with the system.
* It includes:
* Shell (Command Line Interface)
* Compilers (to run code)
* User processes (applications like Jenkins, MySQL, Apache, etc.)

**What is Shell Scripting?**

Shell scripting is a way to write a sequence of commands in a file to automate tasks, such as backups or log analysis. It allows you to save time and reduce human errors in repetitive tasks

**Example:** In Windows, you click on icons to create a file. In Linux, you type a command like:

**touch myfile.txt**  # Creates an empty file

Since servers don’t have a graphical interface, shell commands are the only way to interact with them.

**Role of Shell Scripting in DevOps**

Why do DevOps engineers need shell scripting?

* Infrastructure Maintenance – Automating server setup.
* Code Management – Interacting with Git repositories.
* Configuration Management – Managing system configurations.
* Task Automation – Scheduling tasks with cron jobs.
* System Monitoring – Checking CPU/memory usage.

**Basic Shell Commands You Must Know**

**Listing Files & Directories**

ls → Shows files & folders in the current directory.

ls -ltr → Shows files with details (permissions, owner, size, date modified).

**Navigating Directories**

pwd → Shows your current location in the file system.

cd <directory\_name> → Move inside a directory.

cd .. → Move one step back.

cd ../../ → Move two steps back.

**Creating & Deleting Files**

touch myfile.txt → Creates an empty file.

mkdir myfolder → Creates a folder.

rm myfile.txt → Deletes a file.

rm -r myfolder → Deletes a folder and everything inside it.

**Viewing & Editing Files**

cat myfile.txt → Displays the file’s content.

vi myfile.txt → Opens the file in Vi editor (a text editor).

Press i → Enter insert mode to type.

Press Esc → Exit insert mode.

Type :wq → Save and exit.

**System Monitoring Commands**

free -m → Shows memory usage in MB.

nproc → Shows the number of CPUs.

df -h → Shows disk space usage.

top → Displays real-time CPU & memory usage.

**Important Interview Question:**

"Which command is used to monitor CPU & memory usage?" Answer: top

**The Shebang (#!/bin/bash) – What Is It?**

The first line in a shell script should always be: **#!/bin/bash**

Why? This tells the system which shell should be used to execute the script.

Different shells:

* /bin/sh (Basic shell)
* /bin/bash (More powerful, commonly used)
* /bin/ksh (Korn shell)

/bin/dash (Used in modern Ubuntu)

**What is Hardware?**

Hardware is the physical parts of a computer — things you can touch.

Examples of Hardware:

CPU (Processor) – the brain of the computer

RAM (Memory) – temporarily stores data for fast access

Hard Drive / SSD – stores files and software

Motherboard – connects all the components

Monitor, Keyboard, Mouse, Printer – input/output devices

Network Card, GPU, USB Ports – other components

**What is Software?**

Software is the set of instructions (code/programs) that tell hardware what to do. You can't touch it, but you can run it and interact with it.

**Types of Software:**

**1. System Software:** Controls the hardware and provides a platform for other software.

Example: Operating System like Linux, Windows, macOS

**2. Application Software:** Used by users to do specific tasks.

Examples: MS Word, Chrome, VLC Player, Photoshop

**3. Programming Software:** Tools to write and develop programs.

Examples: Code editors, compilers (like GCC, Java), IDEs like VS Code