**What is Linux Architecture?**

Think of **Linux** like a **big building** (a house or a school).

* The **foundation** = **Hardware** (your computer parts).
* The **middle floors** = **Kernel & System software** (managers).
* The **top floors** = **Applications you use** (games, browsers, editors).

It is built in **layers**, where each layer talks to the one below it.

**Layers of Linux Architecture**

**1. Hardware (Bottom Layer 🖥️)**

* This is your **computer parts**:  
  CPU, RAM, Hard Disk, Keyboard, Mouse, Monitor, etc.
* Hardware can’t understand human language — it only understands **electrical signals (0s and 1s)**.

**2. Kernel (The Boss 👑)**

* The **Kernel** is the **heart of Linux**.
* It is like the **principal of a school** — it controls everything.
* It talks to **hardware** and tells it what to do.
* You never talk to the kernel directly (like you don’t directly talk to the CPU).

Kernel’s main jobs:

1. **Memory Management** – Who can use how much RAM.
2. **Process Management** – Which program runs, and when.
3. **Device Management** – Talking to devices like keyboard, printer.
4. **File Management** – Keeping your files safe.

**3. System Libraries (Helpers 📚)**

* Libraries are **pre-written code** that help programs talk to the kernel.
* Imagine them like **dictionaries** — they translate your request into a language the kernel understands.
* Example: If you write a program that wants to read a file, the library will say to the kernel: “Hey, open this file for him!

**4. System Utilities (Tools 🛠️)**

* These are **basic programs** that help you run your system.
* Example: ls (list files), cp (copy files), mkdir (make folder).
* Think of them like **classroom tools**: chalk, duster, notebook.

**5. User Space (Top Floor 👩‍💻👨‍💻)**

This is where **you** live!

* Applications (like Chrome, VS Code, Games).
* Shell (where you type commands, e.g., Bash).
* Desktop Environment (like GNOME, KDE — your graphical desktop).

The **user** talks to **shell/app**, shell talks to **libraries**, libraries talk to **kernel**, kernel talks to **hardware**.

**Diagram of Linux Architecture**

Here’s a simple diagram:

+-----------------------------+

| User Applications | 👩‍💻 (Games, Browsers, Editors)

+-----------------------------+

| Shell / System Utilities | 🛠️ (ls, cp, mkdir)

+-----------------------------+

| System Libraries | 📚 (glibc, APIs)

+-----------------------------+

| Kernel | 👑 (Memory, Process, Devices, Files)

+-----------------------------+

| Hardware | 🖥️ (CPU, RAM, Disk, Keyboard, Mouse)

+-----------------------------+

**Example: How it all works**

Let’s say you type:

cat hello.txt

(You want to read a file called hello.txt)

1. **You (User)** → type command in Shell.
2. **Shell** → asks System Library to open a file.
3. **Library** → sends request to Kernel.
4. **Kernel** → asks Hardware (Disk) to read the file.
5. **Disk (Hardware)** → sends data back.
6. **Kernel** → sends data to Shell.
7. **Shell** → shows the file on your screen.