Module: 4- Linux server - Manage user and Groups and working with file systems

1. **Manage users and groups with commands like useradd, userdel, groupadd, and passwd**

👤 User Management

➕ Add a User

```
sudo useradd username
```

- Creates a new user.
- Use `-m` to create the home directory:

```
sudo useradd -m username
```

🔐 Set Password for a User

```
sudo passwd username
```

- Prompts you to enter a new password.

🗑️ Delete a User

```
sudo userdel username
```

- Deletes the user (but not the home directory).
- To remove the home directory as well:

```
sudo userdel -r username
```

🔄 Modify a User (e.g., change username or home dir)

```
sudo usermod -l newname oldname          # change username
sudo usermod -d /new/home/dir username # change home directory
```

👥 Group Management

➕ Add a Group

```
sudo groupadd groupname
```

➖ Delete a Group

```
sudo groupdel groupname
```

➕ Add User to a Group

➔ `sudo usermod -aG groupname username`

- The `-aG` means "append to group" – important to avoid overwriting existing groups.

👀 **Check Group Membership**

➔ `groups username`

✅ **Example: Create a New User and Assign to a Group**

```
➔ sudo useradd -m alice
➔ sudo passwd alice
➔ sudo groupadd developers
➔ sudo usermod -aG developers alice
```

2. Explain different file system types in Linux?

➔ 🗂️ What is a File System?

A **file system** is like the rules or method that Linux uses to **organize and store files** on a disk (like our hard drive or USB stick). Think of it like a library system that knows where every book (file) is placed and how to access it.

🧱 Common Linux File System Types

1. ext2 (Second Extended File System)

- 📅 One of the oldest.
- ❌ No journaling (more on that in a second).
- 🐢 Slower recovery after a crash.
- ✅ Still used in USB drives or SD cards because it's light.

2. ext3 (Third Extended File System)

- ✅ Adds **journaling** (keeps a log of file changes to recover quickly after a crash).
- 🔁 Backward-compatible with ext2.
- 📉 Slower than modern file systems, but stable and reliable.

3. ext4 (Fourth Extended File System)

- 🚀 Faster and more efficient than ext2/ext3.
- ✅ Supports large files and drives.
- 🛡️ Has journaling, quicker recovery, and less chance of corruption.
- 📦 Most popular Linux file system today.

4. XFS

- 🏋️ Built for **big files and high-performance**.
- 🧠 Uses journaling.
- 📁 Great for servers or databases with lots of data.
- ❌ Not as flexible for small files or home use.

5. Btrfs (Butter File System or "B-tree FS")

- 🧪 Still under heavy development but very powerful.
- 🧩 Has **advanced features** like:

    - Snapshots (save states of your system).
    - Compression.
    - Built-in RAID (combining disks).
- 🛠️ Ideal for advanced users or servers.

6. FAT32

- 📦 Simple and widely supported (Windows, Linux, Mac).
- ⚠️ Limited: can't store files bigger than 4GB.
- 🎒 Common in USB drives and SD cards.

7. exFAT

- ✨ Improved version of FAT32.
- ✅ Handles big files.
- 💻 Good for USB drives and external disks shared between Windows and Linux.

8. NTFS

- 🏠 Native to Windows.
- ✍️ Linux can read/write to it with some extra support.
- 📚 Useful when sharing drives between Linux and Windows.

3. Explain File Permission groups in Linux?

➔ 🛡️ What Are File Permission Groups?

In Linux, **every file and folder** has **permissions** that control **who can do what** with it. These permissions are divided into **three groups**:

1. User (u) – 👤 The owner of the file

- Usually the person who created the file.
- Has personal access to read, write, or execute the file.

2. Group (g) – 👥 A group of users

- People who are part of the file's group.
- They share some permissions, set by the owner or system.

3. Others (o) – 🌍 Everyone else

- All other users on the system.
- These are general permissions for anyone **not** the owner or in the group.

🔐 Types of Permissions

Each group can have **3 types of permissions**:

Symbol	Name	What it allows
r	Read	View the file or list the folder contents

w	Write	Edit or delete the file/folder
x	Execute	Run the file (if it’s a program or script) or enter the folder

---

📄 Example of File Permissions

Let’s look at this example from the command:

```
ls -l file.txt
```

```
-rw-r--r--
```

Breakdown:

- - → it's a file (if it was d it would be a directory)
- rw- → **User** (owner) can read & write
- r-- → **Group** can only read
- r-- → **Others** can only read

So, in words:

> The owner can read and edit the file. Everyone else (group and others) can only read it.

🛠️ Changing Permissions

You can change permissions using the chmod command.

Example:

chmod u+x script.sh

→ Adds **execute** permission for the **user** (owner) of the file.

4. **How do you switch from one desktop environment to another, such as switching from KDE to Gnome?**

➔ 🖥️ What's a Desktop Environment?

A **desktop environment (DE)** is what gives us the look and feel of our Linux system — like the menu, taskbar, file manager, and system settings.
Examples: **KDE**, **GNOME**, **XFCE**, etc.

🔄 How to Switch from KDE to GNOME (Step-by-Step)

✅ Step 1: Install GNOME (if not already installed)

First, we need to install the GNOME desktop if it's not already on our system.

For Ubuntu/Debian-based systems:

➔ sudo apt update
➔ sudo apt install gnome-shell

For Fedora:

➔ sudo dnf install @gnome-desktop

For Arch:

➔ sudo pacman -S gnome gdm

✅ Step 2: Change the Display Manager (optional but recommended)

The **display manager** is the login screen. KDE uses **SDDM**, and GNOME uses **GDM**.

To switch to GNOME's display manager (**GDM**):

```
➔ sudo dpkg-reconfigure gdm3  # On Ubuntu/Debian
```

Then choose gdm3 when asked.

```
➔ sudo systemctl enable gdm --force
➔ sudo systemctl disable sddm
```

Then restart:

```
➔ sudo reboot
```

✅ Step 3: Choose GNOME at Login Screen

After reboot:

1. At the login screen, click on our username.
2. Look for a **gear icon** ⚙️ or **session option** (usually at the bottom corner).
3. Click it and select **GNOME** or **GNOME on Xorg**.
4. Enter our password and log in.

Boom! we're now in **GNOME**! 🎉

🧹 Optional: Remove KDE (if we want to free space)

If we don't want KDE anymore:

```
➔ sudo apt remove --purge kde-plasma-desktop
```

➔ sudo apt autoremove

(Be careful! This removes packages – make sure we don’t delete something important.)

5.What are the kinds of permissions under Linux?

➔ 🛡️ **What Are Permissions in Linux?**

Permissions in Linux decide **who can do what** to a file or folder.
There are **three kinds of permissions** we need to know:

🔑 1. Read (r)

- 📖 Lets we **see or read** the file’s contents.
- 📁 For folders, it means we can **see what’s inside** (list files).

Example: we can **open and read** a text file, but not change it.

✏️ 2. Write (w)

- ✍️ Lets we **edit, change, or delete** the file.
- 📁 For folders, it lets we **add, remove, or rename files** inside it.

Example: we can **edit a file**, or **add new files** to a folder.

▶️ 3. Execute (x)

- 🚀 Lets we **run the file** as a program or script.
- 📁 For folders, it means we can **enter the folder** (with cd).

Example: we can run a script like ./myscript.sh if it has execute permission.

👥 Who Can Have These Permissions?

Each file/folder gives these permissions to **three groups**:

Group	Who it is
User	The file’s **owner**
Group	A group of users linked to the file
Others	**Everyone else** on the system

🔍 Example Breakdown

Let’s say we run this command:

```
ls -l myfile.txt
```

```
-rwxr-xr--
```

Breakdown:

Group	Permissions	Meaning
User	rwx	Can read, write, and run it
Group	r-x	Can read and run it
Others	r--	Can only read it

6. **What are the different modes when using vi editor?**

➔ 📝 **What is vi?**

vi is a **text editor** used in the Linux terminal. It's fast and powerful, but it works differently from other editors because it has **modes**.

🔁 **vi Has 3 Main Modes:**

1. Normal Mode (also called Command Mode)

- 🚦 This is the **default mode** when we open vi.
- ⌨️ we can **move around**, **copy/paste**, **delete**, or **switch modes**.
- ❌ we **can't type text** here directly.

Examples of what we can do in Normal Mode:

- dd → delete a line
- yy → copy a line
- p → paste
- u → undo

👉 Press Esc to make sure we're in Normal Mode.

---

2. Insert Mode

- ✍️ This is where we can **type text normally**, like in a regular editor.
- we enter Insert Mode from Normal Mode.

How to enter Insert Mode:

- Press i → insert **before** the cursor
- Press a → insert **after** the cursor
- Press o → open a **new line below**

👉 To go **back to Normal Mode**, press Esc.

---

3. Command-Line Mode (also called Last Line Mode)

- 💬 This mode lets we **run commands** like saving or quitting.

- Starts with a **colon :** at the bottom of the screen.

How to enter Command-Line Mode:

- From Normal Mode, press :

we'll see something like this at the bottom:

Examples of commands:

- :w → save the file (write)
- :q → quit
- :wq → save and quit
- :q! → quit **without saving**