1- Create a new group iot_team and add your user to it.

```
ghannam@Ghannam:~$ sudo groupadd iot_team
[sudo] password for ghannam:
ghannam@Ghannam:~$ sudo usermod -aG iot_team $USER
ghannam@Ghannam:~$ groups $USER
ghannam : ghannam adm cdrom sudo dip plugdev users lpadmin iot_team
ghannam@Ghannam:~$
```

2- Create a new developer user, add it to the group.

```
ghannam@Ghannam:~$ sudo useradd developer
ghannam@Ghannam:~$ sudo usermod -aG iot_team developer
ghannam@Ghannam:~$ groups developer
developer : developer iot_team
ghannam@Ghannam:~$
```

3- Change ownership of iot_logger to the developer + group.

```
ghannam@Ghannam:~$ sudo chown developer:iot_team iot_logger
```

4- Set permissions: group can read/write logs, others blocked.

```
ghannam@Ghannam:~$ sudo chmod 660 iot_logger
ghannam@Ghannam:~$ ls -l
total 64
drwxrwxr-x 2 ghannam   ghannam  4096 Aug 29 19:16 client-server-mqtt
drwxr-xr-x 2 ghannam   ghannam  4096 Jul  3 15:16 Desktop
drwxr-xr-x 2 ghannam   ghannam  4096 Jul  3 15:16 Documents
drwxr-xr-x 2 ghannam   ghannam  4096 Jul  6 22:12 Downloads
drwxr-xr-x 1 ghannam   ghannam    40 Sep  1 21:43 host_share
drw-rw---- 5 developer iot_team 4096 Sep  1 20:50 iot_logger
drwxr-xr-x 6 ghannam   ghannam  4096 Jul  6 21:10 mqttfx
drwxrwxr-x 3 ghannam   ghannam  4096 Jul  6 21:11 MQTT-FX
drwxr-xr-x 2 ghannam   ghannam  4096 Jul  3 15:16 Music
drwxr-xr-x 3 ghannam   ghannam  4096 Aug 27 17:25 Pictures
drwxr-xr-x 2 ghannam   ghannam  4096 Jul  3 15:16 Public
drwxrwxr-x 6 ghannam   ghannam  4096 Jul  6 08:12 ros2_ws
drwxrwxr-x 2 ghannam   ghannam  4096 Aug 31 21:51 scripts,
drwx------ 6 ghannam   ghannam  4096 Aug 29 18:43 snap
drwxr-xr-x 2 ghannam   ghannam  4096 Aug 26 19:44 Templates
drwxrwxr-x 3 ghannam   ghannam  4096 Aug 29 19:18 venvs
drwxr-xr-x 2 ghannam   ghannam  4096 Jul  3 15:16 Videos
ghannam@Ghannam:~$
```

5- Test access as new user, then remove test user.

```
ghannam@Ghannam:~$ whoami
ghannam
ghannam@Ghannam:~$ ls iot_logger
ls: cannot open directory 'iot_logger': Permission denied
```

6- permissions on files VS Directories

1. Permissions on files
r (read): you can view the file contents.
w (write): you can edit/modify/delete the file contents.
x (execute): you can run the file as a program.

ls -l script.sh
-rwxr-xr-- 1 user user 1234 Aug 31 20:05 script.sh
Owner can run, edit, or read.
Group can read and run only.
Others can only read.

2. Permissions on directories
r (read): you can list the names of files inside (ls).
w (write): you can create/delete/rename files inside.
x (execute): you can enter the directory (cd dir/) and access files if you also have read permission on them.

ls -ld dir
drwxr-x--x 2 user user 4096 Aug 31 20:10 mydir

Owner: full control.
Group: can see contents and cd into it.
Others: can't ls mydir, but if they know mydir/filename, they can access it (depending on that file's permissions).

1- Octal notation for permissions

Linux file permissions (r, w, x) can be represented in octal numbers:
r (read) = 4
w (write) = 2
x (execute) = 1
add them up for each class (user, group, others).
So permissions like -rwxr-xr-- =
User = 7 (rwx)
Group = 5 (r-x)
Others = 4 (r--)
→ Written as 754

2- umask

The umask (user file-creation mask) defines which permission bits should be removed when a file or directory is created.
New file:
Default: 666
umask: 022 → subtract ---w--w-
Result: 644 → rw-r--r--

1. Root user vs Normal user
"Root user (superuser)"
Has UID 0.
Full administrative privileges: can read, write, and execute any file in the system.
Can install/remove software, modify system configurations, kill any process, change ownership/permissions, cre
No restrictions enforced by the OS.
"Normal user"
Has a unique UID > 0.
Restricted to their home directory (/home/username) by default.
Can only read/write/execute files they own or have been granted permission for.
Cannot perform administrative tasks unless elevated with sudo.

2. Why is root dangerous?
No safety net → root can delete system files (rm -rf /), crash the OS, or lock out users.
Bypasses permissions → root can read/write sensitive files (like /etc/shadow for passwords).
Malware risk → if malicious commands/scripts run as root, they can take full control.
Human mistakes → a single typo as root can destroy data or disable the system.