**DOCKER VOLUMES**

**When we create a Container then Volume will be created. Volume is simply a directory inside our container.**

**First, we have to declare the directory Volume and then share Volume. Even if we stop the container still, we can access the volume.**

**You can declare directory as a volume only while creating container. We can’t create volume from existing container.**

**You can share one volume across many number of Containers. Volume will not be included when you update an image.**

**If Container-1 volume is shared to Container-2 the changes made by Container-2 will be also available in the Container-1.**

**You can map Volume in two ways:**

**1. Container < ------ > Container**

**2. Host < ------- > Container**

**Step1: Lunch instance amazon linux server and connect.**

**sudo su –**

**yum install docker –y**

**systemctl start docker**

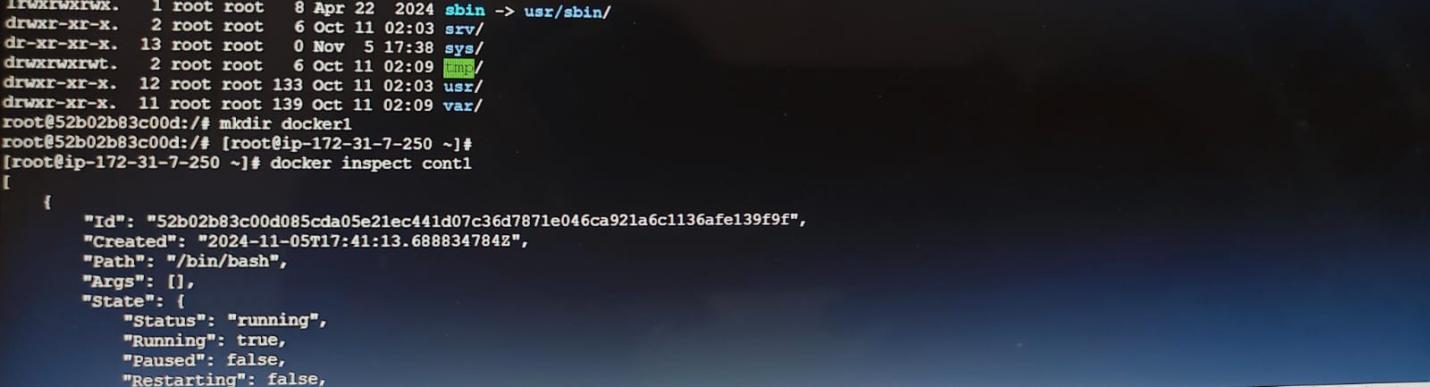
**docker run –it --name cont1 ubuntu**

**ll**

**mkdir docker1 ( create the docker)**

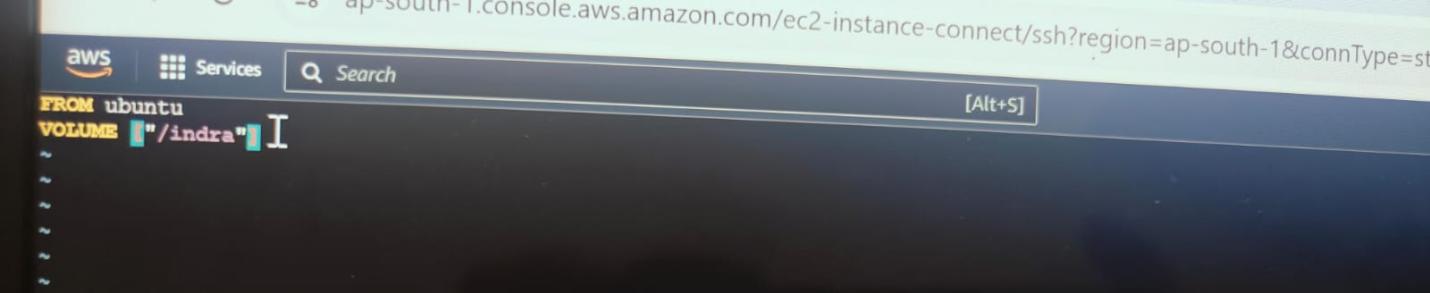
**Ctrl p nd q**

**docker inspect cont1 ( we can see the all information continer)**

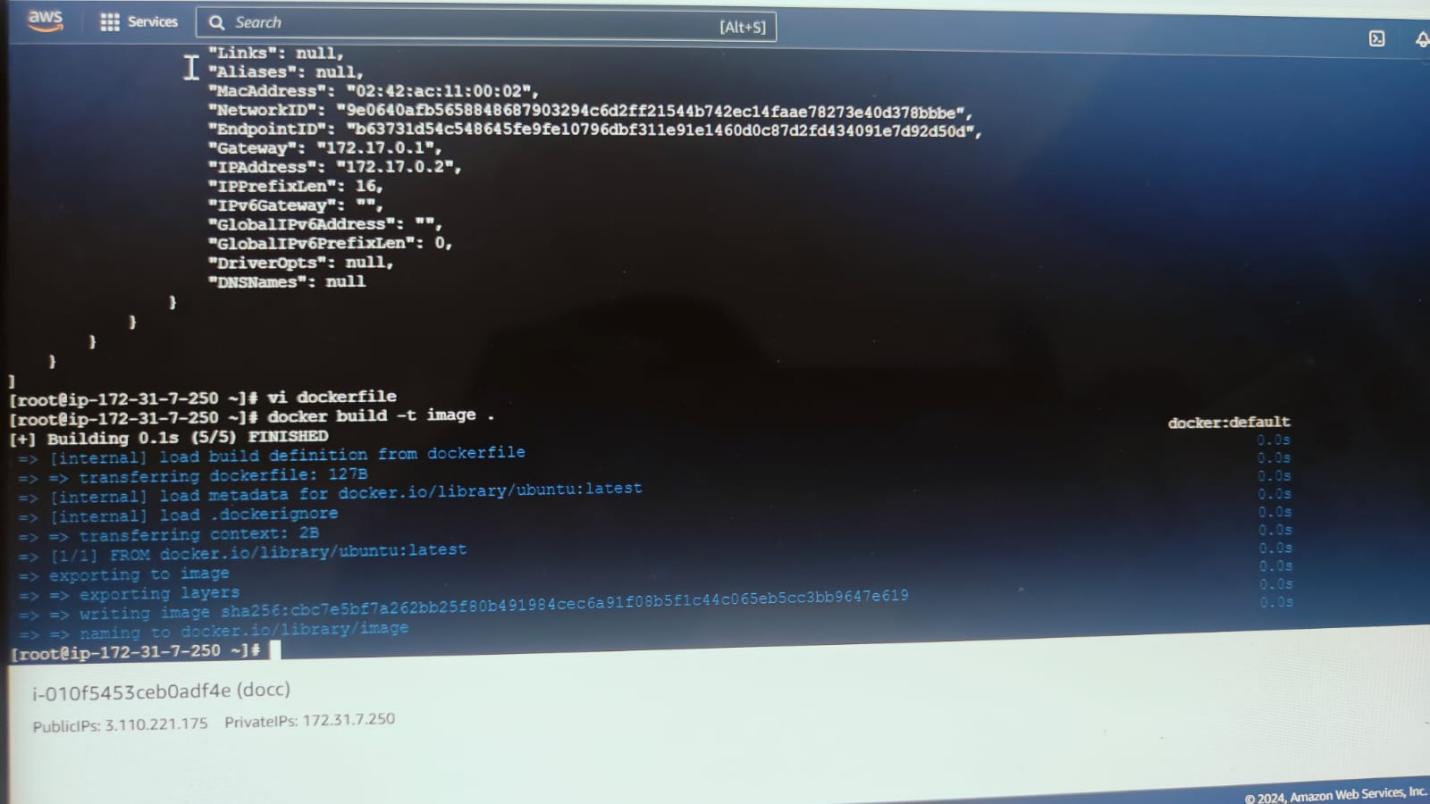
****

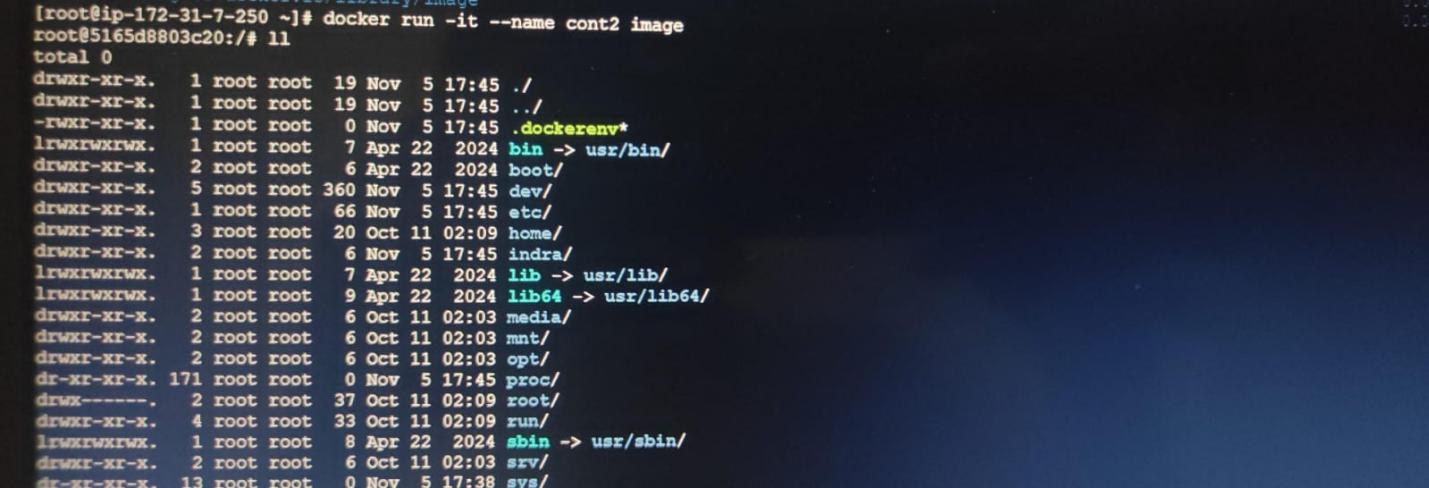
**Step2: create the Docker file:**

**vi Dockerfile**

****

**docker build -t image .**

**docker run –it --name cont2 image**

****

**ll**

**cd indra ( here go to inside the file)**

**ll**

**touch file1**

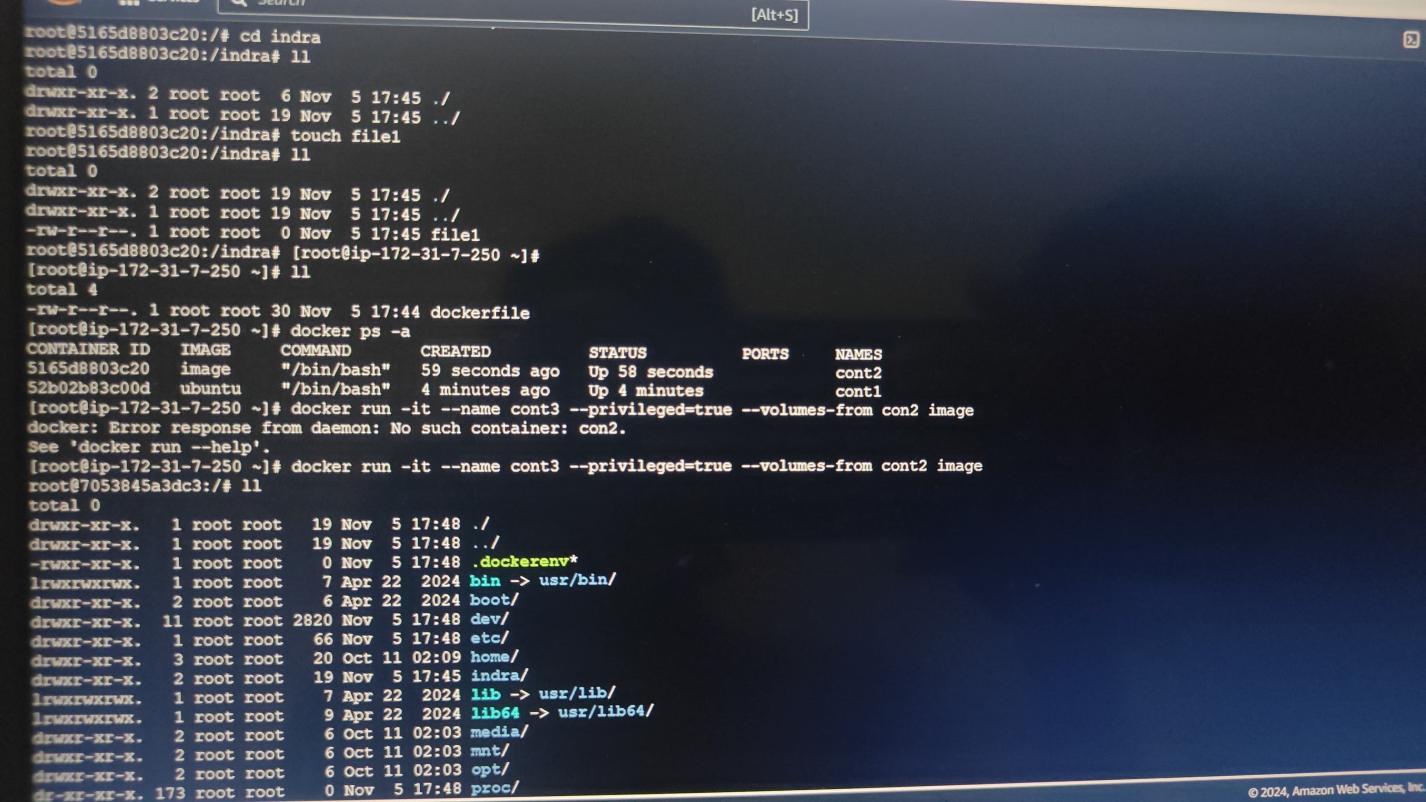
**ll**

**ctrl p q (exit )**

**ll**

**docker ps –a**

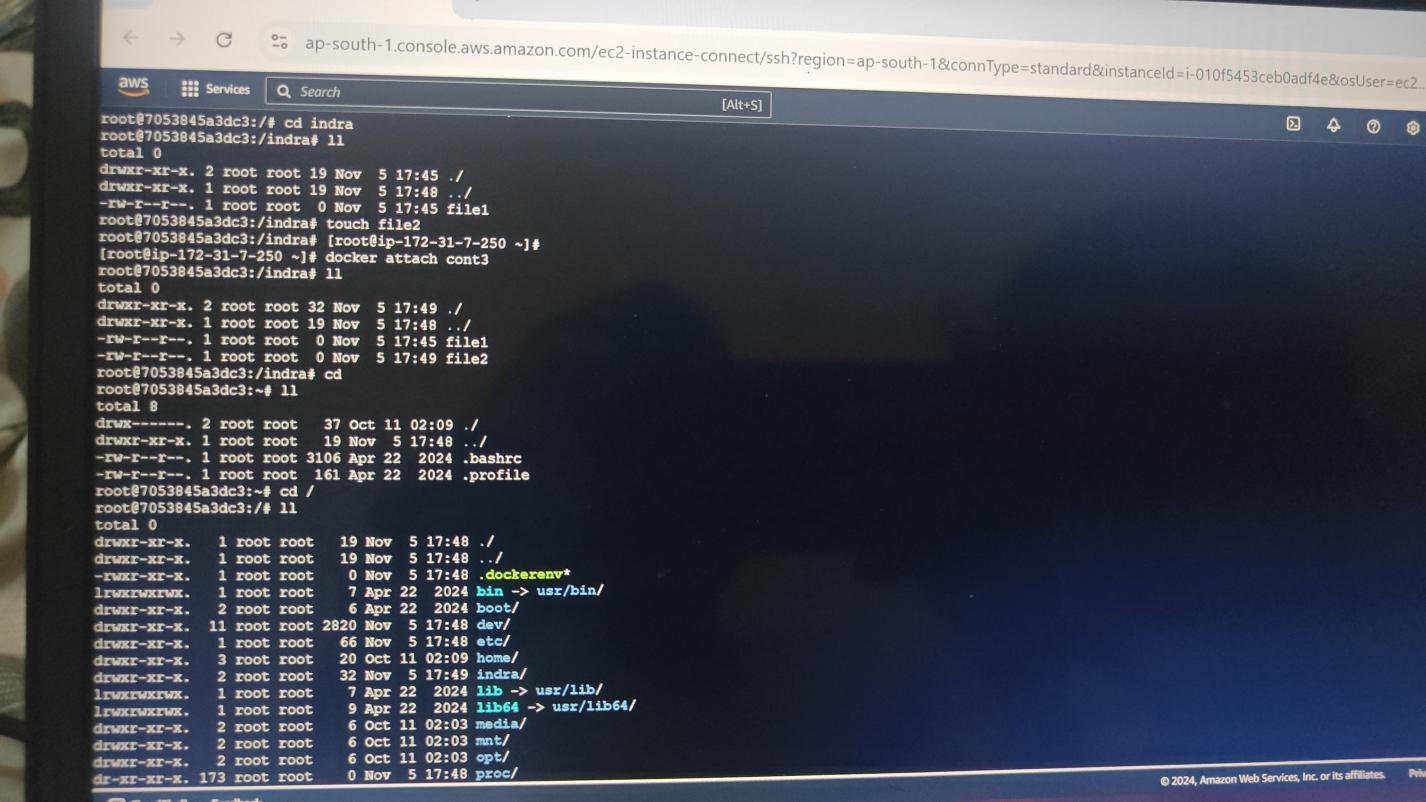
**docker run –it –name cont 3 –privileged=true –volumes-from cont2 image**

**ll**

**cd indra**

**ll**

**touch file2**

****

**ctrl p q**

**docker attach cont3**

**ll**

**cd**

**ll**

**cd /**

**ll**

**rmdir indra ( divice or resource is busy)**

**mkdir dir2**

**rmdir dir2 (delete directory)**

**ll**

**ctrl p q**

**docker run –it –name cont4 –v/ram Ubuntu**

**ll**

**Note: while creation of image we don’t have volume.**

**Here going to inside of the volume:**

**cd ram**

**ll**

**touch file {1…3}**

**ll**

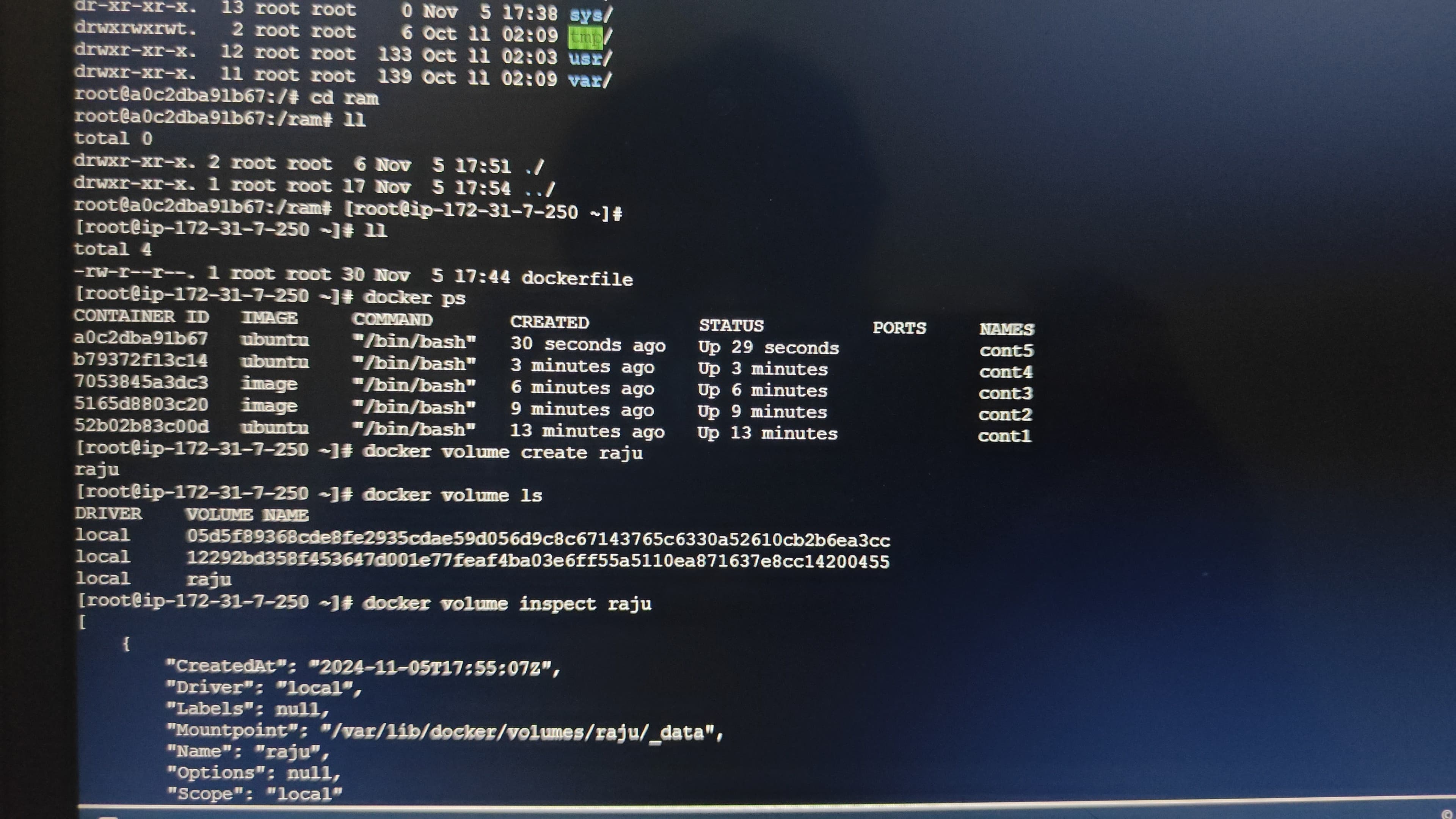
**rm –rf file{1..3}**

**touch file{1..3}**

**ll**

**cd**

**ctrl p q**

**docker ps **

**dpcker run –it --name cont5 –volumes-from cont4 –privileged=true Ubuntu**

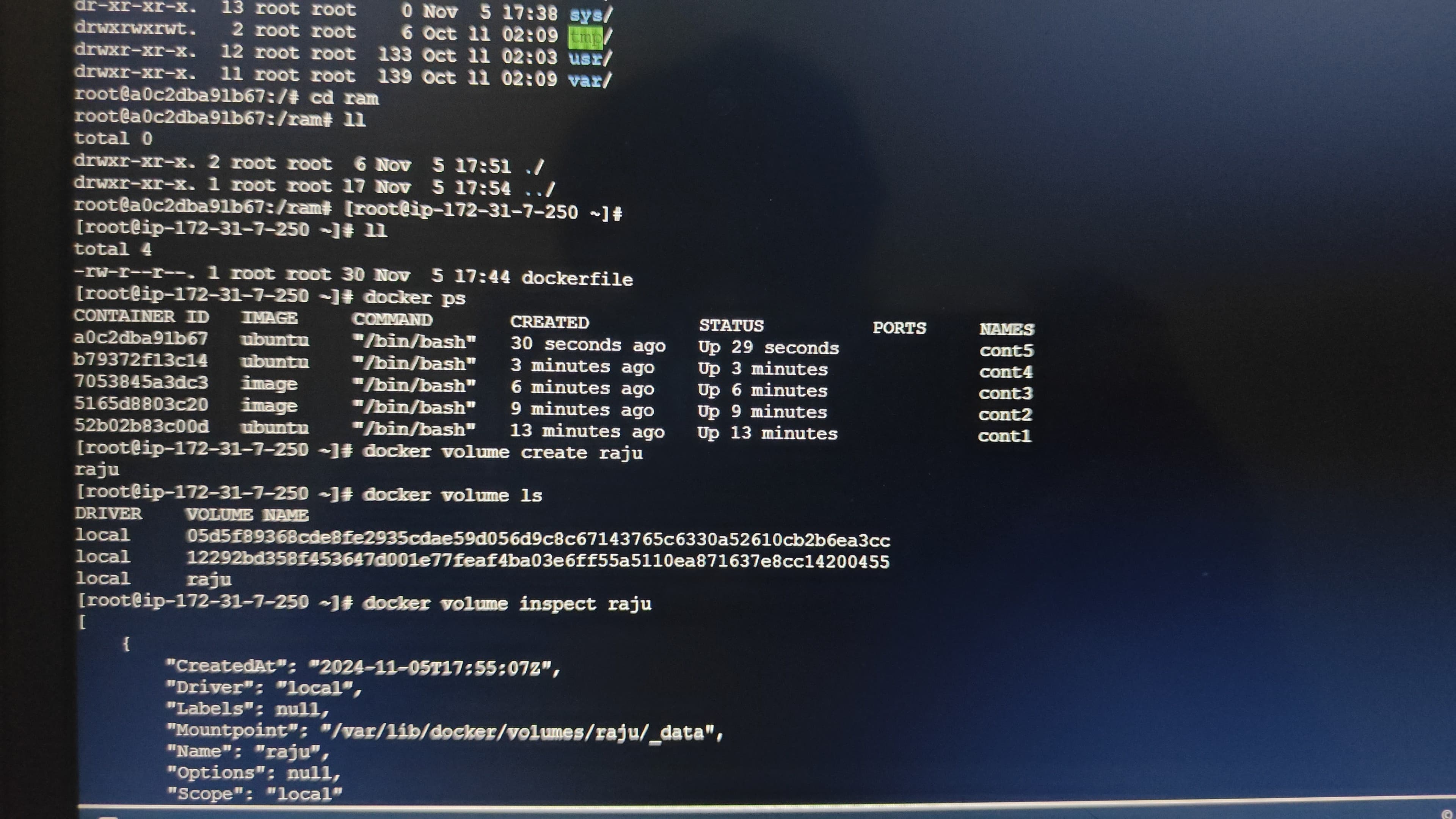
**ll**

**cd ram**

**ll**

**ctrl p q**

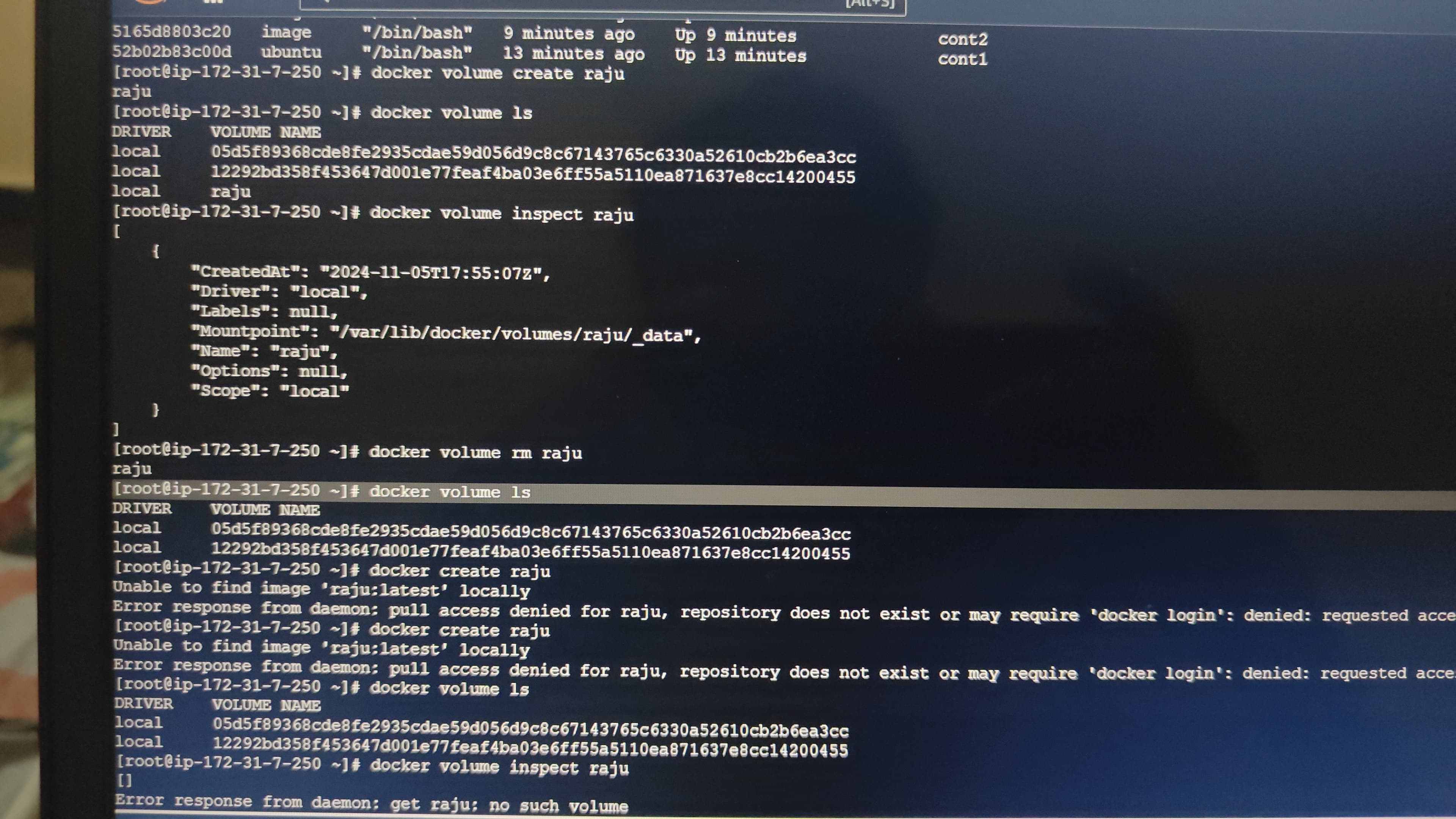
**ll**

**Docker ps **

**Here create the volume:**

**docker volume create raju**

**docker volume ls ( see the all volumes data)**

**volume inspect raju ( when it ha created using mountpoint we can able to attach)**

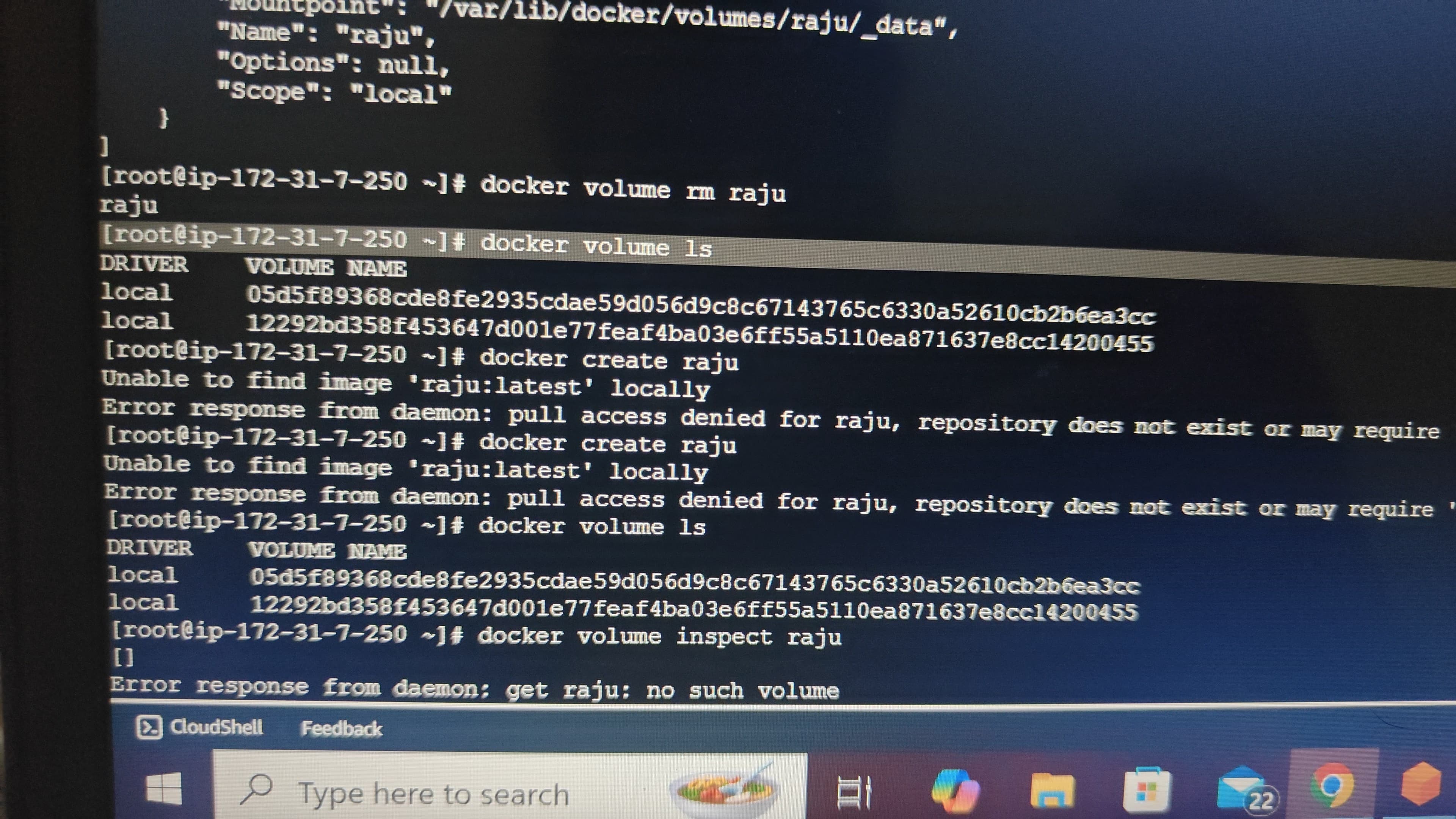
**docker volume rm raju ( remove volume)**

**docker volumes ls (see the volume list)**

**after remove volume and again create the volume**

**docker cretate raju**

**Docker volume ls**

****

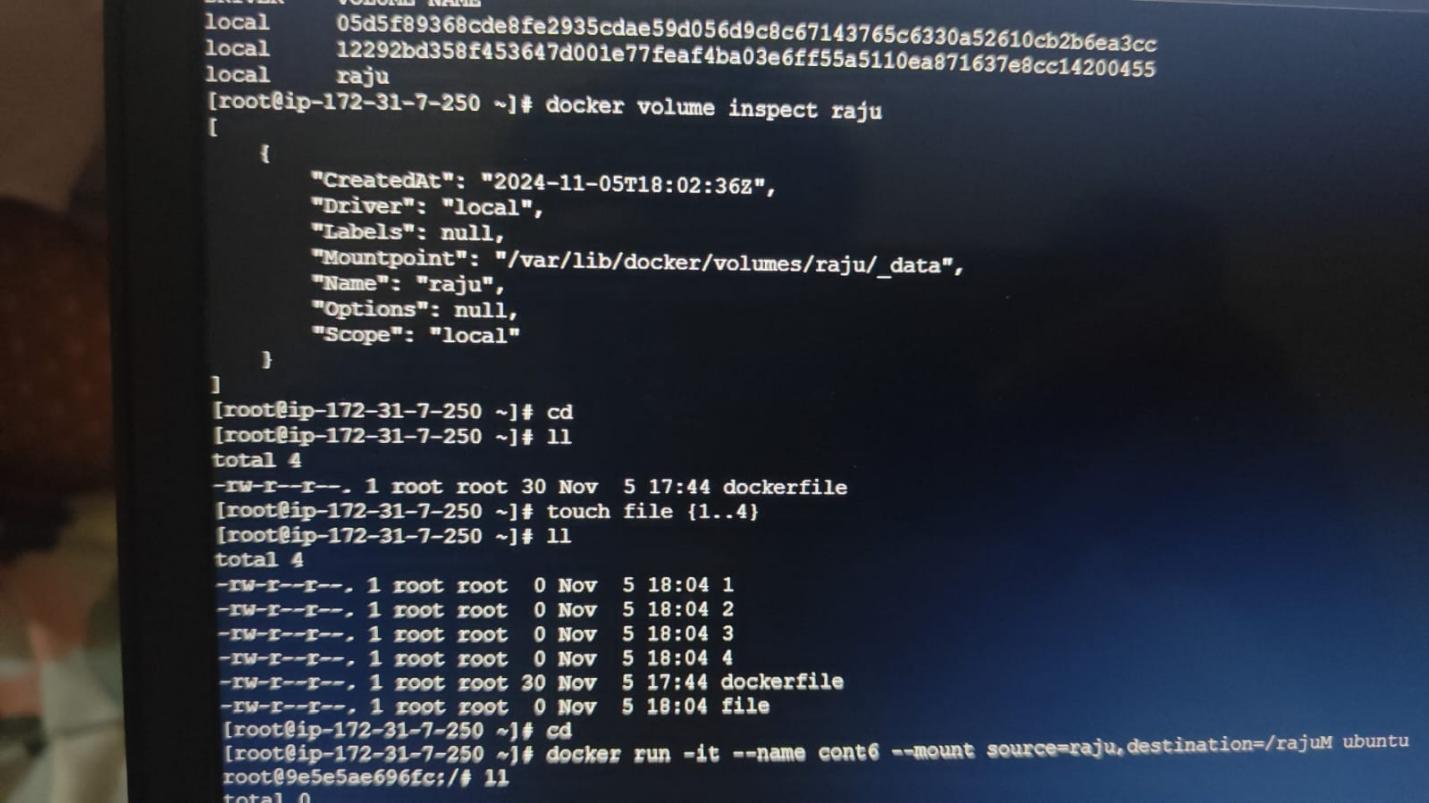
**docker volume lnspect raju**

**cd ( copy the mountpoint go to inside the data)**

**ll**

**touch file {1..4}**

**ll**

**Cd**

**docker run –it –name cont6 –mount source=raju,destination=/rajuM Ubuntu**

**ll**

**cd var**

**ll**

**cd lib**

**ll**

**cd**

**ll**

**cd / ( volume will be created)**

**ll**

**cd rajuM/ ( go to the inside volume)**

**ll**

**rm –rf file{1..4} ( remove the file)**

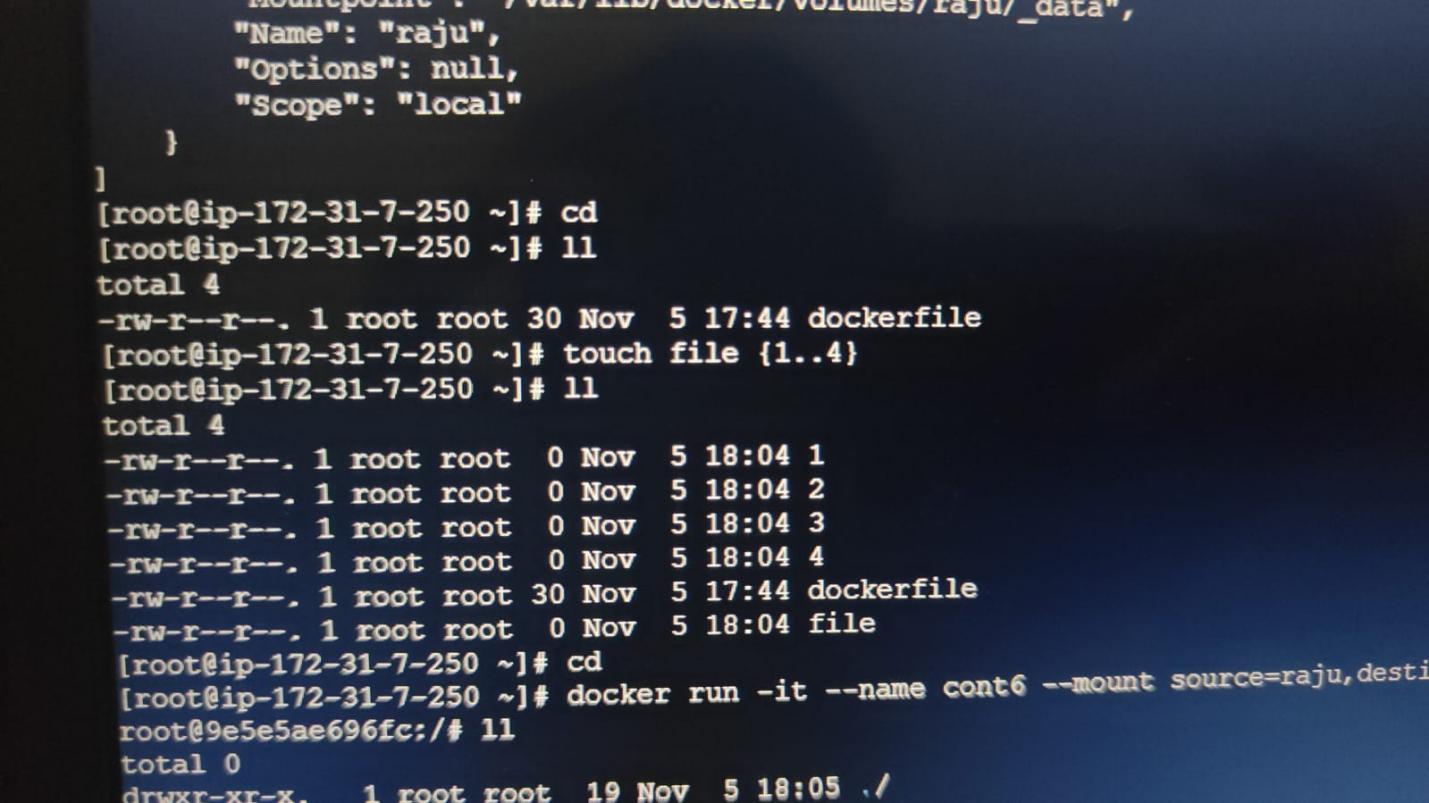
**ctrl p q**

**ll**

**docker volume ls ( here you will see the continer data)**

**docker volume inspect raju**

**cd ( copy “mountup’)**

**ll**

**Here sharing the volume:**

**docker run –it –name cont 7 –-volumes-from cont6 –-privileged=true Ubuntu**

**ll**

**cd rajuM/**

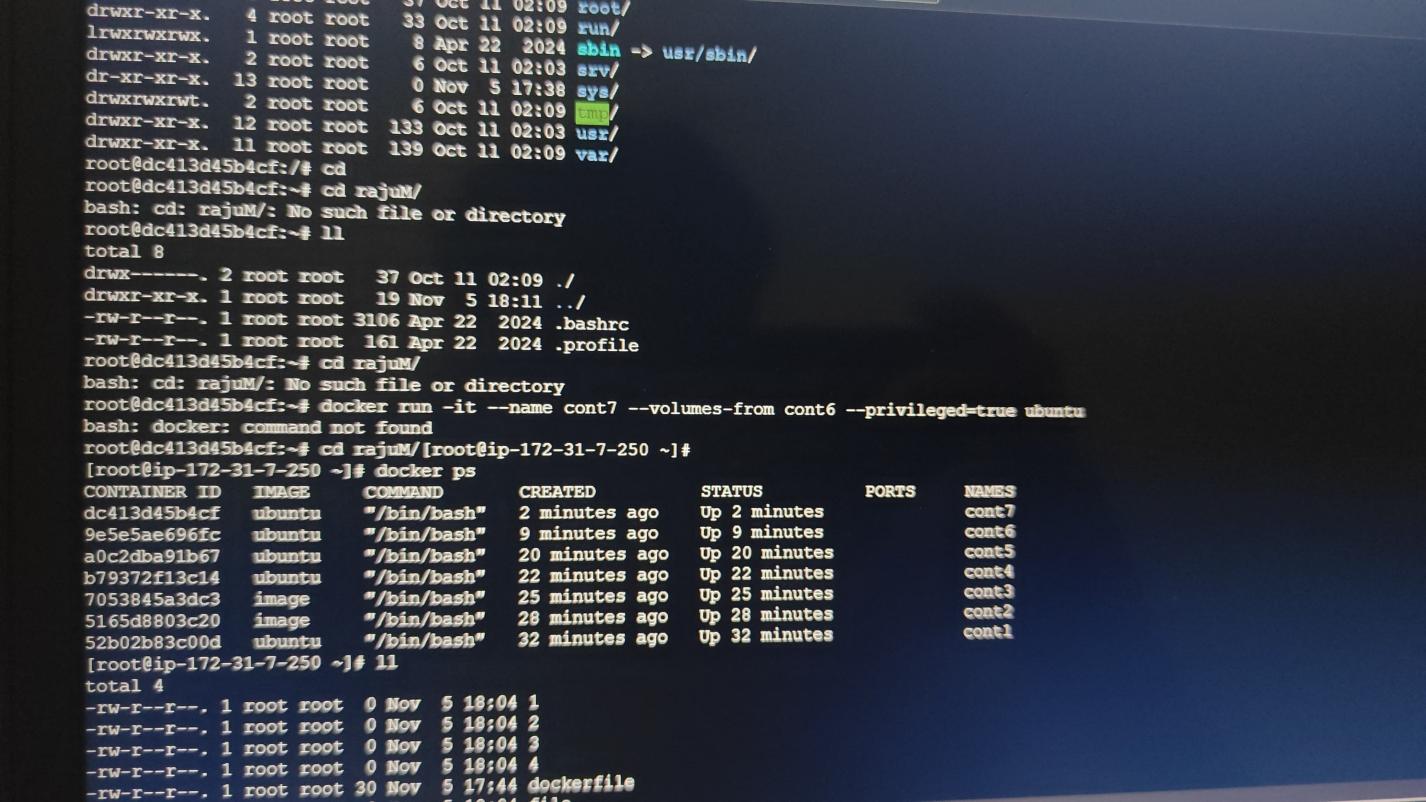
**ll**

**Note : volume will not delete directly and continar will be delete.**

**Cd**

**Ctrl p q**

**Docker ps**

****

**Docker networking allows Docker containers, services, and components to connect with each other and with external components. It also provides isolation for Docker containers.**

**A Docker bridge network is a software bridge that allows containers on the same Docker daemon host to communicate with each other, while keeping them isolated from containers on other bridge networks. It is also default bridge**