**DOCKER**

--> We can start from containerization.

--> Before starting docker we need to understand the Applications from last 20 years.

--> node js,angular js and react js from recently only.

--> in 2012 java and .net this two only there applications.

--> majaroly have java.

--> java/J2EE -- from java-1.5,1.6,1.7 also there in 2012.

--> application like ear

--> ear --> Enterprise archeive

--> web archeive

--> jar --> Java archeive

**--> Enterprise archeive**

Enterprise archive means heavy in this front end is the and backend also there.

.ear --> Frontend+Backend

Front-end and back-end not separate components. (all are same components)

--> catalogue+user+cart+shipping+payment -- this all are same component.

--> Variety of databases are not there. That only one databases.

Mysql or oracle like that -- > mostly will use Oracle DB.

--> oprn resources is very less compared to now. All are heavy heavy licenses and heavy heavy costly.

**: servlet ++ jsp ==** This is imp technology and at the same time nice may technology.

Server(Base for any frame work like spring ) + JSP (Java server pages)

--> A servlet is a mainframe for running web applications in Java. Underground will run only subnets.

--> spring,springboot,java basis is servelet, JSP is frontend.

--> everything is single application.

--> Application size - above 100MB was there min

--> might be 120MB only for Application. This total they will deploy.Everything is single team.

--> Client need to want to change anything immediately we can’t that time we have release process.

**Release Processs**

**-----------------------**

Waterfall model

--> Releases monthly one or twice is nice.

--> For new features

Defects

Our business analyst (CClient side) + Business analyst + Team Manager\_+ Leads + Architectures + Testing+ Build and release

Release Manger --> Coordinate with all teams to make sure release is going as per plan and continuous update clients.

Everything is going good or not,need to prepare document and update the status to clients these all need to do Release manager.

So then release night 9 - morning 6 they did.

Night 12 to Mrng 6AM announced the Downtime.

Stop the application servers.

Build new a[pplication.

Remove old ear, place new ear and then check modules and version.

Restart the server.

Module means dependencies.

Everything is fine it will complete with 15min.

While deployed patch in Prod parallel check develop and testing whether everything is fine or not.

After two it will find out at least one defect.

That Major again need to deploy.

Previous version rollback need to do. If major defects rollback.

Everything is fine also client will ask many question's.

Every-shift persons need to attend the call's. Team leads will manage. If in-case any problem restart the server.

As a rule should not restart need to check logs.

We don’t know error

Logs dump

Traces

Different logs all need to download.

After download restart it. If it’s working fine no problem may be some application slowness.

Slow,no response etc..

No restart

Check logs

Logs dump, traces

Restart

Night 2:30AM application was slow, no response, logs are attached -- to dev

From the side of L2 need to follow up because client will ask.

Not able to replicate the issue. Error not coming now. After Four days discussion ignored the clients.

Bridge cal means all peoples communicate cal --> clients + L2 support + L1 support + service desk (only for cal’s) + net working team + linux admin team + database team + testing team -- all are should attend the cal.

First frent support is L2 support.

Why this many they understood -- because of front end and backend is together

We have two

1. Jqueary
2. Angular JS

Because of this frontend is separate and backend is separate.

How this two will communicate. How backend is connected to frontend.

API -- Rest API

This separation have many advantages.

If backend side issue backend peoples will do, if front end issue frontend peoples will do.

NOW war files

Total back end is war file

.ear = multiple wars + jar (libraries/dependencies)

For running .ear web sphere,web and JBOSS logic is needed.

So these all application servers. For this have separate web servers are there.

War file --> wildfly, jboss

In war file only we have backend.

Same procedures - issue is where will get one clarity.

Last 20years -- in one break through is Rest API -- for this only technology was changed. But issues are same.

.ear -- on work is very difficult.

War file - rotation shift

So now rest API was started after came backend. Again problems.

Catalogue+ user+cart+shipping+payment --> App is down if any component is not working.

If you are opening website also it won’t come from where have problem how we came to know -- need to check logs.

Where we have problem -- need to identify whee we have error -- connect the server -- check the logs.

Then now

**Microservices**

**---------------------**

Catalogue saparate component

User

Cart

Shipping

Payment

All are separate components.

If payment module is not working what about remaining modules whether it’s working fine or not.

One component not working. Remaining components will work.atleast customers browse products.

This is micro service advantages

If one component is not working not maintain the website down.

But there(in war file) if one component not working also cbs was down.

Every component can be deployed in any programming language.

In war file all are combined need to use only one programming language that is only java.

But here everything is separate.

If data is not coming we are infirm them to check.

Loose coupling between the components.

Loose coupling and tide coupling in popular words in IT.

Components always should be loose coupling. High dependency it should not.

Loose coupling -- less dependency

Micro services each one Size is max -- 10MB

**SUMMARY**

**--------------**

.ear here application size was too high.

Iin wat file little bit less.

In micro services application size is to low.

**See one example Joint families vs small families vs individuals**

In one home - 10-15 peoples are there.

Actually that is good -- need big house.

Should be there in landed houses.

Landed house advantages and disadvantages

Advantages

Privacy

Disadvantage

Costly

More time to construct

Maintenance - Electricity, Water, Security etc.

Small families

-----------------

Max 4 mem

Going to town for study or something

Ne need big house 1nhl or 2bhl fine

Here advantage is

Less cost

Less to time to construct

No maintenance

Disadvantage

Privacy

Individuals

--------------------

Hostels, pgs

Advantage

Less cost

Time

No maintenance

Disadvantages

0 privacy

**Compare the application with .ear file**

Application size is big

So home also need big

Physical server - every thing should manage.

No other applications.

This sever is dedicated host.

In this only run in one application.

**.war file**

VM -- Shared

Virtual machine

**Microservices**

Containers

VM - Big break through

Space speed-- containers

With in sec it will create

**Diagram**

**-------------**

This is hard ware. We are installing on OS.

On os application -- this is physical server

VM - physical server

Backgroung we have physical server.

If Application using 1gb -have buy 32GB

Laptop have 16GB

RAM have to keep high

Hard disk in tvs

Resources not utilizing properly

Tis big hardware -- Hypervisor

After this Hypervisor

VM -- Logically divides physical hardware into multiple errors.

This will come guest os - your wish any os -- Windows,ubuntu,centOS

On this Applications

This is big revolution technically

Right side diagram is VIM

In Clouds still using vm only.

If I created ec2 instance what is that one AWS will run background heavy hardware , Install hypervisor

If I asked ubutu providing ubuntu within a minute.

Same like windows and Centos also

Those all run on same underground and same hypervisor.

So this is VM

On this Host os

Every hardware on min one os. Without os impossible to intract.

Here Diadvantage

-----------------------

Have to manage hypervisor it’s vey difficult to manage.

So here advantages

will work

If in underline host os or underline hardware any problem comes - everything will fail(all application).

This is host os . before hardware

Again hypervisor os again OS utilization

For this application 2GB ram and 20GB hard disk remaining all same

Micro services do you think 2gb ram need ah --> Small Microservices no need 2gb ram.

10MB application also no need

I allocated 2gb if whether will use or not it will block.

Generally were it’s wasting ram - OS (os will take 10GB)

2012 one app kbs

I min vedio recorded that in GB

So memory got increased like graphic memory. Size will take extra os.

At least widows that is graphics

Here containersation technology directly came like containerstaion technology.

3 containers - hardware is VM

VM or physical servers that is our wish -- but physical not available now.

In containers - in containres os

Bear min OS - whichever you want that one only.

Application running e whatever I any ivan .

Without mentioning also it’s doen world, noe a ays no word.

Mpc song sice like

Router - emanded os (100MB)

Ib enable device ---> OS (10 Mb) + hard

I need born vedios

Container+barn min os+ system packages+application and code dependencies --> 500MB - this is call at as image

Barn min os -- commands not avilable in Born min OS

**VM vs container compare**

VM (flat) containers(BED)

Costly less cost

More Time less time

More size small size

More resources less resources

Block the resources dynamic resources allocation

Hypervisor no need of extra components

Not portable this is portable(easy to move)

Booting(more time) booting (less time)

More secure less secure (we can achieve security)

Container will not block resources however they want that only will take.

VM server you have allocate the RAM.

VM physical server - no need to give any RAM data.

**interview question**

**Compare vm and containers?**

**Ans))** VM Resource utilization will do blocking RAM or ROM. But in containers will not block.

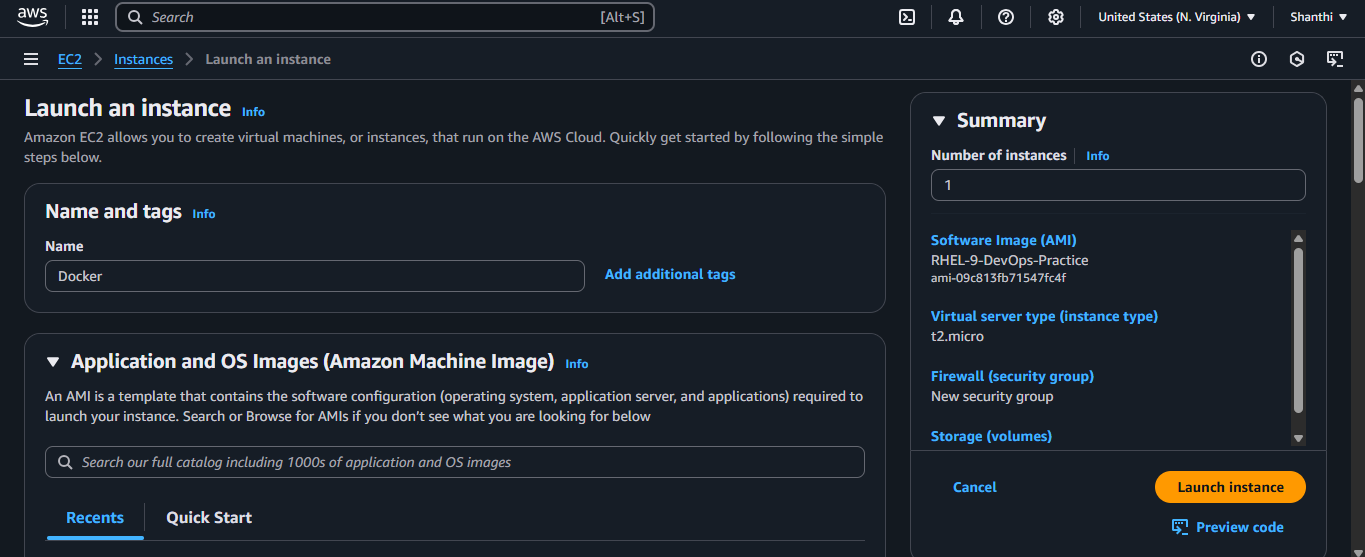
Booting means one system have to on and off. If traffic was increased immediately then want on another system for that automation it will time 5min in VM. Containers will take 5sec.

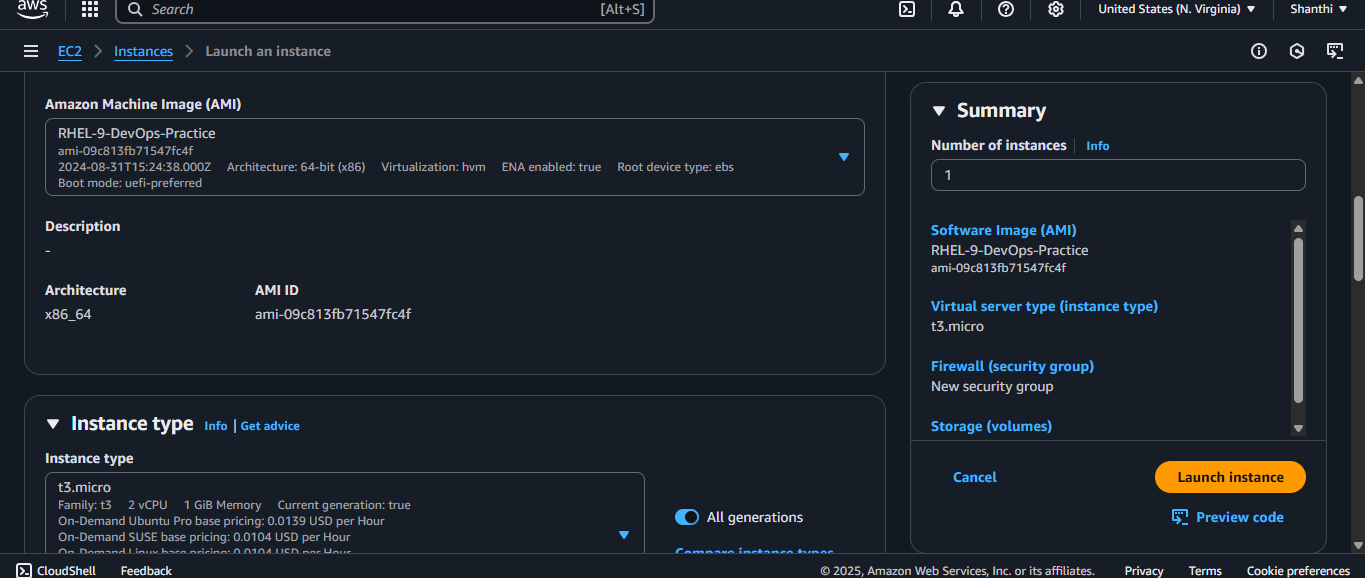
**What is the popular software in containers?**

Ans)) Docker

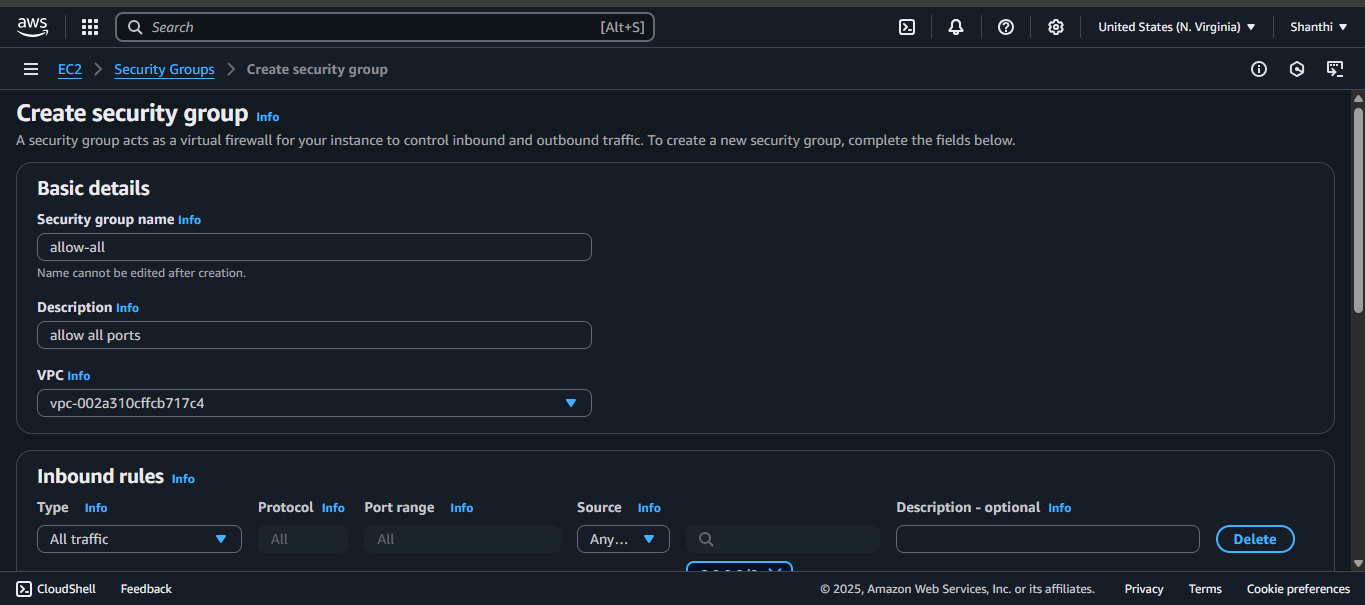
Create One VM and download docker

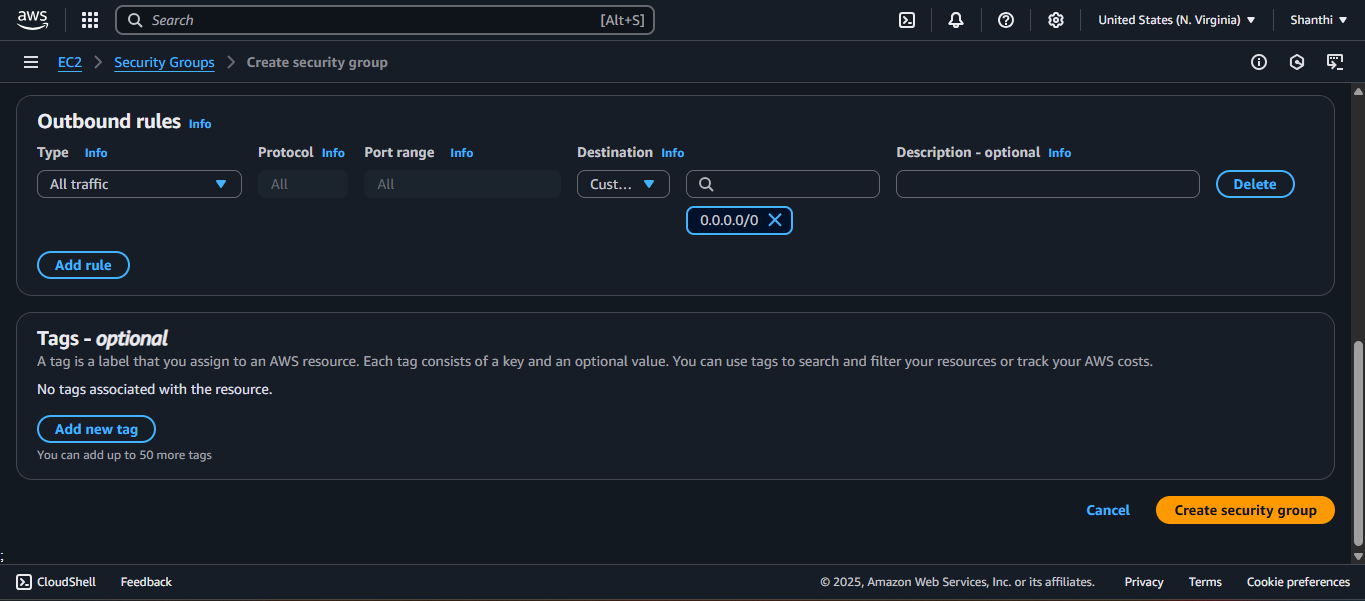
Create lanch instances

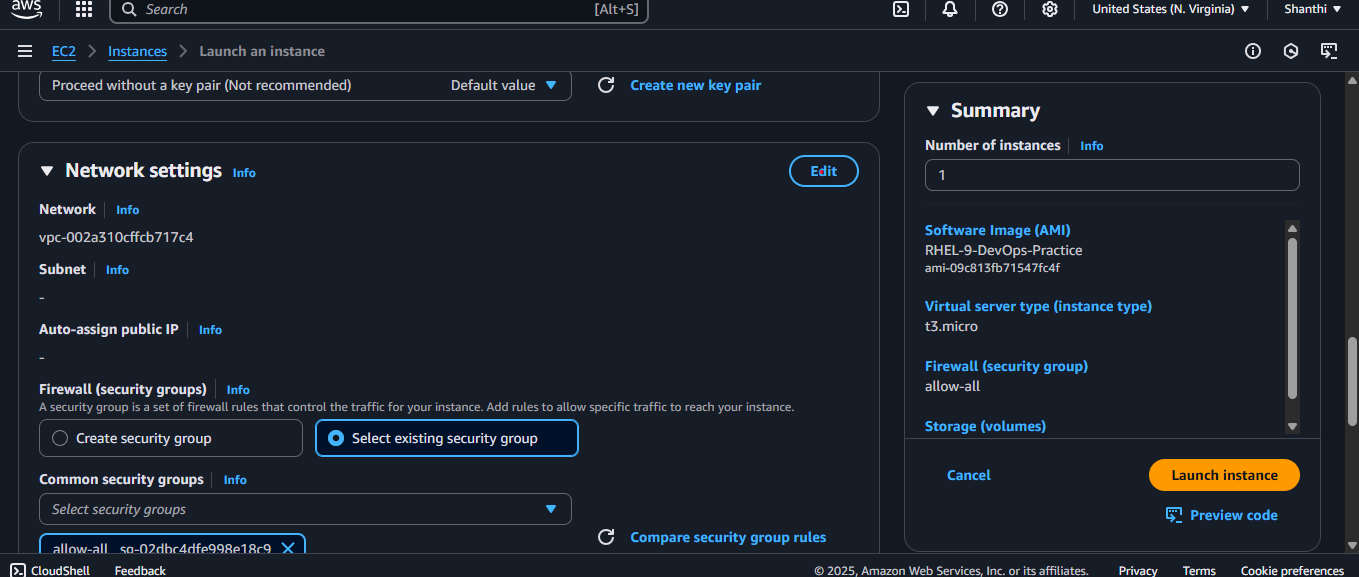




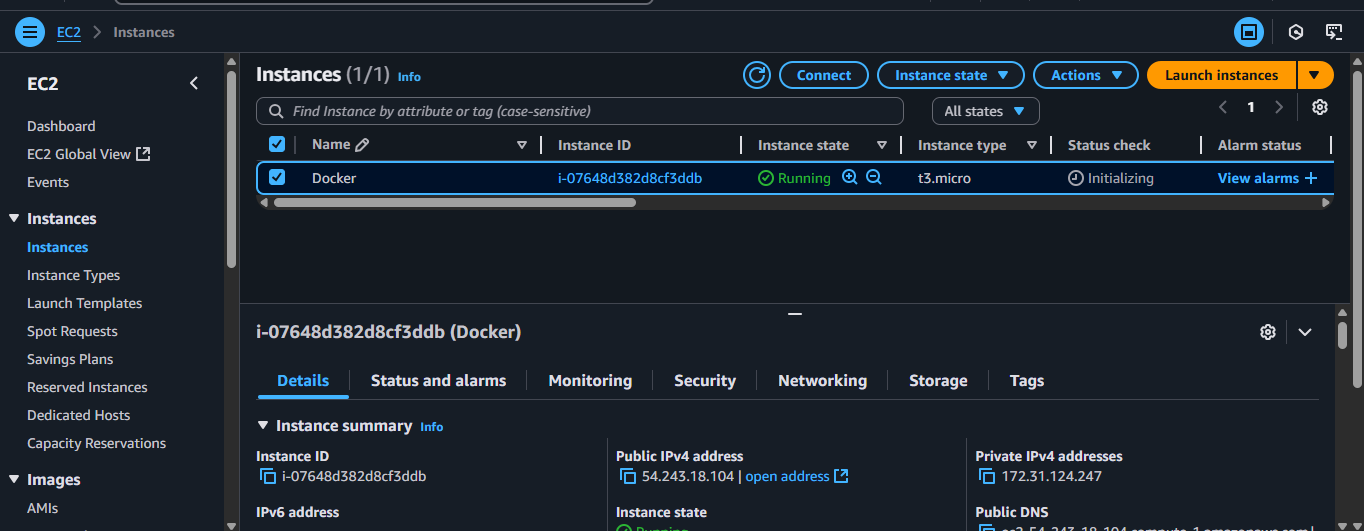
Create security group





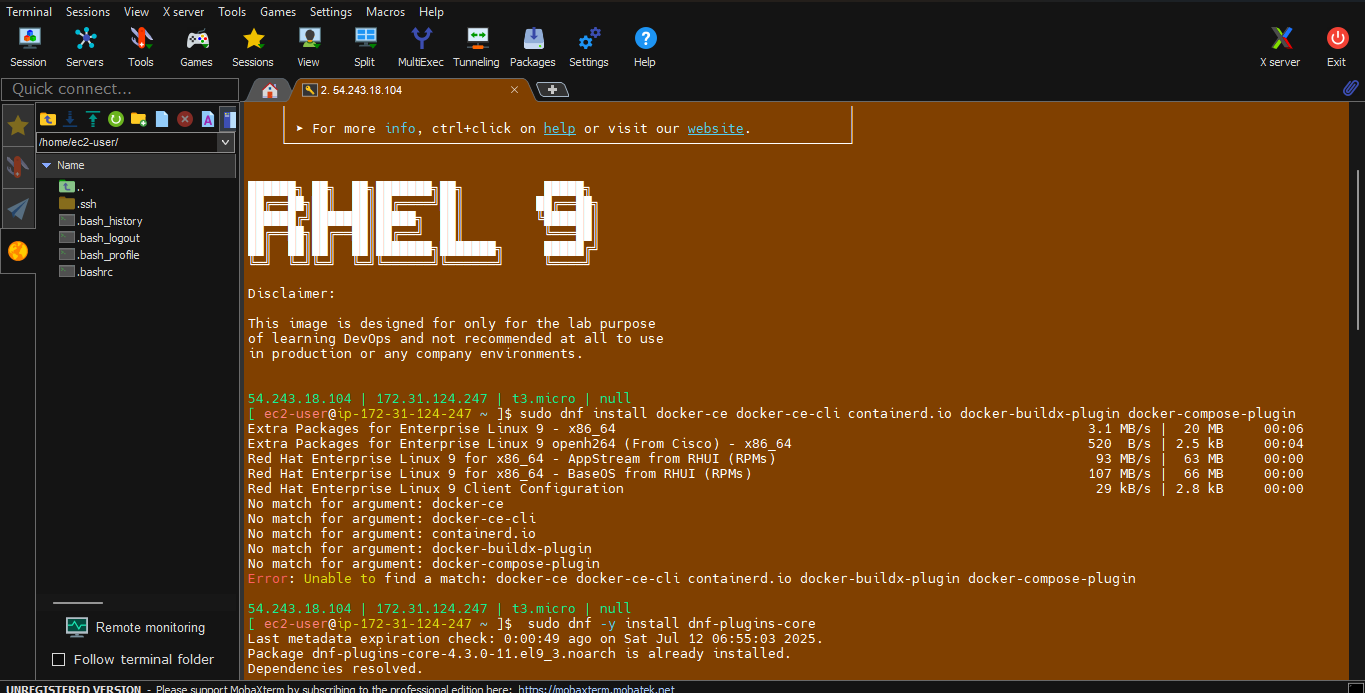


Launch the instance and connect the server



sudo dnf install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

Error got

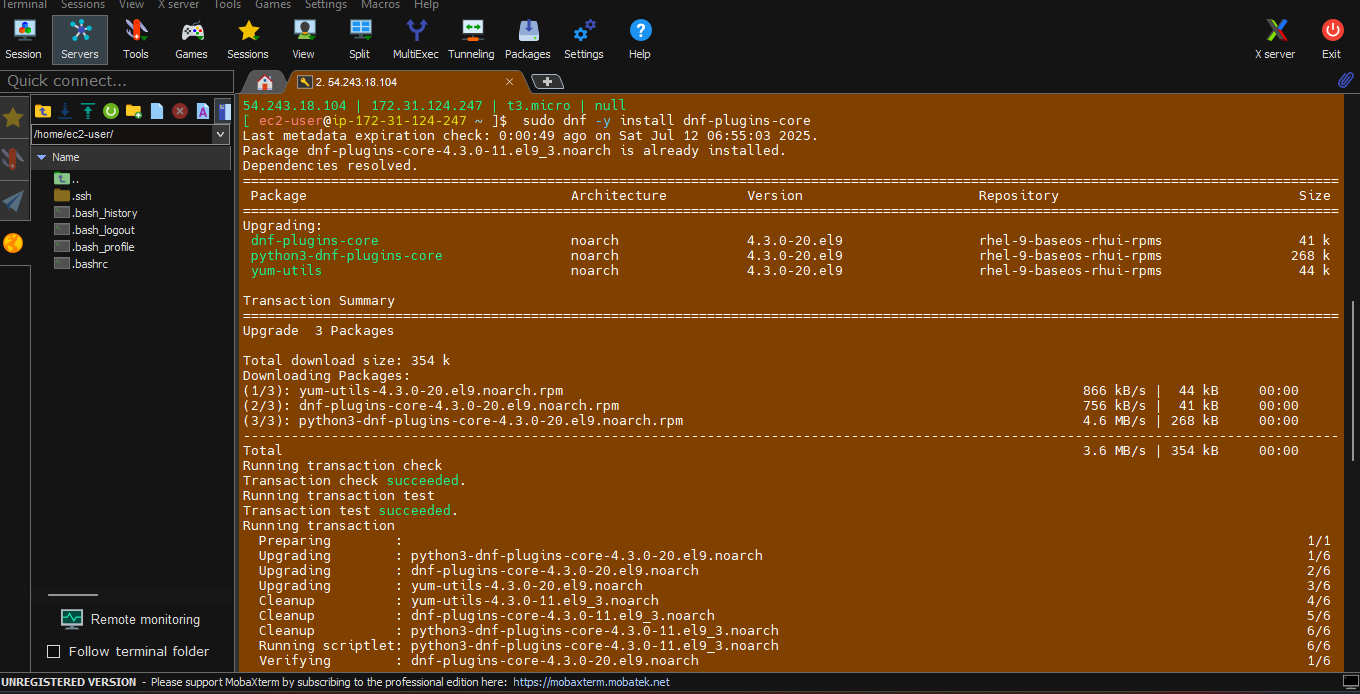


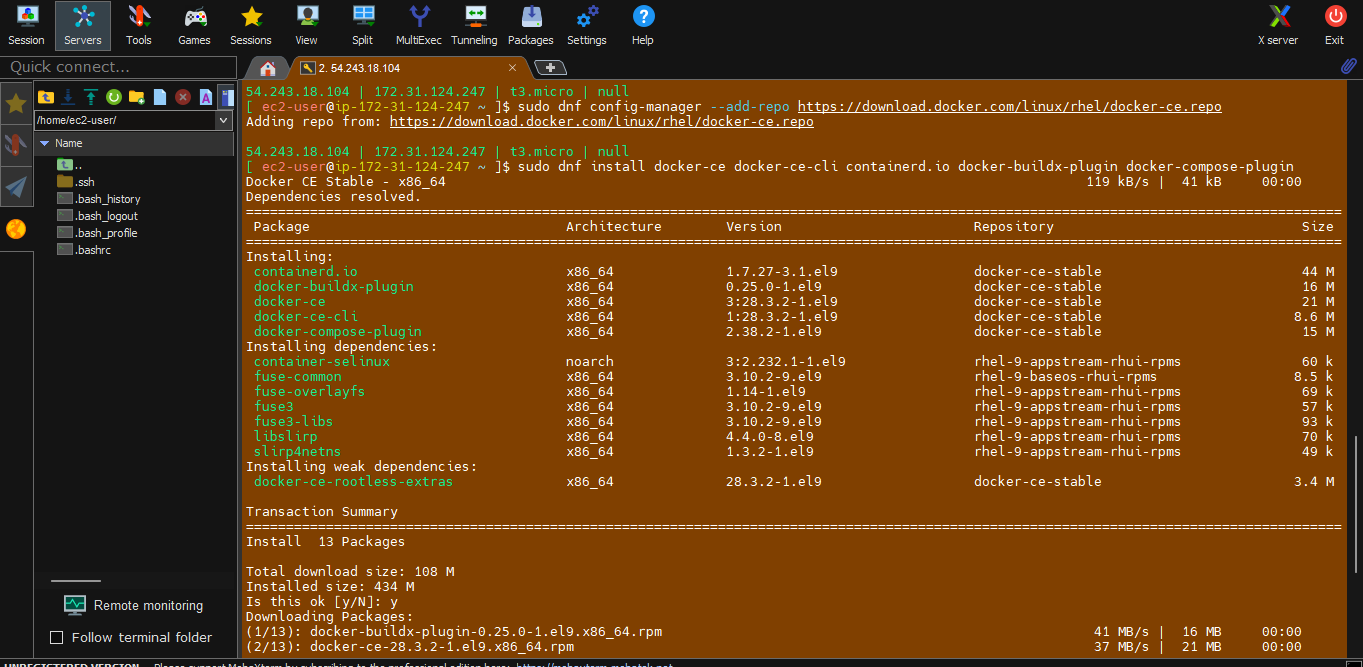
Repo need to add

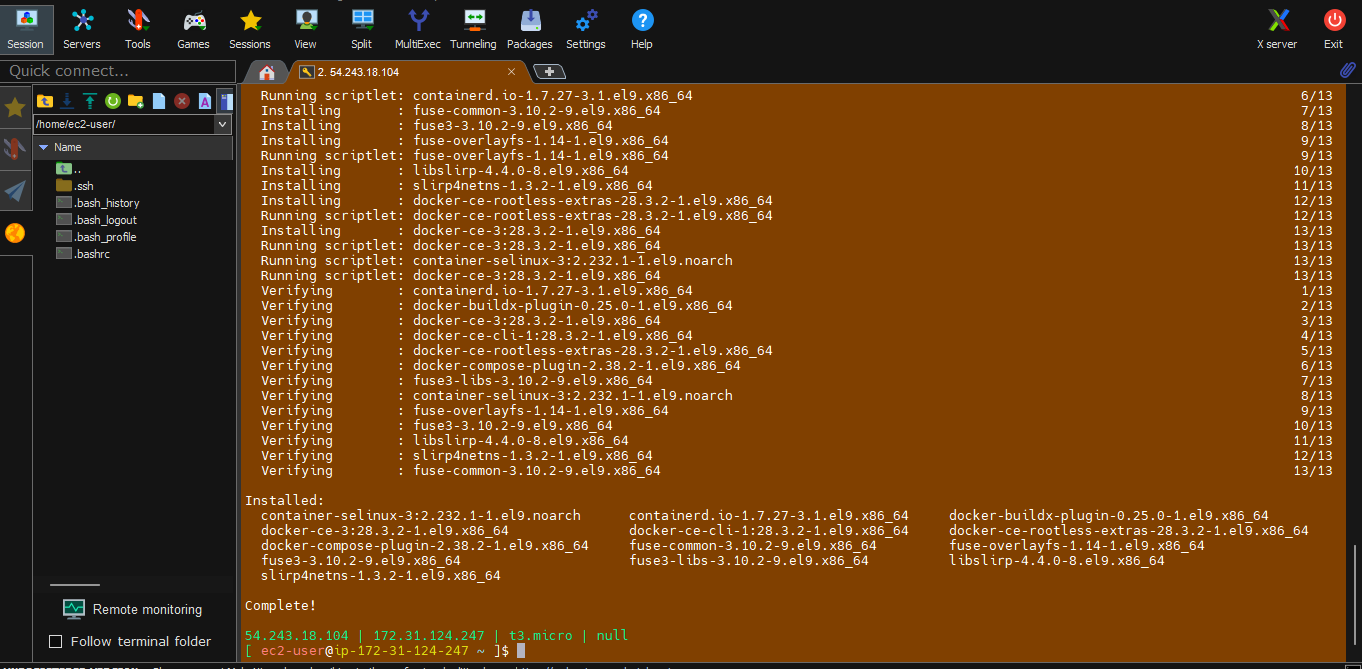
sudo dnf -y install dnf-plugins-core

sudo dnf config-manager --add-repo <https://download.docker.com/linux/rhel/docker-ce.repo>

sudo dnf install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin



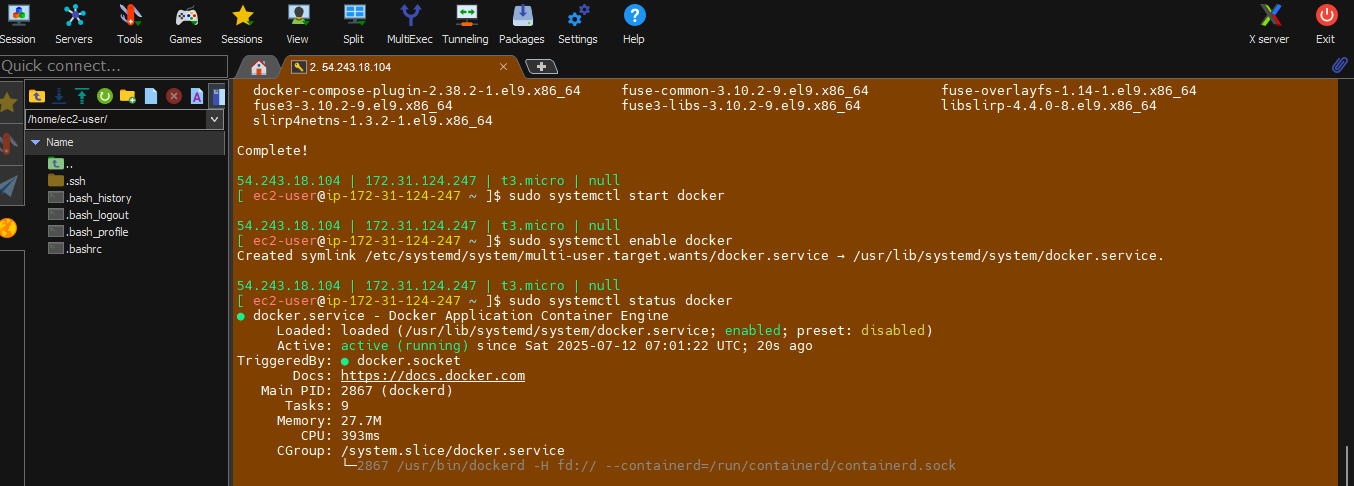




sudo systemctl start docker

sudo systemctl enable docker

sudo systemctl status docker



docker ps

sudo su -

docker ps



Incontainers we sholud not take root access

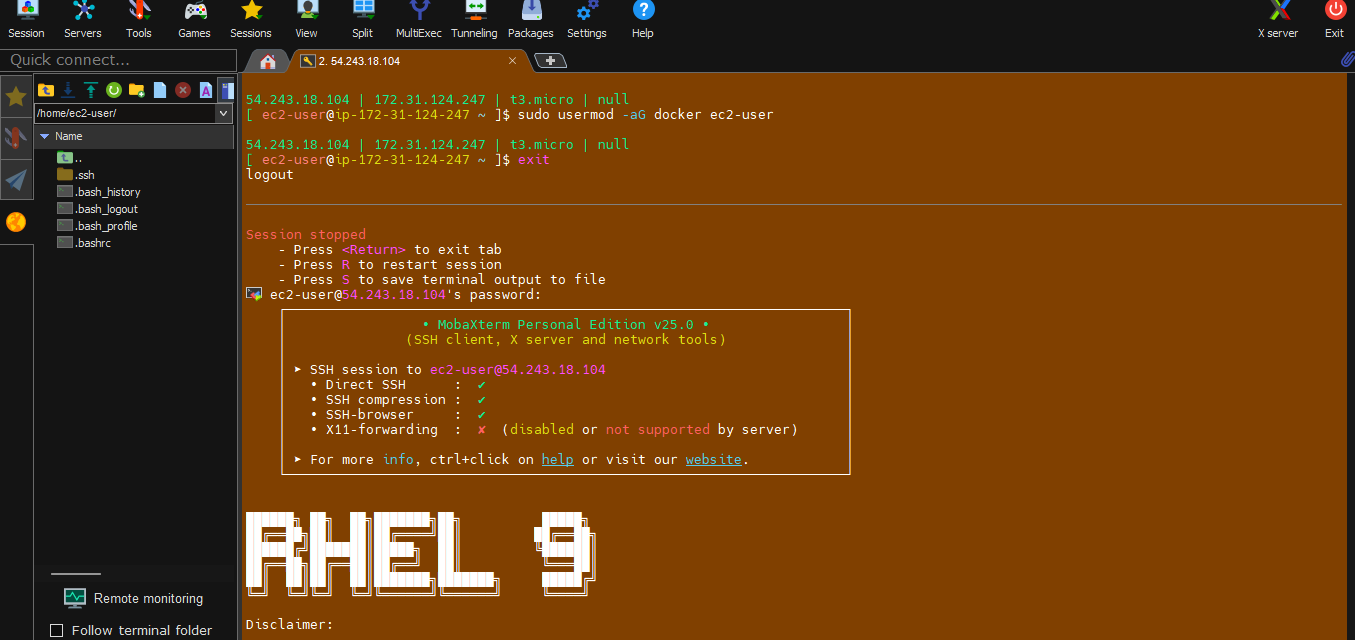
We can address

First will do normal user

Add your normal user to docker group

sudo usermod -aG docker ec2-user

Need to exit and login again



docker ps



Docker ps means processes.