AWS

Amazon Web Services provides on demand cloud computing platforms to companies, individuals, governments on a metered pay as you go basis.

These infrastructures you can choose in a particular region means the applications can be run within specific region and also data will be within the region it doesn’t go outside the region.

It offers solutions for computing, storing & networking that works well together

1. Computing: Computer which is the basic necessity to do anything
   1. You can choose the OS you would like to launch like Linux, Windows, Mac, Ubuntu and so on
   2. You can choose RAM, Processors, Volumes and so on.
2. Storage: You can choose storage
   1. Volumes
   2. Databases
3. Network: You will get a computer with internet so that you can install any software’s you want like Java, Git, Maven, Node.js, Docker, Http Server and so on.

AWS saves cost: You can easily switch your platforms or environment in less cost, i.e., old platforms which you feel is not required can be terminated and no bills will be generated.

Note: If a company purchases any platform they had to pay full money on the platform, if you want a new platform they have to pay full money for that new platform then old platform will be waste of money after some years.

AWS account:

Root user: main Admin

IAM user: restricted users

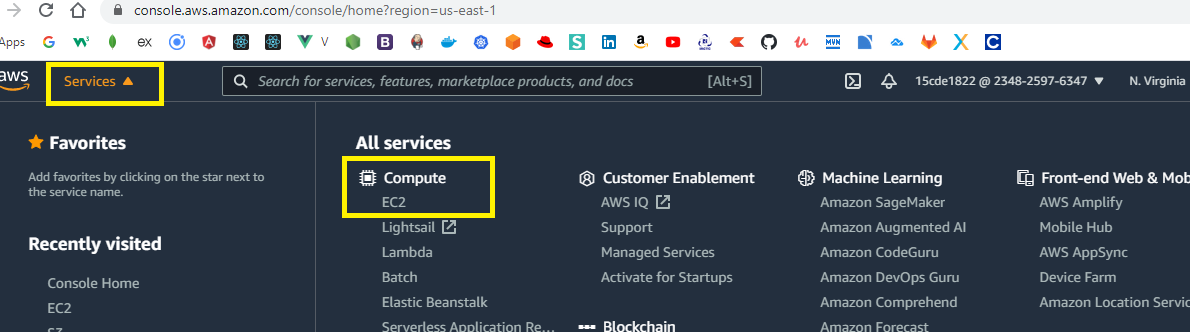
AWS provides many services like

1. EC2(Elastic Cloud Computing): Machines that you can access remotely
2. S3 (Simple Storage Services): A bucket or a container to upload files/folders and so on
3. RDS: A service to launch different relational databases like Mysql, oracle and so on
4. DynamoDB: A service to launch No-SQL database
5. Developer tools: Code commit which is like a GIT.

You can learn AWS from the below link

<https://docs.aws.amazon.com/>

EC2: It is used to create a virtual machine in cloud



Once you select EC2 instance & launch the machines you can access these machines either using

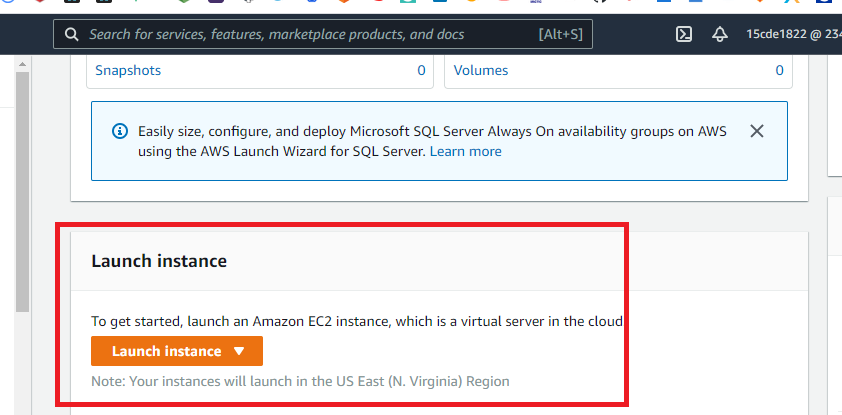
1. Putty & Putty gen
2. SSH Client - Git bash

When you can launch the machine you will get a keypair that should be used to connect to the machine.

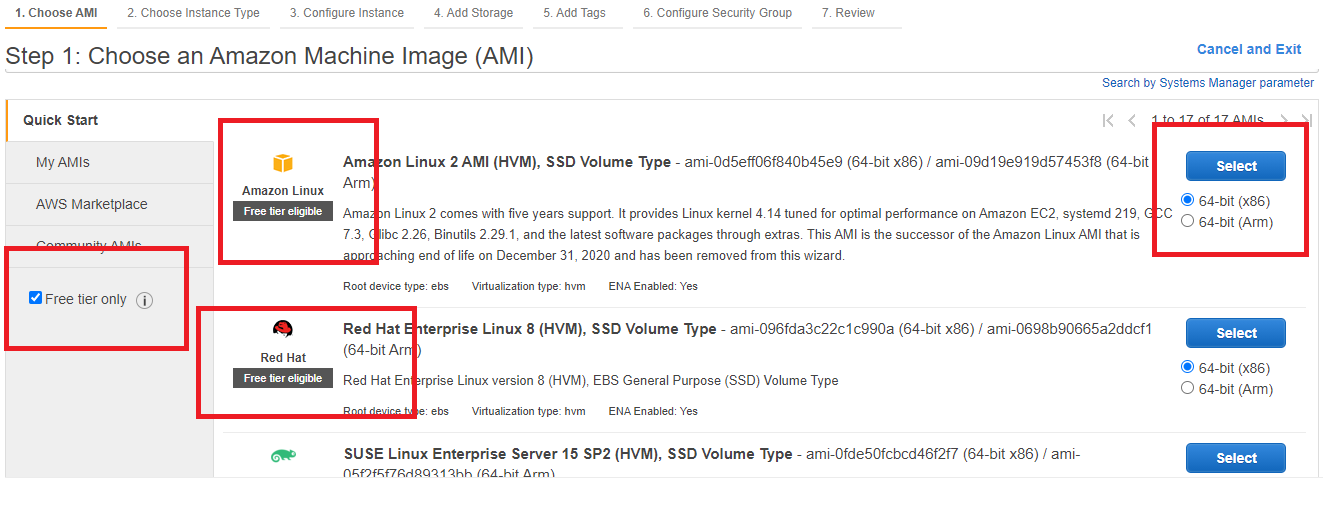
The keypair will be in .pem extension however putty will not recognize .pem it needs .ppk extension, to convert .pem to .ppk we need putty gen, so using putty gen we will generate ppk file and we can use the ppk file and connect to the virtual machine.

Step to launch EC2

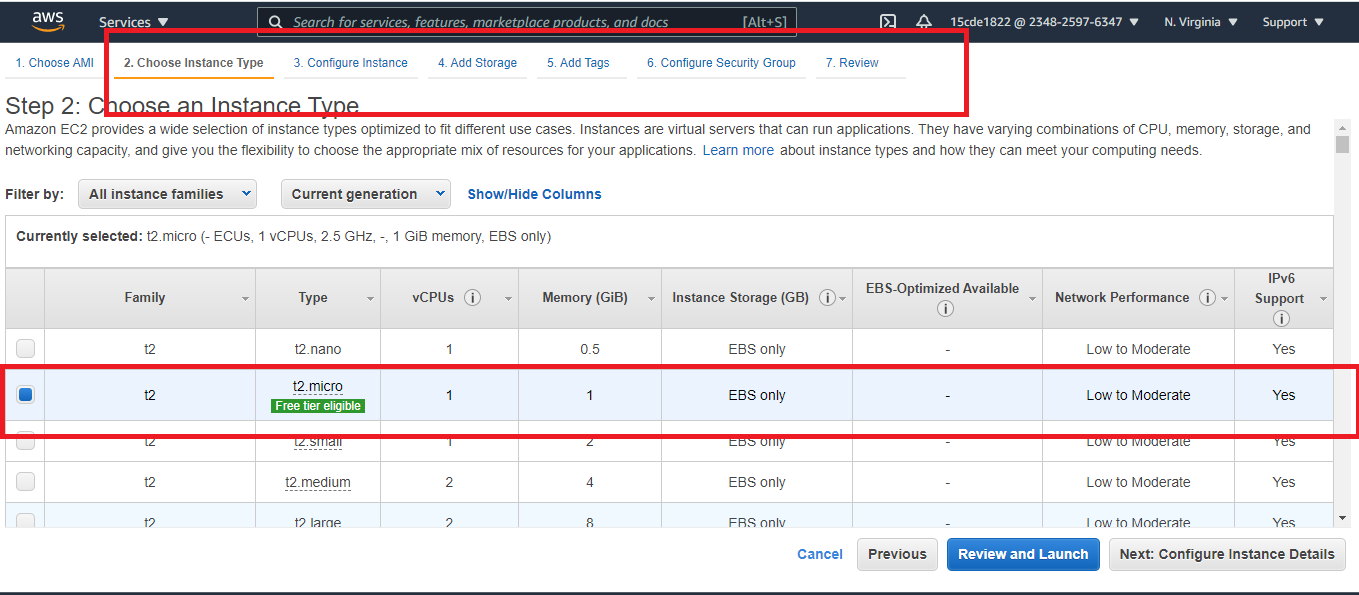
1. Select EC2 from the AWS services
2. Select Launch instances



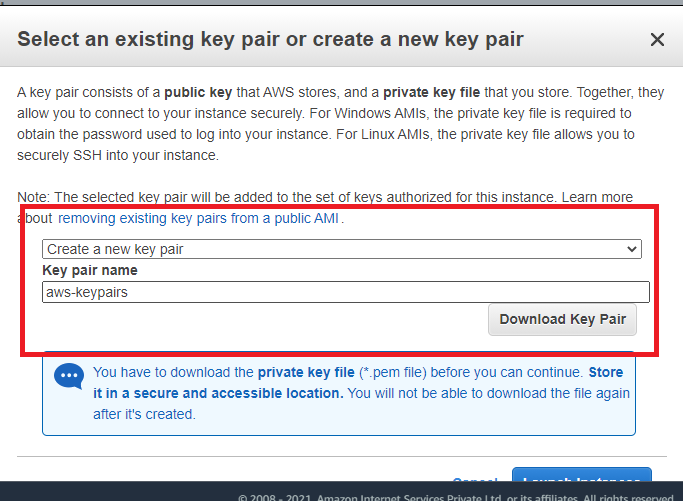
Now you can select the machines which are eligible in free tier



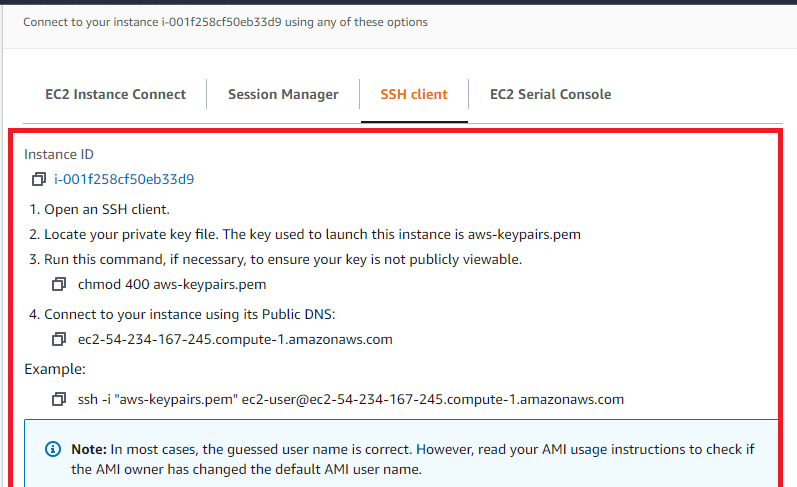
You can see the Instance types



Keep all the default options as it is and launch, you will be asked to create a new key pair create that and download it in a location which you can access.



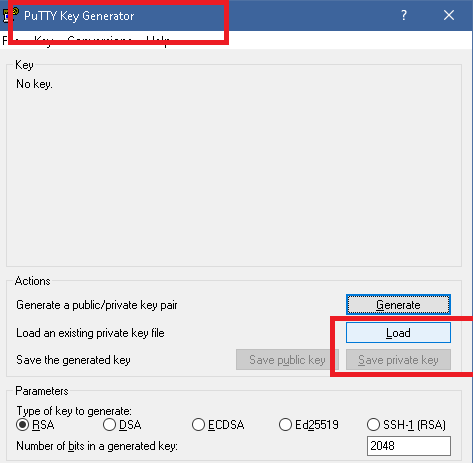
You can select the instance and connect which gives you the commands to connect to the virtual machine from either SSH client or putty



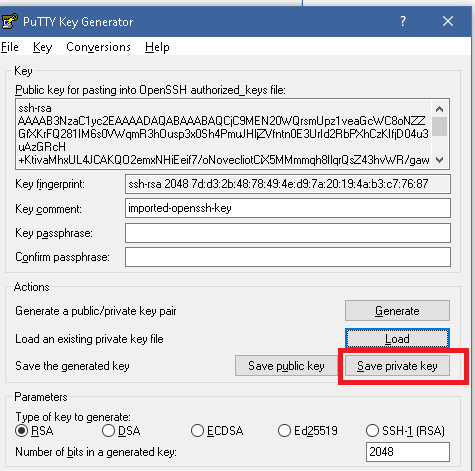
Firstly we will use putty gen and generate the ppk file and using putty we can connect to the virtual machine by loading the ppk file

Steps to generate ppk file

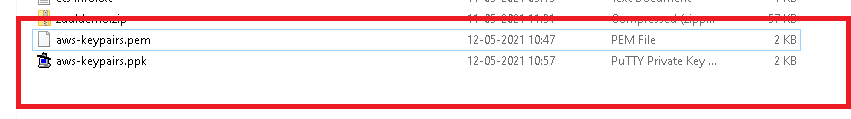
1. Open putty gen, load the pem file



1. Save private key with a name in .ppk extension

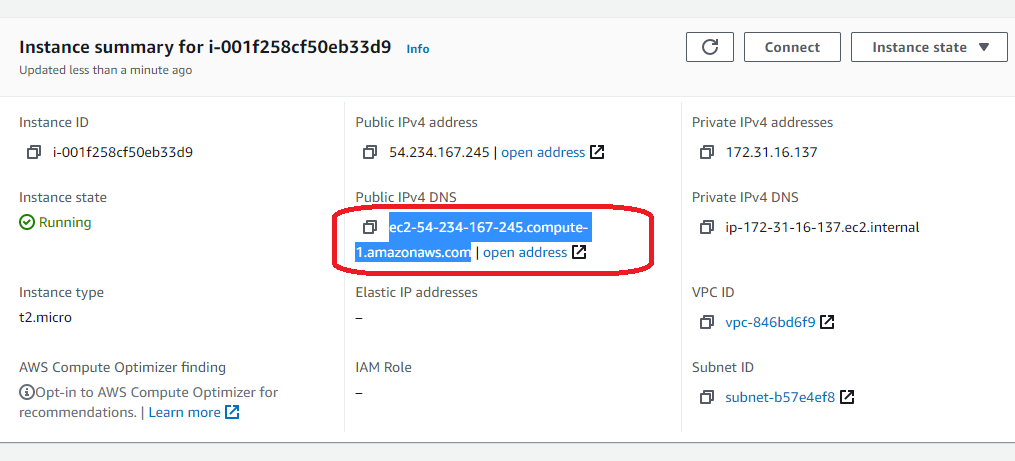


You can now see the ppk file



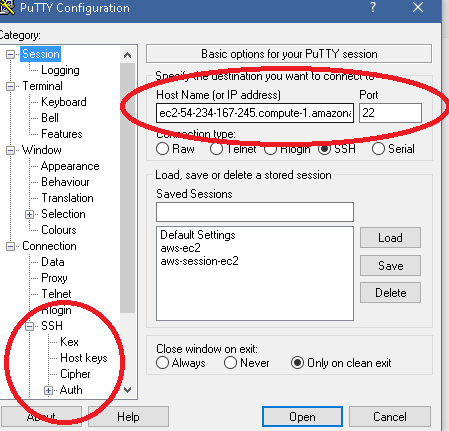
Now you can open the putty and load this ppk file in the SSH auth

You can click on the instance to identify the hostname

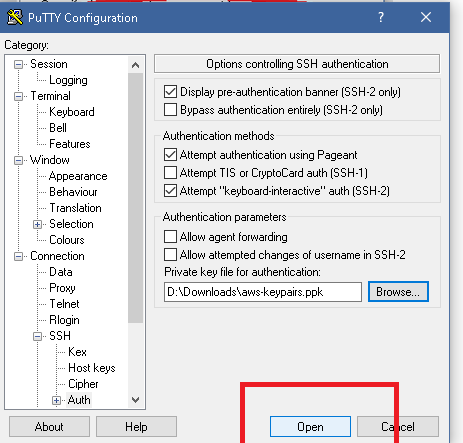


Copy the DNS address and use in the putty hostname

Firstly enter the hostname

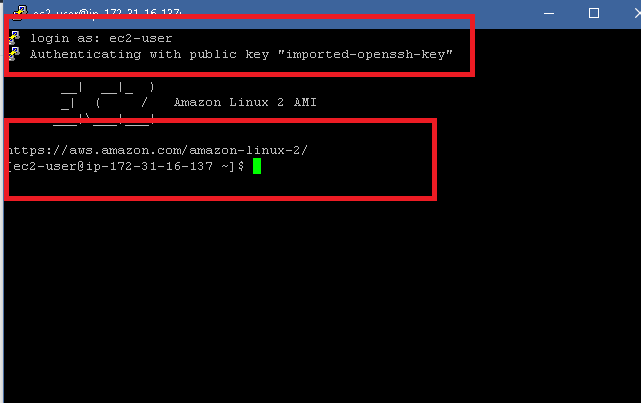


Select the SSH -> Auth and browser the ppk file



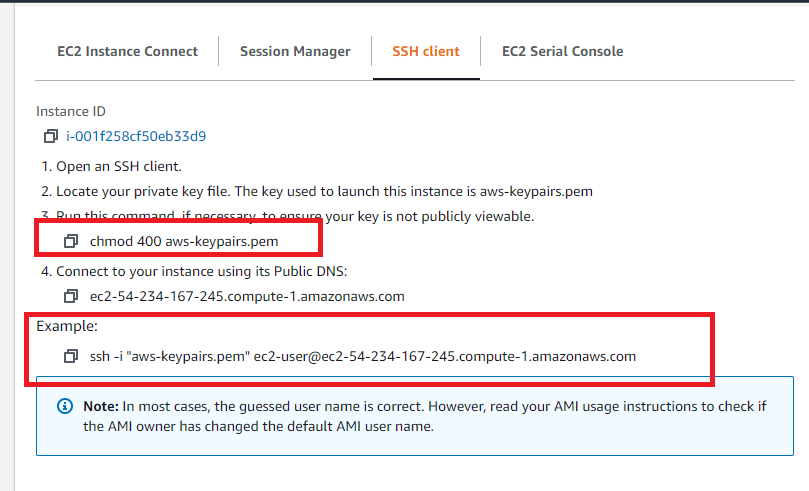
You will get a terminal to connect to the machine as below

You will login as ec2-user

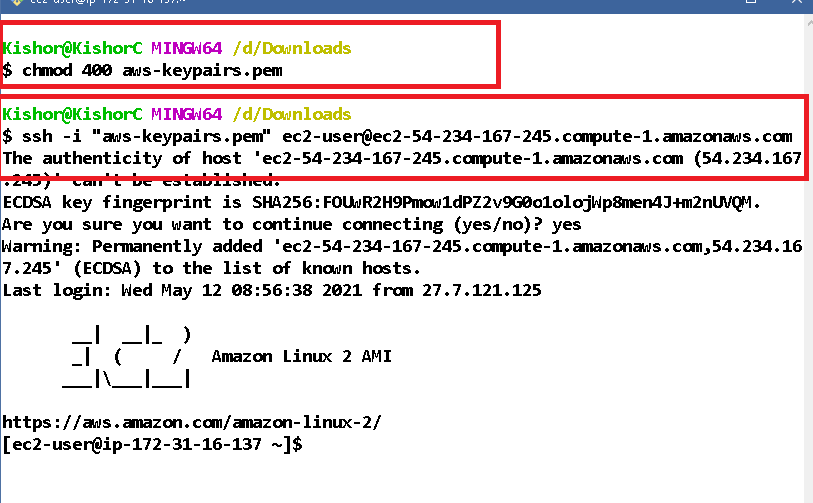


Another way to connect to the instance is using the ssh client

Select the instance & click on connect you will get the page to instruct how to connect



Open git bash and enter the commands



Note: Git bash must be opened in the location which has key-pairs.pem file

Installing the software’s in the virtual machines

Note: Don’t try to byheart the commands to install

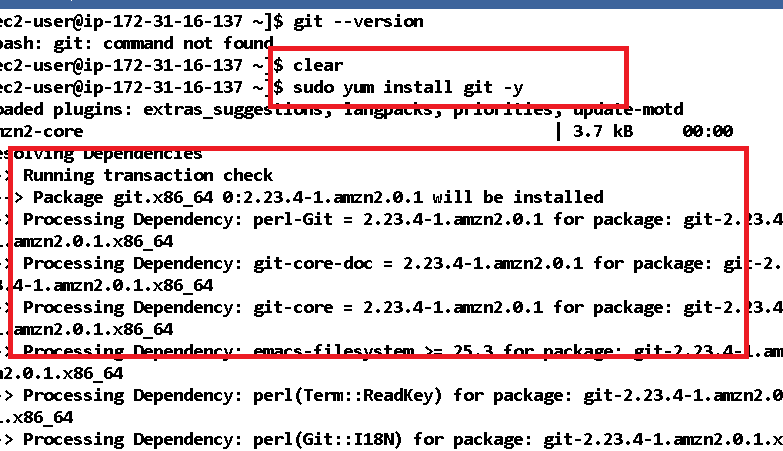
To install the softwares in the linux machines we use

>> sudo yum install software-name -y

Running a simple hello world program in the virtual machine

Command to install git

>> sudo yum install git -y

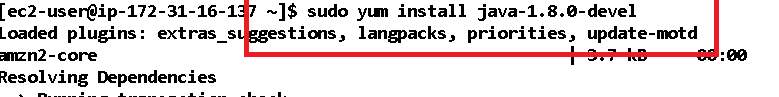


You can observe within seconds installation completes



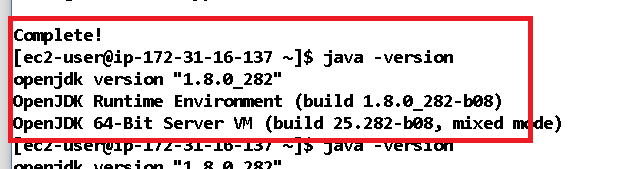
Installing java

>> sudo yum install java-1.8.0-devel



Verifying the java

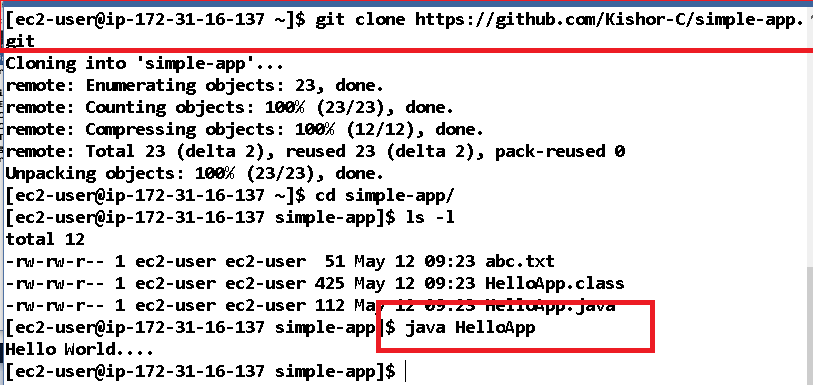
>> java -version



Now you can run the java programs/jar files

Pull the application using GIT

>> git clone url



You can also install the web server, node.js and so on  
Node.js Installation

You can refer the below url

<https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/setting-up-node-on-ec2-instance.html>

Install node version manager (nvm) by typing the following at the command line.

curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.34.0/install.sh | bash

Activate nvm by typing the following at the command line.

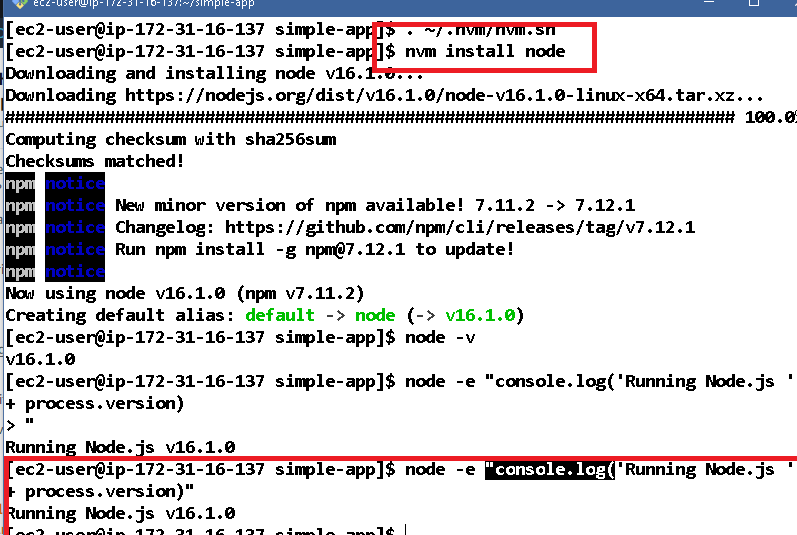
. ~/.nvm/nvm.sh

Use nvm to install the latest version of Node.js by typing the following at the command line.

nvm install node

Test that Node.js is installed and running correctly by typing the following at the command line

node -e "console.log('Running Node.js ' + process.version)"



Installing an apache web server

Reference:

<https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_Tutorials.WebServerDB.CreateWebServer.html>

>> sudo yum update -y

>> sudo amazon-linux-extras install -y lamp-mariadb10.2-php7.2 php7.2

>> cat /etc/system-release

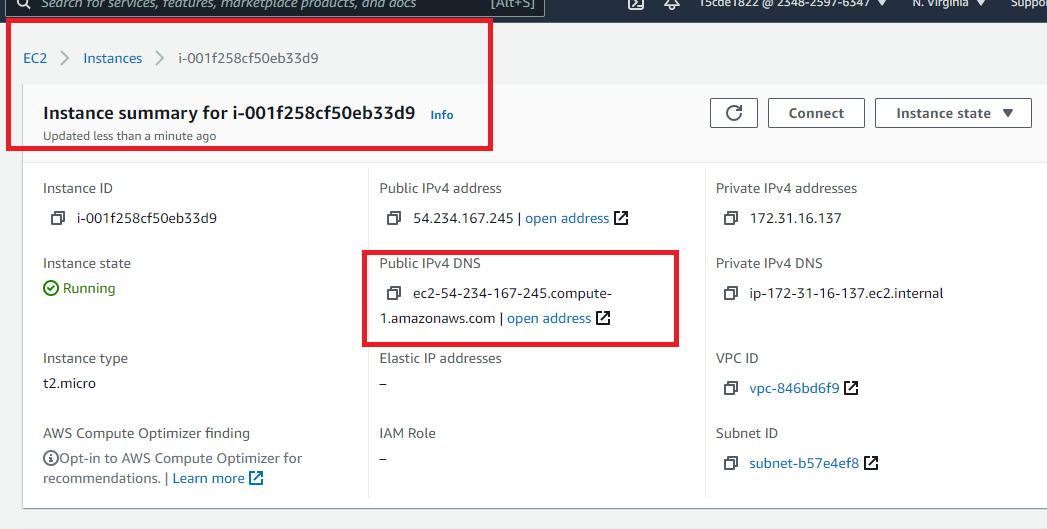
>> sudo yum install -y httpd

>> sudo systemctl enable httpd

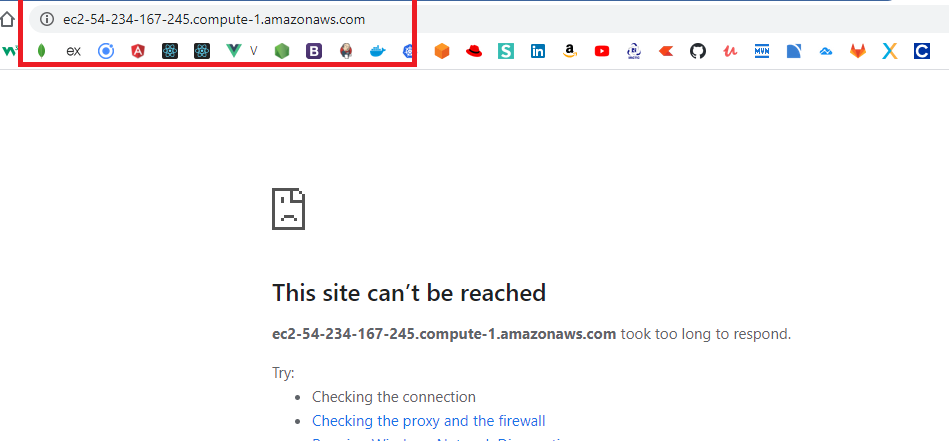
Once the server starts you can see a test page from the public address

Though the webserver is running you may not able see the server page when you enter the ip address of the virtual machine from your browser, this happens if your virtual machine security group doesn’t have inbound traffic.

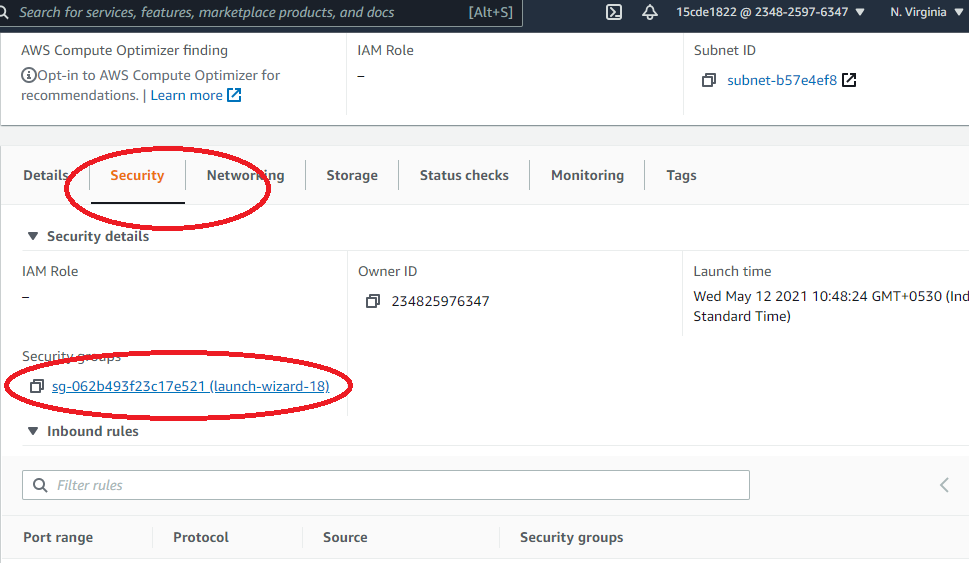
Below page is showing the public IPv4 DNS but when we enter that address we may not see the server page



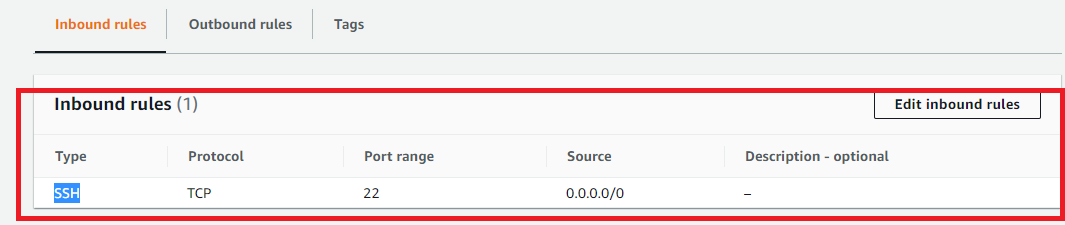
We are not seeing the server page



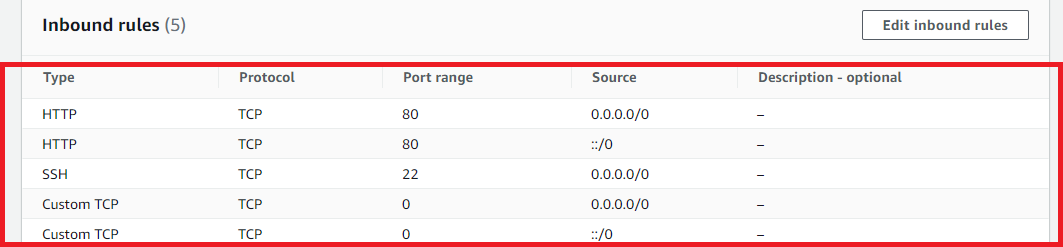
We need to change the security group



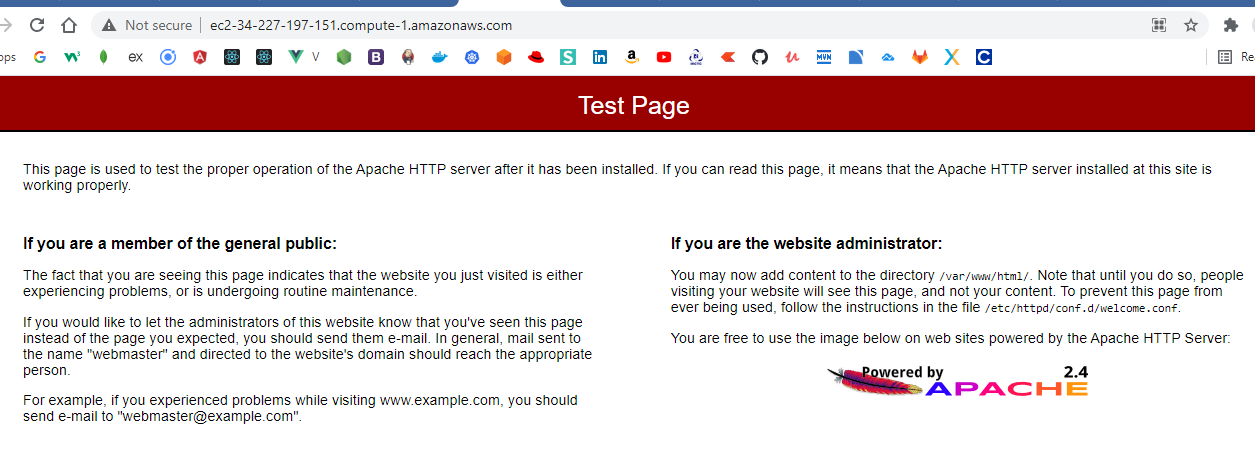
You can observe the instance is not having TCP port open to accept the incoming traffic



You can add inbound rules for TCP & HTTP



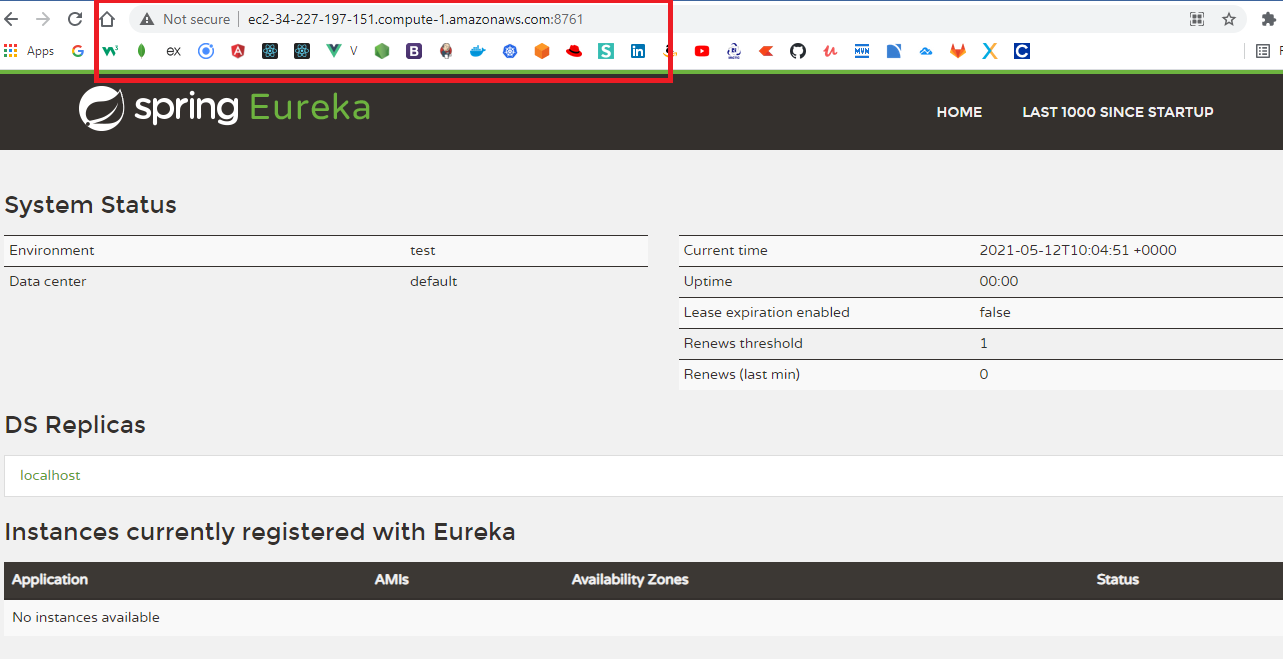
Now you can see the server page from your browser



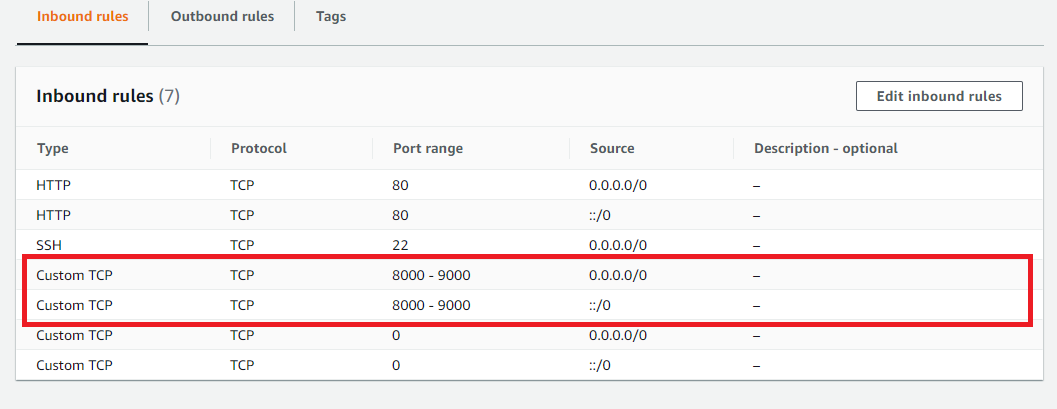
Running the microservice program by pulling the executable jar

Here we are pulling service-discovery.jar file

>> java -jar service-discovery.jar



To accept incoming request we need to open the port as below:



Docker:

Docker allows you to package all the codes, dependencies, libraries softwares to run the application in the container, it helps in hosting the application in any platform without installing the softwares, downloading the libraries, because Docker will have the image which will know all the necessity to host the application.

Earlier:

We installed JDK

We executed Jar file

The jar file will have all dependencies

With Docker:

You should only run the image: The image will have instruction about all the libraries, softwares required to run the application even the command to run the jar file

These images can be stored in docker hub its like GIT but docker hub stores the images created by developers, if you want to run any application, you need to pull that image from the docker and run the image, however the machine must have the docker engine which can run the image.

From the docker hub you need to pull the image

Firstly you must have the docker account then only you can pull the image

Secondly you need to install the docker

Command to install the docker is

>> sudo amazon-linux-extras install docker

>> docker --version



Commands used in docker

>> docker images

The above commands shows all the docker images downloaded

>> docker pull image\_name

The above command will pull the docker image from the repository

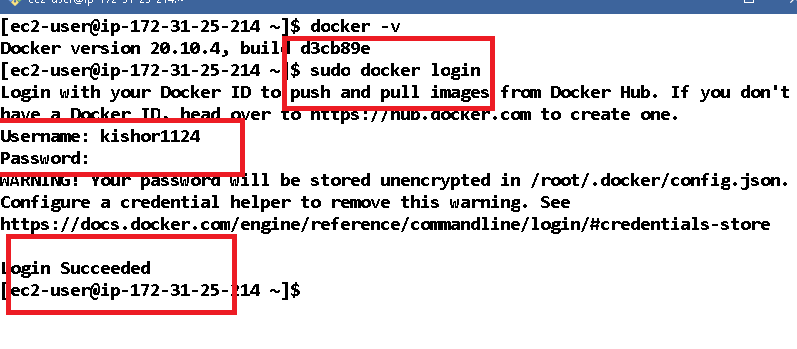
>> docker run -p port image\_name

The above command will run the image in the container

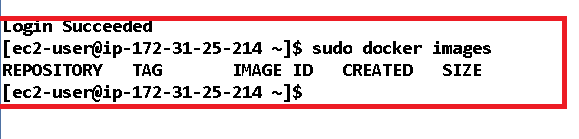
>> docker login

The above command allows you to login to the docker hub, after login only you can pull the images from the repository

First step is to install the docker & login to the account



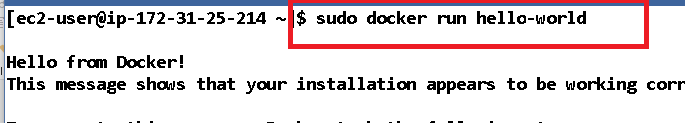
Listing the downloaded images from the respective repository



Download the c program image present in the docker hub

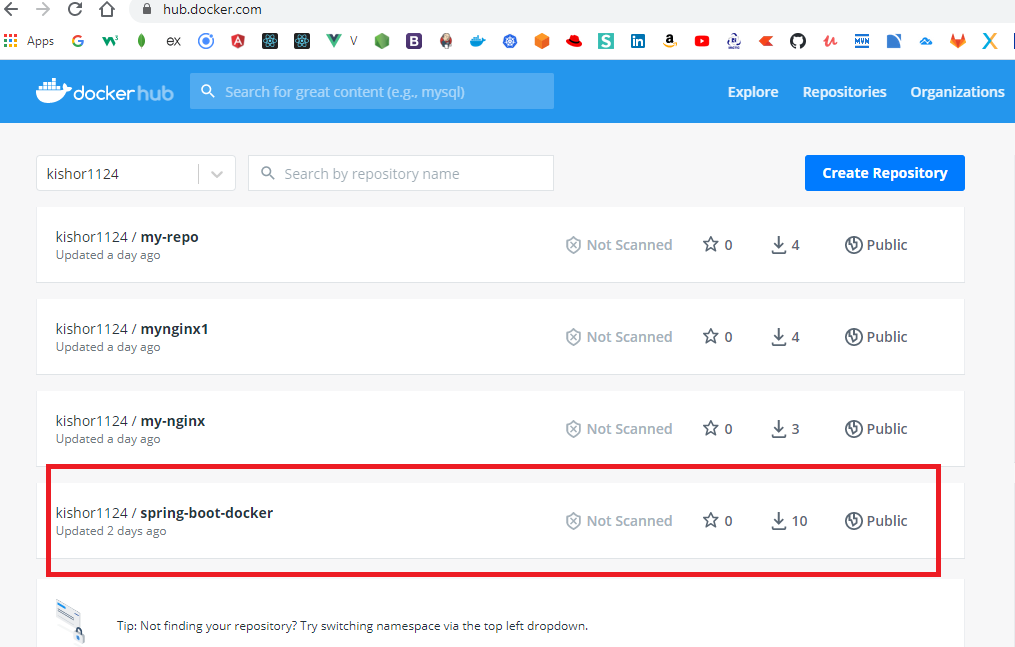


To run this image you can use sudo docker run hello-world, it displays an hello world message, without installing any c program software we can run this.



Now there’s a container which is running this hello-world program, to know the containers running we can use sudo docker ps

There’s an image spring-boot-docker in my repository

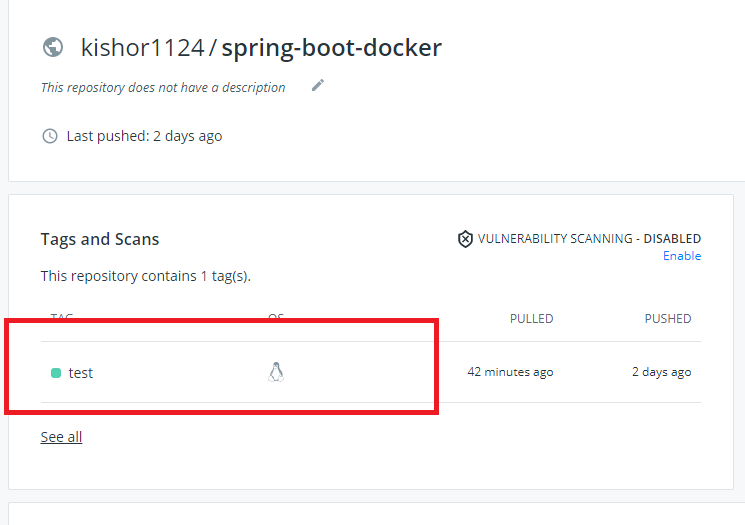


How to pull this image

>> sudo docker pull kishor1124/spring-boot-docker:tagName

Note: tag name is test so you will use

>> sudo docker pull kishor1124/spring-boot-docker:test

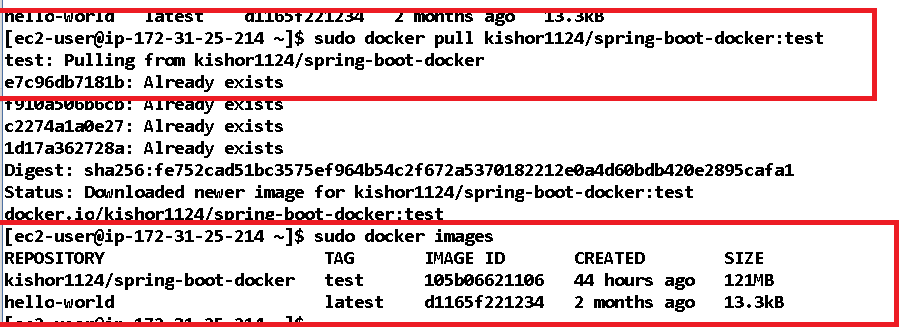


What if you forget the tag name, then docker uses a default tag name called ‘latest’

i.e., if you use sudo docker pull kishor1124/spring-boot-docker without test then docker looks for kishor1124/spring-boot-docker:latest

Since there’s not tag with the name latest, you need to explicitly use test

i.e., sudo docker pull kishor1124/spring-boot-docker:test



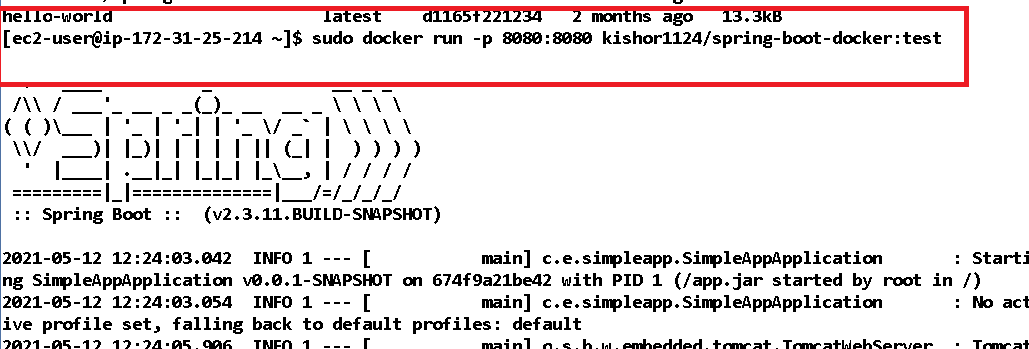
The image already has the instruction to download JDK1.8, downloading the jar file and run the jar file

Note: It means you don’t have to install JDK and download necessary libraries

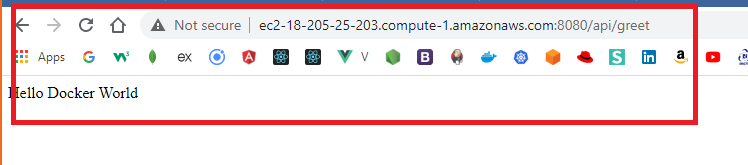
How to run the image

>> sudo docker run -p 8080:8080 kishor1124/spring-boot-docker:test

The above images runs the application in 8080 port and it has a webservice in /api/greet



Now the application is running in 8080 port you can use the EC2 instance IP address and access from any machine.



Docker commands:

1. Docker Hub account

2. login to the docker hub using

sudo docker login

<<Enter username>>

<<Enter password>>

3. Pull the docker image

sudo docker pull image-name

sudo docker pull hello-world

sudo docker pull kishor1124/spring-boot-docker:test

4. View the docker images

sudo docker images

5. Running the docker image

sudo docker run image-name (or) sudo docker run -p port:port image-name

sudo docker run hello-world

sudo docker run -p 8080:8080 kishor1124/spring-boot-docker:test

6. Stopping the docker container

sudo docker stop container-id