Amazon Web Services

AWS provides on demand cloud computing platforms and API’s to individuals, companies and governments on a metered pay as you go basis.

AWS is a platform of webservices offering solutions for computing, storing & networking that work well together

Computing: Computer which is the basic necessity to do anything

* Provides OS like Linux, Windows, Ubuntu and so on.
* Provides RAM, Processor

Storage: When you have a computer you need storage

* Volumes
* Databases

Network: You get a computer with internet so that you can install any softwares on those computers like Java, tomcat, python, git and so on

AWS saves cost: You can easily switch your platforms or environments in less cost i.e., old platforms you may not need can be terminated & then no more bills on them.

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 also then old platform will be waste of money after some years

Any use case can be implemented in AWS, whether it’s a widely used web application or a specialized enterprise application

Different way to interact with AWS:

You can interact with AWS in many different ways

* Web based UI
* Programmatically
* Command line
* Blueprints to setup, modify, or delete your infrastructure on AWS

Pay per use: Compute, storage and networking services are billed similarly to electricity

Companies can scale their servers when the demand increases and reduce it when the demand decreases AWS helps you to rent/purchase the servers for lease with discounts.

AWS data-centres located in many country regions so that request & response time will be less and also data will be secured when information’s don’t go outside the country.

AWS Account

In order to use the AWS services you need to create an account, you must have either credit card/debit card for verification, it deducts some amount as of now 2rs and credits back after 2 or 3 business days, this is just to verify you have a bank account,

Bill will be generated when your Free-tier limit is over or when you use the paid services which are billed when you use, you can see the bill summary or get alerts when the bill is exceeding your budgets

AWS Account users

1. Root user
2. IAM (Identity Access Management) user

Root User: Account owner, created when AWS account is created

IAM User: Created by the root user or IAM administrator of the account

Root user credentials:

* Account owner will have full access to all the resources
* Only root user can close your account

IAM credentials:

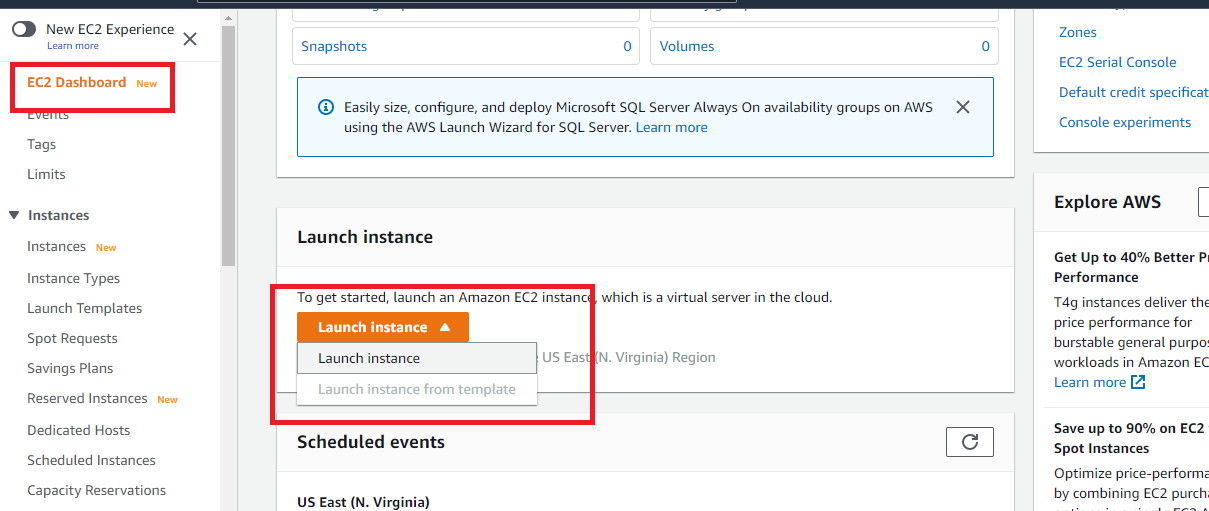
* You can control access to the AWS services and resources
* You can grant administration level permissions to IAM user

Multifactor Authentication (MFA)

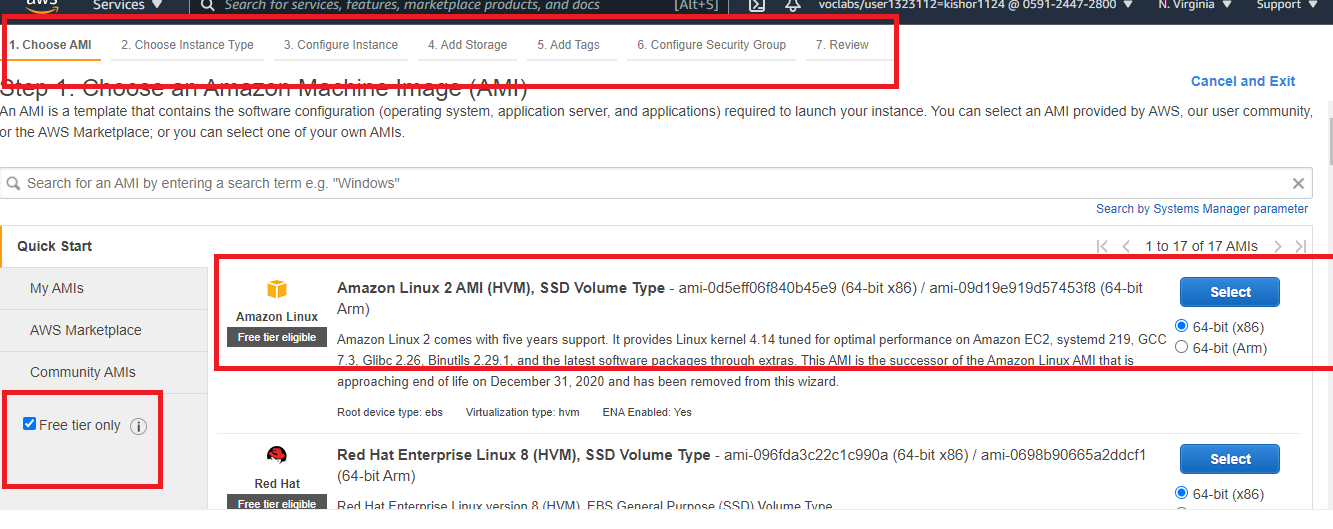
MFA provides an extra level of security, With MFA enabled, when you try to login with username & password you are prompted to enter authentication code from an MFA device, adding MFA provides increased security for your AWS account settings and resources.

EC2

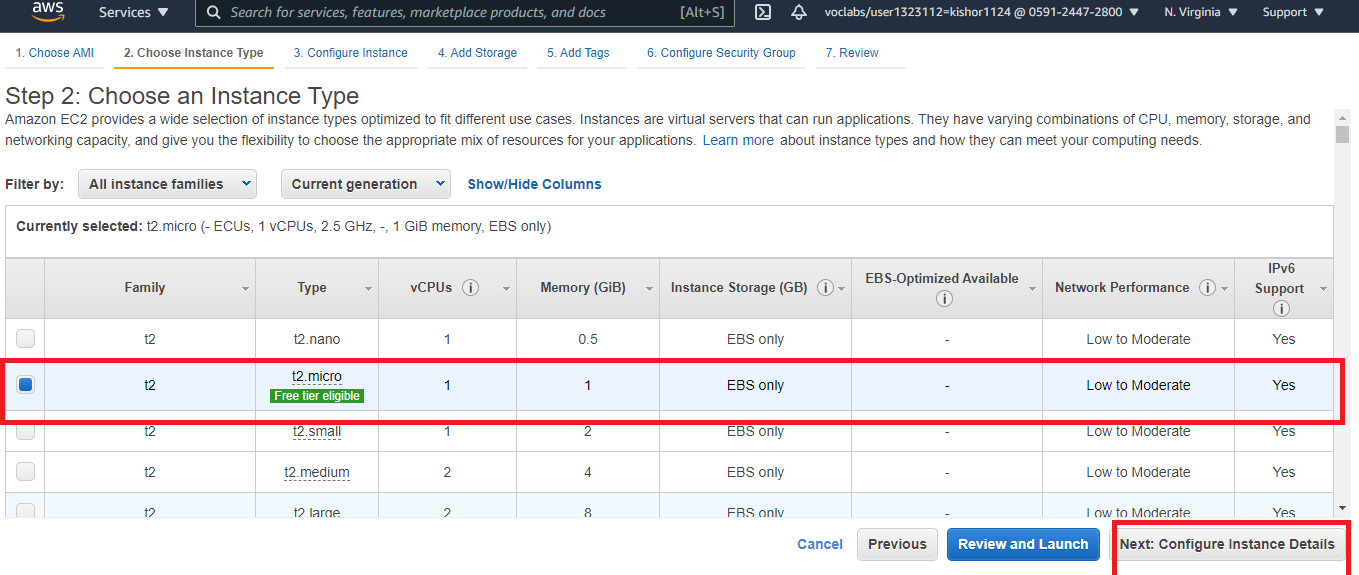
Elastic Computing Cloud: It is to create virtual machine.



Select free tier

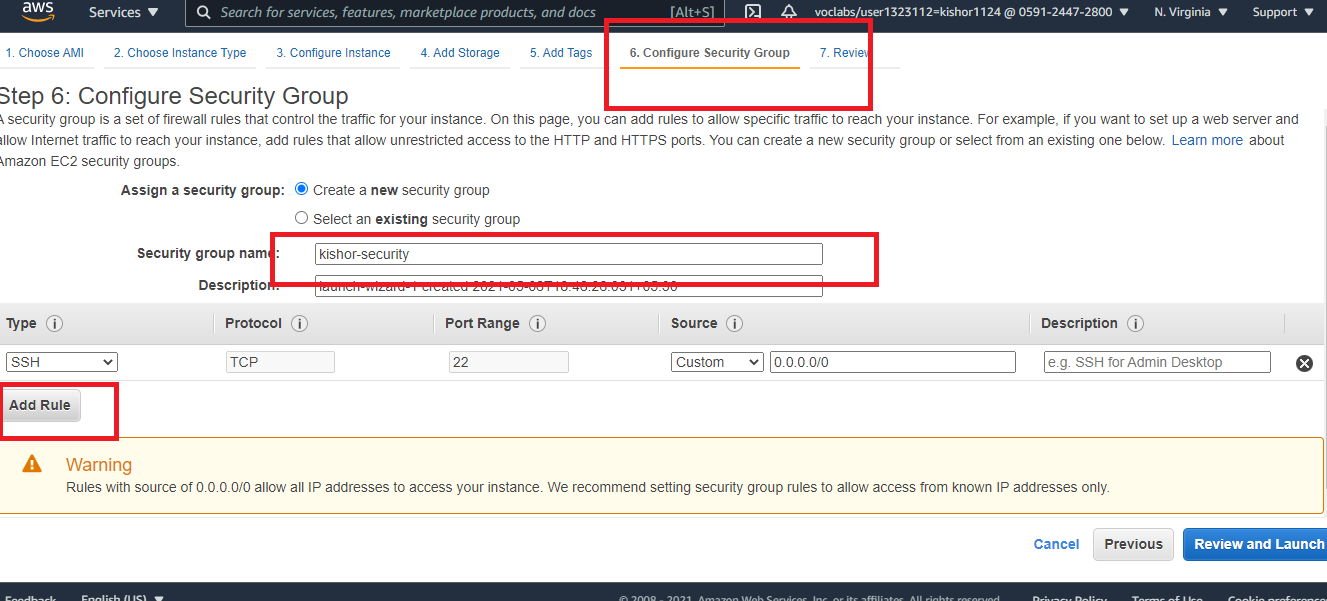


After click on select , you get the instance type

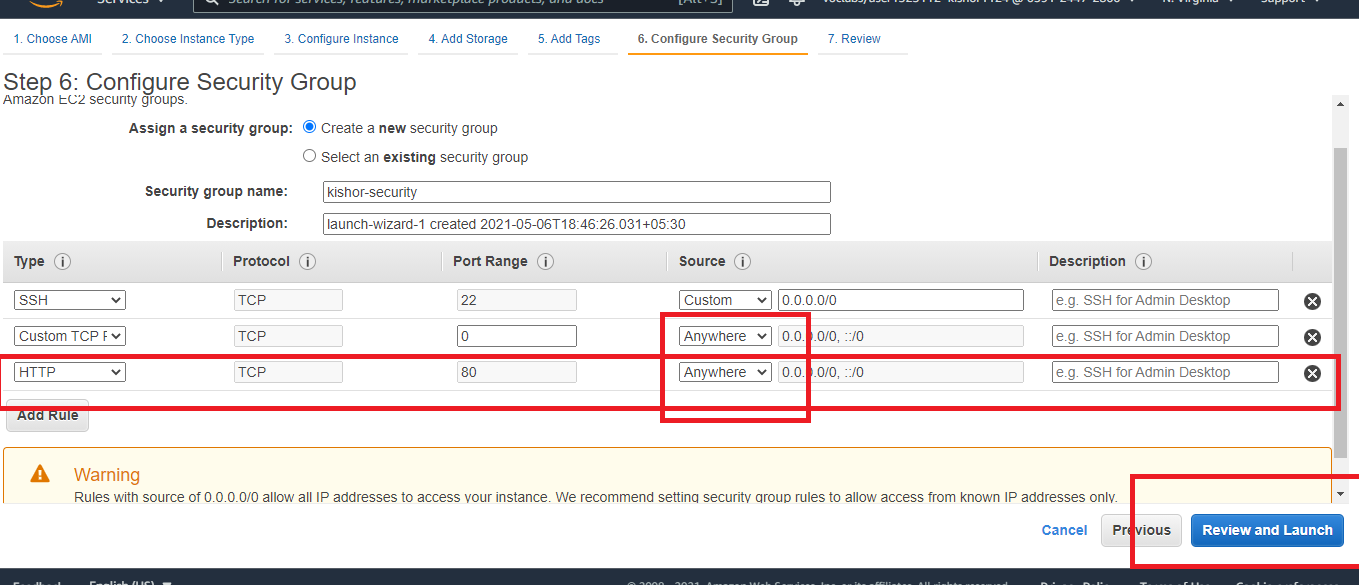


Goto configure security groups

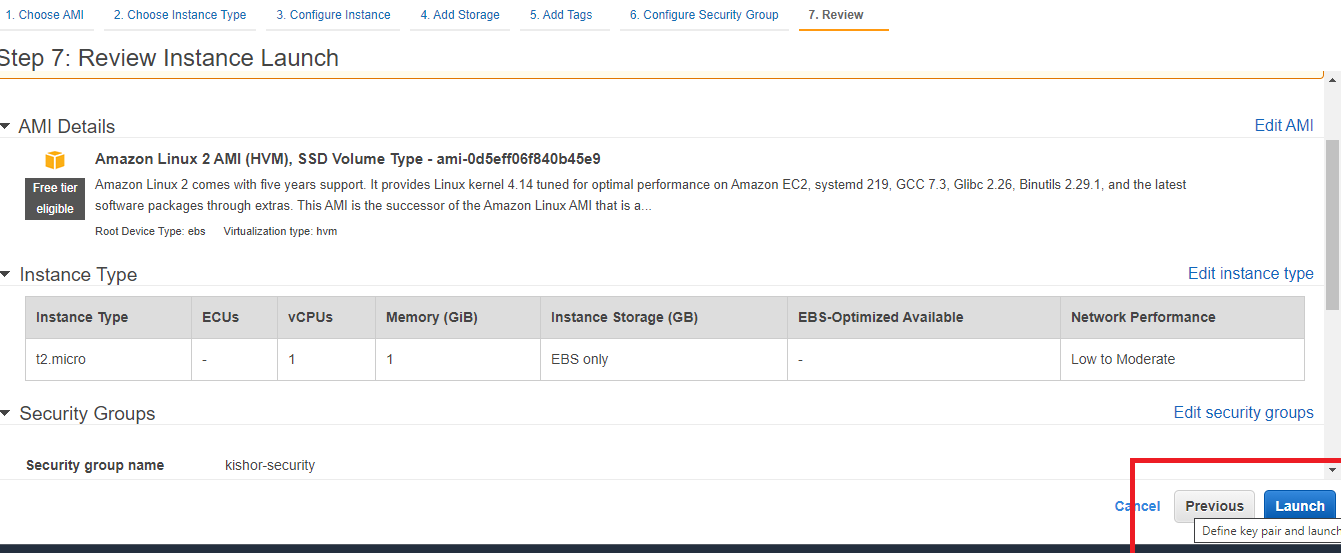
Since we need to access it publicly we will add rules



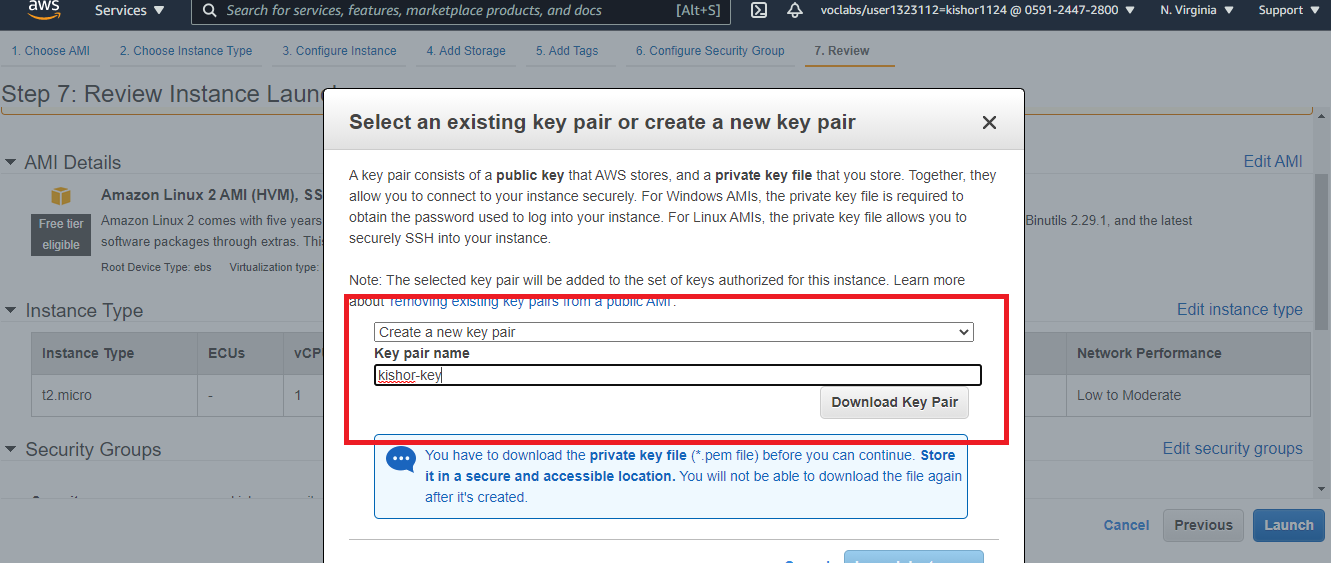
Add custom tcp & http and in source choose anywhere and click on review launch



You will get below window to review and launch, there you will get keypairs

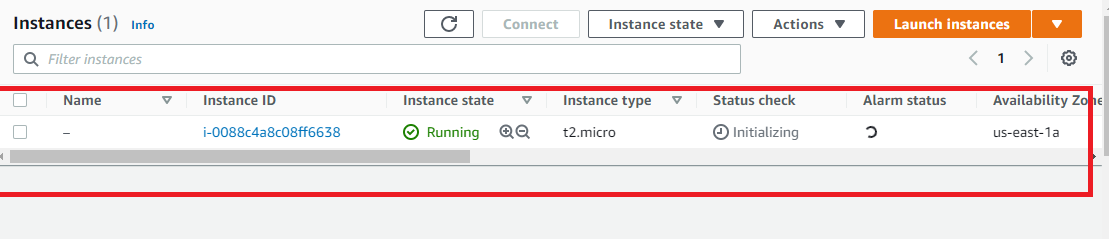


Create a new key pair and download



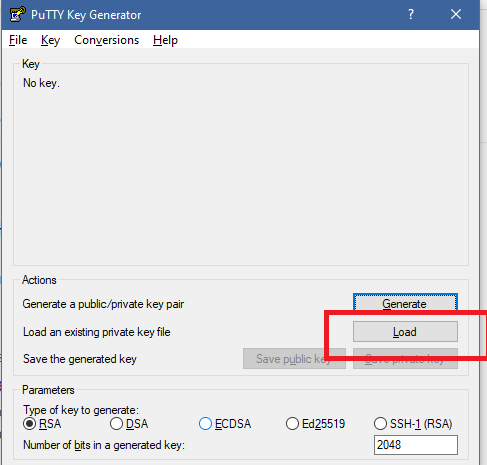
You can notice one pem ( Privacy Enhanced Mail )file will be dowlnoaded, which windows can’t understand so we need to use either Putty or gitbash.

You can click on the launch instance and when you view instance you see the instance as below

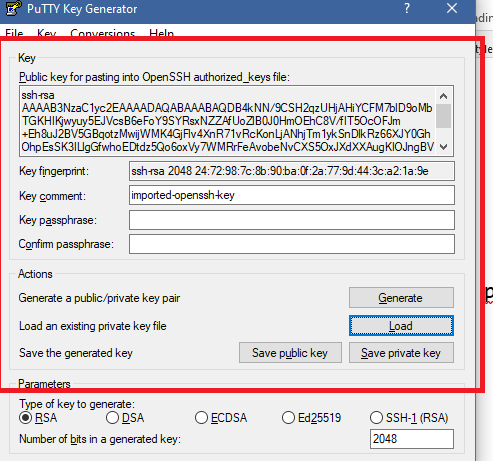


First step using putty to connect to EC2

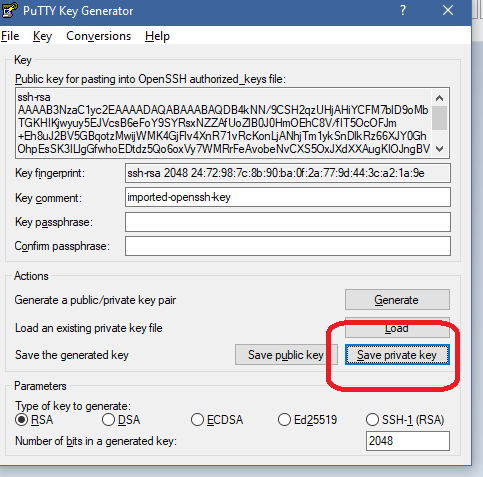
* Download the putty software
* Open the putty key generator, which helps to generate private key from the PEM file



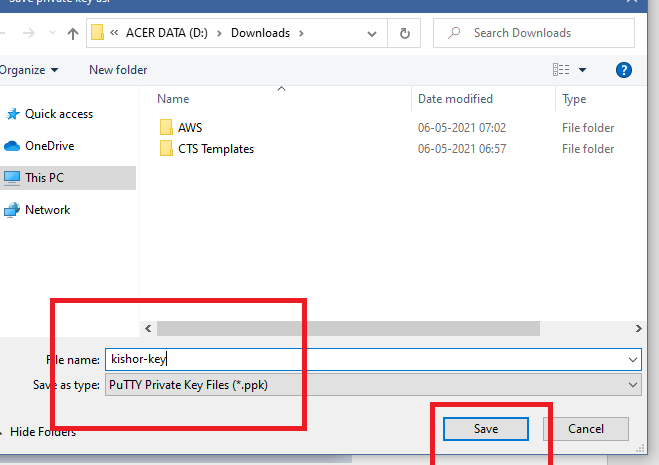
Click on load and select the downloaded key pairs pem file.



You need to save private key and save the file with ppk extension.

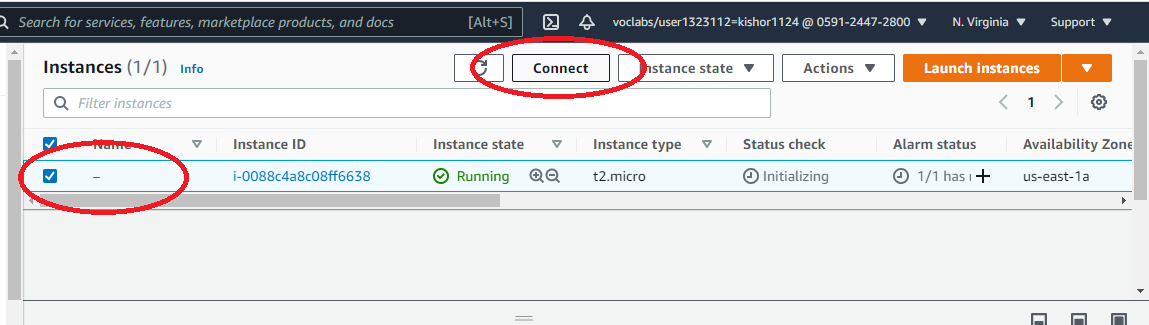


You will get a window click on ok and save with .ppk extension.,

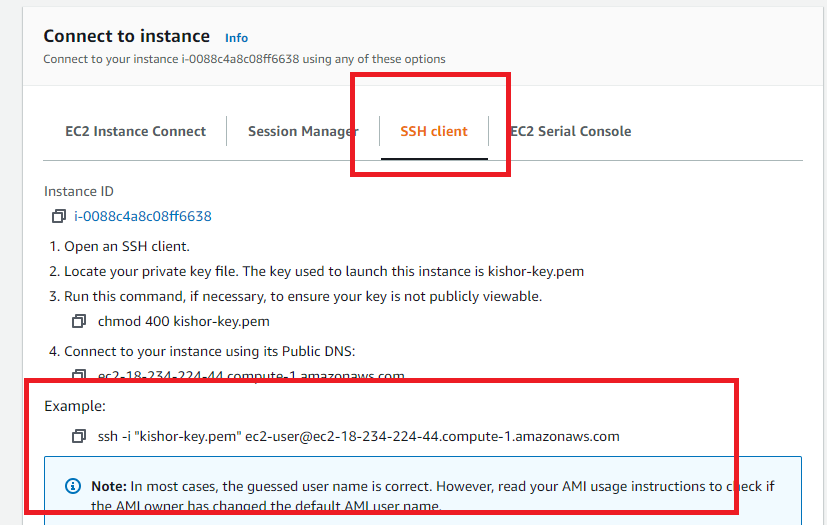


you can close the putty generator,

Goto the instance and select the instance & click connect



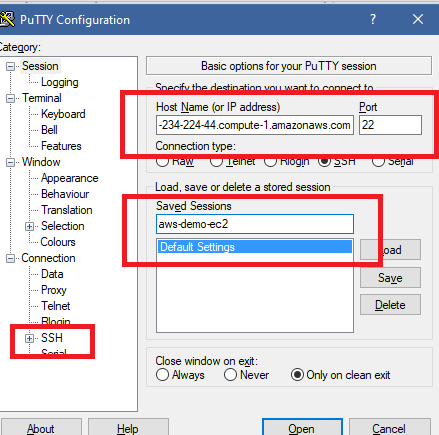
Once you connect a window opens you can select SSH client as below, where you will get the command to connect to the instance from the putty



Note: You need ec2-user@ec2-18-234-224-44.compute-1.amazonaws.com.....

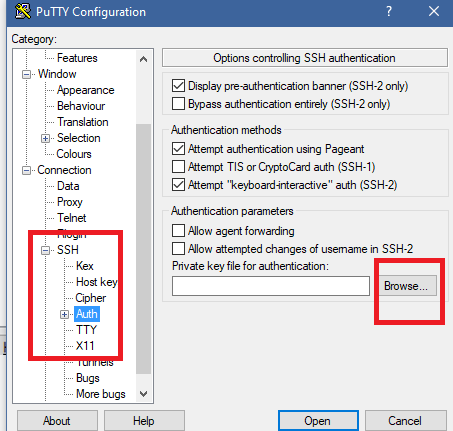
Now open the putty

Enter the hostname and save session with some name and select SSH

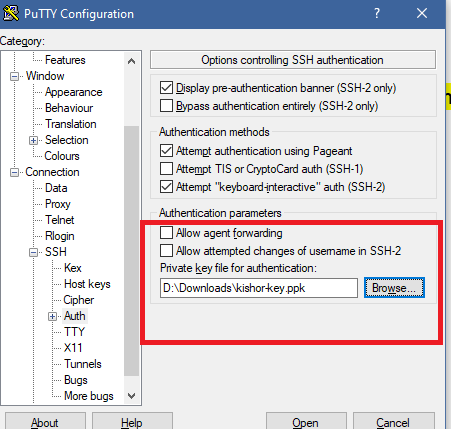


Note: save the session so that next time you can use without configuring from scratch

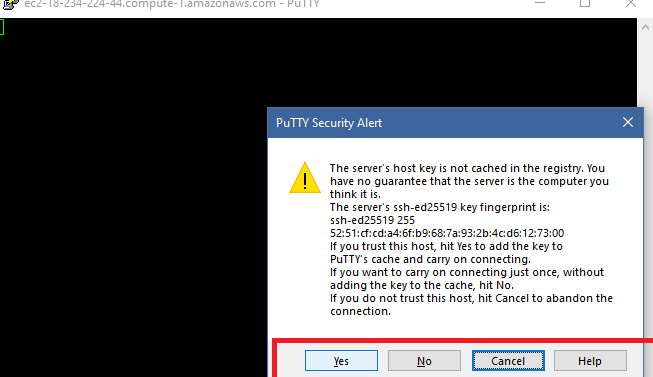
Expand SSH and select Auth, browse and select the ppk file



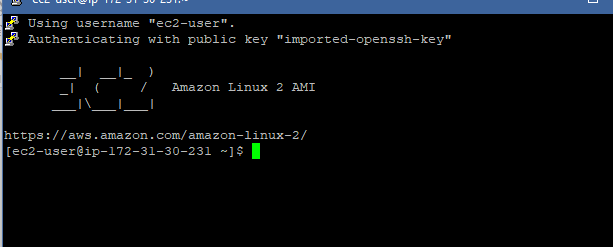
After you select ppk file



Once you click open you will get a security alert you say yes

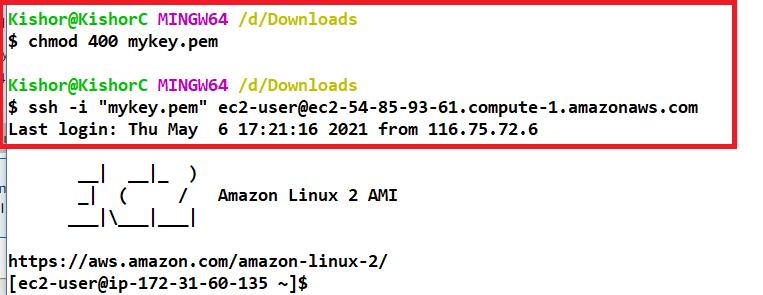


Now you can see the instances connected through putty

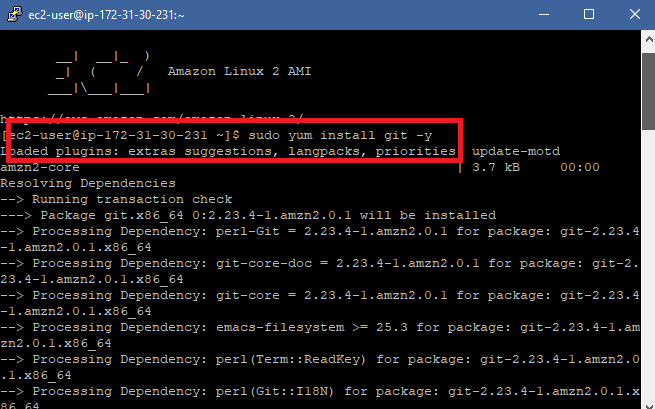


Now we are inside the instance we can install any softwares like java, git and so on

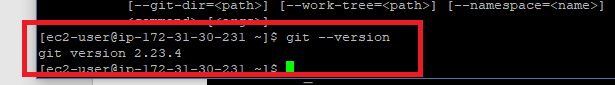
You can also connect to the instance using Git bash without using putty as



How to install GIT



Verifying the GIT

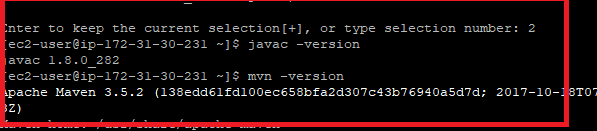


How to install Java

sudo yum install java-1.8.0-devel

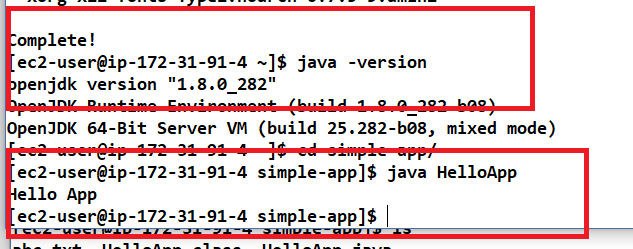
sudo /usr/sbin/alternatives --config java

sudo /usr/sbin/alternatives --config javac

Verify using javac -version  


Clone a repository from git and run a simple java program





Exercise: Try to create a spring boot application & pull from the GIT and run in the EC2.

Installing maven in EC2

1. Connect to your Amazon EC2 instance with an SSH client.
2. Install Apache Maven on your EC2 instance. First, enter the following to add a repository with a Maven package.

sudo wget https://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo -O /etc/yum.repos.d/epel-apache-maven.repo

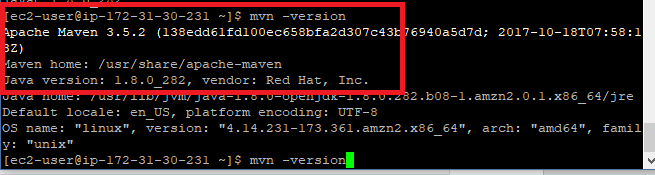
Enter the following to set the version number for the packages.

sudo sed -i s/\$releasever/6/g /etc/yum.repos.d/epel-apache-maven.repo

Then you can use **yum** to install Maven.

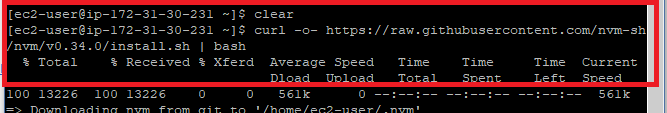
sudo yum install -y apache-maven

Verify the maven

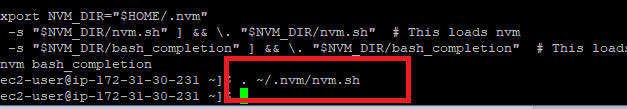


Installing the node.js

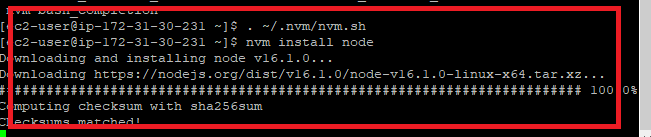
>> curl -o- <https://raw.githubusercontent.com/nvm-sh/nvm/v0.34.0/install.sh> | bash



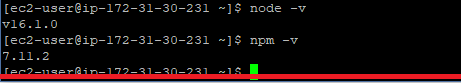
>> . ~/.nvm/nvm.sh



>> nvm install node



Now you can verify node & npm version



Installing http server

Ensure the inbound rules has http with port 80 configured.

Follow the steps given in the

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

>> sudo yum update -y

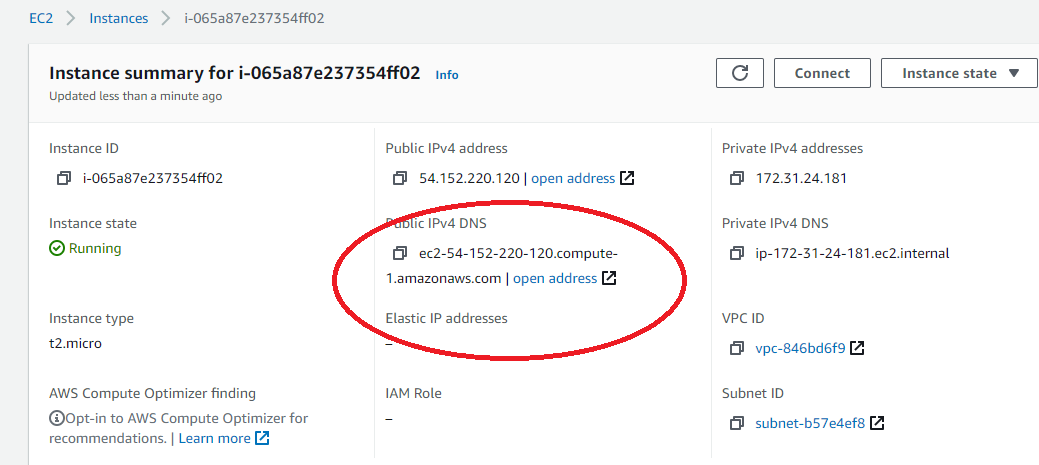
>> sudo amazon-linux-extras install -y lamp-mariadb10.2-php7.2 php7.2

>> cat /etc/system-release

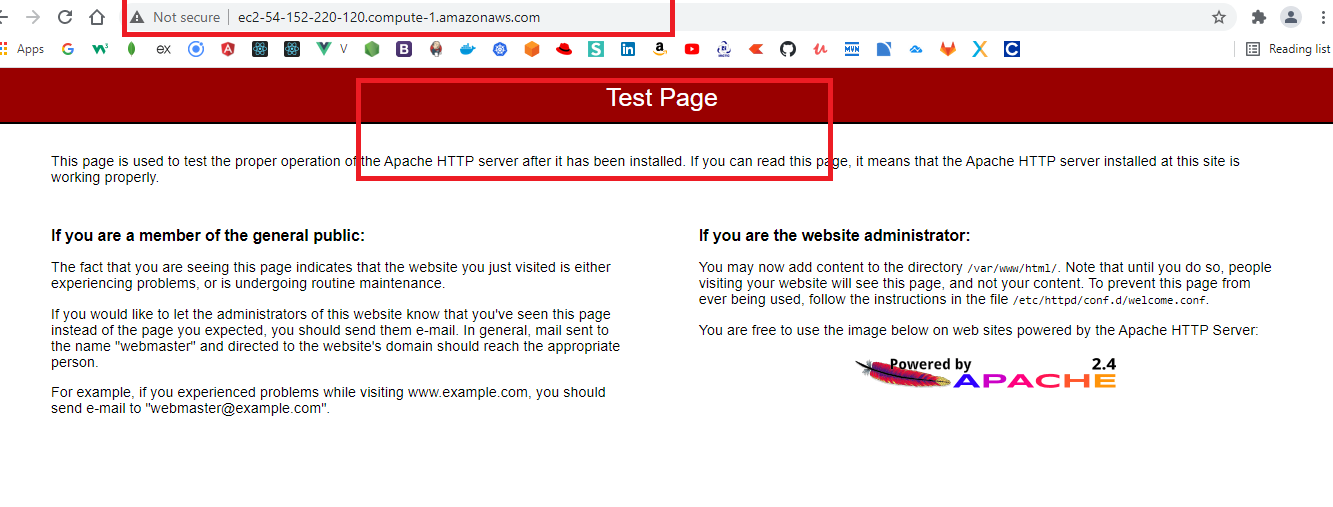
>> sudo yum install -y httpd

>> sudo systemctl start httpd

Check the server running by entering the public url from the browser.



You will see a test page as below:



Installing the docker

>> sudo amazon-linux-extras install docker

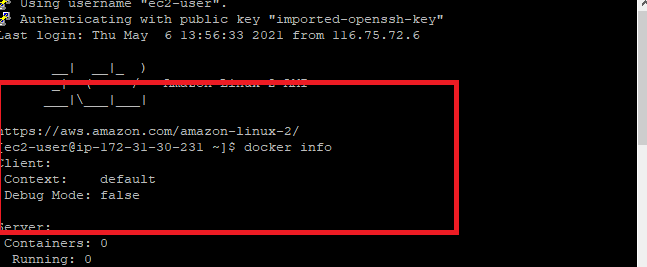
>> sudo yum install docker

>> sudo service docker start

>> sudo docker info

>> sudo usermod -a -G docker ec2-user

Above command enables you to enter the commands without sudo, to apply the settings you need re-login to the session from the putty.



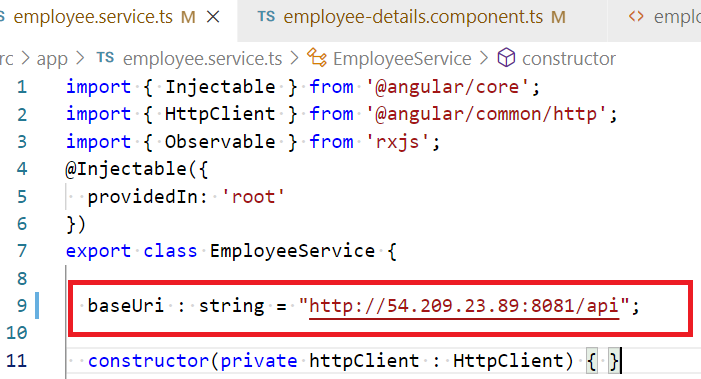
>> docker info

>> sudo docker images

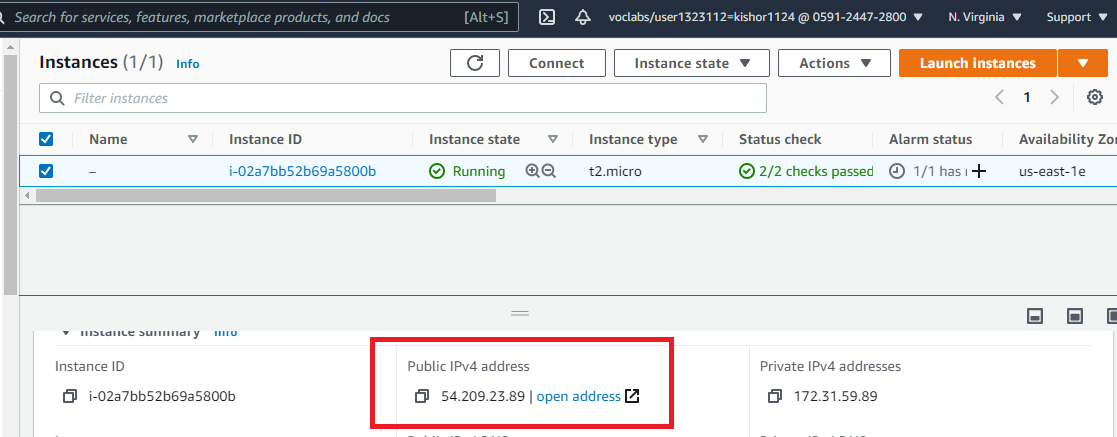
Deploying angular and spring boot application in aws

Step 1: Create angular project that interacts with spring boot rest api.

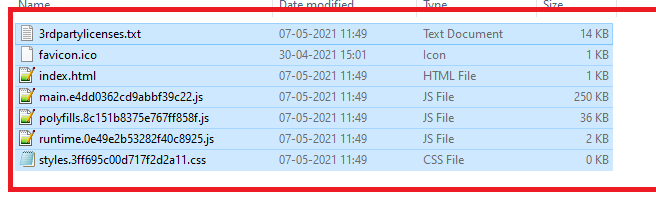
Step 2: Give the IP address of the ES2 instance in angular service which hosts the spring boot application.



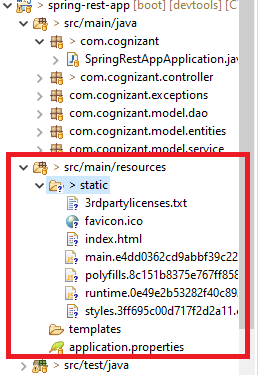
The above IP address is of EC2 instance



Step 3: Build the angular project using >> ng build --prod and copy the build files in to the static folder of spring boot project

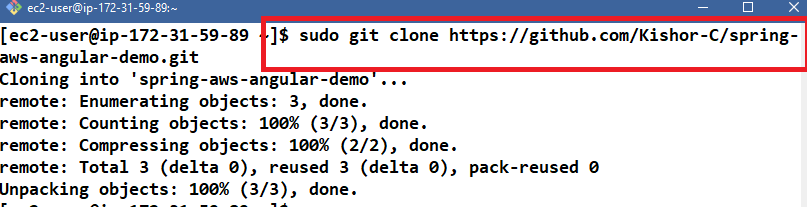


Spring boot project static folder

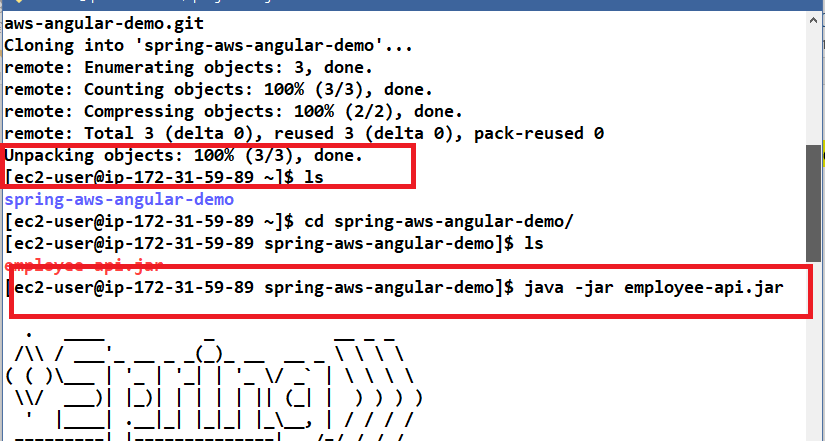


Now build the maven project ‘mvn package’ you will get the executable jar, that needs to be deployed on cloud,

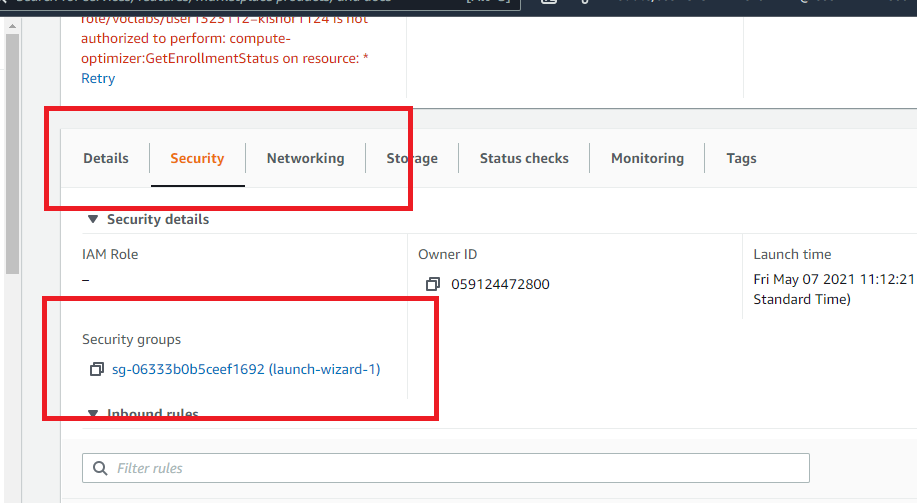
Steps: Use git and push the jar and pull the jar from the EC2.



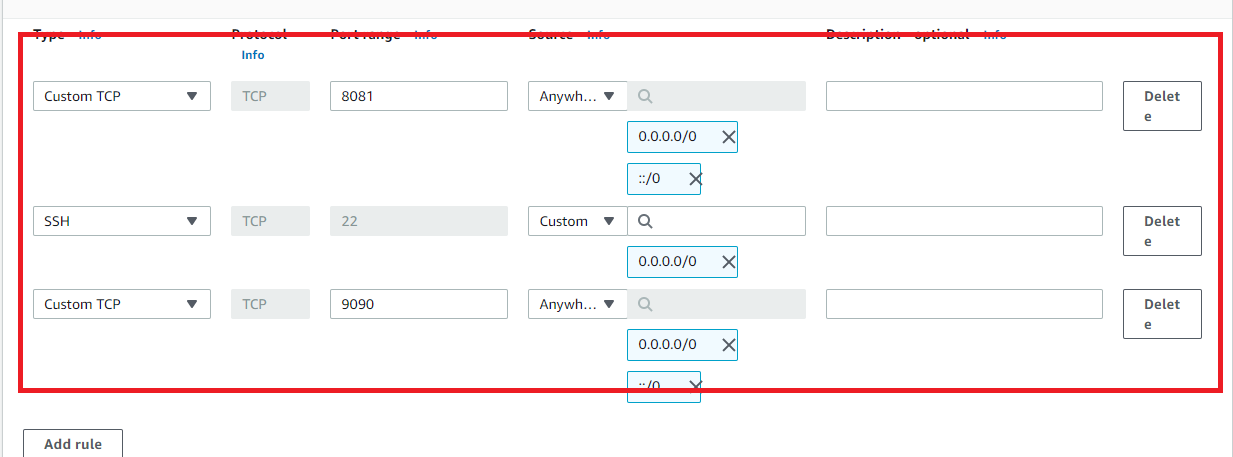
Now you can use ls to find the project downloaded from git and run the jar file using java -jar command



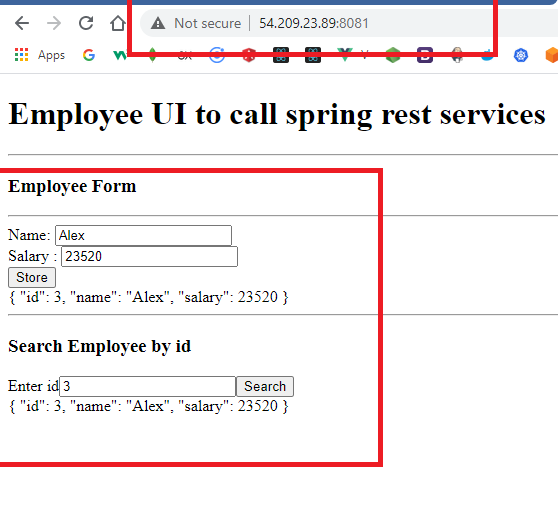
Once the application is run in a specific port, the EC2 instance TCP port should be open to accept the request, hence in the EC2 security group you need to change the security group in bound rules



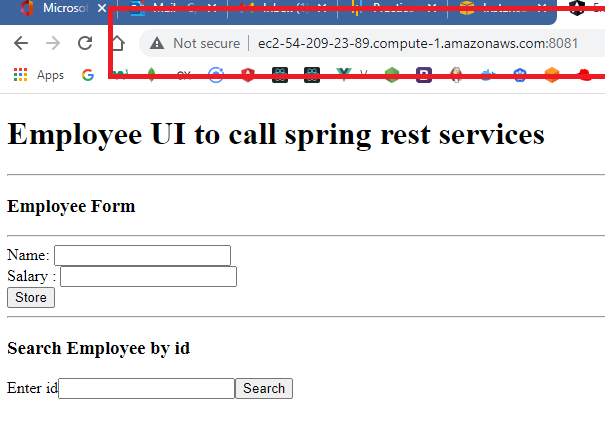
Edit inbound rules and save



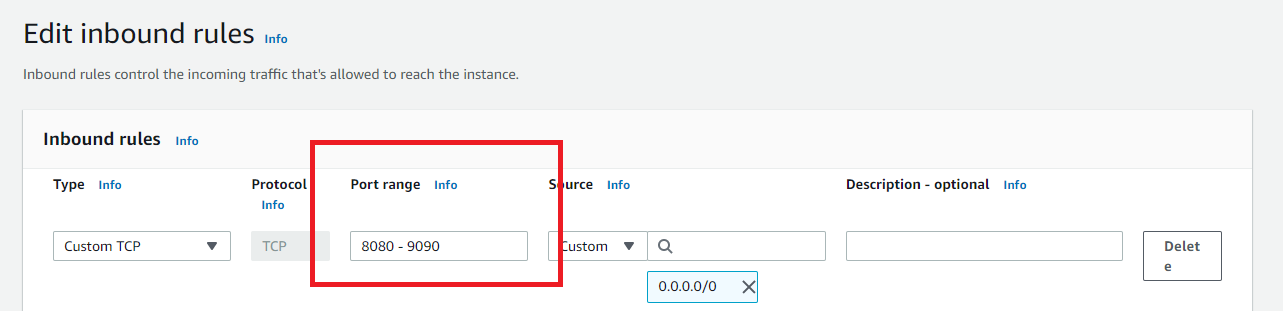
Now you can open the application from the browser



You can enter the machine name also instead of ip address



How to configure range of ports in inbound rules



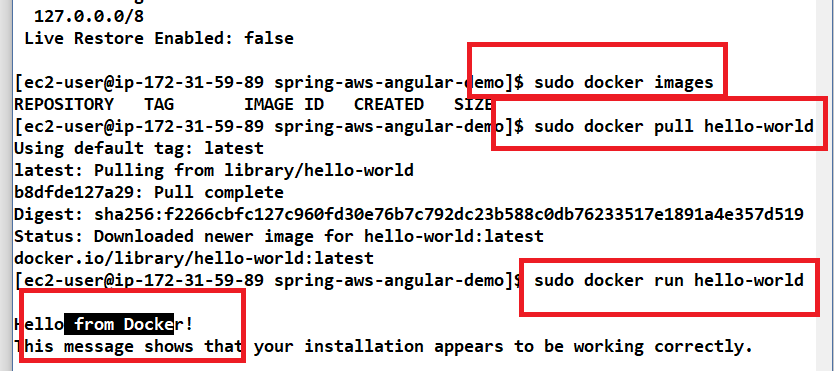
Now you can run the application in the port range 8080 to 9090

Working with docker

>> sudo docker images : this will display all the images

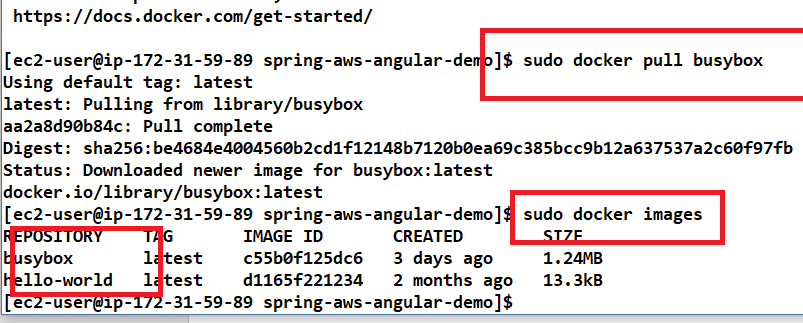
>> sudo docker pull hello-world

>> sudo docker run hello-world

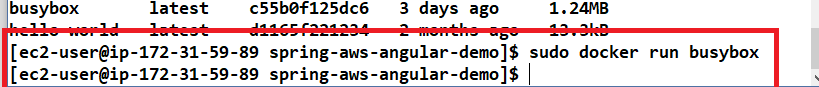


Note: The above program is written in c program

Busybox Image: It has unix utility

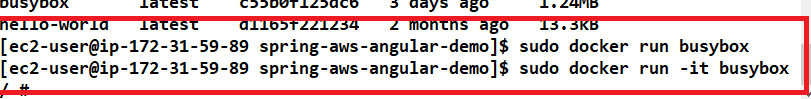


Busybox is the OS so if you run it will exit to the terminal



>> you need to run the busybox in iterative mode

>> sudo docker run -it busybox



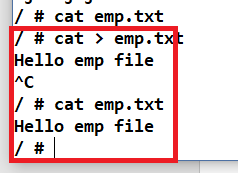
These utilities are helpful in the windows os if the user is used to unix commands

Try out the following commands:

>> ls

>> ls -la

>> mkdir

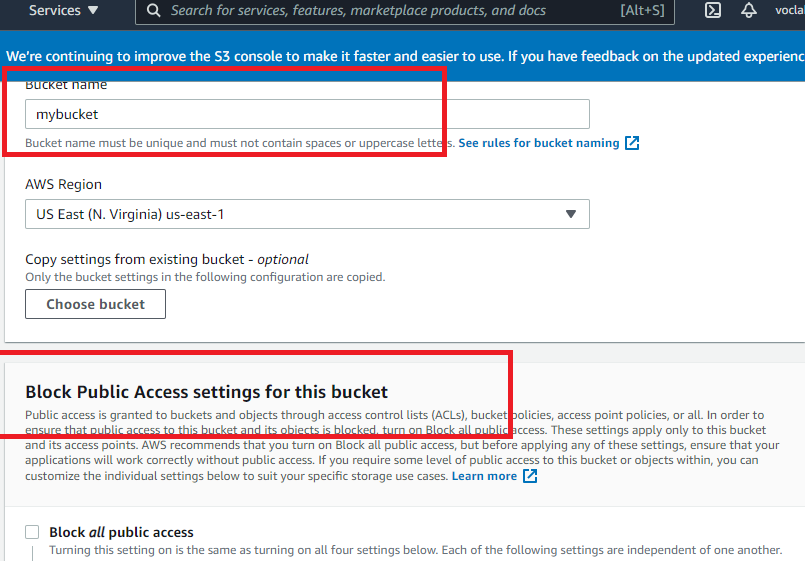


>> sudo docker info

It will give docker details

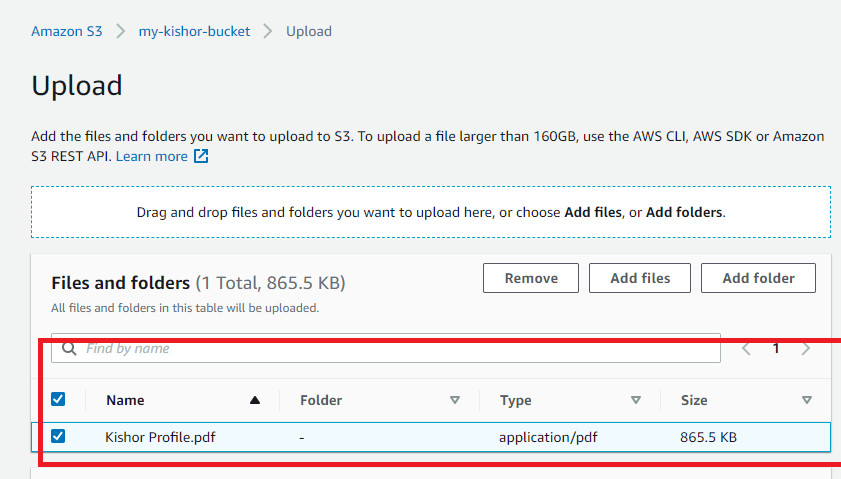
S3 Simple Storage Service:

Choose S3 and create bucket with some name, uncheck block public access

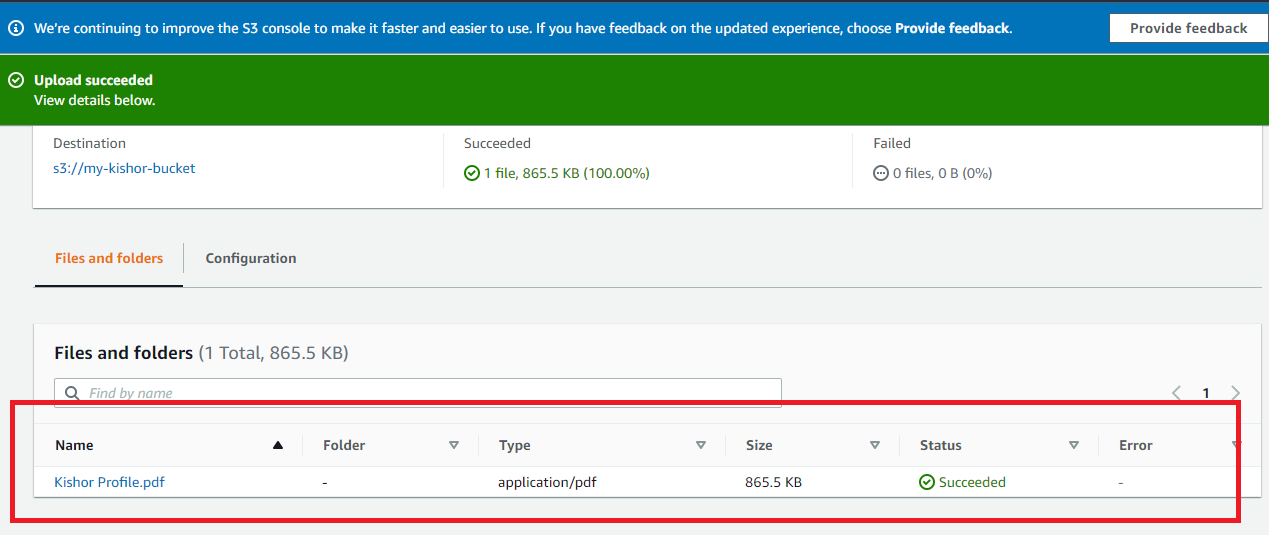


You can see the bucket created

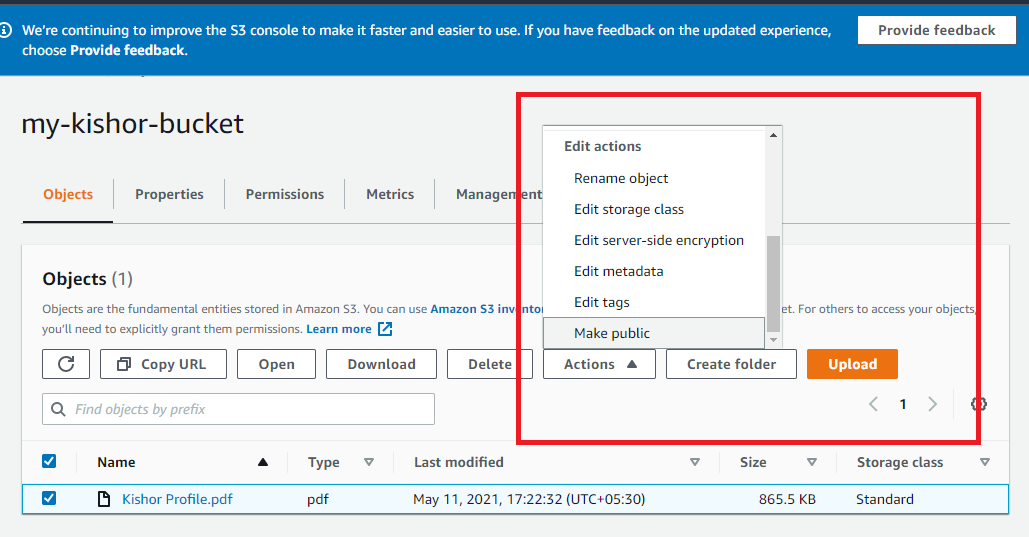
You can select the files and upload



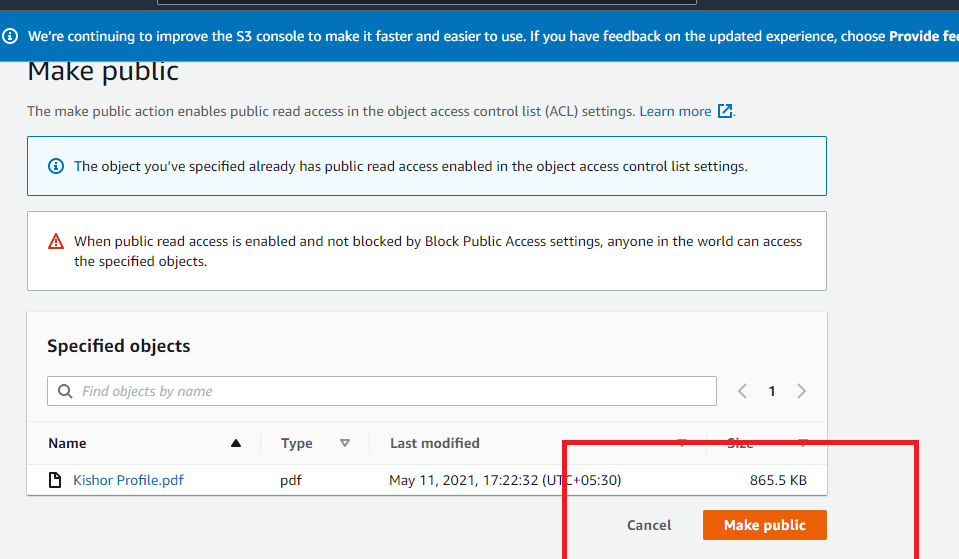
You can see the success message



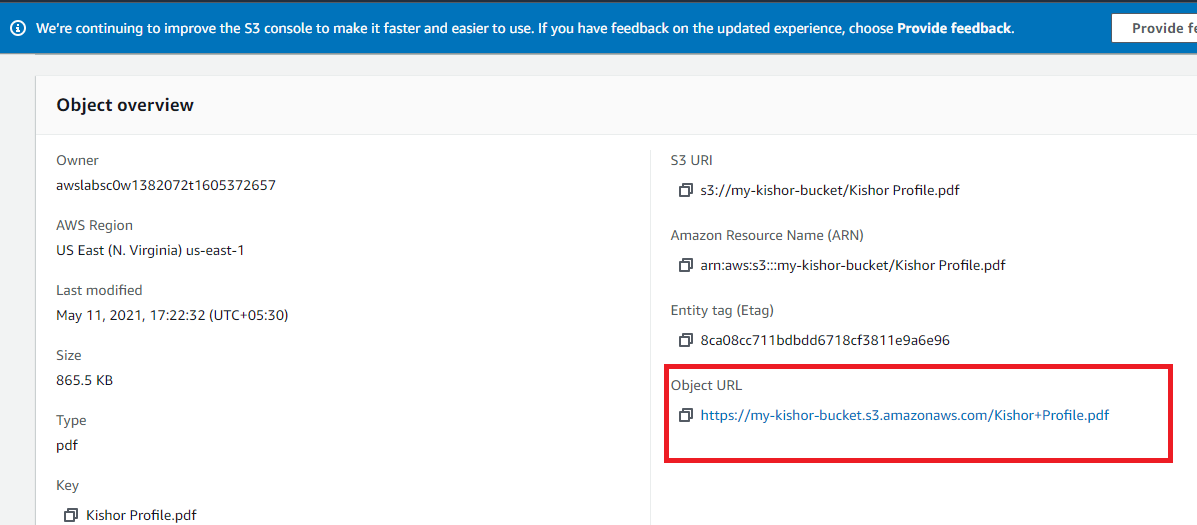
You need to make public



Select public



Click on the file you will get the below window that shows the link to access the file

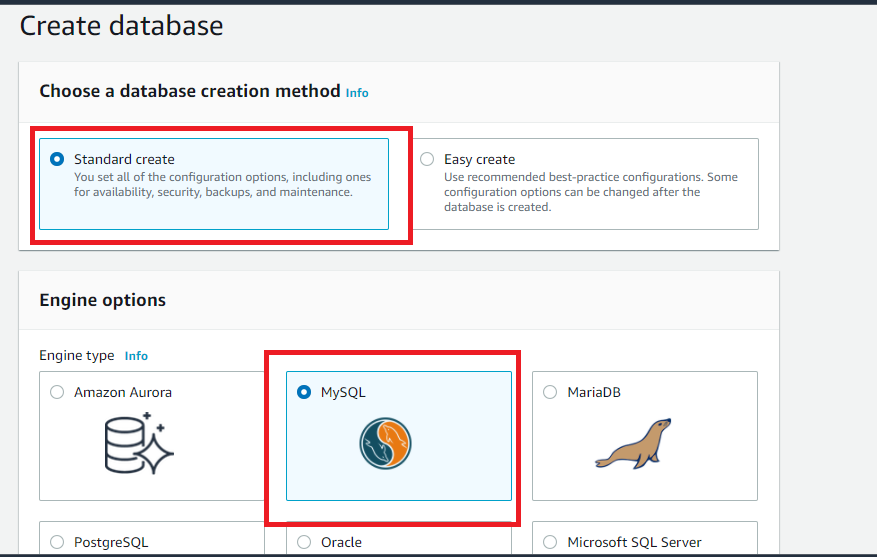


We can also add jar files in the bucket

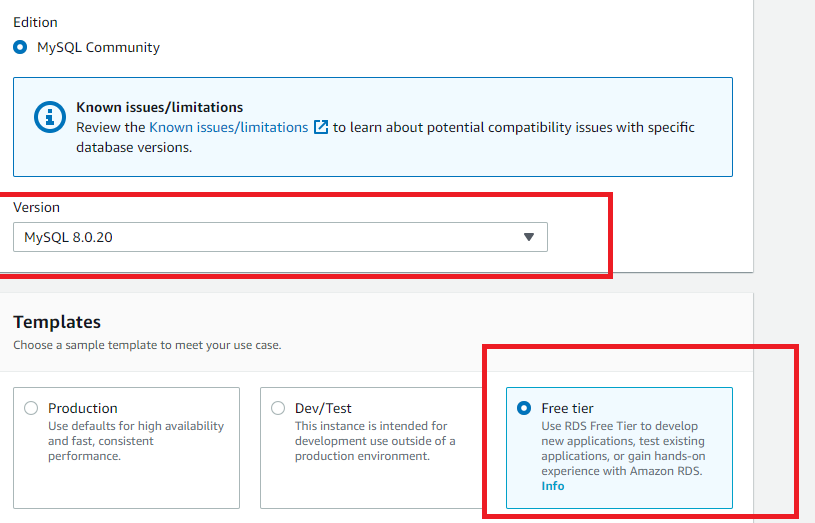
RDS (Relational Database Services)

RDS allows you to create database instances choosing Mysql, oracle, Aurora and etc.

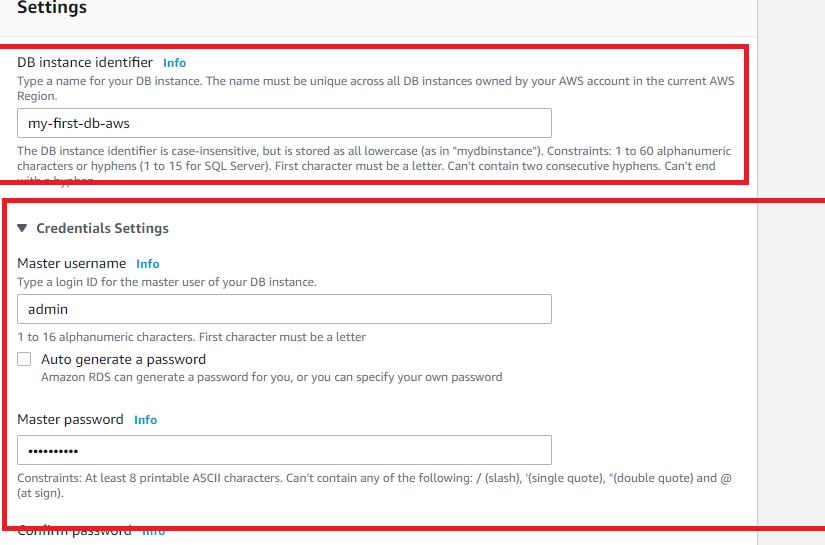
1. Select RDS from AWS services
2. Select Create Database



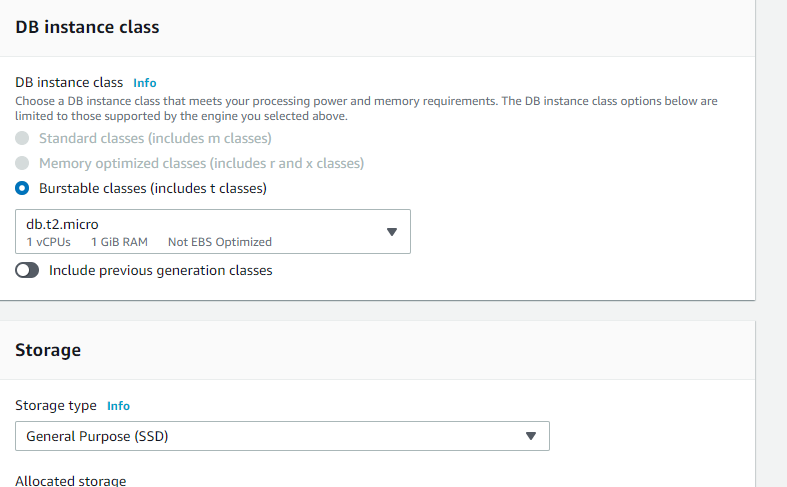
When you scroll down you can see version and free tier



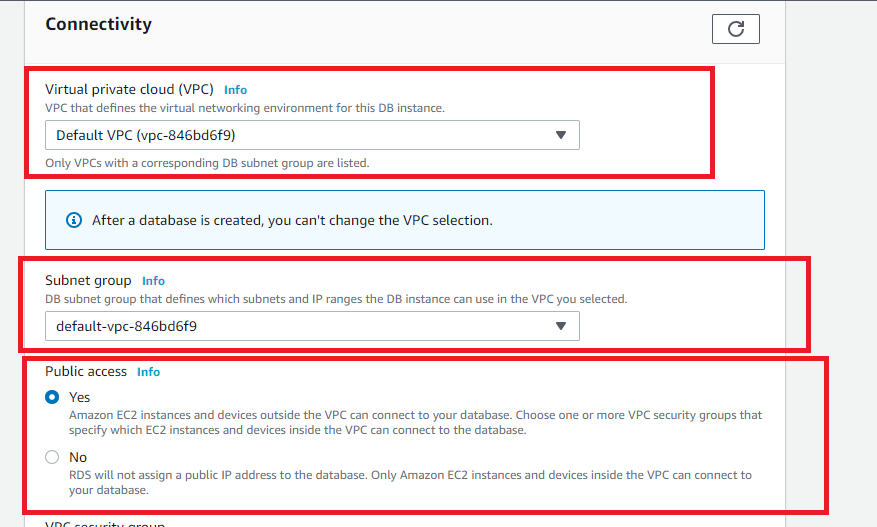
In the setting gives the instance name and master password for admin like ex: admin12345



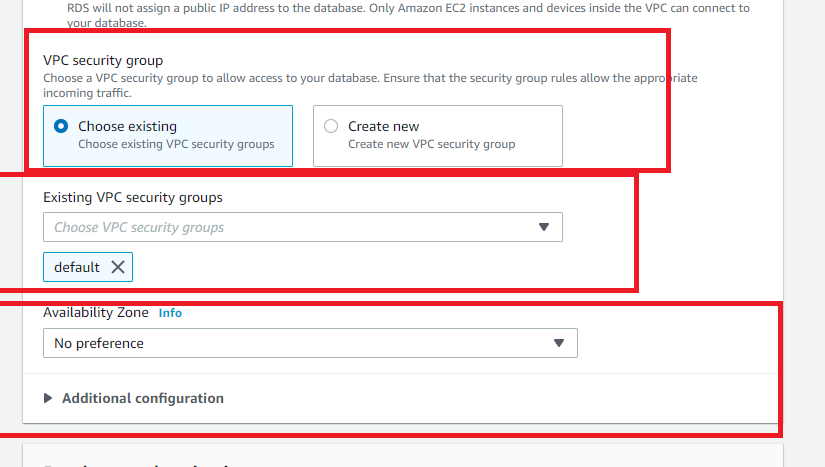
Other setting leave as it is



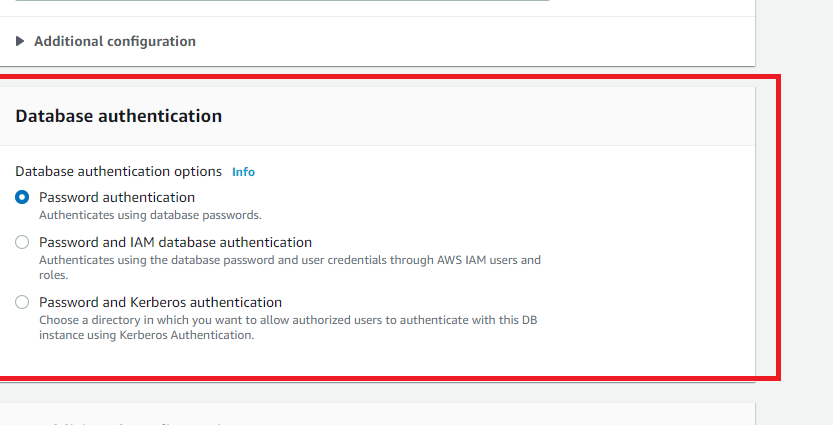
In Connectivity choose public access & keep default vpc as it is



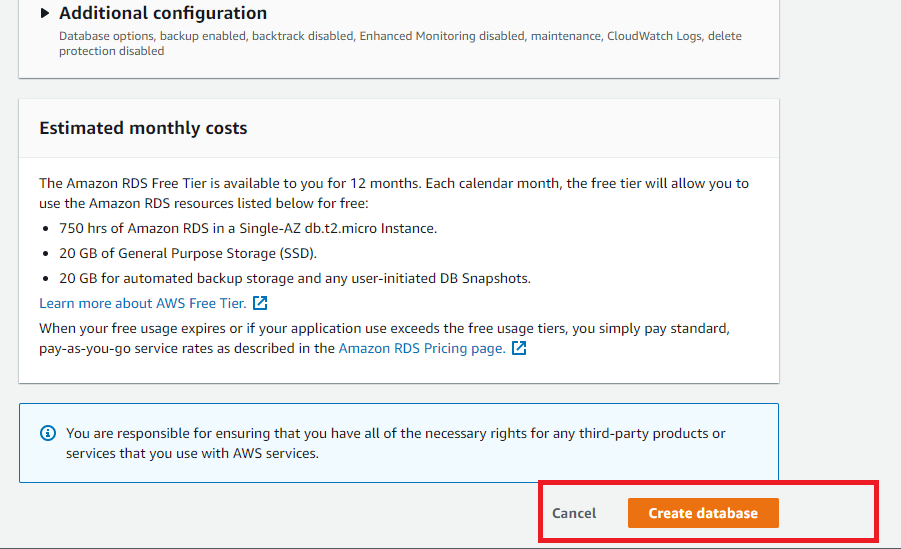
VPC security group and others keep as it is



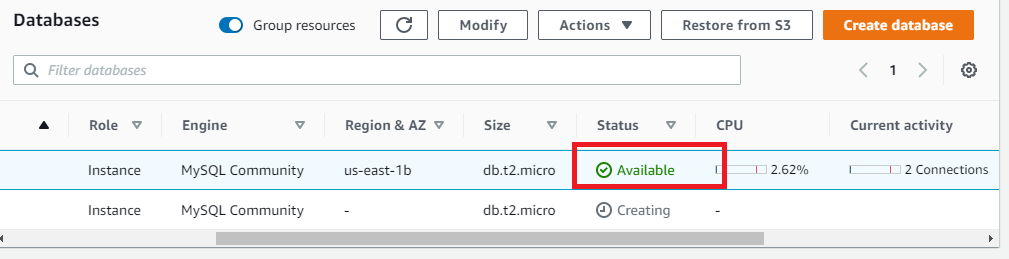
Database authentication keep default



After having all the settings default choose create database

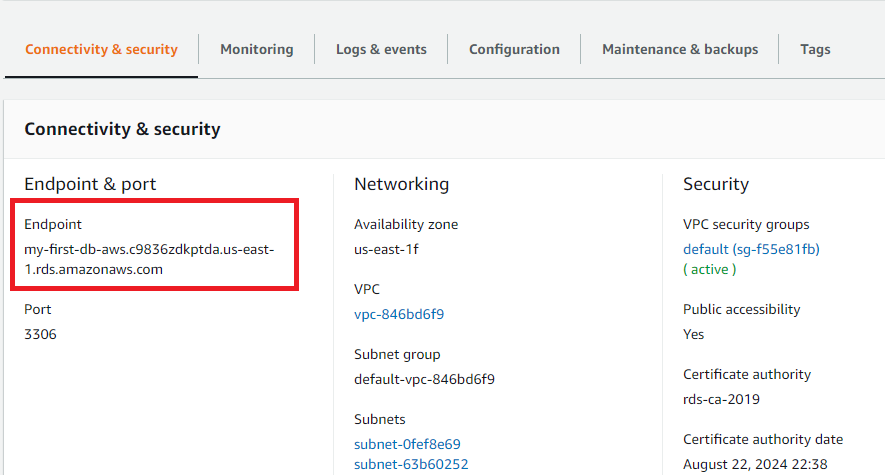


You will see the table with Available or Creating as the status, wait for database to be created

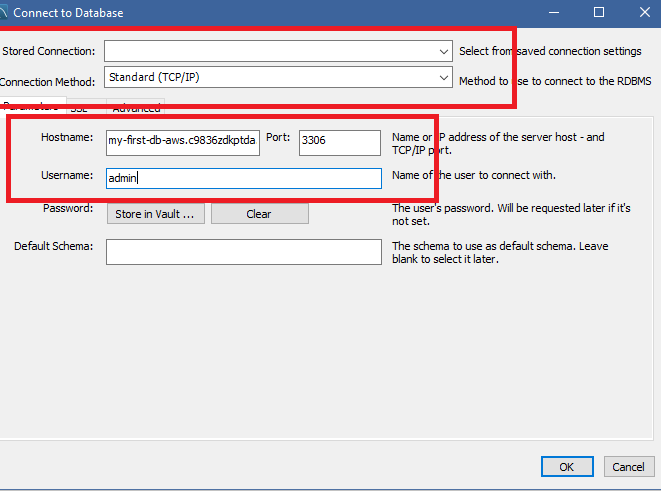


Once the database created you can connect from Mysql work bench

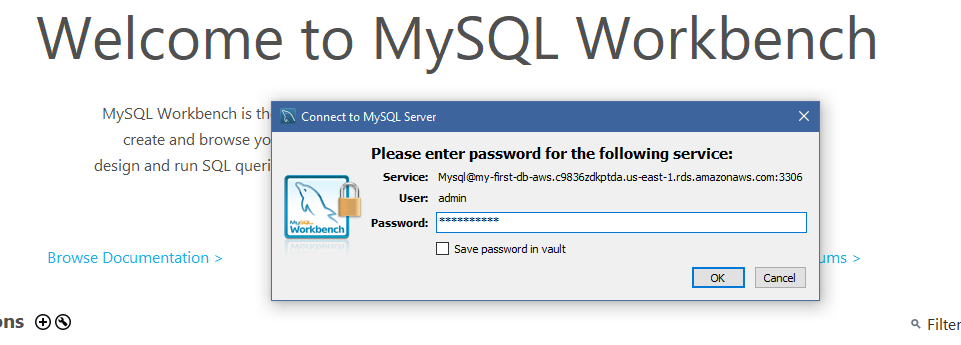
Click on the instance to locate the address in the connectivity & security



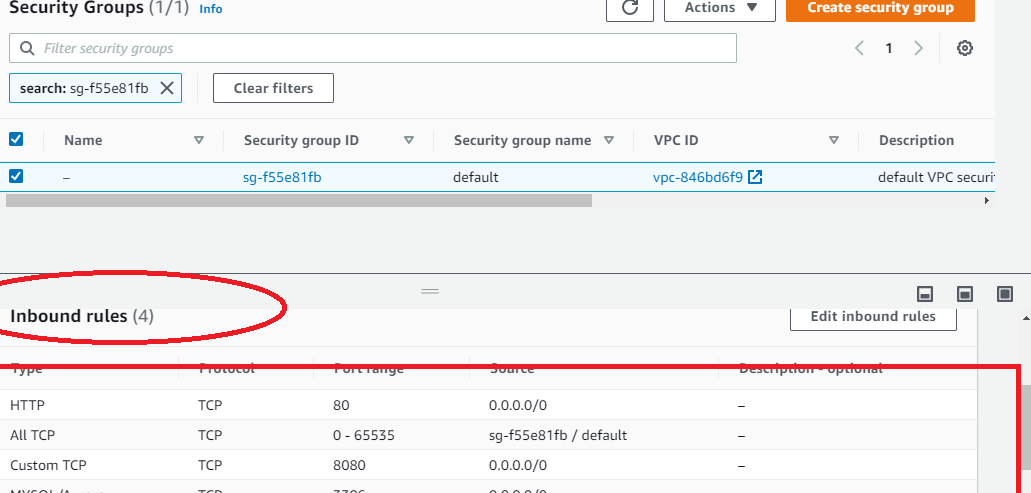
Open workbench and enter the host address



Enter the password you can see workbench connecting to the remote database.

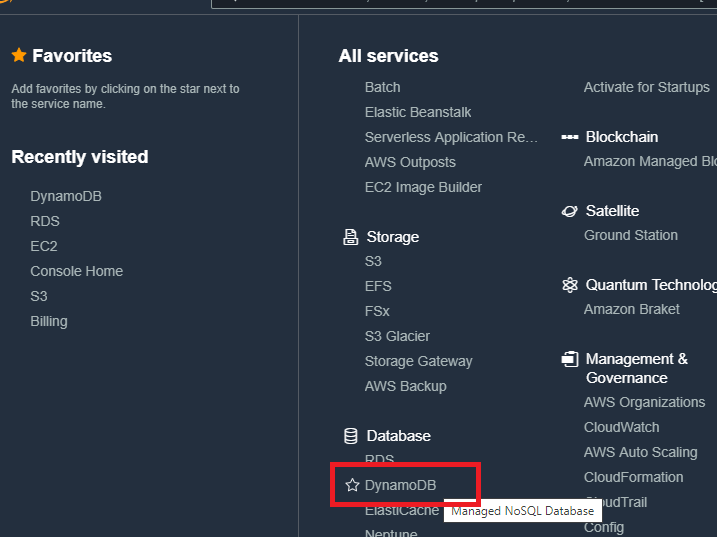


You can enter the queries now, if in case it fails to connect then edit the inbound rules and ensure these settings are present.



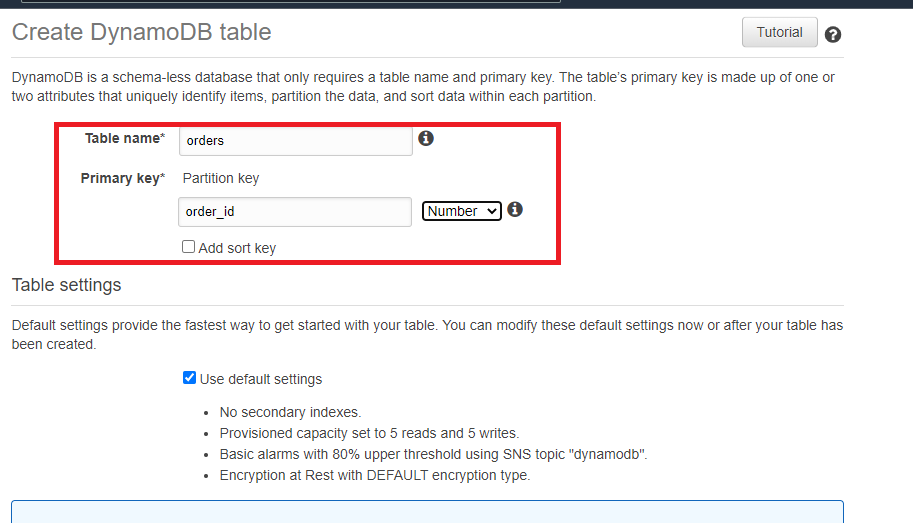
Similarly you can create DynamoDB to create tables and enter details, it is simple & straight forward

DynamoDB: It is for no-sql database

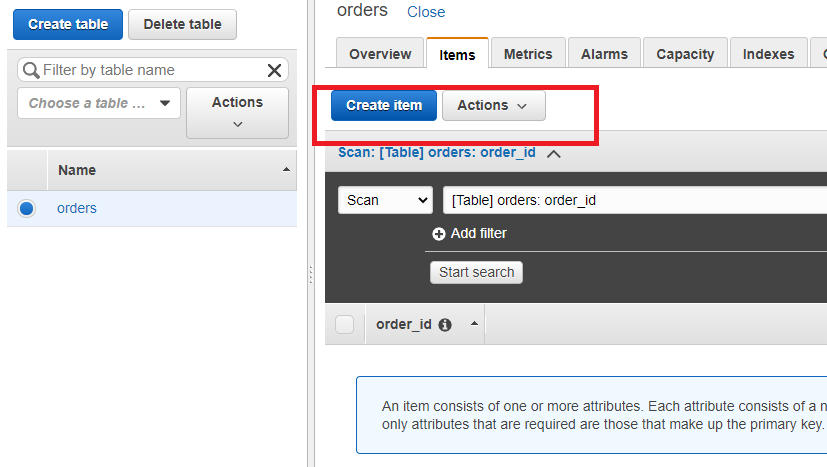


Steps:

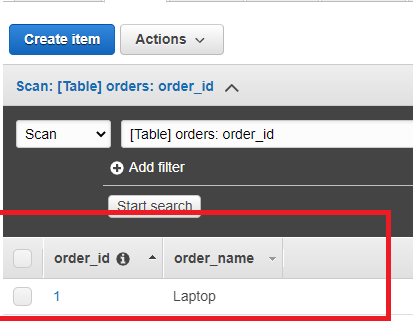
1. Select create table



1. You can select create item to enter the data



You can see the data after creating the item



You can delete the table once done, as DynamoDB is chargeable.