PGPCC

Project Implementation

Team Communication Solution using Mattermost and AWS

--Mahesh Jasti

Contents

Section 1: Problem Statement

Mattermost is an open-source, self-hostable online chat service with file sharing, search, and has other integration options. It is designed as an internal chat platform for organizations and companies, and mostly markets itself as an open-source alternative to Slack. It uses a 3 tier architecture that can be hosted using an IaaS provider or on-premises servers

The purpose of this project is to deploy trial version of Mattermost application along with MySql server in AWS.

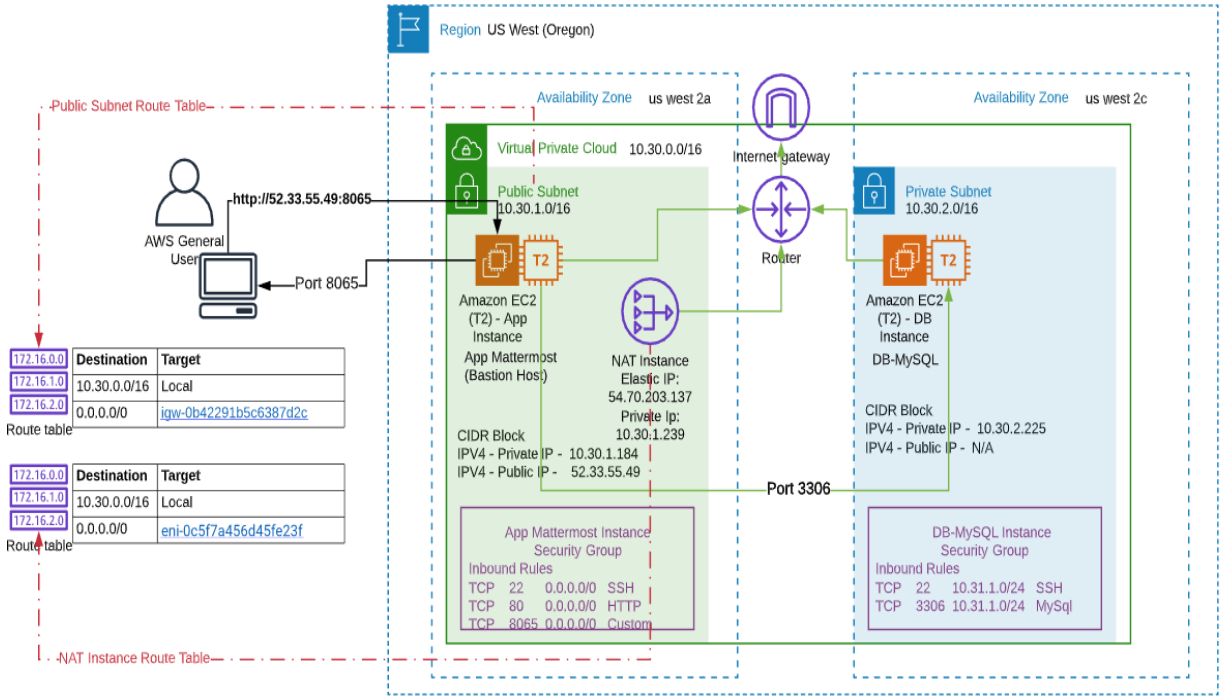
Section 2: Scope of the Project

The scope of the project implementation is as follows.

1. Architecture
2. Implementation
    a. Create VPC with 2 Subnets.
    b. Install and configure MySQL on an Ubuntu 18.04 in Private Subnet.
    c. Install and configure Mattermost on an Ubuntu 18.04 instance on the public subnet.
    d. Configure appropriate security groups.
    e. Validate the application by accessing the IP of the public instance thru port 8065.

Section 3: Implementation Architecture

The following picture outlines the implementation architecture for the project. I have placed all the information related to the IP addresses(Public or Private), Route table configuration, Security Groups rules etc. This will give complete picture on how I have implemented the project



Section 4: High level Steps to implement the project

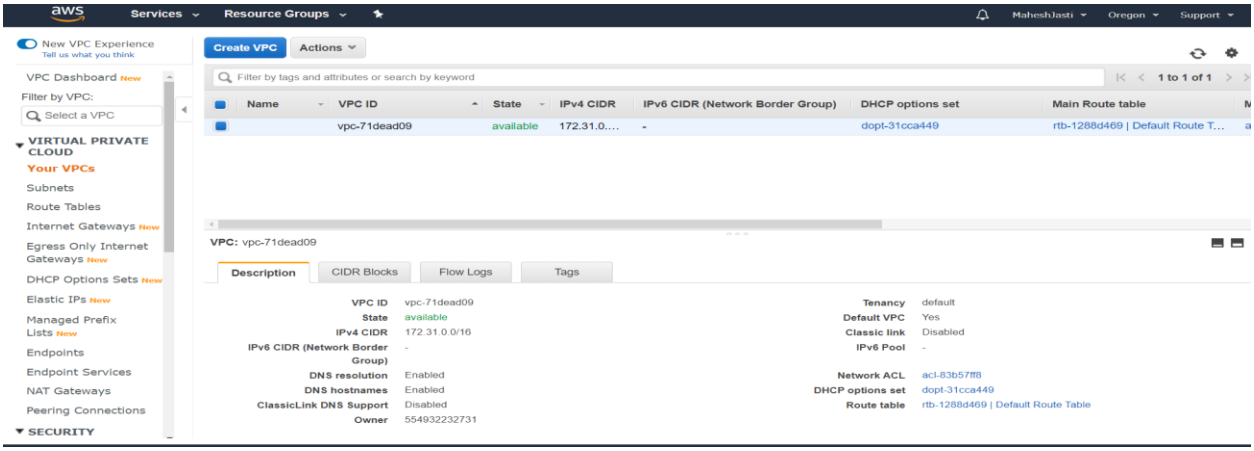
1. VPC, Subnets, Internet Gateway and Route Tables setup
2. Ubuntu EC2 and NAT Instances setup
3. Route table setup for PrivateSubnet setup
4. EC2 instances – Access check b/n Public and Private
5. Installation of Mysql & Mattermost softwares and browse the public IP– Setup

Section 5: Screenshots from AWS console

**********VPC, Subnets, IGWY and Route Tables setup **********

Step#1

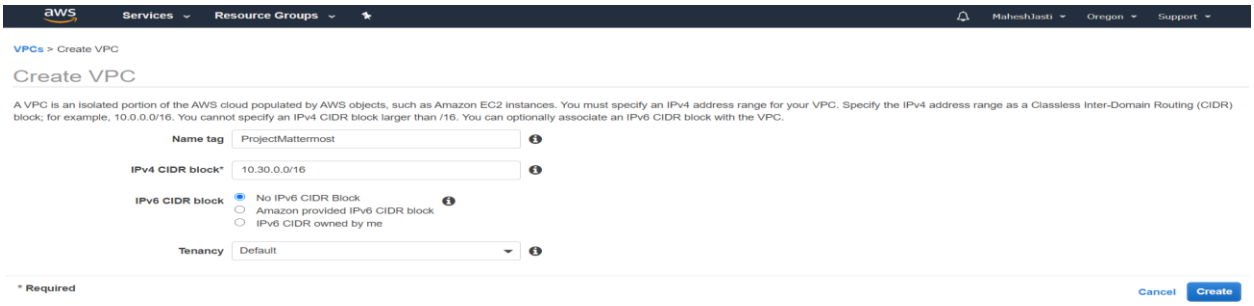
Login to AWS Console and search for VPC



Step#2

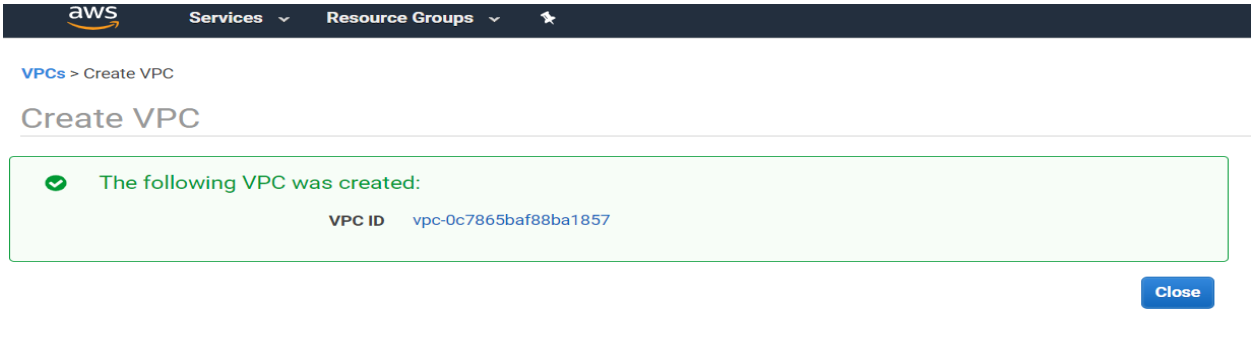
Click on Create VPC button and enter the respective details

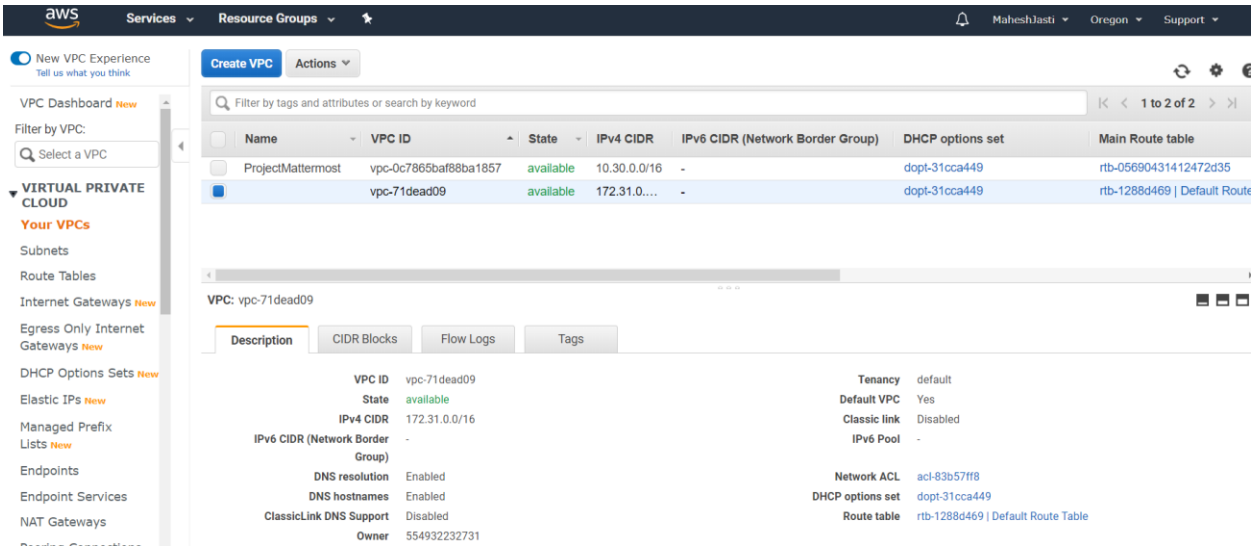
Nametag: ProjectMattermost ; CIDR Block: 10.30.0.0/16



Step#3

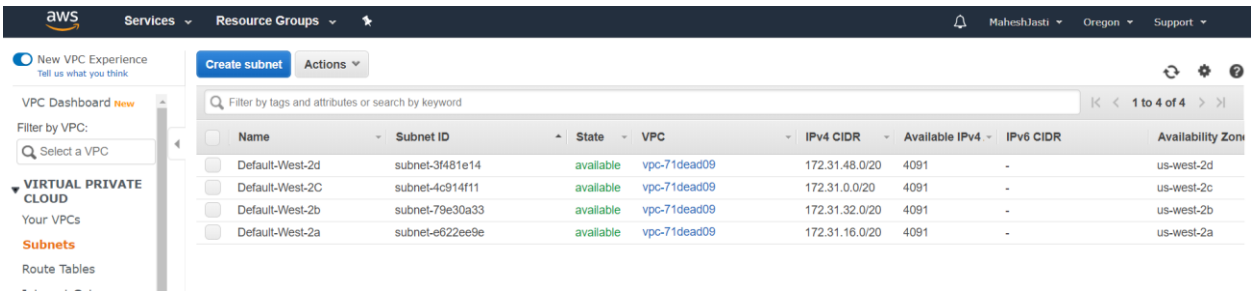
Once click on create, it creates a VPC ID in the main window





Step#4

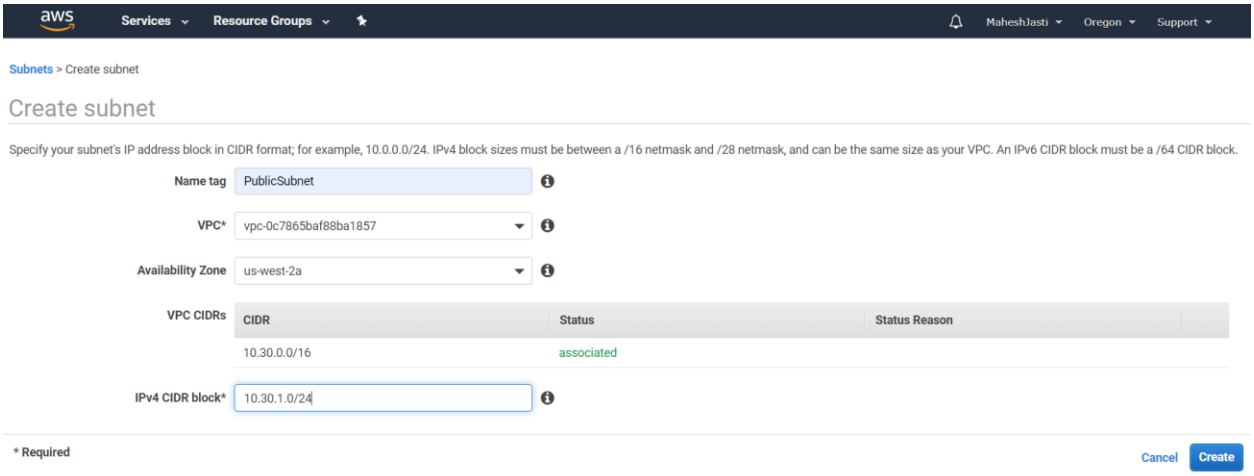
Now after creating the VPC, click on the Subnets option on the left menu..

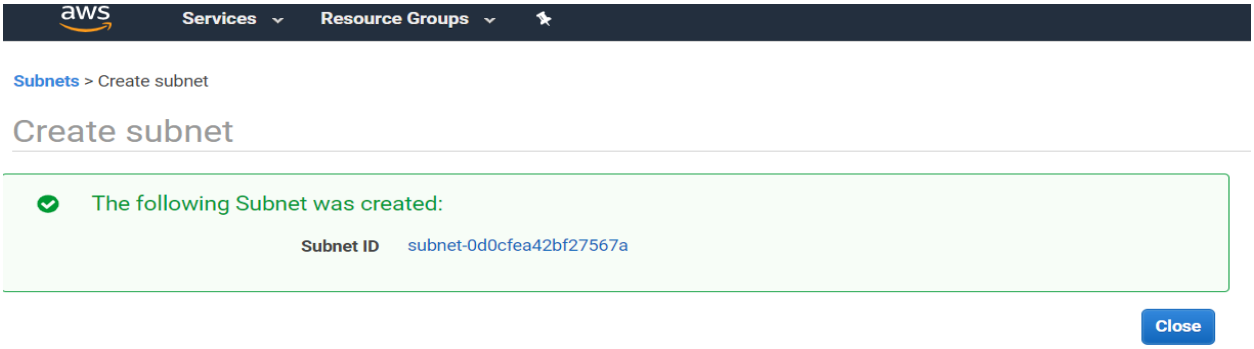


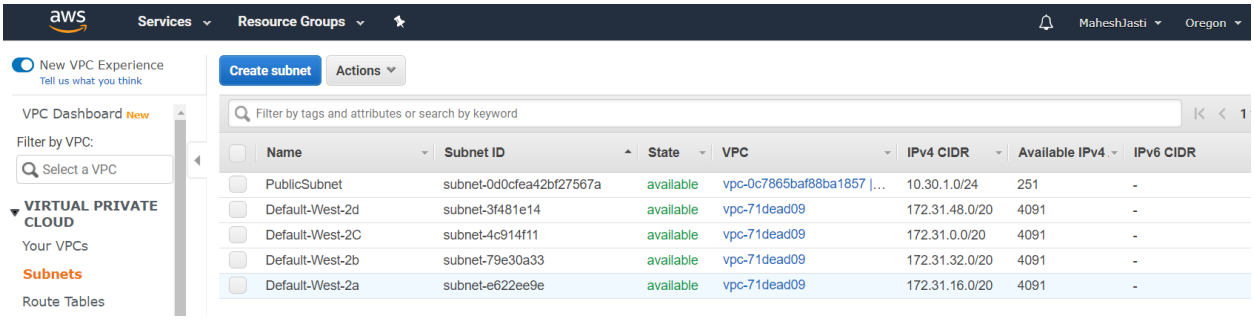
Step#5

Click on “Create subnet” blue button and create a Subnets as shown below.

Subnet#1 – PublicSubnet with CIDR addr as 10.30.1.0/24 under us-west-2a



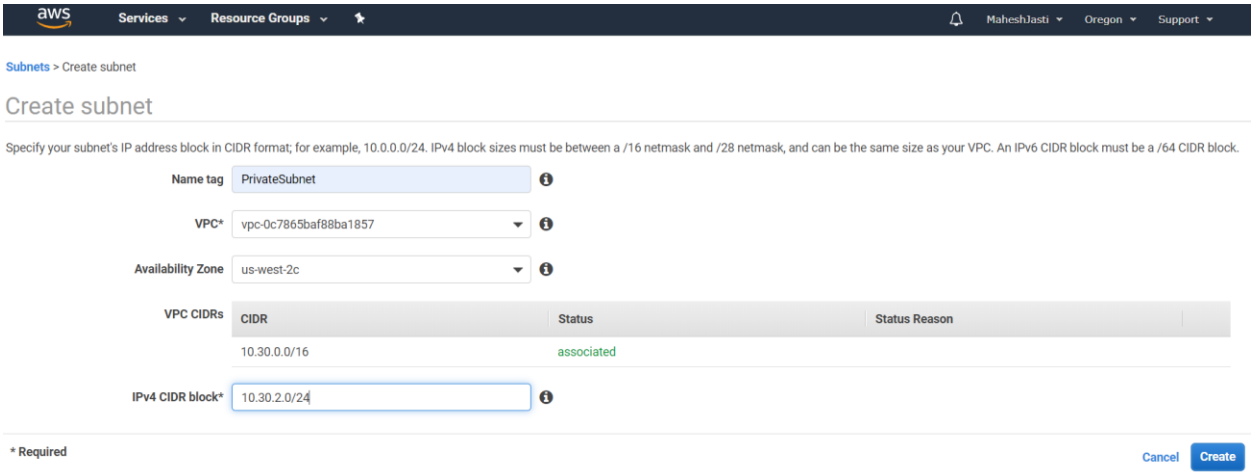


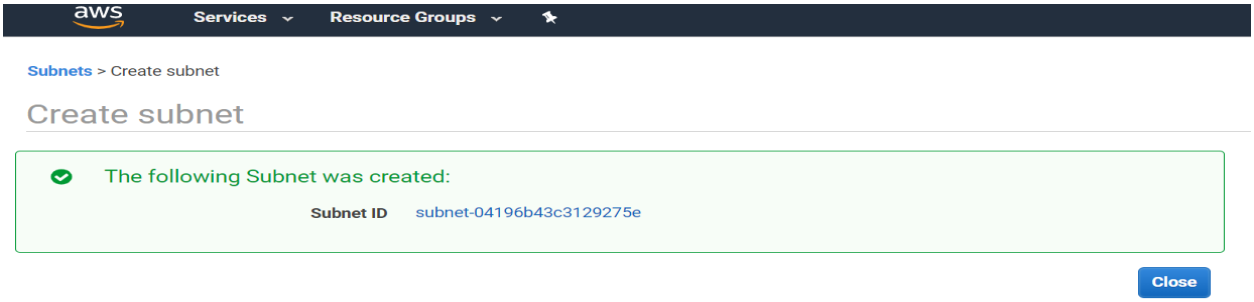


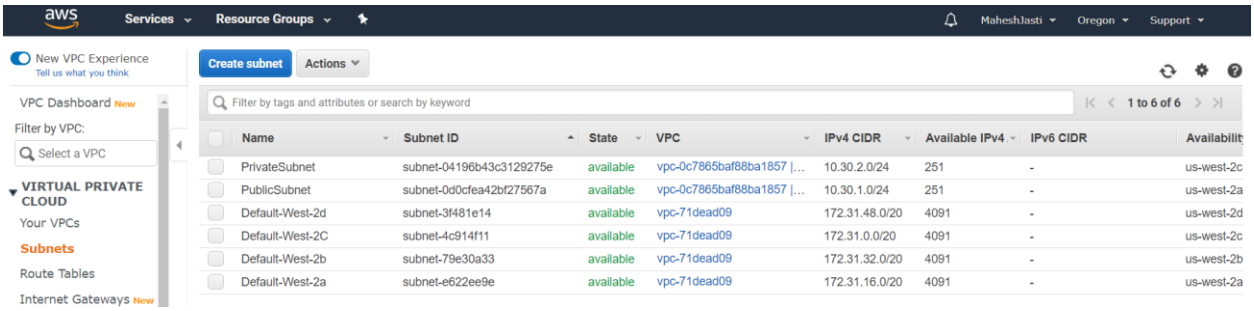
Step#6

Subnet#2 – PrivateSubnet

CIDR: 10.30.2.0/24 under us-west-2c





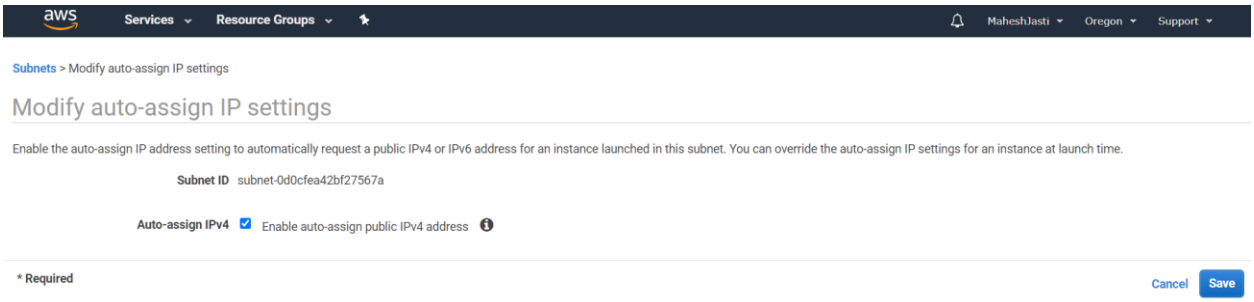


Both the Subnets has been created with 251 Available IPV4 addresses.

Step#7

Now select the "PublicSubnet" and go to actions and select the Modify auto-assign IP settings.

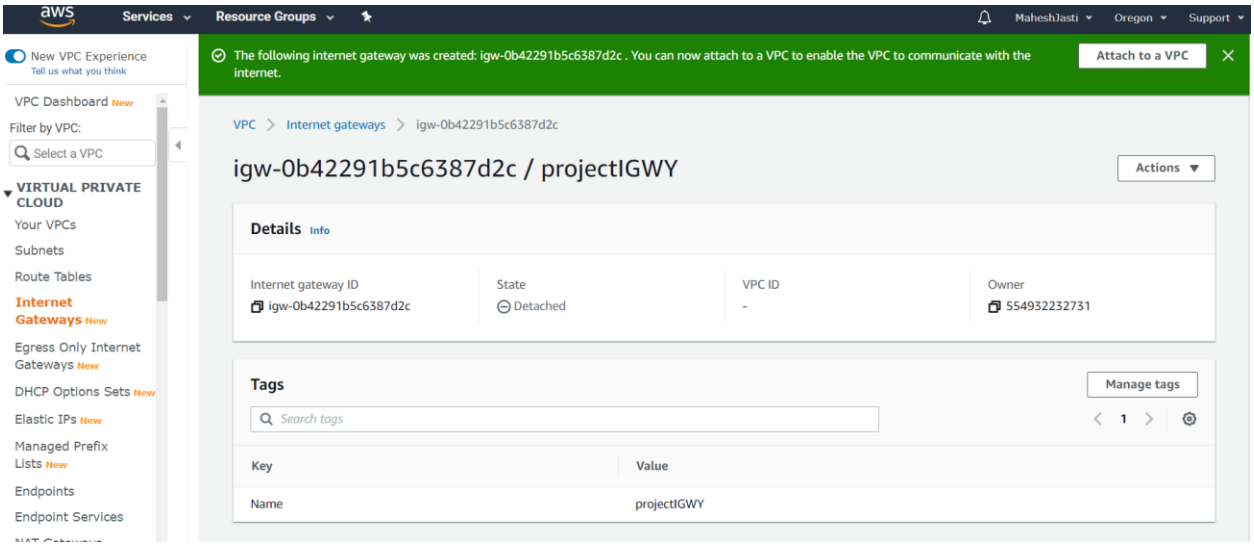
Enable Auto Assign public IPV4 Address for the "PublicSubnet" and click on Save. This will help to create the IP address automatically when EC2 instance created.



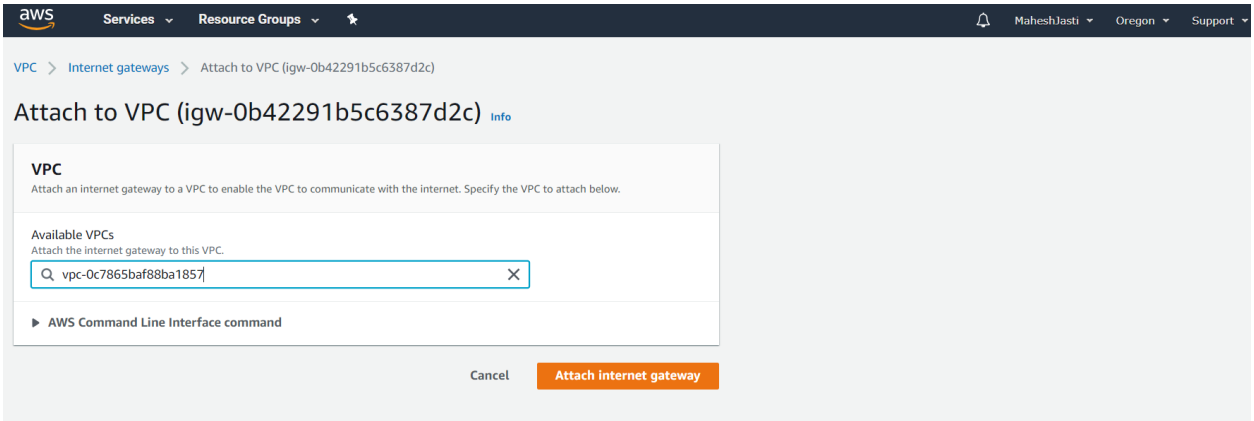
Step#8

Create an Internet Gateway. Click on the Internet Gateways option on the left menu.

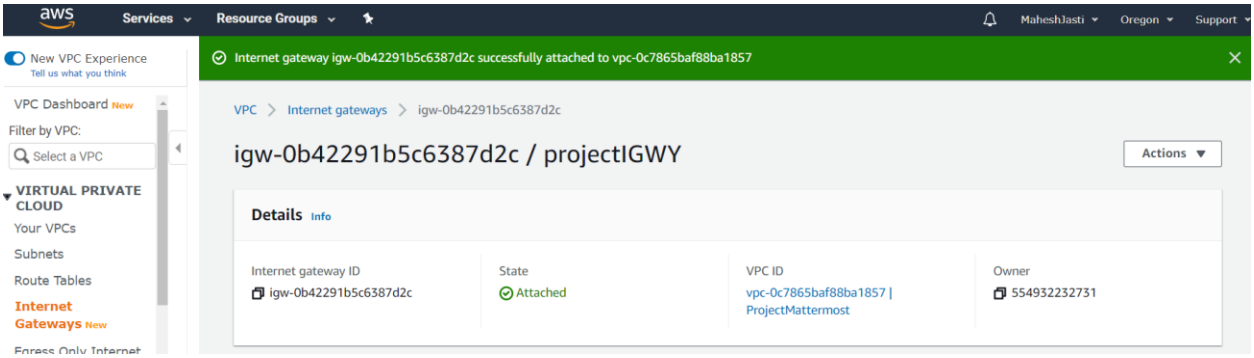
Now click on "Create Internet Gateway" at top right side of the screen. Enter the name tag as "projectIGWY" and click on Create Internet Gateway button.



Now attach to the ProjectMattermost VPC by clicking at the button “Attach to a VPC”

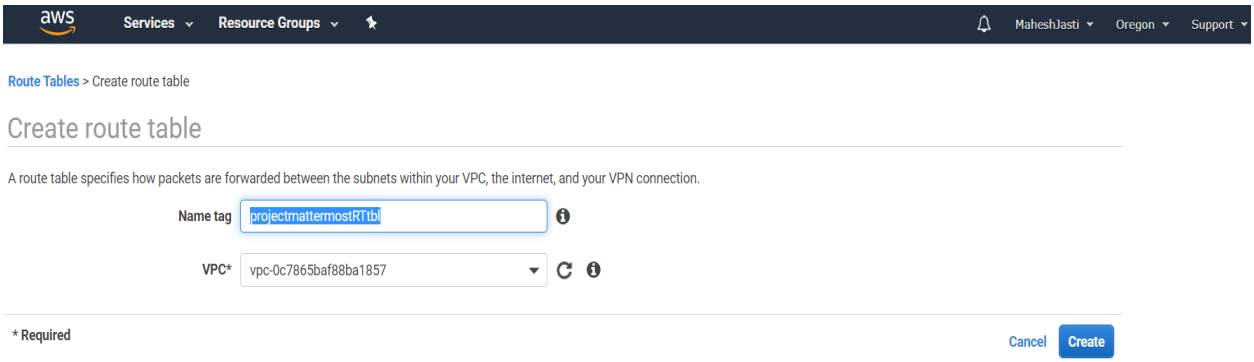


After attaching the IGWY we would see the following screen.

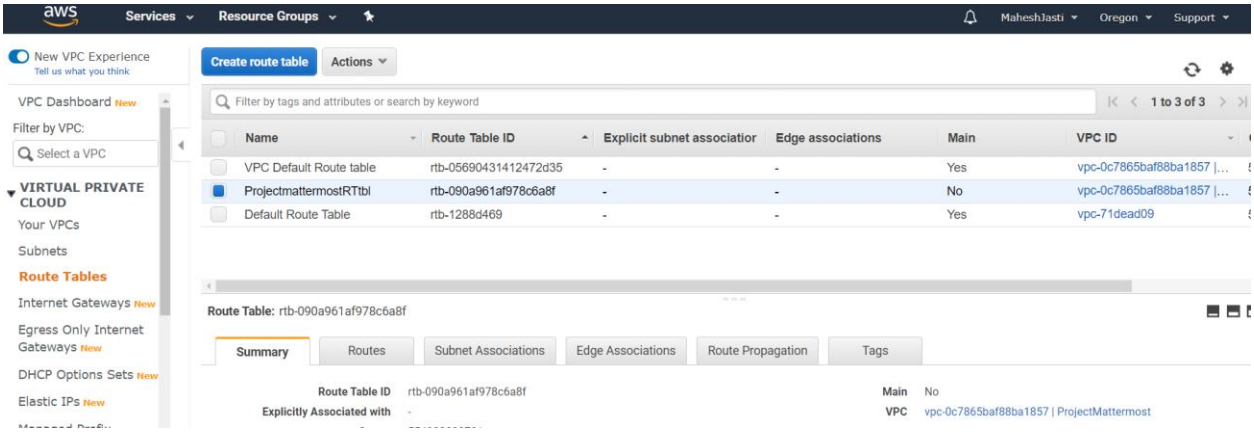


Step#9

Now create a new Route table(“projectmattermostRTtbl”) for the VPC by clicking on the Route Table option.

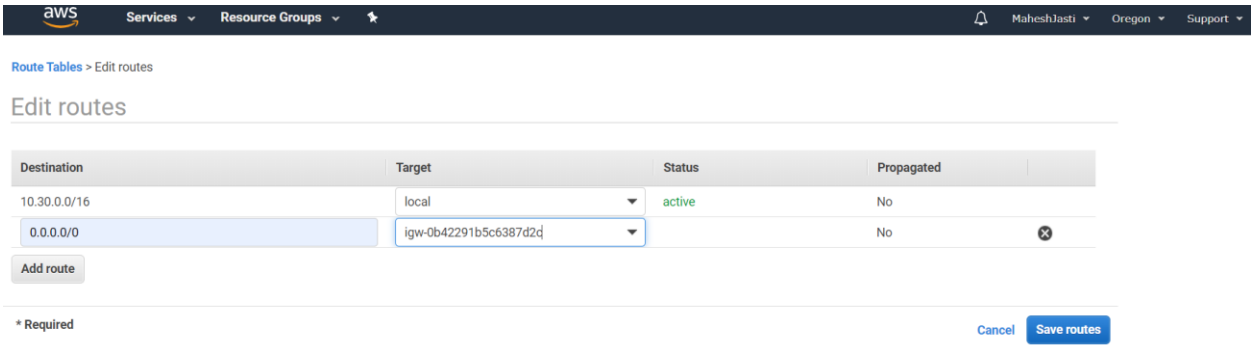


The route table is created and the "Main" attribute should show **"No"**

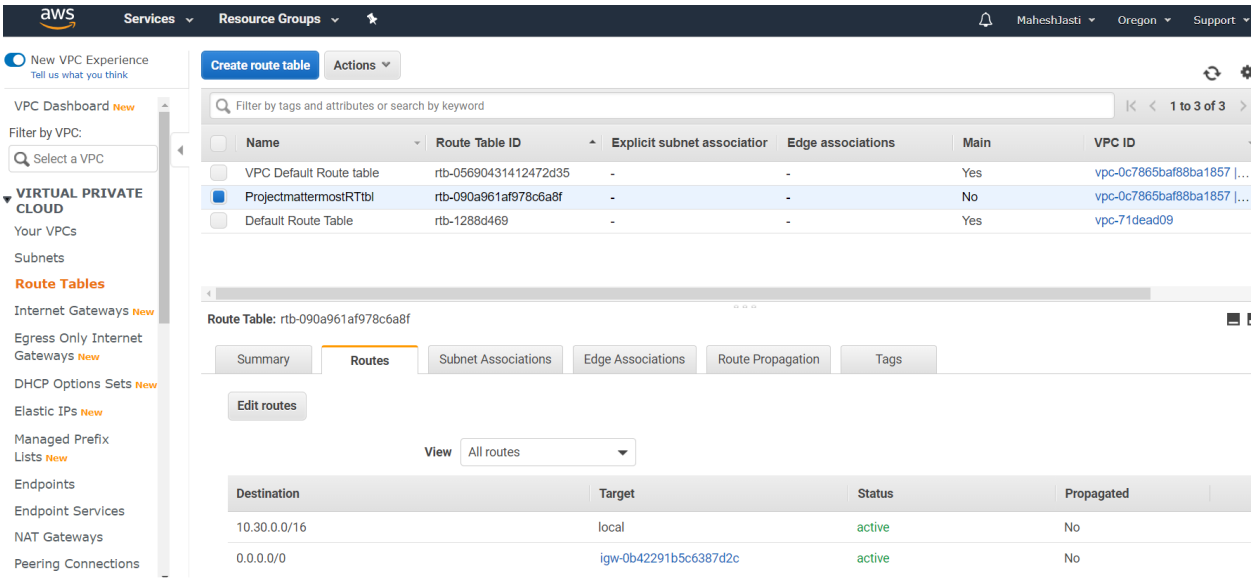


Step#10

Go to Routes tab and click on "Edit Routes" and add the Internet Gateway and click on Save routes button.

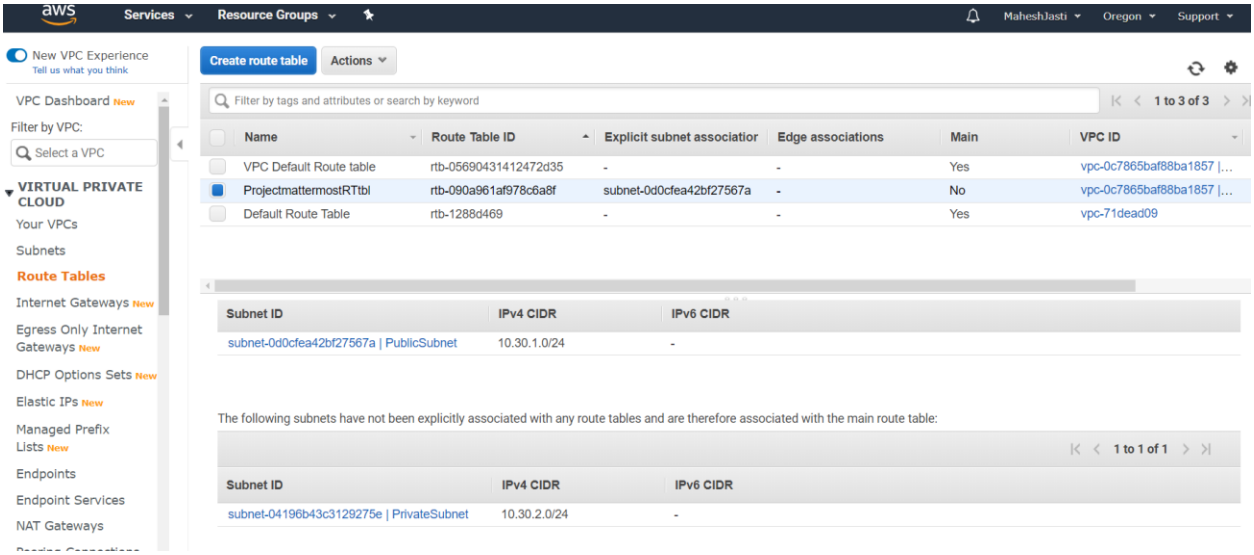


And should show as below after pressing the Save routes button.



Step#11

Go to “Subnet Associations” tab and click on “Edit subnet associations” button and associate the PublicSubnet.



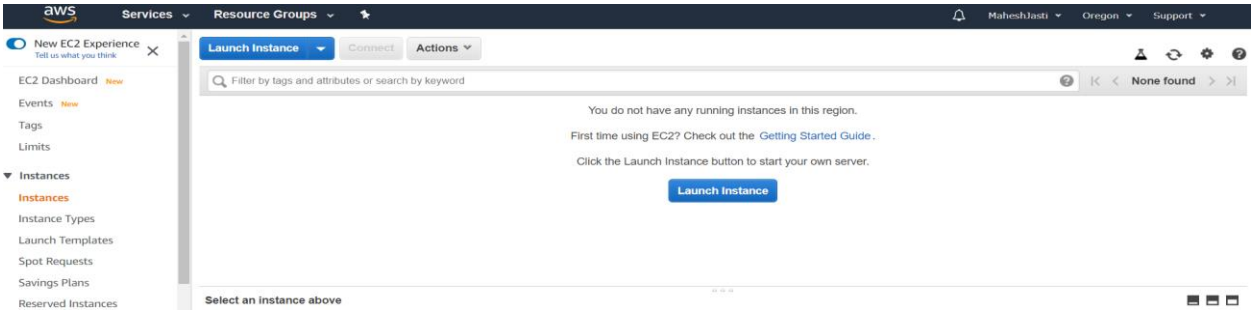
**********VPC, Subnets, IGWY and Route Tables setup is completed**********

**********Ubuntu EC2 and NAT Instances setup **********

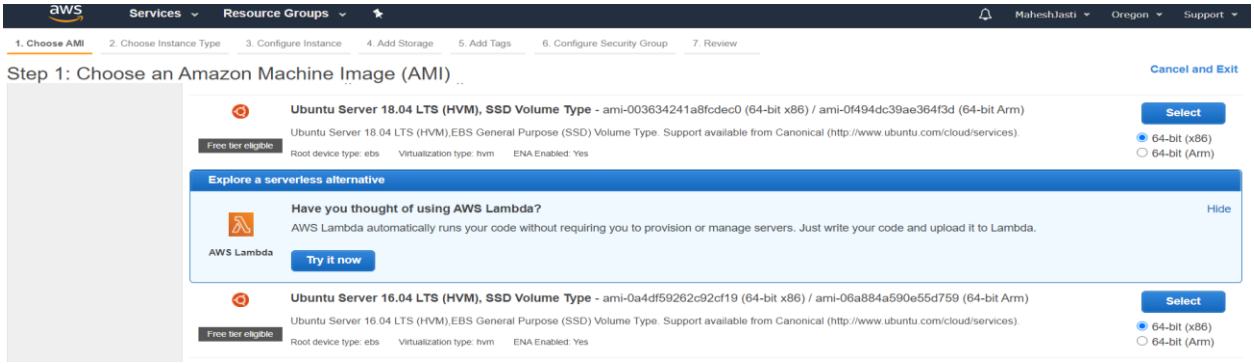
Create 2 EC2 instances – (Ubuntu 18.04 LTS)

Step#12

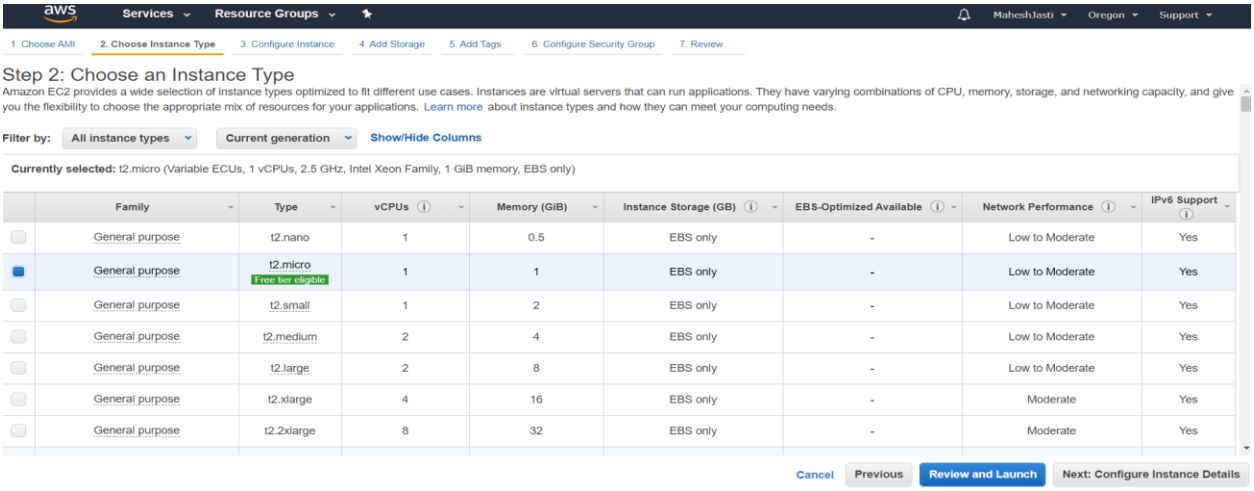
Go to EC2 window, Click on Launch Instance



Select Ubuntu 18.04 LTS AMI

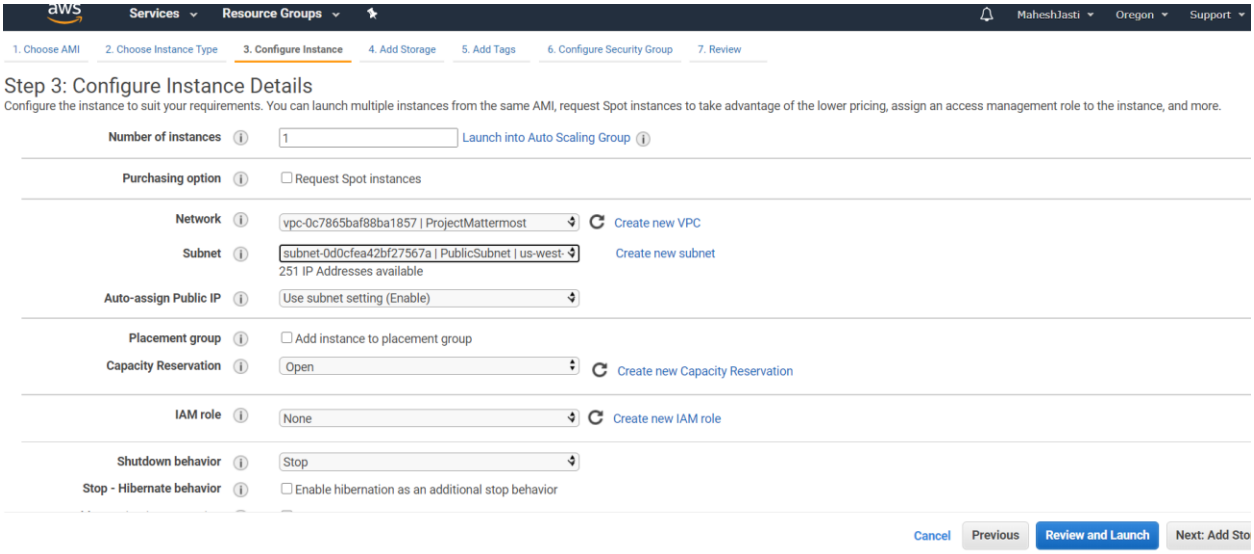


And select Ubuntu Server 18.04 LTS (FreeTier)



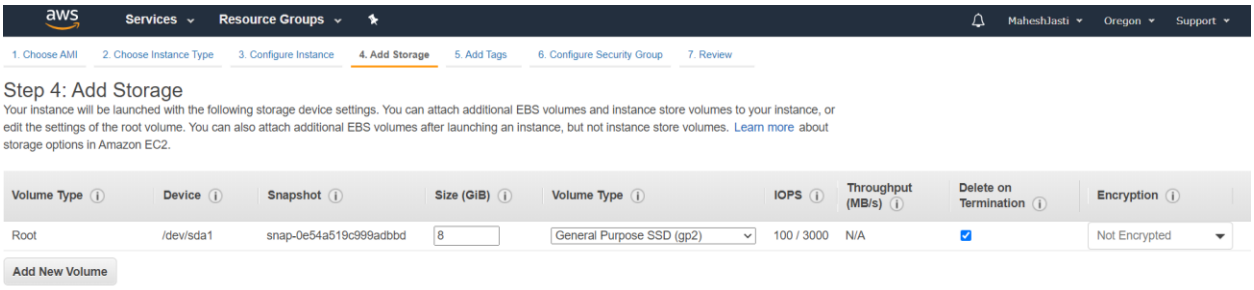
Step#13

Select the ProjectMattermost VPC and Public Subnet and click on Add Storage button below..



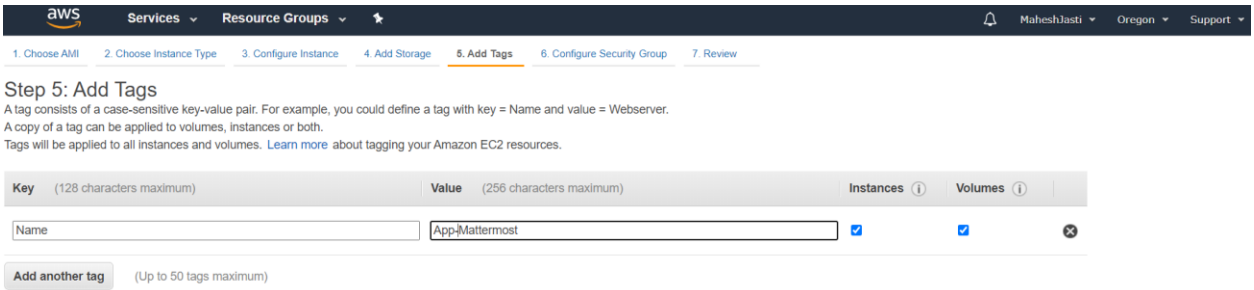
Step#14

No changes to Storage screen



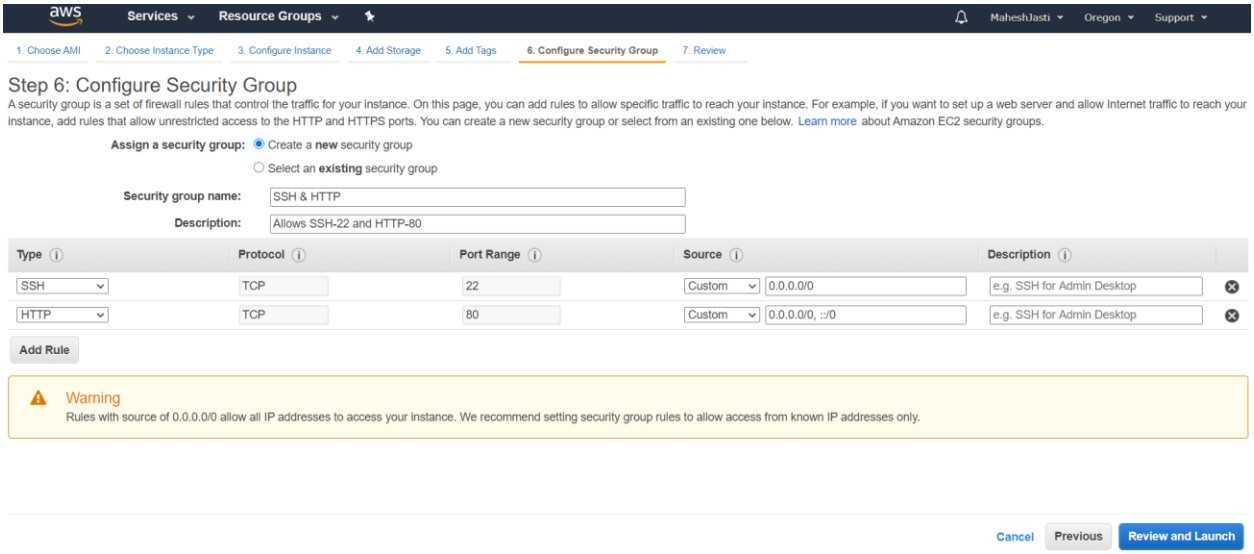
Step#15

Add the name tag as “App-Mattermost”



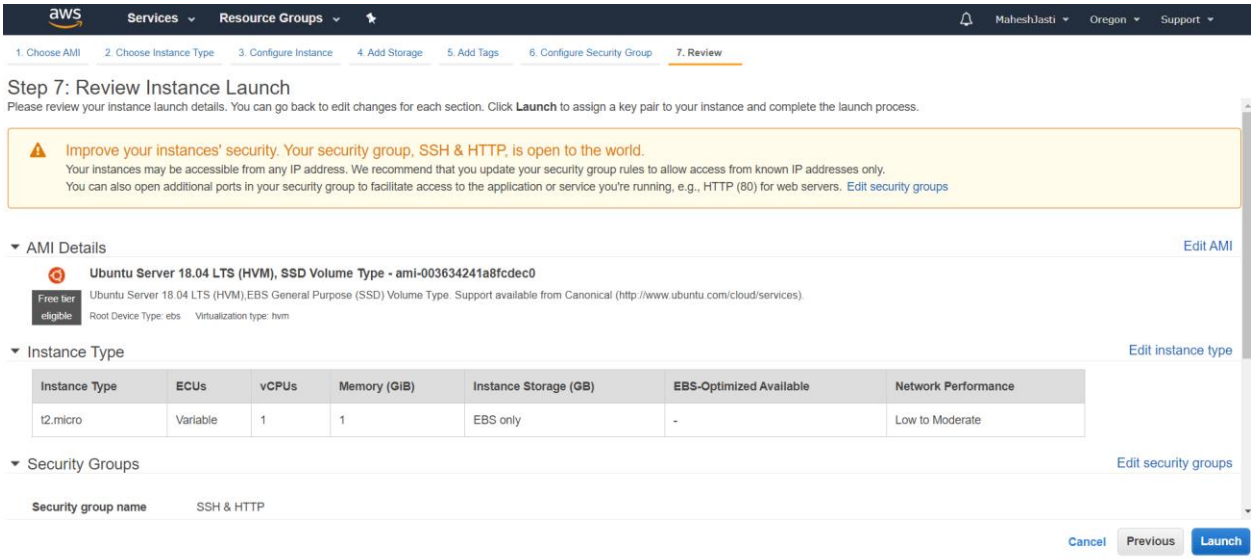
Step#16

Create a security group as below. Allow SSH and HTTP (as to install the software and access the application through outside). Click on “Review and Launch” button



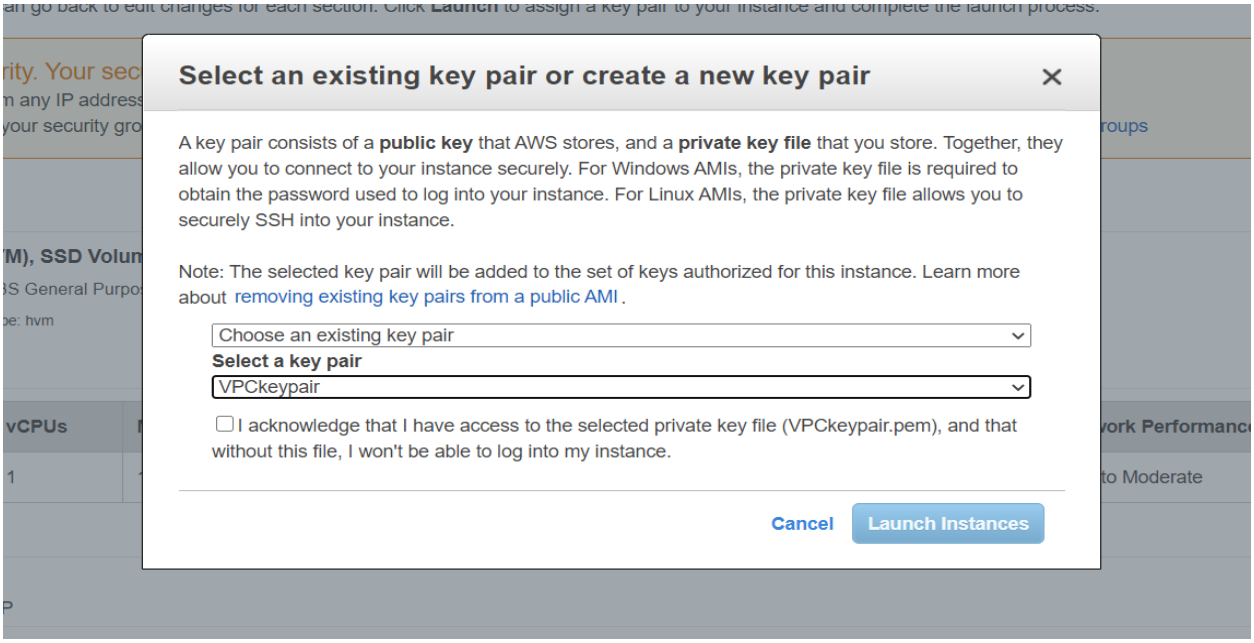
Step#17

And click on Launch button

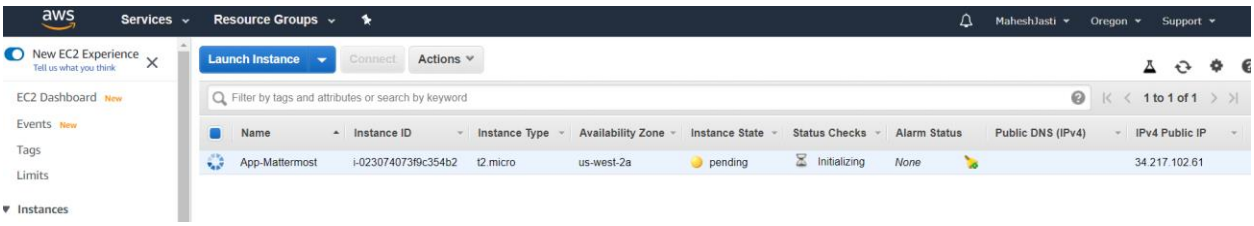


Step#18

Now Choose an existing Key Pair (VPCkeypair)… Acknowledge and Click on “Launch Instances”

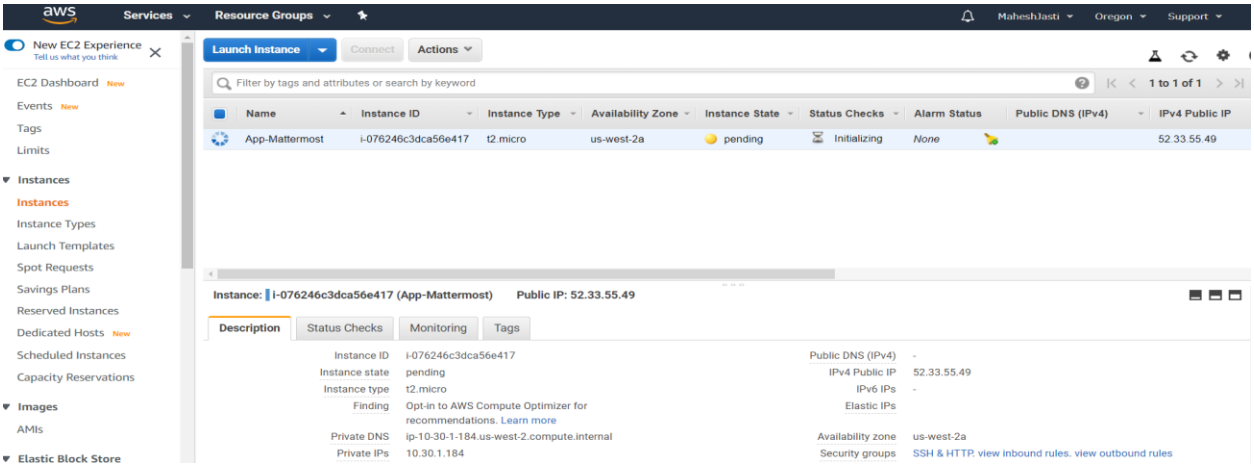


It will show like below.



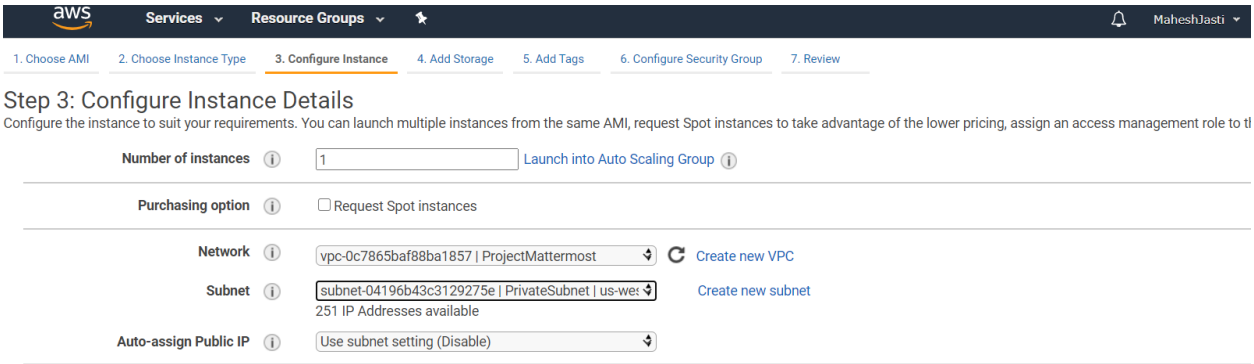
Step#19

The Public IP addr – 52.33.55.49 and the Private IP - 10.30.1.184 has been assigned.



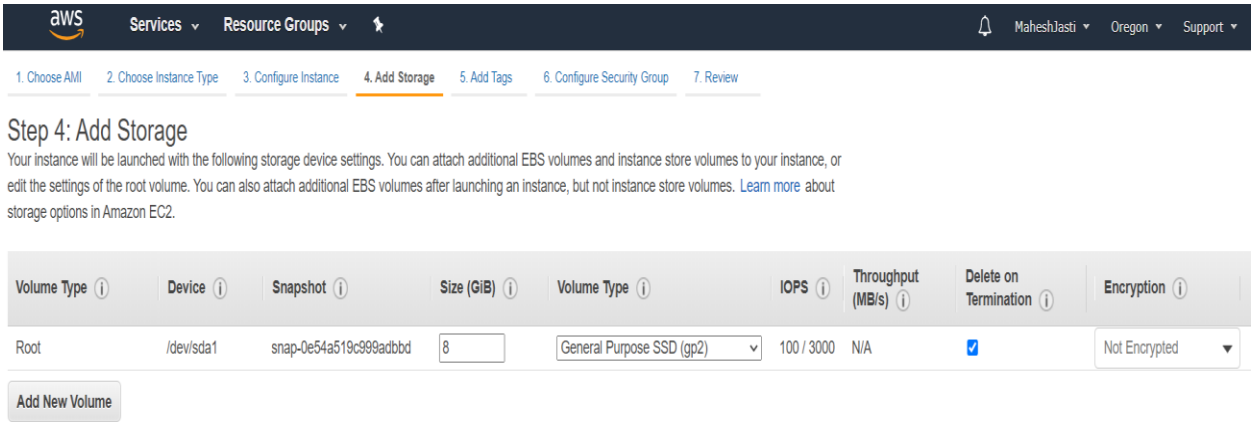
Step#20

Now launch the another Ubuntu EC2 instance and select the subnet "PrivateSubnet" (us-west-2c)



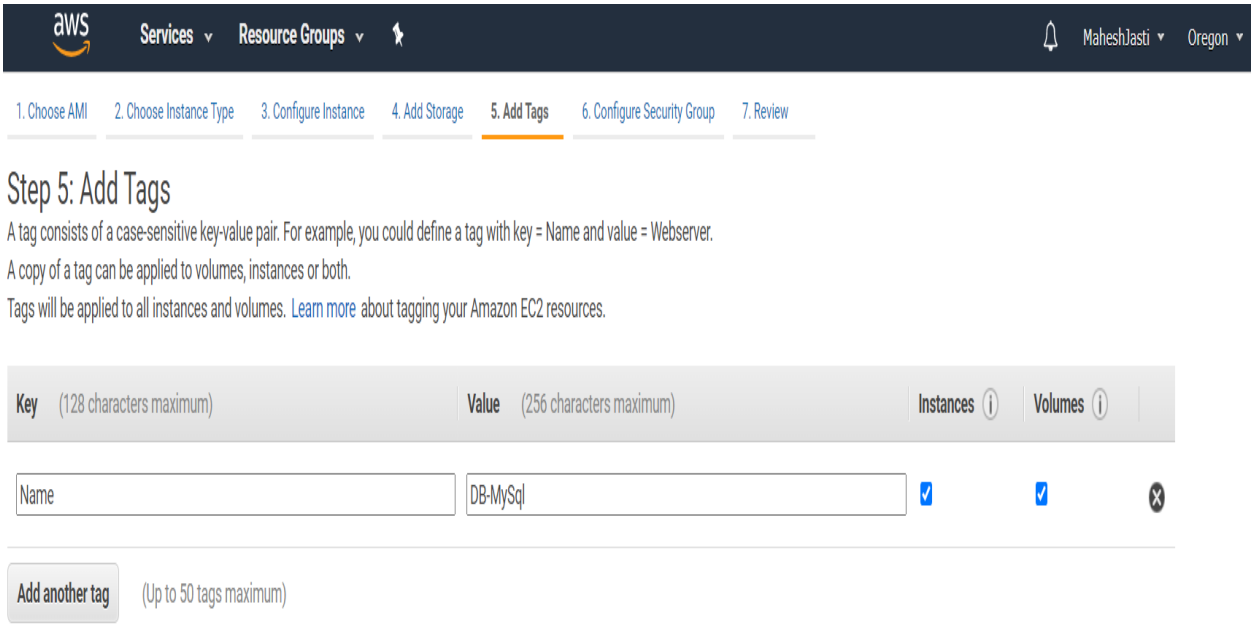
Step#21

No changes to the Add Storage screen



Step#22

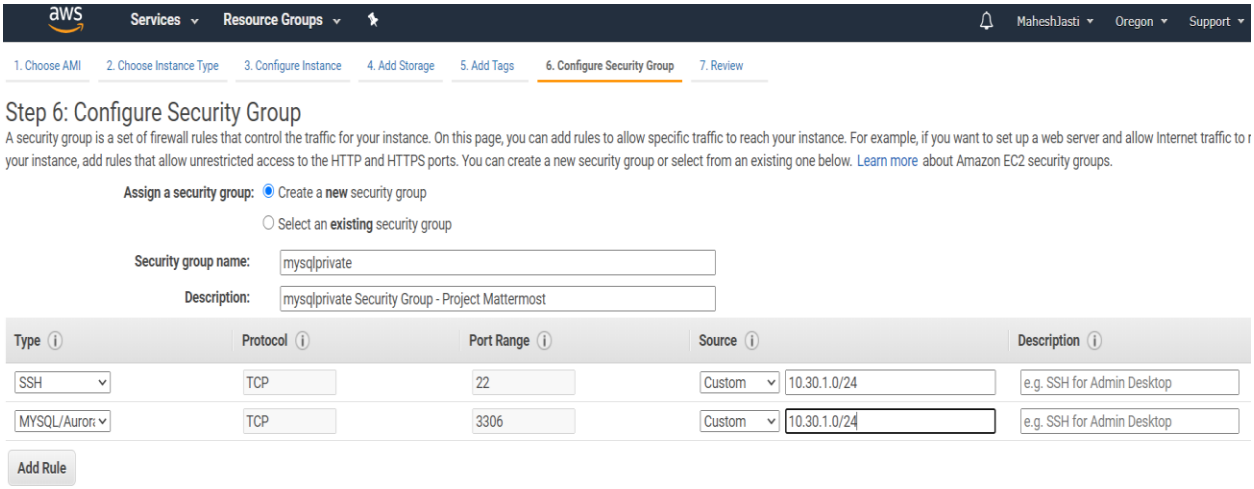
and enter the name tag as “DB-MySql”



Step#23

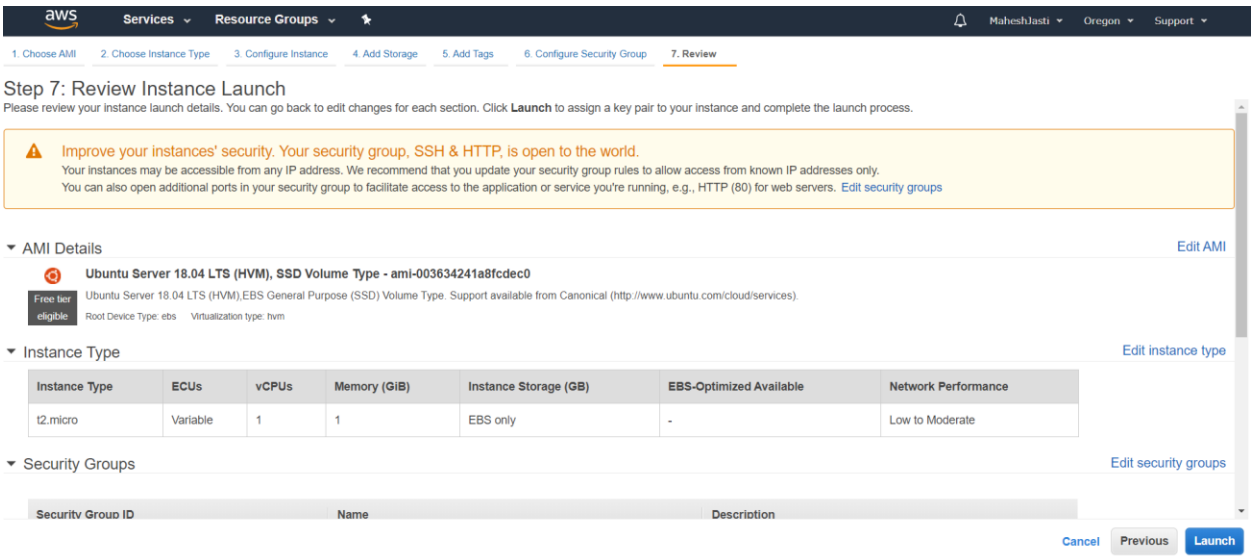
Select existing Security group – mysqlprivate

We want to restrict the access to the private subnet, Hence we are restricting the access to only public CIDR's



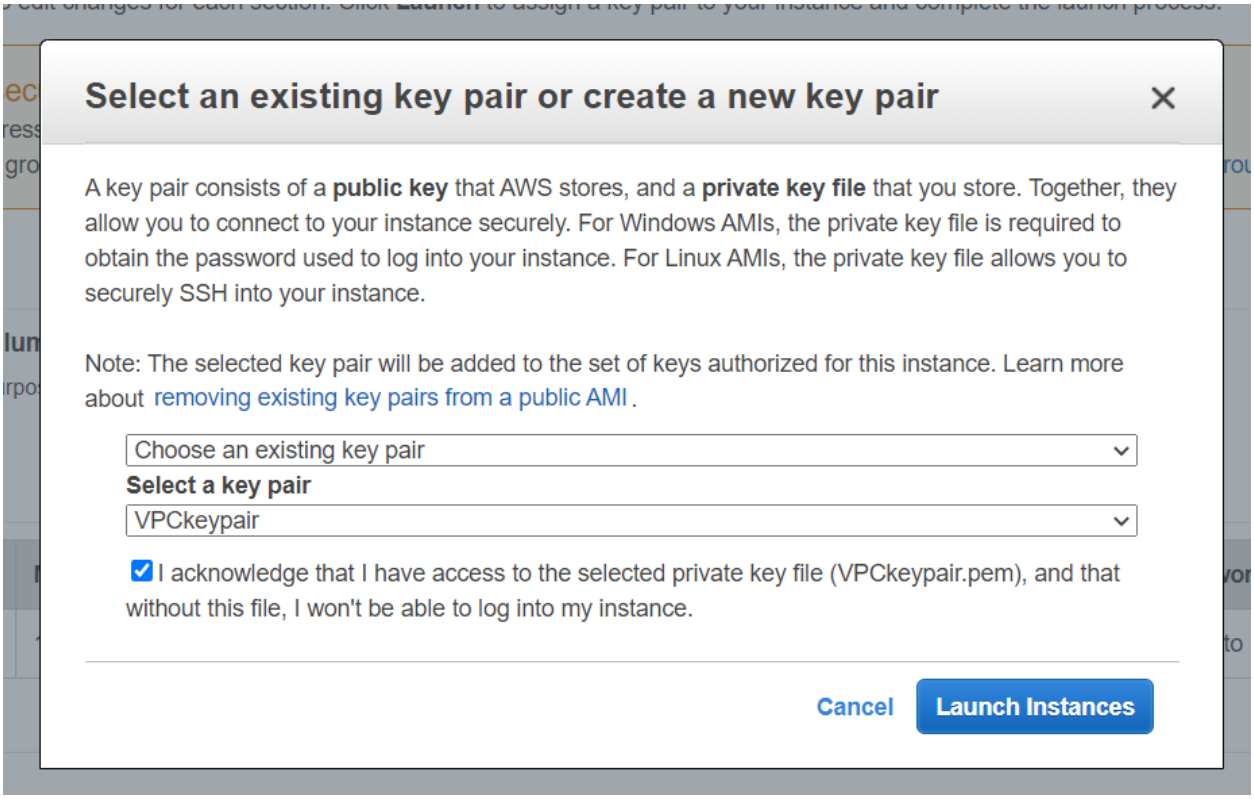
Step#24

Click on Launch



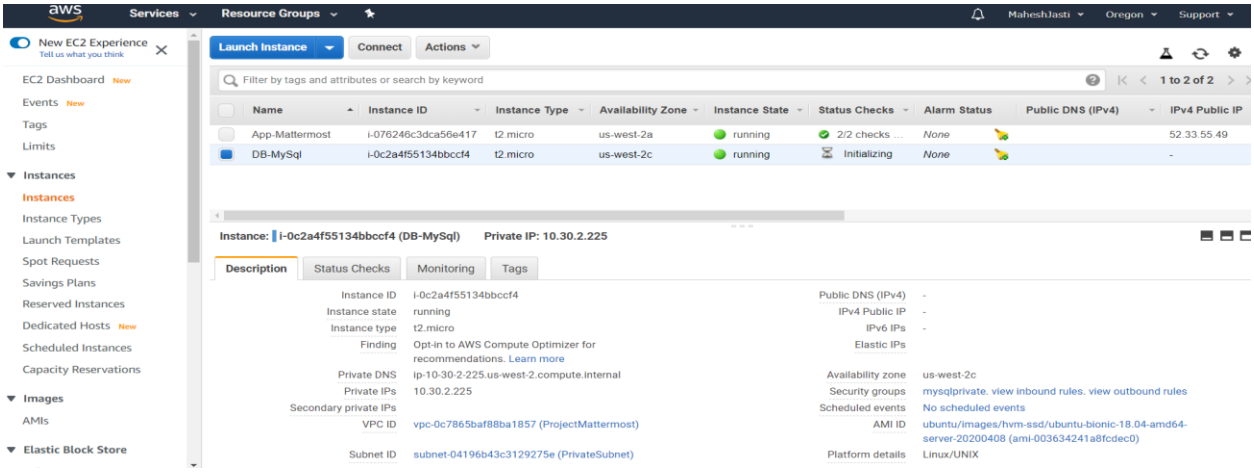
Step#25

Now choose an existing key pair and click on Launch Instances.



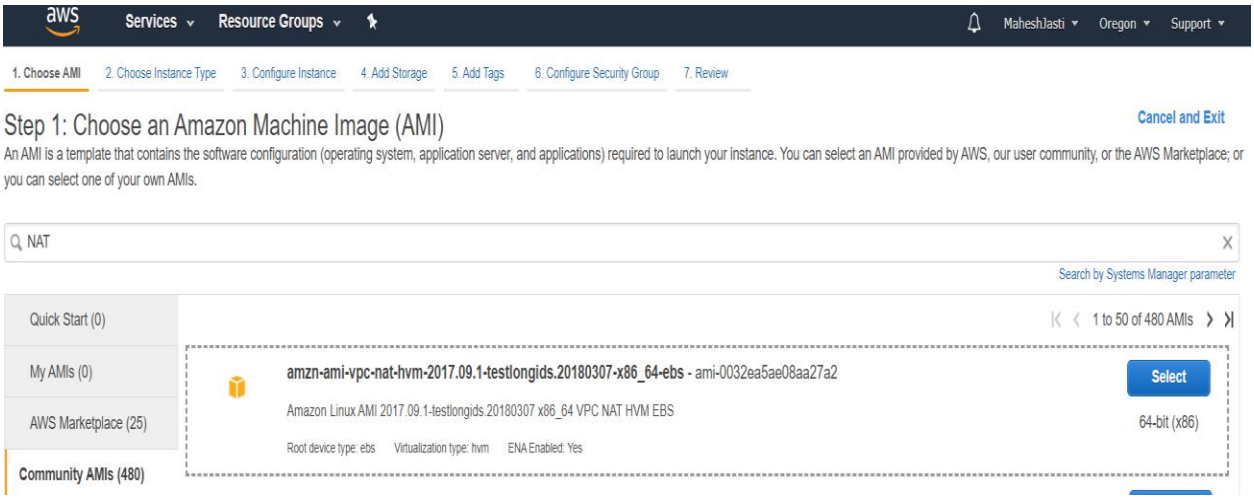
Step#26

The DB-MySQL instance didn't get the public IPV4 address as we haven't enabled the Auto Assign IPV4 Address for the Private Subnet. (Private IP: 10.30.2.225)



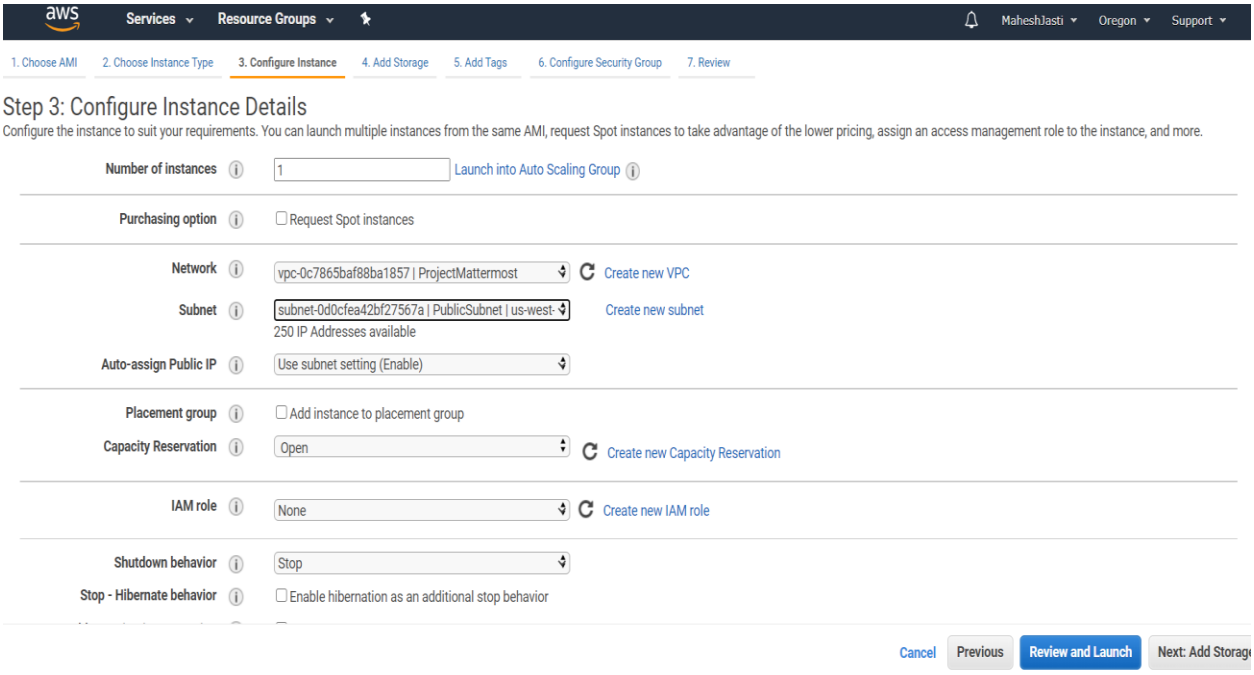
Step#27

Create a NAT Instance in the PublicSubnet and link to the Private Subnet in the Route table.



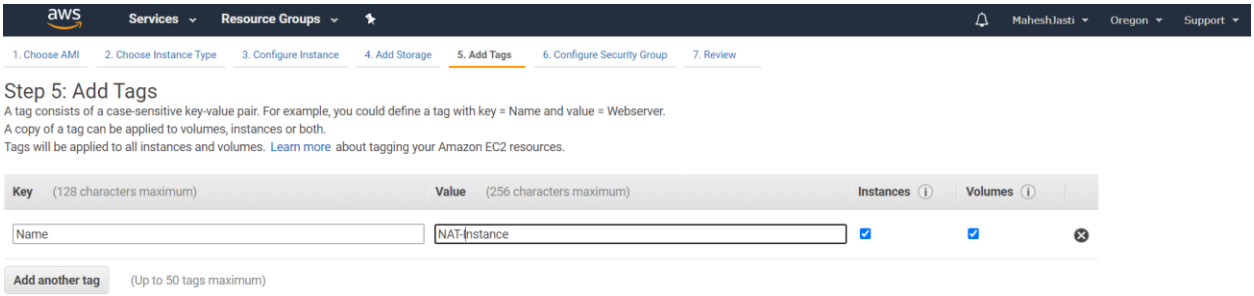
Step#28

Select default T2 Micro and select the VPC ProjectMattermost and PublicSubnet.



Step#29

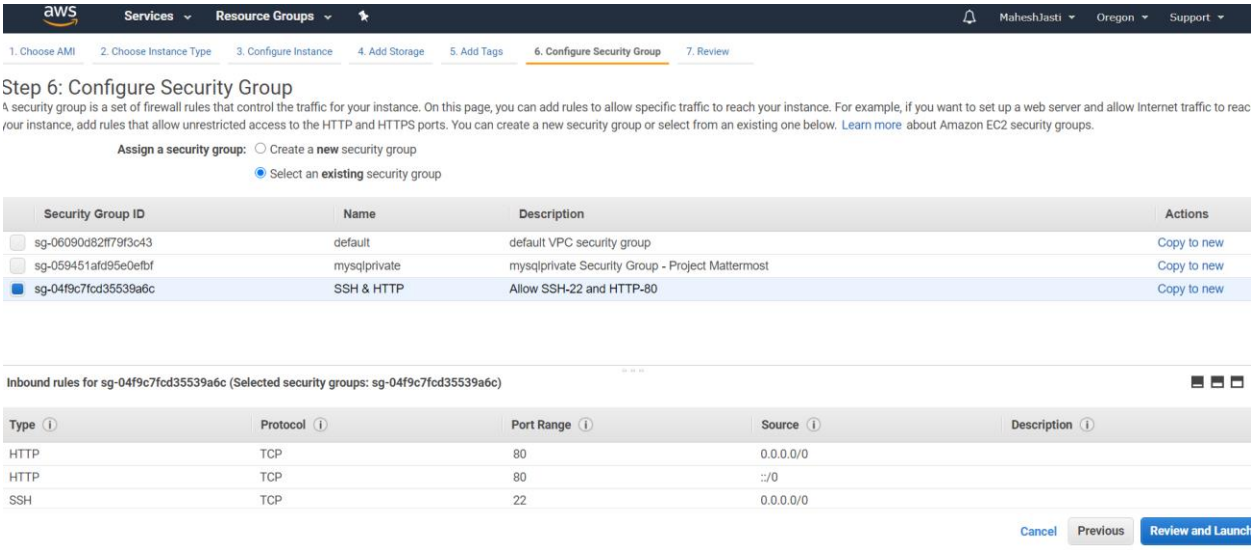
No changes to Add Storage screen..



Step#30

Select the existing Security Group (SSH & HTTP) and launch the instance.

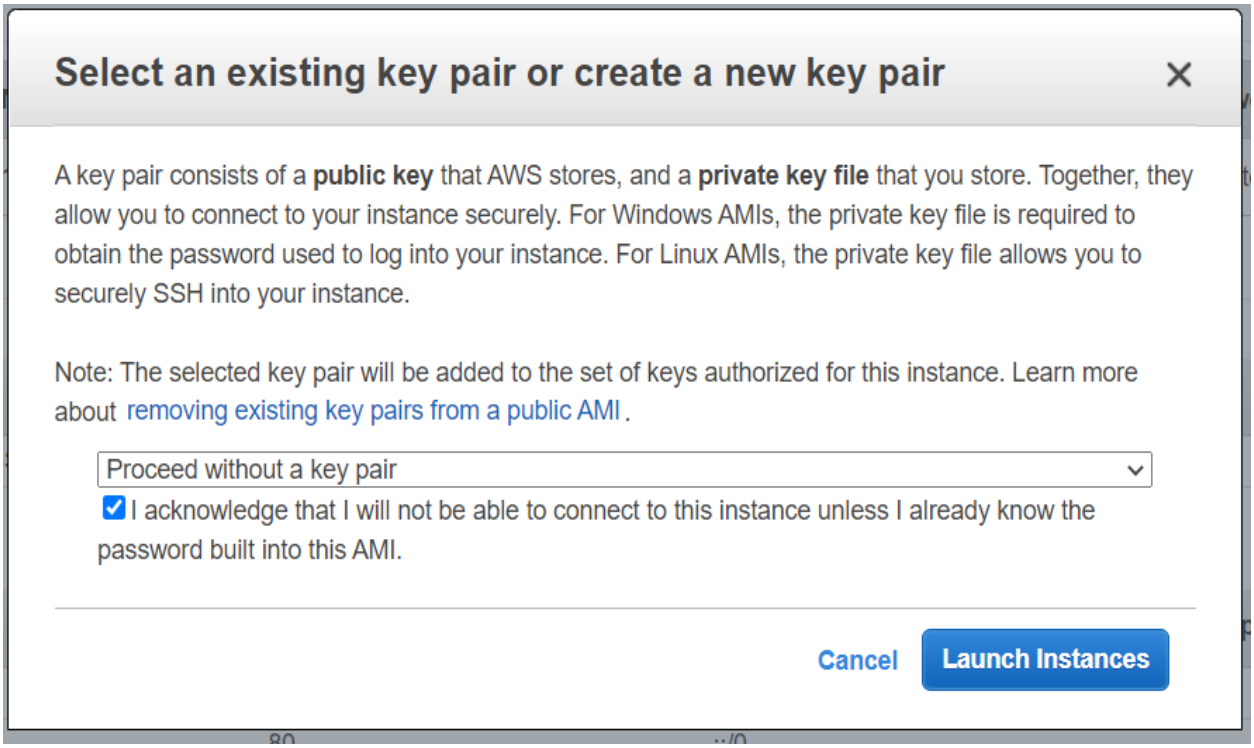
Note: Since it is a NAT instance, we won't login, access or install any software. No need to think about the Security Group and their rules.



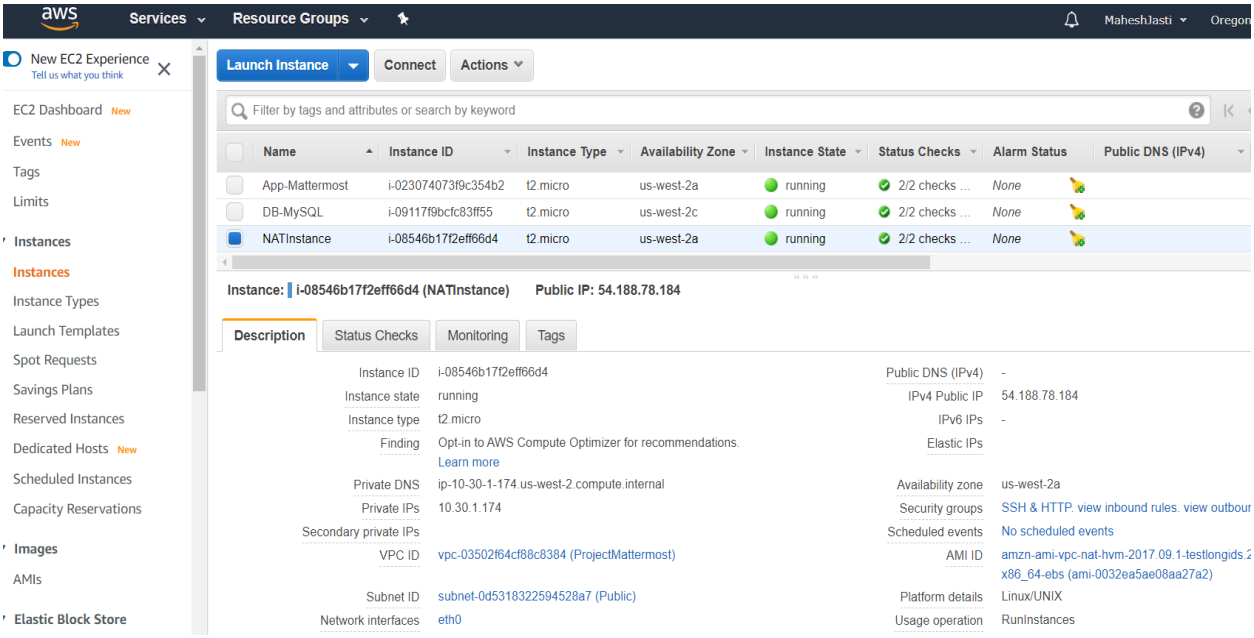
Step#31

Since we won't be logging in or installing into NAT, no need to select any key pair.

And select the drop down values as "Proceed without a key pair", Acknowledge and click on Launch Instances.

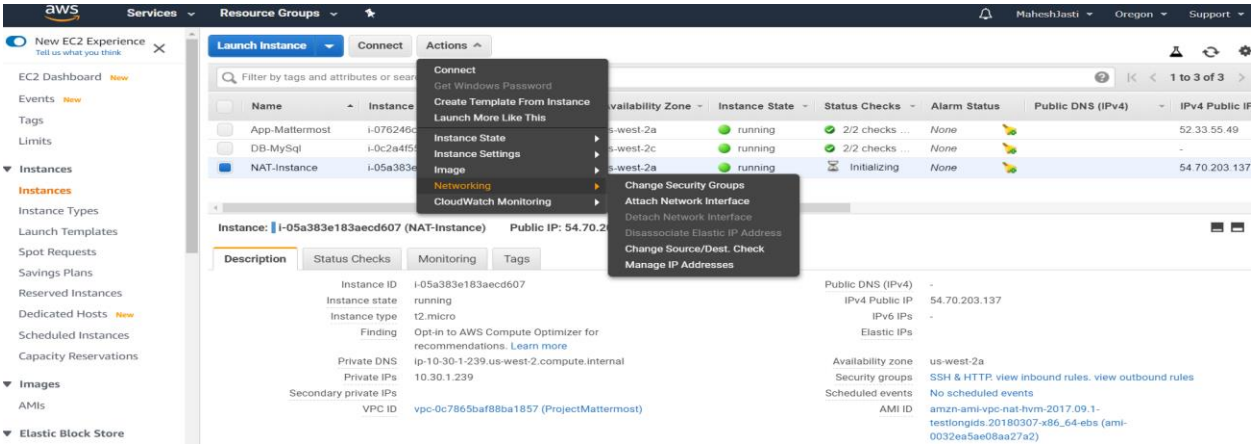


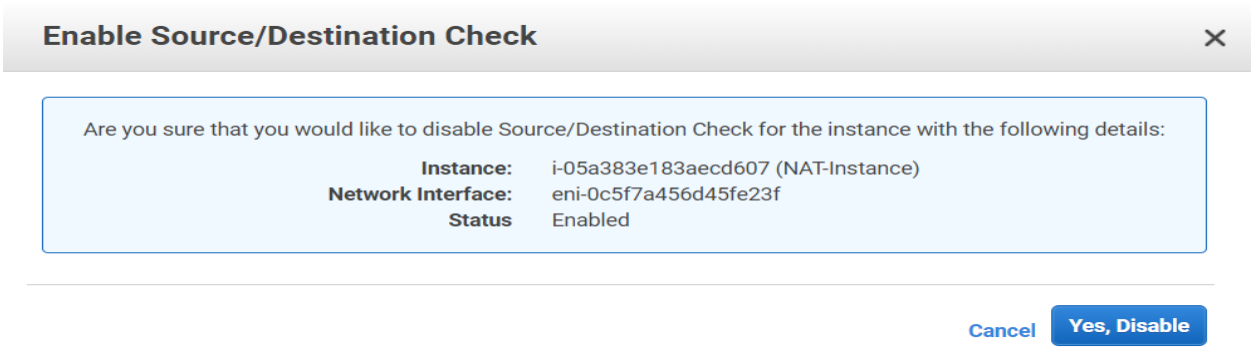
Now NAT instance has been created (Public IP 54.70.203.137 ; Private IP: 10.30.1.239)



Step#32

Now Disable the Change Source/Dest. Check for the NAT Instance..



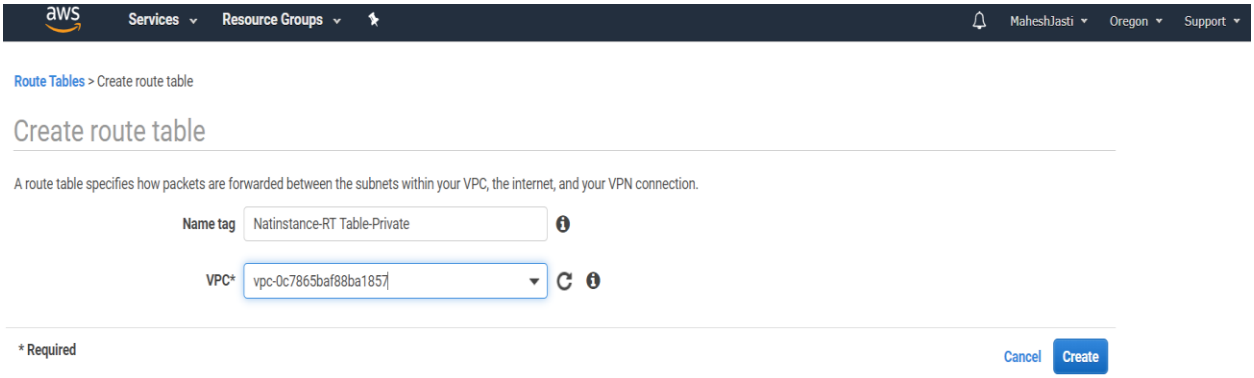


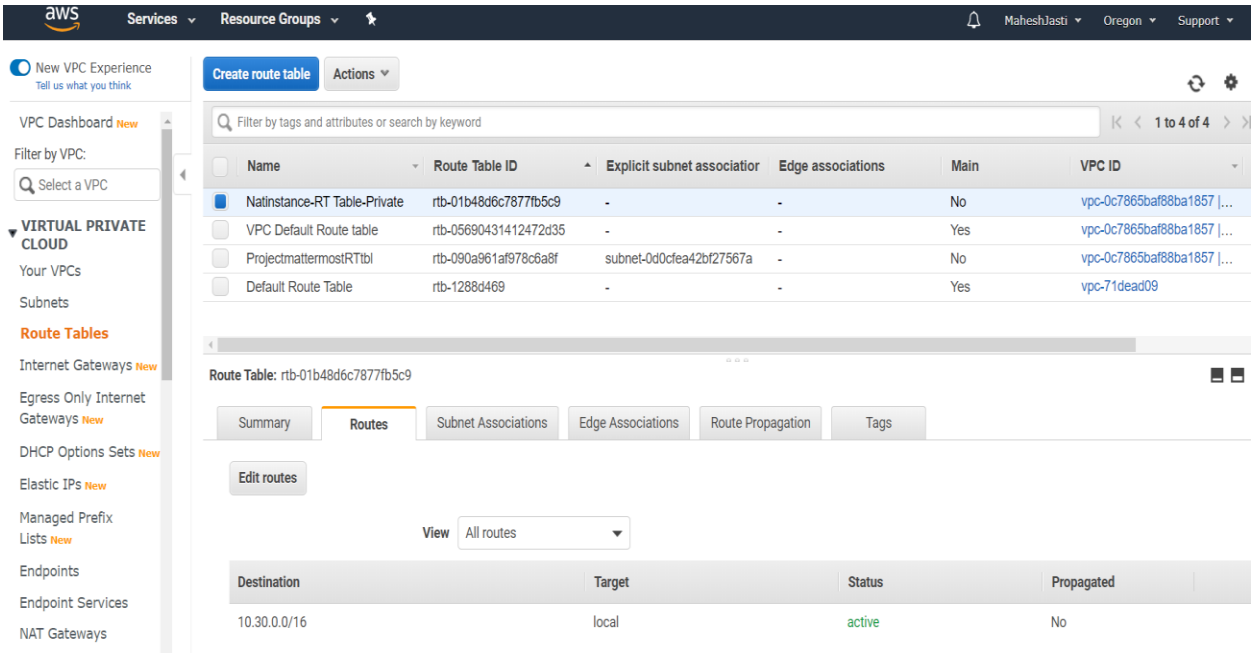
**********Ubuntu EC2 and NAT Instances setup done**********

**********Route table setup for PrivateSubnet **********

Step#33

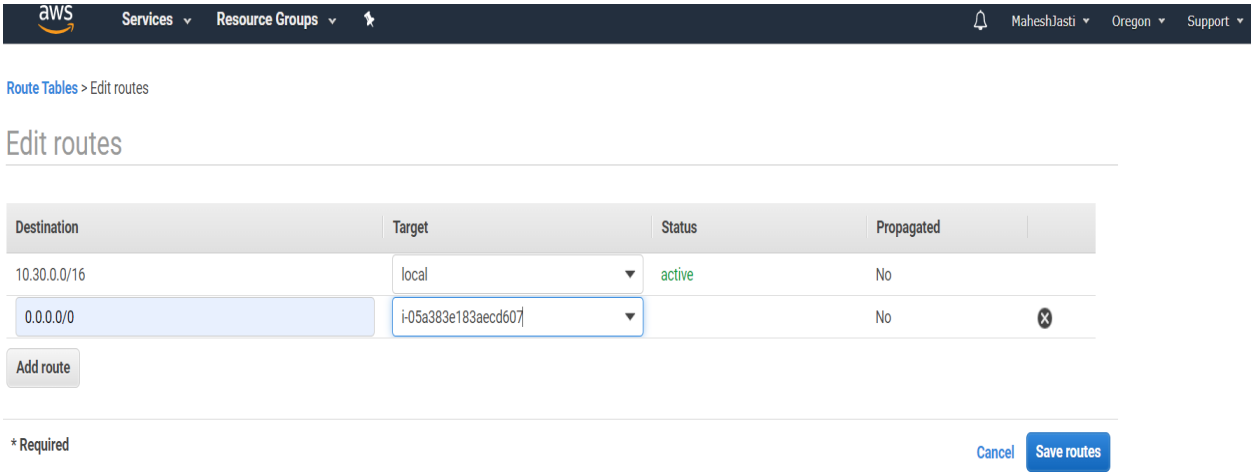
Go to the Route Tables and create a new route table Natinstance-RT Table-Private and map to the PrivateSubnet.

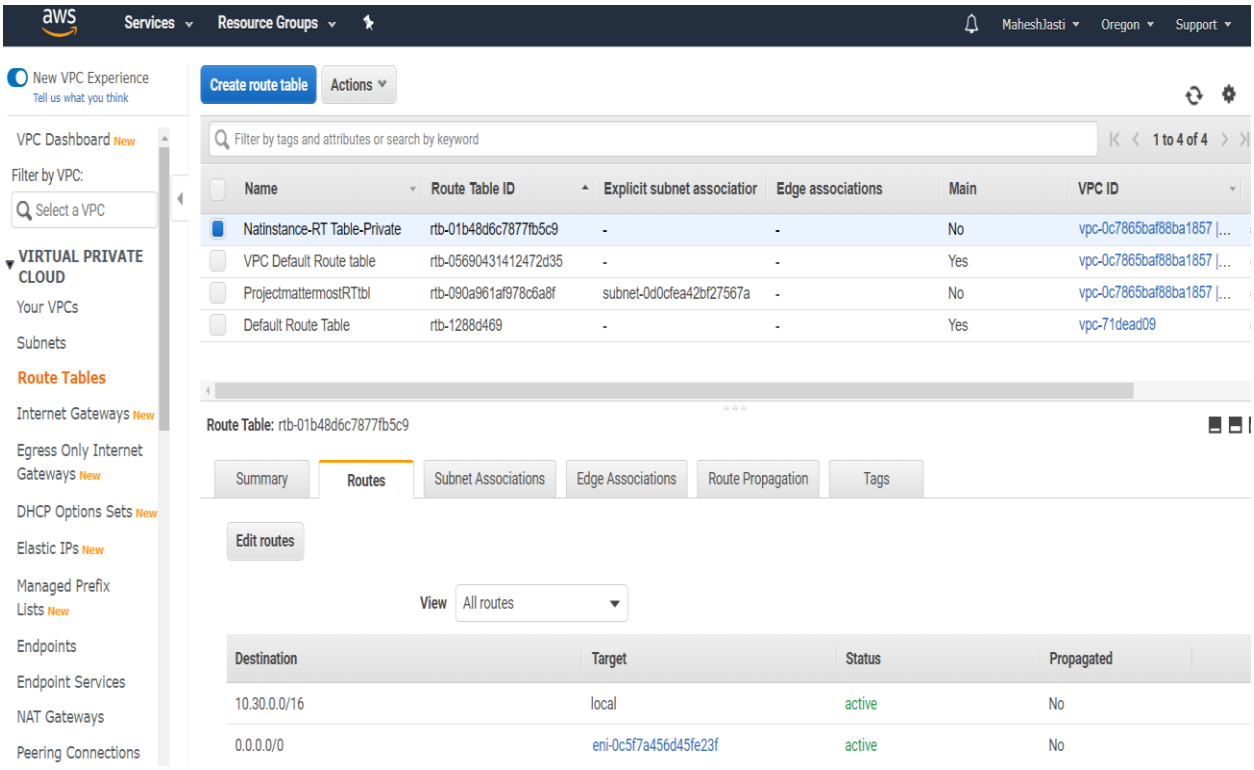




Step#34

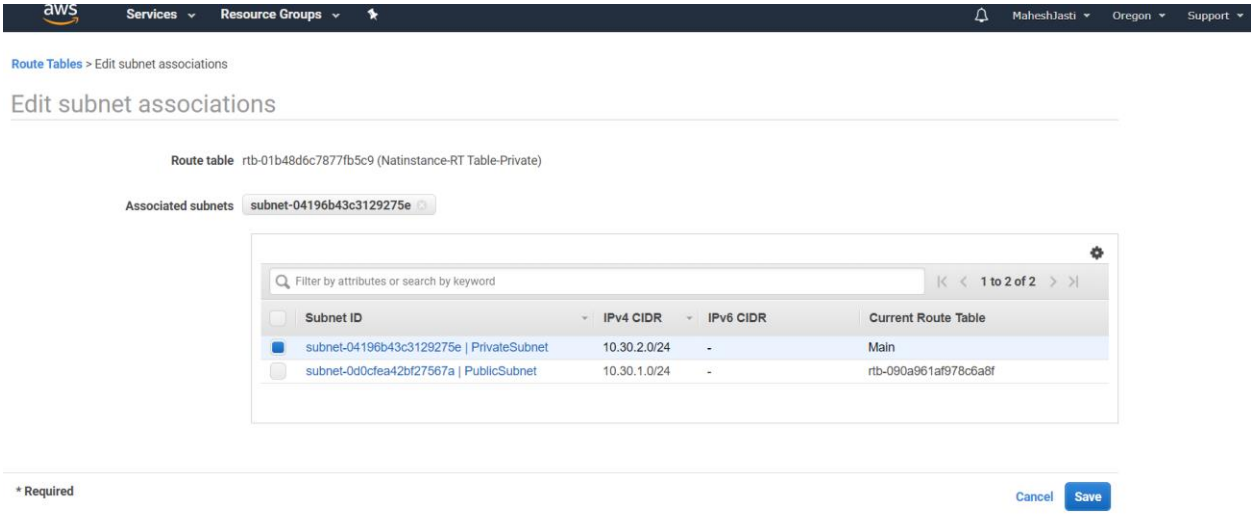
Click on Edit routes and Add route as below and map the target as NAT Instance and click on Save routes.



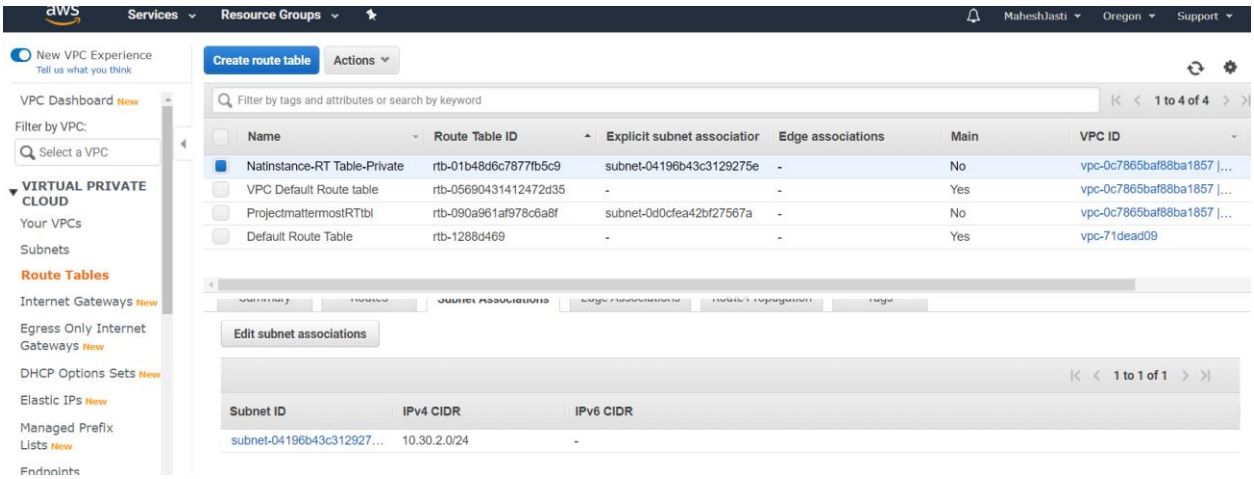


Step#35

Associate the PrivateSubnet in Edit subnet associations tab and click on Save button



Subnet ID associated after clicking on the Save button



**********Route table setup for PrivateSubnet setup done**********

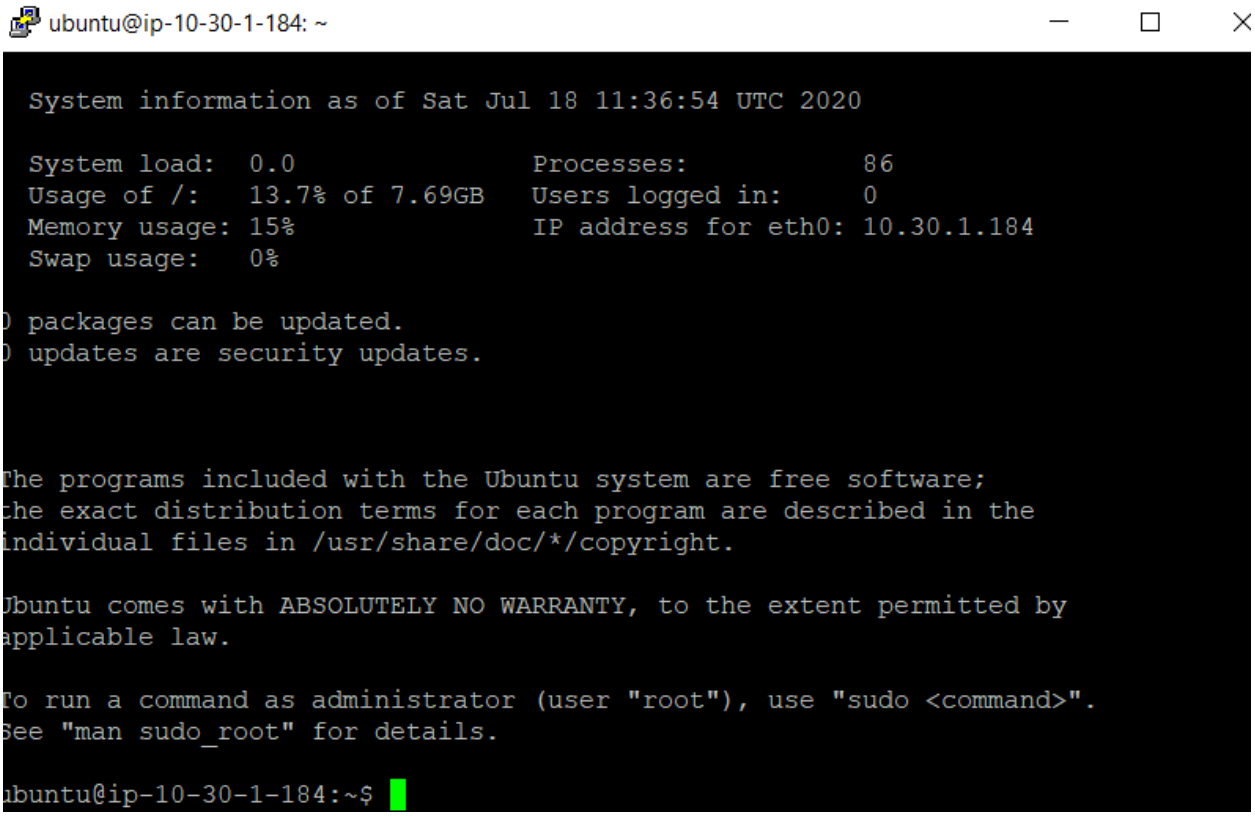
**********EC2 instances – Access check **********

Step#36

Until this we have setup IGWY to Public Subnet and NAT Instance to the Private Subnet.

Now login to the Public Instance thru Putty – ubuntu@52.33.55.49

10.30.1.184

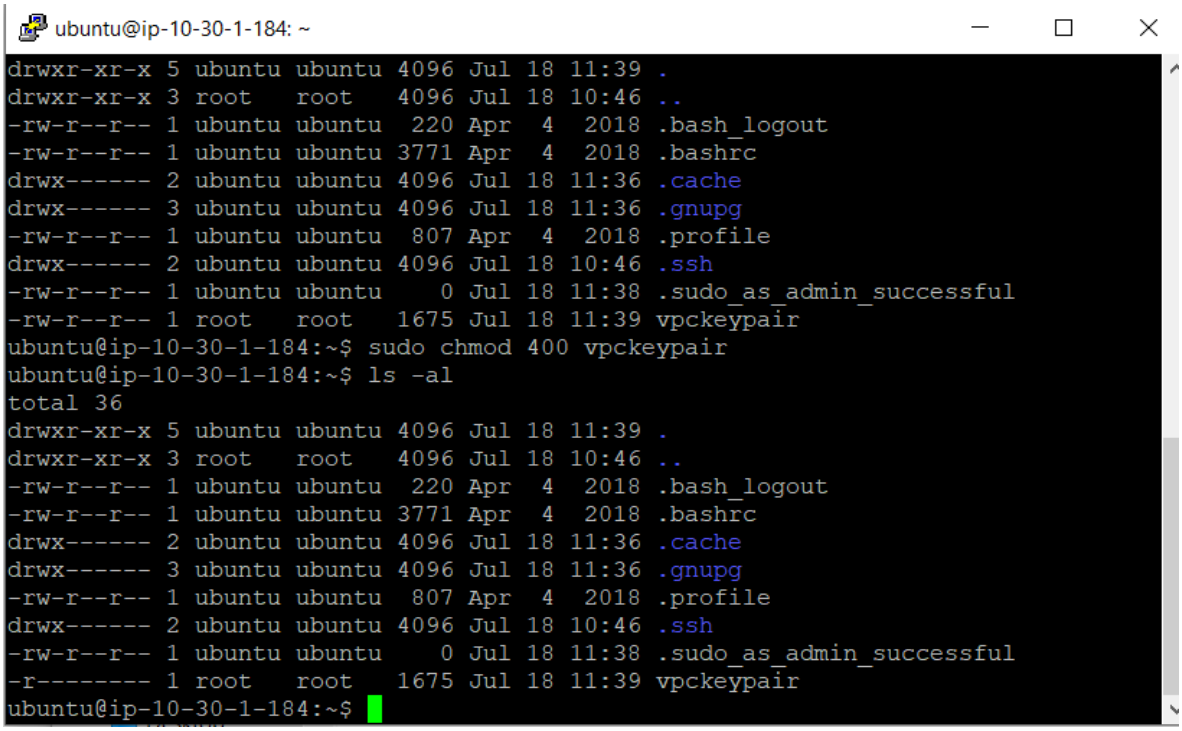


Step#37

We need to login to the 10.30.2.225 Private instance to install the MySQL Server. So we need to SSH from 10.30.1.184. We need the key to login. So create a key file(vpckeypair) and change the permissions of the file to Read only and execute the following commands

sudo nano vpckeypair

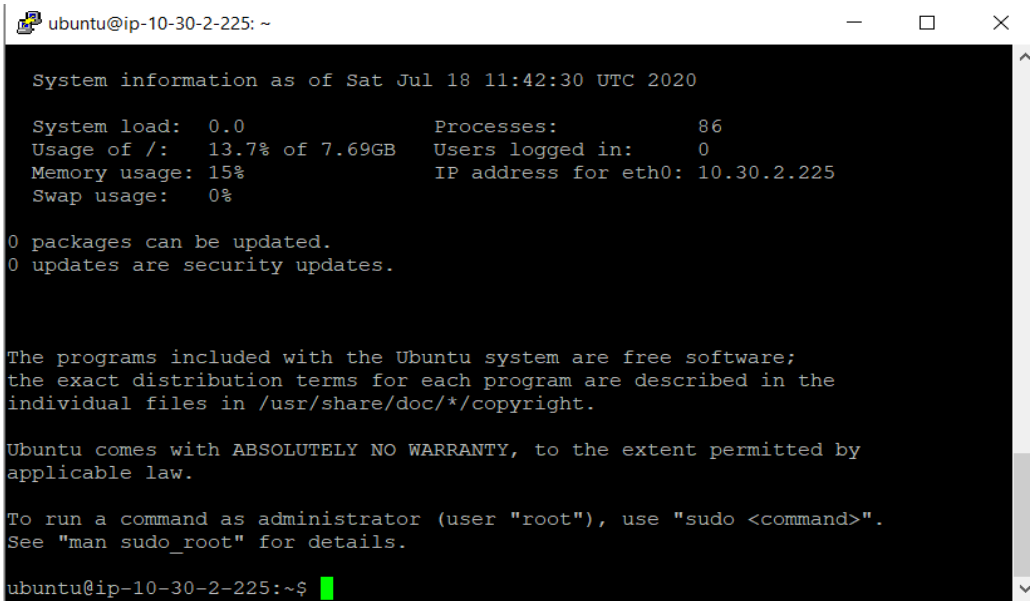
sudo chmod 400 vpckeypair



Step#38

Execute the following command to move to Private instance

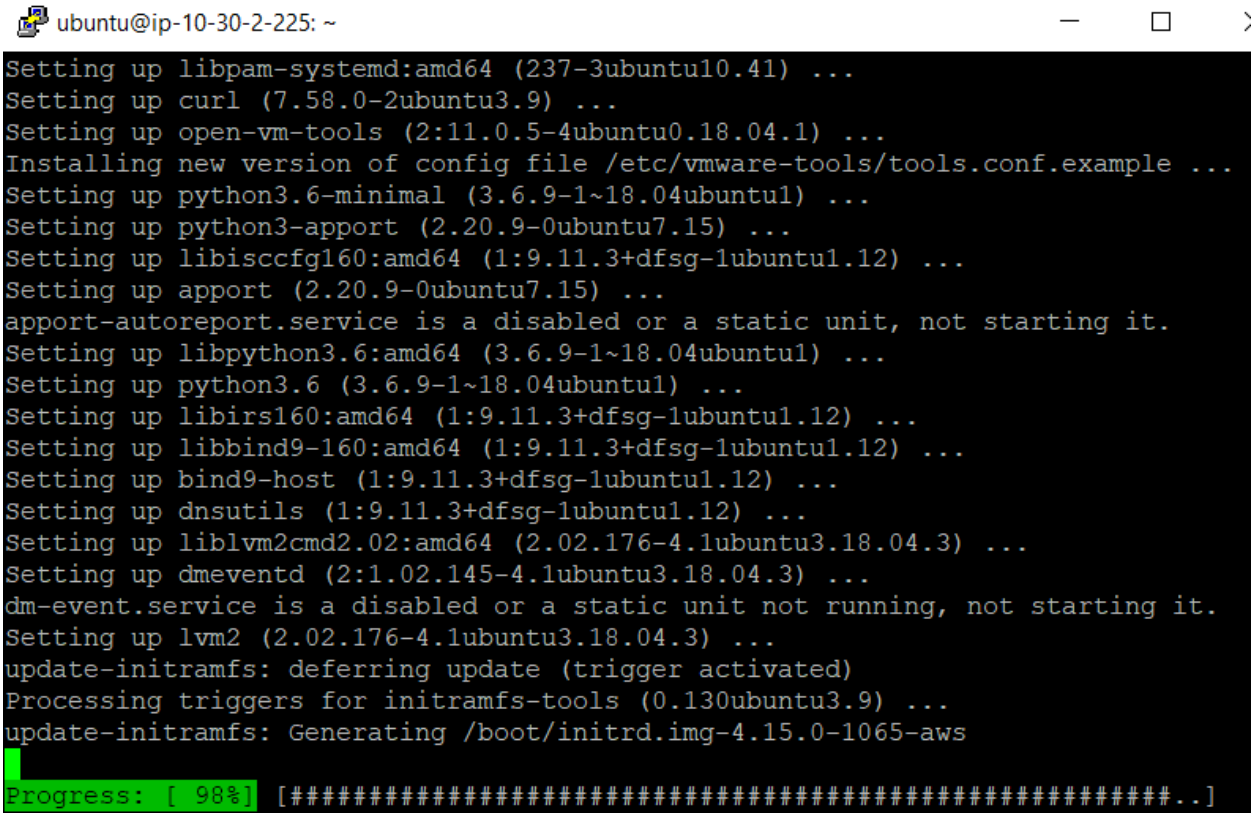
sudo ssh -i vpckeypair ubuntu@10.30.2.225



Step#39

You are into Ubuntu machine (Private Subnet) and execute the following 2 below commands..

sudo apt update and sudo apt upgrade (if you don't upgrade, you will have some problems in installing MySql-Server)



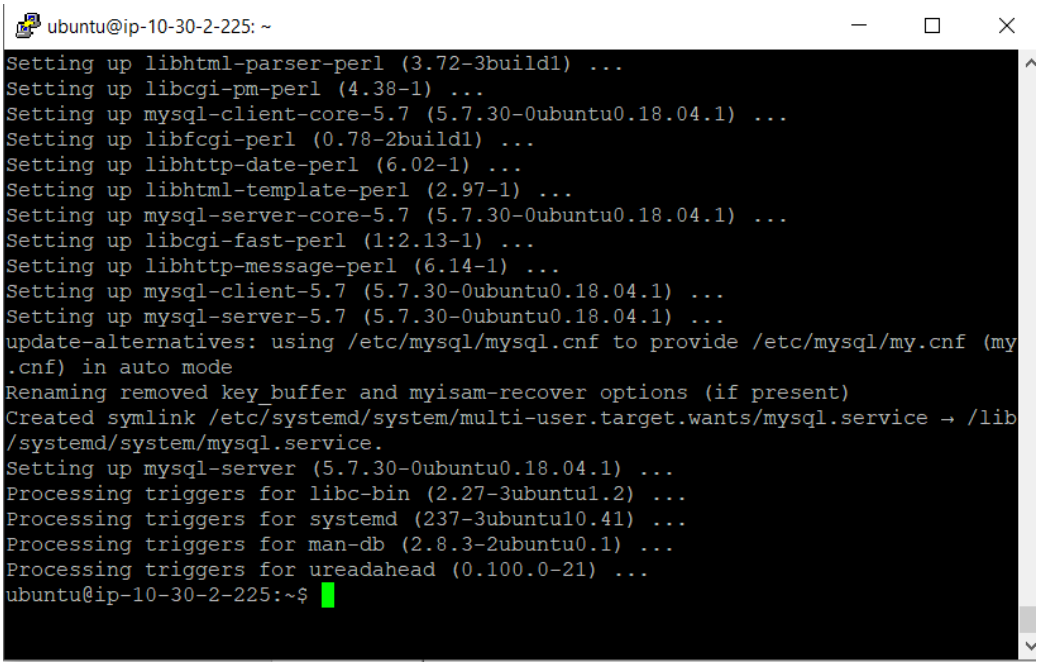
**********EC2 instances – Access check - Completed **********

********Installation of Mysql & Mattermost softwares – Setup *********

Step#40

Install MySQL Server. Use the following command to install.

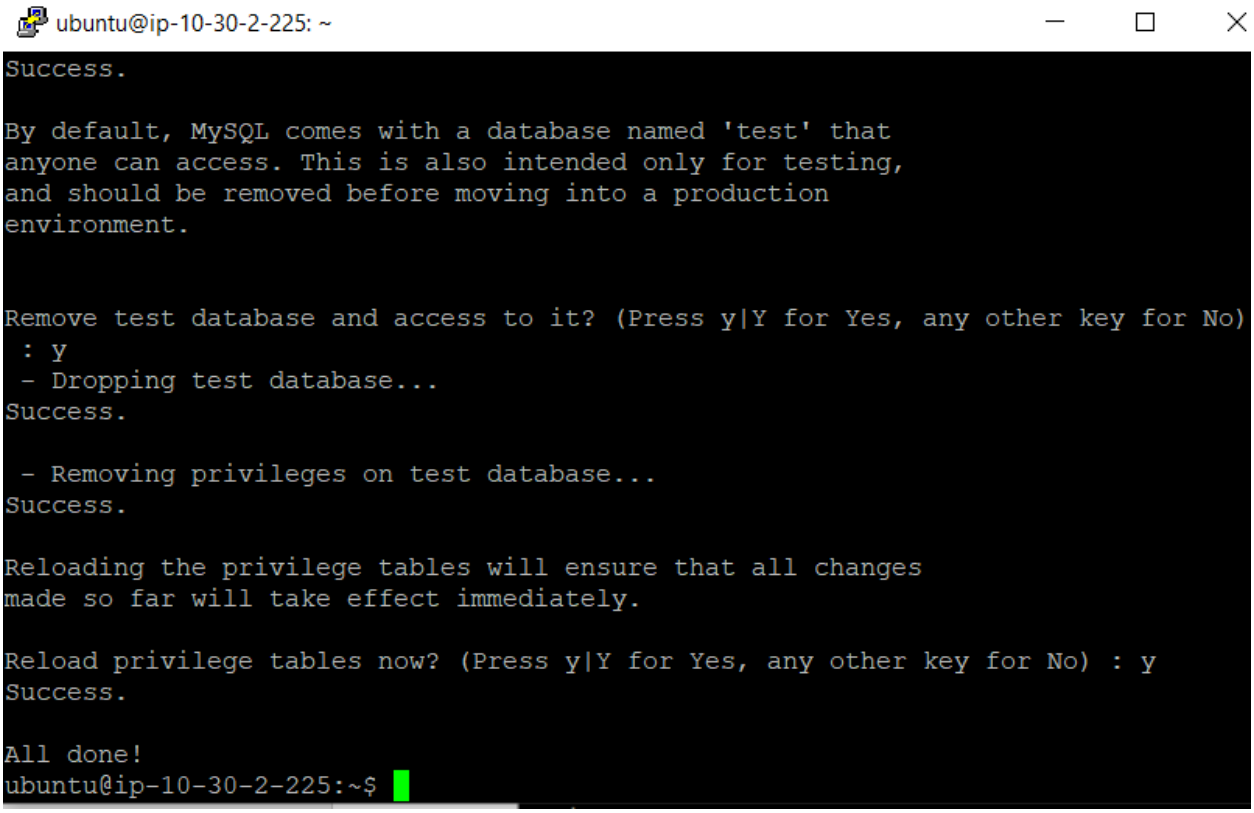
sudo apt install mysql-server



Step#41

Running mysql_secure_installation – To check the password settings and setup root password and admin settings.

sudo mysql_secure_installation (sudo – should be in small letters)



Step#42

sudo mysql

Created an User and the password and also set only the public subnet which has the mattermost app to access the MySQL which is in private subnet.

create user 'mmuser'@'%' identified by 'maheshjast';

create database mattermost;

GRANT ALTER, CREATE, DELETE, DROP, INDEX, INSERT, SELECT, UPDATE ON mattermost.* TO 'mmuser'@'%';

```
ubuntu@ip-10-30-2-225: ~
ubuntu@ip-10-30-2-225:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 5.7.30-0ubuntu0.18.04.1 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create user 'mmuser'@'%' identified by 'maheshjast';
Query OK, 0 rows affected (0.00 sec)

mysql> create database mattermost;
Query OK, 1 row affected (0.00 sec)

mysql> GRANT ALTER, CREATE, DELETE, DROP, INDEX, INSERT, SELECT, UPDATE ON matte
rmost.* TO 'mmuser'@'%';
Query OK, 0 rows affected (0.00 sec)

mysql>
```

Step#43

Exit mysql

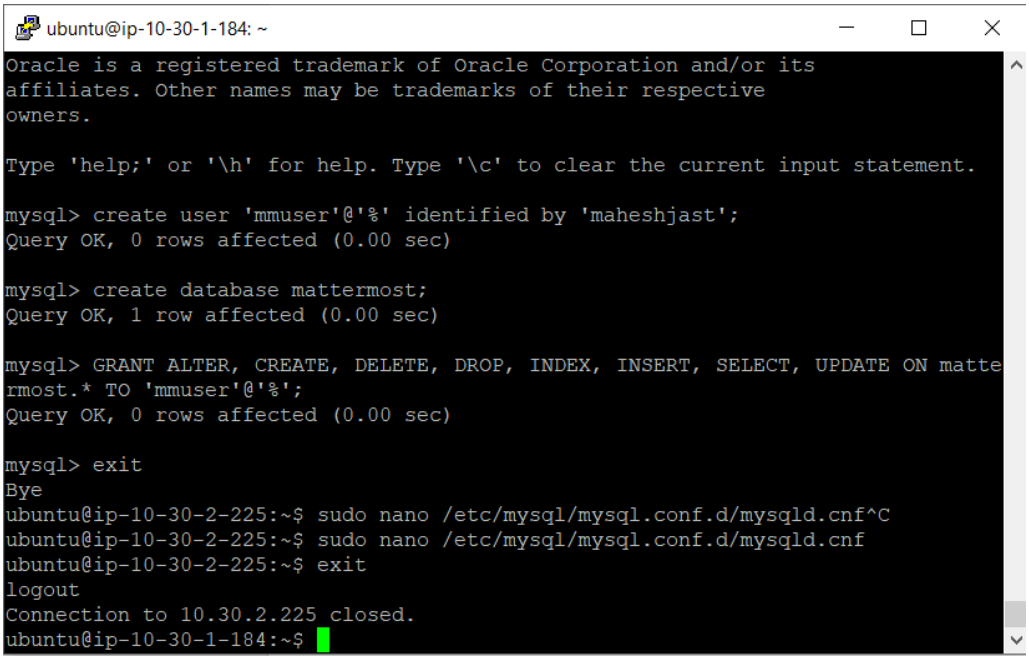
sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf

Comment the bind-address 127.0.0.1

```
ubuntu@ip-10-30-2-225: ~
  GNU nano 2.9.3          /etc/mysql/mysql.conf.d/mysqld.cnf          Modified

user            = mysql
pid-file        = /var/run/mysqld/mysqld.pid
socket          = /var/run/mysqld/mysqld.sock
port            = 3306
basedir         = /usr
datadir         = /var/lib/mysql
tmpdir          = /tmp
lc-messages-dir = /usr/share/mysql
skip-external-locking
#
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
# bind-address          = 127.0.0.1
#
# * Fine Tuning
#
key_buffer_size         = 16M
max_allowed_packet      = 16M
thread_stack            = 192K

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text^T To Spell  ^_ Go To Line
```

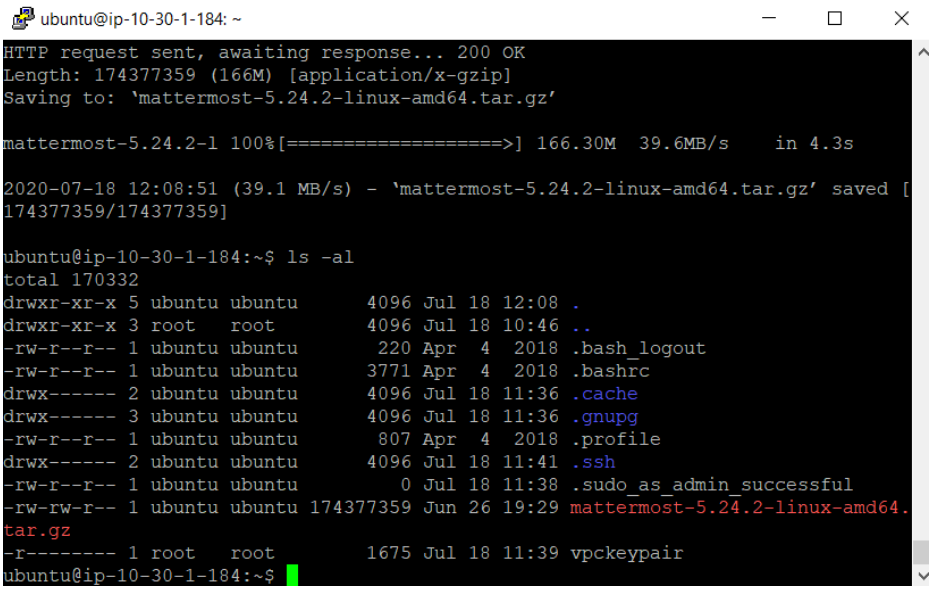


Step#44

Exit the private instance and install the mattermost in the public instance by using the following command

sudo apt update and sudo apt upgrade

wget https://releases.mattermost.com/5.24.2/mattermost-5.24.2-linux-amd64.tar.gz



Step#45

And follow the commands to configure the Mattermost application….

tar -xvzf mattermost*.gz

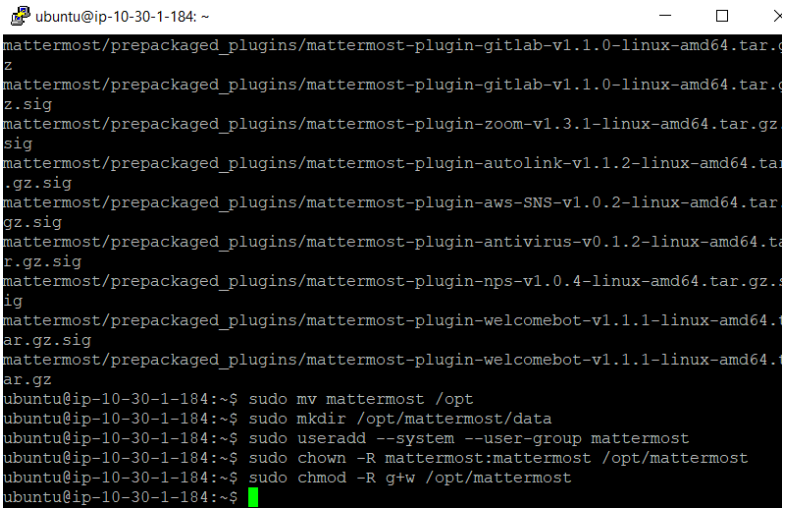
sudo mv mattermost /opt

sudo mkdir /opt/mattermost/data

sudo useradd --system --user-group mattermost

sudo chown -R mattermost:mattermost /opt/mattermost

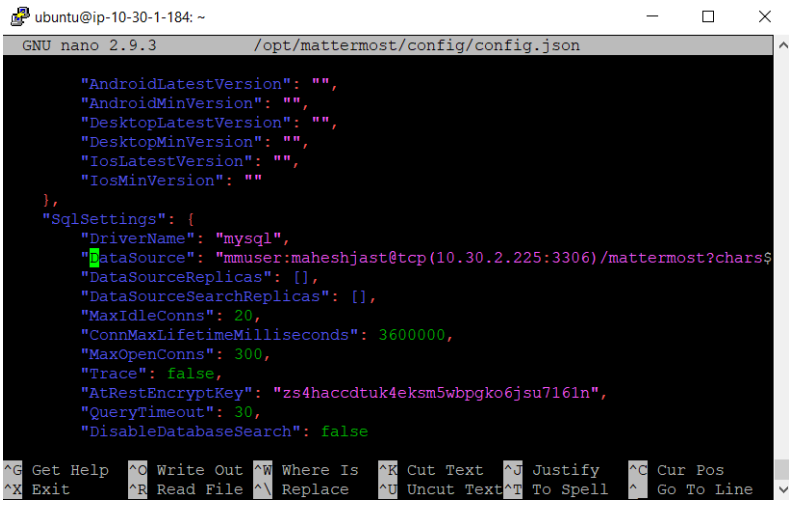
sudo chmod -R g+w /opt/mattermost



Step#46

Setup the database drive and update the datasource details with the user details that has been created for mysql

sudo nano /opt/mattermost/config/config.json



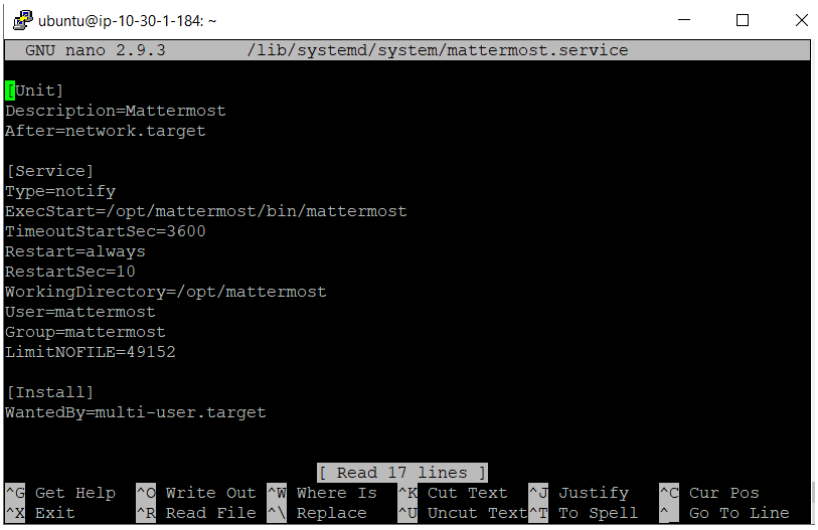
Step#47

Execute the following commands to setup the mattermost service

cd /opt/mattermost

sudo -u mattermost ./bin/mattermost

sudo nano /lib/systemd/system/mattermost.service (update the file as per the comments)



sudo systemctl daemon-reload

sudo systemctl status mattermost.service

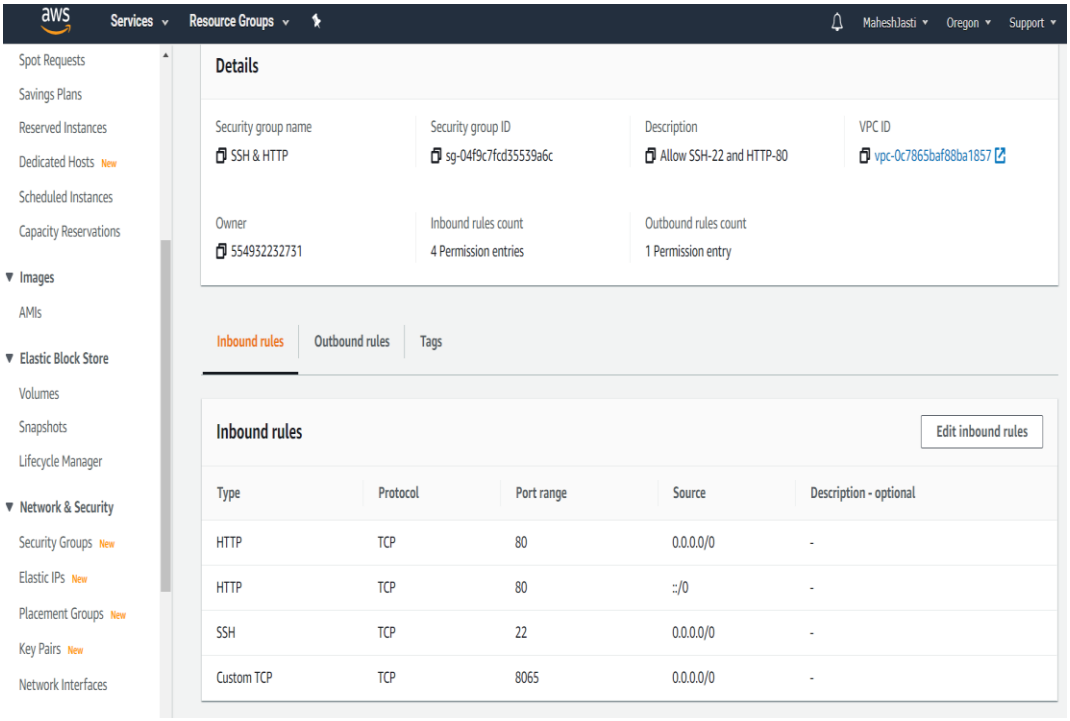
sudo systemctl start mattermost.service

sudo systemctl enable mattermost.service

Step#47

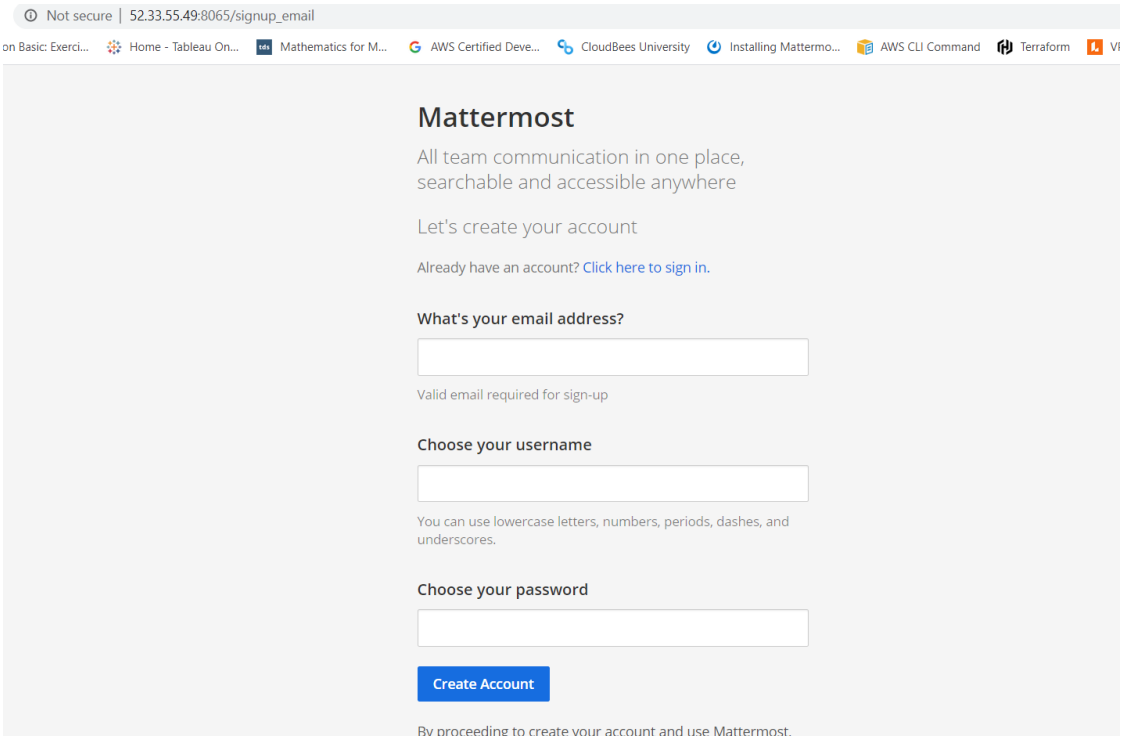
Now update the security group for App-mattermost EC2 instance(Bastion host) with the following

Open the TCP for the port 8065. This is very important step.



Step#48

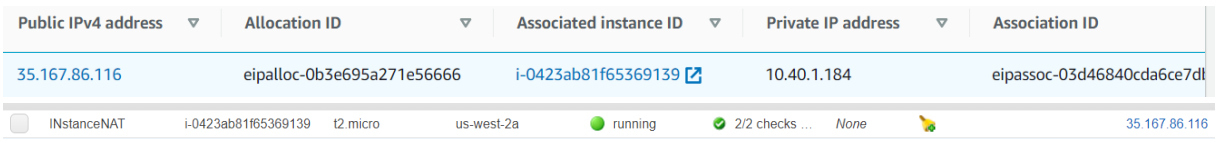
Now open a browser and go to http://52.33.55.49:8065/



*******Installation of Mysql & Mattermost softwares – Setup completed******

Section 6: Lessons learnt / Observations

- Security Groups setup is very important to allow / deny the traffic. Due to 1 small wrong setting in SG, I literally googling for almost 10hours on mattermost community.
- Changes in Security Group(s) will take immediate effect to the instances available in the respective VPCs
- **Security Group operates at instance level and supports allow rules only.**
- **Instances associated with Security Group can't talk to each other unless you add rules allowing in the Security Group**
- VPC and Subnets setup can remain available instead of deleting but where as EC2 instances should be stopped.
- VPC automatically comes with a main route table that can modify
- A subnet can be associated with only one network ACL at a time. If you associate a new one, then the previous subnet association will be removed.
- **Multiple subnets can be associated with network ACLs.**
- **Can associate multiple subnets with the same route table.**
- **A subnet can only be associated with one route table.**
- Since we have to install the softwares in Ubuntu machines, I searched for Ubuntu NAT instances (which has name as natt) and created a setup but it didn't work. As per AWS, only LINUX NAT instances should be used. **(AWS recommends the instances with the name amzn-ami-vpc-nat as NAT instances only)**
- **As we know NAT instance doesn't require elastic IP unless the auto assign IP address option is not checked. If the auto assign IP option is checked and instance initiated then if we create the elastic IP, AWS assign the same public IP which is there for EC2 NAT instance.**



- Since we are installing mattermost and mysql in public and private EC2 machines respectively, we need to make sure we point the correct IP address where mysql is installed in the config.json file
- Proper naming convention should be used for EC2 names, Security Groups, Subnet names. Since we would be creating multiple of these, I sometimes confused especially with the SG names.
- Create the flow steps based on the architecture which helps in the setup creation fastly.

- **Can connect to the private instance (mysql server) in VPC2 and access the mattermost application from public instance which is in VPC1 (10.40.1.0 – VPC1 <> 10.60.1.0 – VPC2) by using VPC peering connection.**

- Route tables should be updated with respective to the VPC peering ID for both VPCs