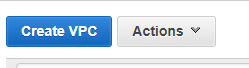
Designing a Three-Tier Architecture in AWS

Step1: Vpc:- Amazon Virtual Private Cloud (Amazon VPC) enables you to launch AWS resources into a virtual network that you've defined. This virtual network closely resembles a traditional network that you'd operate in your own data center, with the benefits of using the scalable infrastructure of AWS.

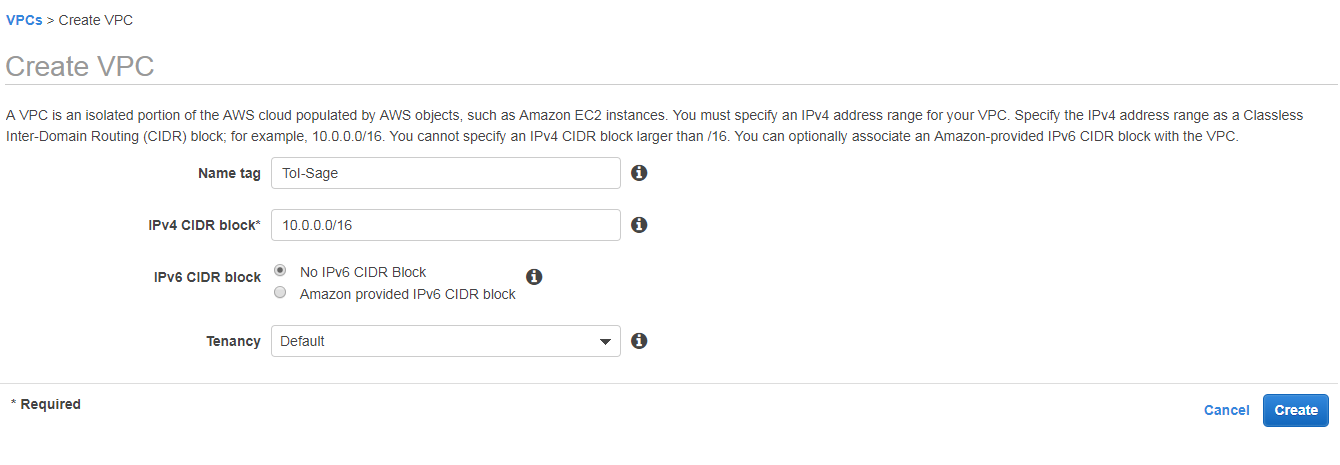
First create one Vpc. Now type vpc And click on your Vpc's.



Now click on Create Vpc.



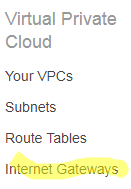
After That give the name,IpVCIDR block. Like below screen shot.



Now click On Create.

Step2:- internet gateway:- An **internet gateway** is a horizontally scaled, redundant, and highly available VPC component that allows communication between instances in your VPC and the **internet**. It therefore imposes no availability risks or bandwidth constraints on your network traffic.

Now click on internet gateway.



Now click on create internet gateway.



Now give the Name. like below screen shot.



Now click on create.

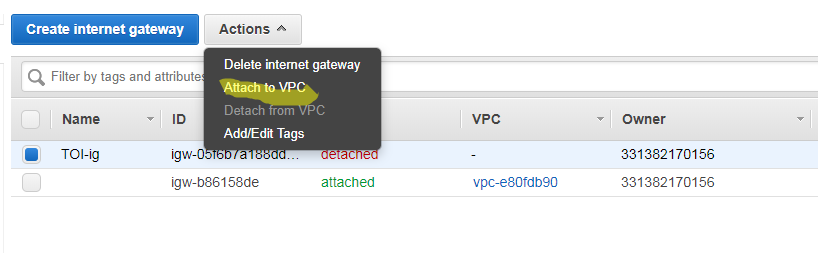
We need to attach our VPC to the internet gateway. To do that:

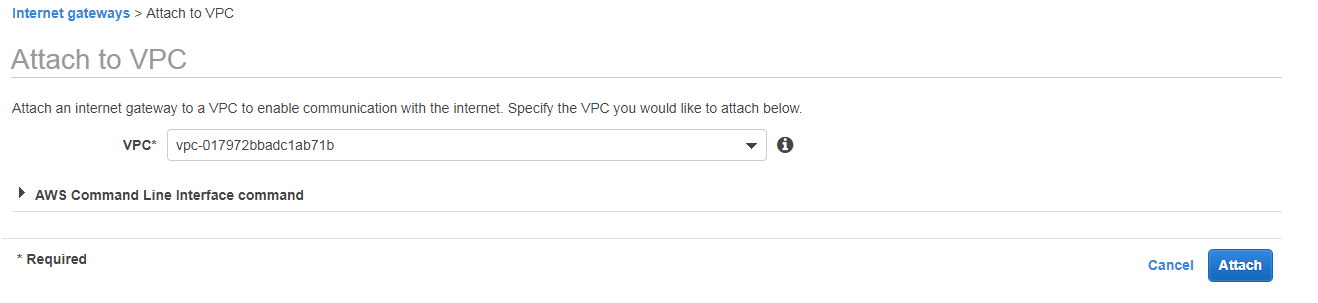
**a.** we select the internet gateway

**b.** Click on the **Actions** button and then select Attach to VPC.

**c.** Select the VPC to attach the internet gateway and click **Attach**

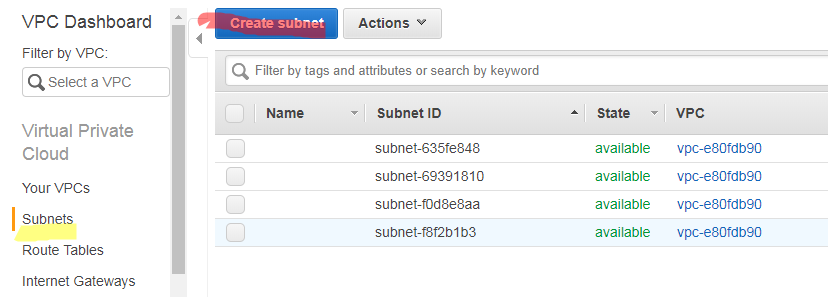
**Like below screen shots.**





Step3:- **Create 4 Subnets:**

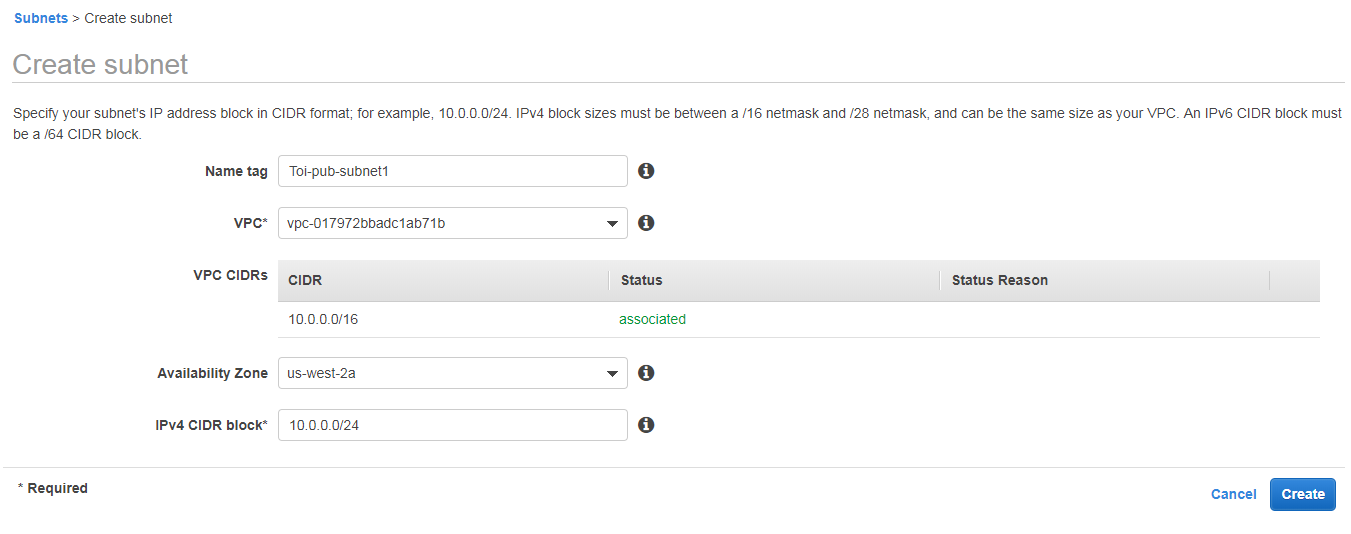
**Now click on subnets. After that click on create subnet.**



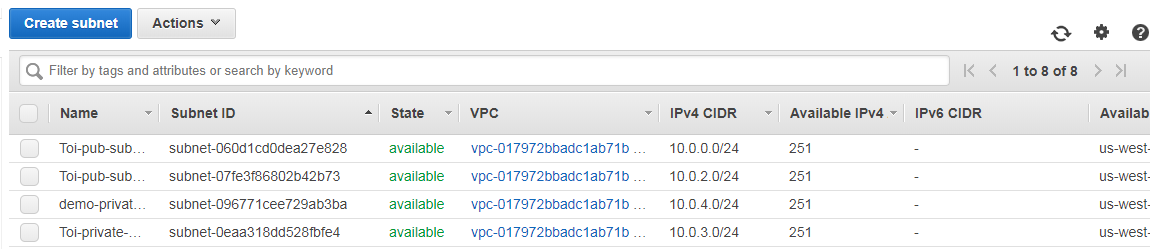
The subnet is a way for us to group our resources within the VPC with their IP range. A subnet can be public or private. EC2 instances within a public subnet have public IPs and can directly access the internet while those in the private subnet does not have public IPs and can only access the internet through a [NAT](https://docs.aws.amazon.com/vpc/latest/userguide/vpc-nat-gateway.html) gateway.

For our setup, we shall be creating the following subnets with the corresponding IP ranges.

* Toi-public-subnet-1 | CIDR (10.0.1.0/24) | Availability Zone (us-west-2a)
* Toi-public-subnet-2 | CIDR (10.0.2.0/24) | Availability Zone (us-west-2b)
* Toi-private-subnet-3 | CIDR (10.0.3.0/24) | Availability Zone (us-west-2a)
* Toi-private-subnet-4 | CIDR(10.0.4.0/24) | Availability Zone (us-west-2b)

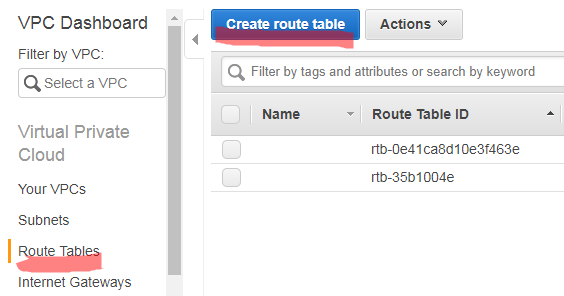


Like below 4 subnets.



**Step4: Create two Route Tables:-**Route tables is a set of rule that determines how data moves within our network. We need two route tables; private route table and public route table. The public route table will define which subnets that will have direct access to the internet ( ie public subnets) while the private route table will define which subnet goes through the NAT gateway (ie private subnet).

Now click on Route table After that click on Create Route table.

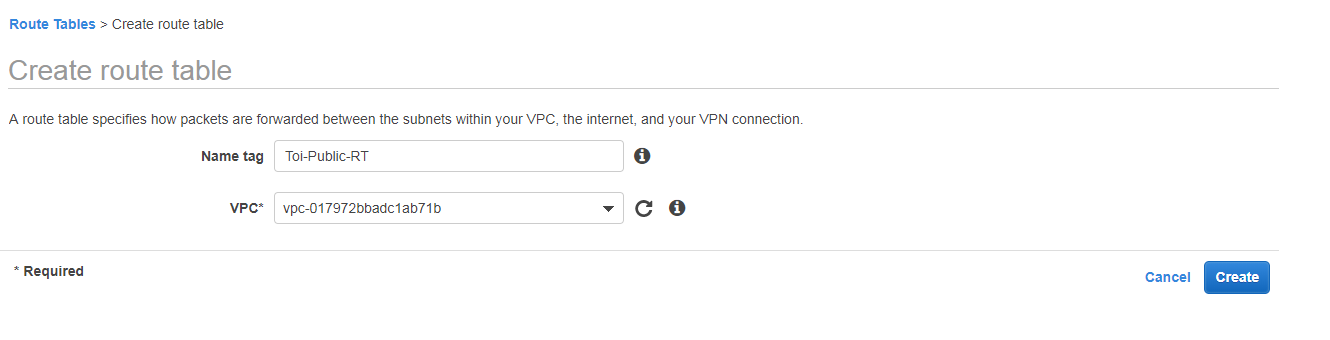


Now we have to create two Route tables.

1)Toi-public-Rt

2)Toi-private-Rt

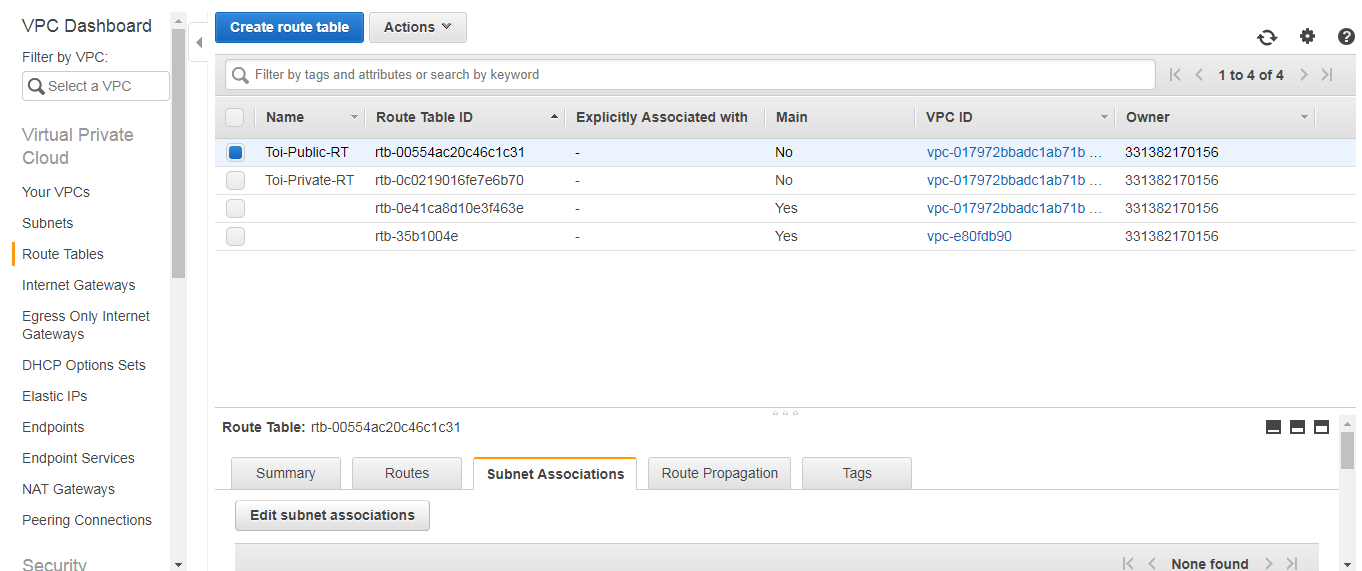
In that give name, select vpc and click on create.



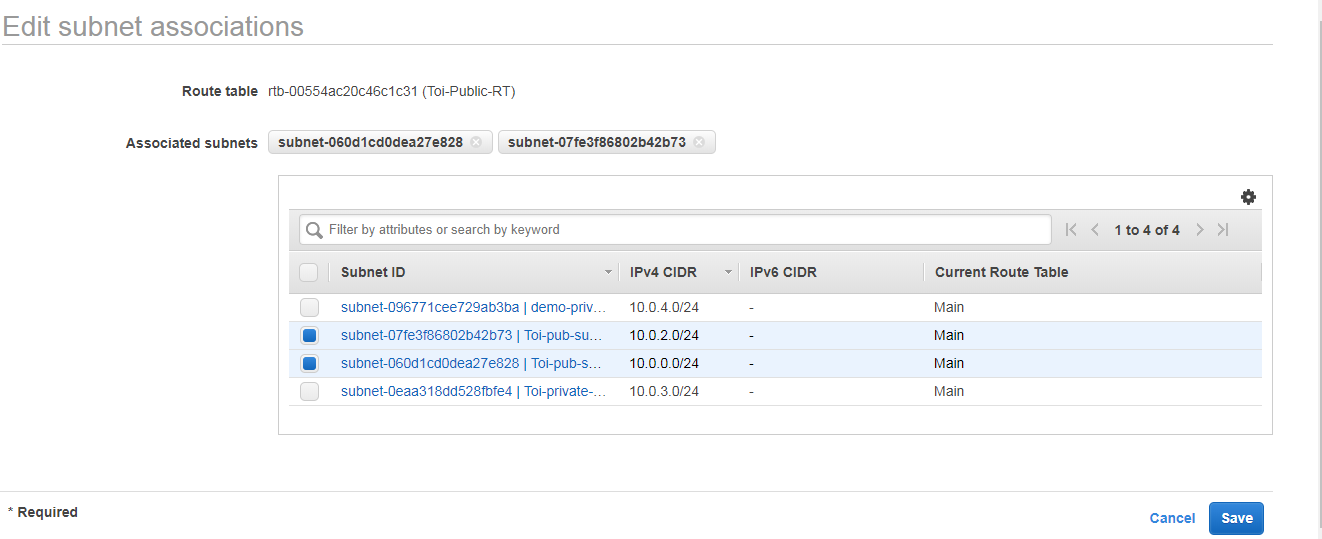


The public and the private subnet needs to be associated with the public and the private route table respectively.

To do that, we select the route table and then choose the **Subnet Association** tab.

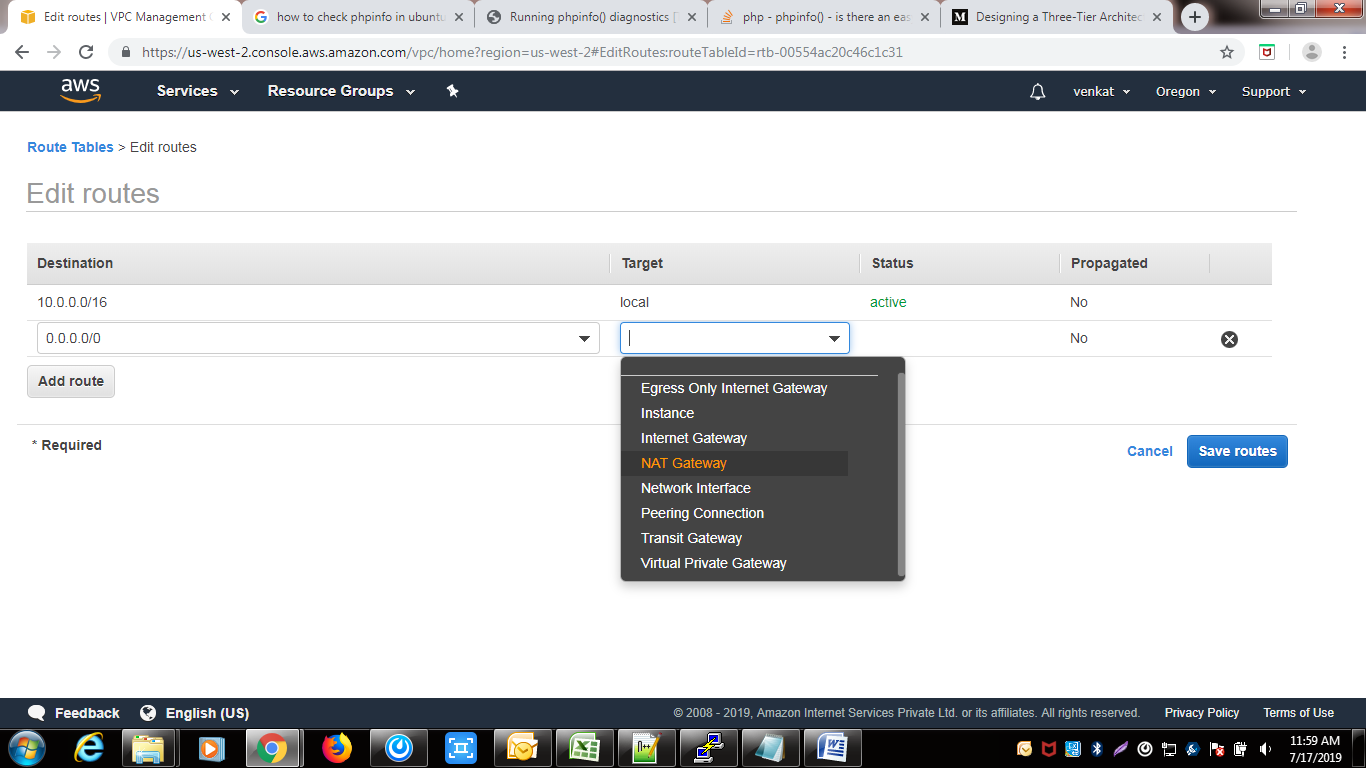


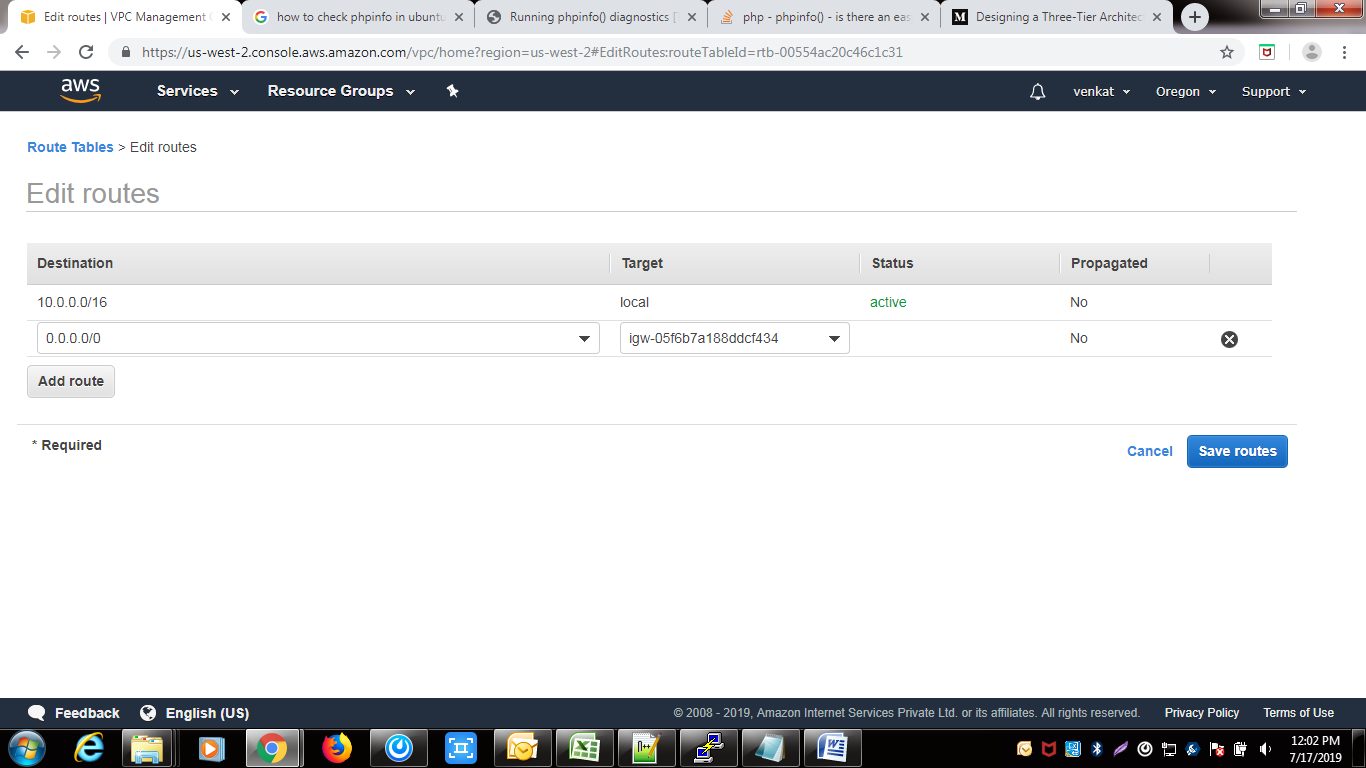
Now click on Edit subnet associations. And select our public subnets. After that click on save.



We also need to route the traffic to the internet through the **internet gateway**for our public route table.

To do that we select the public route table and then choose the **Routes** tab. The rule should be similar to the one shown below:

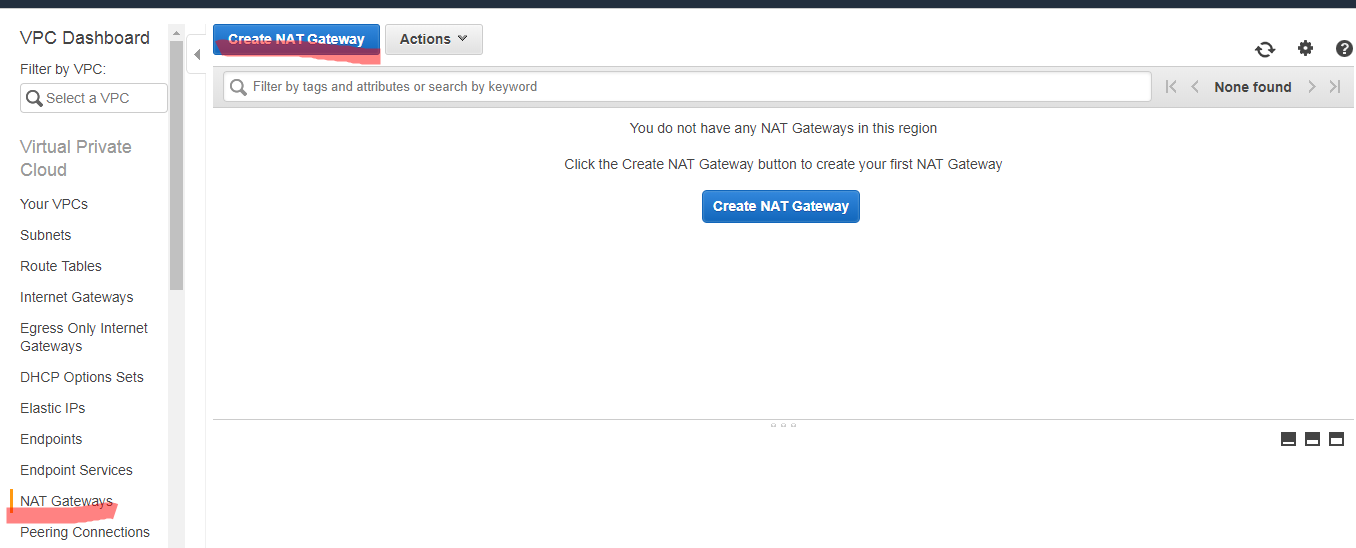




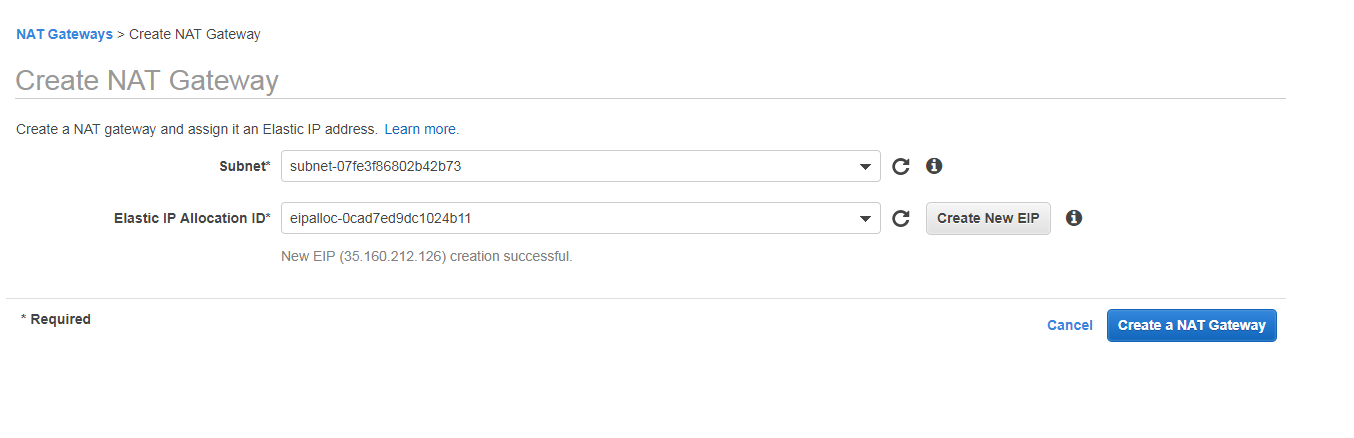
Step5: Create the NAT GATEWAY

The NAT gateway enables the EC2 instances in the private subnet to access the internet. The NAT Gateway is an AWS managed service for the NAT instance. To create the NAT gateway, navigate to the NAT Gateways page, and then click on the **Create NAT Gateway.**

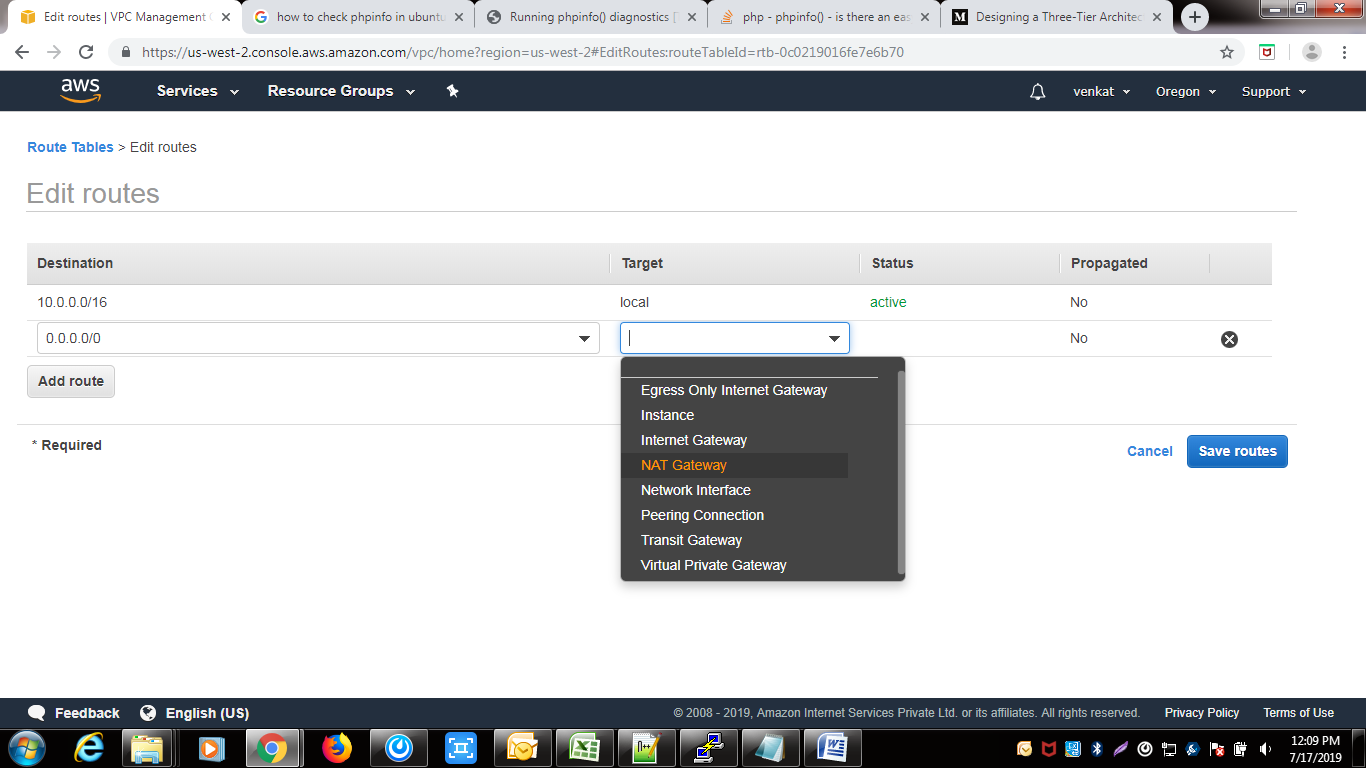
Please ensure that you know the Subnet ID for the **Toi-public-subnet-2.**This will be needed when creating the NAT gateway.

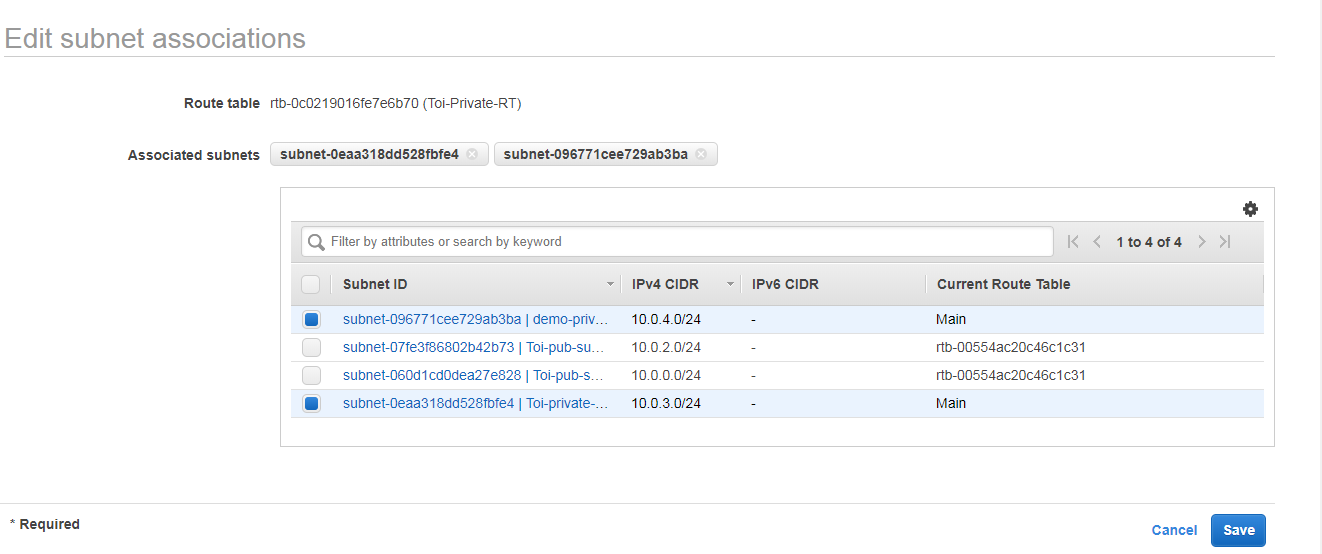


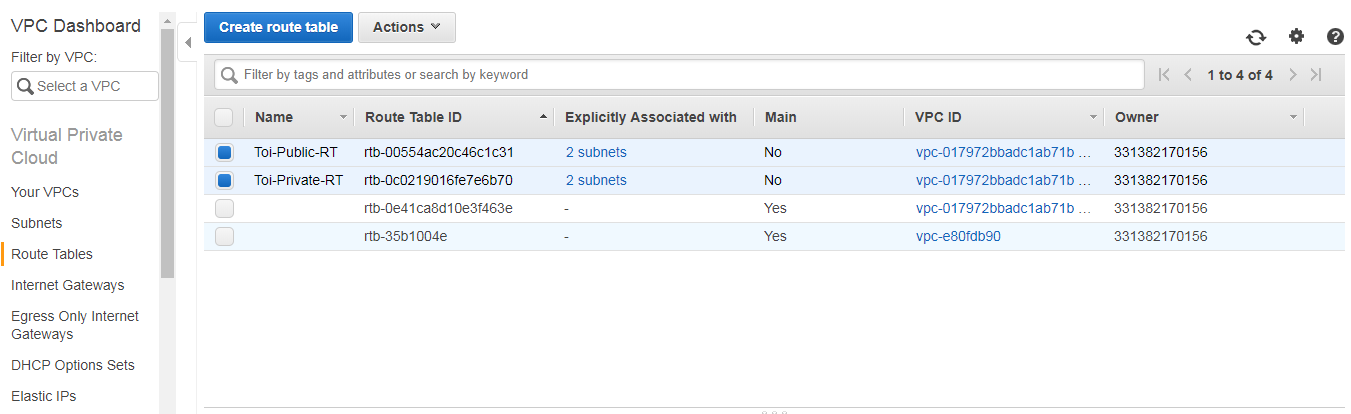
Now Click on Create Nat gate Way. and select subnet and click on create new EIP. and click on Create nat gateway.



Now that we have the NAT gateway, we are going to edit the private route table to make use of the NAT gateway to access the internet.



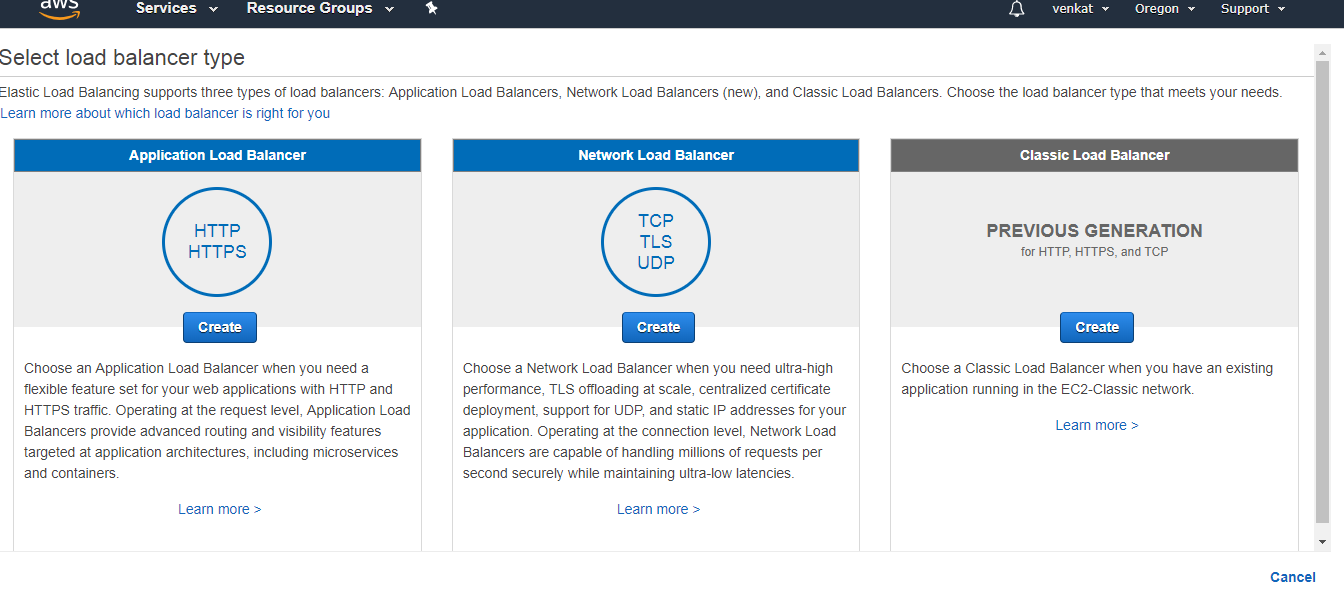




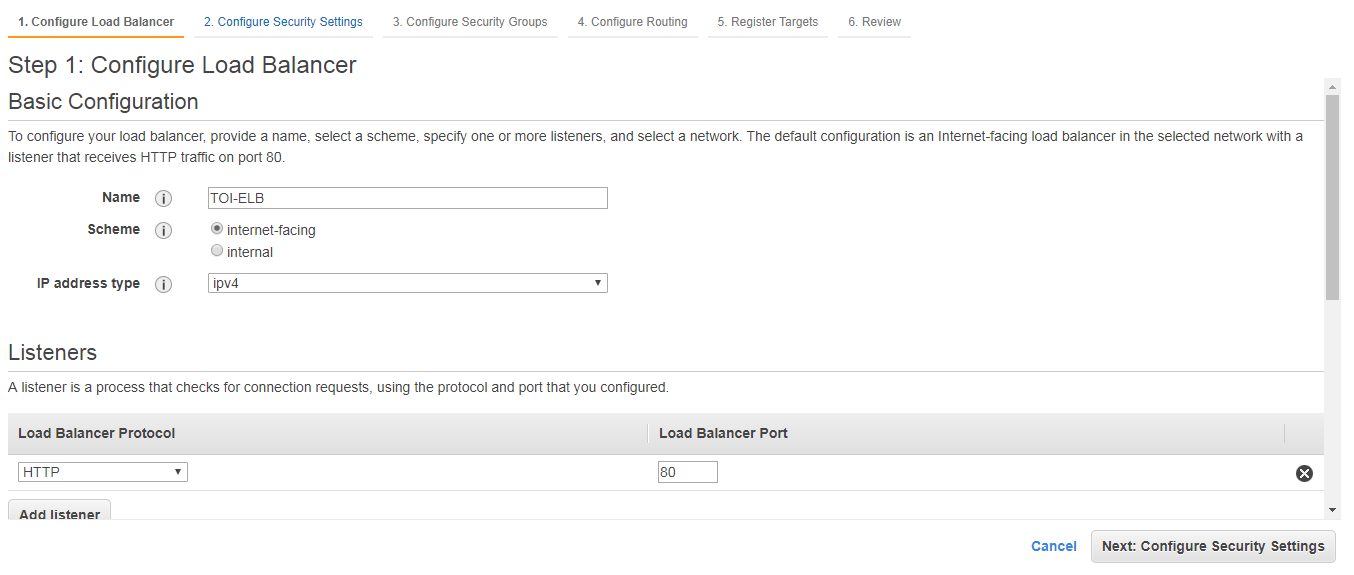
Step6: Create **Elastic Load Balancer:-**

From our architecture, our frontend tier can only accept traffic from the elastic load balancer which connects directly with the internet gateway while our backend tier will receive traffic through the internal load balancer. The essence of the load balancer is to distribute load across the EC2 instances serving that application. If however, the application is using sessions, then the application needs to be rewritten such that sessions can be stored in either the Elastic Cache or the DynamoDB. To create the two load balancers needed in our architecture, we navigate to the **Load Balancer** page and click on **Create Load Balancer.**

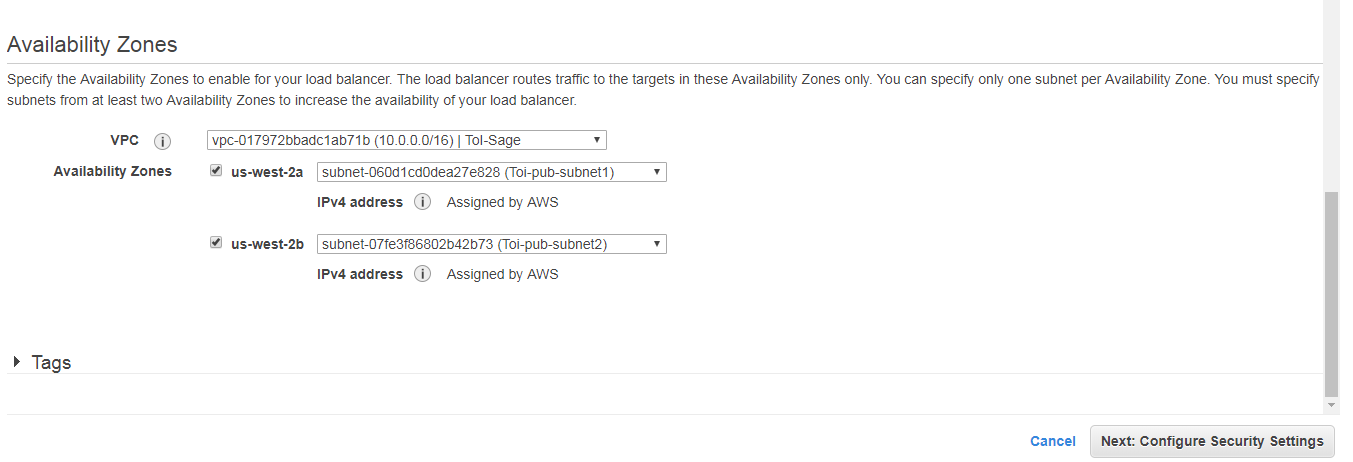
**a.** Select the Application Load Balancer.

**b.**Click on the **Create** button

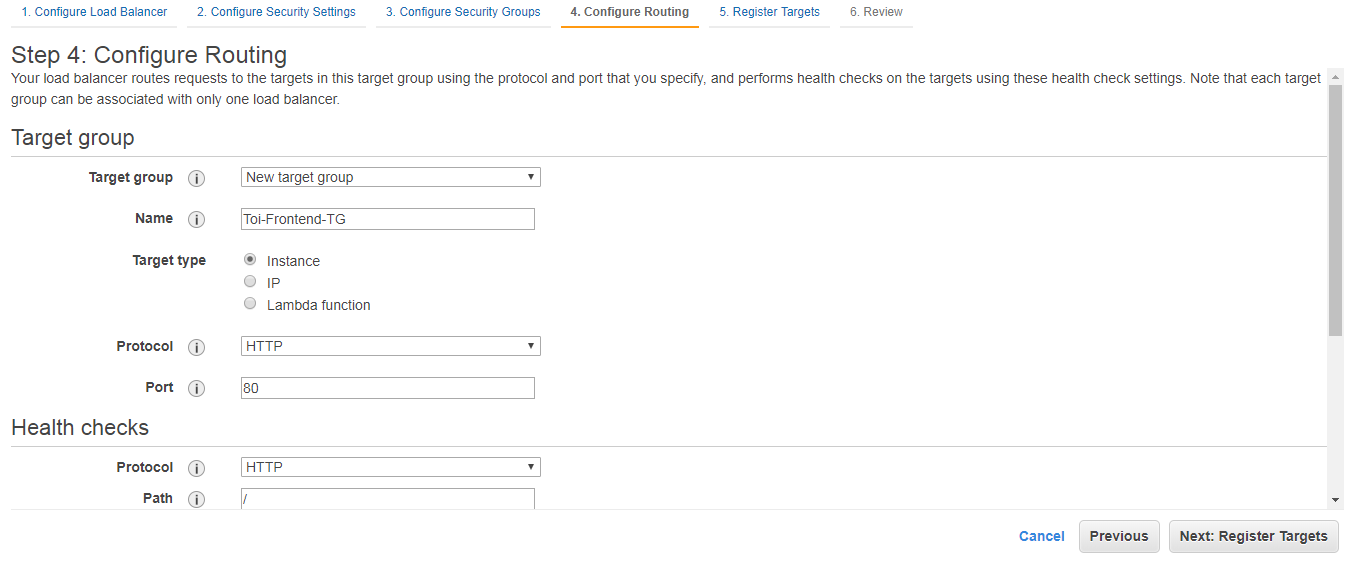
Internet Facing Load Balancer for the Frontend tier



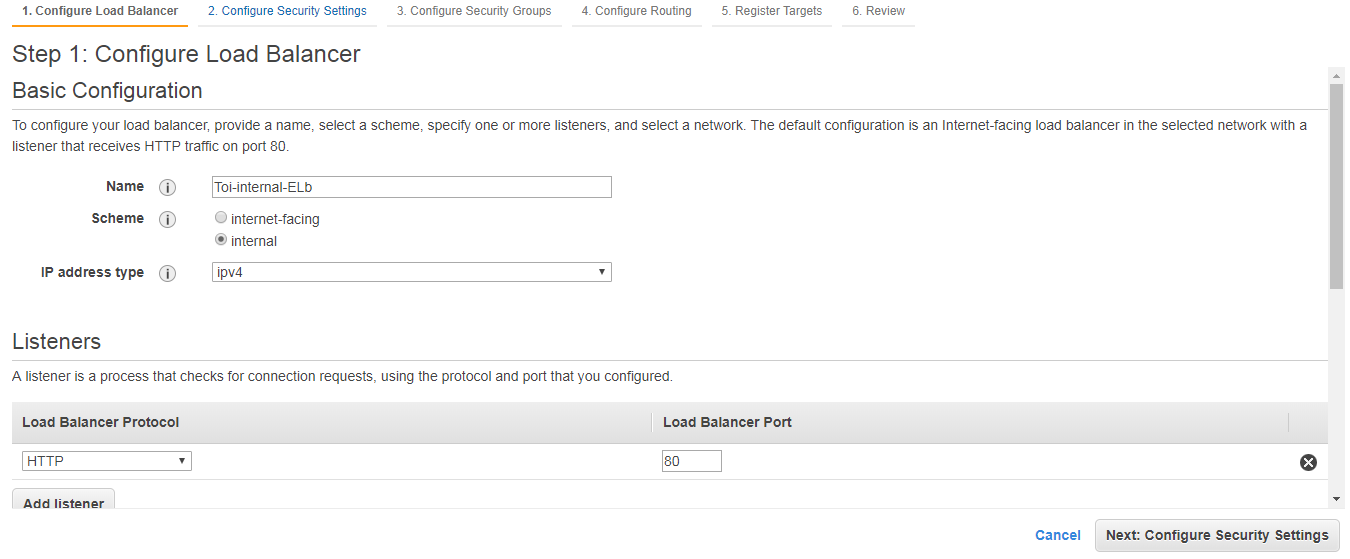
After that click on Availability zones. and select vpc,public subnets.

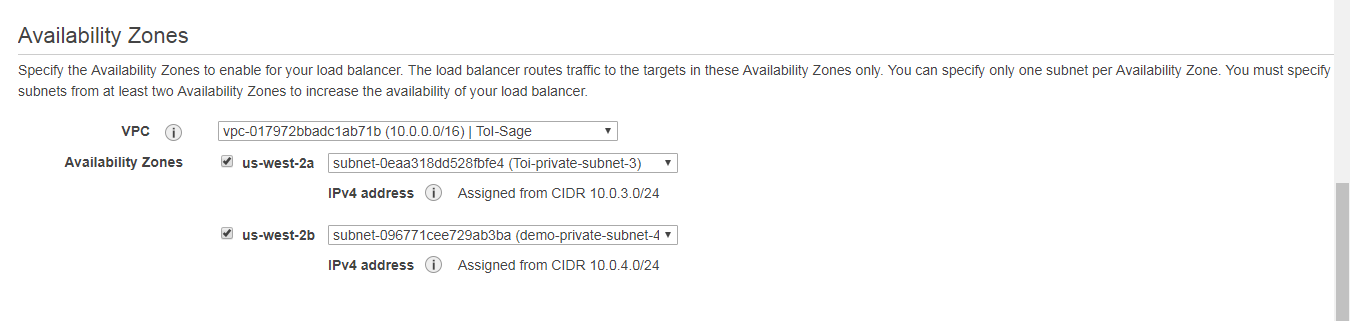


Under the **Configure Routing,**we need to configure our Target Group to have the **Target type** of **instance.**We will give the **Target Group** a name that will enable us to identify it. This is will be needed when we will create our **Auto Scaling Group.**For example, we can name the Target Group of our frontend to be **Demo-Frontend-TG.**



**Now follow same steps and create internal Load balancer.but we select private subnets.**





**Step7:-**

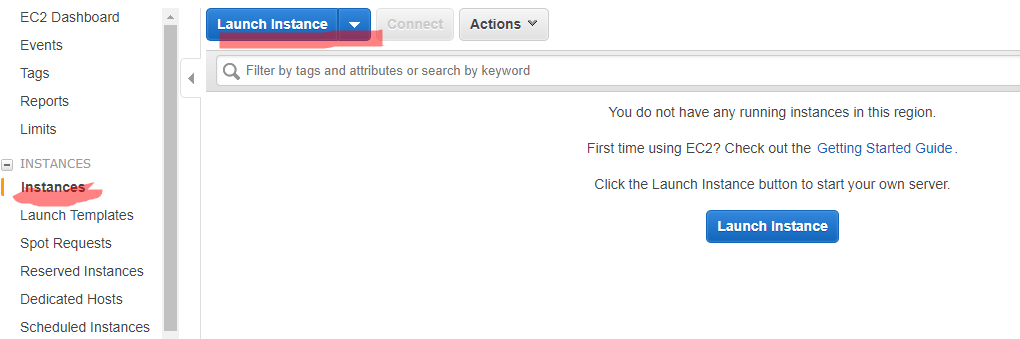
**Auto Scaling Group:**We can simply create like two EC2 instances and directly attach these EC2 instances to our load balancer. The problem with that is that our application will no longer scale to accommodate traffic or shrink when there is no traffic to save cost. With Auto Scaling Group, we can achieve this feat. Auto Scaling Group is can automatically adjust the size of the EC2 instances serving the application based on need. This is what makes it a good approach rather than directly attaching the EC2 instances to the load balancer.

To create an Auto Scaling Group, navigate to the **Auto Scaling Group** page, Click on the **Create Auto Scaling Group** button.

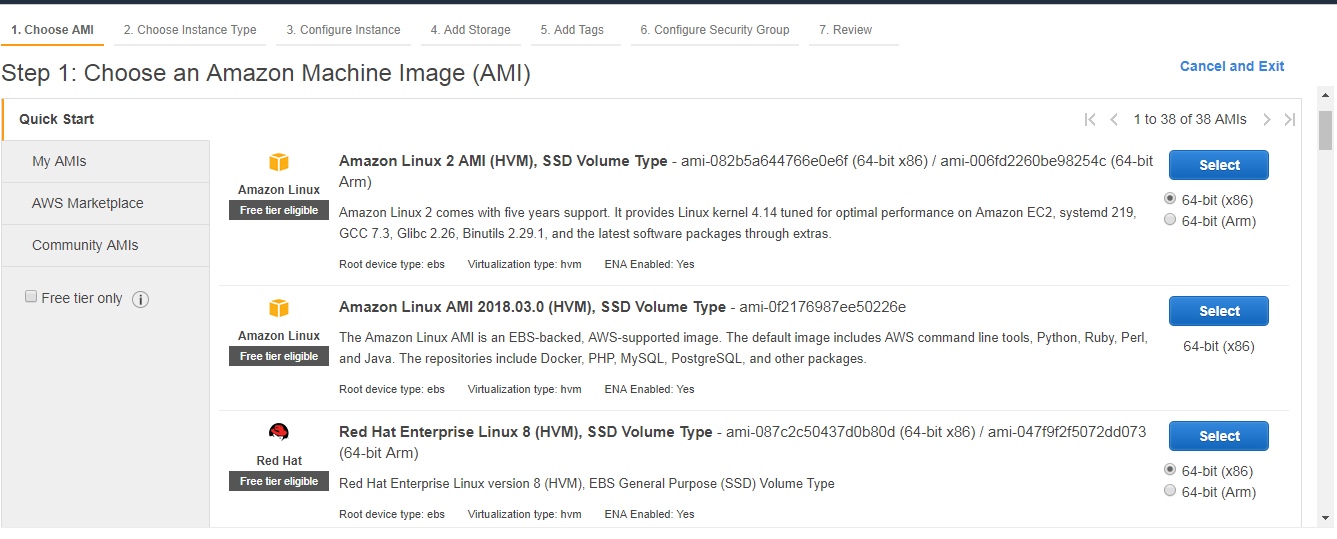
**a.** Auto Scaling Group needs to have a common configuration that instances within it **MUST**have. This common configuration is made possible with the help of the **Launch Configuration**. In our Launch configuration, under the Choose AMI, the best practice is to choose the AMI which contains the application and its dependencies bundled together. You can also create your custom AMI in AWS.

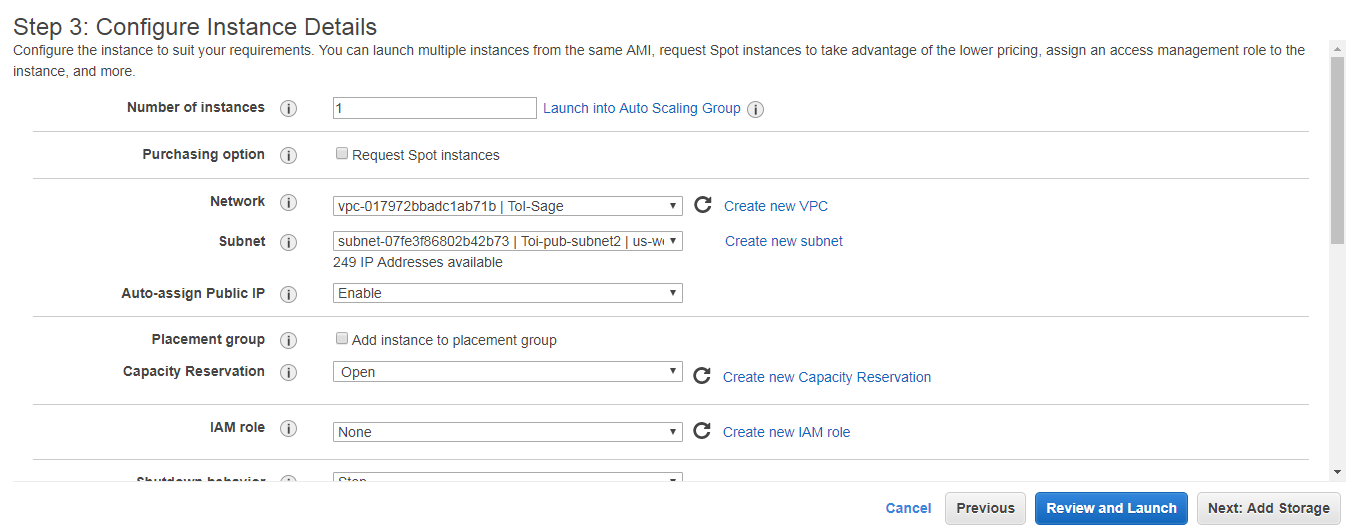
**First we create custom Ami image. like below**

**First we go Ec2 launch configuration.**

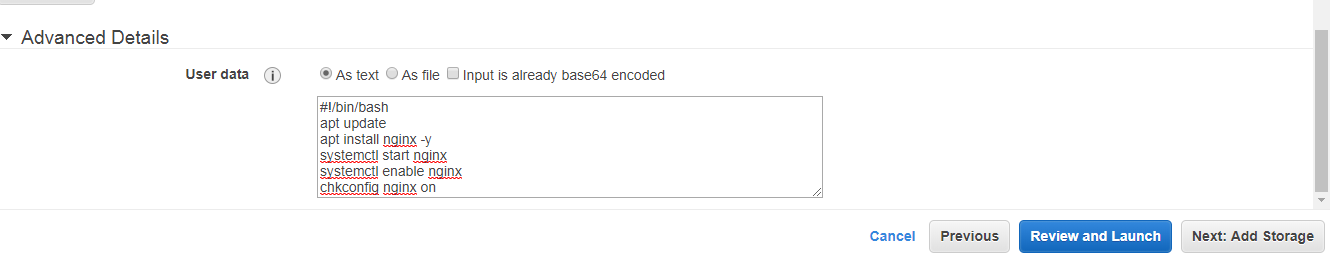


**Choose our Operating system.**



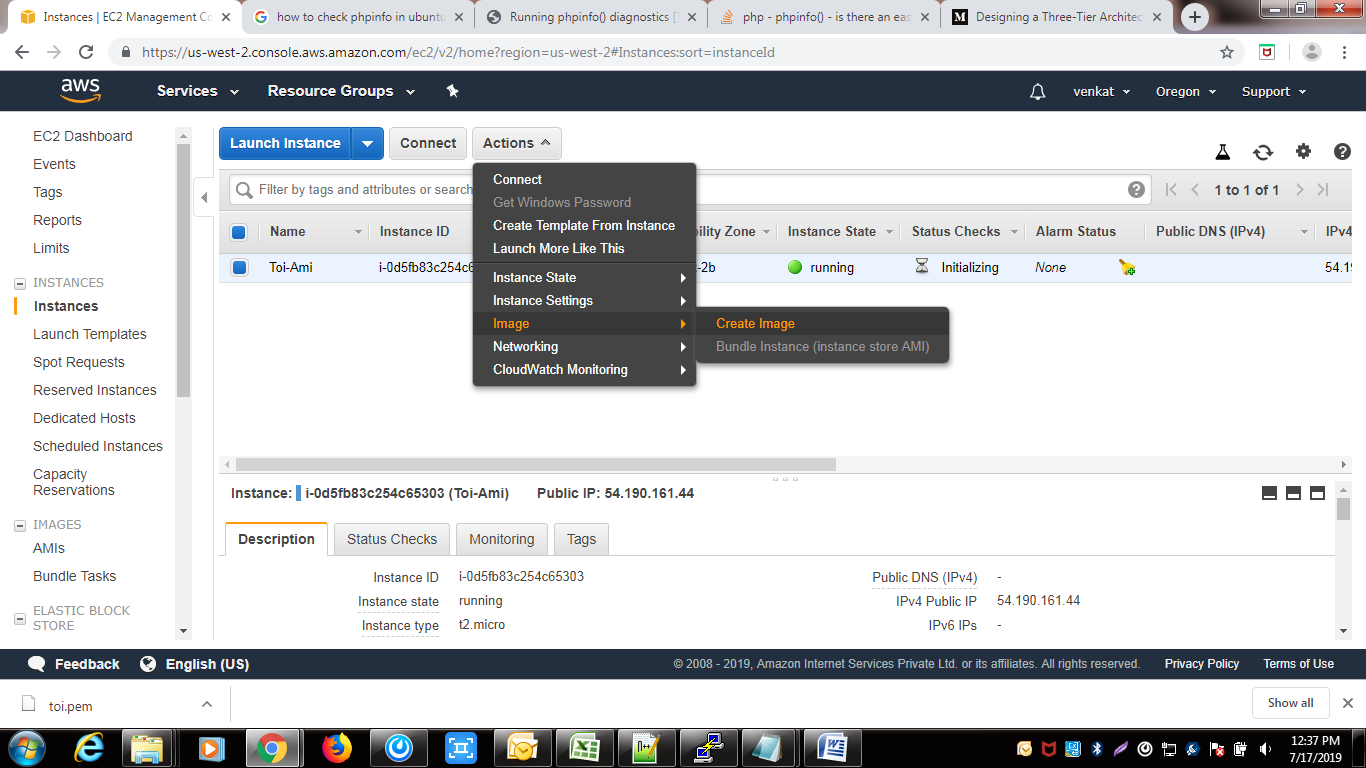


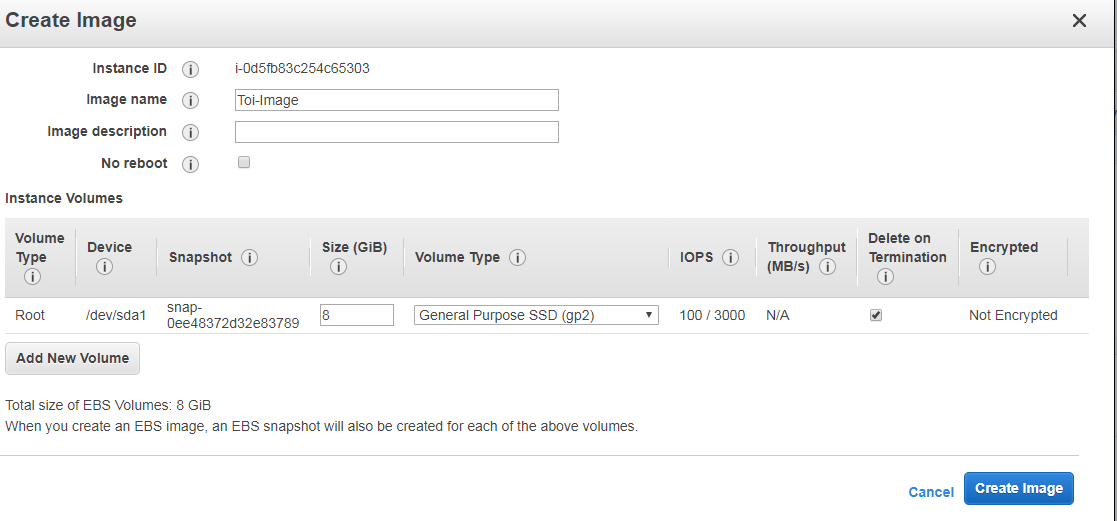
**Go to advanced details .in that user data type the our installation commands.**



**Now launch the instance.**

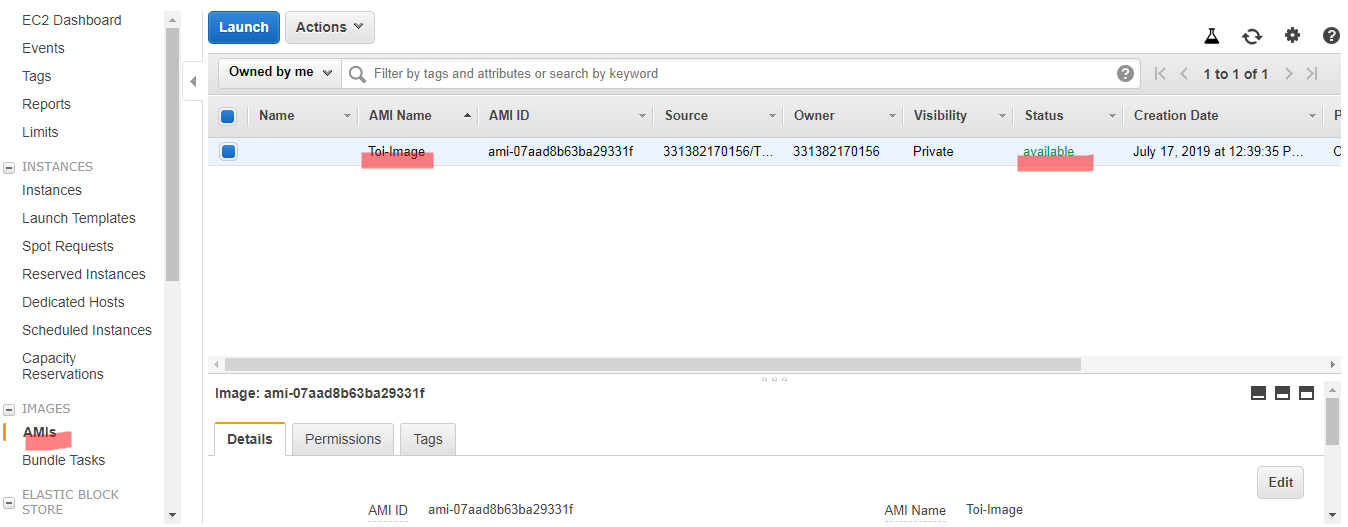
**After launch instance. we have to create image. Like below diagram.**

****

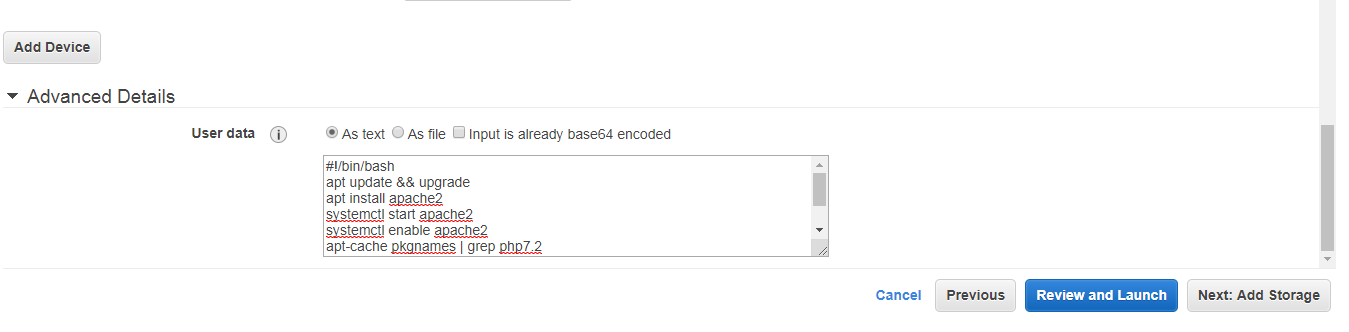


**Give the image name and click n create image.**

**Now click on AMI and check the image created or not.**

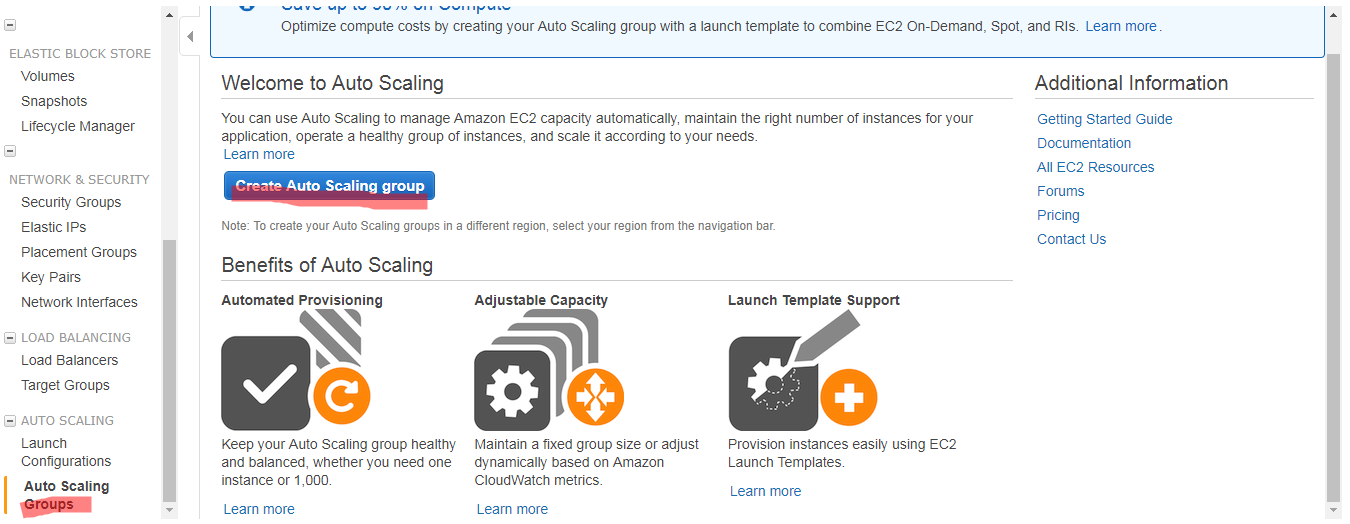


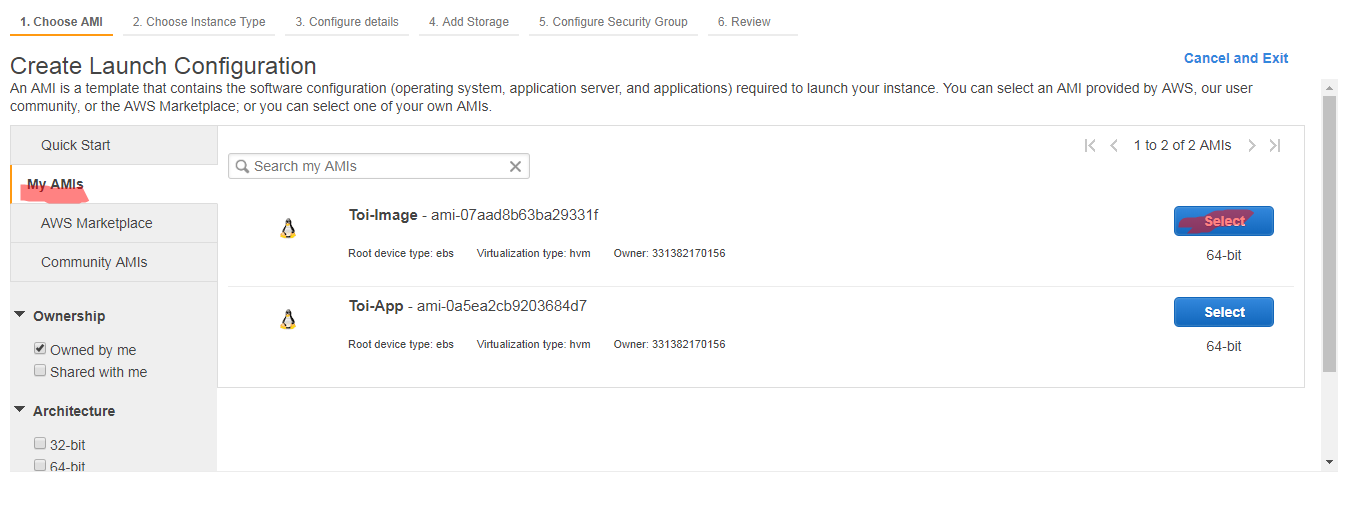
**we have to follow same procedure App server also.**



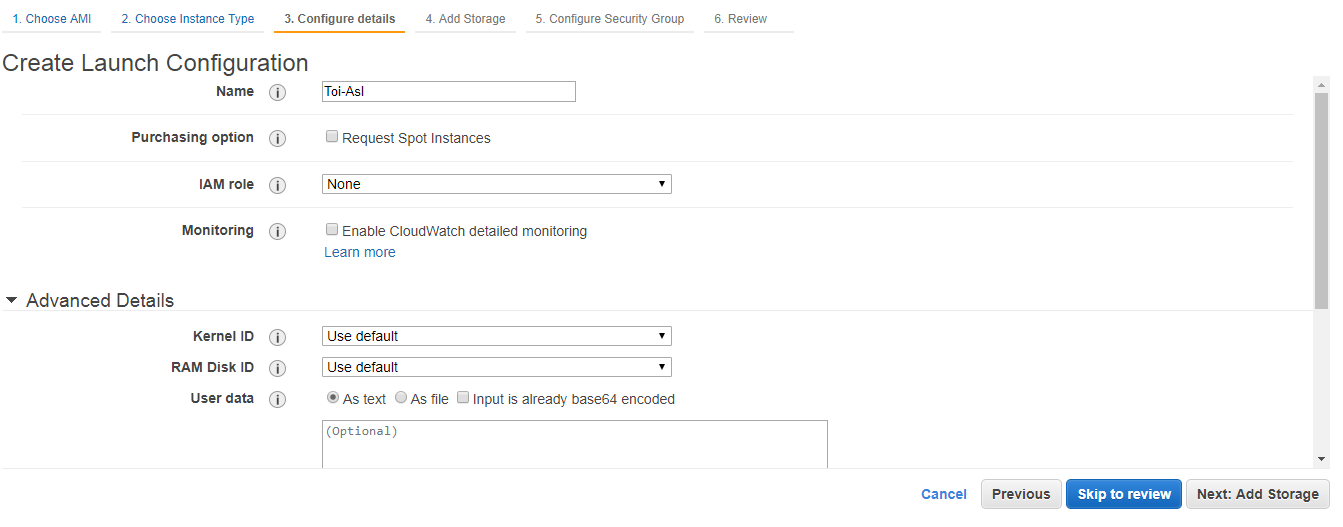
**Creating Auto scale group:-**

**Now click on Auto scaling Group.**

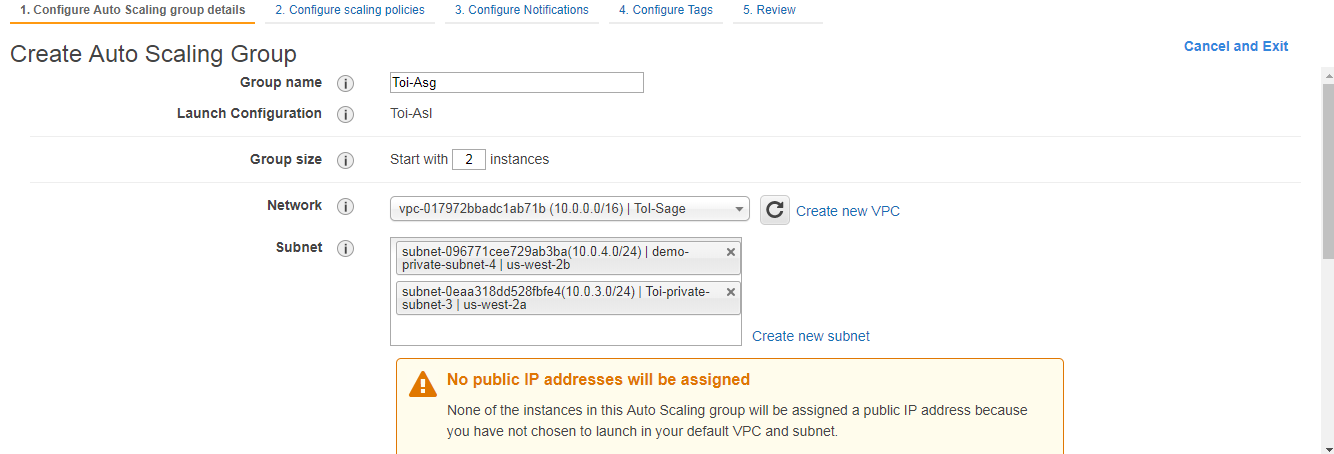




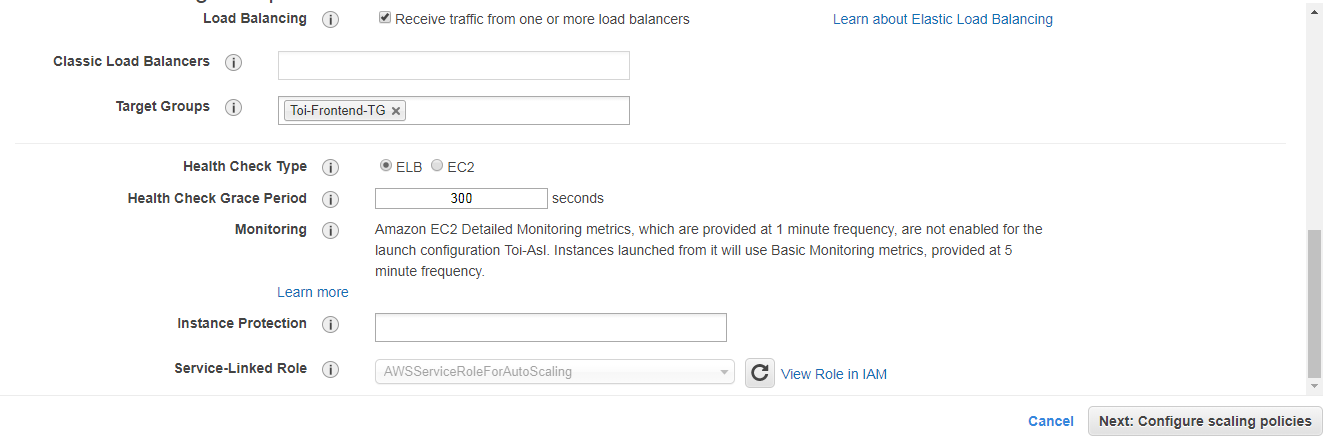
**Now click on Create auto scaling group. and click on my Ami and select our Image.**



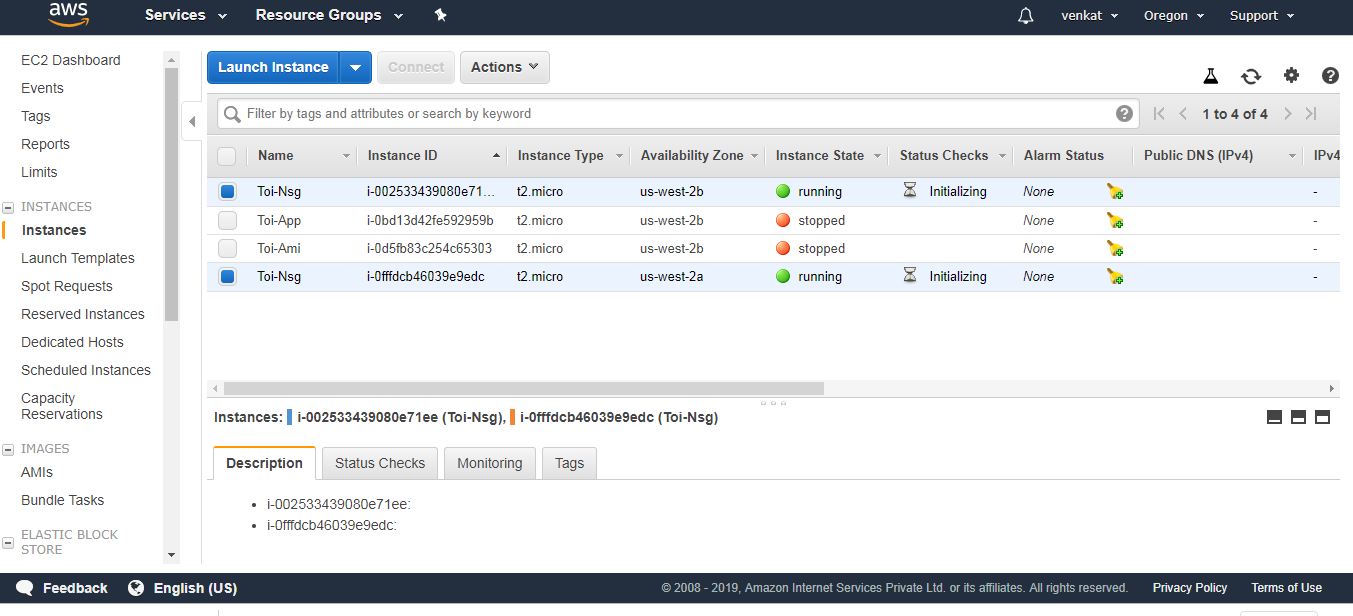
**Give the AsG name and launch configuration.**



**Give the ASg Group name and select group size and select our vpc,and select 2 private subnets.**



**Click on advanced details. And select load balanceing,our target group, health check type and launch instances.**



**After that we have to follow same steps we create the App server.**

**Create the Bastion Host:-**

The bastion host is just an EC2 instance that sits in the public subnet. The best practice is to only allow SSH to this instance from your trusted IP. To create a bastion host, navigate to the EC2 instance page and create an EC2 instance in the **demo-public-subnet-1**subnet within our VPC. Also, ensure that it has public IP.

In just we launch the ec2 instance.

How to login Ec2 instances by using Bastion Host:-

a)Create the test.pem file(key file) in Bastion host.

B)Give the chmod 400 permission

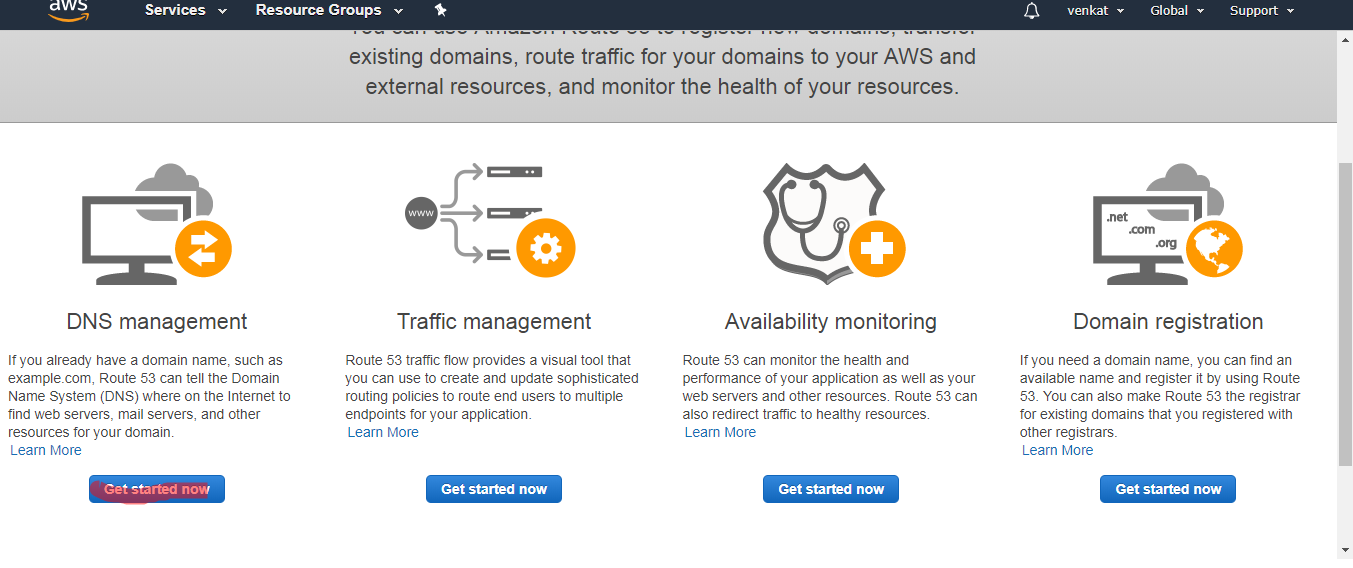
chmod 400 test.pem

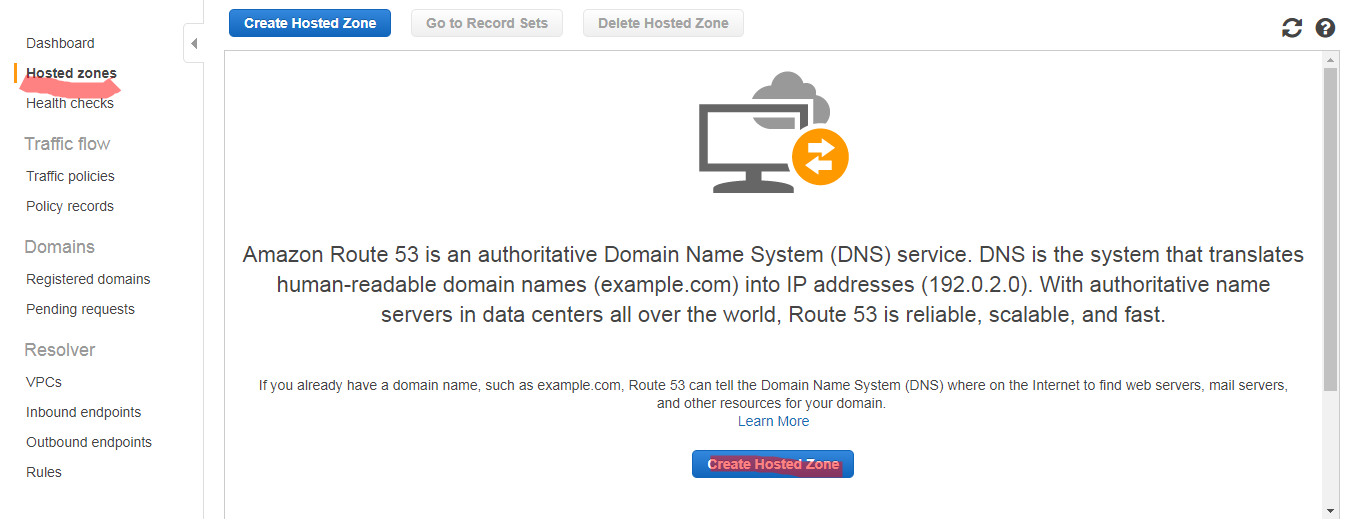
ssh -i test.pem ubuntu@privateip

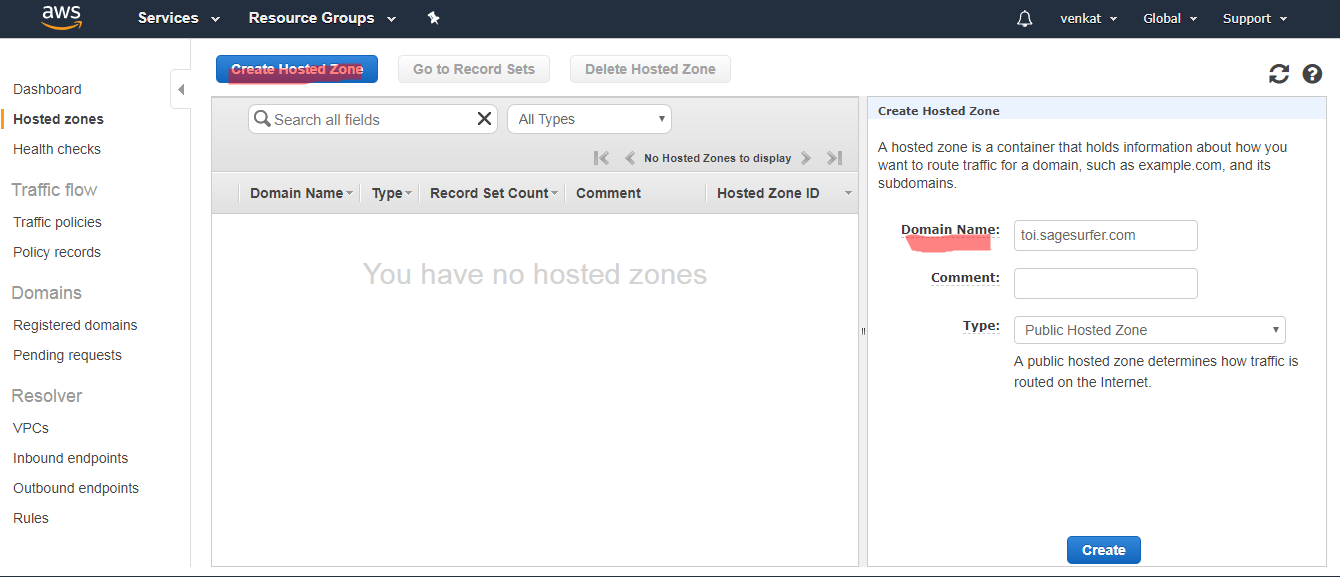
Step8: Configure route53:-

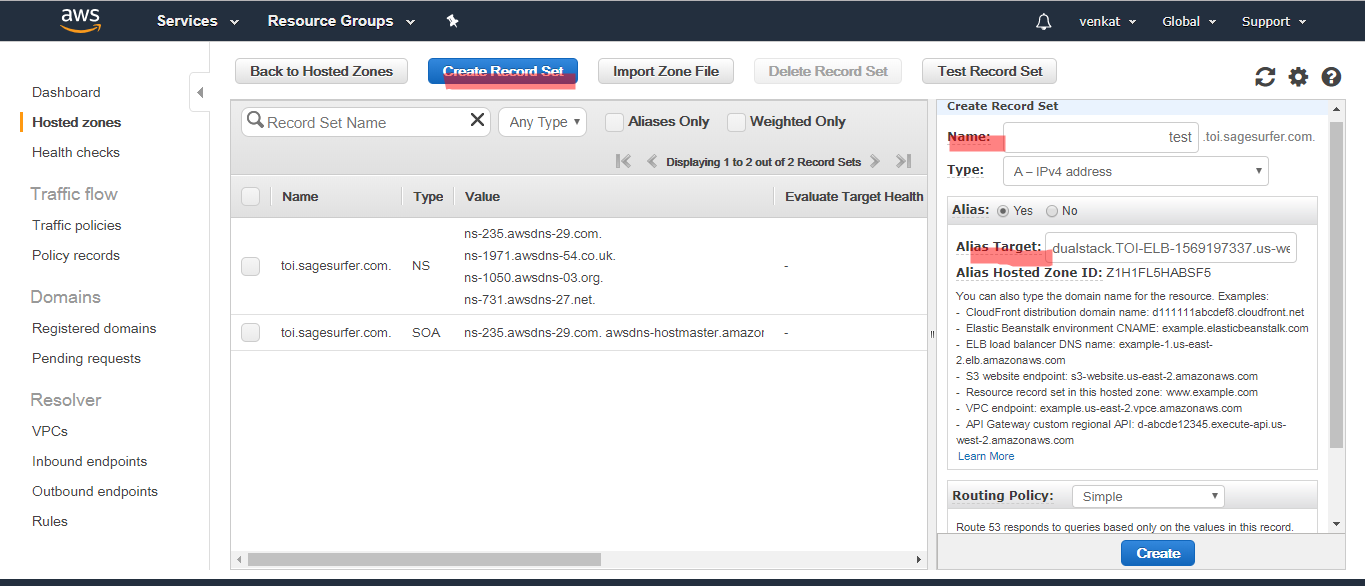
Route53 is used for domain mgmt and traffic policies.

First we register the Domain name in godaddy or Route53.If already we have domain go to Route 53 and click on Dns Mgmt.









**After that create Host zone. In that give the domain name.**

**After that click on create record set. in that name: domain name**

**Type: ip4 address**

**Alias: Yes**

**Alias target: choose ELB**

**Route Policy:- simple**

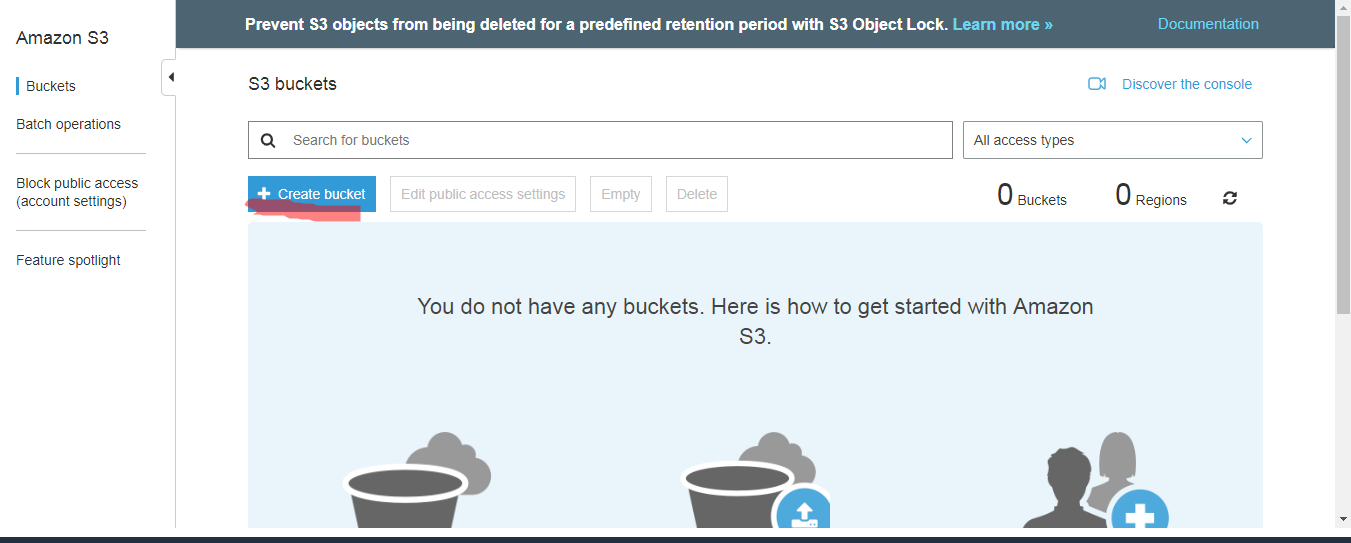
**Now click on create.**

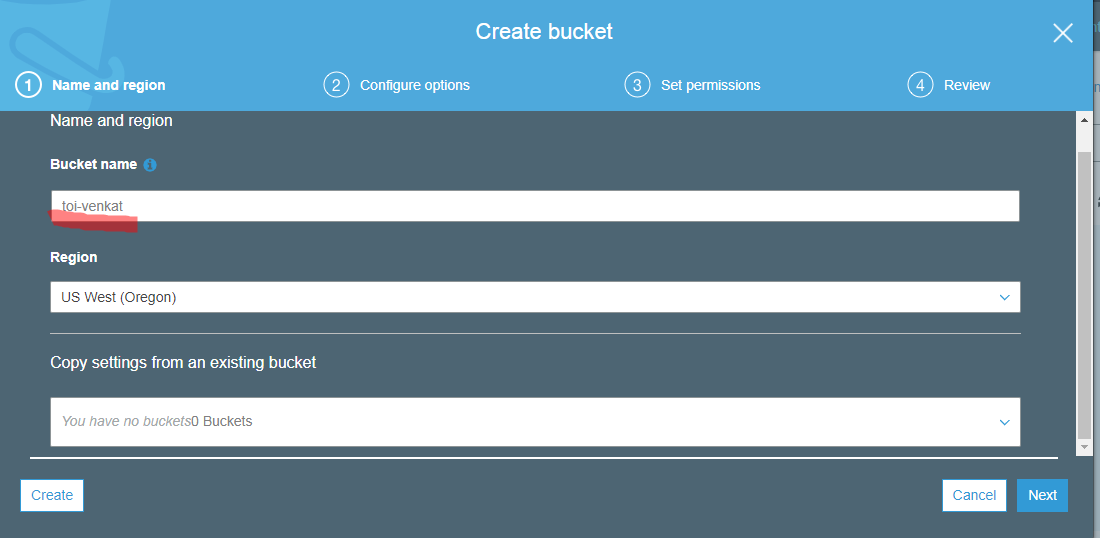
**Step9: Create S3 Bucket:-**

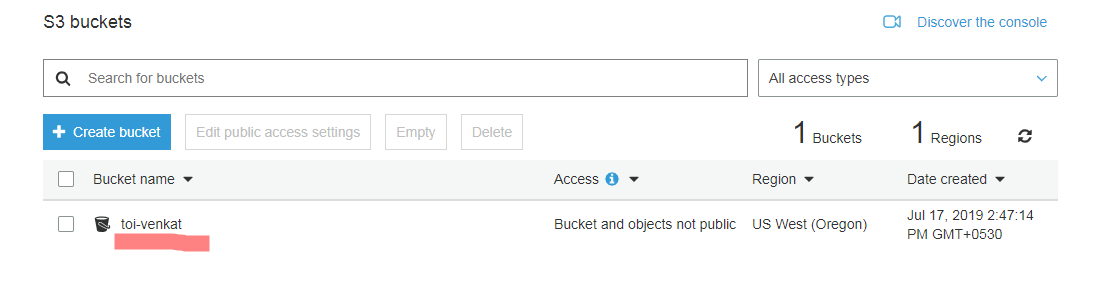
**S3 bucket is used for storing the files. Now we create S3 bucket.**

**Now go to Amazon S3.In that click on Create bucket. give the bucket name and reaming details.**

**If we want to encryption .we choose encryption option.**



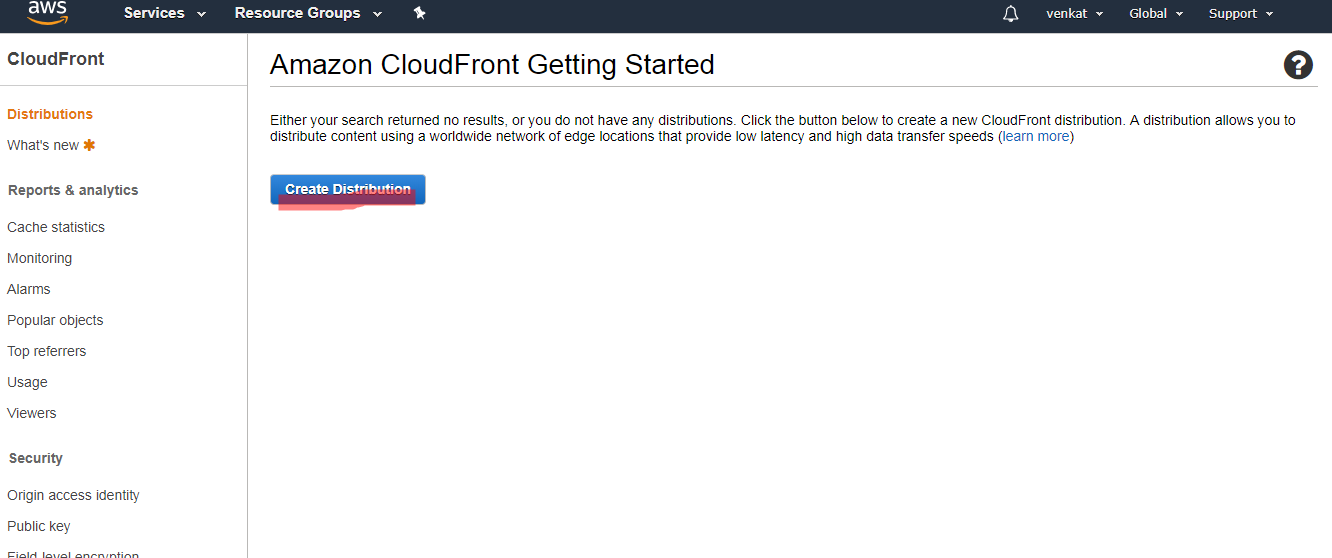


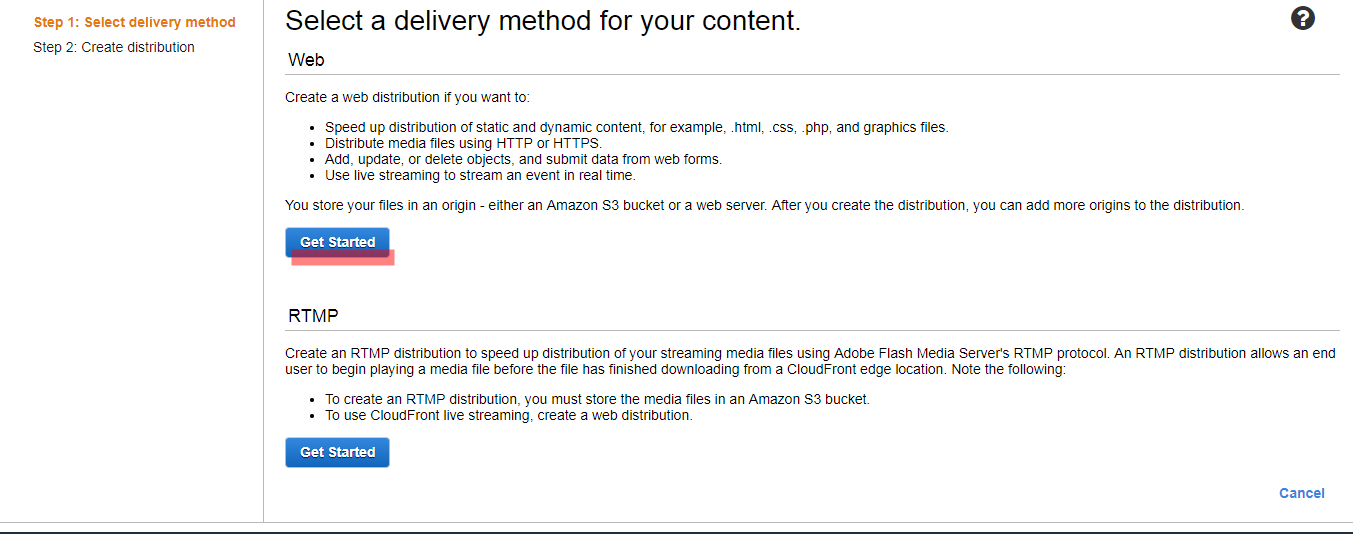


Creating Cloud Front:-(CDN)

Cloud front is used for to distribute the data on Edge locations. In that we create the distribution by using Elb,s3 bucket etc..

Go to Aws console and type Cloud front. After that click on create distribution.





After that click on web .

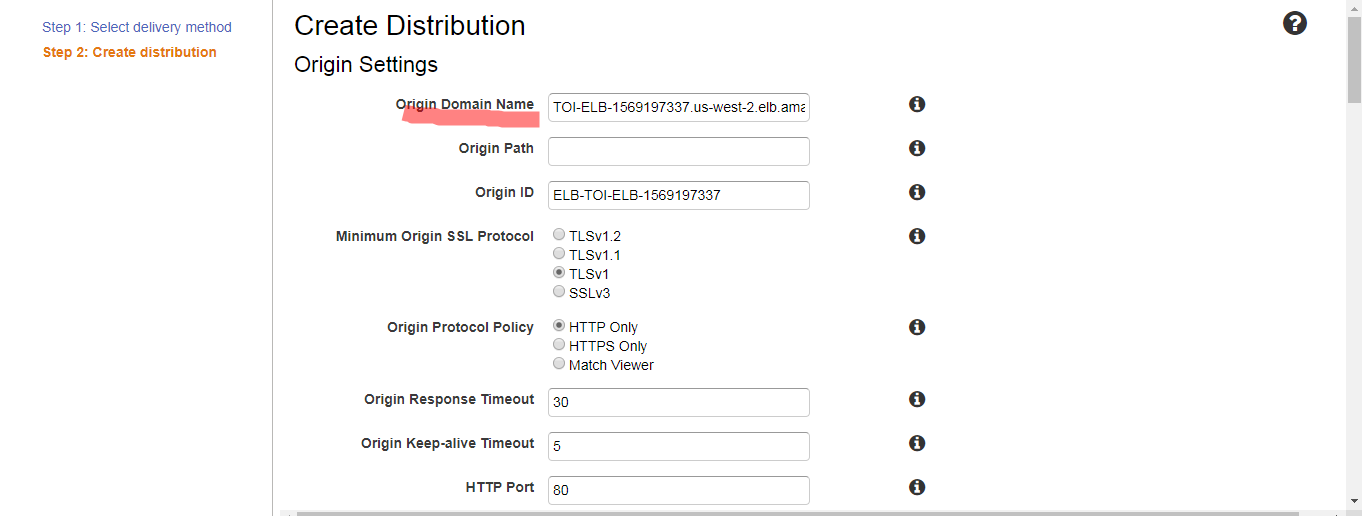
After that choose our origin domain name like Elb,s3 Etc..

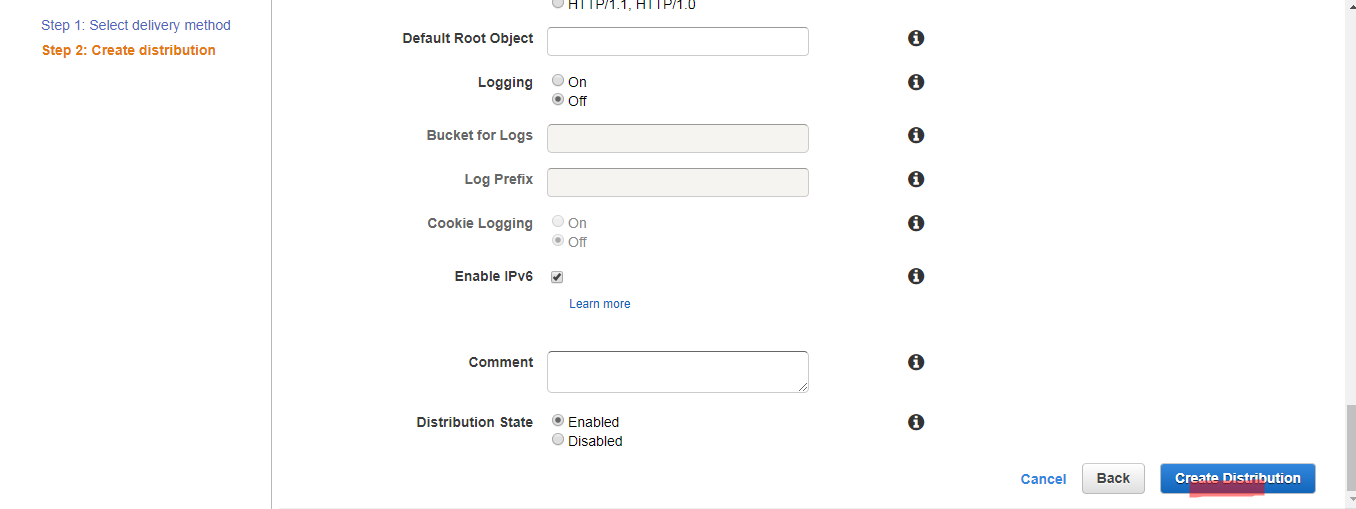
and select what do you need protocals.and click on create the distribution.

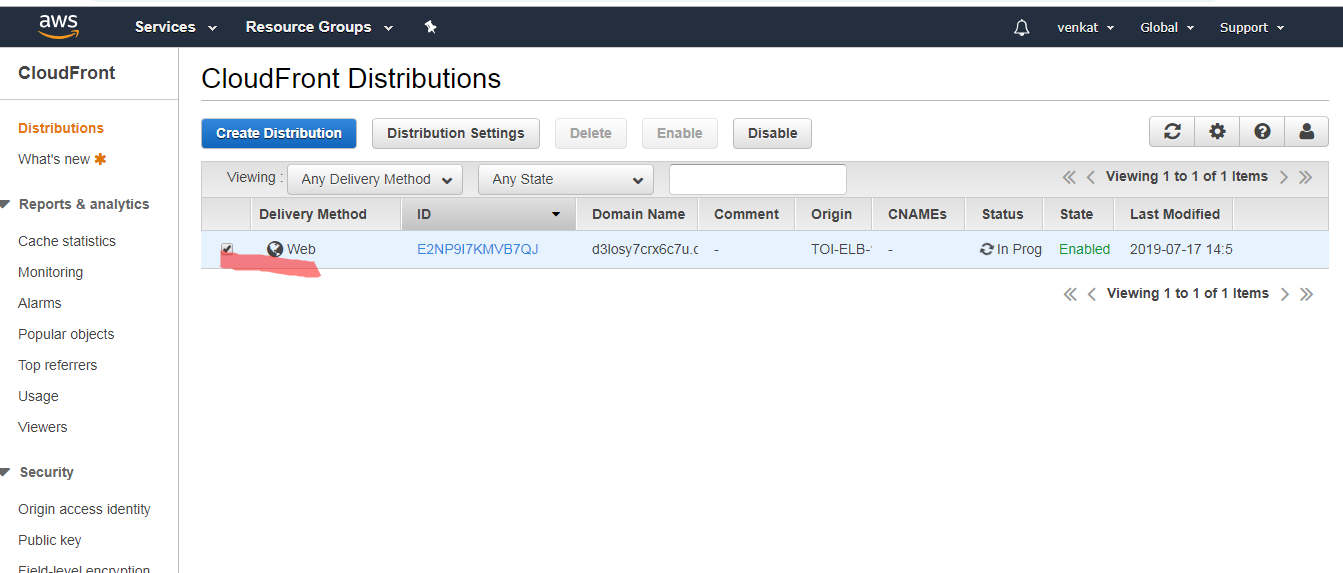
it will take time to create the distribution 15 to 20 mins.

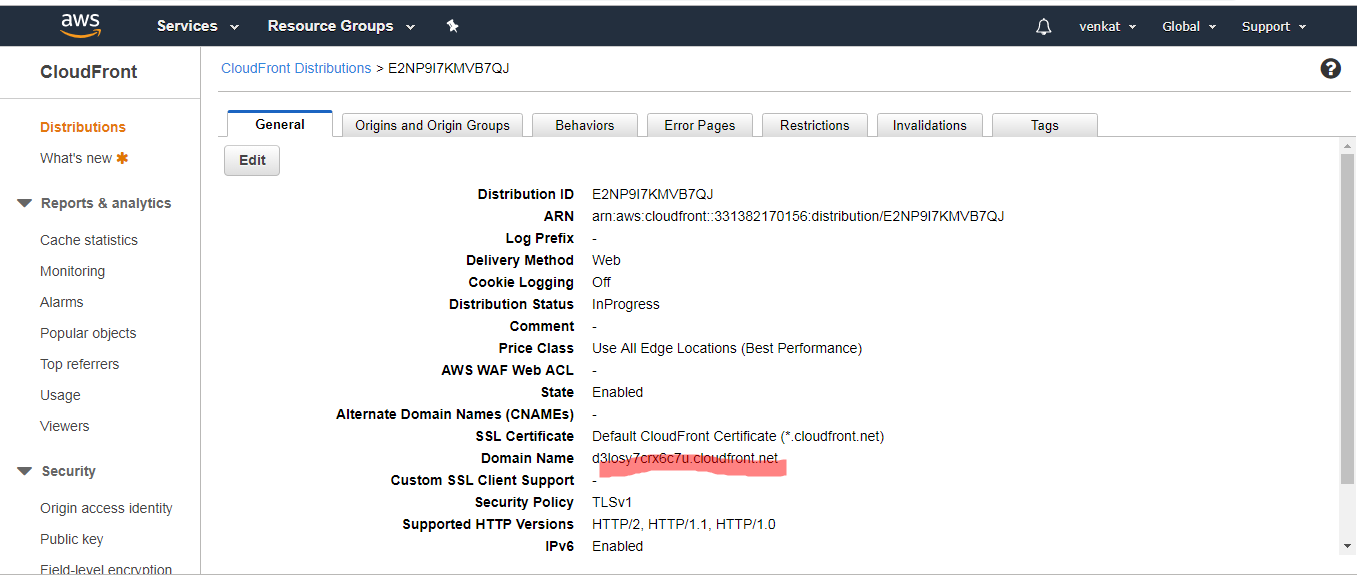
After that click on web. it is showing domain name, in that domain we have to in browser.

Please look into below screenshots.









**Create Rds Instance:-**

Rds instance is used for Database purpose. like We create the Db instance name etc..

Once we create the Db instance we configure the details in config.php file.

Now type rds in aws console.

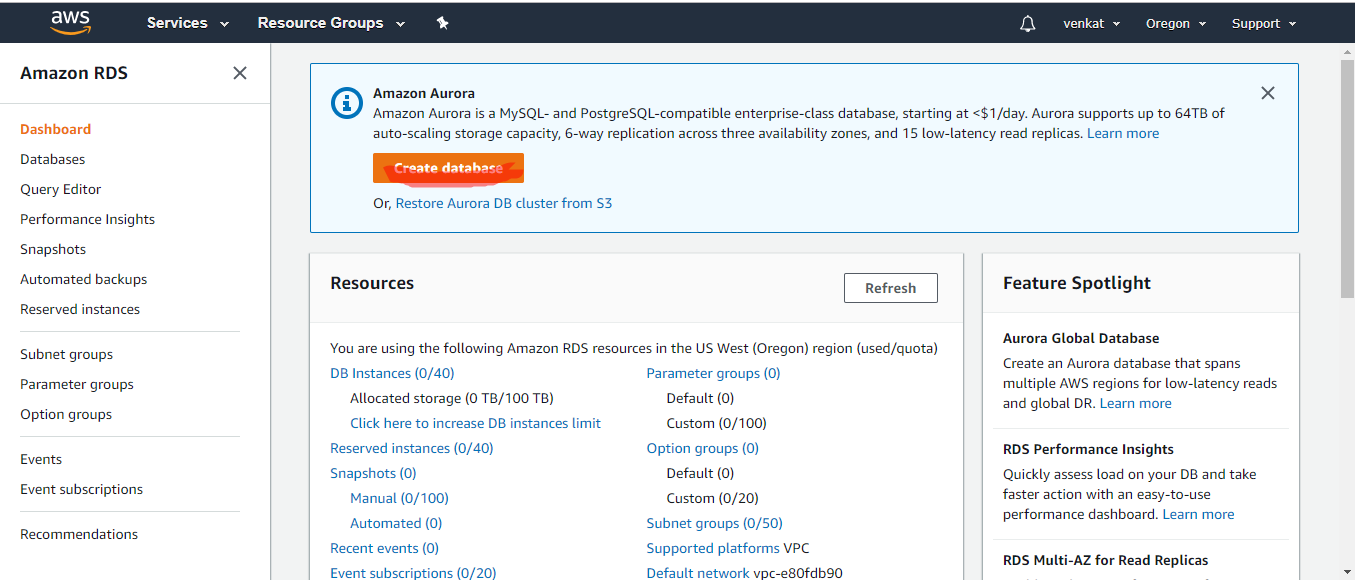
**Now go to Amazon Rds. In that click on Create database.**

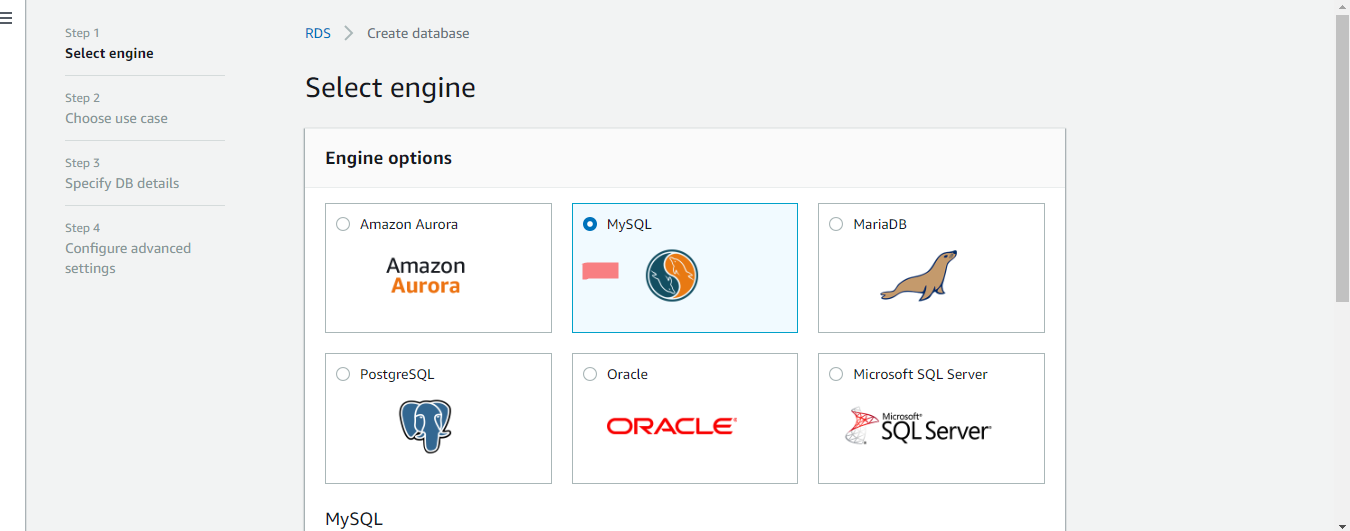
**Now mysql. And choose production-mysql.**

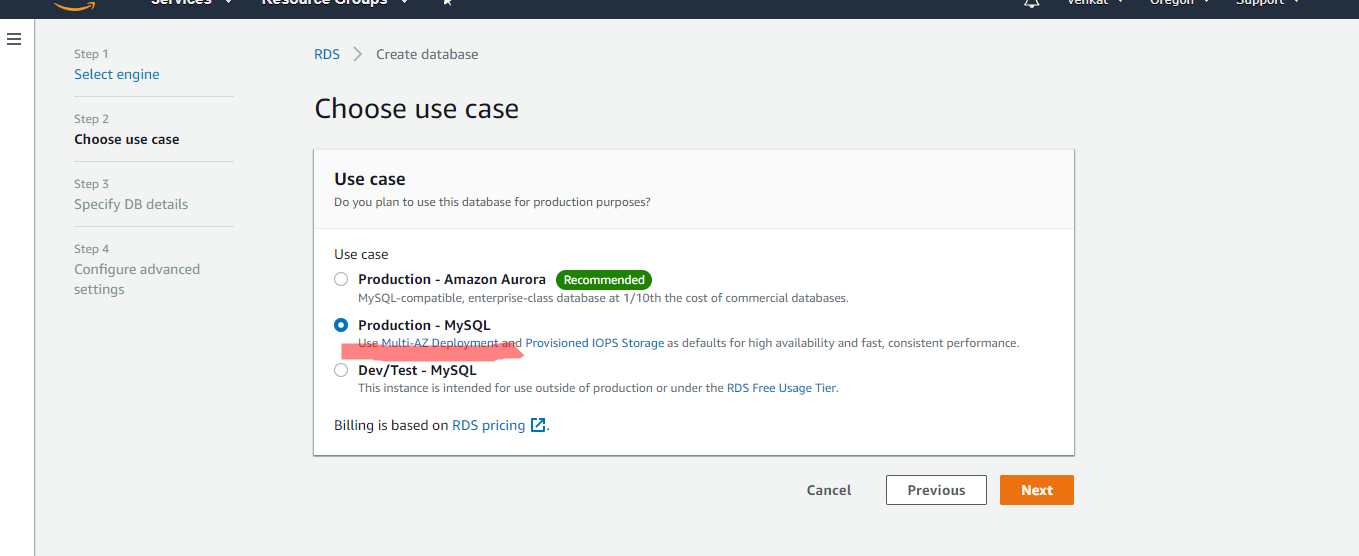
**and choose the mysql version,name,database name etc.**

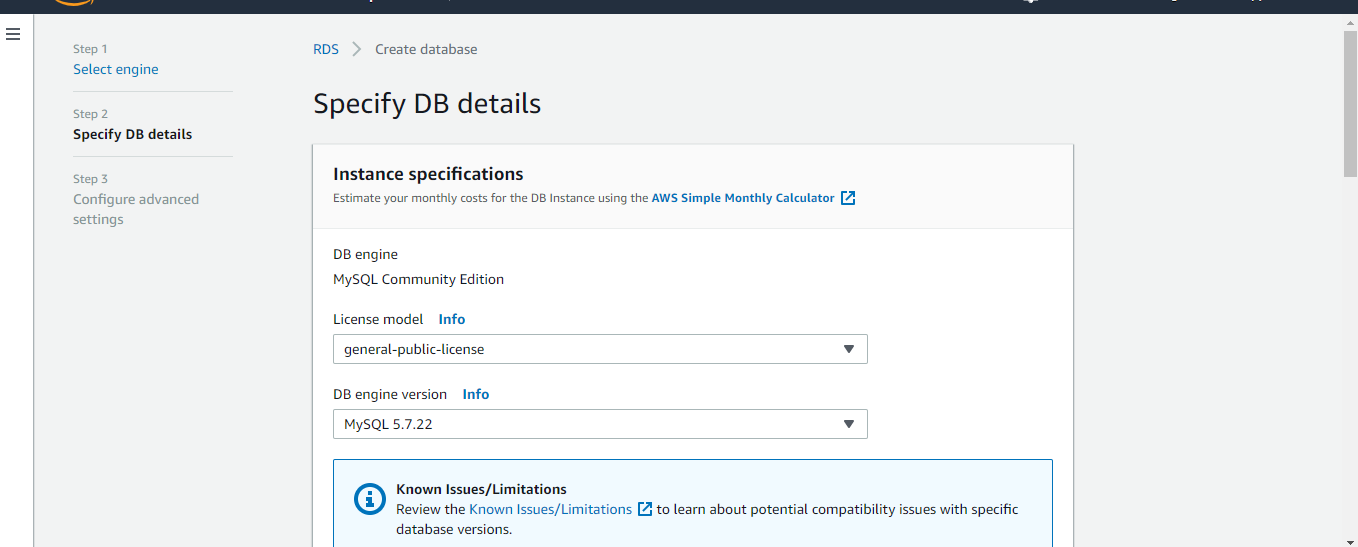
**click on create.**

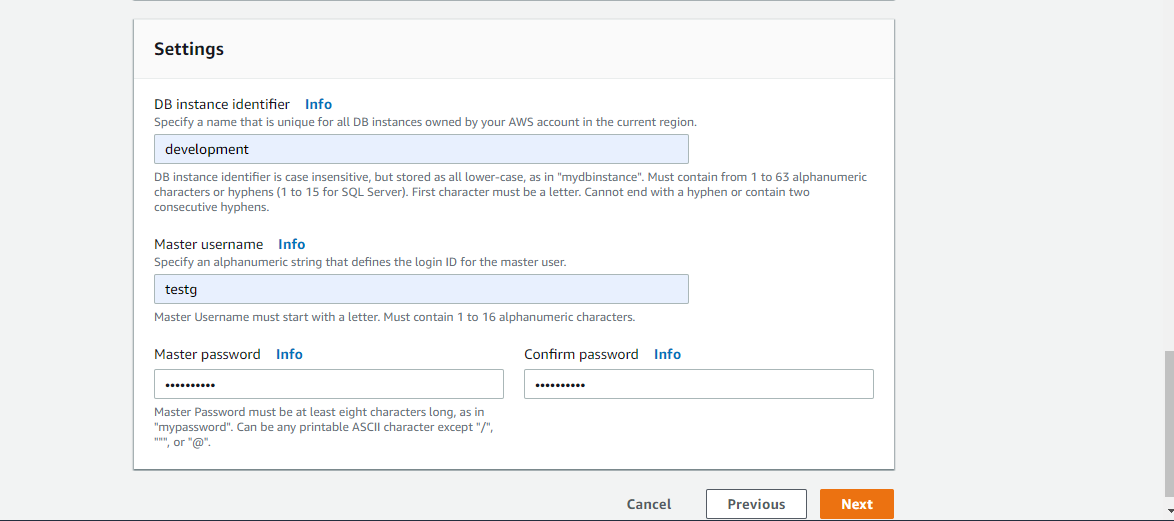
Please look into below screen shots.

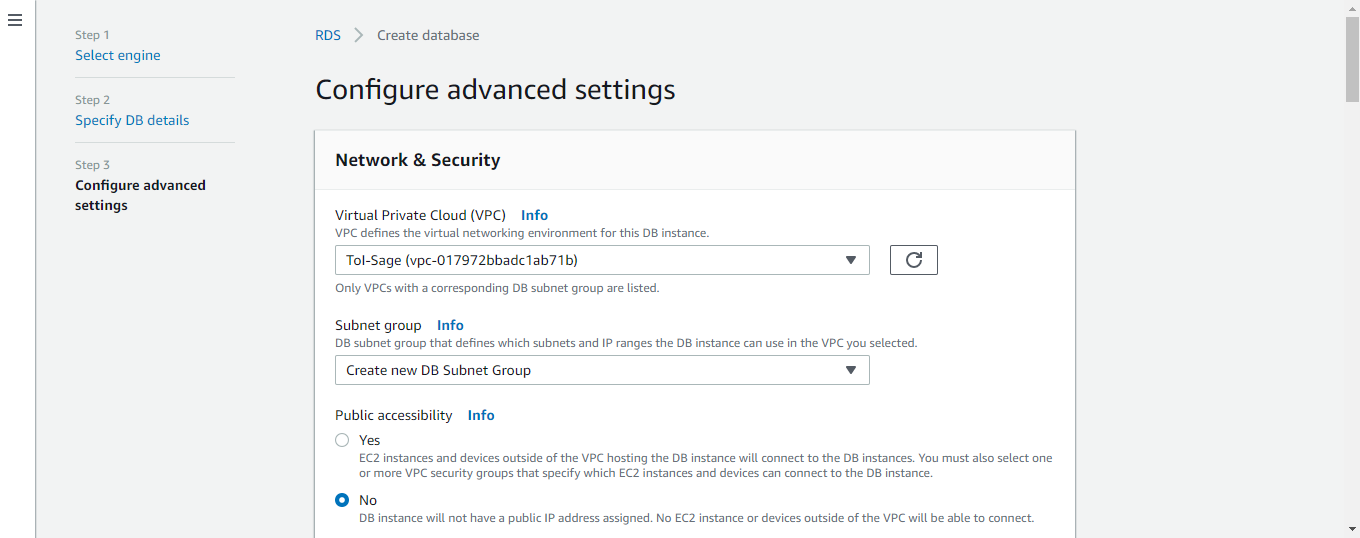


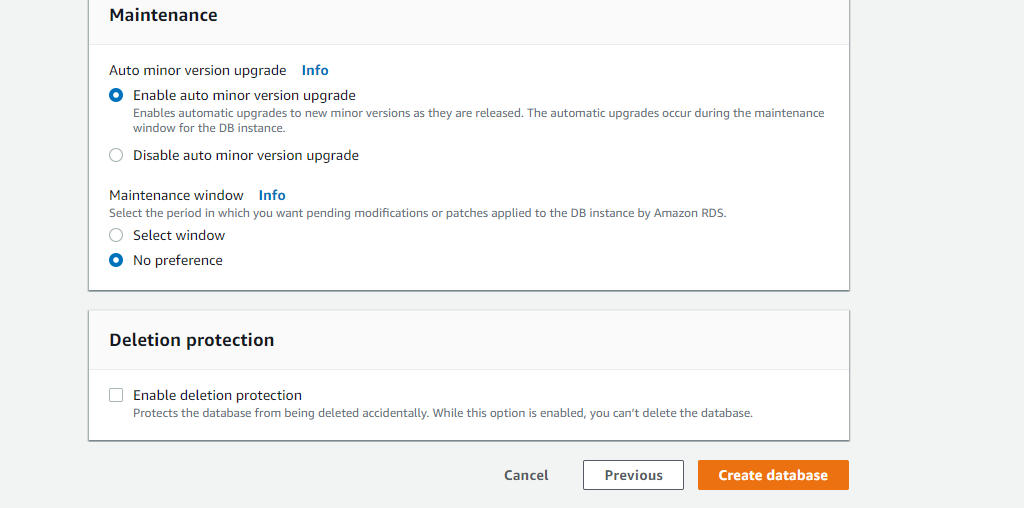










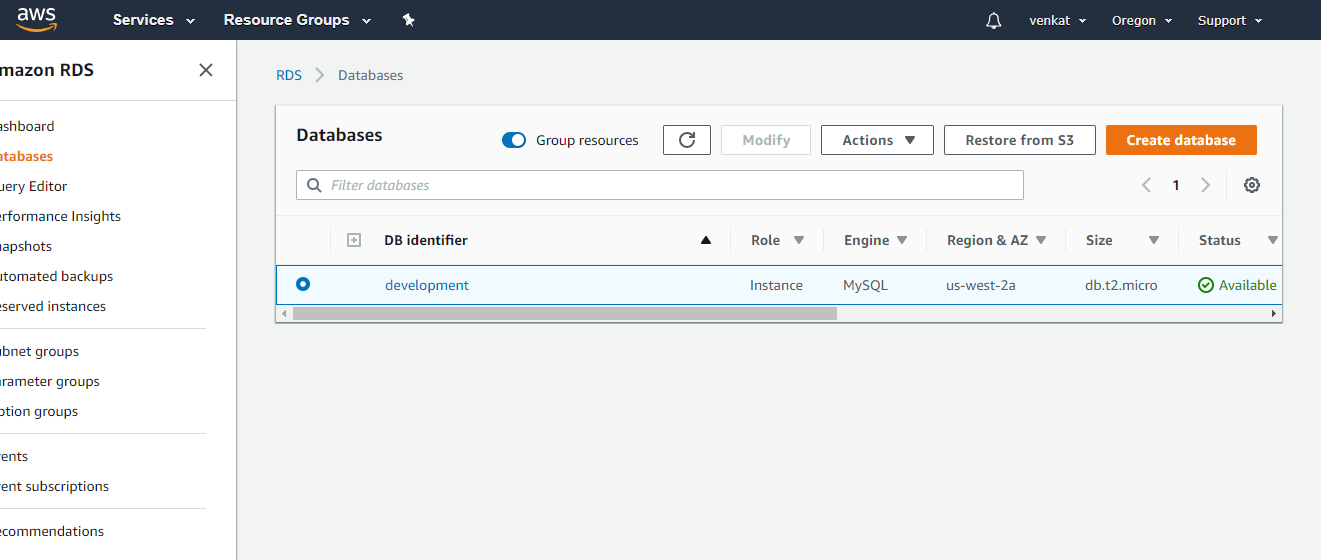


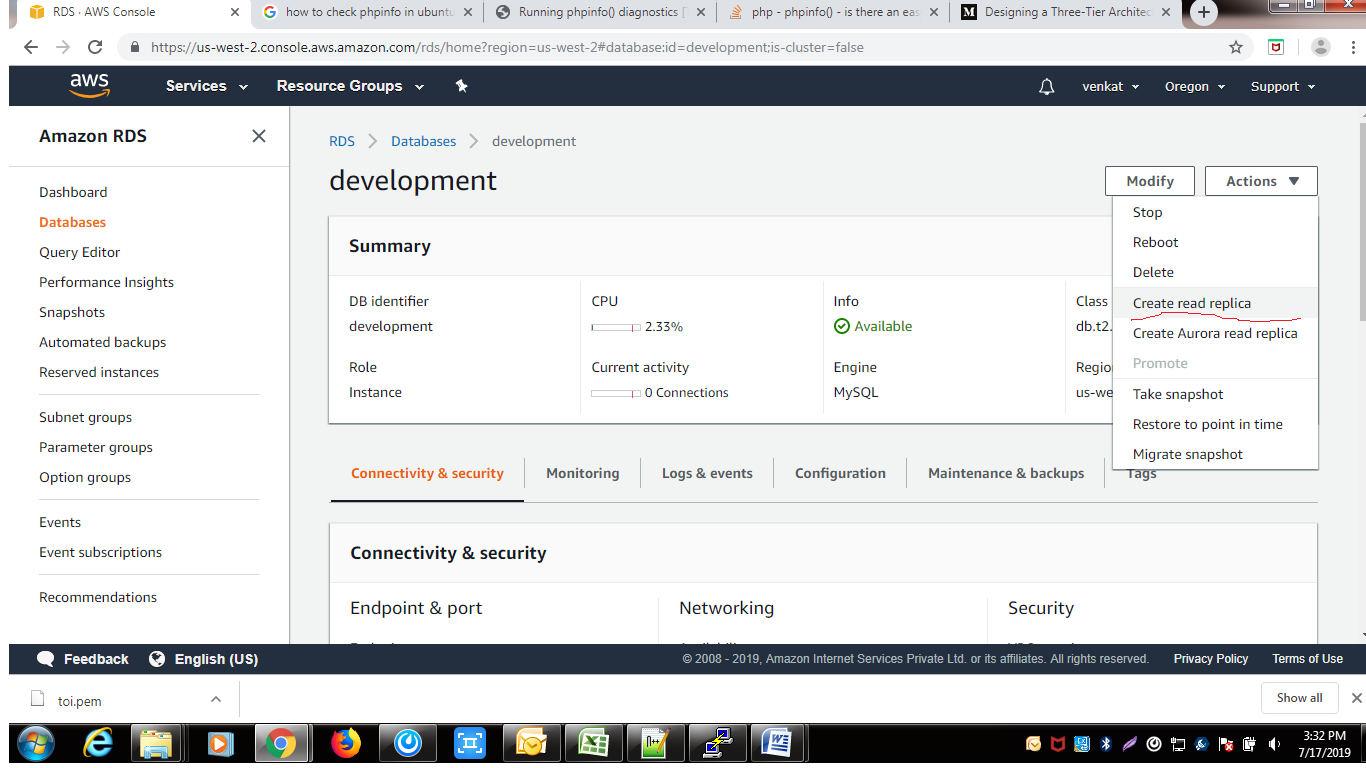
**Create Read Replica instance:-**

**Now our database in that click on actions. in that choose create read replica.**

**Now we have to select storage and usernames etc.**

**By using third party tools we connect the Database like sql yog, work bench etc..**





**Follow the below documents for reference:-**

**https://medium.com/the-andela-way/designing-a-three-tier-architecture-in-aws-e5c24671f124**

**https://media.amazonwebservices.com/architecturecenter/AWS\_ac\_ra\_web\_01.pdf**

**Connecting Db instance on Ec2 instance:-**

**First we install the mysql on Ec2 instance.**

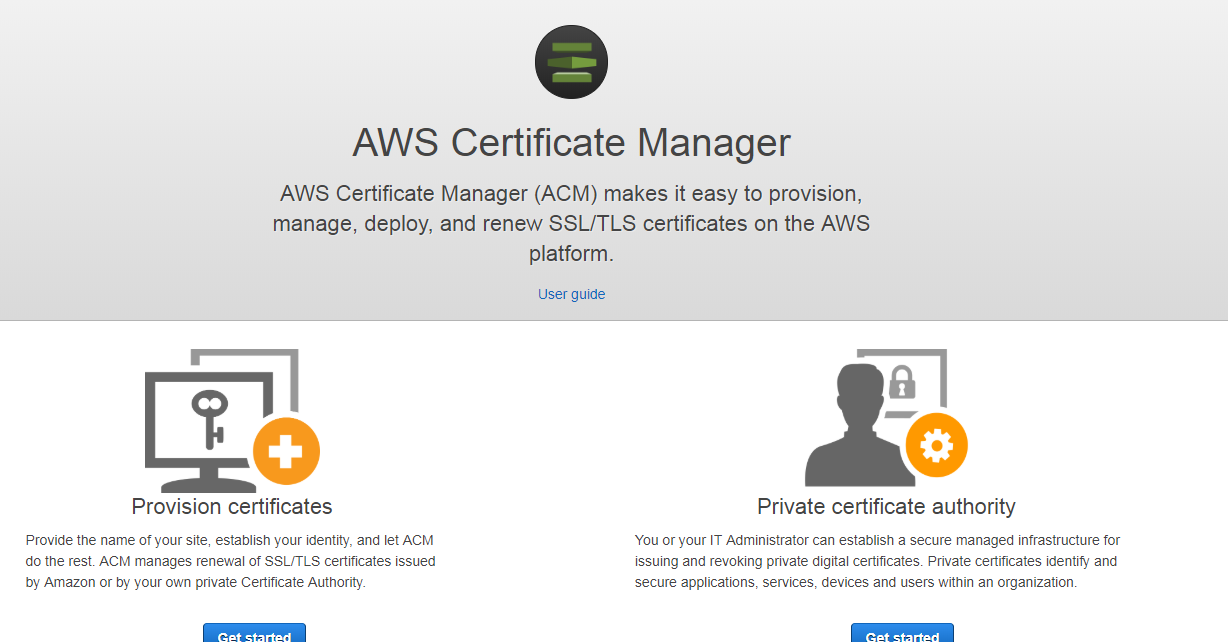
mysql -u root -p -h database endpoint

Issues:-

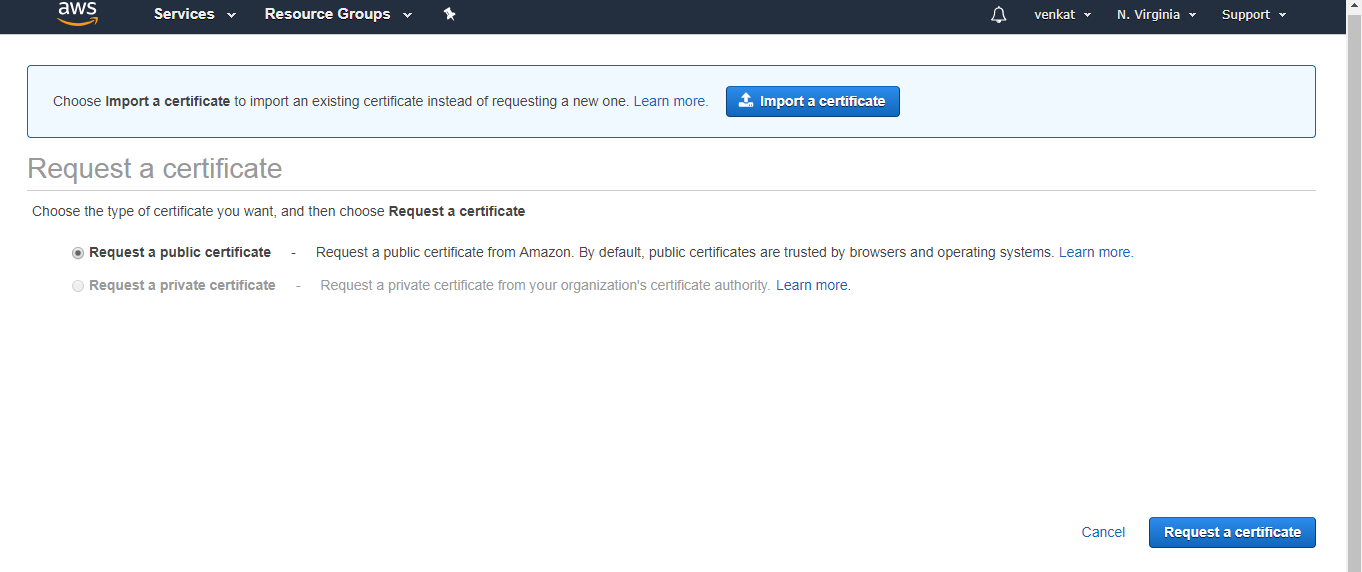
1)Upload the SSL Cerfticates:-

When we click on ELB on choose Acm.That time showing error. So, first we upload the our SSL certificates through Certificate Manager.

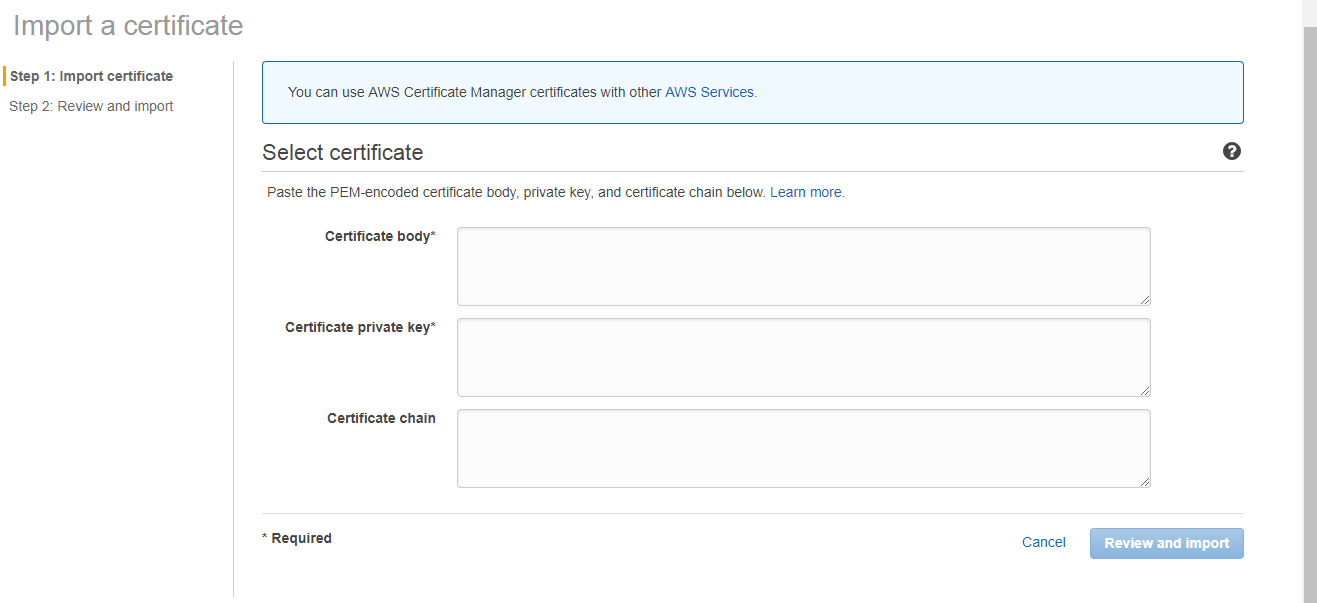
Now click on Certificate Manager.



Click on Provision certificates.



Click on Import Certificate.



In that we copy and paste the our certificate key files.

Now click on Review and import.

ISSUE 2:-

Login our Instances through Bastion host:-

In that we give the file permissions.

chmod 400 test.pem

ISSUE 3:-

when creating Auto scaling scaling group that time we choose min and max instances.

In that we can't use any Sns notifications and configure scaling groups in instance creation starting.

ISSUE 4:-

when we create S3 Bucket that time we can give bucket name as unique.

ISSUE 5:-

when we login Database in Ec2 instances. That time you install mysql on Ec2 instance. Otherwise it 's not connected.

ISSUE 6:-

when we create the Cloud Front (CDN)that time we have to choose origin Domain name ELB or S3.only one we have to choose. Once we create the distribution it will take time 15 to 20 mins.

ISSUE 7:-

In Route53 when we create host zone that time we have to choose public host zone type. Suppose we choose private host zone type. it 's working inside vpc only.

These all are I faced the issues.