AWS:

IaaS (Infrastructure as a service) Example: EC2

PAAS (Platform as a service), Example: AWS RDS

Saas (Software as a service)

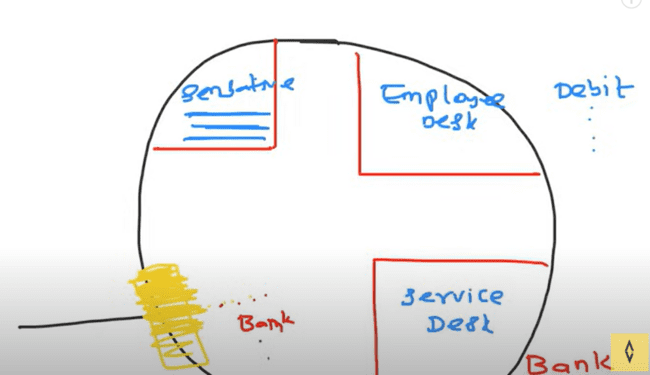
Cloud repatriation:

Moving back from cloud to use their own private cloud and setting up their own data centers.

AWS IAM:

Example: the bank area has been divided into the following categories:

Employee desk, Service desk and other sensitive area:



To enter the bank, you need to have the authentication first, and then you need the authorization like which part of the bank you can access.

IAM is the part responsible for the authentication and authorization.

For the IAM, they have main four services to offer in the bank as:

***Users, Policies, Groups and Roles.***

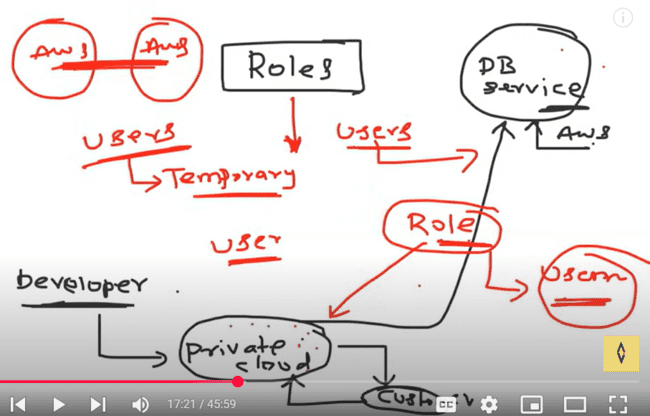
Groups are necessary, when there are users are joining and leaving the company, then you need to update the users and policies on the regular basis, you need to deal with groups in that case.

We can group like Developer, QA, DB and others.

So, in the same case, if the Developer joins the organization, we’ll simply assign the group to that user and then it is done.

ROLES: If there is one AWS service, which has been created from the DB service, and we want to access that AWS from our local cloud, then it can use the temporary roles or take that role from the AWS to access that service, and later we can delete that role.

Also, if two AWS services want to talk to each other, then they need roles for that.



Need to login as the root user in the AWS:

A screenshot of a computer

AI-generated content may be incorrect.

Search for the IAM and then go to that IAM console. Then click on the Users-> Add users:

A screenshot of a computer

AI-generated content may be incorrect.

Firstly, we would be creating the user with no permissions like that user doesn’t have any authorizations, so unable to create or do anything as shown:

* Provide the access to AWS console.
* I want to create an IAM user.
* Autogenerated Password checkbox check.

Also check this one, as users must create a new password at the sign in.

A screenshot of a computer

AI-generated content may be incorrect.

No permissions or groups assigned:

A screenshot of a computer

AI-generated content may be incorrect.

User would be displayed on the screen and one permission would be assigned to the user which is for changing the password as IAMUserChangePassword.

A screenshot of a computer

AI-generated content may be incorrect.

After next, following options would be available, like username, password, email the sign in notifications, and downloading the CSV file.

A screenshot of a computer

AI-generated content may be incorrect.

CSV file would look like this one:

A screen shot of a computer

AI-generated content may be incorrect.

While signing in as IAM user, there would be account ID after the http which needs to be put up here:

A screenshot of a computer

AI-generated content may be incorrect.

After logging in, user is unable to see the S3 buckets and permissions issue would be there in the s3 console that user does not have any permissions:

A screenshot of a computer

AI-generated content may be incorrect.

No permissions:

A screenshot of a computer

AI-generated content may be incorrect.

Now we’ll login with the root user and attach some permissions to the user.

Under Users-> user-test-501-> A screenshot of a computer

AI-generated content may be incorrect.

Add permissions:

A screenshot of a computer

AI-generated content may be incorrect.

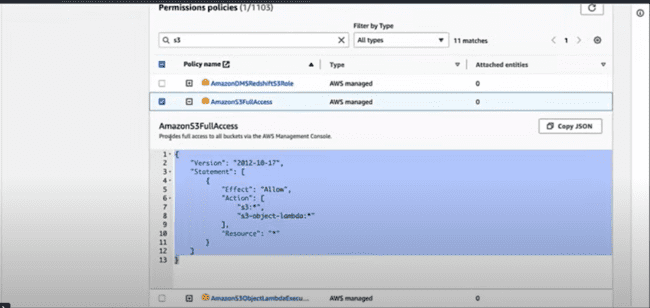
Attach policies directly:

A screenshot of a computer

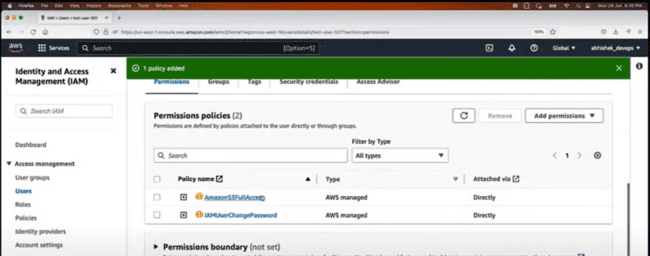
AI-generated content may be incorrect.

There are AWS managed policies, custom policies, we can add by ourselves.

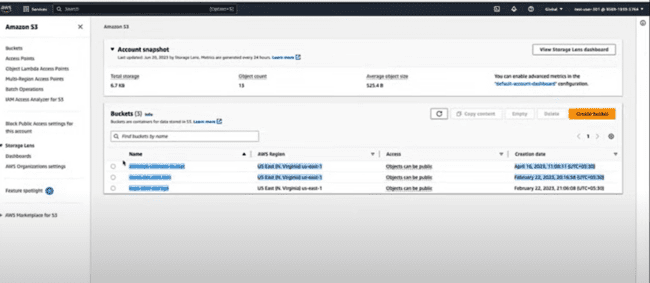
S3FullAccess policy written on the AWS console:



Provide the user amazons3fullaccess as:



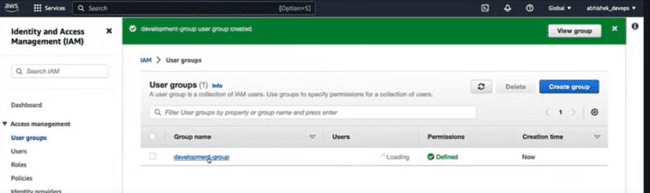
When you login with the test-user-501, we can see that s3 buckets and create as well.



**Now we’ll create the groups and whenever the user is created, that can be attached to that group, instead of doing this policies thing for every user.**

**Example: if there are multiple users with one group and want to add another group, then instead of providing that policy to every user one by one, we would assign that policy to a group in which those users have been added.**

IAM-> User Groups -> Create user group.



Click Development group -> Add users.

A screenshot of a computer

AI-generated content may be incorrect.

Now after that if users want some access, then same way, we can add that.

Click Development group-> Permissions-> Ec2fullaccess.

***EC2 Deep Dive:***

Elastic cloud computing.

Word compute in ECS means to ask AWS to providing a virtual server with CPU, RAM and Disk.

Example: If you want to use your laptop to be used by multiple people, then you need to create a hyper vision and some logical isolated machines on top of that, then multiple persons can used that.

In the similar way, AWS has multiple physical and datacenter around the world.

And in the similar way, elastic means the resources can be scaled up and scaled down as per the requirements. Like elastic Kubernetes services, elastic load balancer.

Now instead of using the EC2 instances, if you buy a physical IBM server and try the hyper vision on top of that, so that you can create the multiple VM machines, then if you are managing 1000 resources like that security issues, upgrading, server issues. So, for that reason, we can buy these things from the AWS.

WHY ECS instance: Major reasons: Maintenance and Cost.

Types of EC2 instances:

* General
* Compute Optimized
* Memory
* Storage
* Accelerated

Computed optimized is heavy computing like bitcoins, and share market operations, which requires very high computing. Or like machine learning models.

Memory optimized is used in big data analytics.

We mostly use the general EC2 instances. Charges for the EC2 instances vary on the basis of their types given above.

Data centers and Regions:

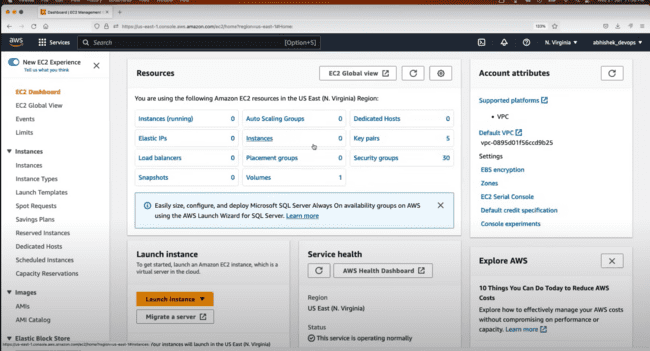
If you are placed in India, and created the instance or selected the US region, then there would be high latency observed, so it is better to keep the data center at the nearest location.

**If all your instances are in the one region…...?**

There is term called Availability zones, that means, Mumbai might have two zones, one is East and other one is West, so, they can create the instances in two zones, and one can provide quick results with no latency, while for other zones, we can except some latency. But when one goes down, the customers can be redirected to the other zone.

**CREATION OF EC2 Instances:**

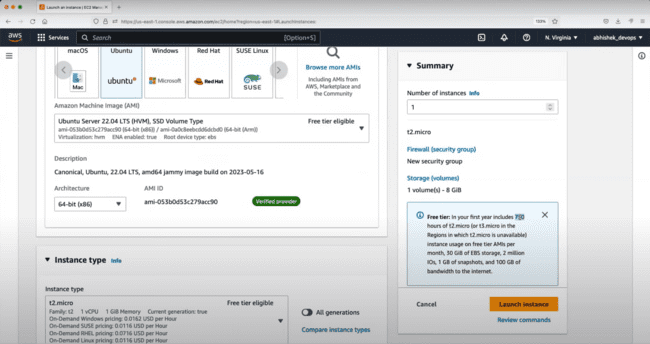
Search EC2-> EC2 dashboard. Then click on the Instances -> Launch instances.



A screenshot of a computer

AI-generated content may be incorrect.

AWS provides the one year of 750 hours of t2.micro usage free of cost. After that, it starts charging.



But there is only one condition as, we need to make the use of one instance only. If we need to use other instances, then we need to use make one instance stop, and other instance run.

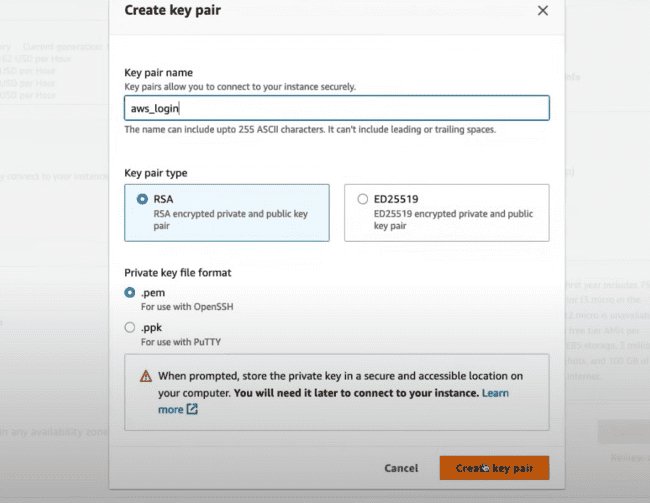
Key-value pair in the AWS:

Key value pair is the combination of the public and private key which uses to login to the specific instance.

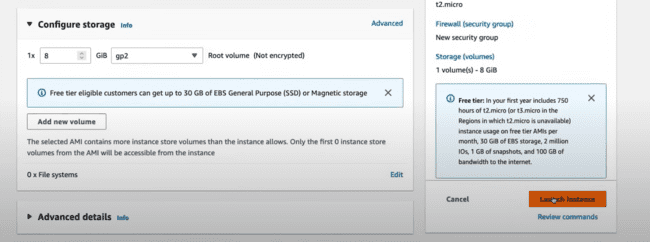
Using the private key, we can login to the instance.

We can share the public key, but private key should not be shared with anyone.

Click on the create a key value pair and then do the following:



We can also increase the RAM as shown, but by default functionality is 8 GB, as shown below:



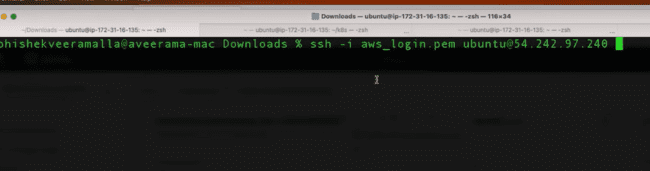
We need to use the putty or the Mobaxterm to connect.

When the instance starts running, then click on the instance and following are the instance details:

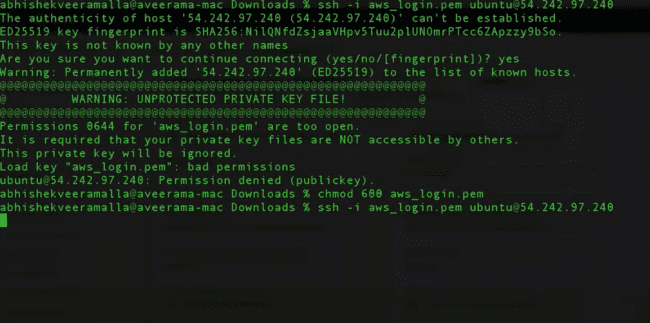
A screenshot of a computer

AI-generated content may be incorrect.

Need to locate the .pem file to login to the AWS using the following command:

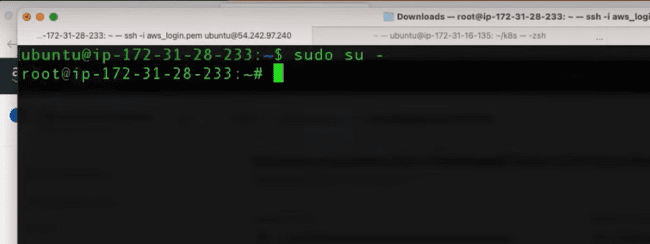


Permissions are not there, so we need to change that and login again.



Now if run the command using the whoami, which would returning the logged in user.

To switch to the root user, we can do that using the following:



If we are login as the super user, then we need to update the packages using the following command:

* apt update

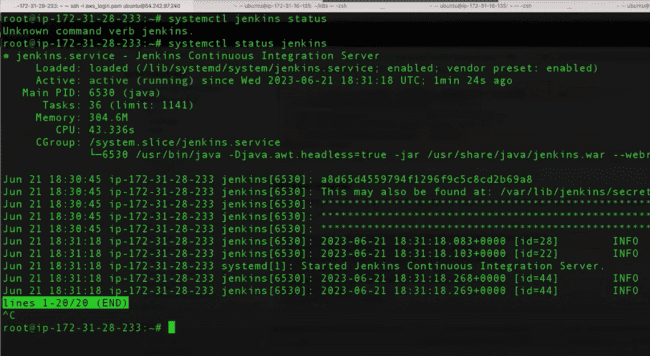
And if we are not logged in as root user then we need to use the following command:

* sudo apt update

Now install Jenkins on the command prompt using the command prompt on the Jenkins site.

To check the status of the Jenkins service running or not, we can check that also as:

* systemctl status Jenkins



Now, if you try to open the public address using 8080 ports, that would not be accessible, as there are lot configurations involved.

Under Security-> there are inbound rules and outbound rules:

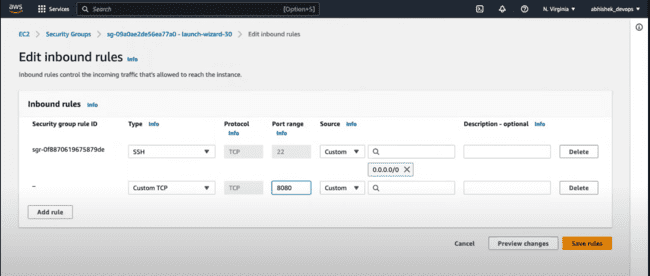
Inbound rules: request is coming into the AWS.

Outbound rules: request is going out to the AWS.

And now go to the Security groups:

Under Inbound rules, edit the inbound rules and then Add a new rule.

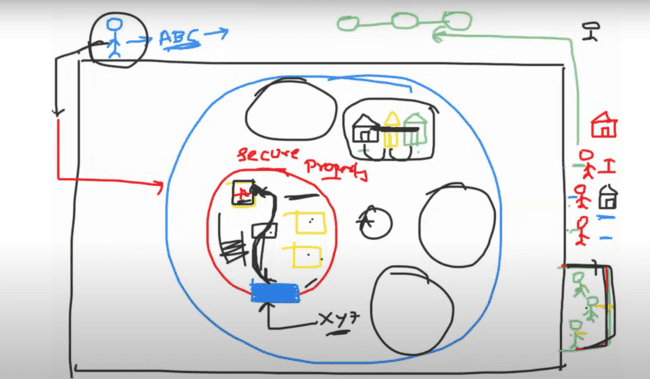
And then save the rule.



Now if you are trying to access the JENKINS, we can access that using the public IPv4 address.

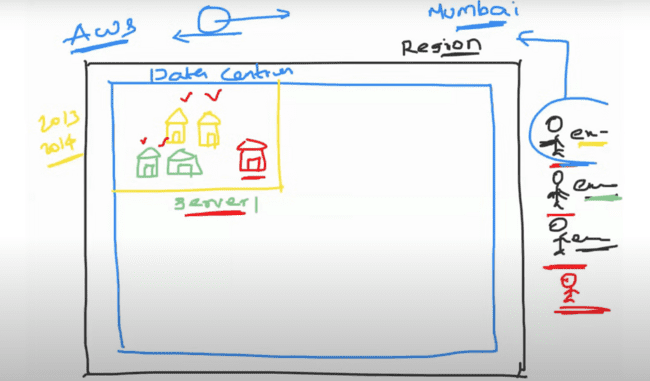
**VIRTUAL PRIVATE CLOUD(VPC):**

Secured area, houses in the secured area, how to access the houses in the secured area.



Like three companies wants to get their site hosted on the internet, they want some virtual machines, so ask AWS to get that.

For the same, AWS provides them all the necessary resources and data center for that.

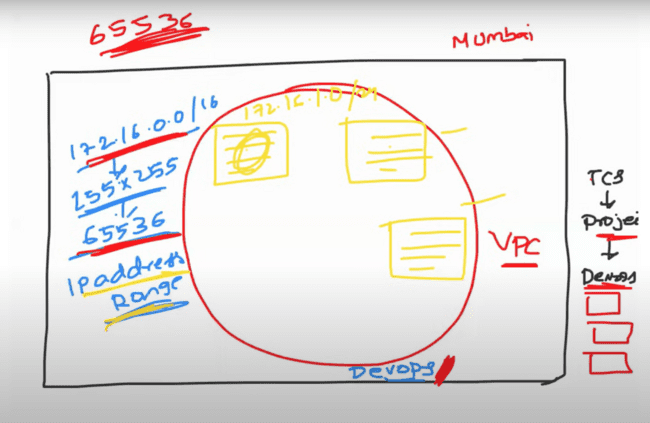


Till 2013, 2014, suppose one machine was accessed by some hacker on the server 1, so he would be able to access the resources of other companies as well.

So, they come up with the concept of VPC, which needs to operate by the DEVOPS engineers.

Size of VPC needs to be defined using some range of IP addresses.

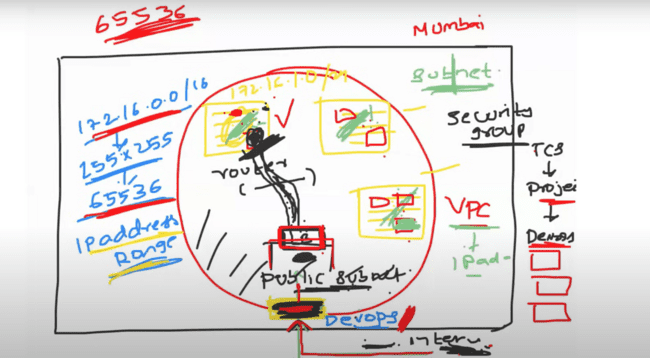
Example: there are multiple sub-projects under the one, so they can provide each one with the IP addresses, so that they can be identified like the following:



After that, DEVOPS engineer needs to define the gateway which needs to be authenticated to access the VPC’s.

**User enter the public subnet using the internet gateway. After that request routes to the specific path using the route table through LOAD BALANCER. As, LOAD BALANCER does not know where to send the request, so it does use the route table for that.**

**After routing, there is one security group which needs to be validated for the AWS, as it needs to check from which port request is coming to the AWS.**

****

NACL: like there is same security mechanism for many servers inside the secured area, which can use NACL there.

NAT GATEWAY: it is masking the IP address of the private subnet server when it tries to access some resources from the internet or public area, so it does not expose its IP address to that site, instead it masks the IP to the public one, and may provide the IP of the load balancer or router.

If it does so by using the Load balancer, we call that SNAT.

But, if it does use the Router, we call this NAT gateway.

**Now, if you want to debug the whole things above, and wants to know how the traffic is going and how the requests are coming and accessing, then you can use the VPC FLOW LOGS for that.**

**DAY 5: SECURITY GROUP AND NACL**

While creating the VPC, we can define the range of IP addresses which mean its size and inside VPC’s there are multiple subnets.

So, basically, private subnets are accessed from the user in the outside that VPC’s using the GATEPWAY, then that request goes to the public subnet and then LOAD BALANCER routes that request to the private subnet using some conditions and validations.

If you add more security on the private subnet, you can use the NACL’s.

AWS says that security is a shared responsibility. AWS would be providing the VPC’s, security groups, NACL, API gateways but also it needs AWS DEVOPS engineer and admin needs to work alongside to manage the security.

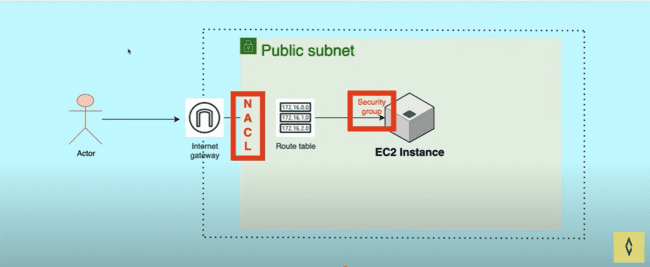
A whiteboard with black text

AI-generated content may be incorrect.

**EC2 instance comes under the private subnet. And private subnet adds a layer of authentication which ports to allow and which to deny.**

Now, instead of applying security to all EC2 instances, one by one, you can add the same in the private subnet and that applies to all the EC2 instances which may be very huge in numbers.

Major differences between NACL and security groups is SG’s have rules only for allowing, while NACL have rules for allowing as well as denying the traffic as well.



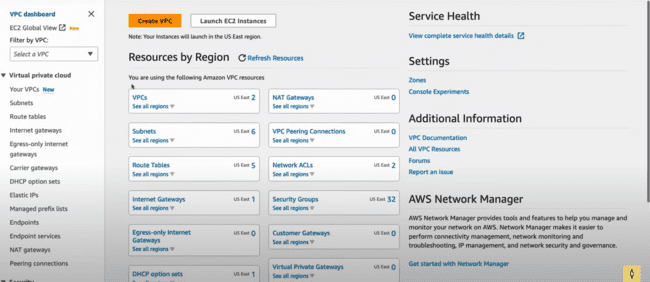
Security group is applied at the EC2 level, but NACL is applied at the subnet level.

As we need to apply the security group to the 1000 EC2 instances one by one, but we can apply directly these to the different subnets using the NACL’s.

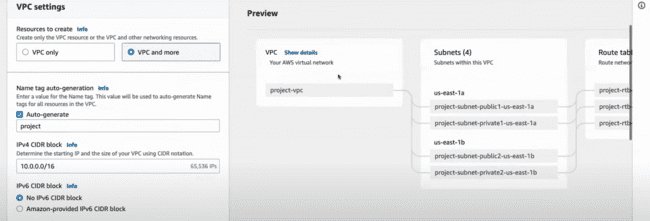
AWS by default creates the internet gateway, NACL, subnet and the route table, we need to create the EC2 instance and add the security group to the same.

And then we need to check how NACL and security group are interacting with each other.

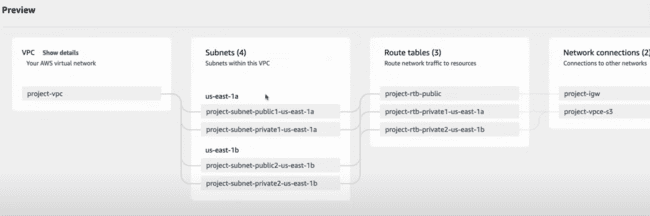
Create a new VPC.



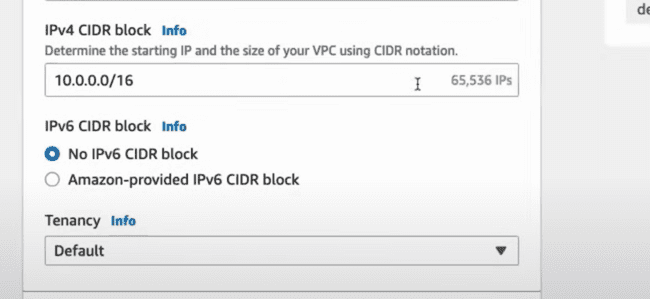
We need to select VPC and more while creating the VPC.



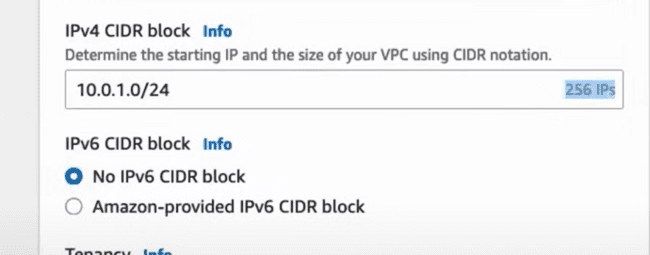
AWS would be creating the public and the private subnet as a combination and route tables as well.



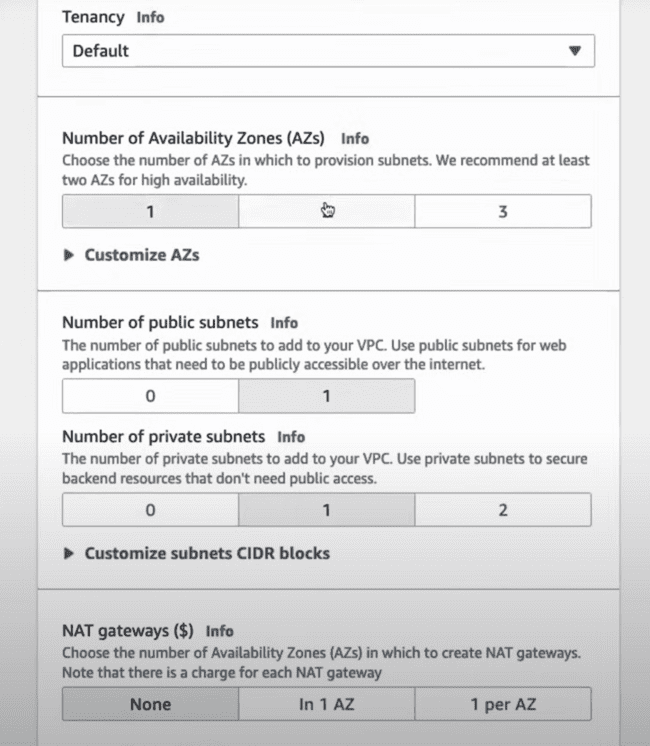
This is the range of IP addresses we need to provide while creating the VPC’s. We can create the range from the number of IP addresses, which we can use.



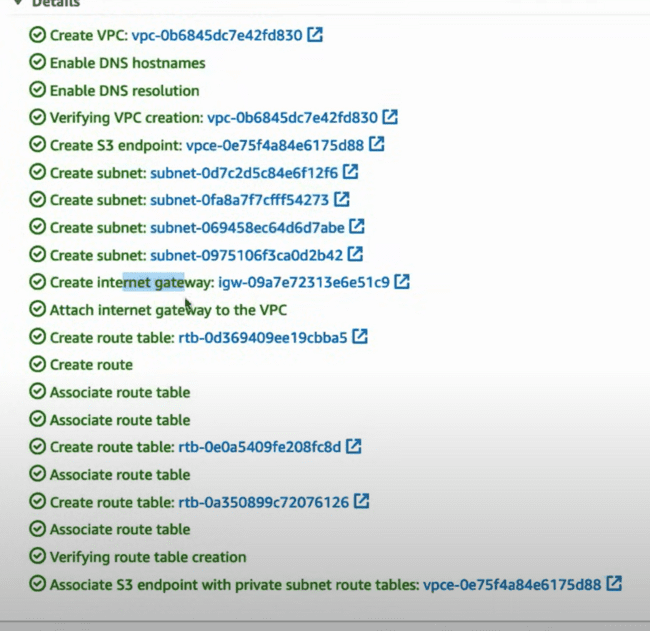
Also lower the IP addresses as well.



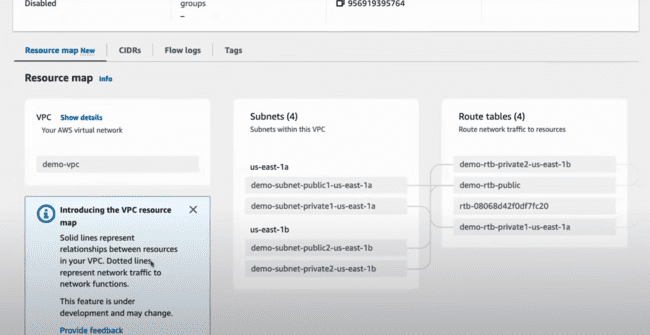
Availability zones, public subnets, and number of private subnets, you can configure.



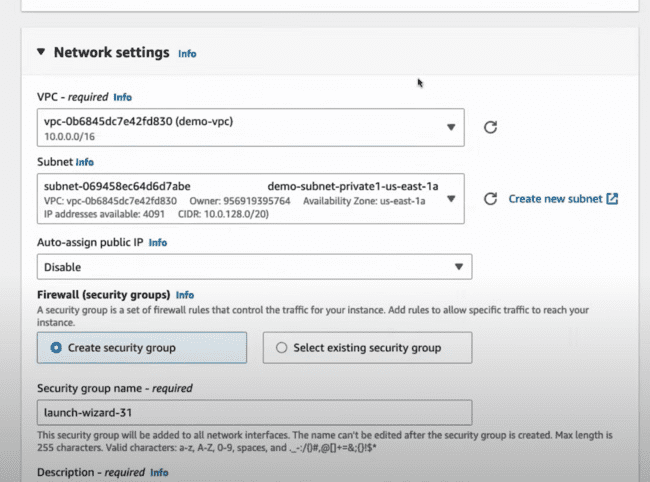
VPC starts creating the configurations:

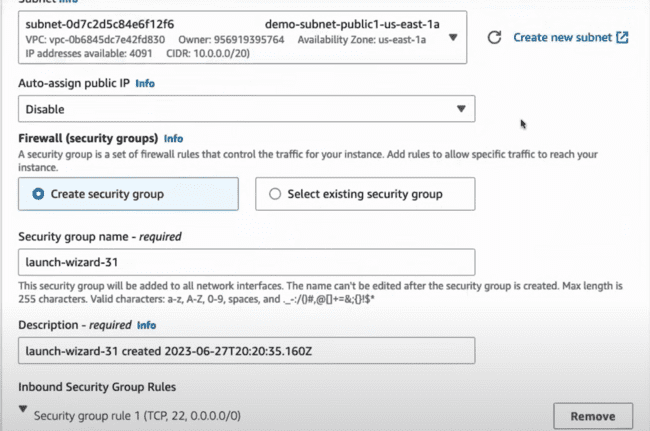


We can also check the resource map of the VPC created as follows:

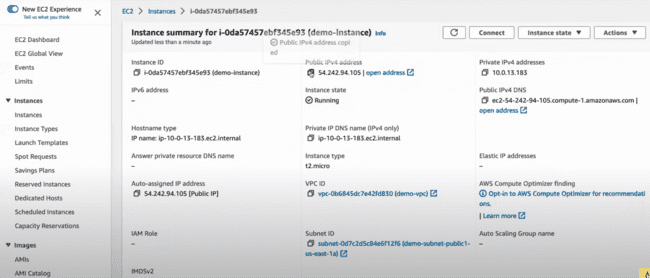


Now creating the EC2 instance. All the streps are same, except in the Network settings, we need to add the VPC created above as shown below and change the subnet from private to public one.

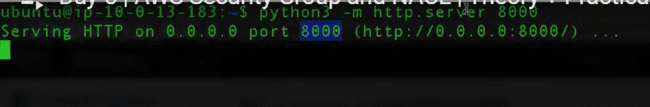




EC2 instance starts running:

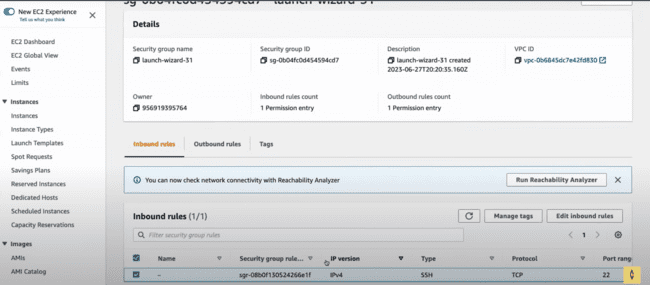


To running a simple server on the http using the python at the port:8080, we can use the following:



But when you try to access that application is not accessible.

And default security group attached to this one is:



Now look at the NACL configuration under the VPC.

A screenshot of a computer

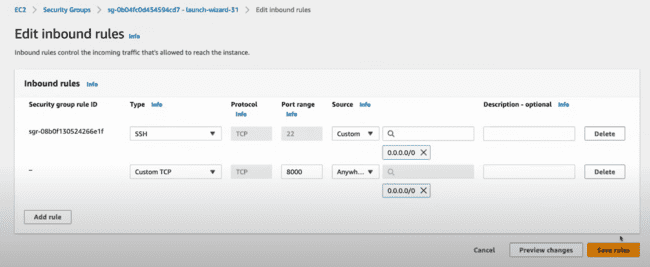
AI-generated content may be incorrect.

Inbound rules:

A screenshot of a computer

AI-generated content may be incorrect.

Now NACL is allowing, but EC2 instance is blocking the 8080 port request. Adding a new inbound rule.

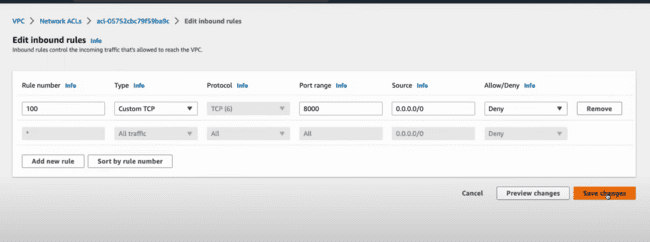


And the page is accessible:

A screenshot of a computer

AI-generated content may be incorrect.

Now, if we add a new rule saying everything coming from the internet, just block that, so we add that rule using the following:

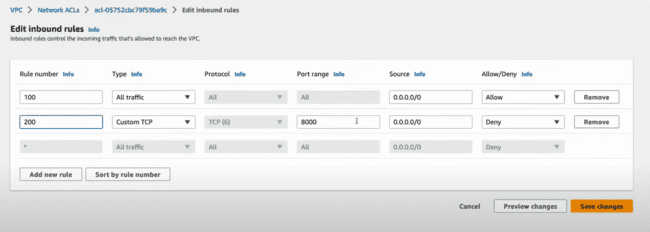


Not accessible:

A screenshot of a computer

AI-generated content may be incorrect.

Now when you create new rules, one for denying and one for allowing, so order would be necessary for the same:



In the above case, 8000 would be accessible.

A screenshot of a computer

AI-generated content may be incorrect.

**ROUTE 53 on AWS:**

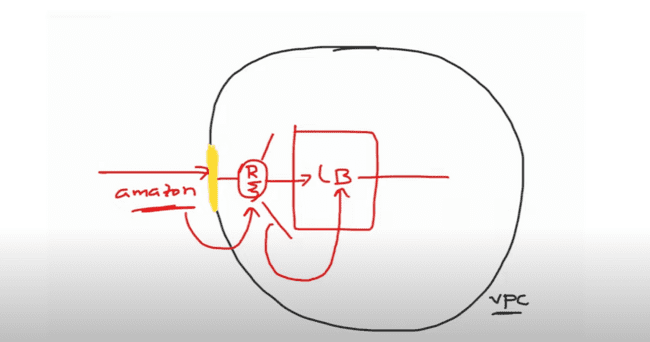
DNS service is basically converting a name to the IP address of the server, on which it needs to send out the request. It resolves the domain name to the IP address.

**DNS has lot of records which maps the name to the IP address. And most important thing is AWS provides this DNS as a service.**

Why it is required, as DNS is very complicated and if you have written one application in your local machine and you want to host that on the net, then you need to buy a domain and then you need to find the hosting solution and also some IP addresses also needs to be maintained by yourself.

To solve this problem, AWS provides the ROUTE 53 for this one.

Now in front of the load balancer, we add a layer of ROUTE 53, which allows to resolve the domain name to the specific IP address and then redirects to the LOAD BALANCER.



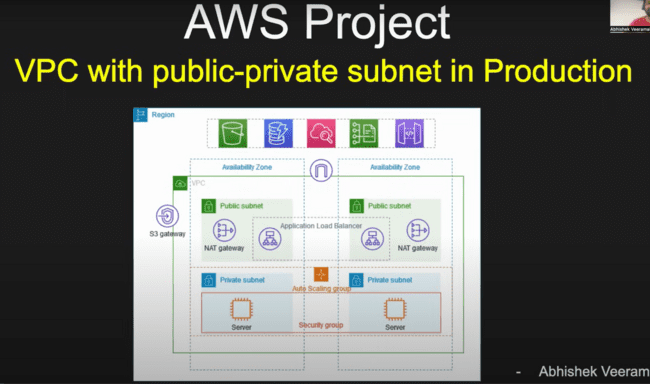
To configure the ROUTE 53, we need to do the following:

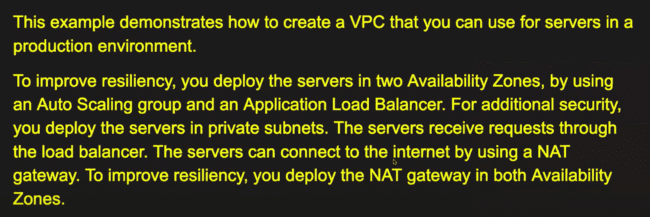
Domain registration.

Hosted zones: Mapping of domain names with IP addresses.

Also perform health checks of multiple servers by sending out the request to the servers after an interval of time to maintain their health.

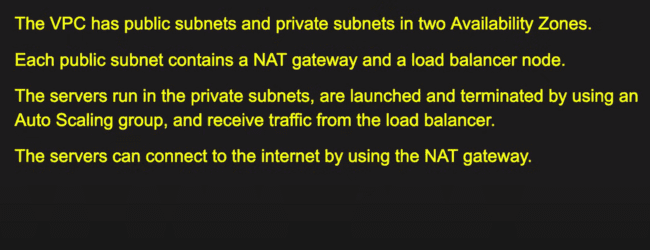
AWS production grade application:





A screenshot of a computer

AI-generated content may be incorrect.



These things we need to learn before starting:

A black background with yellow text

AI-generated content may be incorrect.

AUTO SCALING:

If you want to deploy your application in two availability zones, we need to create the two EC2 servers.

So, what AUTO SCALING GROUP suggests instead of creating the two EC2 instances, we can create the minimum of two replicas and manages the requests and if, in future, the two servers cannot handle more requests. Example: two servers can handle 200 requests, then auto scaling group can multiply them to 4, 5 or 6 in number to handle the more incoming requests.

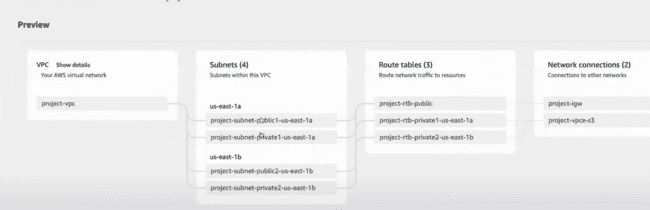
Load balancer is just balancing the load between the two servers.

**BASTIAN HOST OR JUMP SERVER:**

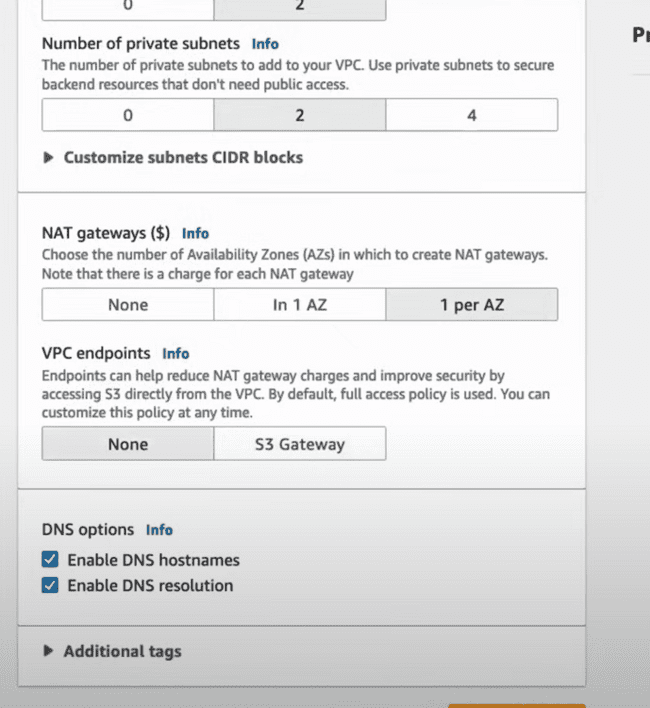
To connect to the private subnets, we cannot connect directly, so we need to connect from public subnets using the BASTIAN HOST. As it follows proper logging mechanism, auditing and who is accessing this private subnet. This information we can access using the BASTIAN HOST. And we can configure a bunch of rules to check how the traffic is actually flowing.

Creating the VPC first:

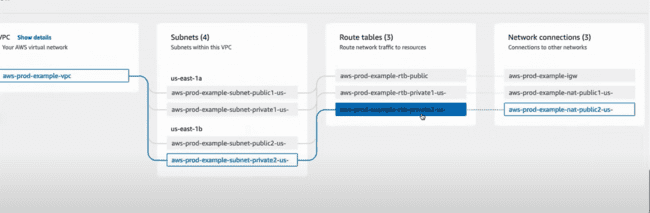
Create VPC and select VPC and more option, as it creates the public and private subnet itself.



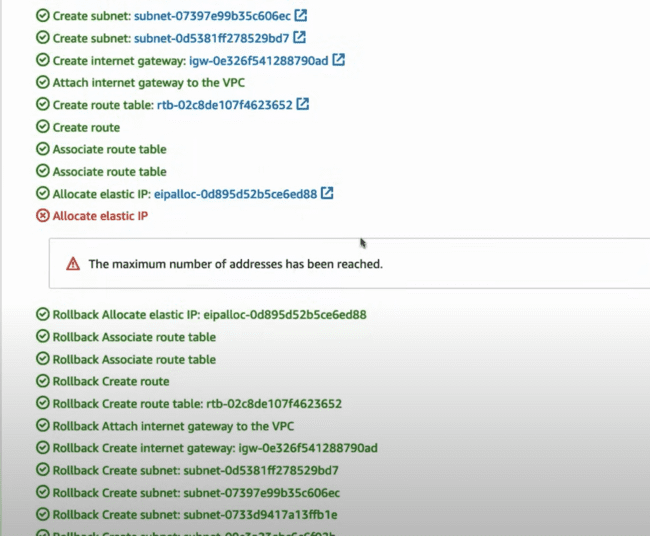
Now select one NAT gateway as “1 per AZ” and VPC endpoints as none.



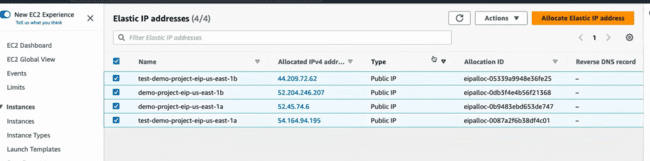
When selecting VPC end points as none, the diagram also changes.



Now while creating this one, an issue comes as shown below:

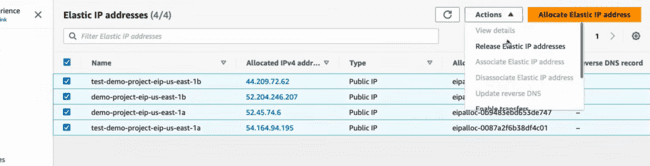


Under the EC2, we need to release the elastic IP’s as shown below:



Under actions-> release IP addresses.

**Elastic IP is a static IP address, which remains same even when the EC2 instance is released.**



After releasing the IP addresses, the instance would be created.

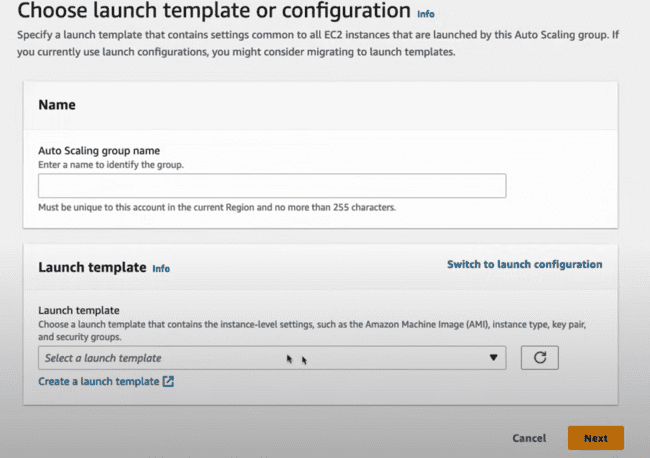
NAT gateway is mapping the IP address of the private server to the public NAT gateway, this is where VPC is using the **ELASTIC IP address**……?

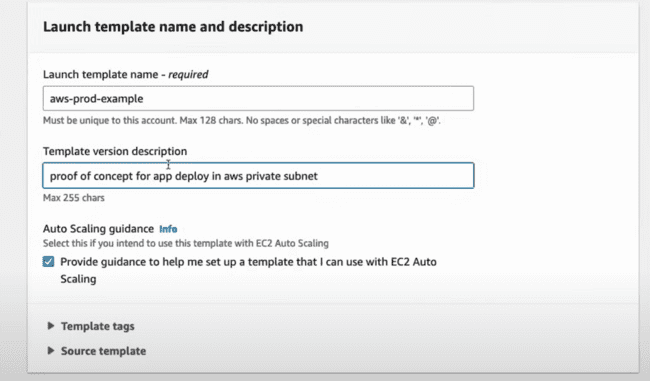
After creating the VPC, we need to create the load balancer, Auto scaling group and EC2 instances as well.

Creating the auto scaling group:

EC2-> Auto Scaling Groups -> Create auto scaling group. And cannot be created directly, we need to use the launch template for the same.

And that template can be used for multiple instances, and we define the behavior of the auto scaling group her, like how many instances, it can scale up to.

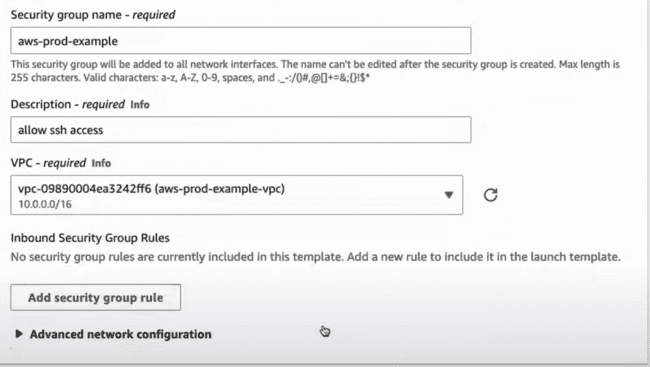




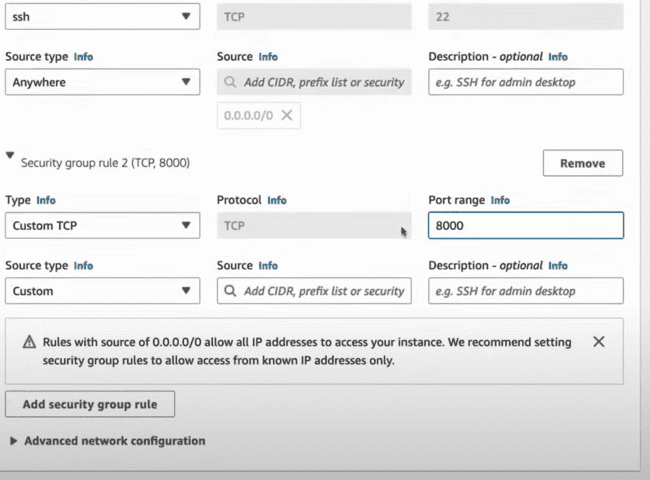
AIM, instance type and key value pair would be same.

Under Network settings, create a new security group.

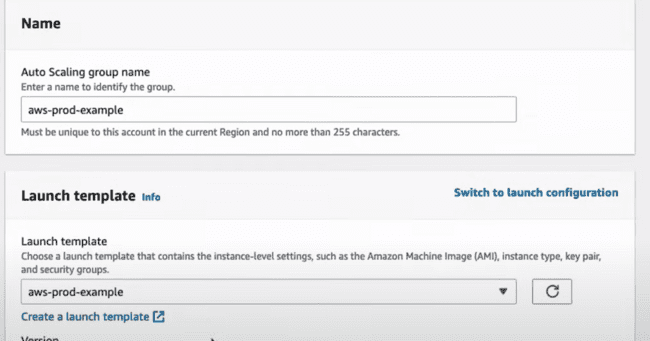
Use the name of the VPC just created.



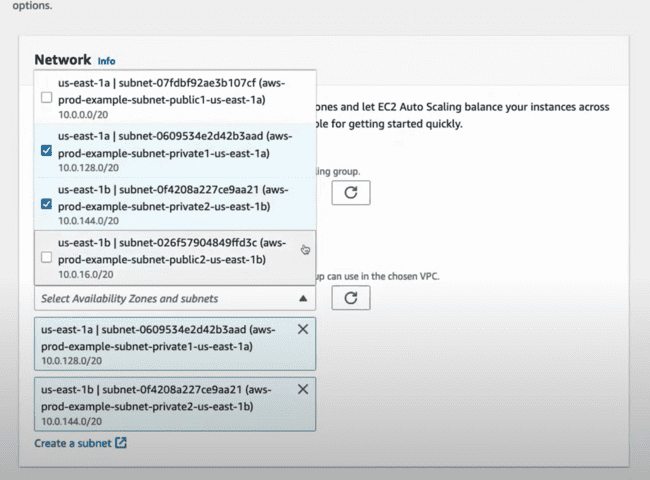
Custom port 8000 and 22 port we need to include.



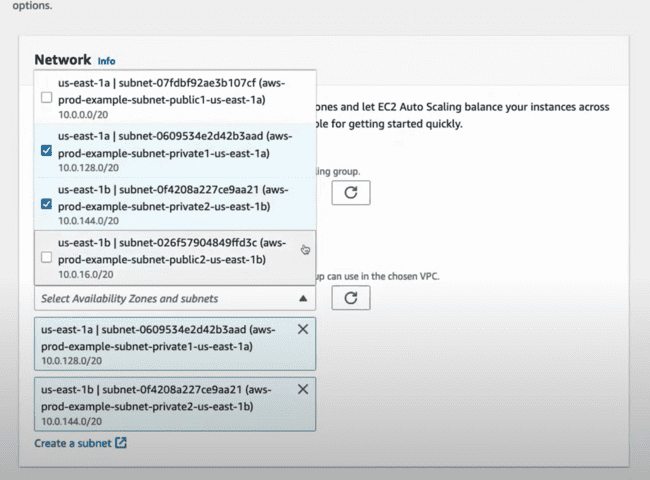
After creating, select the same from the launch template as shown below:



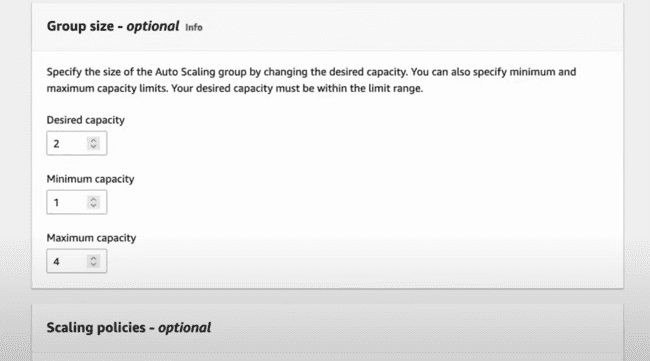
Under Network, select the VPC, Availability zones and subnets and choose them in the private subnets:



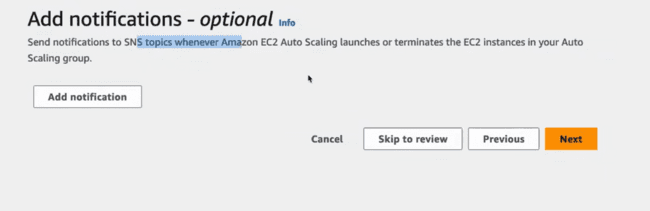
After clicking the next, select No Load Balancer, as we’ll not be creating a load balancer for the auto scaling group and will be creating for the public subnet.



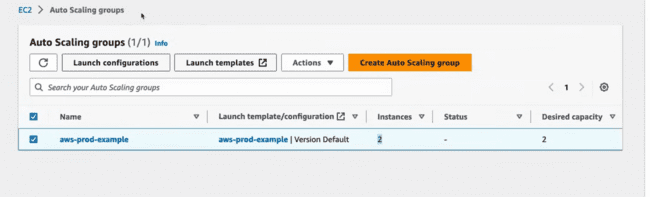
Desire capacity and maximum capacity which is when there are too many requests needs to be handled.



Following is for the sending out the notifications when EC2 instance would be launched or when it would be terminated:

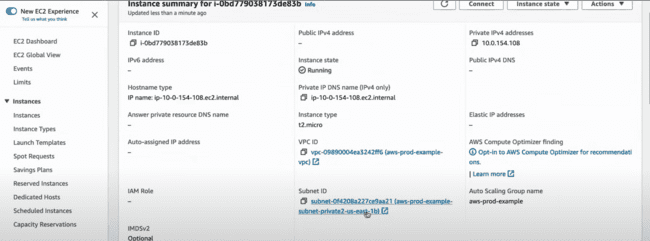


Auto scaling group would be created:

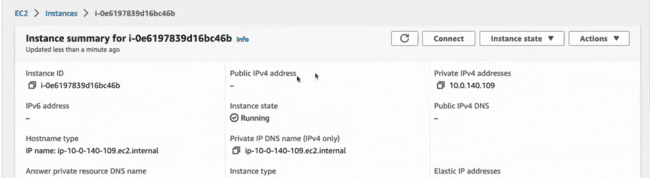
****

For all the instances, there are subnet ID created:





Now when you open the instances, then you can check that there is no public IP available to those instances.



**Now the important part is how to login in these instances, we need to use the BASTIAN HOST comes into the picture….**

Creating a EC2 instance and launch a new instance.

Everything would be same which has been done for the EC2 instances except the following:

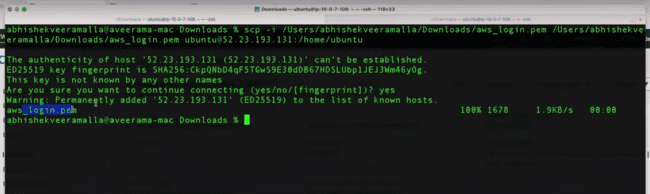
Edit the Network settings and make the VPC same as the VPC created. Auto assign public IP would be “Enable”.



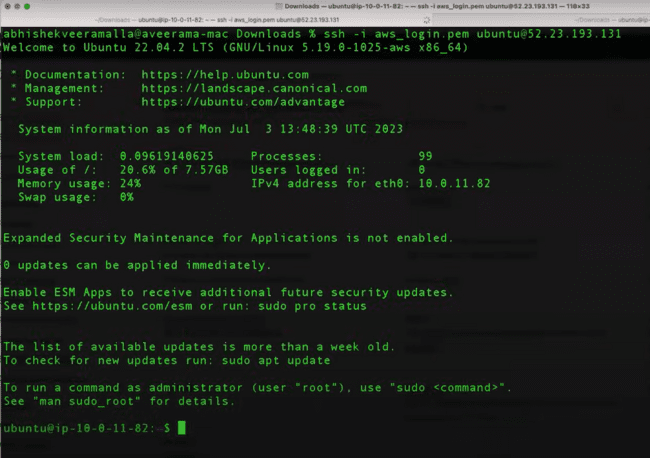
LANUCH the instance.

After that we need to SSH to the Bastian host and then from there we need to do SSH to the private subnet using the key value pair.

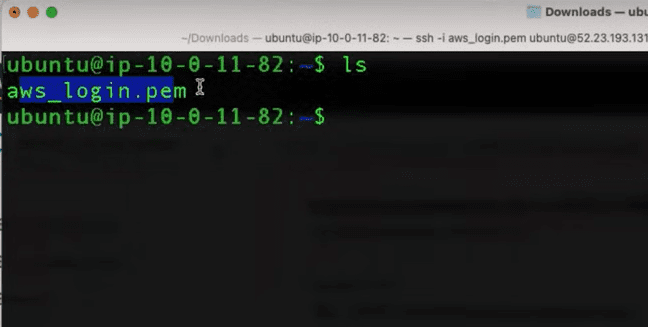
Copying the .pem file for the private subnet to the Bastian host using the secure copy as:



Now we need to do the SSH to the .pem file present on the ubuntu host.

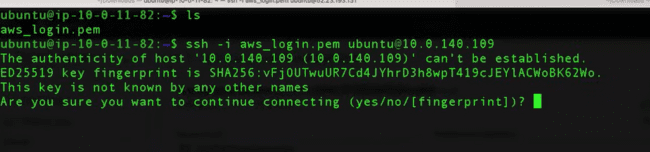


.pem file is available here, so we need to use the same .pem file to login to the private subnets.

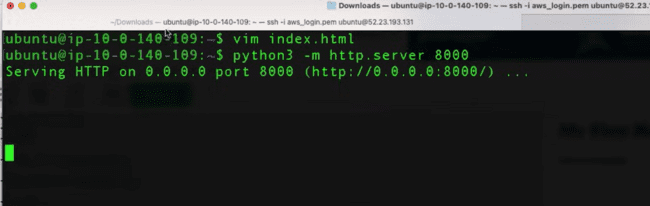


Now we need to install the python application to one of the instance and login them using the private IP address.

SSH to that



Create an index file and run the python application on the same.

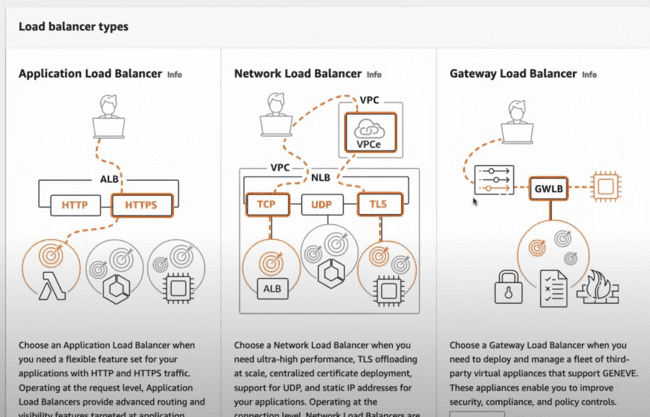


We have created the application on one instance only, as we need to check the functionality of the load balancer, as we need to check the functionality of both the load balancers. As 50 percent of traffic would go to the one instance, and we get the response and 50 percent of traffic goes to other and we did not get any response.

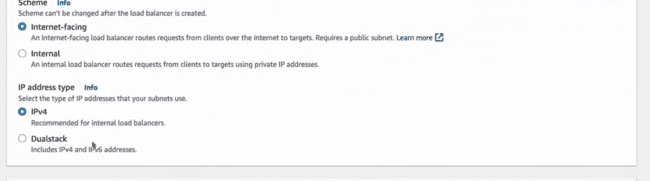
Now create a load balancer:

EC2 -> Load Balancers -> create a load balancer.

Three types of load balancers are there:



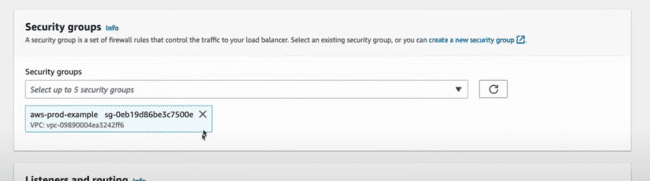
Select the following:



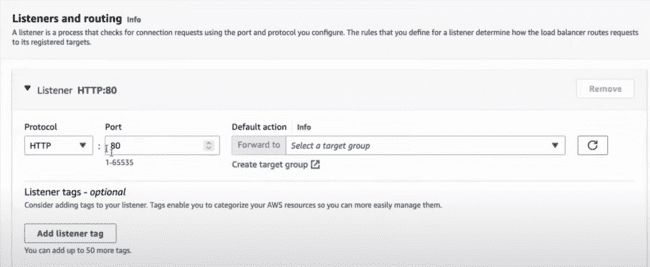
Under network Mappings, Select VPC created and change that to the public subnet as:

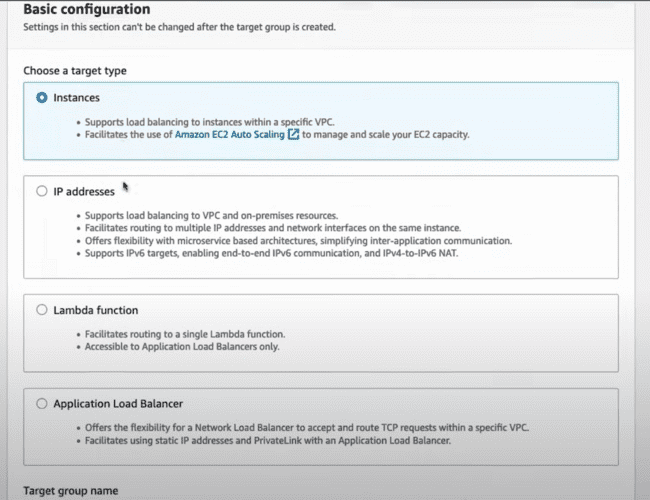


Security groups can be any:

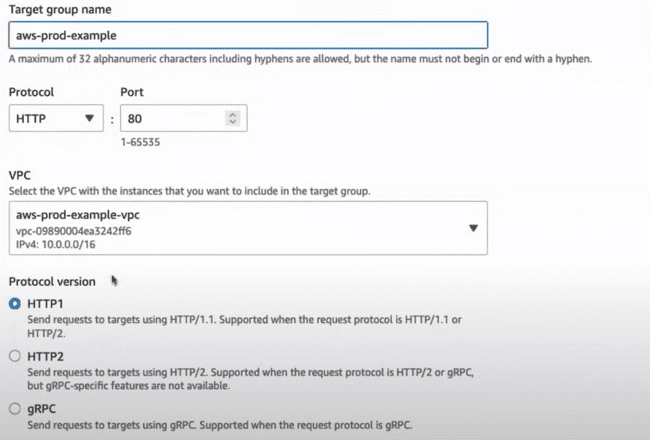


Under Listeners and routing, we need to create a target group:

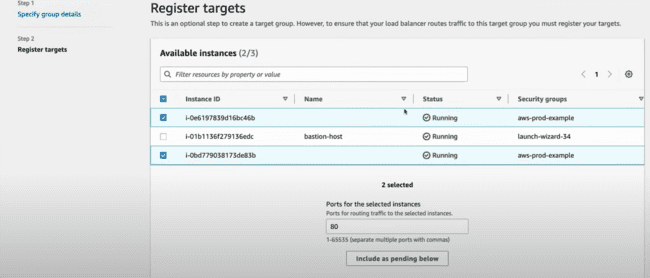




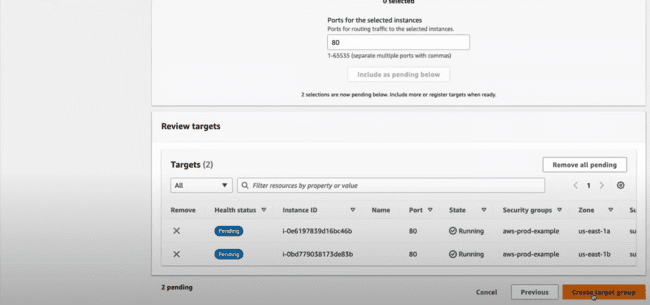
VPC would be the same and provide the target group name.



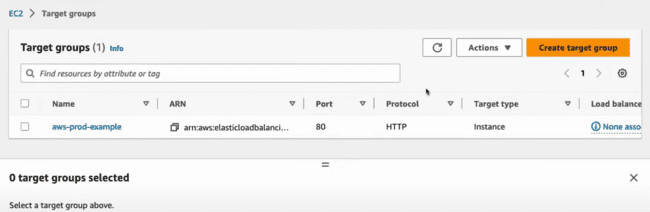
Click next and move to the other page and select instances.



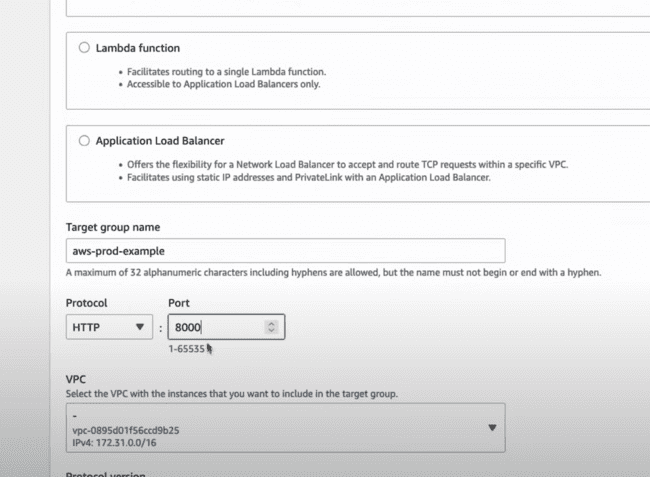
Click on the include as pending and create the target group.



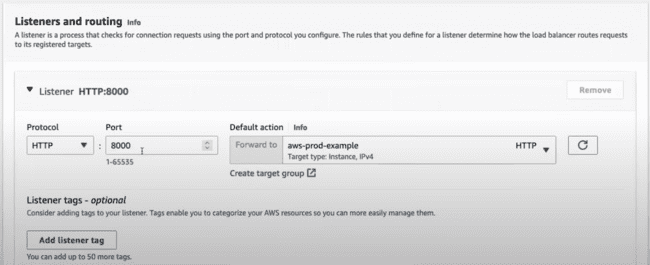
Now the target group is at 80, we need to configure this to the 8000.



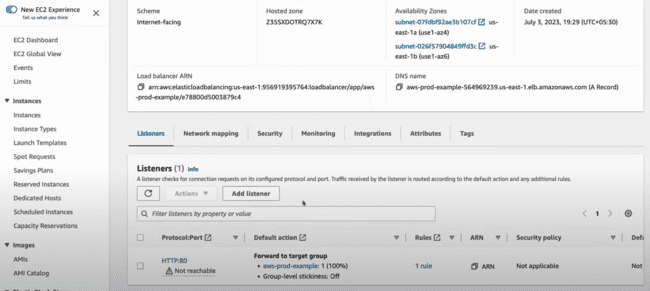
Delete this one and create a new target group using the port 8000 as shown below and other steps would be the same.



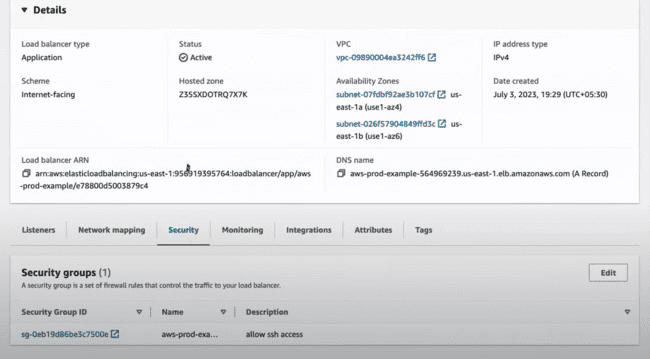
And come back to the load balancer and use the same target group for the load balancer.



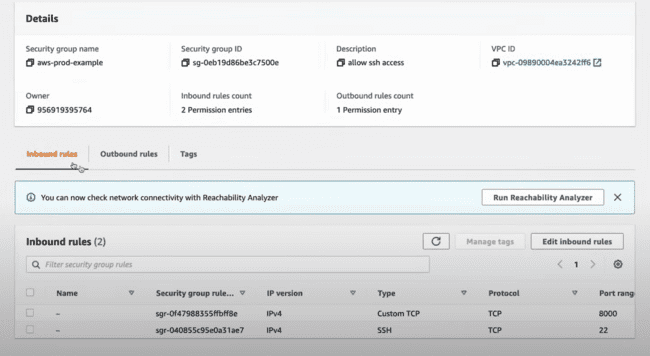
When the load balancer starts, it would give you the following error:



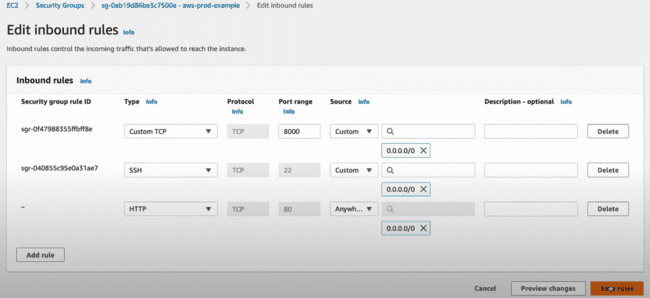
Go to the security group and allow the HTTP traffic on this one.



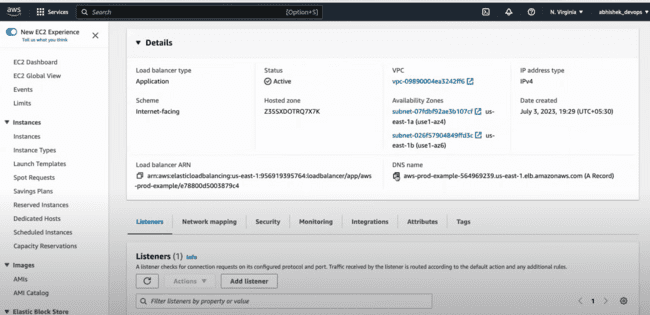
Click on the security group ID.

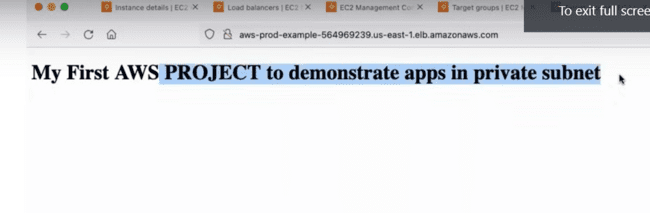


Edit the inbound rules:



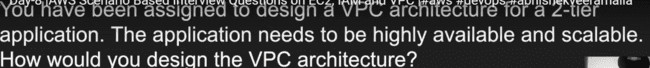
Now when the Load balancer starts, access that from the outside world from the that DNS NAME on the screen:





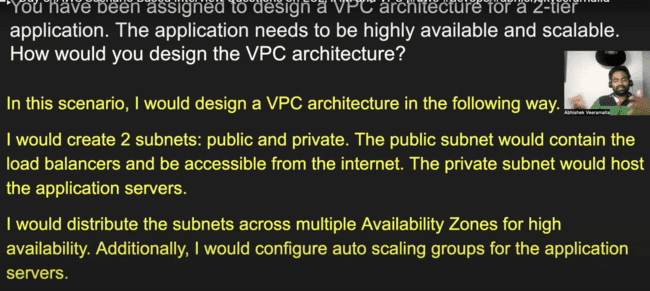
**AWS INTERVIEW QUESTIONS:**

Question:

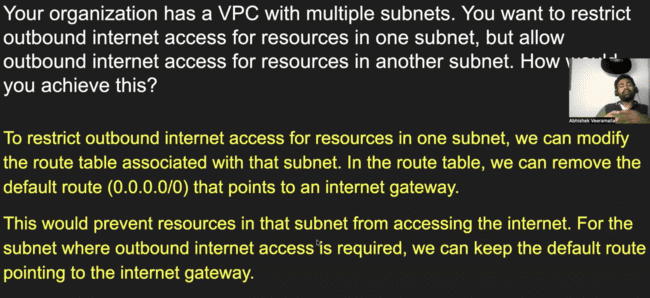


2 tier architecture involves the FRONT END and the BACKEND.

While 3 tier application contains FRONT END, BACKEND as well as DATABASE.

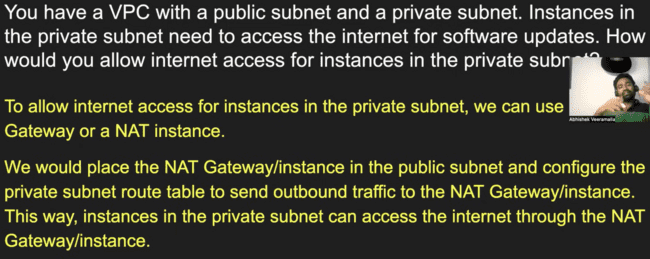


2nd question:



NAT is NETWORK ADDRESS TRANSLATION.

Question 3: NAT GATEWAY takes the IP address of the private subnet and translates it to the public IP address of the NAT GATEWAY and sends out the request. Due to this, the details of the private subnet are not shown to the other website.



QUESTION 4:

