AWS DevOps

INTERNSHIP

MINI PROJECT - 1

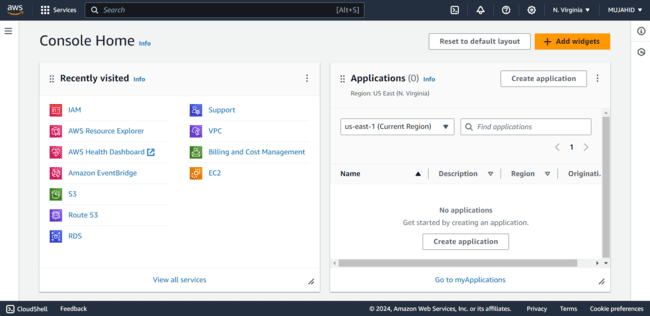
MINI PROJECT – 1

Lab – 1 [IAM HANDS ON]

* *Log into AWS management console.*

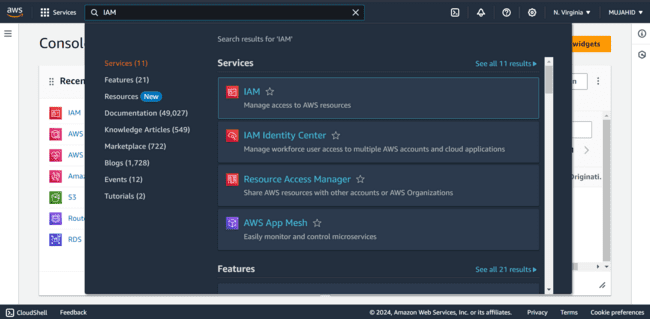


* *I logged in with my AWS credentials and entered into my AWS management console.*

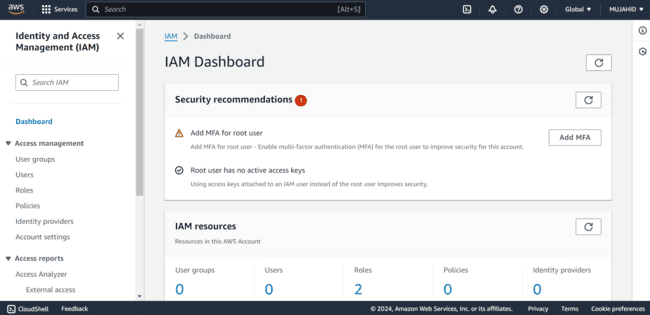


***Setting up Multi Factor Authentication (MFA):***

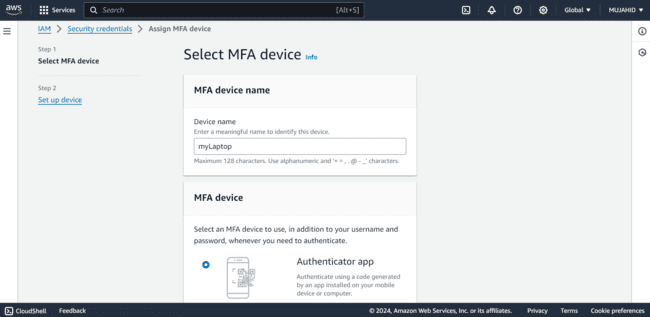
* *On the Services options type IAM and click ENTER.*

******

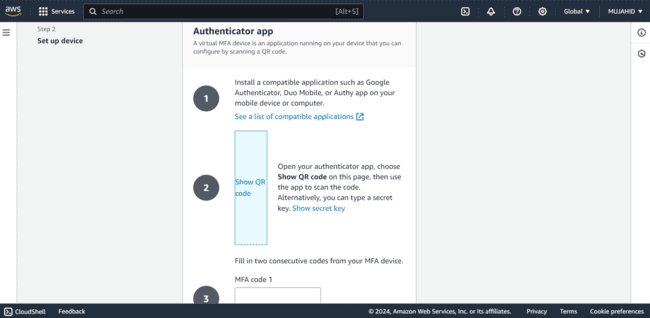
* *After Entering you see an option Add MFA for Root User.*
* *Click on Add MFA.*

******

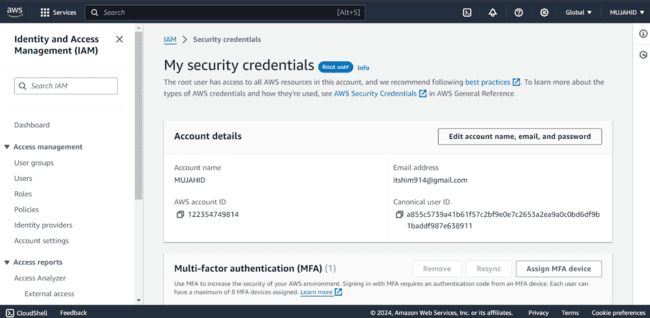
* *Clicking on it opens a window asking for a device name.*
* *Give an appropriate name and select Authenticator App below MFA device option.*
* *Click on Next.*

**

* *Download Authenticator app on your mobile device.*
* *After clicking on next shows some Options to perform to setup Authentication on your mobile device’s Authenticator app.*
* *Like scanning the QR code and entering two MFA codes.*

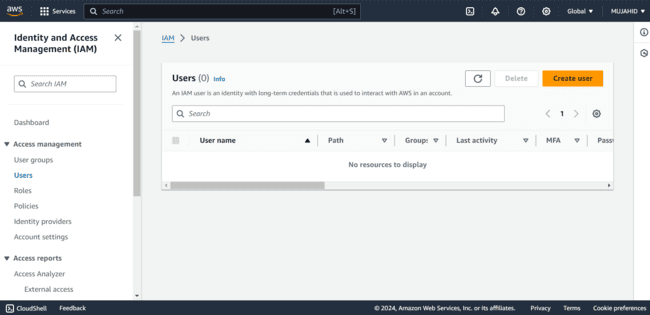


* *You can either scan the QR code or manually Enter the secret key.*
* *This adds a new device on your Authenticator app with the device name that you chose.*

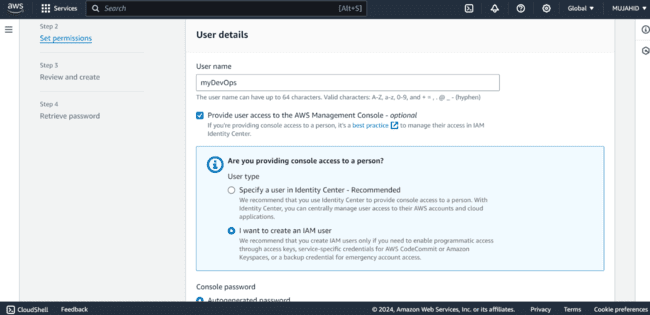
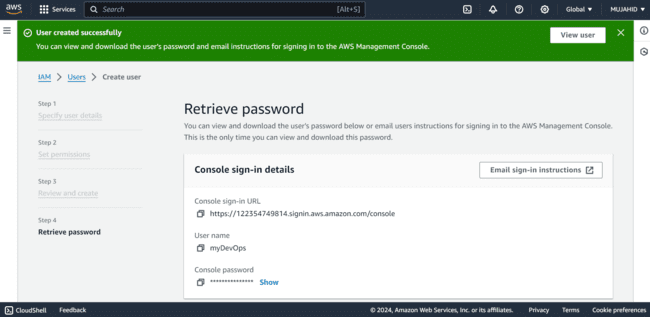


***Creating a User with Console Access:***

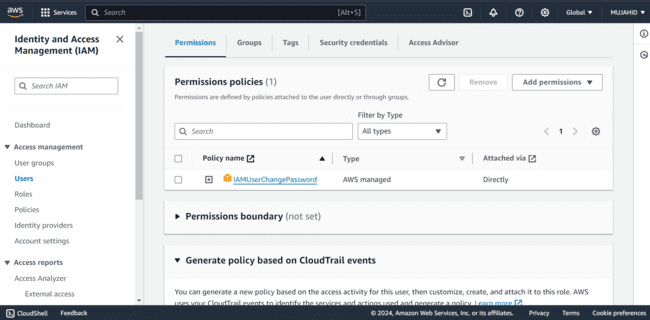
* *On the Left click on option Users.*



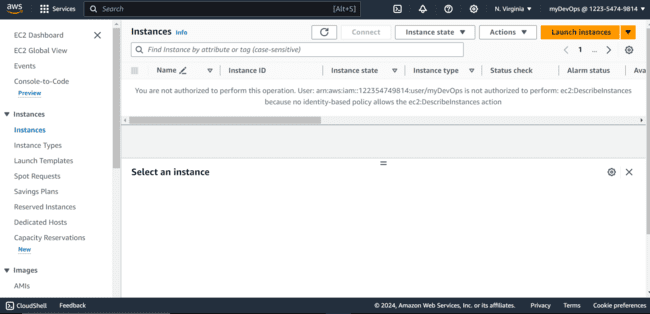
* *Click on create user.*
* *Give a Username and put a check mark on Provide user access to the AWS management console.*
* *Click on I want to create an IAM user.*
* *Setup a password either Autogenerated or Custom for this user.*
* *User is successfully created with only console access and password changing permission for this user.*

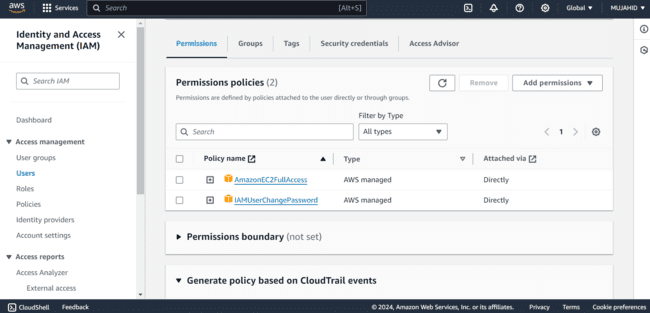
* *We can add MFA to this user like we did to the root account or we can just directly sign in from AWS management console.*
* *Check the default permissions of this user in the permissions tab after clicking on the user that we’ve created.*

**

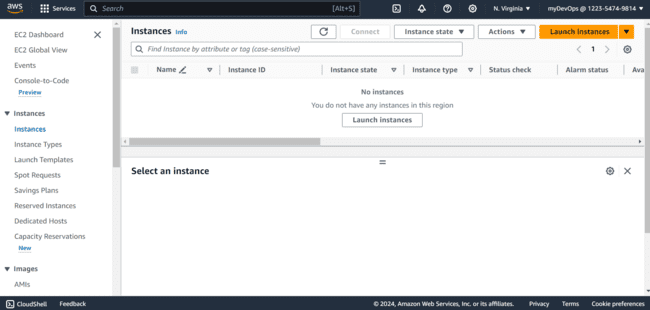
* *Sign out and Login again as the IAM user that you’ve just create and check if we can Explore other services.*
* *Shows error obviously cause the user does not have any permissions to explore other services.*

**

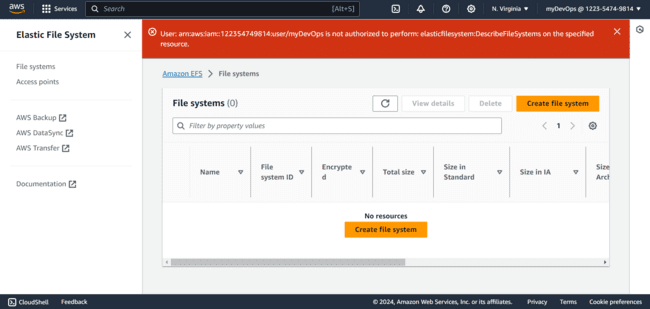
* *Log back into the root account with AWS credentials and MFA.*
* *In the services menu Enter IAM and open the users default permissions tab.*
* *Click on Add permissions and select “attach policies directly”.*
* *Enter AmazonEC2fullaccess and put a check mark on it.*
* *Click on Add permissions.*

**

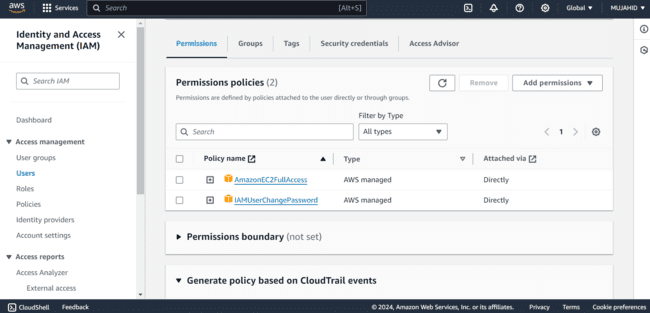
* *Sign out and Login again as IAM user that you’ve just Created.*
* *Explore the EC2 service, now we can access EC2 services only.*

**

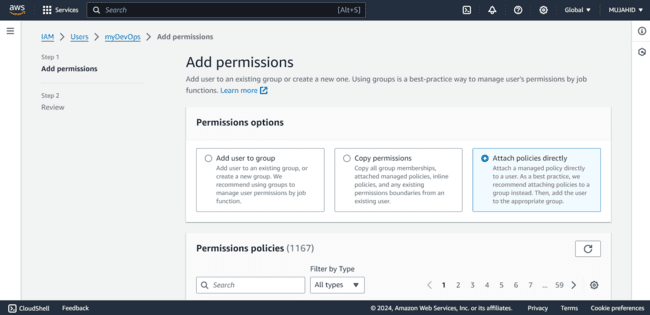
* *We can check that we do not have permissions to Explore any other services.*

**

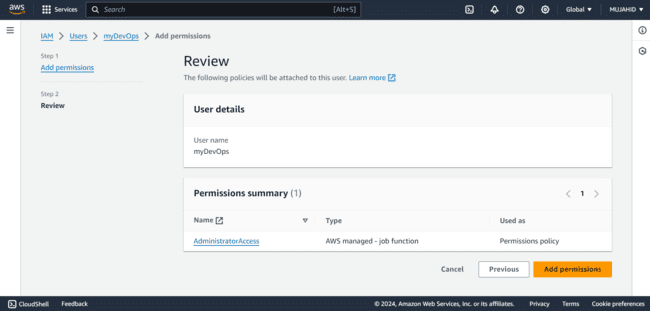
* *So now Log back into your root account and Open the IAM service and click on users tab.*
* *Select the user that we’ve created and click on Add Permissions.*

**

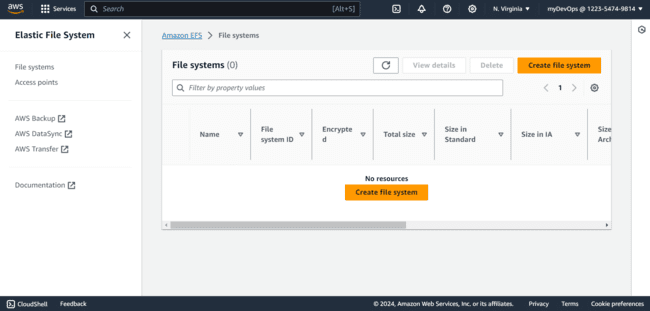
* *Select Attach policies directly option and search for Administrator Access in Permission policies.*

**

* *Put a check mark on it and click on add permissions.*

**

* *Now Log back into the AWS console as IAM user and Explore the Services.*
* *We can Access all other services.*

**

* *So, we created a User with;*

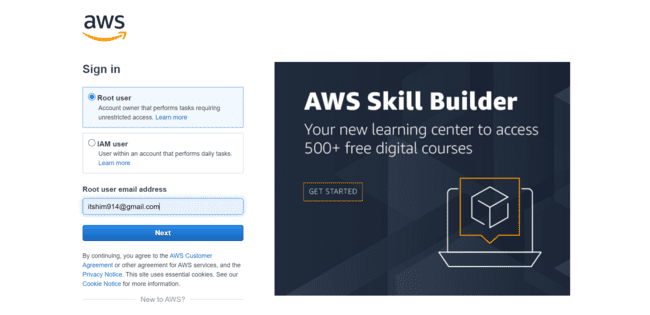
1. *Console access*
2. *Amazon EC2 full access*
3. *Administrator access.*

* *Now the user can Explore all the services on AWS.*

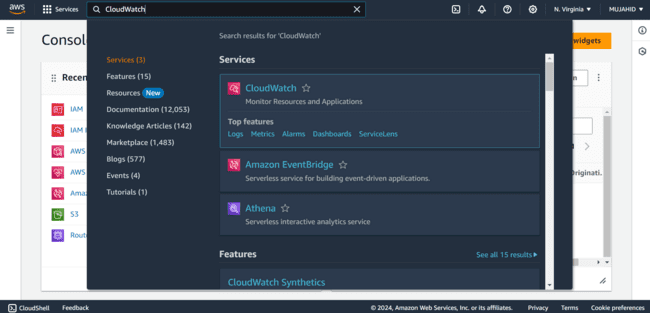
MINI PROJECT – 1

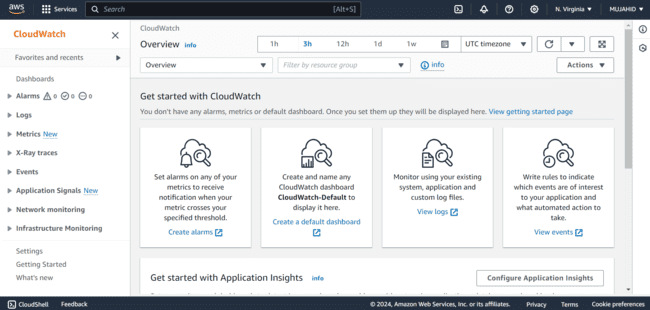
Lab – 2 [BILLING ALARM]

* *Go to the AWS Management Console and Sign in with your Root account.*

**

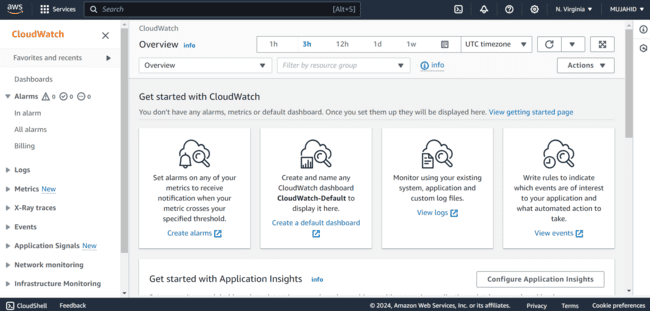
* *After logging open the Services menu and choose CloudWatch.*

**

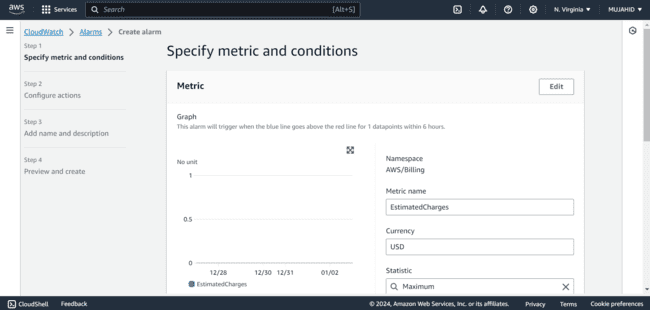
**

***Creating A Billing Alarm:***

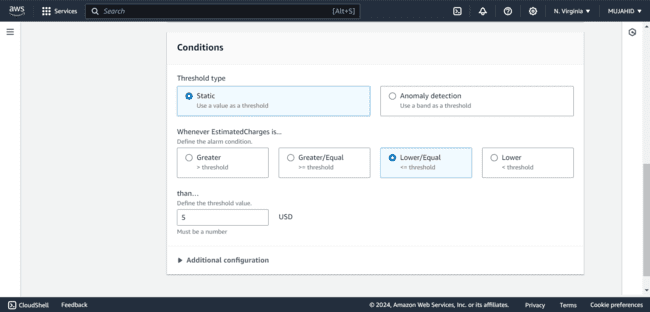
* *Click on Alarms on the left navigation panel.*
* *Select Billing.*

**

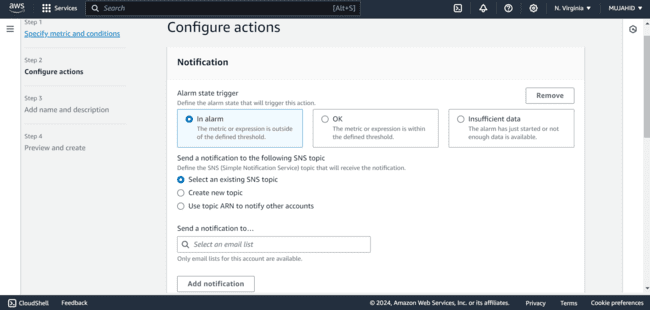
* *Click on create Alarm.*
* *Opens a page to setup your billing alarm where the metric is already selected as Estimated Charges.*

**

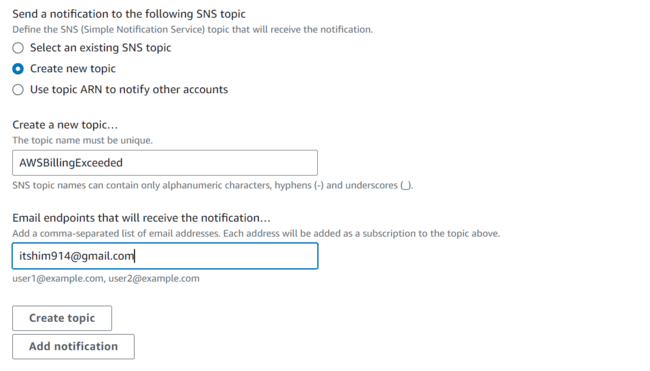
* *Select Static in the threshold type.*
* *Setup the threshold to be notified when the estimated charges exceed a certain amount.*
* *Edit the conditions such as when to be notified like greater than or Lower than the desired threshold.*

**

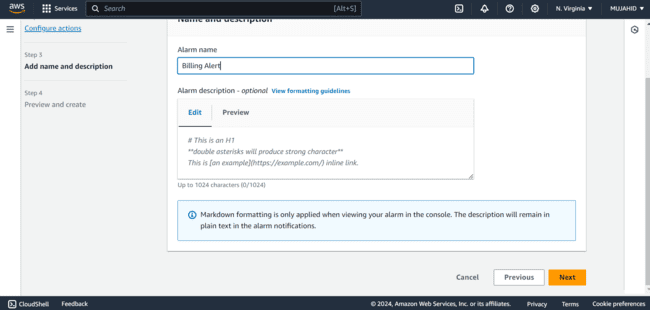
* *I’ve selected the desired threshold value as 5 USD.*
* *And then Click on Next.*
* *In the step 2 configure actions select Alarm state trigger so that it defines the alarm state that will trigger this action.*

**

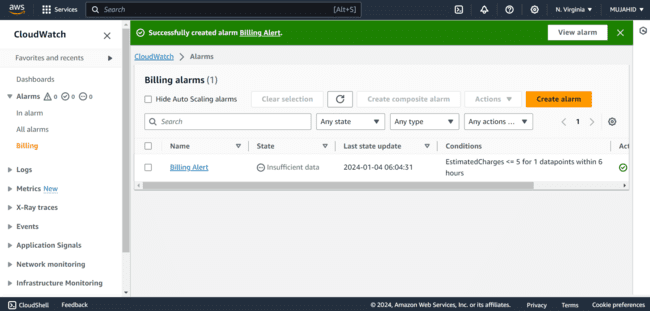
* *To get a notification when alarm is triggered create a new SNS topic.*

**

* *Give a unique name for the topic and email that will receive the notification.*
* *Next click on create topic.*
* *Give an Alarm name and click on next.*

**

* *Preview all the inputs and conditions you gave and click on create alarm.*
* *You receive an email to confirm the subscription that you’ve just created.*
* *Click on confirm subscription and your Alarm is activated.*
* *So, finally an Alarm is created with defined threshold limit and a SNS topic to receive the notification.*

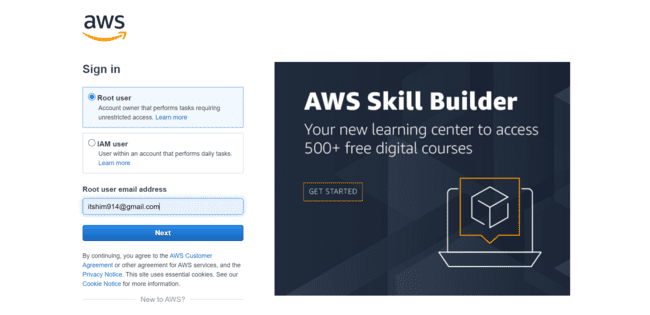
**

MINI PROJECT – 1

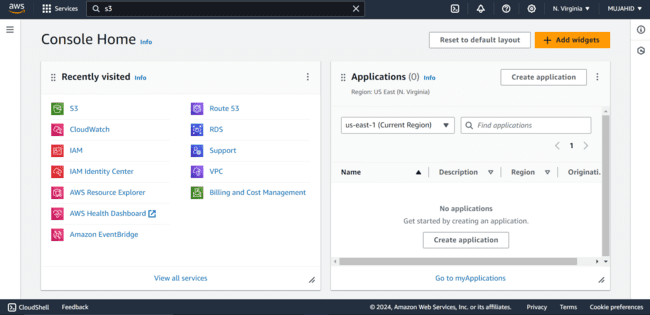
Lab – 3 [S3 BUCKET]

STEP – 1 [CREATING S3 BUCKET]

* *Go to the AWS Management Console and Sign in with your Root account.*

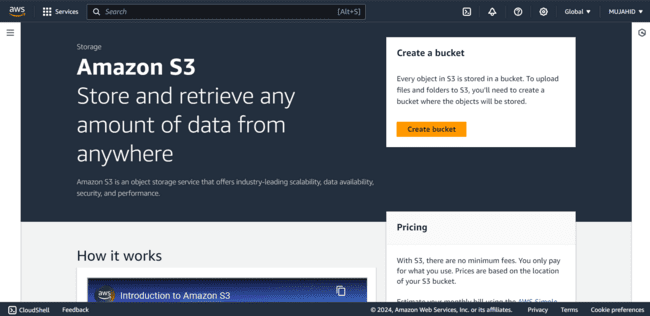
**

* *After logging in the console home choose S3 Bucket.*

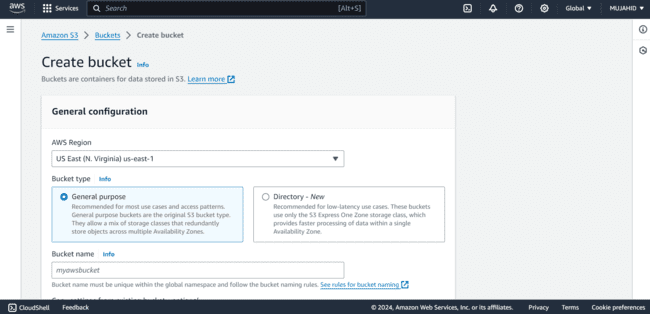
**

***Creating S3 Bucket:***

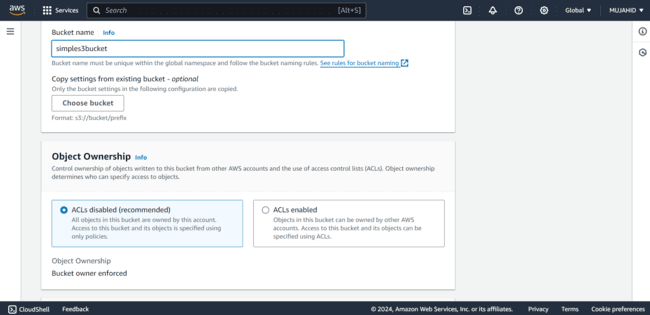
* *Takes you to S3 Bucket home page, Click on Create bucket on the top right.*

**

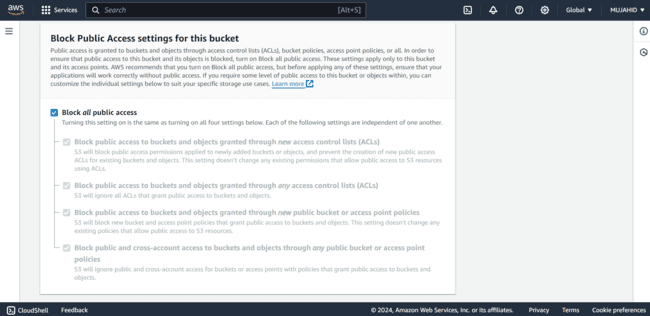
* *Select the region in which the S3 bucket is to be created.*
* *Select bucket type as General Purpose.*

**

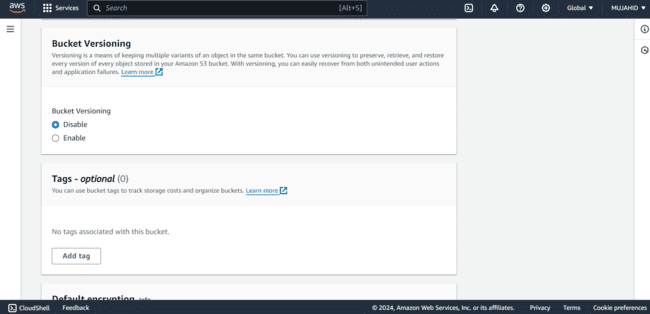
* *Give the bucket a unique name that hasn’t been already taken to your S3 bucket.*
* *In the object ownership choose ACLs disabled, that is only you own the objects in the bucket, which can be modified later.*

**

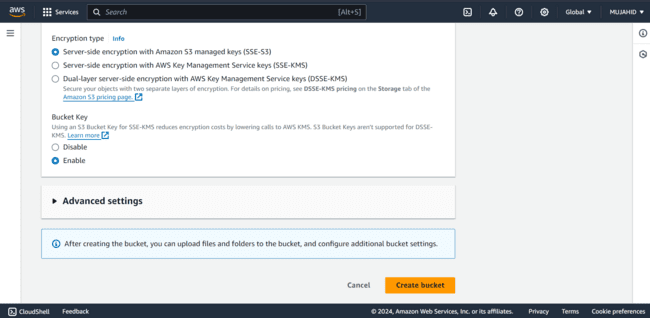
* *Put a check mark on Block all public access so that the public access is denied.*

**

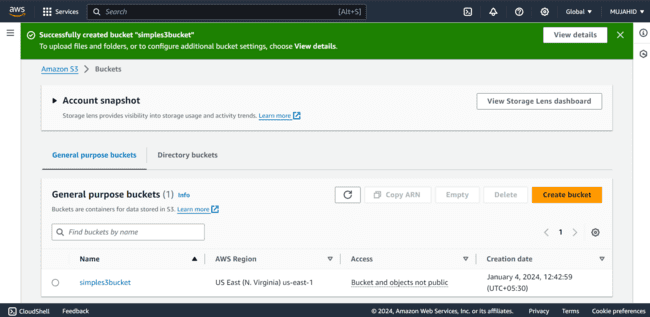
* *Disable bucket versioning, it can be modified later.*
* *Can Add additional tags if you want.*

**

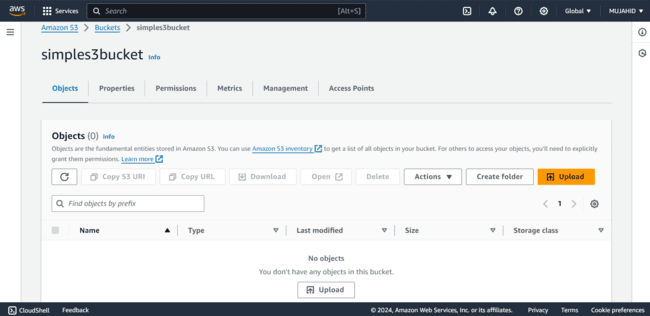
* *Leave everything as it be and click on create bucket.*

**

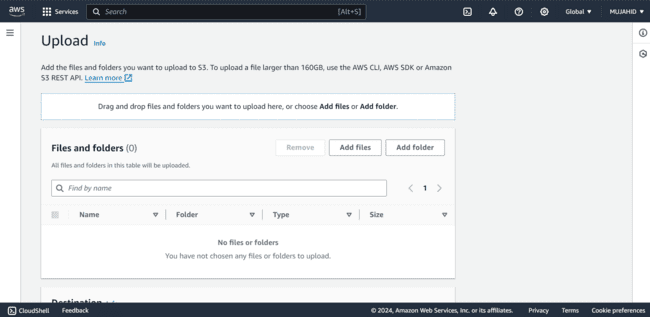
* *Finally, a bucket is created with all the requirements you needed.*

**

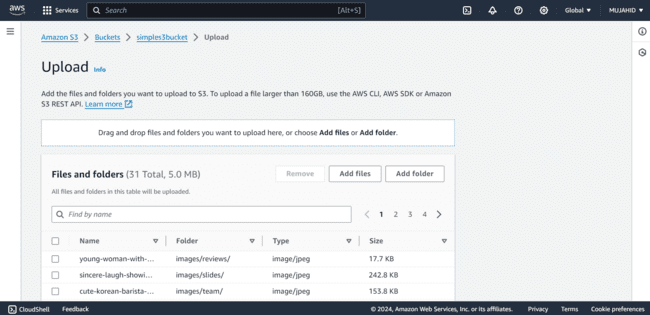
* *Select the bucket that you’ve created to add some files in it.*
* *Click on upload to add files.*

**

* *It opens your PC’s file system to select the content.*
* *Select all the files you want to upload.*

**

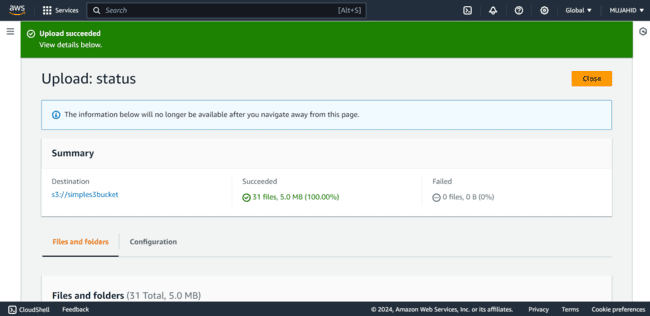
* *Here I selected some files from my PC.*



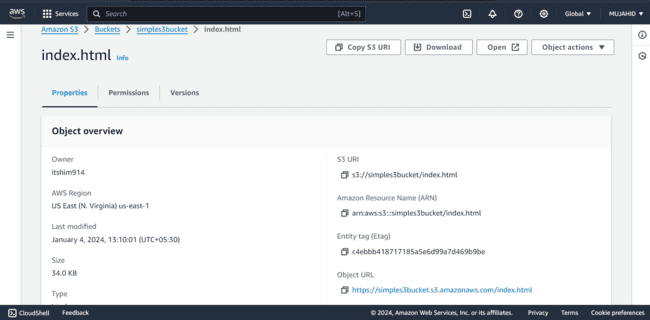
* *Choose the storage class as standard.*



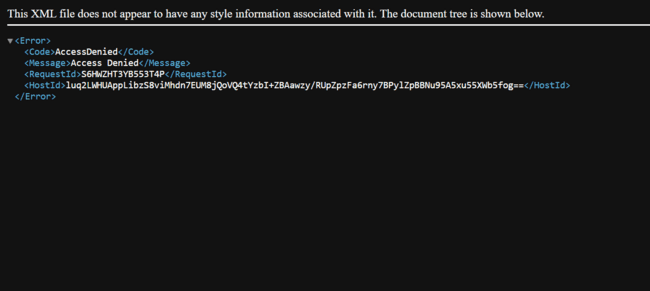
* *Scroll up and click on upload files.*
* *You can see all the files have been uploaded.*



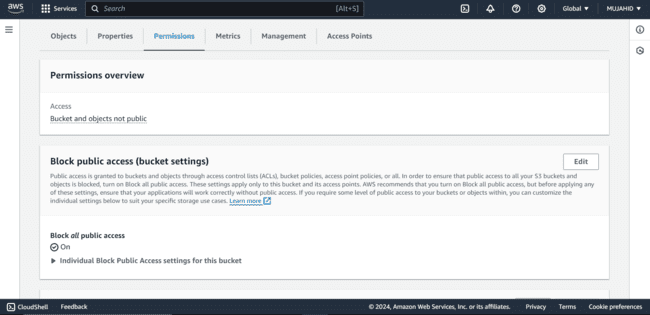
* *I am trying to access the object through its URL in browser.*

**

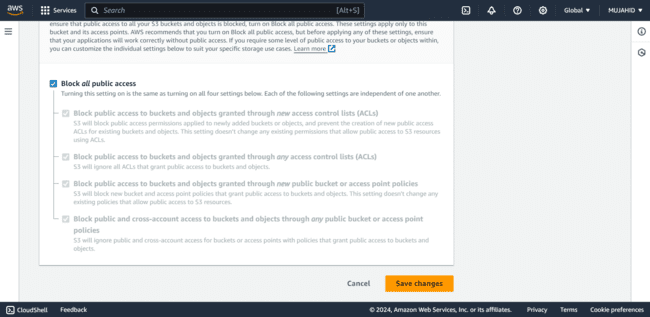
* *It says access denied.*
* *Because public access is denied on this bucket.*

**

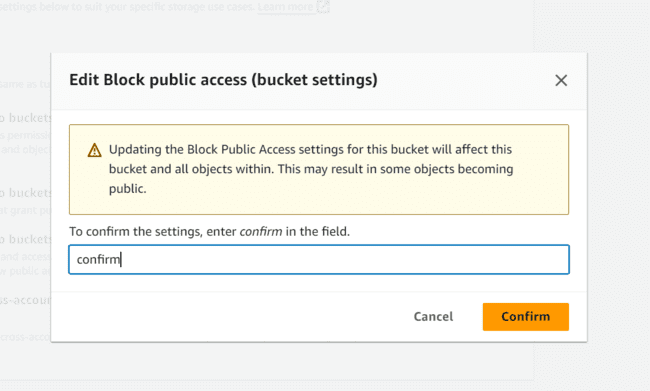
* *To access the files through internet we should make the objects and bucket public.*
* *So, click on the bucket you’ve created and click on permissions tab.*
* *In the block public access section click on edit.*

**

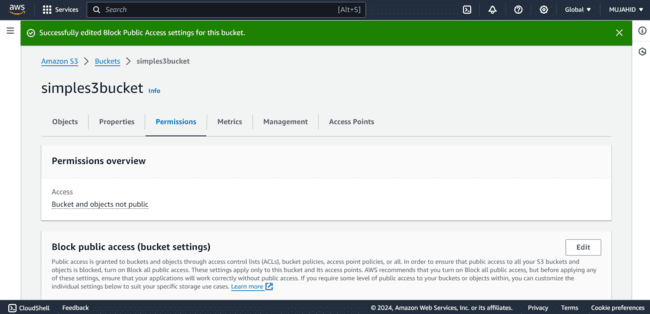
* *Uncheck the option where it says block public access.*
* *Click on save changes.*

**

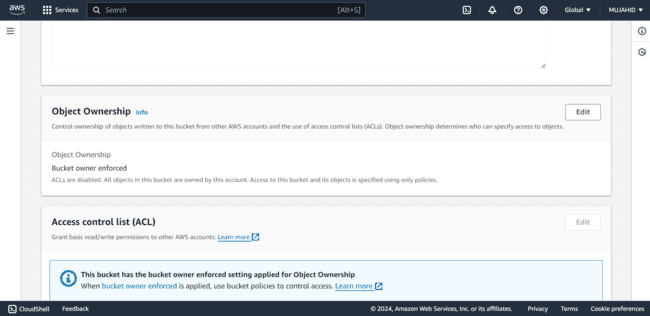
* *To confirm the changes type “confirm” in the given field and click on confirm.*

**

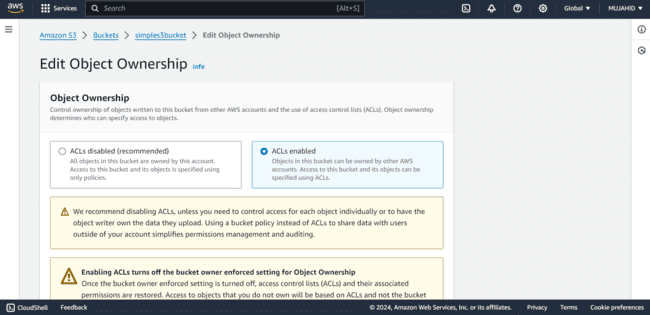
* *Now the changes have been made successfully.*



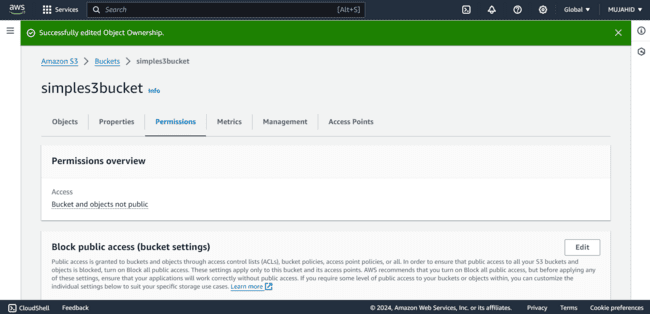
* *Now in the same permissions tab click on edit object ownership tab.*



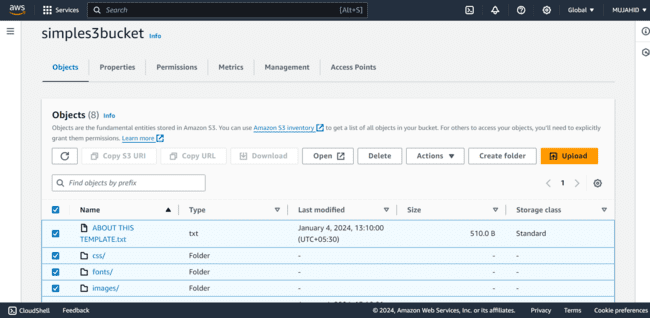
* *Select the option ACL’s enabled and click on comfirm changes.*



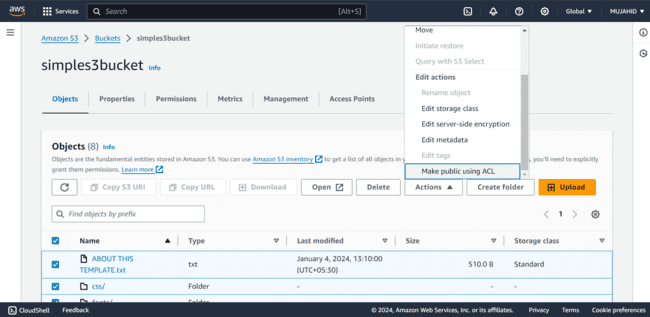
* *You can see the changes have been successfully made.*



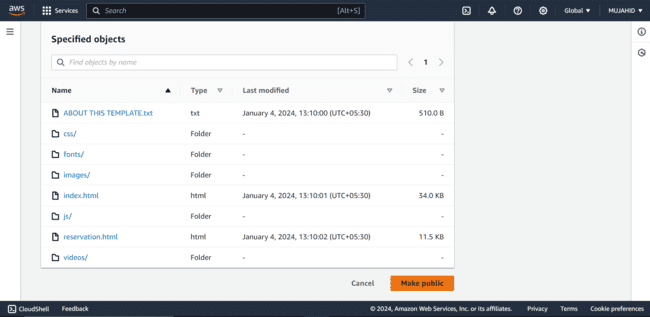
* *Now select all the files in the bucket and click on actions.*



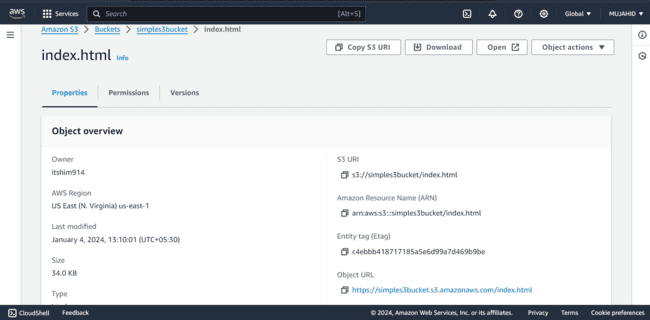
* *Select the option “Make public using ACL”.*

**

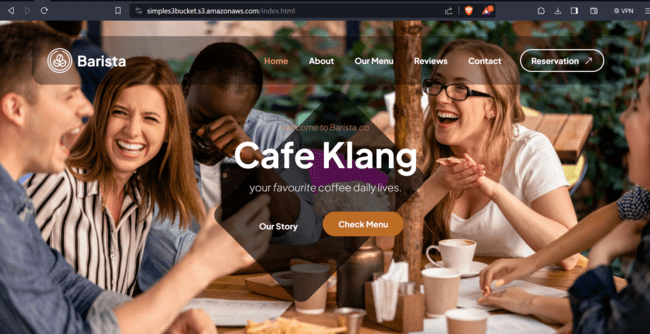
* *Shows the specified objects that are being public.*
* *Click on make public to make the objects public.*

**

* *The objects in the bucket are successfully made public.*
* *Now it can be accessible through internet through objects URL.*

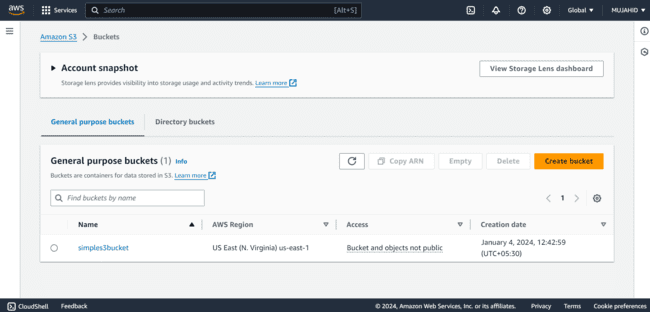
**

* *Click on the URL and you can see the object displaying its content without any error.*

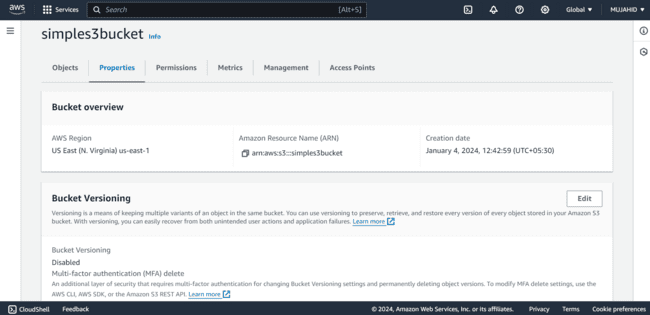


STEP – 2 [ENABLE VERSIONING]

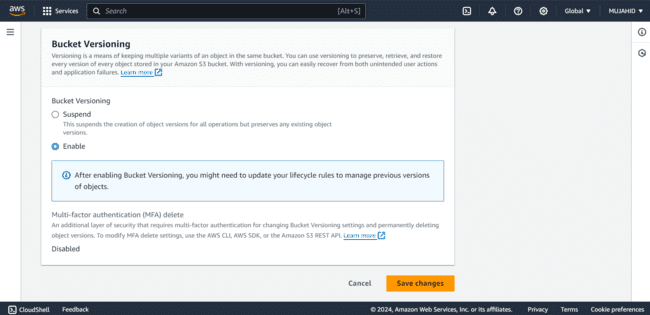
* *Click on the bucket you’ve created.*

**

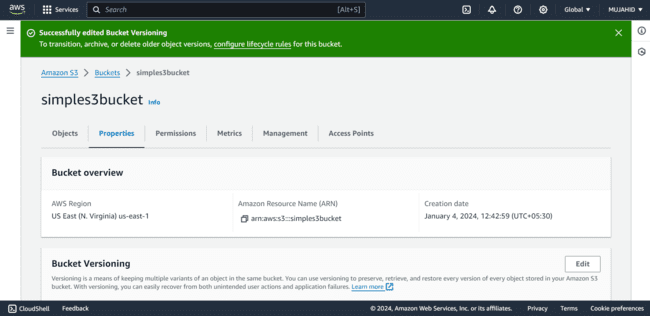
* *Click on the properties tab.*
* *Click on Edit button in Bucket Versioning Section.*

**

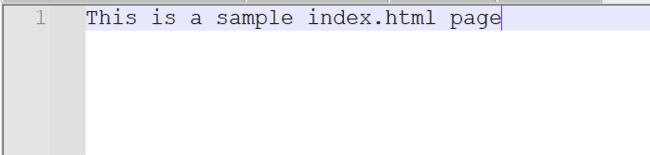
* *Select “Enable” to enable the bucket versioning.*
* *Click on save changes.*



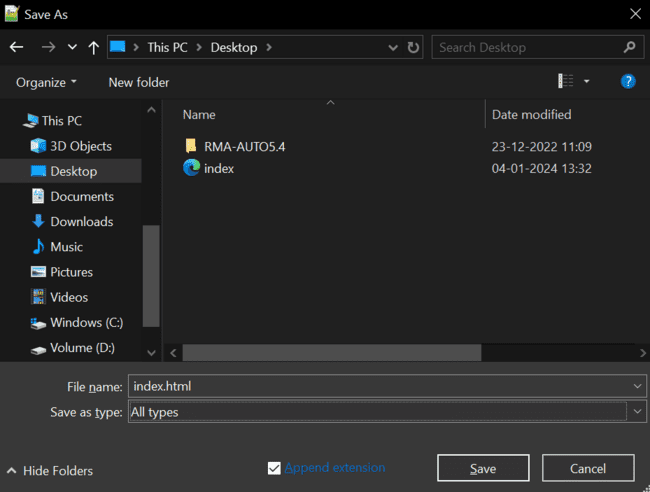
* *The changes have been made successfully.*
* *Now the bucket has bucket versioning enabled.*



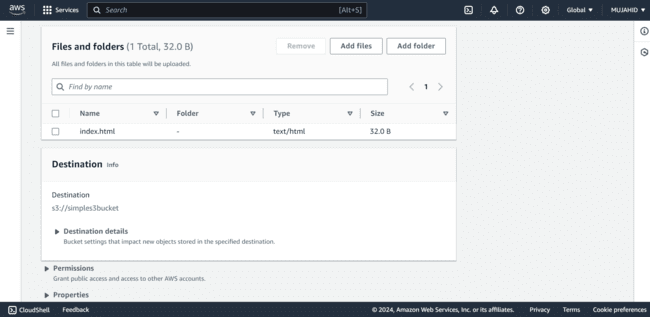
* *I created a sample text file in my text editor*

**

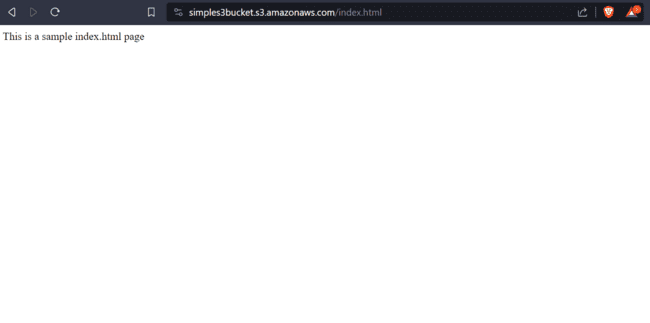
* *Saved it in my local PC with the same name as one of the files in my S3 bucket.*

**

* *Now I uploaded the text file that I’ve created.*

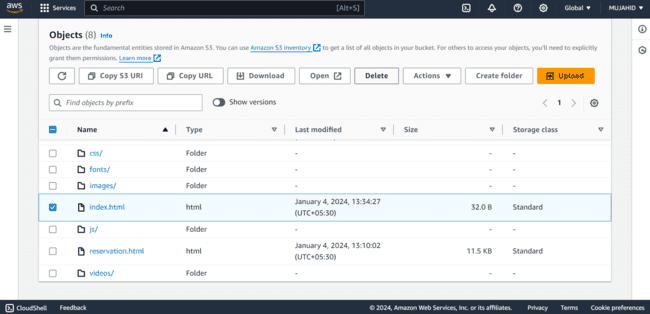
**

* *I tried to access it through the objects URL through browser and this is the content that it displayed.*

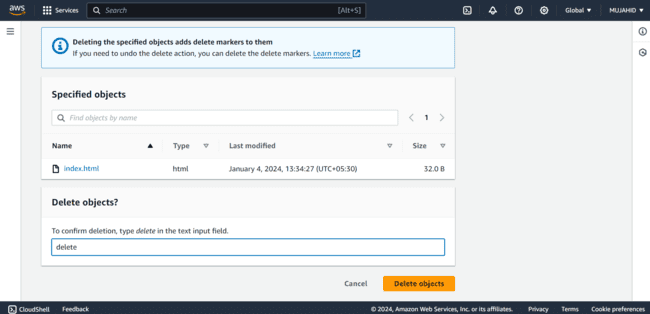


STEP – 3 [RECOVERING THE DELETED FILE]

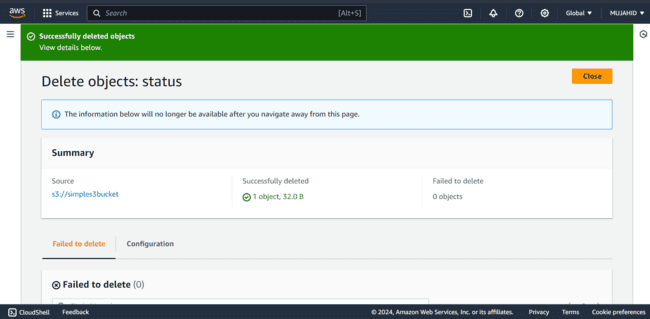
* *I tried to delete the file that I’ve just uploaded.*



* *Type delete to confirm deletion and click on delete objects.*



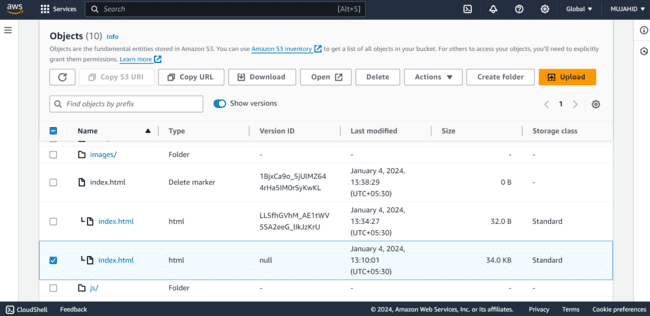
* *Now that the file is successfully deleted, we have our bucket enabled versioning so we can recover it back.*



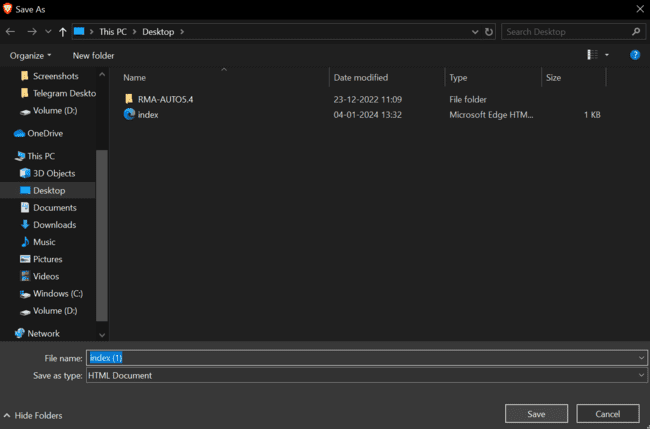
* *In the bucket you can see a option “show versions”.*
* *Enable it to see all the versions of the files of your bucket.*



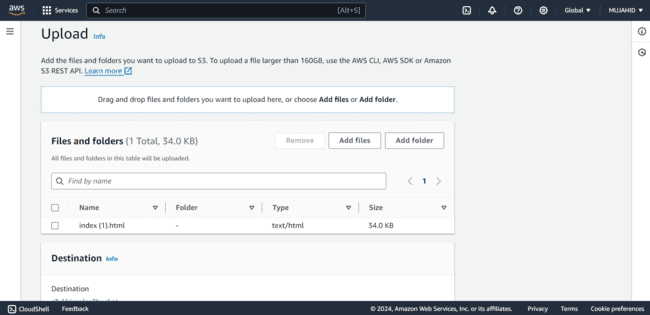
* *We can see the version of the file that we’ve uploaded, updated and deleted.*
* *Click on the version of the file that you want to recover.*



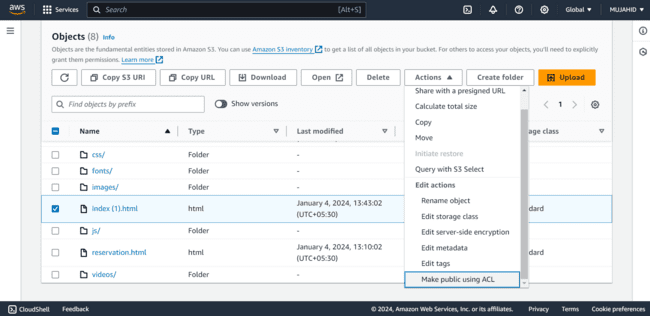
* *Save it in your Local PC.*



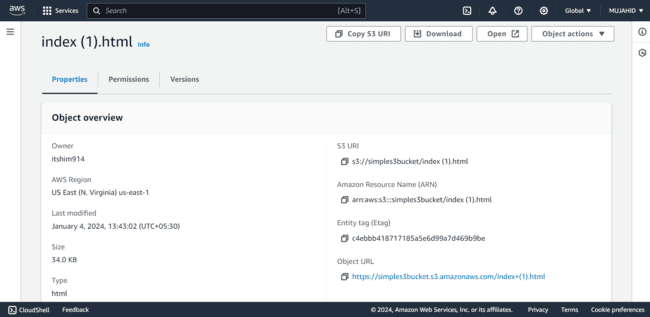
* *Uplaod the file that you’ve saved in your PC in to your bucket.*



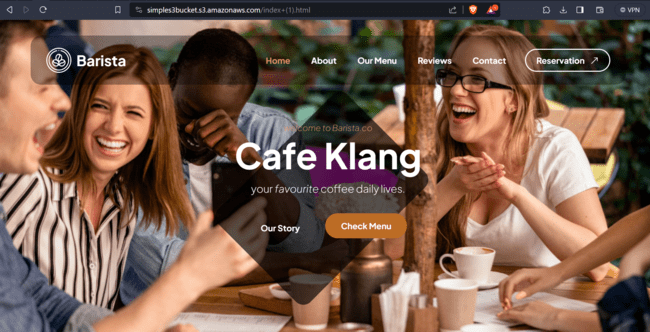
* *Make the file public in the actions section to access it through browser.*



* *Now try to access the file with objects URL through browser.*



* *The content of the object works just fine.*

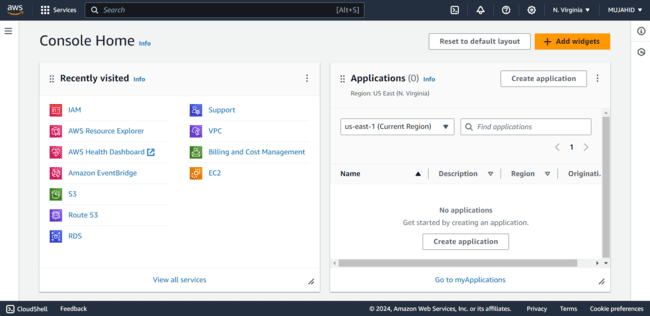


MINI PROJECT – 1

Lab – 4 [EC2-INSTANCE]

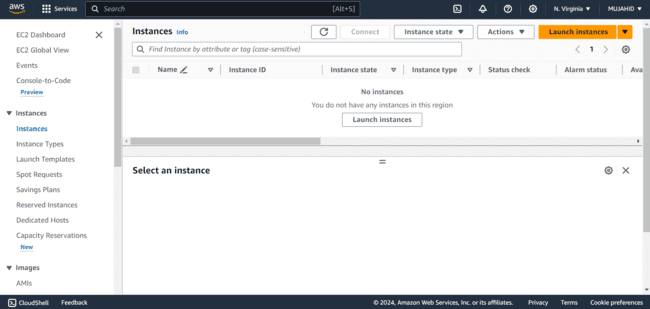
* *Log into AWS management console.*



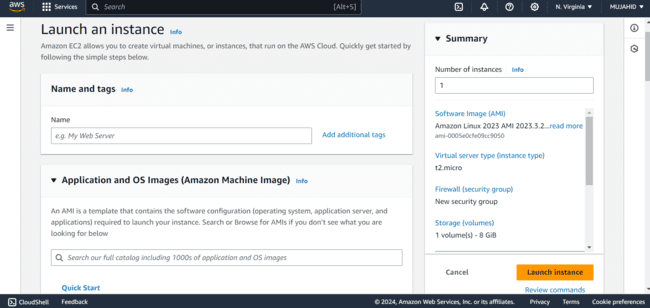
* *I logged in with my AWS credentials and entered into my AWS management console.*

***Creating An EC2 Instance:***

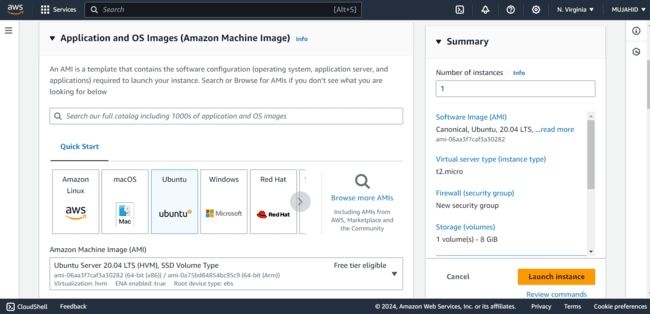
* *Click on EC2 to create instance.*
* *Click on Launch instances.*

**

* *It Opens a configuration page to create an instance.*
* *Firstly, give a name or tag to the instance.*
* *Say “exercise-ec2”.*

**

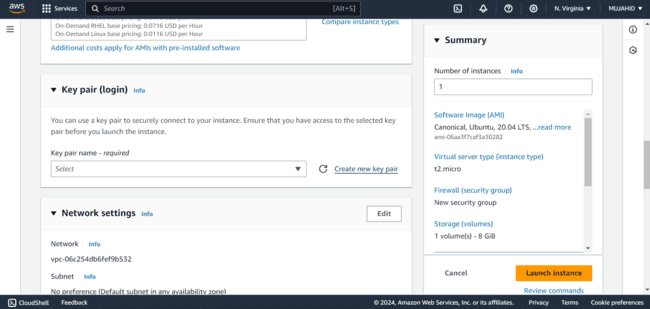
* *Next is AMI [Amazon Machine Image].*
* *Choose an OS [Operating System] to run the instance.*
* *Here I selected ubuntu20.04 operating system.*

**

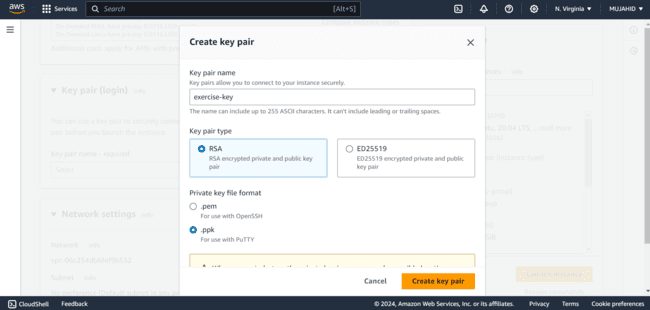
* *Choose instance type as t2micro which gives 1CPU and 1GB of RAM to the instance.*

**

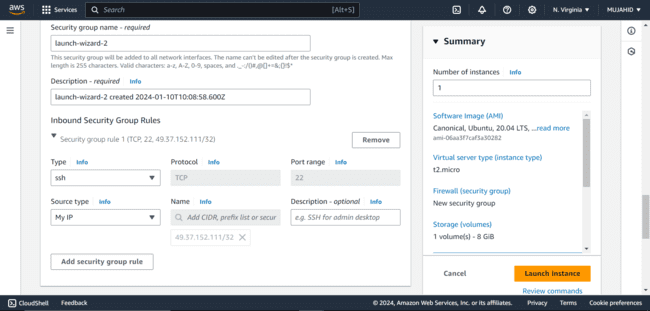
* *In the key pair section click on create new pair.*

**

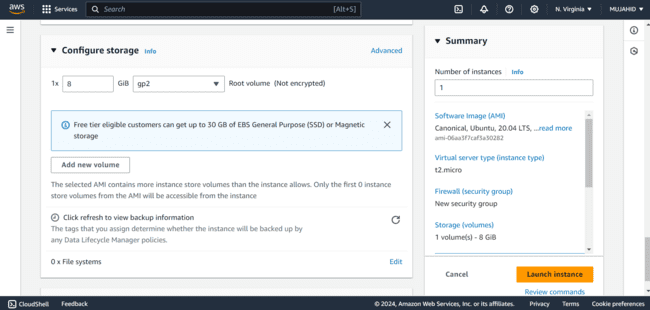
* *Here, Give a name to your key pair and keep the key pair type as RSA.*
* *Select key file format as .ppk and click on create key pair.*
* *Choose a download path for your key pair in your local PC.*

**

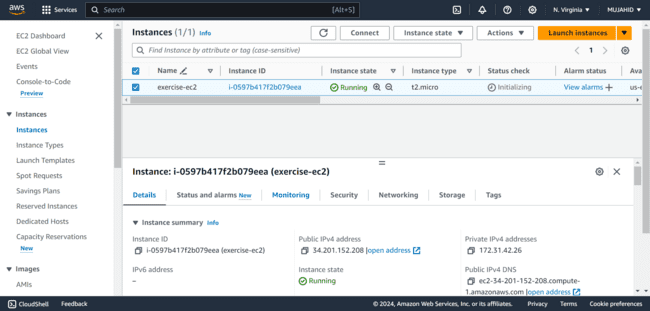
* *In the Network Settings section, give an inbound rule SSH and its port range is 22 allowed from only My IP.*
* *So that only you can login to your instance.*

**

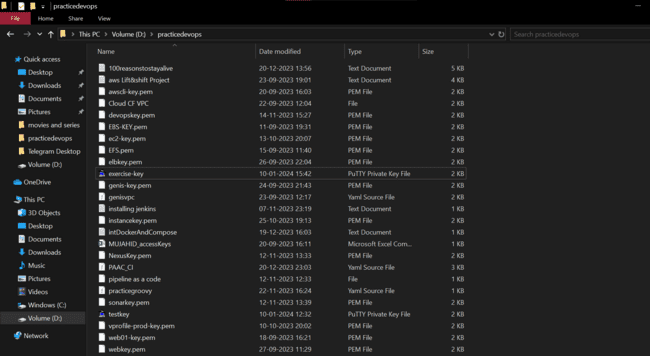
* *Check all the configurations that you’ve choose and click on Launch Instance at the bottom right corner of the page to create an instance.*

**

* *Now instance is successfully Launched and in Running state.*

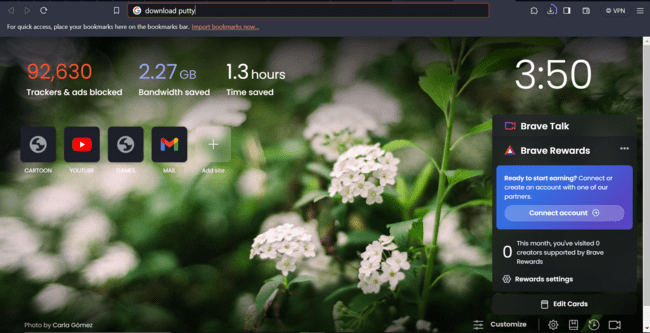
**

* *The .ppk file is saved in my local PC with the name “exercise-key”.*

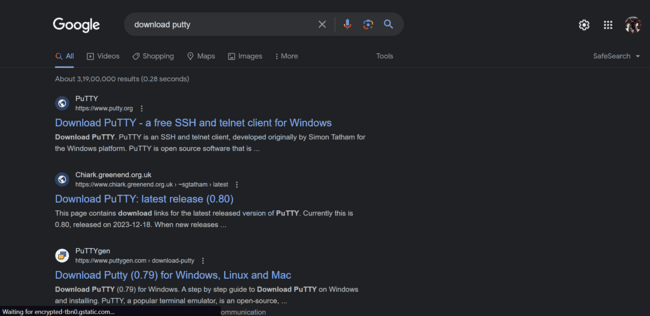
**

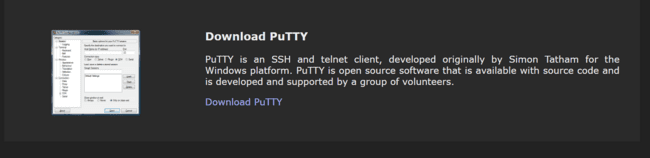
***Downloading Putty Software:***

* *To Access the instance with the putty file we have to download the software putty.*
* *Open your desired browser and search “download putty”.*

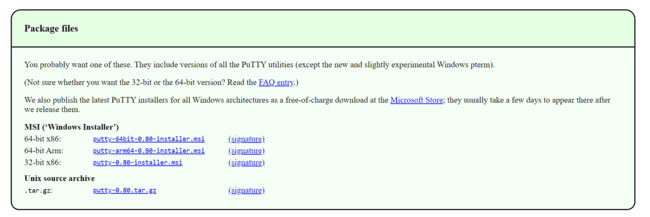
**

* *Click on the first official link to download putty.*
* *Click on download putty in the next page.*

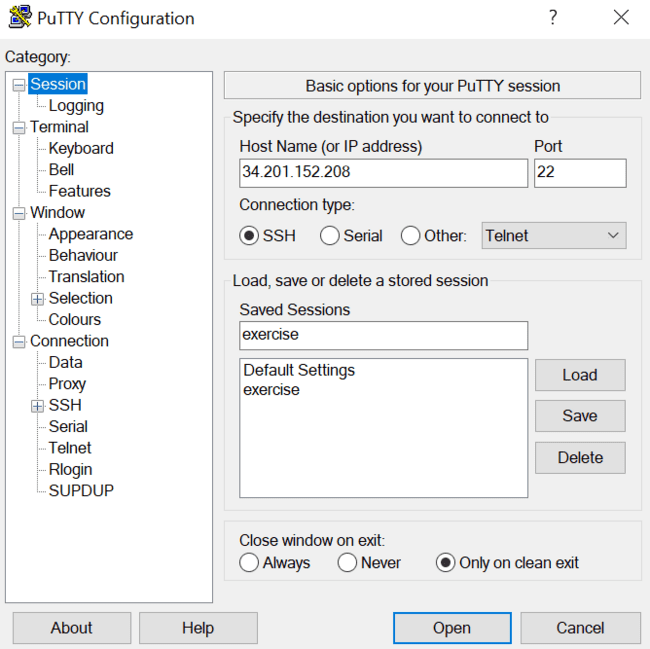
**

**

* *Select the type of installer as per your specifications from the given types.*

**

* *After downloading, go to the download path and install the putty software.*
* *Continue with all the configurations and finish the installation.*
* *After installing open the putty software to Access your EC2 instance.*
* *In the "Session" category Enter the public DNS or IP address of your EC2 instance.*
* *Set the connection type to "SSH."*
* *Enter a name for your session in the "Saved Sessions" field.*
* *Click "Save" to save the session.*

**

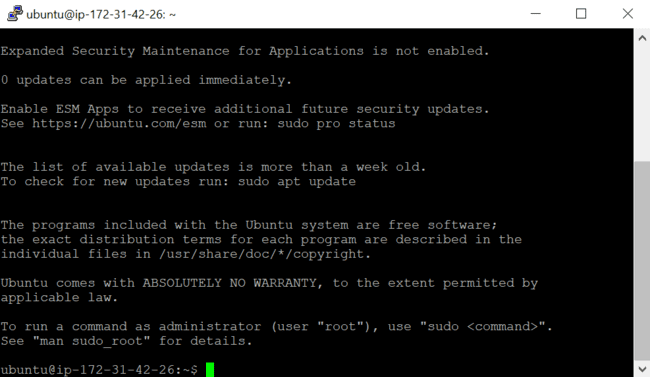
* *In the "Connection" category Expand "SSH" and select "Auth" and select “Credentials”.*
* *Browse and select the PPK file you generated while creating instance and click "Open" to start the SSH session.*

**

* *Opens a shell and asks “login as:”.*
* *Type the username according to the type of OS that you’ve selected.*

**

* *Successfully logged in to your instance and can access it.*

**

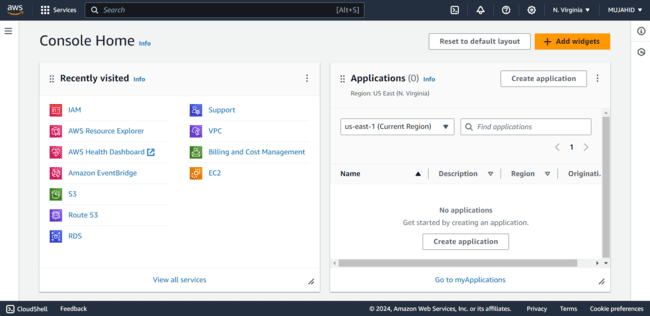
MINI PROJECT – 1

Lab – 5 [SECURITY GROUP]

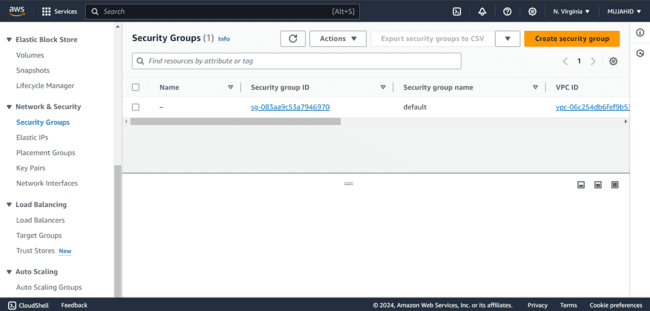
* *Log into AWS management console.*



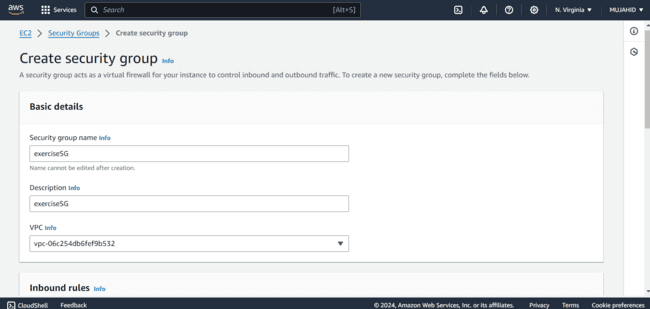
* *I logged in with my AWS credentials and entered into my AWS management console.*
* *Click on EC2 and in the left navigation panel under “Network and Security” click on Security group.*



* *Click on create Security Group to create a new SG.*

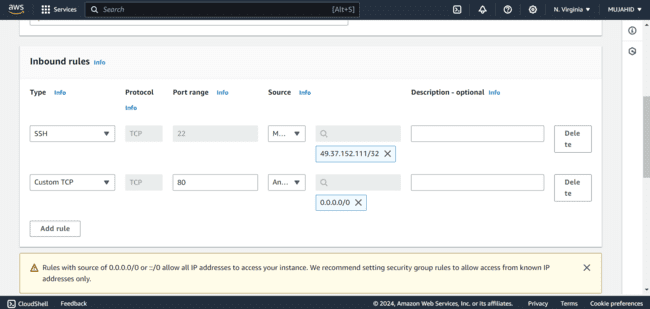
**

* *Give a name to your Security Group and also Description.*
* *I gave the name as “exerciseSG” and same to the description also.*
* *Default VPC is selected.*

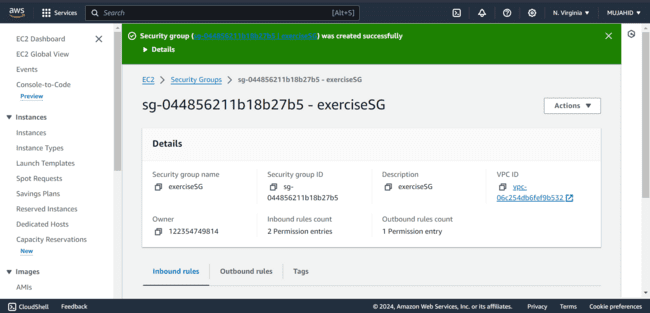
**

* *In the inbound rules section, I gave 2 rules that are;*

1. *SSH Allowed from My IP*
2. *Port 80 Allowed from Anywhere.*

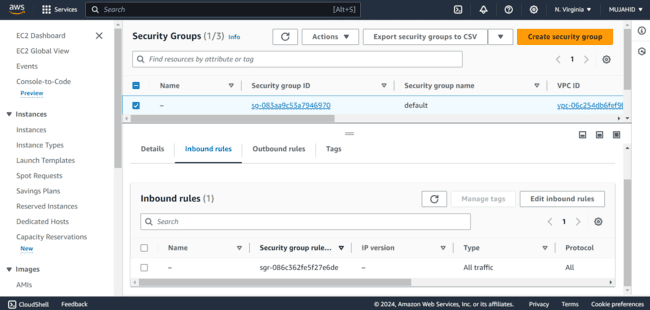
**

* *After adding the rules scroll down and click on create Security Group.*
* *Security Group is successfully created.*

**

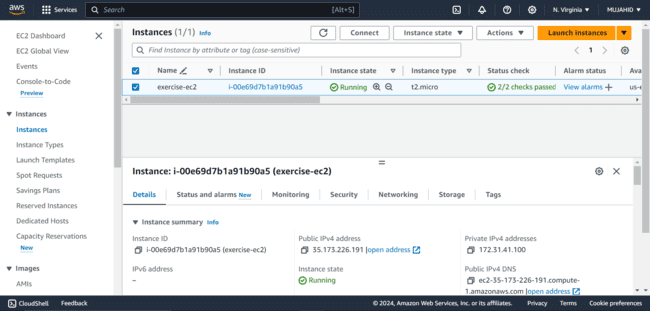
***Rules of Default Security Group:***

* *These are the rules of default security Group. That is “All traffic allowed from Anywhere”.*

**

***Changing Security Group of Existing Instance:***

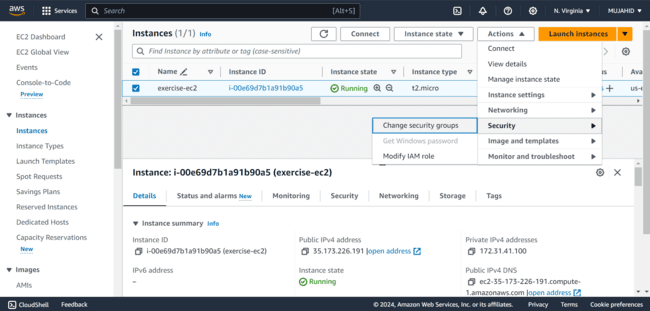
* *Now Attaching this to an existing EC2 Instance, click on the instance that you want to attach the security group to.*

**

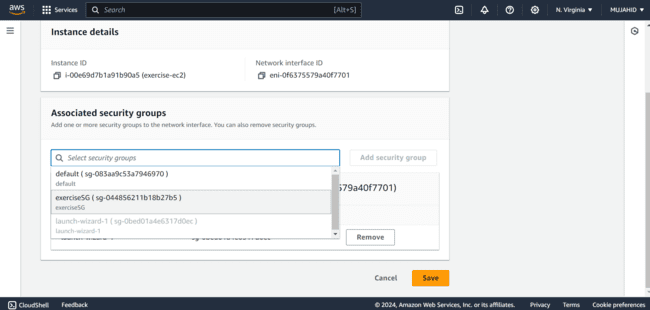
* *Click on “Actions” and then in the dropdown menu select “Security”.*

**

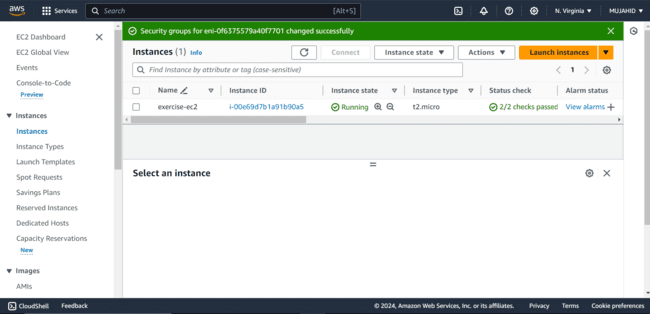
* *Click on Change Security Groups.*

**

* *In the Associated Security Groups Select the SG that you’ve Created with your required Security Inbound Rules*

**

* *Now the SG is successfully attached to the existing Instance.*

**

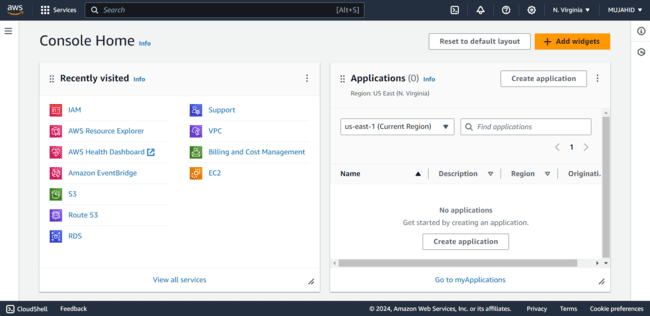
MINI PROJECT – 1

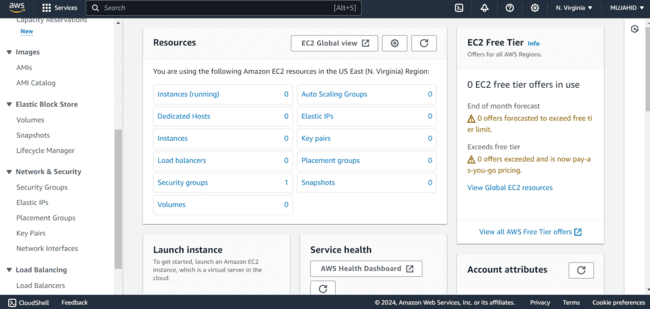
Lab – 6 [VOLUMES AND SNAPSHOTS]

* *Log into AWS management console.*



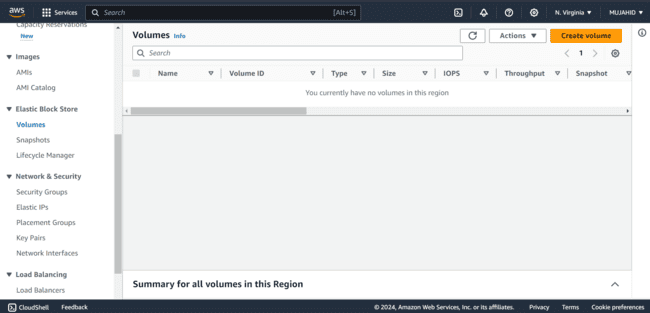
* *I logged in with my AWS credentials and entered into my AWS management console.*
* *Click on EC2 and in the left navigation panel under “Elastic Block Store” click on “Volumes”.*



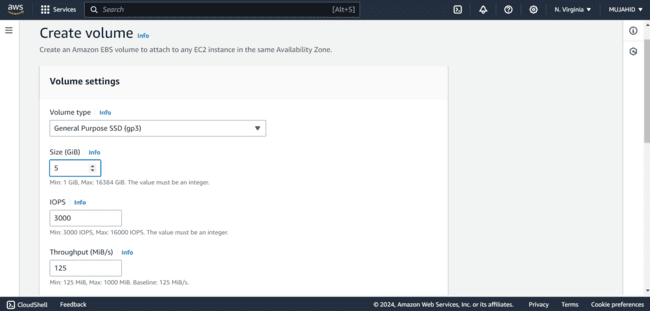
**

***Creating A Volume:***

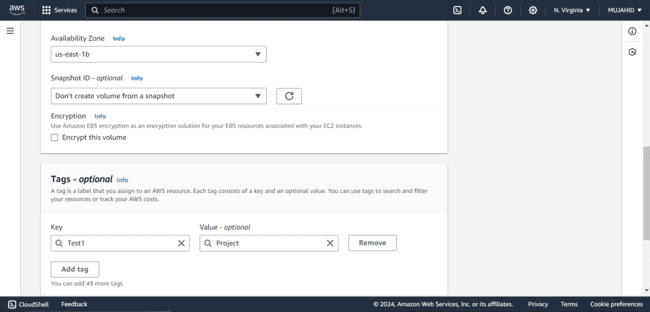
* *Click on Create Volume on the top right Corner of the page.*

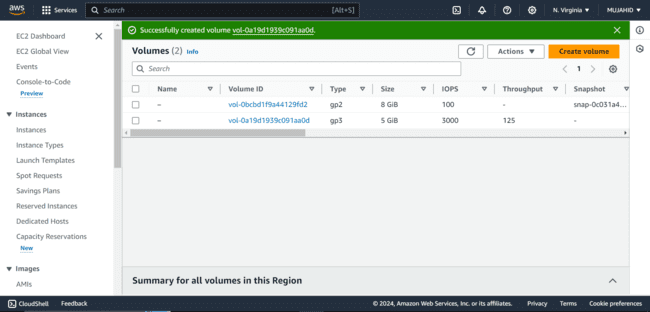
**

* *Opens a configuration page to create a volume.*
* *Keep the volume type as General purpose SSD [gp3].*
* *Set the size of the Volume to 5GB.*
* *Leave everything to default.*

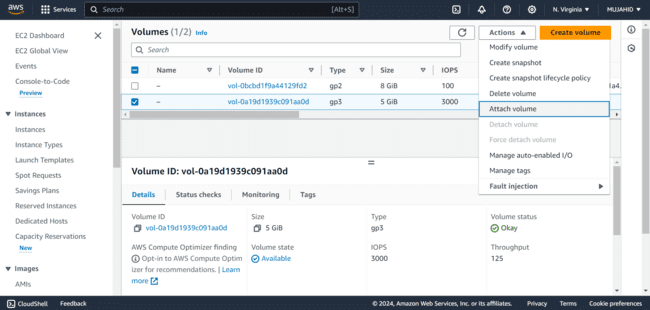
**

* *Check the availability zone, keep it in the same availability zone as in which the EC2 instance is.*
* *Give tags if needed.*
* *Scroll down and click on create volume.*

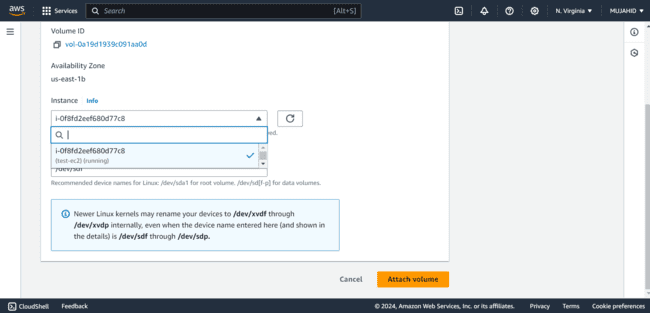
**

**

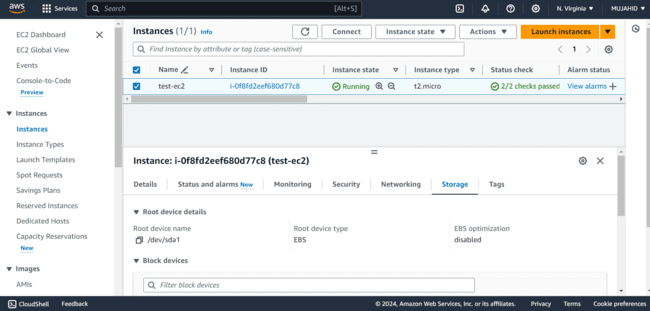
* *To attach it to the EC2 instance, select the volume which you want to attach and click on Actions and then Attach Volume.*

**

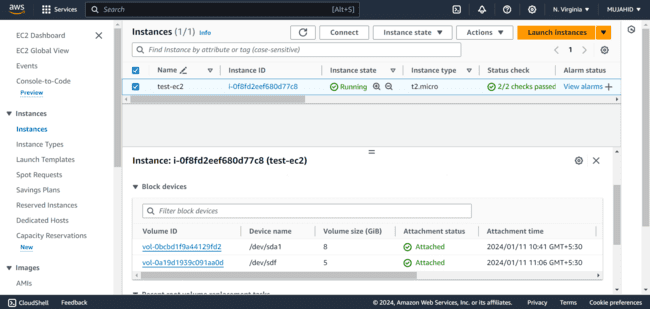
* *Select the Instance to which you want to attach the Volume.*
* *I have attached the volume to my Existing EC2 Instance [test-ec2].*
* *After selecting, click on Attach Volume.*

**

* *To check if the instance is attached or not, Select the instance and click on storage tab.*

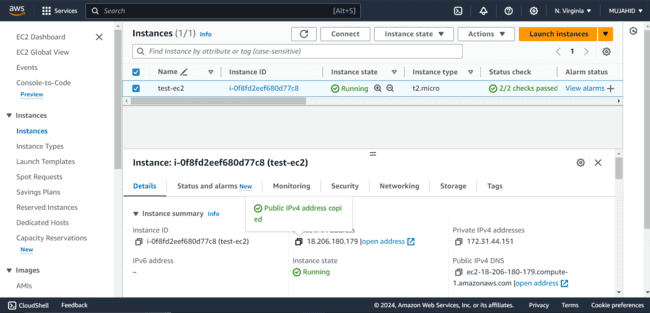
**

* *Here is the instance that we’ve attached to the Instance.*

**

***To Access this Volume in Instance:***

* *To access the volume that we’ve attached we have to SSH to our Instance, to do this copy your Public IP in AWS.*

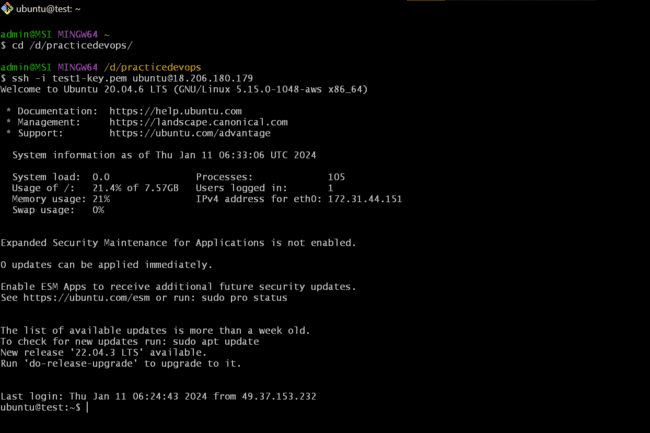


* *Use this command to Login to your Instance.*

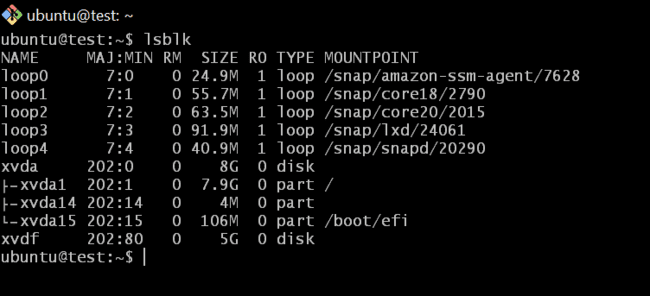
*“ssh -i test1-key.pem* [*ubuntu@18.206.180.179*](mailto:ubuntu@18.206.180.179)*”*

**

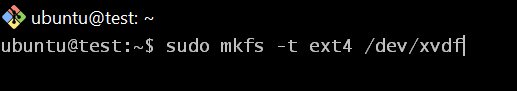
* *Type “Yes” if asked any permissions to login.*
* *Now we are successfully Logged in to Instance.*

**

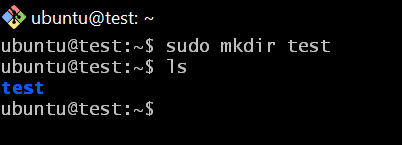
* *Use command “lsblk” to list all the block volumes.*

**

* *To mount this block volume to our EC2 instance we have to create a file system with our block volume.*
* *With the command “sudo mkfs -t ext4 /dev/xvdf”*

**

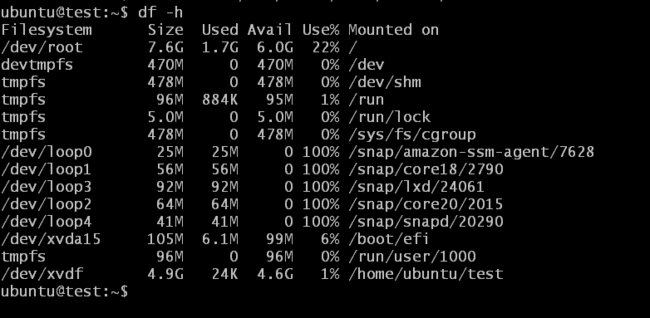
* *Create a directory with command “sudo mkdir test” to mount the volume to a directory.*

**

* *Now to mount our block volume /dev/xvdf to our directory test use command “sudo mount /dev/xvdf /home/ubuntu/test”.*

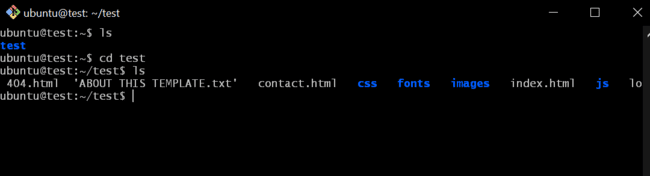
**

* *Now that the volume is successfully mounted to the directory, we can check this by command “df -h”.*

**

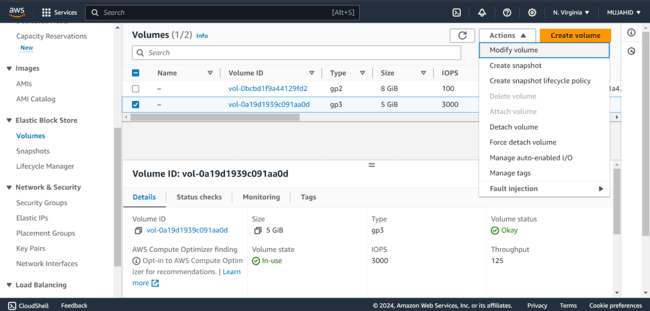
***Putting Some Content In It:***

* *I have put some content in it that are files of a web template.*

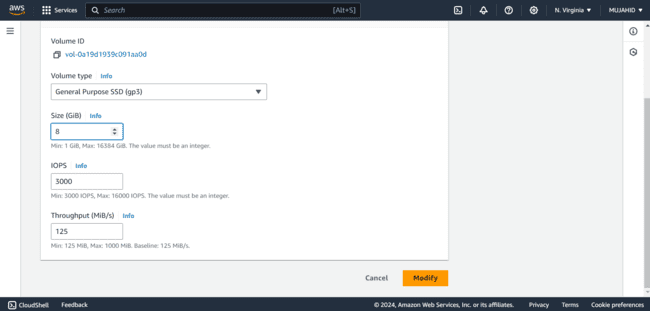
**

***Increasing the Volume to 8GB:***

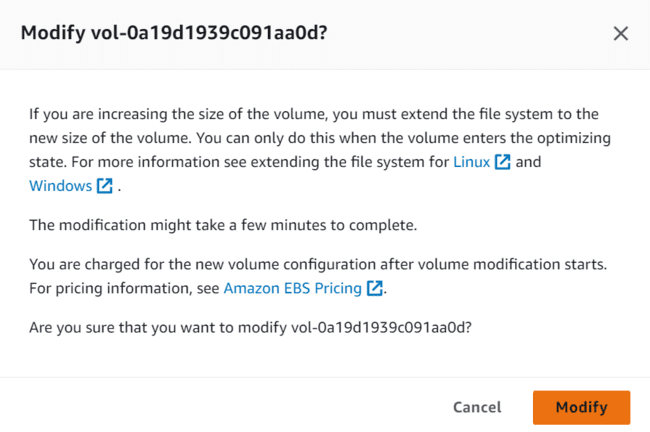
* *To increase the size of the volume from 5GB to 8GB, get into the volume sections of our AWS console and select the volume.*
* *Click on Actions and click on Modify Volume.*

**

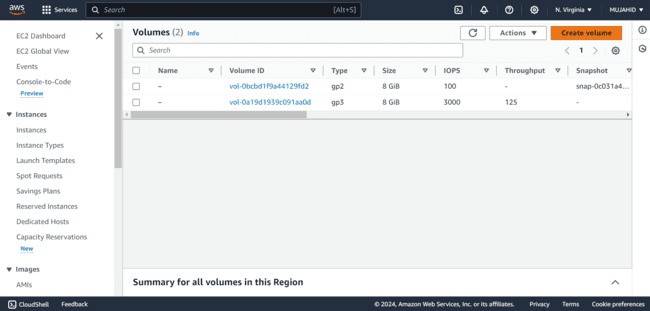
* *Keep everything as it is and increase the size from 5GB to 8GB.*
* *Click on Modify after changing.*

**

* *If the size of the volume is increased it also asks to increase the size of file system to new size of the volume.*

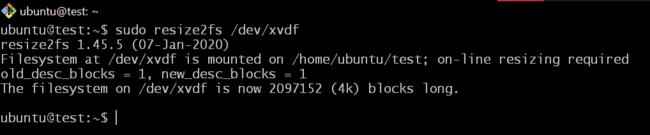


* *Now that the size of the volume is increased its now time to increase the size of file system.*

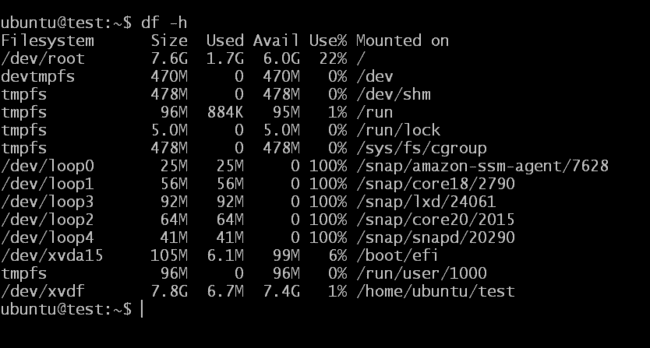


* *Login to your instance and use command*

*“sudo resize2fs /dev/xvdf” to resize the file system.*

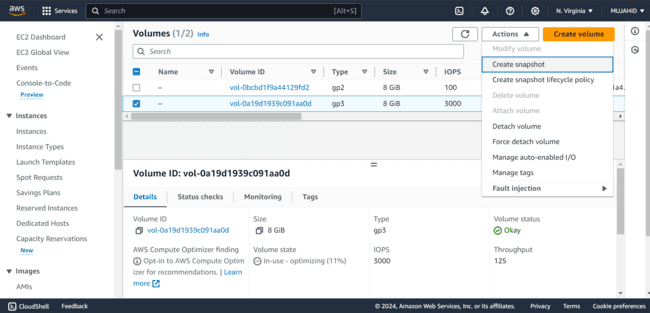


* *Use the command “df -h” to check the increase of the block volume in instance.*
* *Where df -h is used to check disk space of the Instance.*

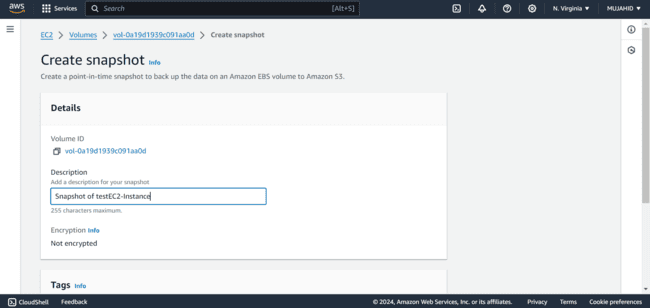


***Taking Snapshot of this Volume:***

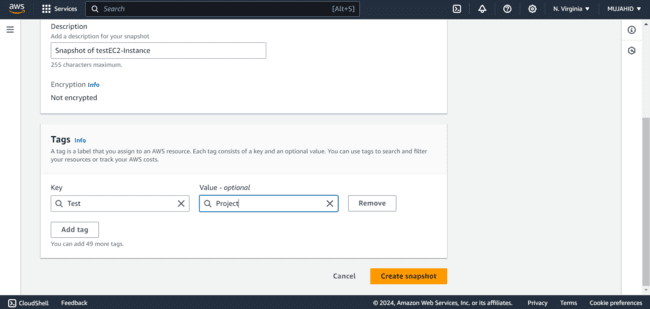
* *Get into the volume sections in AWS and select the volume to which you want to take the snapshot.*
* *Click on Actions and then click on Create snapshot.*

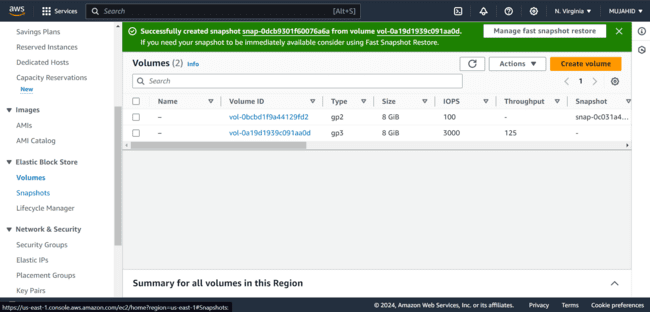
**

* *Give a description and and tag if needed to the snapshot.*

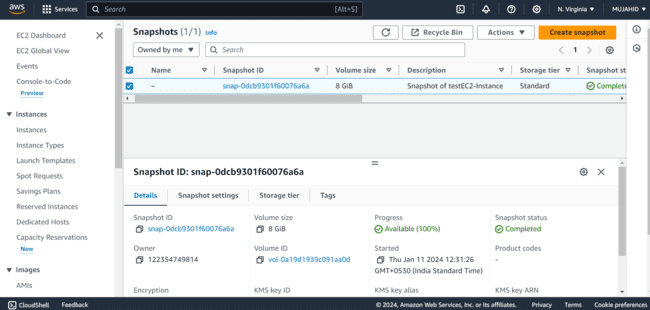
**

* *Scroll down and click on create snapshot.*

**

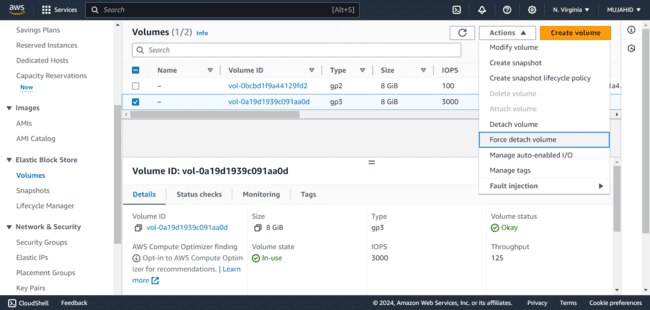
**

* *Now that the snapshot is created from the volume, the volume can be deleted.*

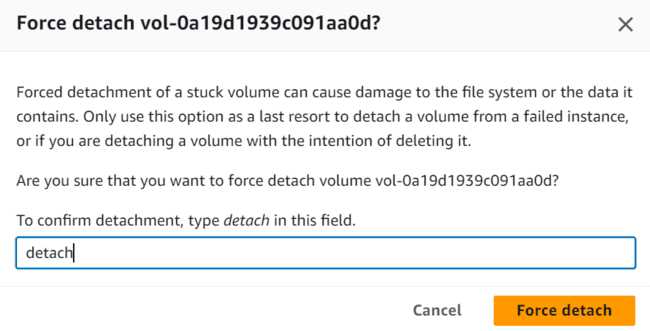
**

***Deleting Volume:***

* *Select the volume which you want to delete.*
* *To delete the volume first we have to detach the volume from the instance that it is attached.*
* *Select the volume and click on Actions and then click on Force detach volume to detach volume from instance quickly.*

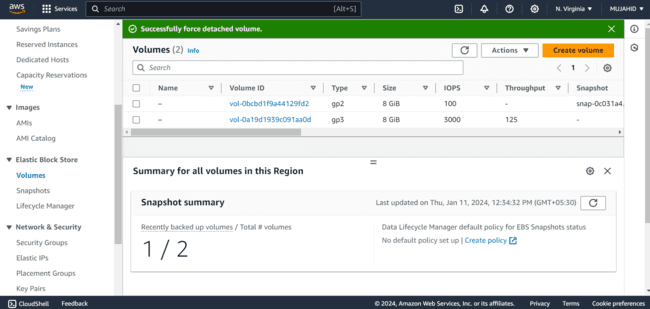
****

* *Asks for a confirmation to detach the volume, type detach and click on force detach.*

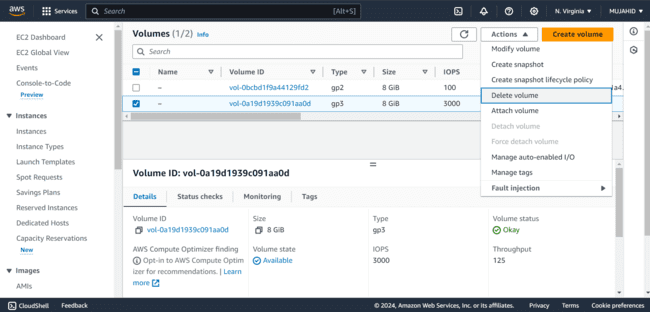
****

* *Now that the volume is detached successfully, it can be deleted.*

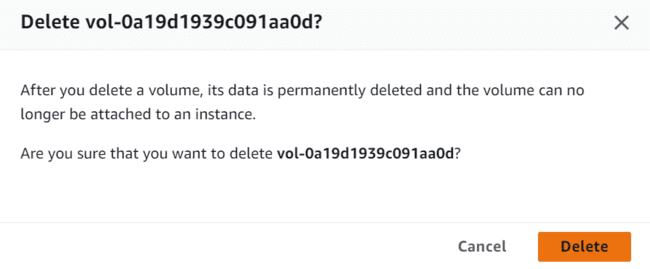
*.*

****

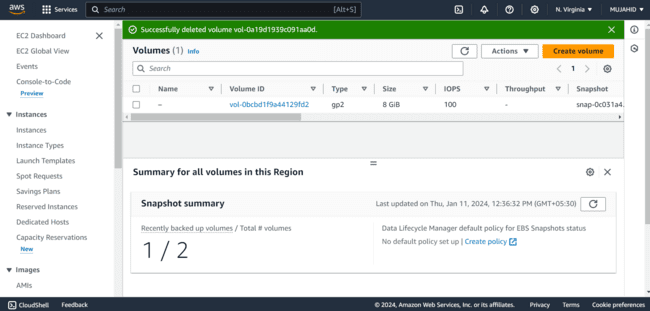
* *Select the volume and click on Actions and then click on Delete Volume*

****

* *Asks for confirmation to delete the volume click on Delete to delete the volume.*

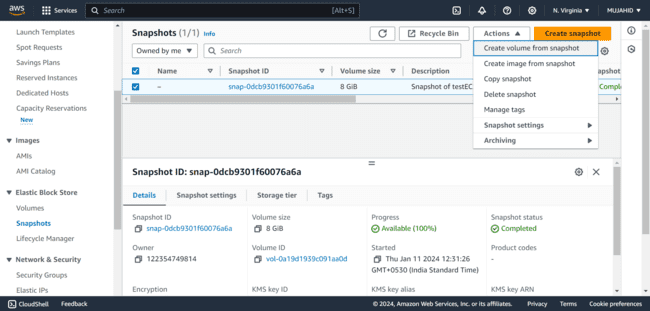


* *Now the Volume is successfully deleted.*

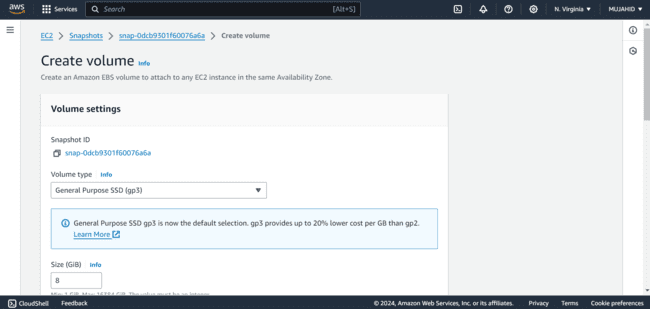


***Creating a New Volume with the Snapshot:***

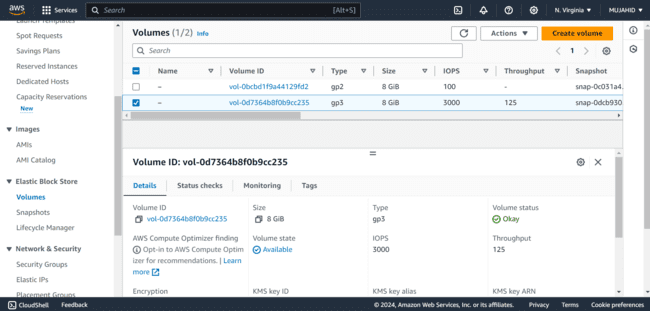
* *To create a New Volume out of snapshot that we’ve created out of a block volume, select the snapshot and click on Actions and then click on Create Volume from Snapshot.*

****

* *Keep everything as it is and scroll down and click on create volume.*

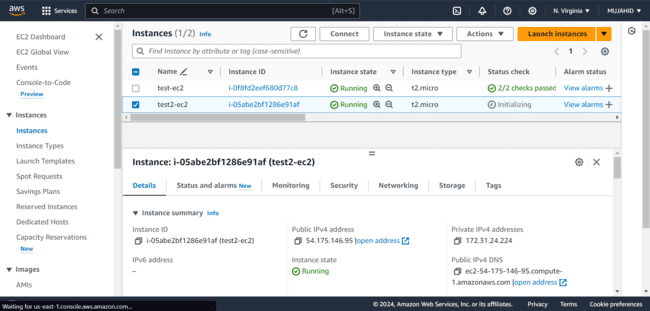
****

* *Now, the Volume is successfully created.*

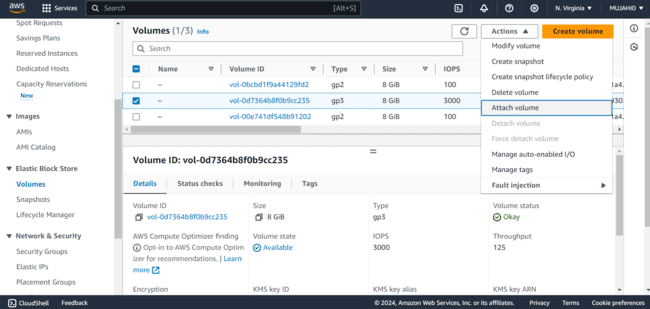
**

***Attaching it to New Server:***

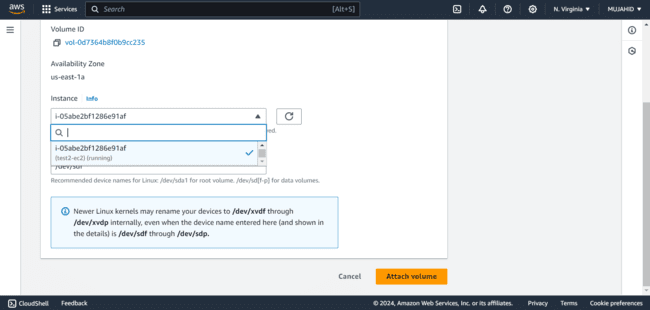
* *I have created a new Instance [test2-ec2] to attach the new volume created from snapshot.*



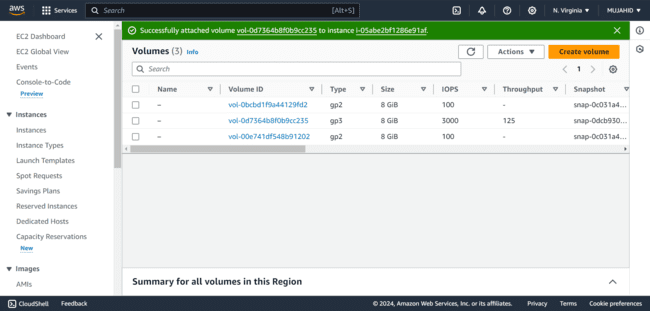
* *Select the new volume that you want to attach to the instance.*
* *Click on Actions and then click on Attach Volume.*



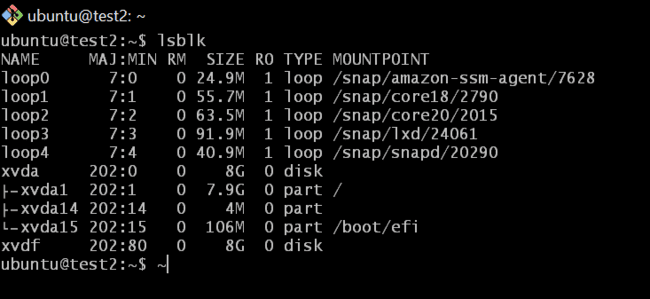
* *Select the Instance to which you want to attach the volume and then click on Attach Volume.*



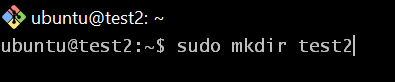
* *Now the volume is Successfully attached to Brand New Instance.*



* *This attached volume should be mounted to a directory inside instance.*
* *You can use command “lsblk” to check the block volume.*



* *Use command “sudo mkdir test2” to create a directory.*

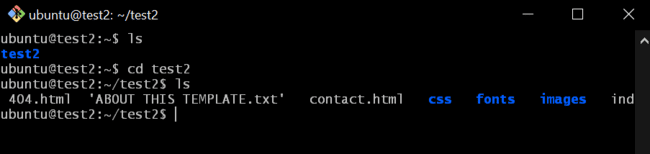


* *Mount the directory to the block volume using command*

*“sudo mount /dev/xvdf /home/ubuntu/test2”.*



* *get into the directory you created to check the content that you’ve added in the first place.*
* *Use command “cd” to enter into the directory and “ls” to list the content in the directory.*



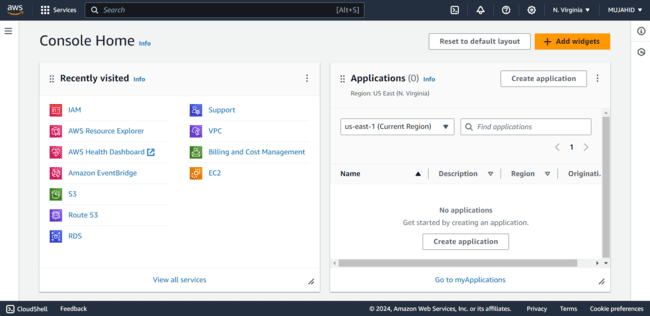
MINI PROJECT – 1

Lab – 7 [AMAZON MACHINE IMAGE AMI]

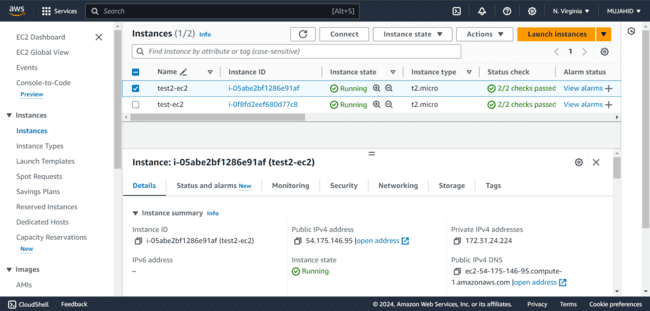
* *Log into AWS management console.*



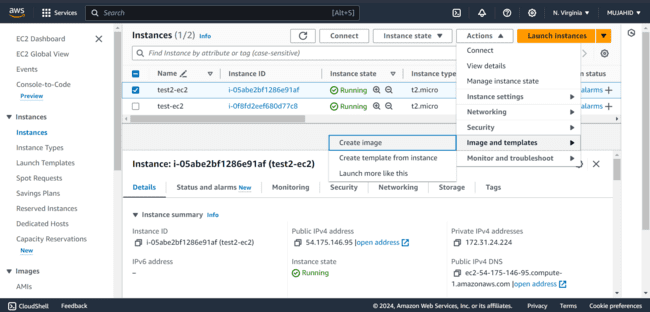
* *I logged in with my AWS credentials and entered into my AWS management console.*
* *Click on EC2 and in the left navigation panel under “Instances” click on “Instances”.*



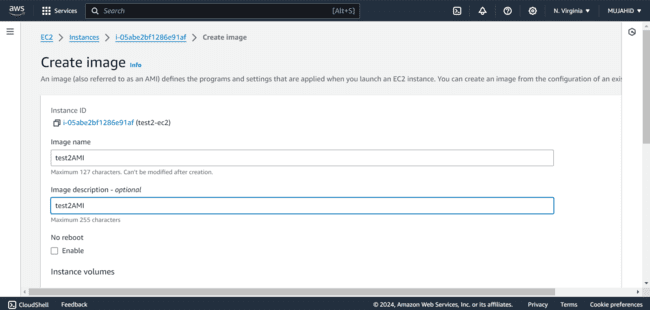
* *Select the running EC2 instance of which you want to create an AMI.*
* *I have selected “test2-ec2” Running Instance to create an AMI.*



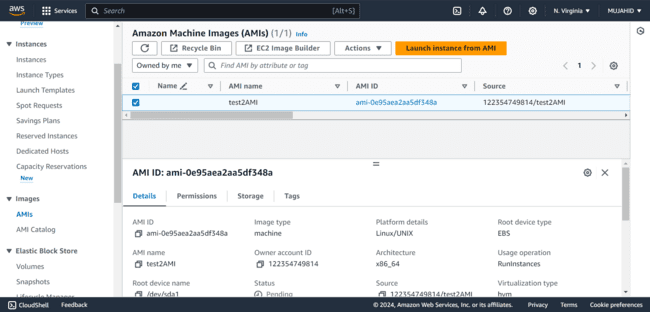
* *Click on Actions and then select Images and Templates and then select Create Image.*



* *Give an Image name and a Description, I have given both the same.*

**

* *Keep all the options as it is and scroll down and click on Create Image.*
* *The Image of a Running instance is successfully created.*
* *You can also check the created AMI in the Images AMIs section.*

**

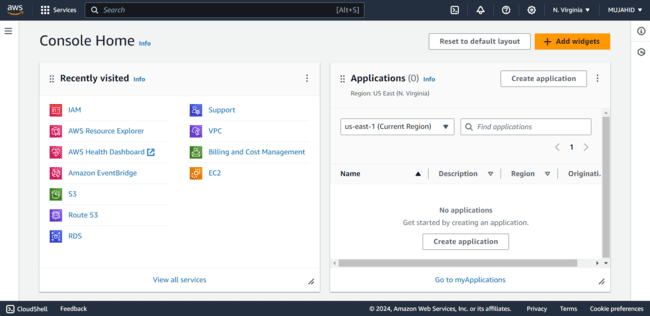
MINI PROJECT – 1

Lab – 8 [LOAD BALANCER]

* *Log into AWS management console.*

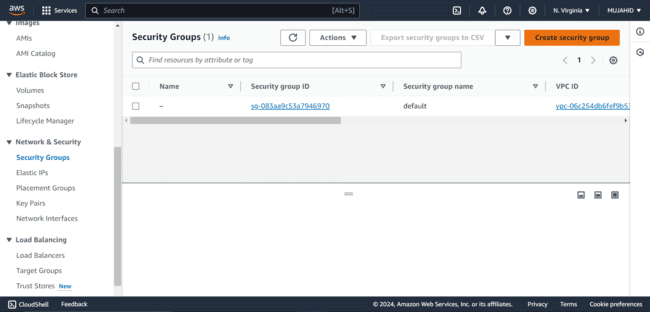


* *I logged in with my AWS credentials and entered into my AWS management console.*
* *Click on EC2 and in the left navigation panel under “Network and Security” click on “Security group”.*

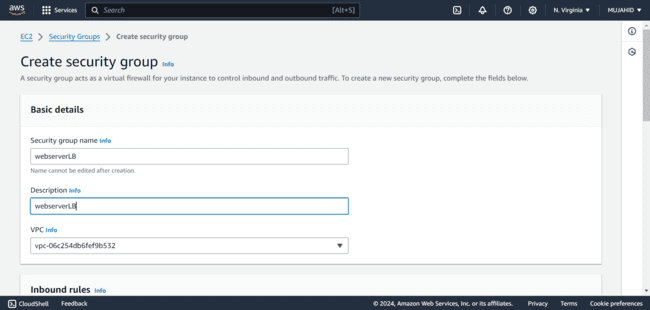


***Creating Security Groups:***

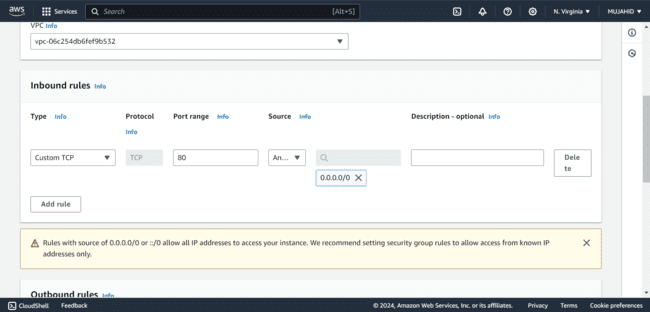
* *Click on create Security group on the top left corner.*

**

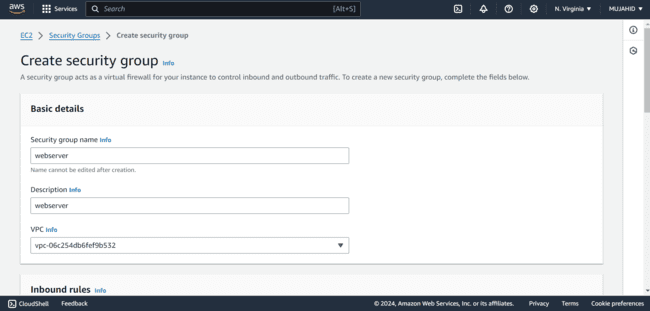
* *We have to create two Security Groups, One for Load Balancer and One for our Instances.*
* *First create a Security Group [SG] for Load Balancer.*
* *Give a name and description for the Load Balancer as show below.*
* *Keep the VPC as default.*

**

* *Give an Inbound rule of Port 80 Allowed Anywhere.*
* *Don’t add any Outbound rule, just keep it default, scroll down and click on Create Security Group.*

**

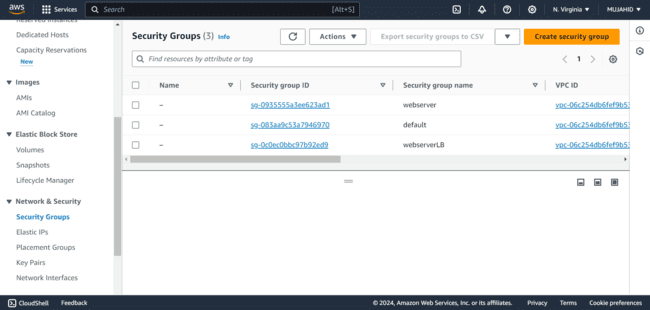
* *Then again click on Create Security Group to create a SG for our Webservers.*
* *Give a name and Description for our SG and keep the default VPC.*

**

* *In the Inbound rules add SSH allowed from My IP to login to the Instance and Port 80 Allowed from the Security Group of Load Balancer that we’ve created in the first place.*

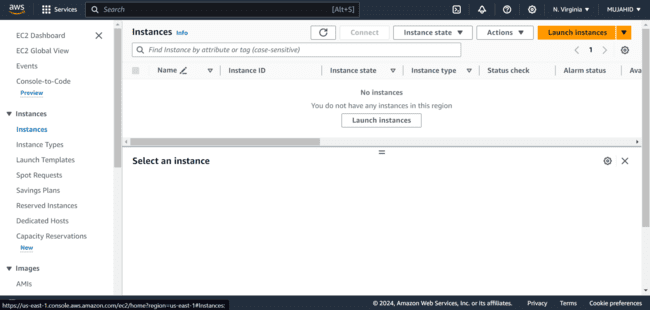
**

* *Scroll down and click on create Security Group.*
* *We have now created two Security Groups, One for the Load Balancer and One for the Webservers.*

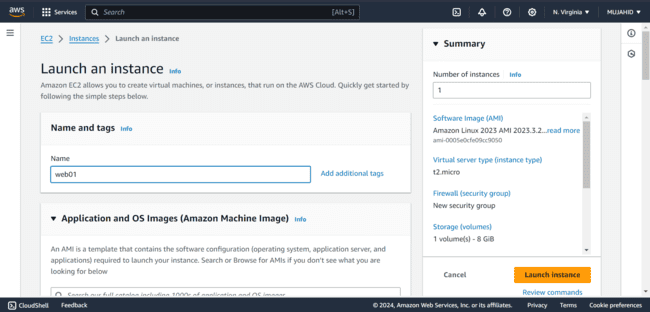
**

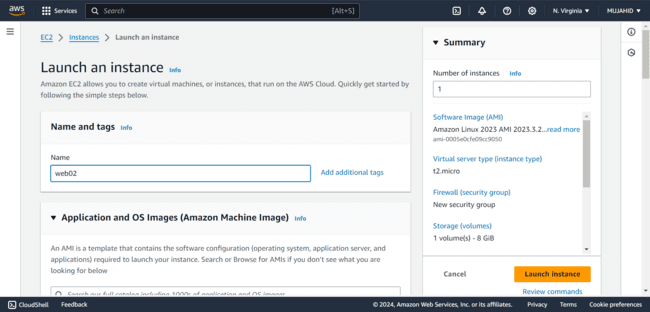
***Creating Instances:***

* *Now to create instances, Click on Instances and then click on Launch Instances.*

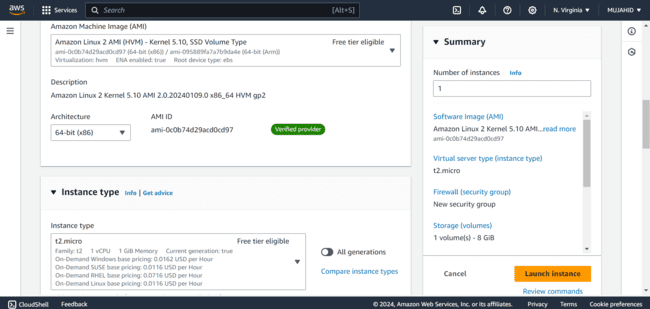
**

* *I have created two Instances, “web01” and “web02” with same AMIs and Configurations.*

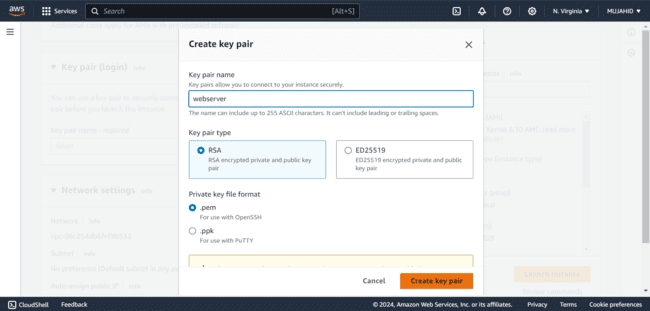
**

**

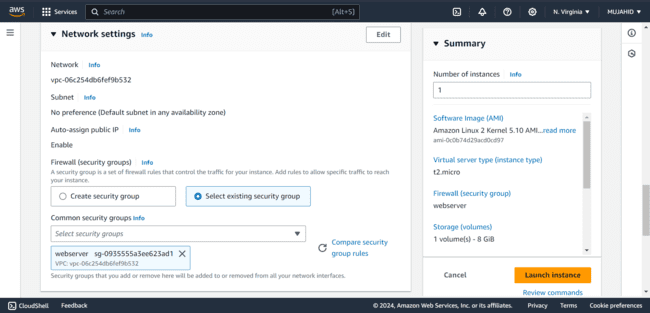
* *I have selected the AMI Amazon Linux 2 and Instance type as t2micro.*

**

* *For key pair I have created a new key pair named webserver for both the instances and saved it in my Local PC.*

**

* *In the Network settings I have selected an existing Security Group that I’ve created for the webserver in the start.*

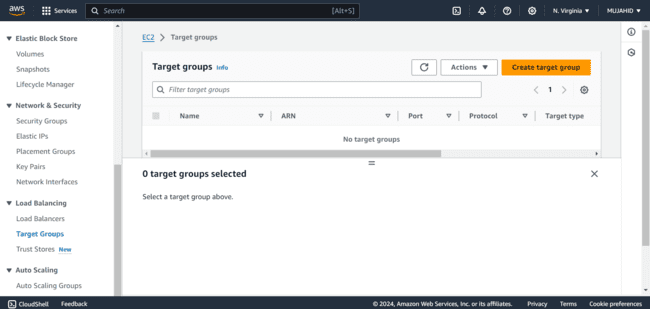
**

* *Now two Instances are created to setup a Load Balancer.*

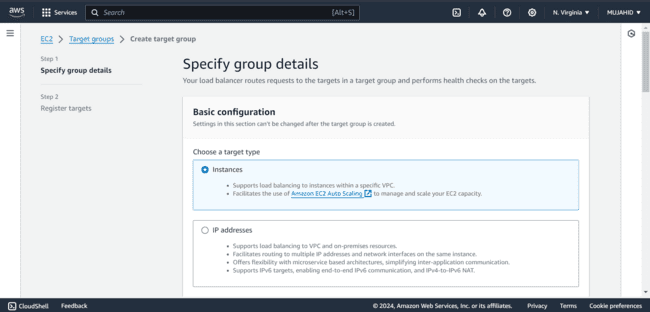
**

***Creating Target group:***

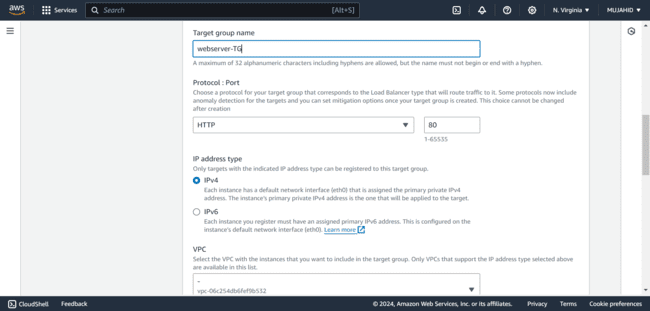
* *Now to Create a Load Balancer we have to create a Target Group first.*
* *On the left navigation panel under Load Balancing click on Target Groups.*
* *Click on Create Target Group.*

**

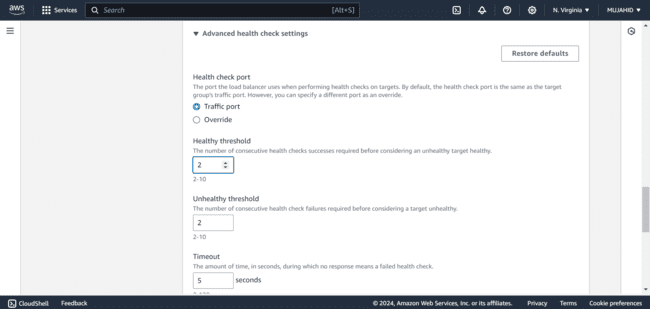
* *In the Basic Configuration select the target type as instances cause we are creating target group for instances that we have created.*

**

* *Give the Target group a name set the port to HTTP 80.*
* *Select IP address type as IPv4.*

**

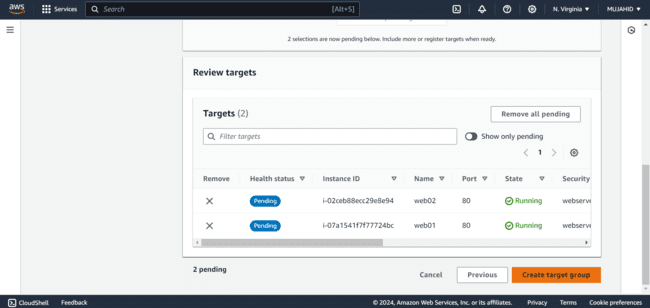
* *In the advanced health check settings, set healthy and unhealthy threshold as 2, it refers that the number of times to run health check to state the instance as Healthy or Unhealthy.*

**

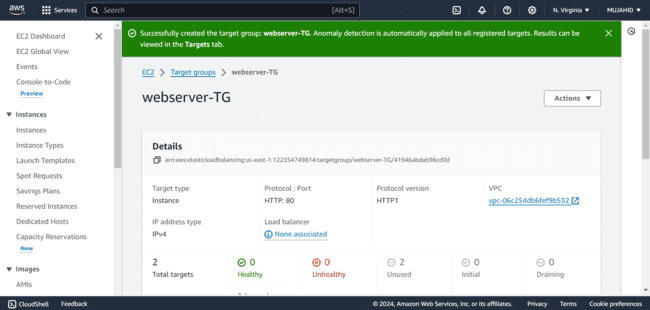
* *Scroll down and click on Next.*
* *In the Register Targets section, Select the Instances that you want to add to the target group and click on Include as pending below.*

**

* *You’ll see the instances that you’ve selected are added to the Target group.*
* *Click on Create Target Group.*

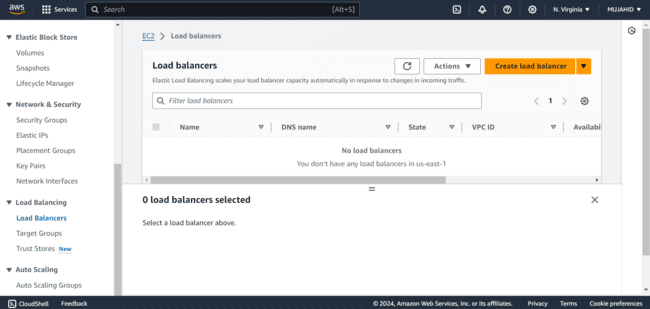
**

* *Now the Target Group is successfully created.*
* *Now its time to create a Load Balancer.*

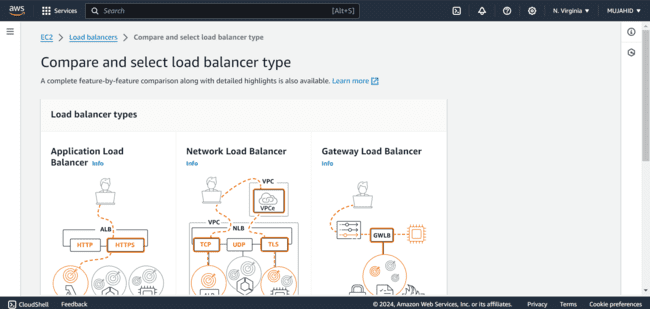
**

***Creating Load Balancer:***

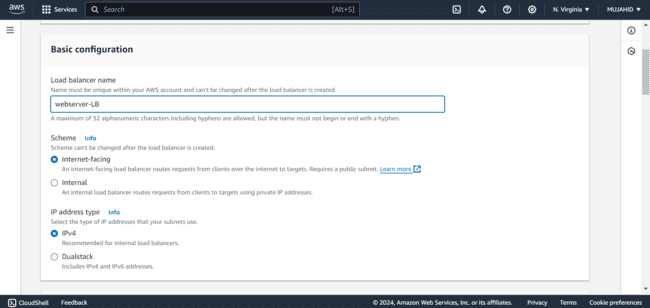
* *Click on Load Balancers on the left navigation panel under Load Balancing.*
* *Click on Create Load Balancer on the top right corner.*

**

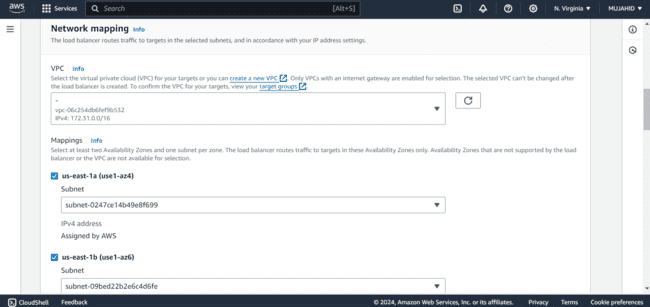
* *Select Application Load Balancer and click Next.*



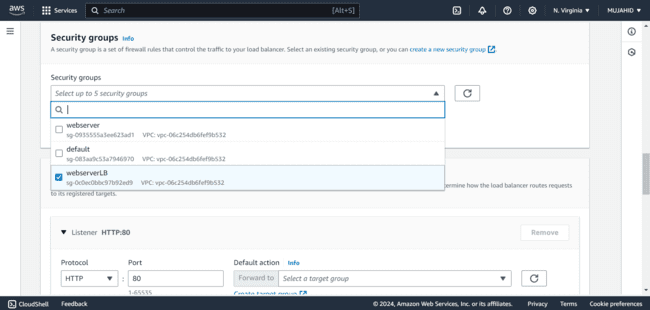
* *Give Load Balancer a name and leave everything to default.*

**

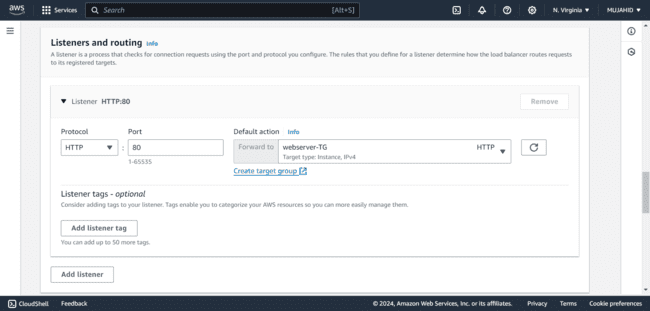
* *In Network Mapping select the availability zones. More number of Availability zones More the availability of web application.*

**

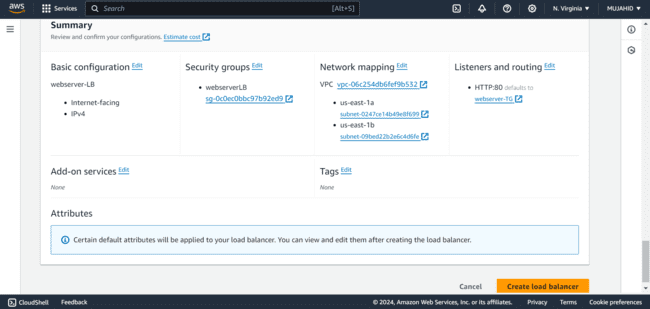
* *Select the existing Security Group that we’ve created in the first place for the Load Balancer.*

**

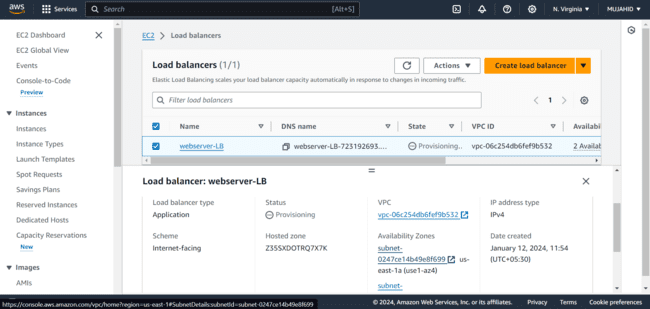
* *In the Listeners and Routing set the Port 80 listened to the Target Group that we created [webserver-TG].*

**

* *Check in the summary all the configurations that you made and click on Create Load balancer.*

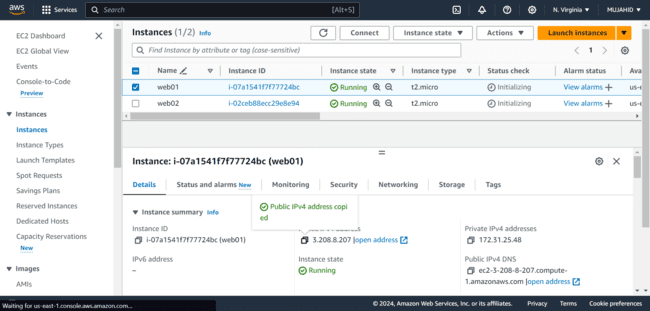
**

* *Now the Load Balancer is Successfully created.*

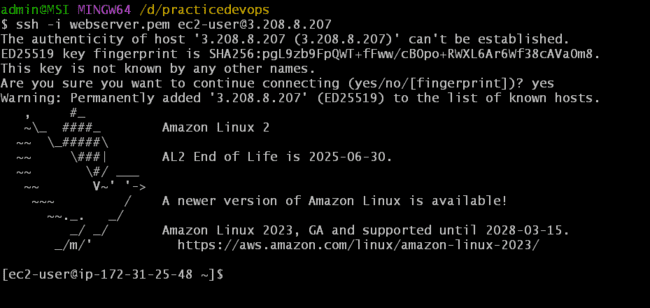
**

***Configuring Instances:***

* *We have to run Static web Application in the instances that we created.*
* *First, select the first instance [Web01] and copy it Public IP to connect to it.*

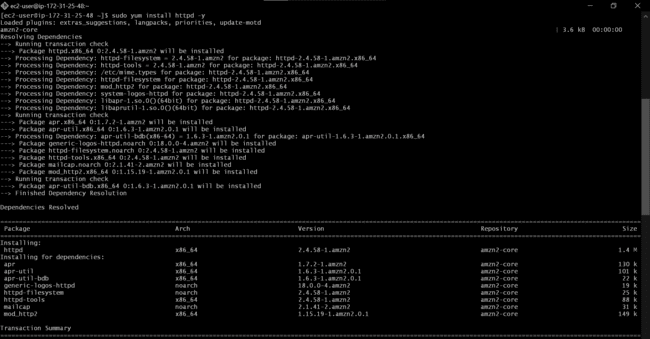
**

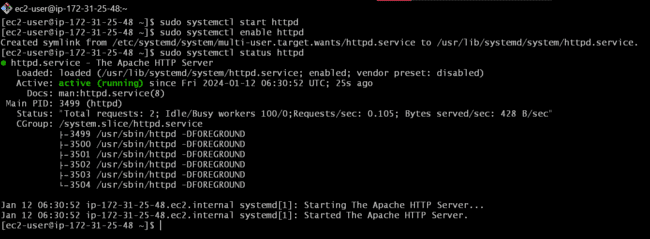
* *Use the command “ssh -i [keypair] [username]@publicIP” at the key path to connect to the instance as I have done below.*

**

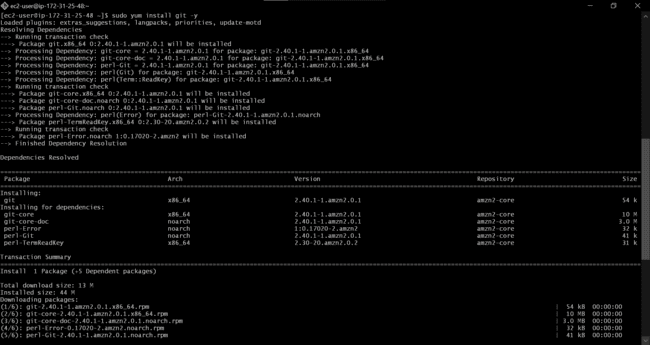
* *I have installed, Started and Enabled the service HTTPD on my web01 instance with the following commands:*

1. *“sudo yum install httpd -y”*
2. *“sudo systemctl start httpd”*
3. *“sudo systemctl enable httpd”*

**

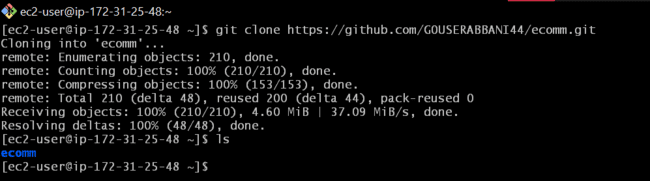
**

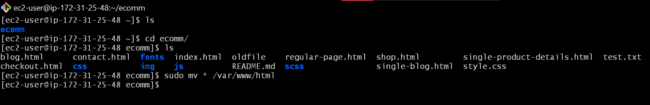
* *I have installed GIT to clone the source code from Git hub remote platform with the command “sudo yum install git -y”.*

**

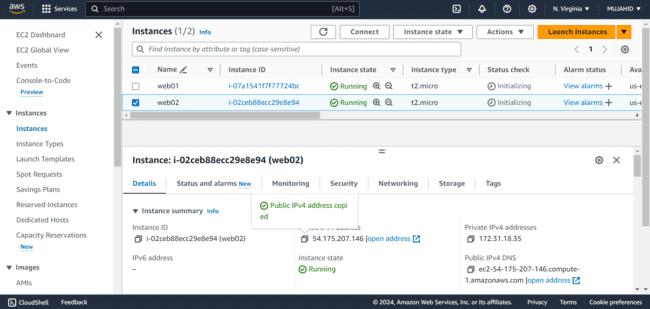
* *Cloned the source code from Git hub using command*

*“git clone URL”*

**

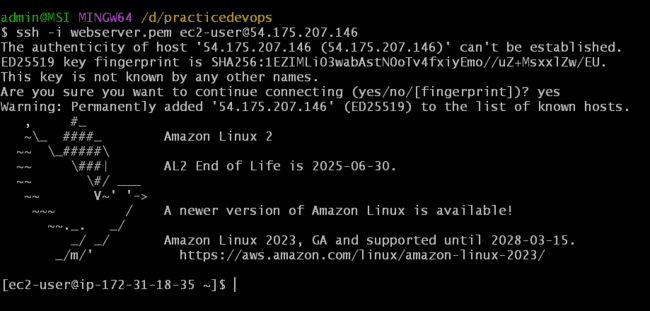
**

* *I did the same process to my other instance [web02] as I did to my first instance [Web01].*
* *Connected to it using public IP of the Instance.*

**

* *Logged in to it using the command*

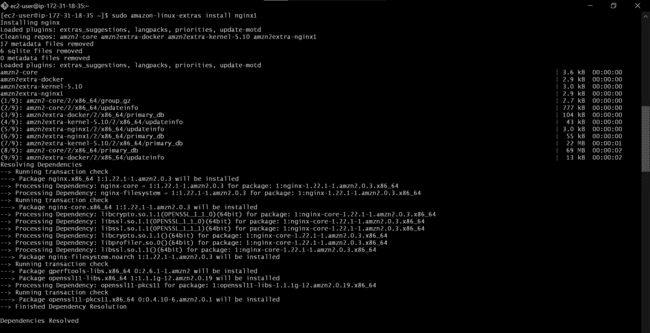
*“ssh -i [keypair] [username]@publicIP”.*

**

* *Installed the services NGINX and GIT using commands*

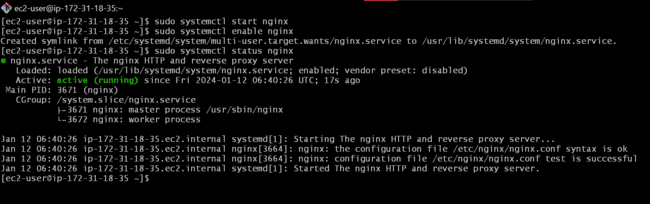
*“sudo amazon-linux-extras install nginx1 -y”*

*“sudo yum install git -y”*

**

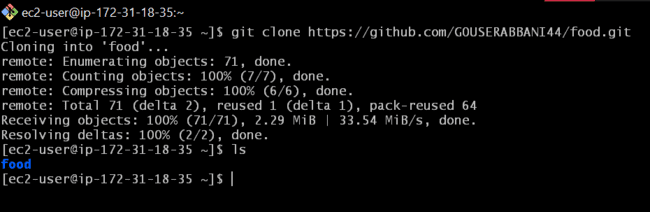
* *Started and Enabled the NGINX service using commands*

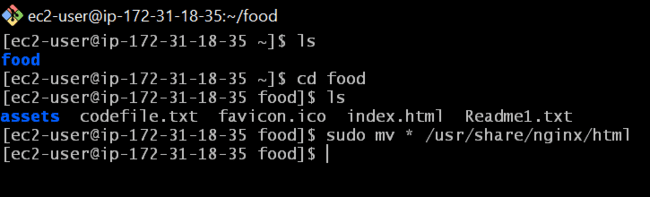
1. *“sudo systemctl start nginx”*
2. *“sudo systemctl enable nginx”*

**

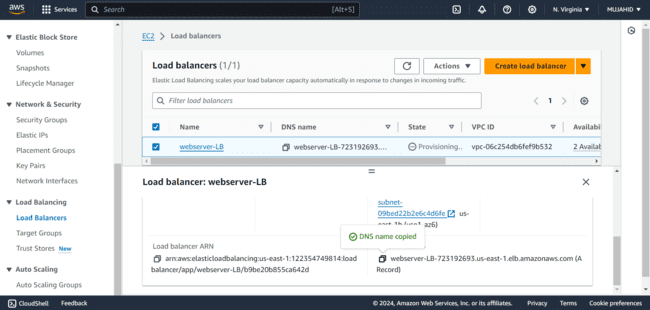
* *Cloned another source code from Git hub using command*

*“git clone URL”.*

**

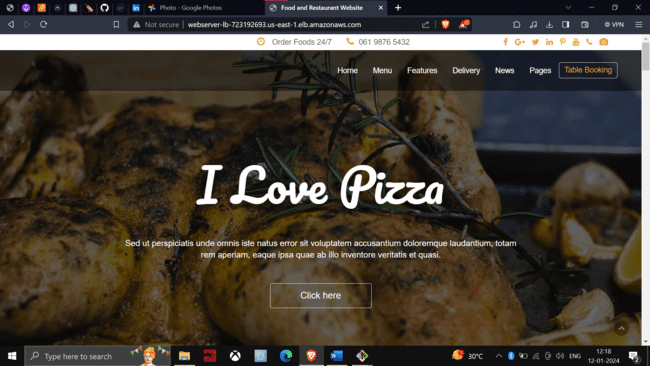
**

* *After configuring both the instances with different services and with different static web applications I’ve copied the DNS name of the Load Balancer and browsed it in the browser.*

**

* *This is the result I got.*
* *Both the applications are shuffling and working fine.*
* *The Load balancer is routing the traffic to one of the instances each time I refreshed.*

**

**

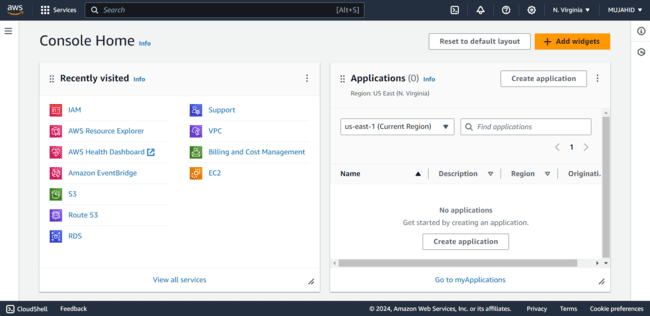
MINI PROJECT – 1

Lab – 9 [ASG AND LC]

* *Log into AWS management console.*

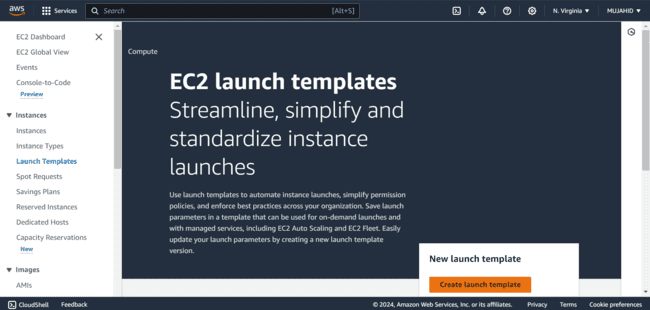


* *I logged in with my AWS credentials and entered into my AWS management console.*
* *Click on EC2 and in the left navigation panel under “Instances” click on “Launch Templates”.*

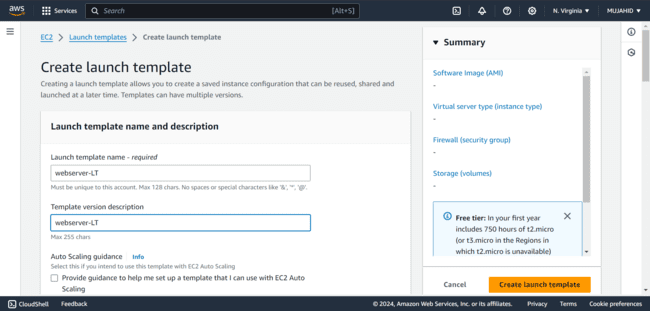


***Creating Launch Template:***

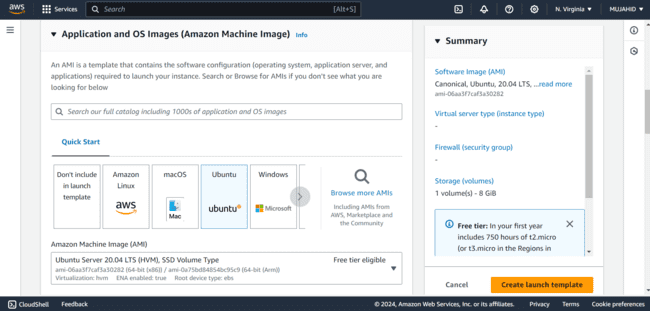
* *Click on Create Launch Template on the bottom right corner of the window.*

**

* *Give a Name and Description to the Launch template.*

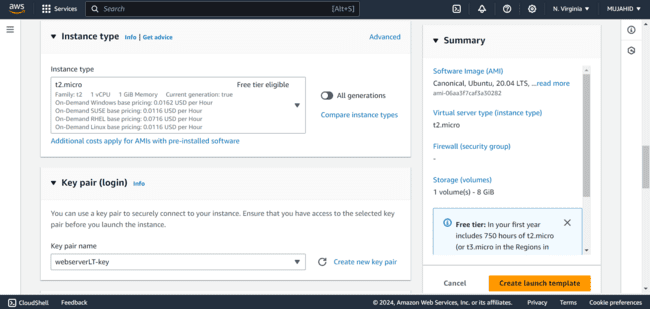
**

* *Select the Ubuntu 20.04 AMI Free tier in the Application and OS images.*

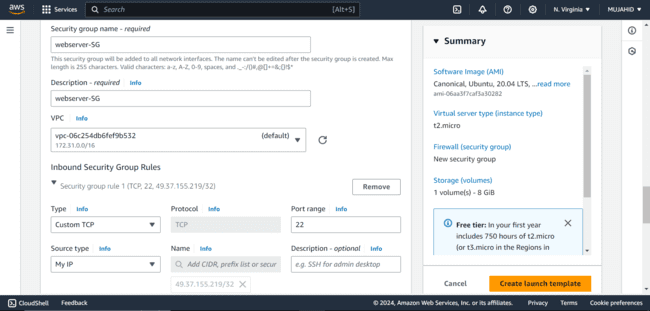
**

* *Select the instance type as t2micro.*
* *Create a Keypair to your Launch Template to login to the instance*

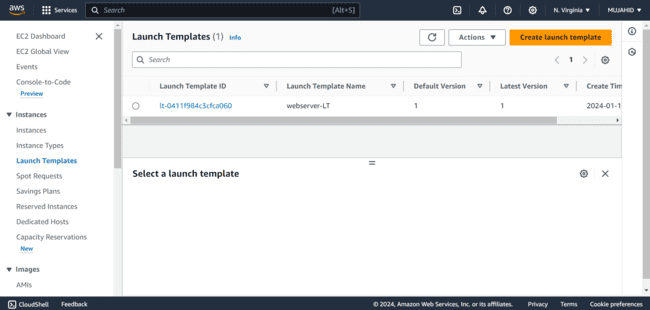
*[webserverLT-key].*

**

* *Create a new SG with name and description [webserver-SG] and keep the VPC as default.*
* *Give an Inbound rule SSH port 22 allowed from MyIP.*

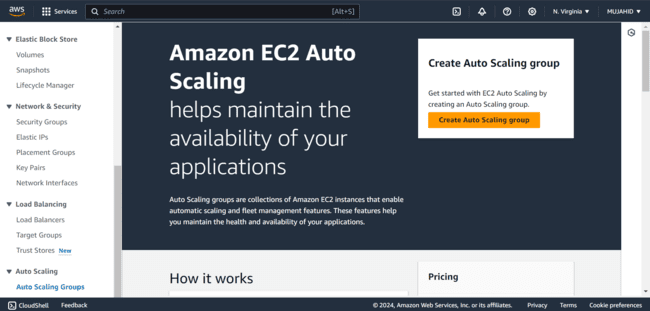
**

* *Scroll down and click on Create Launch template.*
* *Now the Launch Template is successfully created.*

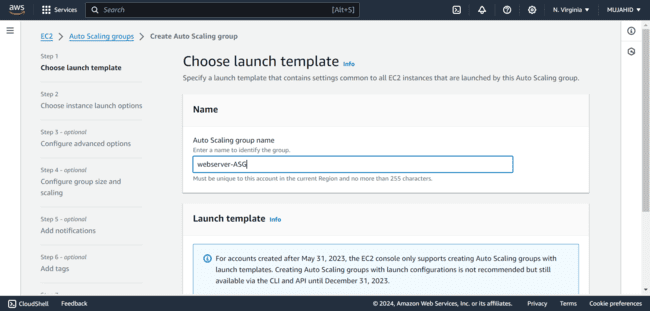
**

***Creating Auto Scaling Group [ASG]:***

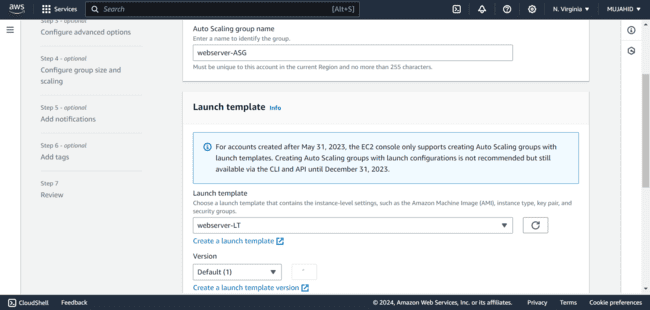
* *On the left navigation panel after clicking on EC2 under Auto Scaling click on Auto scaling Group [ASG].*

**

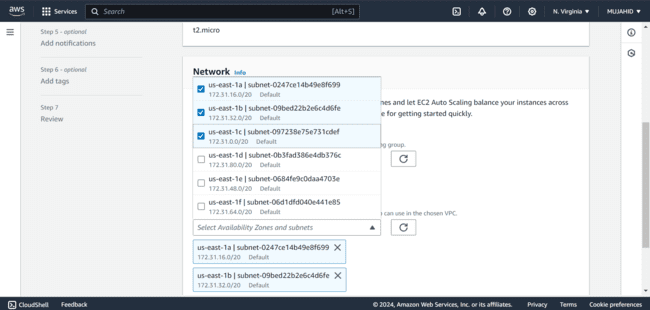
* *Give a name to your Auto scaling group [webserver-ASG].*

**

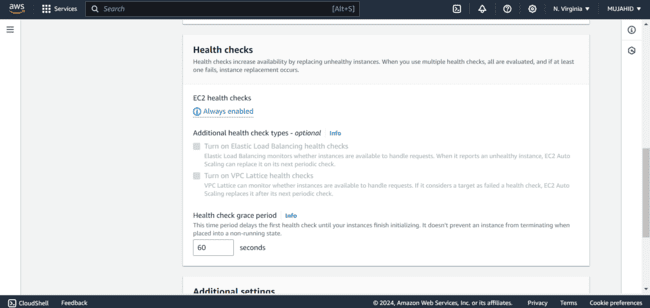
* *In the Launch template section select the launch template that we created earlier.*

**

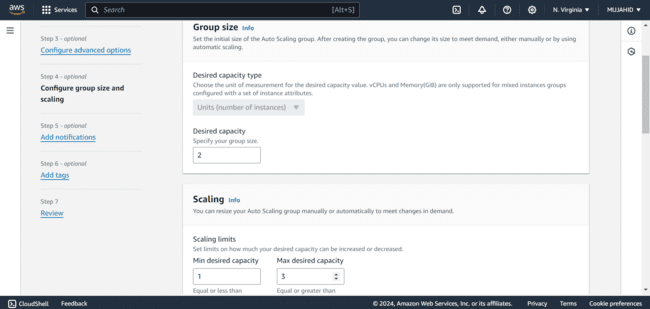
* *In the Network section, select the availability zones in which your instances want to be created.*
* *More the availability zones, High the availability of your instance.*

**

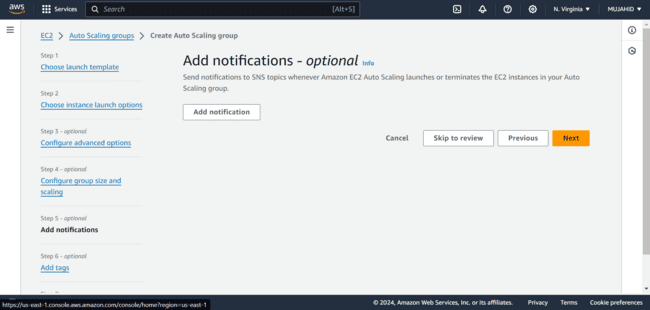
* *In the Health checks sections, set the health check period as 60 seconds or more.*

**

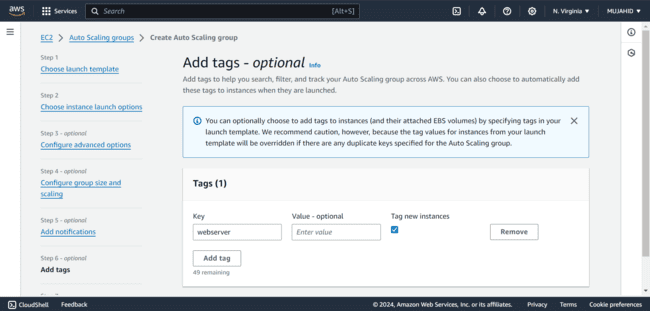
* *In the group size section, set the Desired capacity as 2 and the scaling limits as Minimum desired capacity as 1 and Maximum desired capacity as 3.*

**

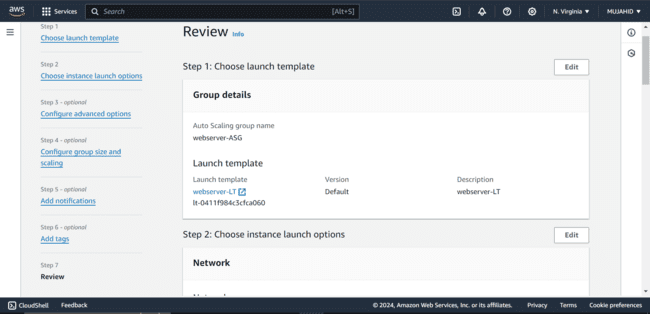
* *You can add Notification if you want by creating a SNS topic if not already created.*

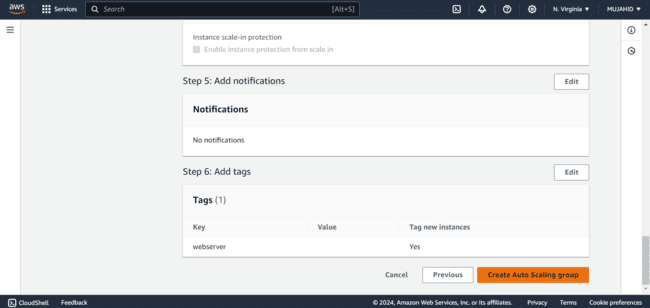
**

* *Add tags [webserver].*

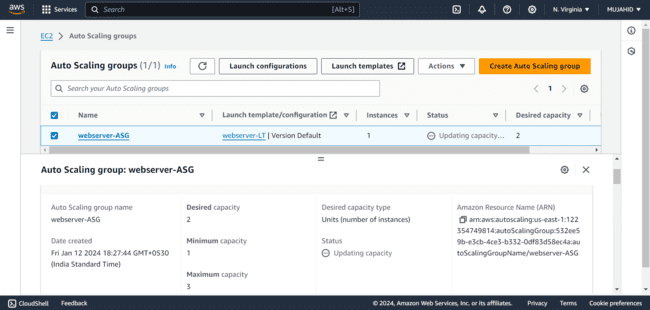
**

* *Review all the configurations you made in the Review section and click on Create Auto Scaling Group.*

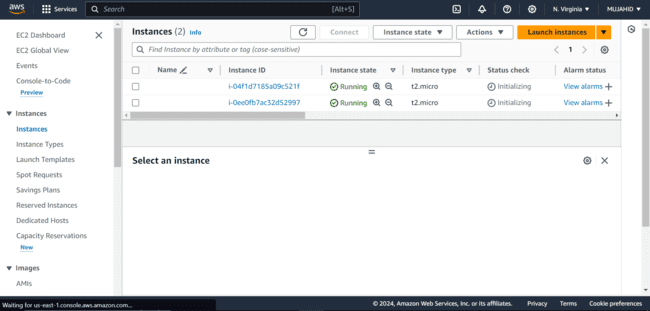
**

**

* *Finally, Auto Scaling Group is created [webserver-ASG].*

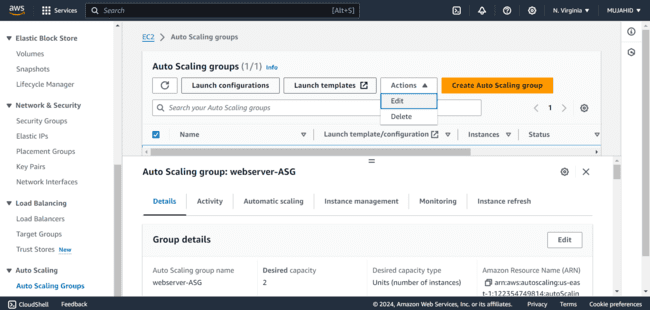
**

* *We can see the instances of our desired capacity are being created.*

**

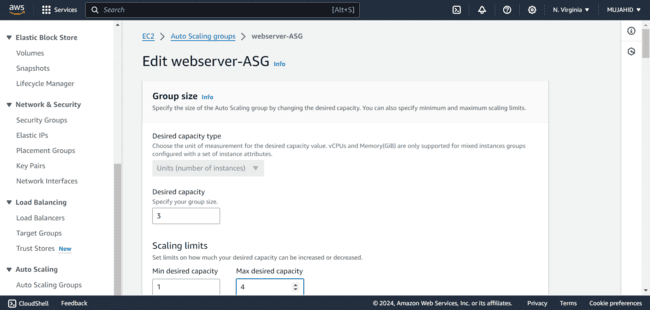
***Changing the Desired Capacity of our Auto Scaling Group:***

* *In the Auto Scaling section select the auto scaling group that you want to modify.*
* *Click on Actions and then click on Edit.*

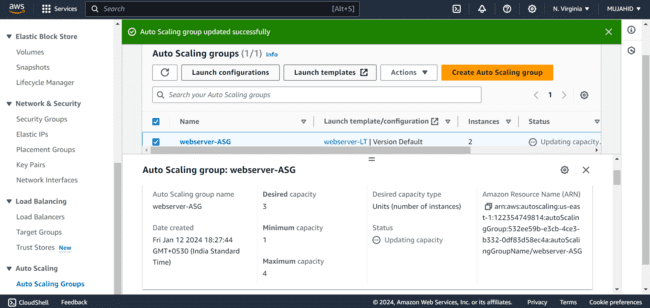
**

* *In the Group size section change the desired capacity from 2 to 3*

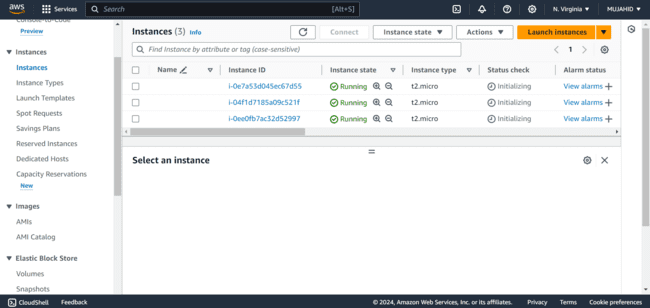
*And change the Maximum capacity from 3 to 4 and then click on update Auto Scaling Group.*

**

* *Auto Scaling Group is Updated successfully.*

**

* *We can see the number of instances being increased from 2 to 3 as per our desired capacity.*

**

MINI PROJECT – 1

Lab – 10 [RELATIONAL DATABASE SERVICE RDS]

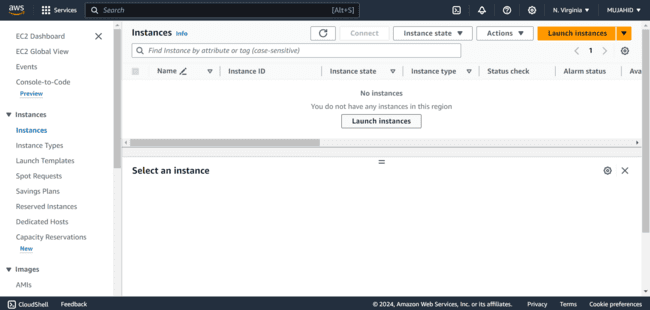
* *Log into AWS management console.*



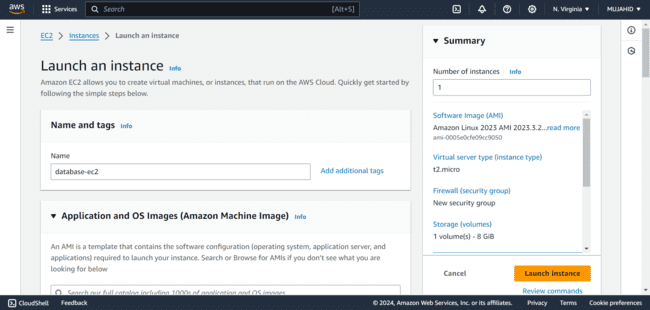
* *I logged in with my AWS credentials and entered into my AWS management console.*
* *Search for EC2 in the services section*

***Creating an Instance:***

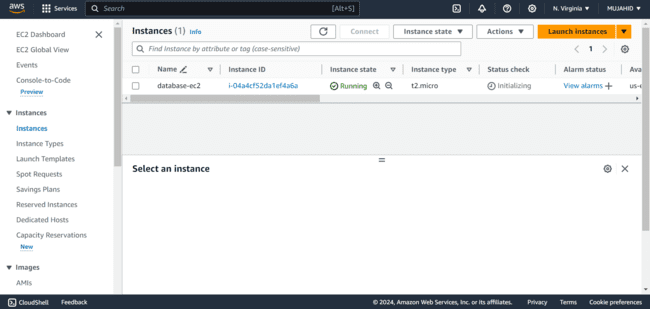
* *Click on launch instances to create one.*

**

* *Give your EC2 Instance a name [database-ec2].*

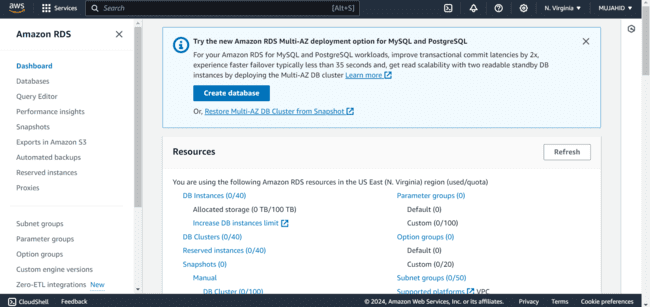
**

* *I chose Amazon linux 2.*
* *Created a new key pair for the Instance.*
* *In the type of instance, I selected t2micro.*
* *Created a new Security Group with an Inbound rule allowing Port 22 from my IP to connect to the Instance later.*
* *Finally an Instance for the Database is created.*

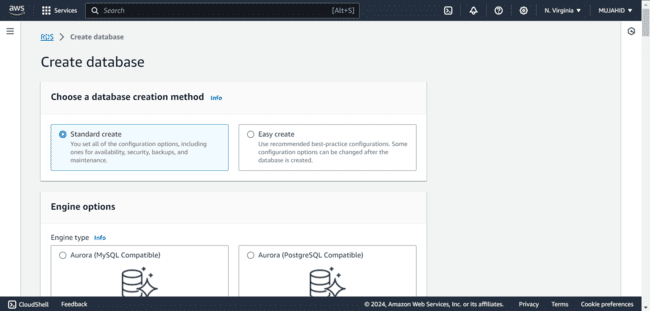
**

***Provisioning RDS Instance:***

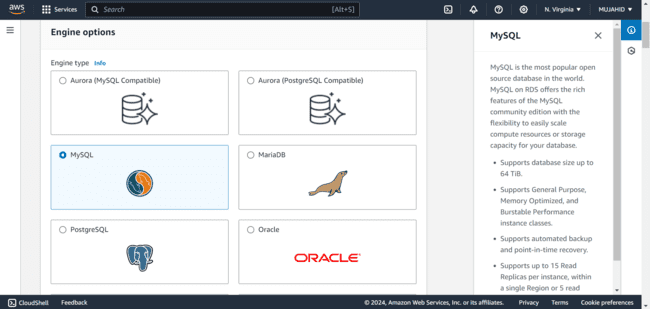
* *Click on Create Database.*

**

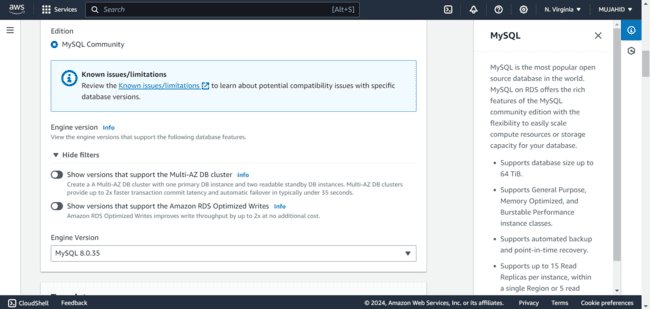
* *In the database creation method select the Standard Create.*
* *It refers that you can select all of the configuration options including security, backups and maintenance.*

**

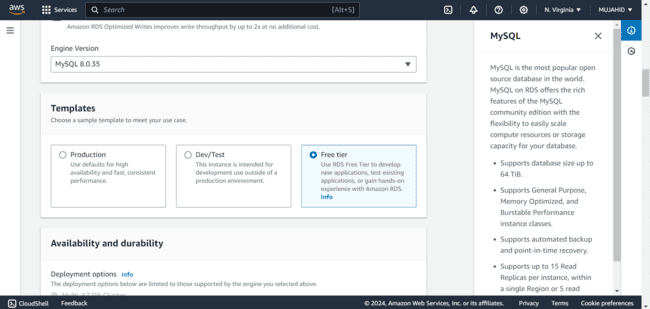
* *In the Engine options select MySQL Database.*

**

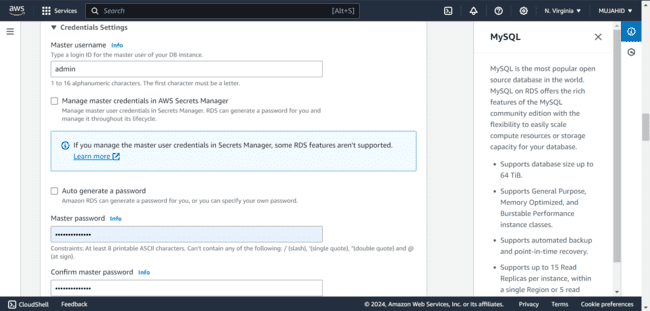
* *In the engine versions, select the version of your selected database.*

**

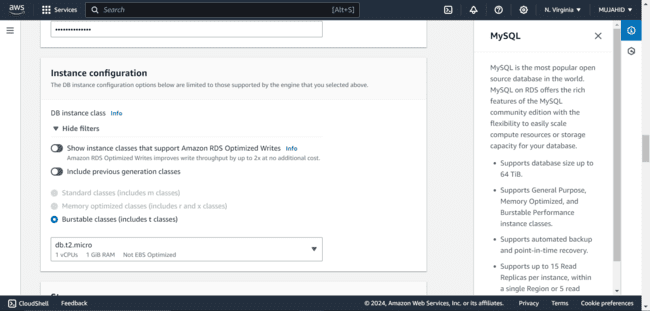
* *Select Free tier in the templates*

**

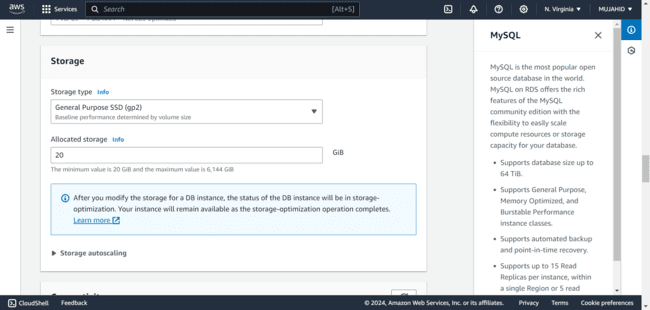
* *Give your database a Master username and Master password or you can allow AWS to generate a password for you.*
* *Remember whatever the Username and Password you gave cause you’ll need them later to access your database.*

**

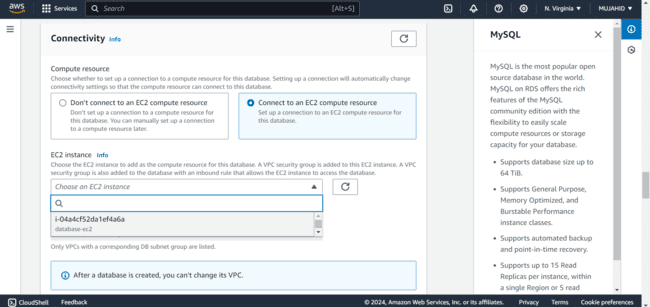
* *In the Instance configuration select the type just like we do while creating an Instance.*
* *Here db.t2micro is selected by default which gives us free tier.*

**

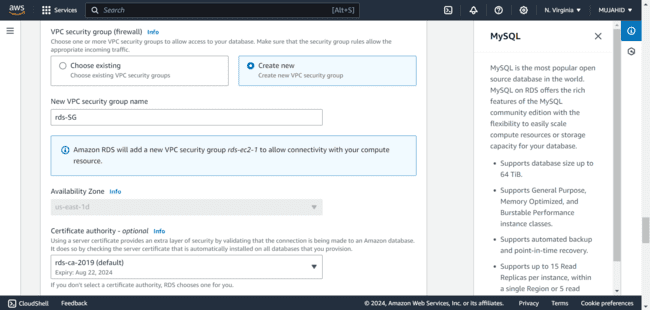
* *In the storage type, General purpose SSD is selected by default and gives us 20GB of allocated storage under free tier.*

**

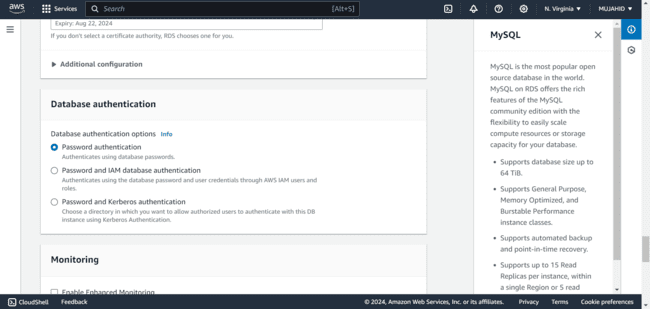
* *In the connectivity, choose connect to an EC2 compute resource and select the instance that we created in the first place for the database.*
* *You’ll get that option in the dropdown menu.*

**

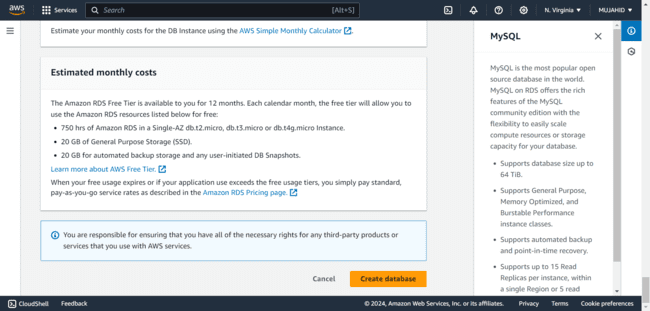
* *Create a new Security Group for the database by giving it a name just I did below.*

**

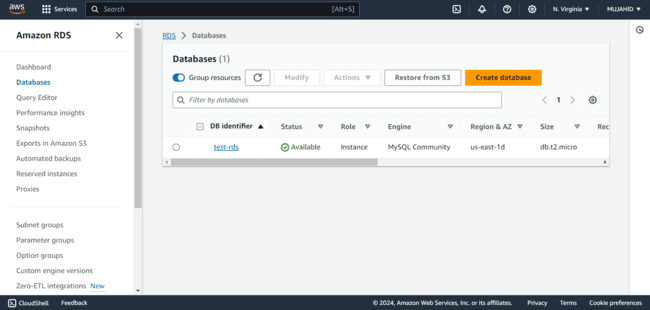
* *In the database Authentication, For simplicity, you can use the default password authentication, and also that is selected by default by AWS.*

**

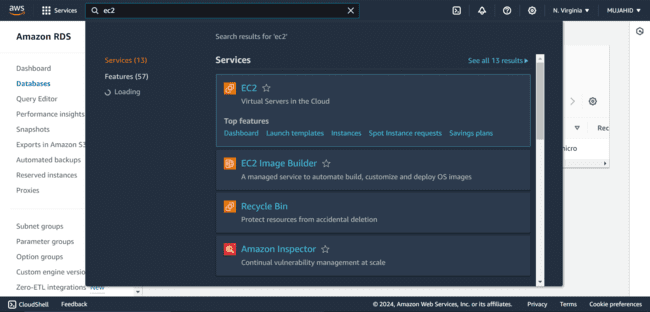
* *Review all the configurations you made and click on Create database.*

**

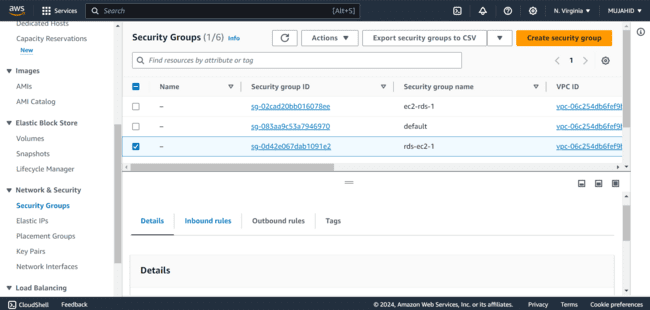
* *After a few minutes the database is created with your configurations.*

**

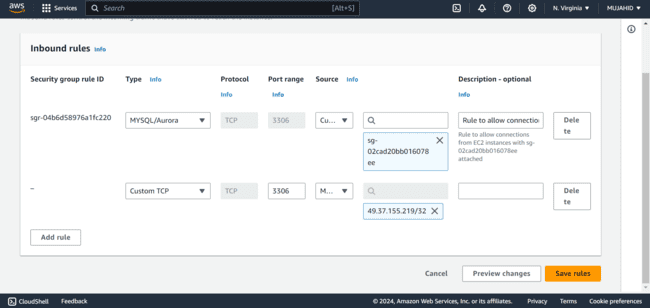
* *In the services search bar, search for EC2 and under Network and Security click on Security groups.*

**

* *Select the Security Group that you created for the RDS database and click on Actions and then click on Edit.*

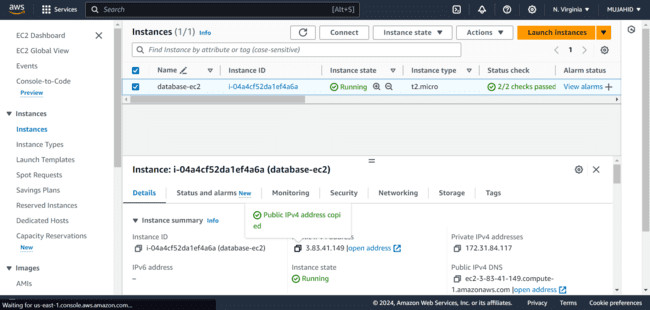
**

* *Add a rule allowing Port 3306 allowed from My IP.*

**

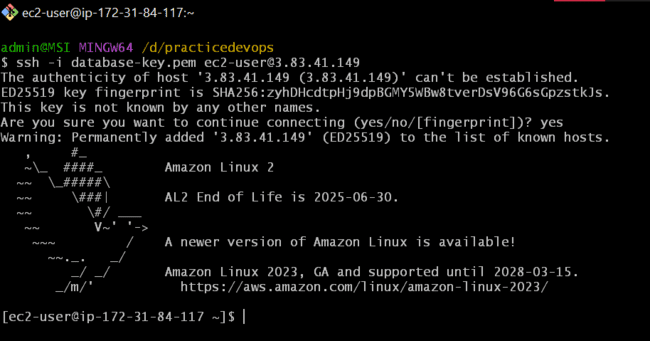
***Accessing RDS from EC2 Instance:***

* *Copy the public IP of your database EC2 Instance and open git bash to access it.*

**

* *Get in to the path where your key pair is saved and use this command to login to your Instance*

*“ssh -i keypair username@IP”*

**

* *Install the MySQL database using command*

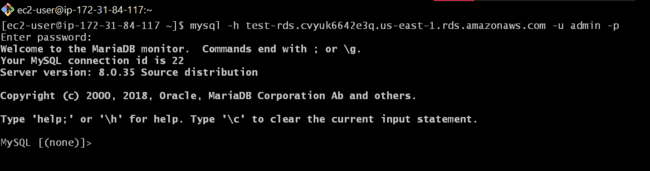
*“sudo yum install mysql -y”*

**

* *To access your database use command*

*[mysql -h <RDS\_ENDPOINT> -u <USERNAME> -p]*

* *Here <RDS\_ENDPOINT> refers to the endpoint provided by your RDS database.*
* *<USERNAME> refers to the username you created while provisioning RDS database.*
* *After clicking upon Enter, you’ll be asked a password to login.*
* *Provide the password you set along with the Username.*
* *You’ll finally enter into your MySQL database.*

**