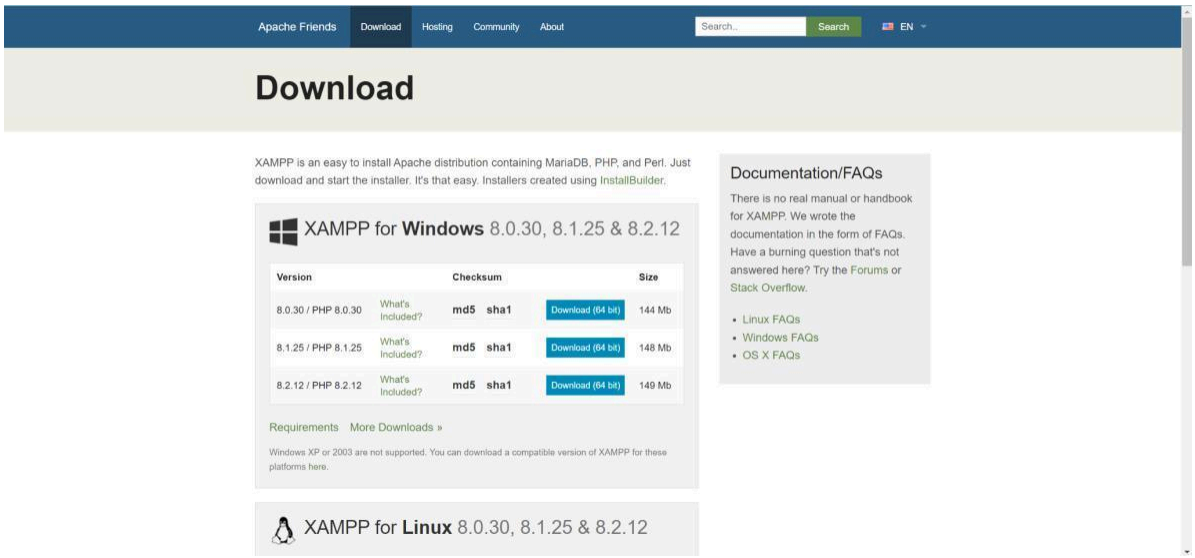
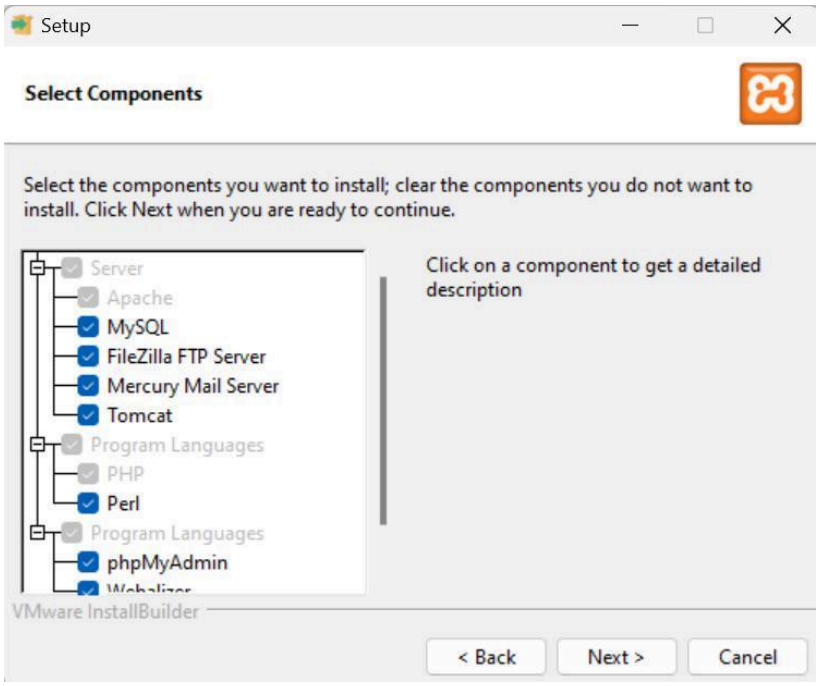
Expt 1a]Website Hosted Locally on XAMPP

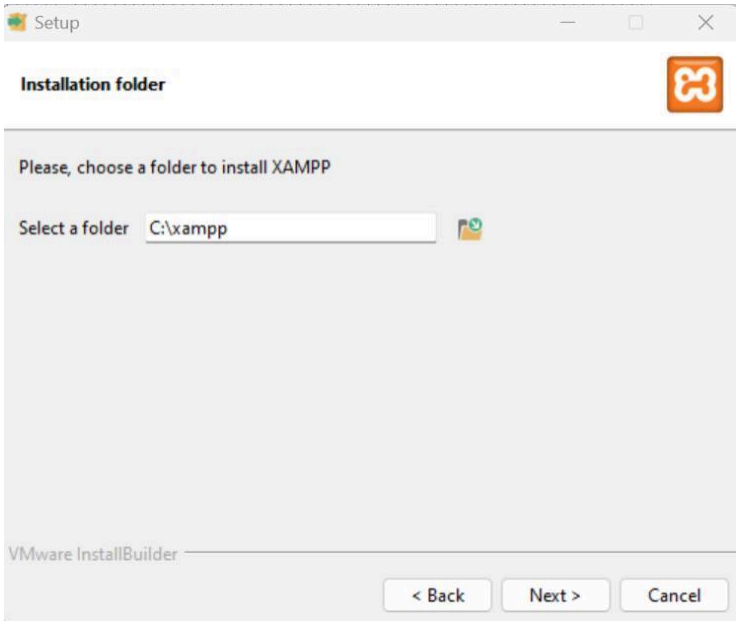
Step1: To download the latest version of XAMPP (8.2.12 / PHP 8.2.12) for Windows, visit the official XAMPP website, navigate to the download section, and select the appropriate version for your system.
https://www.apachefriends.org/download.html



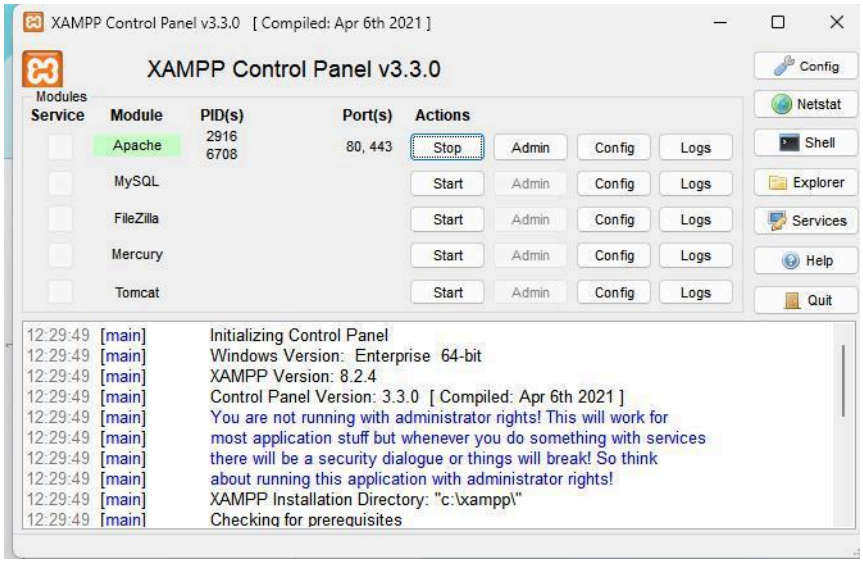
Step 2: XAMPP offers various components during installation, including Apache (web server), MySQL (database management), PHP (server-side scripting language), and more. These components are essential for running a local development environment. You can choose to install only the components you require for your specific project.



Step 3: Choose the destination folder where you want to install XAMPP and store your local files. This path will be where all your web server files and configurations are saved.

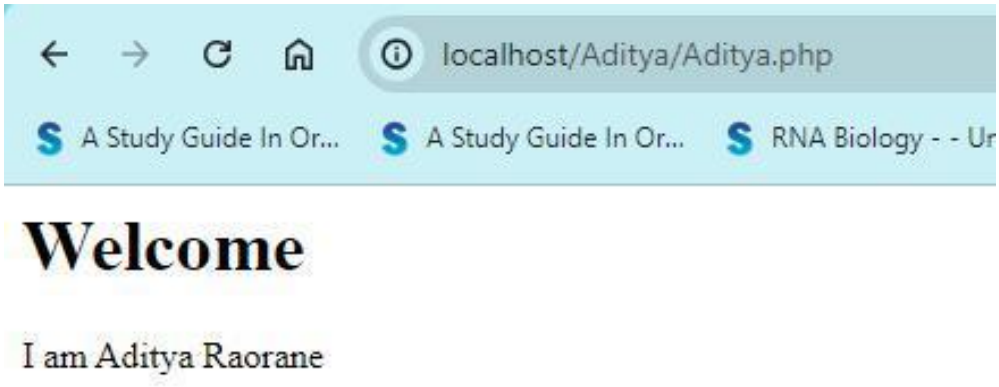


Step 4:After successfully downloading XAMPP, open the XAMPP Control Panel, and under the "Module" section, click "Start" next to Apache to activate your local server.



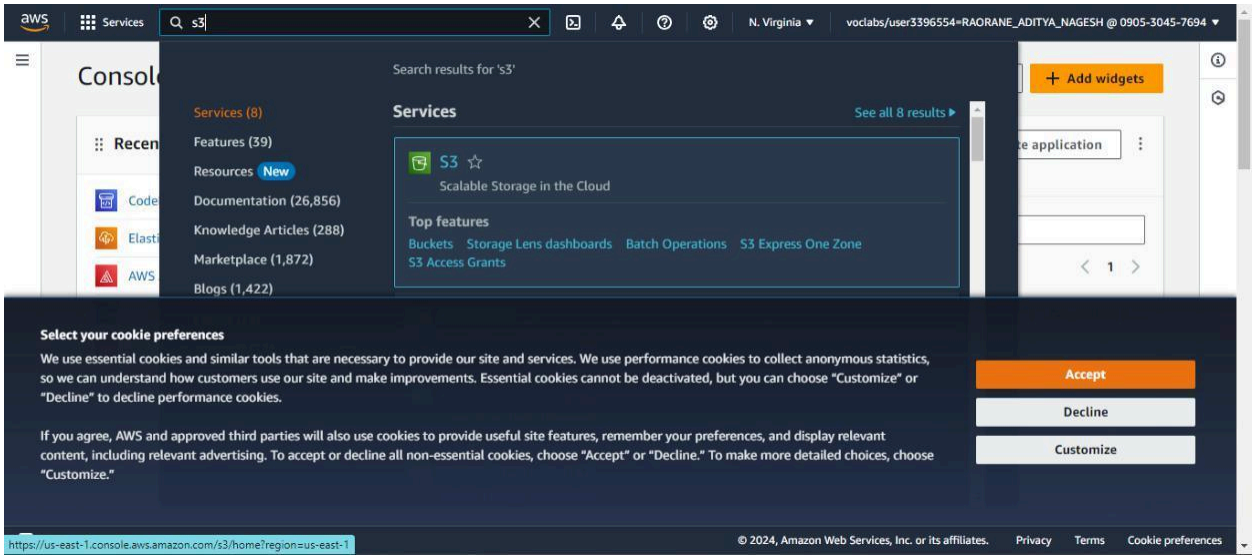
Step 5: Go to the following directory and save your files there so that they can be locally hosted

Step 6: Open your web browser and navigate to `localhost`, followed by your folder name (e.g., `localhost/your-folder-name`). You will see a list of available PHP files—select one to run it, and it will be hosted successfully.

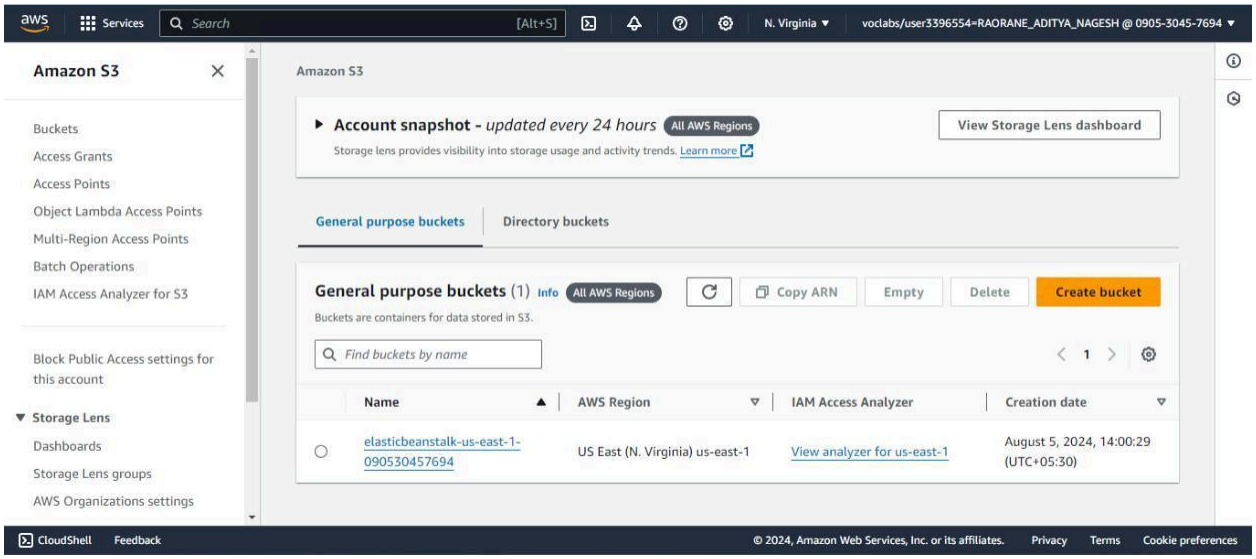


1a] Website Hosted Remotely on AWS S3 Bucket

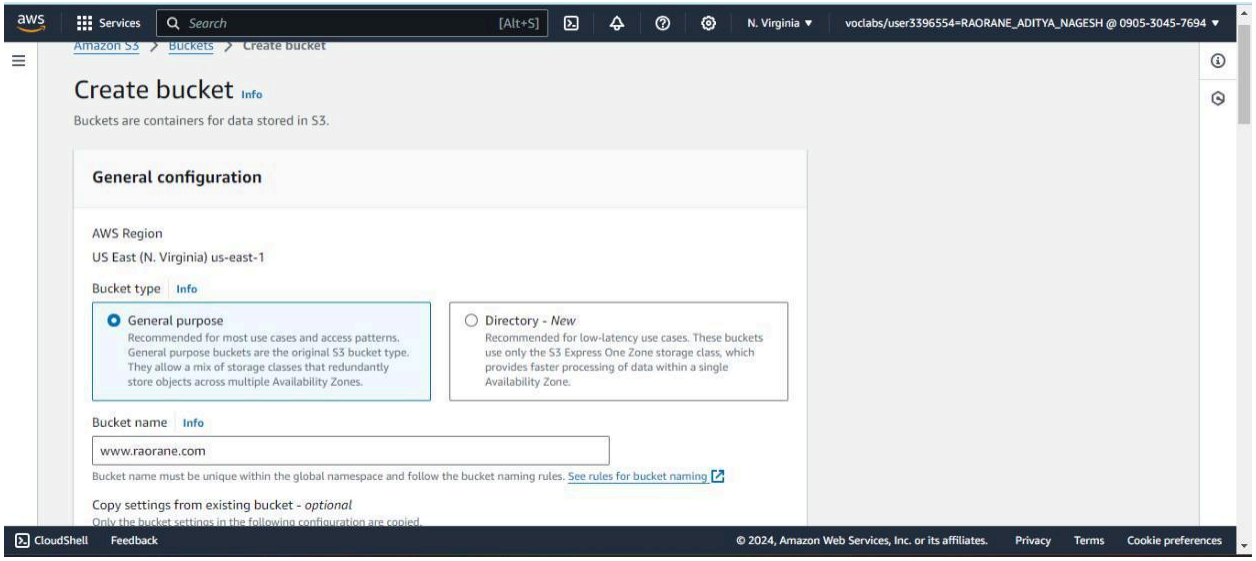
Step 1: In your AWS Academy account, navigate to the "Services" search bar, type "S3" and open it to access the Amazon Simple Storage Service.



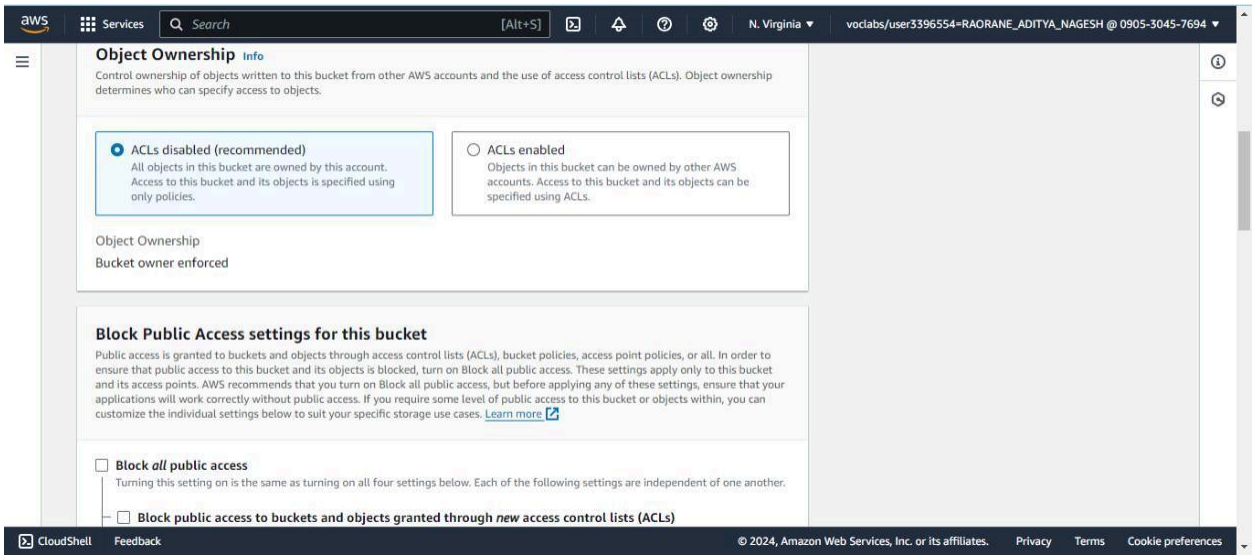
Step 2: In AWS S3, a bucket is a container for storing objects, with globally unique names and customizable configurations for data management and access control.Click on “Create Bucket” to create one.



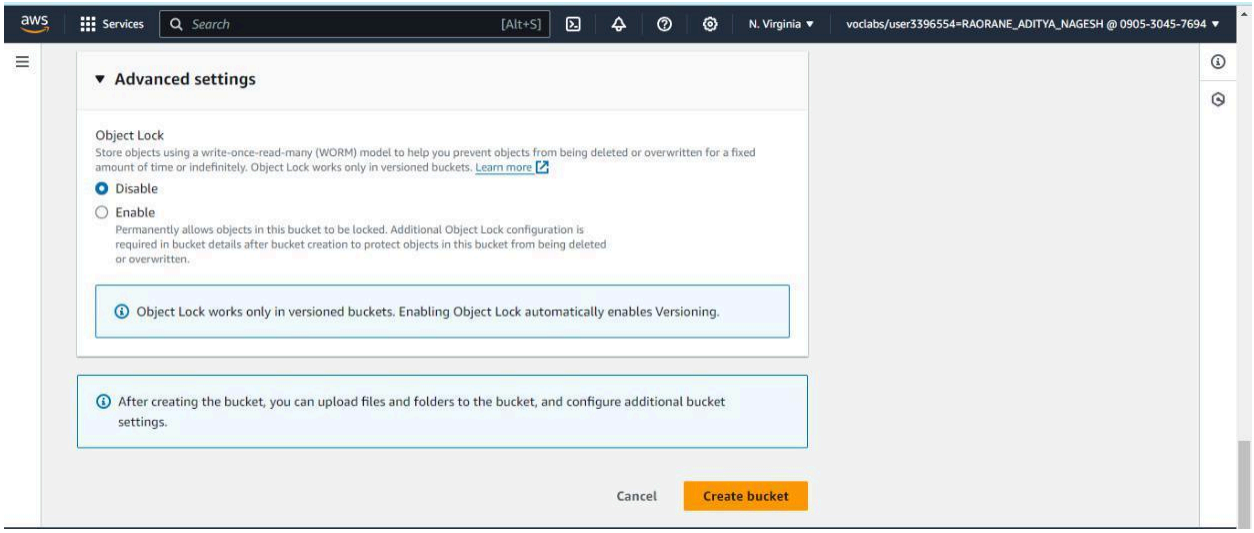
Step 3: Under "Bucket Type," select **General Purpose** because creating a bucket sets up a new storage container with a unique name and configurations. If you select **Directory**, it will organize files within an existing bucket, making it easier to manage and retrieve specific files, but it won't create a new storage container.



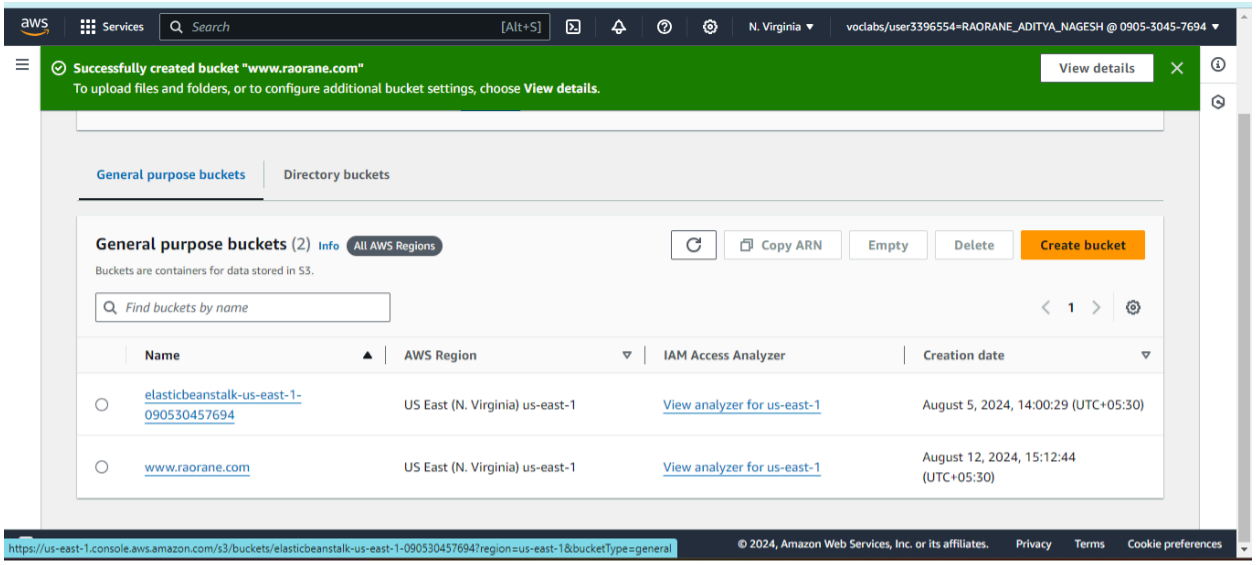
Step 4: Under "Object Ownership," select ACLs disabled to ensure that all objects are owned by the bucket owner, providing simplified permissions management and improved security.

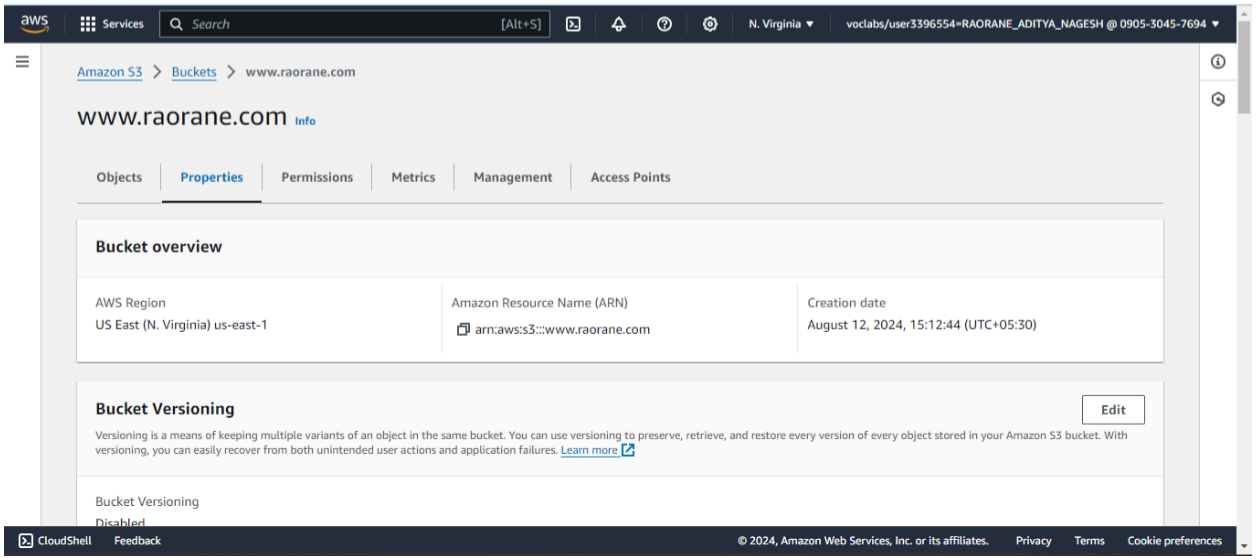


Step 5: Under "Object Lock," I chose **Disable** to allow unrestricted deletion and modification of objects within the bucket.And finally click on “Create Bucket”.

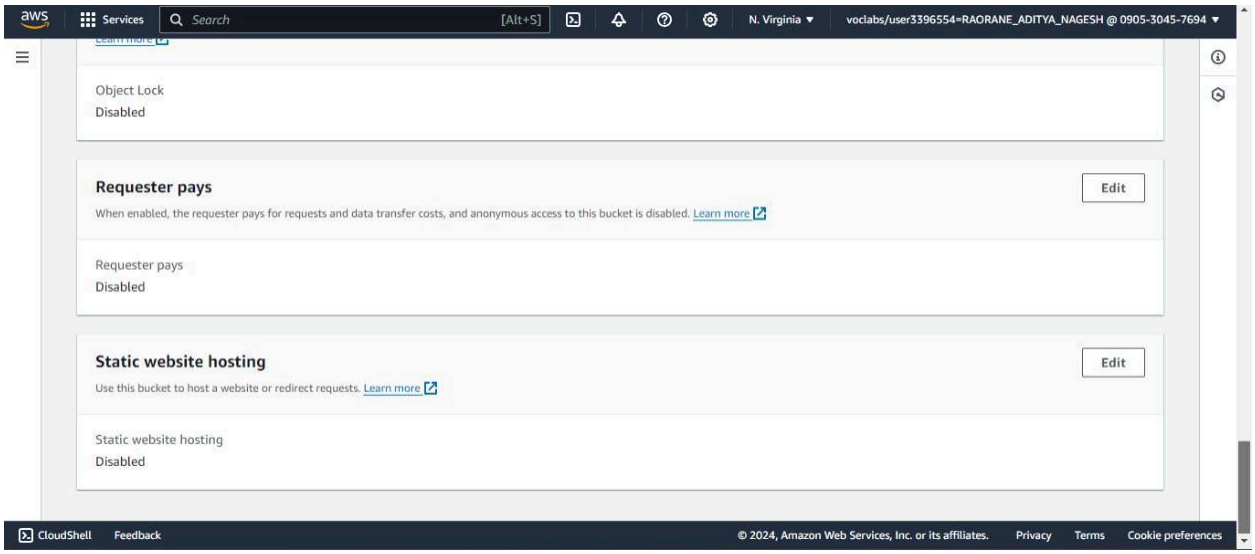


Step 6: Finally, once the bucket is created, click on the bucket link to view its properties and configuration details.

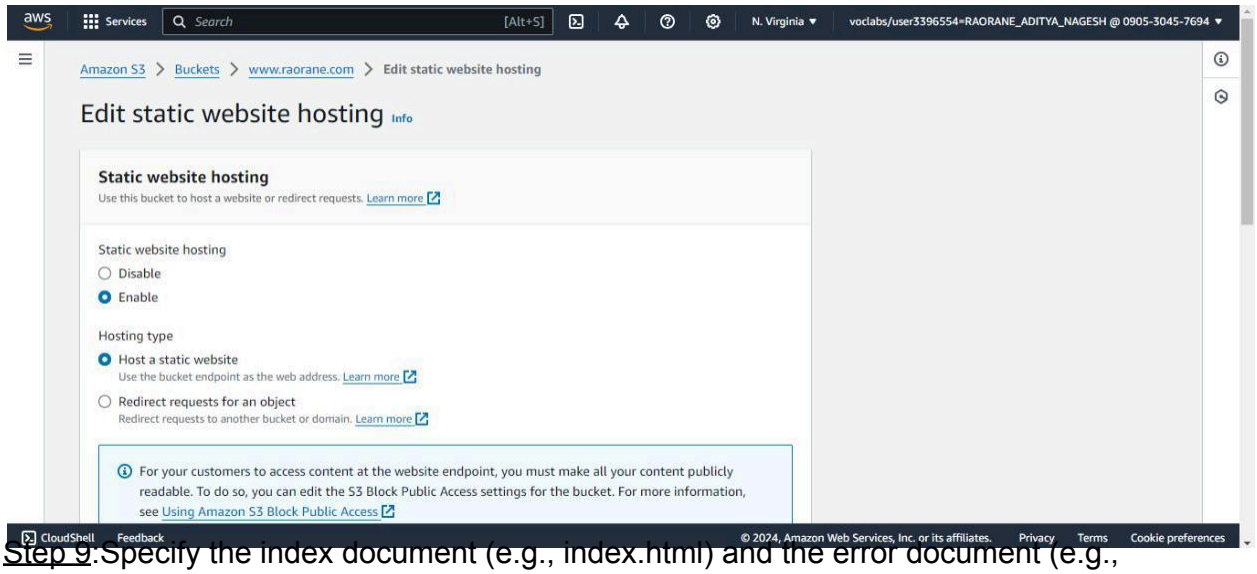




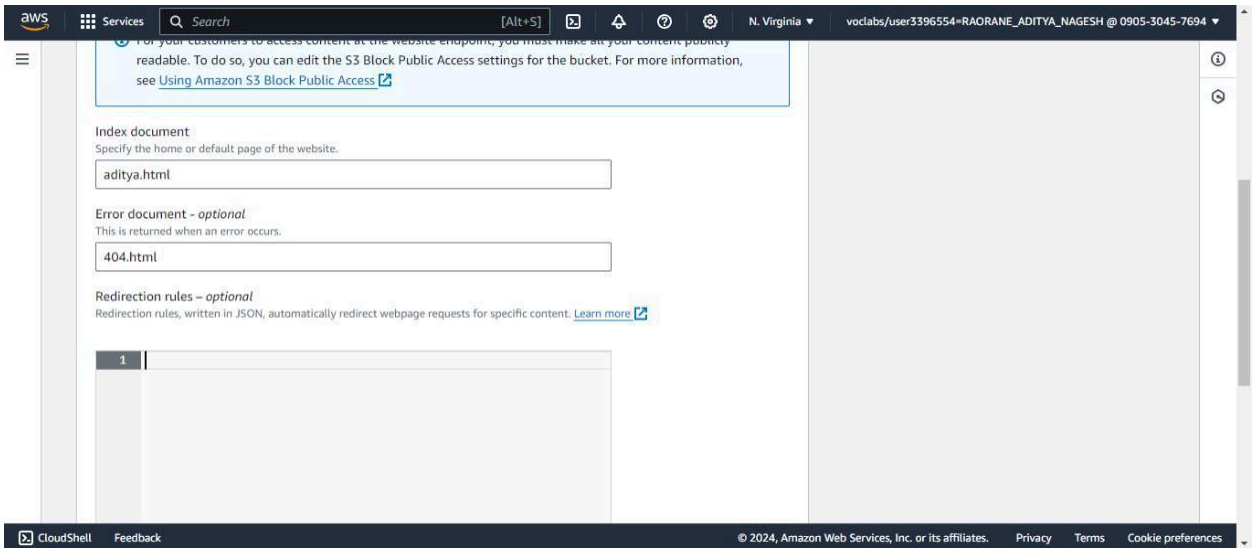
Step 7: Search for Static Website Hosting; it is disabled by default but needs to be enabled to serve static web content from your bucket. Enabling Static Website Hosting allows your S3 bucket to serve static content, such as HTML files. You must specify an index document (e.g., index.html) and optionally an error document. This configuration makes your bucket accessible via a web URL, hosting your static website directly



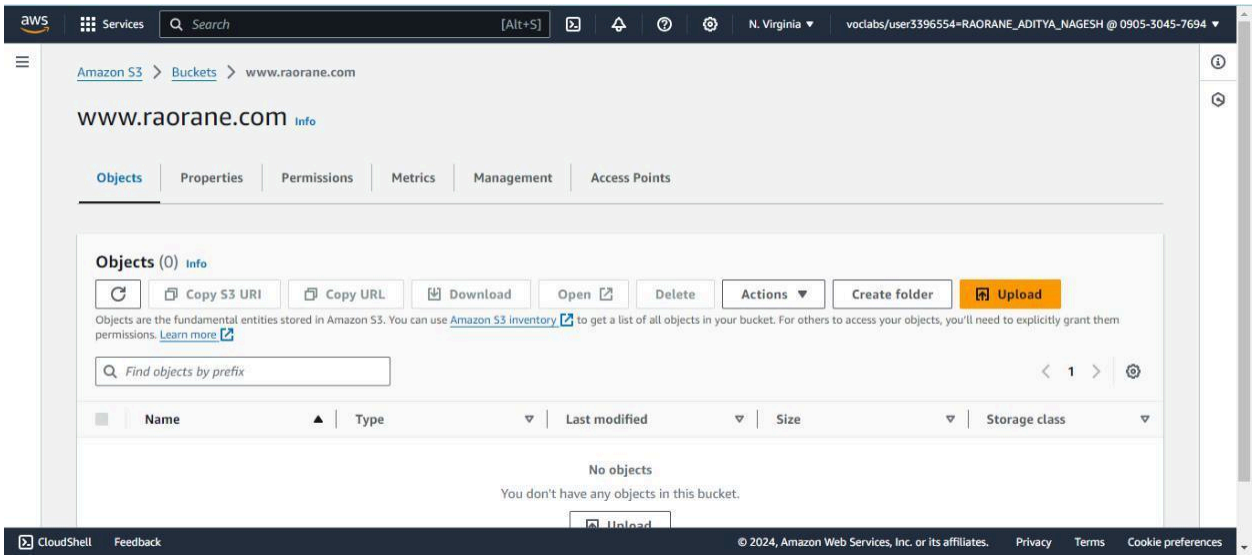
Step 8: **Enable** static website hosting option and select host type as a “static website”



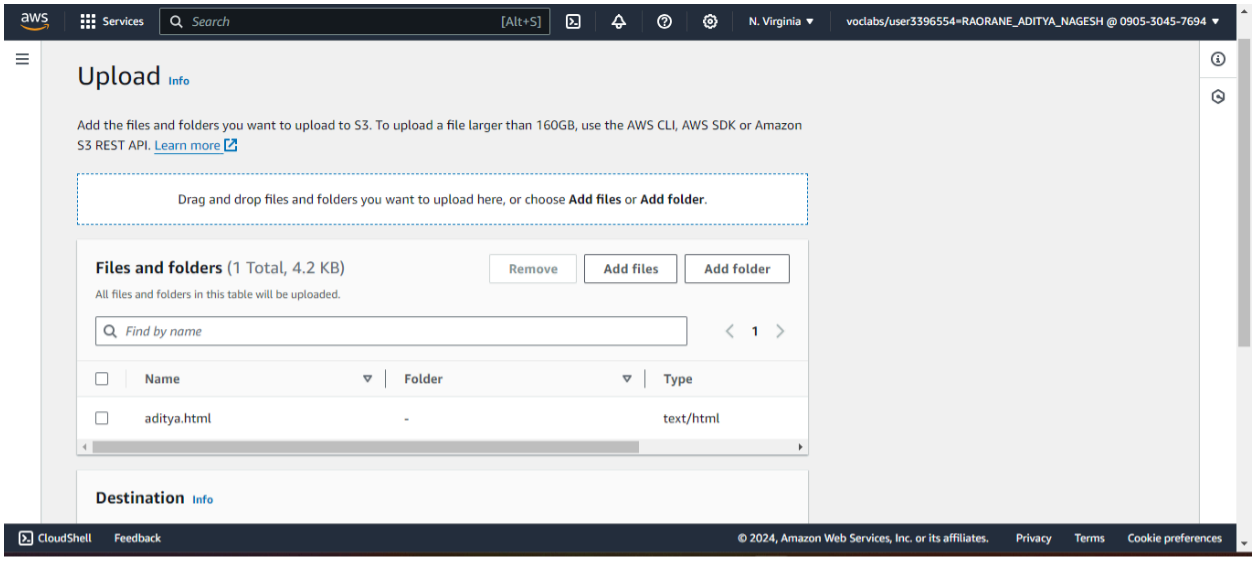
Step 9:Specify the index document (e.g., index.html) and the error document (e.g., error.html) along with the desired HTTP error code to display custom error messages.

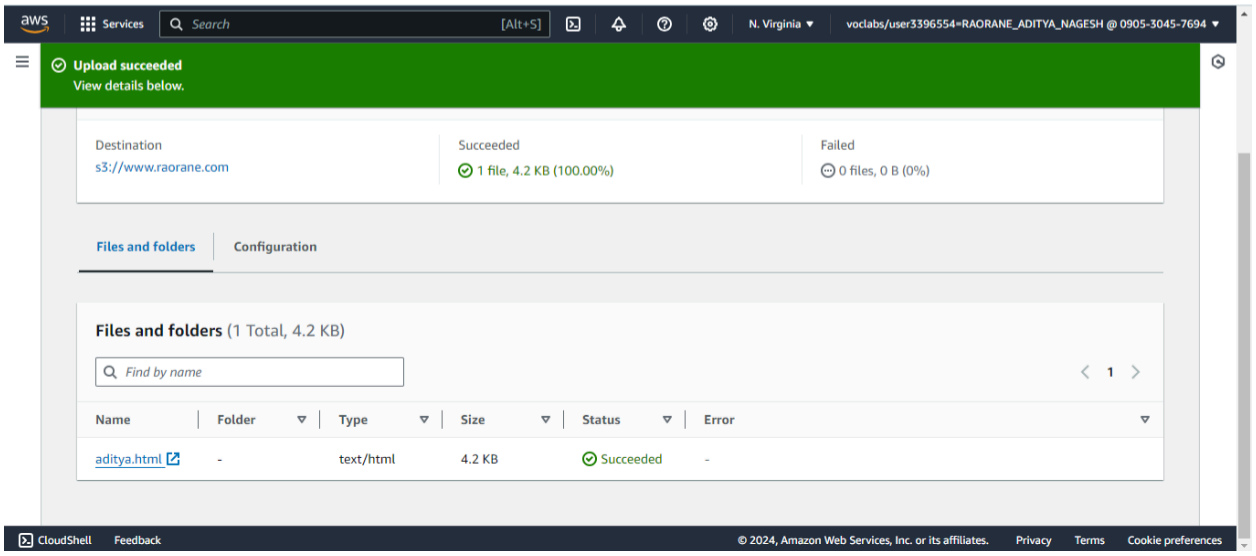


Step 10:Under the “Objects” section, you will see the page to upload your HTML files. Click on the “Upload” button.

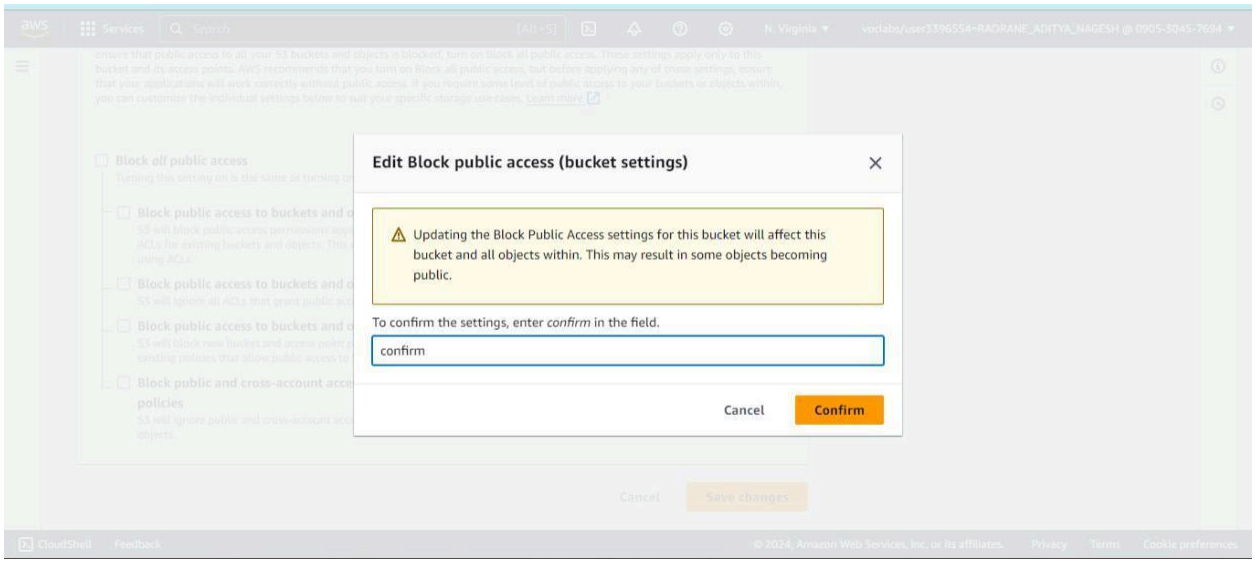


Step 11:Add your file(s) and click the **Upload** button. You will see a confirmation indicating that the file was successfully uploaded.



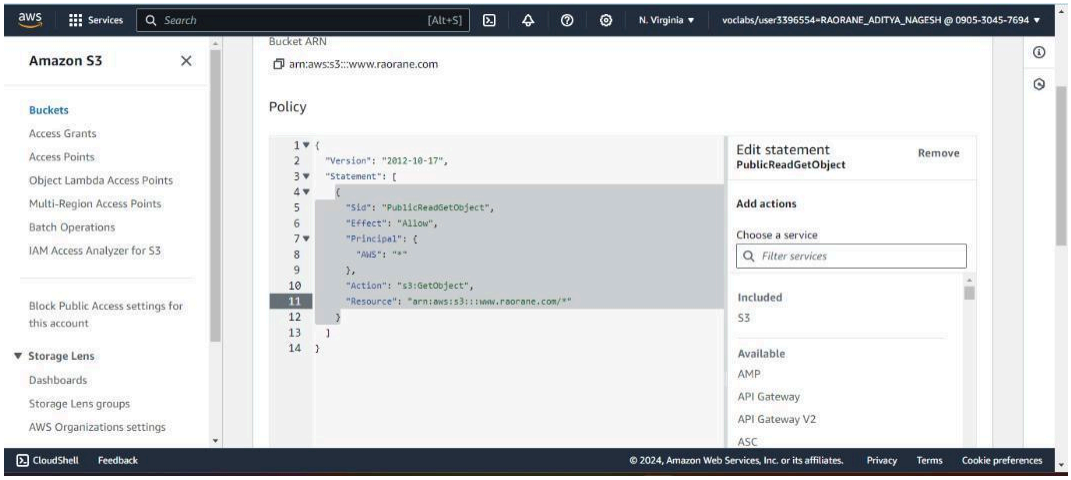


Step 12:Now, if you click on the link, it will show an error 403 Forbidden message due to block policies. To resolve this, you need to configure the bucket's public access settings by unchecking all block public access options.

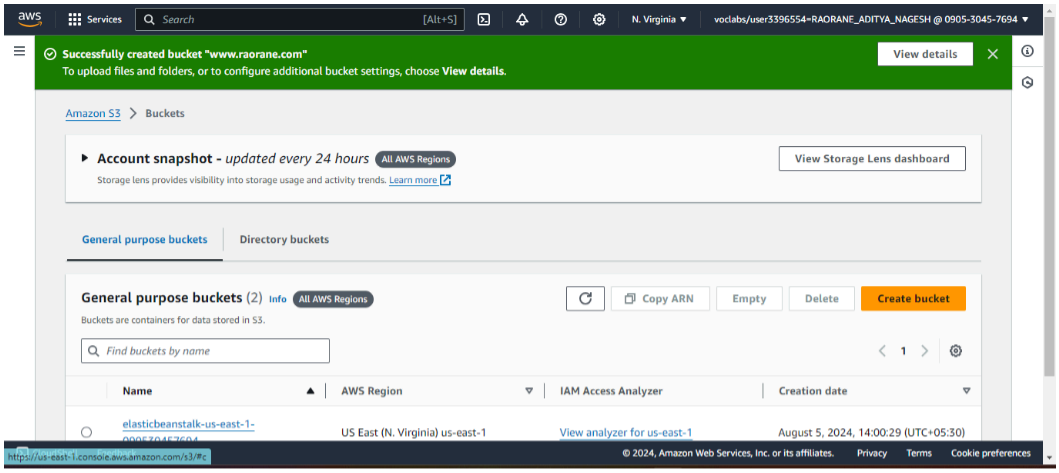


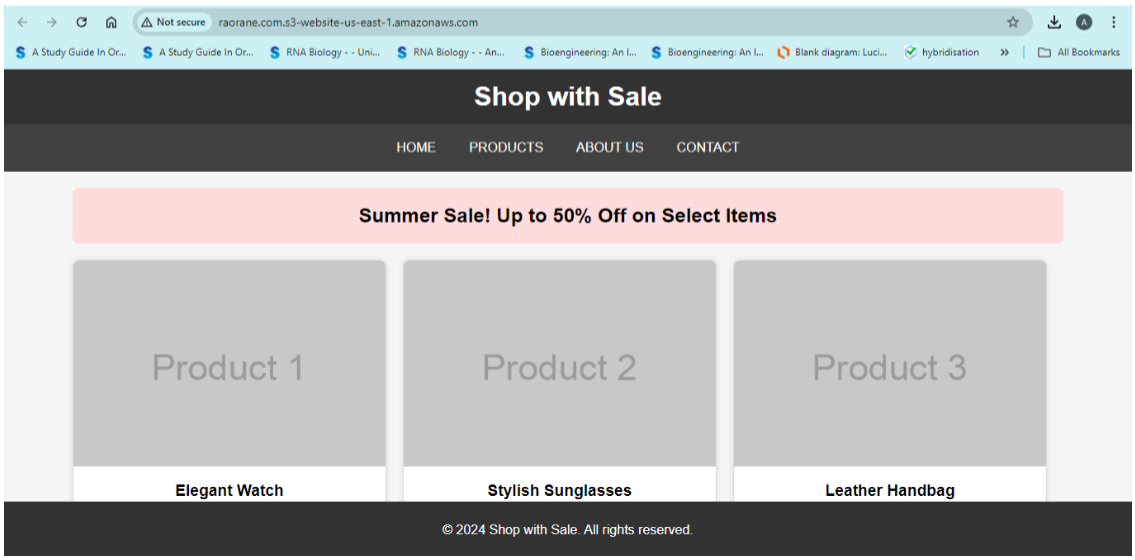
Step 13:Scroll down to the **Bucket Policy** section and paste the policy from the following link:
https://gist.github.com/Savjee/b4b3a21d143a30e7dc07
To configure access permissions:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::www.raorane.com"
    }
  ]
}
```



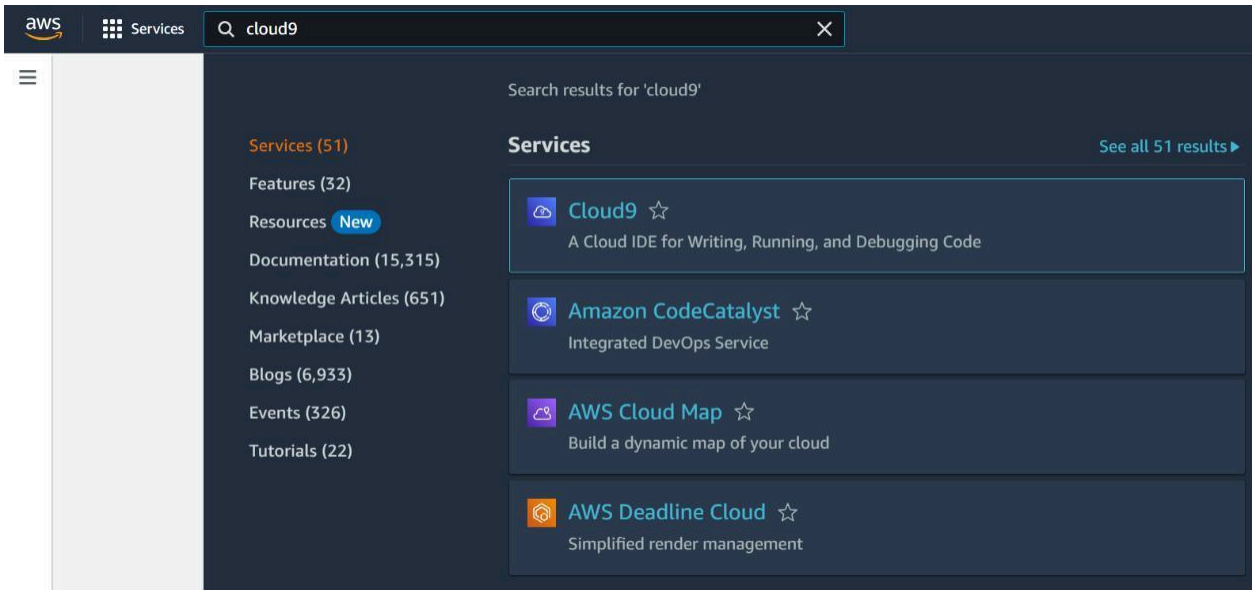
Step 14: Finally, you will see a confirmation about the policies. The website is now ready to run—click on the provided link to view the live website.



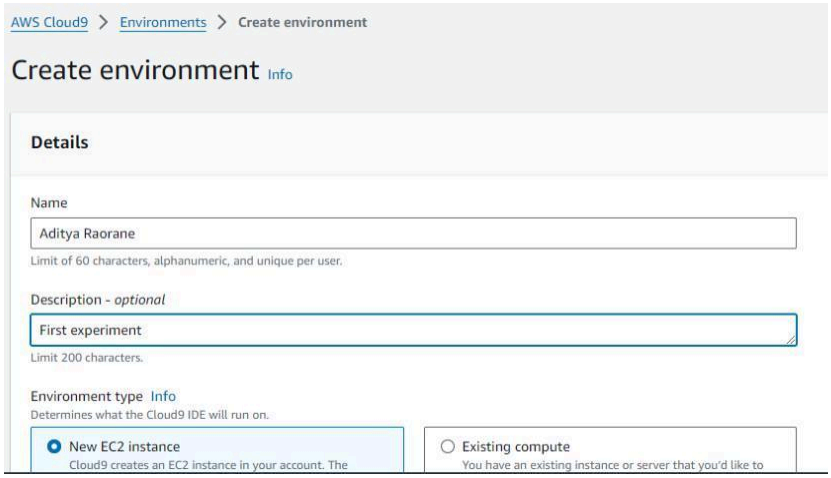


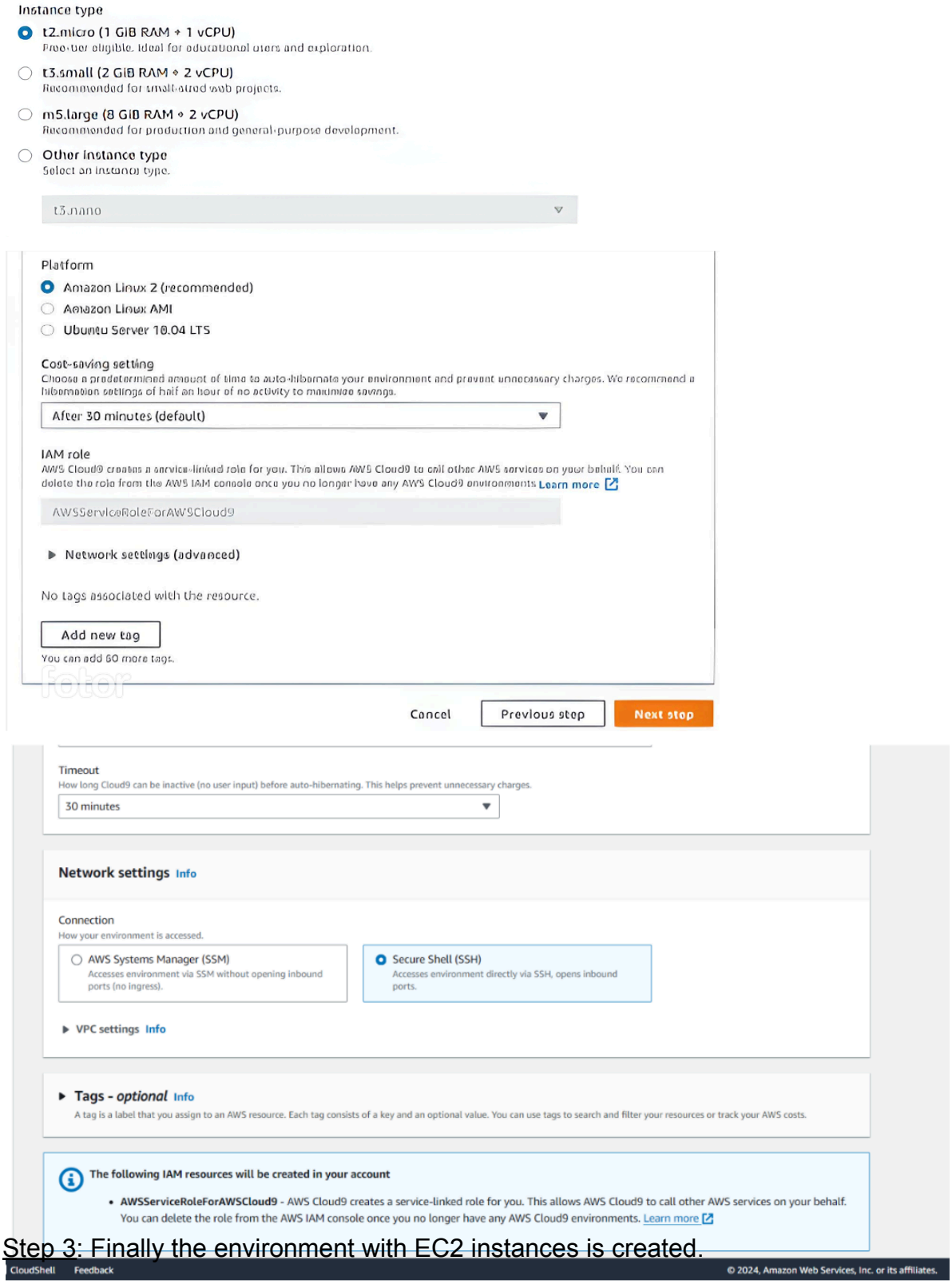
1b]Cloud9 IDE Collaborative Setup

Step 1:In your AWS Academy account, navigate to the "Services" search bar, type "Cloud9" and open it to access the Cloud9 IDE.

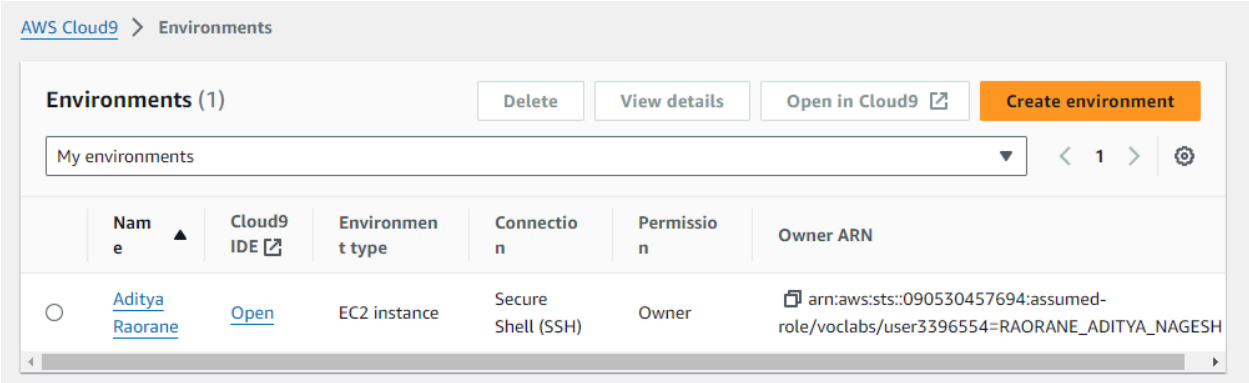


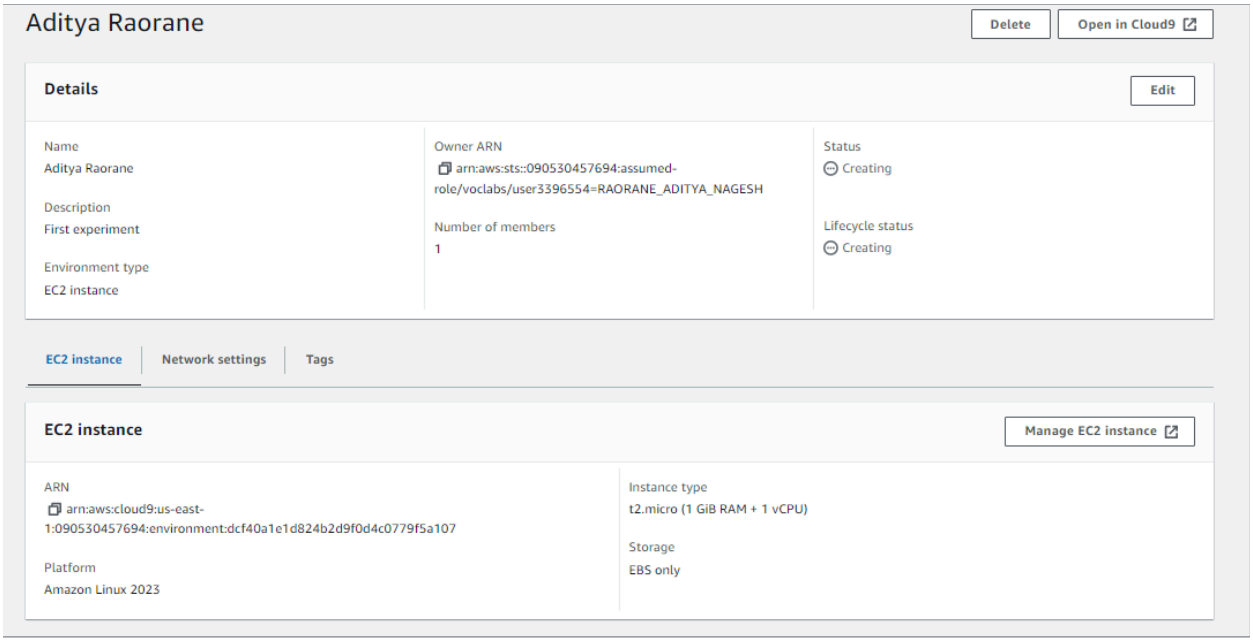
Step 2:To set up an environment in AWS Cloud9, create a **new Cloud9 environment** in the AWS Management Console, choosing your preferred instance type and VPC settings.Select “EC2 instance” in the environment type.



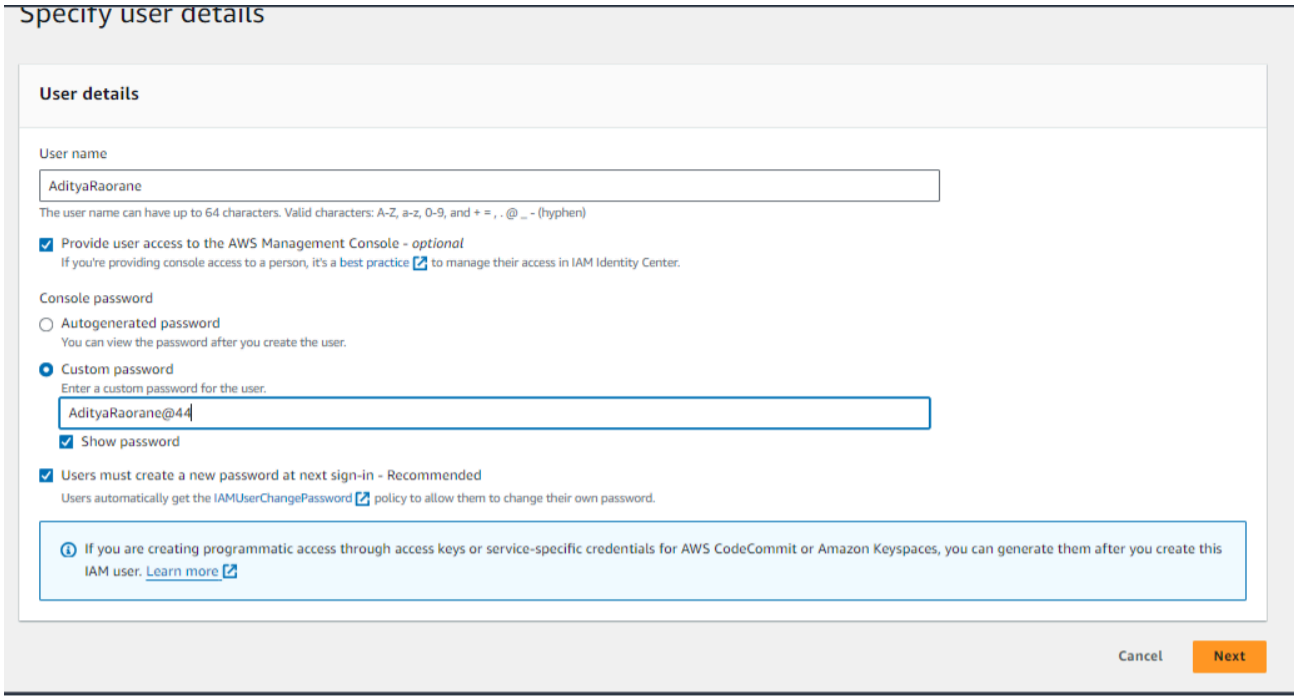


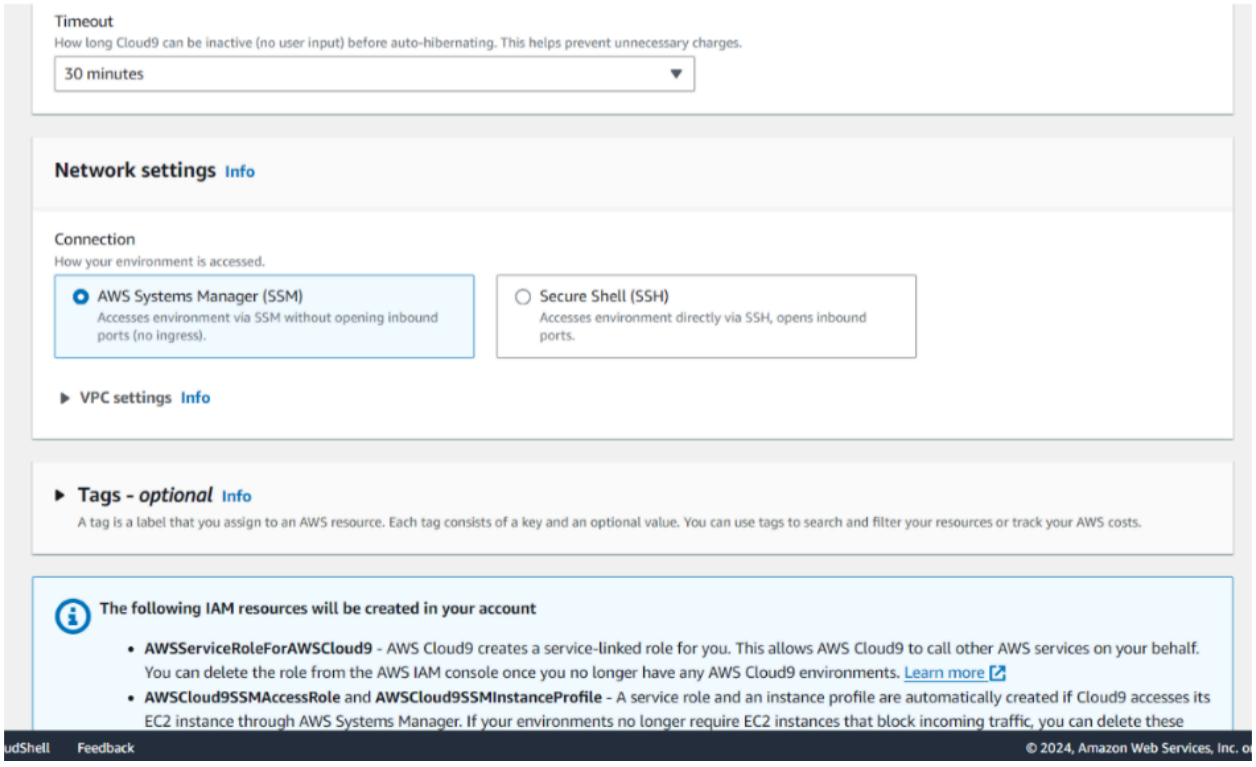
Step 3: Finally the environment with EC2 instances is created.

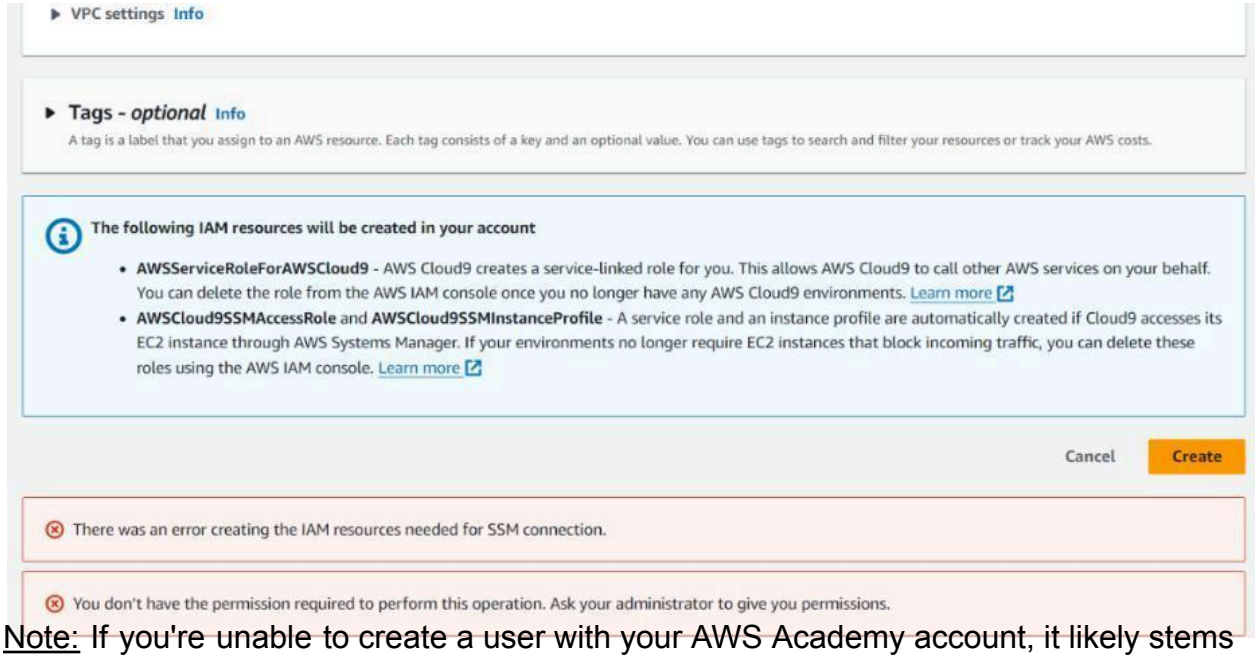




Step 4:Configure a username and password for a user and assign the role.

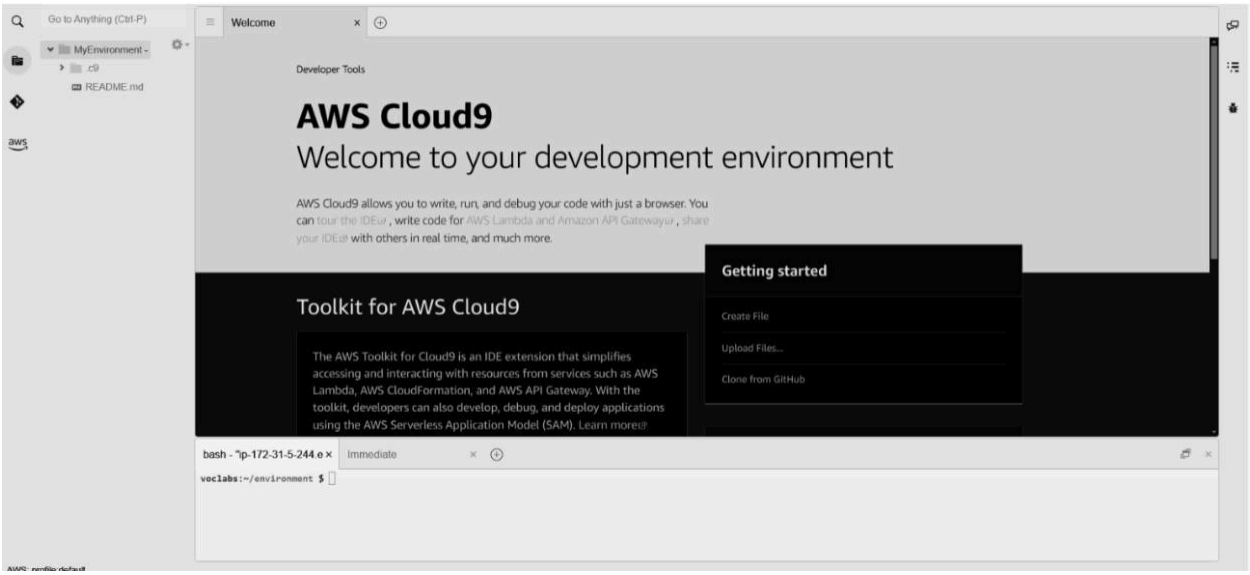

Specify user details
User details
User name
AdityaRaorane
The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)
Provide user access to the AWS Management Console - optional
If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.
Console password
Autogenerated password
You can view the password after you create the user.
Custom password
Enter a custom password for the user.
AdityaRaorane@44
Show password
Users must create a new password at next sign-in - Recommended
Users automatically get the IAMUserChangePassword policy to allow them to change their own password.
If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. Learn more
Cancel
Next


Timeout
How long Cloud9 can be inactive (no user input) before auto-hibernating. This helps prevent unnecessary charges.
30 minutes
Network settings Info
Connection
How your environment is accessed.
AWS Systems Manager (SSM)
Accesses environment via SSM without opening inbound ports (no ingress).
Secure Shell (SSH)
Accesses environment directly via SSH, opens inbound ports.
VPC settings Info
Tags - optional Info
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.
The following IAM resources will be created in your account
AWSServiceRoleForAWSCloud9 - AWS Cloud9 creates a service-linked role for you. This allows AWS Cloud9 to call other AWS services on your behalf. You can delete the role from the AWS IAM console once you no longer have any AWS Cloud9 environments. Learn more
AWSCloud9SSMAccessRole and AWSCloud9SSMInstanceProfile - A service role and an instance profile are automatically created if Cloud9 accesses its EC2 instance through AWS Systems Manager. If your environments no longer require EC2 instances that block incoming traffic, you can delete these
udShell
Feedback
© 2024, Amazon Web Services, Inc. o

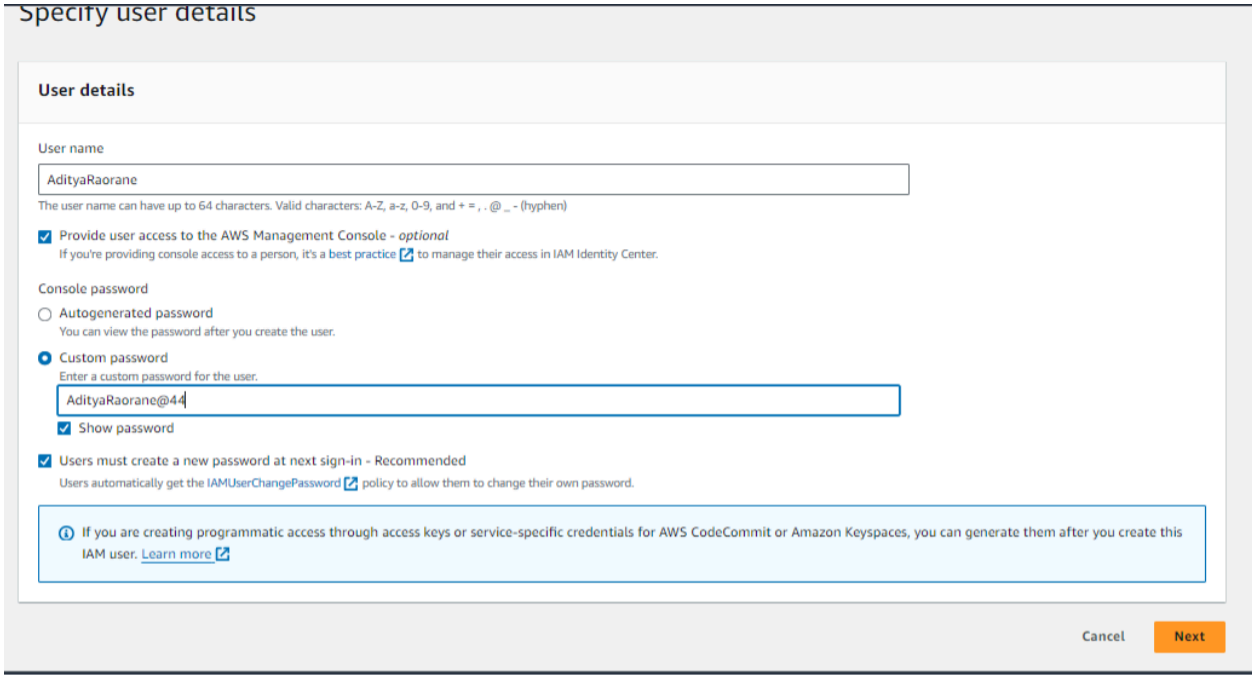


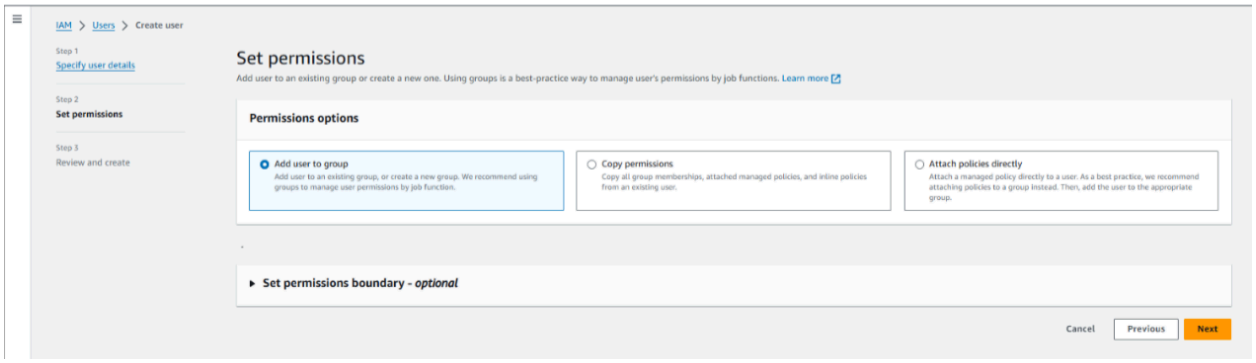
Note: If you're unable to create a user with your AWS Academy account, it likely stems from limited permissions or role restrictions imposed to prevent unauthorized actions or costs. These accounts often have constraints tailored for educational purposes, so you may need to use a personal account or request assistance from your instructor.

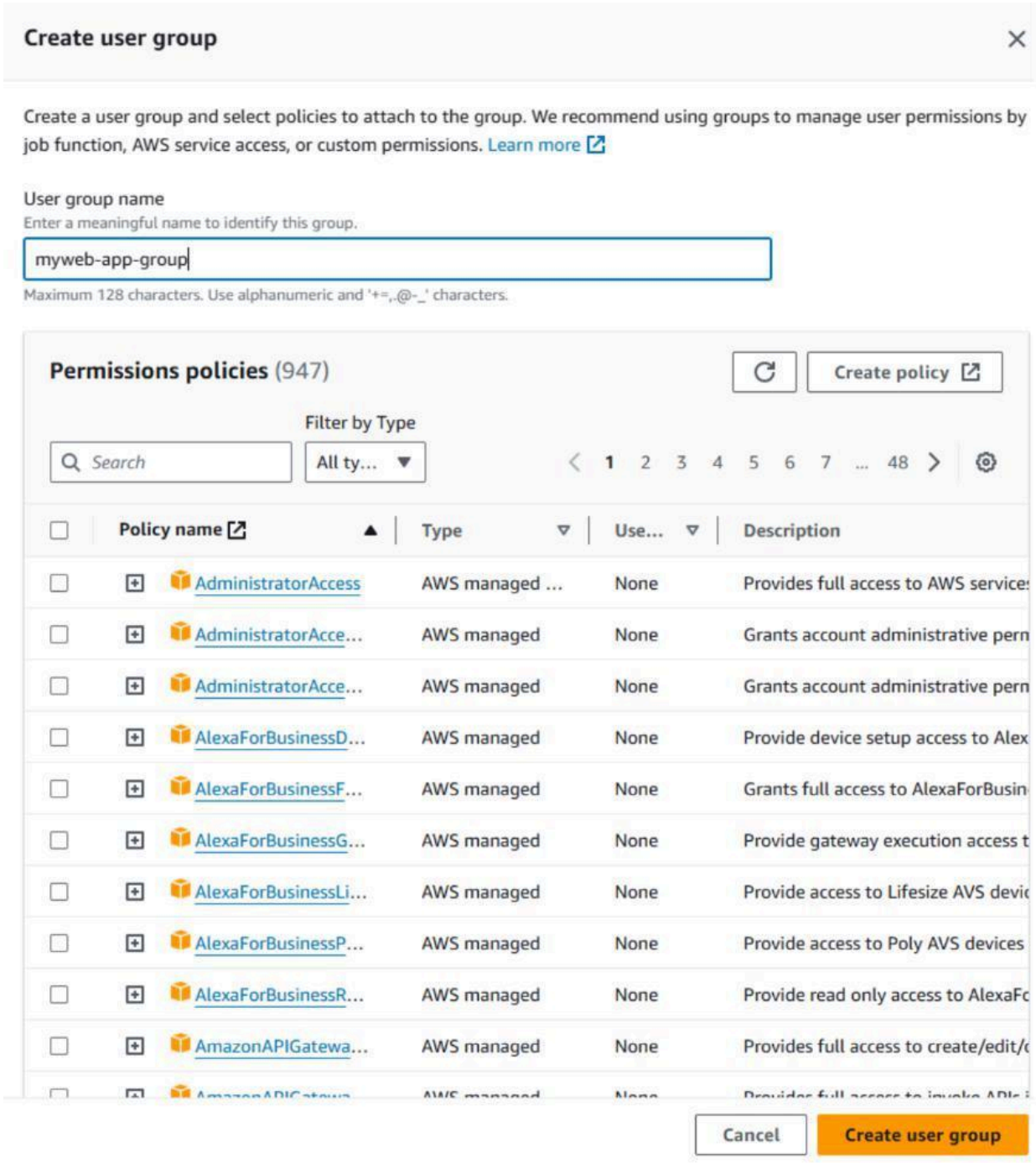
Step 5:Open Cloud9 IDE.



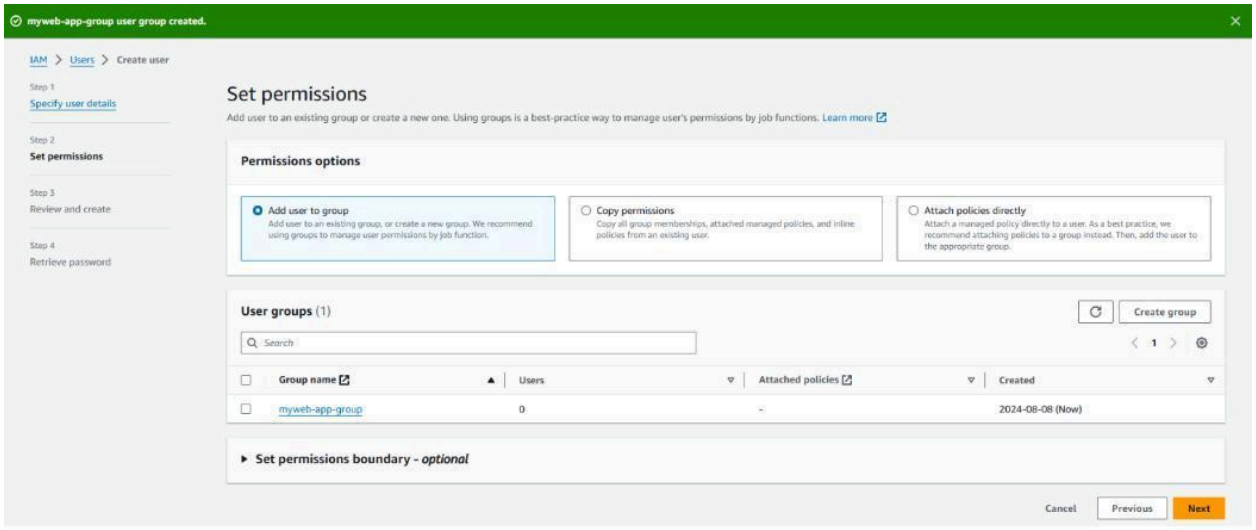
Step 6:Open AWS IAM service.Configure a user and a group.



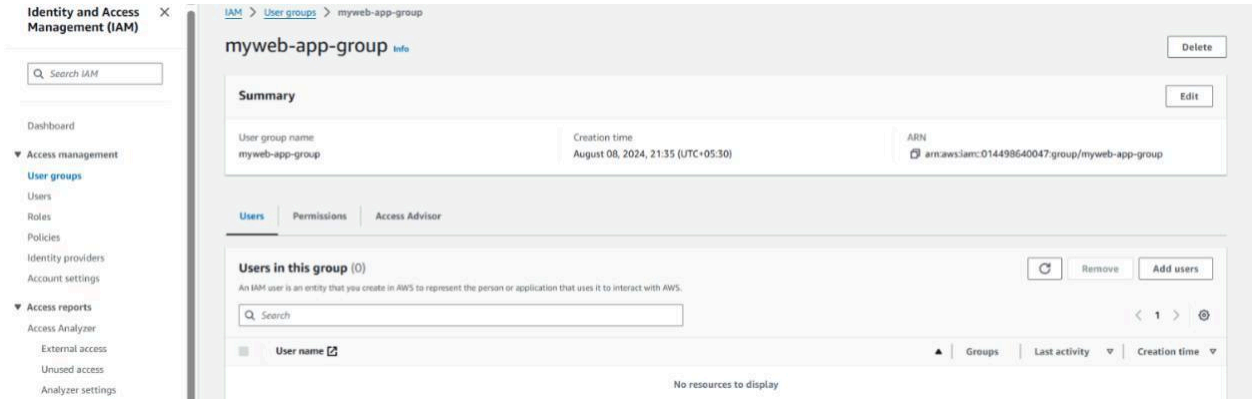



Create user group
Create a user group and select policies to attach to the group. We recommend using groups to manage user permissions by job function, AWS service access, or custom permissions. Learn more
User group name
Enter a meaningful name to identify this group.
myweb-app-group
Maximum 128 characters. Use alphanumeric and '+=,.@-_' characters.
Permissions policies (947)
Create policy
Filter by Type
Search
All ty...
1 2 3 4 5 6 7 ... 48
Policy name
Type
Use...
Description
AdministratorAccess AWS managed ... None Provides full access to AWS service
AdministratorAcce... AWS managed None Grants account administrative perm
AdministratorAcce... AWS managed None Grants account administrative perm
AlexaForBusinessD... AWS managed None Provide device setup access to Alex
AlexaForBusinessF... AWS managed None Grants full access to AlexaForBusin
AlexaForBusinessG... AWS managed None Provide gateway execution access t
AlexaForBusinessLi... AWS managed None Provide access to Lifesize AVS devi
AlexaForBusinessP... AWS managed None Provide access to Poly AVS devices
AlexaForBusinessR... AWS managed None Provide read only access to AlexaFo
AmazonAPIGatewa... AWS managed None Provides full access to create/edit/
Cancel
Create user group

Step 7: Once the user group is created click on the server link created next to the user.

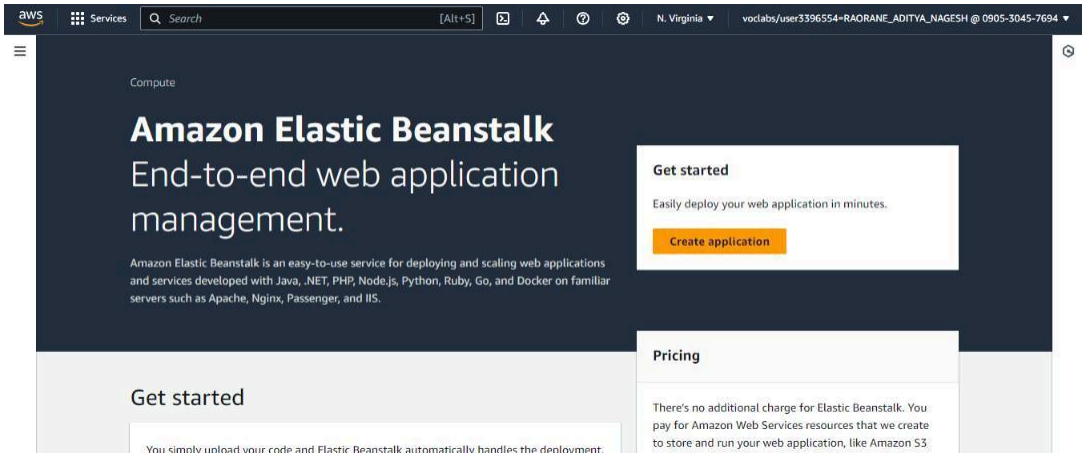


Step 8: Finally search for “AWSCloud9EnvironmentMember” policy and attach it.

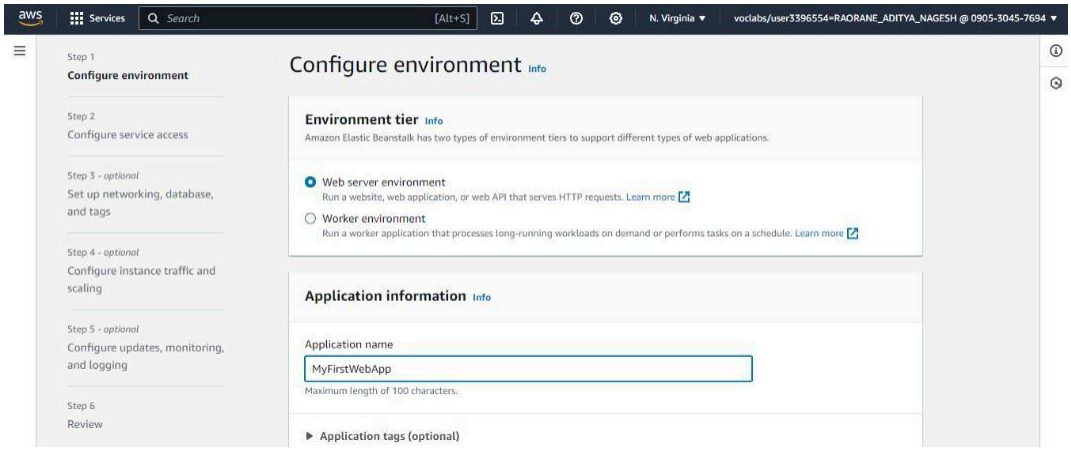


Expt 2]Application Deployment with AWS Elastic Beanstalk and AWS CodePipeline

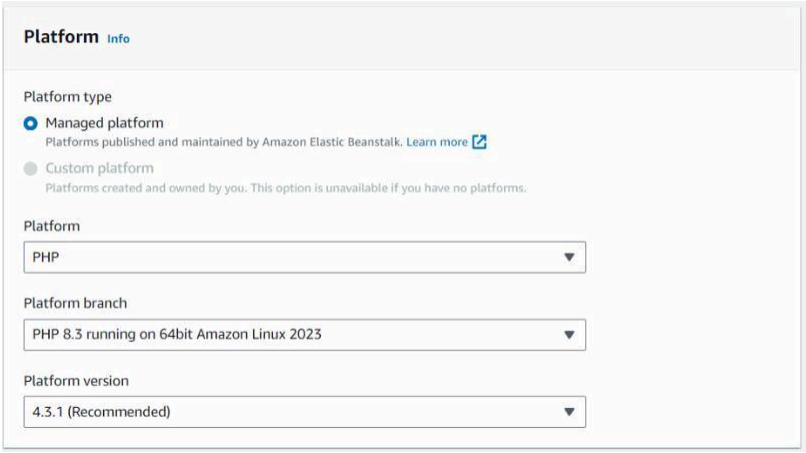
Step 1:To open AWS Elastic Beanstalk, log in to the AWS Management Console, navigate to Elastic Beanstalk under **"Compute**," and click "**Create Application**" to start a new project



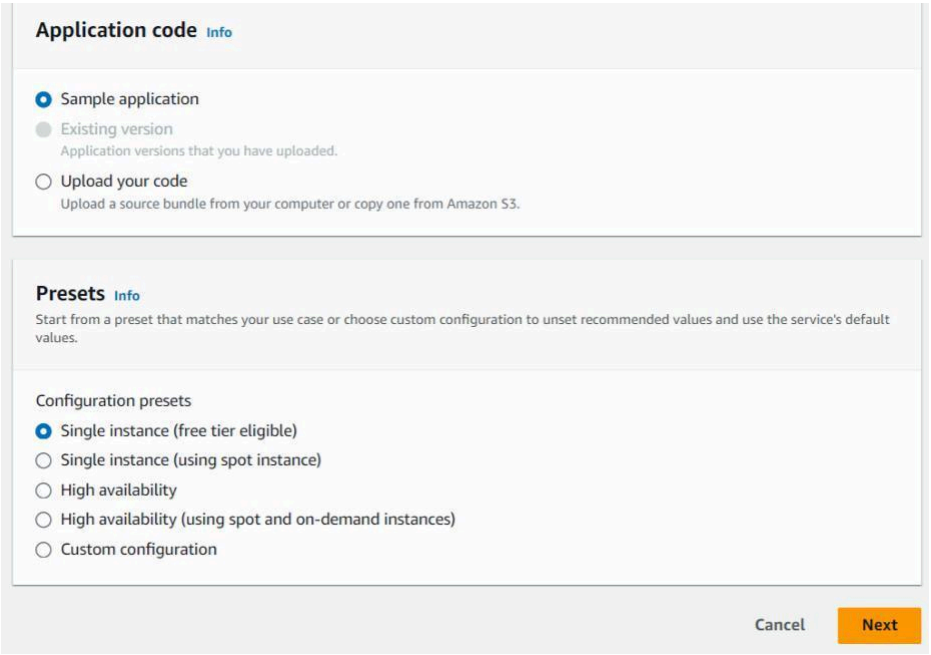
2]After clicking "Create Application" in Elastic Beanstalk, you'll choose between two environment tiers:
Web Server Environment: Handles HTTP requests for web applications with built-in load balancing, auto-scaling, and a pre-configured web server.
Worker Environment: Manages background tasks asynchronously, processing jobs from an SQS queue without direct user interaction.
Click on "Web server Environment".



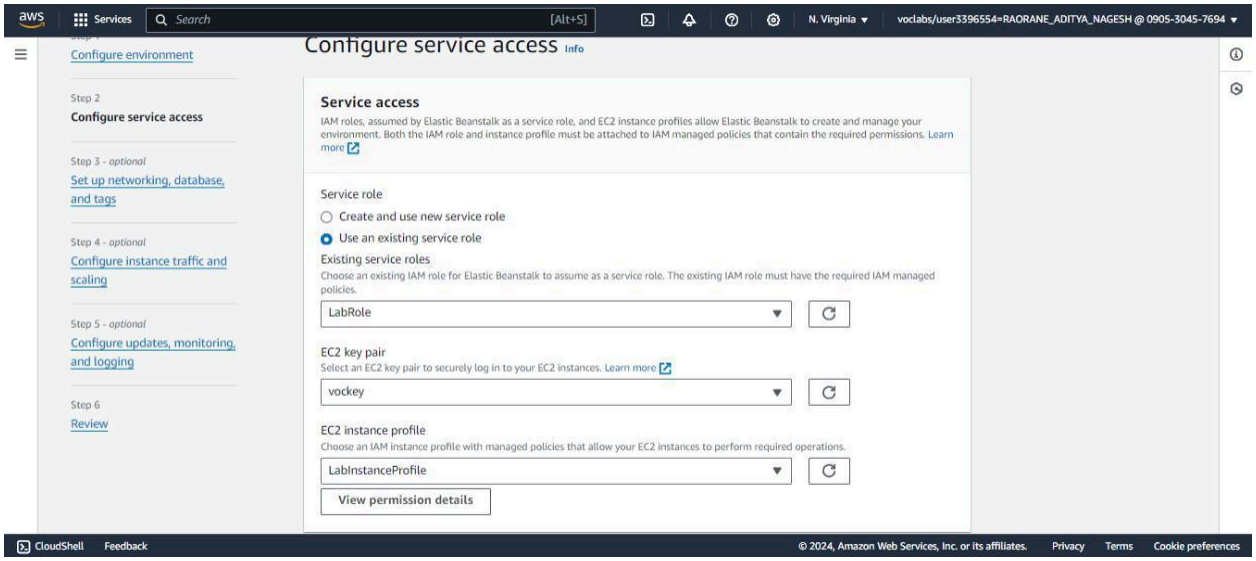
Step 3:Under Platform, you can choose between Managed Platform and Custom Platform. Selected Managed Platform and configured it with PHP for ease of use and automatic updates. **Managed Platform**: AWS manages and updates the environment, providing pre-configured platforms like PHP, making deployment easier and maintenance automated.
Custom Platform: Allows you to create and manage your own platform, giving full control over the environment configuration and software stack.



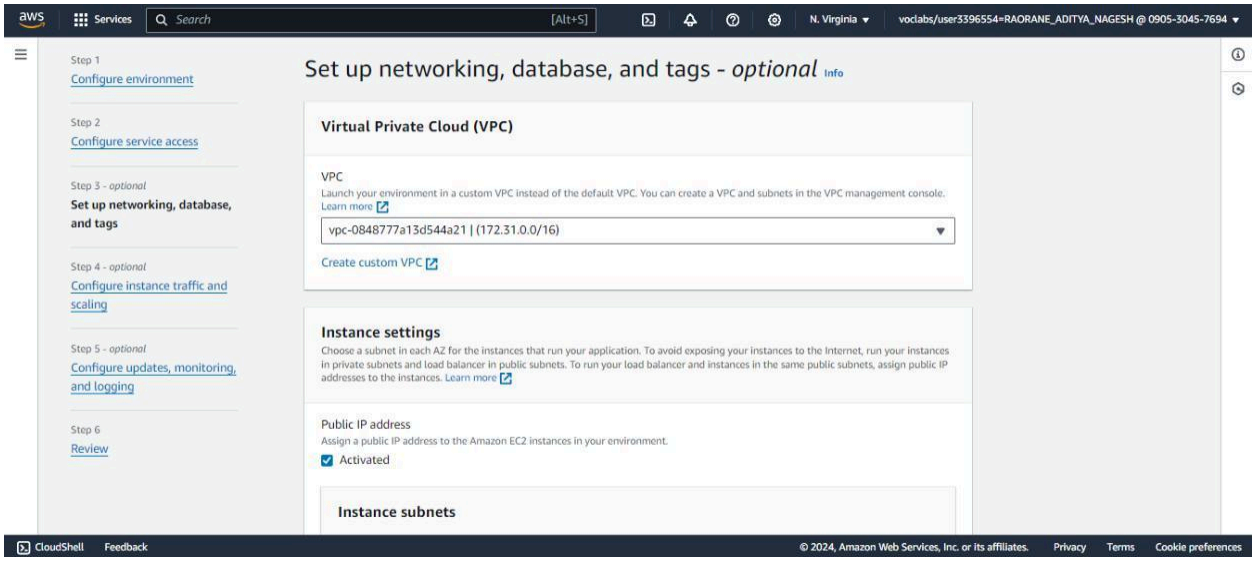
Step 4:**Presets** offer different deployment configurations to match your needs.
Single Instance (Free Tier Eligible) runs your application on one EC2 instance, suitable for development or low-traffic scenarios and eligible for the AWS Free Tier.
Single Instance deploys on one EC2 instance but without Free Tier limitations, ideal for small-scale projects.
High Availability uses multiple instances and a load balancer to ensure fault tolerance and reliability for production environments.
Using Spot Instances leverages unused EC2 capacity at reduced costs, suitable for applications that can handle interruptions and require cost savings.



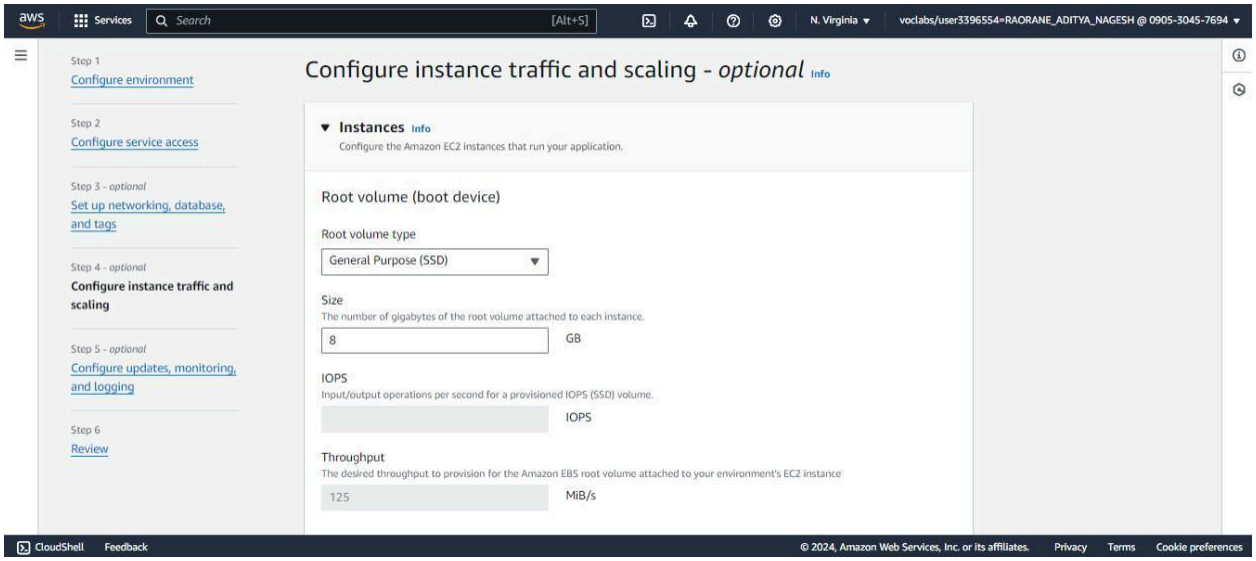
Step 5:Define your application's service configuration, including settings for load balancing, auto-scaling, and health checks.Each EC2 instance can be accessed using its corresponding key pair.Choosing an EC2 key pair as a "vockey" (or a "virtual key") is typically done to enable secure access to your EC2 instances; allowing you to securely connect to your instances using SSH (Linux), ensuring that only authorized users can access the server.



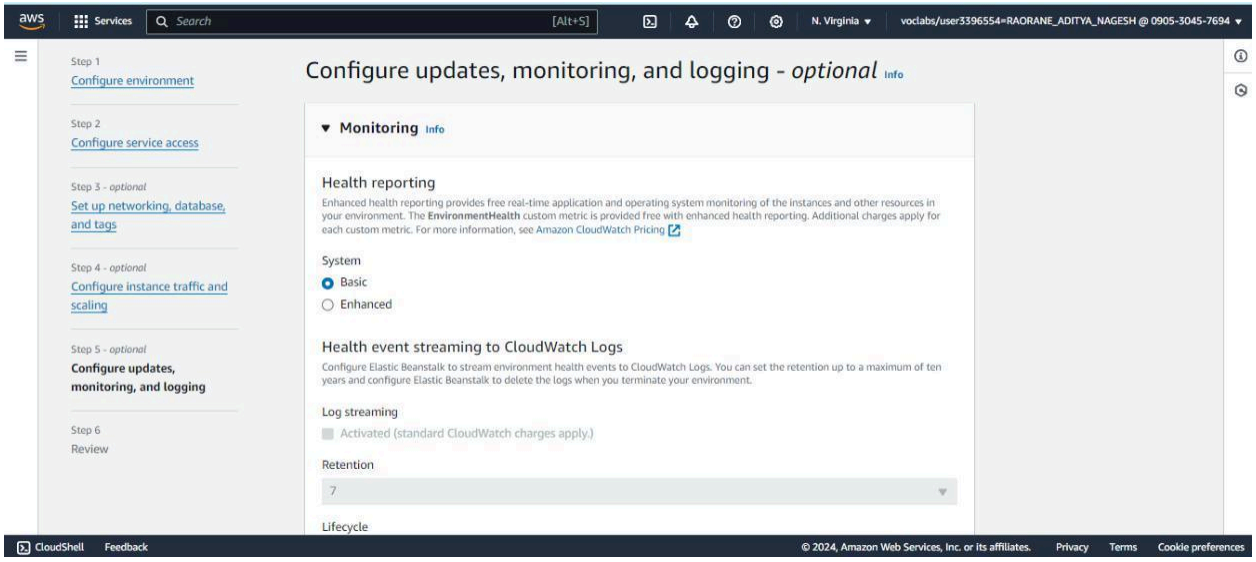
Step 6: Configure VPC, subnets, and security groups for network access. Set up your environment's database if needed and apply tags for resource management.



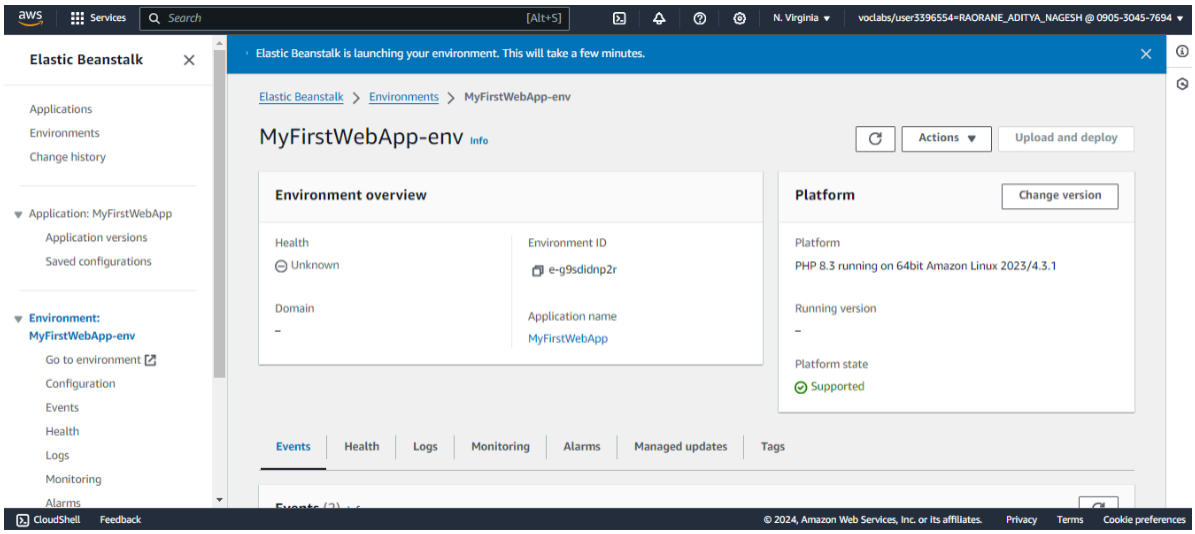
Step 7:Adjust load balancer settings to manage incoming traffic and configure auto-scaling rules to handle changes in traffic volume.The **root volume type** is the primary storage for the OS, and selecting **General Purpose (SSD)** offers balanced performance and durability for most applications.

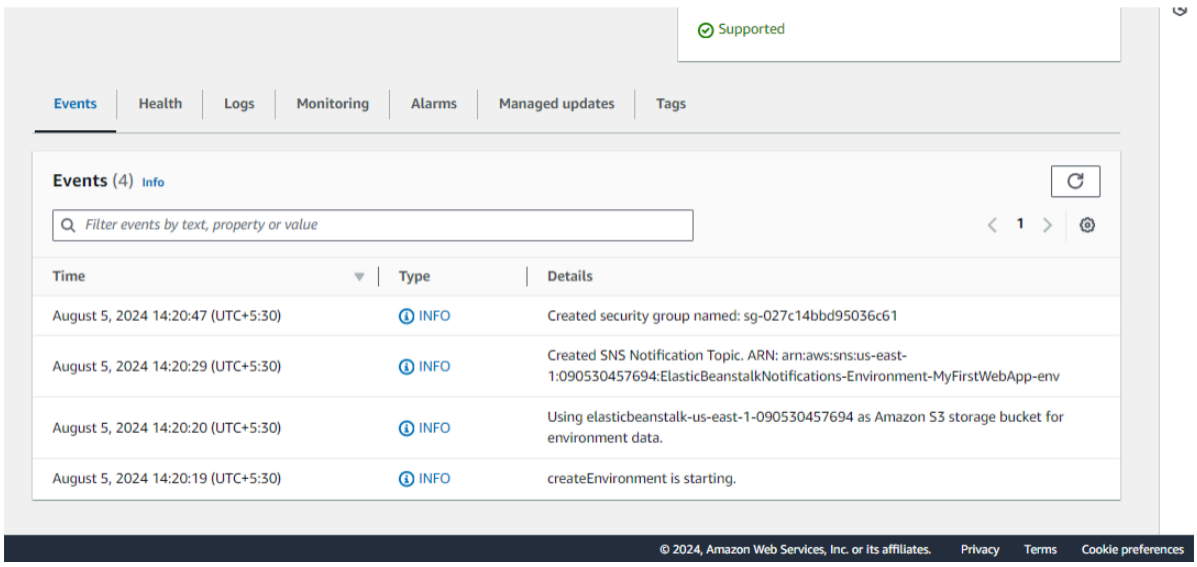


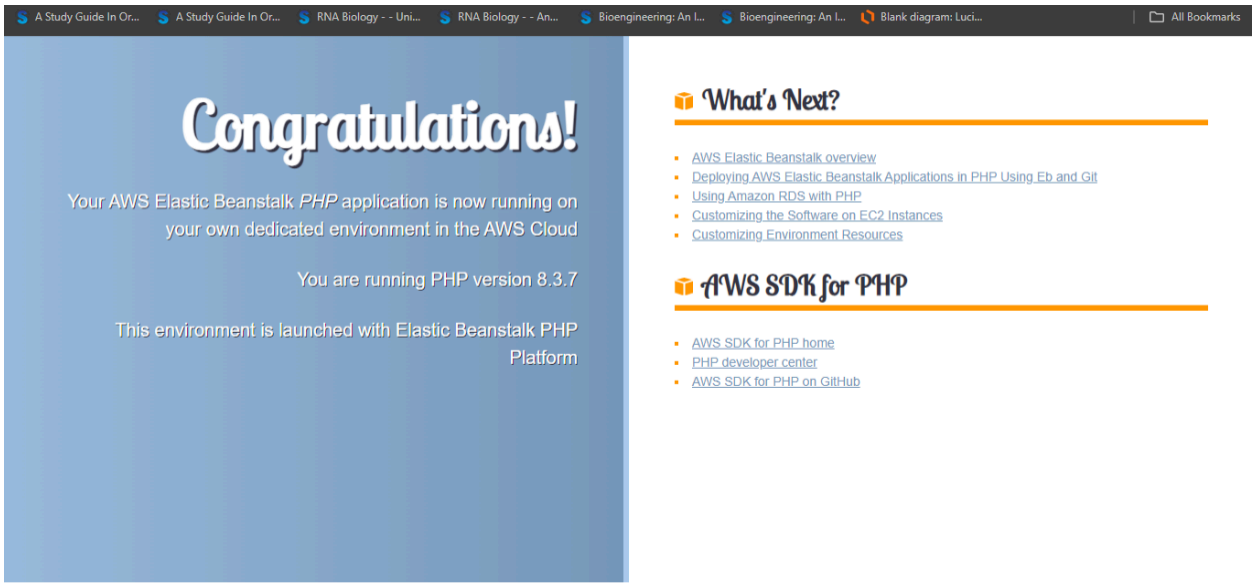
Step 8:Set up deployment policies for updates, enable monitoring for performance metrics, and configure logging to capture and analyze application logs.

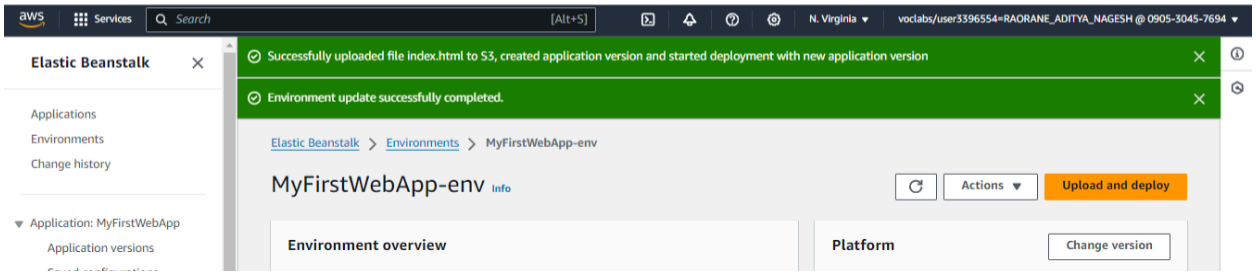


Step 9:This creates our environment .We can now check our sample application uploaded if our environment is successfully deployed.

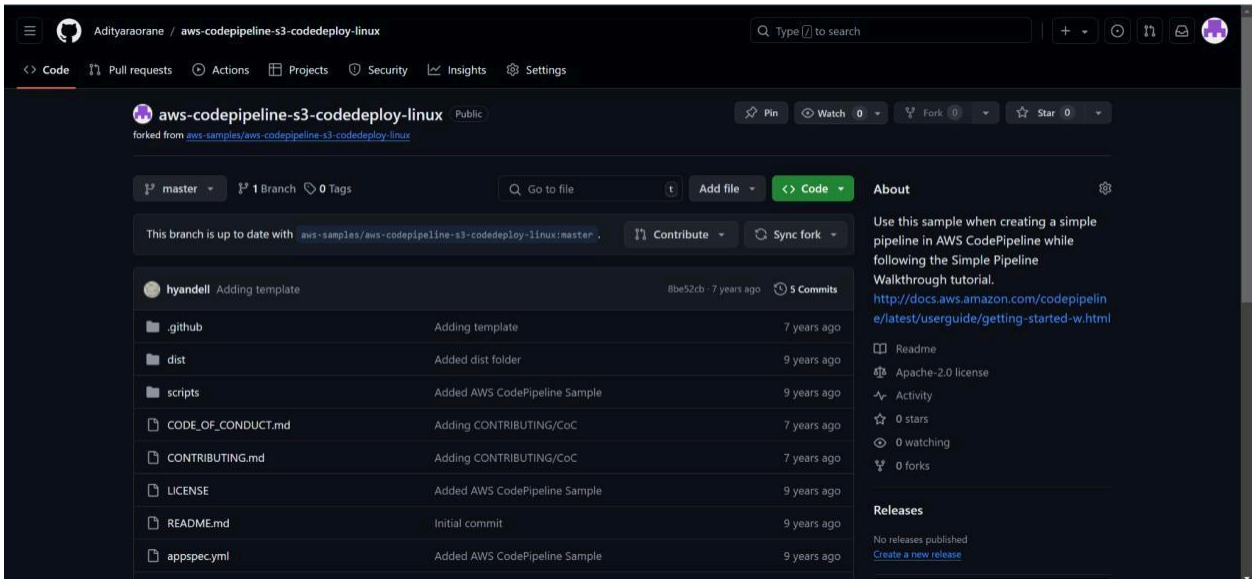




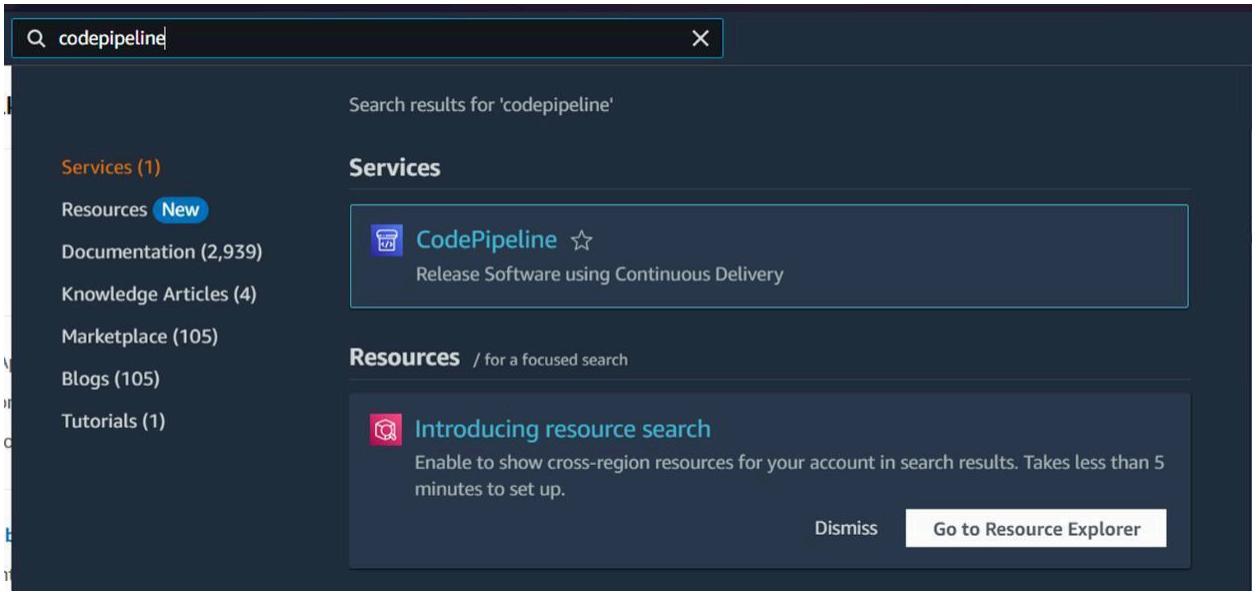




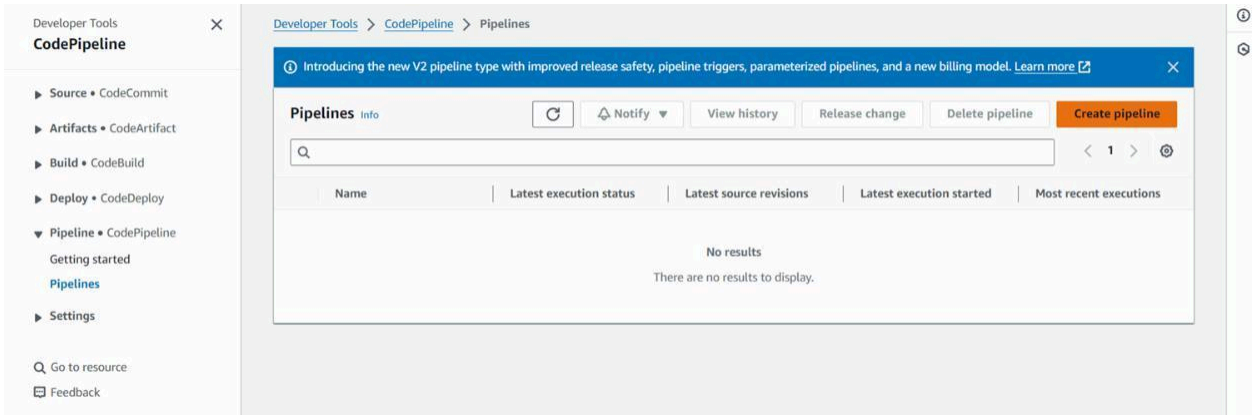
Step 10: Go to the github link below. This is a github with a sample code for deploying a file on AWSCodePipeline. Fork this repository into your personal github.
https://github.com/aws-samples/aws-codepipeline-s3-codedeploy-linux



Step 11:Search **CodePipeline** in the “services” tab and click on it.



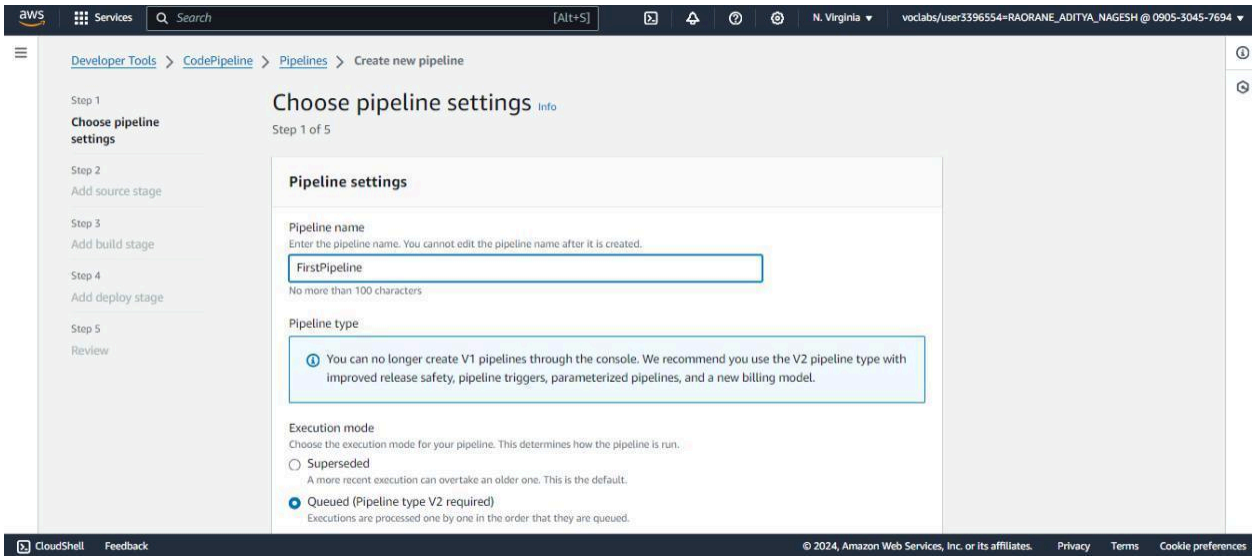
Step 12: After navigation to CodePipeline under “Developer Tools,” and click “Create pipeline” to start setting up your CI/CD.



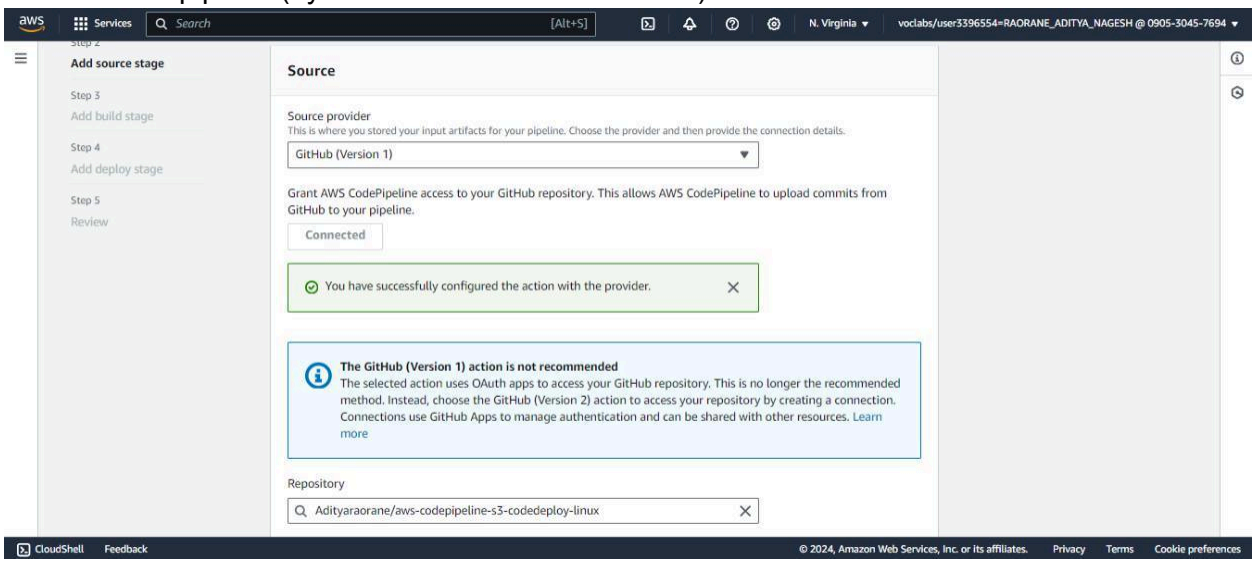
Step 13: Give a name to your Pipeline. A new service role would be created with the name of the pipeline. The **pipeline type** options are:

- **Suspended**: Pauses the pipeline, stopping all execution of actions and deployments until resumed.
- **Queued**: Holds pipeline executions in a queue, executing them sequentially. This is useful for managing limited resources and preventing overload.
- **Parallel**: Runs multiple pipeline executions concurrently, allowing for faster processing but potentially higher resource usage.

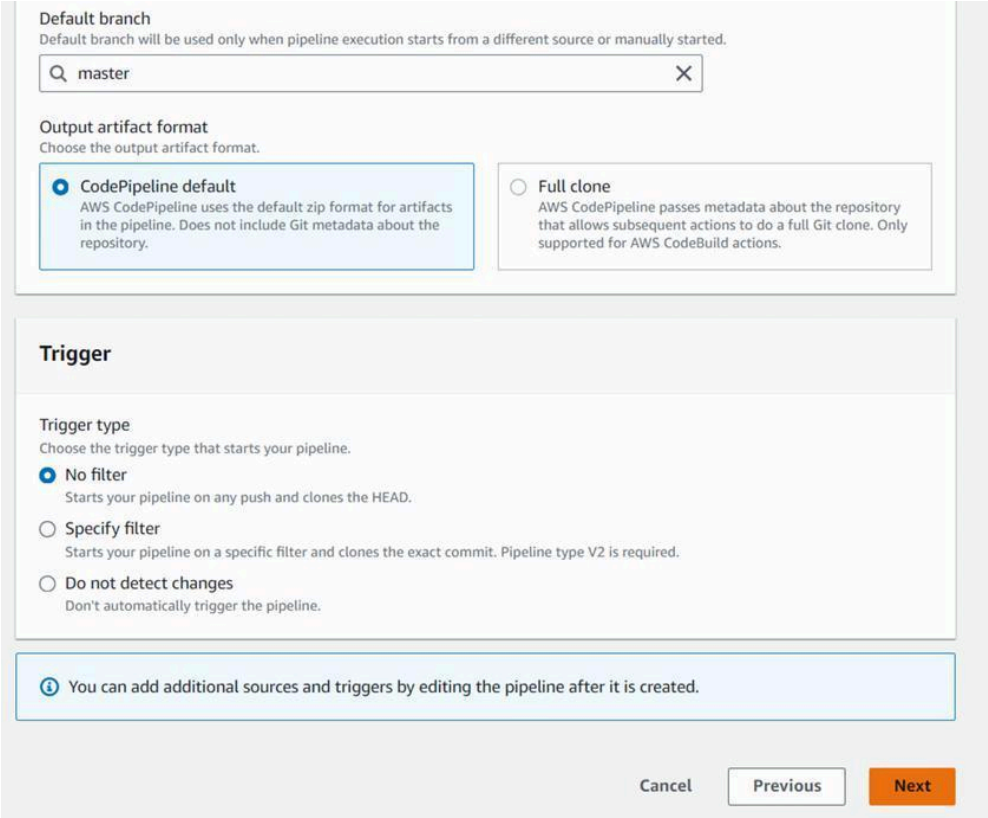
Selecting **Queued** helps manage resource allocation by ensuring that pipelines run one at a time, avoiding conflicts and optimizing resource usage.Finally, click on “**Next**” at the bottom.



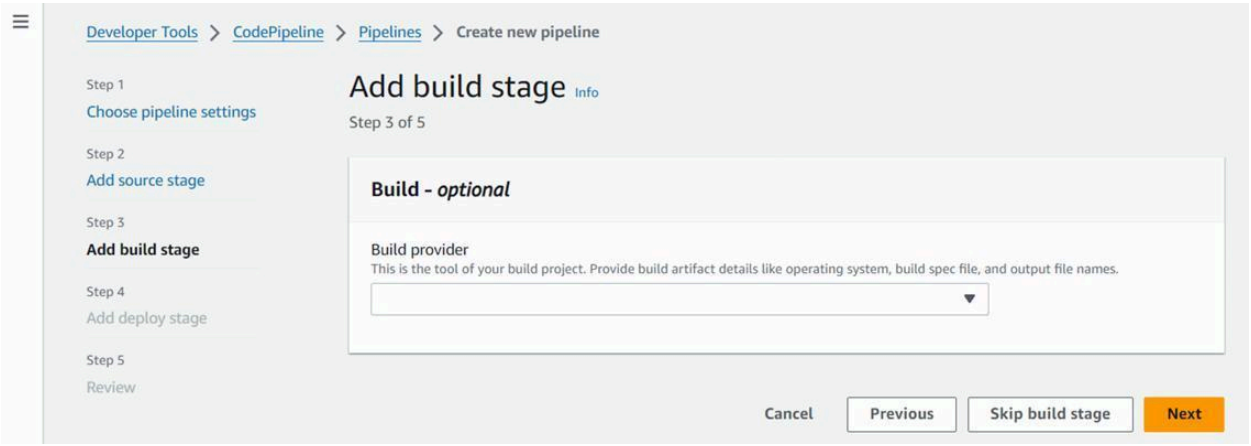
Step 14: In the Source stage of AWS CodePipeline, select GitHub(Version 1) as your source provider, then connect your GitHub account and choose the repository and branch you want to use for the pipeline(by default the branch is **master**). Then click on "**Next**".



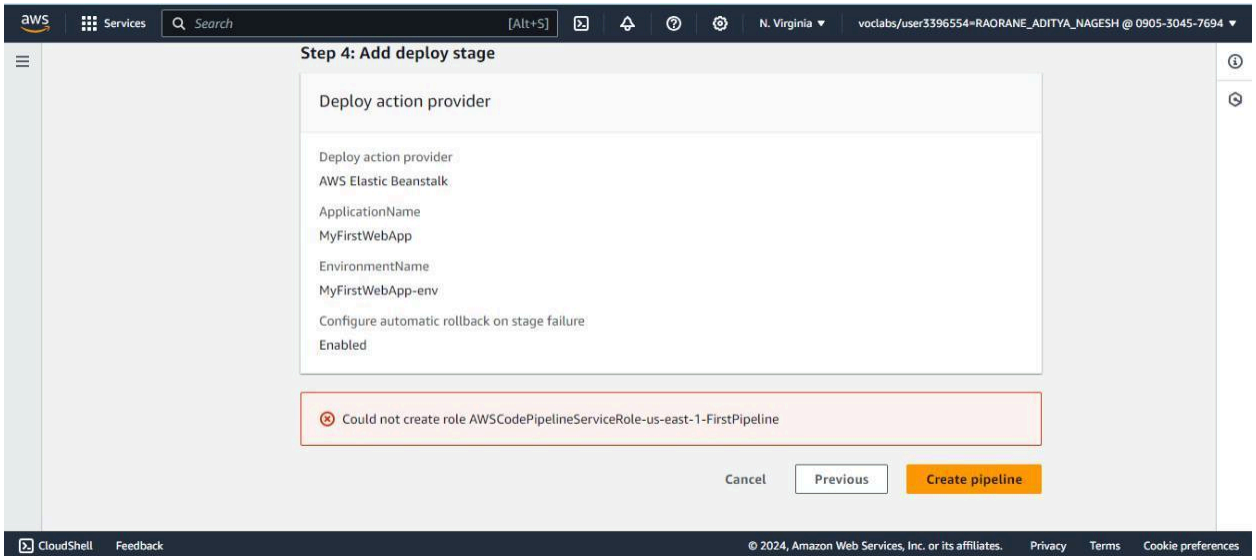
Step 15: Set the Trigger type as no filter. This would allow it to the website to update as soon as some change is made in the github.



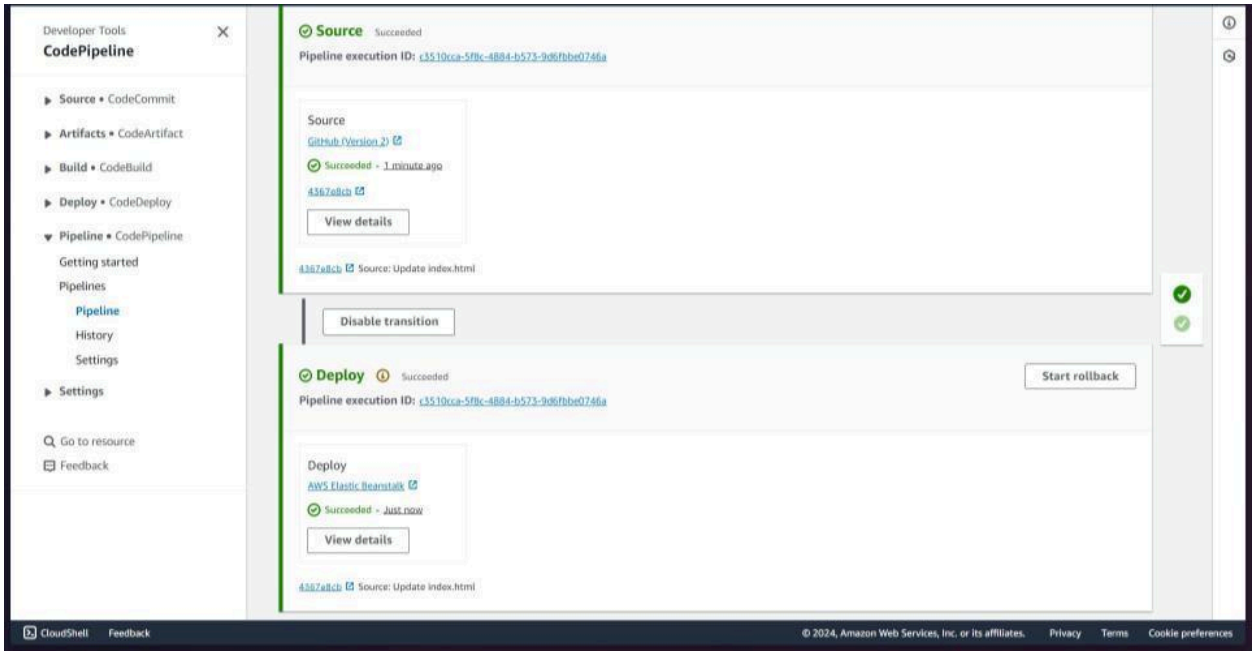
Step 16: Skip the build stage and go to Deploy. Select the deploy provider as AWS Elastic Beanstalk and Input Artifact as SourceArtifact. The application name would be the name of your Elastic Beanstalk. Then click on next.



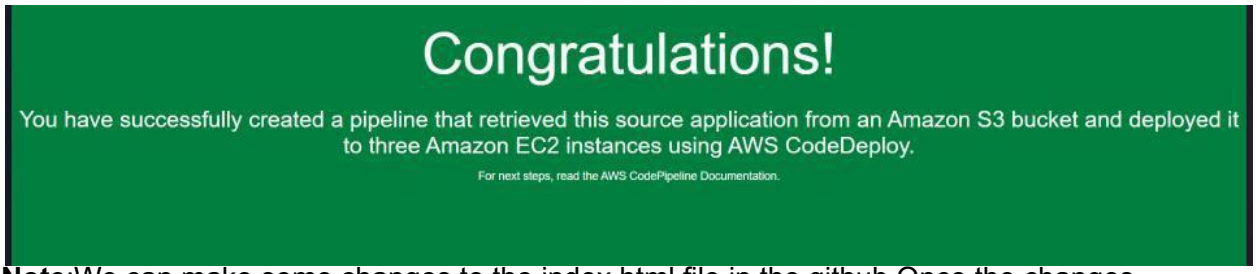
Step 17:The error occurred during the deployment stage because AWS Academy accounts often have **restricted IAM permissions**, preventing the creation of roles needed for the pipeline. This limitation can cause issues when configuring pipelines that require specific roles for execution. To resolve the error, deploy the pipeline using your personal AWS account, which typically has broader permissions and can create the necessary roles for successful deployment.



11]Once the source and deploy stages are successfully completed in AWS CodePipeline, you will see a message indicating "Review" to verify and confirm the pipeline setup before finalizing the deployment.



12]This will successfully show the sample website hosted.



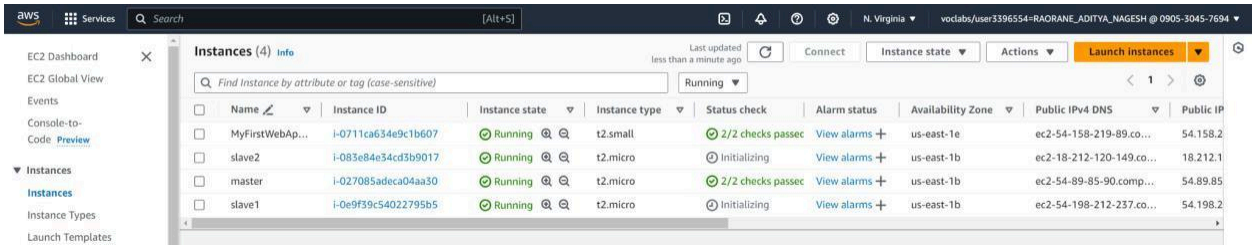
Note:We can make some changes to the index.html file in the github.Once the changes are committed, when the website is refreshed, the changes can be seen.

Name: Aditya Nagesh Raorane Class: D15C/ Batch B Roll No: 44

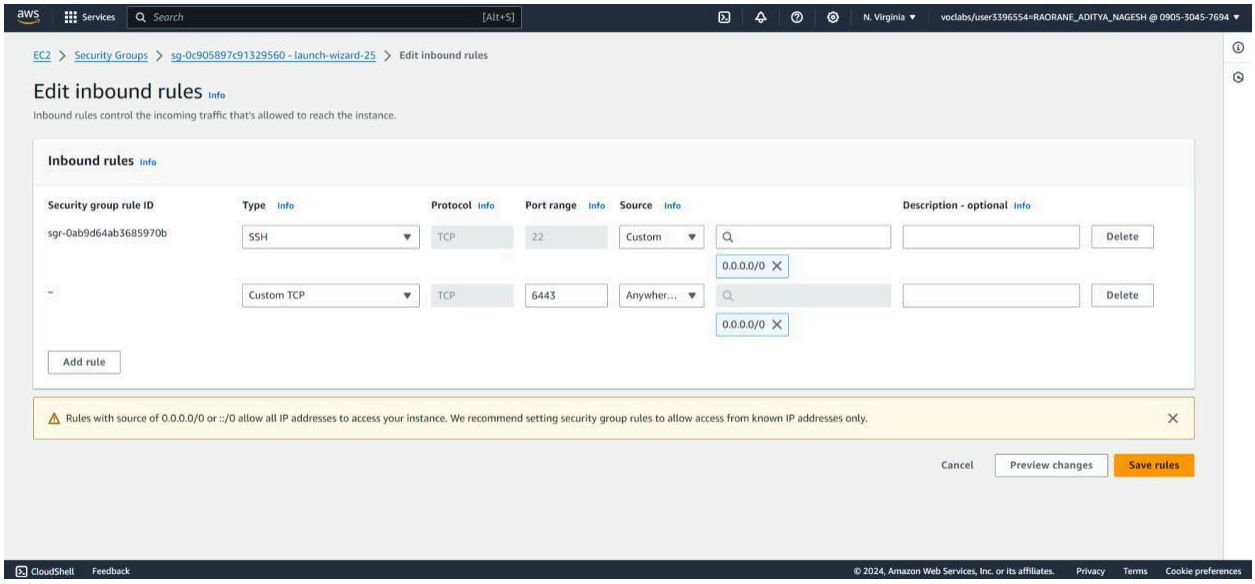
Experiment 03

Aim: To understand the Kubernetes Cluster Architecture, install and spin up a Kubernetes Cluster on Linux Machines/Cloud.

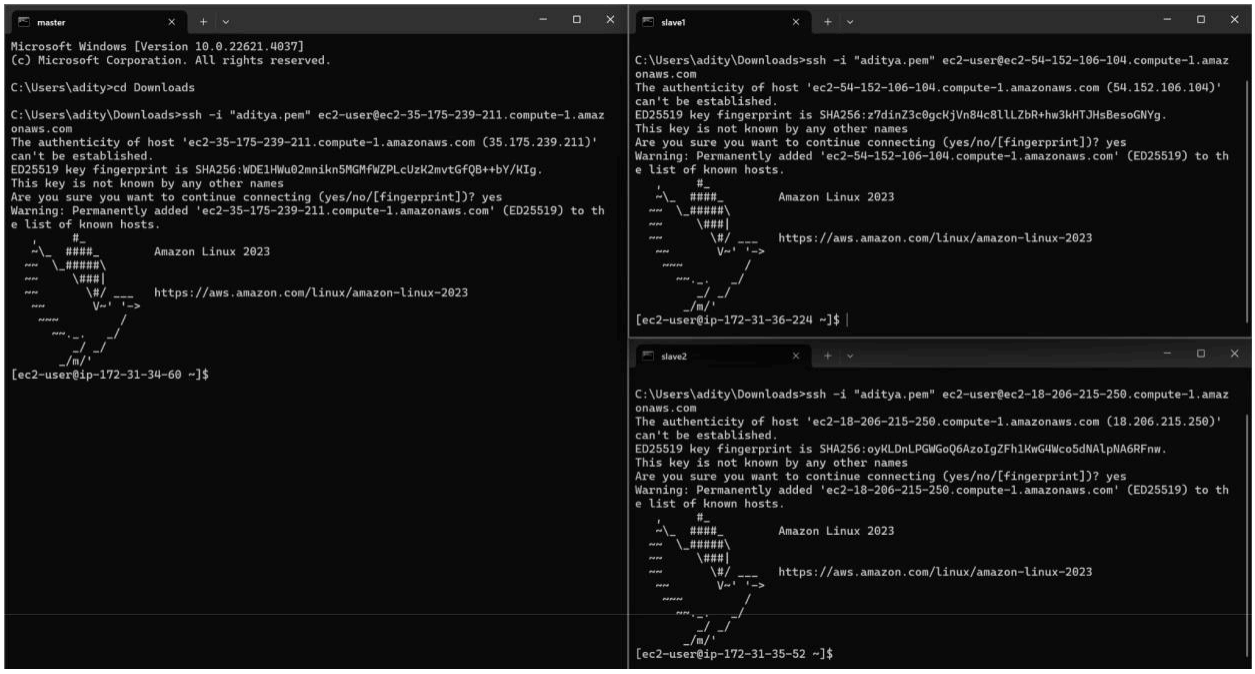
1] Create 3 EC2 instances with all running on Amazon Linux as OS with inbound SSH allowed. To efficiently run the machines it is advised to select the instance type as t2.medium which comes with 4 GiB of memory and 2 vCPU's.



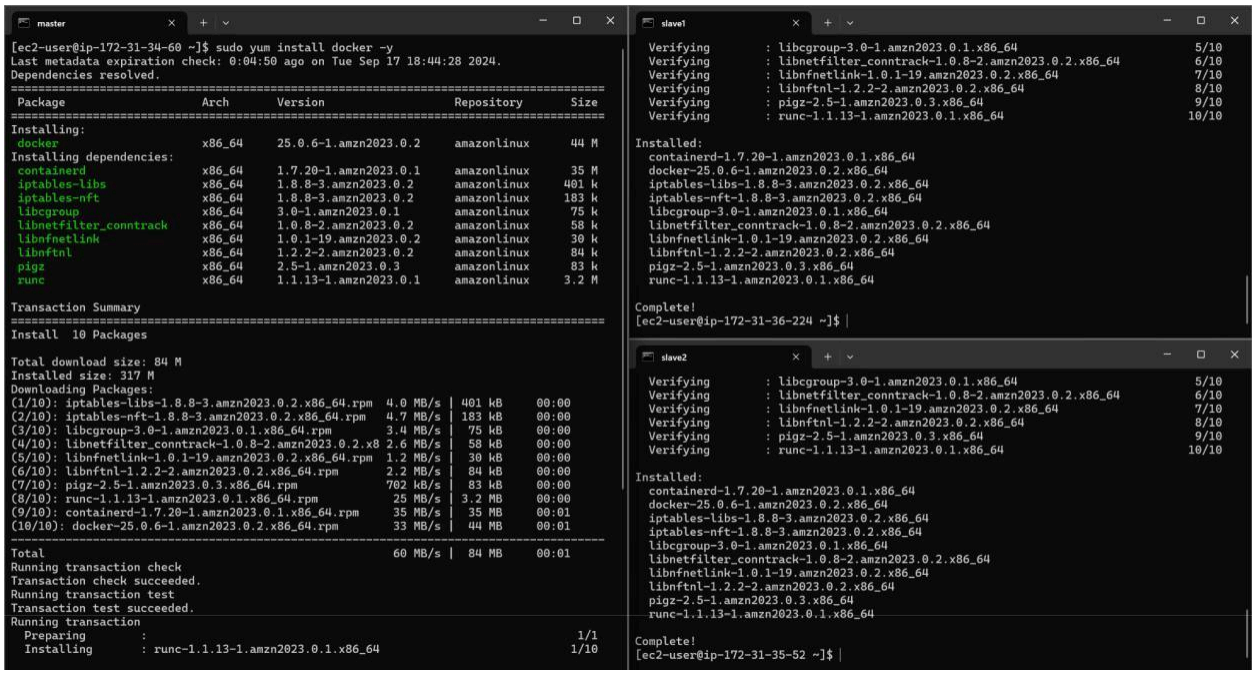
2]Update the inbound rules for the security groups on all three machines to allow TCP traffic on port 6443 from source 0.0.0.0/0.



3] SSH into all 3 machines each in a separate terminal.



4]To install Docker, use the following command:
sudo yum install docker -y

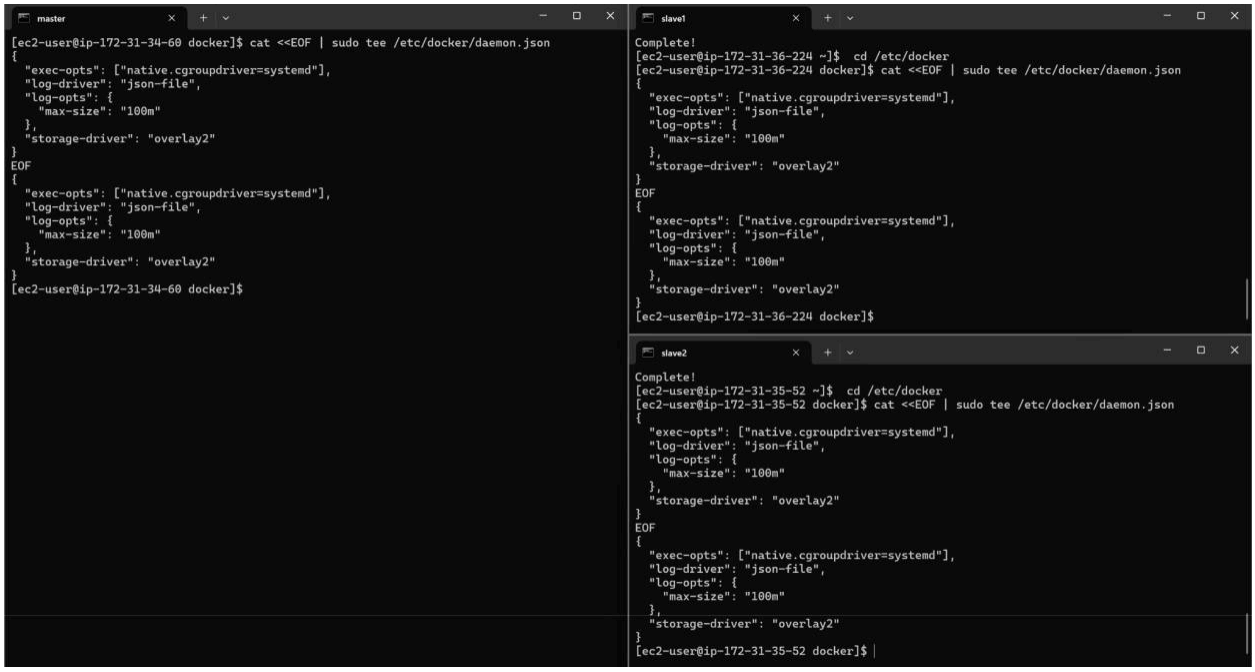


5] Then, configure cgroup in a daemon.json file by using the following commands. This allows kubernetes to manage host more efficiently

- cd /etc/docker

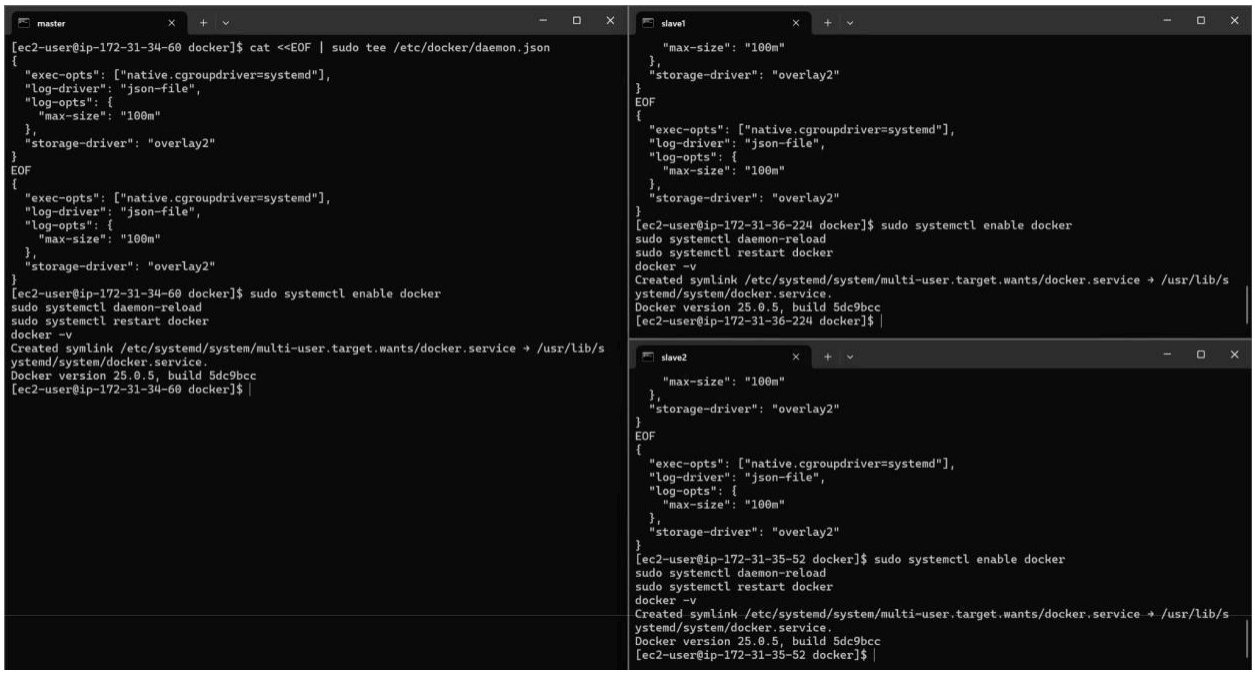
- cat <<EOF | sudo tee /etc/docker/daemon.json

```
{
"exec-opts": ["native.cgroupdriver=systemd"],
"log-driver": "json-file",
"log-opts": {
"max-size": "100m"
},
"storage-driver": "overlay2"
}
EOF
```



6] After configuring restart docker service service :

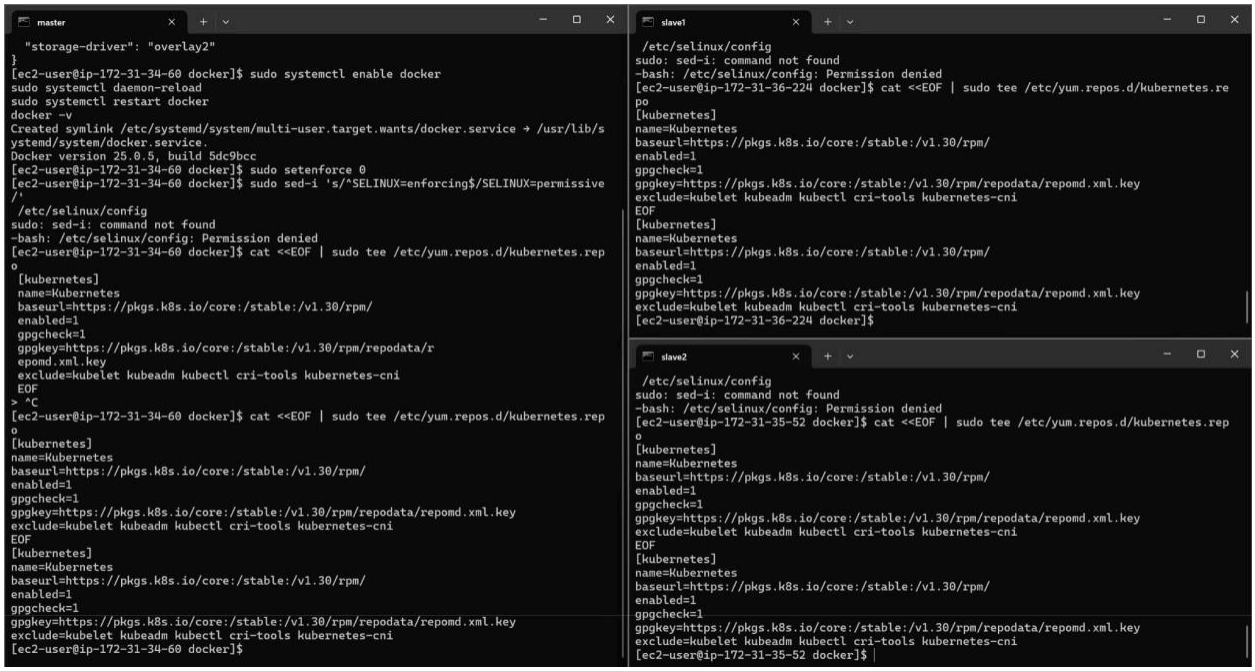
- sudo systemctl enable docker
- sudo systemctl daemon-reload
- sudo systemctl restart docker
- docker-v



7]SELinux needs to be disabled before configuring kubelet to avoid interference with kubernetes api server

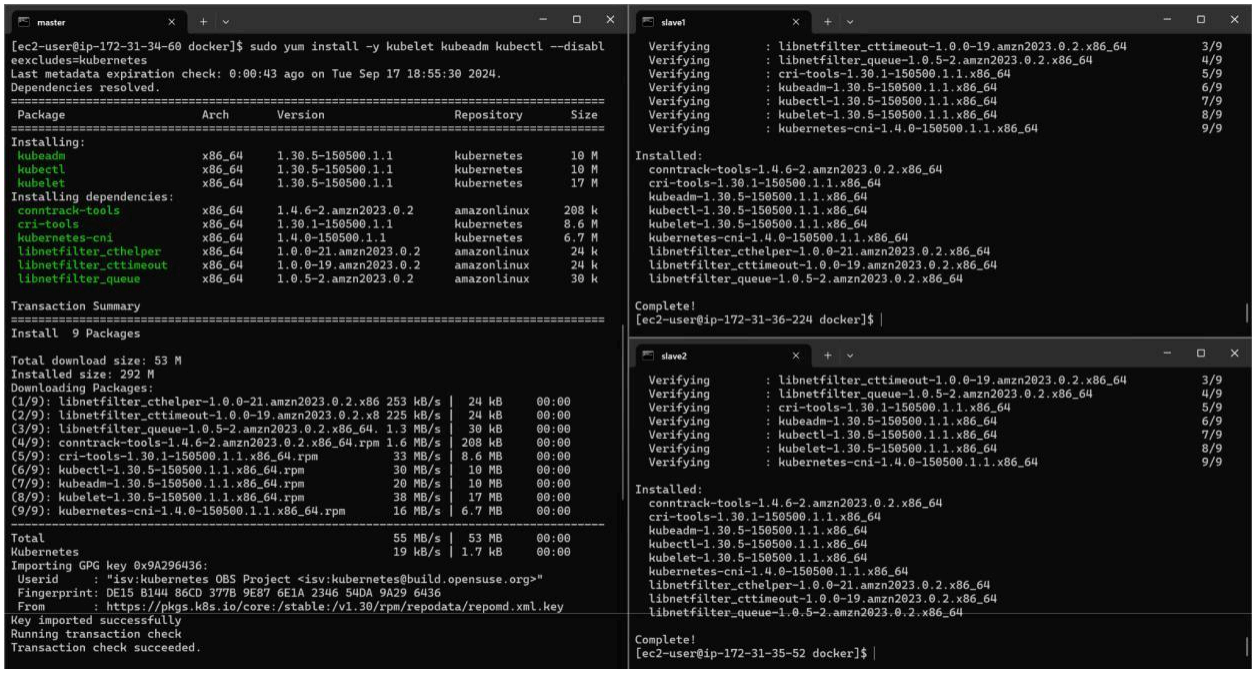
- sudo setenforce 0
- sudo sed-i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config Add kubernetes repository (paste in terminal)

```
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/repodata/r
epomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
```



8] Typefollowingcommandstoinstallsetofkubernetespackages:

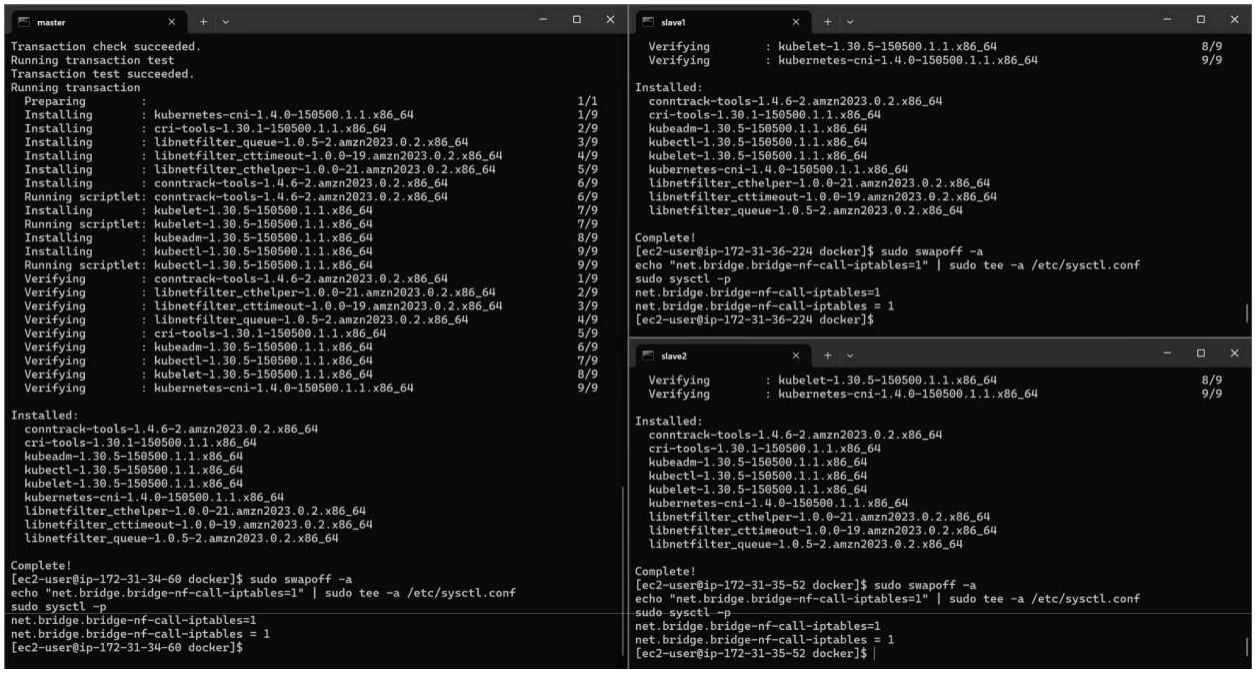
- sudo yum update
- sudo yum install - y kubelet kubeadm kubectl --disableexcludes=kubernetes



9]After installing Kubernetes,we need to configure internet options to allow bridging.

- sudo swapoff -a
- echo "net.bridge.bridge-nf-call-iptables=1"|sudotee-a/etc/sysctl.conf

● sudo sysctl -p



10] **Perform this ONLY on the MASTER**

machine:-a)Initialize Kubernetes By Typing Below Command:

- sudokubeadminit--pod-network-cidr=10.244.0.0/16--ignore-preflight-errors=all

b)Copy the mkdir and chown commands from top and execute them:

- mkdir -p $HOME/.kube
- sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
- sudo chown $(id -u):$(id -g) $HOME/.kube/config

c)Copy the Kubernetes join command from your output to the clipboard

- kubeadm join 172.31.34.60:6443 --token f70o2w.perwi1ygbt7qu724 \ --discovery-token-ca-cert-hash sha256:df9aa1a237965f949c055717ba438799f890a8647b8cd10386fa7ed864e59096

d)Then, add a common networking plugin called flammel file as mentioned code.

- kubectl apply-f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml

```
[ec2-user@ip-172-31-34-60 docker]$ kubectl apply -f https://raw.githubusercontent.com/c
oreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
[ec2-user@ip-172-31-34-60 docker]$
```

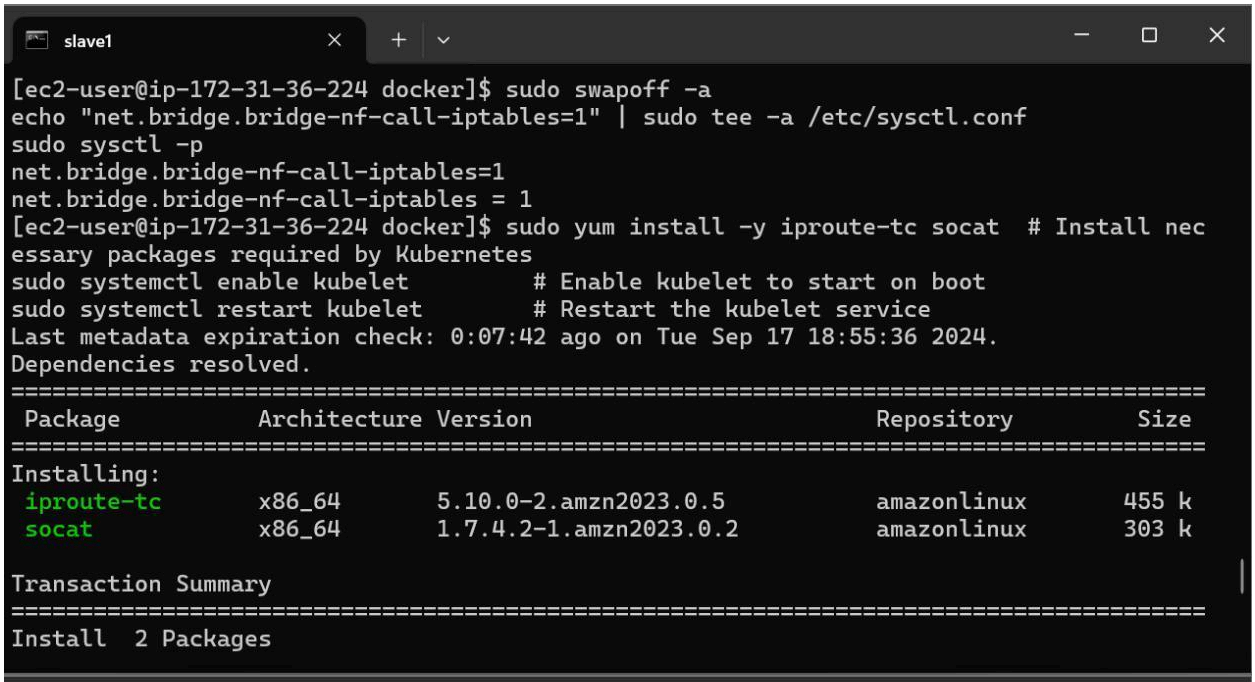
e)Check the created node using this command

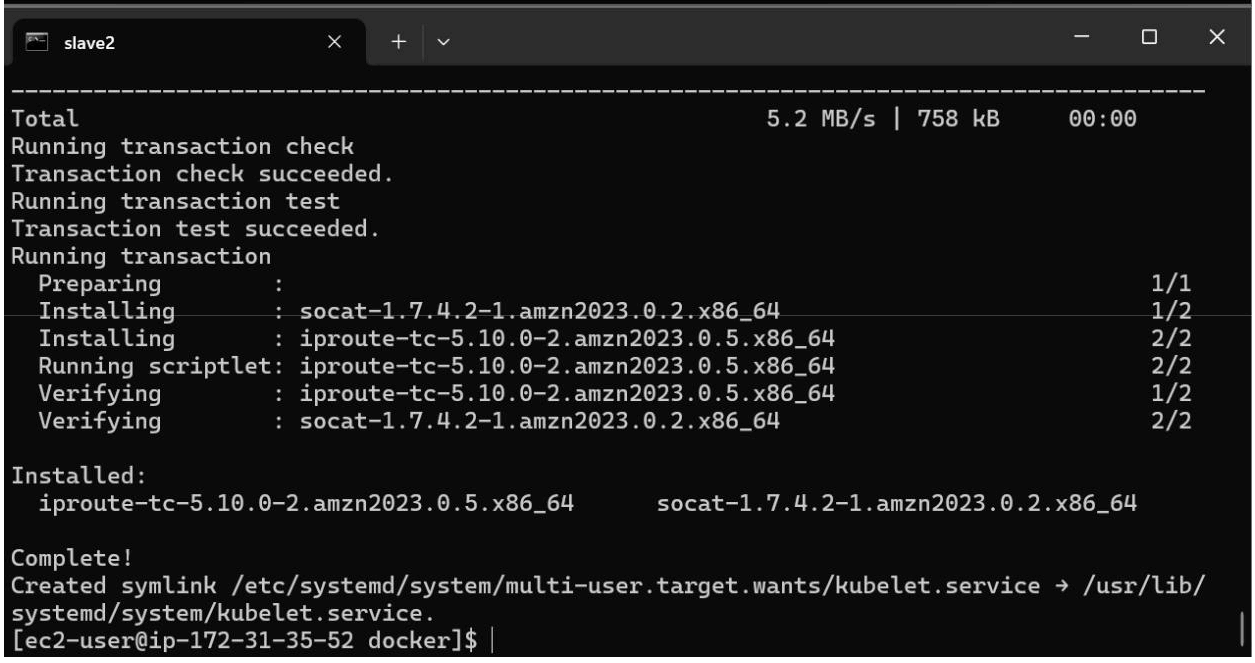
- kubectl get nodes

```
[ec2-user@ip-172-31-34-60 docker]$ kubectl get nodes
NAME                           STATUS   ROLES           AGE     VERSION
ip-172-31-34-60.ec2.internal   Ready    control-plane   5m41s   v1.30.5
```

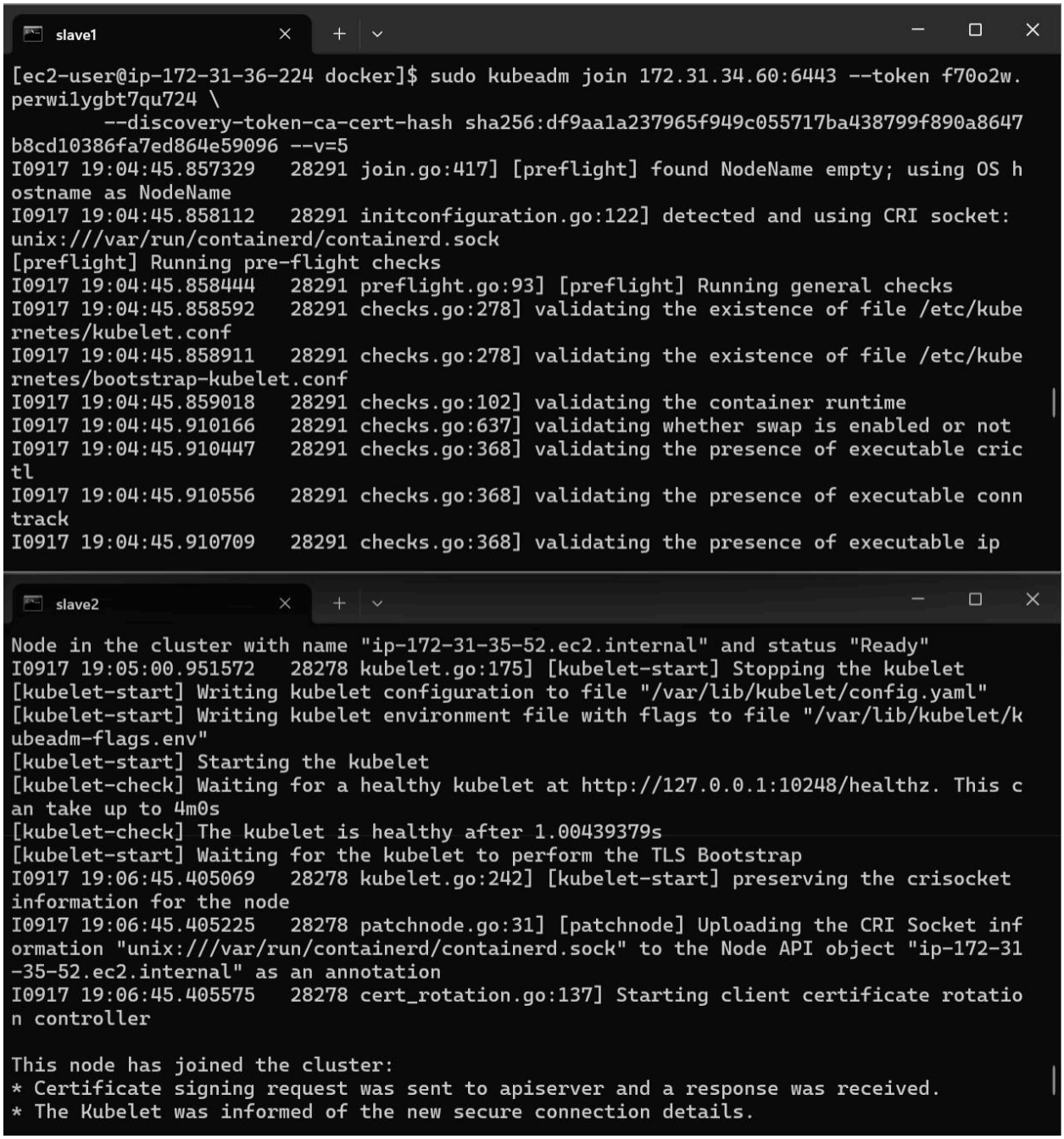
11] **Perform this ONLY on the WORKER machines:-**a)Paste the below command on all 2 worker machines

- sudo yum install iproute-tc socat-y (necessary packages required by kubernetes)
- sudo systemctl enable kubelet
- sudo systemctl restart kubelet





b)kubeadm join 172.31.34.60:6443 --token f70o2w.perwi1ygbt7qu724 \
--discovery-token-ca-cert-hash sha256:df9aa1a237965f949c055717ba438799f890a8647b8cd10386fa7ed864e59096



c) With the help of command the worker nodes are connected to the master node and are ready to do tasks assigned by the master node.

Now we can see in the master/control node of kubernetes that worker nodes are connected by typing **watch kubectl get nodes** in the **master** node instance.

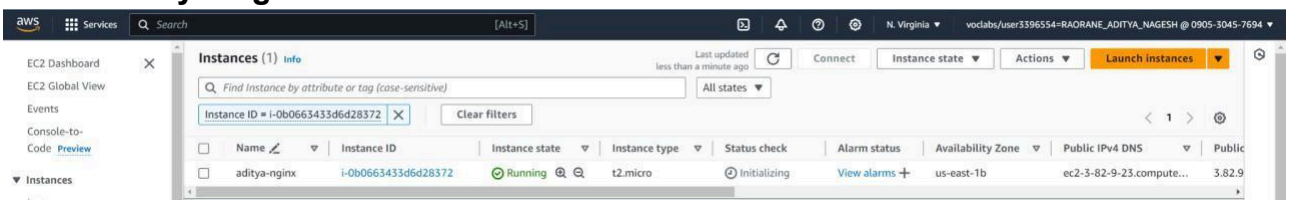
```
[ec2-user@ip-172-31-34-60 docker]$ kubectl get nodes
NAME                            STATUS   ROLES           AGE     VERSION
ip-172-31-34-60.ec2.internal    Ready    control-plane   9m18s   v1.30.5
ip-172-31-35-52.ec2.internal    Ready    <none>          112s    v1.30.5
ip-172-31-36-224.ec2.internal   Ready    <none>          110s    v1.30.5
[ec2-user@ip-172-31-34-60 docker]$
```

Conclusion: We commenced the installation and configuration of the essential Kubernetes packages. While some were accessible via the default Linux repositories, others required the addition of external repositories for proper installation. During the setup, we encountered an issue where the nodes were tainted, resulting in the Kubernetes API server crashing. This was mitigated by removing the taints from the nodes. Additionally, SELinux was disabled to prevent potential conflicts with Kubernetes operations. In conclusion, we successfully integrated the worker nodes with the Kubernetes master node, achieving a fully operational cluster setup.

Experiment 04

Aim: To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

1. Select **Amazon linux** as **OS** image and create an AWS **EC2 instance** named **aditya-nignx**.

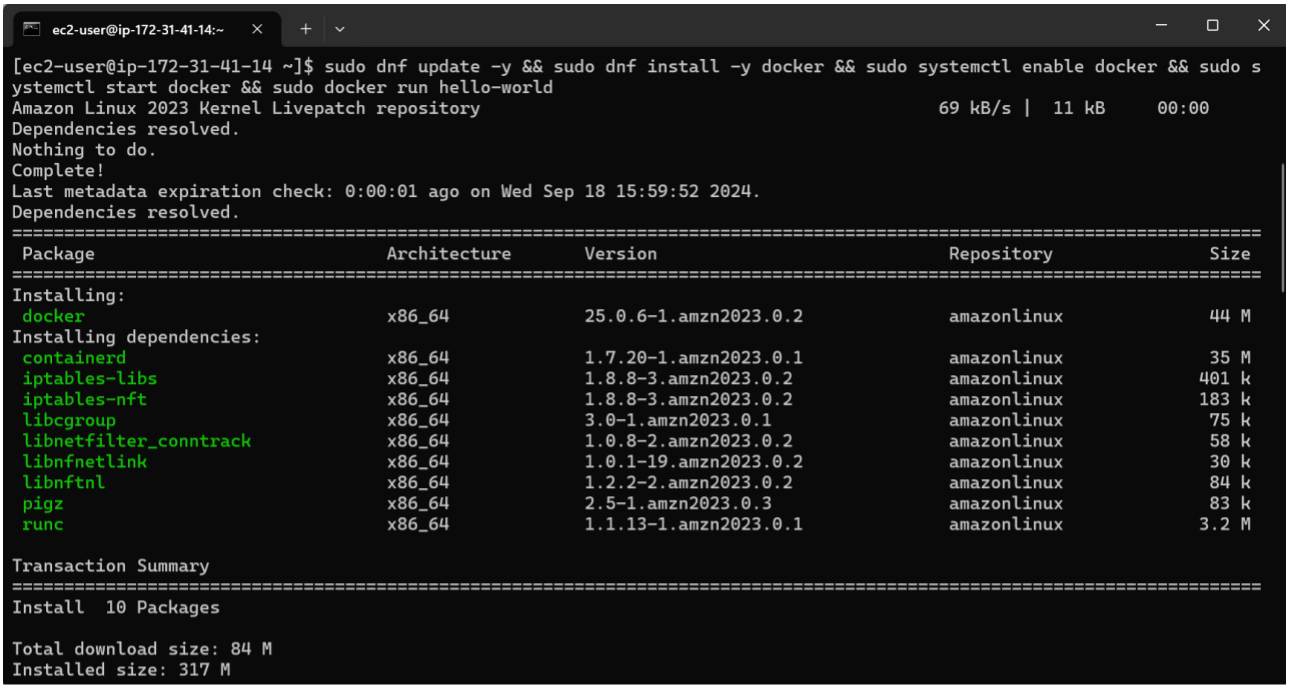


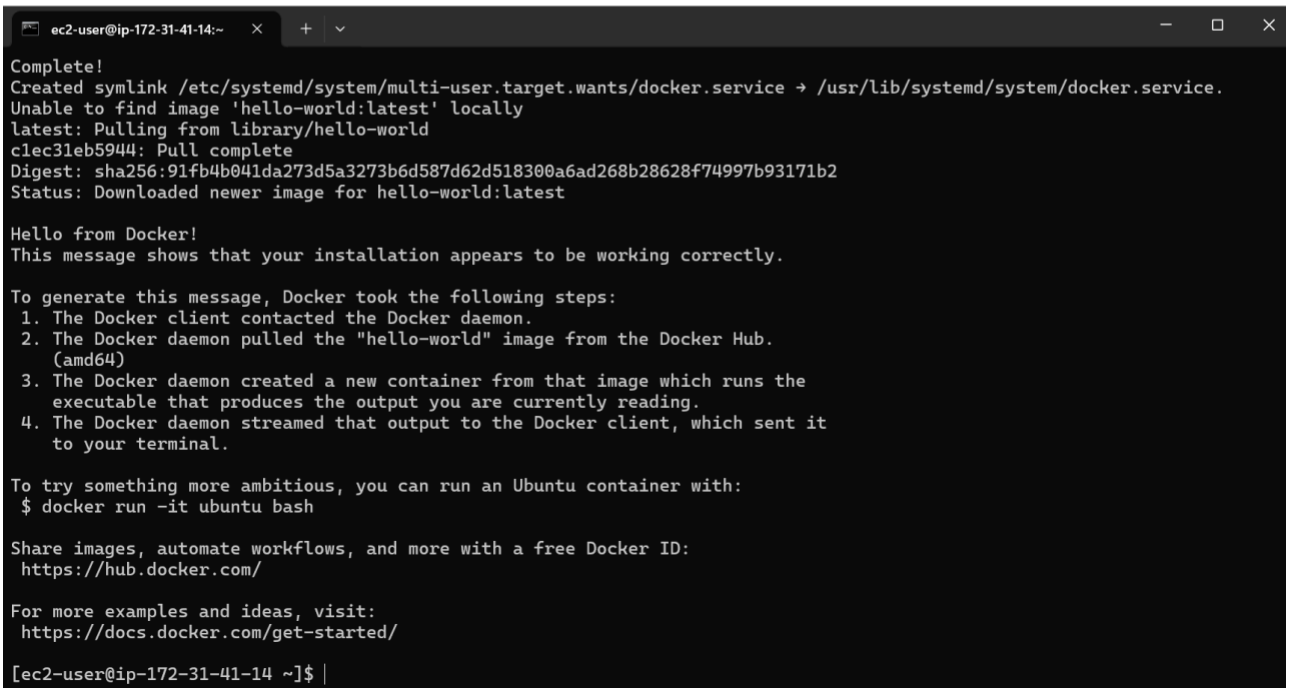
2. Make ssh connection in terminal

```
C:\Users\adity\Downloads>ssh -i "aditya.pem" ec2-user@ec2-3-82-9-23.compute-1.amazonaws.com
The authenticity of host 'ec2-3-82-9-23.compute-1.amazonaws.com (3.82.9.23)' can't be established.
ED25519 key fingerprint is SHA256:Dwfv9Db7wOQNiD7+1XZ6AdAQt8EEo39x5FjlG9DX7dE.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-82-9-23.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
       #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-41-14 ~]$
```

3. Install Docker

- sudo dnf update -y && sudo dnf install -y docker && sudo systemctl enable docker && sudo systemctl start docker && sudo docker run hello-world





Then, configure cgroup in a daemon.json file.This allows kubernetes to manage host more efficiently.

- cd /etc/docker

- cat <<EOF | sudo tee /etc/docker/daemon.json
  {
  "exec-opts":
  ["native.cgroupdriver=systemd"] } EOF

```
[ec2-user@ip-172-31-41-14 docker]$ sudo tee /etc/docker/daemon.json <<EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
[ec2-user@ip-172-31-41-14 docker]$
```

- sudo systemctl daemon-reload
- sudo systemctl restart docker

```
[ec2-user@ip-172-31-41-14 docker]$ sudo systemctl daemon-reload
sudo systemctl restart docker
[ec2-user@ip-172-31-41-14 docker]$
```

4. Install Kubernetes

Note: I'm directly installing binary package you may install from package repository of your distribution

Install CNI plugins (required for most pod network):

CNI_PLUGINS_VERSION="v1.3.0" ARCH="amd64"
DEST="/opt/cni/bin"
sudo mkdir -p "$DEST"
curl -L "https://github.com/containernetworking/plugins/releases/download/${CNI_PLUGINS_VERSION}/cni-plugins-linux-${ARCH}-${CNI_PLUGINS_VERSION}.tgz" | sudo tar -C "$DEST" -xz

```
[ec2-user@ip-172-31-41-14 docker]$ CNI_PLUGINS_VERSION="v1.3.0" ARCH="amd64"
DEST="/opt/cni/bin"
sudo mkdir -p "$DEST"
curl -L "https://github.com/containernetworking/plugins/releases/download/${CNI_PLUGINS_VERSION}/cni-plugins-linux-${ARCH}-${CNI_PLUGINS_VERSION}.tgz" | sudo tar -C "$DEST" -xz
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100 43.2M  100 43.2M    0     0  42.1M      0  0:00:01  0:00:01 --:--:-- 52.2M
[ec2-user@ip-172-31-41-14 docker]$
```

Defne the directory to download command fles:
DOWNLOAD_DIR="/usr/local/bin"
sudo mkdir -p "$DOWNLOAD_DIR"

```
[ec2-user@ip-172-31-41-14 docker]$ DOWNLOAD_DIR="/usr/local/bin"
sudo mkdir -p "$DOWNLOAD_DIR"
```

Optionally install crictl (required for interaction with the Container Runtime Interface (CRI), optional for kubeadm):

CRICTL_VERSION="v1.31.0"
ARCH="amd64"
curl -L
"https://github.com/kubernetes-sigs/cri-tools/releases/download/${CRICTL_VERSION}/
c rictl-${CRICTL_VERSION}-linux-${ARCH}.tar.gz" | sudo tar -C $DOWNLOAD_DIR -xz

```
[ec2-user@ip-172-31-41-14 docker]$ CRICTL_VERSION="v1.31.0" ARCH="amd64"
curl -L "https://github.com/kubernetes-sigs/cri-tools/releases/download/${CRICTL_VERSION}/crictl-${CRICTL_VERSION}-linux
-${ARCH}.tar.gz" | sudo tar -C "$DOWNLOAD_DIR" -xz
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100 17.5M  100 17.5M    0     0  31.3M      0 --:--:-- --:--:-- --:--:-- 66.4M
```

Install `kubeadm`, `kubelet` and add a `kubelet` systemd service:

RELEASE="$(curl -sSL
https://dl.k8s.io/release/stable.txt)" ARCH="amd64"
cd $DOWNLOAD_DIR
sudo curl -L --remote-name-all
https://dl.k8s.io/release/${RELEASE}/bin/linux/${ARCH}/{kubeadm,kubelet }
sudo chmod +x {kubeadm,kubelet}

```
[ec2-user@ip-172-31-41-14 docker]$ RELEASE="$(curl -sSL https://dl.k8s.io/release/stable.txt)"
ARCH="amd64"
cd $DOWNLOAD_DIR
sudo curl -L --remote-name-all https://dl.k8s.io/release/${RELEASE}/bin/linux/${ARCH}/{kubeadm,kubelet}
sudo chmod +x {kubeadm,kubelet}
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   138  100   138    0     0   1643      0 --:--:-- --:--:-- --:--:--  1662
100 55.5M  100 55.5M    0     0  35.9M      0  0:00:01  0:00:01 --:--:-- 26.1M
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   138  100   138    0     0   3349      0 --:--:-- --:--:-- --:--:--  3365
100 73.3M  100 73.3M    0     0  89.4M      0 --:--:-- --:--:-- --:--:--  101M
```

RELEASE_VERSION="v0.16.2"
curl -sSL
"https://raw.githubusercontent.com/kubernetes/release/${RELEASE_VERSION}/cmd/kr
el/templates/latest/kubelet/kubelet.service" | sed "s:/usr/bin:${DOWNLOAD_DIR}:g" |
sudo tee /usr/lib/systemd/system/kubelet.service
sudo mkdir -p /usr/lib/systemd/system/kubelet.service.d
curl -sSL
"https://raw.githubusercontent.com/kubernetes/release/${RELEASE_VERSION}/cmd/kr
el/templates/latest/kubeadm/10-kubeadm.conf" | sed
"s:/usr/bin:${DOWNLOAD_DIR}:g"
| sudo tee /usr/lib/systemd/system/kubelet.service.d/10-kubeadm.conf

```
[ec2-user@ip-172-31-41-14 bin]$ RELEASE_VERSION="v0.16.2"
curl -sSL "https://raw.githubusercontent.com/kubernetes/release/${RELEASE_VERSION}/cmd/krel/templates/latest/kubelet/kub
elet.service" | sed "s:/usr/bin:${DOWNLOAD_DIR}:g" | sudo tee /usr/lib/systemd/system/kubelet.service
sudo mkdir -p /usr/lib/systemd/system/kubelet.service.d
curl -sSL "https://raw.githubusercontent.com/kubernetes/release/${RELEASE_VERSION}/cmd/krel/templates/latest/kubeadm/10-
kubeadm.conf" | sed "s:/usr/bin:${DOWNLOAD_DIR}:g" | sudo tee /usr/lib/systemd/system/kubelet.service.d/10-kubeadm.conf
[Unit]
Description=kubelet: The Kubernetes Node Agent
Documentation=https://kubernetes.io/docs/
Wants=network-online.target
After=network-online.target

[Service]
ExecStart=/usr/local/bin/kubelet
Restart=always
StartLimitInterval=0
RestartSec=10

[Install]
WantedBy=multi-user.target
```

```
# Note: This dropin only works with kubeadm and kubelet v1.11+
[Service]
Environment="KUBELET_KUBECONFIG_ARGS=--bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kub
ernetes/kubelet.conf"
Environment="KUBELET_CONFIG_ARGS=--config=/var/lib/kubelet/config.yaml"
# This is a file that "kubeadm init" and "kubeadm join" generates at runtime, populating the KUBELET_KUBEADM_ARGS variab
le dynamically
EnvironmentFile=-/var/lib/kubelet/kubeadm-flags.env
# This is a file that the user can use for overrides of the kubelet args as a last resort. Preferably, the user should u
se
# the .NodeRegistration.KubeletExtraArgs object in the configuration files instead. KUBELET_EXTRA_ARGS should be sourced
 from this file.
EnvironmentFile=-/etc/sysconfig/kubelet
ExecStart=
ExecStart=/usr/local/bin/kubelet $KUBELET_KUBECONFIG_ARGS $KUBELET_CONFIG_ARGS $KUBELET_KUBEADM_ARGS $KUBELET_EXTRA_ARGS
[ec2-user@ip-172-31-41-14 bin]$
```

Now we need to install kubectl

Set up repository:

```
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repo
md.xml.key
EOF
```

```
[ec2-user@ip-172-31-41-14 bin]$ cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
EOF
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
[ec2-user@ip-172-31-41-14 bin]$
```

sudo yum install -y kubectl

```
ec2-user@ip-172-31-41-14:/us
[ec2-user@ip-172-31-41-14 bin]$ sudo yum install -y kubectl
Kubernetes                                                    66 kB/s | 9.4 kB     00:00
Dependencies resolved.
================================================================================
 Package          Architecture     Version                  Repository        Size
================================================================================
Installing:
 kubectl          x86_64           1.31.1-150500.1.1        kubernetes        11 M

Transaction Summary
================================================================================
Install  1 Package

Total download size: 11 M
Installed size: 54 M
Downloading Packages:
kubectl-1.31.1-150500.1.1.x86_64.rpm                          44 MB/s |  11 MB     00:00
--------------------------------------------------------------------------------
Total                                                         42 MB/s |  11 MB     00:00
Kubernetes                                                    26 kB/s | 1.7 kB     00:00
Importing GPG key 0x9A296436:
 Userid     : "isv:kubernetes OBS Project <isv:kubernetes@build.opensuse.org>"
 Fingerprint: DE15 B144 86CD 377B 9E87 6E1A 2346 54DA 9A29 6436
 From       : https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
Key imported successfully
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : kubectl-1.31.1-150500.1.1.x86_64                       1/1
  Verifying        : kubectl-1.31.1-150500.1.1.x86_64                       1/1

Installed:
  kubectl-1.31.1-150500.1.1.x86_64

Complete!
[ec2-user@ip-172-31-41-14 bin]$
```

```
[ec2-user@ip-172-31-41-14 bin]$ kubectl version
Client Version: v1.31.1
Kustomize Version: v5.4.2
```

We have installed successfully installed kubernetes

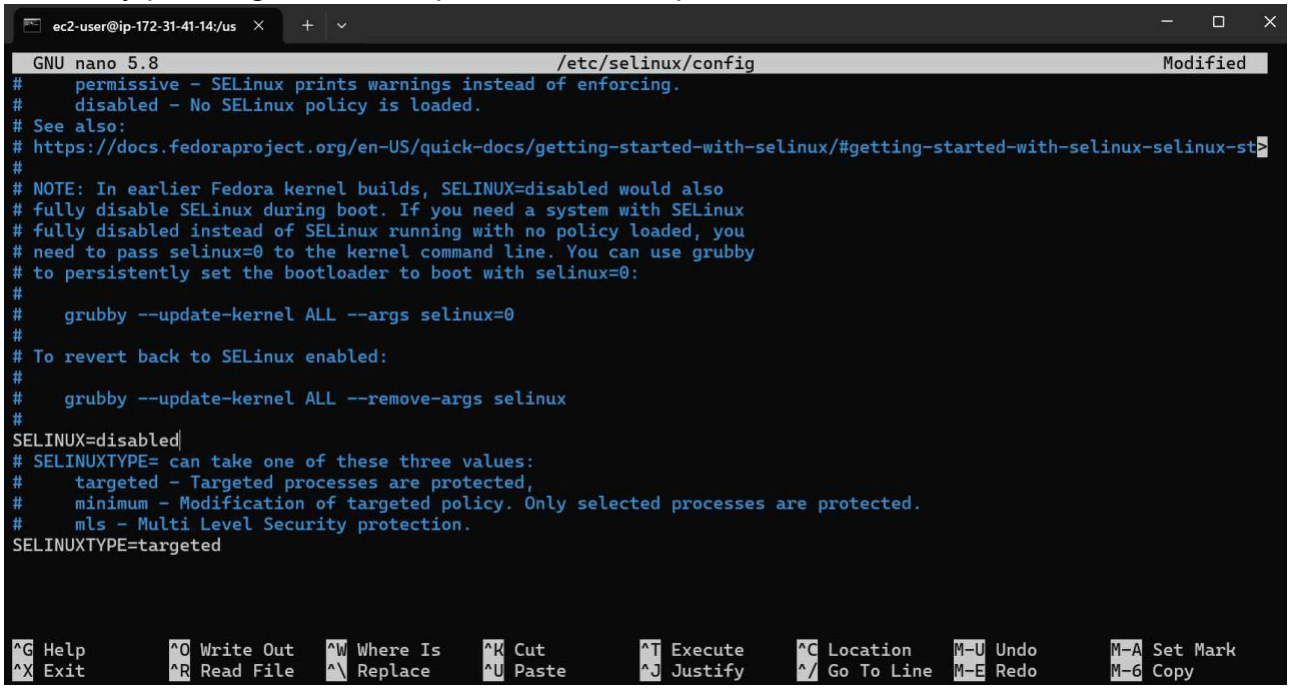
After installing Kubernetes, we need to configure internet options to allow bridging.

```
sudo swapoff -a
&&   echo
"net.bridge.bridg
e-nf-call-iptable
s=1" | sudo tee
-a
/etc/sysctl.conf
&& sudo sysctl -p
```

```
[ec2-user@ip-172-31-41-14 bin]$ sudo swapoff -a && echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl
.conf && sudo sysctl -p
net.bridge.bridge-nf-call-iptables=1
net.bridge.bridge-nf-call-iptables = 1
[ec2-user@ip-172-31-41-14 bin]$
```

Disable SELINUX

Type **sudo nano /etc/selinux/config** and set the value of **SELINUX=disabled** instead of **SELINUX=permissive**
Save the file by pressing ctrl+o then press enter then press ctrl+x



Then reboot the system using **sudo reboot**

After rebooting we need to make ssh connection with machine after it gets disconnected

```
[ec2-user@ip-172-31-41-14 bin]$ sudo nano /etc/selinux/config
[ec2-user@ip-172-31-41-14 bin]$ sudo reboot

Broadcast message from root@localhost on pts/1 (Wed 2024-09-18 16:15:38 UTC):

The system will reboot now!

[ec2-user@ip-172-31-41-14 bin]$ Connection to ec2-3-82-9-23.compute-1.amazonaws.com closed by remote host.
Connection to ec2-3-82-9-23.compute-1.amazonaws.com closed.

C:\Users\adity\Downloads>ssh -i "aditya.pem" ec2-user@ec2-3-82-9-23.compute-1.amazonaws.com
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Wed Sep 18 15:59:22 2024 from 202.179.85.199
[ec2-user@ip-172-31-41-14 ~]$
```

Now if we type command **sestatus,** then it show disabled

```
[ec2-user@ip-172-31-41-14 ~]$ sestatus
SELinux status:                 disabled
[ec2-user@ip-172-31-41-14 ~]$
```

5. Initialize the Kubecluster

Install packages socat and iproute-tc and conntrack to avoid prelight errors

- **sudo dnf install socat iproute-tc conntrack-tools -y**

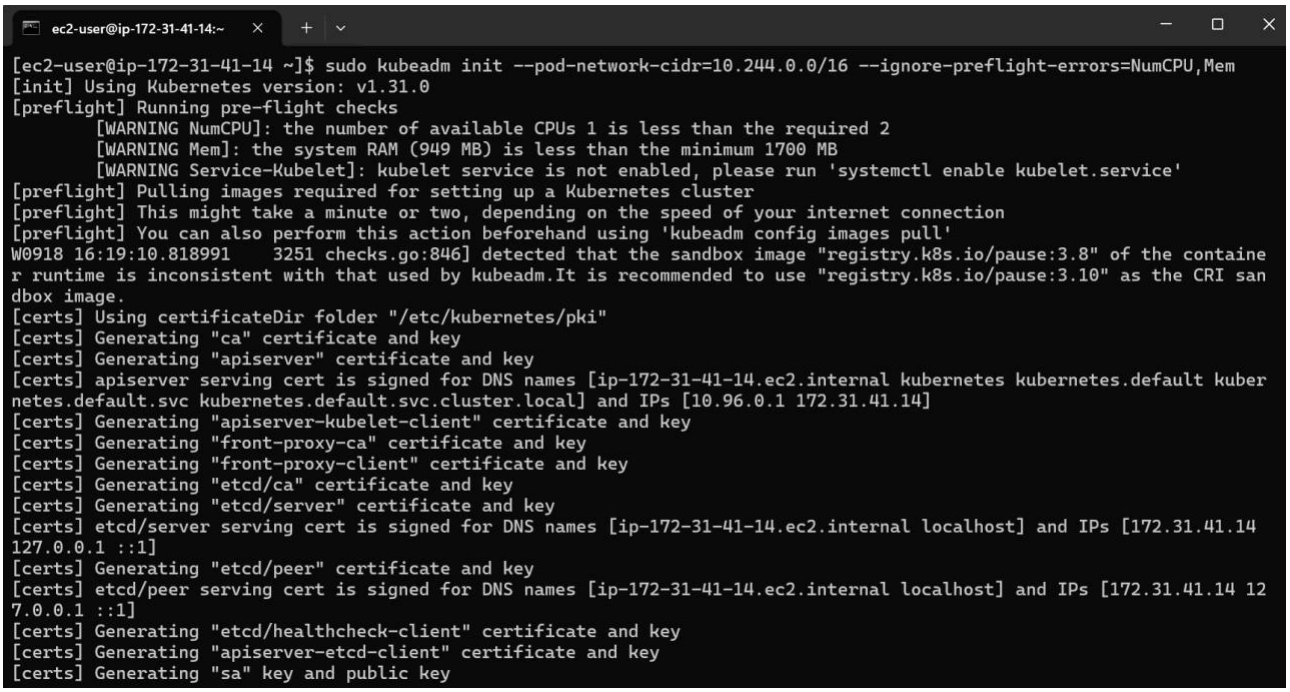
```
ec2-user@ip-172-31-41-14:~
[ec2-user@ip-172-31-41-14 ~]$ sudo dnf install socat iproute-tc conntrack-tools -y
Last metadata expiration check: 0:06:41 ago on Wed Sep 18 16:10:55 2024.
Dependencies resolved.
================================================================================
 Package                Architecture   Version                    Repository   Size
================================================================================
Installing:
 conntrack-tools        x86_64         1.4.6-2.amzn2023.0.2       amazonlinux  208 k
 iproute-tc             x86_64         5.10.0-2.amzn2023.0.5      amazonlinux  455 k
 socat                  x86_64         1.7.4.2-1.amzn2023.0.2     amazonlinux  303 k
Installing dependencies:
 libnetfilter_cthelper  x86_64         1.0.0-21.amzn2023.0.2      amazonlinux   24 k
 libnetfilter_cttimeout x86_64         1.0.0-19.amzn2023.0.2      amazonlinux   24 k
 libnetfilter_queue     x86_64         1.0.5-2.amzn2023.0.2       amazonlinux   30 k

Transaction Summary
================================================================================
Install  6 Packages

Total download size: 1.0 M
Installed size: 2.8 M
Downloading Packages:
(1/6): libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64.rpm  461 kB/s |  24 kB  00:00
(2/6): conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64.rpm         3.0 MB/s | 208 kB  00:00
(3/6): libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64.rpm 1.0 MB/s |  24 kB  00:00
(4/6): libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64.rpm      1.2 MB/s |  30 kB  00:00
(5/6): socat-1.7.4.2-1.amzn2023.0.2.x86_64.rpm                 5.9 MB/s | 303 kB  00:00
(6/6): iproute-tc-5.10.0-2.amzn2023.0.5.x86_64.rpm             3.1 MB/s | 455 kB  00:00
--------------------------------------------------------------------------------
Total                                                          5.3 MB/s | 1.0 MB  00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64         1/6
  Installing       : libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64    2/6
  Installing       : libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64     3/6
  Installing       : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64            4/6
  Running scriptlet: conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64            4/6
  Installing       : socat-1.7.4.2-1.amzn2023.0.2.x86_64                    5/6
  Installing       : iproute-tc-5.10.0-2.amzn2023.0.5.x86_64                6/6
  Running scriptlet: iproute-tc-5.10.0-2.amzn2023.0.5.x86_64                6/6
  Verifying        : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64            1/6
  Verifying        : iproute-tc-5.10.0-2.amzn2023.0.5.x86_64                2/6
  Verifying        : libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64     3/6
  Verifying        : libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64    4/6
  Verifying        : libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64         5/6
  Verifying        : socat-1.7.4.2-1.amzn2023.0.2.x86_64                    6/6

Installed:
  conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64
  iproute-tc-5.10.0-2.amzn2023.0.5.x86_64
  libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64
  libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64
  libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64
  socat-1.7.4.2-1.amzn2023.0.2.x86_64

Complete!
```

- sudo kubeadm init --pod-network-cidr=10.244.0.0/16

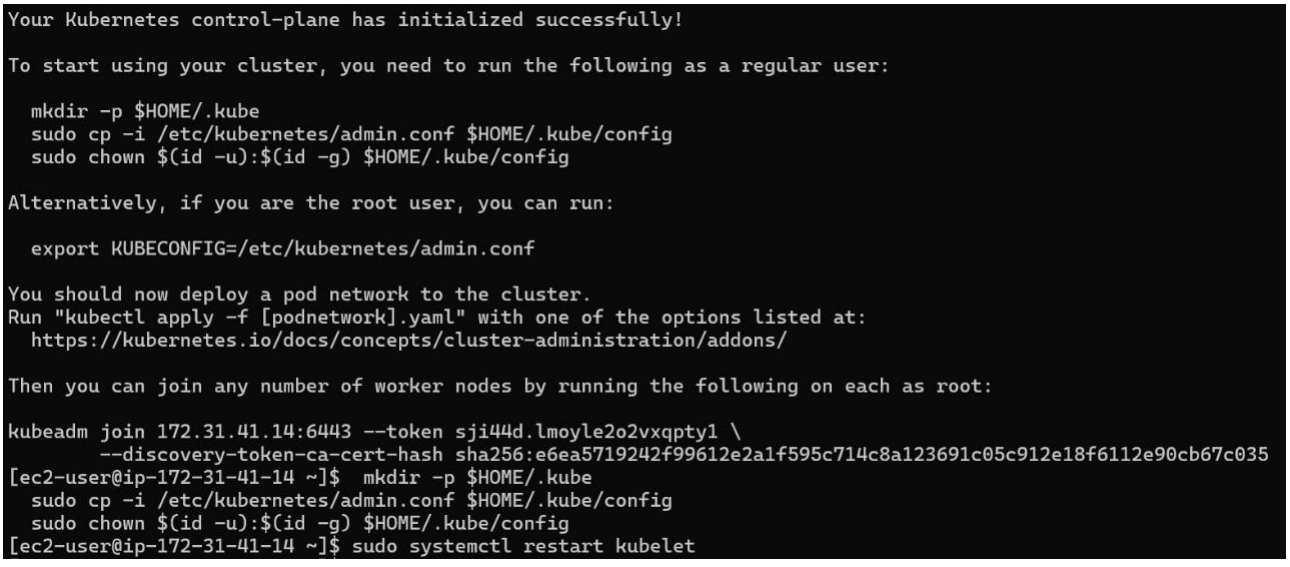
  --ignore-preflight-errors=NumCPU,Mem

Copy the mkdir and chown commands from the top and execute them

- mkdir -p $HOME/.kube
- sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config sudo chown $(id -u):$(id -g) $HOME/.kube/config



- **sudo systemctl restart kubelet**



- Then, add a common networking plugin called flannel as mentioned in the code.
  kubectl apply -f
  https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-fla nnel.yml

```
[ec2-user@ip-172-31-41-14 ~]$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
[ec2-user@ip-172-31-41-14 ~]$
```

Now type **kubectl get nodes**

```
[ec2-user@ip-172-31-41-14 ~]$ kubectl get nodes
NAME                                STATUS   ROLES           AGE     VERSION
ip-172-31-41-14.ec2.internal   Ready    control-plane   3m13s   v1.31.1
[ec2-user@ip-172-31-41-14 ~]$
```

Note: If any time of get error of connection refused just restart the kubelet service (sudo systemctl restart kubelet)

Now that the cluster is up and running, we can deploy our nginx server on this cluster. Apply this deployment file using this command to create a deployment

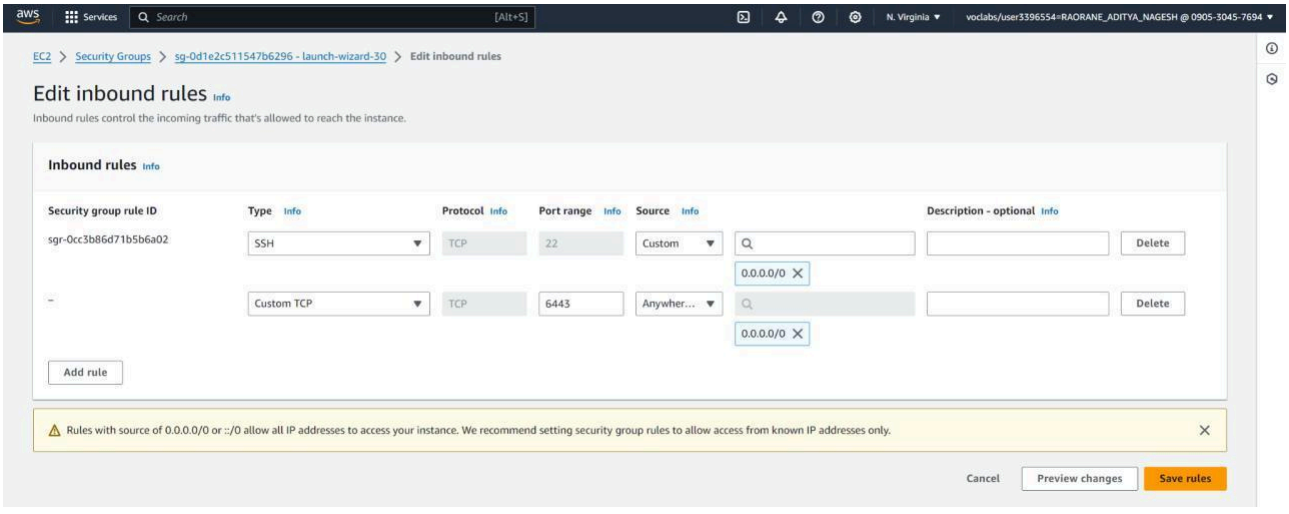
- **kubectl apply -f https://k8s.io/examples/application/deployment.yaml**

```
[ec2-user@ip-172-31-41-14 ~]$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
[ec2-user@ip-172-31-41-14 ~]$
```

Use ‘kubectl get pods’ to verify if the deployment was properly created and the pod is working correctly.

```
[ec2-user@ip-172-31-41-14 ~]$ kubectl get pods
The connection to the server 172.31.41.14:6443 was refused - did you specify the right host or port?
[ec2-user@ip-172-31-41-14 ~]$
```

Add an inbound rule under SSH security groups which will allow the traffic for a **custom TCP port** with port number **6443** with setting the source as **anywhere from IPv4.**



As we can see our pods are in pending state
On checking logs to we came to know the pods are in tainted state (using command **kubectl describe pod nginx-deployment-d556bf558-7zthh**)

To make pods untainted
Type **kubectl get nodes** to see the name of the node.

Then type command **kubectl taint nodes <NODE_NAME> - -all**
In my case it is as follows:
kubectl taint nodes ip-172-31-41-14.ec2.internal node-role.kubernetes.io/control-plane-

After executing the above command, check again the status of pods if still pending then restart kubelet, wait for 1-2 minutes and check again.

```
[ec2-user@ip-172-31-41-14 ~]$ kubectl taint nodes ip-172-31-41-14.ec2.internal node-role.kubernetes.io/control-plane-
node/ip-172-31-41-14.ec2.internal untainted
```

```
[ec2-user@ip-172-31-41-14 ~]$ kubectl get pods
NAME                                READY   STATUS    RESTARTS      AGE
nginx-deployment-d556bf558-7zthh    1/1     Running   2 (27s ago)   60s
nginx-deployment-d556bf558-vdldn    1/1     Running   2 (27s ago)   60s
[ec2-user@ip-172-31-41-14 ~]$
```

As we can see our pods are running

- Lastly, port forward the deployment to your localhost so that you can view it. **kubectl port-forward <POD_NAME> 8080:80**

In my case : **kubectl port-forward nginx-deployment-d556bf558-7zthh 8080:80**

Note: if you are getting connection refused error then restart kubelet.

```
[ec2-user@ip-172-31-41-14 ~]$ kubectl port-forward nginx-deployment-d556bf558-7zthh 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
```

As port forwarding is active so we cannot type other commands.
Open new terminal window and make ssh connection to same machine And type command **curl --head** http://127.0.0.1:8080

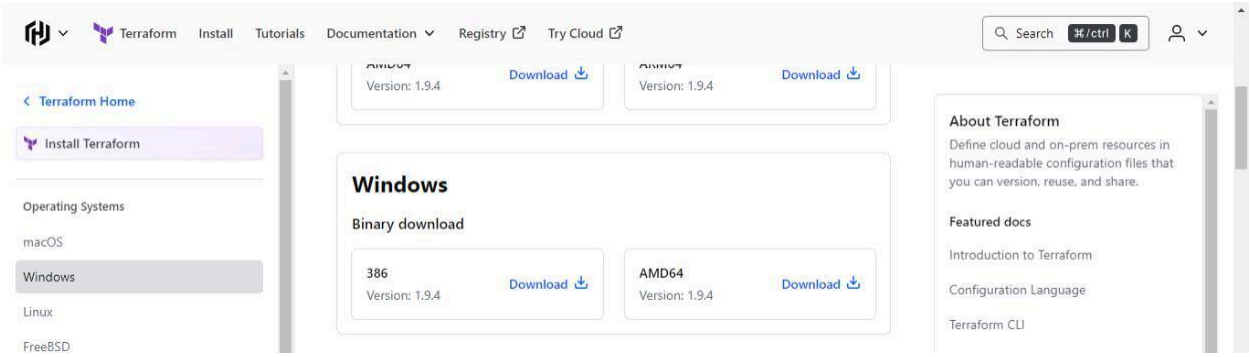
```
[ec2-user@ip-172-31-41-14 ~]$ curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Wed, 18 Sep 2024 16:33:35 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 04 Dec 2018 14:44:49 GMT
Connection: keep-alive
ETag: "5c0692e1-264"
Accept-Ranges: bytes

[ec2-user@ip-172-31-41-14 ~]$
```

Response status 200 (OK) indicates that our nginx server is running successfully on kubernetes

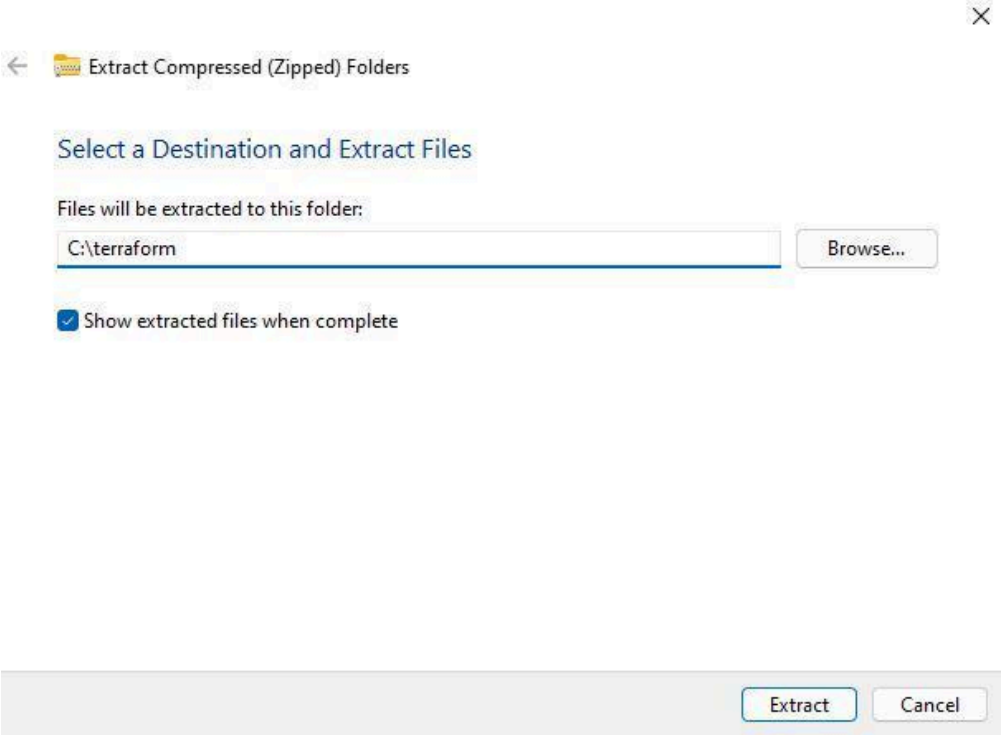
Conclusion: We began by installing and configuring Docker and Kubernetes, encountering some initial issues with the Kubernetes API server, which were resolved by restarting the `kubelet` service. The pods didn't start at first due to taints on the nodes, which we removed to allow normal pod scheduling. After resolving these errors, we successfully deployed Nginx server pods and configured them to be accessible via port forwarding. Additionally, we configured the SSH security group by adding an inbound rule to permit traffic on TCP port 6443 from any IPv4 address.

Experiment 05: Installation of Terraform

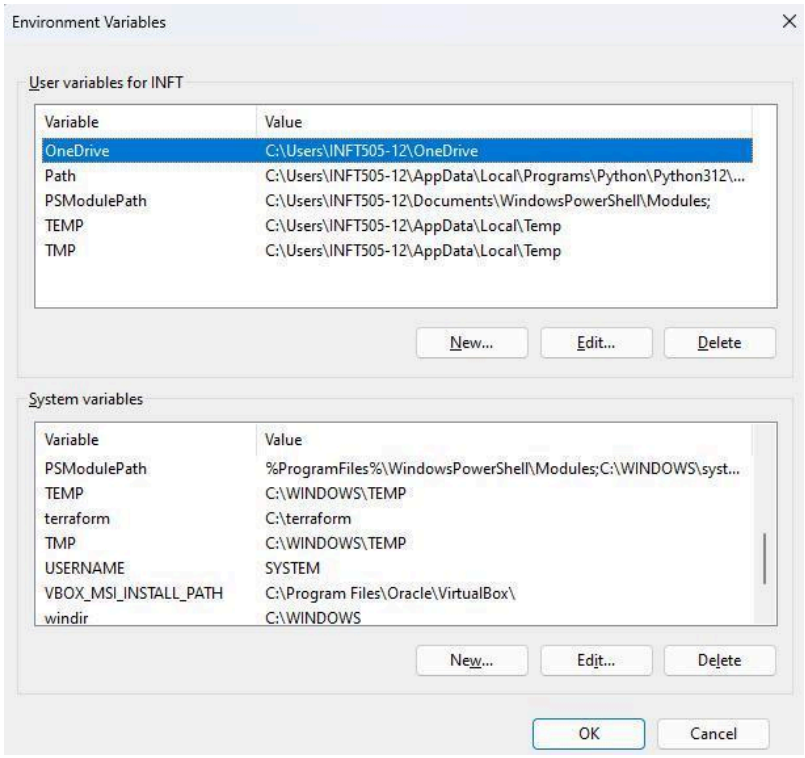
Step 1:To install Terraform, visit the official Terraform website mentioned below, go to the Downloads section, select Windows, and download the 64-bit version for your system.
website:https://www.terraform.io/downloads.html



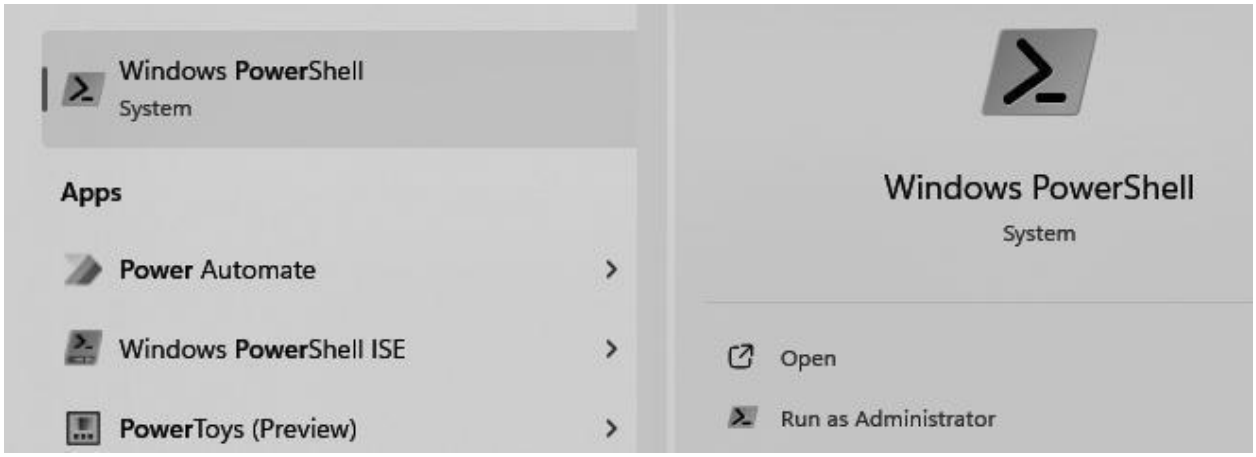
Step 2: Extract the downloaded `Terraform.exe` file to the `C:\Terraform` directory on your system.



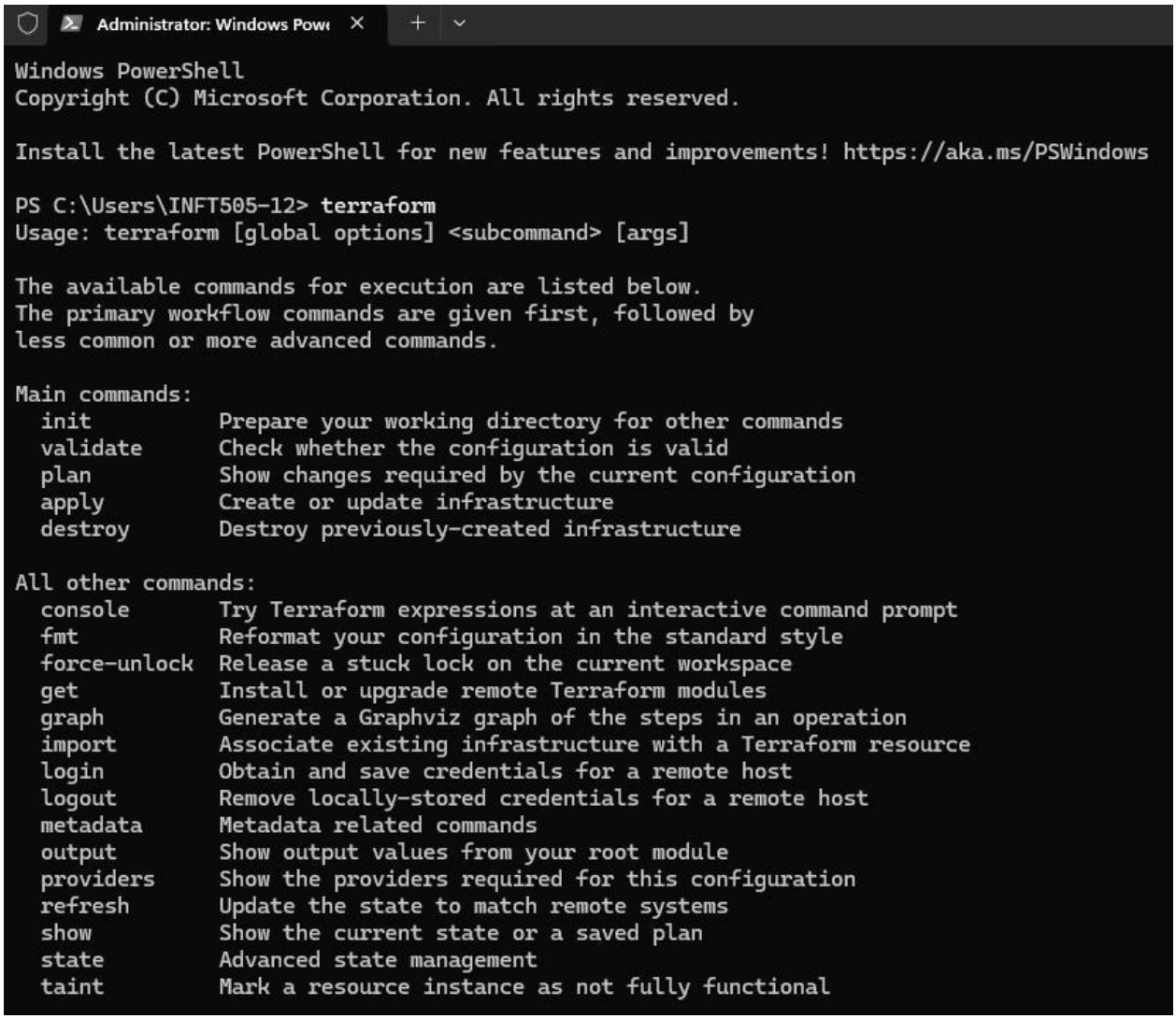
Step 3: Set the System path for Terraform in Environment Variables



Step 4: Run the windows powershell as administrator.



Step 5: Run `terraform` to verify its functionality. If you encounter any errors, double-check or update the Terraform path in your environment variables.



Experiment 06 :Creating docker image using terraform

Step 1: Download and install Docker Desktop by visiting https://www.docker.com. Run the installer and follow the prompts to complete the installation, then verify by launching Docker Desktop or using the `docker --version` command.

```
Microsoft Windows [Version 10.0.22621.3880]
(c) Microsoft Corporation. All rights reserved.

C:\Users\adity>docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Common Commands:
  run         Create and run a new container from an image
  exec        Execute a command in a running container
  ps          List containers
  build       Build an image from a Dockerfile
  pull        Download an image from a registry
  push        Upload an image to a registry
  images      List images
  login       Log in to a registry
  logout      Log out from a registry
  search      Search Docker Hub for images
  version     Show the Docker version information
  info        Display system-wide information

Management Commands:
  builder     Manage builds
  buildx*     Docker Buildx
  compose*    Docker Compose
  container   Manage containers
  context     Manage contexts
  debug*      Get a shell into any image or container
  desktop*    Docker Desktop commands (Alpha)
  dev*        Docker Dev Environments
  extension*  Manages Docker extensions
  feedback*   Provide feedback, right in your terminal!
  image       Manage images
  init*       Creates Docker-related starter files for your project
  manifest    Manage Docker image manifests and manifest lists
  network     Manage networks
  plugin      Manage plugins
  sbom*       View the packaged-based Software Bill Of Materials (SBOM) for an image
  scout*      Docker Scout
  system      Manage Docker
  trust       Manage trust on Docker images
  volume      Manage volumes
```

```
Microsoft Windows [Version 10.0.22621.3880]
(c) Microsoft Corporation. All rights reserved.

C:\Users\adity>docker --version
Docker version 27.0.3, build 7d4bcd8
```

Step 2: Now, create a folder named '**Terraform Scripts**' in which we save our different types of scripts which will be further used in this experiment.

```
C:\Users\adity\Desktop\Project 1>mkdir TerraformScripts

C:\Users\adity\Desktop\Project 1>cd TerraformScripts
```

Step 3**:** First, create a new folder named `Docker` inside the `TerraformScripts` folder. Then, open Notepad and create a new file named `docker.tf` within the `Docker` folder. Write the following contents into the `docker.tf` file to create an Ubuntu Linux container. Save the file when done.

```
C:\Users\adity\Desktop\Project 1\TerraformScripts>mkdir Docker

C:\Users\adity\Desktop\Project 1\TerraformScripts>cd Docker

C:\Users\adity\Desktop\Project 1\TerraformScripts\Docker>notepad docker.tf
```

Script:

```
terraform {
  required_providers {
    docker = {
      source = "kreuzwerker/docker"
      version = "2.21.0"
    }
  }
}

provider "docker" {
  host = "npipe:////./pipe/docker_engine"
}

# Pulls the image
resource "docker_image" "ubuntu" {
  name = "ubuntu:latest"
}

# Create a container
resource "docker_container" "foo" {
  image   = docker_image.ubuntu.image_id
  name    = "foo"
  command = ["sleep", "infinity"]
}
```

This Terraform script configures the Docker provider to communicate with the Docker Engine using a Windows named pipe.
It pulls the latest Ubuntu image from Docker Hub and creates a container named "foo." The container runs the `sleep infinity` command, which keeps it active indefinitely.

This setup is useful for scenarios where the container needs to remain running continuously.

docker.tf

File Edit View

```
terraform {
  required_providers {
    docker = {
      source  = "kreuzwerker/docker"
      version = "2.21.0"
    }
  }
}

provider "docker" {
  host = "npipe:////./pipe/docker_engine"
}

# Pulls the image
resource "docker_image" "ubuntu" {
  name = "ubuntu:latest"
}

# Create a container
resource "docker_container" "foo" {
  image   = docker_image.ubuntu.image_id
  name    = "foo"
  command = ["sleep", "infinity"]
}
```

Step 4: Execute the `terraform init` command to initialize the working directory, download the necessary provider plugins, and set up the backend for managing Terraform state.

```
C:\Users\adity\Desktop\Project 1\TerraformScripts\Docker>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding kreuzwerker/docker versions matching "2.21.0"...
- Installing kreuzwerker/docker v2.21.0...
- Installed kreuzwerker/docker v2.21.0 (self-signed, key ID BD080C4571C6104C)
Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

Step 5: Run `**terraform plan**` to preview the actions Terraform will take to reach the desired state defined in your configuration, including creating, modifying, or deleting resources.

```
C:\Users\adity\Desktop\Project 1\TerraformScripts\Docker>terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  + create

Terraform will perform the following actions:

  # docker_container.foo will be created
  + resource "docker_container" "foo" {
      + attach           = false
      + bridge           = (known after apply)
      + command          = (known after apply)
      + container_logs   = (known after apply)
      + entrypoint       = (known after apply)
      + env              = (known after apply)
      + exit_code        = (known after apply)
      + gateway          = (known after apply)
      + hostname         = (known after apply)
      + id               = (known after apply)
      + image            = (known after apply)
      + init             = (known after apply)
      + ip_address       = (known after apply)
      + ip_prefix_length = (known after apply)
      + ipc_mode         = (known after apply)
      + log_driver       = (known after apply)
      + logs             = false
      + must_run         = true
      + name             = "foo"
      + network_data     = (known after apply)
      + read_only        = false
      + remove_volumes   = true
      + restart          = "no"
      + rm               = false
      + runtime          = (known after apply)
      + security_opts    = (known after apply)
      + shm_size         = (known after apply)
      + start            = true
      + stdin_open       = false
      + stop_signal      = (known after apply)
      + stop_timeout     = (known after apply)
      + tty              = false

      + healthcheck (known after apply)

      + labels (known after apply)
    }

  # docker_image.ubuntu will be created
  + resource "docker_image" "ubuntu" {
      + id          = (known after apply)
      + image_id    = (known after apply)
      + latest      = (known after apply)
      + name        = "ubuntu:latest"
      + output      = (known after apply)
      + repo_digest = (known after apply)
    }

Plan: 2 to add, 0 to change, 0 to destroy.

---

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if
you run "terraform apply" now.
```

Step 6: Execute "**terraform apply**" to apply the configuration, which will automatically create and run the Ubuntu container based on our configuration.

```
C:\Users\adity\Desktop\Project 1\TerraformScripts\Docker>terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  + create

Terraform will perform the following actions:

  # docker_container.foo will be created
  + resource "docker_container" "foo" {
      + attach           = false
      + bridge           = (known after apply)
      + command          = (known after apply)
      + container_logs   = (known after apply)
      + entrypoint       = (known after apply)
      + env              = (known after apply)
      + exit_code        = (known after apply)
      + gateway          = (known after apply)
      + hostname         = (known after apply)
      + id               = (known after apply)
      + image            = (known after apply)
      + init             = (known after apply)
      + ip_address       = (known after apply)
      + ip_prefix_length = (known after apply)
      + ipc_mode         = (known after apply)
      + log_driver       = (known after apply)
      + logs             = false
      + must_run         = true
      + name             = "foo"
      + network_data     = (known after apply)
      + read_only        = false
      + remove_volumes   = true
      + restart          = "no"
      + rm               = false
      + runtime          = (known after apply)
      + security_opts    = (known after apply)
      + shm_size         = (known after apply)
      + start            = true
      + stdin_open       = false
      + stop_signal      = (known after apply)
      + stop_timeout     = (known after apply)
      + tty              = false

      + healthcheck (known after apply)
      + labels (known after apply)
    }

  # docker_image.ubuntu will be created
  + resource "docker_image" "ubuntu" {
      + id          = (known after apply)
      + image_id    = (known after apply)
      + latest      = (known after apply)
      + name        = "ubuntu:latest"
      + output      = (known after apply)
      + repo_digest = (known after apply)
    }

Plan: 2 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

docker_image.ubuntu: Creating...
docker_image.ubuntu: Creation complete after 8s [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Creating...
```

Step 7: The command `**docker images**` lists all Docker images stored locally on your system, showing details like repository names, tags, image IDs, and creation dates.

Docker images, Before Executing Apply step:

```
C:\Users\adity\Desktop\Project 1\TerraformScripts\Docker>docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```

Docker images, After Executing Apply step:

```
C:\Users\adity\Desktop\Project 1\TerraformScripts\Docker>docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    edbfe74c41f8   2 weeks ago   78.1MB
```

Step 8: Execute Terraform destroy to delete the configuration, which will automatically delete the Ubuntu Container.

```
C:\Users\adity\Desktop\Project 1\TerraformScripts\Docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  - destroy

Terraform will perform the following actions:

  # docker_image.ubuntu will be destroyed
  - resource "docker_image" "ubuntu" {
      - id          = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest" -> null
      - image_id    = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
      - latest      = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
      - name        = "ubuntu:latest" -> null
      - repo_digest = "ubuntu@sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e34a9ab63ee" -> null
    }

Plan: 0 to add, 0 to change, 1 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image.ubuntu: Destruction complete after 0s

Destroy complete! Resources: 1 destroyed.
```

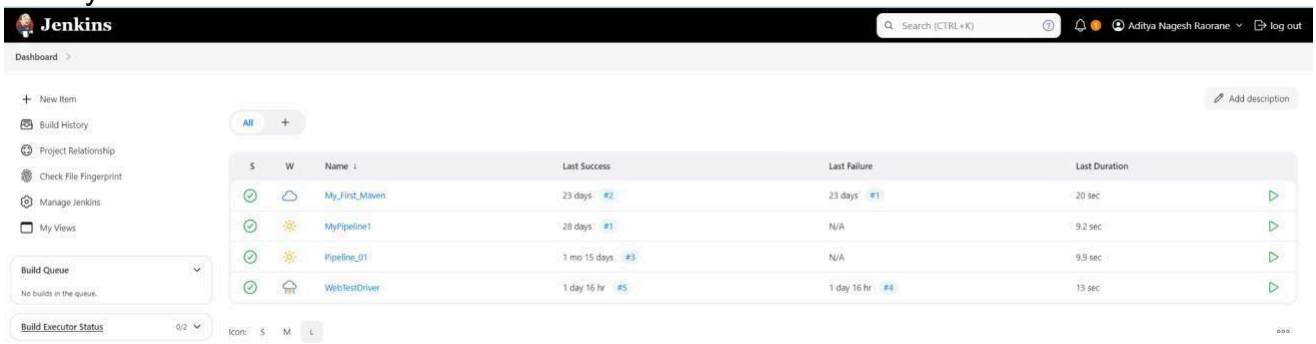
Step 9: Docker images After Executing Destroy step

```
C:\Users\adity\Desktop\Project 1\TerraformScripts\Docker>docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```

Experiment 07

Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

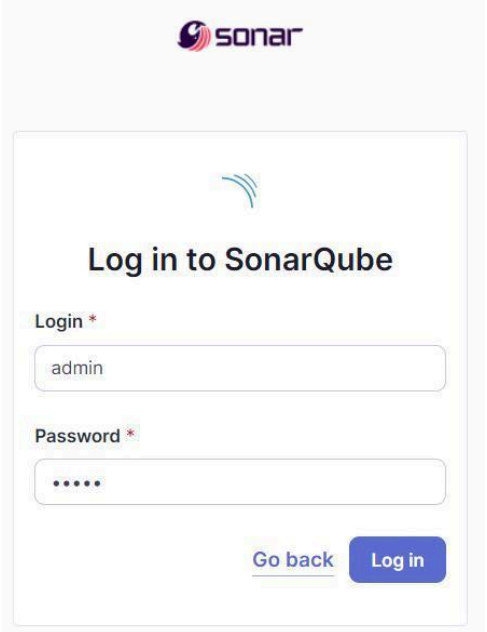
1. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.



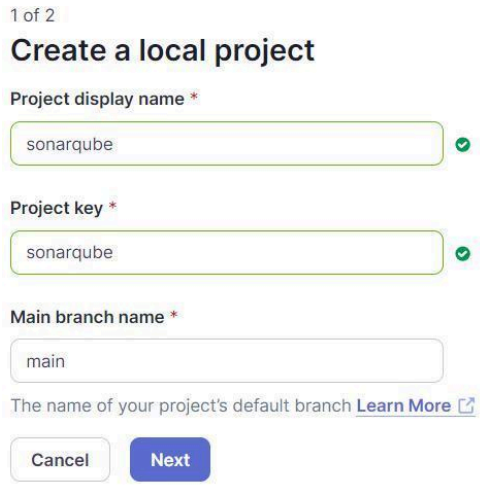
2. Run SonarQube in a Docker container using this command :-a] docker -v
b] docker pull sonarqube
c] docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

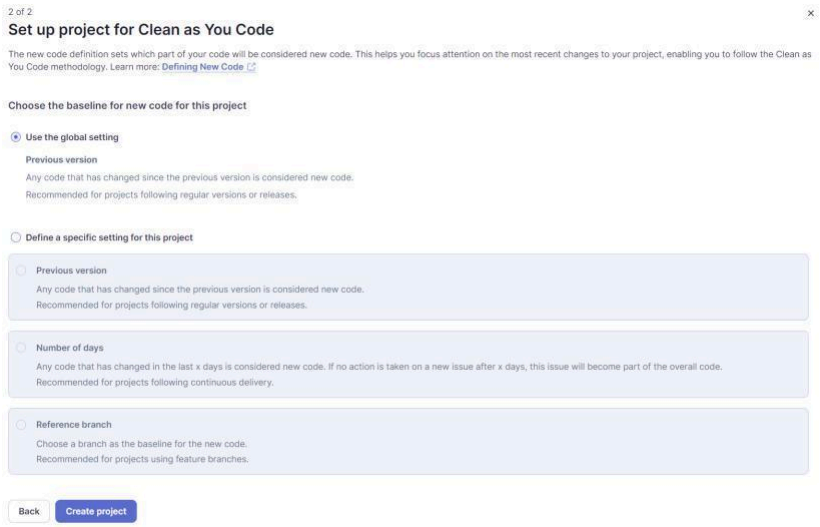
```
C:\Users\adity>docker -v
Docker version 27.0.3, build 7d4bcd8

C:\Users\adity>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
4a6e73f4472de892b1ddead1abe77372a85a7b09408cce3a0abd37c5ab6b49a4
```

3. Once the container is up and running, you can check the status of SonarQube at **localhost port 9000**. The login id is “**admin**” and the password is “**aditya**”.



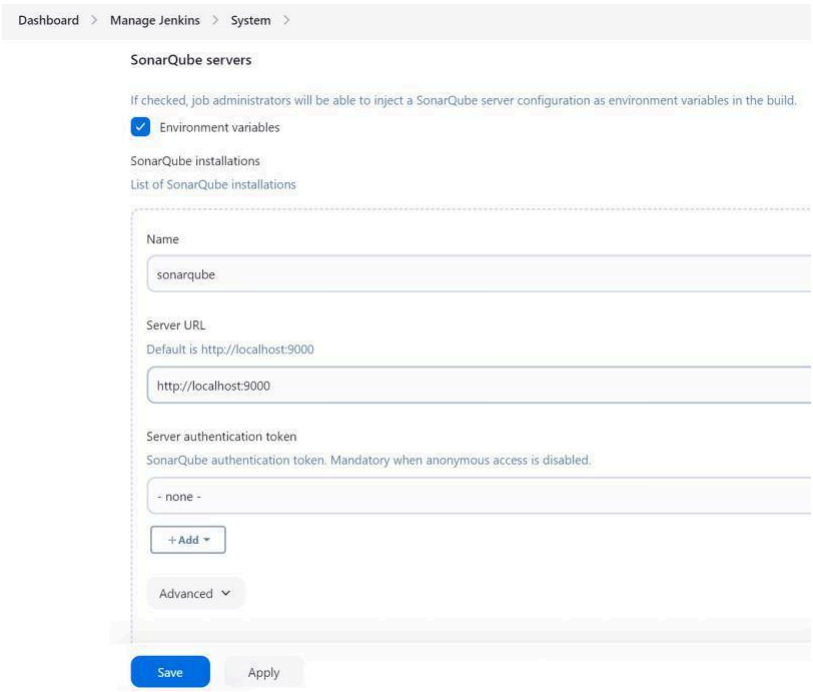
4. **Create a local project in SonarQube** with the name **sonarqube**





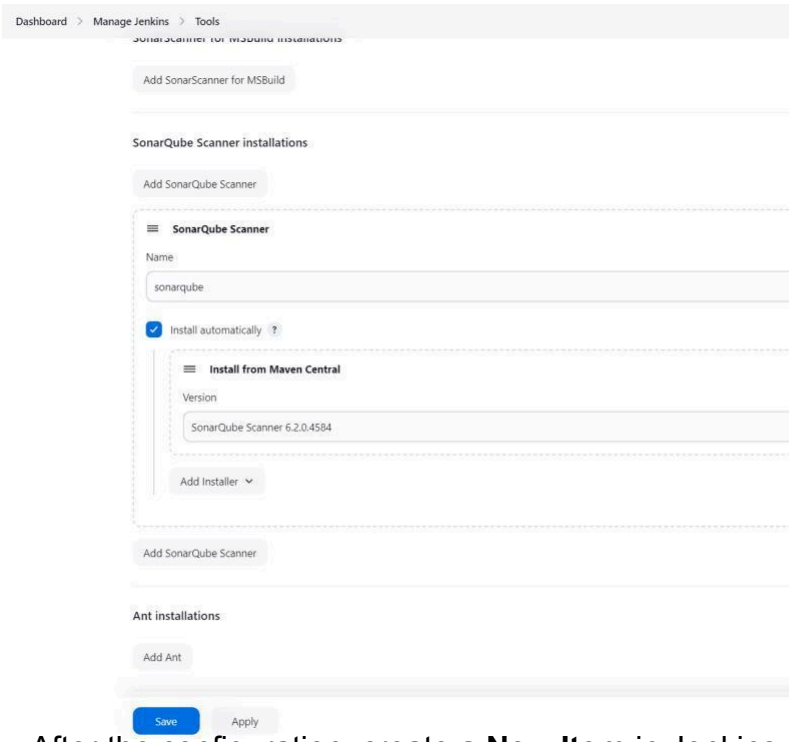
5. Setup the project and come back to Jenkins Dashboard. Go to **Manage Jenkins** → **Plugins** and search for **SonarQube Scanner** in **Available Plugins** and install it.

6. Under '**Manage Jenkins** → **System**', look for **SonarQube Servers** and enter these details.
Name : sonarqube
Server URL : http://localhost:9000

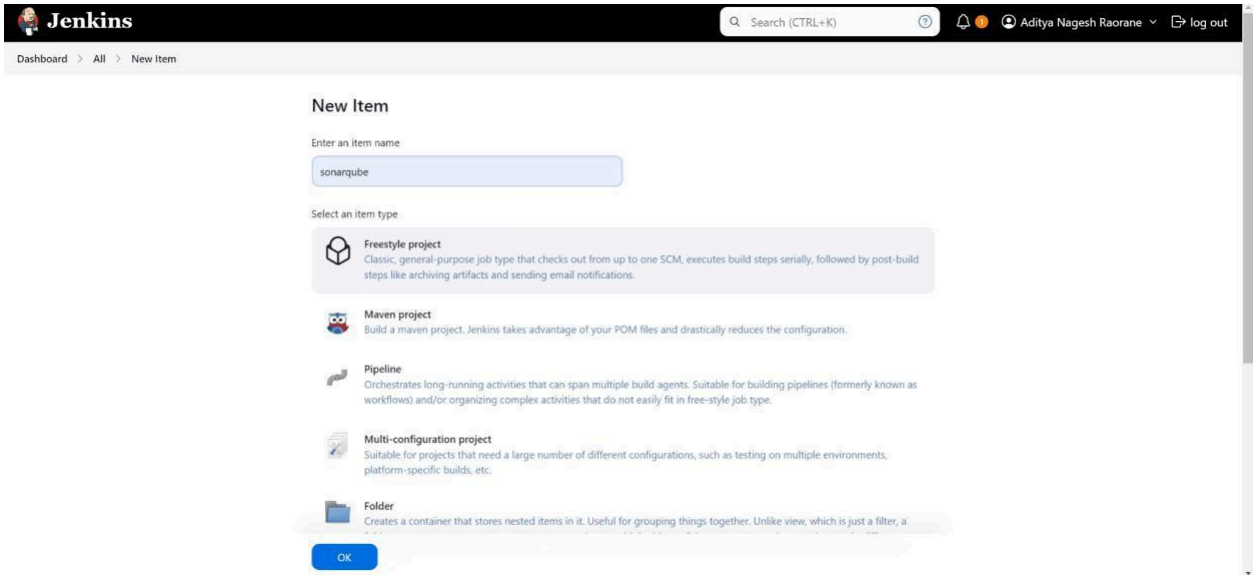


7. Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.

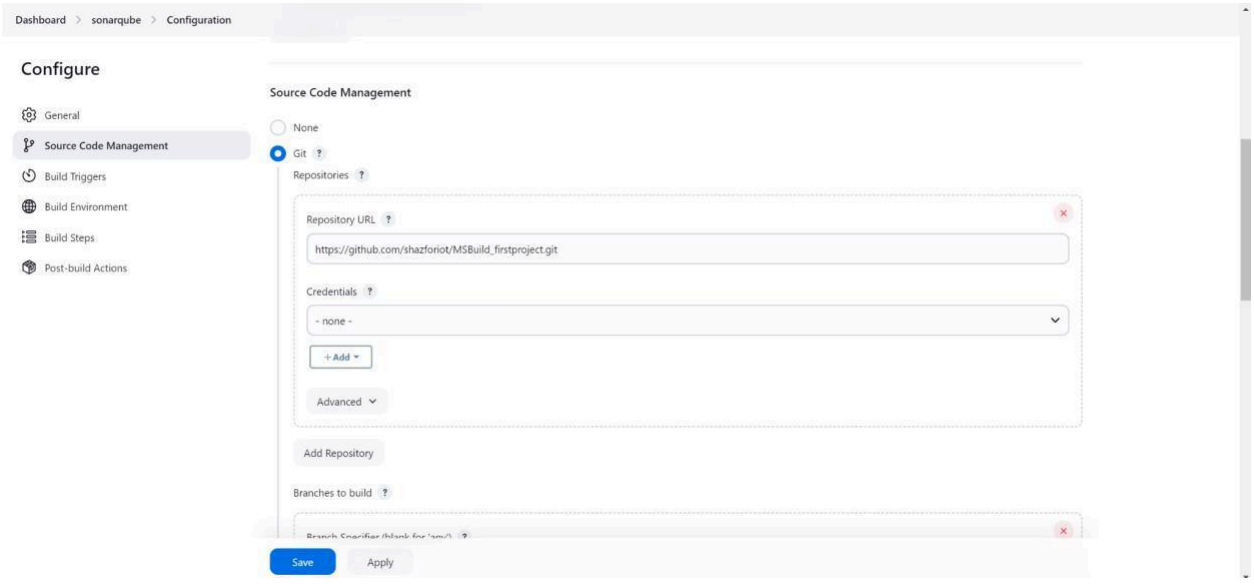
Manage Jeknins → Tools → SonarQube Scanner Installation



8. After the configuration, create a **New Item** in Jenkins, choose a **freestyle project** named **sonarqube**.



9. Choose this GitHub repository in **Source Code Management**. https://github.com/shazforiot/MSBuild_firstproject.git
It is a sample hello-world project with no vulnerabilities and issues, just to test the integration.



10. Under **Build-> Execute SonarQube Scanner**, enter these **Analysis Properties**. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.

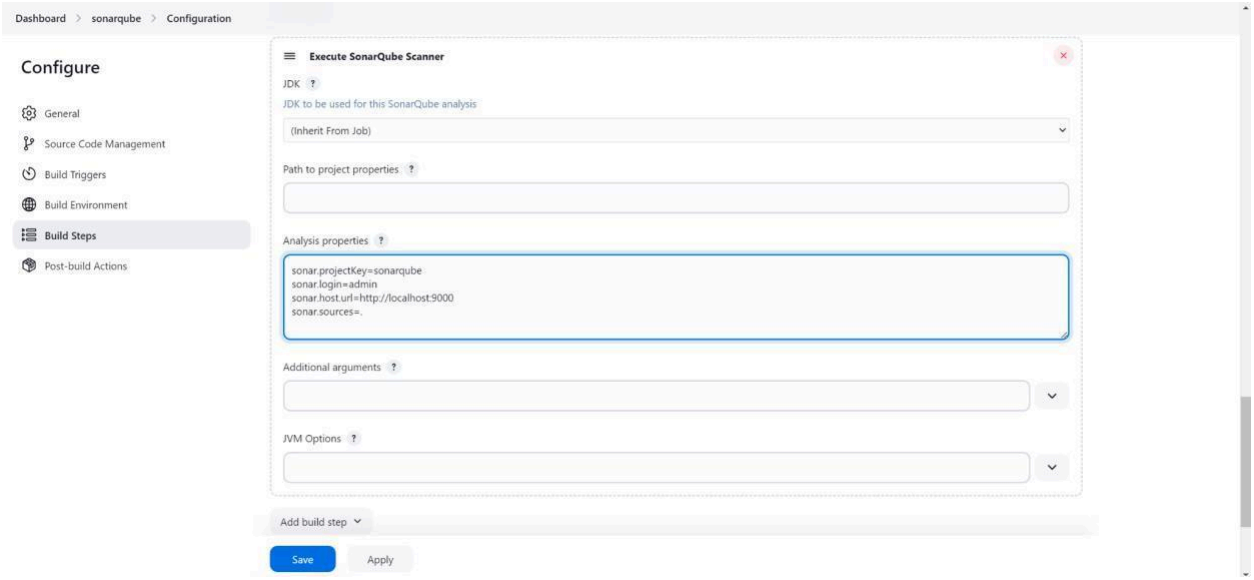
sonar.projectKey=sonarqube

sonar.login=admin

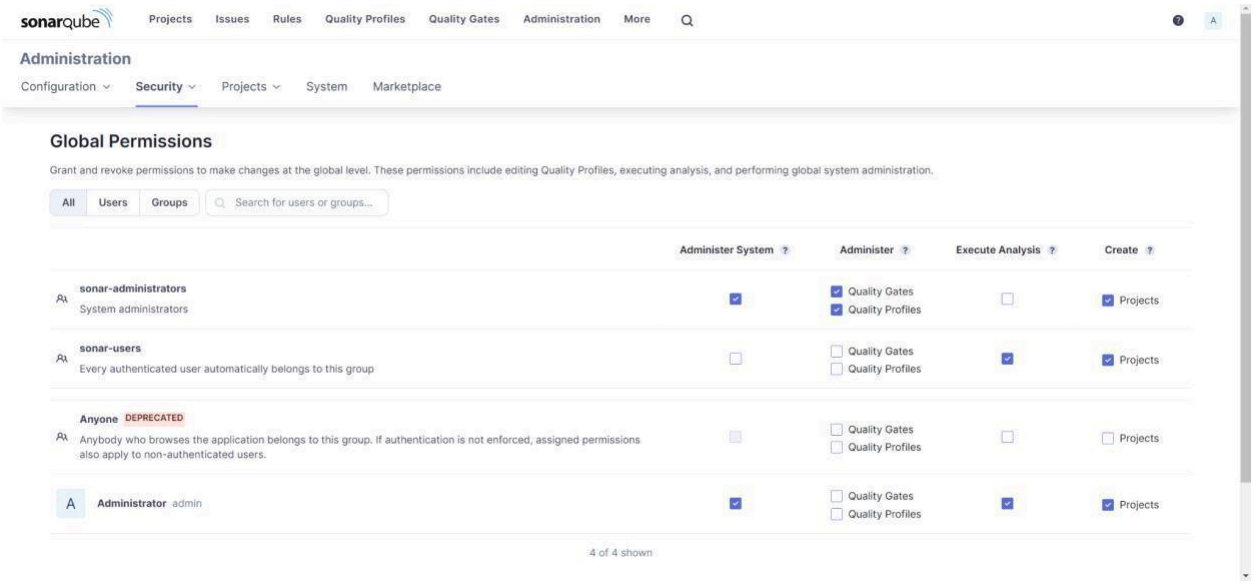
sonar.password=aditya

sonar.sources=.

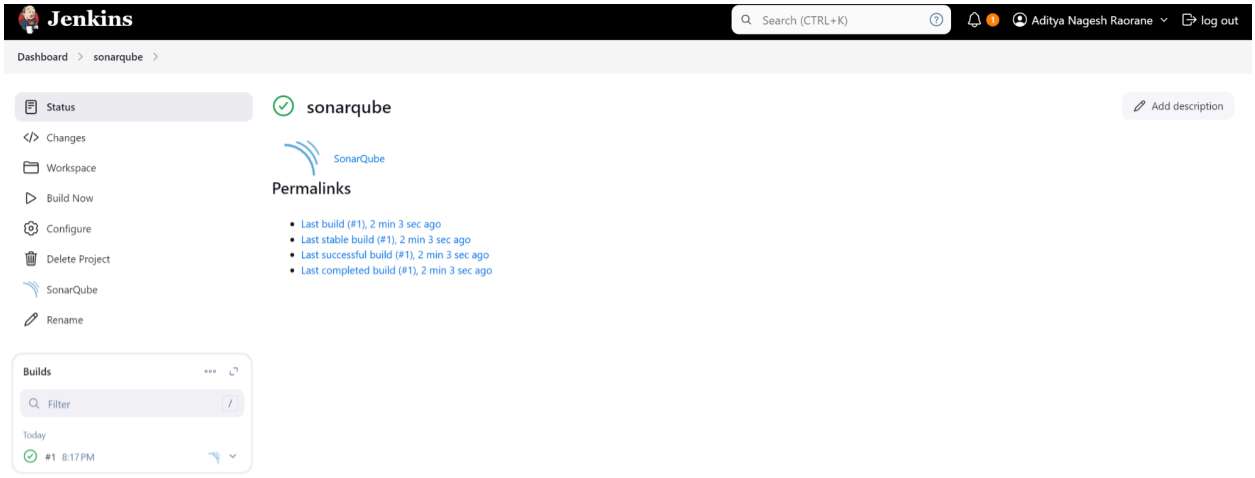
sonar.host.url=http://localhost:9000

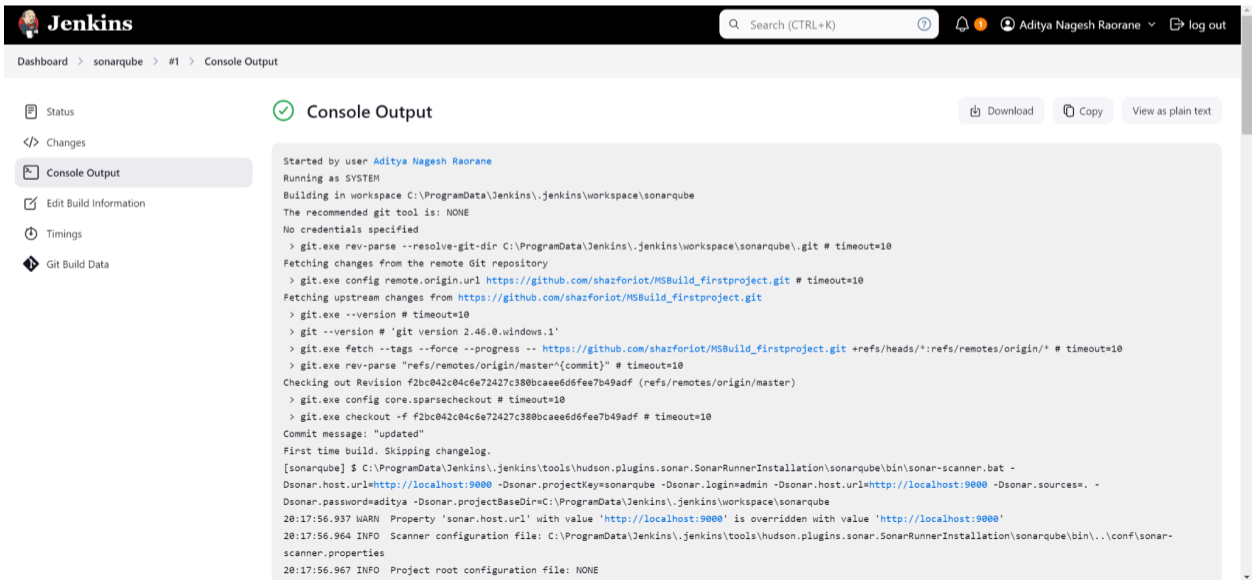


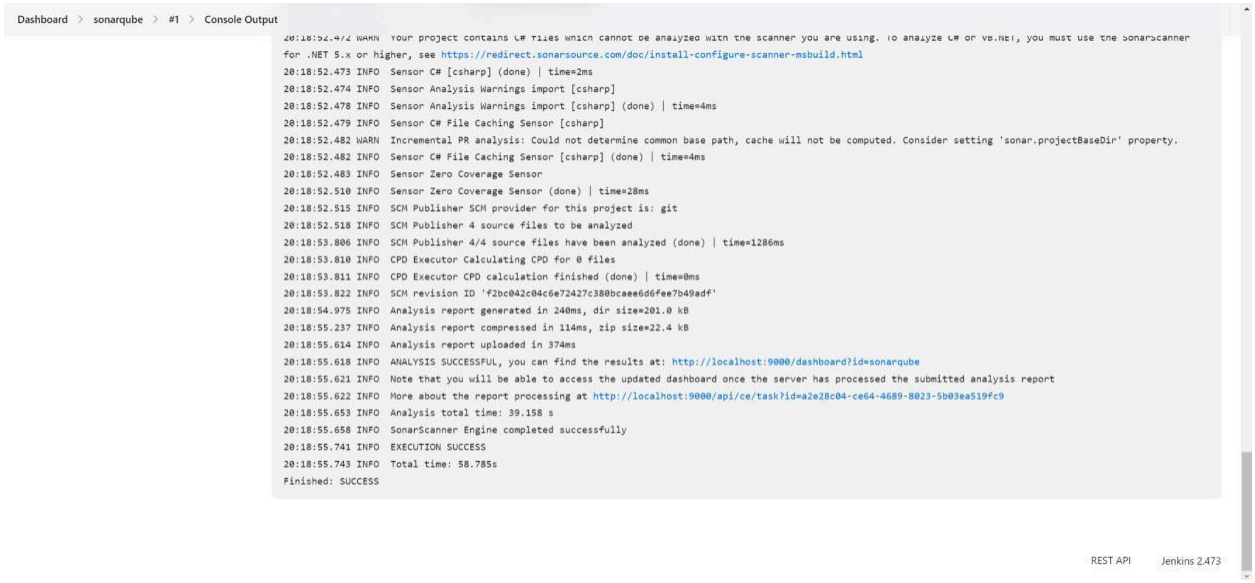
11. Go to http://localhost:9000/admin/permissions and allow Execute Permissions to the Admin user.



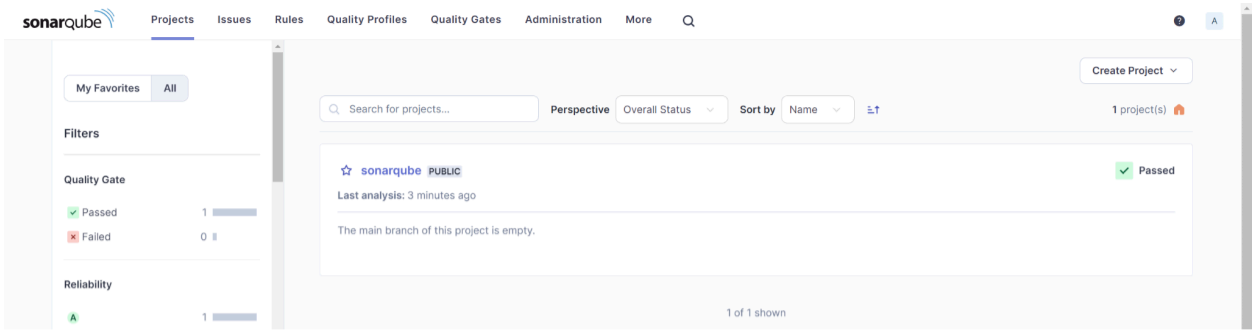
12. Run The **Build** and check the **console output**.

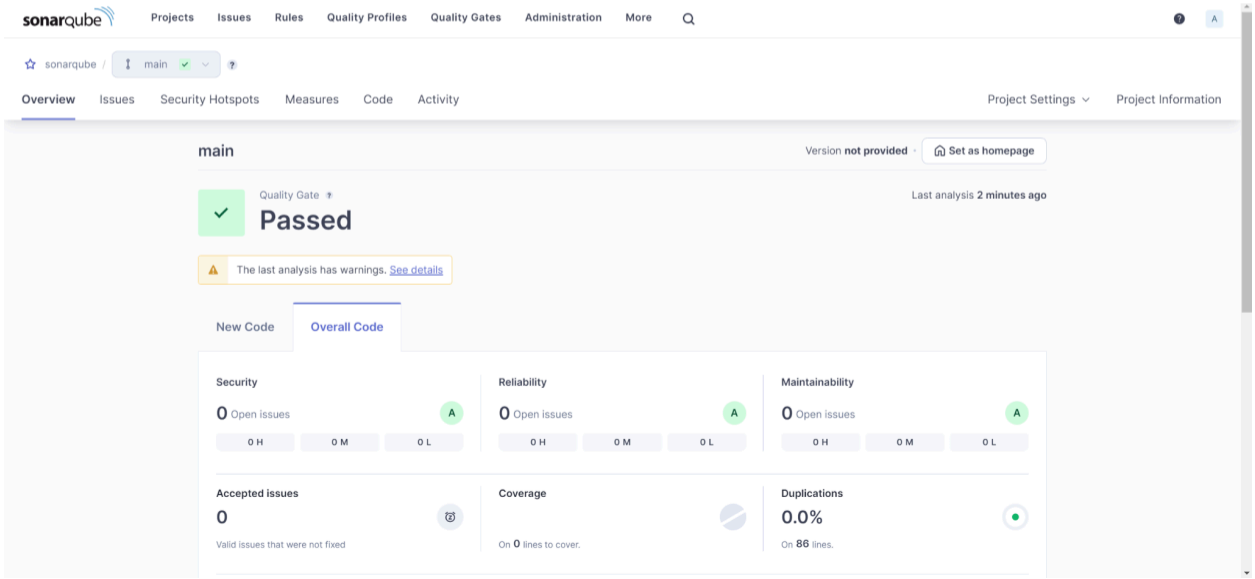






13. Once the build is complete, check the project in SonarQube.





In this way, we have integrated Jenkins with SonarQube for SAST.

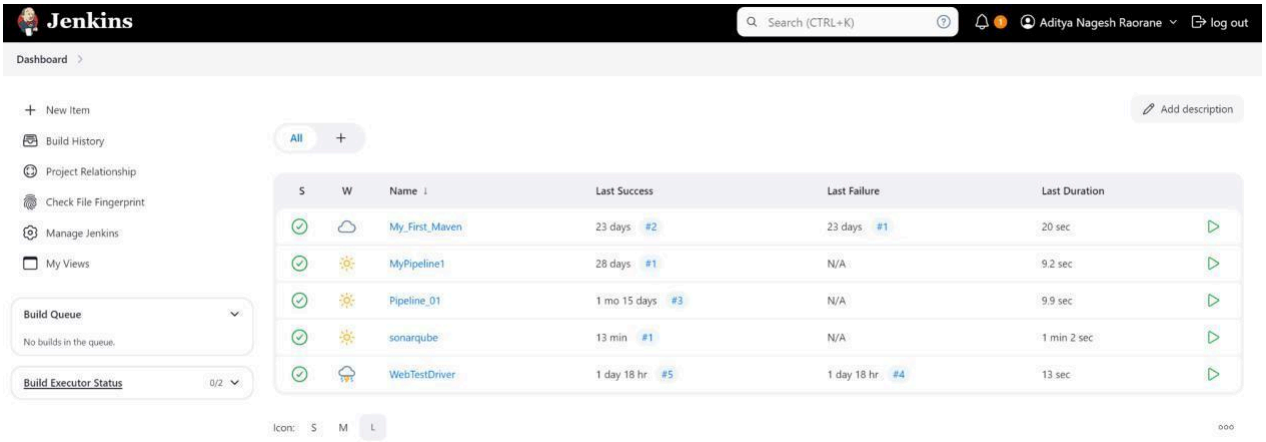
Conclusion:

In this experiment, we have understood the importance of SAST and have successfully integrated Jenkins with SonarQube for Static Analysis and Code Testing.

Experiment 08

Aim: Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web / Java / Python application.

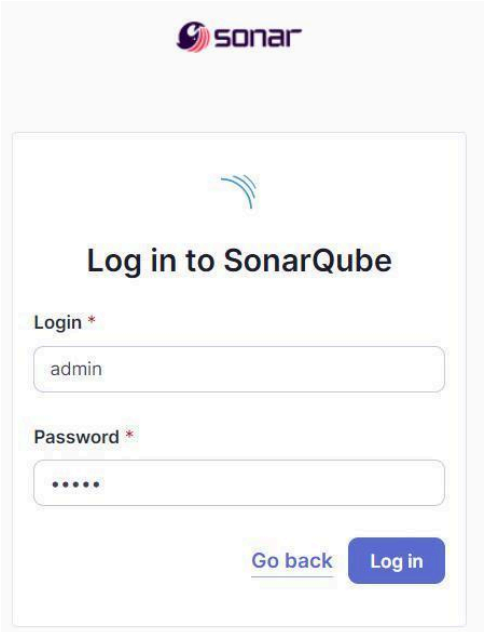
1. Open up Jenkins Dashboard on localhost:8080.



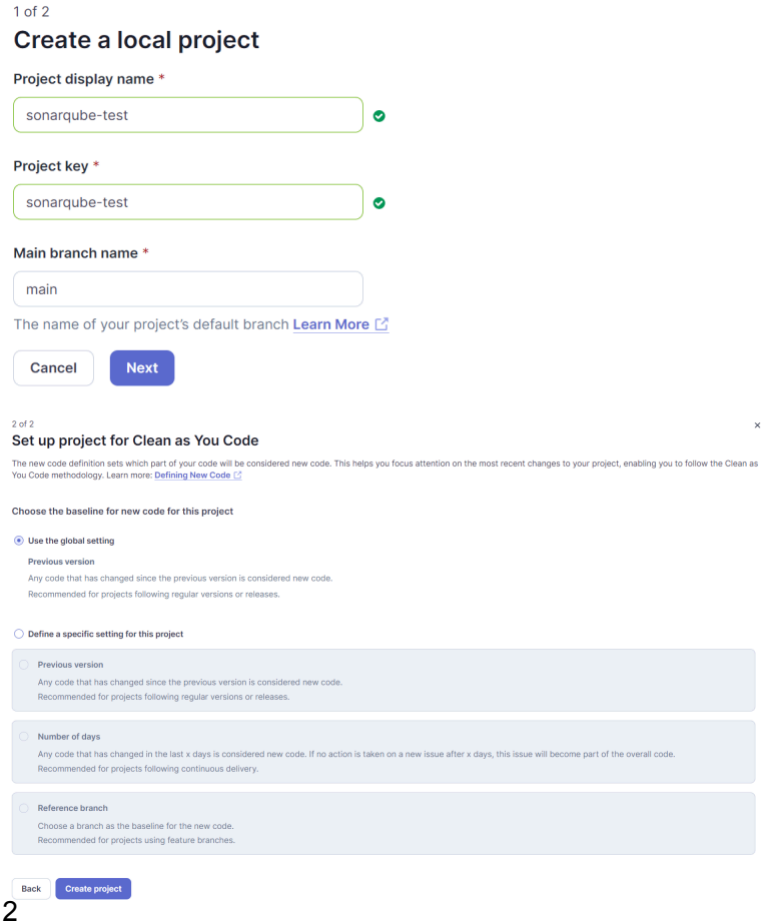
2. Run SonarQube in a Docker container using this command: a] docker -v
b] docker pull sonarqube
c] docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

```
C:\Users\adity>docker -v
Docker version 27.0.3, build 7d4bcd8

C:\Users\adity>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
4a6e73f4472de892b1ddead1abe77372a85a7b09408cce3a0abd37c5ab6b49a4
```

3. Once the container is up and running, you can check the status of SonarQube at **localhost port 9000**. The login id is “**admin**” and the password is “**aditya**”.

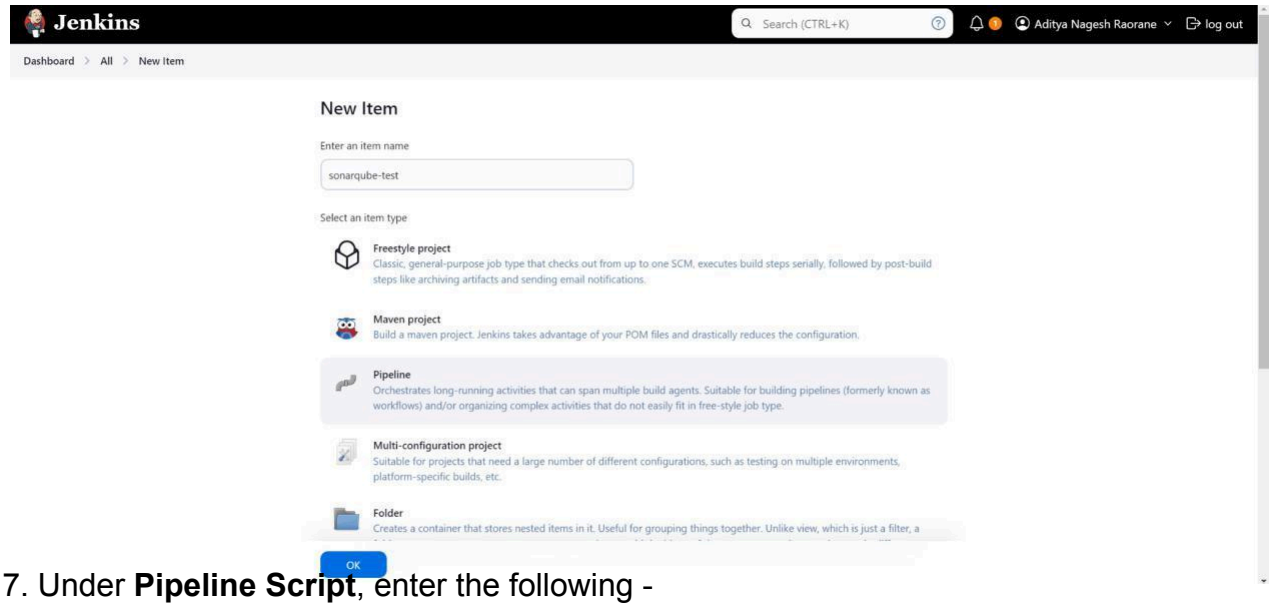


4. **Create a local project in SonarQube** with the name **sonarqube-test**.



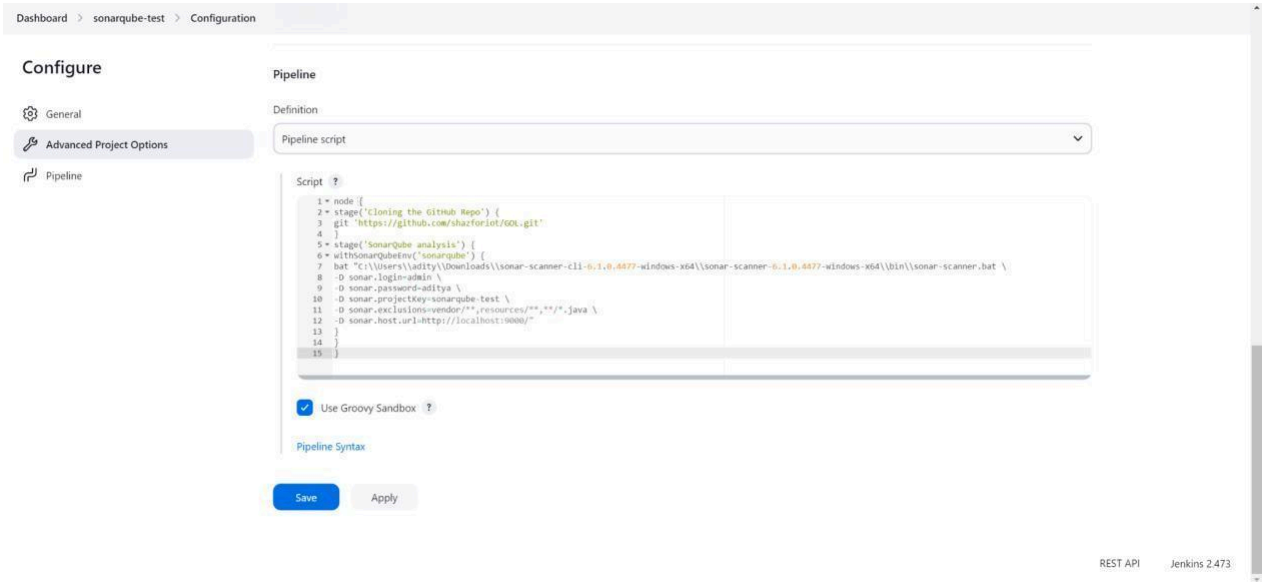
Setup the project and come back to Jenkins Dashboard.

6. Create a New Item in Jenkins, choose **Pipeline**.



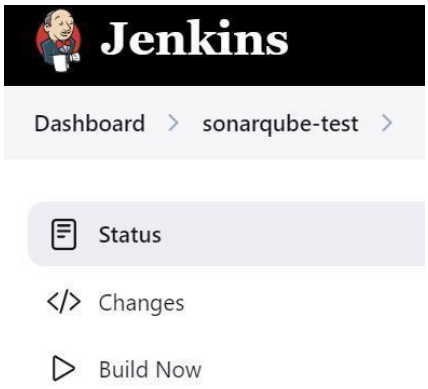
7. Under **Pipeline Script**, enter the following -

```
node {
        stage('Cloning the GitHub Repo')
        {
                git 'https://github.com/shazforiot/GOL.git'
        }
stage('SonarQube analysis') {
        withSonarQubeEnv('sonarqube') {
                bat
        "C:\\Users\\adity\\Downloads\\sonar-scanner-cli-6.1.0.4477-windows-x64\\sonar-s
        canner-6.1.0.4477-windows-x64\\bin\\sonar-scanner.bat \
                -D sonar.login=<YOUR ID> \
                -D sonar.password=<YOUR PASSWORD> \
                -D sonar.projectKey=<YOUR PROJECT KEY> \
                -D sonar.exclusions=vendor/**,resources/**,**/*.java \
                -D sonar.host.url=http://localhost:9000/" }

        }
}
```

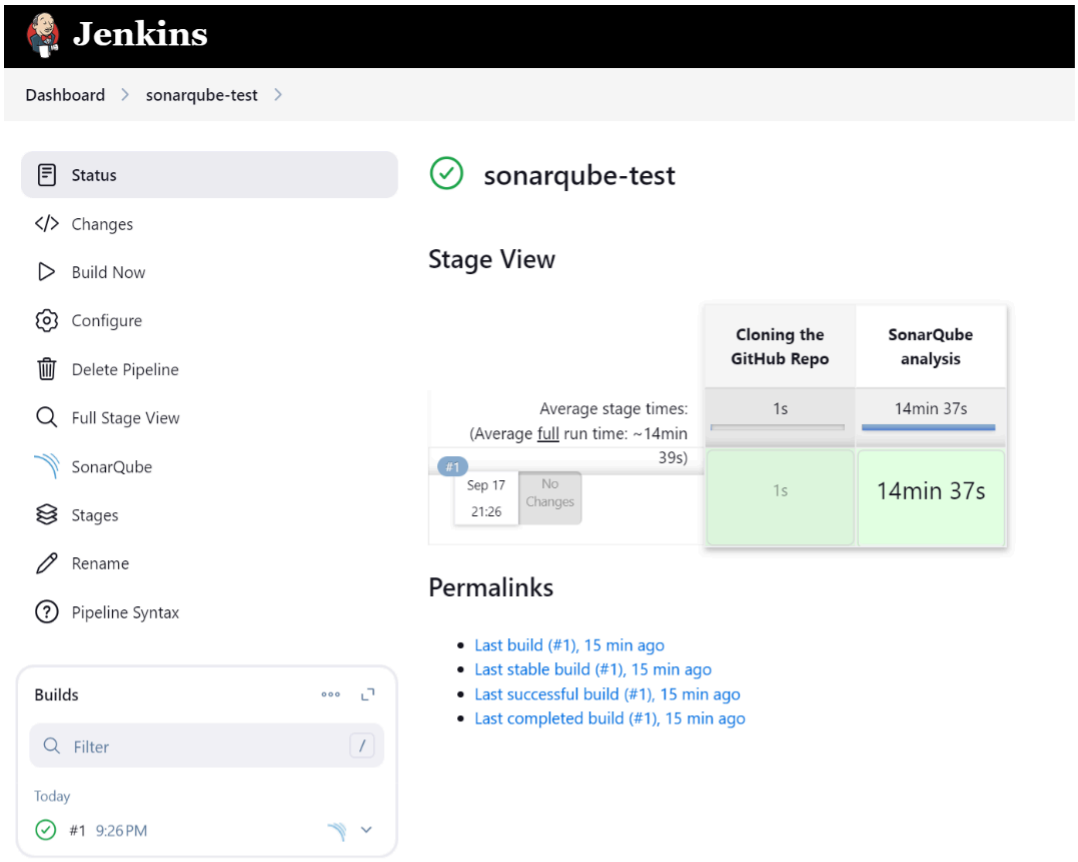


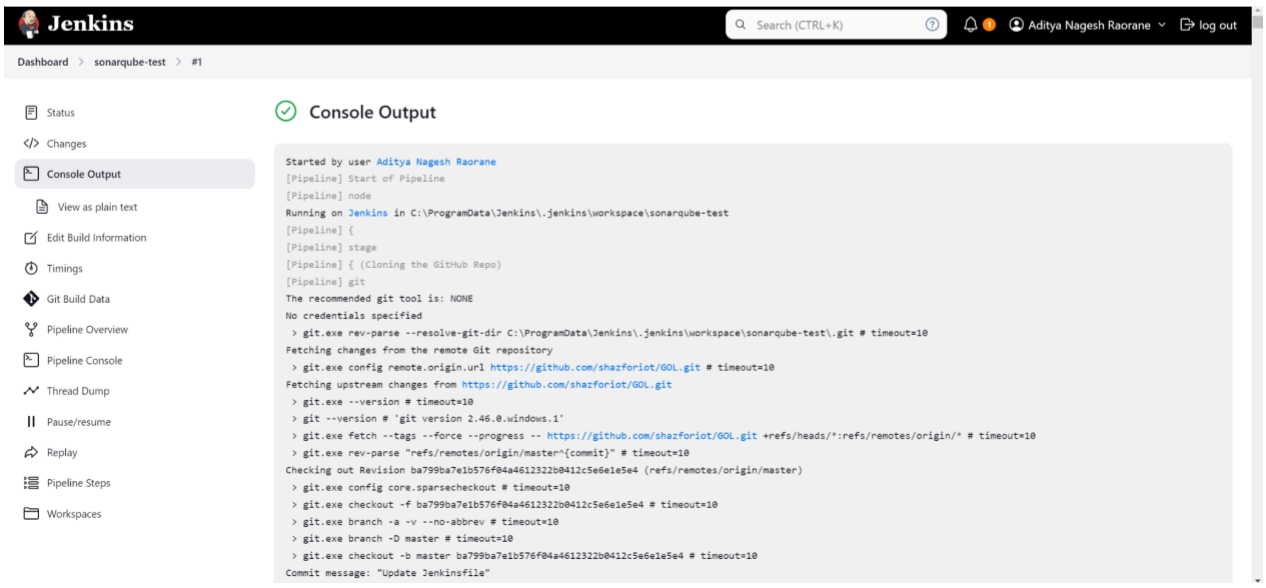
It is a java sample project which has a lot of repetitions and issues that will be detected by SonarQube.

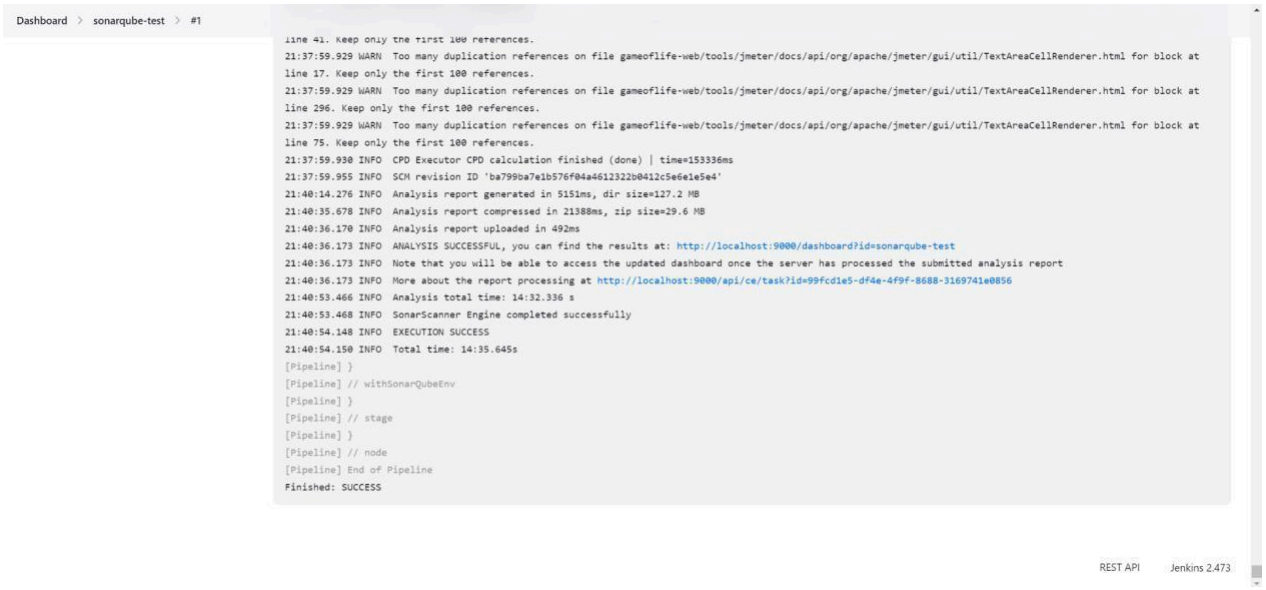
8. Run The Build.



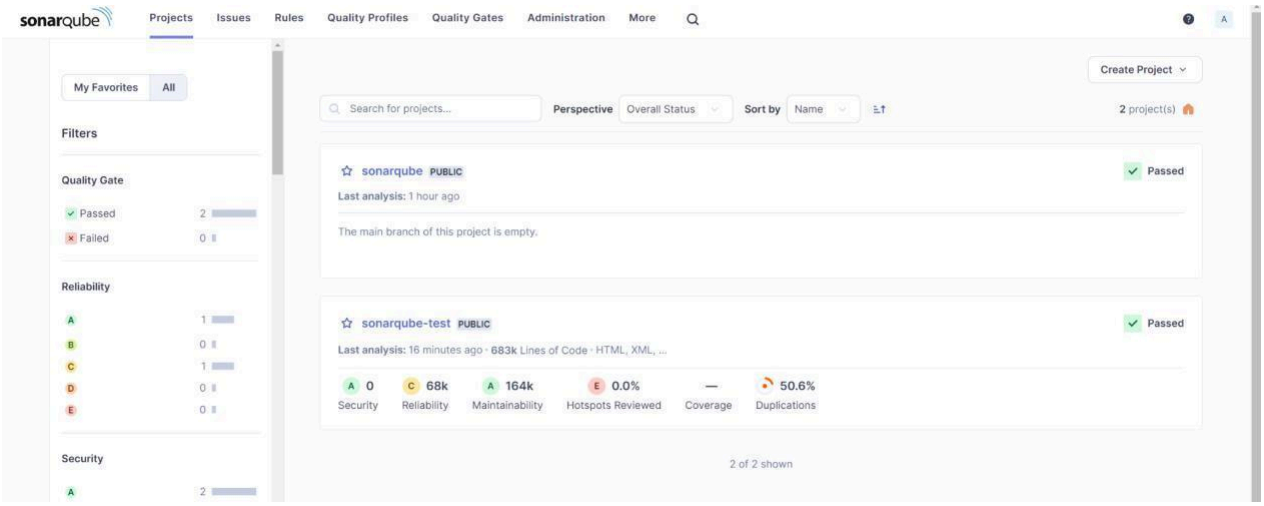
9. Check the console output once the build is complete.

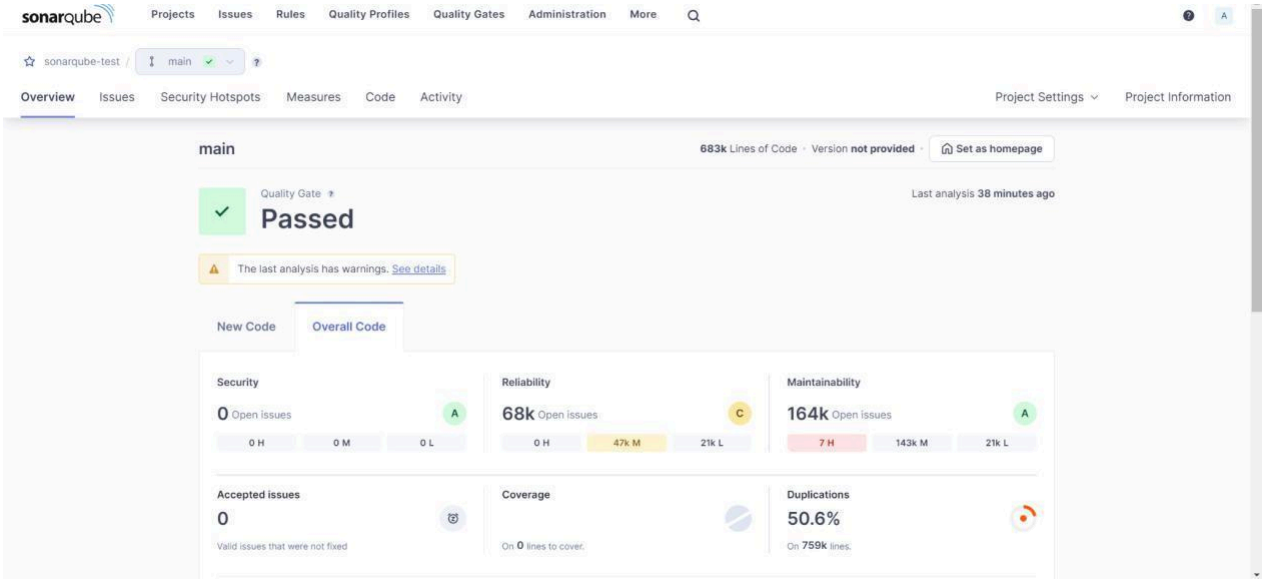






10. After that, check the project in SonarQube.

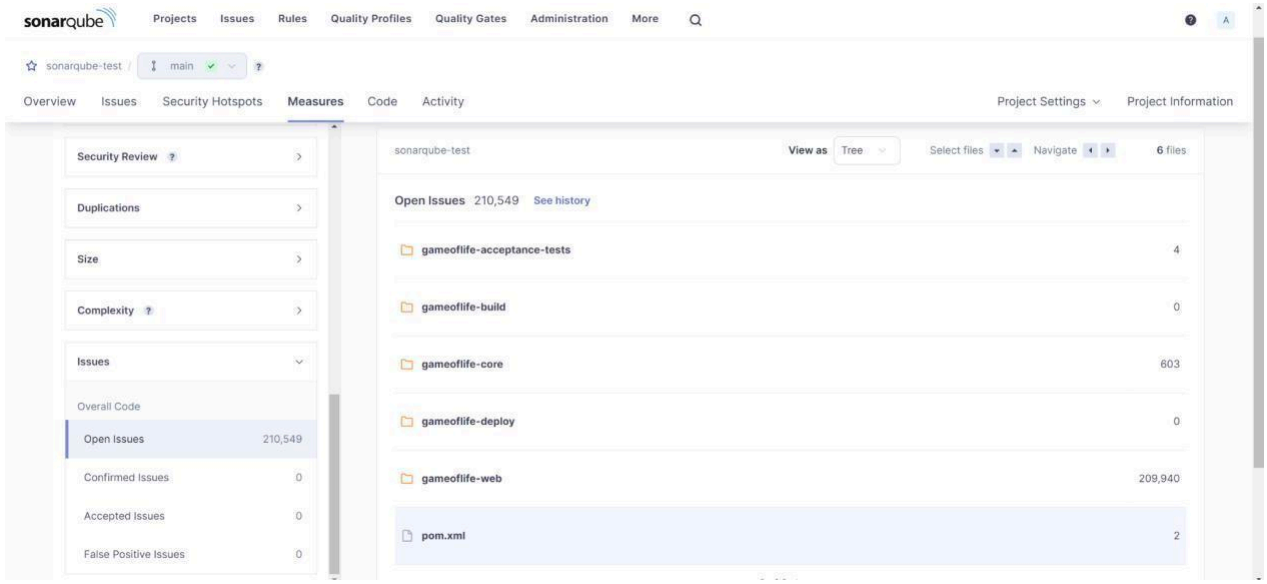




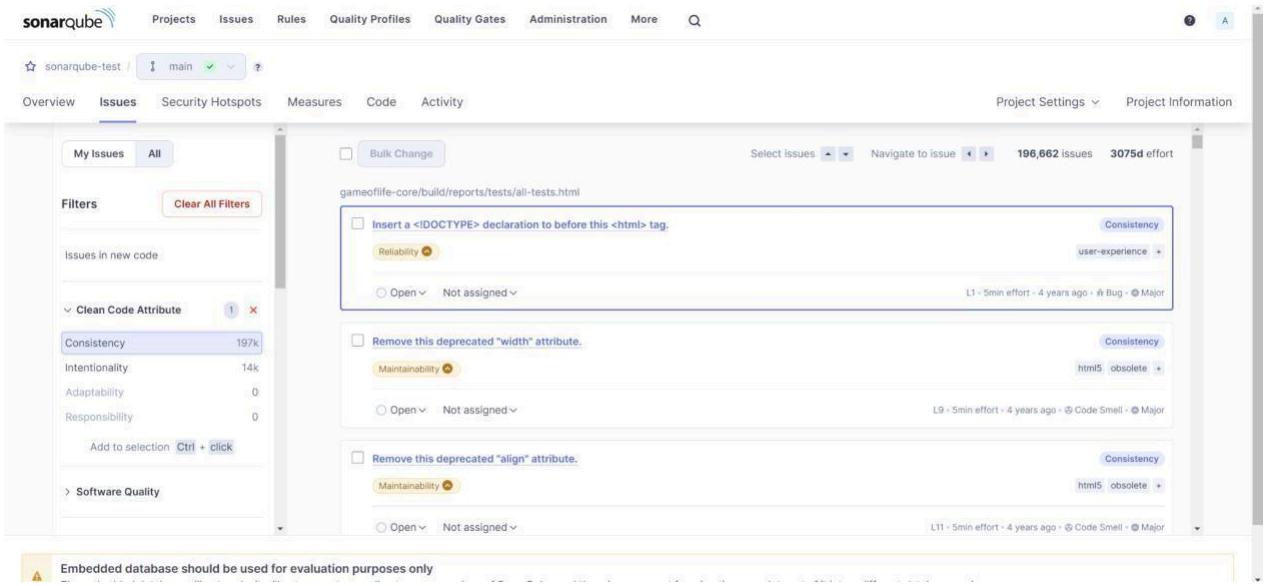
Under different tabs, check all different issues with the code.

11. **Code**

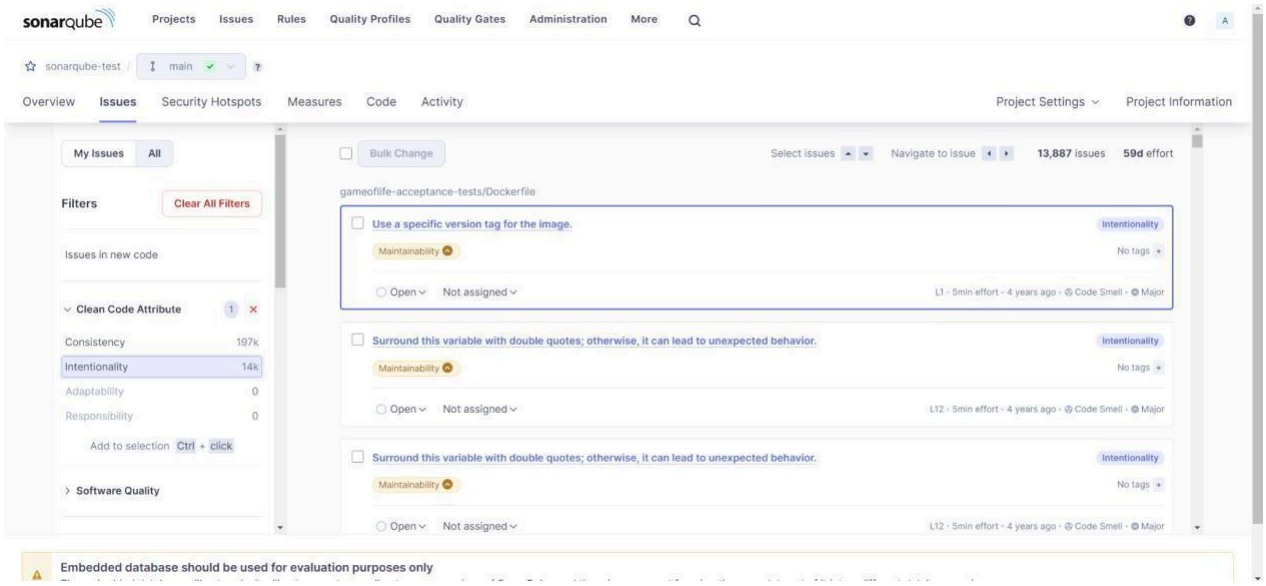
Problems - Open Issues



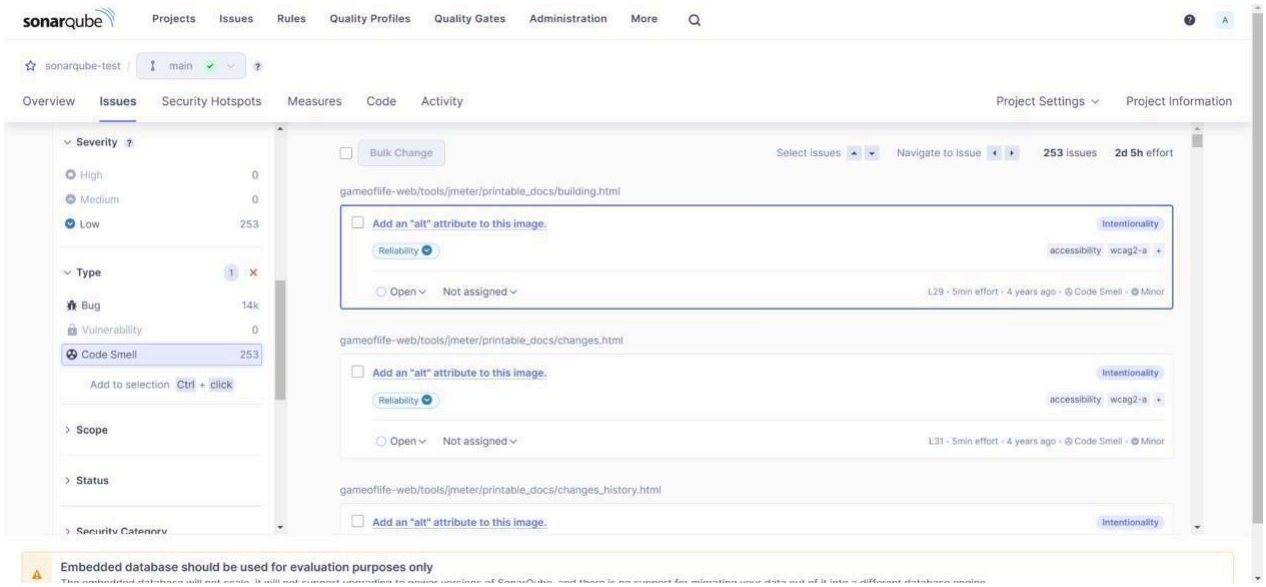
Consistency



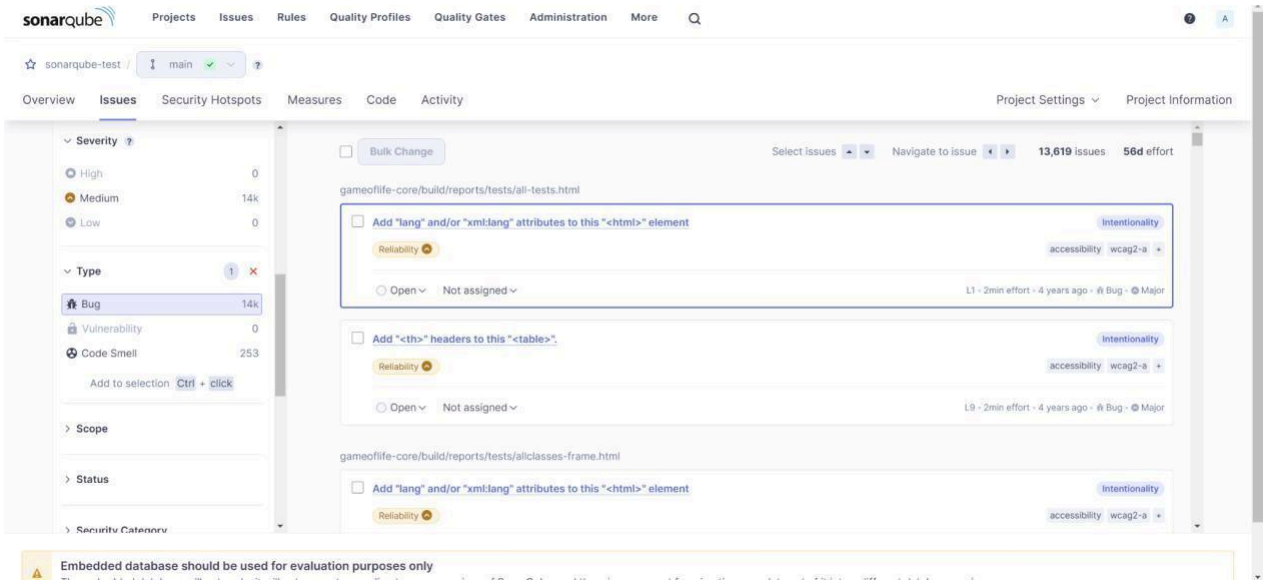
Intentionality



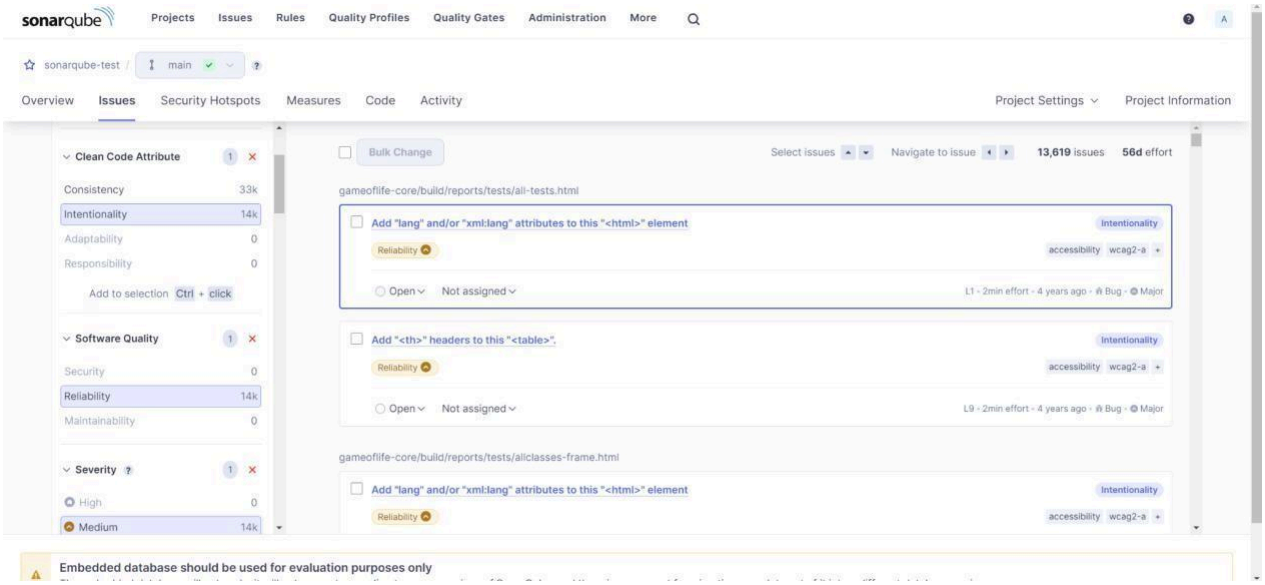
Code Smells


sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
sonarqube-test
main
Overview
Issues
Security Hotspots
Measures
Code
Activity
Project Settings
Project Information
Severity
High 0
Medium 0
Low 253
Type
Bug 14k
Vulnerability 0
Code Smell 253
Add to selection Ctrl + click
Scope
Status
Security Category
Bulk Change
Select issues
Navigate to issue
253 issues
2d 5h effort
gameoflife-web/tools/jmeter/printable_docs/building.html
Add an "alt" attribute to this image.
Intentionality
Reliability
accessibility wcag2-a
Open
Not assigned
L29 · 5min effort · 4 years ago · Code Smell · Minor
gameoflife-web/tools/jmeter/printable_docs/changes.html
Add an "alt" attribute to this image.
Intentionality
Reliability
accessibility wcag2-a
Open
Not assigned
L31 · 5min effort · 4 years ago · Code Smell · Minor
gameoflife-web/tools/jmeter/printable_docs/changes_history.html
Add an "alt" attribute to this image.
Intentionality
Embedded database should be used for evaluation purposes only

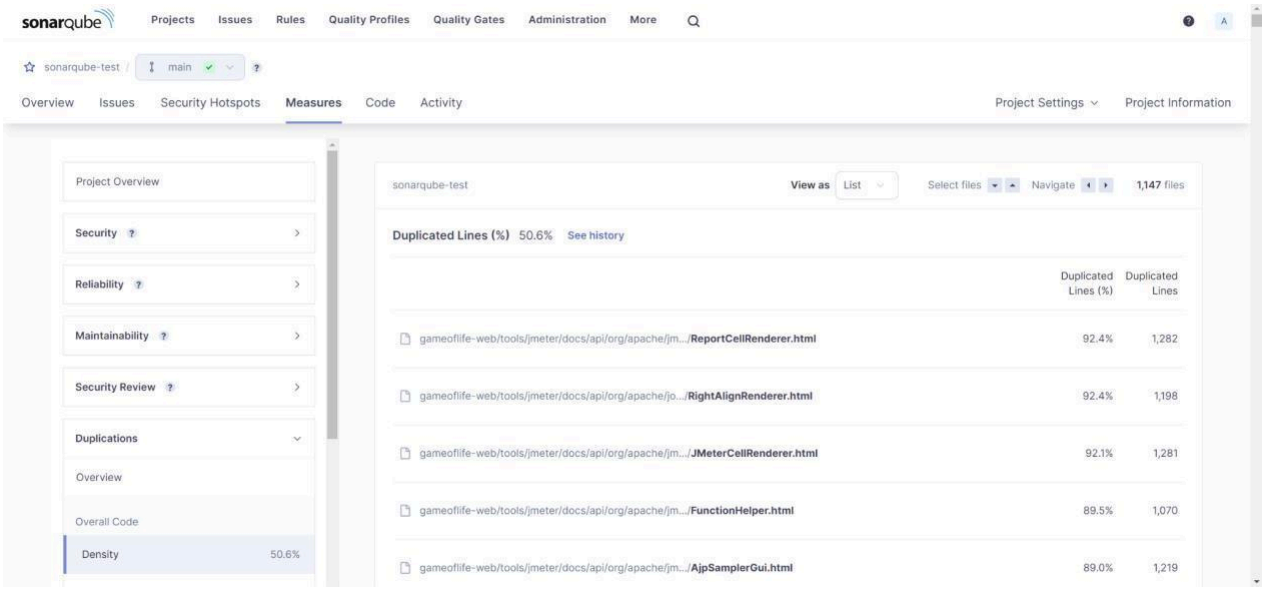
Bugs


sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
sonarqube-test
main
Overview
Issues
Security Hotspots
Measures
Code
Activity
Project Settings
Project Information
Severity
High 0
Medium 14k
Low 0
Type
Bug 14k
Vulnerability 0
Code Smell 253
Add to selection Ctrl + click
Scope
Status
Security Category
Bulk Change
Select issues
Navigate to issue
13,619 issues
56d effort
gameoflife-core/build/reports/tests/all-tests.html
Add "lang" and/or "xml:lang" attributes to this "<html>" element
Intentionality
Reliability
accessibility wcag2-a
Open
Not assigned
L1 · 2min effort · 4 years ago · Bug · Major
Add "<th>" headers to this "<table>".
Intentionality
Reliability
accessibility wcag2-a
Open
Not assigned
L9 · 2min effort · 4 years ago · Bug · Major
gameoflife-core/build/reports/tests/allclasses-frame.html
Add "lang" and/or "xml:lang" attributes to this "<html>" element
Intentionality
Reliability
accessibility wcag2-a
Embedded database should be used for evaluation purposes only

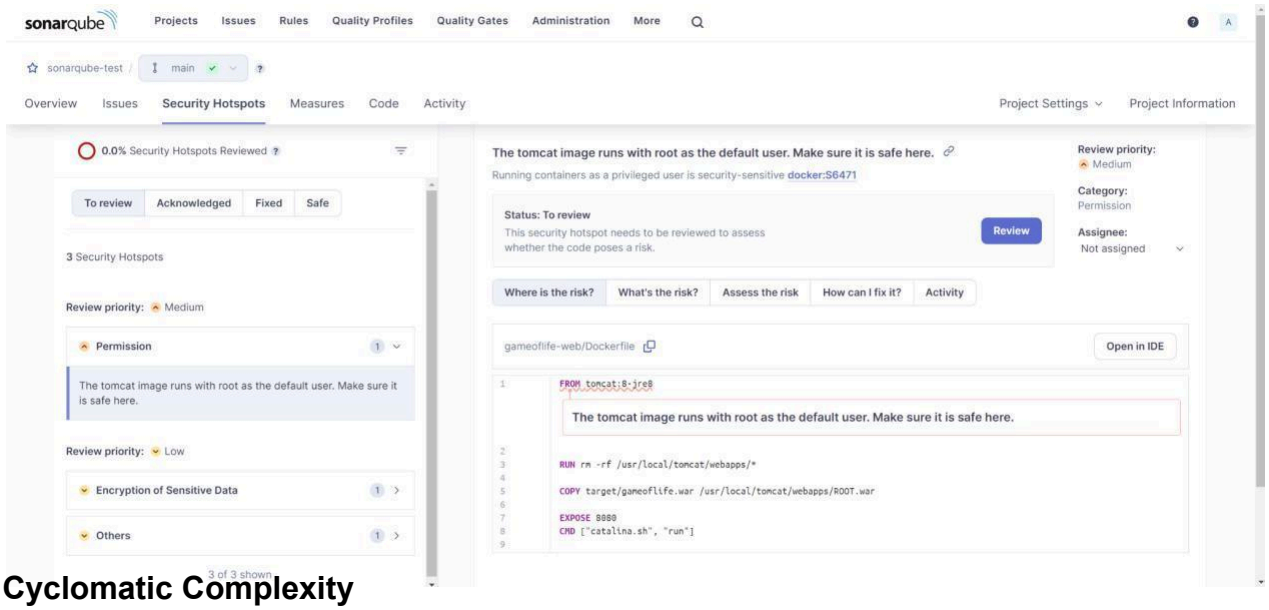
Reliability



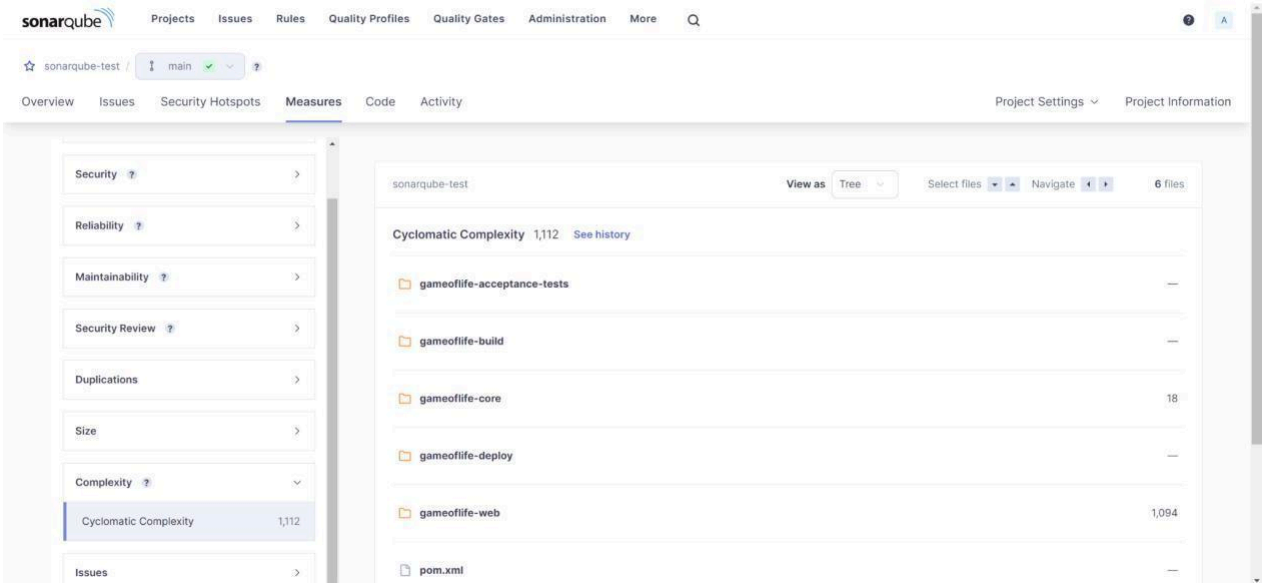
Duplicates



Security Hotspot



Cyclomatic Complexity



In this way, we have created a CI/CD Pipeline with Jenkins and integrated it with SonarQube to find issues in the code like bugs, code smells, duplicates, cyclomatic complexities, etc.

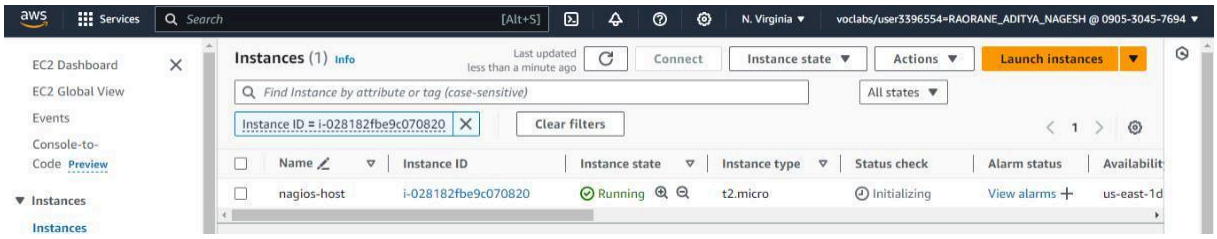
Conclusion:

In this experiment, we performed a static analysis of the code to detect bugs, code smells, and security vulnerabilities on our sample Java application.

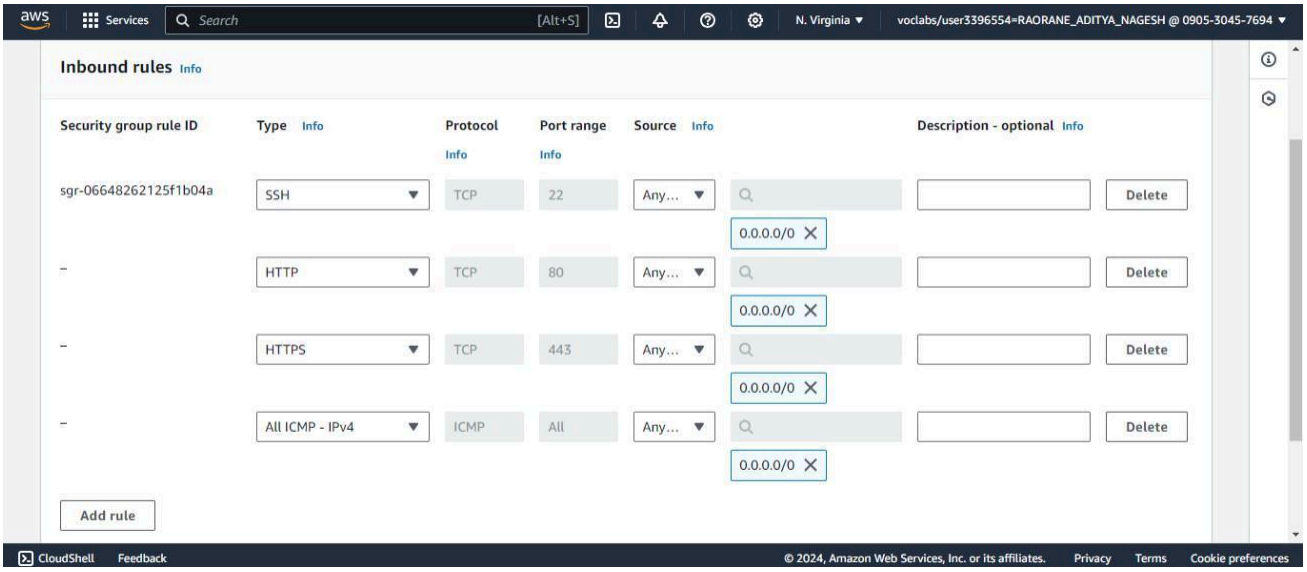
Experiment 09

Aim: To Understand Continuous monitoring and Installation and configuration of Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine.

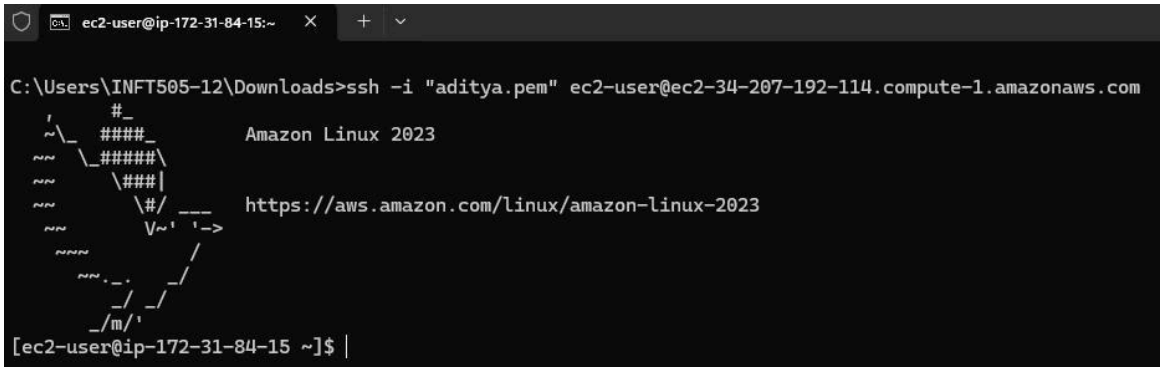
1. Create an Amazon Linux EC2 Instance in AWS and name it - nagios-host



2. Under Security Group, make sure HTTP, HTTPS, SSH, ICMP are open from everywhere.

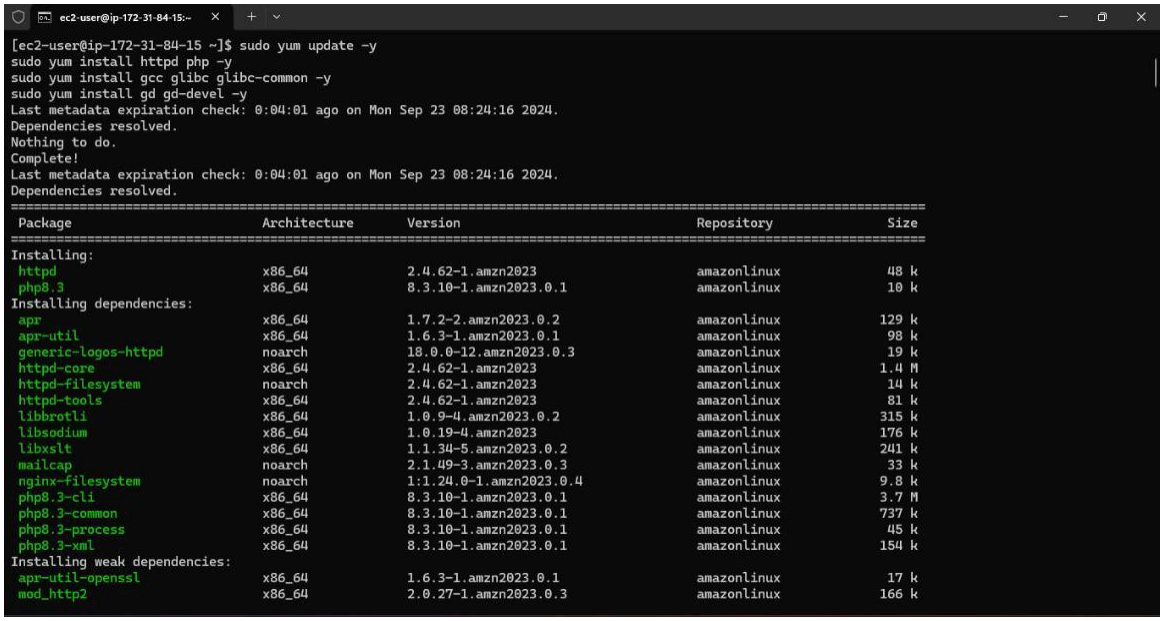


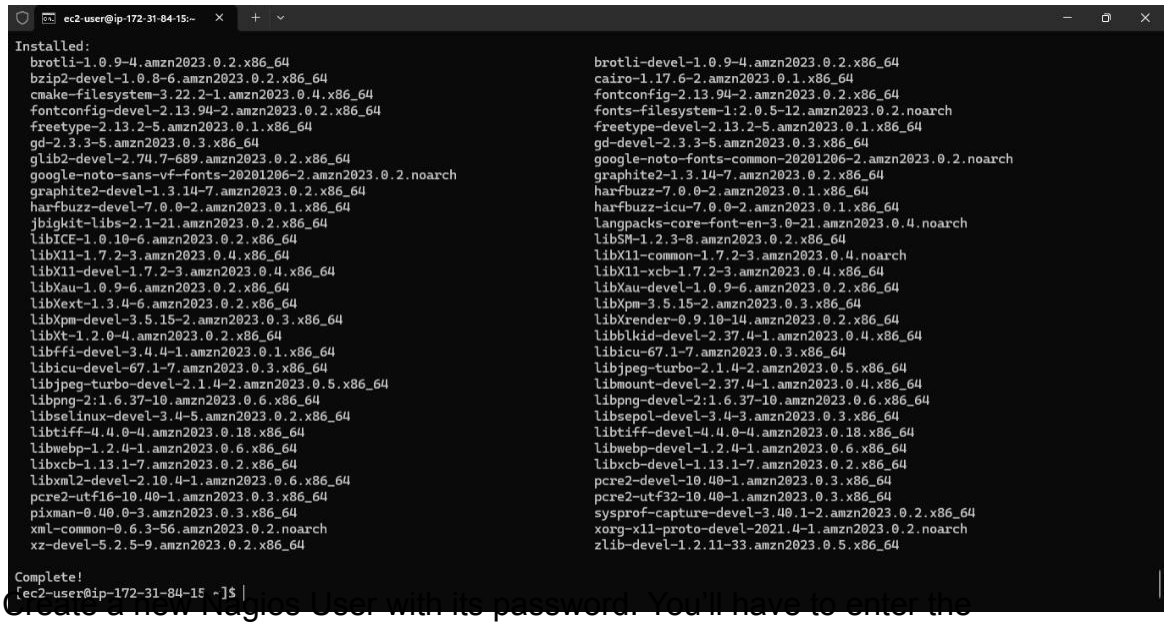
3. SSH into Your EC2 instance.



4. Update the package indices and install the following packages

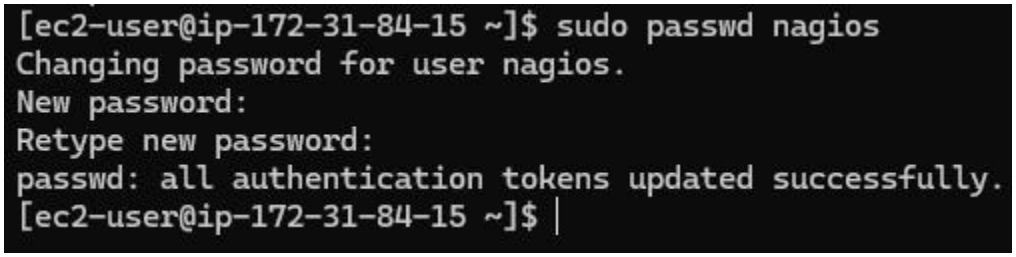
sudo yum update
sudo yum install httpd php
sudo yum install gcc glibc glibc-common
sudo yum install gd gd-devel





5. Create a new Nagios User with its password. You'll have to enter the password twice for confirmation. (adityaraorane)

   sudo adduser -m nagios
   sudo passwd nagios



6. Create a new user group

   sudo groupadd nagcmd

7. Use these commands so that you don't have to use sudo for Apache and Nagios

   sudo usermod -a -G nagcmd nagios
   sudo usermod -a -G nagcmd apache

8. Create a new directory for Nagios downloads

   mkdir ~/downloads
   cd ~/downloads

```
[ec2-user@ip-172-31-84-15 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-84-15 ~]$ sudo usermod -aG nagcmd nagios
sudo usermod -aG nagcmd apache
[ec2-user@ip-172-31-84-15 ~]$ mkdir ~/downloads
cd ~/downloads
[ec2-user@ip-172-31-84-15 downloads]$
```

9. Use wget to download the source zip files.
wget https://go.nagios.org/l/975333/2024-09-17/6kqcx

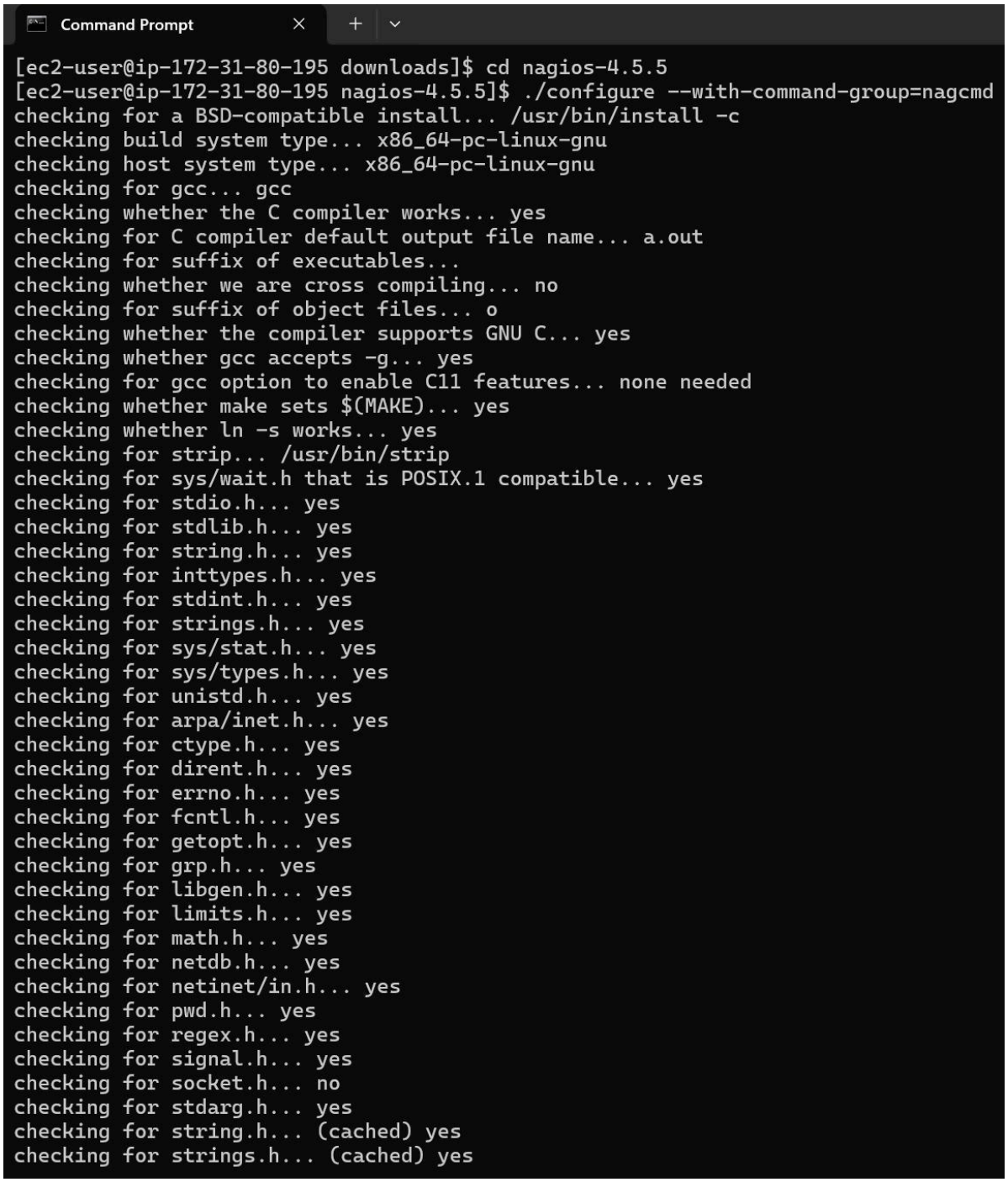
wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz

```
[ec2-user@ip-172-31-84-15 downloads]$ wget http://prdownloads.sourceforge.net/sourceforge/nagios/nagios-4.0.8.tar.gz
wget http://nagios-plugins.org/download/nagios-plugins-2.0.3.tar.gz
--2024-09-23 08:33:27--  http://prdownloads.sourceforge.net/sourceforge/nagios/nagios-4.0.8.tar.gz
Resolving prdownloads.sourceforge.net (prdownloads.sourceforge.net)... 204.68.111.105
Connecting to prdownloads.sourceforge.net (prdownloads.sourceforge.net)|204.68.111.105|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: http://downloads.sourceforge.net/project/nagios/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz [following]
--2024-09-23 08:33:27--  http://downloads.sourceforge.net/project/nagios/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz
Resolving downloads.sourceforge.net (downloads.sourceforge.net)... 204.68.111.105
Reusing existing connection to prdownloads.sourceforge.net:80.
HTTP request sent, awaiting response... 302 Found
Location: http://pilotfiber.dl.sourceforge.net/project/nagios/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz?viasf=1 [following]
--2024-09-23 08:33:27--  http://pilotfiber.dl.sourceforge.net/project/nagios/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz?viasf=1
Resolving pilotfiber.dl.sourceforge.net (pilotfiber.dl.sourceforge.net)... 216.158.133.250
Connecting to pilotfiber.dl.sourceforge.net (pilotfiber.dl.sourceforge.net)|216.158.133.250|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1805059 (1.7M) [application/x-gzip]
Saving to: 'nagios-4.0.8.tar.gz'

nagios-4.0.8.tar.gz                 100%[=====================================================================>]   1.72M  10.8MB/s    in 0.2s

2024-09-23 08:33:28 (10.8 MB/s) - 'nagios-4.0.8.tar.gz' saved [1805059/1805059]

--2024-09-23 08:33:28--  http://nagios-plugins.org/download/nagios-plugins-2.0.3.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2659772 (2.5M) [application/x-gzip]
Saving to: 'nagios-plugins-2.0.3.tar.gz'

nagios-plugins-2.0.3.tar.gz         100%[=====================================================================>]   2.54M  8.81MB/s    in 0.3s

2024-09-23 08:33:28 (8.81 MB/s) - 'nagios-plugins-2.0.3.tar.gz' saved [2659772/2659772]

[ec2-user@ip-172-31-84-15 downloads]$
```

10. Use tar to unzip and change to that directory.

tar zxvf nagios-4.5.5.tar.gz

cd nagios-4.5.5

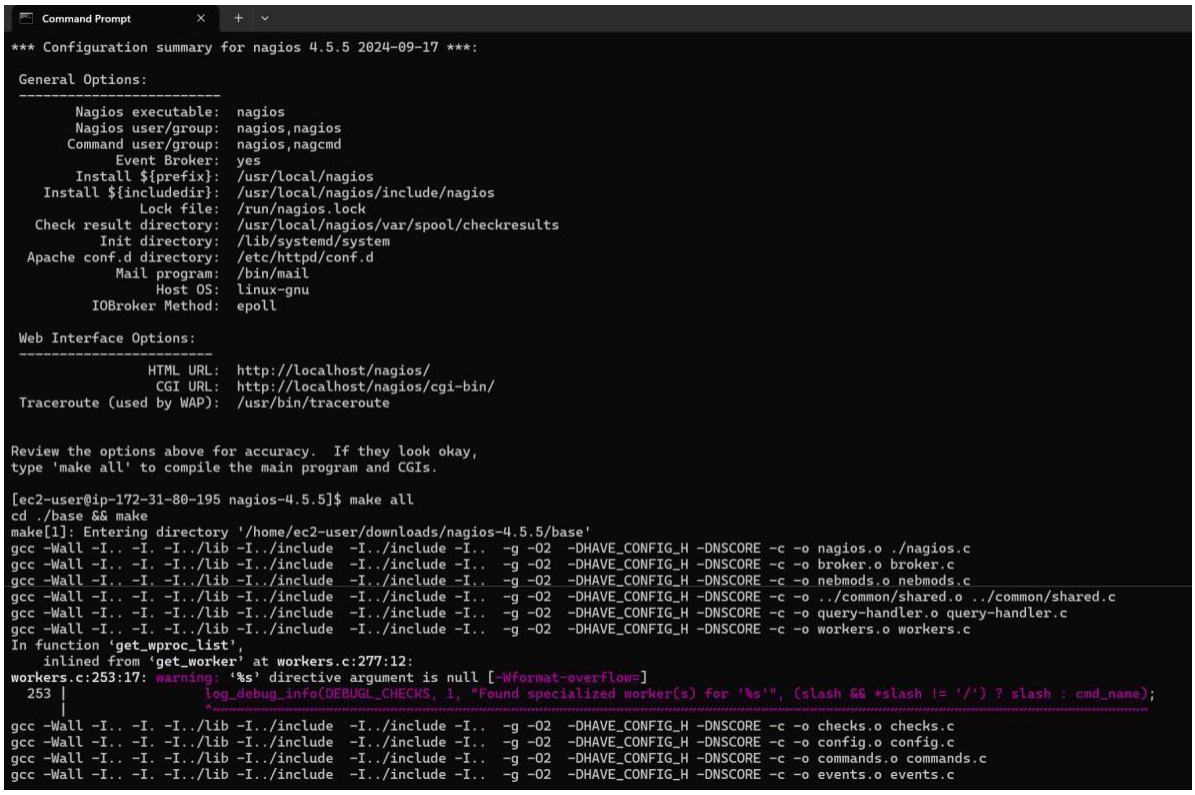

Command Prompt
[ec2-user@ip-172-31-80-195 downloads]$ cd nagios-4.5.5
[ec2-user@ip-172-31-80-195 nagios-4.5.5]$ ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether the compiler supports GNU C... yes
checking whether gcc accepts -g... yes
checking for gcc option to enable C11 features... none needed
checking whether make sets $(MAKE)... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking for sys/wait.h that is POSIX.1 compatible... yes
checking for stdio.h... yes
checking for stdlib.h... yes
checking for string.h... yes
checking for inttypes.h... yes
checking for stdint.h... yes
checking for strings.h... yes
checking for sys/stat.h... yes
checking for sys/types.h... yes
checking for unistd.h... yes
checking for arpa/inet.h... yes
checking for ctype.h... yes
checking for dirent.h... yes
checking for errno.h... yes
checking for fcntl.h... yes
checking for getopt.h... yes
checking for grp.h... yes
checking for libgen.h... yes
checking for limits.h... yes
checking for math.h... yes
checking for netdb.h... yes
checking for netinet/in.h... yes
checking for pwd.h... yes
checking for regex.h... yes
checking for signal.h... yes
checking for socket.h... no
checking for stdarg.h... yes
checking for string.h... (cached) yes
checking for strings.h... (cached) yes

```
[ec2-user@ip-172-31-80-195 nagios-4.5.5]$ sudo yum install openssl-devel
Last metadata expiration check: 0:10:42 ago on Fri Sep 27 11:19:57 2024.
Dependencies resolved.
========================================================================================================================
 Package                     Architecture         Version                               Repository                 Size
========================================================================================================================
Installing:
 openssl-devel               x86_64               1:3.0.8-1.amzn2023.0.14               amazonlinux               3.0 M

Transaction Summary
========================================================================================================================
Install  1 Package

Total download size: 3.0 M
Installed size: 4.7 M
Is this ok [y/N]: y
Downloading Packages:
openssl-devel-3.0.8-1.amzn2023.0.14.x86_64.rpm                                           14 MB/s | 3.0 MB     00:00
------------------------------------------------------------------------------------------------------------------------
Total                                                                                    11 MB/s | 3.0 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                1/1
  Installing       : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64                                                   1/1
  Running scriptlet: openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64                                                   1/1
  Verifying        : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64                                                   1/1

Installed:
  openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64

Complete!
```

11. Run the configuration script with the same group name you previously created.

./configure --with-command-group=nagcmd

12. Compile the source code. make all



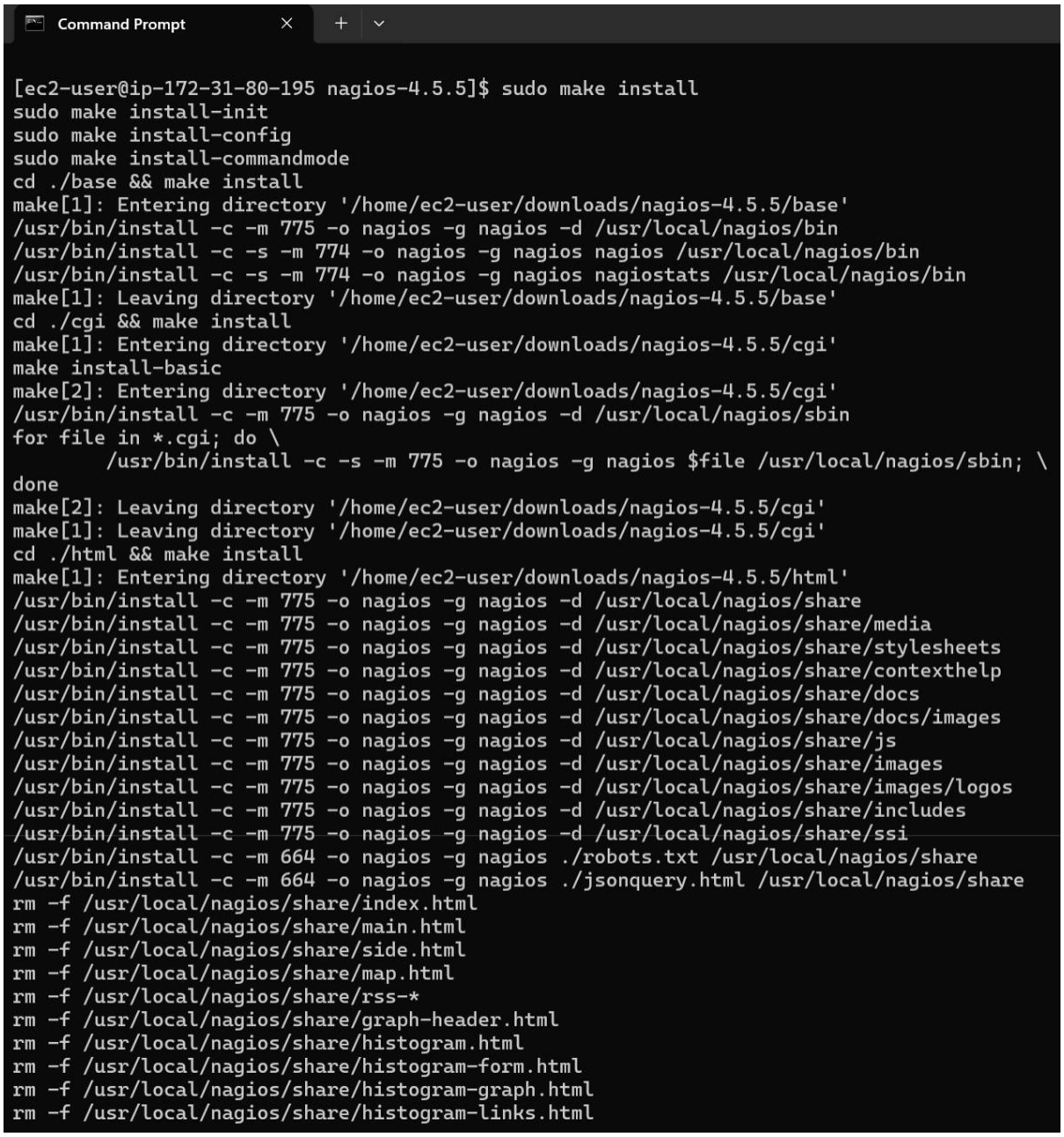
13. Install binaries, init script and sample config files. Lastly, set permissions on the external command directory.

sudo make install sudo

make install-init sudo

make install-config

sudo make install-commandmode



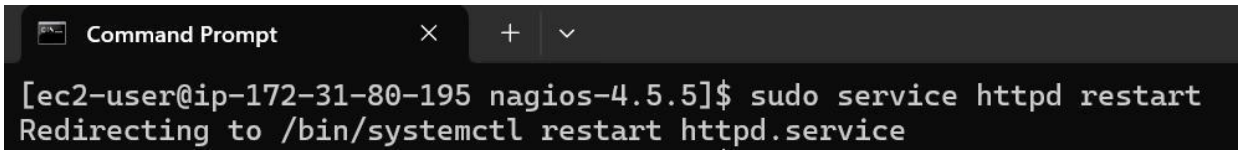
14. Configure the web interface. sudo make install-webconf

15. Create a nagiosadmin account for nagios login along with password. You'll have to specify the password twice. (adityaraorane)

    sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin

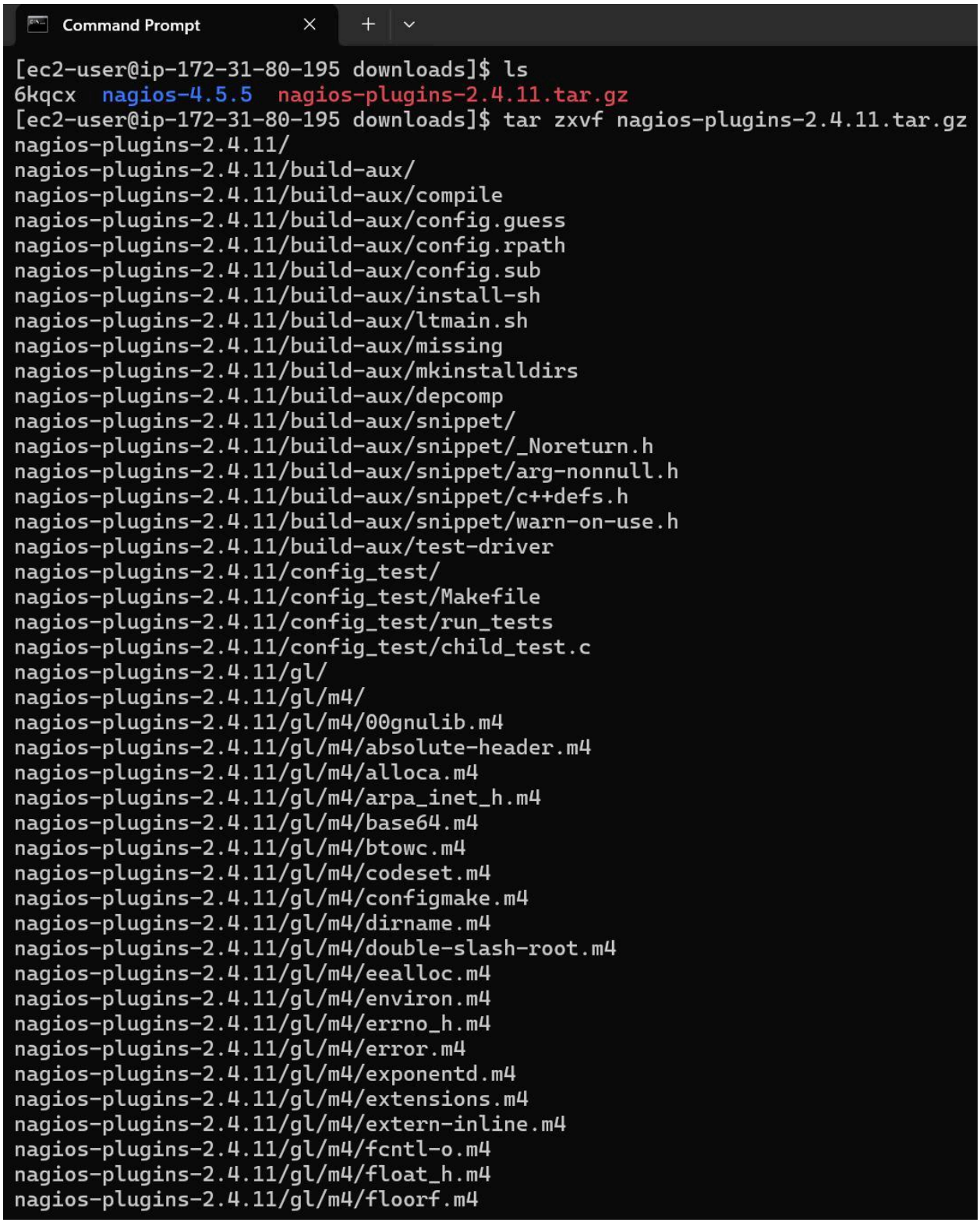
16. Restart Apache

    sudo service httpd restart



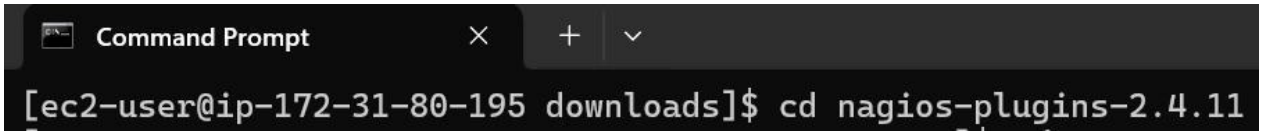
17. Go back to the downloads folder and unzip the plugins zip file.

    cd ~/downloads
    tar zxvf nagios-plugins-2.0.3.tar.gz

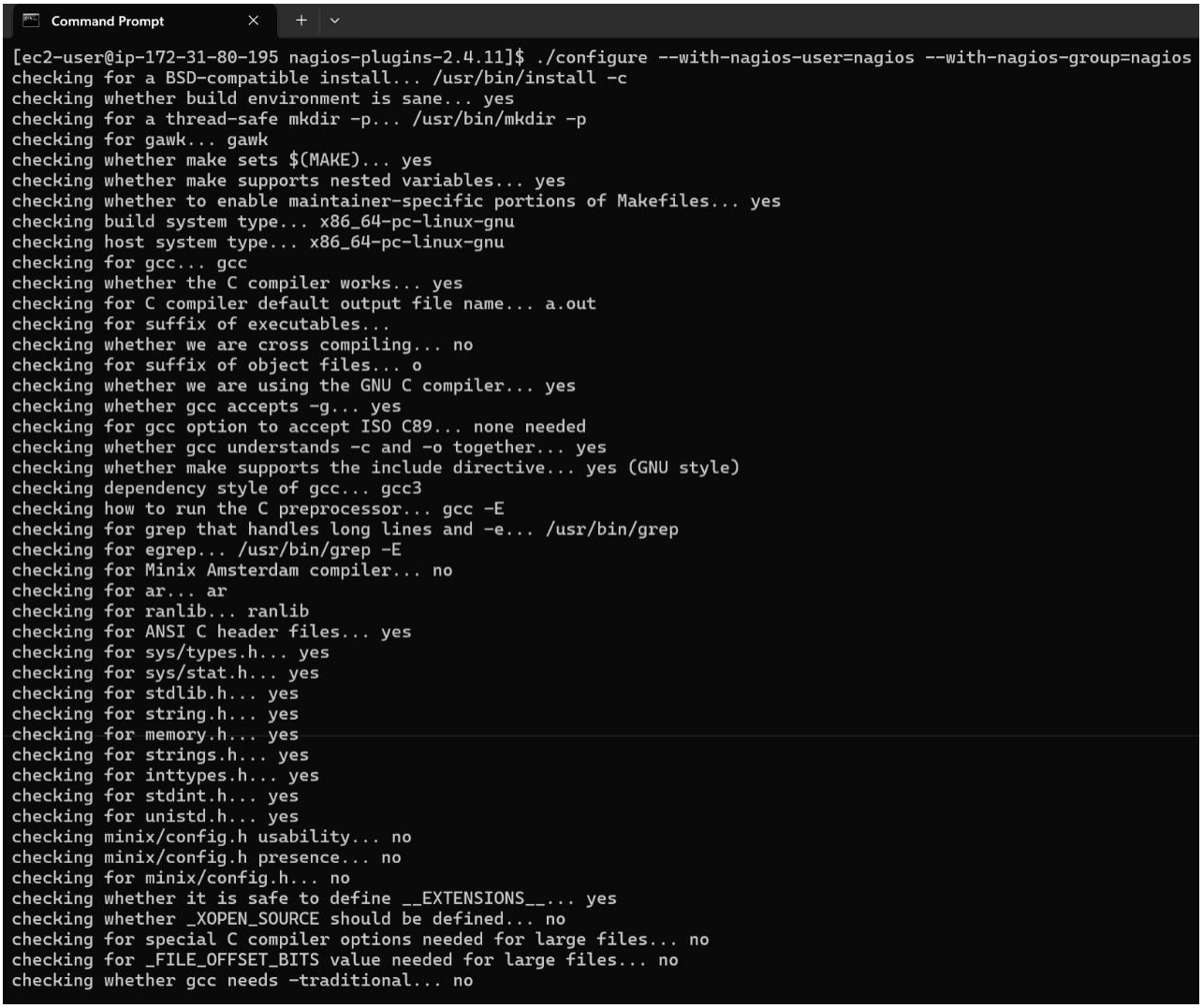

Command Prompt
[ec2-user@ip-172-31-80-195 downloads]$ ls
6kqcx nagios-4.5.5 nagios-plugins-2.4.11.tar.gz
[ec2-user@ip-172-31-80-195 downloads]$ tar zxvf nagios-plugins-2.4.11.tar.gz
nagios-plugins-2.4.11/
nagios-plugins-2.4.11/build-aux/
nagios-plugins-2.4.11/build-aux/compile
nagios-plugins-2.4.11/build-aux/config.guess
nagios-plugins-2.4.11/build-aux/config.rpath
nagios-plugins-2.4.11/build-aux/config.sub
nagios-plugins-2.4.11/build-aux/install-sh
nagios-plugins-2.4.11/build-aux/ltmain.sh
nagios-plugins-2.4.11/build-aux/missing
nagios-plugins-2.4.11/build-aux/mkinstalldirs
nagios-plugins-2.4.11/build-aux/depcomp
nagios-plugins-2.4.11/build-aux/snippet/
nagios-plugins-2.4.11/build-aux/snippet/_Noreturn.h
nagios-plugins-2.4.11/build-aux/snippet/arg-nonnull.h
nagios-plugins-2.4.11/build-aux/snippet/c++defs.h
nagios-plugins-2.4.11/build-aux/snippet/warn-on-use.h
nagios-plugins-2.4.11/build-aux/test-driver
nagios-plugins-2.4.11/config_test/
nagios-plugins-2.4.11/config_test/Makefile
nagios-plugins-2.4.11/config_test/run_tests
nagios-plugins-2.4.11/config_test/child_test.c
nagios-plugins-2.4.11/gl/
nagios-plugins-2.4.11/gl/m4/
nagios-plugins-2.4.11/gl/m4/00gnulib.m4
nagios-plugins-2.4.11/gl/m4/absolute-header.m4
nagios-plugins-2.4.11/gl/m4/alloca.m4
nagios-plugins-2.4.11/gl/m4/arpa_inet_h.m4
nagios-plugins-2.4.11/gl/m4/base64.m4
nagios-plugins-2.4.11/gl/m4/btowc.m4
nagios-plugins-2.4.11/gl/m4/codeset.m4
nagios-plugins-2.4.11/gl/m4/configmake.m4
nagios-plugins-2.4.11/gl/m4/dirname.m4
nagios-plugins-2.4.11/gl/m4/double-slash-root.m4
nagios-plugins-2.4.11/gl/m4/eealloc.m4
nagios-plugins-2.4.11/gl/m4/environ.m4
nagios-plugins-2.4.11/gl/m4/errno_h.m4
nagios-plugins-2.4.11/gl/m4/error.m4
nagios-plugins-2.4.11/gl/m4/exponentd.m4
nagios-plugins-2.4.11/gl/m4/extensions.m4
nagios-plugins-2.4.11/gl/m4/extern-inline.m4
nagios-plugins-2.4.11/gl/m4/fcntl-o.m4
nagios-plugins-2.4.11/gl/m4/float_h.m4
nagios-plugins-2.4.11/gl/m4/floorf.m4

18. Compile and install plugins

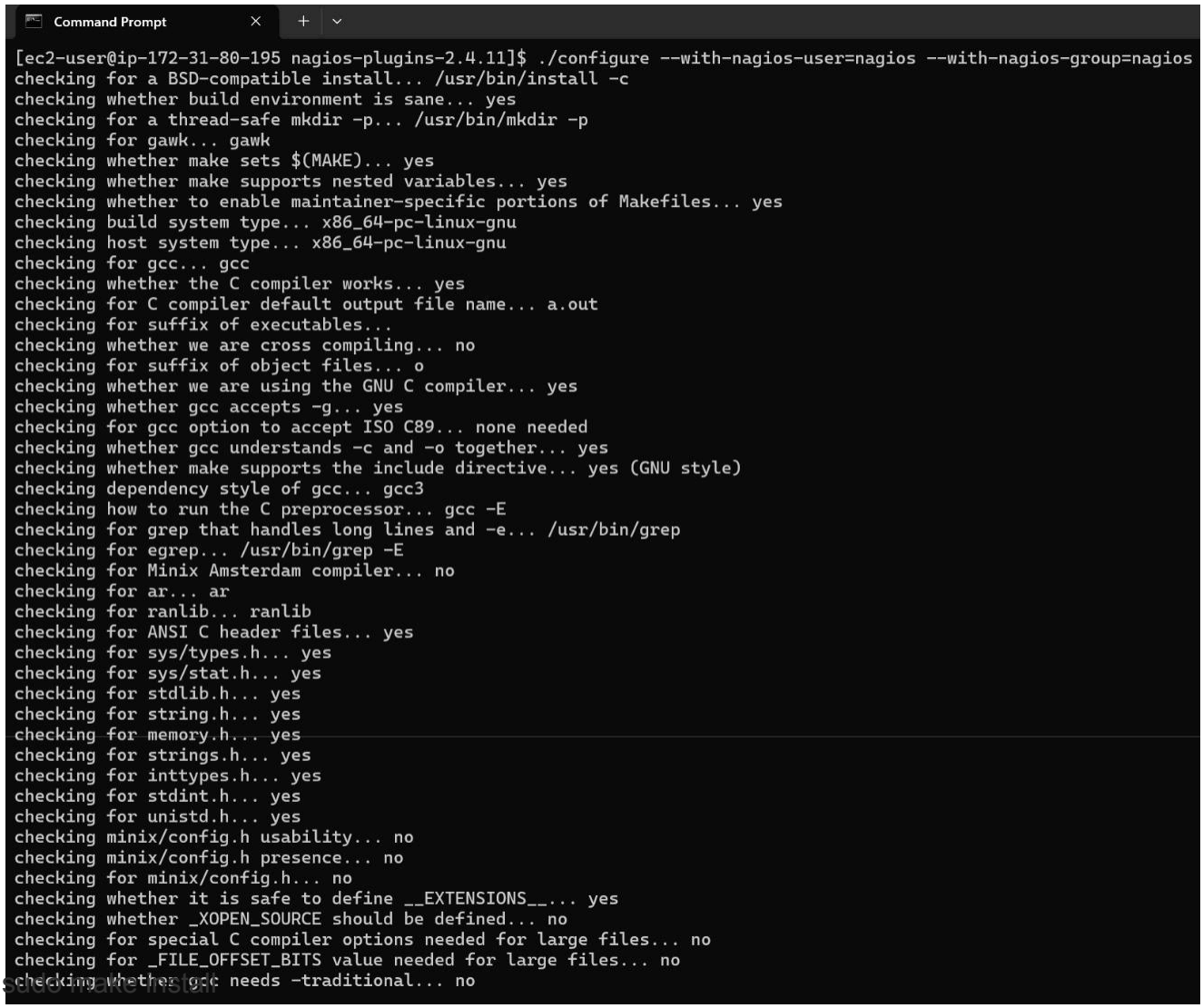
cd nagios-plugins-2.0.3

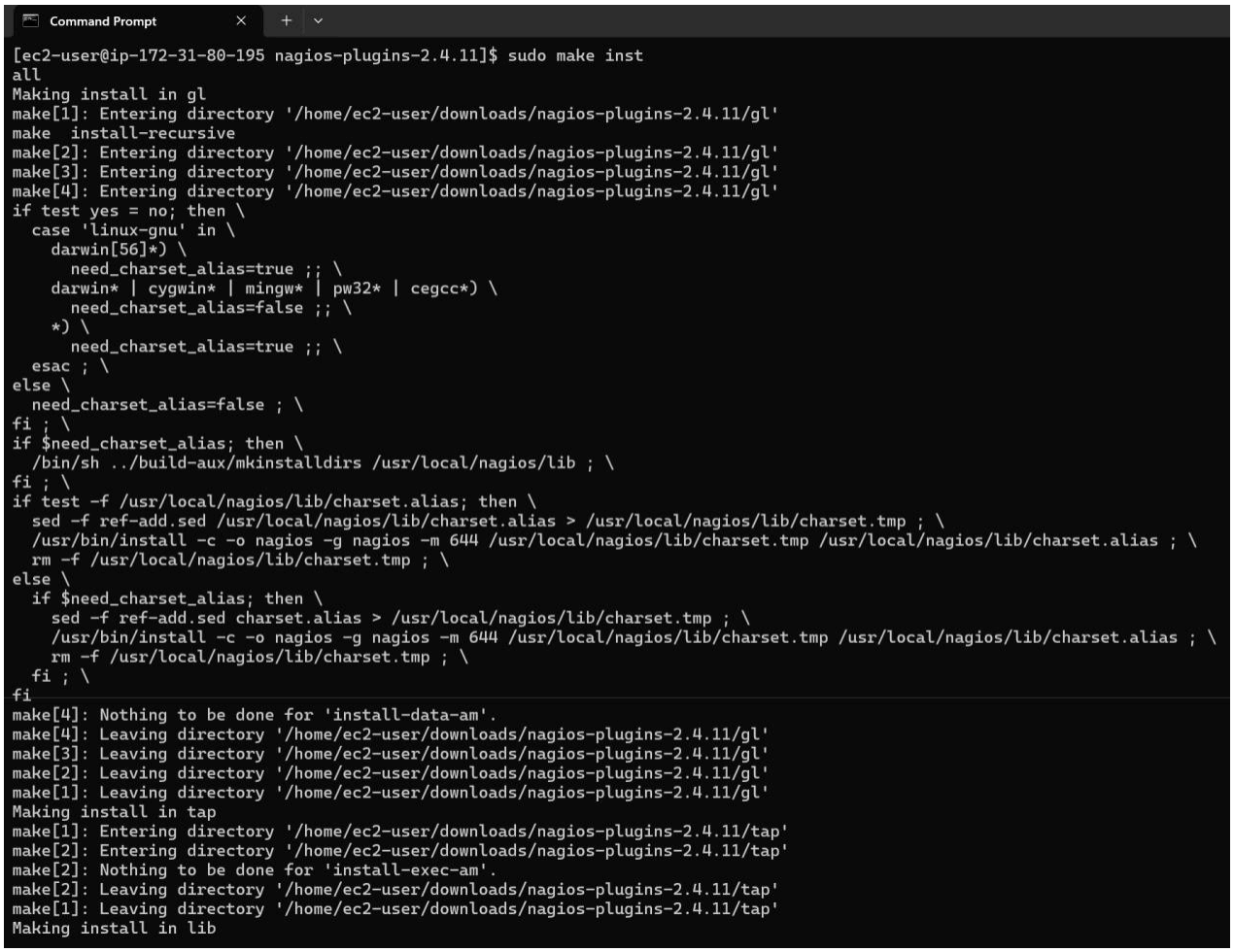


./configure --with-nagios-user=nagios --with-nagios-group=nagios



make

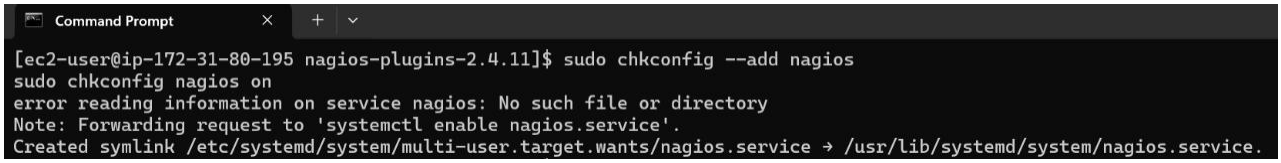

Command Prompt
[ec2-user@ip-172-31-80-195 nagios-plugins-2.4.11]$ ./configure --with-nagios-user=nagios --with-nagios-group=nagios
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether to enable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether gcc understands -c and -o together... yes
checking whether make supports the include directive... yes (GNU style)
checking dependency style of gcc... gcc3
checking how to run the C preprocessor... gcc -E
checking for grep that handles long lines and -e... /usr/bin/grep
checking for egrep... /usr/bin/grep -E
checking for Minix Amsterdam compiler... no
checking for ar... ar
checking for ranlib... ranlib
checking for ANSI C header files... yes
checking for sys/types.h... yes
checking for sys/stat.h... yes
checking for stdlib.h... yes
checking for string.h... yes
checking for memory.h... yes
checking for strings.h... yes
checking for inttypes.h... yes
checking for stdint.h... yes
checking for unistd.h... yes
checking minix/config.h usability... no
checking minix/config.h presence... no
checking for minix/config.h... no
checking whether it is safe to define __EXTENSIONS__... yes
checking whether _XOPEN_SOURCE should be defined... no
checking for special C compiler options needed for large files... no
checking for _FILE_OFFSET_BITS value needed for large files... no
checking whether gcc needs -traditional... no
sudo make install



19. Start Nagios

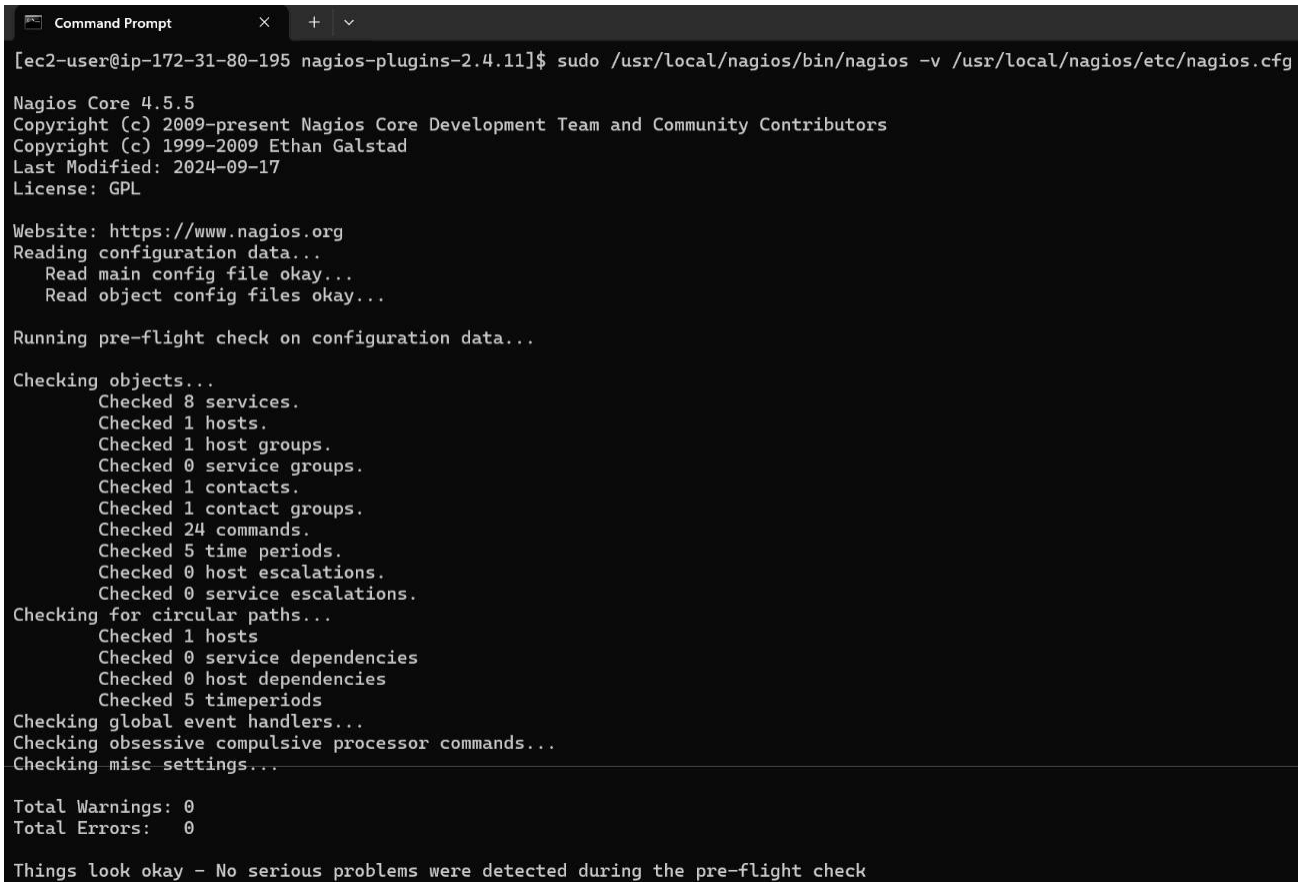
    Add Nagios to the list of system services

    sudo chkconfig --add nagios
    sudo chkconfig nagios on

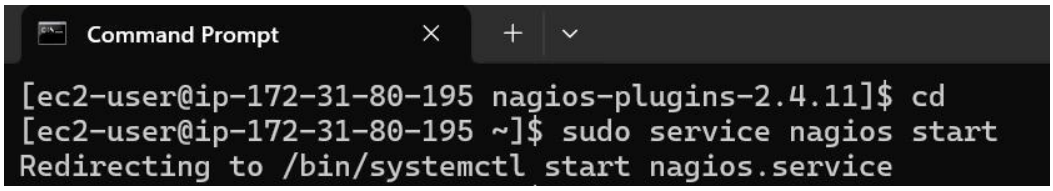


    Verify the sample configuration files

    sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

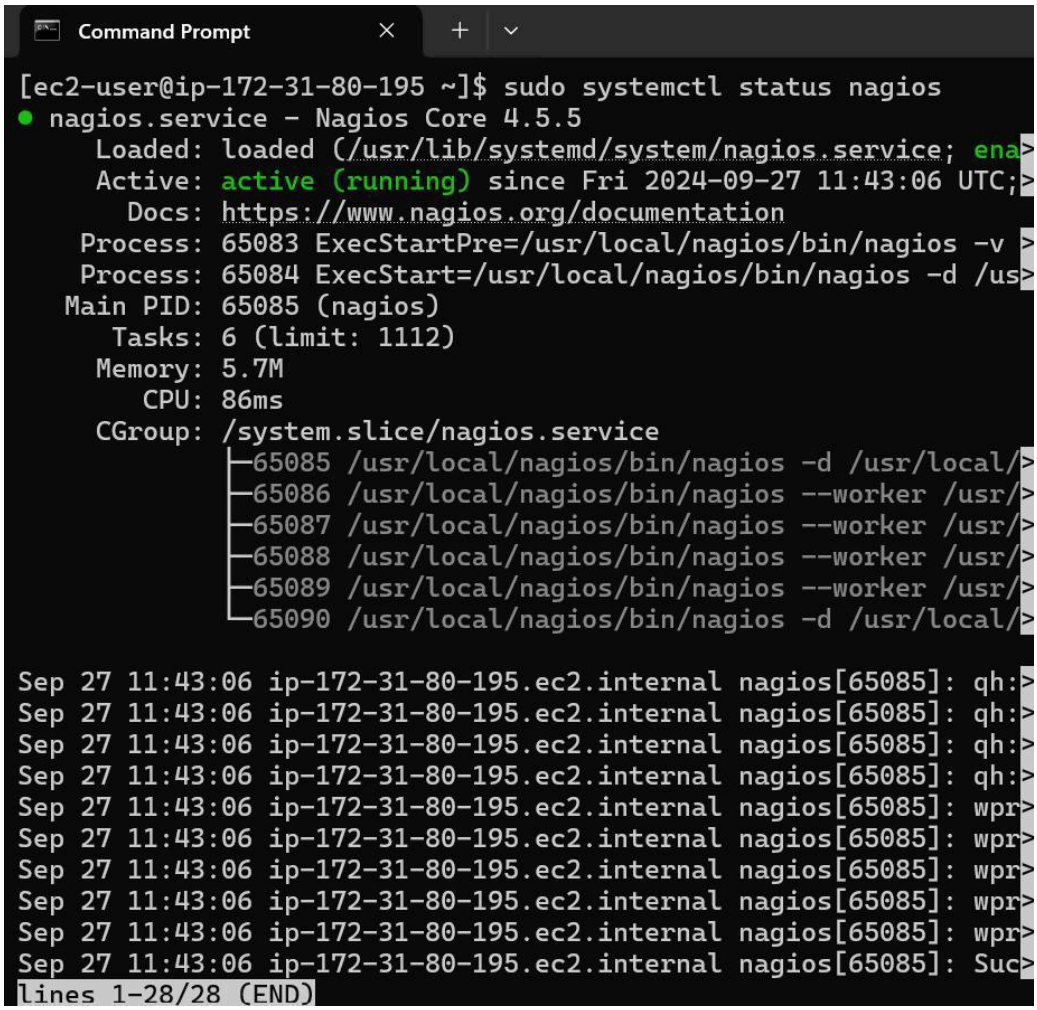


sudo service nagios start



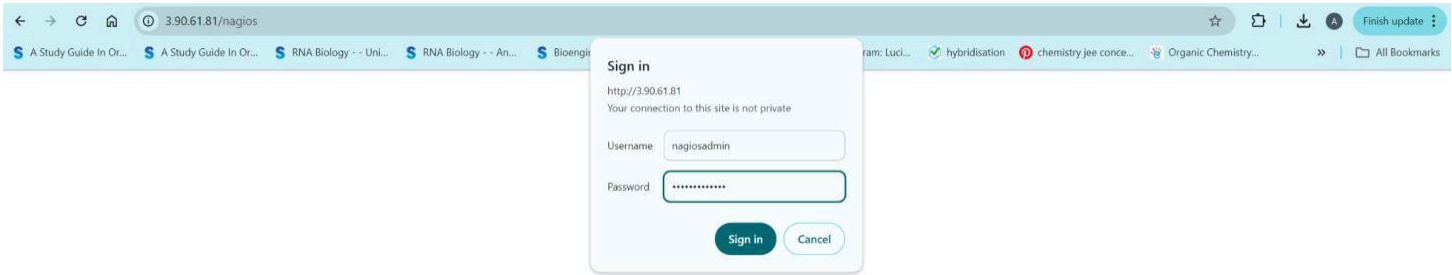
20. Check the status of Nagios

sudo systemctl status nagios



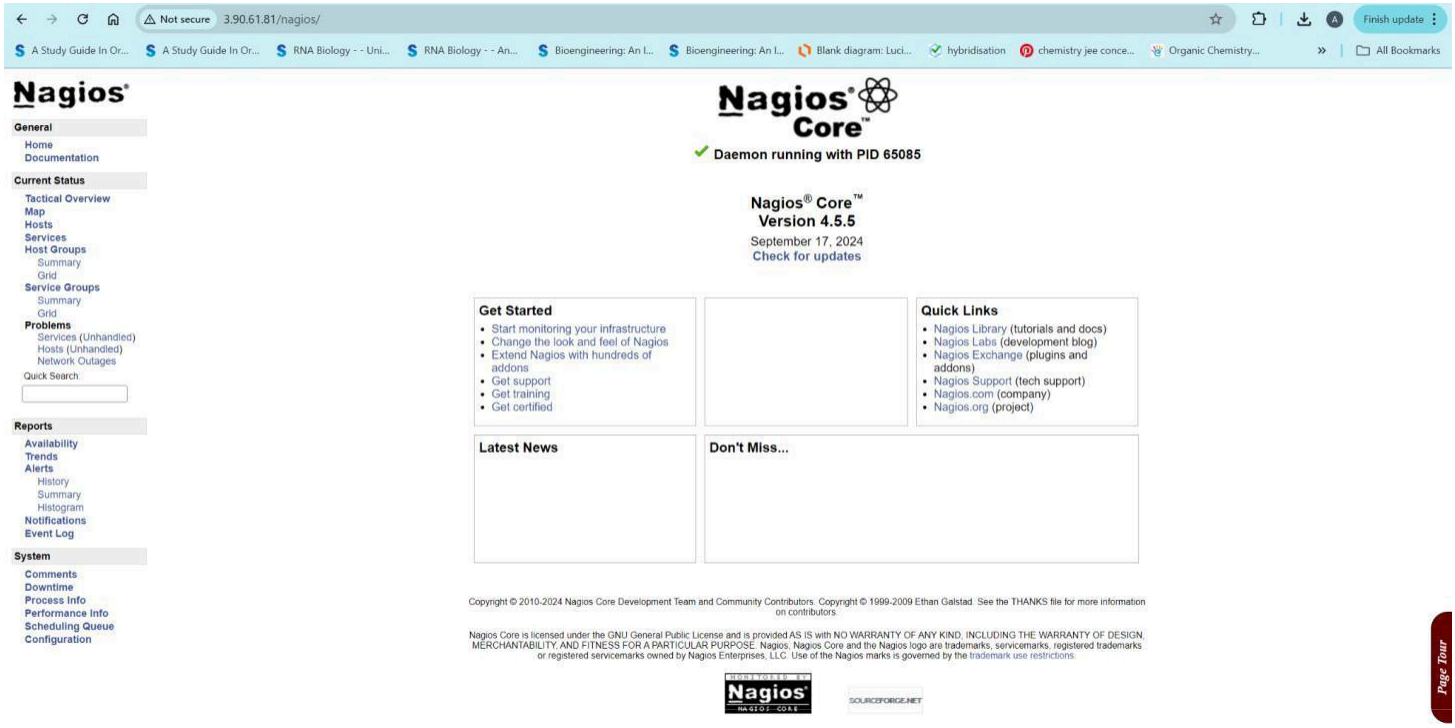
21. Go back to EC2 Console and copy the Public IP address of this instance. Open up your browser and look for **http://<your_public_ip_address>/nagios .**

http://3.90.61.81/nagios



Enter username as **nagiosadmin** and password as **adityaraorane** (which you set in Step 15).

24. After entering the correct credentials, you will see this page.



Conclusion: We have successfully installed and configured Nagios Core, Nagios Plugins, and NRPE on a Linux machine. This enables us to effectively manage system performance and proactively address potential issues.

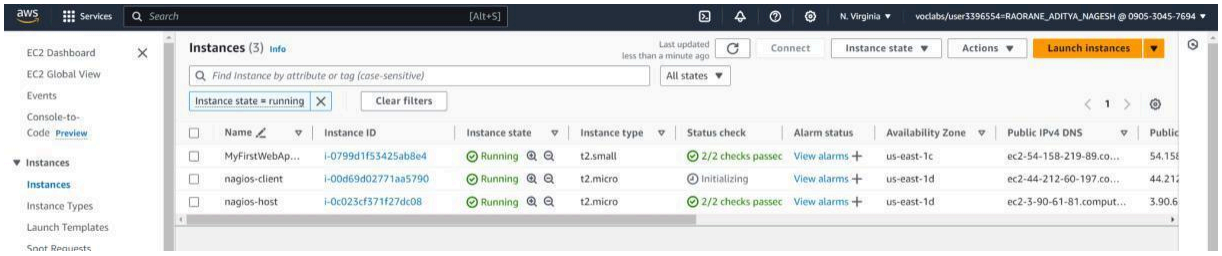
Experiment 10

Aim: To perform Port, Service monitoring, Windows/Linux server monitoring using Nagios.

1. To Confirm that Nagios is running **on the server side**, run this ***sudo systemctl status nagios*** on the **nagios-host**.

```
[ec2-user@ip-172-31-80-195 ~]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; ena>
     Active: active (running) since Fri 2024-09-27 11:43:06 UTC;>
       Docs: https://www.nagios.org/documentation
    Process: 65083 ExecStartPre=/usr/local/nagios/bin/nagios -v >
    Process: 65084 ExecStart=/usr/local/nagios/bin/nagios -d /us>
   Main PID: 65085 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 5.7M
        CPU: 86ms
     CGroup: /system.slice/nagios.service
             ├─65085 /usr/local/nagios/bin/nagios -d /usr/local/>
             ├─65086 /usr/local/nagios/bin/nagios --worker /usr/>
             ├─65087 /usr/local/nagios/bin/nagios --worker /usr/>
             ├─65088 /usr/local/nagios/bin/nagios --worker /usr/>
             ├─65089 /usr/local/nagios/bin/nagios --worker /usr/>
             └─65090 /usr/local/nagios/bin/nagios -d /usr/local/>

Sep 27 11:43:06 ip-172-31-80-195.ec2.internal nagios[65085]: qh:>
Sep 27 11:43:06 ip-172-31-80-195.ec2.internal nagios[65085]: qh:>
Sep 27 11:43:06 ip-172-31-80-195.ec2.internal nagios[65085]: qh:>
Sep 27 11:43:06 ip-172-31-80-195.ec2.internal nagios[65085]: qh:>
Sep 27 11:43:06 ip-172-31-80-195.ec2.internal nagios[65085]: wpr>
Sep 27 11:43:06 ip-172-31-80-195.ec2.internal nagios[65085]: wpr>
Sep 27 11:43:06 ip-172-31-80-195.ec2.internal nagios[65085]: wpr>
Sep 27 11:43:06 ip-172-31-80-195.ec2.internal nagios[65085]: wpr>
Sep 27 11:43:06 ip-172-31-80-195.ec2.internal nagios[65085]: wpr>
Sep 27 11:43:06 ip-172-31-80-195.ec2.internal nagios[65085]: Suc>
```

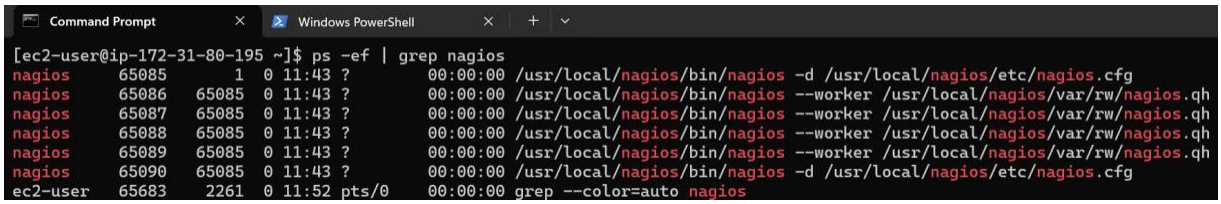
2. To monitor a Linux machine, create an Ubuntu server EC2 Instance in AWS. Provide it with the same security group as the nagios-host and name it ‘nagios-client’ alongside the host.



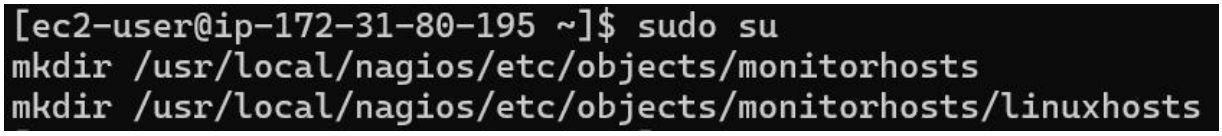
For now, leave this machine as is, and go back to your nagios-host.

3. On the server, run this

   command ps -ef | grep nagios

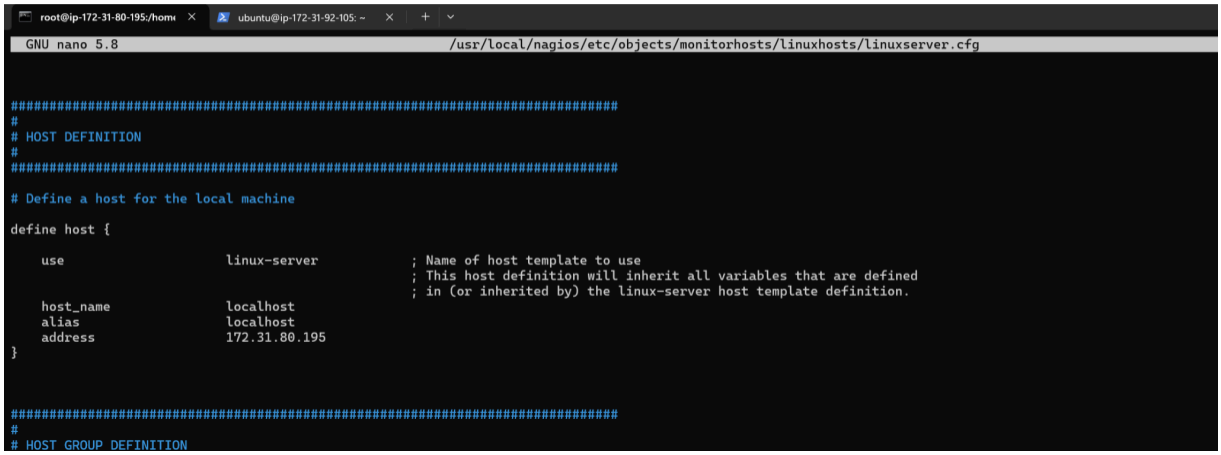


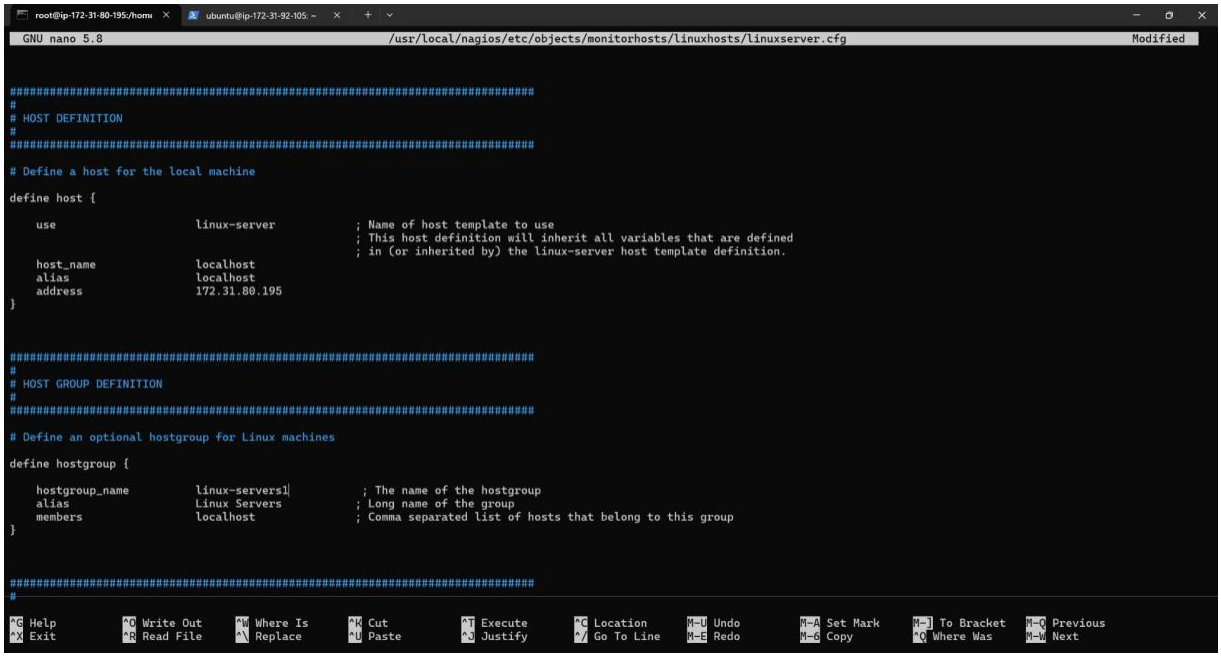
4. Become a root user and create 2
   folders sudo su
   mkdir /usr/local/nagios/etc/objects/monitorhosts
   mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts



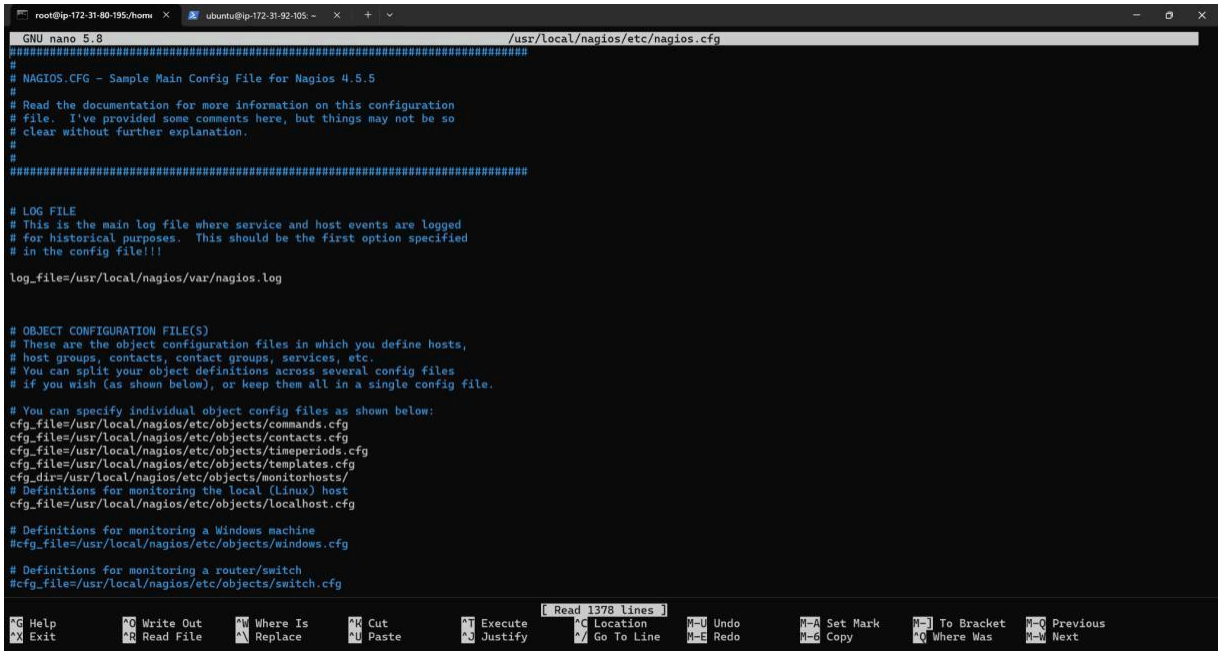
5. Copy the sample localhost.cfg file to linuxhost folder
   cp /usr/local/nagios/etc/objects/localhost.cfg
   /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserve

6. Open linuxserver.cfg using nano and make the following changes
   nano
/usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

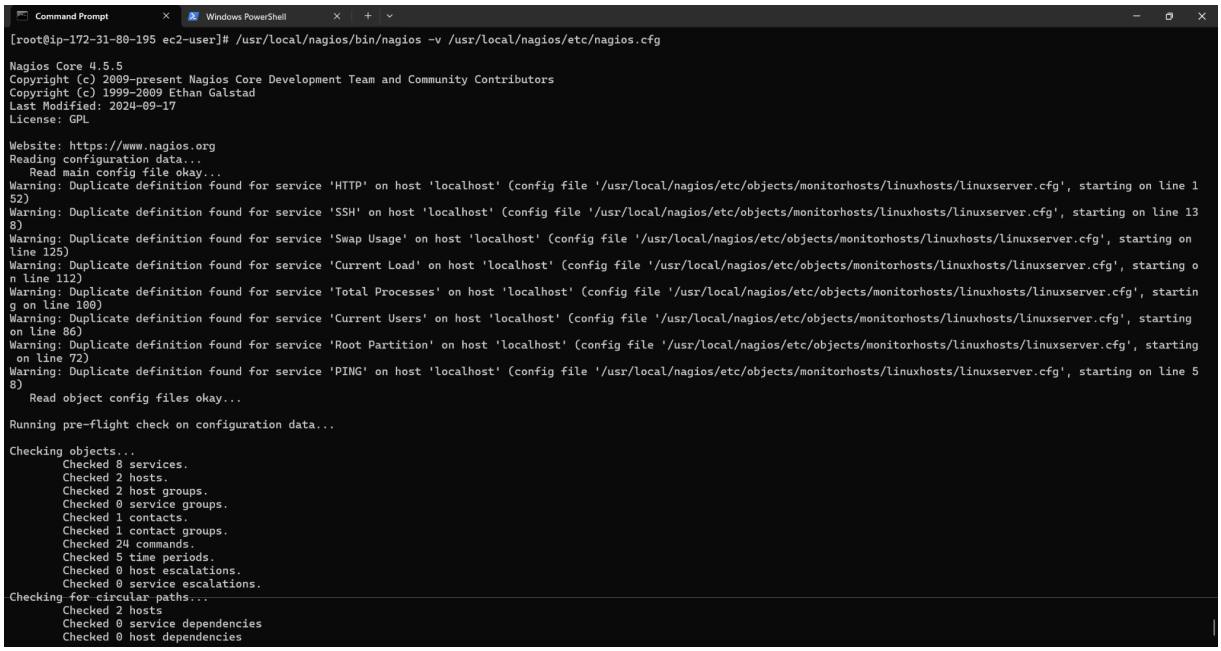




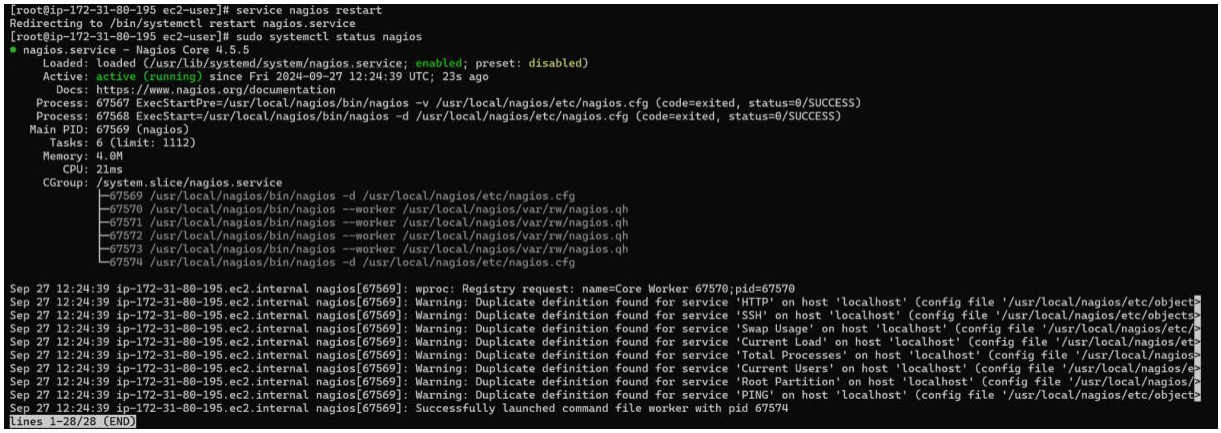
7. Open the Nagios Config file and add the following line nano /usr/local/nagios/etc/nagios.cfg
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts



8. Verify the configuration files



9. Restart nagios service using service nagios restart.



10. Under **nagios-client** ,SSH into the machine or simply use the EC2 Instance Connect feature.

```
Command Prompt | Windows PowerShell

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\adity> cd Downloads
PS C:\Users\adity\Downloads> ssh -i "ar.pem" ubuntu@ec2-44-212-60-197.compute-1.amazonaws.com
The authenticity of host 'ec2-44-212-60-197.compute-1.amazonaws.com (44.212.60.197)' can't be established.
ED25519 key fingerprint is SHA256:Lwrr9v/VJNu2G/UrjnK/N9yYvSdOCbsG7Ppk83eeKWo.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-44-212-60-197.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro
```

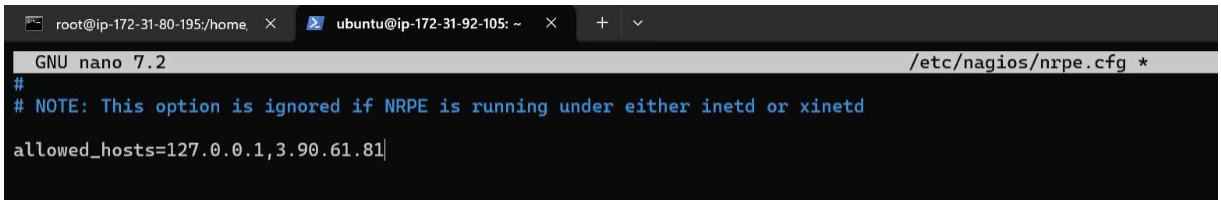
11. Make a package index update and install gcc, nagios-nrpe-server and the plugins.

sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins

```
Command Prompt | Windows PowerShell

ubuntu@ip-172-31-92-105:~$ sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [380 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [535 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [130 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [8636 B]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [377 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [156 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [45.0 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [14.8 kB]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [353 kB]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [68.1 kB]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 c-n-f Metadata [424 B]
Get:24 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [14.4 kB]
Get:25 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [3608 B]
Get:26 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [212 B]
Get:27 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [532 B]
Get:28 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:29 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [112 B]
Get:30 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [10.6 kB]
Get:31 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [10.8 kB]
Get:32 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [17.6 kB]
Get:33 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1104 B]
Get:34 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:35 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:36 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:37 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:38 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [82.9 kB]
Get:39 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4560 B]
Get:40 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [272 kB]
Get:41 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [115 kB]
Get:42 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:43 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.3 kB]
```

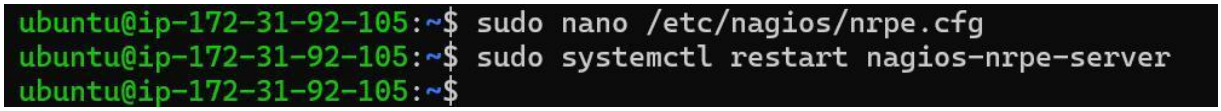
12. Open nrpe.cfg file to make changes.

    sudo nano /etc/nagios/nrpe.cfg
    Under allowed_hosts, add your nagios-host IP address like so

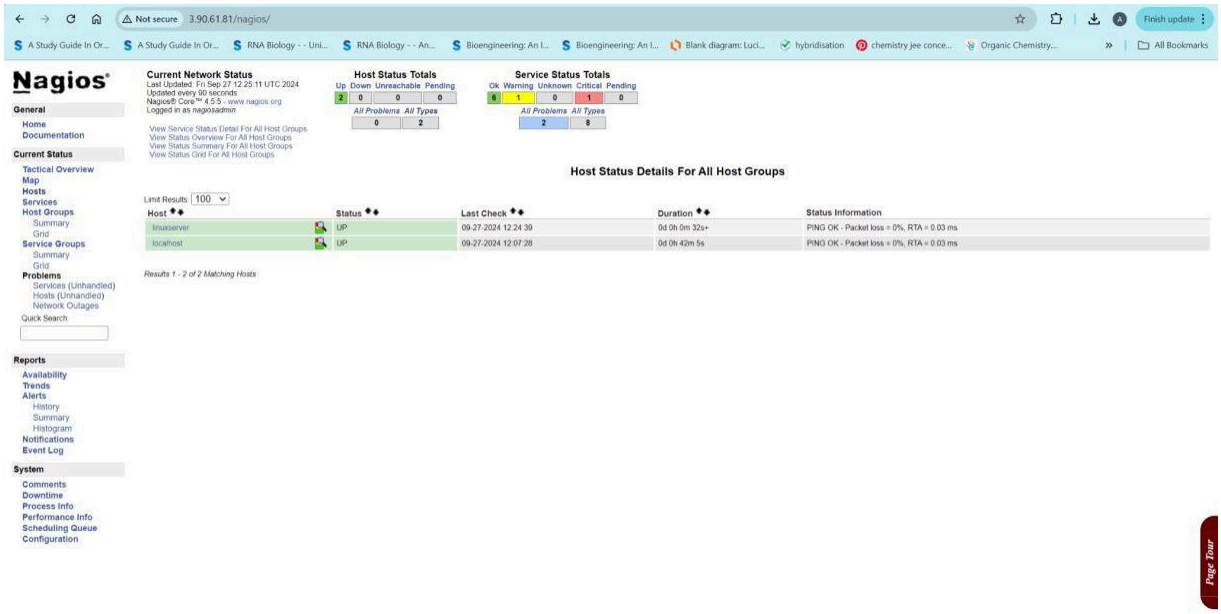


13. Restart the NRPE server

    sudo systemctl restart nagios-nrpe-server



14. Now, check your nagios dashboard and you'll see a new host being added.

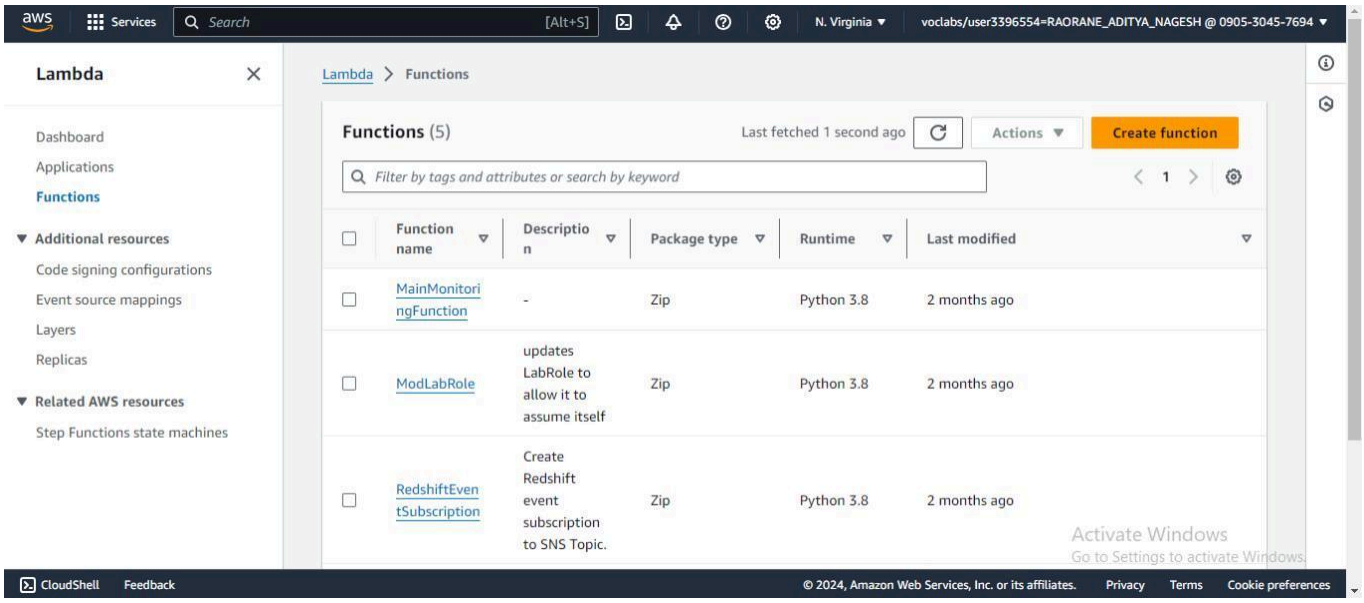


Conclusion: Thus we successfully performed port monitoring of a Linux server using Nagios.Utilizing Nagios for comprehensive monitoring of ports, services, and Windows/Linux servers enhances system reliability, improves performance, and ensures proactive management of IT infrastructure, ultimately driving operational efficiency.

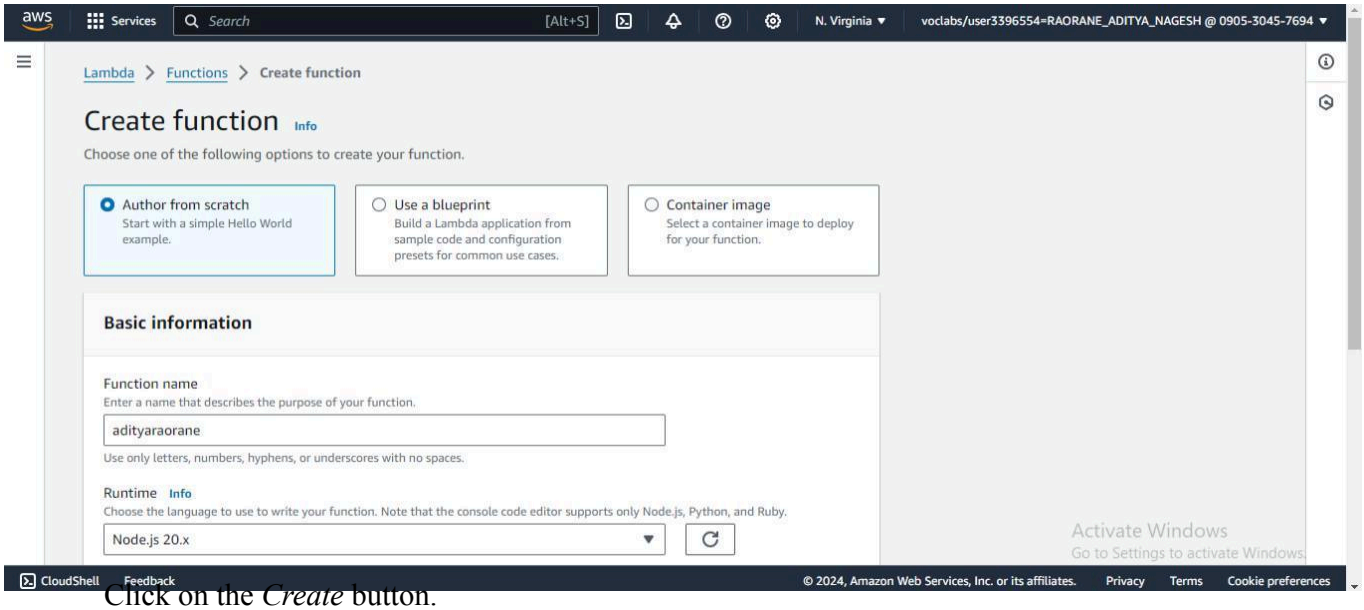
Experiment 11

Aim: To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

1. Open up the Lambda Console and click on the Create button.

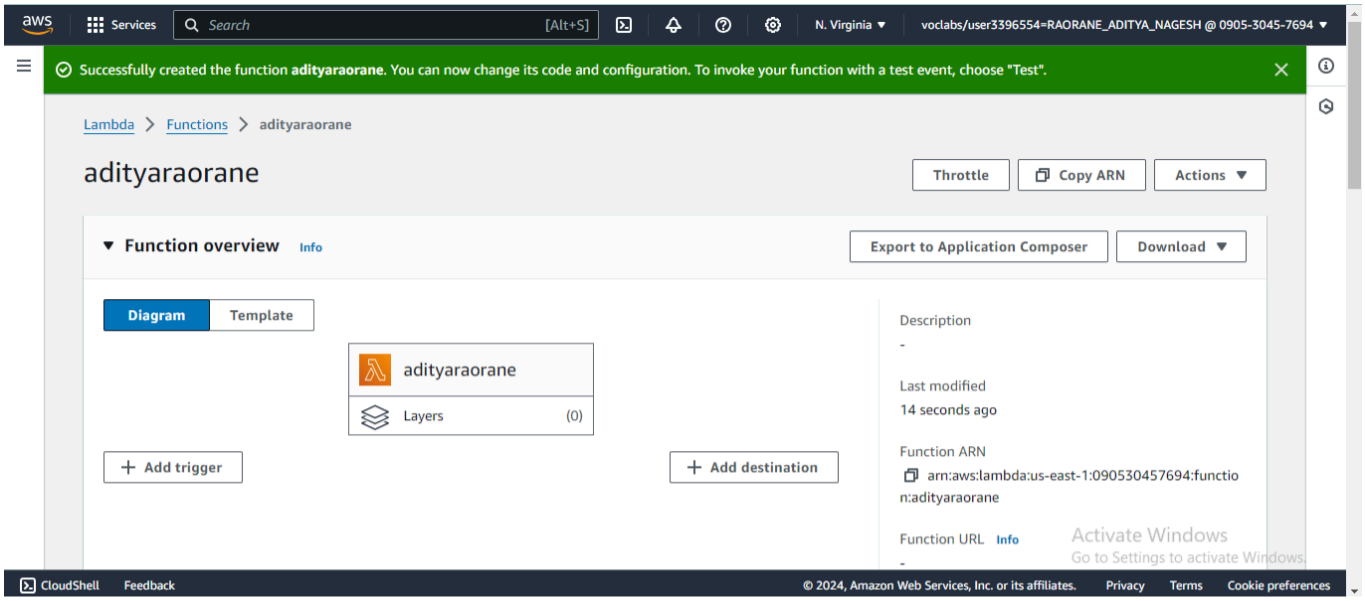


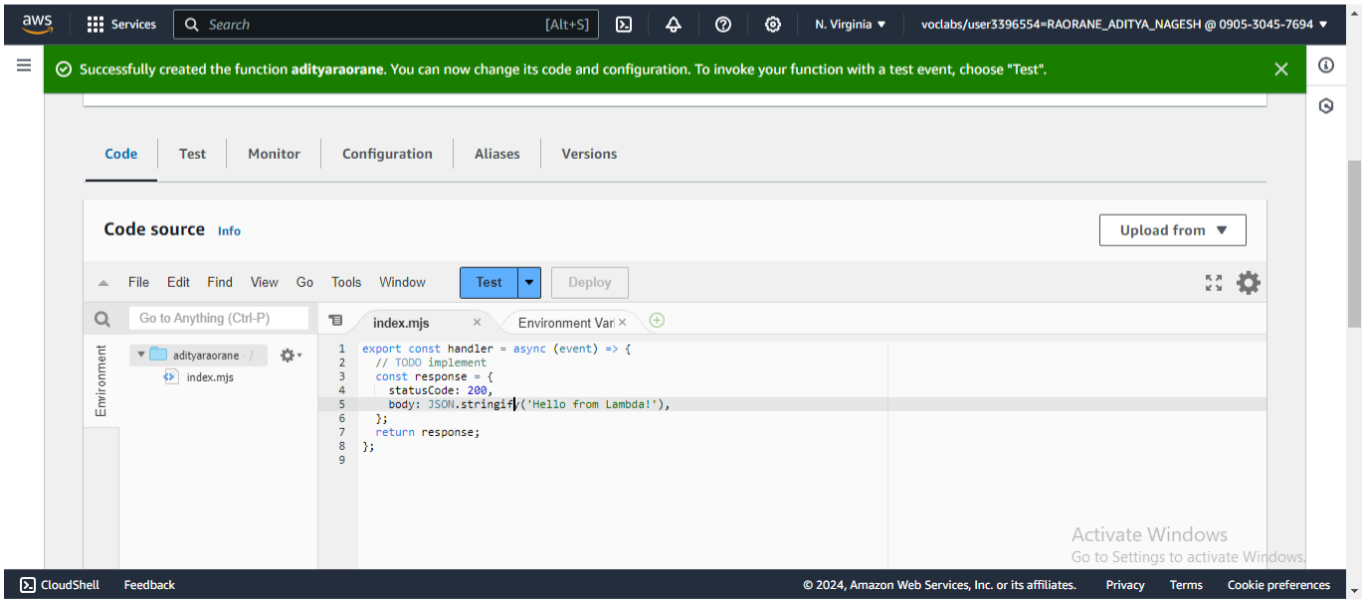
2. Choose to create a function from scratch or use a blueprint, i.e templates defined by AWS for you with all configuration presets required for the most common use cases. After that, choose to create a new role with basic Lambda permissions if you don't have an existing one.



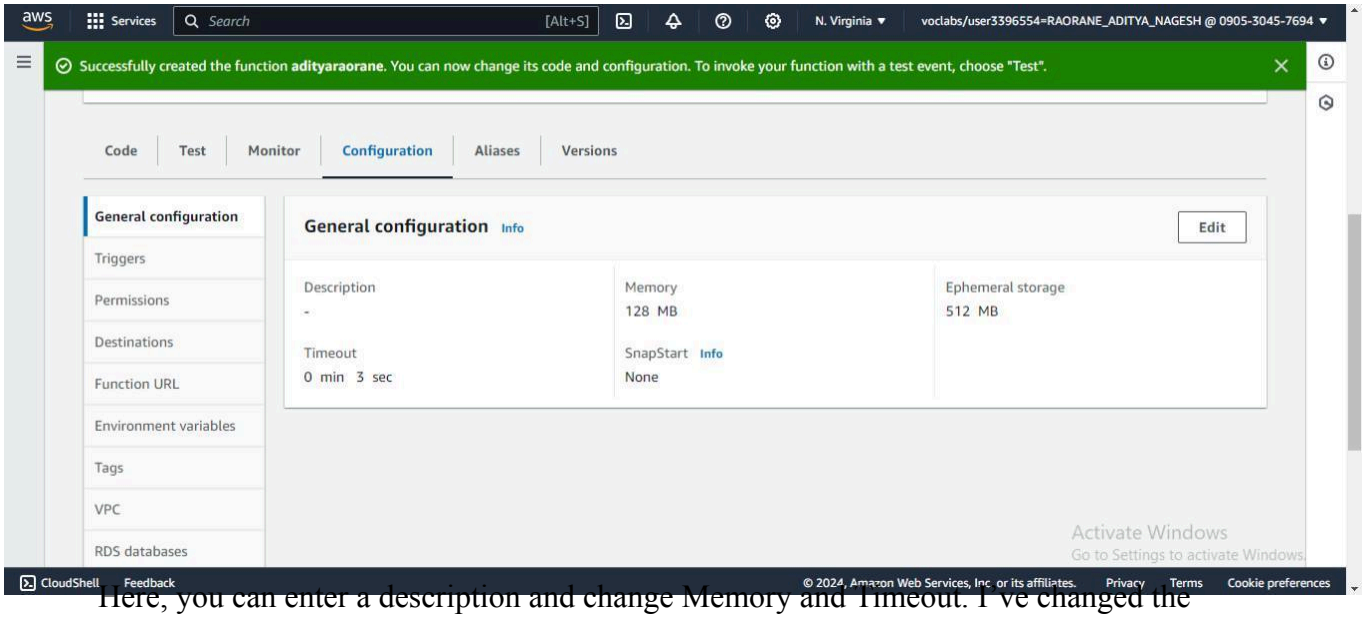
Click on the *Create* button.

3. This process will take a while to finish and after that, you'll get a message that your function was successfully created.

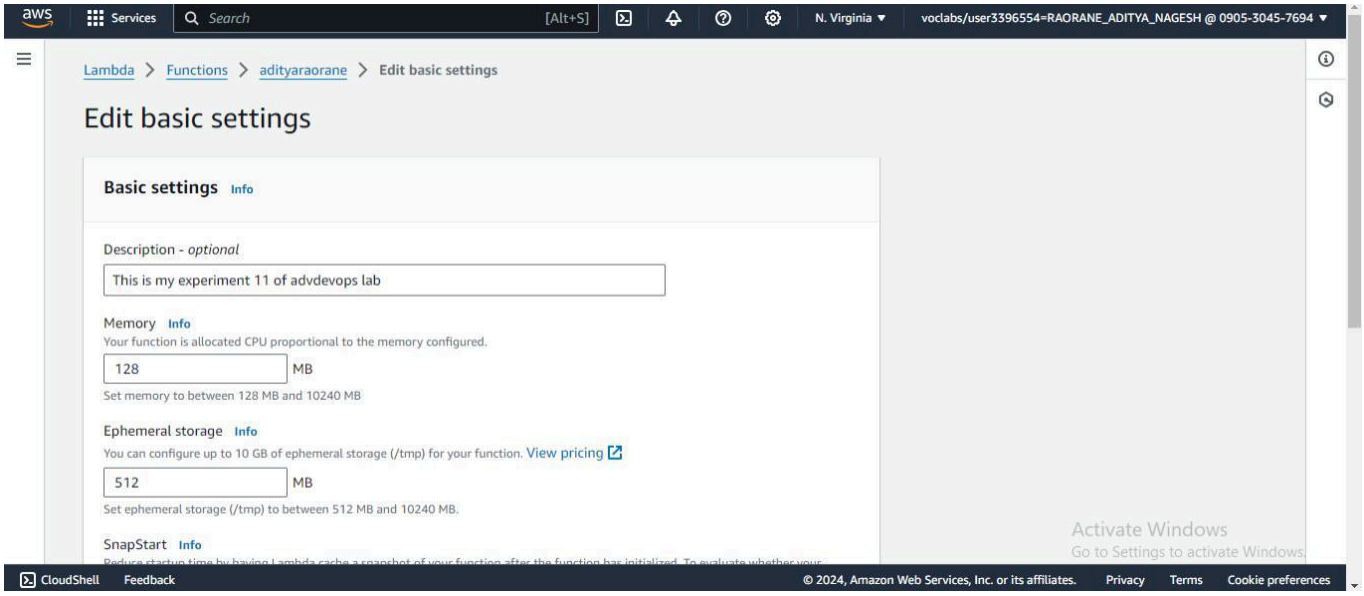


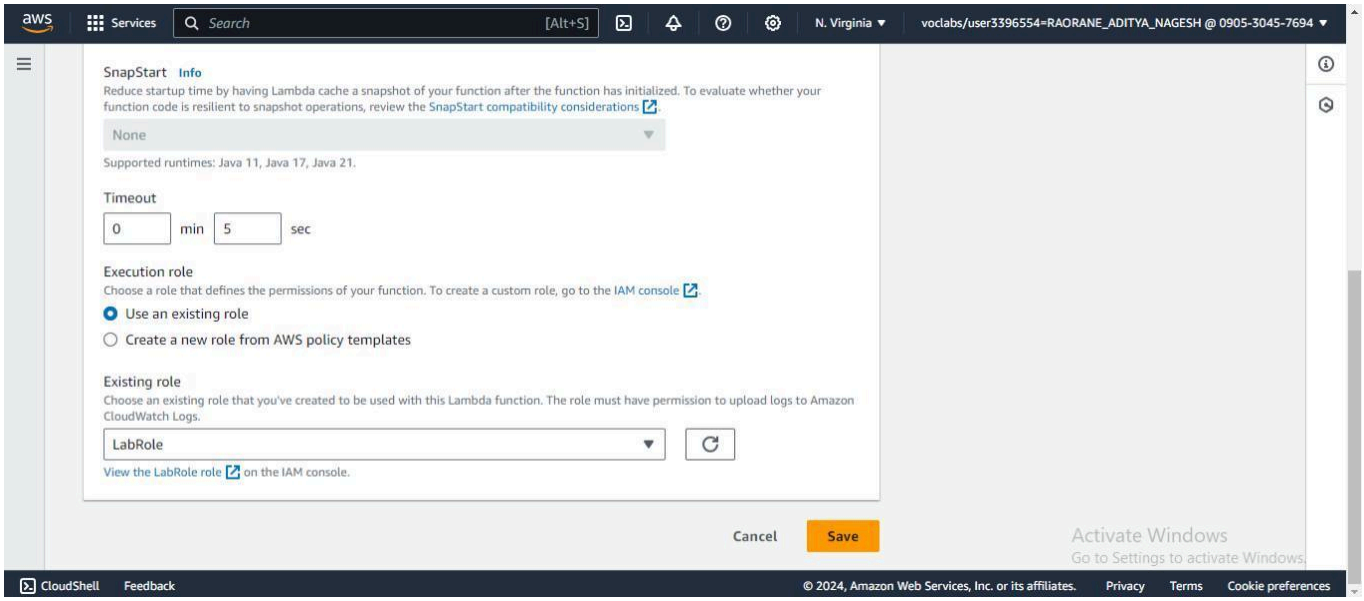


4. To change the configuration, open up the Configuration tab and under General Configuration, choose Edit.

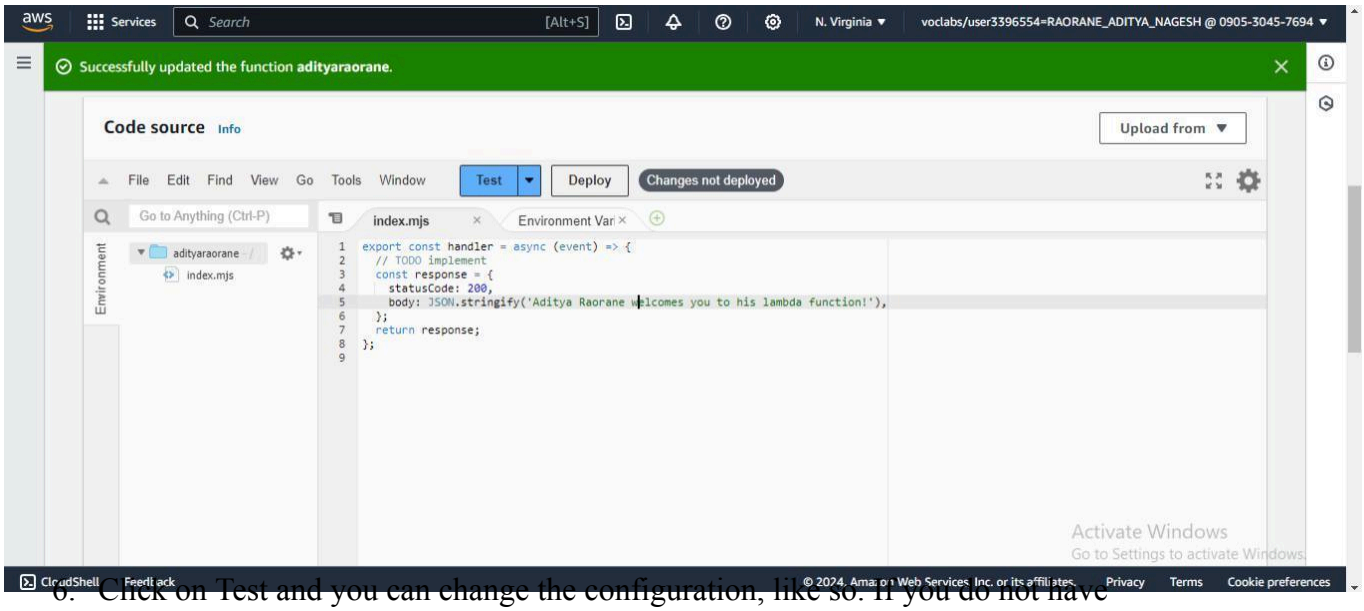


Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 5 sec since that is sufficient for now.

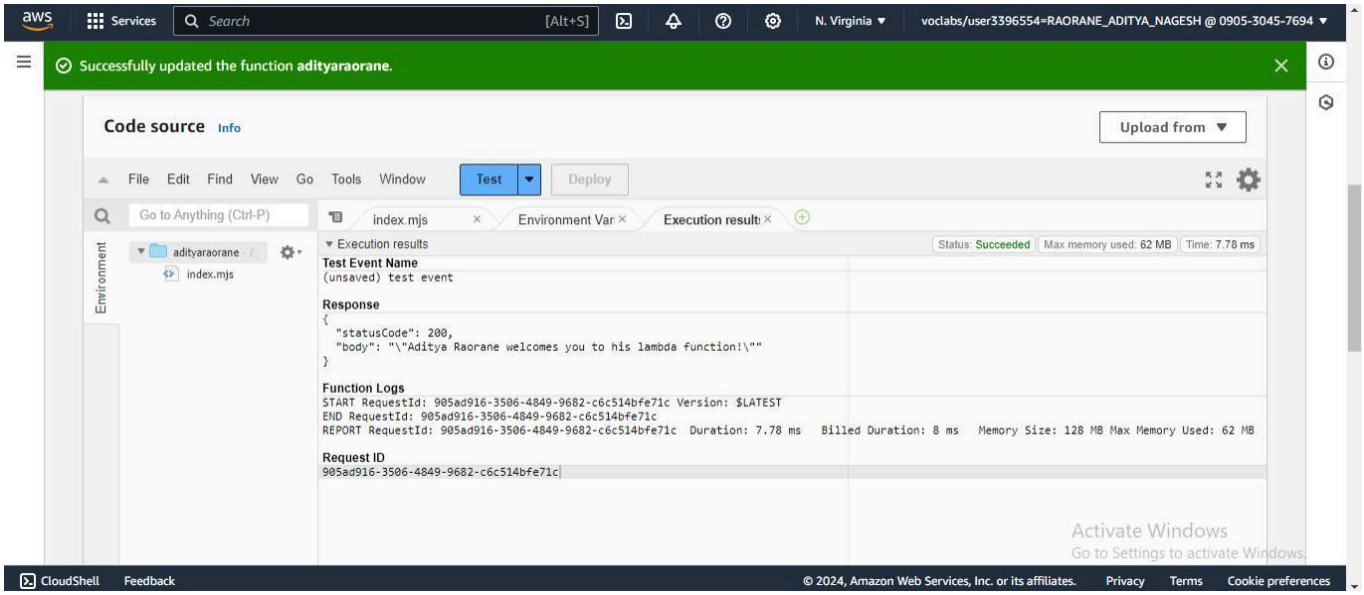




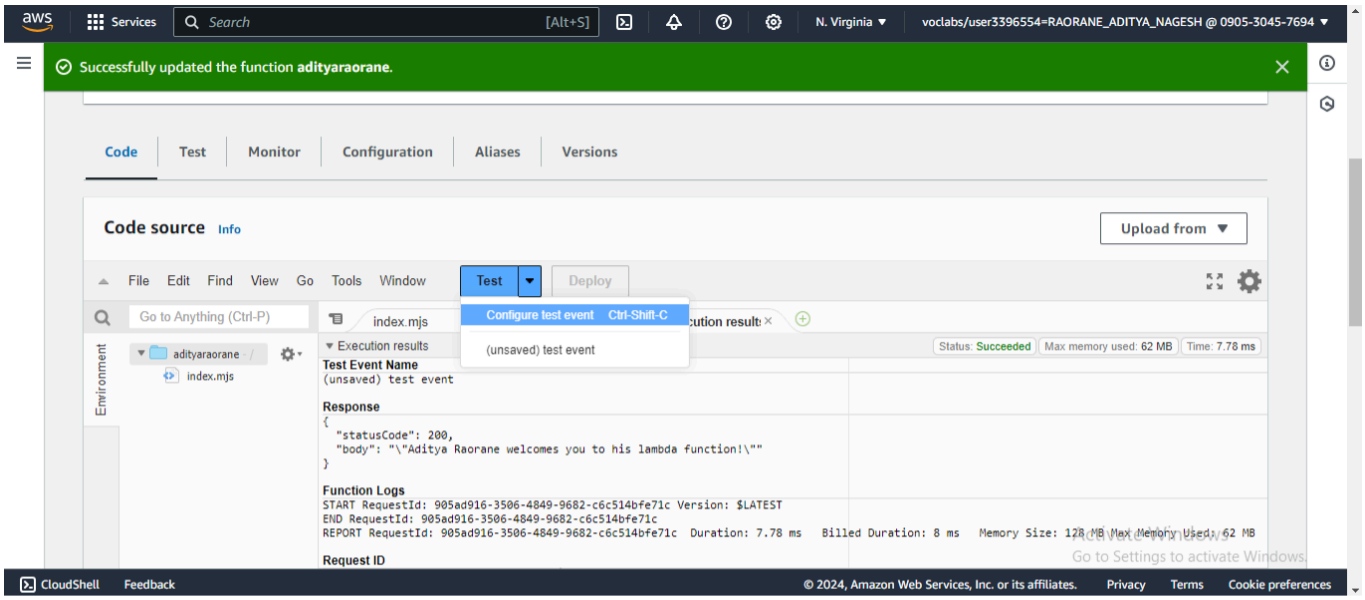
5. You can make changes to your function inside the code editor. You can also upload a zip file of your function or upload one from an S3 bucket if needed.
   Press Ctrl + S to save the file and click Deploy to deploy the changes.

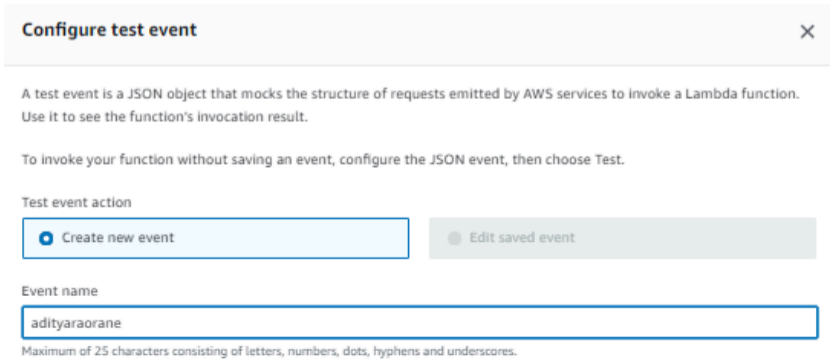


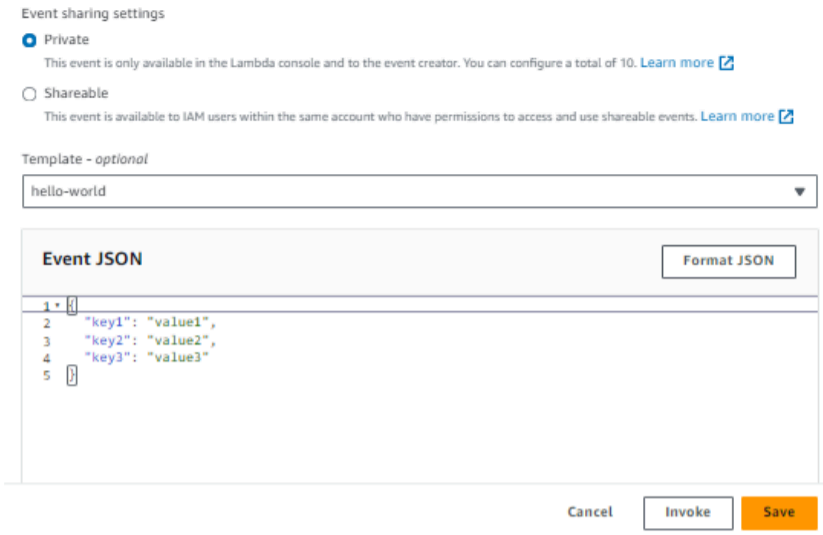
6. Click on Test and you can change the configuration, like so. If you do not have anything in the request body, it is important to specify two curly braces as valid JSON, so make sure they are there.

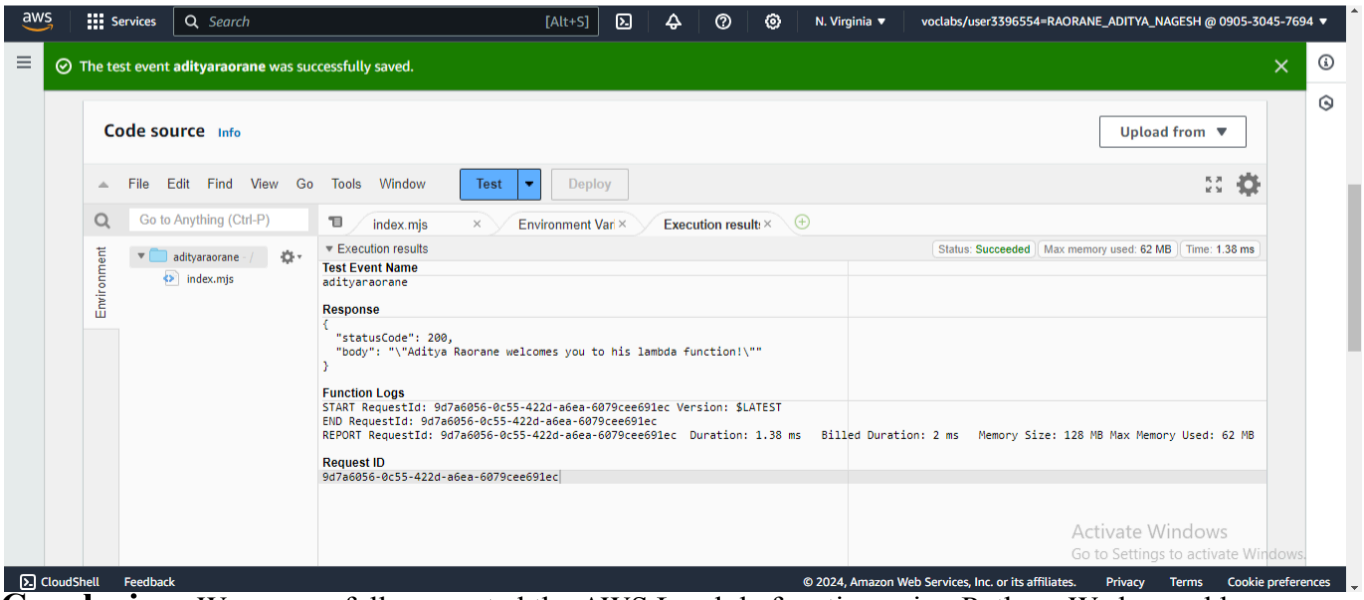


7. Now click on Test and you should be able to see the results.







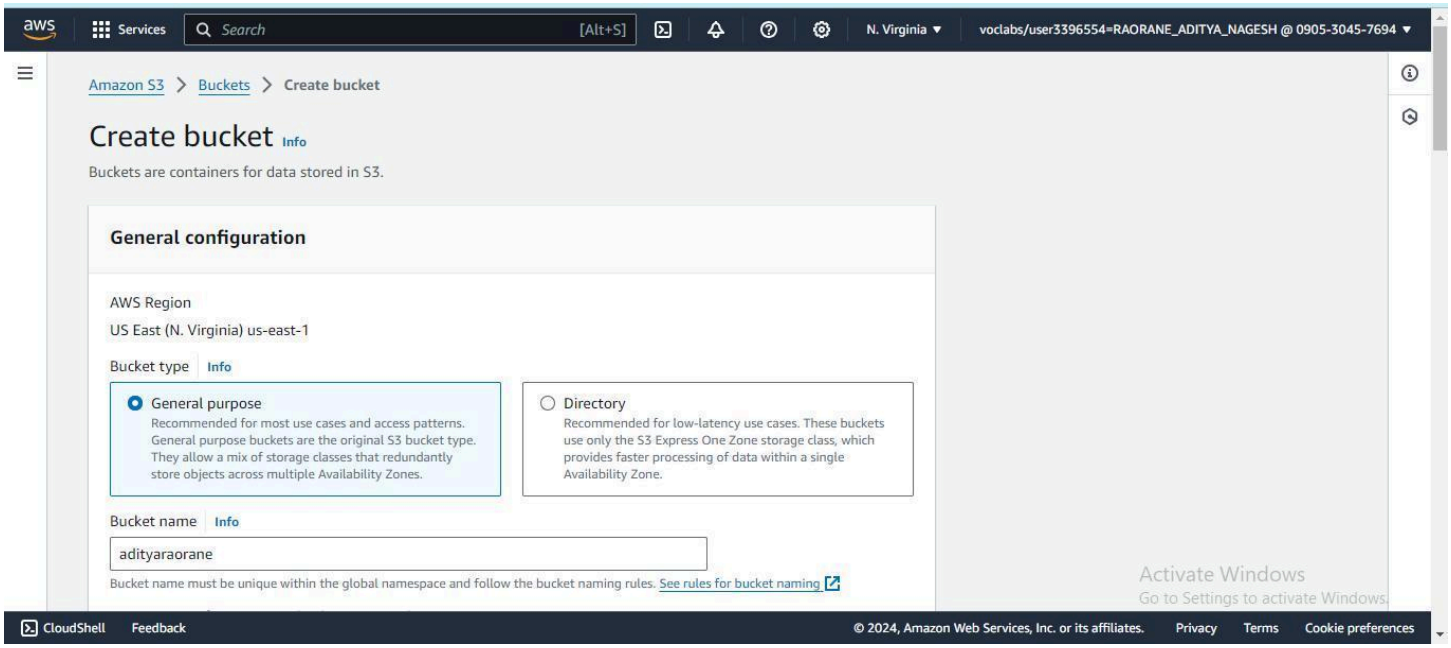


Conclusion: We successfully executed the AWS Lambda function using Python. We learned how AWS Lambda allows you to run code without provisioning or managing servers, making it a cost-effective and scalable solution for serverless computing. The workflow involves uploading code, setting triggers, and letting Lambda manage the execution based on the provided conditions.AWS Lambda's event-driven architecture makes it highly efficient for handling real-time data processing and automation tasks.

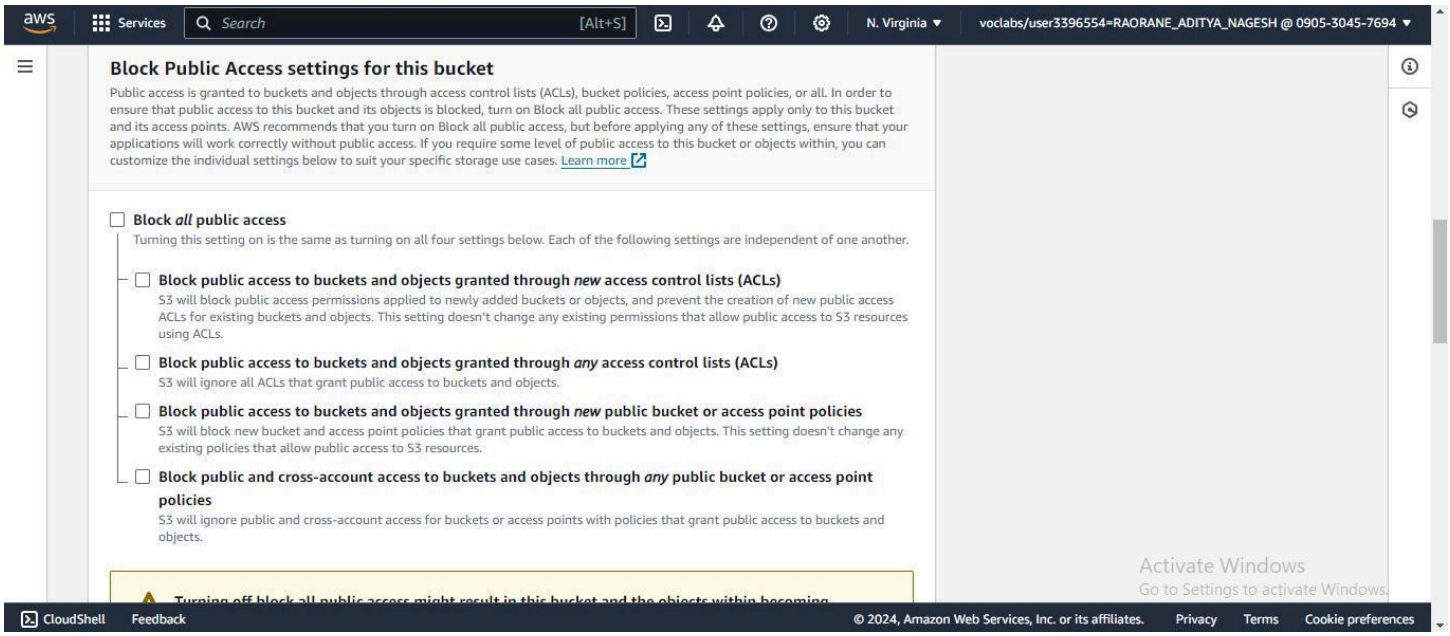
Experiment 12

Aim: To create a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3.

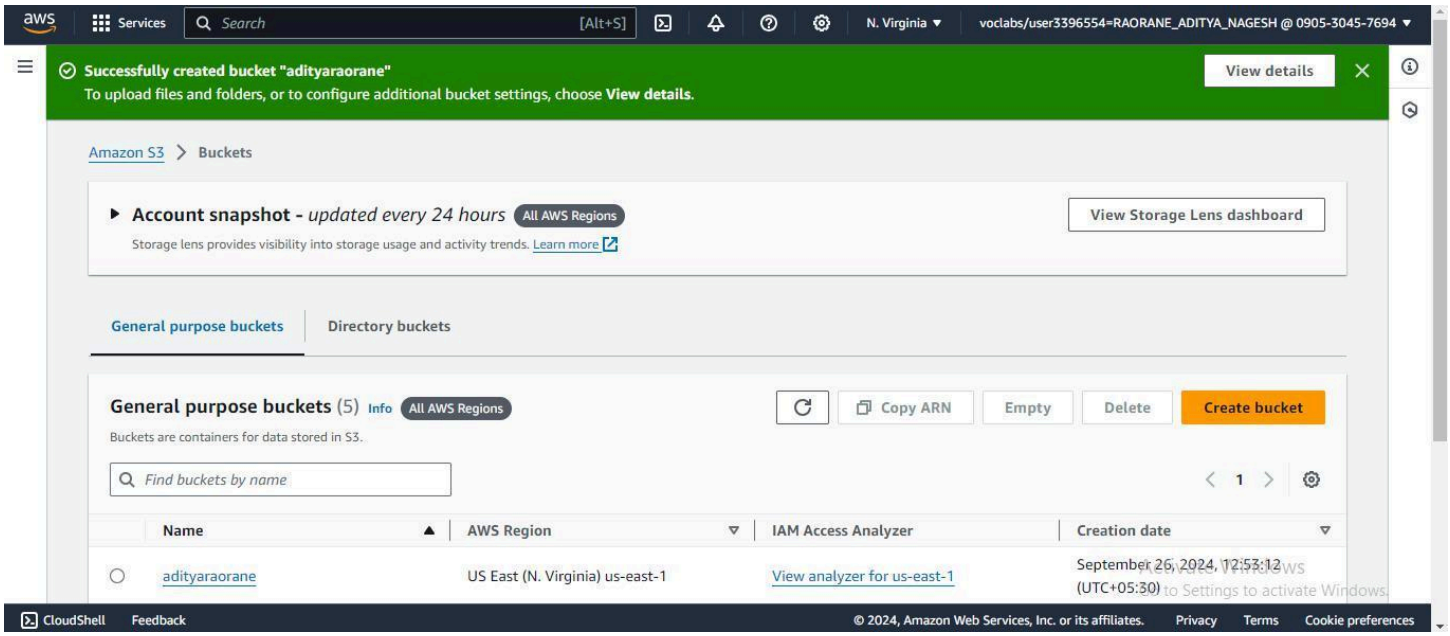
1] Create a AWS S3 Bucket type **General Purpose** named **adityaraorane**.



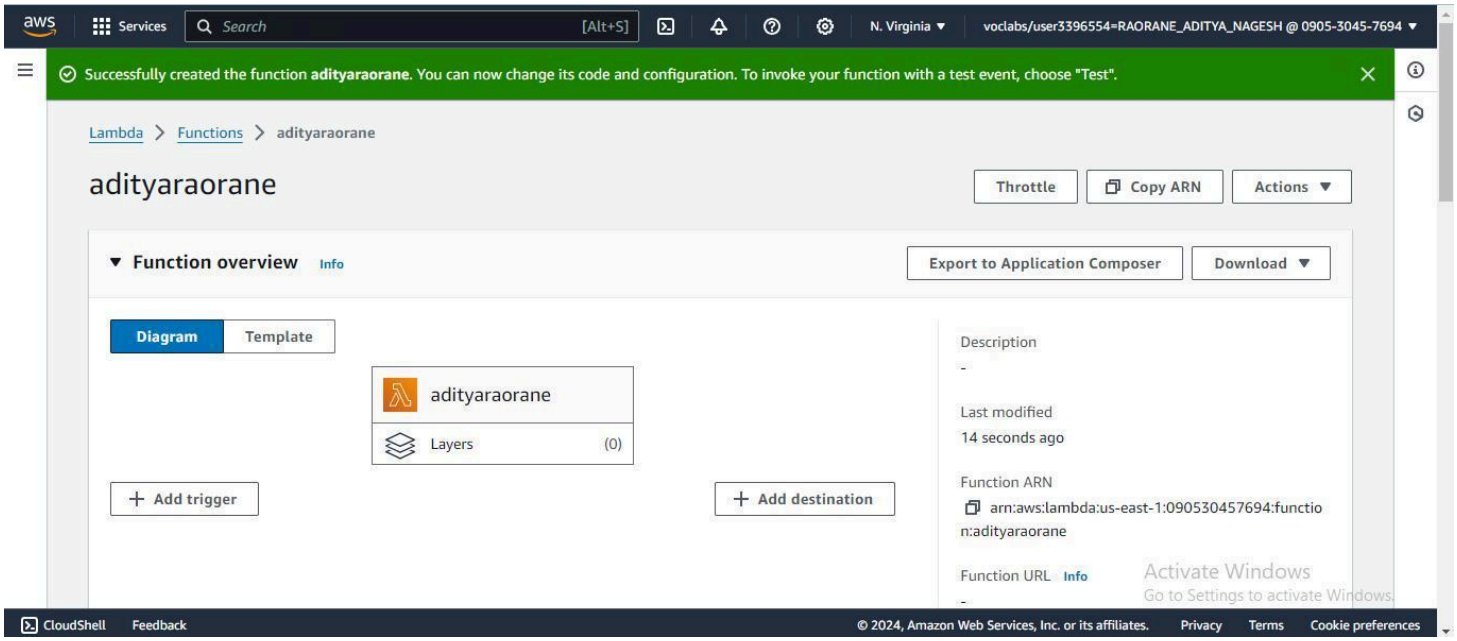
2] Turn off the **Block Public Access for this bucket** so that it is publicly accessible to all users.



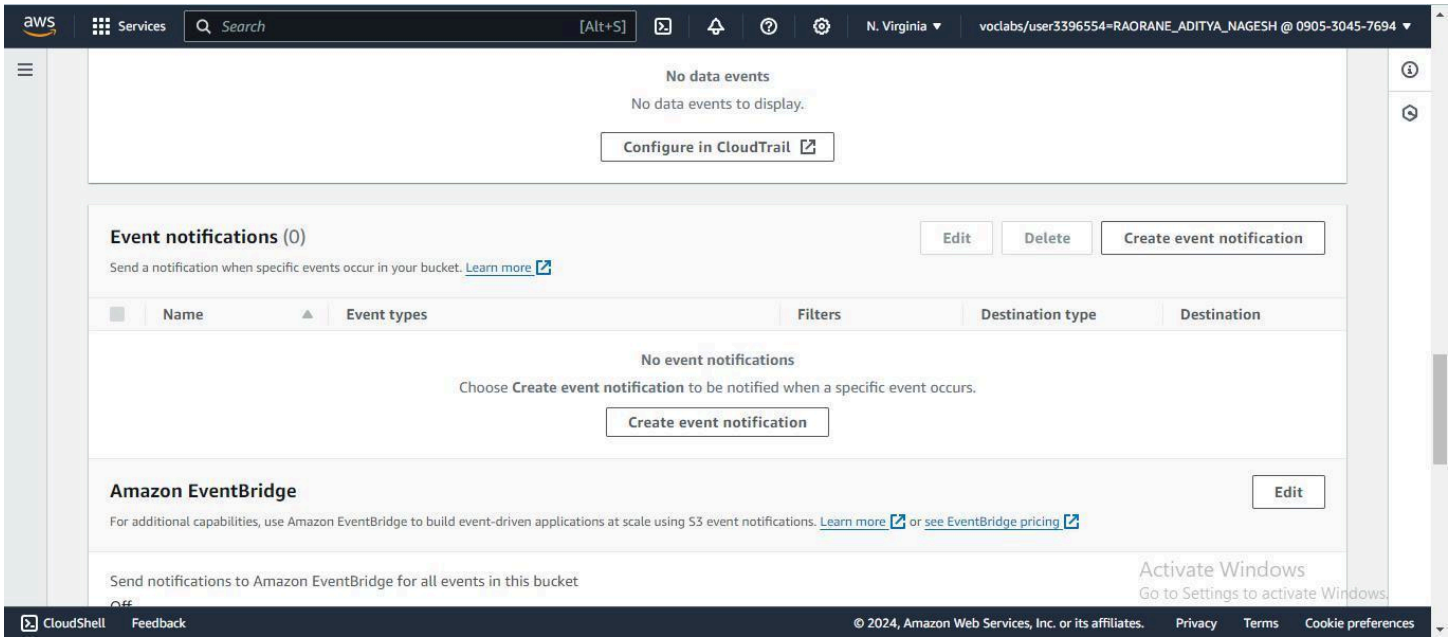
3] Click on the **Create** button. This will successfully create our bucket. Then click on the bucket name **adityaraorane** to open it up.



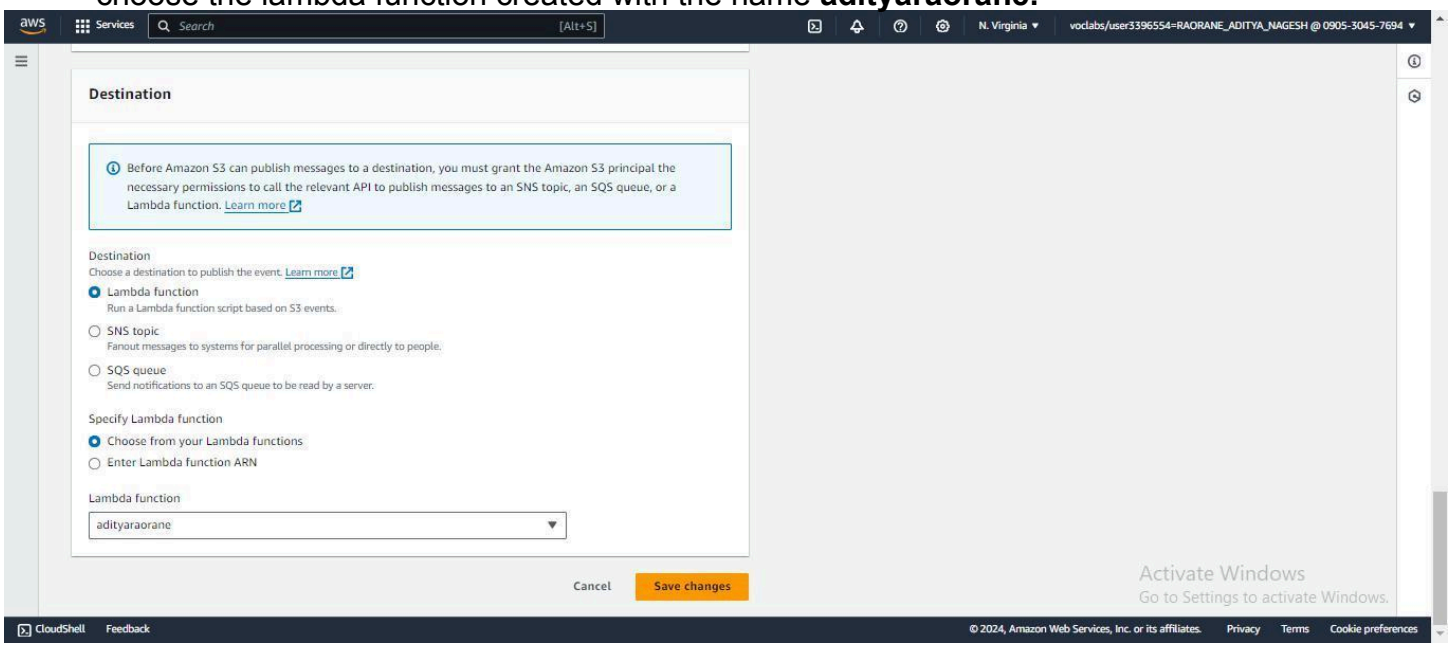
4] Now create a Lambda function named **adityaraorane**.



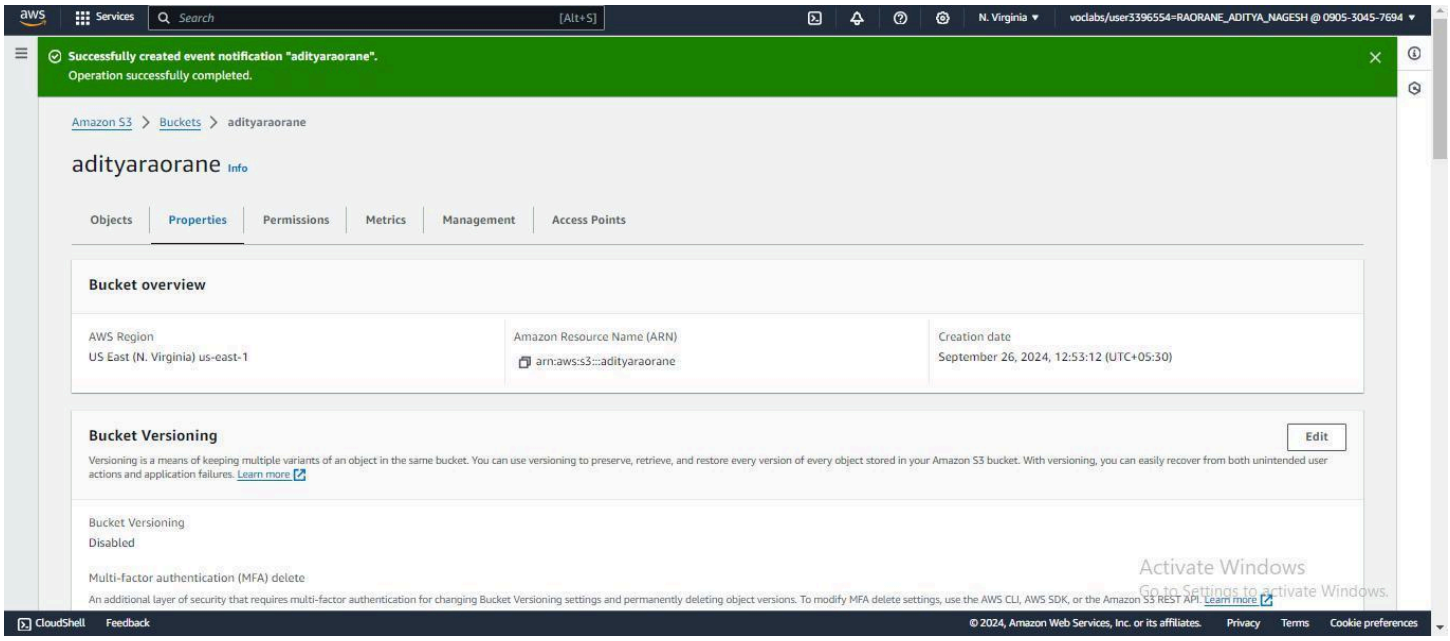
5] Click on **Services** and select **CloudWatch** which is an AWS service that helps you enable operational and risk auditing, governance, and compliance of your AWS account.



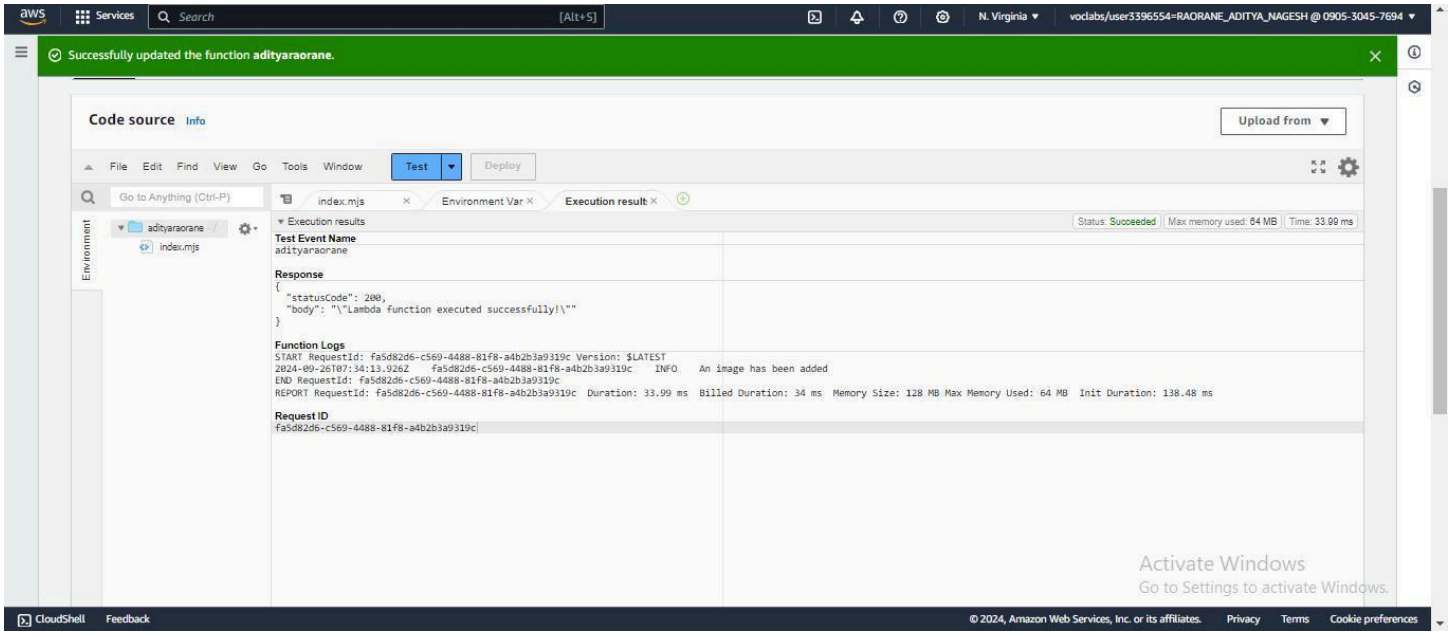
6] Click on **Create event notification** and select **Destination** as **Lambda function** and choose the lambda function created with the name **adityaraorane.**



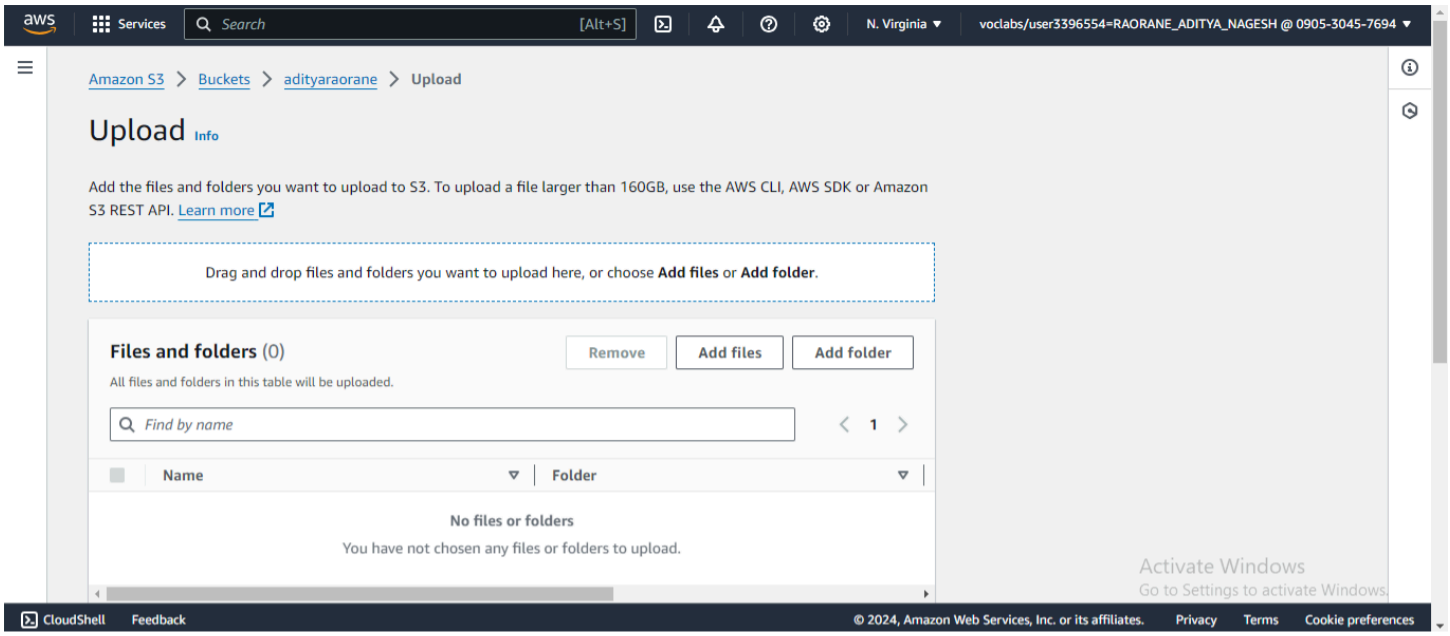
7] Click on **Save Changes**.

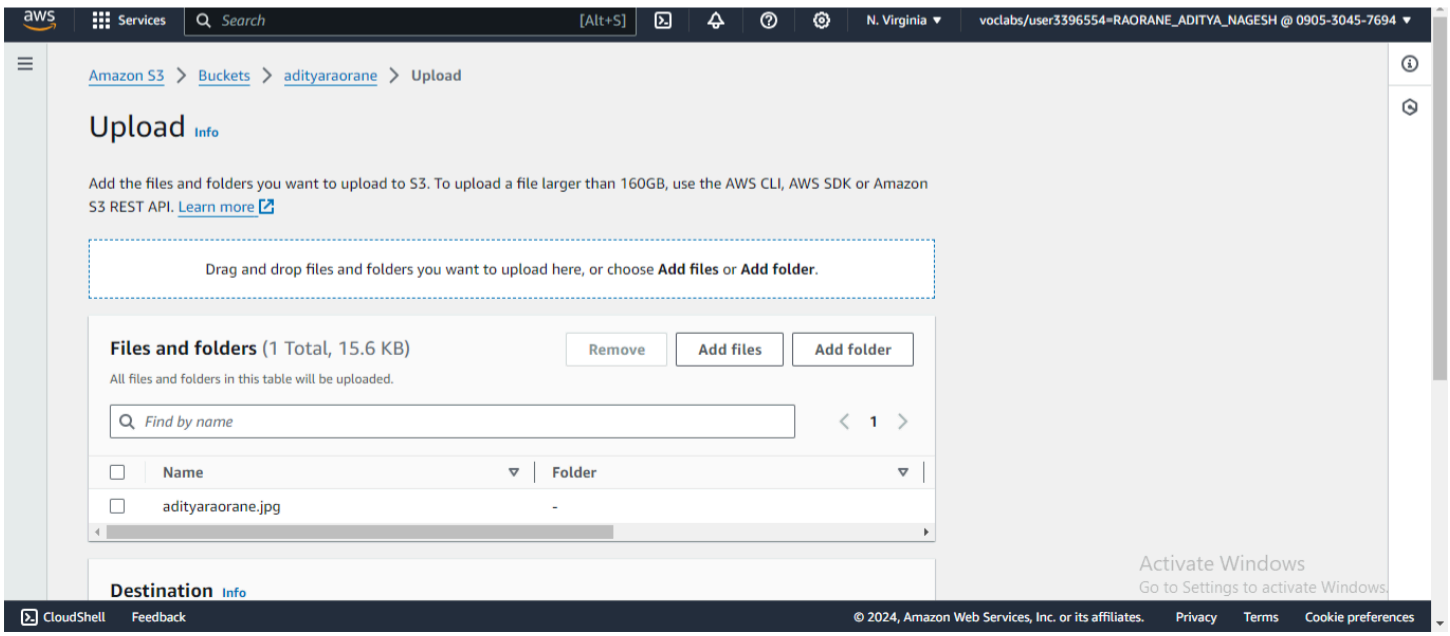


8] **Deploy** the code and **Test** it to check whether the lambda function executed successfully.

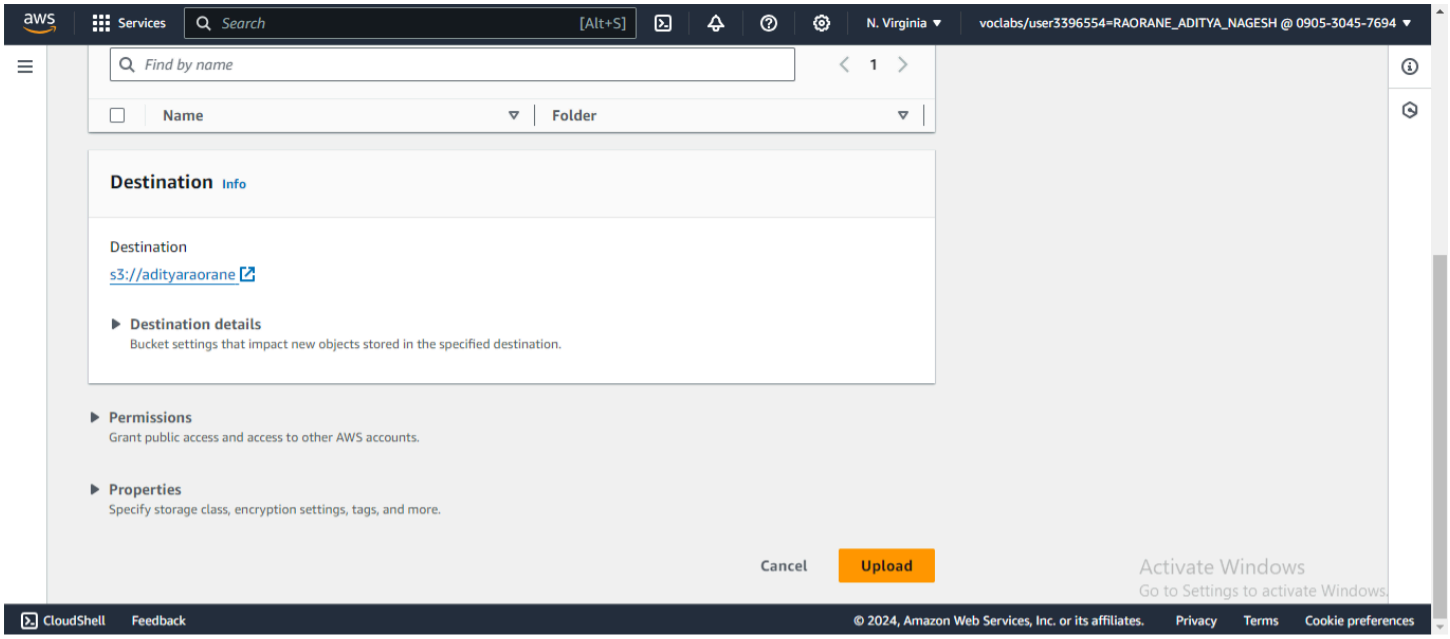


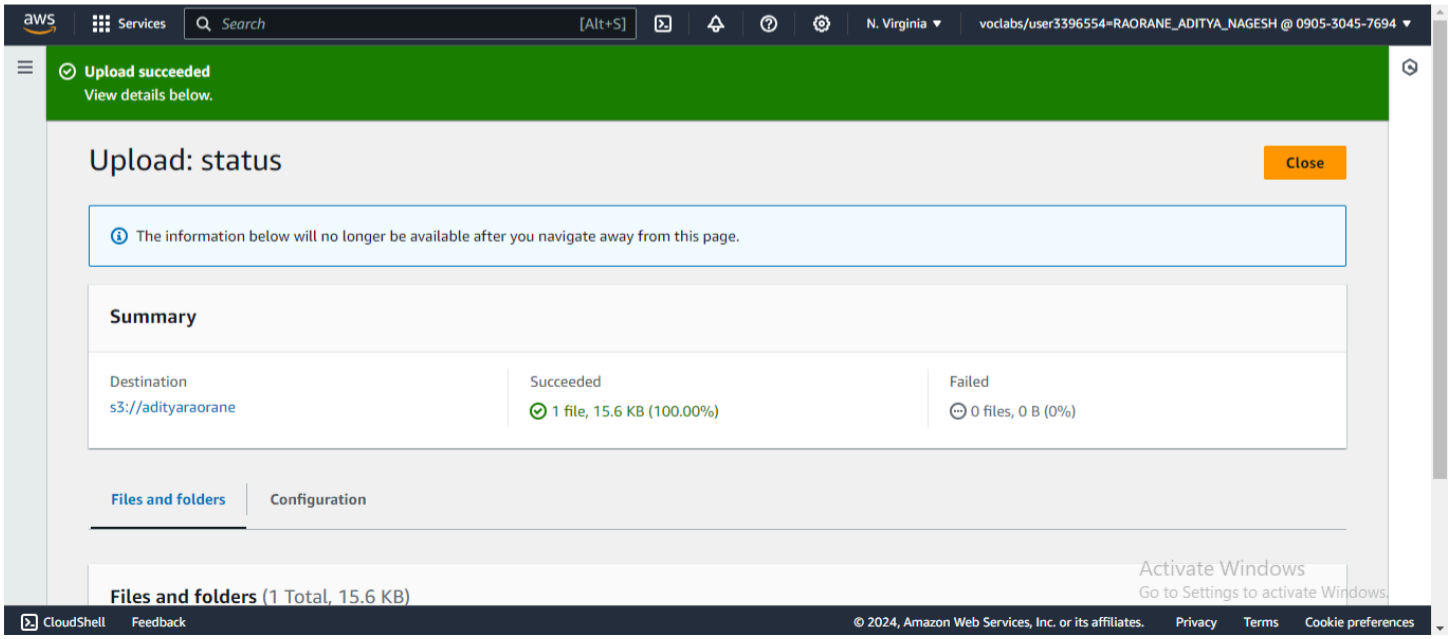
9] Upload a **.jpg** image on the S3 bucket created and click on **Upload**.

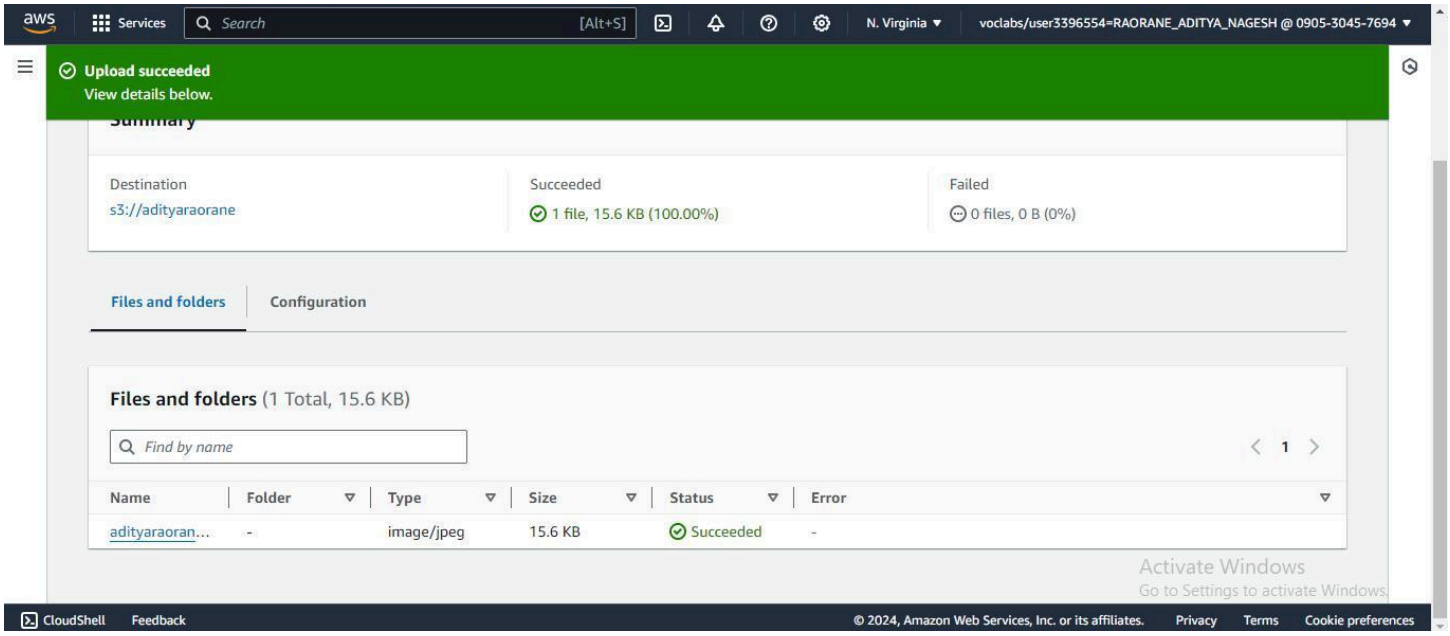




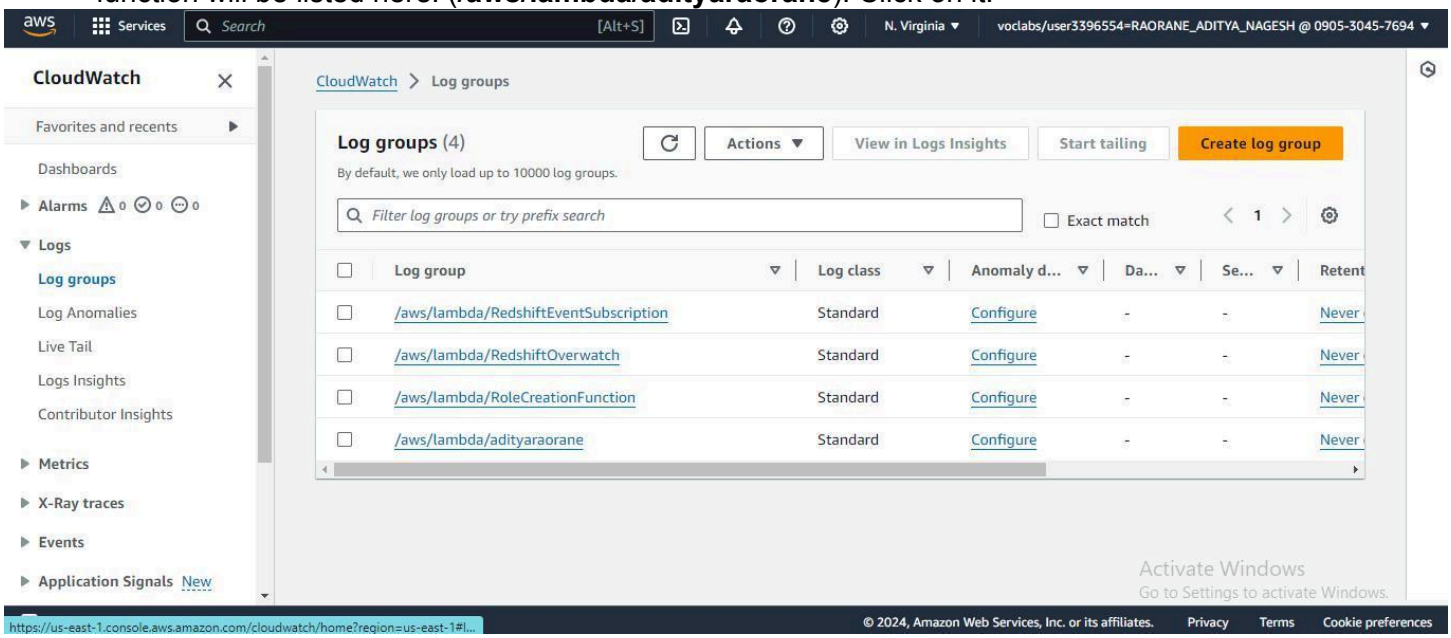
Name: Aditya Nagesh Raorane Class: D15C/ Batch B Roll No: 44

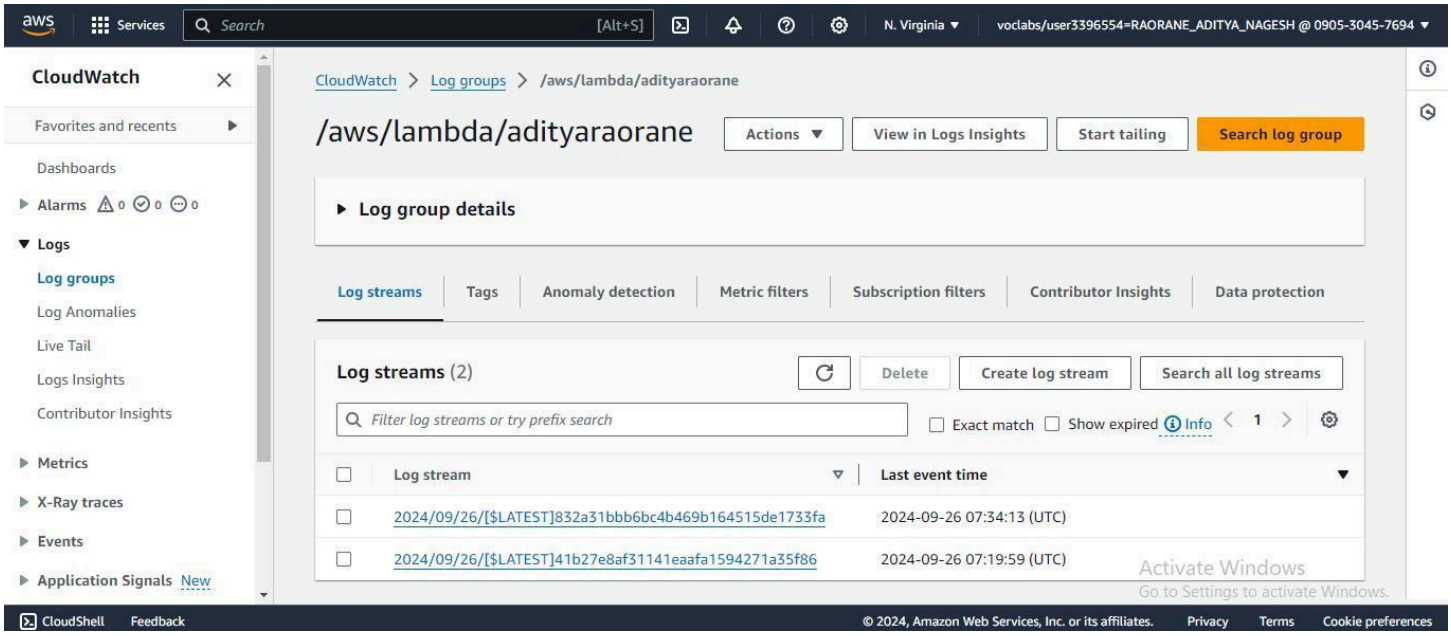






10] In the **CloudWatch** services ,in the left navigation pane select the **Log groups**. Our lambda function will be listed here. (**/aws/lambda/adityaraorane**). Click on it.





Conclusion: Integrating AWS Lambda with S3 allows for real-time, automated processing of events such as file uploads. In this example, a Lambda function is configured to log a message whenever an image is added to a specific S3 bucket. This setup demonstrates the power and flexibility of serverless computing by automating tasks without requiring manual intervention or server management. By leveraging AWS Lambda, developers can efficiently handle event-driven workflows, reduce operational overhead, and quickly deploy scalable solutions that respond to specific actions within cloud environments.