Sannidhi Kailaje / D13A / 22

Experiment 1

AIM: TO understand the benefits of cloud infras and setup AWS Cloud9 IDE, launch AWS cloud9 IDE and perform collaboration.

THEORY:

Amazon Web services (AWS) is a comprehensive cloud computing platform provided by Amazon. It offers wide range of cloud-based services, including computer power, storage and databases, machine learning and more.

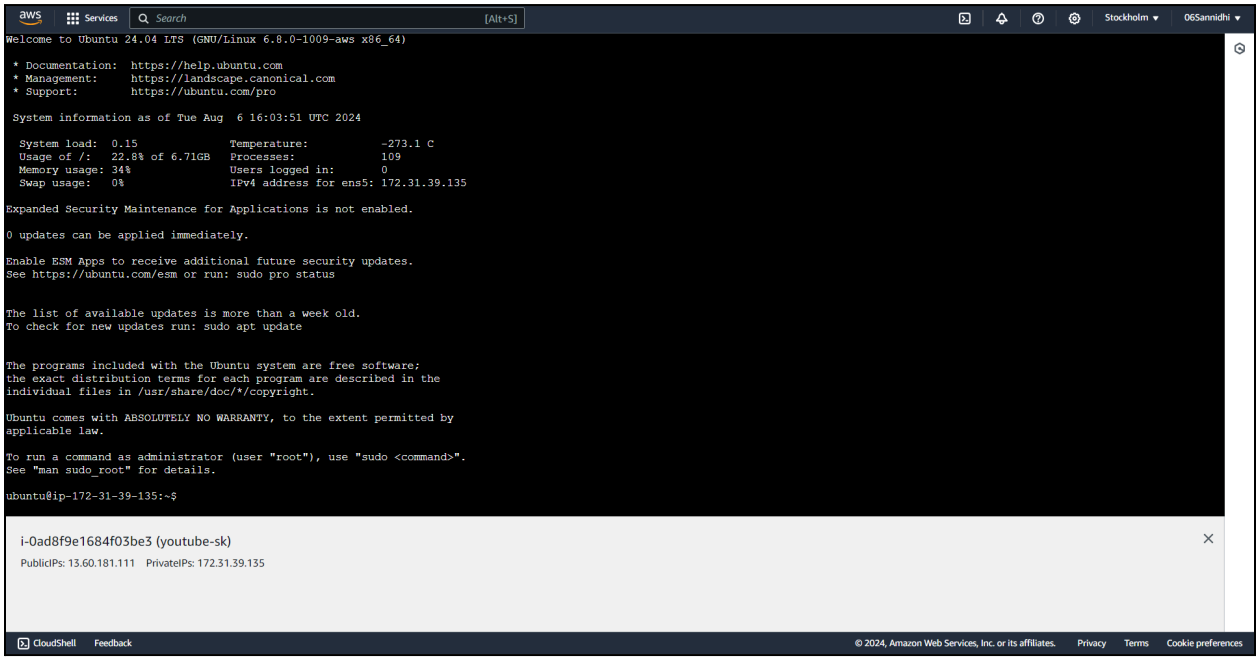
Amazon EC_2 is a core service within AWS that provides scalable virtual servers known as EC2 instance. These instances are designed to handle various workloads from basic web application to high-performance computing tasks. EC2 allows users to launch and manage virtual servers in the cloud providing flexibility in terms of computing power and operating system.

An EC2 instance is a virtual server within the Amazon EC2 service. It represents a single unit of computing capacity in the AWS cloud.

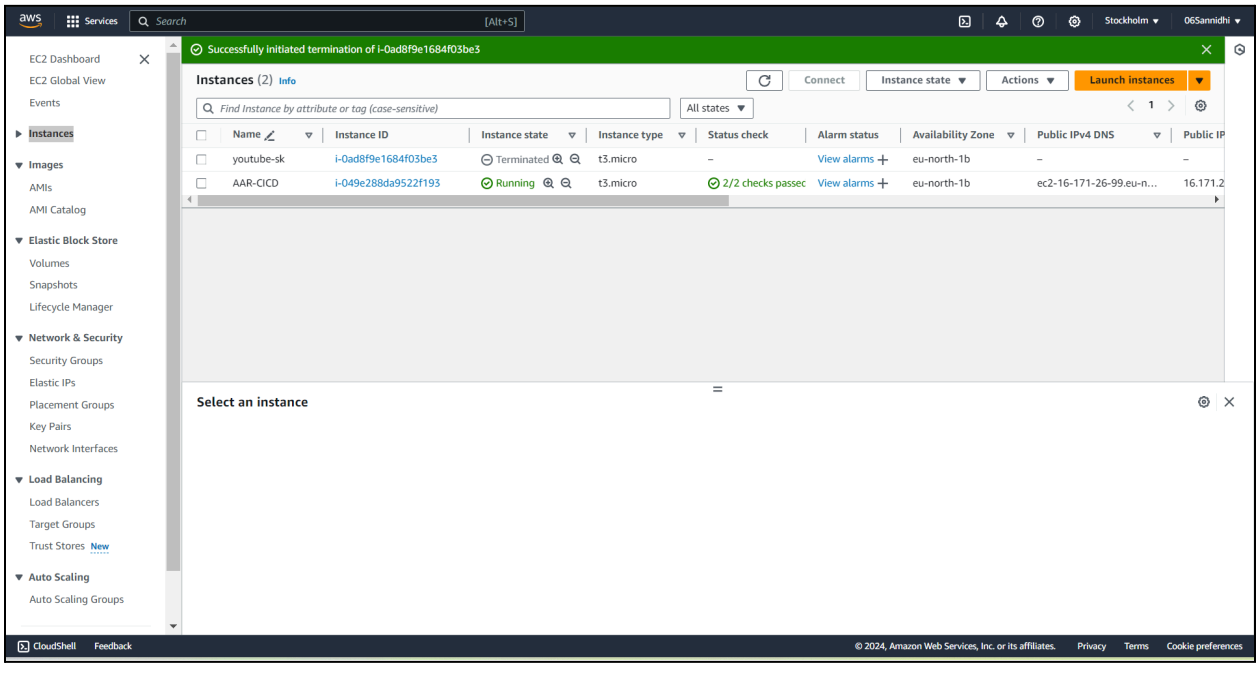
Conclusion: Thus, successfully understood the benefits of cloud infrastructure and setup AWS Cloud9 IDE.

EC2 Instance

Instance launching:



Instance termination:



A) To develop a website and host it on local machine on a VM

EXPLORER

XAMPPWEBPAGE

index.html

pinterest.jpg

index.html × pinterest.jpg

index.html > html > body > a > img

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>skWebsite</title>
    <style >
        body{
            text-align: center;
        }
        img{
            width: 800px;
        }
        img:hover{
            box-shadow: 5px 5px 5px black;
        }
    </style>
</head>
<body>
    <h1 >WELCOME TO SK's WEBSITE</h1>
    <h2>This website is made by sannidhi</h2>
    <p>!!click on the image below to get some art inspiration!!</p>
    <a href="https://in.pinterest.com/#top">
        <img src="pinterest.jpg">
    </a>
    <p>This is a static website hosted using amazon S3</p>
</body>
</html>
```

skWebsite

localhost/xamppwebpage/

WELCOME TO SK's WEBSITE

This website is made by sannidhi

!!click on the image below to get some art inspiration!!

This is a static website hosted using amazon S3

index.html index.php × pinterest.jpg

C: > xampp > htdocs > index.php

```
<?php
    echo "Hello!! This is Sannidhi"
?>

```

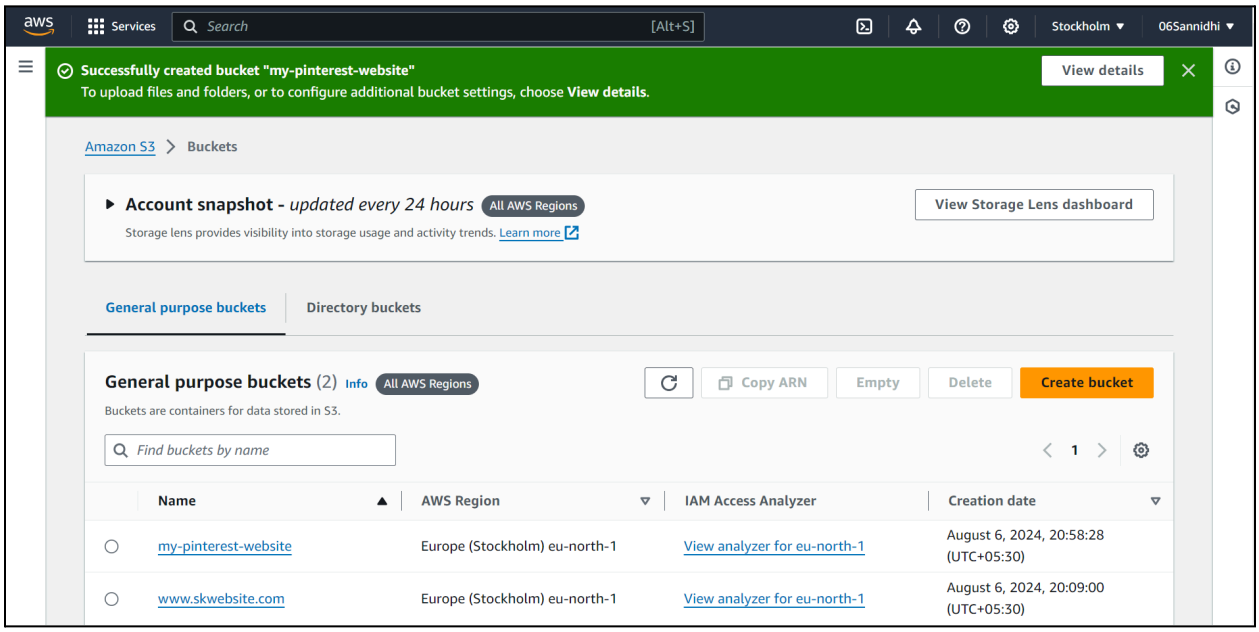
localhost/index.php

localhost/index.php

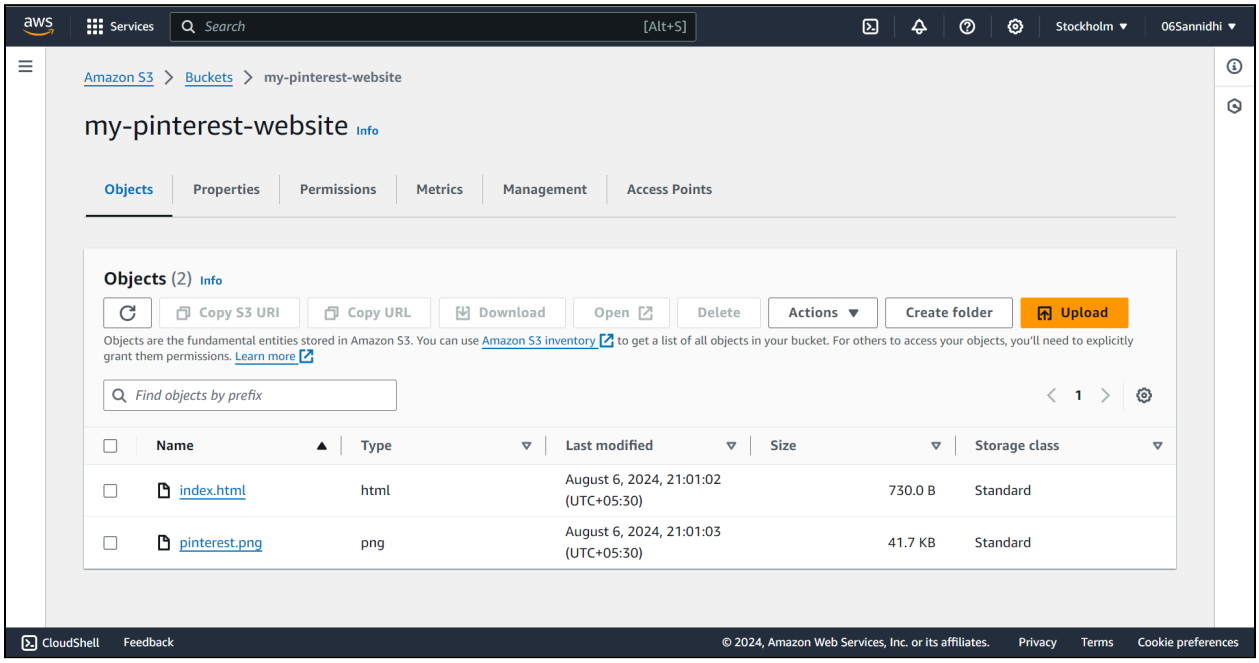
Hello!! This is Sannidhi

B) Hosting a static website on Amazon S3

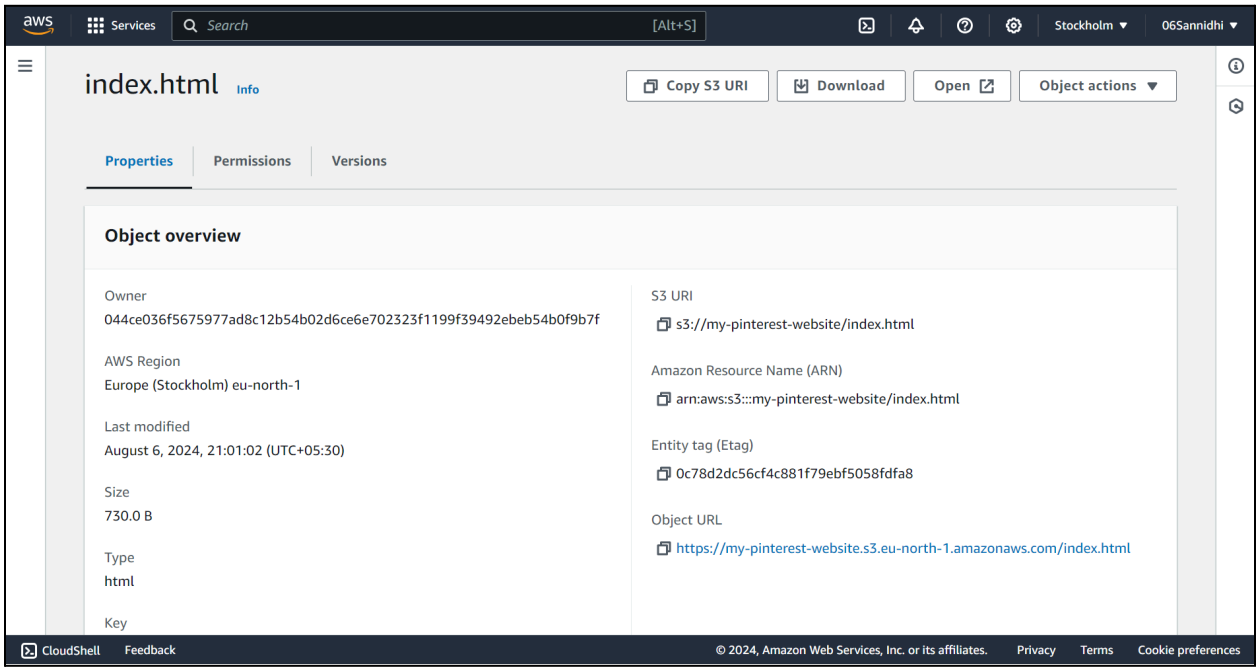
Bucket creation



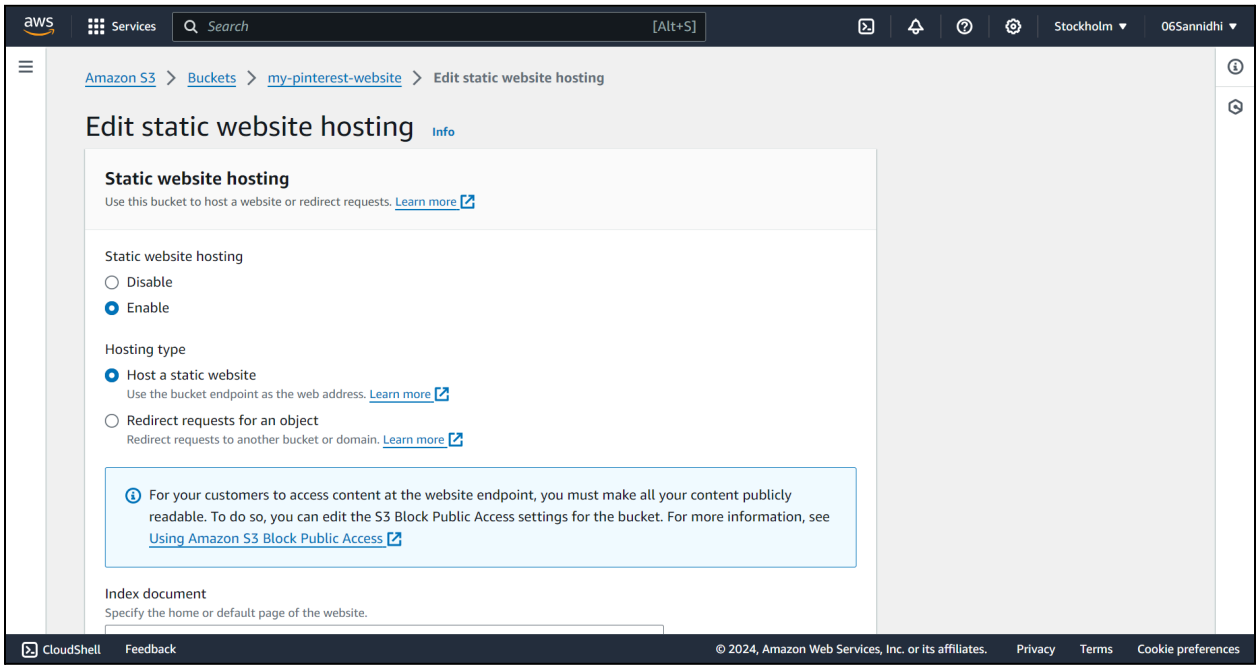
Uploading files



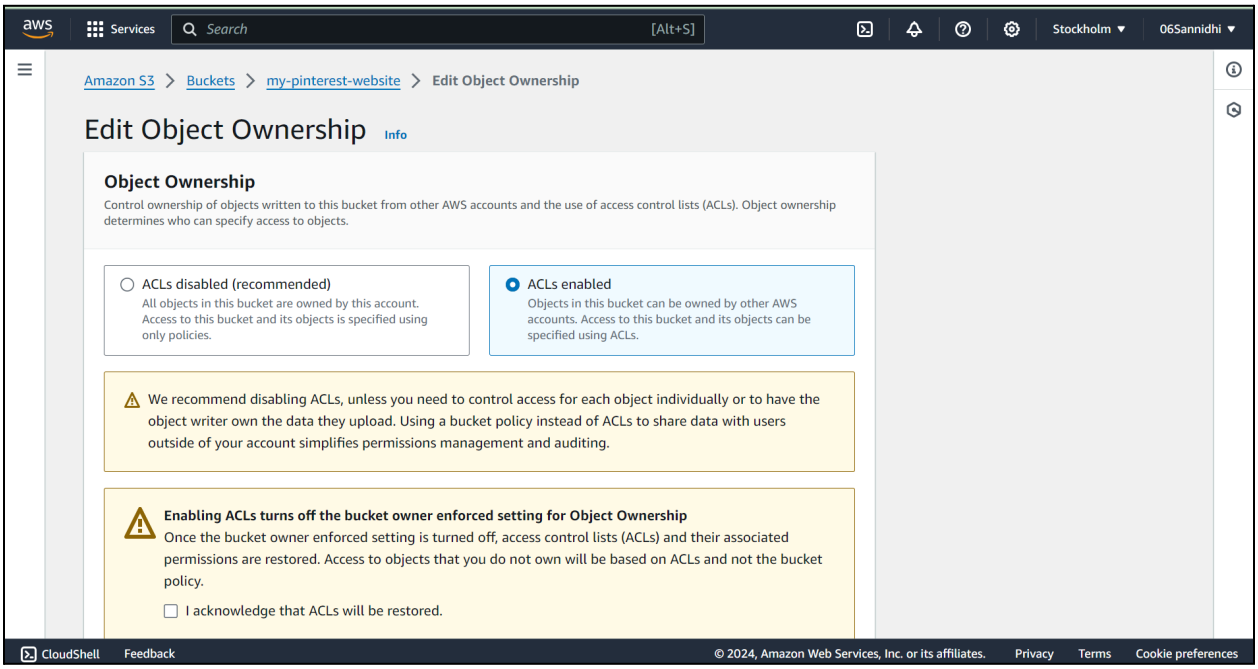
Accessing Index.html file



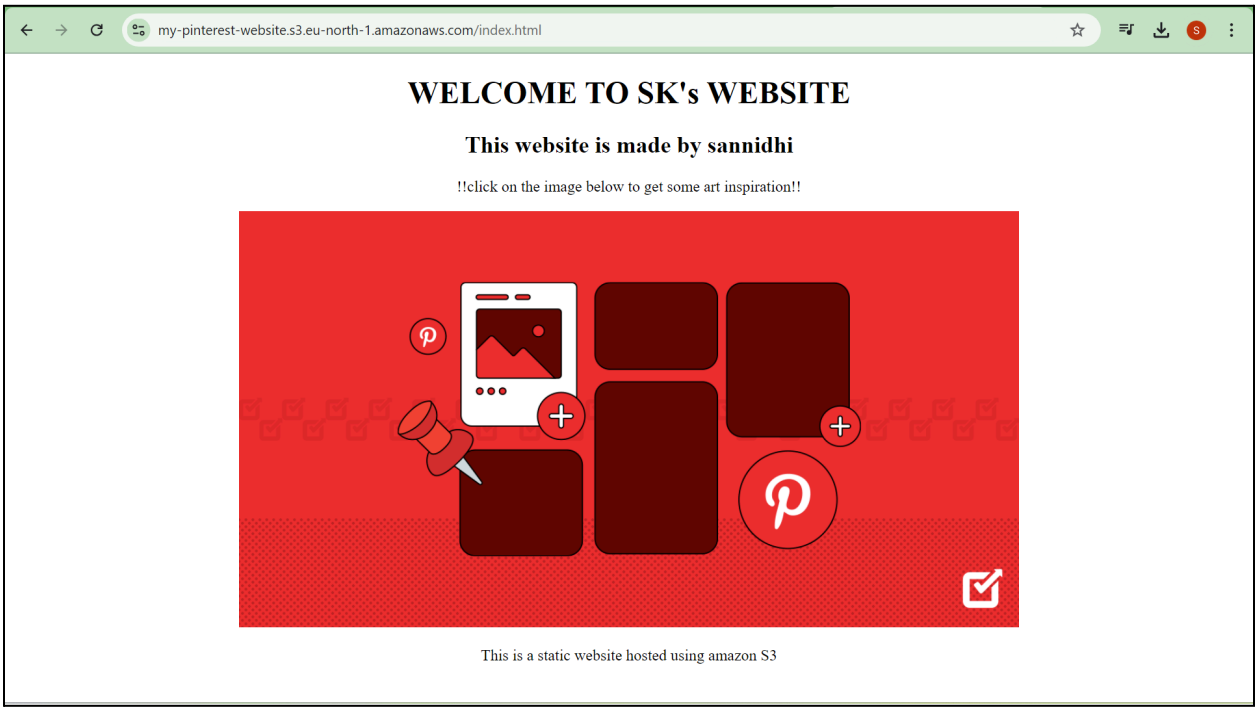
Enabling static website hosting



Enabling object ownership



Hosted Website

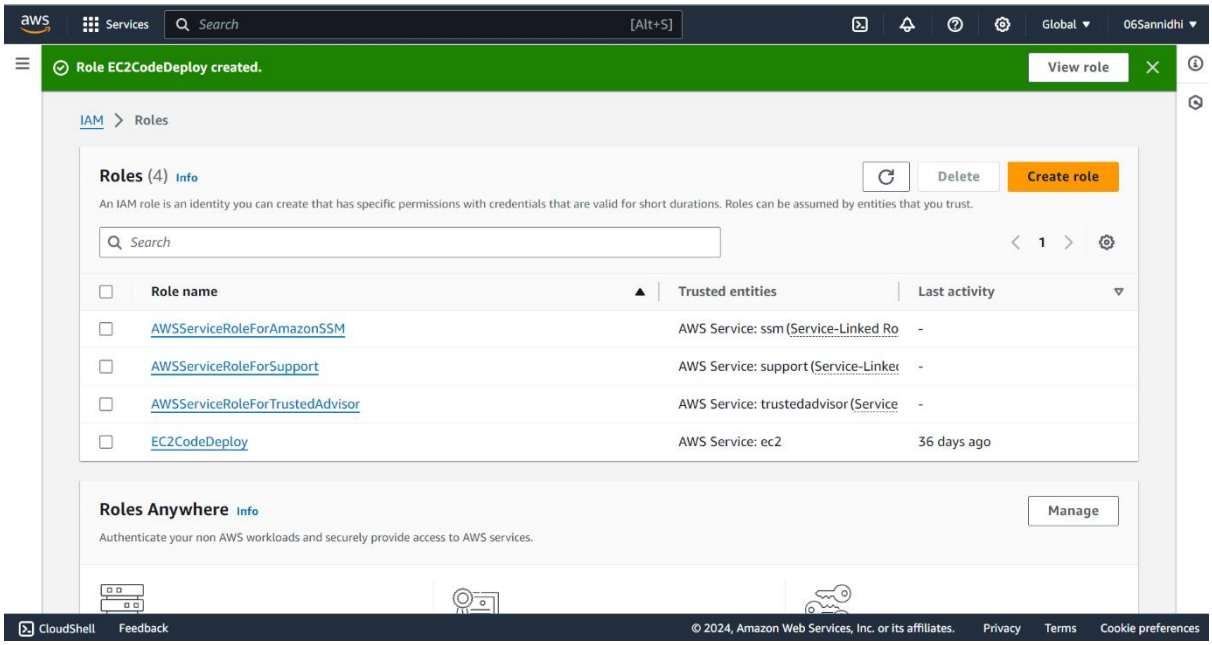


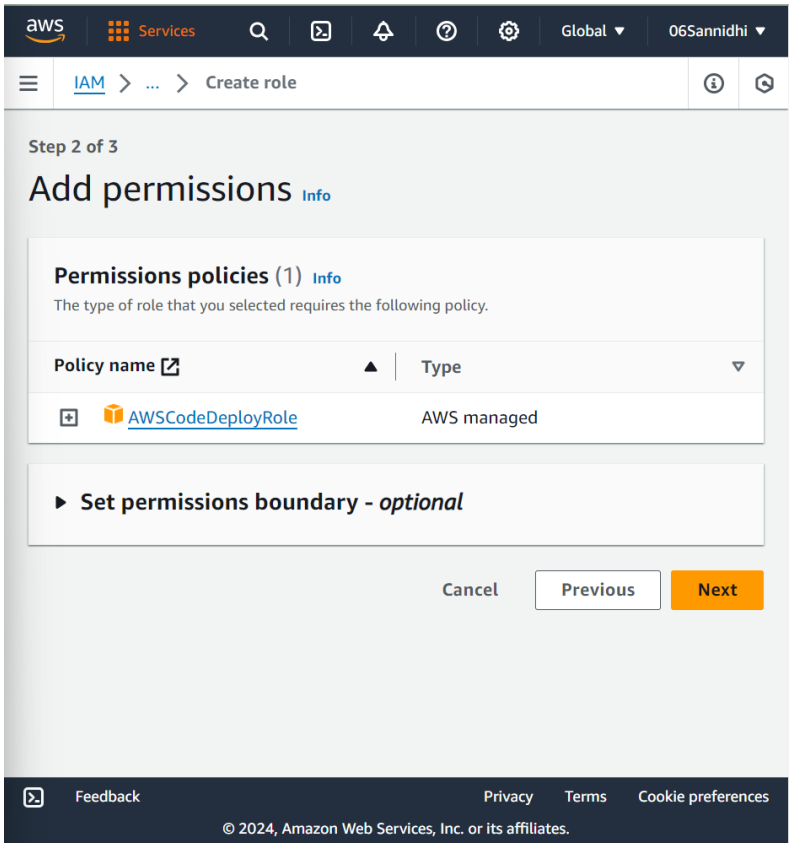
EXPERIMENT-2

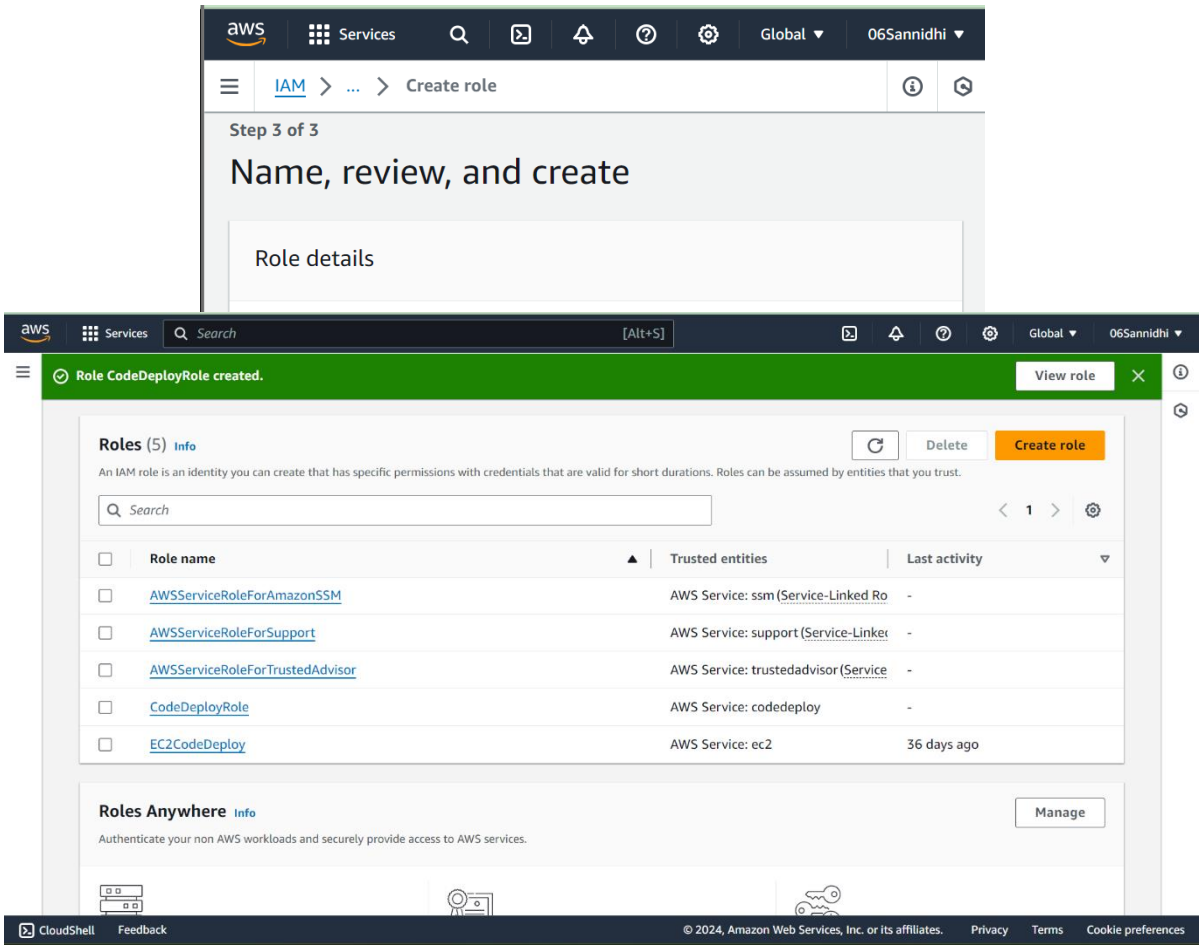
Aim

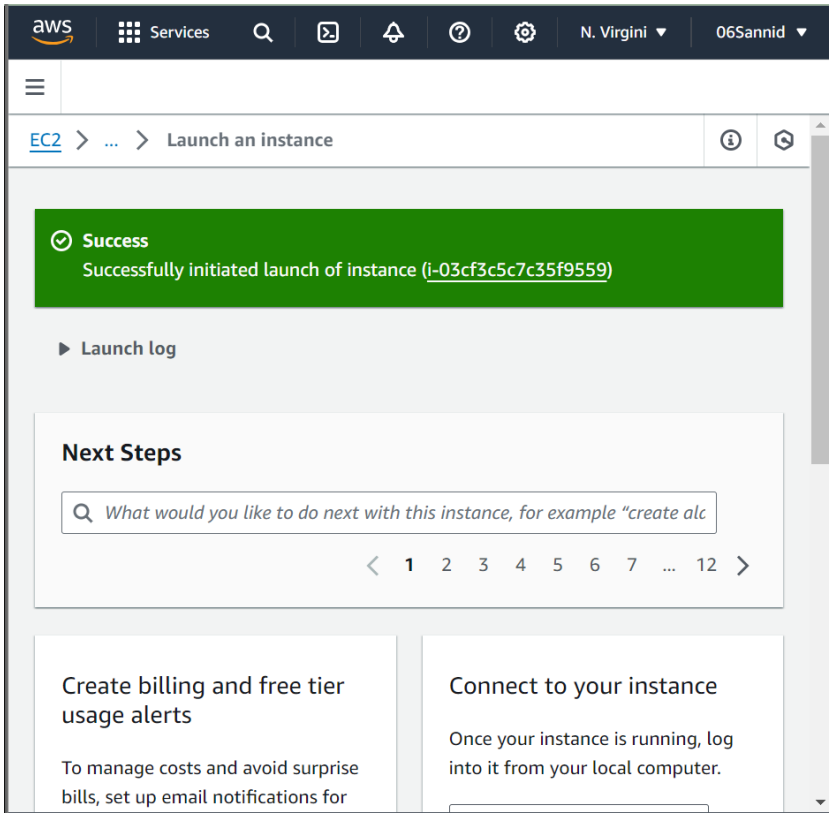
To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.

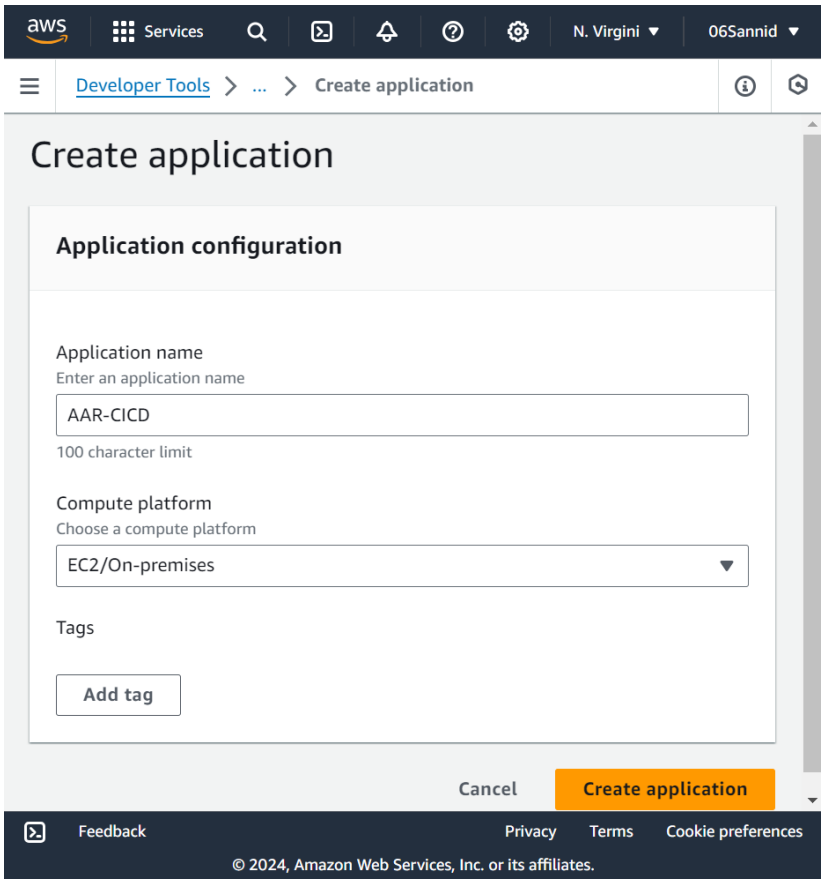
Implementation

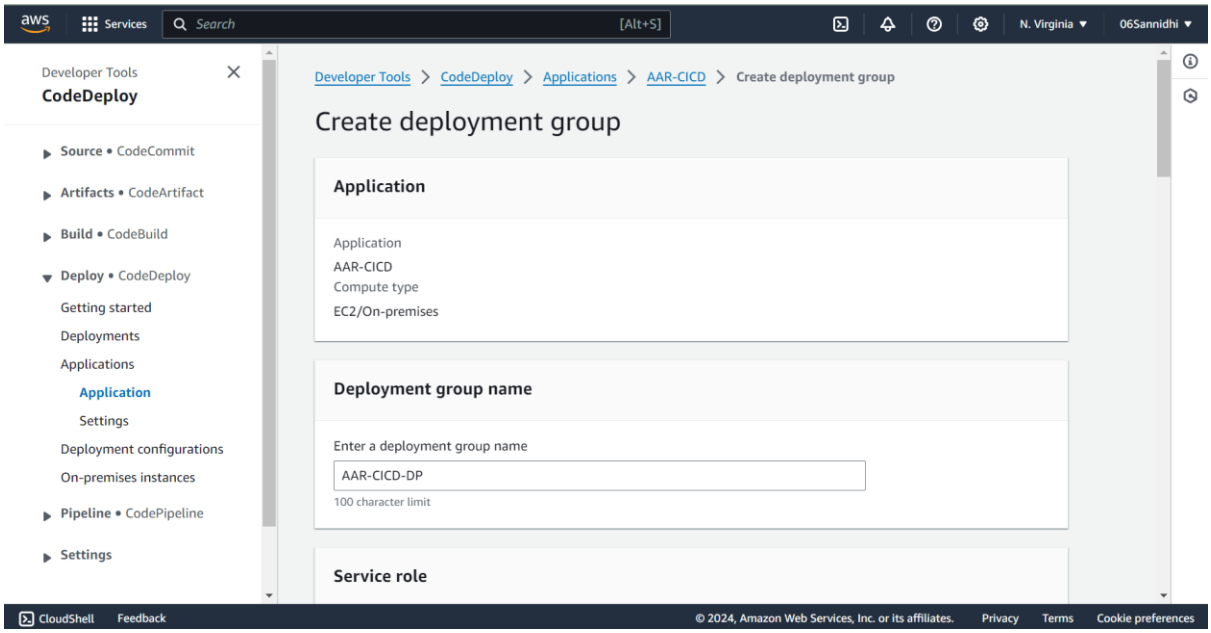


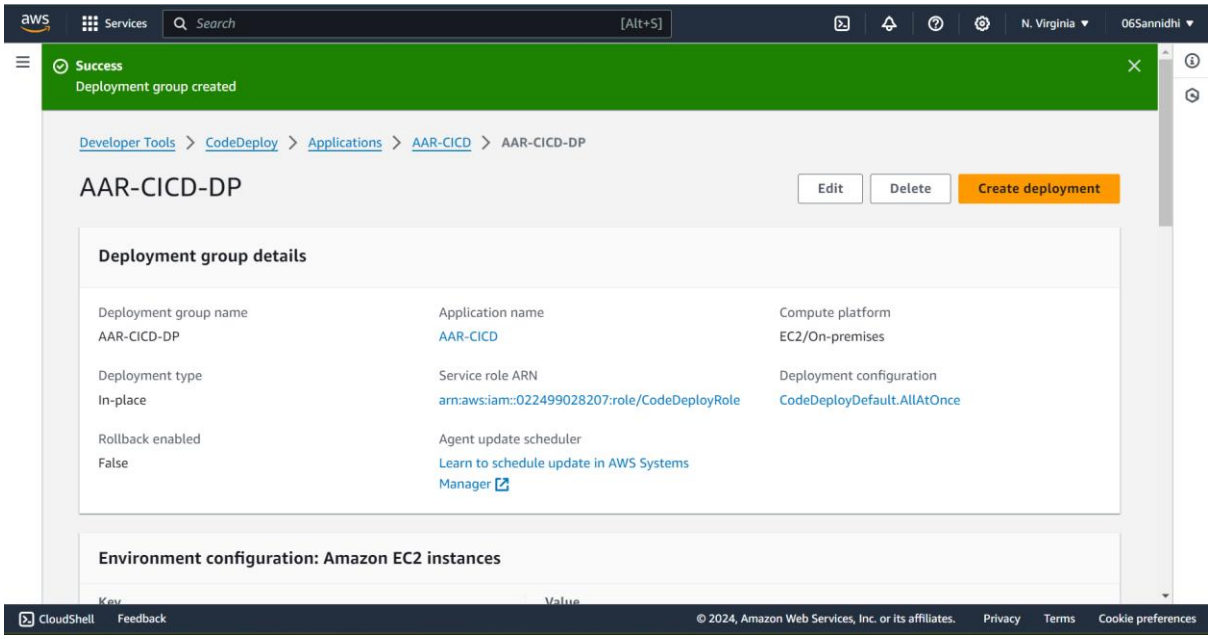

aws
Services
Global
06Sannidhi
IAM
...
Create role
Step 2 of 3
Add permissions Info
Permissions policies (1) Info
The type of role that you selected requires the following policy.
Policy name
Type
AWSCodeDeployRole
AWS managed
Set permissions boundary - optional
Cancel
Previous
Next
Feedback
Privacy
Terms
Cookie preferences
© 2024, Amazon Web Services, Inc. or its affiliates.

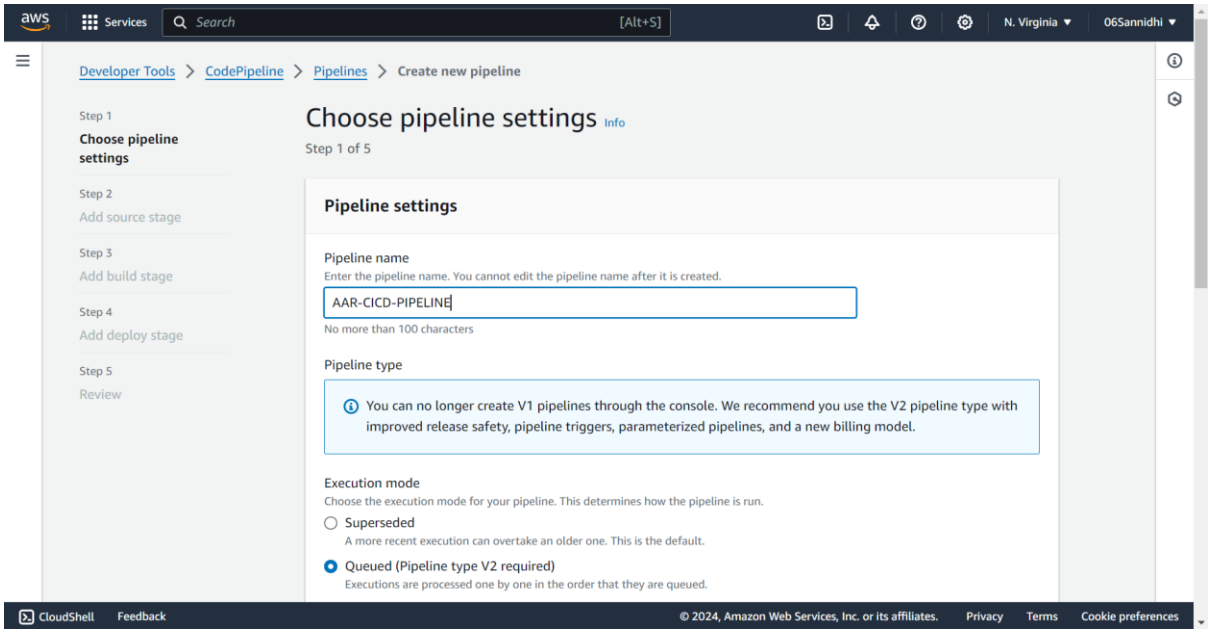

aws
Services
Global
06Sannidhi
IAM
...
Create role
Step 3 of 3
Name, review, and create
Role details
Search
[Alt+S]
Role CodeDeployRole created.
View role
Roles (5) Info
Delete
Create role
An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.
Search
Role name
Trusted entities
Last activity
AWSServiceRoleForAmazonSSM
AWS Service: ssm (Service-Linked Ro
AWSServiceRoleForSupport
AWS Service: support (Service-Linke
AWSServiceRoleForTrustedAdvisor
AWS Service: trustedadvisor (Service
CodeDeployRole
AWS Service: codedeploy
EC2CodeDeploy
AWS Service: ec2
36 days ago
Roles Anywhere Info
Manage
Authenticate your non AWS workloads and securely provide access to AWS services.
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

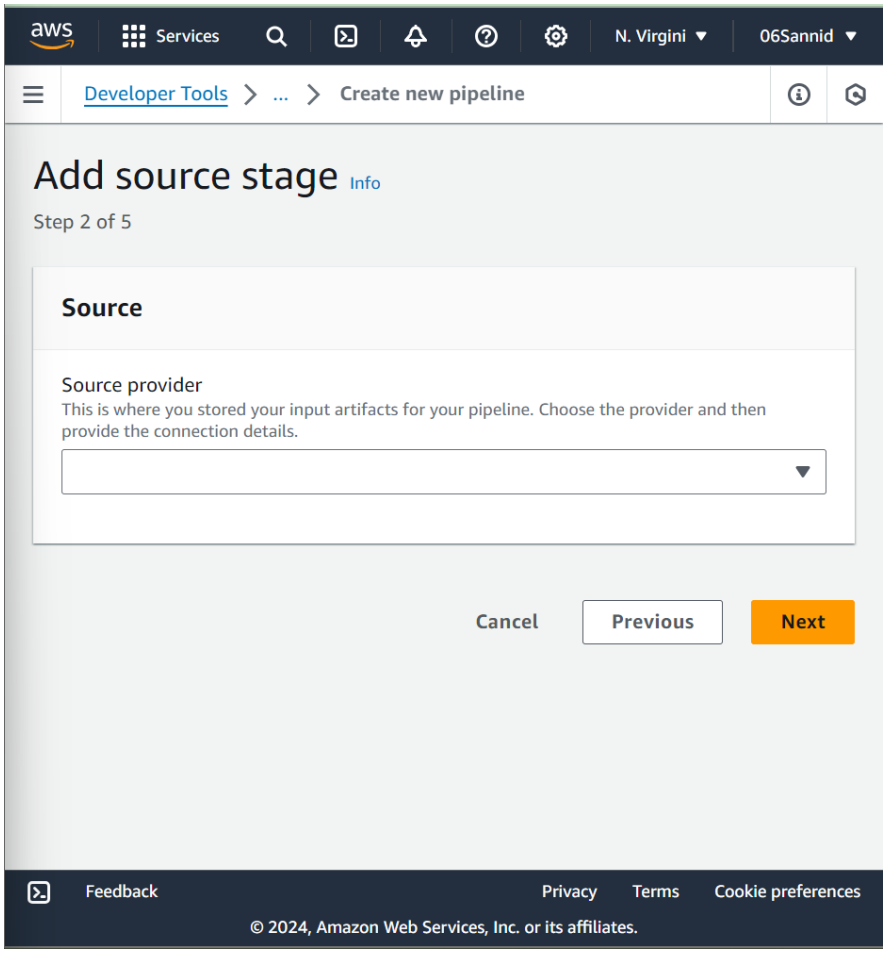

aws
Services
N. Virgini
06Sannid
EC2
...
Launch an instance
Success
Successfully initiated launch of instance (i-03cf3c5c7c35f9559)
Launch log
Next Steps
What would you like to do next with this instance, for example "create alc
1 2 3 4 5 6 7 ... 12
Create billing and free tier usage alerts
To manage costs and avoid surprise bills, set up email notifications for
Connect to your instance
Once your instance is running, log into it from your local computer.

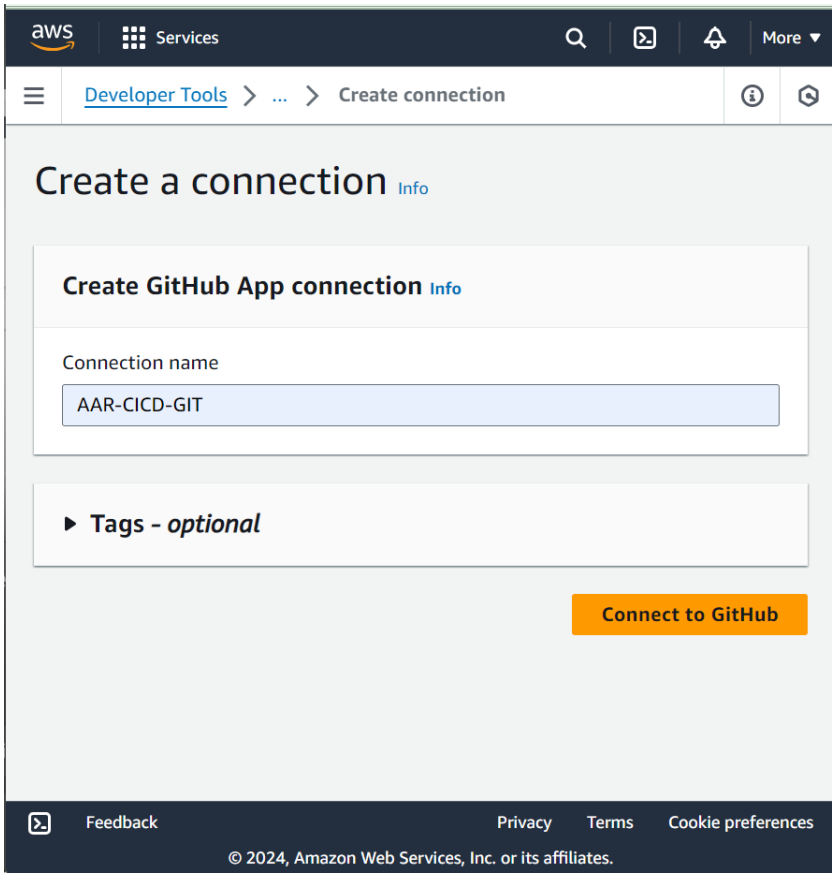

aws
Services
N. Virgini
06Sannid
Developer Tools
...
Create application
Create application
Application configuration
Application name
Enter an application name
AAR-CICD
100 character limit
Compute platform
Choose a compute platform
EC2/On-premises
Tags
Add tag
Cancel
Create application
Feedback
Privacy
Terms
Cookie preferences
© 2024, Amazon Web Services, Inc. or its affiliates.

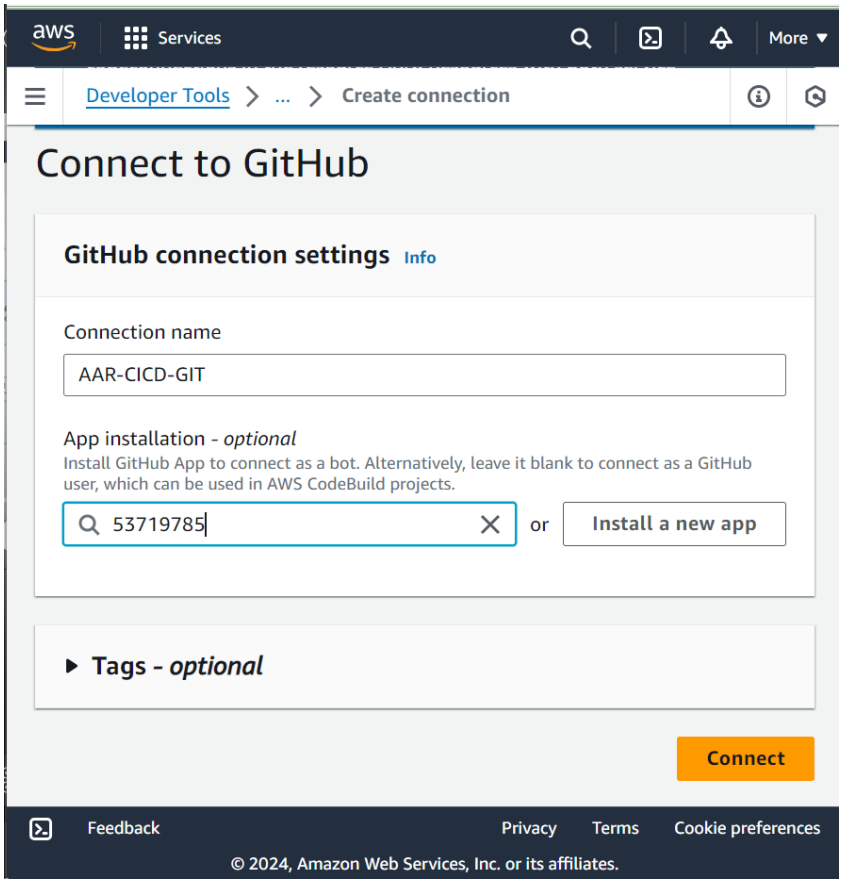

aws
Services
Search
[Alt+S]
N. Virginia
06Sannidhi
Developer Tools
CodeDeploy
Source • CodeCommit
Artifacts • CodeArtifact
Build • CodeBuild
Deploy • CodeDeploy
Getting started
Deployments
Applications
Application
Settings
Deployment configurations
On-premises instances
Pipeline • CodePipeline
Settings
Developer Tools > CodeDeploy > Applications > AAR-CICD > Create deployment group
Create deployment group
Application
Application
AAR-CICD
Compute type
EC2/On-premises
Deployment group name
Enter a deployment group name
AAR-CICD-DP
100 character limit
Service role
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

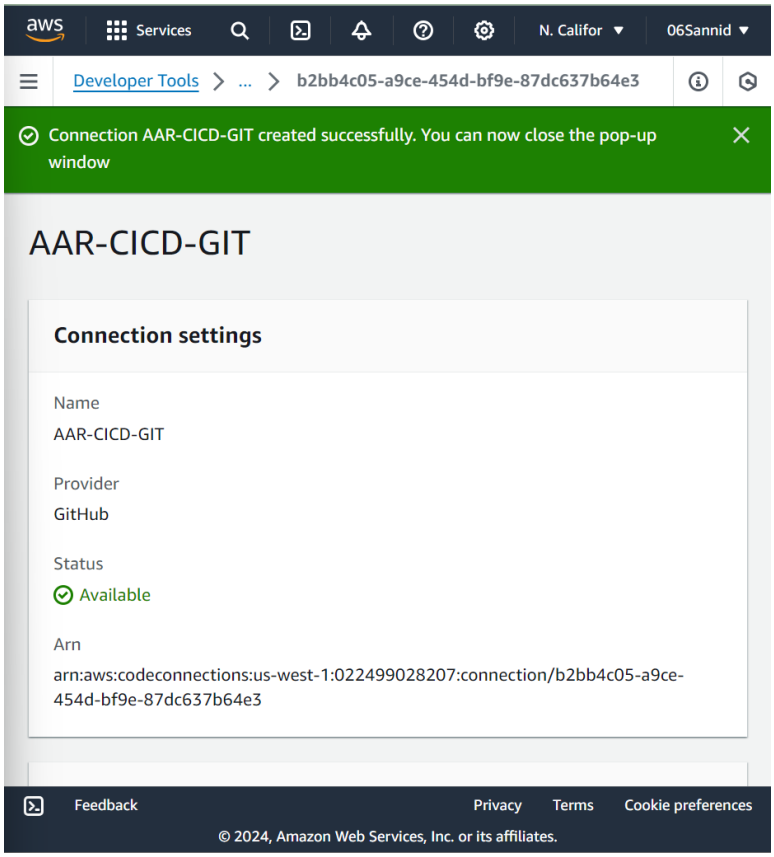

aws
Services
Search
[Alt+S]
N. Virginia
06Sannidhi
Success
Deployment group created
Developer Tools > CodeDeploy > Applications > AAR-CICD > AAR-CICD-DP
AAR-CICD-DP
Edit
Delete
Create deployment
Deployment group details
Deployment group name
AAR-CICD-DP
Application name
AAR-CICD
Compute platform
EC2/On-premises
Deployment type
In-place
Service role ARN
arn:aws:iam::022499028207:role/CodeDeployRole
Deployment configuration
CodeDeployDefault.AllAtOnce
Rollback enabled
False
Agent update scheduler
Learn to schedule update in AWS Systems Manager
Environment configuration: Amazon EC2 instances
Key
Value
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

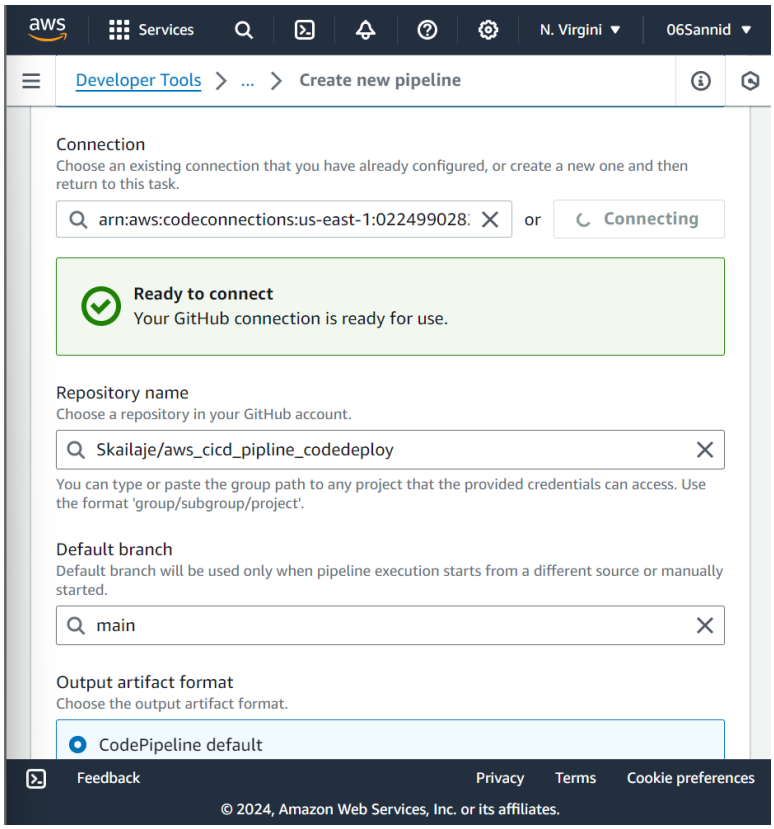

aws
Services
Search
[Alt+S]
N. Virginia
06Sannidhi
Developer Tools > CodePipeline > Pipelines > Create new pipeline
Step 1
Choose pipeline settings
Step 2
Add source stage
Step 3
Add build stage
Step 4
Add deploy stage
Step 5
Review
Choose pipeline settings Info
Step 1 of 5
Pipeline settings
Pipeline name
Enter the pipeline name. You cannot edit the pipeline name after it is created.
AAR-CICD-PIPELINE
No more than 100 characters
Pipeline type
You can no longer create V1 pipelines through the console. We recommend you use the V2 pipeline type with improved release safety, pipeline triggers, parameterized pipelines, and a new billing model.
Execution mode
Choose the execution mode for your pipeline. This determines how the pipeline is run.
Superseded
A more recent execution can overtake an older one. This is the default.
Queued (Pipeline type V2 required)
Executions are processed one by one in the order that they are queued.
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

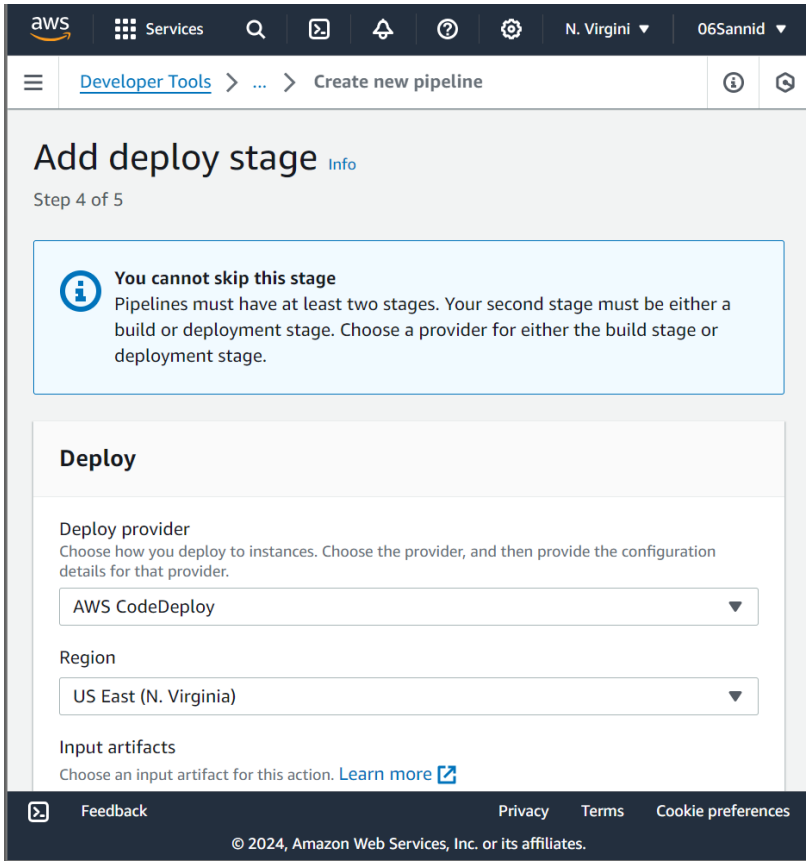

aws
Services
N. Virgini
06Sannid
Developer Tools > ... > Create new pipeline
Add source stage Info
Step 2 of 5
Source
Source provider
This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.
Cancel
Previous
Next
Feedback
Privacy
Terms
Cookie preferences
© 2024, Amazon Web Services, Inc. or its affiliates.

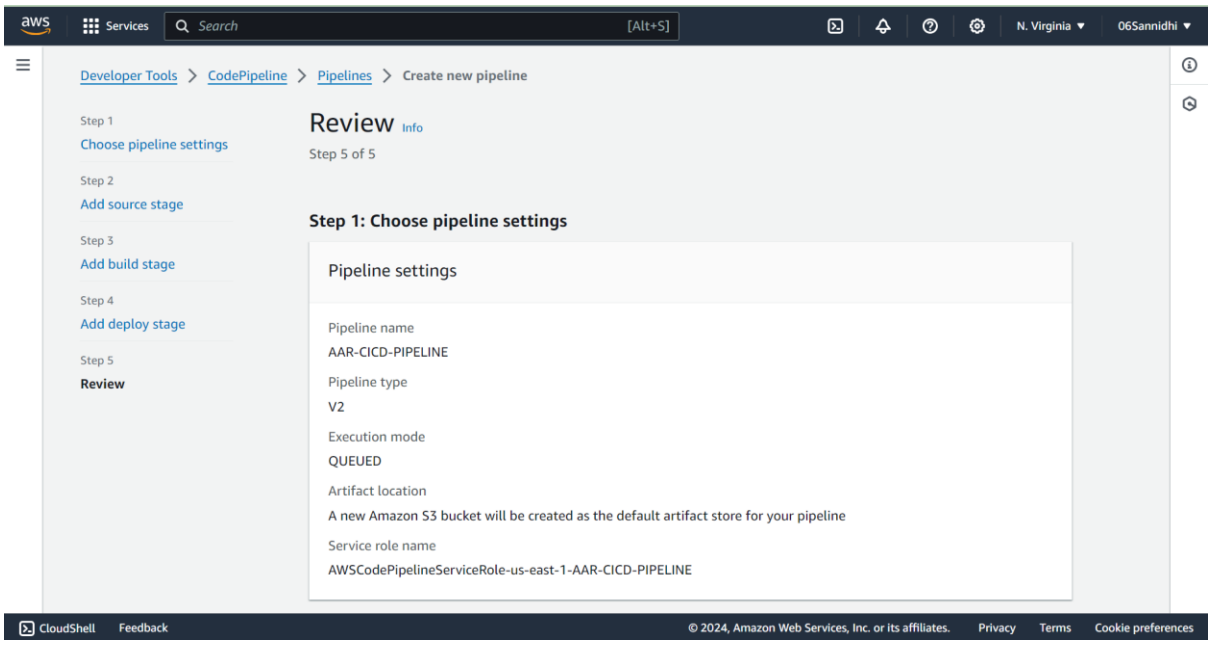

aws
Services
More
Developer Tools
...
Create connection
Create a connection Info
Create GitHub App connection Info
Connection name
AAR-CICD-GIT
Tags - optional
Connect to GitHub
Feedback
Privacy
Terms
Cookie preferences
© 2024, Amazon Web Services, Inc. or its affiliates.

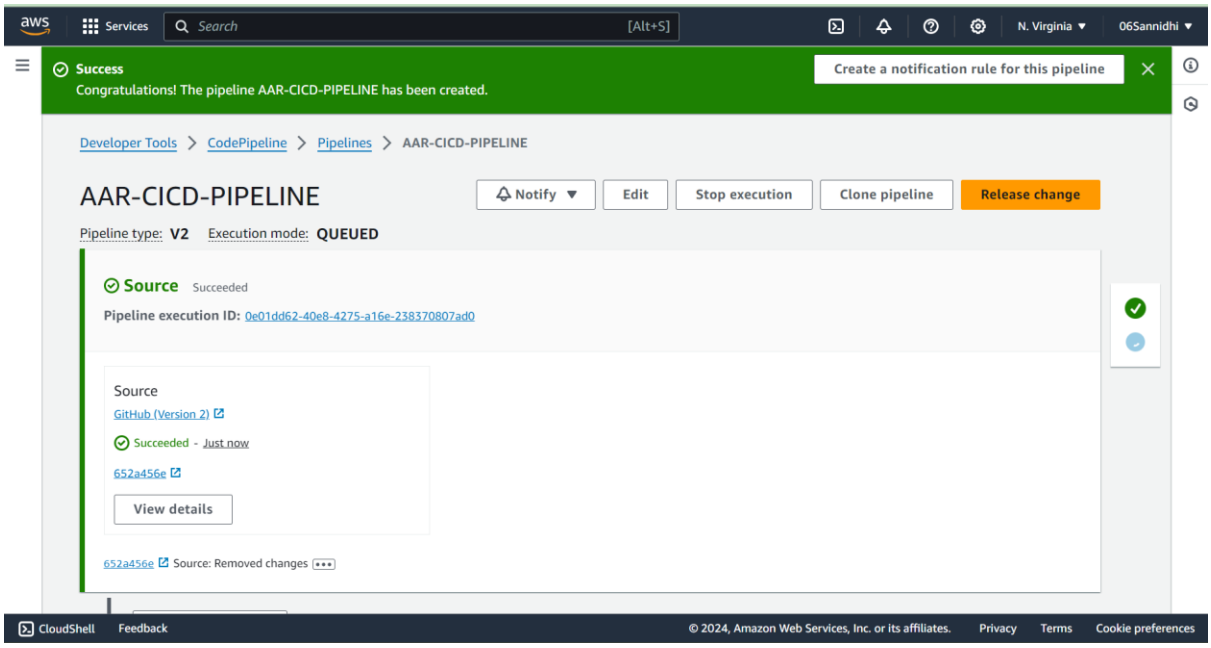

aws
Services
More
Developer Tools
...
Create connection
Connect to GitHub
GitHub connection settings Info
Connection name
AAR-CICD-GIT
App installation - optional
Install GitHub App to connect as a bot. Alternatively, leave it blank to connect as a GitHub user, which can be used in AWS CodeBuild projects.
53719785
or
Install a new app
Tags - optional
Connect
Feedback
Privacy
Terms
Cookie preferences
© 2024, Amazon Web Services, Inc. or its affiliates.

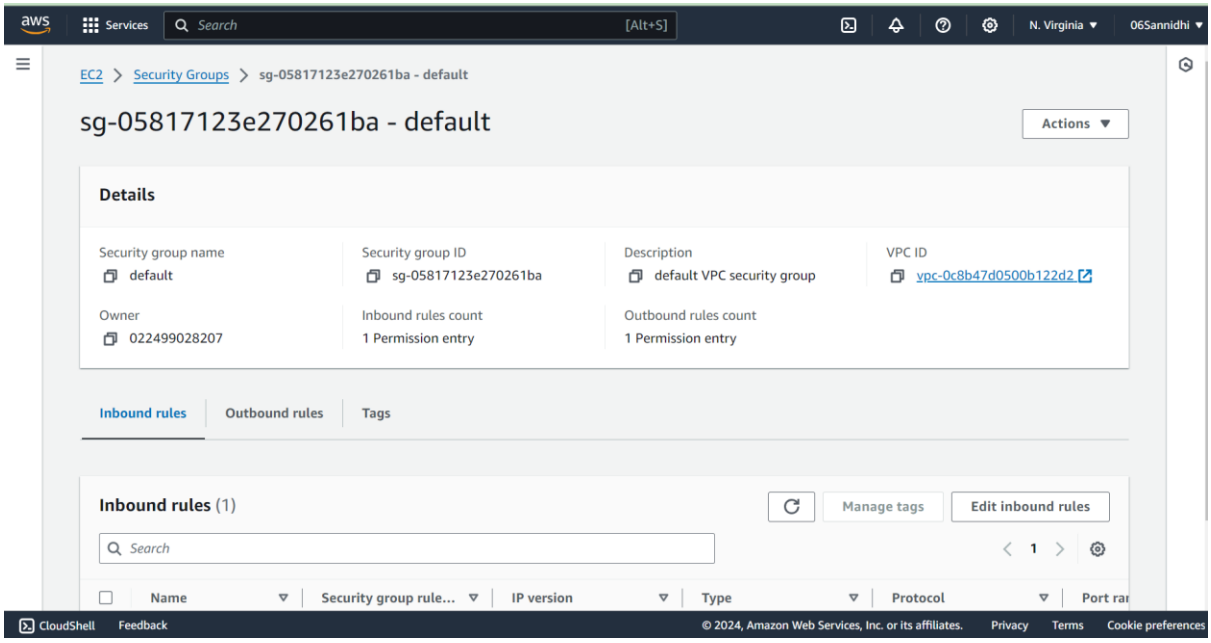

aws
Services
N. Califor
06Sannid
Developer Tools
...
b2bb4c05-a9ce-454d-bf9e-87dc637b64e3
Connection AAR-CICD-GIT created successfully. You can now close the pop-up window
AAR-CICD-GIT
Connection settings
Name
AAR-CICD-GIT
Provider
GitHub
Status
Available
Arn
arn:aws:codeconnections:us-west-1:022499028207:connection/b2bb4c05-a9ce-454d-bf9e-87dc637b64e3
Feedback
Privacy
Terms
Cookie preferences
© 2024, Amazon Web Services, Inc. or its affiliates.

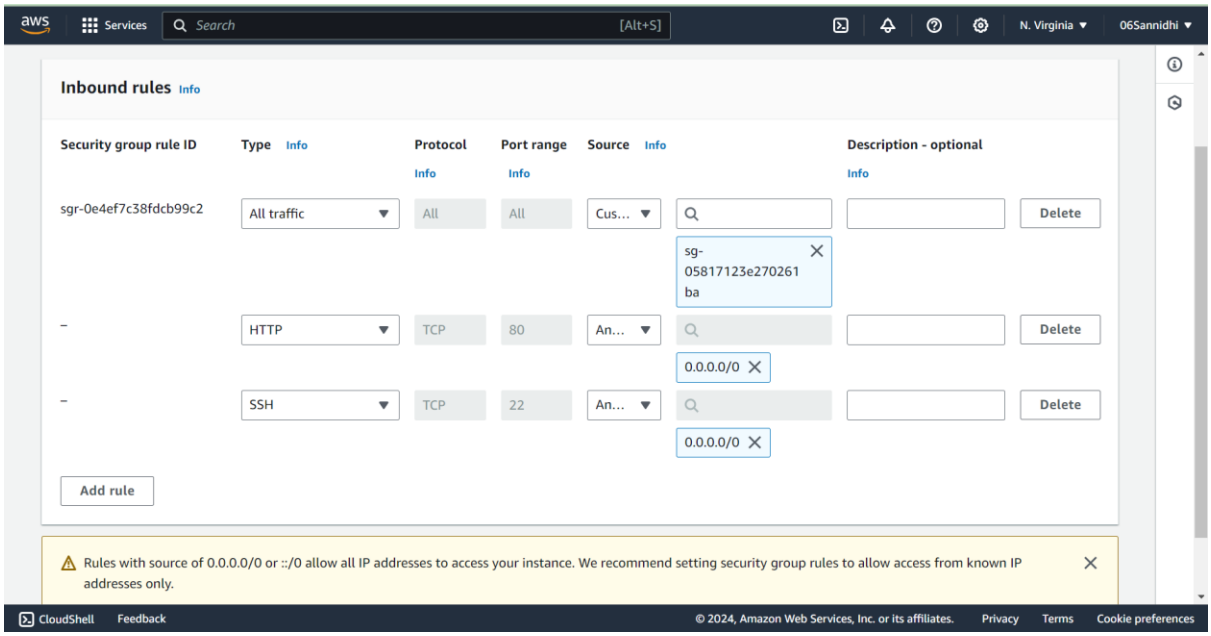

aws
Services
N. Virgini
06Sannid
Developer Tools
...
Create new pipeline
Connection
Choose an existing connection that you have already configured, or create a new one and then return to this task.
arn:aws:codeconnections:us-east-1:022499028:
or
Connecting
Ready to connect
Your GitHub connection is ready for use.
Repository name
Choose a repository in your GitHub account.
Skailaje/aws_cicd_pipline_codedeploy
You can type or paste the group path to any project that the provided credentials can access. Use the format 'group/subgroup/project'.
Default branch
Default branch will be used only when pipeline execution starts from a different source or manually started.
main
Output artifact format
Choose the output artifact format.
CodePipeline default
Feedback
Privacy
Terms
Cookie preferences
© 2024, Amazon Web Services, Inc. or its affiliates.

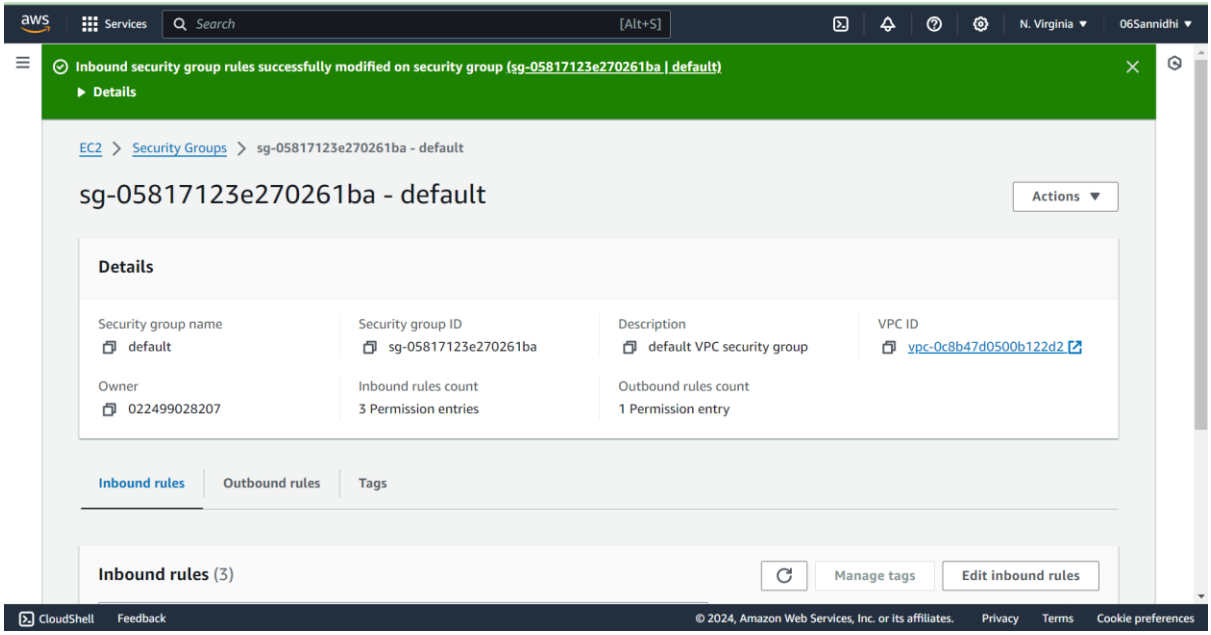

aws
Services
N. Virgini
06Sannid
Developer Tools
...
Create new pipeline
Add deploy stage
Info
Step 4 of 5
You cannot skip this stage
Pipelines must have at least two stages. Your second stage must be either a build or deployment stage. Choose a provider for either the build stage or deployment stage.
Deploy
Deploy provider
Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.
AWS CodeDeploy
Region
US East (N. Virginia)
Input artifacts
Choose an input artifact for this action. Learn more
Feedback
Privacy
Terms
Cookie preferences
© 2024, Amazon Web Services, Inc. or its affiliates.

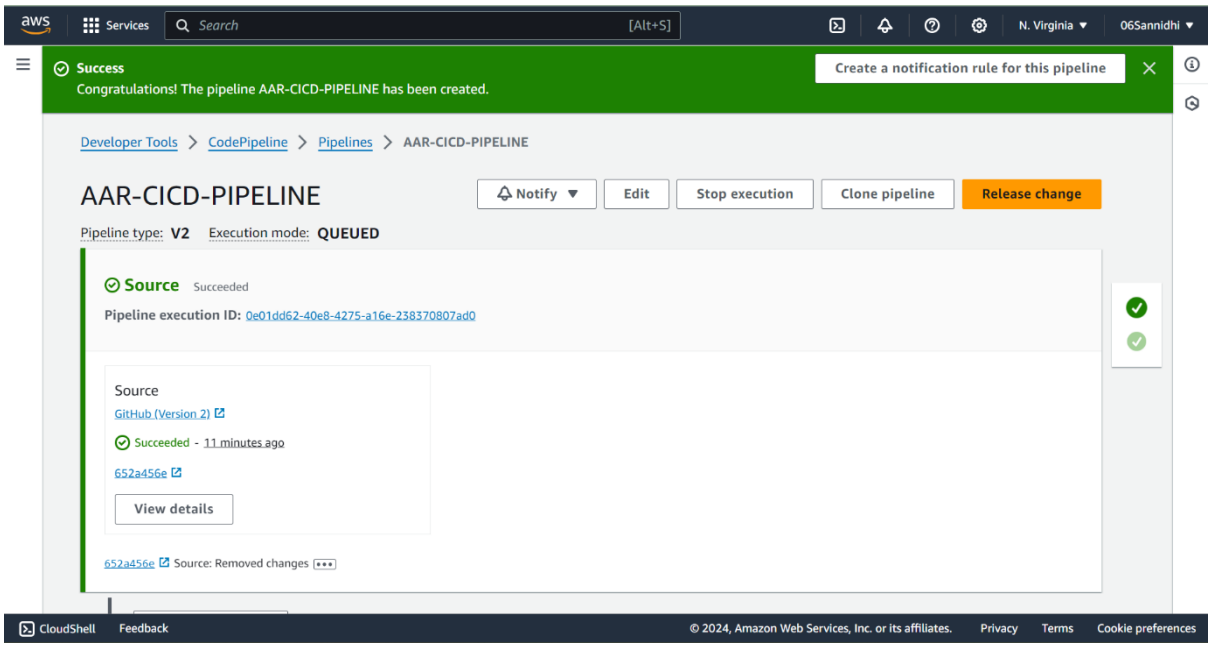

aws
Services
Search
[Alt+S]
N. Virginia
06Sannidhi
Developer Tools
CodePipeline
Pipelines
Create new pipeline
Step 1
Choose pipeline settings
Step 2
Add source stage
Step 3
Add build stage
Step 4
Add deploy stage
Step 5
Review
Review
Info
Step 5 of 5
Step 1: Choose pipeline settings
Pipeline settings
Pipeline name
AAR-CICD-PIPELINE
Pipeline type
V2
Execution mode
QUEUED
Artifact location
A new Amazon S3 bucket will be created as the default artifact store for your pipeline
Service role name
AWSCodePipelineServiceRole-us-east-1-AAR-CICD-PIPELINE
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


aws
Services
Search
[Alt+S]
N. Virginia
06Sannidhi
Success
Congratulations! The pipeline AAR-CICD-PIPELINE has been created.
Create a notification rule for this pipeline
Developer Tools > CodePipeline > Pipelines > AAR-CICD-PIPELINE
AAR-CICD-PIPELINE
Notify
Edit
Stop execution
Clone pipeline
Release change
Pipeline type: V2
Execution mode: QUEUED
Source
Succeeded
Pipeline execution ID: 0e01dd62-40e8-4275-a16e-238370807ad0
Source
GitHub (Version 2)
Succeeded - Just now
652a456e
View details
652a456e Source: Removed changes
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


aws
Services
Search
[Alt+S]
N. Virginia
06Sannidhi
EC2 > Security Groups > sg-05817123e270261ba - default
sg-05817123e270261ba - default
Actions
Details
Security group name
default
Security group ID
sg-05817123e270261ba
Description
default VPC security group
VPC ID
vpc-0c8b47d0500b122d2
Owner
022499028207
Inbound rules count
1 Permission entry
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Tags
Inbound rules (1)
Manage tags
Edit inbound rules
Search
1
Name
Security group rule...
IP version
Type
Protocol
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


aws
Services
Search
[Alt+S]
N. Virginia
06Sannidhi
Inbound rules Info
Security group rule ID
Type Info
Protocol Info
Port range Info
Source Info
Description - optional Info
sgr-0e4ef7c38fdcb99c2
All traffic
All
All
Cus...
sg-05817123e270261ba
Delete
HTTP
TCP
80
An...
0.0.0.0/0
Delete
SSH
TCP
22
An...
0.0.0.0/0
Delete
Add rule
Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


aws
Services
Search
[Alt+S]
N. Virginia
06Sannidhi
Inbound security group rules successfully modified on security group (sg-05817123e270261ba | default)
Details
EC2 > Security Groups > sg-05817123e270261ba - default
sg-05817123e270261ba - default
Actions
Details
Security group name
default
Security group ID
sg-05817123e270261ba
Description
default VPC security group
VPC ID
vpc-0c8b47d0500b122d2
Owner
022499028207
Inbound rules count
3 Permission entries
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Tags
Inbound rules (3)
Manage tags
Edit inbound rules
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences



Conclusion:

Successfully Build Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.

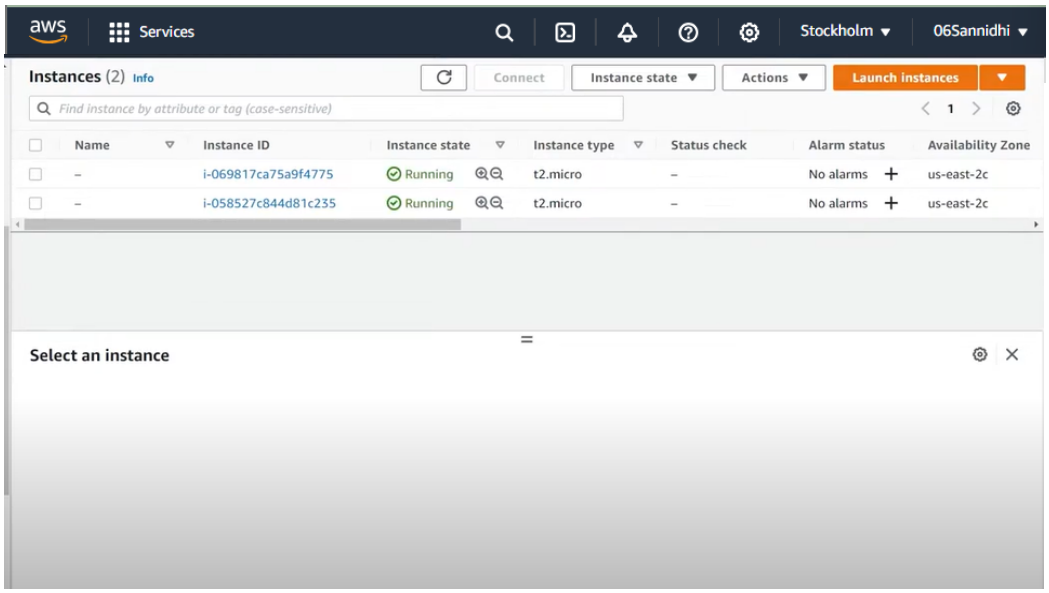
EXPERIMENT 3

Aim

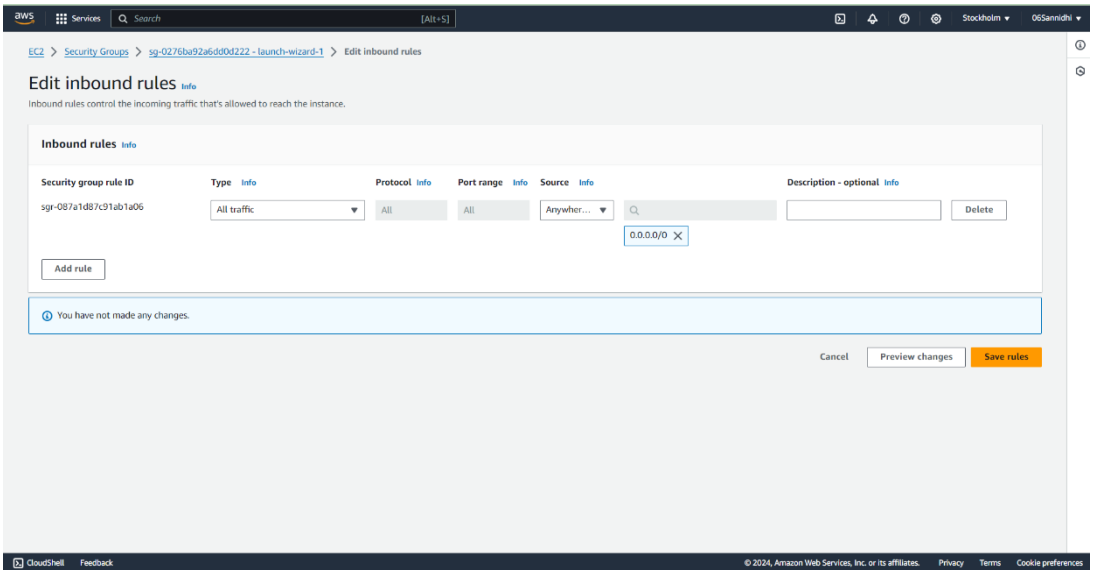
To understand the Kubernetes Cluster Architecture, install and Spin Up a Kubernetes Cluster on Linux Machines/Cloud Platforms.

Implementation

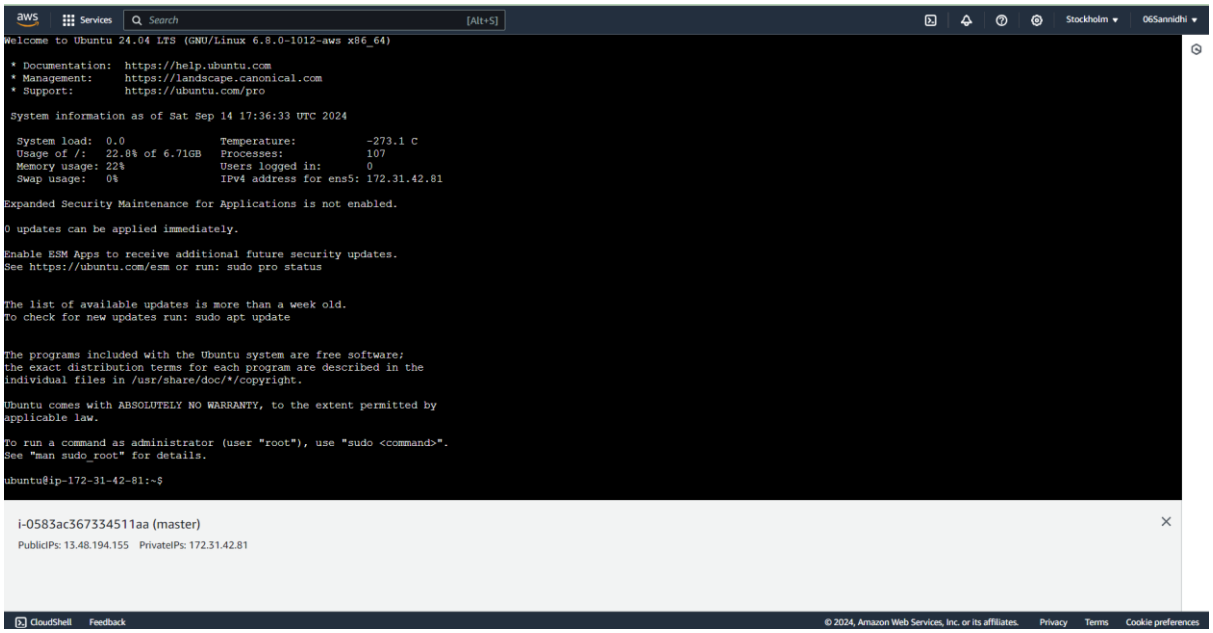
1) Create EC2 Ubuntu Instances on AWS. (Master and Worker)

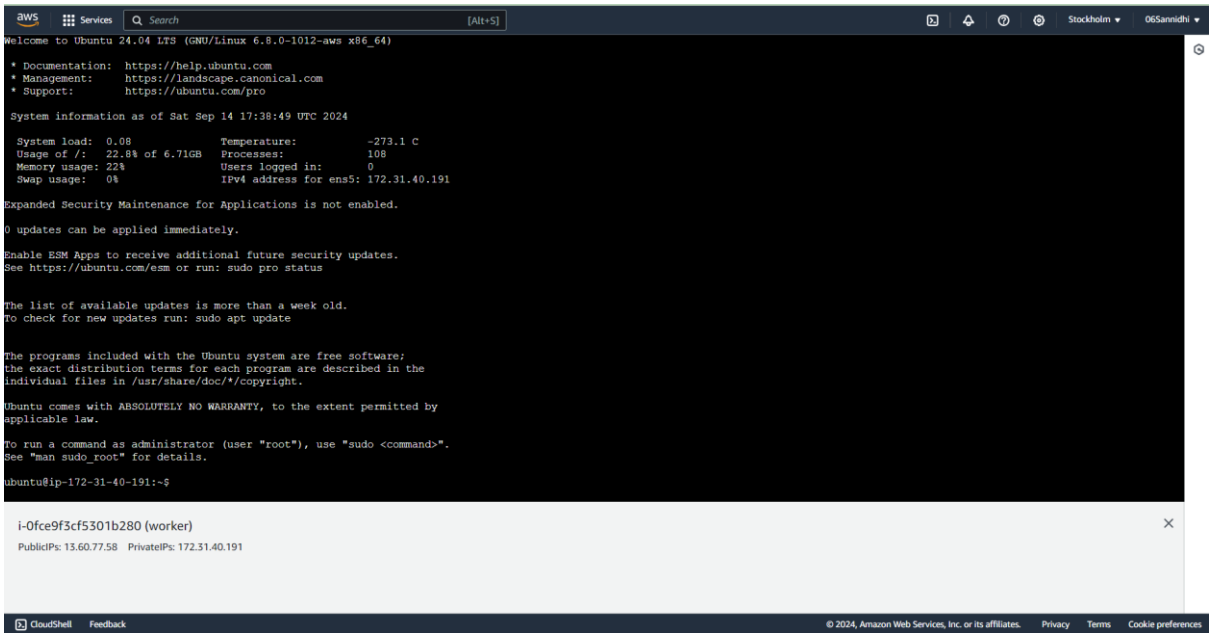


2) Edit the Security Group Inbound Rules to allow SSH



3) AWS CLI for master and worker instances





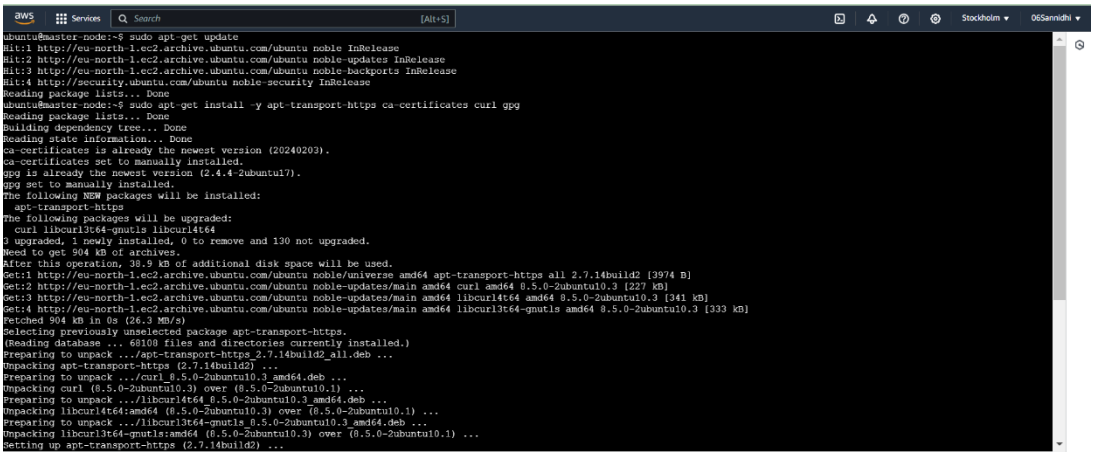
4) Assign Unique Hostname for Each Server Node

```
$ sudo hostnamectl set-hostname master
$ sudo hostnamectl set-hostname worker-1
```

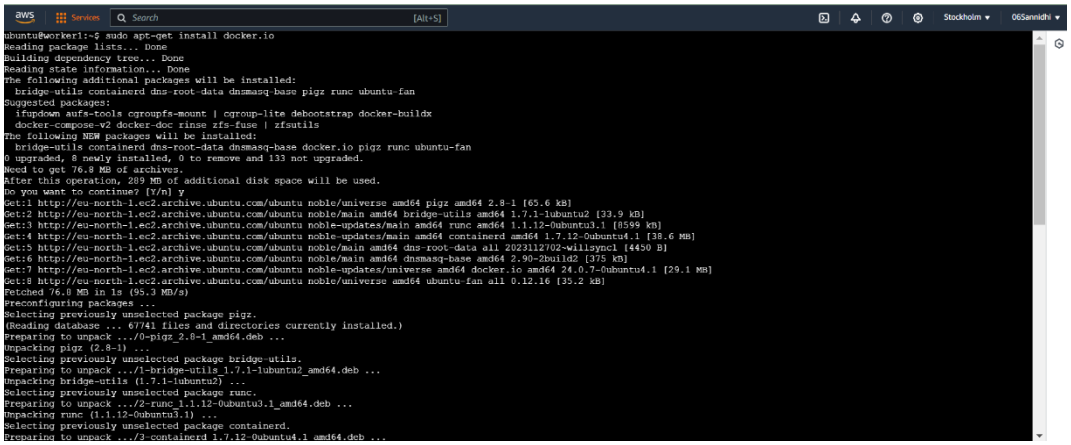
Set up Docker (both master and worker)

5) Install Docker

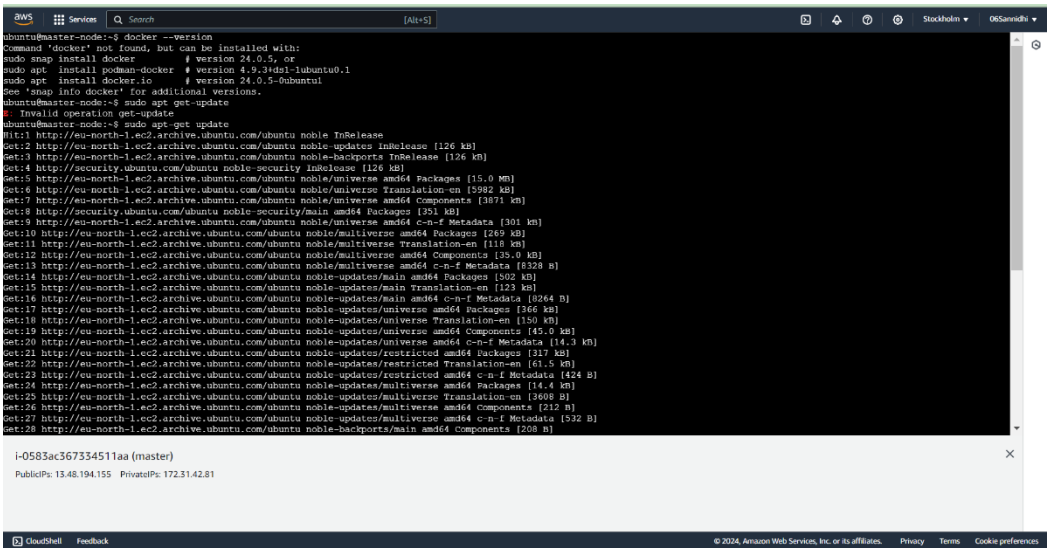
$ sudo apt-get update



$ sudo apt-get install docker.io



$ docker —version

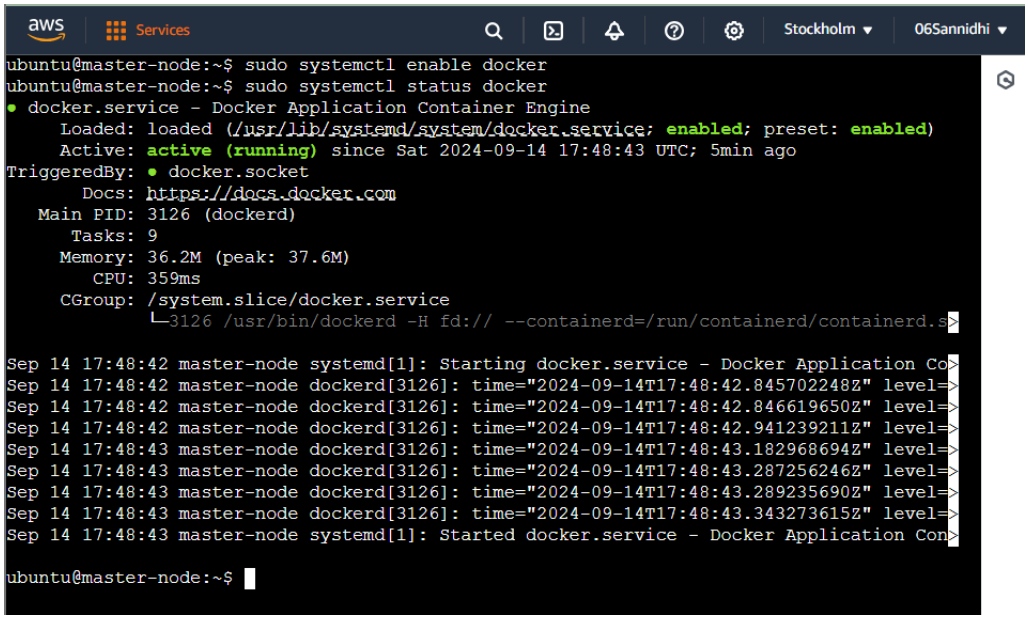


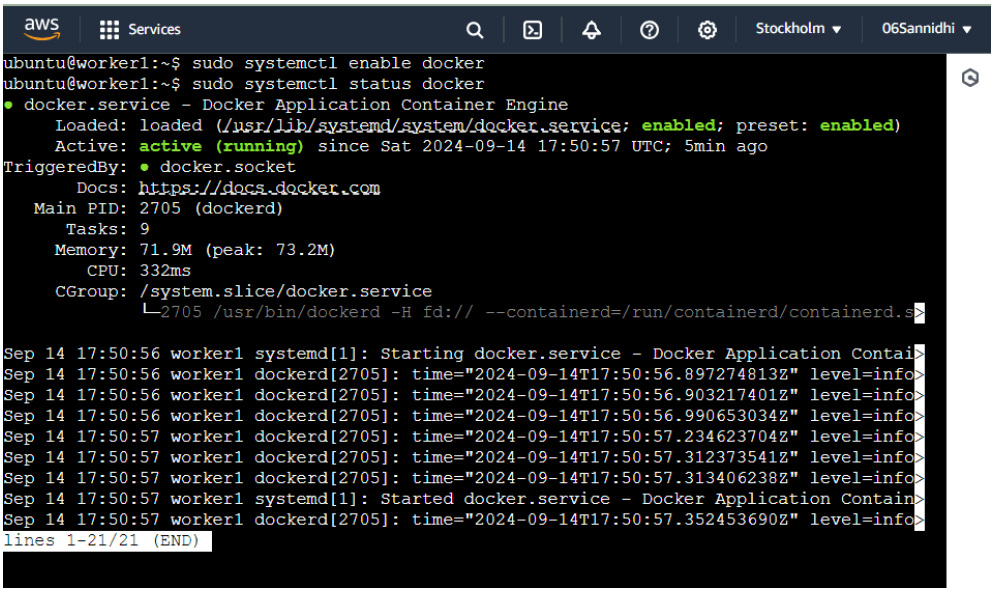
6) Start and Enable Docker

$ sudo systemctl enable docker

$ sudo systemctl status docker

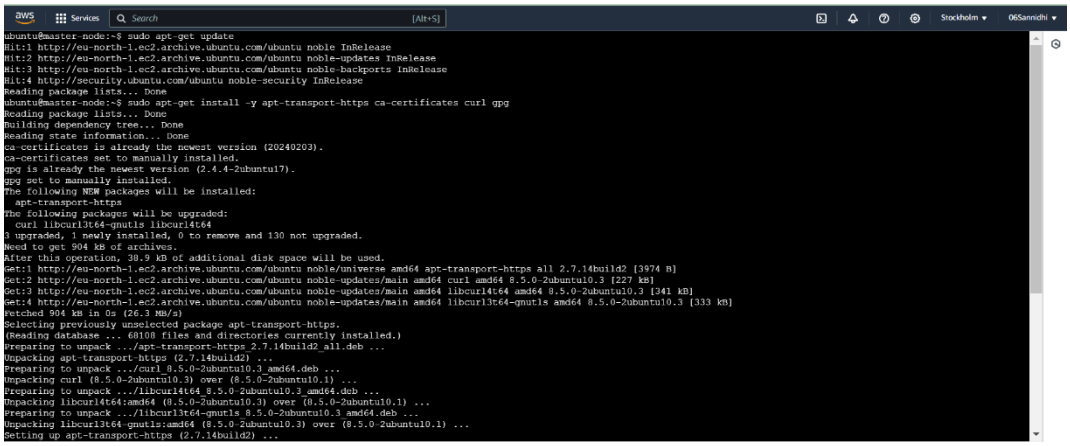
$ sudo systemctl start docker





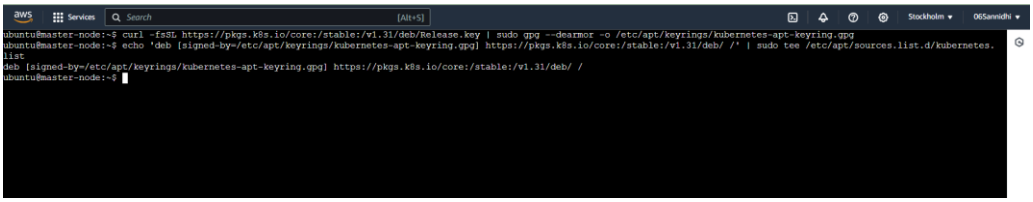
7) Install Kubernetes(both master and worker node)

$ sudo apt-get update
$ sudo apt-get install -y apt-transport-https ca-certificates curl



$ sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg

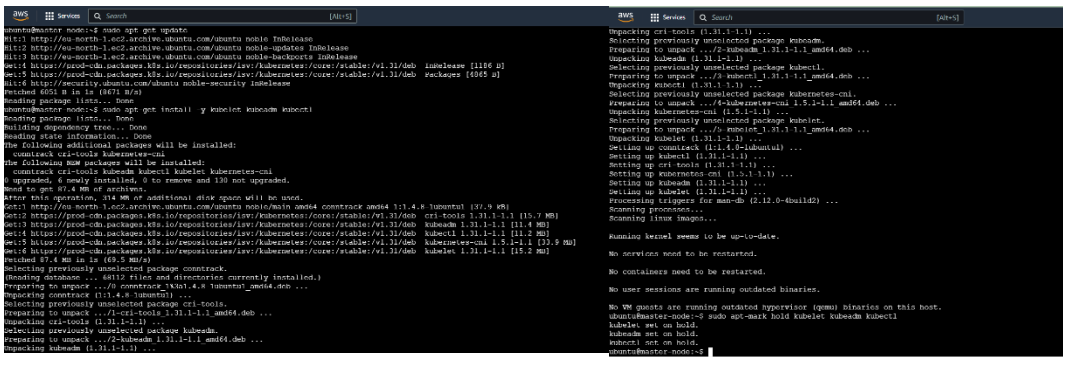
$ echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list

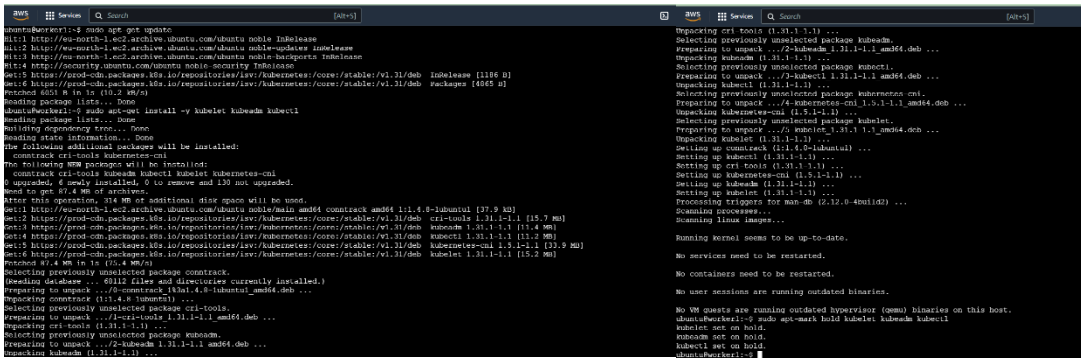


$ sudo apt-get update

$ sudo apt-get install -y kubelet kubeadm kubectl

$ sudo apt-mark hold kubelet kubeadm kubectl

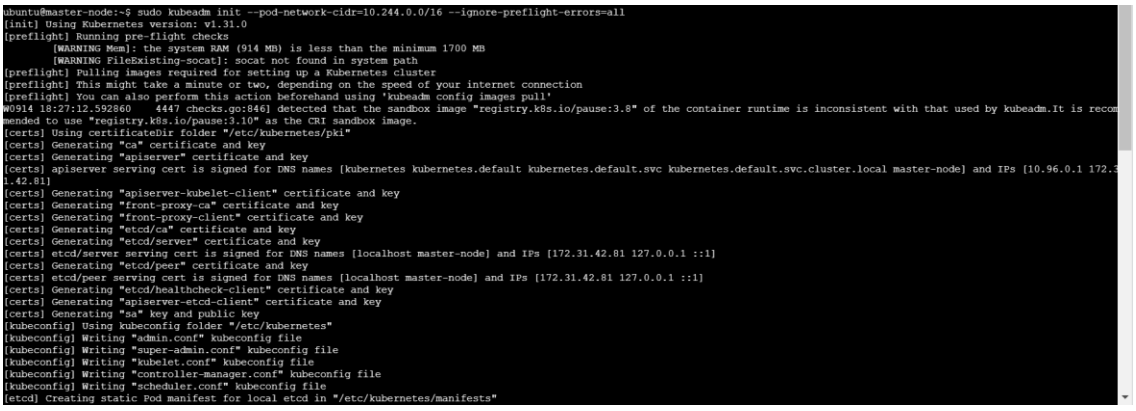




Kubernetes Deployment (master only)

8) Begin Kubernetes Deployment
   $ sudo swapoff –a

9) Initialize Kubernetes on Master Node
   $ sudo kubeadm init --pod-network-cidr=10.244.0.0/16--ignore-preflight-errors=all



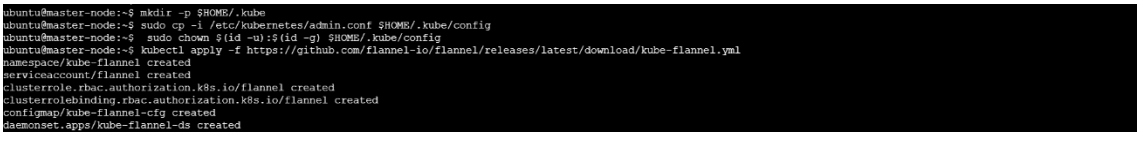
10) Deploy Pod Network to Cluster

    $ mkdir -p $HOME/.kube

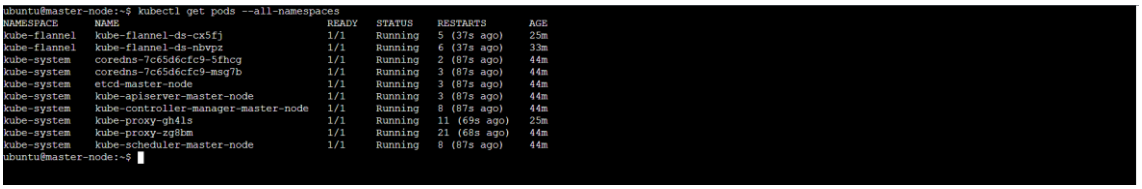
    $ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

    $ sudo chown $(id -u):$(id -g) $HOME/.kube/config

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

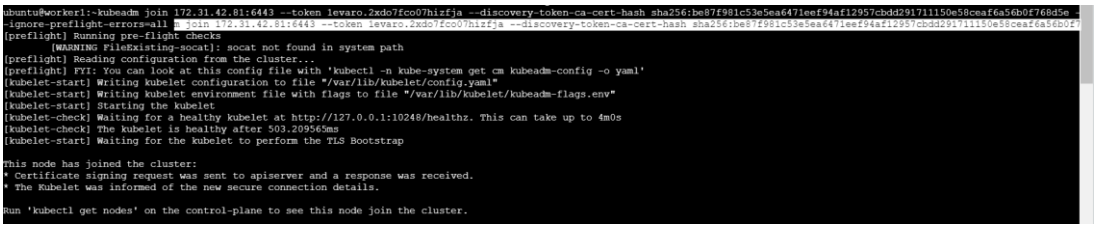


    $ kubectl get pods --all-namespaces

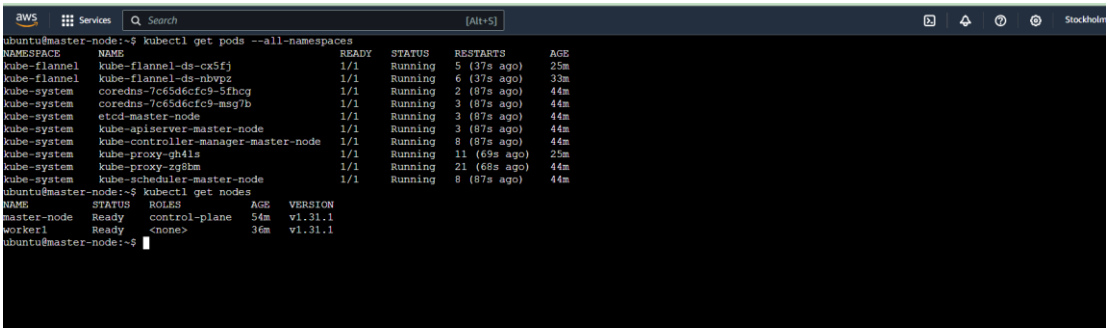


Join Worker Node to Cluster (on worker node)

11) $ kubeadm join 172.31.42.81:6443 --token 1evaro.2xdo7fco07hizfja --discovery-token-ca-cert-hash sha256:be87f981c53e5ea6471eef94af12957cbdd291711150e58ceaf6a56b0f768d5e --ignore-preflight-errors=all



12) $ kubectl get nodes (on master node)



we now have a Kubernetes cluster running across AWS EC2 Instances. This cluster can be used to further deploy applications and their loads being distributed across these machines.

Conclusion

Successfully understood the Kubernetes cluster architecture and deployed a Kubernetes cluster on Linux machines/cloud platforms, demonstrating seamless setup and orchestration.

EXPERIMENT 4

Aim

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

Implementation

1) In your Kubernetes cluster check if all nodes are connected.
   $ kubectl get nodes

```
ubuntu@master-node:~$ kubectl get nodes
NAME          STATUS   ROLES           AGE   VERSION
master-node   Ready    control-plane   54m   v1.31.1
worker1       Ready    <none>          36m   v1.31.1
ubuntu@master-node:~$
```

2) Create deploy.yaml file
   $ sudo nano deploy.yaml

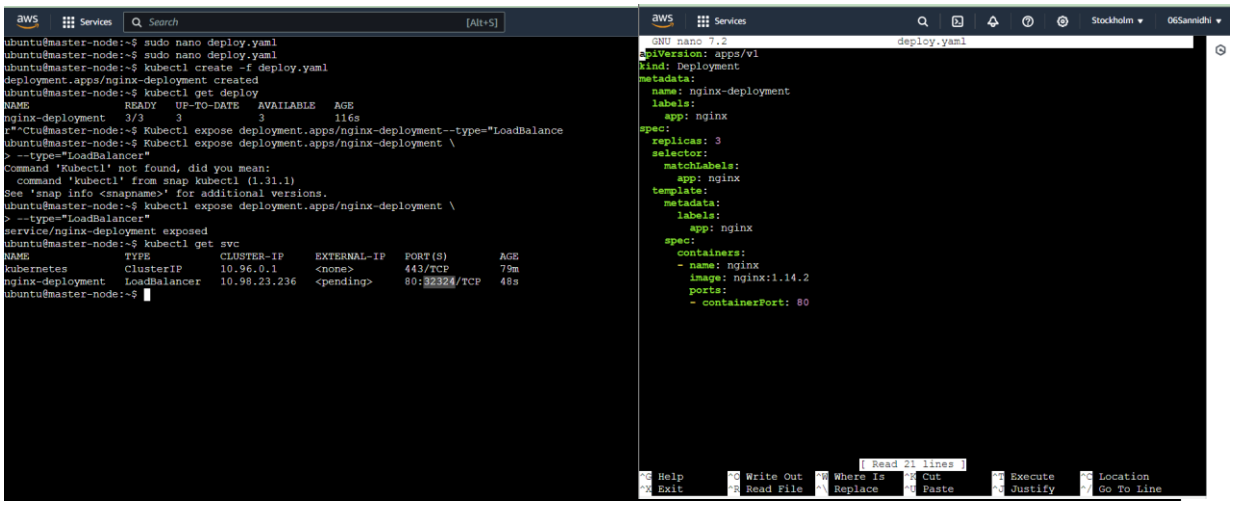
3) Copy nginx-deployement.yaml file from Kubernetes site and paste in above file.

4) Create deployment
   $ kubectl apply -f

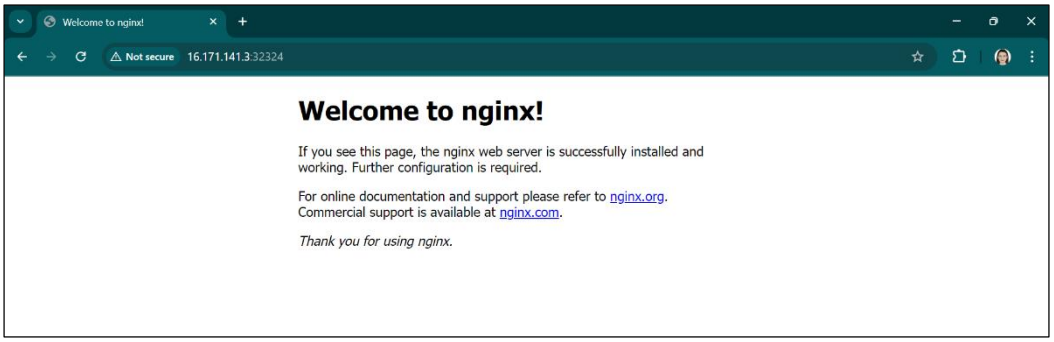
5) Check if deployment is created and all pods are running
   $ kubectl get deploy.po

6) Expose the app to internet using load balancer service.
   $ kubectl expose deployment.apps/nginx-deployment –type= ”load balancer”

7) Kubectl get svc



8) deployed our Nginx server on our EC2 instance.



Conclusion

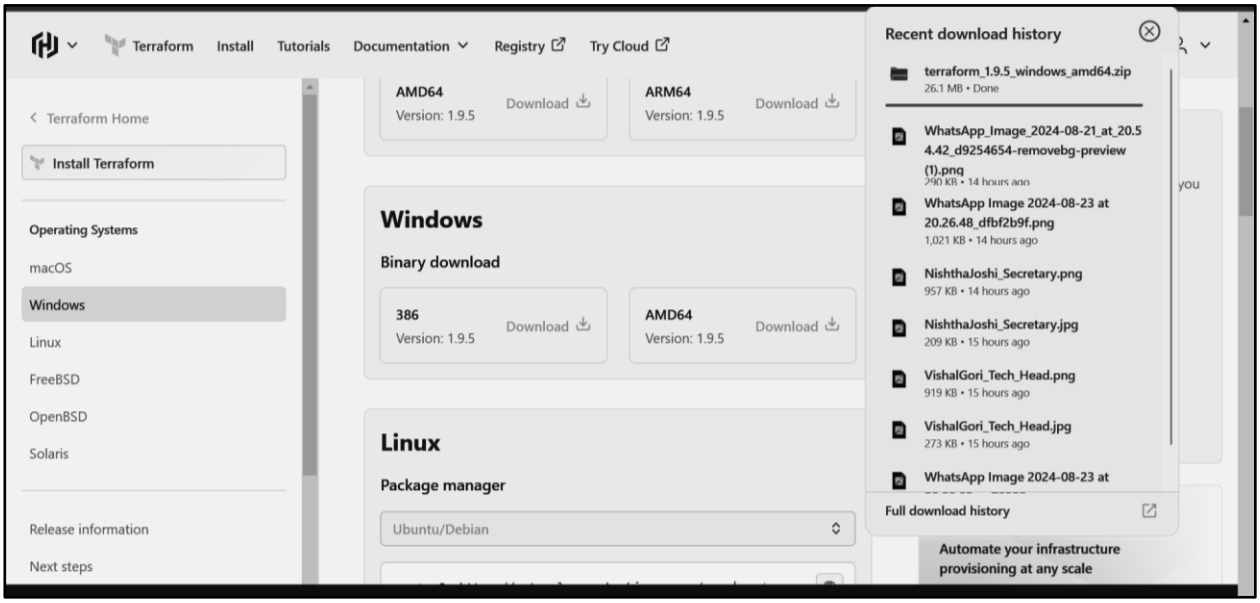
Successfully installed Kubectl and executed commands to manage the Kubernetes cluster, deploying the first application seamlessly.

EXPERIMENT 5

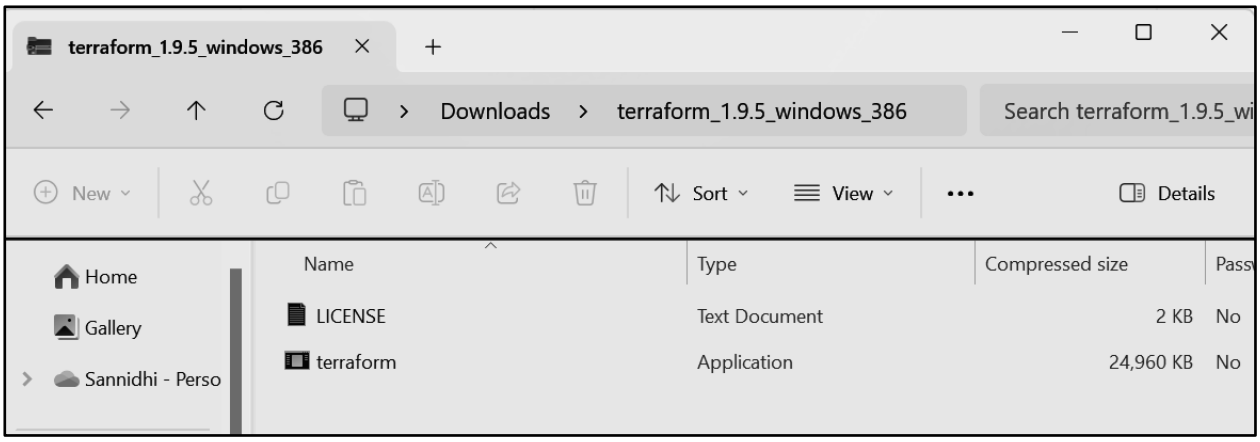
Aim:
To understand terraform lifecycle,core concepts /terminologies and install it on a Linux Machine and Windows

Implementation:

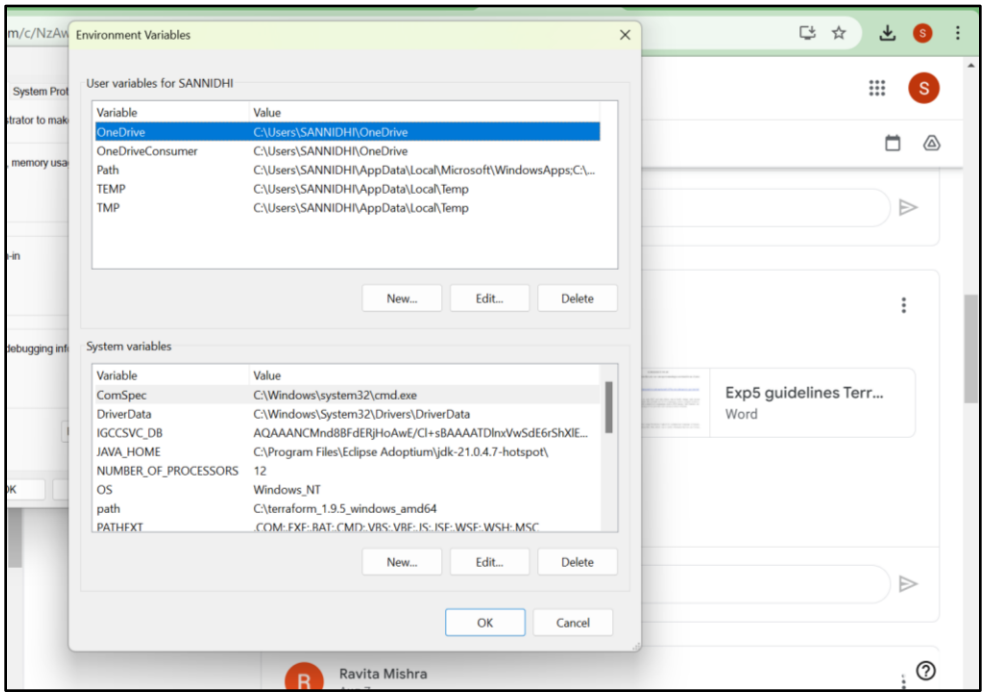
Installing Terraform

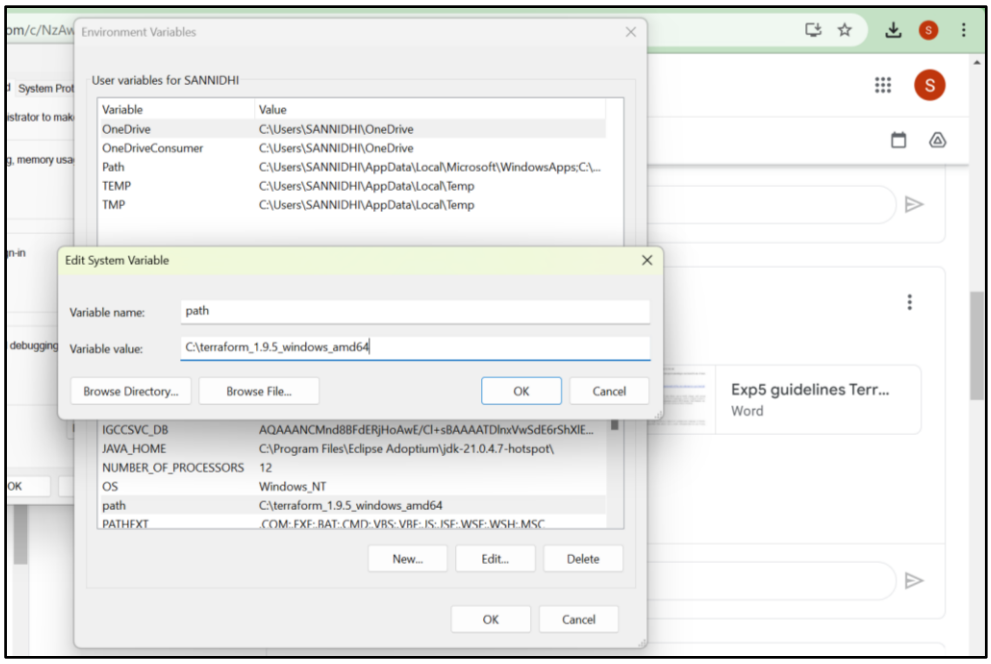


Extract the downloaded setup file Terraform.exe in C:\Terraform directory

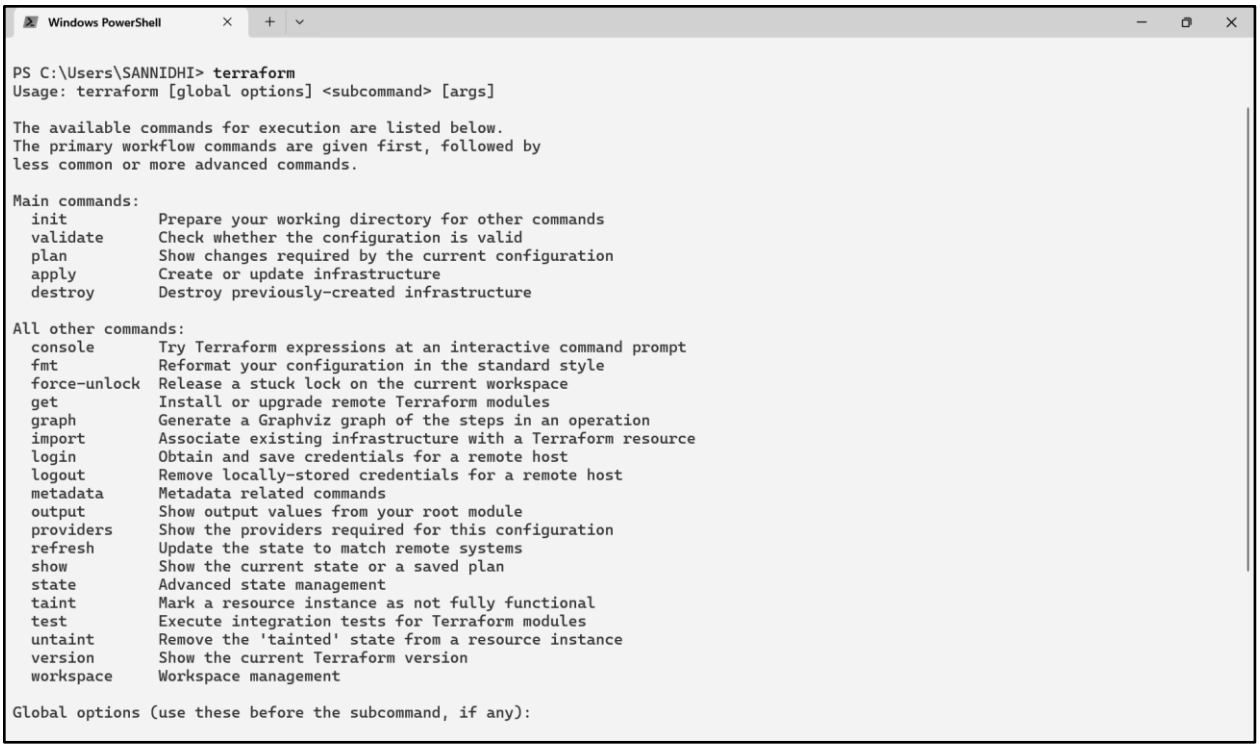


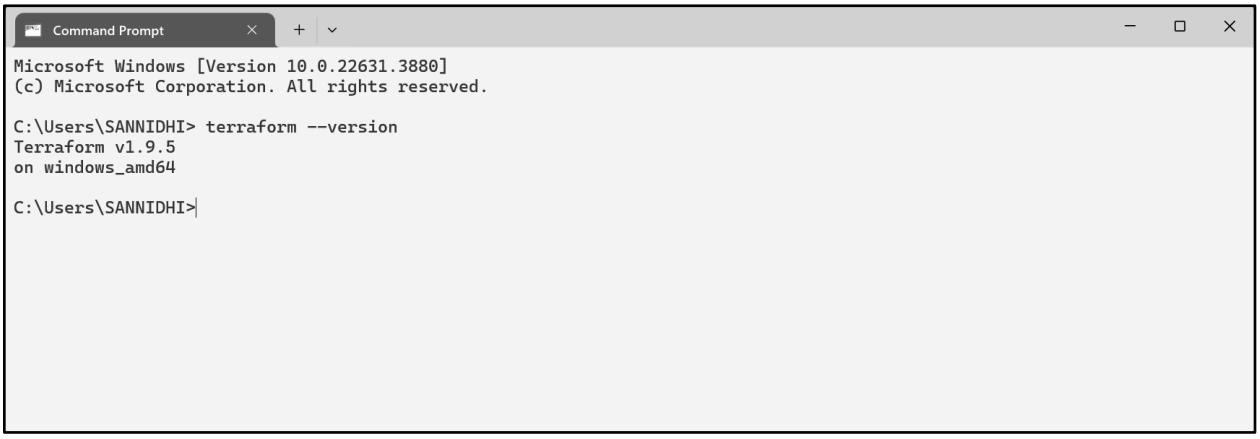
Set the System path for Terraform in Environment Variables





Terraform in PowerShell and its functionality





Conclusion:

Successfully understood the Terraform lifecycle, core concepts, and terminologies, and completed the installation on Windows machines.

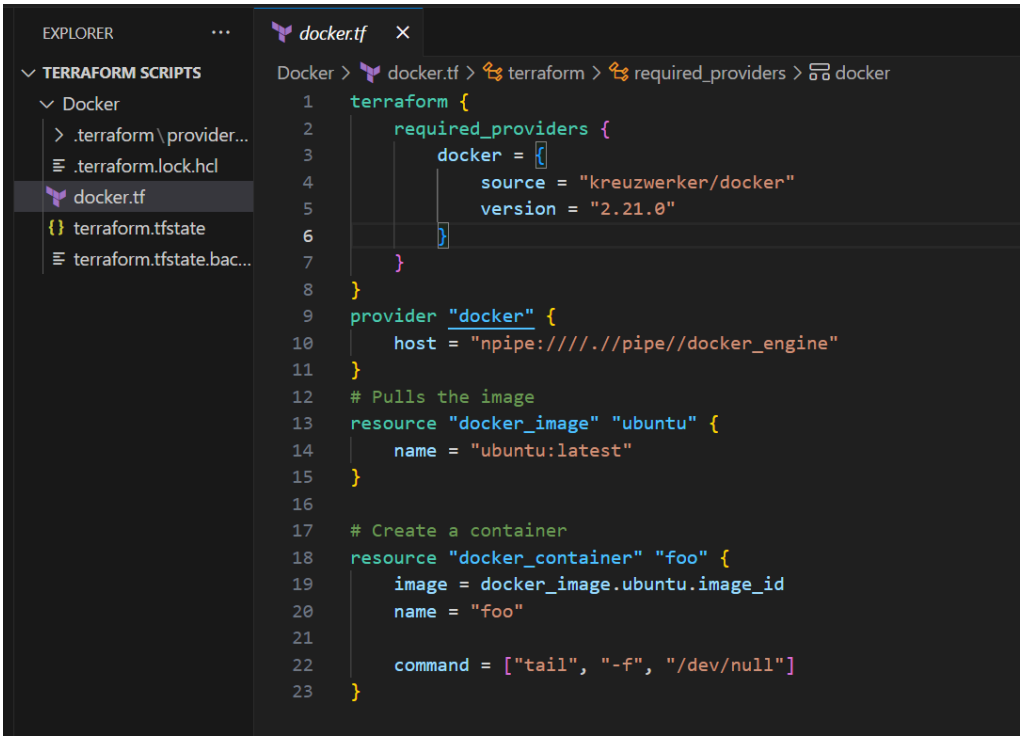
EXPERIMENT – 6

Aim

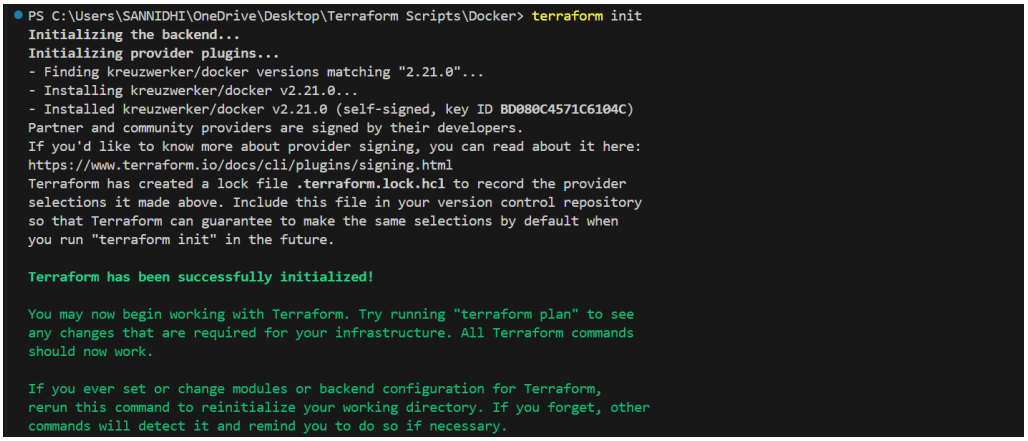
To Build, change, and destroy AWS infrastructure Using Terraform.

Implementation:

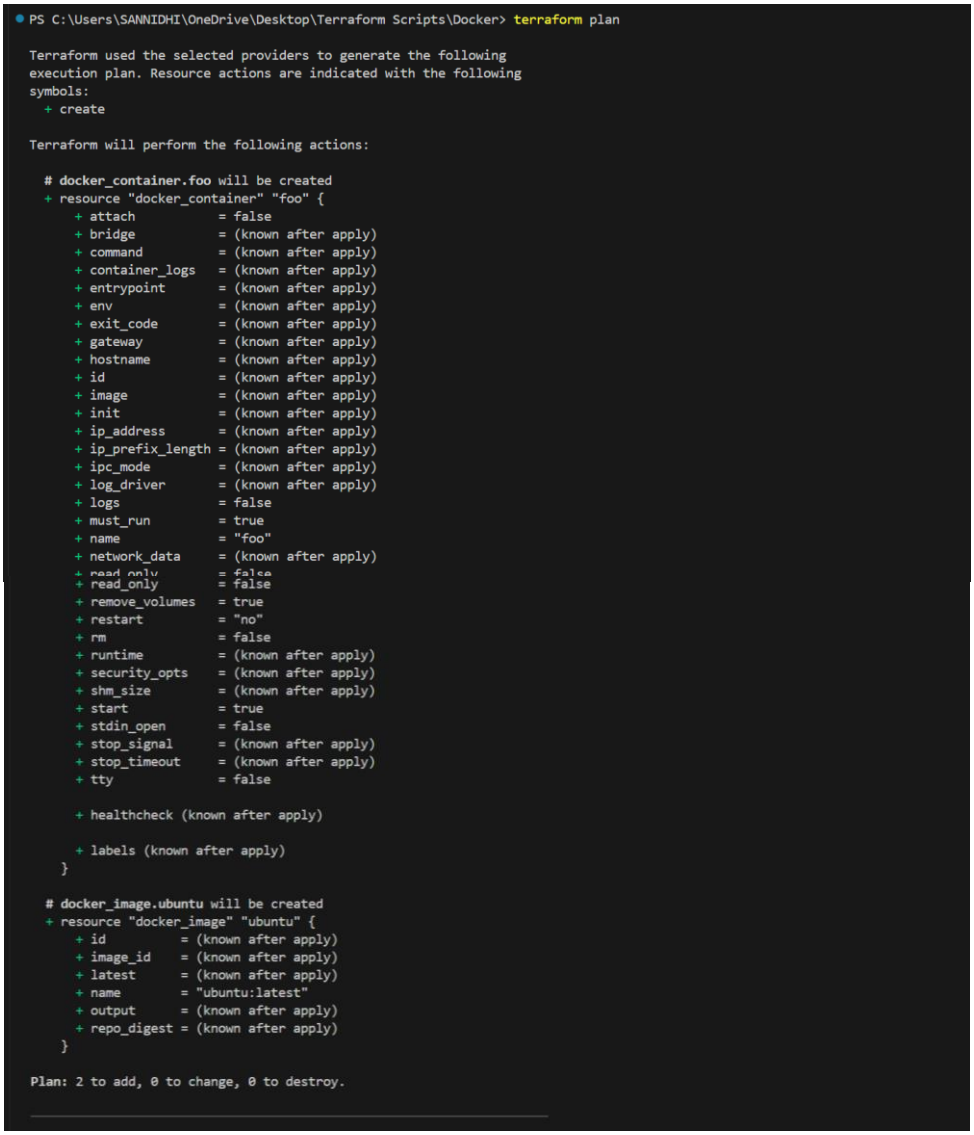
1) create a new folder named ‘Docker’ in the ‘TerraformScripts’ folder. Then create a new docker.tf file



2) Execute Terraform Init command to initialize the resources

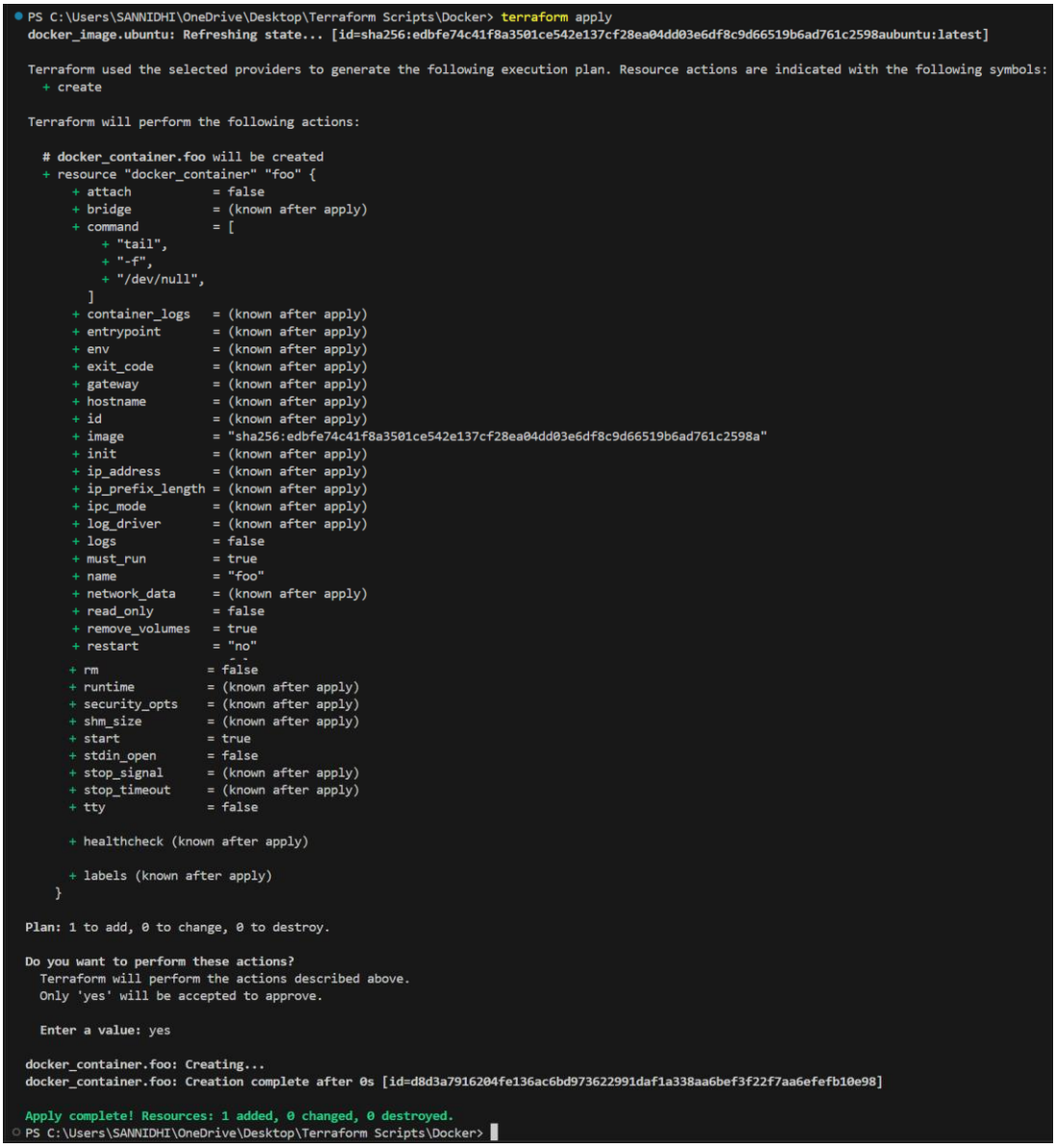


3) Execute Terraform plan to see the available resources

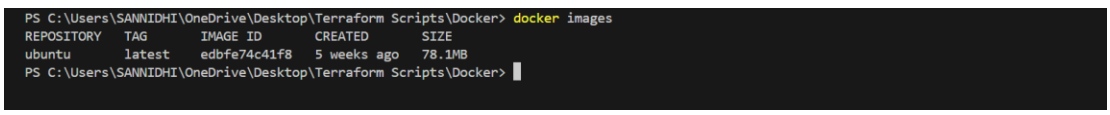


```
PS C:\Users\SANNIDHI\OneDrive\Desktop\Terraform Scripts\Docker> terraform plan

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.
```

4) Execute Terraform apply to apply the configuration, which will automatically create and run the Ubuntu Linux container based on our configuration.



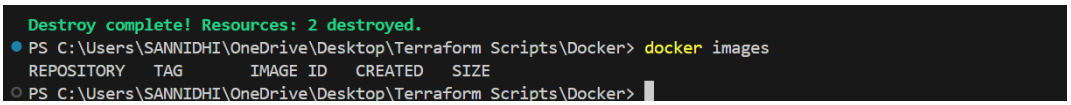
5) Docker images, After Executing Apply step



6) Execute Terraform destroy to delete the configuration, which will automatically delete the Ubuntu Container

```
PS C:\Users\SANNIDHI\OneDrive\Desktop\Terraform Scripts\Docker> terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=d8d3a7916204fe136ac6bd973622991daf1a338aa6bef3f22f7aa6efefb10e98]

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_container.foo will be destroyed
  - resource "docker_container" "foo" {
      - attach           = false -> null
      - command          = [
          - "tail",
          - "-f",
          - "/dev/null",
        ] -> null
      - cpu_shares       = 0 -> null
      - dns              = [] -> null
      - dns_opts         = [] -> null
      - dns_search       = [] -> null
      - entrypoint       = [] -> null
      - env              = [] -> null
      - gateway          = "172.17.0.1" -> null
      - group_add        = [] -> null
      - hostname         = "d8d3a7916204" -> null
      - id               = "d8d3a7916204fe136ac6bd973622991daf1a338aa6bef3f22f7aa6efefb10e98" -> null
      - image            = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
      - init             = false -> null
      - ip_address       = "172.17.0.2" -> null
      - ip_prefix_length = 16 -> null
      - ipc_mode         = "private" -> null
      - links            = [] -> null
      - log_driver       = "json-file" -> null
      - log_opts         = {} -> null
      - logs             = false -> null
      - max_retry_count  = 0 -> null
      - memory           = 0 -> null
      - memory_swap      = 0 -> null
      - must_run         = true -> null
      - name             = "foo" -> null
      - network_data     = [
          - {
              - gateway                   = "172.17.0.1"
              - global_ipv6_prefix_length = 0
              - ip_address                = "172.17.0.2"
              - ip_prefix_length          = 16
              - network_name              = "bridge"
                # (2 unchanged attributes hidden)
      - read_only        = false -> null
      - remove_volumes   = true -> null
      - restart          = "no" -> null
      - rm               = false -> null
      - runtime          = "runc" -> null
      - security_opts    = [] -> null
      - shm_size         = 64 -> null
      - start            = true -> null
      - stdin_open       = false -> null
      - stop_timeout     = 0 -> null
      - storage_opts     = {} -> null
      - sysctls          = {} -> null
      - tmpfs            = {} -> null
      - tty              = false -> null
        # (8 unchanged attributes hidden)
    }

  # 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, 2 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_container.foo: Destroying... [id=d8d3a7916204fe136ac6bd973622991daf1a338aa6bef3f22f7aa6efefb10e98]
docker_container.foo: Destruction complete after 1s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image.ubuntu: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
```

7) Docker images After Executing Destroy step



Conclusion

Successfully implemented Terraform to build, modify, and destroy AWS infrastructure, demonstrating the tool's ability to automate and streamline infrastructure management with consistency and efficiency.

EXPERIMENT 7

Aim:

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

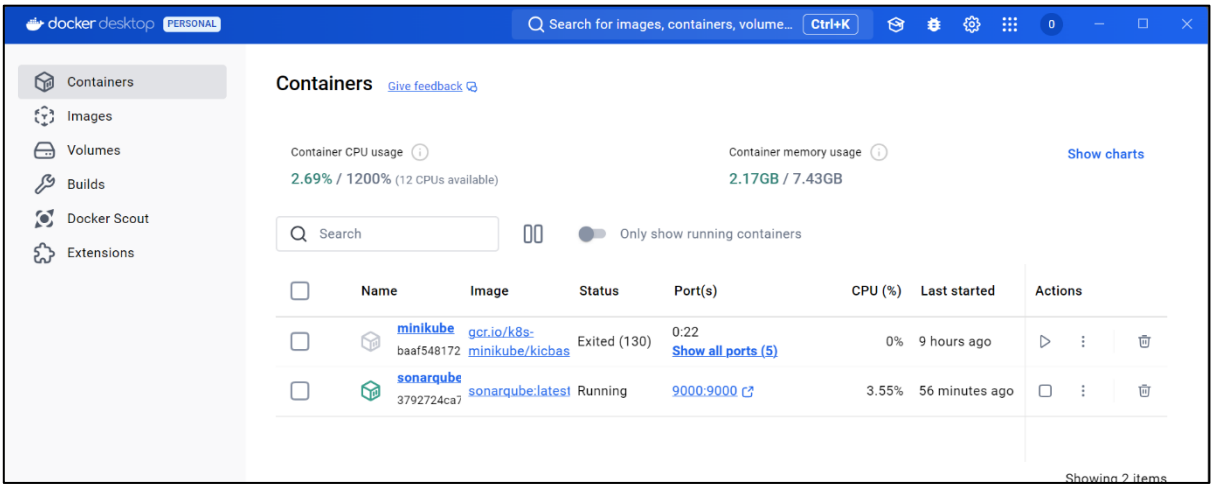
Implementation:

1. Open up Jenkins Dashboard on localhost, port 8080

2. Run SonarQube in a Docker container using command:

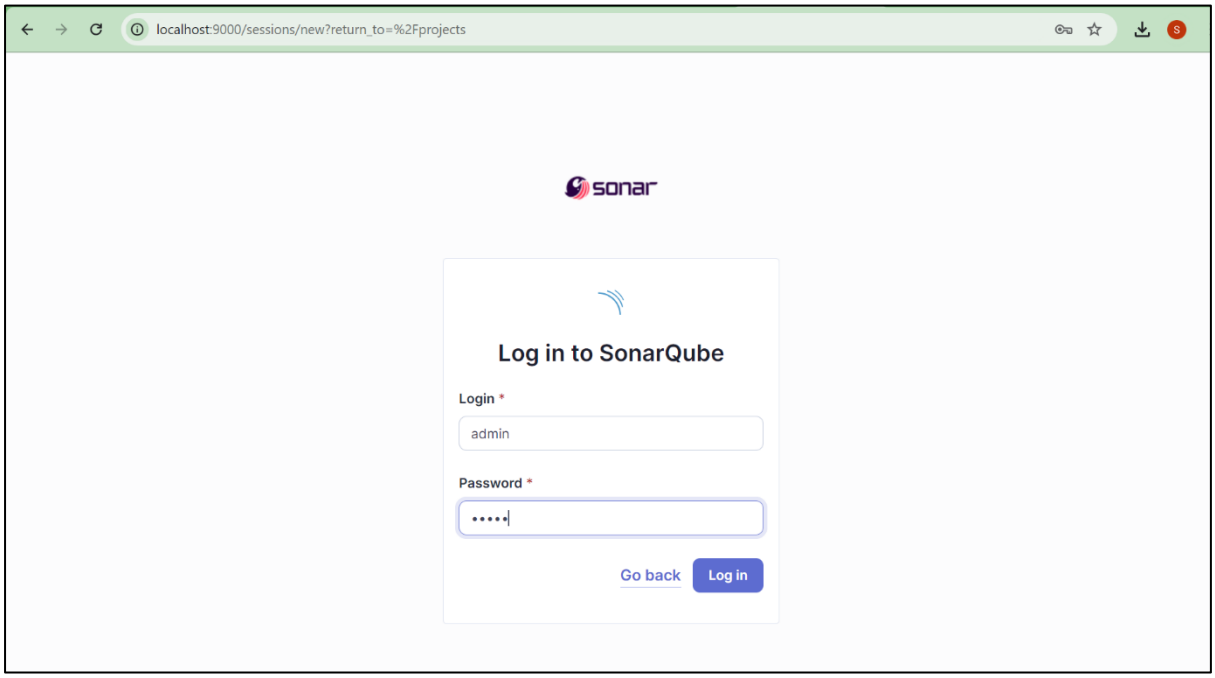
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

```
C:\Users\SANNIDHI>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CH
ECKS_DISABLE=true -p 9000:9000 sonarqube:latest
3792724ca79359d2a71b0612ce535e0afe353d9df28039abd3323e61eee6f9f0

C:\Users\SANNIDHI>
```

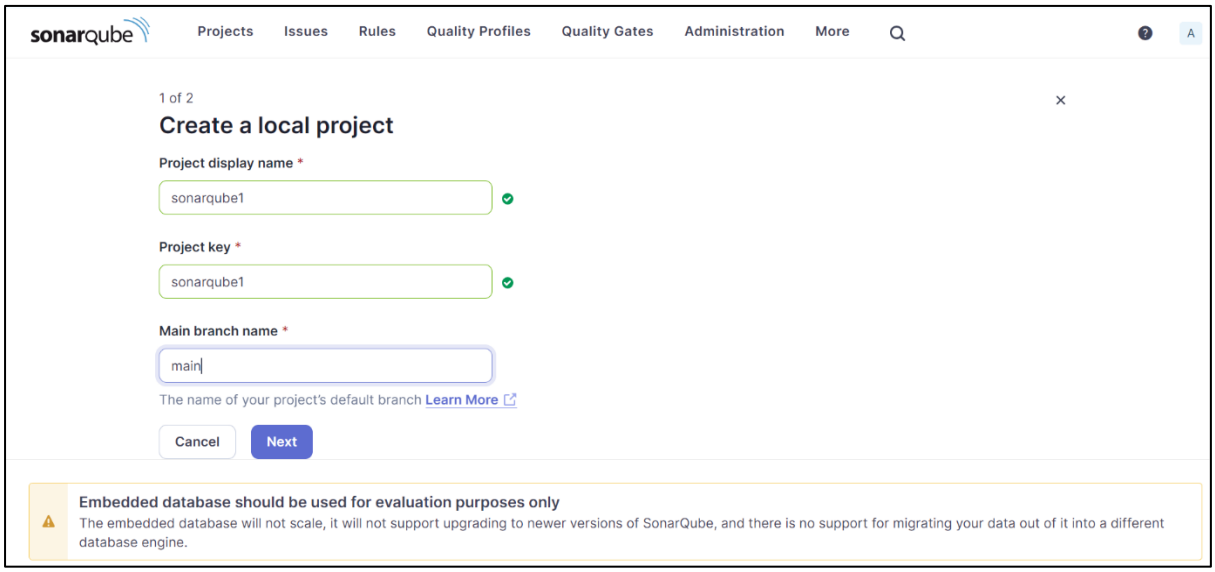


3. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.

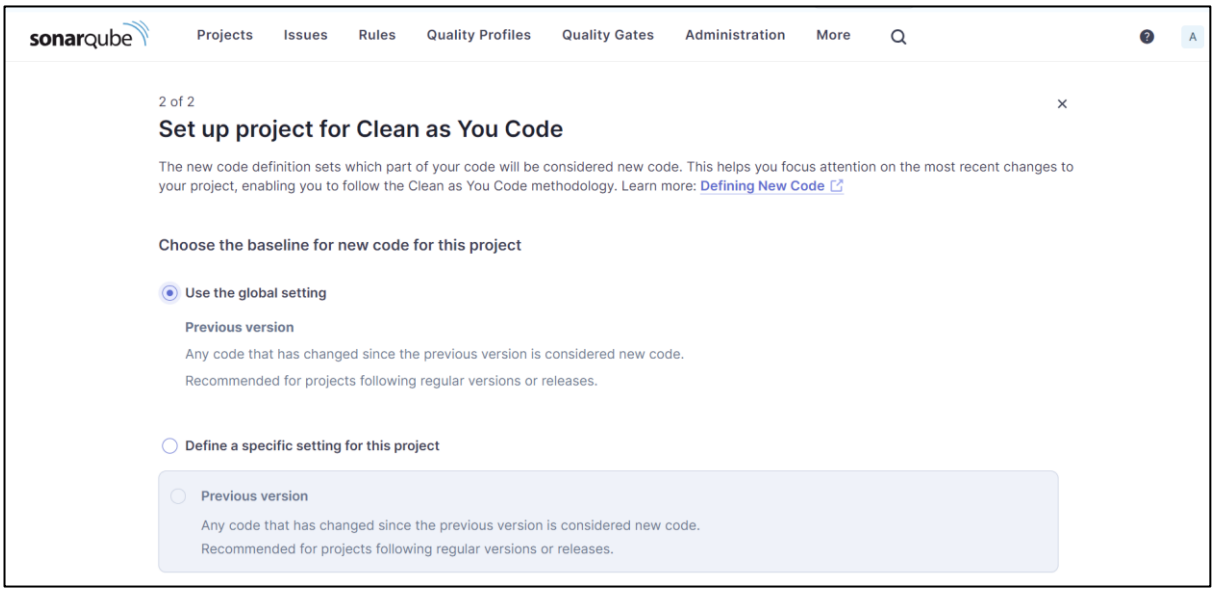
4. Login to SonarQube using username admin and password admin.



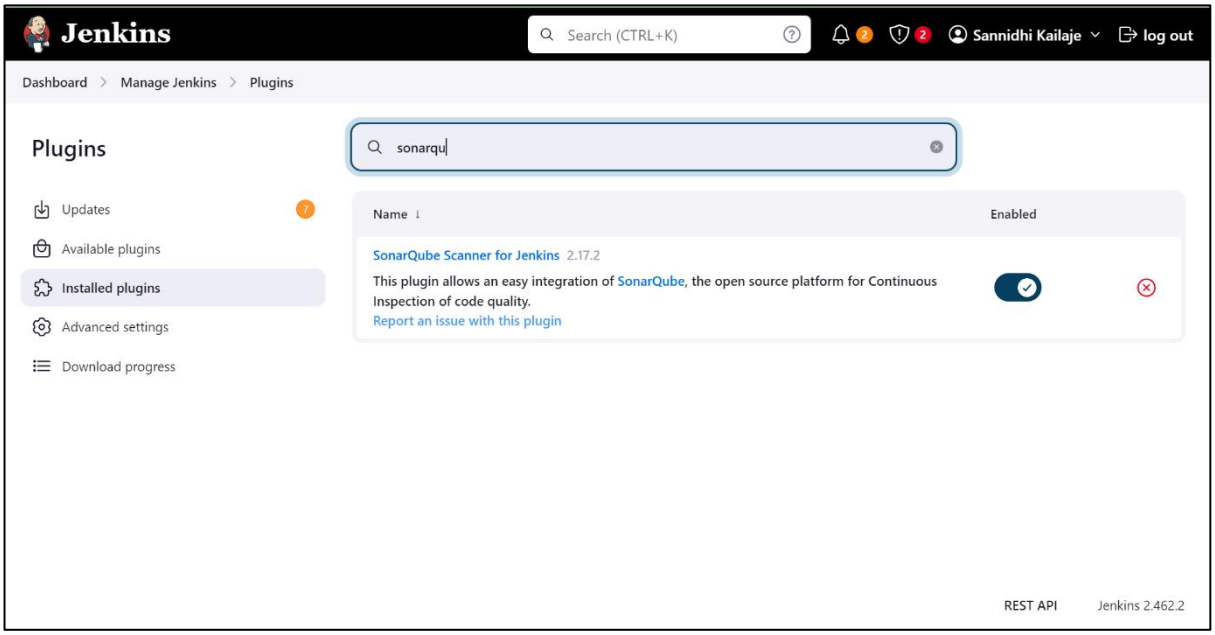
5. Create a manual project in SonarQube with the name sonarqube1.



Setup the project and come back to Jenkins Dashboard.

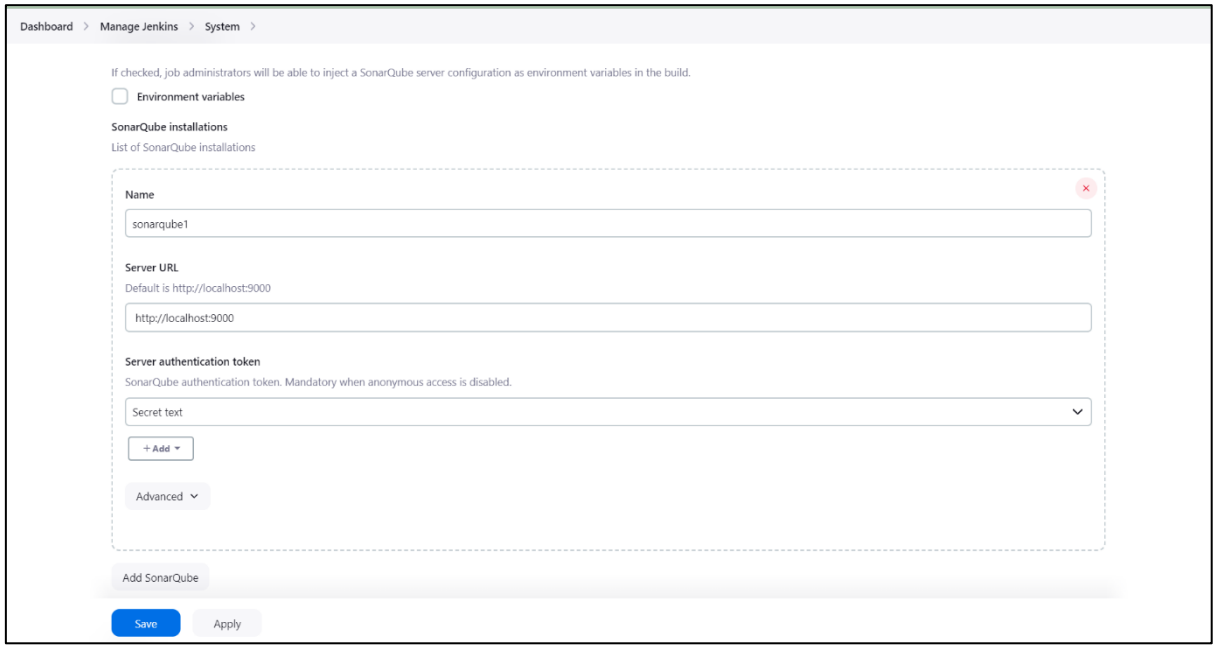


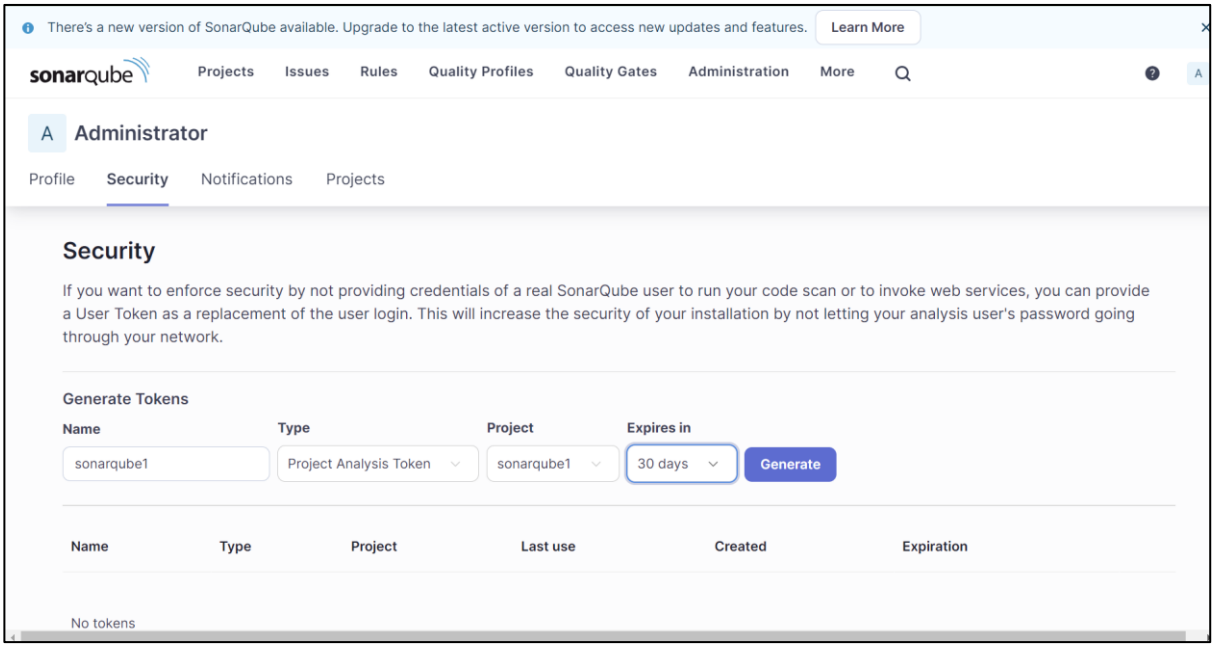
6. Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.



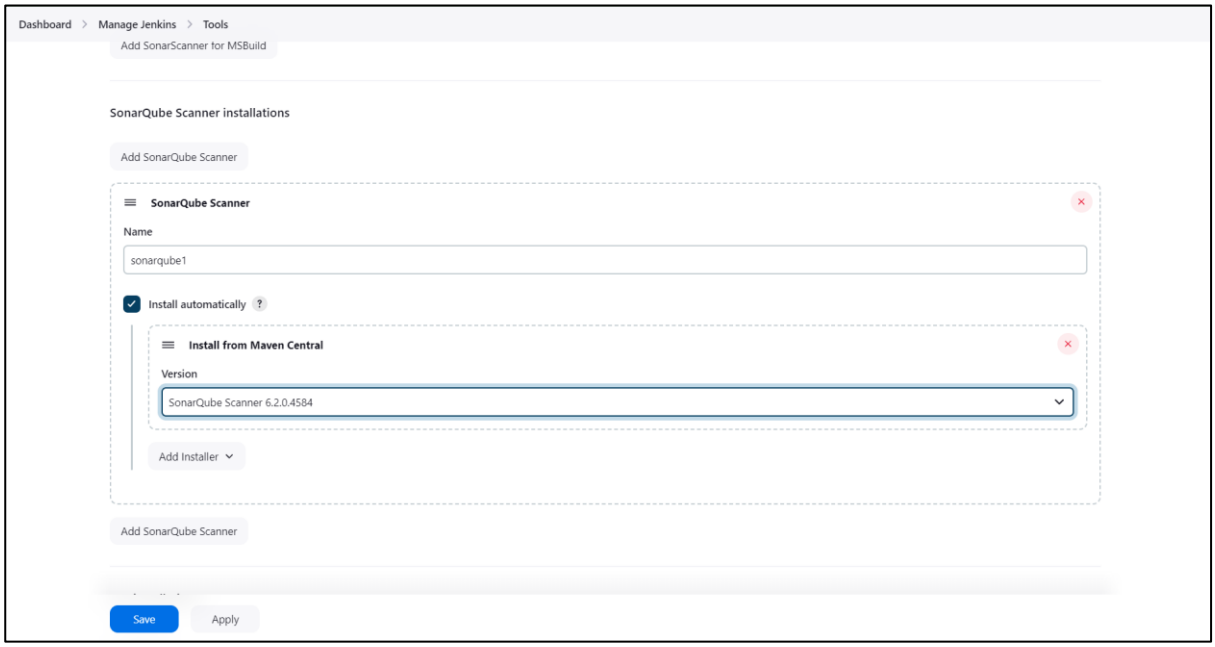
7. Go to Manage Jenkins > Configure System.

- Scroll down to the SonarQube Servers section and enter the required details:
  - Name: Any name you prefer.
  - Server URL: http://localhost:9000
  - Server Authentication Token: (Generate this token in SonarQube under My Account > Security > Generate Tokens).
  - Add Jenkins: Select Kind - Secret Text > Secret (Paste Generated Token)



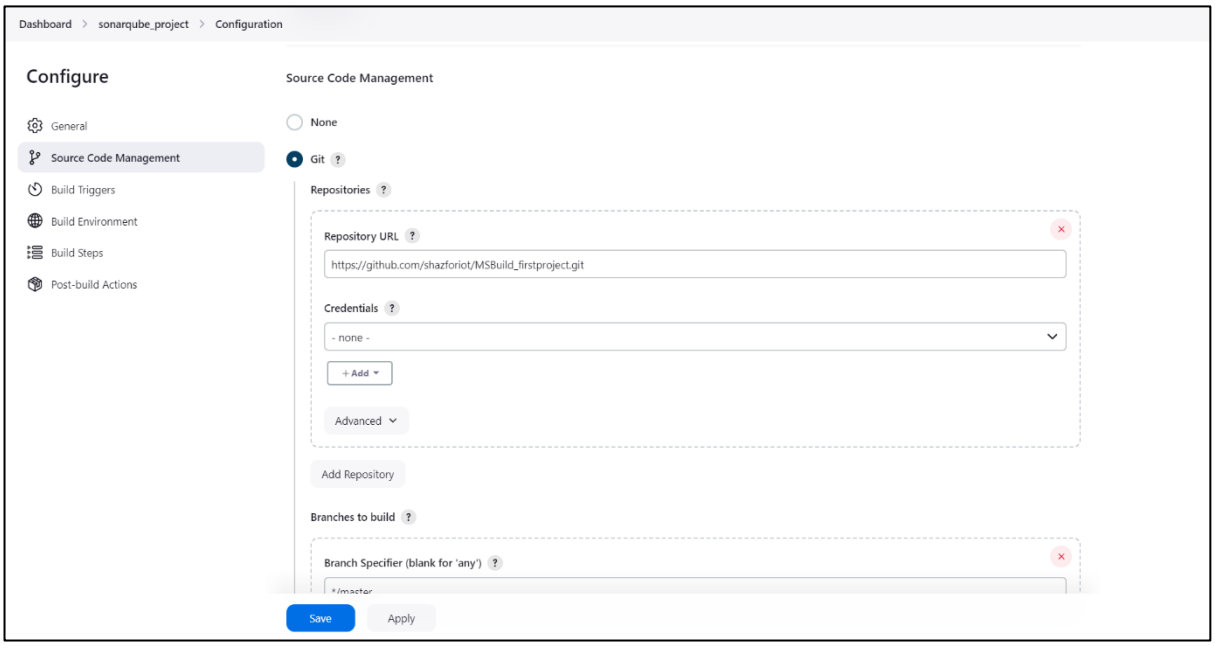


8. Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.

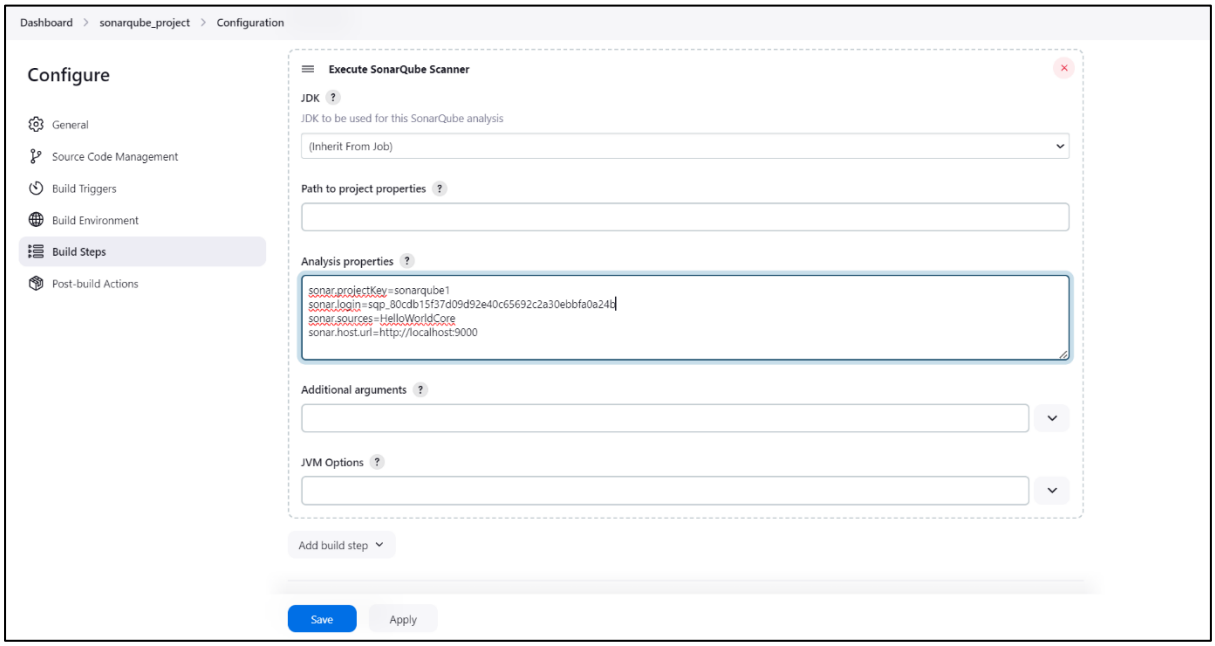


9. After the configuration, create a New Item in Jenkins, choose a freestyle project.

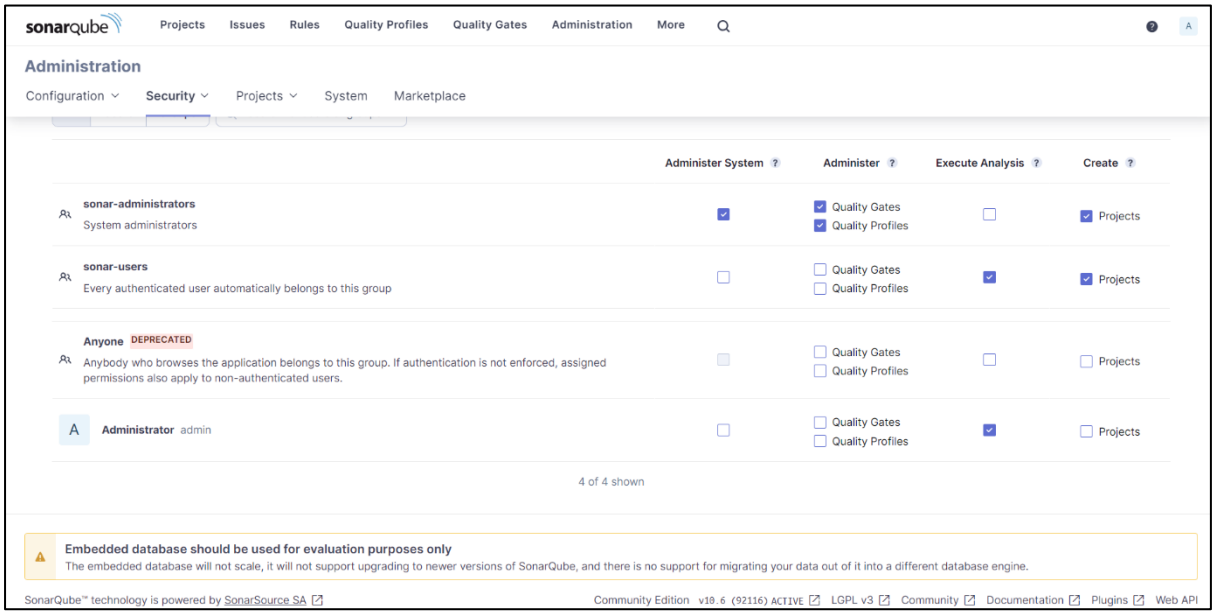
- Under Source Code Management, choose Git and enter the repository URL:
  - https://github.com/shazforiot/MSBuild_firstproject.git



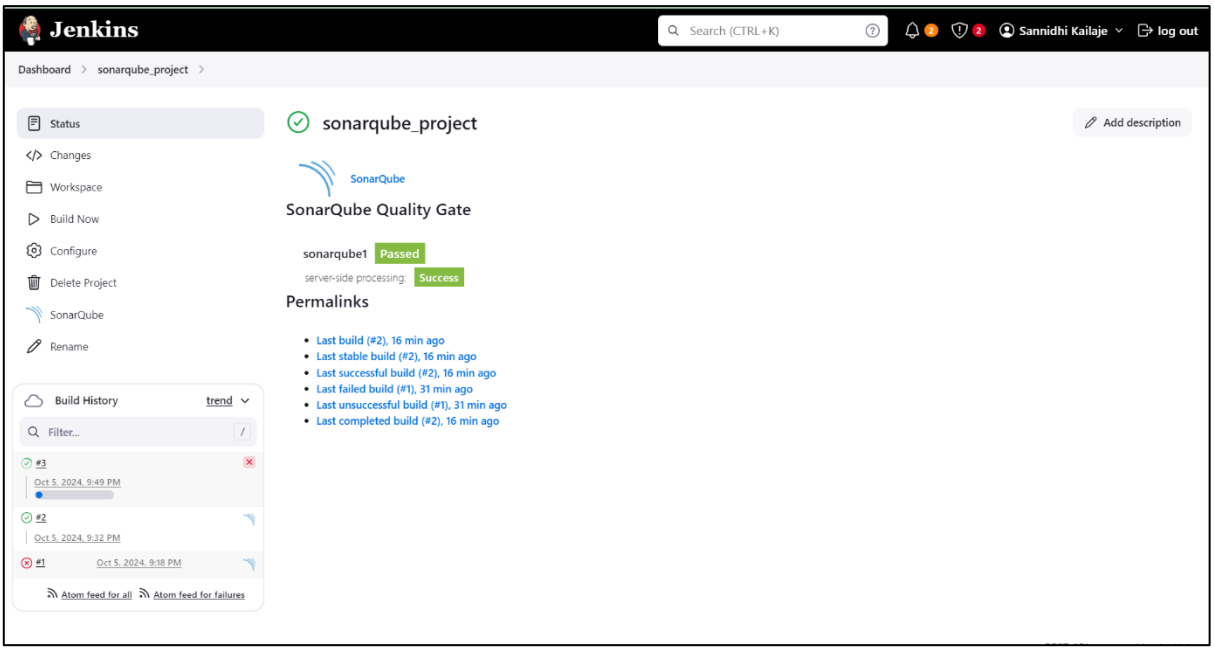
10. Under Build-> Execute SonarQube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.

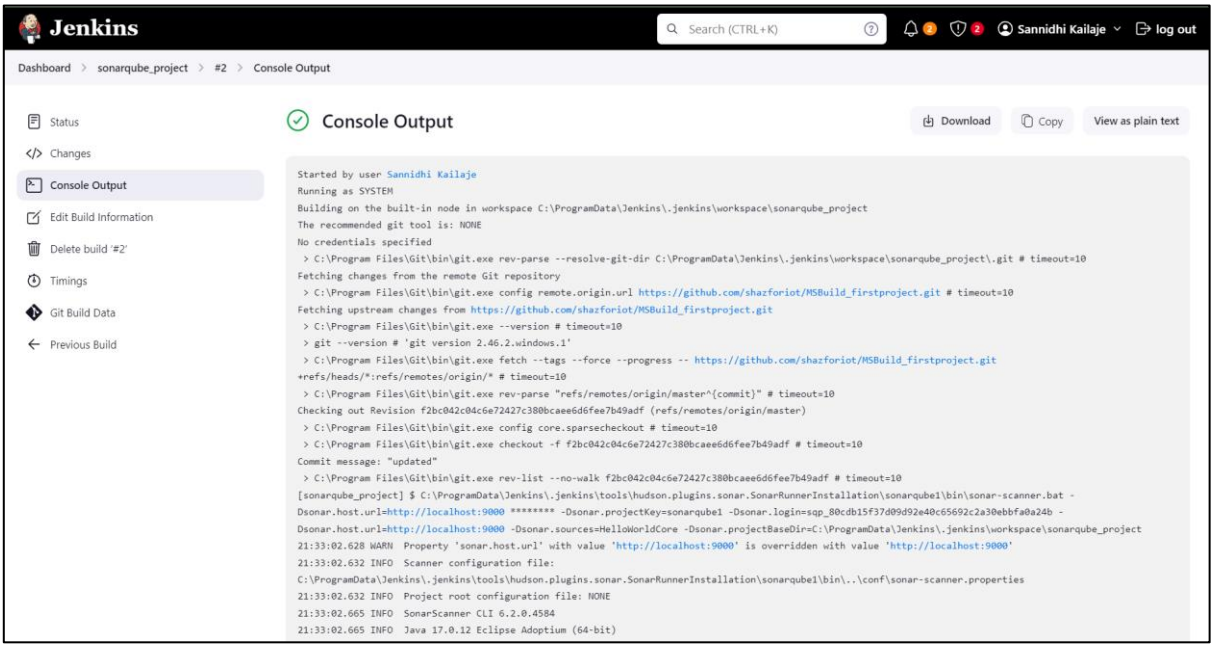


11. Go to http://localhost:9000/<user_name>/permissions and allow Execute Permissions to the Admin user.



12. Run The Build.





EXPERIMENT 8

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.

Implementation:

Prerequisites

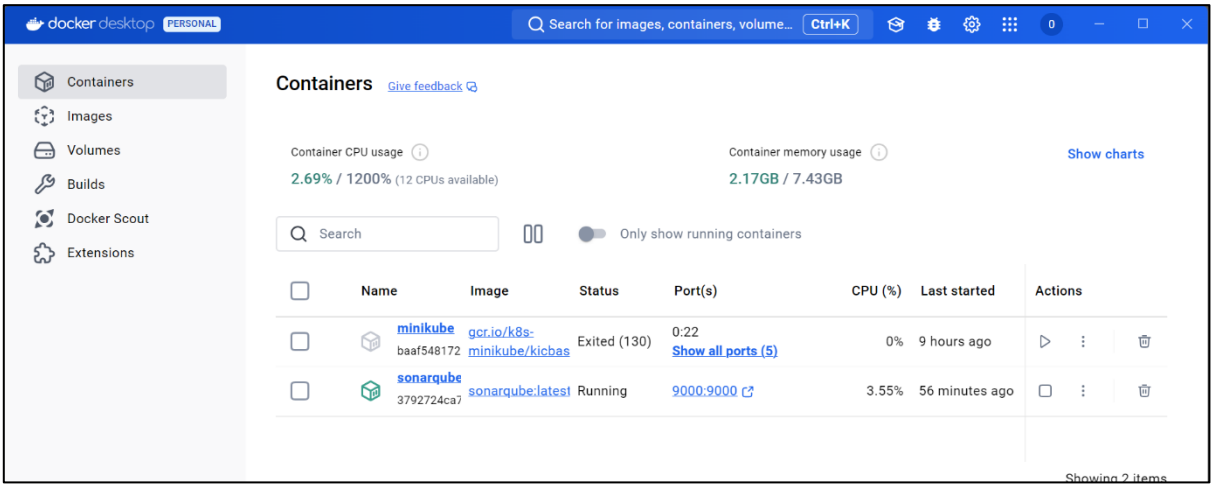
- Jenkins installed on your machine.
- Docker installed to run SonarQube.
- SonarQube installed via Docker.

1. Open up Jenkins Dashboard on localhost, port 8080

2. Run SonarQube in a Docker container using command:

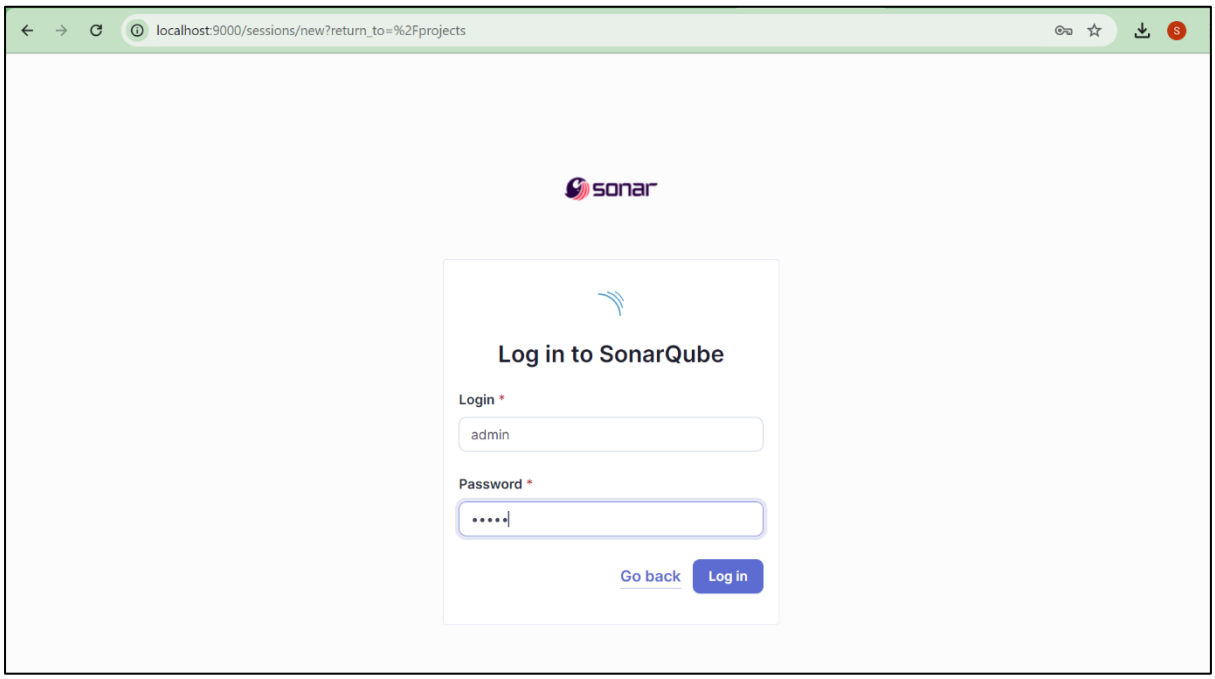
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

```
C:\Users\SANNIDHI>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CH
ECKS_DISABLE=true -p 9000:9000 sonarqube:latest
3792724ca79359d2a71b0612ce535e0afe353d9df28039abd3323e61eee6f9f0

C:\Users\SANNIDHI>
```

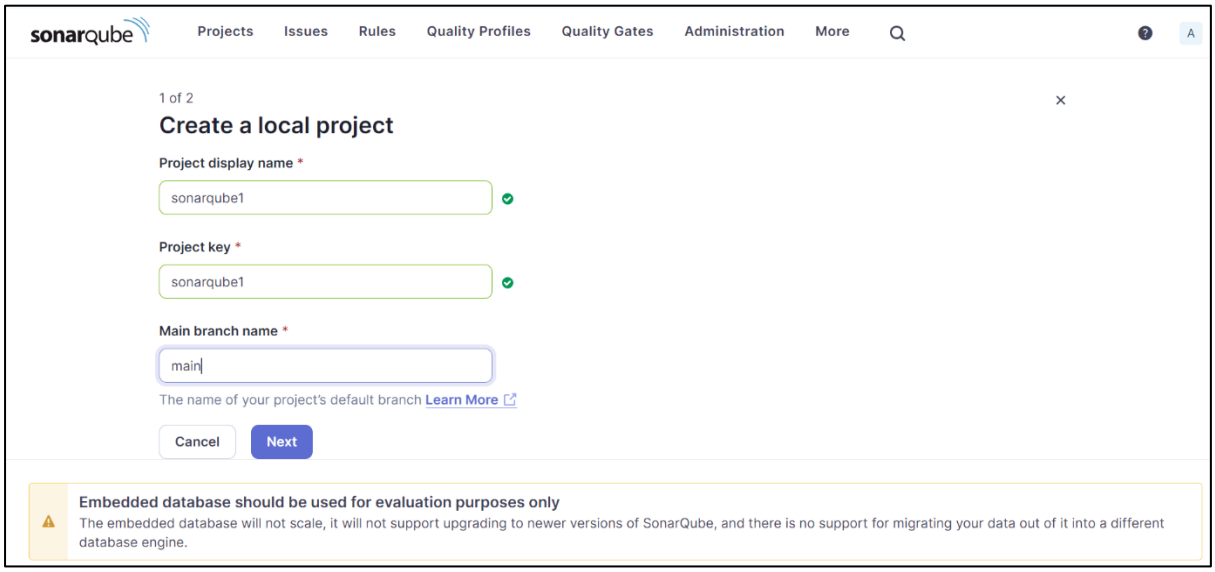


3. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.

4. Login to SonarQube with your credentials.



5. Create a manual project in SonarQube with the name sonarqube1.

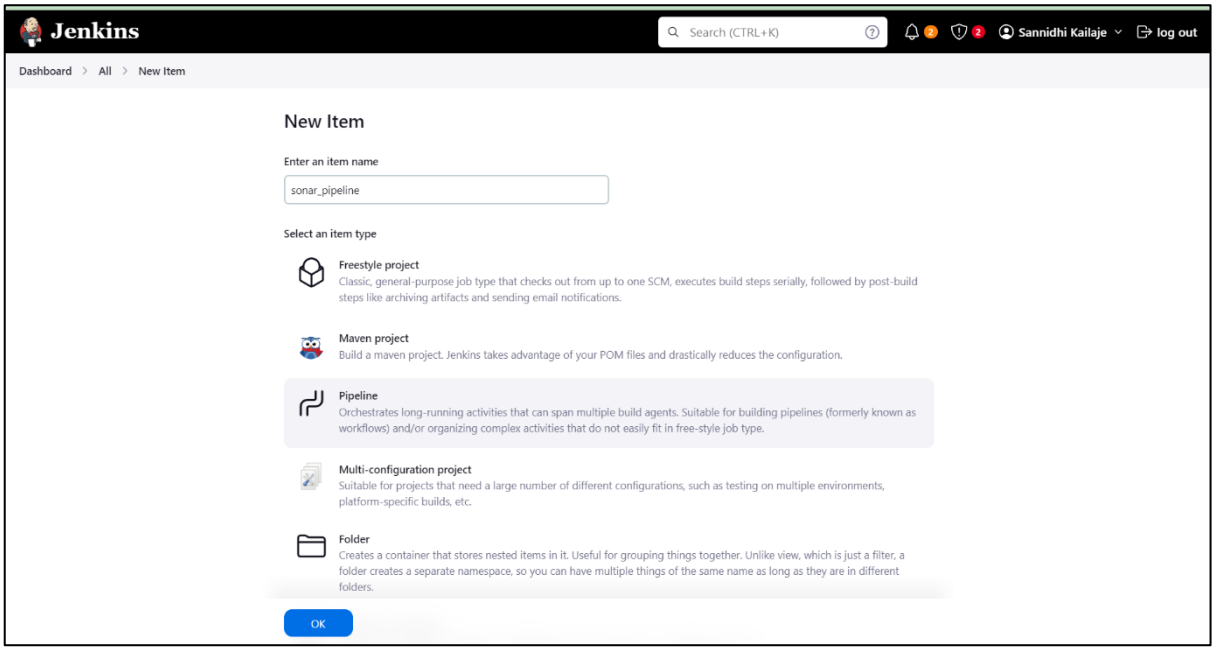


6. Generate SonarQube Token

- Go to My Account > Security > Generate Tokens.
- Copy the generated token for later use

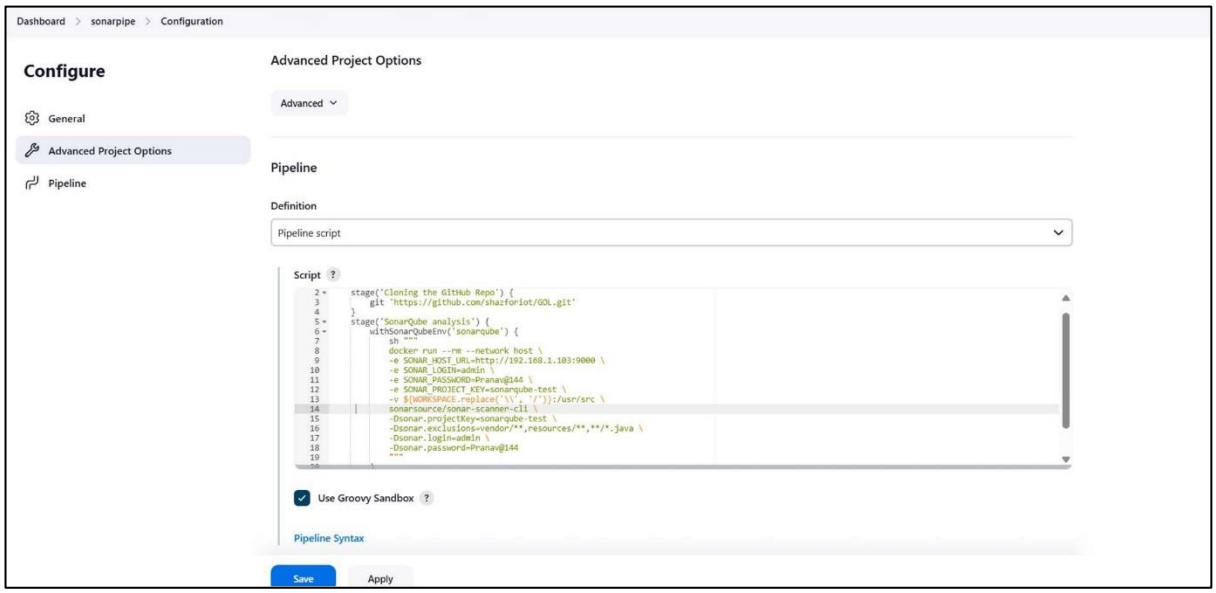
7. Create a Jenkins Pipeline

- Go to Jenkins Dashboard, click New Item, and select Pipeline.



8. Under Pipeline Script, enter the following script:

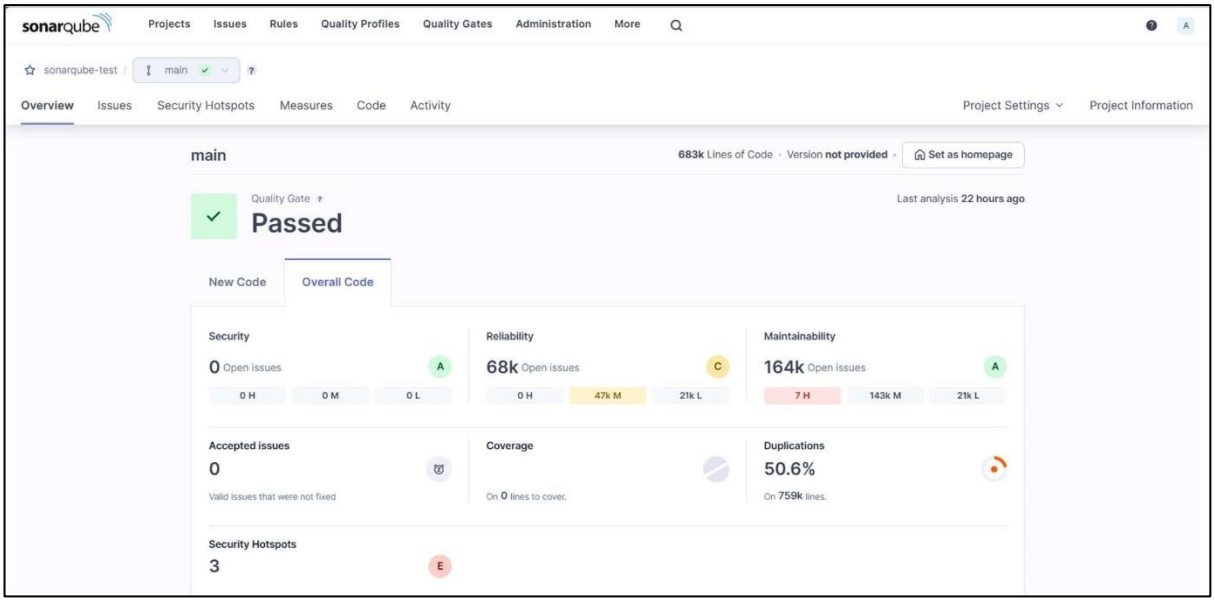
```
docker network create sonarnet
node {
 stage('Cloning the GitHub Repo') {
 git 'https://github.com/shazforiot/GOL.git'
 }
 stage('SonarQube analysis') {
 withSonarQubeEnv('sonarqube') {
        sh """
        docker run --rm --network host \
        -e SONAR_HOST_URL=http://<ip_address>:9000 \
        -e SONAR_LOGIN=admin \
        -e SONAR_PASSWORD=<Sonarqube_password> \
        -e SONAR_PROJECT_KEY=sonarqube-test \
        -v ${WORKSPACE.replace('\\', '/')}:/usr/src \
        sonarsource/sonar-scanner-cli \
        -Dsonar.projectKey=sonarqube-test \
        -Dsonar.exclusions=vendor/**,resources/**,**/*.java \
        -Dsonar.login=admin \
        -Dsonar.password=<Sonarqube_password>
        """
        }
    }
  }
```



9. Run the Pipeline
   - Save the pipeline and click Build Now.
   - Monitor the console output for any errors

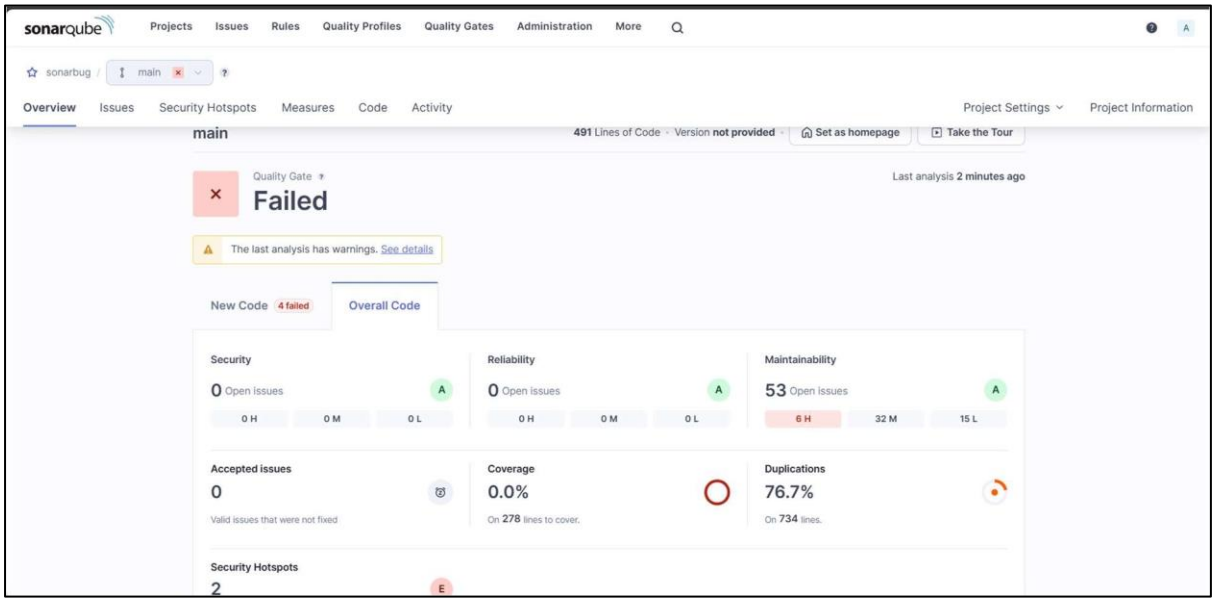
10. Check SonarQube for Analysis Results

- Go to your SonarQube dashboard and check the project for issues such as bugs, code smells, and security vulnerabilities.

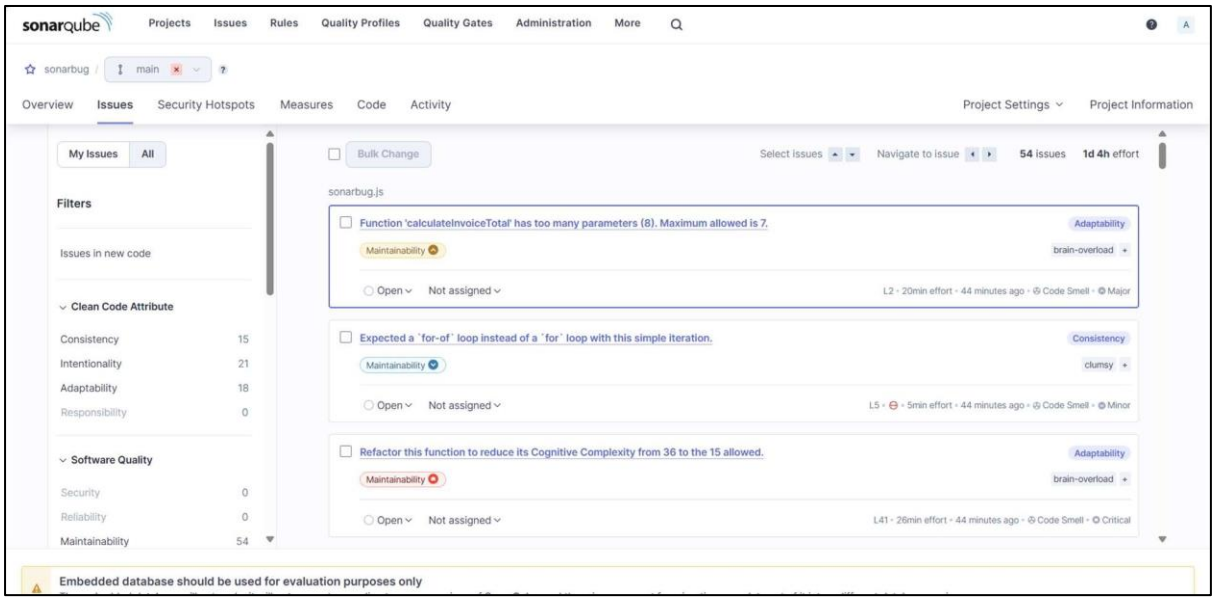


11. Checking SonarQube for Analysis Results of a Code File with Bugs, Code Smells, Security Vulnerabilities, Cyclomatic Complexities and Duplicates.

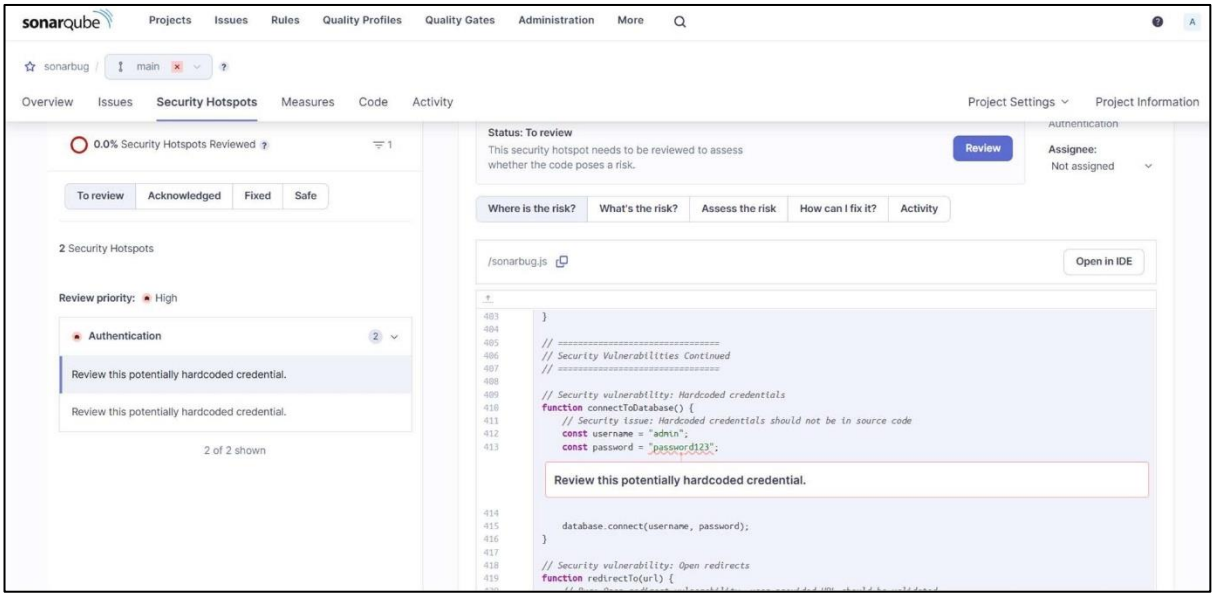
● Overview –



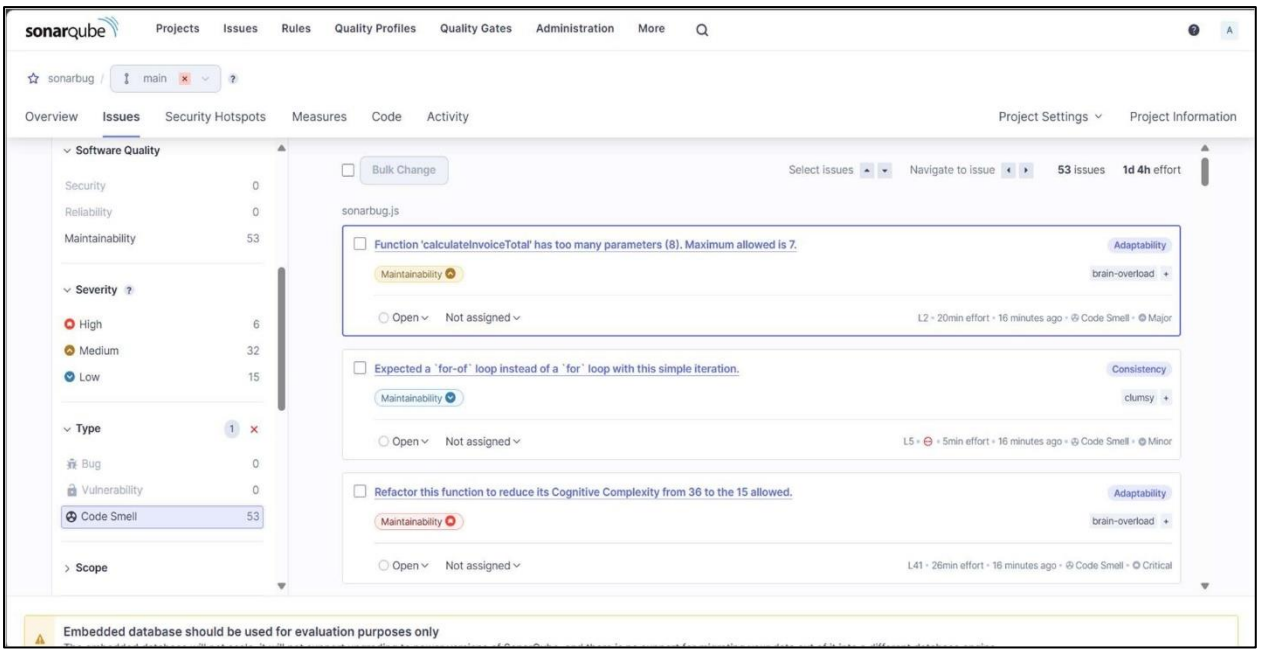
- Issues –

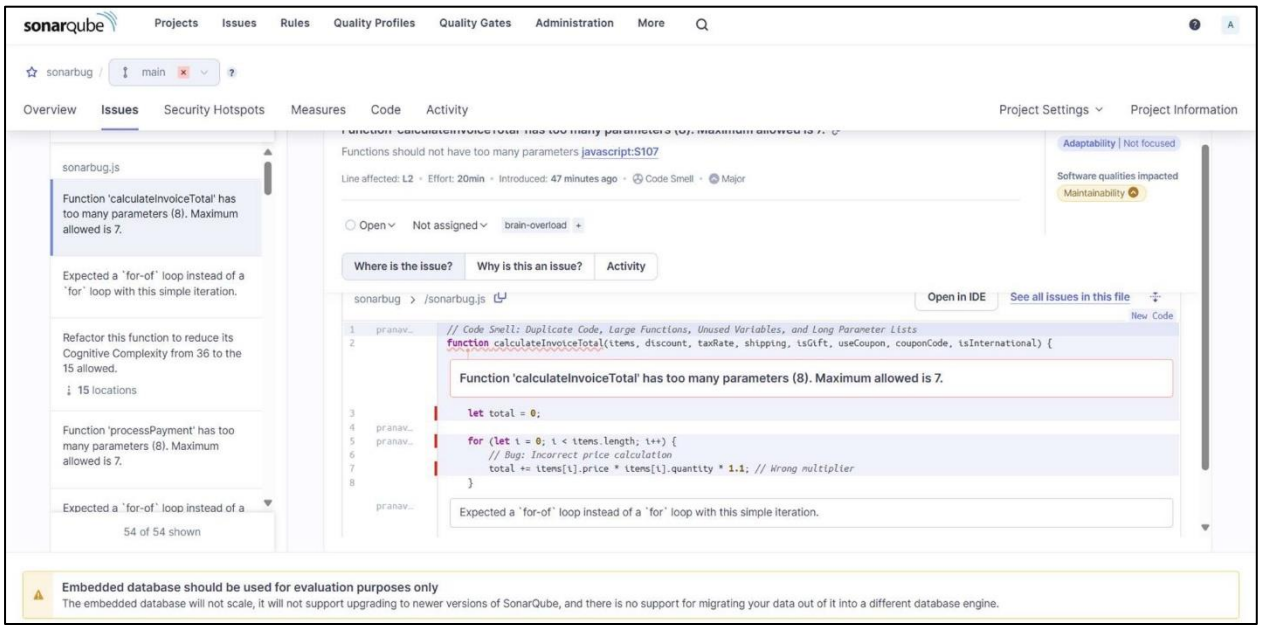


- Security Hotspot (Security Vulnerabilities) –

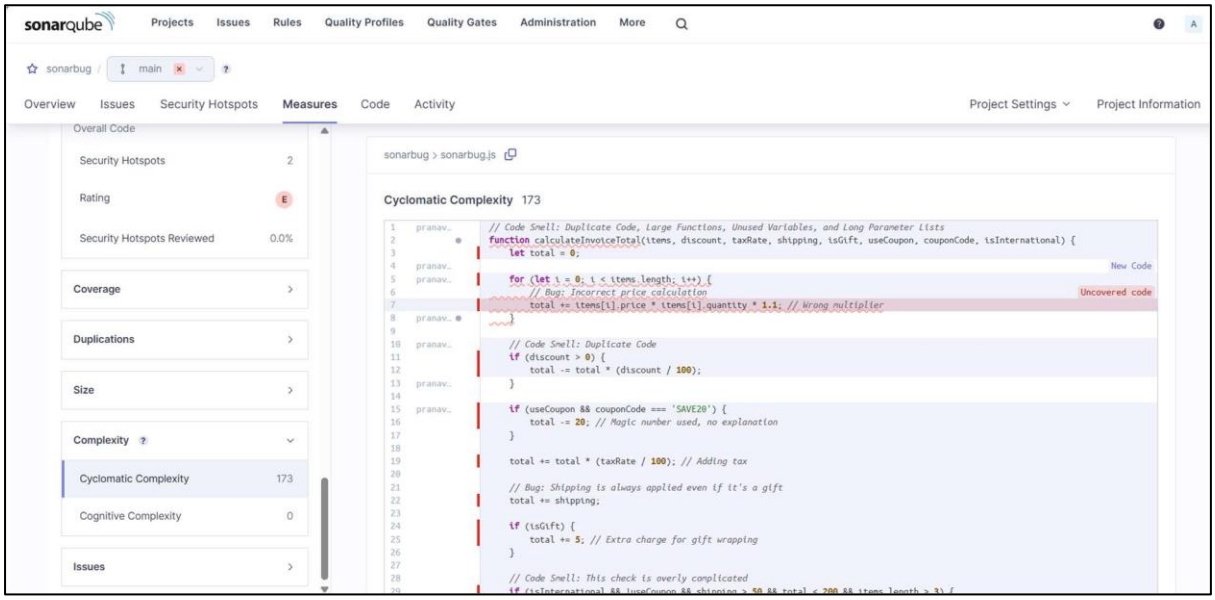


- Codesmells –

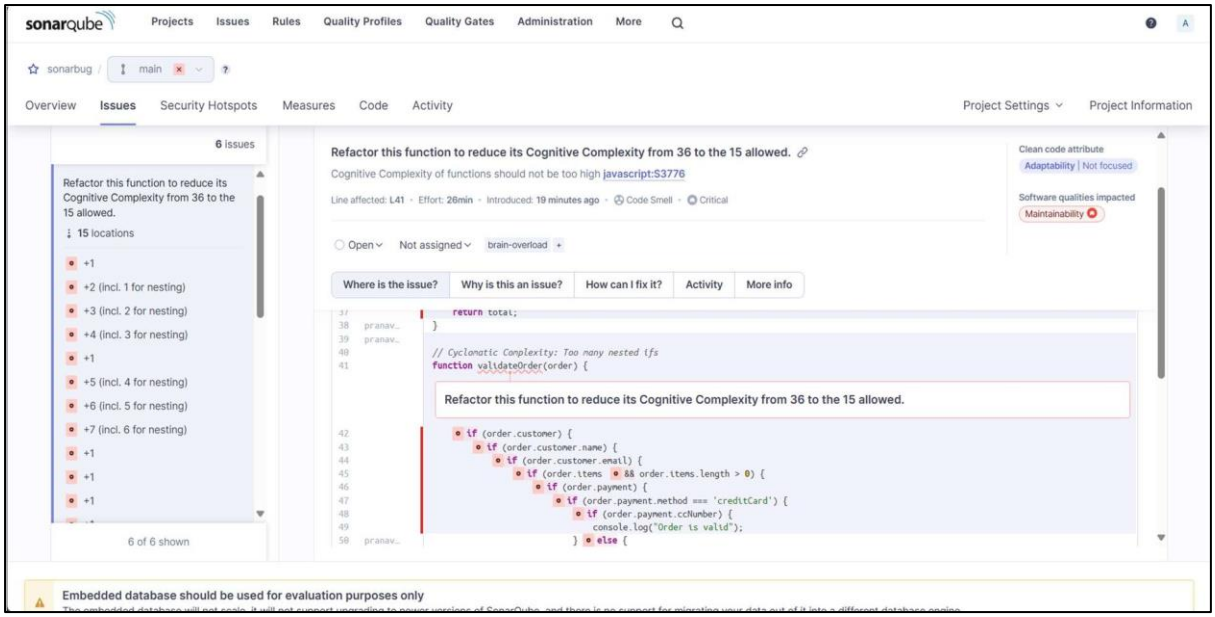

sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
sonarbug
main
Overview
Issues
Security Hotspots
Measures
Code
Activity
Project Settings
Project Information
Software Quality
Security 0
Reliability 0
Maintainability 53
Severity
High 6
Medium 32
Low 15
Type 1
Bug 0
Vulnerability 0
Code Smell 53
Scope
Bulk Change
Select issues
Navigate to issue
53 issues
1d 4h effort
sonarbug.js
Function 'calculateInvoiceTotal' has too many parameters (8). Maximum allowed is 7.
Adaptability
Maintainability
brain-overload
Open
Not assigned
L2 • 20min effort • 16 minutes ago • Code Smell • Major
Expected a `for-of` loop instead of a `for` loop with this simple iteration.
Consistency
clumsy
L5 • 5min effort • 16 minutes ago • Code Smell • Minor
Refactor this function to reduce its Cognitive Complexity from 36 to the 15 allowed.
L41 • 26min effort • 16 minutes ago • Code Smell • Critical
Embedded database should be used for evaluation purposes only


sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
sonarbug
main
Overview
Issues
Security Hotspots
Measures
Code
Activity
Project Settings
Project Information
sonarbug.js
Function 'calculateInvoiceTotal' has too many parameters (8). Maximum allowed is 7.
Expected a `for-of` loop instead of a `for` loop with this simple iteration.
Refactor this function to reduce its Cognitive Complexity from 36 to the 15 allowed.
15 locations
Function 'processPayment' has too many parameters (8). Maximum allowed is 7.
54 of 54 shown
Functions should not have too many parameters javascript:S107
Line affected: L2 • Effort: 20min • Introduced: 47 minutes ago • Code Smell • Major
Adaptability | Not focused
Software qualities impacted
Maintainability
Open
Not assigned
brain-overload
Where is the issue?
Why is this an issue?
Activity
sonarbug > /sonarbug.js
Open in IDE
See all issues in this file
New Code
// Code Smell: Duplicate Code, Large Functions, Unused Variables, and Long Parameter Lists
function calculateInvoiceTotal(items, discount, taxRate, shipping, isGift, useCoupon, couponCode, isInternational) {
Function 'calculateInvoiceTotal' has too many parameters (8). Maximum allowed is 7.
let total = 0;
for (let i = 0; i < items.length; i++) {
// Bug: Incorrect price calculation
total += items[i].price * items[i].quantity * 1.1; // Wrong multiplier
}
Expected a `for-of` loop instead of a `for` loop with this simple iteration.
Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

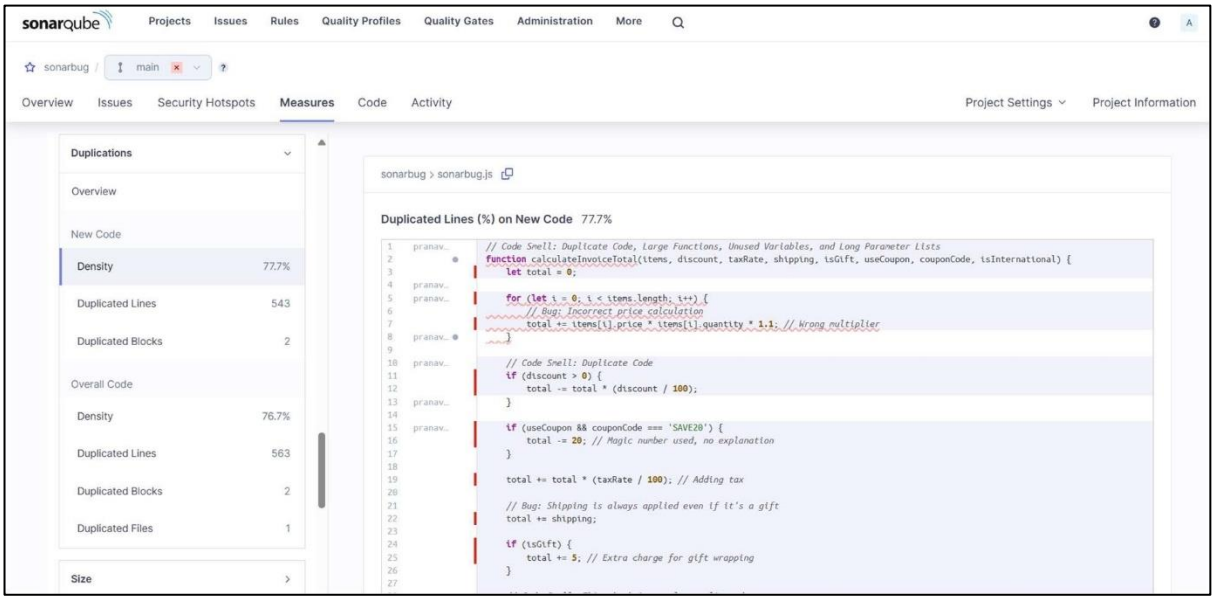
● Cyclomatic Complexity –



● Cognitive Complexity –



● Duplications –



Conclusion:

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

EXPERIMENT NO. 9

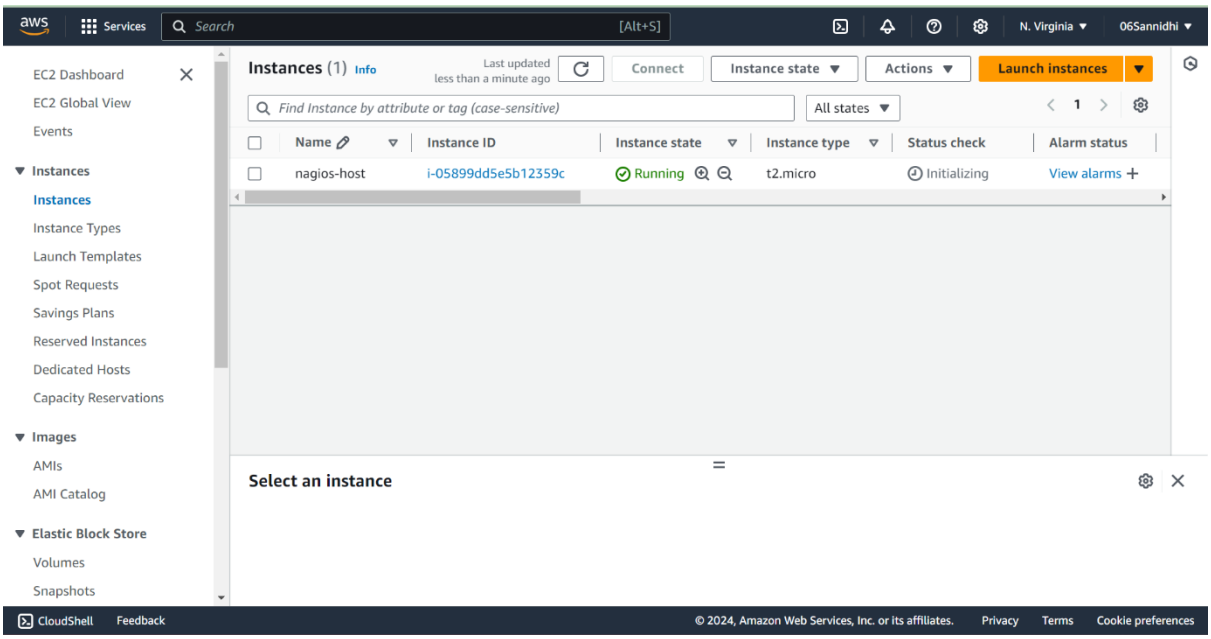
Aim:

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

Implementation:

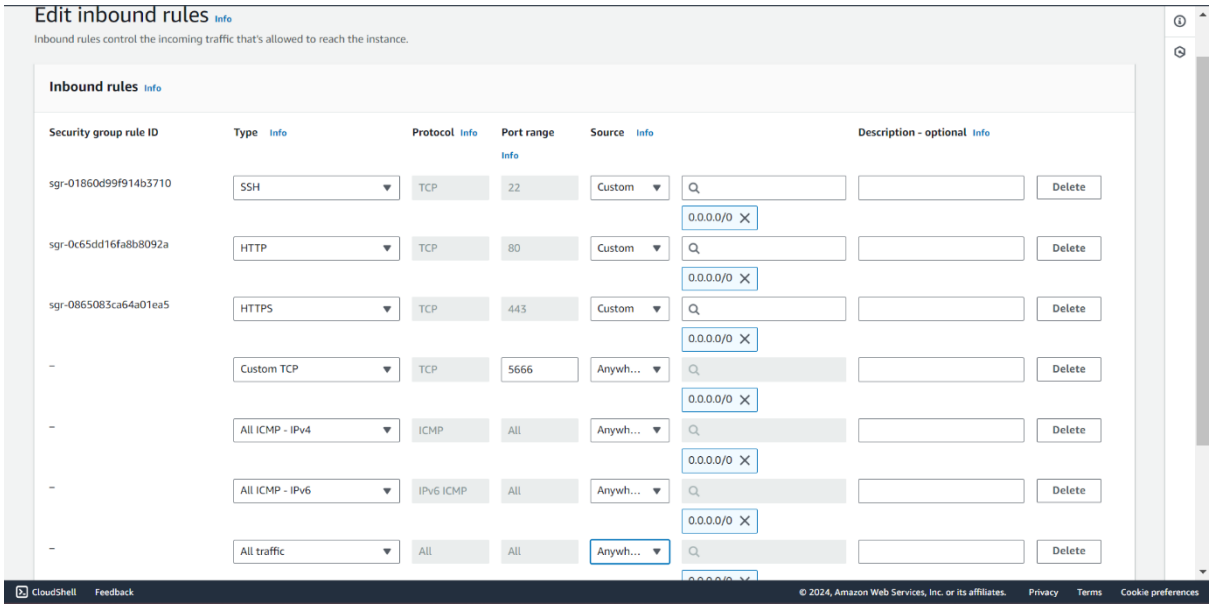
1. Create an Amazon Linux EC2 Instance

- Name it nagios-host.



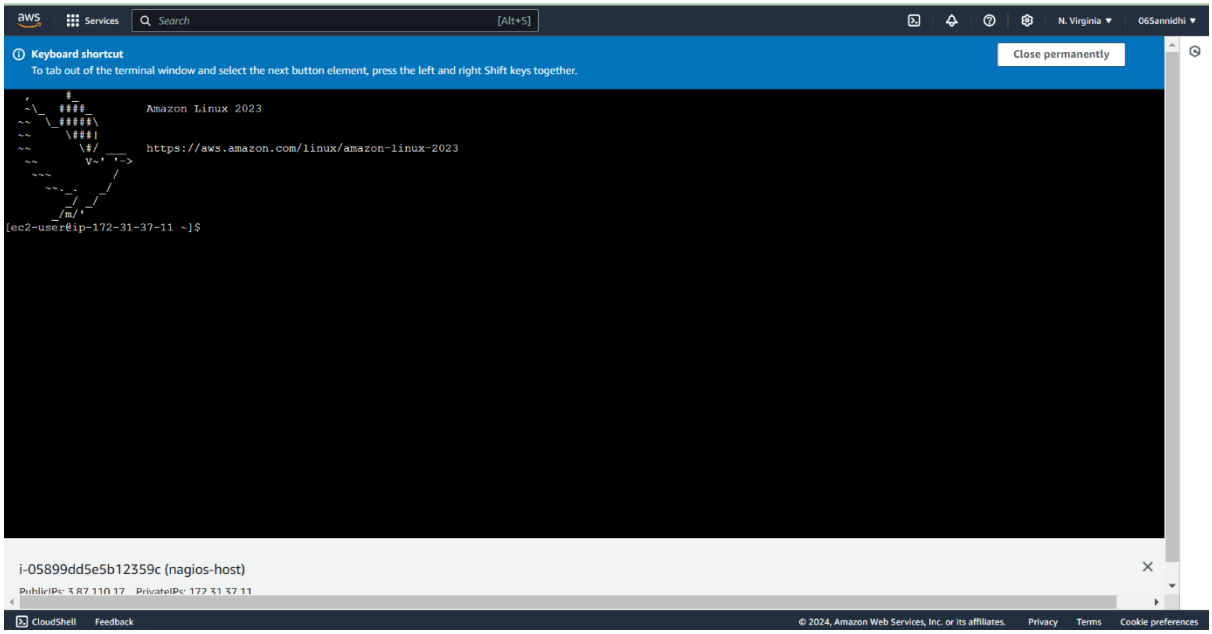
2. Configure Security Group

- Ensure HTTP, HTTPS, SSH, and ICMP are open from everywhere.
- Edit the inbound rules of the specified Security Group.



3. Connect to Your EC2 Instance

- SSH into your EC2 instance or use EC2 Instance Connect from the browser



4. Update Package Indices and Install Required Packages

Commands -

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

```
[ec2-user@ip-172-31-80-22 ~]$ sudo yum update -y
Last metadata expiration check: 0:09:18 ago on Thu Sep 26 08:41:50 2024.
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-80-22 ~]$ sudo yum install -y httpd php
Last metadata expiration check: 0:09:40 ago on Thu Sep 26 08:41:50 2024.
Dependencies resolved.
================================================================================
 Package                                Architecture                Version
================================================================================
```

```
Complete!
[ec2-user@ip-172-31-80-22 ~]$ sudo yum install -y gcc glibc glibc-common
sudo yum install -y gd gd-devel
Last metadata expiration check: 0:09:57 ago on Thu Sep 26 08:41:50 2024.
Package glibc-2.34-52.amzn2023.0.11.x86_64 is already installed.
Package glibc-common-2.34-52.amzn2023.0.11.x86_64 is already installed.
Dependencies resolved.
================================================================================
 Package                                Architecture                Version
================================================================================
Installing:
 gcc                                    x86_64                      11.4.1-2.amzn2023.0.2
Installing dependencies:
```

5. Create a New Nagios User

Commands -

sudo adduser -m nagios

sudo passwd nagios

```
Complete!
[ec2-user@ip-172-31-37-11 ~]$ sudo useradd nagios
[ec2-user@ip-172-31-37-11 ~]$ sudo passwd nagios
Changing password for user nagios.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
Sorry, passwords do not match.
New password:
BAD PASSWORD: The password contains the user name in some form
Retype new password:
Sorry, passwords do not match.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-37-11 ~]$
```

6. Create a New User Group

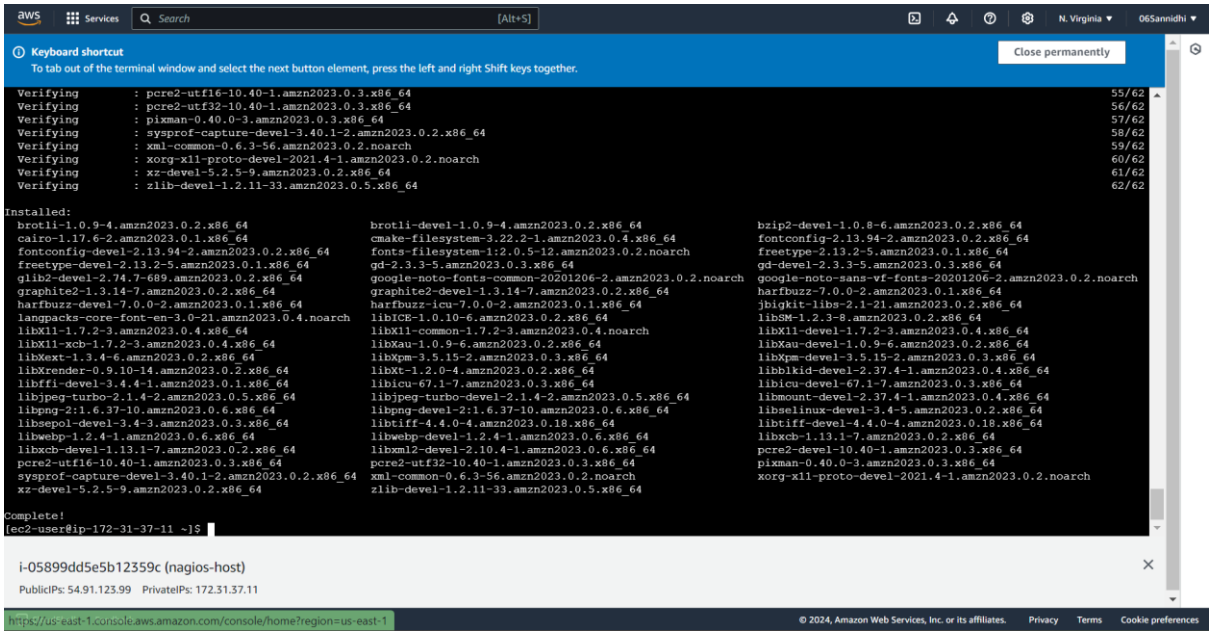
Commands -

sudo groupadd nagcmd

```
[ec2-user@ip-172-31-37-11 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-37-11 ~]$
```

i-05899dd5e5b12359c (nagios-host)

PublicIPs: 54.91.123.99 PrivateIPs: 172.31.37.11



7. Add Users to the Group

Commands -

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

```
[ec2-user@ip-172-31-37-11 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-37-11 ~]$ sudo usermod -aG nagcmd nagios
[ec2-user@ip-172-31-37-11 ~]$ sudo usermod -aG nagcmd apache
```

8. Create a Directory for Nagios Downloads

Commands -

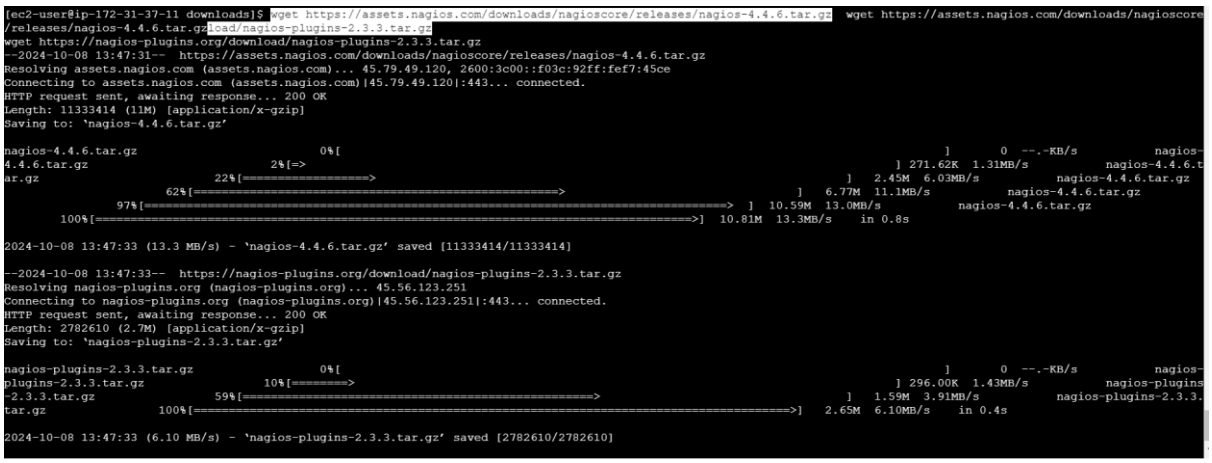
mkdir ~/downloads
cd ~/downloads

```
[ec2-user@ip-172-31-37-11 ~]$ mkdir ~/downloads
[ec2-user@ip-172-31-37-11 ~]$ cd ~/downloads
```

9. Download Nagios and Plugins Source Files

Commands -

Wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.6.tar.gz
wget https://nagios-plugins.org/download/nagios-plugins-2.3.3.tar.gz



10. Extract the Nagios Source File

Commands -

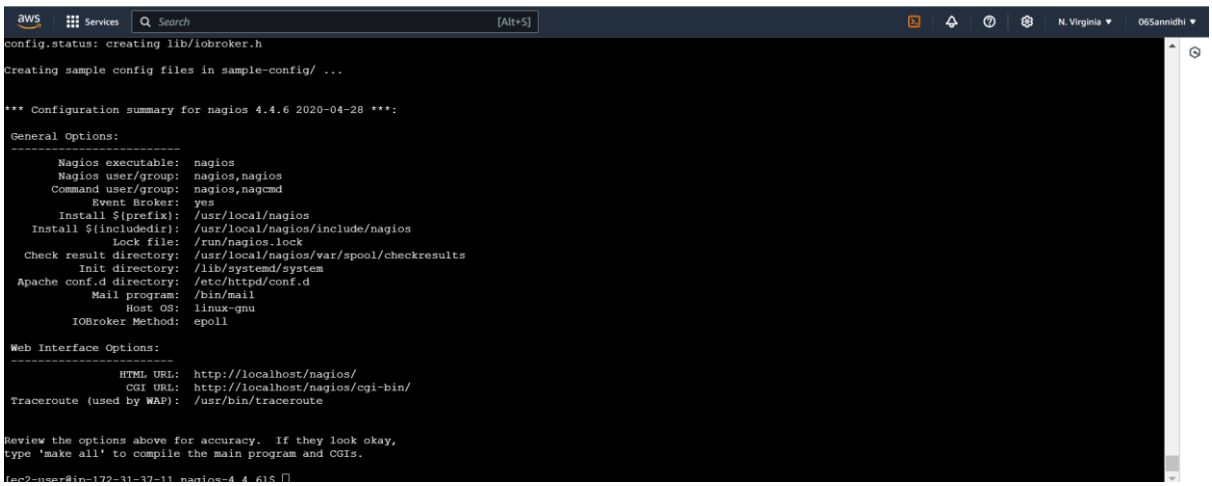
tar zxvf nagios-4.4.6.tar.gz
cd nagios-4.4.6

```
[ec2-user@ip-172-31-37-11 downloads]$ tar zxvf nagios-4.4.6.tar.gz
cd nagios-4.4.6
nagios-4.4.6/
nagios-4.4.6/.gitignore
nagios-4.4.6/.travis.yml
nagios-4.4.6/CONTRIBUTING.md
nagios-4.4.6/Changelog
nagios-4.4.6/INSTALLING
nagios-4.4.6/LEGAL
nagios-4.4.6/LICENSE
nagios-4.4.6/Makefile.in
nagios-4.4.6/README.md
nagios-4.4.6/THANKS
nagios-4.4.6/UPGRADING
nagios-4.4.6/aclocal.m4
nagios-4.4.6/autoconf-macros/
nagios-4.4.6/autoconf-macros/.gitignore
nagios-4.4.6/autoconf-macros/CHANGELOG.md
nagios-4.4.6/autoconf-macros/LICENSE
nagios-4.4.6/autoconf-macros/LICENSE.md
nagios-4.4.6/autoconf-macros/README.md
nagios-4.4.6/autoconf-macros/add_group_user
nagios-4.4.6/autoconf-macros/ax_nagios_get_distrib
nagios-4.4.6/autoconf-macros/ax_nagios_get_files
nagios-4.4.6/autoconf-macros/ax_nagios_get_inetd
nagios-4.4.6/autoconf-macros/ax_nagios_get_init
nagios-4.4.6/autoconf-macros/ax_nagios_get_os
nagios-4.4.6/autoconf-macros/ax_nagios_get_paths
```

11. Run the Configuration Script

Commands -

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



12. Compile the Source Code

Commands: - make all

```
[ec2-user@ip-172-31-37-11 nagios-4.4.6]$ make all
cd ./base && make
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.4.6/base'
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o nagios.o nagios.c
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE   -c -o broker.o broker.c
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE   -c -o nebmods.o nebmods.c
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o ../common/shared.o ../common/
shared.c
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE   -c -o query-handler.o query-handl
er.c
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o workers.o workers.c
In function 'get_wproc_list',
    inlined from 'get_worker' at workers.c:277:12:
workers.c:253:17: warning: '%s' directive argument is null [-Wformat-overflow=]
  253 |                 log_debug_info(DEBUGL_CHECKS, 1, "Found specialized worker
(s) for '%s'", (slash && *slash != '/') ? slash : cmd_name);
      |                 ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE   -c -o checks.o checks.c
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE   -c -o config.o config.c
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE   -c -o commands.o commands.c
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE   -c -o events.o events.c
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE   -c -o flapping.o flapping.c
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE   -c -o logging.o logging.c
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o macros-base.o ../common/macro
s.c
gcc -Wall -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE   -c -o netutils.o netutils.c
```

```
*** Support Notes *******************************************

If you have questions about configuring or running Nagios,
please make sure that you:

     - Look at the sample config files
     - Read the documentation on the Nagios Library at:
           https://library.nagios.com

before you post a question to one of the mailing lists.
Also make sure to include pertinent information that could
help others help you.  This might include:

     - What version of Nagios you are using
     - What version of the plugins you are using
     - Relevant snippets from your config files
     - Relevant error messages from the Nagios log file

For more information on obtaining support for Nagios, visit:

       https://support.nagios.com

*************************************************************

Enjoy.

[ec2-user@ip-172-31-37-11 nagios-4.4.6]$ 
```

13. Install Binaries, Init Script, and Sample Config Files

Commands -

./sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode

```
[ec2-user@ip-172-31-37-11 nagios-4.4.6]$ sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode
cd ./base && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.4.6/base'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagios /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagiostats /usr/local/nagios/bin
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.4.6/base'
cd ./cgi && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.4.6/cgi'
make install-basic
make[2]: Entering directory '/home/ec2-user/downloads/nagios-4.4.6/cgi'
```

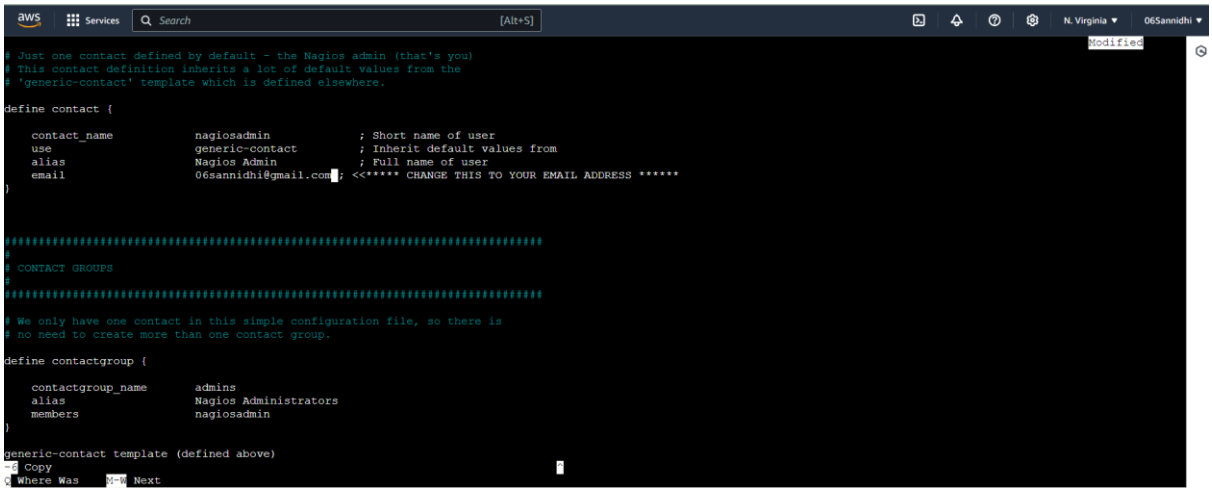
```
*** Config files installed ***

Remember, these are *SAMPLE* config files.  You'll need to read
the documentation for more information on how to actually define
services, hosts, etc. to fit your particular needs.

/usr/bin/install -c -m 775 -o nagios -g nagcmd -d /usr/local/nagios/var/rw
chmod g+s /usr/local/nagios/var/rw

*** External command directory configured ***

[ec2-user@ip-172-31-37-11 nagios-4.4.6]$
```

14. Edit the Config File to Change the Email Address

Commands -

sudo nano /usr/local/nagios/etc/objects/contacts.cfg

- Change the email address in the contacts.cfg file to your preferred email.



15. Configure the Web Interface

Commands -
sudo make install-webconf

```
[ec2-user@ip-172-31-37-11 nagios-4.4.6]$ sudo nano /usr/local/nagios/etc/objects/contacts.cfg
[ec2-user@ip-172-31-37-11 nagios-4.4.6]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios.conf
if [ 0 -eq 1 ]; then \
        ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagios.conf
; \
fi

*** Nagios/Apache conf file installed ***

[ec2-user@ip-172-31-37-11 nagios-4.4.6]$
```

16. Create a Nagios Admin Account

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

You will be prompted to enter and confirm the password for the nagiosadmin user.

```
[ec2-user@ip-172-31-37-11 nagios-4.4.6]$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
[ec2-user@ip-172-31-37-11 nagios-4.4.6]$
```

17. Restart Apache

Commands - systemctl restart httpd

```
[ec2-user@ip-172-31-37-11 nagios-4.4.6]$ sudo systemctl restart httpd
[ec2-user@ip-172-31-37-11 nagios-4.4.6]$
```

18. Extract the Plugins Source File

Commands -
cd ~/downloads
tar zxvf nagios-plugins-2.3.3.tar.gz
cd nagios-plugins-2.3.3

```
[ec2-user@ip-172-31-37-11 nagios-4.4.6]$ sudo systemctl restart httpd
[ec2-user@ip-172-31-37-11 nagios-4.4.6]$ cd ~/downloads
tar zxvf nagios-plugins-2.3.3.tar.gz
cd nagios-plugins-2.3.3
nagios-plugins-2.3.3/
nagios-plugins-2.3.3/perlmods/
nagios-plugins-2.3.3/perlmods/Config-Tiny-2.14.tar.gz
nagios-plugins-2.3.3/perlmods/parent-0.226.tar.gz
nagios-plugins-2.3.3/perlmods/Test-Simple-0.98.tar.gz
nagios-plugins-2.3.3/perlmods/Makefile.in
nagios-plugins-2.3.3/perlmods/version-0.9903.tar.gz
nagios-plugins-2.3.3/perlmods/Makefile.am
nagios-plugins-2.3.3/perlmods/Module-Runtime-0.013.tar.gz
nagios-plugins-2.3.3/perlmods/Module-Metadata-1.000014.tar.gz
nagios-plugins-2.3.3/perlmods/Params-Validate-1.08.tar.gz
nagios-plugins-2.3.3/perlmods/Class-Accessor-0.34.tar.gz
nagios-plugins-2.3.3/perlmods/Try-Tiny-0.18.tar.gz
nagios-plugins-2.3.3/perlmods/Module-Implementation-0.07.tar.gz
nagios-plugins-2.3.3/perlmods/Makefile
nagios-plugins-2.3.3/perlmods/Perl-OSType-1.003.tar.gz
nagios-plugins-2.3.3/perlmods/install_order
nagios-plugins-2.3.3/perlmods/Nagios-Plugin-0.36.tar.gz
nagios-plugins-2.3.3/perlmods/Math-Calc-Units-1.07.tar.gz
nagios-plugins-2.3.3/perlmods/Module-Build-0.4007.tar.gz
nagios-plugins-2.3.3/ABOUT-NLS
nagios-plugins-2.3.3/configure.ac
nagios-plugins-2.3.3/Makefile.in
nagios-plugins-2.3.3/config.h.in
nagios-plugins-2.3.3/ChangeLog
nagios-plugins-2.3.3/AUTHORS
nagios-plugins-2.3.3/lib/
nagios-plugins-2.3.3/lib/parse_ini.h
```

19. Compile and Install Plugins

Commands -

./configure --with-nagios-user=nagios --with-nagios-group=nagios
make
sudo make install

```
[ec2-user@ip-172-31-37-11 nagios-plugins-2.3.3]$ ./configure --with-nagios-user=na
gios --with-nagios-group=nagios                ./configure --with-nagios-user=na
gios --with-nagios-group=nagios
make make install
sudo make install
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 to disable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-unknown-linux-gnu
checking host system type... x86_64-unknown-linux-gnu
checking for gcc... gcc
checking for C compiler default output file name... a.out
checking whether the C compiler works... yes
checking whether we are cross compiling... no
checking for suffix of executables...
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 for style of include used by make... GNU
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
```

20. Start Nagios

Commands -

sudo chkconfig --add nagios
sudo chkconfig nagios on
sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
sudo systemctl start nagios

```
[ec2-user@ip-172-31-37-11 nagios-plugins-2.3.3]$ sudo chkconfig --add nagios
sudo chkconfig nagios on
sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
sudo systemctl start nagios
error reading information on service nagios: No such file or directory
Note: Forwarding request to 'systemctl enable nagios.service'.
Created symlink /etc/systemd/system/multi-user.target.wants/nagios.service → /usr/lib/systemd/system/nagios.service.

Nagios Core 4.4.6
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2020-04-28
License: GPL

Website: https://www.nagios.org
Reading configuration data...
   Read main config file okay...
   Read object config files okay...

Running pre-flight check on configuration data...

Checking objects...
       Checked 8 services.
       Checked 1 hosts.
       Checked 1 host groups.
       Checked 0 service groups.
       Checked 1 contacts.
       Checked 1 contact groups.
       Checked 24 commands.
       Checked 5 time periods.
       Checked 0 host escalations.
       Checked 0 service escalations.
Checking for circular paths...
```

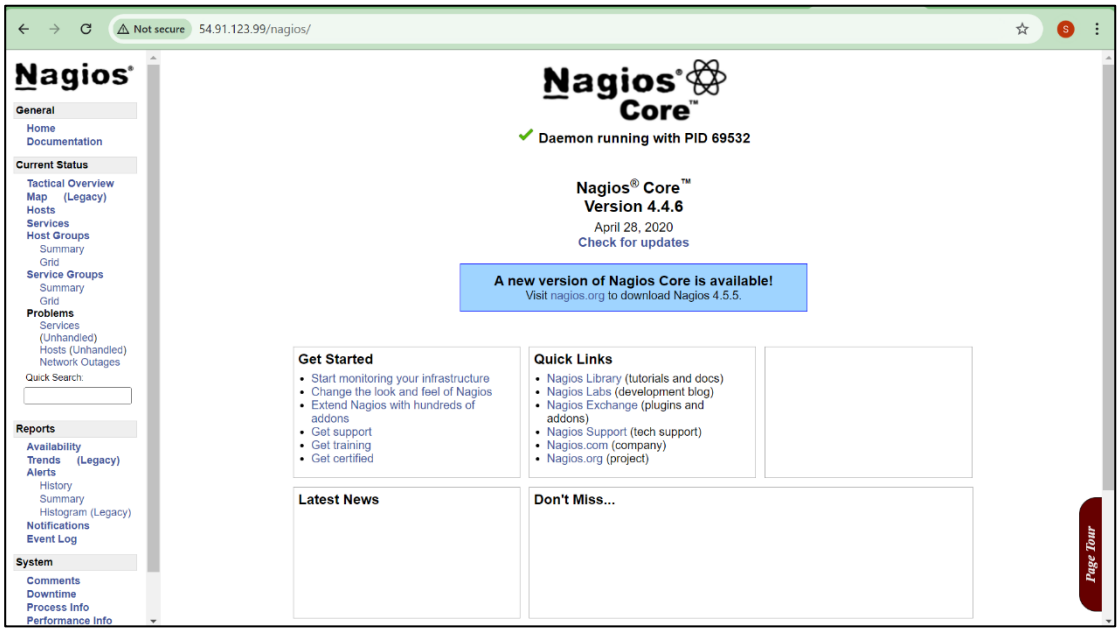
21. Check the Status of Nagios

Commands -
sudo systemctl status nagios

```
Things look okay - No serious problems were detected during the pre-flight check
[ec2-user@ip-172-31-37-11 nagios-plugins-2.3.3]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.4.6
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Tue 2024-10-08 14:14:32 UTC; 1min 33s ago
       Docs: https://www.nagios.org/documentation
    Process: 69530 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 69531 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 69532 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 2.3M
        CPU: 30ms
     CGroup: /system.slice/nagios.service
             ├─69532 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─69533 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─69534 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─69535 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─69536 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─69537 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 08 14:14:32 ip-172-31-37-11.ec2.internal nagios[69532]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully initialized
Oct 08 14:14:32 ip-172-31-37-11.ec2.internal nagios[69532]: qh: core query handler registered
Oct 08 14:14:32 ip-172-31-37-11.ec2.internal nagios[69532]: qh: echo service query handler registered
Oct 08 14:14:32 ip-172-31-37-11.ec2.internal nagios[69532]: qh: help for the query handler registered
Oct 08 14:14:32 ip-172-31-37-11.ec2.internal nagios[69532]: wproc: Successfully registered manager as @wproc with query handler
Oct 08 14:14:32 ip-172-31-37-11.ec2.internal nagios[69532]: wproc: Registry request: name=Core Worker 69536;pid=69536
Oct 08 14:14:32 ip-172-31-37-11.ec2.internal nagios[69532]: wproc: Registry request: name=Core Worker 69535;pid=69535
Oct 08 14:14:32 ip-172-31-37-11.ec2.internal nagios[69532]: wproc: Registry request: name=Core Worker 69534;pid=69534
Oct 08 14:14:32 ip-172-31-37-11.ec2.internal nagios[69532]: wproc: Registry request: name=Core Worker 69533;pid=69533
Oct 08 14:14:32 ip-172-31-37-11.ec2.internal nagios[69532]: Successfully launched command file worker with pid 69537
[ec2-user@ip-172-31-37-11 nagios-plugins-2.3.3]$
```

22. Access Nagios Web Interface

- Copy the Public IP address of your EC2 instance.
- Open your browser and navigate to http://<your_public_ip_address>/nagios.
- Enter the username nagiosadmin and the password you set in Step 16.



Conclusion:

After installing and configuring Nagios Core, Plugins, and NRPE on a Linux machine, We have a robust continuous monitoring setup, ensuring proactive issue detection and optimal system performance.

EXPERIMENT NO. 10

Aim:

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

Implementation:

Prerequisites

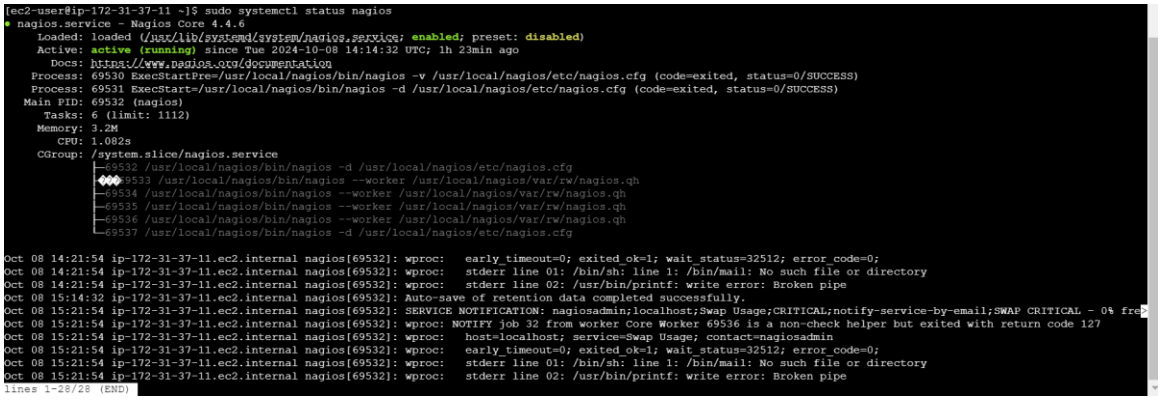
- AWS Free Tier
- Nagios Server running on an Amazon Linux Machine

1. Confirm Nagios is Running on the Server

Commands -

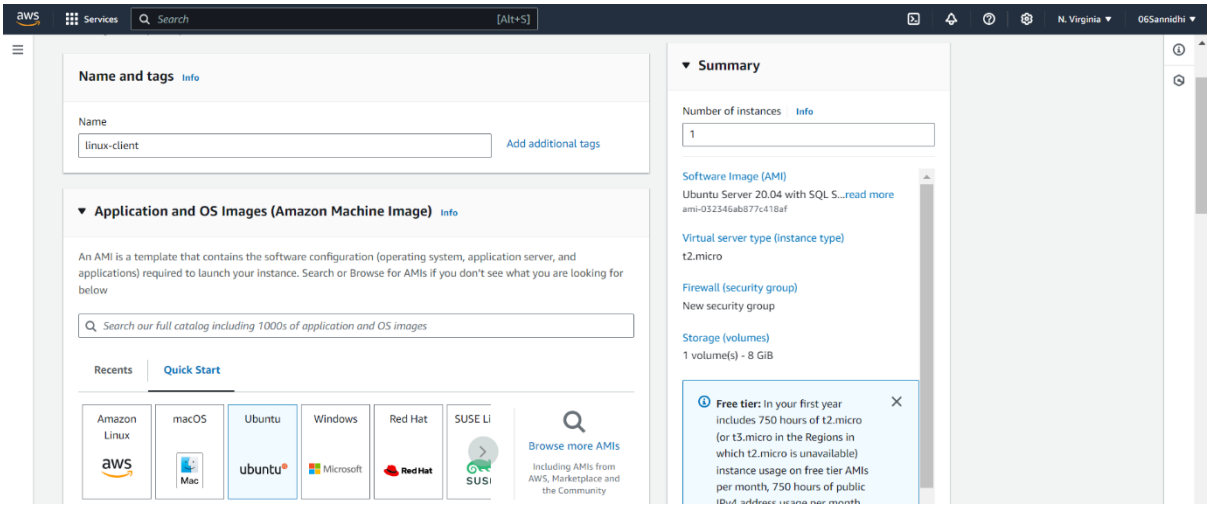
sudo systemctl status nagios

- Proceed if you see that Nagios is active and running.



2. Create an Ubuntu 20.04 Server EC2 Instance

- Name it linux-client.
- Use the same security group as the Nagios Host.



3. Verify Nagios Process on the Server

Commands -
ps -ef | grep nagios

```
[ec2-user@ip-172-31-37-11 ~]$ ps -ef | grep nagios
nagios      69532       1  0 14:14 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios      69533   69532  0 14:14 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      69534   69532  0 14:14 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      69535   69532  0 14:14 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      69536   69532  0 14:14 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      69537   69532  0 14:14 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
ec2-user    75736   74742  0 15:55 pts/0    00:00:00 grep --color=auto nagios
[ec2-user@ip-172-31-37-11 ~]$
```

i-05899dd5e5b12359c (nagios-host)

4. Become Root User and Create Directories

Commands -
sudo su
mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts

```
[ec2-user@ip-172-31-37-11 ~]$ sudo su
[root@ip-172-31-37-11 ec2-user]# mkdir /usr/local/nagios/etc/objects/monitorhosts
[root@ip-172-31-37-11 ec2-user]# mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
sts
[root@ip-172-31-37-11 ec2-user]#
```

5. Copy Sample Configuration File

Commands -
cp /usr/local/nagios/etc/objects/localhost.cfg
/usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

```
[root@ip-172-31-37-11 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
[root@ip-172-31-37-11 ec2-user]# sudo nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
```

6. Edit the Configuration File

Commands -
sudo nano
/usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

- Change hostname to linuxserver everywhere in the file.
- Change address to the public IP address of your linux-client.

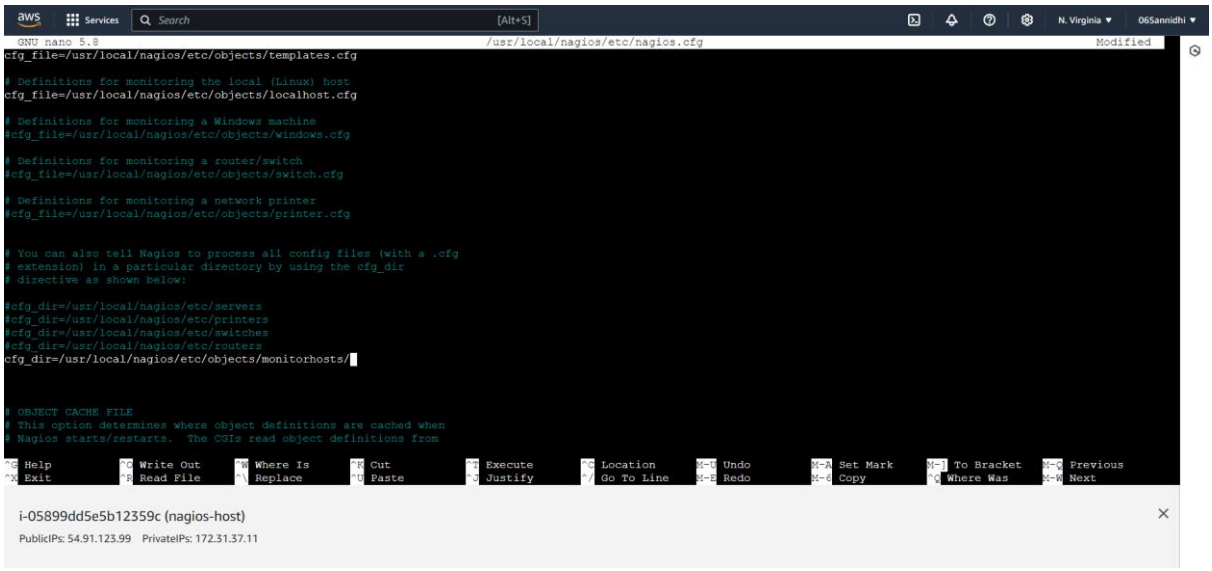
```
# Define a host for the local machine

define host {

    use                     linux-server            ; Name of host template to use
                                                    ; This host definition will inherit all variables that are defined
                                                    ; in (or inherited by) the linux-server host template definition.
    host_name               linuxserver
    alias                   linuxserver
    address                 54.88.143.206
}
```

● Change hostgroup_name under hostgroup to linux-servers1.

```
##############################################################################
#
# HOST GROUP DEFINITION
#
##############################################################################

# Define an optional hostgroup for Linux machines

define hostgroup {

    hostgroup_name          linux-servers           ; The name of the hostgroup
    alias                   Linux Servers           ; Long name of the group
    members                 linuxserver             ; Comma separated list of hosts that belong to this group
}



##############################################################################
```

7. Update Nagios Configuration

Commands -
sudo nano /usr/local/nagios/etc/nagios.cfg
● Add the following line:
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/



8. Verify Configuration Files

Commands -
sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
● Ensure there are no errors.

```
Running pre-flight check on configuration data...

Checking objects...
       Checked 8 services.
       Checked 2 hosts.
       Checked 2 host groups.
       Checked 0 service groups.
       Checked 1 contacts.
       Checked 1 contact groups.
       Checked 24 commands.
       Checked 5 time periods.
       Checked 0 host escalations.
       Checked 0 service escalations.
Checking for circular paths...
       Checked 2 hosts
       Checked 0 service dependencies
       Checked 0 host dependencies
       Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors:   0

Things look okay - No serious problems were detected during the pre-flight check
[root@ip-172-31-37-11 ec2-user]#
```

9. Restart Nagios Service

    Commands -
    sudo systemctl restart nagios

10. SSH into the Client Machine

    - Use SSH or EC2 Instance Connect to access the linux-client.

11. Update Package Index and Install Required Packages

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

```
ubuntu@ip-172-31-46-201:~$ sudo apt update -y
sudo apt install gcc -y
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 [12
6 kB]
0% [3 InRelease 102 kB/126 kB 81%] [Connecting to security.ubuntu.com (185.125.190

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://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [382 kB
]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en
 [5982 kB]
57% [5 Packages store 0 B] [7 Translation-en 0 B/5982 kB 0%] [6 Packages 107 kB/38

Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Componen
ts [3871 kB]
75% [5 Packages store 0 B] [8 Components-amd64 140 kB/3871 kB 4%] [6 Packages 180

Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Me
tadata [301 kB]

86% [5 Packages store 0 B] [9 Commands-amd64 152 kB/301 kB 50%] [6 Packages 205 kB

Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packa
ges [269 kB]
87% [5 Packages store 0 B] [10 Packages 161 kB/269 kB 60%] [6 Packages 208 kB/382
```

12. Edit NRPE Configuration File

Commands -
sudo nano /etc/nagios/nrpe.cfg
● Add your Nagios host IP address under allowed_hosts:
allowed_hosts=<Nagios_Host_IP>

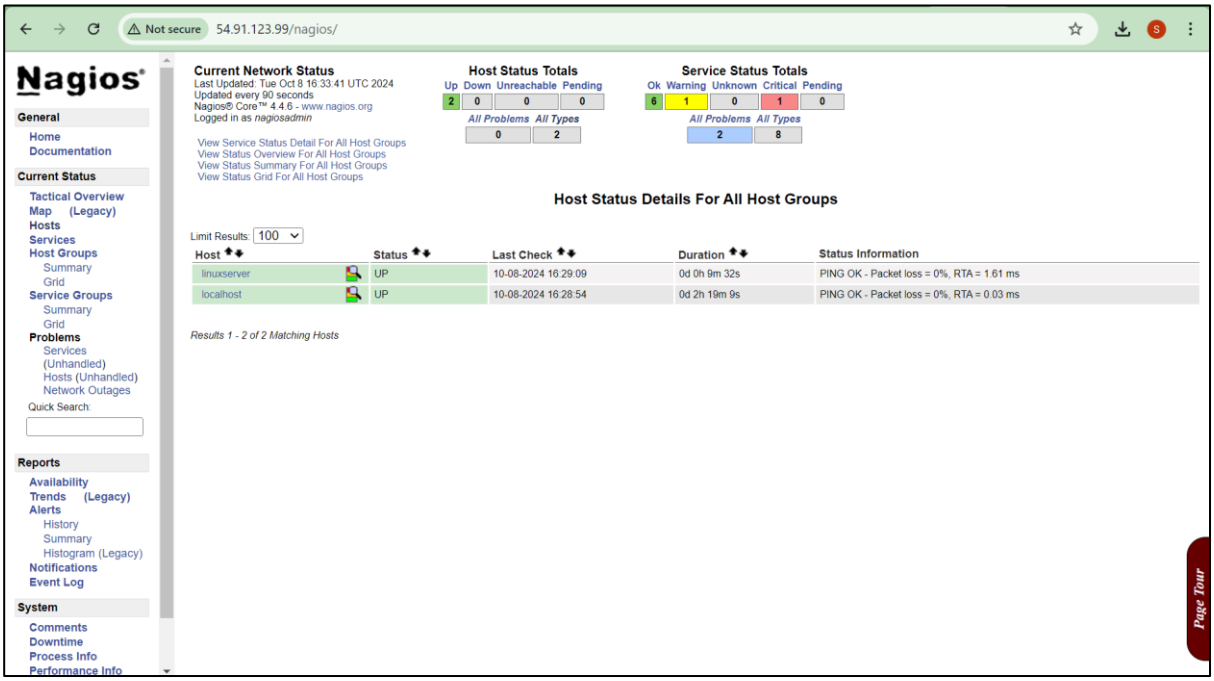
```
ubuntu@ip-172-31-46-201:~$ sudo apt install -y nagios-nrpe-server nagios-plugins

Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'monitoring-plugins' instead of 'nagios-plugins'
The following additional packages will be installed:
  libavahi-client3 libavahi-common-data libavahi-common3 libcups2t64 libdbi1t64 libldb2 libmysqlclient21 libnet-snmp-perl libpq5 libradcli4 libsmbclient0 libsnmp-base
  libsnmp40t64 libtalloc2 libtdb1 libtevent0t64 liburiparser1 libwbclient0 monitoring-plugins-basic monitoring-plugins-common monitoring-plugins-standard mysql-common
  python3-gpg python3-ldb python3-markdown python3-samba python3-talloc python3-tdb rpcbind samba-common samba-common-bin samba-dsdb-modules samba-libs smbclient snmp
Suggested packages:
  cups-common libcrypt-des-perl libdigest-hmac-perl libio-socket-inet6-perl snmp-mibs-downloader icinga2 nagios-plugins-contrib fping postfix | sendmail-bin
  | exim4-daemon-heavy | exim4-daemon-light qstat xinetd | inetd python-markdown-doc heimdal-clients python3-dnspython cifs-utils
The following NEW packages will be installed:
  libavahi-client3 libavahi-common-data libavahi-common3 libcups2t64 libdbi1t64 libldb2 libmysqlclient21 libnet-snmp-perl libpq5 libradcli4 libsmbclient0 libsnmp-base
  libsnmp40t64 libtalloc2 libtdb1 libtevent0t64 liburiparser1 libwbclient0 monitoring-plugins monitoring-plugins-basic monitoring-plugins-common
  monitoring-plugins-standard mysql-common nagios-nrpe-server python3-gpg python3-ldb python3-markdown python3-samba python3-talloc python3-tdb rpcbind samba-common
  samba-common-bin samba-dsdb-modules samba-libs smbclient snmp
0 upgraded, 37 newly installed, 0 to remove and 6 not upgraded.
Need to get 16.1 MB of archives.
After this operation, 72.0 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 nagios-nrpe-server amd64 4.1.0-1ubuntu3 [356 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 rpcbind amd64 1.2.6-7ubuntu2 [46.5 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libavahi-common-data amd64 0.8-13ubuntu6 [29.7 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libavahi-common3 amd64 0.8-13ubuntu6 [23.3 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libavahi-client3 amd64 0.8-13ubuntu6 [26.8 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 libcups2t64 amd64 2.4.7-1.2ubuntu7.3 [272 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libdbi1t64 amd64 0.9.0-6.1build1 [25.7 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libtalloc2 amd64 2.4.2-1build2 [27.3 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libtdb1 amd64 1.4.10-1build1 [46.8 kB]
```

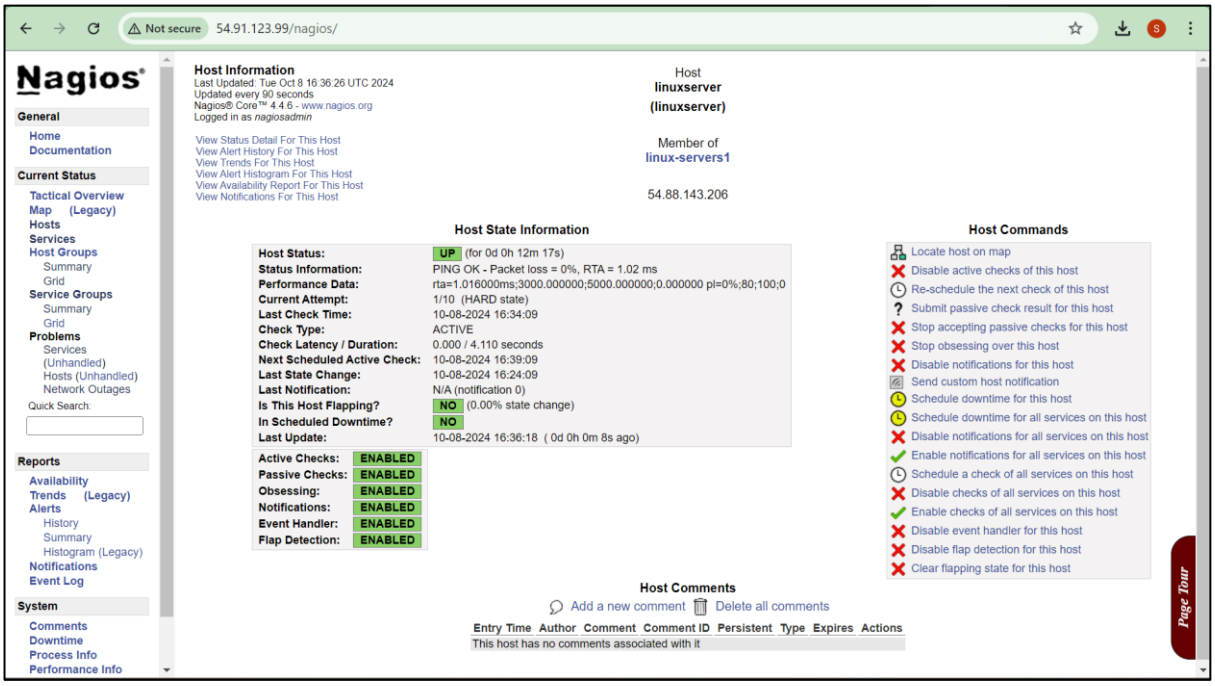
13. Restart NRPE Server Commands –

sudo systemctl restart nagios-nrpe-server

14. Check Nagios Dashboard

● Open your browser and navigate to http://<Nagios_Host_IP>/nagios.
● Log in with nagiosadmin and the password you set earlier.
● You should see the new host linuxserver added.
● Click on Hosts to see the host details.
● Click on Services to see all services and ports being monitored





Conclusion:

To perform port, service, and Windows/Linux server monitoring using Nagios, configure the necessary plugins and agents, define the monitoring parameters in the configuration files, and set up alerting mechanisms to ensure timely notifications of any issues. This comprehensive approach ensures robust monitoring and quick response to potential problems, maintaining the health and performance of your IT infrastructure.

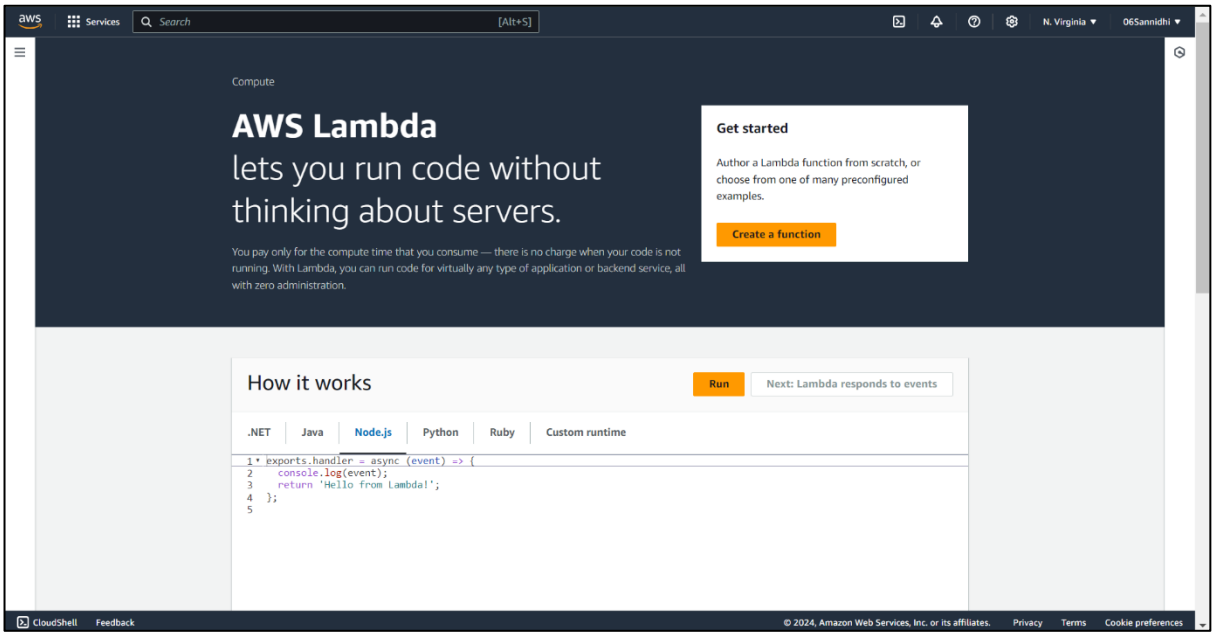
EXPERIMENT NO. 11

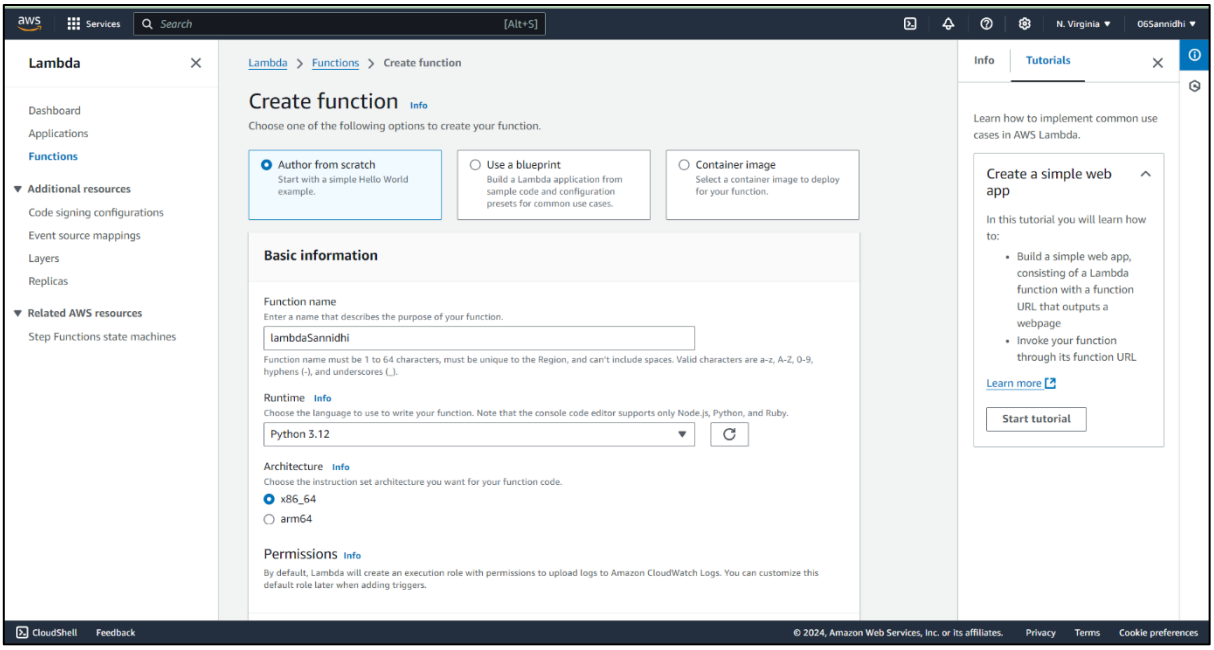
Aim: To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

Implementation:

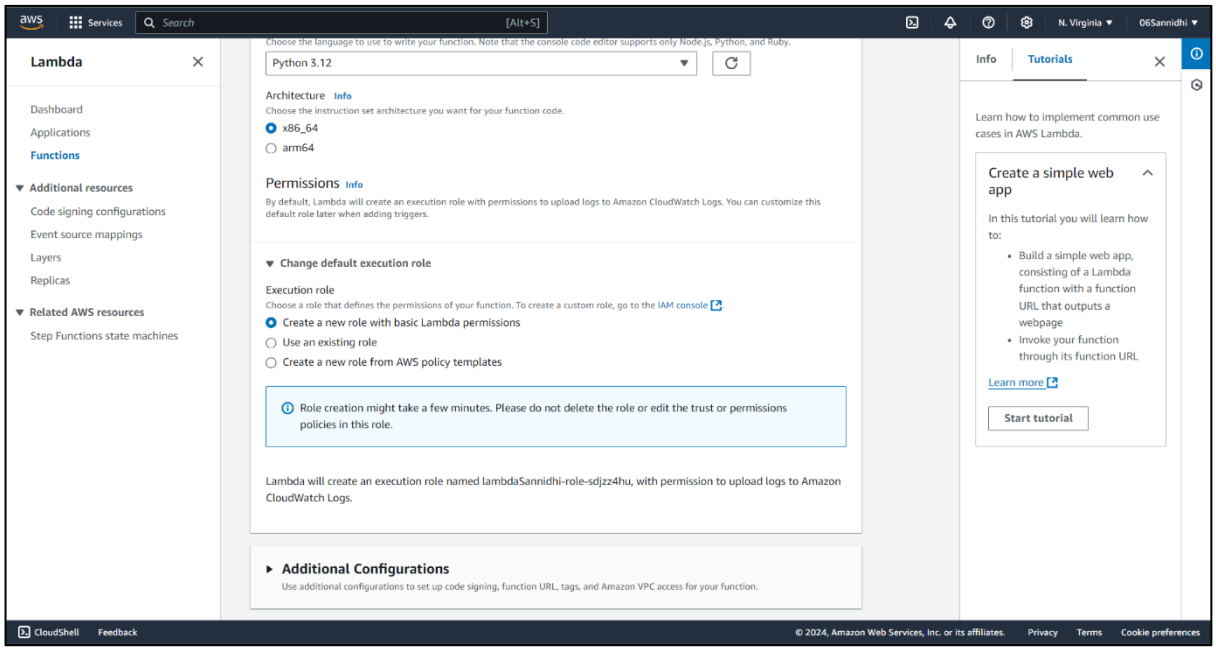
Steps to create an AWS Lambda function

1. Open up the Lambda Console and click on the Create button. Be mindful of where you create your functions since Lambda is region-dependent.

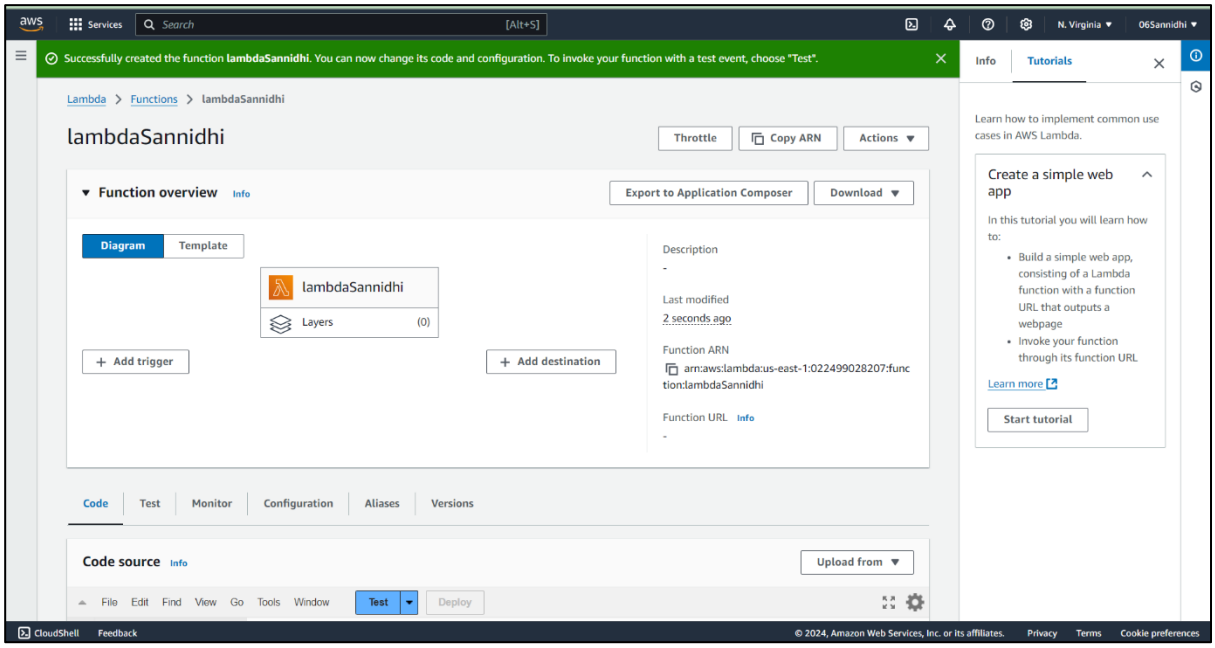




2. Choose to create a function from scratch or use a blueprint, i.e templates defined by AWS for you with all configuration presets required for the most common use cases.
   Then, choose a runtime env for your function, under the dropdown, you can see all the options AWS supports, Python, Nodejs, .NET and Java being the most popular ones. After that, choose to create a new role with basic Lambda permissions if you don't have an existing one.

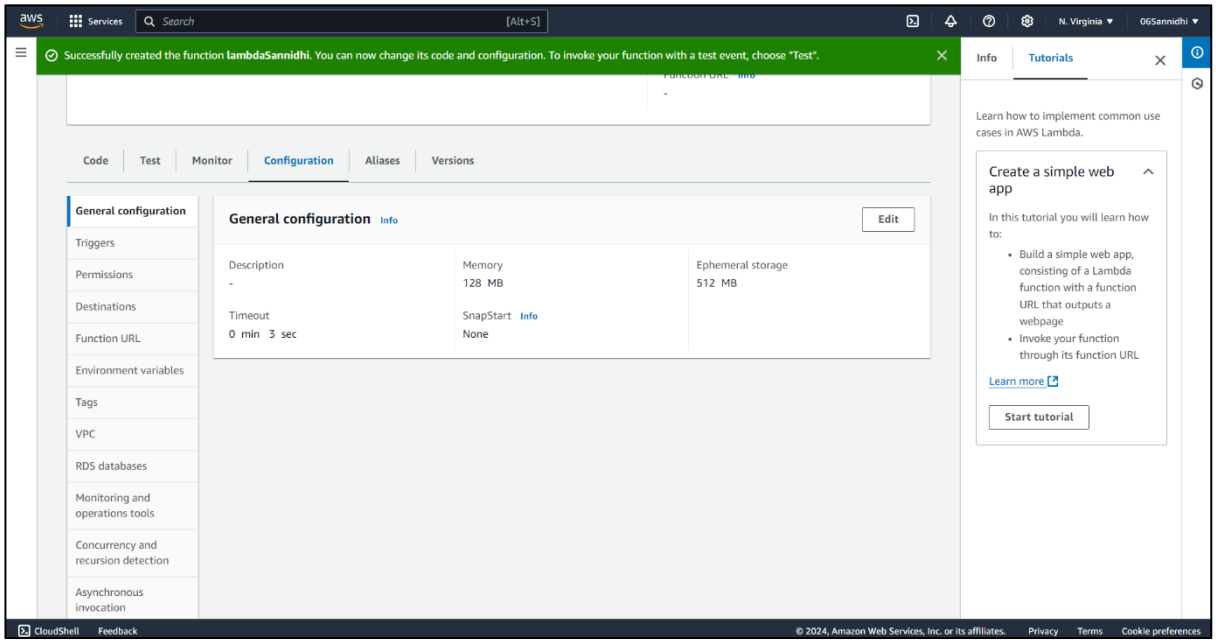


3. This process will take a while to finish and after that, you'll get a message that your function was successfully created.

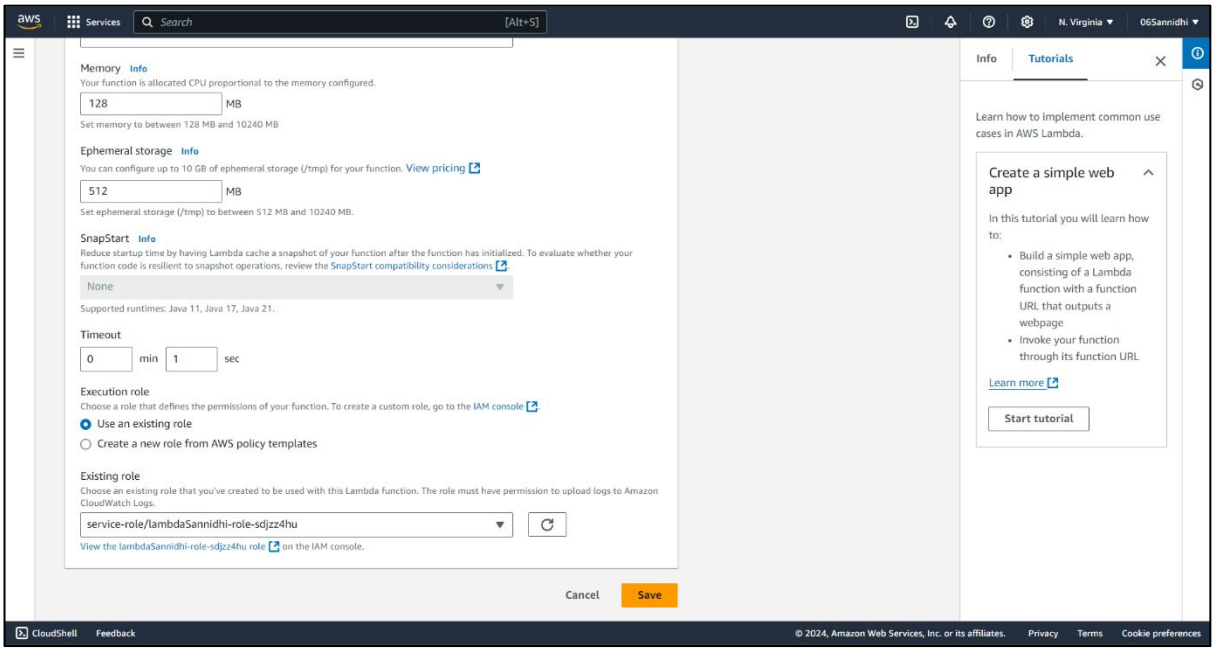


4. Edit Basic Settings:

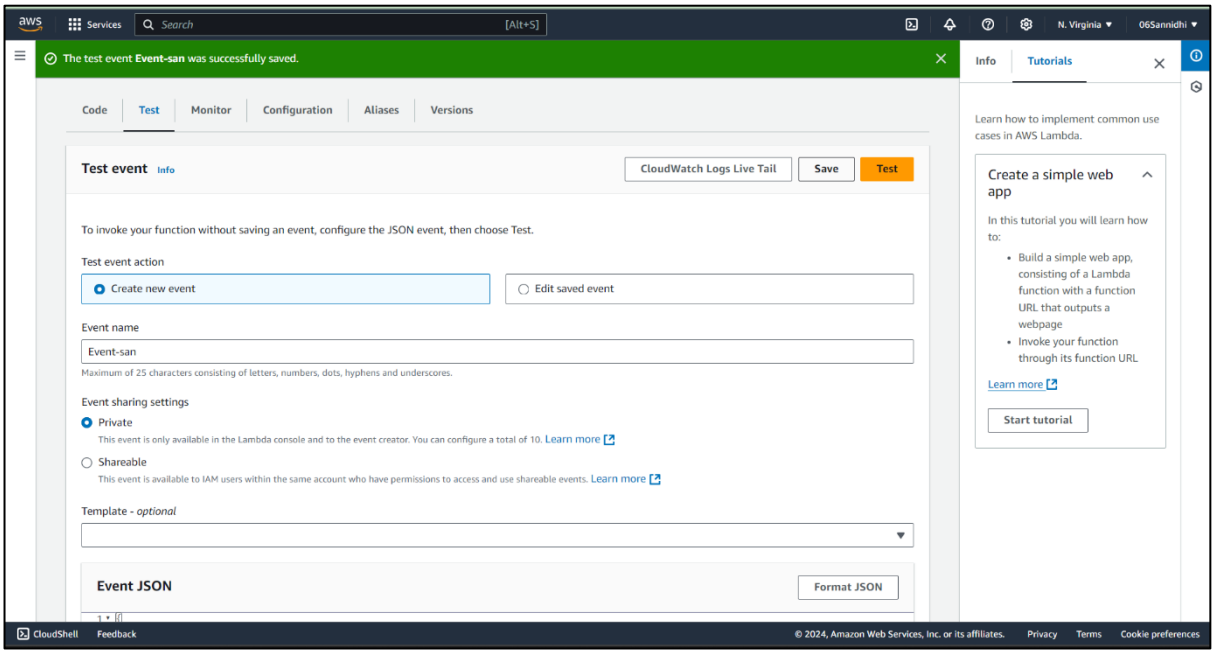
On the function's Configuration tab, locate the Basic settings section



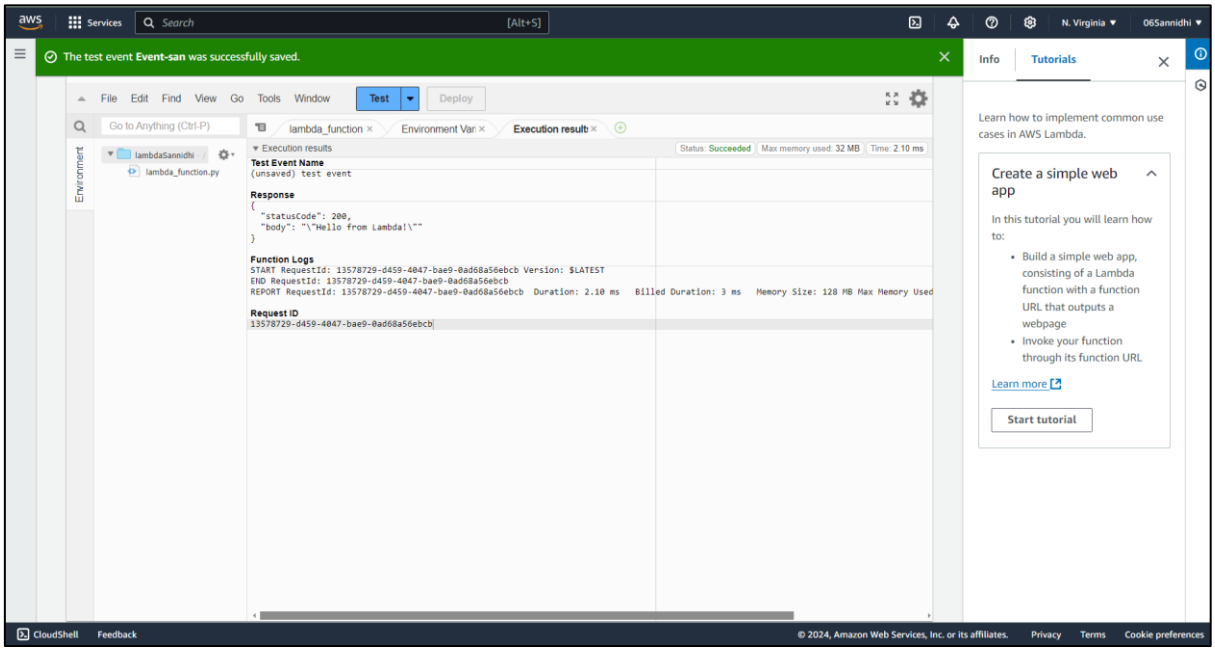
5. Enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.



6. Configuring test event which tiggers when the function is tested



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



Conclusion:

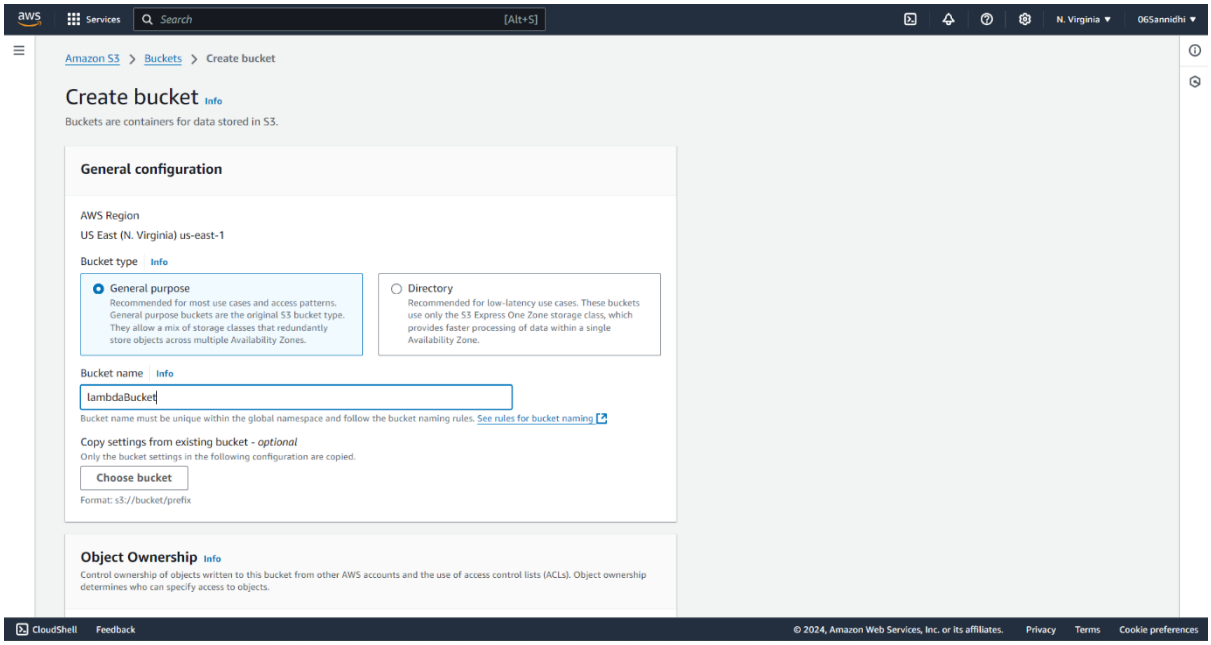
AWS Lambda is a serverless computing service that allows you to run code without managing servers, making it highly scalable, cost-effective, and easy to use. It automatically manages the compute resources, executes your code in response to specific events such as API calls, file uploads, or database updates, and scales based on the demand.

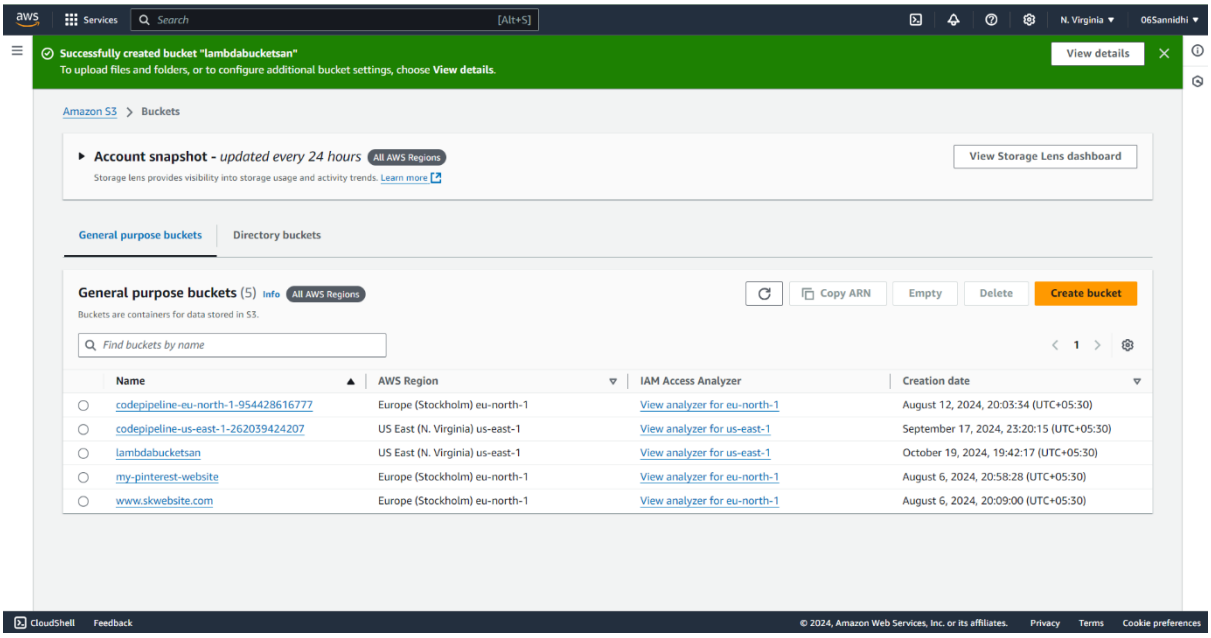
EXPERIMENT NO. 12

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.

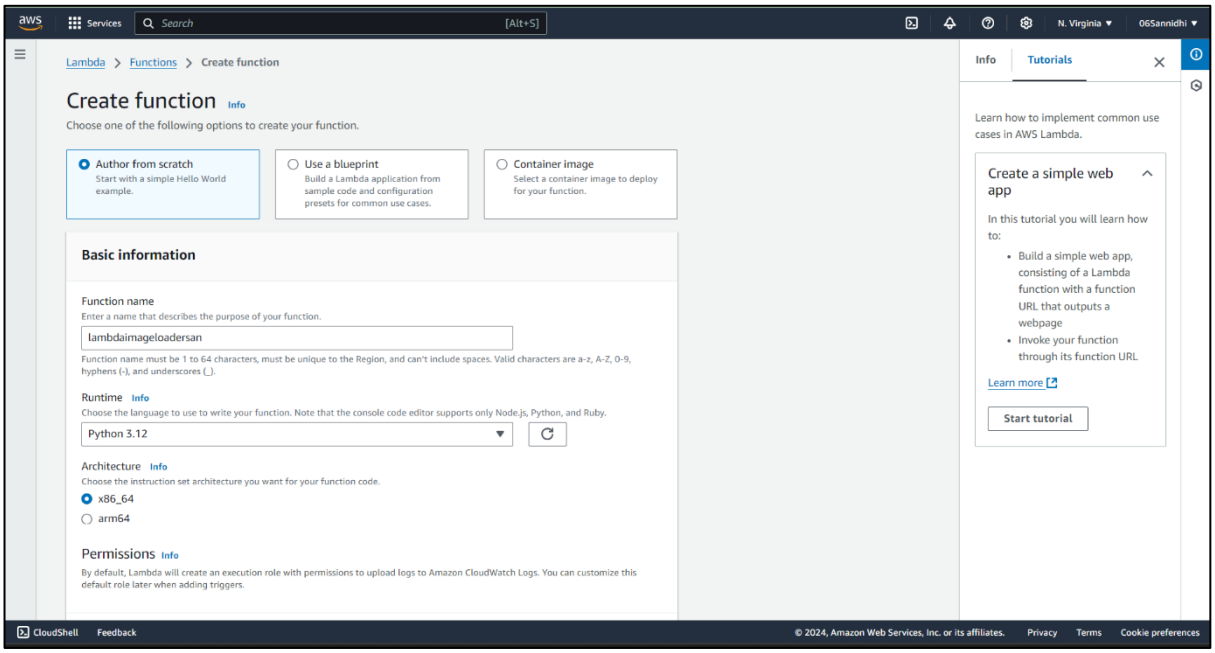
Implementation:

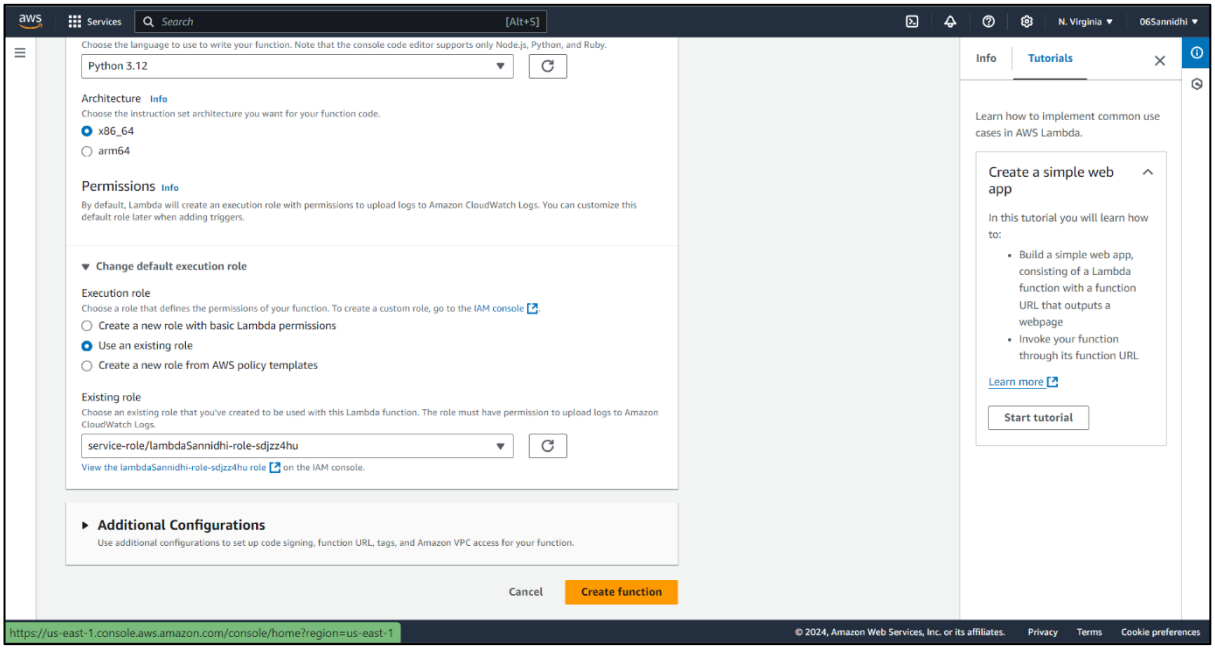
1. Create an S3 Bucket: First, create an S3 bucket that will store the objects. This bucket will act as the trigger source for the Lambda function.



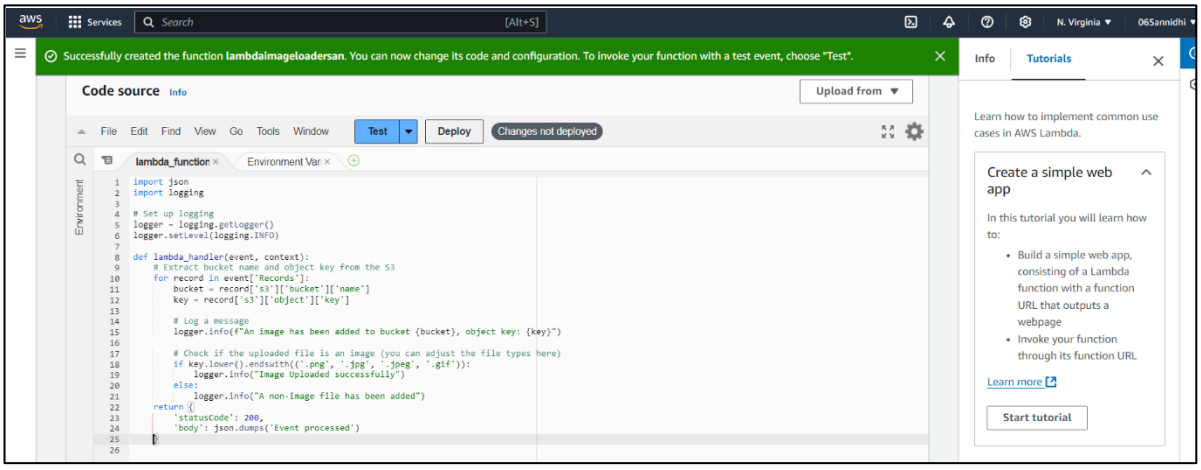


2. Create the Lambda Function: Set up a new Lambda function using AWS Lambda's console. You can choose a runtime environment like Python, Node.js, or Java. Write code that logs a message like "An Image has been added" when triggered.

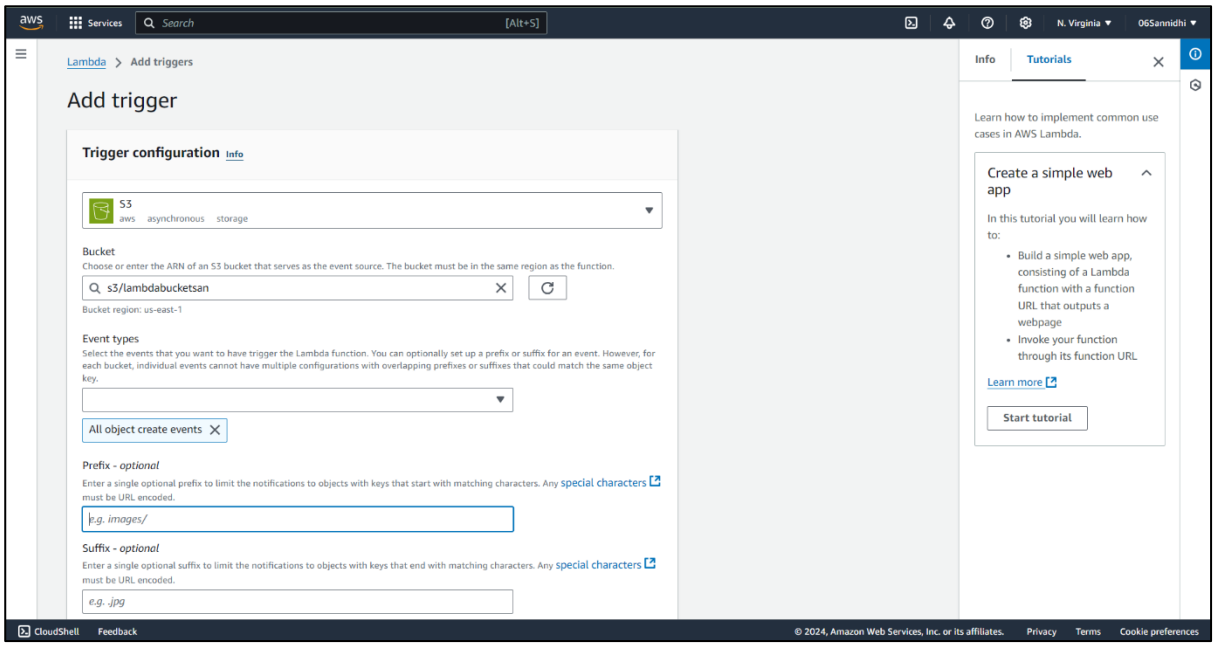


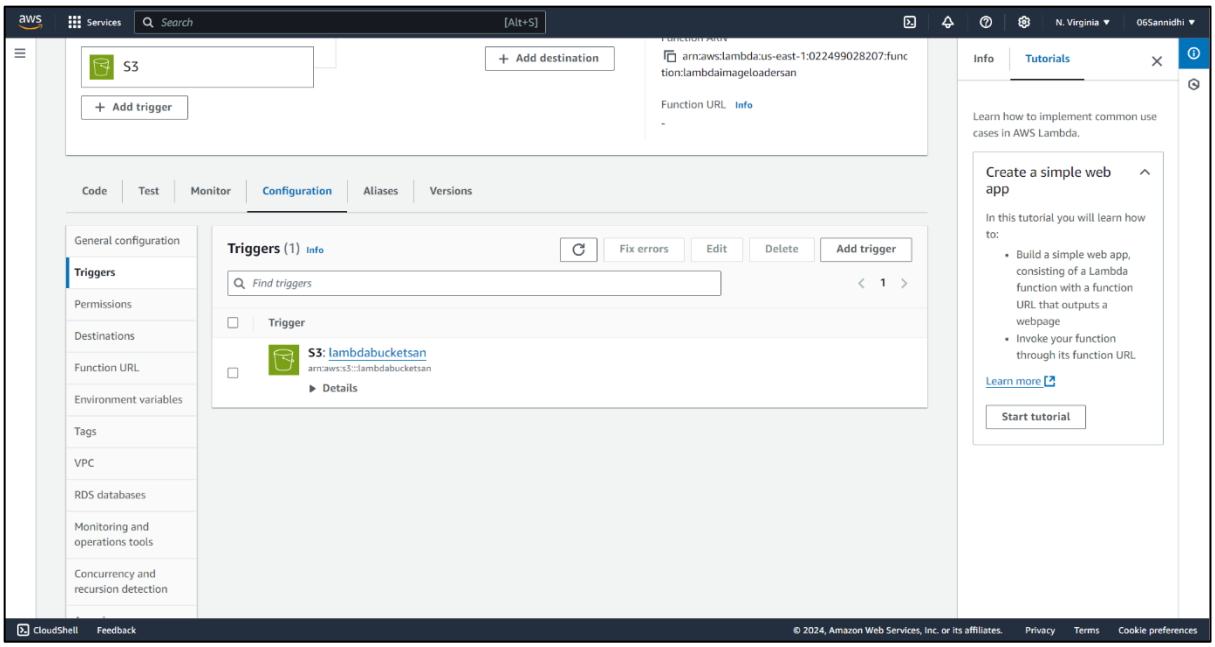


```
import json
import logging

# Set up logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    # Extract bucket name and object key
    from the S3 event for record in
    event['Records']:
        bucket =
        record['s3']['bucket']['name']
        key =
        record['s3']['object']['key']

        # Log a message
        logger.info(f"An image has been added to bucket {bucket}, object key: {key}")

        # Check if the uploaded file is an image (you can adjust the
        file types here) if key.lower().endswith(('.png', '.jpg', '.jpeg',
        '.gif')):
            logger.info("Image
        Uploaded successfully")
        else:
            logger.info("A non-image file has been added")

        return {
            'statusCode': 200,
            'body': json.dumps('Event processed')
        }
```

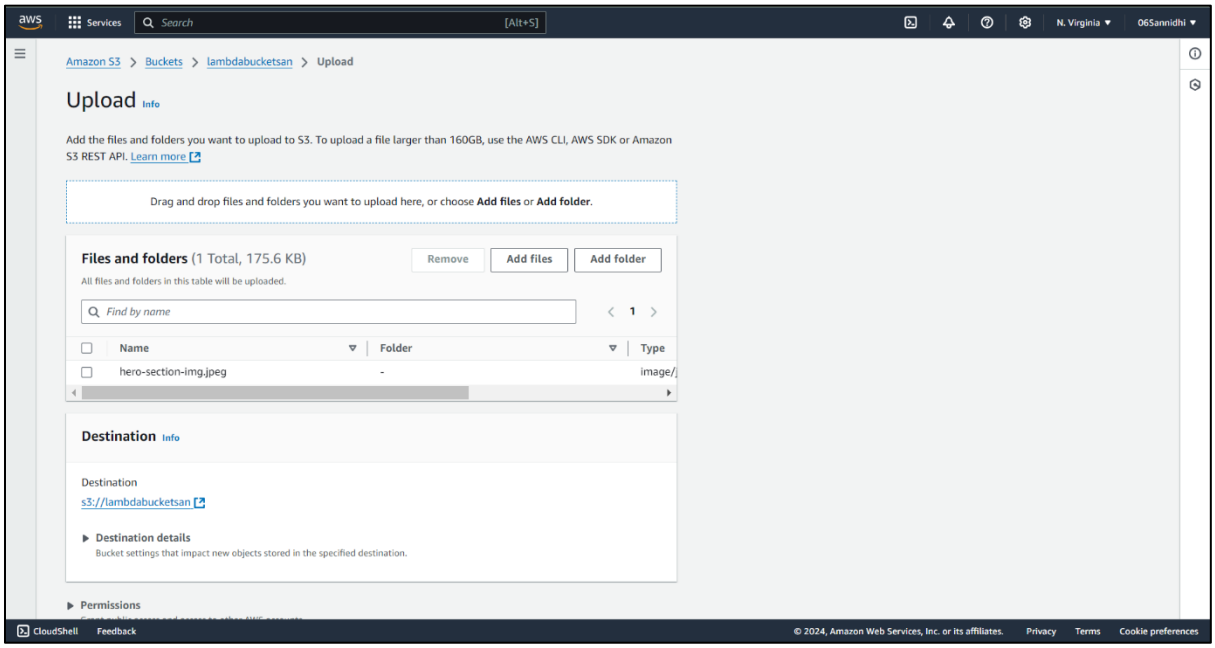


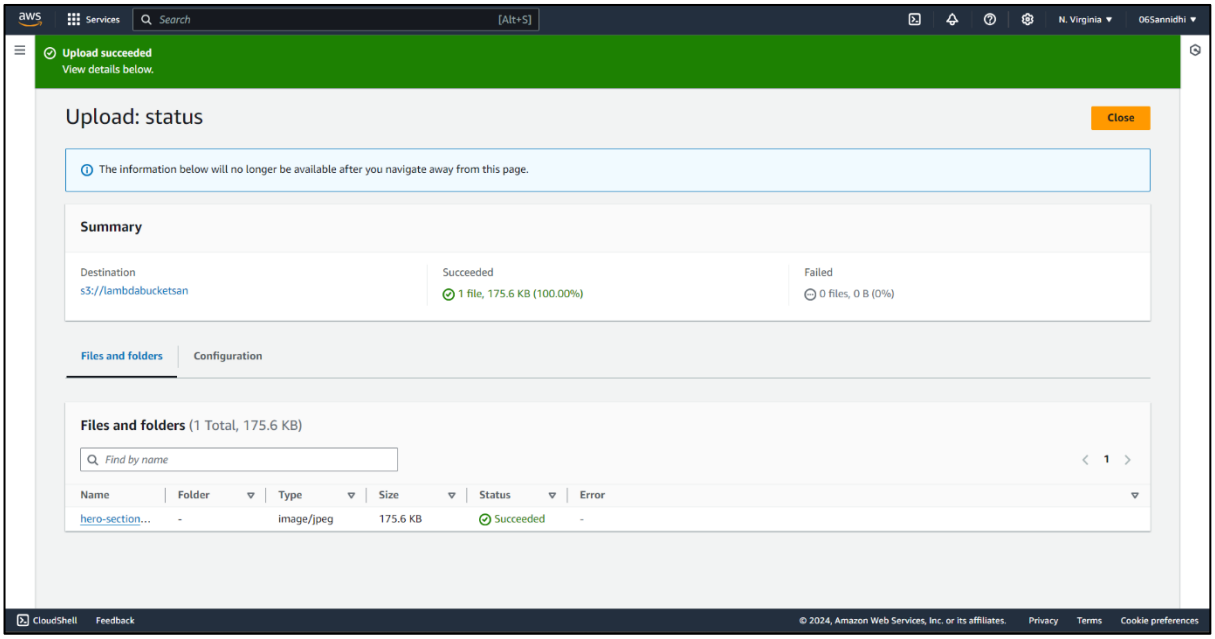
3. Configure S3 Trigger: Link the S3 bucket to the Lambda function by setting up a trigger. Specify that the function should be triggered when an object is created in the bucket (e.g., when an image is uploaded).



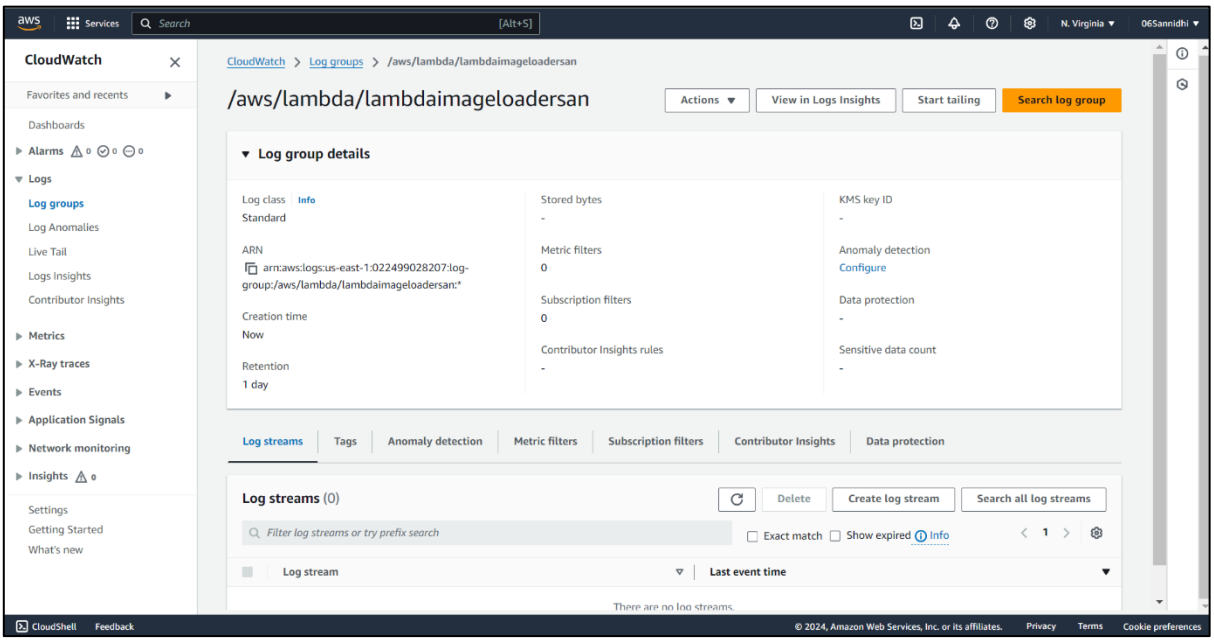


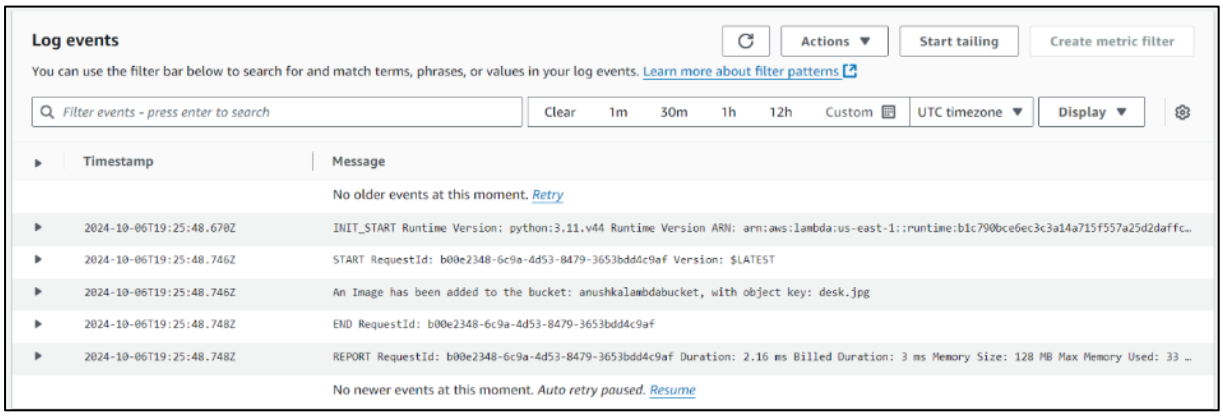
4. Upload an object (e.g., an image) to the S3 bucket to test the trigger





5. Test the Setup: Upload an object (e.g., an image) to the S3 bucket to test the trigger. The Lambda function should execute and log the message “An Image has been added” in AWS CloudWatch Logs





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 environment.