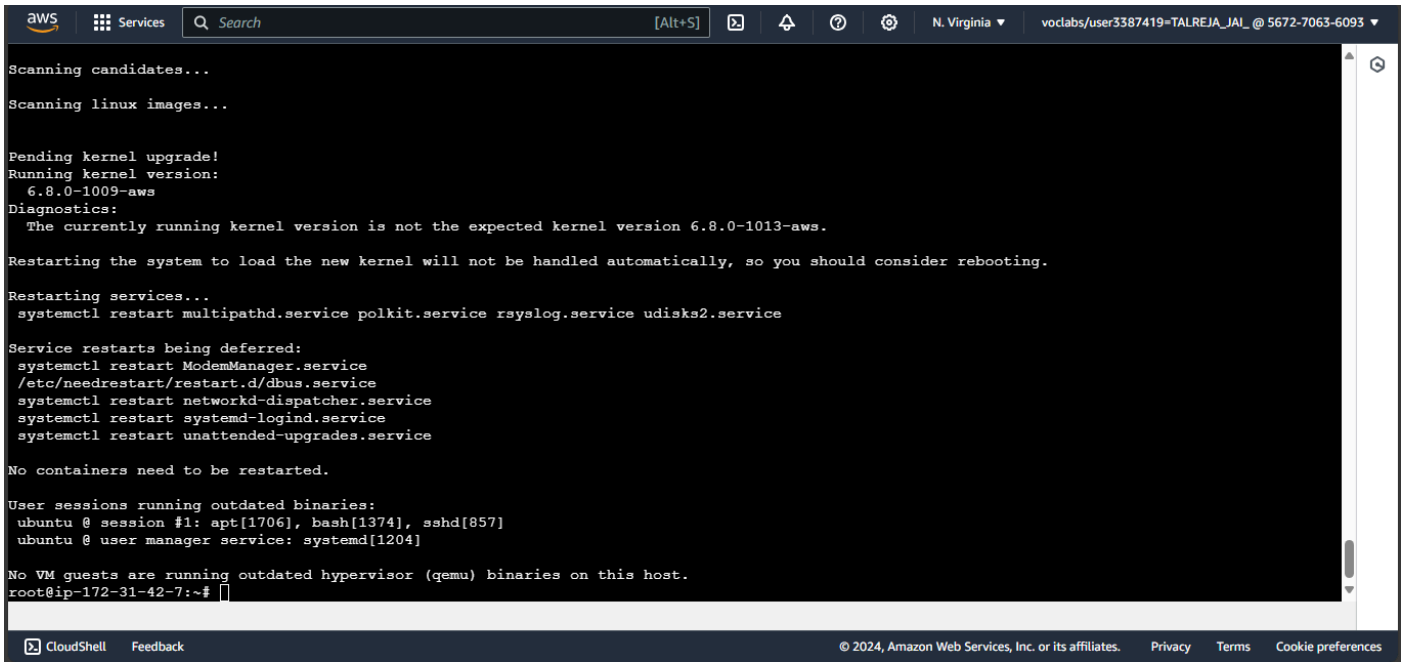
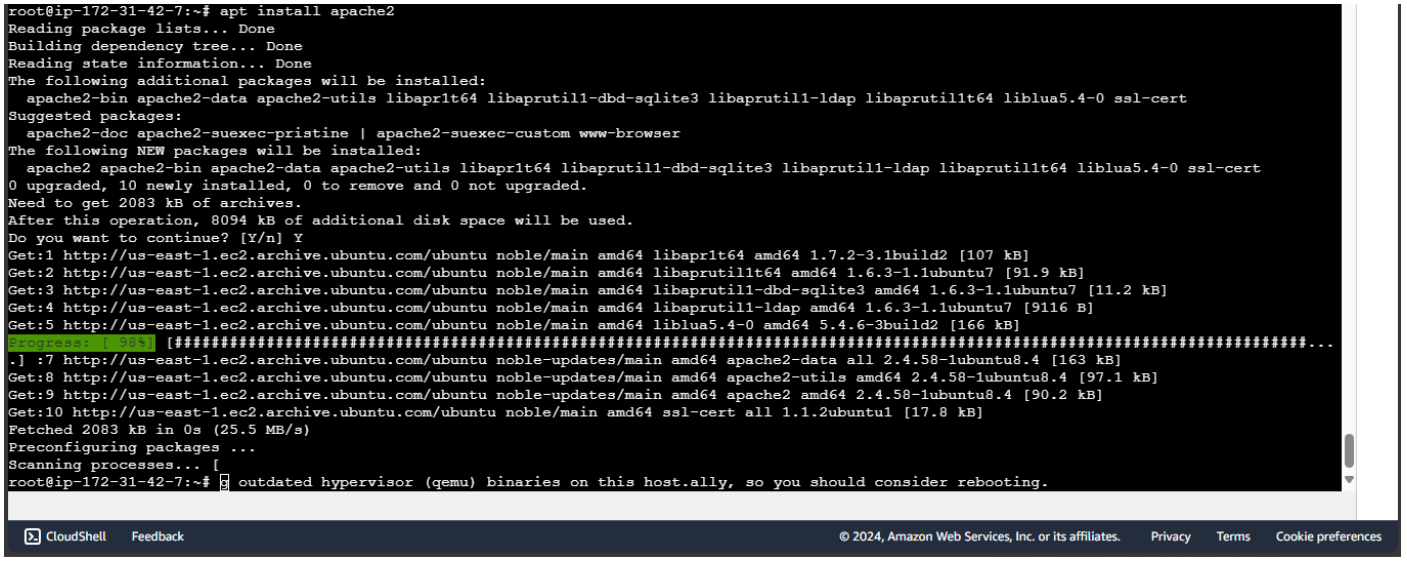
Advance Devops
Hitesh Rohra
D15A - 47

Experiment No 1

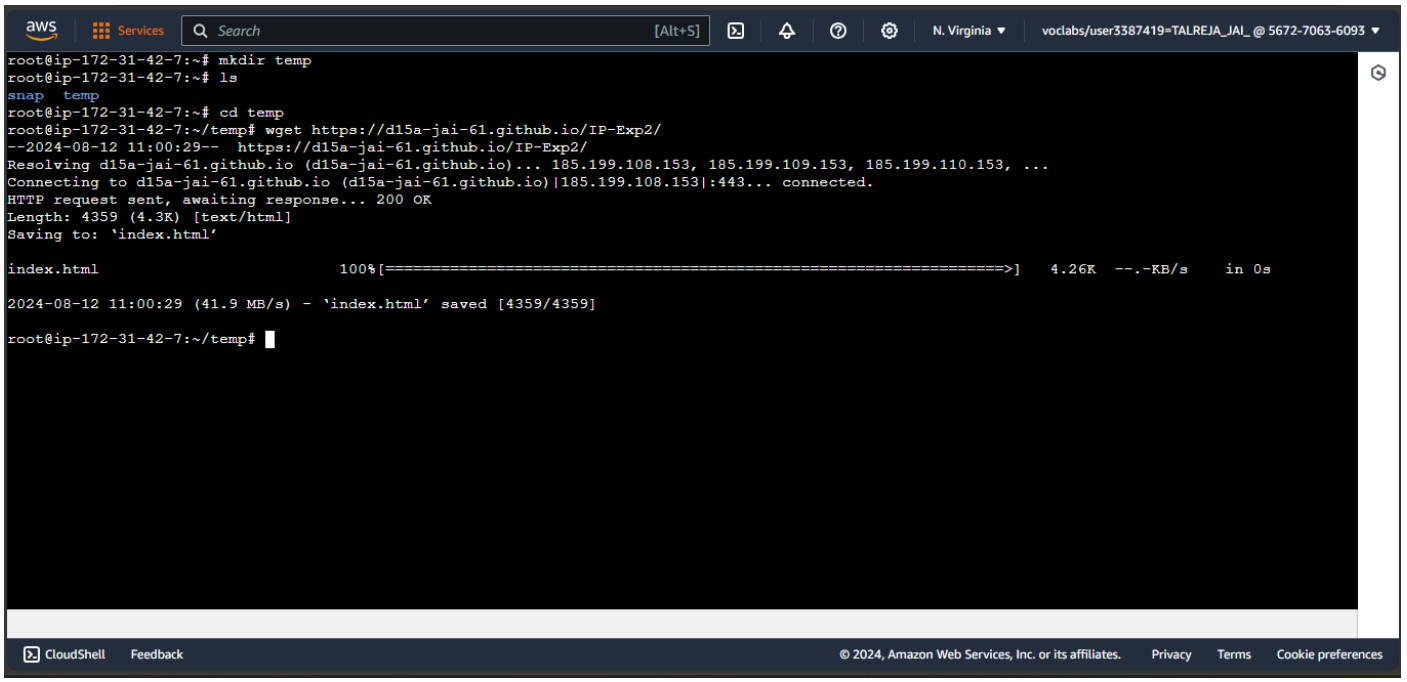
Hitesh Rohra 47 D15A

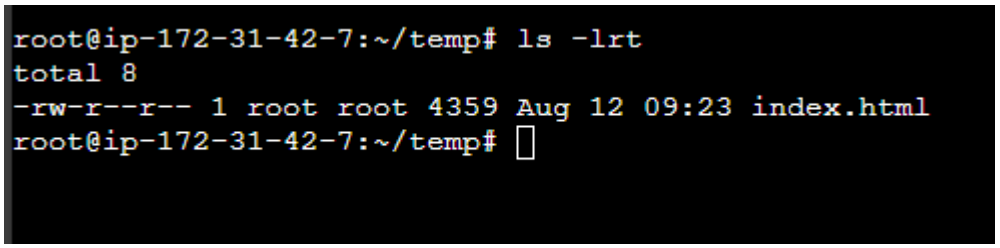
Screenshots for hosting webpage on EC2 instance :-

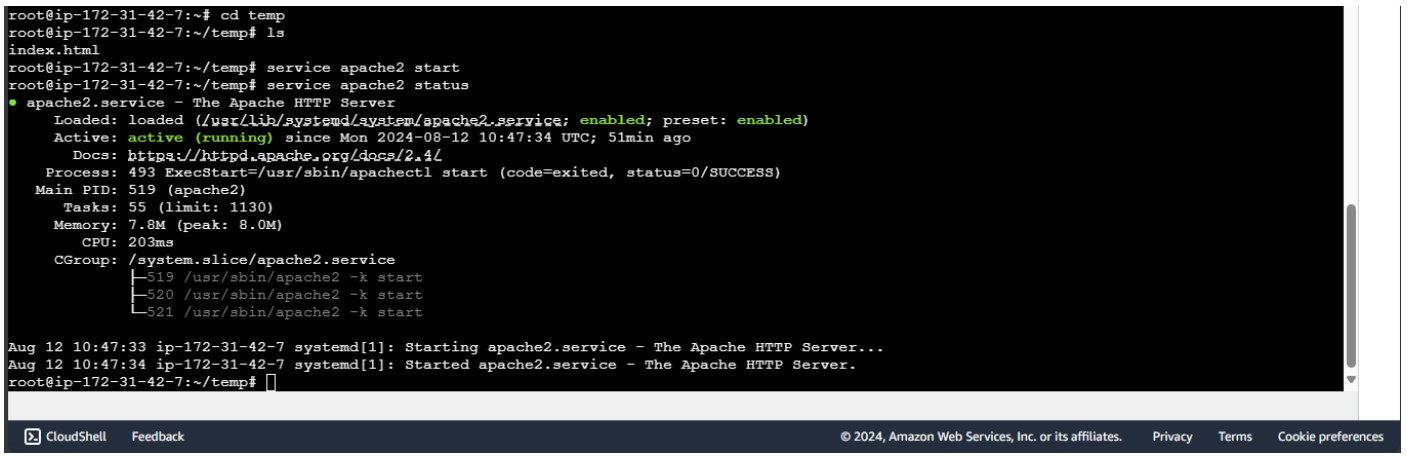




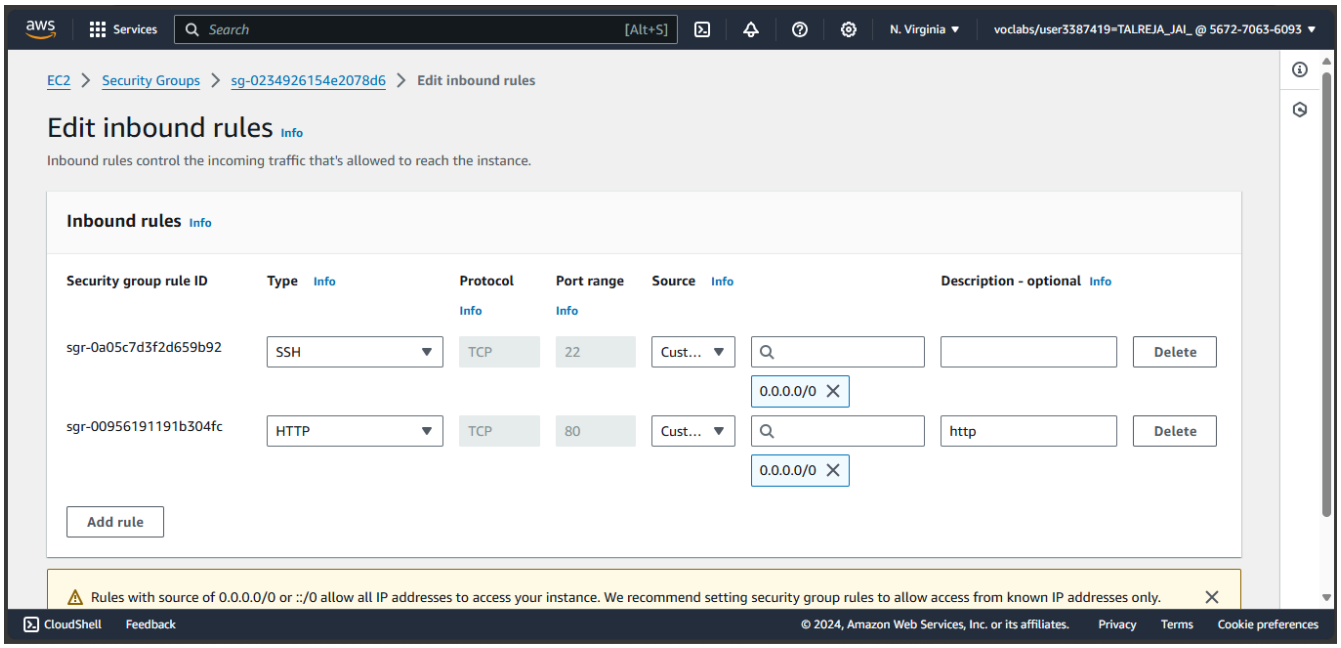
Hitesh Rohra 47 D15A

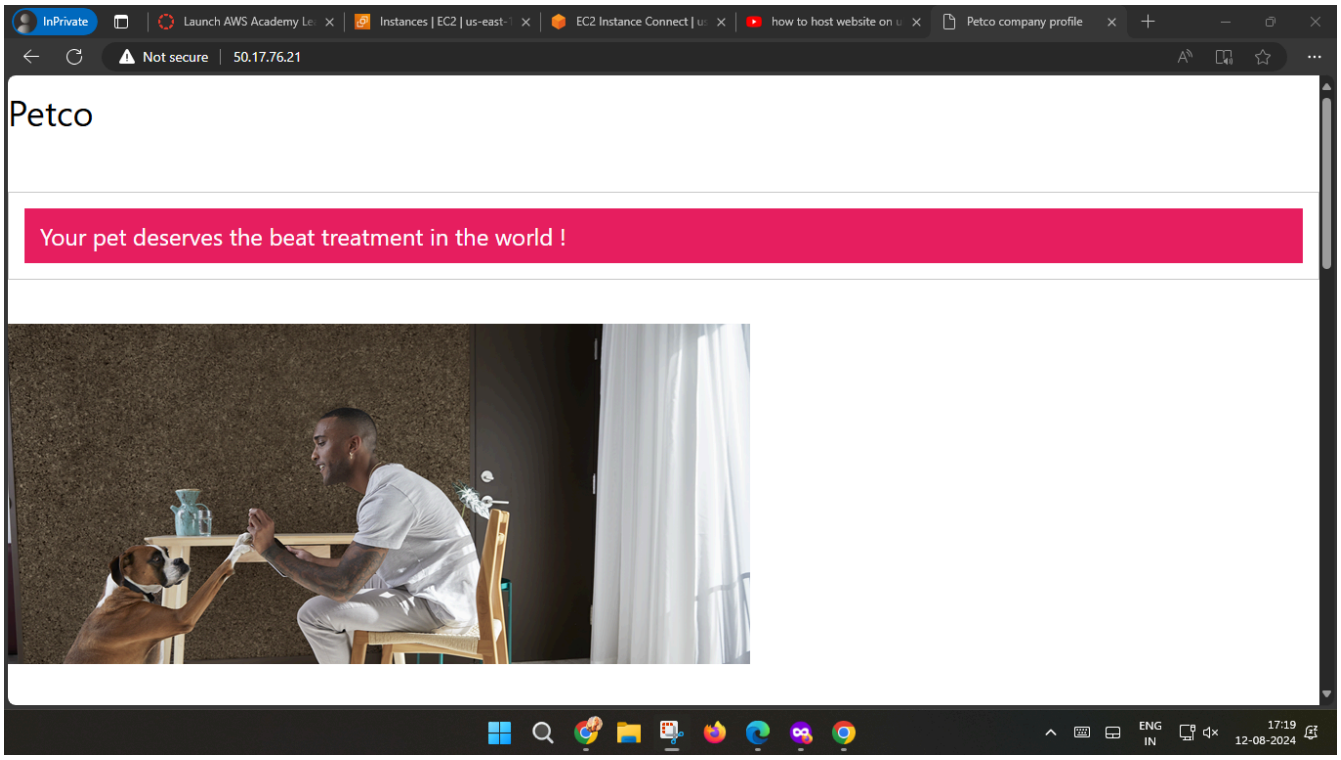




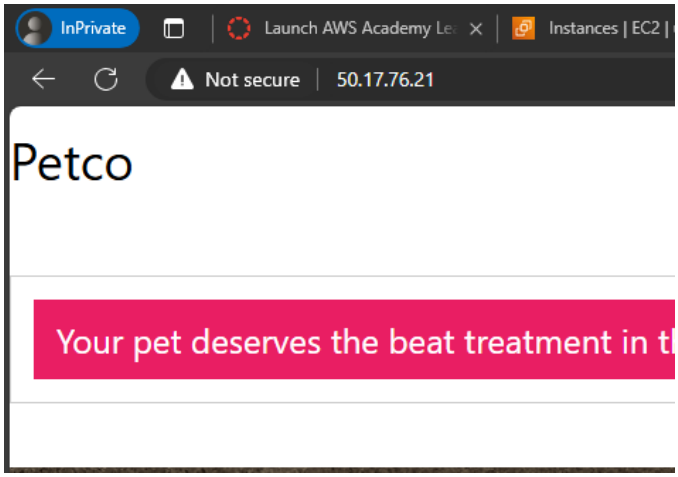


Hitesh Rohra 47 D15A

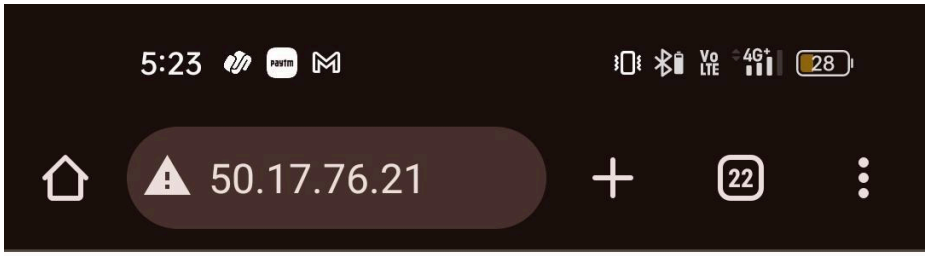




Hitesh Rohra 47 D15A



Hitesh Rohra 47 D15A



Petco

Your pet deserves the beat treatment in the world !

About us

Petco is a category-defining health and wellness company focused on improving the lives of pets, pet parents and our own Petco partners. Since our founding in 1965, we've been trailblazing new standards in pet care, delivering comprehensive wellness solutions through our products and services, and creating communities that deepen the pet-pet parent bond.

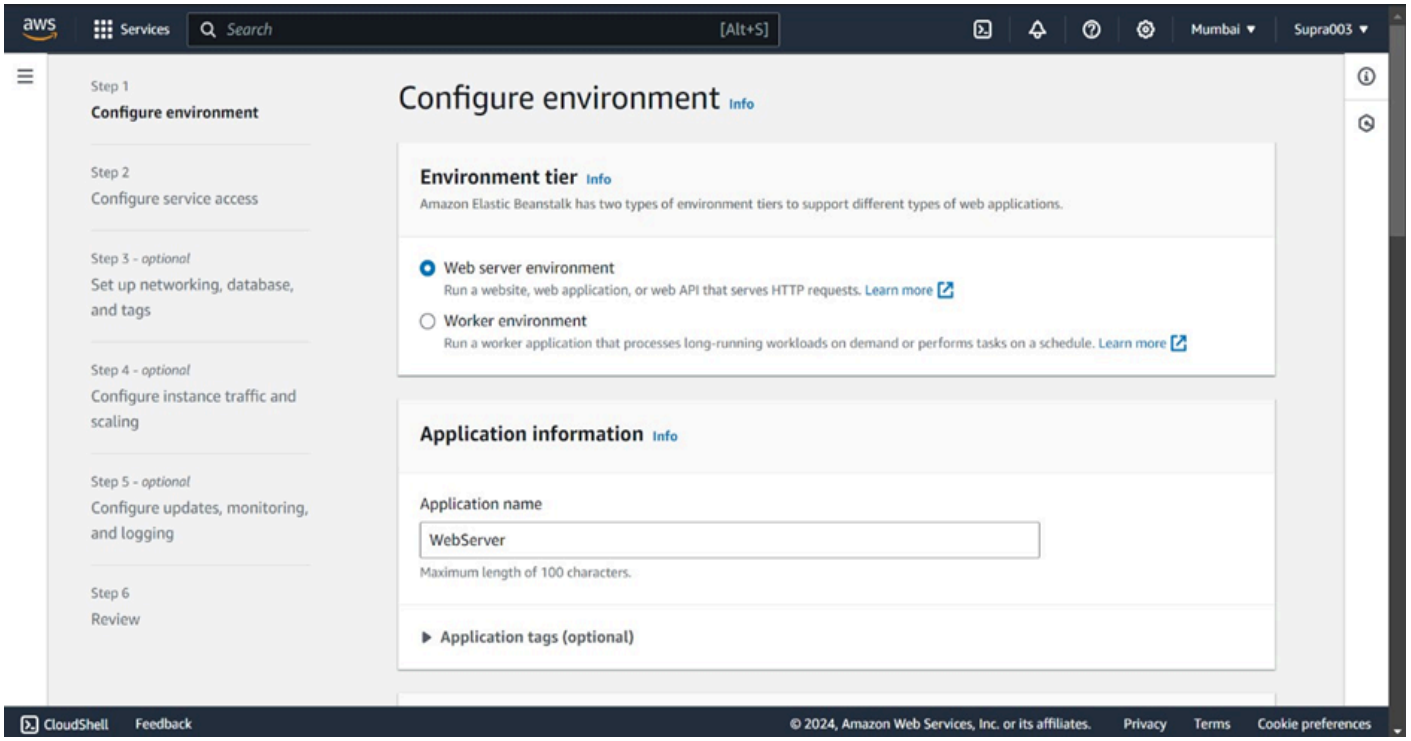
We employ more than 29,000 partners nationwide and operate more than 1,500 Petco locations across the U.S., Mexico and Puerto Rico — including a growing network of more than 200 in-store veterinary hospitals — and offer a complete online resource for pet health and wellness at petco.com and on the Petco app.

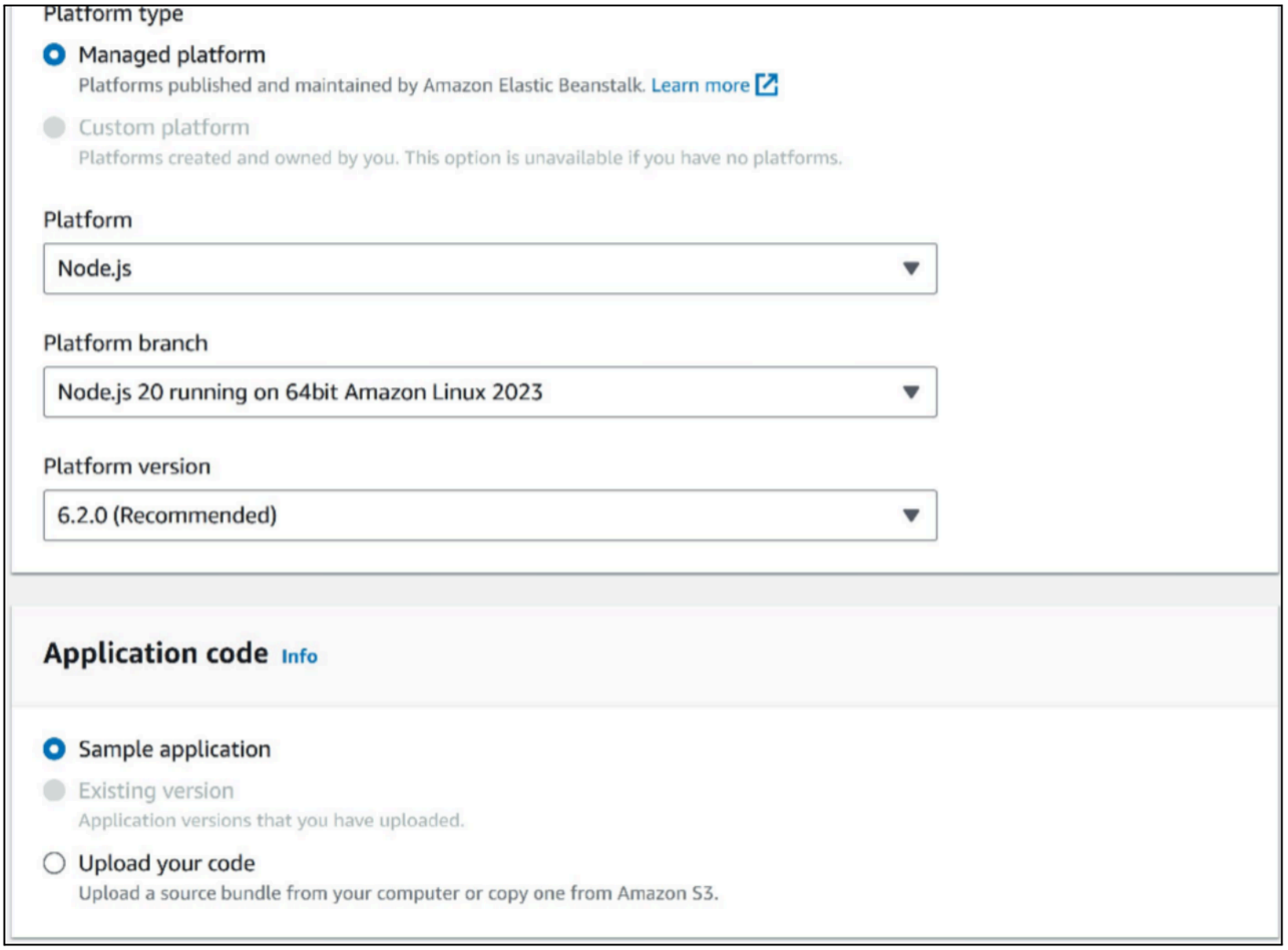
In tandem with Petco Love, an independent nonprofit organization, we have helped find homes for more than 7 million animals through in-store adoption events.

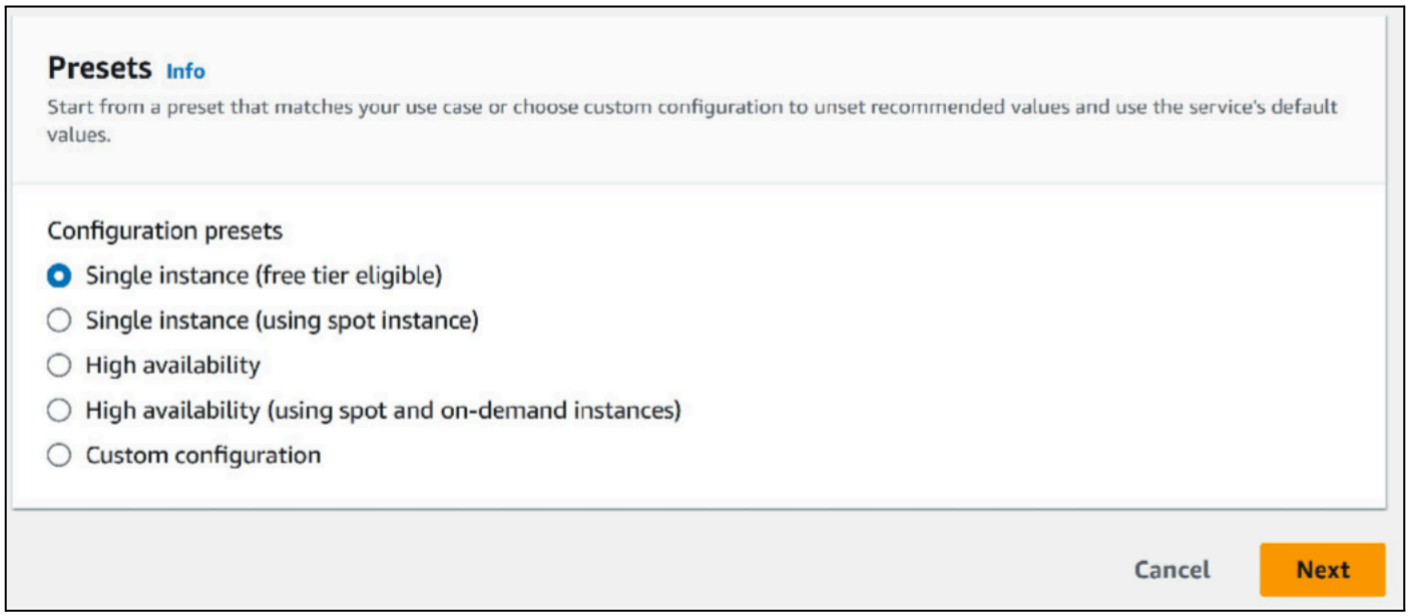
Working at Petco

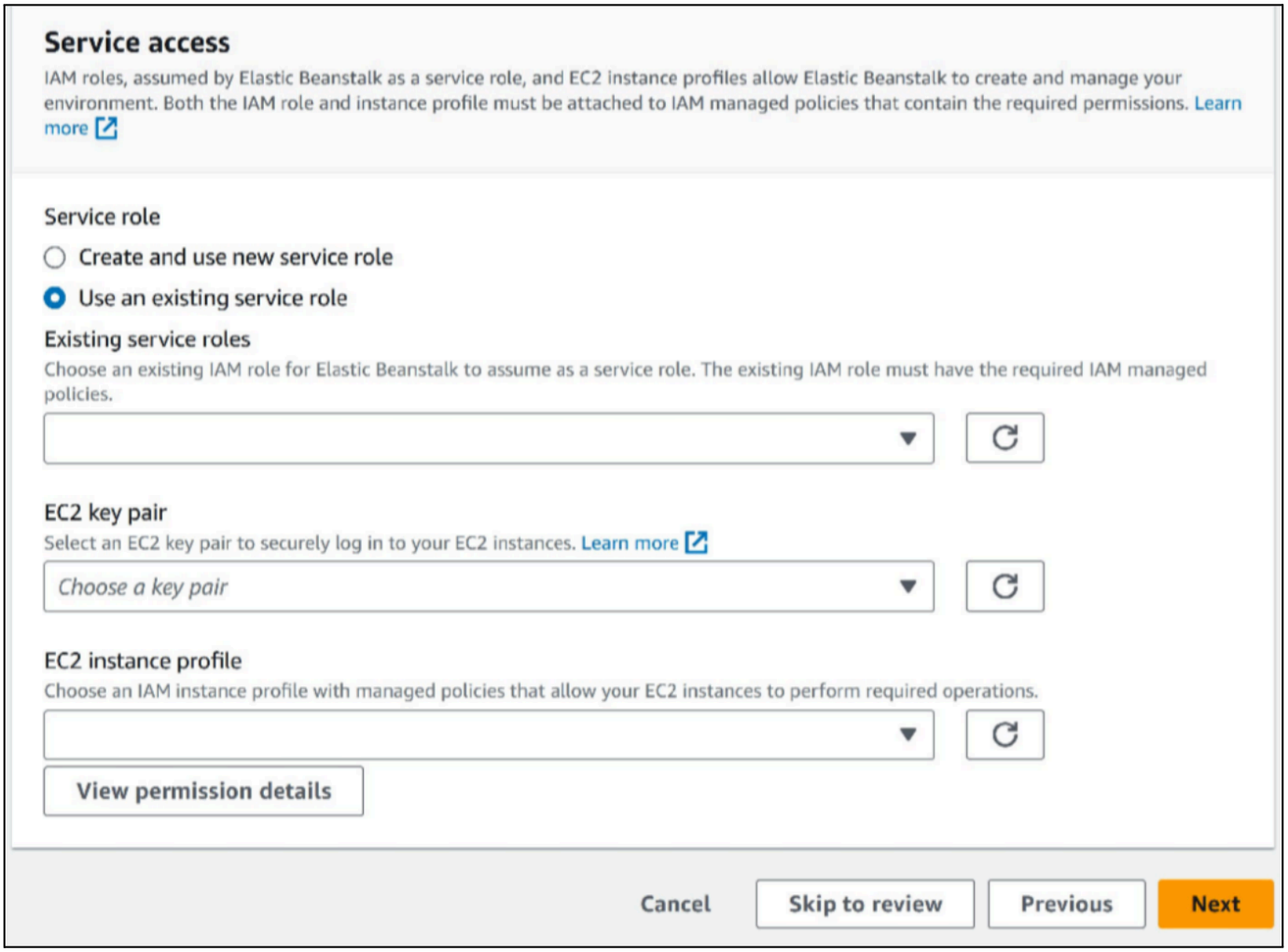
Experiment No 2

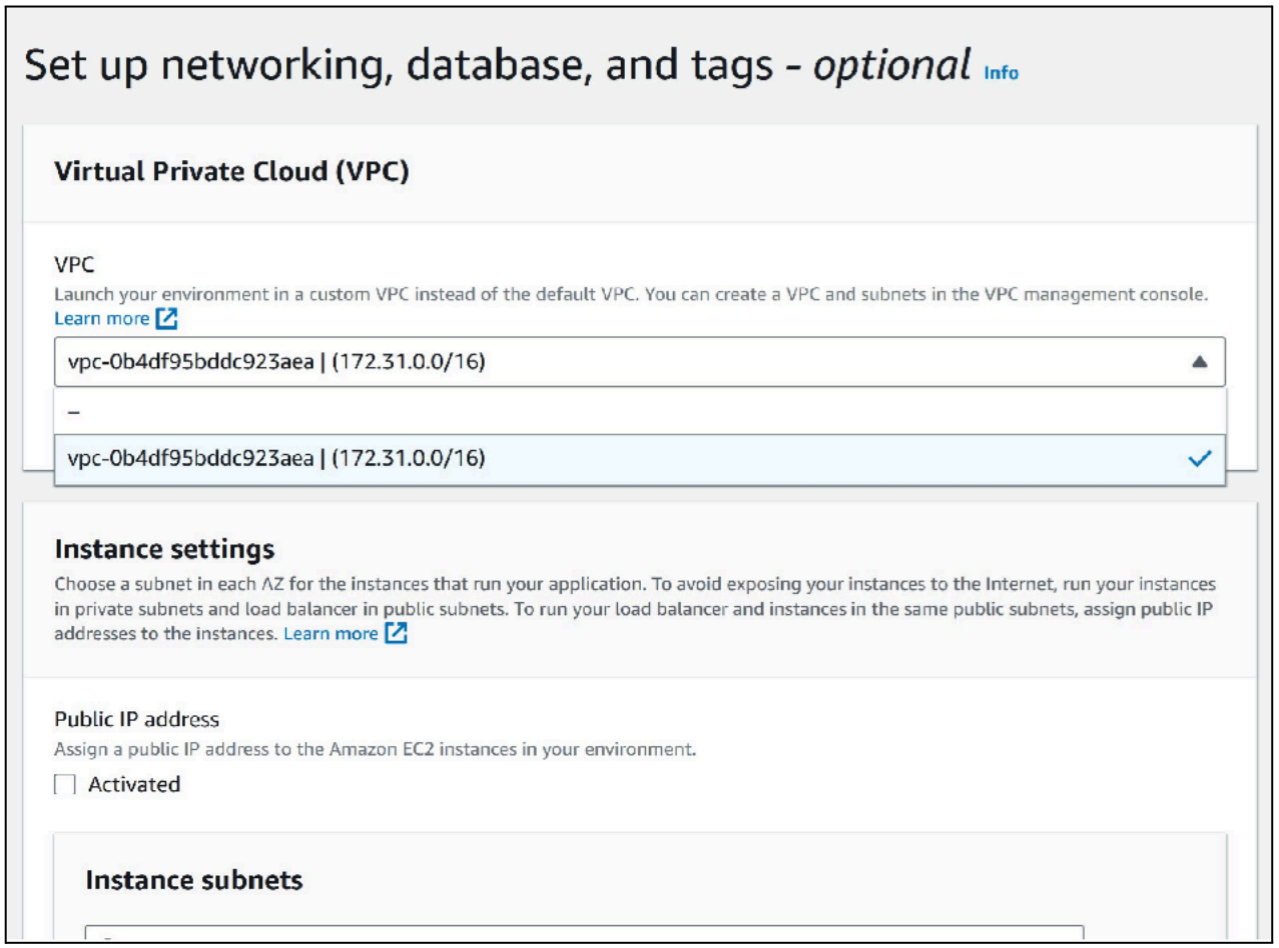
Screenshots for setting up and deploying Elastic Beanstalk application :-

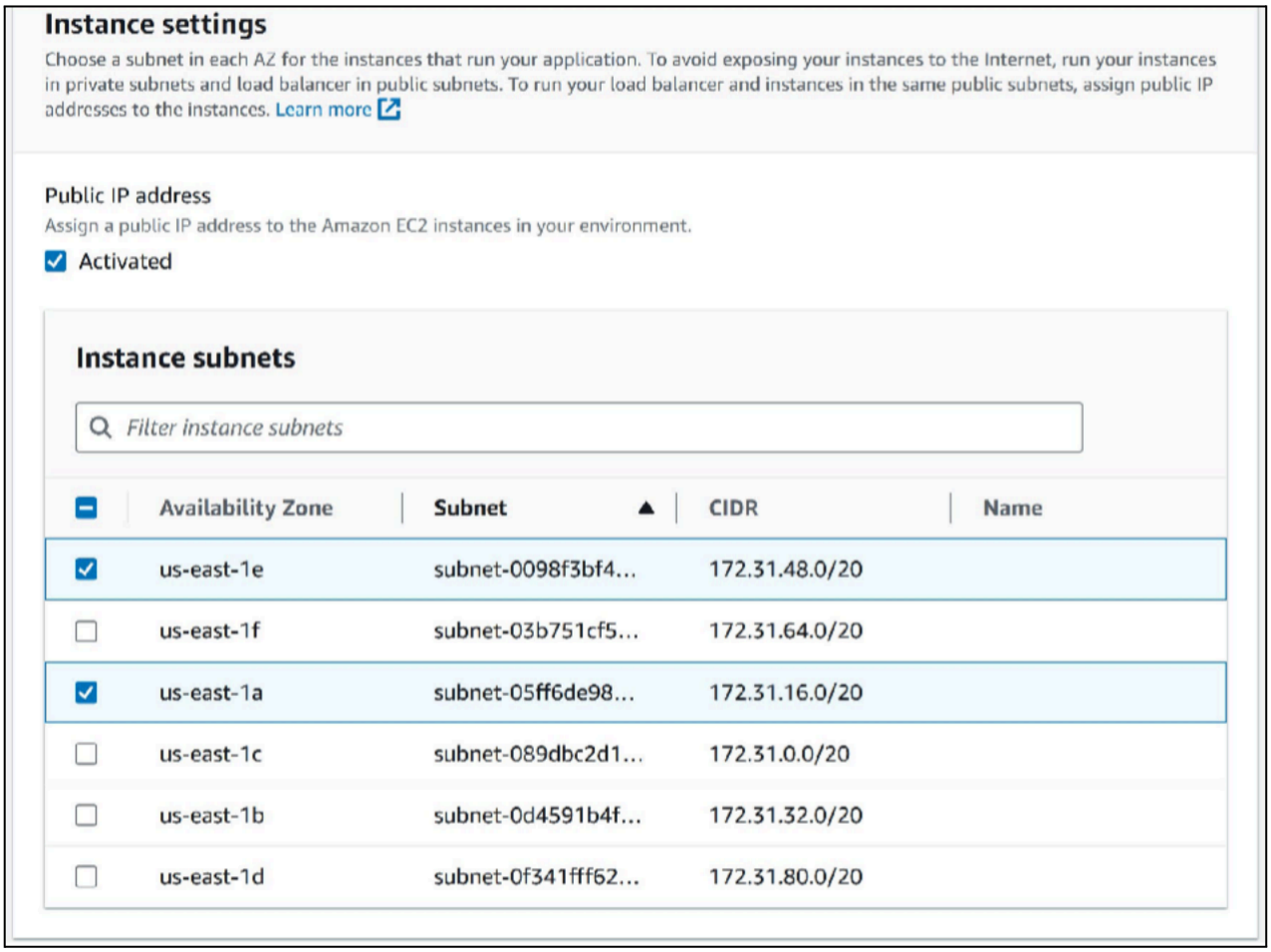





Presets Info
Start from a preset that matches your use case or choose custom configuration to unset recommended values and use the service's default values.
Configuration presets
Single instance (free tier eligible)
Single instance (using spot instance)
High availability
High availability (using spot and on-demand instances)
Custom configuration
Cancel
Next


Service access
IAM roles, assumed by Elastic Beanstalk as a service role, and EC2 instance profiles allow Elastic Beanstalk to create and manage your environment. Both the IAM role and instance profile must be attached to IAM managed policies that contain the required permissions. Learn more
Service role
Create and use new service role
Use an existing service role
Existing service roles
Choose an existing IAM role for Elastic Beanstalk to assume as a service role. The existing IAM role must have the required IAM managed policies.
EC2 key pair
Select an EC2 key pair to securely log in to your EC2 instances. Learn more
Choose a key pair
EC2 instance profile
Choose an IAM instance profile with managed policies that allow your EC2 instances to perform required operations.
View permission details
Cancel
Skip to review
Previous
Next


Set up networking, database, and tags - optional Info
Virtual Private Cloud (VPC)
VPC
Launch your environment in a custom VPC instead of the default VPC. You can create a VPC and subnets in the VPC management console.
Learn more
vpc-0b4df95bddc923aea | (172.31.0.0/16)
–
vpc-0b4df95bddc923aea | (172.31.0.0/16)
Instance settings
Choose a subnet in each AZ for the instances that run your application. To avoid exposing your instances to the Internet, run your instances in private subnets and load balancer in public subnets. To run your load balancer and instances in the same public subnets, assign public IP addresses to the instances. Learn more
Public IP address
Assign a public IP address to the Amazon EC2 instances in your environment.
Activated
Instance subnets


Instance settings
Choose a subnet in each AZ for the instances that run your application. To avoid exposing your instances to the Internet, run your instances in private subnets and load balancer in public subnets. To run your load balancer and instances in the same public subnets, assign public IP addresses to the instances. Learn more
Public IP address
Assign a public IP address to the Amazon EC2 instances in your environment.
Activated
Instance subnets
Filter instance subnets
Availability Zone
Subnet
CIDR
Name
us-east-1e subnet-0098f3bf4... 172.31.48.0/20
us-east-1f subnet-03b751cf5... 172.31.64.0/20
us-east-1a subnet-05ff6de98... 172.31.16.0/20
us-east-1c subnet-089dbc2d1... 172.31.0.0/20
us-east-1b subnet-0d4591b4f... 172.31.32.0/20
us-east-1d subnet-0f341fff62... 172.31.80.0/20

☐	us-east-1f	subnet-03b751cf5...	172.31.64.0/20
☐	us-east-1a	subnet-05ff6de98...	172.31.16.0/20
☐	us-east-1c	subnet-089dbc2d1...	172.31.0.0/20
☐	us-east-1b	subnet-0d4591b4f...	172.31.32.0/20
☐	us-east-1d	subnet-0f341fff62...	172.31.80.0/20

Enable database

Restore a snapshot - *optional*

Restore an existing snapshot from a previously used database.

Snapshot

None

Database settings

Choose an engine and instance type for your environment's database.

Engine

Engine version

Tags

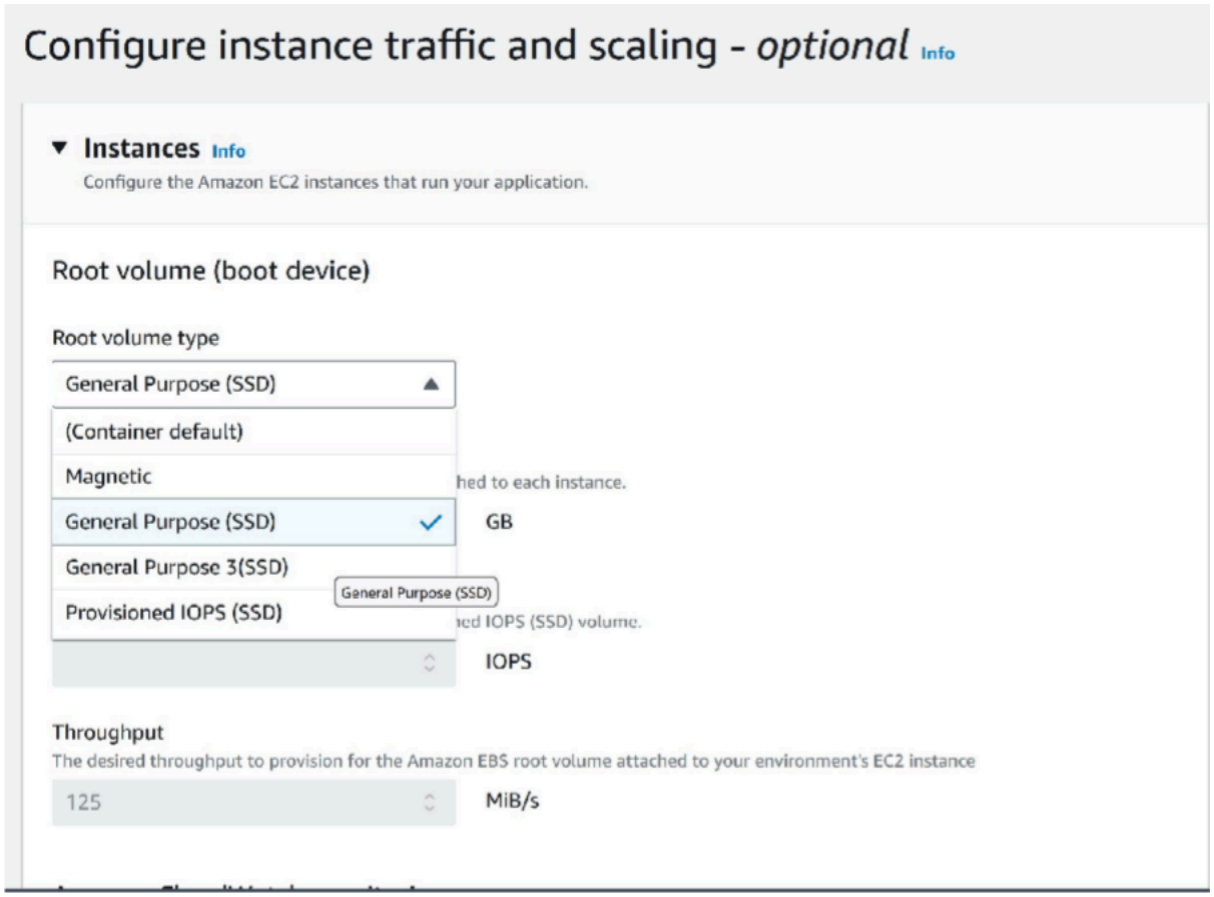
Apply up to 50 tags. You can use tags to group and filter your resources. A tag is a key-value pair. The key must be unique within the resource and is case-sensitive. Learn more

No tags associated with the resource.

Add new tag

You can add 50 more tags.

Cancel | Skip to review | Previous | Next



Root volume type

General Purpose (SSD)

Size

The number of gigabytes of the root volume attached to each instance.

8 GB

IOPS

Input/output operations per second for a provisioned IOPS (SSD) volume.

IOPS

Throughput

The desired throughput to provision for the Amazon EBS root volume attached to your environment's EC2 instance

125 MiB/s

Amazon CloudWatch monitoring

The time interval between when metrics are reported from the EC2 instances

Monitoring interval

5 minute

Instance metadata service (IMDS)

Your environment's platform supports both IMDSv1 and IMDSv2. To enforce IMDSv2, deactivate IMDSv1. Learn more

IMDSv1

With the current setting, the environment enables only IMDSv2.

☑ Deactivated

EC2 security groups

Select security groups to control traffic.

EC2 security groups (3)

Filter security groups

	Group name	Group ID	Name
☑	default	sg-0b6f2684b3eea9987	
☐	launch-wizard-1	sg-0234926154c2078d6	
☐	launch-wizard-2	sg-0f76834772eb69439	

Capacity Info

Configure the compute capacity of your environment and auto scaling settings to optimize the number of instances used.

Auto scaling group

Environment type

Select a single-instance or load-balanced environment. You can develop and test an application in a single-instance environment to save costs and then upgrade to a load-balanced environment when the application is ready for production. Learn more

Single instance

Instances

1 Min

1 Max

Fleet composition

Spot instances are launched at the lowest available price. Learn more

◉ On-Demand instance

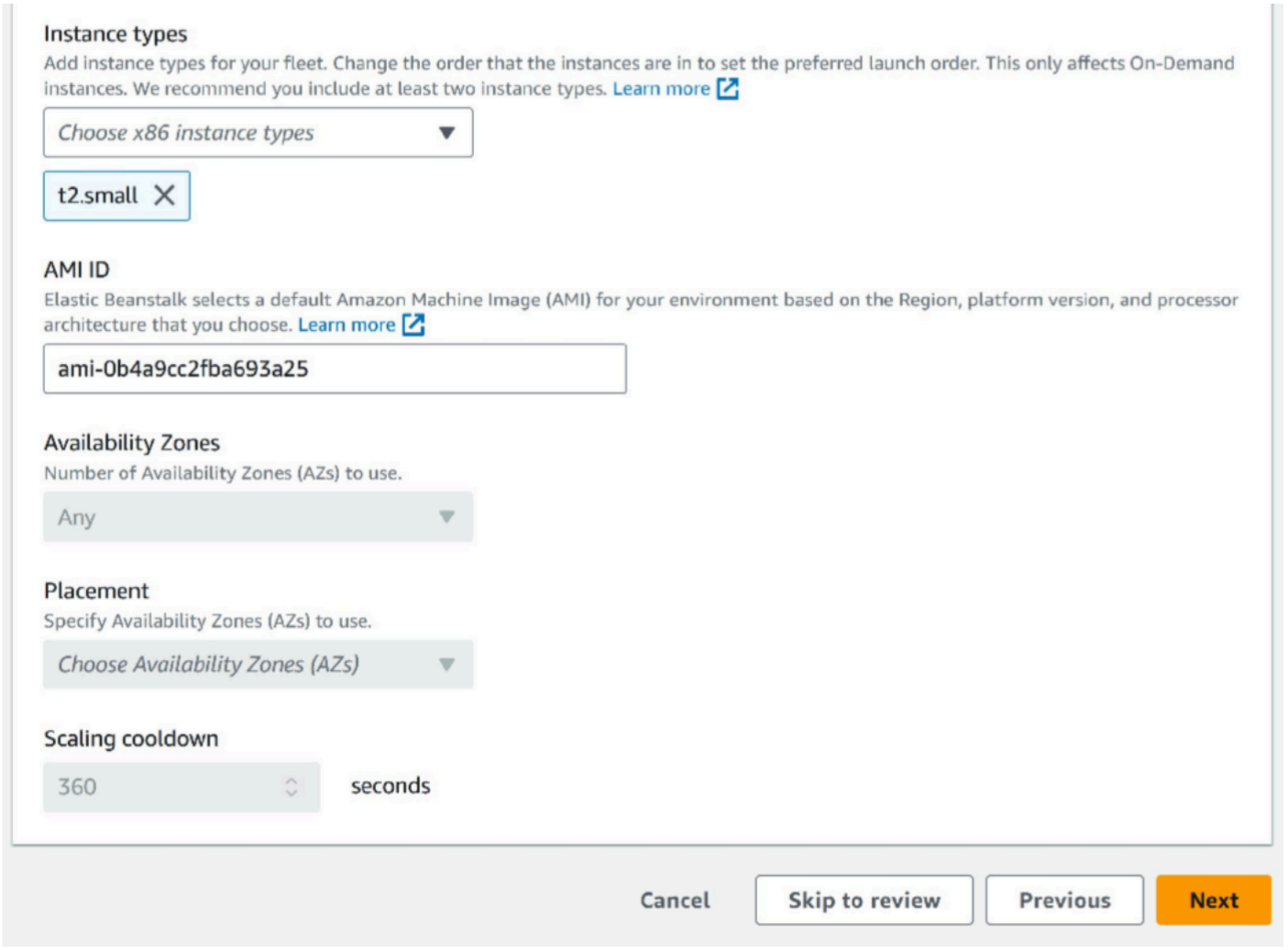
○ Spot instance

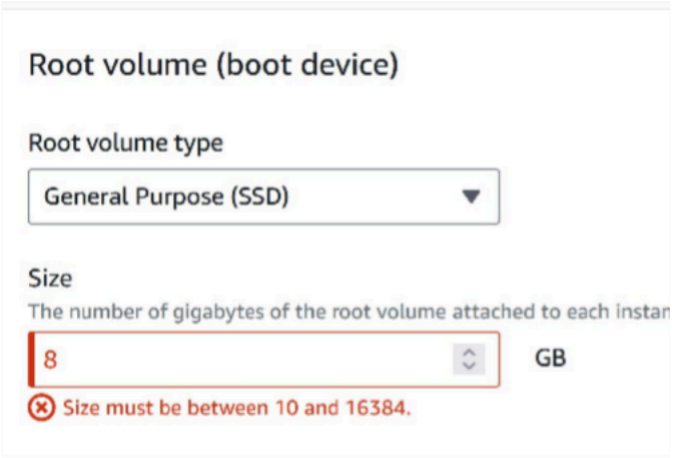
Maximum spot price

The maximum price per instance-hour, in USD, that you're willing to pay for a Spot Instance. Setting a custom price limits your chances to fulfill your target capacity using Spot instances.

○ Default

○ Set your maximum price


Instance types
Add instance types for your fleet. Change the order that the instances are in to set the preferred launch order. This only affects On-Demand instances. We recommend you include at least two instance types. Learn more
Choose x86 instance types
t2.small
AMI ID
Elastic Beanstalk selects a default Amazon Machine Image (AMI) for your environment based on the Region, platform version, and processor architecture that you choose. Learn more
ami-0b4a9cc2fba693a25
Availability Zones
Number of Availability Zones (AZs) to use.
Any
Placement
Specify Availability Zones (AZs) to use.
Choose Availability Zones (AZs)
Scaling cooldown
360
seconds
Cancel
Skip to review
Previous
Next


Root volume (boot device)
Root volume type
General Purpose (SSD)
Size
The number of gigabytes of the root volume attached to each instar
8
GB
Size must be between 10 and 16384.


Root volume type
General Purpose (SSD)
Size
The number of gigabytes of the root volume attached to each instance.
16
GB

Configure updates, monitoring, and logging - *optional* Info

Monitoring Info

Health reporting

Enhanced health reporting provides free real-time application and operating system monitoring of the instances and other resources in your environment. The **EnvironmentHealth** custom metric is provided free with enhanced health reporting. Additional charges apply for each custom metric. For more information, see Amazon CloudWatch Pricing

System

- (•) Basic
- ( ) Enhanced

Health event streaming to CloudWatch Logs

Configure Elastic Beanstalk to stream environment health events to CloudWatch Logs. You can set the retention up to a maximum of ten years and configure Elastic Beanstalk to delete the logs when you terminate your environment.

Log streaming

- [ ] Activated (standard CloudWatch charges apply.)

Retention

7

Lifecycle

Managed platform updates Info

Activate managed platform updates to apply platform updates automatically during a weekly maintenance window that you choose. Your application stays available during the update process.

Managed updates

- [ ] Activated

Weekly update window

Tuesday at 01 : 14 UTC

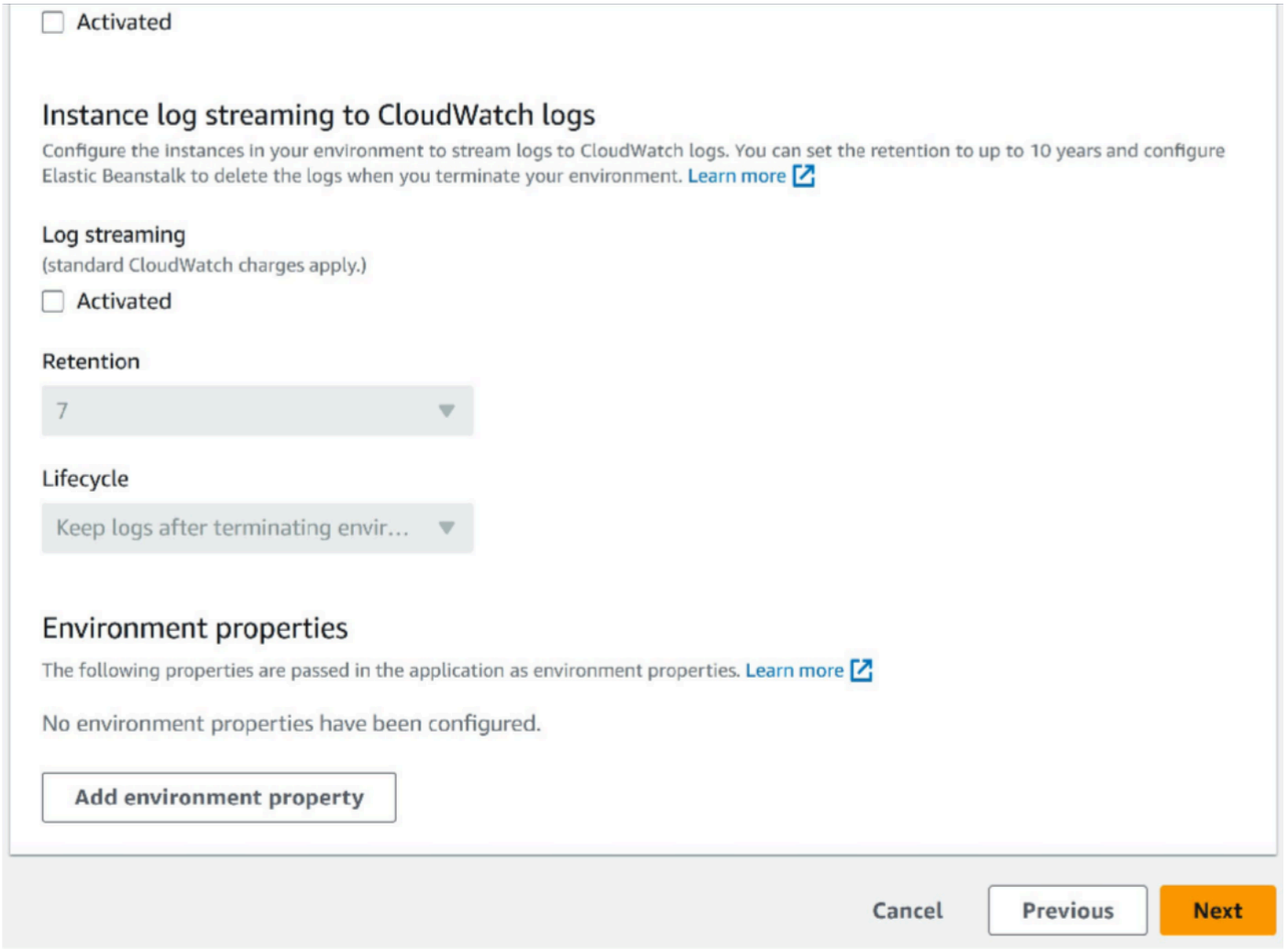
Update level

Minor and patch

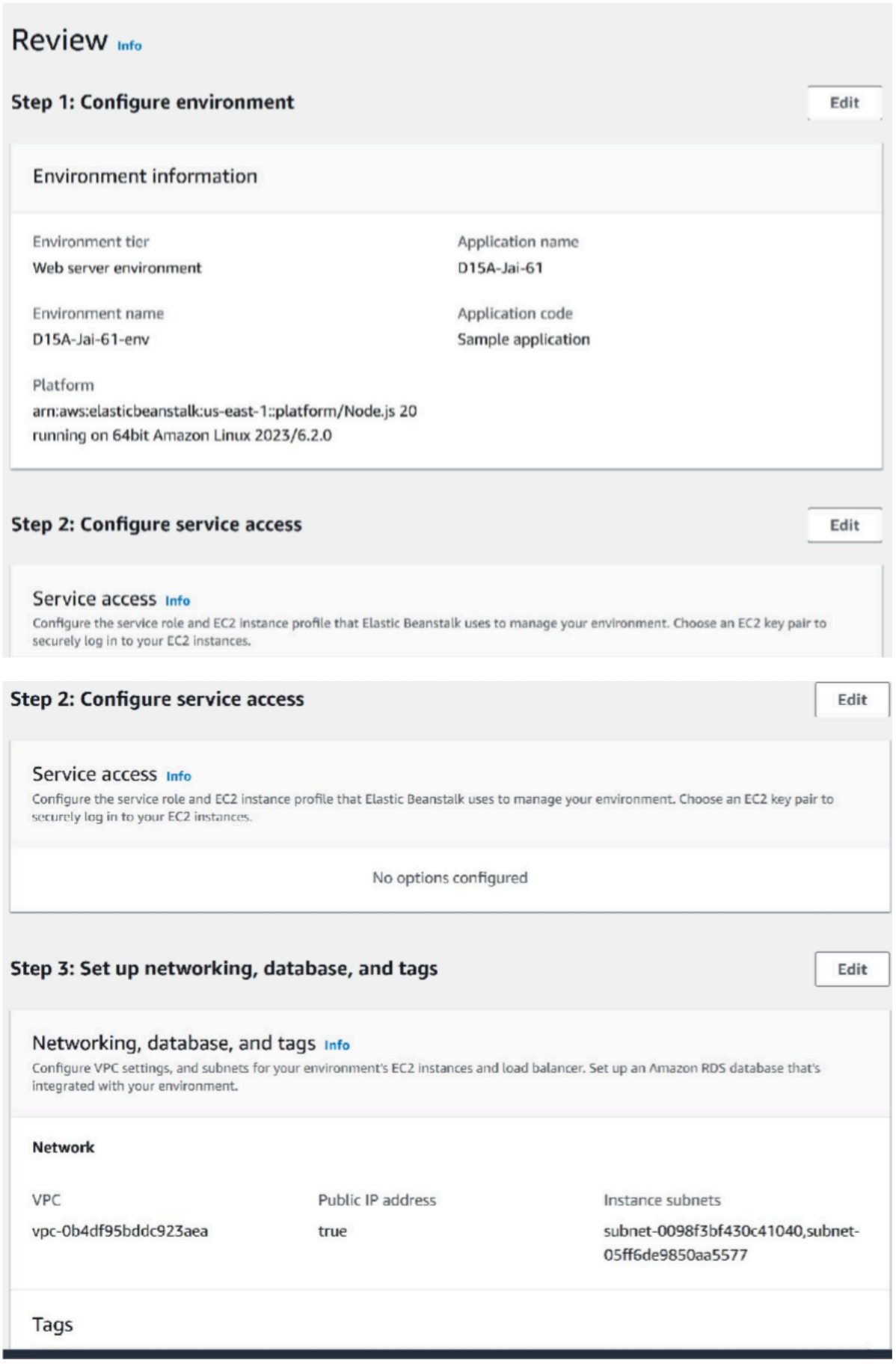
Instance replacement

If enabled, an instance replacement will be scheduled if no other updates are available.

- [ ] Activated


Activated
Instance log streaming to CloudWatch logs
Configure the instances in your environment to stream logs to CloudWatch logs. You can set the retention to up to 10 years and configure Elastic Beanstalk to delete the logs when you terminate your environment. Learn more
Log streaming
(standard CloudWatch charges apply.)
Activated
Retention
7
Lifecycle
Keep logs after terminating envir...
Environment properties
The following properties are passed in the application as environment properties. Learn more
No environment properties have been configured.
Add environment property
Cancel
Previous
Next

Elastic Beanstalk application created, review screen before finalization :-



Step 4: Configure instance traffic and scaling

Edit

Instance traffic and scaling Info

Customize the capacity and scaling for your environment's instances. Select security groups to control instance traffic.Configure the software that runs on your environment's instances by setting platform-specific options.

Instances

Root volume type	Instance size	IMDSv1
gp2	16	Deactivated

EC2 Security Groups

sg-0b6f2684b3eea9987

Capacity

Environment type	Fleet composition	On-demand base
Single instance	On-Demand instance	0
On-demand above base	**Capacity rebalancing**	**Scaling cooldown**
0	Deactivated	360
Processor type	**Instance types**	**AMI ID**
x86_64	t2.small	ami-0b4a9cc2fba693a25

Step 5: Configure updates, monitoring, and logging

Edit

Updates, monitoring, and logging Info

Define when and how Elastic Beanstalk deploys changes to your environment. Manage your application's monitoring and logging settings, instances, and other environment resources.

Monitoring

System	Cloudwatch custom metrics - instance	Cloudwatch custom metrics - environment
basic	—	—
Log streaming	**Retention**	**Lifecycle**
Deactivated	7	false

Updates

Managed updates	Deployment batch size	Deployment batch size type
Deactivated	100	Percentage
Command timeout	**Deployment policy**	**Health threshold**
600	AllAtOnce	Ok
Ignore health check	**Instance replacement**	

Ignore health check
false

Instance replacement
false

Platform software

Lifecycle
false

Log streaming
Deactivated

Proxy server
nginx

Logs retention
7

Rotate logs
Deactivated

Update level
minor

X-Ray enabled
Deactivated

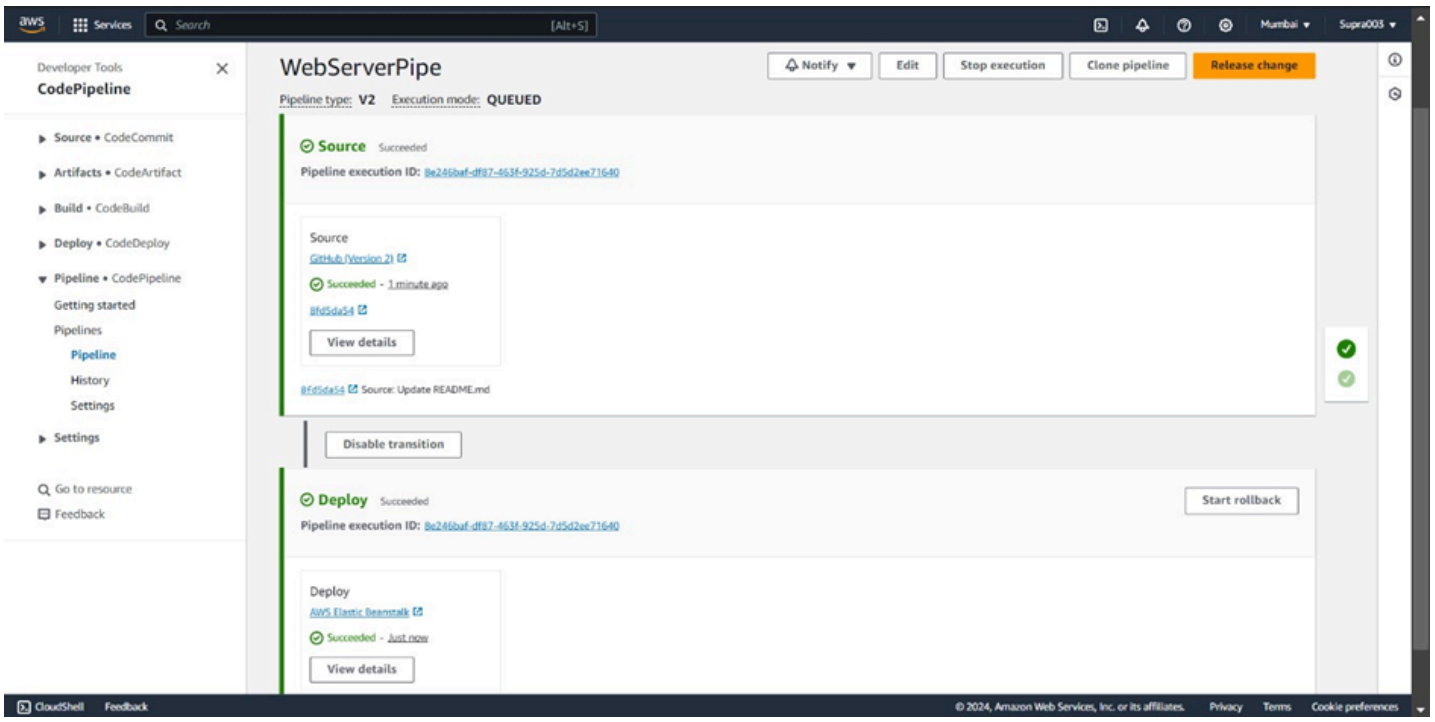
Environment properties

Key	Value

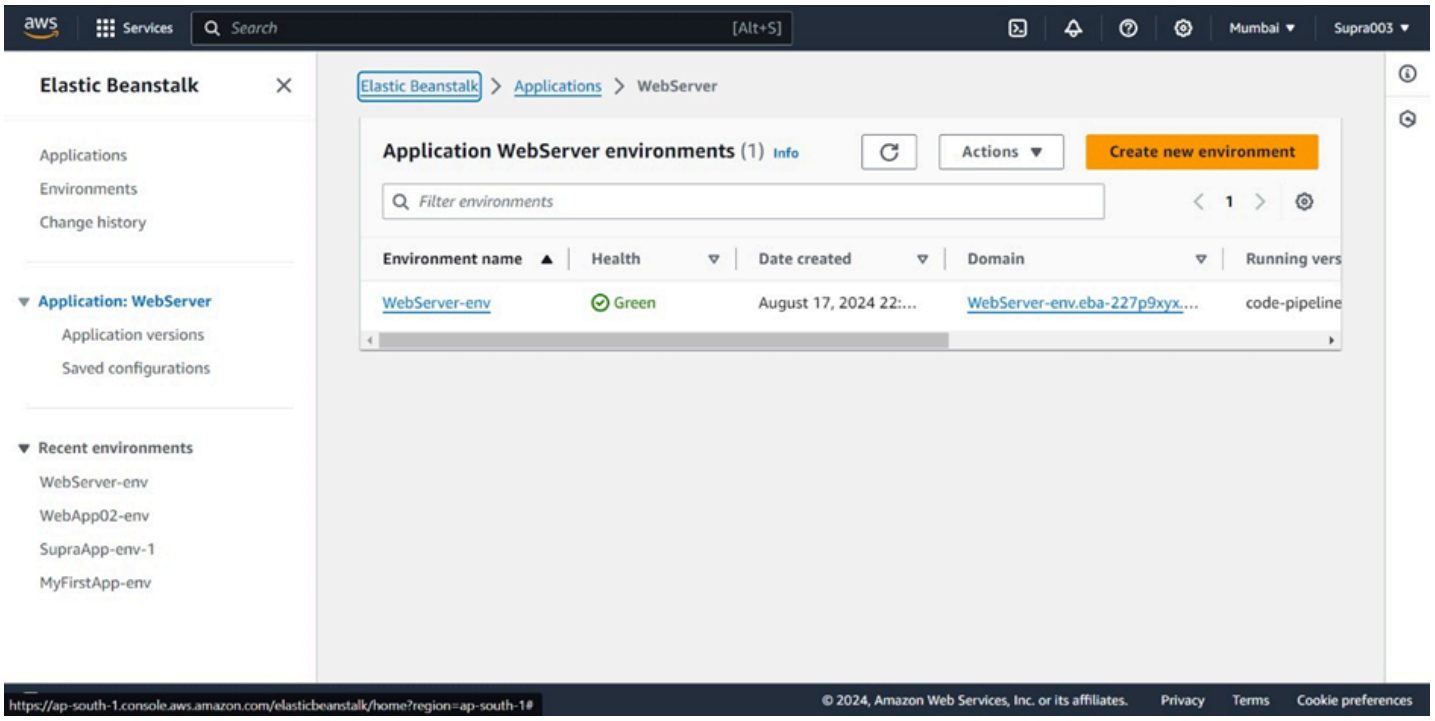
No environment properties

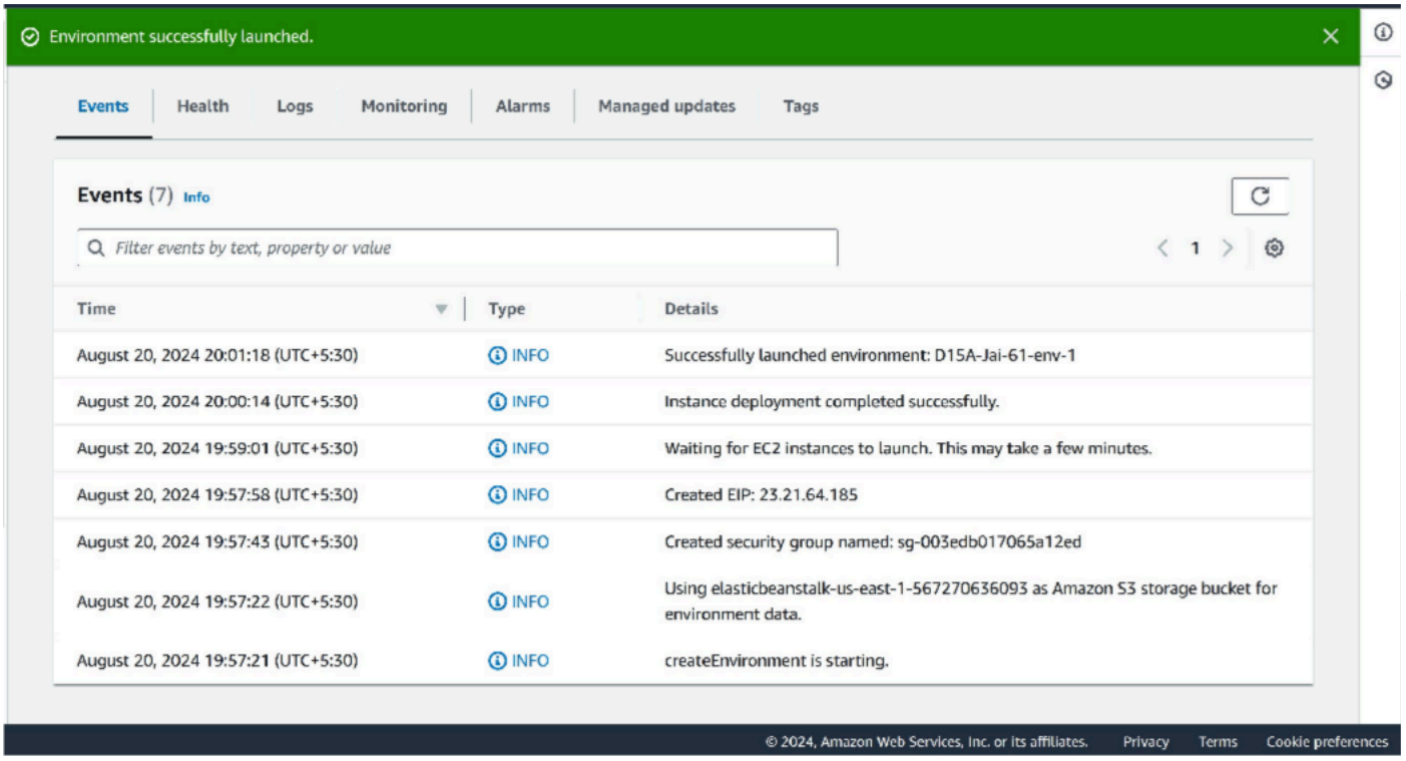
There are no environment properties defined

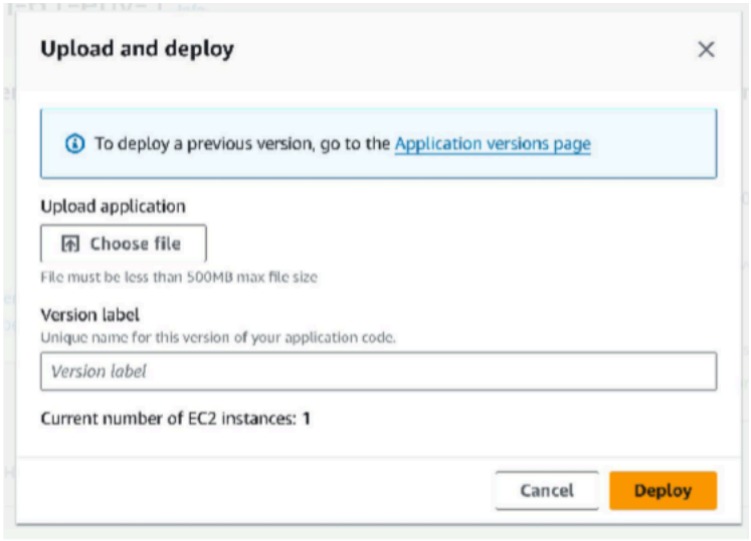
Cancel | Previous | Submit

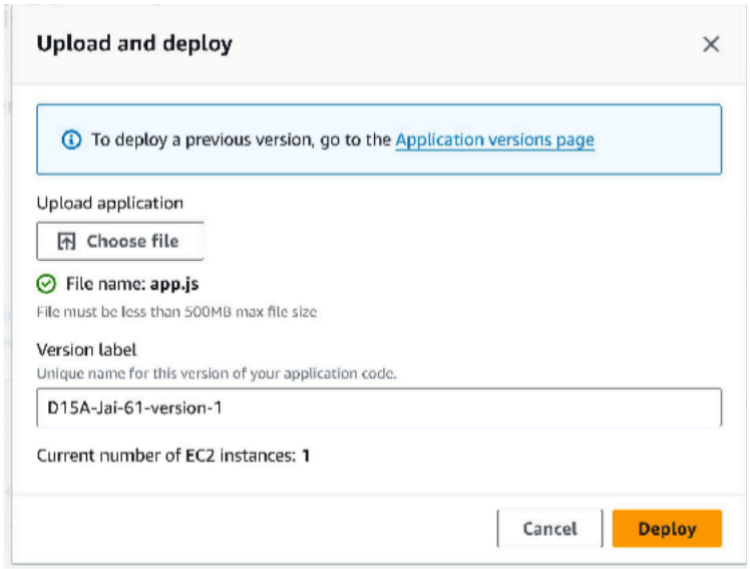


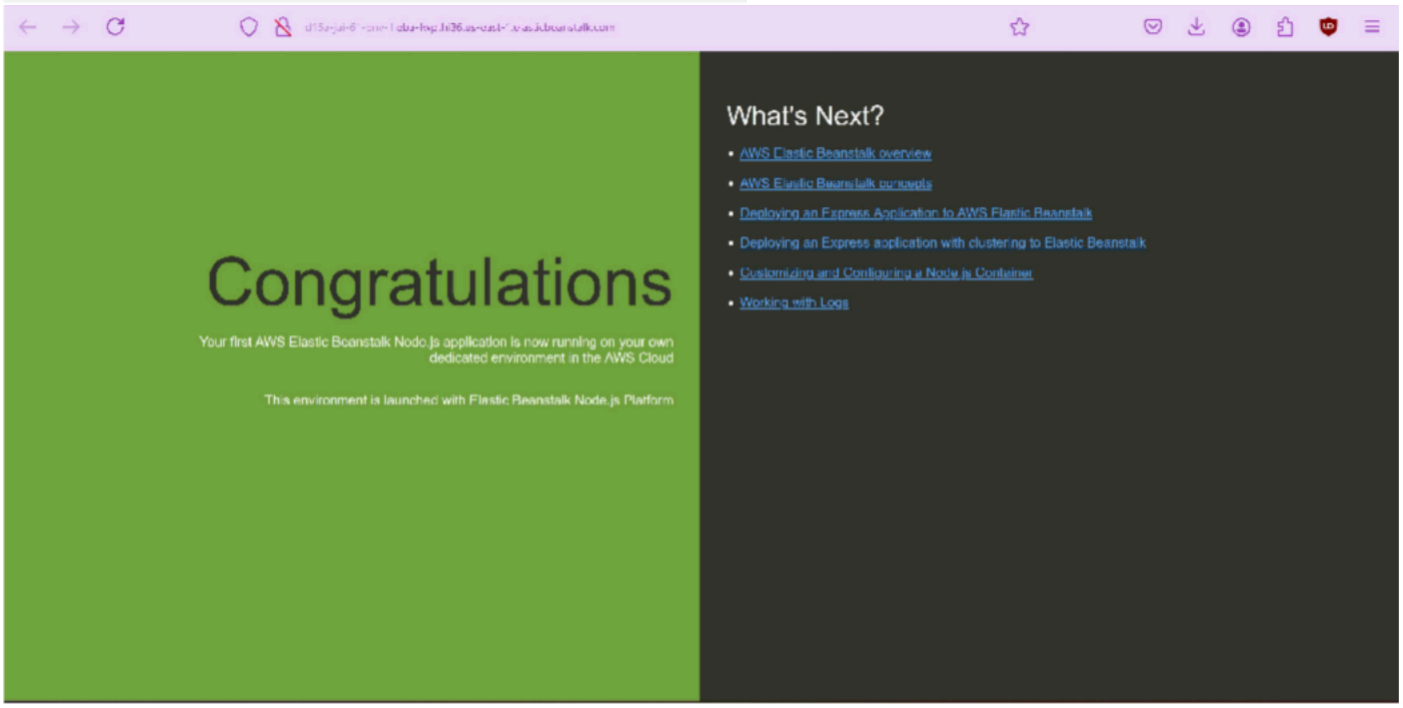
Hitesh Rohra 47 D15A






Upload and deploy
To deploy a previous version, go to the Application versions page
Upload application
Choose file
File must be less than 500MB max file size
Version label
Unique name for this version of your application code.
Version label
Current number of EC2 instances: 1
Cancel
Deploy

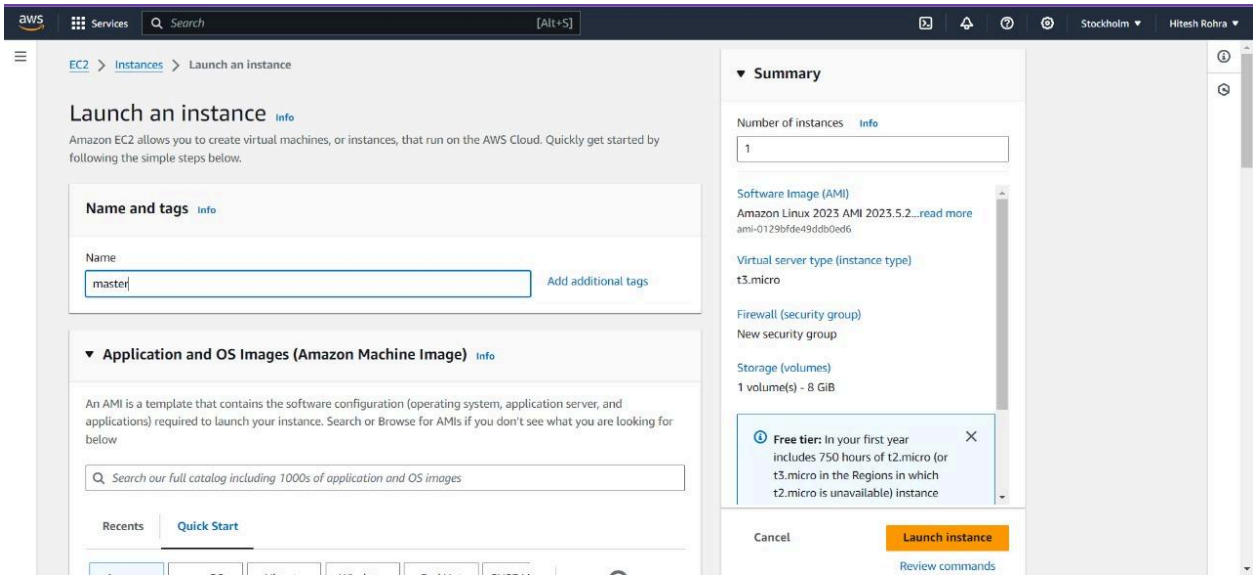

Upload and deploy
To deploy a previous version, go to the Application versions page
Upload application
Choose file
File name: app.js
File must be less than 500MB max file size
Version label
Unique name for this version of your application code.
D15A-Jai-61-version-1
Current number of EC2 instances: 1
Cancel
Deploy

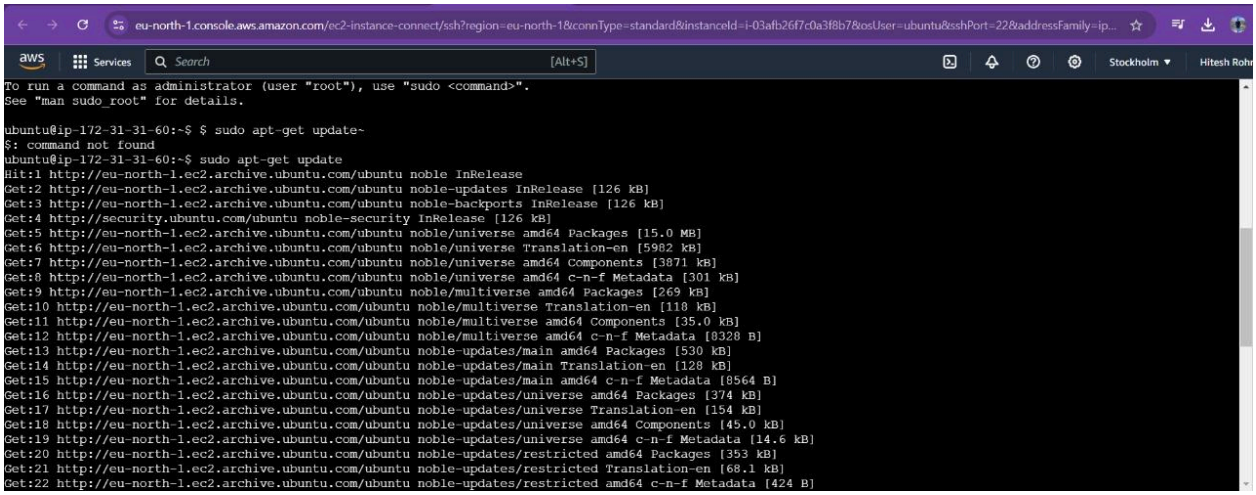

Congratulations
Your first AWS Elastic Beanstalk Node.js application is now running on your own dedicated environment in the AWS Cloud
This environment is launched with Elastic Beanstalk Node.js Platform
What's Next?
AWS Elastic Beanstalk overview
AWS Elastic Beanstalk concepts
Deploying an Express Application to AWS Elastic Beanstalk
Deploying an Express application with clustering to Elastic Beanstalk
Customizing and Configuring a Node.js Container
Working with Logs

Experiment No 3

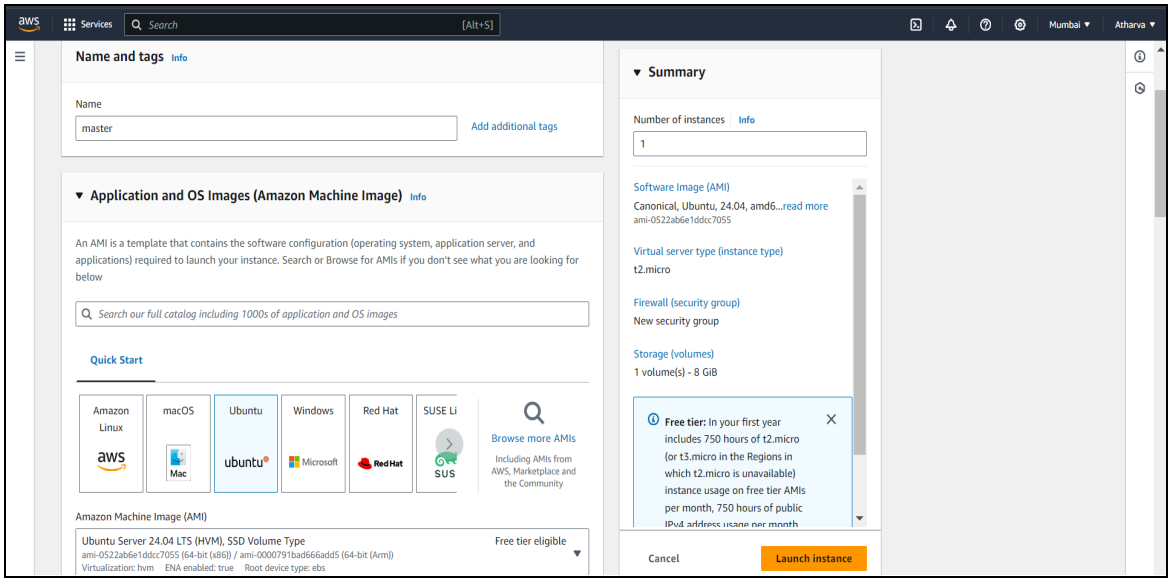
Hitesh Rohra **Experiment 3**
D15A 47
AIM: To understand the Kubernetes Cluster Architecture, install and Spin Up a Kubernetes Cluster on Linux Machines/Cloud Platforms.

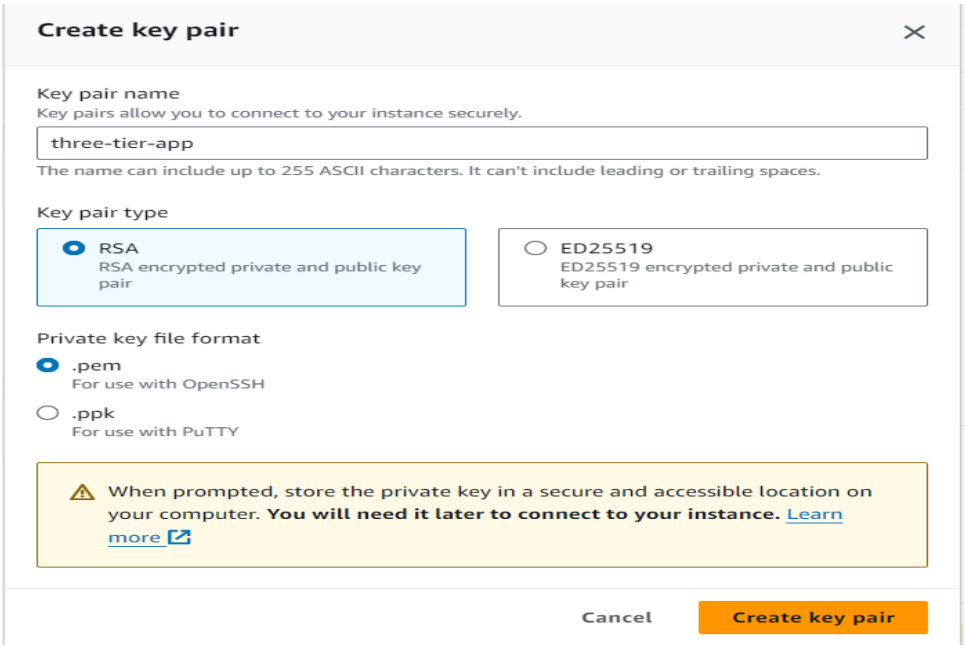
Step 1: Prerequisites
1.1 Create 3 EC2 instances,one for the master node and two for the worker nodes.

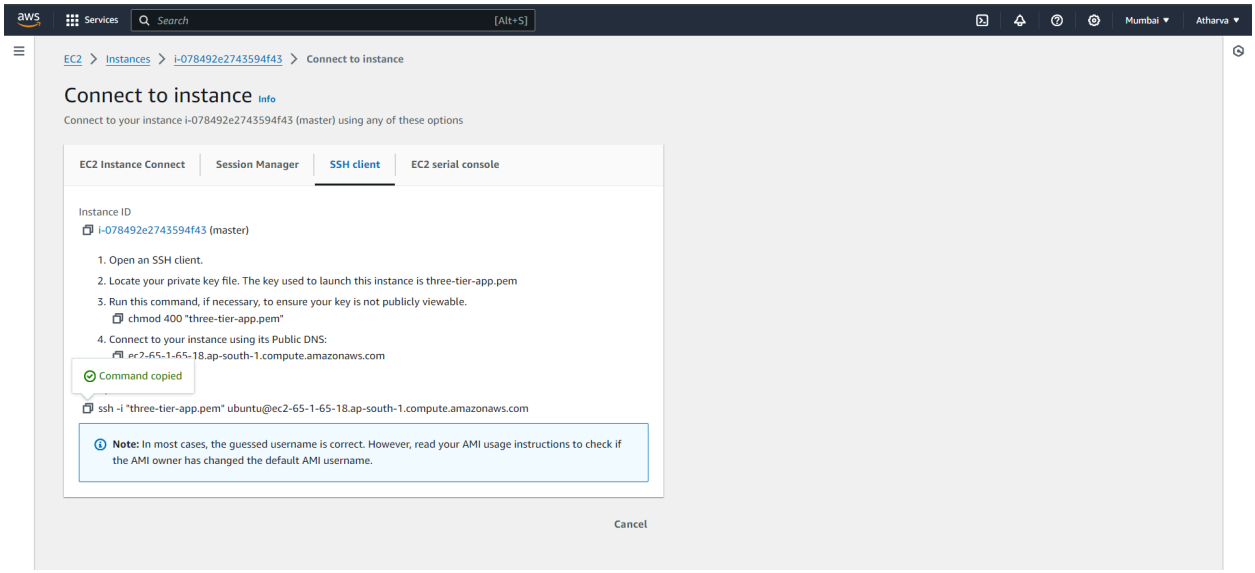




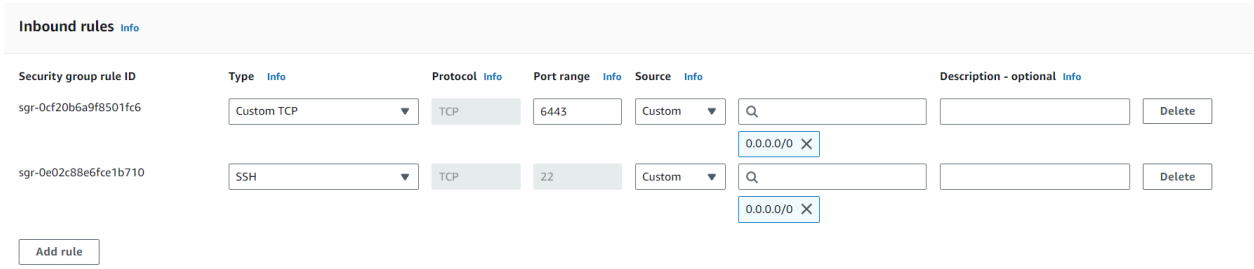
1.2 Proceed with the following settings and create a new key pair as follows(use the same key pair for all the three nodes)


aws
Services
Search
[Alt+S]
Mumbai
Atharva
Name and tags Info
Name
master
Add additional tags
Application and OS Images (Amazon Machine Image) Info
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below
Search our full catalog including 1000s of application and OS images
Quick Start
Amazon Linux
macOS
Ubuntu
Windows
Red Hat
SUSE Li
Browse more AMIs
Including AMIs from AWS, Marketplace and the Community
Amazon Machine Image (AMI)
Ubuntu Server 24.04 LTS (HVM), SSD Volume Type
Free tier eligible
Summary
Number of instances Info
1
Software Image (AMI)
Canonical, Ubuntu, 24.04, amd6...read more
ami-0522ab6e1ddcc7055
Virtual server type (instance type)
t2.micro
Firewall (security group)
New security group
Storage (volumes)
1 volume(s) - 8 GiB
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public
Cancel
Launch instance

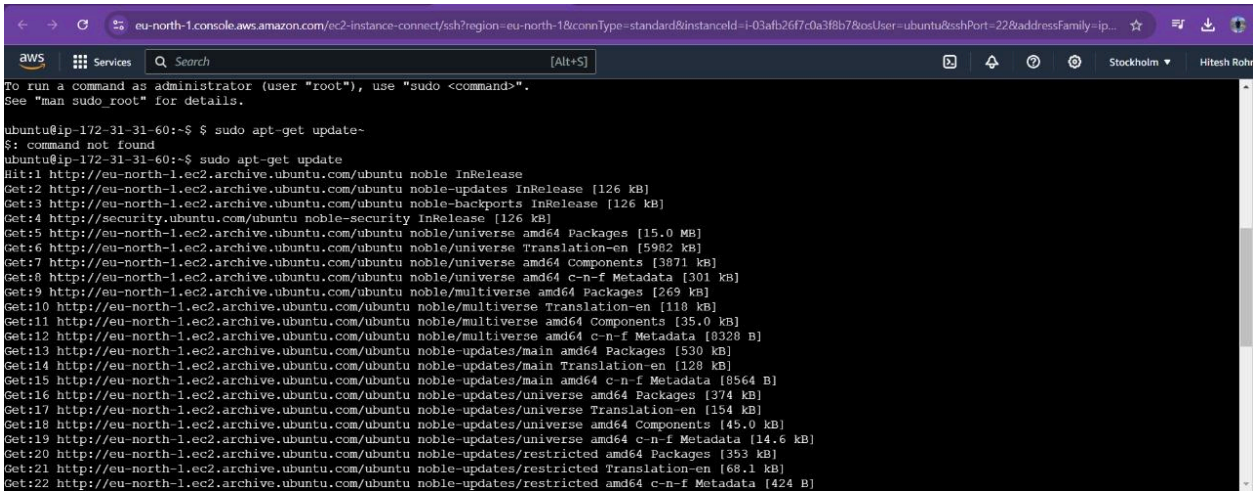

Create key pair
Key pair name
Key pairs allow you to connect to your instance securely.
three-tier-app
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.
Key pair type
RSA
RSA encrypted private and public key pair
ED25519
ED25519 encrypted private and public key pair
Private key file format
.pem
For use with OpenSSH
.ppk
For use with PuTTY
When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. Learn more
Cancel
Create key pair

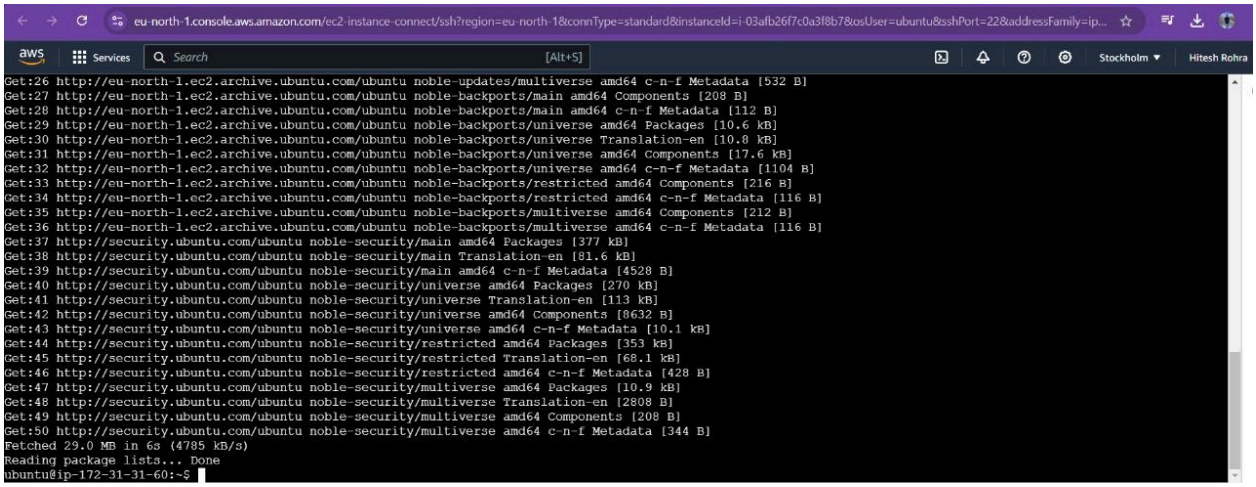


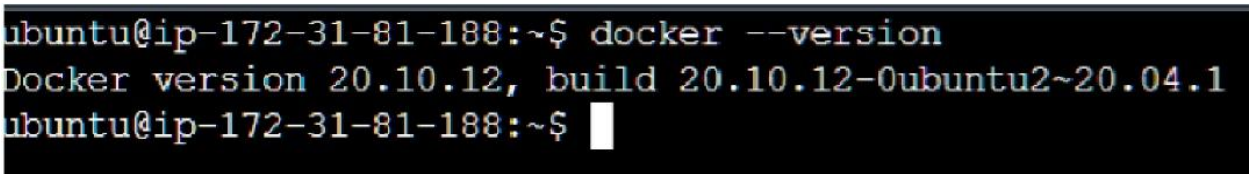
1.3 Add port 6443 in each security group



1.4 After the instances have been created,copy the text given in the example part of each of the three instances into git bash.







Step 2: Run the following commands on both the master and worker nodes to prepare them for kubeadm.

# disable swap
sudo swapoff -a

Create the .conf file to load the modules at bootup
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf

```
overlay
br_netfilter
EOF

sudo modprobe overlay
sudo modprobe br_netfilter

# sysctl params required by setup, params persist across reboots
cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-iptables  = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward                 = 1
EOF

# Apply sysctl params without reboot
sudo sysctl --system

## Install CRIO Runtime
sudo apt-get update -y
sudo apt-get install -y software-properties-common curl apt-transport-https
ca-certificates gpg

sudo curl -fsSL https://pkgs.k8s.io/addons:/cri-o:/prerelease:/main/deb/Release.key |
sudo gpg --dearmor -o /etc/apt/keyrings/cri-o-apt-keyring.gpg
echo "deb [signed-by=/etc/apt/keyrings/cri-o-apt-keyring.gpg]
https://pkgs.k8s.io/addons:/cri-o:/prerelease:/main/deb/ /" | sudo tee
/etc/apt/sources.list.d/cri-o.list

sudo apt-get update -y
sudo apt-get install -y cri-o

sudo systemctl daemon-reload
sudo systemctl enable crio --now
sudo systemctl start crio.service

echo "CRI runtime installed successfully"

# Add Kubernetes APT repository and install required packages
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.29/deb/Release.key | sudo gpg --dearmor
-o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
```

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.29/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

sudo apt-get update -y
sudo apt-get install -y kubelet="1.29.0-*" kubectl="1.29.0-*" kubeadm="1.29.0-*"
sudo apt-get update -y
sudo apt-get install -y jq

sudo systemctl enable --now kubelet
sudo systemctl start kubelet

```
ubuntu@ip-172-31-46-220:~$ # disable swap
sudo swapoff -a

# Create the .conf file to load the modules at bootup
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF

sudo modprobe overlay
sudo modprobe br_netfilter

# sysctl params required by setup, params persist across reboots
cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-iptables  = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward                 = 1
EOF

# Apply sysctl params without reboot
sudo sysctl --system

## Install CRIO Runtime
sudo apt-get update -y
sudo apt-get install -y software-properties-common curl apt-transport-https ca-certificates gpg

sudo curl -fsSL https://pkgs.k8s.io/addons:/cri-o:/prerelease:/main/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/cri-o-apt-keyring.gpg
echo "deb [signed-by=/etc/apt/keyrings/cri-o-apt-keyring.gpg] https://pkgs.k8s.io/addons:/cri-o:/prerelease:/main/deb/ /sudo systemctl start kubeletkubelet29.0-*" kubectl="1.29.0-*" kubeadm="1.29.0-*"k8s.io/co
re:/stable:/v1.29/deb/ /' | sud
overlay
br_netfilter
net.bridge.bridge-nf-call-iptables  = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward                 = 1
* Applying /usr/lib/sysctl.d/10-apparmor.conf ...
* Applying /etc/sysctl.d/10-console-messages.conf ...
* Applying /etc/sysctl.d/10-ipv6-privacy.conf ...
* Applying /etc/sysctl.d/10-kernel-hardening.conf ...
* Applying /etc/sysctl.d/10-magic-sysrq.conf ...
* Applying /etc/sysctl.d/10-map-count.conf ...
* Applying /etc/sysctl.d/10-network-security.conf ...
* Applying /etc/sysctl.d/10-ptrace.conf ...
* Applying /etc/sysctl.d/10-zeropage.conf ...
* Applying /etc/sysctl.d/50-cloudimg-settings.conf ...
* Applying /usr/lib/sysctl.d/50-pid-max.conf ...
* Applying /etc/sysctl.d/99-cloudimg-ipv6.conf ...
* Applying /usr/lib/sysctl.d/99-protect-links.conf ...
* Applying /etc/sysctl.d/99-sysctl.conf ...
```

Step3: Run the above command only on master node

sudo kubeadm config images pull

sudo kubeadm init

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

Network Plugin = calico
kubectl apply -f
https://raw.githubusercontent.com/projectcalico/calico/v3.26.0/manifests/calico.yaml

kubeadm token create --print-join-command

```
ubuntu@ip-172-31-46-220:~$ sudo kubeadm config images pull

sudo kubeadm init

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


# Network Plugin = calico
kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.26.0/manifests/calico.yaml

kubeadm token create --print-join-command
I0921 11:12:21.776309    3963 version.go:256] remote version is much newer: v1.31.0; falling back to: stable-1.29
[config/images] Pulled registry.k8s.io/kube-apiserver:v1.29.9
[config/images] Pulled registry.k8s.io/kube-controller-manager:v1.29.9
[config/images] Pulled registry.k8s.io/kube-scheduler:v1.29.9
[config/images] Pulled registry.k8s.io/kube-proxy:v1.29.9
[config/images] Pulled registry.k8s.io/coredns/coredns:v1.11.1
[config/images] Pulled registry.k8s.io/pause:3.9
[config/images] Pulled registry.k8s.io/etcd:3.5.10-0
I0921 11:12:40.995686    4304 version.go:256] remote version is much newer: v1.31.0; falling back to: stable-1.29
[init] Using Kubernetes version: v1.29.9
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
W0921 11:12:41.763411    4304 checks.go:835] detected that the sandbox image "registry.k8s.io/pause:3.10" of the container runtime is inconsistent with that used by kubeadm. It is recommended that using "regis
try.k8s.io/pause:3.9" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-46-220 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.46.220]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-46-220 localhost] and IPs [172.31.46.220 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-46-220 localhost] and IPs [172.31.46.220 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file
[kubeconfig] Writing "scheduler.conf" kubeconfig file
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"
```

You will get kubeadm token, Copy it.

Step 4: Run the above command only on worker nodes

sudo kubeadm reset pre-flight checks
sudo your-token --v=5

```
ubuntu@ip-172-31-36-212:~$ sudo kubeadm reset pre-flight checks
W0921 11:14:17.713660    3933 preflight.go:56] [reset] WARNING: Changes made to this host by 'kubeadm init' or 'kubeadm join' will be reverted.
[reset] Are you sure you want to proceed? [y/N]: yes
[preflight] Running pre-flight checks
W0921 11:14:20.535200    3933 removeetcdmember.go:106] [reset] No kubeadm config, using etcd pod spec to get data directory
[reset] Deleted contents of the etcd data directory: /var/lib/etcd
[reset] Stopping the kubelet service
[reset] Unmounting mounted directories in "/var/lib/kubelet"
[reset] Deleting contents of directories: [/etc/kubernetes/manifests /var/lib/kubelet /etc/kubernetes/pki]
[reset] Deleting files: [/etc/kubernetes/admin.conf /etc/kubernetes/super-admin.conf /etc/kubernetes/kubelet.conf /etc/kubernetes/bootstrap-kubelet.conf /etc/kubernetes/controller-manager.conf /etc/kubernetes/
scheduler.conf]

The reset process does not clean CNI configuration. To do so, you must remove /etc/cni/net.d

The reset process does not reset or clean up iptables rules or IPVS tables.
If you wish to reset iptables, you must do so manually by using the "iptables" command.

If your cluster was setup to utilize IPVS, run ipvsadm --clear (or similar)
to reset your system's IPVS tables.

The reset process does not clean your kubeconfig files and you must remove them manually.
```

```
ubuntu@ip-172-31-36-212:~$ sudo kubeadm join 172.31.46.220:6443 --token k4psyh.ns1g1yet9he59kd4 --discovery-token-ca-cert-hash sha256:80e7e9abf8f31f0333a9d2f7a680d6bdf961267ae45cf71bada8de069d4a292e --v=5
I0921 11:28:31.063878    4097 join.go:413] [preflight] found NodeName empty; using OS hostname as NodeName
I0921 11:28:31.064085    4097 initconfiguration.go:122] detected and using CRI socket: unix:///var/run/crio/crio.sock
[preflight] Running pre-flight checks
I0921 11:28:31.064143    4097 preflight.go:93] [preflight] Running general checks
I0921 11:28:31.064183    4097 checks.go:280] validating the existence of file /etc/kubernetes/kubelet.conf
I0921 11:28:31.064207    4097 checks.go:280] validating the existence of file /etc/kubernetes/bootstrap-kubelet.conf
I0921 11:28:31.064219    4097 checks.go:104] validating the container runtime
I0921 11:28:31.089669    4097 checks.go:639] validating whether swap is enabled or not
I0921 11:28:31.089763    4097 checks.go:370] validating the presence of executable crictl
I0921 11:28:31.089799    4097 checks.go:370] validating the presence of executable conntrack
I0921 11:28:31.089819    4097 checks.go:370] validating the presence of executable ip
I0921 11:28:31.089843    4097 checks.go:370] validating the presence of executable iptables
I0921 11:28:31.089870    4097 checks.go:370] validating the presence of executable mount
I0921 11:28:31.089897    4097 checks.go:370] validating the presence of executable nsenter
I0921 11:28:31.089919    4097 checks.go:370] validating the presence of executable ebtables
I0921 11:28:31.089954    4097 checks.go:370] validating the presence of executable ethtool
I0921 11:28:31.089977    4097 checks.go:370] validating the presence of executable socat
I0921 11:28:31.089996    4097 checks.go:370] validating the presence of executable tc
I0921 11:28:31.090011    4097 checks.go:370] validating the presence of executable touch
I0921 11:28:31.090035    4097 checks.go:516] running all checks
I0921 11:28:31.103935    4097 checks.go:401] checking whether the given node name is valid and reachable using net.LookupHost
I0921 11:28:31.105638    4097 checks.go:605] validating kubelet version
I0921 11:28:31.162593    4097 checks.go:130] validating if the "kubelet" service is enabled and active
I0921 11:28:31.176512    4097 checks.go:203] validating availability of port 10250
I0921 11:28:31.176737    4097 checks.go:280] validating the existence of file /etc/kubernetes/pki/ca.crt
I0921 11:28:31.176765    4097 checks.go:430] validating if the connectivity type is via proxy or direct
I0921 11:28:31.176803    4097 checks.go:329] validating the contents of file /proc/sys/net/bridge/bridge-nf-call-iptables
I0921 11:28:31.176849    4097 checks.go:329] validating the contents of file /proc/sys/net/ipv4/ip_forward
I0921 11:28:31.176883    4097 join.go:532] [preflight] Discovering cluster-info
I0921 11:28:31.176917    4097 token.go:80] [discovery] Created cluster-info discovery client, requesting info from "172.31.46.220:6443"
I0921 11:28:31.187676    4097 token.go:118] [discovery] Requesting info from "172.31.46.220:6443" again to validate TLS against the pinned public key
I0921 11:28:31.194531    4097 token.go:135] [discovery] Cluster info signature and contents are valid and TLS certificate validates against pinned roots, will use API Server "172.31.46.220:6443"
I0921 11:28:31.194600    4097 discovery.go:52] [discovery] Using provided TLSBootstrapToken as authentication credentials for the join process
I0921 11:28:31.194622    4097 join.go:546] [preflight] Fetching init configuration
I0921 11:28:31.194629    4097 join.go:592] [preflight] Retrieving KubeConfig objects
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
I0921 11:28:31.201909    4097 kubeproxy.go:55] attempting to download the KubeProxyConfiguration from ConfigMap "kube-proxy"
I0921 11:28:31.205146    4097 kubelet.go:74] attempting to download the KubeletConfiguration from ConfigMap "kubelet-config"
I0921 11:28:31.208370    4097 initconfiguration.go:114] skip CRI socket detection, fill with the default CRI socket unix:///var/run/containerd/containerd.sock
I0921 11:28:31.208595    4097 interface.go:432] Looking for default routes with IPv4 addresses
I0921 11:28:31.208617    4097 interface.go:437] Default route transits interface "enX0"
I0921 11:28:31.208751    4097 interface.go:209] Interface enX0 is up
I0921 11:28:31.208803    4097 interface.go:257] Interface "enX0" has 2 addresses :[172.31.36.212/20 fe80::75:41ff:fea5:afb1/64].
I0921 11:28:31.208819    4097 interface.go:224] Checking addr  172.31.36.212/20.
I0921 11:28:31.208829    4097 interface.go:231] IP found 172.31.36.212
I0921 11:28:31.208840    4097 interface.go:263] Found valid IPv4 address 172.31.36.212 for interface "enX0".
I0921 11:28:31.208849    4097 interface.go:443] Found active IP 172.31.36.212
I0921 11:28:31.215002    4097 preflight.go:104] [preflight] Running configuration dependant checks
I0921 11:28:31.215028    4097 controlplaneprepare.go:225] [download-certs] Skipping certs download
```

Step5: Run the given command to verify cluster creation

kubectl get nodes

```
ubuntu@ip-172-31-46-220:~$ kubectl get nodes
NAME               STATUS   ROLES           AGE   VERSION
ip-172-31-36-212   Ready    <none>          47s   v1.29.0
ip-172-31-46-220   Ready    control-plane   16m   v1.29.0
ip-172-31-47-26    Ready    <none>          29s   v1.29.0
```

Experiment No 4

Hitesh Rohra **EXPERIMENT 4**
D15A 47
Step 1: Deploying Your Application on Kubernetes
1.1 Set up Kubernetes Cluster

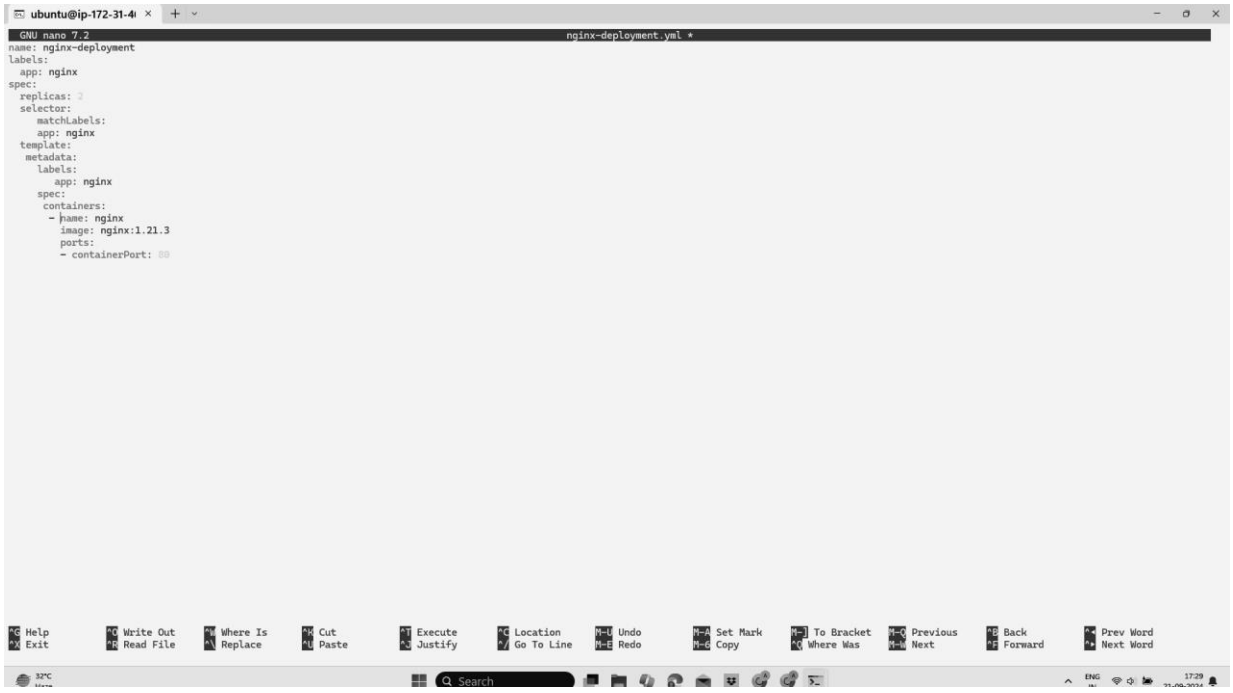
1. If you haven't already set up a Kubernetes cluster (e.g., with kubeadm), use minikube or any managed Kubernetes service (like EKS, GKE, etc.) to get a cluster running.

2. Once your cluster is ready, verify the nodes:

kubectl get nodes

```
ubuntu@ip-172-31-46-220:~$ kubectl get nodes
NAME               STATUS   ROLES           AGE   VERSION
ip-172-31-36-212   Ready    <none>          47s   v1.29.0
ip-172-31-46-220   Ready    control-plane   16m   v1.29.0
ip-172-31-47-26    Ready    <none>          29s   v1.29.0
```

Step 2: Create the Deployment YAML file
a) Create the YAML file: Use a text editor to create a file named nginx-deployment.yaml Add the Deployment Configuration: Copy and paste the following YAML content into the file. Save and exit the editor (Press Ctrl+X, then Y, and Enter).



Step 3: Create the Service YAML File
a) Create the YAML File: Create another file named nginx-service.yaml Add the Service Configuration: Copy and paste the following YAML content into the file given below

Step 4: Apply the YAML Files a) Deploy the Application: Use kubectl to create the Deployment and Service from the YAML files. Verify the Deployment: Check the status of your Deployment,Pods and Services. Describe the deployment(Extra)

```
ubuntu@ip-172-31-46-220:~$ kubectl apply -f nginx-deployment.yaml
deployment.apps/nginx-deployment created
```

```
ubuntu@ip-172-31-46-220:~$ kubectl apply -f nginx-service.yaml
service/nginx-server created
```

Step 5: Ensure Service is Running 6.1 Verify Service: Run the following command to check the services running in your cluster: Kubectl get deployment Kubectl get pods kubectl get service

```
error: the server doesn't have a resource type "deployments"
ubuntu@ip-172-31-46-220:~$ kubectl get deployments
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   3/3     3            3           7m27s
```

```
ubuntu@ip-172-31-46-220:~$ kubectl get services
NAME           TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
kubernetes     ClusterIP      10.96.0.1        <none>        443/TCP        85m
nginx-server   LoadBalancer   10.111.218.213   <pending>     80:30798/TCP   110s
```

Step 6: Forward the Service Port to Your Local Machine kubectl port-forward allows you to forward a port from your local machine to a port on a service running in the Kubernetes cluster.

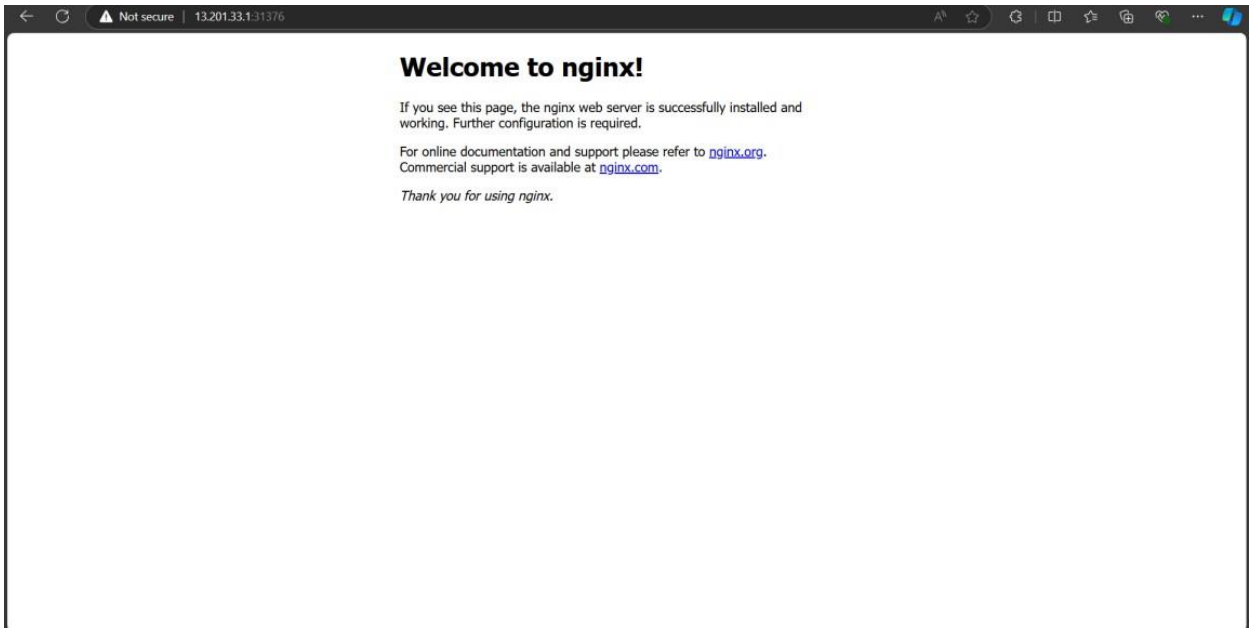
1. Forward the Service Port: Use the following command to forward a local port to the service's target port. kubectl port-forward service/ :
This command will forward local port 8080 on your machine to port 80 of the service nginx-service running inside the cluster.

```
ubuntu@ip-172-31-46-220:~$ kubectl describe deployments
Name:                   nginx-deployment
Namespace:              default
CreationTimestamp:      Sat, 21 Sep 2024 12:30:54 +0000
Labels:                 app=nginx
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               app=nginx
Replicas:               3 desired | 3 updated | 3 total | 3 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginx
  Containers:
   nginx:
    Image:        nginx:1.16
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  <none>
NewReplicaSet:   nginx-deployment-854bc88786 (3/3 replicas created)
Events:
  Type    Reason             Age   From                   Message
  ----    ------             ----  ----                   -------
  Normal  ScalingReplicaSet  11m   deployment-controller  Scaled up replica set nginx-deployment-854bc88786 to 3
```

2. This means port forwarding is now active, and any traffic to localhost:8080 will be routed to the nginx-service on port 80.

```
ubuntu@ip-172-31-46-220:~$ kubectl port-forward service/nginx-server 8080:80
```

Step 7:
Access the Application Locally
1. Open a Web Browser: Now open your web browser and go to the following URL: http://localhost:8080 You should see the application (in this case, Nginx) that you have deployed running in the Kubernetes cluster, served locally via port 8080. In case the port 8080 is unavailable, try using a different port like 8081


Not secure | 13.201.33.1:31376
Welcome to nginx!
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.
Thank you for using nginx.

Experiment No 5

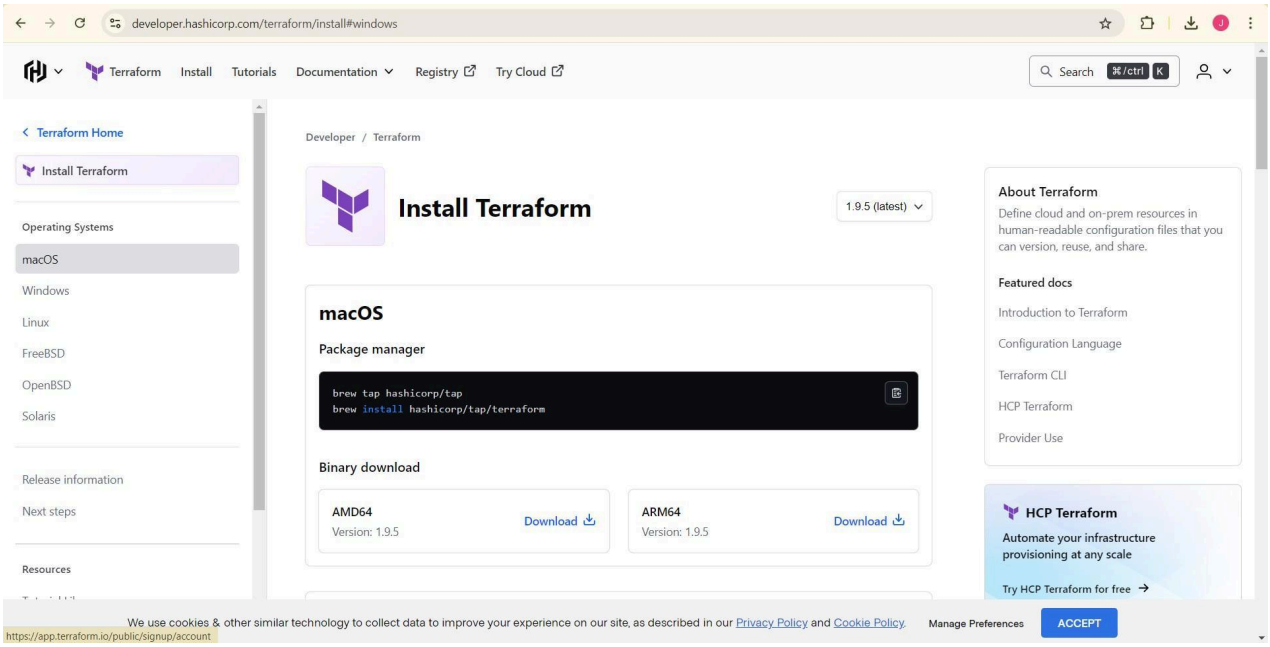
Advance devops Exp:5

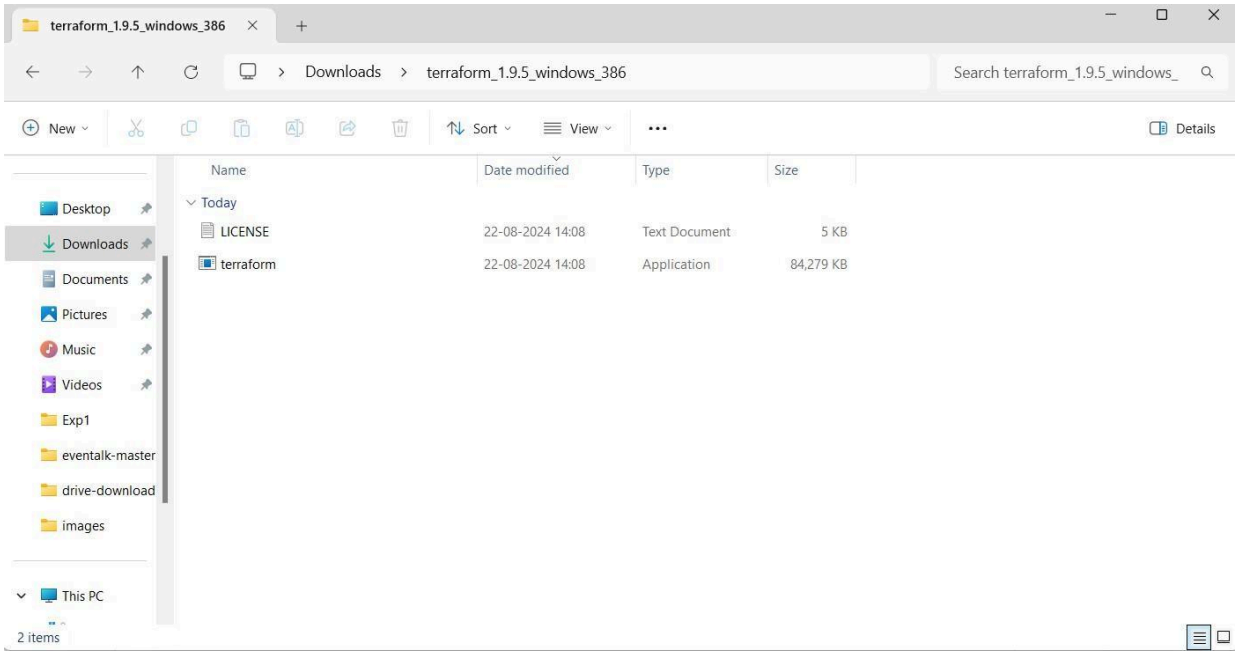
Hitesh Rohra

D15 A-47

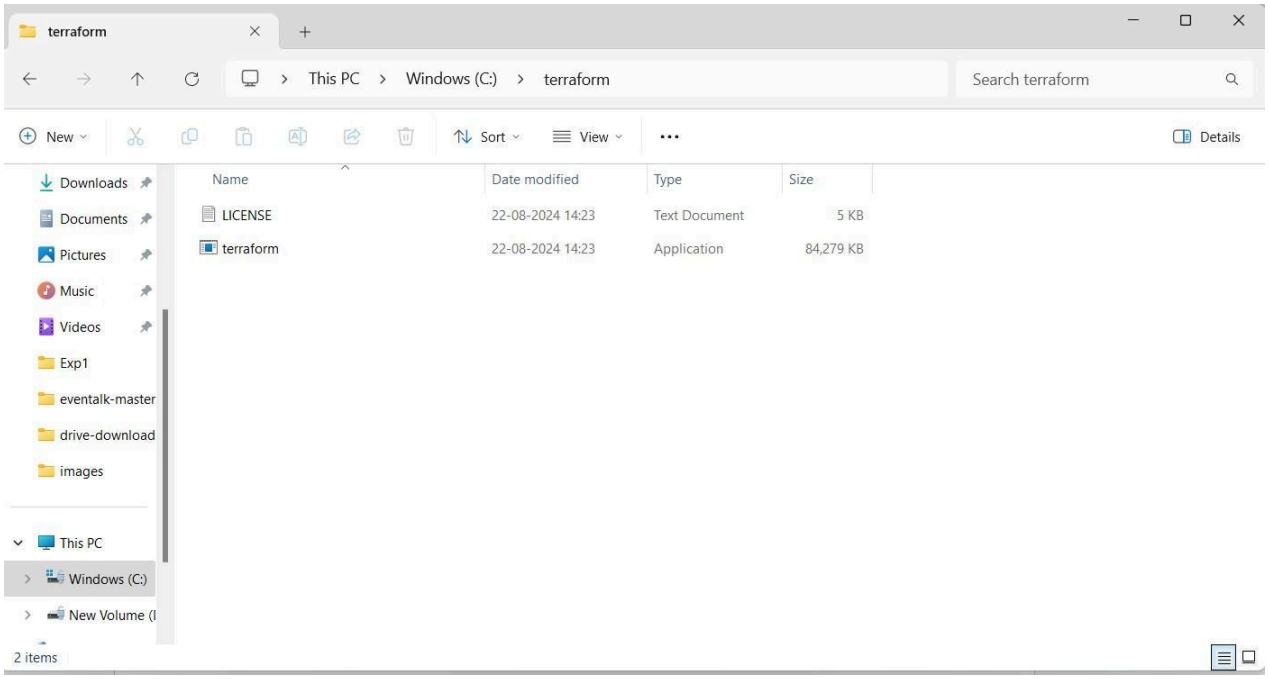
Aim: To understand terraform lifecycle, core concepts/terminologies and install it on a linux machine and windows

Step1: Download Terraform from the official website

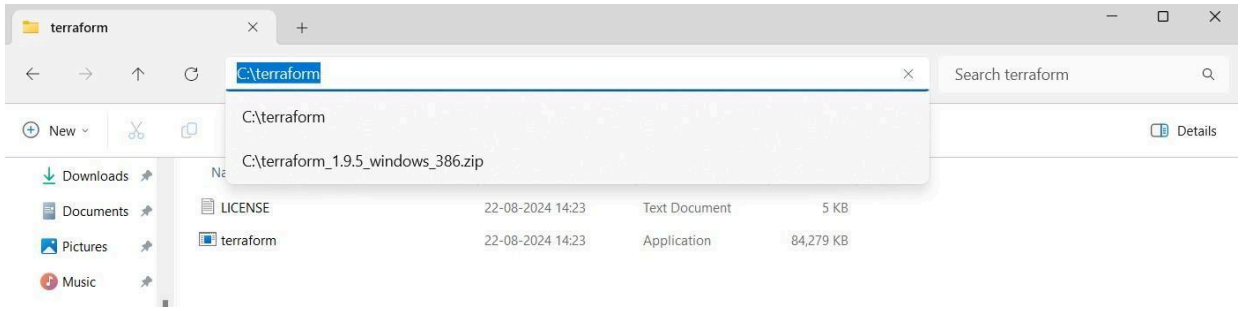




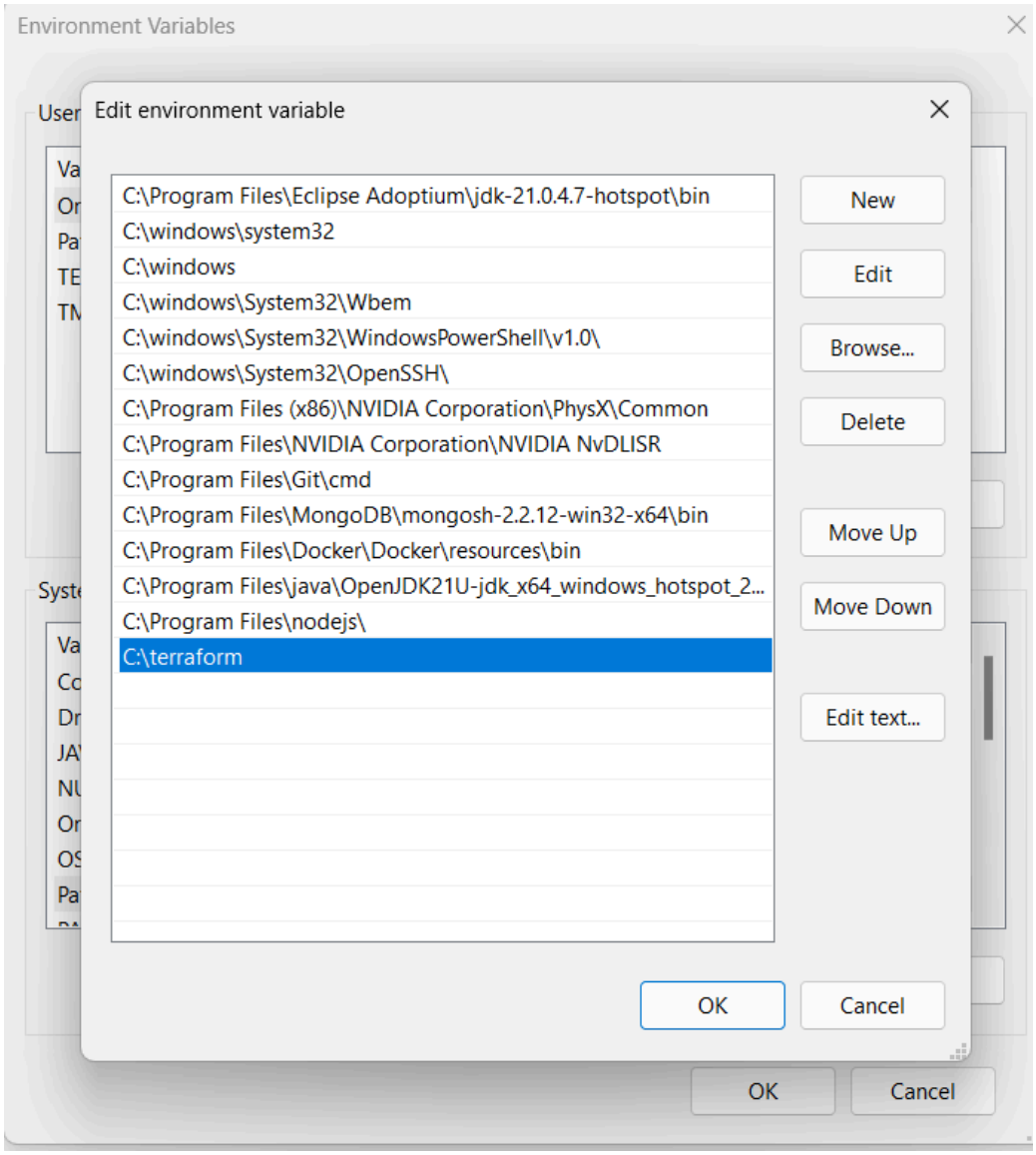
Step 2: Copy and extract Terraform from the downloads and paste it in the C drive



Step 3: Copy the file path to paste in the environment variables



Step 4: Set the environment variables for terraform



Step 5: Check weather the terraform is installed

```
Microsoft Windows [Version 10.0.22631.4037]
(c) Microsoft Corporation. All rights reserved.

C:\Users\navan>terraform --version
Terraform v1.9.5
on windows_386
```

Experiment No 6

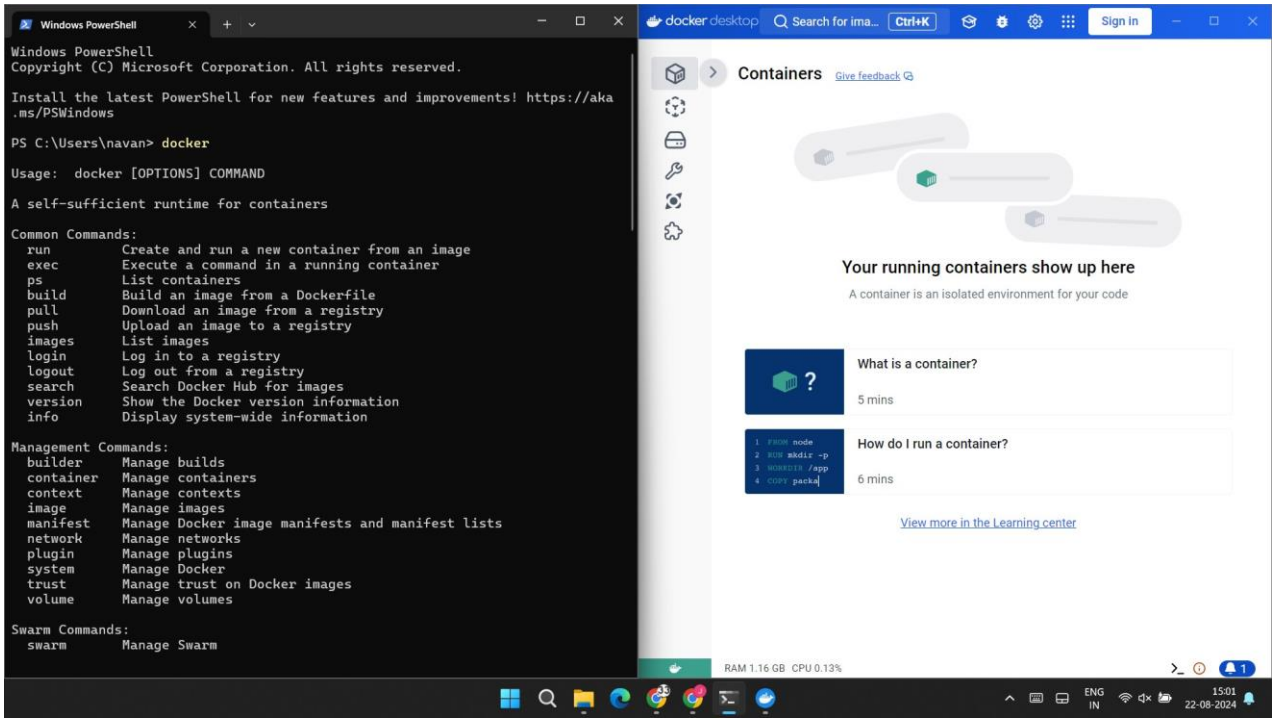
Advance devops Exp:5

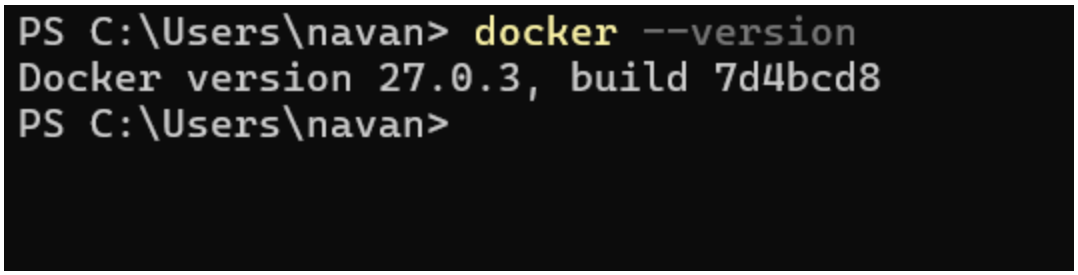
Hitesh Rohra

D15 A-47

Aim: Creating docker image using Terraform

Step 1: Install docker Desktop after installation check the functionality



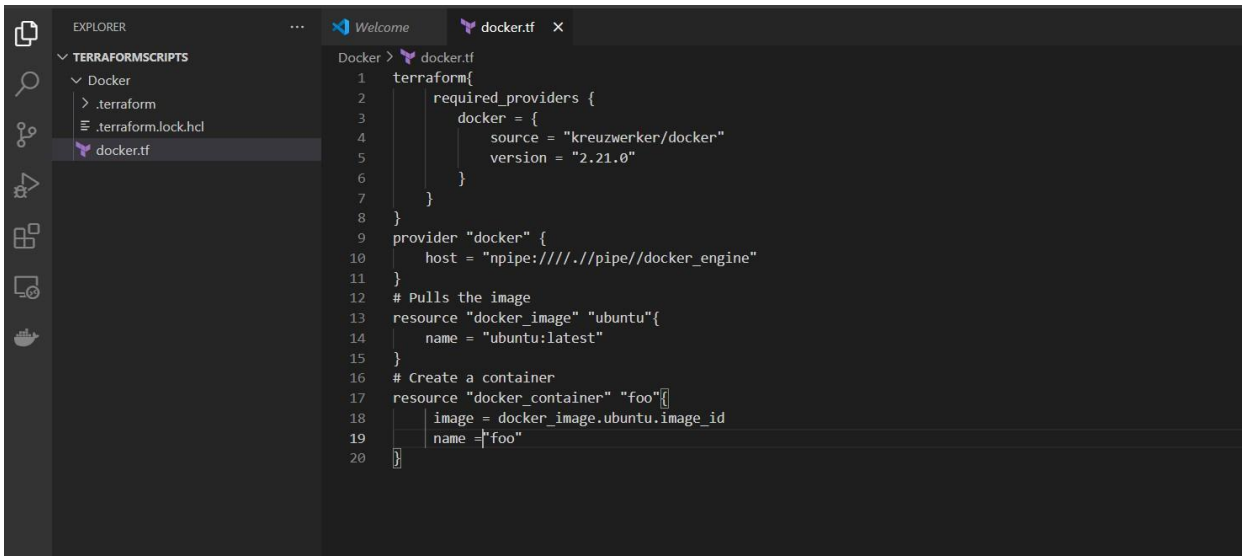


Now, create a folder named ‘Terraform Scripts’ in which we save our different types of scripts which will be further used in this experiment.

Step 2: Firstly create a new folder named ‘Docker’ in the ‘TerraformScripts’ folder. Then create a new docker.tf file using Atom editor and write the followingcontents into it to create a Ubuntu Linux container.

Script:

```
terraform{
    required_providers {
      docker = {
          source = "kreuzwerker/docker"
          version = "2.21.0"
      }
   }
}
provider "docker" {
   host = "npipe:////.//pipe//docker_engine"
}
# Pulls the image
resource "docker_image" "ubuntu"{
   name = "ubuntu:latest"
}
# Create a container
resource "docker_container" "foo"{
    image = docker_image.ubuntu.image_id
    name ="foo"
}
```



Step 3: Execute terraform init command to initialize the resources

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

Terraform has been successfully initialized!

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

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

C:\Users\navan\Desktop\TerraformScripts\Docker>
```

Step 4: Execute Terraform plan to see the available resources

```
C:\Users\navan\Desktop\TerraformScripts\Docker>terraform plan

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

Terraform will perform the following actions:

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

```
      + security_opts    = (known after apply)
      + shm_size         = (known after apply)
      + start            = true
      + stdin_open       = false
      + stop_signal      = (known after apply)
      + stop_timeout     = (known after apply)
      + tty              = false

      + healthcheck (known after apply)

      + labels (known after apply)
    }

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

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

─────────────────────────────────────────────────────────────────────────────────────────────────────────────

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

C:\Users\navan\Desktop\TerraformScripts\Docker>
```

Step 5: Execute Terraform apply to apply the configuration, which will automatically
create and run the Ubuntu Linux container based on our configuration. Using command :
“terraform apply”

```
    }

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

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

  Enter a value: yes

docker_image.ubuntu: Creating...
docker_image.ubuntu: Creation complete after 10s [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Creating...
```

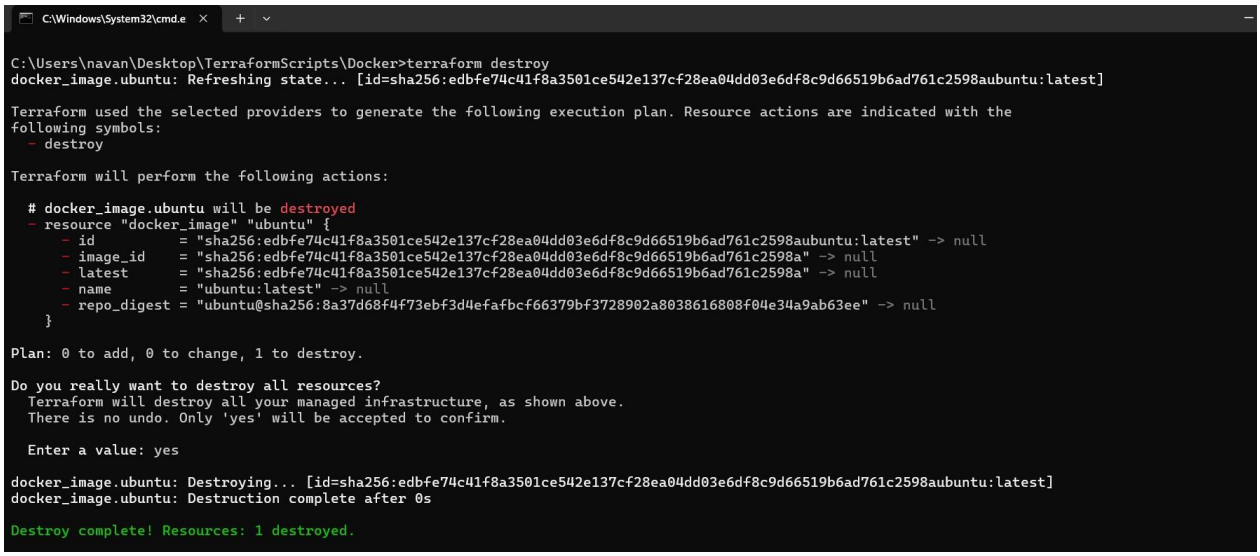
Docker images,before Executing Apply step:

```
C:\Users\navan\Desktop\TerraformScripts\Docker>docker images
REPOSITORY   TAG          IMAGE ID       CREATED       SIZE
ubuntu       latest       edbfe74c41f8   2 weeks ago   78.1MB
node         20-alpine    e2997a3fdff8   4 weeks ago   133MB
```

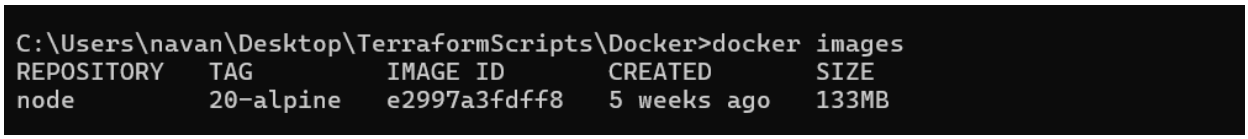
```
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28e
3e6df8c9d66519b6ad761c2598aubuntu:latest]

Note: Objects have changed outside of Terraform

Terraform detected the following changes made outside of Terraform since the last
"terraform apply" which may have affected this plan:

  # docker_image.ubuntu has been deleted
  - resource "docker_image" "ubuntu" {
        id          = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6
2598aubuntu:latest"
      - image_id    = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6
2598a" -> null
        name        = "ubuntu:latest"
        # (2 unchanged attributes hidden)
    }


Unless you have made equivalent changes to your configuration, or ignored the relev
attributes using ignore_changes, the following plan may include actions to undo or
respond to these changes.
```

Step 6: Execute Terraform destroy to delete the configuration, which will automatically delete the ubuntu container.

```
C:\Users\navan\Desktop\TerraformScripts\Docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c259
tu:latest]

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

Terraform will perform the following actions:

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

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

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

  Enter a value:
```



Docker images after executing destroy step



Experiment No 7

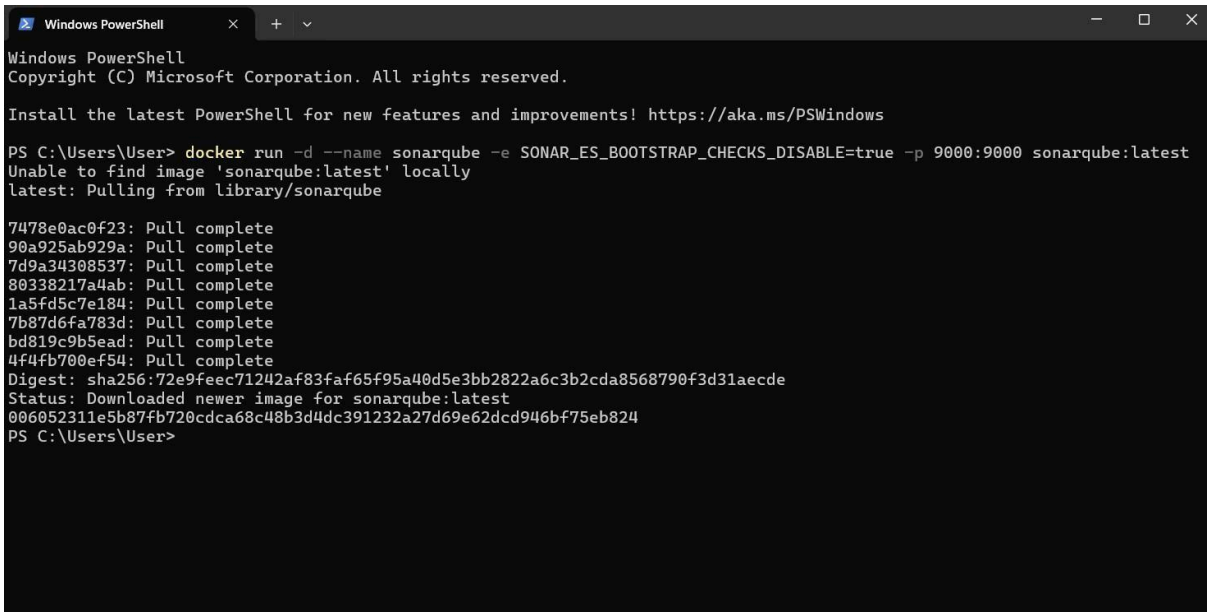
Hitesh Rohra
D15A
47

EXPERIMENT 7

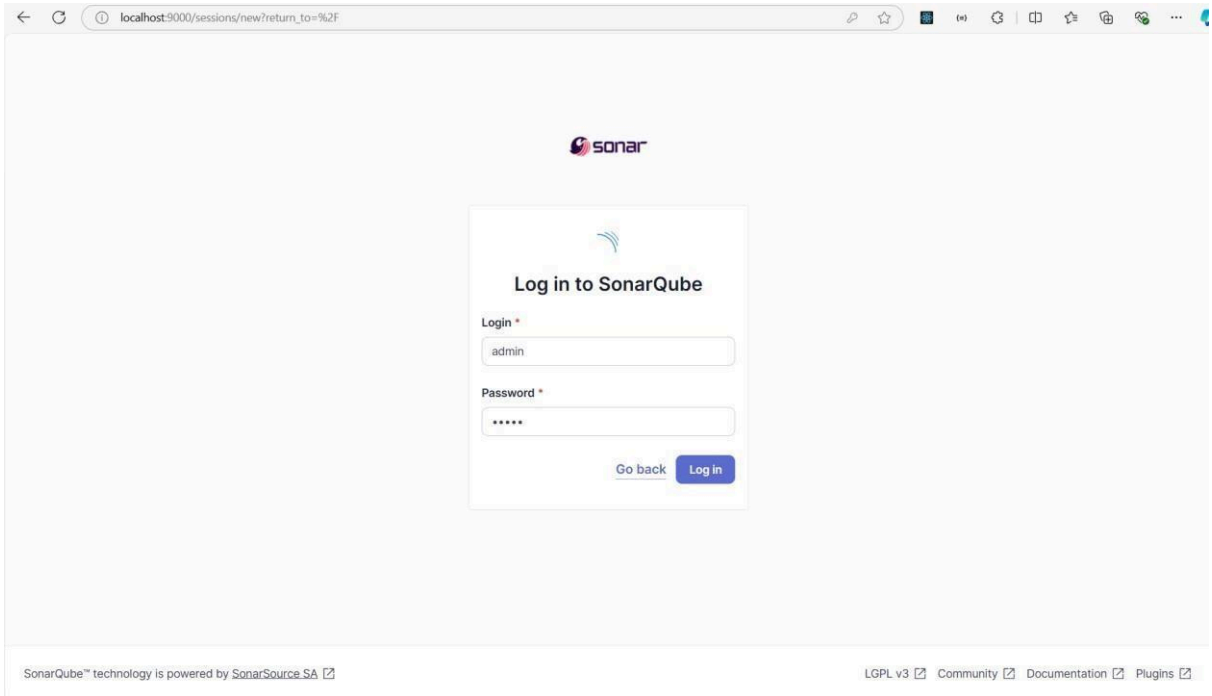
Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

Step 1: Open Windows PowerShell and run the following command –

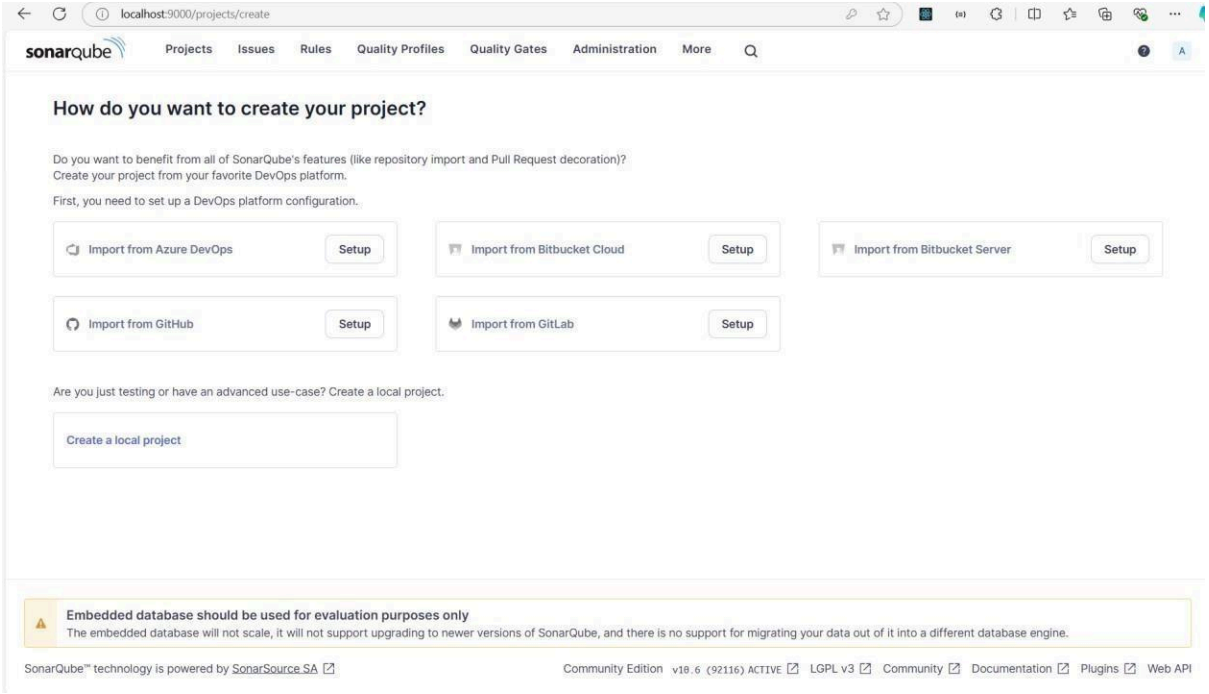
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest WARNING: Run the following command only once

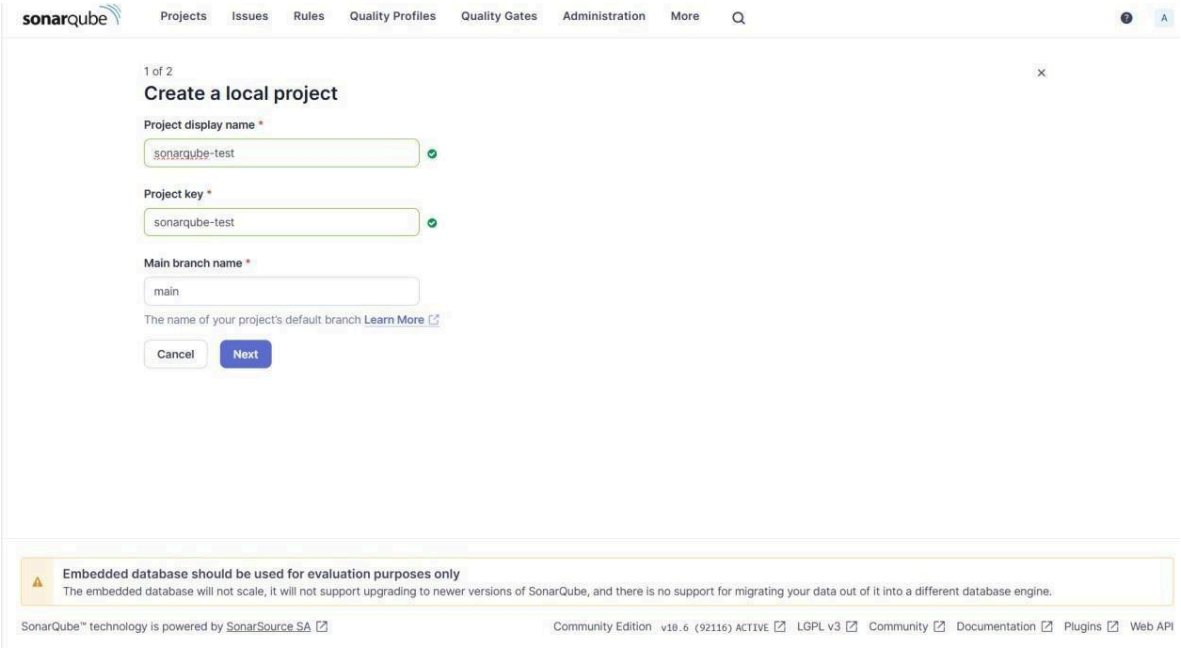


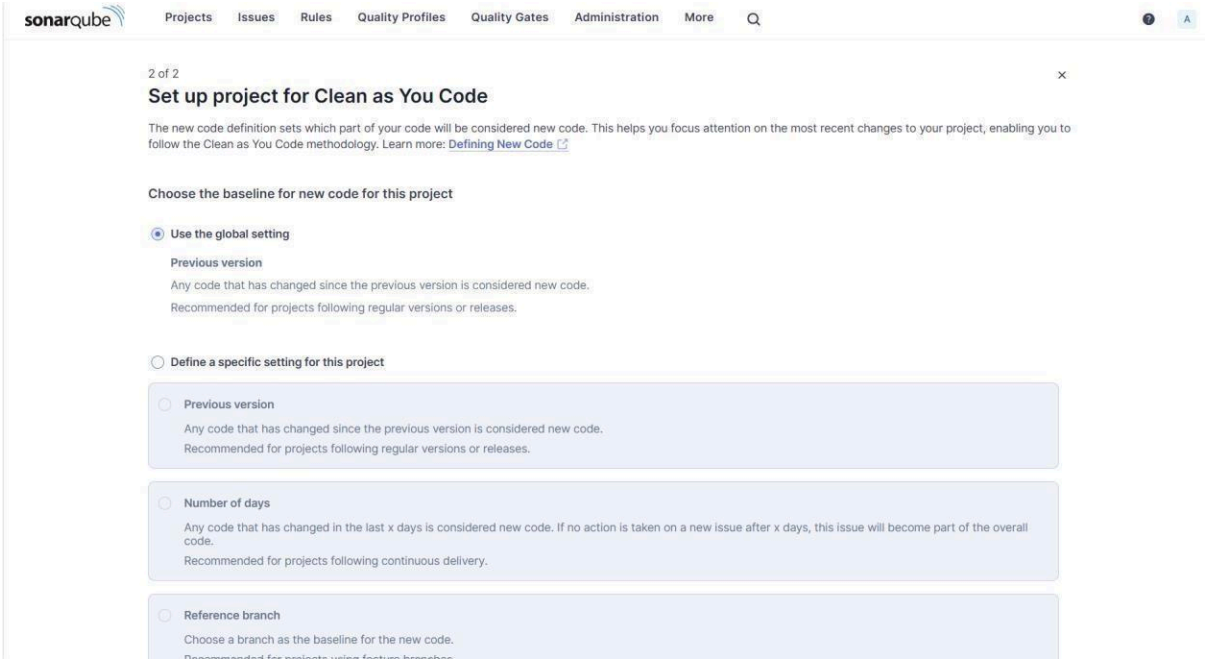
Step 2: Visit http://localhost:9000/ to open SonarQube. Login with username: admin and password: admin.

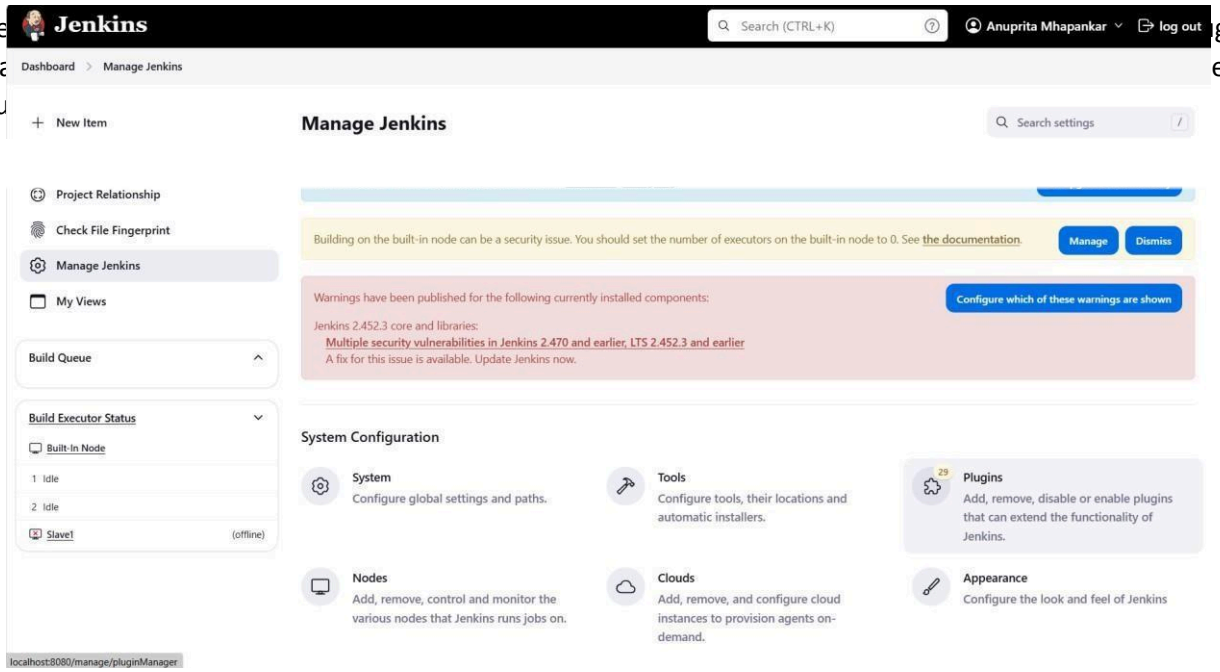


Step 3: Click on create a local project and name the project as sonarqube-test and key as sonarqube-test and click on the next button. In the next step select the “Use the global setting” option and click on create project.

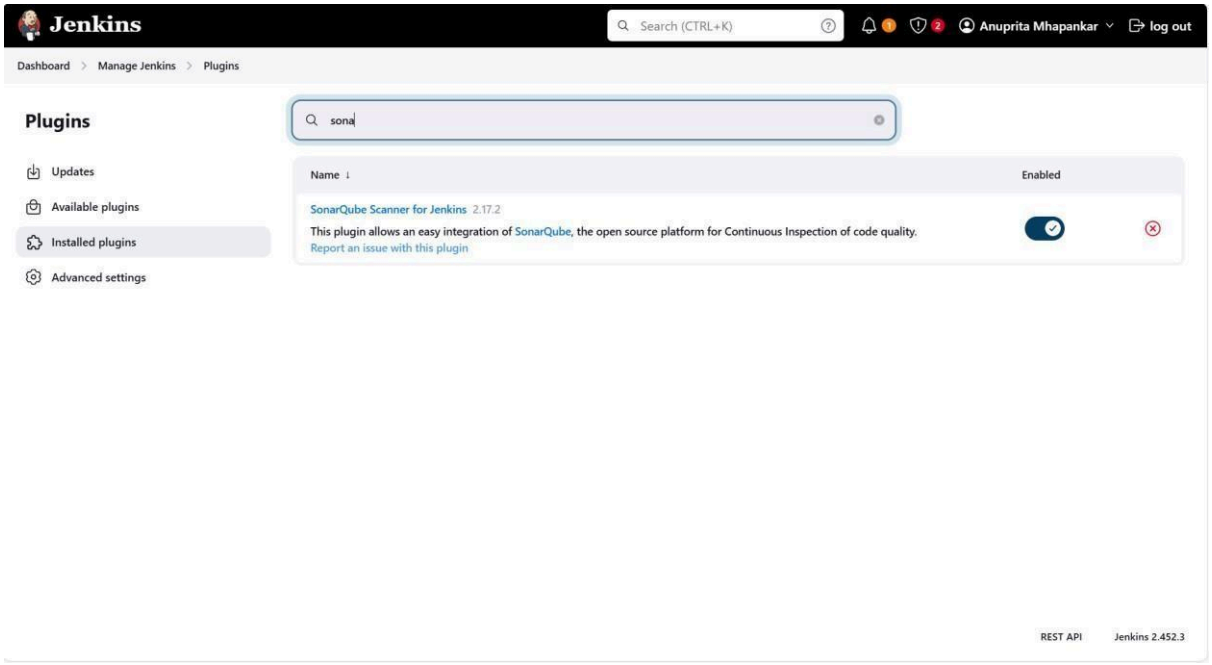

localhost:9000/projects/create
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
How do you want to create your project?
Do you want to benefit from all of SonarQube's features (like repository import and Pull Request decoration)?
Create your project from your favorite DevOps platform.
First, you need to set up a DevOps platform configuration.
Import from Azure DevOps
Setup
Import from Bitbucket Cloud
Setup
Import from Bitbucket Server
Setup
Import from GitHub
Setup
Import from GitLab
Setup
Are you just testing or have an advanced use-case? Create a local project.
Create a local project
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.
SonarQube™ technology is powered by SonarSource SA
Community Edition v10.6 (92116) ACTIVE
LGPL v3
Community
Documentation
Plugins
Web API


sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
1 of 2
Create a local project
Project display name *
sonarqube-test
Project key *
sonarqube-test
Main branch name *
main
The name of your project's default branch Learn More
Cancel
Next
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.
SonarQube™ technology is powered by SonarSource SA
Community Edition v10.6 (92116) ACTIVE
LGPL v3
Community
Documentation
Plugins
Web API

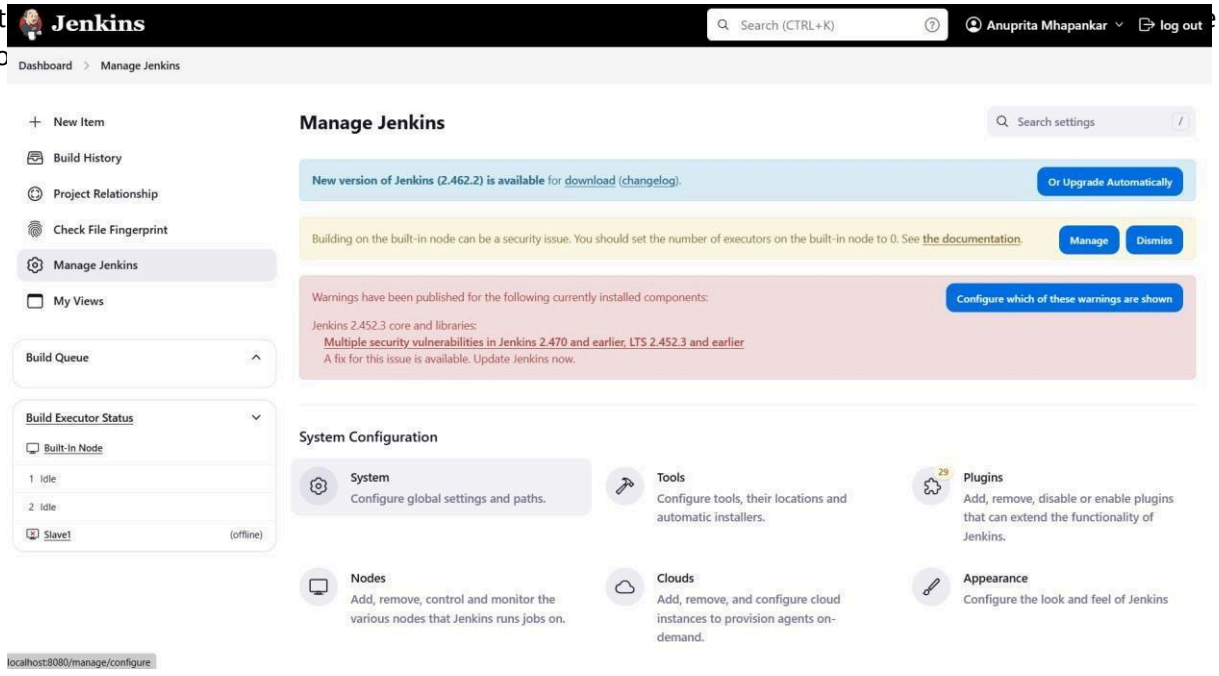


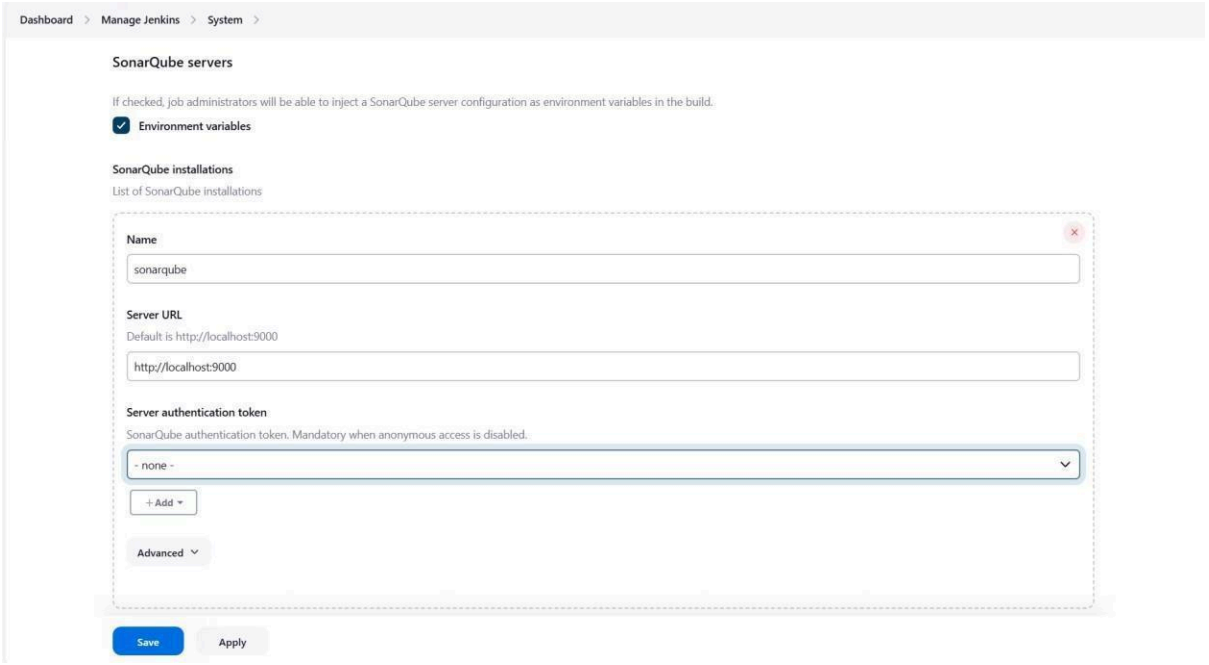
Ste 

gins and select ava ew the installed plu

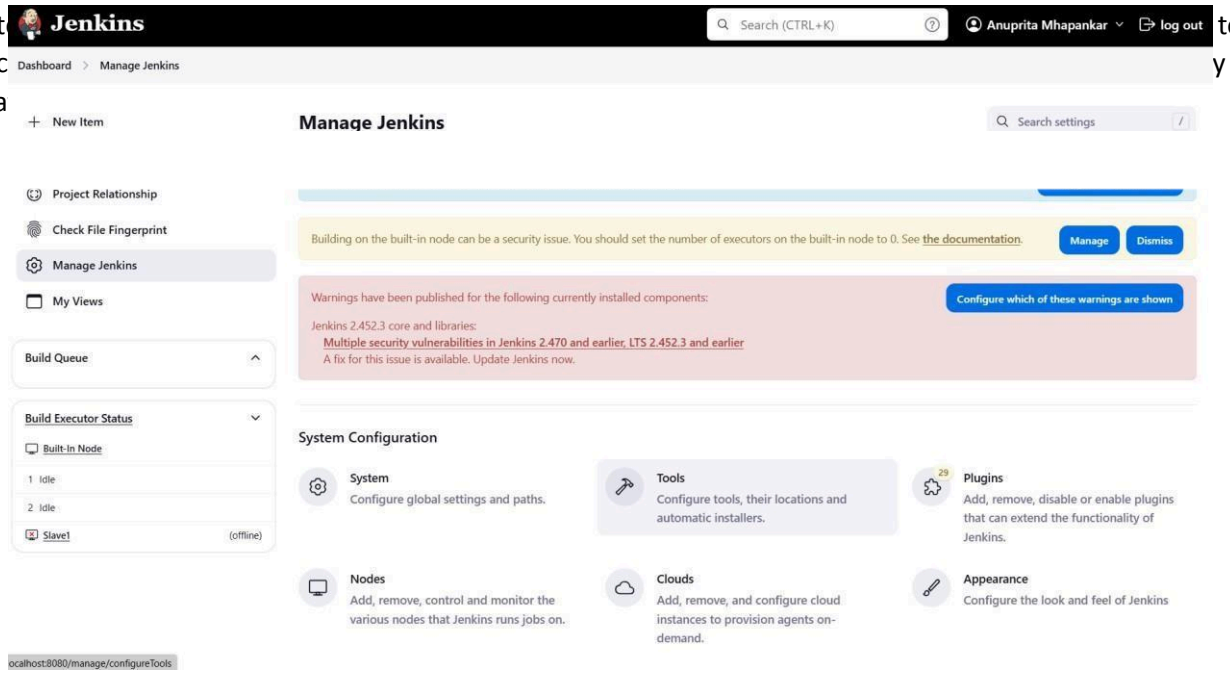


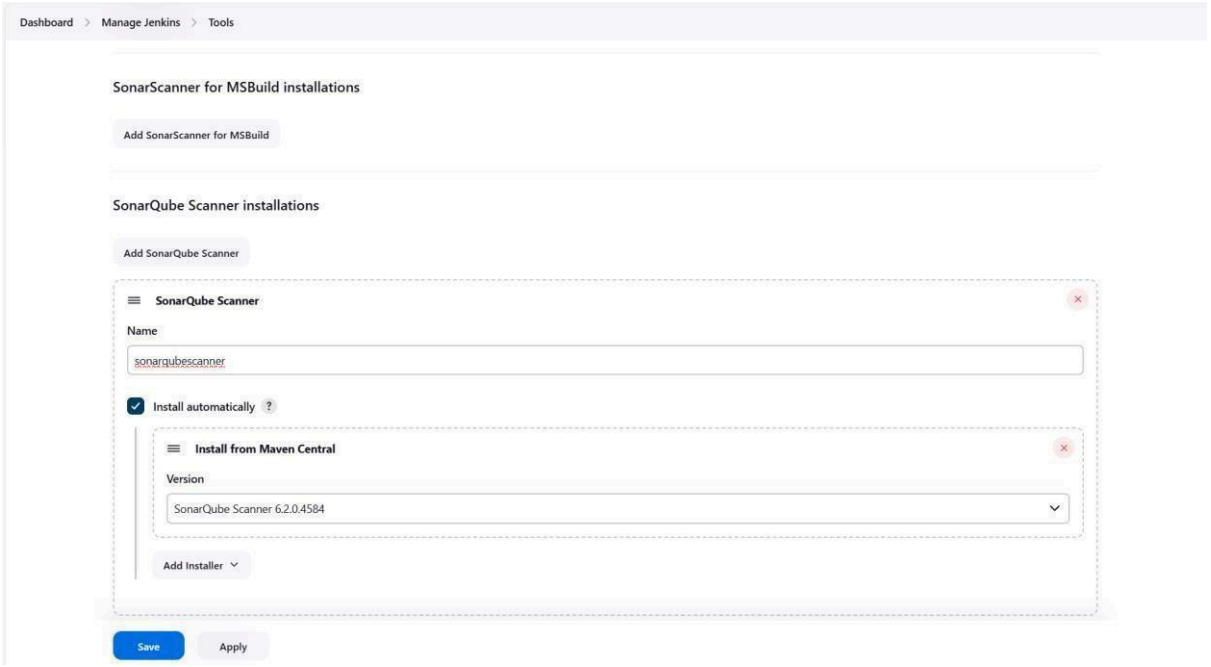
St the server as so



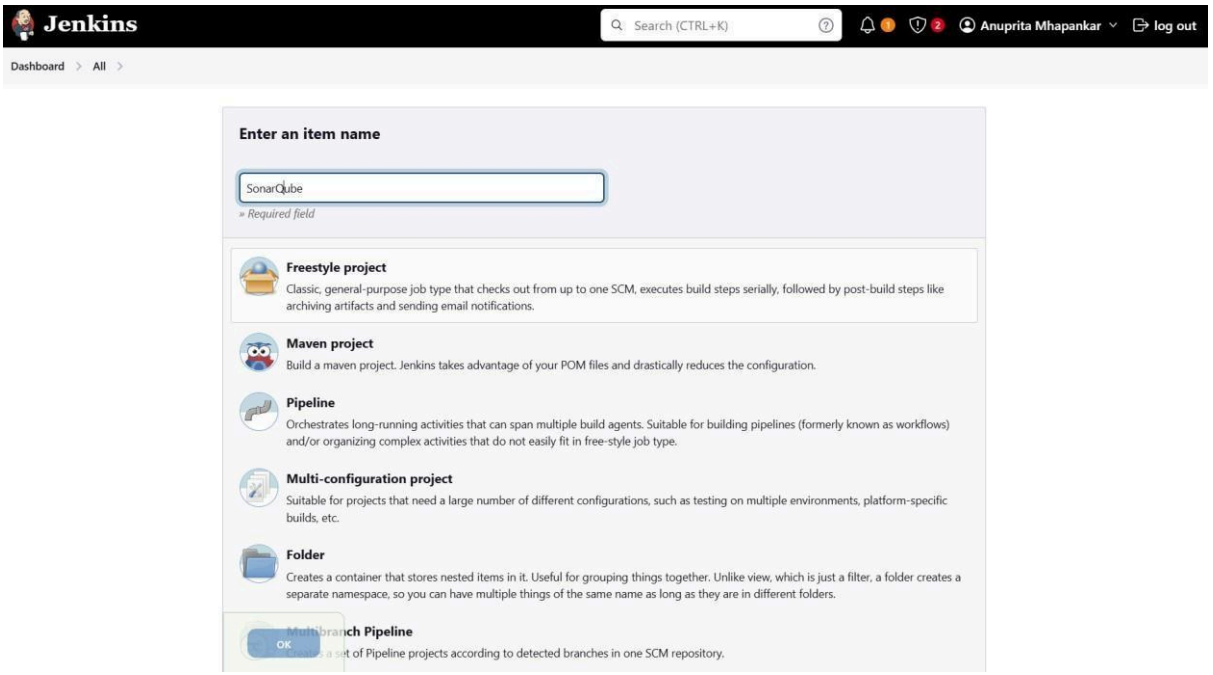


St to SonarQube
Sc y then click on
sa

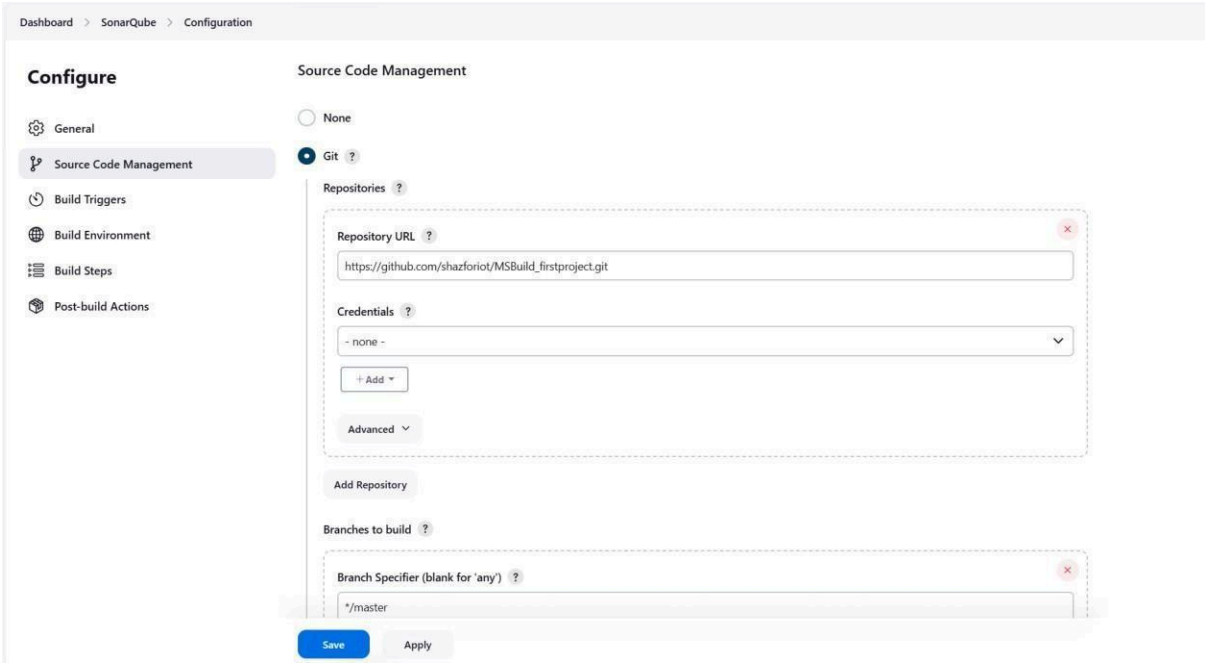




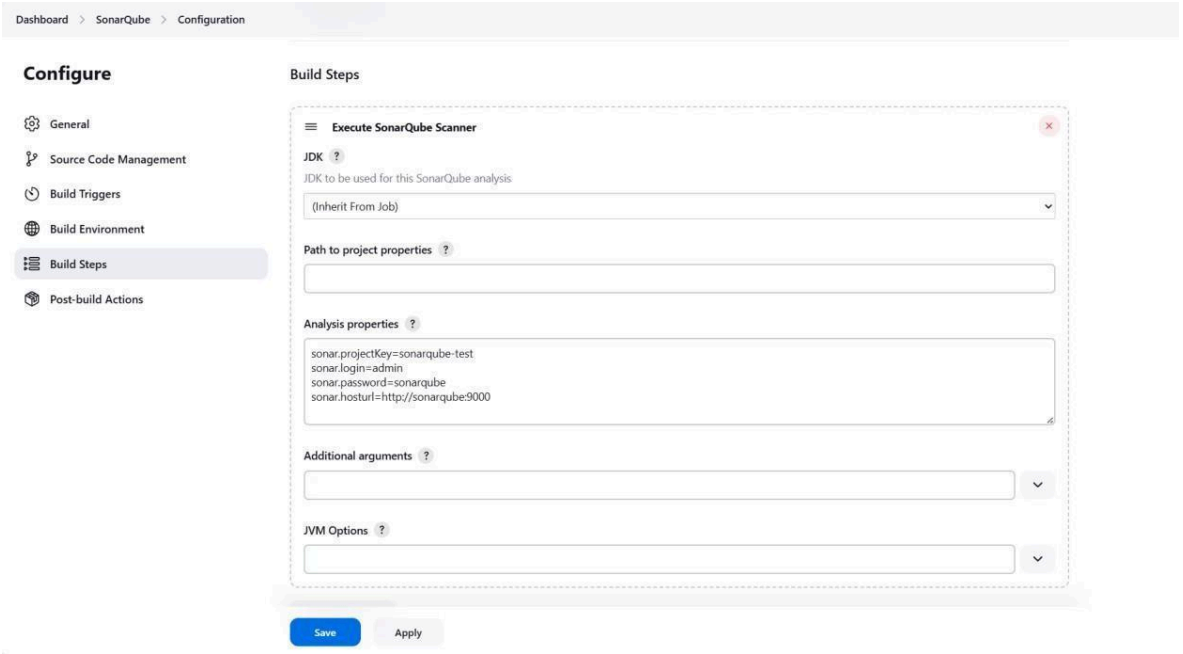
Step 7: Go to Jenkins dashboard and click on New Item and select Freestyle project and name it as SonarQube and then click on ok.

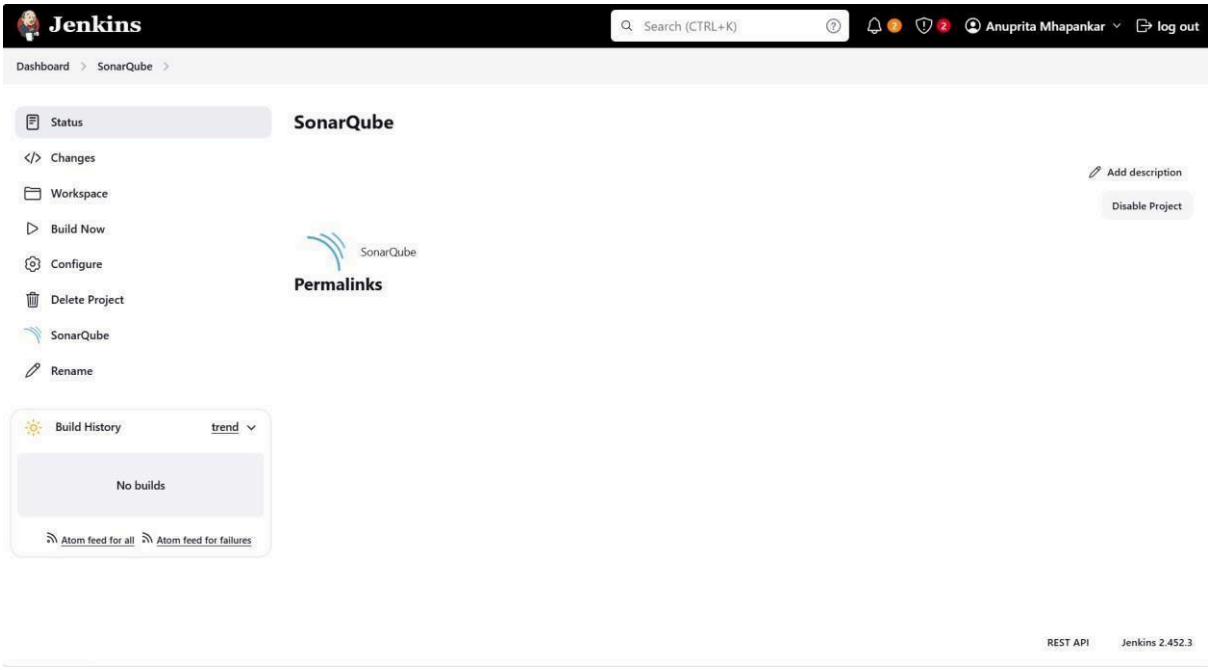


Step 8: For configuration, Select git and paste the following git repository in the repository url.
https://github.com/shazforiot/MSBuild_firstproject
This is a simple Hello world project

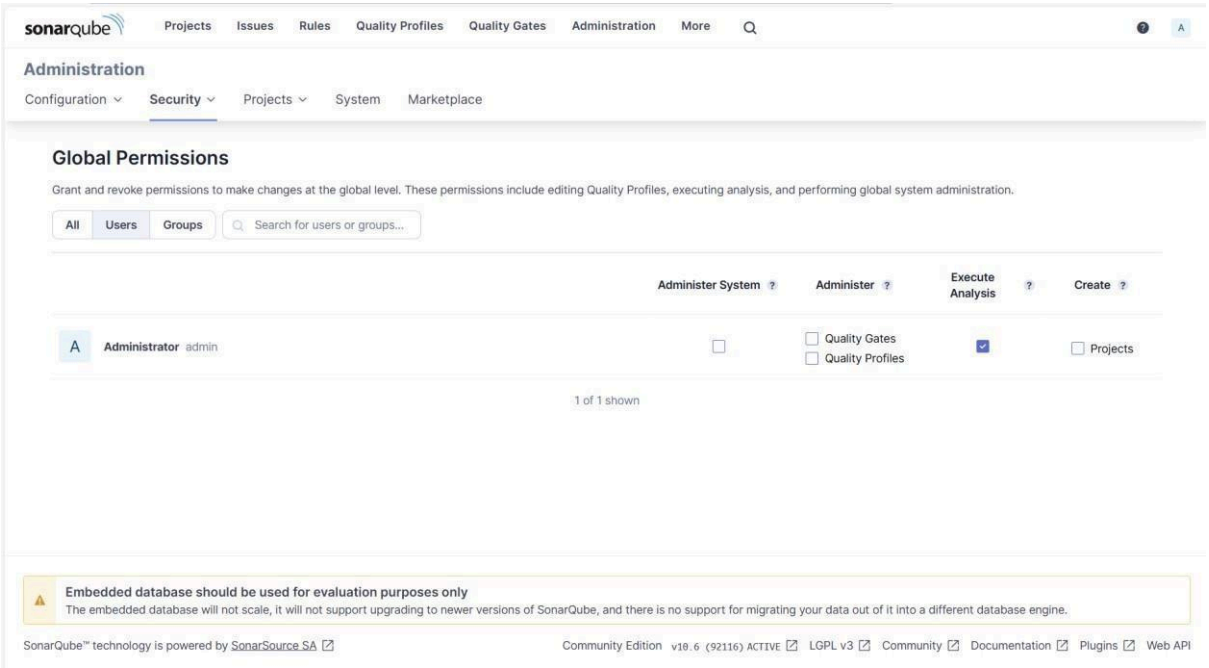


Step 9: Under the Build steps select “Execute SonarQube Scanner” option and under Analysis Properties write the following -
sonar.projectKey=sonarqube-test
sonar.login=admin
sonar.password=sonarqube
sonar.hosturl=http://sonarqube:900
0 Then click on the save button.

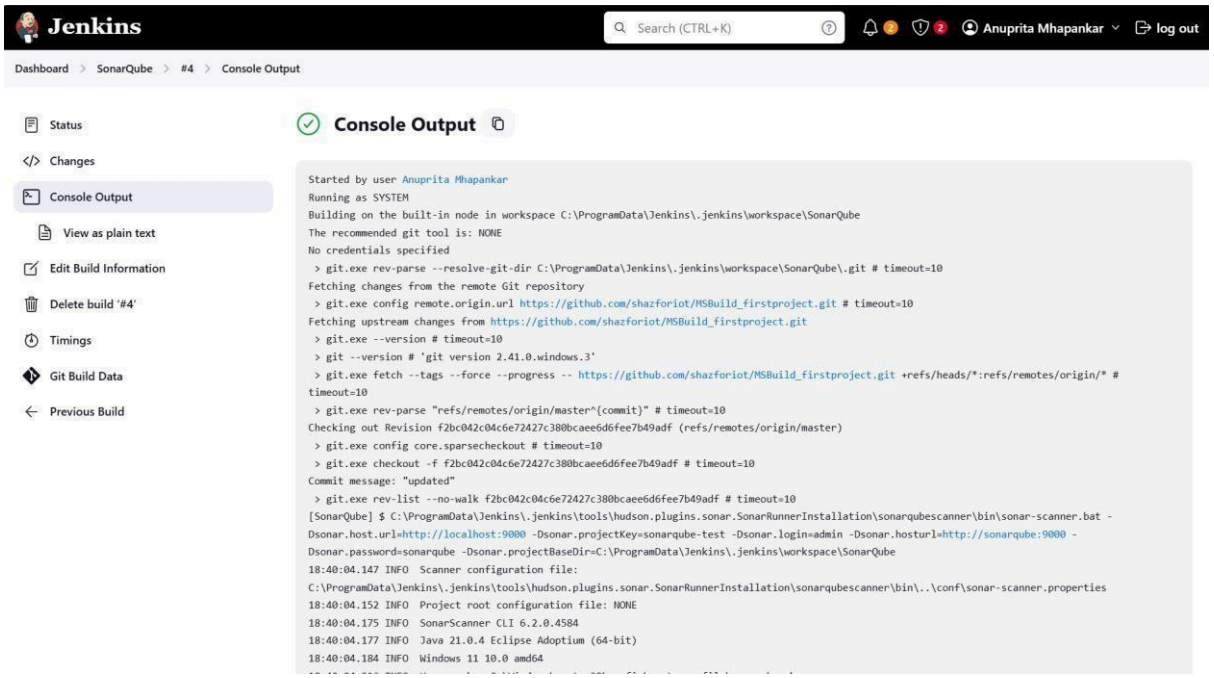


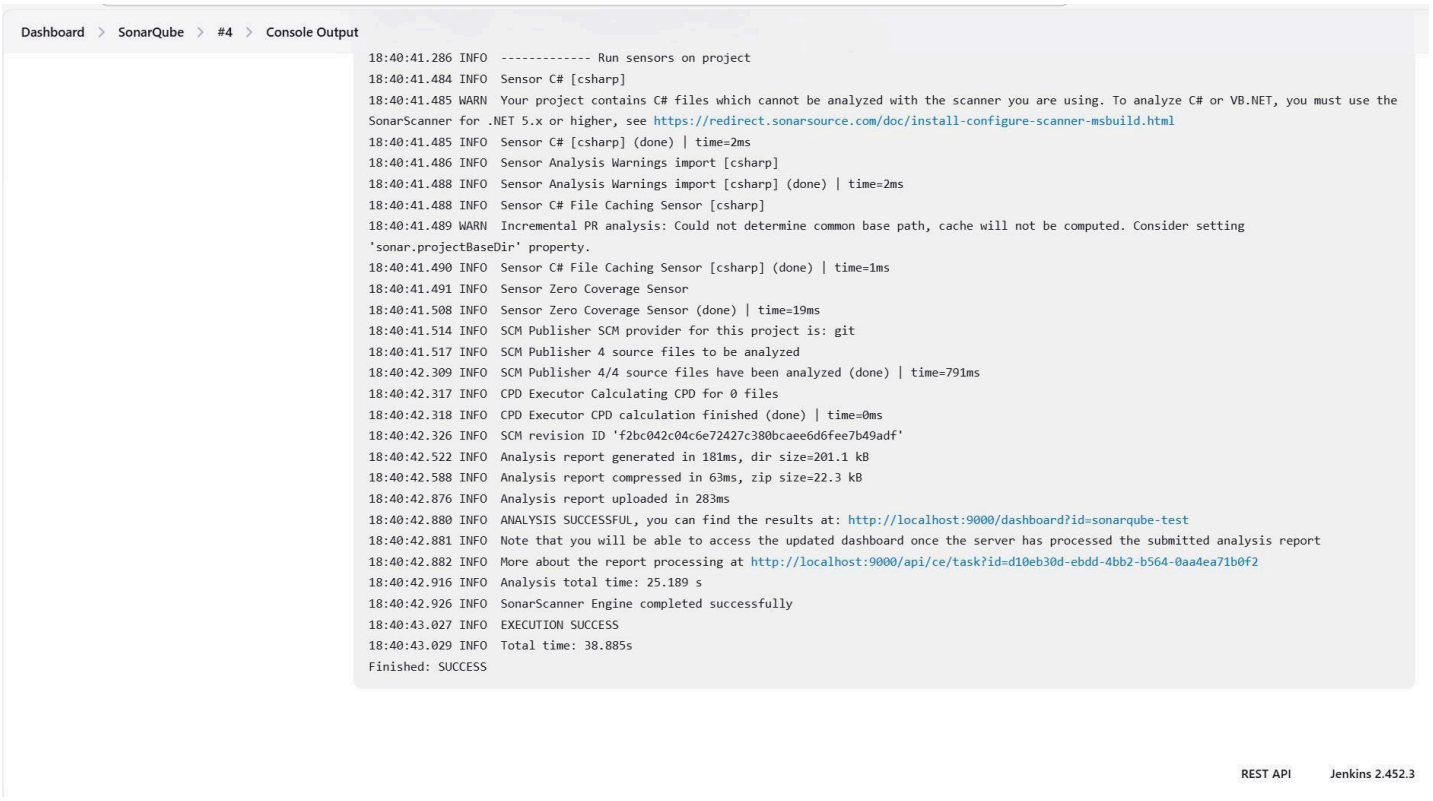


Step 10: Visit http://localhost:9000/admin/permissions and select the Users tab and for Administrator select the checkbox Execute Analysis.

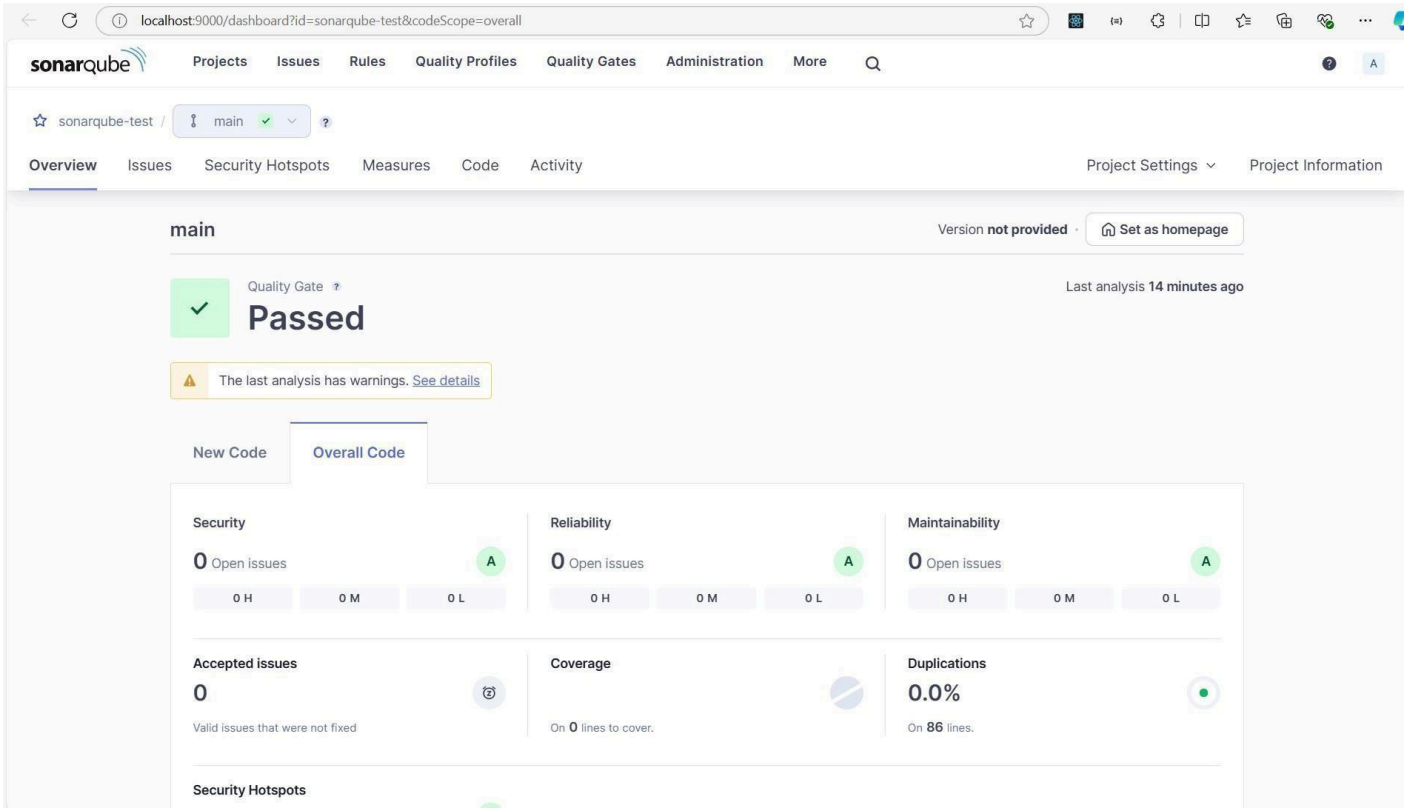


Step 11: Now, come back to Jenkins and click on Build Now. The build is success.





Step 12: Visit the following URL to see the result - http://localhost:9000/dashboard?id=sonarqube-test&codeScope=overall

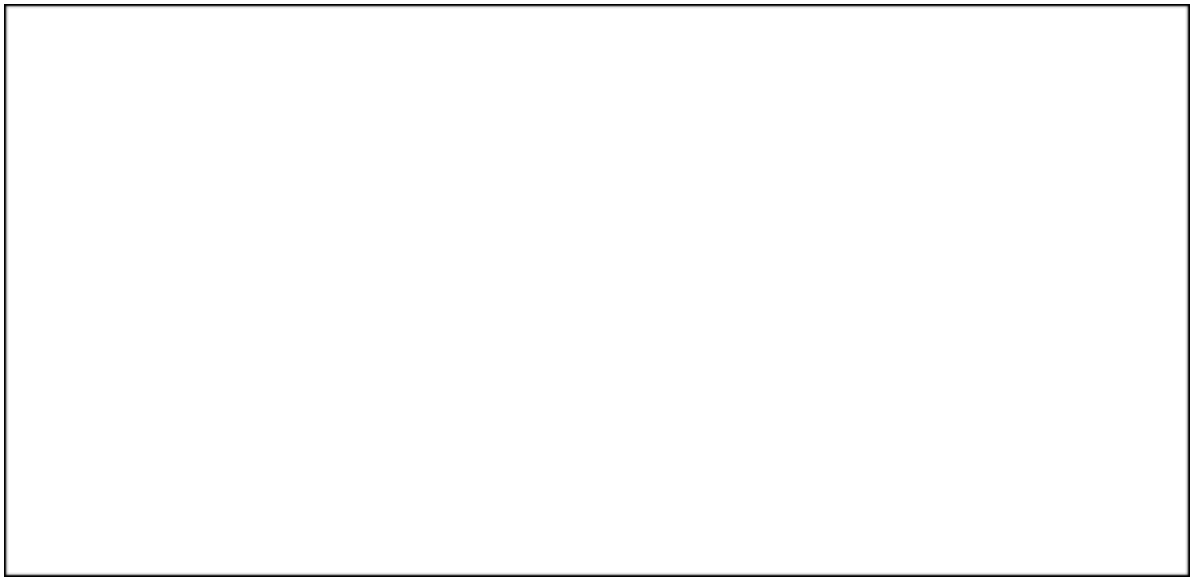

localhost:9000/dashboard?id=sonarqube-test&codeScope=overall
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
sonarqube-test
main
Overview
Issues
Security Hotspots
Measures
Code
Activity
Project Settings
Project Information
main
Version not provided
Set as homepage
Quality Gate
Passed
Last analysis 14 minutes ago
The last analysis has warnings. See details
New Code
Overall Code
Security
0 Open issues
A
0 H
0 M
0 L
Reliability
0 Open issues
A
0 H
0 M
0 L
Maintainability
0 Open issues
A
0 H
0 M
0 L
Accepted issues
0
Valid issues that were not fixed
Coverage
On 0 lines to cover.
Duplications
0.0%
On 86 lines.
Security Hotspots

Experiment No 8

D15-A
Hitesh Rohra - 47

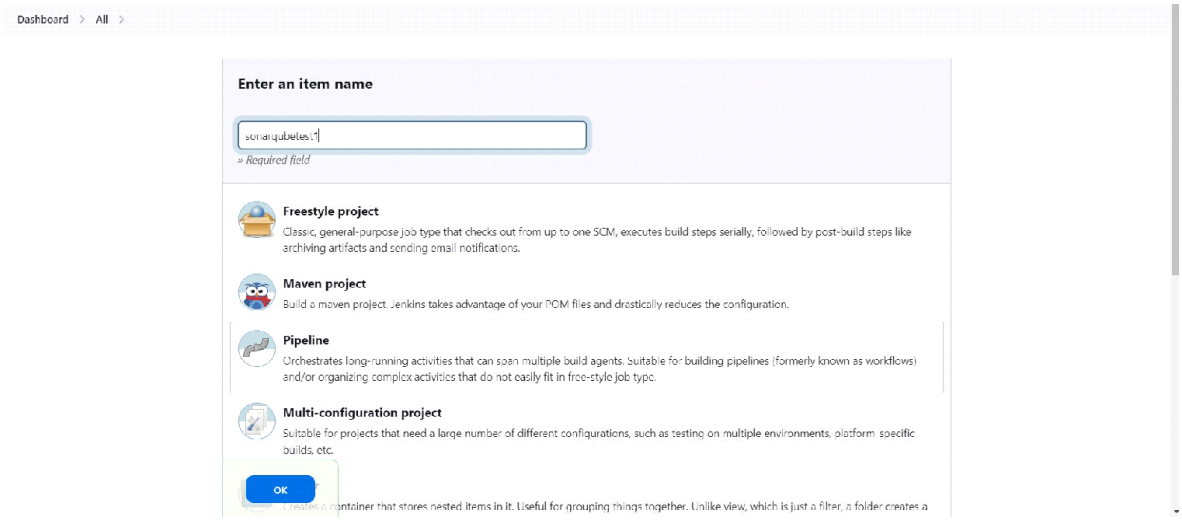
Advance-Devops Experiment no:8

Step 1: Open Windows PowerShell and run the following command – docker run -d --name sonarqube-test1 -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest



```
erShell does not load commands from the current location by default. If you trust this command, instead type: ".\sonar-s
canner.bat". See "get-help about_Command_Precedence" for more details.
PS C:\sonar-scanner\sonar-scanner-6.2.0.4584-windows-x64\bin> .\sonar-scanner.bat
11:02:02.120 INFO  Scanner configuration file: C:\sonar-scanner\sonar-scanner-6.2.0.4584-windows-x64\bin\..\conf\sonar-s
canner.properties
11:02:02.124 INFO  Project root configuration file: NONE
11:02:02.140 INFO  SonarScanner CLI 6.2.0.4584
11:02:02.142 INFO  Java 17.0.12 Eclipse Adoptium (64-bit)
11:02:02.142 INFO  Windows 11 10.0 amd64
11:02:02.160 INFO  User cache: C:\Users\navan\.sonar\cache
11:02:02.644 INFO  JRE provisioning: os[windows], arch[amd64]
11:02:06.241 INFO  EXECUTION FAILURE
11:02:06.243 INFO  Total time: 4.126s
11:02:06.244 ERROR Error during SonarScanner CLI execution
java.lang.IllegalStateException: Error status returned by url [https://api.sonarcloud.io/analysis/jres?os=windows&arch=a
md64]: 401
        at org.sonarsource.scanner.lib.internal.http.ServerConnection.callUrl(ServerConnection.java:182)
        at org.sonarsource.scanner.lib.internal.http.ServerConnection.callApi(ServerConnection.java:145)
        at org.sonarsource.scanner.lib.internal.http.ServerConnection.callRestApi(ServerConnection.java:123)
        at org.sonarsource.scanner.lib.internal.JavaRunnerFactory.getJreMetadata(JavaRunnerFactory.java:159)
        at org.sonarsource.scanner.lib.internal.JavaRunnerFactory.getJreFromServer(JavaRunnerFactory.java:138)
        at org.sonarsource.scanner.lib.internal.JavaRunnerFactory.createRunner(JavaRunnerFactory.java:85)
        at org.sonarsource.scanner.lib.internal.ScannerEngineLauncherFactory.createLauncher(ScannerEngineLauncherFactory
.java:53)
        at org.sonarsource.scanner.lib.ScannerEngineBootstrapper.bootstrap(ScannerEngineBootstrapper.java:118)
        at org.sonarsource.scanner.cli.Main.analyze(Main.java:75)
        at org.sonarsource.scanner.cli.Main.main(Main.java:63)
11:02:06.246 ERROR
11:02:06.246 ERROR Re-run SonarScanner CLI using the -X switch to enable full debug logging.
PS C:\sonar-scanner\sonar-scanner-6.2.0.4584-windows-x64\bin>
```

- Login to SonarQube using username admin and password admin.
- Create a manual project in SonarQube with the name sonarqube-test1

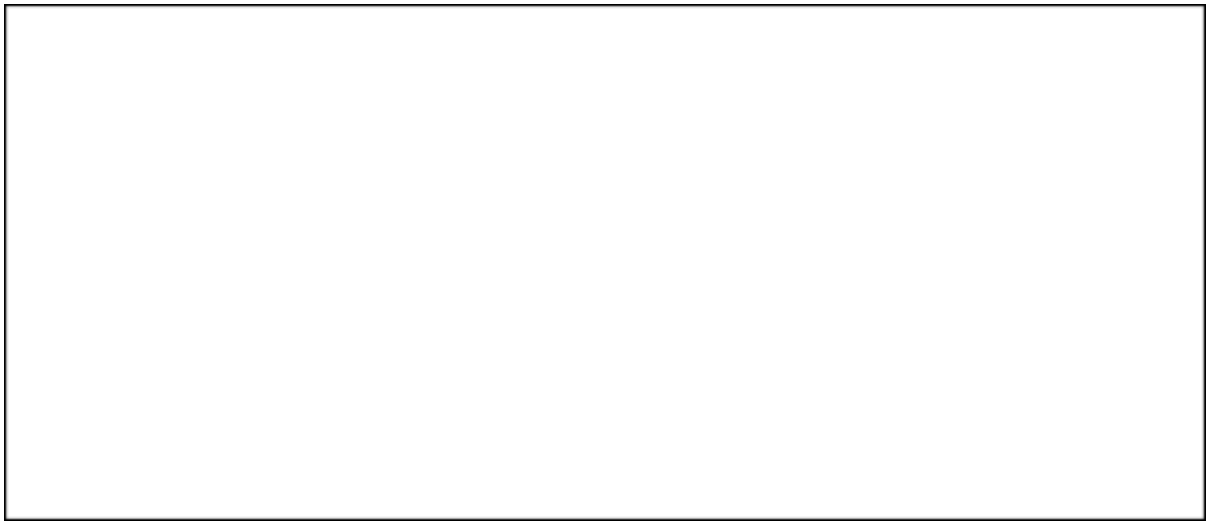


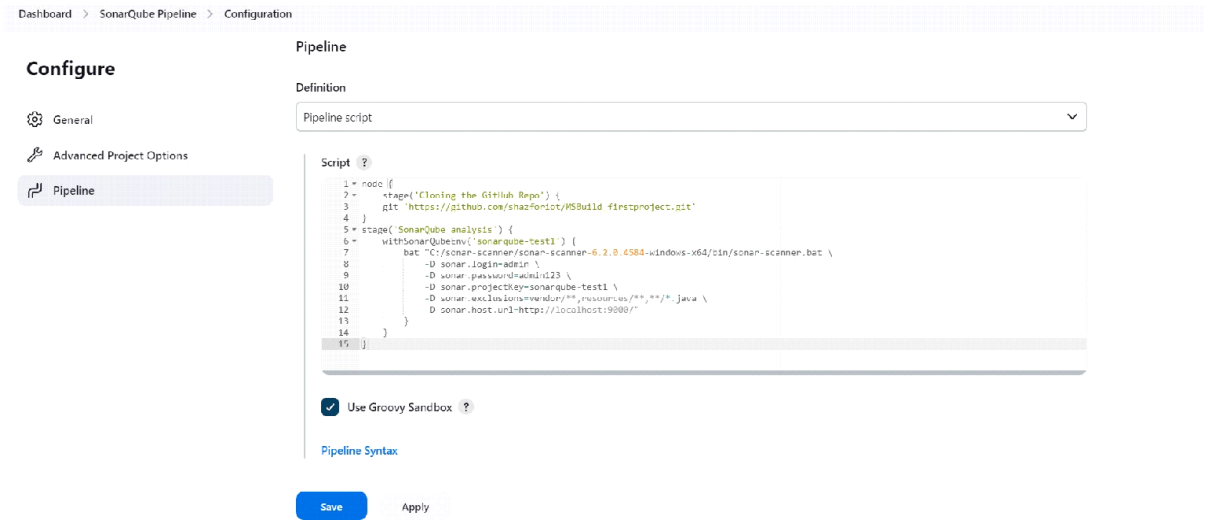
Step2: go to the jenkins and create new item select pipeline:

Under Pipeline Script, enter the following -

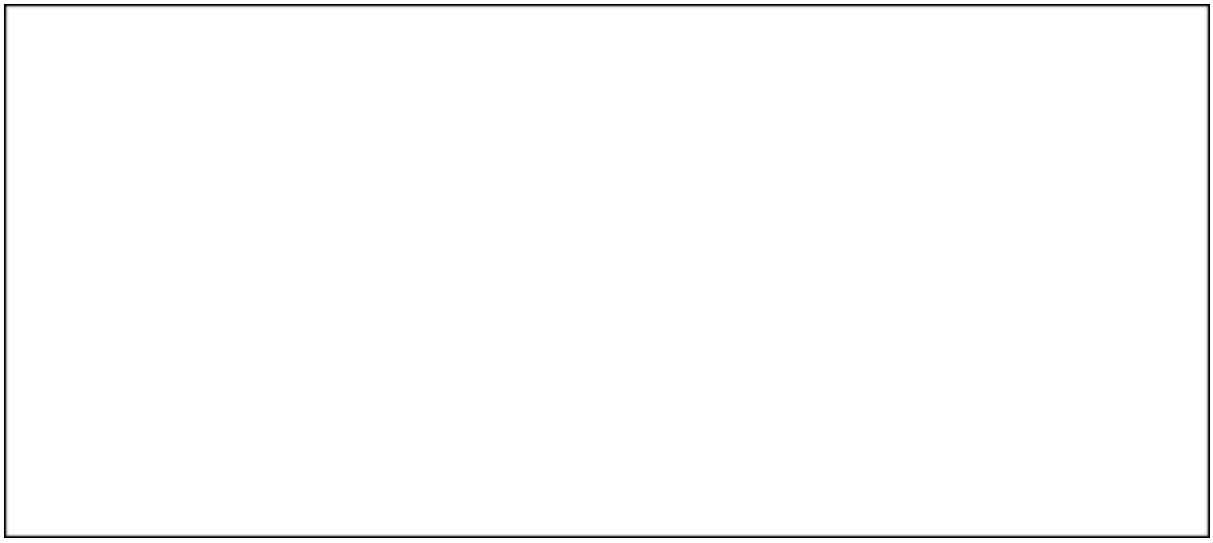
```
node {
        stage('Cloning the GitHub Repo') {
                git 'https://github.com/shazforiot/GOL.git'
        }
stage('SonarQube analysis') {
        withSonarQubeEnv('sonarqube') {
        sh "<PATH_TO_SONARQUBE_FOLDER>//bin//sonar-scanner \
                -D sonar.login=<SonarQube_USERNAME> \
                -D sonar.password=<SonarQube_PASSWORD> \
                -D sonar.projectKey=<Project_KEY> \
                -D sonar.exclusions=vendor/**,resources/**,**/*.java \
                -D sonar.host.url=http://127.0.0.1:9000/"
                }
        }
```

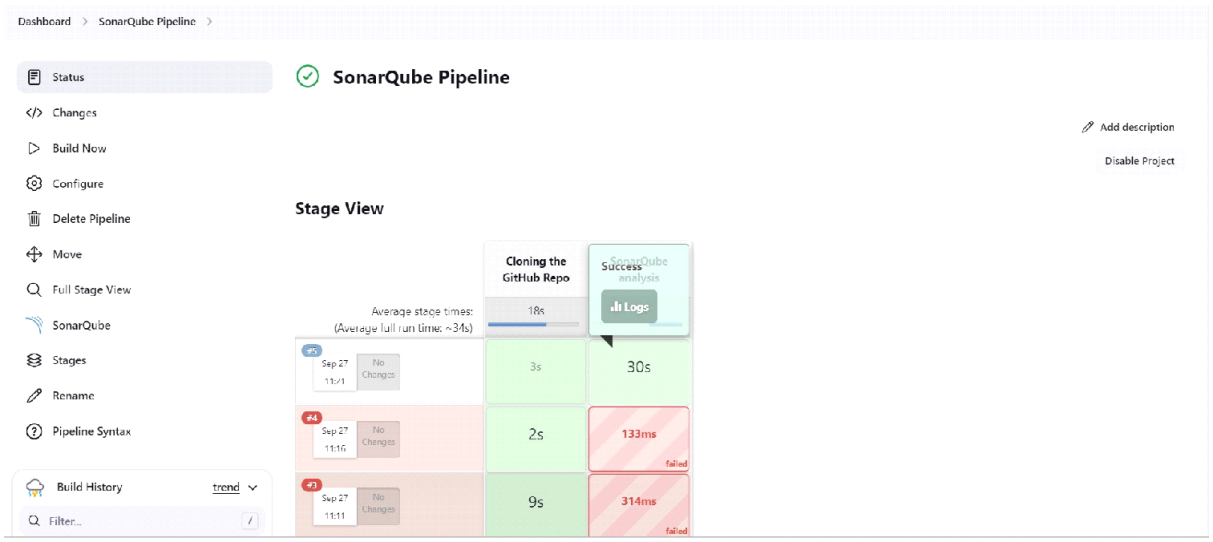
}



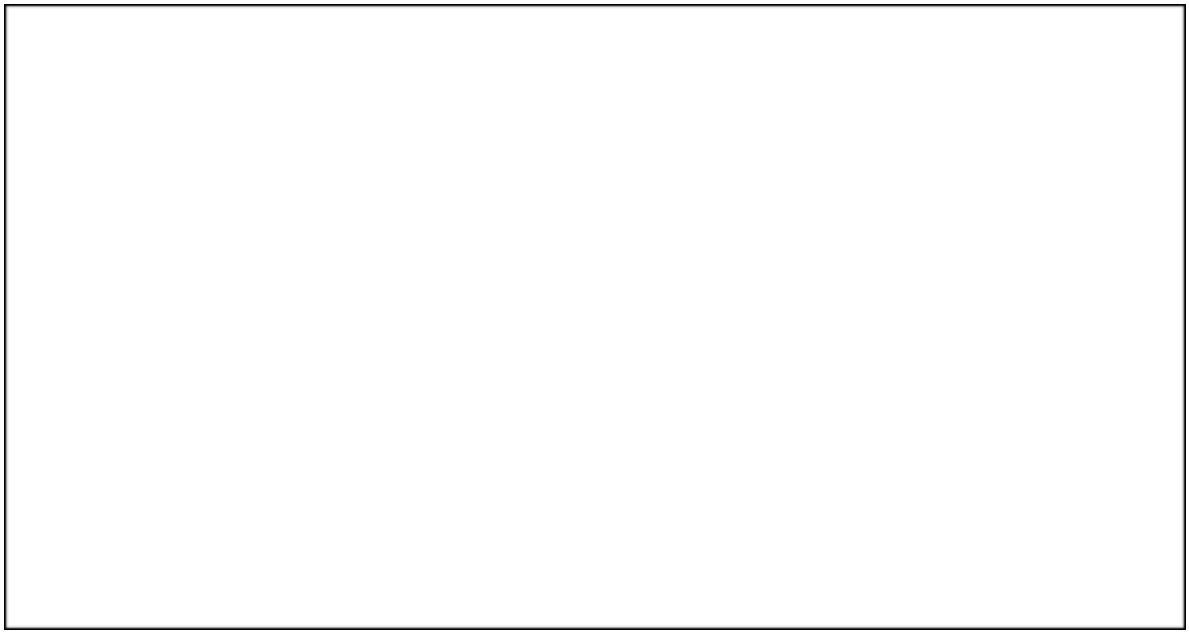


Save the changes and go to build now:



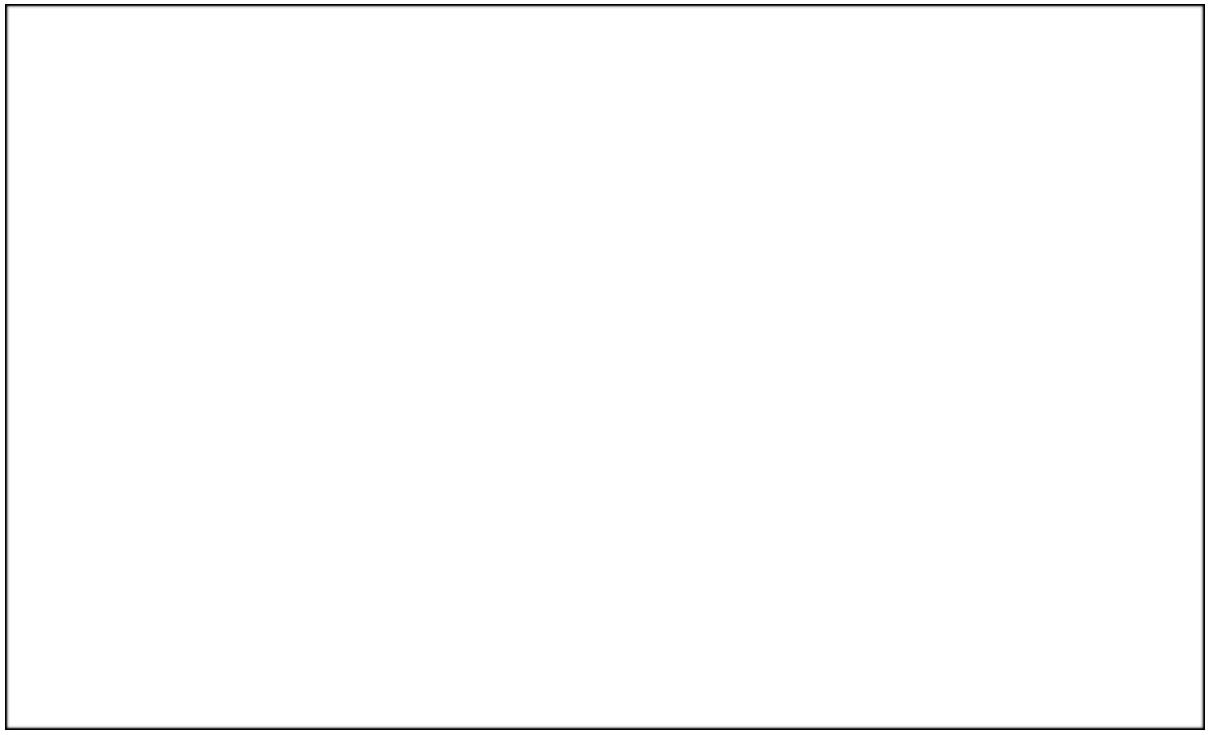


Console output:

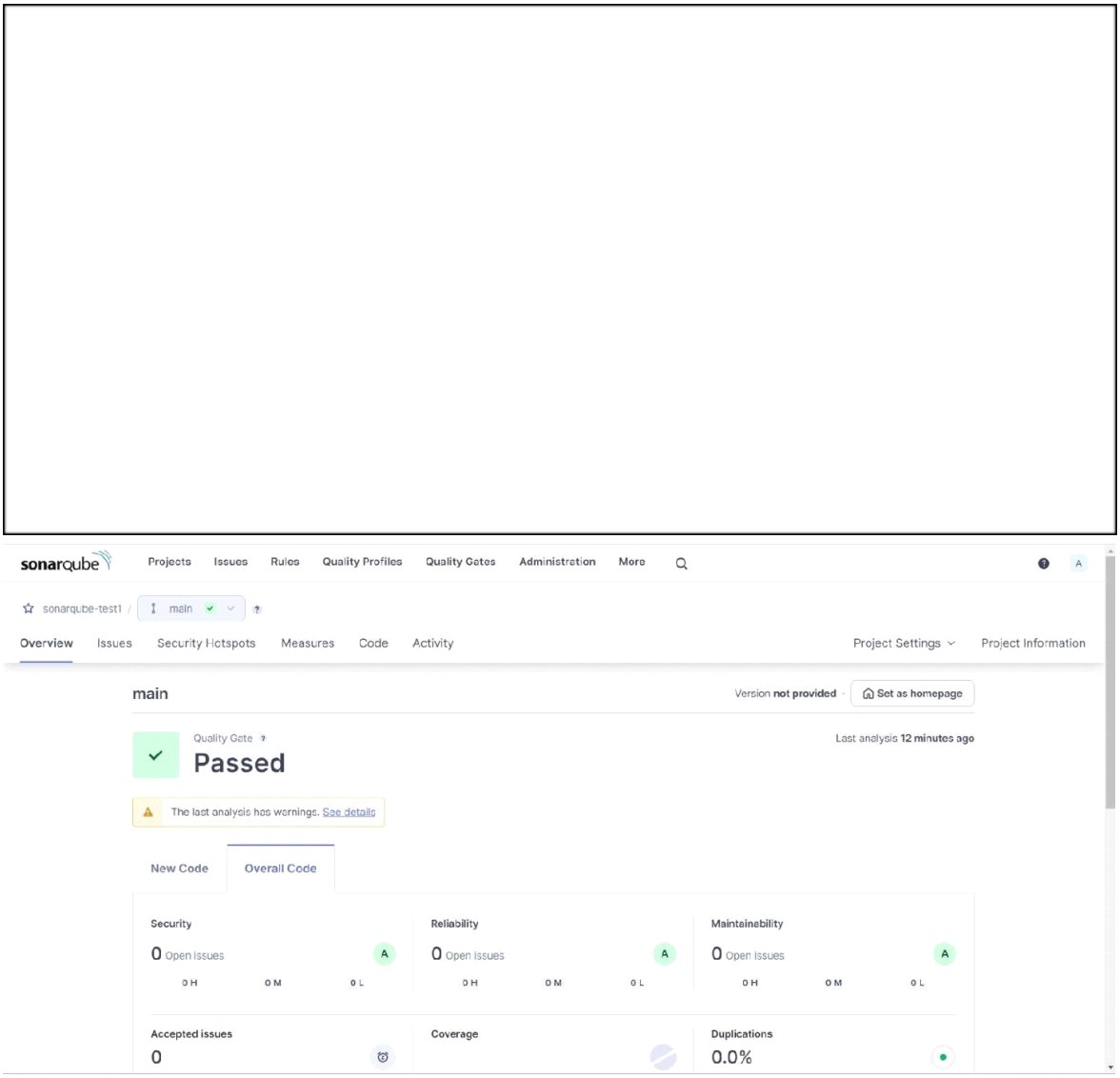


Console Output

```
Started by user jai navani
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in C:\ProgramData\Jenkins\.jenkins\workspace\SonarQube Pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Cloning the GitHub Repo)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
 > git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\SonarQube Pipeline\.git # timeout=10
Fetching changes from the remote Git repository
 > git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstprojcct.git # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject.git
 > git.exe --version # timeout=10
 > git --version # 'git version 2.45.2.windows.1'
 > git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)
 > git.exe config core.sparsecheckout # timeout=10
 > git.exe checkout -f f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
 > git.exe branch -a -v --no-abbrev # timeout=10
 > git.exe branch -D master # timeout=10
```



```
11:22:18.236 INFO  Sensor C# File Caching Sensor [csharp]
11:22:18.237 WARN  Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir'
property.
11:22:18.237 INFO  Sensor C# File Caching Sensor [csharp] (done) | time=1ms
11:22:18.237 INFO  Sensor Zero Coverage Sensor
11:22:18.251 INFO  Sensor Zero Coverage Sensor (done) | time=14ms
11:22:18.256 INFO  SCM Publisher SCM provider for this project is: git
11:22:18.257 INFO  SCM Publisher 4 source files to be analyzed
11:22:18.789 INFO  SCM Publisher 4/4 source files have been analyzed (done) | time=531ms
11:22:18.793 INFO  CPD Executor Calculating CPD for 0 files
11:22:18.795 INFO  CPD Executor CPD calculation finished (done) | time=0ms
11:22:18.810 INFO  SCM revision ID 'f2bc042c04c6e72427c380bcaee6d6fee7b49adf'
11:22:19.074 INFO  Analysis report generated in 134ms, dir size=201.0 kB
11:22:19.137 INFO  Analysis report compressed in 45ms, zip size=22.5 kB
11:22:19.351 INFO  Analysis report uploaded in 212ms
11:22:19.353 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube-test1
11:22:19.354 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
11:22:19.354 INFO  More about the report processing at http://localhost:9000/api/ce/task?id=9/1ae2f2-4e0f-49a/-88c4-ad4a6cceddf8
11:22:19.366 INFO  Analysis total time: 24.819 s
11:22:19.368 INFO  SonarScanner Engine completed successfully
11:22:19.454 INFO  EXECUTION SUCCESS
11:22:19.455 INFO  Total time: 29.696s
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

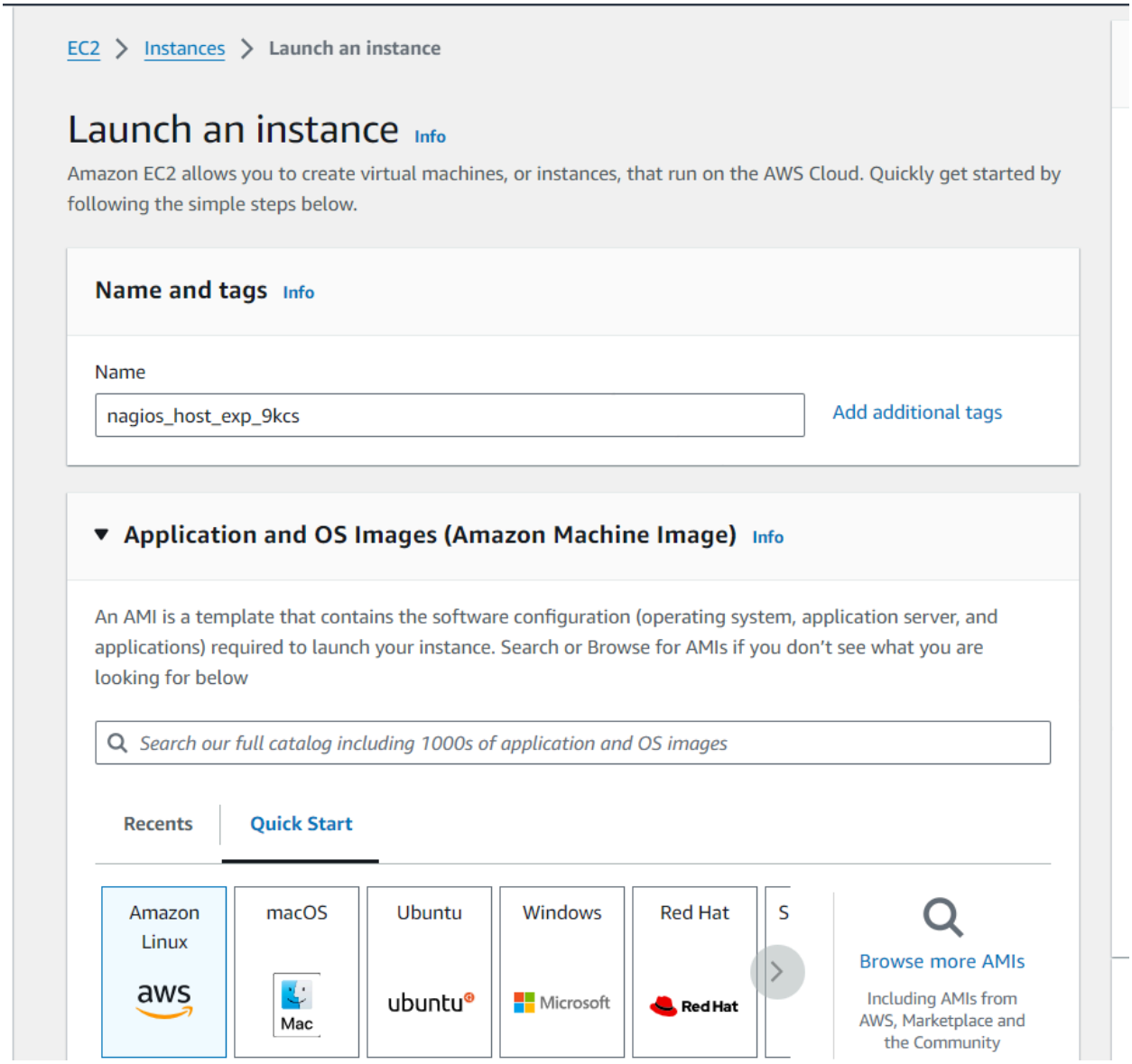

sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
sonarqube-test1
main
Overview
Issues
Security Hotspots
Measures
Code
Activity
Project Settings
Project Information
main
Version not provided
Set as homepage
Quality Gate
Passed
Last analysis 12 minutes ago
The last analysis has warnings. See details
New Code
Overall Code
Security
0 Open issues
A
0 H
0 M
0 L
Reliability
0 Open issues
A
0 H
0 M
0 L
Maintainability
0 Open issues
A
0 H
0 M
0 L
Accepted issues
0
Coverage
Duplications
0.0%

Experiment No 9

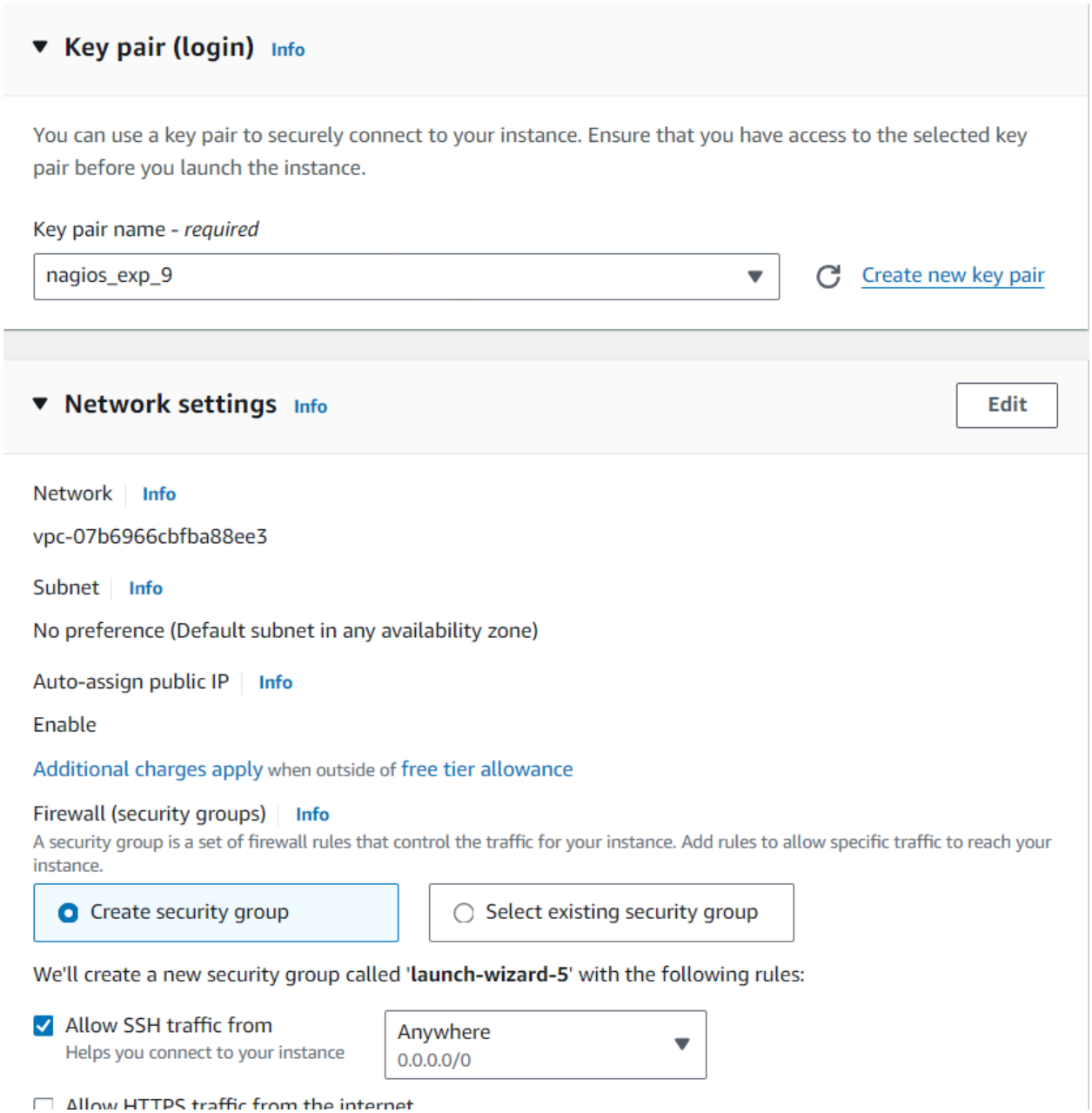
Hitesh Rohra 47 D15A

Screenshots for Experiment 9

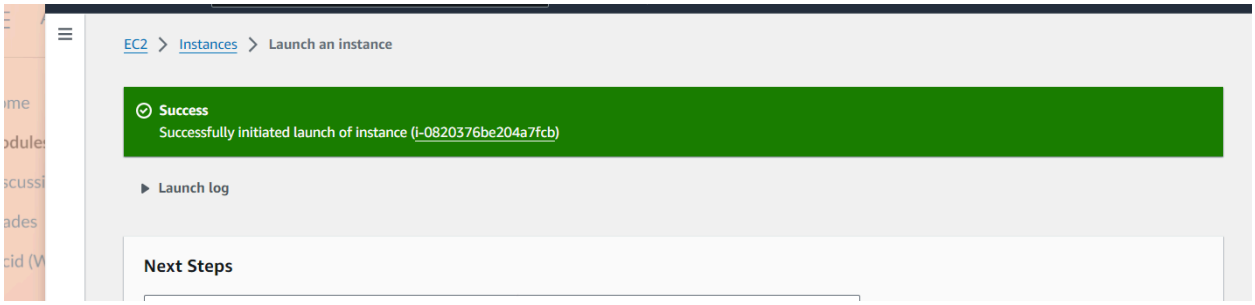
Launch an ec2 instance
Give name use the default OS



Hitesh Rohra 47 D15A
Make a key pair and use it.

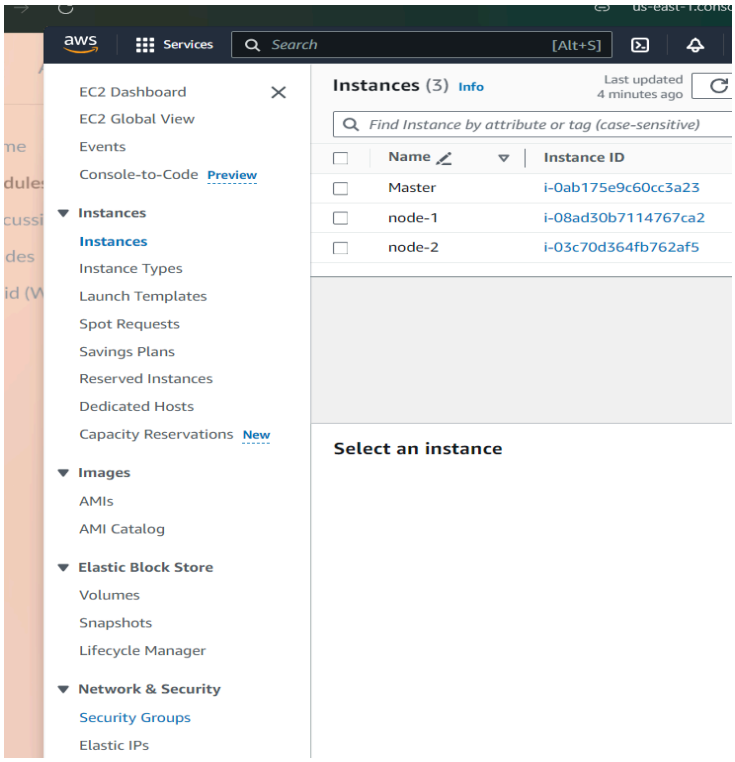


Note the name of the security group that was created for future use:
here it is ‘ launch-wizard-5 ’

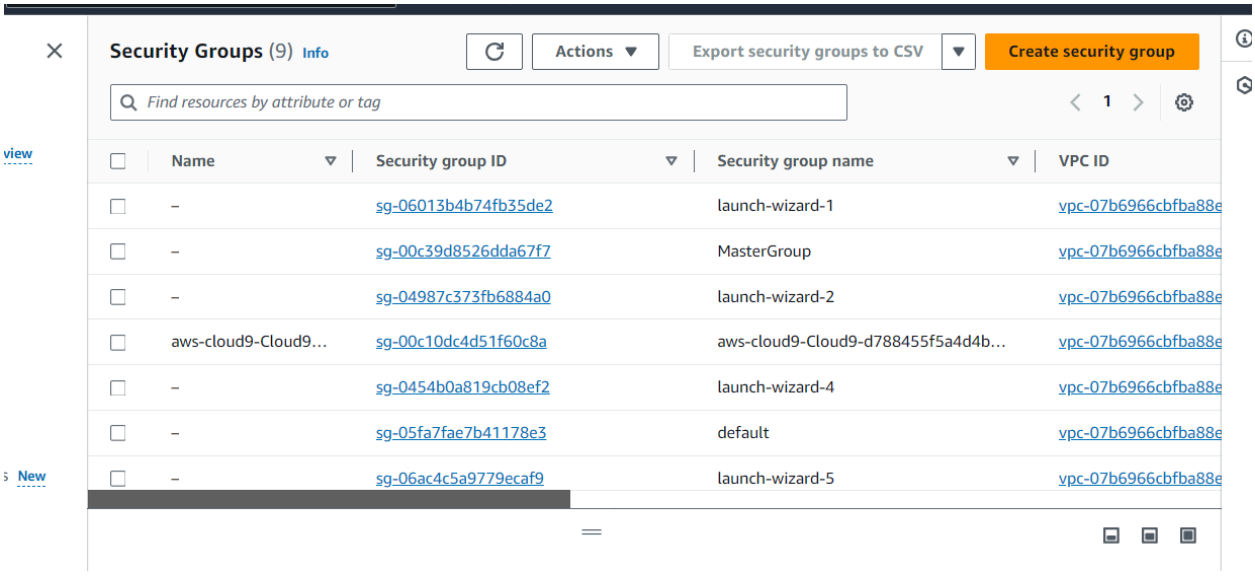


Hitesh Rohra 47 D15A

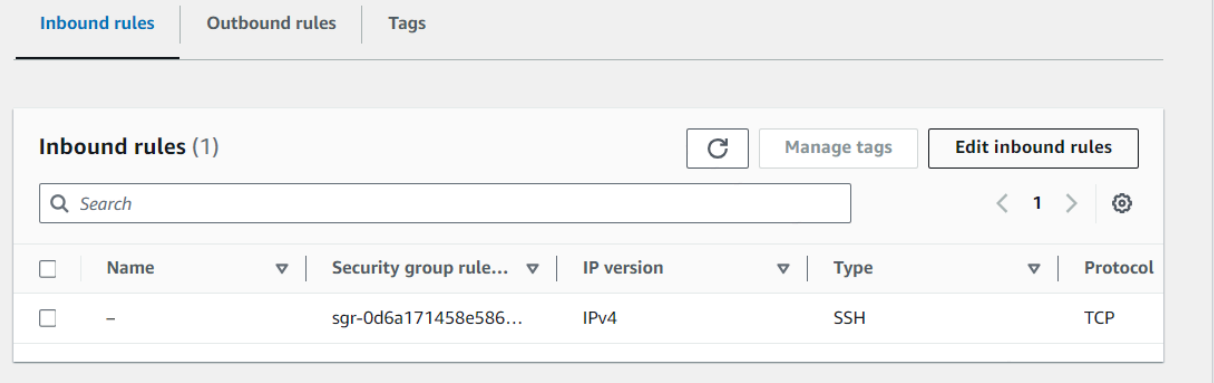
go to security groups:



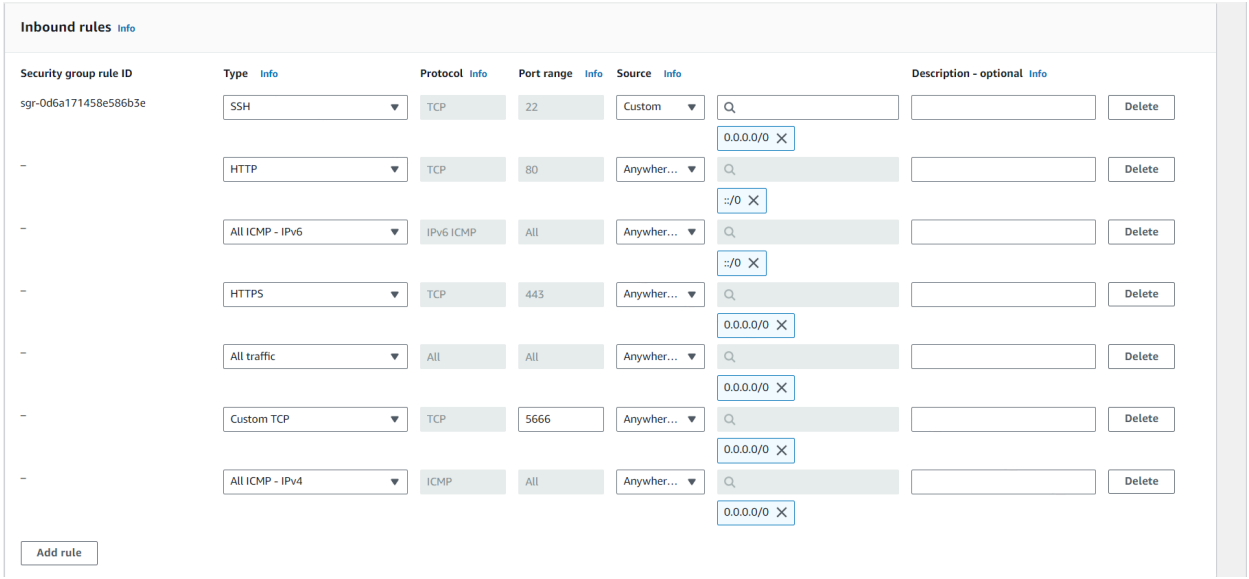
click the security group id which was created while you created the ec2 instance of this experiment.



now click on edit inbound rules

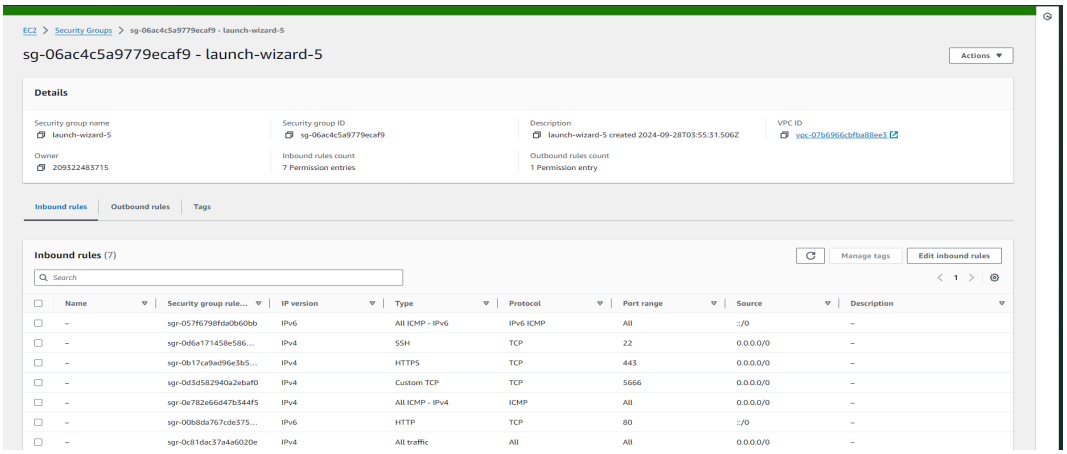


now do the following configurations:
by clicking “add rules”

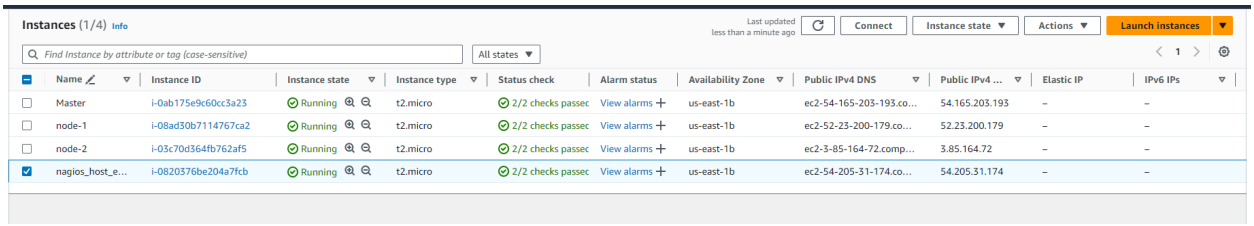


then click on save rules.

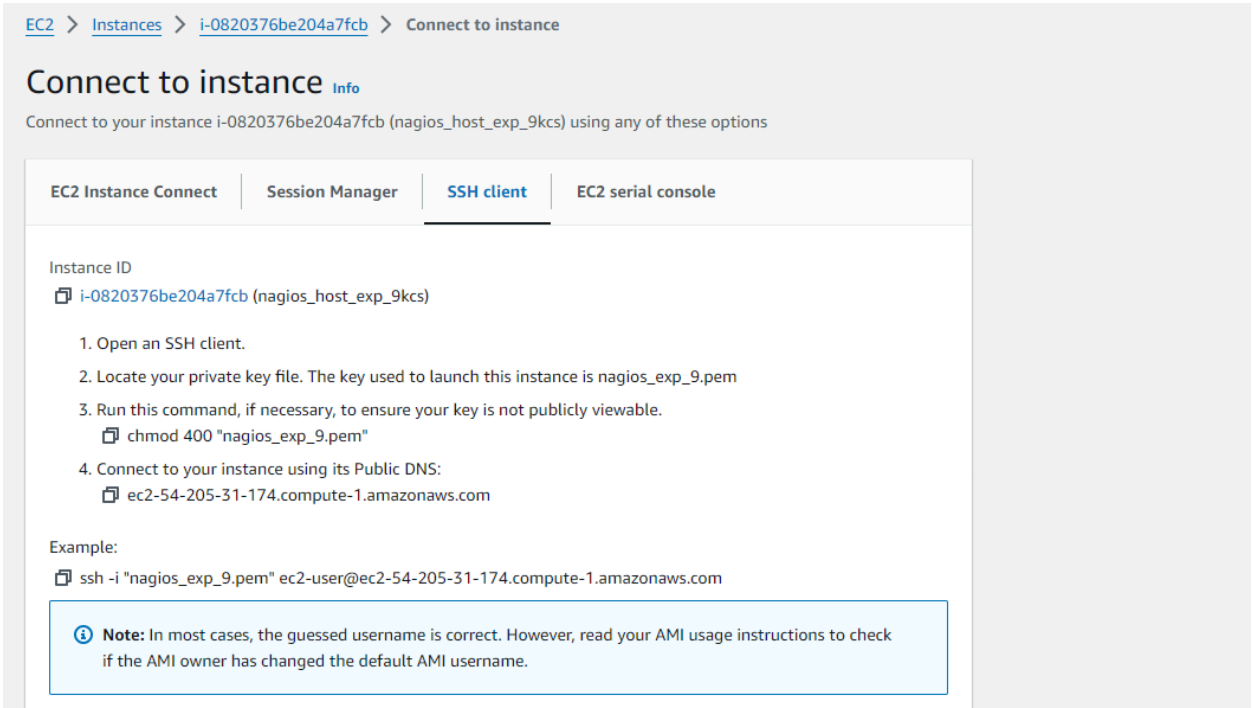
Hitesh Rohra 47 D15A



now navigate to instances, click on the instance which was created earlier and click on connect.



now copy the ssh command and just replace the .pem file with its actual location in your computer.



Hitesh Rohra 47 D15A

paste the command in your terminal and enter after replacing the .pem file with its actual location in your system.

```
C:\Users\Lenovo>ssh -i "C:\Users\Lenovo\Downloads\nagios_exp_9.pem" ec2-user@ec2-54-205-31-174.compute-1.amazonaws.com
The authenticity of host 'ec2-54-205-31-174.compute-1.amazonaws.com (54.205.31.174)' can't be established.
ED25519 key fingerprint is SHA256:+oIS6lcV6qE12x8gFgYVvMsB+yc9vN7UEpF6oBt0jw0.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-205-31-174.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
       #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-80-137 ~]$
```

now paste the following commands in your connected terminal:

sudo yum update

```
       _/m/'
[ec2-user@ip-172-31-80-137 ~]$ sudo yum update
Last metadata expiration check: 2:21:45 ago on Sat Sep 28 03:59:04 2024.
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-80-137 ~]$
```

sudo yum install httpd php

```
[ec2-user@ip-172-31-80-137 ~]$ sudo yum install httpd php
Last metadata expiration check: 2:22:53 ago on Sat Sep 28 03:59:04 2024.
Dependencies resolved.
================================================================================
 Package                 Architecture    Version                 Repository
================================================================================
Installing:
 httpd                   x86_64          2.4.62-1.amzn2023       amazonlinux
 php8.3                  x86_64          8.3.10-1.amzn2023.0.1   amazonlinux
Installing dependencies:
 apr                     x86_64          1.7.2-2.amzn2023.0.2    amazonlinux
 apr-util                x86_64          1.6.3-1.amzn2023.0.1    amazonlinux
 generic-logos-httpd     noarch          18.0.0-12.amzn2023.0.3  amazonlinux
 httpd-core              x86_64          2.4.62-1.amzn2023       amazonlinux
 httpd-filesystem        noarch          2.4.62-1.amzn2023       amazonlinux
```

(type y when prompted)

sudo yum install gcc glibc glibc-common

Hitesh Rohra 47 D15A

```
Dependencies resolved.
================================================================================
 Package                 Architecture       Version                  Repository
================================================================================
Installing:
 gcc                     x86_64             11.4.1-2.amzn2023.0.2    amazonlinux
Installing dependencies:
 annobin-docs            noarch             10.93-1.amzn2023.0.1     amazonlinux
 annobin-plugin-gcc      x86_64             10.93-1.amzn2023.0.1     amazonlinux
 cpp                     x86_64             11.4.1-2.amzn2023.0.2    amazonlinux
 gc                      x86_64             8.0.4-5.amzn2023.0.2     amazonlinux
 glibc-devel             x86_64             2.34-52.amzn2023.0.11    amazonlinux
 glibc-headers-x86       noarch             2.34-52.amzn2023.0.11    amazonlinux
 guile22                 x86_64             2.2.7-2.amzn2023.0.3     amazonlinux
 kernel-headers          x86_64             6.1.109-118.189.amzn2023 amazonlinux
 libmpc                  x86_64             1.2.1-2.amzn2023.0.2     amazonlinux
 libtool-ltdl            x86_64             2.4.7-1.amzn2023.0.3     amazonlinux
 libxcrypt-devel         x86_64             4.4.33-7.amzn2023        amazonlinux
 make                    x86_64             1:4.3-5.amzn2023.0.2     amazonlinux

Transaction Summary
================================================================================
Install  13 Packages

Total download size: 52 M
Installed size: 168 M
Is this ok [y/N]: y
```

sudo yum install gd gd-devel

```
google-noto-sans-vf-fonts-20201206-2.amzn2023.0.2.noarch   graphite2-1.3.14-7.amzn2023.0.2.x86_64
graphite2-devel-1.3.14-7.amzn2023.0.2.x86_64               harfbuzz-7.0.0-2.amzn2023.0.1.x86_64
harfbuzz-devel-7.0.0-2.amzn2023.0.1.x86_64                 harfbuzz-icu-7.0.0-2.amzn2023.0.1.x86_64
jbigkit-libs-2.1-21.amzn2023.0.2.x86_64                    langpacks-core-font-en-3.0-21.amzn2023.0.4.noarch
libICE-1.0.10-6.amzn2023.0.2.x86_64                        libSM-1.2.3-8.amzn2023.0.2.x86_64
libX11-1.7.2-3.amzn2023.0.4.x86_64                         libX11-common-1.7.2-3.amzn2023.0.4.noarch
libX11-devel-1.7.2-3.amzn2023.0.4.x86_64                   libX11-xcb-1.7.2-3.amzn2023.0.4.x86_64
libXau-1.0.9-6.amzn2023.0.2.x86_64                         libXau-devel-1.0.9-6.amzn2023.0.2.x86_64
libXext-1.3.4-6.amzn2023.0.2.x86_64                        libXpm-3.5.15-2.amzn2023.0.3.x86_64
libXpm-devel-3.5.15-2.amzn2023.0.3.x86_64                  libXrender-0.9.10-14.amzn2023.0.2.x86_64
libXt-1.2.0-4.amzn2023.0.2.x86_64                          libblkid-devel-2.37.4-1.amzn2023.0.4.x86_64
libffi-devel-3.4.4-1.amzn2023.0.1.x86_64                   libicu-67.1-7.amzn2023.0.3.x86_64
libicu-devel-67.1-7.amzn2023.0.3.x86_64                    libjpeg-turbo-2.1.4-2.amzn2023.0.5.x86_64
libjpeg-turbo-devel-2.1.4-2.amzn2023.0.5.x86_64            libmount-devel-2.37.4-1.amzn2023.0.4.x86_64
libpng-2:1.6.37-10.amzn2023.0.6.x86_64                     libpng-devel-2:1.6.37-10.amzn2023.0.6.x86_64
libselinux-devel-3.4-5.amzn2023.0.2.x86_64                 libsepol-devel-3.4-3.amzn2023.0.3.x86_64
libtiff-4.4.0-4.amzn2023.0.18.x86_64                       libtiff-devel-4.4.0-4.amzn2023.0.18.x86_64
libwebp-1.2.4-1.amzn2023.0.6.x86_64                        libwebp-devel-1.2.4-1.amzn2023.0.6.x86_64
libxcb-1.13.1-7.amzn2023.0.2.x86_64                        libxcb-devel-1.13.1-7.amzn2023.0.2.x86_64
libxml2-devel-2.10.4-1.amzn2023.0.6.x86_64                 pcre2-devel-10.40-1.amzn2023.0.3.x86_64
pcre2-utf16-10.40-1.amzn2023.0.3.x86_64                    pcre2-utf32-10.40-1.amzn2023.0.3.x86_64
pixman-0.40.0-3.amzn2023.0.3.x86_64                        sysprof-capture-devel-3.40.1-2.amzn2023.0.2.x86_64
xml-common-0.6.3-56.amzn2023.0.2.noarch                    xorg-x11-proto-devel-2021.4-1.amzn2023.0.2.noarch
xz-devel-5.2.5-9.amzn2023.0.2.x86_64                       zlib-devel-1.2.11-33.amzn2023.0.5.x86_64

omplete!
ec2-user@ip-172-31-80-137 ~]$
```

sudo adduser -m nagios
sudo passwd nagios

```
Complete!
[ec2-user@ip-172-31-80-137 ~]$ sudo adduser -m nagios
sudo passwd nagios
Changing password for user nagios.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-80-137 ~]$
```

(add a password here)
sudo groupadd nagcmd

```
[ec2-user@ip-172-31-80-137 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-80-137 ~]$
```

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

```
[ec2-user@ip-172-31-80-137 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-80-137 ~]$ sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache
[ec2-user@ip-172-31-80-137 ~]$
```

mkdir ~/downloads
cd ~/downloads

```
[ec2-user@ip-172-31-80-137 ~]$ mkdir ~/downloads
cd ~/downloads
[ec2-user@ip-172-31-80-137 downloads]$
```

wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz

```
cd ~/downloads
[ec2-user@ip-172-31-80-137 downloads]$ wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
--2024-09-28 06:27:51--  https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2600:3c00::f03c:92ff:fef7:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: ‘nagios-4.5.5.tar.gz’

nagios-4.5.5.tar.gz          100%[=============================================>]   1.97M  5.30MB/s    in 0.4s

2024-09-28 06:27:52 (5.30 MB/s) - ‘nagios-4.5.5.tar.gz’ saved [2065473/2065473]

[ec2-user@ip-172-31-80-137 downloads]$
```

Hitesh Rohra 47 D15A

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

```
[ec2-user@ip-172-31-80-137 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
--2024-09-28 06:28:14--  https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2753049 (2.6M) [application/x-gzip]
Saving to: 'nagios-plugins-2.4.11.tar.gz'

nagios-plugins-2.4.11.tar.gz  100%[=================================================>]   2.62M  5.90MB/s    in 0.4s

2024-09-28 06:28:15 (5.90 MB/s) - 'nagios-plugins-2.4.11.tar.gz' saved [2753049/2753049]

[ec2-user@ip-172-31-80-137 downloads]$
```

tar zxvf nagios-4.5.5.tar.gz

```
[ec2-user@ip-172-31-80-137 downloads]$ tar zxvf nagios-4.5.5.tar.gz
nagios-4.5.5/
nagios-4.5.5/.github/
nagios-4.5.5/.github/workflows/
nagios-4.5.5/.github/workflows/test.yml
nagios-4.5.5/.gitignore
nagios-4.5.5/CONTRIBUTING.md
nagios-4.5.5/Changelog
nagios-4.5.5/INSTALLING
nagios-4.5.5/LEGAL
nagios-4.5.5/LICENSE
nagios-4.5.5/Makefile.in
```

```
nagiosexp9
[ec2-user@ip-172-31-80-137 downloads]$ ls
nagios-4.5.5  nagios-4.5.5.tar.gz  nagios-plugins-2.4.11.tar.gz
[ec2-user@ip-172-31-80-137 downloads]$
```

cd nagios-4.5.5

```
[ec2-user@ip-172-31-80-137 downloads]$  cd nagios-4.5.5
```

sudo yum install openssl-devel

Hitesh Rohra 47 D15A

```
[ec2-user@ip-172-31-80-137 nagios-4.5.5]$ sudo yum install openssl-devel
Last metadata expiration check: 2:31:25 ago on Sat Sep 28 03:59:04 2024.
Dependencies resolved.
================================================================================
 Package                 Architecture        Version                 Repository                 Size
================================================================================
Installing:
 openssl-devel           x86_64              1:3.0.8-1.amzn2023.0.14 amazonlinux                3.0 M

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

Total download size: 3.0 M
Installed size: 4.7 M
Is this ok [y/N]: y
```

```
Total                                                           18 MB/s | 3.0 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                              1/1
  Installing       : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64                 1/1
  Running scriptlet: openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64                 1/1
  Verifying        : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64                 1/1

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

Complete!
[ec2-user@ip-172-31-80-137 nagios-4.5.5]$
```

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

Hitesh Rohra 47 D15A

```
nagiosexp9

[ec2-user@ip-172-31-80-137 nagios-4.5.5]$ ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether the compiler supports GNU C... yes
checking whether gcc accepts -g... yes
checking for gcc option to enable C11 features... none needed
checking whether make sets $(MAKE)... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking for sys/wait.h that is POSIX.1 compatible... yes
checking for stdio.h... yes
checking for stdlib.h... yes
checking for string.h... yes
checking for inttypes.h... yes
checking for stdint.h... yes
```

make all

```
[ec2-user@ip-172-31-80-137 nagios-4.5.5]$ make all
cd ./base && make
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o nagios.o ./nagi
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o broker.o broker
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o nebmods.o nebmo
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o ../common/share
./common/shared.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o query-handler.o
y-handler.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o workers.o worke
In function 'get_wproc_list':
```

```
      - 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.
```

sudo make install

Hitesh Rohra 47 D15A

sudo make install-init

sudo make install-config

sudo make install-commandmode

```
[ec2-user@ip-172-31-80-137 nagios-4.5.5]$ 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-4.5.5/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-4.5.5/base'
cd ./cgi && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
make install-basic
make[2]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/sbin
for file in *.cgi; do \
```

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

```
nagiosexp9
  GNU nano 5.8                          /usr/local/nagios/etc/objects/contacts.cfg
##############################################################################
# CONTACTS.CFG - SAMPLE CONTACT/CONTACTGROUP DEFINITIONS
#
#
# NOTES: This config file provides you with some example contact and contact
#        group definitions that you can reference in host and service
#        definitions.
#
#        You don't need to keep these definitions in a separate file from your
#        other object definitions.  This has been done just to make things
#        easier to understand.
#
##############################################################################



##############################################################################
#
# CONTACTS
#
##############################################################################

# Just one contact defined by default - the Nagios admin (that's you)
                                        [ Read 51 lines ]
^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo
```

navigate down to email: and change it to your email address.

```
# This contact definition inherits a lot of default values from the
# 'generic-contact' template which is defined elsewhere.

define contact {

    contact_name            nagiosadmin             ; Short name of user
    use                     generic-contact         ; Inherit default values from generic-contact template (defined abo
    alias                   Nagios Admin            ; Full name of user
    email                   nagios@localhost ; <<***** CHANGE THIS TO YOUR EMAIL ADDRESS ******
}



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

```
# Just one contact defined by default - the Nagios admin (that's you)
# This contact definition inherits a lot of default values from the
# 'generic-contact' template which is defined elsewhere.

define contact {

    contact_name            nagiosadmin             ; Short name of user
    use                     generic-contact         ; Inherit default values from generic-contact template (define
    alias                   Nagios Admin            ; Full name of user
    email                   2022.shubham.jha@ves.ac.in ; <<***** CHANGE THIS TO YOUR EMAIL ADDRESS ******
}



###############################################################################
#
# CONTACT GROUPS

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set M
```

press Ctrl+O and then enter.
then press Ctrl +X

```
chmod g+s /usr/local/nagios/var/rw

*** External command directory configured ***

[ec2-user@ip-172-31-80-137 nagios-4.5.5]$ sudo nano /usr/local/nagios/etc/objects/contacts.cfg
[ec2-user@ip-172-31-80-137 nagios-4.5.5]$
```

sudo make install-webconf

```
[ec2-user@ip-172-31-80-137 nagios-4.5.5]$ 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-172-31-80-137 nagios-4.5.5]$
```

Adding password for nagios admin

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

```
[ec2-user@ip-172-31-80-137 nagios-4.5.5]$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
[ec2-user@ip-172-31-80-137 nagios-4.5.5]$
```

sudo service httpd restart

```
[ec2-user@ip-172-31-80-137 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-80-137 nagios-4.5.5]$
```

cd ~/downloads
tar zxvf nagios-plugins-2.4.11.tar.gz

```
[ec2-user@ip-172-31-80-137 downloads]$ cd ~/downloads
tar zxvf nagios-plugins-2.4.11.tar.gz
nagios-plugins-2.4.11/
nagios-plugins-2.4.11/build-aux/
nagios-plugins-2.4.11/build-aux/compile
nagios-plugins-2.4.11/build-aux/config.guess
nagios-plugins-2.4.11/build-aux/config.rpath
nagios-plugins-2.4.11/build-aux/config.sub
nagios-plugins-2.4.11/build-aux/install-sh
nagios-plugins-2.4.11/build-aux/ltmain.sh
nagios-plugins-2.4.11/build-aux/missing
nagios-plugins-2.4.11/build-aux/mkinstalldirs
nagios-plugins-2.4.11/build-aux/depcomp
nagios-plugins-2.4.11/build-aux/snippet/
```

cd nagios-plugins-2.4.11
./configure --with-nagios-user=nagios --with-nagios-group=nagios

```
nagiosexp9
[ec2-user@ip-172-31-80-137 downloads]$ cd nagios-plugins-2.4.11
./configure --with-nagios-user=nagios --with-nagios-group=nagios
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether to enable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether gcc understands -c and -o together... yes
checking whether make supports the include directive... yes (GNU style)
```

make
sudo make install

```
ake[1]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/plugins-root'
aking install in po
ake[1]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/po'
usr/bin/mkdir -p /usr/local/nagios/share
nstalling fr.gmo as /usr/local/nagios/share/locale/fr/LC_MESSAGES/nagios-plugins.mo
nstalling de.gmo as /usr/local/nagios/share/locale/de/LC_MESSAGES/nagios-plugins.mo
f test "nagios-plugins" = "gettext-tools"; then \
 /usr/bin/mkdir -p /usr/local/nagios/share/gettext/po; \
 for file in Makefile.in.in remove-potcdate.sin    Makevars.template; do \
   /usr/bin/install -c -o nagios -g nagios -m 644 ./$file \
                   /usr/local/nagios/share/gettext/po/$file; \
 done; \
 for file in Makevars; do \
   rm -f /usr/local/nagios/share/gettext/po/$file; \
 done; \
lse \
 : ; \
i
ake[1]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/po'
ake[1]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
ake[2]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
ake[2]: Nothing to be done for 'install-exec-am'.
ake[2]: Nothing to be done for 'install-data-am'.
ake[2]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
ake[1]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
```

sudo chkconfig --add nagios
sudo chkconfig nagios on

```
[ec2-user@ip-172-31-80-137 nagios-plugins-2.4.11]$ sudo chkconfig --add nagios
sudo chkconfig nagios on
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.
[ec2-user@ip-172-31-80-137 nagios-plugins-2.4.11]$
```

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

```
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.
[ec2-user@ip-172-31-80-137 nagios-plugins-2.4.11]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.5.5
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2024-09-17
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.
```

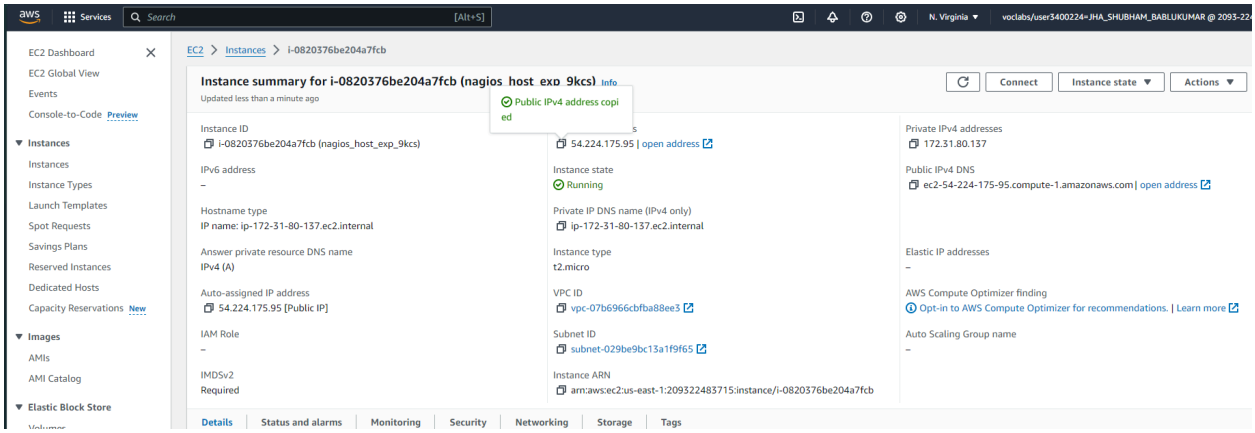
Hitesh Rohra 47 D15A

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

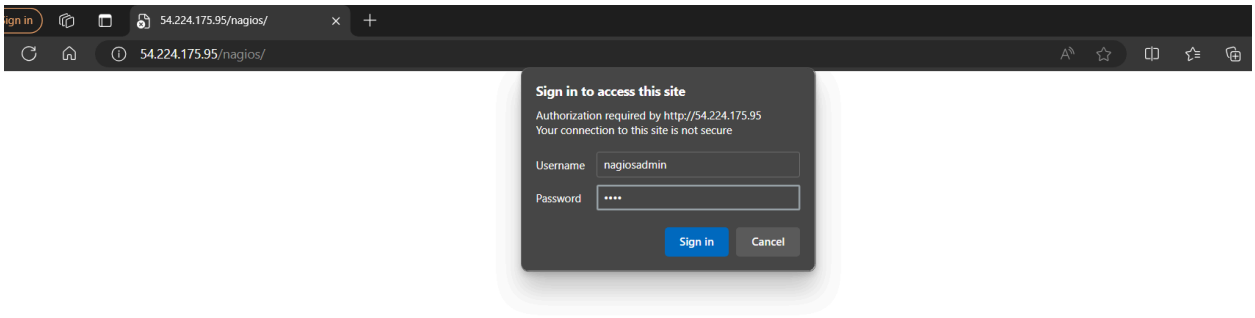
```
Things look okay - No serious problems were detected during the pre-flight check
[ec2-user@ip-172-31-80-137 nagios-plugins-2.4.11]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-80-137 nagios-plugins-2.4.11]$
```

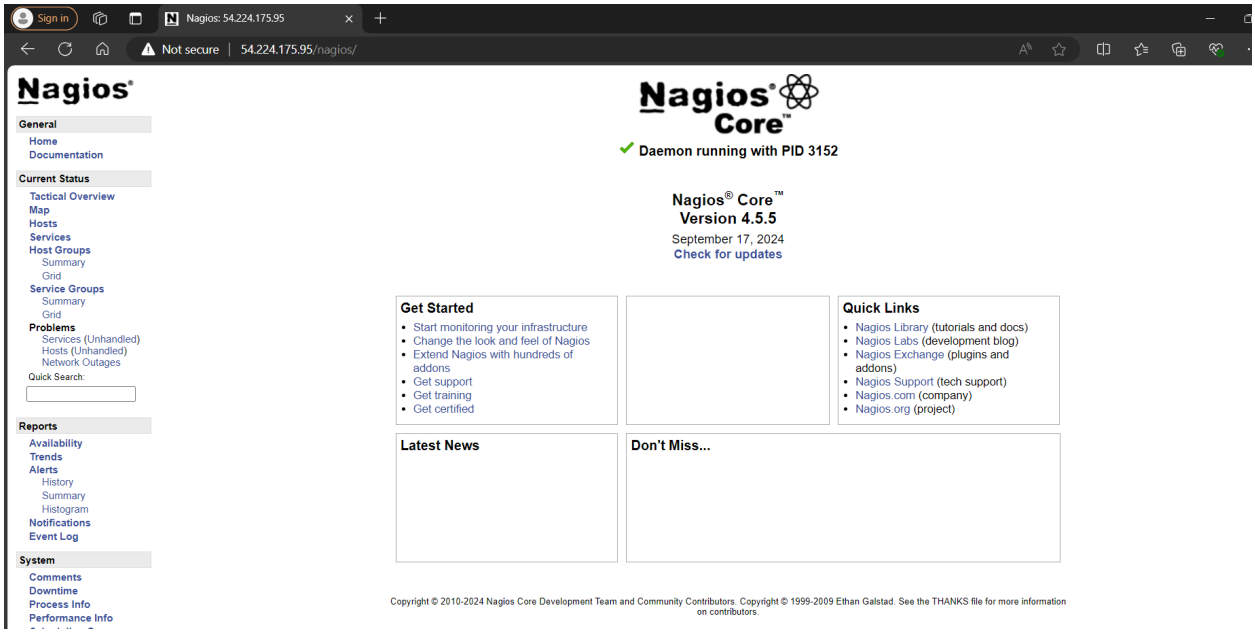
sudo systemctl status nagios

```
[ec2-user@ip-172-31-80-137 nagios-plugins-2.4.11]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Sat 2024-09-28 07:40:16 UTC; 35s ago
       Docs: https://www.nagios.org/documentation
    Process: 71009 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 71010 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 71011 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 5.6M
        CPU: 82ms
     CGroup: /system.slice/nagios.service
             ├─71011 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─71012 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─71013 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─71014 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─71015 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─71016 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 28 07:40:16 ip-172-31-80-137.ec2.internal nagios[71011]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully initial
Sep 28 07:40:16 ip-172-31-80-137.ec2.internal nagios[71011]: qh: core query handler registered
Sep 28 07:40:16 ip-172-31-80-137.ec2.internal nagios[71011]: qh: echo service query handler registered
Sep 28 07:40:16 ip-172-31-80-137.ec2.internal nagios[71011]: qh: help for the query handler registered
Sep 28 07:40:16 ip-172-31-80-137.ec2.internal nagios[71011]: wproc: Successfully registered manager as @wproc with query handler
Sep 28 07:40:16 ip-172-31-80-137.ec2.internal nagios[71011]: wproc: Registry request: name=Core Worker 71015;pid=71015
Sep 28 07:40:16 ip-172-31-80-137.ec2.internal nagios[71011]: wproc: Registry request: name=Core Worker 71014;pid=71014
Sep 28 07:40:16 ip-172-31-80-137.ec2.internal nagios[71011]: wproc: Registry request: name=Core Worker 71013;pid=71013
Sep 28 07:40:16 ip-172-31-80-137.ec2.internal nagios[71011]: wproc: Registry request: name=Core Worker 71012;pid=71012
Sep 28 07:40:17 ip-172-31-80-137.ec2.internal nagios[71011]: Successfully launched command file worker with pid 71016
lines 1-28/28 (END)
```



Hitesh Rohra 47 D15A





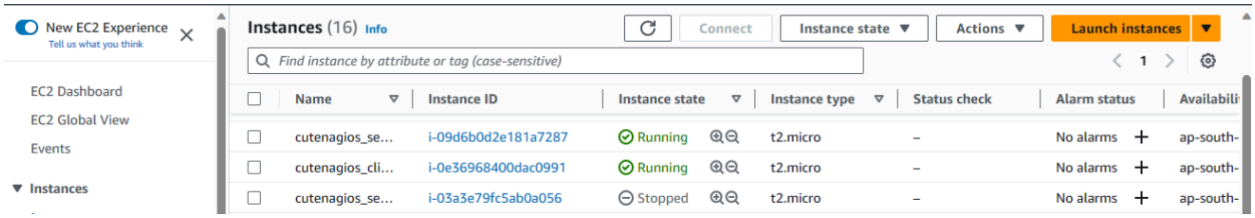
Experiment No 10

Hitesh Rohra 47 D15A

Screenshots for Experiment 10

```
● nagios.service - Nagios Core 4.4.14
     Loaded: loaded (/lib/systemd/system/nagios.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2023-09-30 08:54:01 UTC; 20s ago
       Docs: https://www.nagios.org/documentation
    Process: 55285 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 55286 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 55287 (nagios)
      Tasks: 6 (limit: 1141)
     Memory: 5.3M
        CPU: 252ms
     CGroup: /system.slice/nagios.service
             ├─55287 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─55288 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─55289 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─55290 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─55291 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─55292 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 30 08:54:01 ip-172-31-44-151 nagios[55287]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully initialized
lines 1-19
```

Monitoring a Linux machine, create an Ubuntu 20.04 server EC2 Instance in AWS.



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

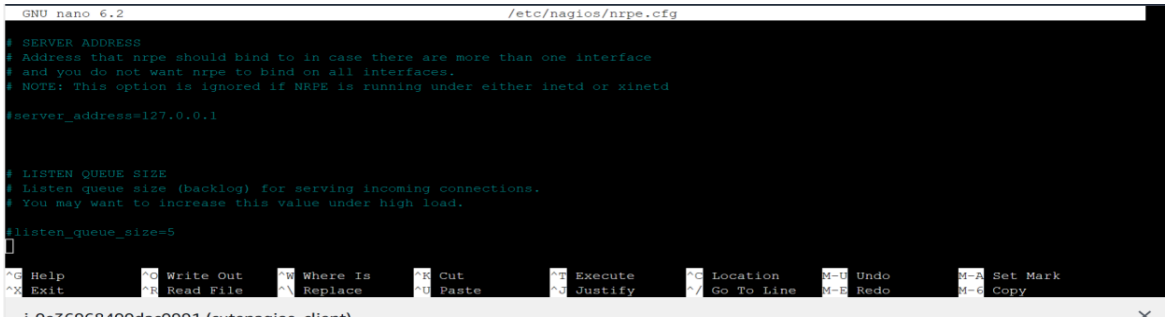
```
*** System restart required ***
Last login: Sat Sep 30 08:31:30 2023 from 13.233.177.3
ubuntu@ip-172-31-44-151:~$ sudo apt install gcc -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
gcc is already the newest version (4:11.2.0-1ubuntu1).
gcc set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
```

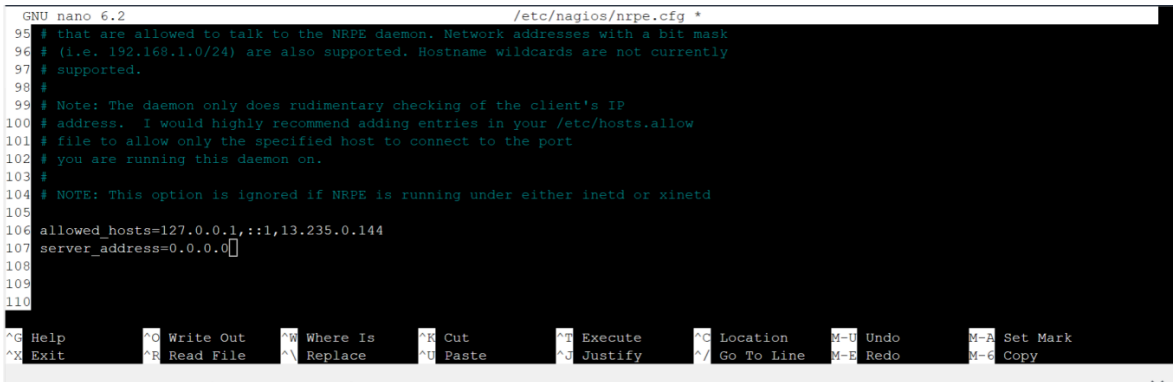
```
root@ip-172-31-44-151:/home/ubuntu# sudo apt install nagios-nrpe-server nagios-plugins
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'monitoring-plugins' instead of 'nagios-plugins'
monitoring-plugins is already the newest version (2.3.1-1ubuntu4).
nagios-nrpe-server is already the newest version (4.0.3-1ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
```

```
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Fetched 229 kB in 1s (290 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
2 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ip-172-31-44-151:/home/ubuntu# sudo apt install gcc -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
gcc is already the newest version (4:11.2.0-1ubuntu1).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
root@ip-172-31-44-151:/home/ubuntu# 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'
monitoring-plugins is already the newest version (2.3.1-1ubuntu4).
nagios-nrpe-server is already the newest version (4.0.3-1ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
```

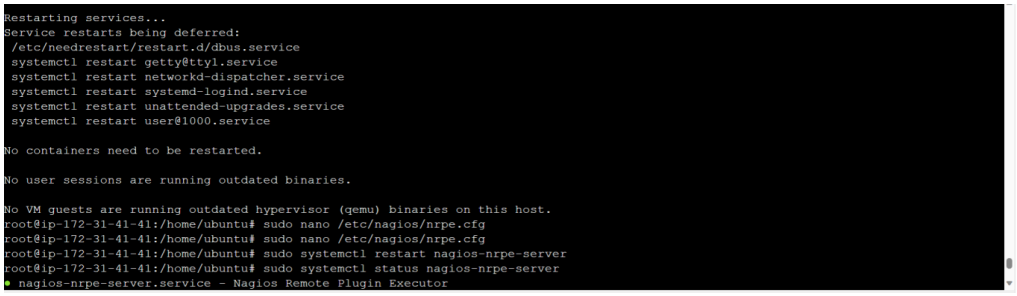
Hitesh Rohra 47 D15A

sudo nano /etc/nagios/nrpe.cfg





sudo systemctl restart nagios-nrpe-server



Hitesh Rohra 47 D15A

```
root@ip-172-31-41-41:/home/ubuntu# sudo systemctl status nagios-nrpe-server
● nagios-nrpe-server.service - Nagios Remote Plugin Executor
     Loaded: loaded (/lib/systemd/system/nagios-nrpe-server.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2023-09-30 09:27:17 UTC; 6s ago
       Docs: http://www.nagios.org/documentation
   Main PID: 7349 (nrpe)
      Tasks: 1 (limit: 1141)
     Memory: 1.5M
        CPU: 9ms
     CGroup: /system.slice/nagios-nrpe-server.service
             └─7349 /usr/sbin/nrpe -c /etc/nagios/nrpe.cfg -f

Sep 30 09:27:17 ip-172-31-41-41 systemd[1]: nagios-nrpe-server.service: Deactivated successfully.
Sep 30 09:27:17 ip-172-31-41-41 systemd[1]: Stopped Nagios Remote Plugin Executor.
Sep 30 09:27:17 ip-172-31-41-41 systemd[1]: Started Nagios Remote Plugin Executor.
Sep 30 09:27:17 ip-172-31-41-41 nrpe[7349]: Starting up daemon
Sep 30 09:27:17 ip-172-31-41-41 nrpe[7349]: Server listening on 0.0.0.0 port 5666.
Sep 30 09:27:17 ip-172-31-41-41 nrpe[7349]: Listening for connections on port 5666
Sep 30 09:27:17 ip-172-31-41-41 nrpe[7349]: Allowing connections from: 127.0.0.1,::1,13.235.0.144
root@ip-172-31-41-41:/home/ubuntu#
```

ps -ef | grep nagios

```
root@ip-172-31-44-151:/home/ubuntu# ps -ef | grep nagios
nagios     55287       1  0 08:54 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios     55288   55287  0 08:54 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     55289   55287  0 08:54 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     55290   55287  0 08:54 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     55291   55287  0 08:54 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     55292   55287  0 08:54 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios     56327       1  0 08:58 ?        00:00:00 /usr/sbin/nrpe -c /etc/nagios/nrpe.cfg -f
root       60903   60158  0 09:32 pts/1    00:00:00 grep --color=auto nagios
root@ip-172-31-44-151:/home/ubuntu# sudo su
root@ip-172-31-44-151:/home/ubuntu# mkdir /usr/local/nagios/etc/objects/monitorhosts
root@ip-172-31-44-151:/home/ubuntu# mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
```

1.sudo su 2.mkdir /usr/local/nagios/etc/objects/monitorhosts 3.mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts Copy the sample localhost.cfg file to linuxhost folder 4.cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

```
root@ip-172-31-44-151:/home/ubuntu# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhos
ts/linuxserver.cfg
root@ip-172-31-44-151:/home/ubuntu# nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
```

Hitesh Rohra 47 D15A

```
  GNU nano 6.2                  /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
##############################################################################

# 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               localhost
    alias                   localhost
    address                 127.0.0.1
}



^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Location      M-U Undo         M-A Set Mark
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Justify       ^/ Go To Line    M-E Redo         M-6 Copy
```

i-03a3e79fc5ab0a056 (cutenagios_server)

```
  GNU nano 6.2                  /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg *
##############################################################################
#
# 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                 localhost               ; Comma separated list of hosts that belong to this group
}

^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Location      M-U Undo         M-A Set Mark
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Justify       ^/ Go To Line    M-E Redo         M-6 Copy
```

```
  GNU nano 6.2                                  /usr/local/nagios/etc/nagios.cfg *

# You can also tell Nagios to process all config files (with a .cfg
# extension) in a particular directory by using the cfg_dir
# directive as shown below:

#cfg_dir=/usr/local/nagios/etc/servers
#cfg_dir=/usr/local/nagios/etc/printers
#cfg_dir=/usr/local/nagios/etc/switches
#cfg_dir=/usr/local/nagios/etc/routers
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/



# OBJECT CACHE FILE
# This option determines where object definitions are cached when
# Nagios starts/restarts.  The CGIs read object definitions from
Save modified buffer?
 Y Yes
 N No            ^C Cancel
```

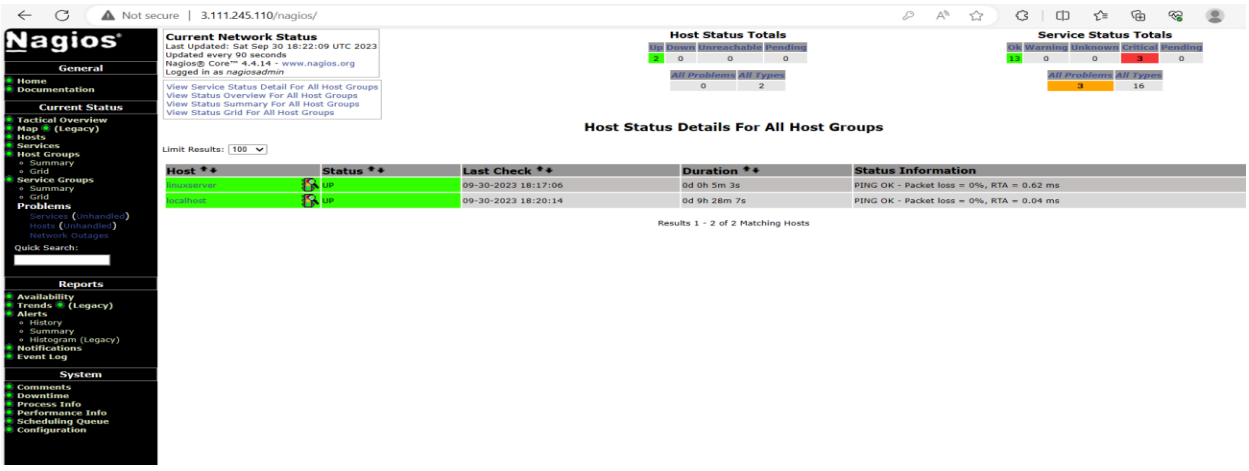
```
root@ip-172-31-44-151:/home/ubuntu# nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
root@ip-172-31-44-151:/home/ubuntu#   /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.4.14
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2023-08-01
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.
```

Hitesh Rohra 47 D15A

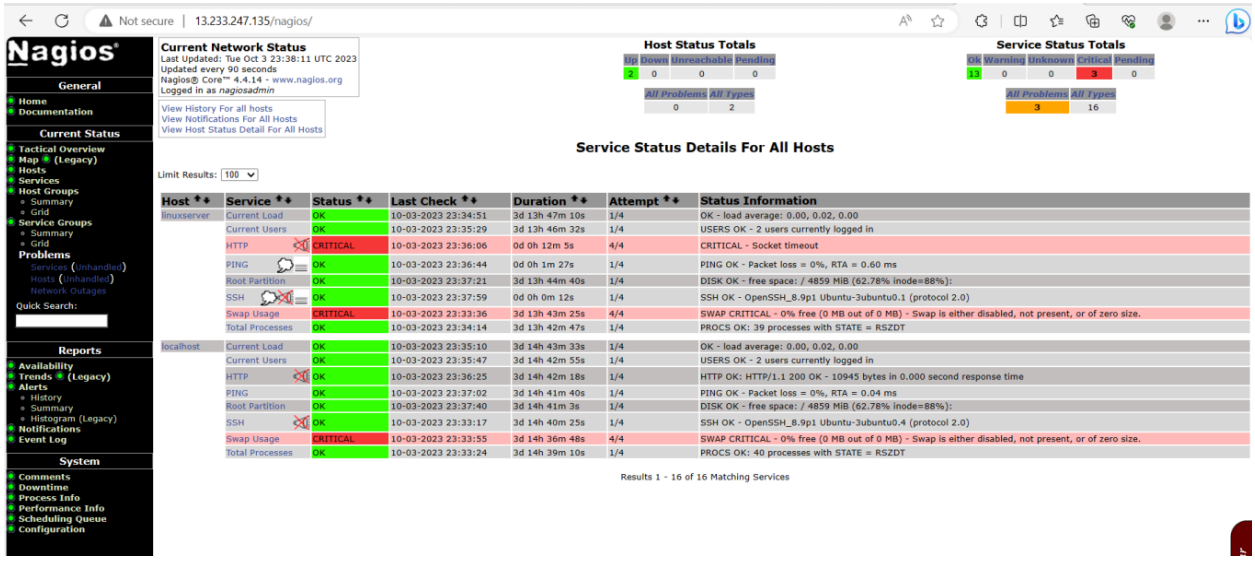
```
        Checked 1 contacts.
        Checked 1 contact groups.
        Checked 24 commands.
        Checked 5 time periods.
        Checked 0 host escalations.
        Checked 0 service escalations.
Checking for circular paths...
        Checked 1 hosts
        Checked 0 service dependencies
        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-172-31-44-151:/home/ubuntu# nano /usr/local/nagios/etc/nagios.cfg
```

Sudo systemctl status nagios

```
● nagios.service - Nagios Core 4.4.14
     Loaded: loaded (/lib/systemd/system/nagios.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2023-09-30 08:54:01 UTC; 20s ago
       Docs: https://www.nagios.org/documentation
    Process: 55285 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 55286 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 55287 (nagios)
      Tasks: 6 (limit: 1141)
     Memory: 5.3M
        CPU: 252ms
     CGroup: /system.slice/nagios.service
             ├─55287 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─55288 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─55289 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─55290 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─55291 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─55292 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 30 08:54:01 ip-172-31-44-151 nagios[55287]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully initialized
lines 1-19
```



Hitesh Rohra 47 D15A



Experiment No 11

Advanced DevOps Exp-11

Hitesh Rohra
D15A 47

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

Theory:-

AWS Lambda

AWS Lambda is a serverless computing service provided by Amazon Web Services (AWS). Users of AWS Lambda create functions, self-contained applications written in one of the supported languages and runtimes, and upload them to AWS Lambda, which executes those functions in an efficient and flexible manner.The Lambda functions can perform any kind of computing task, from serving web pages and processing streams of data to call APIs and integrate with other AWS services.

The concept of “serverless” computing refers to not needing to maintain your own servers to run these functions. AWS Lambda is a fully managed service that takes care of all the infrastructure for you.

Features of AWS Lambda

● AWS Lambda easily scales the infrastructure without any additional configuration. It reduces the operational work involved.

● It offers multiple options like AWS S3, CloudWatch, DynamoDB, API Gateway, Kinesis, CodeCommit, and many more to trigger an event.

● You don’t need to invest upfront. You pay only for the memory used by the lambda function and minimal cost on the number of requests hence cost-efficient.

● AWS Lambda is secure. It uses AWS IAM to define all the roles and security policies.

● It offers fault tolerance for both services running the code and the function. You do not have to worry about the application down.

Packaging Functions

Lambda functions need to be packaged and sent to AWS. This is usually a process of compressing the function and all its dependencies and uploading it to an S3 bucket. And letting AWS know that you want to use this package when a specific event takes place.

To help us with In this process we use the Serverless Stack Framework (SST). We'll go over this in detail later on in this guide.

Execution Model

The container (and the resources used by it) that runs our function is managed completely by AWS. It is brought up when an event takes place and is turned off if it is not being used. If additional requests are made while the original event is being served, a new container is brought up to serve a request. This means that if we are undergoing a usage spike, the cloud provider simply creates multiple instances of the container with our function to serve those requests.

This has some interesting implications. Firstly, our functions are effectively stateless. Secondly, each request (or event) is served by a single instance of a Lambda function. This means that you are not going to be handling concurrent requests in your code. AWS brings up a container whenever there is a new request. It does make some optimizations here. It will hang on to the container for a few minutes (5 - 15 mins depending on the load) so it can respond to subsequent requests without a cold start.

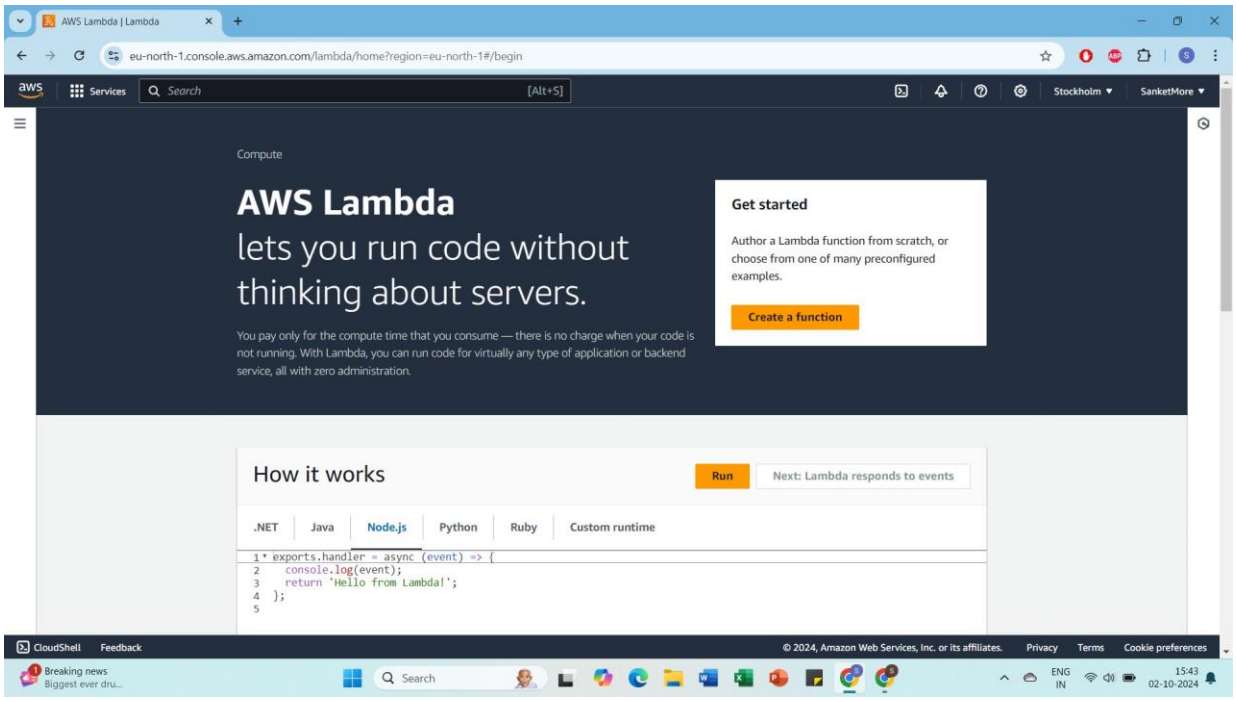
Stateless Functions

The above execution model makes Lambda functions effectively stateless. This means that every
time your Lambda function is triggered by an event it is invoked in a completely new
environment. You don't have access to the execution context of the previous event. However, due to the optimization noted above, the actual Lambda function is invoked only once

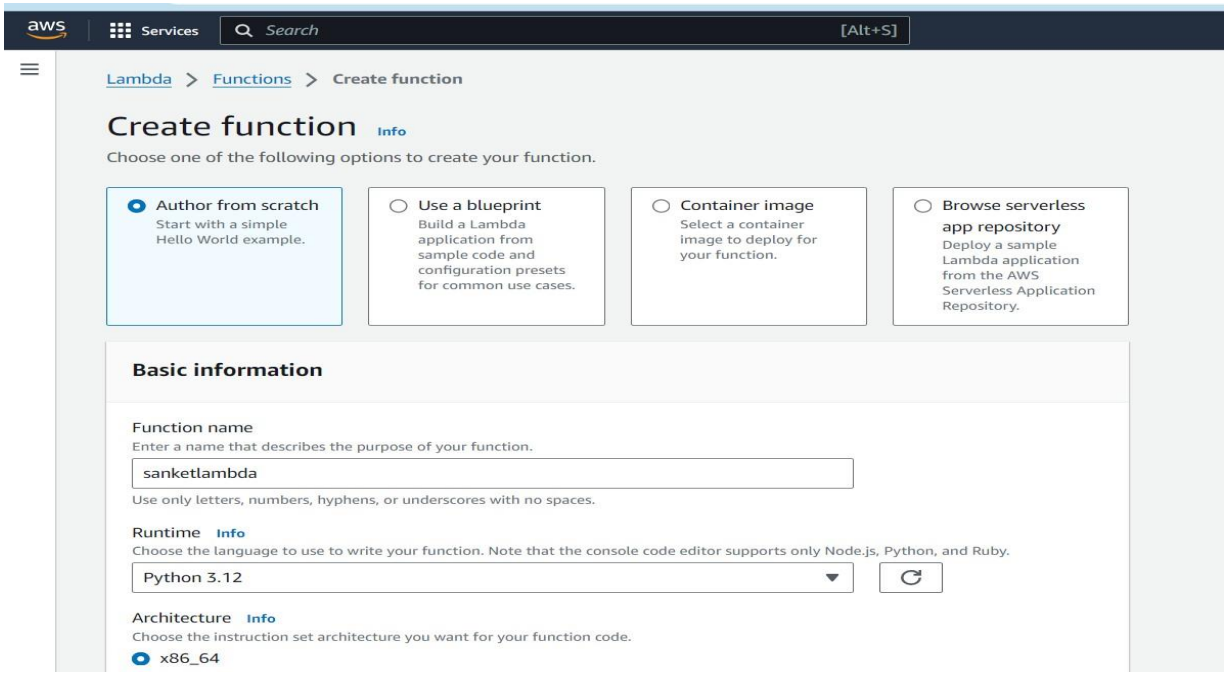
per container instantiation. Recall that our functions are run inside containers. So when a function is first invoked, all the code in our handler function gets executed and the handler
function gets invoked. If the container is still available for subsequent requests, your function
will get invoked and not the code around it.

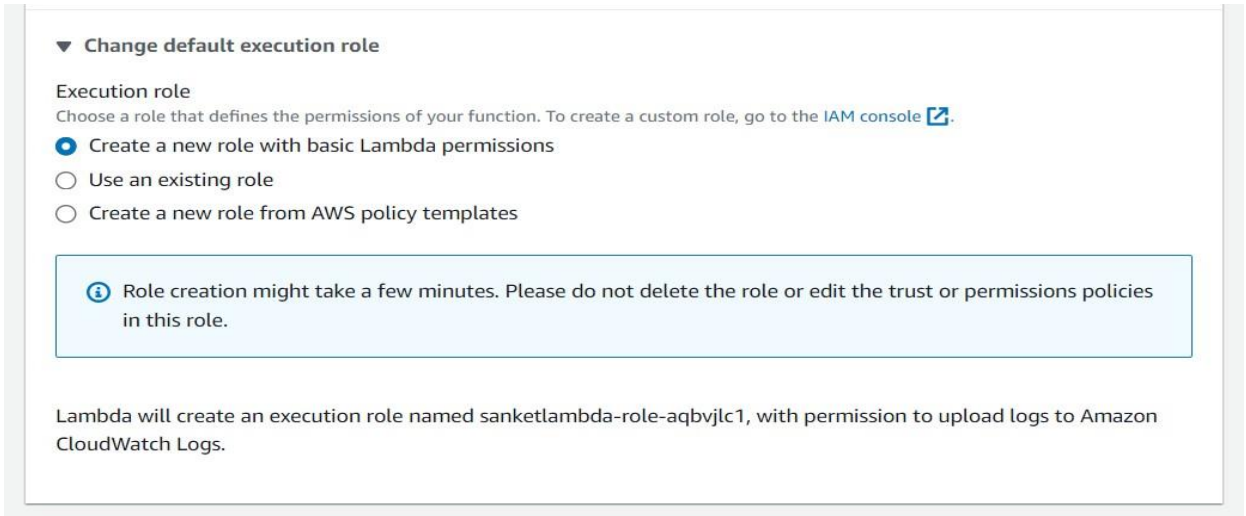
Procedure:-

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.

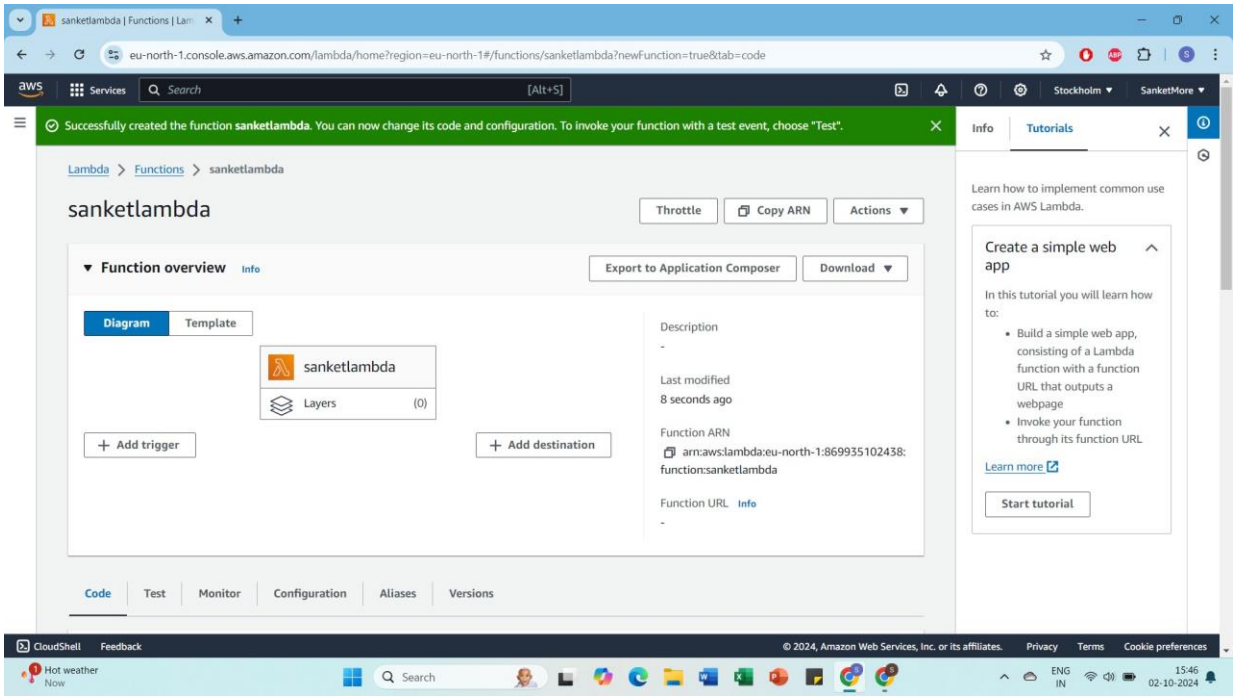


2. Choose to create a function from scratch or use a blueprint, i.e templates defined by AWSfor you with all configuration presets required for the most common use cases.
Then, choose a runtime env for your function, under the dropdown, you can see all the options AWS supports, Python, Nodejs, .NET and 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.

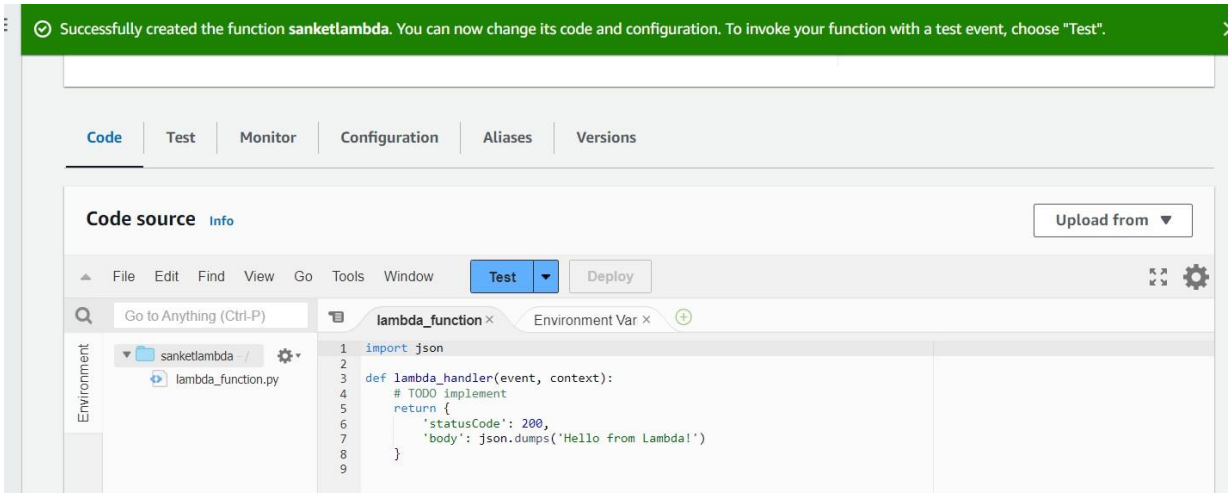




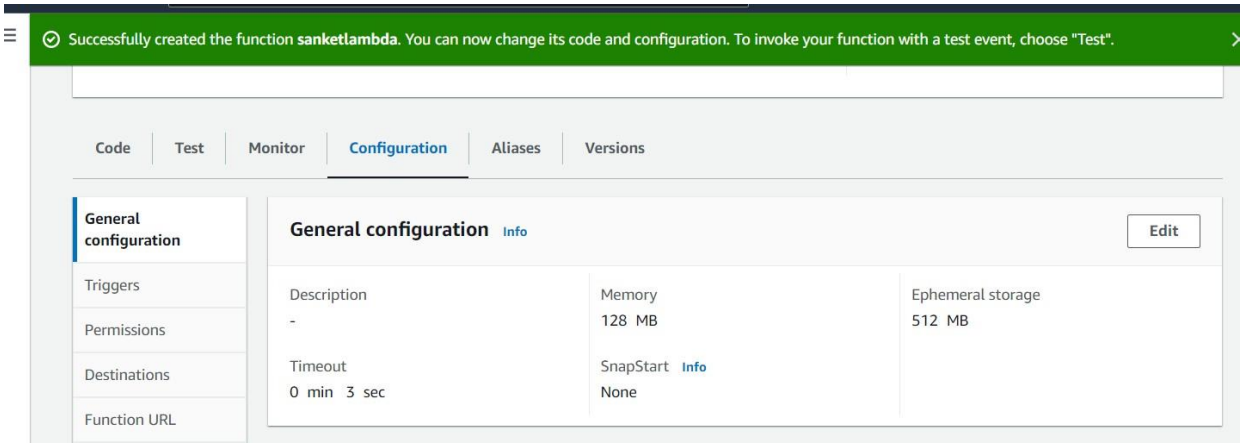
3. This process will take a while to finish and after that, you'll get a message that your function was successfully created.

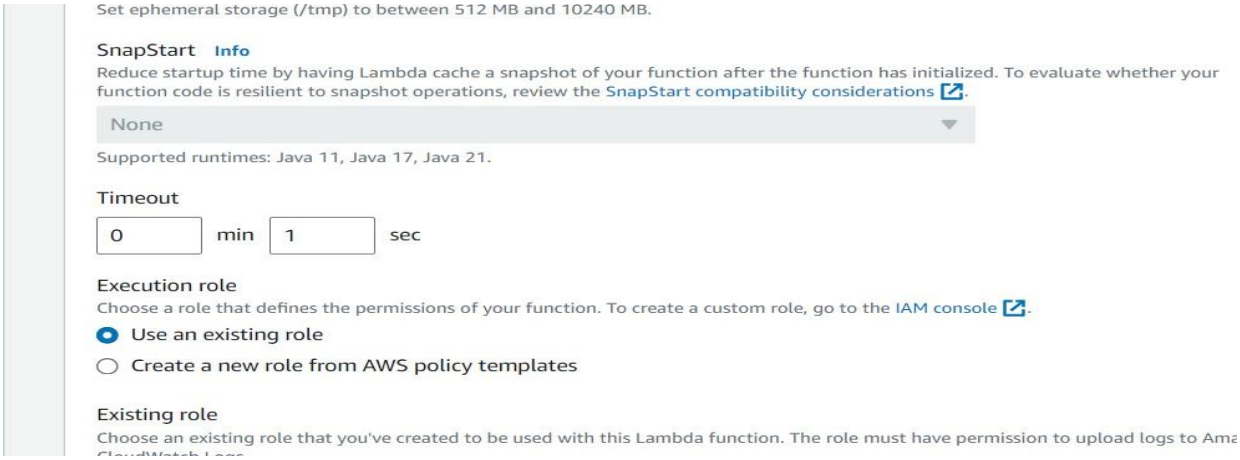


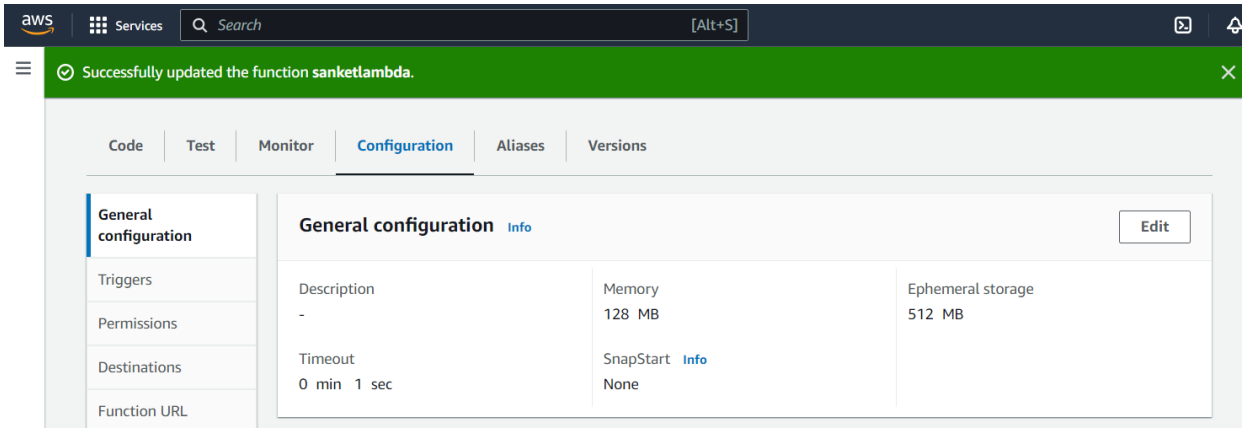
4. You can make changes to your function inside the code editor. You can also upload a zip
file of your function or upload one from an S3 bucket if needed.
Press Ctrl + S to save the file and click Deploy to deploy the changes.



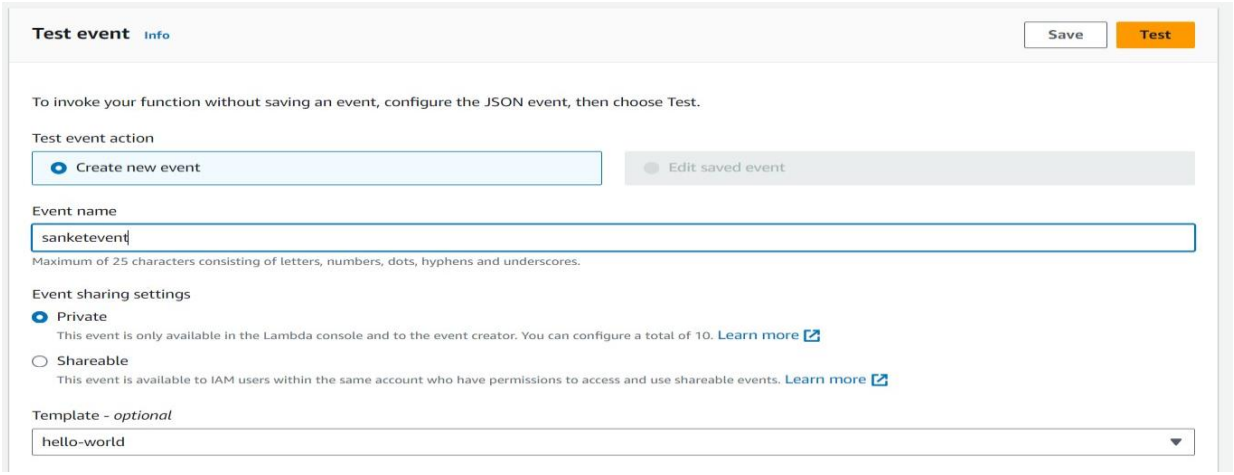
5. To change the configuration, open up the Configuration tab and under General Configuration, choose Edit.
Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.



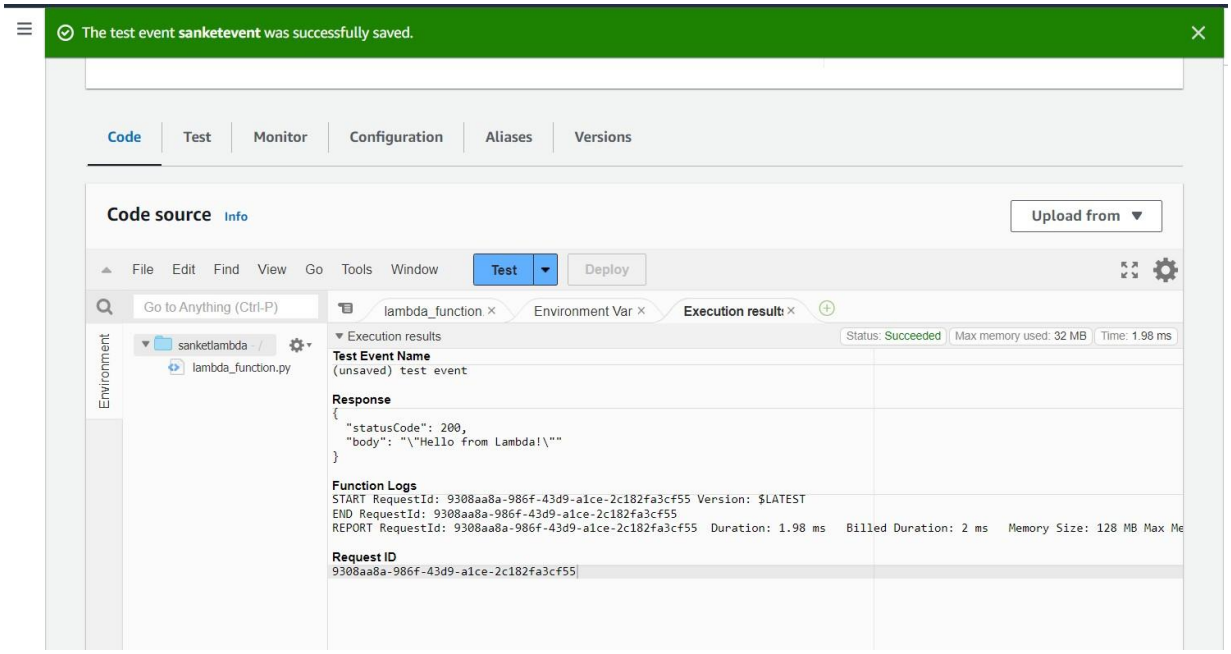




6. Click on Test and you can change the configuration, like so. If you do not have anything in the request body, it is important to specify two curly braces as valid JSON, so make sure they are there.



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



Experiment No 12

Advanced DevOps Exp-12

Hitesh Rohra
D15A 47

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.

Theory:

AWS Lambda and S3 Integration: AWS Lambda allows you to execute code in response to various events, including those triggered by Amazon S3. When an object is added to an S3 bucket, it can trigger a Lambda function to execute, allowing for event-driven processing without managing servers.

Workflow:

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.

3. Set Up Permissions:

○ Ensure that the Lambda function has the necessary permissions to access S3. You can do this by attaching an IAM role with policies that allow reading from the bucket and writing logs to CloudWatch.

4. Configure S3 Trigger:

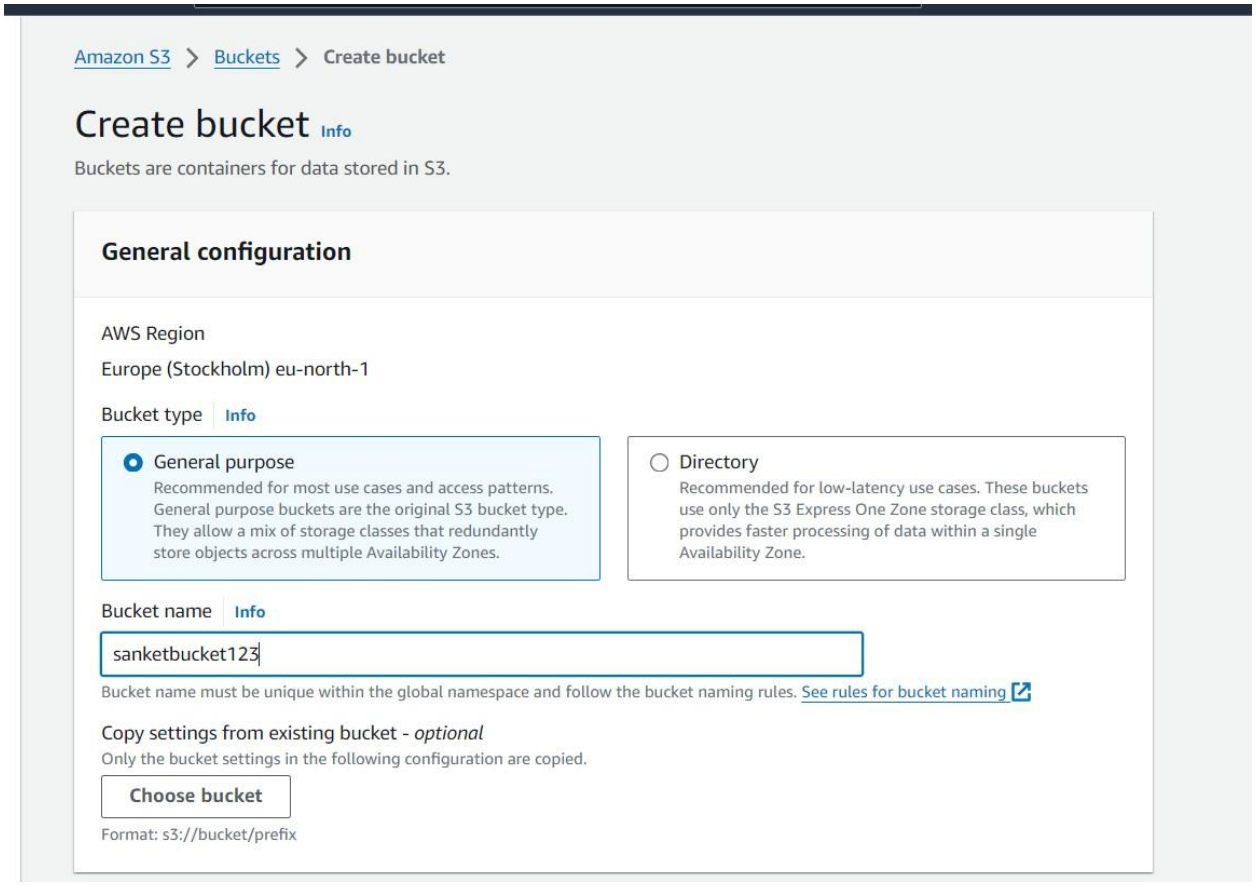
○ Link the S3 bucket to the Lambda function by setting up a trigger. Specify that the function should be triggered when an object is created in the bucket (e.g., when an image is uploaded).

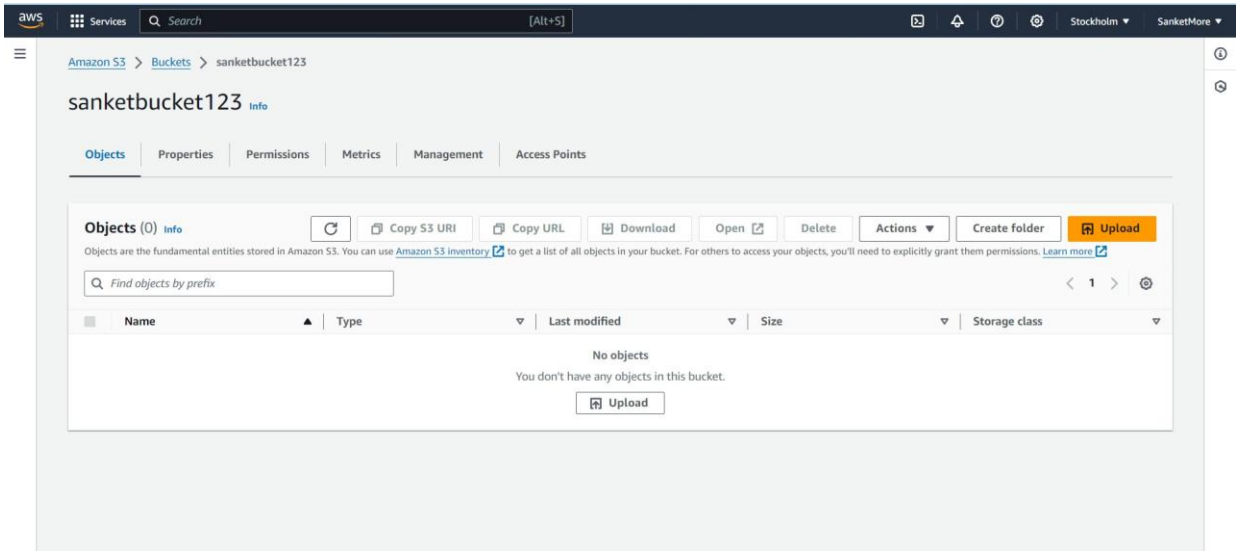
5. Test the Setup:

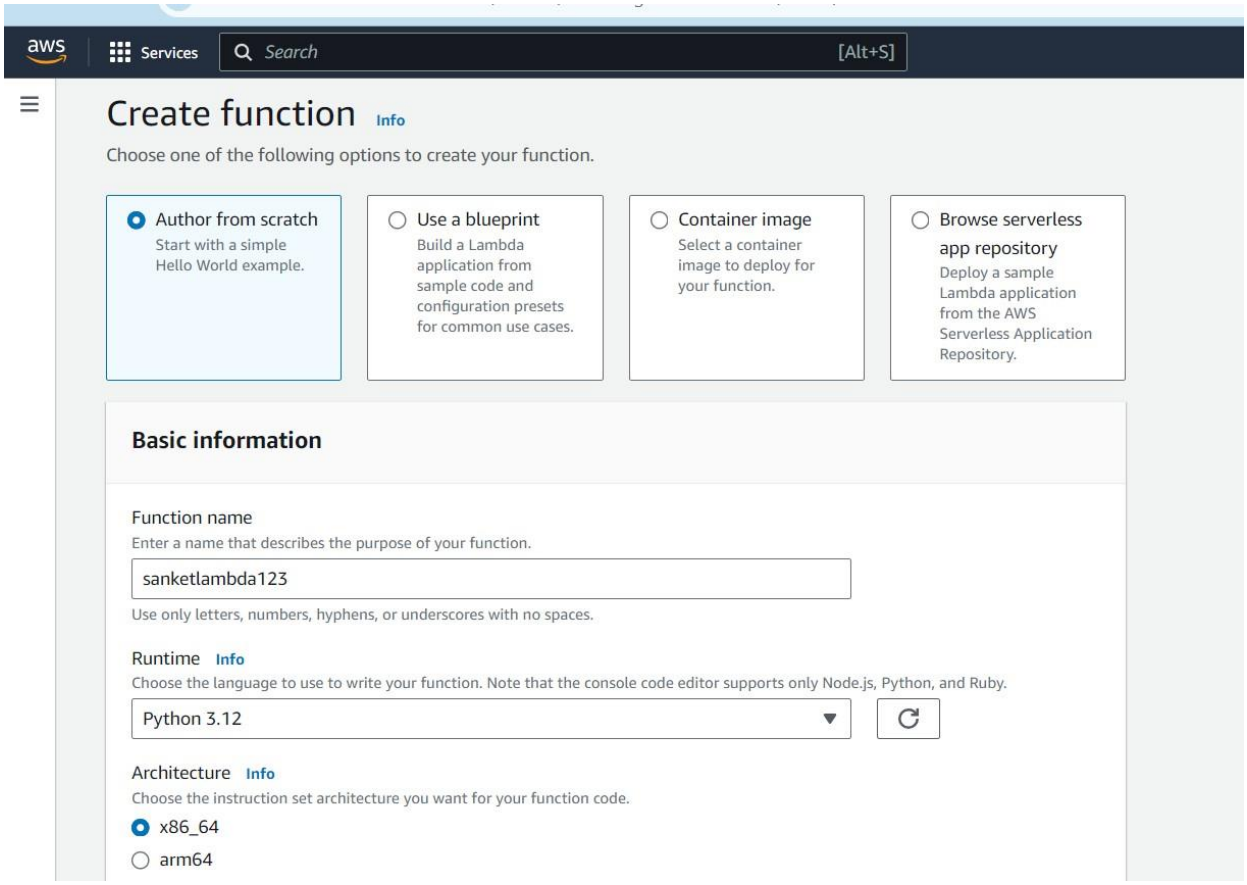
○ Upload an object (e.g., an image) to the S3 bucket to test the trigger. The Lambda function should execute and log the message “An Image has been added” in AWS CloudWatch Logs.

Procedure:-

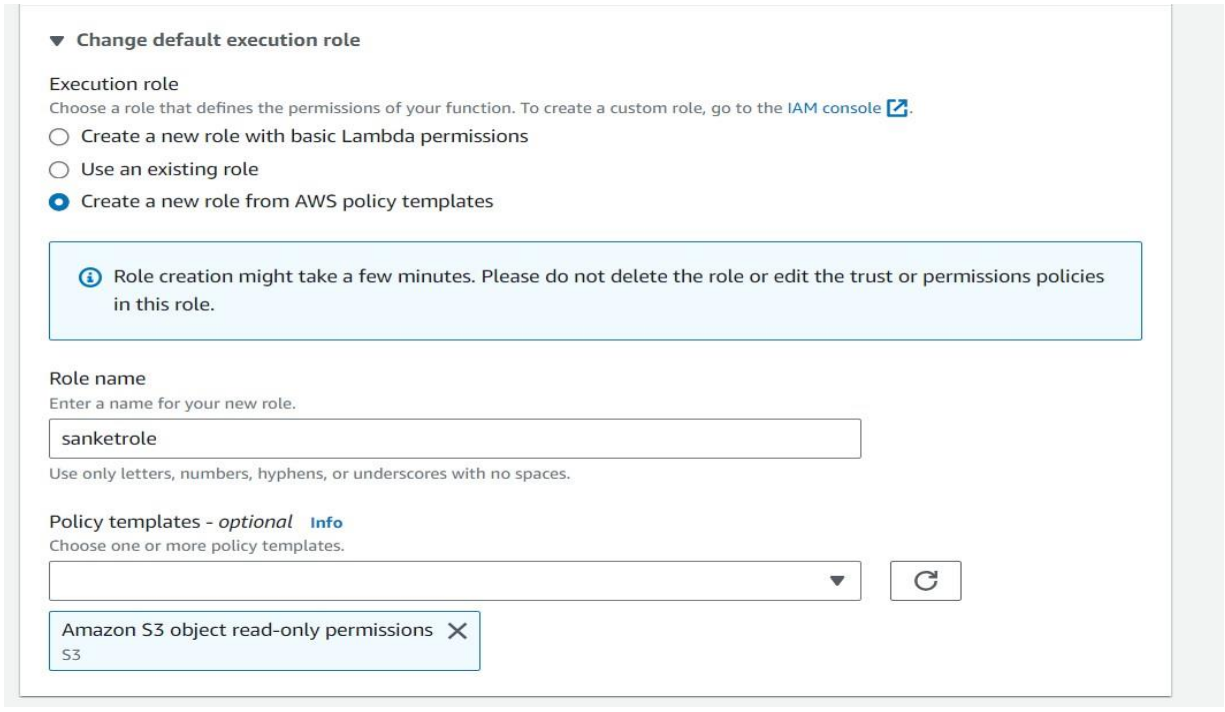
1. Create an S3 bucket of the same location as that of the Lambda function

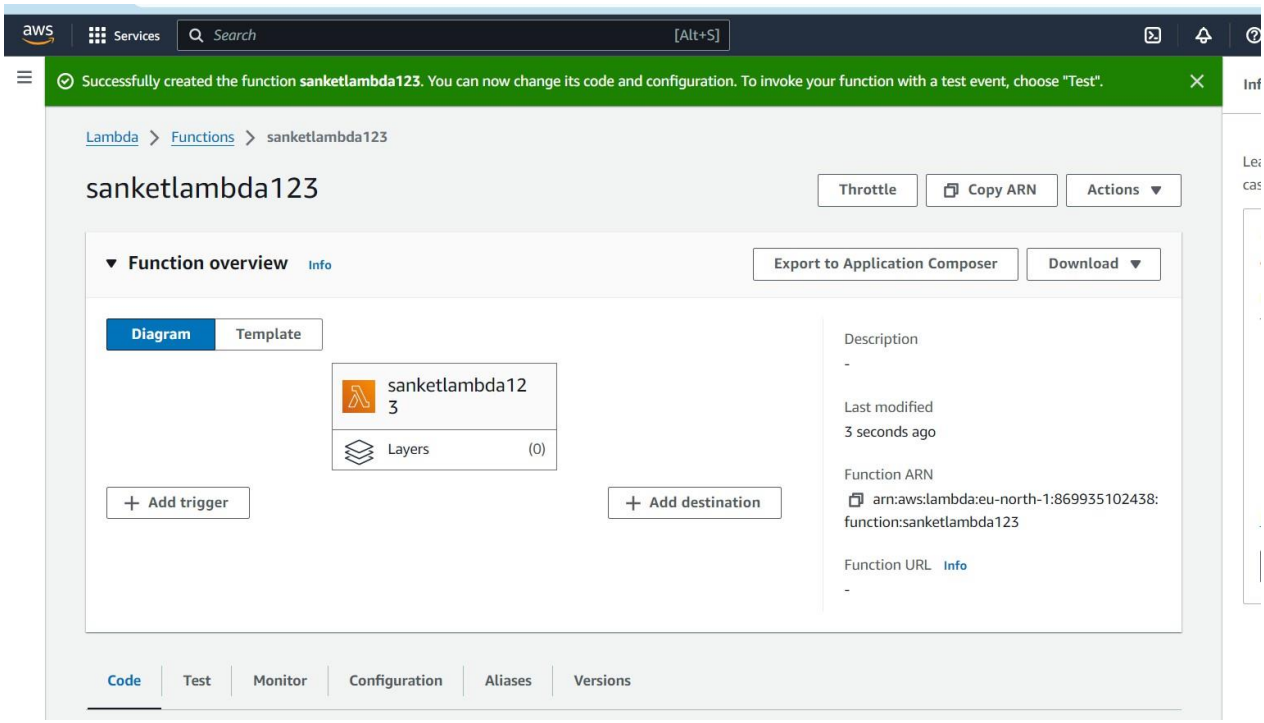




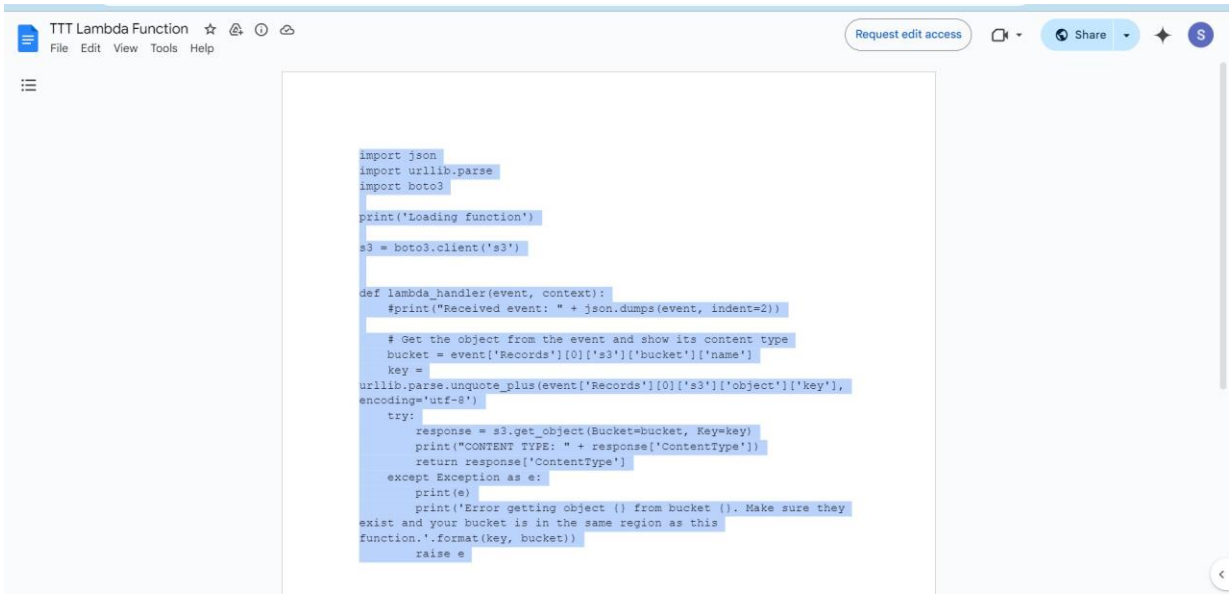


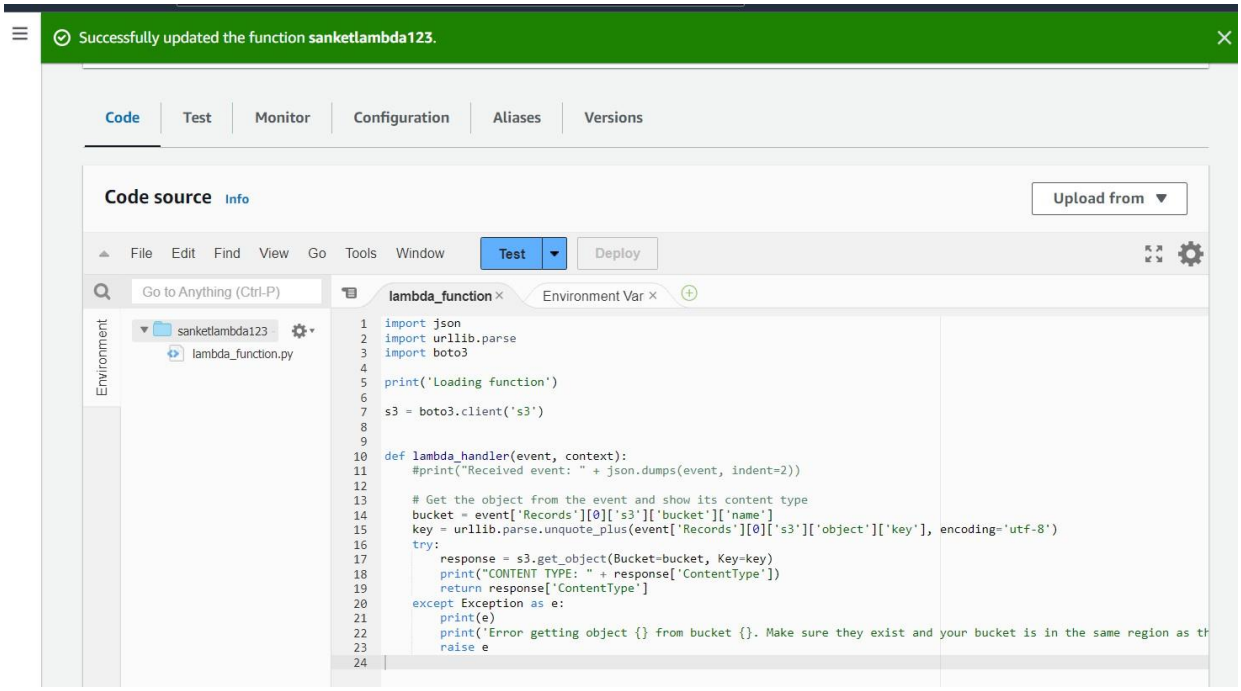
2. Add roles while creating the Lambda function and give permissions for accessing the S3 bucket



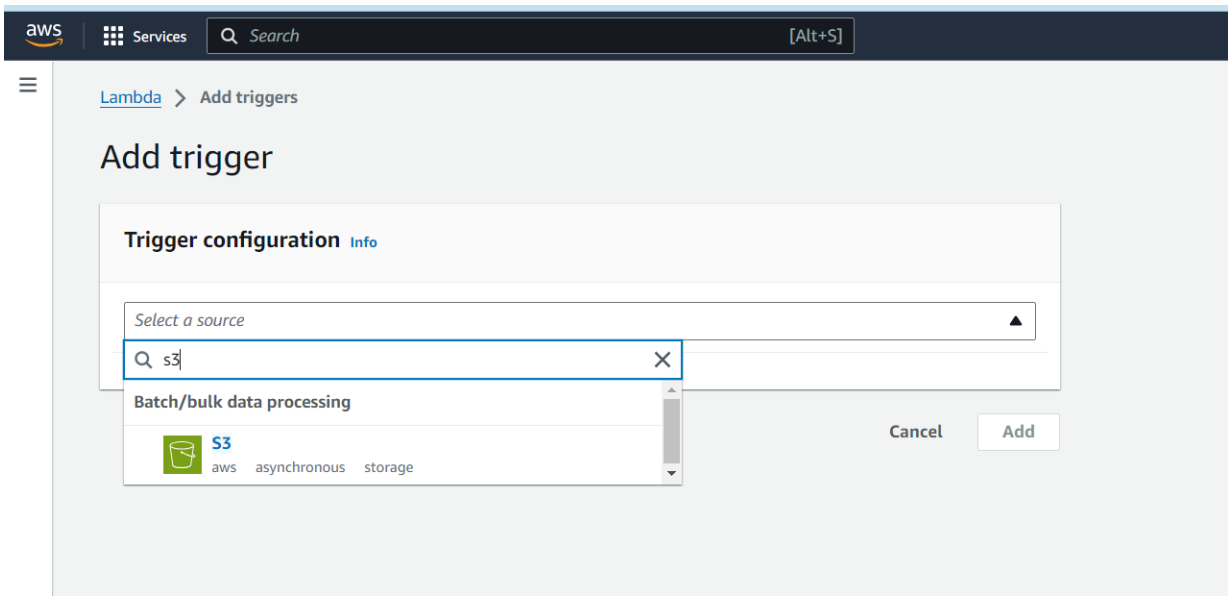


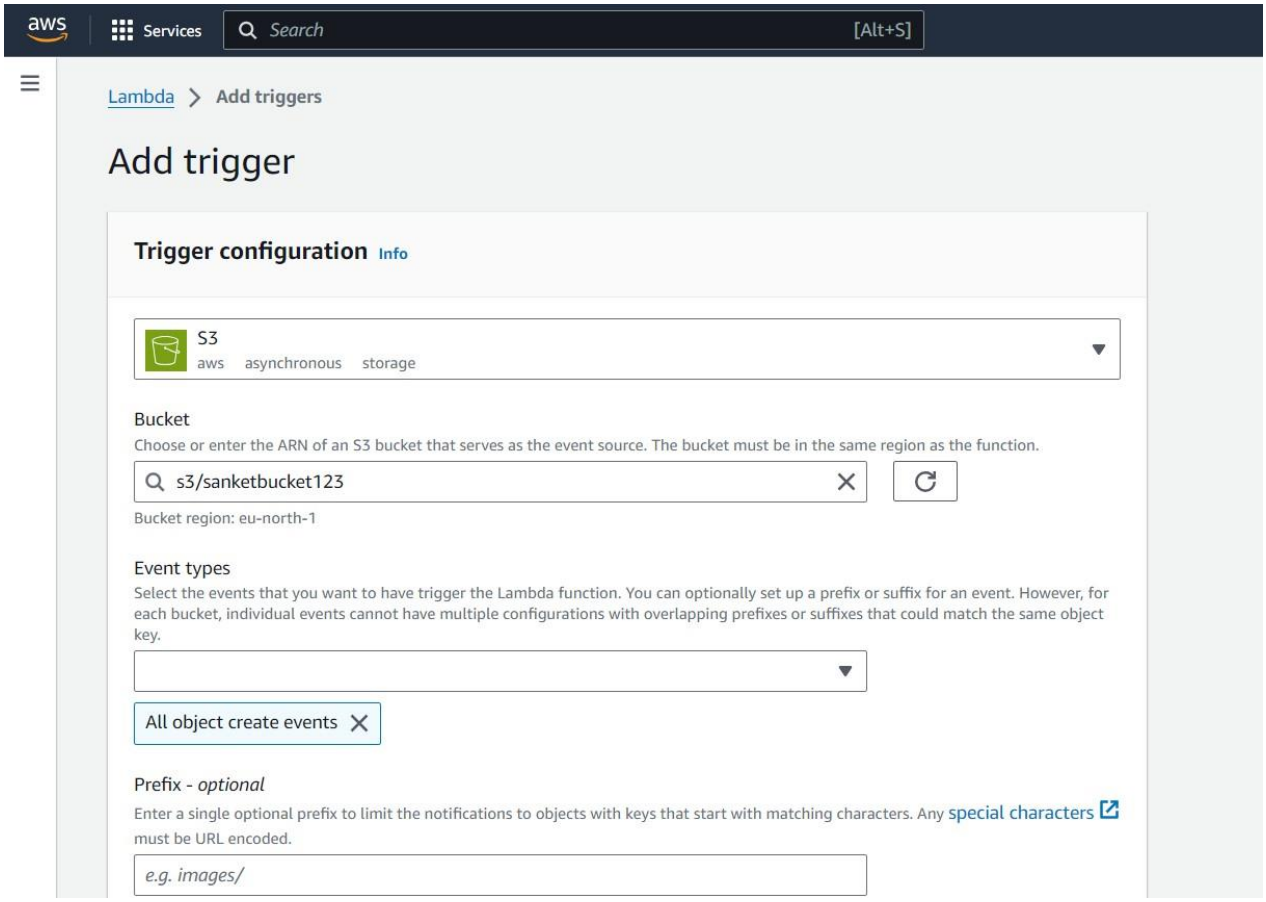
3. After creating the Lambda function copy a code available on the internet which allows the Lambda function to access the S3 bucket contents.

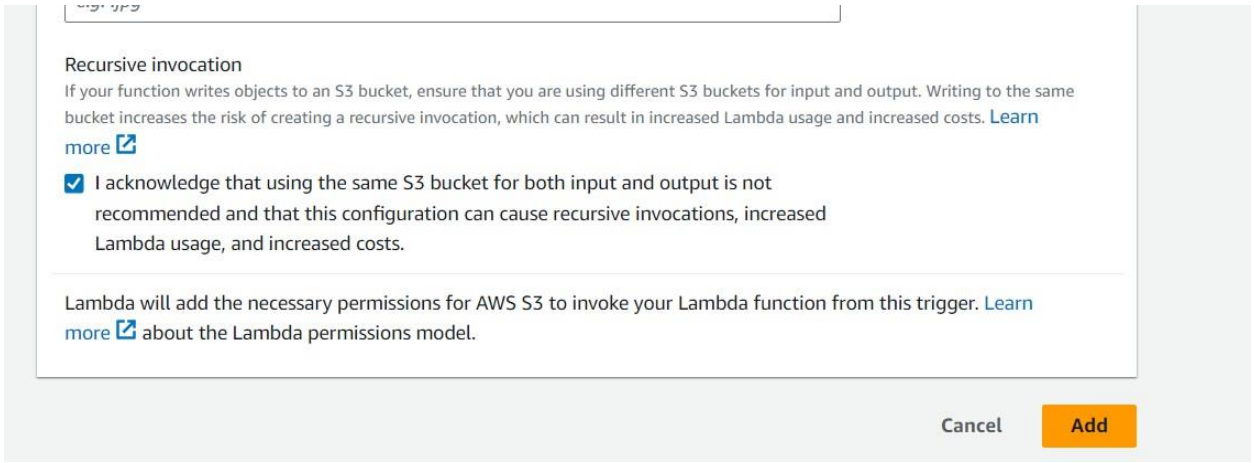


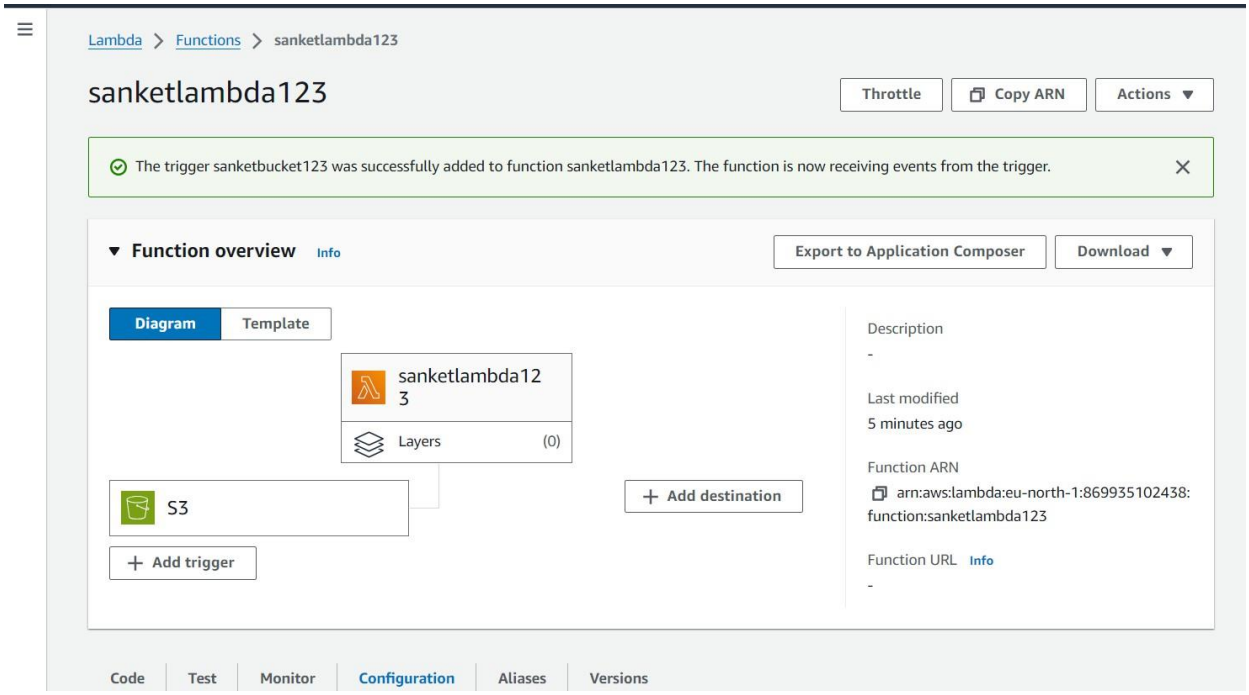


4. Add a trigger to the Lambda function so any changes in the S3 bucket will be first visible to the user.

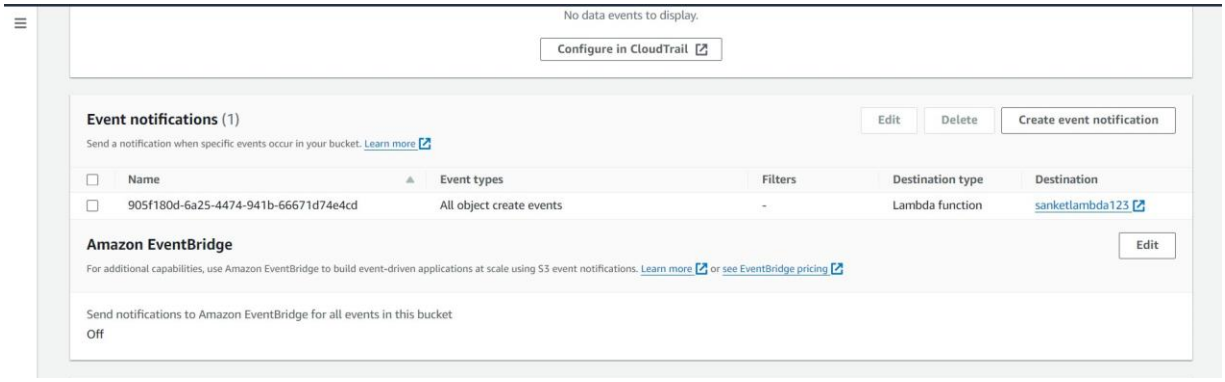


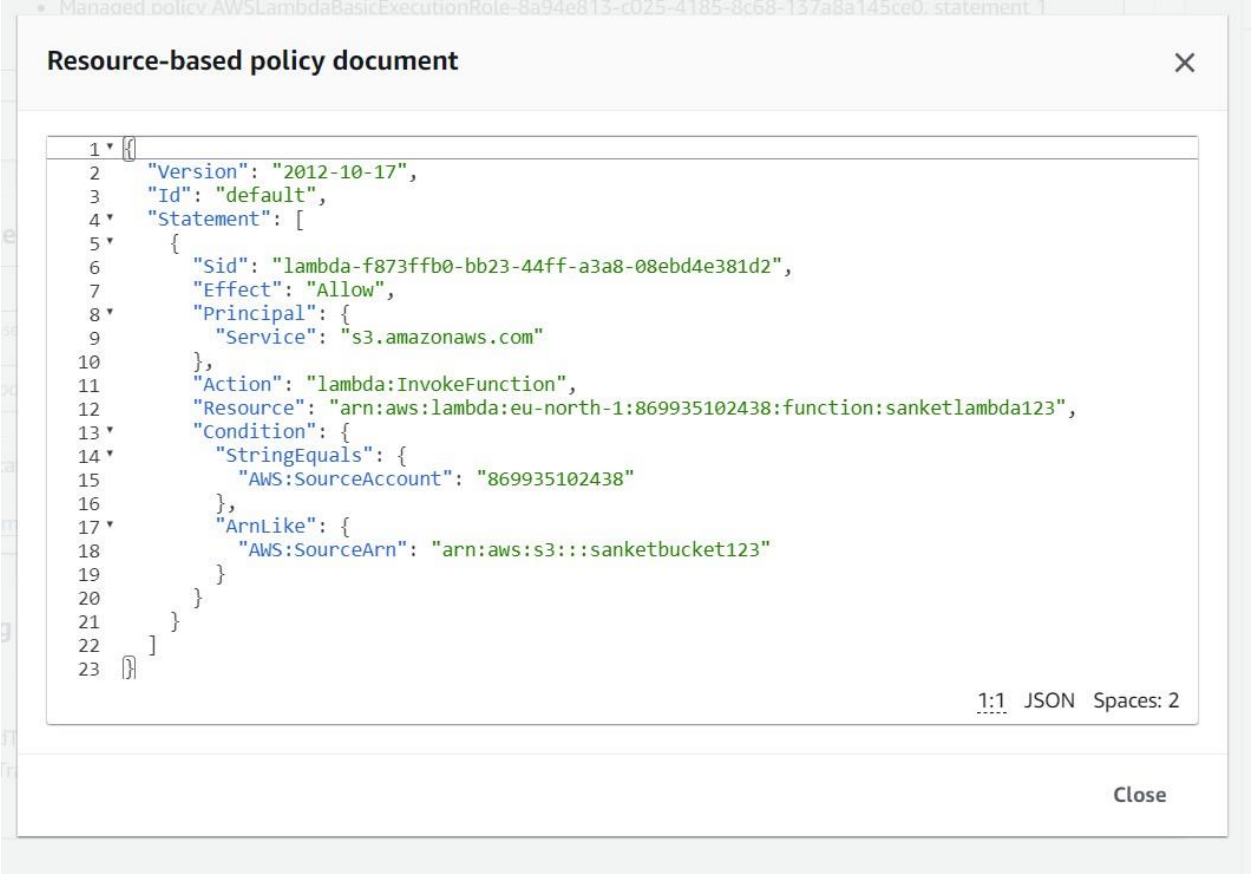

aws
Services
Search
[Alt+S]
Lambda
Add triggers
Add trigger
Trigger configuration
Info
S3
aws
asynchronous
storage
Bucket
Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.
s3/sanketbucket123
Bucket region: eu-north-1
Event types
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.
All object create events
Prefix - optional
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters. Any special characters must be URL encoded.
e.g. images/


Recursive invocation
If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. Learn more
I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.
Lambda will add the necessary permissions for AWS S3 to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.
Cancel
Add

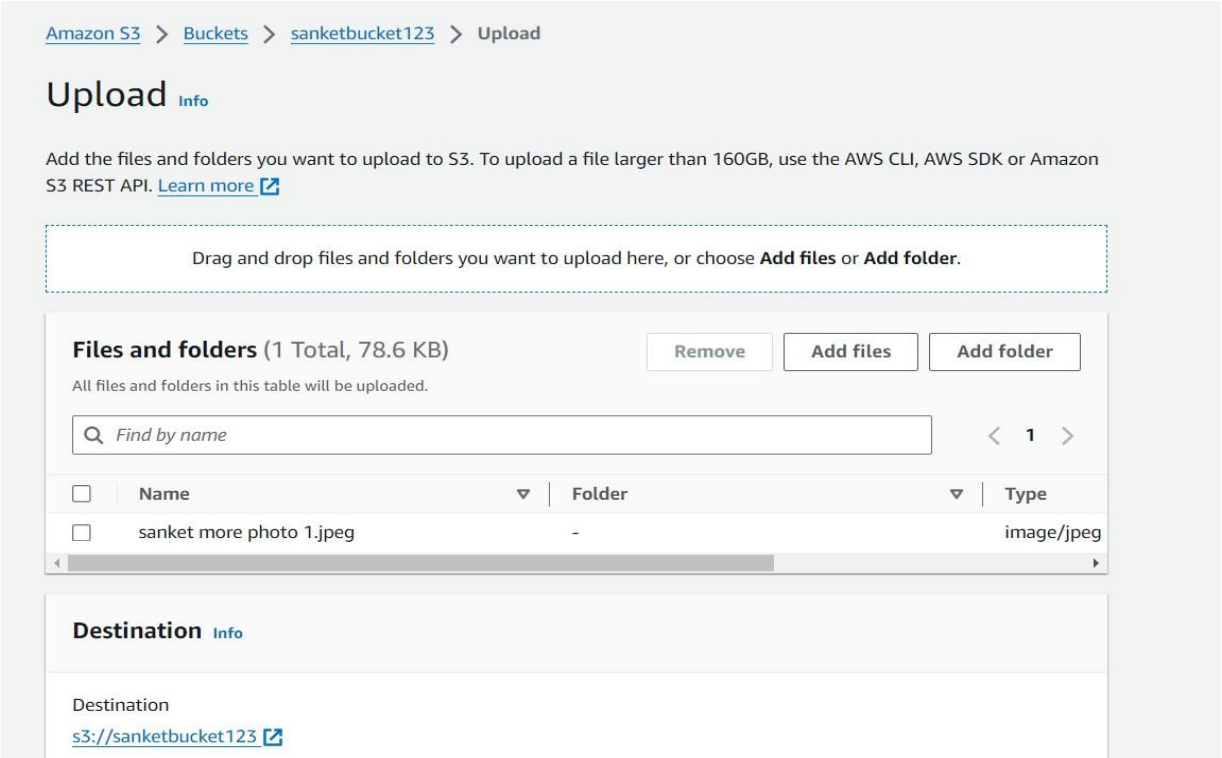


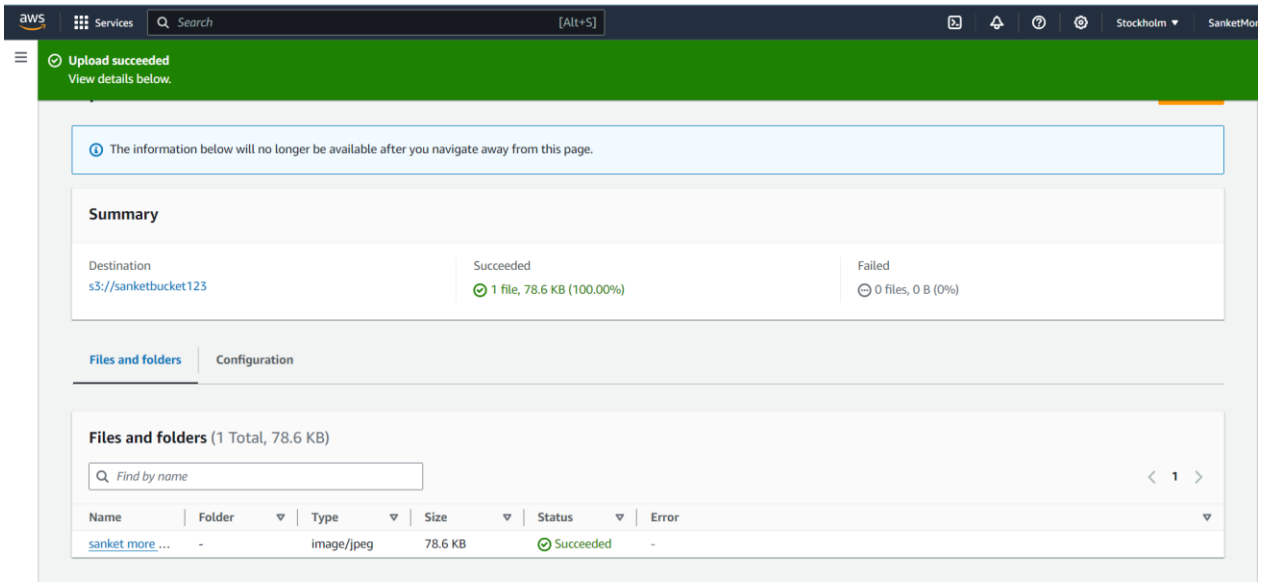
5. In the event notification of the S3 bucket we can see that it has been connected to the Lambda function .



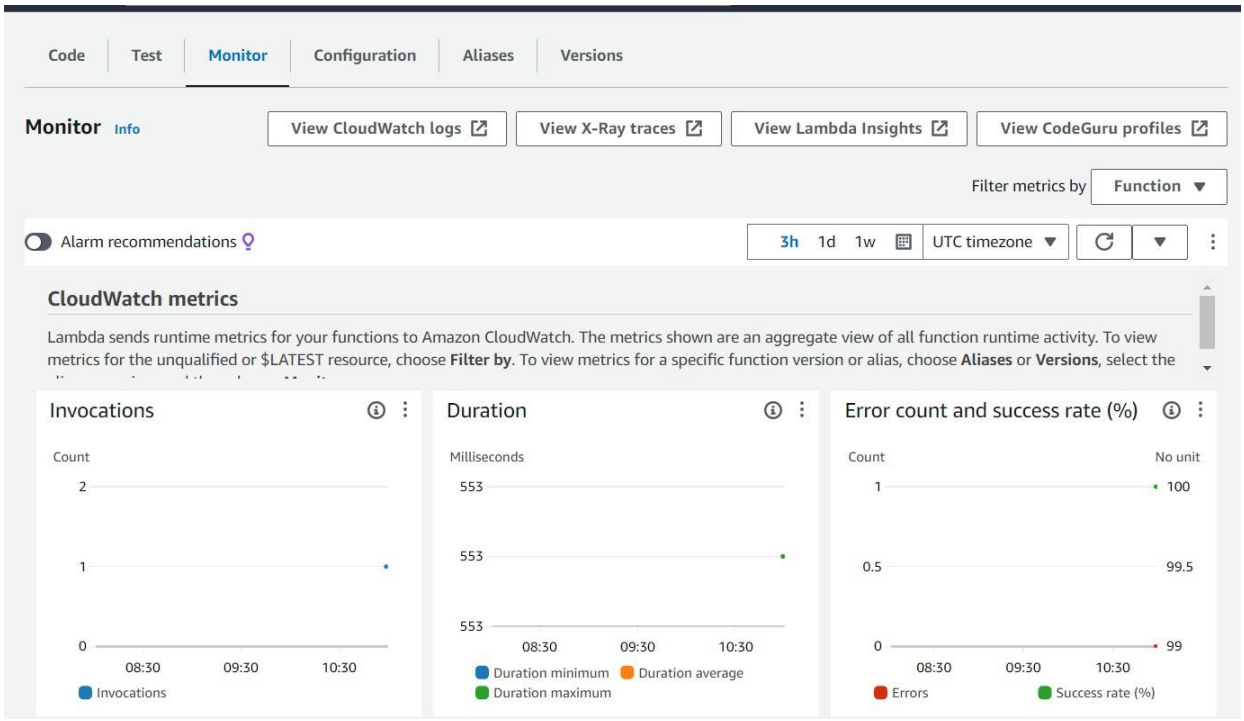


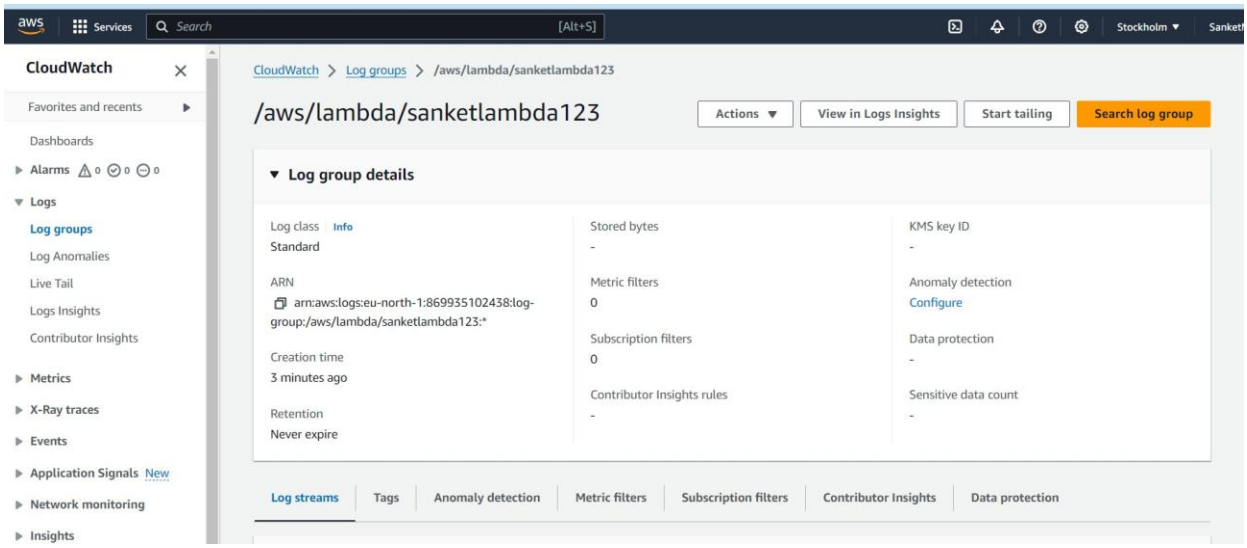
6. Upload a photo to the S3 bucket

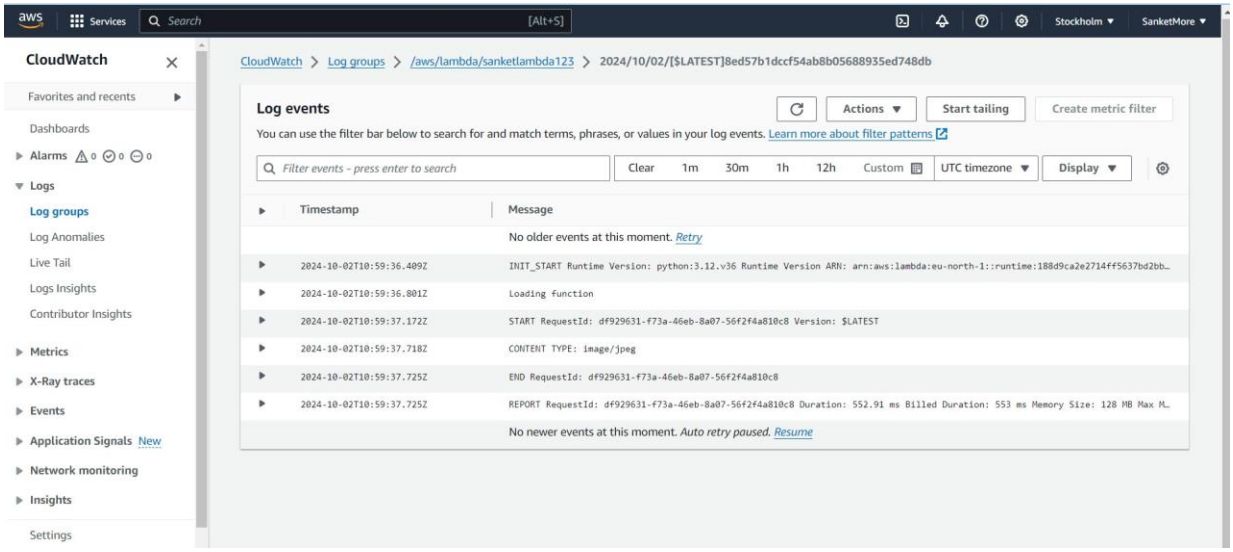




7. Now run the function and in the cloud watch logs of AWS you can see the message printed and all the other details of the working of the Lambda function.




aws
Services
Search
[Alt+S]
Stockholm
CloudWatch
Favorites and recents
Dashboards
Alarms
Logs
Log groups
Log Anomalies
Live Tail
Logs Insights
Contributor Insights
Metrics
X-Ray traces
Events
Application Signals New
Network monitoring
Insights
CloudWatch > Log groups > /aws/lambda/sanketlambda123
/aws/lambda/sanketlambda123
Actions
View in Logs Insights
Start tailing
Search log group
Log group details
Log class Info
Standard
ARN
arn:aws:logs:eu-north-1:869935102438:log-group:/aws/lambda/sanketlambda123:*
Creation time
3 minutes ago
Retention
Never expire
Stored bytes
-
Metric filters
0
Subscription filters
0
Contributor Insights rules
-
KMS key ID
-
Anomaly detection
Configure
Data protection
-
Sensitive data count
-
Log streams
Tags
Anomaly detection
Metric filters
Subscription filters
Contributor Insights
Data protection


aws
Services
Search
[Alt+S]
Stockholm
SanketMore
CloudWatch
Favorites and recents
Dashboards
Alarms
Logs
Log groups
Log Anomalies
Live Tail
Logs Insights
Contributor Insights
Metrics
X-Ray traces
Events
Application Signals New
Network monitoring
Insights
Settings
CloudWatch > Log groups > /aws/lambda/sanketlambda123 > 2024/10/02/[$LATEST]8ed57b1dccf54ab8b05688935ed748db
Log events
Actions
Start tailing
Create metric filter
You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns
Filter events - press enter to search
Clear
1m
30m
1h
12h
Custom
UTC timezone
Display
Timestamp
Message
No older events at this moment. Retry
2024-10-02T10:59:36.409Z INIT_START Runtime Version: python:3.12.v36 Runtime Version ARN: arn:aws:lambda:eu-north-1::runtime:188d9ca2e2714ff5637bd2bb…
2024-10-02T10:59:36.801Z Loading function
2024-10-02T10:59:37.172Z START RequestId: df929631-f73a-46eb-8a07-56f2f4a810c8 Version: $LATEST
2024-10-02T10:59:37.718Z CONTENT TYPE: image/jpeg
2024-10-02T10:59:37.725Z END RequestId: df929631-f73a-46eb-8a07-56f2f4a810c8
2024-10-02T10:59:37.725Z REPORT RequestId: df929631-f73a-46eb-8a07-56f2f4a810c8 Duration: 552.91 ms Billed Duration: 553 ms Memory Size: 128 MB Max M…
No newer events at this moment. Auto retry paused. Resume

Assignment No 1

D15A - 47 Mitesh Rohra

04/05 R. DATE:

Q1) Use S3 bucket and host video streaming

-> 1) log into AWS console

- Go to AWS management console
- Enter your login credentials.

2) create an S3 bucket

- In the console search for S3 in the search bar and select S3 from the results.
- click create bucket
- Give your bucket a unique name.
- choose a region
- scroll down and uncheck Block all public access
- confirm by cheking the acknowledgement box
- click create bucket.

3) upload your videos file to S3

- click on your newly created bucket
- click on the upload button
- Add your video file from compiter
- click upload to start ~~file~~ the upload process.

DATE:

Q2) Discuss BMW and Hot Star case studing using Aws.

-> BMW : A connected driving Expreince BMW is a leader in car technology and AWS has helped them a lot in creating connected cars

- connected cars features : BMW has developed features like remote diagonastic over the air- updates and real- time traffic ~~over the~~ info Aws provides the necessary tools and infrastructure to make the features. works.
- Improving customer experience : They use Aws to enhance the customer internet with their cars for example BMW has added voice activated assistance and personalized suggestions using services like Amazon lex.
- Better manufactering : BMW uses Aws to improve its manufactering By analyzing data from machines they can find problem early and increase efficiency
- A streaming revolution :- Hotstar is India's biggest video streaming platforms offering a huge range of movies, TV shows and sports AWS has been key of Hotstar sucess helping them.
- Manage huge traffic : During big events like IPL many people watch at once. Aws helps hotstar handle this stage in traffic without slowing down.

DATE:

set permission for public Access.
- confirm the action by clicking make public again

select the video url

- After making the file public click on the radio file
- you will see a url for the video under Objet URL

6.) Edit the Bucket policy

```
{
  "version": 2012-10-17",
  "statement": [
    {
      "sid": PublicReadGetObjet"
      "Efect": "Allow",
      "Principal": " ",
      "Action": "s3 GetObjet"
      "Resource": "arn: aws: s3:: video-
                  bucket /*"
    }
  ]
}
```

DATE:

- Qvality streaming : Aws services like Amazon Kinesis ensures that users highly stream qvality.

- Personalized Recommendation :- Hotstar uses Aws machine learning tools to suggest contents to users making their viewing expirence more enjoyable

Key AWS services :-

- Compute : Amazon EC2
- Storage :- Amazon S3, Amazon EPS

Q3) why Kubernets and Advantages and disadvantages of Kubernates. Explain How adidas uses Kubernates

-> Kubernets is popular because its simplifies the management of containerized applications It automates tasks such as deployment, scaling and monitoring, mating, it easier for organizations to manage their applications in a cloud environment.

Advantages of Kubernetes :-

1) portability :- Application can be moved easily between different environment without major changes.

DATE:

2) scalability :- Kubernetes can automatically scale apps up or down based on traffic and demand

3) Reliability :- It features self-healing capabilities meaning it can restart failed containers and balance workload.

Disadvantages of kubernates :-

1) complexity :- It can be complicated to setup and manage espically for those new to container

2) stop learning curve :- Requires time and knowledge to fully understand and utilizes its features.

3) Resource Intensive :- It may require more computing resources than simpler solutions.

Adidas has adopted kubernetes to enhance its IT infrastructure and improve its ability to respond to market needs.

Q4) what are Nagios and Explain how Nagios are used in E-services?

-> Nagios is an open source monitoring tools that helps the organization keeps track of their It infrastructure including servers networks and application. It provides a way to ensure that

DATE:

2) scalability :- Kubernetes can automatically scale apps up or down based on traffic and demand

3) Reliability :- It features self-healing capabilities meaning it can restart failed containers and balance workload.

Disadvantages of kubernates :-

1) complexity :- It can be complicated to setup and manage espically for those new to container

2) stop learning curve :- Requires time and knowledge to fully understand and utilizes its features.

3) Resource Intensive :- It may require more computing resources than simpler solutions.

Adidas has adopted kubernetes to enhance its IT infrastructure and improve its ability to respond to market needs.

Q4) what are Nagios and Explain how Nagios are used in E-services?

-> Nagios is an open source monitoring tools that helps the organization keeps track of their It infrastructure including servers networks and application. It provides a way to ensure that

Assignment No 2

Adv Devops Assignment 2

Q Create a REST API with serverless framework

-> creating REST API with serverless framework is an afficient way to deploy serverless ~~framework is an afficient~~ applications that can scale automatically without managing server it serverless framework :- A powerful tool that deployment of services and serverless applications across various cloud periods such as AWS, And Google cloud

ii) serverless architecture :- This design model allows developers to build applications without warrying about underlying infrastructure enabling focus on code & bussiness logic

iii) REST API :- Representational state transfer is architecture style for designing network applications

Slips for creating REST API for serverless framework

i) Inshell serverless framework

you start by installing serverless framework CCI globaly using node package manager (npm) This allow you to manage serverless apps directly from your terminal

ii) creating a node is serverless project :-

A directory is created for your project where you will initialize a serverless service (project) This service will house all your lambda function

configurations and cloud resources using the commands serverless create you set up a template for Aws Node is microservices that will eventually deploy to Aws lambda

iii) project structure :-
The project scaffold creates essential files like handler.js cahich contains code for lambda function and serverless YML.

iv) create a REST API Resource
In the serverless yml file you define function that handles part request of HTTP

v) Deploy the service
With the s|s deploy commands serverless framework packages your applications, uploads necessary resources Aws and set up the infrastructure

vi) Testing the API :- once deployed you can text REST API using tools like curl or postman by mating POST request to generated API

vii) staring Data in ~~Database~~ Dynamods :- To store submitted candidate data you integrate Aws DynamoDB as database

viii) Adding more functionality :- Adding functionality like, install candidates get candidates by ID

ix) AWS IAM permissions
You need to ensure that serverless framework is given right permissions to interact with AWS resources like DynamoDB

x) Monitoring and maintenence
After deployment serverless framework provides services information like Deployed endpoints API keys, log streams

Q2) case study for SonarQube
creating your own profile in ~~SonarQube~~ for testing project Quality Use SonarQube to analyze your code Install java ~~file~~ ide and analyze java code.

-> SonarQube is an open source platform used for continous inspection of quality. It detects bug code smells and security vulnerabilities in project accross pgming languages

1) profile creation in sonarQube
Quality profiles in sonarQube are essentia configuration that define rules applied during code analysis. Each project has a quality profile for every supported lang with default being Sonar way profile comes buil

52

in for all languages custom profile can be created by copying or extending existing ones copying creats you can activate or deactivate rules priortize certain rules and configure parameters to profile to specific projects.

2) using sonarQube to analyse github code sonarQube is a cloud based counter part of sonarQube that triggers directly with github Bit Bucket, and github repositories To get started with sonar cloud via github signup product page and connect your github organization or personal account once connect sonar ~~qube~~cloud mirroos your git set up with each project corresponding to the github repos:- After setting up organization where each github repos a sonar cloud projeit. Define new cloud to focus on recent changes and choose between automatic analysis or CI based analysis Automatic analysis happens dicectly in sonarcloud while CI based analysis integrates with your build process once the analysis results can be viewed in both sonar cloud and github including security import issue.

3) sonarlint in Java IDE:-

sonarlint is an IDE that performs on the fly code analysis as your write code It helps develops in the developing envioonment sveh as intellic idea or Eclipse. To set it up install the sonarlint plugin. configure the connection with sonarqube or sonar cloud and select the project profile to analyse Java code in code quality, promoting clean & maintainable code from begining.

4) Analyzing python projects with sonarqube

Sonarqube supports python test coverage reporting but it requires third party too like coverage part to enable and adjest your build process so that coverage tools runs before sonnar scanner and ensures report file is saved in diff path. for set up you can use TUX and coverage to py to configure and run test. In your ton.in : include configuration for py test and coverage to generate report in .xml format. The build process can also be automated using github Actions which install dependencies run test and involves sonarqube scan. Ensure report in XML format and place where scanner can access it.

5) Analyzing Node.js projects with SonarQube

for node.js project sonarqube can analyze Java script and typescript code. similar to the python setup you can configure sonarqube to analyze node.js project by installing the appropriate plugin and using sonar scanner to scan the projects sonarqube will check the code against Industry standard rules and best practices flagging issues related to security vulnerabilities bys and perfomances optimization.

Q3) At a large organization your centralized operation team may get many repatabile infrastructure reqvest you can vse form to builda self services infrastructure mode that lets product team manage their own infrastruct independently you can create and use Terraform modules that codify the standards for deploying & managing services in your organization allowing teams to efficently deploy services

-> Implementing a self service infrasetructure model vsing Terraform can transform how large organization manage their infrastructure independently organization can enhance efficiency

reduce benefits and ensure compliance with established needs.

The need for self service infrastructure:- In large organization centralized operators teams often face an overhelming number repetitive request This can lead to delay in service delivery and frustration among product teams who need to move quickly. A self service model allows team to provision and manage their infrastructure without relying on the operations team for every request

- Benefits of using Terraform:-

1. Modelarity & reusability
2. Standardization
3. Increased Efficiency
4. Integration with ticketing systems

- Implementation steps

i) Identify Infrastructure components.

- Begin by Identifying which components your infrastructure can be modularized lay VPCs, security groups, lead balanced.

2) Develop Terraform modules

- create reusable modules that define the desired configuration & resources
- Ensure each module includes input variables for customization and outputs for integration with other modules.

3) Establish Groverance and Best practices
- Define guidelines for module usage version and documentation to ensure clarity and maitainability
- Encourage teams to contribute to module development and share impovements.

4) Testing and validation

- Implement a testing removable to validate functionality before development.
- Best practices for module management
- utilize the terraform registry
- levelage existing community modules from the Terrafoom Registry to avoid reinventing solutions and ensure adherence to best practices.