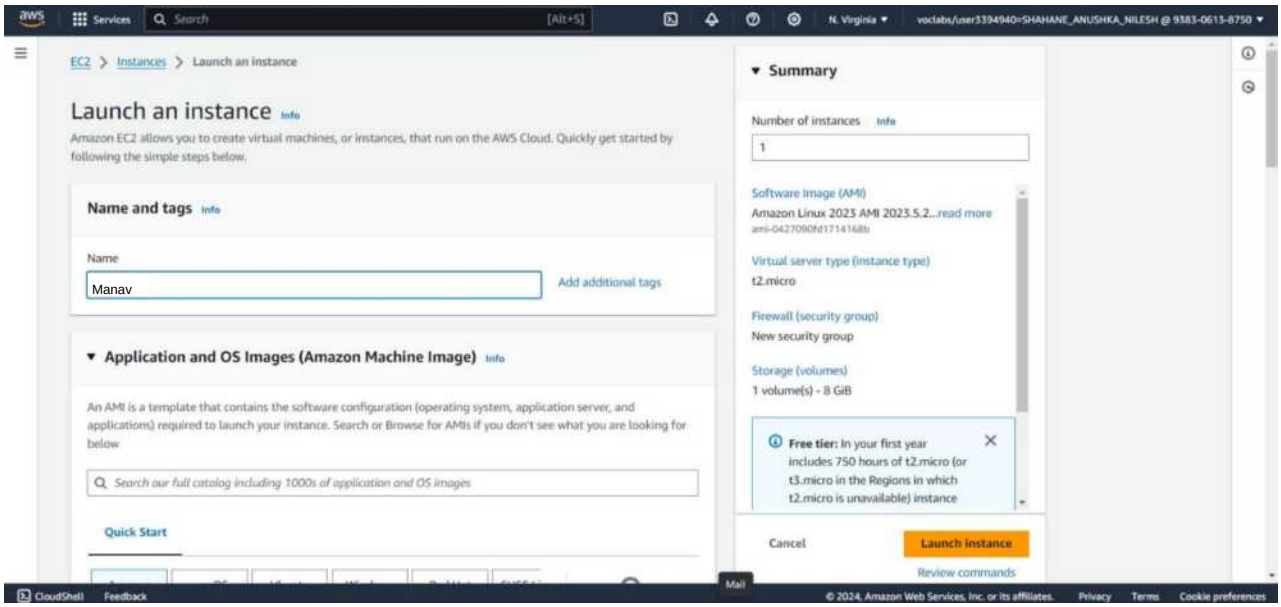
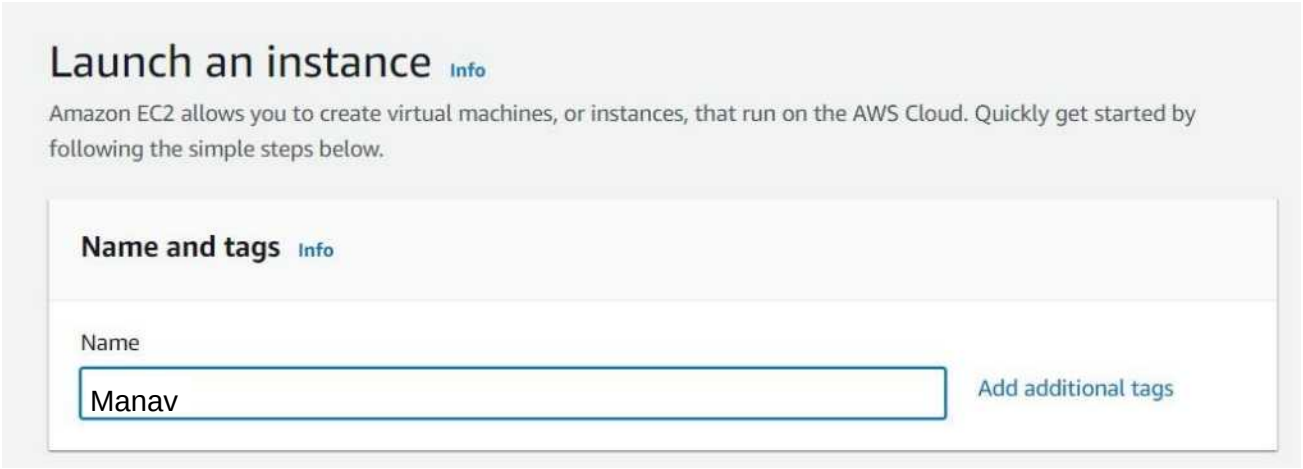
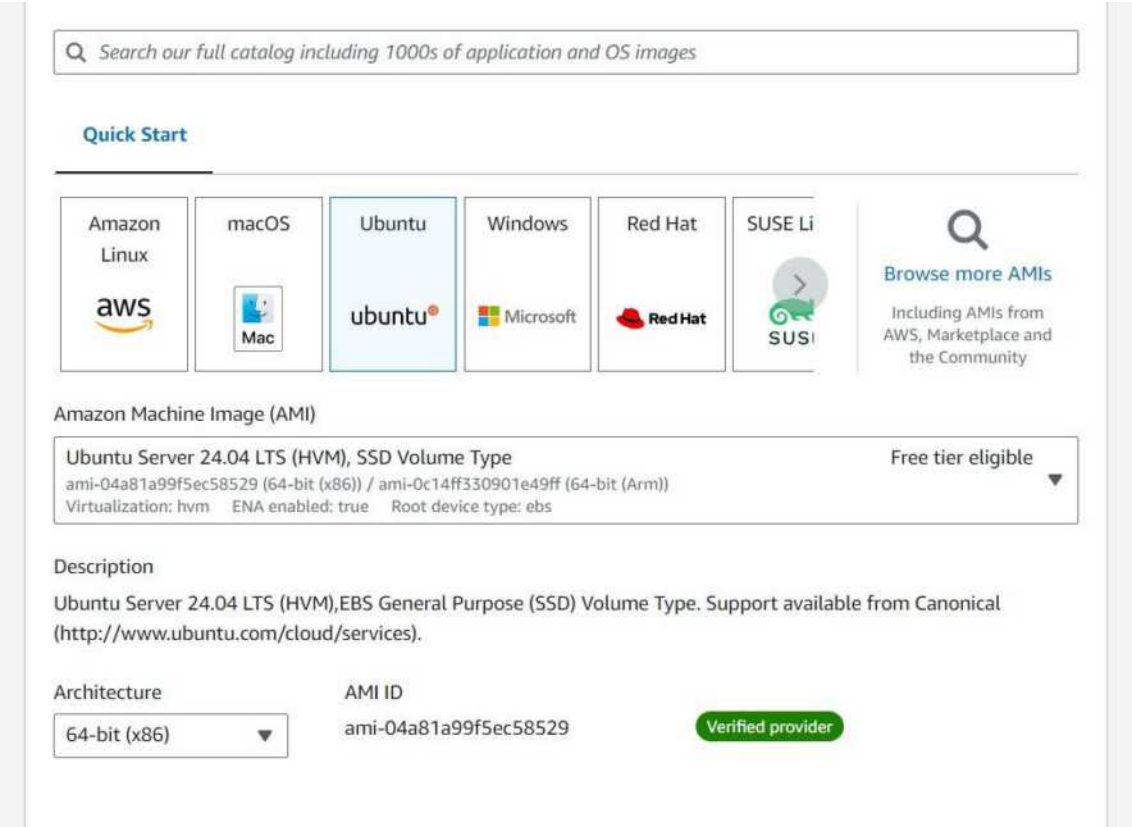
a anav Punjabi D15A 45

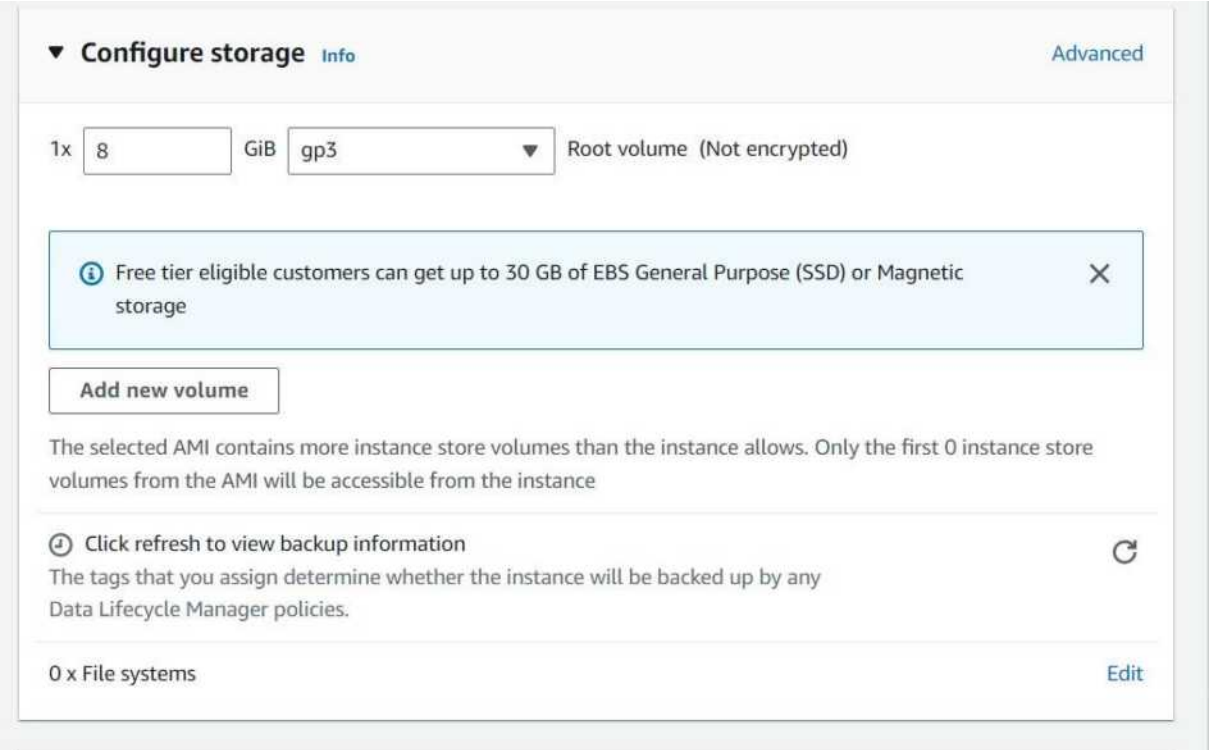
ADVANCE DEVOPS EXPERIMENT 1



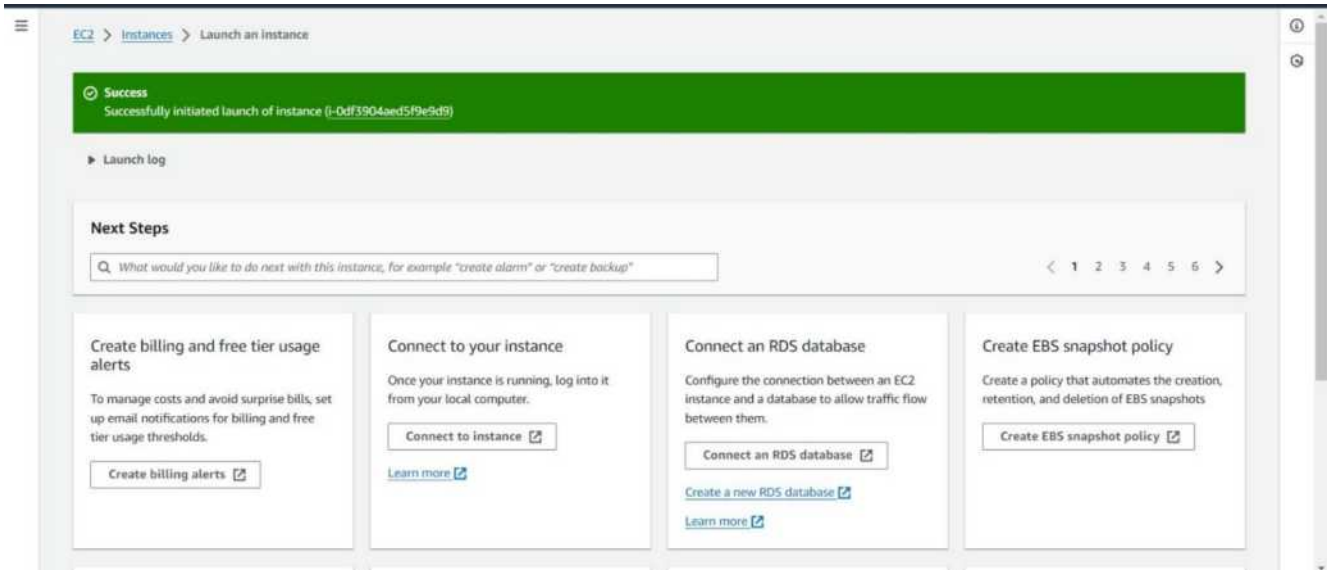


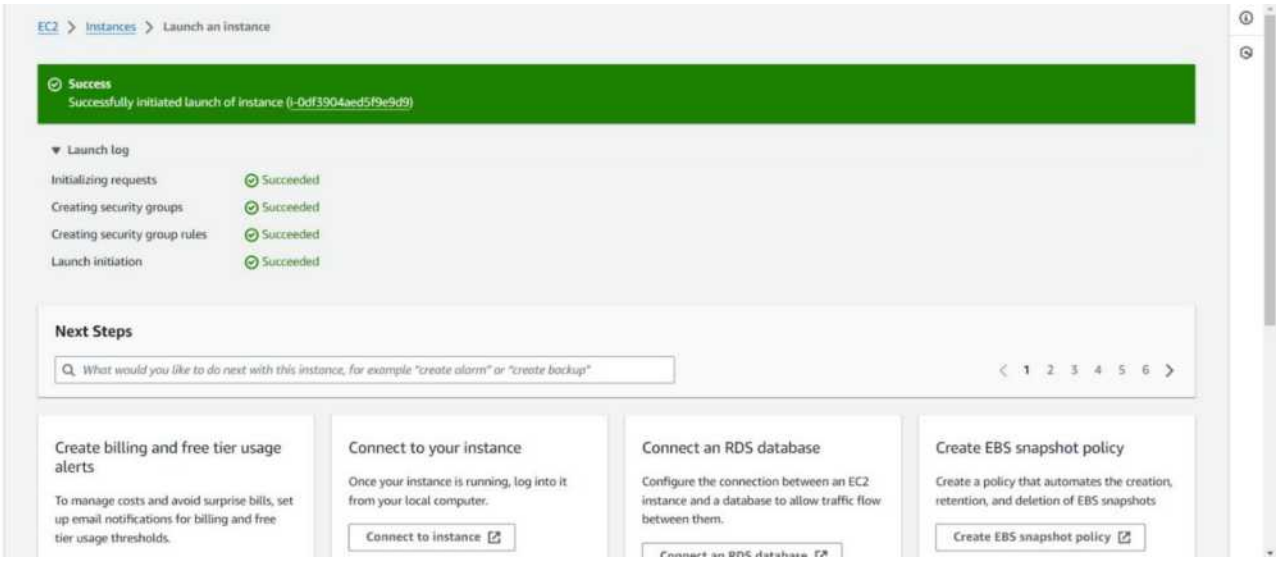
a anav Punjabi D15A 45



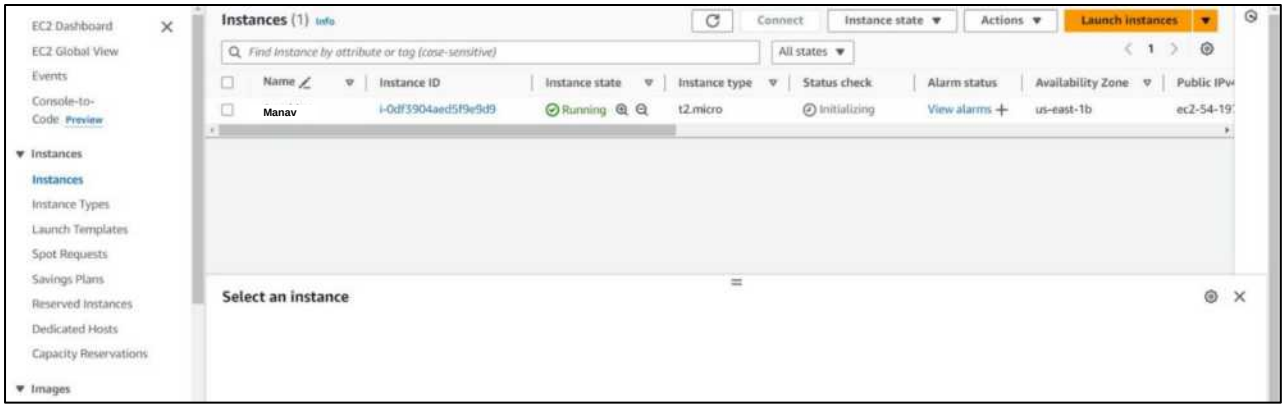


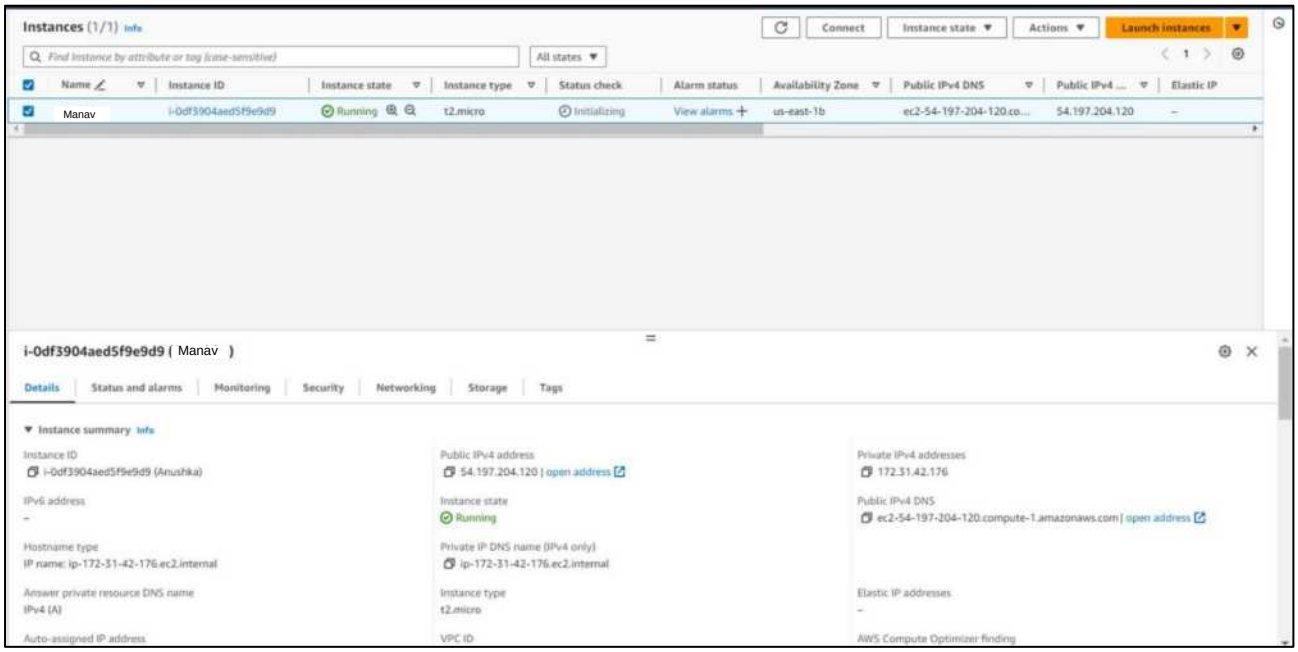
a anav Punjabi D15A 45





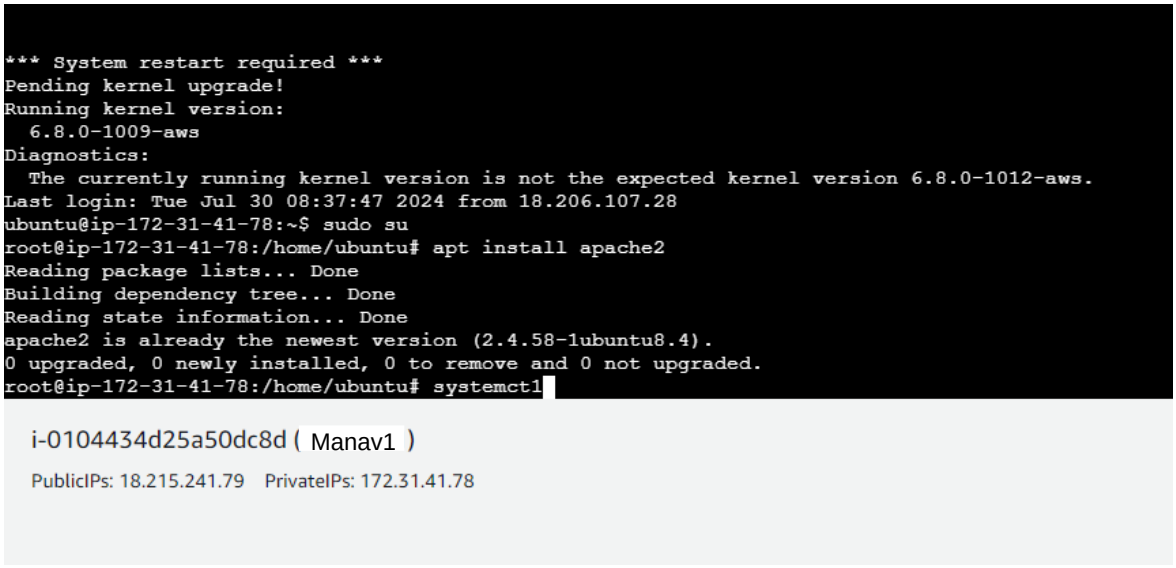
a anav Punjabi D15A 45

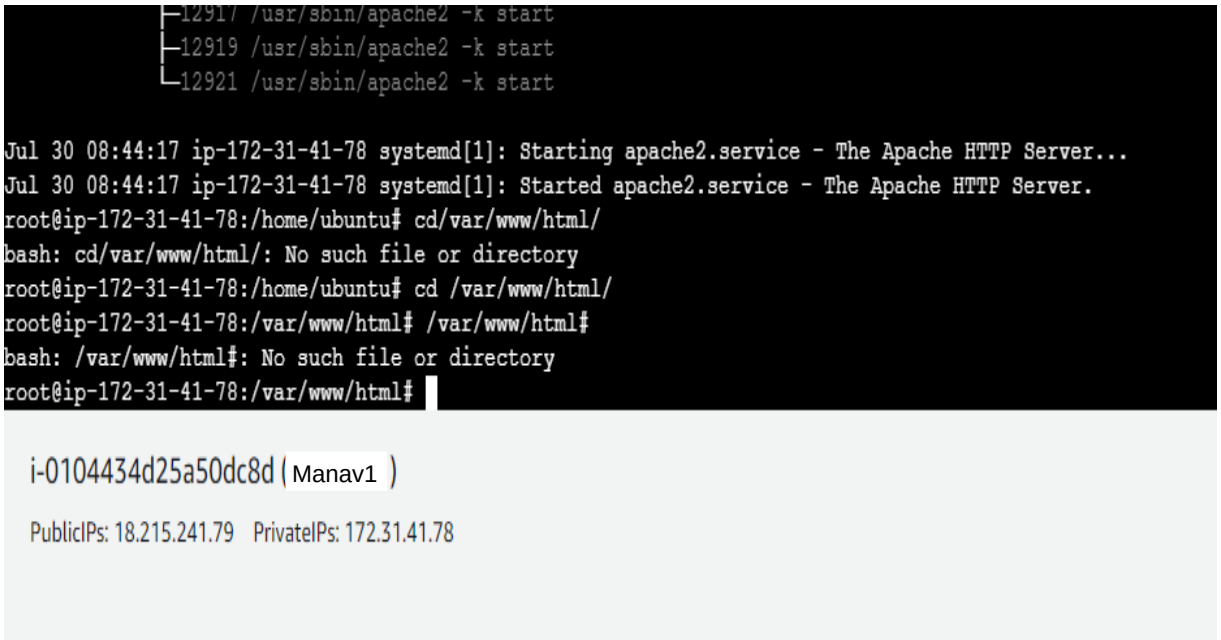





```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-42-176:~$ ls
ubuntu@ip-172-31-42-176:~$ echo "hello"
hello
ubuntu@ip-172-31-42-176:~$ cat > myfile.txt
This is Advance devops lab
^C
ubuntu@ip-172-31-42-176:~$ cat myfile
cat: myfile: No such file or directory
ubuntu@ip-172-31-42-176:~$ cat myfile.txt
This is Advance devops lab
ubuntu@ip-172-31-42-176:~$
```

Hosting a static website using EC2 instance:



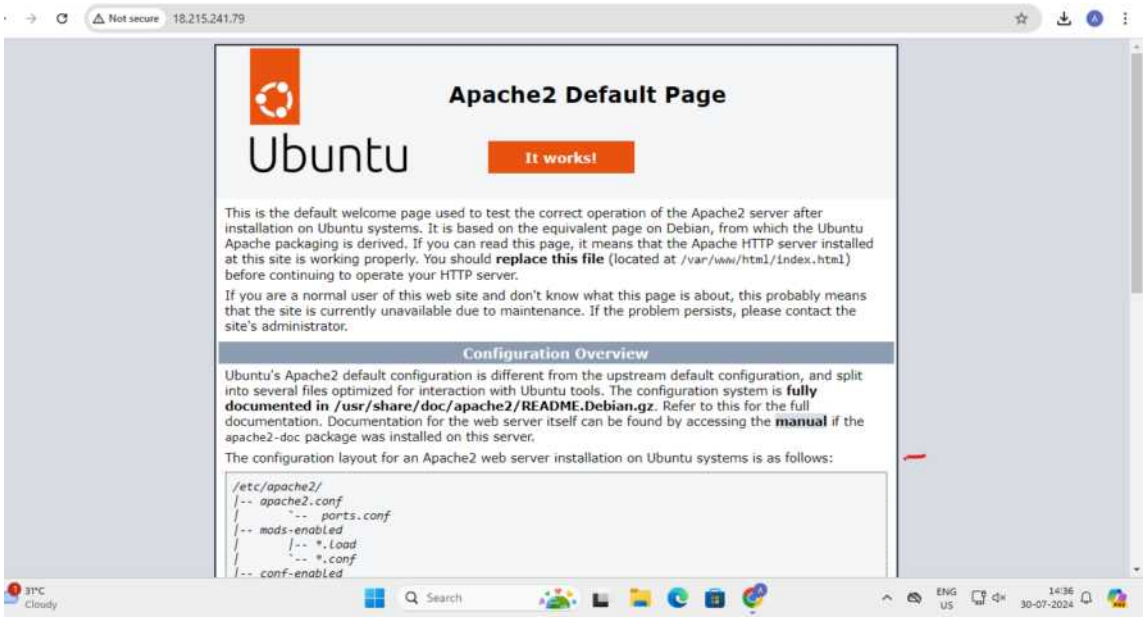


a anav Punjabi D15A 45

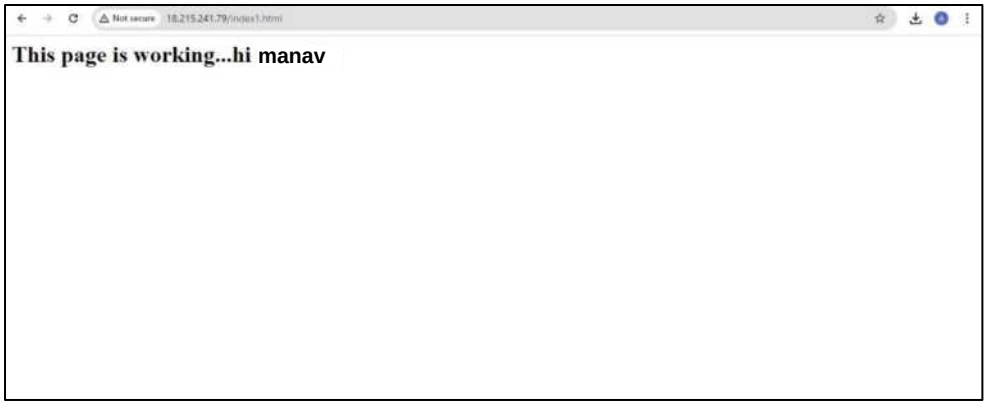
```
  command 'systemctl' from deb systemctl (1.4.4181-1.1)
Try: apt install <deb name>
root@ip-172-31-41-78:/home/ubuntu# systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Tue 2024-07-30 08:44:17 UTC; 12min ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 12917 (apache2)
      Tasks: 55 (limit: 1130)
     Memory: 5.3M (peak: 5.4M)
        CPU: 74ms
     CGroup: /system.slice/apache2.service
             ├─12917 /usr/sbin/apache2 -k start
             ├─12919 /usr/sbin/apache2 -k start
             └─12921 /usr/sbin/apache2 -k start

Jul 30 08:44:17 ip-172-31-41-78 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Jul 30 08:44:17 ip-172-31-41-78 systemd[1]: Started apache2.service - The Apache HTTP Server.
root@ip-172-31-41-78:/home/ubuntu#
```

i-0104434d25a50dc8d (Manav1)

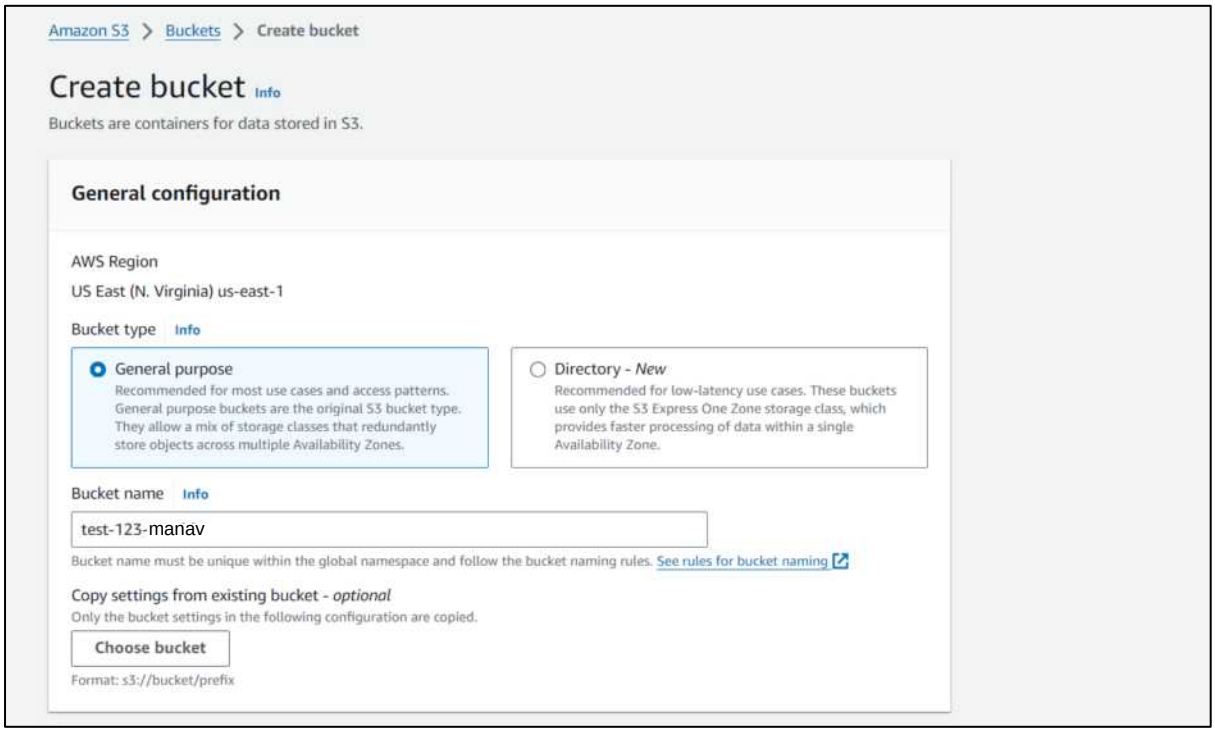
PublicIPs: 18.215.241.79 PrivateIPs: 172.31.41.78



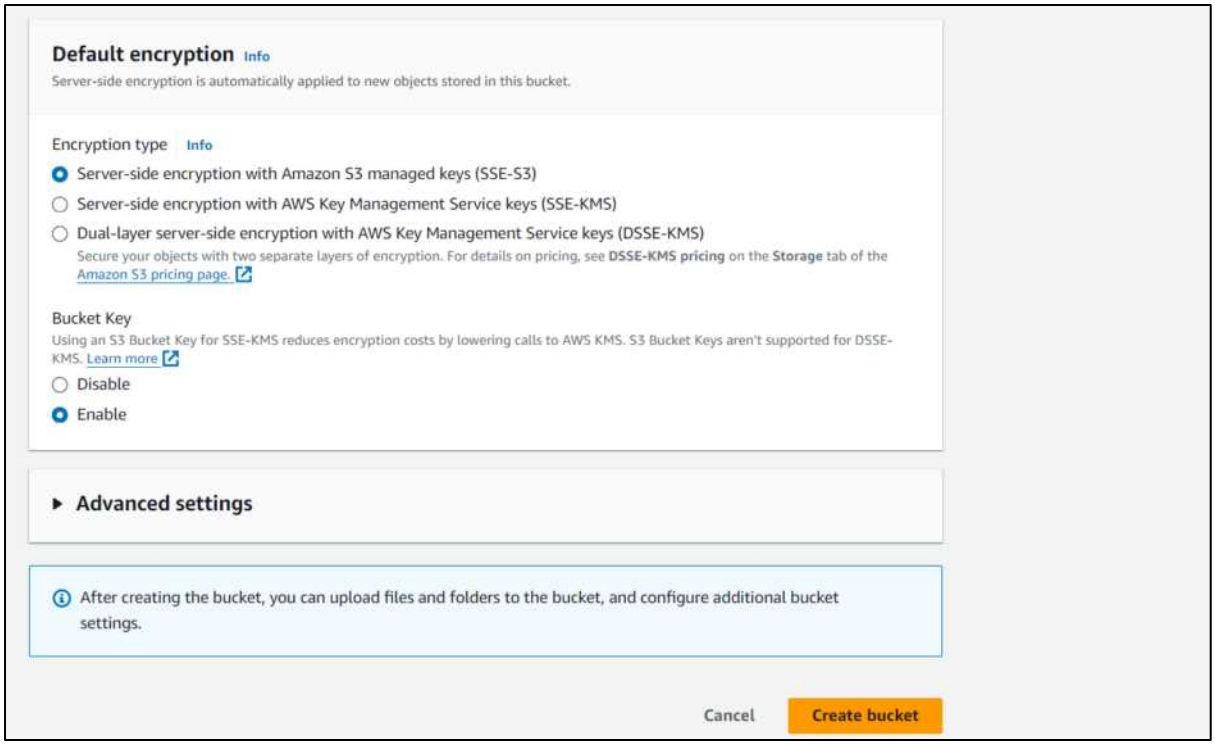
a anav Punjabi D15A 45

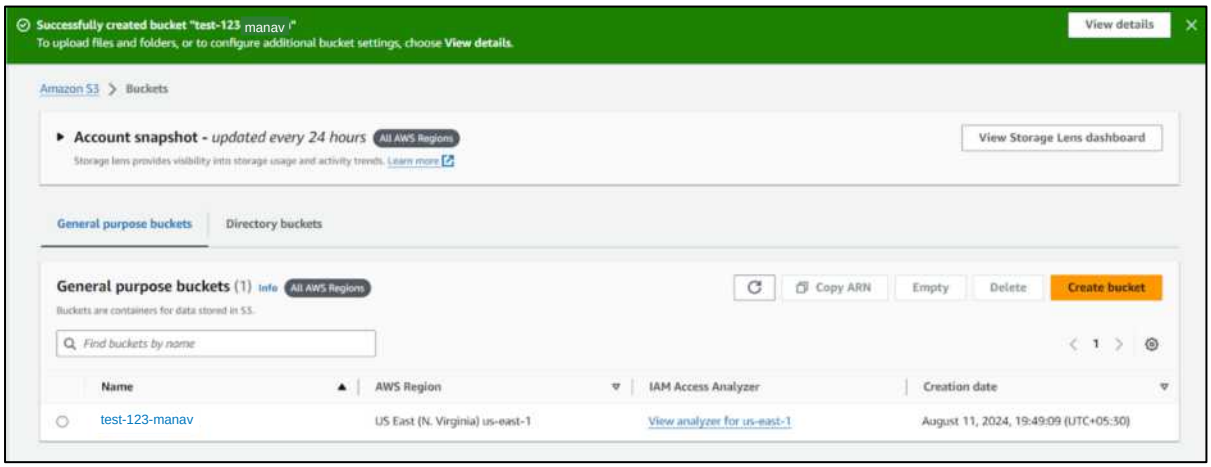


Hosting using S3 bucket :

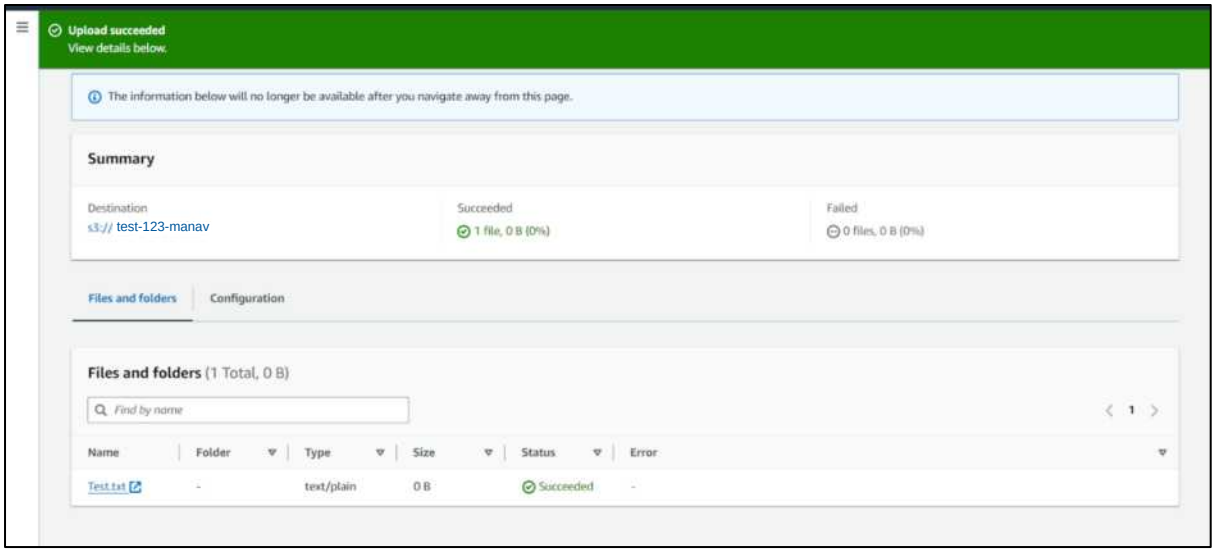


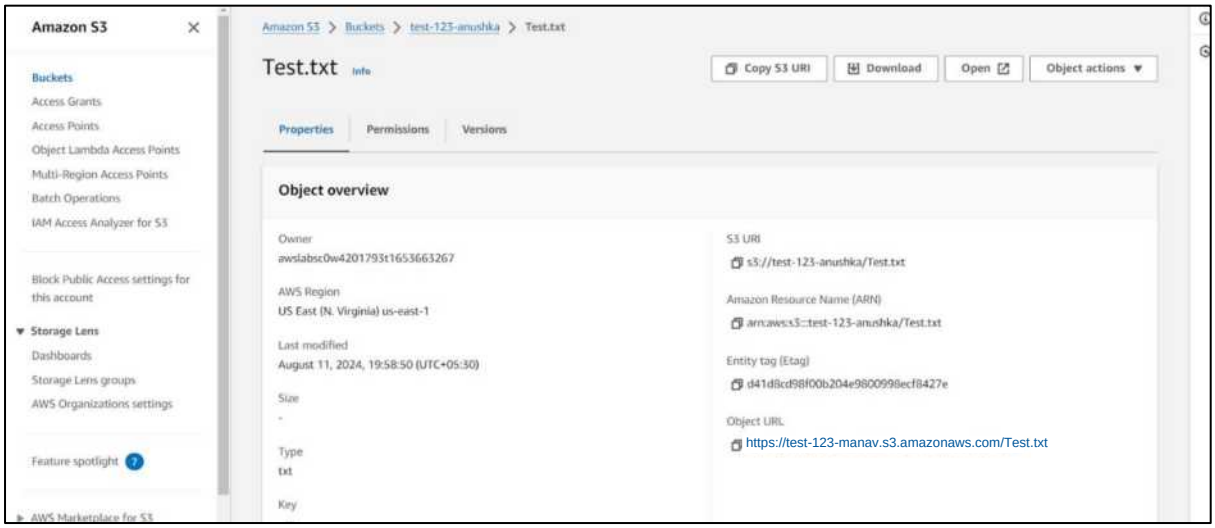
a anav Punjabi D15A 45



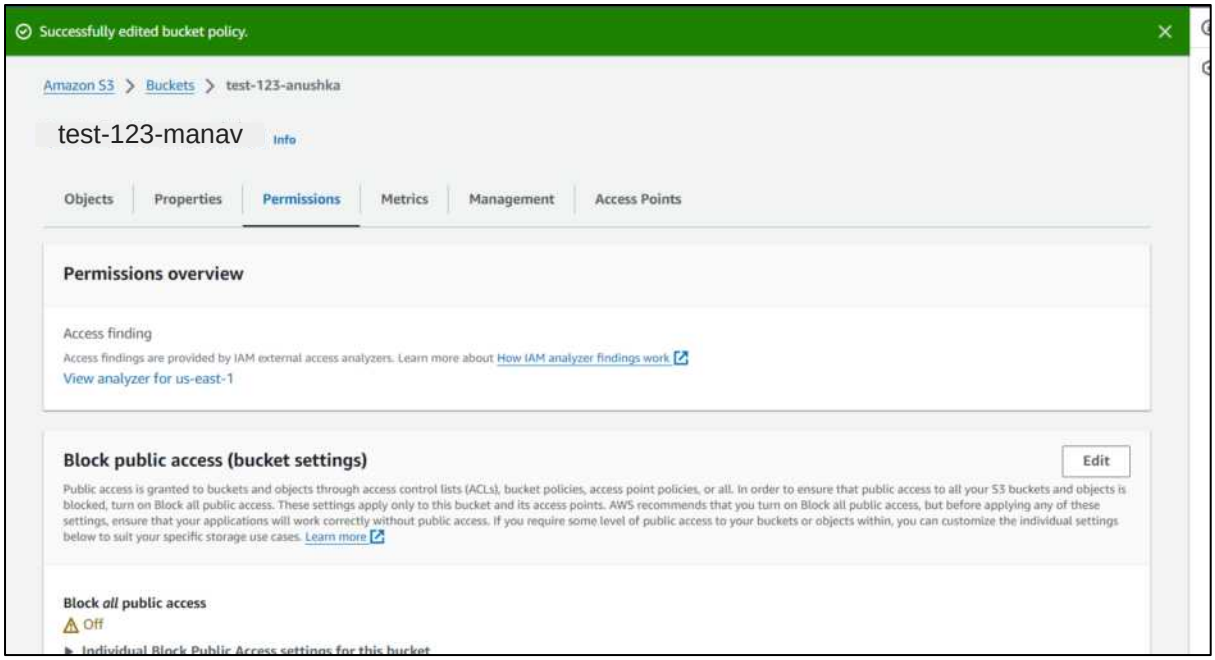


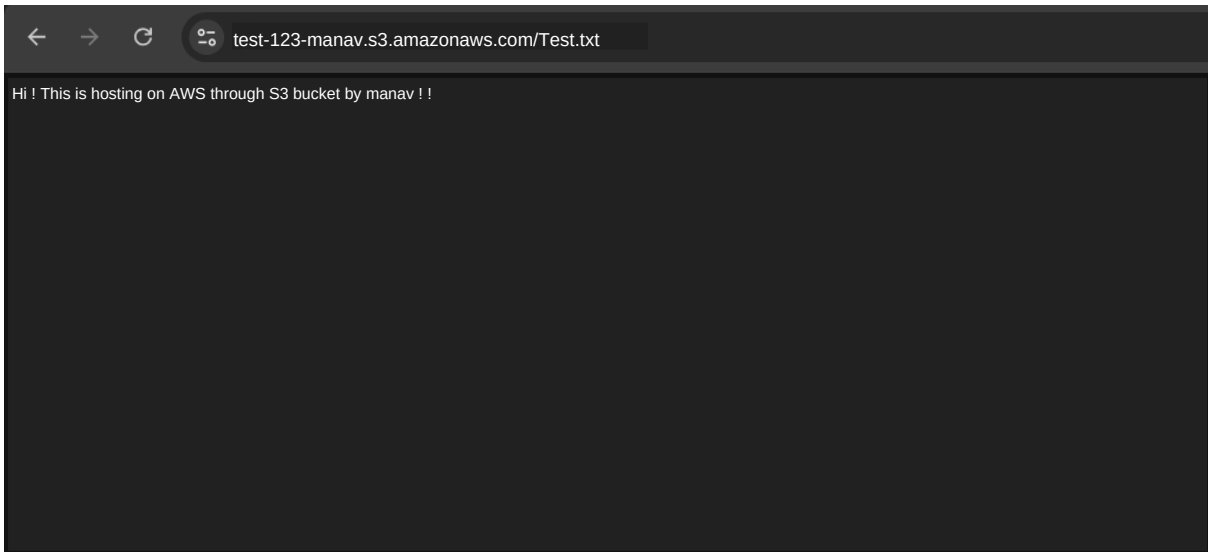
a anav Punjabi D15A 45





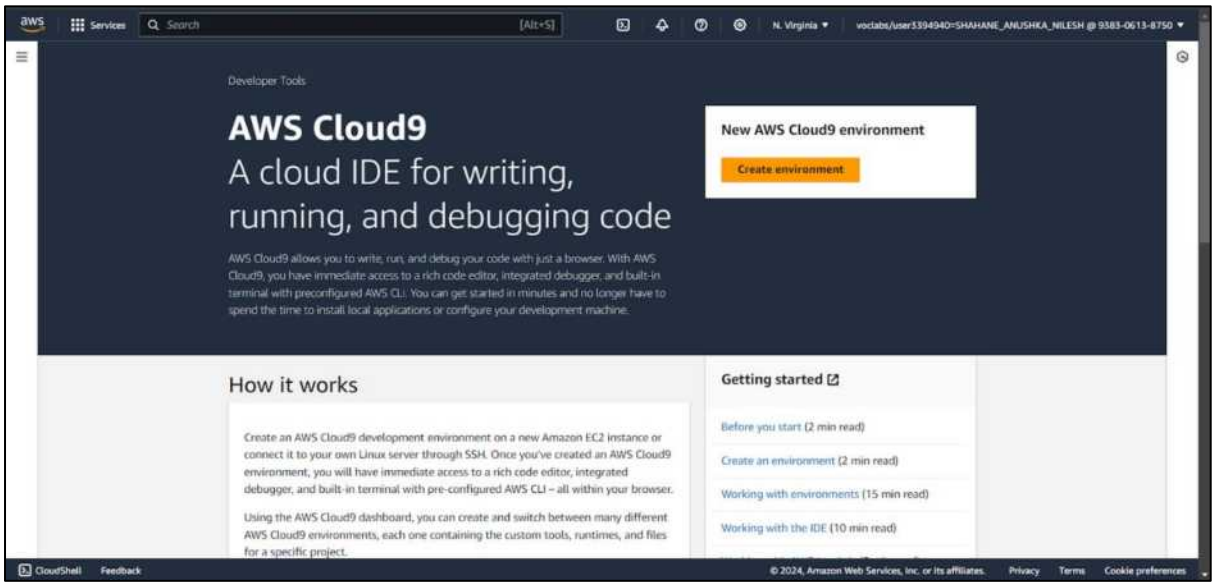
a anav Punjabi D15A 45

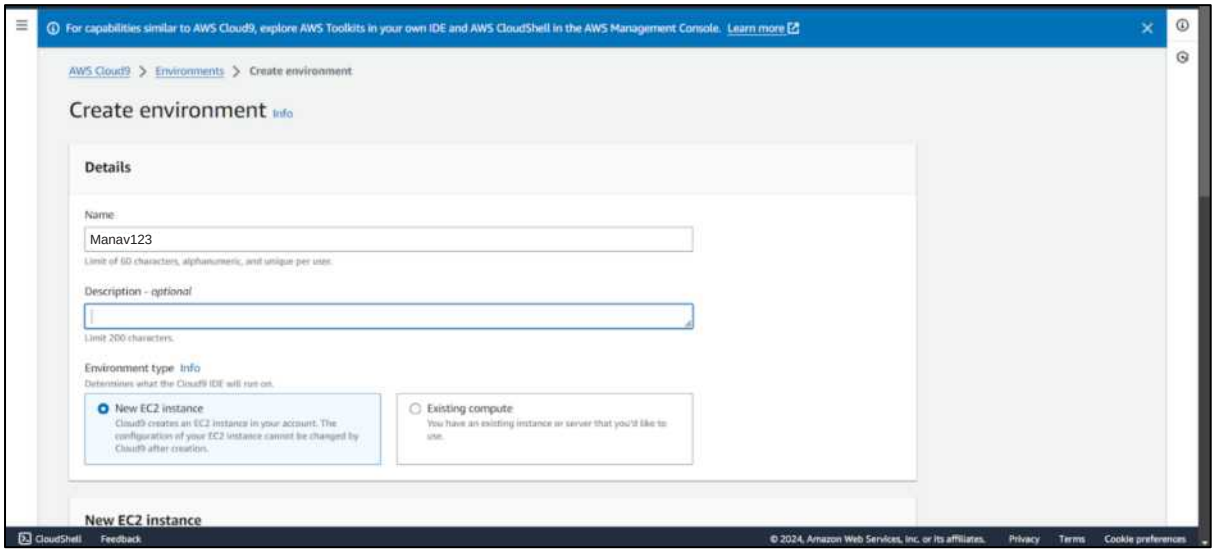




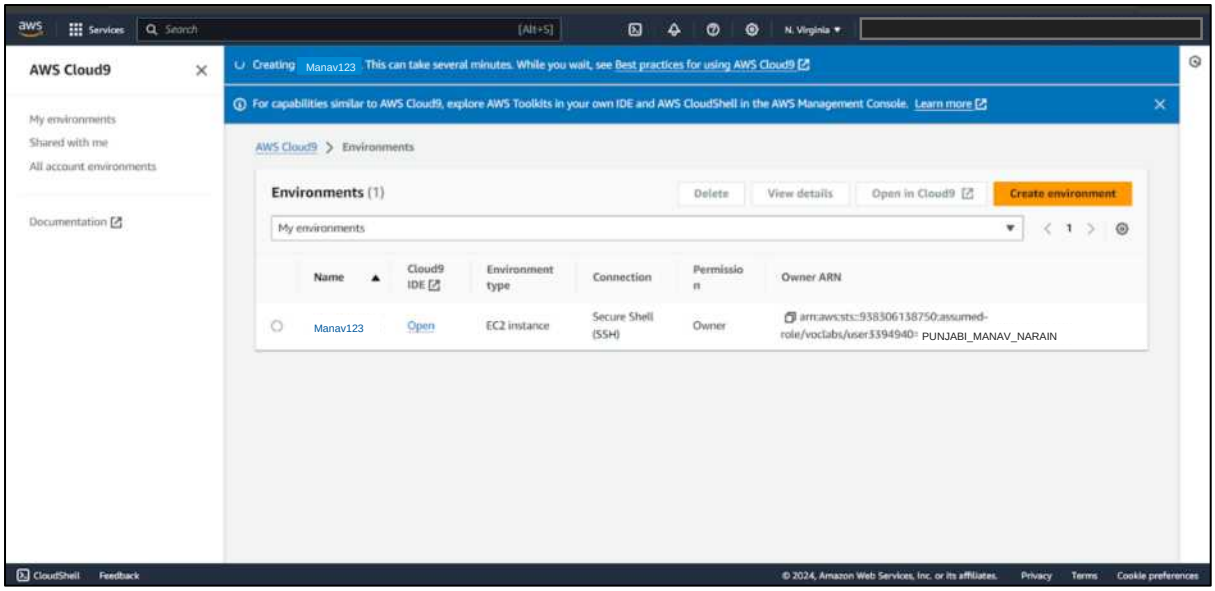
a anav Punjabi D15A 45

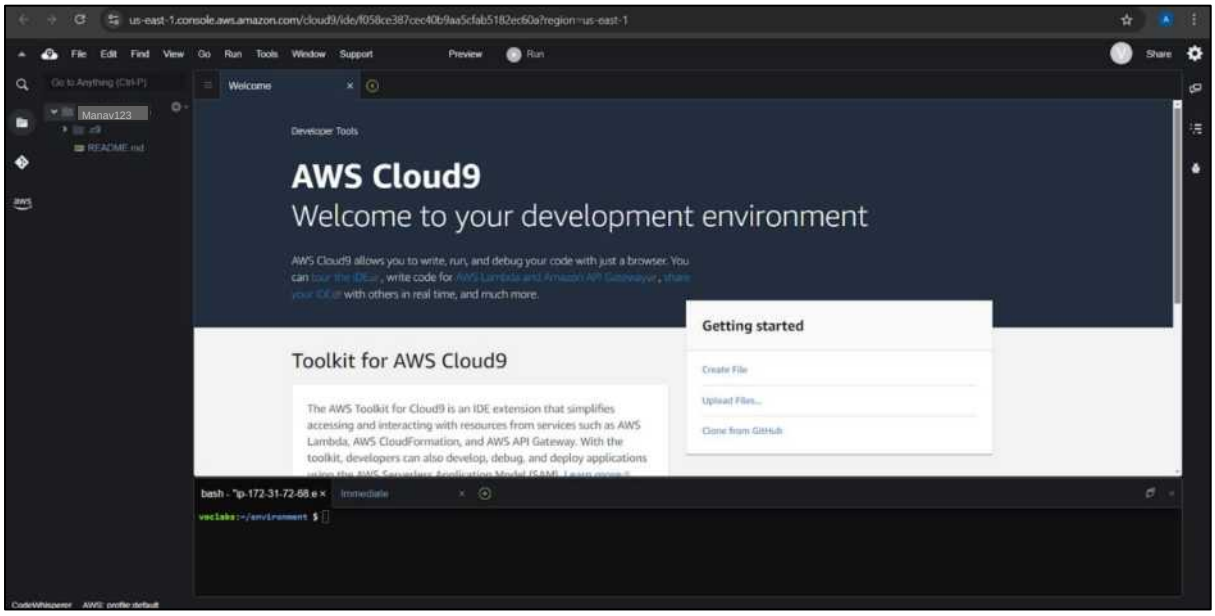
Hosting using Cloud 9 :



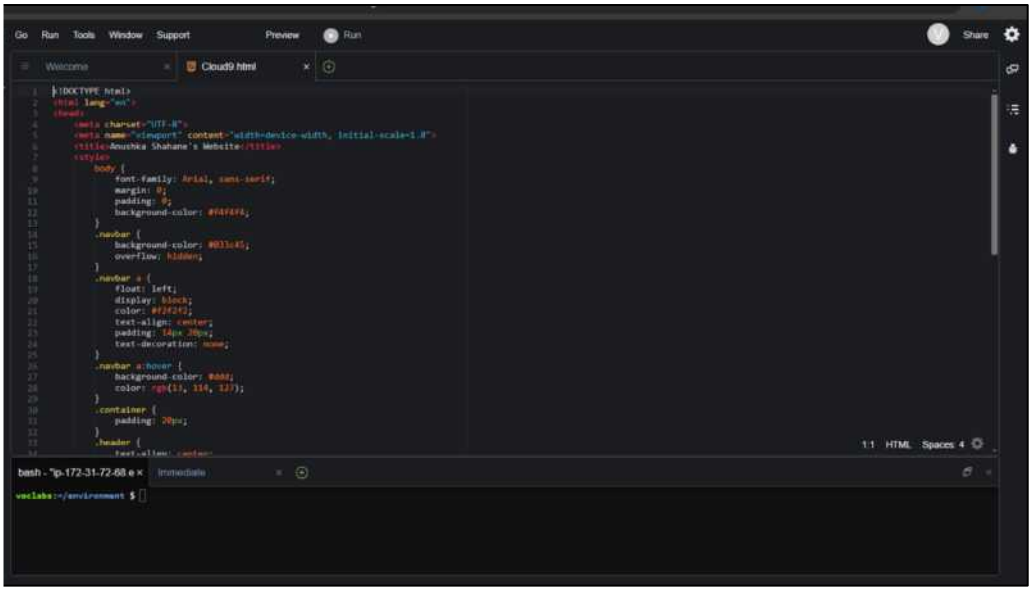


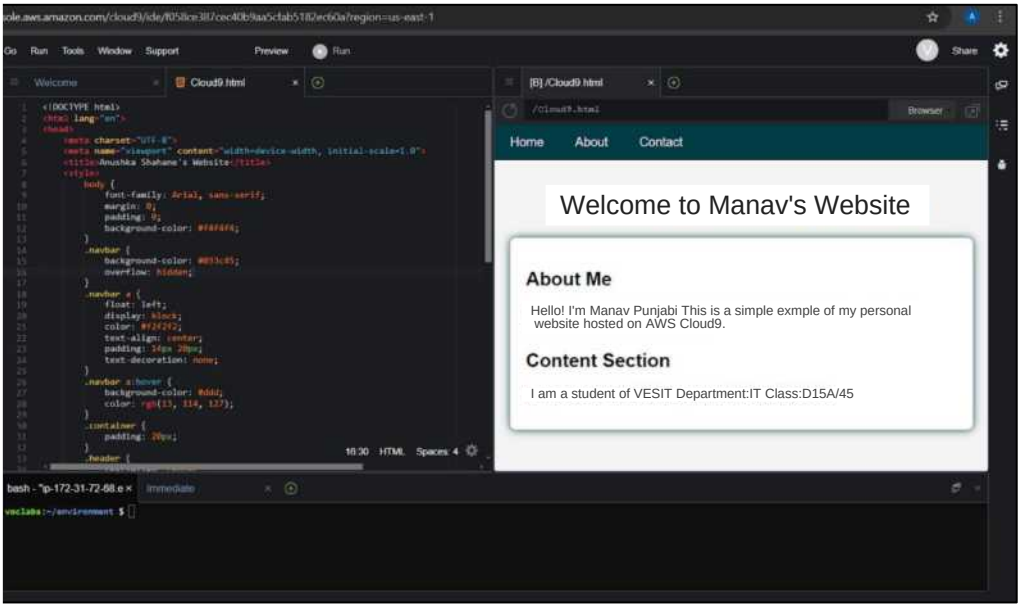
a anav Punjabi D15A 45





a anav Punjabi D15A 45





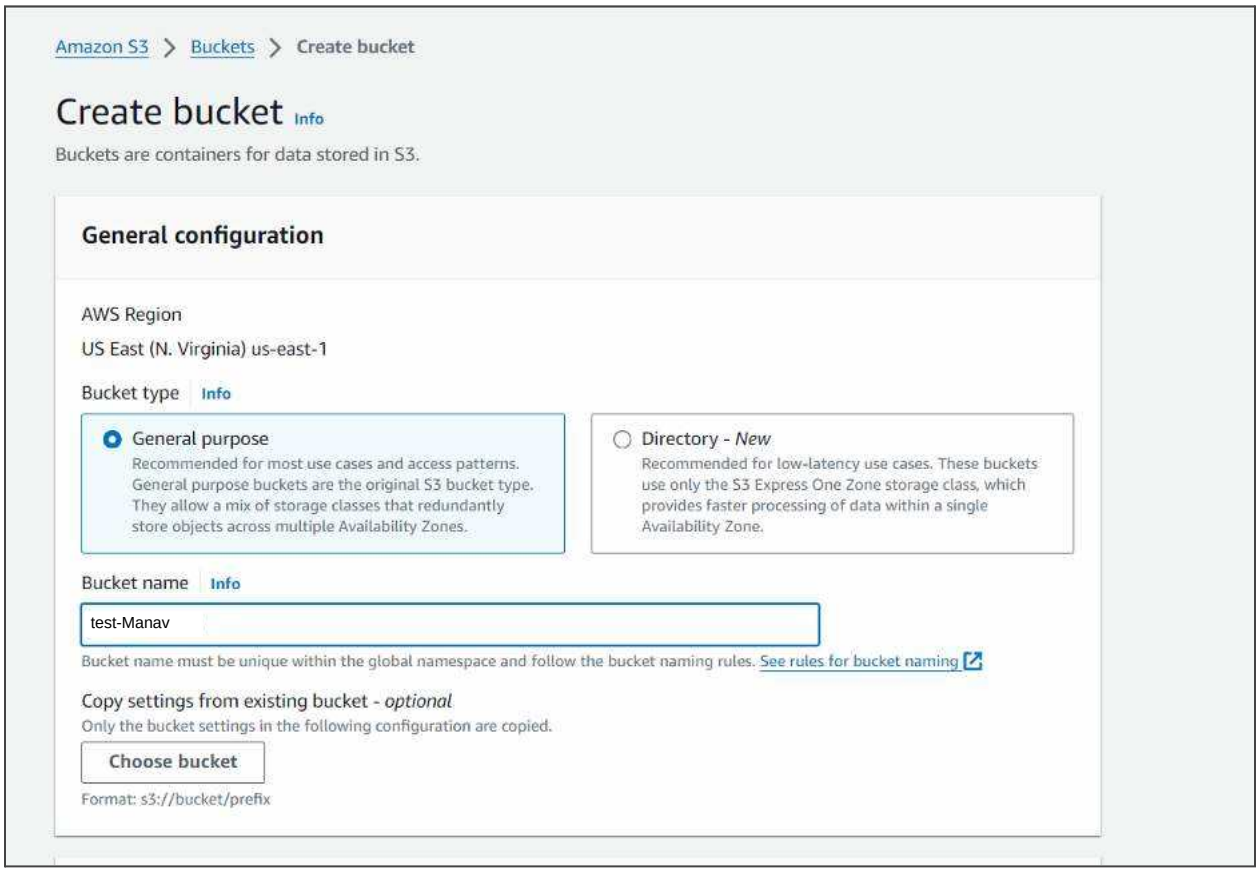
Experiment 2

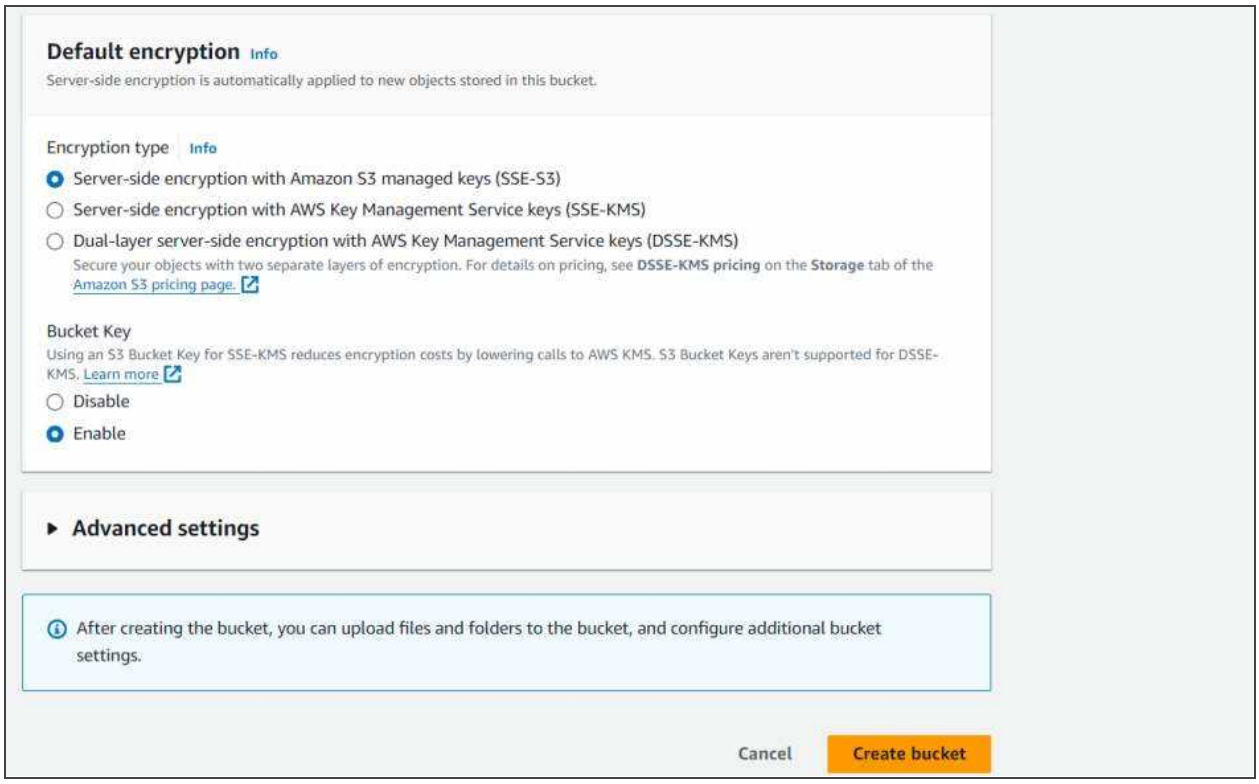
Aim: To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.

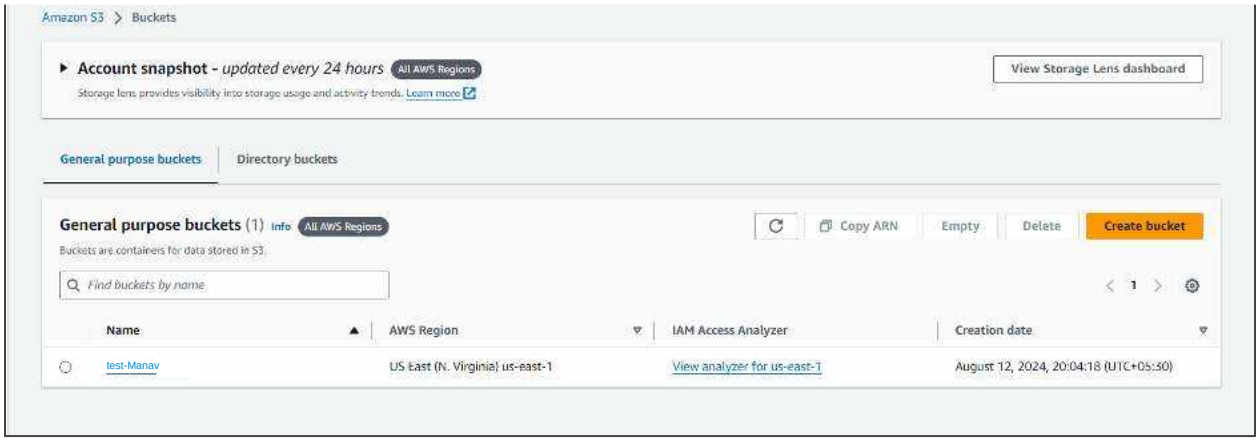
Contents:

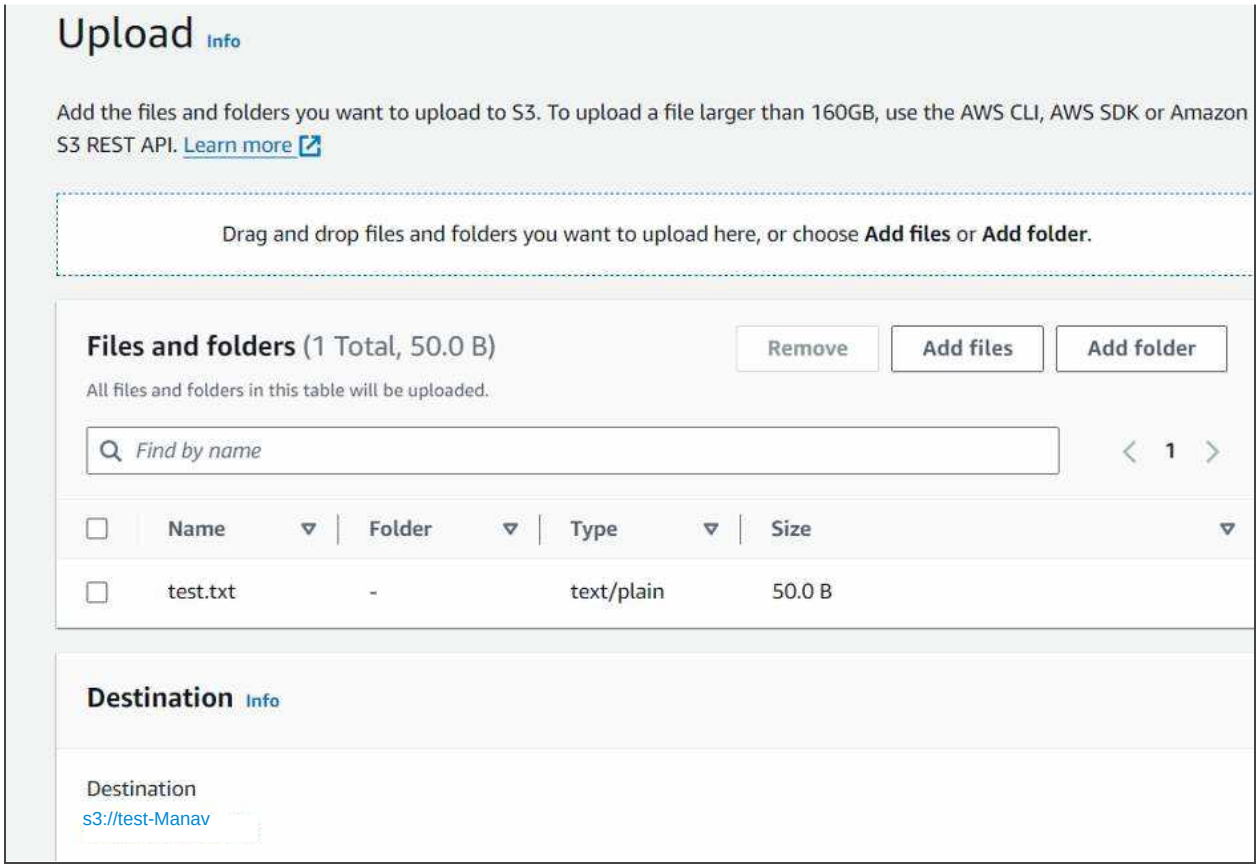
1. s3 bucket
2. ec2 instance
3. elastic beanstalk

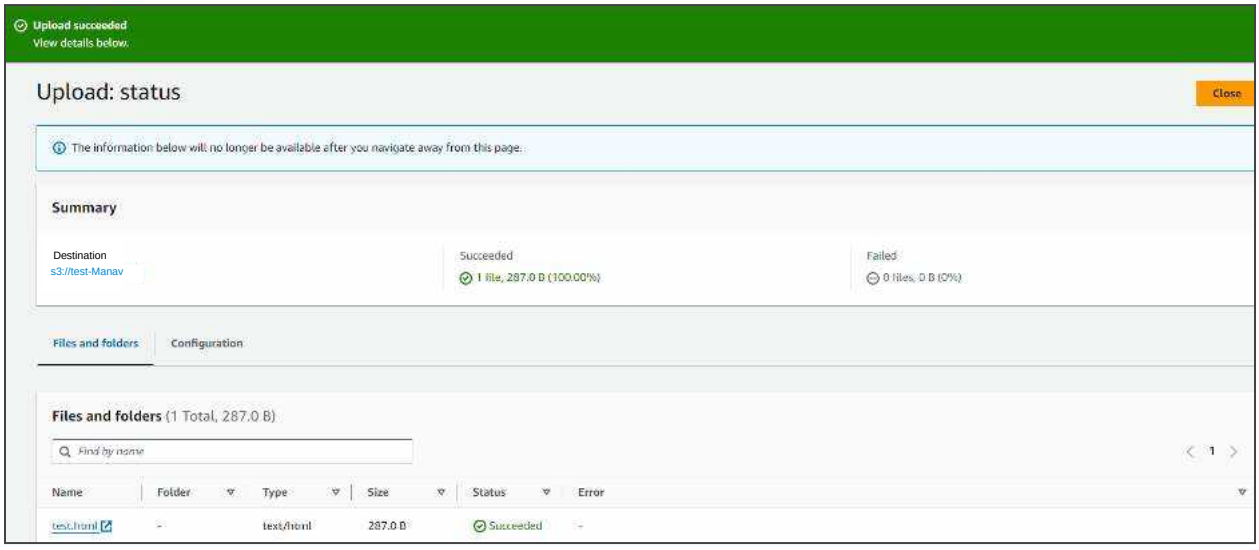
- s3 bucket

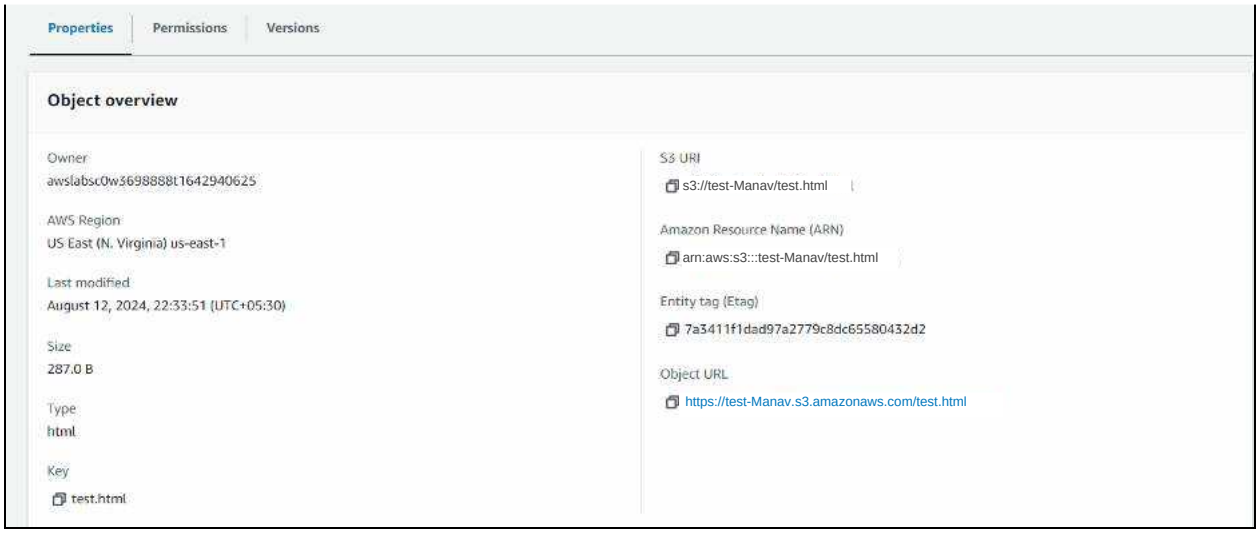


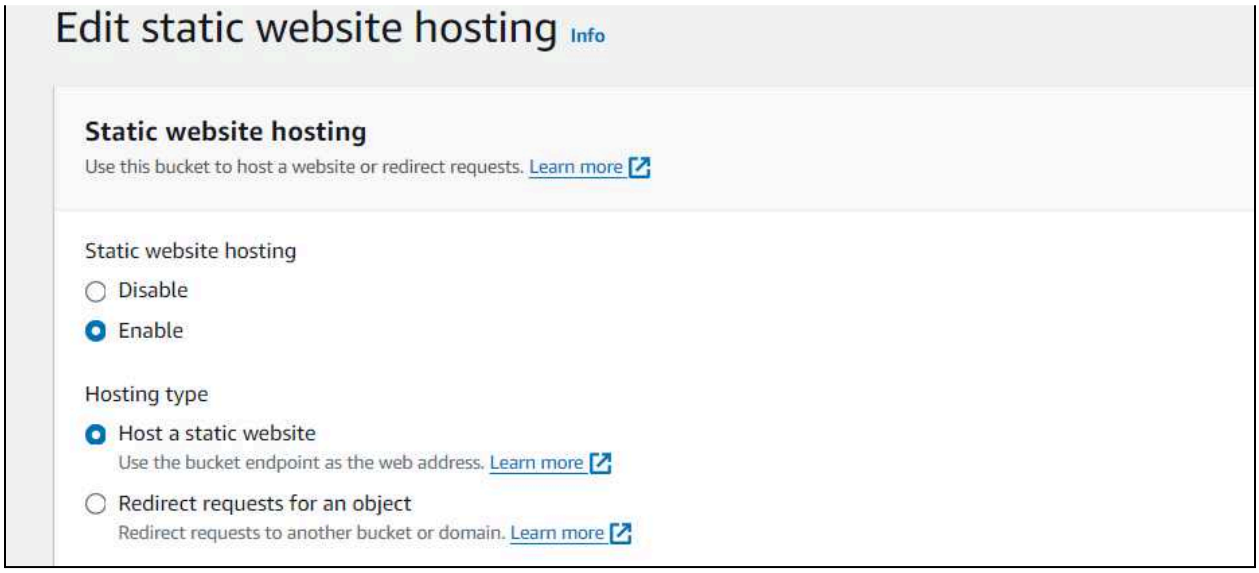

Default encryption Info
Server-side encryption is automatically applied to new objects stored in this bucket.
Encryption type Info
Server-side encryption with Amazon S3 managed keys (SSE-S3)
Server-side encryption with AWS Key Management Service keys (SSE-KMS)
Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)
Secure your objects with two separate layers of encryption. For details on pricing, see DSSE-KMS pricing on the Storage tab of the Amazon S3 pricing page.
Bucket Key
Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. Learn more
Disable
Enable
Advanced settings
After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.
Cancel
Create bucket

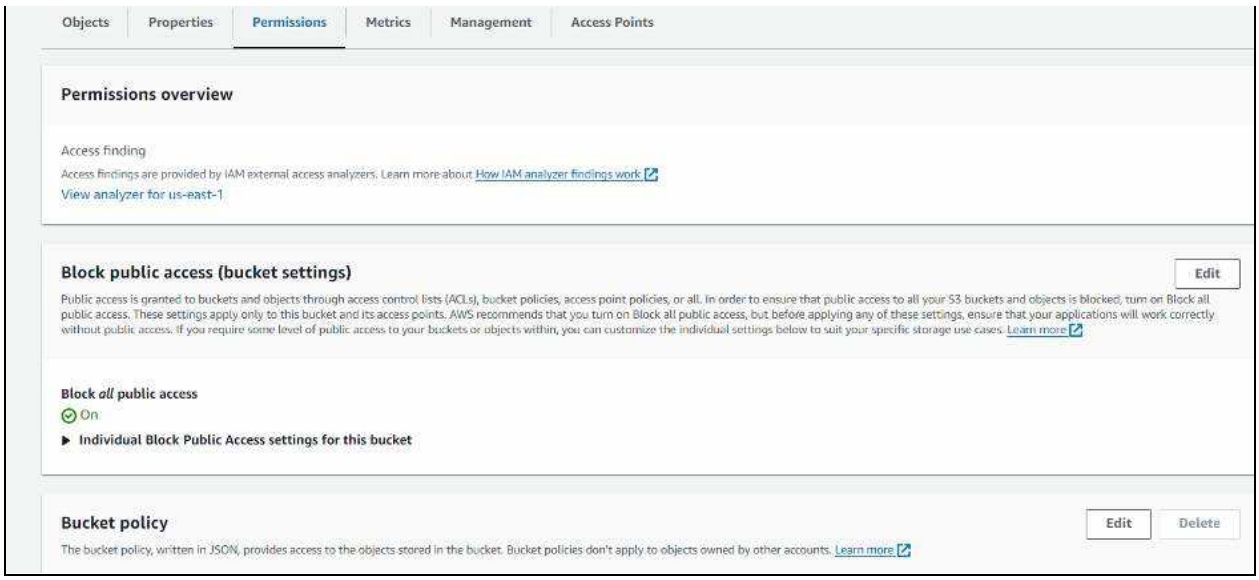

Amazon S3 > Buckets
Account snapshot - updated every 24 hours All AWS Regions
Storage lens provides visibility into storage usage and activity trends. Learn more
View Storage Lens dashboard
General purpose buckets
Directory buckets
General purpose buckets (1) Info All AWS Regions
Buckets are containers for data stored in S3.
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
Name
AWS Region
IAM Access Analyzer
Creation date
test-Manav
US East (N. Virginia) us-east-1
View analyzer for us-east-1
August 12, 2024, 20:04:18 (UTC+05:30)

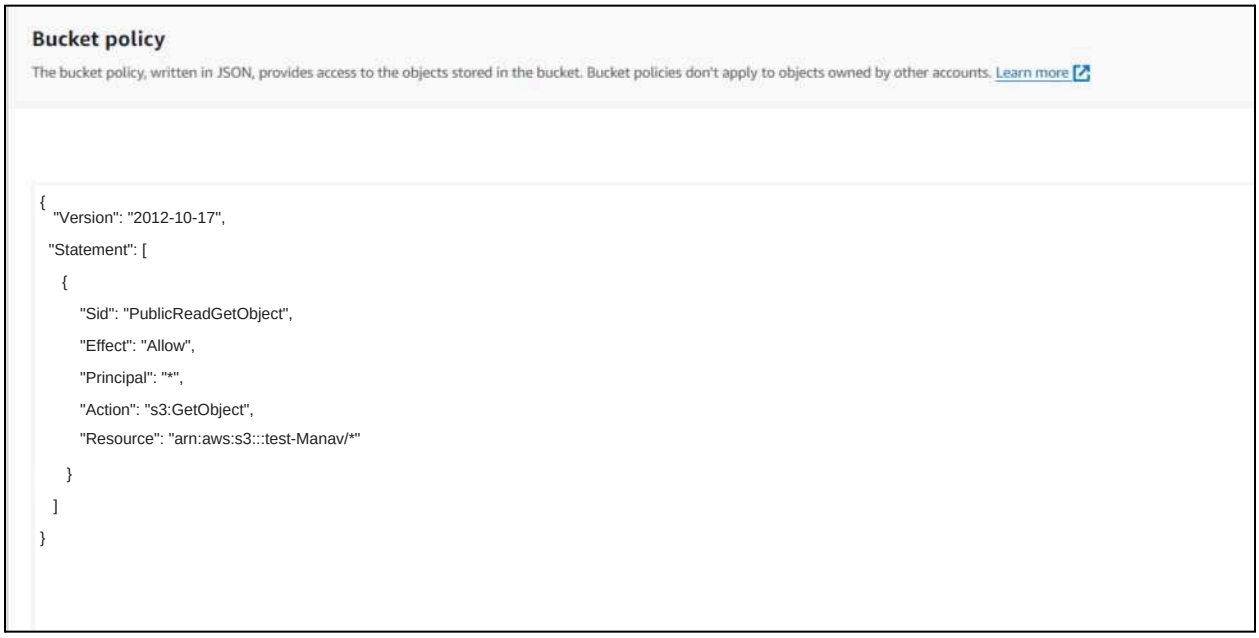

Upload Info
Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more
Drag and drop files and folders you want to upload here, or choose Add files or Add folder.
Files and folders (1 Total, 50.0 B)
Remove
Add files
Add folder
All files and folders in this table will be uploaded.
Find by name
1
Name
Folder
Type
Size
test.txt
-
text/plain
50.0 B
Destination Info
Destination
s3://test-Manav

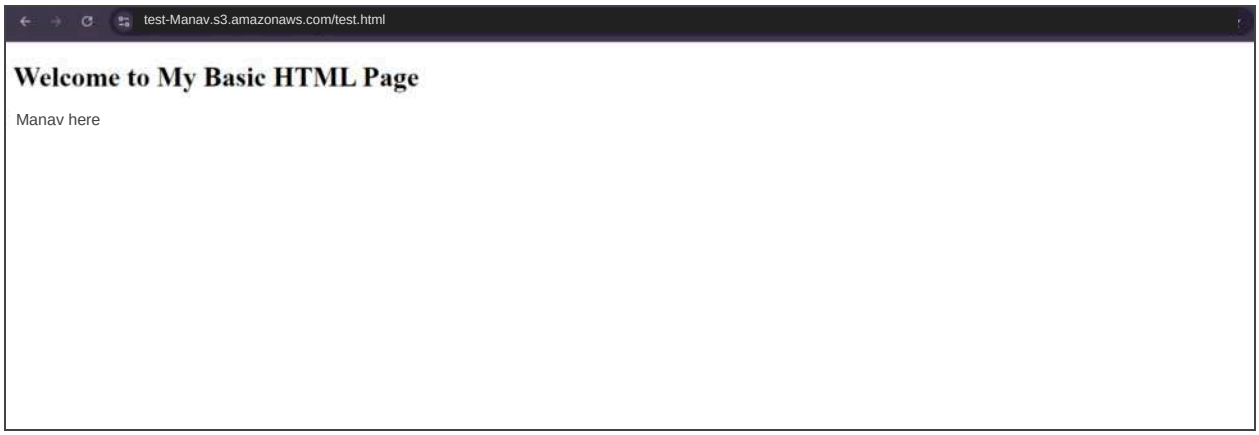

Upload succeeded
View details below.
Upload: status
Close
The information below will no longer be available after you navigate away from this page.
Summary
Destination
s3://test-Manav
Succeeded
1 file, 287.0 B (100.00%)
Failed
0 files, 0 B (0%)
Files and folders
Configuration
Files and folders (1 Total, 287.0 B)
Find by name
1
Name
Folder
Type
Size
Status
Error
test.html
-
text/html
287.0 B
Succeeded
-


Properties
Permissions
Versions
Object overview
Owner
awslabsc0w3698888t1642940625
AWS Region
US East (N. Virginia) us-east-1
Last modified
August 12, 2024, 22:33:51 (UTC+05:30)
Size
287.0 B
Type
html
Key
test.html
S3 URI
s3://test-Manav/test.html
Amazon Resource Name (ARN)
arn:aws:s3:::test-Manav/test.html
Entity tag (Etag)
7a3411f1dad97a2779c8dc65580432d2
Object URL
https://test-Manav.s3.amazonaws.com/test.html

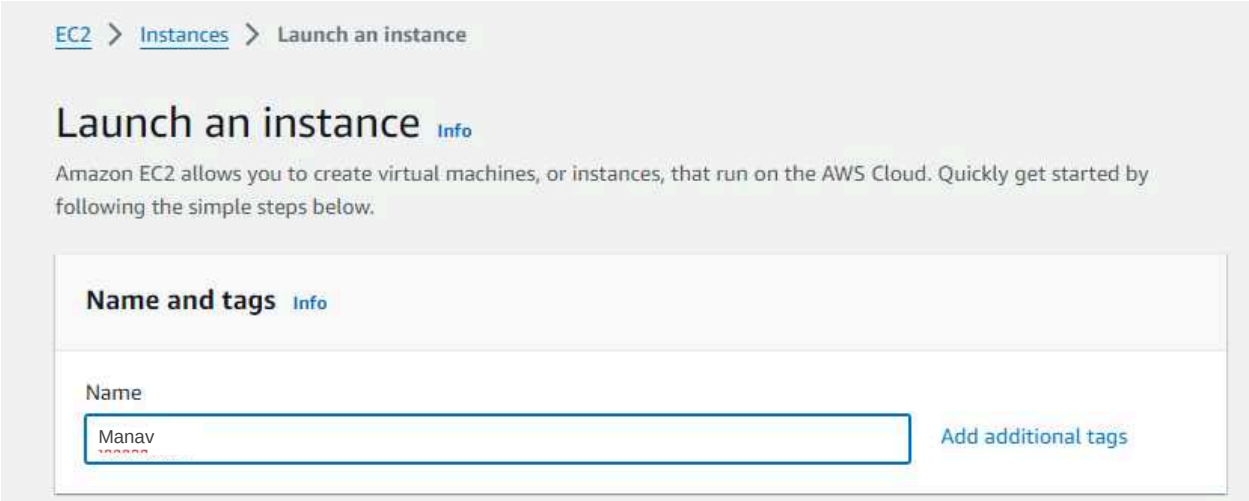

Edit static website hosting Info
Static website hosting
Use this bucket to host a website or redirect requests. Learn more
Static website hosting
Disable
Enable
Hosting type
Host a static website
Use the bucket endpoint as the web address. Learn more
Redirect requests for an object
Redirect requests to another bucket or domain. Learn more

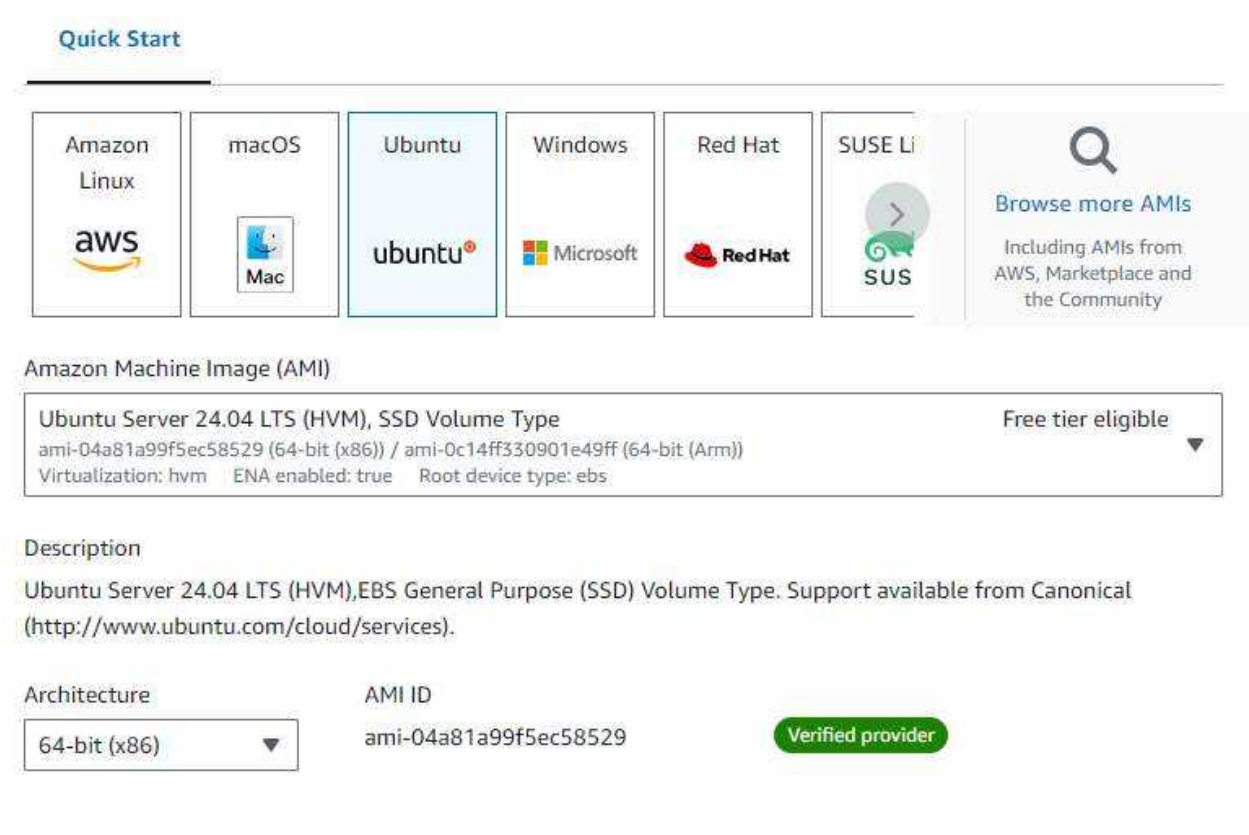

Objects
Properties
Permissions
Metrics
Management
Access Points
Permissions overview
Access finding
Access findings are provided by IAM external access analyzers. Learn more about How IAM analyzer findings work
View analyzer for us-east-1
Block public access (bucket settings)
Edit
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more
Block all public access
On
Individual Block Public Access settings for this bucket
Bucket policy
Edit
Delete
The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more


Bucket policy
The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more
{
"Version": "2012-10-17",
"Statement": [
{
"Sid": "PublicReadGetObject",
"Effect": "Allow",
"Principal": "*",
"Action": "s3:GetObject",
"Resource": "arn:aws:s3:::test-Manav/*"
}
]
}


test-Manav.s3.amazonaws.com/test.html
Welcome to My Basic HTML Page
Manav here

- Launching an EC2 instance





Instance type Info | Get advice

Instance type

t2.micro — Free tier eligible

Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0162 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour
On-Demand RHEL base pricing: 0.026 USD per Hour
On-Demand Linux base pricing: 0.0116 USD per Hour

All generations

Compare instance types

Additional costs apply for AMIs with pre-installed software

Configure storage Info

Advanced

1x 8 GiB gp3 Root volume (Not encrypted)

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage

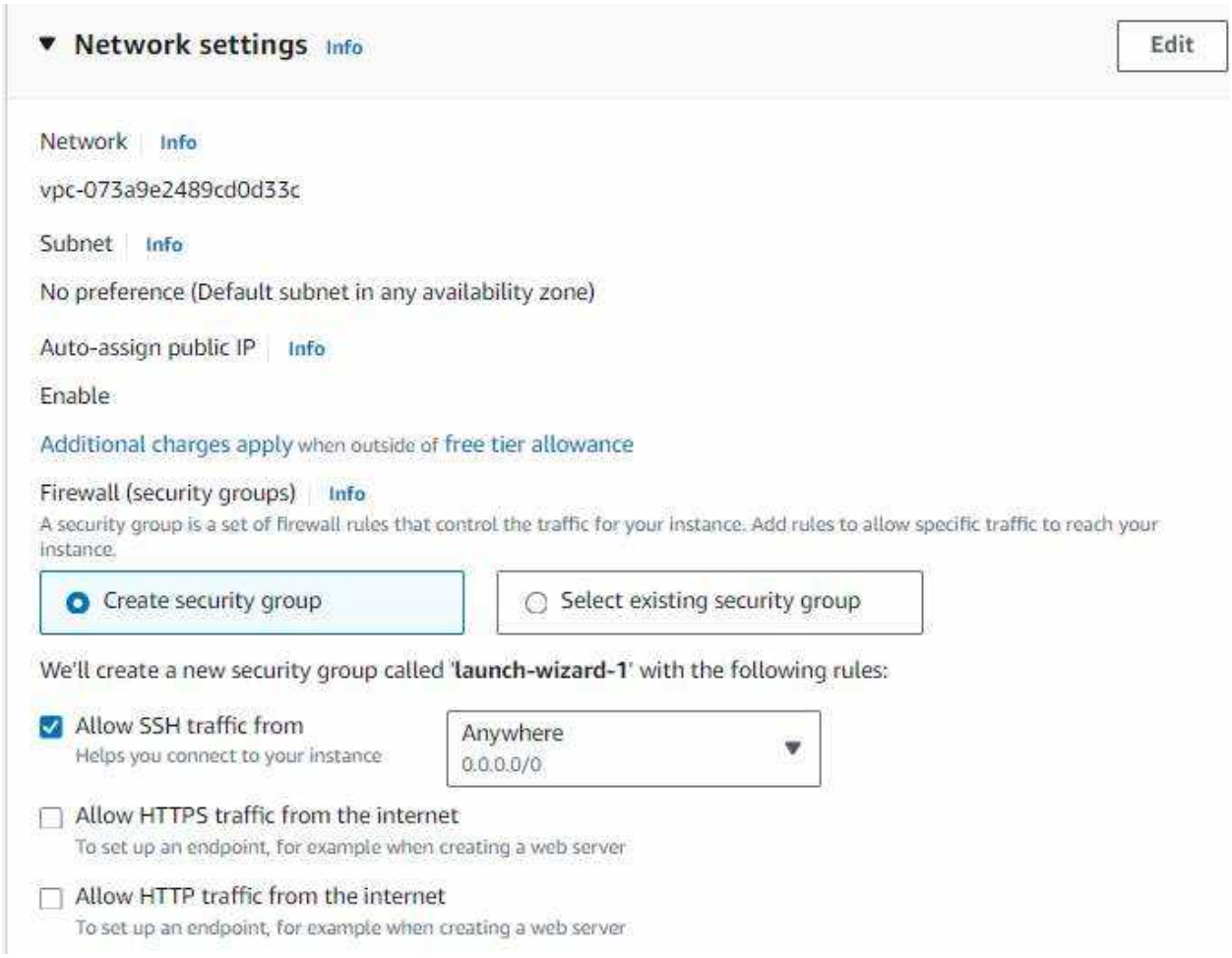
Add new volume

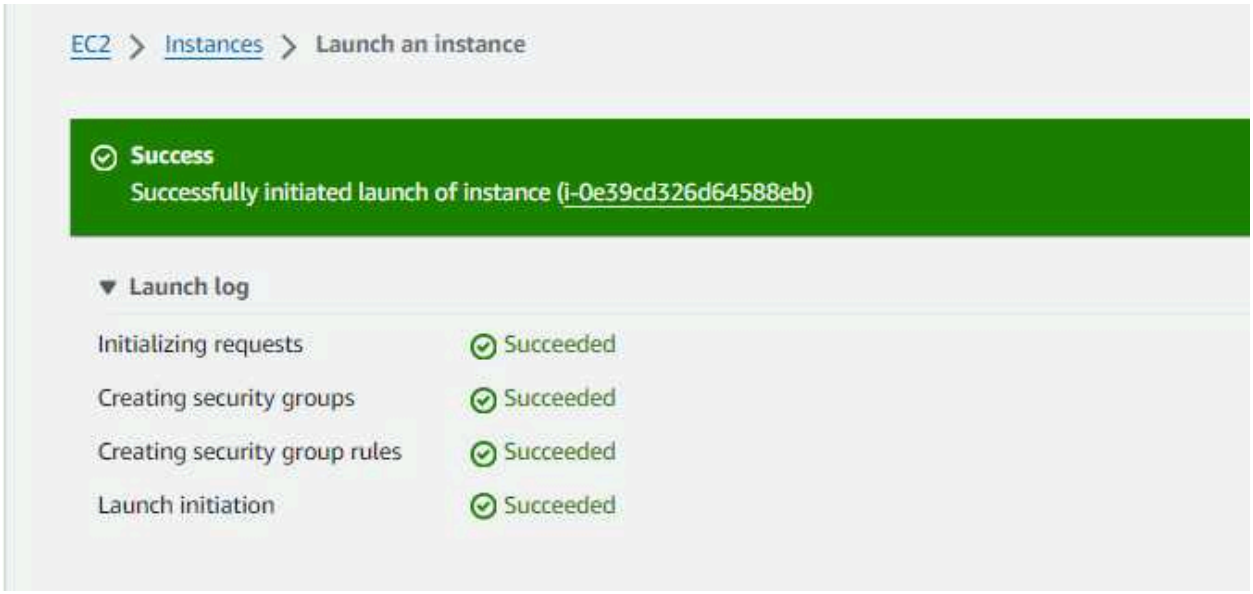
Click refresh to view backup information

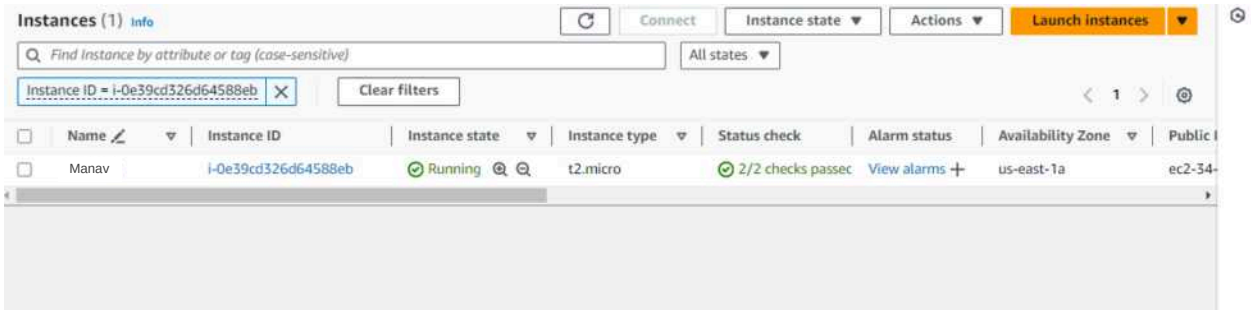
The tags that you assign determine whether the instance will be backed up by any Data Lifecycle Manager policies.

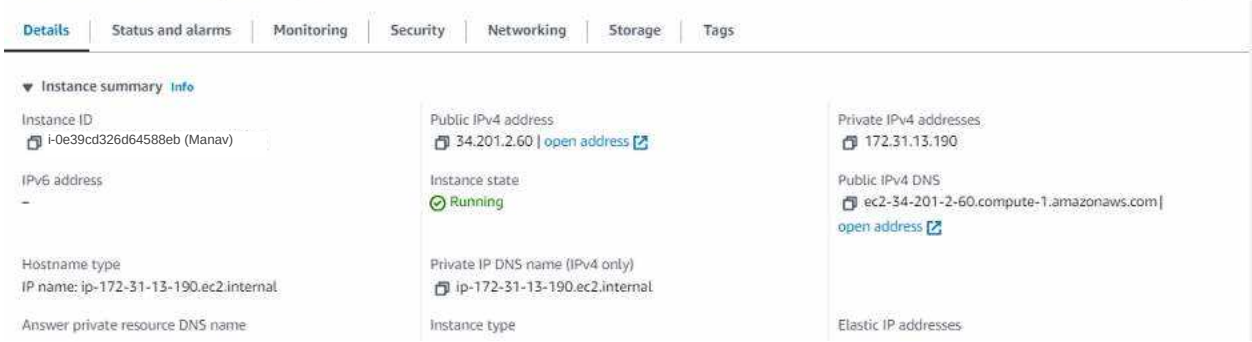
0 x File systems

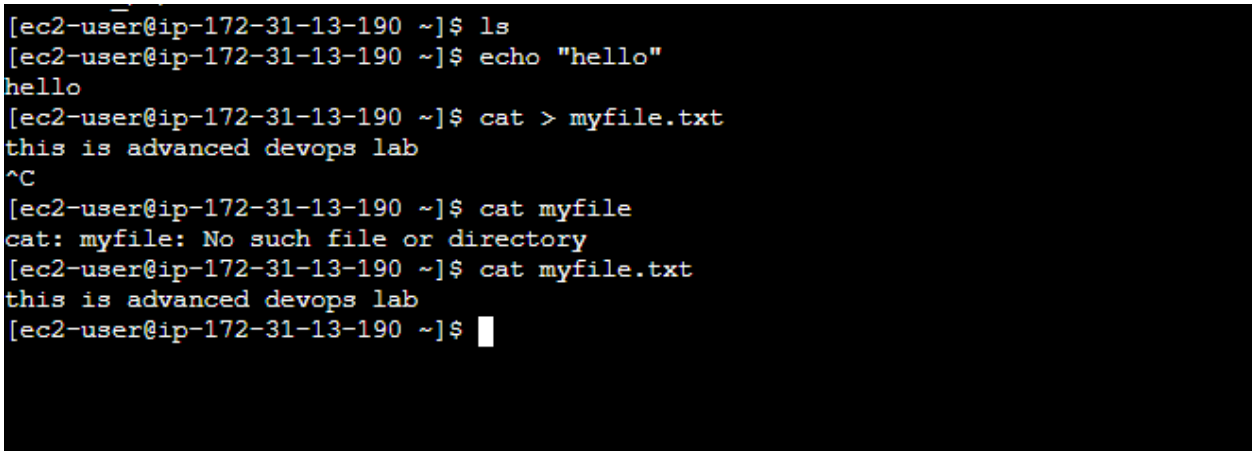
Edit


Network settings Info
Edit
Network Info
vpc-073a9e2489cd0d33c
Subnet Info
No preference (Default subnet in any availability zone)
Auto-assign public IP Info
Enable
Additional charges apply when outside of free tier allowance
Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
Create security group
Select existing security group
We'll create a new security group called 'launch-wizard-1' with the following rules:
Allow SSH traffic from
Helps you connect to your instance
Anywhere
0.0.0.0/0
Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server
Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server


EC2 > Instances > Launch an instance
Success
Successfully initiated launch of instance (i-0e39cd326d64588eb)
Launch log
Initializing requests
Succeeded
Creating security groups
Succeeded
Creating security group rules
Succeeded
Launch initiation
Succeeded


Instances (1) Info
Connect
Instance state
Actions
Launch instances
Find instance by attribute or tag (case-sensitive)
All states
Instance ID = i-0e39cd326d64588eb
Clear filters
1
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Manav
i-0e39cd326d64588eb
Running
t2.micro
2/2 checks passed
View alarms
us-east-1a

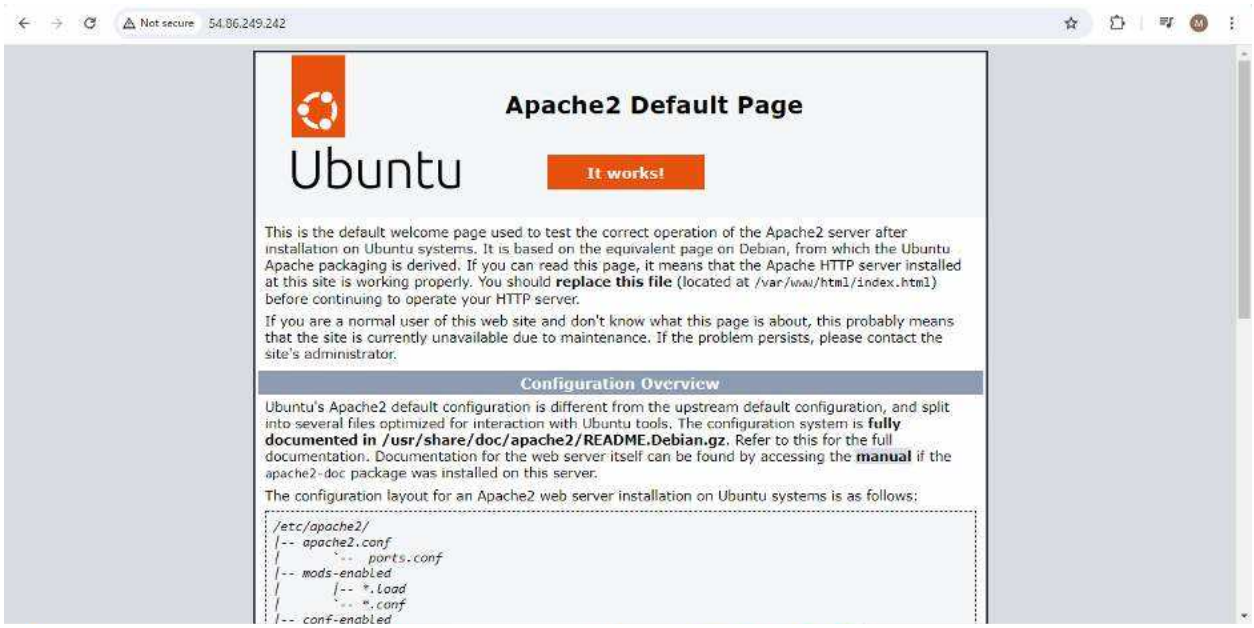

Details
Status and alarms
Monitoring
Security
Networking
Storage
Tags
Instance summary Info
Instance ID
i-0e39cd326d64588eb (Manav)
Public IPv4 address
34.201.2.60 | open address
Private IPv4 addresses
172.31.13.190
IPv6 address
-
Instance state
Running
Public IPv4 DNS
ec2-34-201-2-60.compute-1.amazonaws.com | open address
Hostname type
IP name: ip-172-31-13-190.ec2.internal
Private IP DNS name (IPv4 only)
ip-172-31-13-190.ec2.internal
Answer private resource DNS name
Instance type
Elastic IP addresses


[ec2-user@ip-172-31-13-190 ~]$ ls
[ec2-user@ip-172-31-13-190 ~]$ echo "hello"
hello
[ec2-user@ip-172-31-13-190 ~]$ cat > myfile.txt
this is advanced devops lab
^C
[ec2-user@ip-172-31-13-190 ~]$ cat myfile
cat: myfile: No such file or directory
[ec2-user@ip-172-31-13-190 ~]$ cat myfile.txt
this is advanced devops lab
[ec2-user@ip-172-31-13-190 ~]$

```
root@ip-172-31-32-173:~# sudo su
root@ip-172-31-32-173:~# apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sql
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-brows
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1
0 upgraded, 10 newly installed, 0 to remove and 26 not upgraded.
Need to get 1680 kB/2083 kB of archives.
After this operation, 8094 kB of additional disk space will be used.
```

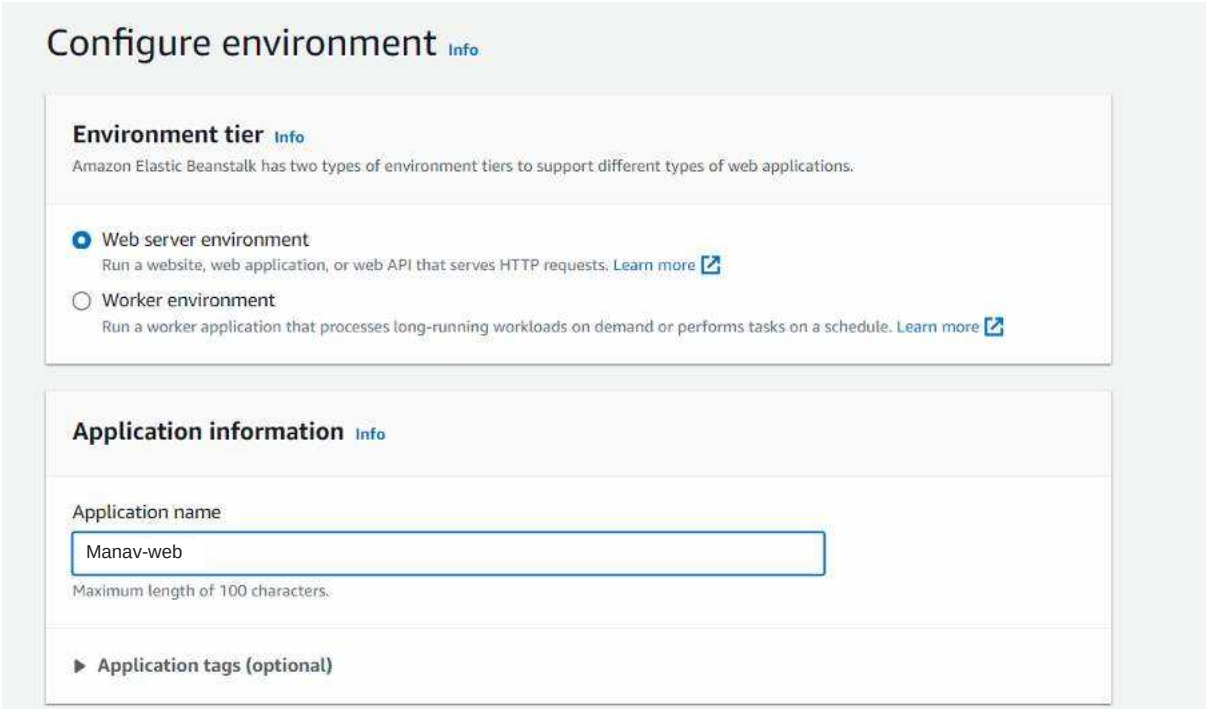
```
root@ip-172-31-32-173:~# systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Tue 2024-07-30 08:58:11 UTC; 44s ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 2619 (apache2)
      Tasks: 55 (limit: 1130)
     Memory: 5.4M (peak: 5.5M)
        CPU: 40ms
     CGroup: /system.slice/apache2.service
             ├─2619 /usr/sbin/apache2 -k start
             ├─2621 /usr/sbin/apache2 -k start
             └─2623 /usr/sbin/apache2 -k start

Jul 30 08:58:11 ip-172-31-32-173 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Jul 30 08:58:11 ip-172-31-32-173 systemd[1]: Started apache2.service - The Apache HTTP Server.
```

```
Jul 30 08:58:11 ip-172-31-32-173 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Jul 30 08:58:11 ip-172-31-32-173 systemd[1]: Started apache2.service - The Apache HTTP Server.
root@ip-172-31-32-173:~# cd /var/www/html
```


Not secure
54.86.249.242
Apache2 Default Page
Ubuntu
It works!
This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should replace this file (located at /var/www/html/index.html) before continuing to operate your HTTP server.
If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.
Configuration Overview
Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is fully documented in /usr/share/doc/apache2/README.Debian.gz. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the manual if the apache2-doc package was installed on this server.
The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:
/etc/apache2/
|-- apache2.conf
| `-- ports.conf
|-- mods-enabled
| |-- *.load
| `-- *.conf
|-- conf-enabled


Not secure
54.86.249.242/index2.html
Hi..Manav here

Elastic beanstalk



Platform Info

Platform type

- (•) Managed platform
  Platforms published and maintained by Amazon Elastic Beanstalk. Learn more
- ( ) Custom platform
  Platforms created and owned by you. This option is unavailable if you have no platforms.

Platform

Node.js

Platform branch

Node.js 20 running on 64bit Amazon Linux 2023

Platform version

6.2.0 (Recommended)

Review Info

Step 1: Configure environment

Edit

Environment information

Environment tier Web server environment	Application name Manav-web
Environment name Manav-web-env	Application code Sample application
Platform arn:aws:elasticbeanstalk:eu-north-1::platform/PHP 8.3 running on 64bit Amazon Linux 2023/4.3.2	

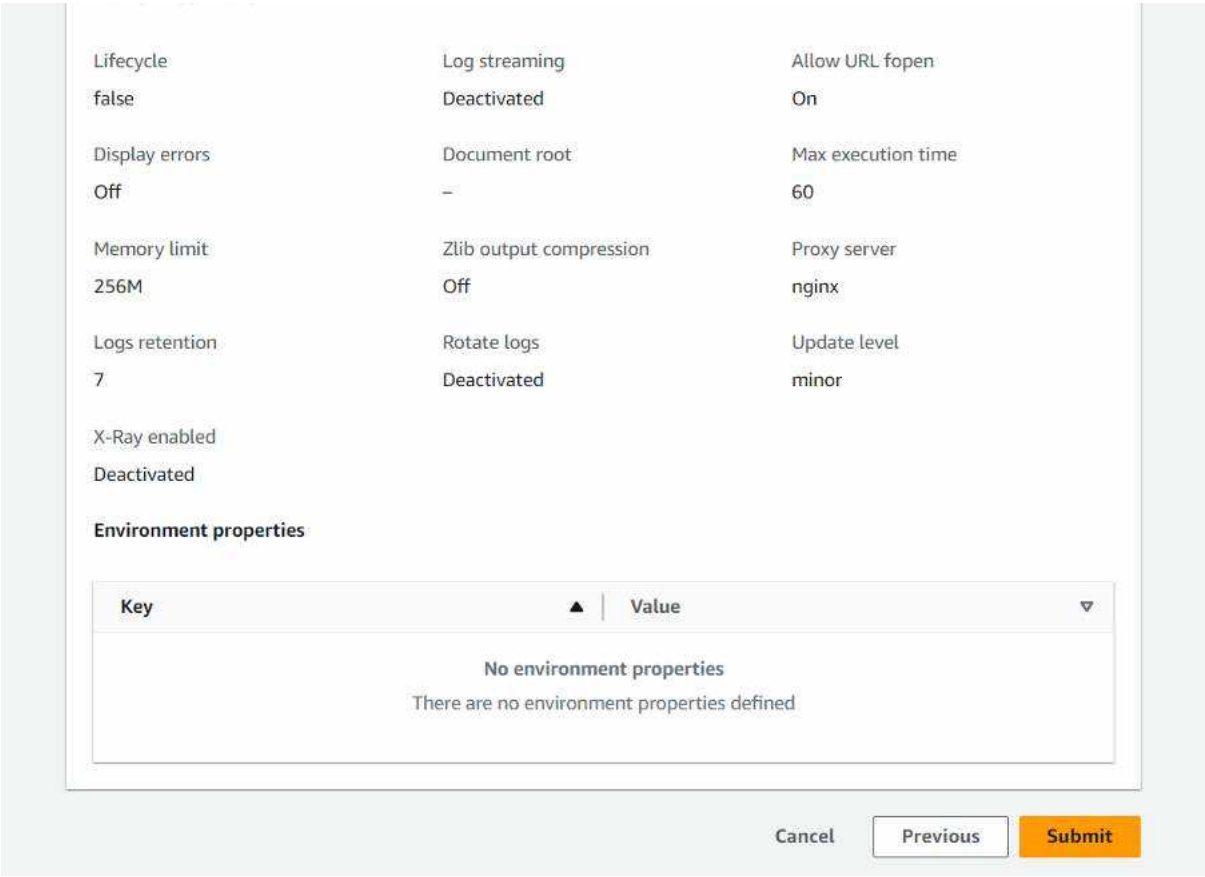
Step 2: Configure service access

Edit

Service access Info

Configure the service role and EC2 instance profile that Elastic Beanstalk uses to manage your environment. Choose an EC2 key pair to securely log in to your EC2 instances.

Service role	EC2 instance profile


Lifecycle
false
Log streaming
Deactivated
Allow URL fopen
On
Display errors
Off
Document root
–
Max execution time
60
Memory limit
256M
Zlib output compression
Off
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


Environment overview
Health
Unknown
Environment ID
e-trkmirvjuz
Domain
–
Application name
Manav-web
Platform
Change version
Platform
Node.js 20 running on 64bit Amazon Linux 2023/6.2.0
Running version
–
Platform state
Supported

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

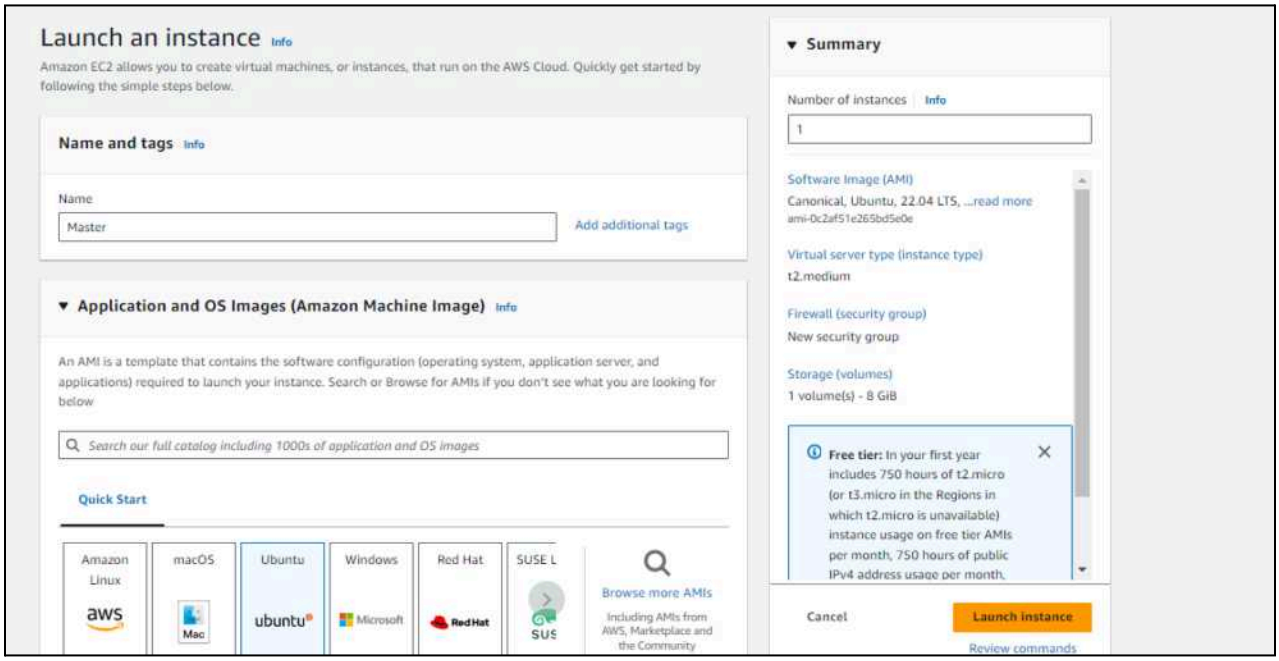
ADVANCE DEVOPS EXP 3

Name:Manav Punjabi
Class:D15A
Roll No:45

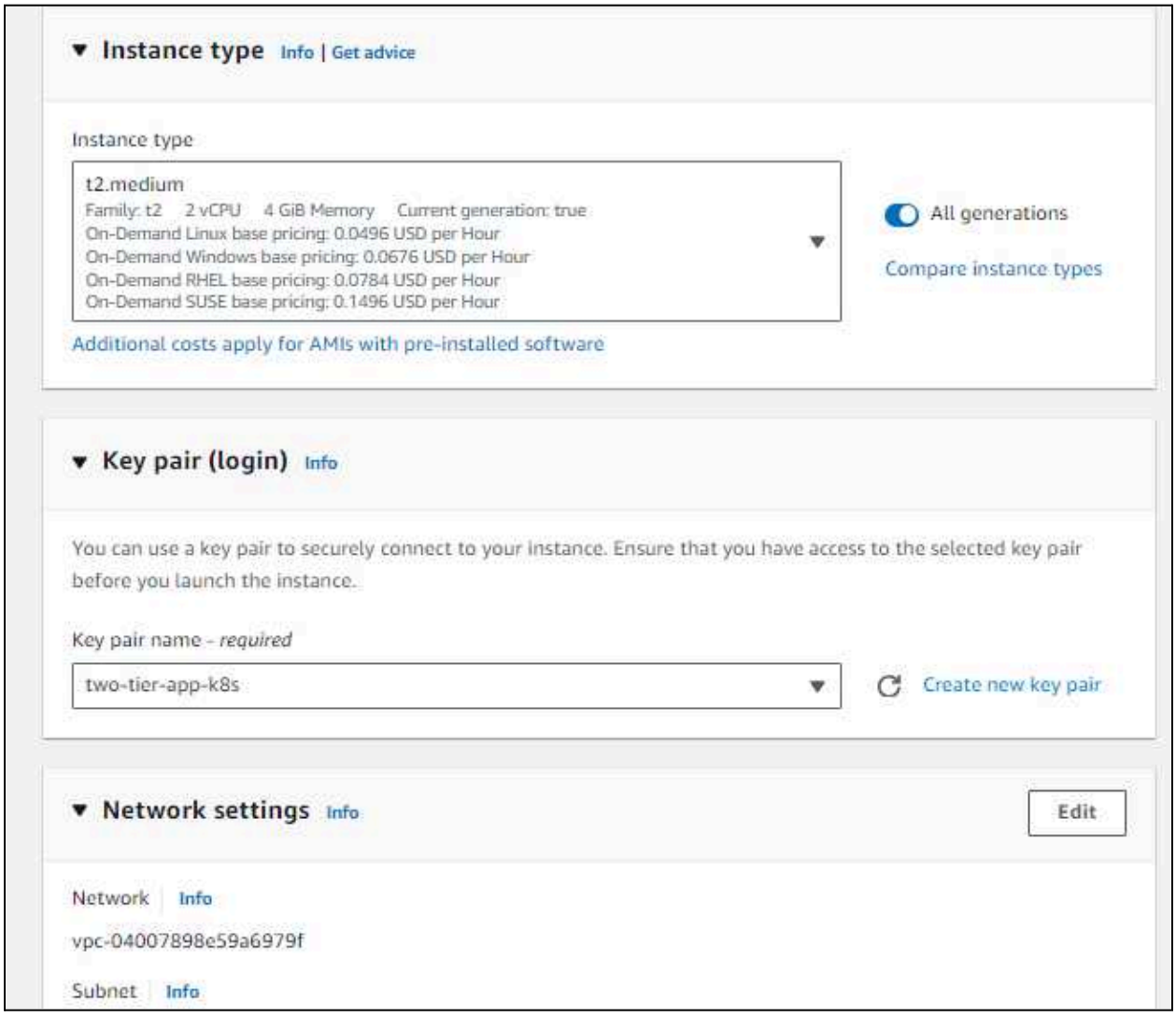
Aim:To understand the Kubernetes Cluster Architecture, install and Spin Up a Kubernetes Cluster on Linux Machines/Cloud Platforms.

Step 1: Pre-requisites

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)



Create key pair

Key pair name

Key pairs allow you to connect to your instance securely.

two-tier-app-k8s

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

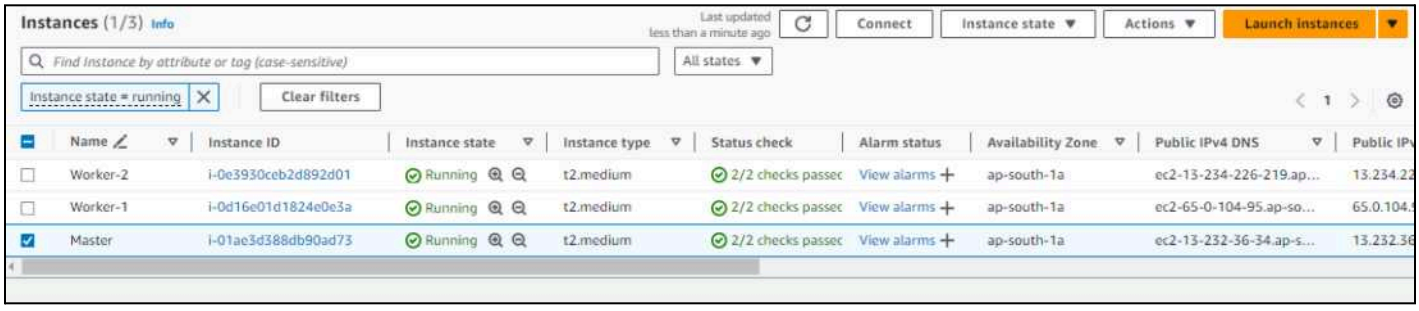
- RSA — RSA encrypted private and public key pair
- ED25519 — ED25519 encrypted private and public key pair

Private key file format

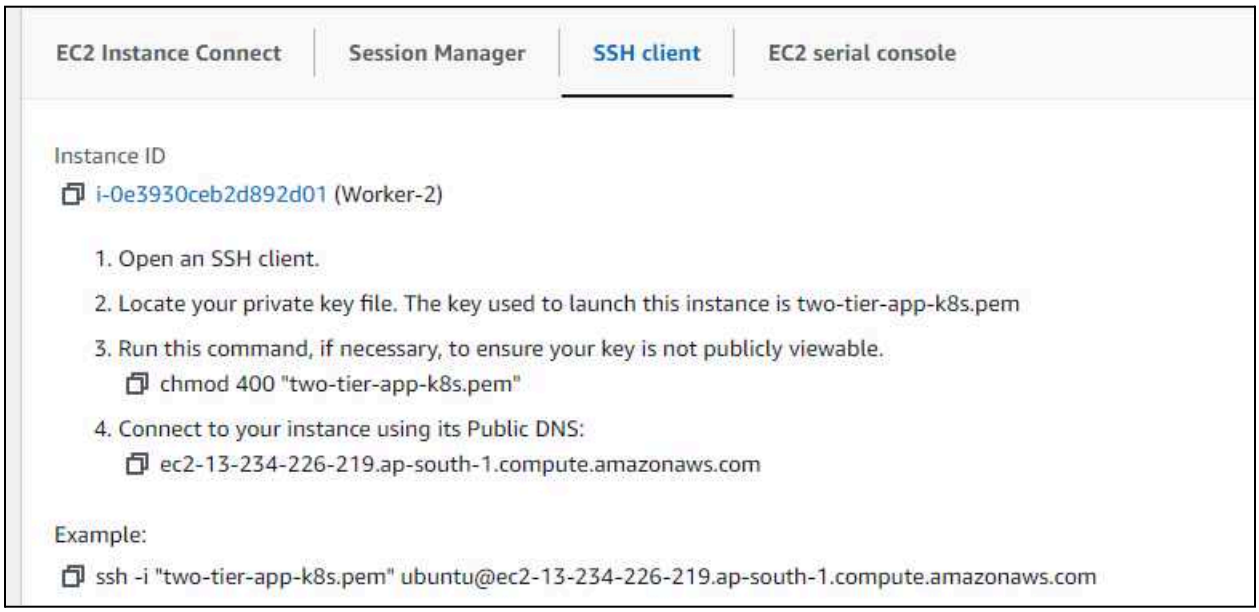
- .pem — For use with OpenSSH
- .ppk — For use with PuTTY

⚠ When prompted, store the private key in a secure and accessible location on

Cancel | Create key pair



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

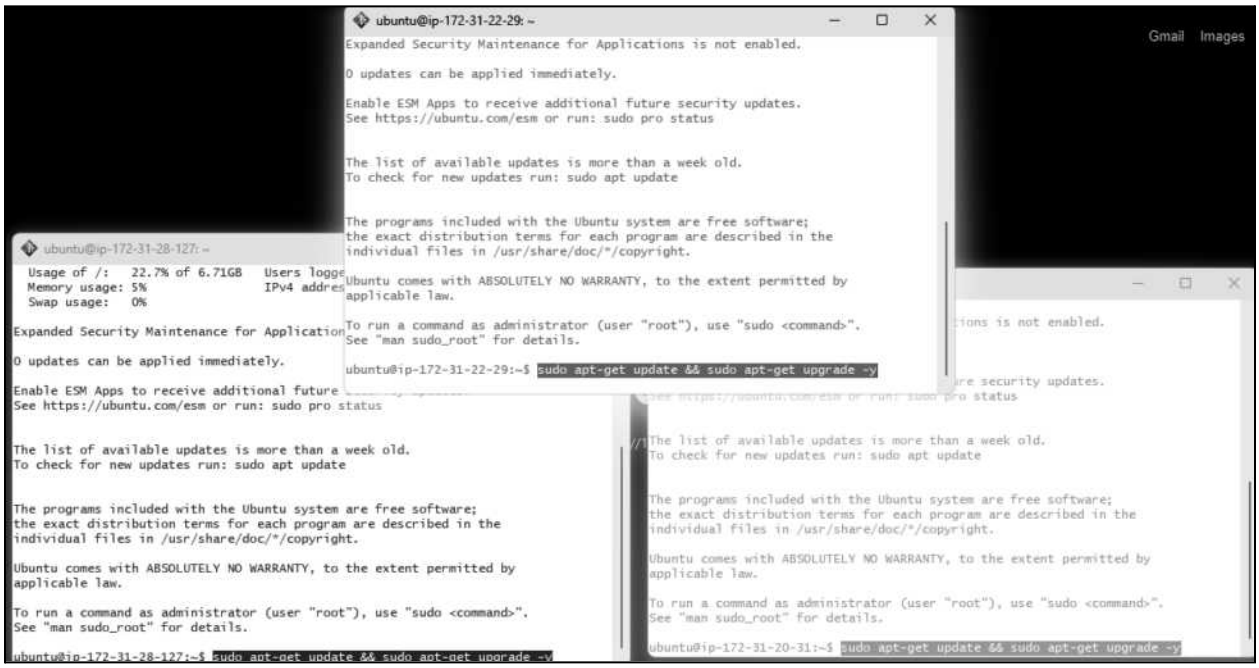



```
acer@TMP214-53 MINGW64 ~/Downloads
$ ssh -i "two-tier-app-k8s.pem" ubuntu@ec2-13-232-36-34.ap-south-1.compute.amazo
naws.com
The authenticity of host 'ec2-13-232-36-34.ap-south-1.compute.amazonaws.com (13.
232.36.34)' can't be established.
ED25519 key fingerprint is SHA256:uVGEO+FWYefj60j0ft70Sralv8NrzEi/IwxAtBY+EPE.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-13-232-36-34.ap-south-1.compute.amazonaws.com' (
ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.5.0-1022-aws x86_64)

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

 System information as of Wed Sep 11 14:07:10 UTC 2024

  System load:  0.0               Processes:             106
  Usage of /:   20.7% of 7.57GB   Users logged in:       0
  Memory usage: 5%                IPv4 address for eth0: 172.31.45.227
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status
```

Step 2:Prepare Nodes

2.1. Update the package manager on all nodes:

sudo apt-get update && sudo apt-get upgrade -y




```
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packag
es [14.1 MB]
Get:5 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-
en [5652 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f
Metadata [286 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Pack
ages [217 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translatio
n-en [112 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n
-f Metadata [8372 B]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 P
ackages [2023 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Transla
tion-en [352 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c
-n-f Metadata [17.8 kB]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted a
md64 Packages [2437 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted T
ranslation-en [419 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted a
```

2.2. Disable Swap (Kubernetes requires swap to be off):
sudo swapoff -a
sudo sed -i '/ swap / s/^/#/' /etc/fstab

```
ubuntu@ip-172-31-22-29:~$ sudo swapoff -a
sudo sed -i '/ swap / s/^/#/' /etc/fstab
```

2.3. Load necessary kernel modules for networking and iptables:
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF
sudo modprobe overlay
sudo modprobe br_netfilter

```
ubuntu@ip-172-31-22-29:~$ cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF
sudo modprobe overlay
sudo modprobe br_netfilter
overlay
br_netfilter
```

2.4. Configure sysctl settings for Kubernetes networking:
cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
EOF
sudo sysctl --system

```
ubuntu@ip-172-31-22-29:~$ cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF
sudo modprobe overlay
sudo modprobe br_netfilter
overlay
br_netfilter
ubuntu@ip-172-31-22-29:~$ cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
EOF
sudo sysctl --system
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
* Applying /etc/sysctl.d/10-console-messages.conf ...
kernel.printk = 4 4 1 7
* Applying /etc/sysctl.d/10-ipv6-privacy.conf ...
net.ipv6.conf.all.use_tempaddr = 2
net.ipv6.conf.default.use_tempaddr = 2
* Applying /etc/sysctl.d/10-kernel-hardening.conf ...
kernel.kptr_restrict = 1
```

Step 3: Install Docker

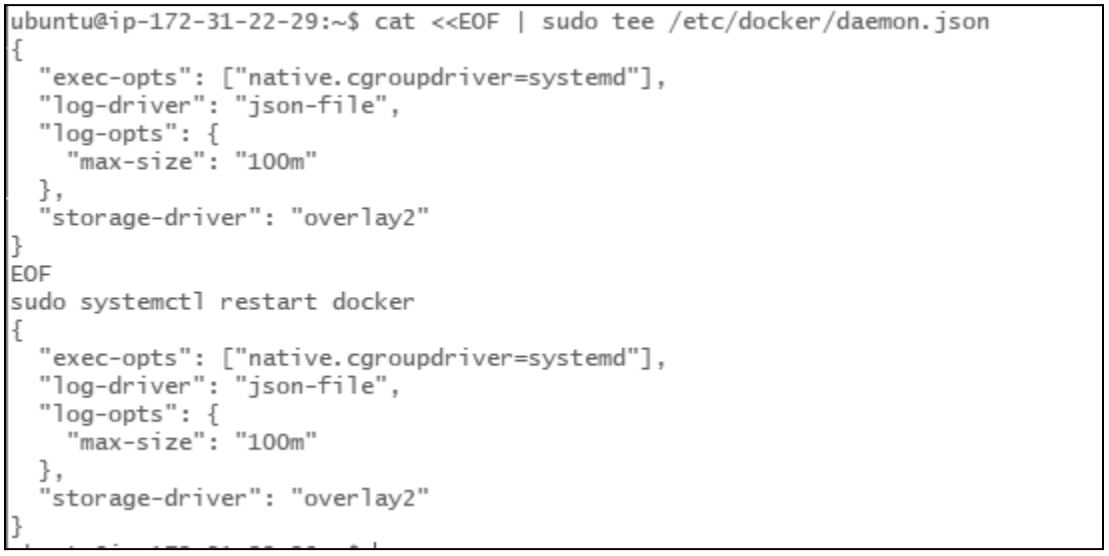
Kubernetes uses container runtimes like Docker. Install Docker on all nodes.

sudo apt-get update
sudo apt-get install -y apt-transport-https ca-certificates curl software-properties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
sudo apt-get update
sudo apt-get install -y docker-ce docker-ce-cli containerd.io

```
ubuntu@ip-172-31-22-29:~$ sudo apt-get update
sudo apt-get install -y apt-transport-https ca-certificates curl software-proper
ties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubun
tu $(lsb_release -cs) stable"
sudo apt-get update
sudo apt-get install -y docker-ce docker-ce-cli containerd.io
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Fetched 129 kB in 1s (241 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20230311ubuntu0.22.04.1).
ca-certificates set to manually installed.
curl is already the newest version (7.81.0-1ubuntu1.17).
curl set to manually installed.
software-properties-common is already the newest version (0.99.22.9).
software-properties-common set to manually installed.
```

Configure Docker for Kubernetes:

```
cat <<EOF | sudo tee /etc/docker/daemon.json
{
 "exec-opts": ["native.cgroupdriver=systemd"],
 "log-driver": "json-file",
 "log-opts": {
  "max-size": "100m"
 },
 "storage-driver": "overlay2"
}
EOF
sudo systemctl restart docker
```



Step 4: Install kubeadm, kubelet, kubectl

Install Kubernetes tools on all nodes.

4.1. Add Kubernetes APT repository:

```
sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg
https://packages.cloud.google.com/apt/doc/apt-key.gpg
echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg]
https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee
/etc/apt/sources.list.d/kubernetes.list
```

```
ubuntu@ip-172-31-22-29:~$ sudo curl -fsSLo /usr/share/keyrings/kubernetes-archiv
e-keyring.gpg https://packages.cloud.google.com/apt/doc/apt-key.gpg
echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https:/
/apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/k
ubernetes.list
deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.k
ubernetes.io/ kubernetes-xenial main
```

4.2. Install kubeadm, kubelet, and kubectl:

sudo apt-get update

sudo apt-get install -y kubelet kubeadm kubectl

sudo apt-mark hold kubelet kubeadm kubectl

```
ubuntu@ip-172-31-22-29:~$ sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:5 https://download.docker.com/linux/ubuntu jammy InRelease
```

Step 5: Initialize the Kubernetes Cluster on Master Node

On the master node:

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

```
ubuntu@ip-172-31-22-29:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --v
=5
Found multiple CRI endpoints on the host. Please define which one do you wish to
 use by setting the 'criSocket' field in the kubeadm configuration file: unix://
/var/run/containerd/containerd.sock, unix:///var/run/crio/crio.sock
k8s.io/kubernetes/cmd/kubeadm/app/util/runtime.detectCRISocketImpl
        cmd/kubeadm/app/util/runtime/runtime.go:167
k8s.io/kubernetes/cmd/kubeadm/app/util/runtime.DetectCRISocket
        cmd/kubeadm/app/util/runtime/runtime.go:175
k8s.io/kubernetes/cmd/kubeadm/app/util/config.SetNodeRegistrationDynamicDefaults
        cmd/kubeadm/app/util/config/initconfiguration.go:118
k8s.io/kubernetes/cmd/kubeadm/app/util/config.SetInitDynamicDefaults
        cmd/kubeadm/app/util/config/initconfiguration.go:64
k8s.io/kubernetes/cmd/kubeadm/app/util/config.DefaultedInitConfiguration
        cmd/kubeadm/app/util/config/initconfiguration.go:248
k8s.io/kubernetes/cmd/kubeadm/app/util/config.LoadOrDefaultInitConfiguration
        cmd/kubeadm/app/util/config/initconfiguration.go:282
k8s.io/kubernetes/cmd/kubeadm/app/cmd.newInitData
        cmd/kubeadm/app/cmd/init.go:319
k8s.io/kubernetes/cmd/kubeadm/app/cmd.newCmdInit.func3
        cmd/kubeadm/app/cmd/init.go:170
k8s.io/kubernetes/cmd/kubeadm/app/cmd/phases/workflow.(*Runner).InitData
        cmd/kubeadm/app/cmd/phases/workflow/runner.go:183
k8s.io/kubernetes/cmd/kubeadm/app/cmd.newCmdInit.func1
```

5.1. Set up kubectl on the master node:
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

```
ubuntu@ip-172-31-22-29:~$ 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 --v=5
Found multiple CRI endpoints on the host. Please define which one do you wish to use by setting the 'criSocket' field in the kubeadm configuration file: unix:///var/run/containerd/containerd
ock, unix:///var/run/crio/crio.sock
To see the stack trace of this error execute with --v=5 or higher
Found multiple CRI endpoints on the host. Please define which one do you wish to use by setting the 'criSocket' field in the kubeadm configuration file: unix:///var/run/containerd/containerd
ock, unix:///var/run/crio/crio.sock
```

Step 6: Install a Pod Network Add-on

To enable communication between pods, install a pod network plugin like Flannel or Calico.

Install Flannel:

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

```
ubuntu@ip-172-31-22-29:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml --validate=false

E0913 15:35:04.261458   19259 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.1:808
E0913 15:35:04.261902   19259 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.1:808
E0913 15:35:04.263424   19259 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.1:808
E0913 15:35:04.263795   19259 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.1:808
E0913 15:35:04.265840   19259 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.1:808
E0913 15:35:04.266524   19259 memcache.go:265] couldn't get current server API group list: Get "http://localhost:8080/api?timeout=32s": dial tcp 127.0.0.1:808
unable to recognize "https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml": Get "http://localhost:8080/api?timeout=32s": dia
```

Step 7: Join Worker Nodes to the Cluster

On the **worker nodes**, run the command provided by the master node during initialization . It looks something like this:

sudo kubeadm join <master-ip>:6443 --token <token> --discovery-token-ca-cert-hash sha256:<hash>

```
clusterrolebinding.rbac.authorization.k8s.io/calico-cni-plugin created
daemonset.apps/calico-node created
deployment.apps/calico-kube-controllers created
kubeadm join 172.31.62.216:6443 --token br7fe5.hq28adbmn1mu17ky --discovery-token-ca-cert-hash sha256:2bc469a8d14fbeb0f879328d2b416fad
32b29a8505d3f448b98703fff3b014d9
```

Step 8: Verify the Cluster

Once the worker node joins, check the status on the **master node**

```
ubuntu@ip-172-31-45-227:~$ kubectl get nodes
NAME               STATUS   ROLES           AGE     VERSION
ip-172-31-43-211   Ready    <none>          50s     v1.29.0
ip-172-31-45-13    Ready    <none>          34s     v1.29.0
ip-172-31-45-227   Ready    control-plane   5m17s   v1.29.0
ubuntu@ip-172-31-45-227:~$
```

ADVANCE DEVOPS EXP 4

Name:Manav Punjabi
Class:D15A
Roll No:45

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

Step 1: Install Kubectl on Ubuntu

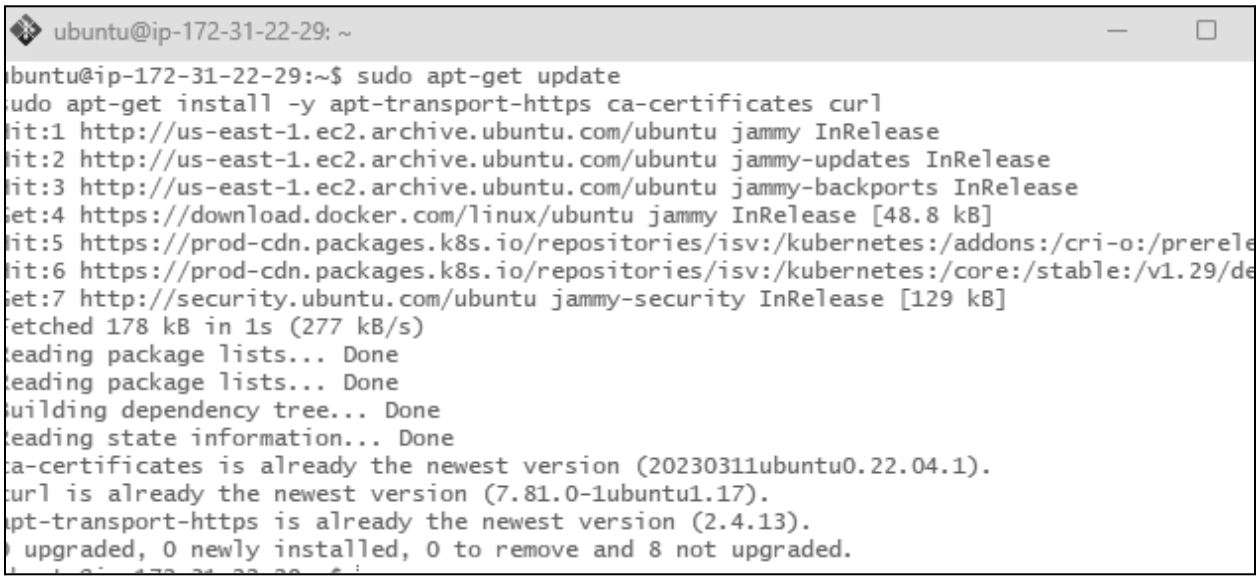
1.1 Add Kubernetes APT repository

First, add the Kubernetes repository to your system.

1. **Install prerequisites:**

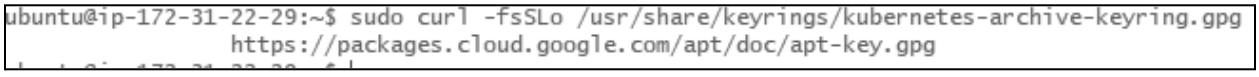
   sudo apt-get update

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



2. **Add the GPG key for Kubernetes:**

   sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg https://packages.cloud.google.com/apt/doc/apt-key.gpg



3. **Add the Kubernetes repository:**

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

```
ubuntu@ip-172-31-22-29:~$  echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring
.gpg]  https://apt.kubernetes.io/ kubernetes-focal main" | sudo tee /etc/apt/sources.list.d/ku
bernetes.list
deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg]  https://apt.kubernetes.io/
 kubernetes-focal main
```

1.2 Install kubectl

Now install kubectl:

sudo apt-get update

sudo apt-get install -y kubectl

```
ubuntu@ip-172-31-22-29:~$ sudo apt-get update
sudo apt-get install -y kubectl
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:5 https://download.docker.com/linux/ubuntu jammy InRelease
Hit:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/addons:/cri-o:/prerelease:/main/deb  InRelease
Ign:7 https://packages.cloud.google.com/apt kubernetes-focal InRelease
Err:8 https://packages.cloud.google.com/apt kubernetes-focal Release
  404  Not Found [IP: 172.253.62.138 443]
Reading package lists... Done
E: The repository 'https://apt.kubernetes.io kubernetes-focal Release' does not have a Release file.
N: Updating from such a repository can't be done securely, and is therefore disabled by default.
N: See apt-secure(8) manpage for repository creation and user configuration details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
kubectl is already the newest version (1.29.0-1.1).
0 upgraded, 0 newly installed, 0 to remove and 5 not upgraded.
```

Verify the installation(extra):

kubectl version --client

```
ubuntu@ip-172-31-22-29:~$ kubectl version --client
Client Version: v1.29.0
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
```

Step 2: Deploying Your Application on Kubernetes

2.1 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

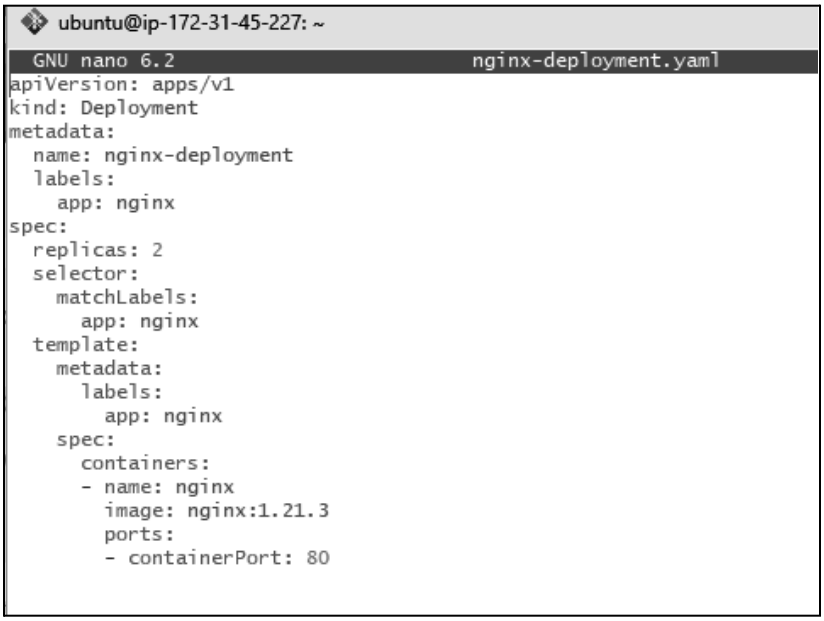
```
ubuntu@ip-172-31-45-227:~$ kubectl get nodes
NAME                STATUS   ROLES           AGE     VERSION
ip-172-31-43-211    Ready    <none>          50s     v1.29.0
ip-172-31-45-13     Ready    <none>          34s     v1.29.0
ip-172-31-45-227    Ready    control-plane   5m17s   v1.29.0
ubuntu@ip-172-31-45-227:~$
```

Step 3:**Create the Deployment YAML file**

a)Create the YAML file: Use a text editor to create a file named nginx-deployment.yaml

```
ubuntu@ip-172-31-45-227:~$ nano nginx-deployment.yaml
```

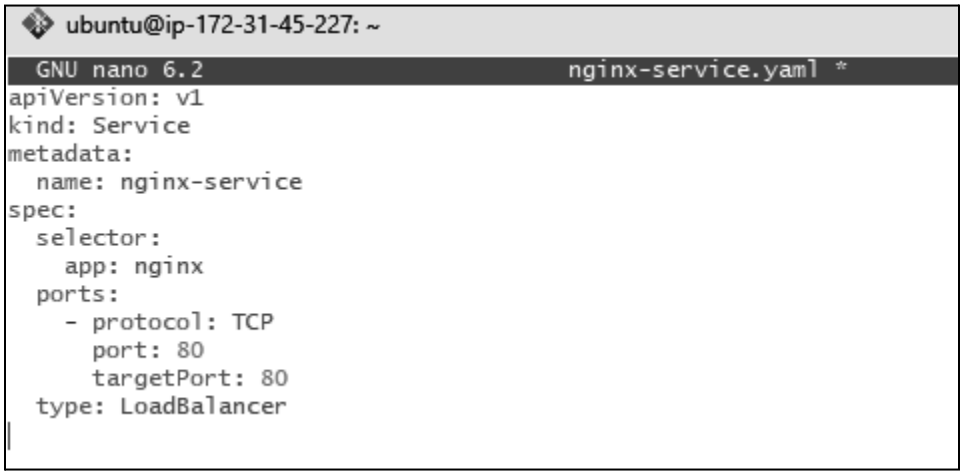
b)Add the Deployment Configuration: Copy and paste the following YAML content into the file. Save and exit the editor (Press Ctrl+X, then Y, and Enter).



Step 4:Create the Service YAML File

a)Create the YAML File: Create another file named nginx-service.yaml

b)Add the Service Configuration: Copy and paste the following YAML content into the file given below.



Step 5:Apply the YAML Files

a)Deploy the Application: Use kubectl to create the Deployment and Service from the YAML files.

```
ubuntu@ip-172-31-45-227:~$ kubectl apply -f nginx-deployment.yaml
kubectl apply -f nginx-service.yaml
deployment.apps/nginx-deployment created
service/nginx-service created
```

b)Verify the Deployment: Check the status of your Deployment,Pods and Services.

```
ubuntu@ip-172-31-45-227:~$ kubectl get deployments
kubectl get pods
kubectl get services
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   2/2     2            2           40s
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-6b4d6fdbf-6k84m    1/1     Running   0          40s
nginx-deployment-6b4d6fdbf-9d8j6    1/1     Running   0          40s
NAME            TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
kubernetes      ClusterIP      10.96.0.1        <none>        443/TCP        40m
nginx-service   LoadBalancer   10.106.182.152   <pending>     80:32317/TCP   40s
```

Describe the deployment(Extra)

```
ubuntu@ip-172-31-45-227:~$ kubectl get deployments
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   1/1     1            1           14h
ubuntu@ip-172-31-45-227:~$ kubectl describe deployment
Name:                   nginx-deployment
Namespace:              default
CreationTimestamp:      Wed, 11 Sep 2024 17:16:17 +0000
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 2
Selector:               app=nginx
Replicas:               1 desired | 1 updated | 1 total | 1 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginx
  Containers:
   nginx:
    Image:        nginx:latest
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:
      /usr/share/nginx/html from website-volume (rw)
  Volumes:
   website-volume:
    Type:      ConfigMap (a volume populated by a ConfigMap)
    Name:      nginx-website
    Optional:  false
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  nginx-deployment-6b4d6fdbf (0/0 replicas created)
NewReplicaSet:   nginx-deployment-776b8fd845 (1/1 replicas created)
Events:          <none>
```

Step 6:Ensure Service is Running

6.1 **Verify Service**: Run the following command to check the services running in your cluster:

kubectl get service

```
ubuntu@ip-172-31-45-227:~$ kubectl get service
NAME            TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
kubernetes      ClusterIP   10.96.0.1        <none>        443/TCP        16h
nginx           NodePort    10.106.0.176     <none>        80:32618/TCP   76m
nginx-service   NodePort    10.106.182.152   <none>        80:30007/TCP   15h
nginx2          NodePort    10.99.32.156     <none>        80:31421/TCP   8s
```

Step 7: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/<service-name> <local-port>:<service-port>

```
ubuntu@ip-172-31-45-227:~$ kubectl port-forward service/nginx-service 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
```

This command will forward local port 8080 on your machine to port 80 of the service nginx-service running inside the cluster.

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-172-31-45-227:~$ kubectl port-forward service/nginx-service 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
^Cubuntu@ip-172-31-45-227:~kubectl port-forward service/nginx-service 8081:8080
Forwarding from 127.0.0.1:8081 -> 80
Forwarding from [::1]:8081 -> 80
^Cubuntu@ip-172-31-45-227:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-776b8fd845-k9cx4   1/1     Running   0          113m
ubuntu@ip-172-31-45-227:~$ kubectl logs nginx-deployment-776b8fd845-k9cx4
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/09/12 06:35:51 [notice] 1#1: using the "epoll" event method
2024/09/12 06:35:51 [notice] 1#1: nginx/1.27.1
2024/09/12 06:35:51 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2024/09/12 06:35:51 [notice] 1#1: OS: Linux 6.5.0-1022-aws
2024/09/12 06:35:51 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2024/09/12 06:35:51 [notice] 1#1: start worker processes
2024/09/12 06:35:51 [notice] 1#1: start worker process 24
2024/09/12 06:35:51 [notice] 1#1: start worker process 25
```

Step 8: Access the Application Locally

1. **Open a Web Browser**: Now open your web browser and go to the following URL:

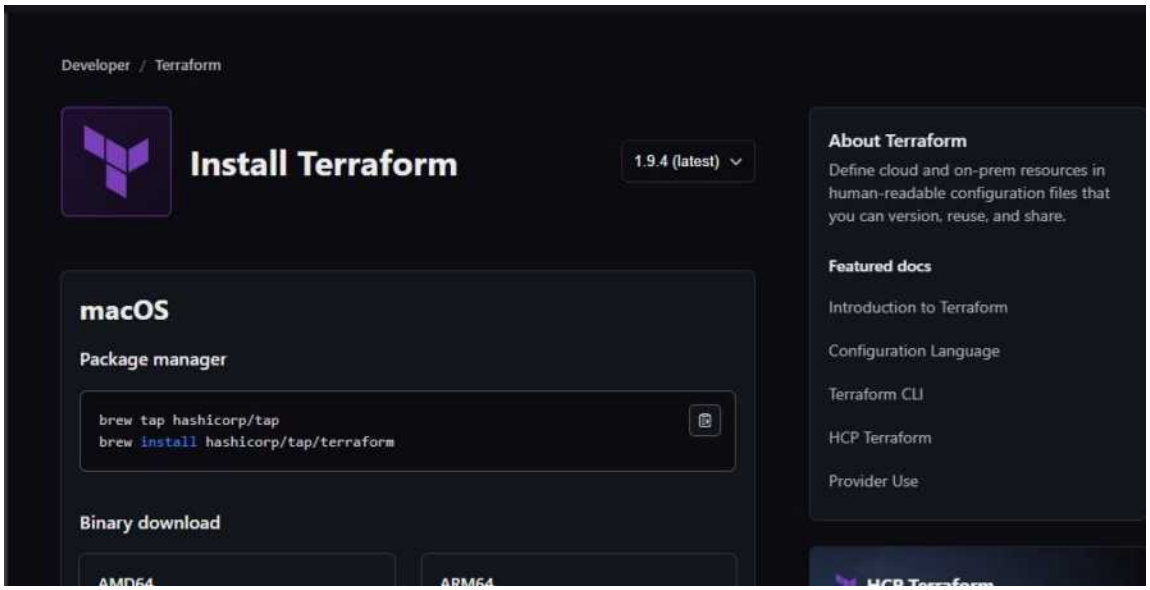
   http://localhost:8080

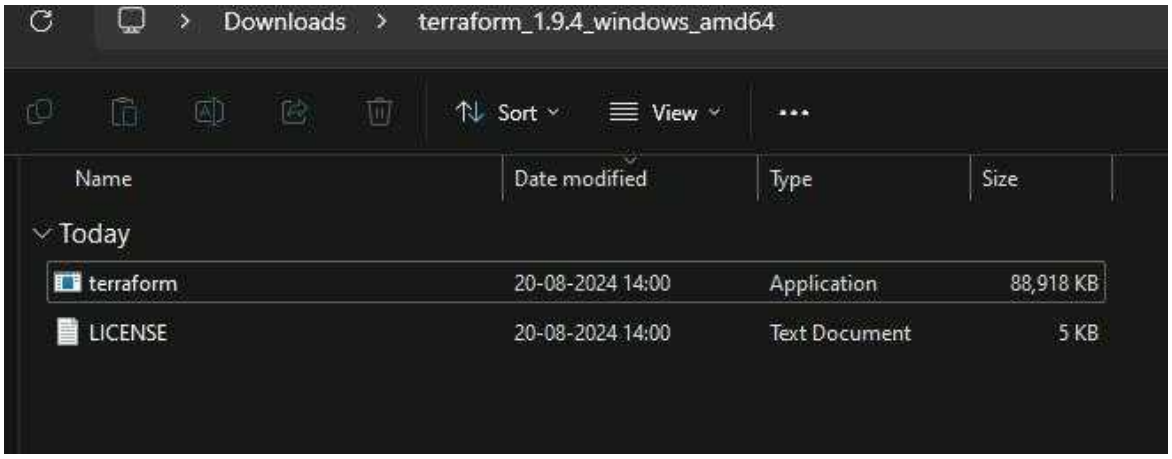
You should see the application (in this case, Nginx) that you have deployed running in the Kubernetes cluster, served locally via port 8080.

In case the port 8080 is unavailable, try using a different port like 8081

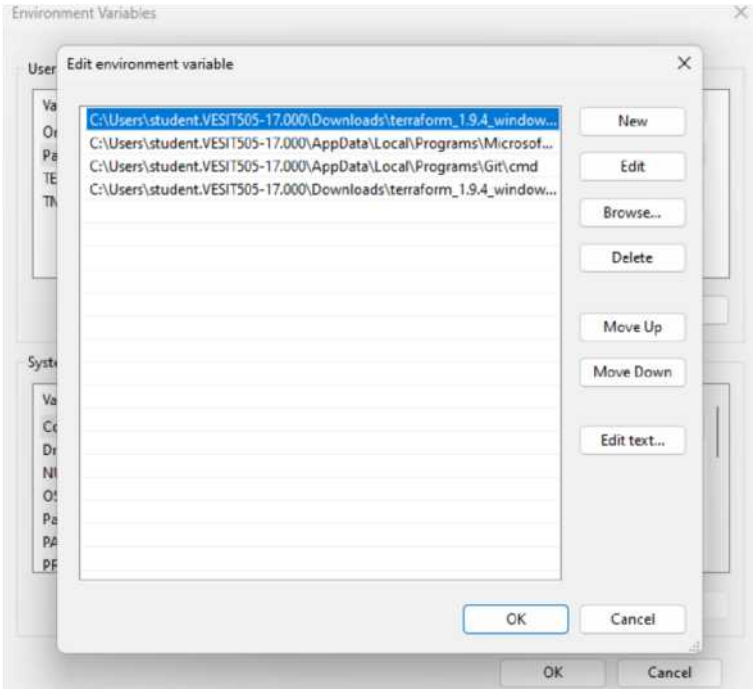
AdvDevops Experiment 5

AIM: Installation and configuration of terraform on Windows





M anav Punjabi D15A 45




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

PS C:\Users\student.VESIT505-17.000\Downloads\terraform_1.9.4_windows_amd64> terraform
Usage: terraform [global options] <subcommand> [args]

The available commands for execution are listed below.
The primary workflow commands are given first, followed by
less common or more advanced commands.

Main commands:
  init          Prepare your working directory for other commands
  validate      Check whether the configuration is valid
  plan          Show changes required by the current configuration
  apply         Create or update infrastructure
  destroy       Destroy previously-created infrastructure

All other commands:
  console       Try Terraform expressions at an interactive command prompt
  fmt           Reformat your configuration in the standard style
  force-unlock  Release a stuck lock on the current workspace
  get           Install or upgrade remote Terraform modules
  graph         Generate a Graphviz graph of the steps in an operation
  import        Associate existing infrastructure with a Terraform resource
  login         Obtain and save credentials for a remote host
  logout        Remove locally-stored credentials for a remote host
  metadata      Metadata related commands
  output        Show output values from your root module
  providers     Show the providers required for this configuration
  refresh       Update the state to match remote systems
  show          Show the current state or a saved plan
  state         Advanced state management
  taint         Mark a resource instance as not fully functional
  test          Execute integration tests for Terraform modules
  untaint       Remove the 'tainted' state from a resource instance
```

```
PS C:\Users\student.VESIT505-17.000\Downloads\terraform_1.9.4_windows_amd64> terraform --version
Terraform v1.9.4
on windows_amd64
PS C:\Users\student.VESIT505-17.000\Downloads\terraform_1.9.4_windows_amd64>
```

Manav Punjabi D15A 45

ADVANCE DEVOPS EXP 6

Aim : To Build, change, and destroy AWS / GCP /Microsoft Azure/ DigitalOcean infrastructure Using Terraform. (S3 bucket or Docker) fdp.

Part A: Creating docker image using terraform

Step 1: Check Docker functionality

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

C:\Users\student>docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

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

Management Commands:
  builder     Manage builds
  buildx*     Docker Buildx
  checkpoint  Manage checkpoints
  compose*    Docker Compose
  container   Manage containers
  context     Manage contexts
  debug*      Get a shell into any image or container
  desktop*    Docker Desktop commands (Alpha)
  dev*        Docker Dev Environments
  extension*  Manages Docker extensions
  feedback*   Provide feedback, right in your terminal!
```

Check for the docker version with the following command.

```
C:\Users\student>docker --version
Docker version 27.1.1, build 6312585

C:\Users\student>
```

Create a folder named ‘Terraform Scripts’ in which we save our different typesof scripts which will be further used in this experiment.

Step 2:

Creating a new folder named ‘Docker’ in the ‘TerraformScripts’ folder.
Creating a new docker.tf file using Atom editor and write the following contents into.

This will create a Ubuntu Linux container

docker.tf

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

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

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

# Create a container
resource "docker_container" "foo" {
  image = docker_image.ubuntu.image_id
  name  = "foo"
  command = ["sleep", "3600"]
}
```

Step 3: Execute Terraform Init command to initialize the resources

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

Terraform has been successfully initialized!

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

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

Step 4: Execute Terraform plan to see the available resources

```
PS C:\Users\Admin\TerraformScripts\Docker> terraform plan

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

Terraform will perform the following actions:

  # docker_container.foo will be created
  + resource "docker_container" "foo" {
      + attach           = false
      + bridge           = (known after apply)
      + command          = [
          + "sleep",
          + "3600",
        ]
      + 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.
```

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"**

```
PS C:\Users\Admin\TerraformScripts\Docker> terraform apply

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

Terraform will perform the following actions:

  # docker_container.foo will be created
  + resource "docker_container" "foo" {
      + attach           = false
      + bridge           = (known after apply)
      + command          = [
          + "sleep",
          + "3600",
        ]
      + container_logs   = (known after apply)
      + entrypoint       = (known after apply)
      + env              = (known after apply)
      + exit_code        = (known after apply)
      + gateway          = (known after apply)
      + hostname         = (known after apply)
      + id               = (known after apply)
      + image            = (known after apply)
      + init             = (known after apply)
      + ip_address       = (known after apply)
      + ip_prefix_length = (known after apply)
      + ipc_mode         = (known after apply)
      + log_driver       = (known after apply)
      + logs             = false
      + must_run         = true
      + name             = "foo"
      + network_data     = (known after apply)
      + read_only        = false
```

```
      + remove_volumes   = true
      + restart          = "no"
      + rm               = false
      + runtime          = (known after apply)
      + security_opts    = (known after apply)
      + shm_size         = (known after apply)
      + start            = true
      + stdin_open       = false
      + stop_signal      = (known after apply)
      + stop_timeout     = (known after apply)
      + tty              = false

      + healthcheck (known after apply)

      + labels (known after apply)
    }

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

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

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

  Enter a value: yes
```

```
docker_image.ubuntu: Creating...
docker_image.ubuntu: Creation complete after 9s [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Creating...
docker_container.foo: Creation complete after 2s [id=01adf07e5918931fee9b90073726a03671037923dd92032ce0e15bbb764a6f24]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```

Before Executing Apply step:

```
PS C:\Users\Admin\TerraformScripts\Docker> docker images
REPOSITORY                          TAG       IMAGE ID       CREATED         SIZE
```

After Executing Apply step:

```
PS C:\Users\Admin\TerraformScripts\Docker> docker images
REPOSITORY                          TAG       IMAGE ID       CREATED         SIZE
ubuntu                              latest    edbfe74c41f8   3 weeks ago     78.1MB
```

Step 6: Execute Terraform destroy to delete the configuration, which will automatically delete the Ubuntu Container.

```
PS C:\Users\Admin\TerraformScripts\Docker> terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=01adf07e5918931fee9b90073726a03671037923dd92032ce0e15bbb764a6f24]

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

Terraform will perform the following actions:

  # docker_container.foo will be destroyed
  - resource "docker_container" "foo" {
      - attach           = false -> null
      - command          = [
          - "sleep",
          - "3600",
        ] -> null
      - cpu_shares       = 0 -> null
      - dns              = [] -> null
      - dns_opts         = [] -> null
      - dns_search       = [] -> null
      - entrypoint       = [] -> null
      - env              = [] -> null
      - gateway          = "172.17.0.1" -> null
      - group_add        = [] -> null
      - hostname         = "01adf07e5918" -> null
      - id               = "01adf07e5918931fee9b90073726a03671037923dd92032ce0e15bbb764a6f24" -> null
      - image            = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
      - init             = false -> null
      - ip_address       = "172.17.0.2" -> null
      - ip_prefix_length = 16 -> null
      - ipc_mode         = "private" -> null
      - links            = [] -> null
      - log_driver       = "json-file" -> null
      - log_opts         = {} -> null
      - logs             = false -> null
      - max_retry_count  = 0 -> null
```

```
      - memory            = 0 -> null
      - memory_swap       = 0 -> null
      - must_run          = true -> null
      - name              = "foo" -> null
      - network_data      = [
          - {
              - gateway                   = "172.17.0.1"
              - global_ipv6_prefix_length = 0
              - ip_address                = "172.17.0.2"
              - ip_prefix_length          = 16
              - network_name              = "bridge"
                # (2 unchanged attributes hidden)
            },
        ] -> null
      - network_mode      = "default" -> null
      - privileged        = false -> null
      - publish_all_ports = false -> null
      - read_only         = false -> null
      - remove_volumes    = true -> null
      - restart           = "no" -> null
      - rm                = false -> null
      - runtime           = "runc" -> null
      - security_opts     = [] -> null
      - shm_size          = 64 -> null
      - start             = true -> null
      - stdin_open        = false -> null
      - stop_timeout      = 0 -> null
      - storage_opts      = {} -> null
      - sysctls           = {} -> null
      - tmpfs             = {} -> null
      - tty               = false -> null
        # (8 unchanged attributes hidden)
    }
```

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

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

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

  Enter a value: yes

docker_container.foo: Destroying... [id=01adf07e5918931fee9b90073726a03671037923dd92032ce0e15bbb764a6f24]
docker_container.foo: Destruction complete after 0s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image.ubuntu: Destruction complete after 1s

Destroy complete! Resources: 2 destroyed.
```

Docker images After Executing Destroy step

```
PS C:\Users\Admin\TerraformScripts\Docker> docker images
REPOSITORY                          TAG       IMAGE ID       CREATED         SIZE
```

ADVANCE DEVOPS EXP 7

Name:Manav Punjabi
Class:D15A
Roll No:45

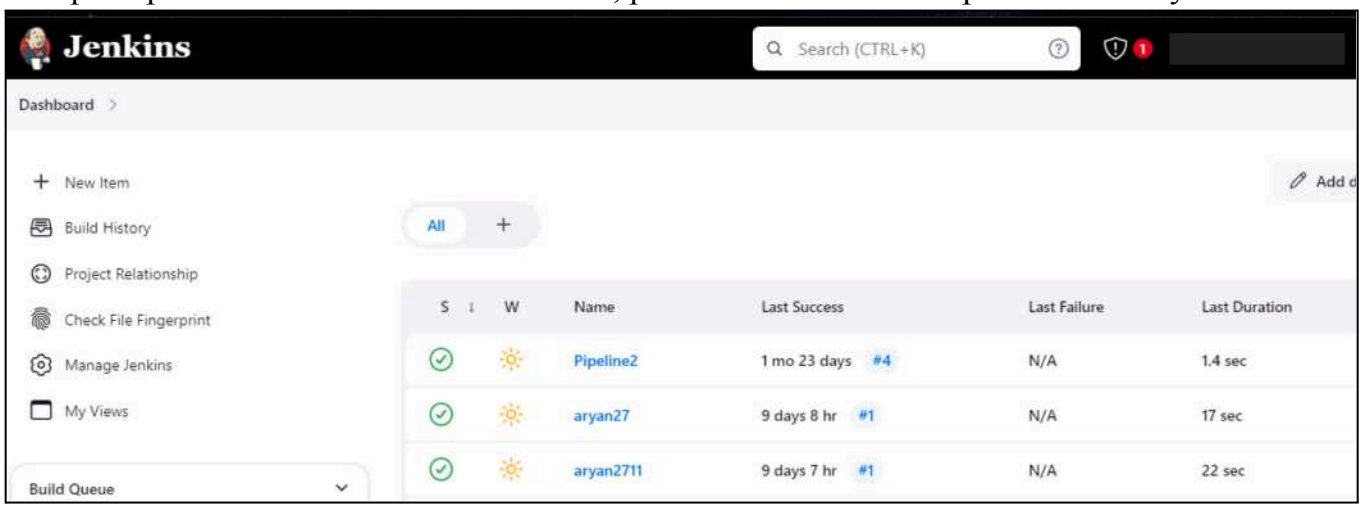
Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

Integrating Jenkins with SonarQube:

- Jenkins installed
- Docker Installed (for SonarQube)
- SonarQube Docker Image

Steps to integrate Jenkins with SonarQube

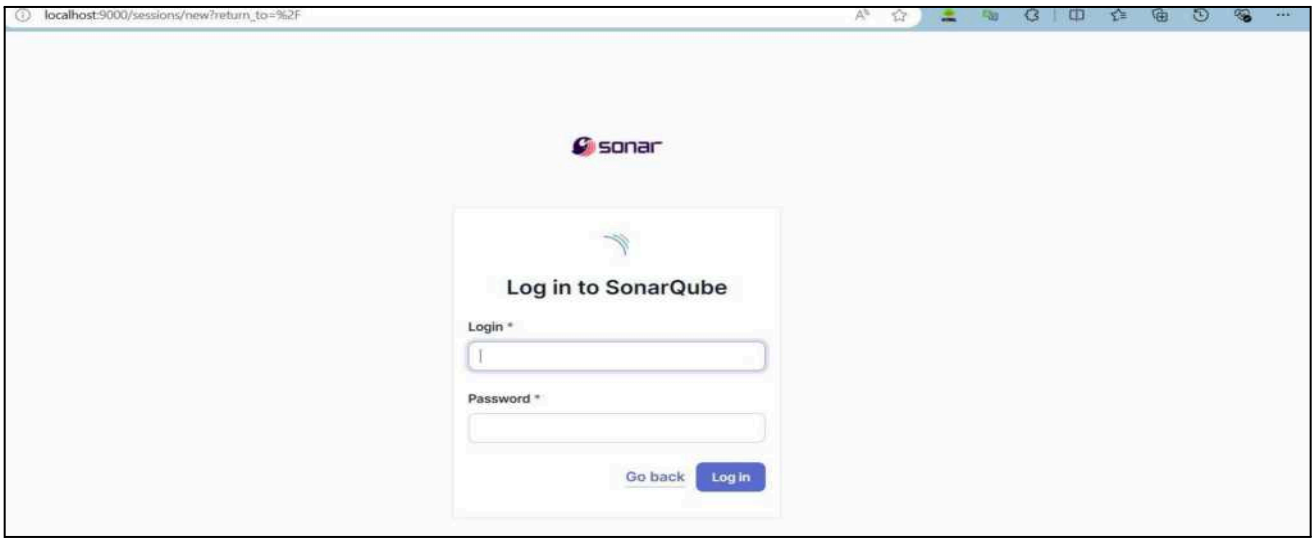
1. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.



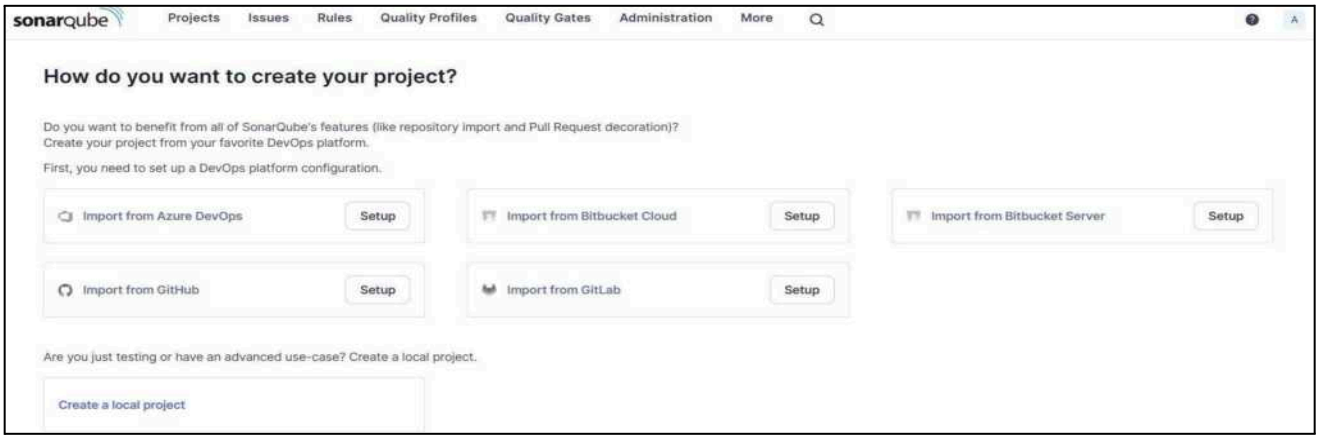
2. Run SonarQube in a Docker container using this command -
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

```
PS C:\Windows\system32> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
5ab3928e5e27607e3661d129731e4e600a9019574c7dc2767aa9b3bfdaa941be
```

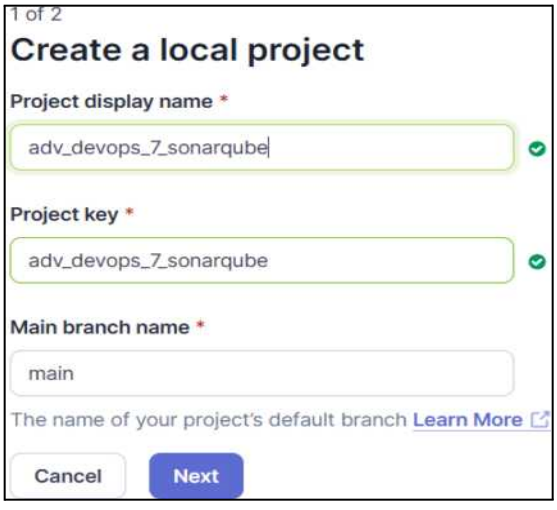
3. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.

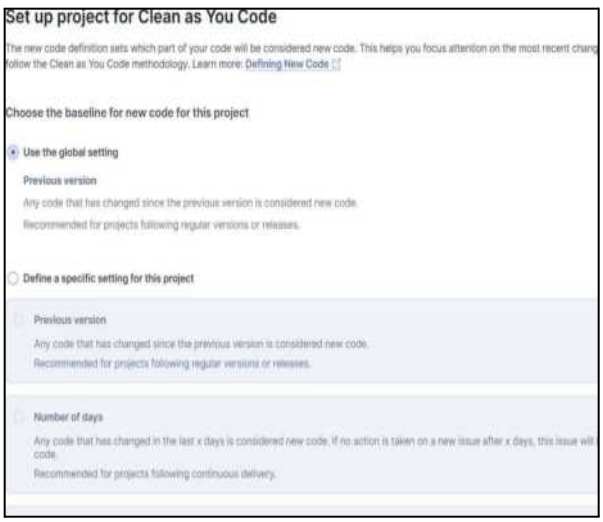


4. Login to SonarQube using username admin and password admin.

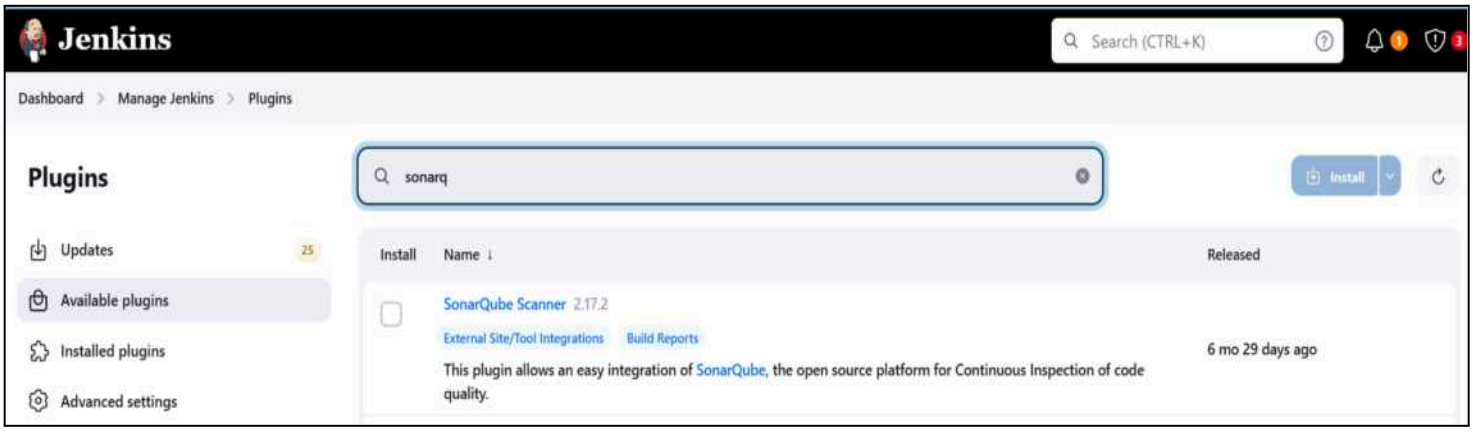


5. Create a manual project in SonarQube with the name sonarqube





Setup the project and come back to Jenkins Dashboard.
Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.



6. Under Jenkins ‘Manage Jenkins’ then go to ‘system’, scroll and look for **SonarQube Servers**
and enter the details.
Enter the Server Authentication token if needed.

In SonarQube installations: Under **Name** add <project name of sonarqube>,here we have named it as **adv_devops_7_sonarqube**
In **Server URL** Default is **http://localhost:9000**


SonarQube servers
If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.
Environment variables
SonarQube installations
List of SonarQube installations
Name
adv_devops_7_sonarqube
Server URL
Default is http://localhost:9000
https://localhost:9000
Server authentication token
SonarQube authentication token. Mandatory when anonymous access is disabled.
- none -
+ Add
Advanced

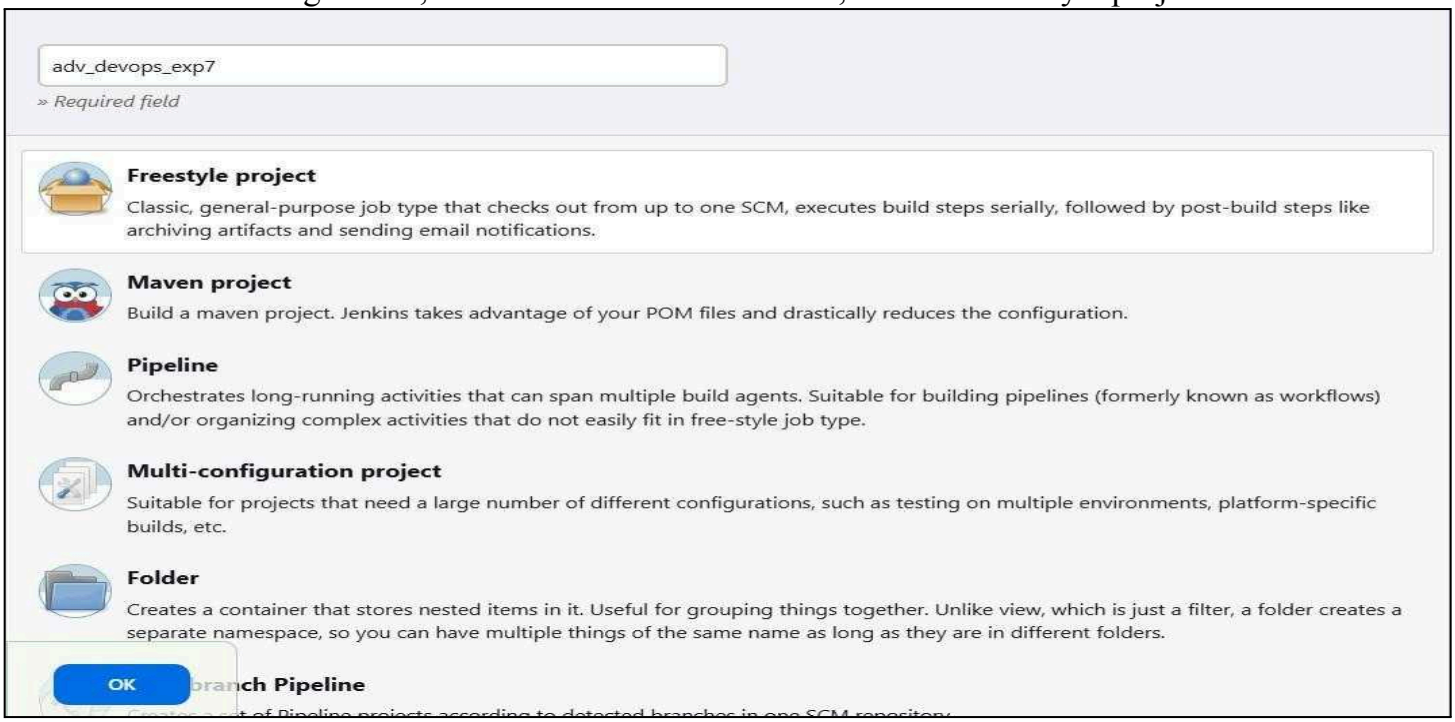
7. Search for SonarQube Scanner under Global Tool Configuration.

Choose the latest configuration and choose Install automatically.

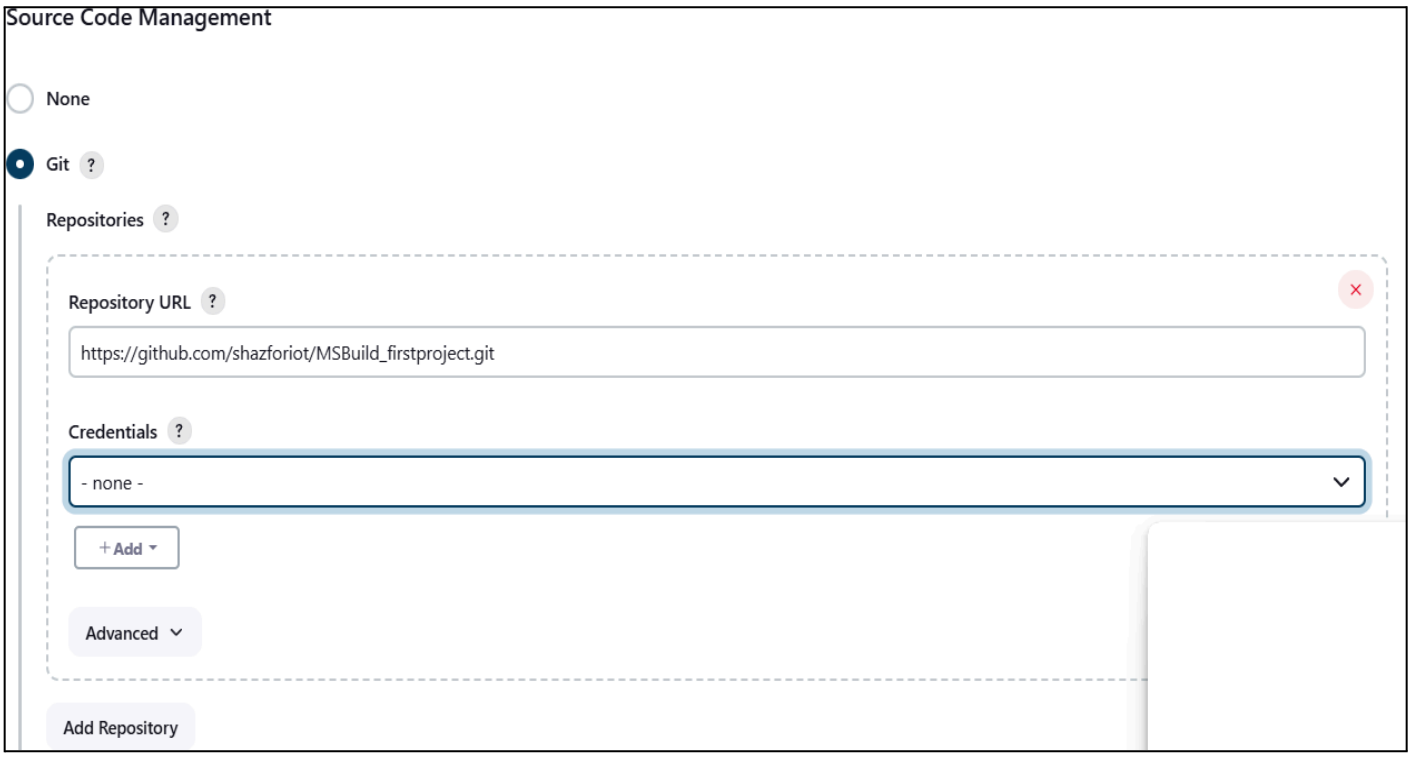
Dashboard > Manage Jenkins > Tools

Check the “Install automatically” option. → Under name any name as identifier → Check the “Install automatically” option.

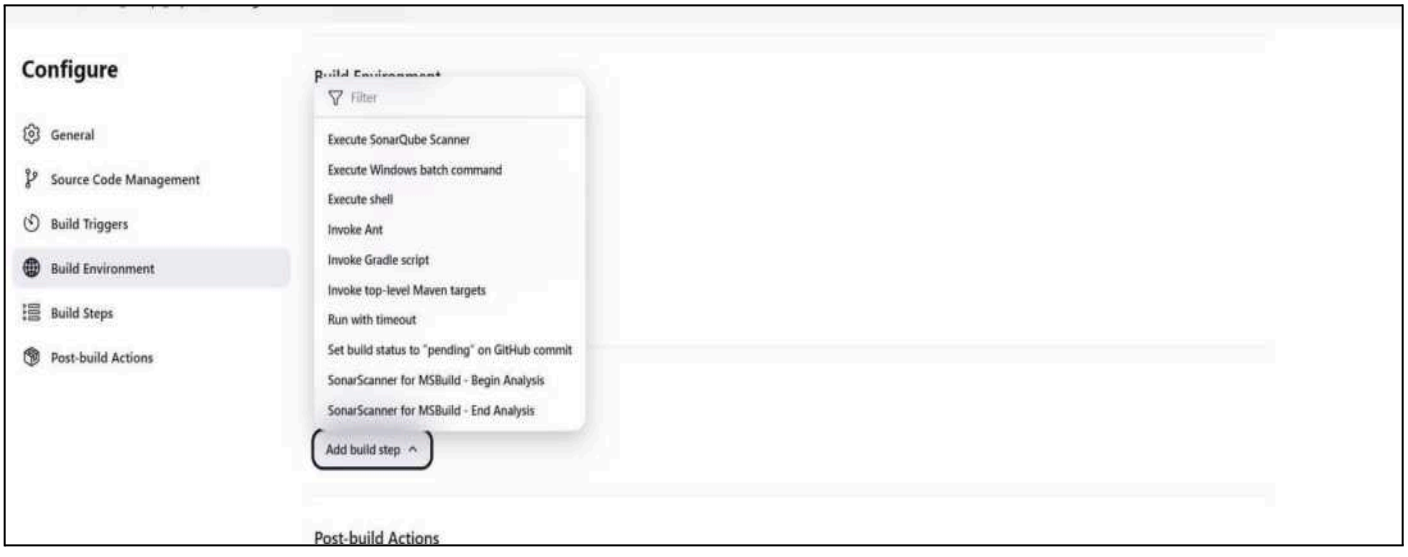
8. After the configuration, create a New Item in Jenkins, choose a freestyle project.



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

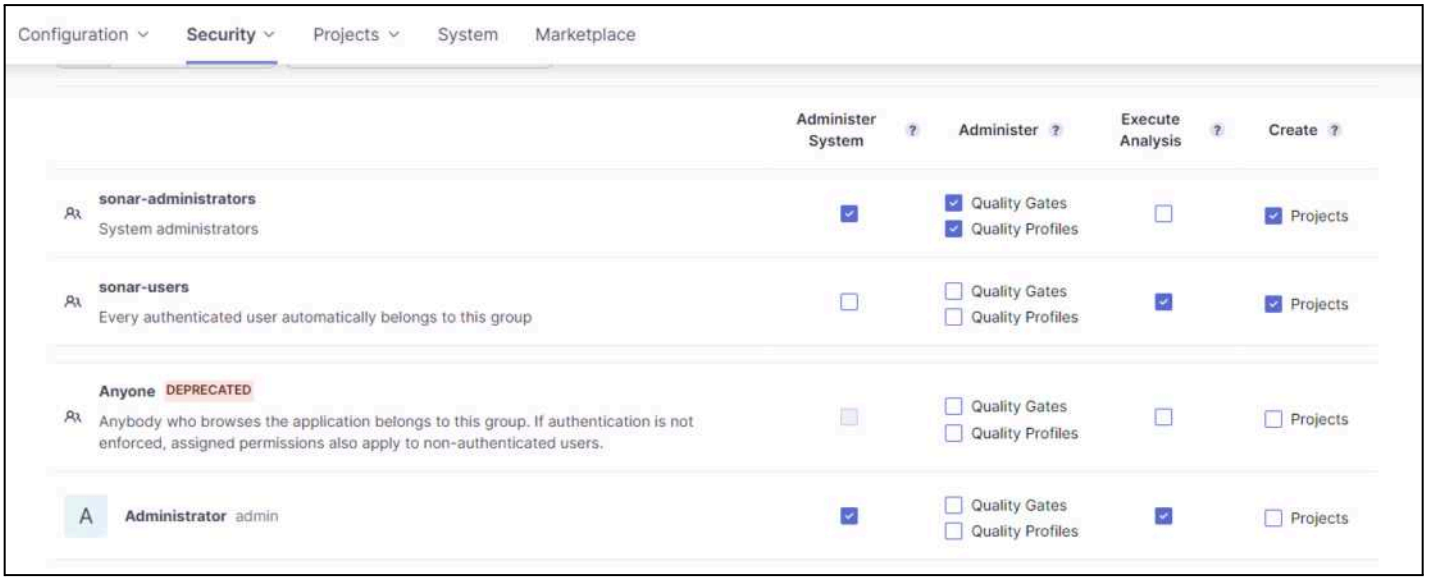


10. Under **Select project → Configuration → Build steps → Execute SonarQube Scanner**, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.



Then save

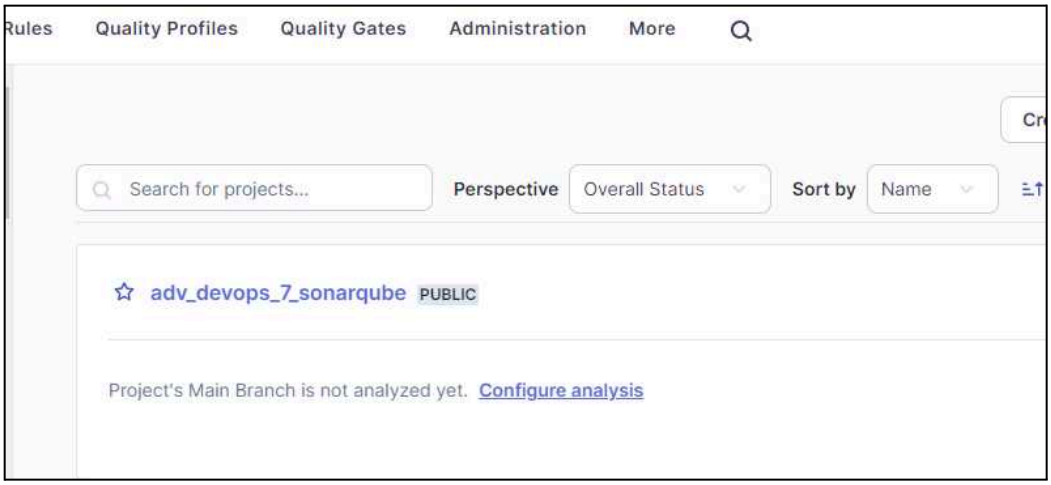
11. Go to http://localhost:9000/<user_name>/permissions and allow Execute Permissions to the Admin user



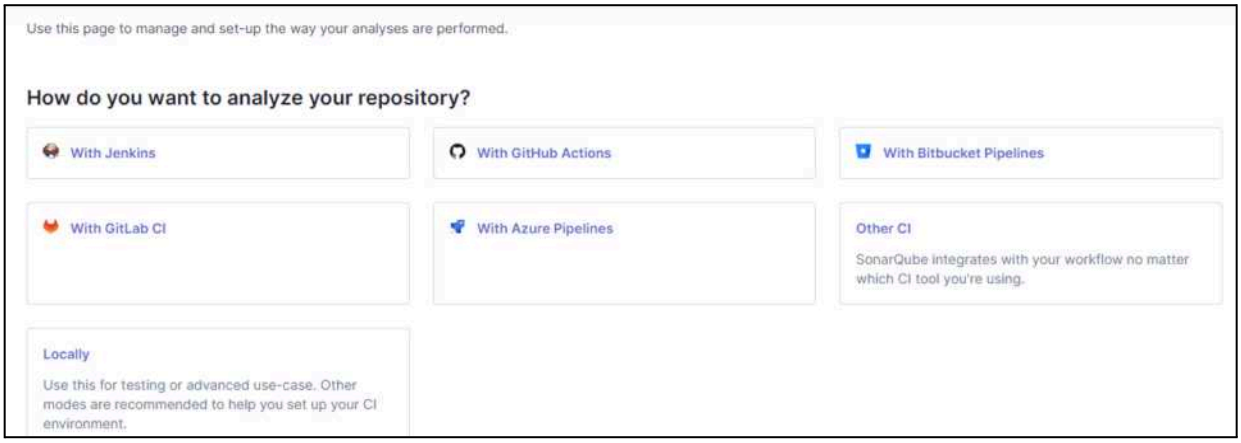
IF CONSOLE OUTPUT FAILED:

Step 1: Generate a New Authentication Token in SonarQube

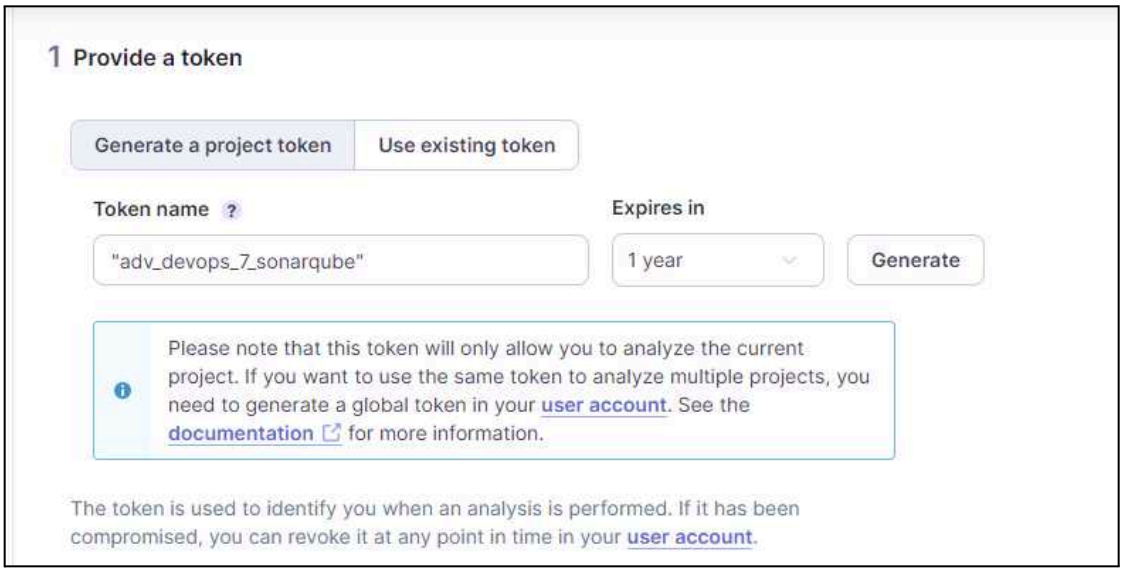
1. **Login to SonarQube:**
    - Open your browser and go to **http://localhost:9000**.
    - Log in with your admin credentials (default username is **admin**, and the password is either **admin** or your custom password if it was changed).
2. **Generate a New Token:**
    - Go to the project that you have created on SonarQube.



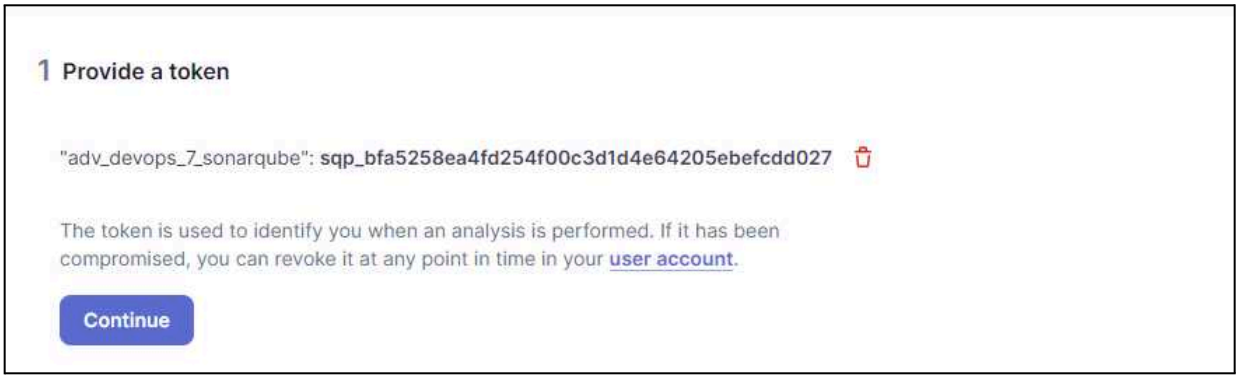
    - Click on **Locally**



    - Further,Generate a Project token with the following details and click on generate.

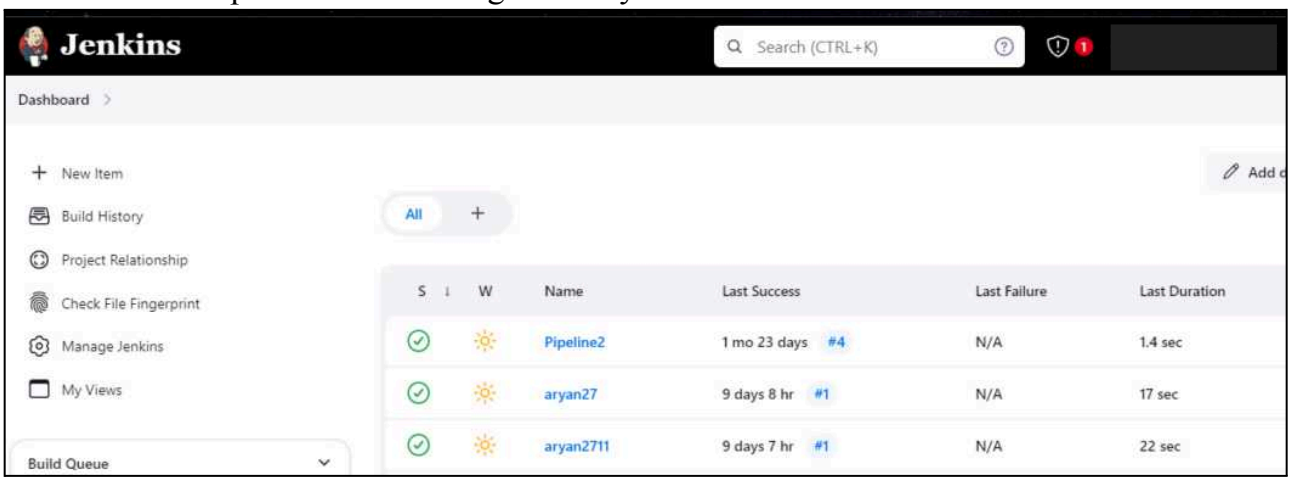


- Copy the token you get here and save it securely as we would need it in Jenkins.



Step 2: Update the Token in Jenkins

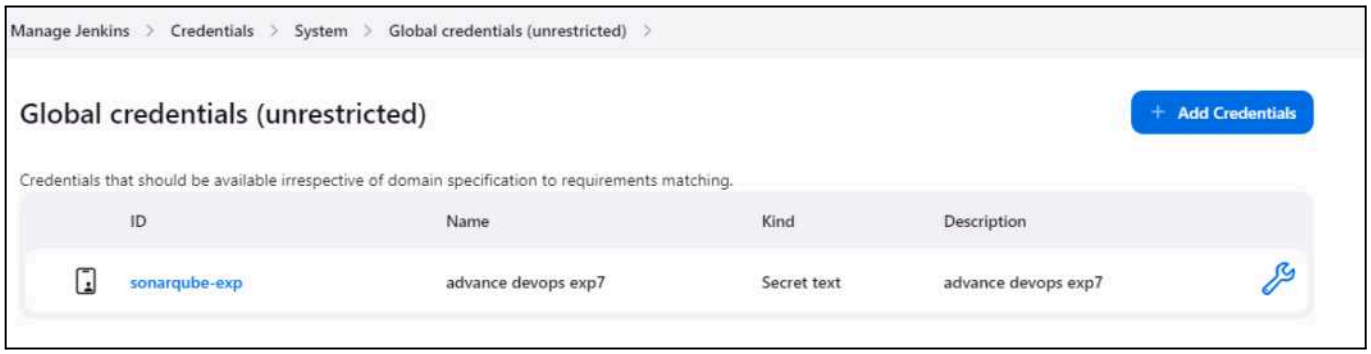
1. **Go to Jenkins Dashboard:**
   - Open Jenkins and log in with your credentials.



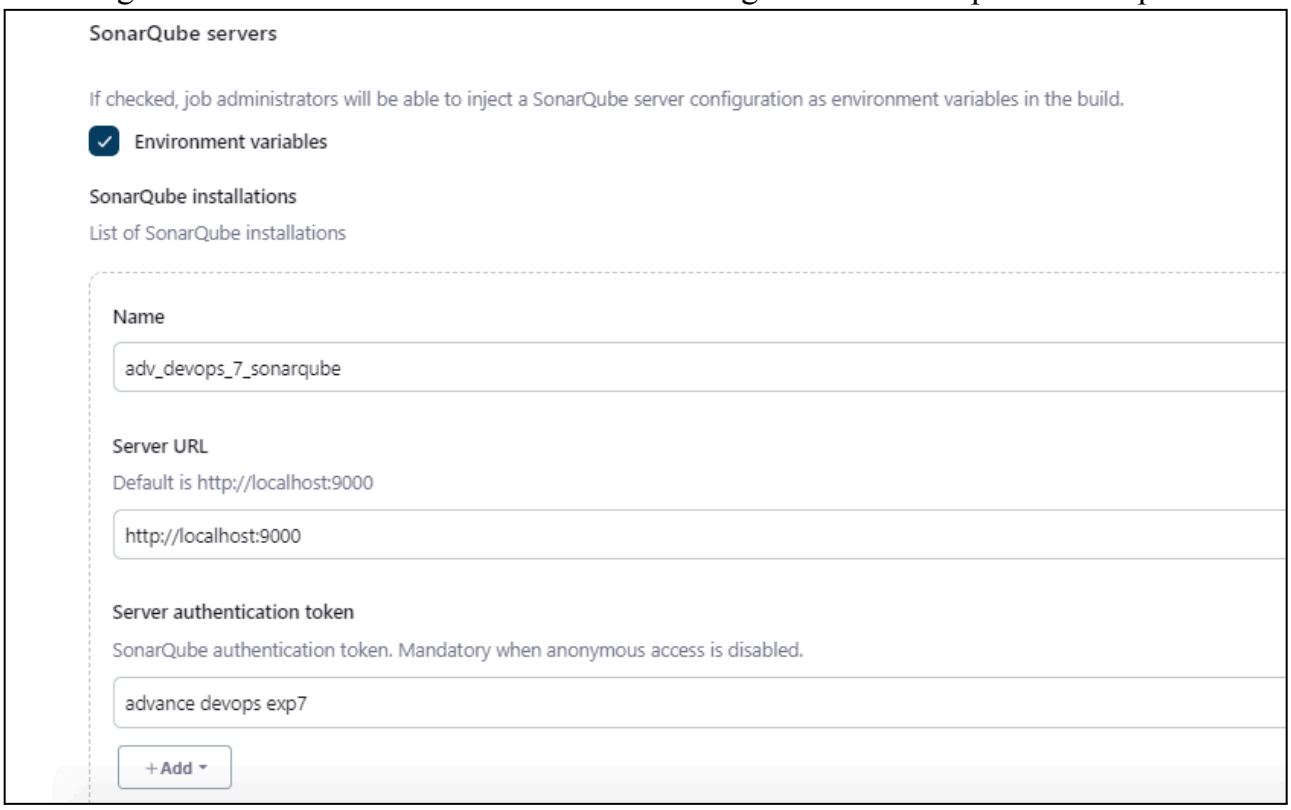
2. Go to Dashboard—>Manage Jenkins—>Credentials

3. Click on **global** under the domains part of Stores scoped to Jenkins section.Further click on add credentials.Proceed with the following details.Make sure to copy the token generated earlier in sonarqube and give any suitable name as the ID.

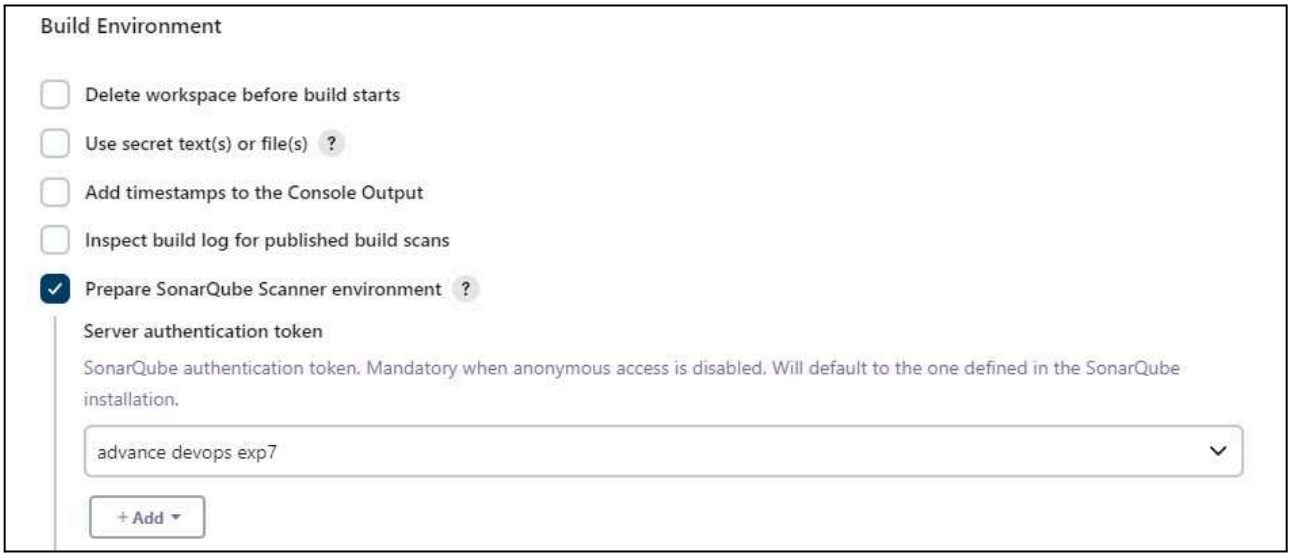
4.After clicking on create we see that the given token has been added in Jenkins credentials.

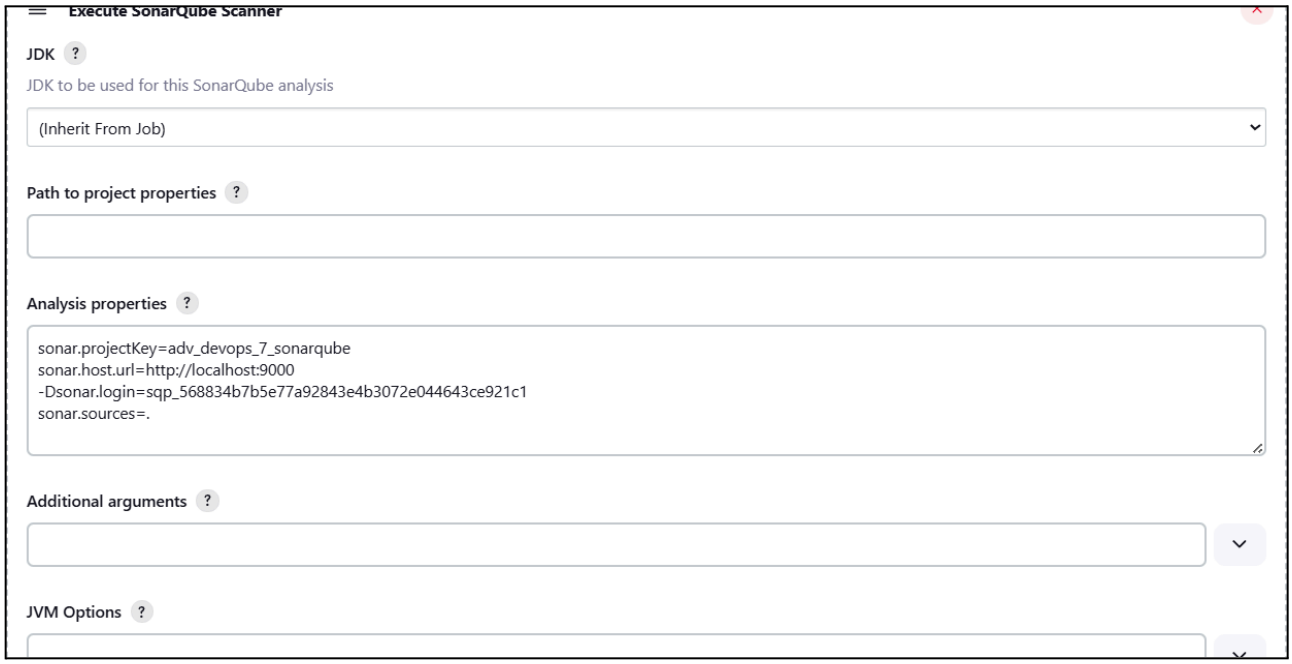


5.Now go to **Manage Jenkins—>System—>SonarQube servers** and proceed with the following details. Reference the authentication token generated in the previous step.

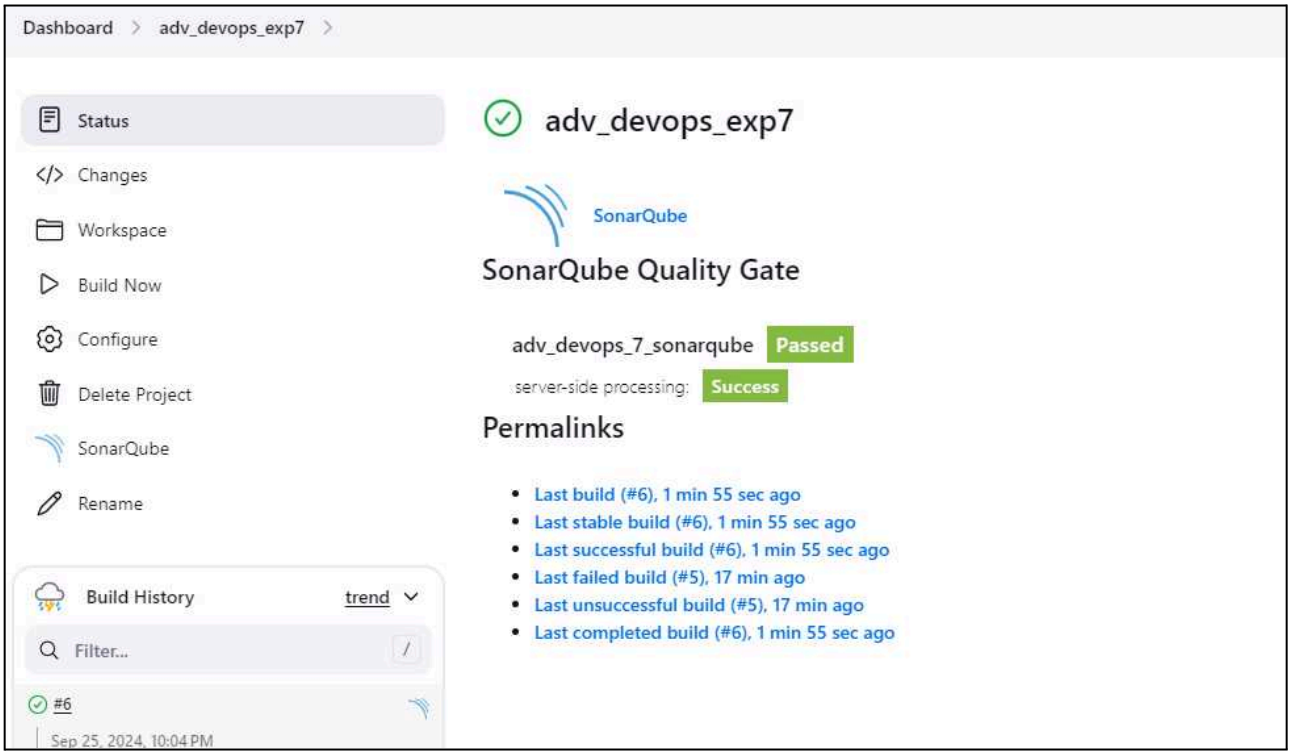


6.Check the SonarQube Scanner Environment and add the server authentication token

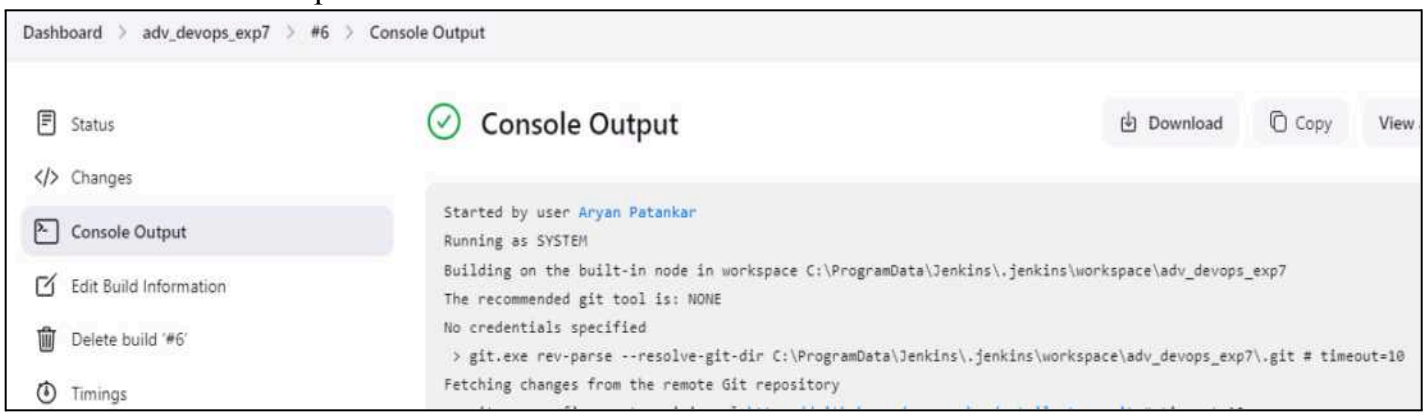




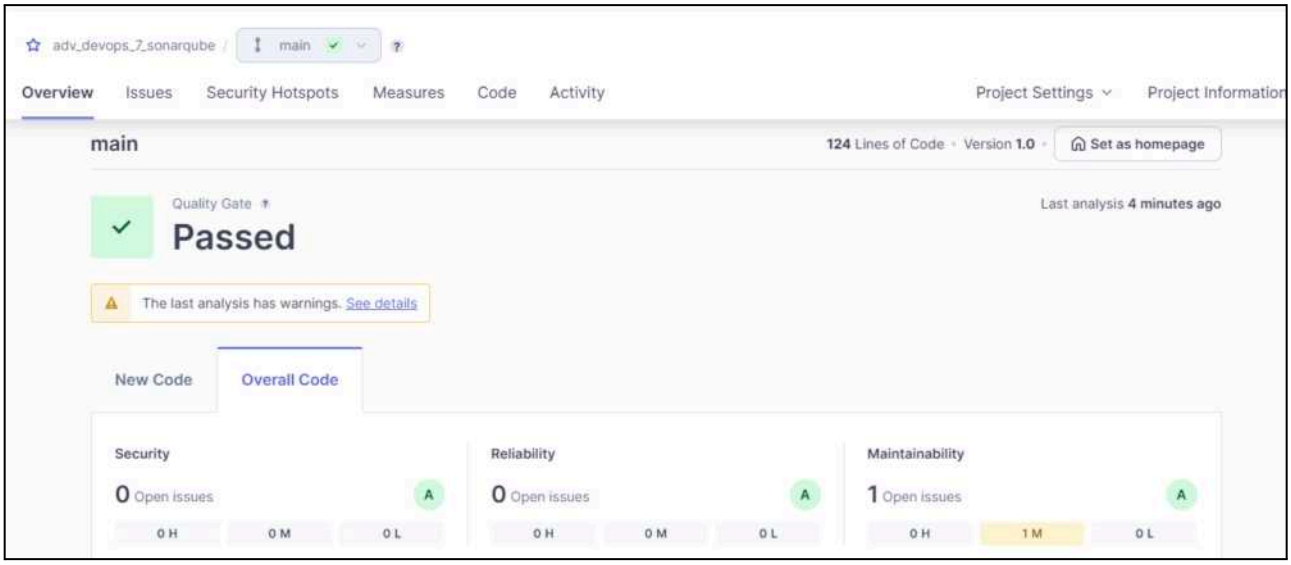
12.Run the Jenkins build.



Check the console Output



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



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

ADVANCE DEVOPS EXP 8

Name:Manav punjabi
Class:D15A
Roll No:45

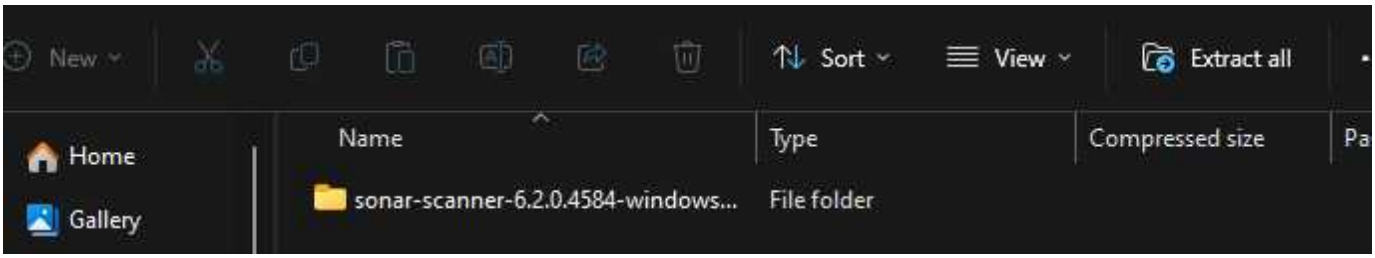
Aim: Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web / Java / Python application.

Step 1: Download sonar scanner

https://docs.sonarsource.com/sonarqube/latest/analyzing-source-code/scanners/sonarscanner/

ner/ Visit this link and download the sonarqube scanner CLI.

Extract the downloaded zip file in a folder.

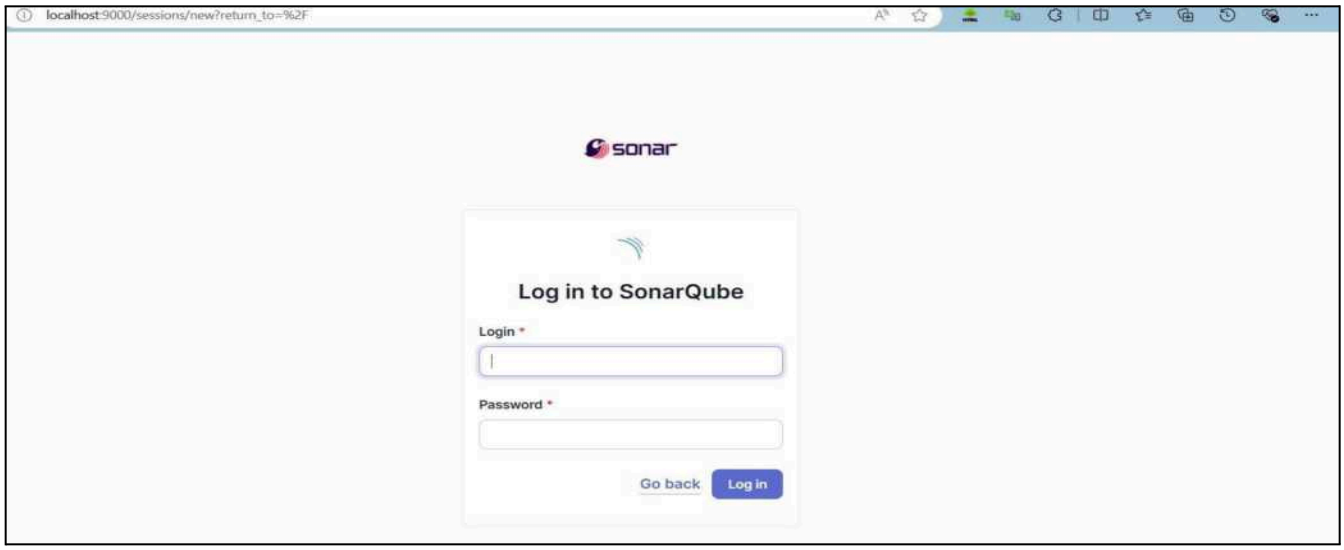


1. Install sonarqube image

Command: **docker pull sonarqube**

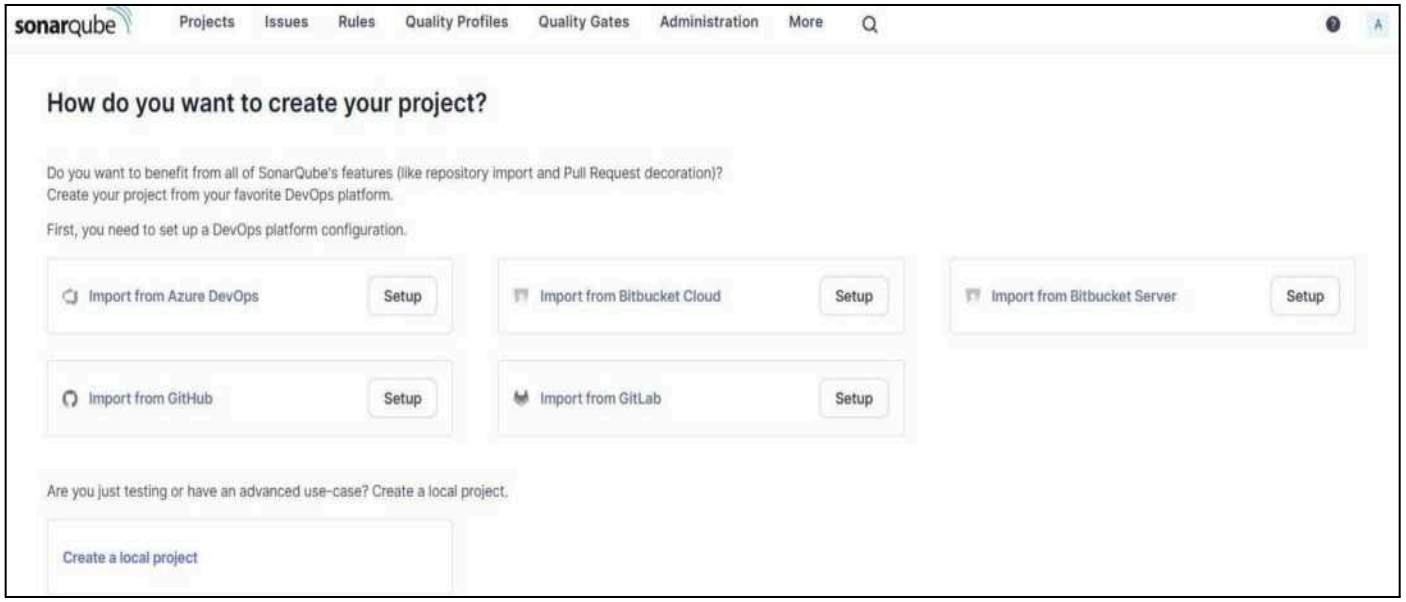
```
C:\Windows\System32>docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Image is up to date for sonarqube:latest
docker.io/library/sonarqube:latest
```

2. Once the container is up and running, you can check the status of

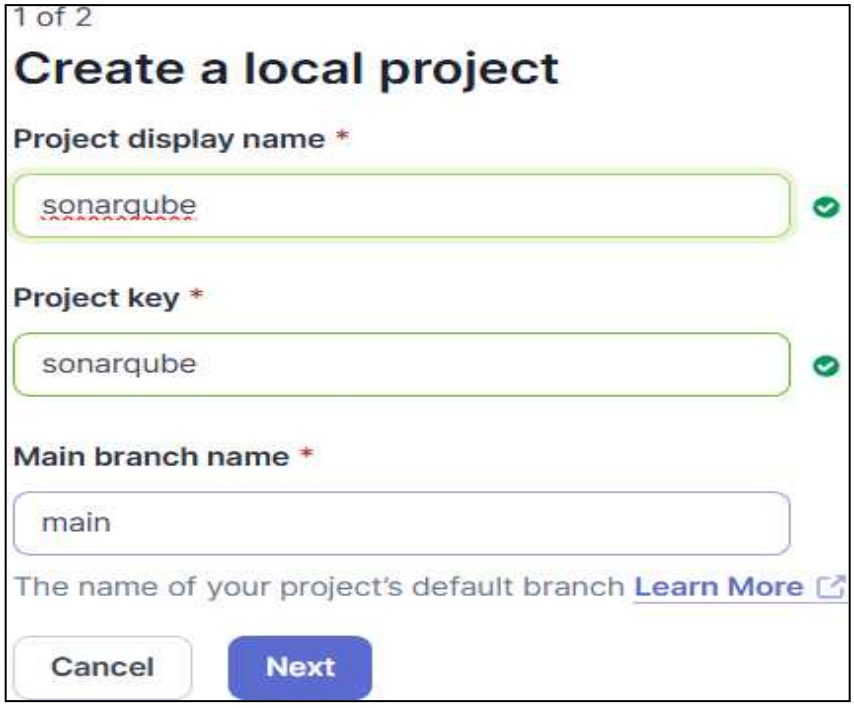


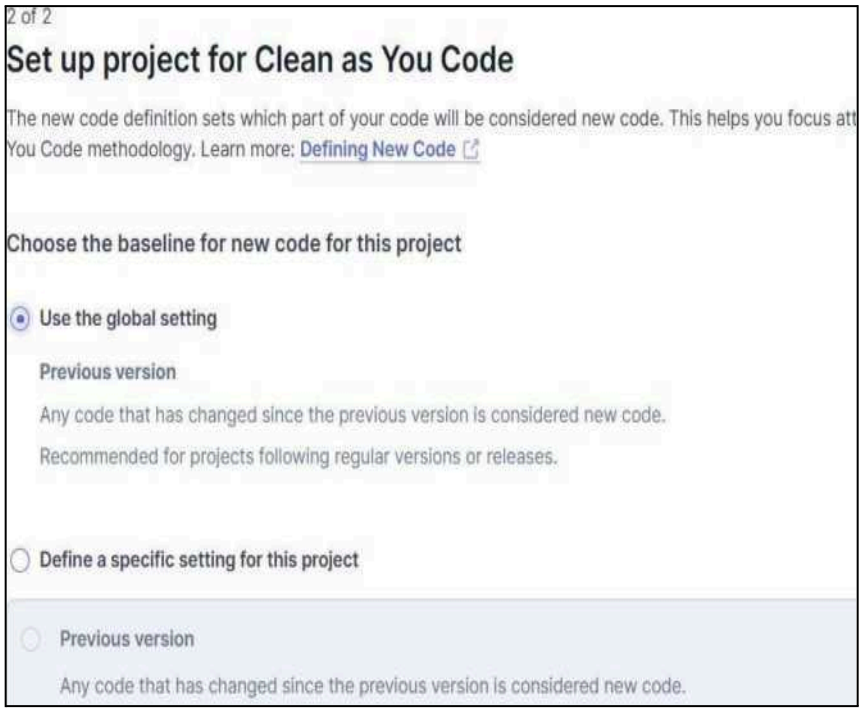
SonarQube at localhost port 9000.

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

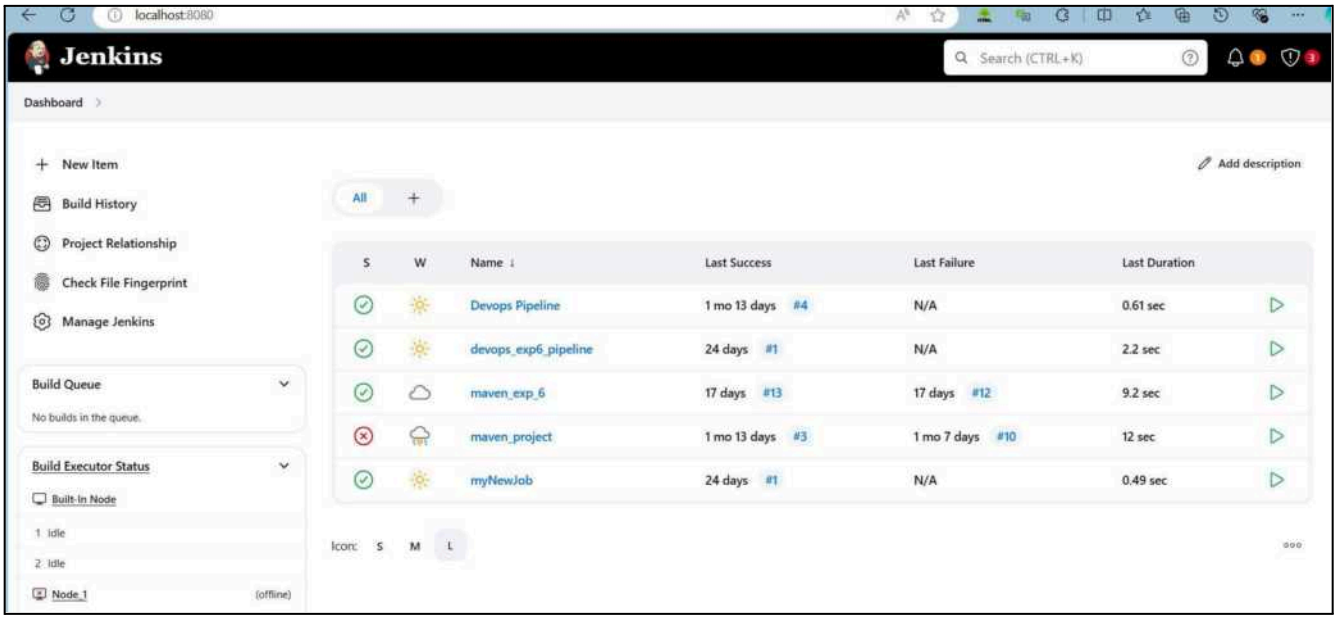


4. Create a manual project in SonarQube with the name sonarqube

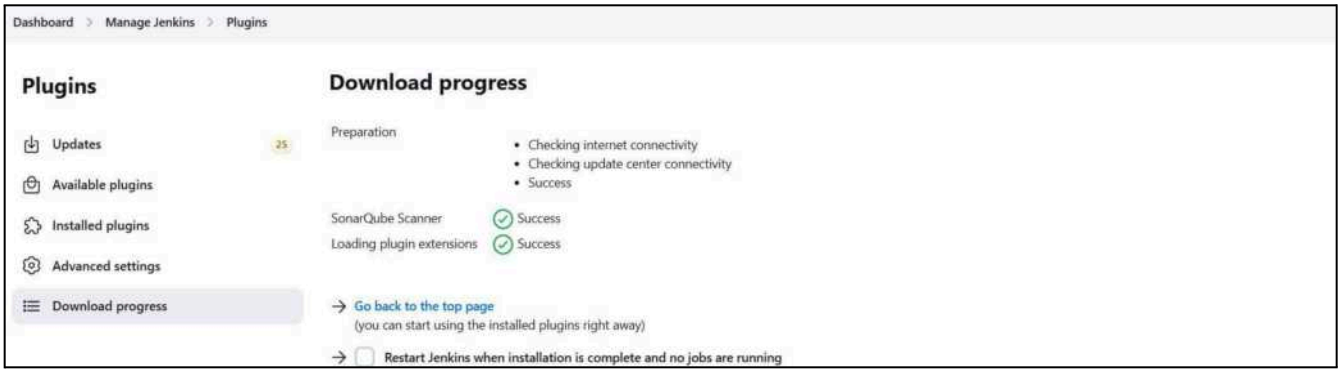




5. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.



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



7. Under Jenkins ‘Manage Jenkins’ then go to ‘system’, scroll and look for **SonarQube Servers**

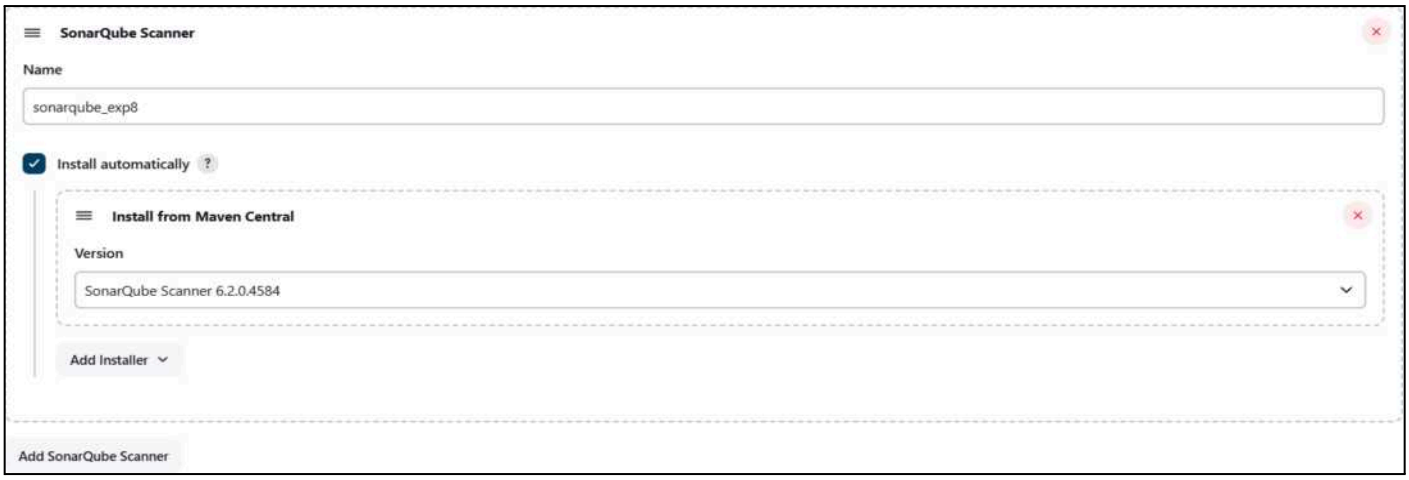
and enter the details.
Enter the Server Authentication token if needed.

In SonarQube installations: Under **Name** add <project name of sonarqube> for me **adv_devops_7_sonarqube**
In **Server URL** Default is **http://localhost:9000**

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

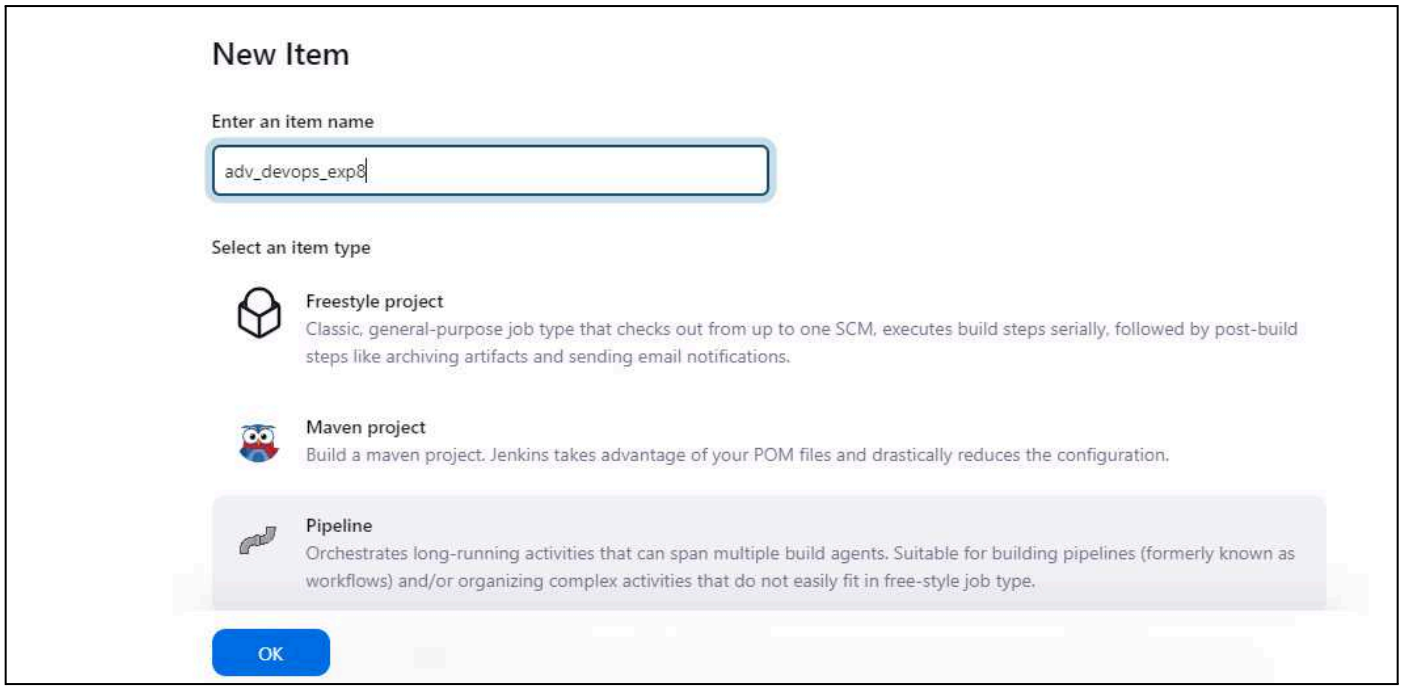
Dashboard > Manage Jenkins > Tools

Check the “Install automatically” option. → Under name any name as identifier → Check



the “Install automatically” option.

9. After configuration, create a New Item → choose a pipeline project.

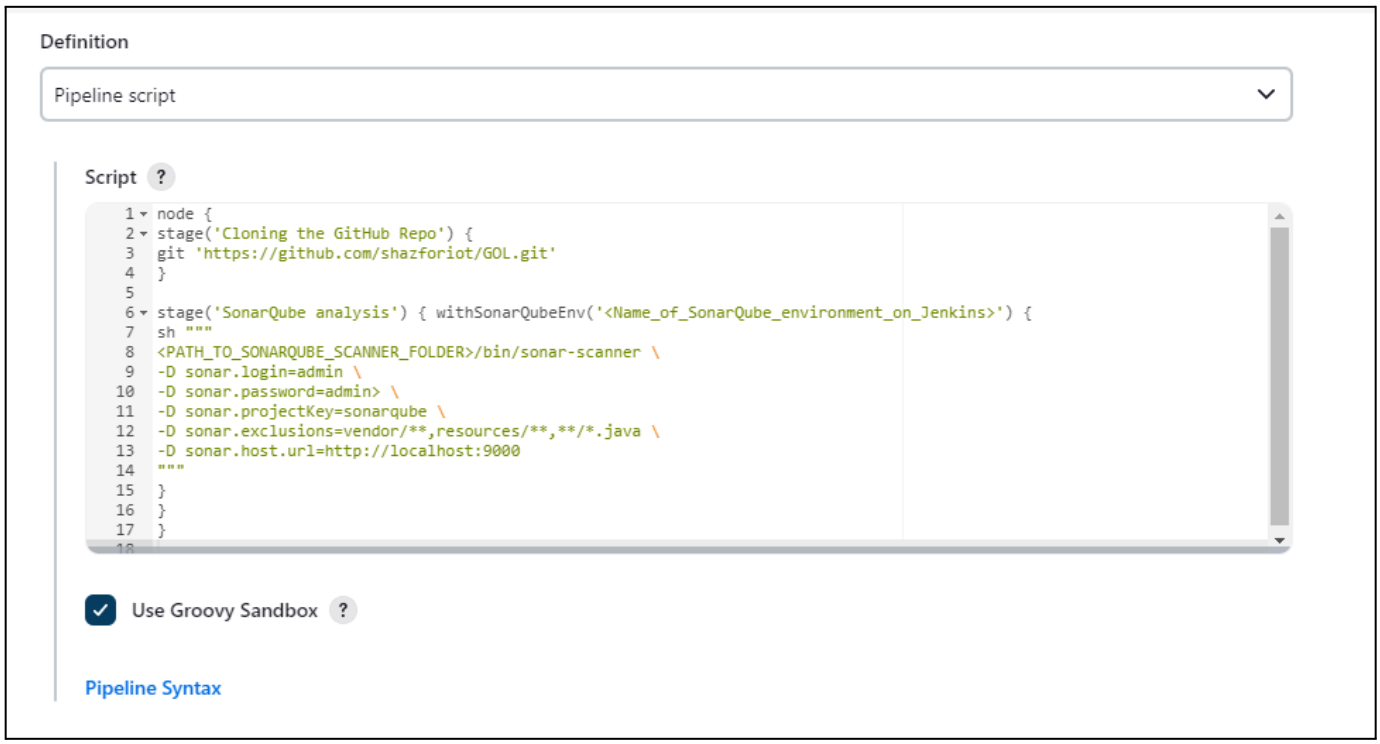


10. Under Pipeline script, enter the following:

```
    node {
stage('Cloning the GitHub Repo') {
  git 'https://github.com/shazforiot/GOL.git'
}

stage('SonarQube analysis') {
  withSonarQubeEnv('<Name_of_SonarQube_environment_on_Jenki
  ns>') {
    sh """
      <PATH_TO_SONARQUBE_SCANNER_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=<SonarQube_URL>(default: http://localhost:9000/)
    """
  }
}
```

}

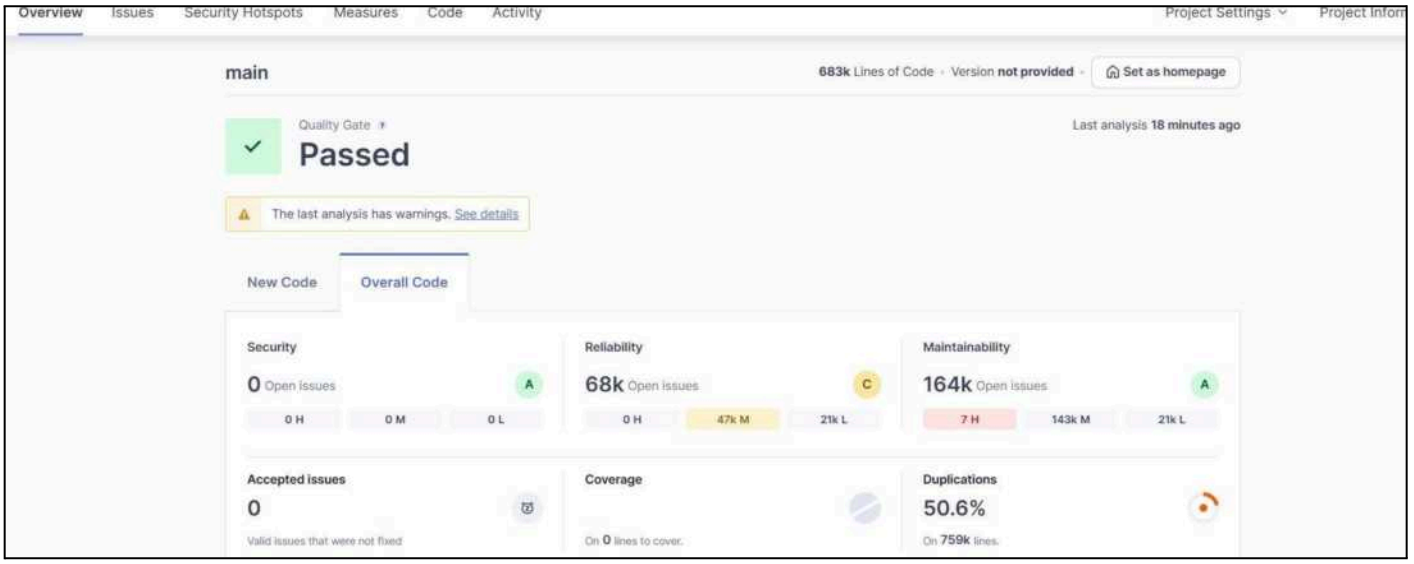
It is a java sample project which has a lot of repetitions and issues that will be detected by SonarQube.



11. Build project

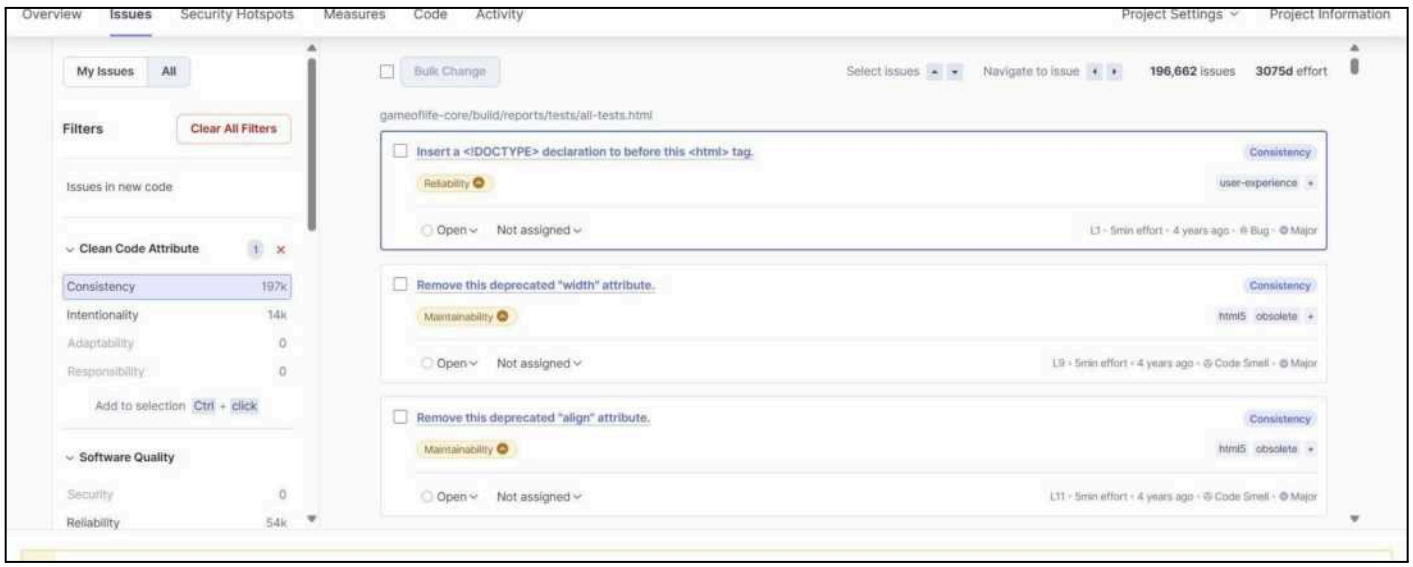
12. Check console

13. Now, check the project in SonarQube

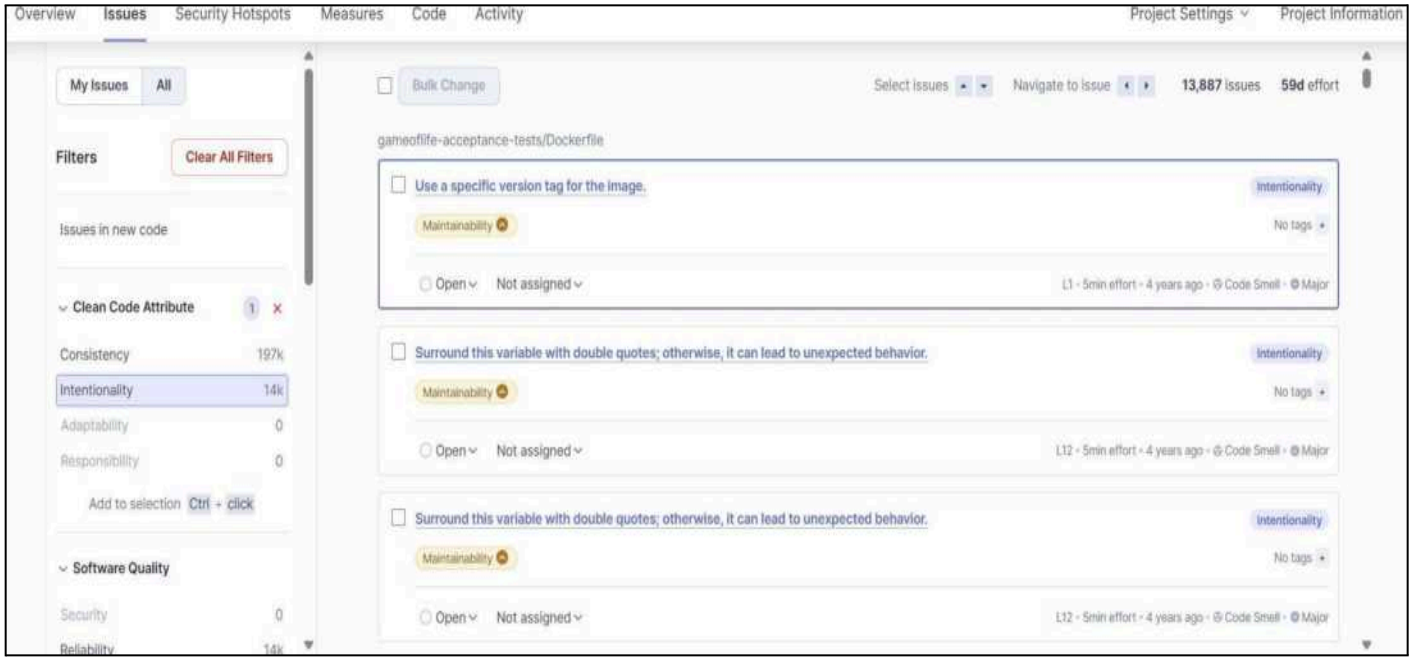


14. Code Problems

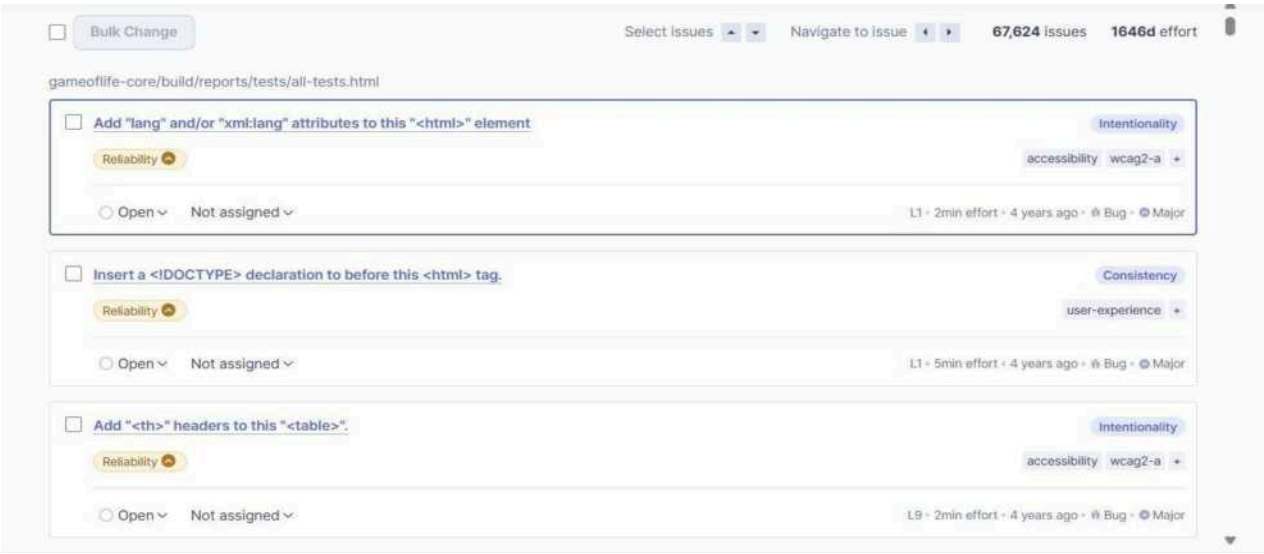
- Consistency



- Intentionality

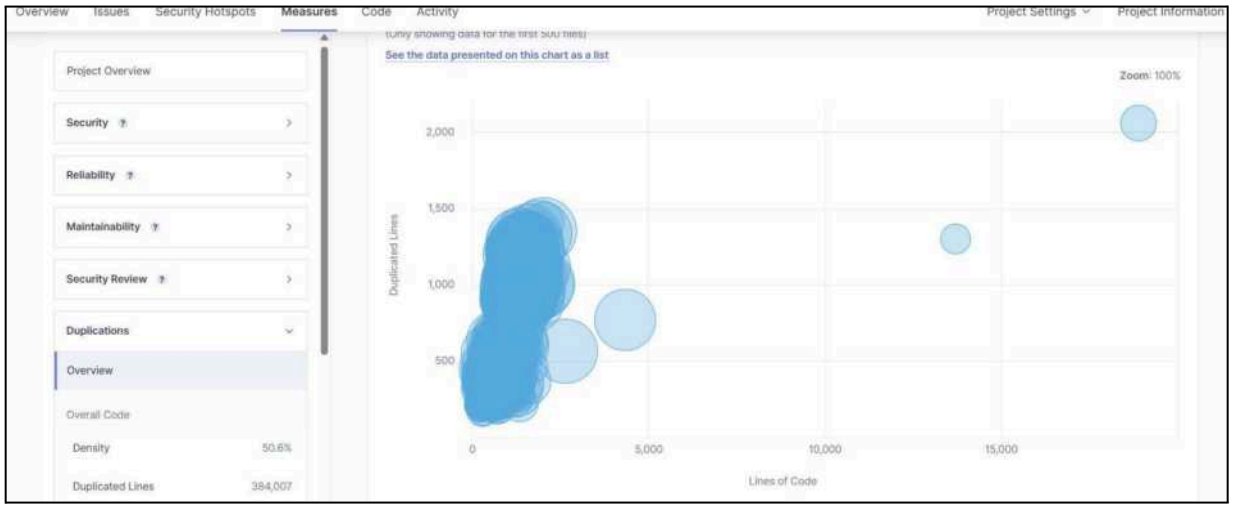


- Bugs



- Code Smells

- Duplications



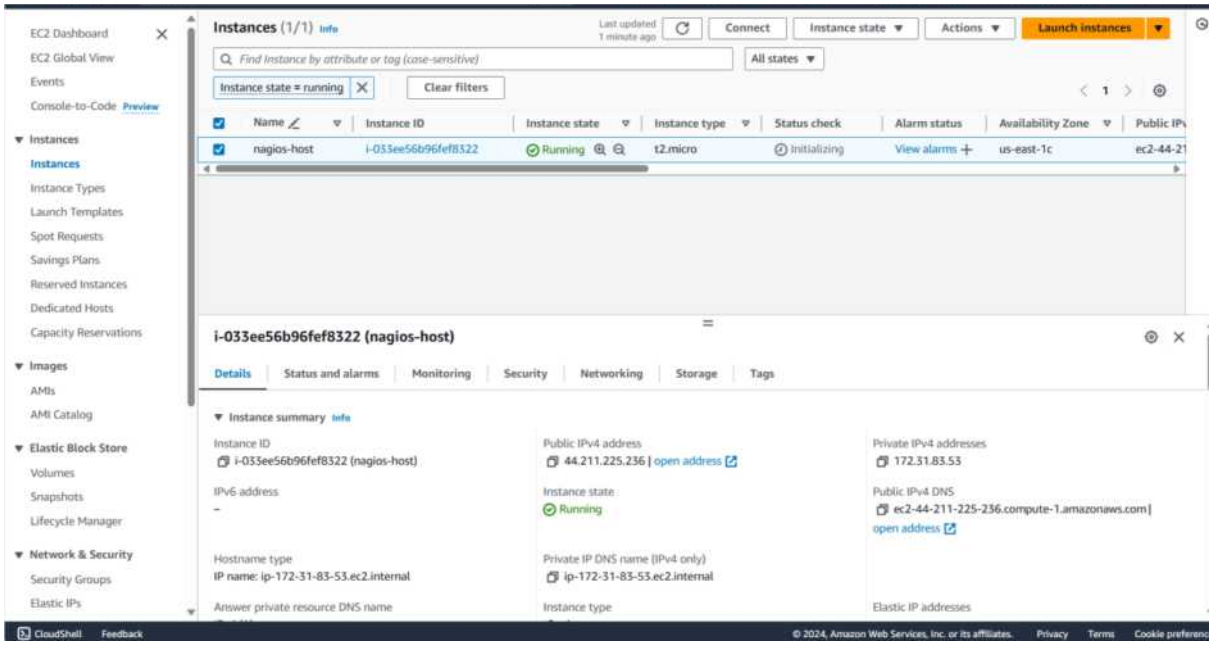
- Cyclomatic Complexities

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

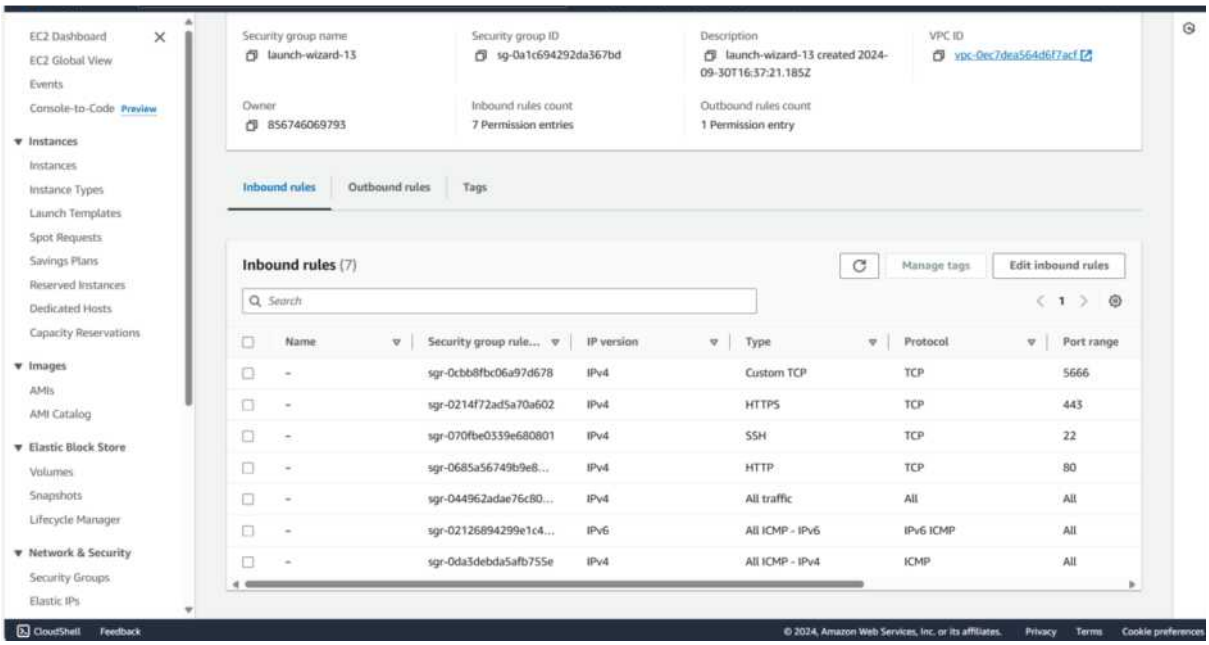
Experiment 9

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

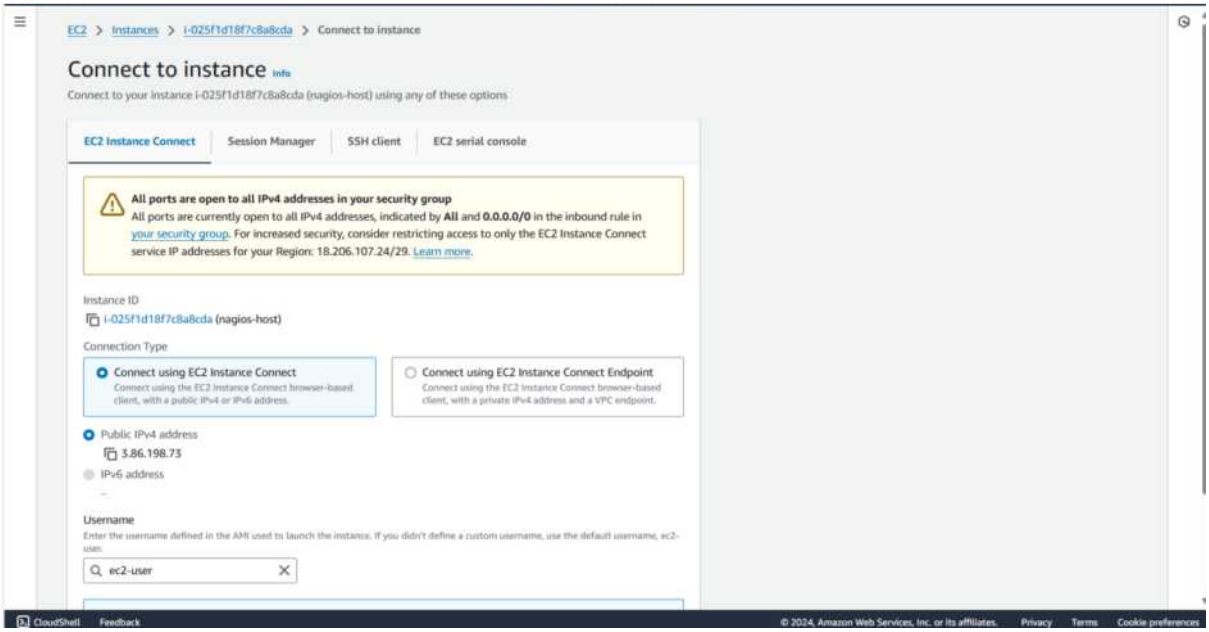
Step 1: Create an EC2 Instance and name it as nagios-host



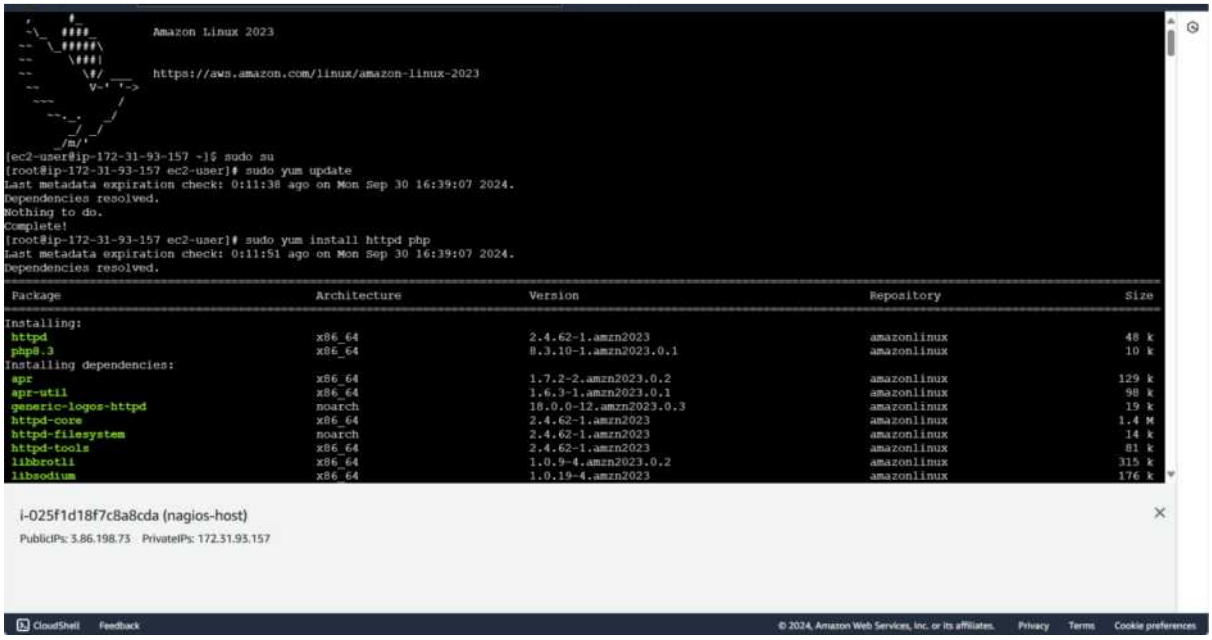
Step 2: Under the security groups, click on edit inbound rules and set as shown in the figure below

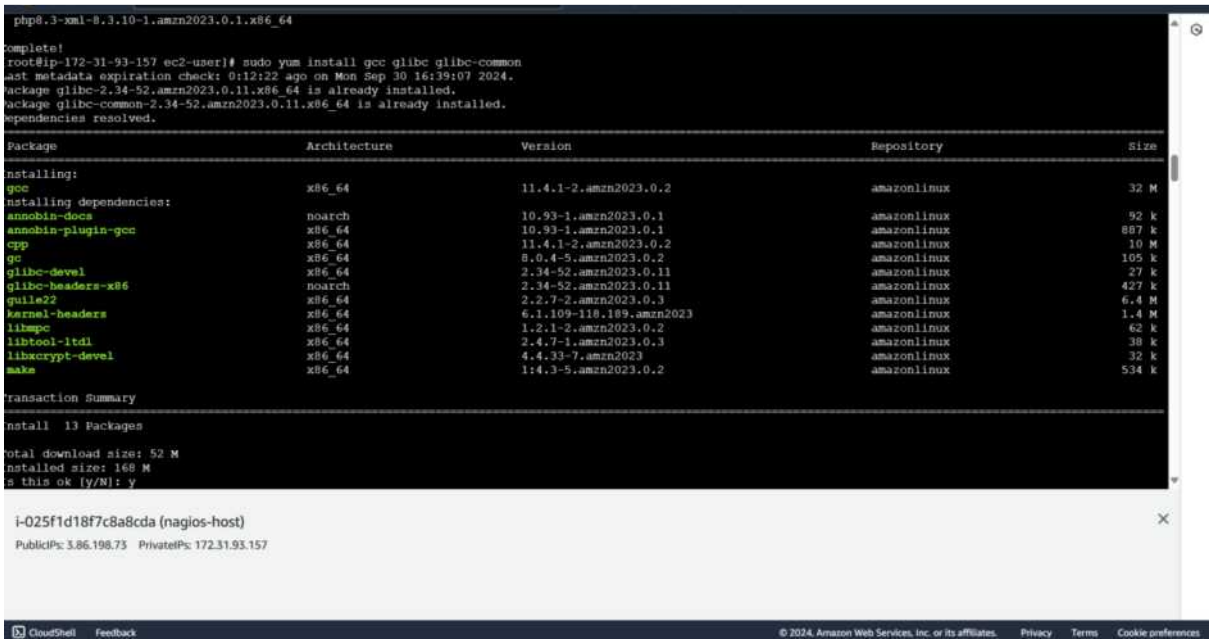


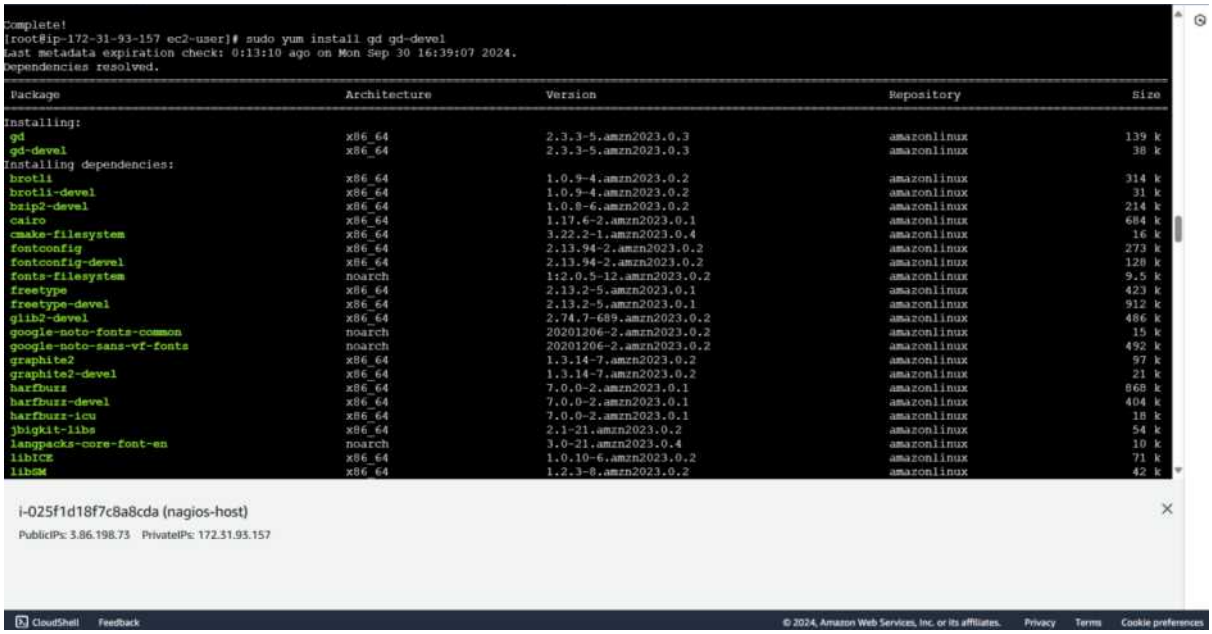
Step 3: Then select the instance nagios-host and then connect the instance.



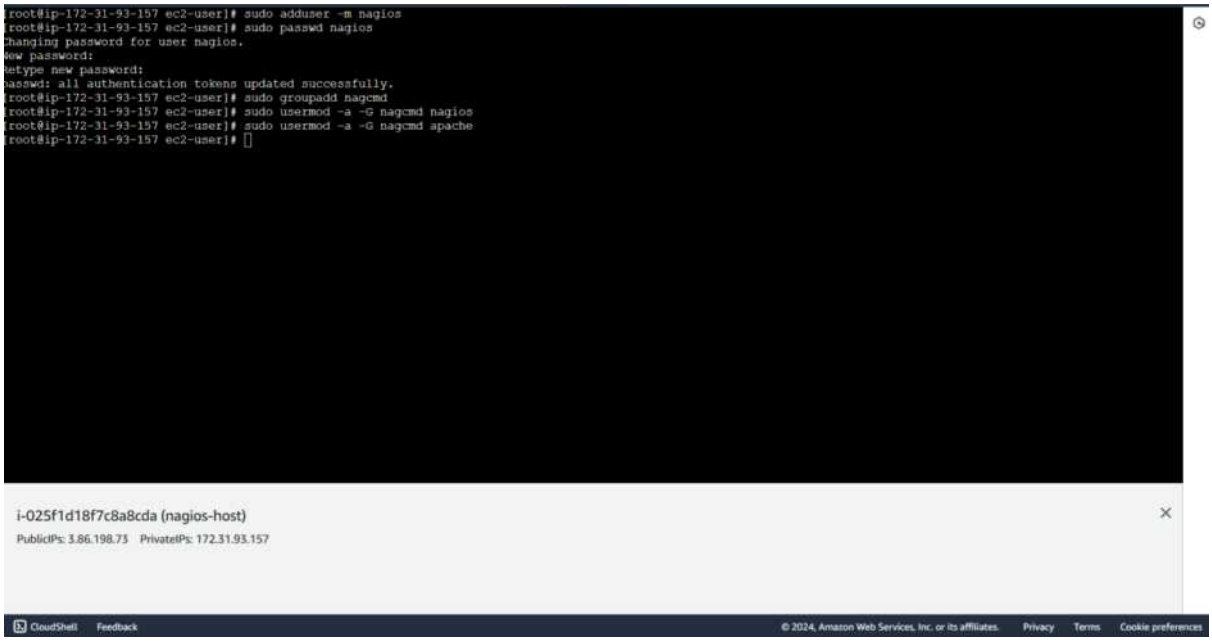
Step 3: Now, run the following commands -
sudo su
sudo yum update
sudo yum install httpd php
sudo yum install gcc glibc glibc-common
sudo yum install gd gd-devel



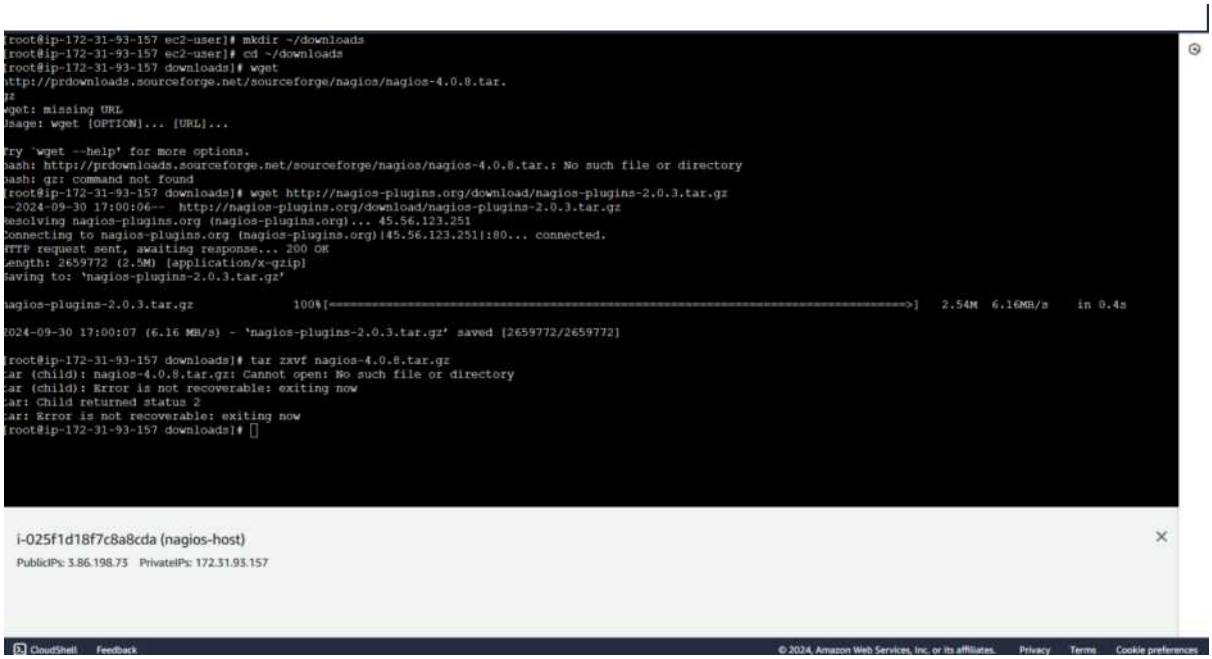




Step 4: Create a new nagios user with its password.
sudo adduser -m nagios
sudo passwd nagios
sudo groupadd nagcmd
sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache



Step 5: Now, run the following commands -
mkdir ~/downloads
cd ~/downloads
wget http://prdownloads.sourceforge.net/sourceforge/nagios/nagios-4.0.8.tar.gz
wget http://nagios-plugins.org/download/nagios-plugins-2.0.3.tar.gz
tar zxvf nagios-4.0.8.tar.gz



To resolve the error run the following commands -
wget http://prdownloads.sourceforge.net/sourceforge/nagios/nagios-4.0.8.tar.gz
wget http://nagios-plugins.org/download/nagios-plugins-2.0.3.tar.gz
tar zxvf nagios-4.0.8.tar.gz
tar zxvf nagios-plugins-2.0.3.tar.gz
cd nagios-4.0.8

```
[root@ip-172-31-93-157 downloads]# wget http://prdownloads.sourceforge.net/sourceforge/nagios/nagios-4.0.8.tar.gz
wget http://nagios-plugins.org/download/nagios-plugins-2.0.3.tar.gz
--2024-09-30 17:03:04--  http://prdownloads.sourceforge.net/sourceforge/nagios/nagios-4.0.8.tar.gz
Resolving prdownloads.sourceforge.net (prdownloads.sourceforge.net)... 204.68.111.105
Connecting to prdownloads.sourceforge.net (prdownloads.sourceforge.net)|204.68.111.105|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: http://downloads.sourceforge.net/project/nagios/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz [following]
--2024-09-30 17:03:04--  http://downloads.sourceforge.net/project/nagios/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz
Resolving downloads.sourceforge.net (downloads.sourceforge.net)... 204.68.111.105
Reusing existing connection to prdownloads.sourceforge.net:80.
HTTP request sent, awaiting response... 302 Found
Location: http://versaweb.dl.sourceforge.net/project/nagios/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz?viasf=1 [following]
--2024-09-30 17:03:04--  http://versaweb.dl.sourceforge.net/project/nagios/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz?viasf=1
Resolving versaweb.dl.sourceforge.net (versaweb.dl.sourceforge.net)... 162.251.232.173
Connecting to versaweb.dl.sourceforge.net (versaweb.dl.sourceforge.net)|162.251.232.173|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1805059 (1.7M) [application/x-gzip]
Saving to: 'nagios-4.0.8.tar.gz'

nagios-4.0.8.tar.gz                 100%[=============================================================================>]   1.72M  2.21MB/s    in 0.8s

2024-09-30 17:03:05 (2.21 MB/s) - 'nagios-4.0.8.tar.gz' saved [1805059/1805059]

--2024-09-30 17:03:05--  http://nagios-plugins.org/download/nagios-plugins-2.0.3.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2659772 (2.5M) [application/x-gzip]
Saving to: 'nagios-plugins-2.0.3.tar.gz.1'

nagios-plugins-2.0.3.tar.gz.1           100%[=============================================================================>]   2.54M  7.26MB/s    in 0.3s

2024-09-30 17:03:05 (7.26 MB/s) - 'nagios-plugins-2.0.3.tar.gz.1' saved [2659772/2659772]
```

i-025f1d18f7c8a8cda (nagios-host)

PublicIPs: 3.86.198.73 PrivateIPs: 172.31.93.157

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

```
nagios-plugins-2.0.3/plugins-scripts/check_ifoperstatus.pl
nagios-plugins-2.0.3/plugins-scripts/Makefile.am
nagios-plugins-2.0.3/plugins-scripts/subst.in
nagios-plugins-2.0.3/plugins-scripts/check_breeze.pl
nagios-plugins-2.0.3/plugins-scripts/check_log.sh
nagios-plugins-2.0.3/plugins-scripts/check_flexlm.pl
nagios-plugins-2.0.3/plugins-scripts/check_rpc.pl
nagios-plugins-2.0.3/plugins-scripts/check_oracle.sh
nagios-plugins-2.0.3/plugins-scripts/utils.pm.in
nagios-plugins-2.0.3/plugins-scripts/check_disk_smb.pl
nagios-plugins-2.0.3/plugins-scripts/t/
nagios-plugins-2.0.3/plugins-scripts/t/check_ifoperstatus.t
nagios-plugins-2.0.3/plugins-scripts/t/check_rpc.t
nagios-plugins-2.0.3/plugins-scripts/t/check_file_age.t
nagios-plugins-2.0.3/plugins-scripts/t/check_disk_smb.t
nagios-plugins-2.0.3/plugins-scripts/t/check_ifstatus.t
nagios-plugins-2.0.3/plugins-scripts/t/utils.t
nagios-plugins-2.0.3/plugins-scripts/check_mailq.pl
nagios-plugins-2.0.3/plugins-scripts/check_wave.pl
nagios-plugins-2.0.3/plugins-scripts/check_ircd.pl
nagios-plugins-2.0.3/plugins-scripts/utils.sh.in
nagios-plugins-2.0.3/plugins-scripts/check_ifstatus.pl
nagios-plugins-2.0.3/plugins-scripts/check_sensors.sh
nagios-plugins-2.0.3/pkg/
nagios-plugins-2.0.3/pkg/fedora/
nagios-plugins-2.0.3/pkg/fedora/requires
nagios-plugins-2.0.3/pkg/solaris/
nagios-plugins-2.0.3/pkg/solaris/preinstall
nagios-plugins-2.0.3/pkg/solaris/solpkg
nagios-plugins-2.0.3/pkg/solaris/pkginfo.in
nagios-plugins-2.0.3/pkg/solaris/pkginfo
nagios-plugins-2.0.3/pkg/redhat/
nagios-plugins-2.0.3/pkg/redhat/requires
[root@ip-172-31-93-157 downloads]#
```

i-025f1d18f7c8a8cda (nagios-host)

PublicIPs: 3.86.198.73 PrivateIPs: 172.31.93.157

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

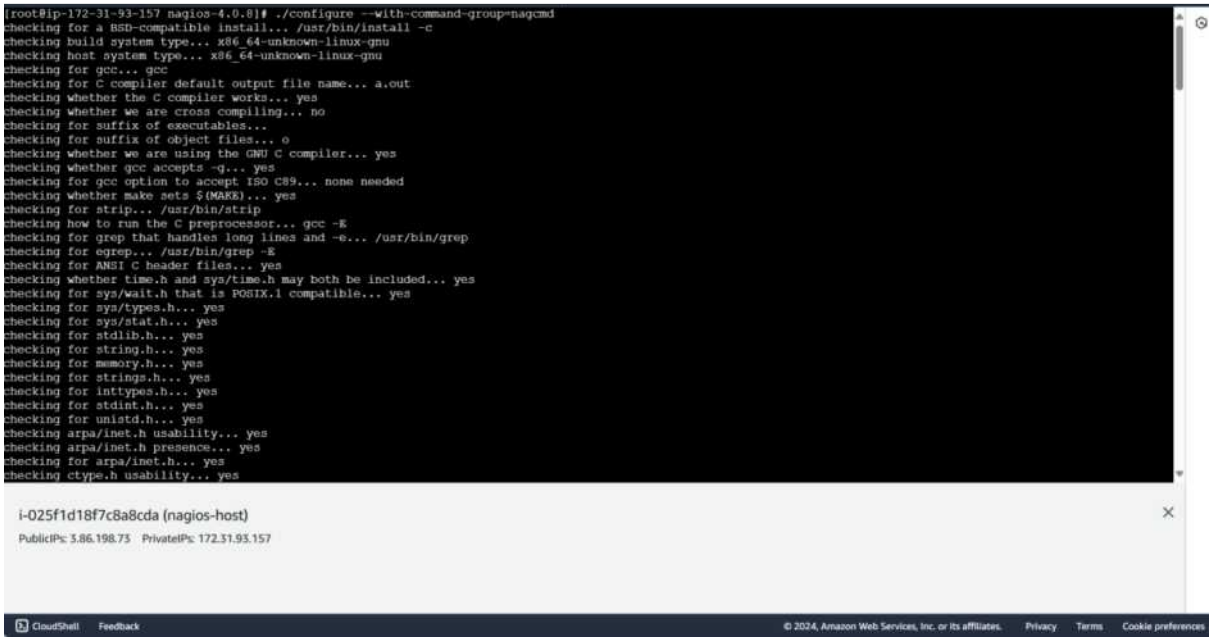
```
nagios-plugins-2.0.3/plugins-scripts/Makefile.am
nagios-plugins-2.0.3/plugins-scripts/subst.in
nagios-plugins-2.0.3/plugins-scripts/check_breeze.pl
nagios-plugins-2.0.3/plugins-scripts/check_log.sh
nagios-plugins-2.0.3/plugins-scripts/check_flexlm.pl
nagios-plugins-2.0.3/plugins-scripts/check_rpc.pl
nagios-plugins-2.0.3/plugins-scripts/check_oracle.sh
nagios-plugins-2.0.3/plugins-scripts/utils.pm.in
nagios-plugins-2.0.3/plugins-scripts/check_disk_smb.pl
nagios-plugins-2.0.3/plugins-scripts/t/
nagios-plugins-2.0.3/plugins-scripts/t/check_ifoperstatus.t
nagios-plugins-2.0.3/plugins-scripts/t/check_rpc.t
nagios-plugins-2.0.3/plugins-scripts/t/check_file_age.t
nagios-plugins-2.0.3/plugins-scripts/t/check_disk_smb.t
nagios-plugins-2.0.3/plugins-scripts/t/check_ifstatus.t
nagios-plugins-2.0.3/plugins-scripts/t/utils.t
nagios-plugins-2.0.3/plugins-scripts/check_mailq.pl
nagios-plugins-2.0.3/plugins-scripts/check_wave.pl
nagios-plugins-2.0.3/plugins-scripts/check_ircd.pl
nagios-plugins-2.0.3/plugins-scripts/utils.sh.in
nagios-plugins-2.0.3/plugins-scripts/check_ifstatus.pl
nagios-plugins-2.0.3/plugins-scripts/check_sensors.sh
nagios-plugins-2.0.3/pkg/
nagios-plugins-2.0.3/pkg/fedora/
nagios-plugins-2.0.3/pkg/fedora/requires
nagios-plugins-2.0.3/pkg/solaris/
nagios-plugins-2.0.3/pkg/solaris/preinstall
nagios-plugins-2.0.3/pkg/solaris/solpkg
nagios-plugins-2.0.3/pkg/solaris/pkginfo.in
nagios-plugins-2.0.3/pkg/solaris/pkginfo
nagios-plugins-2.0.3/pkg/redhat/
nagios-plugins-2.0.3/pkg/redhat/requires
[root@ip-172-31-93-157 downloads]# cd nagios-4.0.8
[root@ip-172-31-93-157 nagios-4.0.8]#
```

i-025f1d18f7c8a8cda (nagios-host)

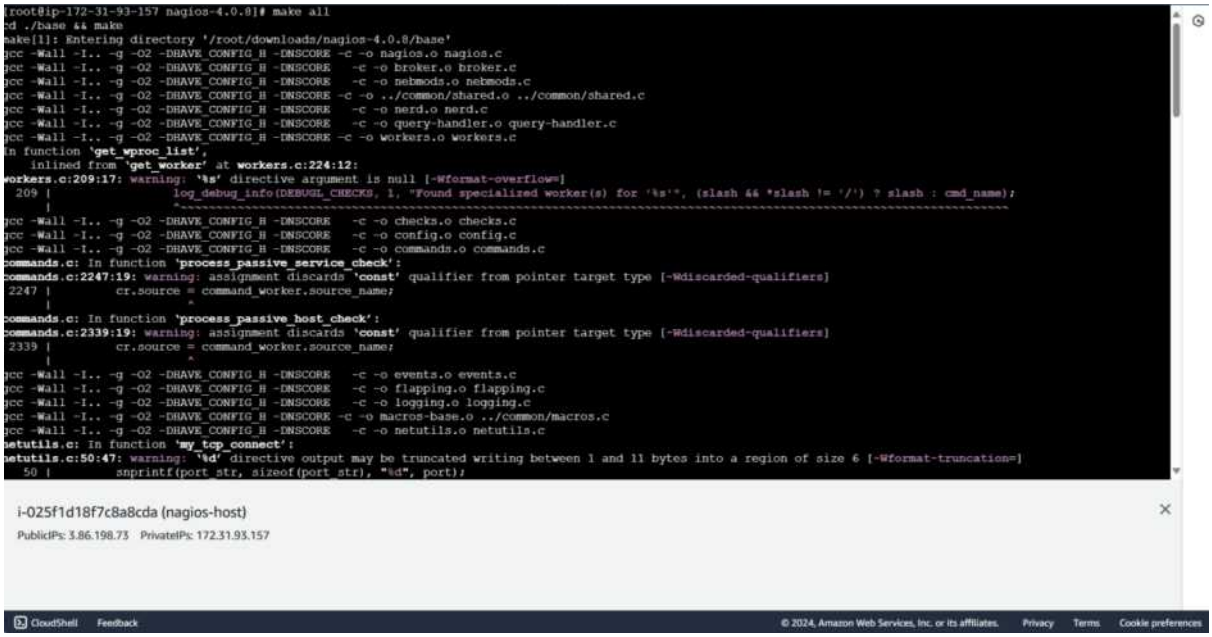
PublicIPs: 3.86.198.73 PrivateIPs: 172.31.93.157

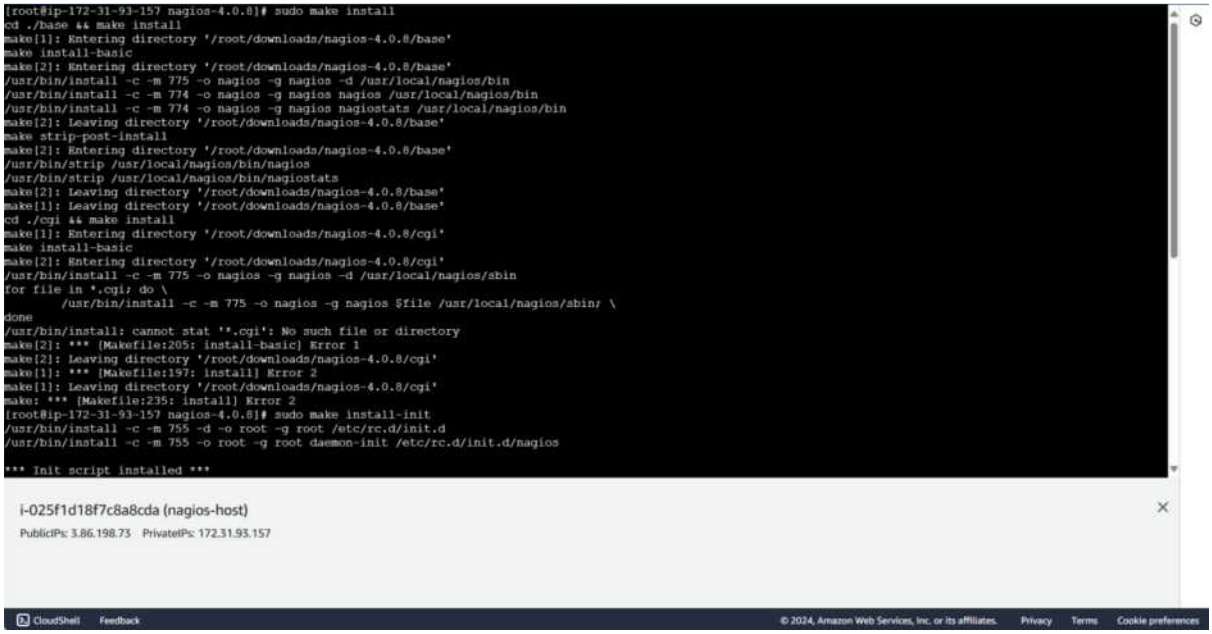
CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

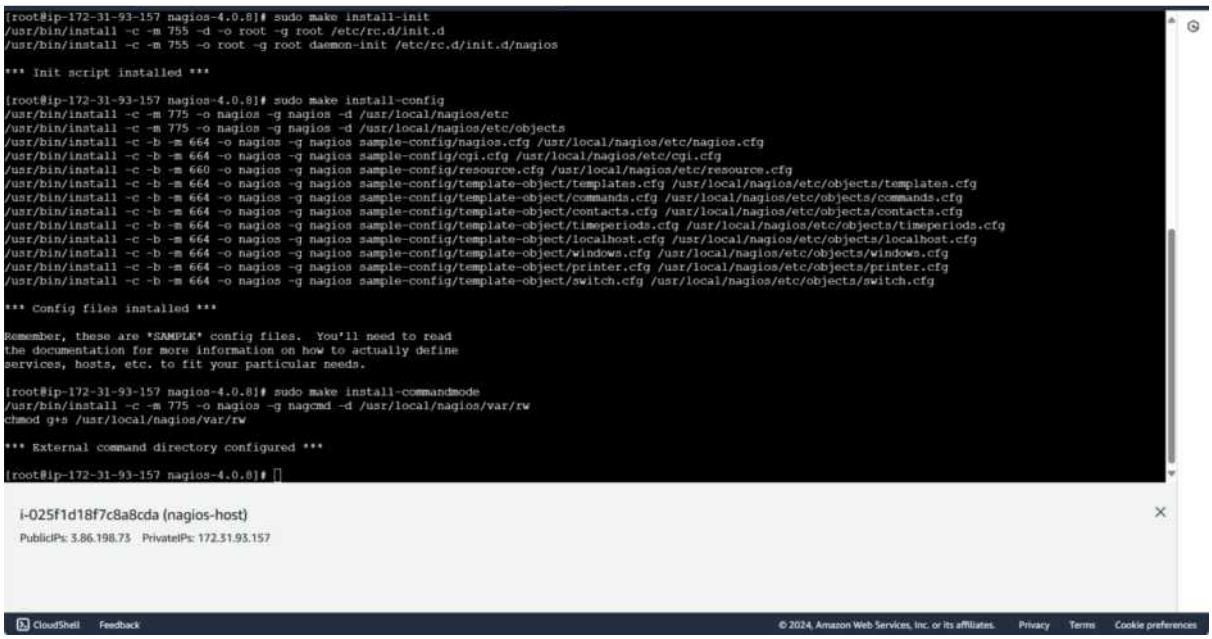
Step 6: Now to run the configuration script run the following command.
./configure --with-command-group=nagcmd



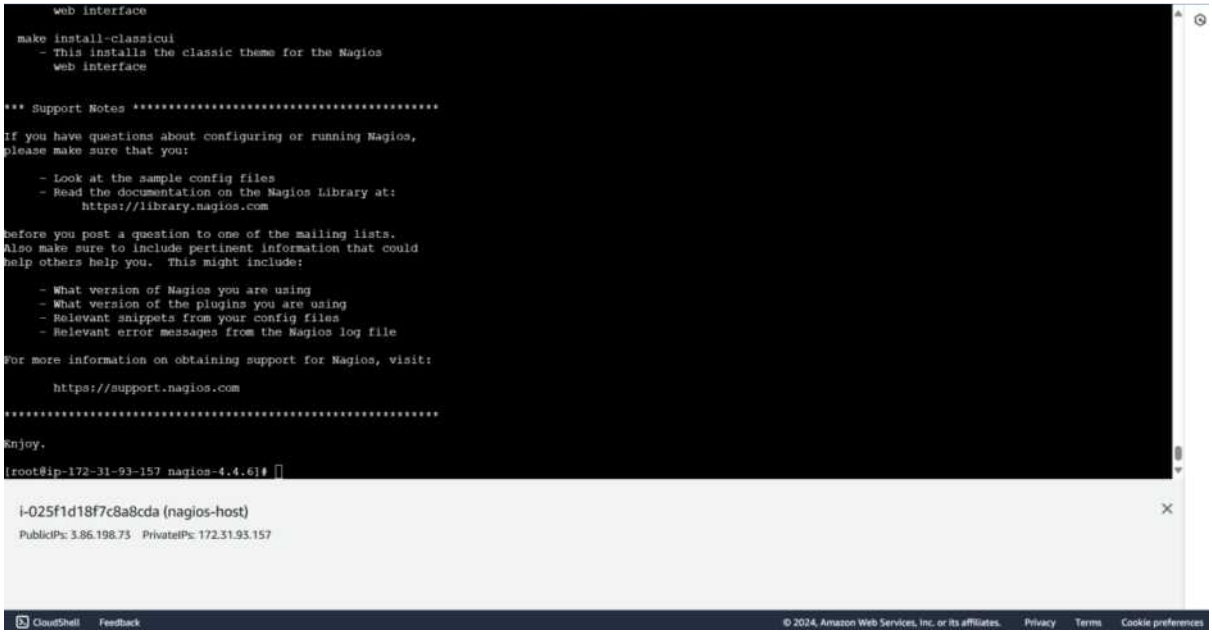
Step 7: Now, to compile the source code run the following command -
make all



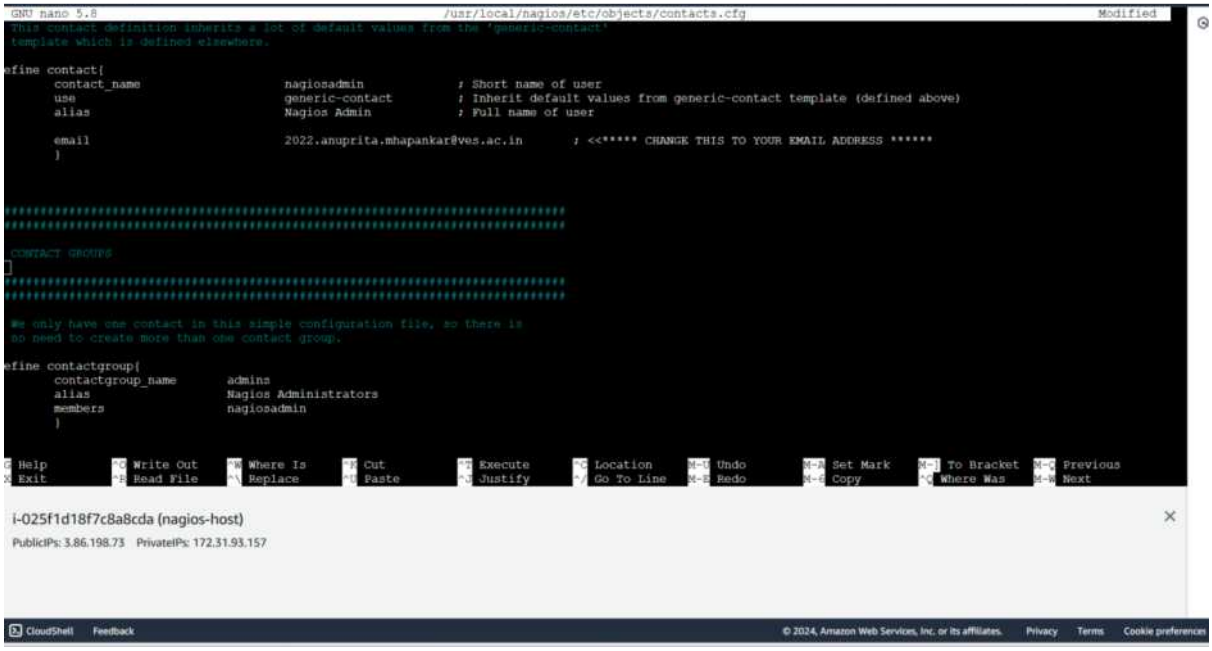




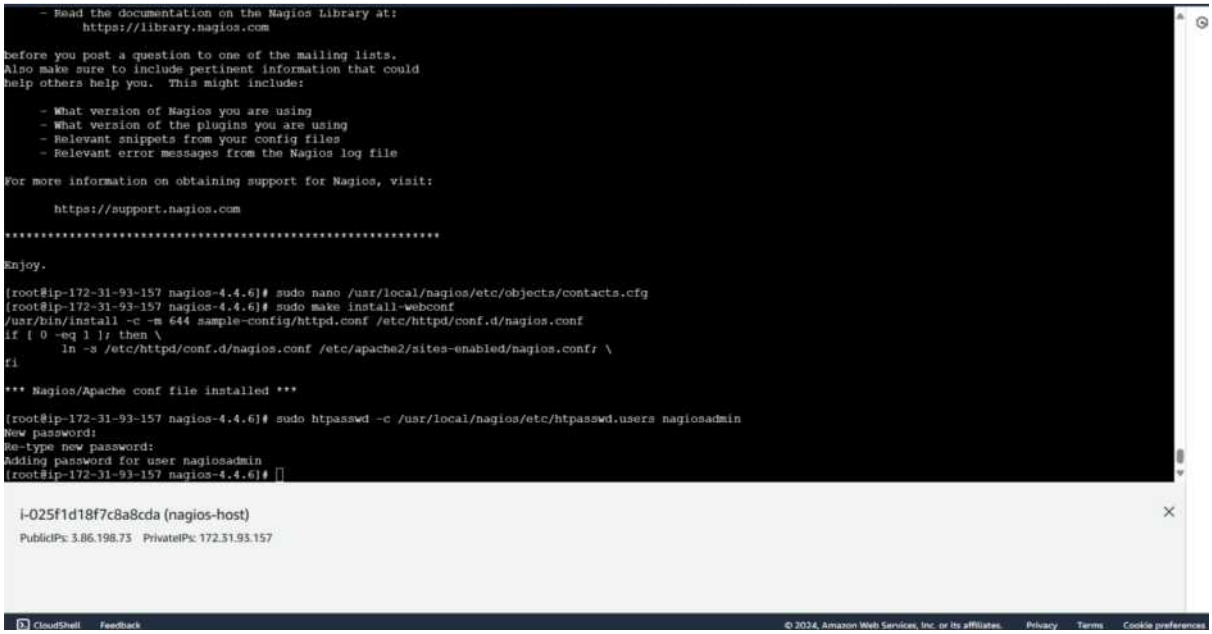
To resolve the errors run the following commands -
sudo yum install -y gcc glibc glibc-common gd gd-devel make net-snmp openssl-devel
rm -rf nagios-4.0.8
cd ~/downloads/nagios-4.4.6
./configure --with-command-group=nagcmd
make all
sudo make install

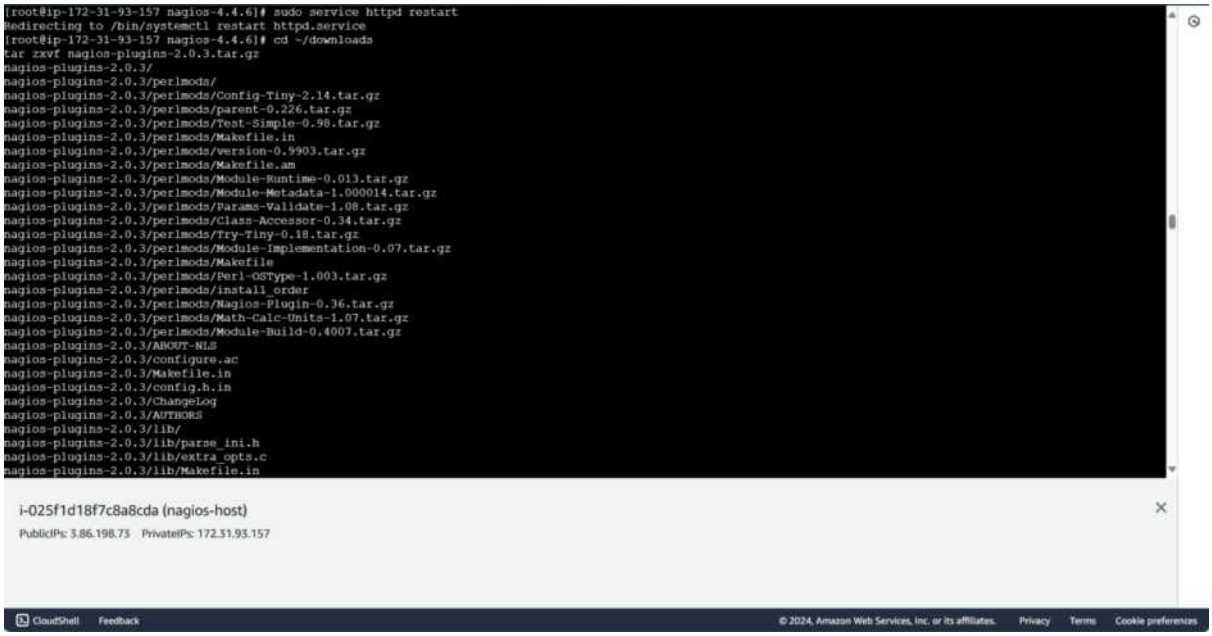


Step 8: Edit the config file and change the email address.
sudo nano /usr/local/nagios/etc/objects/contacts.cfg

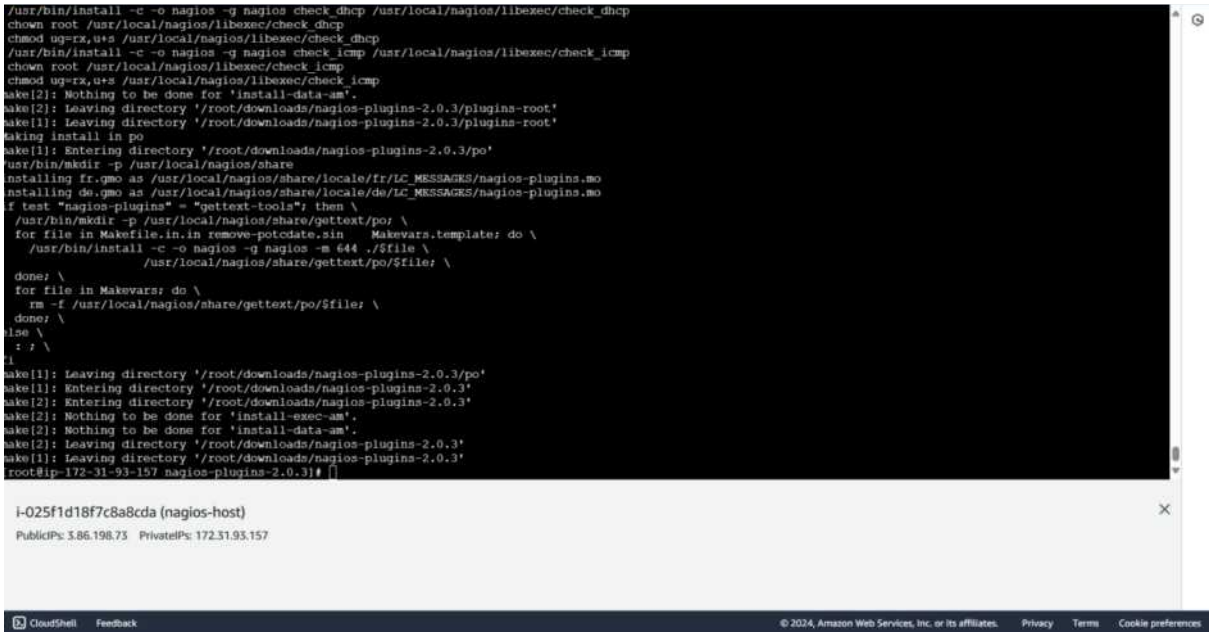


Step 9: Now run the following commands –
sudo make install-webconf
sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
sudo service httpd restart
cd ~/downloads
tar zxvf nagios-plugins-2.0.3.tar.gz





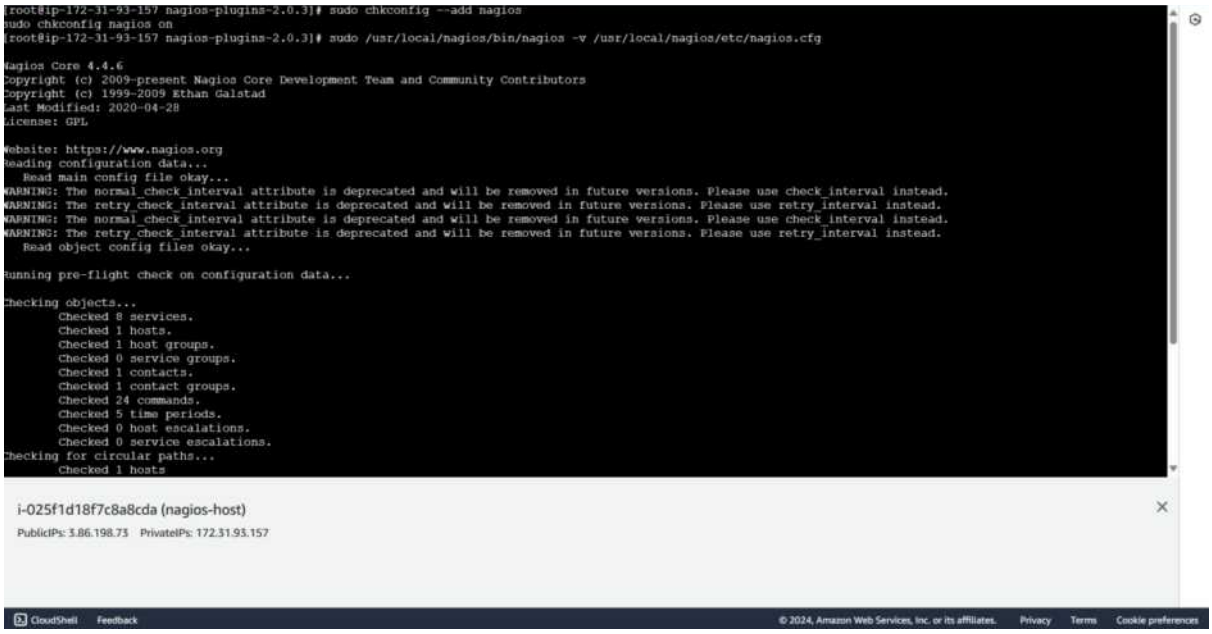
Step 10: Compile and install plugins
cd nagios-plugins-2.0.3
./configure --with-nagios-user=nagios --with-nagios-group=nagios
make
sudo make install



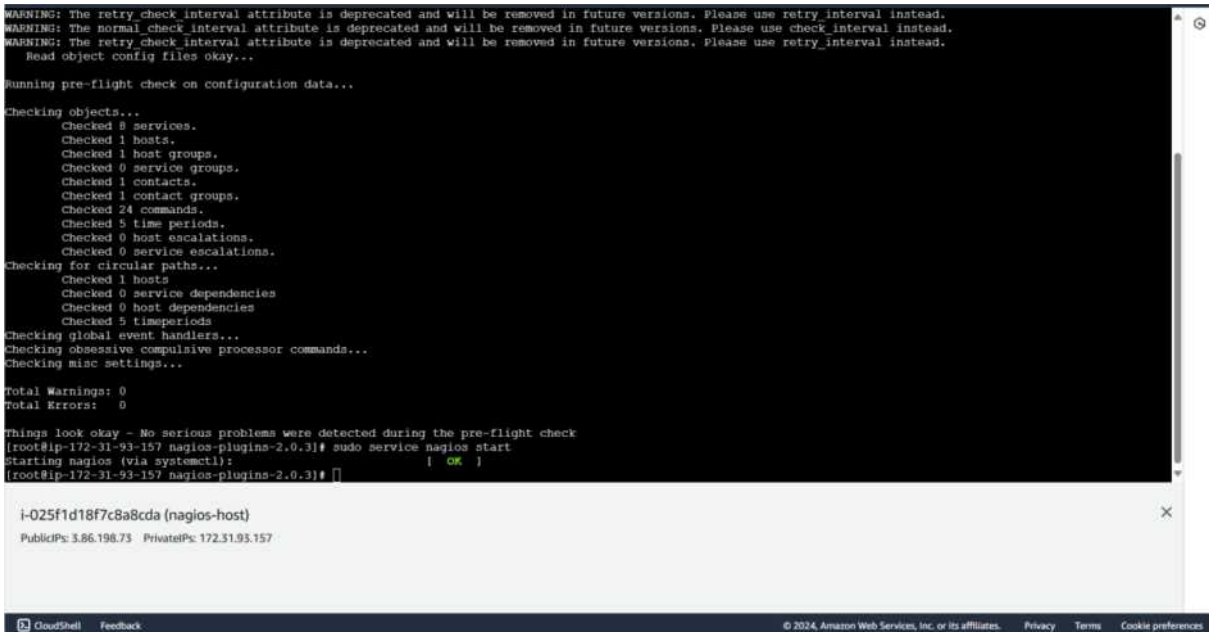
Step 11: To start nagios run the following commands –
sudo chkconfig --add nagios

sudo chkconfig nagios on

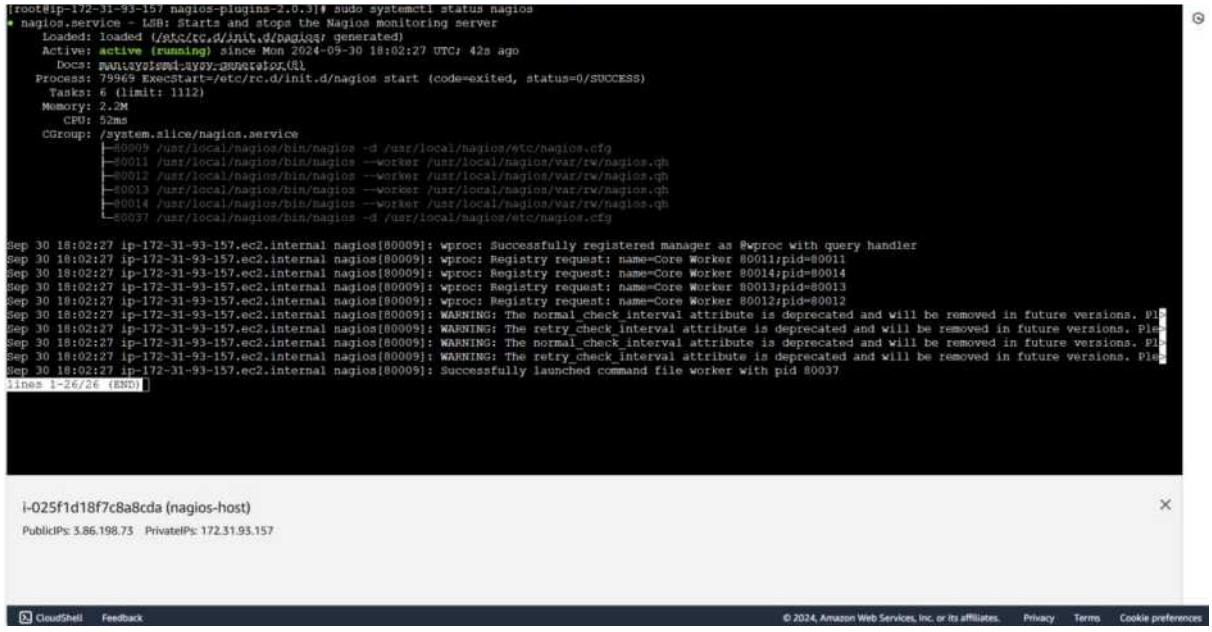
Verify using the following command -
sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg



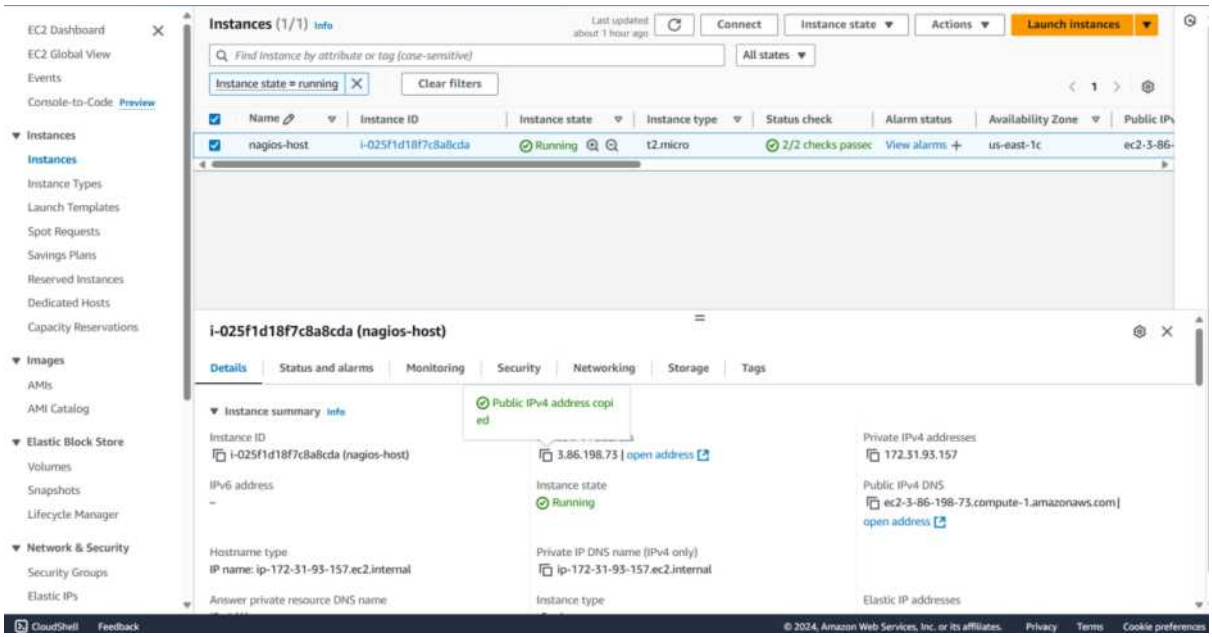
If there are no errors run the following command –
sudo service nagios start



Check status using the following command -
sudo systemctl status nagios



Step 12: Go to EC2 instance and copy the public IP address of the instance

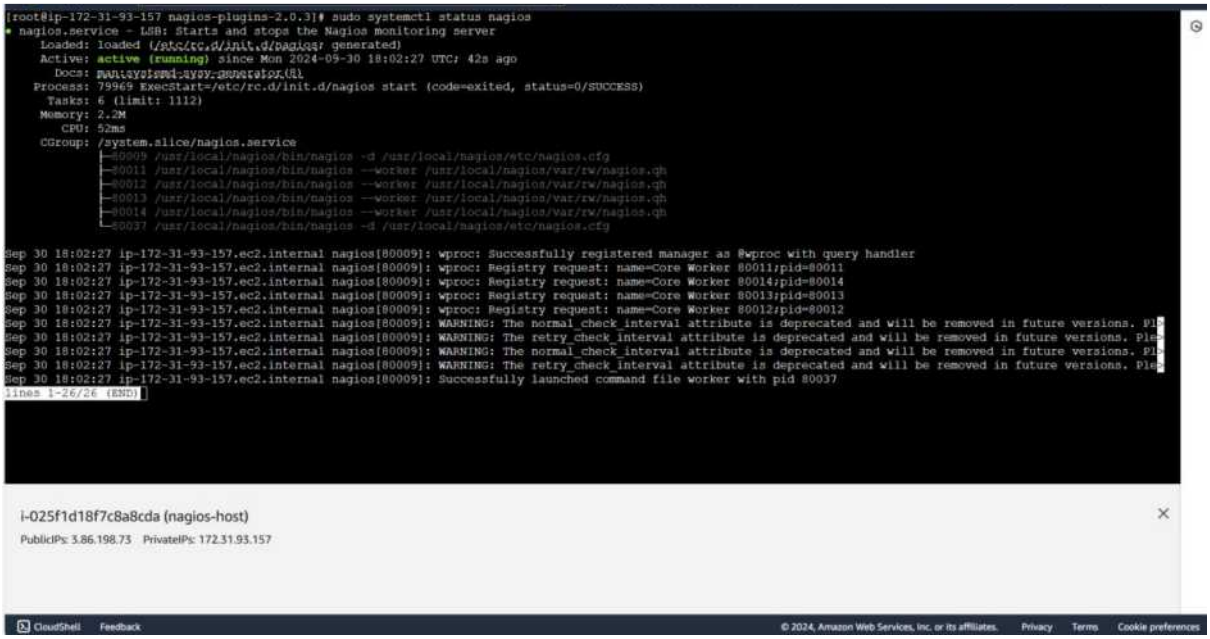


Step 13: Now visit http://<your_public_ip_address>/nagios Enter correct credentials and then you will see this page.

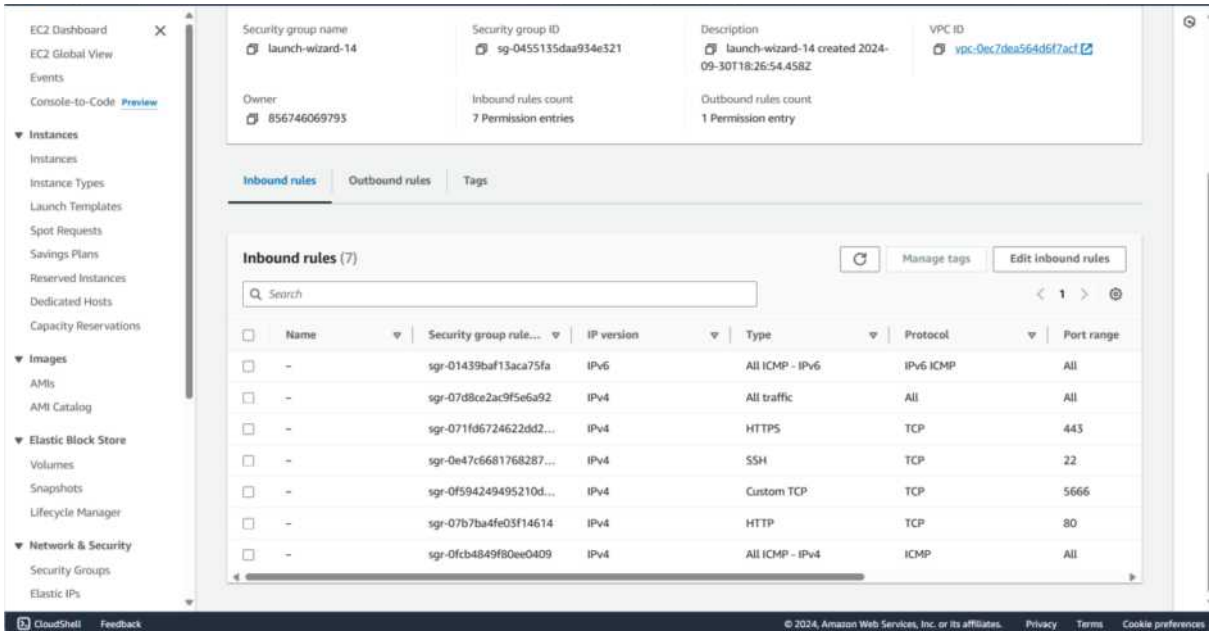
Experiment 10

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

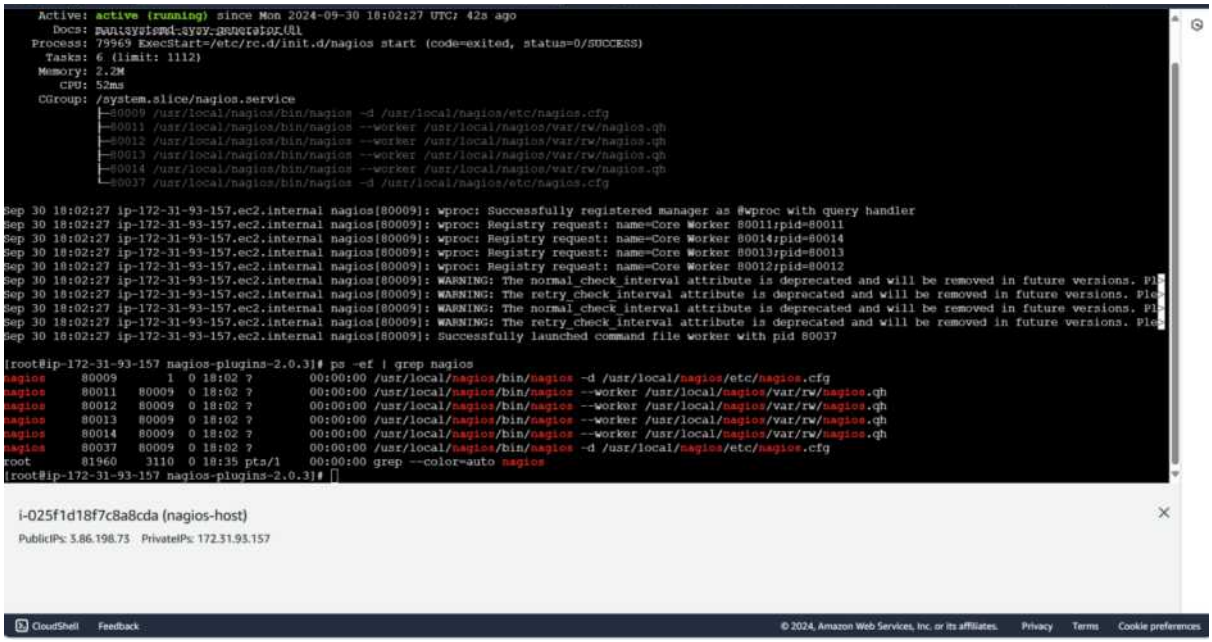
Step 1: Initially confirm that Nagios is running on the server side. For this run the following command -
sudo systemctl status nagios
on the nagios-host instance.



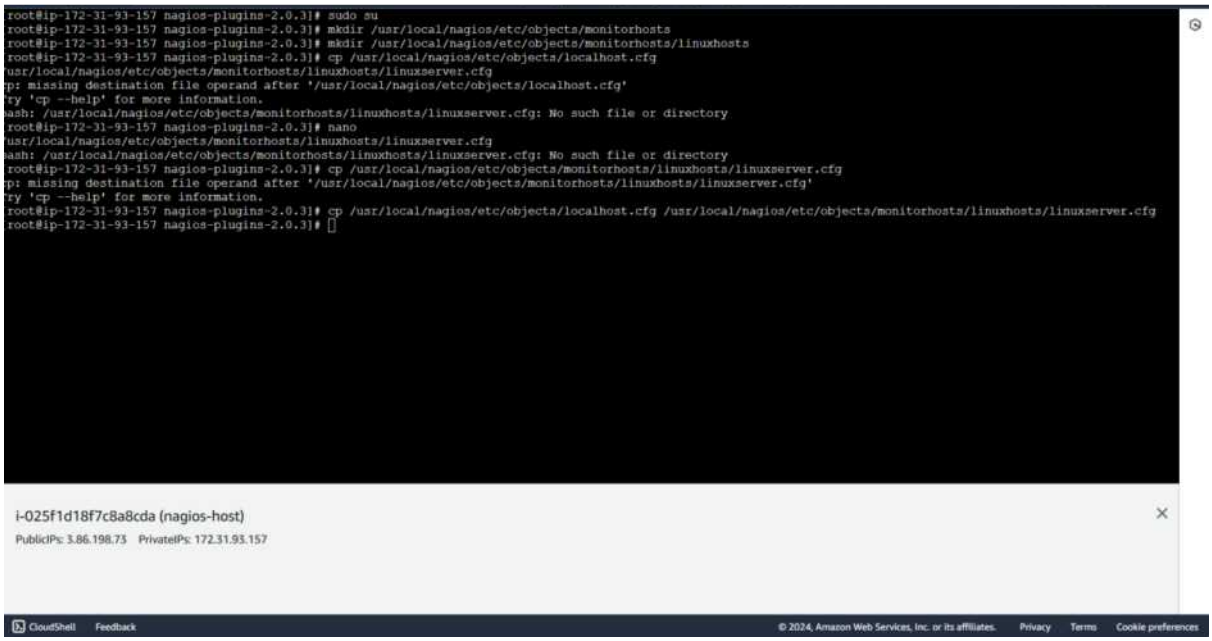
Step 2: Once confirmed, make another instance with the same security group as that of nagios-host.
For now, leave this machine as it is, and go back to your nagios-host machine.



Step 3: Now run the following command -
ps -ef | grep nagios

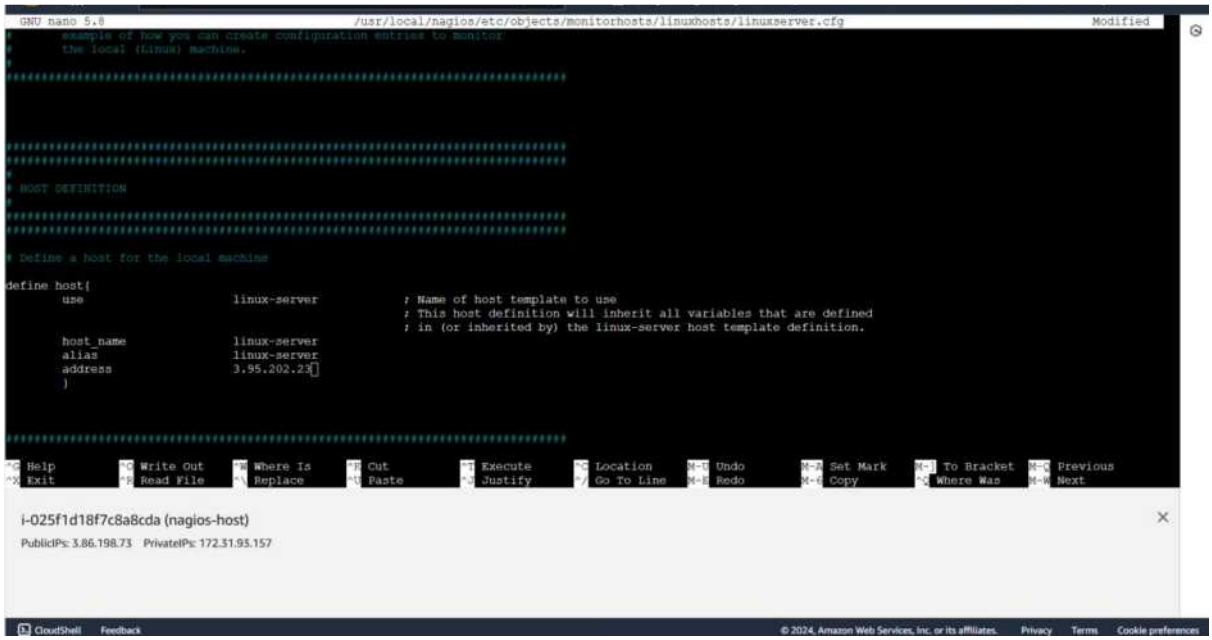


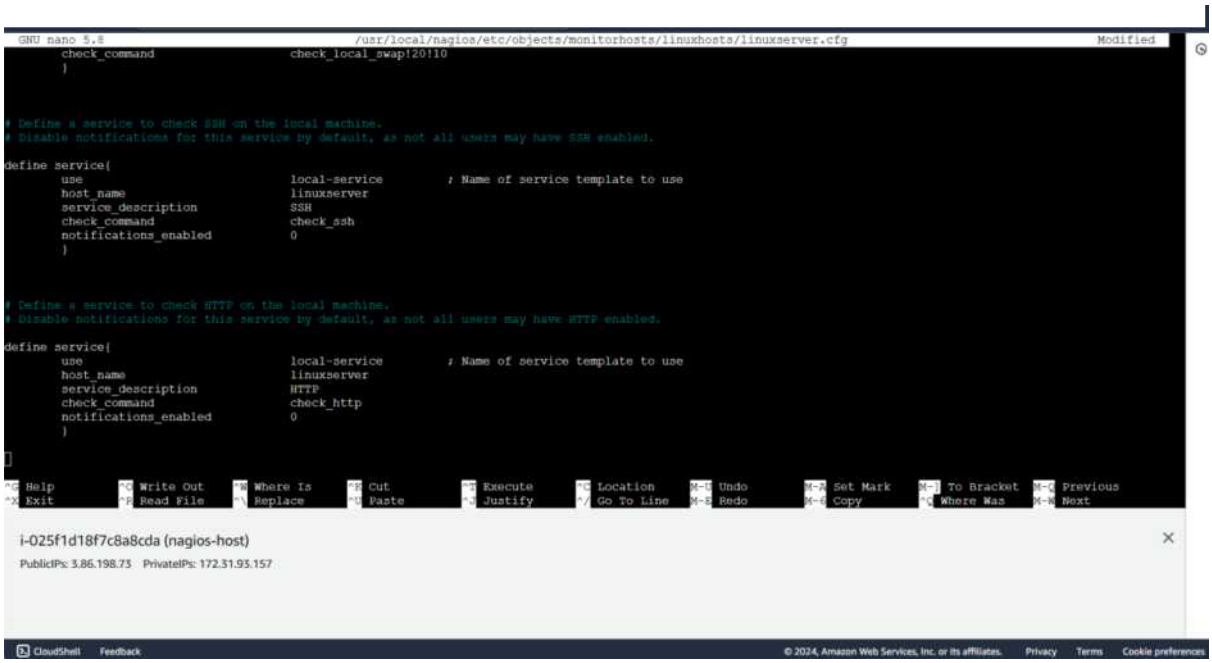
Step 4: Now, run the following commands -
sudo su
mkdir /usr/local/nagios/etc/objects/monitorhosts
mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
cp/usr/local/nagios/etc/objects/localhost.cfg/usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg



Step 5: Open linuxserver.cfg using the the following command -
nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

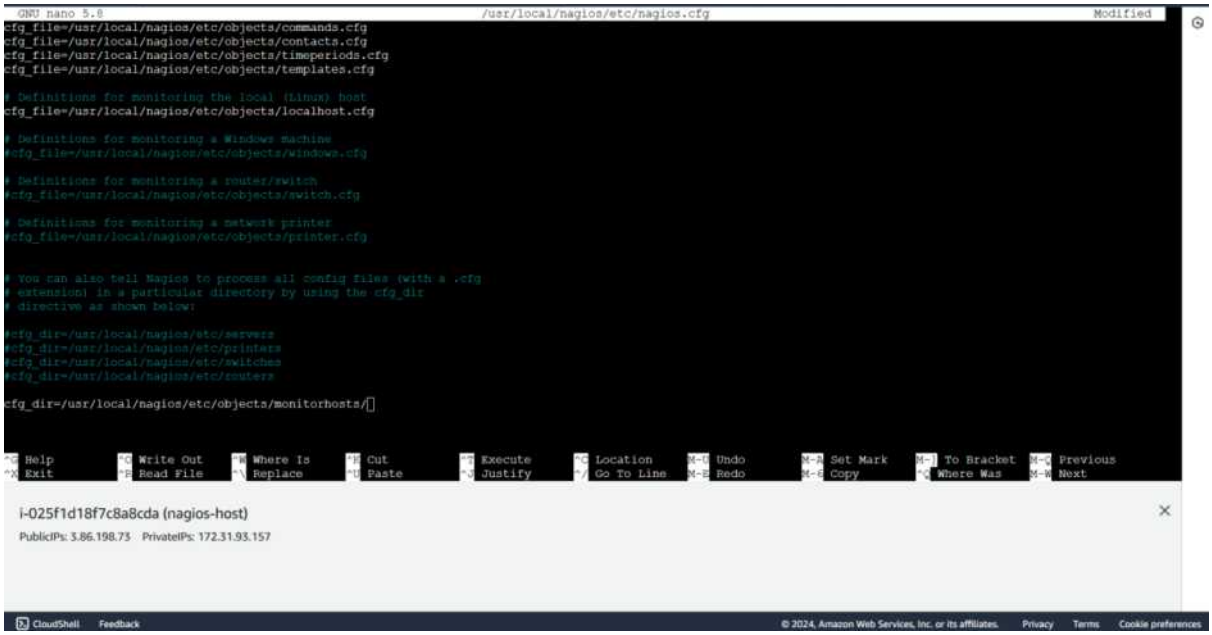
Change the hostname to linuxserver (EVERYWHERE ON THE FILE)
Change address to the public IP address of your LINUX CLIENT.
Change hostgroup_name under hostgroup to linux-servers1





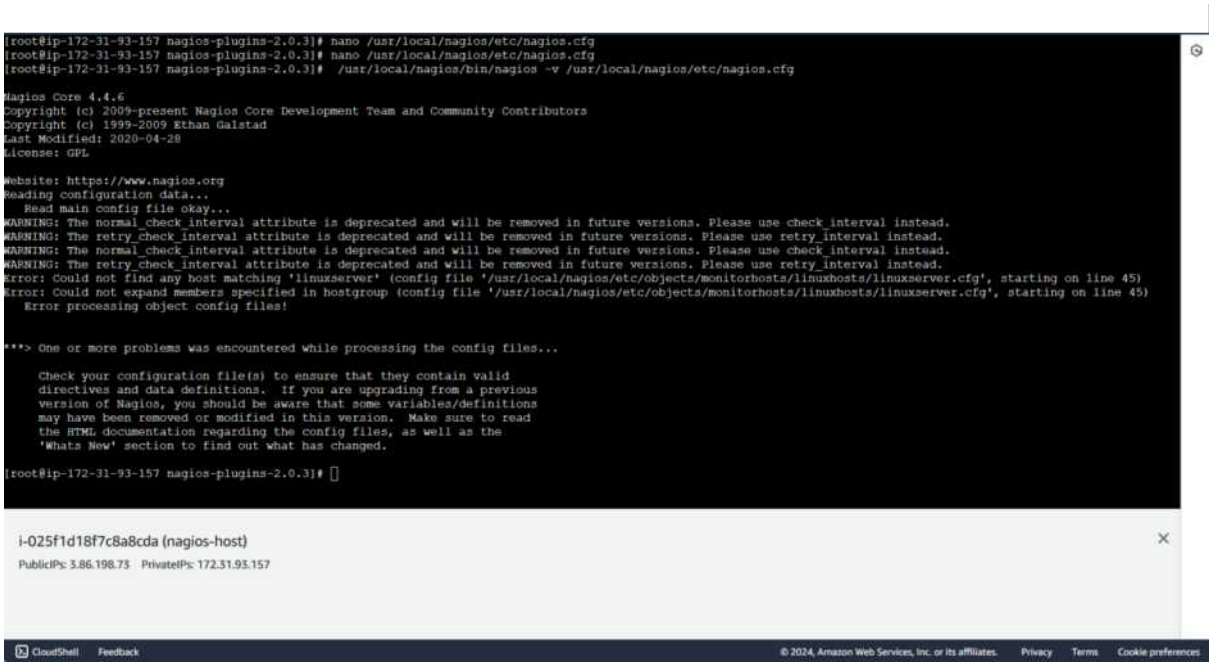
Step 6: Open Nagios config file and add the following line -
nano /usr/local/nagios/etc/nagios.cfg

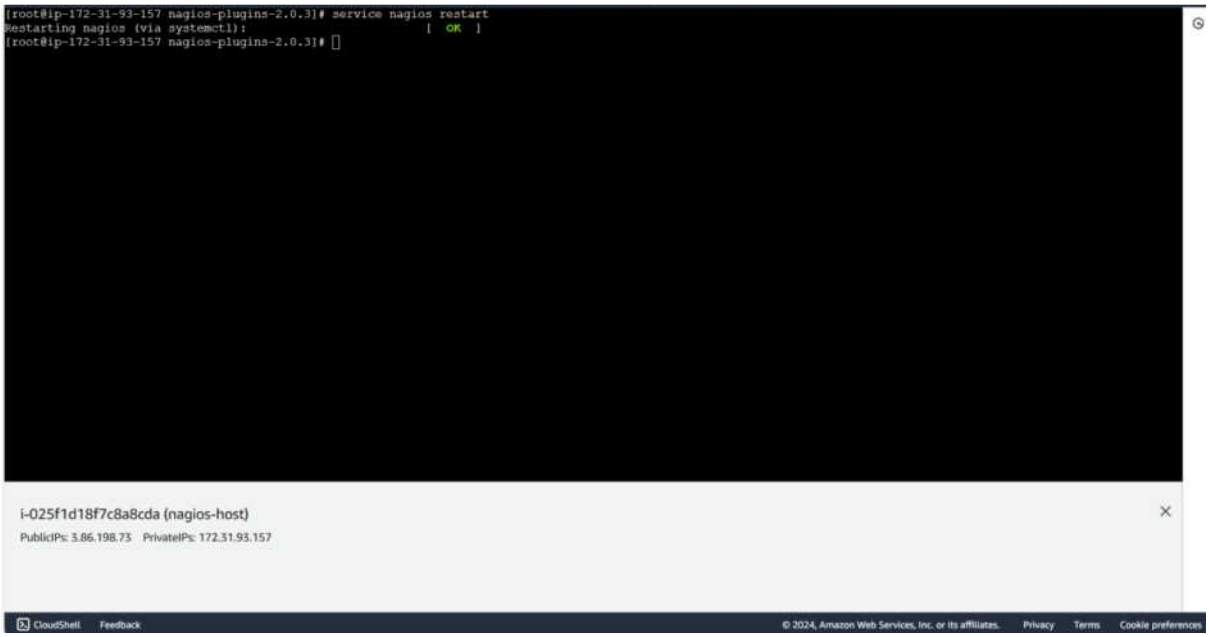
Then add this line -
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/



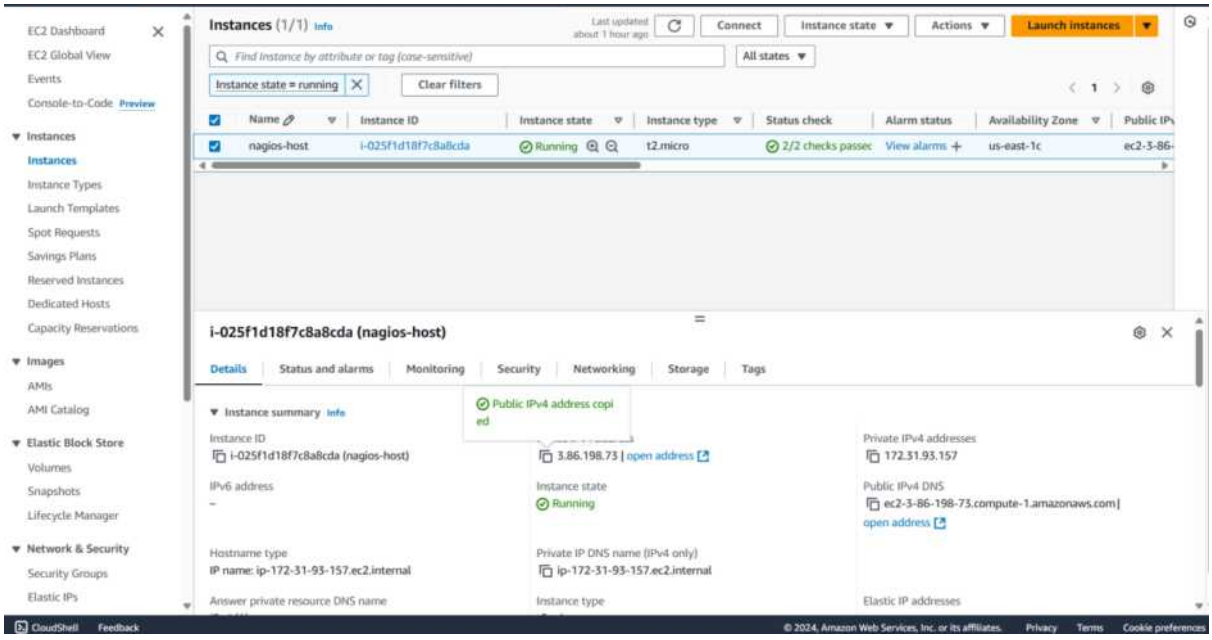
Step 8: Verify configuration files using the following command -
sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

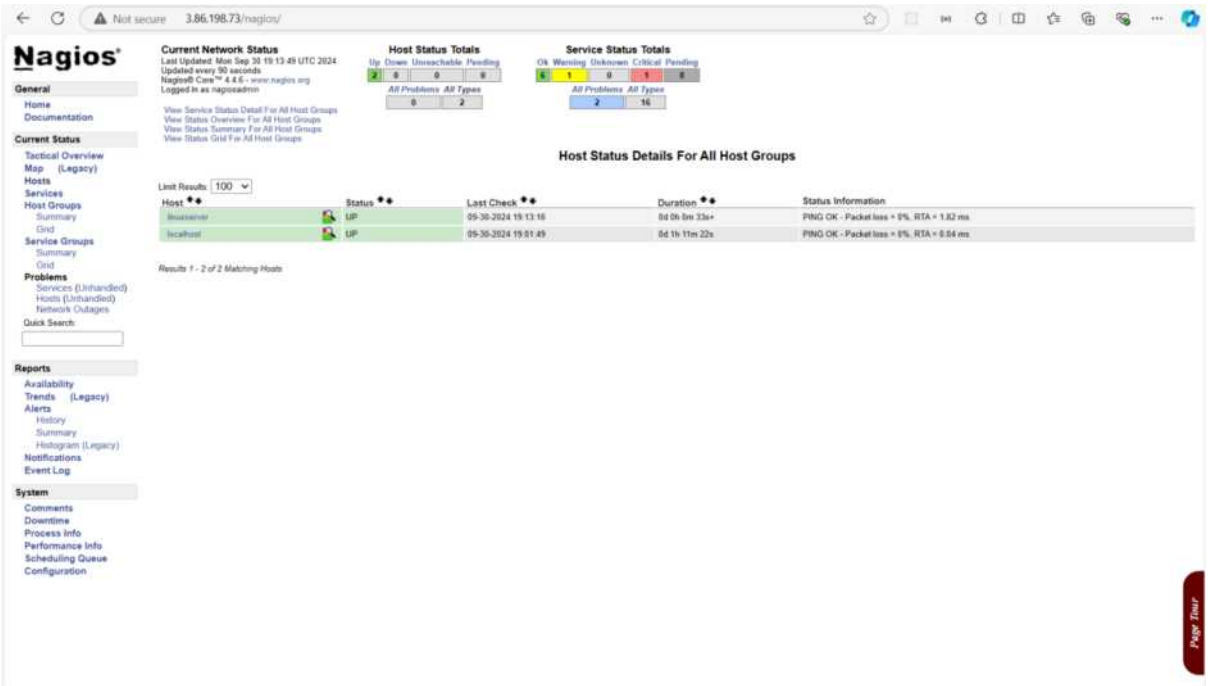
If there are no errors, run the following command -
sudo service nagios start





Step 9: After entering the correct credentials, you will see this page.




Not secure 3.86.198.73/nagios/
Nagios
General
Home
Documentation
Current Status
Tactical Overview
Map (Legacy)
Hosts
Services
Host Groups
Summary
Grid
Service Groups
Summary
Grid
Problems
Services (Unhandled)
Hosts (Unhandled)
Network Outages
Quick Search:
Reports
Availability
Trends (Legacy)
Alerts
History
Summary
Histogram (Legacy)
Notifications
Event Log
System
Comments
Downtime
Process Info
Performance Info
Scheduling Queue
Configuration
Current Network Status
Last Updated: Mon Sep 30 19:13:49 UTC 2024
Updated every 90 seconds
Nagios® Core™ 4.4.6 - www.nagios.org
Logged in as nagiosadmin
View Service Status Detail For All Host Groups
View Status Overview For All Host Groups
View Status Summary For All Host Groups
View Status Grid For All Host Groups
Host Status Totals
Up Down Unreachable Pending
2 0 0 0
All Problems All Types
0 2
Service Status Totals
Ok Warning Unknown Critical Pending
5 1 0 1 8
All Problems All Types
2 16
Host Status Details For All Host Groups
Limit Results: 100
Host
Status
Last Check
Duration
Status Information
Bouncerver
UP
09-30-2024 19:13:16
0d 0h 0m 33s+
PING OK - Packet loss = 0%, RTA = 1.82 ms
localhost
UP
09-30-2024 19:01:49
0d 1h 11m 22s
PING OK - Packet loss = 0%, RTA = 0.04 ms
Results 1 - 2 of 2 Matching Hosts
Page Tour

ADVANCE DEVOPS EXP 11

Name:Manav Punjabi

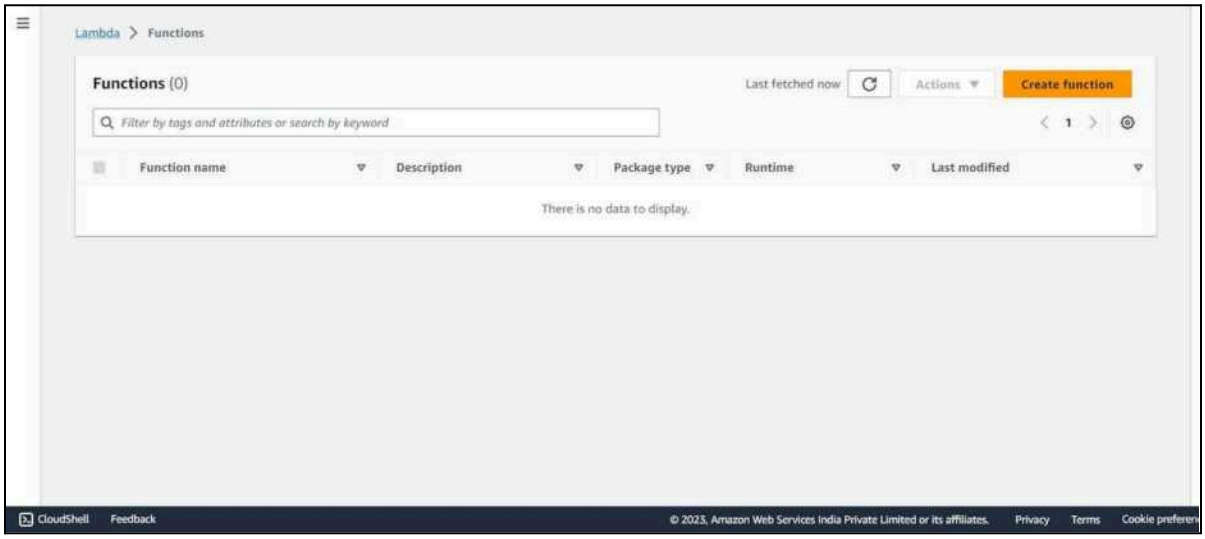
Class: D15A

Roll No:45

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

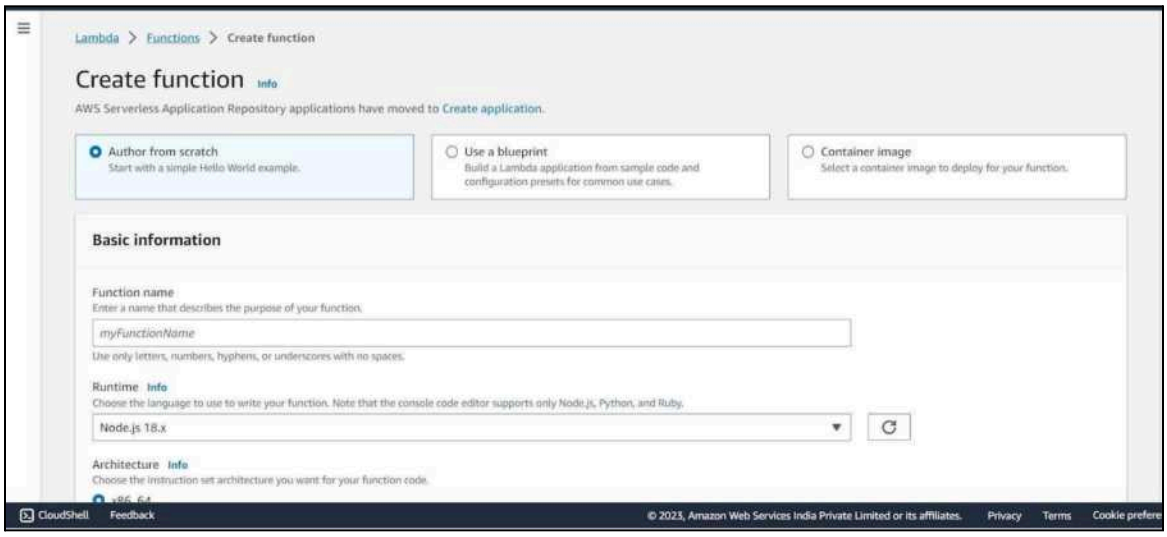
Steps to create an AWS Lambda function

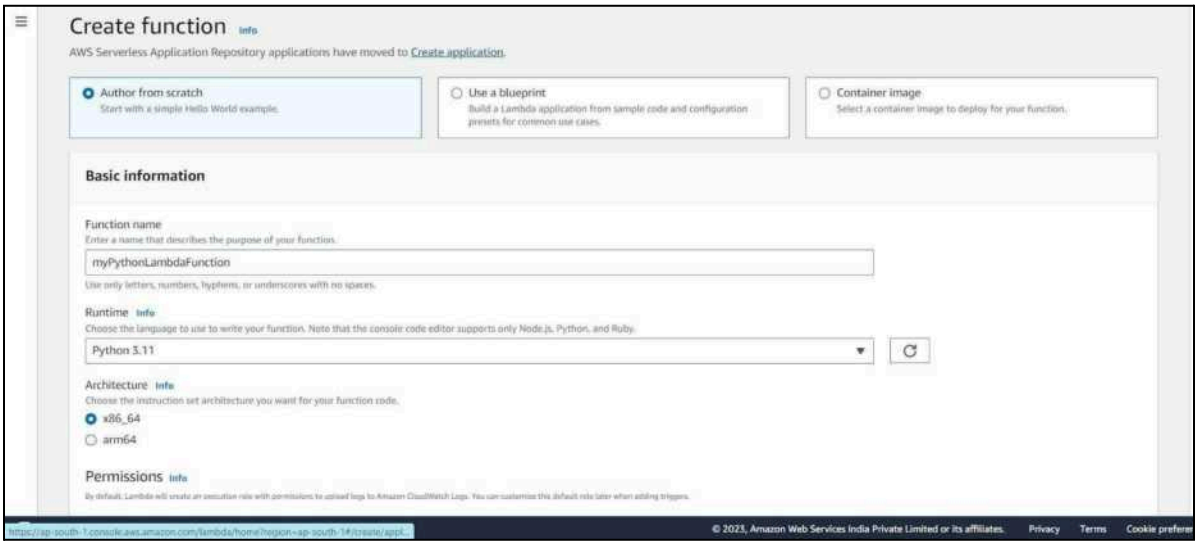
Step 1:Open up the Lambda Console and click on the Create button.
Be mindful of where you create your functions since Lambda is region-dependent.

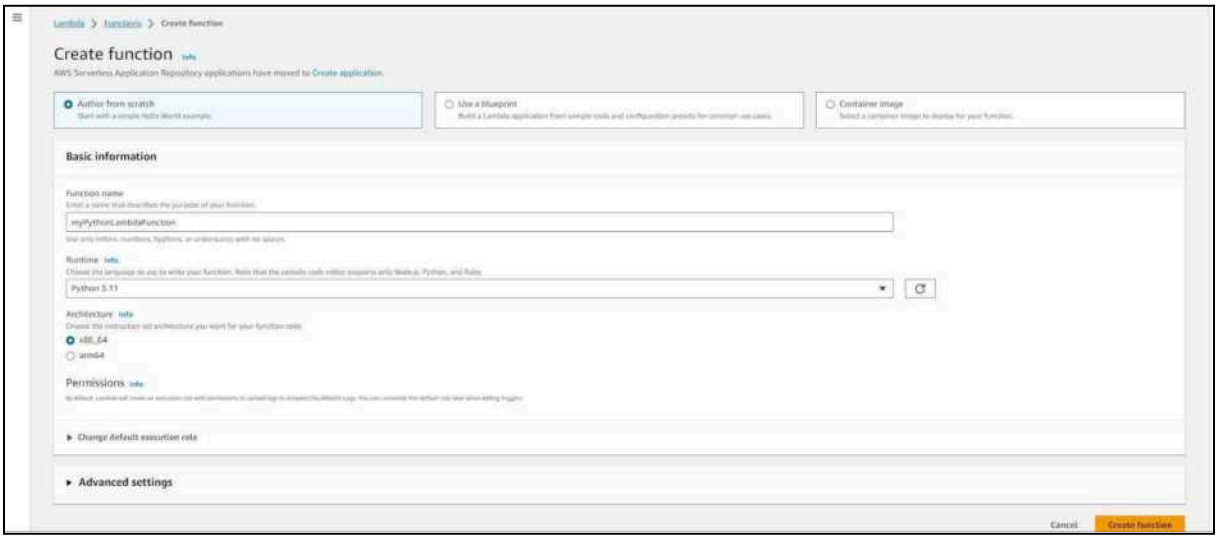


2. Choose to create a function from scratch or use a blueprint, i.e templates defined by AWS for you with all configuration presets required for the most common use cases.

Then, choose a runtime env for your function, under the dropdown, you can see all the options AWS supports, Python, Nodejs, .NET and Java being the most popular ones. After that, choose to create a new role with basic Lambda permissions if you don't have an existing one.

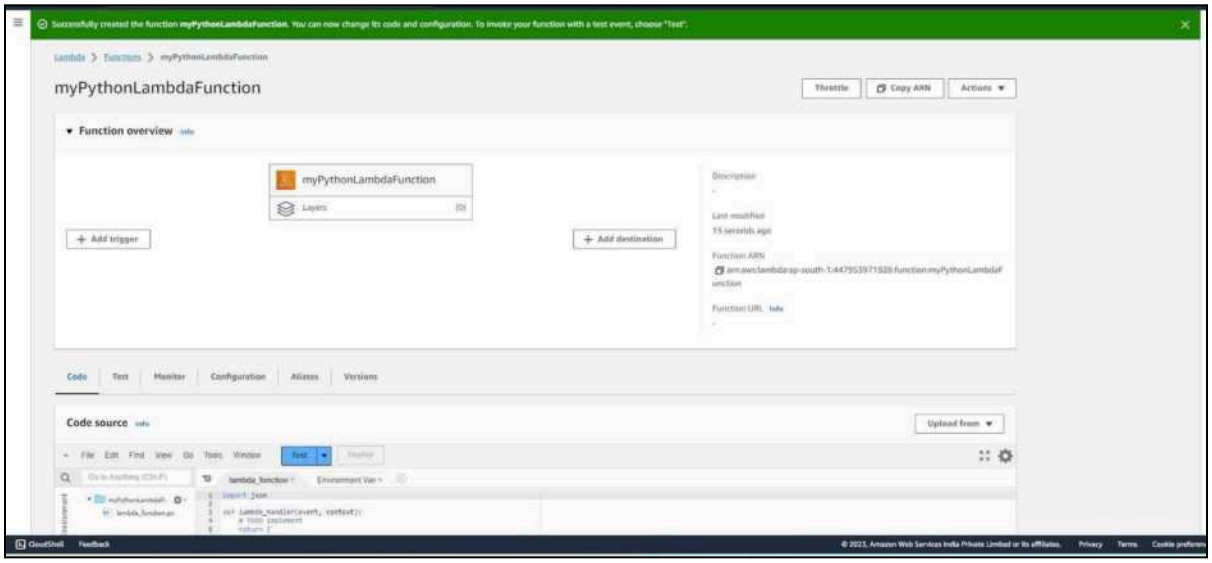


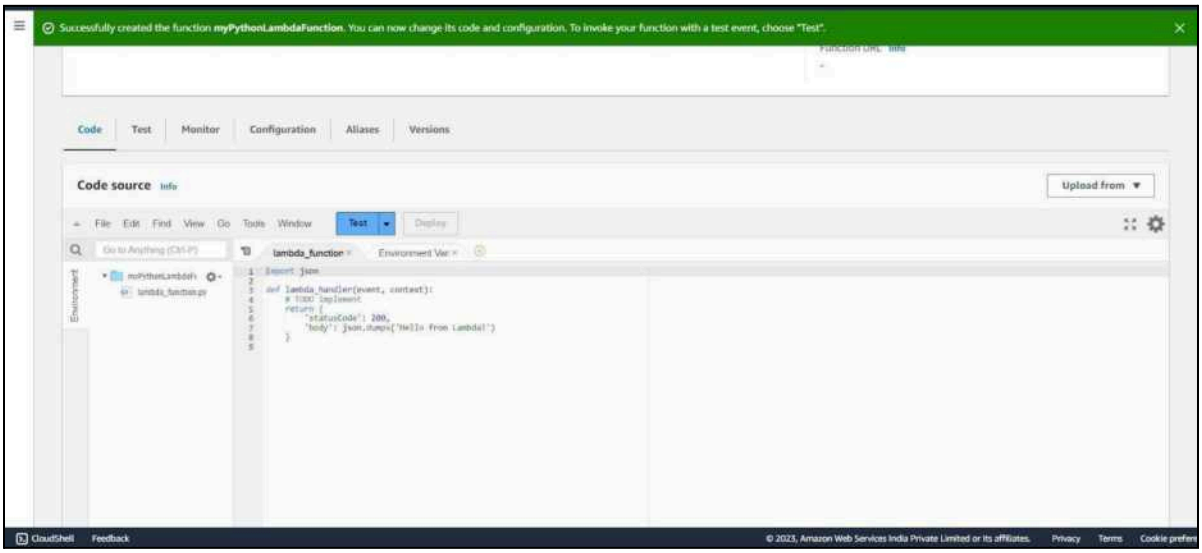




Click on the Create button.

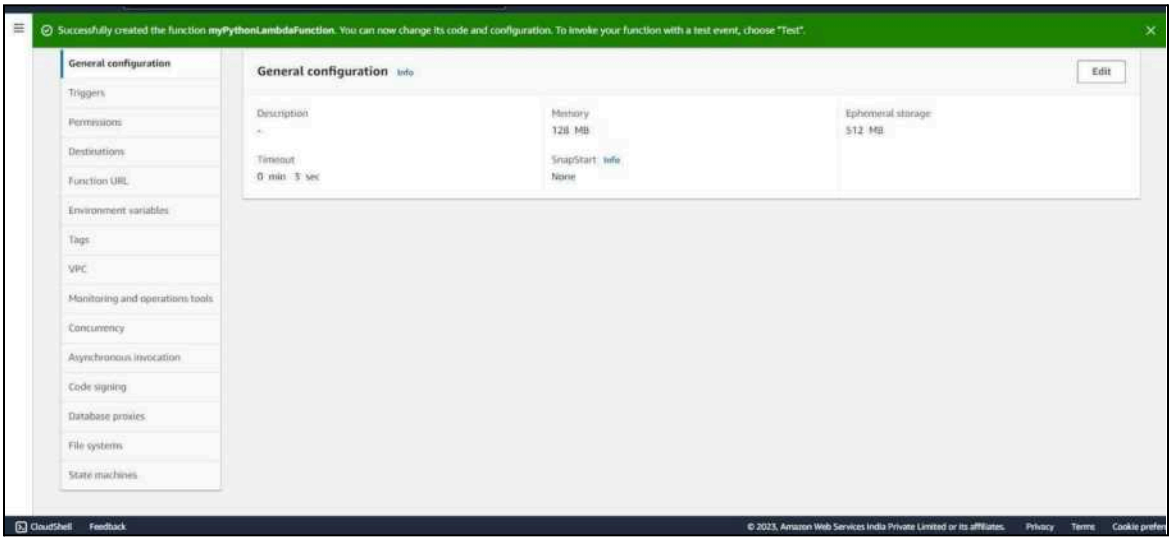
3. This process will take a while to finish and after that, you'll get a message that your function was successfully created.

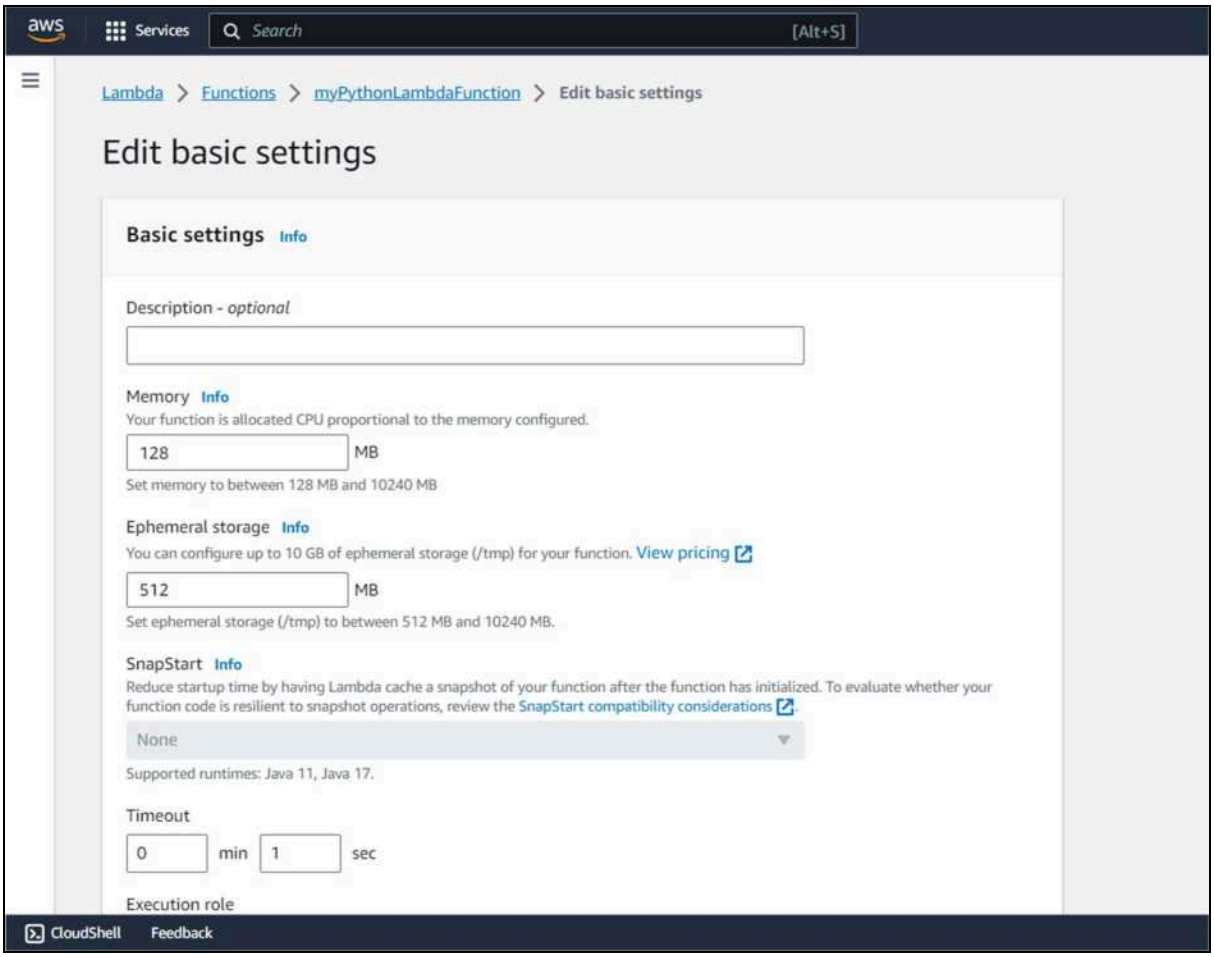




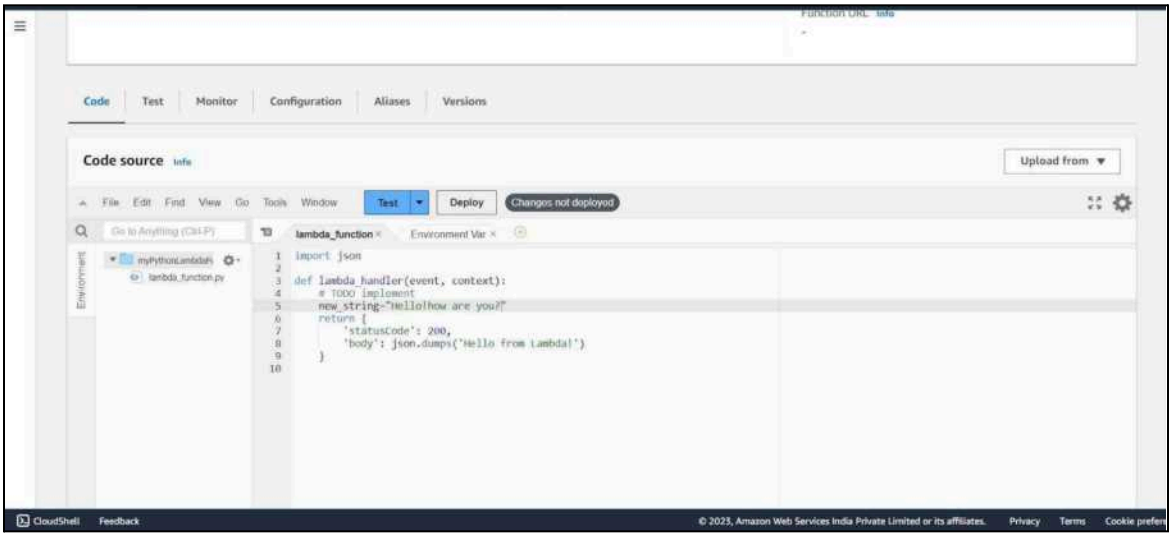
4. To change the configuration, open up the Configuration tab and under General Configuration, choose Edit.

Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.

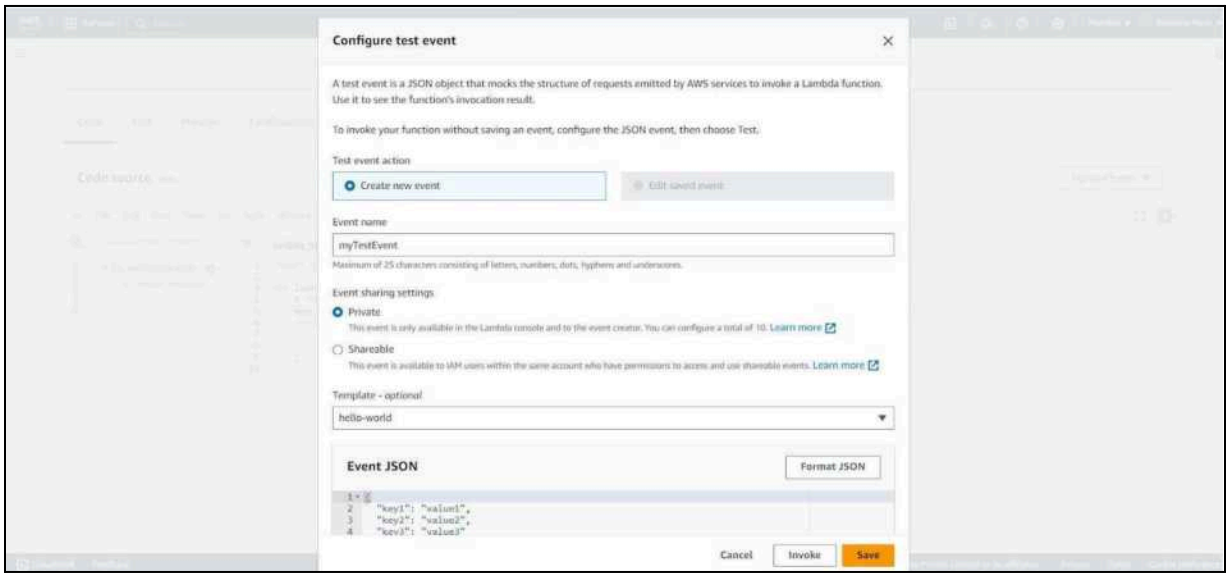




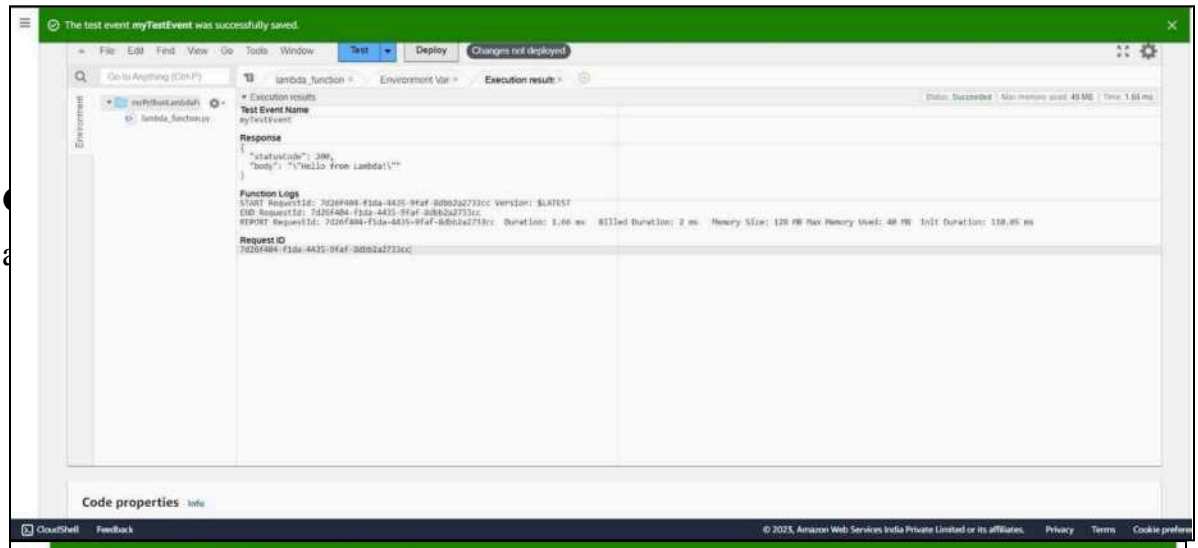
5. You can make changes to your function inside the code editor. You can also upload a zip file of your function or upload one from an S3 bucket if needed.
Press Ctrl + S to save the file and click Deploy to deploy the changes.

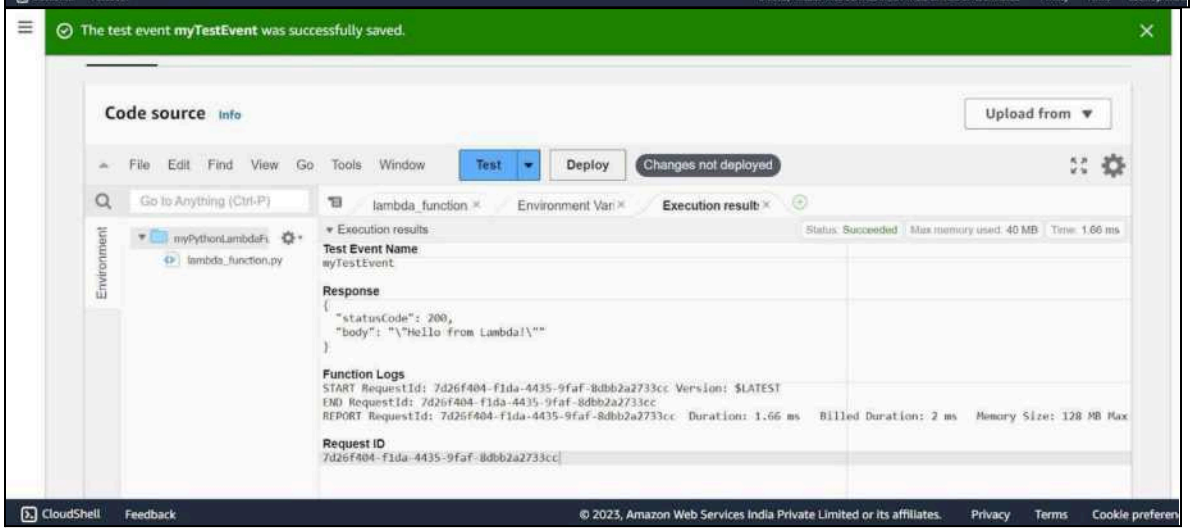


6. Click on Test and you can change the configuration, like so. If you do not have anything in the request body, it is important to specify two curly braces as valid JSON, so make sure they are there.



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

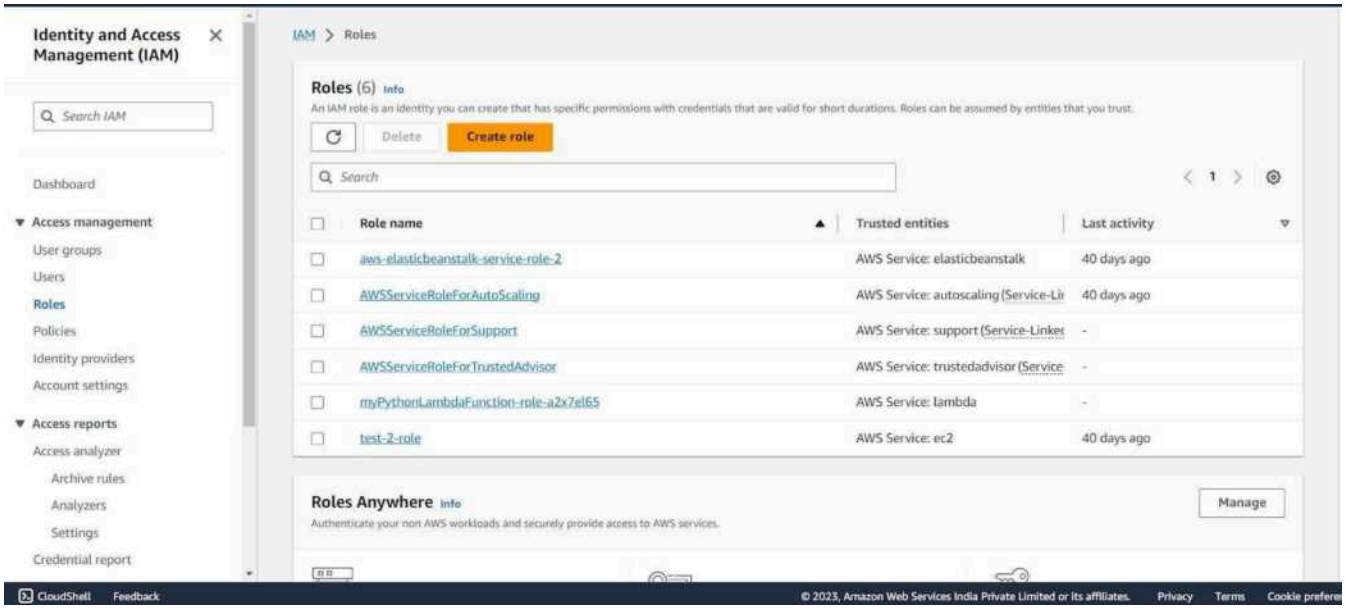

The test event myTestEvent was successfully saved.
Code properties Info
CloudShell Feedback


The test event myTestEvent was successfully saved.
Code source Info
Upload from
File Edit Find View Go Tools Window
Test
Deploy
Changes not deployed
Go to Anything (Ctrl-P)
lambda_function
Environment Var
Execution result
Environment
myPythonLambdaFi
lambda_function.py
Execution results
Status: Succeeded
Max memory used: 40 MB
Time: 1.66 ms
Test Event Name
myTestEvent
Response
{
"statusCode": 200,
"body": "\"Hello from Lambda!\""
}
Function Logs
START RequestId: 7d26f404-f1da-4435-9faf-8dbb2a2733cc Version: $LATEST
END RequestId: 7d26f404-f1da-4435-9faf-8dbb2a2733cc
REPORT RequestId: 7d26f404-f1da-4435-9faf-8dbb2a2733cc Duration: 1.66 ms Billed Duration: 2 ms Memory Size: 128 MB Max
Request ID
7d26f404-f1da-4435-9faf-8dbb2a2733cc
CloudShell
Feedback
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferen

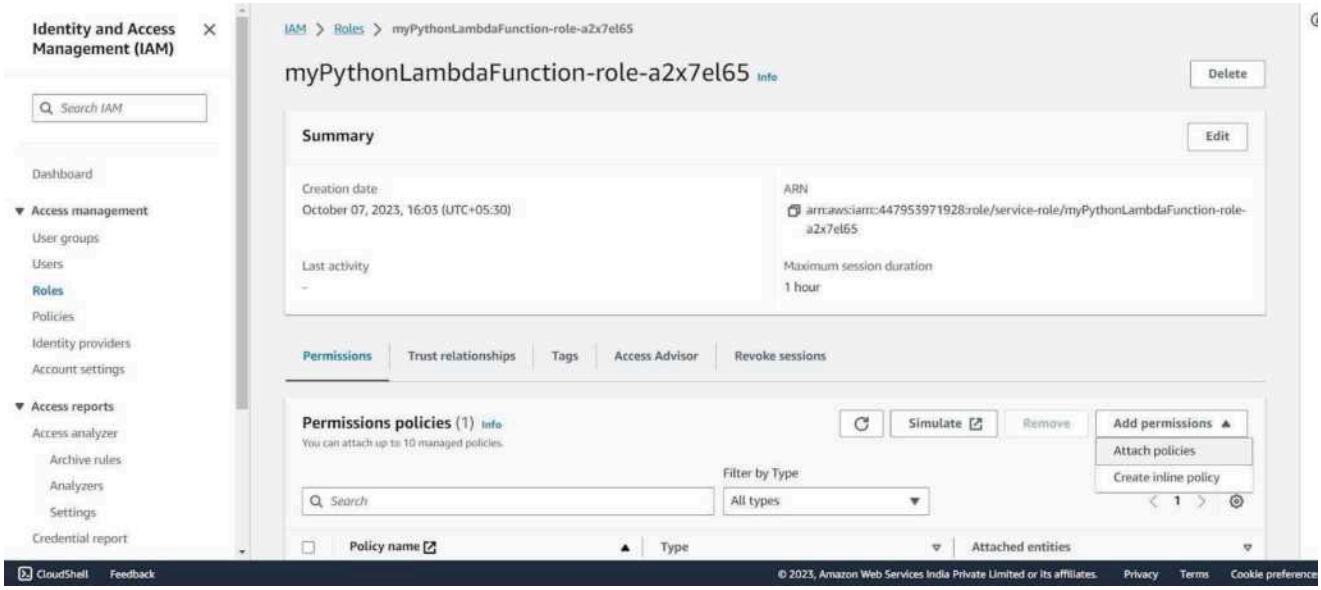
Adv. DevOps Exp. 12

Name-Manav Punjabi
Class-D15A
Roll No-45

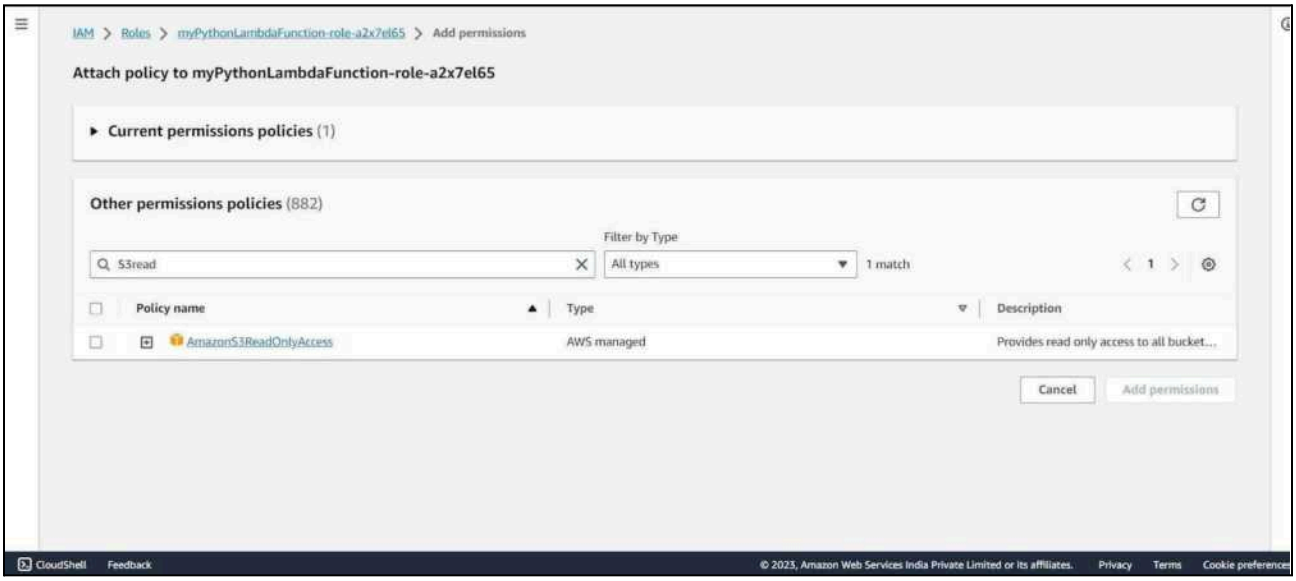
Step 1: Open up the IAM Console and under Roles, choose the Role we previously created for the Python Lambda Function (You can find your role name configuration of your Lambda function).



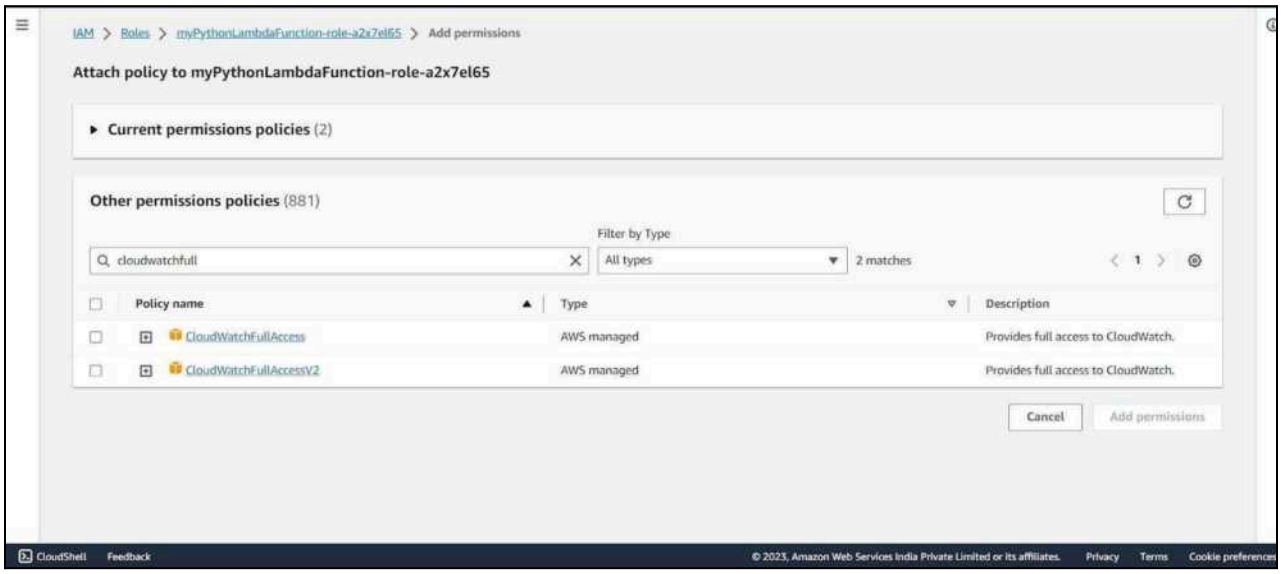
Step 2: Under Attach Policies, add S3-ReadOnly and CloudWatchFull permissions to this role.



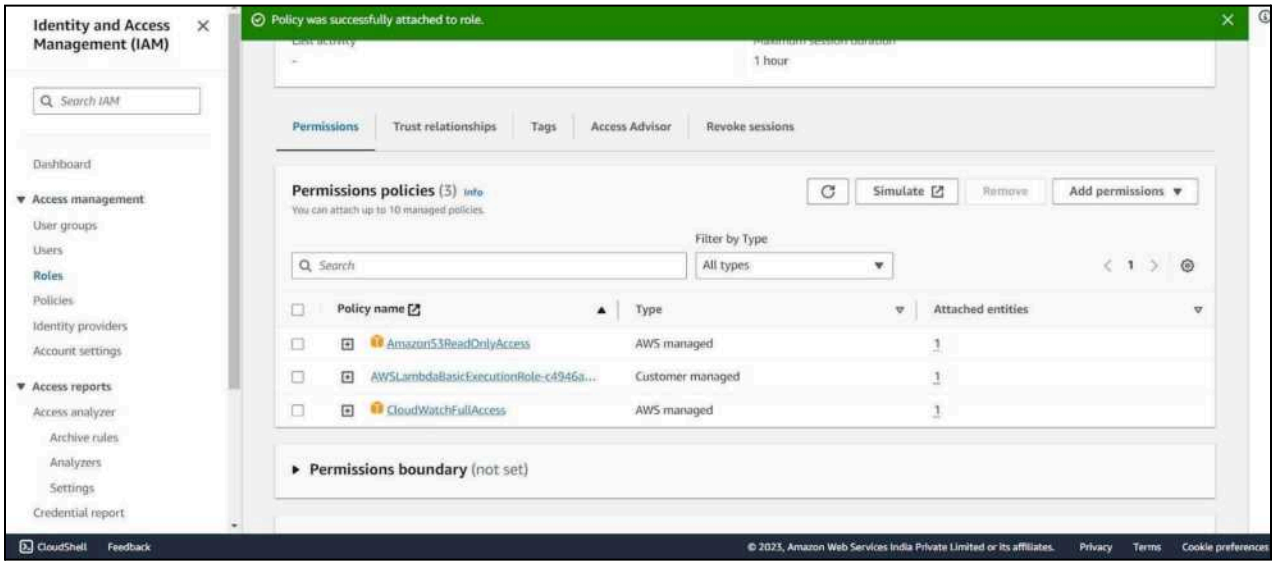
S3-ReadOnly

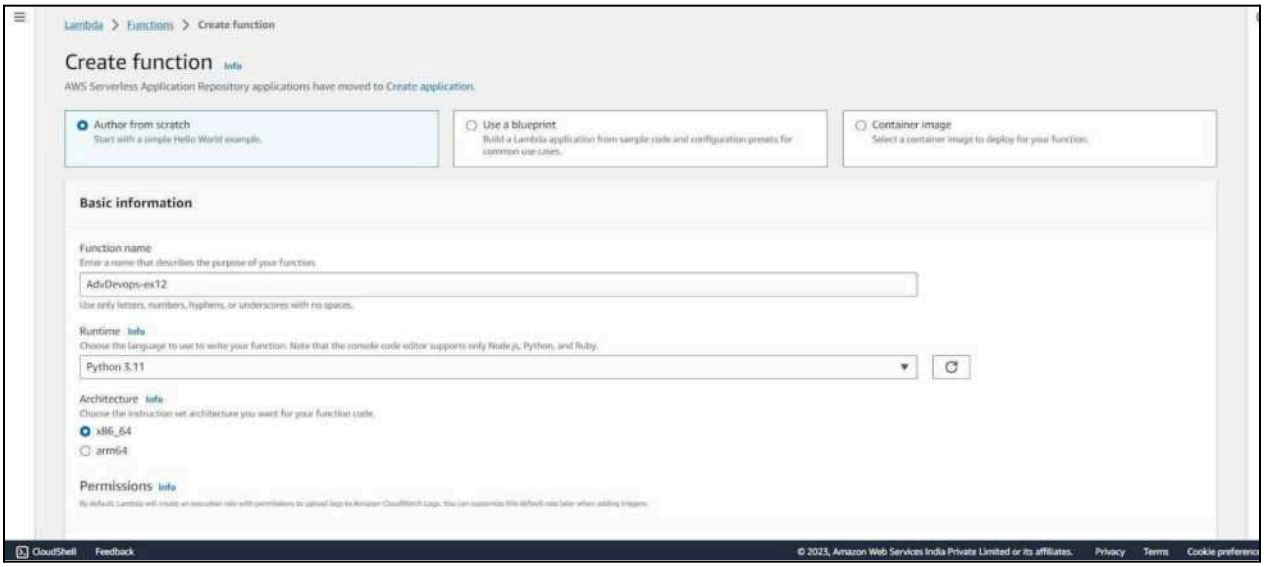


CloudWatchFull



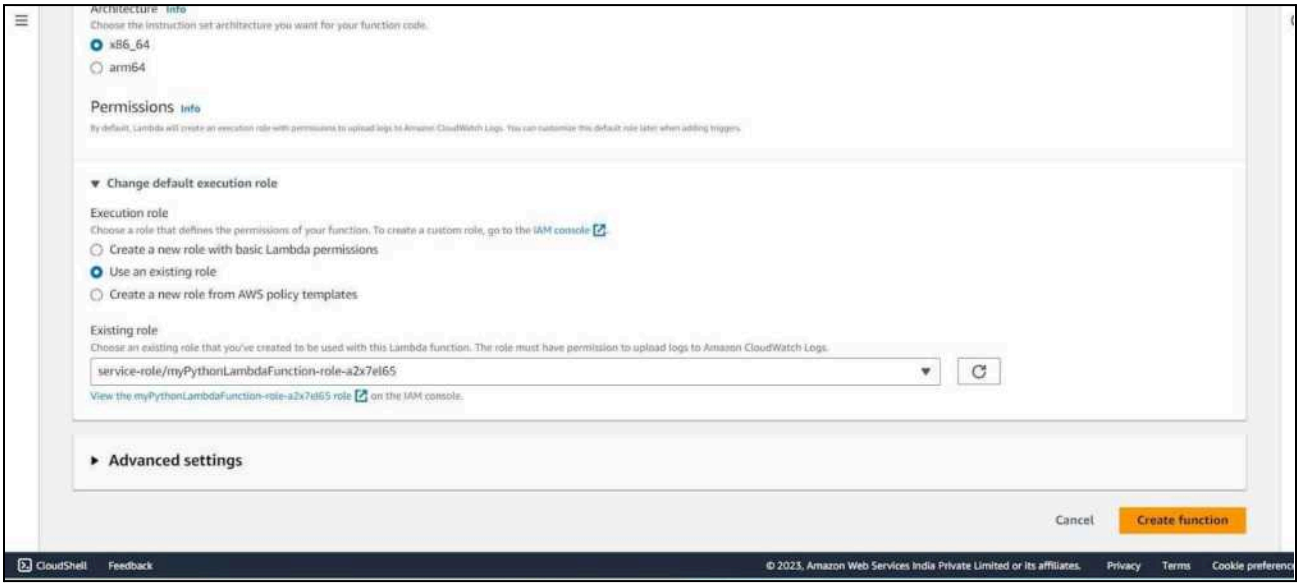
After successful attachment of policy you will see something like this you will be able to see the updated policies.



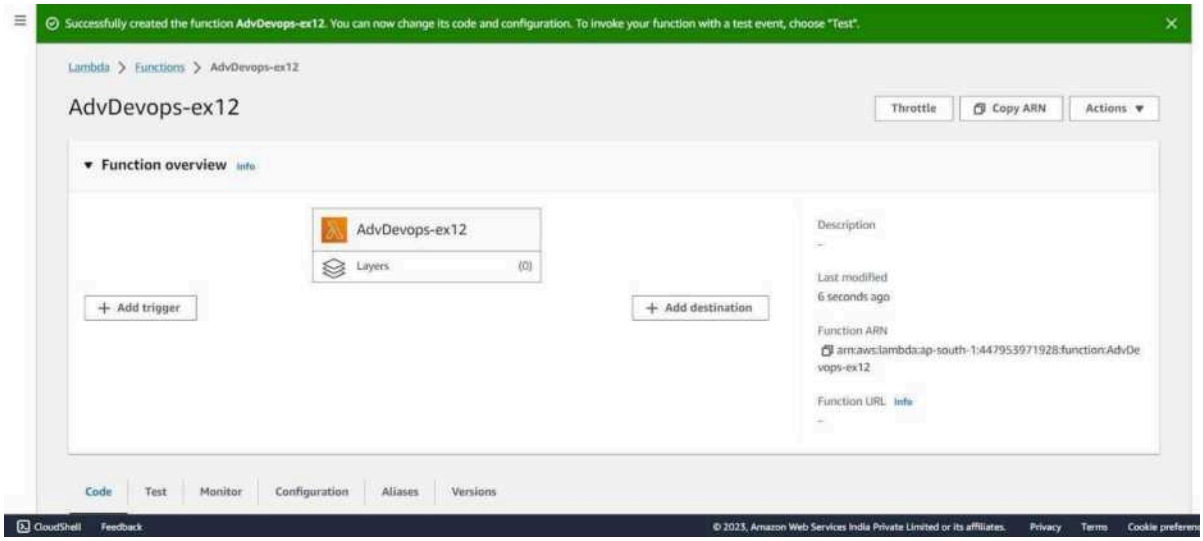


Step 3: Open up AWS Lambda and create a new Python function.

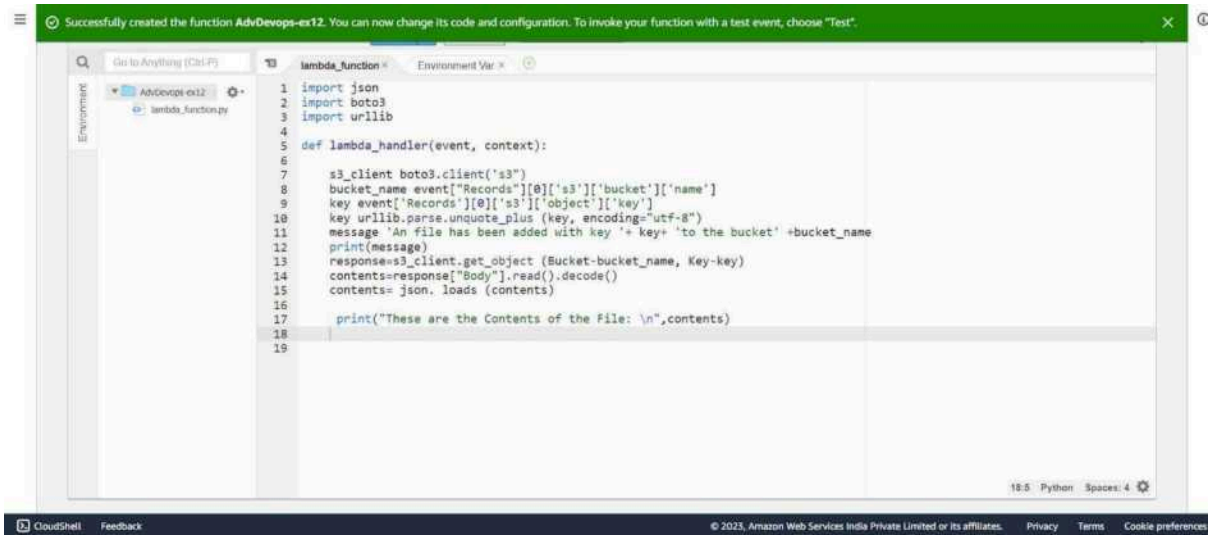
Under Execution Role, choose the existing role, then select the one which was previously created and to which we just added permissions.



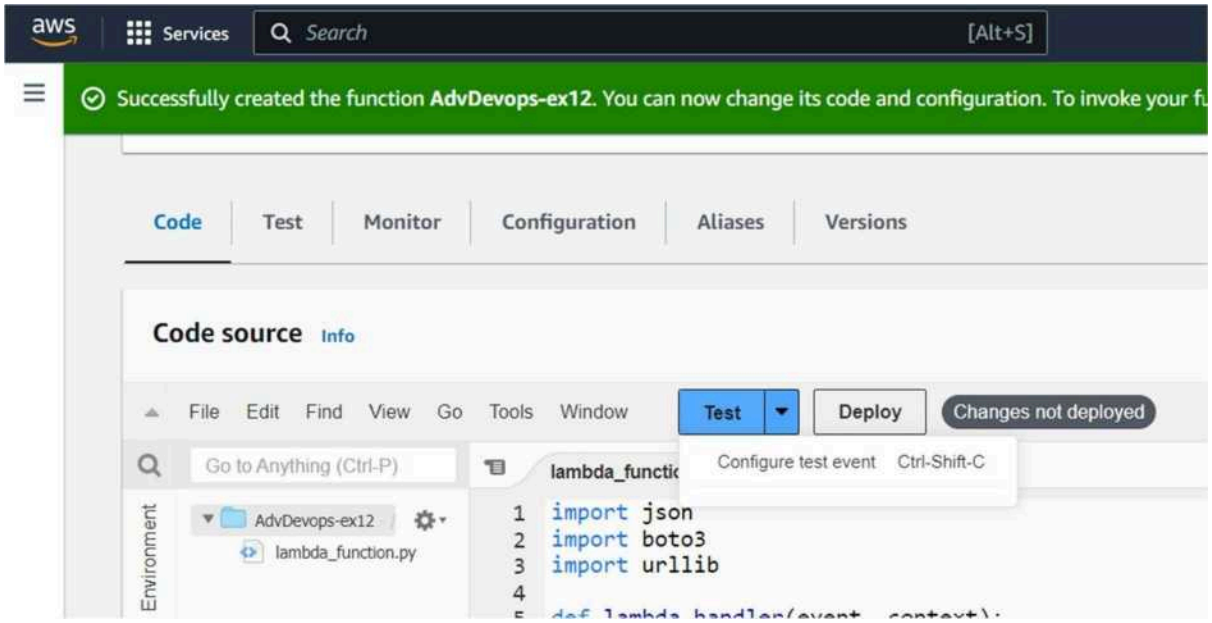
Step 4: The function is up and running.

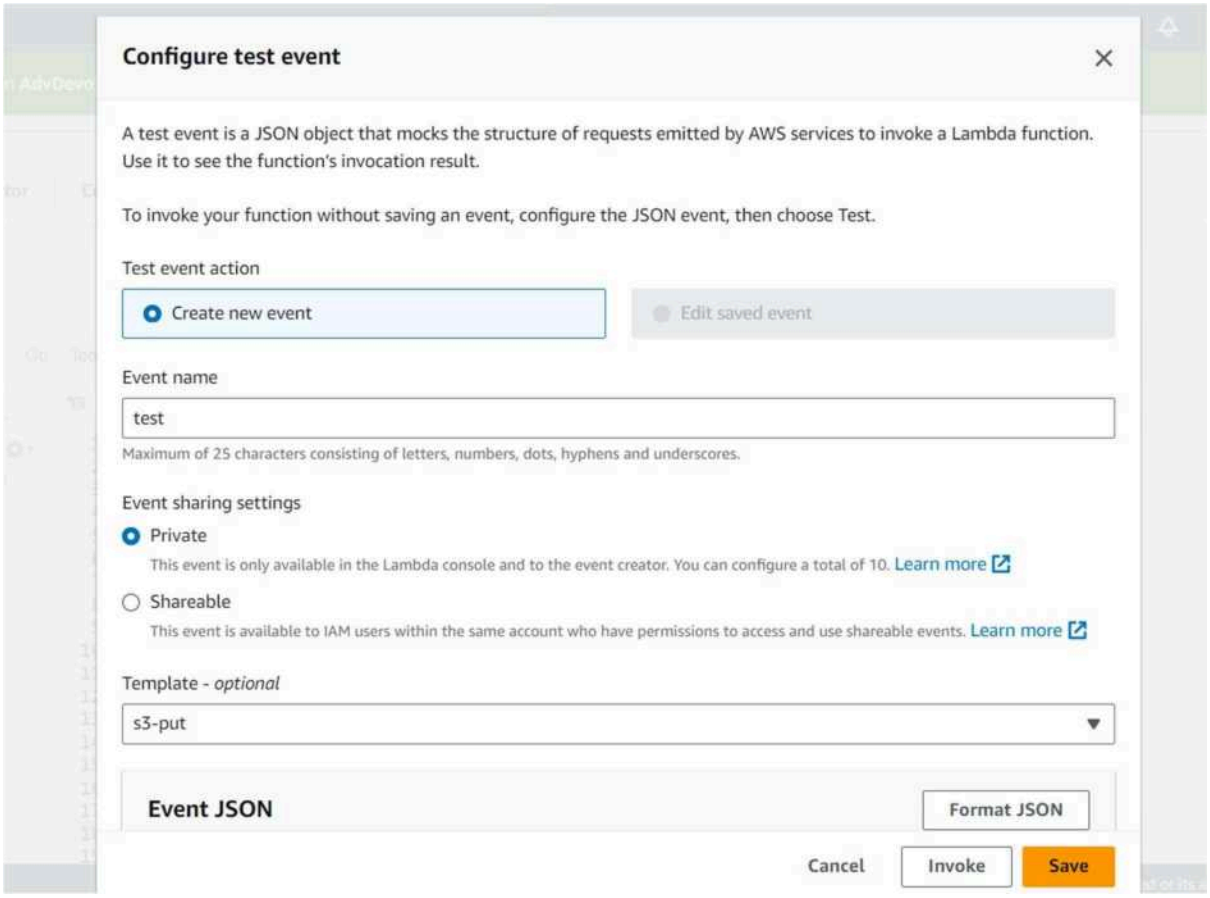


Step 5: Make the following changes to the function and click on the deploy button. This code basically logs a message and logs the contents of a JSON file which is uploaded to an S3 Bucket and then deploy the code.



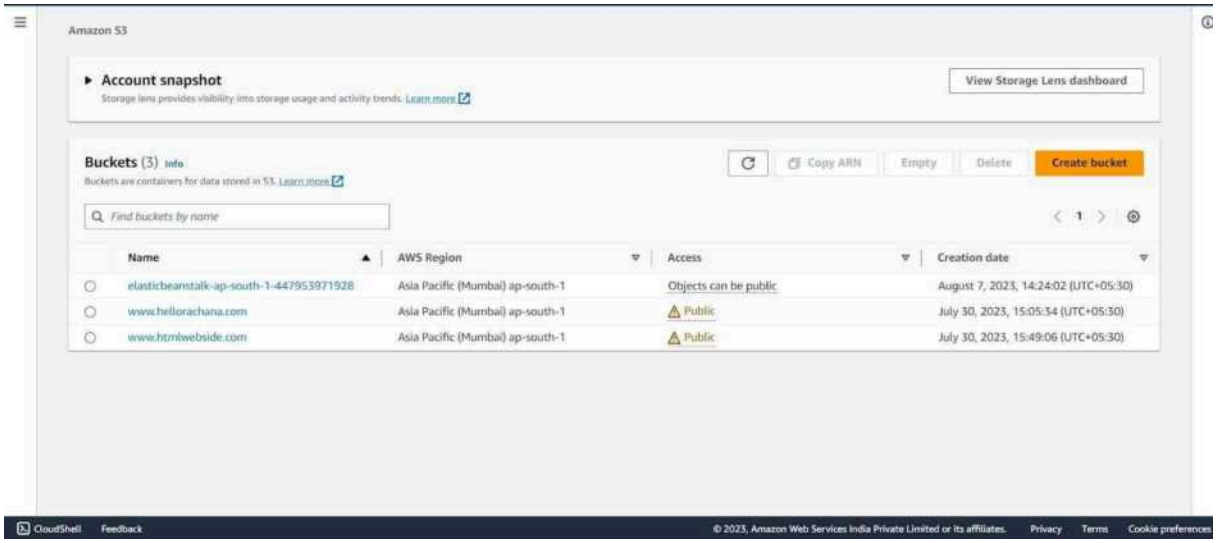
Step 6: Click on Test and choose the ‘S3 Put’ Template.



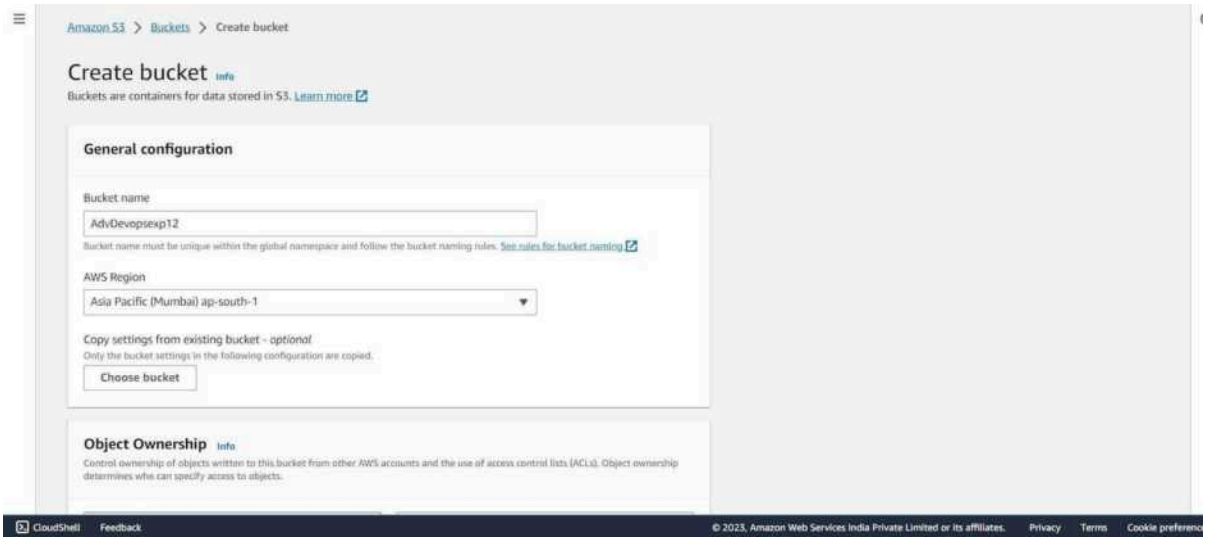


And Save it.

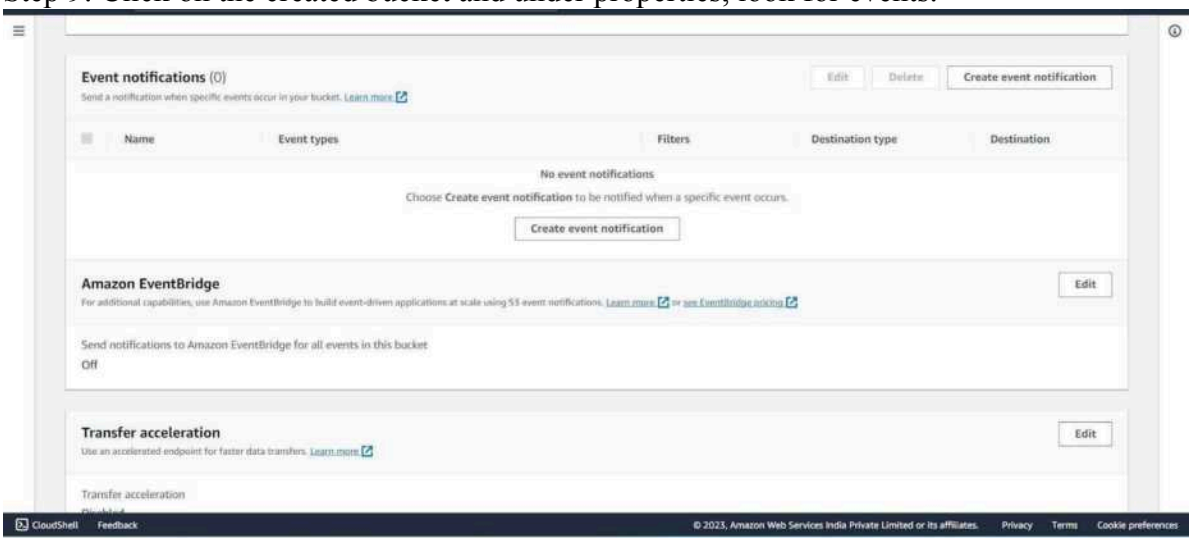
Step 7: Open up the S3 Console and create a new bucket.



Step 8: With all general settings, create the bucket in the same region as the function.

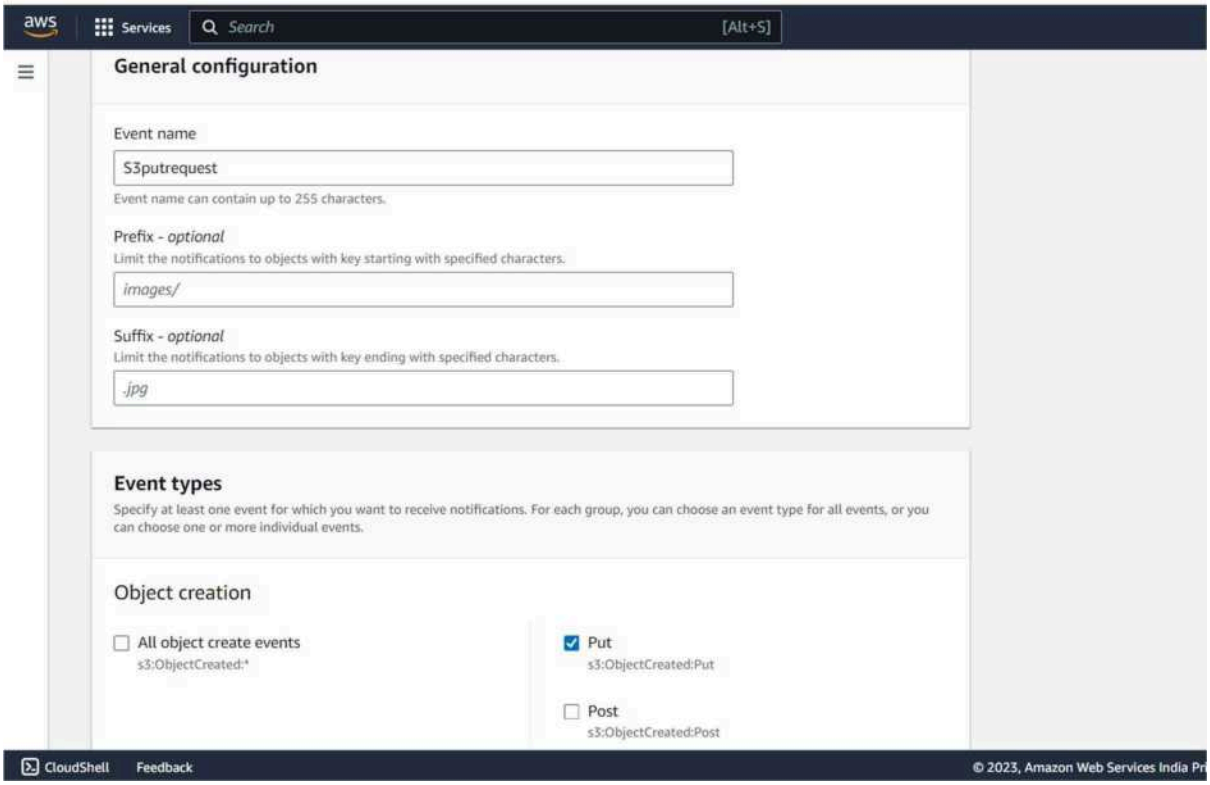


Step 9: Click on the created bucket and under properties, look for events.

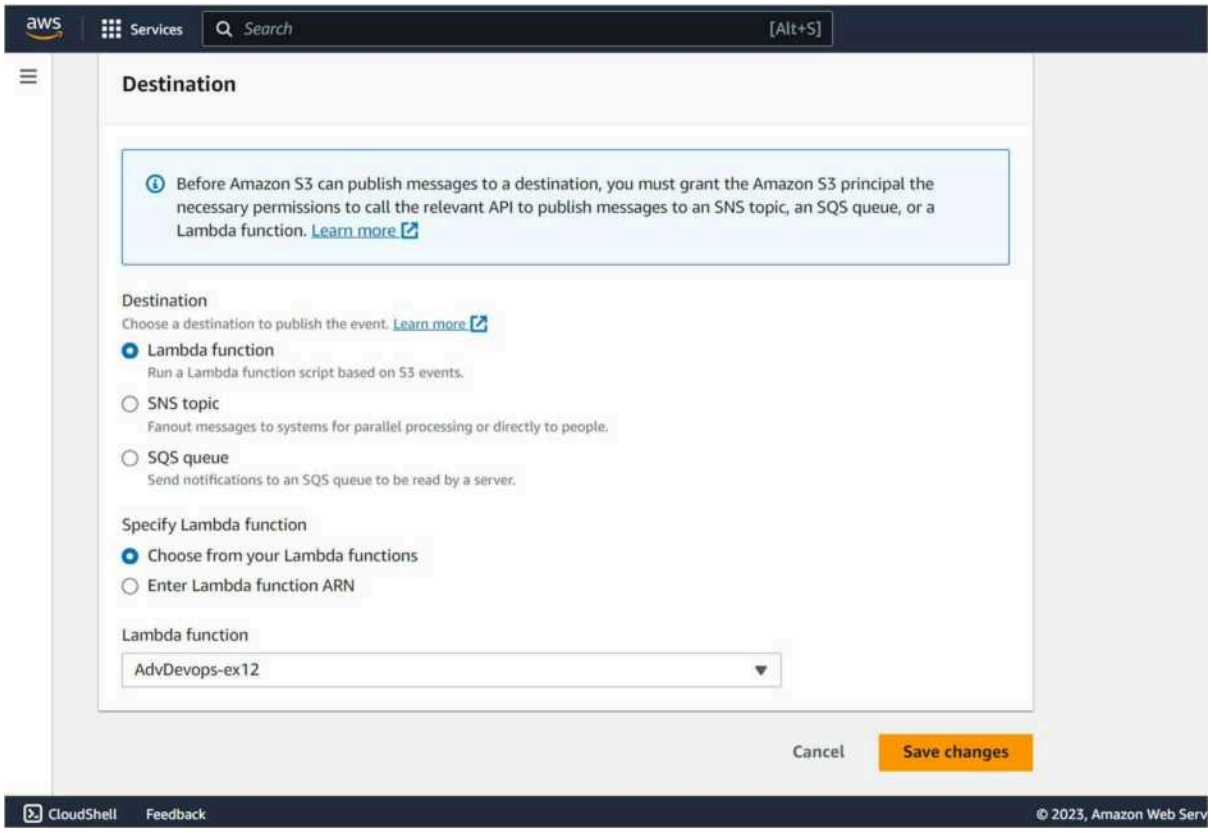


Click on Create Event Notification.

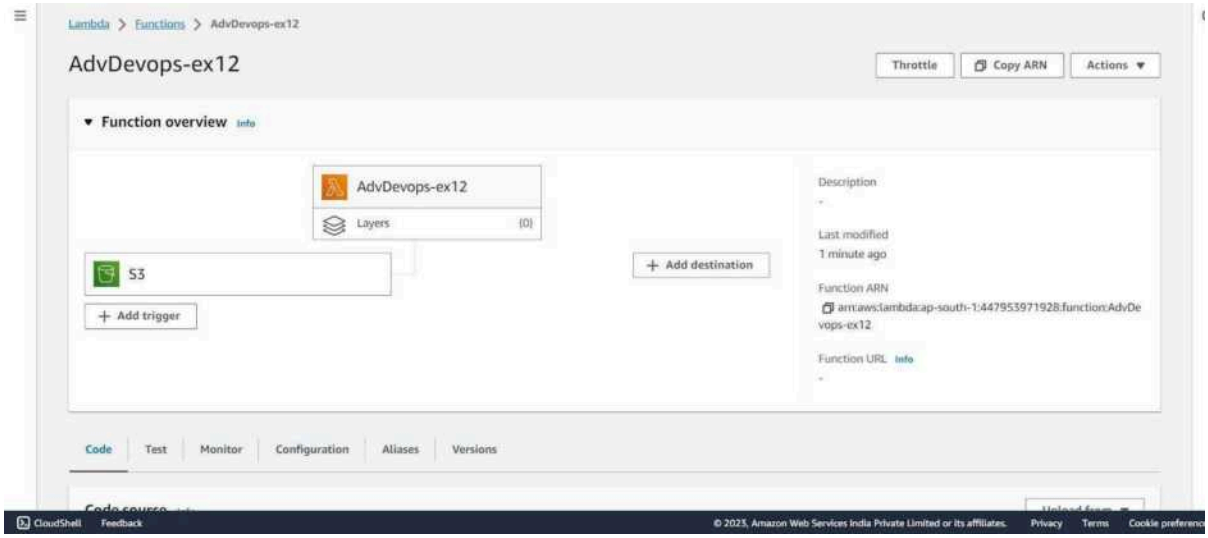
Step 10: Mention an event name and check Put under event types.



Choose Lambda function as destination and choose your lambda function and save the changes.

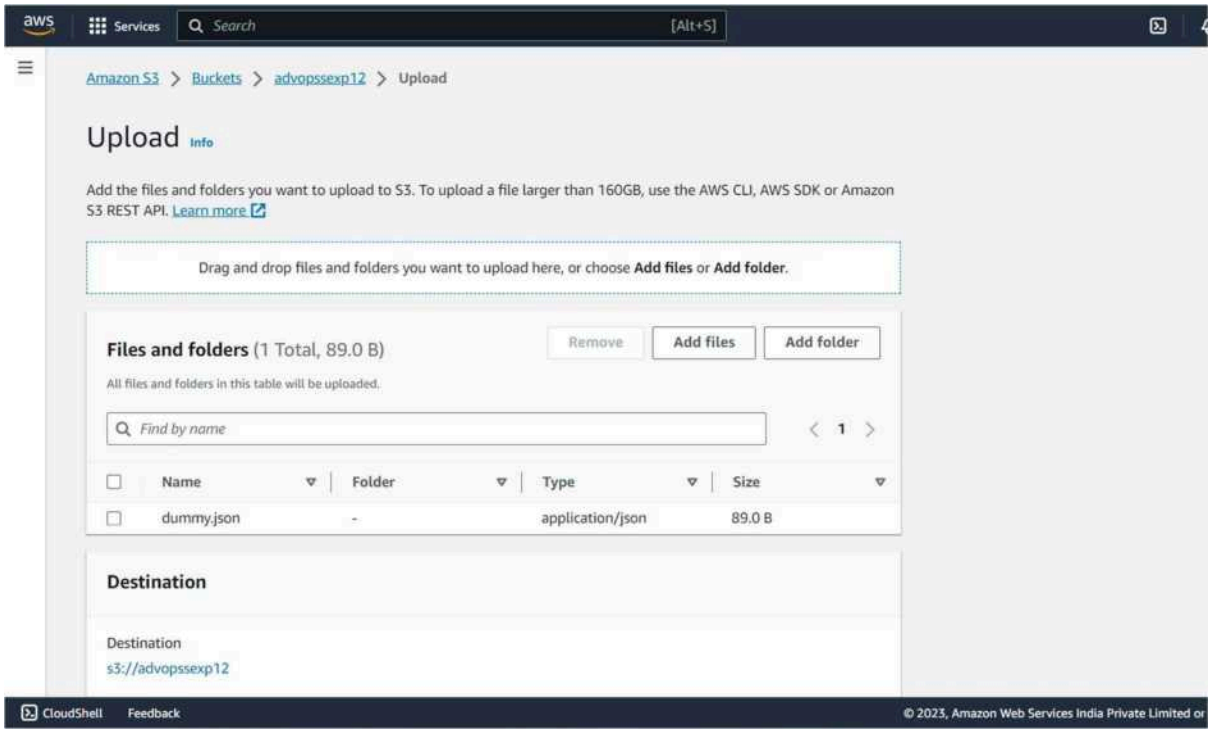


Step 11: Refresh the Lambda function console and you should be able to see an S3 Trigger in the overview.

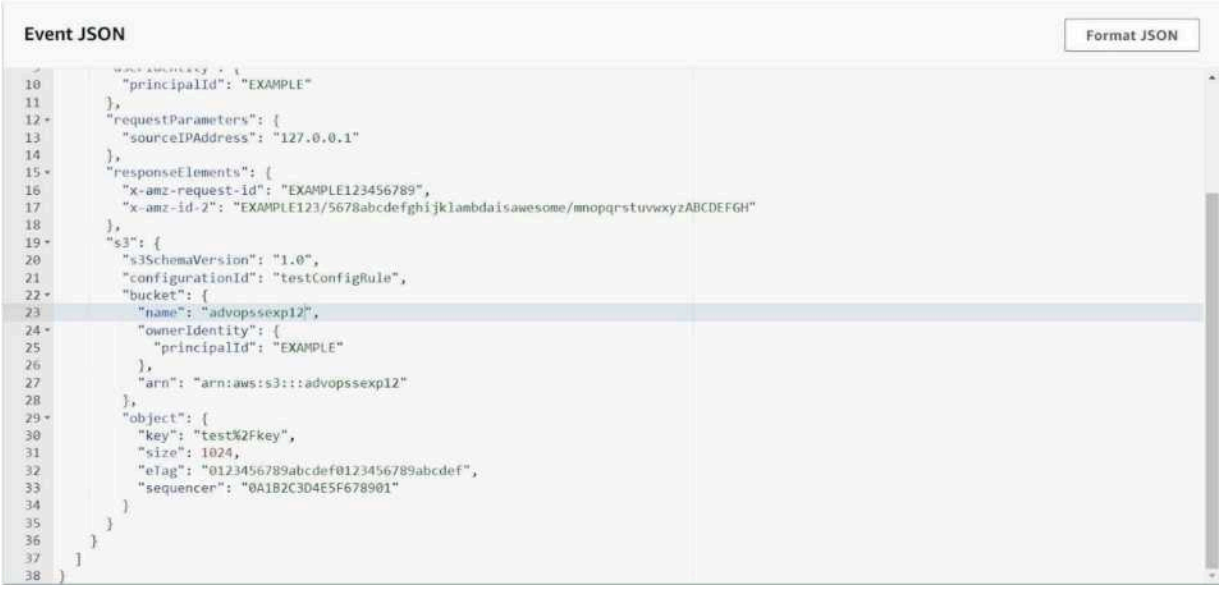


Step 12: Now, create a dummy JSON file locally.

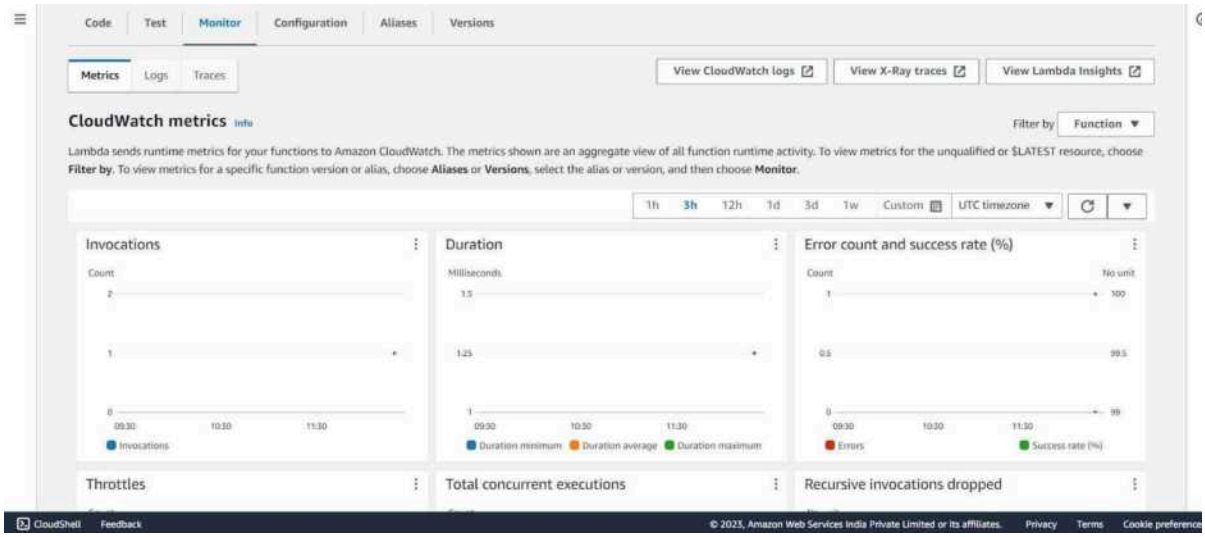
Step 13: Go back to your S3 Bucket and click on Add Files to upload a new file.
Step 14: Select the dummy data file from your computer and click Upload.



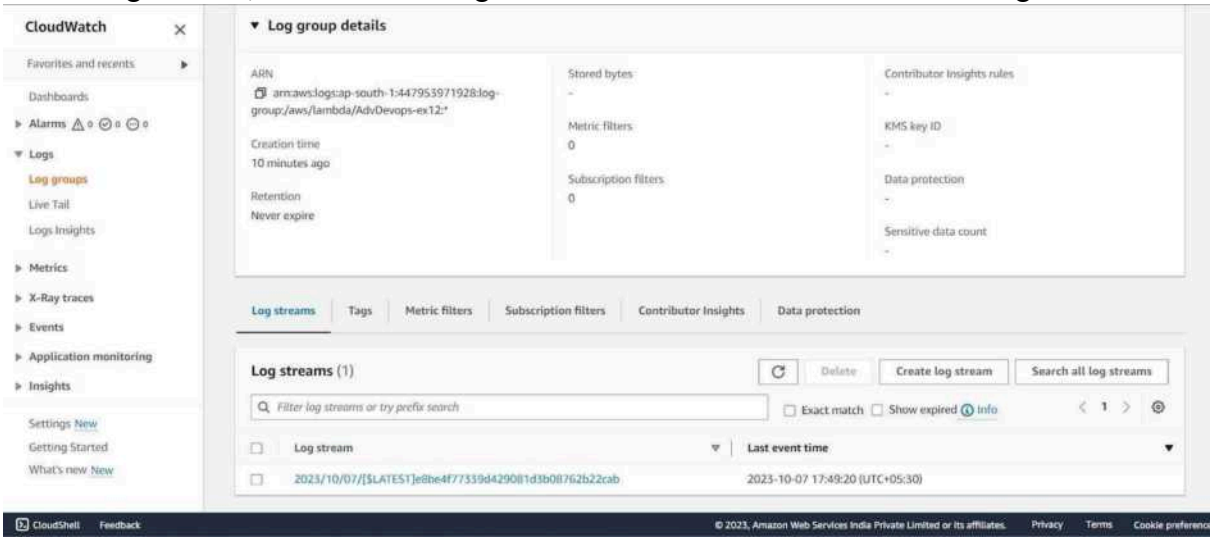
Step 15: After this make the necessary changes in the Test configuration file which we created it previously by replacing the Bucket Name and the ARN of Bucket.



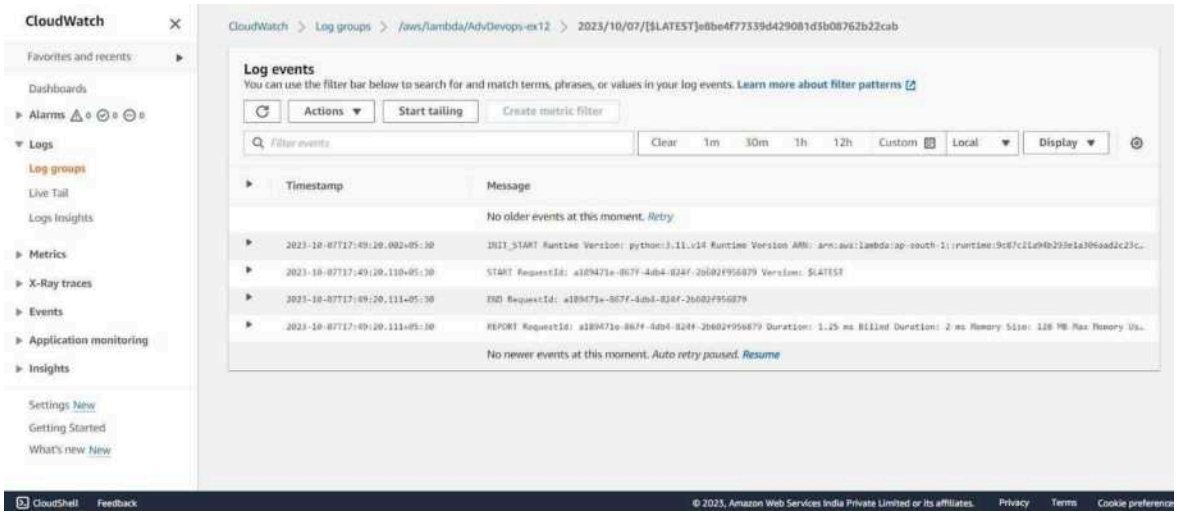
Step 16: Go back to your Lambda function , Refresh it and check the Monitor tab.



Under Log streams, click on View logs in Cloudwatch to check the Function logs.



Step 17: Click on this log Stream that was created to view what was logged by your function.



Conclusion: Thus, we have created a Lambda function which logs “An Image has been added” once you add an object to a specific bucket in S3.

Manav Punjabi
D15A - 45

04/05

Assignment - 1 Ad-Devops

Q.1] Use S3 bucket and host video streaming.

Step 1. Log in to AWS code
- Go to AWS Management console.
- Enter your login credentials.

Step 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 (like "my-video streaming-bucket")
- Choose a region (closer to your audience)
- Scroll down and uncheck Block all public acces
- confirm by checking the acknowledgment box.
- Click create bucket.

Step 3: Upload your video file to S3
- Click on your newly created bucket..
- click the Upload button.
- Add your video file from your computer.
- Click Upload to start the upload process.

Step 4: Set Permissions for Public Acess.
- once the video file is uploaded, you need to make it Publicly accessible.
- Select your video file in the S3 bucket.
- Click the Actions dropdown and choose Make Public.

• confirm the action by clicking make public ag

Step 5. Get the video URL:
• After making the file public, click on the video fi
• you will see a URL for the video under object URL. This is the effect direct link to your object.
• Copy this URL.

Conclusion:- While solving this i'm get an error during bucket Policies.
Solved this error using this steps:-

Steps 1: Go to Permissions Tab of your Bucket:
• Click on the Permissions tab of your bucket (not the individual file.)

Step 2: Edit Bucket Policy:
• Click on Bucket Policy.
• Add the following Policy (replace your-bucket-name with your actual bucket name):

```
json
Copy code.
{
"version": "2012-10-17",
"Statement": [
{
"Sid": "PublicReadGetObject",
"Effect": "Allow",
```

```
"Principal" : "*",
"Action" : "s3:GetObject",
"Resource" : "arn:aws:s3:::your-bucket-name/*"
  }
 }
}
```

Step3: Save the Policy.

final step : Testing Public Access.

- once you've made your video Public, copy the object URL (you'll find it in video file's)
- Paste that URL into your web browser to see if the video plays.

Q.2] Discuss BMW and Hot Star case studies using AWS.

⇒ BMW and Hostar case studies using AWS.

BMW:

1. Connected car features:
   - Remote diagnostics for car health
   - over-the-air software updates.
   - Real-time traffic information.

2. Improving customer experience: They use AWS to enhance how customers interact with their cars. for example, BMW has added voice-activitated affistants and personalized suggestions using services like Amazon Lex and Amazon Sagemaker.

3. Better manufacturing: BMW used AWS to improve its manufacturing. By analyzing data from machines, they can find problems easly and increase efficiency.

HotStar:

1. Manage Huge traffic: During big events like the Indian Premier league (IPL), many people what at once.

2. Quality Streaming: AWS services like Amazon kinesis and Amazon Elastic Transcoder ensure that users get high-quality streaming.

3. Personalized Recommendations: Hotstar used AW's machine learning tools to suggest content to users, making...

their veiwing experience more enjoyable.

key AWS Services used:
- Compute: Amazon Ec2, lambda
- Storage: Amazon s3, Efs
- Database: Amazon RDS, DynamoDB
- Networking: Amazon vpc, Direct connect
- Analytic: Amazon kinesis, Redshift
- machine learning: Sagemaker Rekognition.

AWS helps both companies innovate and deliver top-notch services to their customers.

Q.3] why kubernetes and advantages and disadvantages of kubernates. Explain how adidas used kubernates.

⇒ why kubernates?
kubernates is polular because it simplifies the management of containerized applications. It automates tasks such as deployment, scaling and monitoring, making it easier for organizations to manage their applications in a dobd environment.

Advantages of kubes nets:
1. Portability: Applications can be moved easily between different environments without major changes.

2. Scalability: kubernets can automatically scale applications up or down based on traffic and demand.

3. Reliability: It features self-healing capabilities, meaning it can restart failed containers and balance workloads to ensure high availability.

4. Efficiency: It optimzed resource usage by running containers on a single host, improving overall efficiency.

Disadvantages of kubernetes:

1. Complexity: It can be complicated to set up and manage, especiany for those new to containes technology.

2. Steep learning curve: Required time and knowledge to fully understand and utilize its feature.

3. Resource intensive: It may require more computing resources than simples solutions, which can increase costs.

4. Management overhead: Required ongoing management and maintenance, which can add to operational workload.

How Adidas use kubernetes:

1. Faster Development: Streamlined deployment for quicker product launches.
2. Operational Efficiency: Automated tasks, reducing managing time and increasing reliability.
3. Scalability: Easily handled traffic spikes during peak shopping periods.

4. Encouraging innovation: flexible platform for experimenting with new technologies and ideas.

Specific use cases at adidas:

1. Microservices Architecture: Independent services for better management and deployment.
2. Continuous Delivery: Supports quick and efficient build, test, and deployment processes.
3. Hybrid Cloud Deployments: Runs apps on both on-premises and cloud for flexibility and cost savings.

Benefits: modernized IT, improved agility, and enhanced innovation, keeping adidas competitive.

Q.g What are Nagios and explain how Nagios are used in E-O services?

⇒ What is Nagios?

Nagios is an open-source monitoring tool that helps organizations keep track of their IT infrastructure including servers, network, and applications.

Key features of Nagios:

1. Monitoring: Tracks the performance and availability of servers, applications, and network devices.
2. Alerts: Sends notifications via email or SMS when problems occur, so teams can respond quickly.
3. Reporting: offers detailed reports system performance and uptime, helping identify trends and areas for improvement.

How Nagios used in E-Services:

1. Infrastructure Monitoring: Tracks servers and databases, alerting the IT team if any issue arise.

2. Service Availability: Monitors web services and APIs for accessibility, enabling quick response to outages.

3. Performance Metrics: Collects data on system performance to optimize resources and enhance user experience.

4. Incident Management: Integrates with tools to streamline issue resolution, minimizing downtime.

5. User Experience Monitoring: Checks applications performance from the User's perspective to ensure smooth operation.

Benefits: High availability, improved reliability, and better user experience, essential for maintaining service quality and customer satisfaction.

Manav Punjabi
D15A - 45 Batch - C

AdvDevOps Assignment - 2

Q.1] Create a REST API with Serverless framework

Ans Creating REST API with Serverless framework is an efficient way to deploy Serverless applications that can scale automatically without managing servers.

i) Serverless framework: A powerful tool that automates deployment of services and Serverless applications across various cloud providers such as AWS, Azure and Google cloud.

ii) Serverless architecture: This design model allows developers to build application without worrying about underlying infrastructure, enabling focus on code and business logic.

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

Steps for creating REST API for ~~Serviced~~ framework:

1) Install Serverless framework:
You start by installing Serverless framework globally using node package manager. This allows you to manage Serverless applications directly from your terminal.

2) Creating a Node.js Serverless Project:
A directory is created by your Project where you will initialize a Serverless service. This service will house all your lambda function, configuration and cloud resources.

3) Project Structure:
The Project staffing creates essential files like handler.js (which contains code for lambda functions) and serverless.yml.

4) Create a REST API Resource:
In the serverless.yml file you define function that handles Post request of HTTP

5) Deploy the service:
with the 'sls deploy' command serverless framework packages your application, uploads necessary resources

6) Testing the API: once deployed you can test Rest API using tools like curl or Postman.

7) Storing data in Dynamic: To store submitted candidate data you integrated AWS.

8) Adding more functionalities: Adding functionalities like 'list all candidate, get candidate by ID'

9) AWS IAM permissions
you need to ensure that serverless framework is given right permissions to interact with AWS resources the Dynamic.

10) Monitoring and maintaince
After deployment serverless framework is given right permissions to interact with AWS resources like Dynamic.

8.2) Case study for sonarable
creating your own Profile in sonarable for testing Project quality. we sonarable to analyse your Github code. Install sonarlin in your Java intellij ide and analyze Java code. Analyze Python Project with sonarable.

→ sonarable is an open source platform used for continuous inspection of quality. It detects bugs, code smells and security vulnerability in project across various programming languages.

1) Profile creation in sonarable.
Quality Profiles in sonarable are essential configuration that define rules applied during code analysis. Each Project has a quality Profile for every supported language with default being 'sonar way' Profile comes built in for all languages. Custom Profiles can be created by copying or extending existing ones. Copying creates an independent.

Profile, while extending inroast races for Parent Profile and rejects future changes automatically you can activate or deactivate rules, Prioritize certain rules and configure Parameters to tailor Profile to specific Profiles. Permissions to manage quality Profile are restricted to users with administrative Privileges. Sonar allows for the comparision of two Profiles to check for differences in activated rules and users can track changes via event tag.

2) using sonarcloud to analyze Github
Sonarcloud is cloud-based code Post c
Sonarqube that integrates directly with
Github - BitBucket, Azure and Github
respositioned. To get started with
sonarcloud via Github signup via son
cloud Product page and connect
your Github organization or Perso
account. once connect, sonarcloud
mirror your Github setup with each
Project corresponding to Github
repository. After setting up the
organization choose subscription plan
(free for public repo). Next, import
repositories into your sonarcloud
organization where each 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 developers detect bugs, security vulnerabilities and code smells directly in the development environment such as IntelliJ IDEA or Eclipse. To set it up, install the Sonarlint Plugin, configure the connection with Sonarcloud or Sonarqube and select the project profile to analyze Java code. This approach offers immediate feedback in code quality, promoting clean and maintainable code from beginning.

4) Analyzing Python Projects with Sonarcloud.
Sonarcloud support Python test coverage, reporting but it requires third party tool like coverage.py to generate the coverage report. To enable coverage adjust your build process so that coverage tool runs before Sonar scanner that ensures report file is saved in different path.
For setup you can use Tox, PyTest and coverage and run test. In your tox.ini include configurations for Pytest and coverage to generate coverage report in xml format. The build process can also be automated using Github Actions, which install dependencies, runs tests and invokes Sonarcloud scan.

5) Analyzing Node.JS Projects with Sonarlable. 3.
For Node.JS Project Sonarsoube can analyze
Javascript and Typescript code. Similar
the Python setup, you can configure 4.
Sonarlouble analyze Node.JS Projects by
installing the appropriate Pulgins and
using Sonar Scanner to Scan the
Projects. Sonarsoube will check the code
against Industry standard rules and
best Parties.

Q.3] In large organizations, centralized operation
teams often face many repetitive infrastruc
ure requests, causing delays. Using a Se
Service infrastructure model with Terra
can help Product means teams manage the
infrastore model with Terraform can
help Product team manage independently

key benefits of Tereform:-

1. Modularity & Reusability: Terraform
modules encapsulate standard configu
rations (database, compute resources)
for reuse across Projects.

2. Standardization: Best Practices are buil
into modules, enforcing compliance wi
organizational Policies.

3. version control:- versioning helps manage dependencies and reduce disruptions.

4. Documentation: Comprehensive documentation promotes collaboration and understanding.

Implementation steps:-

1. Identify Infrastructure components: Determine which components (security grps, load balancers) to modularize.

2. Develop Reusable modules:- Define configurations with input variables and output for integration.

3. Governance and Best Practices:- Set guidelines for module usage, versioning, and encourage team contributions.

4. Testing and validation: Implement testing frameworks to validate module functionality.

Terraform can also integrate with ticketing system like ServiceNow to automate infrastructure requests, improving workflows and reducing manual interactions. This self-service model increases agility.