NAME-Laksh.V.Sodhai

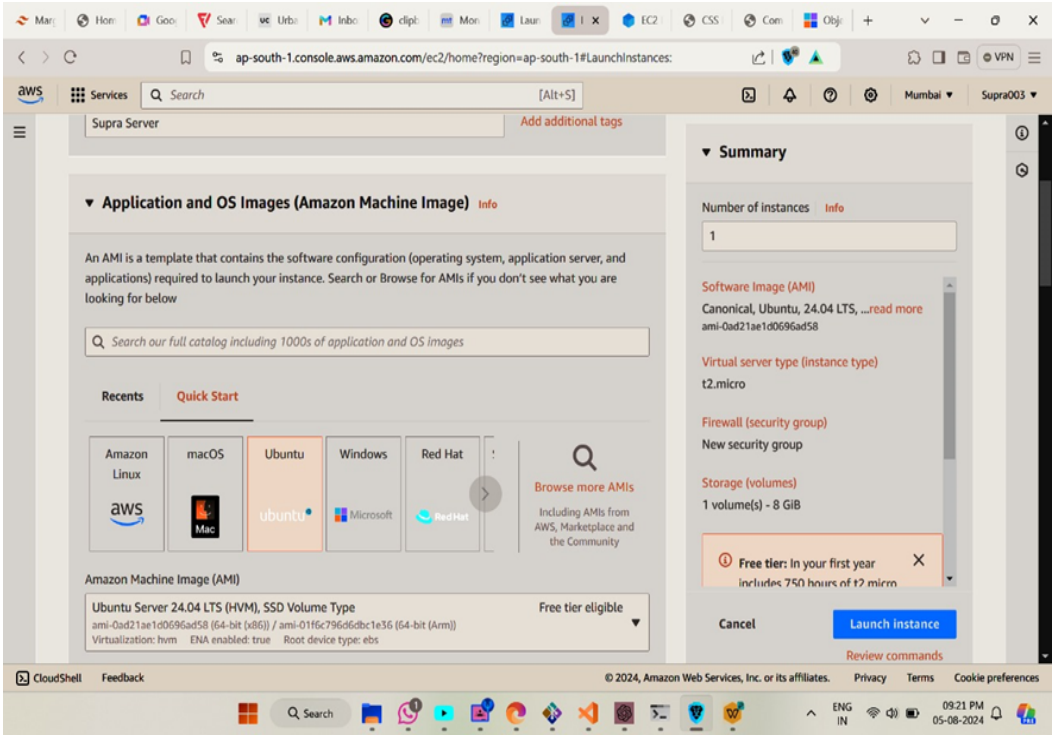
CLASS-D-15A

ROLL.NO-59

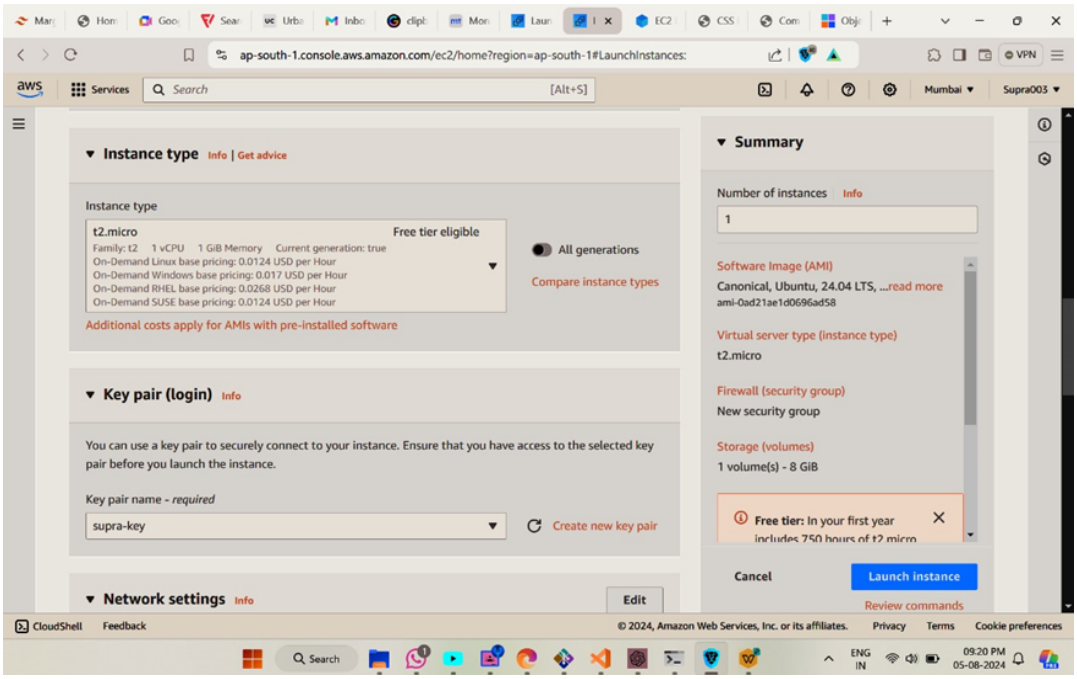
Experiment 01

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

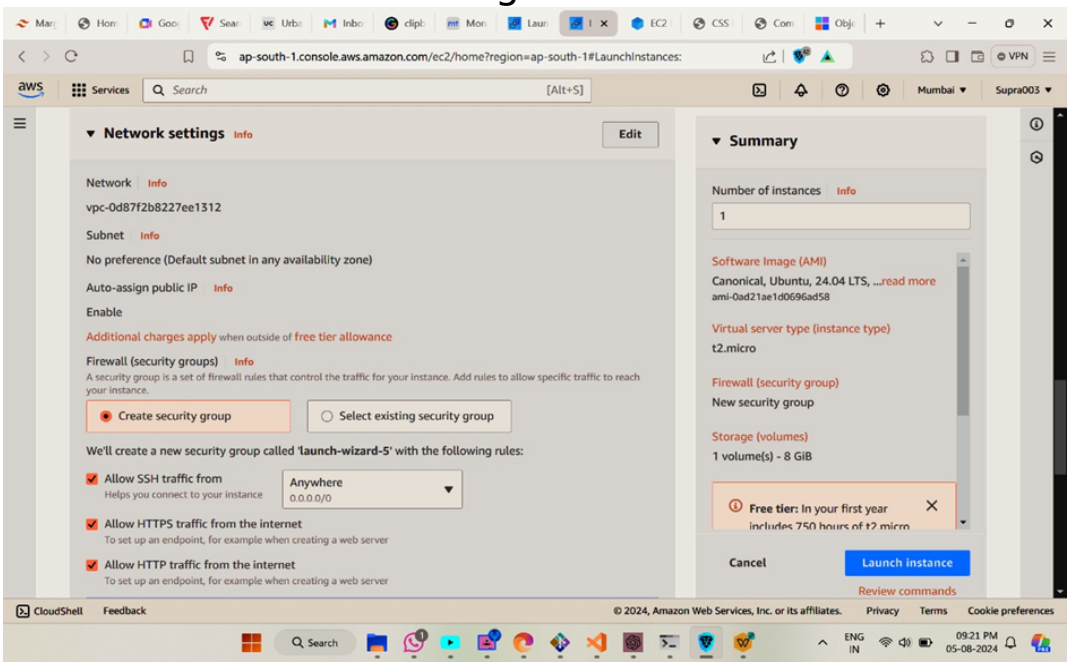
1. Give name to EC2 instance



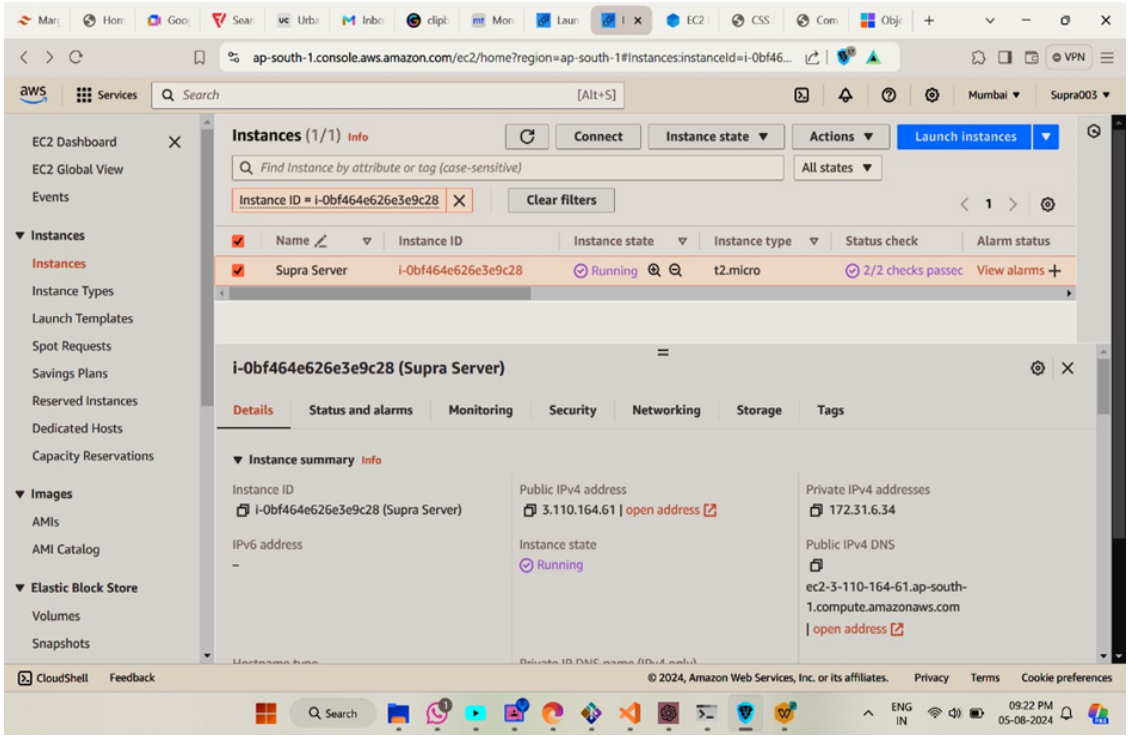
2. Create a key-pair login.



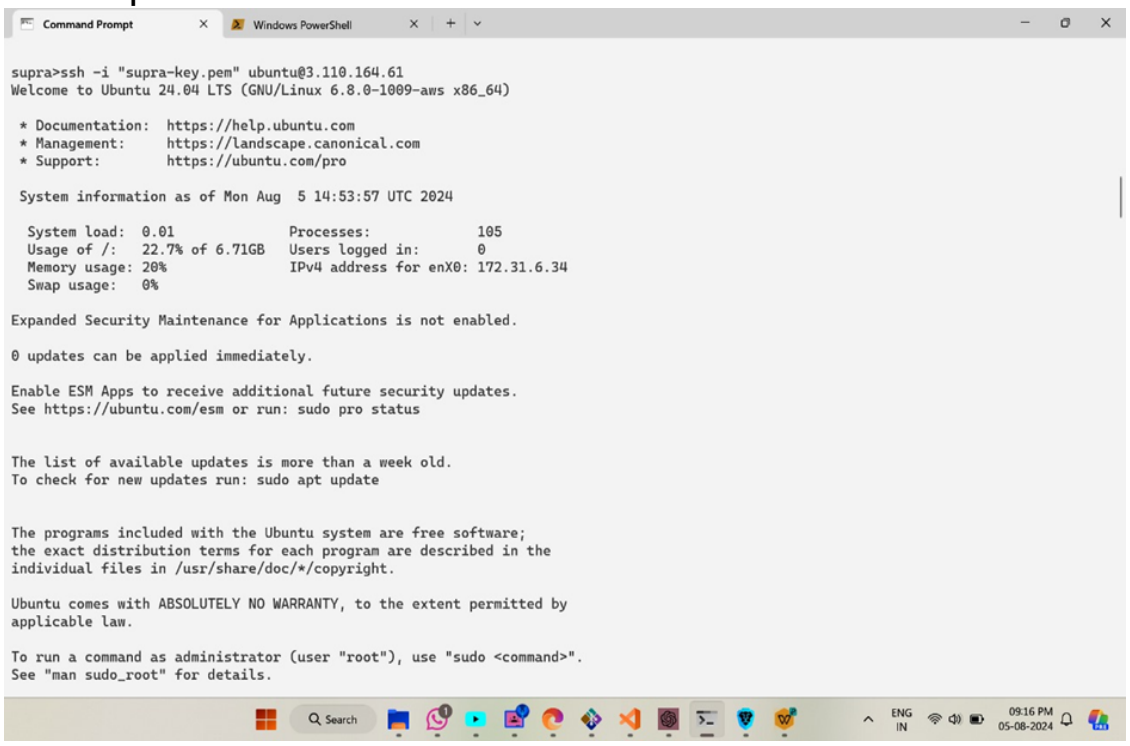
3. Edit network settings. And Launch Instance.



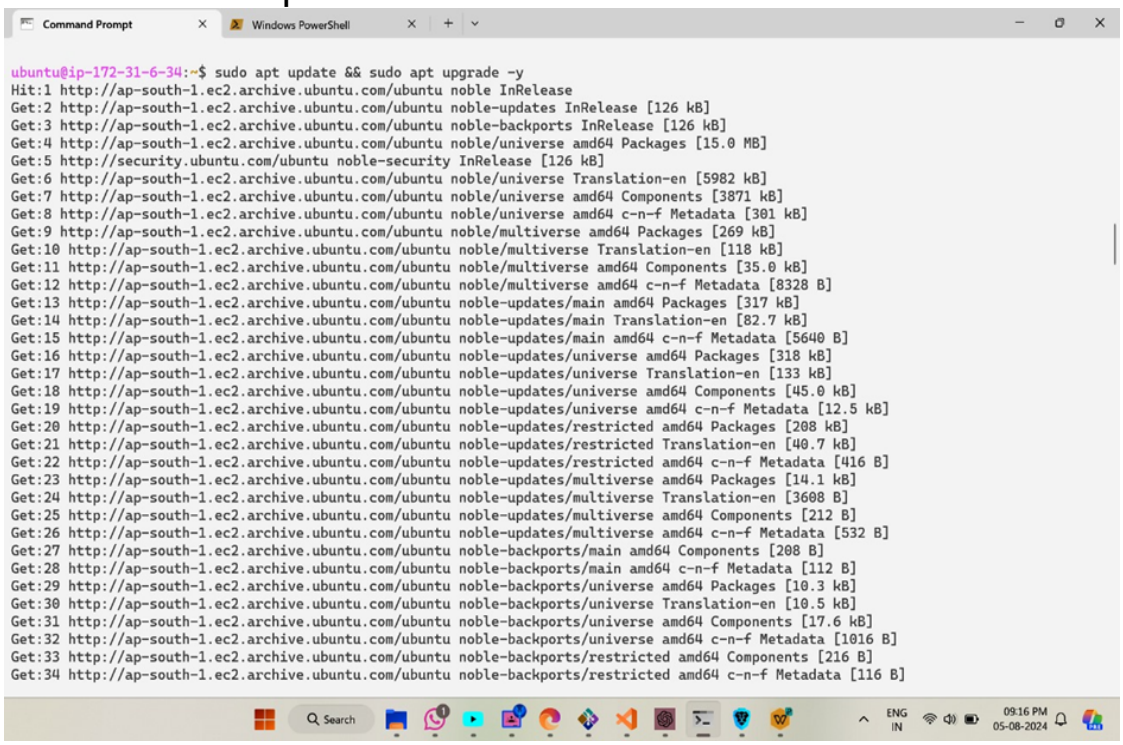
4. Instance successfully lanunched.



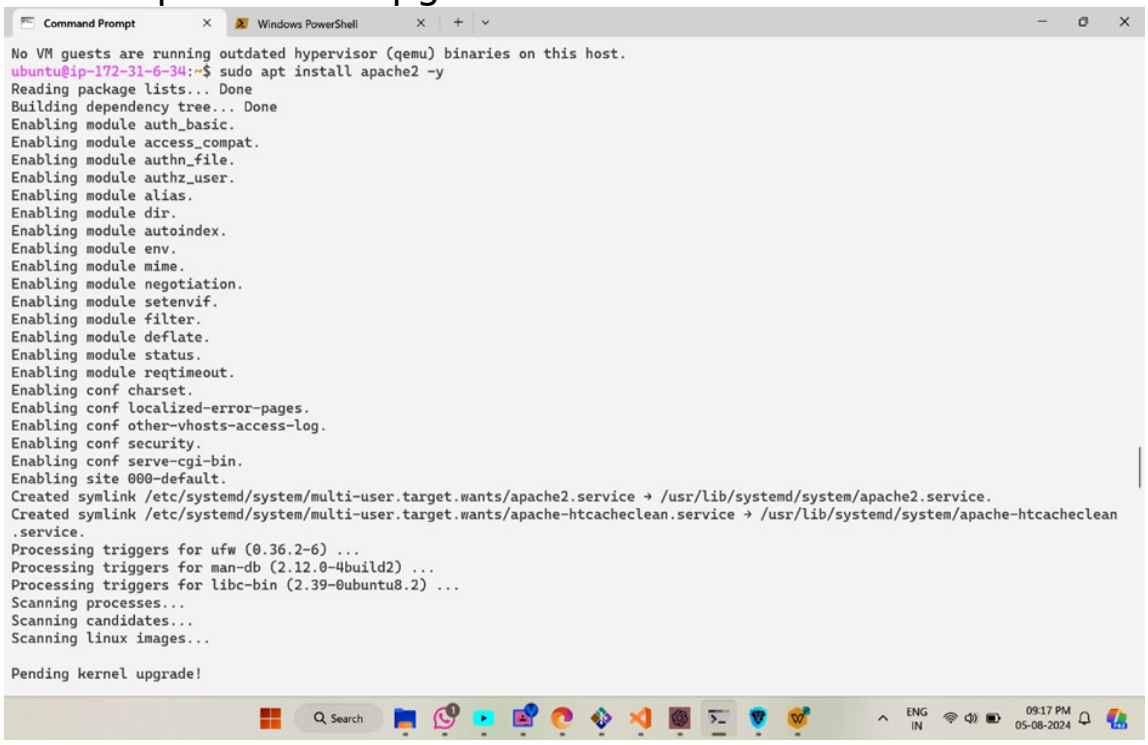
5. Create a virtual environment with key pair permission.



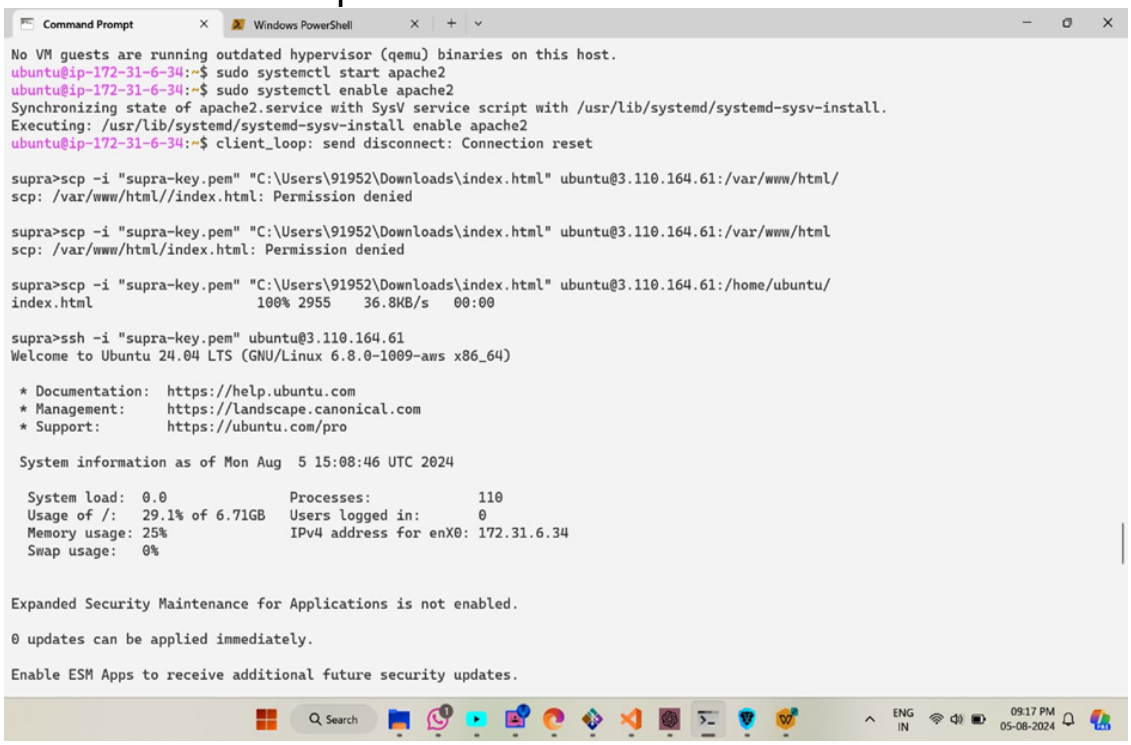
6. Install apache on virtual machine.



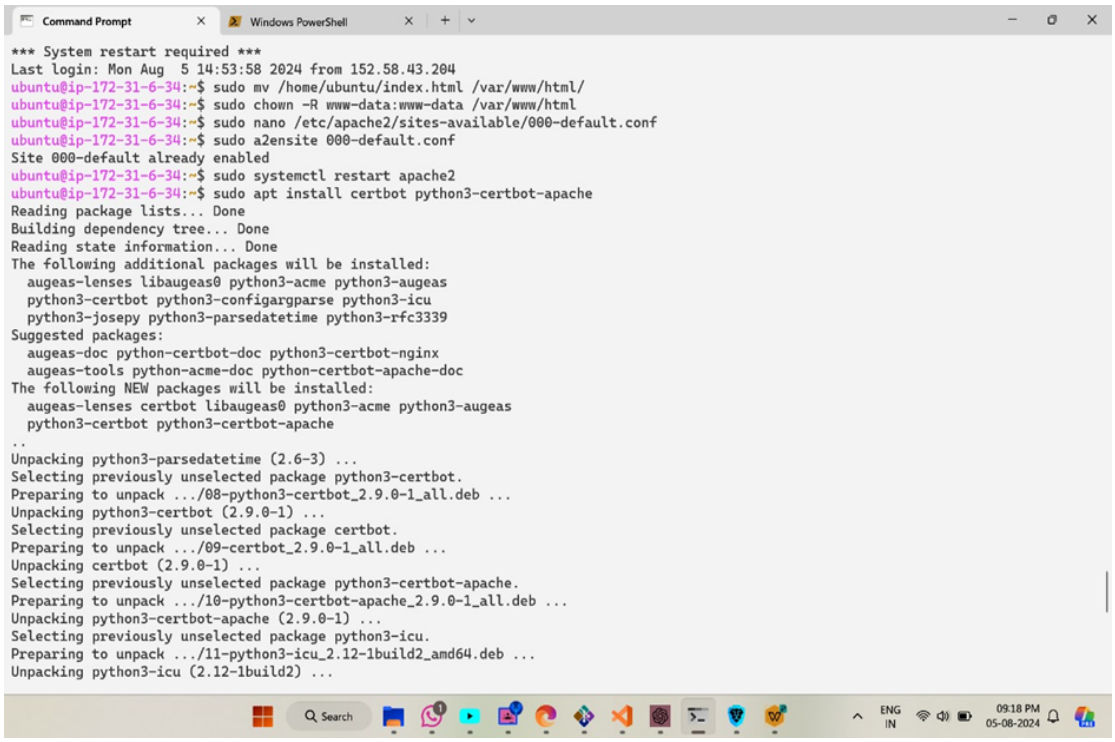
7. Update and upgrade it.



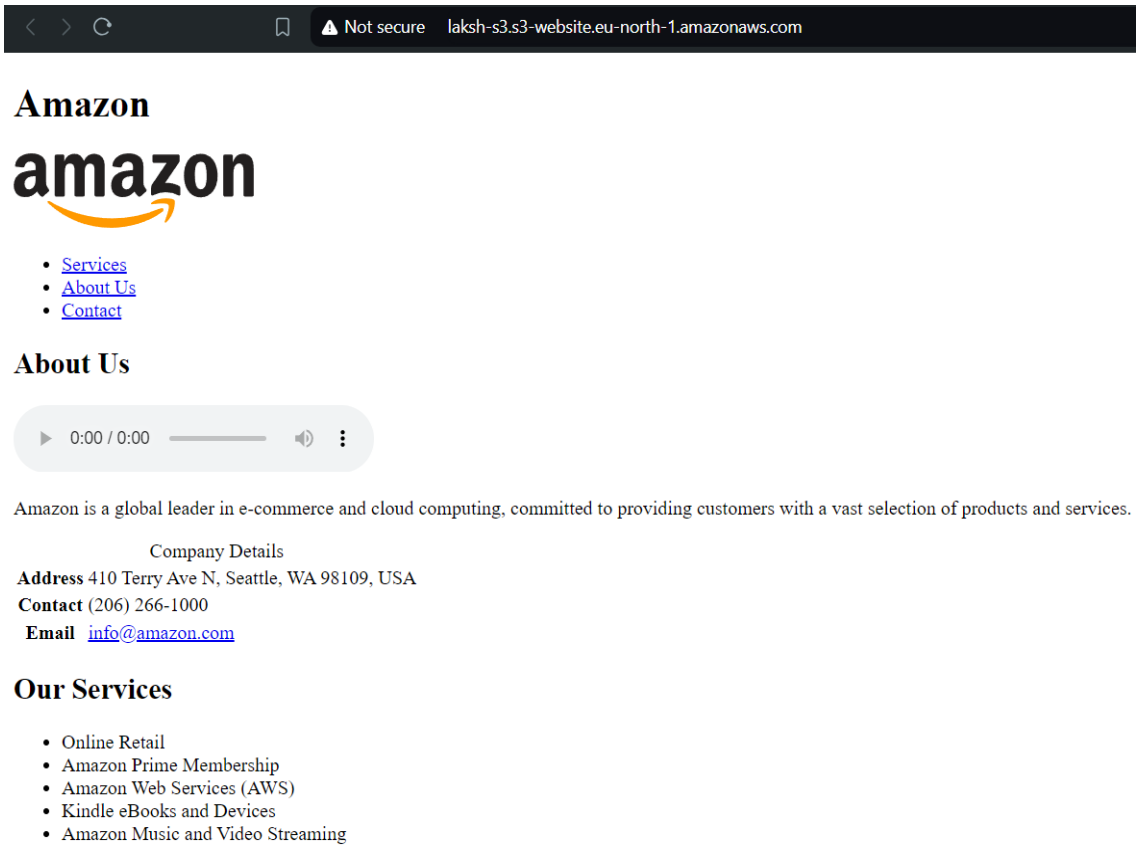
8. Start the apche server.



9. Give file location to be executed.



10. Output of the code on local machine.



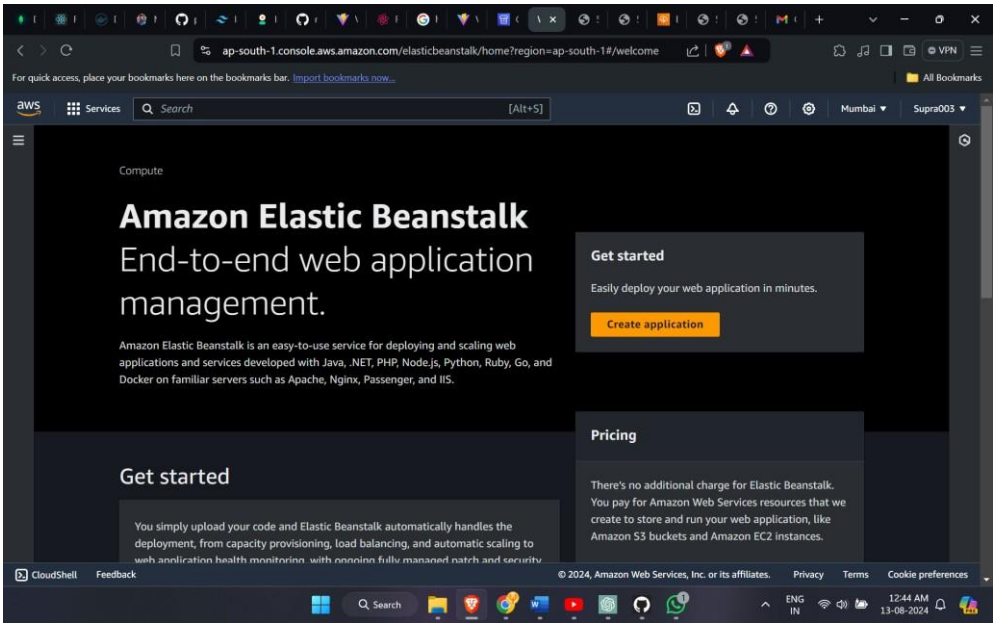
Experiment 02

Name:Laksh Sodhai **Roll no: 59**

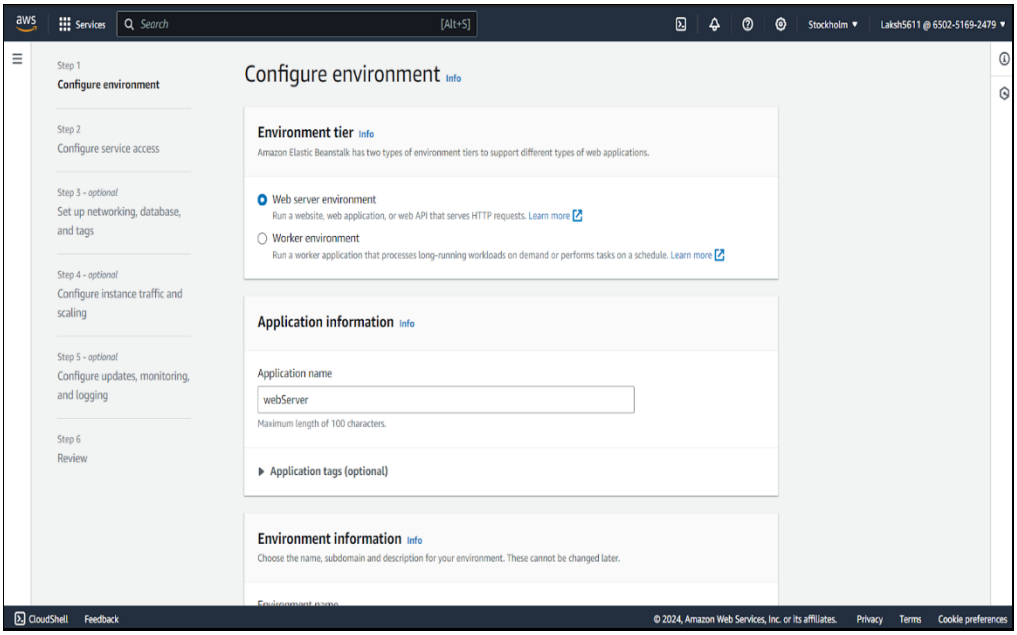
Aim: To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.

Step-1: Beanstalk Deployment

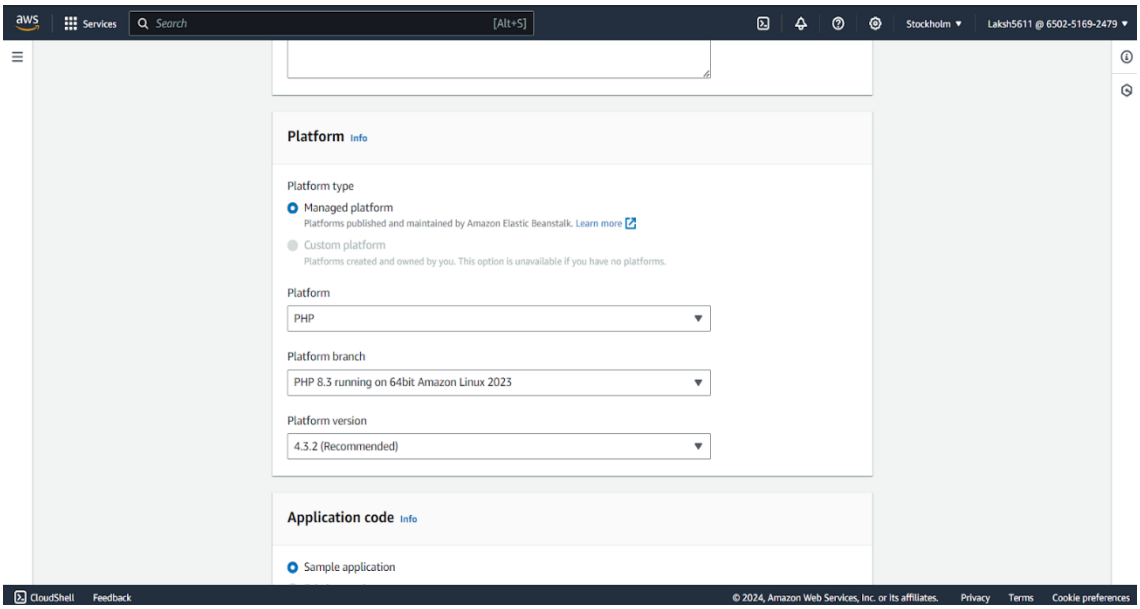
1. Login to your AWS account and search for Elastic Beanstalk in the search box



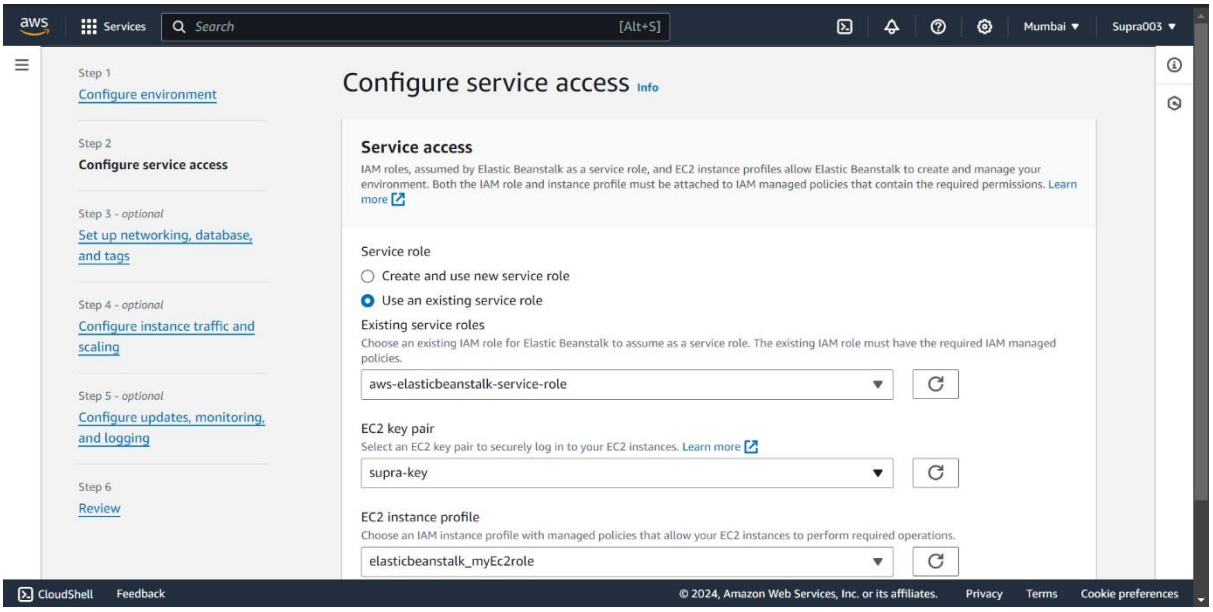
2. Open up Elastic Beanstalk and name your web app.

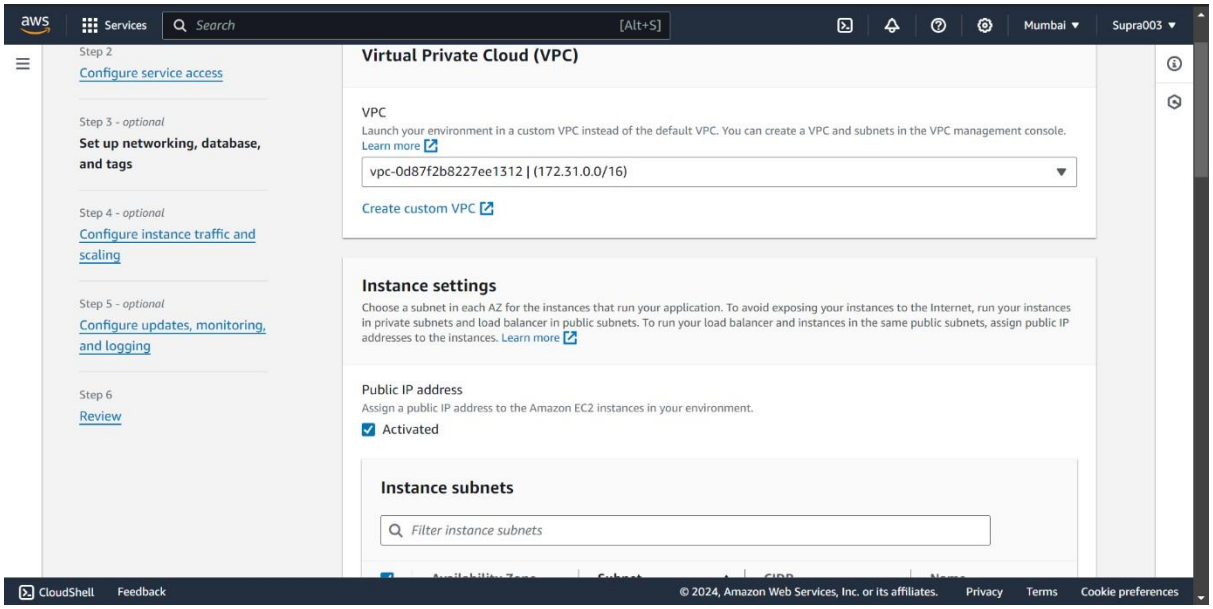


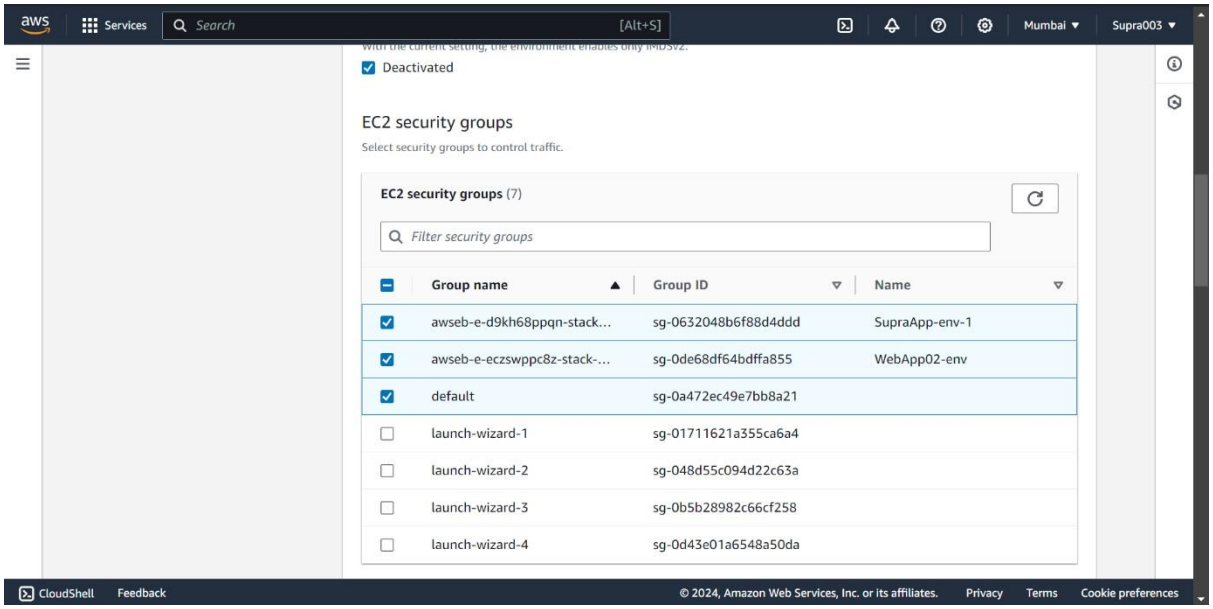
3. Choose Python from the drop-down menu and then click Create Application.

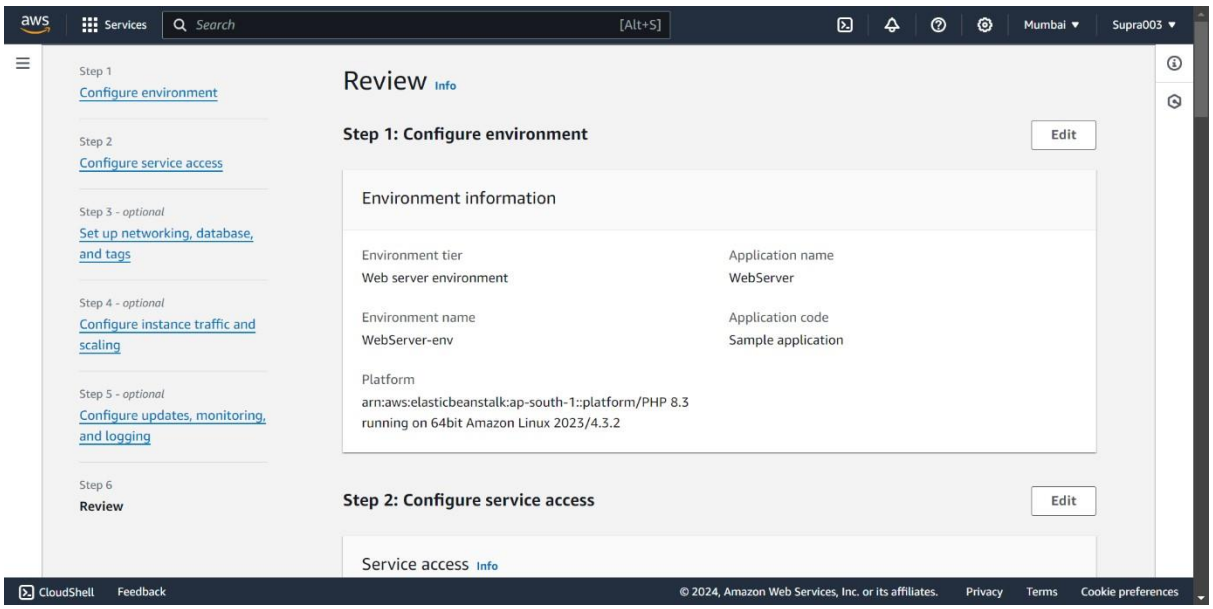


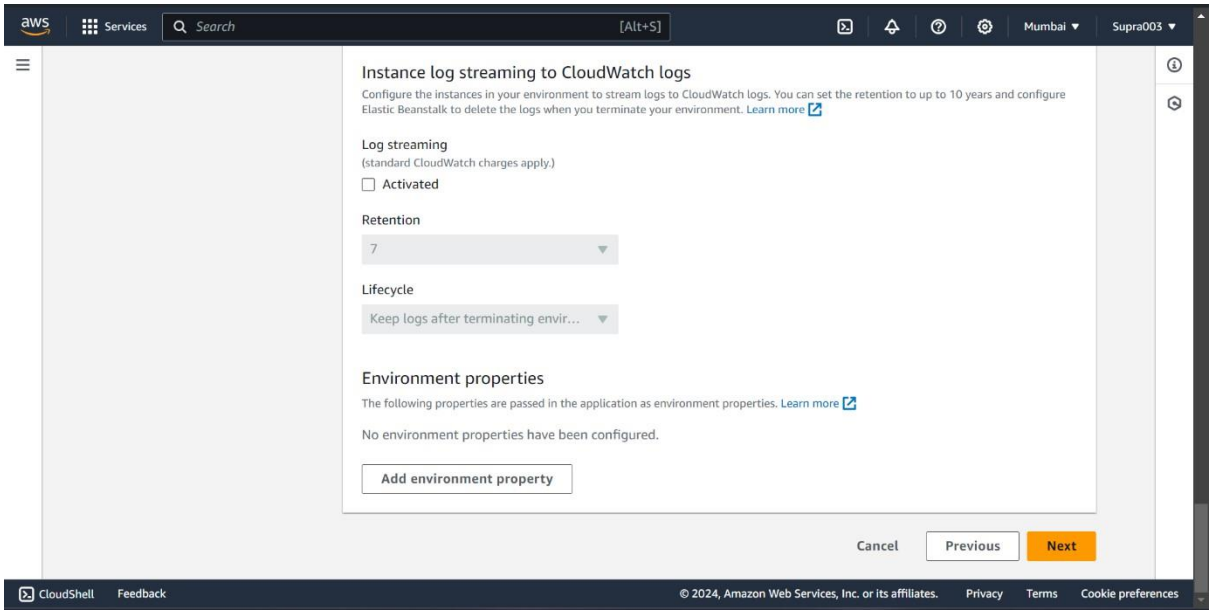
4. Choose the proper options

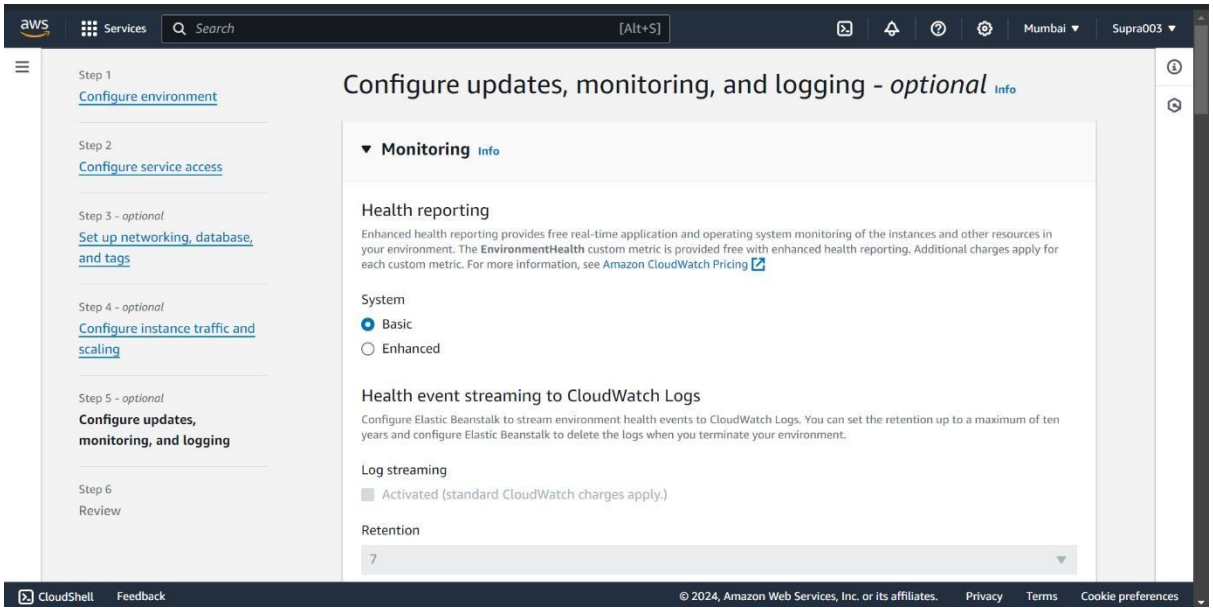



Services
Search
[Alt+S]
Mumbai
Supra003
Step 2
Configure service access
Step 3 - optional
Set up networking, database, and tags
Step 4 - optional
Configure instance traffic and scaling
Step 5 - optional
Configure updates, monitoring, and logging
Step 6
Review
Virtual Private Cloud (VPC)
VPC
Launch your environment in a custom VPC instead of the default VPC. You can create a VPC and subnets in the VPC management console.
Learn more
vpc-0d87f2b8227ee1312 | (172.31.0.0/16)
Create custom VPC
Instance settings
Choose a subnet in each AZ for the instances that run your application. To avoid exposing your instances to the Internet, run your instances in private subnets and load balancer in public subnets. To run your load balancer and instances in the same public subnets, assign public IP addresses to the instances. Learn more
Public IP address
Assign a public IP address to the Amazon EC2 instances in your environment.
Activated
Instance subnets
Filter instance subnets
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Services
Search
[Alt+S]
Mumbai
Supra003
Deactivated
EC2 security groups
Select security groups to control traffic.
EC2 security groups (7)
Filter security groups
Group name
Group ID
Name
awseb-e-d9kh68ppqn-stack...
sg-0632048b6f88d4ddd
SupraApp-env-1
awseb-e-eczswppc8z-stack-...
sg-0de68df64bdffa855
WebApp02-env
default
sg-0a472ec49e7bb8a21
launch-wizard-1
sg-01711621a355ca6a4
launch-wizard-2
sg-048d55c094d22c63a
launch-wizard-3
sg-0b5b28982c66cf258
launch-wizard-4
sg-0d43e01a6548a50da
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Services
Search
[Alt+S]
Mumbai
Supra003
Step 1
Configure environment
Step 2
Configure service access
Step 3 - optional
Set up networking, database, and tags
Step 4 - optional
Configure instance traffic and scaling
Step 5 - optional
Configure updates, monitoring, and logging
Step 6
Review
Review Info
Step 1: Configure environment
Edit
Environment information
Environment tier
Web server environment
Application name
WebServer
Environment name
WebServer-env
Application code
Sample application
Platform
arn:aws:elasticbeanstalk:ap-south-1::platform/PHP 8.3 running on 64bit Amazon Linux 2023/4.3.2
Step 2: Configure service access
Edit
Service access Info
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

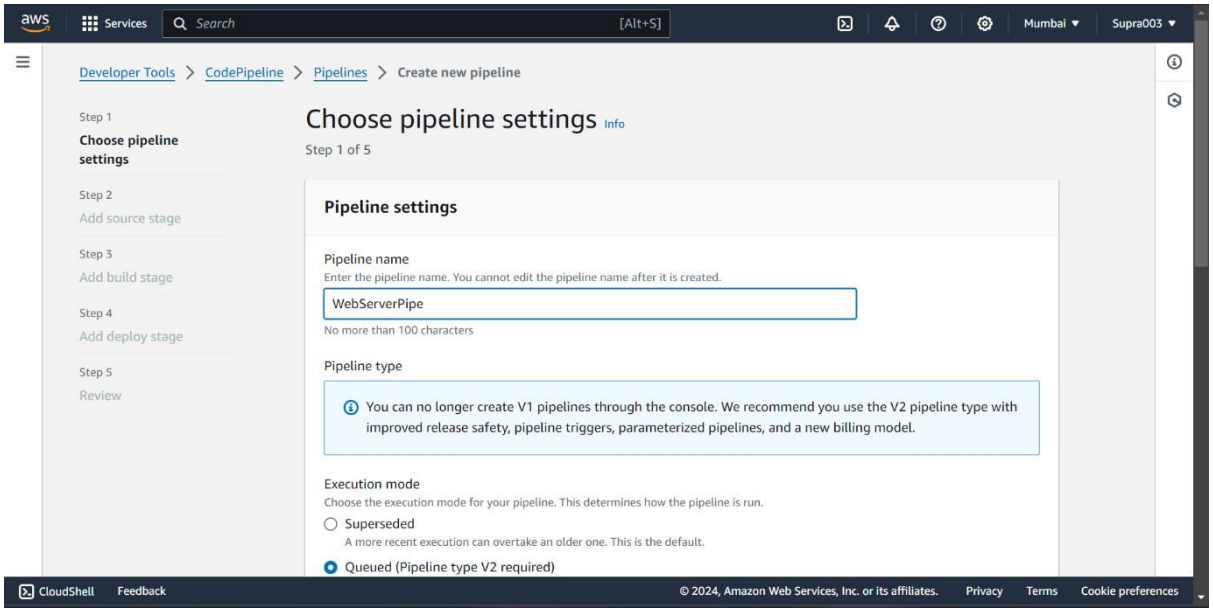

aws
Services
Search
[Alt+S]
Mumbai
Supra003
Instance log streaming to CloudWatch logs
Configure the instances in your environment to stream logs to CloudWatch logs. You can set the retention to up to 10 years and configure Elastic Beanstalk to delete the logs when you terminate your environment. Learn more
Log streaming
(standard CloudWatch charges apply.)
Activated
Retention
7
Lifecycle
Keep logs after terminating envir...
Environment properties
The following properties are passed in the application as environment properties. Learn more
No environment properties have been configured.
Add environment property
Cancel
Previous
Next
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


aws
Services
Search
[Alt+S]
Mumbai
Supra003
Step 1
Configure environment
Step 2
Configure service access
Step 3 - optional
Set up networking, database, and tags
Step 4 - optional
Configure instance traffic and scaling
Step 5 - optional
Configure updates, monitoring, and logging
Step 6
Review
Configure updates, monitoring, and logging - optional Info
Monitoring Info
Health reporting
Enhanced health reporting provides free real-time application and operating system monitoring of the instances and other resources in your environment. The EnvironmentHealth custom metric is provided free with enhanced health reporting. Additional charges apply for each custom metric. For more information, see Amazon CloudWatch Pricing
System
Basic
Enhanced
Health event streaming to CloudWatch Logs
Configure Elastic Beanstalk to stream environment health events to CloudWatch Logs. You can set the retention up to a maximum of ten years and configure Elastic Beanstalk to delete the logs when you terminate your environment.
Log streaming
Activated (standard CloudWatch charges apply.)
Retention
7
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

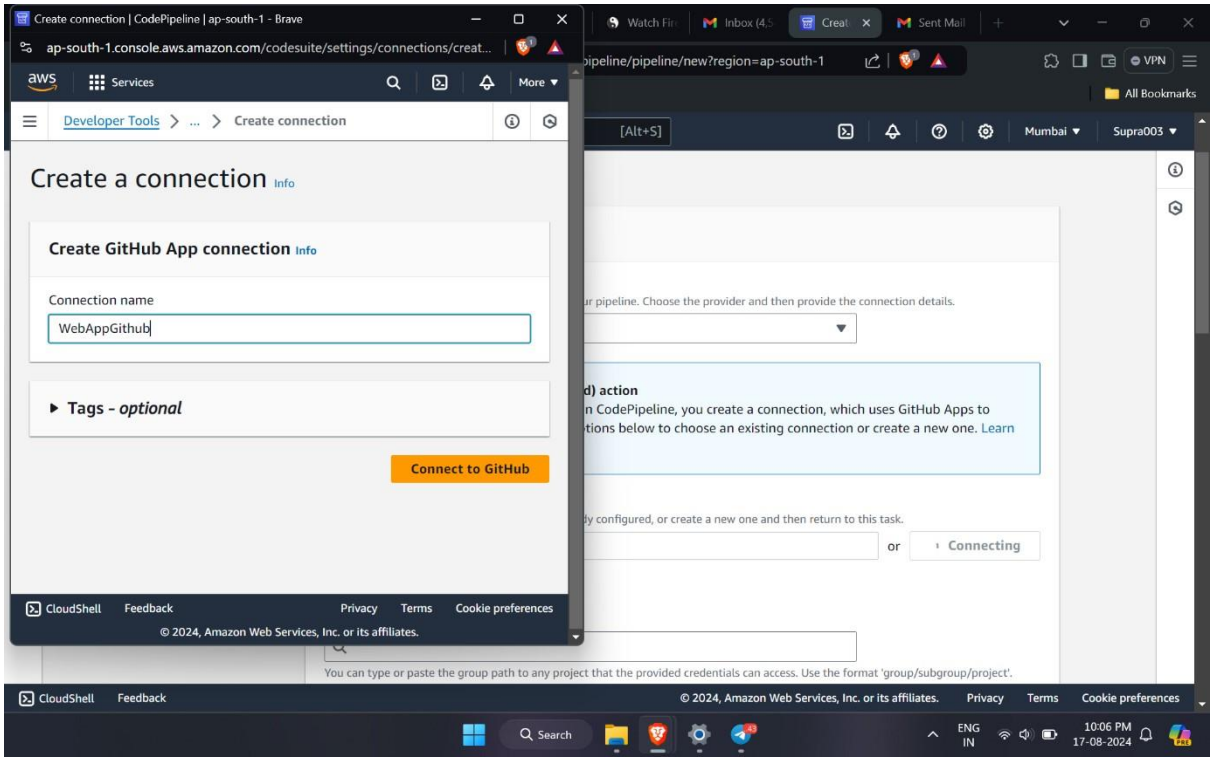
Step-2: Creating CodePipeline

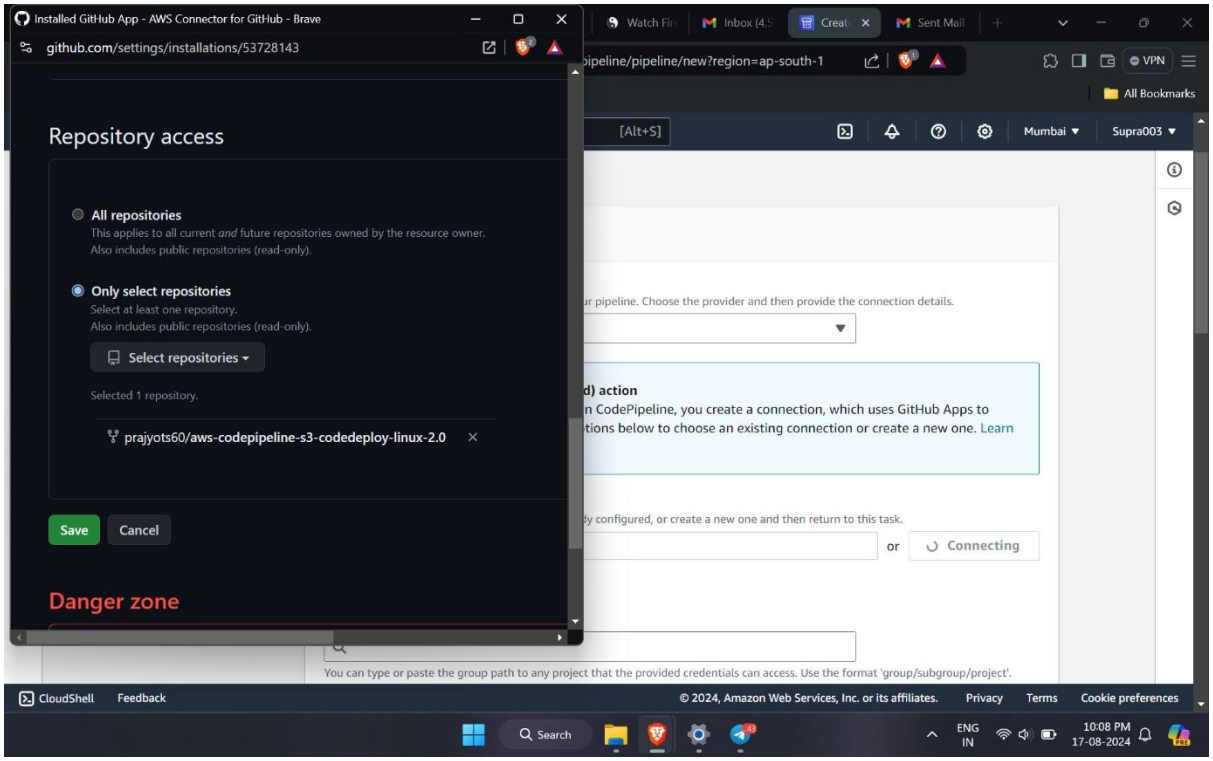
Beanstalk creates a sample environment for you to deploy your application. By default, it creates an EC2 instance, a security group, an Auto Scaling group, an Amazon S3 Bucket, Amazon CloudWatch alarms and a domain name for your Application.

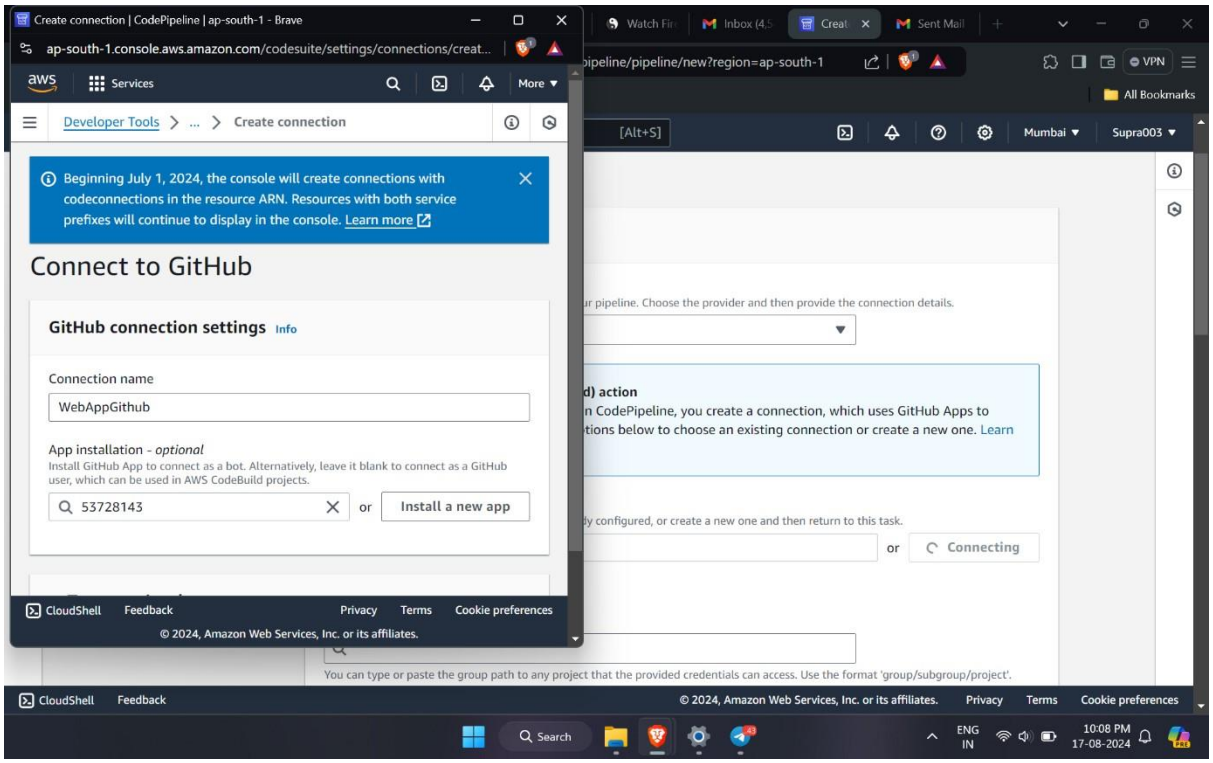
Create a CodePipeline and give pipeline name

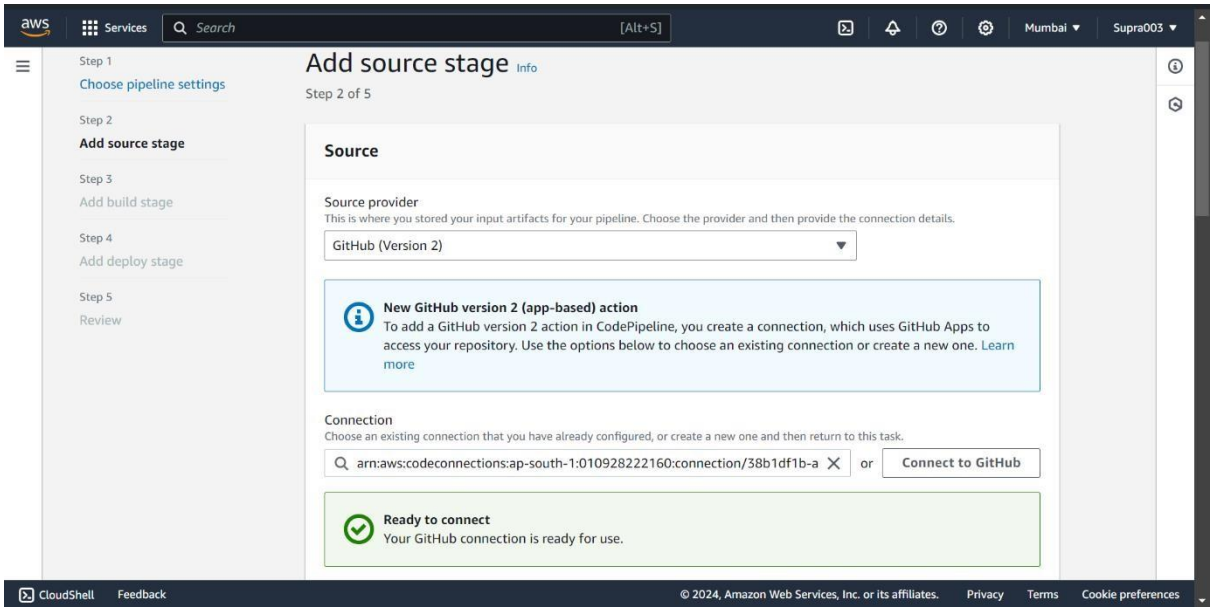


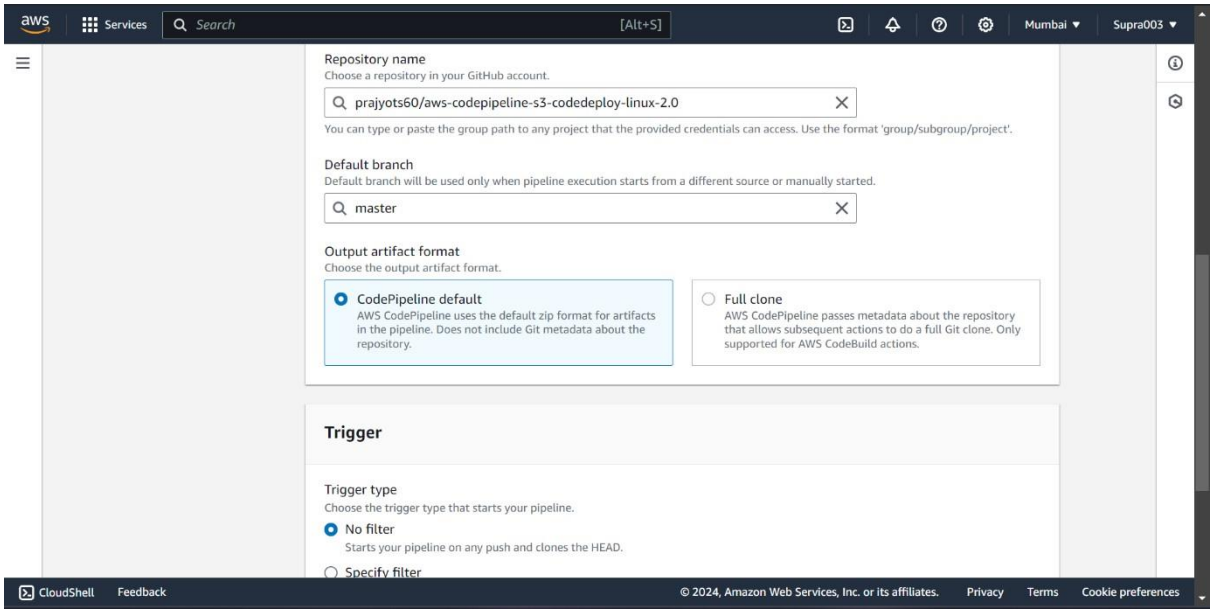
Create a Github connection



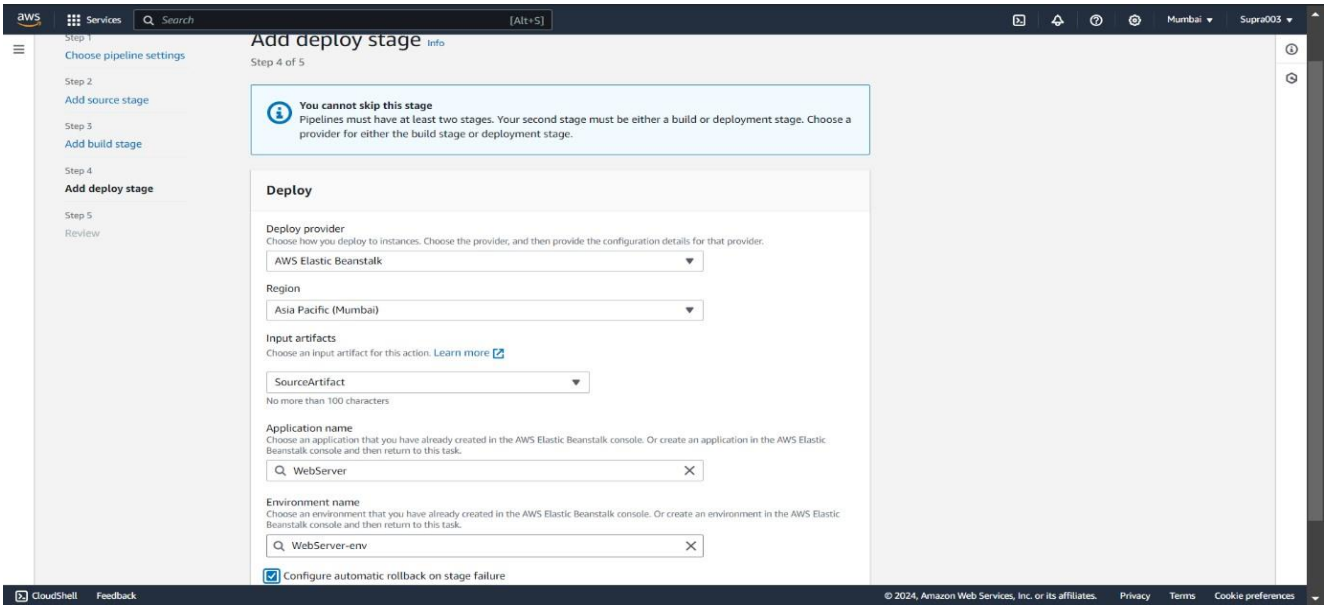

Installed GitHub App - AWS Connector for GitHub - Brave
github.com/settings/installations/53728143
Repository access
All repositories
This applies to all current and future repositories owned by the resource owner.
Also includes public repositories (read-only).
Only select repositories
Select at least one repository.
Also includes public repositories (read-only).
Select repositories
Selected 1 repository.
prajyots60/aws-codepipeline-s3-codedeploy-linux-2.0
Save
Cancel
Danger zone
pipeline/pipeline/new?region=ap-south-1
[Alt+S]
Mumbai
Supra003
pipeline. Choose the provider and then provide the connection details.
d) action
n CodePipeline, you create a connection, which uses GitHub Apps to
tions below to choose an existing connection or create a new one. Learn
ly configured, or create a new one and then return to this task.
or
Connecting
You can type or paste the group path to any project that the provided credentials can access. Use the format 'group/subgroup/project'.
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
10:08 PM
17-08-2024

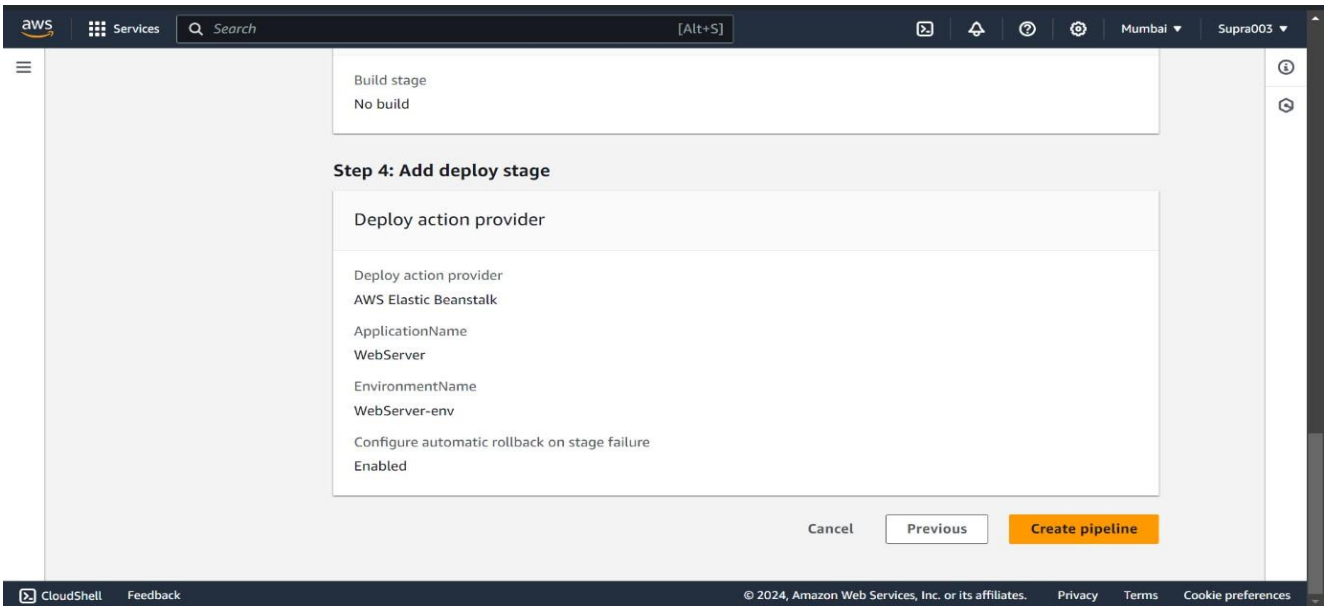

Create connection | CodePipeline | ap-south-1 - Brave
ap-south-1.console.aws.amazon.com/codesuite/settings/connections/creat...
aws
Services
More
Developer Tools
...
Create connection
Beginning July 1, 2024, the console will create connections with codeconnections in the resource ARN. Resources with both service prefixes will continue to display in the console. Learn more
Connect to GitHub
GitHub connection settings Info
Connection name
WebAppGithub
App installation - optional
Install GitHub App to connect as a bot. Alternatively, leave it blank to connect as a GitHub user, which can be used in AWS CodeBuild projects.
53728143
or
Install a new app
CloudShell
Feedback
Privacy
Terms
Cookie preferences
© 2024, Amazon Web Services, Inc. or its affiliates.
pipeline/pipeline/new?region=ap-south-1
[Alt+S]
Mumbai
Supra003
pipeline. Choose the provider and then provide the connection details.
d) action
n CodePipeline, you create a connection, which uses GitHub Apps to
tions below to choose an existing connection or create a new one. Learn
ly configured, or create a new one and then return to this task.
or
Connecting
You can type or paste the group path to any project that the provided credentials can access. Use the format 'group/subgroup/project'.
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
10:08 PM
17-08-2024


Services
Search
[Alt+S]
Mumbai
Supra003
Step 1
Choose pipeline settings
Step 2
Add source stage
Step 3
Add build stage
Step 4
Add deploy stage
Step 5
Review
Add source stage Info
Step 2 of 5
Source
Source provider
This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.
GitHub (Version 2)
New GitHub version 2 (app-based) action
To add a GitHub version 2 action in CodePipeline, you create a connection, which uses GitHub Apps to access your repository. Use the options below to choose an existing connection or create a new one. Learn more
Connection
Choose an existing connection that you have already configured, or create a new one and then return to this task.
arn:aws:codeconnections:ap-south-1:010928222160:connection/38b1df1b-a
or
Connect to GitHub
Ready to connect
Your GitHub connection is ready for use.
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

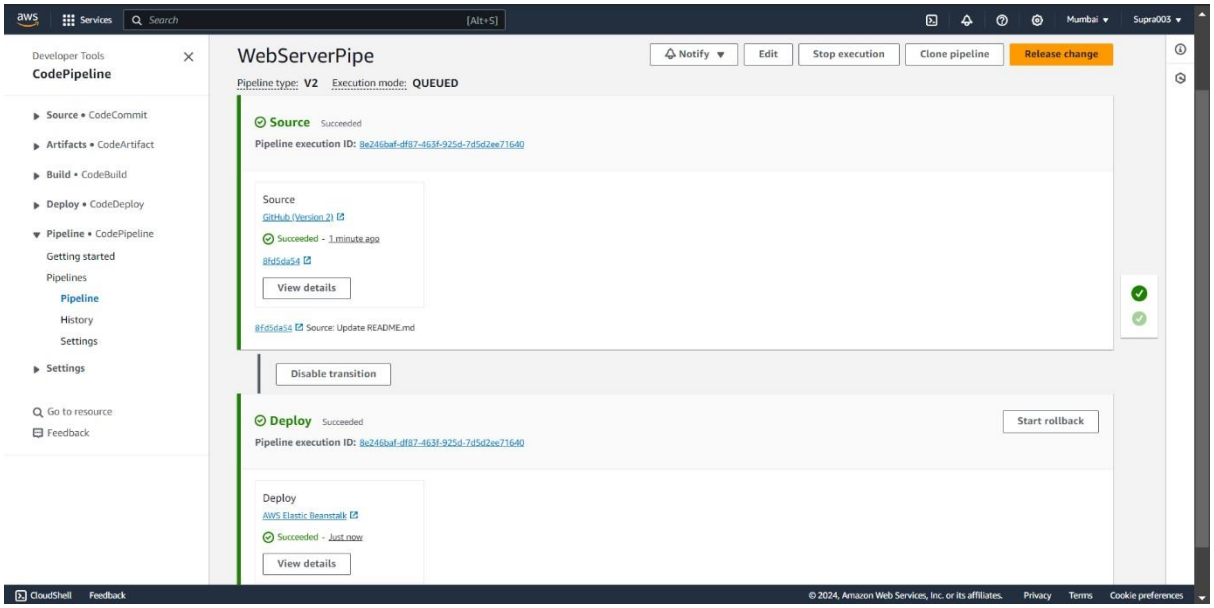

Services
Search
[Alt+S]
Mumbai
Supra003
Repository name
Choose a repository in your GitHub account.
prajyots60/aws-codepipeline-s3-codedeploy-linux-2.0
You can type or paste the group path to any project that the provided credentials can access. Use the format 'group/subgroup/project'.
Default branch
Default branch will be used only when pipeline execution starts from a different source or manually started.
master
Output artifact format
Choose the output artifact format.
CodePipeline default
AWS CodePipeline uses the default zip format for artifacts in the pipeline. Does not include Git metadata about the repository.
Full clone
AWS CodePipeline passes metadata about the repository that allows subsequent actions to do a full Git clone. Only supported for AWS CodeBuild actions.
Trigger
Trigger type
Choose the trigger type that starts your pipeline.
No filter
Starts your pipeline on any push and clones the HEAD.
Specify filter
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

For deploy choose elastic beanstalk

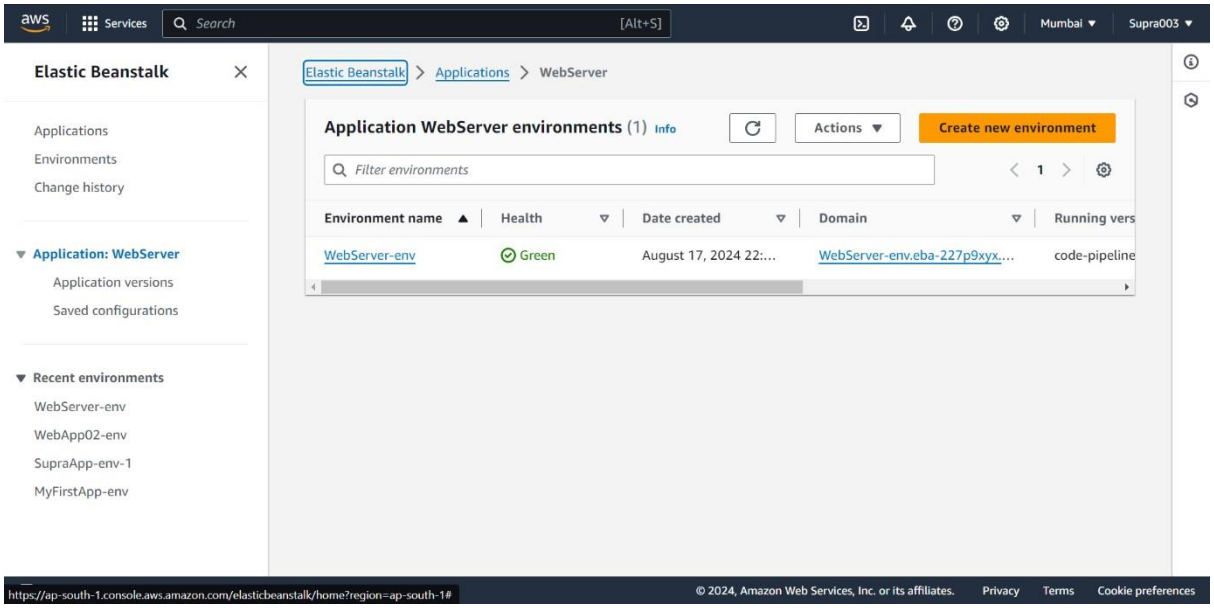


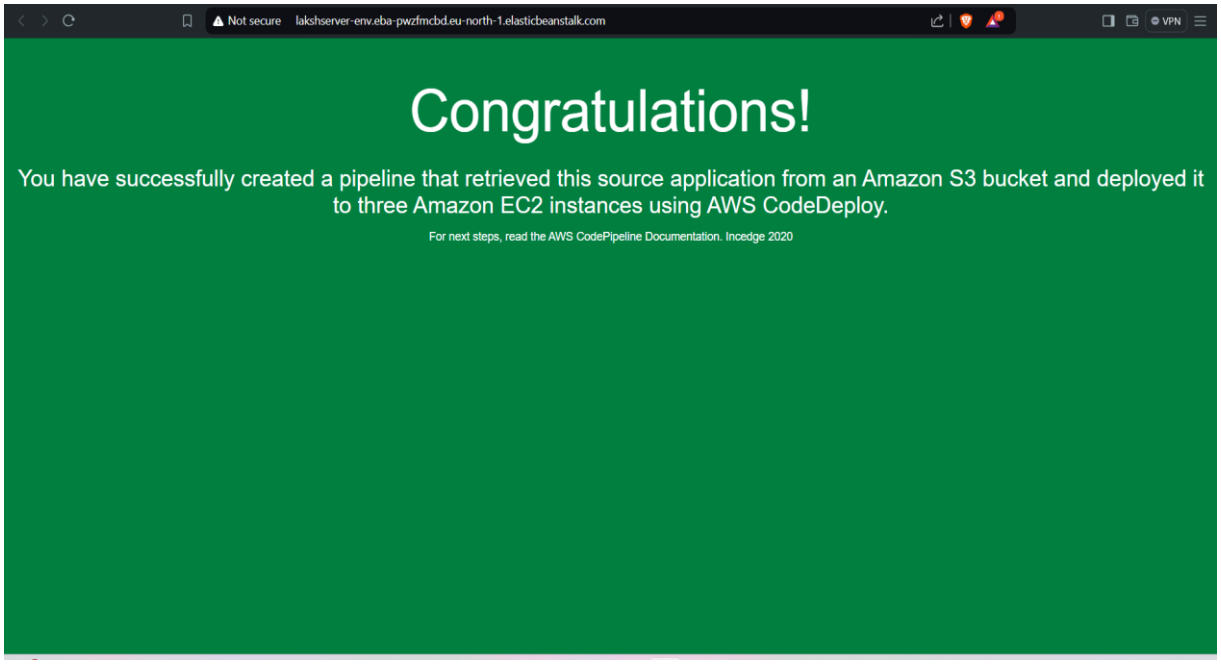


Make connection between code pipeline and beanstalk. And Green means successfully connected.

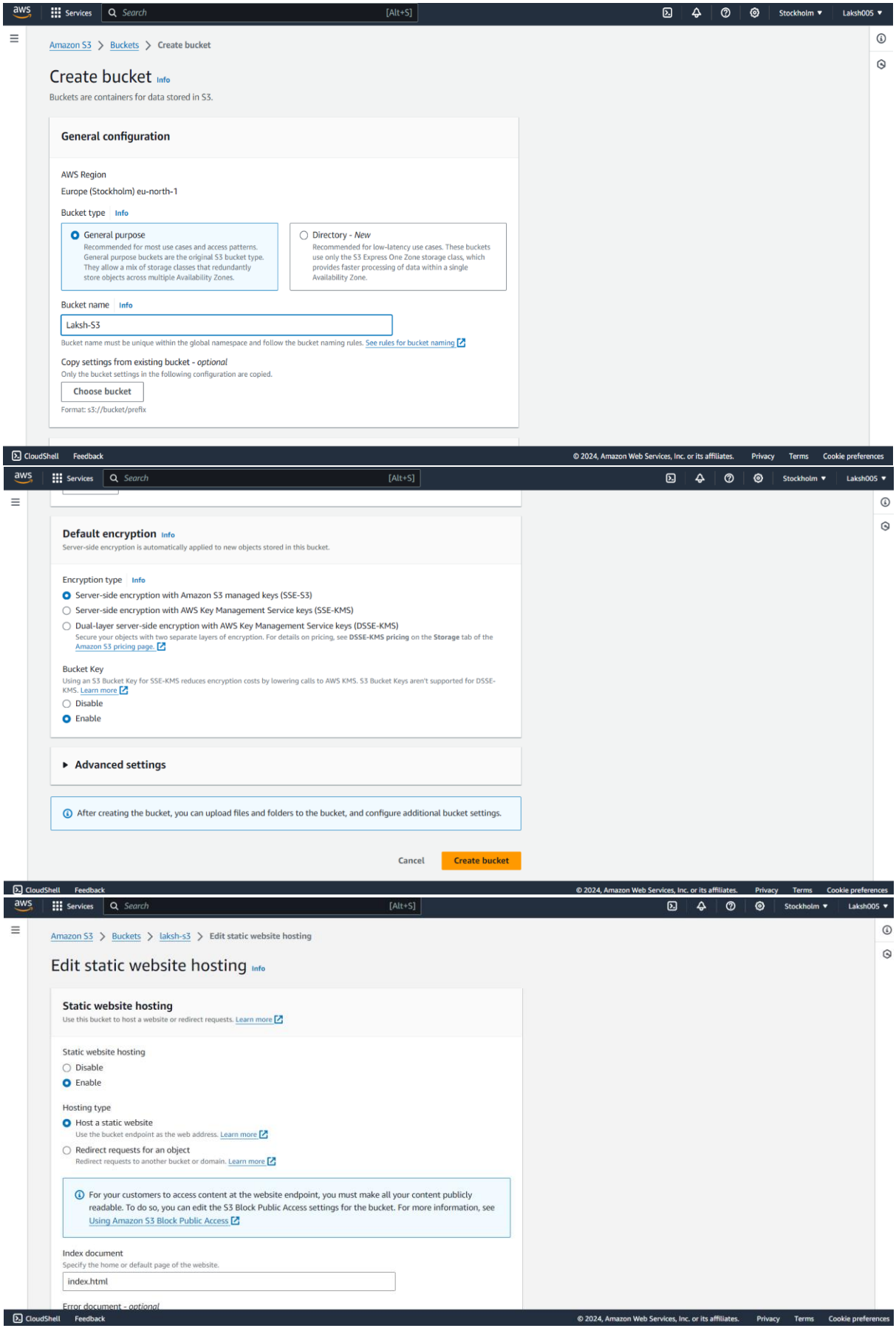


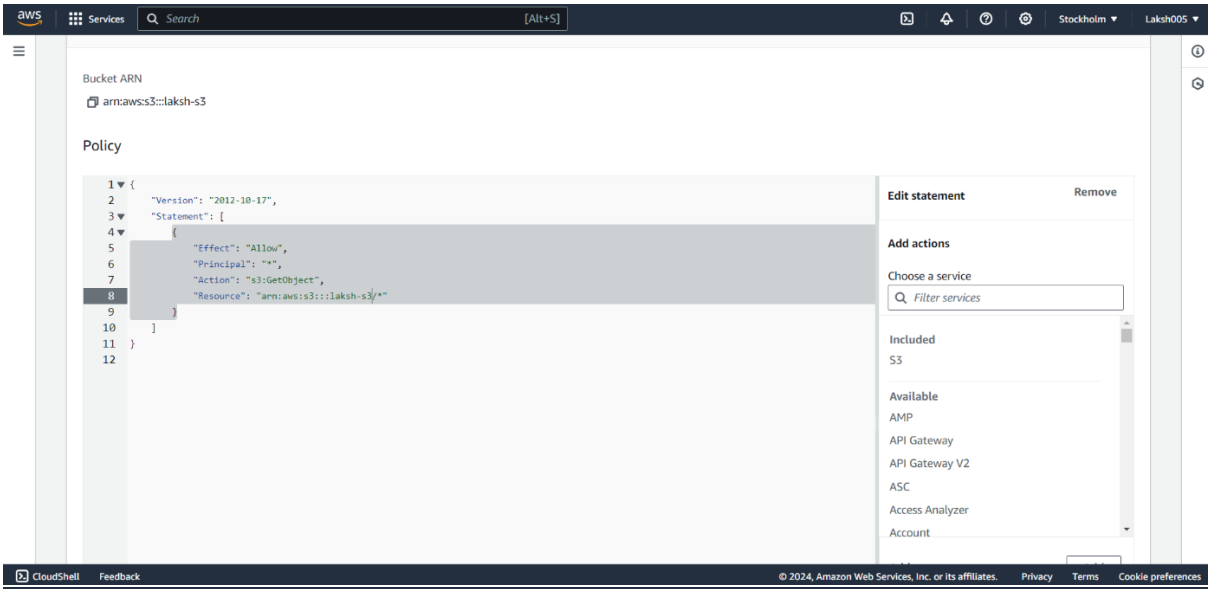
Click on domain to view you hosting site

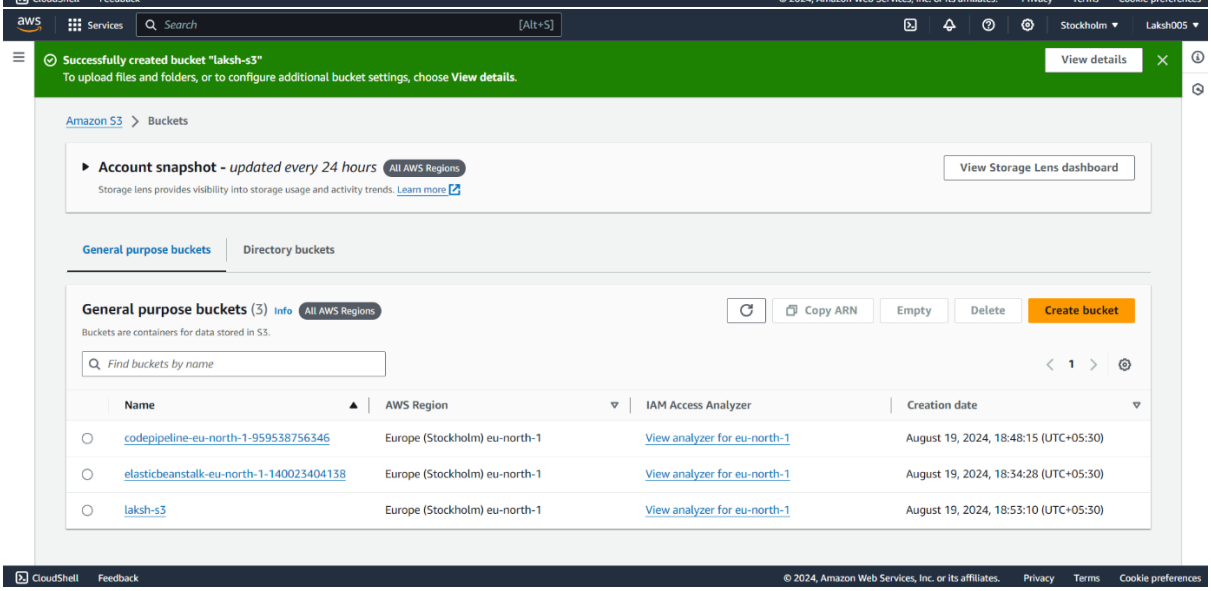


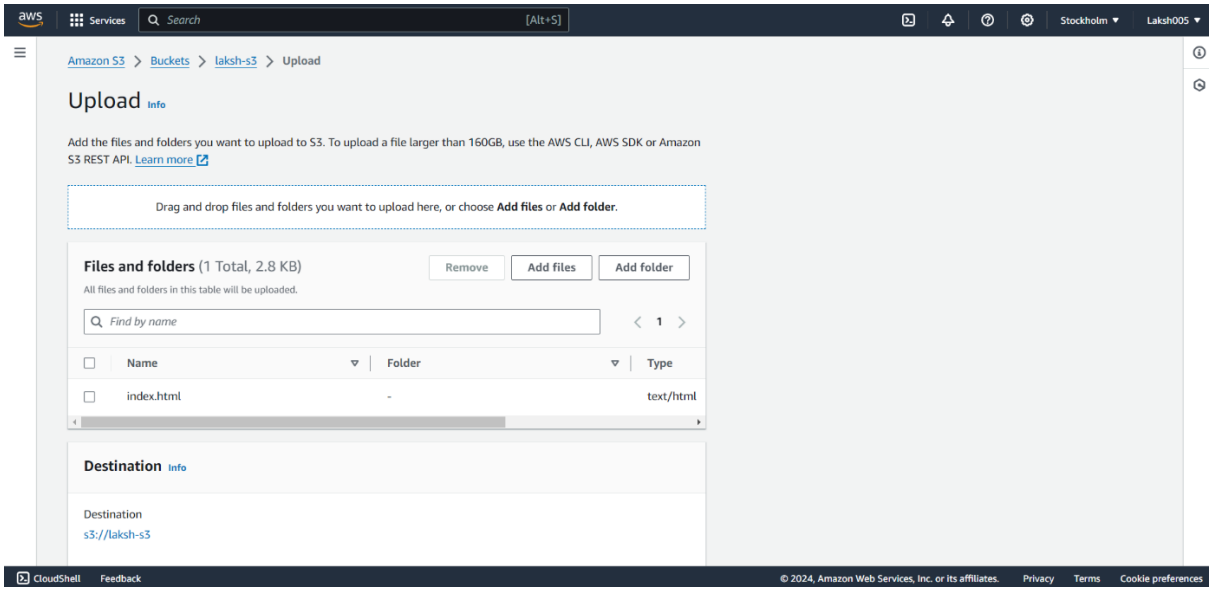

Not secure
lakshserver-env.eba-pwzfmcbd.eu-north-1.elasticbeanstalk.com
VPN
Congratulations!
You have successfully created a pipeline that retrieved this source application from an Amazon S3 bucket and deployed it to three Amazon EC2 instances using AWS CodeDeploy.
For next steps, read the AWS CodePipeline Documentation. Incedge 2020

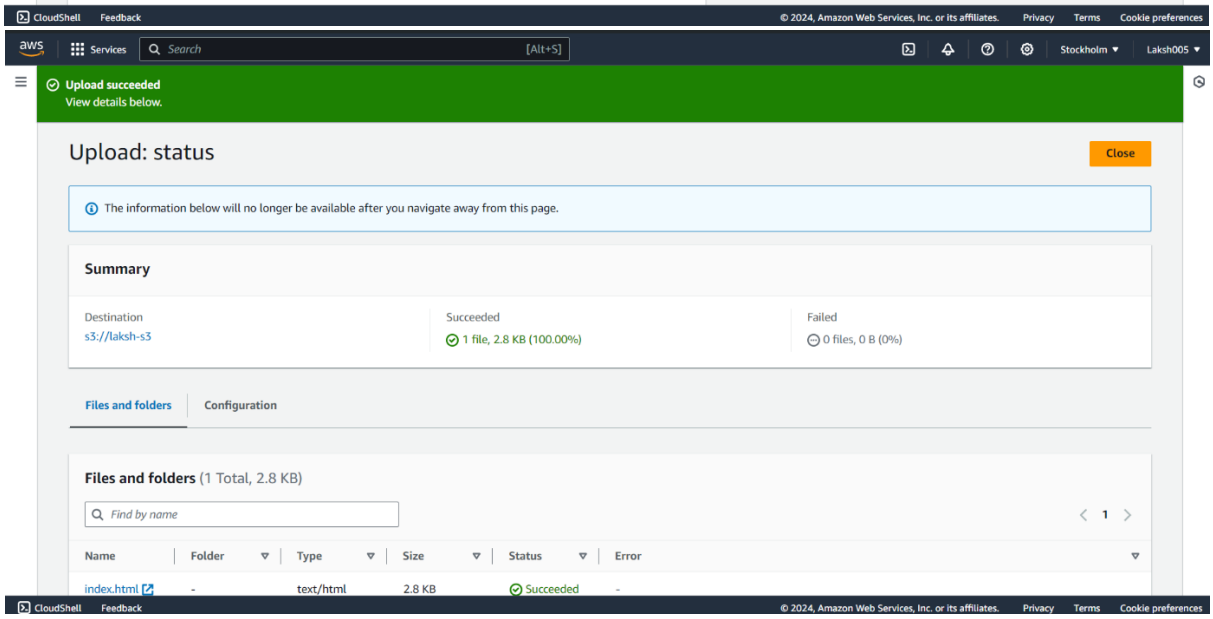
Hosting a static website on Amazon S3

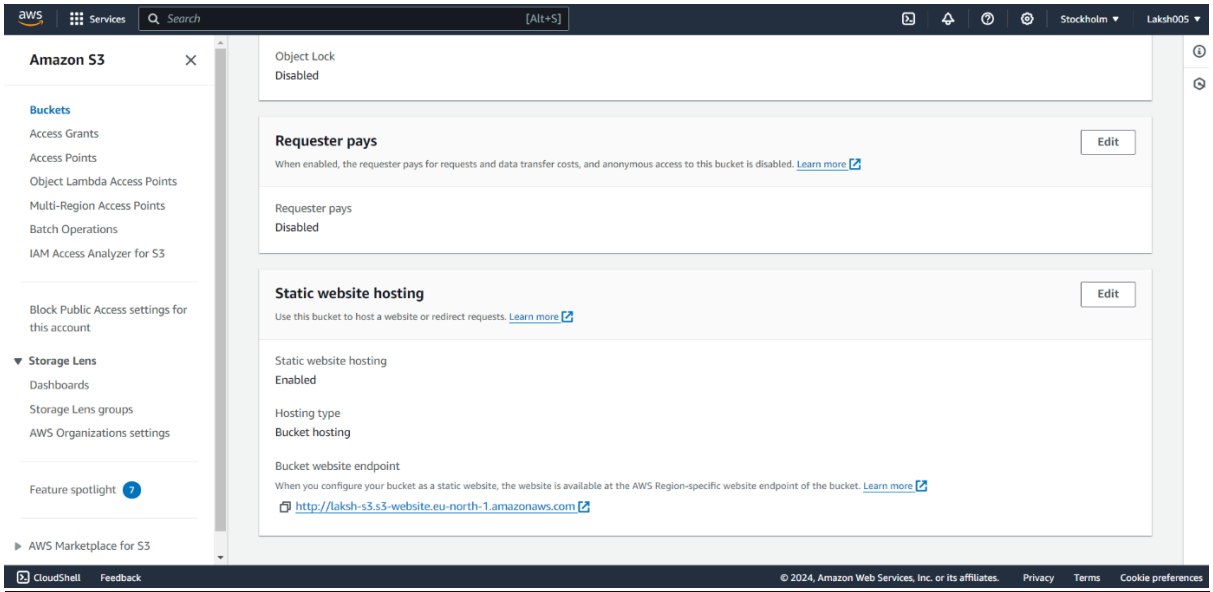


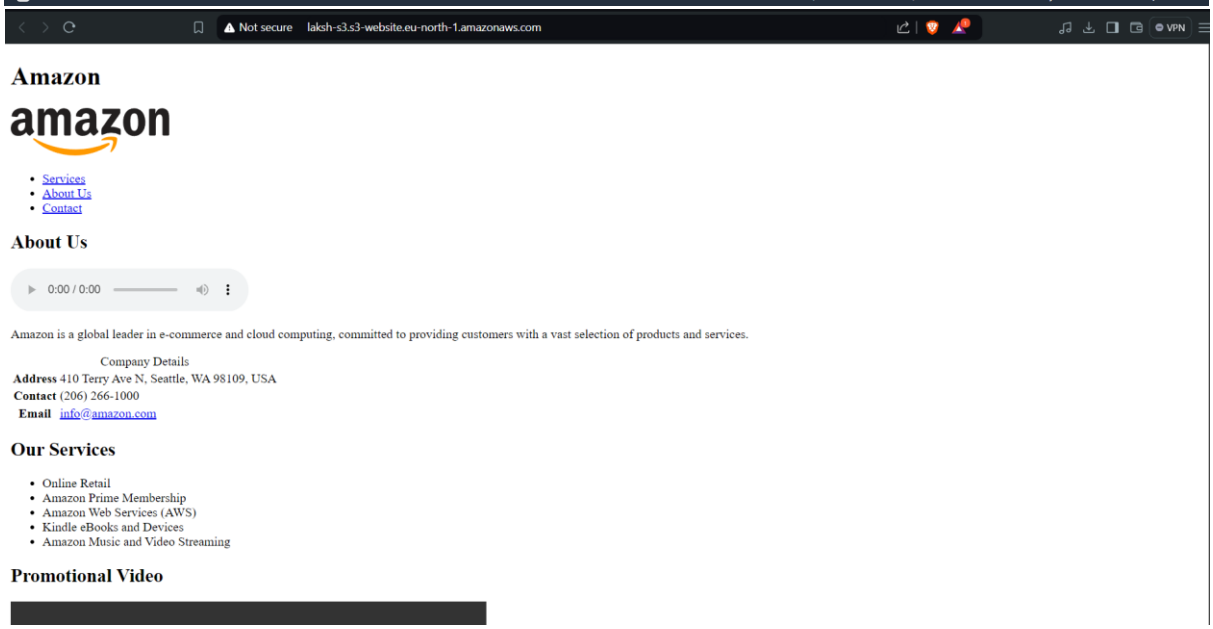

Services
Search
[Alt+S]
Stockholm
Laksh005
Bucket ARN
arn:aws:s3:::laksh-s3
Policy
{
"Version": "2012-10-17",
"Statement": [
{
"Effect": "Allow",
"Principal": "*",
"Action": "s3:GetObject",
"Resource": "arn:aws:s3:::laksh-s3/*"
}
]
}
Edit statement
Remove
Add actions
Choose a service
Filter services
Included
S3
Available
AMP
API Gateway
API Gateway V2
ASC
Access Analyzer
Account
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Services
Search
[Alt+S]
Stockholm
Laksh005
Successfully created bucket "laksh-s3"
To upload files and folders, or to configure additional bucket settings, choose View details.
View details
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 (3) 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
codepipeline-eu-north-1-959538756346
Europe (Stockholm) eu-north-1
View analyzer for eu-north-1
August 19, 2024, 18:48:15 (UTC+05:30)
elasticbeanstalk-eu-north-1-140023404138
Europe (Stockholm) eu-north-1
View analyzer for eu-north-1
August 19, 2024, 18:34:28 (UTC+05:30)
laksh-s3
Europe (Stockholm) eu-north-1
View analyzer for eu-north-1
August 19, 2024, 18:53:10 (UTC+05:30)
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Services
Search
[Alt+S]
Stockholm
Laksh005
Amazon S3 > Buckets > laksh-s3 > Upload
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, 2.8 KB)
Remove
Add files
Add folder
All files and folders in this table will be uploaded.
Find by name
1
Name
Folder
Type
index.html
-
text/html
Destination Info
Destination
s3://laksh-s3
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Services
Search
[Alt+S]
Stockholm
Laksh005
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://laksh-s3
Succeeded
1 file, 2.8 KB (100.00%)
Failed
0 files, 0 B (0%)
Files and folders
Configuration
Files and folders (1 Total, 2.8 KB)
Find by name
1
Name
Folder
Type
Size
Status
Error
index.html
-
text/html
2.8 KB
Succeeded
-
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


aws
Services
Search
[Alt+S]
Stockholm
Laksh005
Amazon S3
Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Feature spotlight
7
AWS Marketplace for S3
Object Lock
Disabled
Requester pays
When enabled, the requester pays for requests and data transfer costs, and anonymous access to this bucket is disabled. Learn more
Edit
Requester pays
Disabled
Static website hosting
Use this bucket to host a website or redirect requests. Learn more
Edit
Static website hosting
Enabled
Hosting type
Bucket hosting
Bucket website endpoint
When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. Learn more
http://laksh-s3.s3-website.eu-north-1.amazonaws.com
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Not secure
laksh-s3.s3-website.eu-north-1.amazonaws.com
VPN
Amazon
amazon
Services
About Us
Contact
About Us
0:00 / 0:00
Amazon is a global leader in e-commerce and cloud computing, committed to providing customers with a vast selection of products and services.
Company Details
Address 410 Terry Ave N, Seattle, WA 98109, USA
Contact (206) 266-1000
Email info@amazon.com
Our Services
Online Retail
Amazon Prime Membership
Amazon Web Services (AWS)
Kindle eBooks and Devices
Amazon Music and Video Streaming
Promotional Video

PRACTICAL NO: - 3

Case Study: Understanding Kubernetes Cluster Architecture

Introduction: -

Kubernetes is an open-source platform used to deploy and maintain a group of containers in a virtualized environment. In practice, Kubernetes is most commonly used alongside Docker for better control and implementation of containerized applications. Containerized applications "bundle" applications together with all its files, libraries, and packages required for it to run reliably and efficiently on different platforms. However, it operates at the container level rather than at the hardware level.

The name Kubernetes is derived from a Greek term meaning 'helmsman' or 'pilot.' True to this word, Kubernetes provides the guiding force for developer platforms to transition from virtual machines (VMs) to containers and the statically scheduled to the dynamically scheduled. This means no more manual integration and configuration when you move from a testing environment to an actual production environment or from on-premise to the cloud! The Kubernetes logical compute environment offers common services to all the applications in the cluster as part of the ecosystem for the software to run consistently.

Features of Kubernetes

- Automates various manual processes and controls server hosting and launching
- Manages containers offer security, and networking and storage services
- Monitors and continuously checks the health of nodes and containers
- Automates rollback for changes that go wrong
- Mounts and adds a storage system to run apps

Purpose of Kubernetes

The primary purpose of Kubernetes is to enable developers to write and deploy applications that can run seamlessly across multiple operating environments. Traditionally, application performance and deployment were tightly coupled to specific infrastructures, often requiring adherence to cloud provider-specific constructs and back-end storage systems. This dependency resulted in infrastructure lock-in, limiting flexibility and scalability

Kubernetes addresses this challenge by abstracting the underlying infrastructure, allowing developers to deploy cloud-native applications in containers without restrictions. This means applications can be managed and scaled consistently across different environments—be it in the cloud, on-premises, or in hybrid setups—providing true infrastructure independence and operational flexibility.

Working of Kubernetes

Before exploring how Kubernetes operates, it's essential to grasp the concept of containers and their significance in modern application development. A container is a small, lightweight virtual machine (VM) that does not have device drivers and shares its operating system among the applications. It is a good way to bundle and run applications in a production environment. However, you need to manage these containers in a proper way so that there is no downtime. This is where Kubernetes comes to the rescue.

Kubernetes works as a "container orchestration system" that manages the lifecycle of containerized applications and automates the deployment of several containers. Containers running the same applications are usually grouped together into Pods. There can be one or multiple containers in a single Pod and each of them shares the same IP address and resources such as memory and storage. By grouping the containers in this manner, Kubernetes eliminates the need to cram multiple functionalities in one single container. There is a dedicated container orchestrator which supervises these groups and ensures that they operate correctly.

Kubernetes in DevOps

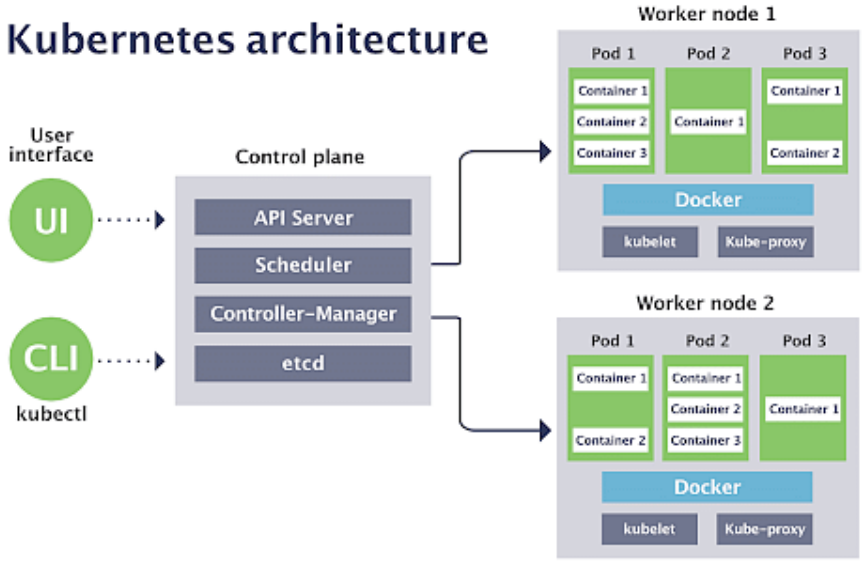
Kubernetes is more than just a container orchestration tool; it's a powerful enabler of DevOps practices. By bridging the gap between IT operations and development, Kubernetes fosters a collaborative DevOps environment, ensuring that software and its dependencies are shared seamlessly across different environments.

Kubernetes facilitates various stages of the software lifecycle, enhancing the build-test-deploy timeline:

- **Developer Environment:** Helps run software consistently in any setting, ensuring that applications behave the same across different environments.
- **QA/Testing Process:** Coordinates pipelines between test and production environments, streamlining the testing process.
- **Sys-Admin:** Once configured, Kubernetes runs anything, simplifying system administration tasks.
- **Operations:** Provides a comprehensive solution for building, shipping, and scaling software, making operations smoother and more efficient.

Kubernetes Cluster Architecture

A Kubernetes cluster consists of a set of worker machines, called nodes, that run containerized applications. The cluster is managed by a control plane, which is responsible for maintaining the desired state of the cluster, such as which applications are running and where they are running.



Control Plane: The control plane is the brain of the Kubernetes cluster, responsible for managing the desired state of the cluster, making decisions on scheduling, and responding to cluster events.

- **API Server:** The API server acts as the front-end for the Kubernetes control plane. It exposes the Kubernetes API, which is used by both internal components and external users (via CLI or UI) to communicate with the cluster.
- **Scheduler:** The scheduler is responsible for assigning newly created pods to nodes in the cluster. It evaluates the resource requirements of the pods against the available resources on the nodes, ensuring optimal placement.
- **Controller Manager:** This component runs various controllers that manage the state of the cluster. For example, it ensures that the number of pod replicas matches the desired configuration and handles node failures.
- **etcd:** A key-value store that holds all the configuration data for the Kubernetes cluster, including the current state and the desired state of the objects in the cluster. It is essential for maintaining cluster consistency and recovery.

Worker Nodes: Worker nodes are the machines where the application workloads run. Each worker node contains the services necessary to run pods and communicate with the control plane.

- **Kubelet:** The kubelet is an agent that runs on each worker node. It ensures that containers are running in a pod as expected by the control plane. It communicates with the API server to receive instructions and report back the status of the node and its workloads.
- **Kube-proxy:** Kube-proxy is a network proxy that runs on each worker node. It manages the networking for the pods, ensuring that each pod can communicate with others, both within and outside the cluster.
- **Docker (or other container runtimes):** Docker is the container runtime that runs and manages the containers on the worker nodes. It pulls container images from a registry, starts and stops containers, and manages container storage and networking.

User Interfaces: Users interact with the Kubernetes cluster through two primary interfaces:

- **UI (User Interface):** A graphical interface that provides an easy way to manage and monitor the cluster.
- **CLI (Command-Line Interface, e.g., kubectl):** A more powerful tool for managing the cluster, allowing users to interact with the API server directly via command-line commands.

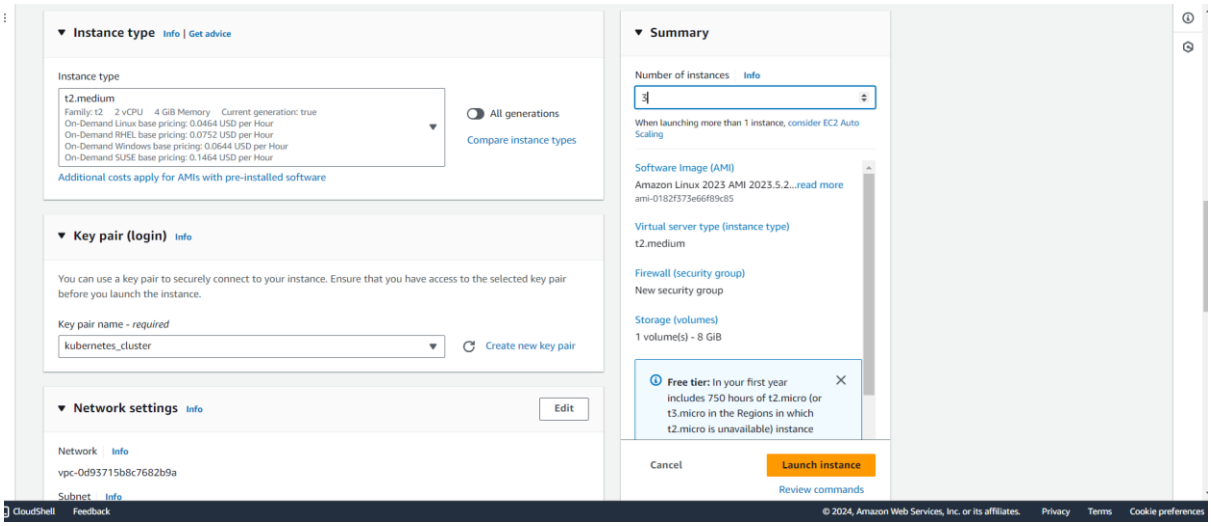
This architecture allows Kubernetes to abstract away the underlying infrastructure, providing a consistent and scalable environment for deploying and managing containerized applications across different environments.

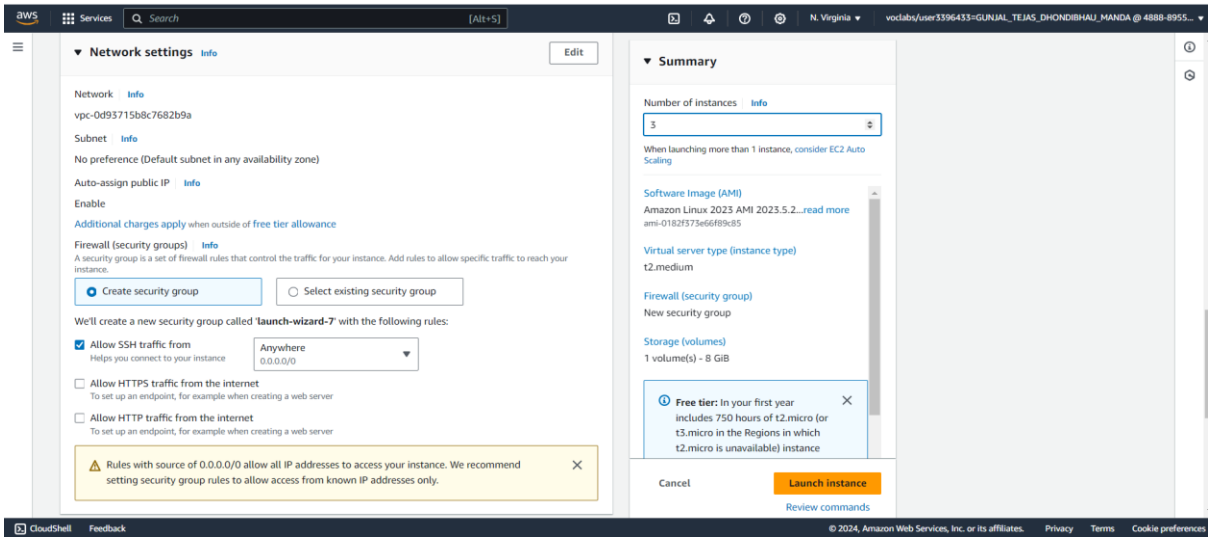
Conclusion:-

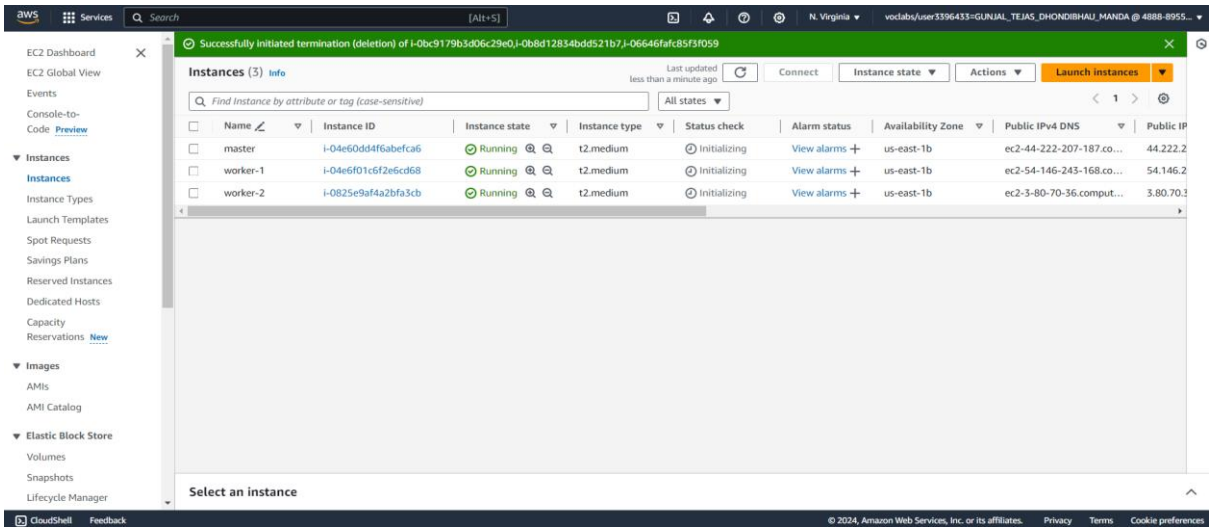
In conclusion, Kubernetes is a powerful, open-source platform that automates the deployment and management of containerized applications. Its architecture ensures seamless operation across diverse environments, promoting flexibility and scalability. By abstracting the underlying infrastructure and supporting DevOps practices, Kubernetes enhances the software development lifecycle. This results in streamlined operations, reduced downtime, and consistent application performance.

Install and Spin Up a Kubernetes Cluster on Linux Machines/Cloud Platforms.

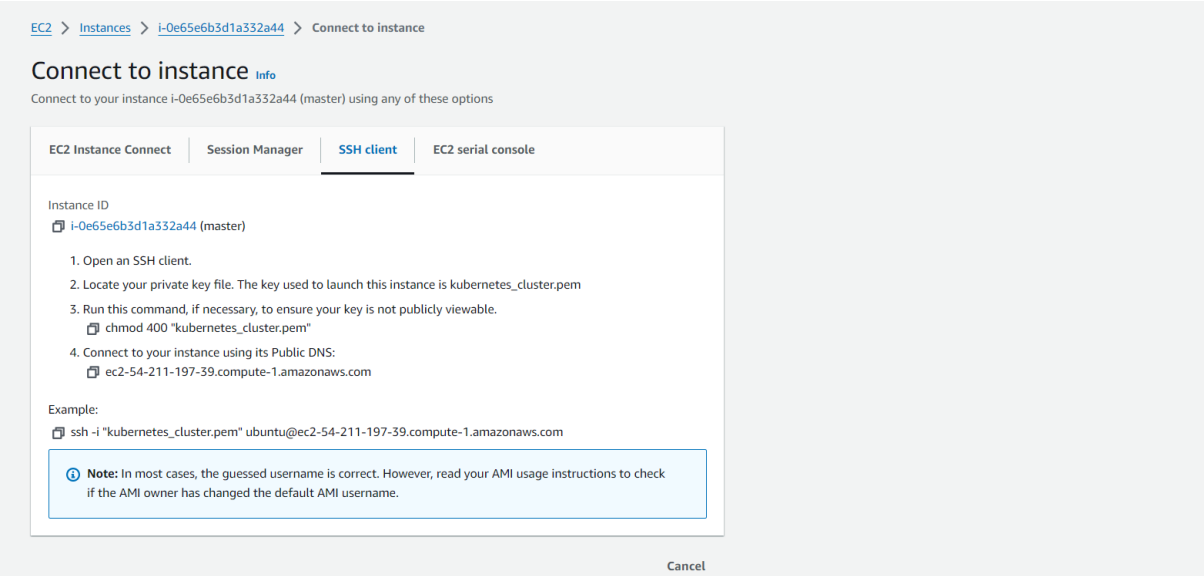
- Create 3 EC2 instances, one for the master node and two for the worker nodes







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

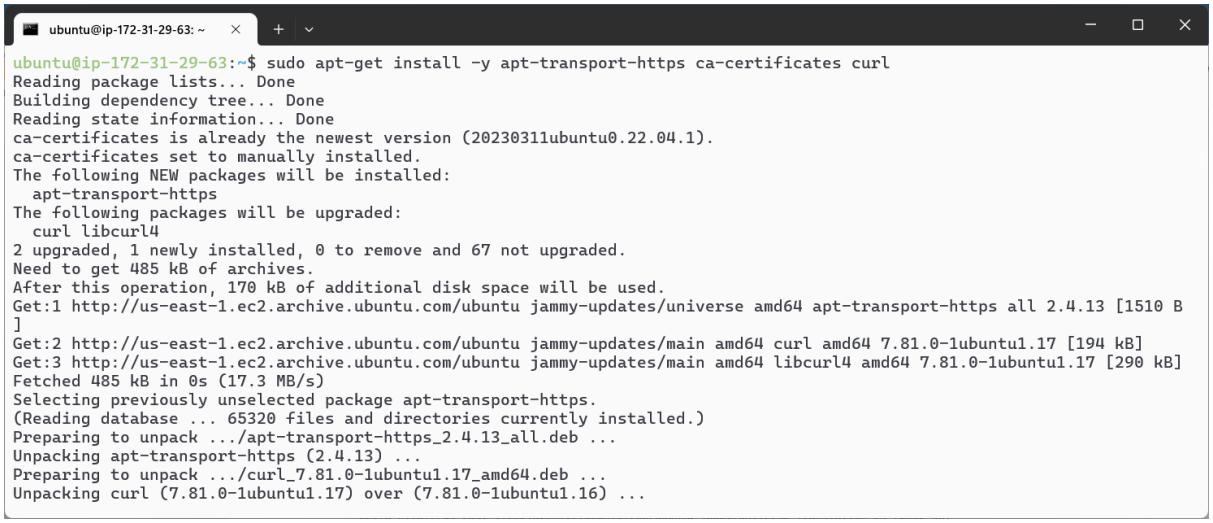



```
ubuntu@ip-172-31-29-63: ~
Microsoft Windows [Version 10.0.22631.4112]
(c) Microsoft Corporation. All rights reserved.

C:\Users\TEJAS\Downloads>ssh -i "kubernetes_cluster.pem" ubuntu@ec2-54-211-197-39.compute-1.amazonaws.com
The authenticity of host 'ec2-54-211-197-39.compute-1.amazonaws.com (54.211.197.39)' can't be established.
ED25519 key fingerprint is SHA256:E7RRK6LxQFHSnbWLb8zgsskyIdjOYXpInW0bO8/5vnA.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-211-197-39.compute-1.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 Sun Sep 15 17:48:58 UTC 2024

  System load:  0.28              Processes:             115
  Usage of /:   20.7% of 7.57GB   Users logged in:       0
  Memory usage: 5%                IPv4 address for eth0: 172.31.29.63
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.
```

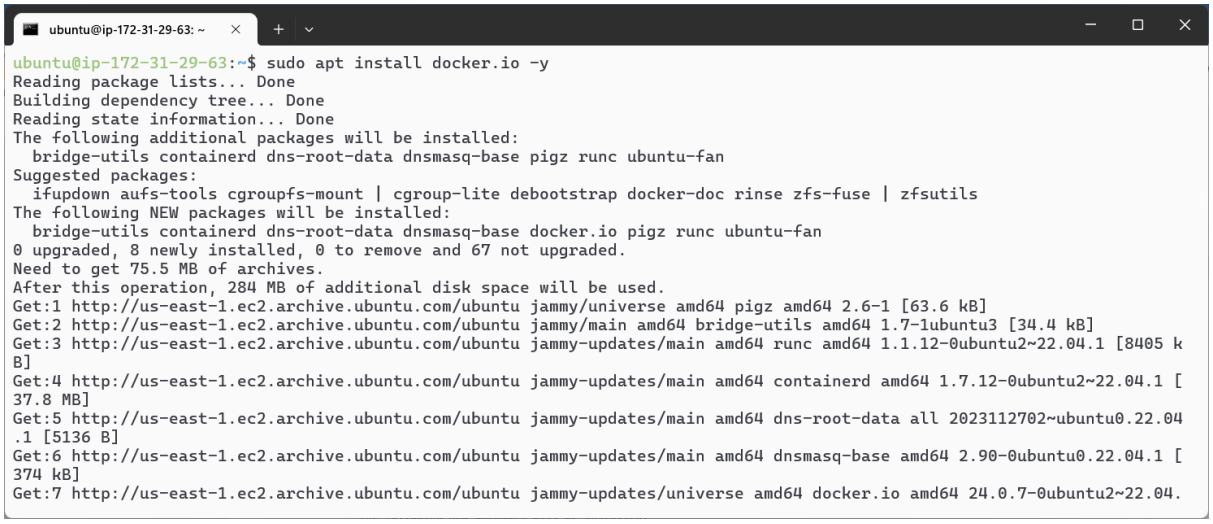
- Update the package manager on all nodes:

```
ubuntu@ip-172-31-29-63: ~
ubuntu@ip-172-31-29-63:~$ sudo apt update
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]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]
Get:7 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [1806 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [217 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 B]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [2023 kB]
Get:13 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [295 kB]
Get:14 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [13.3 kB]
Get:15 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [2377 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [352 kB]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [17.8 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [2437 kB]
Get:19 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [409 kB]
Get:20 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 c-n-f Metadata [584 B]
Get:21 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [902 kB]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [419 kB]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 c-n-f Metadata [616 B]
```

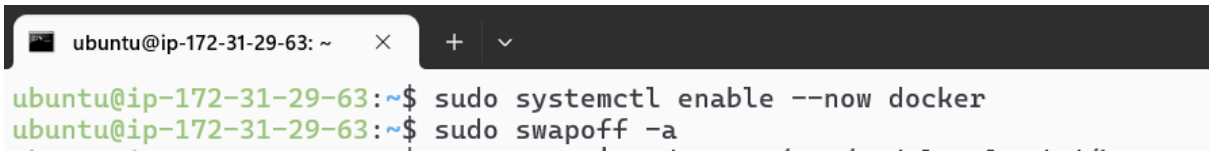
- Installing Required Packages for HTTPS and Certificate Transport on Ubuntu



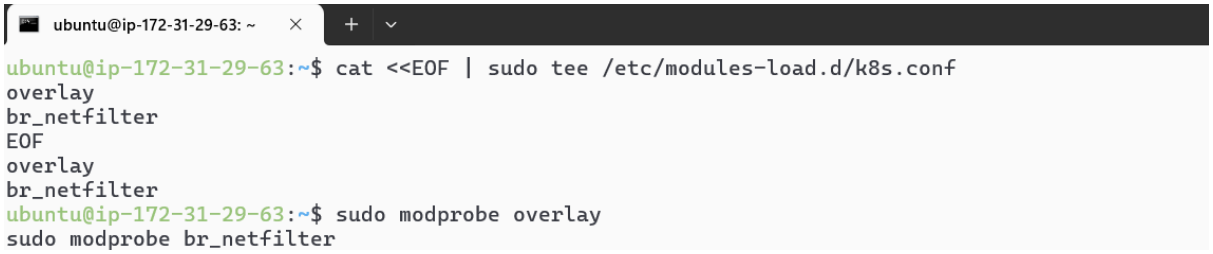
- Installing Docker on Ubuntu



- Enabling Docker and Disabling Swap on Ubuntu



- Load necessary kernel modules for networking and iptables:



- Configure sysctl settings for Kubernetes networking:

```
ubuntu@ip-172-31-29-63:~$ cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-iptables  = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward                 = 1
EOF
net.bridge.bridge-nf-call-iptables  = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward                 = 1
```

```
ubuntu@ip-172-31-29-63:~$ sudo sysctl --system
* 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
* Applying /etc/sysctl.d/10-magic-sysrq.conf ...
kernel.sysrq = 176
* Applying /etc/sysctl.d/10-network-security.conf ...
net.ipv4.conf.default.rp_filter = 2
net.ipv4.conf.all.rp_filter = 2
* Applying /etc/sysctl.d/10-ptrace.conf ...
kernel.yama.ptrace_scope = 1
* Applying /etc/sysctl.d/10-zeropage.conf ...
vm.mmap_min_addr = 65536
* Applying /etc/sysctl.d/50-cloudimg-settings.conf ...
net.ipv4.neigh.default.gc_thresh2 = 15360
net.ipv4.neigh.default.gc_thresh3 = 16384
net.netfilter.nf_conntrack_max = 1048576
* Applying /usr/lib/sysctl.d/50-default.conf ...
kernel.core_uses_pid = 1
net.ipv4.conf.default.rp_filter = 2
```

- Install Kubernetes tools on all nodes.

```
ubuntu@ip-172-31-29-63:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.29/deb/Release.key | sudo gpg --dearmor -o /et
c/apt/keyrings/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.29/deb/ /' | sud
o tee /etc/apt/sources.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.29/deb/ /

ubuntu@ip-172-31-29-63:~$ sudo apt-get update -y
sudo apt-get install -y kubelet="1.29.0-*" kubectl="1.29.0-*" kubeadm="1.29.0-*"
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
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.29/deb  InRelease [1189 B]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.29/deb  Packages [14.0 kB]
Fetched 15.1 kB in 0s (31.9 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Selected version '1.29.0-1.1' (isv:kubernetes:core:stable:v1.29:pkgs.k8s.io [amd64]) for 'kubelet'
Selected version '1.29.0-1.1' (isv:kubernetes:core:stable:v1.29:pkgs.k8s.io [amd64]) for 'kubectl'
Selected version '1.29.0-1.1' (isv:kubernetes:core:stable:v1.29:pkgs.k8s.io [amd64]) for 'kubeadm'
The following additional packages will be installed:
  conntrack cri-tools ebtables kubernetes-cni socat
The following NEW packages will be installed:
```

```
ubuntu@ip-172-31-29-63:~$ sudo apt-mark hold kubelet kubeadm kubectl
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
```

Only on master node

- Initialize the Kubernetes Cluster on Master Node

```
ubuntu@ip-172-31-29-63: ~
ubuntu@ip-172-31-29-63:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
I0915 17:58:19.113429    3356 version.go:256] remote version is much newer: v1.31.0; falling back to: stable-1.29
[init] Using Kubernetes version: v1.29.8
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
W0915 17:58:28.696218    3356 checks.go:835] detected that the sandbox image "registry.k8s.io/pause:3.8" of the containe
r runtime is inconsistent with that used by kubeadm. It is recommended that using "registry.k8s.io/pause:3.9" as the CRI
 sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-29-63 kubernetes kubernetes.default kubernetes.default
.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.29.63]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-29-63 localhost] and IPs [172.31.29.63 127.0.0.1 ::1
]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-29-63 localhost] and IPs [172.31.29.63 127.0.0.1 ::1]
```

```
ubuntu@ip-172-31-29-63: ~

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

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

Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.29.63:6443 --token 05rvqu.66v5246nmoe81d5e \
        --discovery-token-ca-cert-hash sha256:ff6e3056ea0d41a598919b1be9dbe765739c40a5aa6d37b5e4cbc45b1256c1c7
```

- Set up kubectl on the master node

```
ubuntu@ip-172-31-29-63:~$ 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-172-31-29-63:~$ kubectl get nodes
NAME              STATUS     ROLES           AGE   VERSION
ip-172-31-29-63   NotReady   control-plane   3m    v1.29.0
ubuntu@ip-172-31-29-63:~$
```

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

```
ubuntu@ip-172-31-29-63:~$ sudo systemctl restart kubelet
ubuntu@ip-172-31-29-63:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
cp: overwrite '/home/ubuntu/.kube/config'? y
ubuntu@ip-172-31-29-63:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-fl
annel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
ubuntu@ip-172-31-29-63:~$
```

```
ubuntu@ip-172-31-29-63:~$ kubeadm token create --print-join-command
kubeadm join 172.31.29.63:6443 --token avprdu.7xw3ox1x6l9w6zfw --discovery-token-ca-cert-hash sha256:ff6e3056ea0d41a5989
19b1be9dbe765739c40a5aa6d37b5e4cbc45b1256c1c7
ubuntu@ip-172-31-29-63:~$
```

Only on Worker nodes

- Join Worker Nodes to the Cluster

```
ubuntu@ip-172-31-20-115:~$ sudo kubeadm reset pre-flight checks
W0915 18:24:16.207647    3366 preflight.go:56] [reset] WARNING: Changes made to this host by 'kubeadm init' or 'kubeadm
join' will be reverted.
[reset] Are you sure you want to proceed? [y/N]: y
[preflight] Running pre-flight checks
W0915 18:24:24.608981    3366 removeetcdmember.go:106] [reset] No kubeadm config, using etcd pod spec to get data direct
ory
[reset] Deleted contents of the etcd data directory: /var/lib/etcd
[reset] Stopping the kubelet service
[reset] Unmounting mounted directories in "/var/lib/kubelet"
[reset] Deleting contents of directories: [/etc/kubernetes/manifests /var/lib/kubelet /etc/kubernetes/pki]
[reset] Deleting files: [/etc/kubernetes/admin.conf /etc/kubernetes/super-admin.conf /etc/kubernetes/kubelet.conf /etc/k
ubernetes/bootstrap-kubelet.conf /etc/kubernetes/controller-manager.conf /etc/kubernetes/scheduler.conf]

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

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

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

- On the worker nodes, run the command provided by the master node during initialization . It looks something like this: sudo kubeadm join :6443 --token --discovery-token-ca-cert-hash sha256:

```
ubuntu@ip-172-31-20-115:~$ sudo  kubeadm join 172.31.29.63:6443 --token avprdu.7xw3ox1x6l9w6zfw --discovery-token-ca-cer
t-hash sha256:ff6e3056ea0d41a598919b1be9dbe765739c40a5aa6d37b5e4cbc45b1256c1c7 --v=5
I0915 18:26:15.816542    3382 join.go:413] [preflight] found NodeName empty; using OS hostname as NodeName
I0915 18:26:15.816650    3382 initconfiguration.go:122] detected and using CRI socket: unix:///var/run/containerd/contai
nerd.sock
[preflight] Running pre-flight checks
I0915 18:26:15.816686    3382 preflight.go:93] [preflight] Running general checks
I0915 18:26:15.816765    3382 checks.go:280] validating the existence of file /etc/kubernetes/kubelet.conf
I0915 18:26:15.816774    3382 checks.go:280] validating the existence of file /etc/kubernetes/bootstrap-kubelet.conf
I0915 18:26:15.816784    3382 checks.go:104] validating the container runtime
I0915 18:26:15.833043    3382 checks.go:639] validating whether swap is enabled or not
I0915 18:26:15.833129    3382 checks.go:370] validating the presence of executable crictl
I0915 18:26:15.833152    3382 checks.go:370] validating the presence of executable conntrack
I0915 18:26:15.833171    3382 checks.go:370] validating the presence of executable ip
I0915 18:26:15.833203    3382 checks.go:370] validating the presence of executable iptables
I0915 18:26:15.833227    3382 checks.go:370] validating the presence of executable mount
I0915 18:26:15.833242    3382 checks.go:370] validating the presence of executable nsenter
I0915 18:26:15.833276    3382 checks.go:370] validating the presence of executable ebtables
I0915 18:26:15.833292    3382 checks.go:370] validating the presence of executable ethtool
I0915 18:26:15.833320    3382 checks.go:370] validating the presence of executable socat
I0915 18:26:15.833340    3382 checks.go:370] validating the presence of executable tc
```

```
ootstrap-kubelet.conf
I0915 18:26:15.933343    3382 kubelet.go:136] [kubelet-start] writing CA certificate at /etc/kubernetes/pki/ca.crt
I0915 18:26:15.933723    3382 kubelet.go:157] [kubelet-start] Checking for an existing Node in the cluster with name "ip
-172-31-20-115" and status "Ready"
I0915 18:26:15.936219    3382 kubelet.go:172] [kubelet-start] Stopping the kubelet
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...
I0915 18:26:17.103583    3382 cert_rotation.go:137] Starting client certificate rotation controller
I0915 18:26:17.104245    3382 kubelet.go:220] [kubelet-start] preserving the crisocket information for the node
I0915 18:26:17.104261    3382 patchnode.go:31] [patchnode] Uploading the CRI Socket information "unix:///var/run/contain
erd/containerd.sock" to the Node API object "ip-172-31-20-115" as an annotation

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

ubuntu@ip-172-31-20-115:~$
```

- Verify the Cluster
  Once the worker node joins, check the status on the master node

```
ubuntu@ip-172-31-29-63:~$ kubectl get nodes
NAME               STATUS   ROLES           AGE   VERSION
ip-172-31-20-115   Ready    <none>          27m   v1.29.0
ip-172-31-20-200   Ready    <none>          24m   v1.29.0
ip-172-31-29-63    Ready    control-plane   55m   v1.29.0
ubuntu@ip-172-31-29-63:~$
```

PRACTICAL NO: - 4

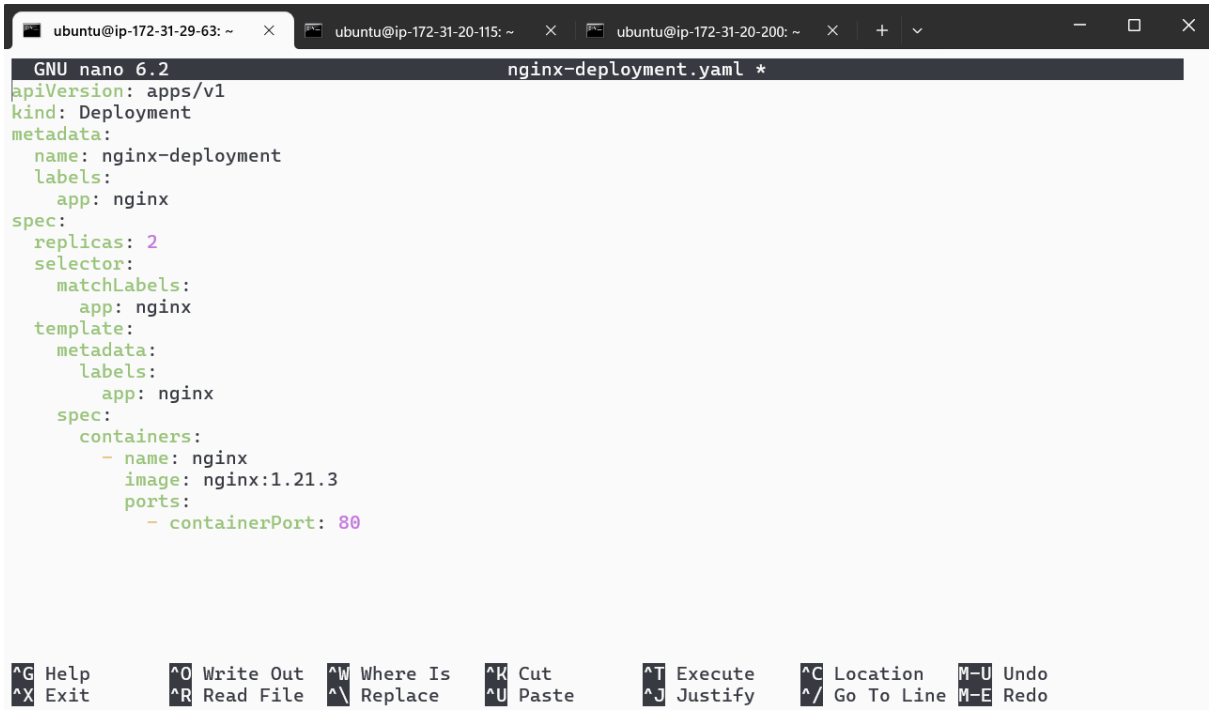
In previous practical, we have setup the Kubernetes cluster

➢ Deploying Your Application on Kubernetes

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

```
ubuntu@ip-172-31-29-63:~$ nano nginx-deployment.yaml
ubuntu@ip-172-31-29-63:~$
```

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).



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

```
ubuntu@ip-172-31-29-63:~$ nano nginx-service.yaml
ubuntu@ip-172-31-29-63:~$
```

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

```
ubuntu@ip-172-31-29-63: ~   ubuntu@ip-172-31-20-115: ~   ubuntu@ip-172-31-20-200: ~
  GNU nano 6.2                                  nginx-service.yaml
apiVersion:v1
kind: Service
metadata:
   name:nginx-service
spec:
  selector:
    app: nginx
  ports:
       - protocol: TCP
         port: 80
         targetPort: 80
  type: LoadBalancer
```

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

```
ubuntu@ip-172-31-29-63:~$ kubectl apply -f nginx-deployment.yaml --validate=false
deployment.apps/nginx-deployment created

ubuntu@ip-172-31-29-63:~$ kubectl apply -f nginx-service.yaml --validate=false
service/nginx-service created
ubuntu@ip-172-31-29-63:~$
```

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

```
ubuntu@ip-172-31-29-63:~$ kubectl get deployments
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   1/2     2            1           9m25s
ubuntu@ip-172-31-29-63:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS      AGE
nginx-deployment-6b4d6fdbf-6w4bm    1/1     Running   4 (98s ago)   9m18s
nginx-deployment-6b4d6fdbf-bhcwm    1/1     Running   4 (70s ago)   9m18s
ubuntu@ip-172-31-29-63:~$ kubectl get services
NAME            TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes      ClusterIP      10.96.0.1       <none>        443/TCP        111m
nginx-service   LoadBalancer   10.110.88.111   <pending>     80:30132/TCP   110s
ubuntu@ip-172-31-29-63:~$
```

Describe the deployment(Extra)

```
ubuntu@ip-172-31-29-63:~$ kubectl get deployments
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   2/2     2            2           11m
ubuntu@ip-172-31-29-63:~$ kubectl describe deployment
Name:                   nginx-deployment
Namespace:              default
CreationTimestamp:      Sun, 15 Sep 2024 19:39:41 +0000
Labels:                 app=nginx
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               app=nginx
Replicas:               2 desired | 2 updated | 2 total | 1 available | 1 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginx
  Containers:
   nginx:
    Image:        nginx:1.21.3
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Progressing    True    NewReplicaSetAvailable
  Available      False   MinimumReplicasUnavailable
OldReplicaSets:  <none>
NewReplicaSet:   nginx-deployment-6b4d6fdbf (2/2 replicas created)
Events:
  Type    Reason             Age   From                   Message
  ----    ------             ----  ----                   -------
  Normal  ScalingReplicaSet  11m   deployment-controller  Scaled up replica set nginx-deployment-6b4d6fdbf to 2
ubuntu@ip-172-31-29-63:~$
```

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

```
ubuntu@ip-172-31-29-63:~$ kubectl get service
NAME            TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes      ClusterIP      10.96.0.1       <none>        443/TCP        114m
nginx-service   LoadBalancer   10.110.88.111   <pending>     80:30132/TCP   4m59s
ubuntu@ip-172-31-29-63:~$
```

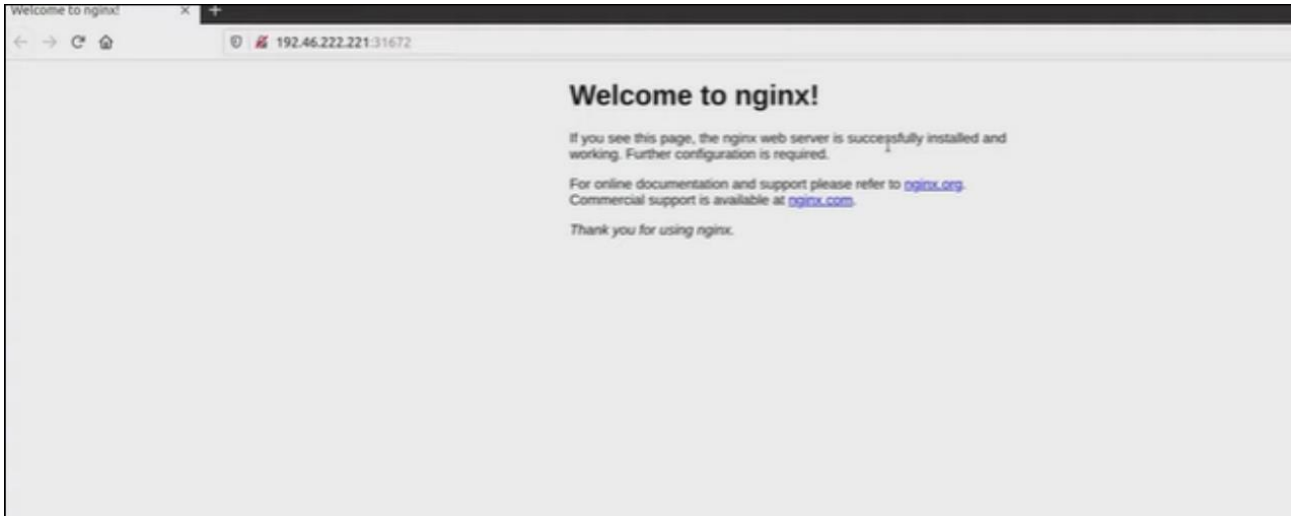
Forward the Service Port to Your Local Machine

```
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
```

```
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
```

Access the Application Locally

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.



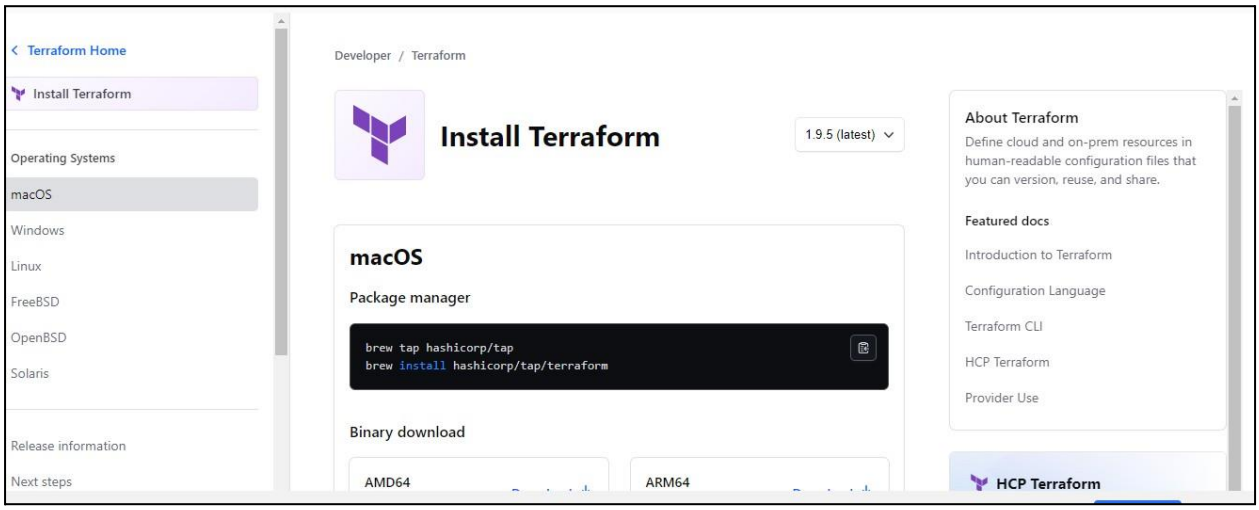
ADVANCE DEVOPS EXP 5

Name:Laksh Vijay Sodhai
Class:D15A
Roll No:59

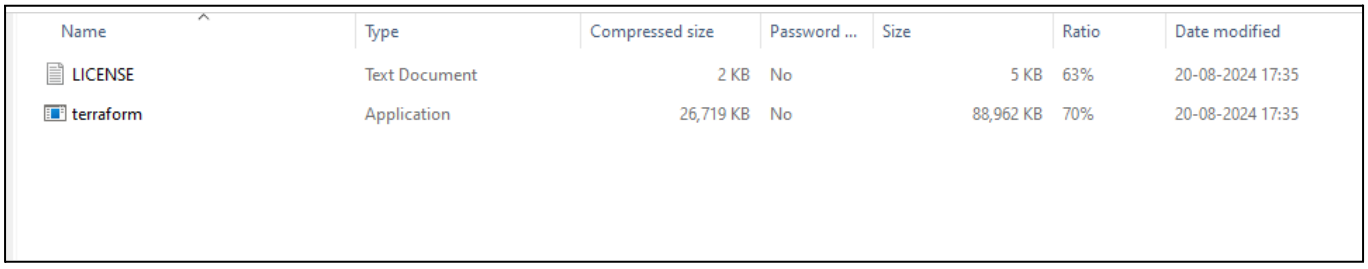
Aim:To understand terraform lifecycle, core concepts/terminologies and install it on a Linux Machine and Windows.

Installation for Windows:

Step 1: Go to the website terraform.io and install Terraform from there..Select the AMD64 option for Windows and download Terraform.

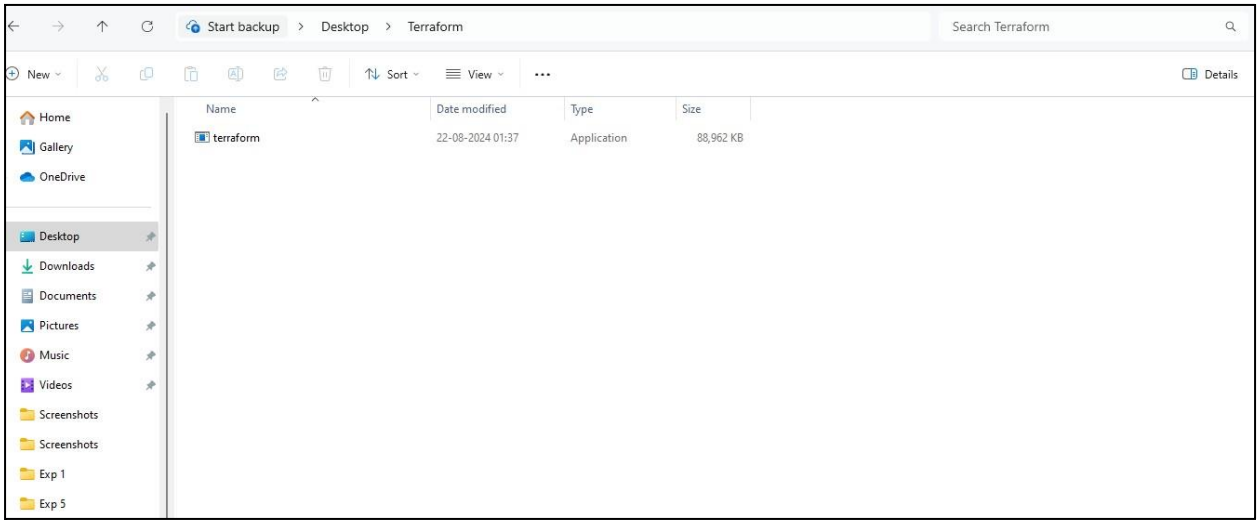


Step 2: Go to the zip file where Terraform is installed.

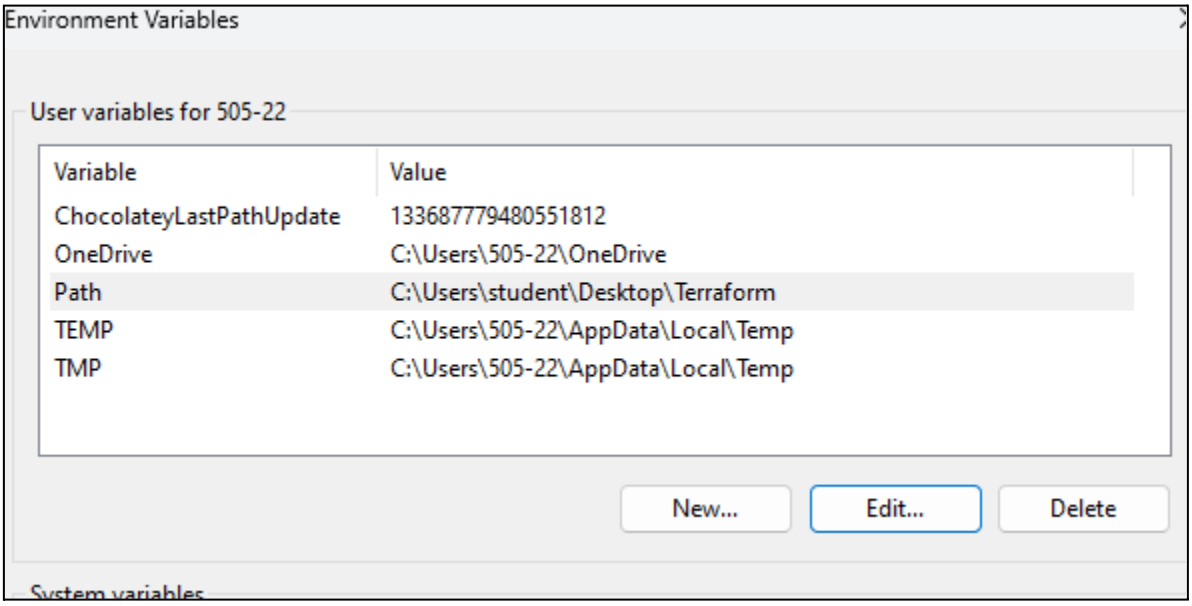


Name	Type	Compressed size	Password ...	Size	Ratio	Date modified
LICENSE	Text Document	2 KB	No	5 KB	63%	20-08-2024 17:35
terraform	Application	26,719 KB	No	88,962 KB	70%	20-08-2024 17:35

Step 3: Since the installed file is a zip file,create a new folder on desktop and copy the installed terraform application there.



Step 4: Now go to search bar,select edit environment variables option,then go to the path option.Now add the file path of the directory wherein we have installed the terraform application.



Step 5: Now go to the folder where we have installed terraform and open Powershell inside it.After this type ‘terraform’ to make sure that terraform has been installed on the system.

The command ‘terraform –version simply checks the current version of terraform that has been installed.

```
PS C:\Users\student\Desktop\Terraform> 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
  version       Show the current Terraform version
  workspace     Workspace management
```

```
Global options (use these before the subcommand, if any):
  -chdir=DIR    Switch to a different working directory before executing the
                given subcommand.
  -help         Show this help output, or the help for a specified subcommand.
  -version      An alias for the "version" subcommand.
PS C:\Users\student\Desktop\Terraform> terraform --version
Terraform v1.9.4
on windows_amd64

Your version of Terraform is out of date! The latest version
is 1.9.5. You can update by downloading from https://www.terraform.io/downloads.html
PS C:\Users\student\Desktop\Terraform>
```

ADVANCE DEVOPS EXP 6

Name:Laksh Vijay Sodhai
Class:D15A
Roll No:59

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

Prerequisite:
1) Download and Install Docker Desktop from https://www.docker.com/

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>
```

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

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

Script:

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

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

# 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"]
```

}

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.
```

Docker images, Before Executing Apply step:

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

Docker images, 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:Laksh Sodhai
Class:D15A
Roll No:59

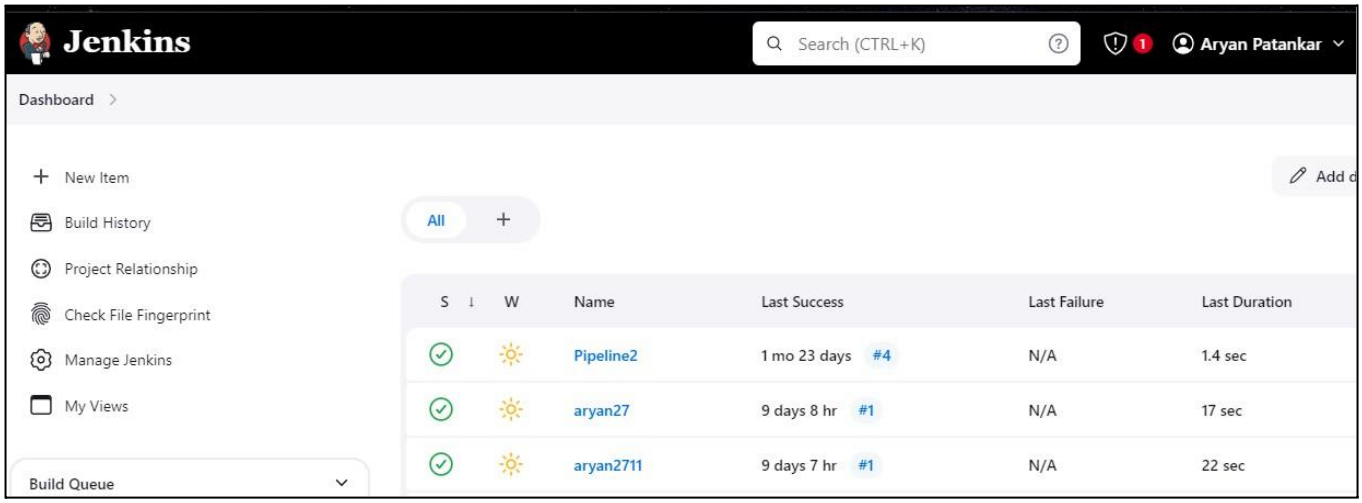
Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

Integrating Jenkins with SonarQube:

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

Steps to integrate Jenkins with SonarQube

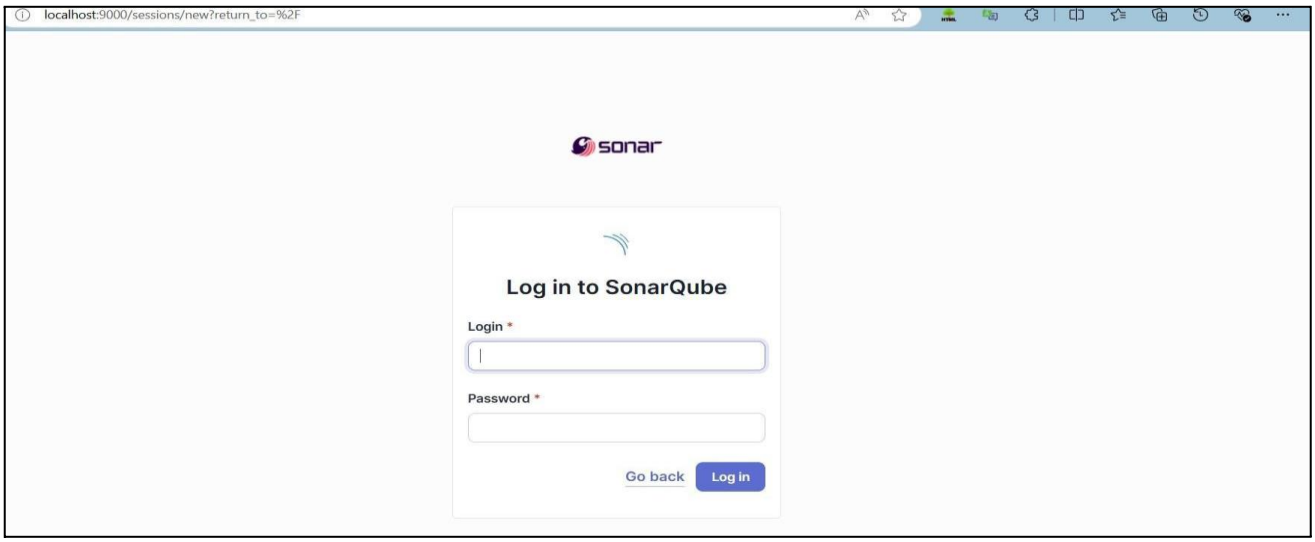
1. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.



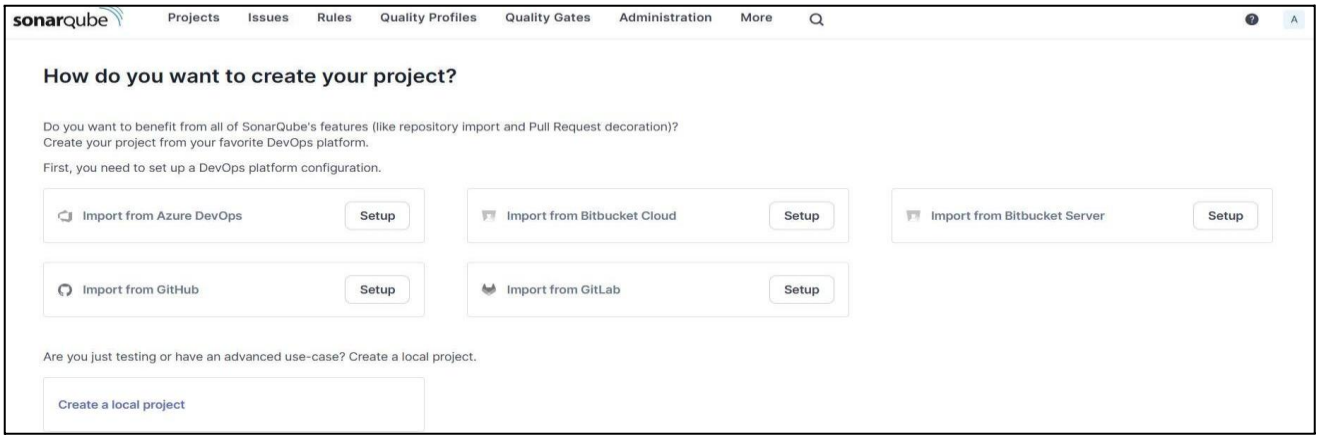
2. Run SonarQube in a Docker container using this command -
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

```
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
```

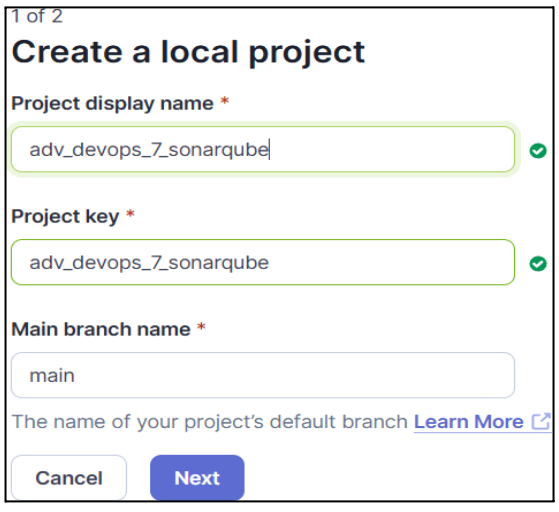
3. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.

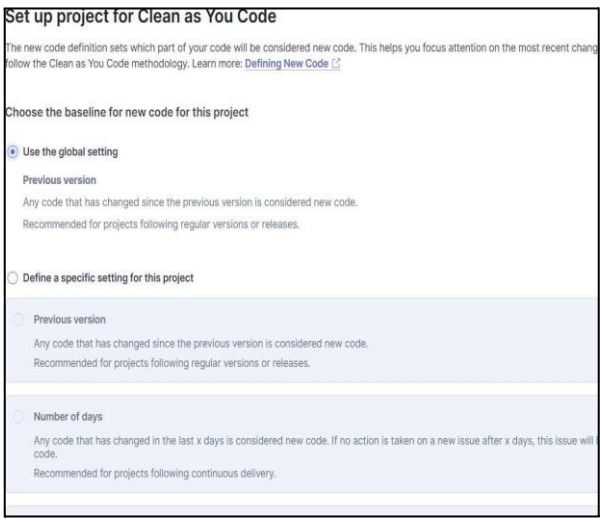


4. Login to SonarQube using username admin and password admin.

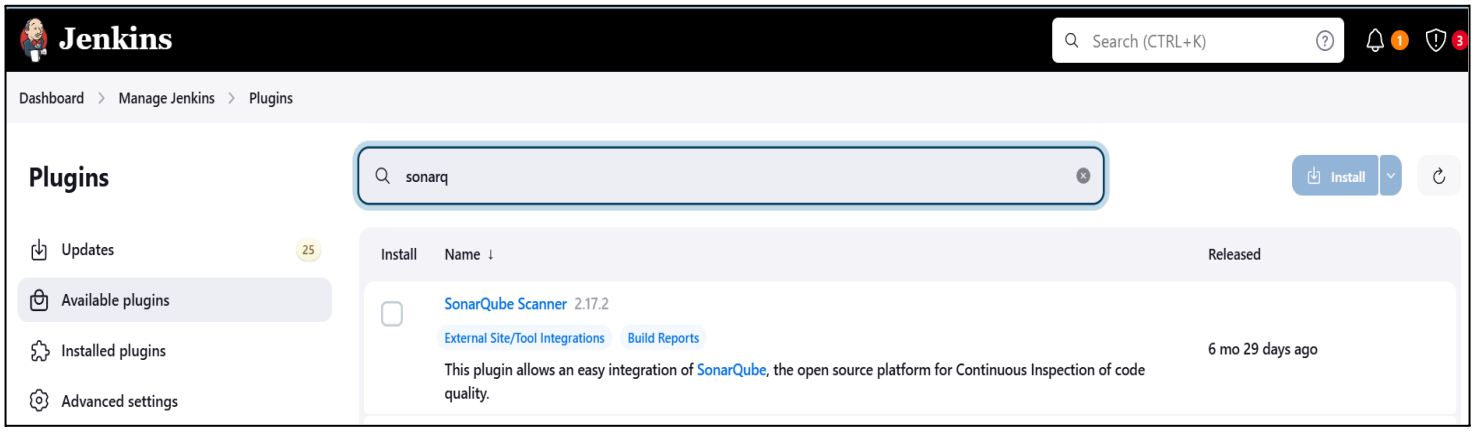


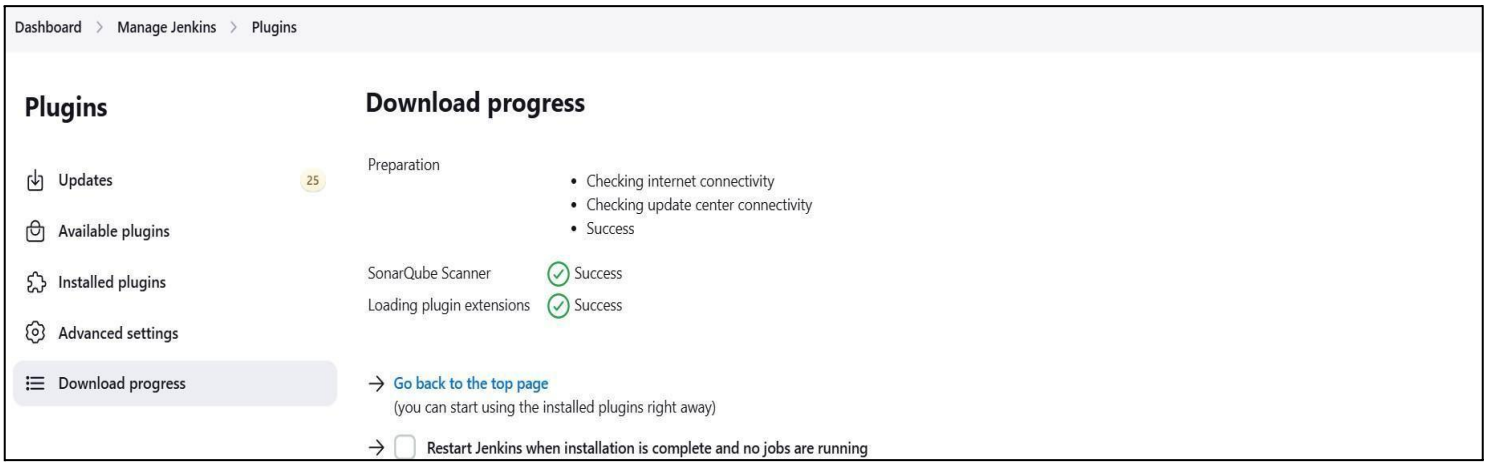
5. Create a manual project in SonarQube with the name sonarqube





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.

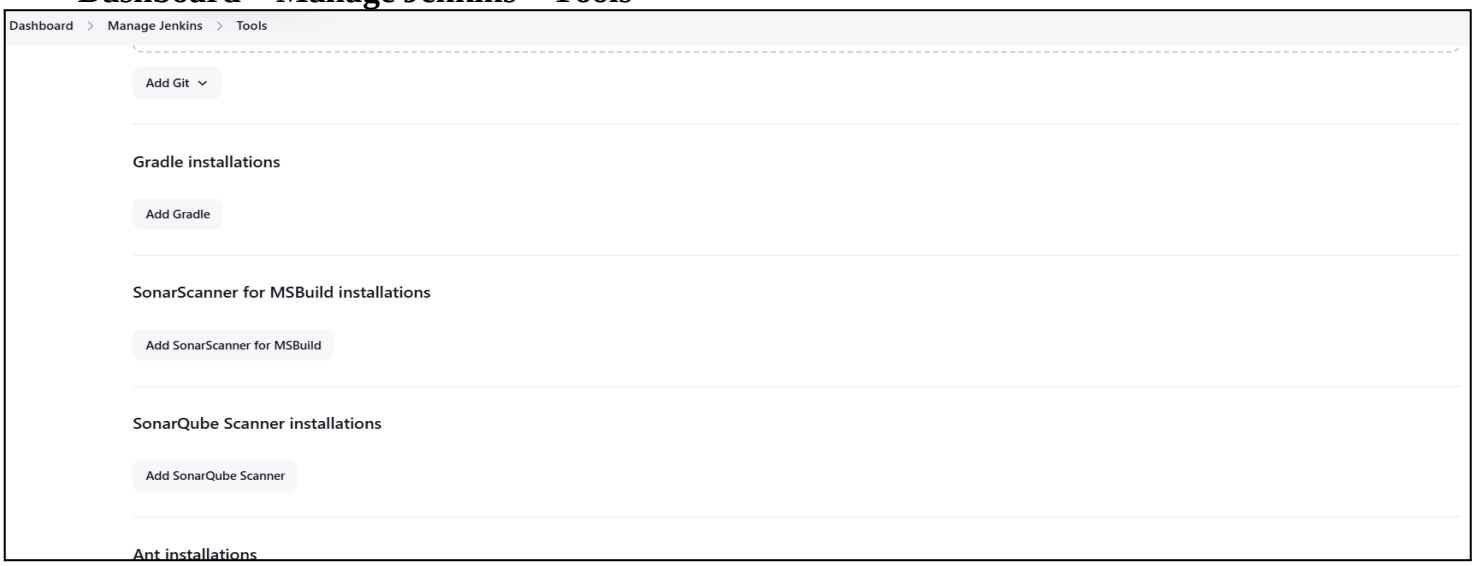
In SonarQube installations: Under **Name** add <project name of sonarqube>,here we have named it as **adv_devops_7_sonarqube**
In **Server URL** Default is **http://localhost:9000**


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

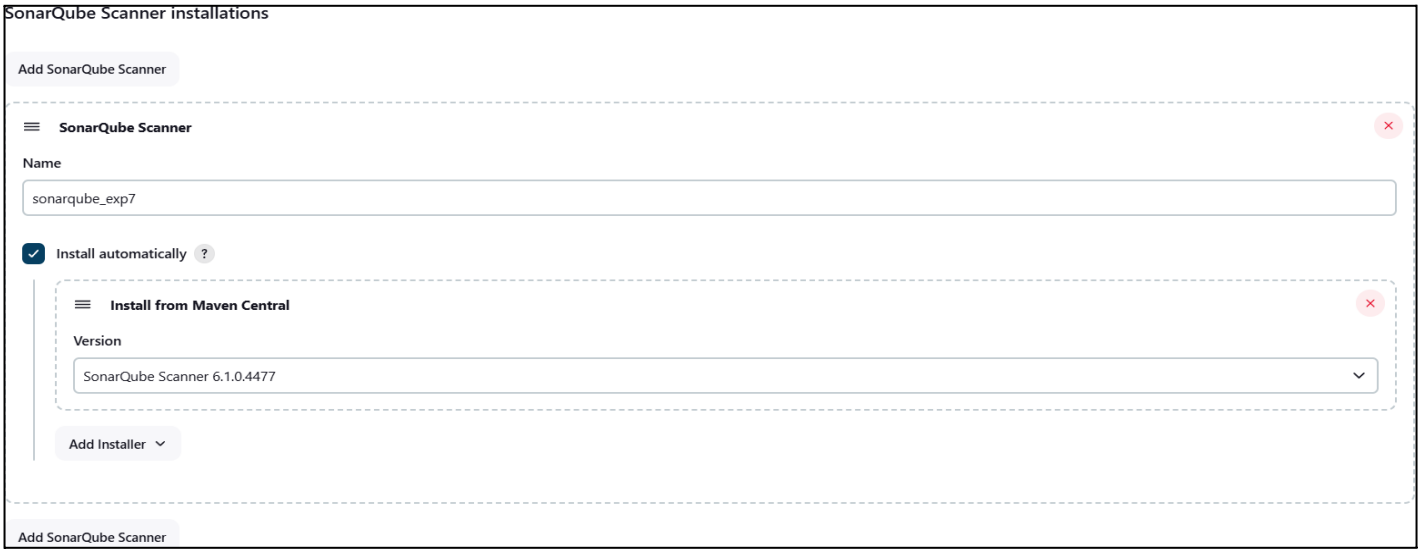
7. Search for SonarQube Scanner under Global Tool Configuration.

Choose the latest configuration and choose Install automatically.

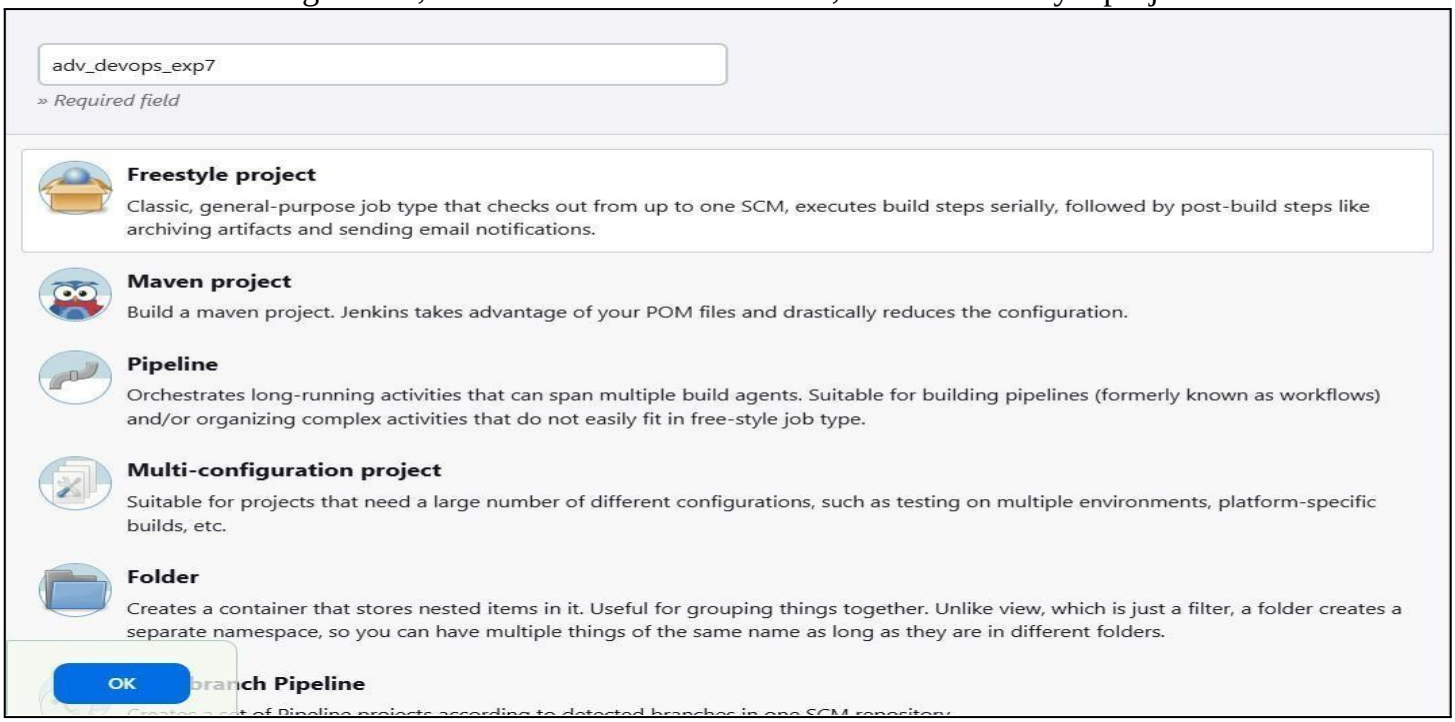
Dashboard > Manage Jenkins > Tools



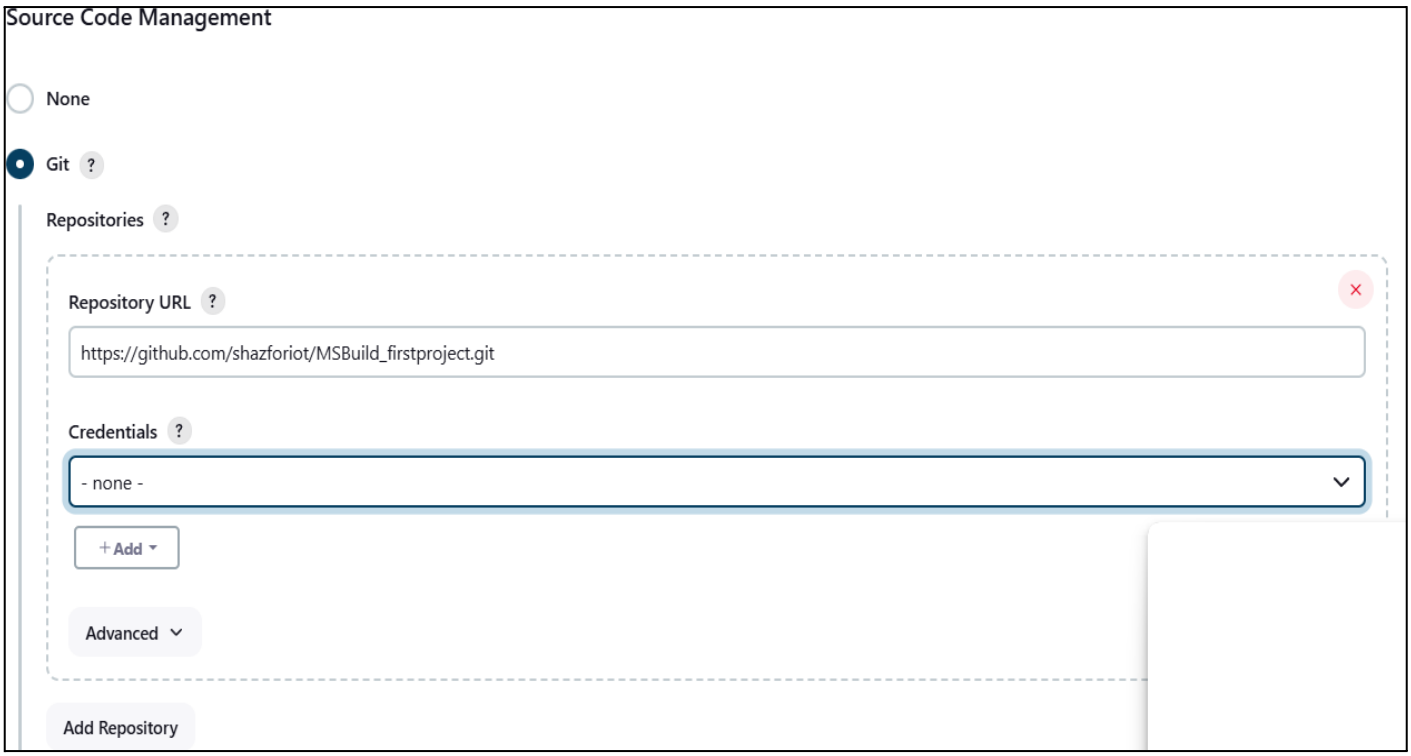
Check the “Install automatically” option. → Under name any name as identifier → Check the “Install automatically” option.



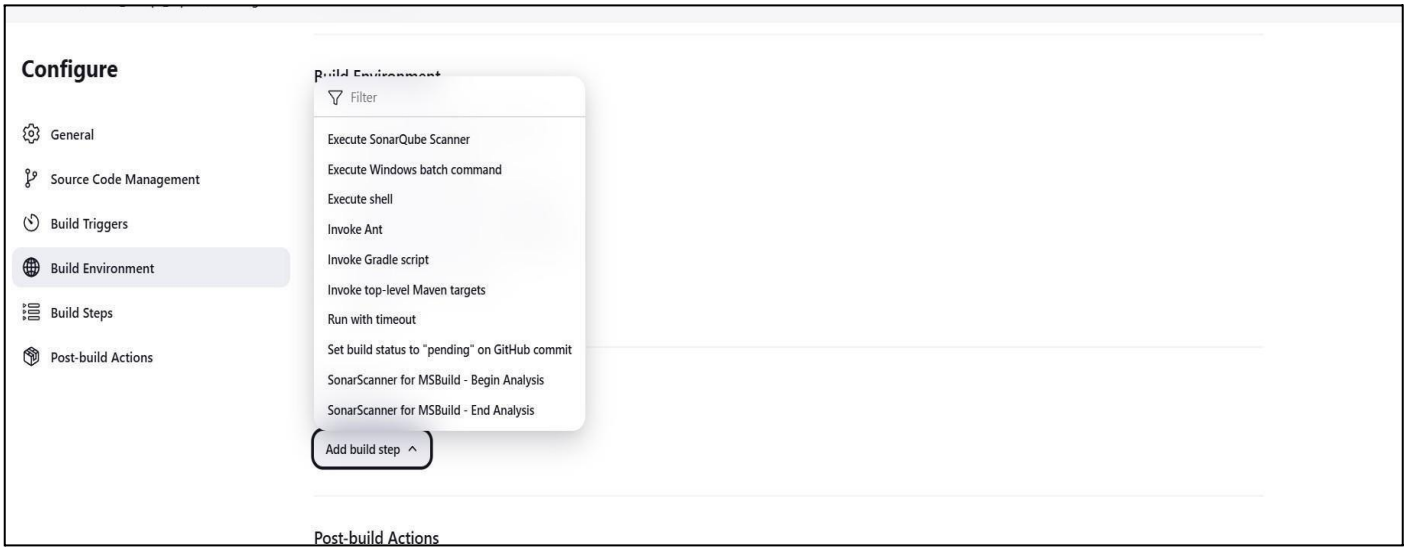
8. After the configuration, create a New Item in Jenkins, choose a freestyle project.



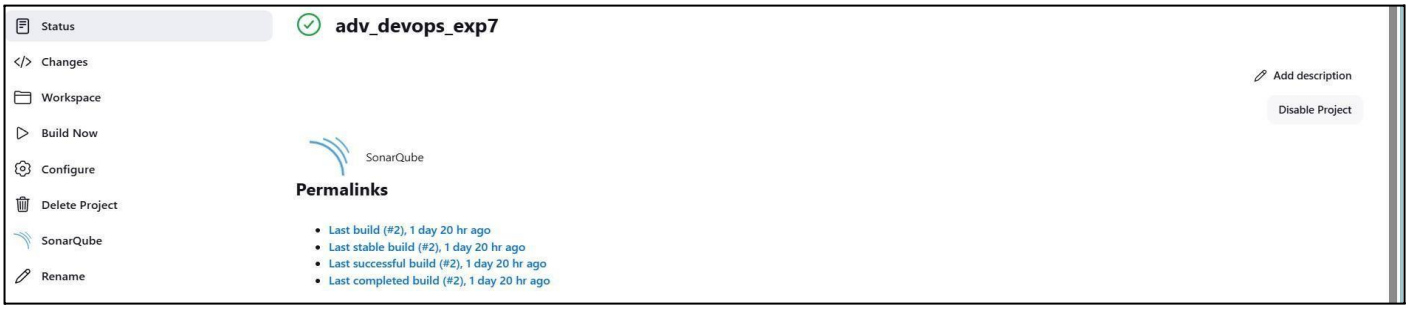
9. Choose this GitHub repository in Source Code Management.
https://github.com/shazforiot/MSBuild_firstproject.git
It is a sample hello-world project with no vulnerabilities and issues, just to test the integration.



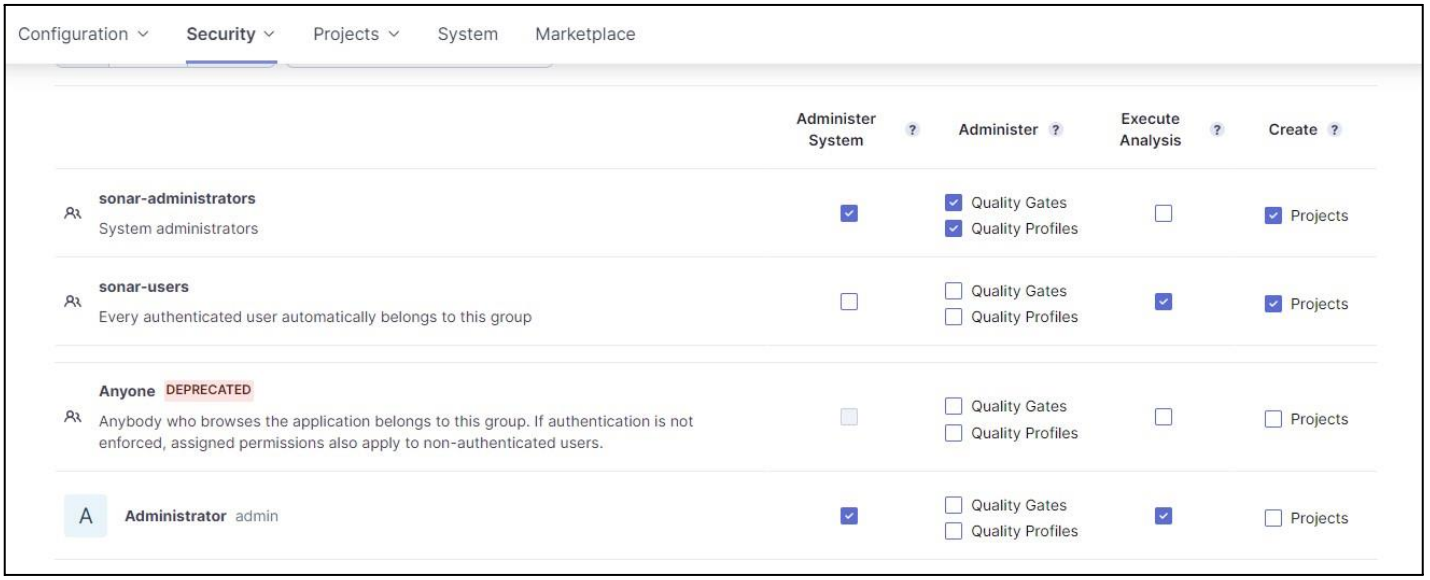
10. Under **Select project → Configuration → Build steps → Execute SonarQube Scanner**, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.



Then save



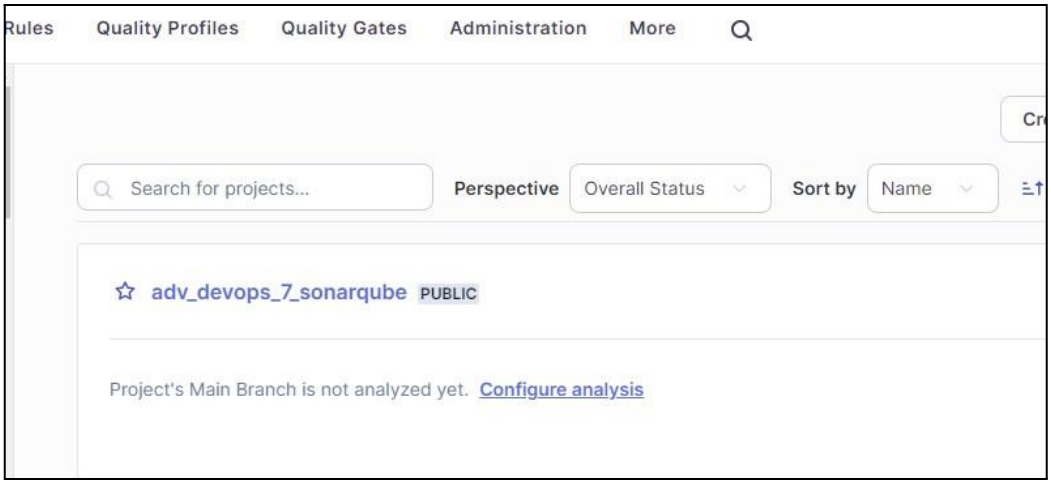
11. Go to http://localhost:9000/<user_name>/permissions and allow Execute Permissions to the Admin user



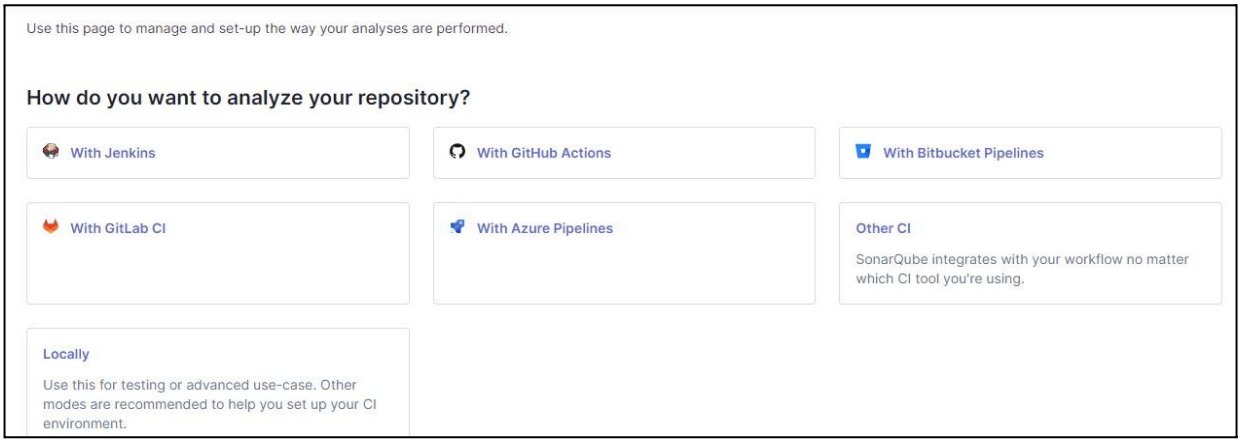
IF CONSOLE OUTPUT FAILED:

Step 1: Generate a New Authentication Token in SonarQube

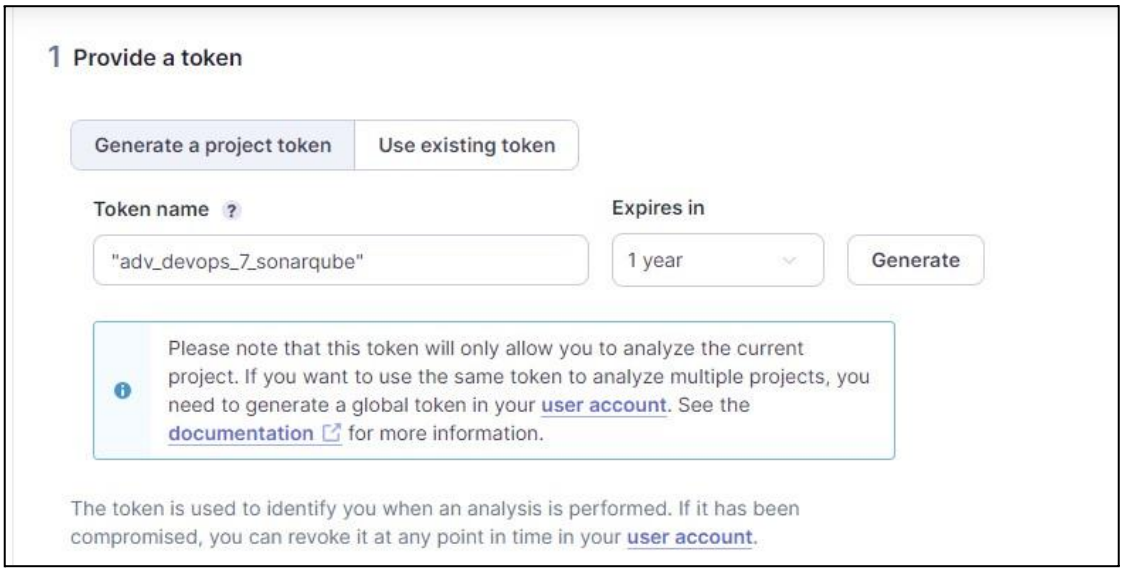
1. **Login to SonarQube:**
    - Open your browser and go to **http://localhost:9000**.
    - Log in with your admin credentials (default username is **admin**, and the password is either **admin** or your custom password if it was changed).
2. **Generate a New Token:**
    - Go to the project that you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



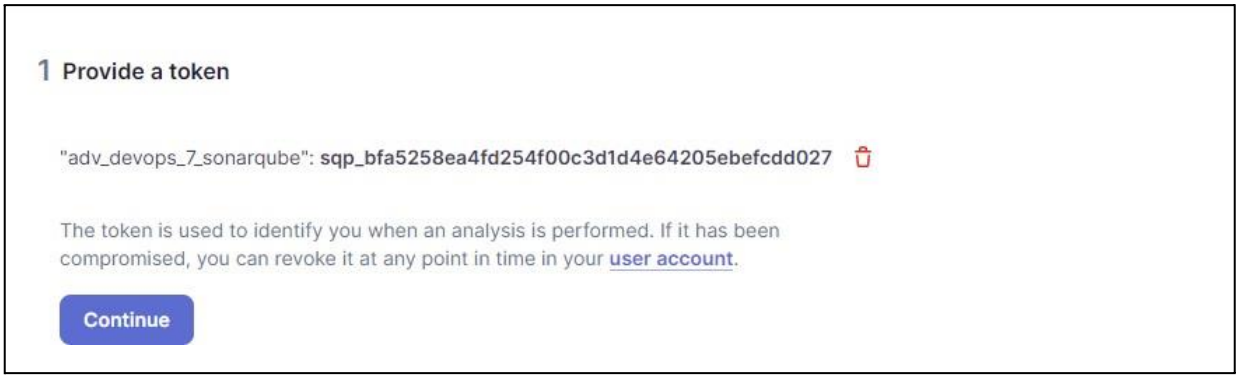
    - Click on **Locally**



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

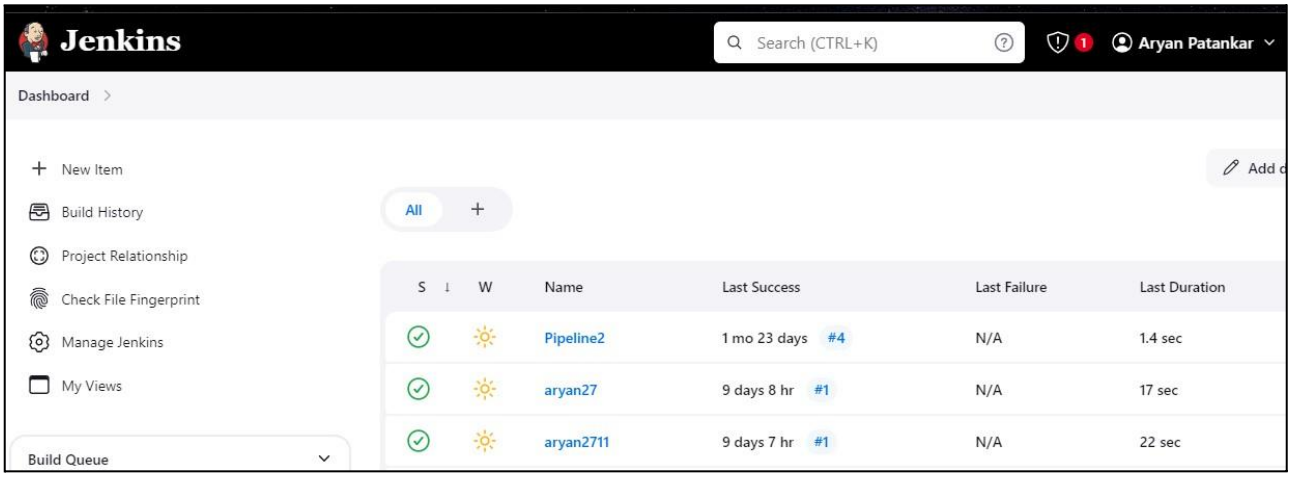


- Copy the token you get here and save it securely as we would need it in Jenkins.



Step 2: Update the Token in Jenkins

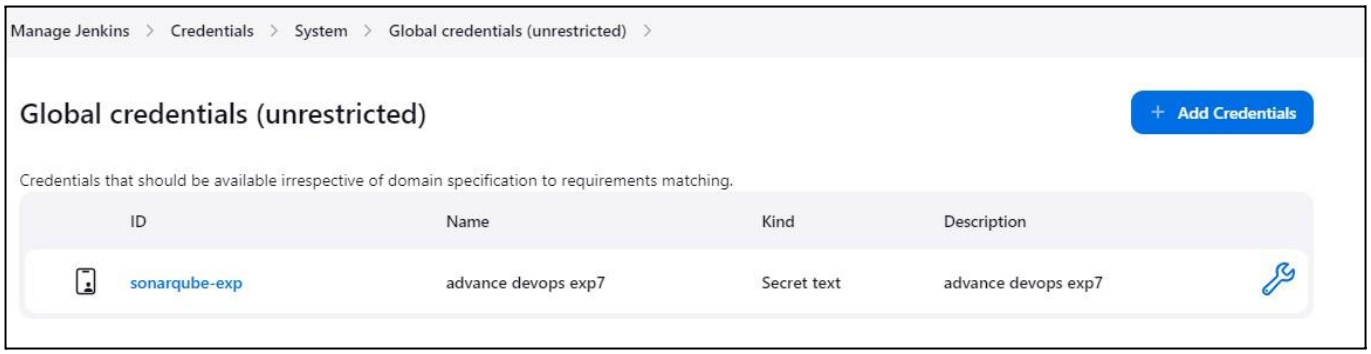
1. **Go to Jenkins Dashboard:**
   - Open Jenkins and log in with your credentials.



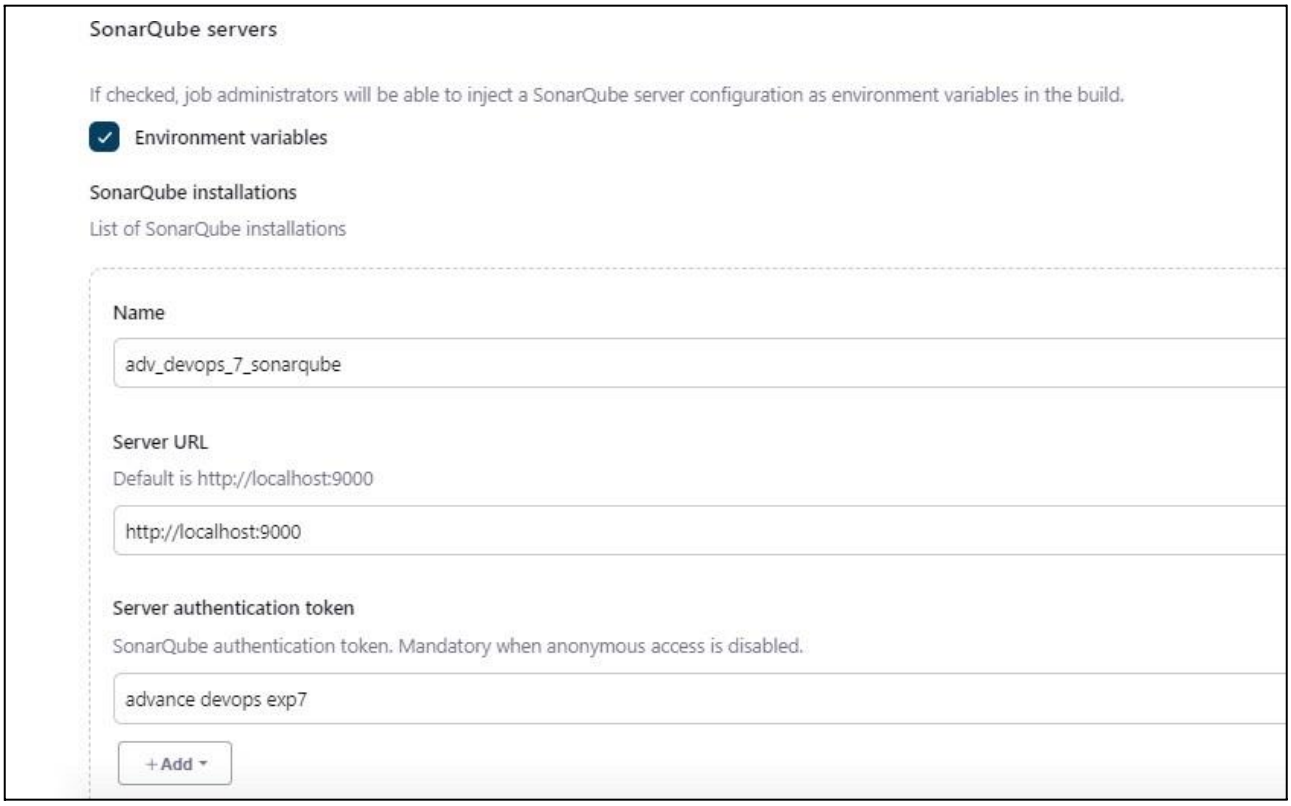
2. Go to Dashboard—>Manage Jenkins—>Credentials

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.

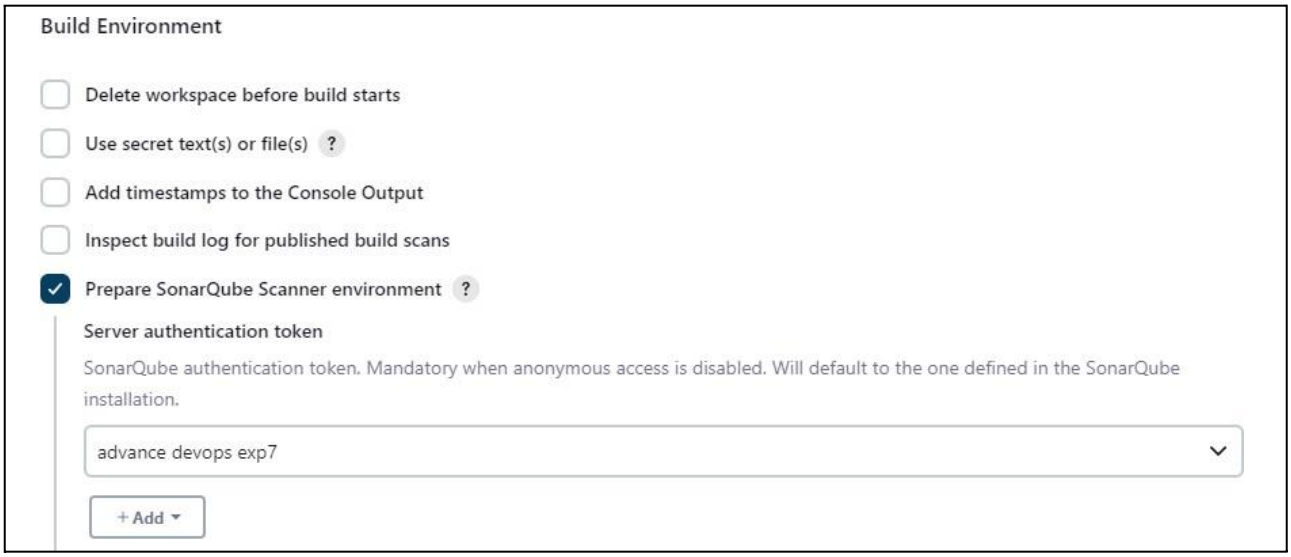
4. After clicking on create we see that the given token has been added in Jenkins credentials.



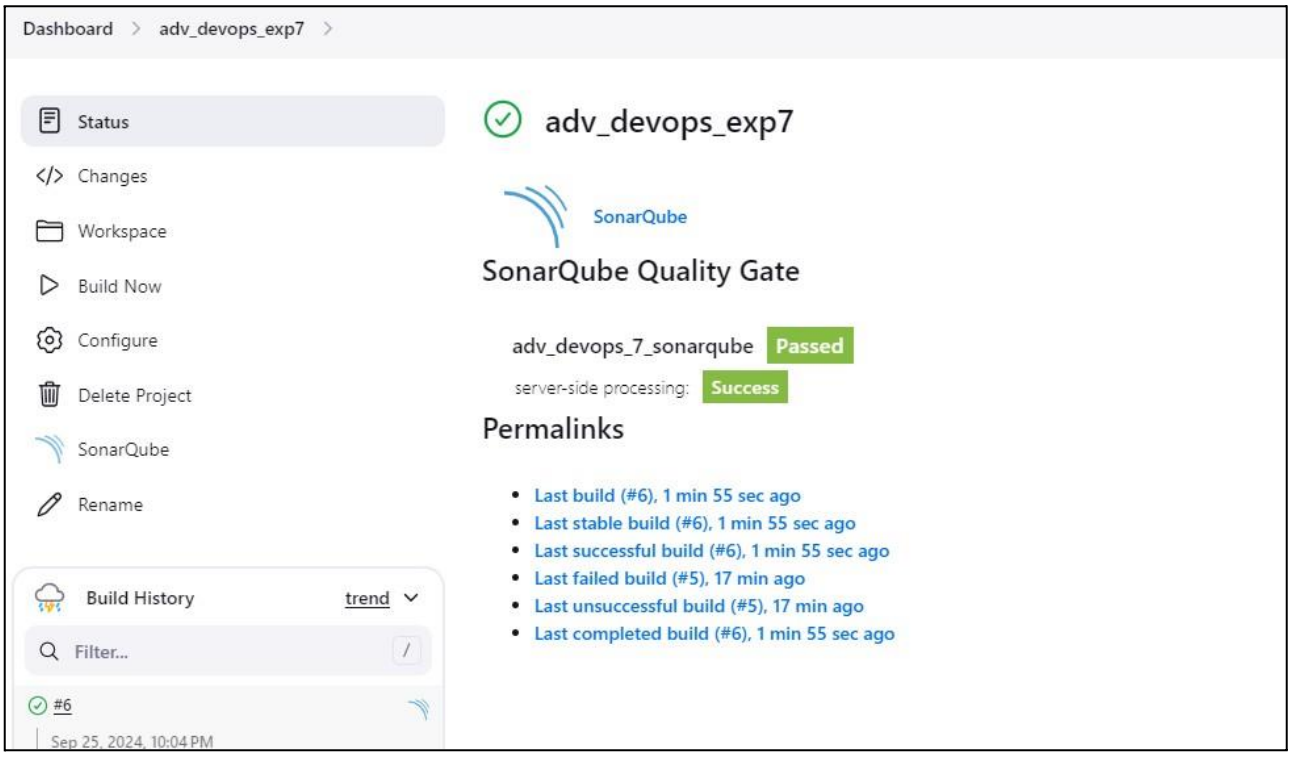
5.Now go to **Manage Jenkins—>System—>SonarQube servers** and proceed with the following details. Reference the authentication token generated in the previous step.



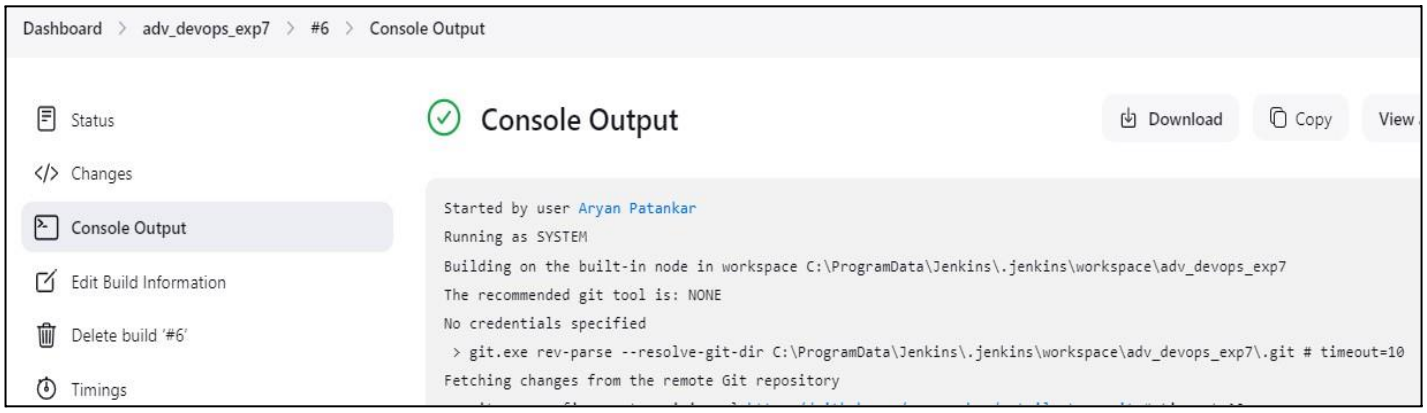
6.Check the SonarQube Scanner Environment and add the server authentication token



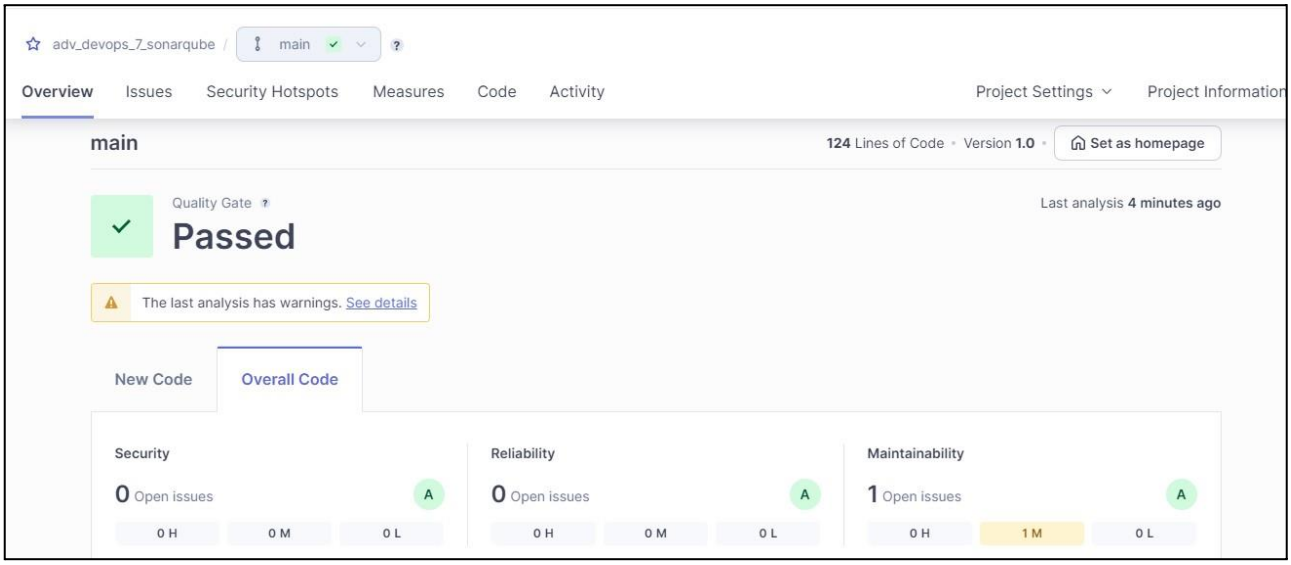
12. Run the Jenkins build.



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.

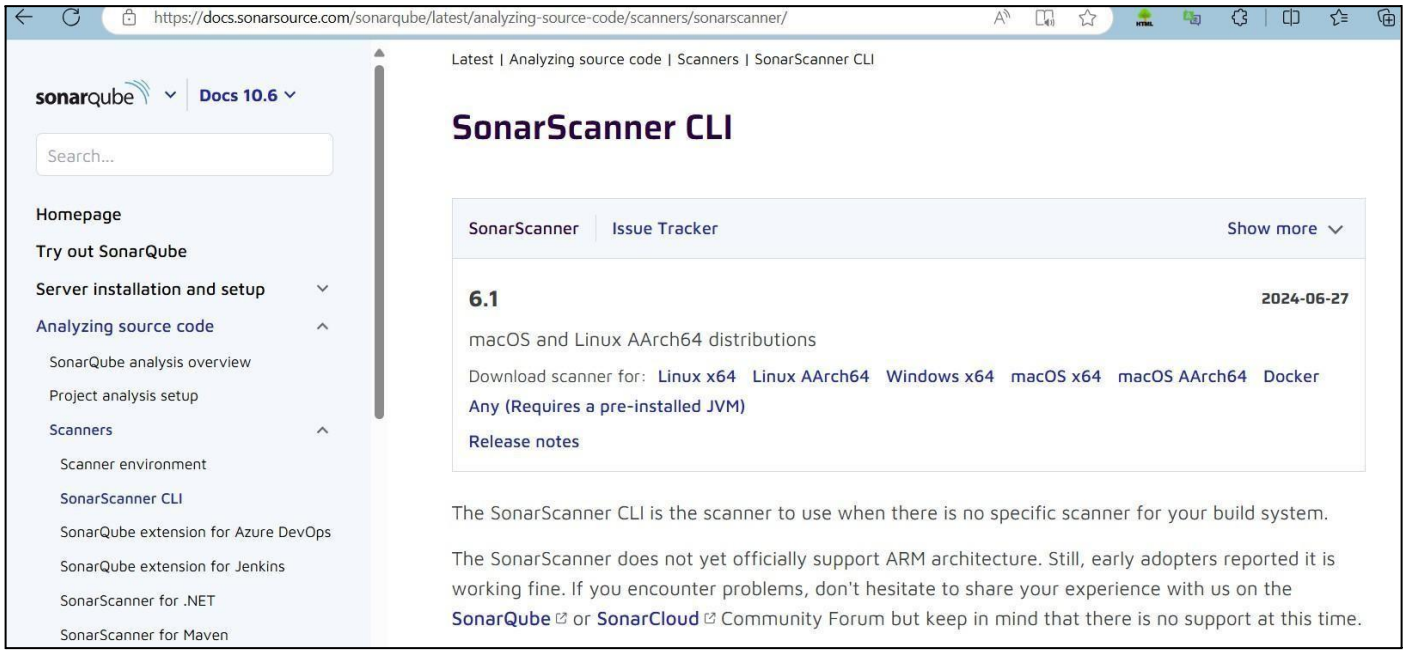
ADVANCE DEVOPS EXP 8

Name:Laksh Sodhai
Class:D15A
Roll No:59

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.

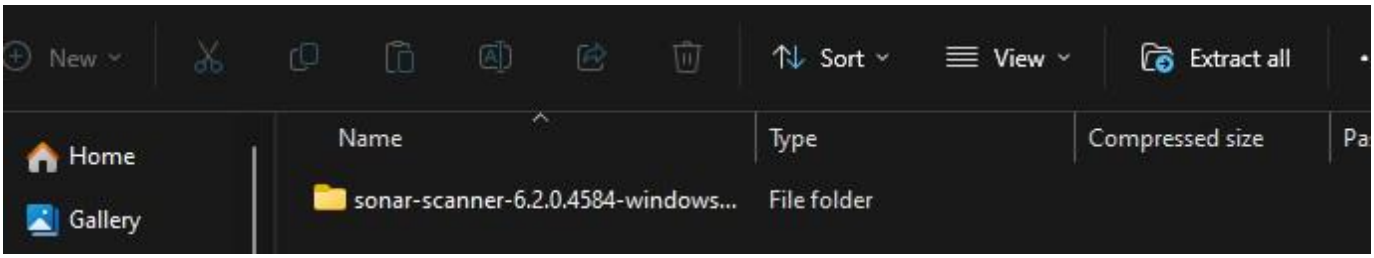
Step 1: Download sonar scanner

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



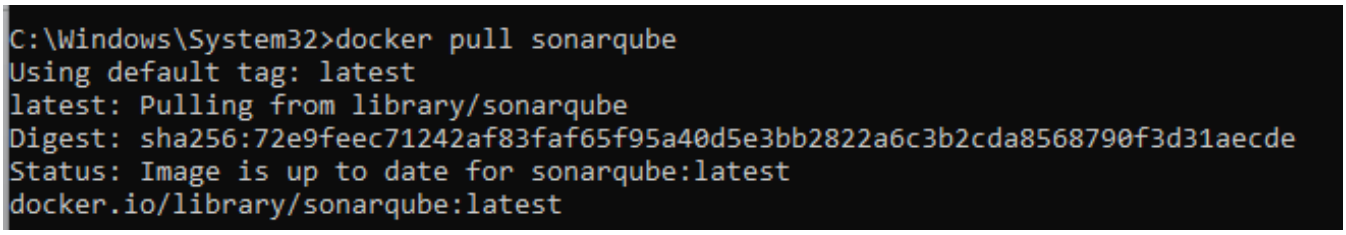
ner/ Visit this link and download the sonarqube scanner CLI.

Extract the downloaded zip file in a folder.

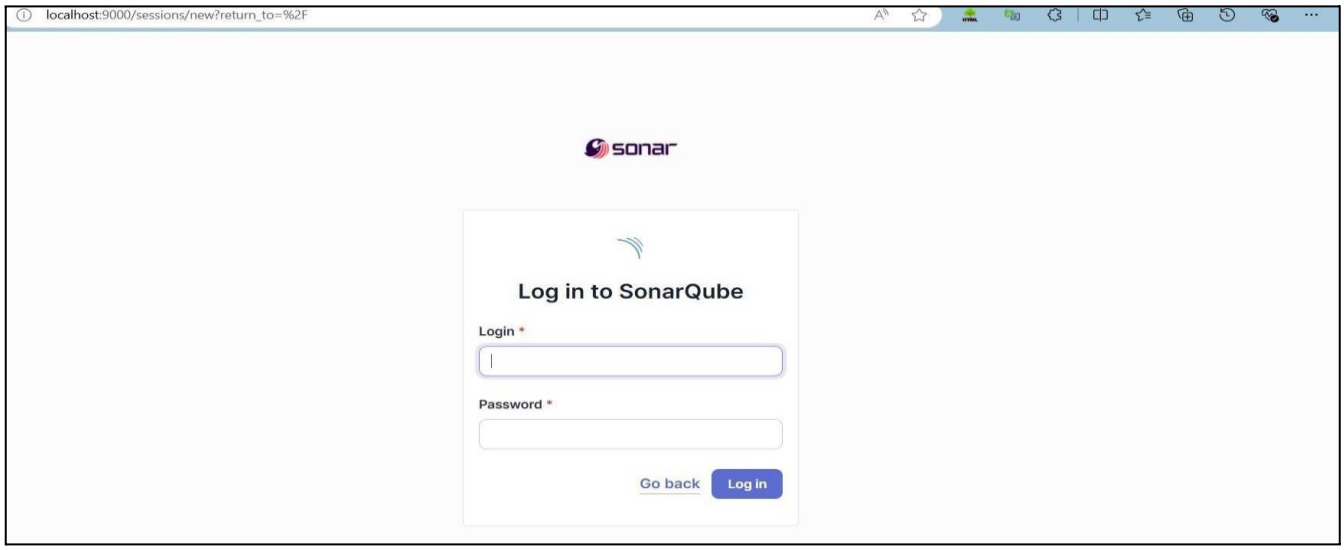


1. Install sonarqube image

Command: **docker pull sonarqube**

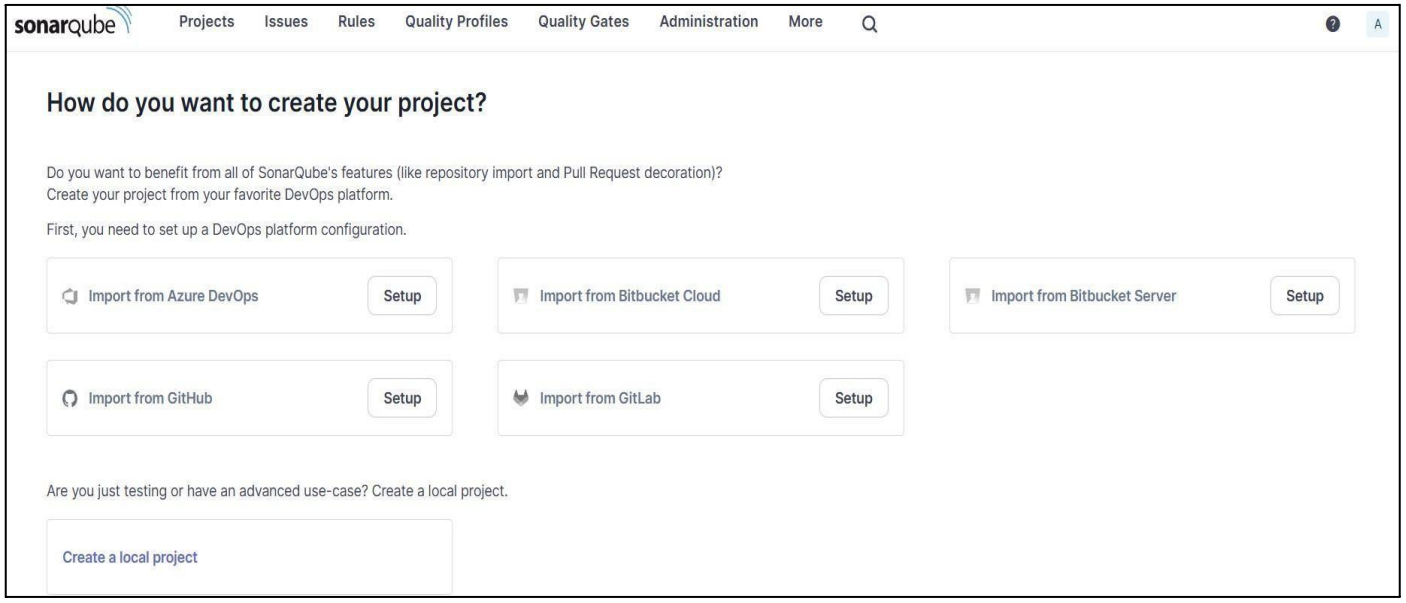


2. Once the container is up and running, you can check the status of

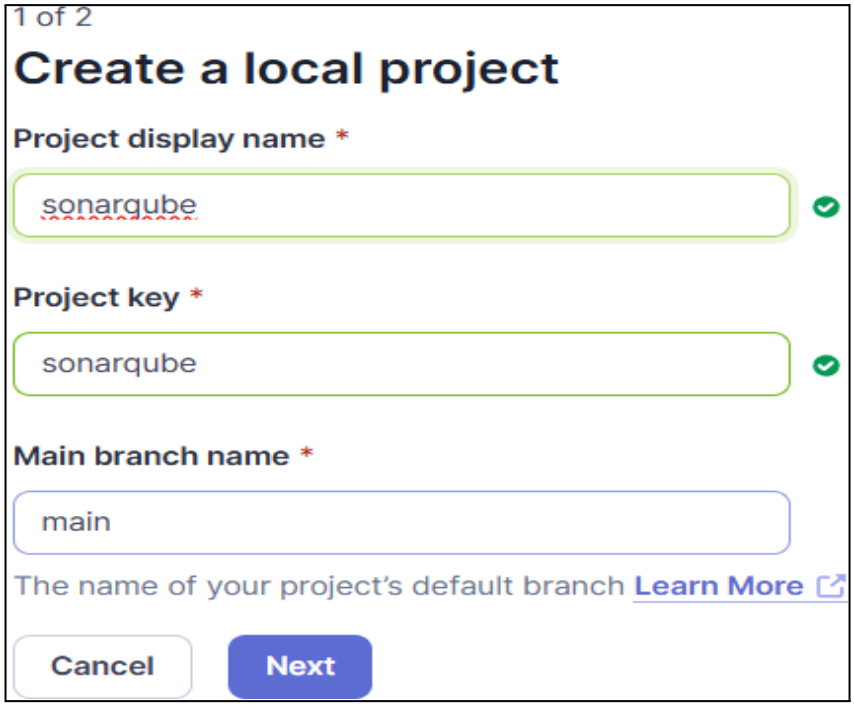


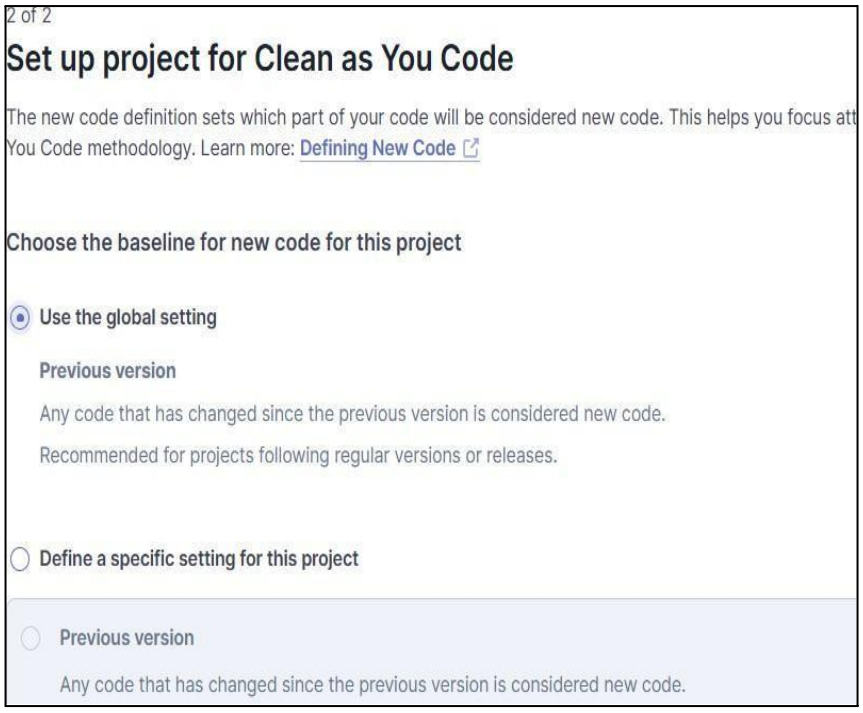
SonarQube at localhost port 9000.

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

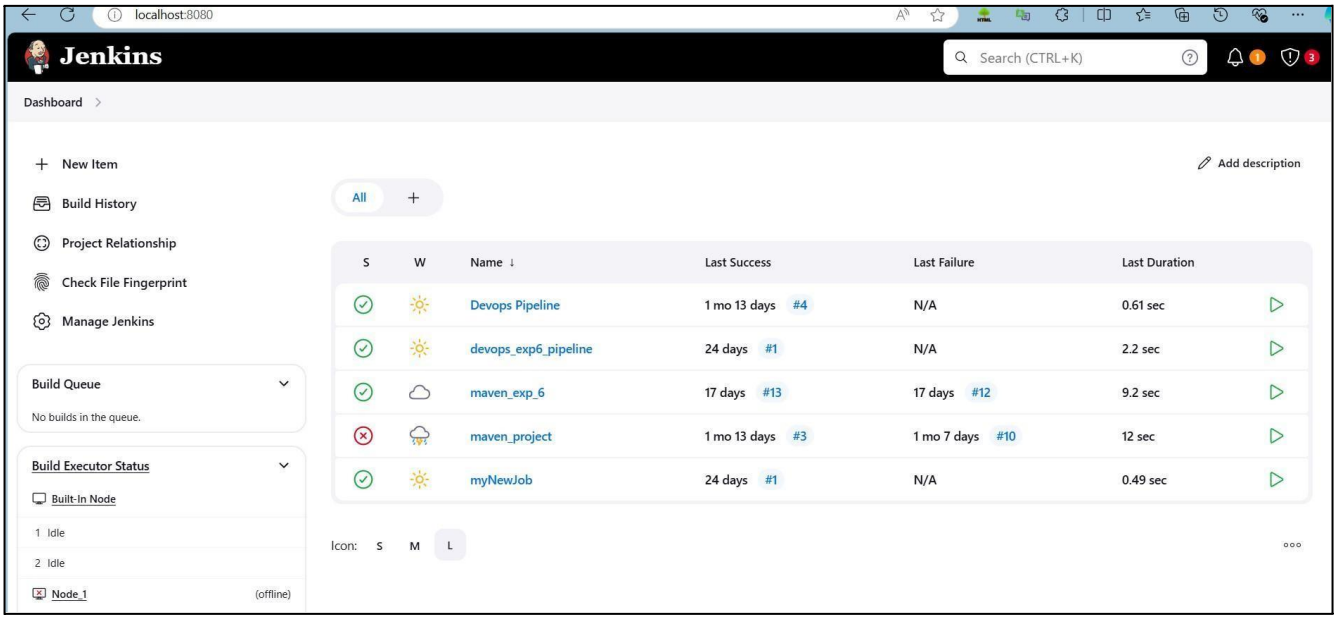


4. Create a manual project in SonarQube with the name sonarqube

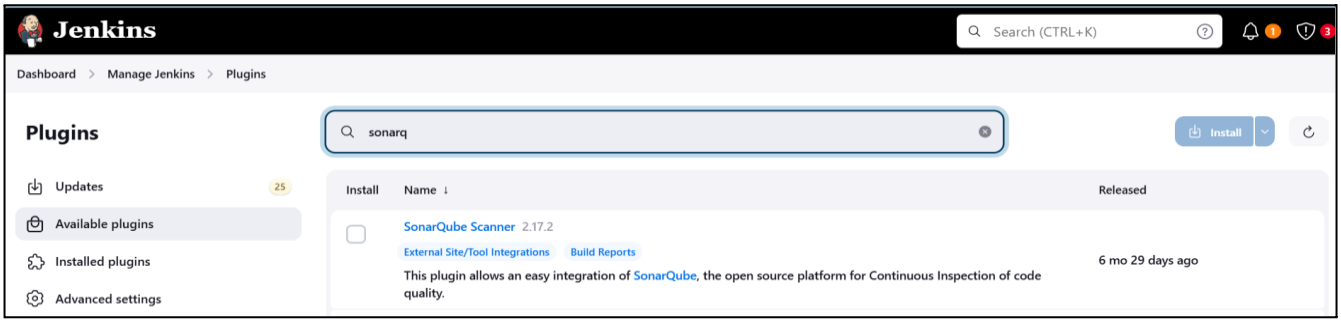


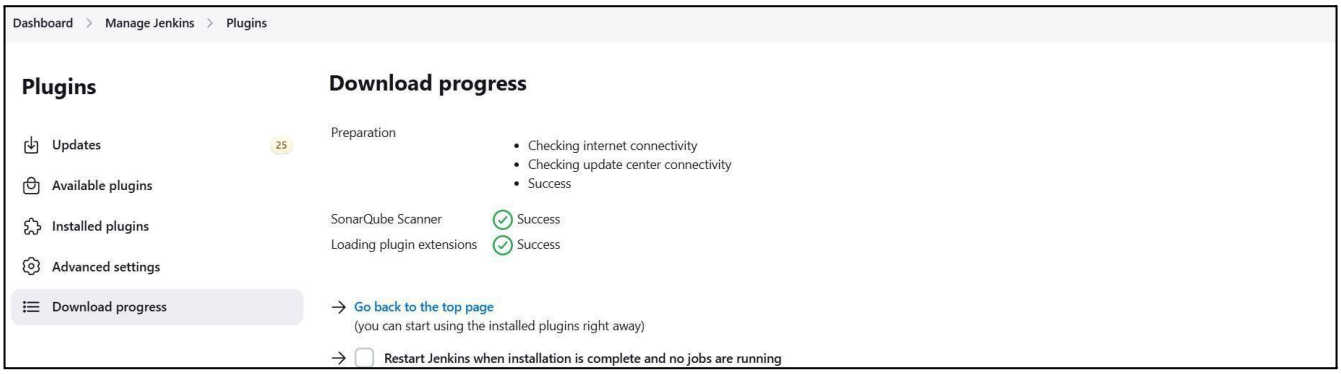


5. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.



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

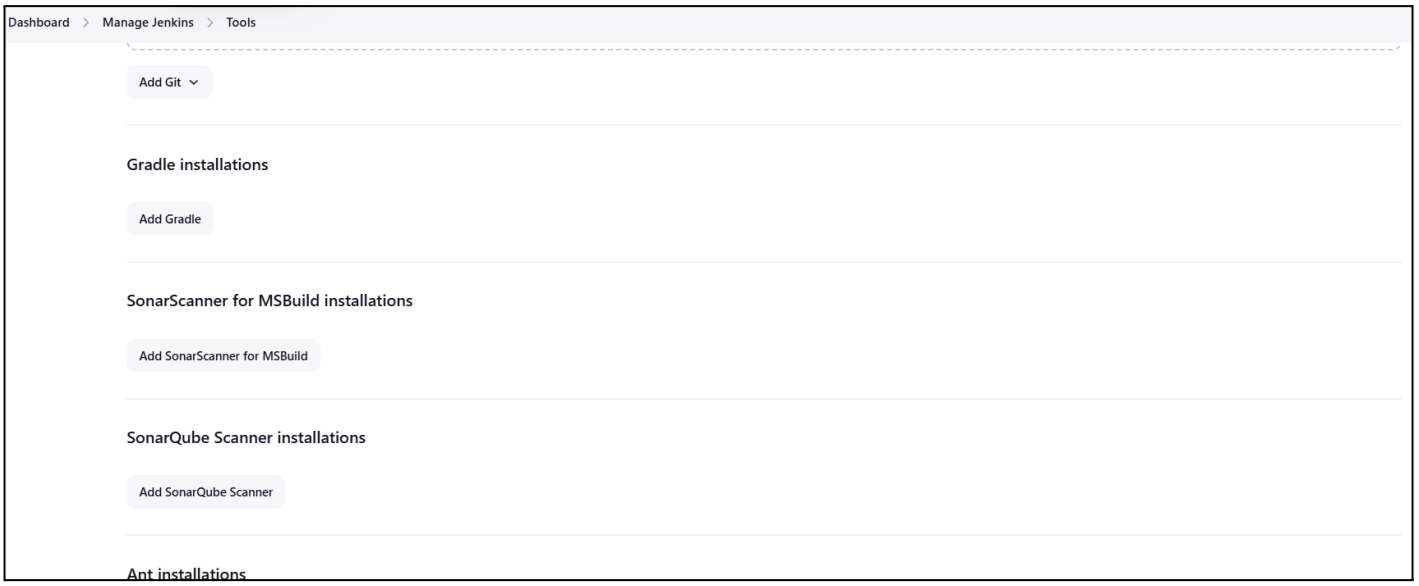




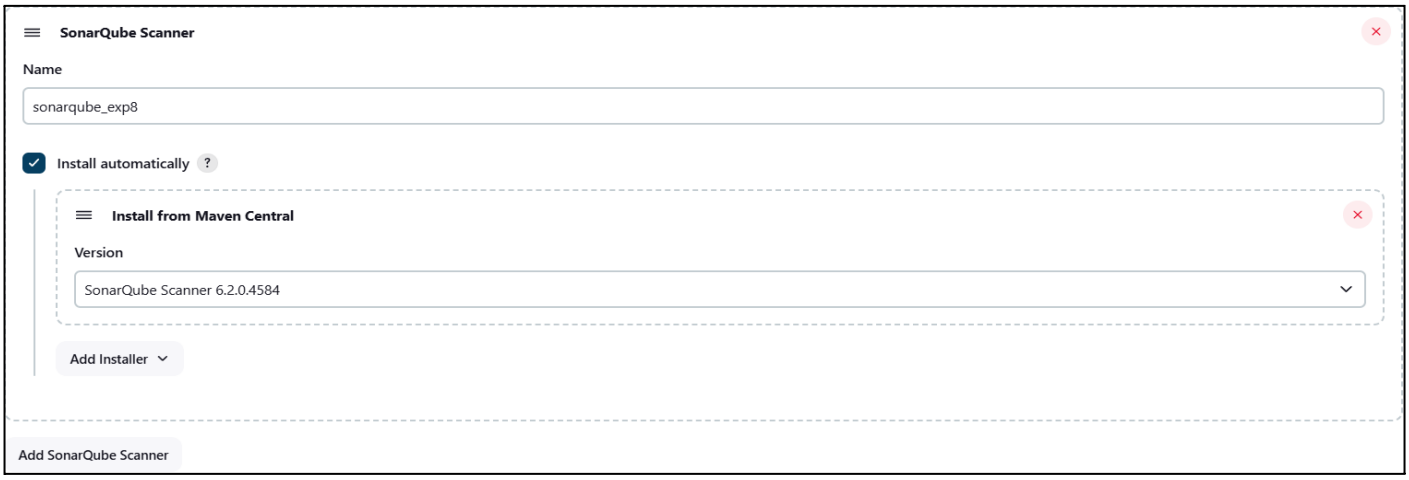
7. Under Jenkins ‘Manage Jenkins’ then go to ‘system’, scroll and look for **SonarQube Servers**
and enter the details.
Enter the Server Authentication token if needed.

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

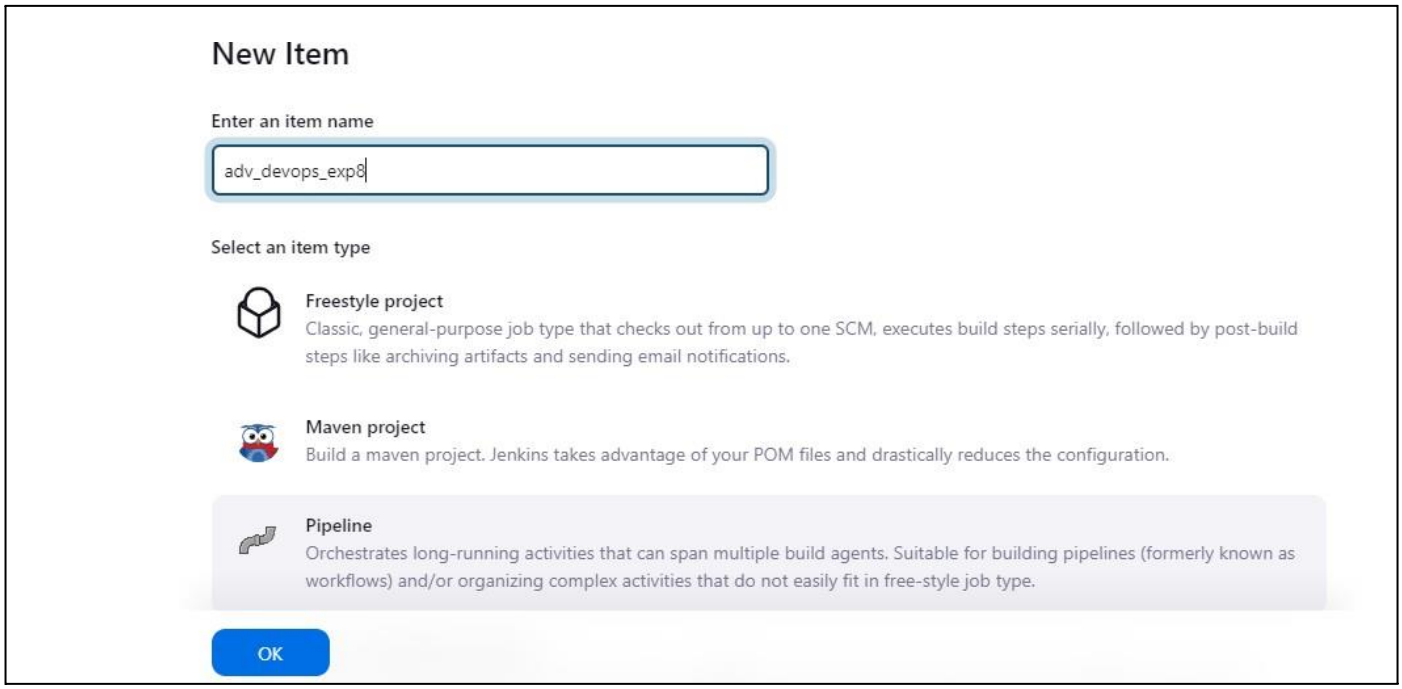


Check the “Install automatically” option. → Under name any name as identifier → Check



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/)
    """
  }
}
```

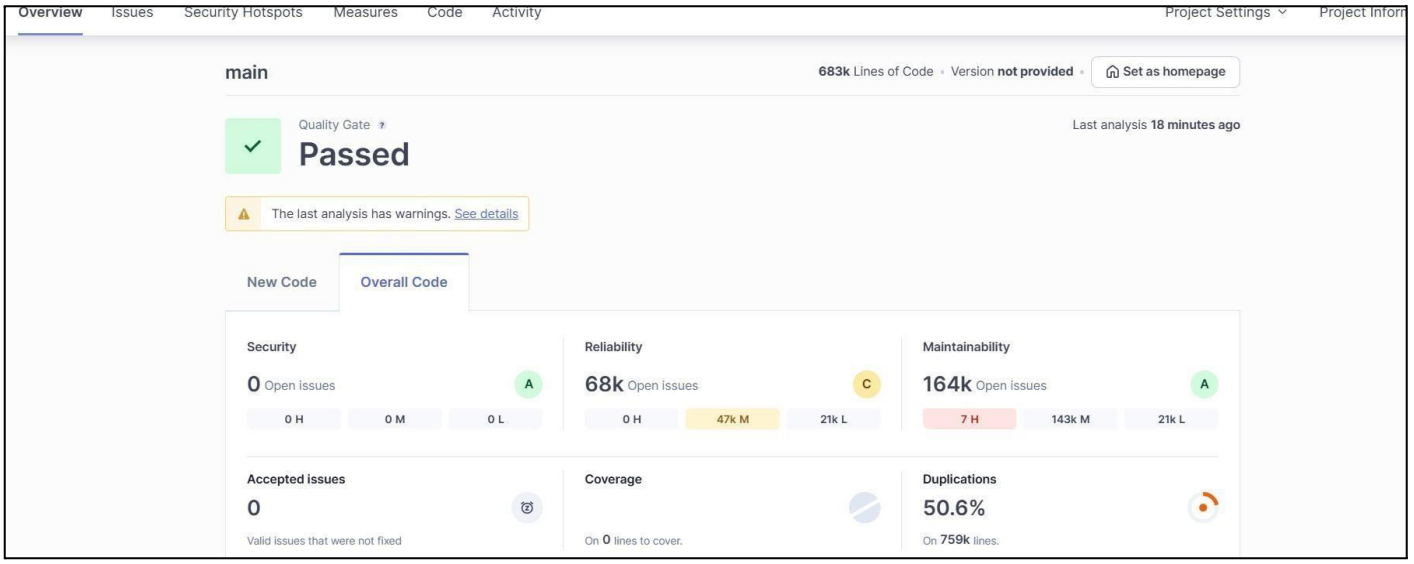
}

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

11. Build project

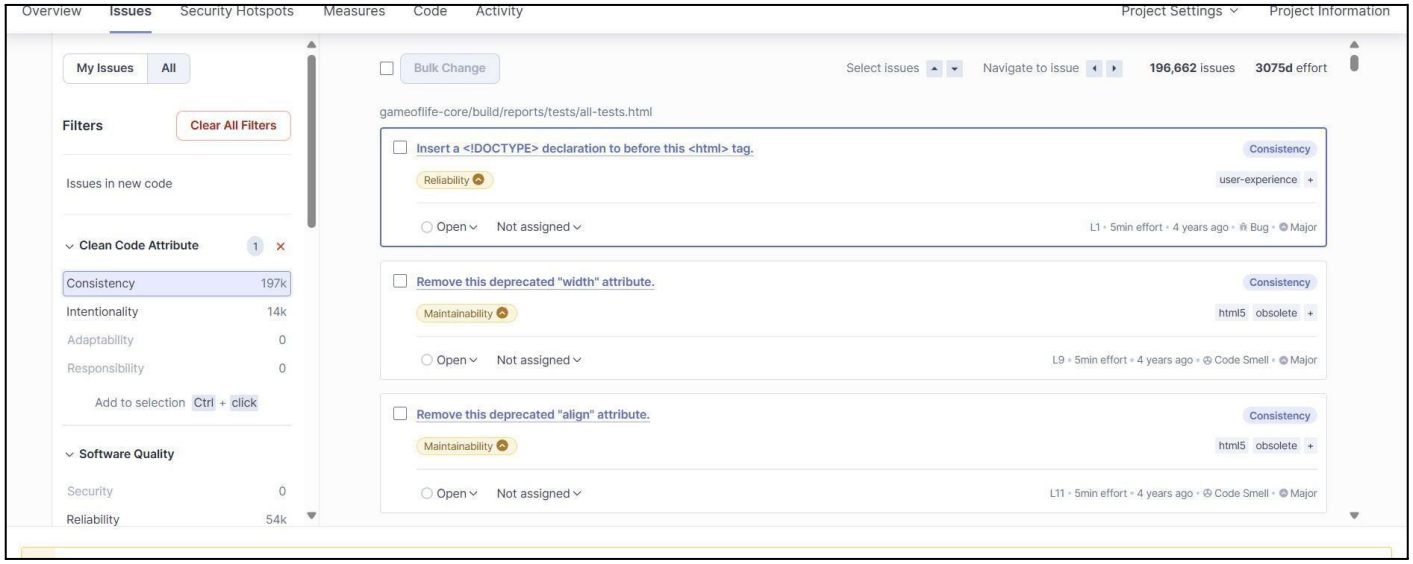
12. Check console

13. Now, check the project in SonarQube

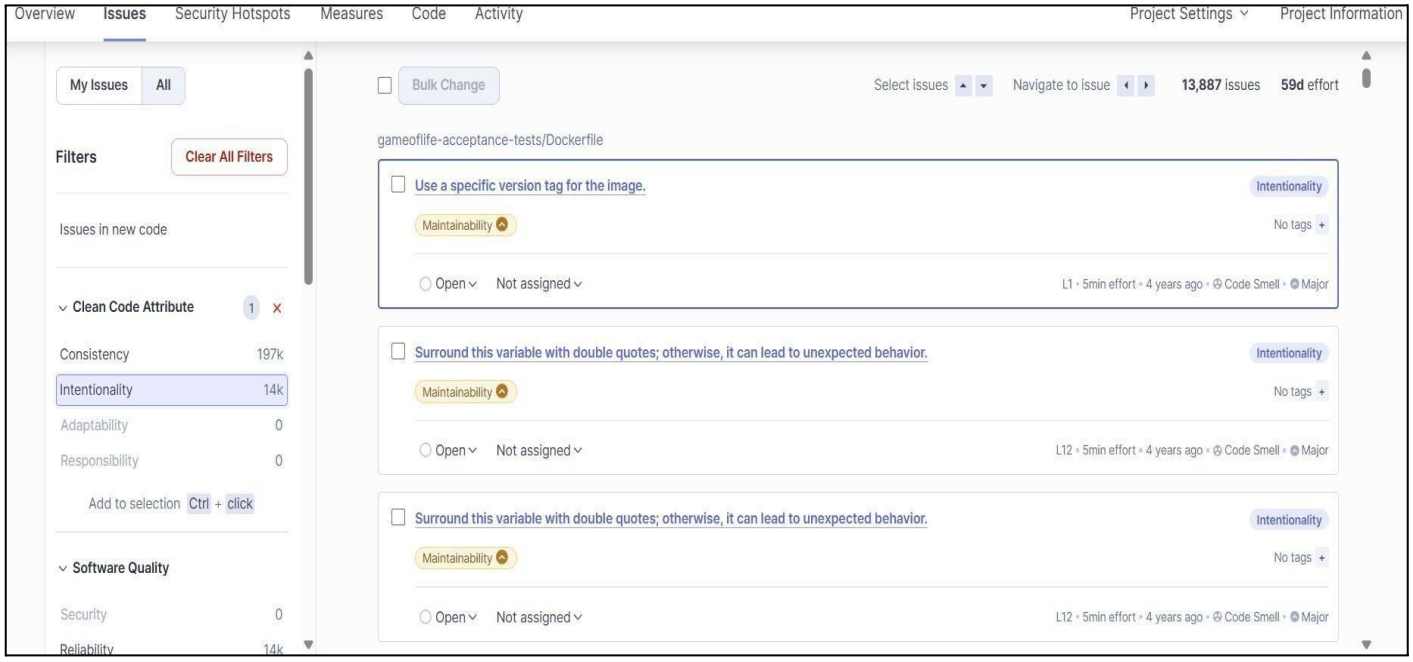


14. Code Problems

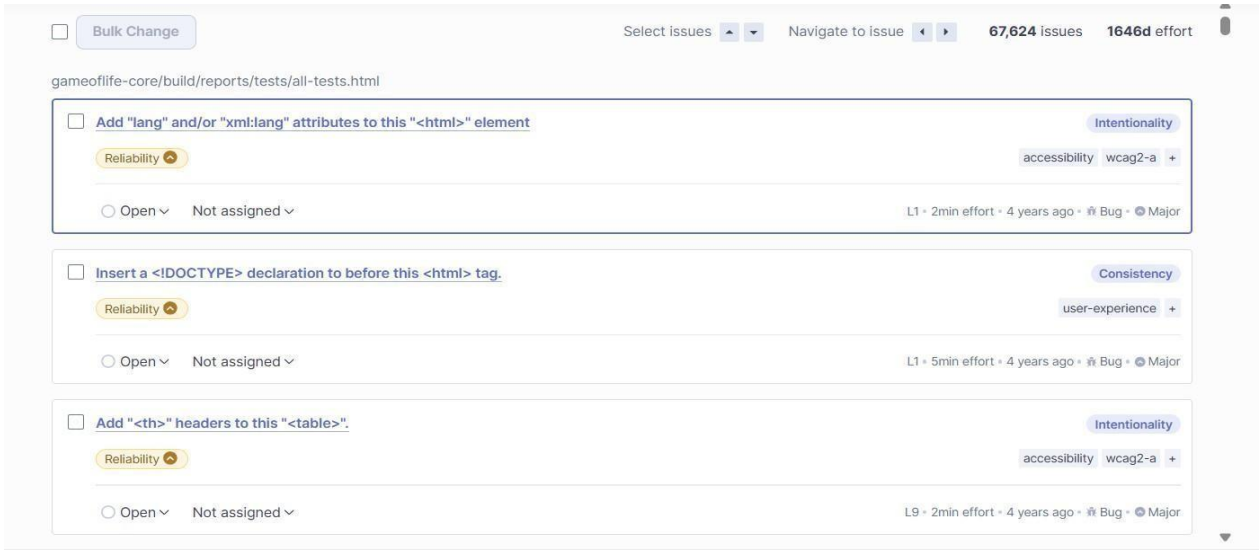
- Consistency



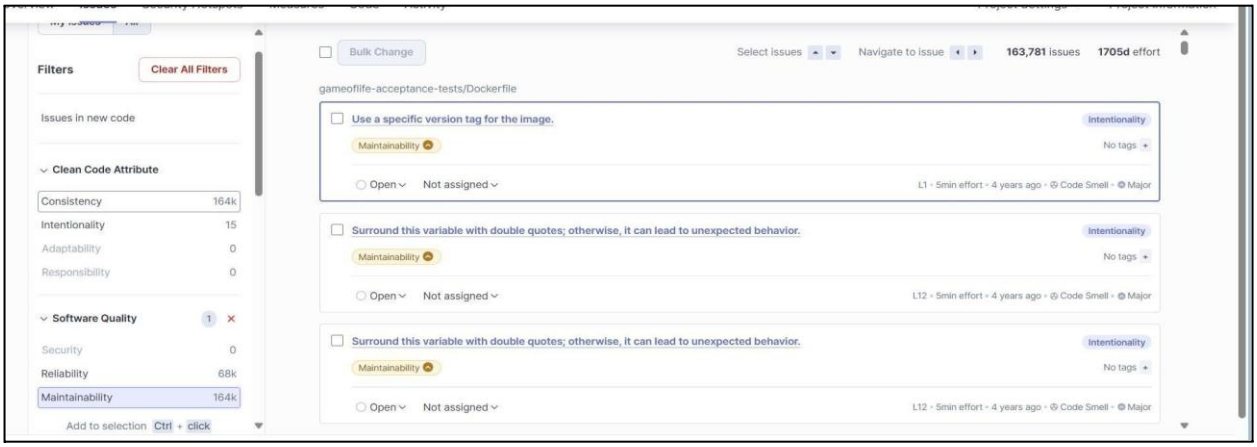
- Intentionality



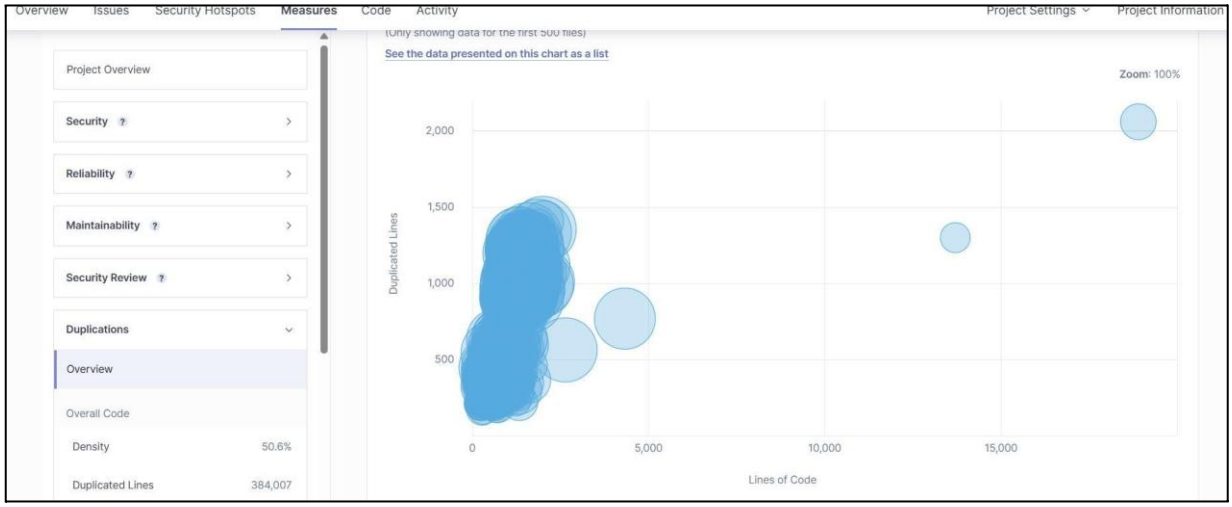
- Bugs



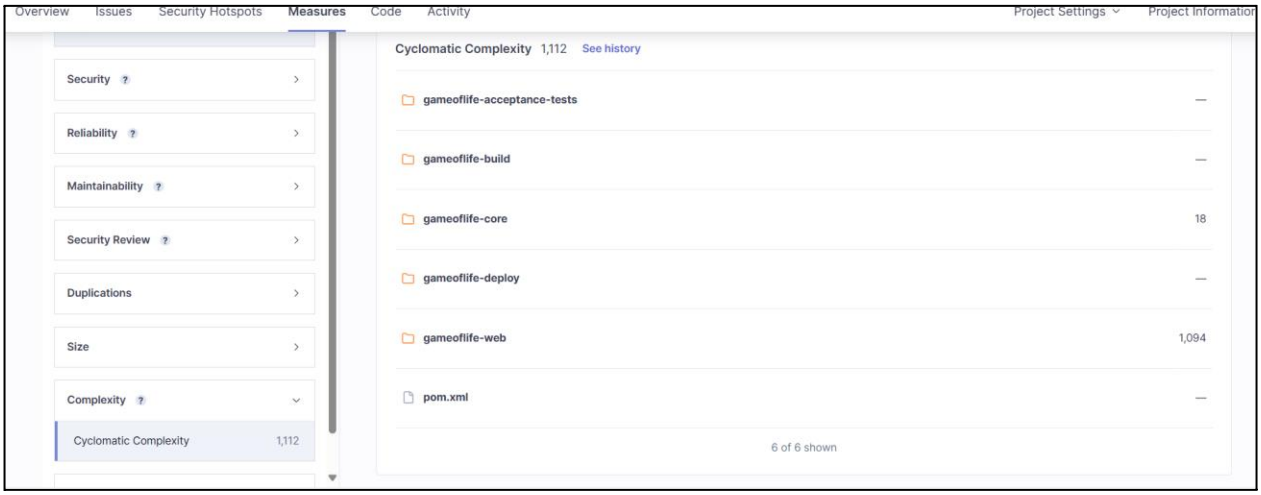
- Code Smells



- Duplications



- Cyclomatic Complexities



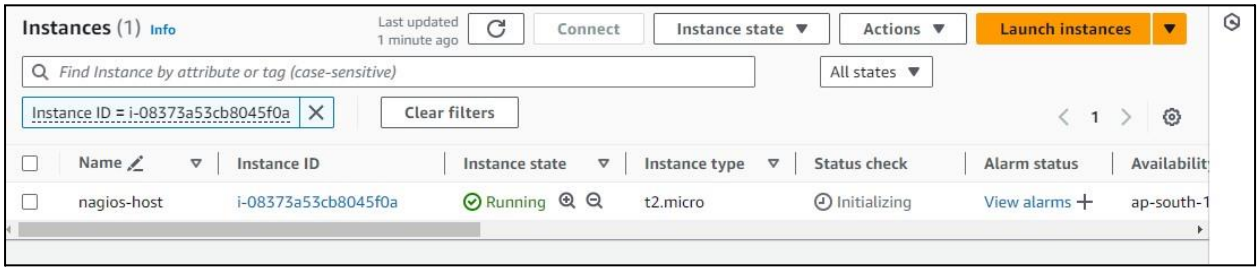
In this way, we have integrated Jenkins with SonarQube for SAST.

ADVANCE DEVOPS EXPERIMENT 9

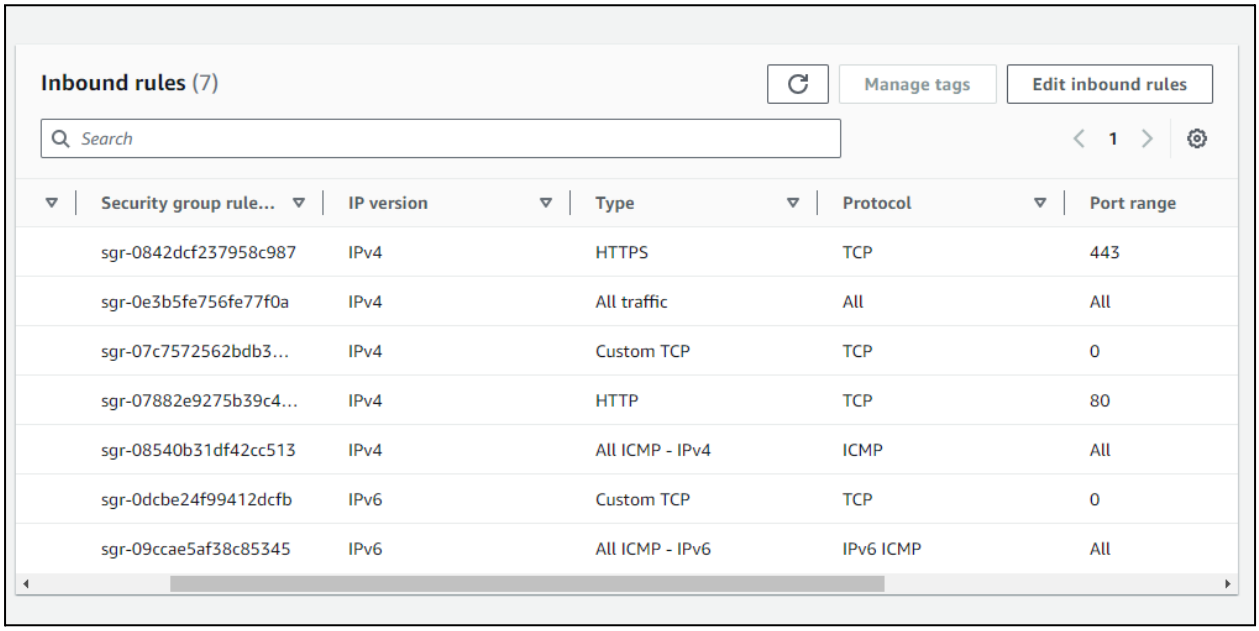
Name:Laksh .V. Sodhai
Class:D15A
Roll No:59

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 Amazon Linux EC2 instance and name it as nagios-host



Step 2: Edit the following inbound rules of the specified security groups and ensure HTTP.HTTPS,SSH,ICMP are accessible from anywhere



Step 3:Connect to your EC2 instance via the connect option available in EC2 instances menu

```
[ec2-user@ip-172-31-33-14 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:19:23 ago on Thu Sep 26 08:42:17 2024.
Dependencies resolved.
```

Package	Architecture	Version	Repository	Size
Installing:				
httpd	x86_64	2.4.62-1.amzn2023	amazonlinux	48 k
php8.3	x86_64	8.3.10-1.amzn2023.0.1	amazonlinux	10 k
Installing dependencies:				
apr	x86_64	1.7.2-2.amzn2023.0.2	amazonlinux	129 k
apr-util	x86_64	1.6.3-1.amzn2023.0.1	amazonlinux	98 k
generic-logos-httpd	noarch	18.0.0-12.amzn2023.0.3	amazonlinux	19 k
httpd-core	x86_64	2.4.62-1.amzn2023	amazonlinux	1.4 M
httpd-filesystem	noarch	2.4.62-1.amzn2023	amazonlinux	14 k
httpd-tools	x86_64	2.4.62-1.amzn2023	amazonlinux	81 k
libbrotli	x86_64	1.0.9-4.amzn2023.0.2	amazonlinux	315 k
libsodium	x86_64	1.0.19-4.amzn2023	amazonlinux	176 k
libxslt	x86_64	1.1.34-5.amzn2023.0.2	amazonlinux	241 k
mailcap	noarch	2.1.49-3.amzn2023.0.3	amazonlinux	33 k

Step 4: Update and install the required packages

Use the following commands:

sudo yum update

sudo yum install httpd php

sudo yum install gcc glibc glibc-common

sudo yum install gd gd-devel

```
[ec2-user@ip-172-31-33-14 ~]$ sudo yum install gcc glibc glibc-common
Last metadata expiration check: 0:20:32 ago on Thu Sep 26 08:42:17 2024.
Package glibc-2.34-52.amzn2023.0.11.x86_64 is already installed.
Package glibc-common-2.34-52.amzn2023.0.11.x86_64 is already installed.
Dependencies resolved.
```

Package	Architecture	Version	Repository	Size
Installing:				
gcc	x86_64	11.4.1-2.amzn2023.0.2	amazonlinux	32 M
Installing dependencies:				
annobin-docs	noarch	10.93-1.amzn2023.0.1	amazonlinux	92 k
annobin-plugin-gcc	x86_64	10.93-1.amzn2023.0.1	amazonlinux	887 k
cpp	x86_64	11.4.1-2.amzn2023.0.2	amazonlinux	10 M
gc	x86_64	8.0.4-5.amzn2023.0.2	amazonlinux	105 k
glibc-devel	x86_64	2.34-52.amzn2023.0.11	amazonlinux	27 k
glibc-headers-x86	noarch	2.34-52.amzn2023.0.11	amazonlinux	427 k
guile22	x86_64	2.2.7-2.amzn2023.0.3	amazonlinux	6.4 M
kernel-headers	x86_64	6.1.109-118.189.amzn2023	amazonlinux	1.4 M
libmpc	x86_64	1.2.1-2.amzn2023.0.2	amazonlinux	62 k
libtool-ltdl	x86_64	2.4.7-1.amzn2023.0.3	amazonlinux	38 k
libxcrypt-devel	x86_64	4.4.33-7.amzn2023	amazonlinux	32 k

```
[ec2-user@ip-172-31-33-14 ~]$ sudo yum install gd gd-devel
Last metadata expiration check: 0:21:27 ago on Thu Sep 26 08:42:17 2024.
Dependencies resolved.
```

Package	Architecture	Version	Repository	Size
Installing:				
gd	x86_64	2.3.3-5.amzn2023.0.3	amazonlinux	139 k
gd-devel	x86_64	2.3.3-5.amzn2023.0.3	amazonlinux	38 k
Installing dependencies:				
brotli	x86_64	1.0.9-4.amzn2023.0.2	amazonlinux	314 k
brotli-devel	x86_64	1.0.9-4.amzn2023.0.2	amazonlinux	31 k
bzip2-devel	x86_64	1.0.8-6.amzn2023.0.2	amazonlinux	214 k
cairo	x86_64	1.17.6-2.amzn2023.0.1	amazonlinux	684 k
cmake-filesystem	x86_64	3.22.2-1.amzn2023.0.4	amazonlinux	16 k
fontconfig	x86_64	2.13.94-2.amzn2023.0.2	amazonlinux	273 k
fontconfig-devel	x86_64	2.13.94-2.amzn2023.0.2	amazonlinux	128 k
fonts-filesystem	noarch	1:2.0.5-12.amzn2023.0.2	amazonlinux	9.5 k
freetype	x86_64	2.13.2-5.amzn2023.0.1	amazonlinux	423 k

Step 5:Create a new nagios user by writing the following commands
sudo adduser -m nagios
sudo passwd nagios

```
Complete!
[ec2-user@ip-172-31-33-14 ~]$ sudo adduser -m nagios
[ec2-user@ip-172-31-33-14 ~]$ sudo passwd nagios
Changing password for user nagios.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-33-14 ~]$
```

Step 6: Create a new user group using **sudo groupadd nagcmd** and
Add users to the group using the following commands:
sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache

```
Complete!
[ec2-user@ip-172-31-33-14 ~]$ sudo adduser -m nagios
[ec2-user@ip-172-31-33-14 ~]$ sudo passwd nagios
Changing password for user nagios.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-33-14 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-33-14 ~]$ sudo usermod -a -G nagcmd nagios
[ec2-user@ip-172-31-33-14 ~]$ sudo usermod -a -G nagcmd apache
[ec2-user@ip-172-31-33-14 ~]$ mkdir downloads
[ec2-user@ip-172-31-33-14 ~]$ cd downloads
[ec2-user@ip-172-31-33-14 downloads]$ wget https://sourceforge.net/projects/nagios/files/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz/download?use
_mirror=excellmedia
--2024-09-26 09:15:54--  https://sourceforge.net/projects/nagios/files/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz/download?use_mirror=excellmedi
a
Resolving sourceforge.net (sourceforge.net)... 172.64.150.145, 104.18.37.111, 2606:4700:4400::6812:256f, ...
Connecting to sourceforge.net (sourceforge.net)|172.64.150.145|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://downloads.sourceforge.net/project/nagios/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz?ts=gAAAAABm9SZKLFW7LwD1QAJ2jNzqmSJwAPA1mQ-
eAJYK8z5NmrV ifVkhbsV-qOfPsLUyICC6yvdHu6UeeIyvNZsVGUtir9BeQ%3D%3D&use mirror=excellmedia&r= [following]
```

Step 7: Create a directory for Nagios downloads using the following commands-
Commands -
mkdir ~/downloads
cd ~/downloads

Also download Nagios and plugin source files
Commands -
wget
https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.6.tar.gz
wget https://nagios-plugins.org/download/nagios-plugins-2.3.3.tar.gz

```
Connecting to prdownloads.sourceforge.net (prdownloads.sourceforge.net)|204.68.111.105|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://downloads.sourceforge.net/project/nagios/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz [following]
--2024-09-26 09:38:43--  https://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
Connecting to downloads.sourceforge.net (downloads.sourceforge.net)|204.68.111.105|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://excellmedia.dl.sourceforge.net/project/nagios/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz?viasf=1 [following]
--2024-09-26 09:38:45--  https://excellmedia.dl.sourceforge.net/project/nagios/nagios-4.x/nagios-4.0.8/nagios-4.0.8.tar.gz?viasf=1
Resolving excellmedia.dl.sourceforge.net (excellmedia.dl.sourceforge.net)... 202.153.32.19, 2401:fb00:0:1fe:8000::5
Connecting to excellmedia.dl.sourceforge.net (excellmedia.dl.sourceforge.net)|202.153.32.19|:443... 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  8.14MB/s    in 0.2s

2024-09-26 09:38:45 (8.14 MB/s) - 'nagios-4.0.8.tar.gz' saved [1805059/1805059]

[ec2-user@ip-172-31-33-14 downloads]$ ls
'download?use_mirror=excellmedia'   nagios-4.0.8.tar.gz
[ec2-user@ip-172-31-33-14 downloads]$ tar -xzf nagios-4.0.8.tar.gz
[ec2-user@ip-172-31-33-14 downloads]$
```

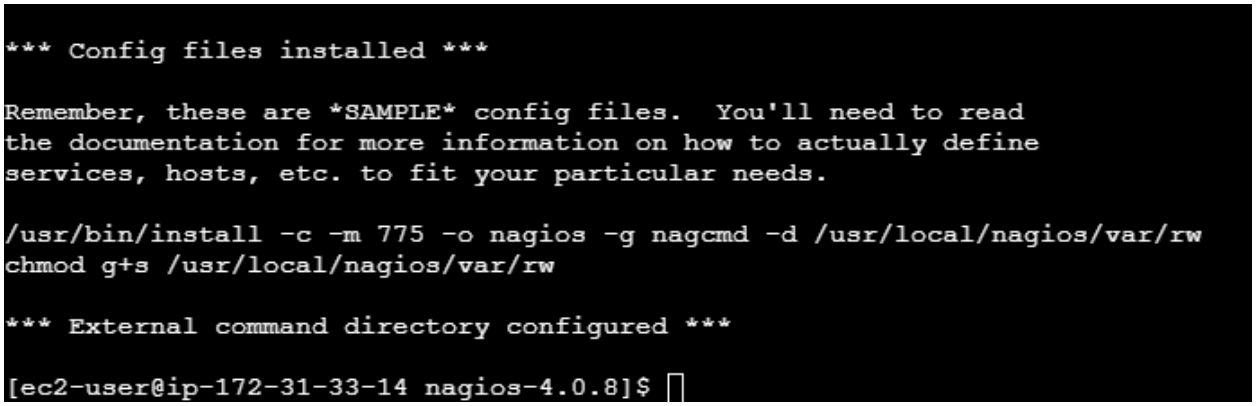
Step 8-Extract the nagios source file with the following commands
tar zxvf nagios-4.4.6.tar.gz
cd nagios-4.4.6

Then run the configuration script with the following command
/configure --with-command-group=nagcmd

```
        Nagios user/group:  nagios,nagios
       Command user/group:  nagios,nagcmd
             Event Broker:  yes
        Install ${prefix}:  /usr/local/nagios
    Install ${includedir}:  /usr/local/nagios/include/nagios
                Lock file:  ${prefix}/var/nagios.lock
   Check result directory:  ${prefix}/var/spool/checkresults
           Init directory:  /etc/rc.d/init.d
  Apache conf.d directory:  /etc/httpd/conf.d
             Mail program:  /bin/mail
                  Host OS:  linux-gnu
          IOBroker Method:  epoll

 Web Interface Options:
 ------------------------
                 HTML URL:  http://localhost/nagios/
                  CGI URL:  http://localhost/nagios/cgi-bin/
 Traceroute (used by WAP):  /usr/bin/traceroute


Review the options above for accuracy.  If they look okay,
type 'make all' to compile the main program and CGIs.

[ec2-user@ip-172-31-33-14 nagios-4.0.8]$
```

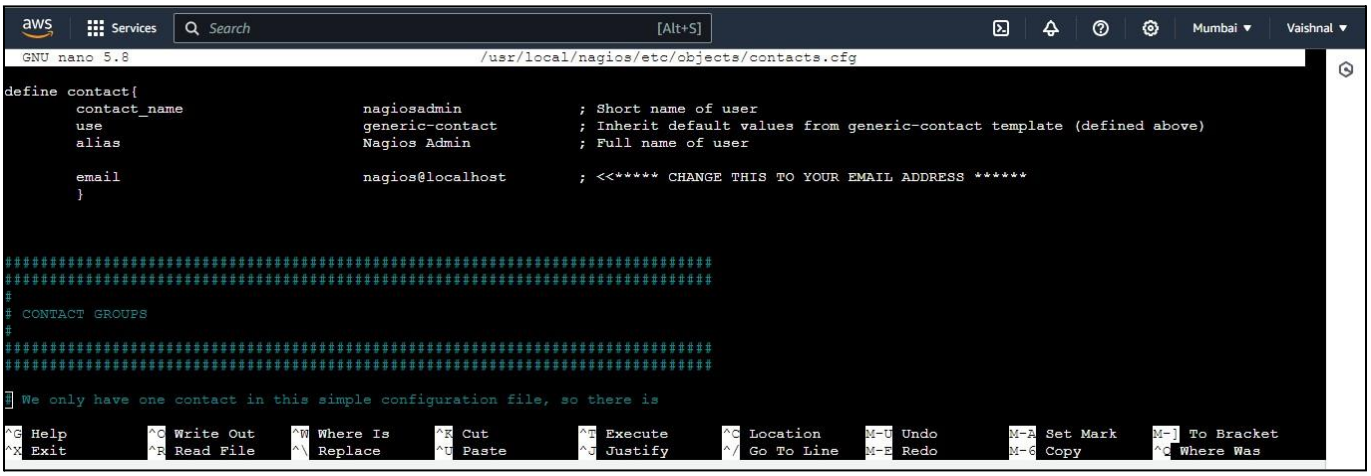
Step 9-Compile the source code with the following commands
make all

Step 10-Install binaries,init script and sample config files
Commands -
./sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode



Step 11-Edit the Config File to Change the Email Address
Commands -
sudo nano /usr/local/nagios/etc/objects/contacts.cfg
● Change the email address in the contacts.cfg file to your preferred email

Step 12-Configure the Web Interface
Commands -
sudo make install-webconf

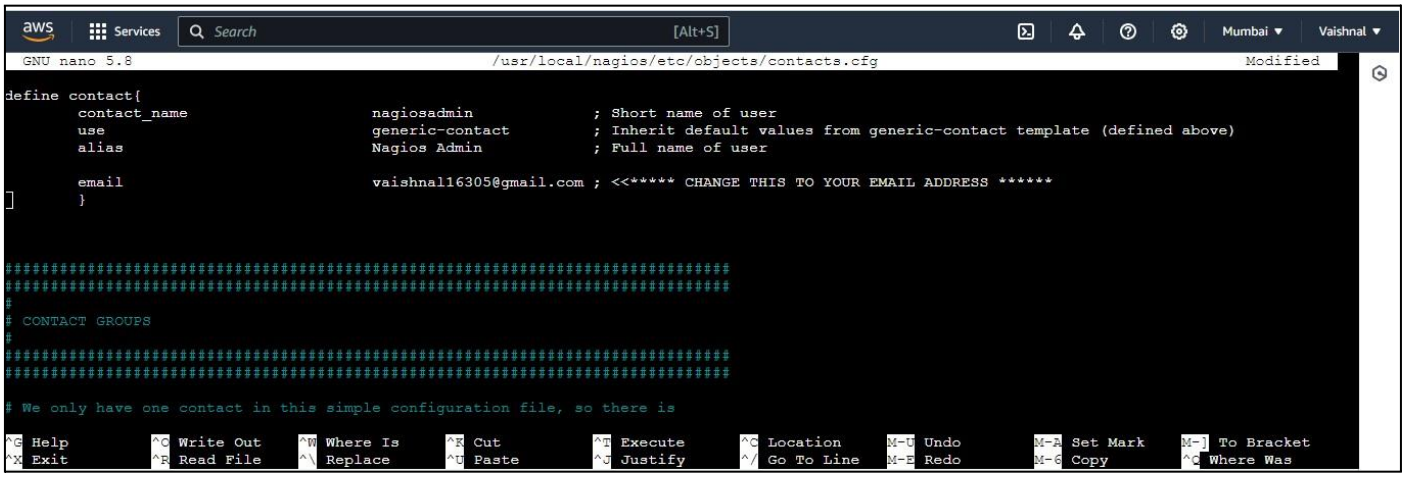


Step 13-Create a Nagios Admin Account

Commands -

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

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



Step 14-. Extract the Plugins Source File

Commands -

cd ~/downloads

tar zxvf nagios-plugins-2.3.3.tar.gz

cd nagios-plugins-2.3.3

```
*** External command directory configured ***

[ec2-user@ip-172-31-33-14 nagios-4.0.8]$ sudo nano /usr/local/nagios/etc/objects/contacts.cfg
[ec2-user@ip-172-31-33-14 nagios-4.0.8]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios.conf

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

Step 15-19. Compile and Install Plugins

Commands -

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

make

sudo make install

```
[ec2-user@ip-172-31-33-14 nagios-4.0.8]$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
```

Step 16-Start Nagios

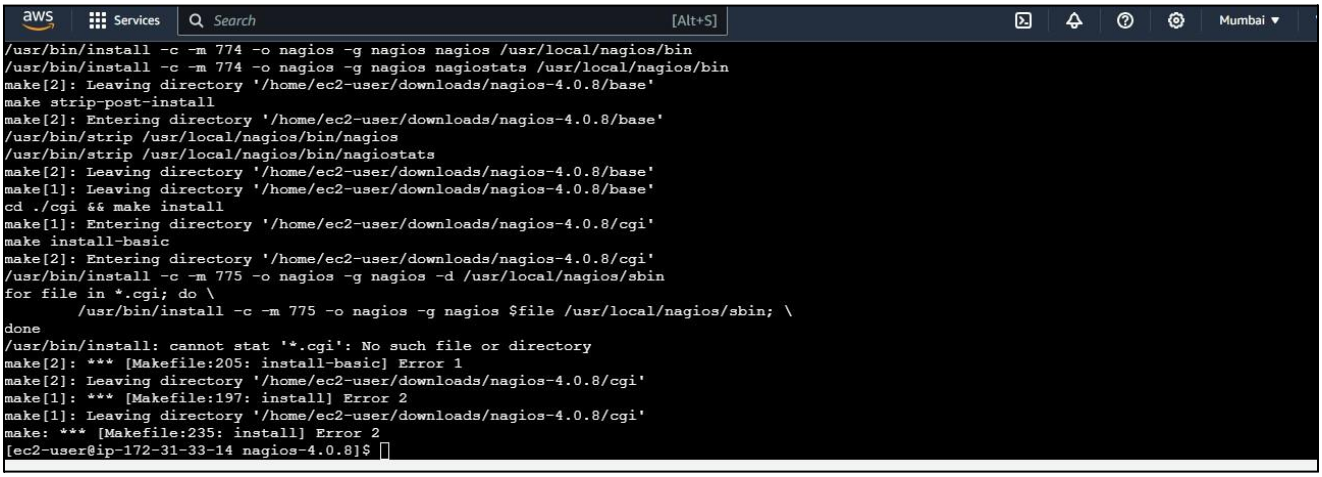
Commands -

sudo chkconfig --add nagios

sudo chkconfig nagios on

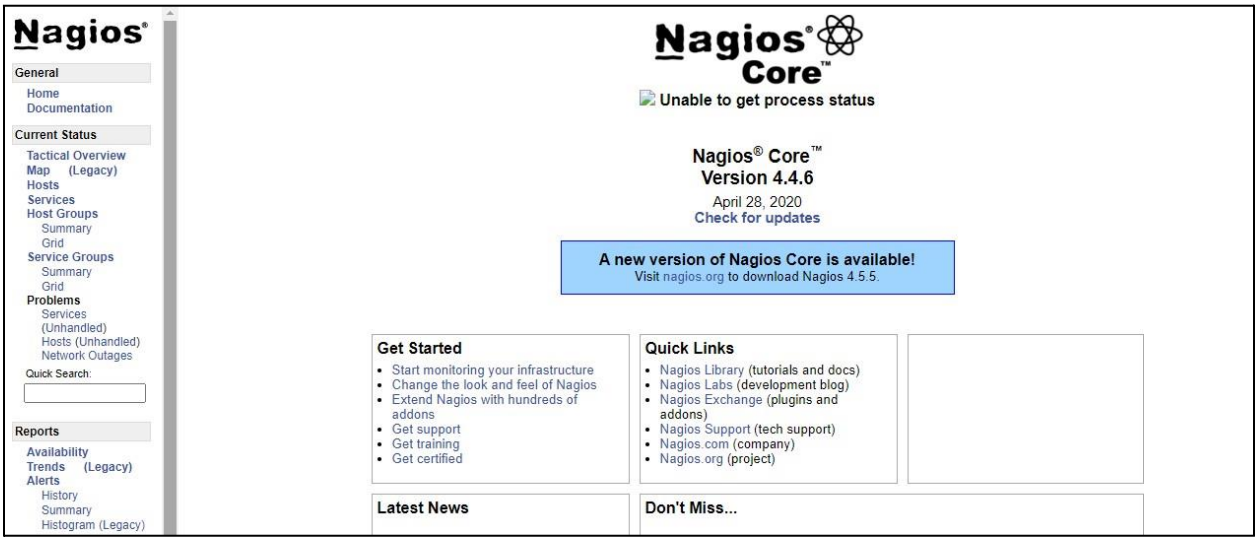
sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

sudo systemctl start nagios



Step 17-Access Nagios Web Interface

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

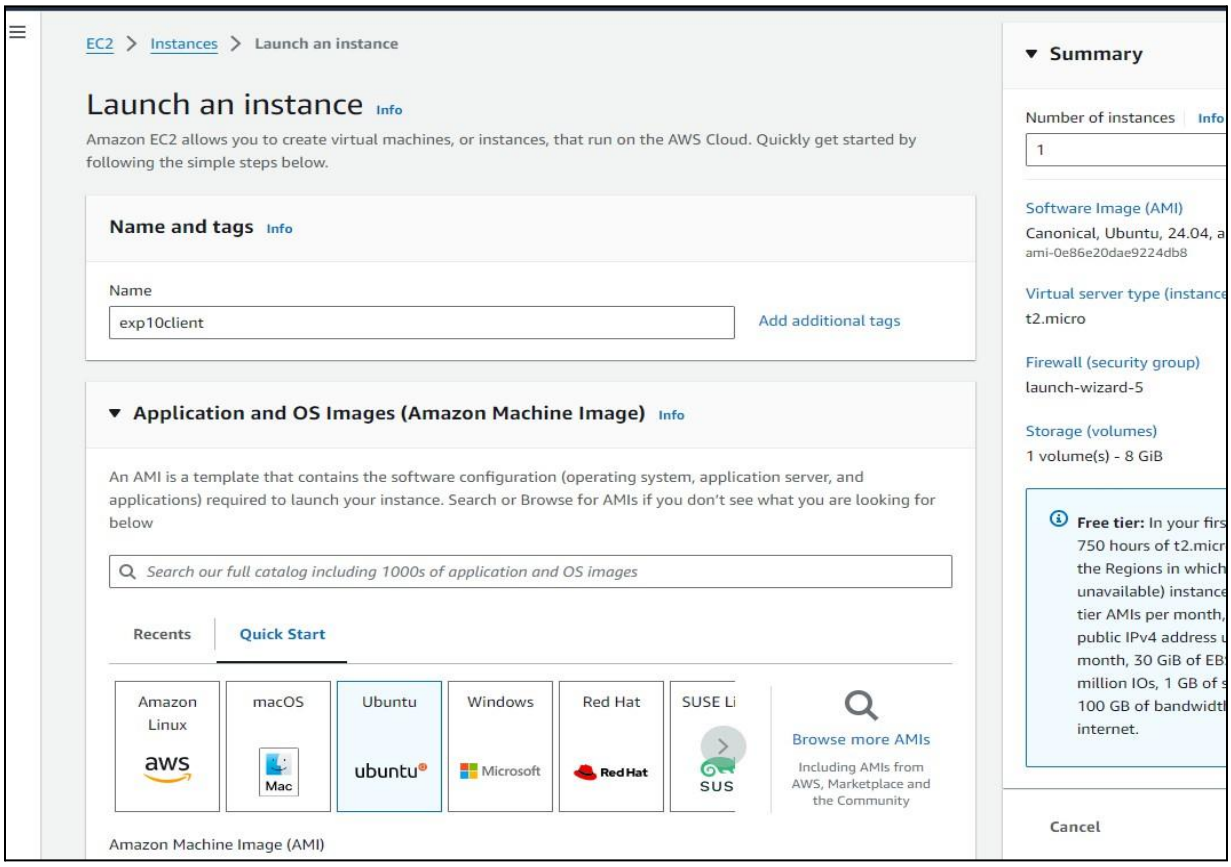

Nagios
General
Home
Documentation
Current Status
Tactical Overview
Map (Legacy)
Hosts
Services
Host Groups
Summary
Grid
Service Groups
Summary
Grid
Problems
Services (Unhandled)
Hosts (Unhandled)
Network Outages
Quick Search:
Reports
Availability
Trends (Legacy)
Alerts
History
Summary
Histogram (Legacy)
Nagios® Core™
Unable to get process status
Nagios® Core™
Version 4.4.6
April 28, 2020
Check for updates
A new version of Nagios Core is available!
Visit nagios.org to download Nagios 4.5.5.
Get Started
Start monitoring your infrastructure
Change the look and feel of Nagios
Extend Nagios with hundreds of addons
Get support
Get training
Get certified
Quick Links
Nagios Library (tutorials and docs)
Nagios Labs (development blog)
Nagios Exchange (plugins and addons)
Nagios Support (tech support)
Nagios.com (company)
Nagios.org (project)
Latest News
Don't Miss...

ADVANCE DEVOPS EXPERIMENT 10

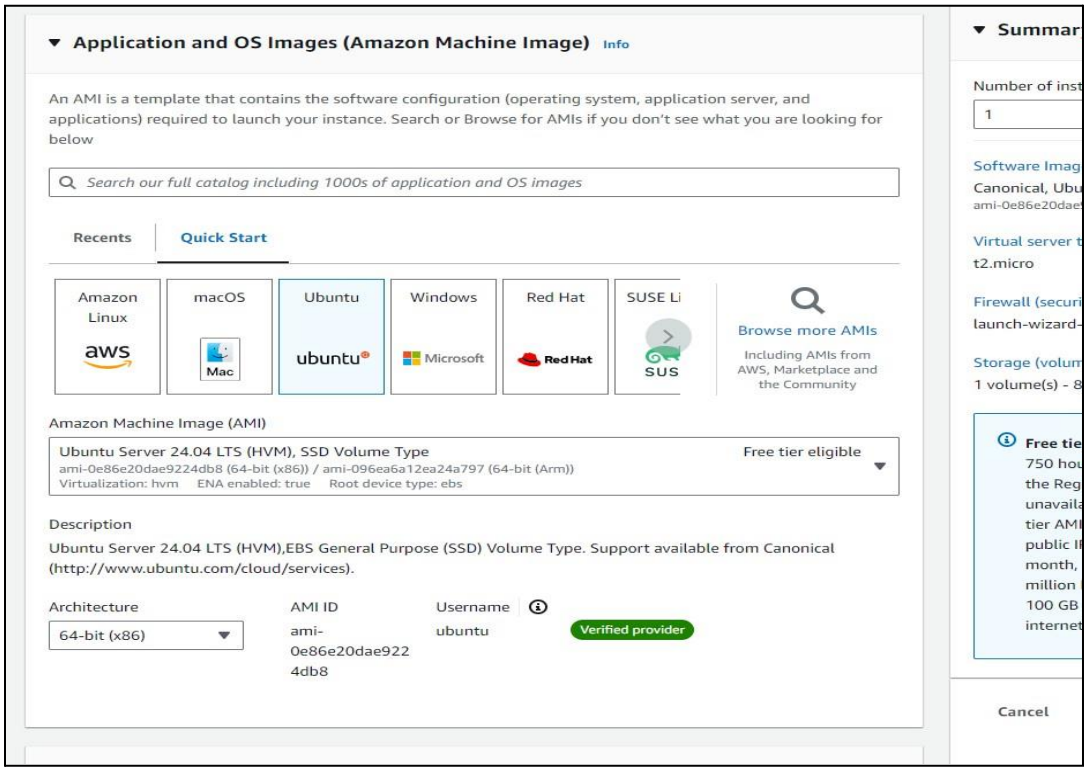
Name:Laksh .V. Sodhai
Class;D15A
Roll No:59

1) Launch an instance

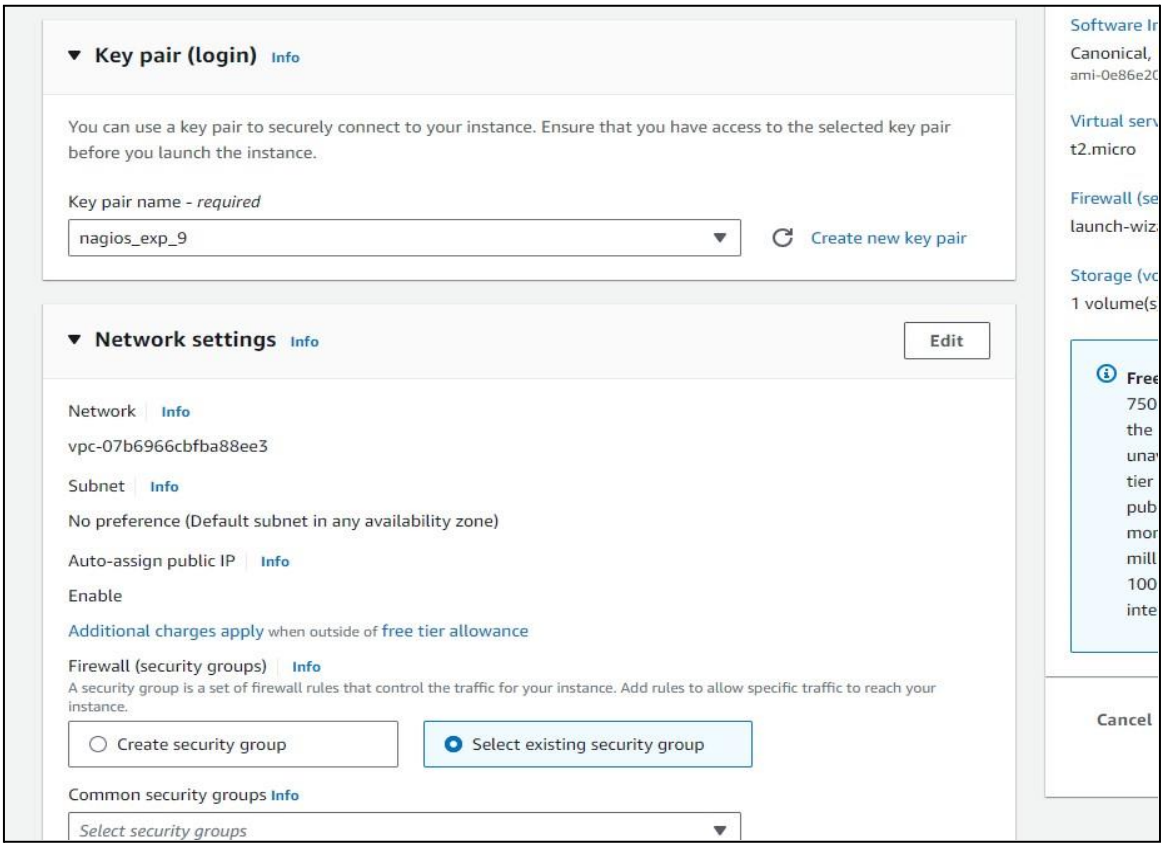
Launch an ec2 instance.
Select Ubuntu as the os give a meaningful name of the instance.



Select the same security group as given in exp9.

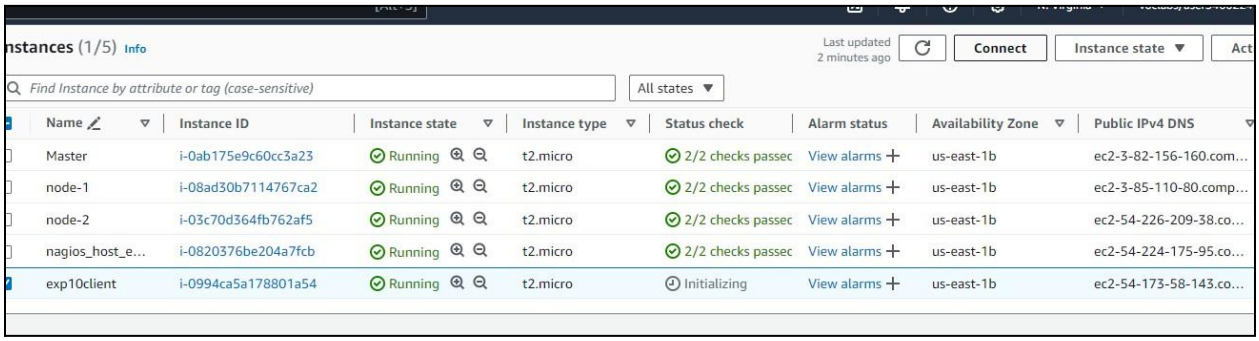


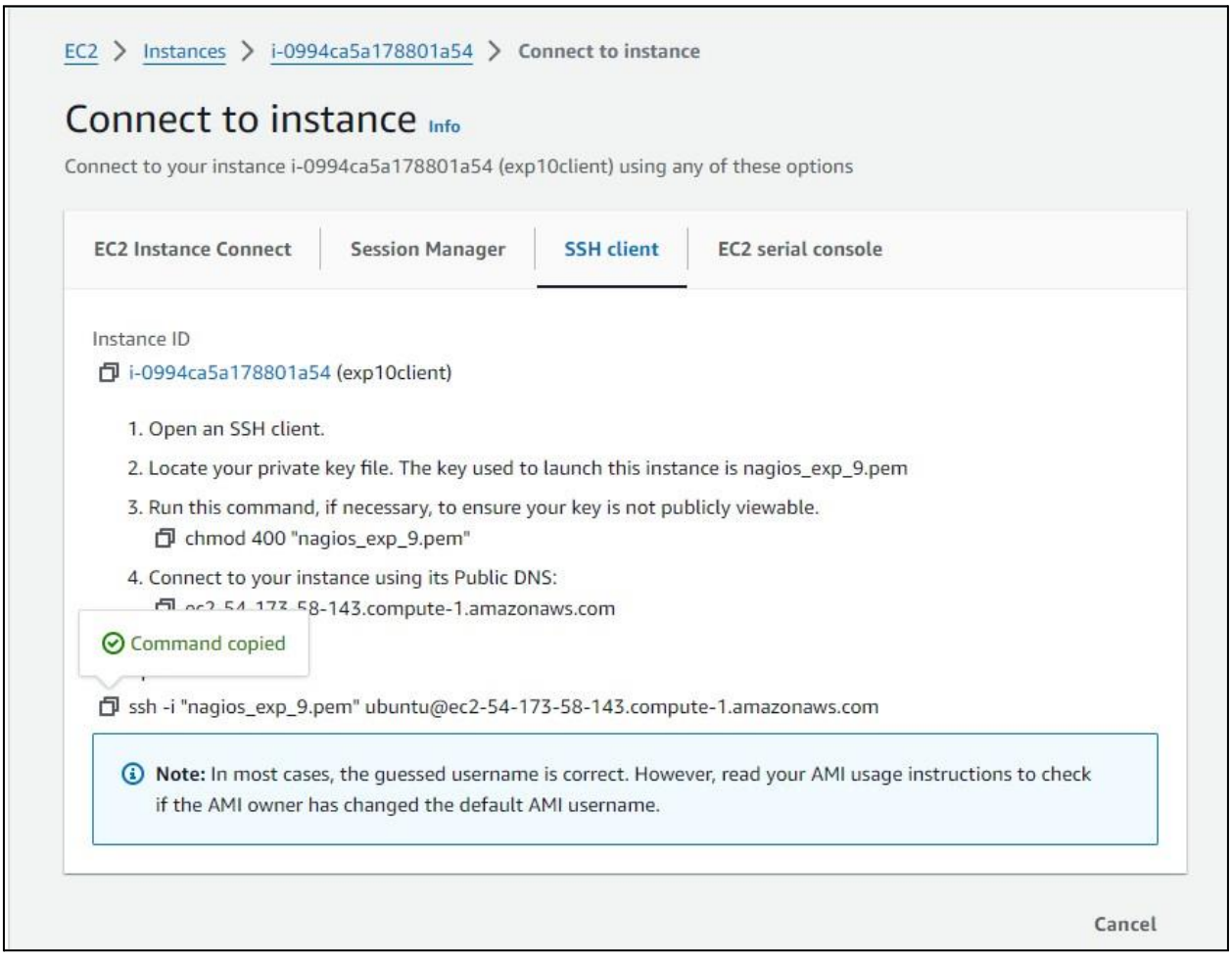
Make sure to select the same key-pair login used in the exp9 machine.



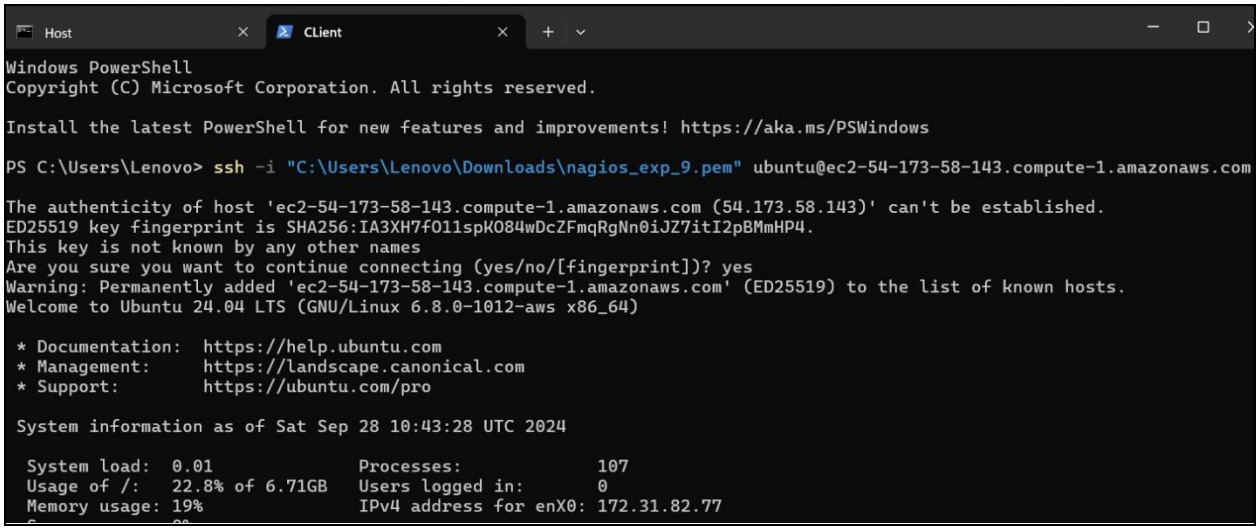
click on launch instance.

Now connect with this client machine using the ssh through your terminal(open a new terminal in your local machine and we will need both of the terminals open)





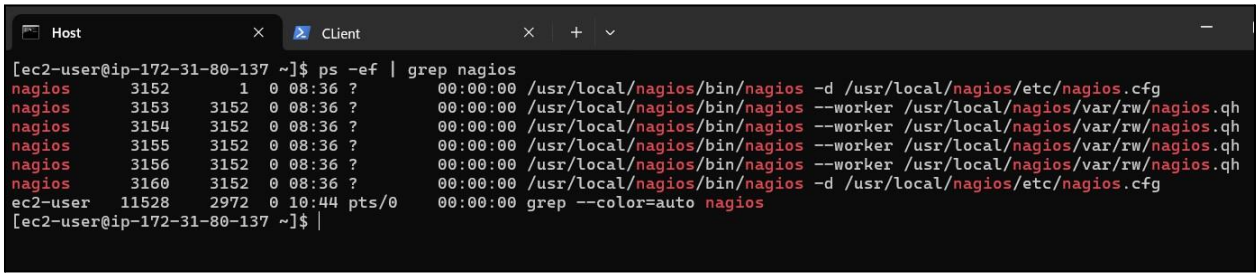
Note to change the path of the .pem file.



2) Go to nagios host machine (Host machine)

Perform the following commands

ps -ef | grep nagios



sudo su

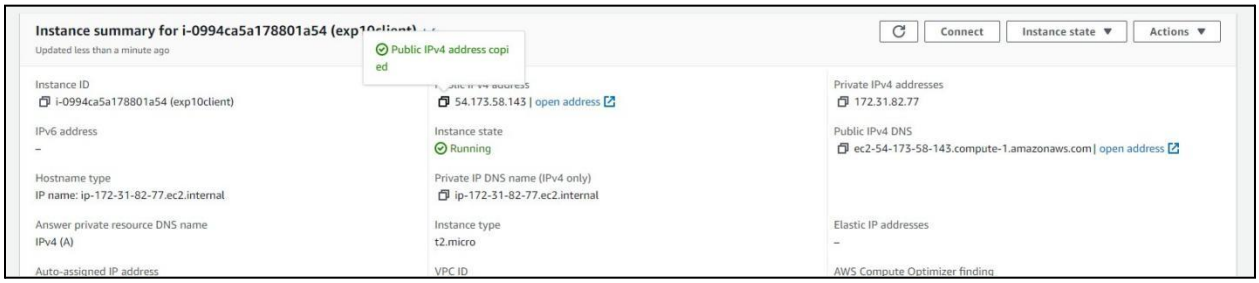
mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts

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

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

Change hostname and alias to linuxserver

Change address to public ip address of client instance (Ubuntu instance) you can get the ip address by clicking on the instance id on the instances section there you will get the public ipv4 address



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

# Define a host for the local machine

define host {

   use                     linux-server            ; Name of host template to use
                                                   ; This host definition will in>
                                                   ; in (or inherited by) the lin>
   host_name               linuxserver
   alias                   linuxserver
   address                 54.173.58.143
```

Change hostgroup_name to linux-servers1

```
# Define an optional hostgroup for Linux machines

define hostgroup {

    hostgroup_name          linux-servers1          ; The name of the hostgroup
    alias                   Linux Servers           ; Long name of the group
    members                 localhost               ; Comma separated list of host>
}
```

Change the occurrences of hostname further in the document from localhost to linuxserver
example like:

```
   host_name                localhost
```

changed to

```
define service {

    use                     local-service           ; Name of service templat
    host_name               linuxserver
    service_description     PING
    check_command           check_ping!100.0,20%!500.0,60%
}
```

This is the last one

```
define service {

    use                     local-service           ; Name of service template to >
    host_name               linuxserver
    service_description     HTTP
    check_command           check_http
    notifications_enabled   0
```

now ctrl+O and enter to save and then ctrl+X for exiting.
Open nagios configuration file and add the line shown below
nano /usr/local/nagios/etc/nagios.cfg

```
[root@ip-172-31-80-137 ec2-user]# nano /usr/local/nagios/etc/nagios.cfg
```

##Add this line below the opened nano interface where similar lines are commented.
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/

```
 GNU nano 5.8                                                    /usr/local/nagios/etc/nagios.cfg
# These are the object configuration files in which you define hosts,
# host groups, contacts, contact groups, services, etc.
# You can split your object definitions across several config files
# if you wish (as shown below), or keep them all in a single config file.

# You can specify individual object config files as shown below:
cfg_file=/usr/local/nagios/etc/objects/commands.cfg
cfg_file=/usr/local/nagios/etc/objects/contacts.cfg
cfg_file=/usr/local/nagios/etc/objects/timeperiods.cfg
cfg_file=/usr/local/nagios/etc/objects/templates.cfg

# Definitions for monitoring the local (Linux) host
cfg_file=/usr/local/nagios/etc/objects/localhost.cfg

# Definitions for monitoring a Windows machine
#cfg_file=/usr/local/nagios/etc/objects/windows.cfg

# Definitions for monitoring a router/switch
#cfg_file=/usr/local/nagios/etc/objects/switch.cfg

# Definitions for monitoring a network printer
#cfg_file=/usr/local/nagios/etc/objects/printer.cfg


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

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



# OBJECT CACHE FILE
# This option determines where object definitions are cached when
# Nagios starts/restarts.  The CGIs read object definitions from
```

ctrl+o and enter for saving and ctrl+x to exit nano editor.

Verify configuration files

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

```
[root@ip-172-31-80-137 ec2-user]# /usr/local/nagios/bin/nagios -v /usr/local/nagios
/etc/nagios.cfg

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

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

Running pre-flight check on configuration data...

Checking objects...
        Checked 0 service dependencies
        Checked 0 host dependencies
        Checked 5 timeperiods
 Checking global event handlers...
 Checking obsessive compulsive processor commands...
 Checking misc settings...

 Total Warnings: 0
 Total Errors:   0

 Things look okay - No serious problems were detected during the pre-flight check
 [root@ip-172-31-80-137 ec2-user]#
```

Restart nagios service.

service nagios restart

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

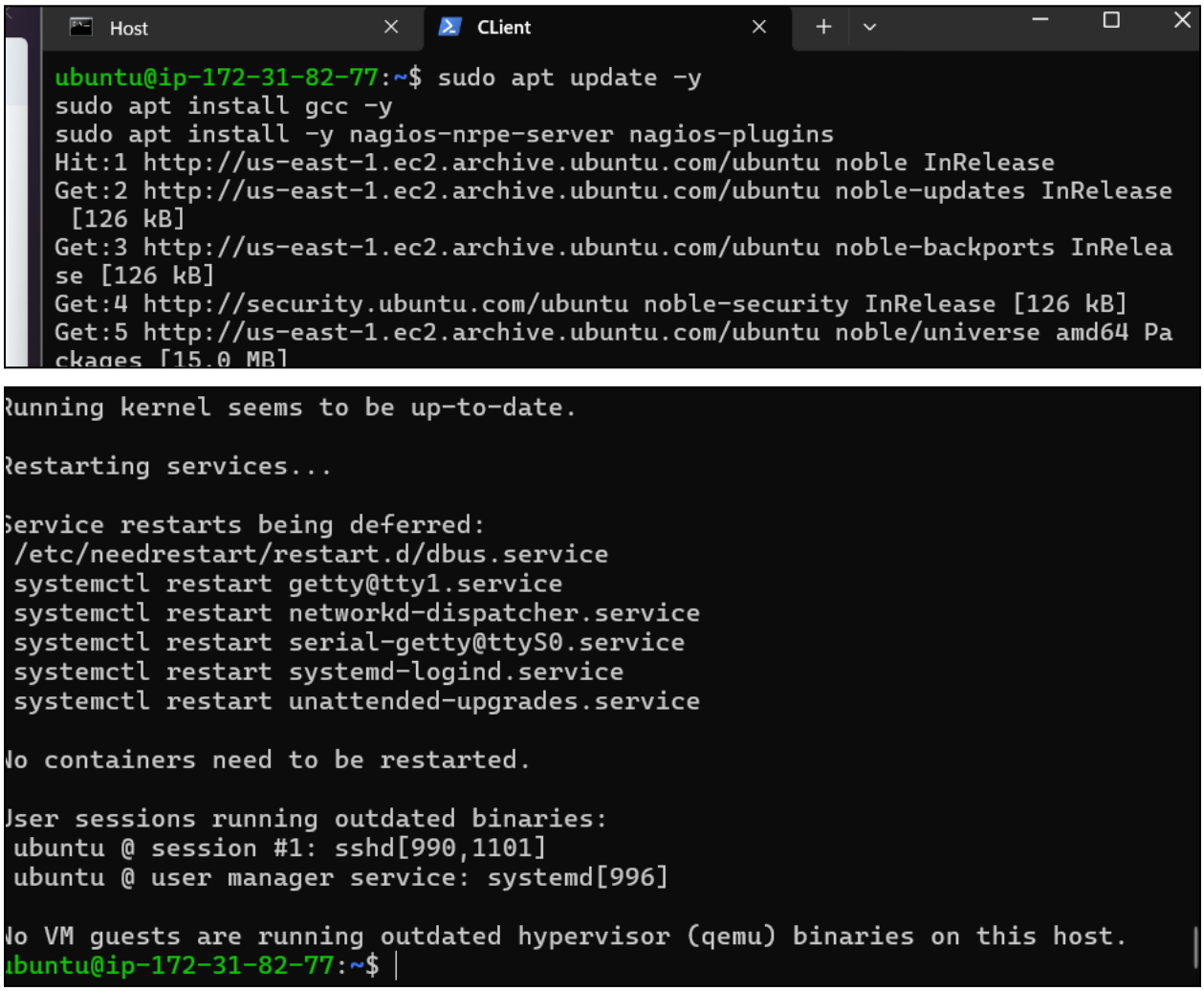
3) Go to client machine (ubuntu machine)

Perform the following commands

sudo apt update -y

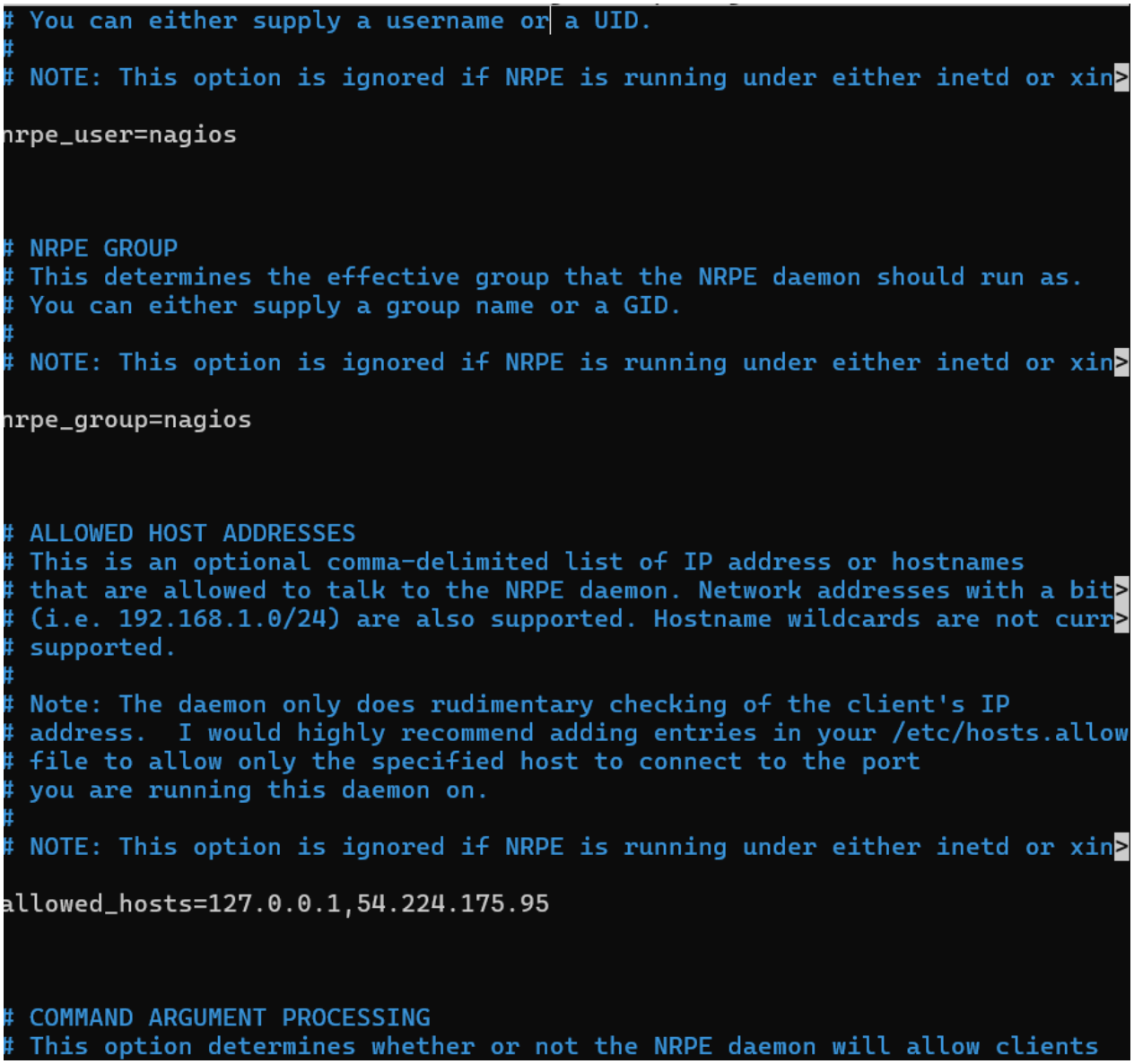
sudo apt install gcc -y

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



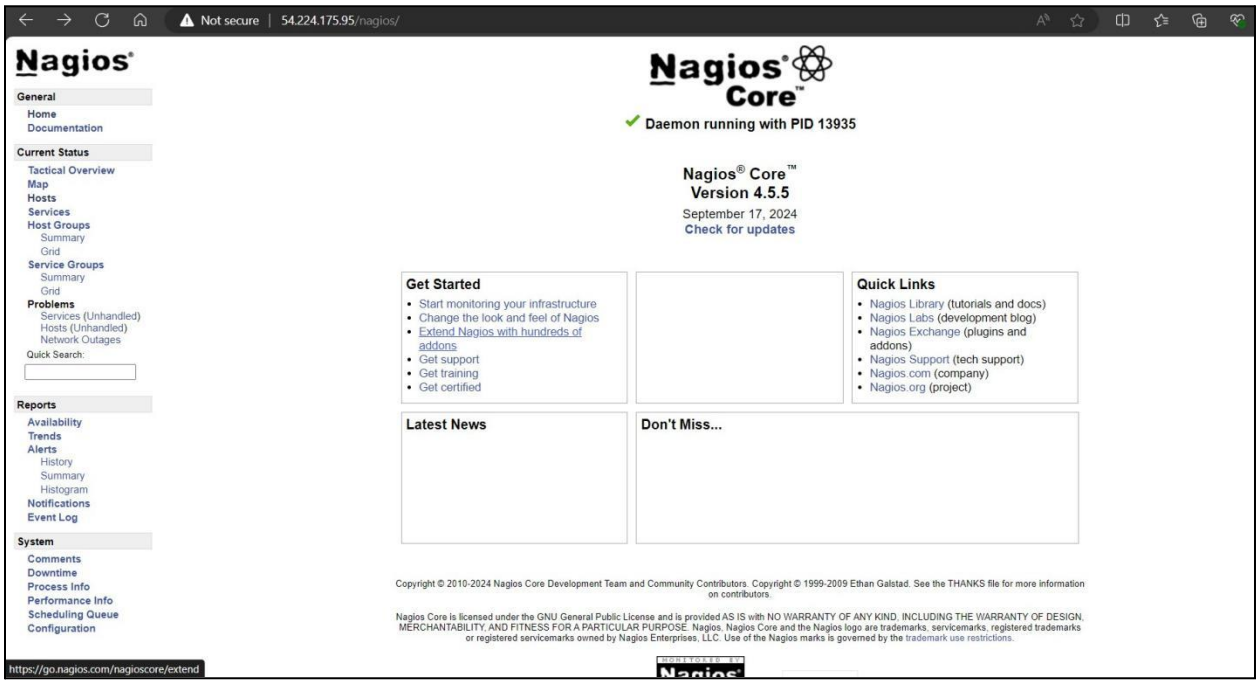
Open the nrpe.cfg file in nano editor
sudo nano /etc/nagios/nrpe.cfg

Under allowed_hosts, add the nagios host ip address (public)

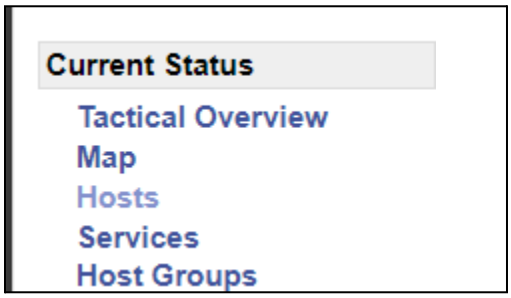


again save and exit the nano editor.

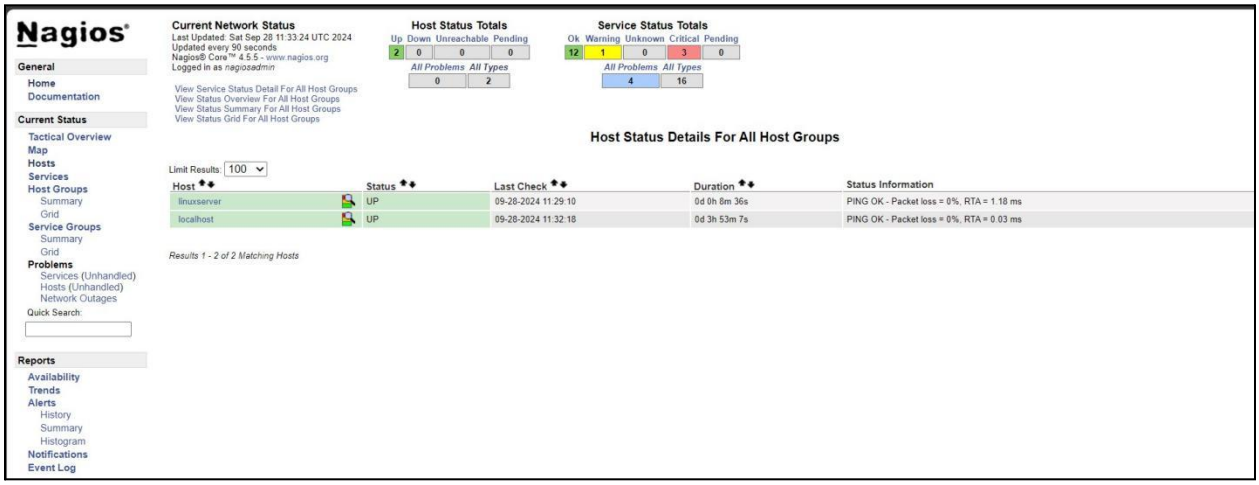
4) Go to nagios dashboard and click on hosts

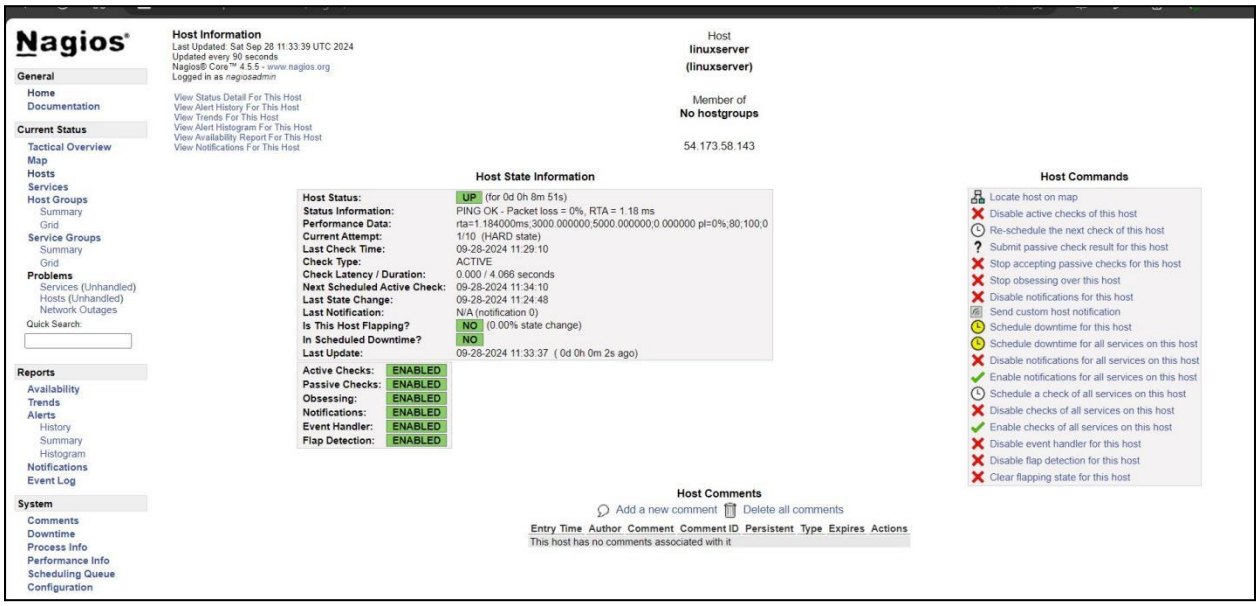


Click on hosts

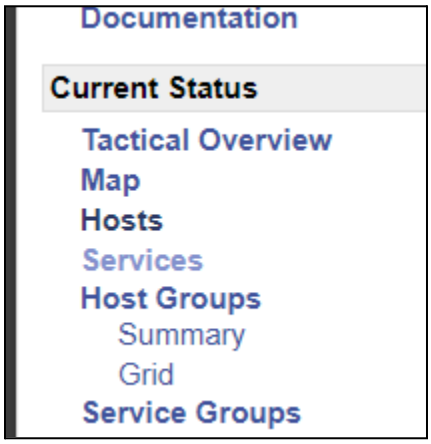


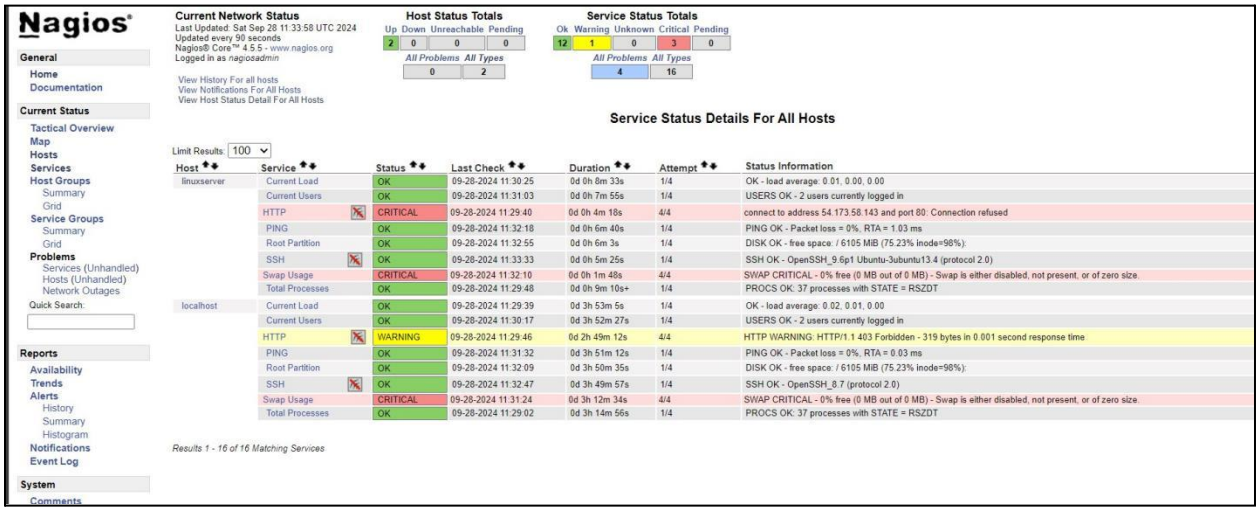
5) Click on linux server





6) Click on nagios services





Conclusion:

In this lab, we successfully configured a monitoring setup between a Nagios host machine (referred to as "exp9 machine") and a client machine (created specifically for this experiment). The goal was to set up Nagios to monitor a remote Linux server, which involved configuring both the Nagios host and client machine (Ubuntu instance) in an EC2 environment.

ADVANCE DEVOPS EXP 11

Name:Laksh .V. Sodhai

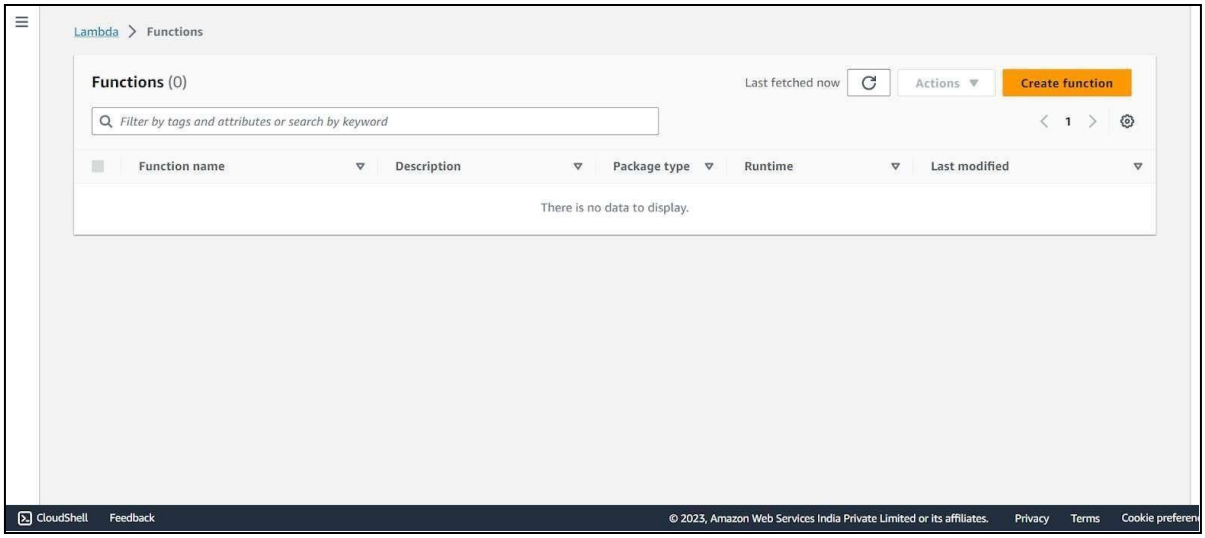
Class: D15A

Roll No:59

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

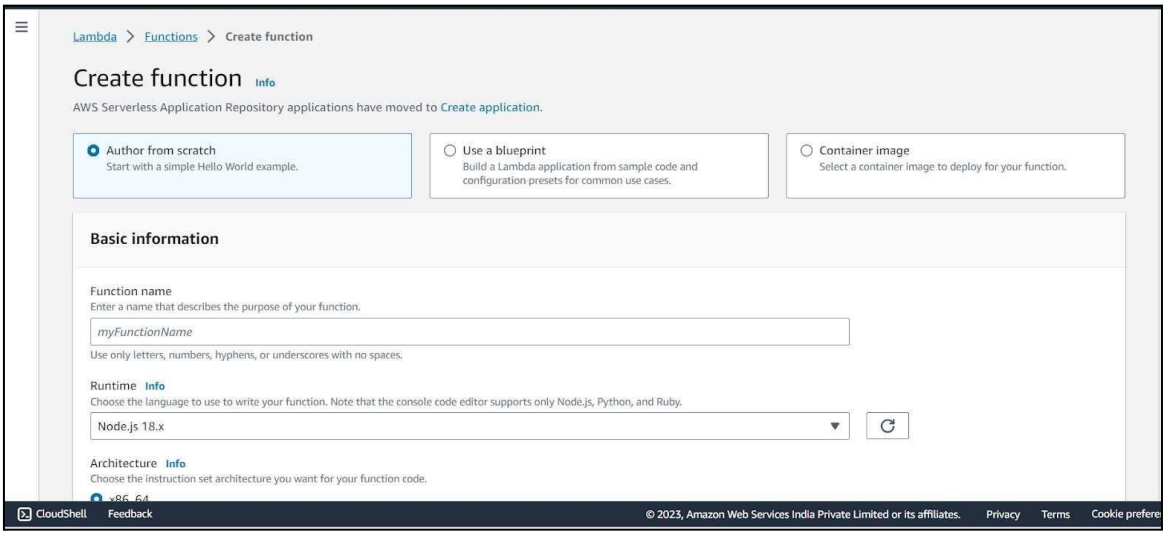
Steps to create an AWS Lambda function

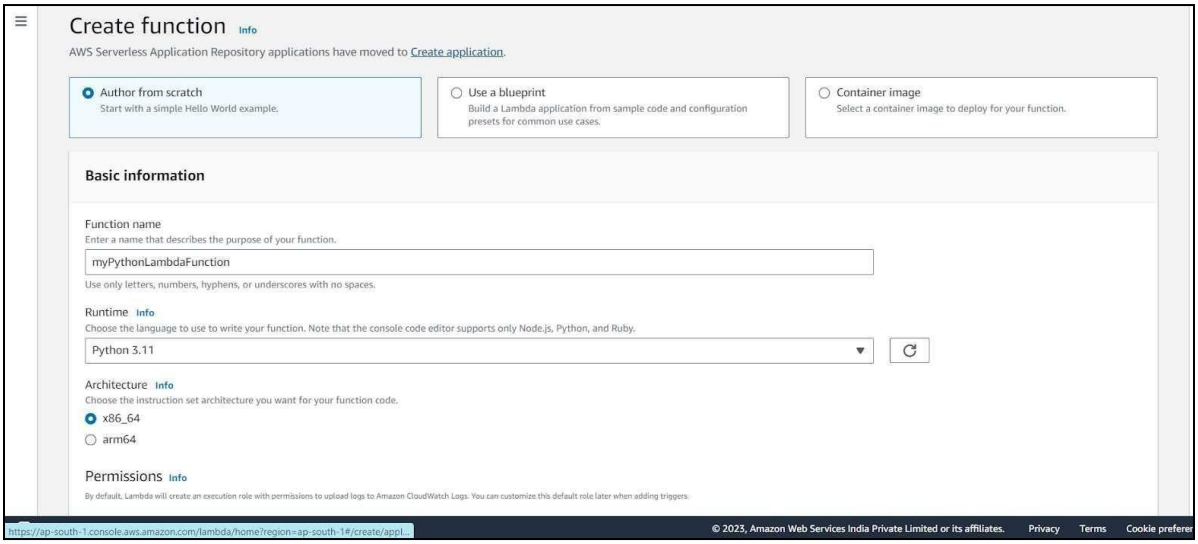
Step 1:Open up the Lambda Console and click on the Create button.
Be mindful of where you create your functions since Lambda is region-dependent.

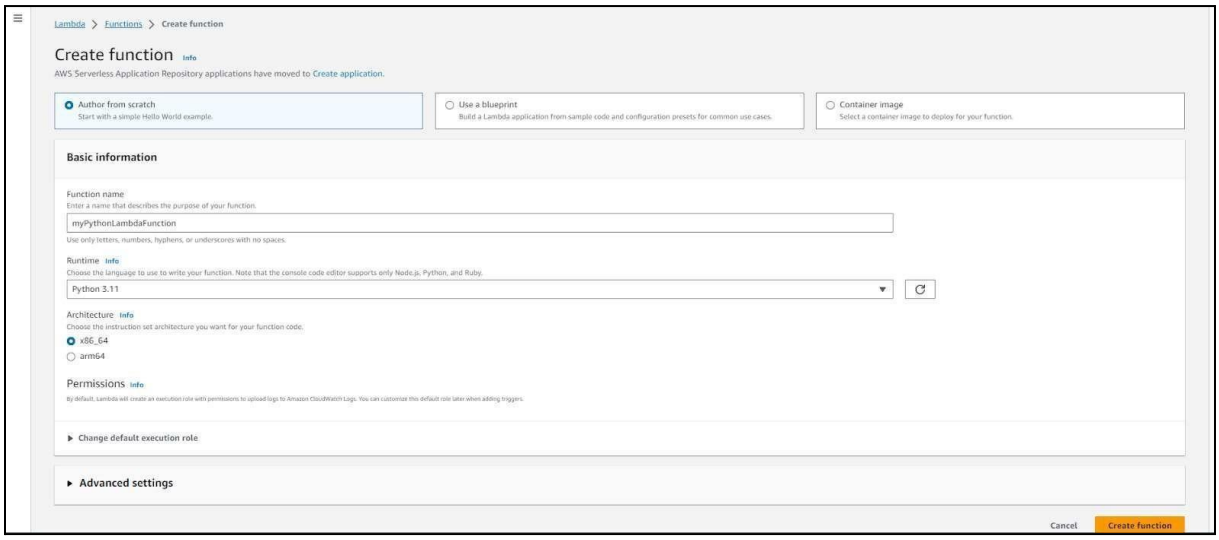


2. Choose to create a function from scratch or use a blueprint, i.e templates defined by AWS for you with all configuration presets required for the most common use cases.

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.

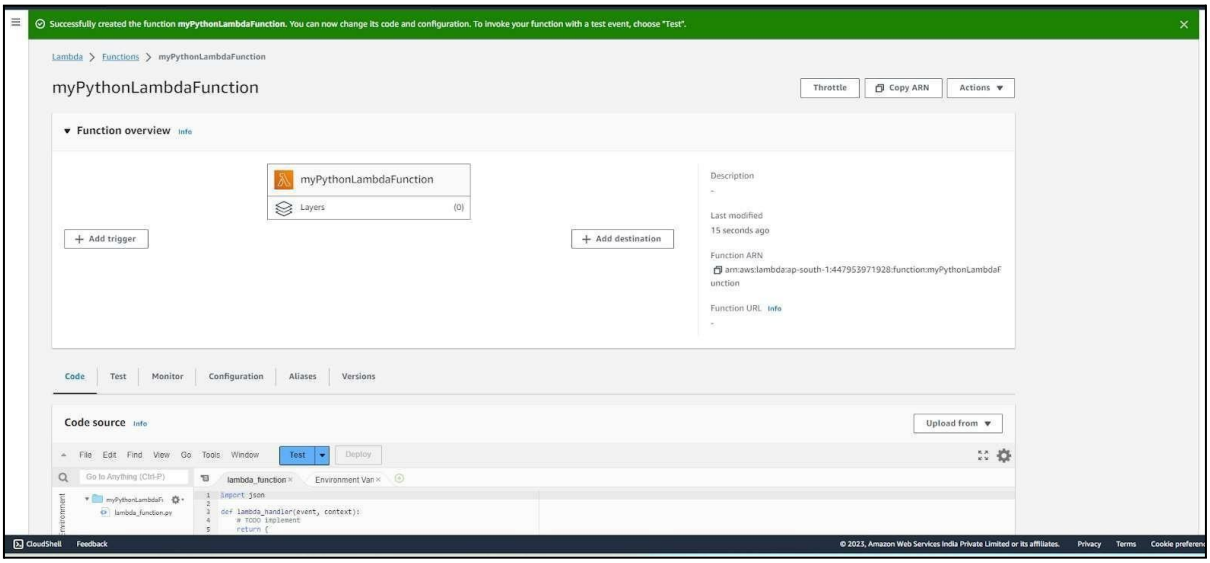


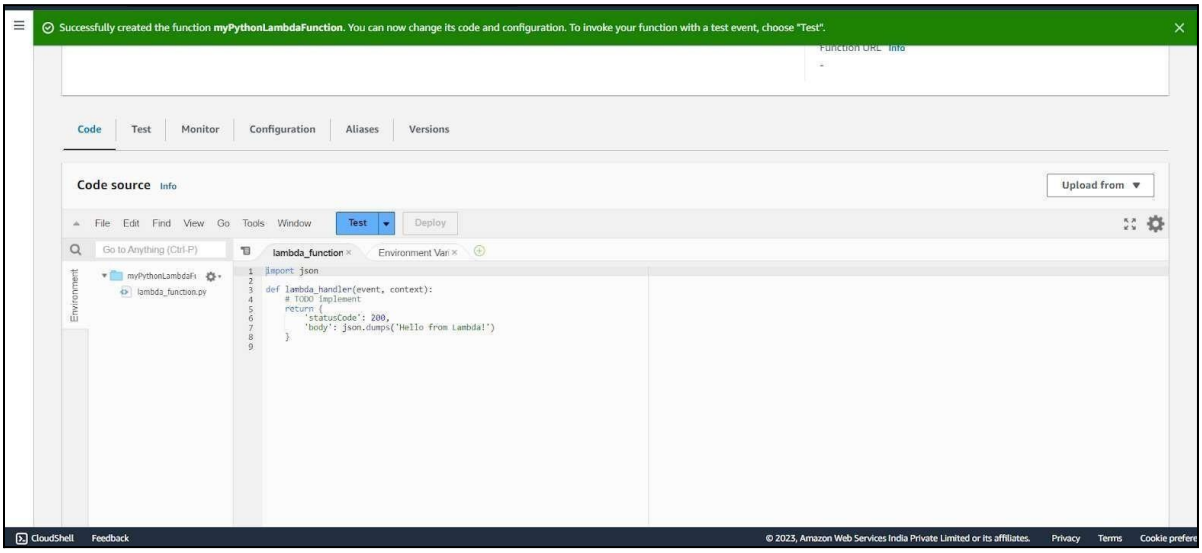




Click on the Create button.

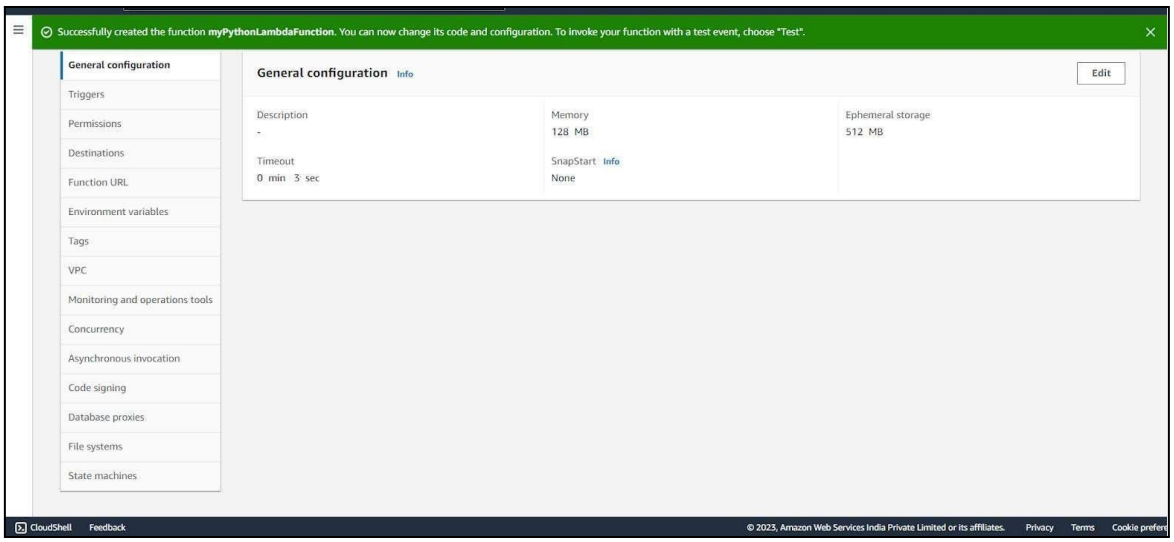
3. This process will take a while to finish and after that, you'll get a message that your function was successfully created.

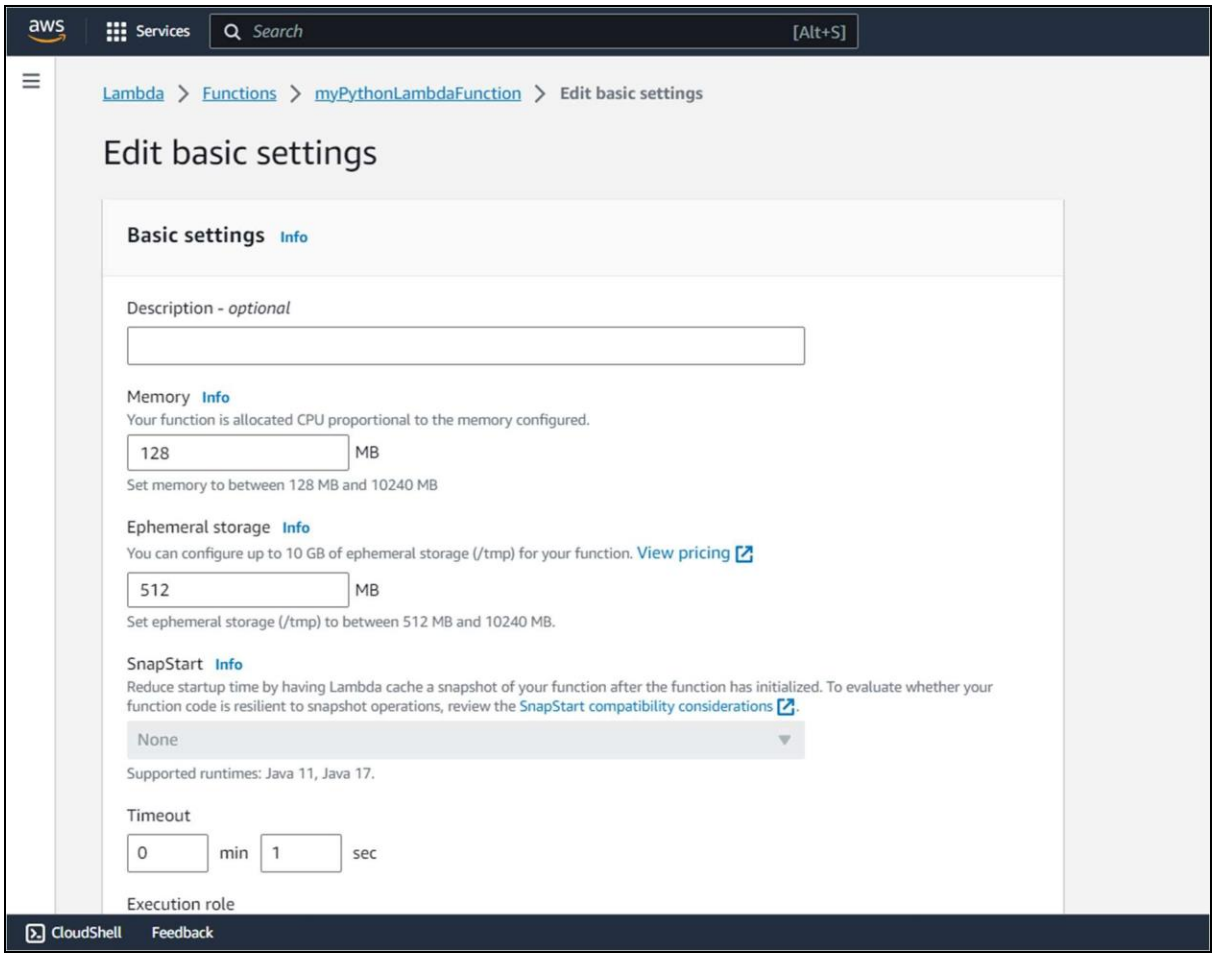




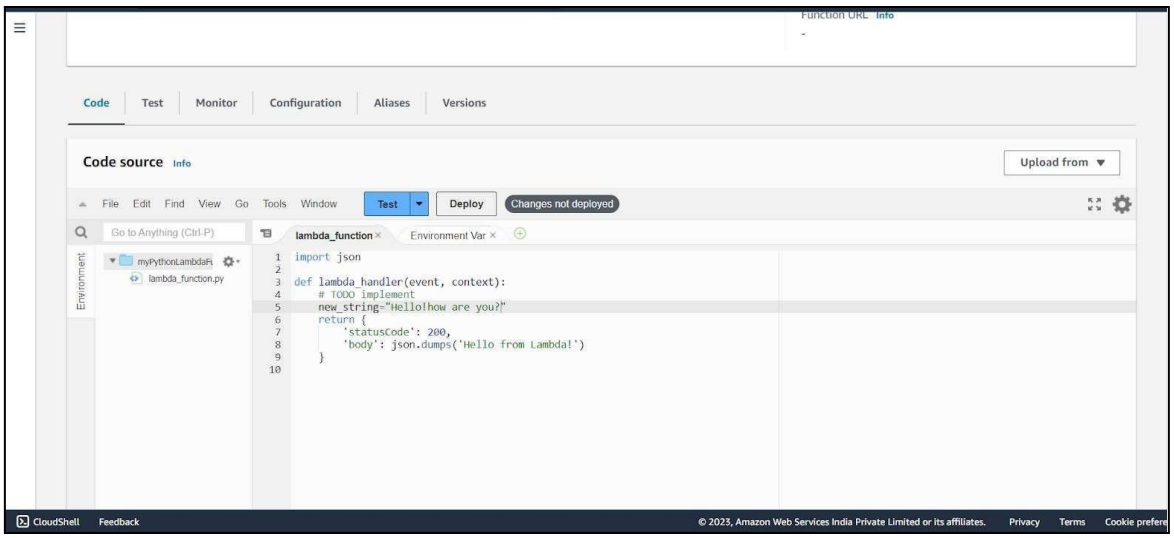
4. To change the configuration, open up the Configuration tab and under General Configuration, choose Edit.

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.

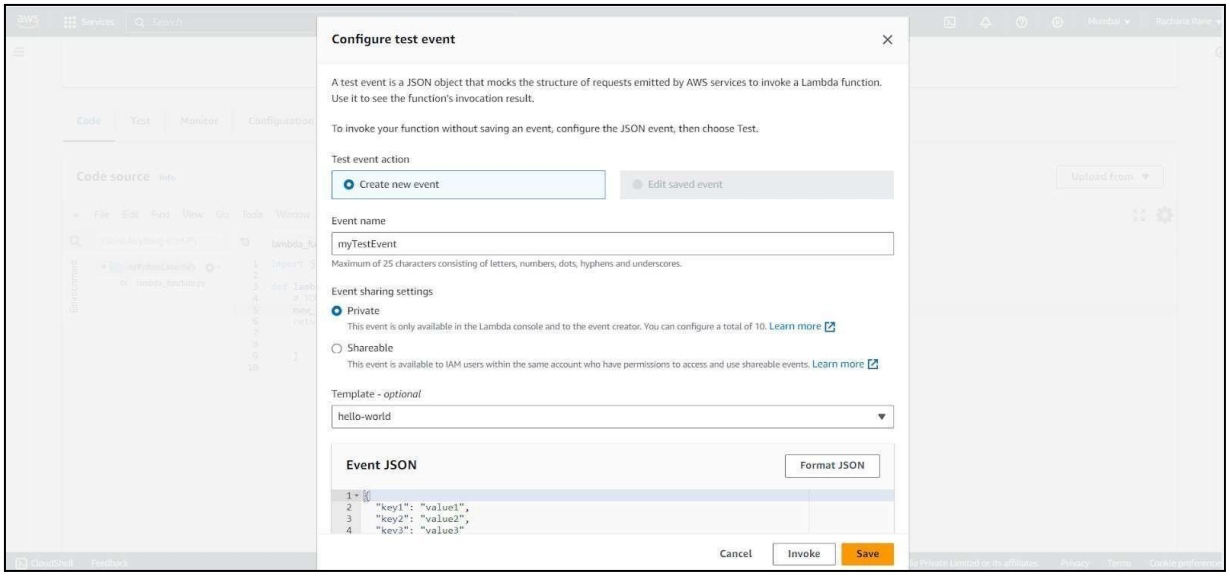




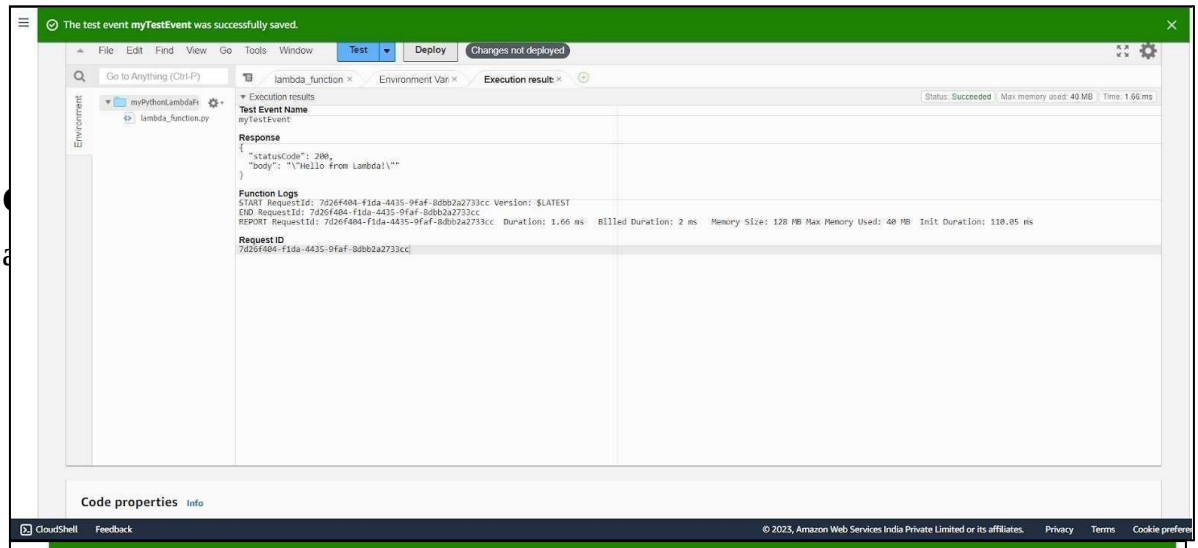
5. You can make changes to your function inside the code editor. You can also upload a zip file of your function or upload one from an S3 bucket if needed. Press Ctrl + S to save the file and click Deploy to deploy the changes.

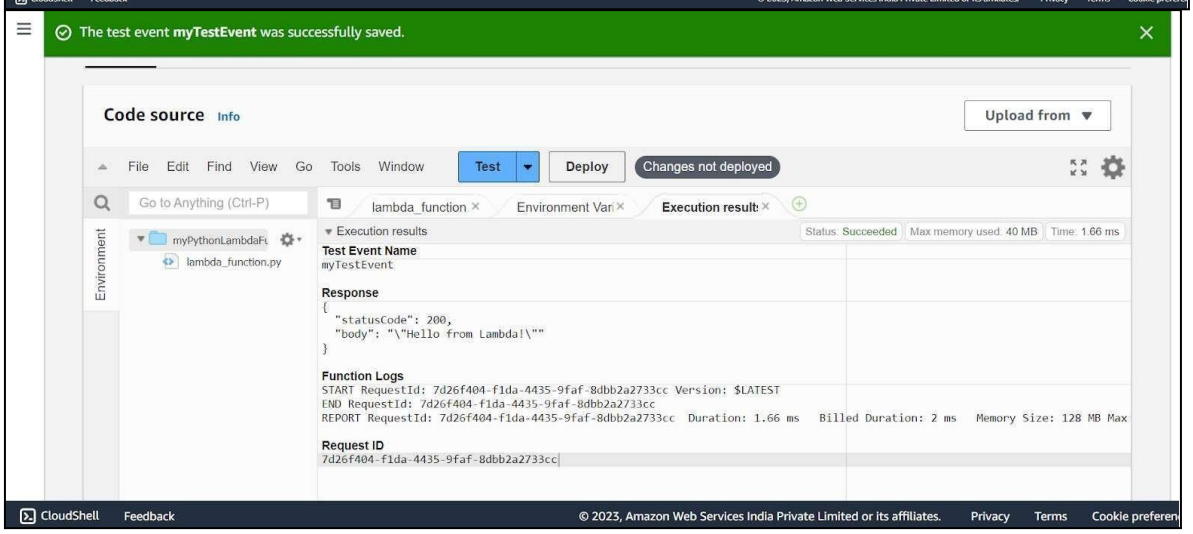


6. Click on Test and you can change the configuration, like so. If you do not have anything in the request body, it is important to specify two curly braces as valid JSON, so make sure they are there.



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


The test event myTestEvent was successfully saved.
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 Memory Used: 40 MB Init Duration: 110.05 ms
Request ID
7d26f404-f1da-4435-9faf-8dbb2a2733cc
Code properties Info
CloudShell
Feedback
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie prefere


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

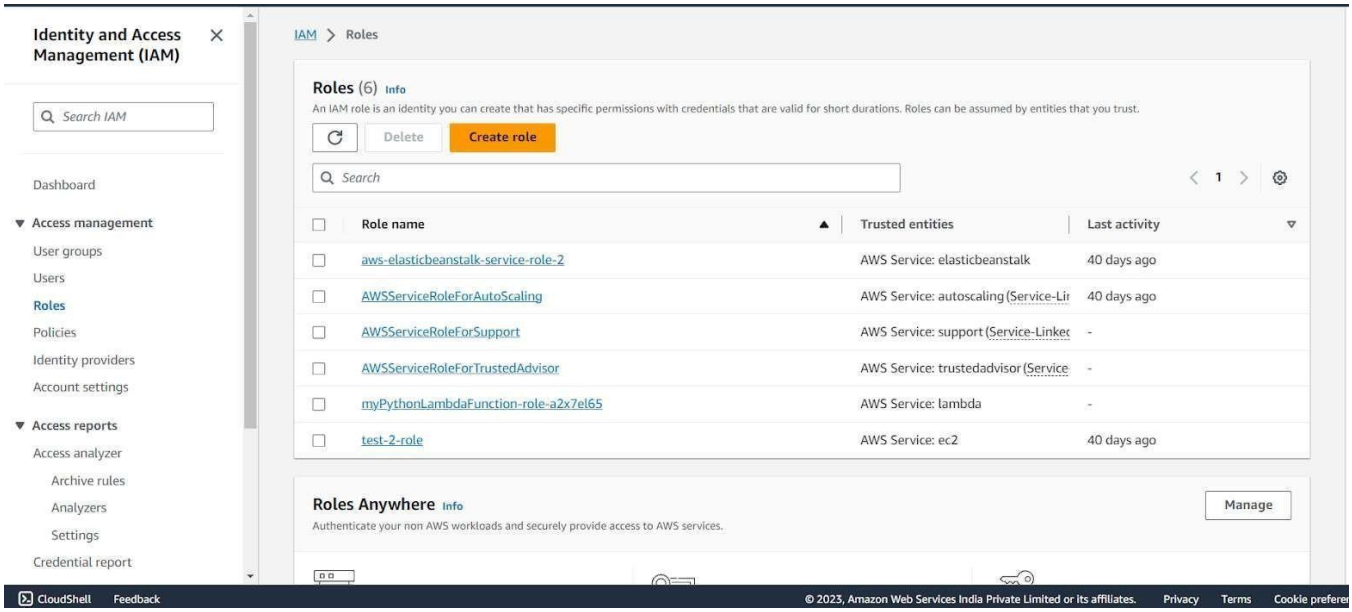
Adv. DevOps Exp. 12

Name-Laksh .V. Sodhai

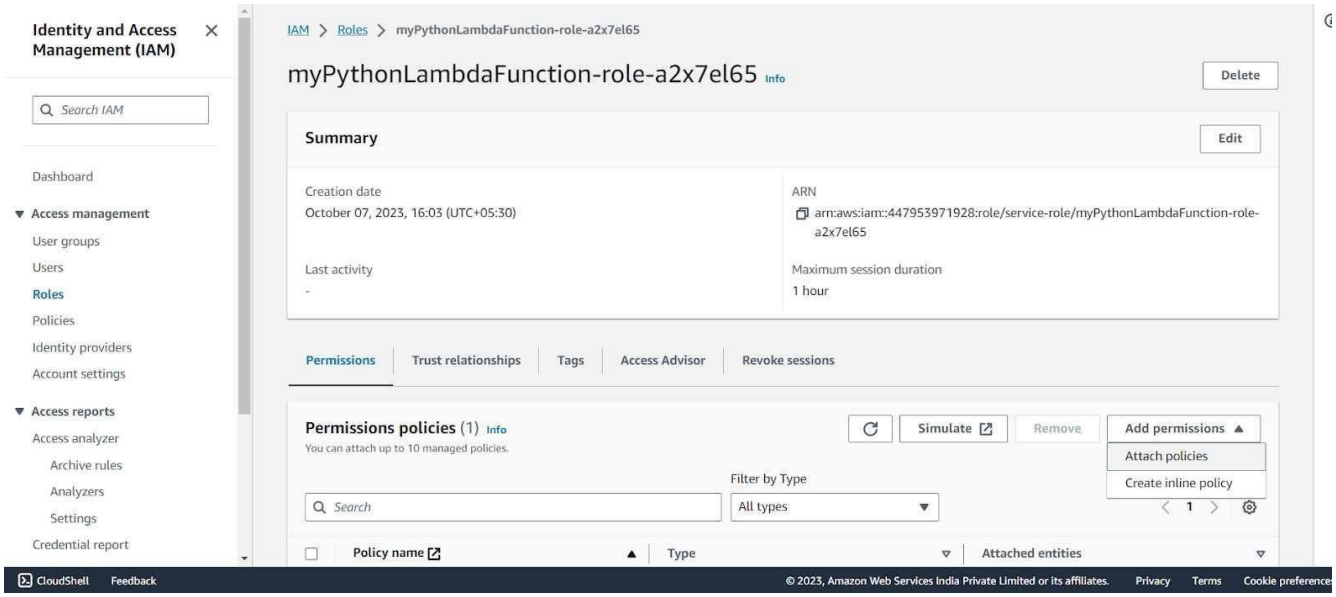
Class-D15A

Roll No-59

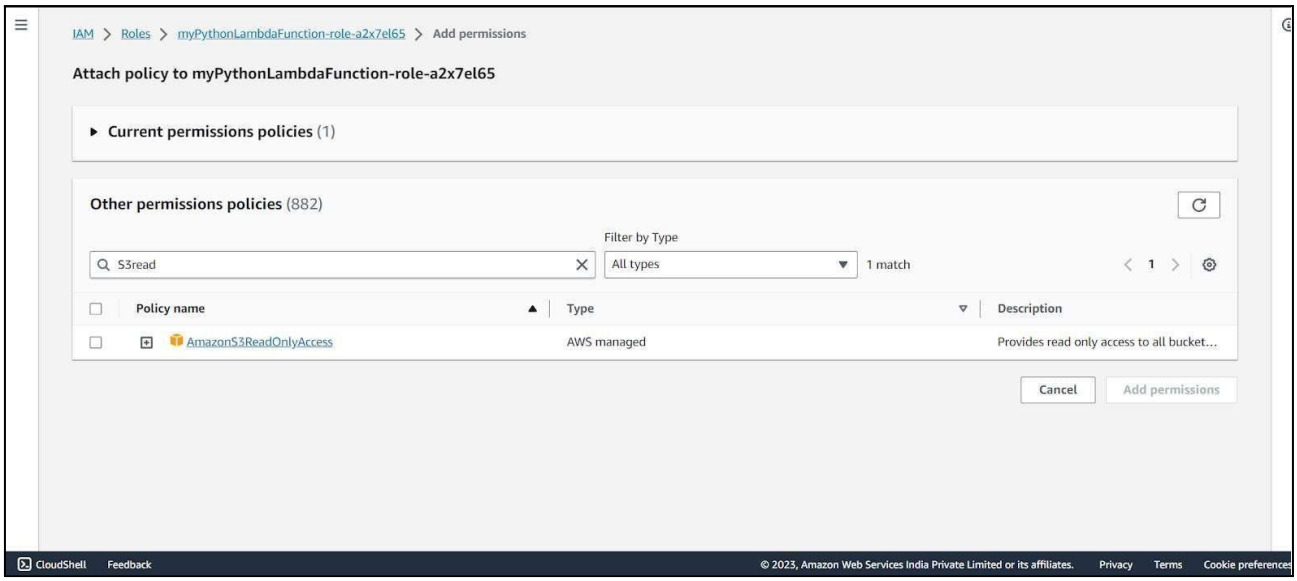
Step 1: Open up the IAM Console and under Roles, choose the Role we previously created for the Python Lambda Function (You can find your role name configuration of your Lambda function).



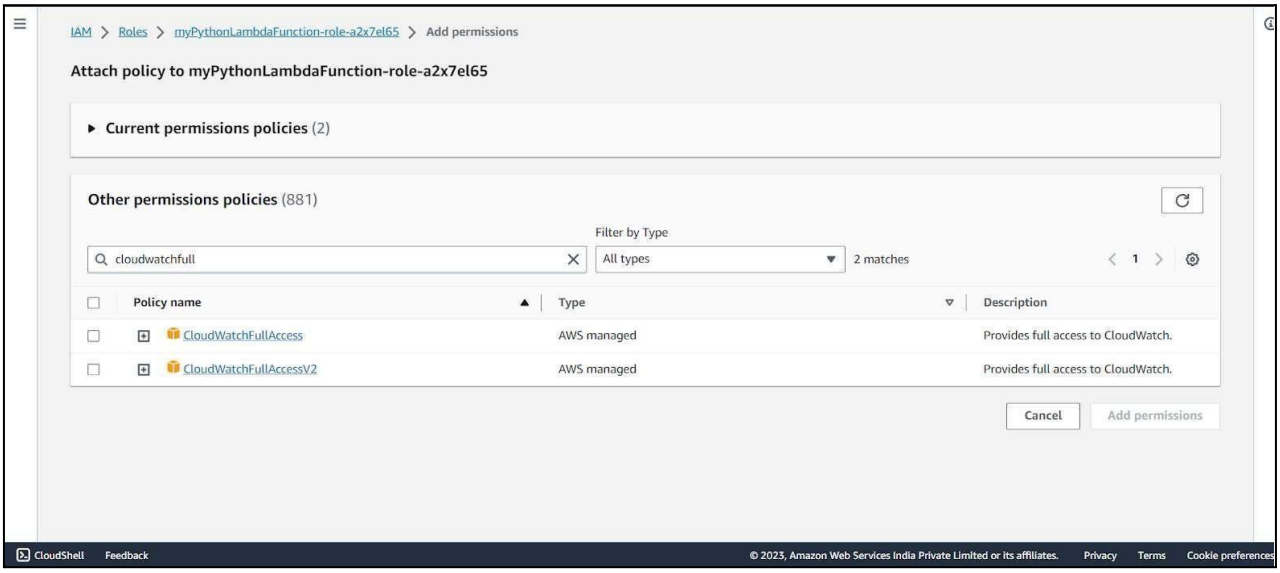
Step 2: Under Attach Policies, add S3-ReadOnly and CloudWatchFull permissions to this role.



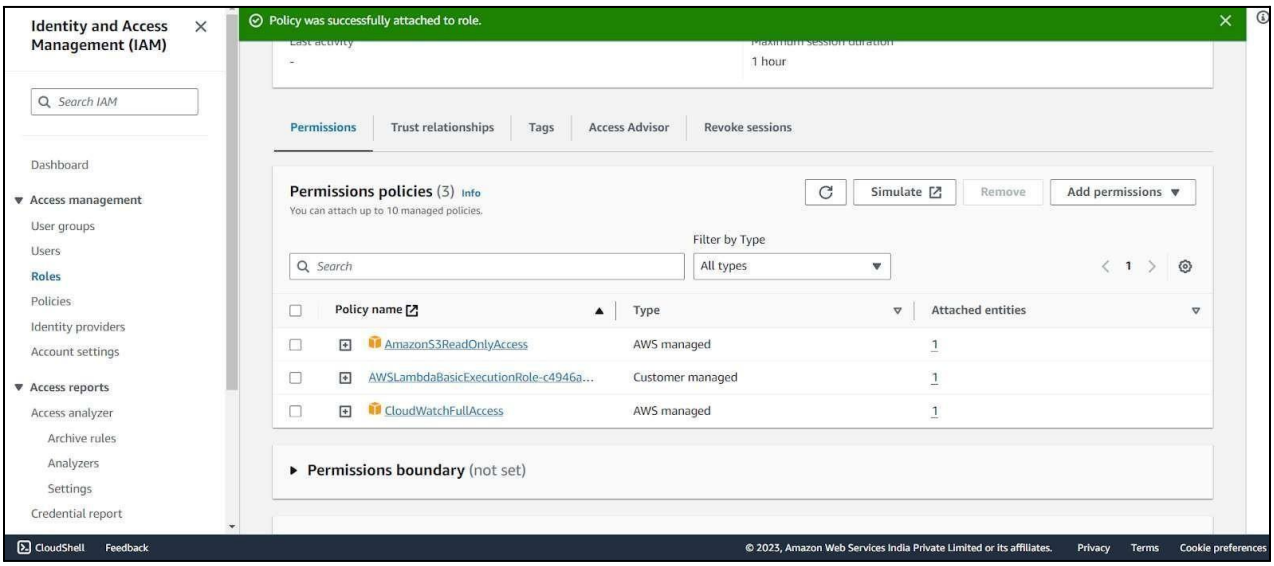
S3-ReadOnly

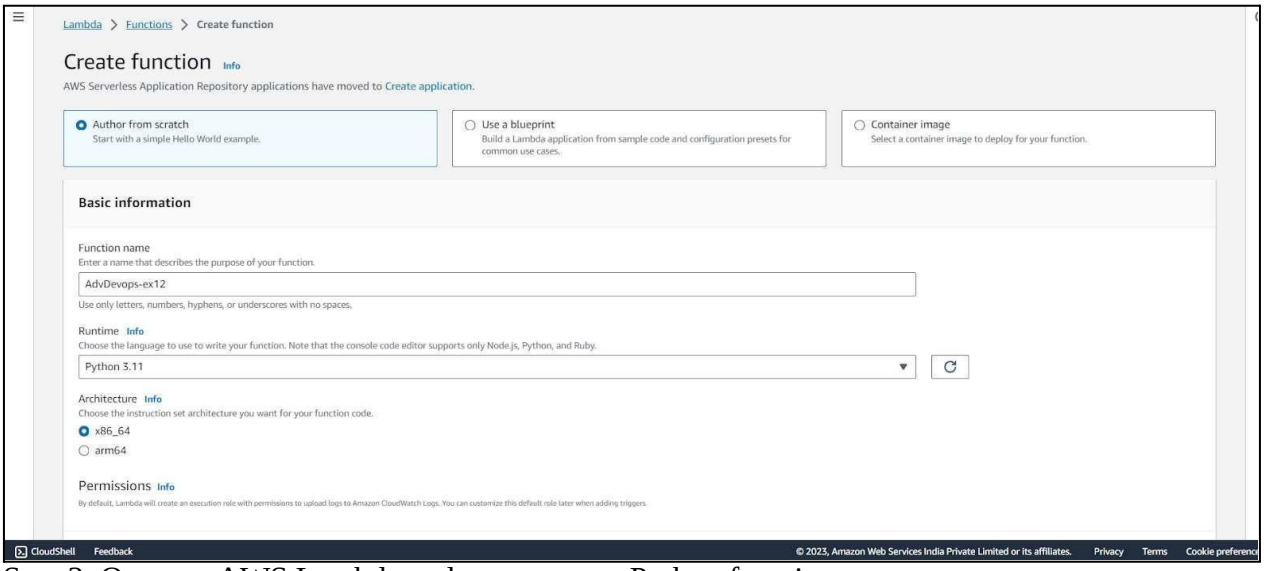


CloudWatchFull



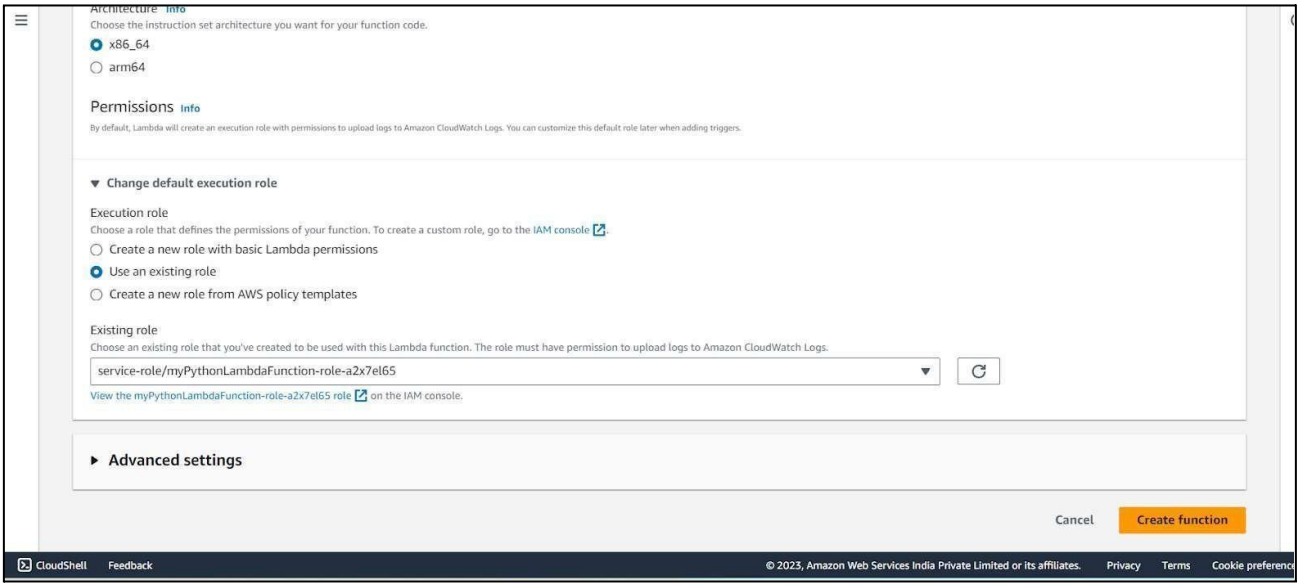
After successful attachment of policy you will see something like this you will be able to see the updated policies.



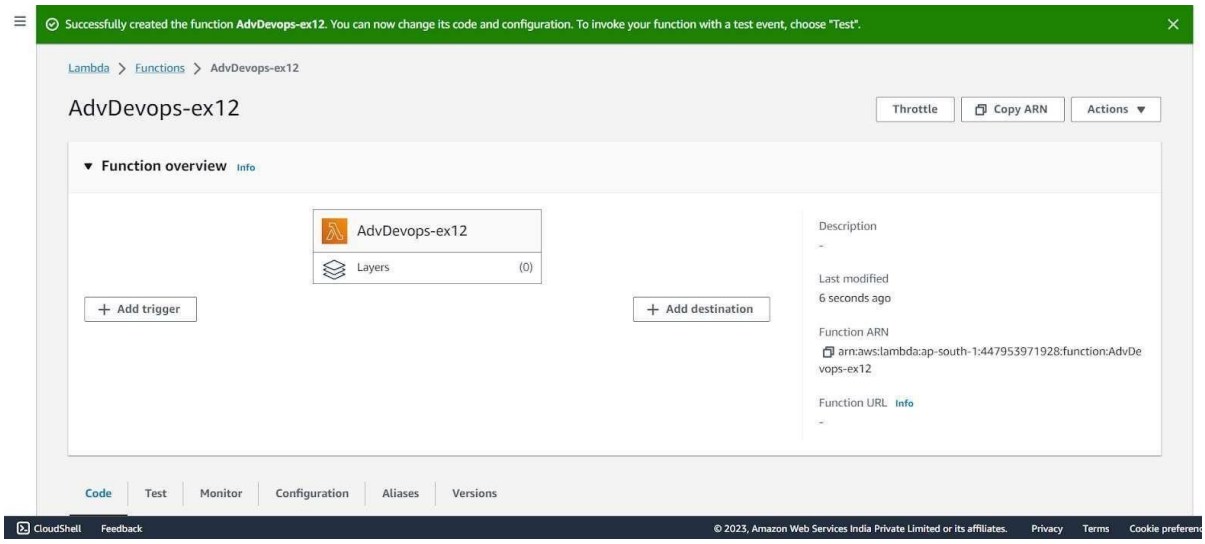


Step 3: Open up AWS Lambda and create a new Python function.

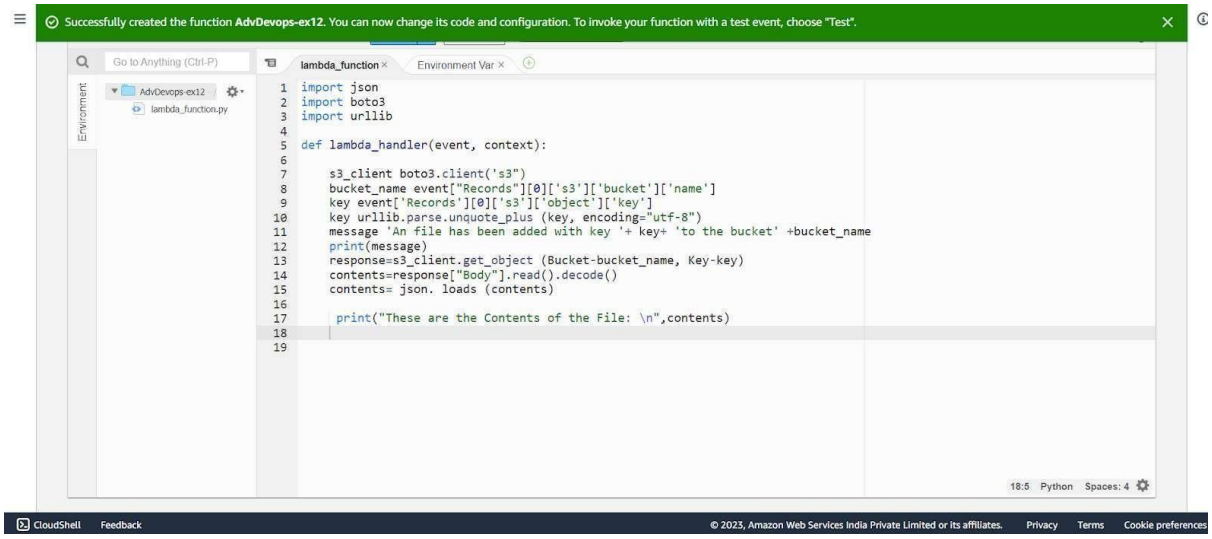
Under Execution Role, choose the existing role, then select the one which was previously created and to which we just added permissions.



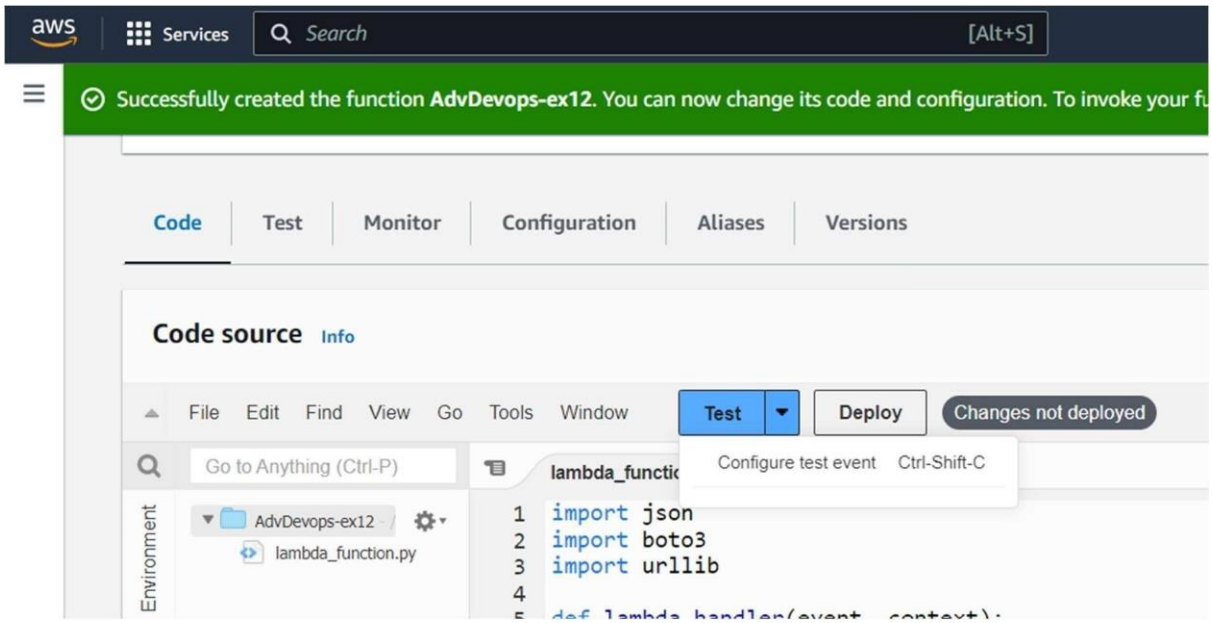
Step 4: The function is up and running.

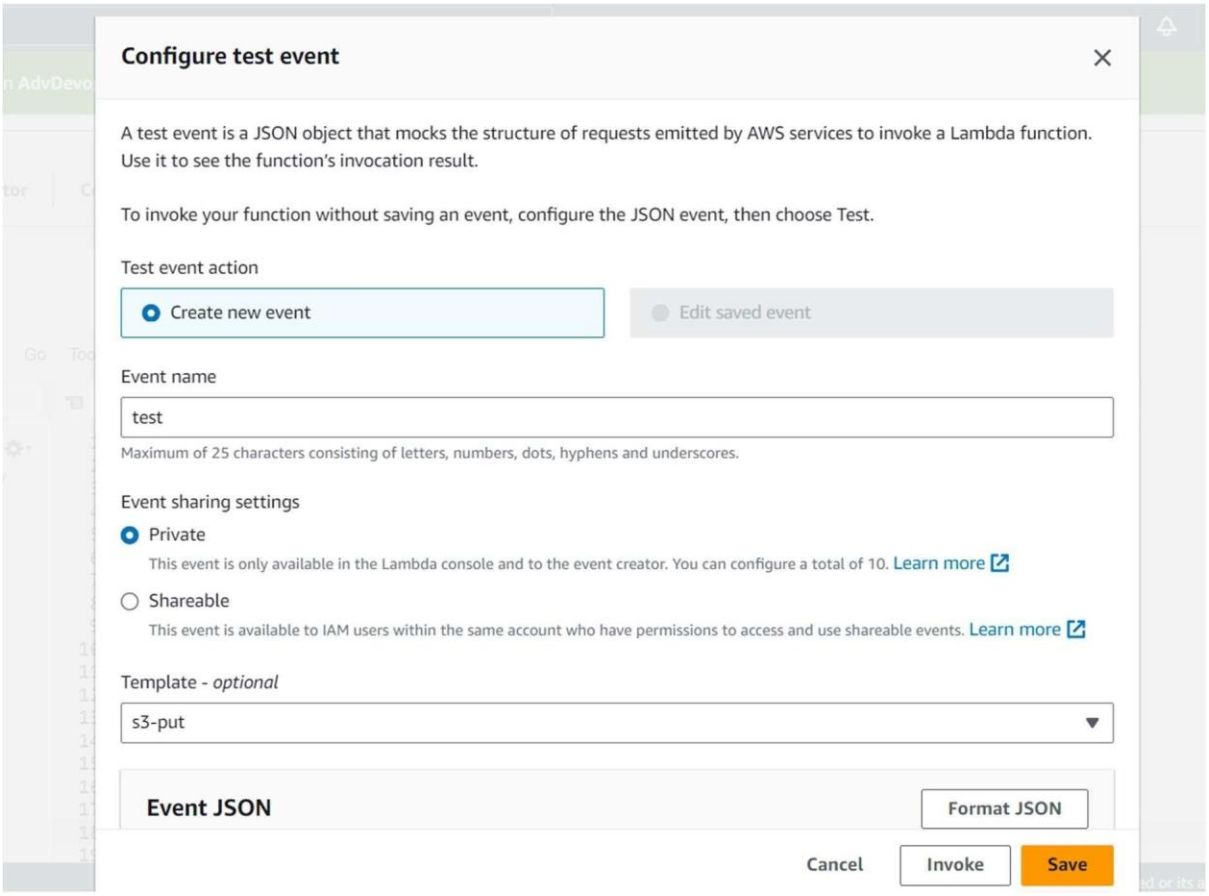


Step 5: Make the following changes to the function and click on the deploy button. This code basically logs a message and logs the contents of a JSON file which is uploaded to an S3 Bucket and then deploy the code.



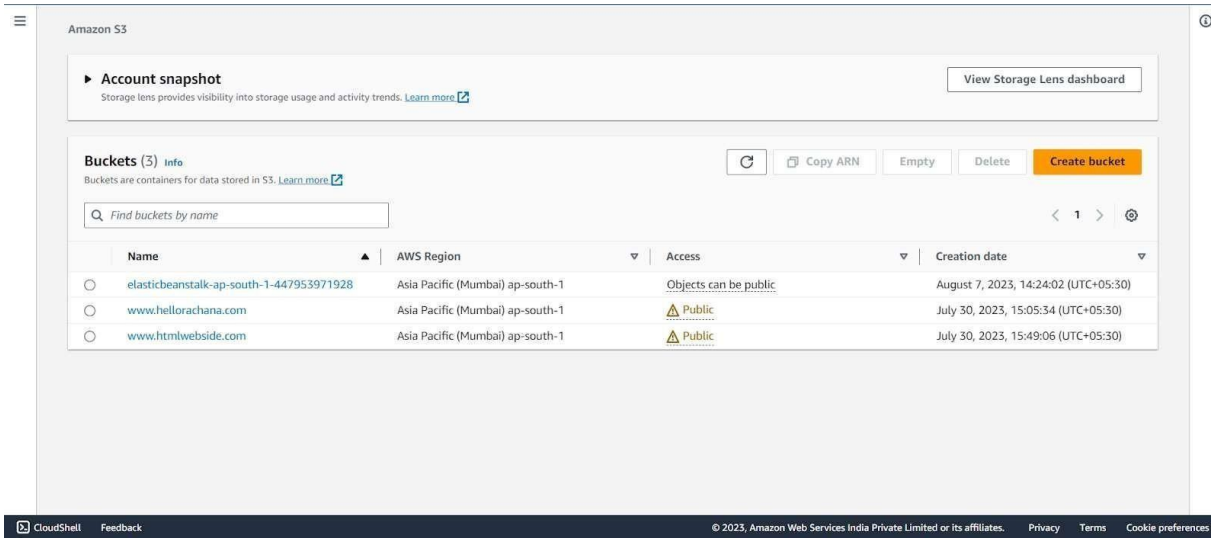
Step 6: Click on Test and choose the ‘S3 Put’ Template.



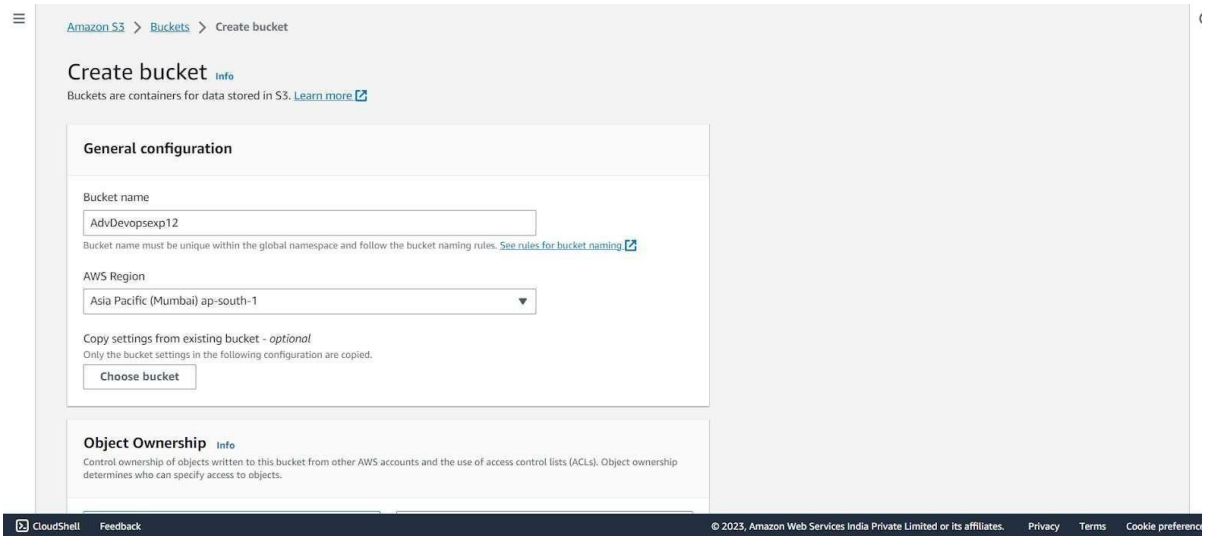


And Save it.

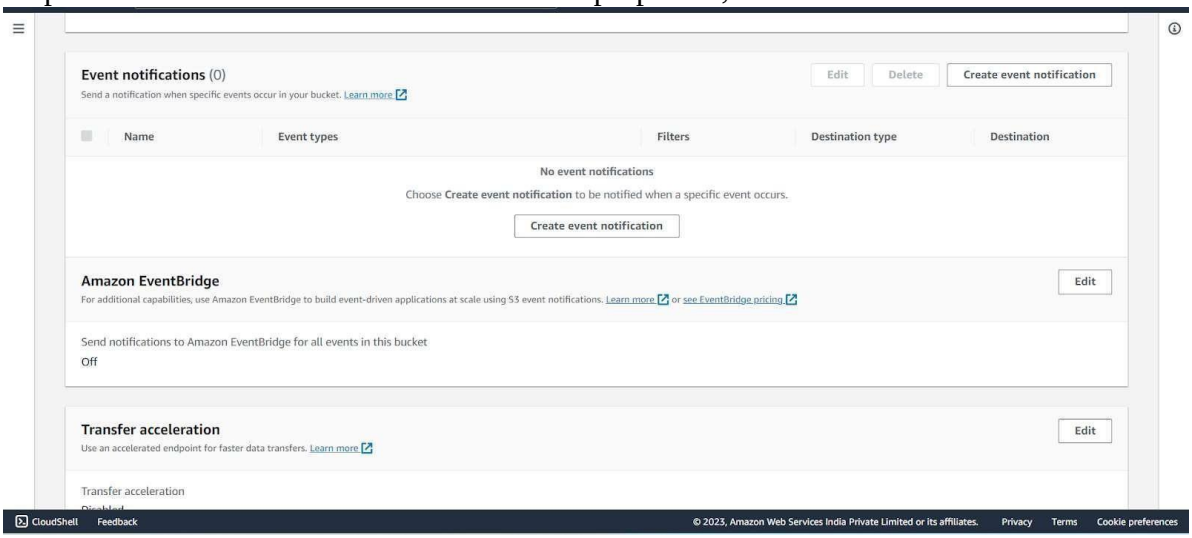
Step 7: Open up the S3 Console and create a new bucket.



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

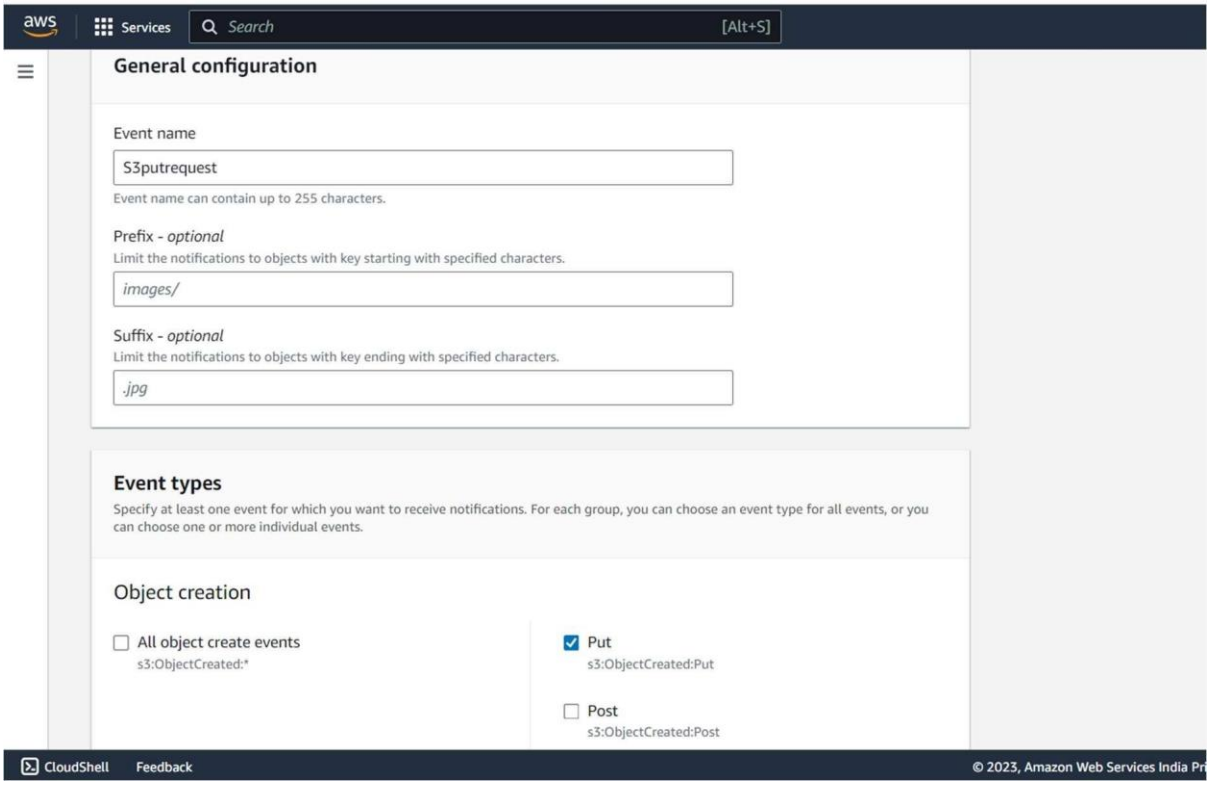


Step 9: Click on the created bucket and under properties, look for events.

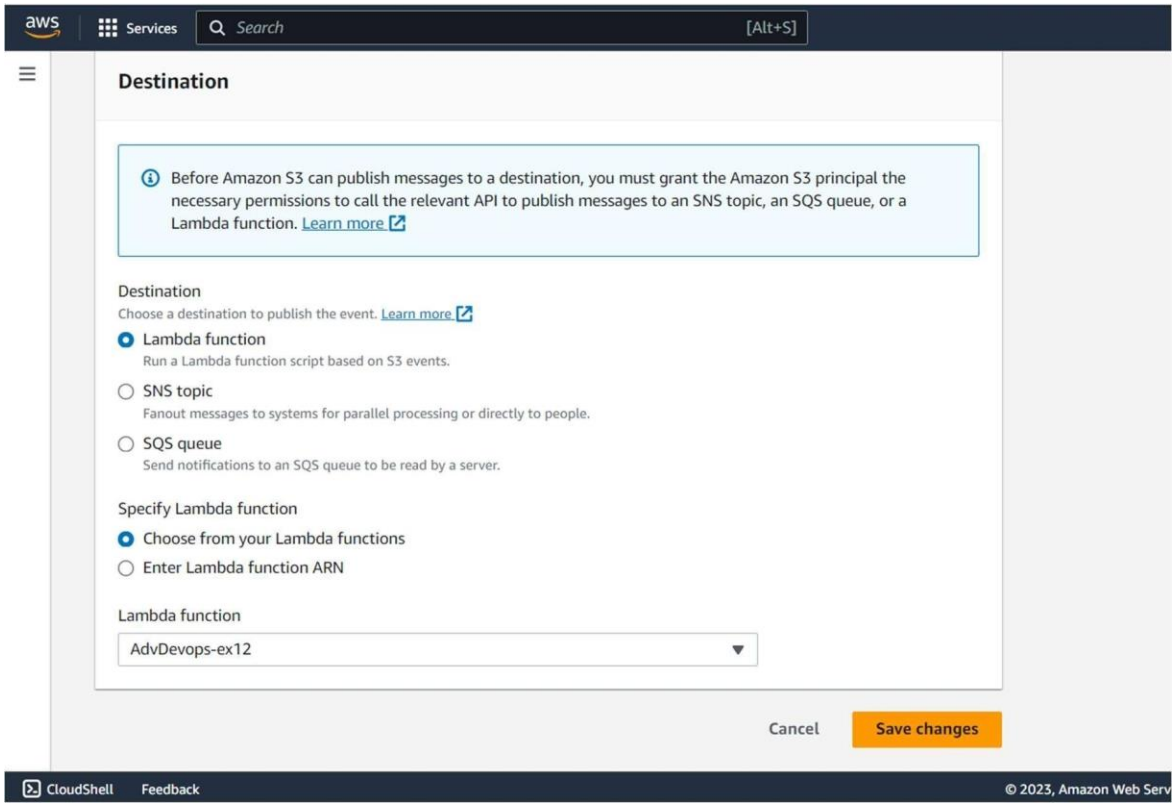


Click on Create Event Notification.

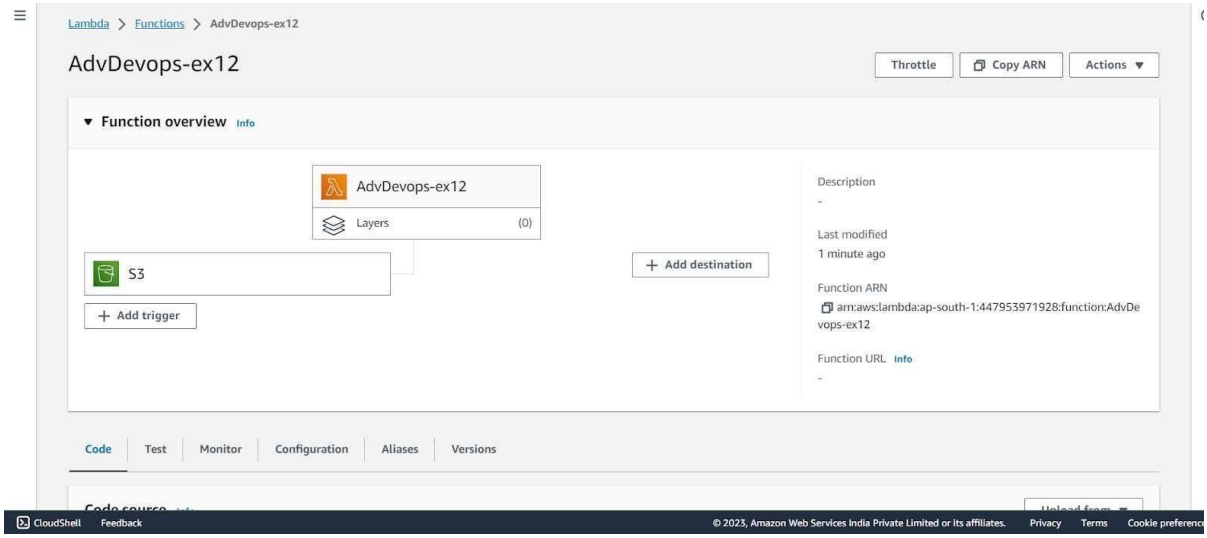
Step 10: Mention an event name and check Put under event types.



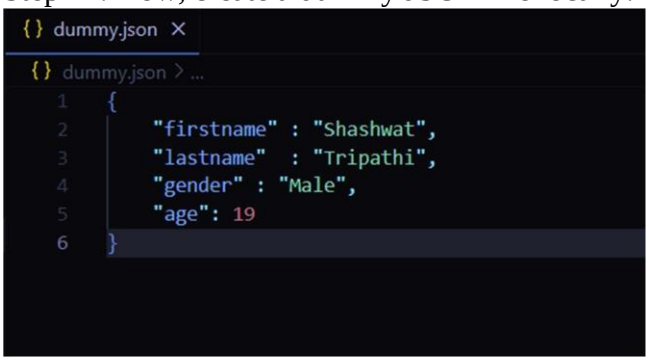
Choose Lambda function as destination and choose your lambda function and save the changes.



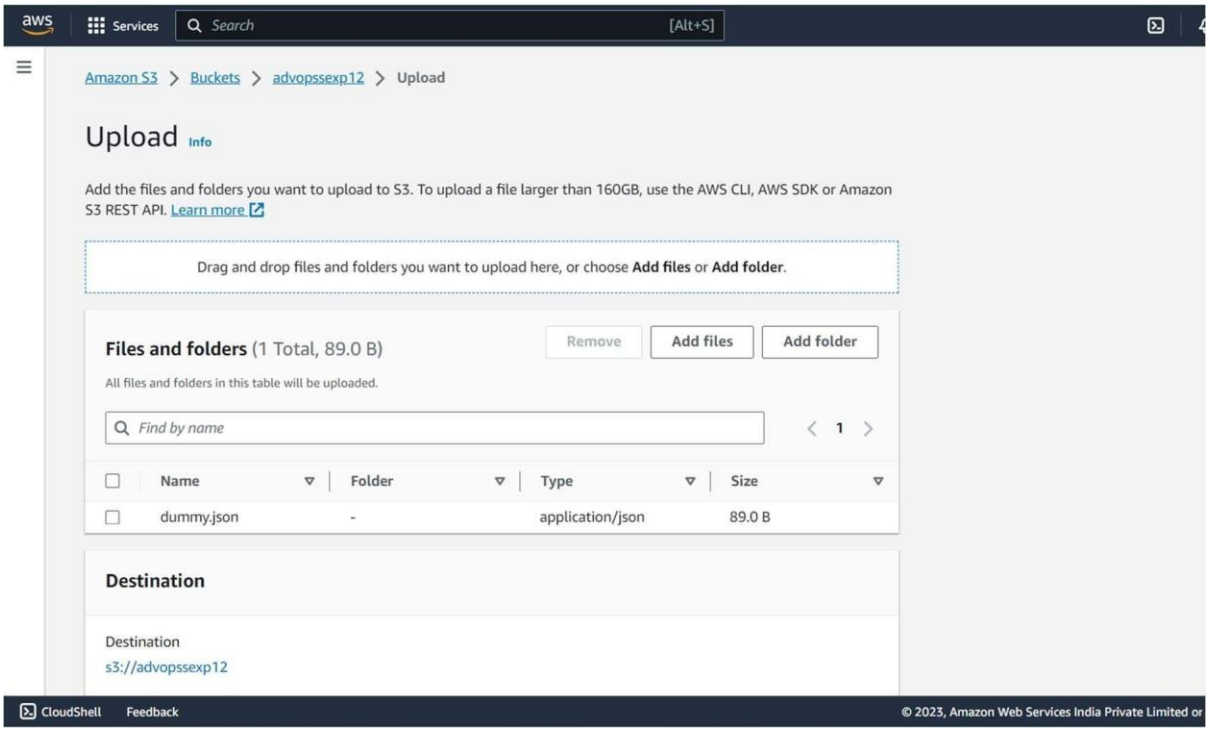
Step 11: Refresh the Lambda function console and you should be able to see an S3 Trigger in the overview.



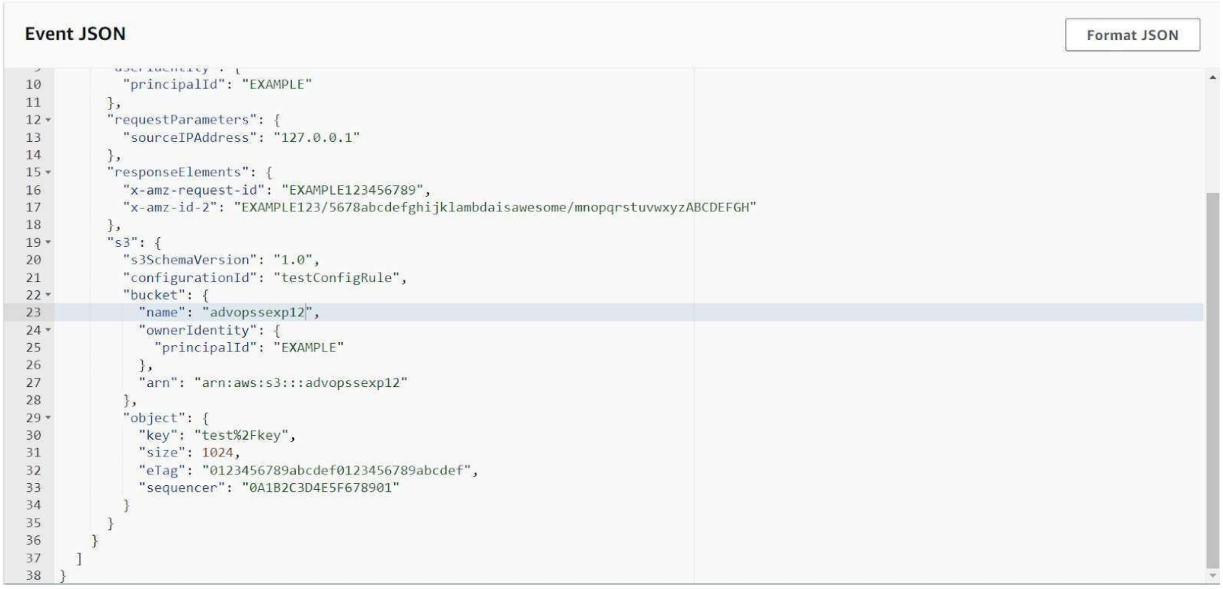
Step 12: Now, create a dummy JSON file locally.



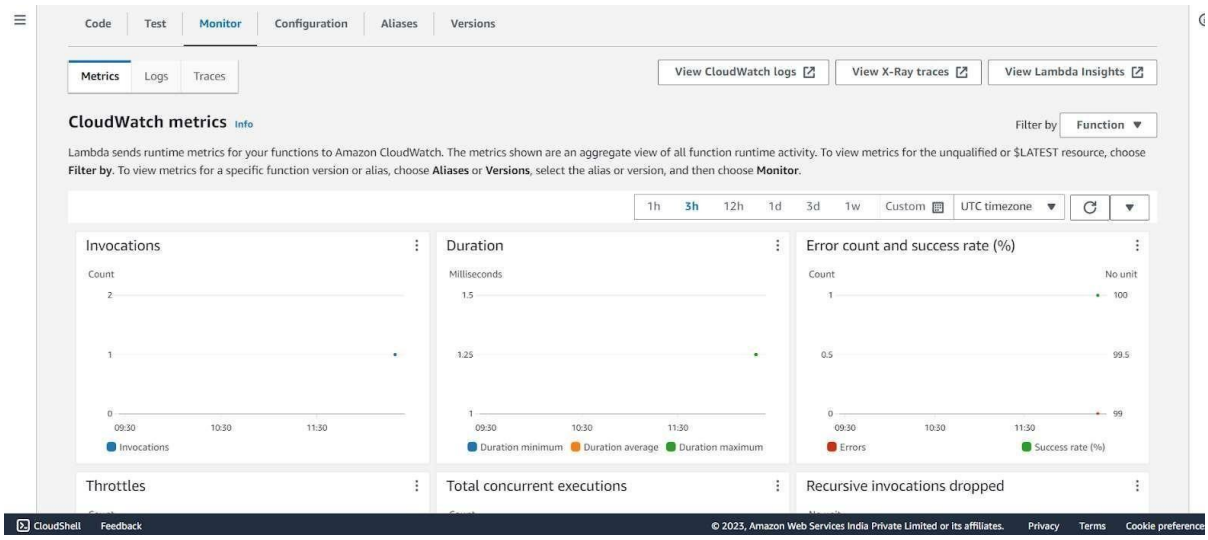
Step 13: Go back to your S3 Bucket and click on Add Files to upload a new file.
Step 14: Select the dummy data file from your computer and click Upload.



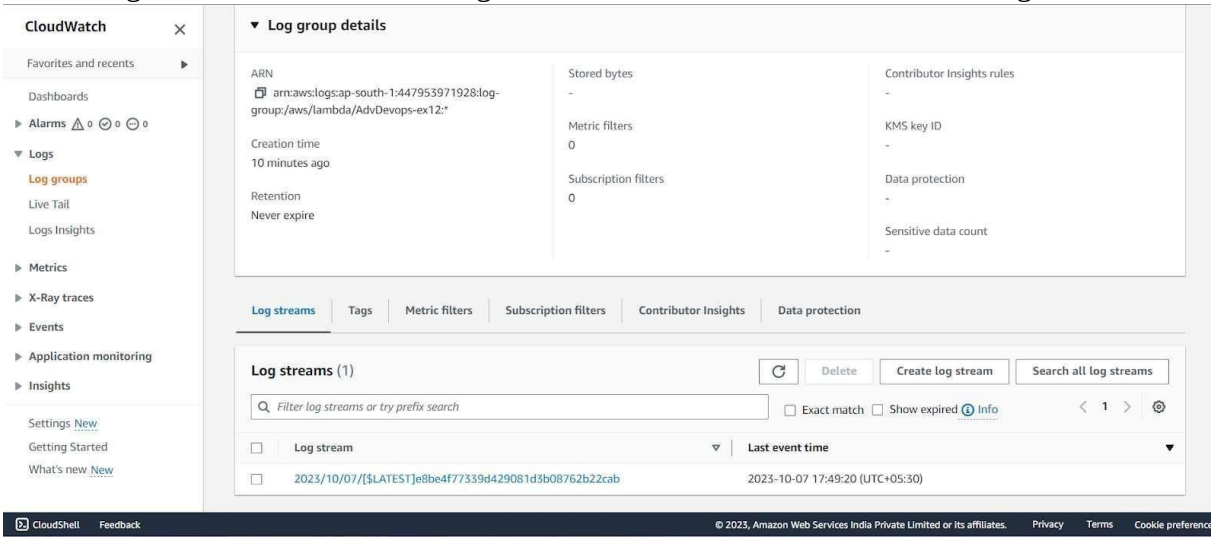
Step 15: After this make the necessary changes in the Test configuration file which we created it previously by replacing the Bucket Name and the ARN of Bucket.



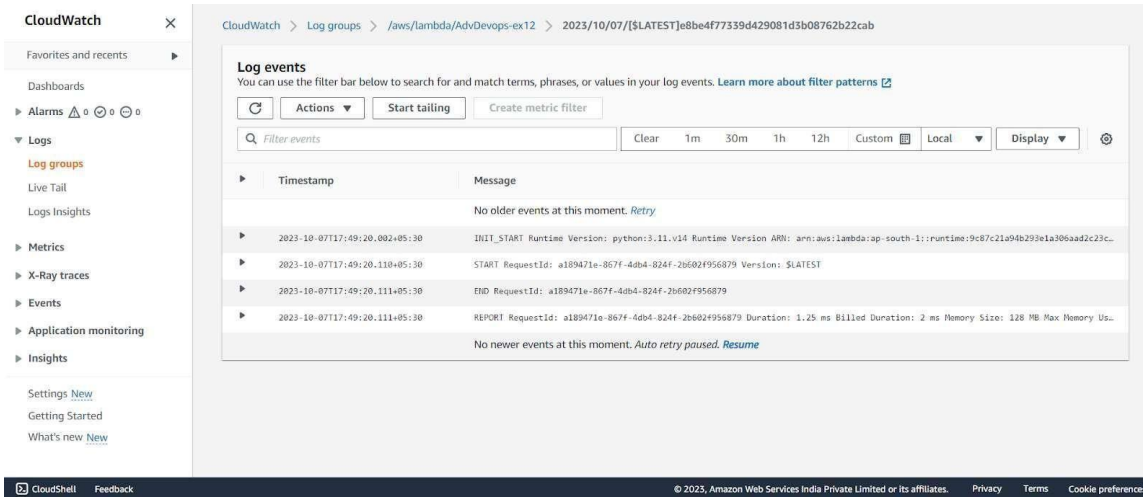
Step 16: Go back to your Lambda function , Refresh it and check the Monitor tab.



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.