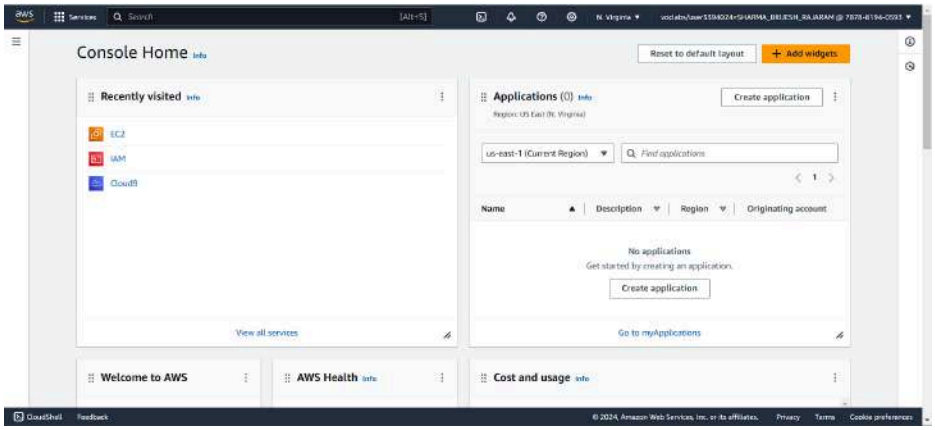
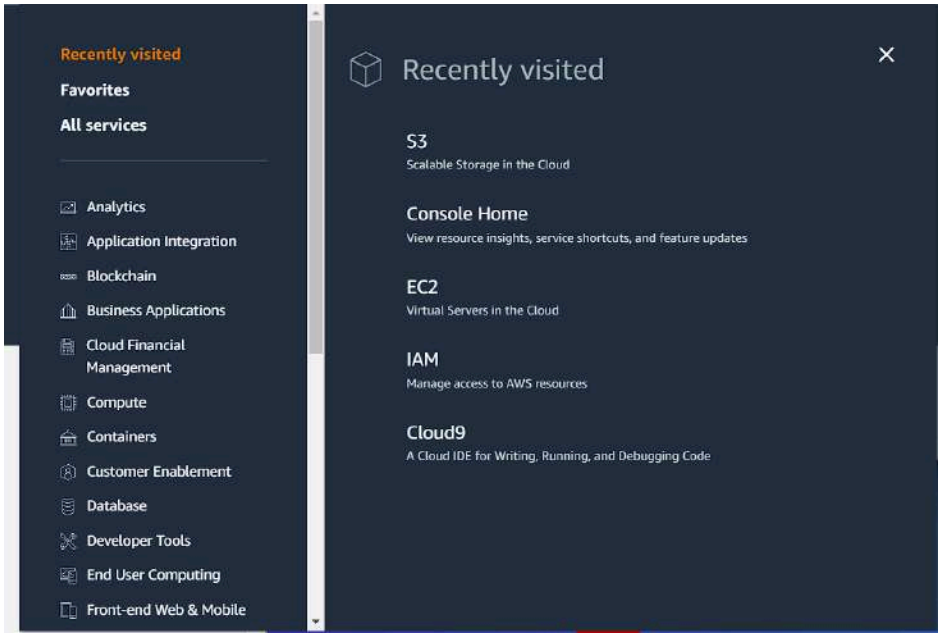
Name:Krushikesh Shelar D15-C Roll No:51

Aim: Hosting a static website on Amazon Web Services (S3)

1) Open the AWS console home

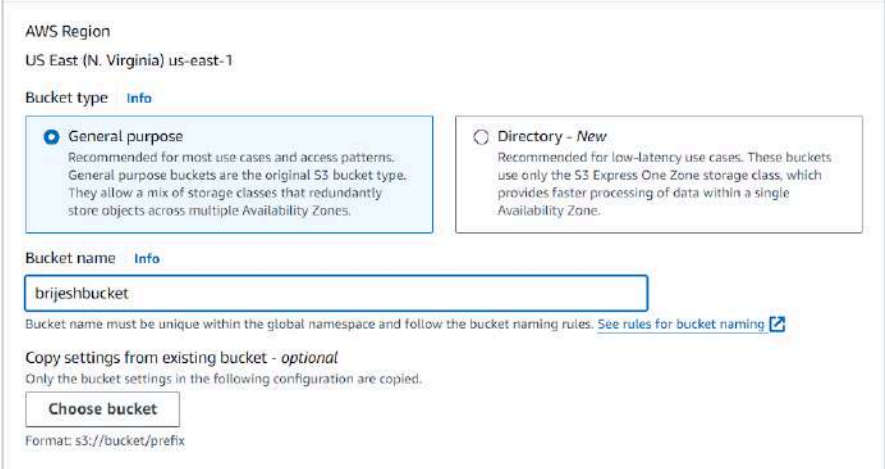


2) Navigate to the S3 to host the website



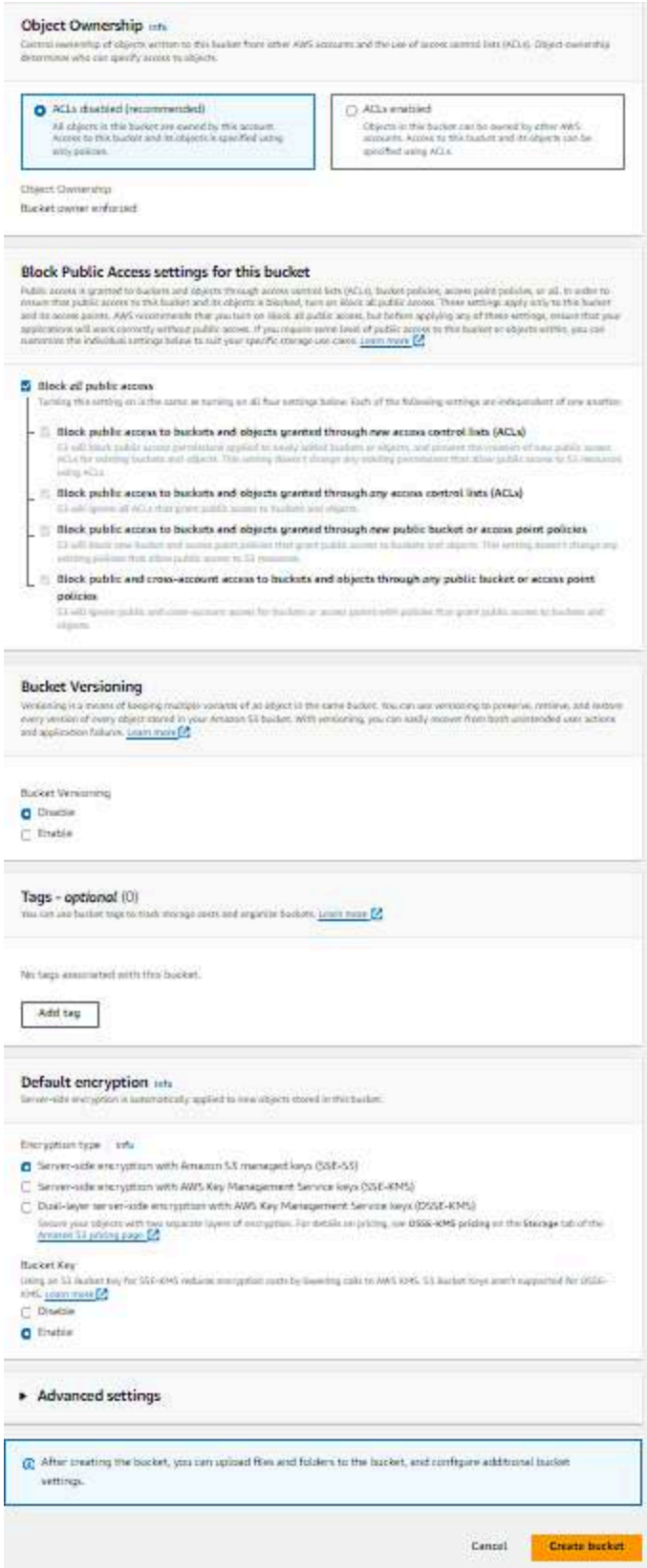
3) On S3, click on create bucket

4) Click on Bucket type as General Purpose and name the bucket.

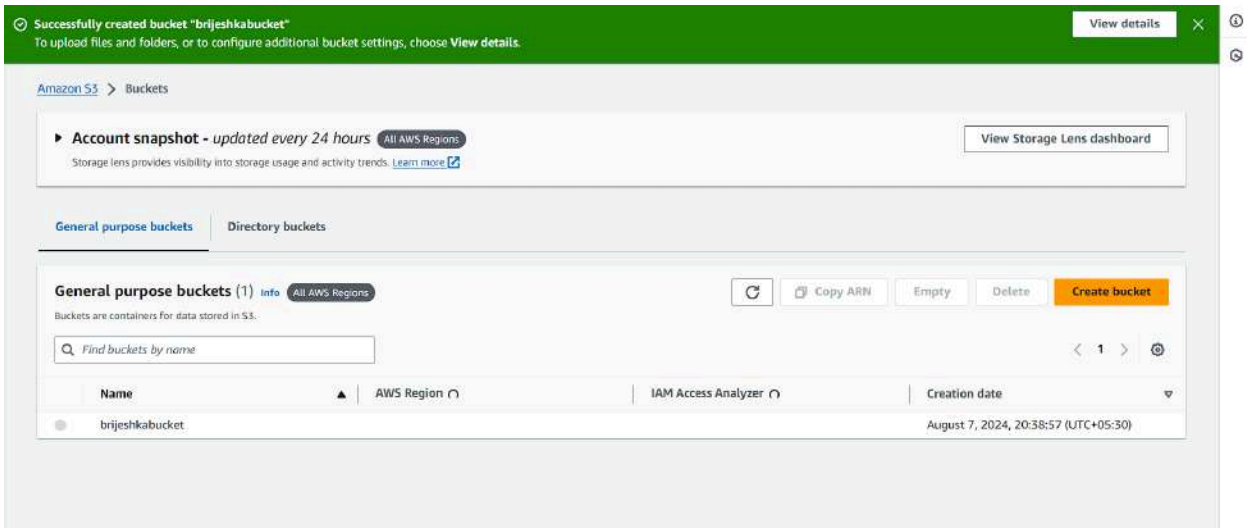


5) Keep the default settings intact, checkingfor bucket versioning as disbale and bucket key enabled.

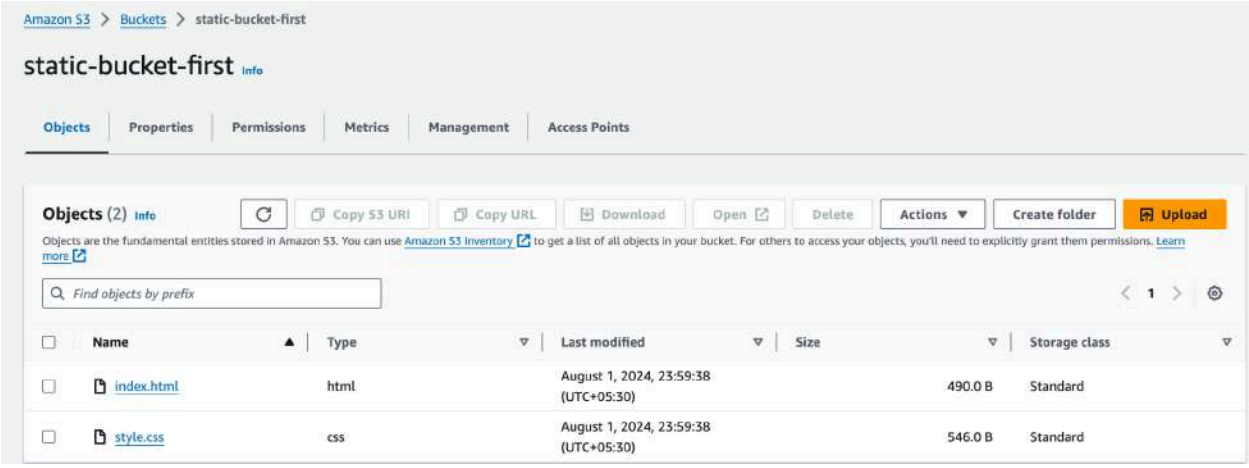
Name:Krushikesh Shelar D15-C Roll No:51



6) After successfully creating the bucket, click on bucket name to change the settings to host the website.



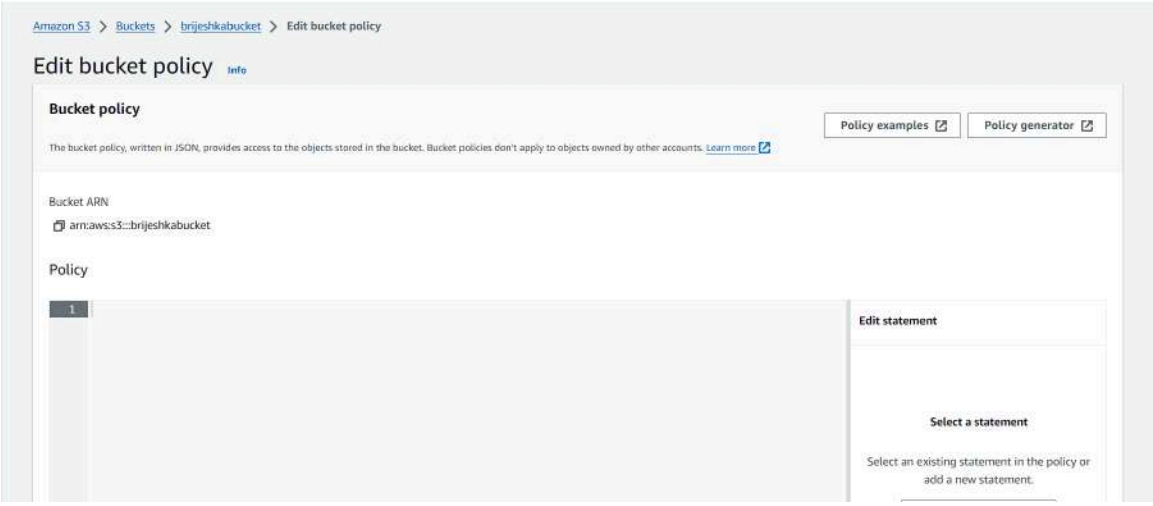
7) Go on Permissions tab and check for Block public access



8) Block public access is default on, we need to uncheck it to ensure the hosted website is public.

9) Now the block public access option is unchecked and hence the website can be hosted successfully.

10) Now, navigate to the edit bucket policy in Properties tab to provide access to the services.

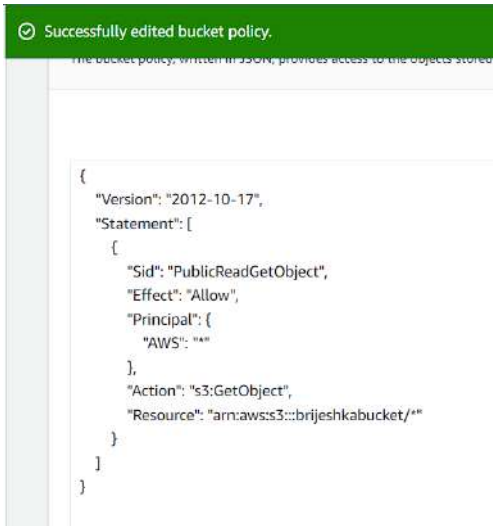


11) Fill the following policy in the empty policy space. Ensure that you change the name of the bucket in Resource with the name of your bucket.

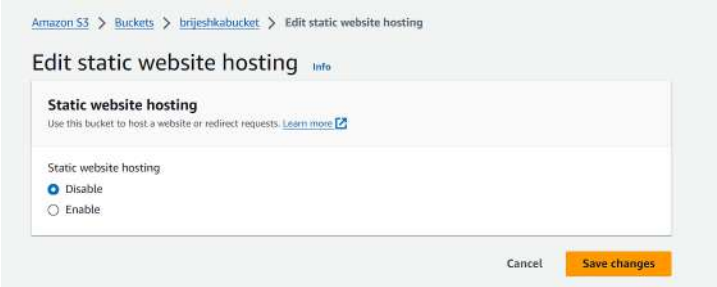
Policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::brijeshkabucket/*"
        }
    ]
}
```

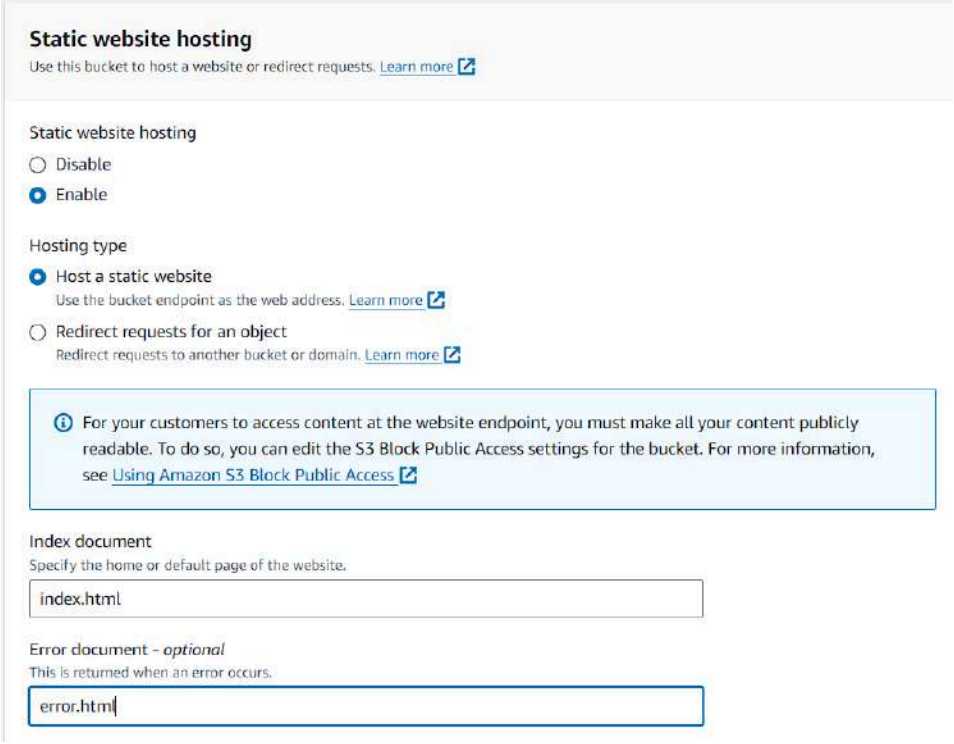
12) After saving the changes, you will see a message.



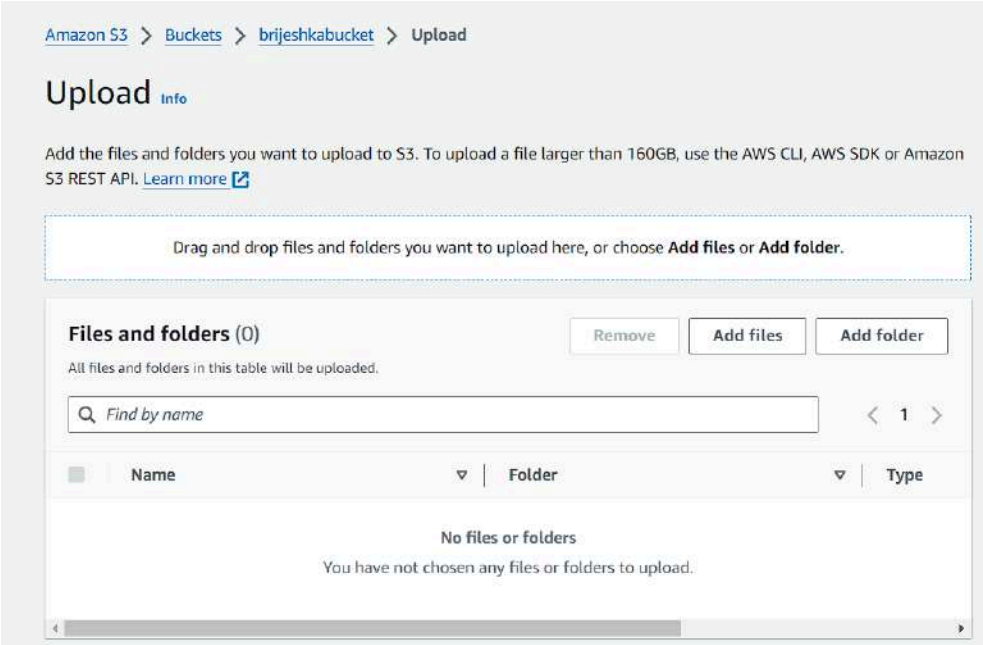
13) Go to the edit static website hosting in the properties tab to use bucket to host websites.



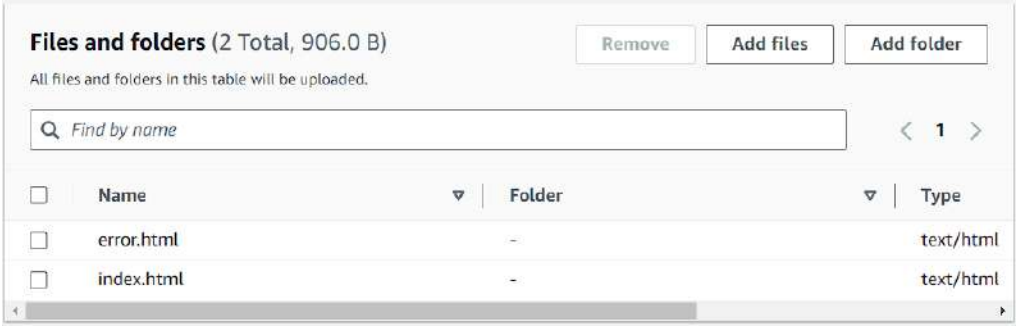
14) Check the options as shown below, and add the names of the file.



15) Navigate to the Upload section and upload the documents with the name as mentioned in the previous section.



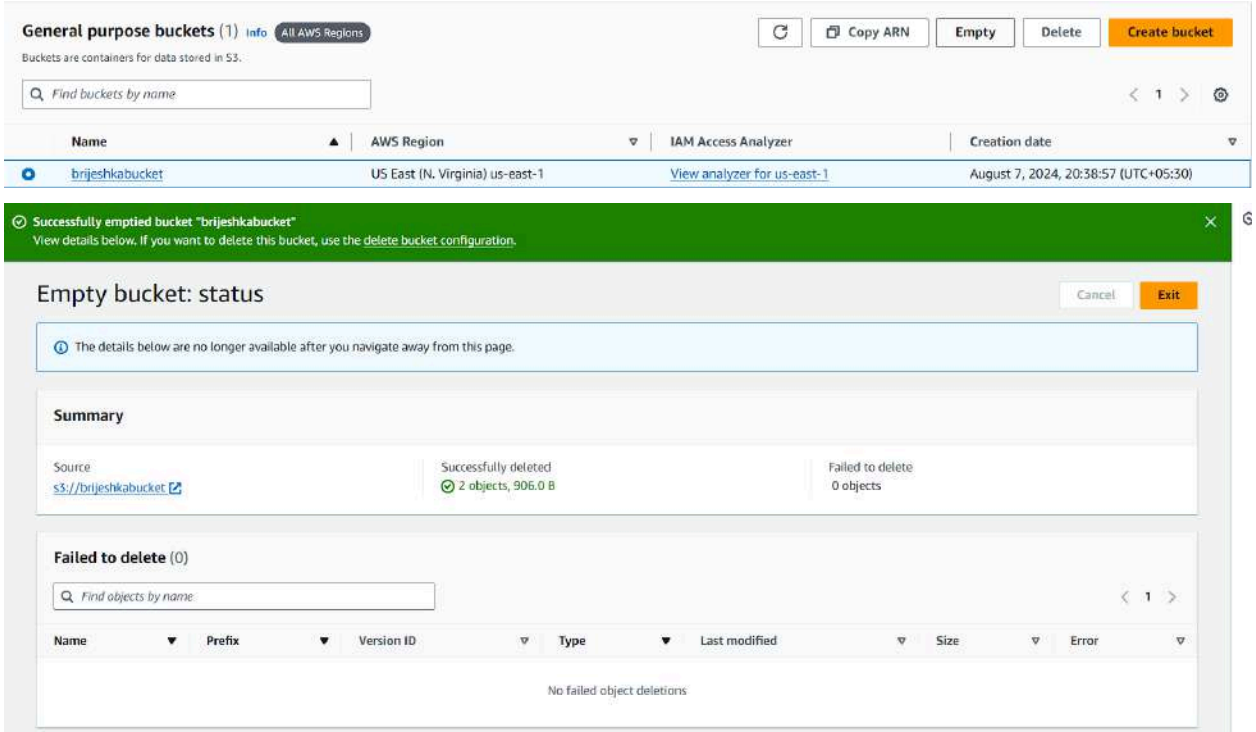
16) Uploaded files will be visible after successful upload.



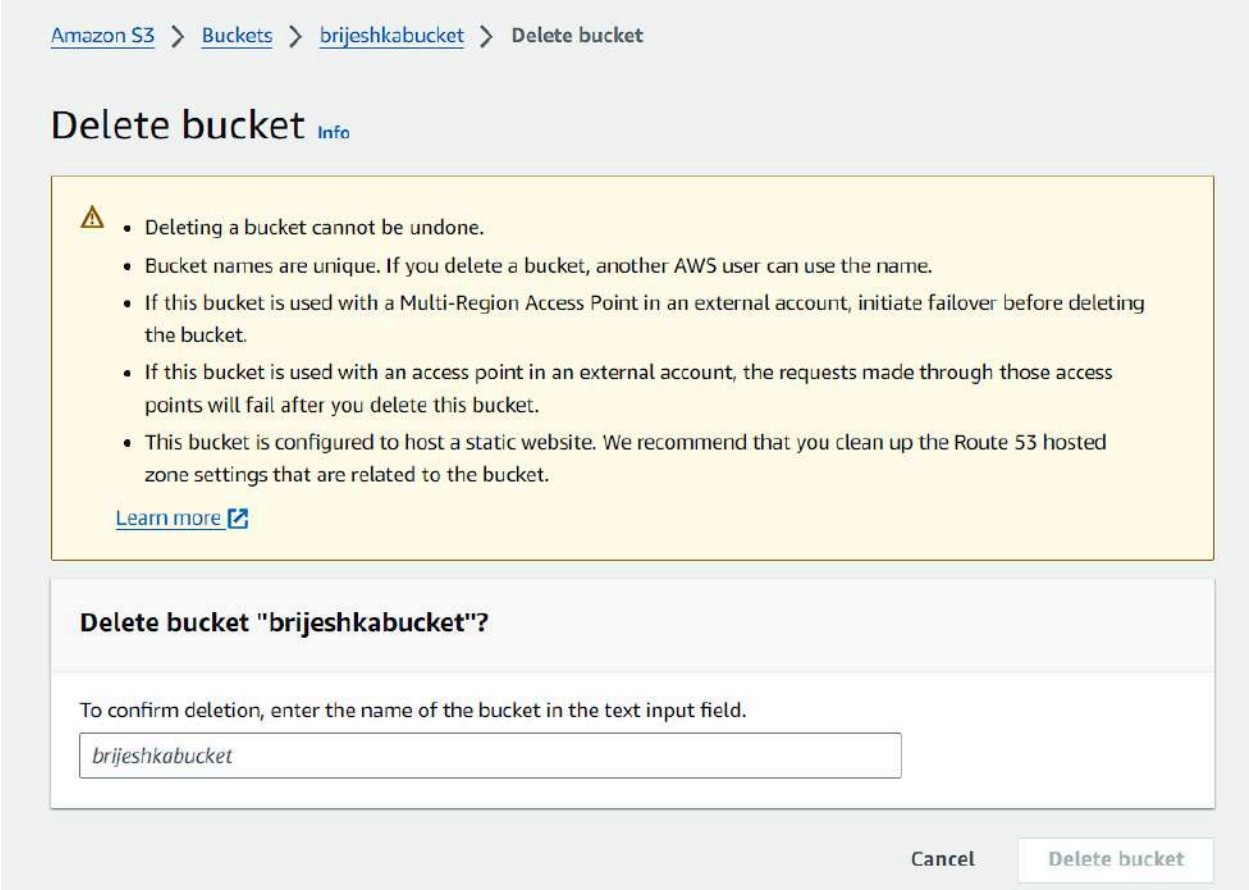
17) Get the link for the hosted website in the properties tab at the bottom.

18) The hosted website using AWS S3.

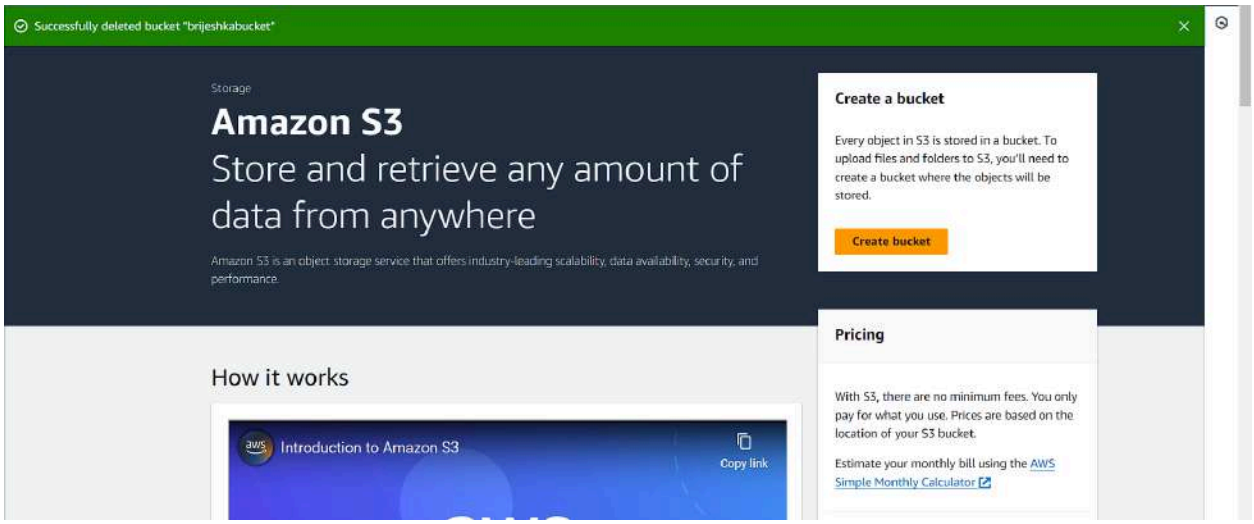
19) To terminate the S3 bucket, first empty the bucket by selecting the files and clicking on Empty.



20) Then navigate to the Delete bucket option and enter the name of the bucket and delete the bucket.



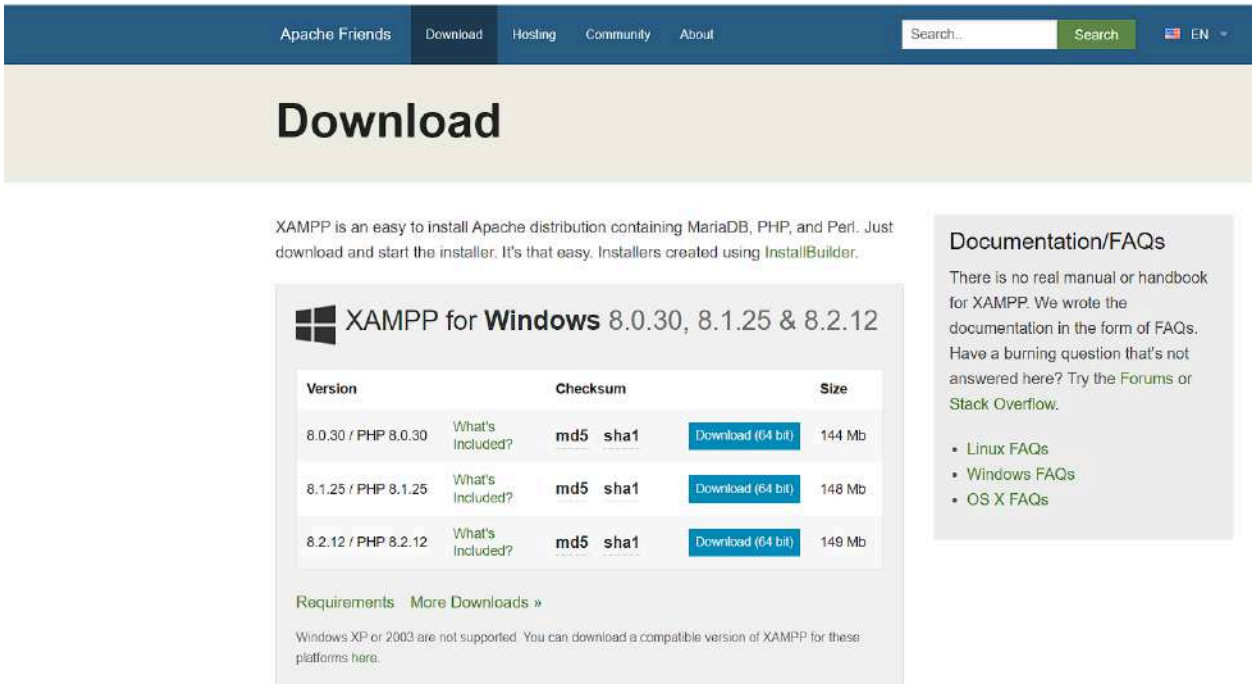
21) After deleting the bucket, a message will appear.



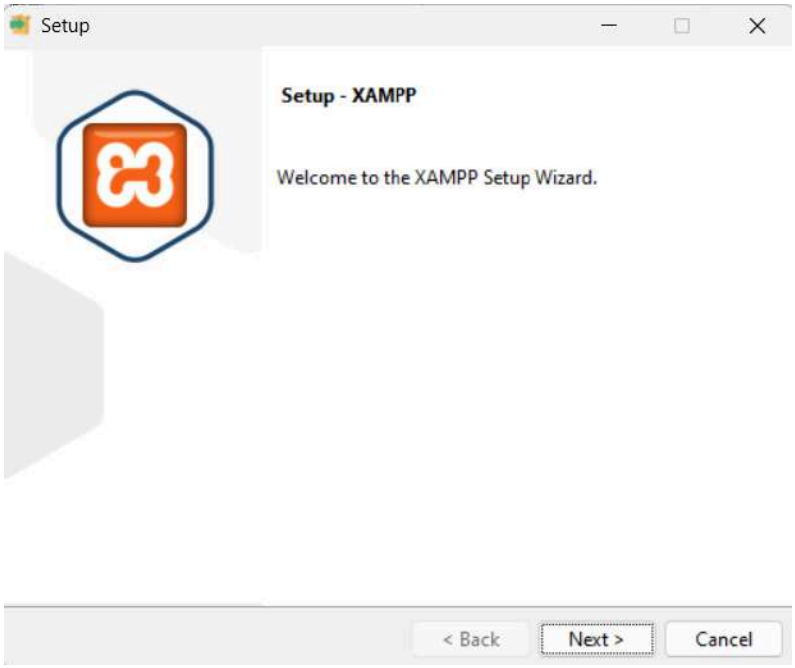
XAMPP Hosting

Name:Krushikesh Shelar D15-C Roll No:51

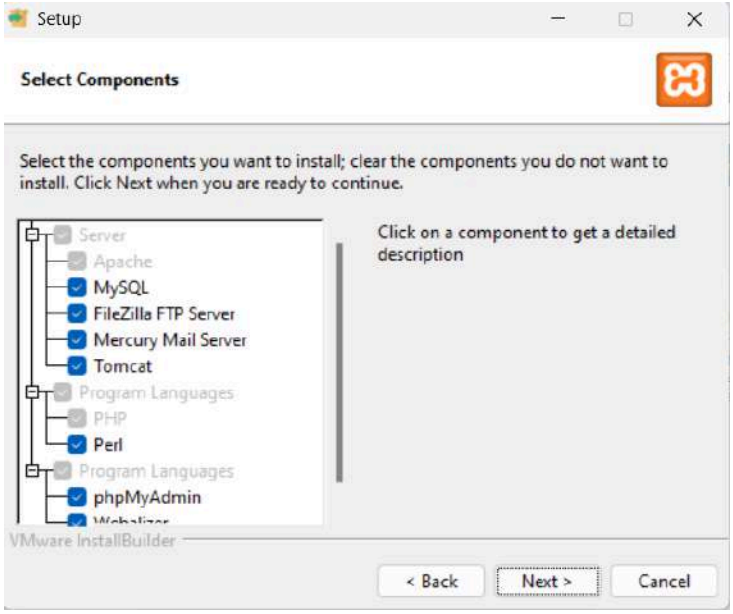
1) Search for XAMPP download and navigate to the xampp official website and click on download as per your system.



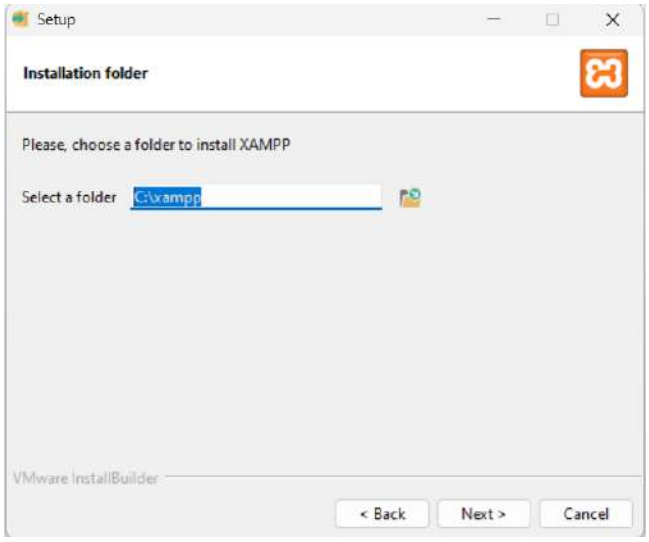
2) Xampp installer will be installed in the system.
3) Window asking to give permission will appear. Click pn 'Yes'.
4) A window will appear for setup. Click on 'Next'.



5) Keep the default settings and click on Next.



6) Choose the folder path.



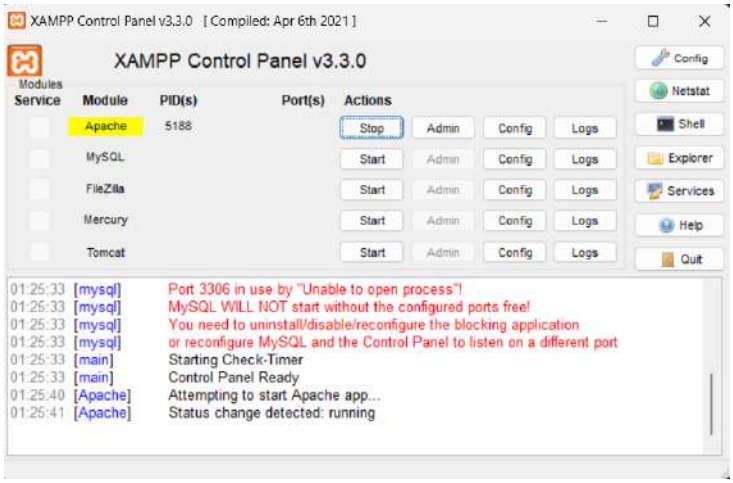
7) Then the XAMPP installation will be done.
8) Locate the folder and then locate the ‘htdocs’ folder in the xampp folder.

9) Create a test.php file in the htdocs folder and write php code.

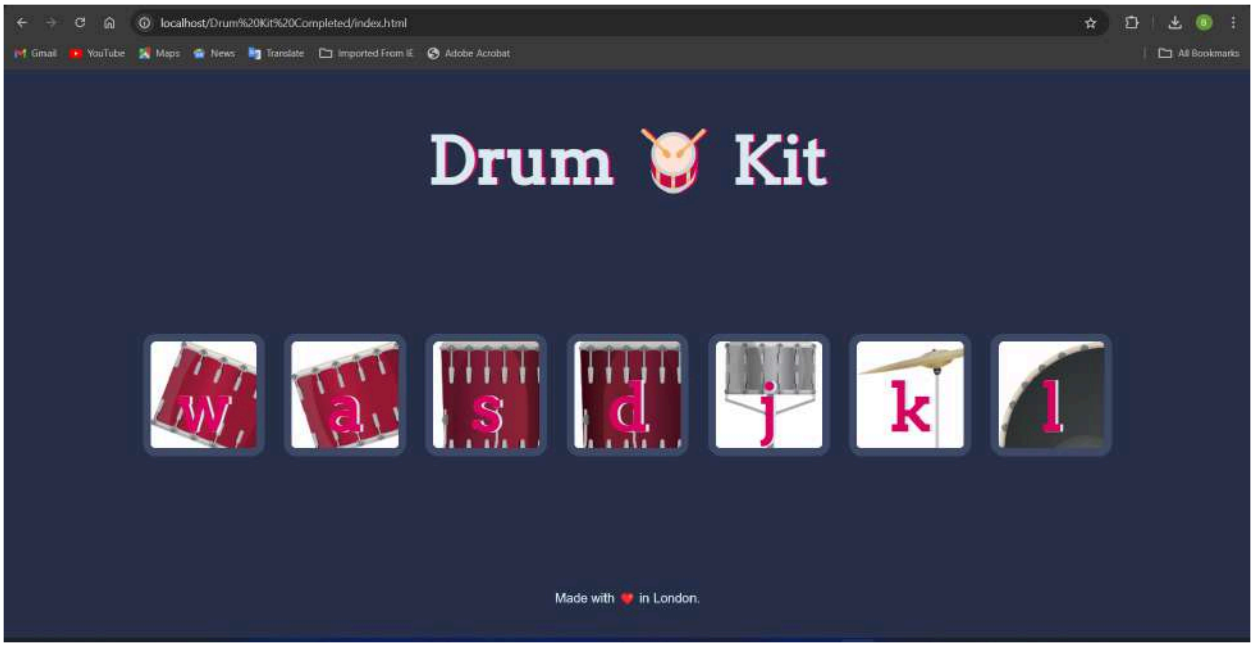
10) Now go to the xampp contol panel in the xampp folder.

xampp_start	30-03-2013 17:59	Application	116 KB
xampp_stop	30-03-2013 17:59	Application	116 KB
xampp-control	06-04-2021 17:08	Application	3,290 KB
xampp-control	01-08-2024 21:45	Configuration setti...	1 KB

11) Start the Apache server.



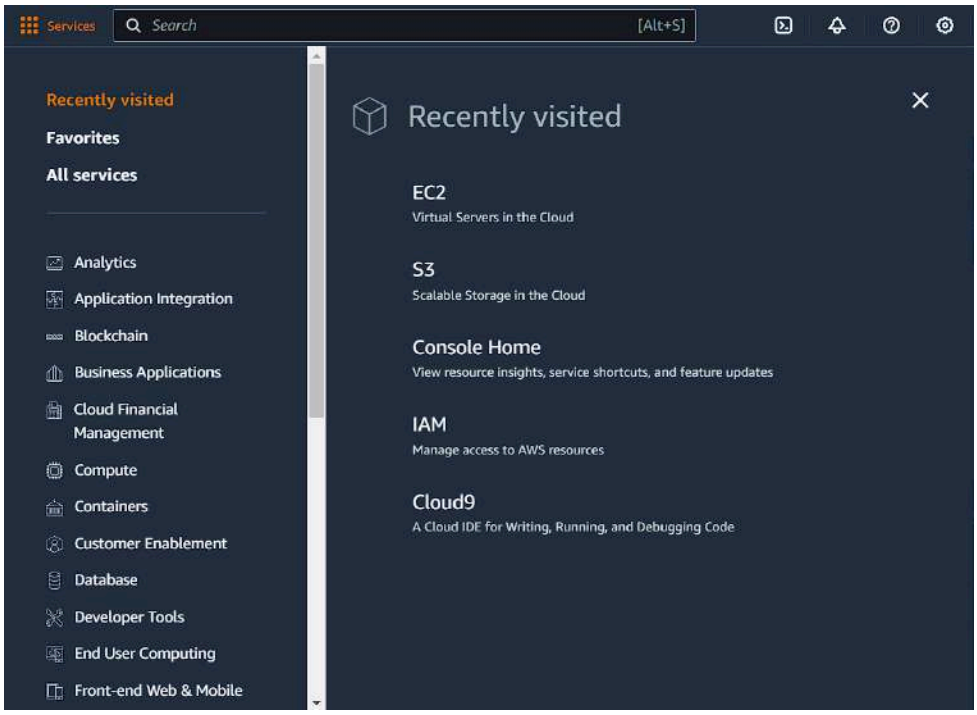
12) After strating the service, got to “localhost/file_name”, then the output window will appear.



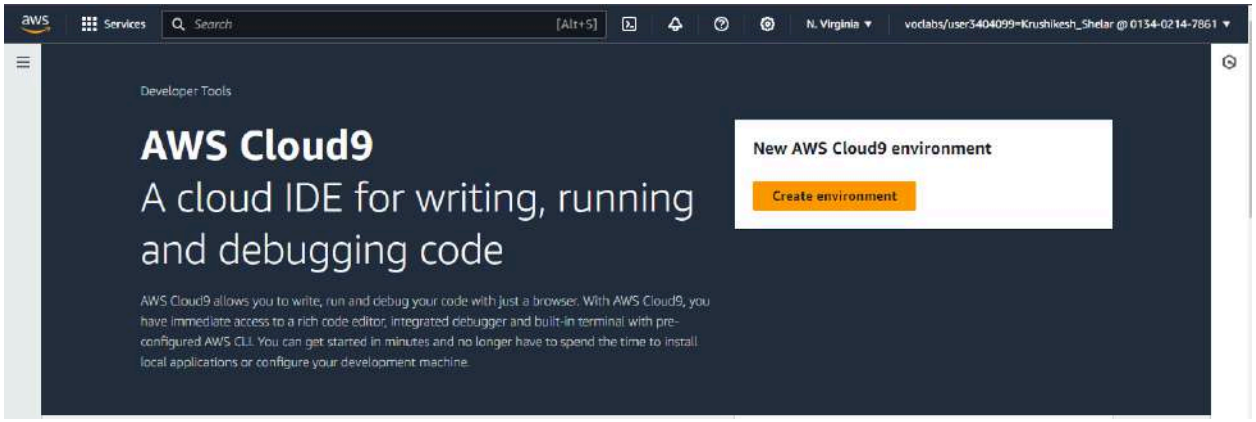
Name:Krushikesh Shelar D15-C Roll No:51

AIM: Setup Cloud 9 - IDE and perform collaboration demonstration.

1) Navigate to developer tools -> Cloud9 and start creating Cloud9 environment.

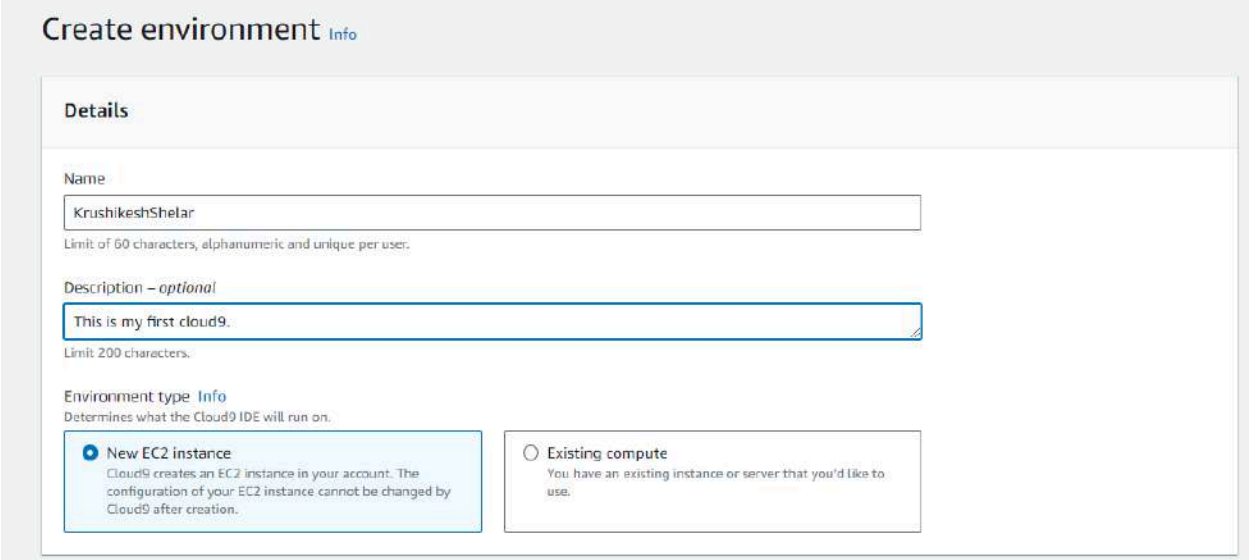


2) Click on Create Enviornment and start creating the environment

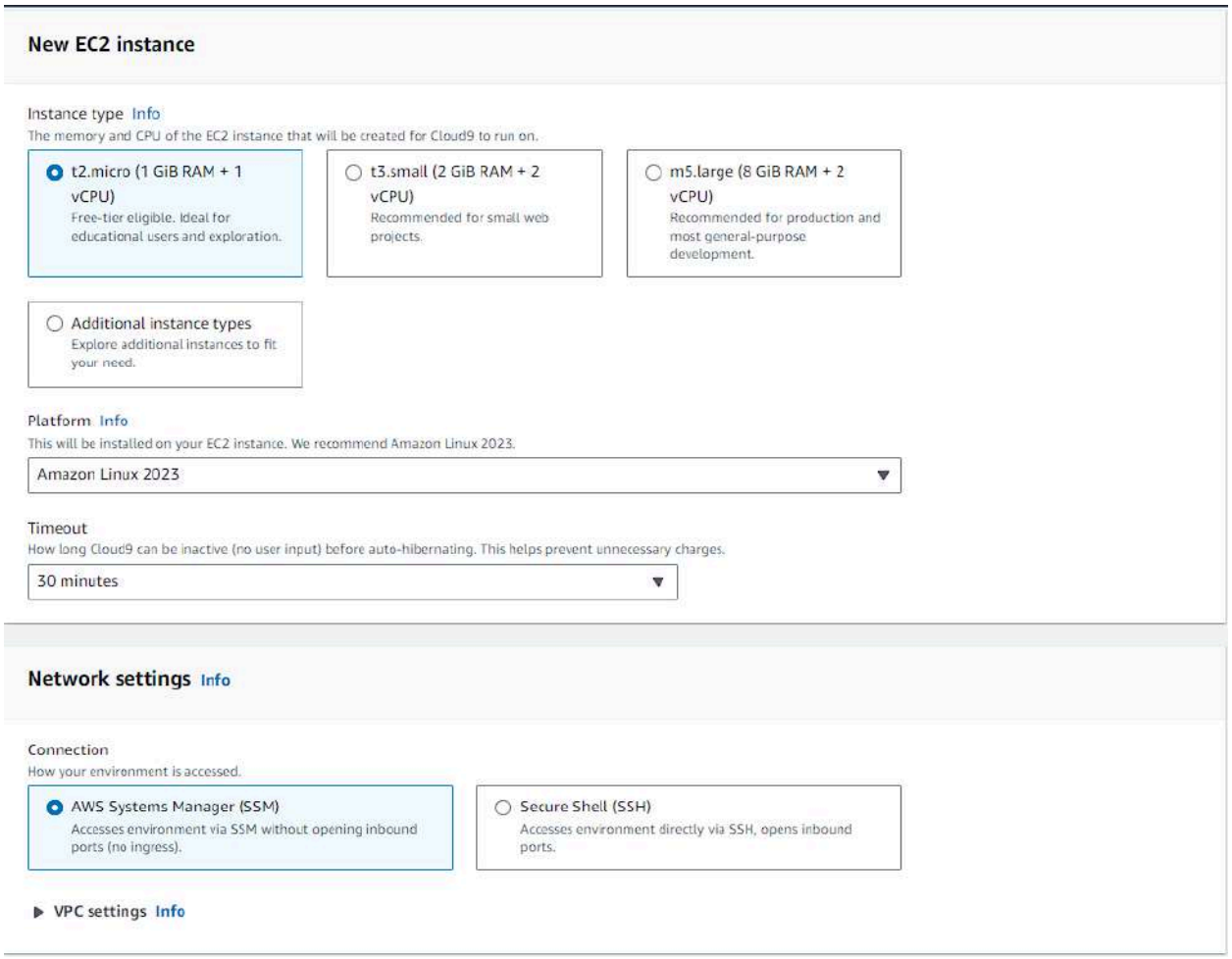


Name:Krushikesh Shelar D15-C Roll No:51

3) Name the environment and select new EC2 instance.

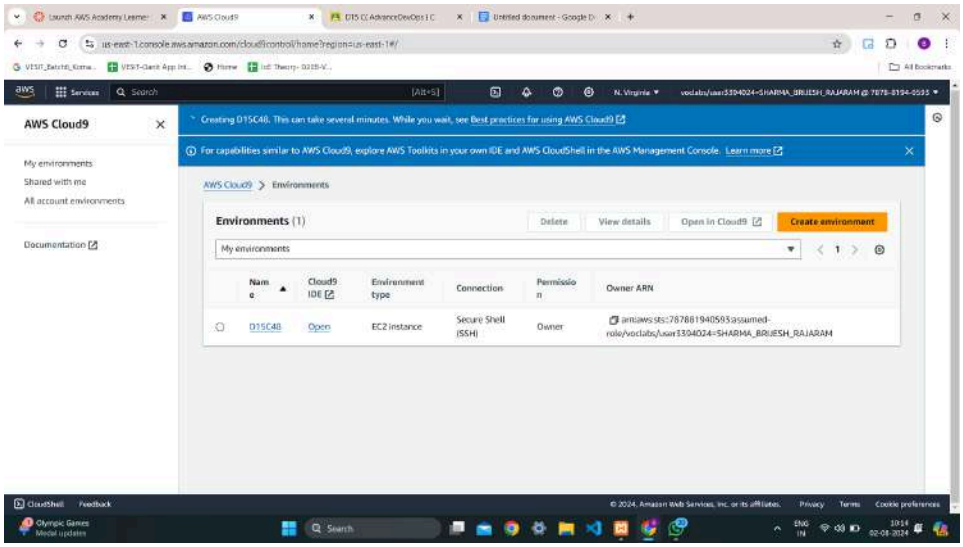


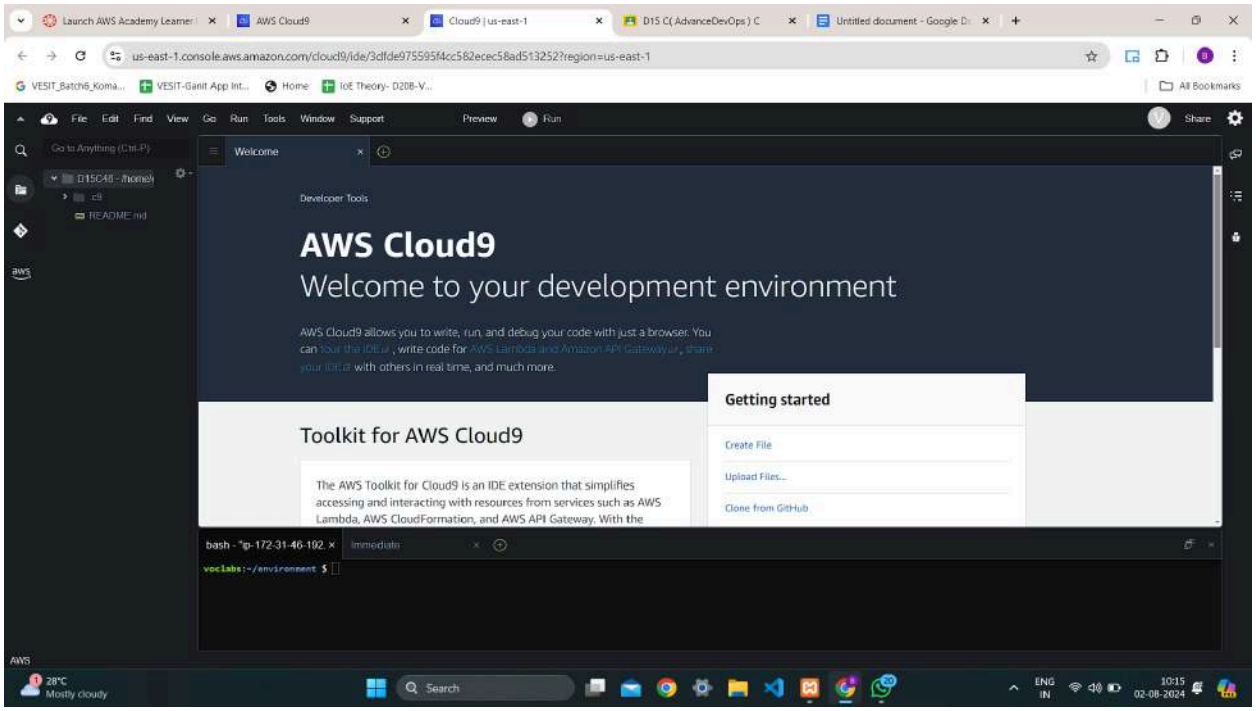
4) Keep the options default and proceed



Name:Krushikesh Shelar D15-C Roll No:51

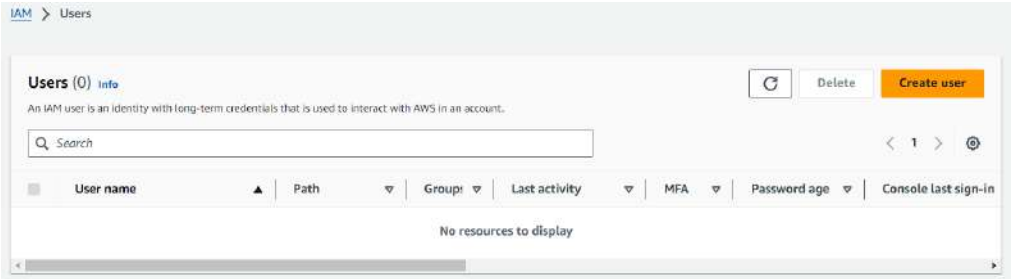
5) Environment created successfully.



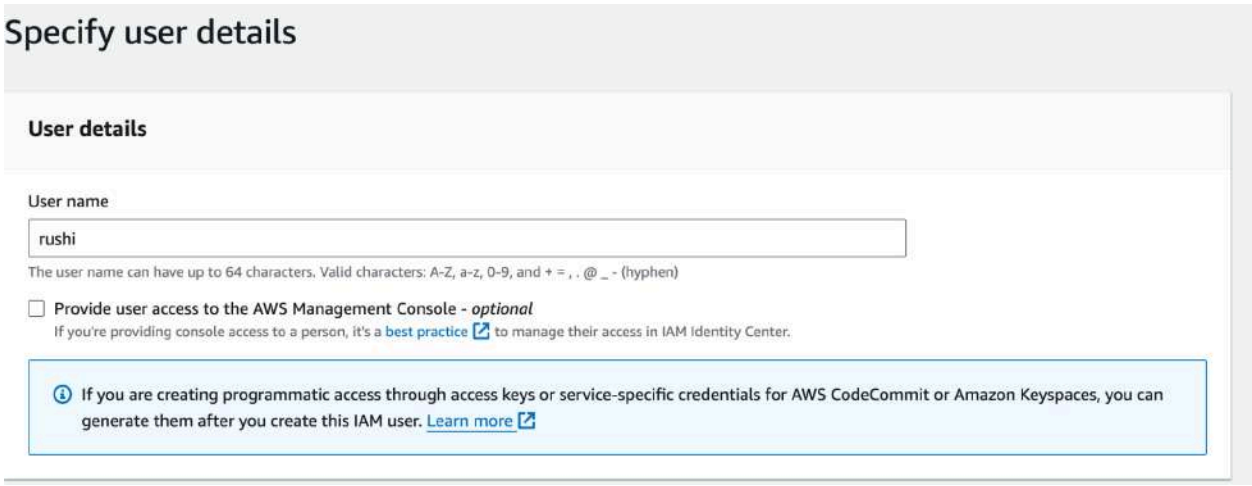


Name:Krushikesh Shelar D15-C Roll No:51

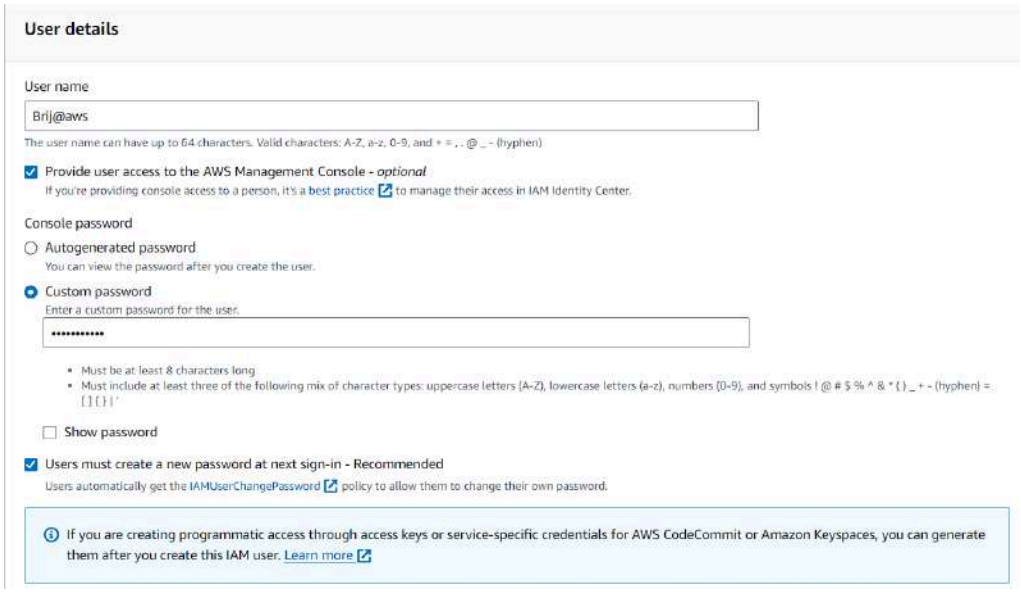
6) Create user using the IAM.



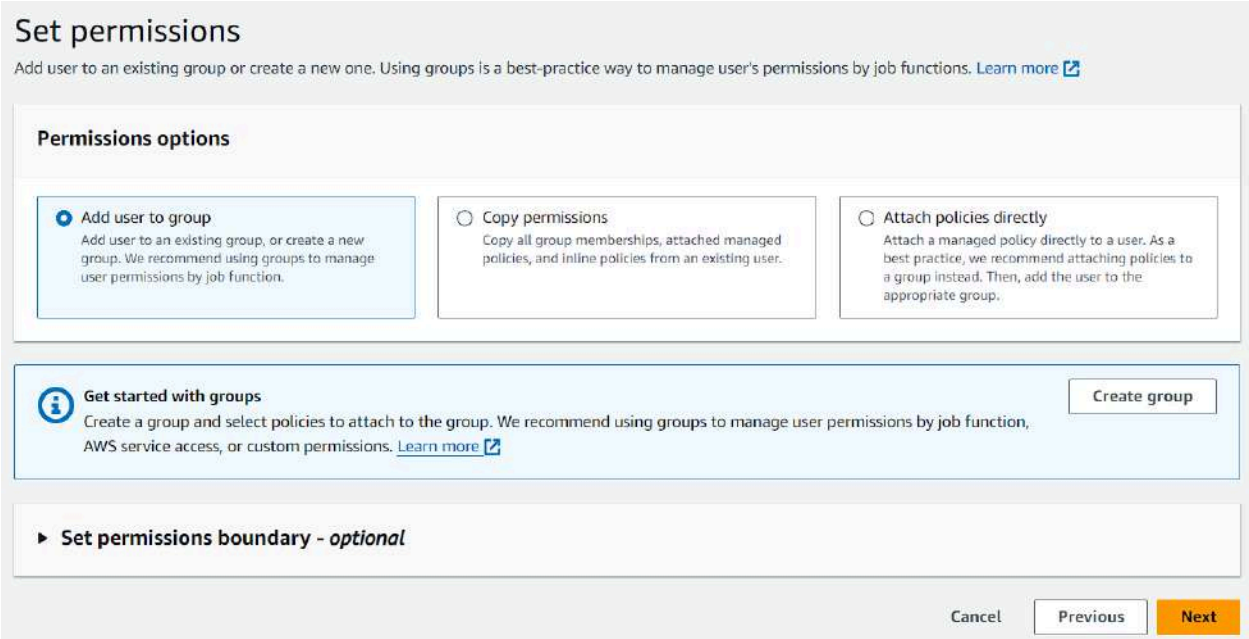
7) Add the username



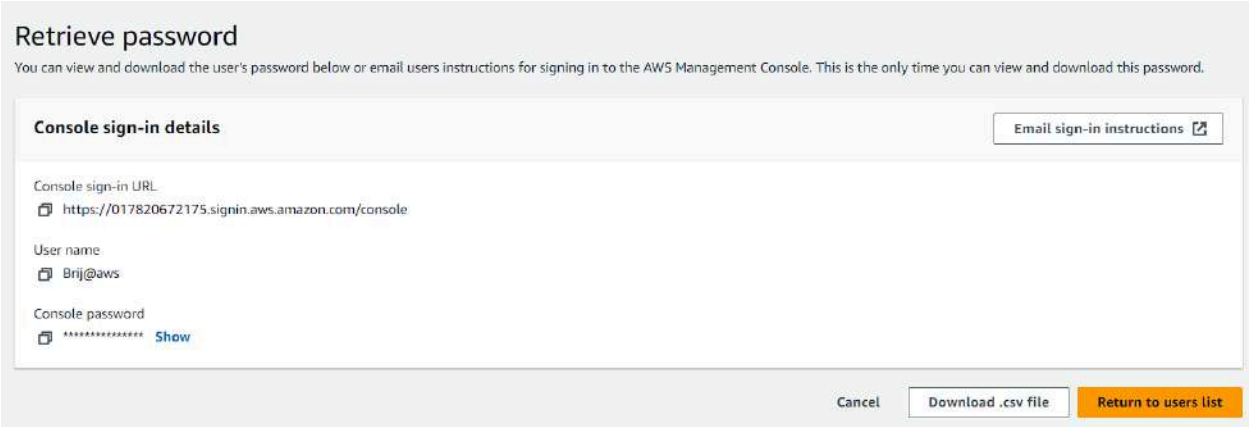
8) Add the remaining user details and provide access to the AWS Management Console



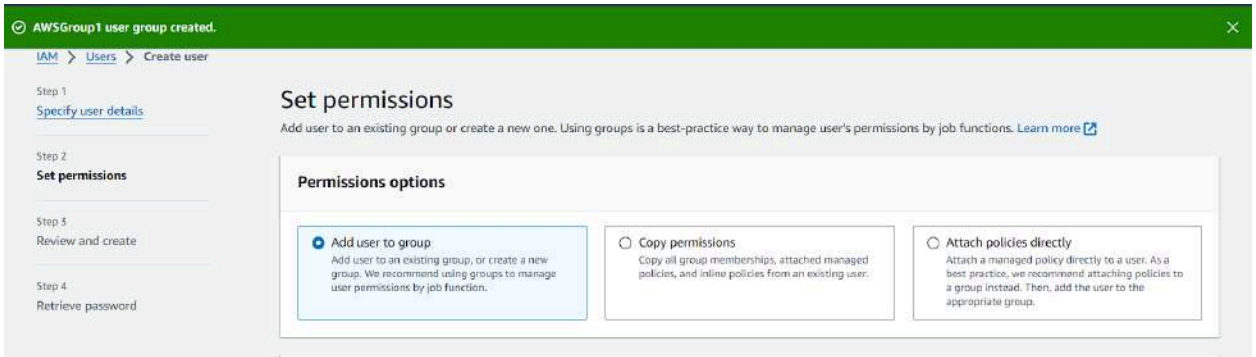
9) User created successfully and can be added to user groups.



10) User credentials can be downloaded.

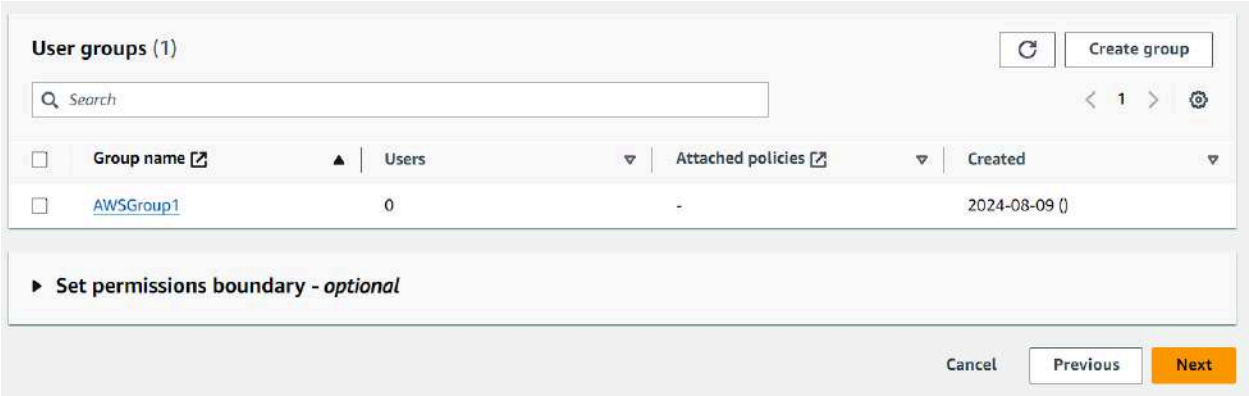


11) Add user to group if group exists else create a new group.

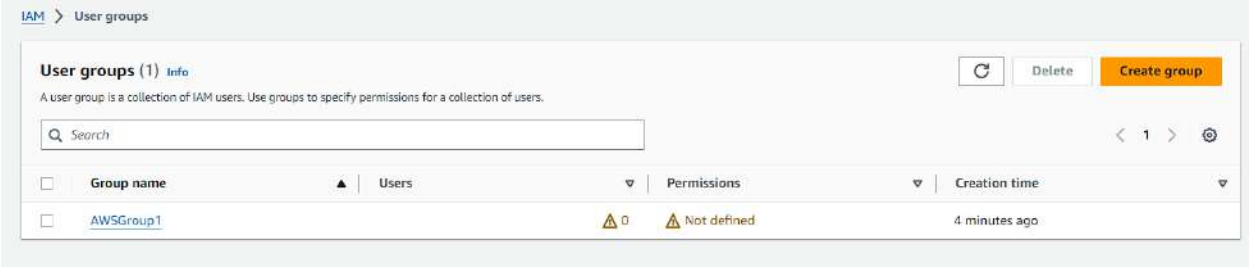


Name:Krushikesh Shelar D15-C Roll No:51

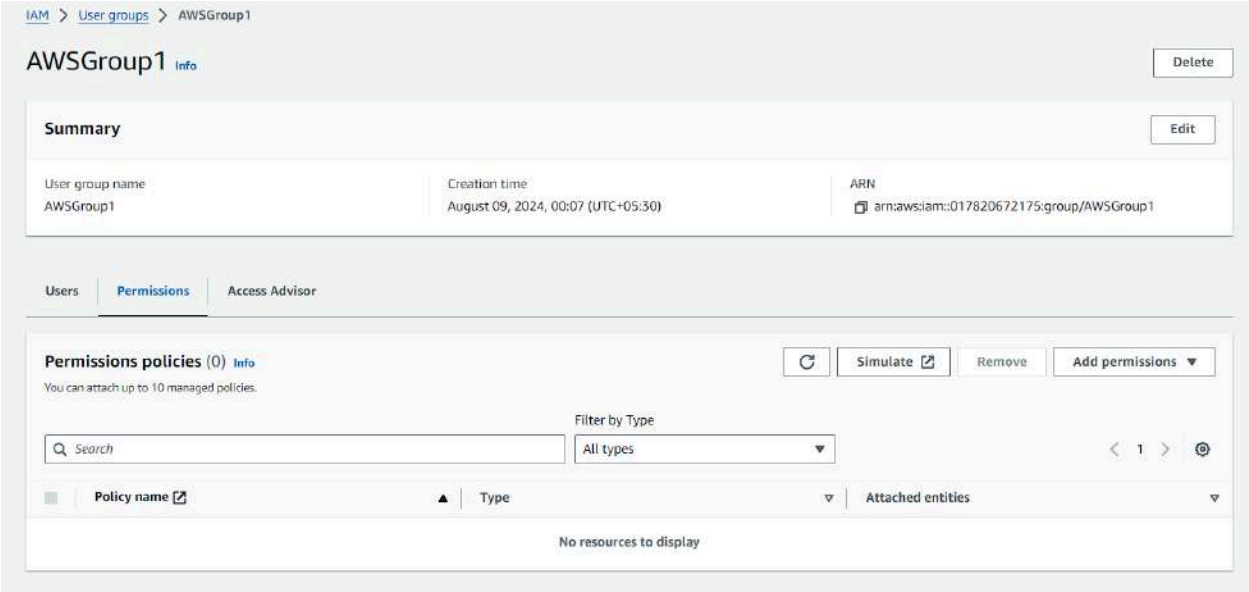
12) Write the user group name and proceed.



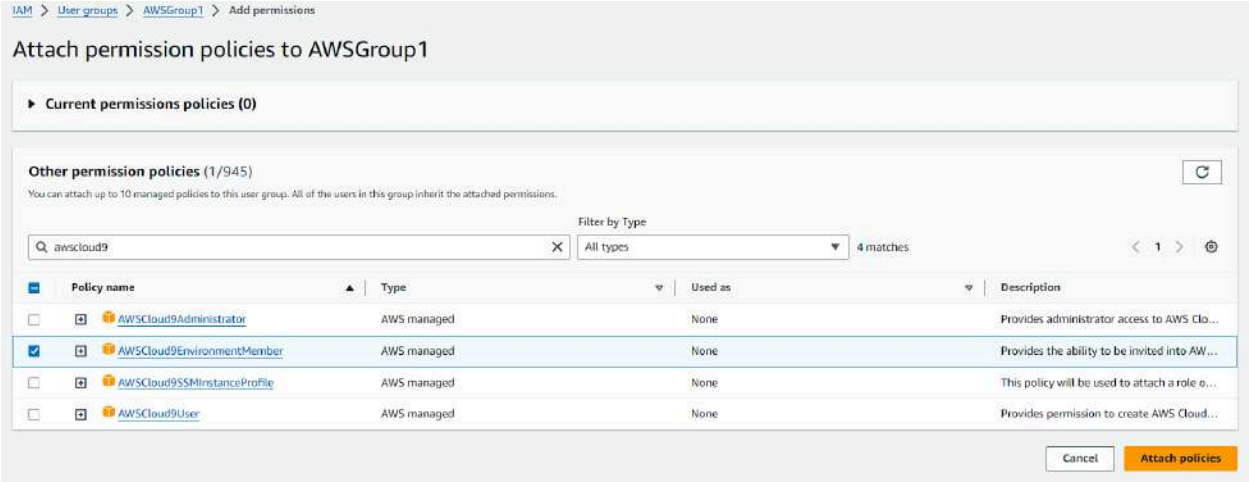
13) Click on *group_name.*



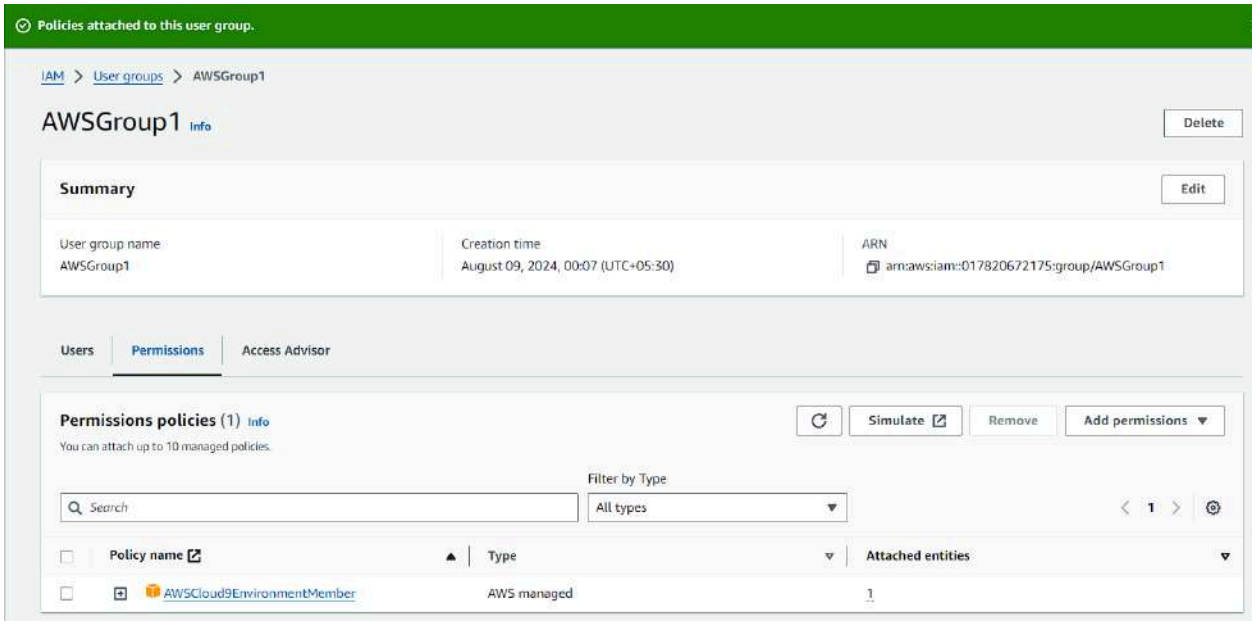
14) Go to Add permissions and click on Add Permissions



15) On attach policies, select AWSCloud9EnvironmentMember and click on Attach policies.

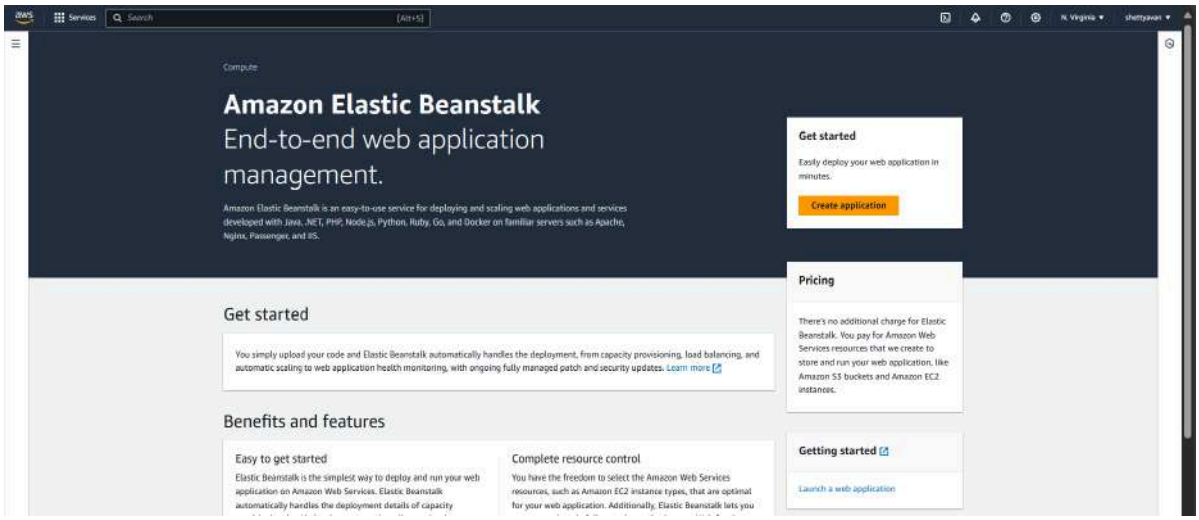


16) User group is created successfully.

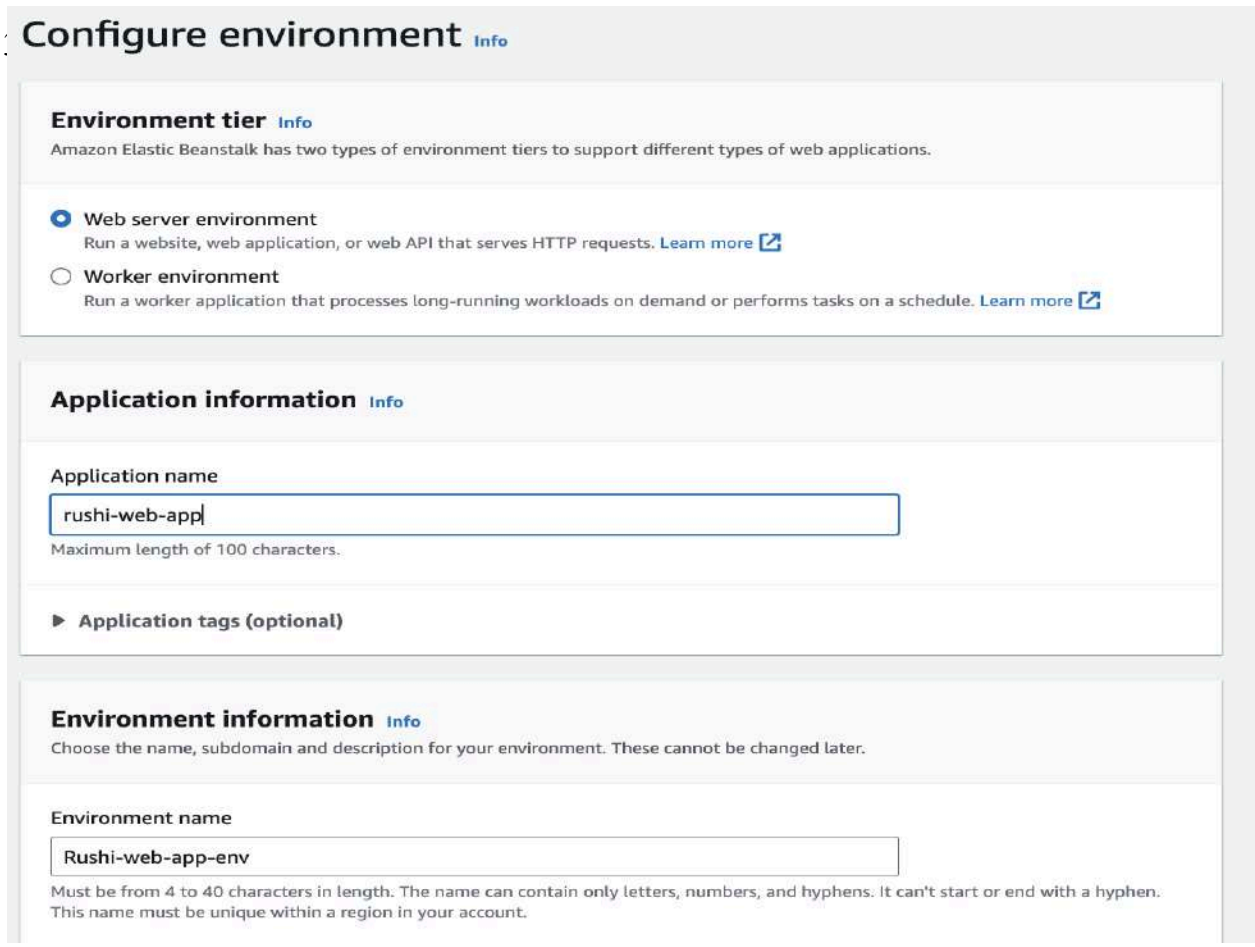


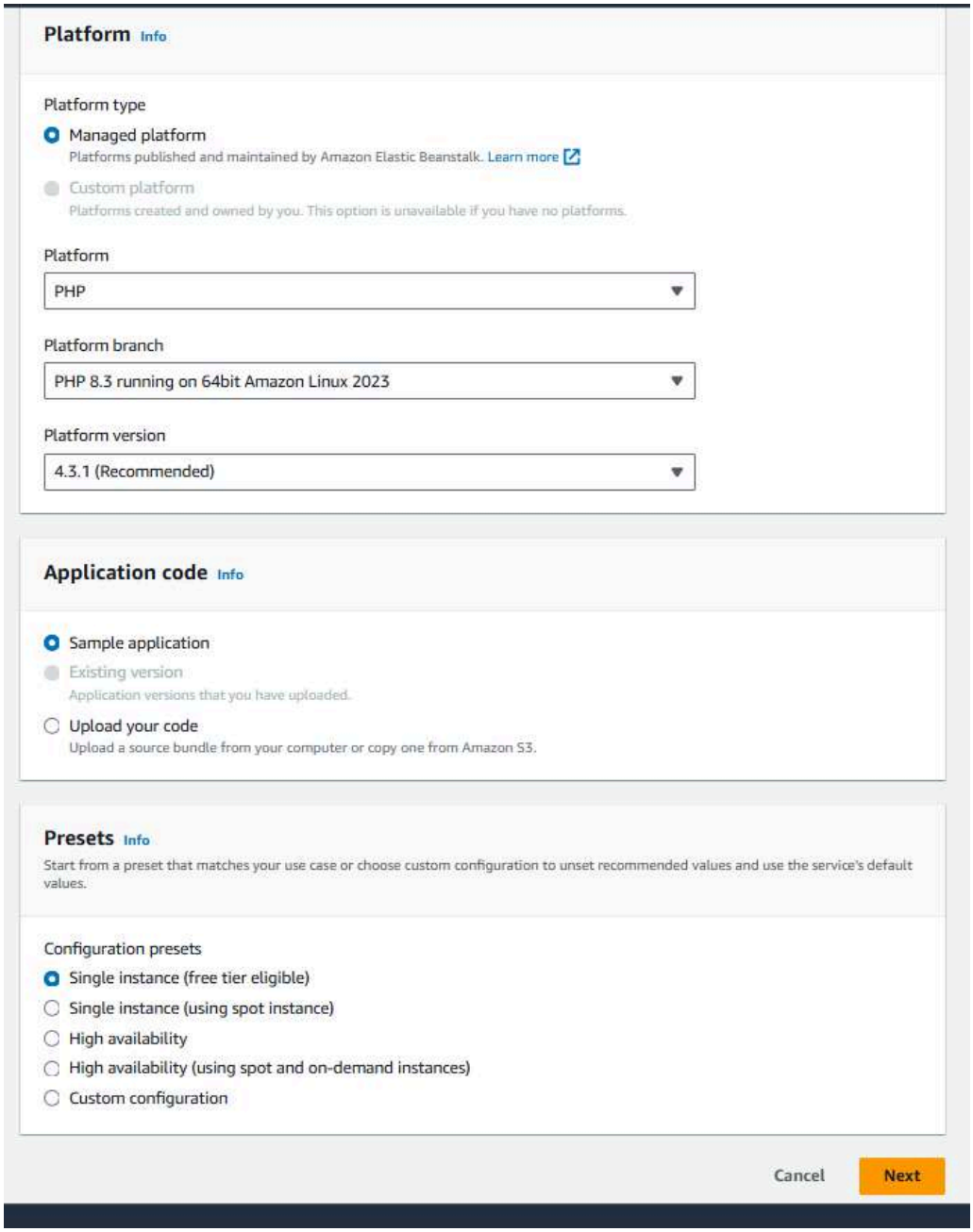
Aim : To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.

1. Open the aws console and then search Elastic Beanstalk (Opens a dashboard as seen below)

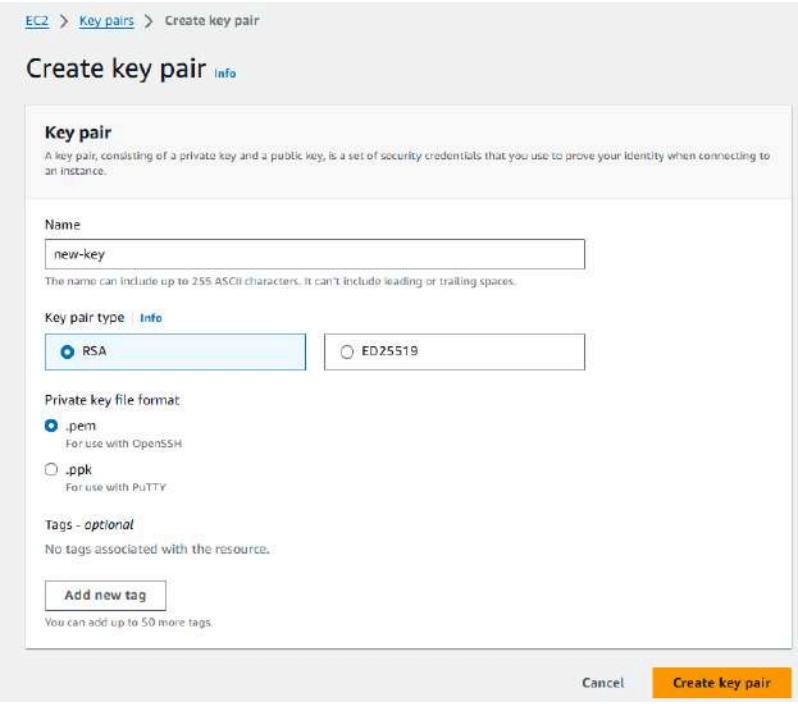


2. Click on create application and configure the environment by adding your application name

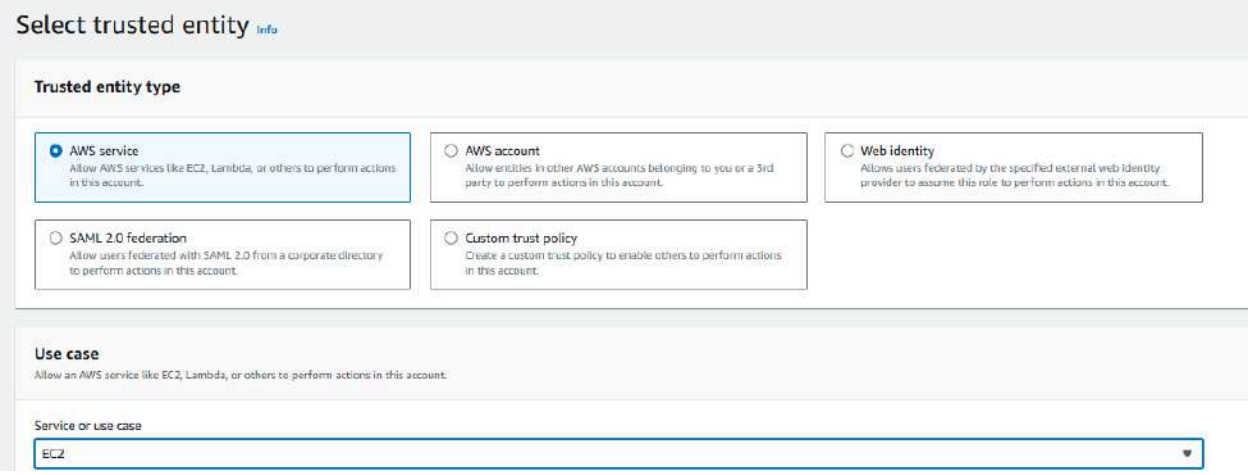




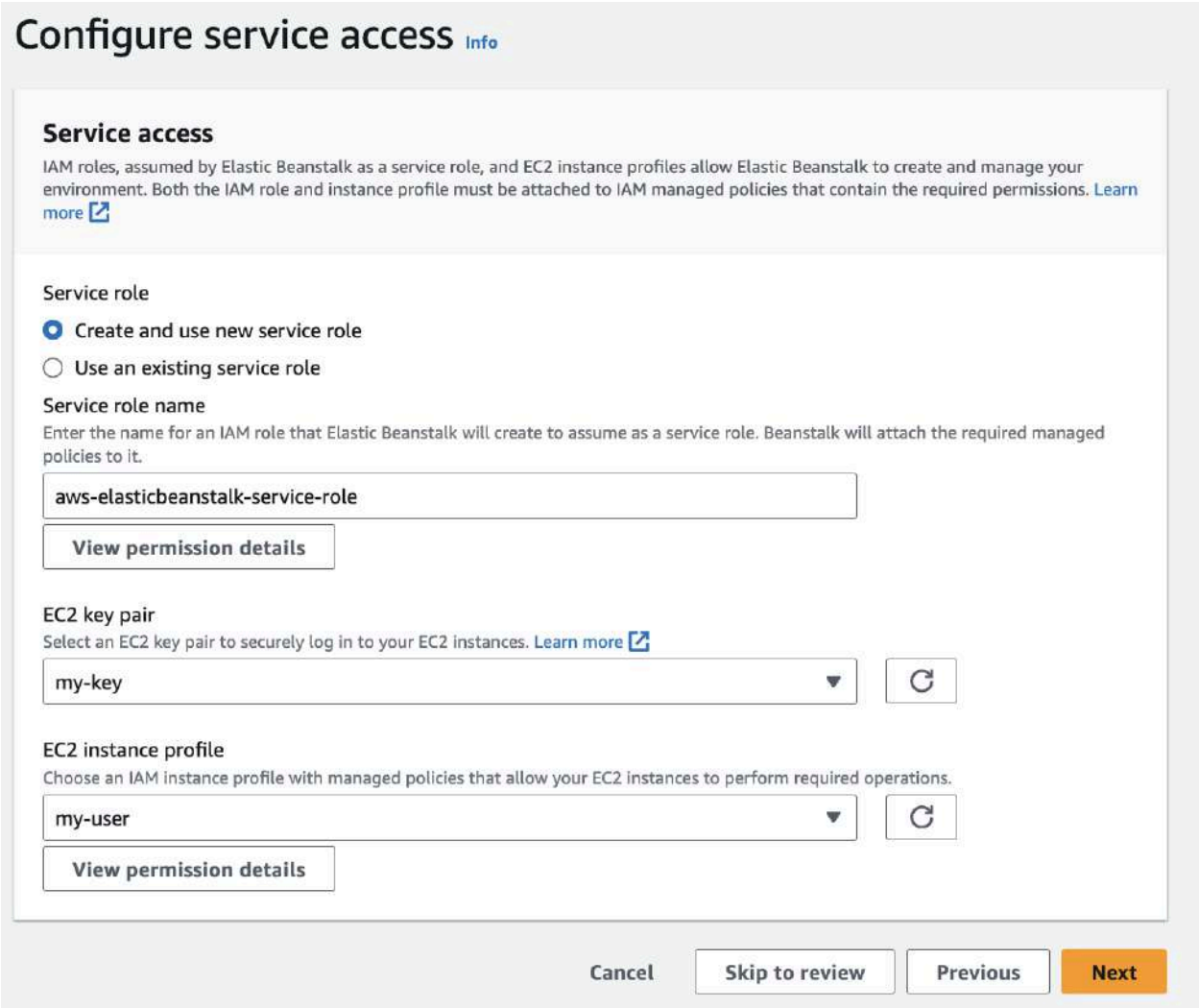
4. In this step you have to create a key pair
   a. Go to EC2 instance tab and from the left panel create a key pair. As the key pair might be useful for the further process



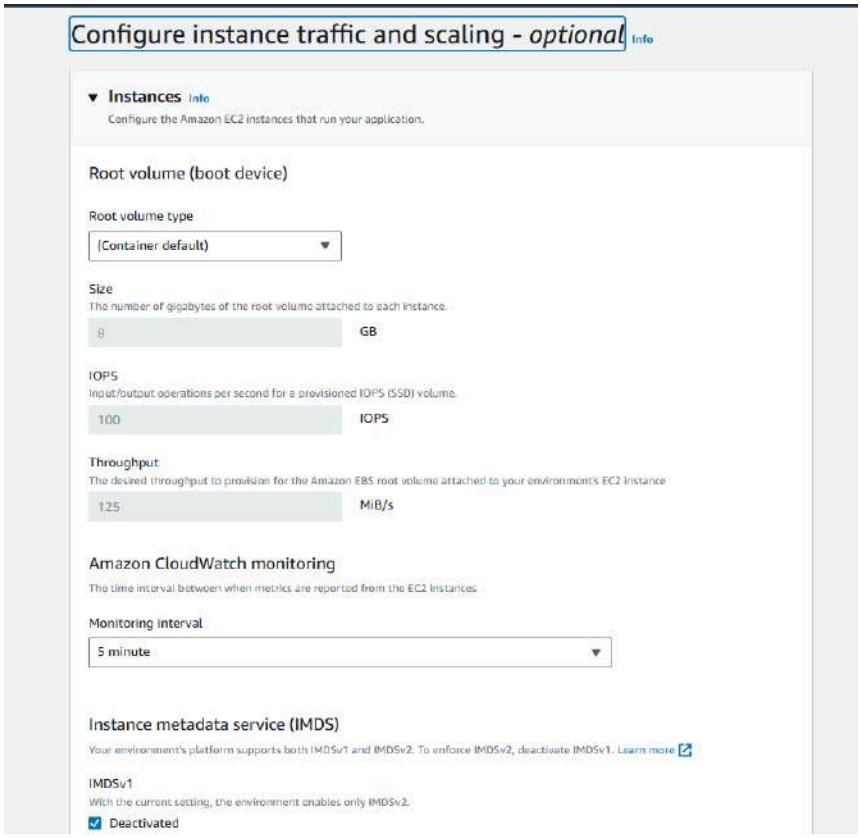
b. In the same fashion go to IAM and then under roles section click create role and then Select AWS Service and under instances select EC2. Your EC2 Instance role is created

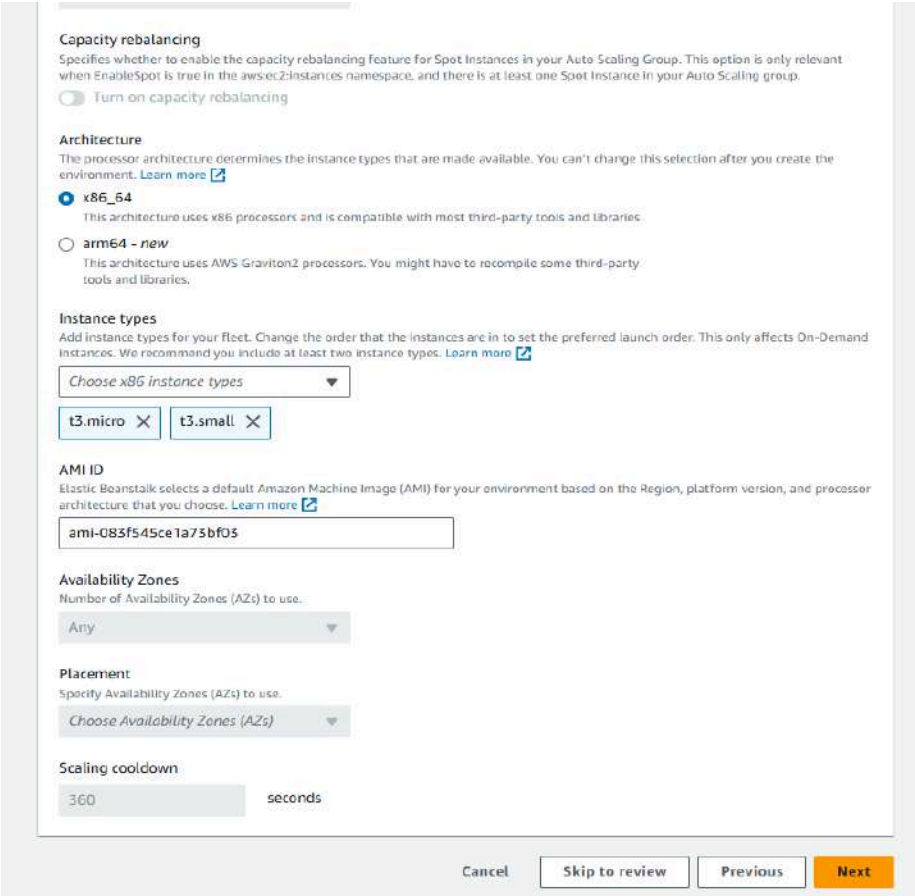


5. Now comeback to Elastic Beanstalk page and from the drop down menu select the newly created key pair and instance profile



6. No changes further in the Configure instance traffic section


Configure instance traffic and scaling - optional Info
Instances Info
Configure the Amazon EC2 instances that run your application.
Root volume (boot device)
Root volume type
(Container default)
Size
The number of gigabytes of the root volume attached to each instance.
8
GB
IOPS
Input/output operations per second for a provisioned IOPS (SSD) volume.
100
IOPS
Throughput
The desired throughput to provision for the Amazon EBS root volume attached to your environment's EC2 instance
125
MiB/s
Amazon CloudWatch monitoring
The time interval between when metrics are reported from the EC2 instances
Monitoring interval
5 minute
Instance metadata service (IMDS)
Your environment's platform supports both IMDSv1 and IMDSv2. To enforce IMDSv2, deactivate IMDSv1. Learn more
IMDSv1
With the current setting, the environment enables only IMDSv2.
Deactivated



7. Again no changes in the configure updates, monitoring, and logging part just click on next.

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

CloudWatch Custom Metrics - Instance

Choose metrics

CloudWatch Custom Metrics - Environment

Choose metrics

Health event streaming to CloudWatch Logs

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

Log streaming

☐ Activated (standard CloudWatch charges apply.)

Retention

7

Lifecycle

Keep logs after terminating environment

Managed platform updates Info

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

Amazon X-Ray

Amazon X-Ray is a service that collects data about the requests and responses that your application serves and receives. You can use the tools that X-Ray offers to view and filter the data that it provides to identify potential issues and optimization opportunities.

X-Ray daemon

(service charges may apply.)

☐ Activated

S3 log storage

Configure the instances in your environment to upload rotated logs to Amazon S3. Learn more

Rotate logs

(standard S3 charges apply.)

☐ Activated

Instance log streaming to CloudWatch logs

Configure the instances in your environment to stream logs to CloudWatch logs. You can set the retention to up to 10 years and configure Elastic Beanstalk to delete the logs when you terminate your environment. Learn more

Log streaming

(standard CloudWatch charges apply.)

☐ Activated

Retention

7

Lifecycle

Keep logs after terminating envir...

Environment properties

The following properties are passed in the application as environment properties. Learn more

No environment properties have been configured.

Add environment property

Cancel Previous Next

8. Now review the changes made and click on create application

Review Info

Step 1: Configure environment

Edit

Environment information

Environment tier
Web server environment

Application name
rushi-web-app

Environment name
Rushi-web-app-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

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

Service role
arn:aws:iam::017820665164:role/service-role/aws-elasticbeanstalk-service-role

EC2 key pair
my-key

EC2 instance profile
my-user

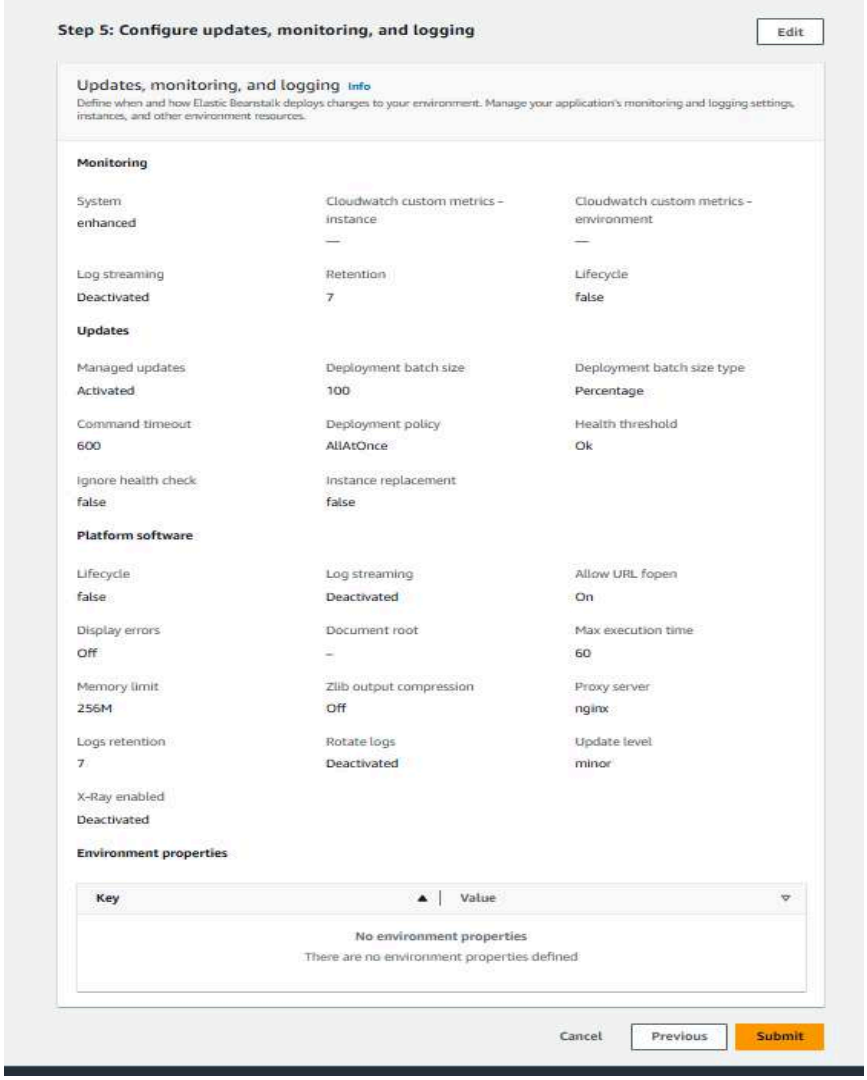
Step 3: Set up networking, database, and tags

Edit

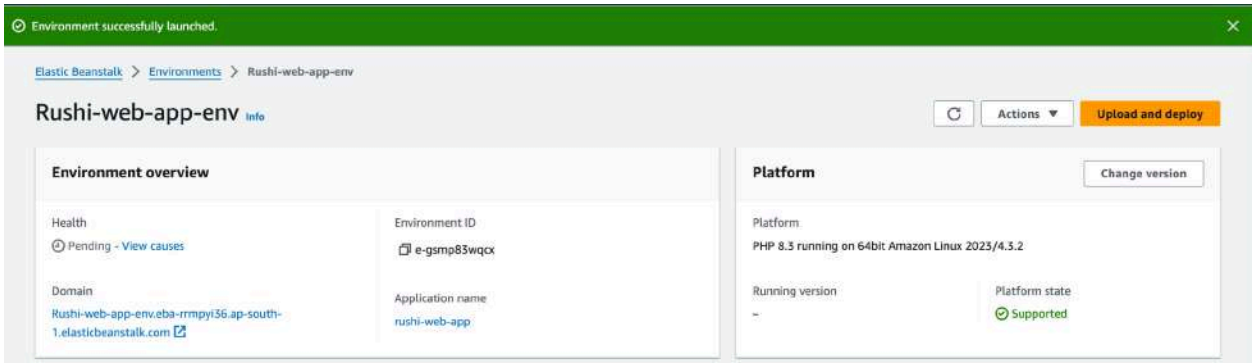
Networking, database, and tags Info

Configure VPC settings, and subnets for your environment's EC2 instances and load balancer. Set up an Amazon RDS database that's integrated with your environment.

No options configured



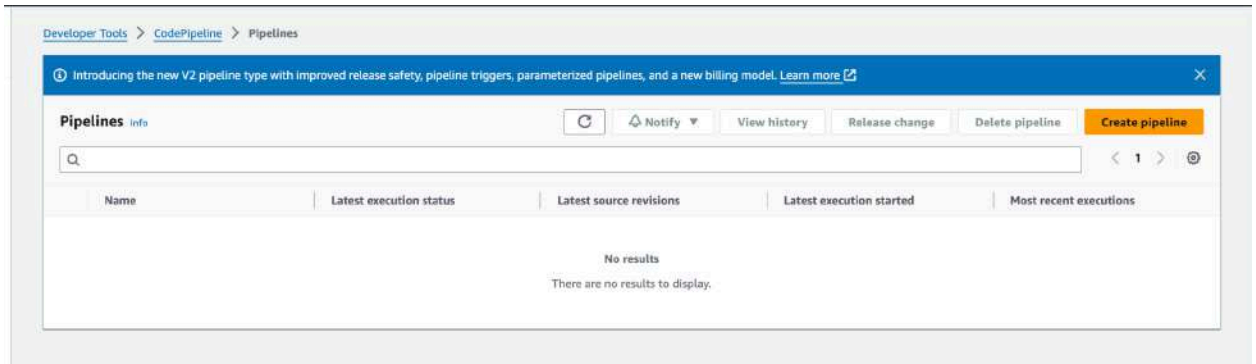
9. Your sample environment is created for you to deploy your application. By default, it creates an EC2 instance, a security group, an Auto Scaling group, an Amazon S3 Bucket, Amazon CloudWatch alarms and a domain name for your Application.



Pipeline creation:

1. Fork a github repo for aws codepipeline available as The pipeline takes code from the source and then performs actions on it.We don't need to code from scratch in this manner

2. Go to developer tools and select CodePipeline and create a new pipeline



3. Name your pipeline and select the desired service role

Pipeline settings

Pipeline name
Enter the pipeline name. You cannot edit the pipeline name after it is created.

my-pipeline

No more than 100 characters

Pipeline type

You can no longer create V1 pipelines through the console. We recommend you use the V2 pipeline type with improved release safety, pipeline triggers, parameterized pipelines, and a new billing model.

Execution mode
Choose the execution mode for your pipeline. This determines how the pipeline is run.

- Superseded
  A more recent execution can overtake an older one. This is the default.
- Queued (Pipeline type V2 required)
  Executions are processed one by one in the order that they are queued.
- Parallel (Pipeline type V2 required)
  Executions don't wait for other runs to complete before starting or finishing.

Service role

- New service role
  Create a service role in your account
- Existing service role
  Choose an existing service role from your account

Role name

AWSCodePipelineServiceRole-ap-south-1-my-pipeline

Type your service role name

☑ Allow AWS CodePipeline to create a service role so it can be used with this new pipeline

Variables

You can add variables at the pipeline level. You can choose to assign the value when you start the pipeline. Choosing this option requires pipeline type V2. Learn more

No variables defined at the pipeline level in this pipeline.

Add variable

You can add up to 50 variables.

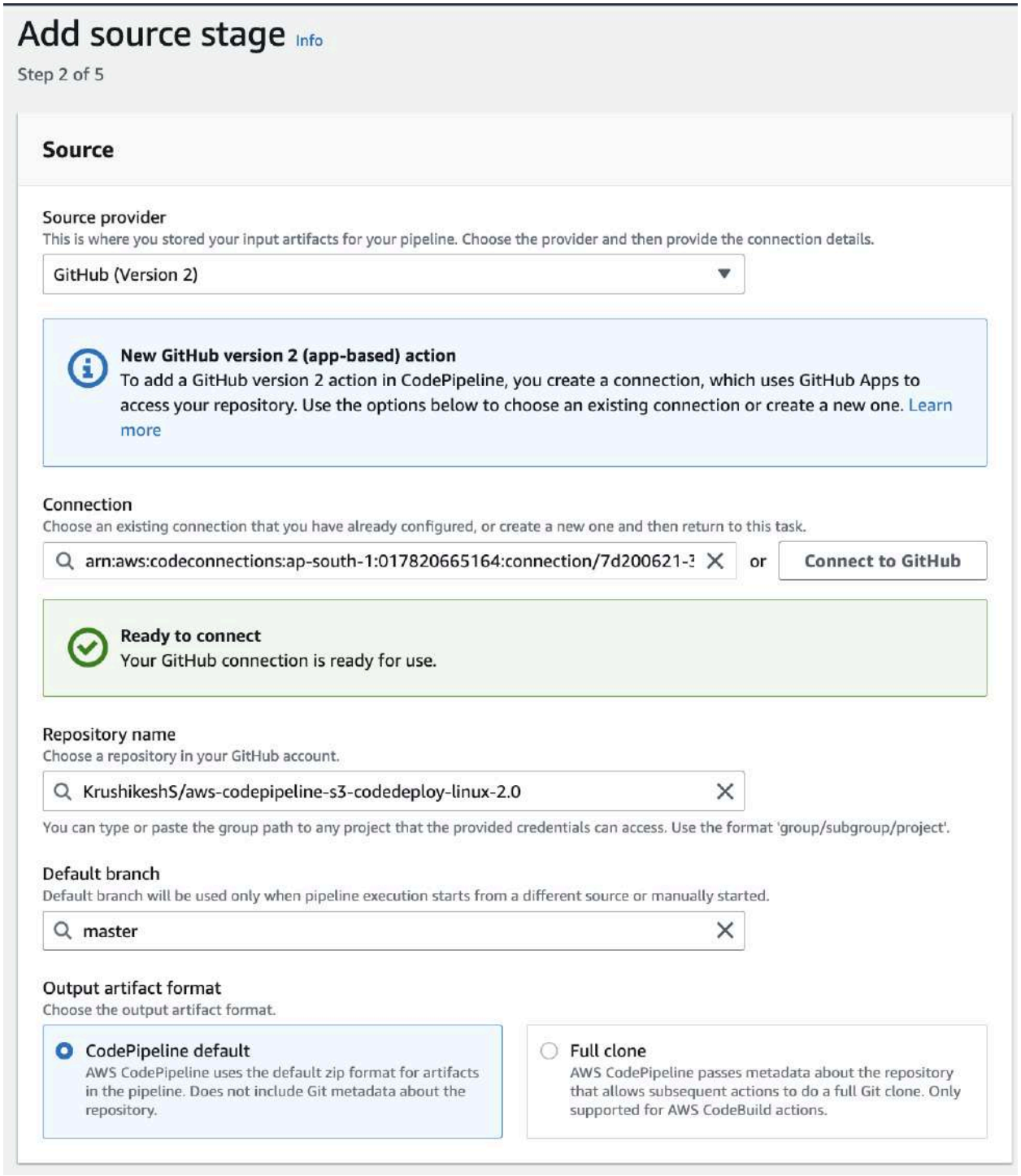
The first pipeline execution will fail if variables have no default values.

▸ Advanced settings

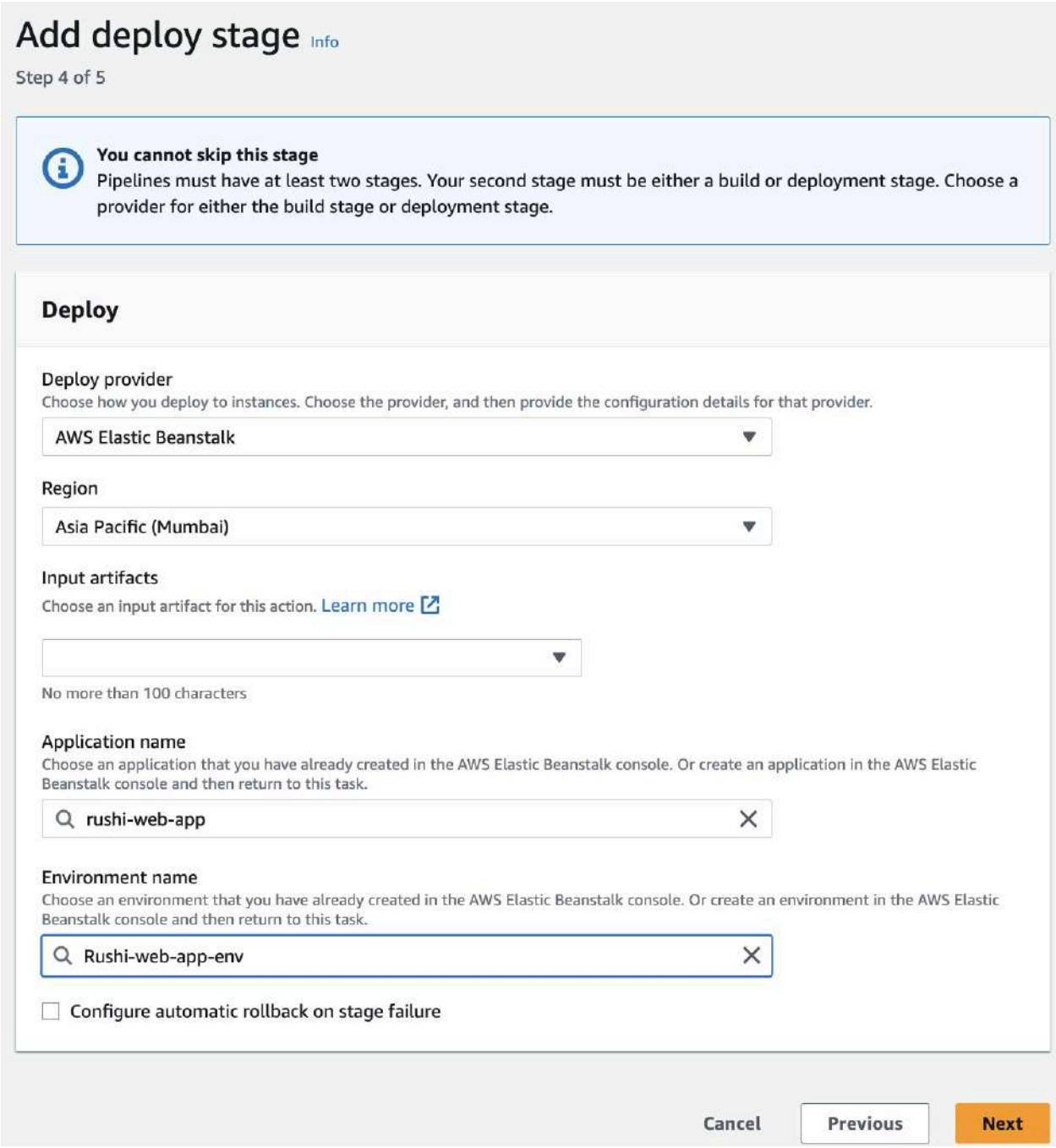
4. In the source stage select Github v2 as the provider and then connect your github connect so that the pipeline can access the forked source code .

a. For this purpose create aws github connection and with your credentials install the AWS under the forked repository

5. Once the connection is established from the drop down menu select the repository and the branch



6. Skip the build stage part as we are not plugging in any build provider and in choose Beanstalk as the Deploy Provider, same region as the Bucket and Beanstalk, name and environment name.



7. Review the settings and click on create pipeline

Review Info

Step 5 of 5

Step 1: Choose pipeline settings

Pipeline settings

Pipeline name

my-pipeline

Pipeline type

V2

Execution mode

QUEUED

Artifact location

A new Amazon S3 bucket will be created as the default artifact store for your pipeline

Service role name

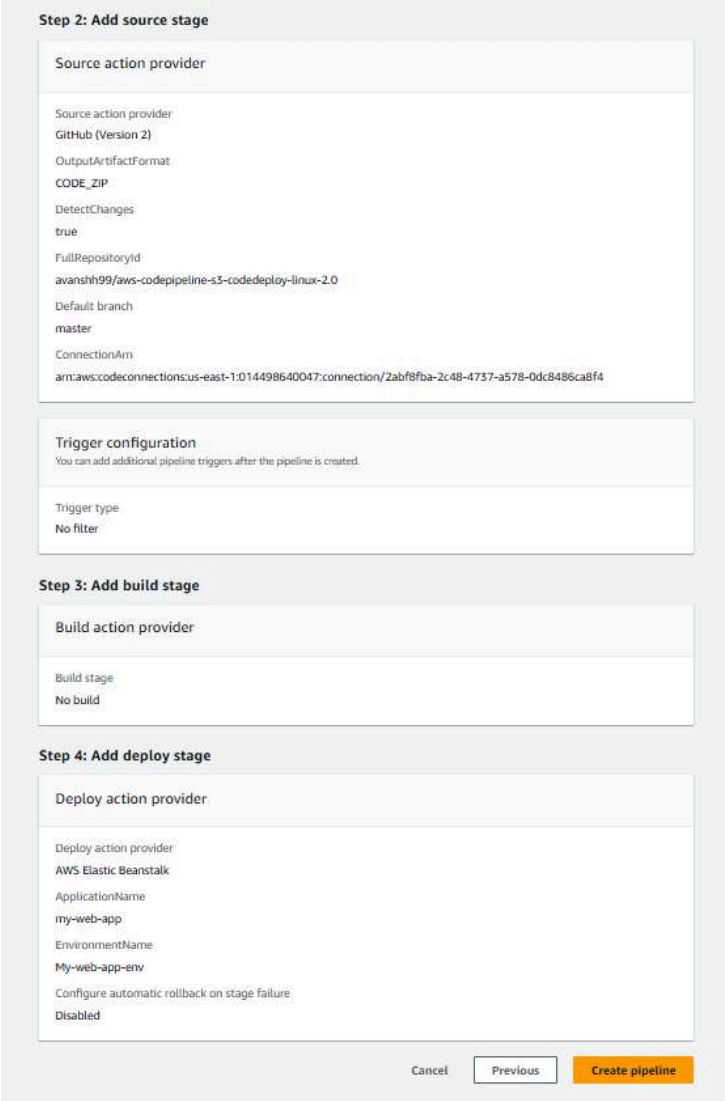
AWSCodePipelineServiceRole-ap-south-1-my-pipeline

Variables

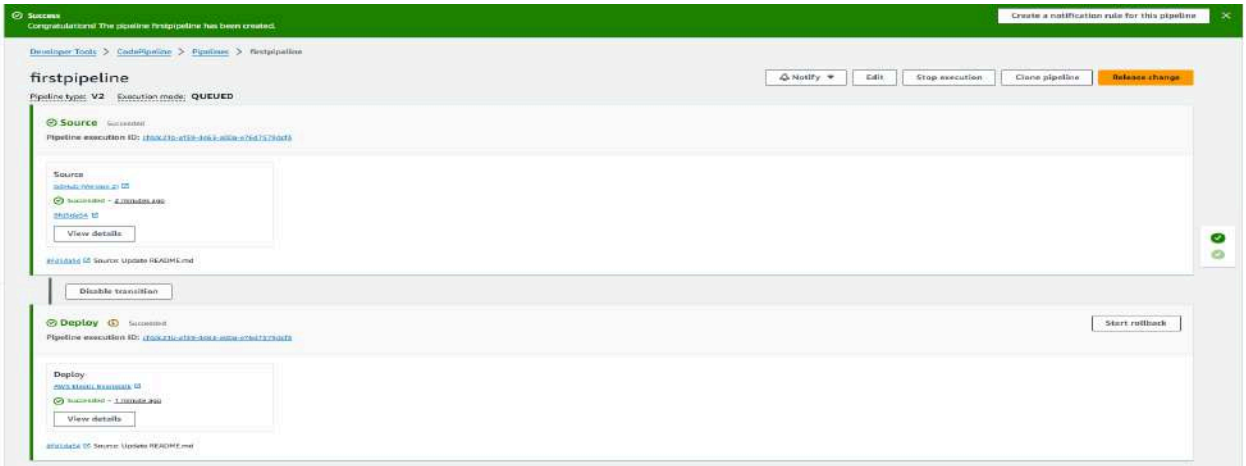
Name	Default value	Description
	No variables	
	No variables defined at the pipeline level in this pipeline.	

Step 2: Add source stage

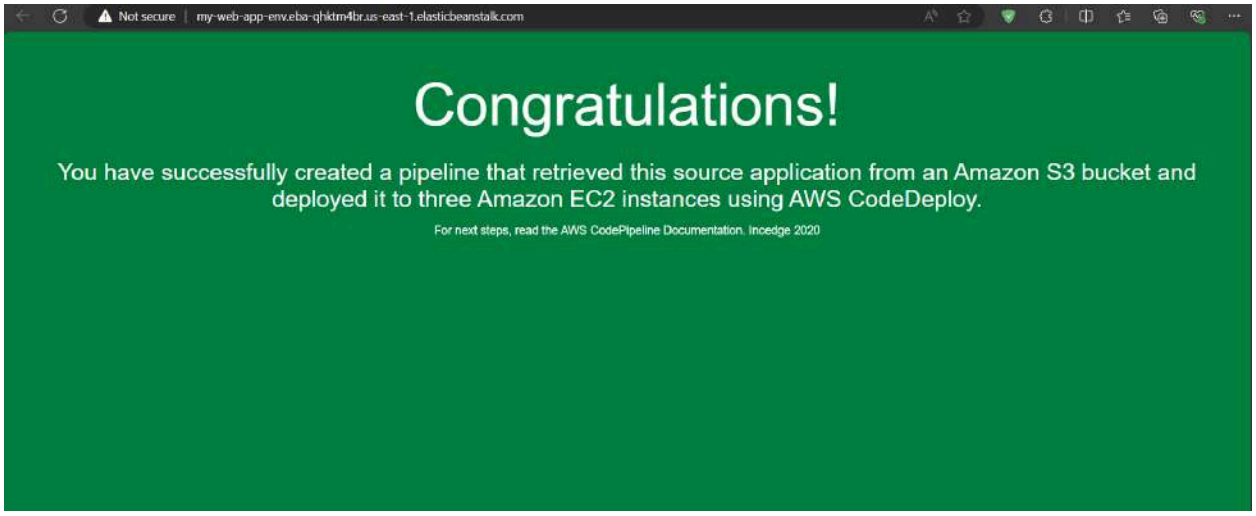
Source action provider



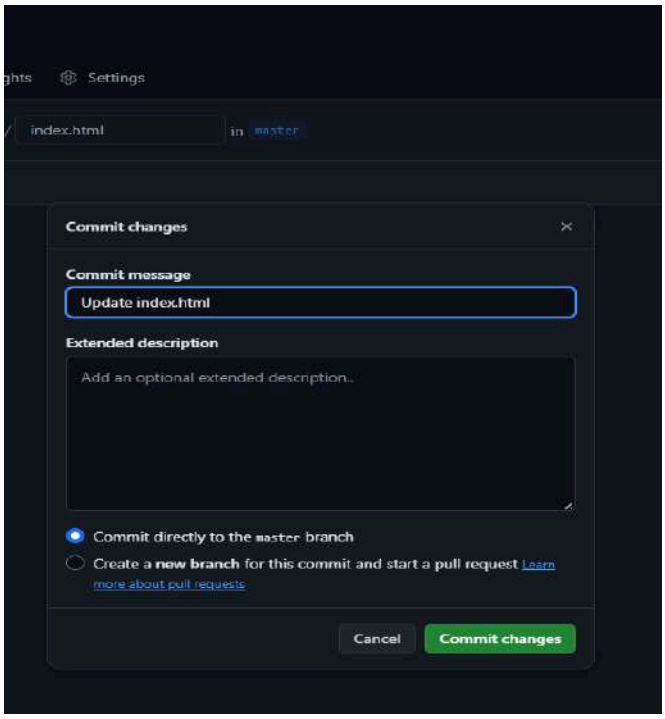
8. Once the Successfully created message appears, your pipeline is created.Then go ahead and check the URL provided in the EBS environment.



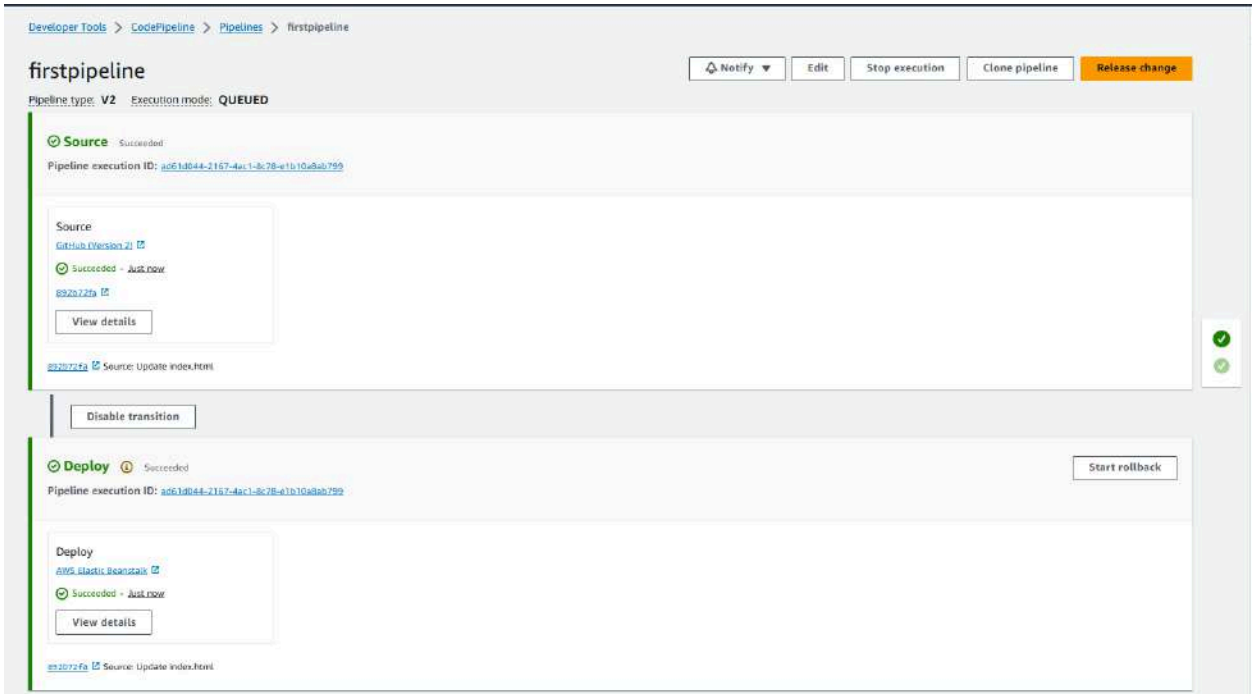
9. This is the website hosted from that forked repo in our beanstalk environment



10. Go to the repository and make the changes in the index.html file and commit them



11.The changes that are committed can be noticed in the source panel in real time and to view the changes check the url (refresh it) and you can view the changes once the deployment section shows success.


Developer Tools > CodePipeline > Pipelines > firstpipeline
firstpipeline
Notify
Edit
Stop execution
Clone pipeline
Release change
Pipeline type: V2
Execution mode: QUEUED
Source
Succeeded
Pipeline execution ID: a061d044-2167-4ac1-8c78-e1b10a8ab799
Source
GitHub (Version 2)
Succeeded - Just now
892b72fa
View details
892b72fa Source: Update index.html
Disable transition
Deploy
Succeeded
Start rollback
Pipeline execution ID: a061d044-2167-4ac1-8c78-e1b10a8ab799
Deploy
AWS Elastic Beanstalk
Succeeded - Just now
View details
892b72fa Source: Update index.html

Congratulations!
Krushikesh Shelar Roll No. 51
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

Name:Krushikesh Shelar 2024-2025 D15C Roll No:51

Experiment 3:

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

Theory:

Container-based microservices architectures have revolutionized how development and operations teams test and deploy modern software. Containers allow companies to scale and deploy applications more efficiently, but they also introduce new challenges, adding complexity by creating a whole new infrastructure ecosystem.

Today, both large and small software companies are deploying thousands of container instances daily. Managing this level of complexity at scale requires advanced tools. Enter Kubernetes.

Originally developed by Google, Kubernetes is an open-source container orchestration platform designed to automate the deployment, scaling, and management of containerized applications. Kubernetes has quickly become the de facto standard for container orchestration and is the flagship project of the Cloud Native Computing Foundation (CNCF), supported by major players like Google, AWS, Microsoft, IBM, Intel, Cisco, and Red Hat.

Kubernetes simplifies the deployment and operation of applications in a microservice architecture by providing an abstraction layer over a group of hosts. This allows development teams to deploy their applications while Kubernetes takes care of key tasks, including:

- Managing resource consumption by applications or teams
- Distributing application load evenly across the infrastructure
- Automatically load balancing requests across multiple instances of an application
- Monitoring resource usage to prevent applications from exceeding resource limits and automatically restarting them if needed
- Moving application instances between hosts when resources are low or if a host fails
- Automatically utilizing additional resources when new hosts are added to the cluster
- Facilitating canary deployments and rollbacks with ease

Necessary Requirements:

● **EC2 Instance:** The experiment required launching a t2.medium EC2 instance with 2 CPUs, as Kubernetes demands sufficient resources for effective functioning.

● **Minimum Requirements:**

- ○ **Instance Type:** t2.medium
- ○ **CPUs:** 2
- ○ **Memory:** Adequate for container orchestration.

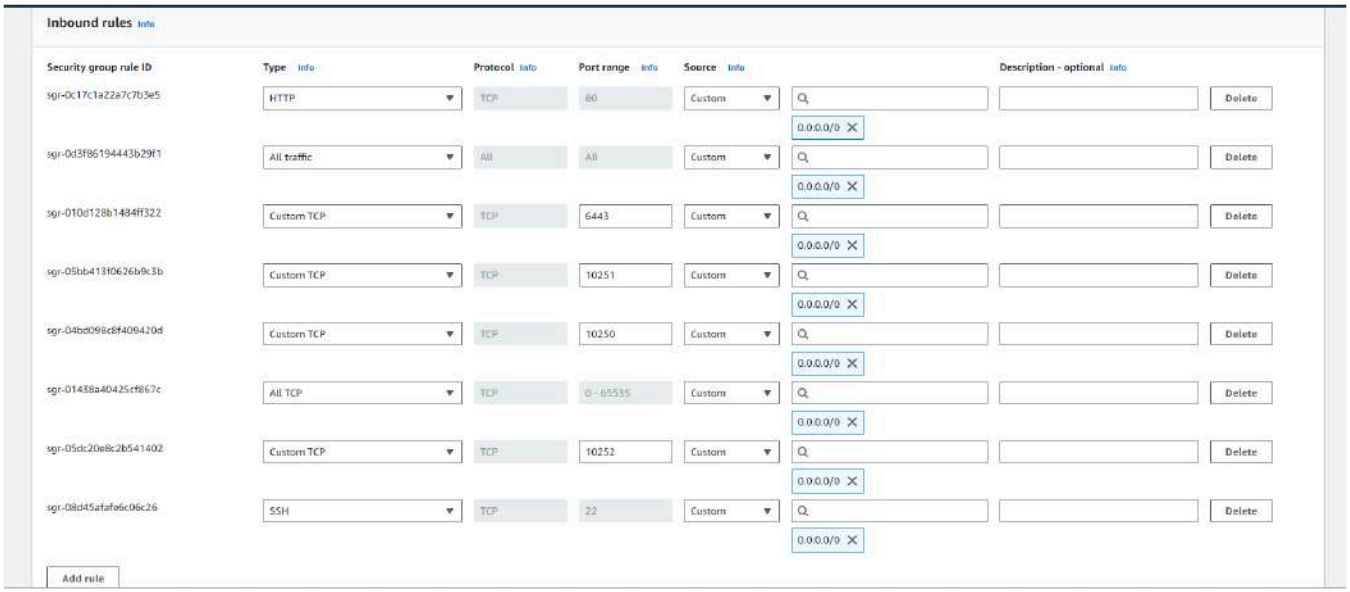
This ensured that the Kubernetes cluster had the necessary resources to function smoothly

Prerequisites :

Create 2 Security Groups for Master and Nodes and add the following rules inbound rules in those Groups.

Name:Krushikesh Shelar **2024-2025** **D15C** **Roll No:51**

Master:

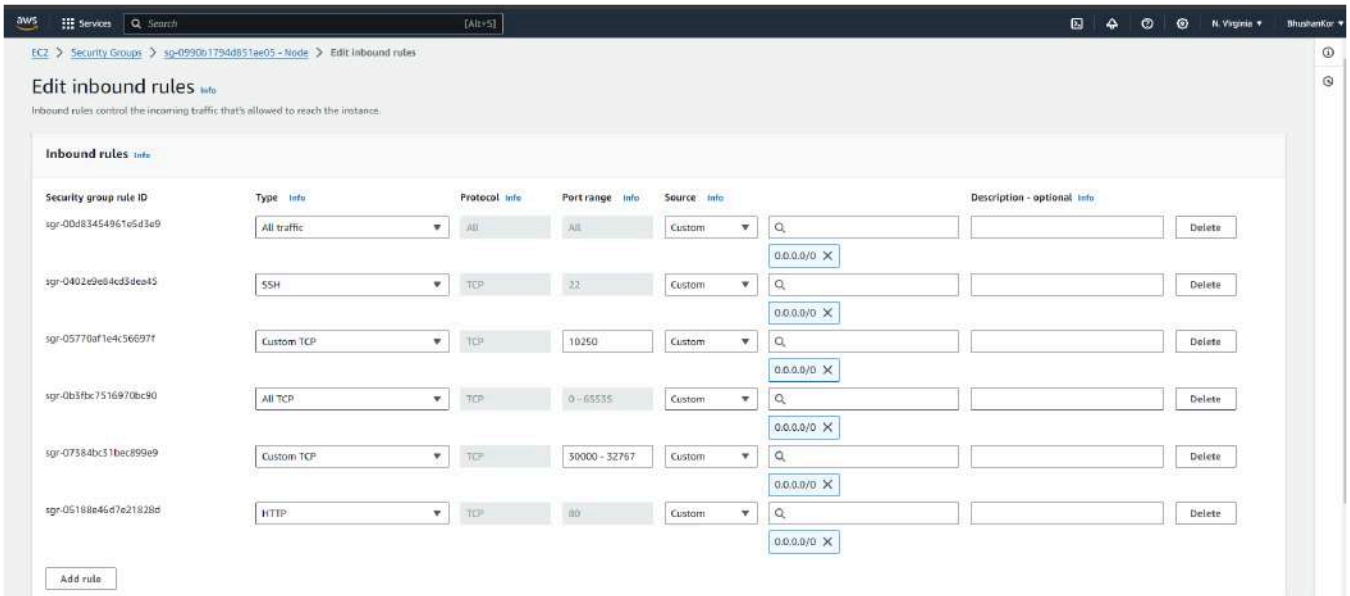


Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-0c17c1a22a7c7b3e5	HTTP	TCP	80	Custom	0.0.0.0/0		Delete
sgr-0d3f86194443b29f1	All traffic	All	All	Custom	0.0.0.0/0		Delete
sgr-010d128b1484ff322	Custom TCP	TCP	6443	Custom	0.0.0.0/0		Delete
sgr-05bb413f0626b9c3b	Custom TCP	TCP	10251	Custom	0.0.0.0/0		Delete
sgr-04bd098c8f409420d	Custom TCP	TCP	10250	Custom	0.0.0.0/0		Delete
sgr-01438a40425cf867c	All TCP	TCP	0 - 65535	Custom	0.0.0.0/0		Delete
sgr-05dc20e8c2b541402	Custom TCP	TCP	10252	Custom	0.0.0.0/0		Delete
sgr-08d45afafa6c06c26	SSH	TCP	22	Custom	0.0.0.0/0		Delete

Add rule

Node :



EC2 > Security Groups > sg-0990b1794d851ae05 - Node > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-00d83454961e5d3e9	All traffic	All	All	Custom	0.0.0.0/0		Delete
sgr-0402e9e84cd3dea45	SSH	TCP	22	Custom	0.0.0.0/0		Delete
sgr-05770af1e4c56697f	Custom TCP	TCP	10250	Custom	0.0.0.0/0		Delete
sgr-0b3fbc7516970bc90	All TCP	TCP	0 - 65535	Custom	0.0.0.0/0		Delete
sgr-07384bc31bec899e9	Custom TCP	TCP	30000 - 32767	Custom	0.0.0.0/0		Delete
sgr-05188e46d7e21828d	HTTP	TCP	80	Custom	0.0.0.0/0		Delete

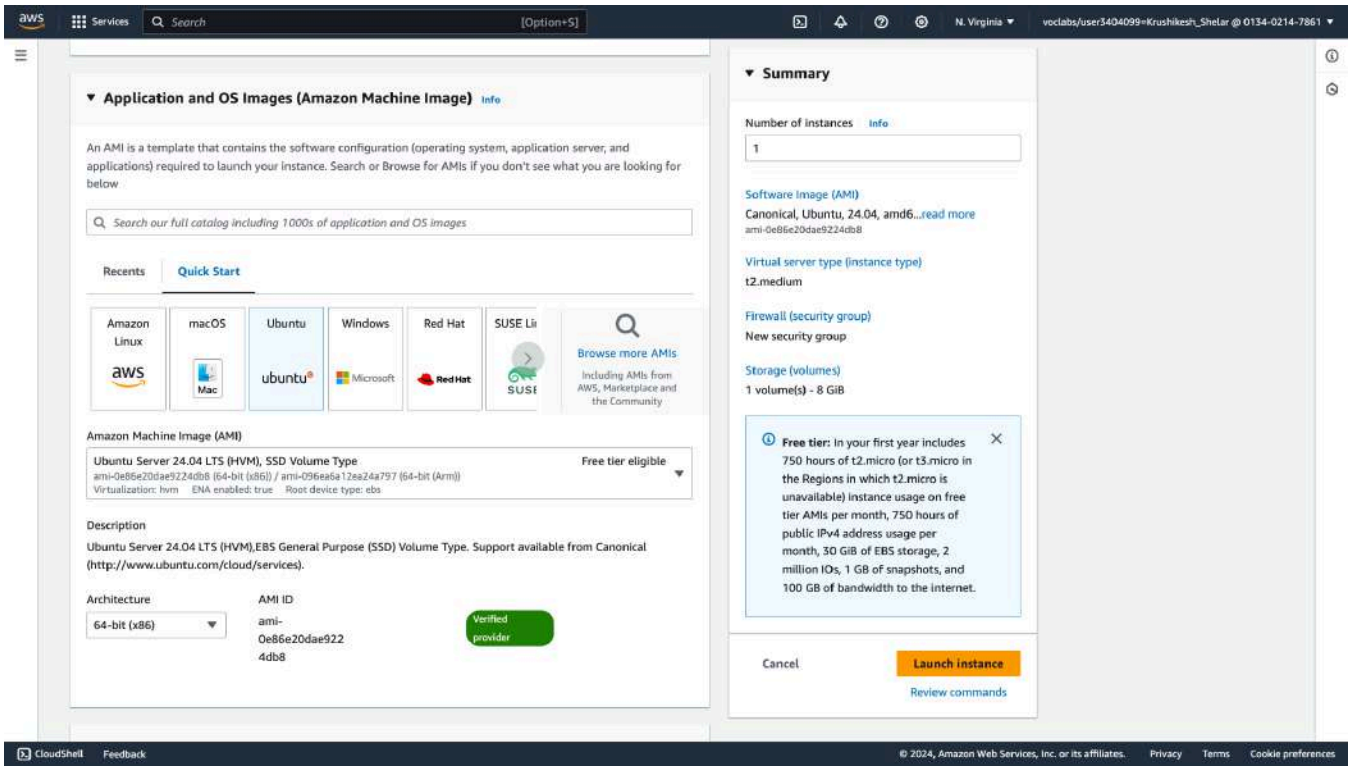
Add rule

Step 1: Log in to your AWS Academy/personal account and launch 3 new Ec2 Instances. Select Ubuntu as AMI and t2.medium as Instance Type and create a key of type RSA with .pem extension and move the downloaded key to the new folder.We can use 3 Different keys or 1 common key also.

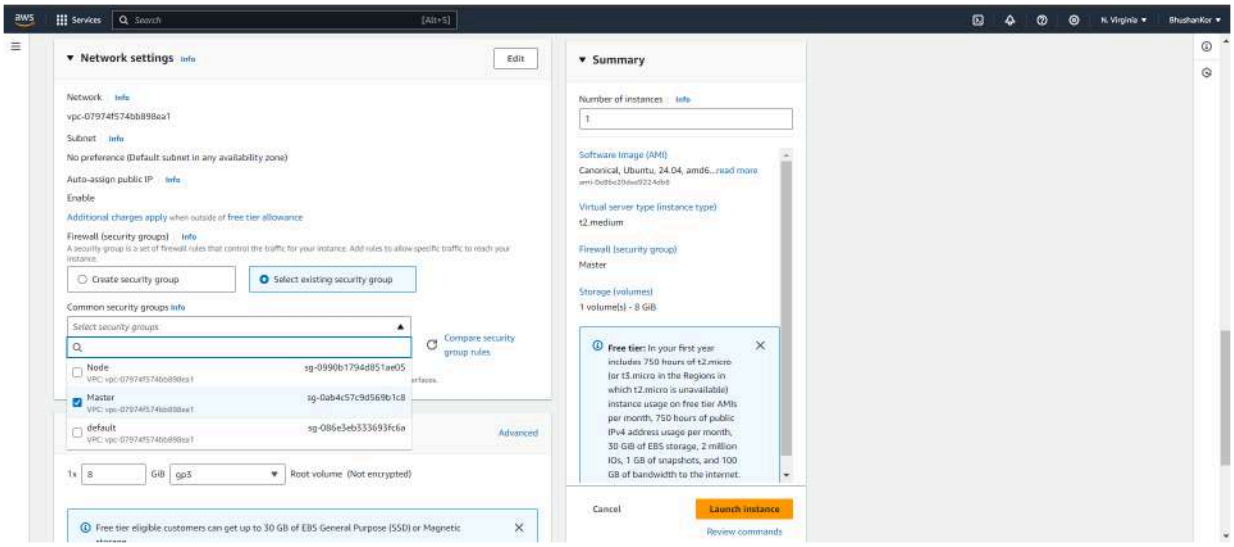
Name:Krushikesh Shelar **2024-2025** **D15C** **Roll No:51**

Note: A minimum of 2 CPUs are required so Please select t2.medium and do not forget to stop the instance after the experiment because it is not available in the free tier.
Also Select Security groups from existing.

Master:



Do Same for 2 Workers and use security groups of Worker for that.



Step 2: After creating the instances click on Connect & connect all 3 instances and navigate to EC2 Instance Connect.

Name:Krushikesh Shelar 2024-2025 D15C Roll No:51

Find Instance by attribute or tag (case-sensitive) | All states

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
Master	i-04e9c4f197de4c227	Running	t2.medium	Initializing	View alarms +	us-east-1b	ec2-54-84-20-71.comp...
Worker1	i-0ce16fc1050fd845f	Running	t2.medium	Initializing	View alarms +	us-east-1b	ec2-34-224-78-70.com...
Worker2	i-0f19cd9b03f5b7c7d	Running	t2.medium	Initializing	View alarms +	us-east-1b	ec2-3-85-184-179.com...

EC2 > Instances > i-0ce16fc1050fd845f > Connect to instance

Connect to instance Info

Connect to your instance i-0ce16fc1050fd845f (Worker1) using any of these options

EC2 Instance Connect | Session Manager | SSH client | EC2 serial console

All ports are open to all IPv4 addresses in your security group
All ports are currently open to all IPv4 addresses, indicated by **All** and **0.0.0.0/0** in the inbound rule in your security group. For increased security, consider restricting access to only the EC2 Instance Connect service IP addresses for your Region: 18.206.107.24/29. Learn more.

Instance ID
i-0ce16fc1050fd845f (Worker1)

Connection Type
- Connect using EC2 Instance Connect — Connect using the EC2 Instance Connect browser-based client, with a public IPv4 address.
- Connect using EC2 Instance Connect Endpoint — Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.

Public IPv4 address
34.224.78.70

Username
Enter the username defined in the AMI used to launch the instance. If you didn't define a custom username, use the default username, ubuntu.
ubuntu

Note: In most cases, the default username, ubuntu, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Cancel | Connect

Successful Connection:

Name:Krushikesh Shelar 2024-2025 D15C Roll No:51

```
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

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

 System information as of Thu Sep 19 07:24:05 UTC 2024

  System load:  0.08               Processes:             115
  Usage of /:   22.7% of 6.71GB    Users logged in:       0
  Memory usage: 5%                 IPv4 address for enX0: 172.31.26.25
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

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

ubuntu@ip-172-31-26-25:~$
```

i-0ce16fc1050fd845f (Worker1)
PublicIPs: 34.224.78.70 PrivateIPs: 172.31.26.25

Step 4: Run on Master,Node 1,and Node 2 the below commands to install and setup Docker in Master, Node1, and Node2.

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

```
ubuntu@ip-172-31-17-92:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [13.8 kB]
```

sudo apt-get update
sudo apt-get install -y docker-ce

```
Setting up pigz (2.8-1) ...
Setting up docker-ce-rootless-extras (5:27.2.1-1~ubuntu.24.04~noble) ...
Setting up slirp4netns (1.2.1-1build2) ...
Setting up docker-ce (5:27.2.1-1~ubuntu.24.04~noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-17-92:~$ docker --version
Docker version 27.2.1, build 9e34c9b
```

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

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

```
ubuntu@ip-172-31-17-92:~$ docker --version
Docker version 27.2.1, build 9e34c9b
ubuntu@ip-172-31-17-92:~$ sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
    "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
    "exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-17-92:~$ sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
```

Step 5: Run the below command to install Kubernetes.
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

Name:Krushikesh Shelar 2024-2025 D15C Roll No:51

```
ubuntu@ip-172-31-20-171:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyri
ng.gpg
ubuntu@ip-172-31-20-171:~$ echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /et
c/apt/sources.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
```

```
ubuntu@ip-172-31-27-176:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.3
1/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-k
eyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://p
kgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/ku
bernetes.list

deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8
s.io/core:/stable:/v1.31/deb/ /
ubuntu@ip-172-31-27-176:~$
```

sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl

```
Setting up kubernetes-cni (1.5.1-1.1) ...
Setting up kubeadm (1.31.1-1.1) ...
Setting up kubelet (1.31.1-1.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
ubuntu@ip-172-31-17-92:~$
```

sudo systemctl enable --now kubelet
sudo apt-get install -y containerd

```
Unpacking runc (1.1.12-0ubuntu3.1) ...
Selecting previously unselected package containerd.
Preparing to unpack .../containerd_1.7.12-0ubuntu4.1_amd64.deb ...
Unpacking containerd (1.7.12-0ubuntu4.1) ...
Setting up runc (1.1.12-0ubuntu3.1) ...
Setting up containerd (1.7.12-0ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-17-92:~$
```

Name:Krushikesh Shelar 2024-2025 D15C Roll No:51

sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml

```
ubuntu@ip-172-31-27-176:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

[debug]
  address = ""
  format = ""
  gid = 0
  level = ""
  uid = 0

[grpc]
  address = "/run/containerd/containerd.sock"
  gid = 0
```

```
[timeouts]
  "io.containerd.timeout.bolt.open" = "0s"
  "io.containerd.timeout.metrics.shimstats" = "2s"
  "io.containerd.timeout.shim.cleanup" = "5s"
  "io.containerd.timeout.shim.load" = "5s"
  "io.containerd.timeout.shim.shutdown" = "3s"
  "io.containerd.timeout.task.state" = "2s"

[ttrpc]
  address = ""
  gid = 0
  uid = 0
```

sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd

```
ubuntu@ip-172-31-17-92:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd

● containerd.service - containerd container runtime
     Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
     Active: active (running) since Thu 2024-09-19 07:47:29 UTC; 269ms ago
       Docs: https://containerd.io
   Main PID: 5079 (containerd)
      Tasks: 8
     Memory: 13.3M (peak: 14.0M)
        CPU: 69ms
     CGroup: /system.slice/containerd.service
             └─5079 /usr/bin/containerd

Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2743305662" level=info msg="Start subscribing containerd event"
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2743708522" level=info msg="Start recovering state"
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2744179092" level=info msg="Start event monitor"
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2744280712" level=info msg=serving... address=/run/containerd/containerd.sock.ttrpc
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2744300452" level=info msg="Start snapshots syncer"
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2744582012" level=info msg="Start cni network conf syncer for default"
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2744618082" level=info msg=serving... address=/run/containerd/containerd.sock
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2744654812" level=info msg="Start streaming server"
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2745640952" level=info msg="containerd successfully booted in 0.028249s"
Sep 19 07:47:29 ip-172-31-17-92 systemd[1]: Started containerd.service - containerd container runtime.
```

sudo apt-get install -y socat

```
ubuntu@ip-172-31-17-92:~$  sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 133 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.0.0-4build3 [374 kB]
Fetched 374 kB in 0s (10.8 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68108 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.0-4build3_amd64.deb ...
Unpacking socat (1.8.0.0-4build3) ...
Setting up socat (1.8.0.0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

Step 6: Initialize the Kubecluster .Now Perform this Command only for Master.

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

Run this command on master and also copy and save the Join command from above**.**

```
[mark-control-plane] Marking the node ip-172-31-17-92 as control-plane by adding the taints [node-role.kubernetes.io/control-pla
[bootstrap-token] Using token: ikfl3a.g5zi9o3q6ac1cw2a
[bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Roles
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to get nodes
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certific
[bootstrap-token] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Tok
[bootstrap-token] Configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

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 172.31.17.92:6443 --token ikfl3a.g5zi9o3q6ac1cw2a \
        --discovery-token-ca-cert-hash sha256:43d27d0955ab782c8877f38d04e1146411c2c510ff75b82f8212bbf7f50db3dd
```

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

```
kubeadm join 172.31.17.92:6443 --token ikfl3a.g5zi9o3q6ac1cw2a \
        --discovery-token-ca-cert-hash sha256:43d27d0955ab782c8877f38d04e1146411c2c510ff75b82f8212bbf7f50db3dd
ubuntu@ip-172-31-17-92:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Step 7: Now Run the command kubectl get nodes to see the nodes before executing Join command on nodes.

```
ubuntu@ip-172-31-17-92:~$ kubectl get nodes
NAME              STATUS     ROLES           AGE     VERSION
ip-172-31-17-92   NotReady   control-plane   2m56s   v1.31.1
```

Step 8: Now Run the following command on Node 1 and Node 2 to Join to master.

sudo kubeadm join 172.31.17.92:6443 --token ikfl3a.g5zi9o3q6ac1cw2a
--discovery-token-ca-cert-hash
sha256:43d27d0955ab782c8877f38d04e1146411c2c510ff75b82f8212bbf7f50db3dd

Worker 1:

```
ubuntu@ip-172-31-26-25:~$ kubeadm join 172.31.17.92:6443 --token ikfl3a.g5zi9o3q6ac1cw2a \
        --discovery-token-ca-cert-hash sha256:43d27d0955ab782c8877f38d04e1146411c2c510ff75b82f8212bbf7f50db3dd
[preflight] Running pre-flight checks
error execution phase preflight: [preflight] Some fatal errors occurred:
        [ERROR IsPrivilegedUser]: user is not running as root
[preflight] If you know what you are doing, you can make a check non-fatal with `--ignore-preflight-errors=...`
To see the stack trace of this error execute with --v=5 or higher
ubuntu@ip-172-31-26-25:~$ sudo kubeadm join 172.31.17.92:6443 --token ikfl3a.g5zi9o3q6ac1cw2a \
        --discovery-token-ca-cert-hash sha256:43d27d0955ab782c8877f38d04e1146411c2c510ff75b82f8212bbf7f50db3dd
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[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-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 501.106777ms
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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.
```

Worker 2:

```
adm join 172.31.17.92:6443 --token ikfl3a.g5zi9o3q6ac1cw2a         --discovery-token-ca-cert-hash sha256:43d27d0955ab
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[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-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 1.000966554s
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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.
```

Step 9: Now Run the command kubectl get nodes to see the nodes after executing Join command on nodes.

```
ubuntu@ip-172-31-17-92:~$ kubectl get nodes
NAME                STATUS     ROLES           AGE     VERSION
ip-172-31-17-92     NotReady   control-plane   2m56s   v1.31.1
ubuntu@ip-172-31-17-92:~$ kubectl get nodes
NAME                STATUS     ROLES           AGE     VERSION
ip-172-31-17-92     NotReady   control-plane   8m13s   v1.31.1
ip-172-31-19-166    NotReady   <none>          73s     v1.31.1
ip-172-31-26-25     NotReady   <none>          2m9s    v1.31.1
```

Step 10: Since Status is NotReady we have to add a network plugin. And also we have to give the name to the nodes.

kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml

Name:Krushikesh Shelar 2024-2025 D15C Roll No:51

```
ubuntu@ip-172-31-17-92:~$ kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
poddisruptionbudget.policy/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
serviceaccount/calico-node created
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgppeers.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/blockaffinities.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/caliconodestatuses.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/clusterinformations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/felixconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworksets.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/hostendpoints.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamblocks.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamconfigs.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamhandles.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippools.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipreservations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/kubecontrollersconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networksets.crd.projectcalico.org created
clusterrole.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrole.rbac.authorization.k8s.io/calico-node created
clusterrolebinding.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
daemonset.apps/calico-node created
deployment.apps/calico-kube-controllers created
```

sudo systemctl status kubelet

```
ubuntu@ip-172-31-17-92:~$ sudo systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
     Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; preset: enabled)
    Drop-In: /usr/lib/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: active (running) since Thu 2024-09-19 07:52:00 UTC; 11min ago
       Docs: https://kubernetes.io/docs/
   Main PID: 6095 (kubelet)
      Tasks: 11 (limit: 4676)
     Memory: 33.4M (peak: 33.9M)
        CPU: 10.601s
     CGroup: /system.slice/kubelet.service
             └─6095 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf --config=/var/lib/kubelet/config.>

Sep 19 08:03:09 ip-172-31-17-92 kubelet[6095]:          : unknown
Sep 19 08:03:09 ip-172-31-17-92 kubelet[6095]:  > pod="kube-system/kube-controller-manager-ip-172-31-17-92" podUID="6b8789cacbbef1a009680071caa0155b" containerName="kube-cont>
Sep 19 08:03:09 ip-172-31-17-92 kubelet[6095]: E0919 08:03:09.952283    6095 log.go:32] "StopPodSandbox from runtime service failed" err=<
Sep 19 08:03:09 ip-172-31-17-92 kubelet[6095]:         rpc error: code = Unknown desc = failed to stop container "6e2944b387635ee7a4b188076822bbec229e1c514e6c5ca4aca41090e1ec>
Sep 19 08:03:09 ip-172-31-17-92 kubelet[6095]:          : unknown
Sep 19 08:03:09 ip-172-31-17-92 kubelet[6095]:  > podSandboxID="2d281910fe2ca3000c55d391cb651d4941eb3d1b239d3658e4396afd0b3755b4"
Sep 19 08:03:09 ip-172-31-17-92 kubelet[6095]: E0919 08:03:09.952334    6095 kuberuntime_manager.go:1479] "Failed to stop sandbox" podSandboxID={"Type":"containerd","ID":"2d2>
Sep 19 08:03:09 ip-172-31-17-92 kubelet[6095]: E0919 08:03:09.952373    6095 kubelet.go:1865] "KillPod failed" err="[failed to \"KillContainer\" for \"kube-controller-manager>
Sep 19 08:03:11 ip-172-31-17-92 kubelet[6095]: E0919 08:03:11.145213    6095 kubelet.go:2902] "Container runtime network not ready" networkReady="NetworkReady=false reason:Ne>
Sep 19 08:03:18 ip-172-31-17-92 kubelet[6095]: I0919 08:03:18.207896    6095 pod_startup_latency_tracker.go:104] "Observed pod startup duration" pod="kube-system/calico-node->
...skipping...
● kubelet.service - kubelet: The Kubernetes Node Agent
     Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; preset: enabled)
    Drop-In: /usr/lib/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: active (running) since Thu 2024-09-19 07:52:00 UTC; 11min ago
```

Now Run command kubectl get nodes -o wide we can see Status is ready.

```
ubuntu@ip-172-31-17-92:~$ kubectl get nodes -o wide
NAME              STATUS   ROLES           AGE     VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-17-92   Ready    control-plane   14m     v1.31.1   172.31.17.92    <none>        Ubuntu 24.04 LTS     6.8.0-1012-aws   containerd://1.7.12
ip-172-31-19-166  Ready    <none>          7m11s   v1.31.1   172.31.19.166   <none>        Ubuntu 24.04 LTS     6.8.0-1012-aws   containerd://1.7.12
ip-172-31-26-25   Ready    <none>          8m7s    v1.31.1   172.31.26.25    <none>        Ubuntu 24.04 LTS     6.8.0-1012-aws   containerd://1.7.12
```

Now to Rename run this command

```
ubuntu@ip-172-31-27-176:~$ sudo systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
     Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; preset: enabled)
    Drop-In: /usr/lib/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: active (running) since Mon 2024-09-16 15:40:01 UTC; 11min ago
       Docs: https://kubernetes.io/docs/
   Main PID: 5989 (kubelet)
      Tasks: 10 (limit: 4676)
     Memory: 32.6M (peak: 33.2M)
        CPU: 10.705s
     CGroup: /system.slice/kubelet.service
             └─5989 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf --config=/var/>

Sep 16 15:51:29 ip-172-31-27-176 kubelet[5989]: I0916 15:51:29.497458    5989 reconciler_common.go:245] "operationExecutor.VerifyControllerAttachedVolume s>
Sep 16 15:51:29 ip-172-31-27-176 kubelet[5989]: I0916 15:51:29.497516    5989 reconciler_common.go:245] "operationExecutor.VerifyControllerAttachedVolume s>
Sep 16 15:51:29 ip-172-31-27-176 kubelet[5989]: I0916 15:51:29.497569    5989 reconciler_common.go:245] "operationExecutor.VerifyControllerAttachedVolume s>
Sep 16 15:51:29 ip-172-31-27-176 kubelet[5989]: I0916 15:51:29.497620    5989 reconciler_common.go:245] "operationExecutor.VerifyControllerAttachedVolume s>
Sep 16 15:51:29 ip-172-31-27-176 kubelet[5989]: I0916 15:51:29.497669    5989 reconciler_common.go:245] "operationExecutor.VerifyControllerAttachedVolume s>
Sep 16 15:51:29 ip-172-31-27-176 kubelet[5989]: I0916 15:51:29.497719    5989 reconciler_common.go:245] "operationExecutor.VerifyControllerAttachedVolume s>
Sep 16 15:51:31 ip-172-31-27-176 kubelet[5989]: E0916 15:51:31.605091    5989 kubelet.go:2902] "Container runtime network not ready" networkReady="NetworkR>
Sep 16 15:51:32 ip-172-31-27-176 kubelet[5989]: I0916 15:51:32.366237    5989 scope.go:117] "RemoveContainer" containerID="f44f06967c5b3e567e07841a7b4352ae>
Sep 16 15:51:36 ip-172-31-27-176 kubelet[5989]: E0916 15:51:36.606675    5989 kubelet.go:2902] "Container runtime network not ready" networkReady="NetworkR>
Sep 16 15:51:41 ip-172-31-27-176 kubelet[5989]: E0916 15:51:41.608404    5989 kubelet.go:2902] "Container runtime network not ready" networkReady="NetworkR>
```

kubectl label node ip-172-31-18-135 kubernetes.io/role=worker

Rename to Node 1:kubectl label node ip-172-31-28-117 kubernetes.io/role=worker1
Rename to Node 2:kubectl label node ip-172-31-18-135 kubernetes.io/role=worker2

Step 11: Run command kubectl get nodes -o wide . And Hence we can see we have Successfully connected Node 1 and Node 2 to the Master.

run kubectl get nodes

```
ubuntu@ip-172-31-17-92:~$ kubectl get nodes -o wide
NAME               STATUS   ROLES           AGE     VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE           KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-17-92    Ready    control-plane   14m     v1.31.1   172.31.17.92    <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-19-166   Ready    <none>          7m11s   v1.31.1   172.31.19.166   <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-26-25    Ready    <none>          8m7s    v1.31.1   172.31.26.25    <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ubuntu@ip-172-31-17-92:~$ kubectl label node ip-172-31-26-25  kubernetes.io/role=worker1
node/ip-172-31-26-25 labeled
ubuntu@ip-172-31-17-92:~$ kubectl label node ip-172-31-19-166  kubernetes.io/role=worker2
node/ip-172-31-19-166 labeled
ubuntu@ip-172-31-17-92:~$ kubectl get nodes
NAME               STATUS   ROLES           AGE   VERSION
ip-172-31-17-92    Ready    control-plane   19m   v1.31.1
ip-172-31-19-166   Ready    worker2         12m   v1.31.1
ip-172-31-26-25    Ready    worker1         13m   v1.31.1
```

Conclusion: In this experiment, we successfully established a Kubernetes cluster consisting of one master node and two worker nodes on AWS EC2 instances. After setting up Docker and installing essential Kubernetes tools (kubelet, kubeadm, kubectl) as well as containerd on all nodes, we proceeded to initialize the master node. The worker nodes were then joined to the cluster. Initially, we encountered a NotReady state for the nodes, but this issue was promptly resolved by installing the Calico network plugin. Additionally, we labeled the nodes to designate their roles (control-plane and worker). As a result, the cluster became fully operational, with all nodes transitioning to the Ready state, effectively demonstrating the successful configuration and orchestration of Kubernetes.

Name: Krushikesh Sunil Shelar **2024-2025** **D15-C** **Roll No:51**

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

Theory:

Kubernetes, originally developed by Google, is an open-source container orchestration platform. It automates the deployment, scaling, and management of containerized applications, ensuring high availability and fault tolerance. Kubernetes is now the industry standard for container orchestration and is governed by the **Cloud Native Computing Foundation (CNCF)**, with contributions from major cloud and software providers like Google, AWS, Microsoft, IBM, Intel, Cisco, and Red Hat.

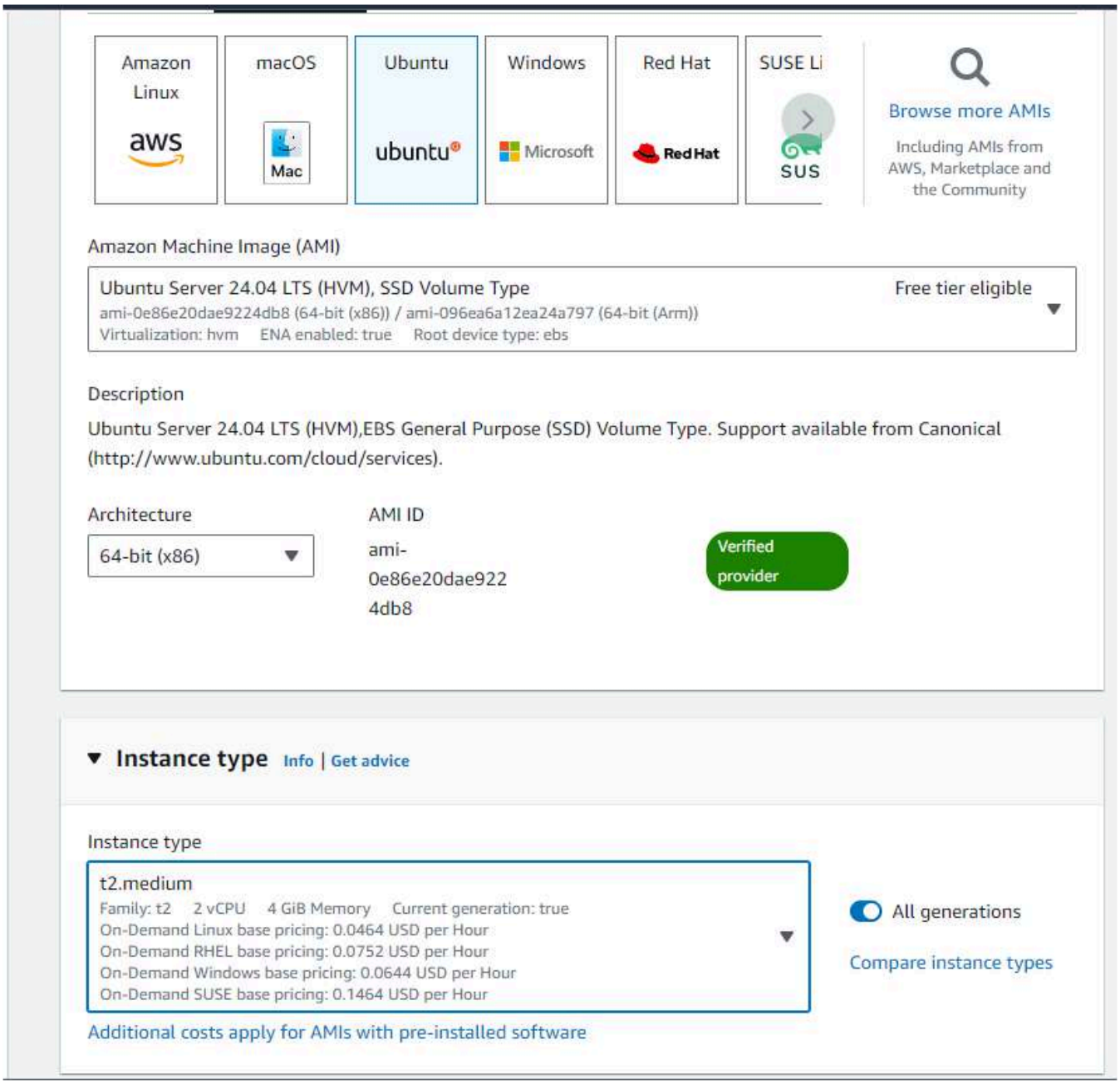
Kubernetes Deployment: Is a resource in Kubernetes that provides declarative updates for Pods and ReplicaSets. With a Deployment, you can define how many replicas of a pod should run, roll out new versions of an application, and roll back to previous versions if necessary. It ensures that the desired number of pod replicas are running at all times.

Necessary Requirements:

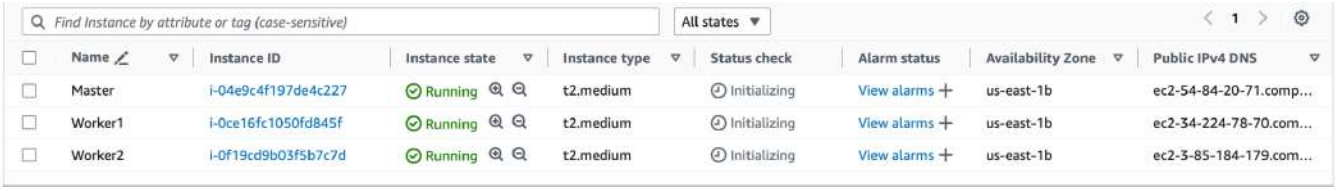
- **EC2 Instance:** The experiment required launching a t2.medium EC2 instance with 2 CPUs, as Kubernetes demands sufficient resources for effective functioning.
- **Minimum Requirements:**
    - **Instance Type:** t2.medium
    - **CPUs:** 2
    - **Memory:** Adequate for container orchestration.

This ensured that the Kubernetes cluster had the necessary resources to function smoothly.

Step 1: Log in to your AWS Academy/personal account and launch a new Ec2 Instance.
Select Ubuntu as AMI and t2.medium as Instance Note: A minimum of 2 CPUs are required so Please select t2.medium and do not forget to stop the instance after the experiment because it is not available in the free tier.



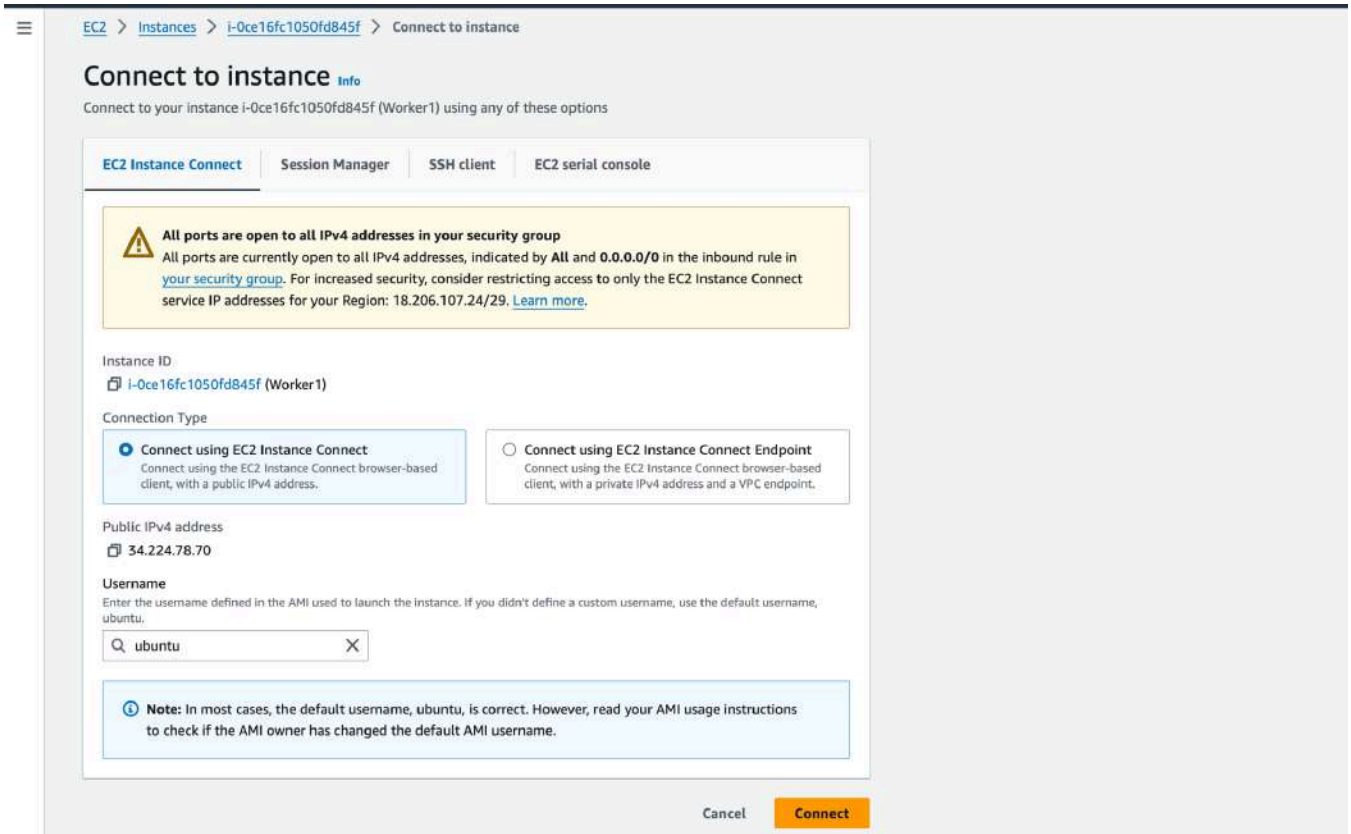
Step 2: After creating the instance click on Connect the instance and navigate to SSH Client.



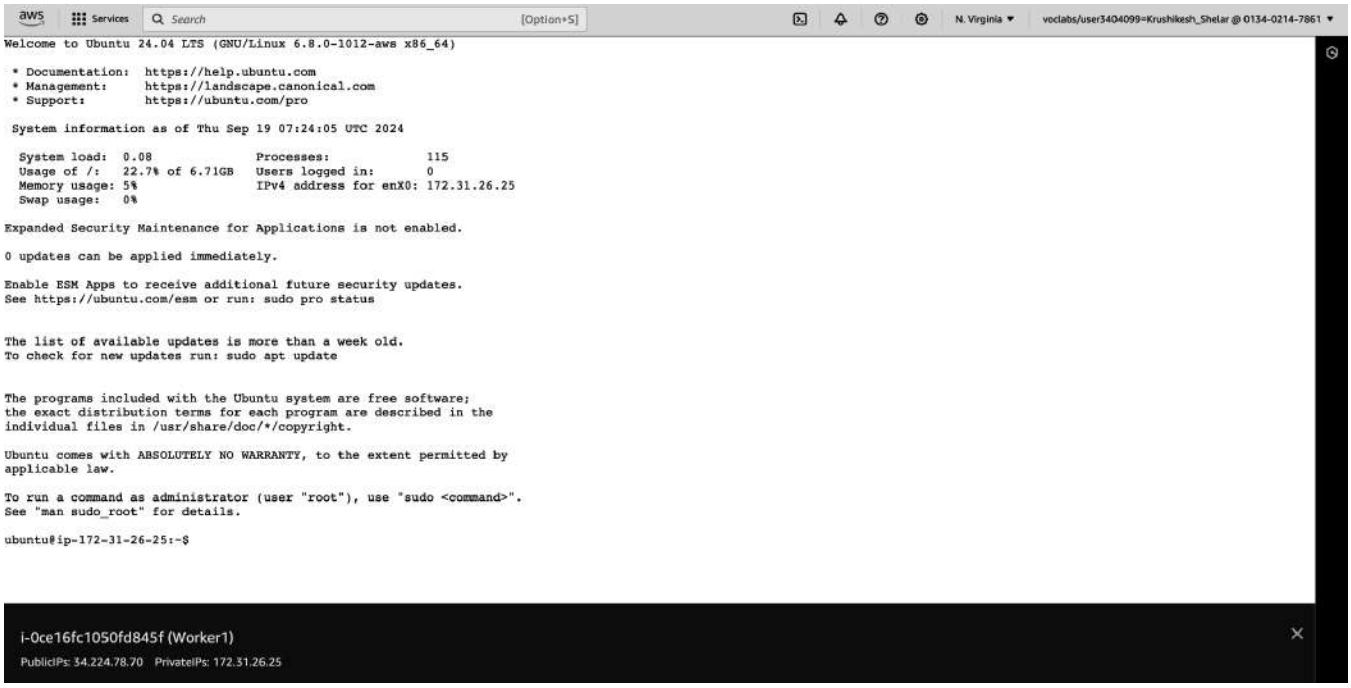
Find Instance by attribute or tag (case-sensitive) | All states | 1

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
Master	i-04e9c4f197de4c227	Running	t2.medium	Initializing	View alarms +	us-east-1b	ec2-54-84-20-71.comp...
Worker1	i-0ce16fc1050fd845f	Running	t2.medium	Initializing	View alarms +	us-east-1b	ec2-34-224-78-70.com...
Worker2	i-0f19cd9b03f5b7c7d	Running	t2.medium	Initializing	View alarms +	us-east-1b	ec2-3-85-184-179.com...

Name: Krushikesh Sunil Shelar **2024-2025** **D15-C** **Roll No:51**



Step 3: Successful Connection:



Step 4: Run on Master the below commands to install and setup Docker in Master.

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

```
ubuntu@ip-172-31-17-92:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [13.8 kB]
```

sudo apt-get update
sudo apt-get install -y docker-ce

```
Setting up pigz (2.8-1) ...
Setting up docker-ce-rootless-extras (5:27.2.1-1~ubuntu.24.04~noble) ...
Setting up slirp4netns (1.2.1-1build2) ...
Setting up docker-ce (5:27.2.1-1~ubuntu.24.04~noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-17-92:~$ docker --version
Docker version 27.2.1, build 9e34c9b
```

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

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

```
ubuntu@ip-172-31-17-92:~$ docker --version
Docker version 27.2.1, build 9e34c9b
ubuntu@ip-172-31-17-92:~$ sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
    "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
    "exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-17-92:~$ sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
```

Step 5: Run the below command to install Kubernetes.
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

```
ubuntu@ip-172-31-20-171:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyri
ng.gpg
ubuntu@ip-172-31-20-171:~$ echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /et
c/apt/sources.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
```

```
ubuntu@ip-172-31-27-176:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.3
1/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-k
eyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://p
kgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/ku
bernetes.list

deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8
s.io/core:/stable:/v1.31/deb/ /
ubuntu@ip-172-31-27-176:~$
```

sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl

```
Setting up kubernetes-cni (1.5.1-1.1) ...
Setting up kubeadm (1.31.1-1.1) ...
Setting up kubelet (1.31.1-1.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
ubuntu@ip-172-31-17-92:~$
```

sudo systemctl enable --now kubelet
sudo apt-get install -y containerd

```
Unpacking runc (1.1.12-0ubuntu3.1) ...
Selecting previously unselected package containerd.
Preparing to unpack .../containerd_1.7.12-0ubuntu4.1_amd64.deb ...
Unpacking containerd (1.7.12-0ubuntu4.1) ...
Setting up runc (1.1.12-0ubuntu3.1) ...
Setting up containerd (1.7.12-0ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-17-92:~$
```

sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml

```
ubuntu@ip-172-31-27-176:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

[debug]
  address = ""
  format = ""
  gid = 0
  level = ""
  uid = 0

[grpc]
  address = "/run/containerd/containerd.sock"
  gid = 0
```

```
[timeouts]
  "io.containerd.timeout.bolt.open" = "0s"
  "io.containerd.timeout.metrics.shimstats" = "2s"
  "io.containerd.timeout.shim.cleanup" = "5s"
  "io.containerd.timeout.shim.load" = "5s"
  "io.containerd.timeout.shim.shutdown" = "3s"
  "io.containerd.timeout.task.state" = "2s"

[ttrpc]
  address = ""
  gid = 0
  uid = 0
```

sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd

```
ubuntu@ip-172-31-17-92:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd

● containerd.service - containerd container runtime
     Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
     Active: active (running) since Thu 2024-09-19 07:47:29 UTC; 269ms ago
       Docs: https://containerd.io
   Main PID: 5079 (containerd)
      Tasks: 8
     Memory: 13.3M (peak: 14.0M)
        CPU: 69ms
     CGroup: /system.slice/containerd.service
             └─5079 /usr/bin/containerd

Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2743305662" level=info msg="Start subscribing containerd event"
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2743708522" level=info msg="Start recovering state"
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2744179092" level=info msg="Start event monitor"
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2744280712" level=info msg=serving... address=/run/containerd/containerd.sock.ttrpc
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2744300452" level=info msg="Start snapshots syncer"
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2744582012" level=info msg="Start cni network conf syncer for default"
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2744618082" level=info msg=serving... address=/run/containerd/containerd.sock
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2744654812" level=info msg="Start streaming server"
Sep 19 07:47:29 ip-172-31-17-92 containerd[5079]: time="2024-09-19T07:47:29.2745640952" level=info msg="containerd successfully booted in 0.028249s"
Sep 19 07:47:29 ip-172-31-17-92 systemd[1]: Started containerd.service - containerd container runtime.
```

sudo apt-get install -y socat

```
ubuntu@ip-172-31-17-92:~$  sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 133 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.0.0-4build3 [374 kB]
Fetched 374 kB in 0s (10.8 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68108 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.0-4build3_amd64.deb ...
Unpacking socat (1.8.0.0-4build3) ...
Setting up socat (1.8.0.0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

Step 6: Initialize the Kubecluster .Now Perform this Command only for Master.

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

Run this command on master and also copy and save the Join command from above**.**

```
[mark-control-plane] Marking the node ip-172-31-17-92 as control-plane by adding the taints [node-role.kubernetes.io/control-pla
[bootstrap-token] Using token: ikfl3a.g5zi9o3q6ac1cw2a
[bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Roles
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to get nodes
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certific
[bootstrap-token] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Tok
[bootstrap-token] Configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

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 172.31.17.92:6443 --token ikfl3a.g5zi9o3q6ac1cw2a \
        --discovery-token-ca-cert-hash sha256:43d27d0955ab782c8877f38d04e1146411c2c510ff75b82f8212bbf7f50db3dd
```

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

```
kubeadm join 172.31.17.92:6443 --token ikfl3a.g5zi9o3q6ac1cw2a \
        --discovery-token-ca-cert-hash sha256:43d27d0955ab782c8877f38d04e1146411c2c510ff75b82f8212bbf7f50db3dd
ubuntu@ip-172-31-17-92:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

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

```
ubuntu@ip-172-31-17-92:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
```

Step 7: Now that the cluster is up and running, we can deploy our nginx server on this cluster.Apply this deployment file using this command to create a deployment

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

```
ubuntu@ip-172-31-17-92:~$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
```

kubectl get pods

Name: Krushikesh Sunil Shelar 2024-2025 D15-C Roll No:51

```
ubuntu@ip-172-31-17-92:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-2s7ks    1/1     Running   0          24s
nginx-deployment-d556bf558-r8cpz    1/1     Running   0          24s
```

POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")

kubectl port-forward $POD_NAME 8080:80

```
ubuntu@ip-172-31-17-92:~$ kubectl port-forward $POD_NAME 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
```

```
^Cubuntu@ip-172-31-17-92:~kubectl taint nodes --all node-role.kubernetes.io/control-plane-node/ip-172-31-20-171 untainted

kubectl get nodes
error: at least one taint update is required
NAME               STATUS   ROLES           AGE   VERSION
ip-172-31-17-92    Ready    control-plane   41m   v1.31.1
ip-172-31-19-166   Ready    worker2         34m   v1.31.1
ip-172-31-26-25    Ready    worker1         35m   v1.31.1
```

kubectl get pods

```
ubuntu@ip-172-31-17-92:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-2s7ks    1/1     Running   0          4m2s
nginx-deployment-d556bf558-r8cpz    1/1     Running   0          4m2s
```

POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")

kubectl port-forward $POD_NAME 8080:80

Step 8: Verify your deployment
Open up a new terminal and ssh to your EC2 instance.
Then, use this curl command to check if the Nginx server is running.

curl --head http://127.0.0.1:8080

```
Last login: Thu Sep 19 07:21:00 2024 from 18.206.107.27
ubuntu@ip-172-31-17-92:~$  curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Thu, 19 Sep 2024 08:35:44 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 04 Dec 2018 14:44:49 GMT
Connection: keep-alive
ETag: "5c0692e1-264"
Accept-Ranges: bytes
```

If the response is 200 OK and you can see the Nginx server name, your deployment was successful.
We have successfully deployed our Nginx server on our EC2 instance.

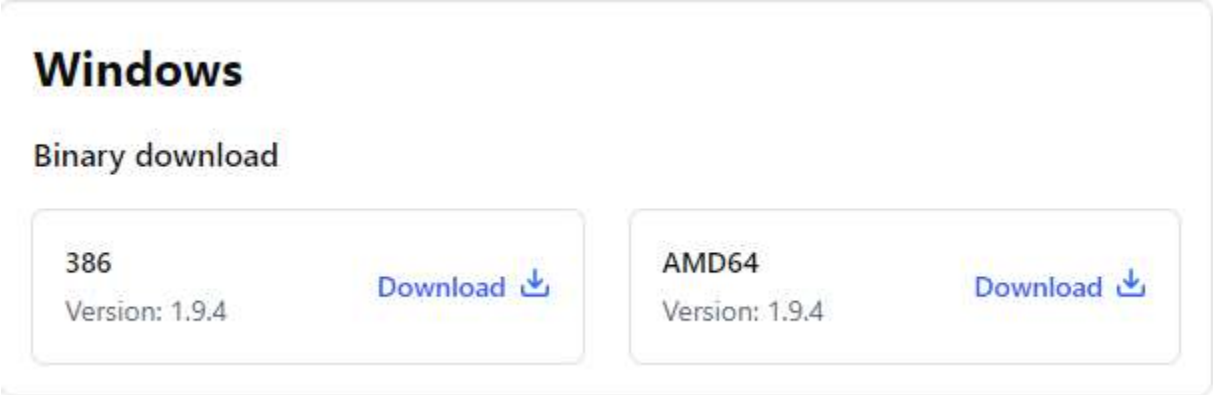
Conclusion:
In this experiment, we successfully installed Kubernetes on an EC2 instance and deployed an Nginx server using Kubectl commands. We did not encounter any major errors during the process. We efficiently managed the setup by ensuring all components were correctly configured from the start. To meet the necessary resource requirements for the Kubernetes environment, we used a t2.medium EC2 instance with 2 CPUs, which facilitated smooth deployment and operation.

Name:Krushikesh Shelar D15-C Roll No:51

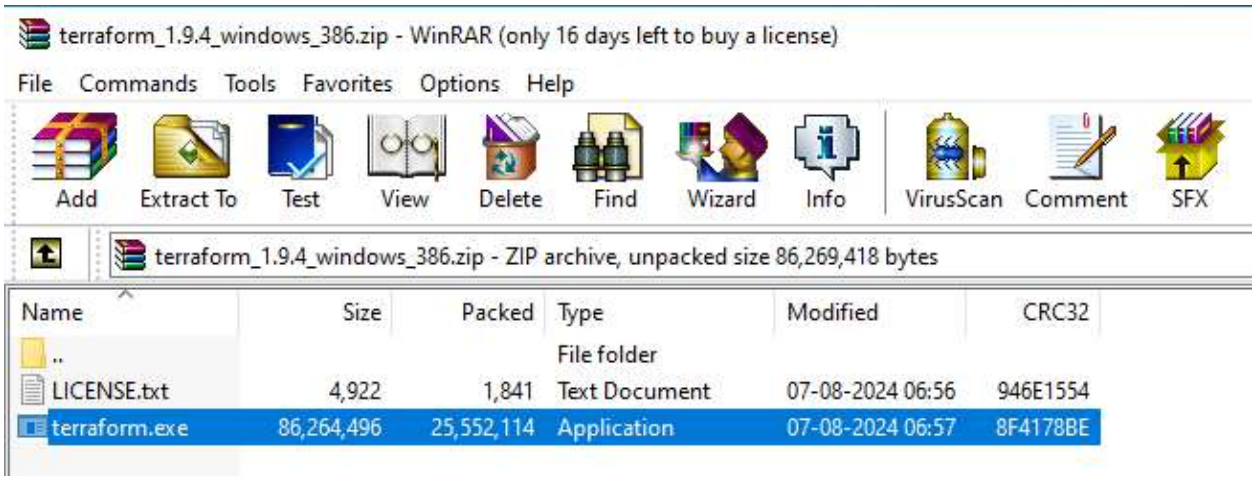
EXPERIMENT NO. 05

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

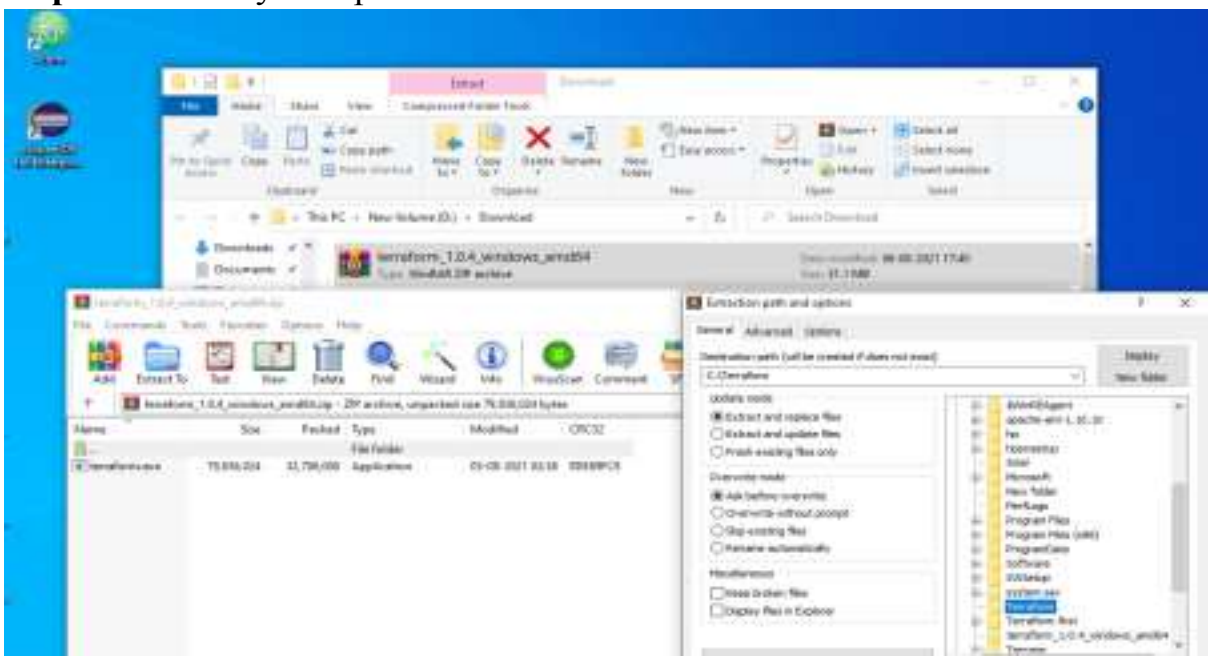
A) Installation and Configuration of Terraform in Windows
Step 1: Download terraform
To install Terraform, First Download the Terraform Cli Utility for windows from terraforms official website
website:https://www.terraform.io/downloads.html
Select the Operating System Windows followed by either 32bit or 64 bit based on your OS type.

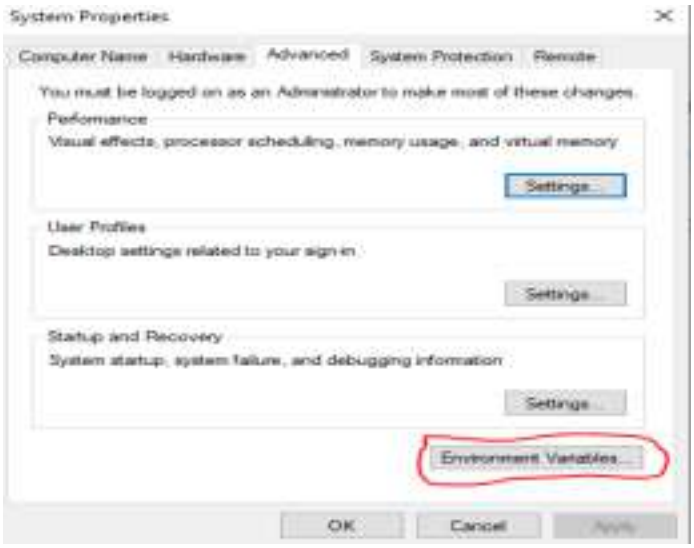


Step 2: Extract the downloaded setup file Terraform.exe in C:\Terraform directory



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





Step 5 : Open Terraform in PowerShell and check its functionality

```
C:\Users\student>terraform -version
Terraform v1.9.4
on windows_386
```

Name:Krushikesh Shelar D15-C Roll No:51

```
C:\Users\student>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
```

Aim:Creating docker image using terraform

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

Step 1: Check the docker functionality

```
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air Docker % docker -v
Docker version 27.0.3, build 7d4bcd8
```

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 nano 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 = "unix:///var/run/docker.sock"
}
# Pulls the image
resource "docker_image" "ubuntu" {
  name = "ubuntu:latest"
}
# Create a container
resource "docker_container" "foo" {
  image = docker_image.ubuntu.image_id
  name  = "foo"
  command = ["/bin/bash", "-c", "while true; do sleep 3600; done"]
}
```

```
  UW PICO 5.09

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

provider "docker" {
  host = "unix:///var/run/docker.sock"
}

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

# Create a container
resource "docker_container" "foo" {
  image = docker_image.ubuntu.image_id
  name  = "foo"
  command = ["/bin/bash", "-c", "while true; do sleep 3600; done"]
}
```

Step 3: Execute Terraform Init command to initialize the resources:

```
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air 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

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

Terraform will perform the following actions:

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

      + healthcheck (known after apply)

      + labels (known after apply)
    }

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

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

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

```
Plan: 1 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_container.foo: Creating...
docker_container.foo: Creation complete after 0s [id=f2b095b9576b22cc90eaae6860991144a11cc4a1255f9e1c4c99e5f4ca070857]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air Docker %
```

Docker images, After Executing Apply step:

```
docker_image.ubuntu: Still creating... [10s elapsed]
docker_image.ubuntu: Creation complete after 12s [id=sha256:1a799365aa63eed3c0ebb1c01aa5fd9d90320c46fe52938b03fb007d530d
docker_container.foo: Creating...
docker_container.foo: Creation complete after 1s [id=d2fa174ea2233d74813507a43ac8bd8136f623420da4ea251f0e43503b63446c]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air Docker % docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    1a799365aa63   3 weeks ago   101MB
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air Docker % docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED              STATUS              PORTS     NAMES
d2fa174ea223   1a799365aa63   "/bin/bash -c 'while…"   About a minute ago   Up About a minute             foo
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air Docker %
```

Docker Validate and Docker providers:

```
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air Docker % terraform validate
Success! The configuration is valid.

(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air Docker % terraform providers

Providers required by configuration:
.
└── provider[registry.terraform.io/kreuzwerker/docker] 2.21.0

Providers required by state:

    provider[registry.terraform.io/kreuzwerker/docker]

(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air Docker %
```

Step 6: Execute Terraform destroy to delete the configuration, which will automatically delete the Ubuntu Container.
Docker images After Executing Destroy step

```
  # docker_image.ubuntu will be destroyed
  - resource "docker_image" "ubuntu" {
      - id          = "sha256:1a799365aa63eed3c0ebb1c01aa5fd9d90320c46fe52938b03fb007d530d8b02ubuntu:latest" -> null
      - image_id    = "sha256:1a799365aa63eed3c0ebb1c01aa5fd9d90320c46fe52938b03fb007d530d8b02" -> null
      - latest      = "sha256:1a799365aa63eed3c0ebb1c01aa5fd9d90320c46fe52938b03fb007d530d8b02" -> 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=d2fa174ea2233d74813507a43ac8bd8136f623420da4ea251f0e43503b63446c]
docker_container.foo: Destruction complete after 1s
docker_image.ubuntu: Destroying... [id=sha256:1a799365aa63eed3c0ebb1c01aa5fd9d90320c46fe52938b03fb007d530d8b02ubuntu:latest]
docker_image.ubuntu: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air Docker % docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air Docker %
```

EXP 07:

Prerequisites:

1) Docker
   Run docker -v command.

```
Last login: Sat Sep 14 19:35:44 on ttys004
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air ~ % docker -v
Docker version 27.0.3, build 7d4bcd8
```

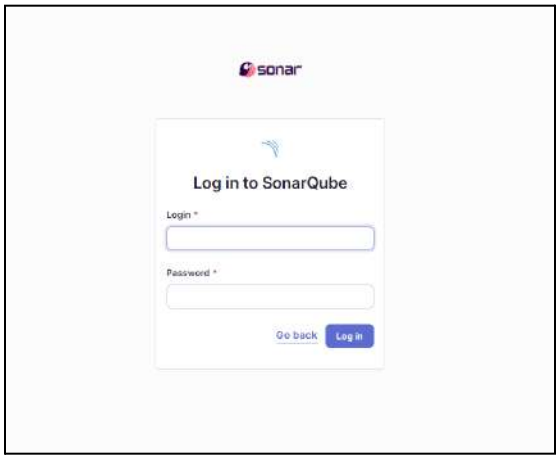
2) Install sonarqube image
   Command: docker pull sonarqube

```
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air ~ % docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
4be1db8bbbeb: Pull complete
8cc429601029: Pull complete
f3f704211ab9: Pull complete
cbee39a89b4f: Pull complete
5d25eb3700d3: Pull complete
3098d7b8f6ca: Pull complete
b5ea2a39b0fb: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview sonarqube
```

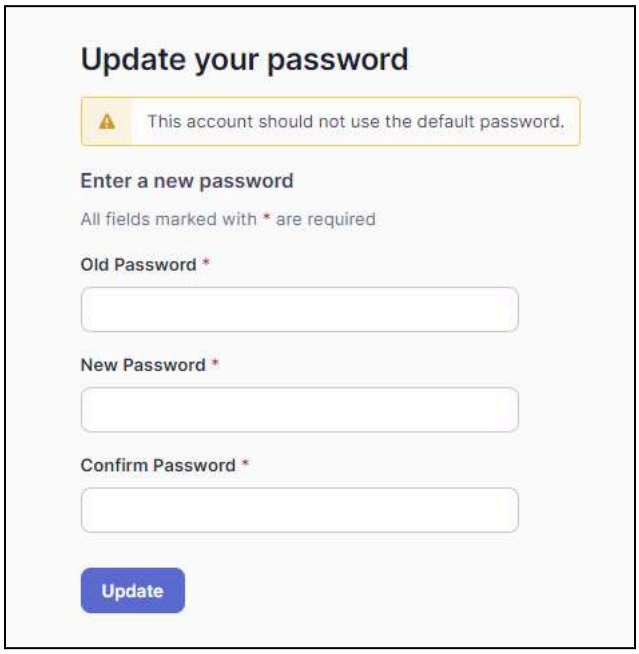
3) Run sonarqube image
   docker run -d --name sonarqube -e
   SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

```
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air ~ % docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

4d9bb0253ea4b9a05f65cbf91cd1f09286915bca2c999daf42dc1088157ba855
```

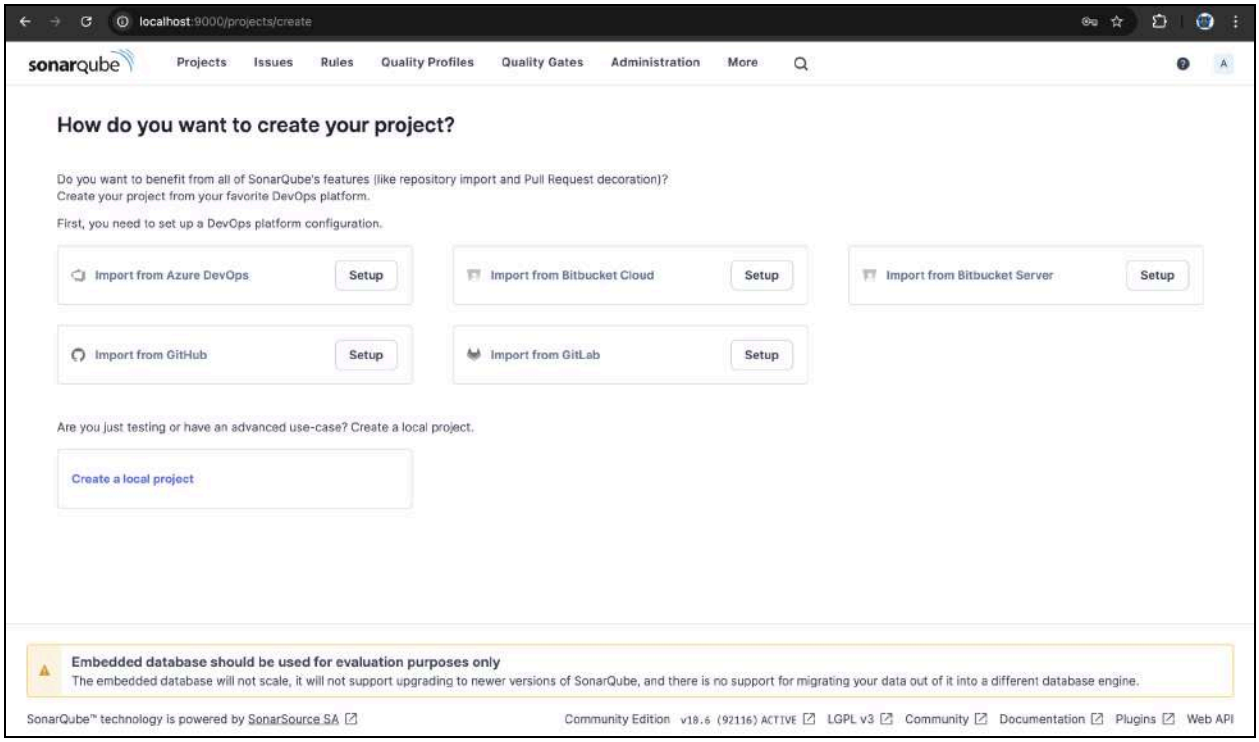
4) Run localhost:9000

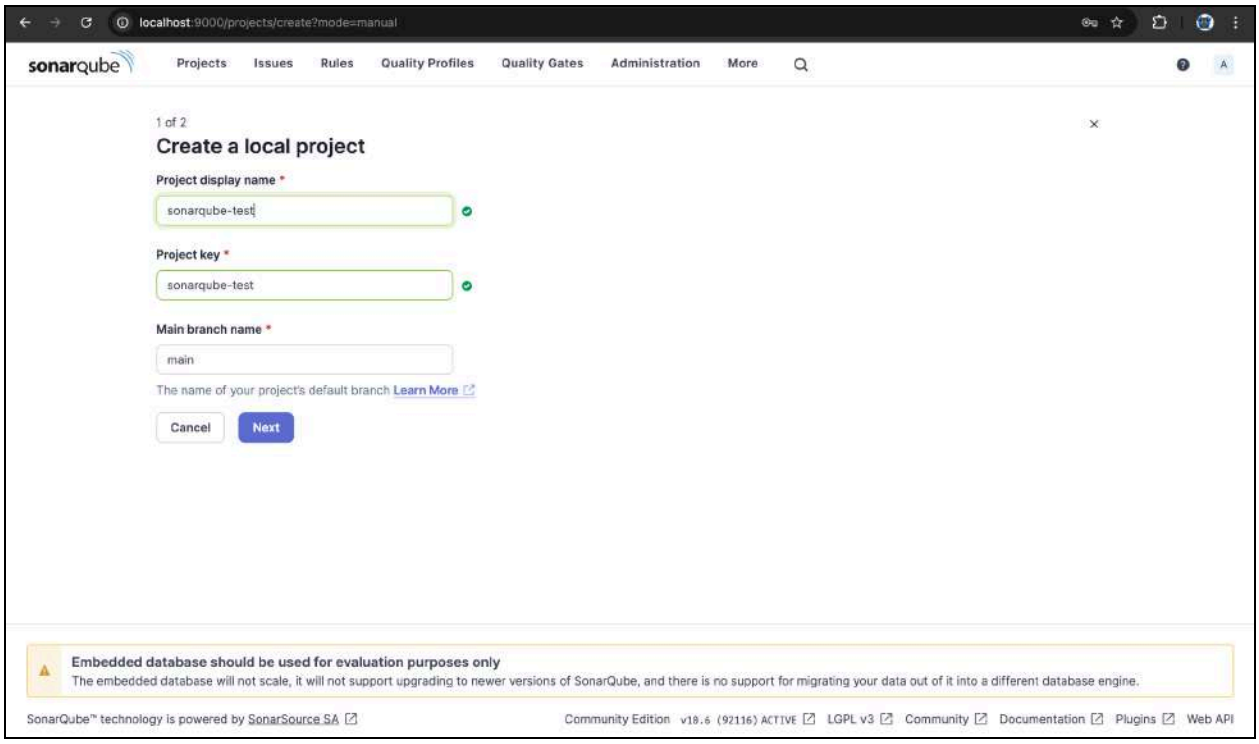


5) Login using username=”admin”, password=”admin”. It will promt you to set a new password.

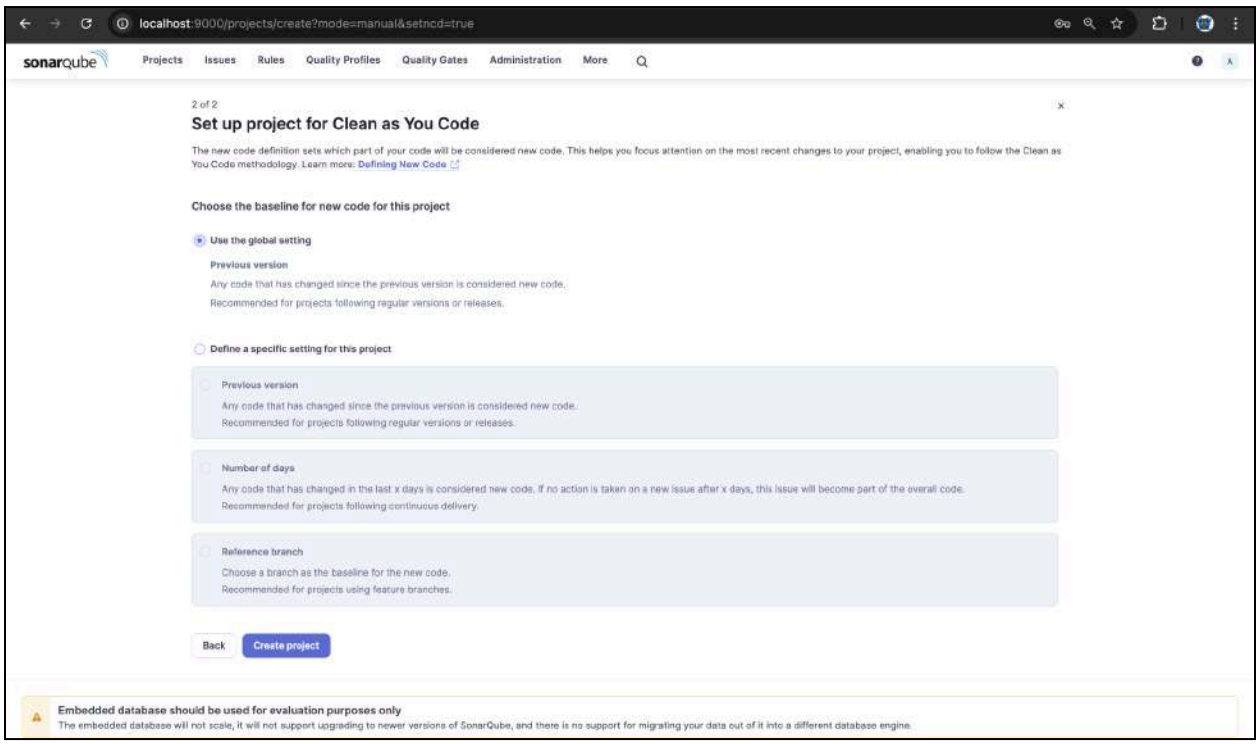


6) This is the interface. Create a local project with name sonarqube-test.

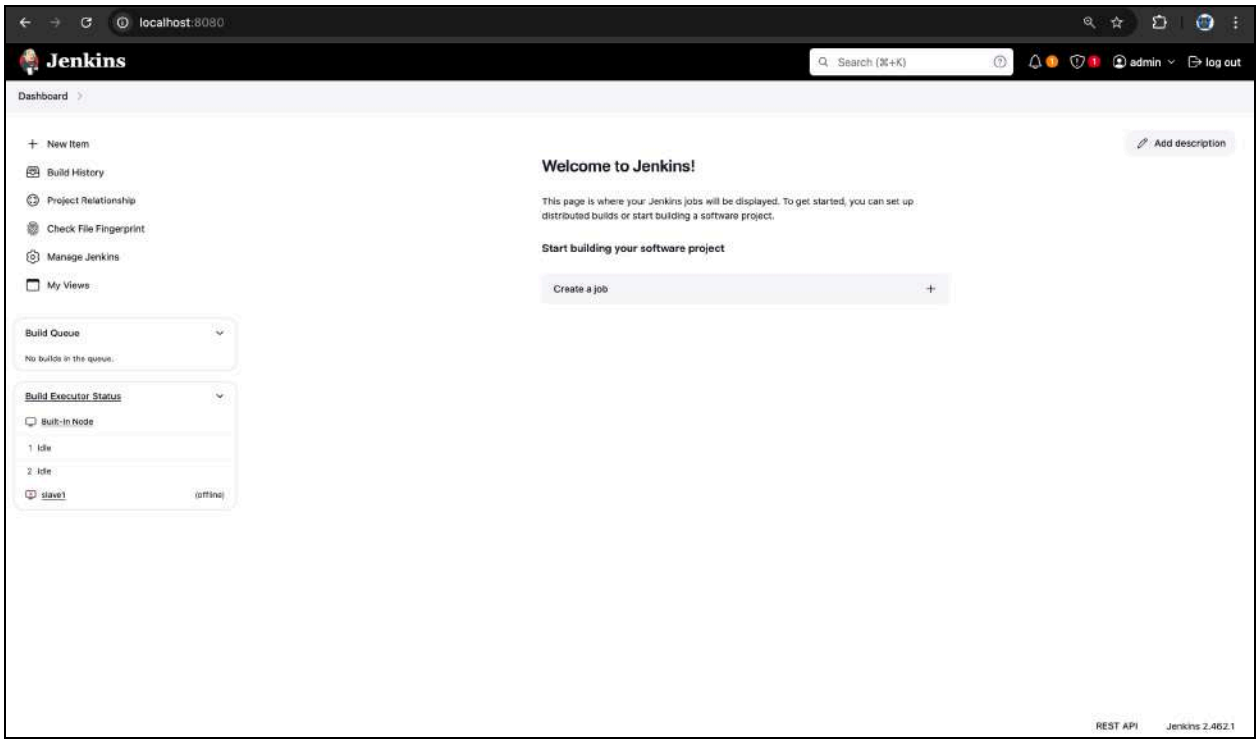




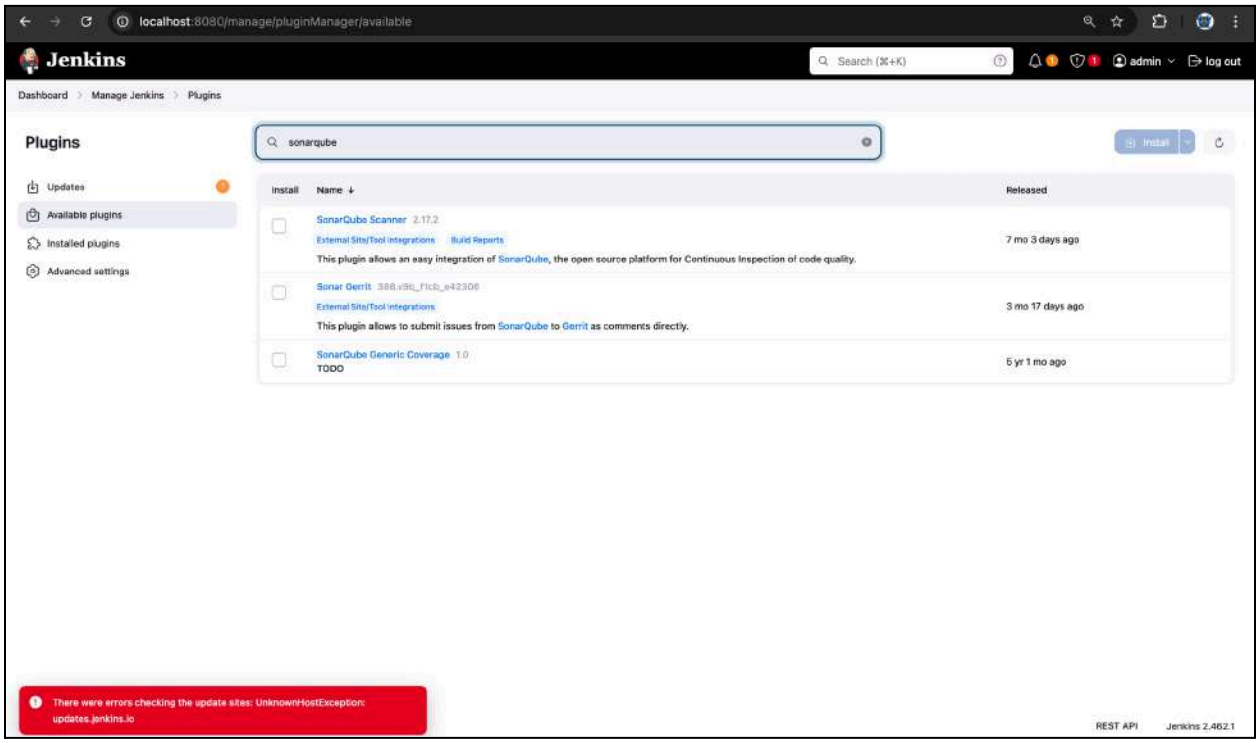
Set up the project as required and click on create.



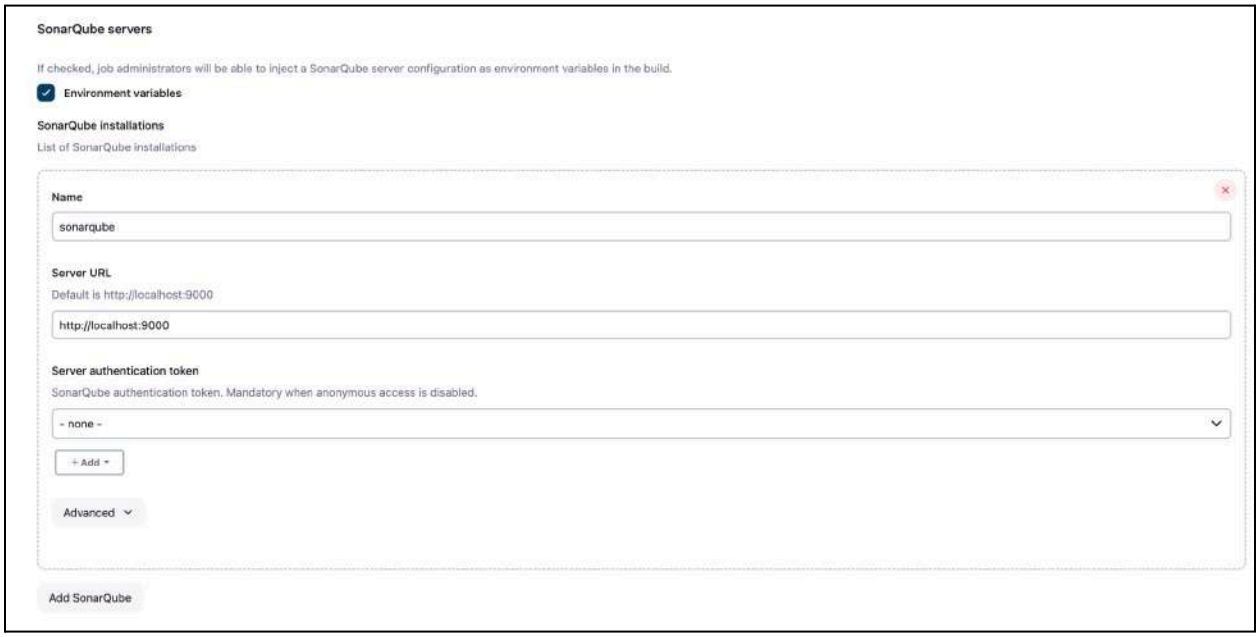
7) Open jenkins dashboard using localhost on whichever port it is hosted.



8) Go to manage jenkins → Search for Sonarqube Scanner for Jenkins and install it.

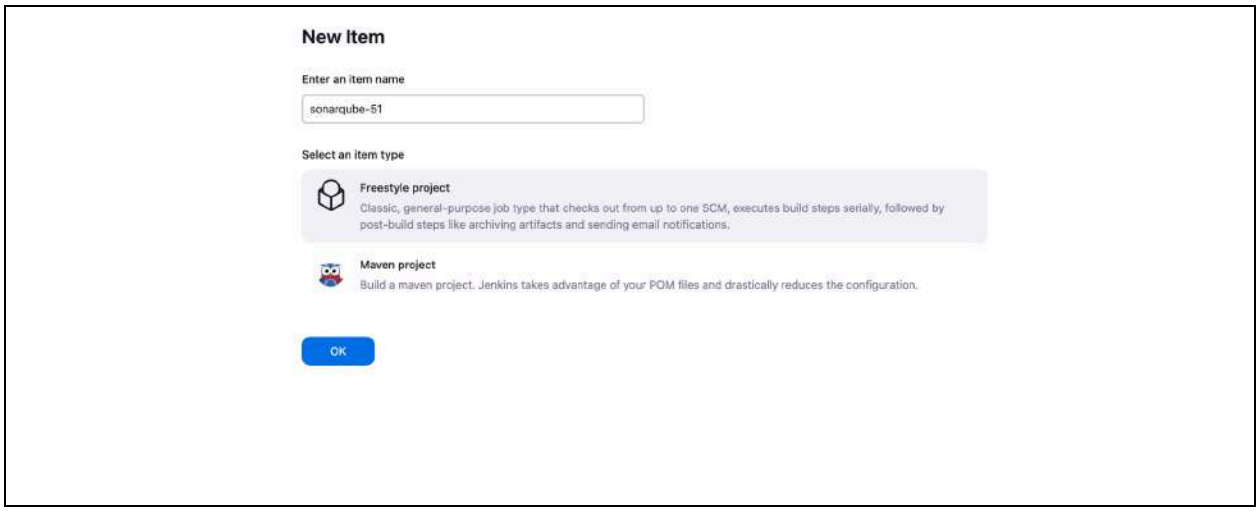


9) Now, go to Manage Jenkins → System. Under Sonarqube servers, add a server. Add server authentication token if needed.

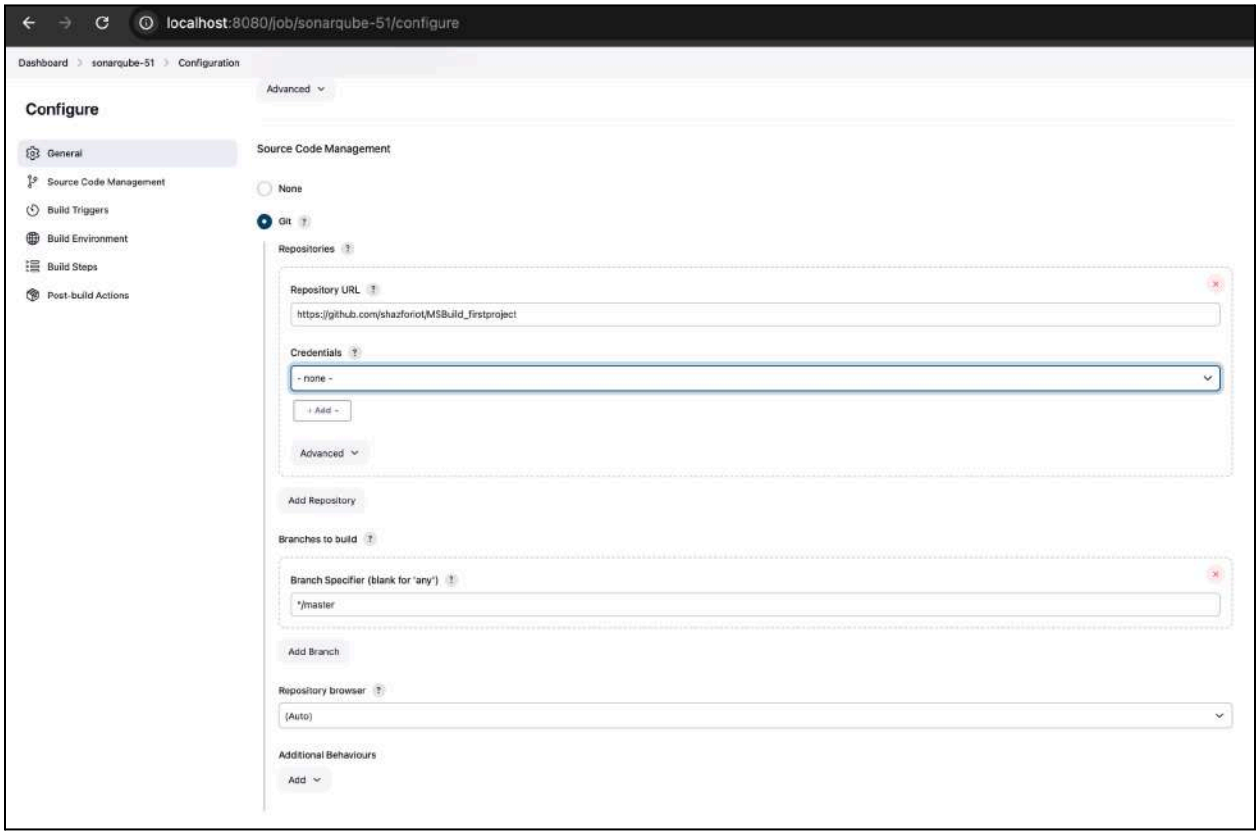


10) Go to Manage Jenkins → Tools. Go to SonarQube scanner, choose the latest configuration and choose install automatically.

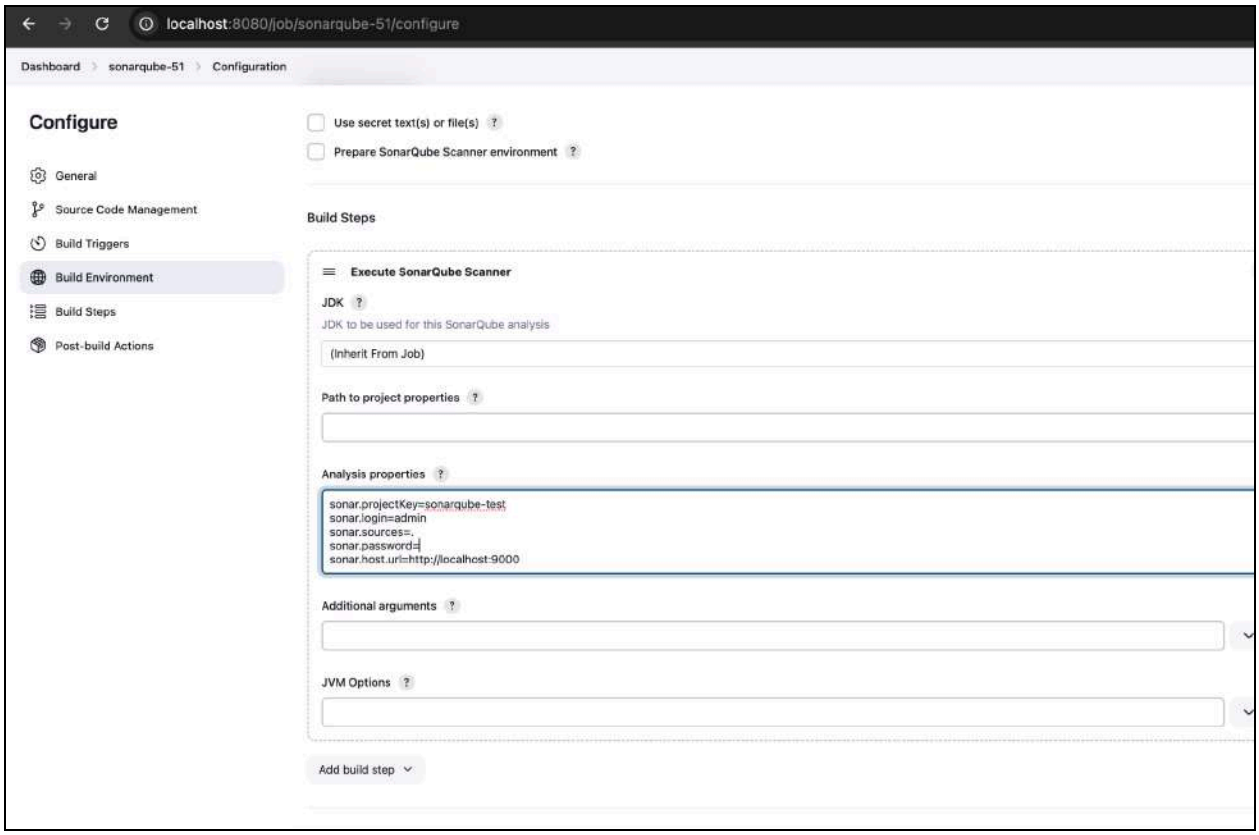
11) After configuration, create a New Item → choose a freestyle project.



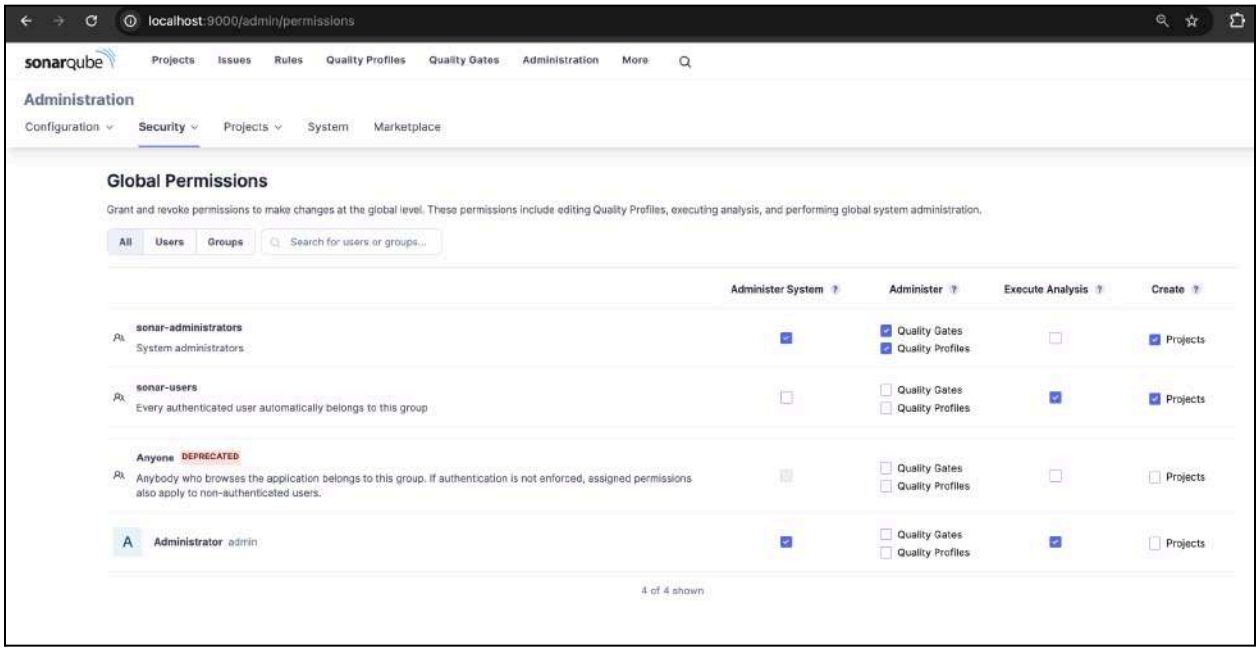
12) Use this github repository in Source Code Management.
https://github.com/shazforiot/MSBuild_firstproject
It is a sample hello-world project with no vulnerabilities.



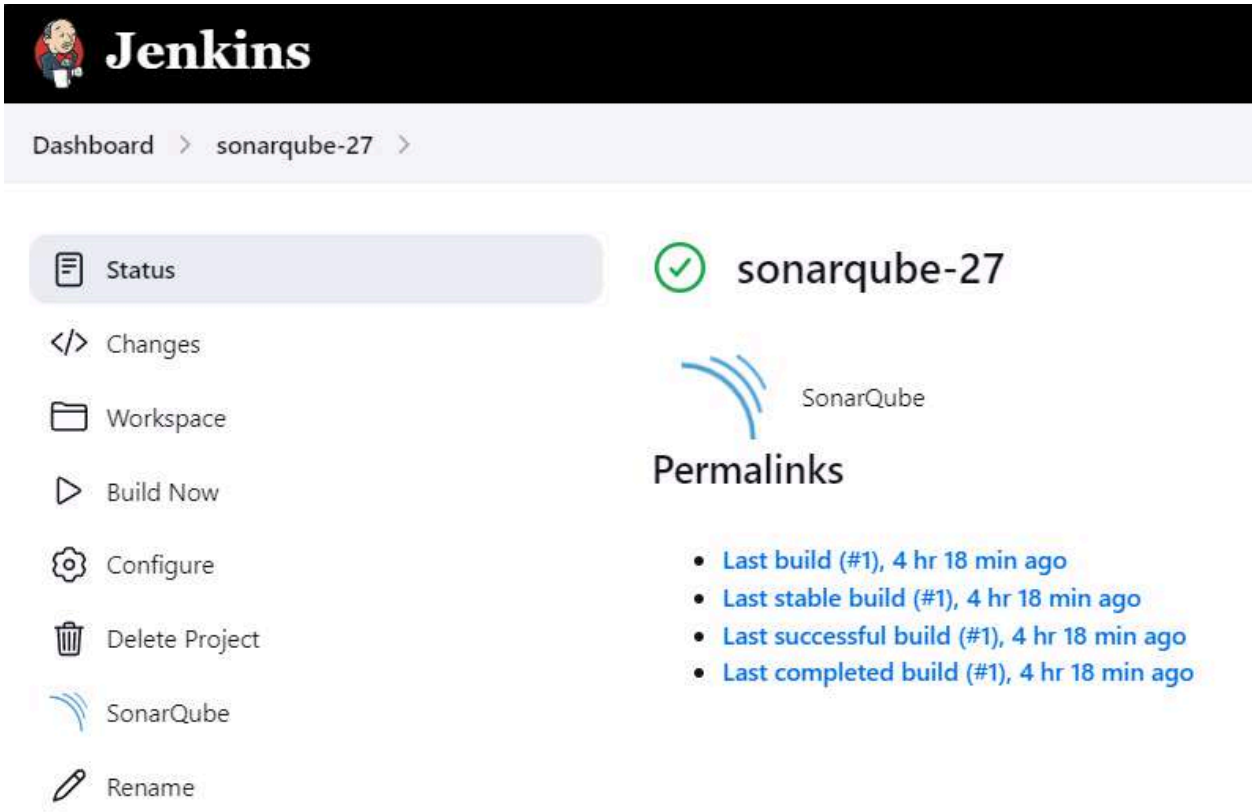
13) Under Build Steps, enter Sonarqube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.



14) Go to http://localhost:9000/<user_name>/permissions and allow Execute Permissions to admin user.



15) Run the Jenkins build.



Check the console Output

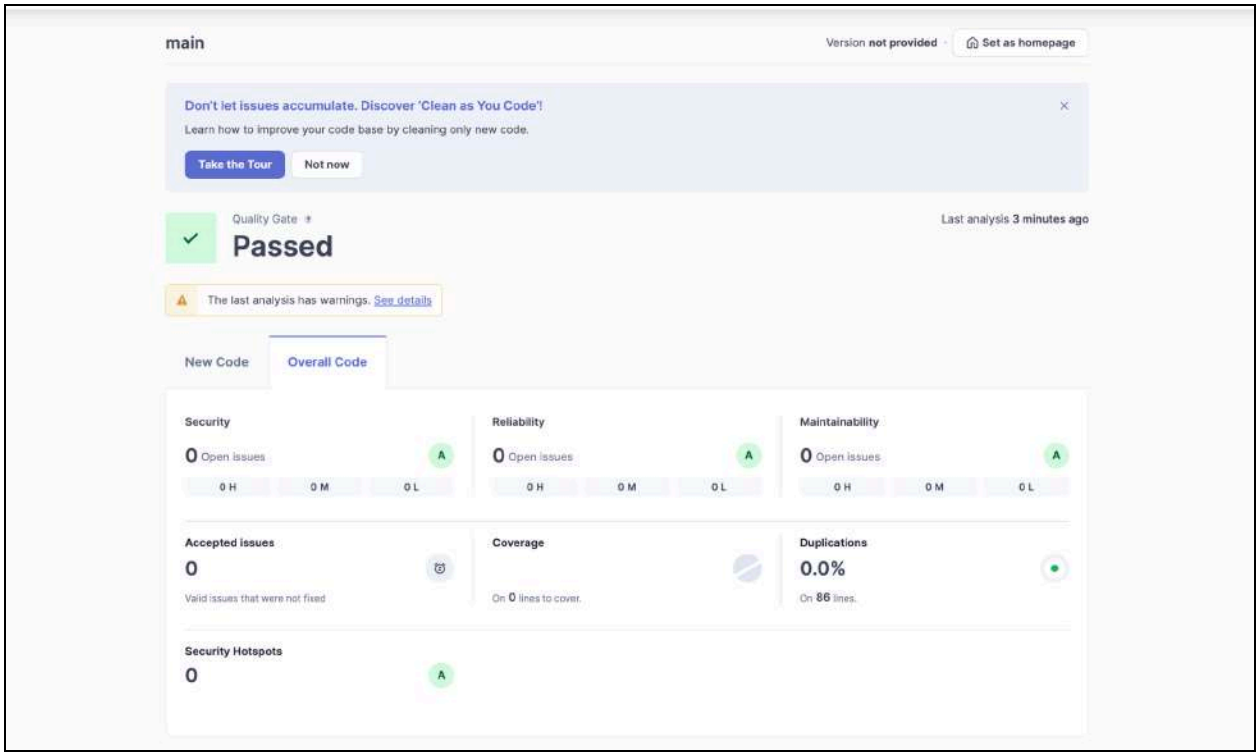
Console Output

Download Copy View as plain text

```
Started by user admin
Running as SYSTEM
Building on the built-in node in workspace /Users/krushikeshsunilshelar/.jenkins/workspace/sonarqube-51
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/shazforiot/MSBuild_firstproject
 > git init /Users/krushikeshsunilshelar/.jenkins/workspace/sonarqube-51 # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject
 > git --version # timeout=10
 > git --version # 'git version 2.39.3 (Apple Git-146)'
 > git fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
Commit message: "updated"
First time build. Skipping changelog.
Unpacking https://repo1.maven.org/maven2/org/sonarsource/scanner/cli/sonar-scanner-cli/6.2.0.4584/sonar-scanner-cli-6.2.0.4584.zip to
/Users/krushikeshsunilshelar/.jenkins/tools/hudson.plugins.sonar.SonarRunnerInstallation/sonarqube-scanner on Jenkins
[sonarqube-51] $ /Users/krushikeshsunilshelar/.jenkins/tools/hudson.plugins.sonar.SonarRunnerInstallation/sonarqube-scanner/bin/sonar-scanner -Dsonar.host.url=http://localhost:9000 -
Dsonar.projectKey=sonarqube-test -Dsonar.login=admin -Dsonar.host.url=http://localhost:9000 -Dsonar.sources=. -Dsonar.password=Rushi@1418 -
Dsonar.projectBaseDir=/Users/krushikeshsunilshelar/.jenkins/workspace/sonarqube-51
10:15:43.472 WARN  Property 'sonar.host.url' with value 'http://localhost:9000' is overridden with value 'http://localhost:9000'
10:15:43.475 INFO  Scanner configuration file: /Users/krushikeshsunilshelar/.jenkins/tools/hudson.plugins.sonar.SonarRunnerInstallation/sonarqube-scanner/conf/sonar-scanner.properties
10:15:43.476 INFO  Project root configuration file: NONE
10:15:43.486 INFO  SonarScanner CLI 6.2.0.4584
10:15:43.486 INFO  Java 17.0.8 Oracle Corporation (64-bit)
10:15:43.487 INFO  Mac OS X 14.6.1 aarch64
10:15:43.525 INFO  User cache: /Users/krushikeshsunilshelar/.sonar/cache
10:15:44.070 INFO  JRE provisioning: os[macos], arch[arm64]
10:15:47.173 INFO  Communicating with SonarQube Server 10.6.0.92116
10:15:47.342 INFO  Starting SonarScanner Engine...
10:15:47.342 INFO  Java 17.0.11 Eclipse Adoptium (64-bit)
10:15:47.691 INFO  Load global settings
```

```
10:16:13.003 INFO  Sensor IaC Docker Sensor [iac] (done) | time=11ms
10:16:13.003 INFO  Sensor TextAndSecretsSensor [text]
10:16:13.003 INFO  Available processors: 8
10:16:13.003 INFO  Using 8 threads for analysis.
10:16:13.164 INFO  The property "sonar.tests" is not set. To improve the analysis accuracy, we categorize a file as a test file if any of the following is true:
  * The filename starts with "test"
  * The filename contains "test." or "tests."
  * Any directory in the file path is named: "doc", "docs", "test" or "tests"
  * Any directory in the file path has a name ending in "test" or "tests"

10:16:13.182 INFO  Using git CLI to retrieve untracked files
10:16:13.196 INFO  Analyzing language associated files and files included via "sonar.text.inclusions" that are tracked by git
10:16:13.219 INFO  14 source files to be analyzed
10:16:13.347 INFO  14/14 source files have been analyzed
10:16:13.347 INFO  Sensor TextAndSecretsSensor [text] (done) | time=344ms
10:16:13.349 INFO  ------------- Run sensors on project
10:16:13.372 INFO  Sensor C# [csharp]
10:16:13.372 WARN  Your project contains C# files which cannot be analyzed with the scanner you are using. To analyze C# or VB.NET, you must use the SonarScanner for .NET 5.x or higher, see
https://redirect.sonarsource.com/doc/install-configure-scanner-msbuild.html
10:16:13.372 INFO  Sensor C# [csharp] (done) | time=1ms
10:16:13.372 INFO  Sensor Analysis Warnings import [csharp]
10:16:13.372 INFO  Sensor Analysis Warnings import [csharp] (done) | time=0ms
10:16:13.372 INFO  Sensor C# File Caching Sensor [csharp]
10:16:13.372 WARN  Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir' property.
10:16:13.372 INFO  Sensor C# File Caching Sensor [csharp] (done) | time=0ms
10:16:13.372 INFO  Sensor Zero Coverage Sensor
10:16:13.374 INFO  Sensor Zero Coverage Sensor (done) | time=2ms
10:16:13.379 INFO  CPD Executor Calculating CPD for 0 files
10:16:13.379 INFO  CPD Executor CPD calculation finished (done) | time=0ms
10:16:13.380 INFO  SCM revision ID 'f2bc042c04c6e72427c380bcaee6d6fee7b49adf'
10:16:13.411 INFO  Analysis report generated in 27ms, dir size=200.4 kB
10:16:13.421 INFO  Analysis report compressed in 9ms, zip size=22.1 kB
10:16:13.600 INFO  Analysis report uploaded in 179ms
10:16:13.601 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube-test
10:16:13.601 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
10:16:13.601 INFO  More about the report processing at http://localhost:9000/api/ce/task?id=5f495a8e-aea7-4beb-9bd3-e3faa2fd4347
10:16:13.607 INFO  Analysis total time: 10.105 s
10:16:13.607 INFO  SonarScanner Engine completed successfully
10:16:13.653 INFO  EXECUTION SUCCESS
10:16:13.655 INFO  Total time: 12.133s
Finished: SUCCESS
```

16) Once the build is complete, check project on SonarQube



Conclusion:

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

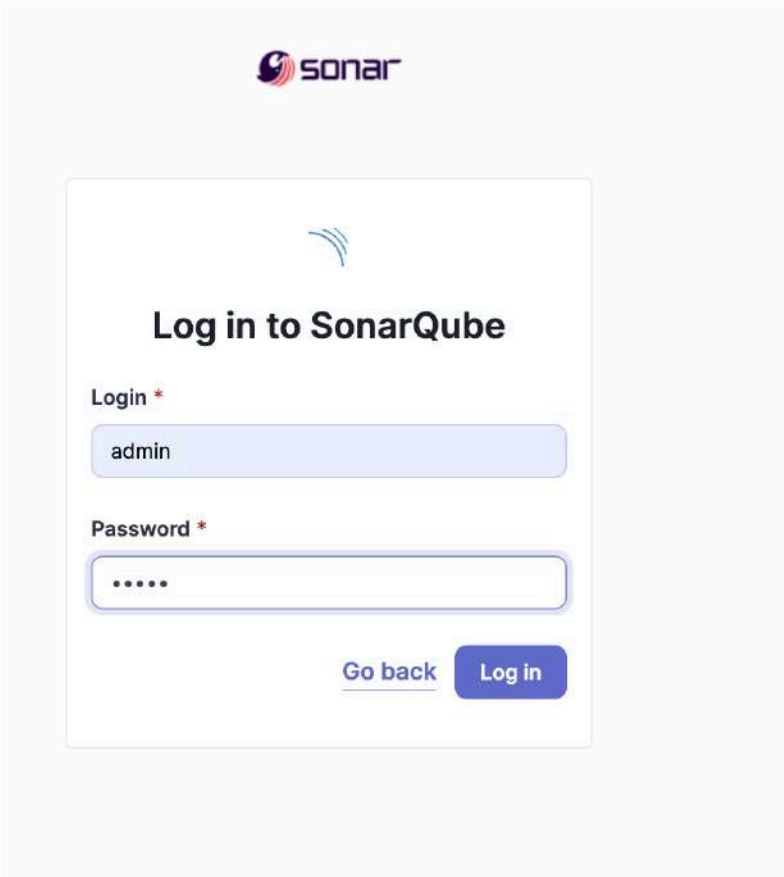
08 Advanced DevOps Lab

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.

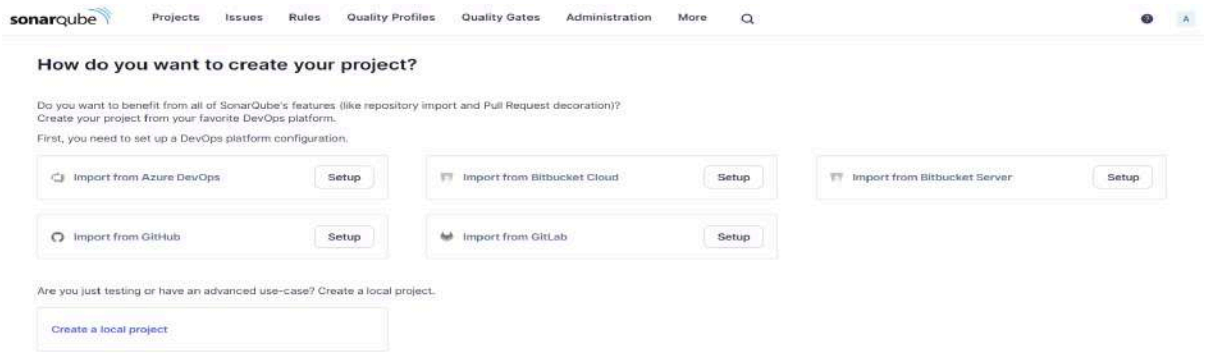
Step 1: Install sonarqube image

```
Last login: Thu Sep 26 10:20:34 on ttys000
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air ~ % 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
4be1db8bbbeb: Pull complete
8cc429601029: Pull complete
f3f704211ab9: Pull complete
cbee39a89b4f: Pull complete
5d25eb3700d3: Pull complete
3098d7b8f6ca: Pull complete
b5ea2a39b0fb: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
f9a6a5d47cdaa6ca0b3a21305bb19fff331592071b09b052d4122c601985660f
(base) krushikeshsunilshelar@Krushikeshs-MacBook-Air ~ %
```

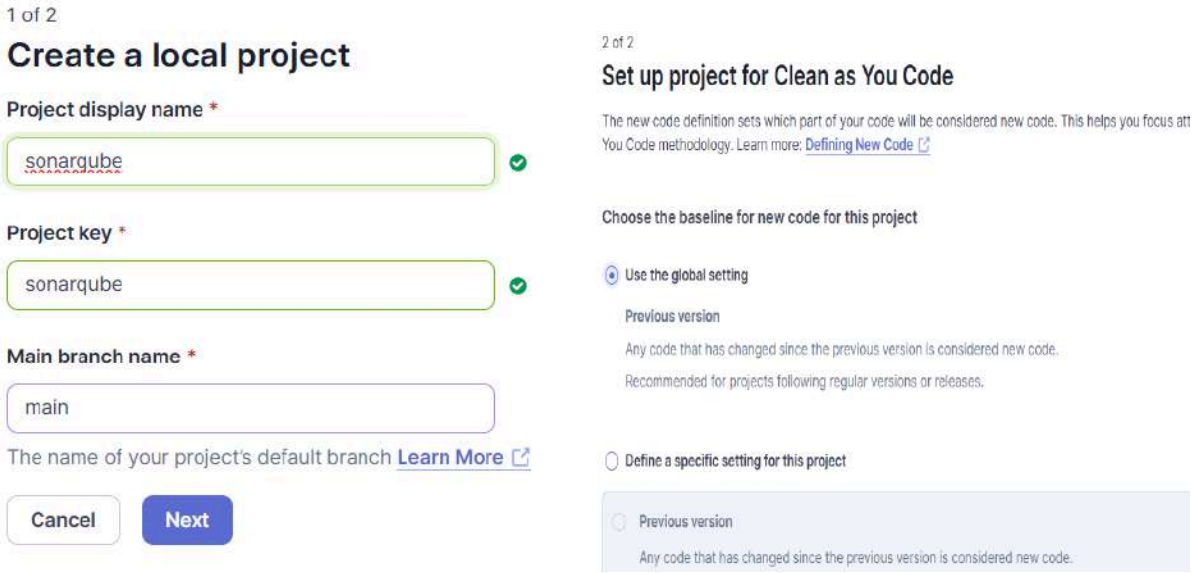
2. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.



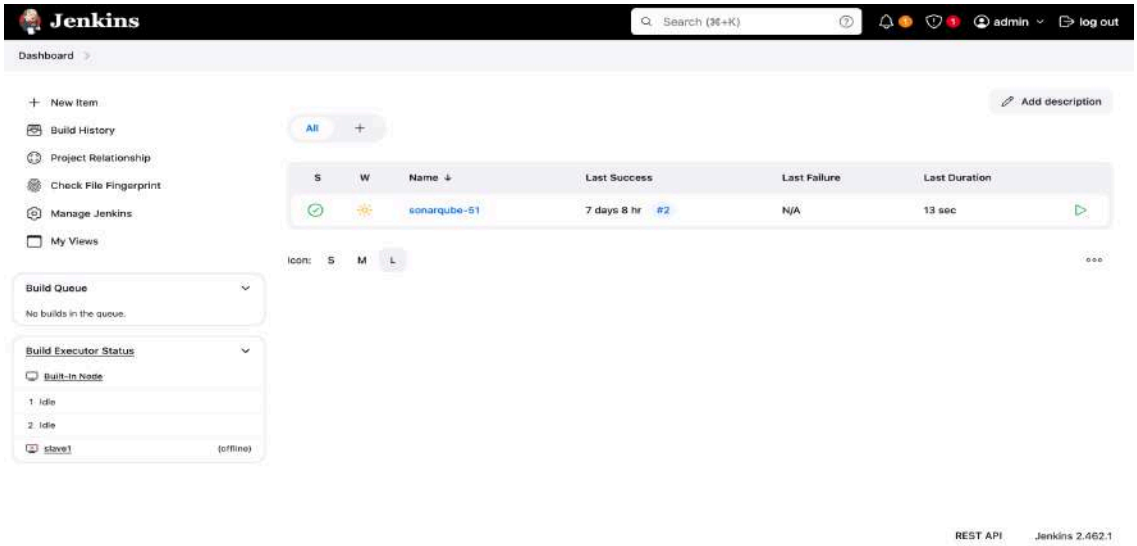
3. Login to SonarQube using username admin and password admin.



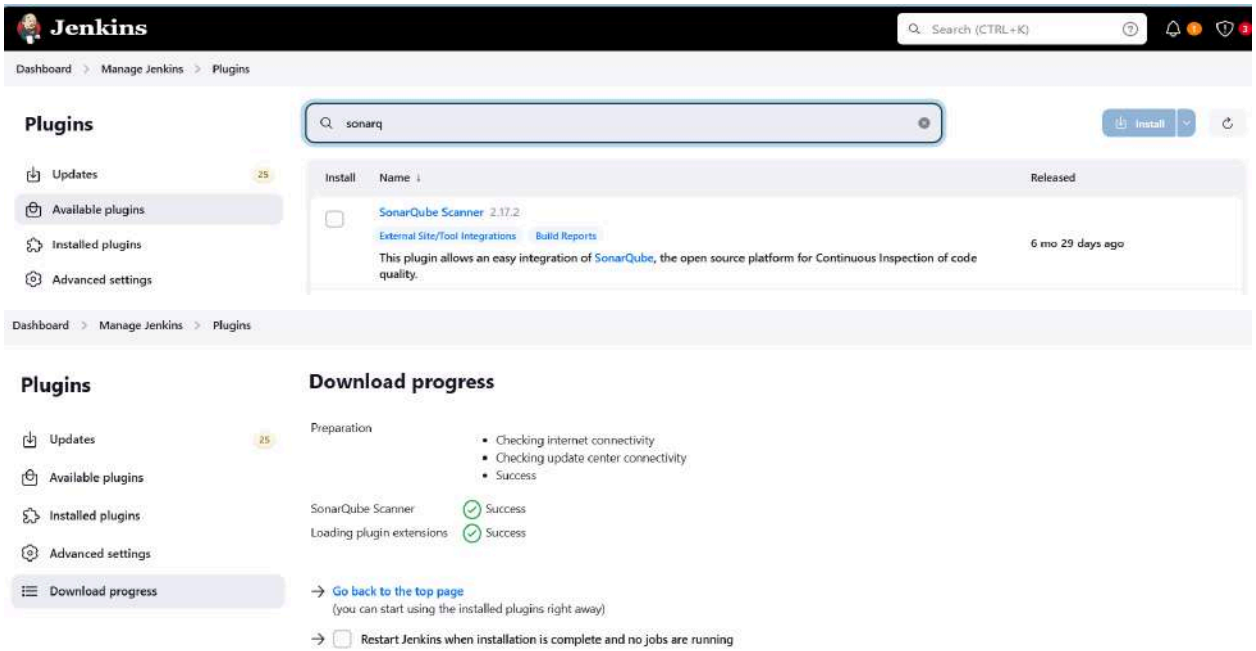
4. Create a manual project in SonarQube with the name sonarqube



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



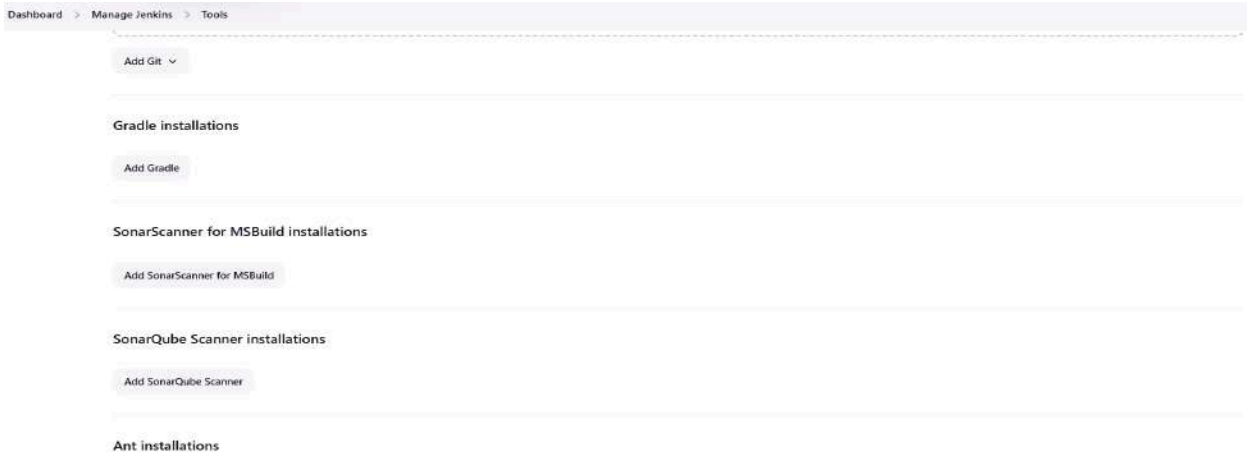
6. Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.



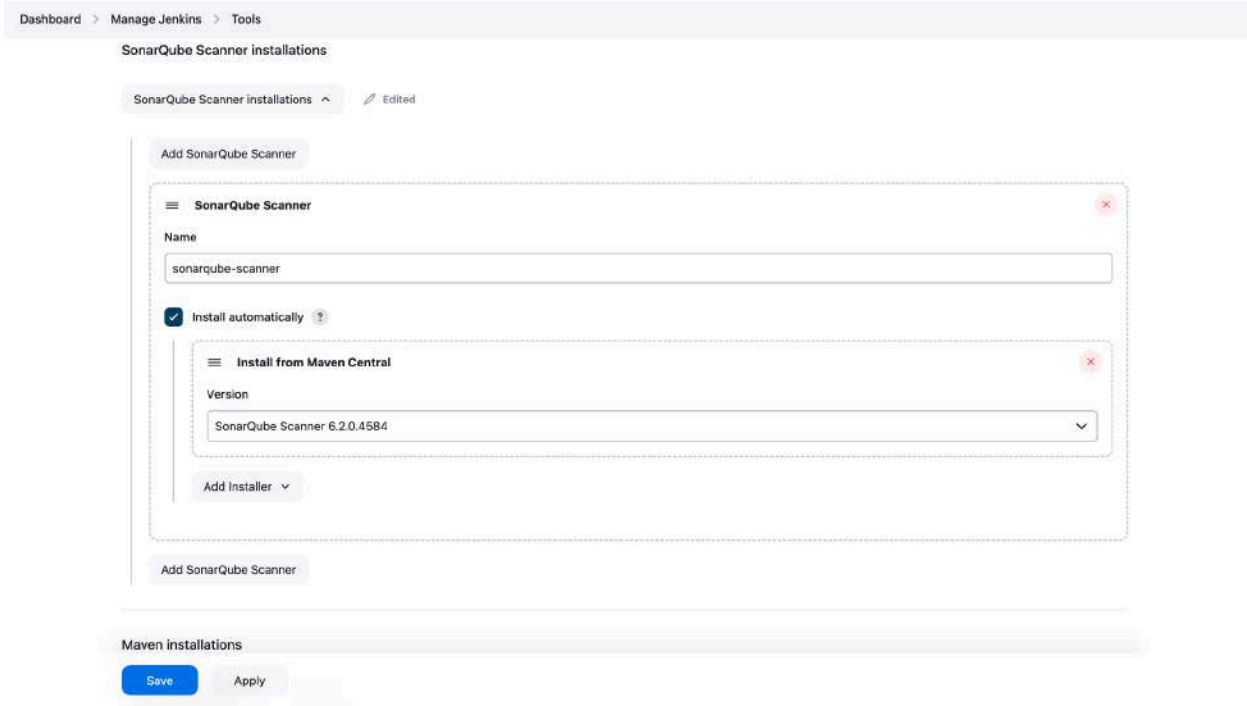
7. Under Jenkins 'Manage Jenkins' then go to 'system', scroll and look for **SonarQube Servers** and enter the details.
Enter the Server Authentication token if needed.

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

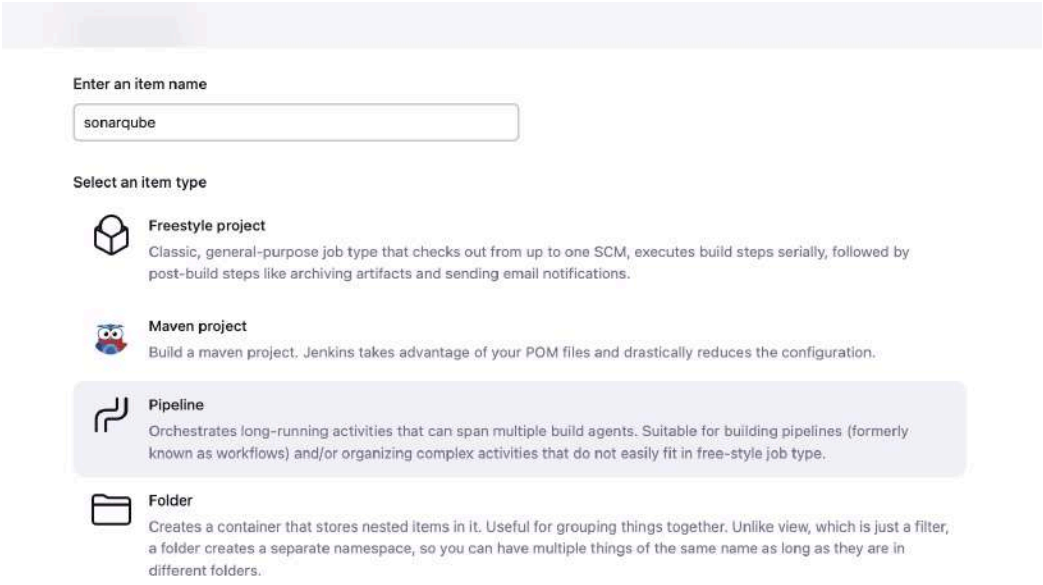
8. Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.
Dashboard > Manage Jenkins > Tools



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:

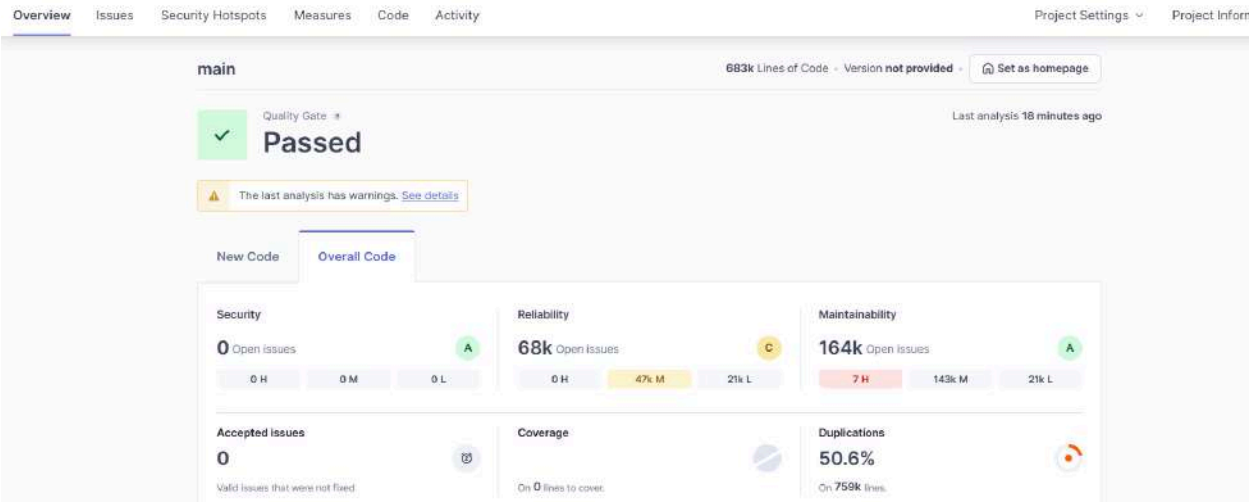
```
        node {
  stage('Cloning the GitHub Repo') {
     git 'https://github.com/shazforiot/GOL.git'
  }

  stage('SonarQube analysis') {
     withSonarQubeEnv('<Name_of_SonarQube_environment_on_Jenkins>') {
        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

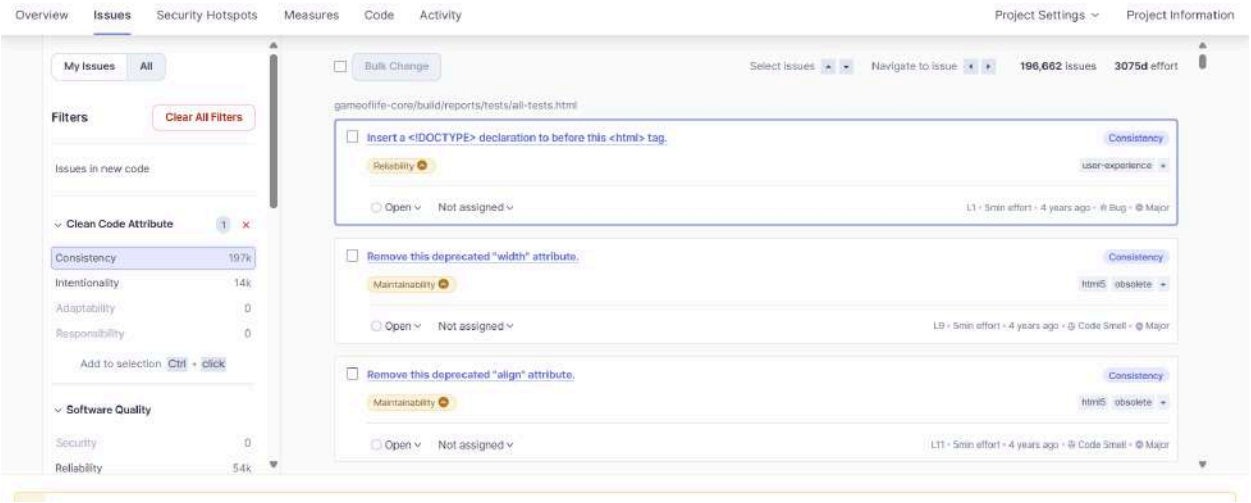
12. Check console

13. Now, check the project in SonarQube

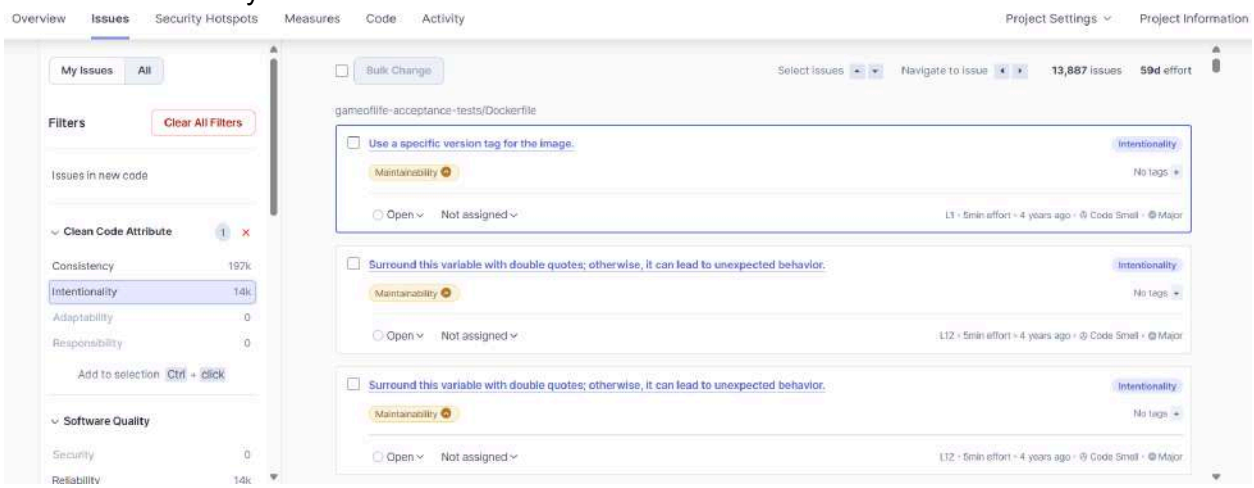


14. Code Problems

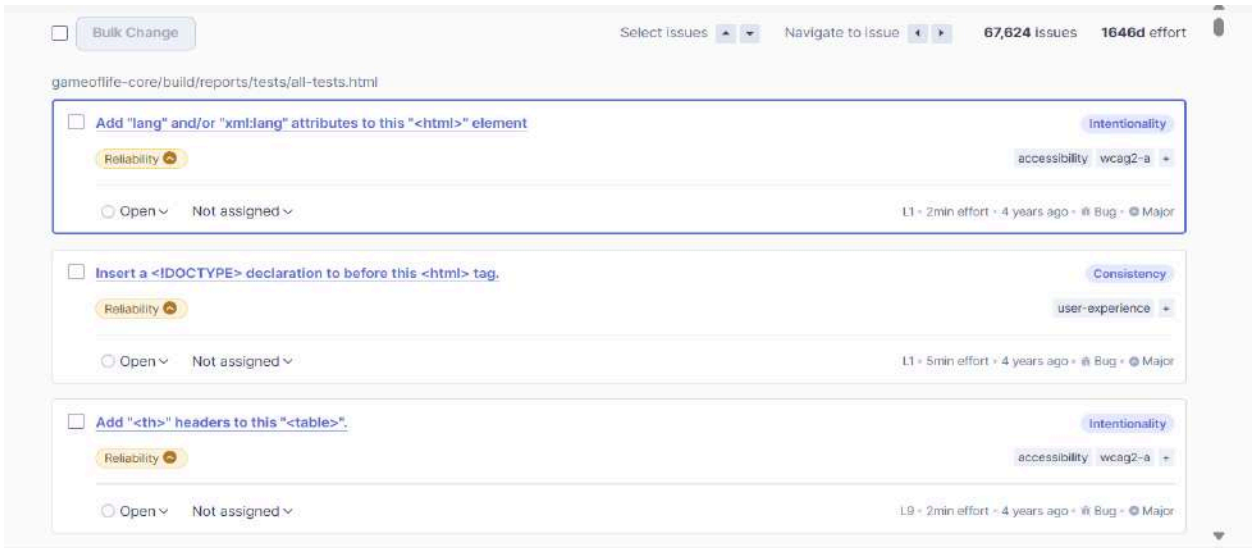
- Consistency



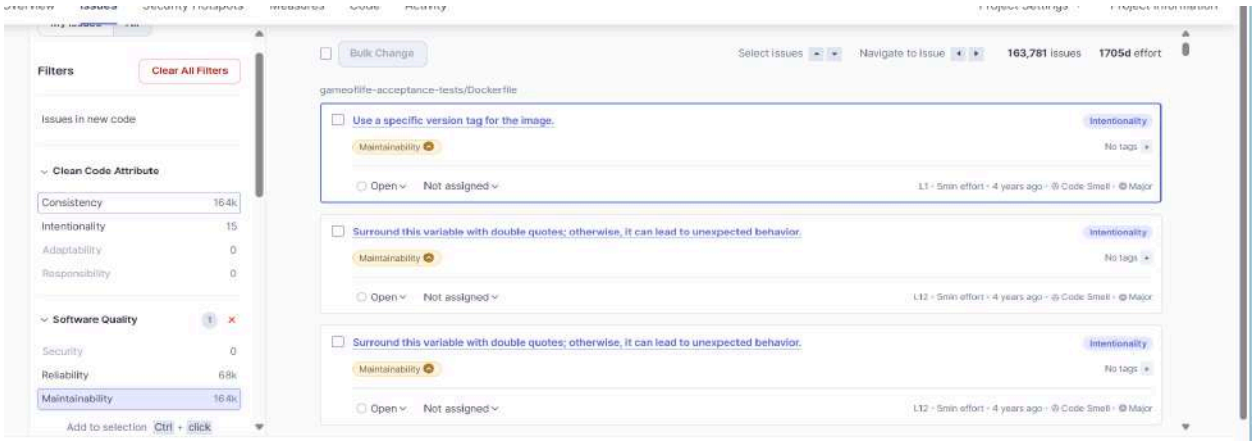
- Intentionality



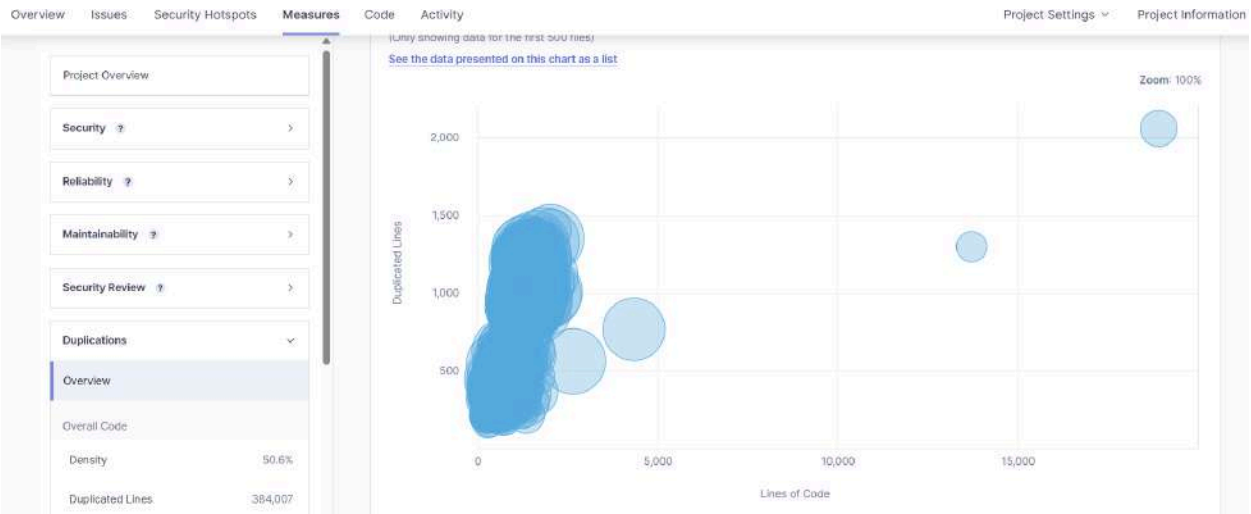
- Bugs



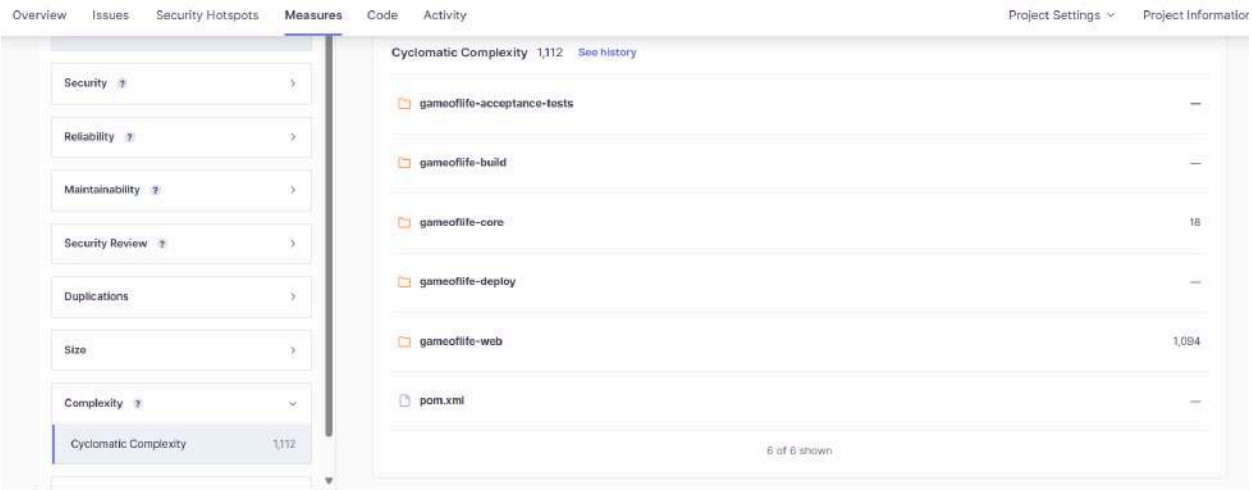
- Code Smells



- Duplications



- Cyclomatic Complexities



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

Conclusion:

In this experiment, we integrated Jenkins with SonarQube to automate code quality checks in our CI/CD pipeline. We deployed SonarQube using Docker, set up a project for analysis, and configured Jenkins with the SonarQube Scanner plugin. We then created a Jenkins pipeline to clone a GitHub repository and run SonarQube analysis on the code. This integration ensures continuous monitoring of code quality, helping to identify bugs, code smells, and security vulnerabilities throughout the development process.

Adv DevOps Practical 9

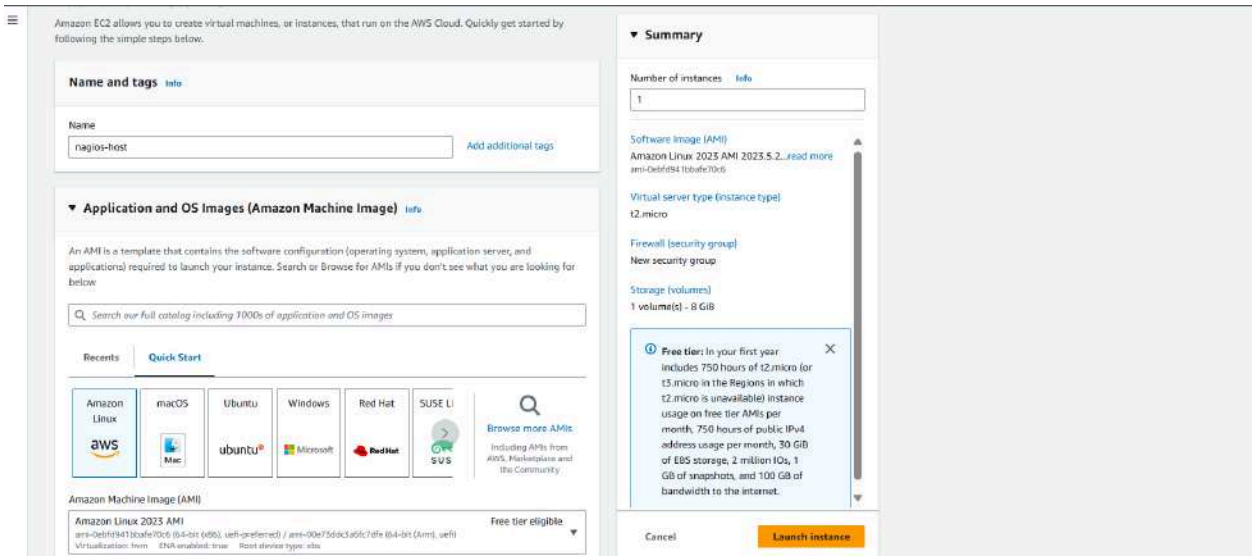
Aim: To Understand Continuous monitoring and Installation and configuration of Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine.

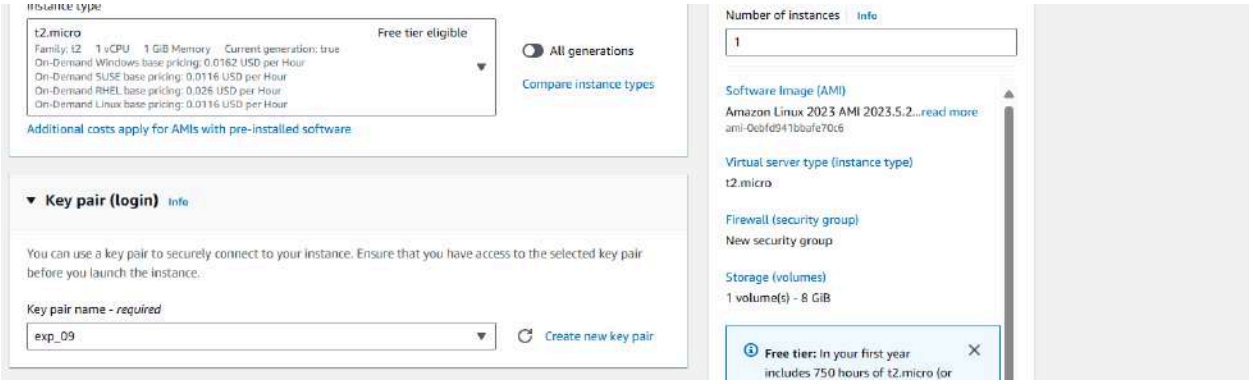
Theory:

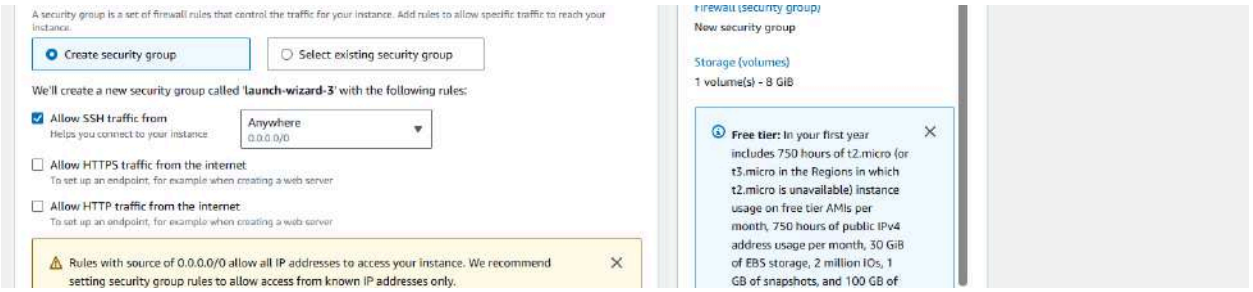
What is Nagios?
Nagios is an open-source software for continuous monitoring of systems, networks, and infrastructures. It runs plugins stored on a server that is connected with a host or another server on your network or the Internet. In case of any failure, Nagios alerts about the issues so that the technical team can perform the recovery process immediately.
Nagios is used for continuous monitoring of systems, applications, service and business processes in a DevOps culture
.
Installation of Nagios

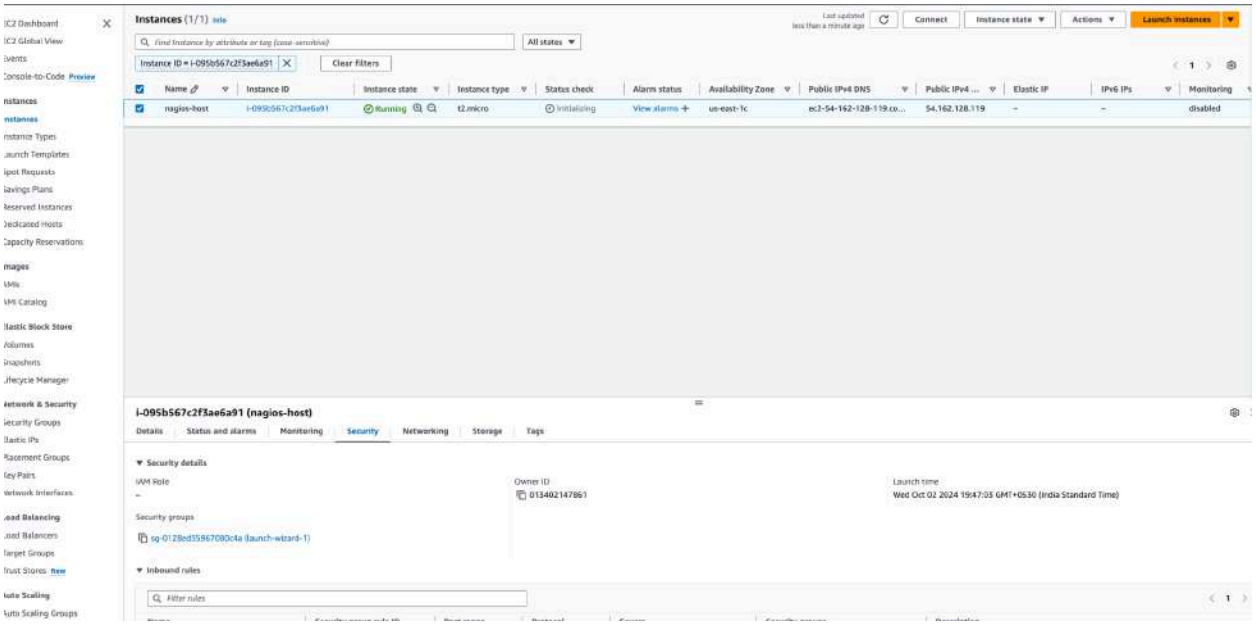
Prerequisites: AWS Free Tier

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

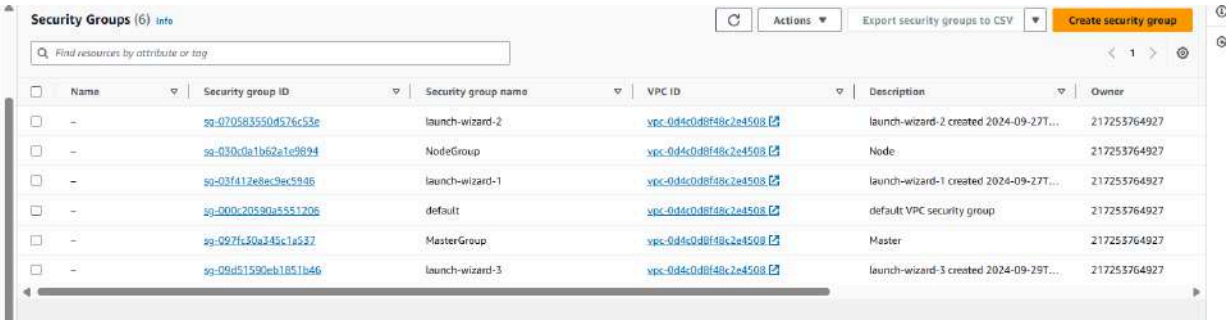








2. Under Security Group, make sure HTTP, HTTPS, SSH, ICMP are open from everywhere.



EC2 > Security Groups > sg-09d51590eb1851b46

sg-09d51590eb1851b46 - launch-wizard-3

Actions

Details

Security group name	Security group ID	Description	VPC ID
launch-wizard-3	sg-09d51590eb1851b46	launch-wizard-3 created 2024-09-29T06:49:51.498Z	vpc-0d4c0d8f48c2e4508
Owner	**Inbound rules count**	**Outbound rules count**	
217253764927	1 Permission entry	1 Permission entry	

Inbound rules | Outbound rules | Tags

Inbound rules (1)

Manage tags | Edit inbound rules

Search

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
-	sgr-0ec19557ab93305...	IPv4	SSH	TCP	22	0.0.0.0/0	-

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-0ec19557ab930565	SSH	TCP	22	Custom	0.0.0.0/0		Delete
-	HTTP	TCP	80	Anywhere-I...	0.0.0.0/0		Delete
-	All ICMP - IPv6	IPv6 ICMP	All	Anywhere-I...	0.0.0.0/0		Delete
-	HTTPS	TCP	443	Anywhere-I...	0.0.0.0/0		Delete
-	All traffic	All	All	Anywhere-I...	0.0.0.0/0		Delete
-	Custom TCP	TCP	5666	Anywhere-I...	0.0.0.0/0		Delete
-	All ICMP - IPv4	ICMP	All	Anywhere-I...	0.0.0.0/0		Delete

Add rule

Security group name	Security group ID	Description	VPC ID
launch-wizard-3	sg-09d51590eb1851b46	launch-wizard-3 created 2024-09-29T06:49:51.498Z	vpc-0d4c0d8f48c2e4508
Owner	**Inbound rules count**	**Outbound rules count**	
217253764927	7 Permission entries	1 Permission entry	

Inbound rules | Outbound rules | Tags

Inbound rules (7)

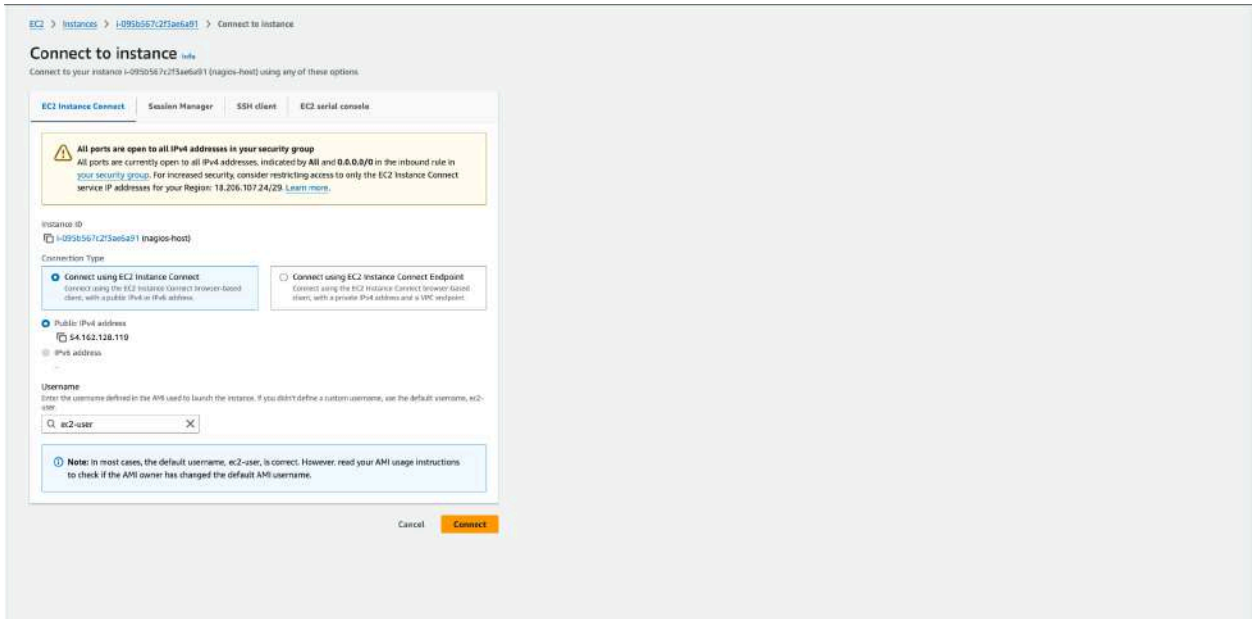
Manage tags | Edit inbound rules

Search

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
-	sgr-034c50eeff5e5fa00	IPv4	All ICMP - IPv6	IPv6 ICMP	All	0.0.0.0/0	-
-	sgr-038d0d3791dfcc60e	IPv4	HTTPS	TCP	443	0.0.0.0/0	-
-	sgr-0e8ad1dd008b14...	IPv4	All ICMP - IPv4	ICMP	All	0.0.0.0/0	-
-	sgr-0ec19557ab93305...	IPv4	SSH	TCP	22	0.0.0.0/0	-
-	sgr-00a0e56d560959f45	IPv4	HTTP	TCP	80	0.0.0.0/0	-
-	sgr-064c062d69916fa84	IPv4	Custom TCP	TCP	5666	0.0.0.0/0	-
-	sgr-0613b7b6aa9d30def	IPv4	All traffic	All	All	0.0.0.0/0	-

You have to edit the inbound rules of the specified Security Group for this.

3. SSH into Your EC2 instance or simply use EC2 Instance Connect from the browser.



sudo yum update

```
[ec2-user@ip-172-31-91-91 ~]$
sudo yum update
Last metadata expiration check: 0:19:03 ago on Sun Sep 29 06:56:15 2024.
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-91-91 ~]$ |
```

sudo yum install httpd php

```
[ec2-user@ip-172-31-91-91 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:19:29 ago on Sun Sep 29 06:56:15 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
 nginx-filesystem               noarch             1:1.24.0-1.amzn2023.0.4                amazonlinux         9.8 k
 php8.3-cli                     x86_64             8.3.10-1.amzn2023.0.1                  amazonlinux         3.7 M
 php8.3-common                  x86_64             8.3.10-1.amzn2023.0.1                  amazonlinux         737 k
 php8.3-process                 x86_64             8.3.10-1.amzn2023.0.1                  amazonlinux          45 k
 php8.3-xml                     x86_64             8.3.10-1.amzn2023.0.1                  amazonlinux         154 k
Installing weak dependencies:
 apr-util-openssl               x86_64             1.6.3-1.amzn2023.0.1                   amazonlinux          17 k
 mod_http2                      x86_64             2.0.27-1.amzn2023.0.3                  amazonlinux         166 k
 mod_lua                        x86_64             2.4.62-1.amzn2023                      amazonlinux          61 k
 php8.3-fpm                     x86_64             8.3.10-1.amzn2023.0.1                  amazonlinux         1.9 M
 php8.3-mbstring                x86_64             8.3.10-1.amzn2023.0.1                  amazonlinux         528 k
 php8.3-opcache                 x86_64             8.3.10-1.amzn2023.0.1                  amazonlinux         379 k
 php8.3-pdo                     x86_64             8.3.10-1.amzn2023.0.1                  amazonlinux          89 k
 php8.3-sodium                  x86_64             8.3.10-1.amzn2023.0.1                  amazonlinux          41 k
```

```
Total                                                                                    22 MB/s |  10 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                          1/1
  Installing       : php8.3-common-8.3.10-1.amzn2023.0.1.x86_64                                               1/25
  Installing       : apr-1.7.2-2.amzn2023.0.2.x86_64                                                          2/25
  Installing       : apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64                                             3/25
  Installing       : apr-util-1.6.3-1.amzn2023.0.1.x86_64                                                     4/25
  Installing       : mailcap-2.1.49-3.amzn2023.0.3.noarch                                                     5/25
  Running scriptlet: httpd-filesystem-2.4.62-1.amzn2023.noarch                                                6/25
```

sudo yum install gcc glibc glibc-common

```
[ec2-user@ip-172-31-91-91 ~]$ sudo yum install gcc glibc glibc-common
Last metadata expiration check: 0:20:41 ago on Sun Sep 29 06:56:15 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
 make                           x86_64             1:4.3-5.amzn2023.0.2                   amazonlinux         534 k

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

Total download size: 52 M
```

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

Complete!
```

sudo yum install gd gd-devel

```
[ec2-user@ip-172-31-91-91 ~]$ sudo yum install gd gd-devel
Last metadata expiration check: 0:21:30 ago on Sun Sep 29 06:56:15 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
```

```
Installed:
  brotli-1.0.9-4.amzn2023.0.2.x86_64                    brotli-devel-1.0.9-4.amzn2023.0.2.x86_64                    bzip2-devel-1.0.8-6.amzn2023.0.2.x86_64
  cairo-1.17.6-2.amzn2023.0.1.x86_64                    cmake-filesystem-3.22.2-1.amzn2023.0.4.x86_64               fontconfig-2.13.94-2.amzn2023.0.2.x86_64
  fontconfig-devel-2.13.94-2.amzn2023.0.2.x86_64        fonts-filesystem-1:2.0.5-12.amzn2023.0.2.noarch             freetype-2.13.2-5.amzn2023.0.1.x86_64
  freetype-devel-2.13.2-5.amzn2023.0.1.x86_64           gd-2.3.3-5.amzn2023.0.3.x86_64                              gd-devel-2.3.3-5.amzn2023.0.3.x86_64
  glib2-devel-2.74.7-689.amzn2023.0.2.x86_64            google-noto-fonts-common-20201206-2.amzn2023.0.2.noarch     google-noto-sans-vf-fonts-20201206-2.amzn2023.0.2.noarch
  graphite2-1.3.14-7.amzn2023.0.2.x86_64                graphite2-devel-1.3.14-7.amzn2023.0.2.x86_64                harfbuzz-7.0.0-2.amzn2023.0.1.x86_64
  harfbuzz-devel-7.0.0-2.amzn2023.0.1.x86_64            harfbuzz-icu-7.0.0-2.amzn2023.0.1.x86_64                    jbigkit-libs-2.1-21.amzn2023.0.2.x86_64
  langpacks-core-font-en-3.0-21.amzn2023.0.4.noarch     libICE-1.0.10-6.amzn2023.0.2.x86_64                         libSM-1.2.3-8.amzn2023.0.2.x86_64
  libX11-1.7.2-3.amzn2023.0.4.x86_64                    libX11-common-1.7.2-3.amzn2023.0.4.noarch                   libX11-devel-1.7.2-3.amzn2023.0.4.x86_64
  libX11-xcb-1.7.2-3.amzn2023.0.4.x86_64                libXau-1.0.9-6.amzn2023.0.2.x86_64                          libXau-devel-1.0.9-6.amzn2023.0.2.x86_64
  libXext-1.3.4-6.amzn2023.0.2.x86_64                   libXpm-3.5.15-2.amzn2023.0.3.x86_64                         libXpm-devel-3.5.15-2.amzn2023.0.3.x86_64
  libXrender-0.9.10-14.amzn2023.0.2.x86_64              libXt-1.2.0-4.amzn2023.0.2.x86_64                           libblkid-devel-2.37.4-1.amzn2023.0.4.x86_64
  libffi-devel-3.4.4-1.amzn2023.0.1.x86_64              libicu-67.1-7.amzn2023.0.3.x86_64                           libicu-devel-67.1-7.amzn2023.0.3.x86_64
  libjpeg-turbo-2.1.4-2.amzn2023.0.5.x86_64             libjpeg-turbo-devel-2.1.4-2.amzn2023.0.5.x86_64             libmount-devel-2.37.4-1.amzn2023.0.4.x86_64
  libpng-2:1.6.37-10.amzn2023.0.6.x86_64                libpng-devel-2:1.6.37-10.amzn2023.0.6.x86_64                libselinux-devel-3.4-5.amzn2023.0.2.x86_64
  libsepol-devel-3.4-3.amzn2023.0.3.x86_64              libtiff-4.4.0-4.amzn2023.0.18.x86_64                        libtiff-devel-4.4.0-4.amzn2023.0.18.x86_64
  libwebp-1.2.4-1.amzn2023.0.6.x86_64                   libwebp-devel-1.2.4-1.amzn2023.0.6.x86_64                   libxcb-1.13.1-7.amzn2023.0.2.x86_64
  libxcb-devel-1.13.1-7.amzn2023.0.2.x86_64             libxml2-devel-2.10.4-1.amzn2023.0.6.x86_64                  pcre2-devel-10.40-1.amzn2023.0.3.x86_64
  pcre2-utf16-10.40-1.amzn2023.0.3.x86_64               pcre2-utf32-10.40-1.amzn2023.0.3.x86_64                     pixman-0.40.0-3.amzn2023.0.3.x86_64
  sysprof-capture-devel-3.40.1-2.amzn2023.0.2.x86_64    xml-common-0.6.3-56.amzn2023.0.2.noarch                     xorg-x11-proto-devel-2021.4-1.amzn2023.0.2.noarch
  xz-devel-5.2.5-9.amzn2023.0.2.x86_64                  zlib-devel-1.2.11-33.amzn2023.0.5.x86_64

Complete!
```

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

sudo adduser -m nagios

sudo passwd nagios

(password : *krushikesh)*

```
[ec2-user@ip-172-31-34-174 ~]$ sudo adduser -m nagios
sudo passwd nagios
Changing password for user nagios.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

6. Create a new user group

sudo groupadd nagcmd

```
[ec2-user@ip-172-31-91-91 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-91-91 ~]$
```

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

sudo usermod -a -G nagcmd nagios

sudo usermod -a -G nagcmd apache

```
[ec2-user@ip-172-31-91-91 ~]$ sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache
[ec2-user@ip-172-31-91-91 ~]$
```

8. Create a new directory for Nagios downloads

mkdir ~/downloads

cd ~/downloads

```
[ec2-user@ip-172-31-91-91 ~]$ mkdir ~/downloads
cd ~/downloads
```

9. Use wget to download the source zip files.

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

```
[ec2-user@ip-172-31-91-91 downloads]$ cd ..
[ec2-user@ip-172-31-91-91 ~]$ cd ~/downloads
[ec2-user@ip-172-31-91-91 downloads]$ wget https://assets.nagios.com/downloa
ds/nagioscore/releases/nagios-4.5.5.tar.gz
--2024-09-29 09:11:59--  https://assets.nagios.com/downloads/nagioscore/rele
ases/nagios-4.5.5.tar.gz
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2600:3c00::
f03c:92ff:fef7:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... con
nected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: 'nagios-4.5.5.tar.gz'

nagios-4.5.5.tar.g 100%[=================>]   1.97M  5.07MB/s    in 0.4s

2024-09-29 09:11:59 (5.07 MB/s) - 'nagios-4.5.5.tar.gz' saved [2065473/20654
73]

[ec2-user@ip-172-31-91-91 downloads]$ |
```

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

```
[ec2-user@ip-172-31-91-91 nagios-4.5.5]$ cd ..
[ec2-user@ip-172-31-91-91 downloads]$ wget https://nagios-plugins.org/downlo
ad/nagios-plugins-2.4.11.tar.gz
--2024-09-29 09:14:28--  https://nagios-plugins.org/download/nagios-plugins-
2.4.11.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 2753049 (2.6M) [application/x-gzip]
Saving to: 'nagios-plugins-2.4.11.tar.gz'

nagios-plugins-2.4 100%[=================>]   2.62M  6.92MB/s    in 0.4s
```

10. Use tar to unzip and change to that directory.
tar zxvf nagios-4.5.5.tar.gz

```
[ec2-user@ip-172-31-91-91 downloads]$ tar zxvf nagios-4.0.8.tar.gz
nagios-4.0.8/
nagios-4.0.8/.gitignore
nagios-4.0.8/Changelog
nagios-4.0.8/INSTALLING
nagios-4.0.8/LEGAL
nagios-4.0.8/LICENSE
nagios-4.0.8/Makefile.in
nagios-4.0.8/README
nagios-4.0.8/README.asciidoc
nagios-4.0.8/THANKS
nagios-4.0.8/UPGRADING
nagios-4.0.8/base/
nagios-4.0.8/base/.gitignore
```

11. Run the configuration script with the same group name you previously created.
./configure --with-command-group=nagcmd

Here we get an error

```
[ec2-user@ip-172-31-91-91 downloads]$ ./configure --with-command-group=nagcmd
-bash: ./configure: No such file or directory
[ec2-user@ip-172-31-91-91 downloads]$
```

Solution

Navigate to nagios folder in downloads

```
[ec2-user@ip-172-31-91-91 downloads]$ ls
nagios-4.0.8  nagios-4.0.8.tar.gz  nagios-plugins-2.0.3.tar.gz
[ec2-user@ip-172-31-91-91 downloads]$ cd nagios-4.0.8
[ec2-user@ip-172-31-91-91 nagios-4.0.8]$
```

Error 2: Cannot find SSL headers.
Solution: Install openssl dev library

Steps:
sudo yum install openssl-devel

```
[ec2-user@ip-172-31-91-91 nagios-4.5.5]$ sudo yum install openssl-devel
Last metadata expiration check: 2:24:05 ago on Sun Sep 29 06:56:15 2024.
Dependencies resolved.
================================================================================
 Package          Arch      Version                    Repository      Size
================================================================================
Installing:
 openssl-devel    x86_64    1:3.0.8-1.amzn2023.0.14    amazonlinux    3.0 M

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

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

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

```
             Event Broker:  yes
         Install ${prefix}:  /usr/local/nagios
     Install ${includedir}:  /usr/local/nagios/include/nagios
                 Lock file:  /run/nagios.lock
    Check result directory:  /usr/local/nagios/var/spool/checkresults
            Init directory:  /lib/systemd/system
   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-172-31-91-91 nagios-4.5.5]$
```

12. Compile the source code.

make all

```
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o nagios.o ./nagios.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o broker.o broker.c
```

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

sudo make install

sudo make install-init

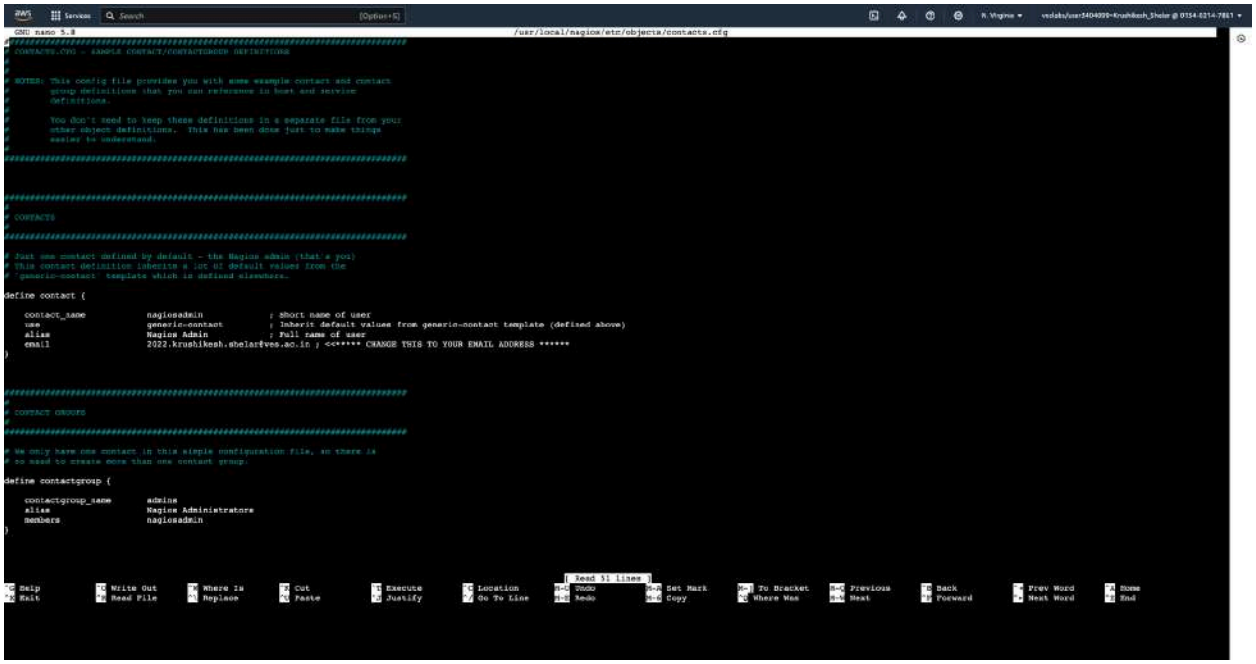
sudo make install-config

sudo make install-commandmode

```
[ec2-user@ip-172-31-91-91 nagios-4.5.5]$ make all

sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode
cd ./base && make
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o nagios.o ./nagios.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o broker.o broker.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o nebmods.o nebmods.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o ../common/shared.o ../common/shared.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o query-handler.o query-handler.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o workers.o workers.c
In function ‘get_wproc_list’,
    inlined from ‘get_worker’ at workers.c:277:12:
workers.c:253:17: warning: ‘%s’ directive argument is null [-Wformat-overflo
w=]
  253 |                 log_debug_info(DEBUGL_CHECKS, 1, "Found specialized
worker(s) for '%s'", (slash && *slash != '/') ? slash : cmd_name);
      |                 ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o checks.o checks.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o config.o config.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
```

14. Edit the config file and change the email address.

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



And change email with your email

15. Configure the web interface.

sudo make install-webconf

```
[ec2-user@ip-172-31-91-91 nagios-4.5.5]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios
.conf
if [ 0 -eq 1 ]; then \
        ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagio
s.conf; \
fi

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

[ec2-user@ip-172-31-91-91 nagios-4.5.5]$
```

16. Create a nagiosadmin account for nagios login along with password. You'll have to specify the password twice.

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

```
[ec2-user@ip-172-31-91-91 nagios-4.5.5]$ sudo htpasswd -c /usr/local/nagios/
etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
[ec2-user@ip-172-31-91-91 nagios-4.5.5]$
```

Password: krushikesh

17. Restart Apache

sudo service httpd restart

```
Adding password for user nagiosadmin
[ec2-user@ip-172-31-91-91 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-91-91 nagios-4.5.5]$
```

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

cd ~/downloads

tar zxvf nagios-plugins-2.4.11.tar.gz

```
[ec2-user@ip-172-31-91-91 downloads]$ cd ~/downloads
[ec2-user@ip-172-31-91-91 downloads]$ tar zxvf nagios-plugins-2.4.11.tar.gz
nagios-plugins-2.4.11/
nagios-plugins-2.4.11/build-aux/
nagios-plugins-2.4.11/build-aux/compile
nagios-plugins-2.4.11/build-aux/config.guess
nagios-plugins-2.4.11/build-aux/config.rpath
nagios-plugins-2.4.11/build-aux/config.sub
nagios-plugins-2.4.11/build-aux/install-sh
nagios-plugins-2.4.11/build-aux/ltmain.sh
nagios-plugins-2.4.11/build-aux/missing
nagios-plugins-2.4.11/build-aux/mkinstalldirs
nagios-plugins-2.4.11/build-aux/depcomp
nagios-plugins-2.4.11/build-aux/snippet/
nagios-plugins-2.4.11/build-aux/snippet/_Noreturn.h
nagios-plugins-2.4.11/build-aux/snippet/arg-nonnull.h
nagios-plugins-2.4.11/build-aux/snippet/c++defs.h
nagios-plugins-2.4.11/build-aux/snippet/warn-on-use.h
nagios-plugins-2.4.11/build-aux/test-driver
nagios-plugins-2.4.11/config_test/
```

19. Compile and install plugins

cd nagios-plugins-2.4.11

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

```
[ec2-user@ip-172-31-91-91 downloads]$ cd nagios-plugins-2.4.11
./configure --with-nagios-user=nagios --with-nagios-group=nagios
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether to enable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether gcc understands -c and -o together... yes
checking whether make supports the include directive... yes (GNU style)
checking dependency style of gcc... gcc3
checking how to run the C preprocessor... gcc -E
checking for grep that handles long lines and -e... /usr/bin/grep
checking for egrep... /usr/bin/grep -E
checking for Minix Amsterdam compiler... no
checking for ar... ar
checking for ranlib... ranlib
```

make
sudo make install

```
[ec2-user@ip-172-31-91-91 nagios-plugins-2.4.11]$ make
sudo make install
make  all-recursive
make[1]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
Making all in gl
make[2]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/
gl'
rm -f alloca.h-t alloca.h && \
{ echo '/* DO NOT EDIT! GENERATED AUTOMATICALLY! */'; \
  cat ./alloca.in.h; \
} > alloca.h-t && \
mv -f alloca.h-t alloca.h
rm -f c++defs.h-t c++defs.h && \
sed -n -e '/_GL_CXXDEFS/,$p' \
  < ../build-aux/snippet/c++defs.h \
  > c++defs.h-t && \
mv c++defs.h-t c++defs.h
rm -f warn-on-use.h-t warn-on-use.h && \
sed -n -e '/^.ifndef/,$p' \
  < ../build-aux/snippet/warn-on-use.h \
  > warn-on-use.h-t && \
mv warn-on-use.h-t warn-on-use.h
rm -f arg-nonnull.h-t arg-nonnull.h && \
sed -n -e '/GL_ARG_NONNULL/,$p' \
  < ../build-aux/snippet/arg-nonnull.h \
  > arg-nonnull.h-t && \
mv arg-nonnull.h-t arg-nonnull.h
/usr/bin/mkdir -p arpa
```

```
make[1]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
make[2]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
[ec2-user@ip-172-31-91-91 nagios-plugins-2.4.11]$
```

20. Start Nagios
Add Nagios to the list of system services
sudo chkconfig --add nagios
sudo chkconfig nagios on

```
[ec2-user@ip-172-31-91-91 nagios-plugins-2.4.11]$ sudo chkconfig --add nagio
s
sudo chkconfig nagios on
Note: Forwarding request to 'systemctl enable nagios.service'.
Synchronizing state of nagios.service with SysV service script with /usr/lib
/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable nagios
Created symlink /etc/systemd/system/multi-user.target.wants/nagios.service →
 /usr/lib/systemd/system/nagios.service.
[ec2-user@ip-172-31-91-91 nagios-plugins-2.4.11]$
```

Verify the sample configuration files
sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Error

```
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.0.8
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 08-12-2014
License: GPL

Website: http://www.nagios.org
Reading configuration data...
Error in configuration file '/usr/local/nagios/etc/nagios.cfg' - Line 452 (Check result path '/usr/local/nagios/var/spool/checkres
lts' is not a valid directory)
   Error processing main config file!
```

Solution:

 # <u>Create the missing directory</u>: <u>If the directory is missing, create it with the necessary permissions:</u>

sudo mkdir -p /usr/local/nagios/var/spool/checkresults

sudo chown nagios:nagios /usr/local/nagios/var/spool/checkresults

sudo chmod 775 /usr/local/nagios/var/spool/checkresults

```
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$ sudo mkdir -p /usr/local/nagios/var/spool/checkresults
sudo chown nagios:nagios /usr/local/nagios/var/spool/checkresults
sudo chmod 775 /usr/local/nagios/var/spool/checkresults
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$
```

Now run again

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

```
[ec2-user@ip-172-31-91-91 nagios-plugins-2.4.11]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

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

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

Running pre-flight check on configuration data...

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

Total Warnings: 0
Total Errors:   0
```

sudo service nagios start

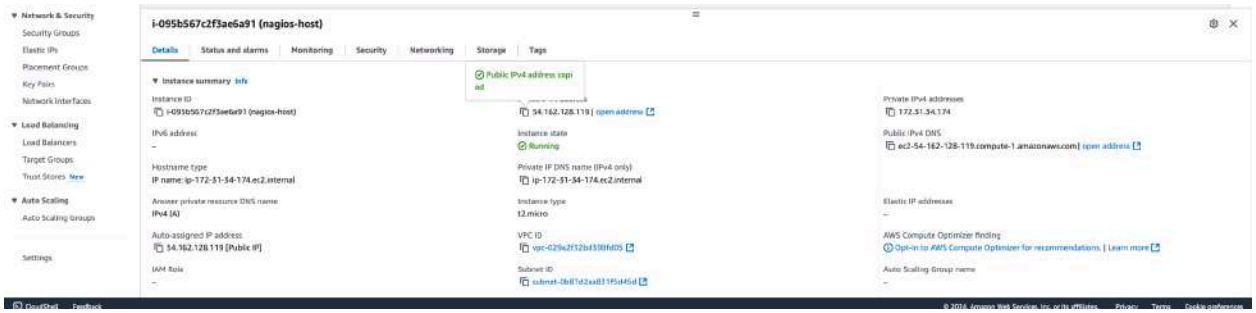
```
[ec2-user@ip-172-31-91-91 nagios-plugins-2.4.11]$ sudo service nagios start
Starting nagios (via systemctl):                           [  OK  ]
```

21. Check the status of Nagios

sudo systemctl status nagios

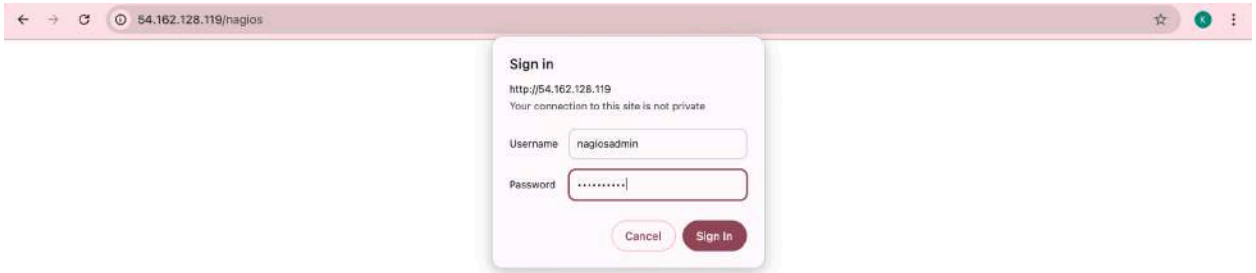
```
[ec2-user@ip-172-31-34-174 nagios-plugins-2.4.11]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-34-174 nagios-plugins-2.4.11]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Wed 2024-10-02 15:10:02 UTC; 12s ago
       Docs: https://www.nagios.org/documentation
    Process: 65649 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 65651 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 65652 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 5.6M
        CPU: 94ms
     CGroup: /system.slice/nagios.service
             ├─65652 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─65653 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─65654 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─65655 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─65656 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─65657 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 02 15:10:02 ip-172-31-34-174.ec2.internal nagios[65652]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully initialized
Oct 02 15:10:02 ip-172-31-34-174.ec2.internal nagios[65652]: qh: core query handler registered
Oct 02 15:10:02 ip-172-31-34-174.ec2.internal nagios[65652]: qh: echo service query handler registered
Oct 02 15:10:02 ip-172-31-34-174.ec2.internal nagios[65652]: qh: help for the query handler registered
Oct 02 15:10:02 ip-172-31-34-174.ec2.internal nagios[65652]: wproc: Successfully registered manager as @wproc with query handler
Oct 02 15:10:02 ip-172-31-34-174.ec2.internal nagios[65652]: wproc: Registry request: name=Core Worker 65656;pid=65656
Oct 02 15:10:02 ip-172-31-34-174.ec2.internal nagios[65652]: wproc: Registry request: name=Core Worker 65655;pid=65655
Oct 02 15:10:02 ip-172-31-34-174.ec2.internal nagios[65652]: wproc: Registry request: name=Core Worker 65654;pid=65654
Oct 02 15:10:02 ip-172-31-34-174.ec2.internal nagios[65652]: wproc: Registry request: name=Core Worker 65653;pid=65653
Oct 02 15:10:03 ip-172-31-34-174.ec2.internal nagios[65652]: Successfully launched command file worker with pid 65657
```

22. Go back to EC2 Console and copy the Public IP address of this instance

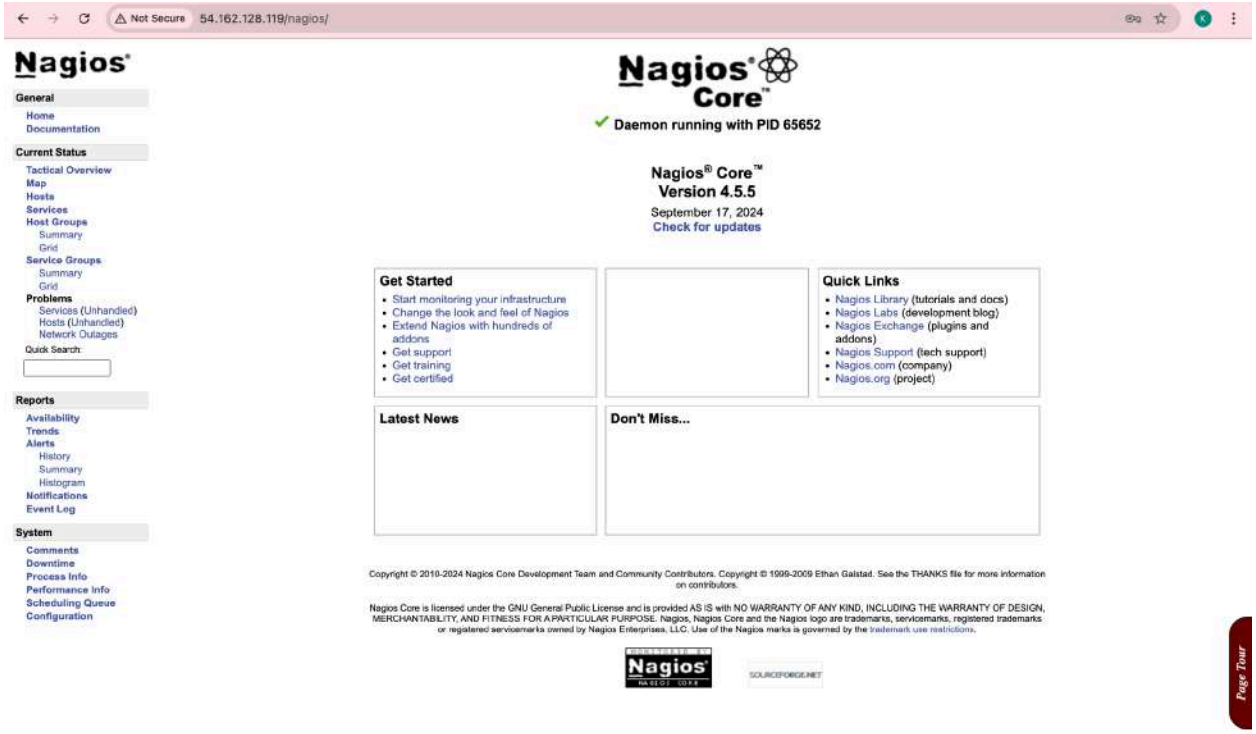


23. Open up your browser and look for http://<your_public_ip_address>/nagios

Enter username as nagiosadmin and password which you set in Step 16.



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



This means that Nagios was correctly installed and configured with its plugins so far.

Conclusion:

In this practical, we successfully installed and configured Nagios Core along with Nagios plugins and NRPE on an Amazon EC2 instance. We created a Nagios user, set up necessary permissions, and resolved common installation errors. Finally, we verified the setup by accessing the Nagios web interface, confirming that our monitoring system was fully operational.

Adv DevOps Practical 10

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

Theory:

Port and Service Monitoring

Port and service monitoring in Nagios involves checking the availability and responsiveness of network services running on specific ports. This ensures that critical services (like HTTP, FTP, or SSH) are operational. Nagios uses plugins to ping the ports and verify whether services are up and responding as expected, allowing administrators to be alerted in case of outages.

Windows/Linux Server Monitoring

Windows/Linux server monitoring with Nagios entails tracking the performance and health of servers running these operating systems. It includes monitoring metrics such as CPU usage, memory consumption, disk space, and system logs. Nagios employs various plugins to gather data, enabling administrators to ensure optimal performance, identify potential issues, and maintain uptime across their server infrastructure.

Prerequisites:

AWS Academy or Personal account.

Nagios Server running on Amazon Linux Machine. (Refer Experiment No 9)

Monitoring Using Nagios:

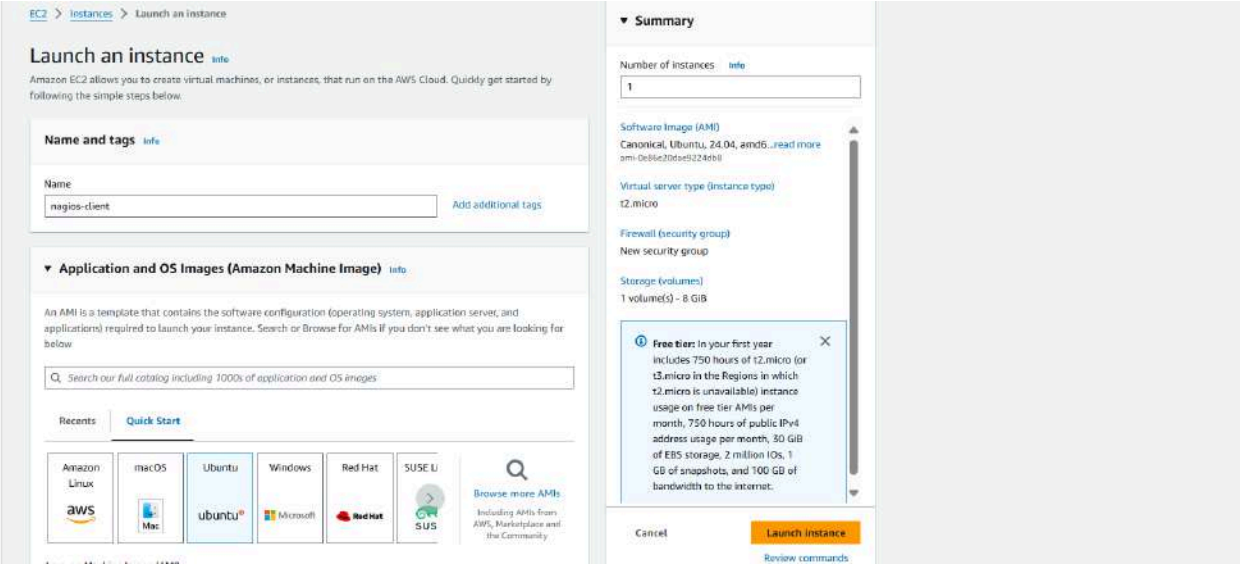
Step 1: To Confirm Nagios is running on the server side Perform the following command on your Amazon Linux Machine (Nagios-host).

sudo systemctl status nagios

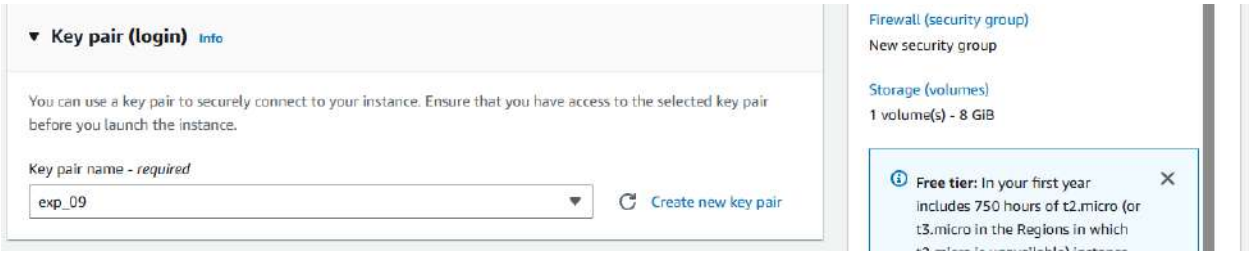
```
[ec2-user@ip-172-31-91-91 ~]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Sun 2024-09-29 16:18:08 UTC; 21min ago
       Docs: https://www.nagios.org/documentation
    Process: 1942 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 1944 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 1946 (nagios)
      Tasks: 8 (limit: 1112)
     Memory: 7.7M
        CPU: 387ms
     CGroup: /system.slice/nagios.service
             ├─1946 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─1947 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─1948 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─1949 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─1950 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─1956 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─3088 /usr/local/nagios/libexec/check_ping -H 127.0.0.1 -w 3000.0,80% -c 5000.0,100% -p 5
             └─3089 /usr/bin/ping -n -U -w 30 -c 5 127.0.0.1

Sep 29 16:18:08 ip-172-31-91-91.ec2.internal systemd[1]: Starting nagios.service - Nagios Core 4.5.5...
Sep 29 16:18:08 ip-172-31-91-91.ec2.internal systemd[1]: Started nagios.service - Nagios Core 4.5.5.
Sep 29 16:20:00 ip-172-31-91-91.ec2.internal nagios[1946]: SERVICE FLAPPING ALERT: localhost;HTTP;STARTED; Service appears to have started flapping (20.0% >
Sep 29 16:20:00 ip-172-31-91-91.ec2.internal nagios[1946]: SERVICE ALERT: localhost;HTTP;CRITICAL;HARD;4;connect to address 127.0.0.1 and port 80: Connecti>
Sep 29 16:22:30 ip-172-31-91-91.ec2.internal nagios[1946]: SERVICE NOTIFICATION: nagiosadmin;localhost;Swap Usage;CRITICAL;notify-service-by-email;SWAP CRI>
Sep 29 16:22:30 ip-172-31-91-91.ec2.internal nagios[1946]: wproc: NOTIFY job 2 from worker Core Worker 1948 is a non-check helper but exited with return co>
Sep 29 16:22:30 ip-172-31-91-91.ec2.internal nagios[1946]: wproc:   host=localhost; service=Swap Usage; contact=nagiosadmin
Sep 29 16:22:30 ip-172-31-91-91.ec2.internal nagios[1946]: wproc:   early_timeout=0; exited_ok=1; wait_status=32512; error_code=0;
Sep 29 16:22:30 ip-172-31-91-91.ec2.internal nagios[1946]: wproc:   stderr line 01: /bin/sh: line 1: /bin/mail: No such file or directory
Sep 29 16:22:30 ip-172-31-91-91.ec2.internal nagios[1946]: wproc:   stderr line 02: /usr/bin/printf: write error: Broken pipe
lines 1-30/30 (END)
```

You can now proceed if you get the above message/output.

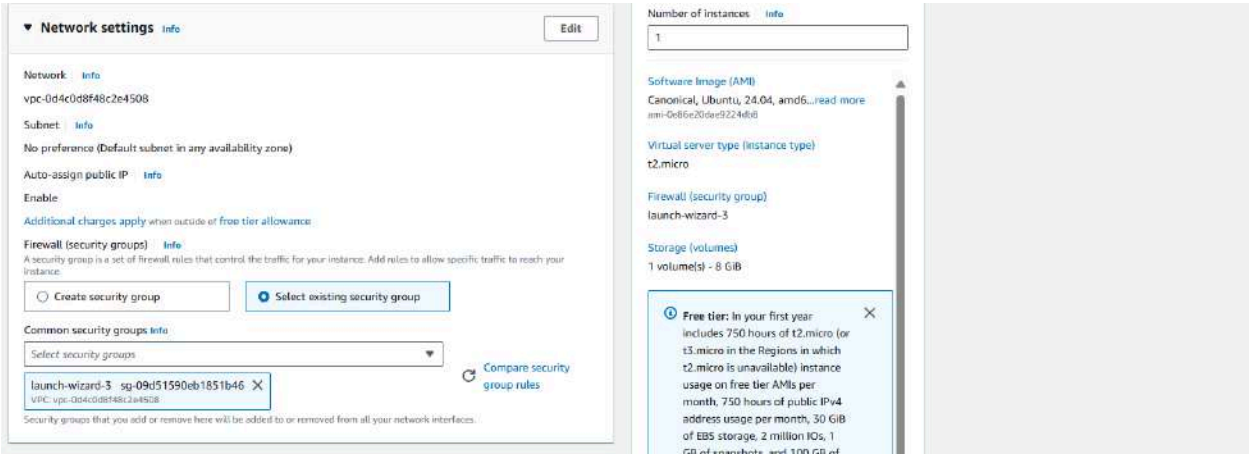
Step 2: Now Create a new EC2 instance. Name: Nagios-client, AMI: Ubuntu Instance Type: t2.micro.



For Key pair : Click on create key and make key of type RSA with extension .pem . Key will be downloaded to your local machine.
Now select that key in key pair if you already have key with type RSA and extension .pem no need to create new key but you must have that key downloaded.



Select the Existing Security Group and select the Security Group that we have created in Experiment no 9 or the same one you have used for the Nagios server (Nagios-host).



Now perform all the commands on the Nagios-host till step 10

Step 4: Now on the server Nagios-host run the following command.
ps -ef | grep nagios

```
[ec2-user@ip-172-31-91-91 ~]$ ps -ef | grep nagios
nagios      1946       1  0 16:18 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios      1947    1946  0 16:18 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      1948    1946  0 16:18 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      1949    1946  0 16:18 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      1950    1946  0 16:18 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      1956    1946  0 16:18 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
root        3090    3055  0 16:40 pts/0    00:00:00 sudo systemctl status nagios
root        3092    3090  0 16:40 pts/1    00:00:00 sudo systemctl status nagios
root        3093    3092  0 16:40 pts/1    00:00:00 systemctl status nagios
ec2-user    3914    3890  0 16:59 pts/2    00:00:00 grep --color=auto nagios
[ec2-user@ip-172-31-91-91 ~]$
```

Step 5: Now Become root user and create root directories.
sudo su
mkdir /usr/local/nagios/etc/objects/monitorhosts
mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts

```
[ec2-user@ip-172-31-91-91 ~]$ sudo su
[root@ip-172-31-91-91 ec2-user]# mkdir /usr/local/nagios/etc/objects/monitorhosts mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-91-91 ec2-user]#
```

Step 6: Copy the sample localhost.cfg to linuxhost.cfg by running the following command.(Below command should come in one line see screenshot below)
cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

```
[root@ip-172-31-91-91 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
[root@ip-172-31-91-91 ec2-user]#
```

Step 7: Open linuxserver.cfg using nano and make the following changes in all positions?everywhere in file.
> **nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg**

Change hostname to **linuxserver.**
Change address to the **public IP of your Linux client**.
Set hostgroup_name to **linux-servers1.**

```
#
# HOST DEFINITION
#

# Define a host for the local machine

define host {

    use                     linux-server            ; Name of host template to use
                                                    ; This host definition will inherit all variables that are defined
                                                    ; in (or inherited by) the linux-server host template definition.
    host_name               linuxserver
    alias                   localhost
    address                 172.31.92.146
}


#
# HOST GROUP DEFINITION
#

# 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 hosts that belong to this group
}
```

Step 8: Now update the Nagios config file .Add the following line in the file. Line to add :
> nano /usr/local/nagios/etc/nagios.cfg

cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/

```
#
# NAGIOS.CFG - Sample Main Config File for Nagios 4.5.5
#
# Read the documentation for more information on this configuration
# file.  I've provided some comments here, but things may not be so
# clear without further explanation.
#
#


# LOG FILE
# This is the main log file where service and host events are logged
# for historical purposes.  This should be the first option specified
# in the config file!!!

log_file=/usr/local/nagios/var/nagios.log


# OBJECT CONFIGURATION FILE(S)
# 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
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/
# Definitions for monitoring the local (Linux) host
cfg_file=/usr/local/nagios/etc/objects/localhost.cfg
```

Step 9: Now Verify the configuration files by running the following commands.
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

```
Things look okay - No serious problems were detected during the pre-flight check
[root@ip-172-31-91-91 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
```

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

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

Total Warnings: 0
Total Errors:   0

Things look okay - No serious problems were detected during the pre-flight check
```

Step 10: Now restart the services of nagios by running the following command.
service nagios restart

```
Things look okay - No serious problems were detected during the pre-flight check
[root@ip-172-31-91-91 ec2-user]# service nagios restart
Restarting nagios (via systemctl):                         [  OK  ]
[root@ip-172-31-91-91 ec2-user]#
```

Step 11: Now Go to the Nagios-client ssh terminal and update and install the packages by running the following command.
sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins

```
ubuntu@ip-172-31-92-146:~$ sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins

Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [380 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [82.9 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4560 B]
Get:11 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [272 kB]
Get:12 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [115 kB]
Get:13 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:14 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.3 kB]
Get:15 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [353 kB]
Get:16 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [68.1 kB]
Get:17 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [428 B]
Get:18 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [10.9 kB]
Get:19 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [2808 B]
Get:20 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:21 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [344 B]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:24 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
```

```
Service restarts being deferred:
 /etc/needrestart/restart.d/dbus.service
 systemctl restart getty@tty1.service
 systemctl restart networkd-dispatcher.service
 systemctl restart serial-getty@ttyS0.service
 systemctl restart systemd-logind.service
 systemctl restart unattended-upgrades.service

No containers need to be restarted.

User sessions running outdated binaries:
 ubuntu @ session #2: sshd[992,1102]
 ubuntu @ session #7: sshd[1190,1248]
ubuntu@ip-172-31-92-146:~$
```

Step 12: Open nrpe.cfg file to make changes.Under allowed_hosts, add your nagios host IP address.

sudo nano /etc/nagios/nrpe.cfg

```
# NRPE GROUP
# This determines the effective group that the NRPE daemon should run as.
# You can either supply a group name or a GID.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd

nrpe_group=nagios



# ALLOWED HOST ADDRESSES
# This is an optional comma-delimited list of IP address or hostnames
# that are allowed to talk to the NRPE daemon. Network addresses with a bit mask
# (i.e. 192.168.1.0/24) are also supported. Hostname wildcards are not currently
# supported.
#
# Note: The daemon only does rudimentary checking of the client's IP
# address.  I would highly recommend adding entries in your /etc/hosts.allow
# file to allow only the specified host to connect to the port
# you are running this daemon on.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd

allowed_hosts=127.0.0.1,::1,34.207.68.187



# COMMAND ARGUMENT PROCESSING
# This option determines whether or not the NRPE daemon will allow clients
# to specify arguments to commands that are executed.  This option only works
# if the daemon was configured with the --enable-command-args configure script
# option.
#
# *** ENABLING THIS OPTION IS A SECURITY RISK! ***
```

Step 13: Now restart the NRPE server by this command.

sudo systemctl restart nagios-nrpe-server

```
0 upgraded, 0 newly installed, 0 to remove and 139 not upgraded.
ubuntu@ip-172-31-92-146:~$ sudo nano /etc/nagios/nrpe.cfg
ubuntu@ip-172-31-92-146:~$ sudo systemctl restart nagios-nrpe-server
ubuntu@ip-172-31-92-146:~$
```

Step 14: Now again check the status of Nagios by running this command on Nagios-host and also check httpd is active and run the command to active it.

sudo systemctl status nagios

```
[ec2-user@ip-172-31-91-91 ~]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Sun 2024-09-29 17:20:07 UTC; 12min ago
       Docs: https://www.nagios.org/documentation
    Process: 4761 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 4762 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 4763 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 4.1M
        CPU: 234ms
     CGroup: /system.slice/nagios.service
             ├─4763 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─4764 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─4765 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─4766 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─4767 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─4768 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 29 17:20:07 ip-172-31-91-91.ec2.internal nagios[4763]: Warning: Duplicate definition found for service 'Current Users' on host 'localhost' (config file
Sep 29 17:20:07 ip-172-31-91-91.ec2.internal nagios[4763]: Warning: Duplicate definition found for service 'Root Partition' on host 'localhost' (config fil
Sep 29 17:20:07 ip-172-31-91-91.ec2.internal nagios[4763]: Warning: Duplicate definition found for service 'PING' on host 'localhost' (config file '/usr/lo
Sep 29 17:20:07 ip-172-31-91-91.ec2.internal nagios[4763]: Successfully launched command file worker with pid 4768
Sep 29 17:22:30 ip-172-31-91-91.ec2.internal nagios[4763]: SERVICE NOTIFICATION: nagiosadmin;localhost;Swap Usage;CRITICAL;notify-service-by-email;SWAP CRI
Sep 29 17:22:30 ip-172-31-91-91.ec2.internal nagios[4763]: wproc: NOTIFY job 1 from worker Core Worker 4766 is a non-check helper but exited with return co
Sep 29 17:22:30 ip-172-31-91-91.ec2.internal nagios[4763]: wproc:   host=localhost; service=Swap Usage; contact=nagiosadmin
Sep 29 17:22:30 ip-172-31-91-91.ec2.internal nagios[4763]: wproc:   early_timeout=0; exited_ok=1; wait_status=32512; error_code=0;
Sep 29 17:22:30 ip-172-31-91-91.ec2.internal nagios[4763]: wproc:   stderr line 01: /bin/sh: line 1: /bin/mail: No such file or directory
Sep 29 17:22:30 ip-172-31-91-91.ec2.internal nagios[4763]: wproc:   stderr line 02: /usr/bin/printf: write error: Broken pipe
lines 1-28/28 (END)
```

sudo systemctl status httpd

```
[ec2-user@ip-172-31-91-91 ~]$ sudo systemctl status httpd
○ httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
    Drop-In: /usr/lib/systemd/system/httpd.service.d
             └─php-fpm.conf
     Active: inactive (dead)
       Docs: man:httpd.service(8)
[ec2-user@ip-172-31-91-91 ~]$
```

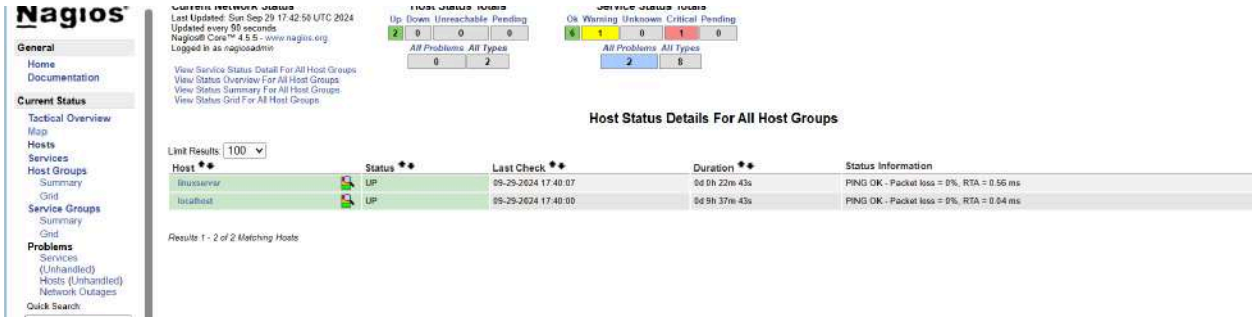
sudo systemctl start httpd
sudo systemctl enable httpd

```
[ec2-user@ip-172-31-91-91 ~]$ sudo systemctl start httpd
[ec2-user@ip-172-31-91-91 ~]$ sudo systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-172-31-91-91 ~]$
```

Step 15: Now to check Nagios dashboard go to http://<nagios host ip>/nagios
Eg. http://34.207.68.187/nagios

Enter username as nagiosadmin and password which you set in Exp 9.

Now Click on Hosts from left side panel



Conclusion:

In this exercise, we established a Nagios host and client to oversee service and server performance across both Linux and Windows environments. We configured Nagios on an Amazon Linux instance to track essential services such as HTTP, SSH, and system resources, guaranteeing their availability and health. By provisioning a new EC2 instance as the Nagios client, we facilitated effective communication between the client and host, enabling streamlined service monitoring. This configuration enhances uptime and allows for swift identification of any issues within the infrastructure.

Name: Krushikesh Shelar **2024-2025** **Roll No:51** **Div:D15-C**

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

Theory:

Overview of AWS Lambda

AWS Lambda is a serverless computing service that allows you to run code without managing servers. It automatically scales in response to incoming requests or events, ensuring efficient resource use. You are billed only for the time your code is executed, making it a budget-friendly option for on-demand applications.

How AWS Lambda Works

1. **Function Development**: Begin by writing your code and designating a handler (the entry point). You can use the AWS Management Console, CLI, or upload a package.
2. **Trigger Setup**: Determine the events that will trigger your function, such as uploads to S3 or changes in a DynamoDB table.
3. **Execution**: Upon triggering, Lambda runs your function and handles the necessary logic while automatically scaling to accommodate the volume of events.
4. **Scaling and Concurrency**: Lambda dynamically scales by creating additional instances of your function for simultaneous requests. You can also configure reserved concurrency to manage peak traffic.
5. **Monitoring and Logging**: Integrated with Amazon CloudWatch, Lambda provides logging and monitoring capabilities. You can view logs for each execution, helping you track performance and troubleshoot issues.

Programming Languages Supported

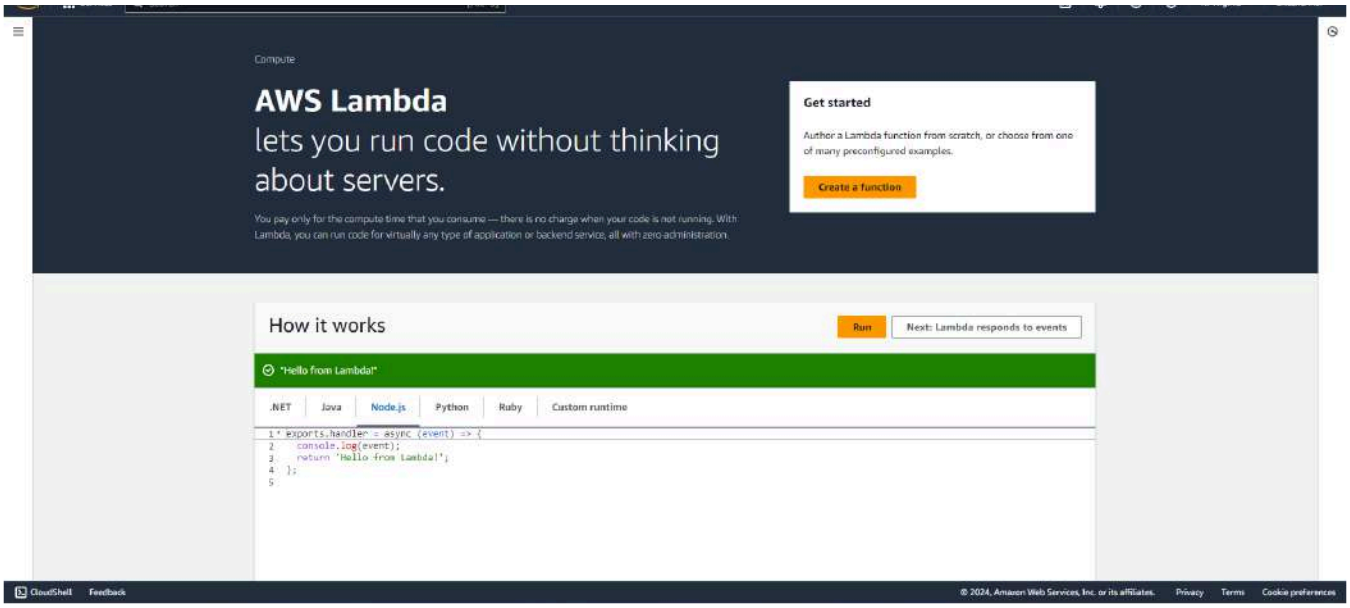
- **Python**: Excellent for quick development and lightweight tasks due to its extensive standard library.
- **Java**: Suitable for more demanding, compute-intensive applications, though it may have longer cold start times.
- **Node.js**: Well-suited for I/O-heavy applications like APIs, offering fast startup times and efficient resource use.

Prerequisites: AWS Personal/Academy Account

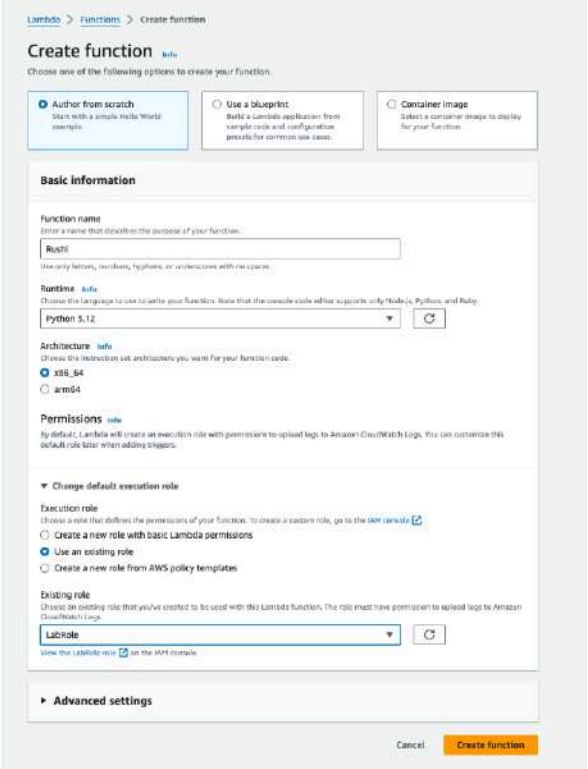
Name: Krushikesh Shelar **2024-2025** **Roll No:51** **Div:D15-C**

Steps To create the lambda function:

Step 1: Login to your AWS Personal/Academy Accout.Open lambda and click on create function button.

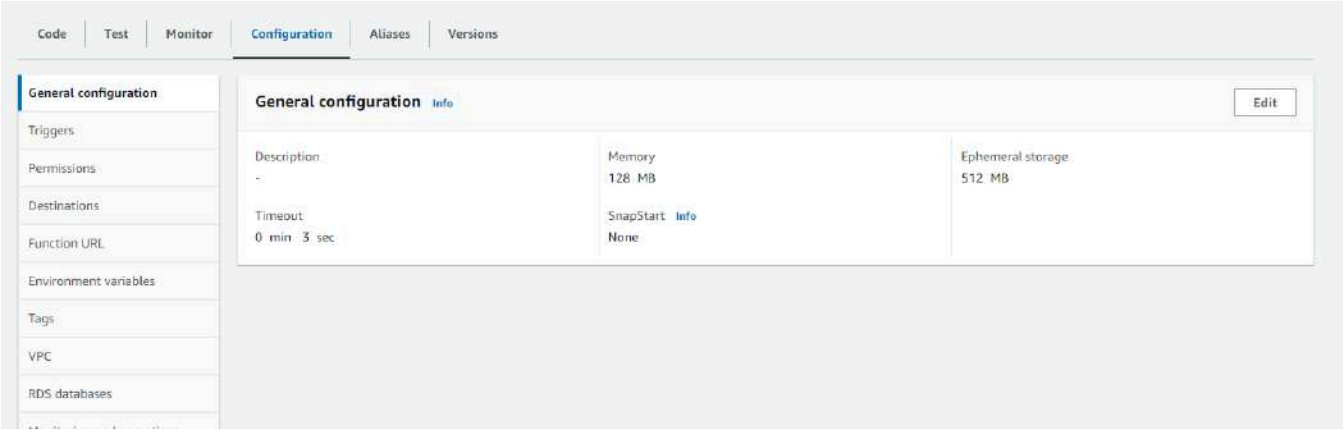


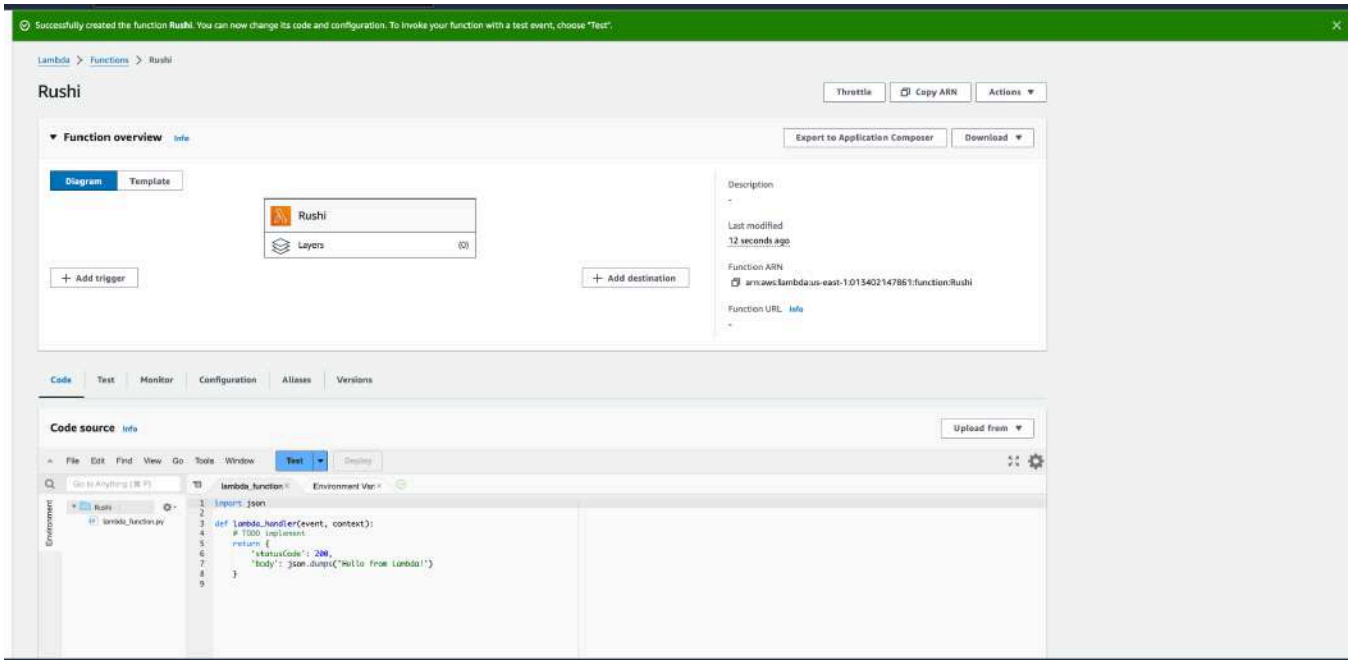
Step 2: Now Give a name to your Lambda function, Select the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby. So will select Python 3.12 , Architecture as x86, and Existing(Lab Role) role to Create a new role with basic Lambda permissions.



So See or Edit the basic settings go to configuration then click on edit general setting.

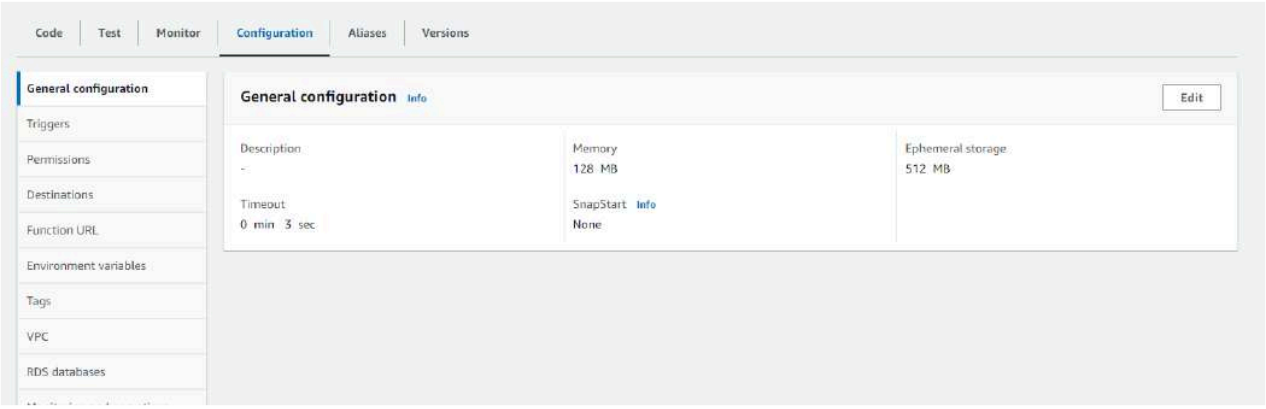
Name: Krushikesh Shelar **2024-2025** **Roll No:51** **Div:D15-C**



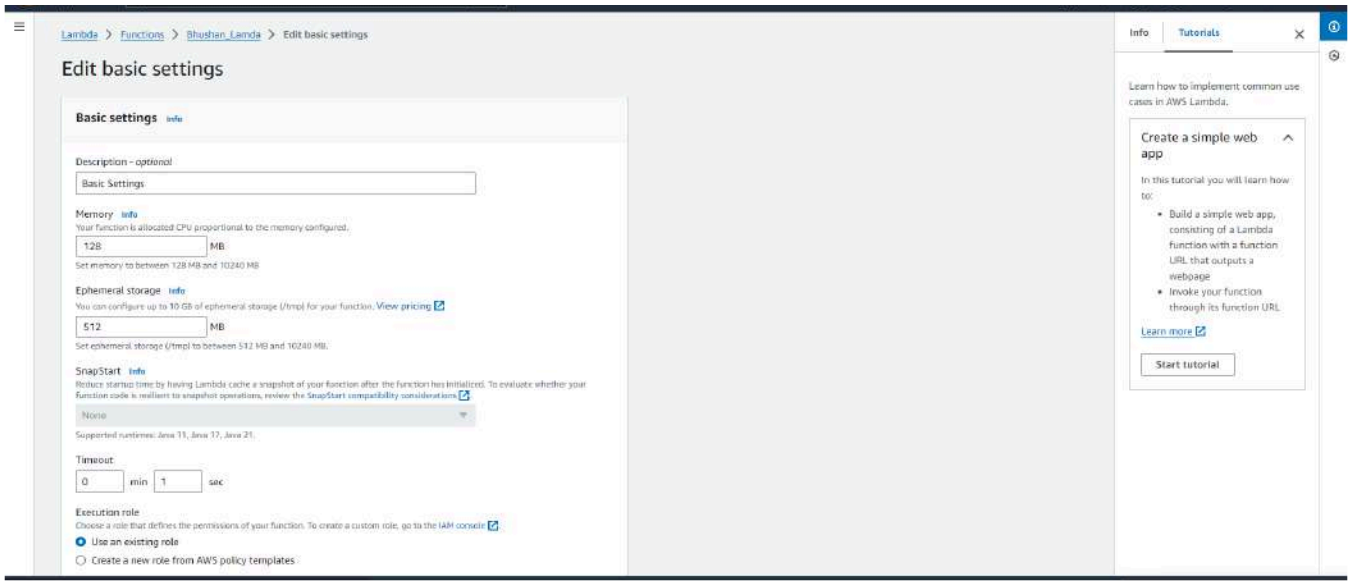


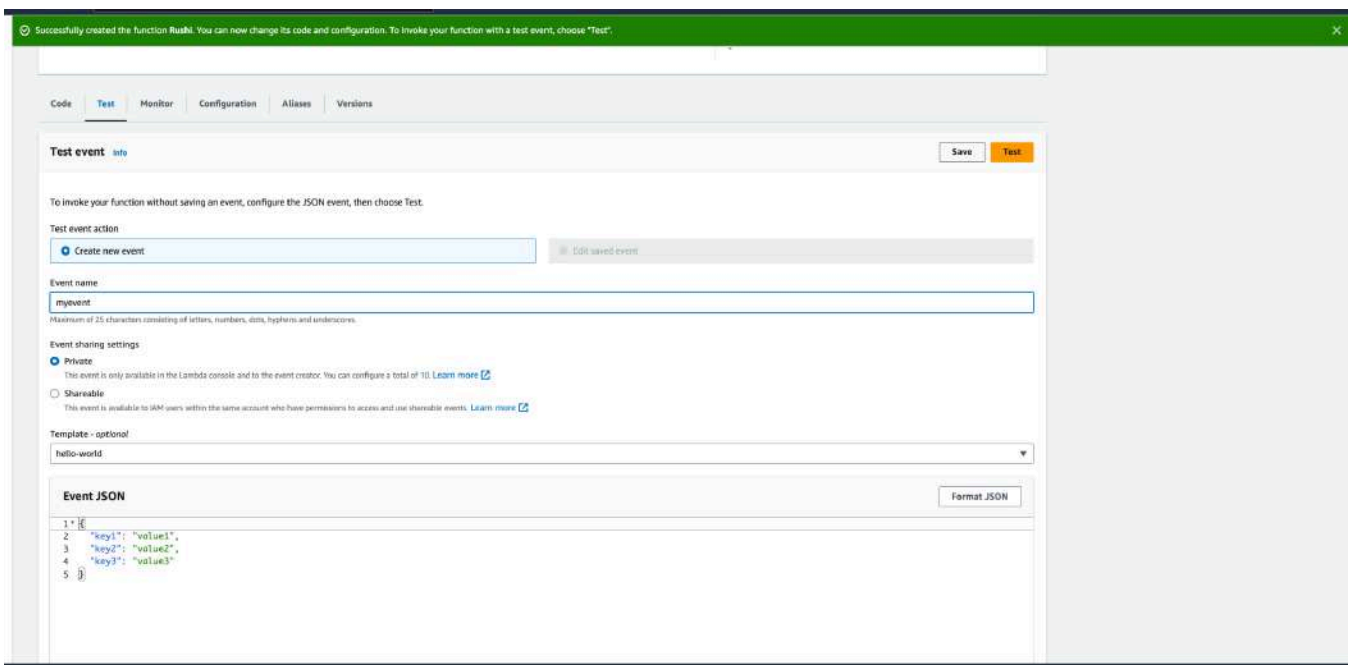
Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.

Step 3: Now Click on the Test tab then select Create a new event, give a name to the event and select Event Sharing to private, and select hello-world template.

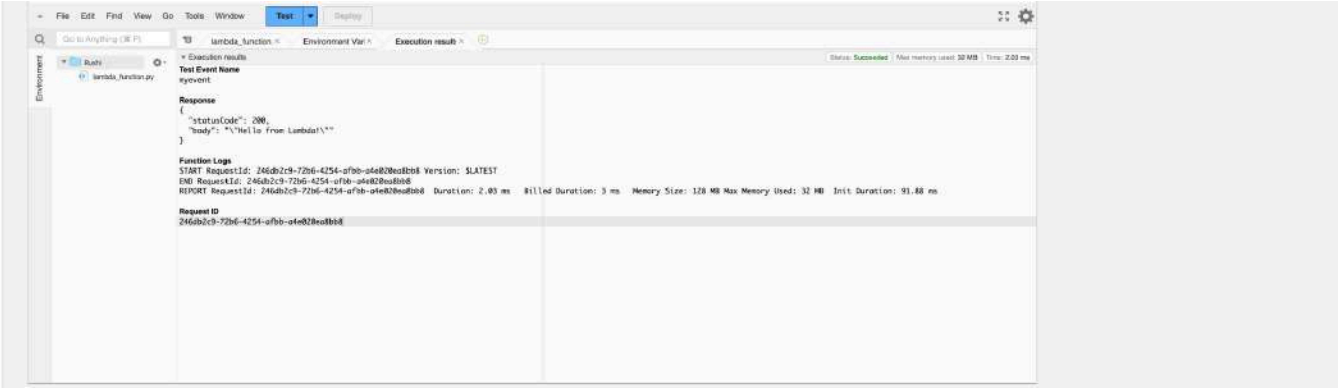


Name: Krushikesh Shelar **2024-2025** **Roll No:51** **Div:D15-C**



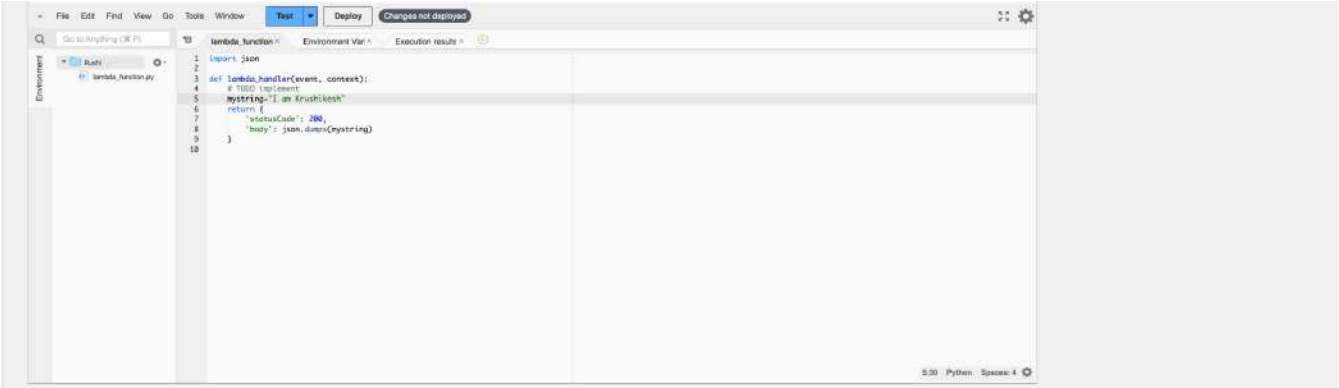


Step 4: Now In Code section select the created event from the dropdown of test then click on test . You will see the below output.

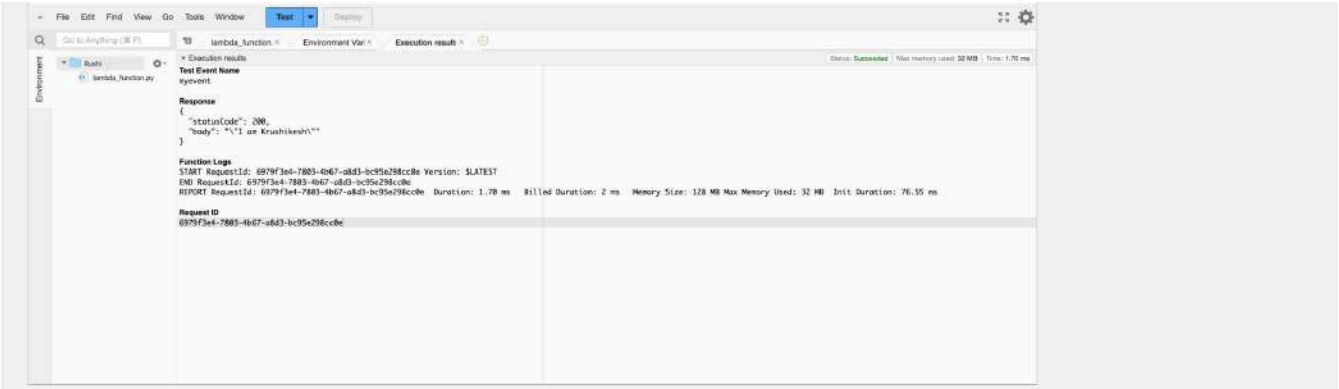


Name: Krushikesh Shelar **2024-2025** **Roll No:51** **Div:D15-C**

Step 5: You can edit your lambda function code. I have changed the code to display the new String. Now ctrl+s to save and click on deploy to deploy the changes.



Step 6: Now click on the test and observe the output. We can see the status code 200 and your string output and function logs. On successful deployment.



Conclusion:

We successfully built an AWS Lambda function and explored the core steps involved. After developing the function in Python, we adjusted the key settings, including a timeout of 1 second. We then generated a test event, deployed the function, and checked the output. Furthermore, we updated the function's code and redeployed it to observe the immediate impact of those changes.

This hands-on session showcased the user-friendly nature and adaptability of AWS Lambda for building serverless applications, allowing us to focus on writing code while AWS efficiently handles infrastructure and scaling concerns.

Name:Krushikesh Shelar 2024-2025 D15C Roll No:51

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

Theory:

Overview of AWS Lambda

AWS Lambda is a serverless computing service that allows you to run code without managing servers. It automatically scales in response to incoming requests or events, ensuring efficient resource use. You are billed only for the time your code is executed, making it a budget-friendly option for on-demand applications.

How AWS Lambda Works

1. **Function Development**: Begin by writing your code and designating a handler (the entry point). You can use the AWS Management Console, CLI, or upload a package.
2. **Trigger Setup**: Determine the events that will trigger your function, such as uploads to S3 or changes in a DynamoDB table.
3. **Execution**: Upon triggering, Lambda runs your function and handles the necessary logic while automatically scaling to accommodate the volume of events.
4. **Scaling and Concurrency**: Lambda dynamically scales by creating additional instances of your function for simultaneous requests. You can also configure reserved concurrency to manage peak traffic.
5. **Monitoring and Logging**: Integrated with Amazon CloudWatch, Lambda provides logging and monitoring capabilities. You can view logs for each execution, helping you track performance and troubleshoot issues.

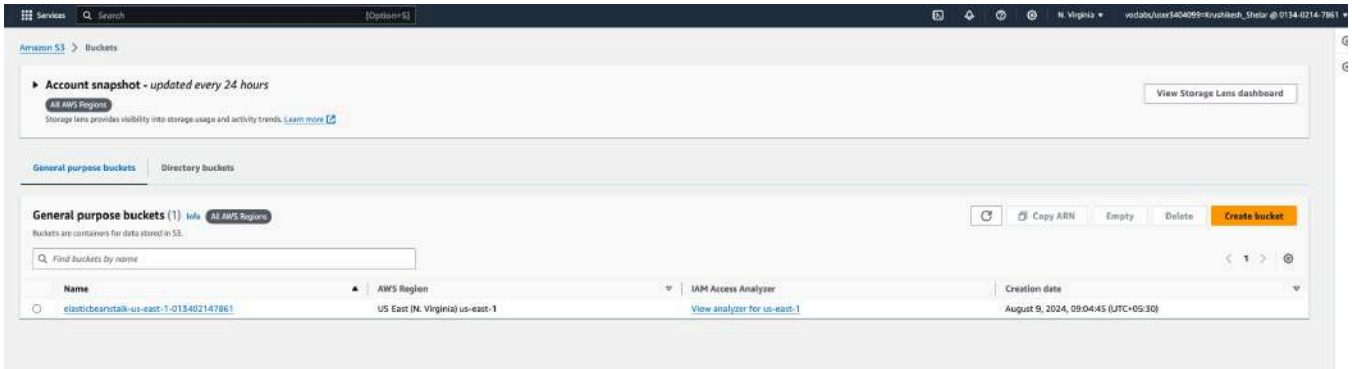
Programming Languages Supported

- **Python**: Excellent for quick development and lightweight tasks due to its extensive standard library.
- **Java**: Suitable for more demanding, compute-intensive applications, though it may have longer cold start times.
- **Node.js**: Well-suited for I/O-heavy applications like APIs, offering fast startup times and efficient resource use.

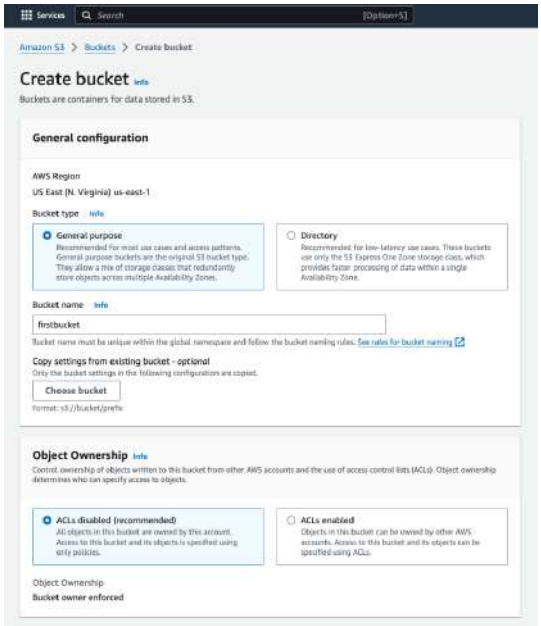
Steps To create the lambda function:

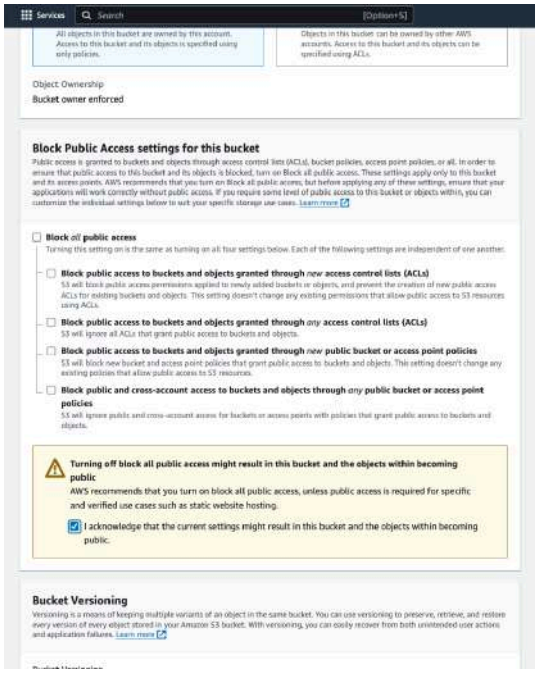
Name:Krushikesh Shelar 2024-2025 D15C Roll No:51

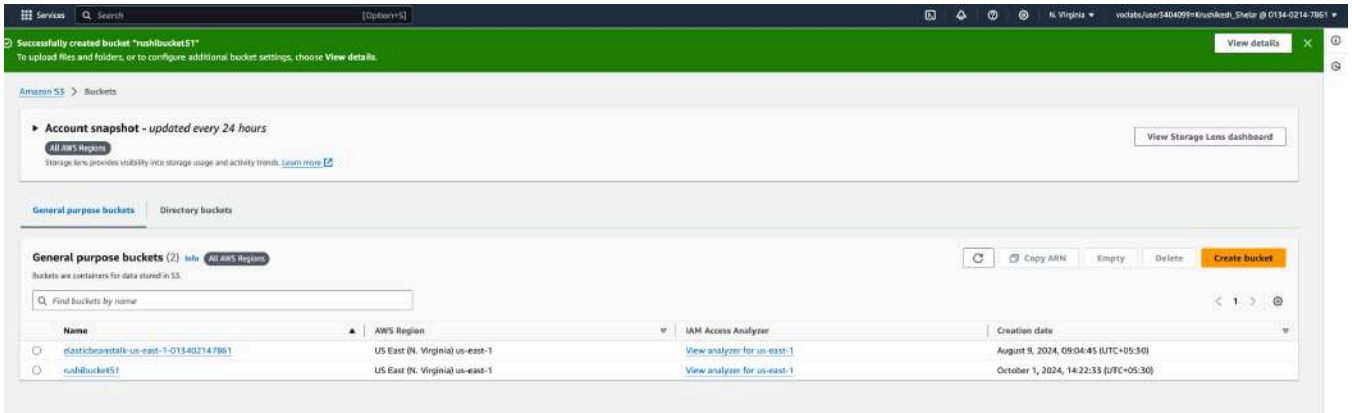
Step 1: Login to your AWS Academy account. Now open S3 from services and click on create S3 bucket.



Step 2: Now Give a name to the Bucket, select general purpose project and deselect the Block public access and keep other this to default.

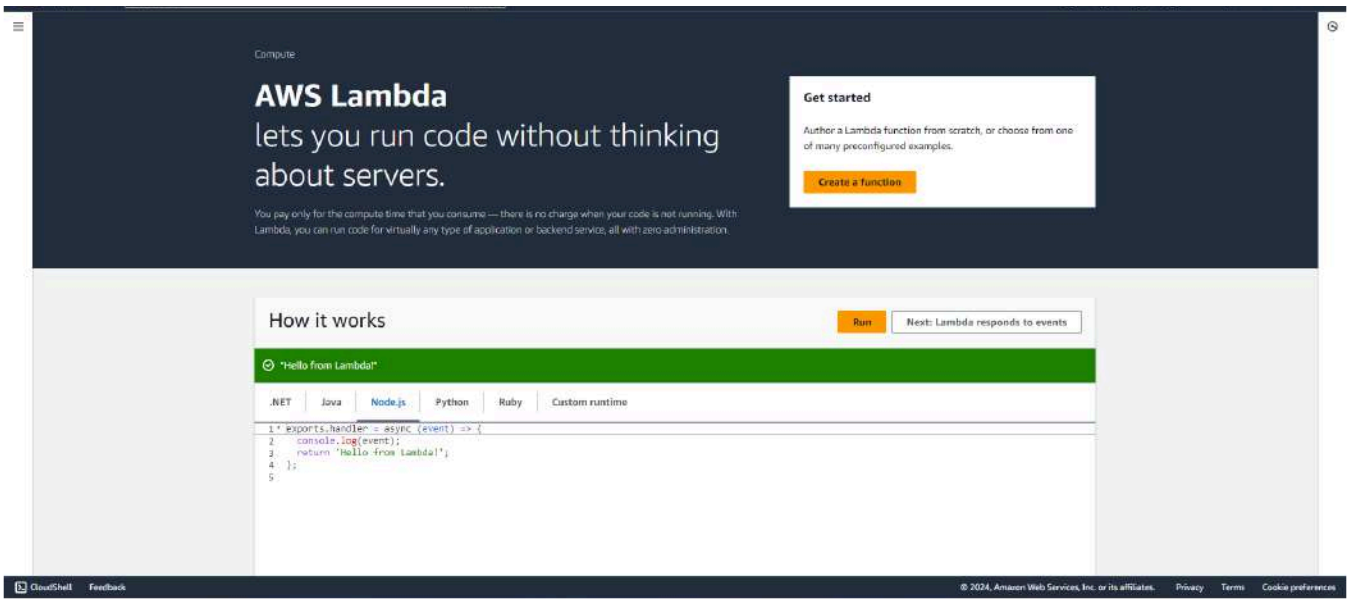




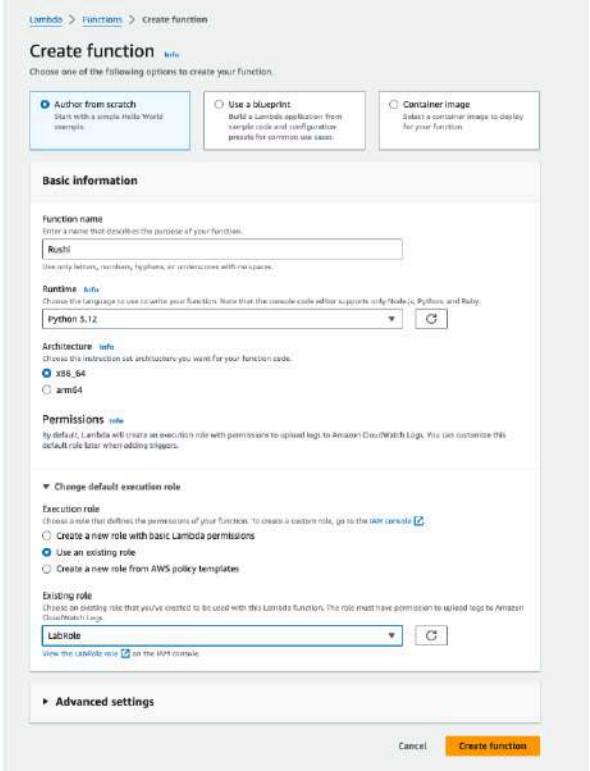


Name:Krushikesh Shelar 2024-2025 D15C Roll No:51

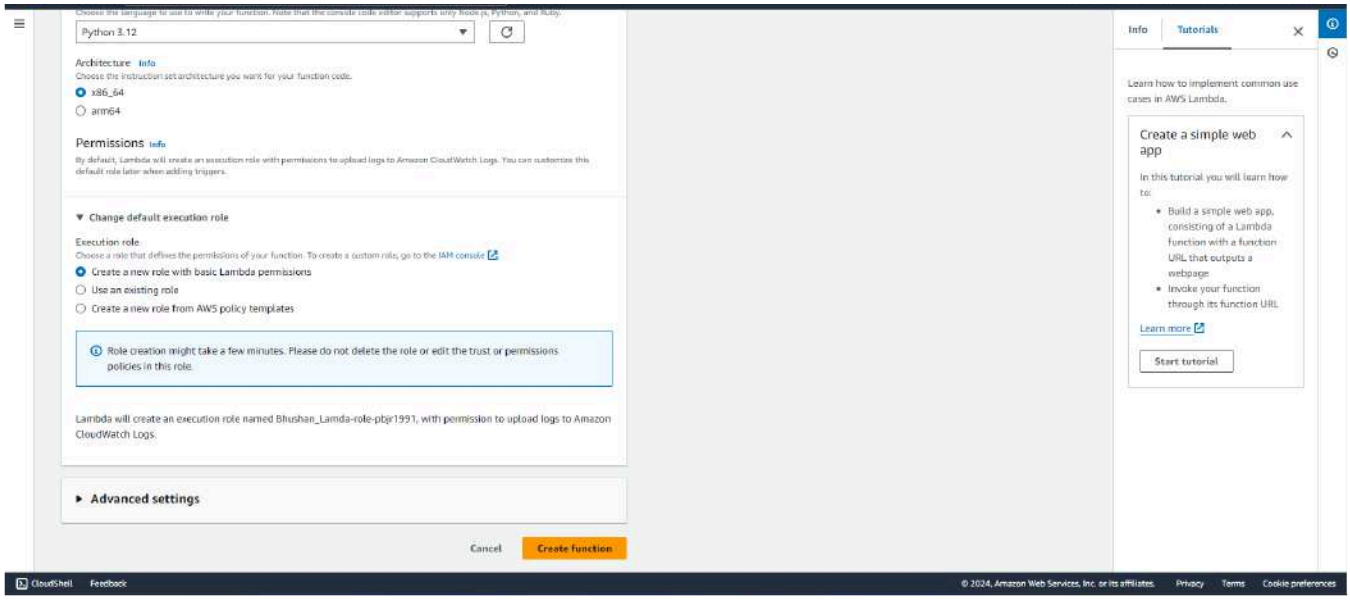
Step 3: Open lambda console and click on create function button.



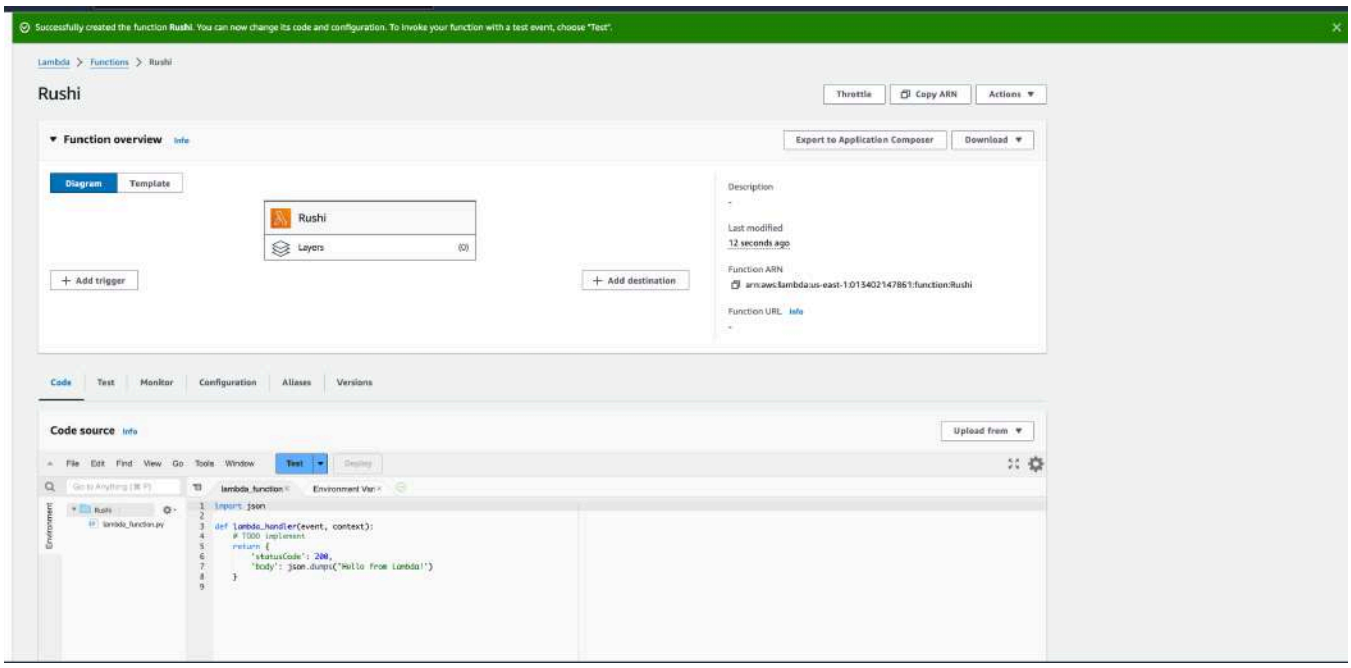
Step 4: Now Give a name to your Lambda function, Select the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby. So will select Python 3.12 , Architecture as x86, and Exceution role to Create a new role with basic Lambda permissions.



Name:Krushikesh Shelar 2024-2025 D15C Roll No:51



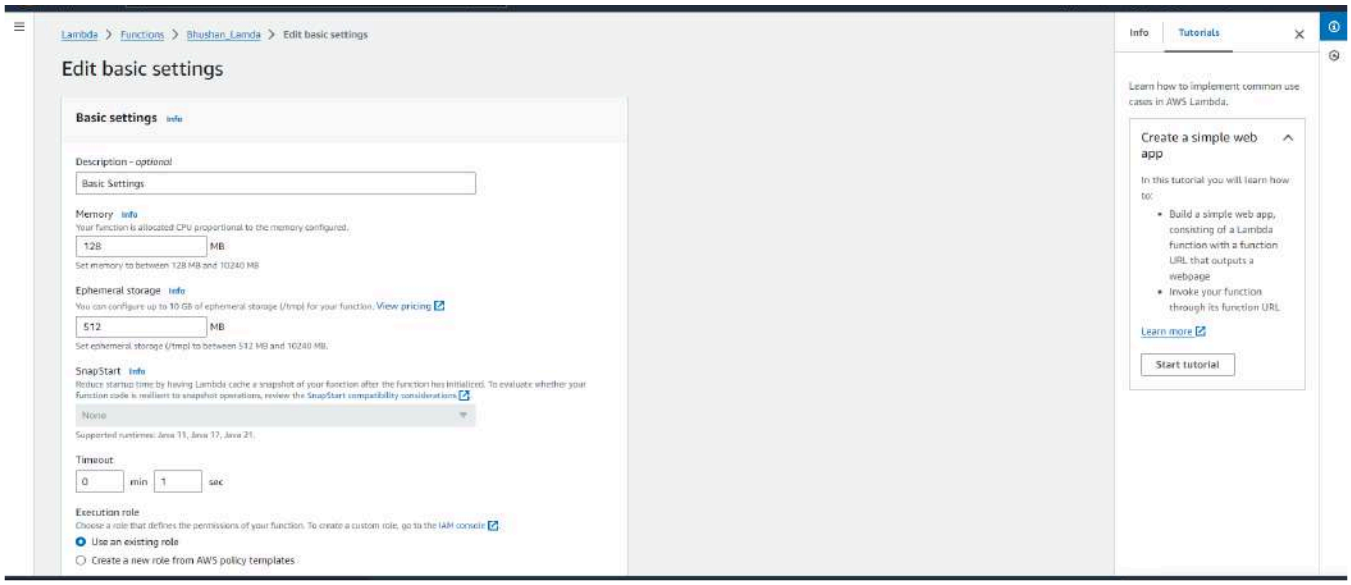
Lambda function code:

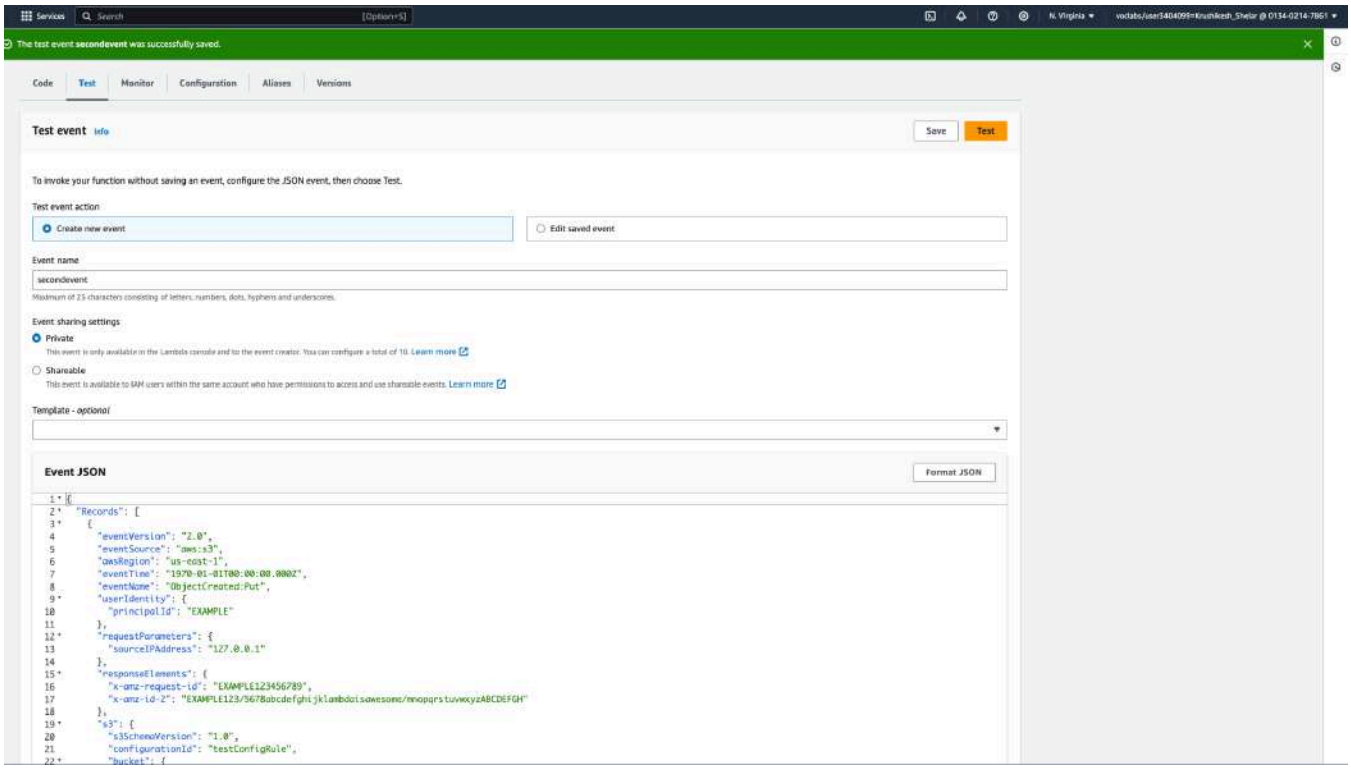


So See or Edit the basic settings go to configuration then click on edit general setting. Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.

Step 5: Now Click on the Test tab then select Create a new event, give a name to the event and select Event Sharing to private, and select s3 put template.

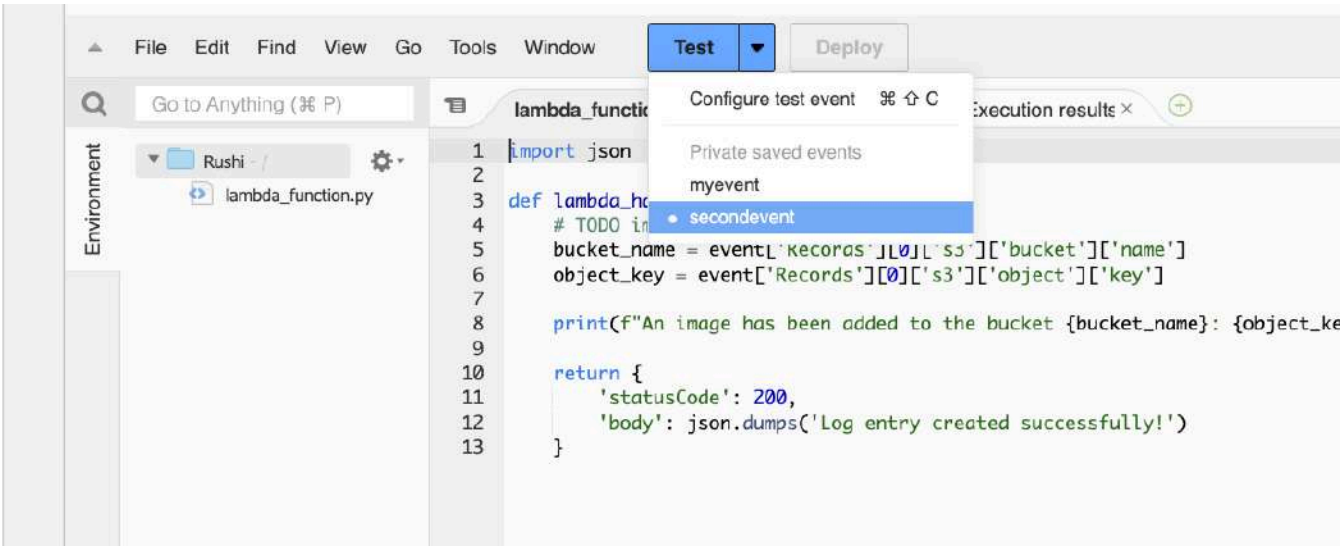
Name:Krushikesh Shelar 2024-2025 D15C Roll No:51



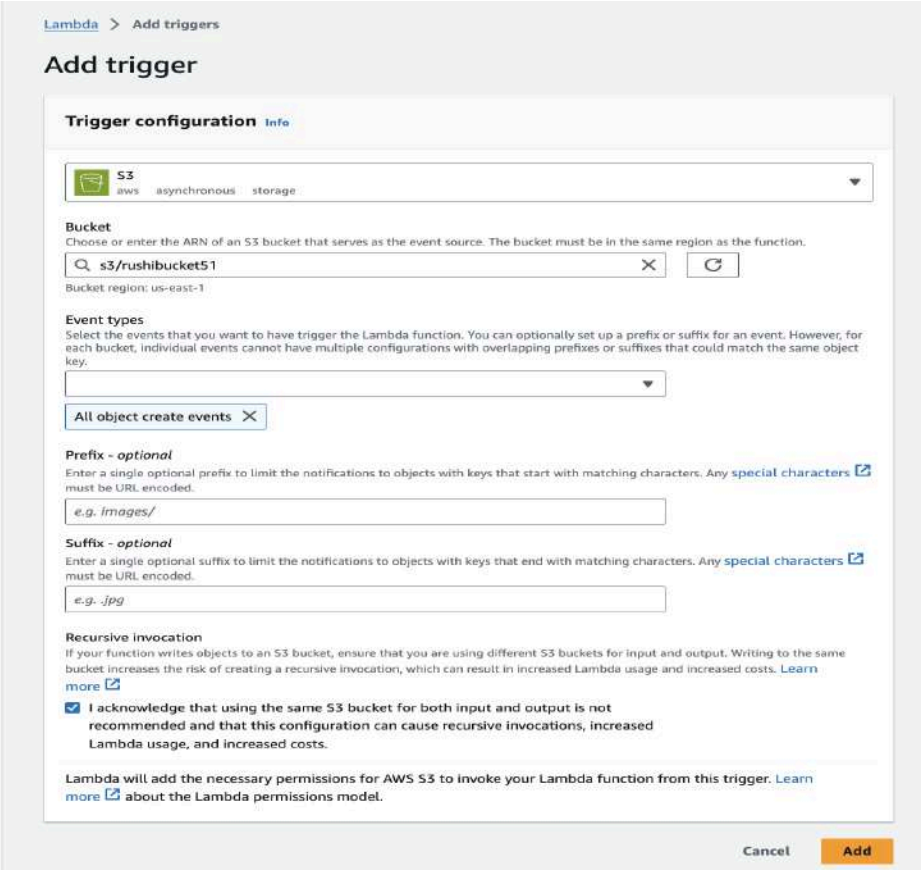


Step 6: Now In Code section select the created event from the dropdown .

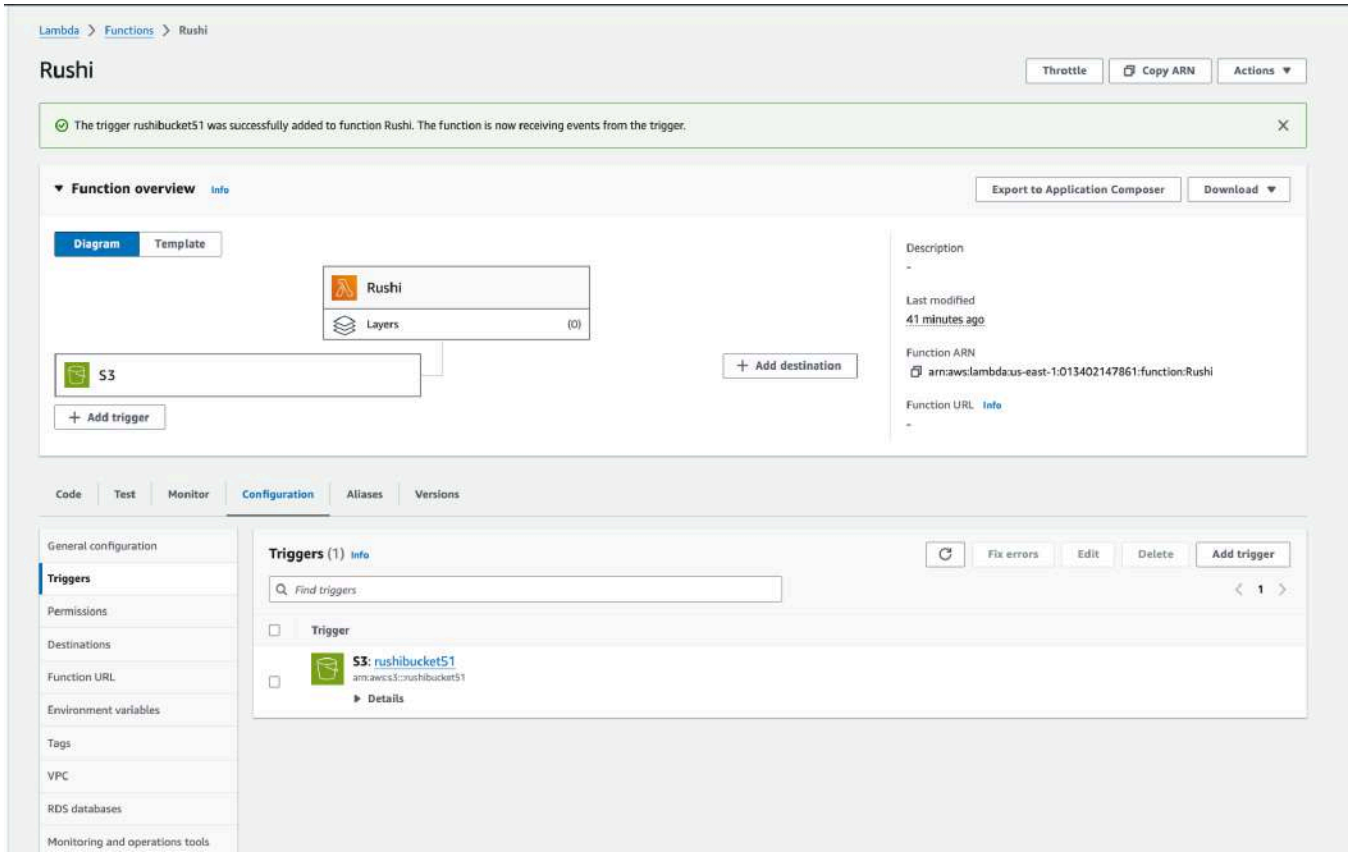
Name:Krushikesh Shelar 2024-2025 D15C Roll No:51



Step 7: Now In the Lambda function click on add tigger.
Now select the source as S3 then select the bucket name from the dropdown, keep other things to default and also you can add prefix to image.



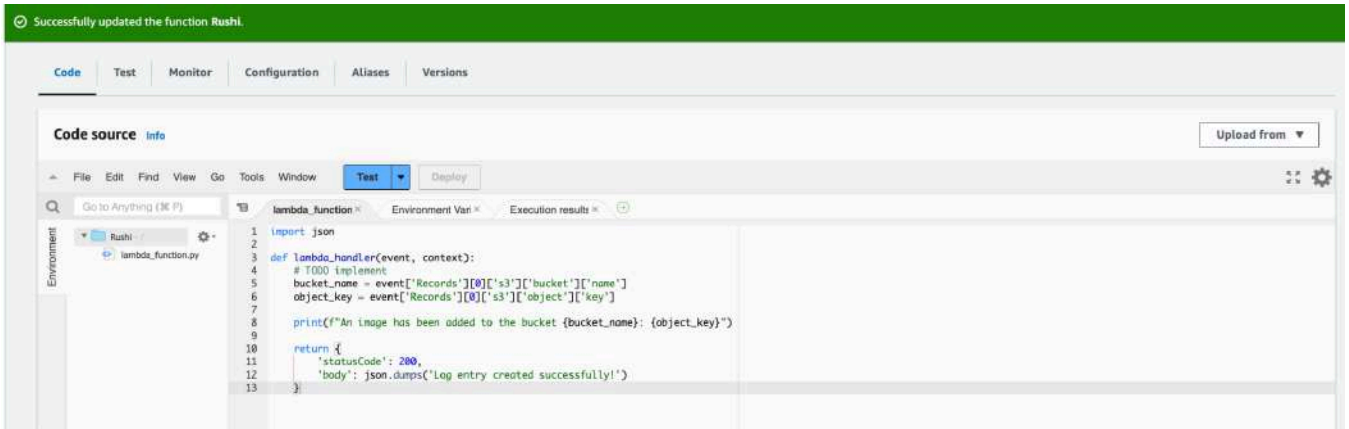
Name:Krushikesh Shelar 2024-2025 D15C Roll No:51



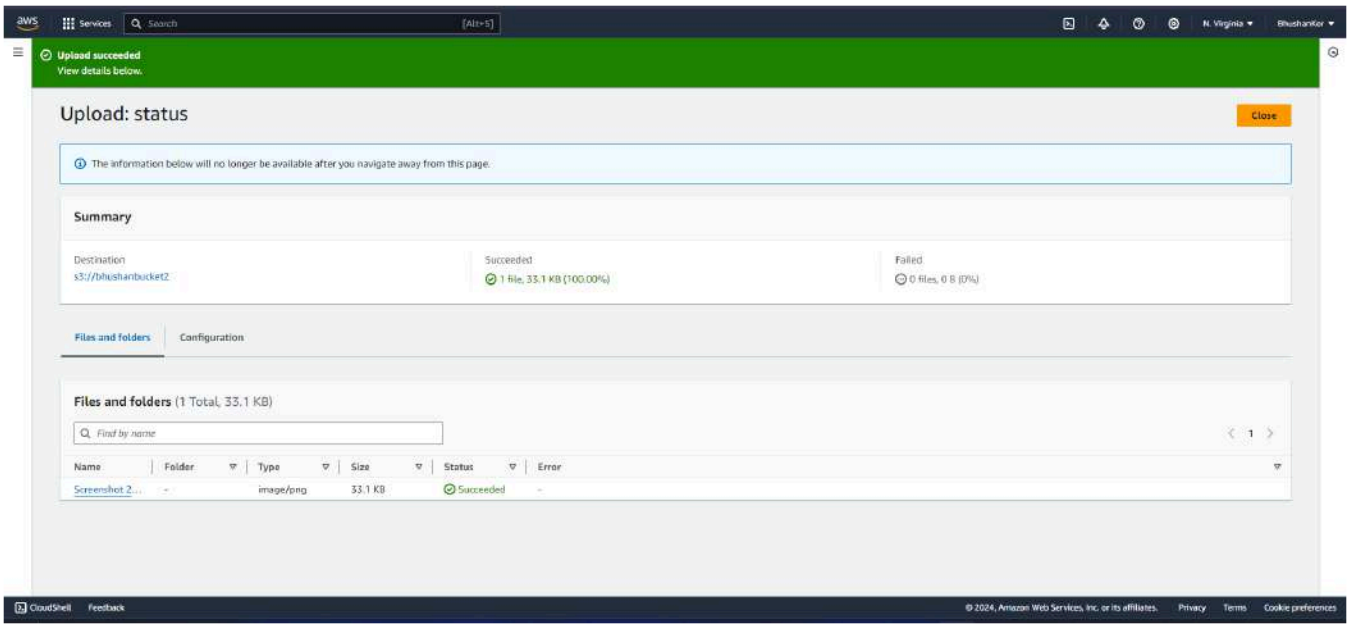
Step 8: Now Write code that logs a message like “An Image has been added” when triggered. Save the file and click on deploy.

```
import json

def lambda_handler(event, context):

    # TODO implement

    bucket_name = event['Records'][0]['s3']['bucket']['name']

    object_key = event['Records'][0]['s3']['object']['key']

    print(f"An image has been added to the bucket {bucket_name}: {object_key}")

    return {

        'statusCode': 200,

        'body': json.dumps('Log entry created successfully!')

    }
```

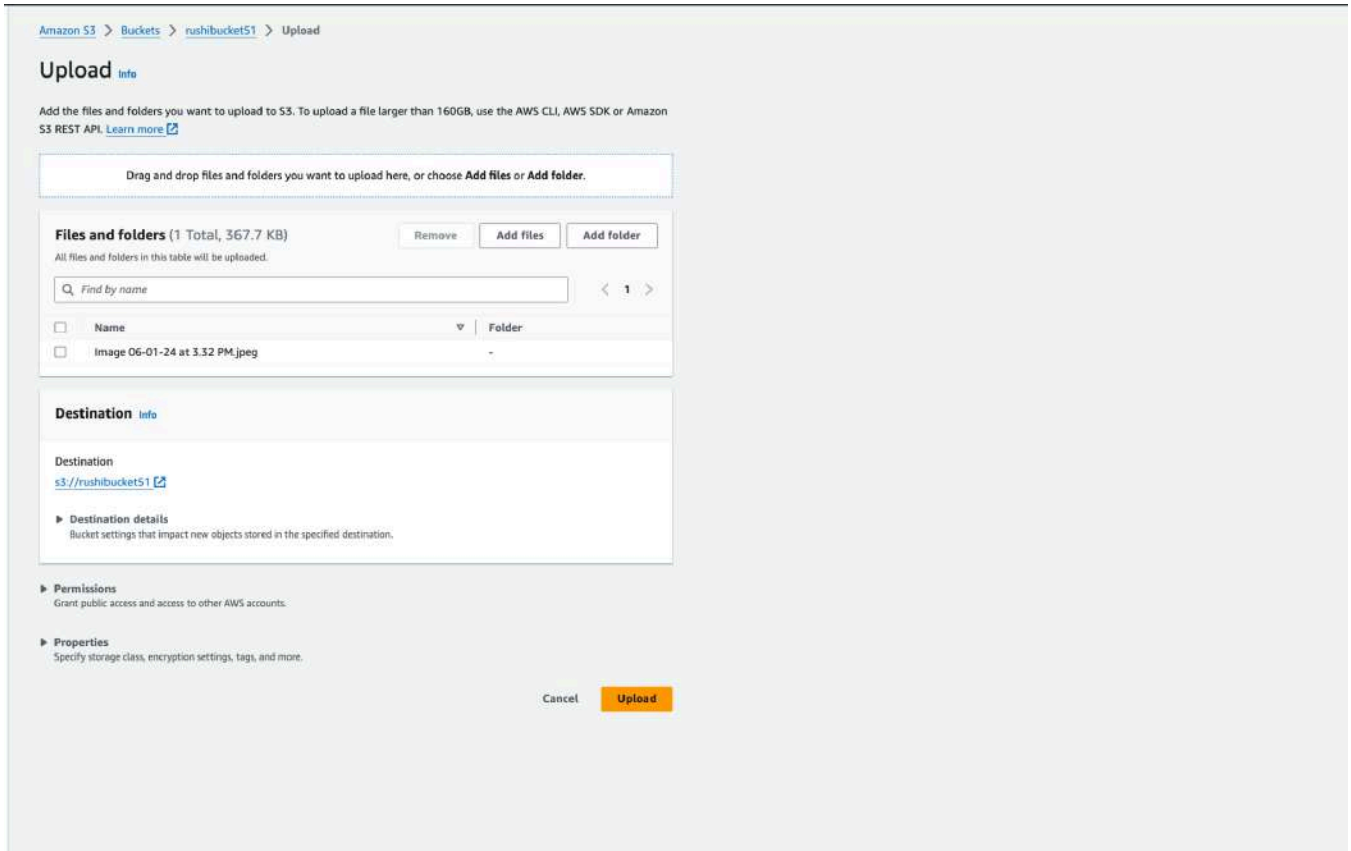
Name:Krushikesh Shelar 2024-2025 D15C Roll No:51

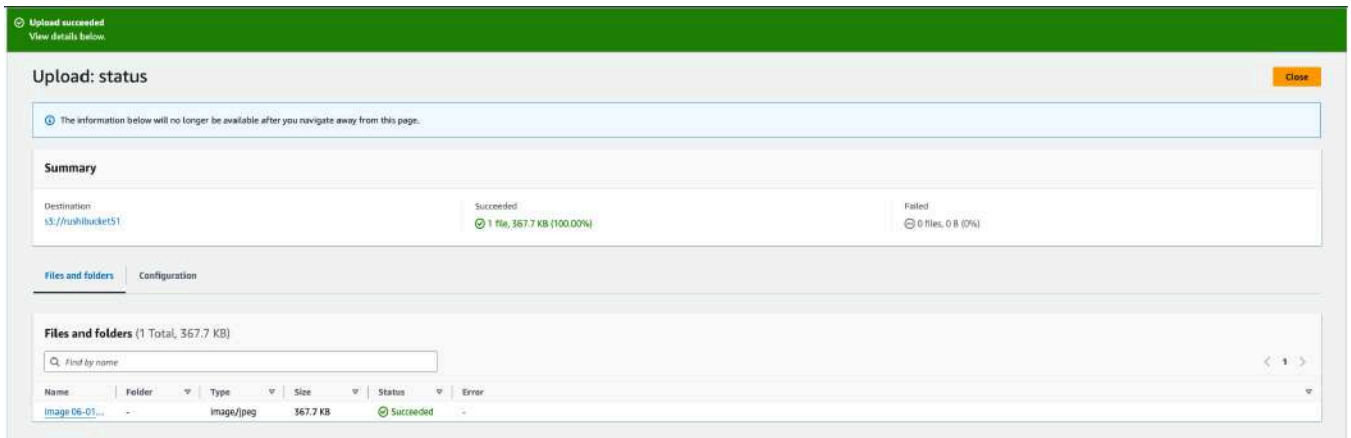


Step 9: Now upload any image to the bucket.



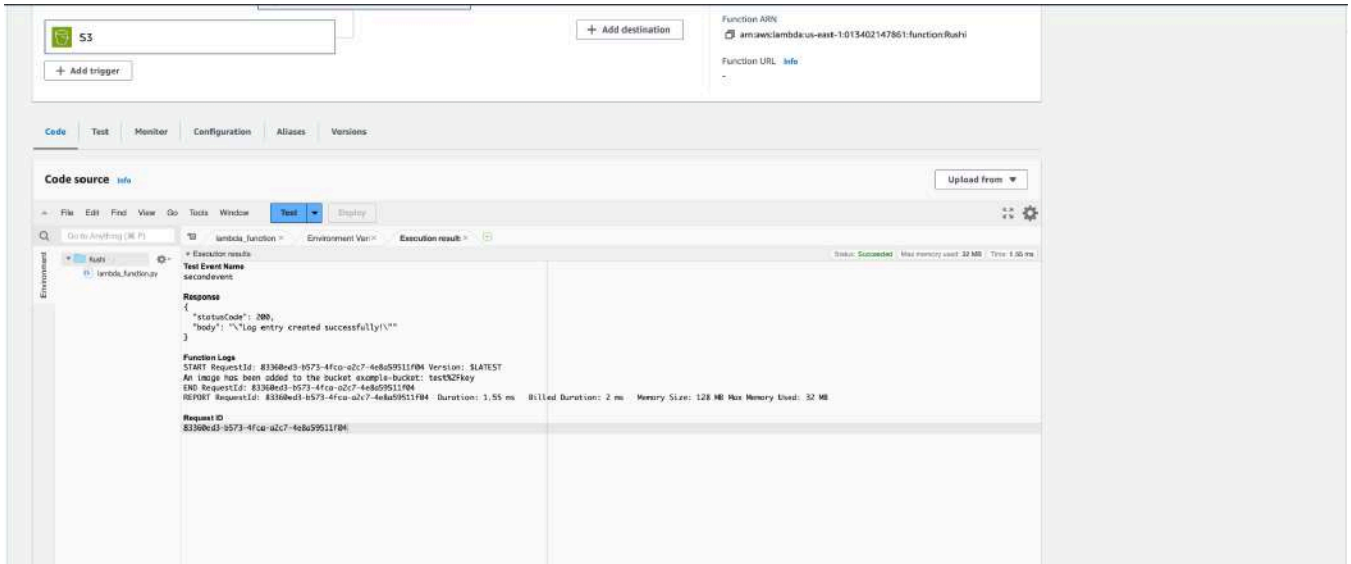
Name:Krushikesh Shelar 2024-2025 D15C Roll No:51



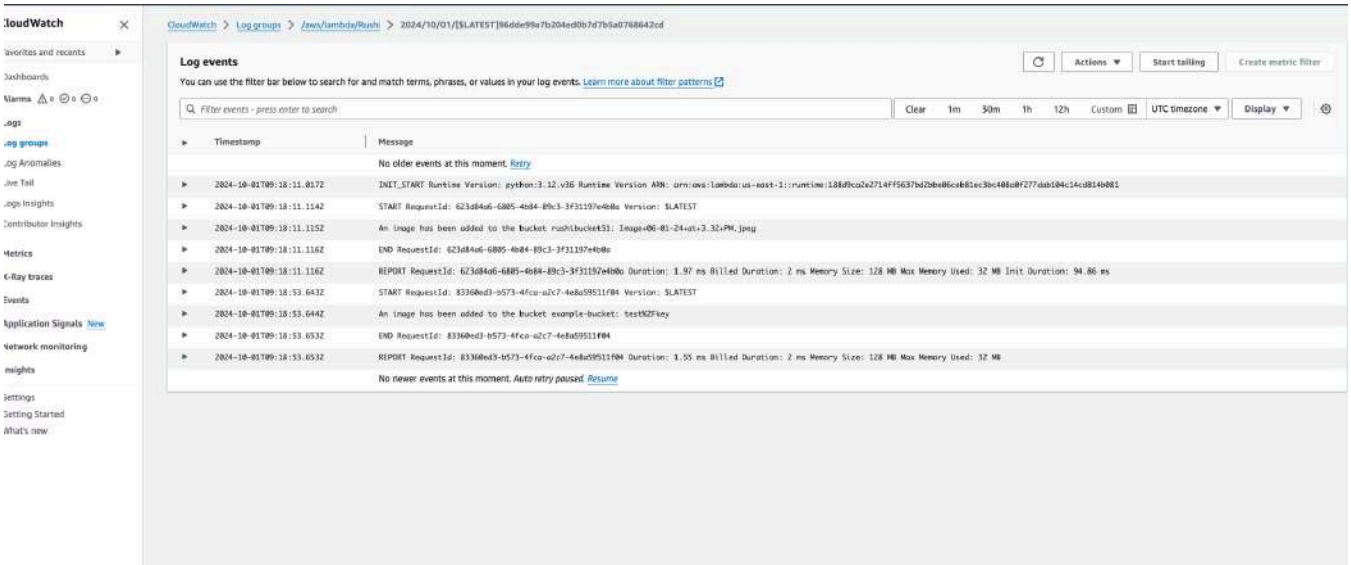


Step 10: Now to click on test in lambda to check whether it is giving log when image is added to S3.

Name:Krushikesh Shelar 2024-2025 D15C Roll No:51



Step 11: Now Lets see the log on Cloud watch.To see it go to monitor section and then click on view cloudwatch logs.



Conclusion: In this experiment, we successfully created an AWS Lambda function that logs a message when an image is uploaded to an S3 bucket. It is important to note that we have to select S3-put template in event other wise code will give an error.The function was successfully triggered by S3 object uploads, validating the functionality of Lambda's event-driven architecture. This experiment demonstrated how Lambda can efficiently respond to S3 events and how to troubleshoot common issues with event structure.