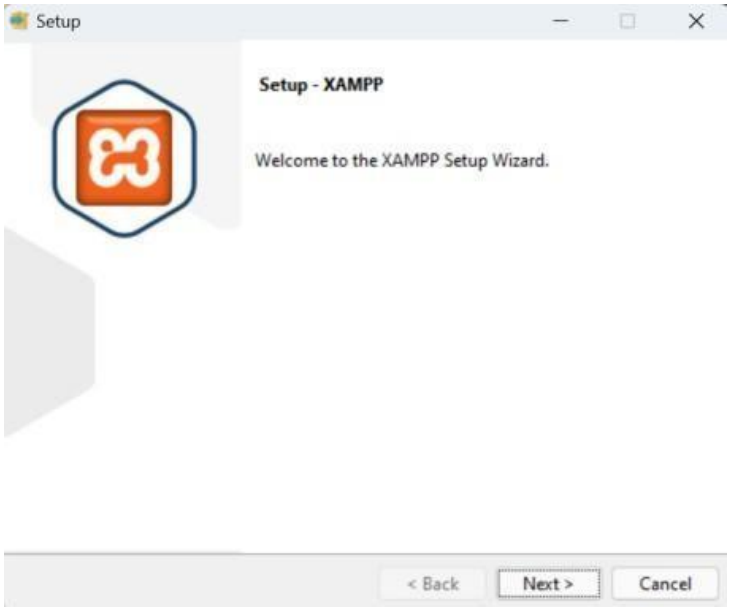
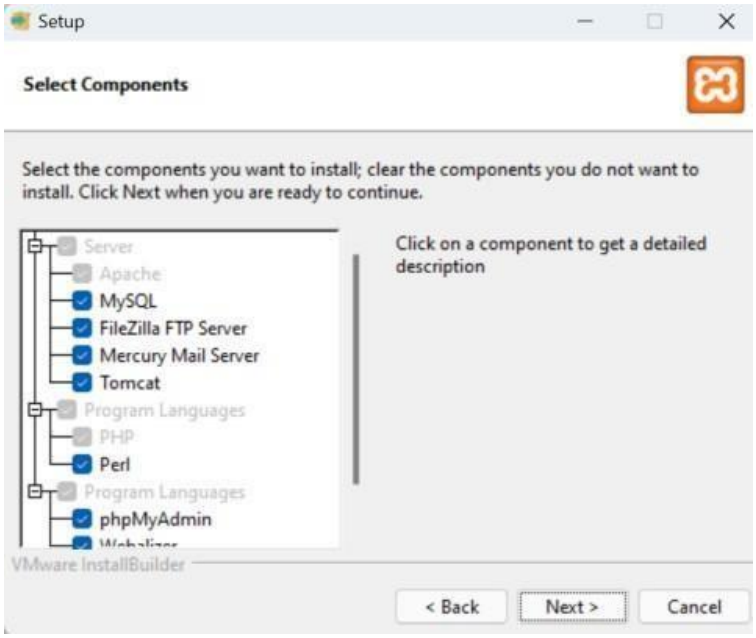
EXPERIMENT 1A

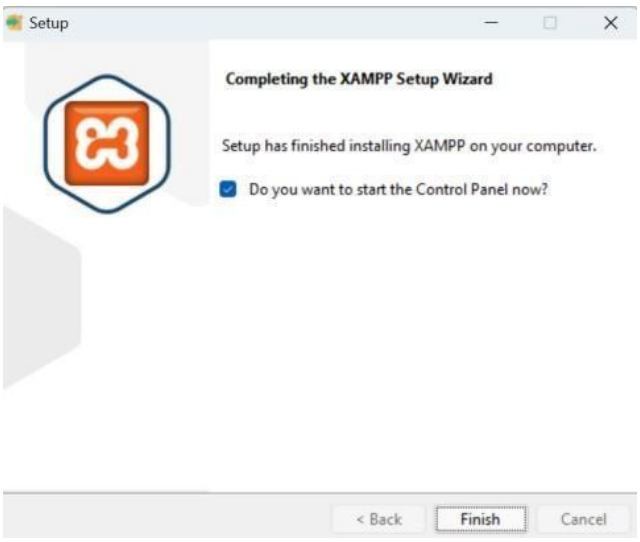
1. Hosting Website using Xampp

Install the required files from the internet and open up the setup

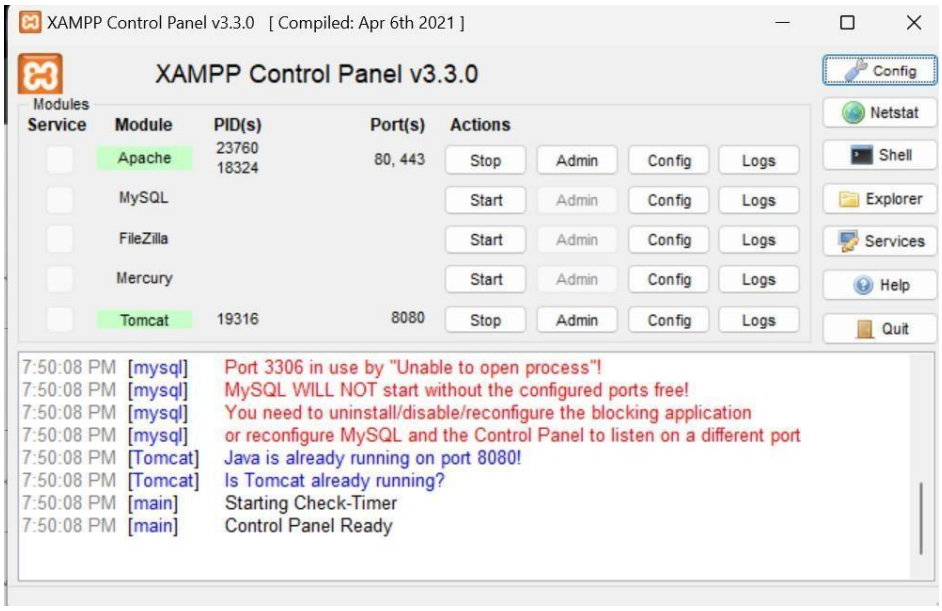


Make the changes according to your requirements



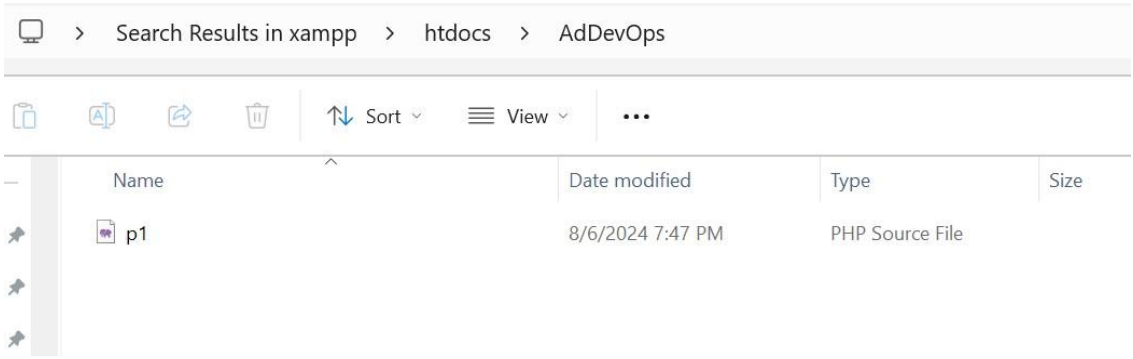


Open the control panel and click start for Apache



Write down your code and save it inside the htdocs folder in xampp files

```
<html>
<title>Lab 1 Roshan</title>
<body>
    <?php
    echo""D15C";
    ?>
```



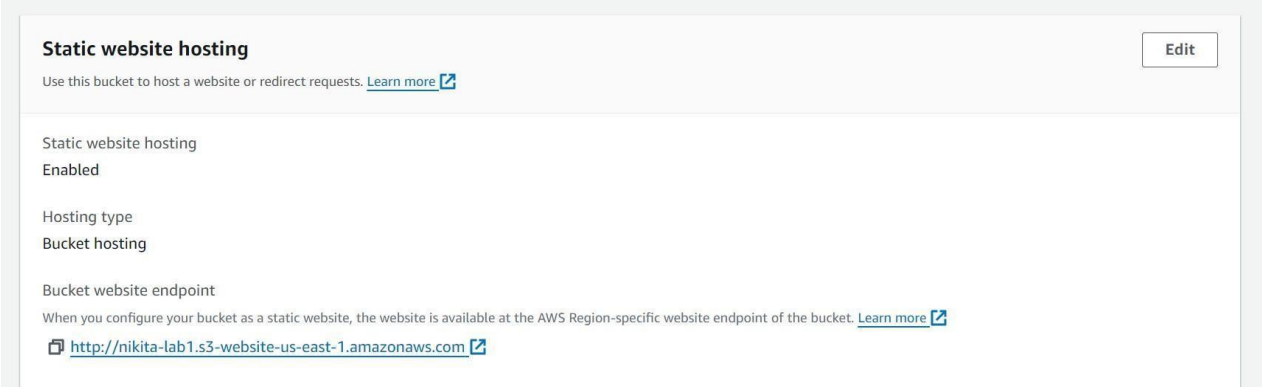
Go to your browser and type <localhost/filename/>

Hosting website with Amazon S3

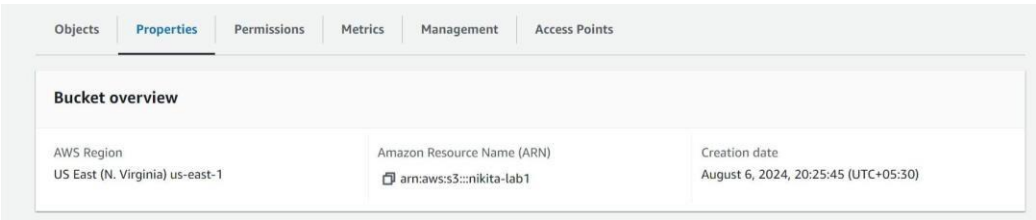
Open your learner lab's and Go to Amazon S3 in services

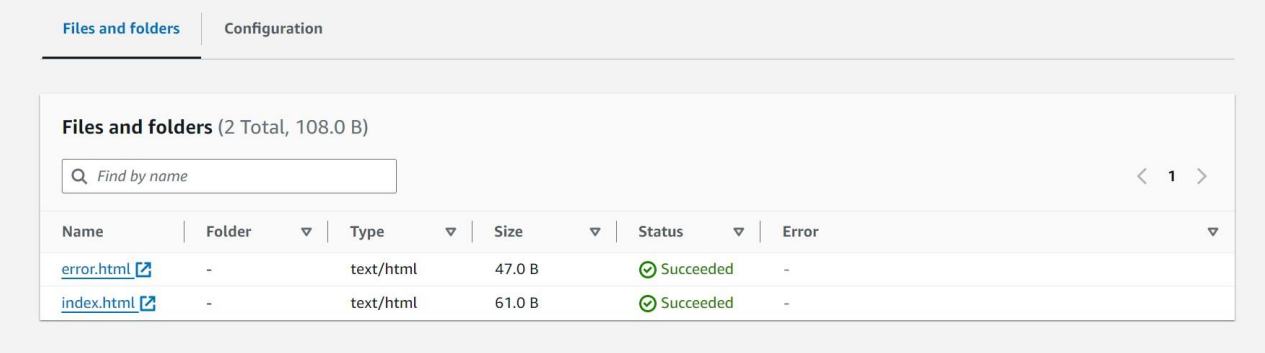
Click on create bucket and make a bucket with any name

In properties go to static website hosting and enable it

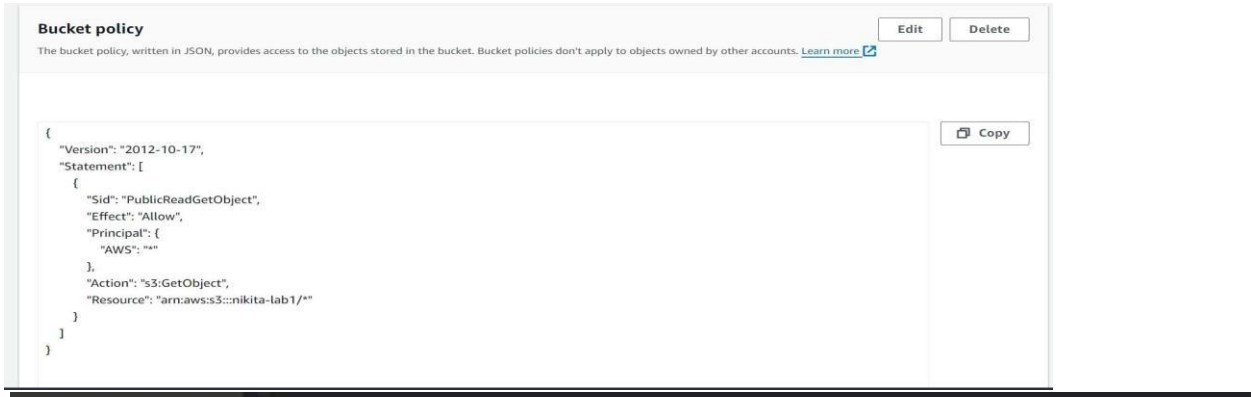


Go to Uploads option and upload the code files in them

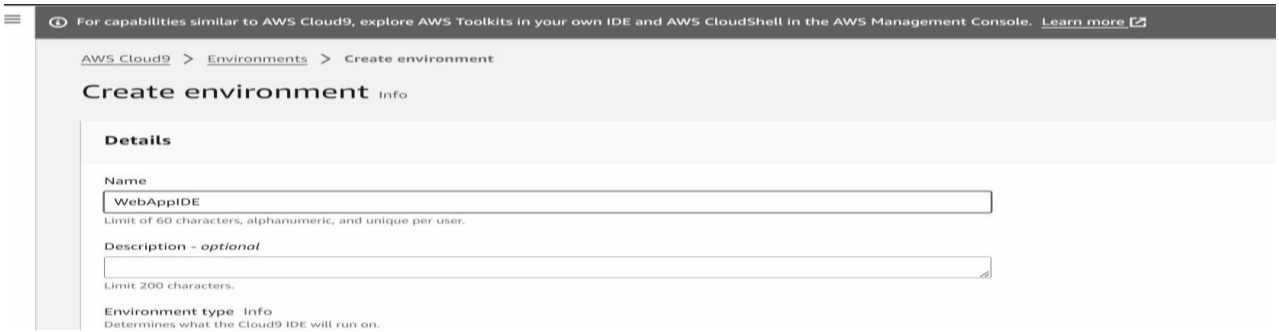


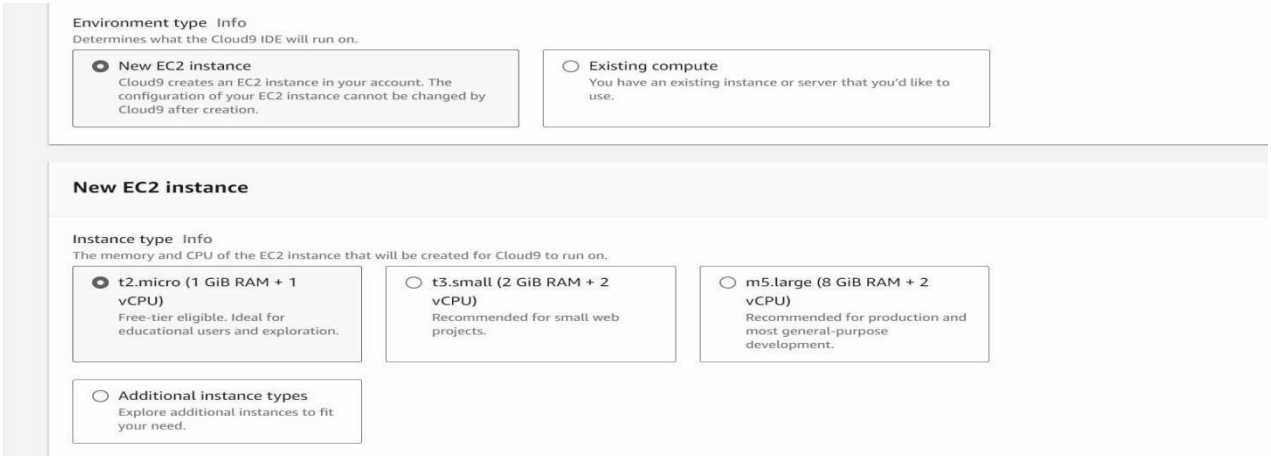


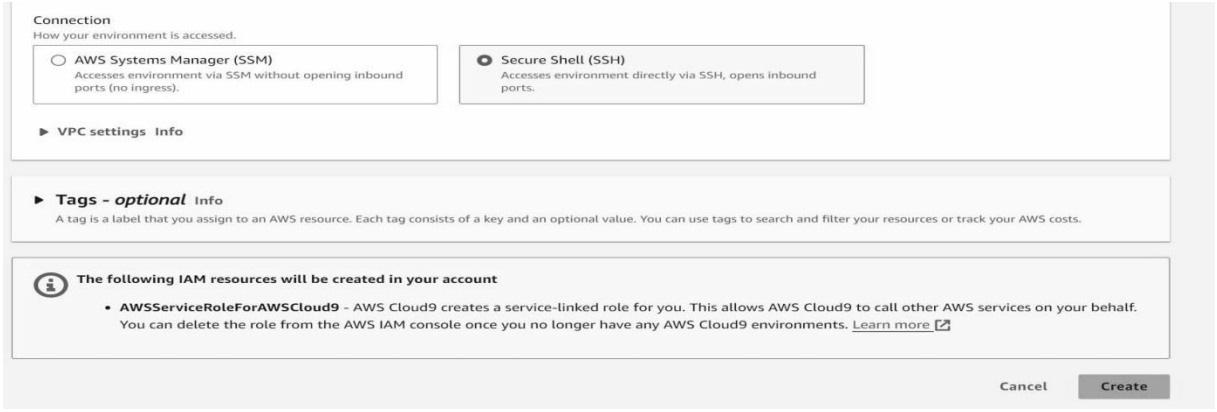
To host the website we need some permissions which can be changed using the following code

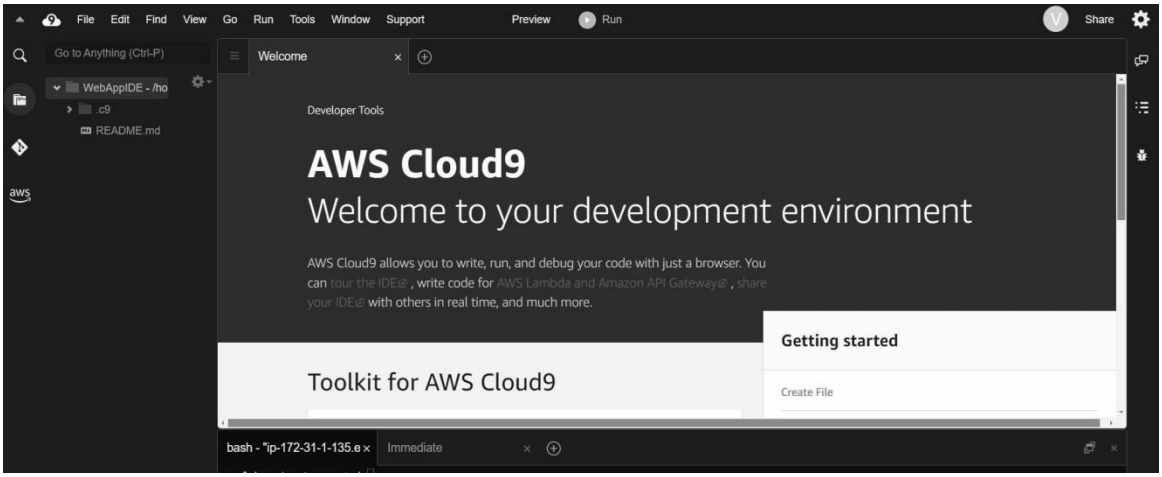


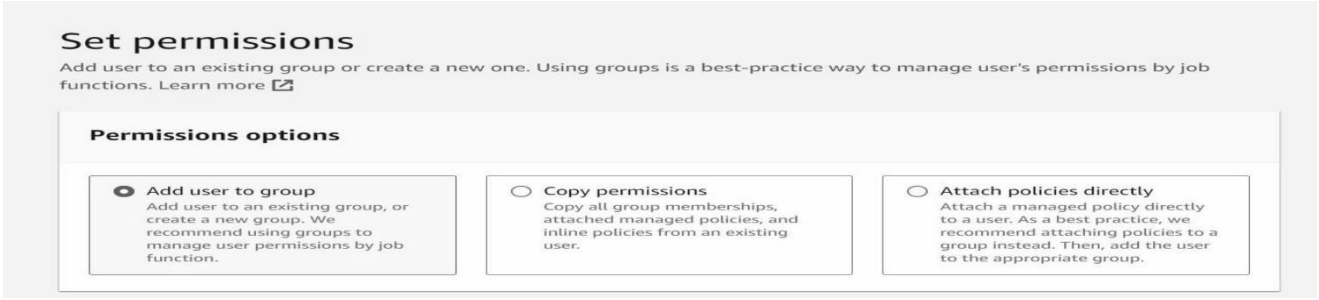
Experiment 1 B

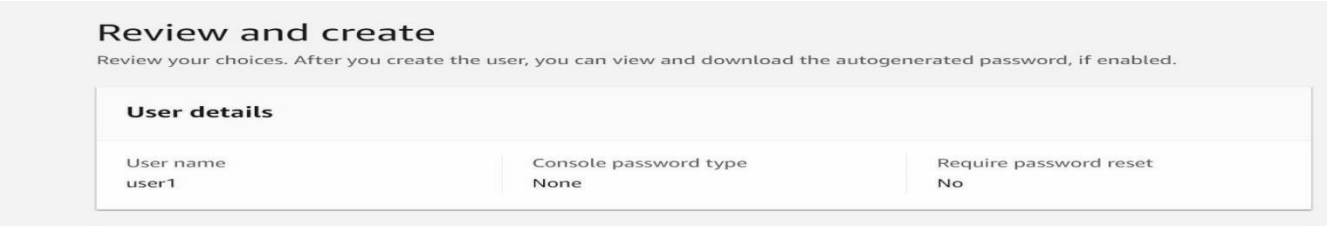




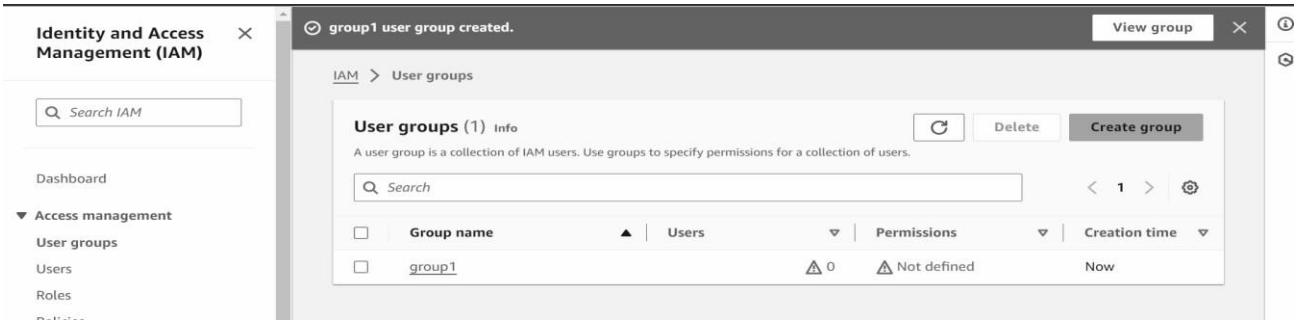

Connection
How your environment is accessed.
AWS Systems Manager (SSM)
Accesses environment via SSM without opening inbound ports (no ingress).
Secure Shell (SSH)
Accesses environment directly via SSH, opens inbound ports.
VPC settings Info
Tags - optional Info
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.
The following IAM resources will be created in your account
AWSServiceRoleForAWSCloud9 - AWS Cloud9 creates a service-linked role for you. This allows AWS Cloud9 to call other AWS services on your behalf. You can delete the role from the AWS IAM console once you no longer have any AWS Cloud9 environments. Learn more
Cancel
Create

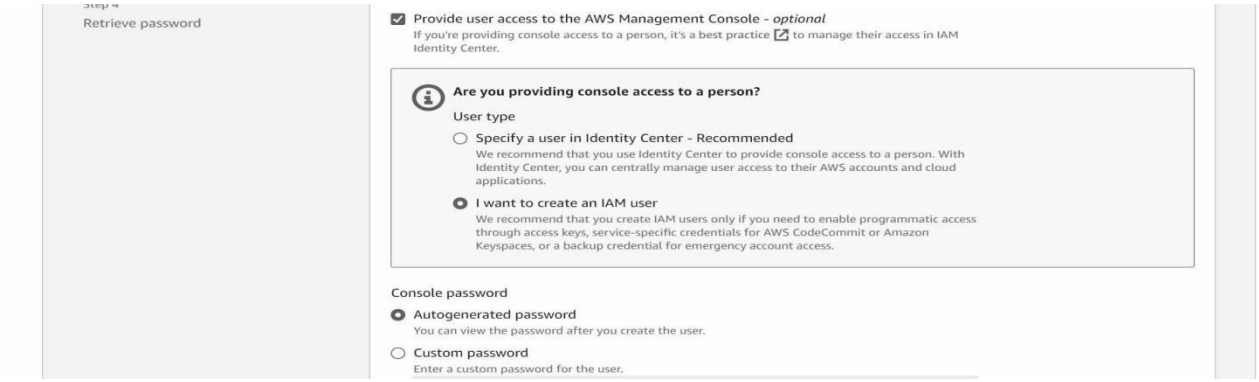

File Edit Find View Go Run Tools Window Support
Preview
Run
Share
Go to Anything (Ctrl-P)
WebAppIDE - /ho
.c9
README.md
Welcome
Developer Tools
AWS Cloud9
Welcome to your development environment
AWS Cloud9 allows you to write, run, and debug your code with just a browser. You can tour the IDE, write code for AWS Lambda and Amazon API Gateway, share your IDE with others in real time, and much more.
Getting started
Create File
Toolkit for AWS Cloud9
bash - "ip-172-31-1-135.e
Immediate

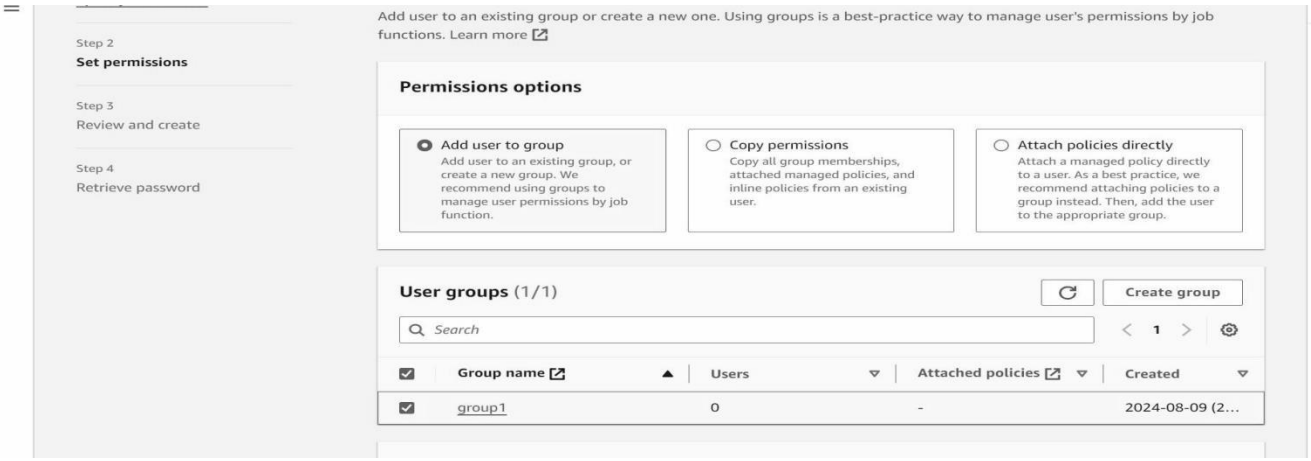

Set permissions
Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. Learn more
Permissions options
Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.
Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

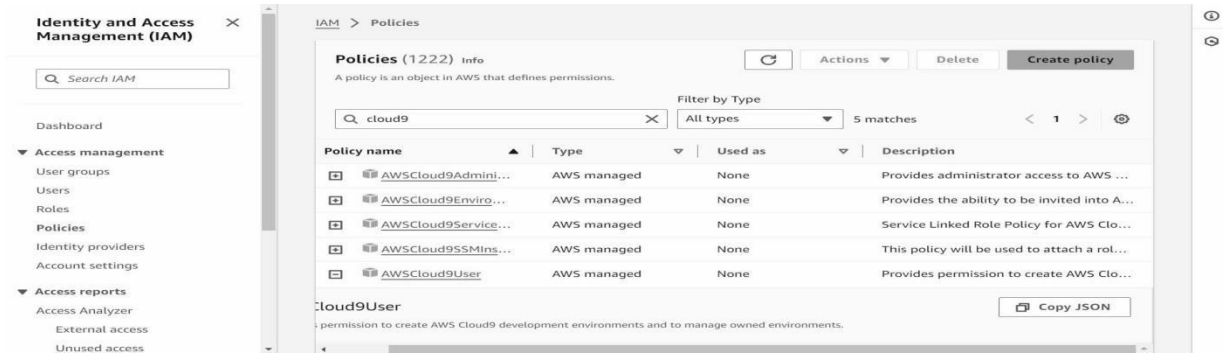

Review and create
Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.
User details
User name
user1
Console password type
None
Require password reset
No

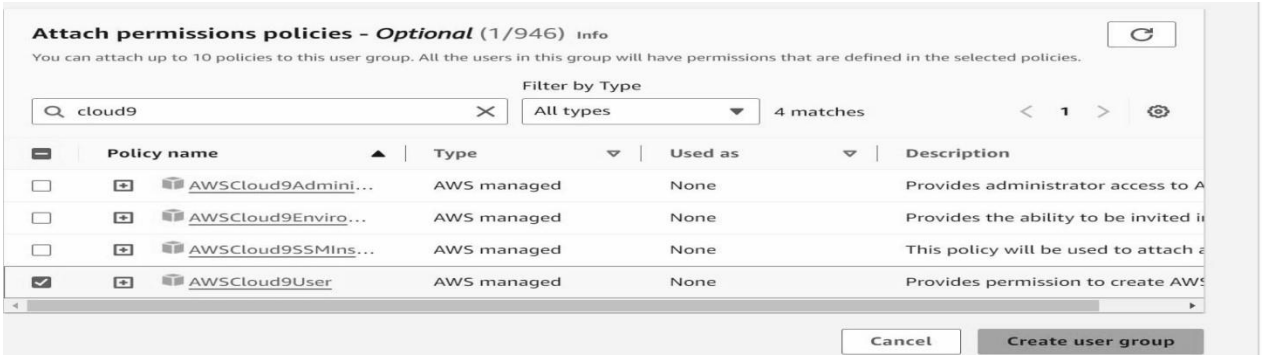

Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
IAM > User groups > Create user group
Create user group
Name the group
User group name
Enter a meaningful name to identify this group.
group1
Maximum 128 characters. Use alphanumeric and '+=,.@-_' characters.

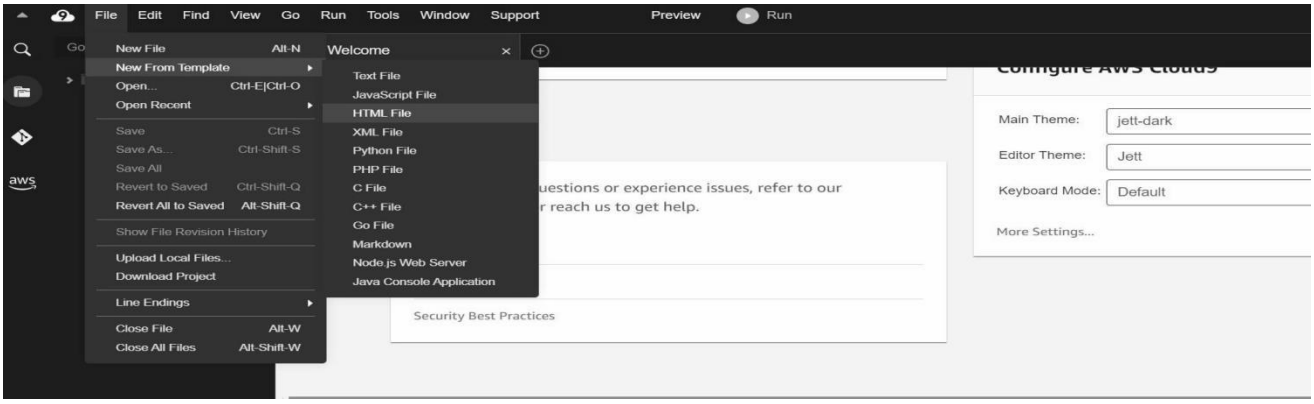

Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
group1 user group created.
View group
IAM > User groups
User groups (1) Info
Delete
Create group
A user group is a collection of IAM users. Use groups to specify permissions for a collection of users.
Search
1
Group name
Users
Permissions
Creation time
group1
0
Not defined
Now

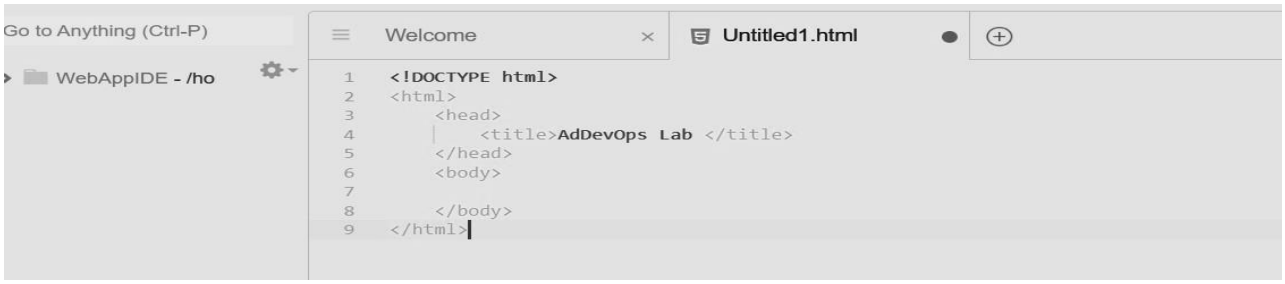

Retrieve password
Provide user access to the AWS Management Console - optional
If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.
Are you providing console access to a person?
User type
Specify a user in Identity Center - Recommended
We recommend that you use Identity Center to provide console access to a person. With Identity Center, you can centrally manage user access to their AWS accounts and cloud applications.
I want to create an IAM user
We recommend that you create IAM users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon Keyspaces, or a backup credential for emergency account access.
Console password
Autogenerated password
You can view the password after you create the user.
Custom password
Enter a custom password for the user.


Step 2
Set permissions
Step 3
Review and create
Step 4
Retrieve password
Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. Learn more
Permissions options
Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.
Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.
User groups (1/1)
Create group
Search
1
Group name
Users
Attached policies
Created
group1
0
-
2024-08-09 (2...


Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access Analyzer
External access
Unused access
IAM > Policies
Policies (1222) Info
A policy is an object in AWS that defines permissions.
Actions
Delete
Create policy
Filter by Type
cloud9
All types
5 matches
1
Policy name
Type
Used as
Description
AWSCloud9Admini...
AWS managed
None
Provides administrator access to AWS ...
AWSCloud9Enviro...
AWS managed
None
Provides the ability to be invited into A...
AWSCloud9Service...
AWS managed
None
Service Linked Role Policy for AWS Clo...
AWSCloud9SSMIns...
AWS managed
None
This policy will be used to attach a rol...
AWSCloud9User
AWS managed
None
Provides permission to create AWS Clo...
Copy JSON
permission to create AWS Cloud9 development environments and to manage owned environments.

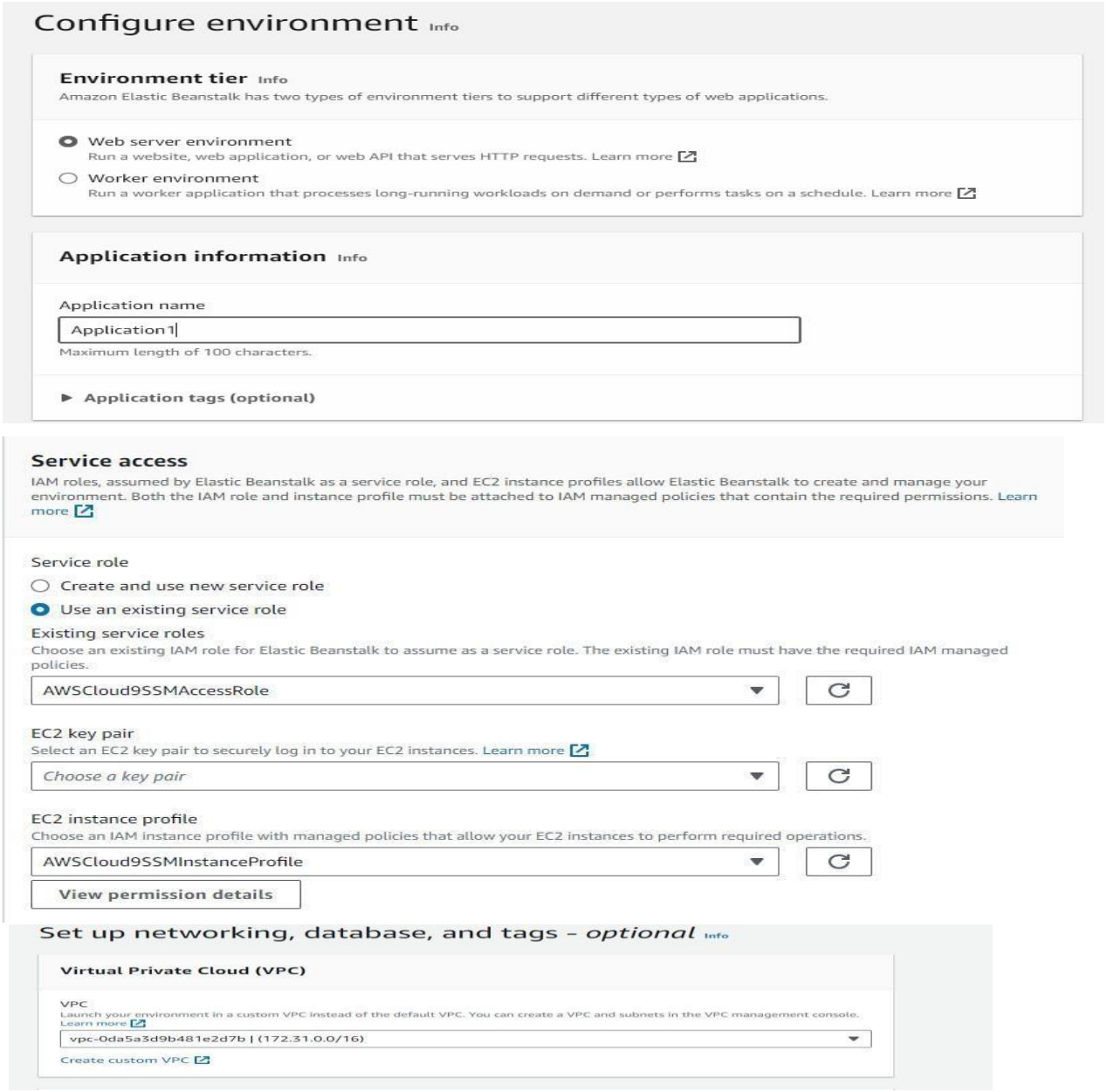

Attach permissions policies - Optional (1/946) Info
You can attach up to 10 policies to this user group. All the users in this group will have permissions that are defined in the selected policies.
Filter by Type
cloud9
All types
4 matches
Policy name
Type
Used as
Description
AWSCloud9Admini...
AWS managed
None
Provides administrator access to A
AWSCloud9Enviro...
AWS managed
None
Provides the ability to be invited i
AWSCloud9SSMIns...
AWS managed
None
This policy will be used to attach a
AWSCloud9User
AWS managed
None
Provides permission to create AWS
Cancel
Create user group


File Edit Find View Go Run Tools Window Support
Preview
Run
New File Alt-N
New From Template
Open... Ctrl-E|Ctrl-O
Open Recent
Save Ctrl-S
Save As... Ctrl-Shift-S
Save All
Revert to Saved Ctrl-Shift-Q
Revert All to Saved Alt-Shift-Q
Show File Revision History
Upload Local Files...
Download Project
Line Endings
Close File Alt-W
Close All Files Alt-Shift-W
Welcome
Text File
JavaScript File
HTML File
XML File
Python File
PHP File
C File
C++ File
Go File
Markdown
Node.js Web Server
Java Console Application
Main Theme: jett-dark
Editor Theme: Jett
Keyboard Mode: Default
More Settings...
Security Best Practices


Go to Anything (Ctrl-P)
WebAppIDE - /ho
Welcome
Untitled1.html
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>AdDevOps Lab </title>
5 </head>
6 <body>
7
8 </body>
9 </html>


Share this environment
Links to share
Environment:
https://us-east-1.console.aws.amazon.com/cloud9/ide/dfa17a27c775
Application:
44.200.154.228
To make your application accessible from the internet, please follow our documentation.
Who has access
ReadWrite
You (online)
RW
Don't allow members to save their tab state
Invite Members
R RW
Invite
Invite an existing IAM user or create a new user.

Experiment 2

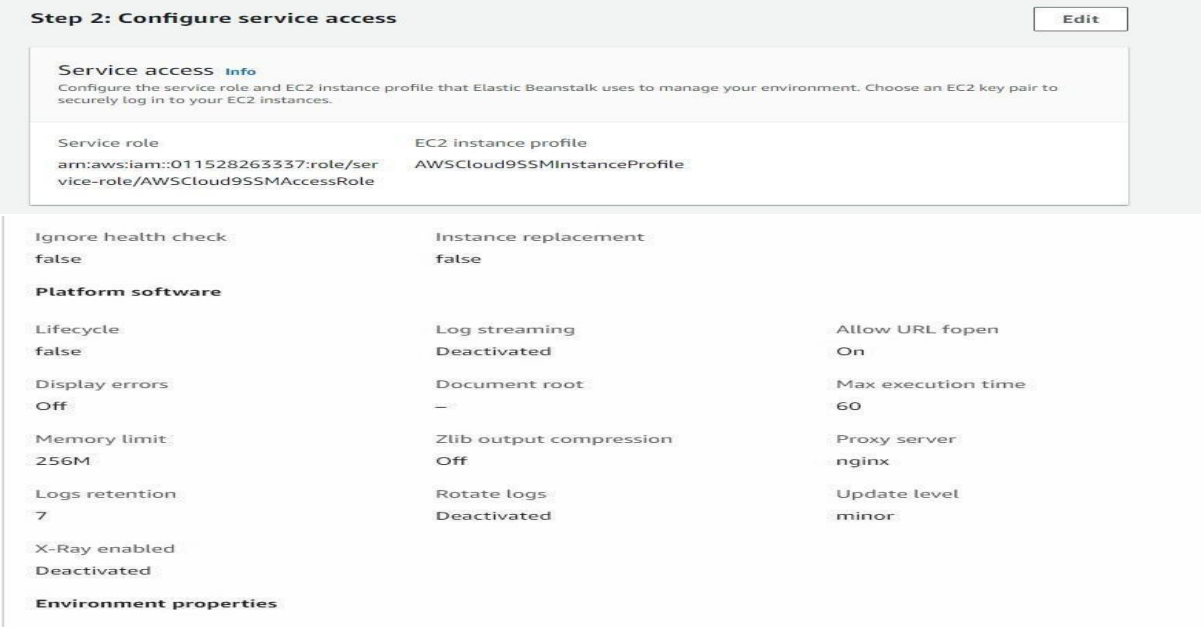


Review Info

Step 1: Configure environment Edit

Environment information

Environment tier Web server environment	Application name Application1
Environment name Application1-env	Application code Sample application
Platform arn:aws:elasticbeanstalk:us-east-1::platform/PHP 8.3 running on 64bit Amazon Linux 2023/4.3.1	



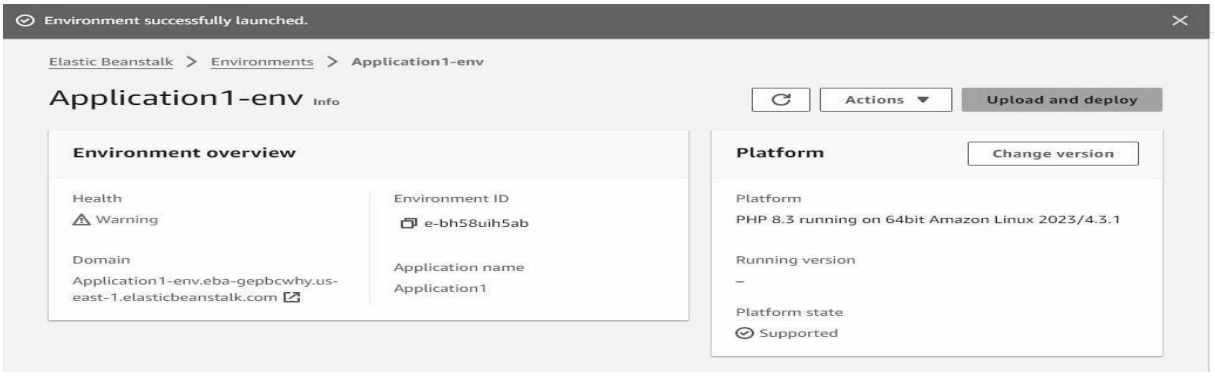
Step 2: Configure service access Edit

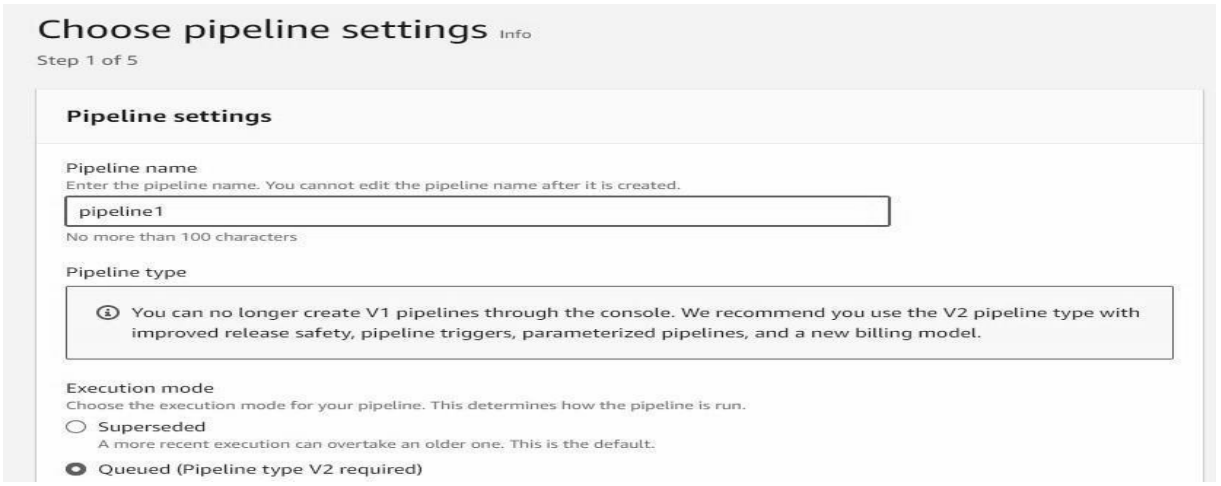
Service access Info

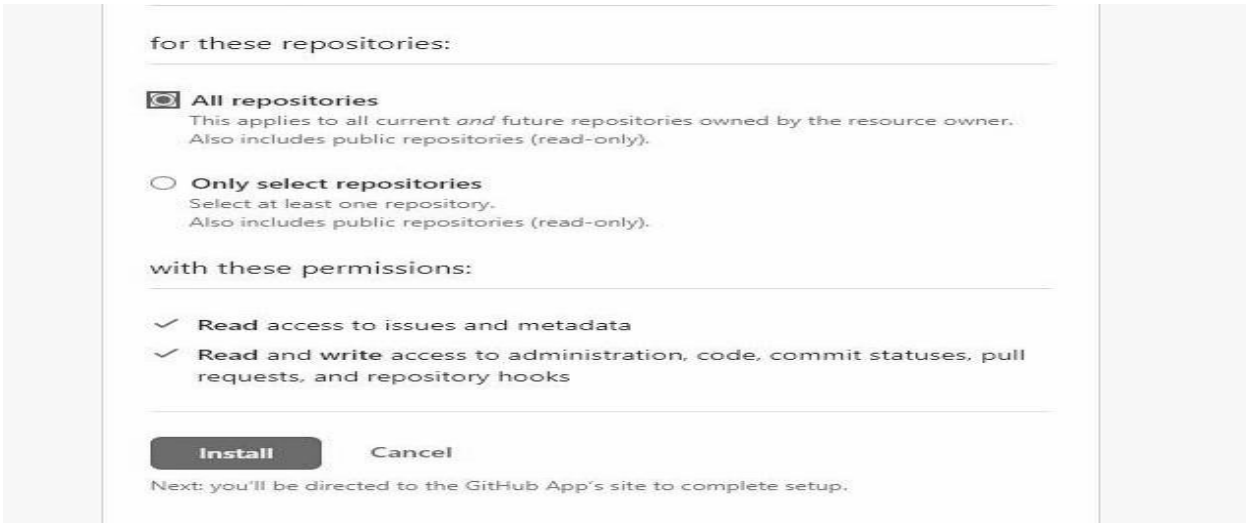
Configure the service role and EC2 instance profile that Elastic Beanstalk uses to manage your environment. Choose an EC2 key pair to securely log in to your EC2 instances.

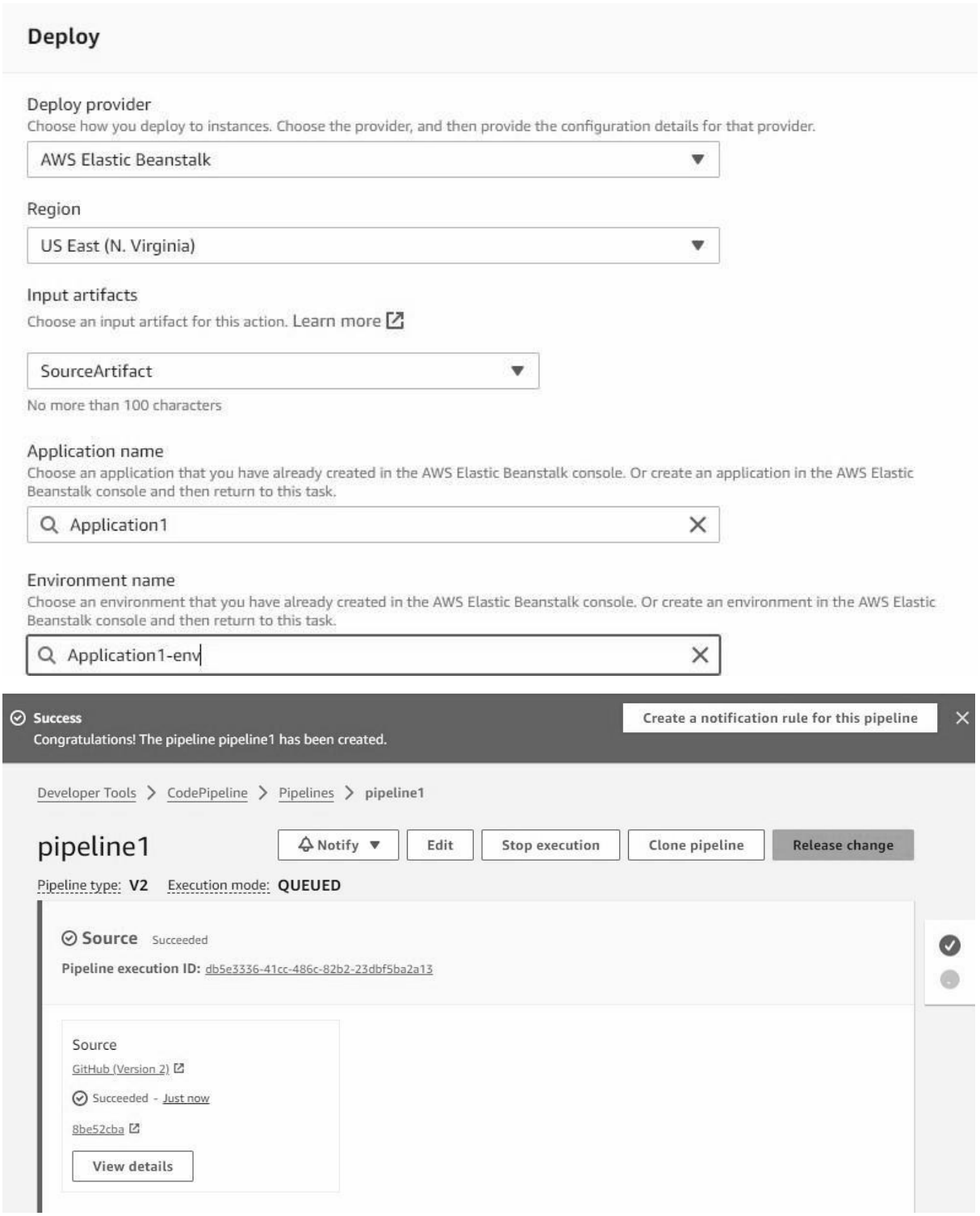
Service role arn:aws:iam::011528263337:role/service-role/AWSCloud9SSMAccessRole	EC2 instance profile AWSCloud9SSMInstanceProfile

Ignore health check false	Instance replacement false	
Platform software		
Lifecycle false	Log streaming Deactivated	Allow URL fopen On
Display errors Off	Document root –	Max execution time 60
Memory limit 256M	Zlib output compression Off	Proxy server nginx
Logs retention 7	Rotate logs Deactivated	Update level minor
X-Ray enabled Deactivated		
Environment properties		

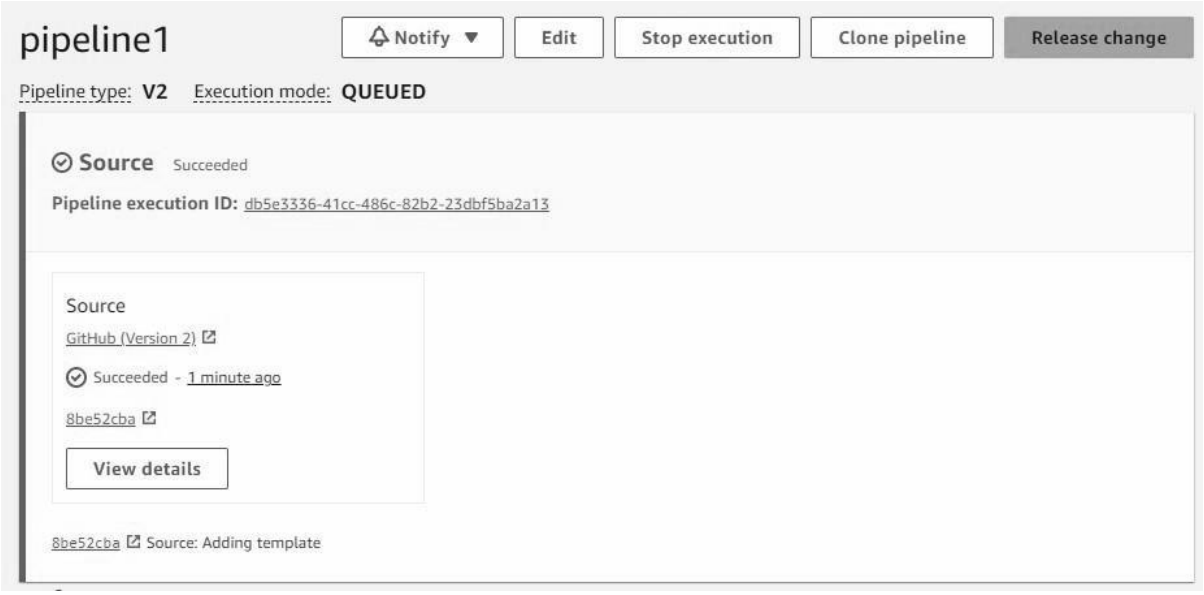

Environment successfully launched.
Elastic Beanstalk > Environments > Application1-env
Application1-env Info
Actions
Upload and deploy
Environment overview
Health
Warning
Environment ID
e-bh58uih5ab
Domain
Application1-env.eba-gepbcwhy.us-east-1.elasticbeanstalk.com
Application name
Application1
Platform
Change version
Platform
PHP 8.3 running on 64bit Amazon Linux 2023/4.3.1
Running version
–
Platform state
Supported


Choose pipeline settings Info
Step 1 of 5
Pipeline settings
Pipeline name
Enter the pipeline name. You cannot edit the pipeline name after it is created.
pipeline1
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)

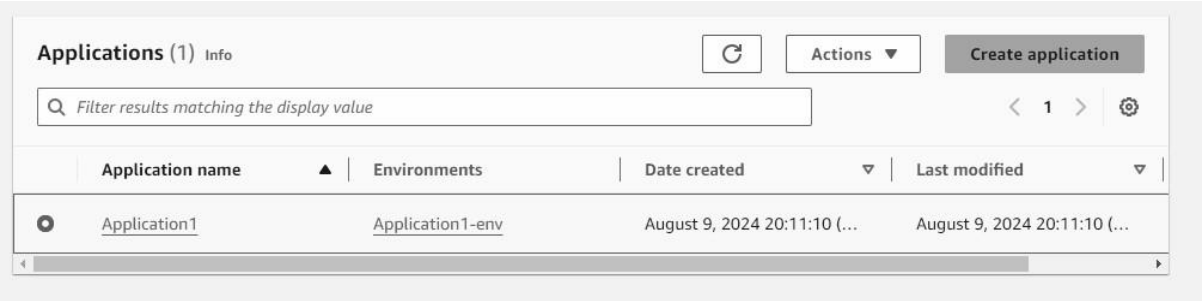

for these repositories:
All repositories
This applies to all current and future repositories owned by the resource owner.
Also includes public repositories (read-only).
Only select repositories
Select at least one repository.
Also includes public repositories (read-only).
with these permissions:
Read access to issues and metadata
Read and write access to administration, code, commit statuses, pull requests, and repository hooks
Install
Cancel
Next: you'll be directed to the GitHub App's site to complete setup.


Deploy
Deploy provider
Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.
AWS Elastic Beanstalk
Region
US East (N. Virginia)
Input artifacts
Choose an input artifact for this action. Learn more
SourceArtifact
No more than 100 characters
Application name
Choose an application that you have already created in the AWS Elastic Beanstalk console. Or create an application in the AWS Elastic Beanstalk console and then return to this task.
Application1
Environment name
Choose an environment that you have already created in the AWS Elastic Beanstalk console. Or create an environment in the AWS Elastic Beanstalk console and then return to this task.
Application1-env
Success
Congratulations! The pipeline pipeline1 has been created.
Create a notification rule for this pipeline
Developer Tools > CodePipeline > Pipelines > pipeline1
pipeline1
Notify
Edit
Stop execution
Clone pipeline
Release change
Pipeline type: V2
Execution mode: QUEUED
Source
Succeeded
Pipeline execution ID: db5e3336-41cc-486c-82b2-23dbf5ba2a13
Source
GitHub (Version 2)
Succeeded - Just now
8be52cba
View details


Deploy
In progress
Pipeline execution ID: db5e3336-41cc-486c-82b2-23dbf5ba2a13
Deploy
AWS Elastic Beanstalk
In progress - Just now
View details
8be52cba Source: Adding template


pipeline1
Notify
Edit
Stop execution
Clone pipeline
Release change
Pipeline type: V2 Execution mode: QUEUED
Source Succeeded
Pipeline execution ID: db5e3336-41cc-486c-82b2-23dbf5ba2a13
Source
GitHub (Version 2)
Succeeded - 1 minute ago
8be52cba
View details
8be52cba Source: Adding template


Deploy Succeeded
Start rollback
Pipeline execution ID: db5e3336-41cc-486c-82b2-23dbf5ba2a13
Deploy
AWS Elastic Beanstalk
Succeeded - Just now
View details
8be52cba Source: Adding template



Congratulations!

You have successfully created a pipeline that retrieved this source application from an Amazon S3 bucket and deployed it to three Amazon EC2 instances using AWS CodeDeploy.

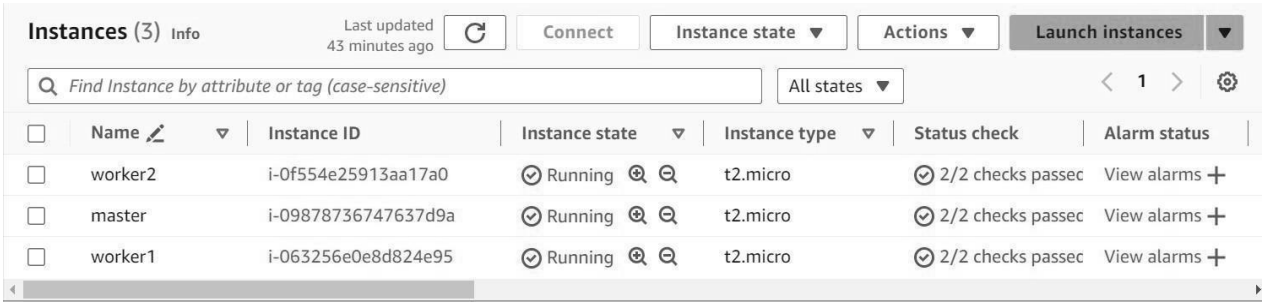
For next steps, read the AWS CodePipeline Documentation.

EXPERIMENT 3

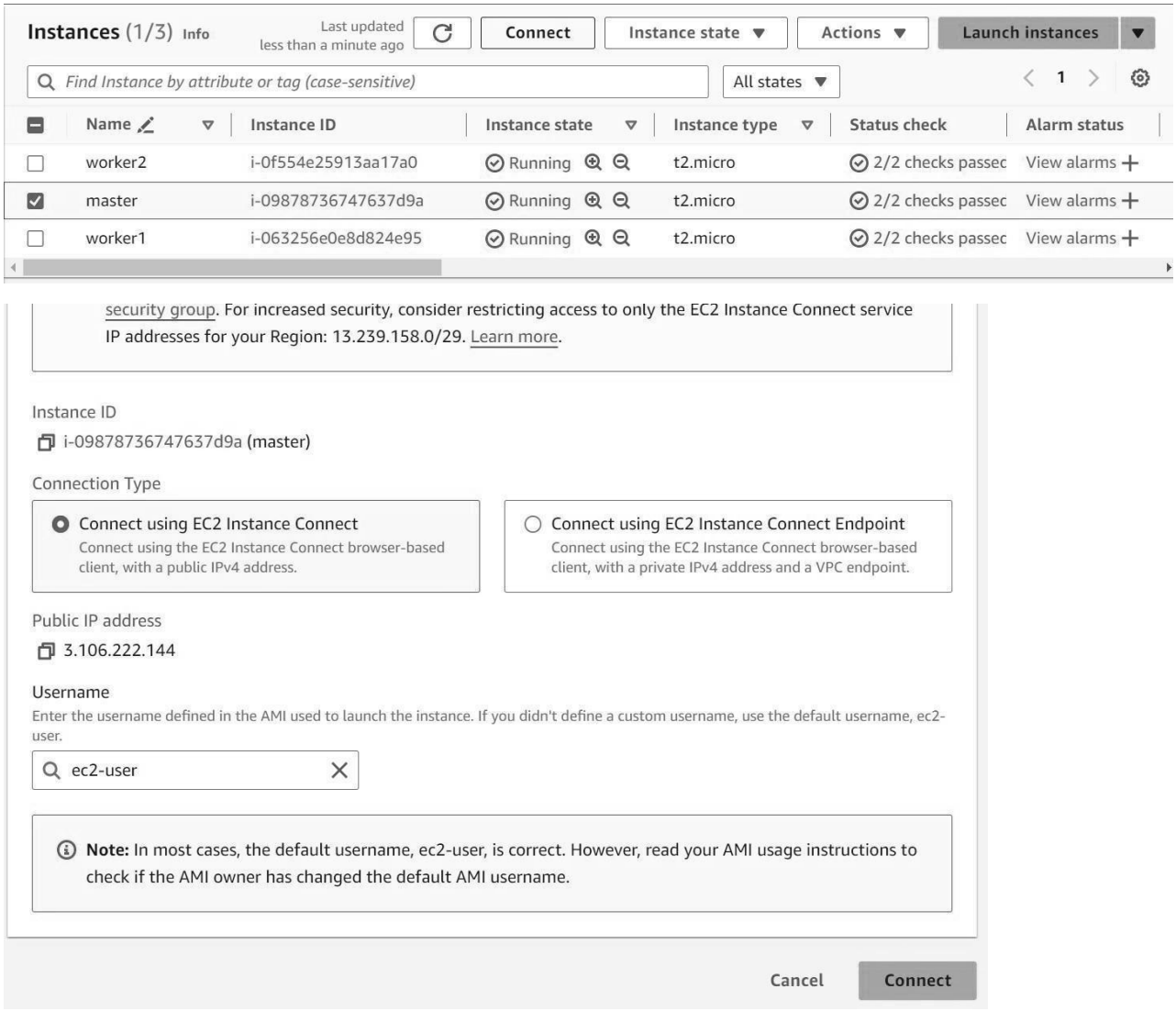
Aim: To understand the Kubernetes Cluster Architecture, install and Spin Up a Kubernetes Cluster on Linux Machines/Cloud Platforms.

Steps:

1. We will create 3 EC2 instances. One will be the master node and the other 2 will be slave/worker nodes.



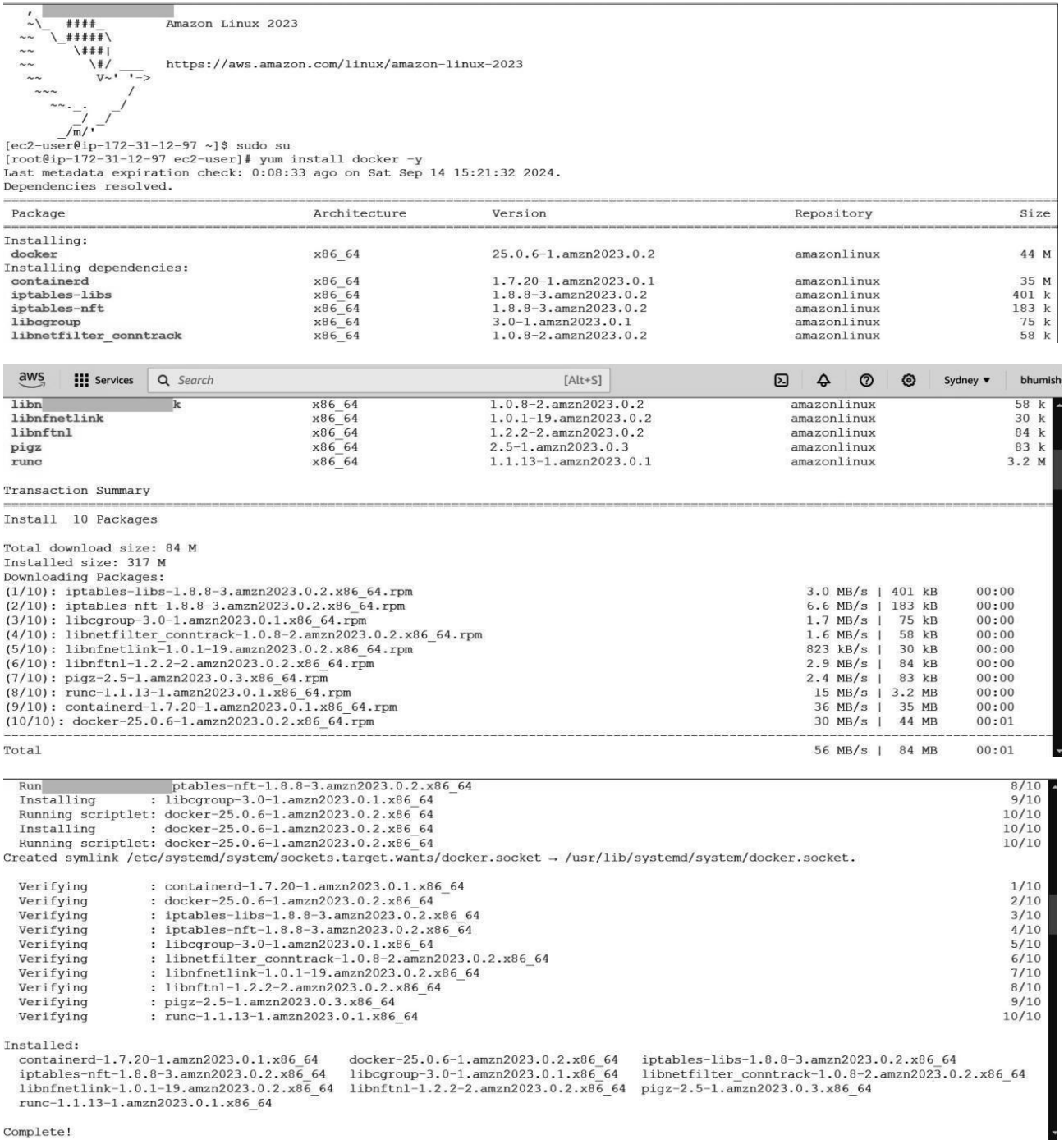
2. After the instances have been created, we will connect them one by one.



3. Docker installation:

   This step has to be performed on all the 3 instances. The following command has to be run:

   yum install docker -y



```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-12-97 ~]$ sudo su
[root@ip-172-31-12-97 ec2-user]# yum install docker -y
Last metadata expiration check: 0:08:33 ago on Sat Sep 14 15:21:32 2024.
Dependencies resolved.
================================================================================
 Package                 Architecture    Version                 Repository    Size
================================================================================
Installing:
 docker                  x86_64          25.0.6-1.amzn2023.0.2   amazonlinux   44 M
Installing dependencies:
 containerd              x86_64          1.7.20-1.amzn2023.0.1   amazonlinux   35 M
 iptables-libs           x86_64          1.8.8-3.amzn2023.0.2    amazonlinux  401 k
 iptables-nft            x86_64          1.8.8-3.amzn2023.0.2    amazonlinux  183 k
 libcgroup               x86_64          3.0-1.amzn2023.0.1      amazonlinux   75 k
 libnetfilter_conntrack  x86_64          1.0.8-2.amzn2023.0.2    amazonlinux   58 k
```



```
aws  Services  Search [Alt+S]                                  Sydney ▾  bhumish
 libn                    k  x86_64  1.0.8-2.amzn2023.0.2   amazonlinux   58 k
 libnfnetlink            x86_64  1.0.1-19.amzn2023.0.2  amazonlinux   30 k
 libnftnl                x86_64  1.2.2-2.amzn2023.0.2   amazonlinux   84 k
 pigz                    x86_64  2.5-1.amzn2023.0.3     amazonlinux   83 k
 runc                    x86_64  1.1.13-1.amzn2023.0.1  amazonlinux  3.2 M

Transaction Summary
================================================================================
Install  10 Packages

Total download size: 84 M
Installed size: 317 M
Downloading Packages:
(1/10): iptables-libs-1.8.8-3.amzn2023.0.2.x86_64.rpm      3.0 MB/s | 401 kB  00:00
(2/10): iptables-nft-1.8.8-3.amzn2023.0.2.x86_64.rpm       6.6 MB/s | 183 kB  00:00
(3/10): libcgroup-3.0-1.amzn2023.0.1.x86_64.rpm            1.7 MB/s |  75 kB  00:00
(4/10): libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64 1.6 MB/s |  58 kB  00:00
(5/10): libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64.rpm      823 kB/s |  30 kB  00:00
(6/10): libnftnl-1.2.2-2.amzn2023.0.2.x86_64.rpm           2.9 MB/s |  84 kB  00:00
(7/10): pigz-2.5-1.amzn2023.0.3.x86_64.rpm                 2.4 MB/s |  83 kB  00:00
(8/10): runc-1.1.13-1.amzn2023.0.1.x86_64.rpm               15 MB/s | 3.2 MB  00:00
(9/10): containerd-1.7.20-1.amzn2023.0.1.x86_64.rpm         36 MB/s |  35 MB  00:00
(10/10): docker-25.0.6-1.amzn2023.0.2.x86_64.rpm            30 MB/s |  44 MB  00:01
--------------------------------------------------------------------------------
Total                                                       56 MB/s |  84 MB  00:01
```



```
  Run                    ptables-nft-1.8.8-3.amzn2023.0.2.x86_64                8/10
  Installing       : libcgroup-3.0-1.amzn2023.0.1.x86_64                        9/10
  Running scriptlet: docker-25.0.6-1.amzn2023.0.2.x86_64                       10/10
  Installing       : docker-25.0.6-1.amzn2023.0.2.x86_64                       10/10
  Running scriptlet: docker-25.0.6-1.amzn2023.0.2.x86_64                       10/10
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.

  Verifying        : containerd-1.7.20-1.amzn2023.0.1.x86_64                    1/10
  Verifying        : docker-25.0.6-1.amzn2023.0.2.x86_64                        2/10
  Verifying        : iptables-libs-1.8.8-3.amzn2023.0.2.x86_64                  3/10
  Verifying        : iptables-nft-1.8.8-3.amzn2023.0.2.x86_64                   4/10
  Verifying        : libcgroup-3.0-1.amzn2023.0.1.x86_64                        5/10
  Verifying        : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64         6/10
  Verifying        : libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64                  7/10
  Verifying        : libnftnl-1.2.2-2.amzn2023.0.2.x86_64                       8/10
  Verifying        : pigz-2.5-1.amzn2023.0.3.x86_64                             9/10
  Verifying        : runc-1.1.13-1.amzn2023.0.1.x86_64                         10/10

Installed:
  containerd-1.7.20-1.amzn2023.0.1.x86_64   docker-25.0.6-1.amzn2023.0.2.x86_64   iptables-libs-1.8.8-3.amzn2023.0.2.x86_64
  iptables-nft-1.8.8-3.amzn2023.0.2.x86_64  libcgroup-3.0-1.amzn2023.0.1.x86_64   libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64
  libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64 libnftnl-1.2.2-2.amzn2023.0.2.x86_64 pigz-2.5-1.amzn2023.0.3.x86_64
  runc-1.1.13-1.amzn2023.0.1.x86_64

Complete!
```

4. After successfully docker has been installed it has to be started on all machines by using the command “systemctl start docker”

```
Complete!
[root@ip-172-31-12-97 ec2-user]# systemctl start docker
```

5. Kubernetes installation:

   Search kubeadm installation on your browser and scroll down and select red hatbased distributions.

1. Set SELinux to `permissive` mode:

These instructions are for Kubernetes 1.31.

```
Linux in permissive mode (effectively disabling it)
enforce 0
 -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
```

```
# This overwrites any existing configuration in /etc/yum.repos.d/
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repom
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
```

3. Install kubelet, kubeadm and kubectl:

```
yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
```

4. (Optional) Enable the kubelet service before running kubeadm:

```
sudo systemctl enable --now kubelet
```

Copy the above given steps and paste in the terminal. This will create a Kubernetes repository, install kubelet, kubeadm and kubectl and also enable the services.

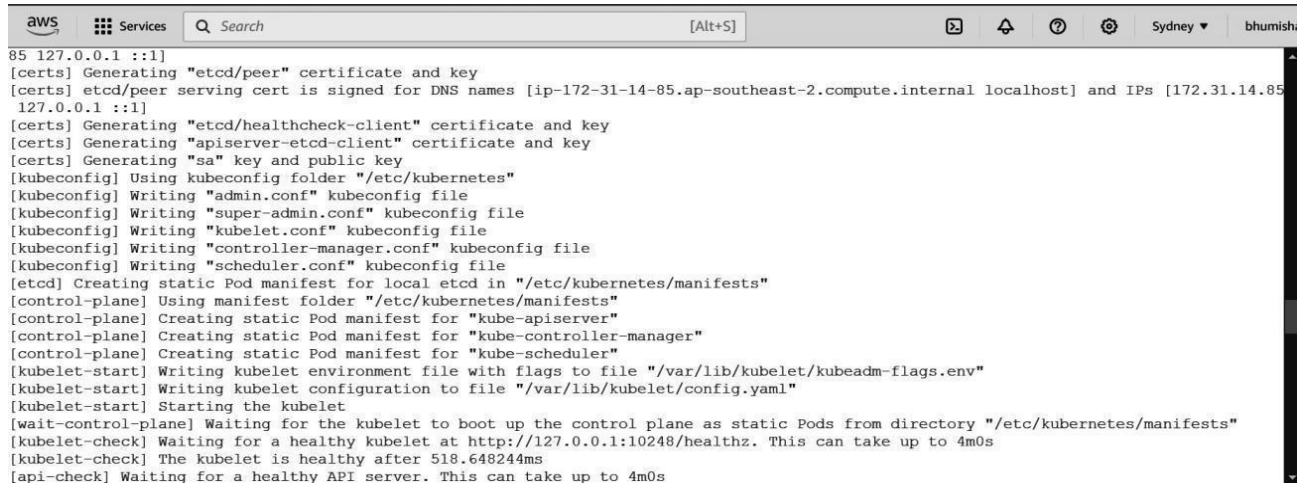
```
[root                ec2-user]# cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
[root@ip-172-31-12-97 ec2-user]# yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
Kubernetes                                                                    21 kB/s | 9.4 kB     00:00
Dependencies resolved.
================================================================================
 Package                  Architecture        Version                 Repository                Size
================================================================================
Installing:
 kubeadm                  x86_64              1.31.1-150500.1.1       kubernetes                11 M
 kubectl                  x86_64              1.31.1-150500.1.1       kubernetes                11 M
 kube                     x86_64              1.31.1-150500.1.1       kubernetes                15 M
Installing dependencies:
 conntrack-tools          x86_64              1.4.6-2.amzn2023.0.2    amazonlinux              208 k
 cri-tools                x86_64              1.31.1-150500.1.1       kubernetes               6.9 M
 kubernetes-cni           x86_64              1.5.1-150500.1.1        kubernetes               7.1 M
 libnetfilter_cthelper    x86_64              1.0.0-21.amzn2023.0.2   amazonlinux               24 k
 libnetfilter_cttimeout   x86_64              1.0.0-19.amzn2023.0.2   amazonlinux               24 k
 libnetfilter_queue       x86_64              1.0.5-2.amzn2023.0.2    amazonlinux               30 k

Transaction Summary
================================================================================
Install  9 Packages

Total download size: 51 M
Installed size: 269 M
Downloading Packages:
(1/9): libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64.rpm   500 kB/s |  24 kB     00:00
(2/9): libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64.rpm  475 kB/s |  24 kB     00:00
(3/9): conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64.rpm          3.6 MB/s | 208 kB     00:00
(4/9): libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64.rpm       1.4 MB/s |  30 kB     00:00
(5/9): kubeadm-1.31.1-150500.1.1.x86_64.rpm                      17 MB/s |  11 MB     00:00
(6/9): kubectl-1.31.1-150500.1.1.x86_64.rpm                      15 MB/s |  11 MB     00:00
(7/9): cri-tools-1.31.1-150500.1.1.x86_64.rpm                   8.0 MB/s | 6.9 MB     00:00
(8/9): kubernetes-cni-1.5.1-150500.1.1.x86_64.rpm                14 MB/s | 7.1 MB     00:00
(9/9): kubelet-1.31.1-150500.1.1.x86_64.rpm                      25 MB/s |  15 MB     00:00
  Ins                ibnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64                5/9
  Installing       : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64                      6/9
  Running scriptlet: conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64                      6/9
  Installing       : kubelet-1.31.1-150500.1.1.x86_64                                 7/9
  Running scriptlet: kubelet-1.31.1-150500.1.1.x86_64                                 7/9
  Installing       : kubeadm-1.31.1-150500.1.1.x86_64                                 8/9
  Installing       : kubectl-1.31.1-150500.1.1.x86_64                                 9/9
  Running scriptlet: kubectl-1.31.1-150500.1.1.x86_64                                 9/9
  Verifying        : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64                      1/9
  Verifying        : libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64               2/9
  Verifying        : libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64              3/9
  Verifying        : libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64                   4/9
  Verifying        : cri-tools-1.31.1-150500.1.1.x86_64                               5/9
  Verifying        : kubeadm-1.31.1-150500.1.1.x86_64                                 6/9
  Verifying        : kubectl-1.31.1-150500.1.1.x86_64                                 7/9
  Verifying        : kubelet-1.31.1-150500.1.1.x86_64                                 8/9
  Verifying        : kubernetes-cni-1.5.1-150500.1.1.x86_64                           9/9

Installed:
  conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64
  cri-tools-1.31.1-150500.1.1.x86_64
  kubeadm-1.31.1-150500.1.1.x86_64
  kubectl-1.31.1-150500.1.1.x86_64
  kubelet-1.31.1-150500.1.1.x86_64
  kubernetes-cni-1.5.1-150500.1.1.x86_64
  libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64
  libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64
  libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64
```

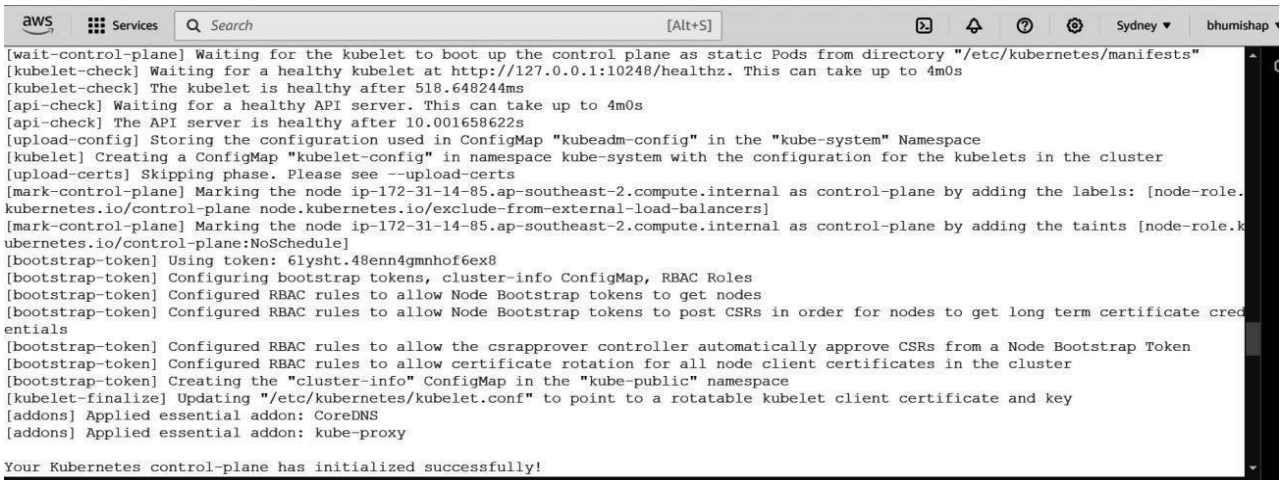
6. We can check if repository has been created by using yum repolist command.

```
[root@ip-172-31-14-85 ec2-user]# yum repolist
repo id                                                repo name
amazonlinux                                            Amazon Linux 2023 repository
kernel-livepatch                                       Amazon Linux 2023 Kernel Livepatch repository
kubernetes                                             Kubernetes
[root@ip-172-31-14-85 ec2-user]#
```

7. Now we will be initializing the kubeadm. For that “kubeadm init” command has to be used. It may show errors but those can be ignored by using --ignore-preflighterrors=all

```
[root@ip-172-31-14-85 ec2-user]# kubeadm init --ignore-preflight-errors=NumCPU --ignore-preflight-errors=Mem
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
        [WARNING NumCPU]: the number of available CPUs 1 is less than the required 2
        [WARNING Mem]: the system RAM (949 MB) is less than the minimum 1700 MB
        [WARNING FileExisting-socat]: socat not found in system path
        [WARNING FileExisting-tc]: tc not found in system path
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W0914 15:50:31.271160   29520 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inc
onsistent with that used by kubeadm.It is recommended to use "registry.k8s.io/pause:3.10" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-14-85.ap-southeast-2.compute.internal kubernetes kubernetes.default ku
bernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.14.85]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-14-85.ap-southeast-2.compute.internal localhost] and IPs [172.31.14.
85 127.0.0.1 ::1]
```





8. On successful initialization we need to copy and paste the following commands on the master machine itself:

```
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
```

9. Next copy and paste the join link in the worker nodes so that the worker nodes can join the cluster.

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

kubeadm join 172.31.14.85:6443 --token 61ysht.48enn4gmnhof6ex8 \
        --discovery-token-ca-cert-hash sha256:461819c971fe032e04a78e18fde8e28755825e8468d468a2c86d88c52dba4945
```

10. After performing join commands on the worker nodes, we will get following output:

```
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.
```

11. Once again when you run kubectl get nodes you will now see all 3 nodes have joined the cluster.

```
NAME                           STATUS     ROLES           AGE    VERSION
ip-172-31-85-89.ec2.internal   NotReady   control-plane   119s   v1.26.0
ip-172-31-89-46.ec2.internal   NotReady   <none>          19s    v1.26.0
ip-172-31-94-70.ec2.internal   NotReady   <none>          12s    v1.26.0
```

Conclusion:

This experiment successfully demonstrated the creation of a Kubernetes cluster and the successful addition of all three nodes using various commands. Errors encountered during initialization can be addressed in two ways: 1) by ignoring the errors, or 2) by

upgrading the instance type to t3.medium or t3.large if the issues are due to insufficient memory or CPU resources.

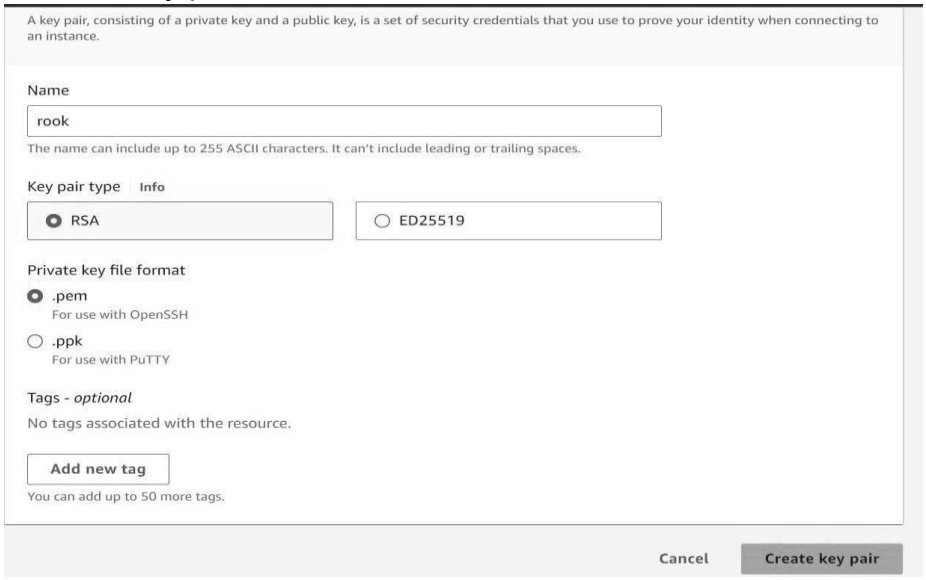
Experiment 4

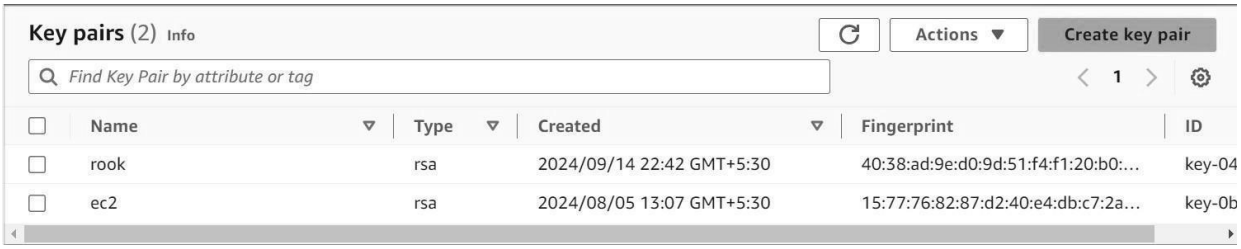
Aim:

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

Steps:

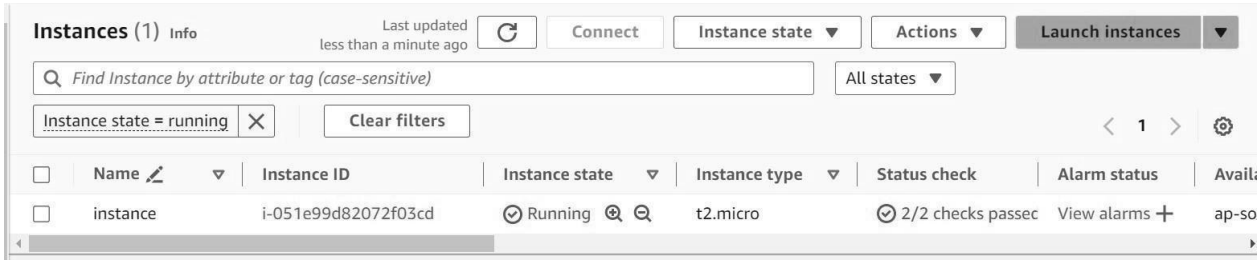
1. Create a key pair.



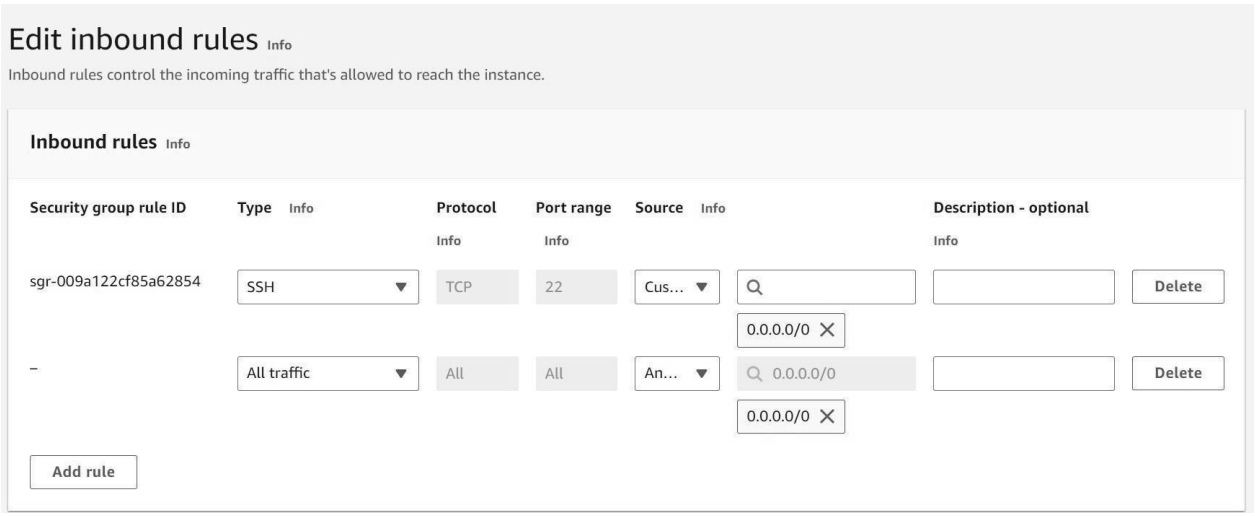


The .pem file will be downloaded on your machine and will be required in the further steps.

2. Now we will create an EC2 Ubuntu instance. Select the key pair which you just created while creating this instance.



3. Now edit the inbound rules to allow ssh.



4. Open git bash and go to the directory where pem file is located and use chmod to provide permissions.

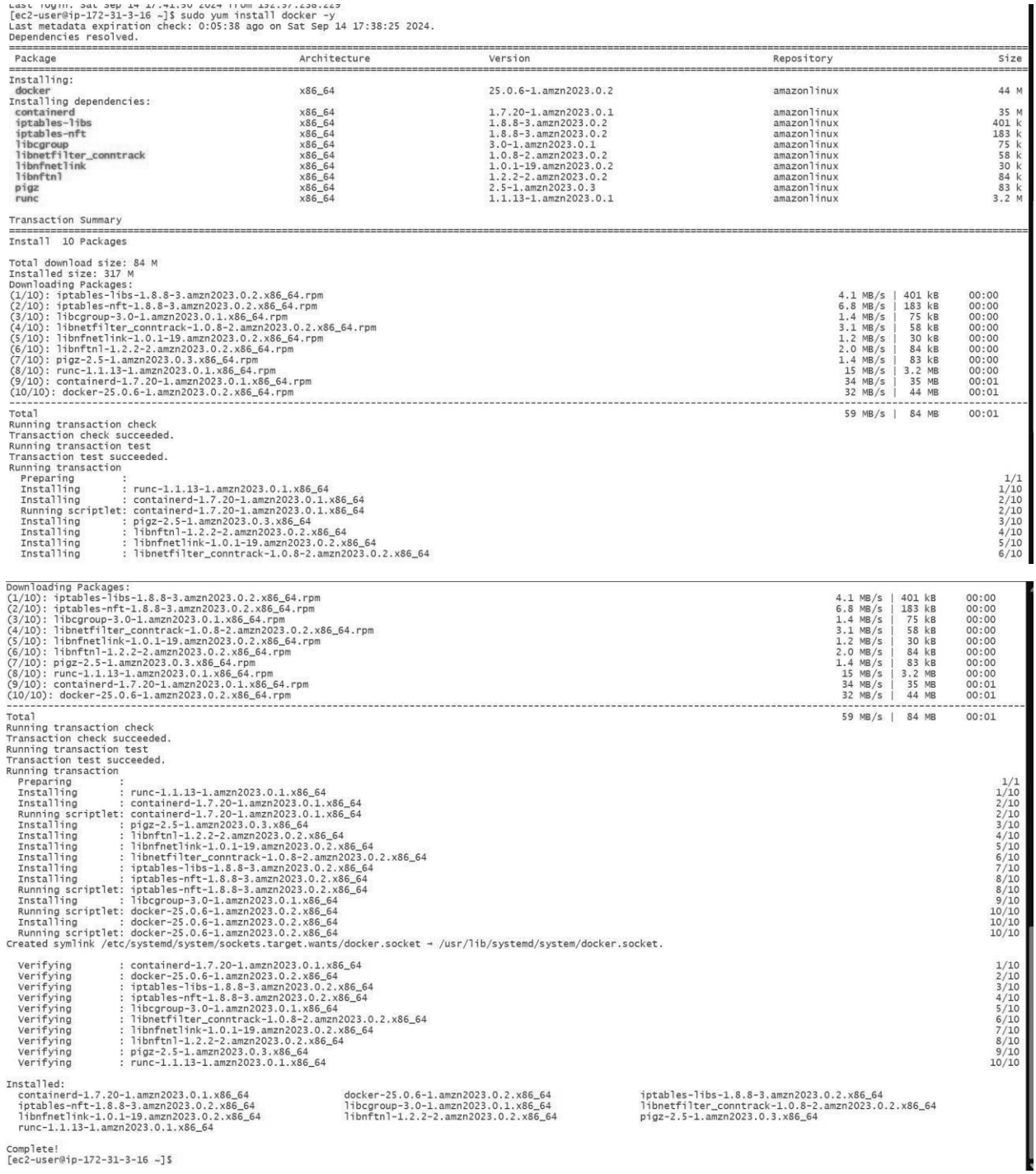
```
bhumi@LAPTOP-RVJC2CFS MINGW64 ~/Downloads
$ chmod 400 rook.pem

bhumi@LAPTOP-RVJC2CFS MINGW64 ~/Downloads
$
```

5. Now use this command on the terminal: ssh -i <keyname>.pem ubuntu@ and replace
   - Keyname with the name of your key pair, in our case test1.
   - As we are using amazon Linux instead of ubuntu we will have ec2-user • Replace public ip address with its value. Go to your instance and scroll down and you will find the public ip address there.

```
bhumi@LAPTOP-RVJC2CFS MINGW64 ~/Downloads
$ ssh -i "rook.pem" ec2-user@ec2-3-106-253-36.ap-southeast-2.compute.amazonaws.com
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sat Sep 14 17:41:50 2024 from 152.57.238.229
[ec2-user@ip-172-31-3-16 ~]$
```

6. Docker installation:

We will be installing docker by using “sudo yum install docker -y”



```
Last login: Sat Sep 14 17:41:36 2024 from 152.57.238.229
[ec2-user@ip-172-31-3-16 ~]$ sudo yum install docker -y
Last metadata expiration check: 0:05:38 ago on Sat Sep 14 17:38:25 2024.
Dependencies resolved.
================================================================================
 Package                 Architecture  Version                  Repository     Size
================================================================================
Installing:
 docker                  x86_64        25.0.6-1.amzn2023.0.2    amazonlinux    44 M
Installing dependencies:
 containerd              x86_64        1.7.20-1.amzn2023.0.1    amazonlinux    35 M
 iptables-libs           x86_64        1.8.8-3.amzn2023.0.2     amazonlinux   401 k
 iptables-nft            x86_64        1.8.8-3.amzn2023.0.2     amazonlinux   183 k
 libcgroup               x86_64        3.0-1.amzn2023.0.1       amazonlinux    75 k
 libnetfilter_conntrack  x86_64        1.0.8-2.amzn2023.0.2     amazonlinux    58 k
 libnfnetlink            x86_64        1.0.1-19.amzn2023.0.2    amazonlinux    30 k
 libnftnl                x86_64        1.2.2-2.amzn2023.0.2     amazonlinux    84 k
 pigz                    x86_64        2.5-1.amzn2023.0.3       amazonlinux    83 k
 runc                    x86_64        1.1.13-1.amzn2023.0.1    amazonlinux   3.2 M

Transaction Summary
================================================================================
Install  10 Packages

Total download size: 84 M
Installed size: 317 M
Downloading Packages:
(1/10): iptables-libs-1.8.8-3.amzn2023.0.2.x86_64.rpm      4.1 MB/s | 401 kB  00:00
(2/10): iptables-nft-1.8.8-3.amzn2023.0.2.x86_64.rpm       6.8 MB/s | 183 kB  00:00
(3/10): libcgroup-3.0-1.amzn2023.0.1.x86_64.rpm            1.4 MB/s |  75 kB  00:00
(4/10): libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64.rpm 3.1 MB/s |  58 kB  00:00
(5/10): libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64.rpm      1.2 MB/s |  30 kB  00:00
(6/10): libnftnl-1.2.2-2.amzn2023.0.2.x86_64.rpm           2.0 MB/s |  84 kB  00:00
(7/10): pigz-2.5-1.amzn2023.0.3.x86_64.rpm                 1.4 MB/s |  83 kB  00:00
(8/10): runc-1.1.13-1.amzn2023.0.1.x86_64.rpm               15 MB/s | 3.2 MB  00:00
(9/10): containerd-1.7.20-1.amzn2023.0.1.x86_64.rpm         34 MB/s |  35 MB  00:01
(10/10): docker-25.0.6-1.amzn2023.0.2.x86_64.rpm            32 MB/s |  44 MB  00:01
--------------------------------------------------------------------------------
Total                                                       59 MB/s |  84 MB  00:01
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : runc-1.1.13-1.amzn2023.0.1.x86_64                      1/10
  Installing       : containerd-1.7.20-1.amzn2023.0.1.x86_64                2/10
  Running scriptlet: containerd-1.7.20-1.amzn2023.0.1.x86_64                2/10
  Installing       : pigz-2.5-1.amzn2023.0.3.x86_64                         3/10
  Installing       : libnftnl-1.2.2-2.amzn2023.0.2.x86_64                   4/10
  Installing       : libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64              5/10
  Installing       : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64     6/10
```


```
Downloading Packages:
(1/10): iptables-libs-1.8.8-3.amzn2023.0.2.x86_64.rpm      4.1 MB/s | 401 kB  00:00
(2/10): iptables-nft-1.8.8-3.amzn2023.0.2.x86_64.rpm       6.8 MB/s | 183 kB  00:00
(3/10): libcgroup-3.0-1.amzn2023.0.1.x86_64.rpm            1.4 MB/s |  75 kB  00:00
(4/10): libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64.rpm 3.1 MB/s |  58 kB  00:00
(5/10): libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64.rpm      1.2 MB/s |  30 kB  00:00
(6/10): libnftnl-1.2.2-2.amzn2023.0.2.x86_64.rpm           2.0 MB/s |  84 kB  00:00
(7/10): pigz-2.5-1.amzn2023.0.3.x86_64.rpm                 1.4 MB/s |  83 kB  00:00
(8/10): runc-1.1.13-1.amzn2023.0.1.x86_64.rpm               15 MB/s | 3.2 MB  00:00
(9/10): containerd-1.7.20-1.amzn2023.0.1.x86_64.rpm         34 MB/s |  35 MB  00:01
(10/10): docker-25.0.6-1.amzn2023.0.2.x86_64.rpm            32 MB/s |  44 MB  00:01
--------------------------------------------------------------------------------
Total                                                       59 MB/s |  84 MB  00:01
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
```


```
  Preparing        :                                                        1/1
  Installing       : runc-1.1.13-1.amzn2023.0.1.x86_64                      1/10
  Installing       : containerd-1.7.20-1.amzn2023.0.1.x86_64                2/10
  Running scriptlet: containerd-1.7.20-1.amzn2023.0.1.x86_64                2/10
  Installing       : pigz-2.5-1.amzn2023.0.3.x86_64                         3/10
  Installing       : libnftnl-1.2.2-2.amzn2023.0.2.x86_64                   4/10
  Installing       : libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64              5/10
  Installing       : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64     6/10
  Installing       : iptables-libs-1.8.8-3.amzn2023.0.2.x86_64              7/10
  Installing       : iptables-nft-1.8.8-3.amzn2023.0.2.x86_64               8/10
  Running scriptlet: iptables-nft-1.8.8-3.amzn2023.0.2.x86_64               8/10
  Installing       : libcgroup-3.0-1.amzn2023.0.1.x86_64                    9/10
  Running scriptlet: docker-25.0.6-1.amzn2023.0.2.x86_64                    10/10
  Installing       : docker-25.0.6-1.amzn2023.0.2.x86_64                    10/10
  Running scriptlet: docker-25.0.6-1.amzn2023.0.2.x86_64                    10/10
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.

  Verifying        : containerd-1.7.20-1.amzn2023.0.1.x86_64                1/10
  Verifying        : docker-25.0.6-1.amzn2023.0.2.x86_64                    2/10
  Verifying        : iptables-libs-1.8.8-3.amzn2023.0.2.x86_64              3/10
  Verifying        : iptables-nft-1.8.8-3.amzn2023.0.2.x86_64               4/10
  Verifying        : libcgroup-3.0-1.amzn2023.0.1.x86_64                    5/10
  Verifying        : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64     6/10
  Verifying        : libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64              7/10
  Verifying        : libnftnl-1.2.2-2.amzn2023.0.2.x86_64                   8/10
  Verifying        : pigz-2.5-1.amzn2023.0.3.x86_64                         9/10
  Verifying        : runc-1.1.13-1.amzn2023.0.1.x86_64                     10/10

Installed:
  containerd-1.7.20-1.amzn2023.0.1.x86_64   docker-25.0.6-1.amzn2023.0.2.x86_64   iptables-libs-1.8.8-3.amzn2023.0.2.x86_64
  iptables-nft-1.8.8-3.amzn2023.0.2.x86_64  libcgroup-3.0-1.amzn2023.0.1.x86_64   libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64
  libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64 libnftnl-1.2.2-2.amzn2023.0.2.x86_64 pigz-2.5-1.amzn2023.0.3.x86_64
  runc-1.1.13-1.amzn2023.0.1.x86_64

Complete!
[ec2-user@ip-172-31-3-16 ~]$
```

7. Then to configure cgroup in a daemon json file we will run cd /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

```
[ec2-user@ip-172-31-3-16 ~]$ cd /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
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
[ec2-user@ip-172-31-3-16 docker]$
```

8. Kubernetes installation:

Search kubeadm installation on your browser and scroll down and select red hatbased distributions.

1. Set SELinux to `permissive` mode:

These instructions are for Kubernetes 1.31.

```
Linux in permissive mode (effectively disabling it)
enforce 0
 -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
```

```
# This overwrites any existing configuration in /etc/yum.repos.d/
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repom
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
```

3. Install kubelet, kubeadm and kubectl:

```
yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
```

4. (Optional) Enable the kubelet service before running kubeadm:

```
sudo systemctl enable --now kubelet
```

```
Error: This command has to be run with superuser privileges (under the root user on most systems).
[ec2-user@ip-172-31-3-16 docker]$ sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
Kubernetes                                                                                                  5.7 kB/s | 9.4 kB     00:01
Last metadata expiration check: 0:00:02 ago on Sat Sep 14 17:47:29 2024.
Dependencies resolved.
================================================================================================================================================
 Package                         Architecture            Version                                 Repository                                  Size
================================================================================================================================================
Installing:
 kubeadm                         x86_64                  1.31.1-150500.1.1                       kubernetes                                  11 M
 kubectl                         x86_64                  1.31.1-150500.1.1                       kubernetes                                  11 M
 kubelet                         x86_64                  1.31.1-150500.1.1                       kubernetes                                  15 M
Installing dependencies:
 conntrack-tools                 x86_64                  1.4.6-2.amzn2023.0.2                    amazonlinux                                208 k
 cri-tools                       x86_64                  1.31.1-150500.1.1                       kubernetes                                 6.9 M
 kubernetes-cni                  x86_64                  1.5.1-150500.1.1                        kubernetes                                 7.1 M
 libnetfilter_cthelper           x86_64                  1.0.0-21.amzn2023.0.2                   amazonlinux                                 24 k
 libnetfilter_cttimeout          x86_64                  1.0.0-19.amzn2023.0.2                   amazonlinux                                 24 k
 libnetfilter_queue              x86_64                  1.0.5-2.amzn2023.0.2                    amazonlinux                                 30 k

Transaction Summary
================================================================================================================================================
Install  9 Packages

Total download size: 51 M
Installed size: 269 M
Downloading Packages:
(1/9): libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64.rpm                                              499 kB/s |  24 kB     00:00
(2/9): libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64.rpm                                             376 kB/s |  24 kB     00:00
(3/9): libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64.rpm                                                  1.6 MB/s |  30 kB     00:00
(4/9): conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64.rpm                                                     1.7 MB/s | 208 kB     00:00
(5/9): cri-tools-1.31.1-150500.1.1.x86_64.rpm                                                               15 MB/s | 6.9 MB     00:00
(6/9): kubeadm-1.31.1-150500.1.1.x86_64.rpm                                                                 21 MB/s |  11 MB     00:00
(7/9): kubectl-1.31.1-150500.1.1.x86_64.rpm                                                                 17 MB/s |  11 MB     00:00
(8/9): kubernetes-cni-1.5.1-150500.1.1.x86_64.rpm                                                           21 MB/s | 7.1 MB     00:00
(9/9): kubelet-1.31.1-150500.1.1.x86_64.rpm                                                                 29 MB/s |  15 MB     00:00
------------------------------------------------------------------------------------------------------------------------------------------------
Total                                                                                                       45 MB/s |  51 MB     00:01
Kubernetes                                                                                                 8.0 kB/s | 1.7 kB     00:00
Importing GPG key 0x9A296436:
 Userid     : "isv:kubernetes OBS Project <isv:kubernetes@build.opensuse.org>"
 Fingerprint: DE15 B144 86CD 377B 9E87 6E1A 2346 54DA 9A29 6436
 From       : https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
Key imported successfully
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                                        1/1
  Installing       : kubernetes-cni-1.5.1-150500.1.1.x86_64                                                                                 1/9
  Installing       : cri-tools-1.31.1-150500.1.1.x86_64                                                                                     2/9
  Installing       : libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64                                                                         3/9
```

```
Running transaction
  Preparing        :                                                                                                                        1/1
  Installing       : kubernetes-cni-1.5.1-150500.1.1.x86_64                                                                                 1/9
  Installing       : cri-tools-1.31.1-150500.1.1.x86_64                                                                                     2/9
  Installing       : libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64                                                                         3/9
  Installing       : libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64                                                                    4/9
  Installing       : libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64                                                                     5/9
  Installing       : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64                                                                            6/9
  Running scriptlet: conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64                                                                            6/9
  Installing       : kubelet-1.31.1-150500.1.1.x86_64                                                                                       7/9
  Running scriptlet: kubelet-1.31.1-150500.1.1.x86_64                                                                                       7/9
  Installing       : kubeadm-1.31.1-150500.1.1.x86_64                                                                                       8/9
  Installing       : kubectl-1.31.1-150500.1.1.x86_64                                                                                       9/9
  Running scriptlet: kubectl-1.31.1-150500.1.1.x86_64                                                                                       9/9
  Verifying        : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64                                                                            1/9
  Verifying        : libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64                                                                     2/9
  Verifying        : libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64                                                                    3/9
  Verifying        : libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64                                                                         4/9
  Verifying        : cri-tools-1.31.1-150500.1.1.x86_64                                                                                     5/9
  Verifying        : kubeadm-1.31.1-150500.1.1.x86_64                                                                                       6/9
  Verifying        : kubectl-1.31.1-150500.1.1.x86_64                                                                                       7/9
  Verifying        : kubelet-1.31.1-150500.1.1.x86_64                                                                                       8/9
  Verifying        : kubernetes-cni-1.5.1-150500.1.1.x86_64                                                                                 9/9

Installed:
  conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64              cri-tools-1.31.1-150500.1.1.x86_64              kubeadm-1.31.1-150500.1.1.x86_64
  kubectl-1.31.1-150500.1.1.x86_64                         kubelet-1.31.1-150500.1.1.x86_64                kubernetes-cni-1.5.1-150500.1.1.x86_64
  libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64       libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64  libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64

Complete!
[ec2-user@ip-172-31-3-16 docker]$
```

9. After installing Kubernetes, we need to configure internet options to allow bridging.

sudo swapoff -a echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee a /etc/sysctl.conf sudo sysctl -p

```
[ec2-user@ip-172-31-3-16 docker]$ sudo swapoff -a
[ec2-user@ip-172-31-3-16 docker]$ echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf
net.bridge.bridge-nf-call-iptables=1
[ec2-user@ip-172-31-3-16 docker]$ sudo sysctl -p
net.bridge.bridge-nf-call-iptables = 1
net.bridge.bridge-nf-call-iptables = 1
[ec2-user@ip-172-31-3-16 docker]$
```

10. Initializing kubecluster:

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

```
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 172.31.3.16:6443 --token ekhyop.xkge2agz07jxxqqs \
        --discovery-token-ca-cert-hash sha256:8206263b4e2632eb03dafa4819c7c8505d47b21e8ba8c4901d5802c791c806f7
[ec2-user@ip-172-31-3-16 docker]$
```

11. The mkdir command that is generated after initialization has to be copy pasted in the terminal.

```
[ec2-user@ip-172-31-3-16 docker]$ mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config
[ec2-user@ip-172-31-3-16 docker]$
```

12. Then, add a common networking plugin called flannel: kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kubeflannel.yml

```
[ec2-user@ip-172-31-3-16 docker]$ 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
[ec2-user@ip-172-31-3-16 docker]$
```

13. 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

```
[ec2-user@ip-172-31-3-16 docker]$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
[ec2-user@ip-172-31-3-16 docker]$
```

https://k8s.io/examples/application/deployment.yaml

14. Use kubectl get pods to check if the pod is working correctly.

```
[ec2-user@ip-172-31-3-16 docker]$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-mvnj7    0/1     Pending   0          18s
nginx-deployment-d556bf558-w2pd8    0/1     Pending   0          18s
[ec2-user@ip-172-31-3-16 docker]$
```

15. To change status from pending to running use the following command: kubectl describe pod nginx.

```
nginx-deployment-d556bf558-w2pd8   0/1     Pending   0          18s
[ec2-user@ip-172-31-3-16 docker]$ kubectl describe pod nginx
Name:             nginx-deployment-d556bf558-mvnj7
Namespace:        default
Priority:         0
Service Account:  default
Node:             <none>
Labels:           app=nginx
                  pod-template-hash=d556bf558
Annotations:      <none>
Status:           Pending
IP:
IPs:              <none>
Controlled By:    ReplicaSet/nginx-deployment-d556bf558
Containers:
  nginx:
    Image:        nginx:1.14.2
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-8cms7 (ro)
Conditions:
  Type           Status
  PodScheduled   False
Volumes:
  kube-api-access-8cms7:
    Type:                     Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:   3607
    ConfigMapName:            kube-root-ca.crt
    ConfigMapOptional:        <nil>
    DownwardAPI:              true
QoS Class:                    BestEffort
Node-Selectors:               <none>
Tolerations:                  node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                              node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type     Reason            Age   From               Message
  ----     ------            ----  ----               -------
  Warning  FailedScheduling  57s   default-scheduler  0/1 nodes are available: 1 node(s) had untolerated taint {node-role.kubernetes.io/control-plane: }. preemption: 0/1 no
des are available: 1 Preemption is not helpful for scheduling.

Name:             nginx-deployment-d556bf558-w2pd8
Namespace:        default
Priority:         0
Service Account:  default
Node:             <none>
Labels:           app=nginx
                  pod-template-hash=d556bf558
Annotations:      <none>
Status:           Pending
IP:
IPs:              <none>
Controlled By:    ReplicaSet/nginx-deployment-d556bf558
```

```
Priority:         0
Service Account:  default
Node:             <none>
Labels:           app=nginx
                  pod-template-hash=d556bf558
Annotations:      <none>
Status:           Pending
IP:
IPs:              <none>
Controlled By:    ReplicaSet/nginx-deployment-d556bf558
Containers:
  nginx:
    Image:        nginx:1.14.2
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-6fl8b (ro)
Conditions:
  Type           Status
  PodScheduled   False
Volumes:
  kube-api-access-6fl8b:
    Type:                     Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:   3607
    ConfigMapName:            kube-root-ca.crt
    ConfigMapOptional:        <nil>
    DownwardAPI:              true
QoS Class:                    BestEffort
Node-Selectors:               <none>
Tolerations:                  node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                              node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type     Reason            Age   From               Message
  ----     ------            ----  ----               -------
  Warning  FailedScheduling  57s   default-scheduler  0/1 nodes are available: 1 node(s) had untolerated taint {node-role.kubernetes.io/control-plane: }. preemption: 0/1 no
des are available: 1 Preemption is not helpful for scheduling.
```

Use the below command to remove taints.

```
[ec2-user@ip-172-31-3-16 docker]$ kubectl taint nodes --all node-role.kubernetes
.io/control-plane-
node/ip-172-31-3-16.ap-southeast-2.compute.internal untainted
```

16. Check the pod status.

```
NAME    READY   STATUS    RESTARTS      AGE
nginx   1/1     Running   1 (6s ago)    90s
```

17. port forward the deployment to your localhost so that you can view it

```
Forwarding from 127.0.0.1:8081 -> 80
Forwarding from [::1]:8081 -> 80
```

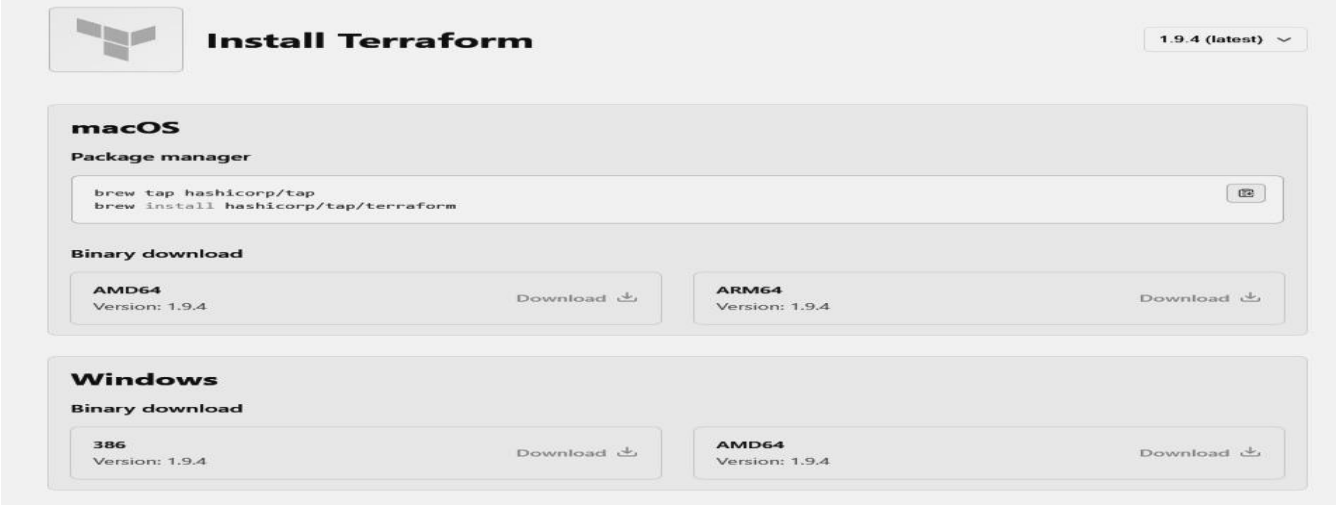
18. Verify your deployment Open up a new terminal and ssh to your EC2 instance. Then, use this curl command to check if the Nginx server is running. curl --head http://127.0.0.1:8080

Conclusion: In this experiment, we launched an EC2 instance and configured SSH access by updating the inbound rules. Next, we installed Docker and Kubernetes, and adjusted network settings to enable bridging. After completing the setup, we installed the Flannel networking plugin to ensure proper communication within the cluster. Once the cluster was up and running, we successfully deployed an NGINX server and verified its deployment.

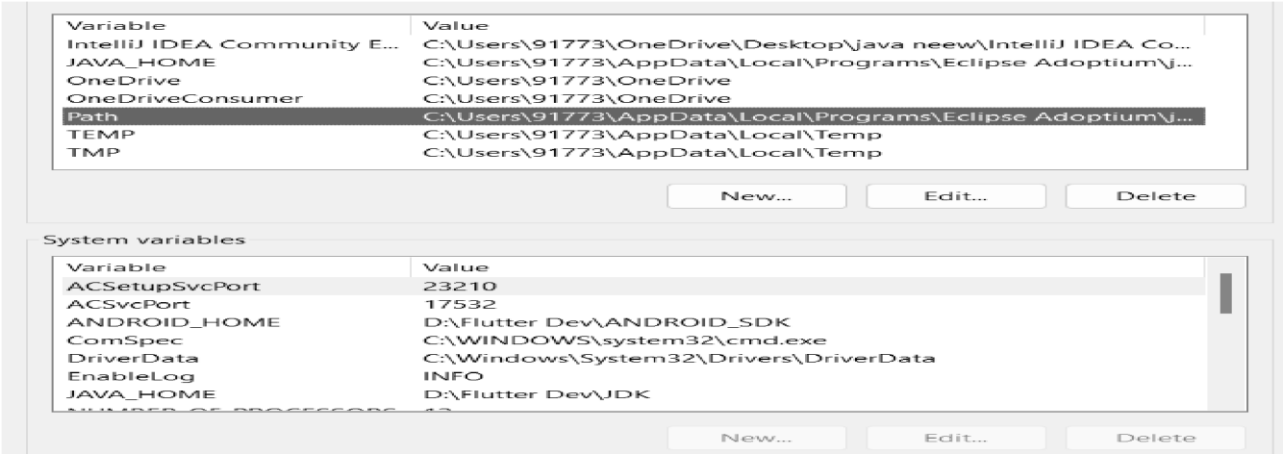
Roshan Bhagtani
D15C
4

EXPERIMENT 5

Step 1) Install Terraform

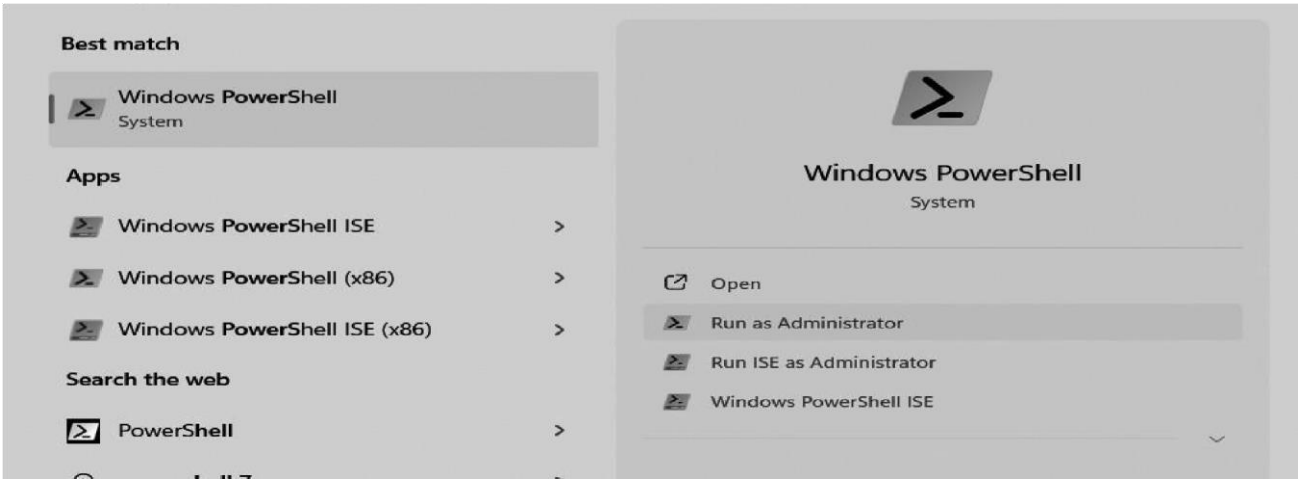


Step 2) Setup path in environment variables.



Step 3) Select the C drive Terraform folder as variable value.

Step 4) Open Windows powershell as Administrator.



Step 5) Run the Terraform command in powershell.

Roshan Bhagtani
D15C
4

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

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

PS C:\WINDOWS\system32> terraform
Usage: terraform [global options] <subcommand> [args]

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

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

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

Global options (use these before the subcommand, if any):
  -chdir=DIR    Switch to a different working directory before executing the
                given subcommand.
  -help         Show this help output, or the help for a specified subcommand.
  -version      An alias for the "version" subcommand.
PS C:\WINDOWS\system32>
```

Experiment No:6

Implementation:

A. Creating docker image using terraform

Prerequisites:

1. Download and install Docker Desktop from
Website: https://www.docker.com.

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

```
C:\Users\excel>docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

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

Step 1:To Verify Docker Functionality

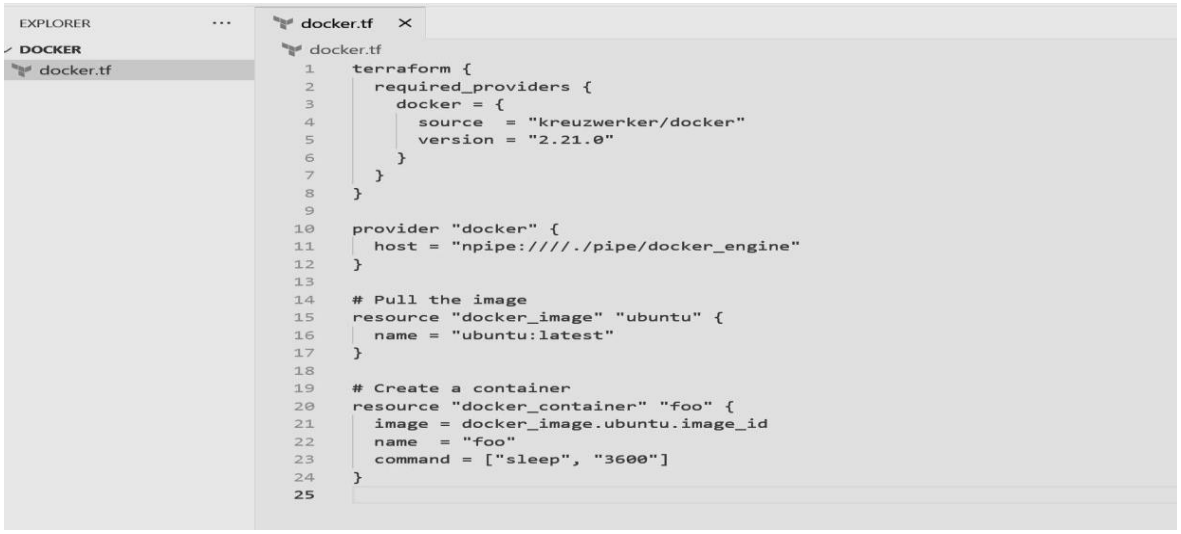
1. Create a folder named `Terraform Scripts` to store various scripts for this experiment.

Step 2:To Set Up Terraform Configuration

1. Inside the `Terraform Scripts` folder, create a new folder named `Docker`.
2. Within the `Docker` folder, create a file named `docker.tf` using Atom editor and insert the following content to configure an Ubuntu Linux container:

terraform {

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

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

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

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



Step 3:To Initialize Terraform

Run the command `terraform init` to initialize the Terraform configuration.

```
C:\Users\excel\Documents\college\Terraform scripts\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:To Review Terraform Plan

Execute `terraform plan` to preview the resources that will be created.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>terraform plan

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

Terraform will perform the following actions:

  # docker_container.foo will be created
  + resource "docker_container" "foo" {
      + attach           = false
      + bridge           = (known after apply)
      + command          = [
          + "sleep",
          + "3600",
        ]
      + container_logs   = (known after apply)
      + entrypoint       = (known after apply)
      + env              = (known after apply)
      + exit_code        = (known after apply)
      + gateway          = (known after apply)
      + hostname         = (known after apply)
      + id               = (known after apply)
      + image            = (known after apply)
      + init             = (known after apply)
      + ip_address       = (known after apply)
      + ip_prefix_length = (known after apply)
      + ipc_mode         = (known after apply)
      + log_driver       = (known after apply)
      + logs             = false
      + must_run         = true
      + name             = "foo"
      + network_data     = (known after apply)
      + read_only        = false
      + remove_volumes   = true
      + restart          = "no"
      + rm               = false
      + runtime          = (known after apply)
      + security_opts    = (known after apply)
      + shm_size         = (known after apply)
      + start            = true
```

```
      + 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:To Apply Terraform Configuration

Run `terraform apply` to apply the configuration and create the Ubuntu Linux container.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>terraform apply

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

Terraform will perform the following actions:

  # docker_container.foo will be created
  + resource "docker_container" "foo" {
      + attach           = false
      + bridge           = (known after apply)
      + command          = [
          + "sleep",
          + "3600",
        ]
      + container_logs   = (known after apply)
      + entrypoint       = (known after apply)
      + env              = (known after apply)
      + exit_code        = (known after apply)
      + gateway          = (known after apply)
      + hostname         = (known after apply)
      + id               = (known after apply)
      + image            = (known after apply)
      + init             = (known after apply)
      + ip_address       = (known after apply)
      + ip_prefix_length = (known after apply)
      + ipc_mode         = (known after apply)
      + log_driver       = (known after apply)
      + logs             = false
      + must_run         = true
      + name             = "foo"
      + network_data     = (known after apply)
      + read_only        = false
      + remove_volumes   = true
      + restart          = "no"
      + rm               = false
      + runtime          = (known after apply)
      + security_opts    = (known after apply)
      + shm_size         = (known after apply)
      + start            = true
```

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

      + healthcheck (known after apply)

      + labels (known after apply)
    }

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

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

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

  Enter a value: yes

docker_image.ubuntu: Creating...
docker_image.ubuntu: Still creating... [10s elapsed]
docker_image.ubuntu: Still creating... [20s elapsed]
docker_image.ubuntu: Still creating... [30s elapsed]
docker_image.ubuntu: Still creating... [41s elapsed]
docker_image.ubuntu: Creation complete after 43s [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Creating...
docker_container.foo: Creation complete after 1s [id=71bffb28b5cee3d1699c27dbcceb992b931000a847e6dfb219b3ca85ce5c6131]

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

Before executing `terraform apply`, list the Docker images.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>docker images
REPOSITORY   TAG        IMAGE ID   CREATED   SIZE
```

After executing `terraform apply`, list the Docker images again.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    edbfe74c41f8   3 weeks ago   78.1MB
```

Step 6: Clean Up

To delete the created Ubuntu container, run `terraform destroy`.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=71bffb28b5cee3d1699c27dbcceb992b931000a847e6dfb219b3ca85ce5c6131]

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

Terraform will perform the following actions:

  # docker_container.foo will be destroyed
  - resource "docker_container" "foo" {
      - attach           = false -> null
      - command          = [
          - "sleep",
          - "3600",
        ] -> null
      - cpu_shares       = 0 -> null
      - dns              = [] -> null
      - dns_opts         = [] -> null
      - dns_search       = [] -> null
      - entrypoint       = [] -> null
      - env              = [] -> null
      - gateway          = "172.17.0.1" -> null
      - group_add        = [] -> null
      - hostname         = "71bffb28b5ce" -> null
      - id               = "71bffb28b5cee3d1699c27dbcceb992b931000a847e6dfb219b3ca85ce5c6131" -> null
      - image            = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
      - init             = false -> null
      - ip_address       = "172.17.0.2" -> null
      - ip_prefix_length = 16 -> null
      - ipc_mode         = "private" -> null
      - links            = [] -> null
      - log_driver       = "json-file" -> null
      - log_opts         = {} -> null
      - logs             = false -> null
      - max_retry_count  = 0 -> null
      - memory           = 0 -> null
      - memory_swap      = 0 -> null
      - must_run         = true -> null
      - name             = "foo" -> null
```

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

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

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

```
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=71bffb28b5cee3d1699c27dbcceb992b931000a847e6dfb219b3ca85ce5c6131]
docker_container.foo: Destruction complete after 0s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:la
docker_image.ubuntu: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
```

After executing `terraform destroy`, list the Docker images one more time.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```

Step 7:To check correctness of configured files.

Execute `terraform validate` to check the correctness of your Terraform configuration files.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>terraform validate
Success! The configuration is valid.
```

Step 8:To verify the details.

Run `terraform providers` to list the providers used in your configuration and verify their details.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>terraform providers

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

Step 9:To generate visual representation.

Generate a visual representation of the dependency graph of your Terraform resources.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>terraform graph
digraph G {
  rankdir = "RL";
  node [shape = rect, fontname = "sans-serif"];
  "docker_container.foo" [label="docker_container.foo"];
  "docker_image.ubuntu" [label="docker_image.ubuntu"];
  "docker_container.foo" -> "docker_image.ubuntu";
}
```

Name :Roshan Bhagtani **Roll No : 04** **Class : D15C**

Experiment 7

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

Theory:

What is SAST?

Static application security testing (SAST), or static analysis, is a testing methodology that analyzes source code to find security vulnerabilities that make your organization's applications susceptible to attack. SAST scans an application before the code is compiled. It's also known as white box testing.

What problems does SAST solve?

SAST takes place very early in the software development life cycle (SDLC) as it does not require a working application and can take place without code being executed. It helps developers identify vulnerabilities in the initial stages of development and quickly resolve issues without breaking builds or passing on vulnerabilities to the final release of the application.

SAST tools give developers real-time feedback as they code, helping them fix issues before they pass the code to the next phase of the SDLC. This prevents security-related issues from being considered an afterthought. SAST tools also provide graphical representations of the issues found, from source to sink. These help you navigate the code easier. Some tools point out the exact location of vulnerabilities and highlight the risky code. Tools can also provide in-depth guidance on how to fix issues and the best place in the code to fix them, without requiring deep security domain expertise.

It's important to note that SAST tools must be run on the application on a regular basis, such as during daily/monthly builds, every time code is checked in, or during a code release.

Why is SAST important?

Developers dramatically outnumber security staff. It can be challenging for an organization to find the resources to perform code reviews on even a fraction of its applications. A key strength of SAST tools is the ability to analyze 100% of the codebase. Additionally, they are much faster than manual secure code reviews performed by humans. These tools can scan millions of lines of code in a matter of minutes. SAST tools automatically identify critical vulnerabilities—such as buffer overflows, SQL injection, cross-site scripting, and others—with high confidence. Thus, integrating

static analysis into the SDLC can yield dramatic results in the overall quality of the code developed.

What are the key steps to run SAST effectively?

There are six simple steps needed to perform SAST efficiently in organizations that have a very large number of applications built with different languages, frameworks, and platforms.

1. **Finalize the tool.** Select a static analysis tool that can perform code reviews of applications written in the programming languages you use. The tool should also be able to comprehend the underlying framework used by your software.

2. **Create the scanning infrastructure, and deploy the tool.** This step involves handling the licensing requirements, setting up access control and authorization, and procuring the resources required (e.g., servers and databases) to deploy the tool.

3. **Customize the tool.** Fine-tune the tool to suit the needs of the organization. For example, you might configure it to reduce false positives or find additional security vulnerabilities by writing new rules or updating existing ones. Integrate the tool into the build environment, create dashboards for tracking scan results, and build custom reports.

4. **Prioritize and onboard applications.** Once the tool is ready, onboard your applications. If you have a large number of applications, prioritize the high-risk applications to scan first. Eventually, all your applications should be onboarded and scanned regularly, with application scans synced with release cycles, daily or monthly builds, or code check-ins.

5. **Analyze scan results.** This step involves triaging the results of the scan to remove false positives. Once the set of issues is finalized, they should be tracked and provided to the deployment teams for proper and timely remediation.

6. **Provide governance and training.** Proper governance ensures that your development teams are employing the scanning tools properly. The software security touchpoints should be present within the SDLC. SAST should be incorporated as part of your application development and deployment process.

Integrating Jenkins with SonarQube:

Prerequisites:

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

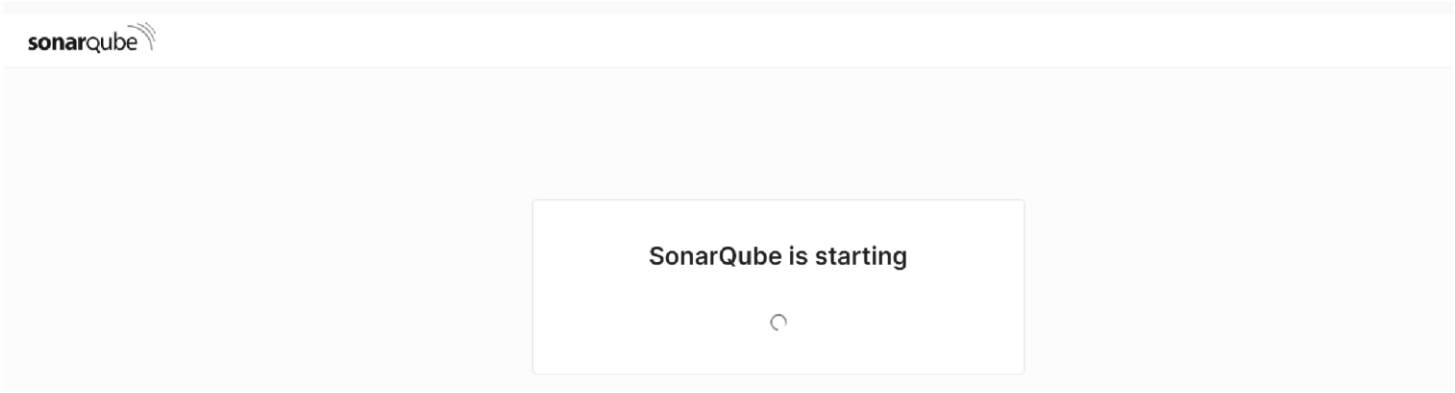
Steps to integrate Jenkins with SonarQube

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

2. Run SonarQube in a Docker container using this command -

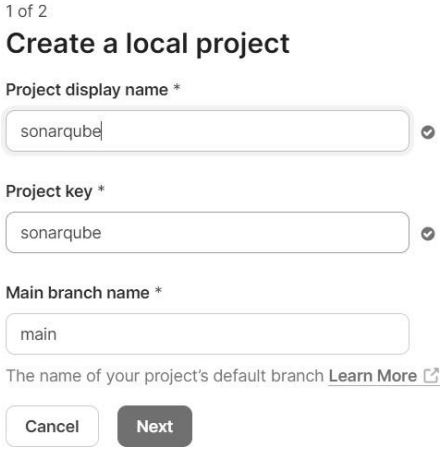
```
C:\Users\ADMIN>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
de76efbeef2054aeb442b86ba54c2916039b8757b388482d9780ffc69f5d8bbe
```

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



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

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

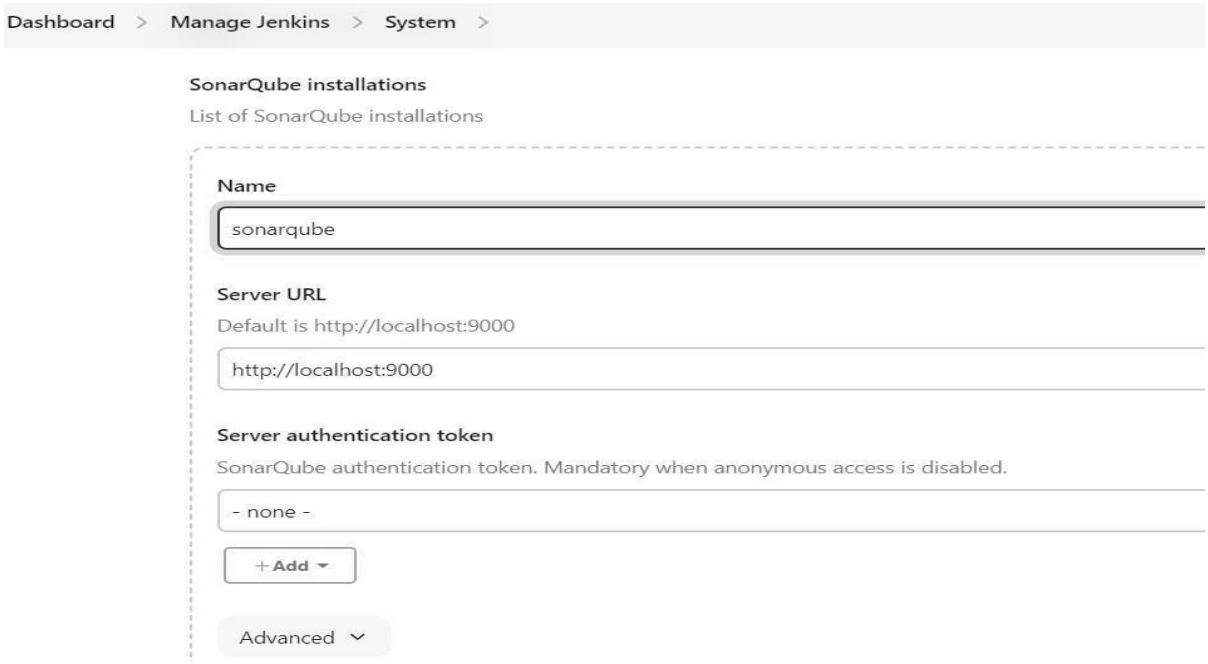


   Setup the project and come back to Jenkins Dashboard.

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

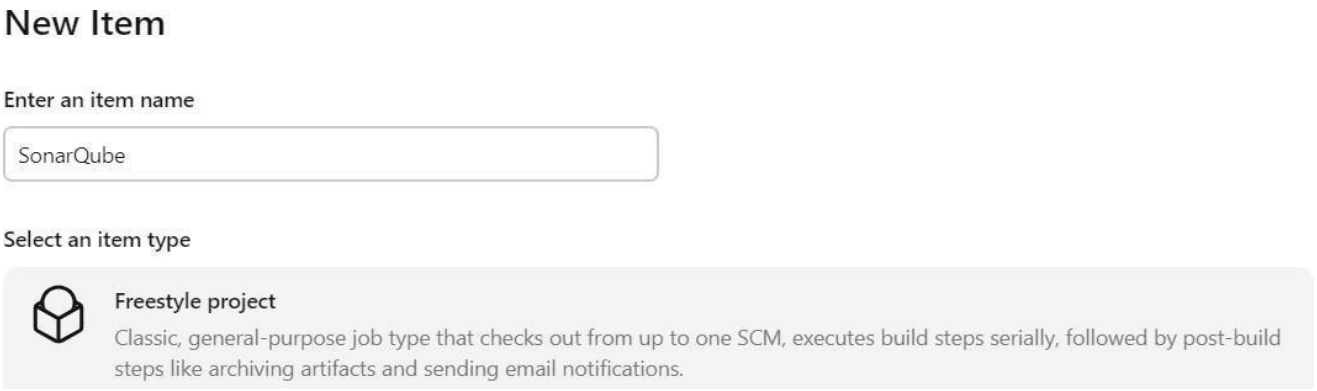
7. Under Jenkins ‘Configure System’, look for SonarQube Servers and enter the details.

   Enter the Server Authentication token if needed.



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

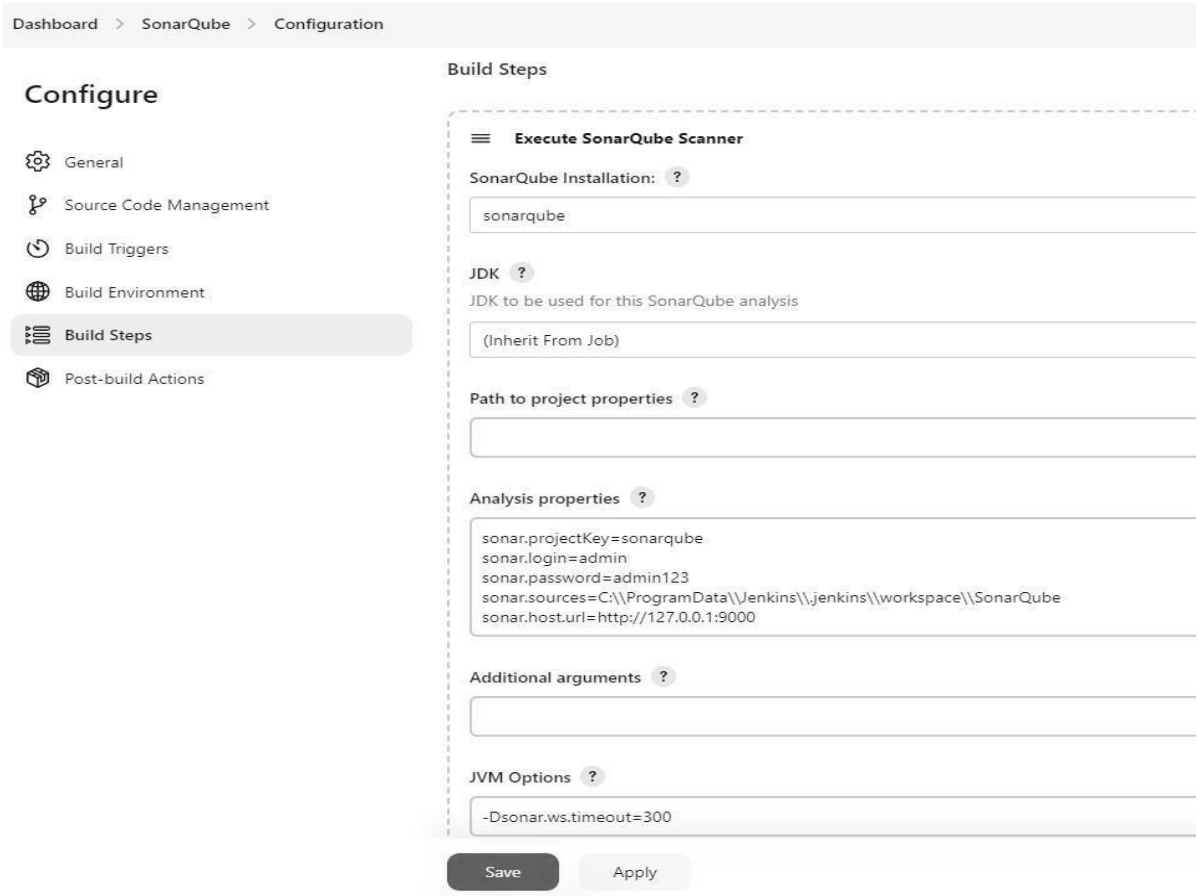
9. After the configuration, create a New Item in Jenkins, choose a freestyle project.



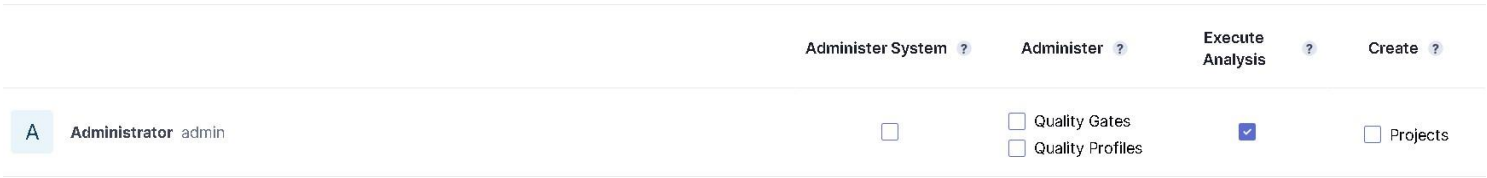
10. Choose this GitHub repository in Source Code Management.
    https://github.com/shazforiot/MSBuild_firstproject.git

    It is a sample hello-world project with no vulnerabilities and issues, just to test the integration.

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



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



13. Run The Build.

Check the console output.

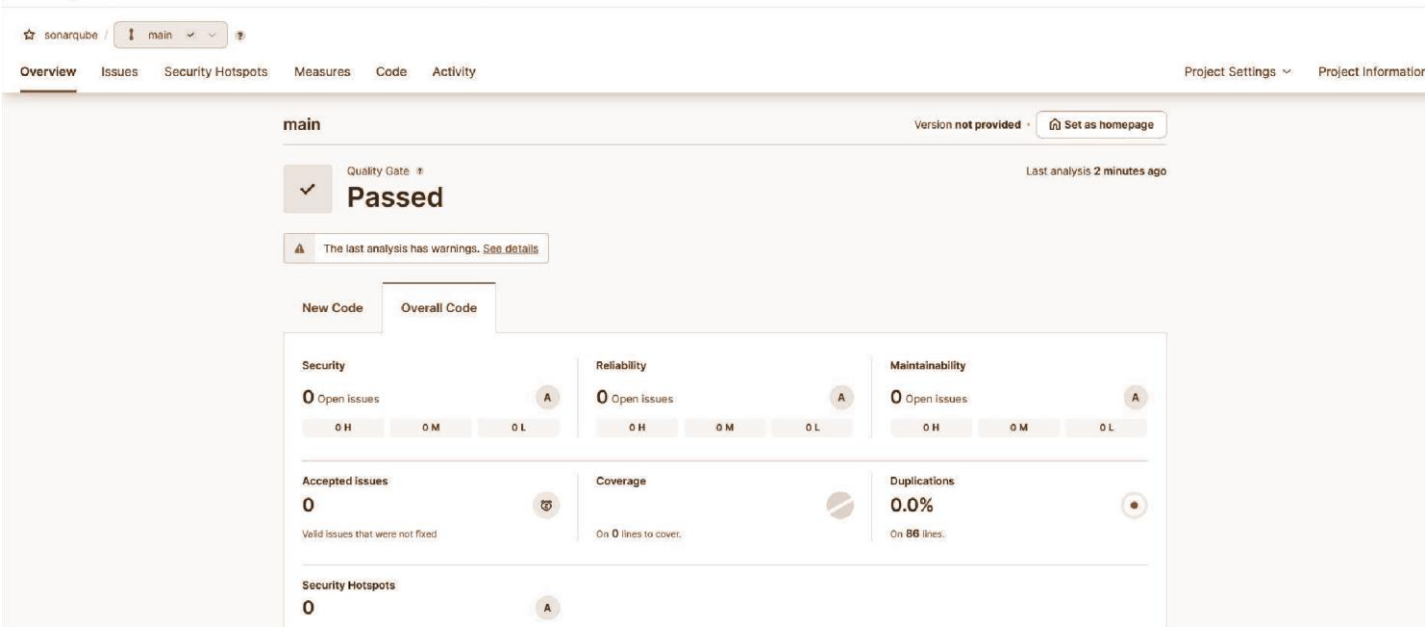
Console Output

Download Copy View as plain tex

```
Started by user Shiven Bansal
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\.jenkins\workspace\SonarQube
The recommended git tool is: NONE
No credentials specified
 > git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\SonarQube\.git # timeout=10
Fetching changes from the remote Git repository
 > git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject.git # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject.git
 > git.exe --version # timeout=10
 > git --version # 'git version 2.45.2.windows.1'
 > git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)
 > git.exe config core.sparsecheckout # timeout=10
 > git.exe checkout -f f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
Commit message: "updated"
 > git.exe rev-list --no-walk f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
[SonarQube] $ C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube\bin\sonar-scanner.bat -Dsonar.host.url=http://localhost:9000 -Dsonar.projectKey=sonarqube -
Dsonar.login=admin -Dsonar.host.url=http://127.0.0.1:9000 -Dsonar.sources=C:\ProgramData\Jenkins\.jenkins\workspace\SonarQube -Dsonar.password=admin123 -
Dsonar.projectBaseDir=C:\ProgramData\Jenkins\.jenkins\workspace\SonarQube
16:16:39.198 WARN  Property 'sonar.host.url' with value 'http://localhost:9000' is overridden with value 'http://127.0.0.1:9000'
16:16:39.206 INFO  Scanner configuration file: C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube\bin\..\conf\sonar-scanner.properties
16:16:39.206 INFO  Project root configuration file: NONE
16:16:39.230 INFO  SonarScanner CLI 6.1.0.4477
16:16:39.230 INFO  Java 21.0.4 Eclipse Adoptium (64-bit)
16:16:39.230 INFO  Windows 11 10.0 amd64
16:16:39.230 INFO  SONAR_SCANNER_OPTS=-Dsonar.ws.timeout=300
16:16:39.254 INFO  User cache: C:\Windows\system32\config\systemprofile\.sonar\cache
```

```
16:16:58.734 INFO  Using git CLI to retrieve untracked files
16:16:58.791 INFO  Analyzing language associated files and files included via "sonar.text.inclusions" that are tracked by git
16:16:58.856 INFO  14 source files to be analyzed
16:16:59.154 INFO  14/14 source files have been analyzed
16:16:59.154 INFO  Sensor TextAndSecretsSensor [text] (done) | time=1306ms
16:16:59.163 INFO  ------------ Run sensors on project
16:16:59.373 INFO  Sensor C# [csharp]
16:16:59.373 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
16:16:59.373 INFO  Sensor C# [csharp] (done) | time=0ms
16:16:59.373 INFO  Sensor Analysis Warnings import [csharp]
16:16:59.379 INFO  Sensor Analysis Warnings import [csharp] (done) | time=0ms
16:16:59.379 INFO  Sensor C# File Caching Sensor [csharp]
16:16:59.379 WARN  Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir' property.
16:16:59.379 INFO  Sensor C# File Caching Sensor [csharp] (done) | time=6ms
16:16:59.379 INFO  Sensor Zero Coverage Sensor
16:16:59.389 INFO  Sensor Zero Coverage Sensor (done) | time=10ms
16:16:59.389 INFO  SCM Publisher SCM provider for this project is: git
16:16:59.389 INFO  SCM Publisher 4 source files to be analyzed
16:16:59.838 INFO  SCM Publisher 4/4 source files have been analyzed (done) | time=449ms
16:16:59.846 INFO  CPD Executor Calculating CPD for 0 files
16:16:59.846 INFO  CPD Executor CPD calculation finished (done) | time=0ms
16:16:59.854 INFO  SCM revision ID 'f2bc042c04c6e72427c380bcaee6d6fee7b49adf'
16:17:00.121 INFO  Analysis report generated in 120ms, dir size=201.1 kB
16:17:00.195 INFO  Analysis report compressed in 57ms, zip size=22.4 kB
16:17:00.393 INFO  Analysis report uploaded in 195ms
16:17:00.394 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://127.0.0.1:9000/dashboard?id=sonarqube
16:17:00.395 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
16:17:00.395 INFO  More about the report processing at http://127.0.0.1:9000/api/ce/task?id=acd819f5-9e70-42ab-bff7-3cc893e2cae4
16:17:00.405 INFO  Analysis total time: 18.743 s
16:17:00.408 INFO  SonarScanner Engine completed successfully
16:17:00.494 INFO  EXECUTION SUCCESS
16:17:00.494 INFO  Total time: 21.288s
Finished: SUCCESS
```

14. Once the build is complete, check the project in SonarQube.



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

Conclusion:

In this experiment, I learned how to integrate Jenkins with SonarQube for performing Static Application Security Testing (SAST). I set up SonarQube in a Docker container and configured Jenkins to use the SonarQube scanner. By creating a manual project in SonarQube and configuring the necessary authentication and tools in Jenkins, I established a seamless connection between Jenkins and SonarQube for static code analysis.

I tested the integration with a sample GitHub repository, successfully running a build and analyzing the project's code quality through SonarQube. This hands-on experience enhanced my understanding of the SAST process, Jenkins automation, and SonarQube's capabilities for identifying potential code vulnerabilities.

Name :Roshan B **Roll No : 04** **Class : D15C**

Experiment - 8

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

Theory:

What is SAST?

Static application security testing (SAST), or static analysis, is a testing methodology that analyzes source code to find security vulnerabilities that make your organization's applications susceptible to attack. SAST scans an application before the code is compiled. It's also known as white box testing.

What problems does SAST solve?

SAST takes place very early in the software development life cycle (SDLC) as it does not require a working application and can take place without code being executed. It helps developers identify vulnerabilities in the initial stages of development and quickly resolve issues without breaking builds or passing on vulnerabilities to the final release of the application.

SAST tools give developers real-time feedback as they code, helping them fix issues before they pass the code to the next phase of the SDLC. This prevents security-related issues from being considered an afterthought. SAST tools also provide graphical representations of the issues found, from source to sink. These help you navigate the code easier. Some tools point out the exact location of vulnerabilities and highlight the risky code. Tools can also provide in-depth guidance on how to fix issues and the best place in the code to fix them, without requiring deep security domain expertise.

It's important to note that SAST tools must be run on the application on a regular basis, such as during daily/monthly builds, every time code is checked in, or during a code release.

Why is SAST important?

Developers dramatically outnumber security staff. It can be challenging for an organization to find the resources to perform code reviews on even a fraction of its applications. A key strength of SAST tools is the ability to analyze 100% of the codebase. Additionally, they are much faster than manual secure code reviews performed by humans. These tools can scan millions of lines of code

in a matter of minutes. SAST tools automatically identify critical vulnerabilities—such as buffer overflows, SQL injection, cross-site scripting, and others—with high confidence.

What is a CI/CD Pipeline?

CI/CD pipeline refers to the Continuous Integration/Continuous Delivery pipeline. Before we dive deep into this segment, let's first understand what is meant by the term ‘pipeline’?

A pipeline is a concept that introduces a series of events or tasks that are connected in a sequence to make quick software releases. For example, there is a task, that task has got five different stages, and each stage has got some steps. All the steps in phase one have to be completed, to mark the latter stage to be complete.

Now, consider the CI/CD pipeline as the backbone of the DevOps approach. This Pipeline is responsible for building codes, running tests, and deploying new software versions. The Pipeline executes the job in a defined manner by first coding it and then structuring it inside several blocks that may include several steps or tasks.

What is SonarQube?

SonarQube is an open-source platform developed by SonarSource for continuous inspection of code quality. Sonar does static code analysis, which provides a detailed report of bugs, code smells, vulnerabilities, code duplications.
It supports 25+ major programming languages through built-in rulesets and can also be extended with various plugins.

Benefits of SonarQube

- **Sustainability -** Reduces complexity, possible vulnerabilities, and code duplications, optimizing the life of applications.
- **Increase productivity -** Reduces the scale, cost of maintenance, and risk of the application; as such, it removes the need to spend more time changing the code
- **Quality code -** Code quality control is an inseparable part of the process of software development.
- **Detect Errors -** Detects errors in the code and alerts developers to fix them automatically before submitting them for output.
- **Increase consistency -** Determines where the code criteria are breached and enhances the quality
- **Business scaling -** No restriction on the number of projects to be evaluated
- **Enhance developer skills -** Regular feedback on quality problems helps developers to improve their coding skills

Integrating Jenkins with SonarQube:

Prerequisites:

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

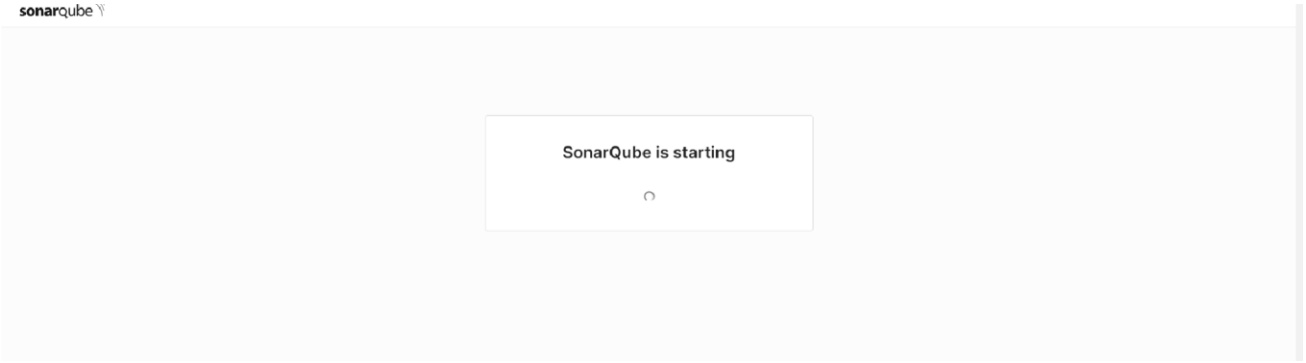
Steps to create a Jenkins CI/CD Pipeline and use SonarQube to perform SAST

1. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.
2. Run SonarQube in a Docker container using this command -

```
C:\Users\ADMIN>docker run -d --name sonarqube2 -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9001:9000 sonarqube:latest
fda86b00e3989f3eb5aca8396b29b2a0adc95bcfe0fc5d85cf1237491e7678b9
```

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

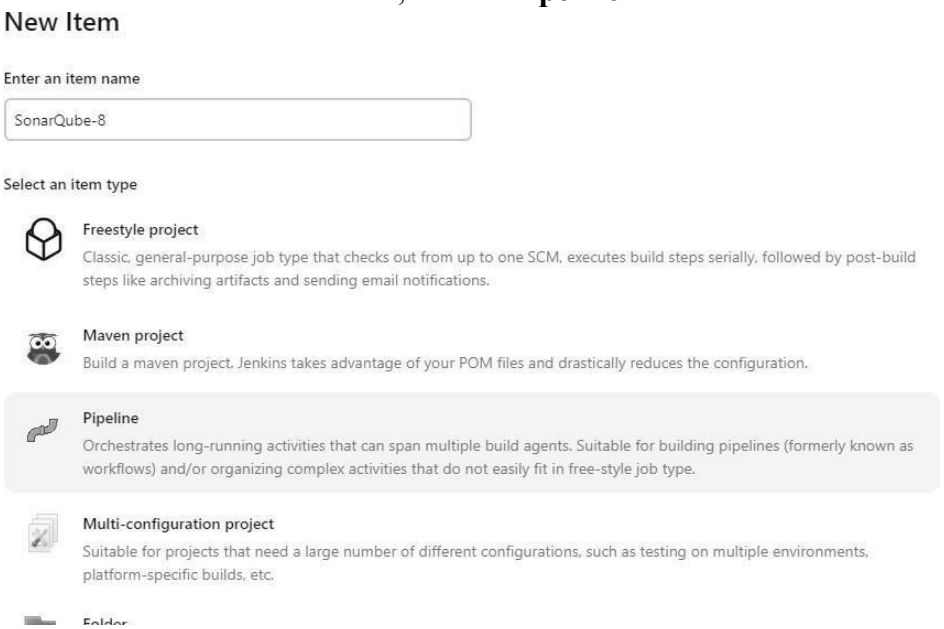
4. Login to SonarQube using username *admin* and password *admin*.



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

Setup the project and come back to Jenkins Dashboard.

6. Create a New Item in Jenkins, choose **Pipeline**.



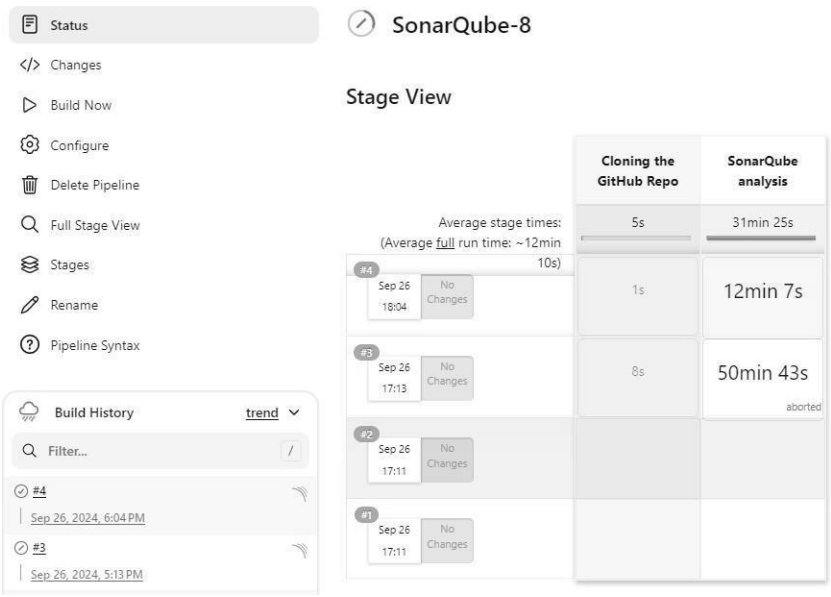
7. Under Pipeline Script, enter the following -

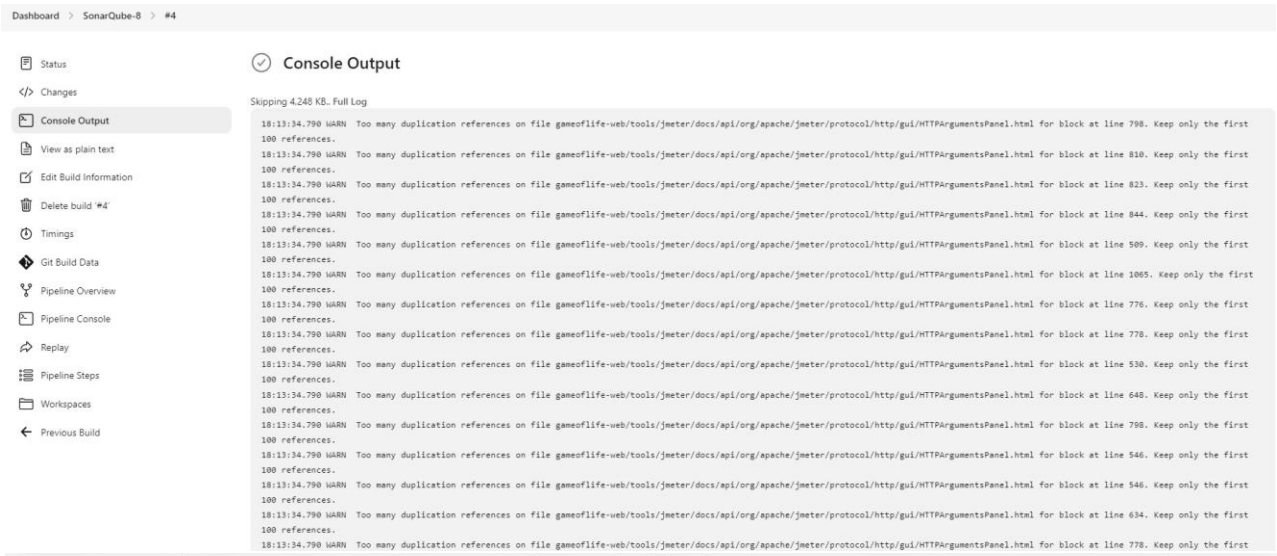
```
      node {
        stage('Cloning the GitHub Repo') {
          git 'https://github.com/shazforiot/GOL.git'
        } stage('SonarQube
        analysis') {
          withSonarQubeEnv('sonarqube') {
            sh "<PATH_TO_SONARQUBE_FOLDER>//bin//sonar-scanner \
            -D sonar.login=<SonarQube_USERNAME>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

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

8. Run The Build.

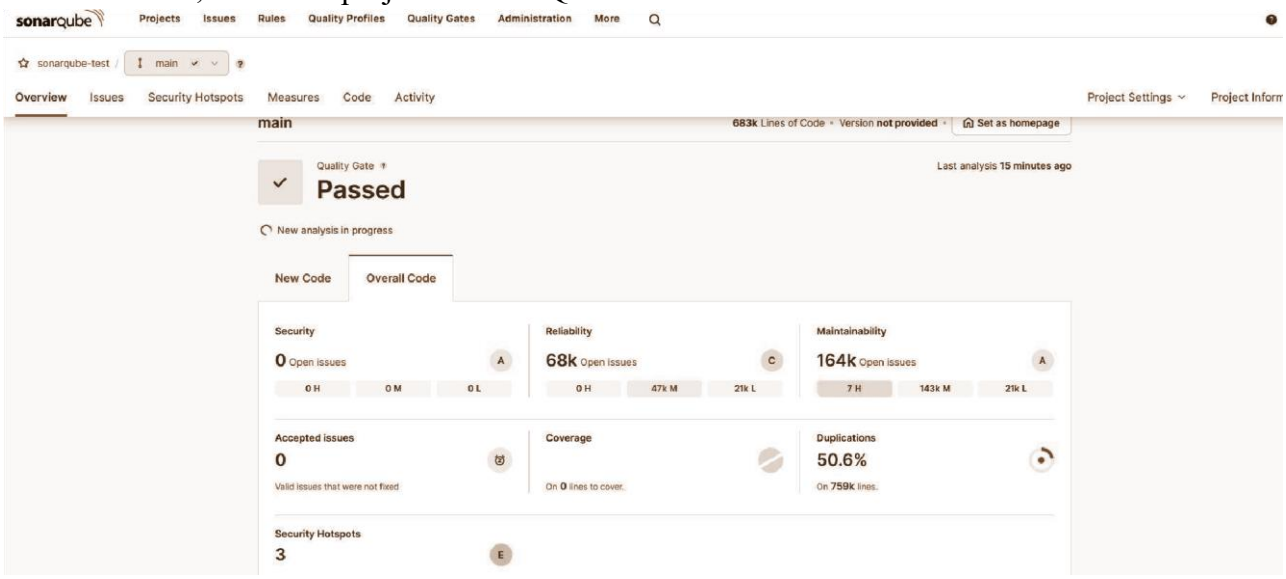
9. Check the console output once the build is complete.






```
references.
18:13:39.657 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/util/TextAreaCellRenderer.html for block at line 75. Keep only the first 100
references.
18:13:39.657 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/util/TextAreaCellRenderer.html for block at line 41. Keep only the first 100
references.
18:13:39.657 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/util/TextAreaCellRenderer.html for block at line 17. Keep only the first 100
references.
18:13:39.657 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/util/TextAreaCellRenderer.html for block at line 296. Keep only the first 100
references.
18:13:39.657 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/util/TextAreaCellRenderer.html for block at line 75. Keep only the first 100
references.
18:13:39.657 INFO  CPD Executor CPD calculation finished (done) | time=158971ms
18:13:39.674 INFO  SCM revision ID 'ba799ba7e1b576f04a4612322b0412c5e6e1e5e4'
18:15:49.696 INFO  Analysis report generated in 5022ms, dir size=127.2 MB
18:16:08.759 INFO  Analysis report compressed in 19048ms, zip size=29.6 MB
18:16:09.884 INFO  Analysis report uploaded in 1125ms
18:16:09.887 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://127.0.0.1:9001/dashboard?id=sonarqube-test
18:16:09.887 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
18:16:09.887 INFO  More about the report processing at http://127.0.0.1:9001/api/ce/task?id=6f22c333-3777-4a21-b058-0ab4c049625c
18:16:22.970 INFO  Analysis total time: 12:02.242 s
18:16:22.975 INFO  SonarScanner Engine completed successfully
18:16:23.699 INFO  EXECUTION SUCCESS
18:16:23.706 INFO  Total time: 12:05.758s
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

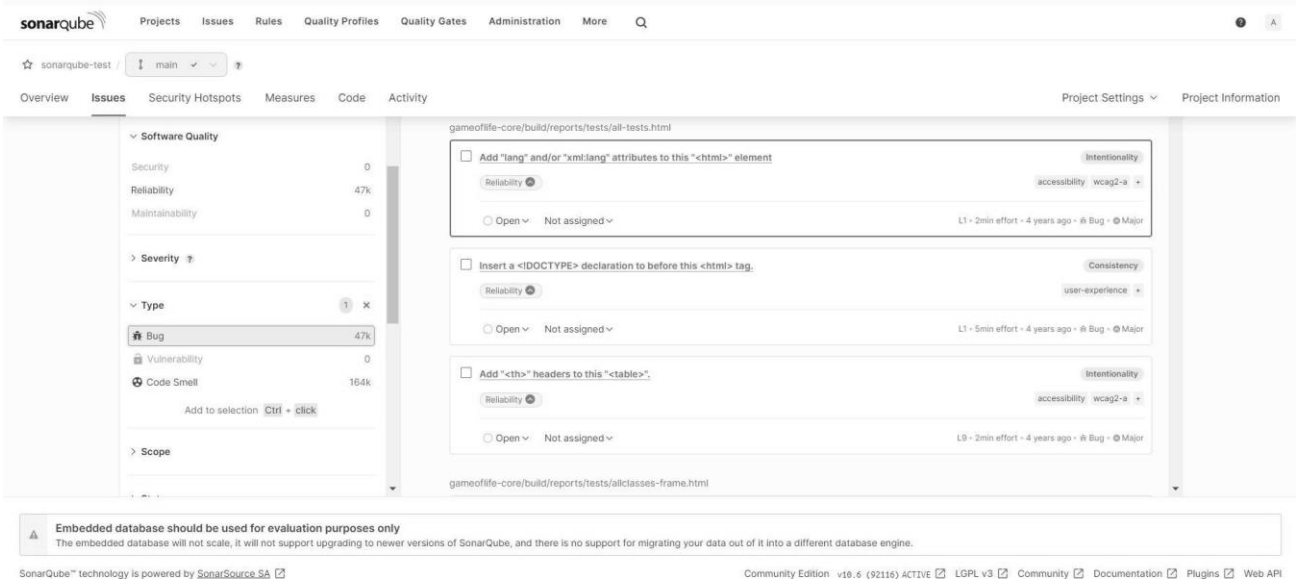
10. After that, check the project in SonarQube.



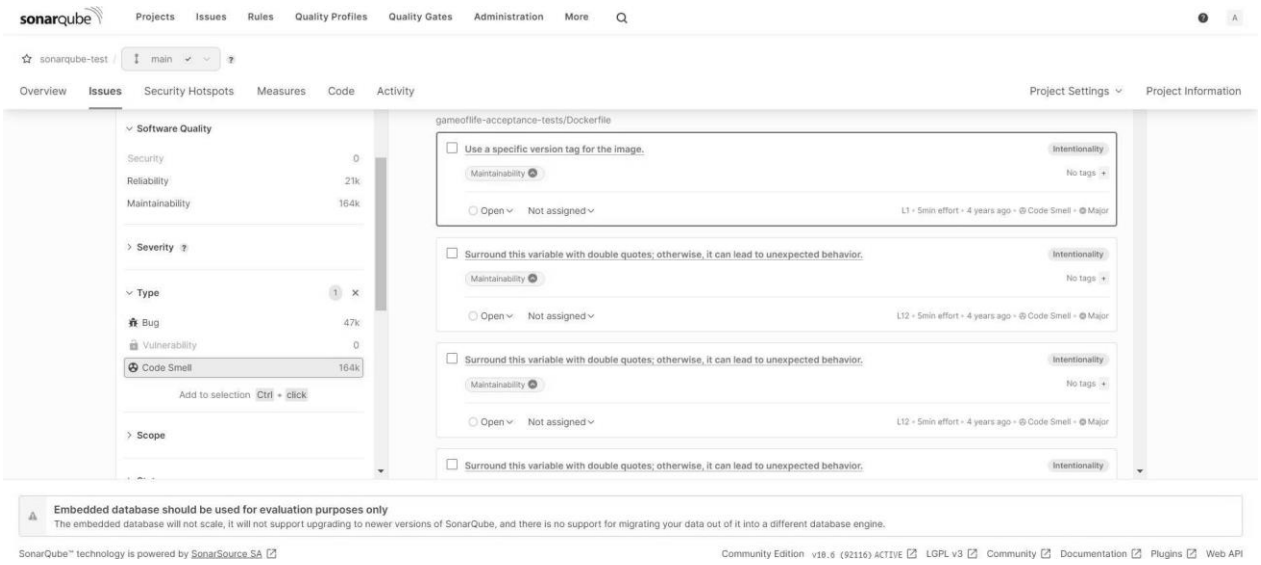
Under different tabs, check all different issues with the code.

11. **Code Problems -**

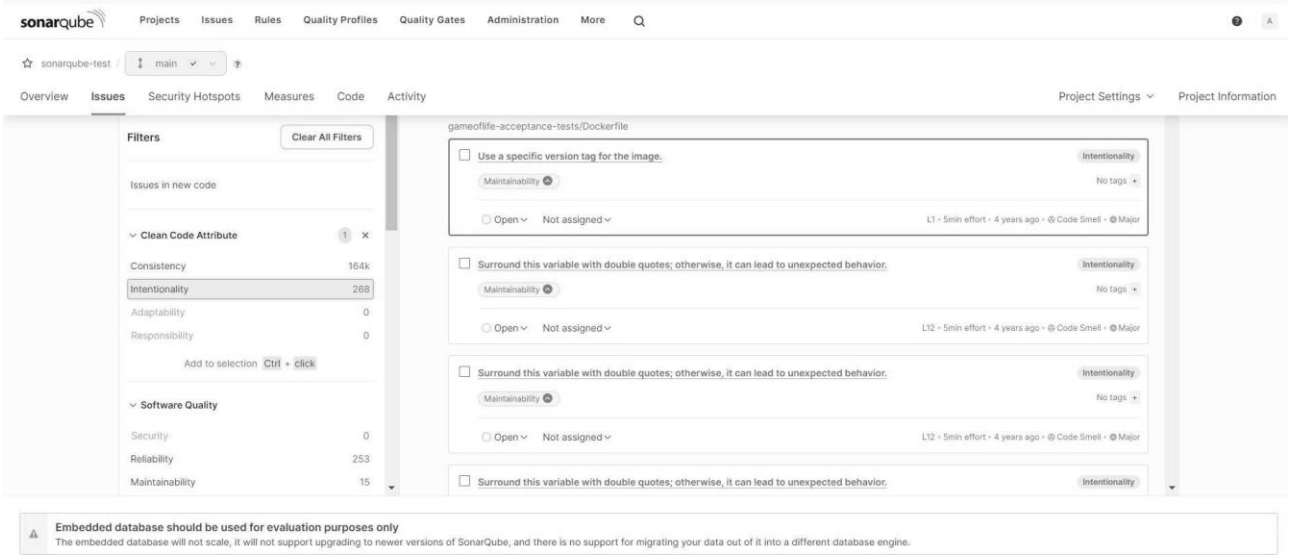
Bugs



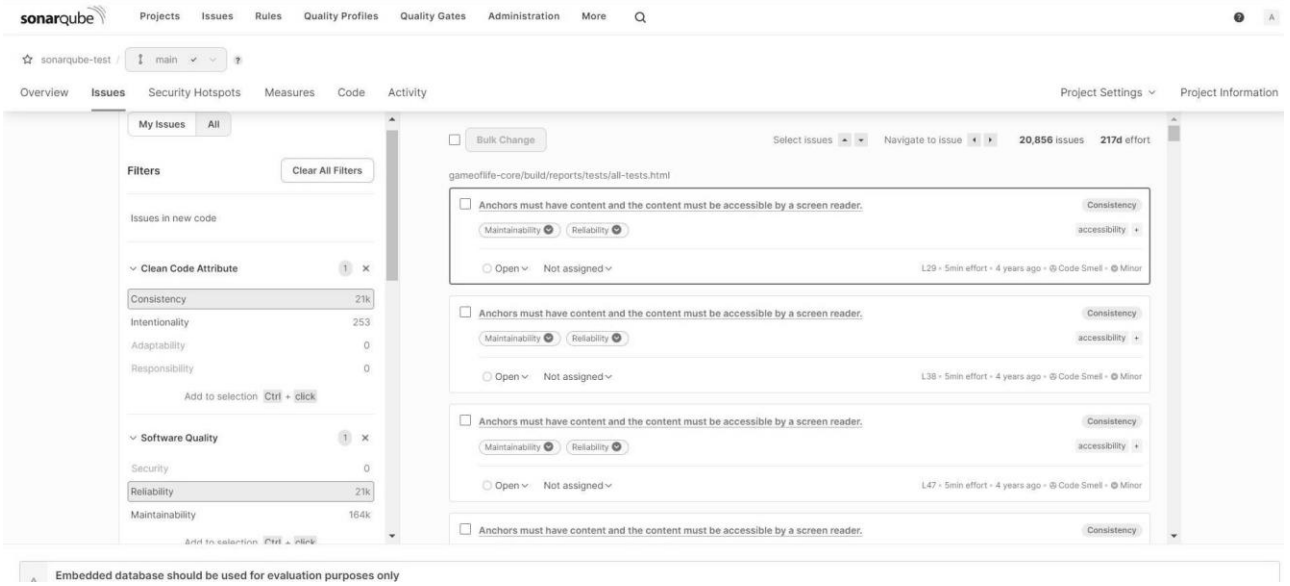
Code Smells



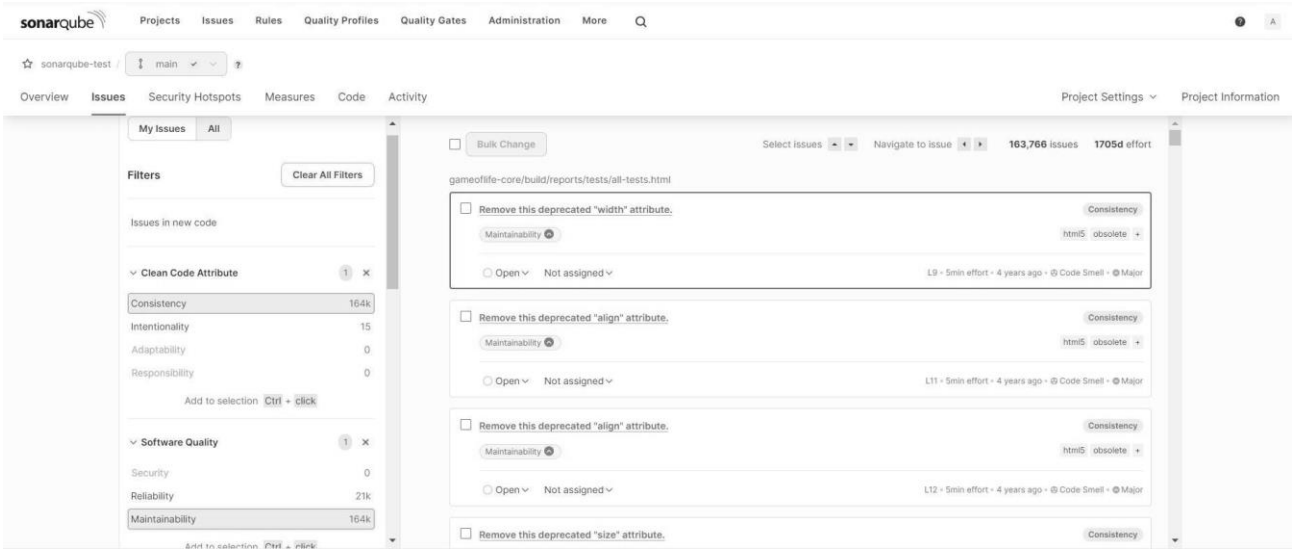
Intentional Issues



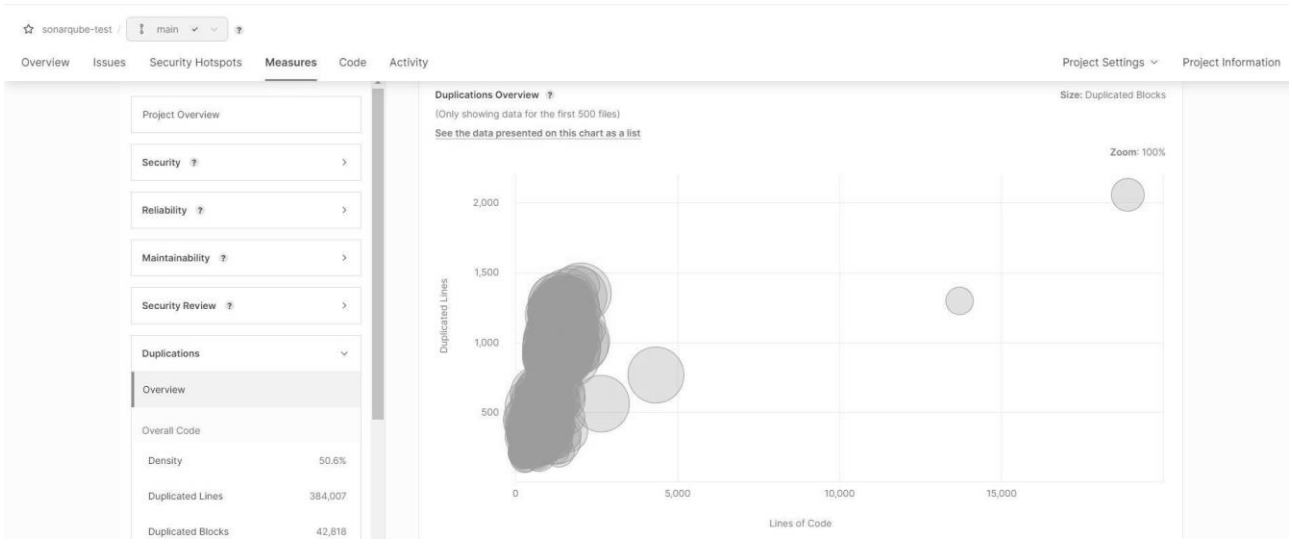
Reliability Issue



Maintainability Issue



Duplicates



In this way, we have created a CI/CD Pipeline with Jenkins and integrated it with SonarQube to find issues in the code like bugs, code smells, duplicates, cyclomatic complexities, etc.

Conclusion:

In this experiment, I successfully created a CI/CD pipeline using Jenkins integrated with SonarQube for static code analysis on a sample Java application. I set up SonarQube in a Docker container and configured Jenkins to clone the GitHub repository and perform the analysis. The pipeline detected various issues, including bugs, code smells, and security vulnerabilities, which I reviewed in SonarQube. This experience enhanced my skills in configuring CI/CD tools and highlighted the importance of maintaining code quality through automation. Overall, I gained valuable insights into integrating tools for effective software development practices.

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

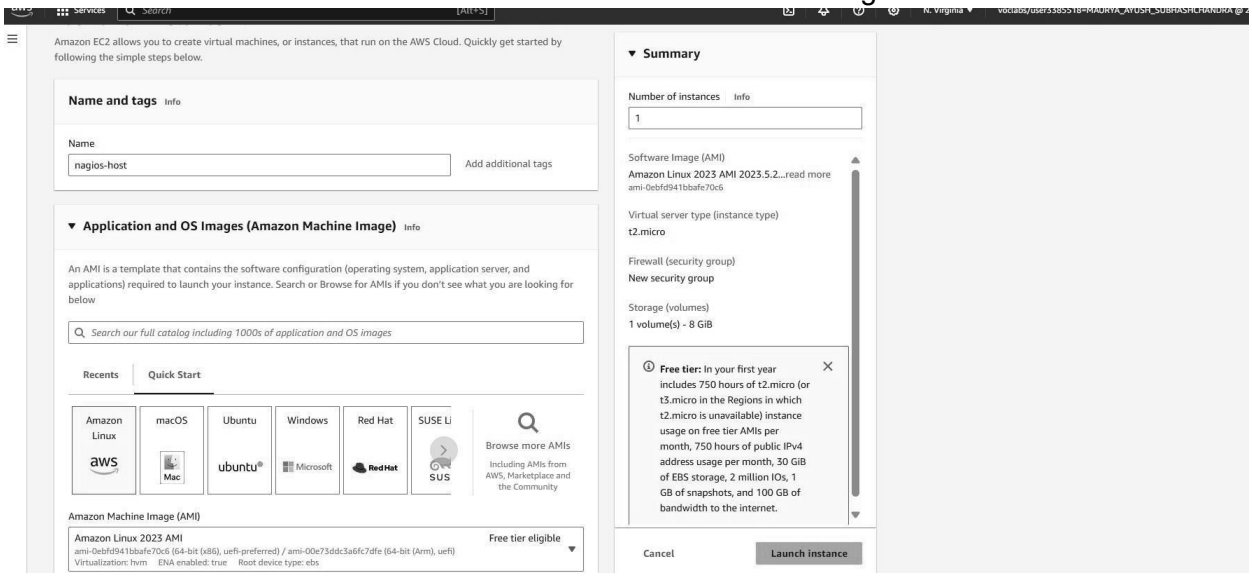
.

Installation of Nagios

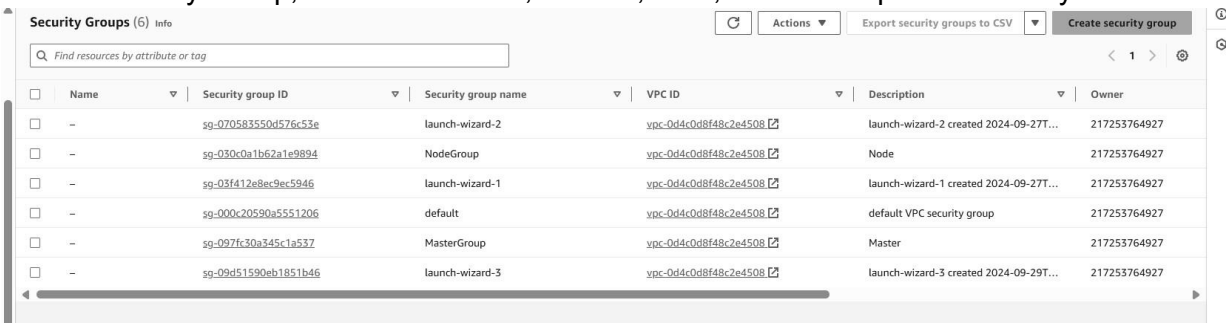
Prerequisites: AWS Free Tier

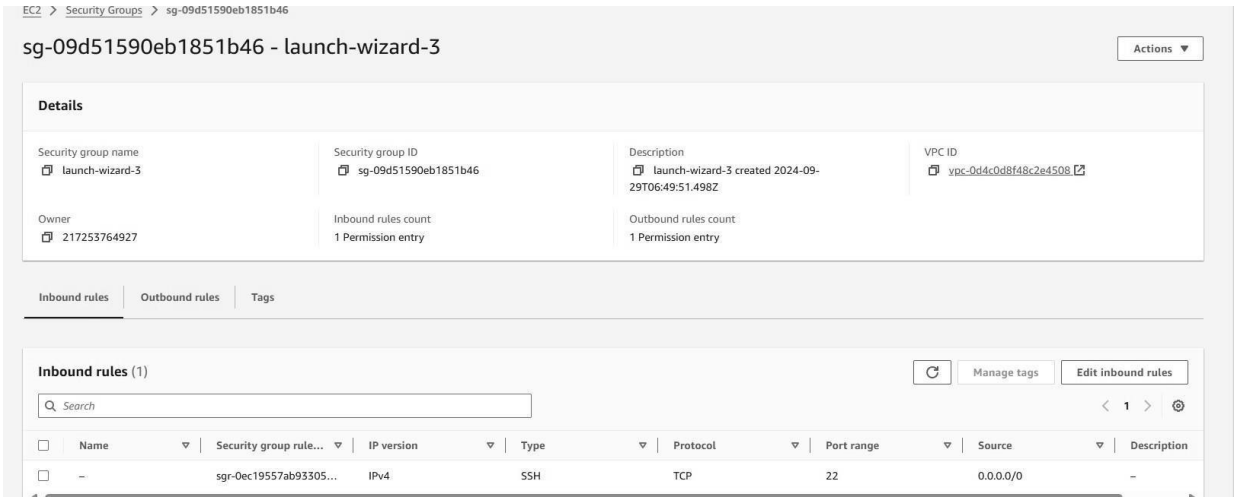
Steps:

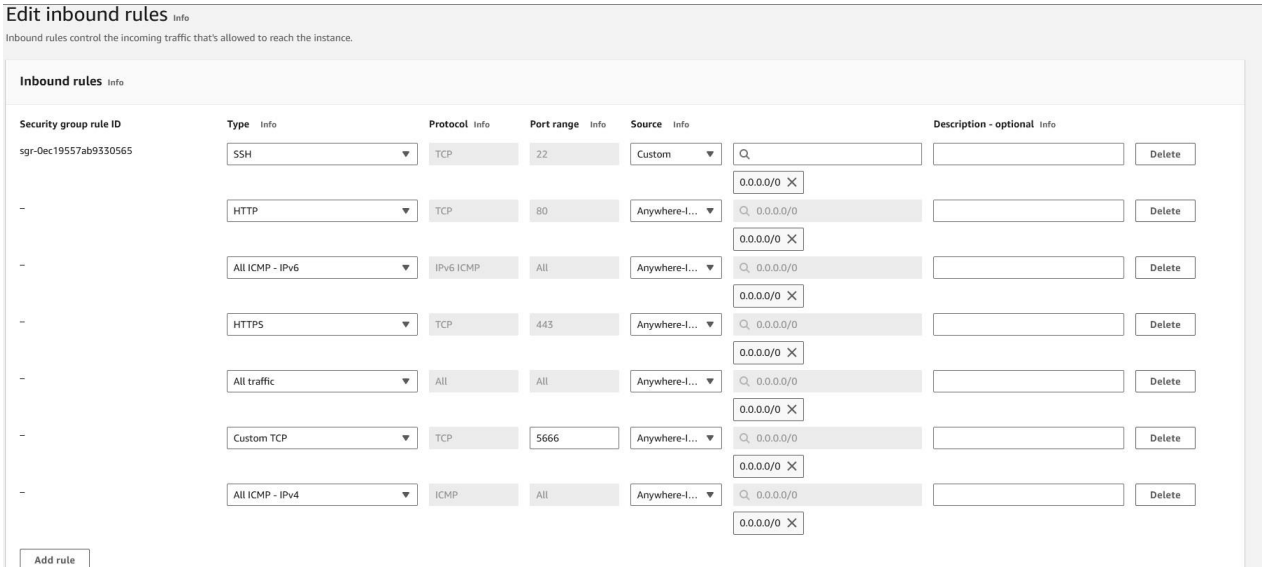
1. Create an Amazon Linux EC2 Instance in AWS and name it - nagios-host

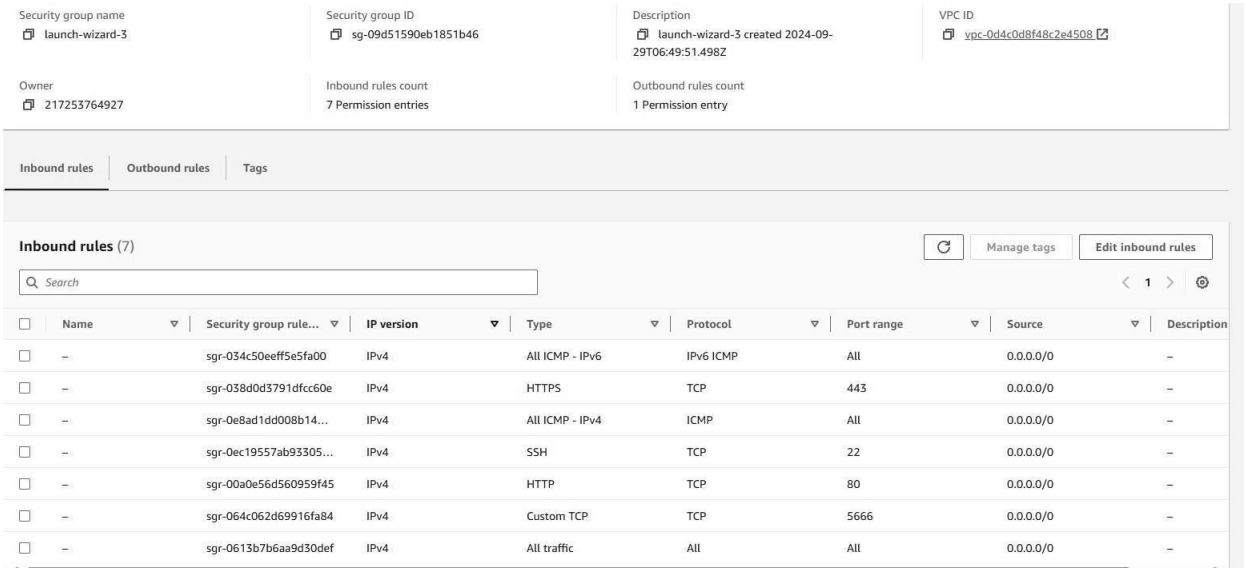


2. Under Security Group, make sure HTTP, HTTPS, SSH, ICMP are open from everywhere.



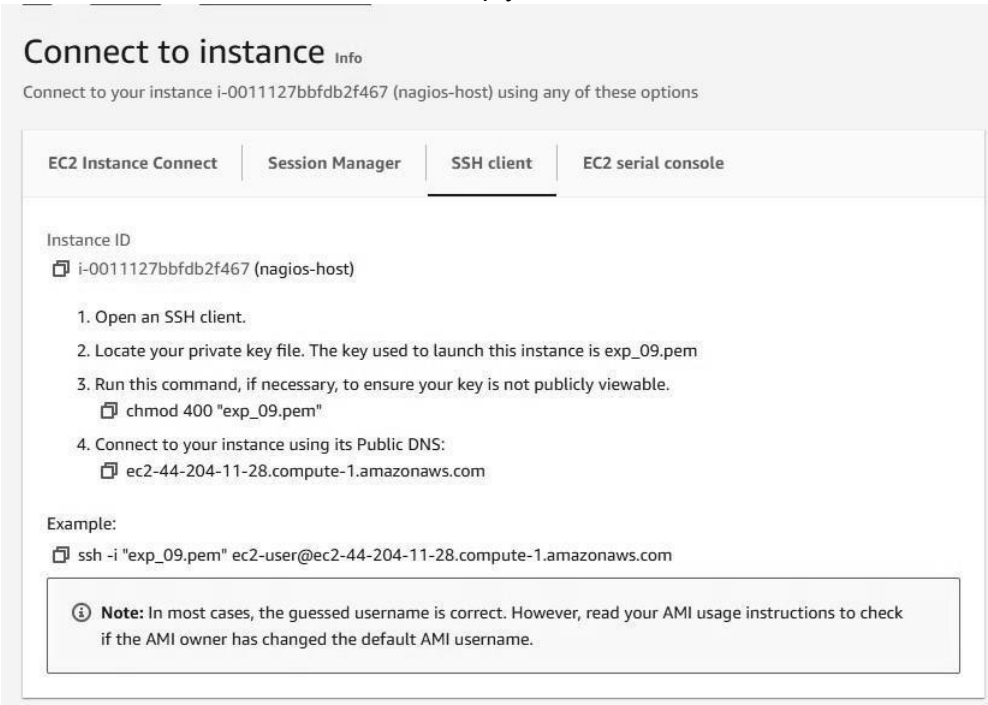

EC2 > Security Groups > sg-09d51590eb1851b46
sg-09d51590eb1851b46 - launch-wizard-3
Actions
Details
Security group name
launch-wizard-3
Security group ID
sg-09d51590eb1851b46
Description
launch-wizard-3 created 2024-09-29T06:49:51.498Z
VPC ID
vpc-0d4c0d8f48c2e4508
Owner
217253764927
Inbound rules count
1 Permission entry
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Tags
Inbound rules (1)
Manage tags
Edit inbound rules
Search
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-0ec19557ab9330565
SSH
TCP
22
Custom
0.0.0.0/0
Delete
HTTP
80
Anywhere-I...
All ICMP - IPv6
IPv6 ICMP
All
HTTPS
443
All traffic
Custom TCP
5666
All ICMP - IPv4
ICMP
Add rule


Security group name
launch-wizard-3
Security group ID
sg-09d51590eb1851b46
Description
launch-wizard-3 created 2024-09-29T06:49:51.498Z
VPC ID
vpc-0d4c0d8f48c2e4508
Owner
217253764927
Inbound rules count
7 Permission entries
Outbound rules count
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
HTTPS
TCP
443
sgr-0e8ad1dd008b14...
All ICMP - IPv4
ICMP
sgr-0ec19557ab93305...
SSH
22
sgr-00a0e56d560959f45
HTTP
80
sgr-064c062d69916fa84
Custom TCP
5666
sgr-0613b7b6aa9d30def
All traffic

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.



Or open command prompt and paste ssh command.

```
Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Ayush Maurya>ssh -i "Downloads/exp_09.pem" ec2-user@ec2-44-204-11-28.compute-1.amazonaws.com
The authenticity of host 'ec2-44-204-11-28.compute-1.amazonaws.com (44.204.11.28)' can't be established.
ED25519 key fingerprint is SHA256:v2OKH/ezl9iu7/RT6m8LWkgWzEJnnQIqrG9gKWzwC14.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-44-204-11-28.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sun Sep 29 07:11:40 2024 from 18.206.107.27
[ec2-user@ip-172-31-91-91 ~]$
```

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

Transaction Summary
```

```
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 : *ayushmau)*

```
Complete!
[ec2-user@ip-172-31-91-91 ~]$ sudo adduser -m nagios
sudo passwd nagios
Changing password for user nagios.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
Sorry, passwords do not match.
New password:
BAD PASSWORD: The password contains the user name in some form
Retype new password:
Sorry, passwords do not match.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-91-91 ~]$
```

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**

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

~/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 go 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**

```
#
# CONTACTS
#
###############################################################################

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

define contact {

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



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

# We only have one contact in this simple configuration file, so there is
# no need to create more than one contact group.

define contactgroup {

    contactgroup_name       admins
    alias                   Nagios Administrators
    members                 nagiosadmin
}

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

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/nagi
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: Ayushmau

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
```

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-172-31-91-91 nagios-plugins-2.0.3]$ sudo systemctl status nagios
● nagios.service - LSB: Starts and stops the Nagios monitoring server
     Loaded: loaded (/etc/rc.d/init.d/nagios; generated)
     Active: active (running) since Sun 2024-09-29 08:04:30 UTC; 37s ago
       Docs: man:systemd-sysv-generator(8)
    Process: 68037 ExecStart=/etc/rc.d/init.d/nagios start (code=exited, status=0/SUCCESS)
      Tasks: 6 (limit: 1112)
     Memory: 2.0M
        CPU: 47ms
     CGroup: /system.slice/nagios.service
             ├─68059 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─68061 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─68062 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─68063 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─68064 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─68065 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: wproc: Registry request: name=Core Worker 68063;pid=68063
Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: wproc: Registry request: name=Core Worker 68062;pid=68062
Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: wproc: Registry request: name=Core Worker 68064;pid=68064
Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: wproc: Registry request: name=Core Worker 68061;pid=68061
Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: Warning: Could not open object cache file '/usr/local/nagios/var/objec>
Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmpx2N>
Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: Successfully launched command file worker with pid 68065
Sep 29 08:04:39 ip-172-31-91-91.ec2.internal nagios[68059]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmpTmg>
Sep 29 08:04:49 ip-172-31-91-91.ec2.internal nagios[68059]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmpAfy>
Sep 29 08:04:59 ip-172-31-91-91.ec2.internal nagios[68059]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmpCtQ>
lines 1-26/26 (END)
```

Error:

The log messages suggest that Nagios is unable to create temporary files, particularly in the directory /usr/local/nagios/var/. This is typically caused by permission issues, or the directory might not exist. ***Solution:***

Firstly check whether /usr/local/nagios/var/ is there or not. If yes.....

ls -ld /usr/local/nagios/var/

Change ownership: Set the correct ownership for the Nagios user and group:

sudo chown -R nagios:nagcmd /usr/local/nagios/var

Set permissions: Ensure the directory has the right permissions:

sudo chmod -R 775 /usr/local/nagios/var

Restart Nagios: After adjusting the ownership and permissions, restart the Nagios service:

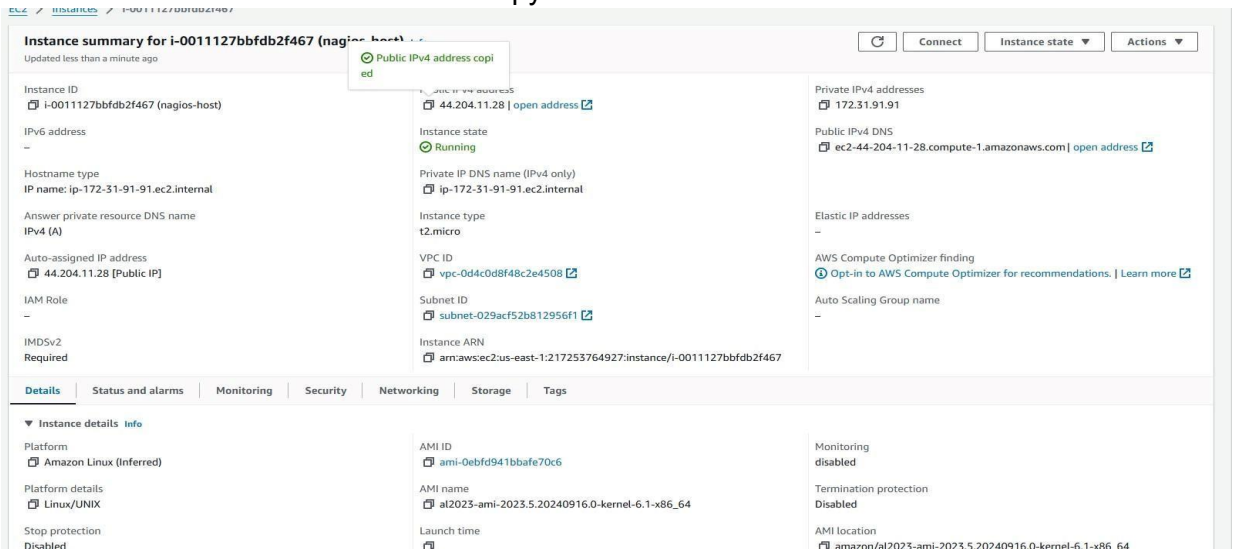
sudo systemctl restart nagios

```
drwxr-xr-x. 4 root root 112 Sep 29 08:04 /usr/local/nagios/var/
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$ sudo chown -R nagios:nagcmd /usr/local/nagios/var
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$ sudo chmod -R 775 /usr/local/nagios/var
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$ sudo systemctl restart nagios
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$
```

Now run again

```
[ec2-user@ip-172-31-91-91 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 Sun 2024-09-29 08:51:47 UTC; 42min ago
       Docs: https://www.nagios.org/documentation
      Tasks: 6 (limit: 1112)
     Memory: 2.9M
        CPU: 562ms
     CGroup: /system.slice/nagios.service
             ├─71188 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─71190 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─71191 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─71192 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─71193 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─71194 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 29 08:51:47 ip-172-31-91-91.ec2.internal nagios[71188]: wproc: Registry request: name=Core Worker 71191;pid=71191
Sep 29 08:51:47 ip-172-31-91-91.ec2.internal nagios[71188]: wproc: Registry request: name=Core Worker 71190;pid=71190
Sep 29 08:51:47 ip-172-31-91-91.ec2.internal nagios[71188]: Successfully launched command file worker with pid 71194
Sep 29 08:59:22 ip-172-31-91-91.ec2.internal nagios[71188]: SERVICE ALERT: localhost;HTTP;WARNING;HARD;4;HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes i>
Sep 29 09:11:52 ip-172-31-91-91.ec2.internal nagios[71188]: SERVICE NOTIFICATION: nagiosadmin;localhost;Swap Usage;CRITICAL;notify-service-by-email;SWAP CR>
Sep 29 09:11:52 ip-172-31-91-91.ec2.internal nagios[71188]: wproc: NOTIFY job 10 from worker Core Worker 71192 is a non-check helper but exited with return>
Sep 29 09:11:52 ip-172-31-91-91.ec2.internal nagios[71188]: wproc:   host=localhost; service=Swap Usage; contact=nagiosadmin
Sep 29 09:11:52 ip-172-31-91-91.ec2.internal nagios[71188]: wproc:   early_timeout=0; exited_ok=1; wait_status=32512; error_code=0;
Sep 29 09:11:52 ip-172-31-91-91.ec2.internal nagios[71188]: wproc:   stderr line 01: /bin/sh: line 1: /bin/mail: No such file or directory
Sep 29 09:11:52 ip-172-31-91-91.ec2.internal nagios[71188]: wproc:   stderr line 02: /usr/bin/printf: write error: Broken pipe
lines 1-25/25 (END)
```

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

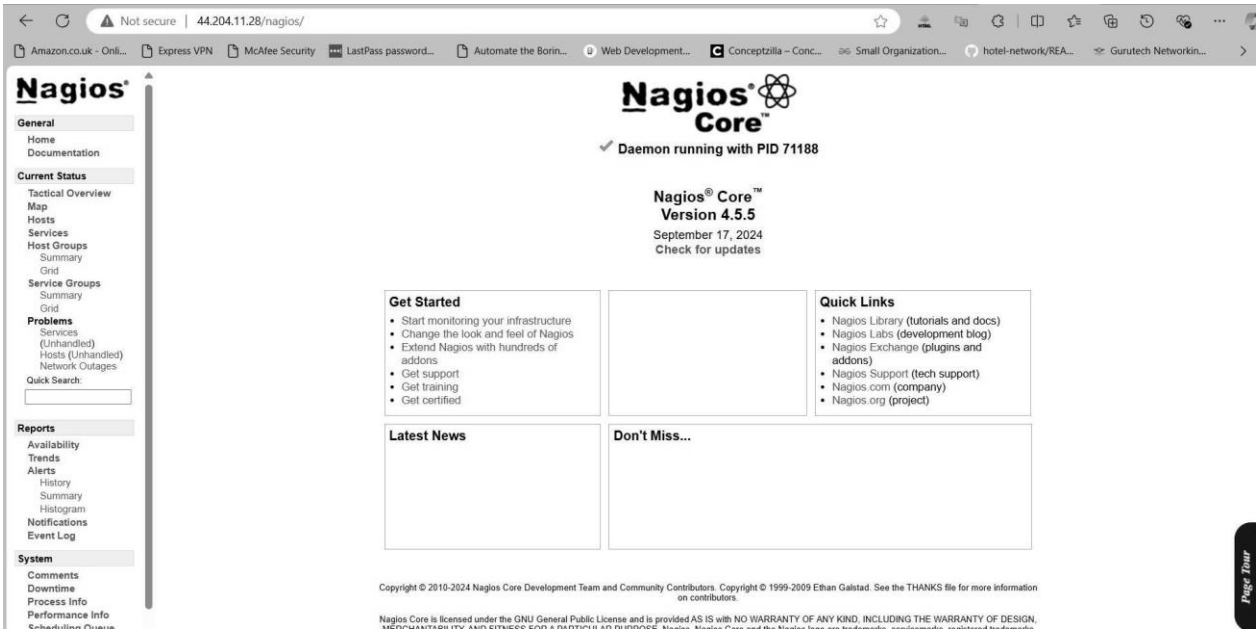


23. Open up your browser and look for

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)

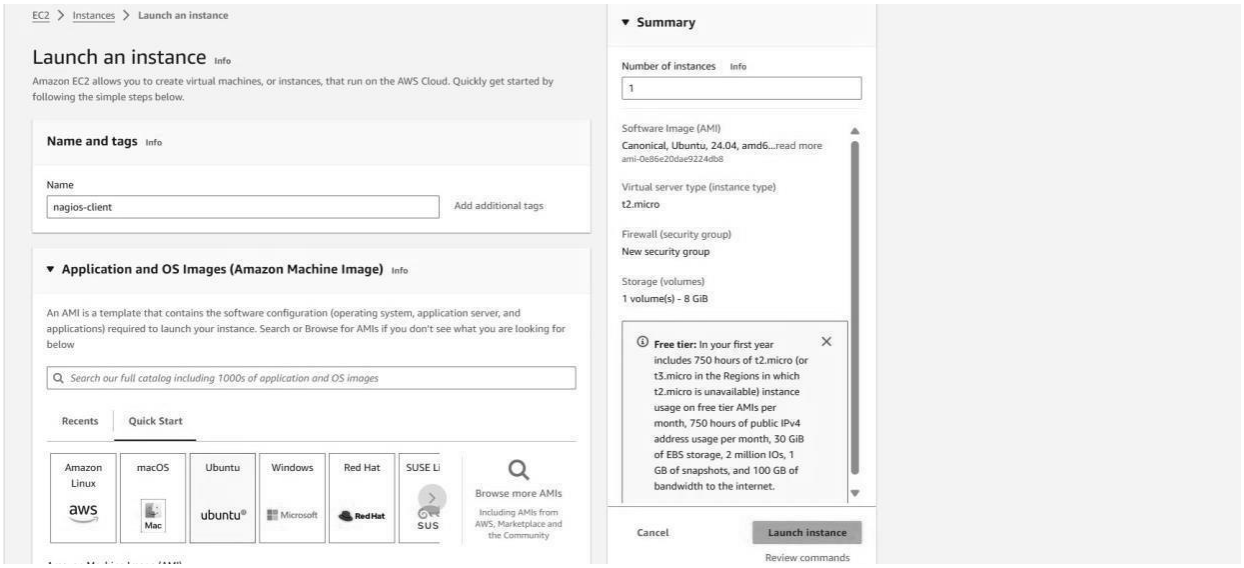
Monitoring Using Nagios:

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)
```

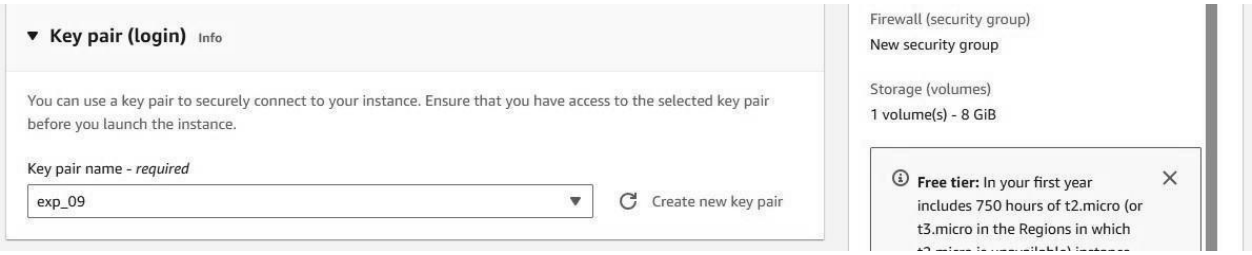
You can now proceed if you get the above message/output.

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

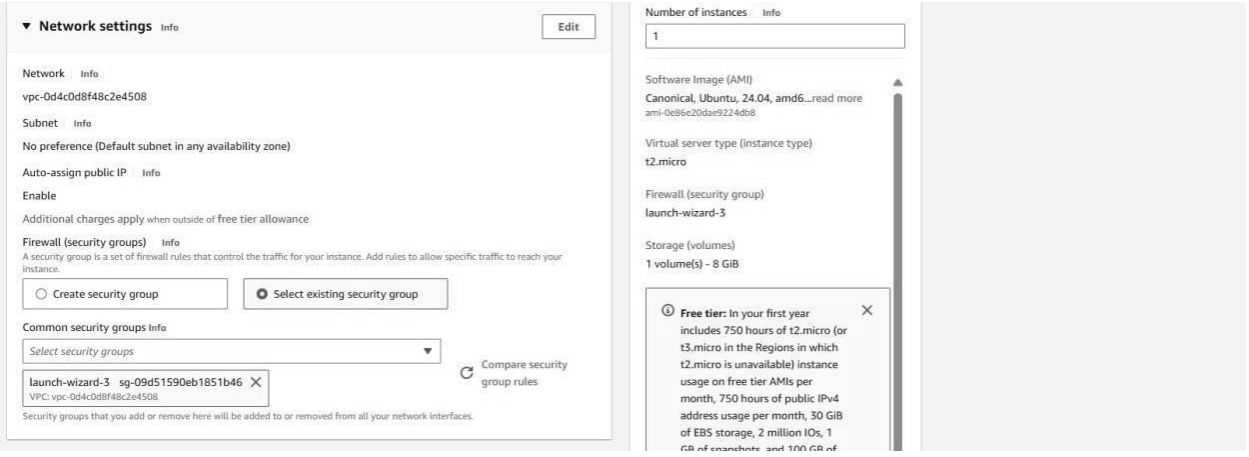


For Key pair : Click on create key and make key of type RSA with extension .pem . Key will be downloaded to your local machine.

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).



Step 3: Now After creating the EC2 Instance click on connect and then copy the command which is given as example in the SSH Client section .
Now open the terminal in the folder where your key(RSA key with .pem) is located. and paste that copied command.

```
PS C:\Users\ MUSKAANNN ı> ssh -i "Downloads/exp_09.pem" ubuntu@ec2-44-206-245-149.compute-1.amazonaws.com
The authenticity of host 'ec2-44-206-245-149.compute-1.amazonaws.com (44.206.245.149)' can't be established.
ED25519 key fingerprint is SHA256:DT+AA+mkcydh3kOJ2vEpm4ZsA6FL+LM4m1QSImddAHg.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-44-206-245-149.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
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

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-172-31-92-146:~$
```

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 ~]$
```

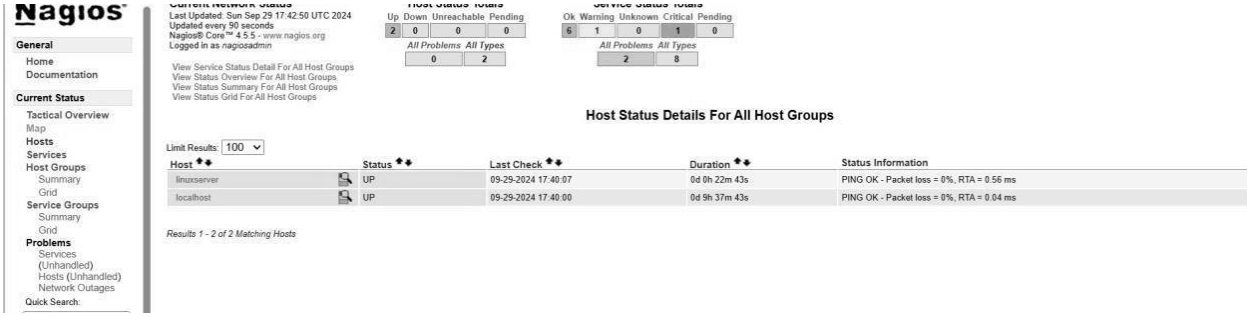
sudo systemctl start httpd
sudo systemctl enable httpd

```
[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



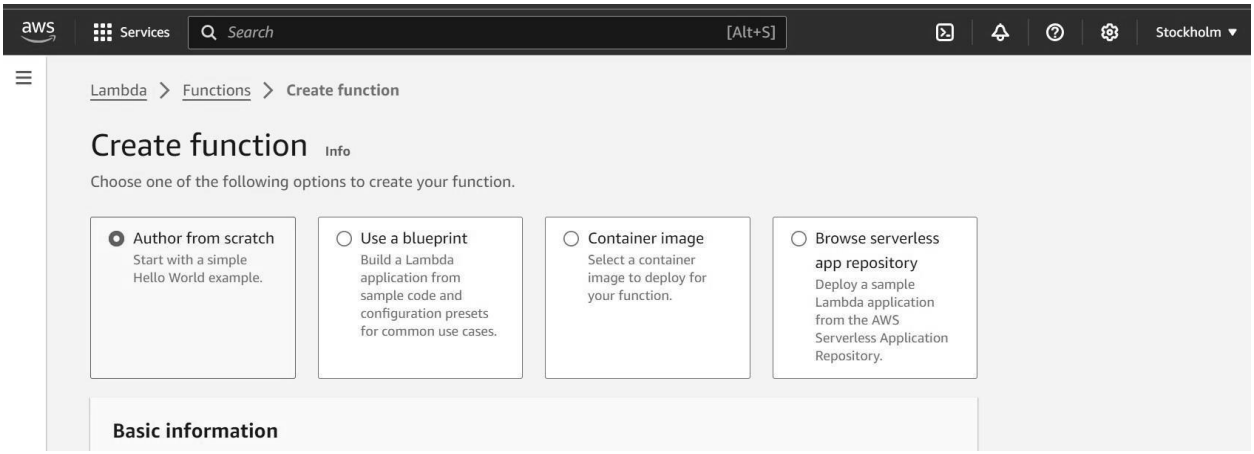
Conclusion:

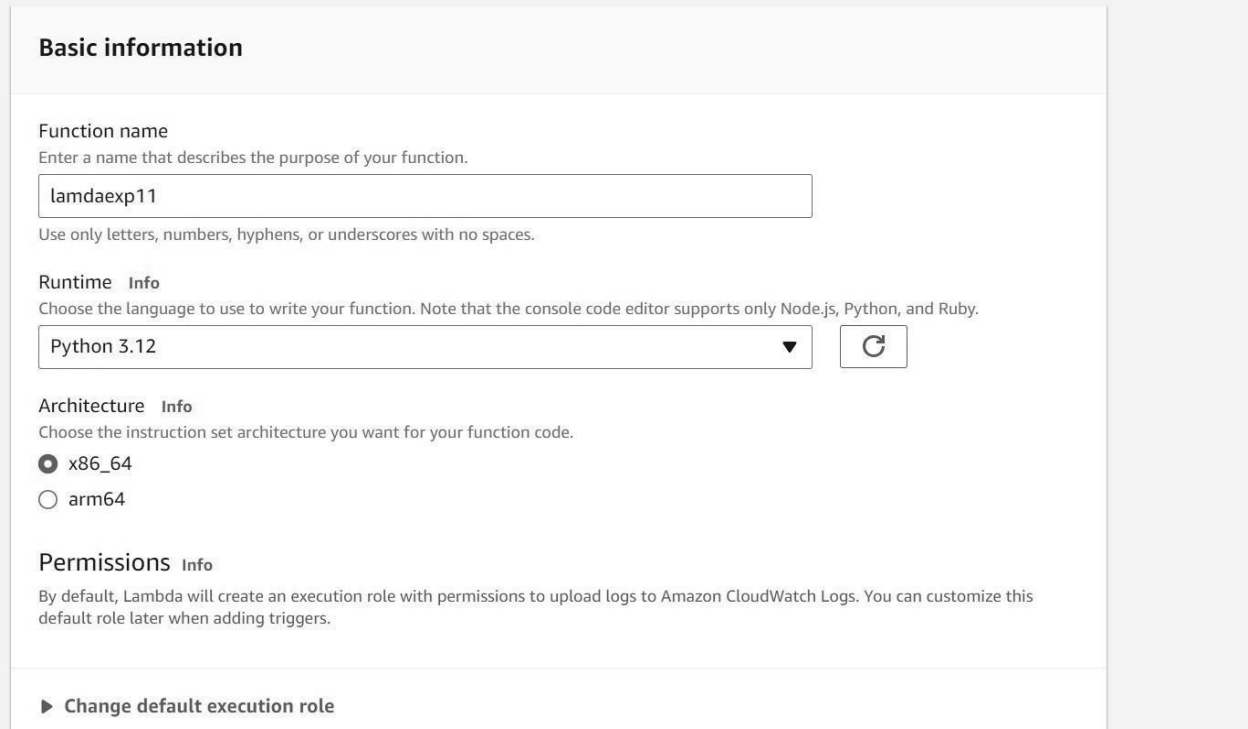
In this practical, we set up a Nagios host and client to monitor services and server performance on both Linux and Windows servers. We configured Nagios on an Amazon Linux machine to monitor critical services like HTTP, SSH, and system resources, ensuring their availability and health. By creating and configuring a new EC2 instance as the Nagios client, we enabled seamless communication between the client and host for efficient service monitoring. This setup helps ensure uptime and quick detection of issues across the infrastructure.

EXPERIMENT-11

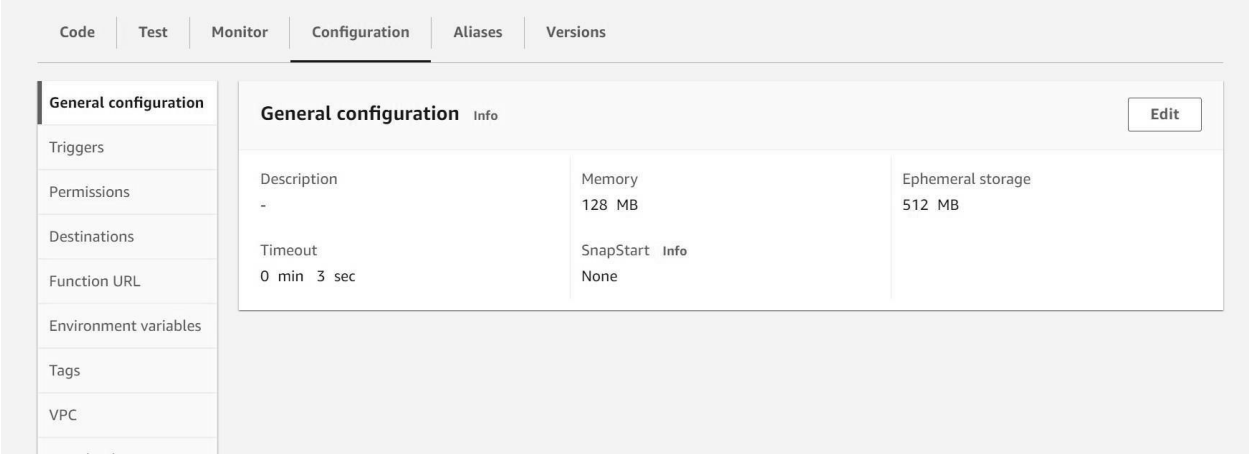
AIM :To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

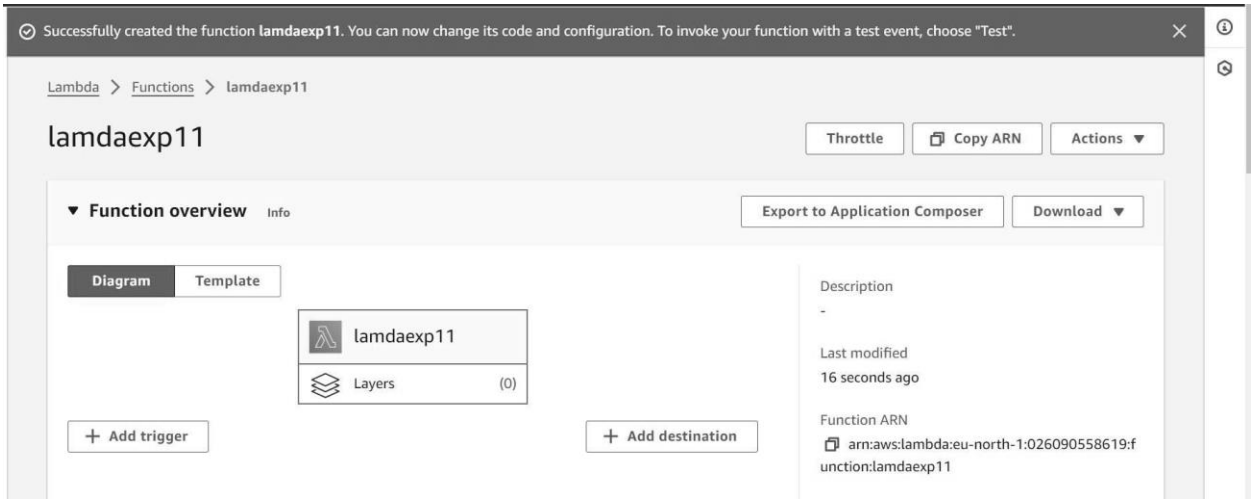
STEP1:Go on your AWS console account and search for lamba and then go on create function Select the author from scratch, add function name and then, choose a runtime env for your function, under the dropdown, you can see all the options AWS supports, Python, Nodejs, .NET and Java being the most popular ones.

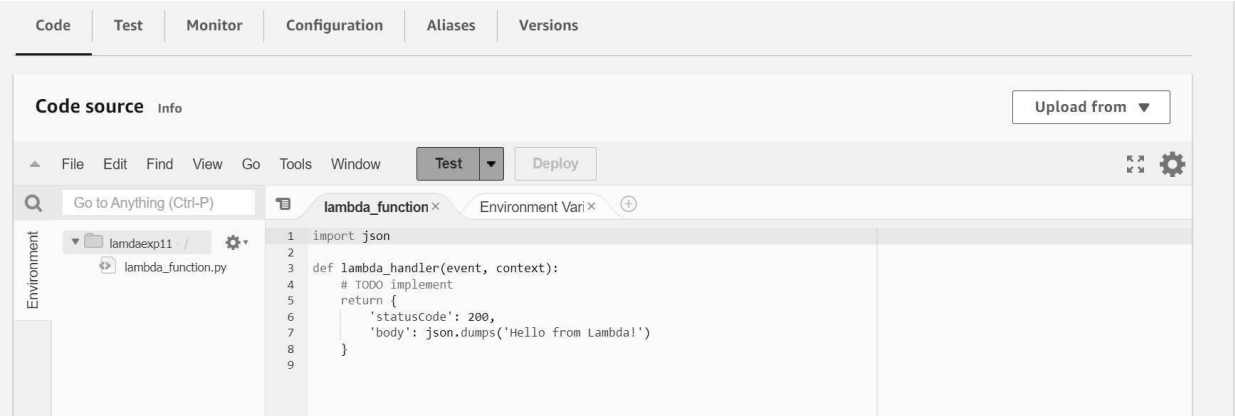




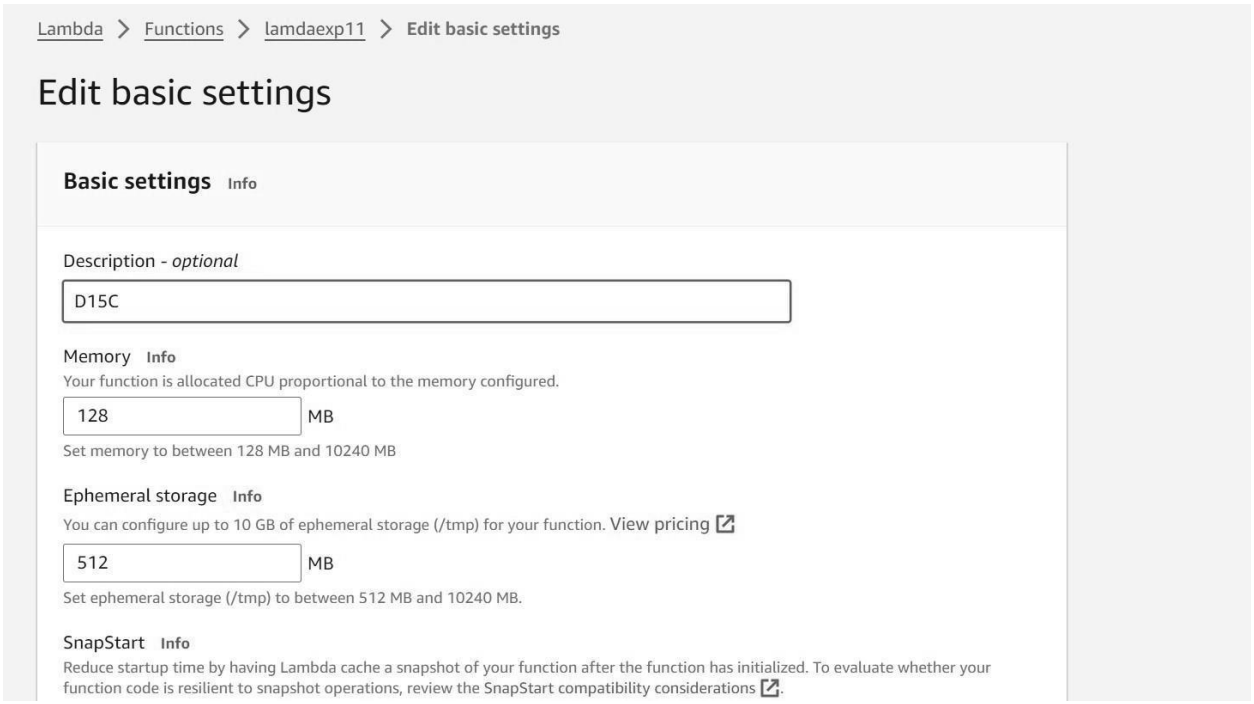
STEP 2:After the function is created successfully go on code write the default code and then configure them.



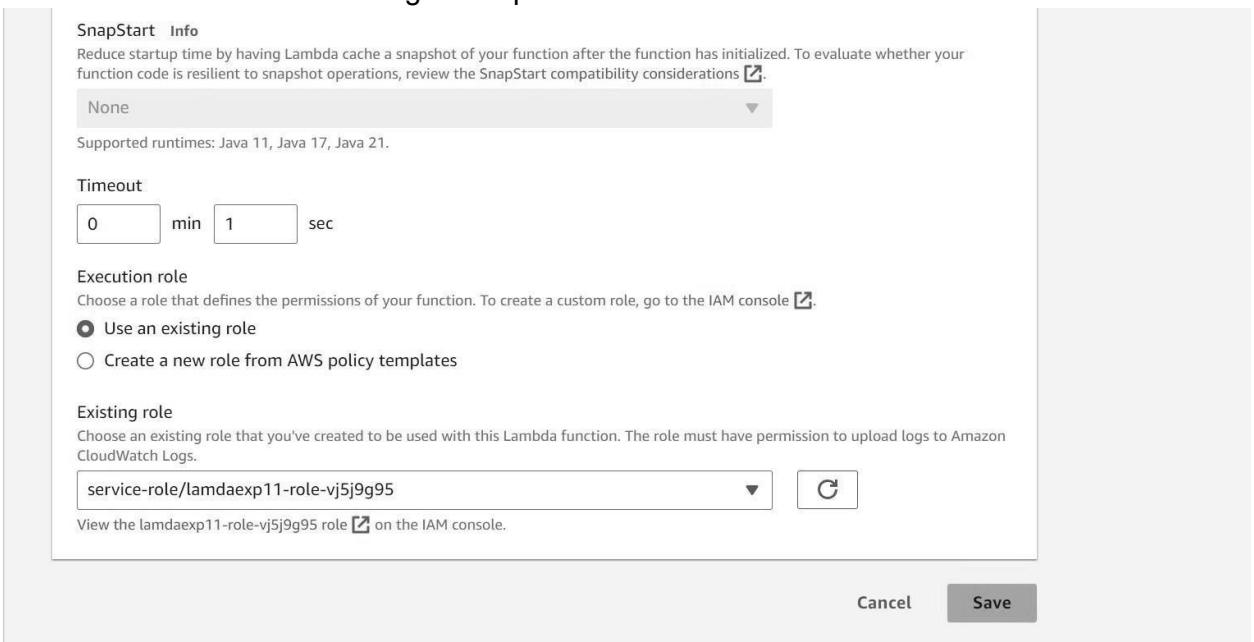




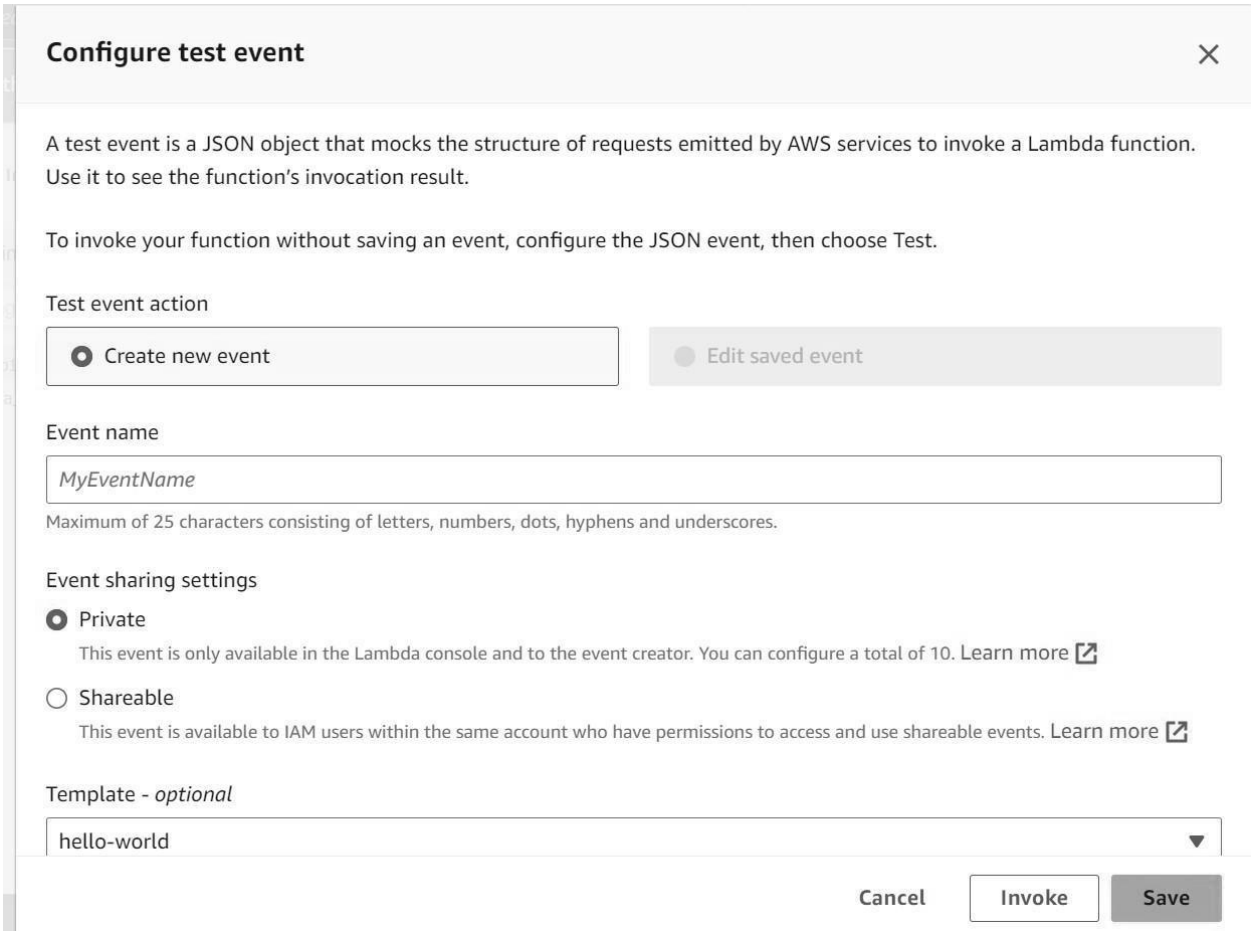
STEP 3: Then go on edit basic settings and add the description and then save it .



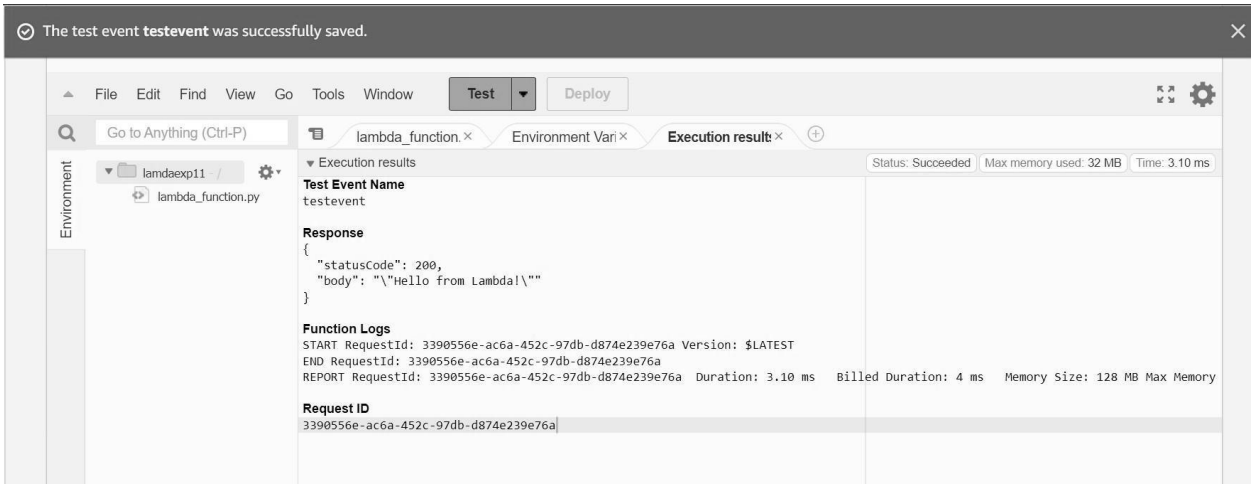
STEP 4:Click on “use an existing role “option and then ahead add the role and save it.



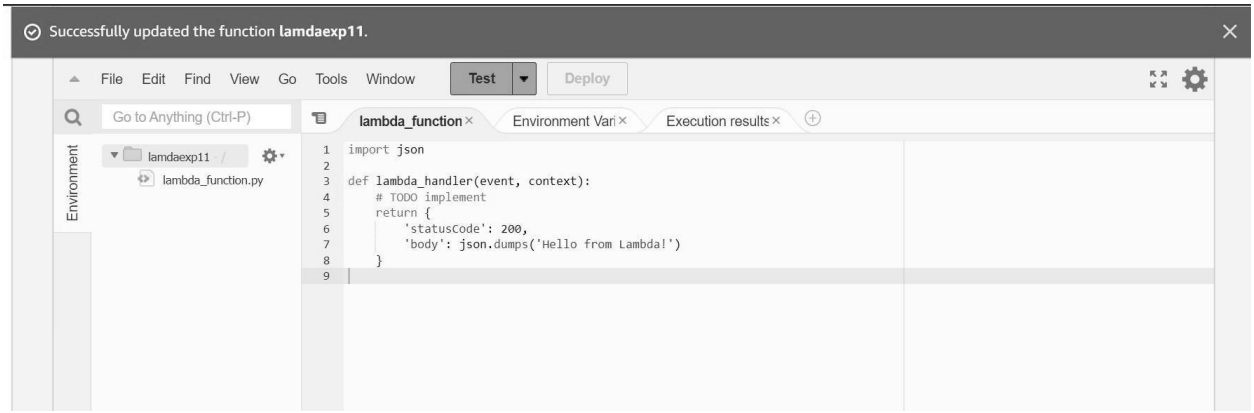
STEP 5:Go on configure test event click on “create new event” edit the event sharing accordingly and select hello world template for template option and then save it.



STEP 6: Click on the test and test the code.



STEP 7: The function is successfully added .



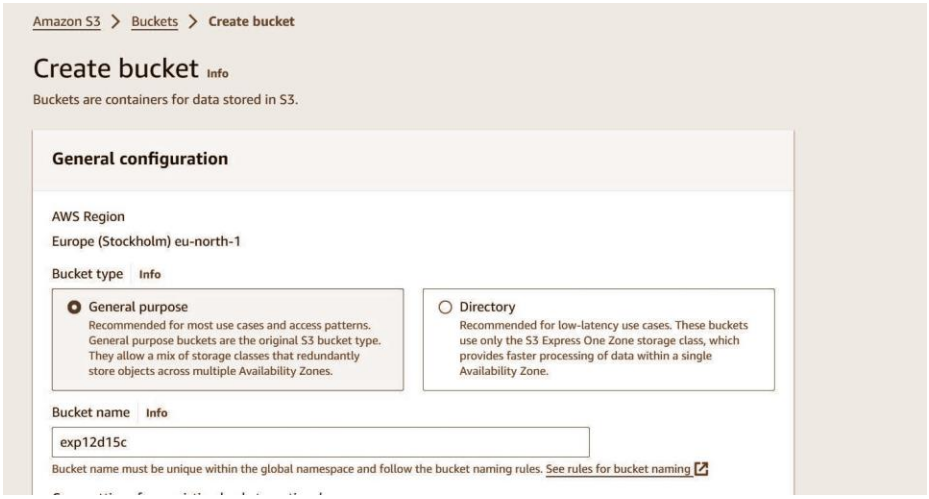
Conclusion:In conclusion, the experiment successfully involved the creation, coding, and deployment of AWS Lambda function. By writing and refining the source code, we demonstrated the ability to implement specific functionality within the Lambda environment. The successful testing of the function confirmed its operational integrity and effectiveness in executing the desired tasks

Roshan Bhagtani D15C Roll no. 4

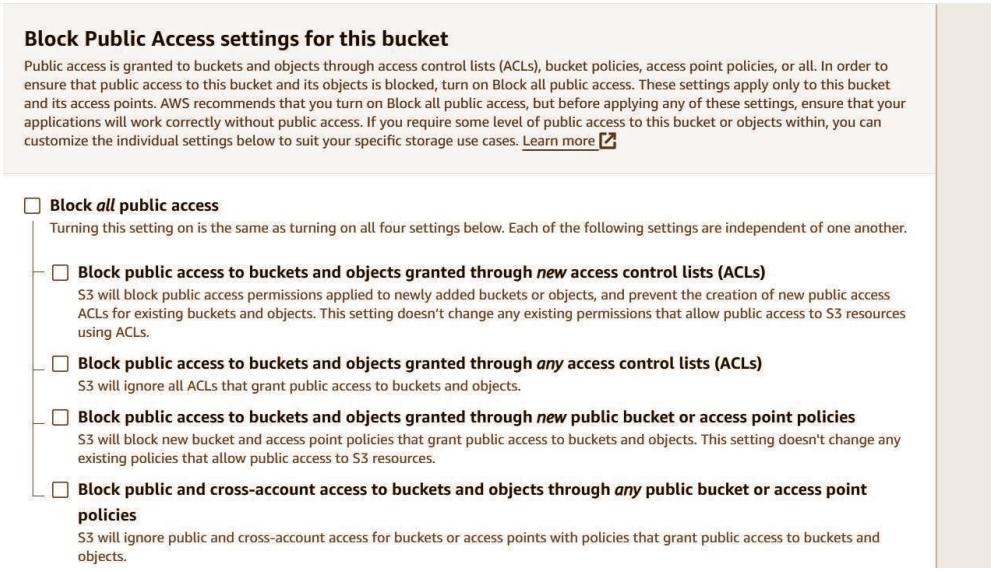
EXPERIMENT 12

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

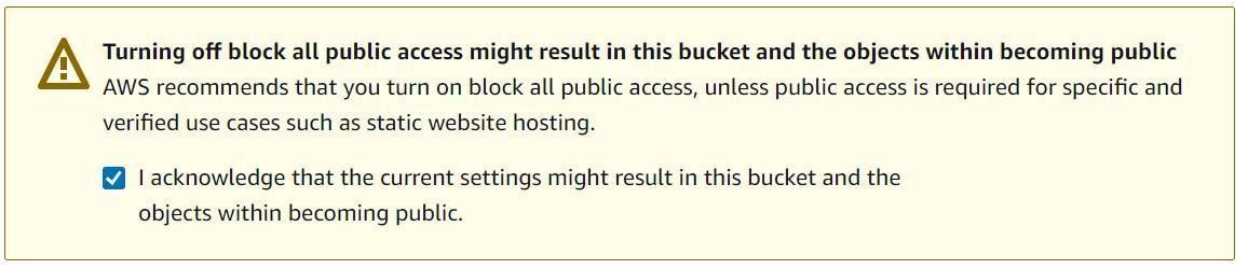
1. Create a S3 bucket and give it a bucket name



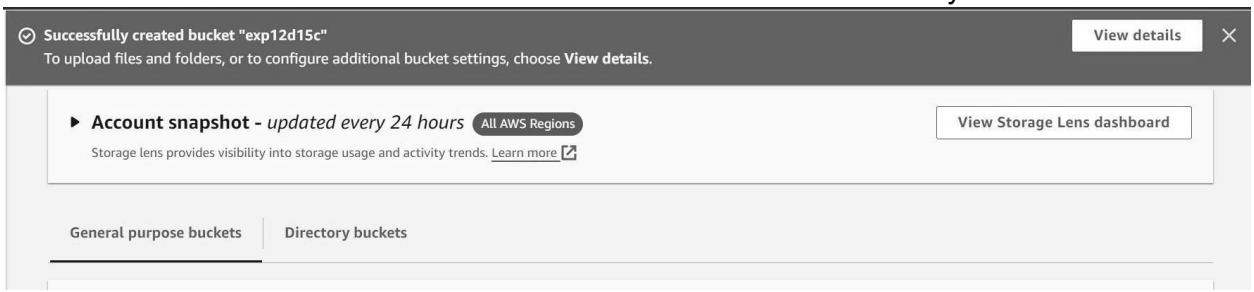
2. Allow public access to the bucket as we are going to add this bucket as a trigger for our lamda function



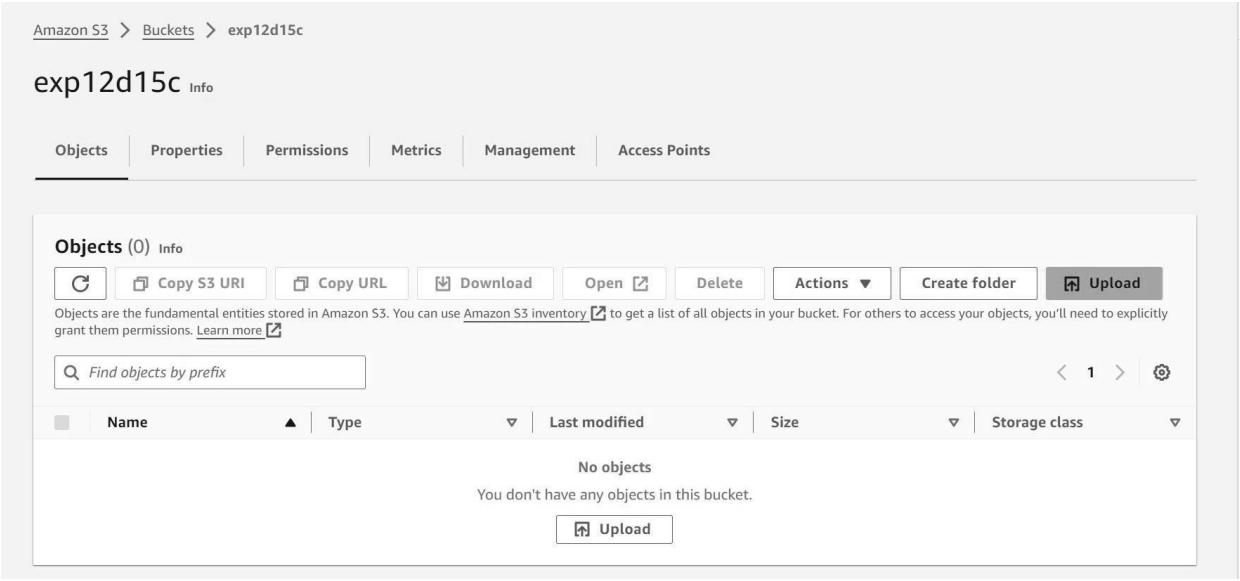
3. Give confirmation that you want to allow full public access and create the bucket



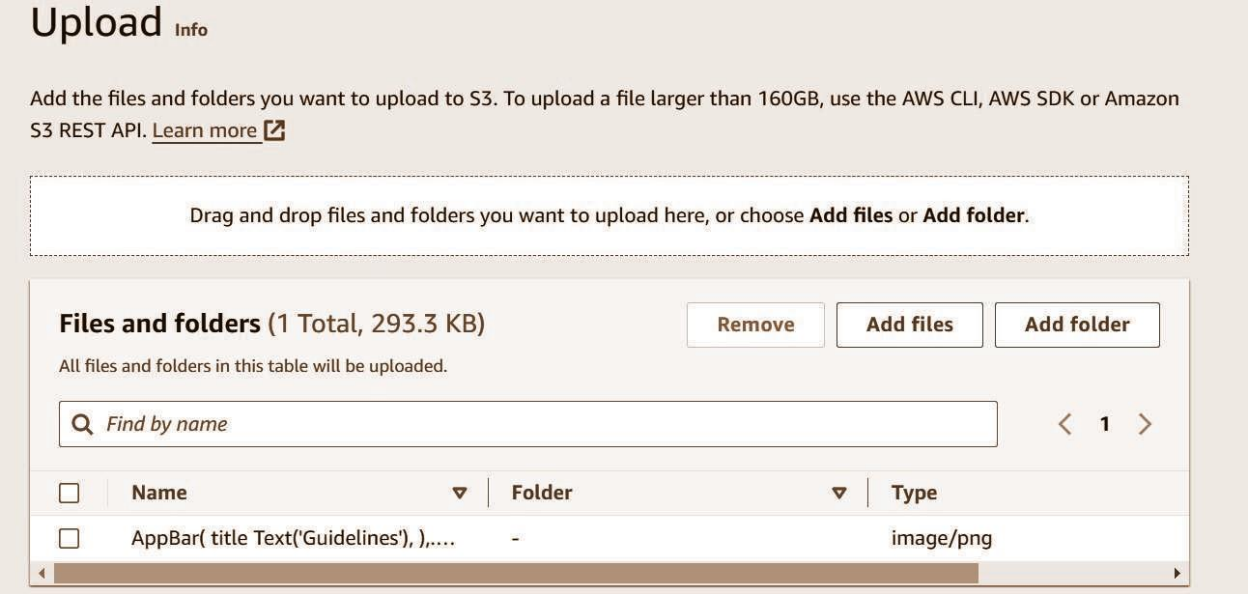
4. You will see the confirmation that the bucket is created successfully



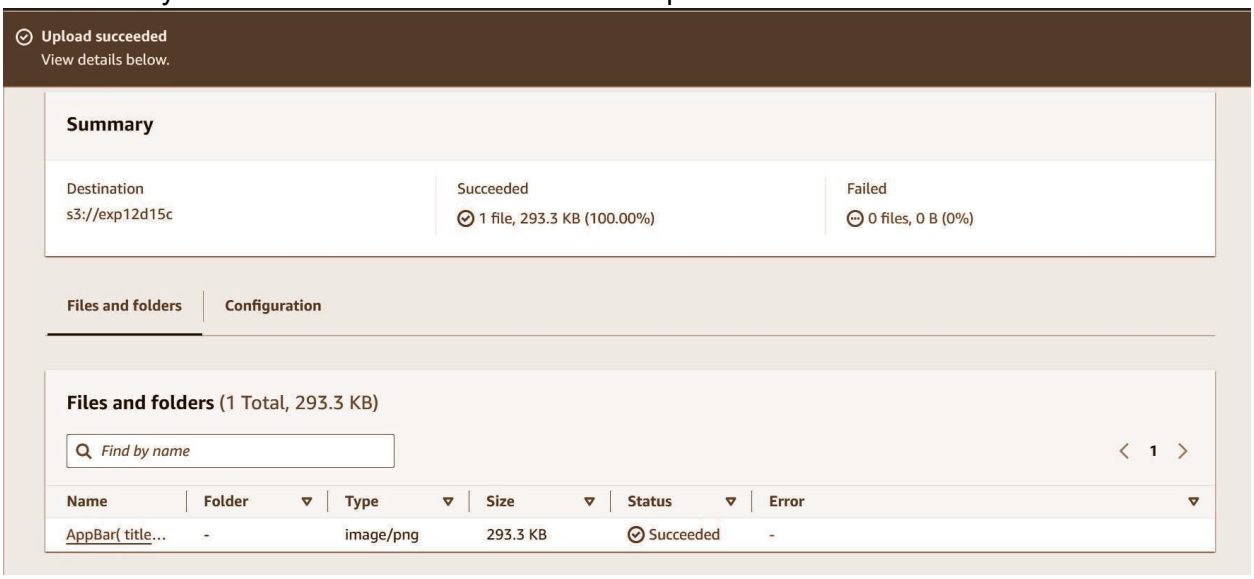
5. Now we need to upload something in the bucket so click on the upload button and add a file



6. I have added a .png extension file; You can upload a .txt file as well

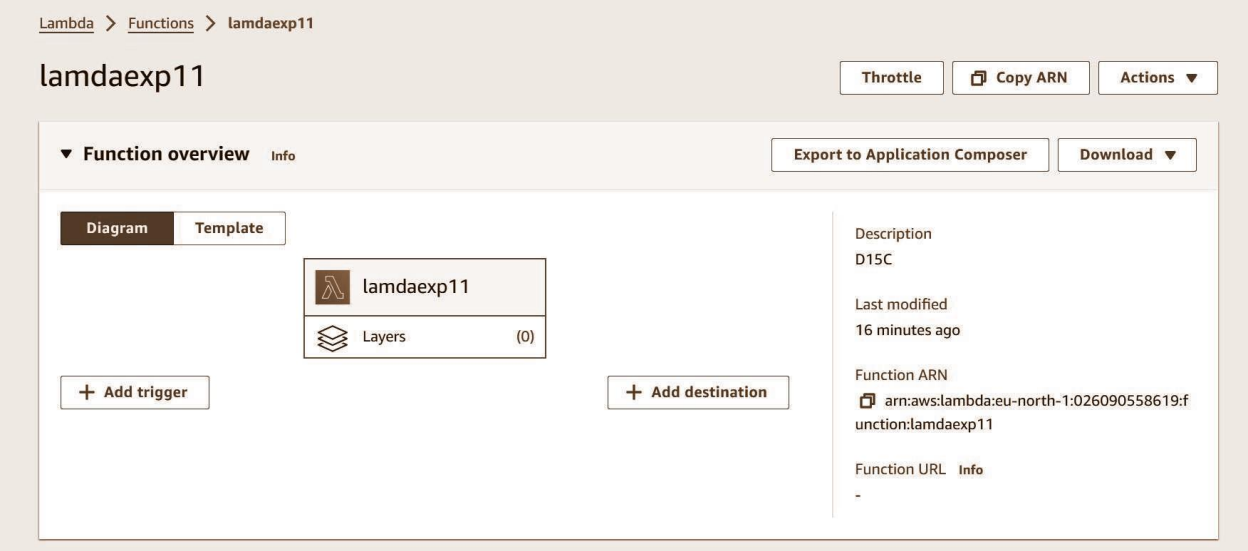


7. Here you can see the confirmation that the upload was a success

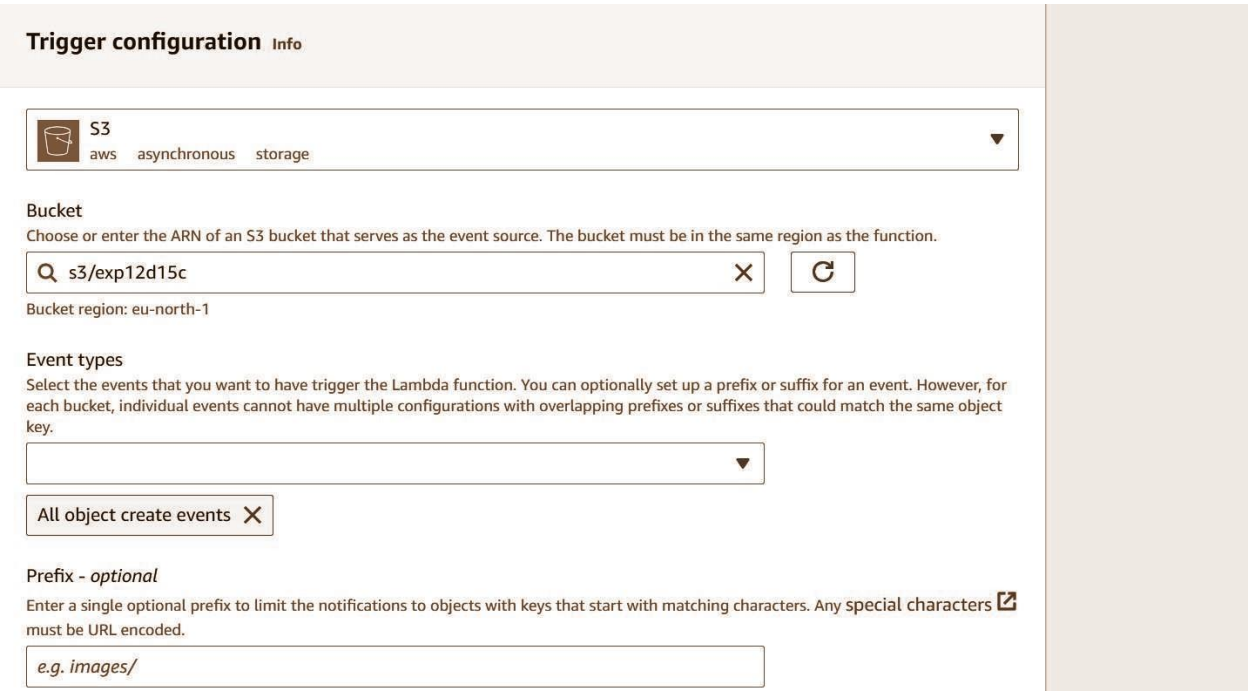


8. Now go back to the aws dashboard and search for lamda function service, Open the function we created in experiment 10. We are going to add this bucket as a trigger to this function

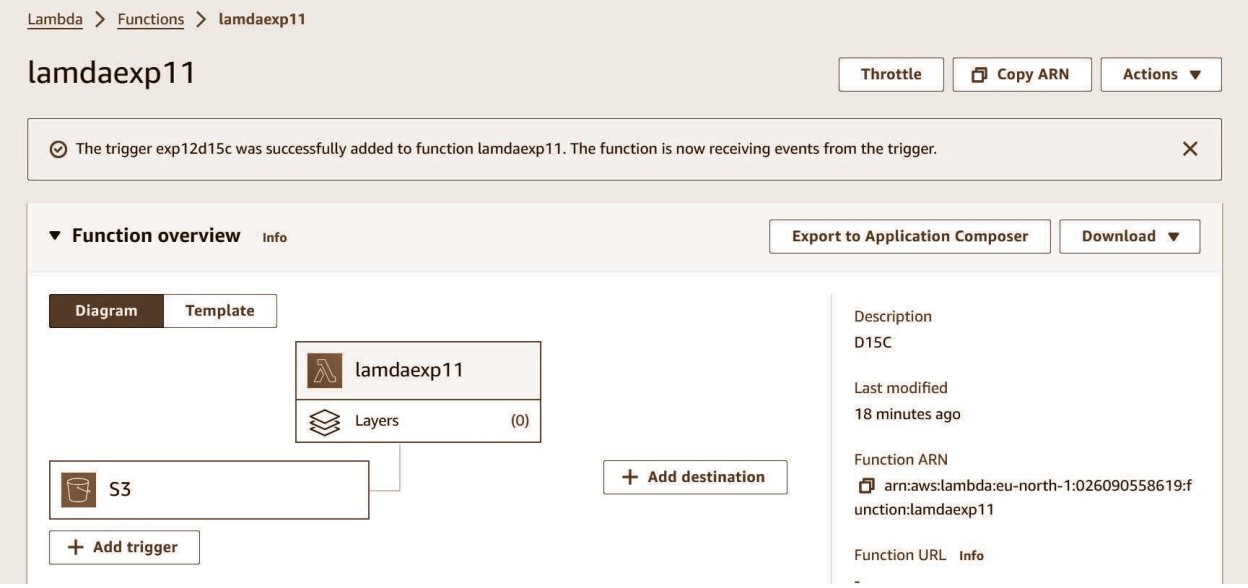
9. On the function overview section of the dashboard you can see the “Add trigger” button. Click on that



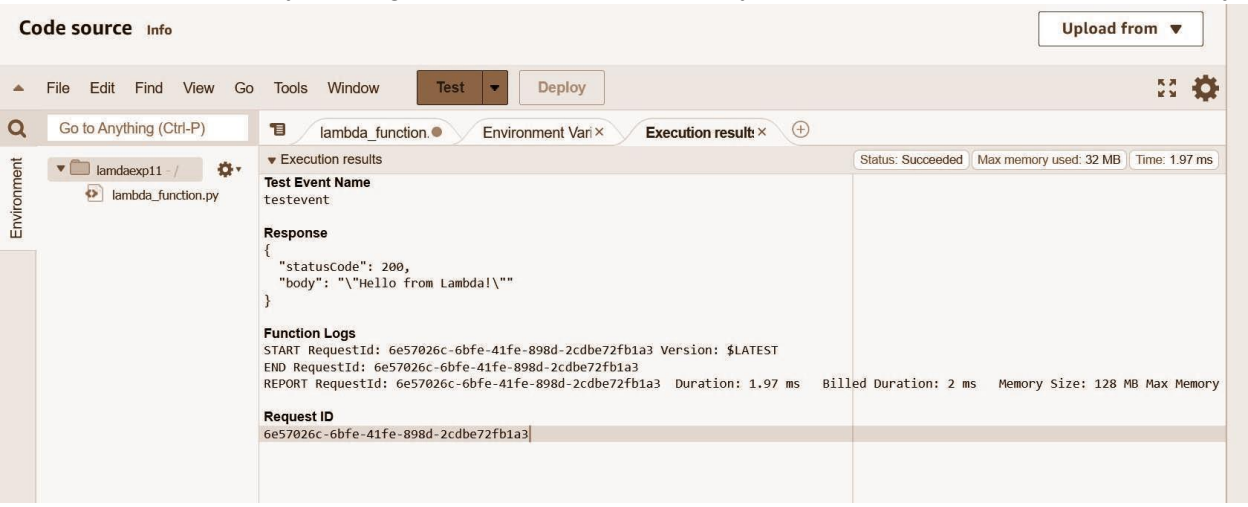
10. It will lead you to the trigger configuration tab; Where you have to select the service and the bucket you created. Add the required configuration information and then save.



11. Here you can see we have the confirmation message as well the the s3 bucket added to our triggers



12. Test the code by clicking on the Test tab ; Here as you can see our code ran successfully



Conclusion: In conclusion, the experiment successfully demonstrated the integration of an S3 bucket with an AWS Lambda function as a trigger. By creating the S3 bucket and configuring it to invoke the Lambda function upon object uploads, we established a seamless workflow for automated processing.