EXPERIMENT 1A

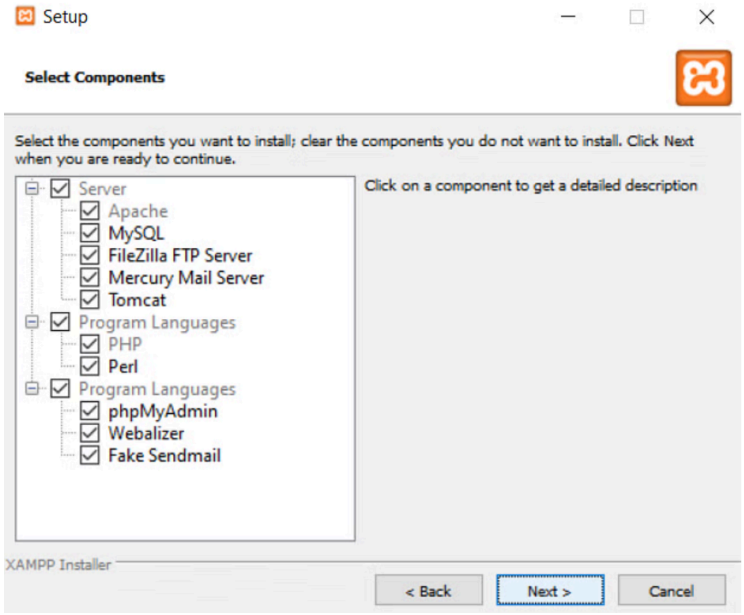
Aim: To develop a website and host it on i) local machine or virtual machine
ii) Amazon S3 Bucket

Static Hosting:

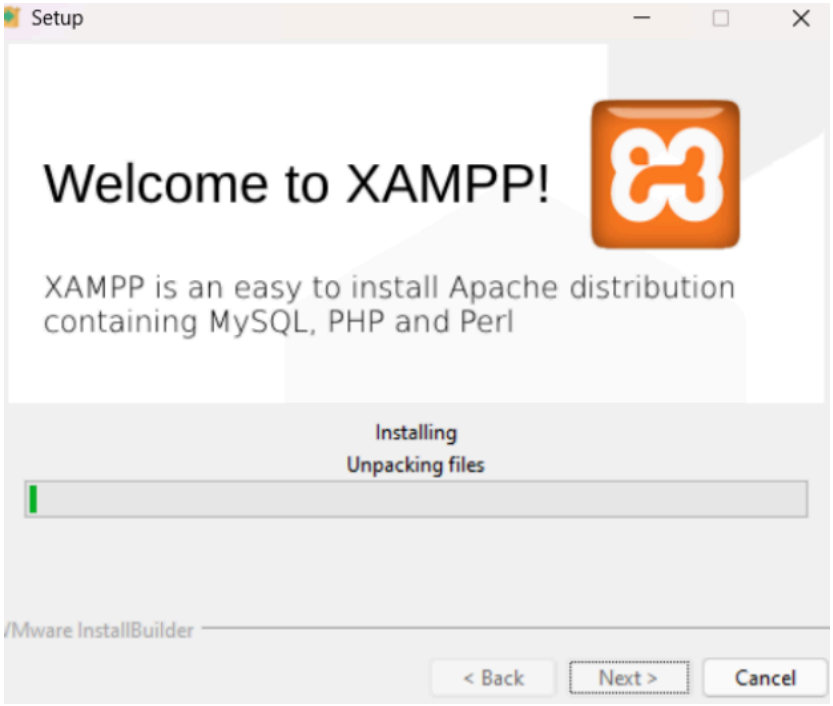
1) On local server (XAMPP)
Step 1: Install XAMPP from https://www.apachefriends.org/ .

1) Select your OS. It will automatically start downloading.

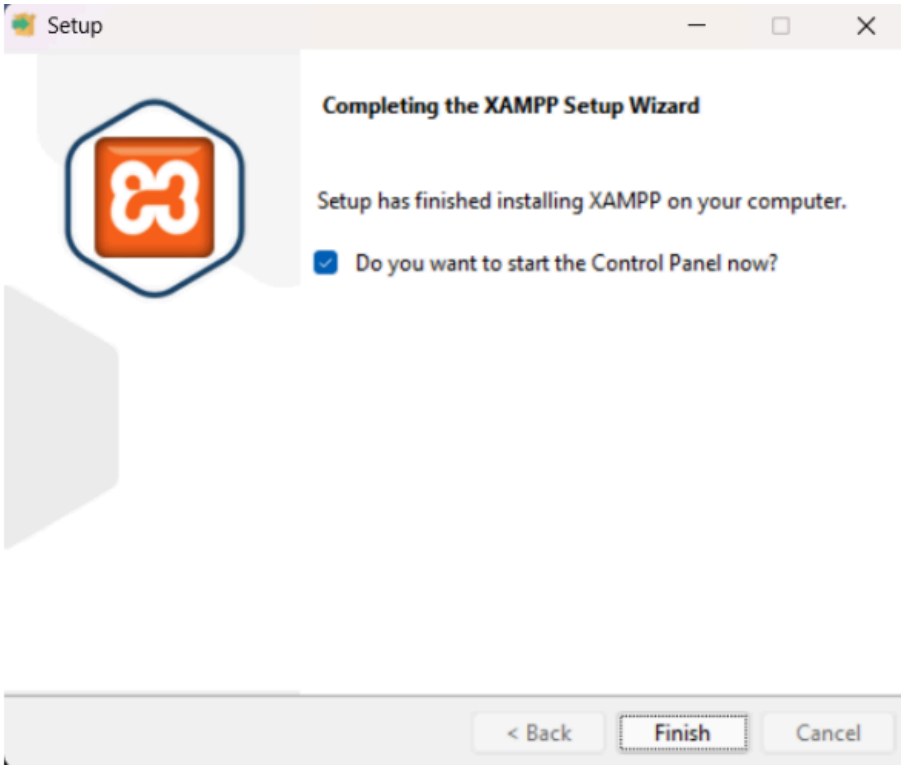
2) Open the setup file. Select all the required components and click next.



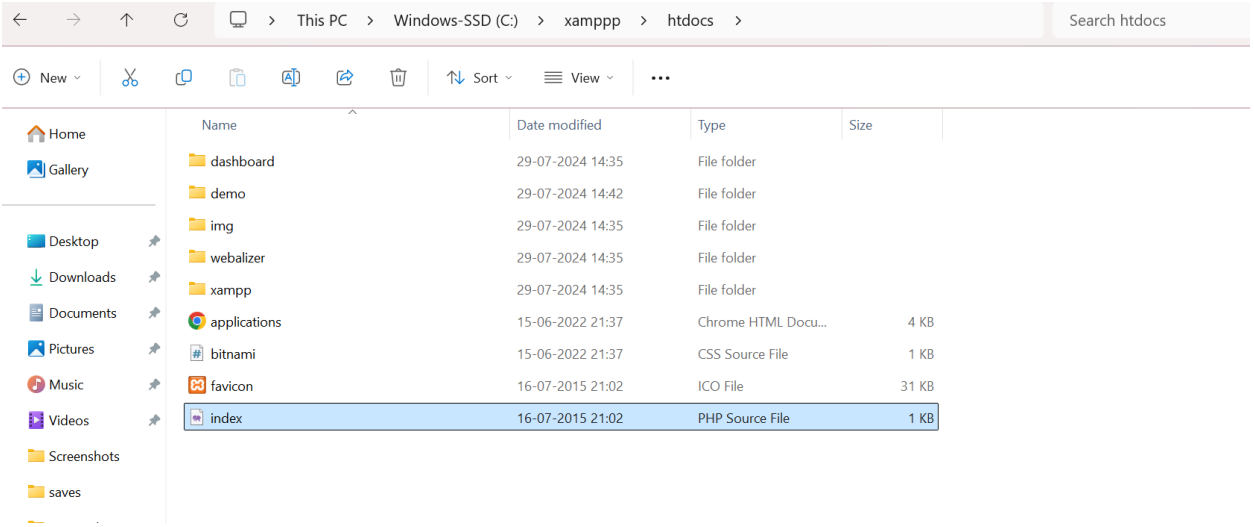
3) Select the language, click next. XAMPP starts to install.



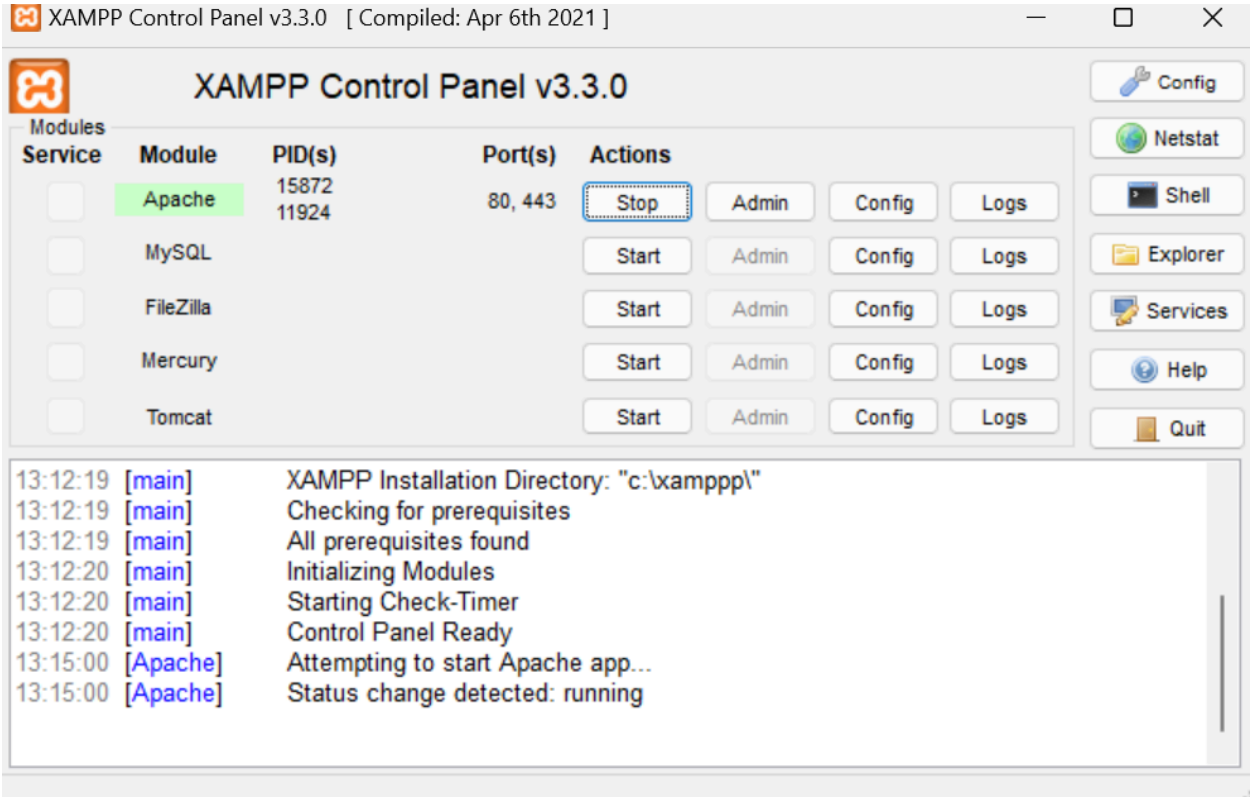
4) The installation is complete. Click Finish.



Step 2: Go to the directory where XAMPP was installed. Go to htdocs folder. Place your folder in this directory.



Step 3: Open XAMPP Control Panel, start the Apache service (Required) and mySQL service (if needed)

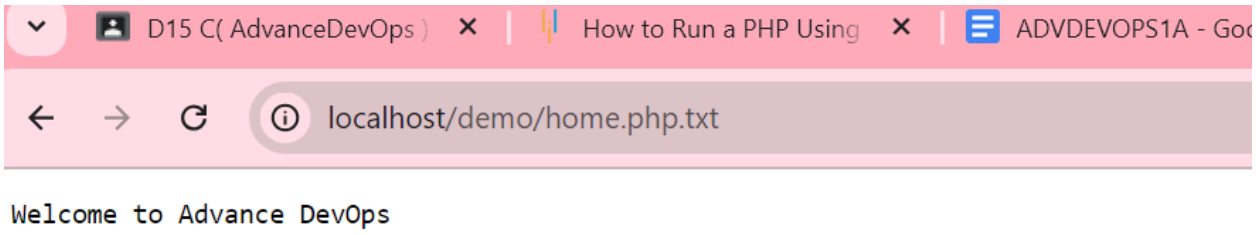


Step 4: Write a php file for your website.

File Edit View

```
<?php
        echo "Welcome to Advance DevOps"
?>
```

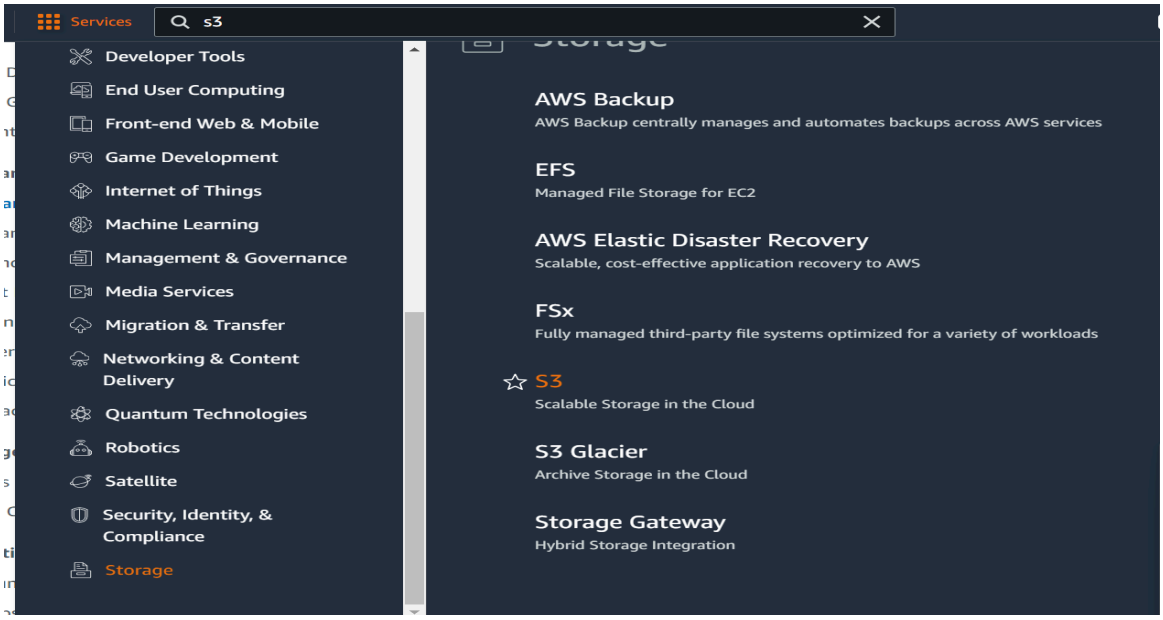
Step 5: Open your web browser. Type localhost/YOUR_FILENAME.php. This will open your website on your browser



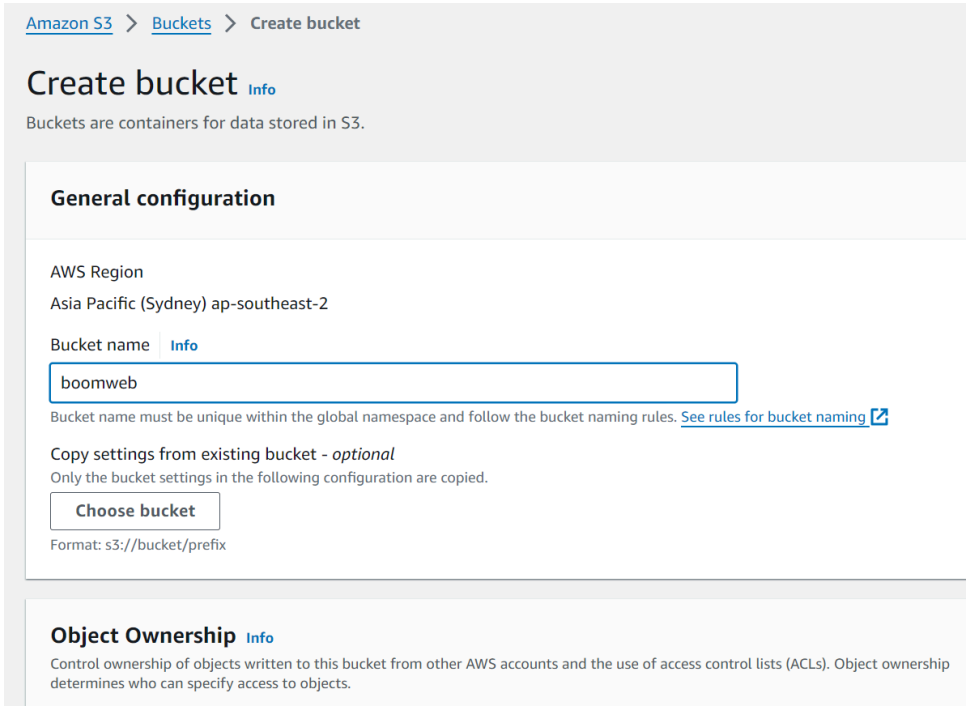
Welcome to Advance DevOps

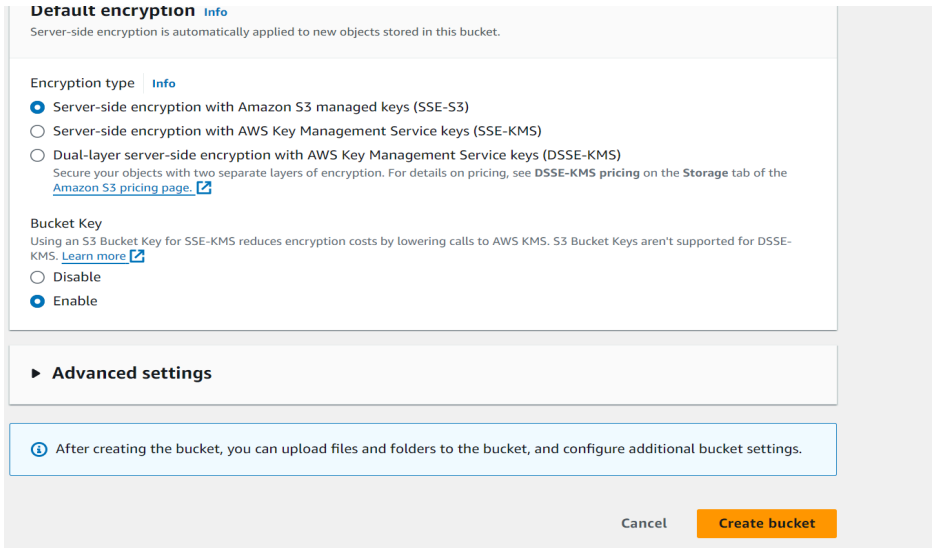
2) AWS S3

Step 1: Login to your AWS account. Go to services and open S3.

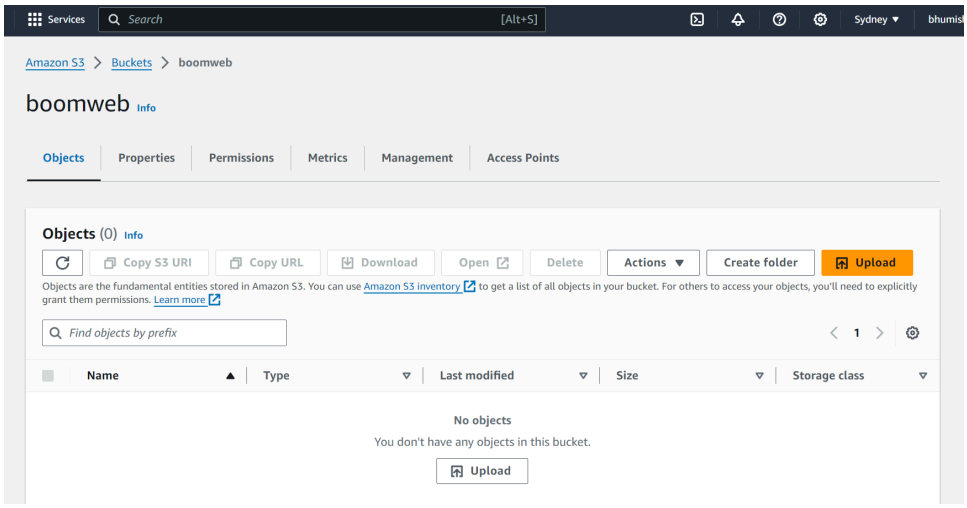


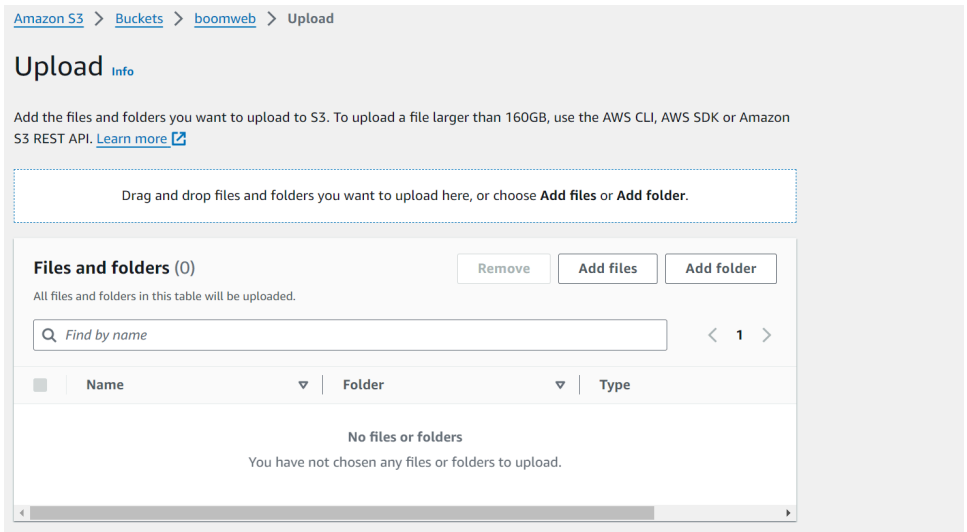
Step 2: Click on Create Bucket. Give a name to your bucket, keeping other options default, scroll down and click on Create Bucket.



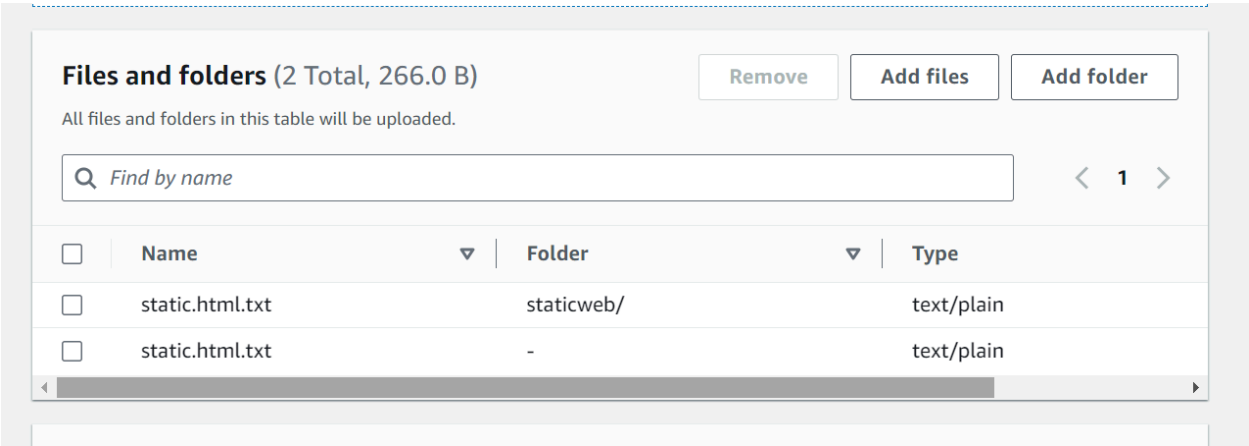


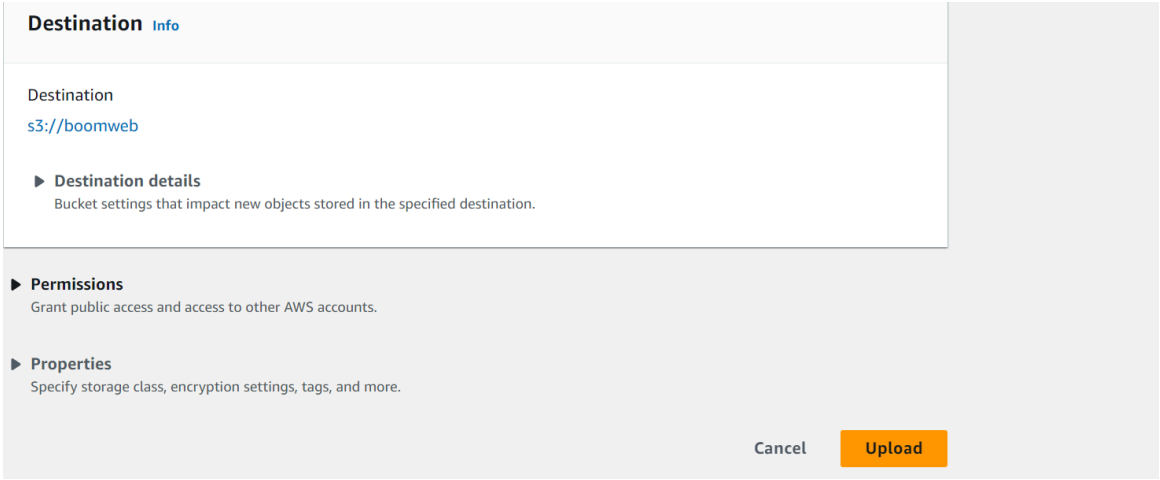
Step 3: Go to the Objects tab and click on upload file.



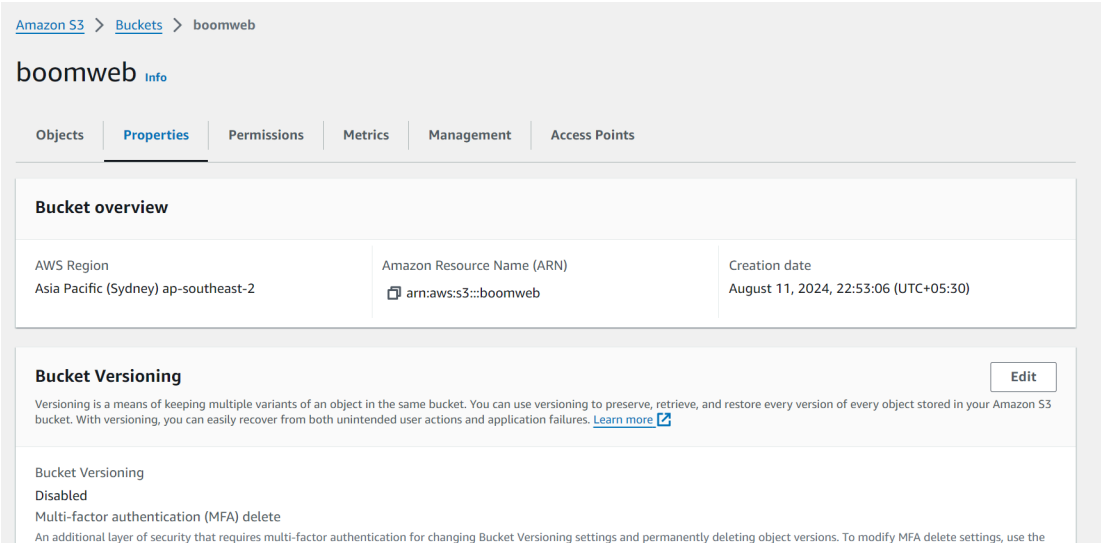


Step 4: Click on Add files. Add all the files you want to upload. Then scroll down and click on Upload.

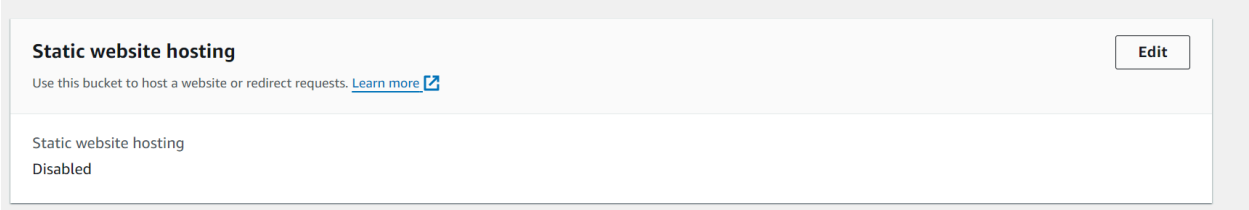




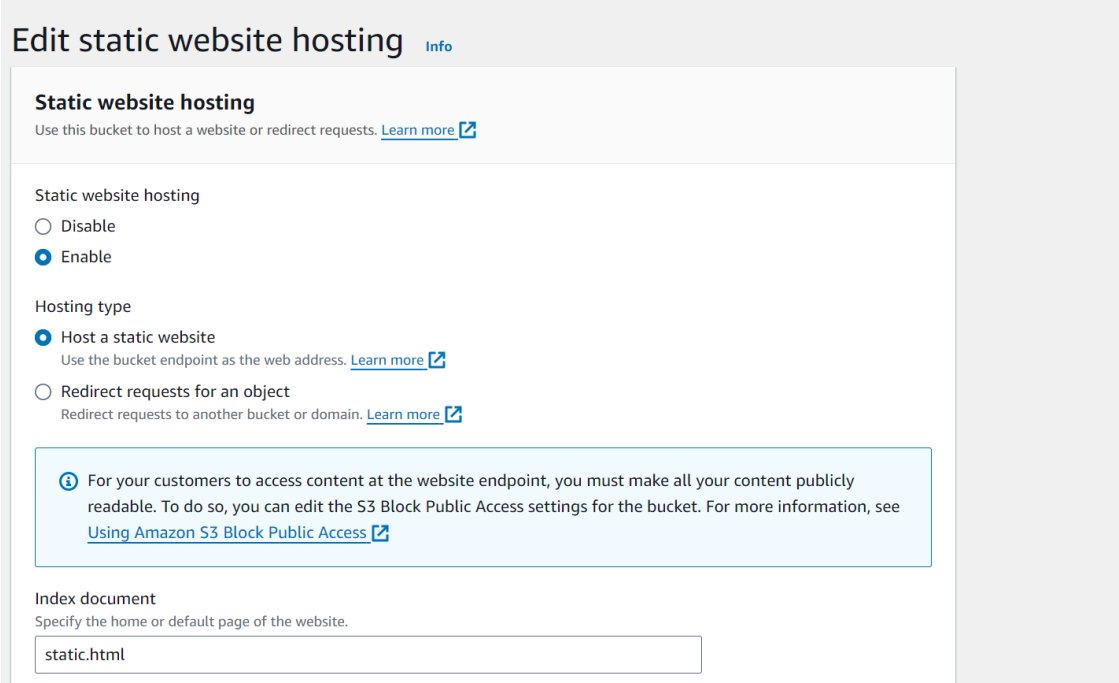
Step 5: This will take you to the Objects screen. Switch to Properties and scroll down to Static Website Hosting. There you would find the link (Bucket website endpoint) to your website.



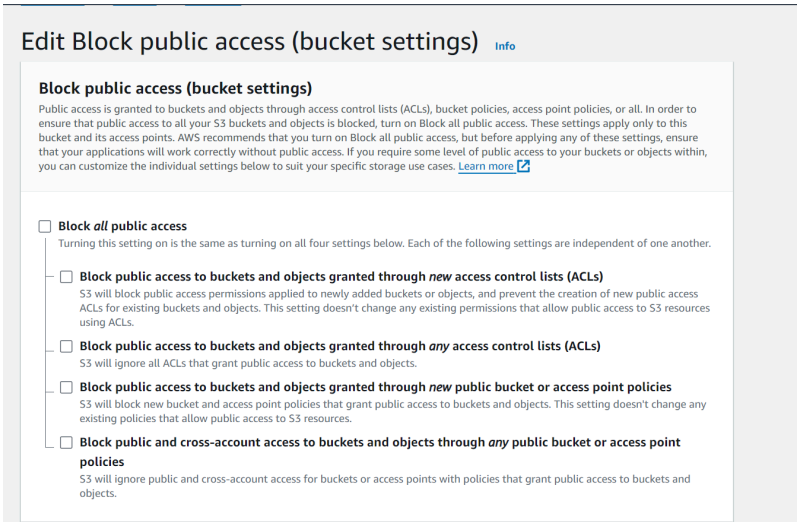
Step 6: Scroll down till you find Static website hosting, click on edit.



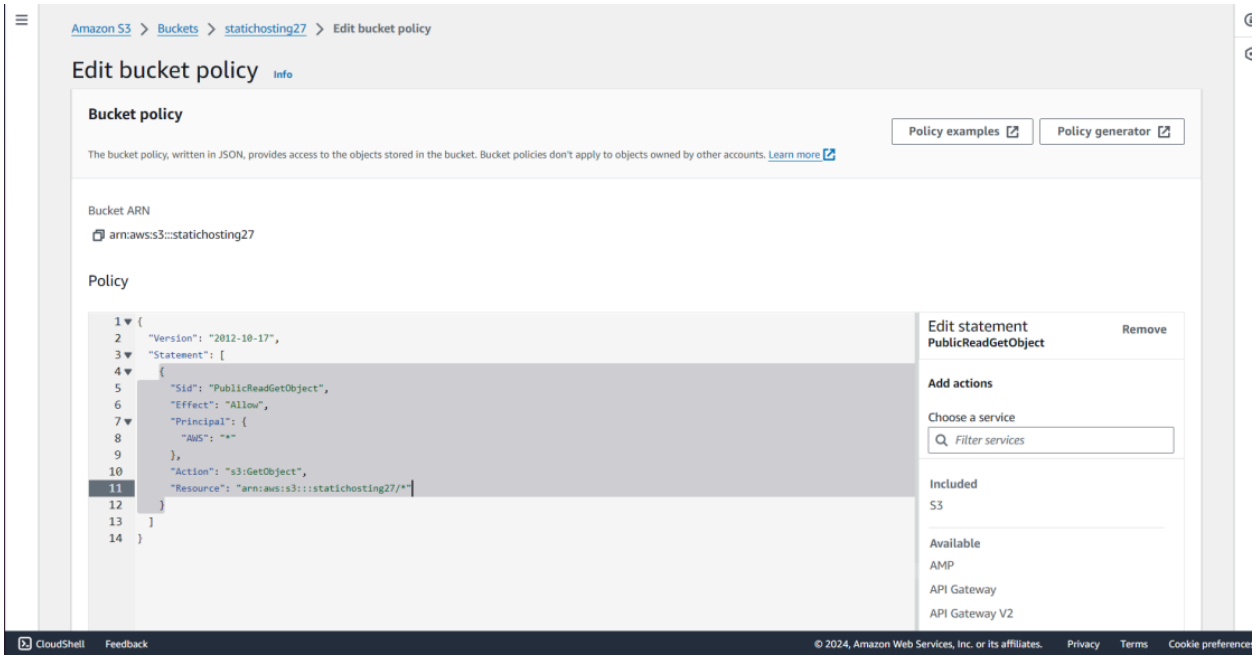
Step 7: Enable static website hosting, in Index document, write the name of your document. Save your changes.



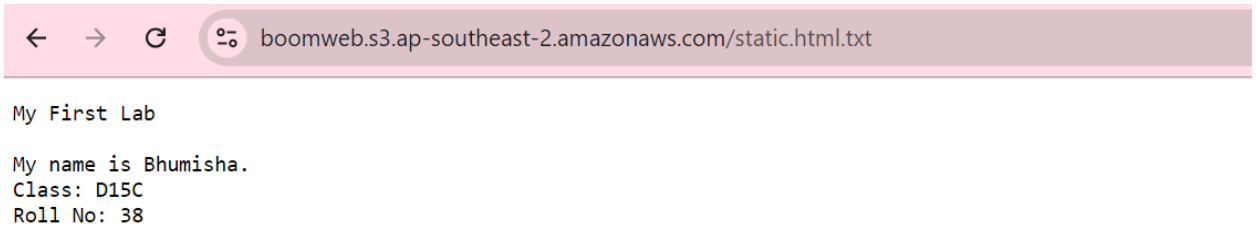
Step 8: Uncheck the Block all public access checkbox and click on save changes.



Step 9: Scroll down to bucket policy and click edit.



Step 10:You can access your website now.



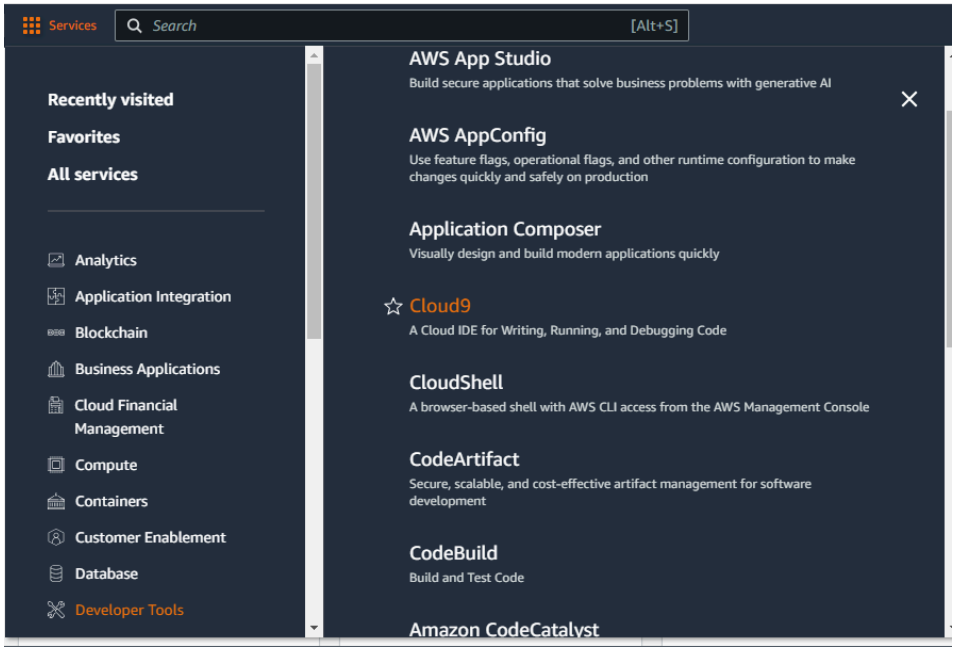
EXPERIMENT 1(B)

Aim: To understand the benefits of Cloud Infrastructure and Setup AWS Cloud9 IDE, Launch AWS Cloud9 IDE and Perform Collaboration Demonstration.

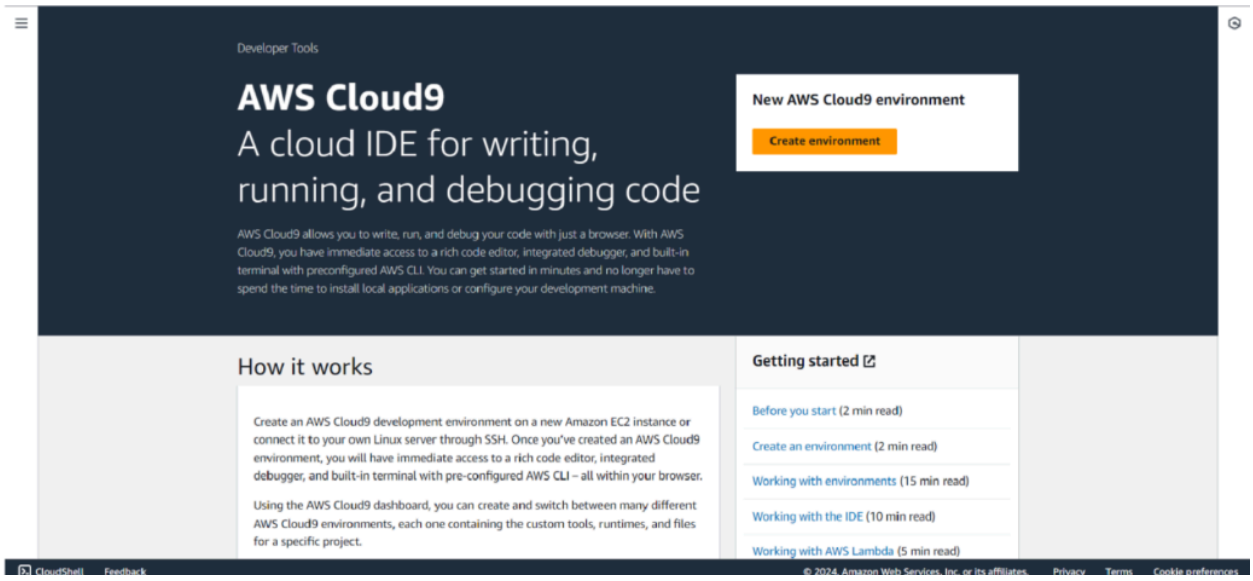
Steps:

Step 1: Set up a Cloud9 environment.

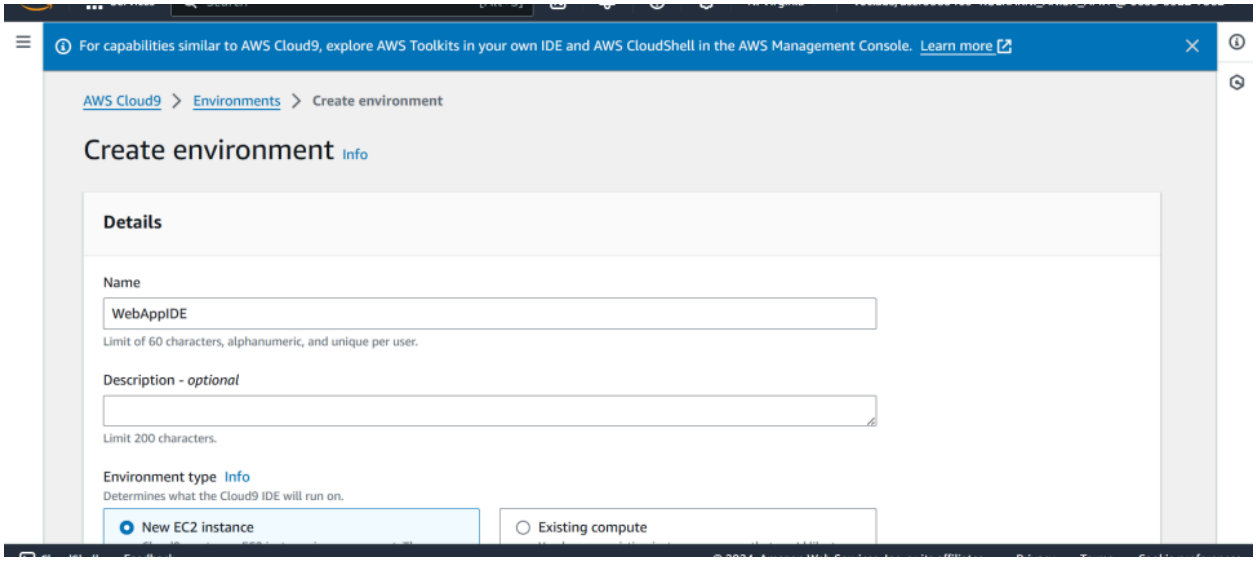
1) Search Cloud9 in the services tab and open it



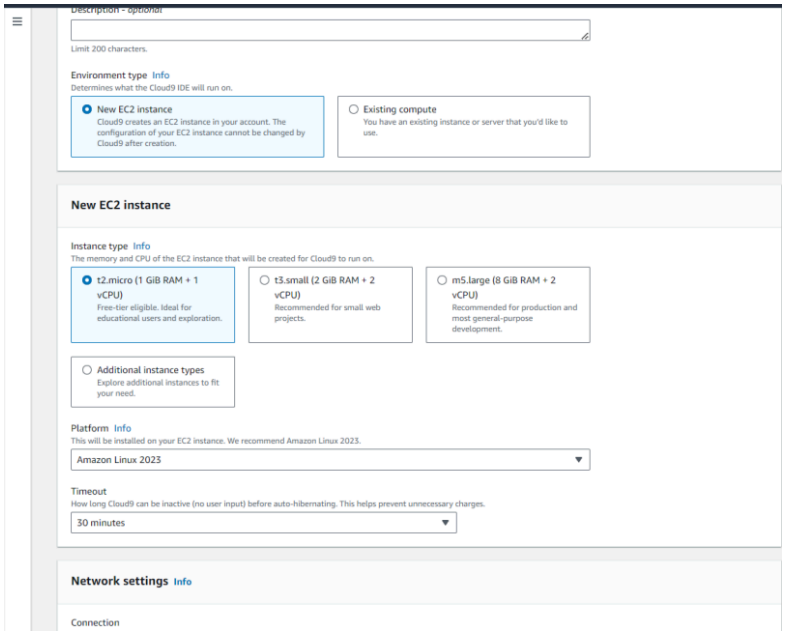
2) Click on Create Environment.



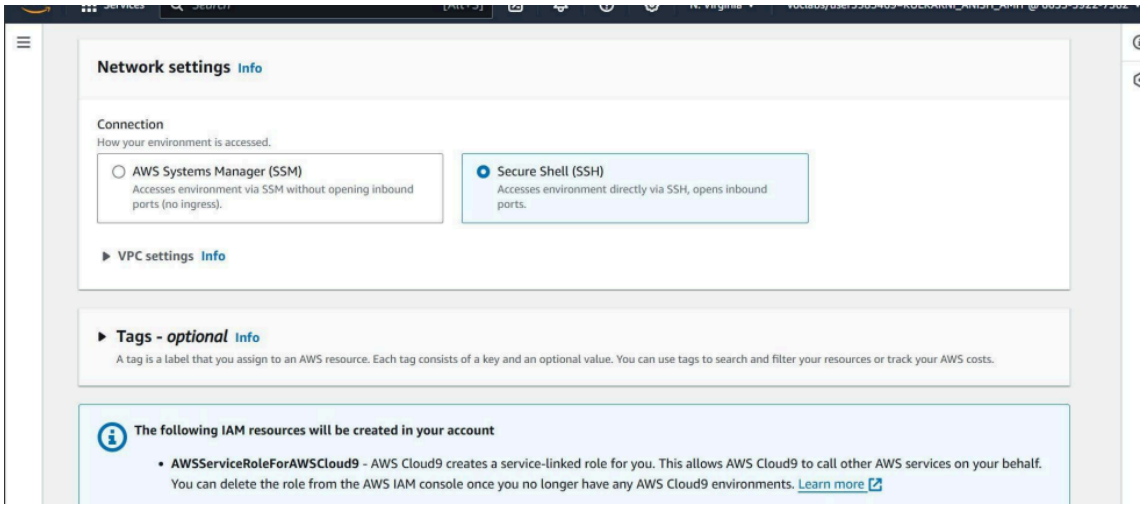
3) Give a name to your Cloud9 Environment. You can add a description if needed.



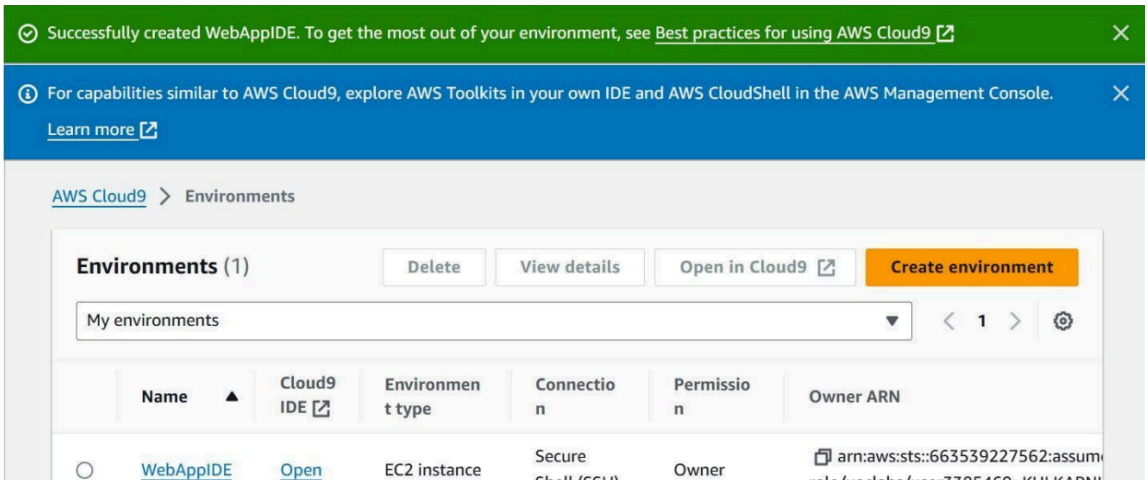
4) Select the option new EC2 instance if you do not have one ready for the environment. Give the specifications of that EC2 instance ahead.



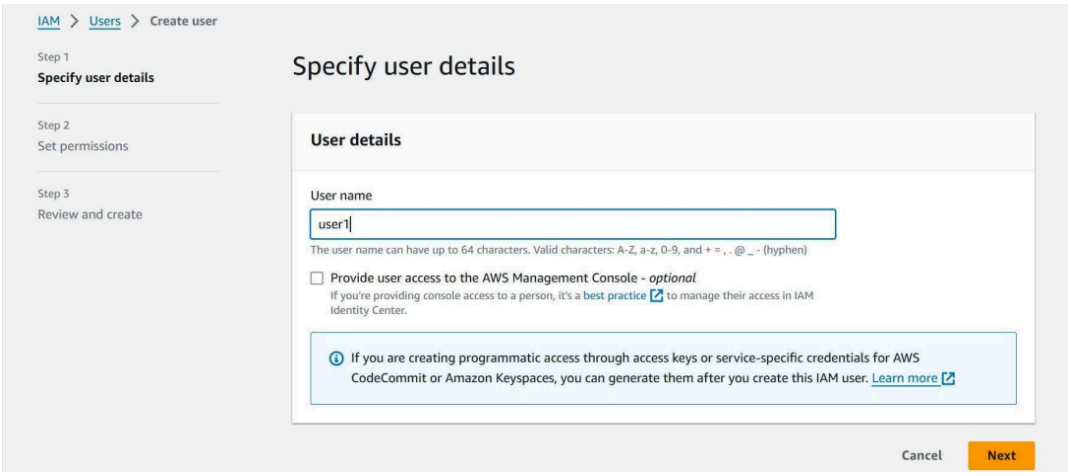
5) On the AWS Academy account, if we select AWS System Manager (SSM) in Network settings, it gives an error as the account does not have permissions to use the setting. So we select Secure Shell (SSH). After that click on Create.



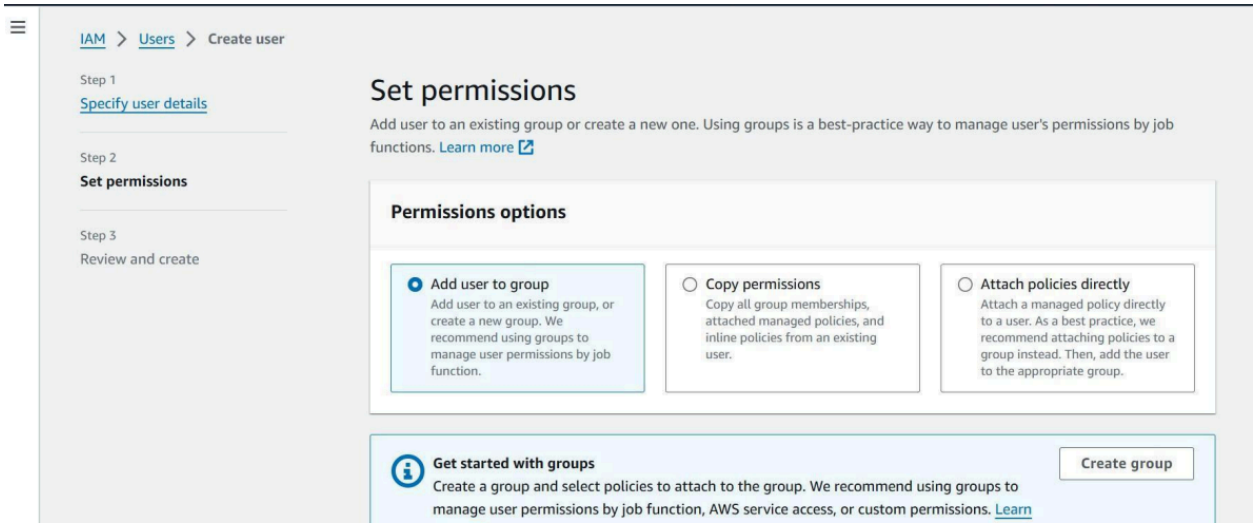
6) The environment has been created.

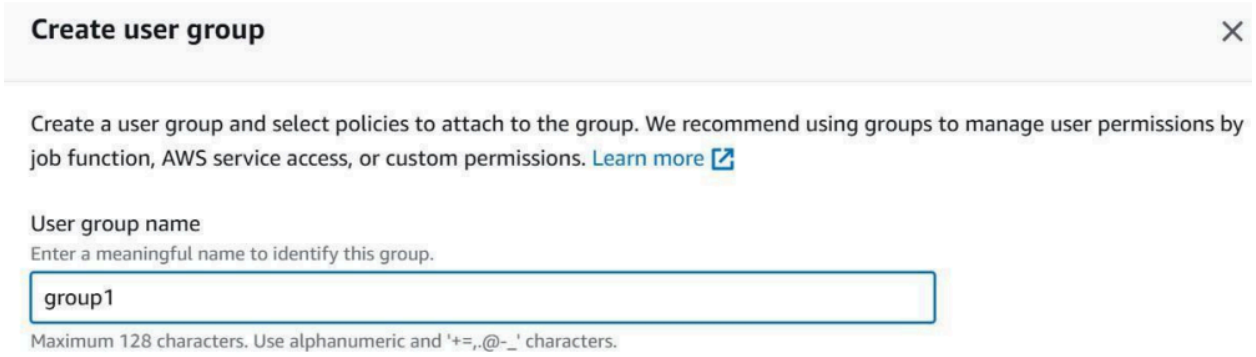


7) Search IAM on the services search bar and open it. Click on Create User. Give a username to your user and click Next.

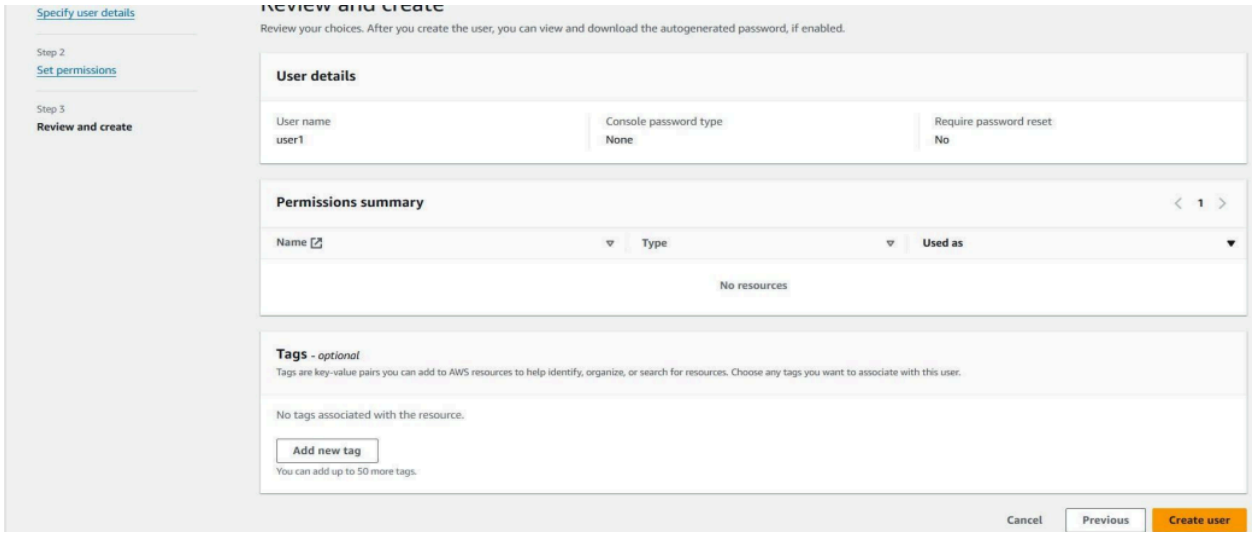


8) Select add User to Group. If there are no user groups on your accounts, you will have to create one. Click on Create Group.

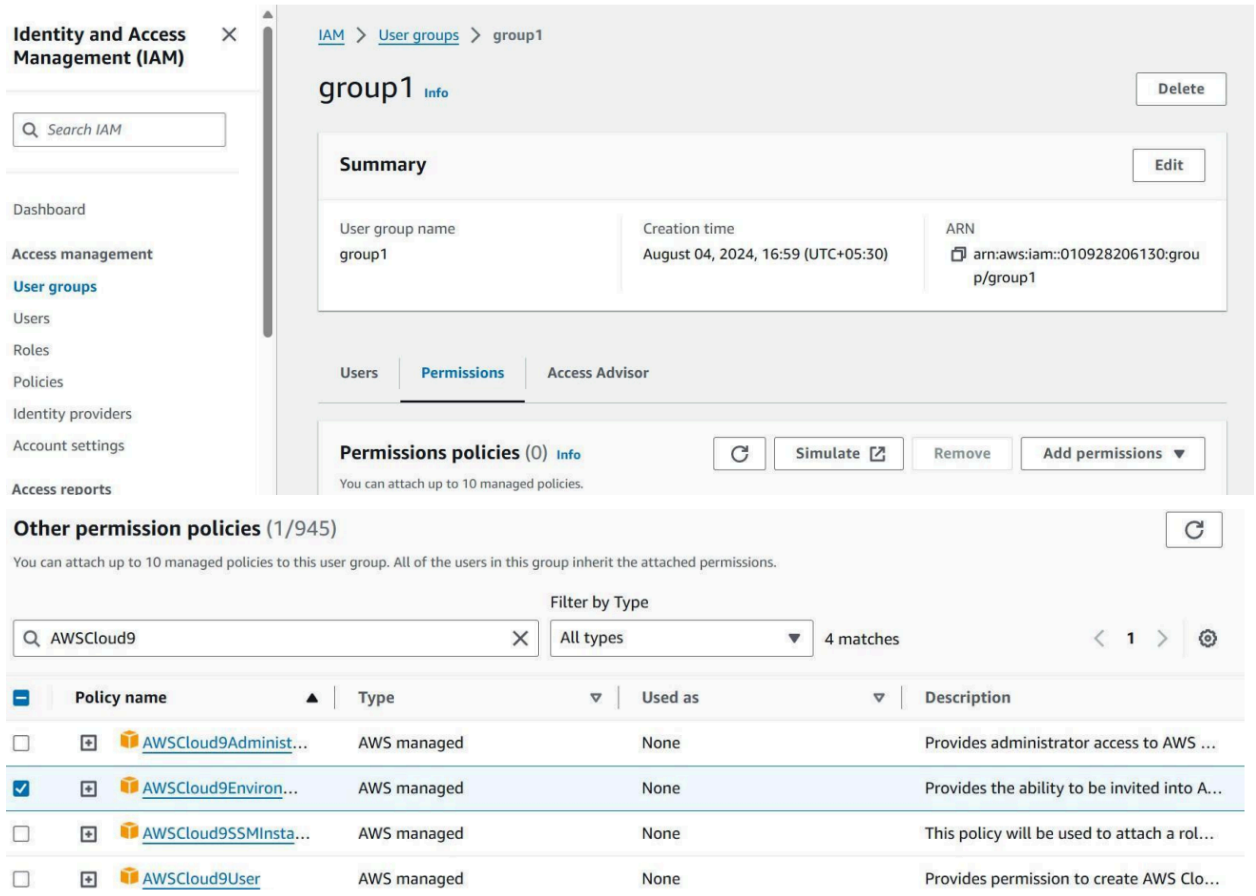




9) Review all the Information, then click on Create user.



10) Go to permissions and click on Add permissions. Click on Attach Policies. Search for AWSCloud9EnvironmentMember, select it and click on Attach policies.

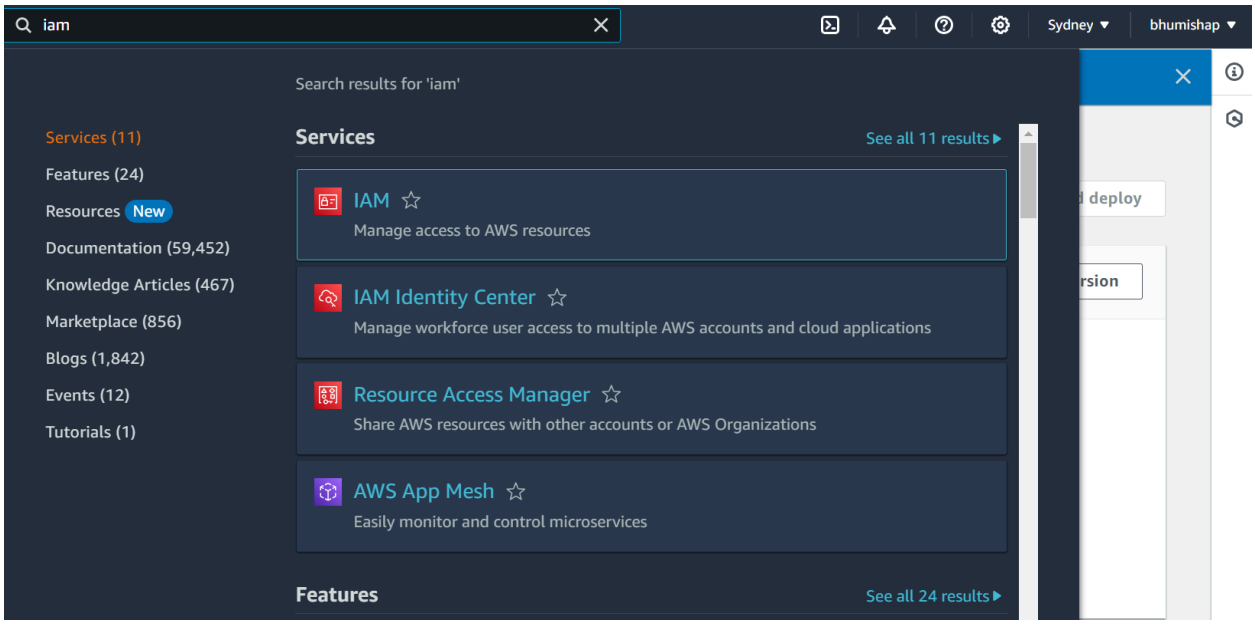


Experiment No. 2

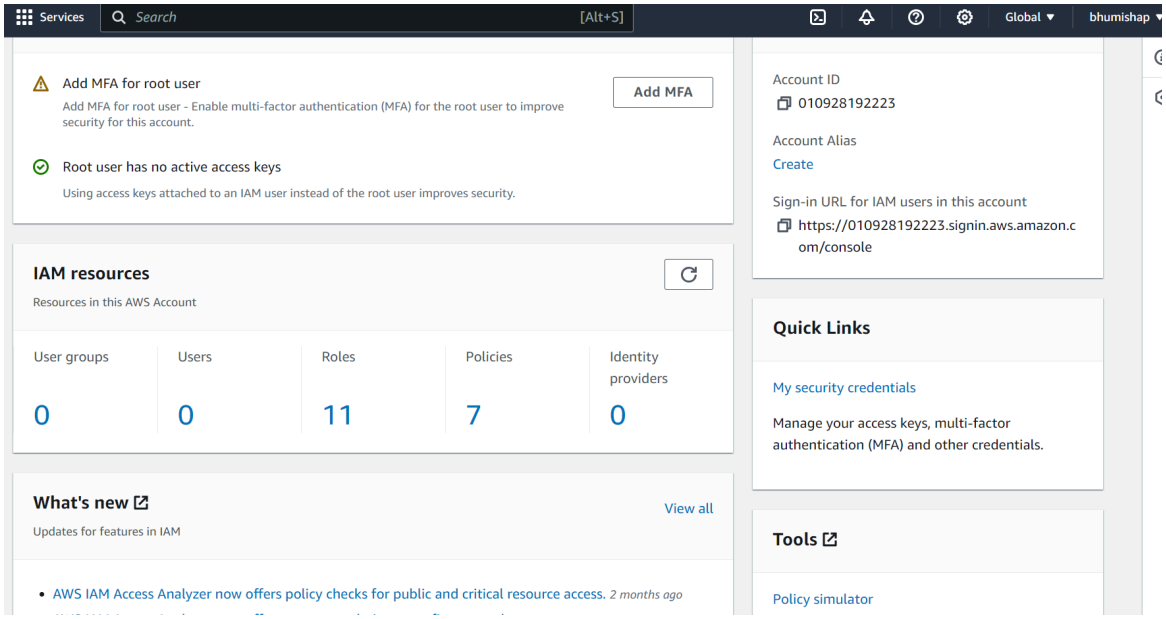
Aim: To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.

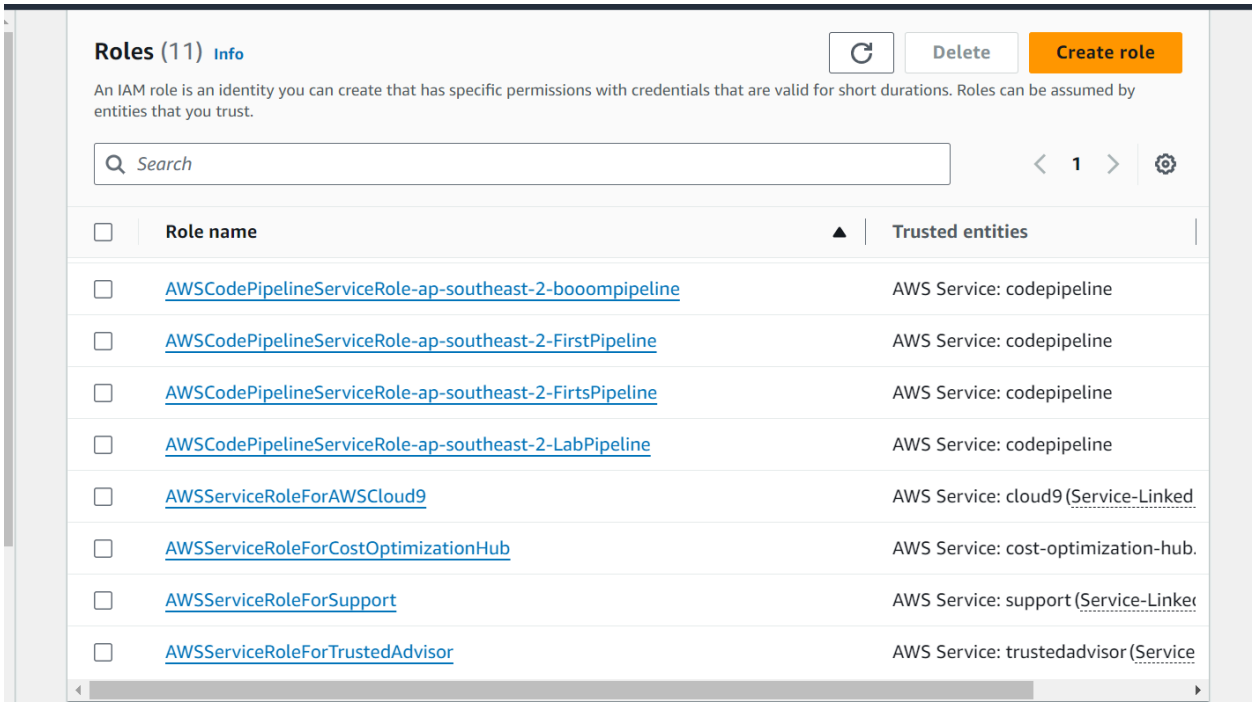
Step1:- Creation of role:-

1. Login to your AWS account and search for IAM

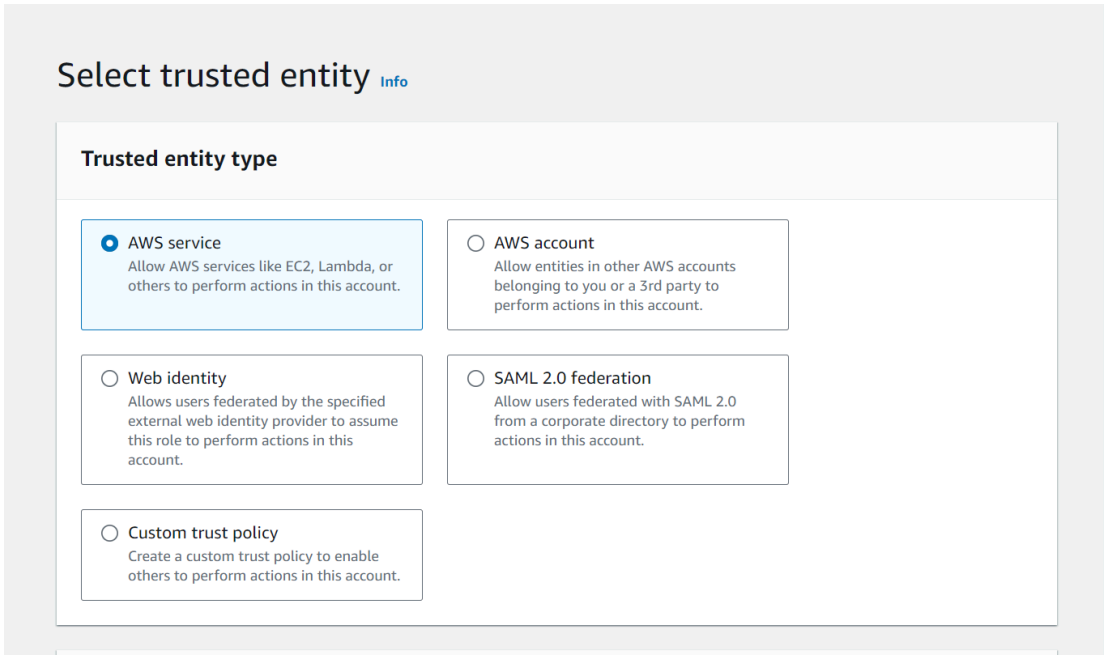


2. Then go into the role section and click on create role.

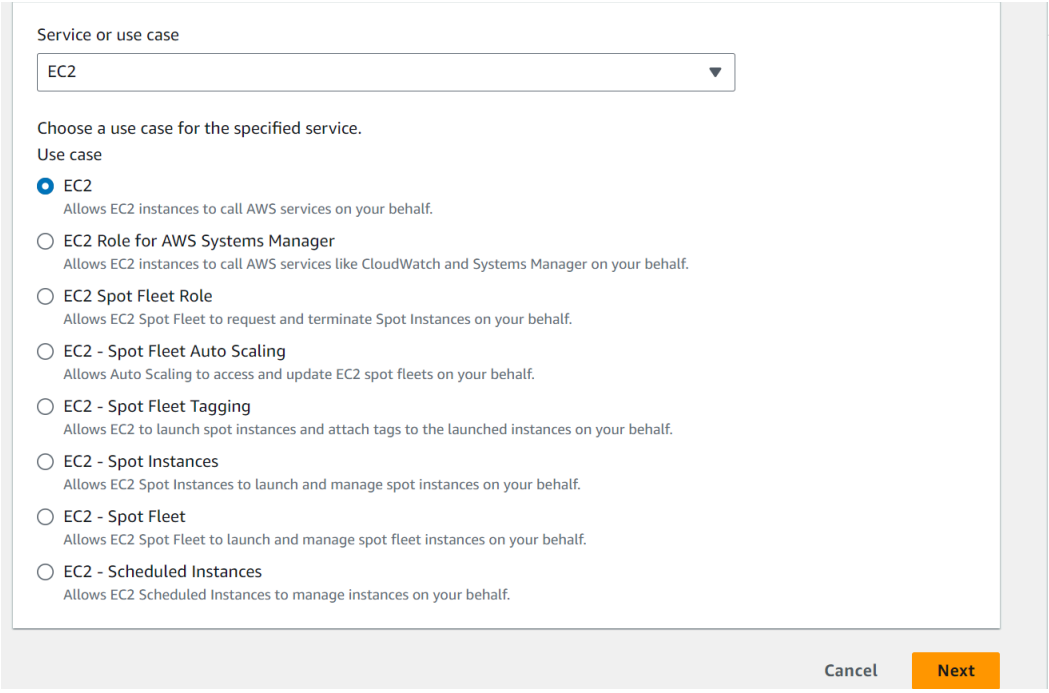




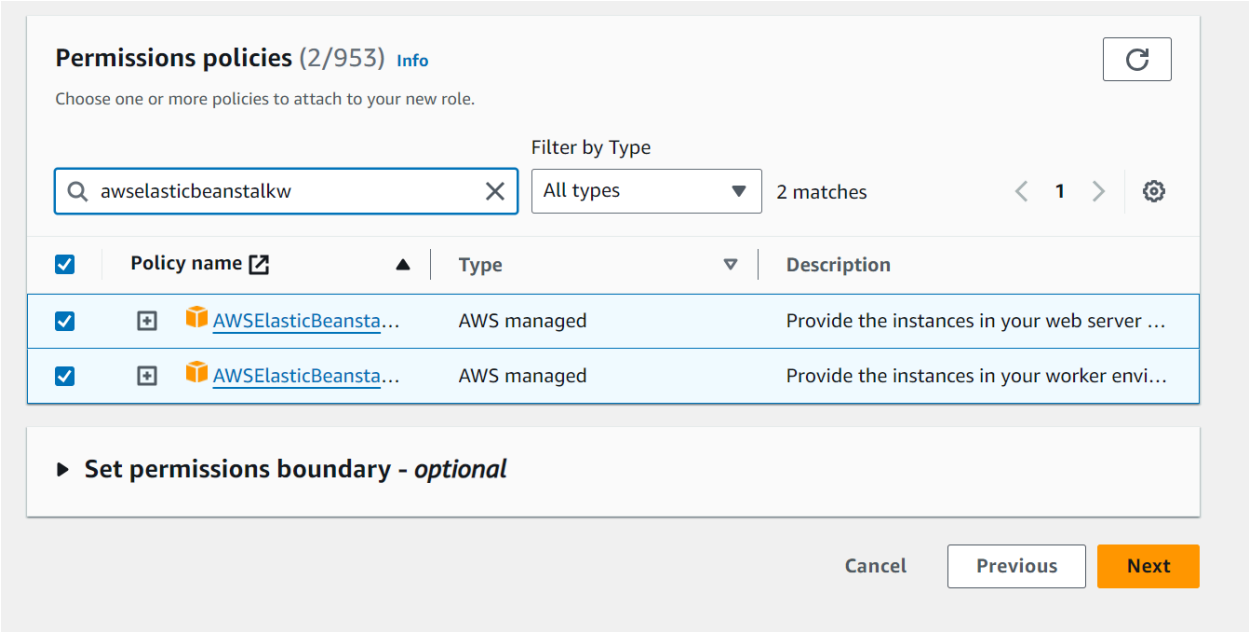
3. Then select a trusted entity as AWS service.



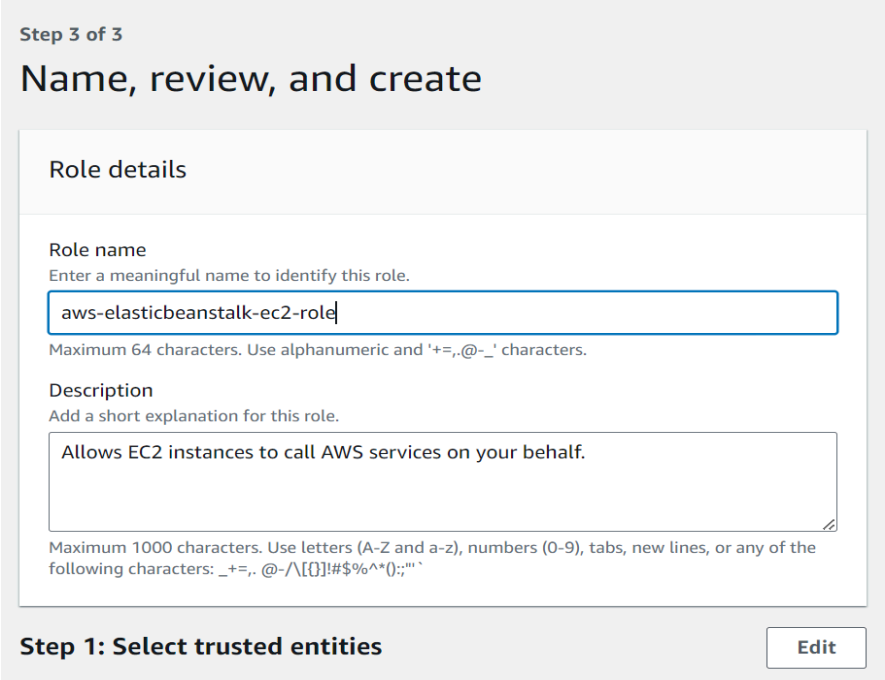
4. Select use case as EC2.



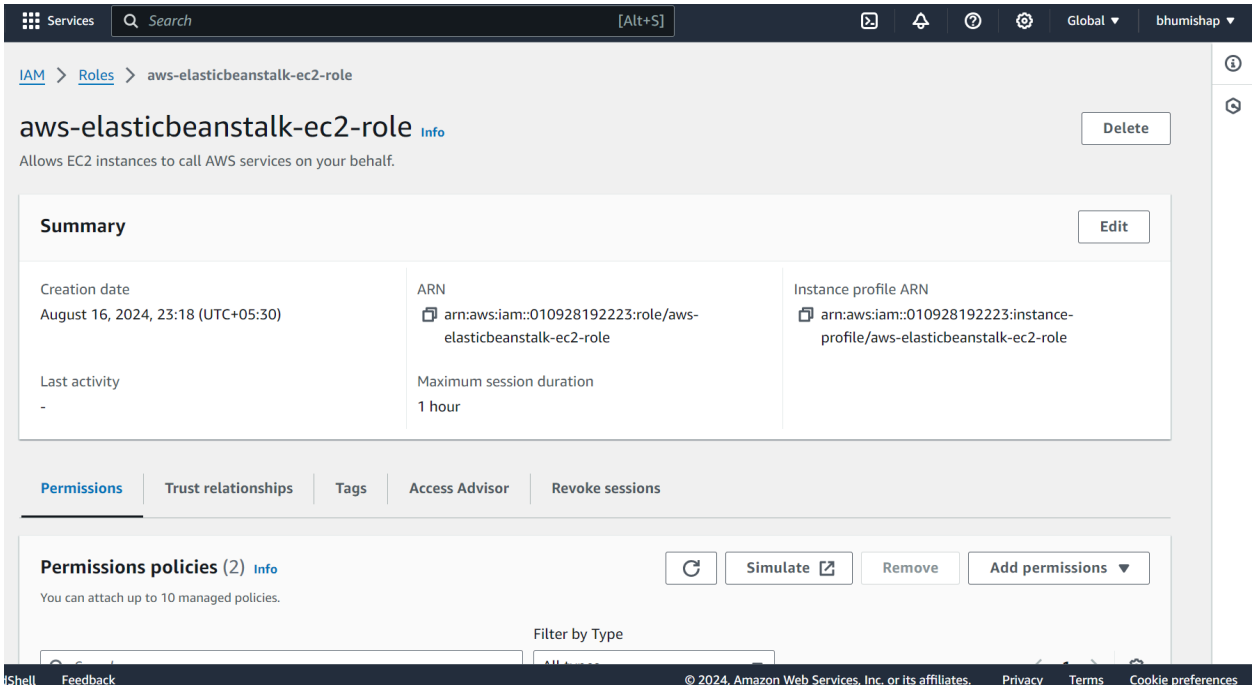
5. Select permissions as AWS Elastic Beanstalk Web Tier.and AWS elastic Beanstalk worker tier.



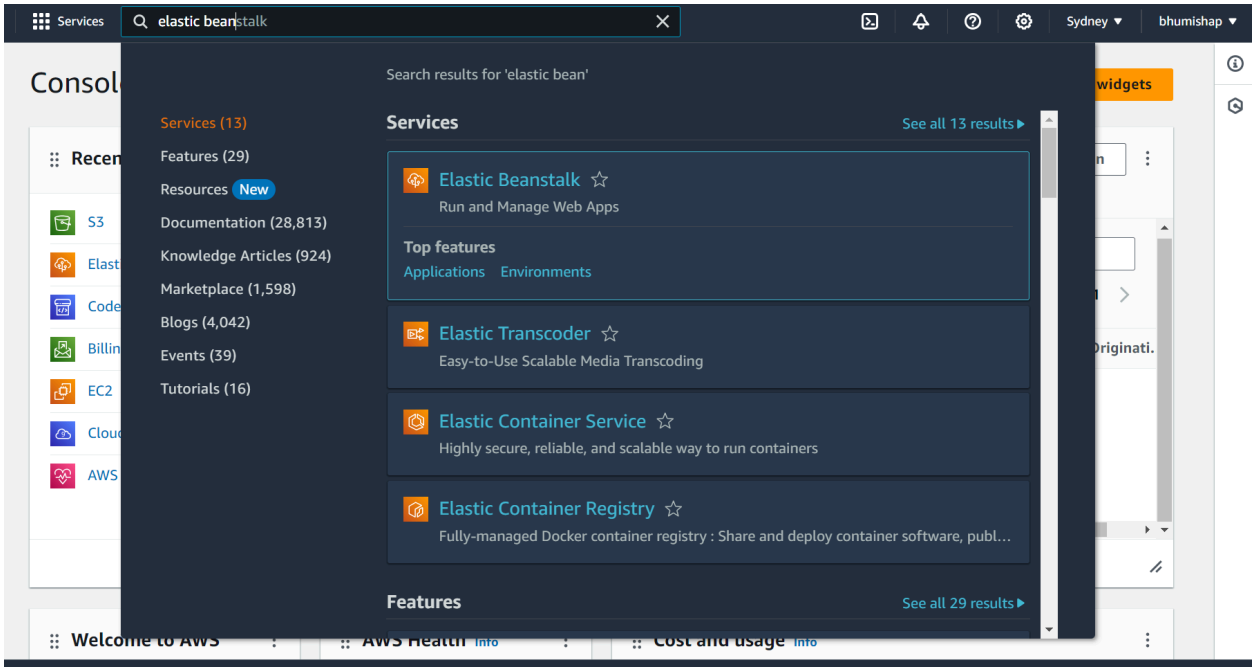
6. Give a name to Role. Here I have given my role name as aws -elasticbenstalk -ec2 role.



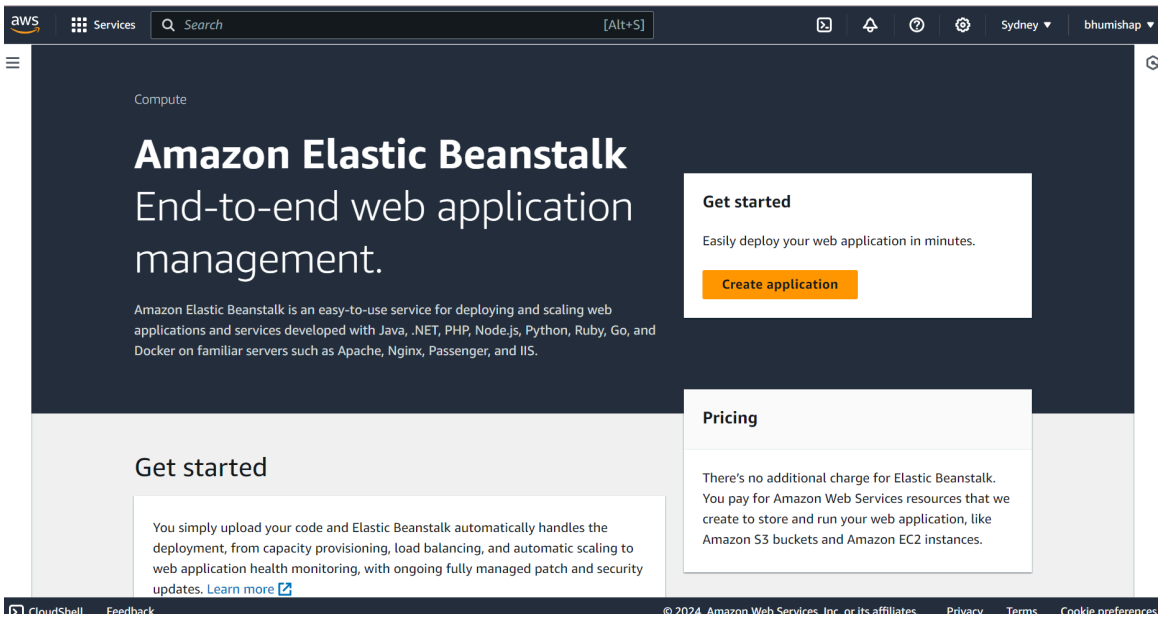
7. Then the role gets created.



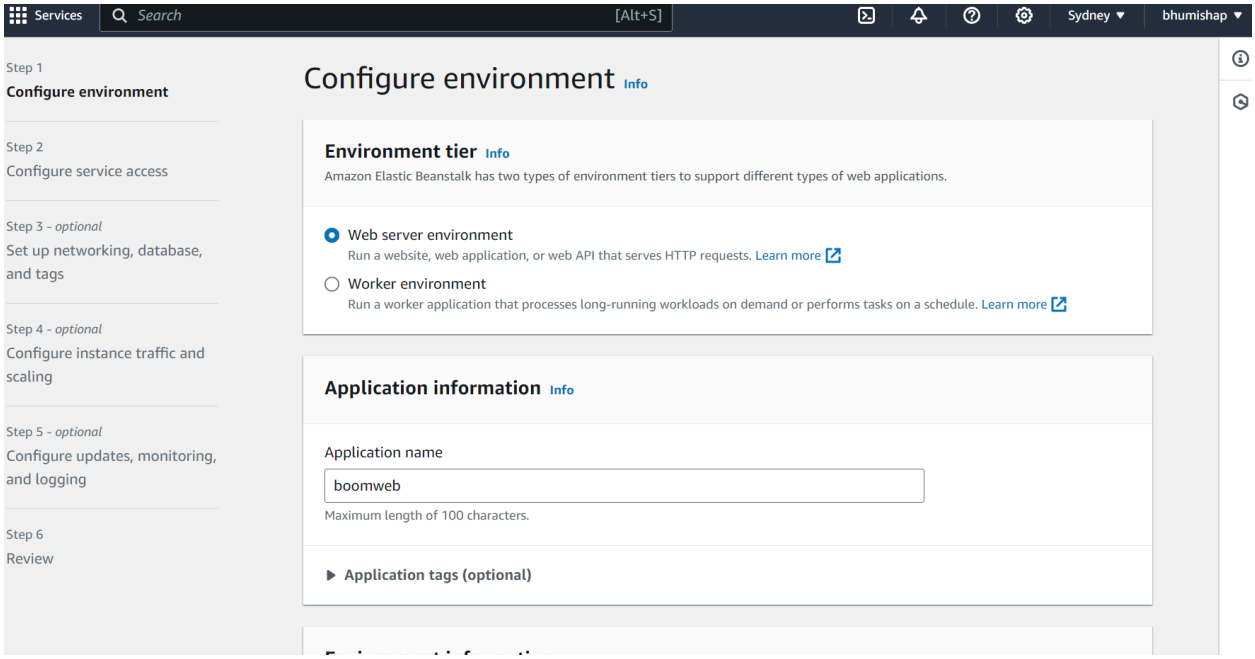
Step 2: Search for Elastic Beanstalk in the searchbar near services

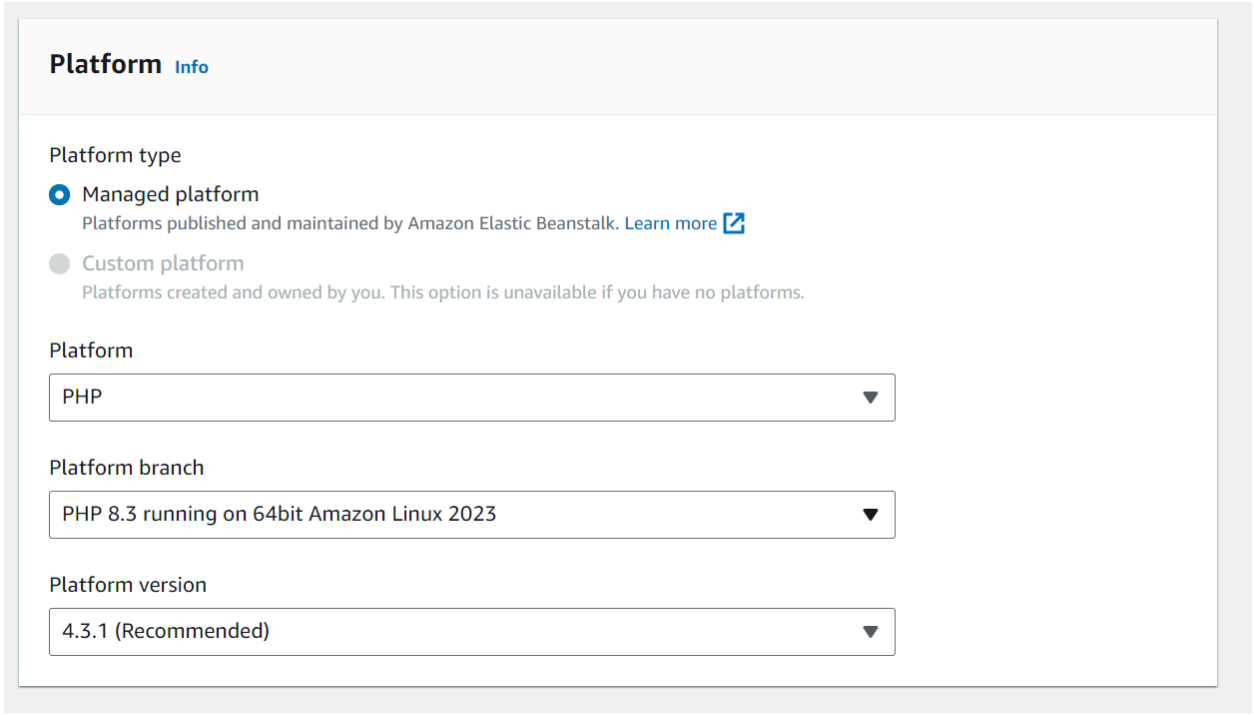


Step 3: Go to Elastic Beanstalk and click on Create Application

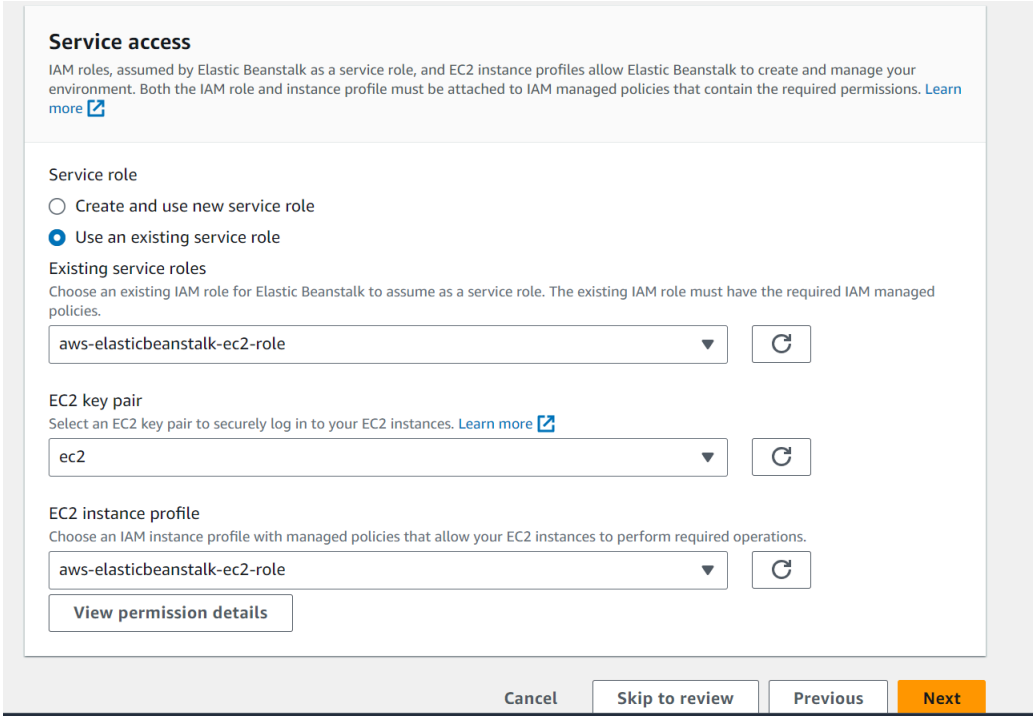


Step 4: Enter the name of your application. Scroll down and in the platform, select platform as PHP. Keep the application code as Sample Application. Set the instance to a single instance. Click on NEXT

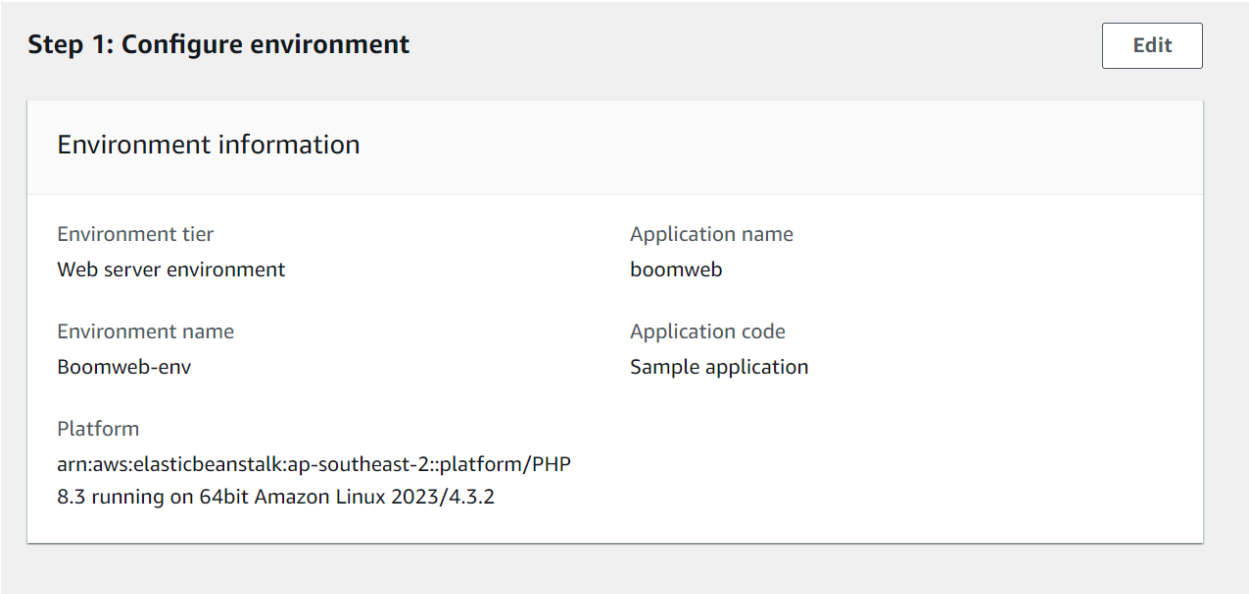




Step 5: Use an existing service role and choose whatever service role is available on your account.



Step 6: Review the settings that you have set up for your application and submit your application.



Step 2: Configure service access

Edit

Service access Info

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

Service role	EC2 key pair	EC2 instance profile
arn:aws:iam::010928192223:role/aws-elasticbeanstalk-ec2-role	ec2	aws-elasticbeanstalk-ec2-role

Step 3: Set up networking, database, and tags

Edit

Networking, database, and tags Info

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

No options configured

Tags

Platform software

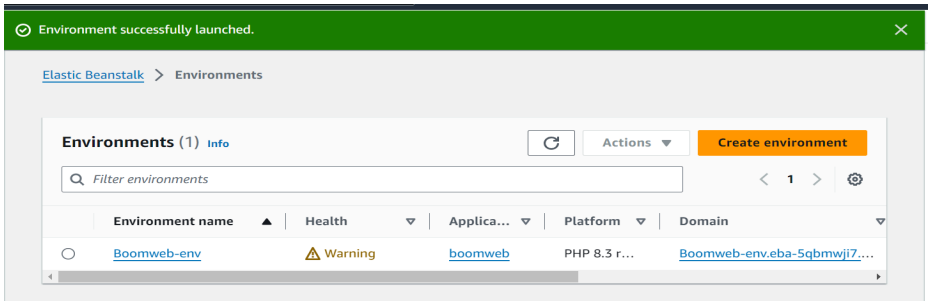
Lifecycle	Log streaming	Allow URL fopen
false	Deactivated	On
Display errors	**Document root**	**Max execution time**
Off	–	60
Memory limit	**Zlib output compression**	**Proxy server**
256M	Off	nginx
Logs retention	**Rotate logs**	**Update level**
7	Deactivated	minor

X-Ray enabled

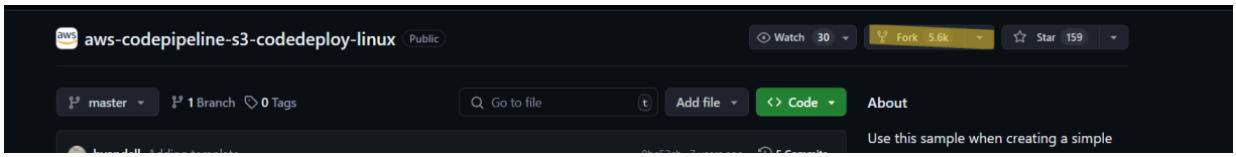
Deactivated

Environment properties

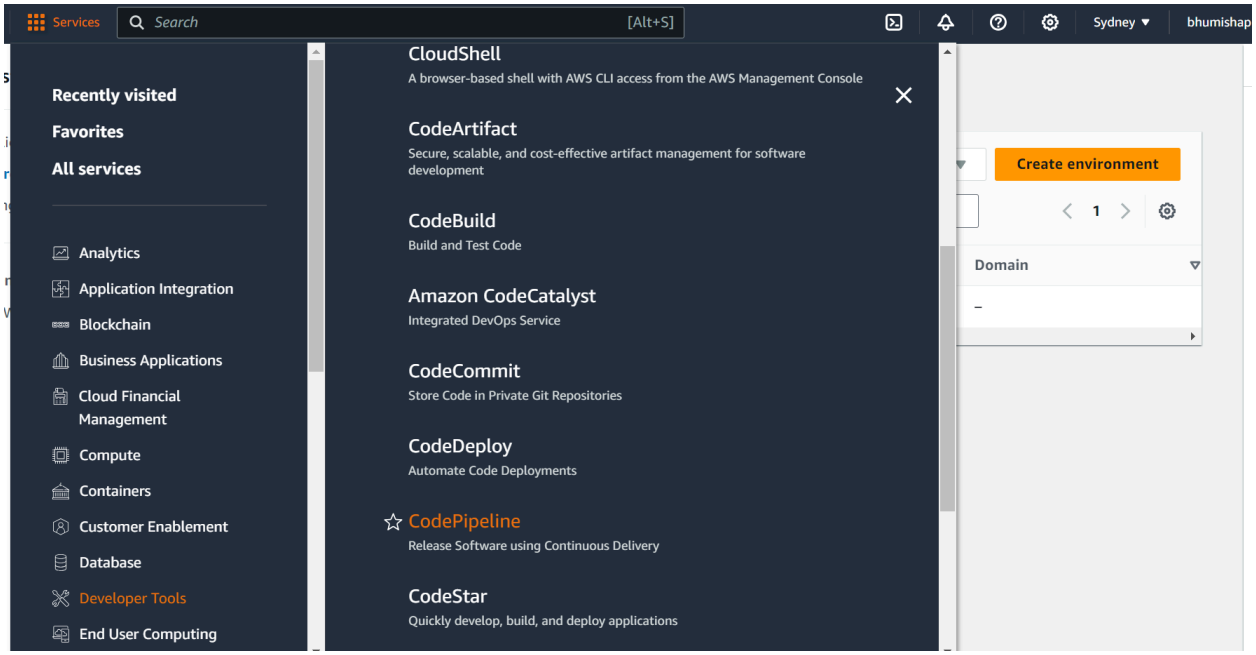
Key	Value

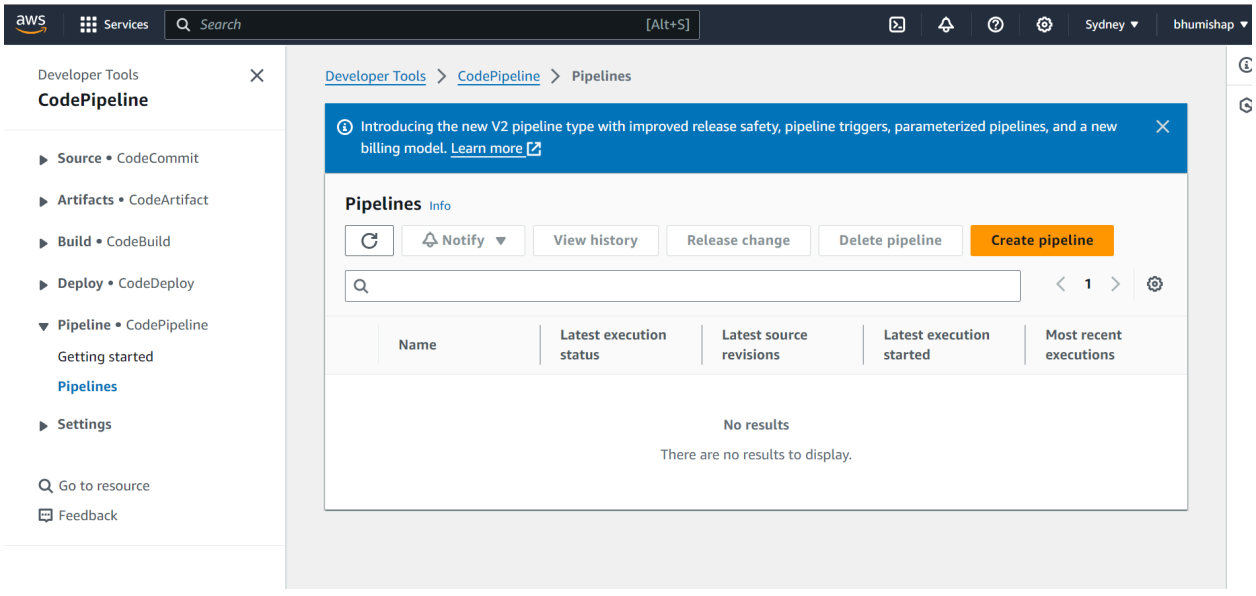


Step 7: Go to the github link below. This is a github with a sample code for deploying a file on AWS CodePipeline. Fork this repository into your personal github.
https://github.com/aws-samples/aws-codepipeline-s3-codedeploy-linux

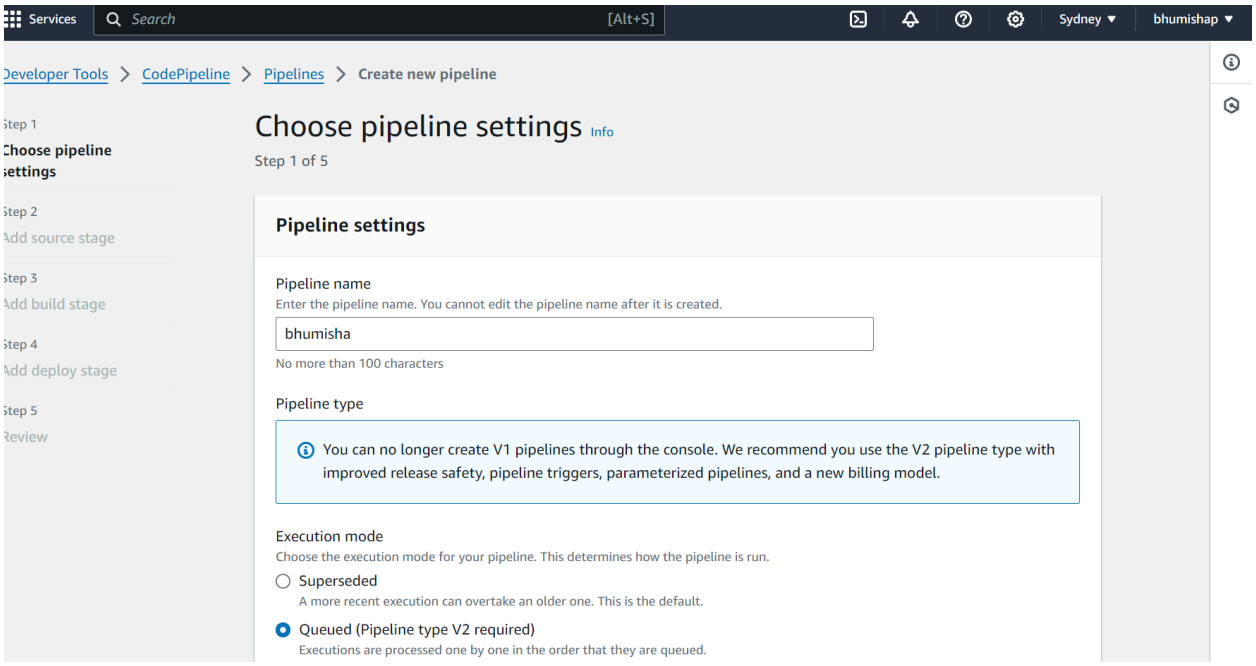


Step 8: Click on Create Pipeline.

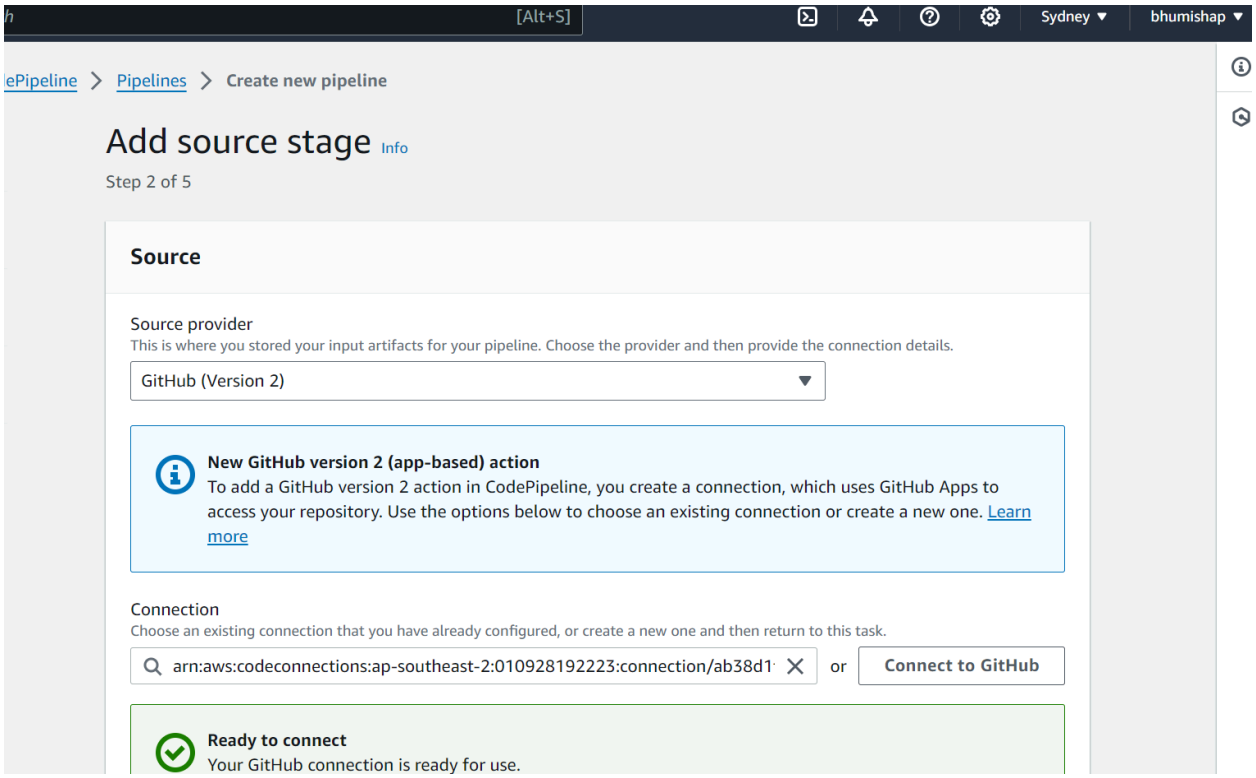




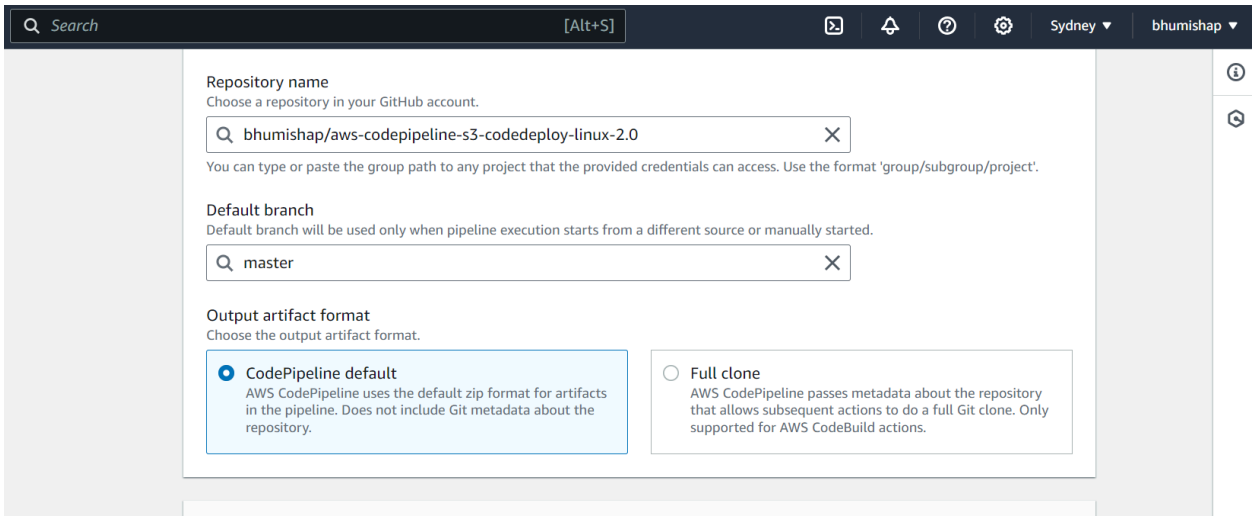
Step 9: Give a name to your Pipeline. A new service role would be created with the name of the Pipeline.



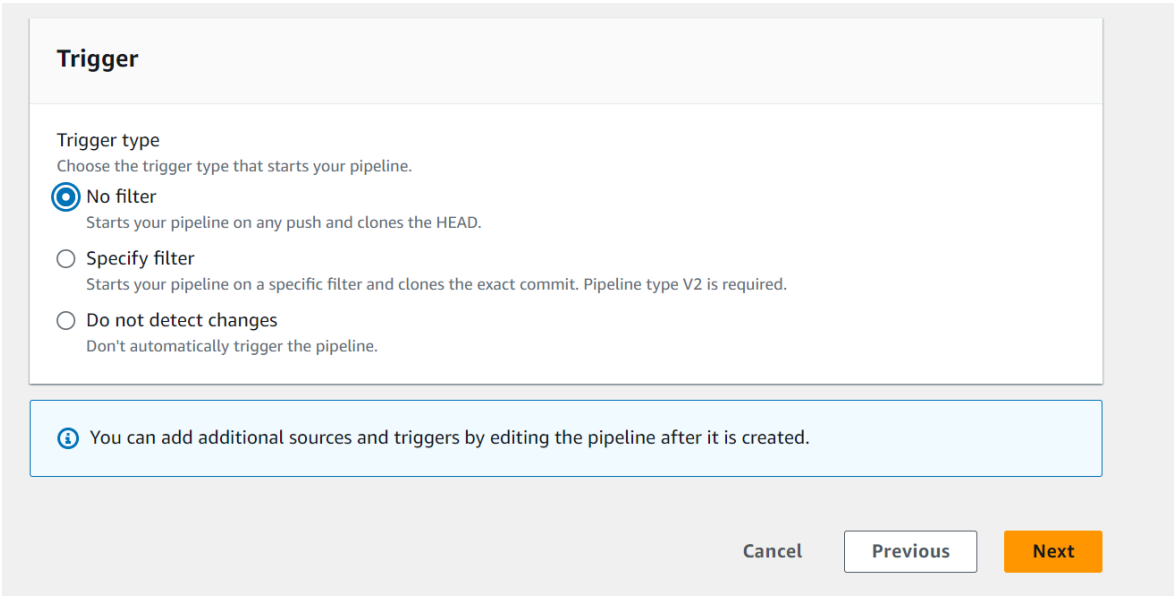
Step 10: Select a source provider (as Github (Version 2)). Click on Connect to Github to connect your github.



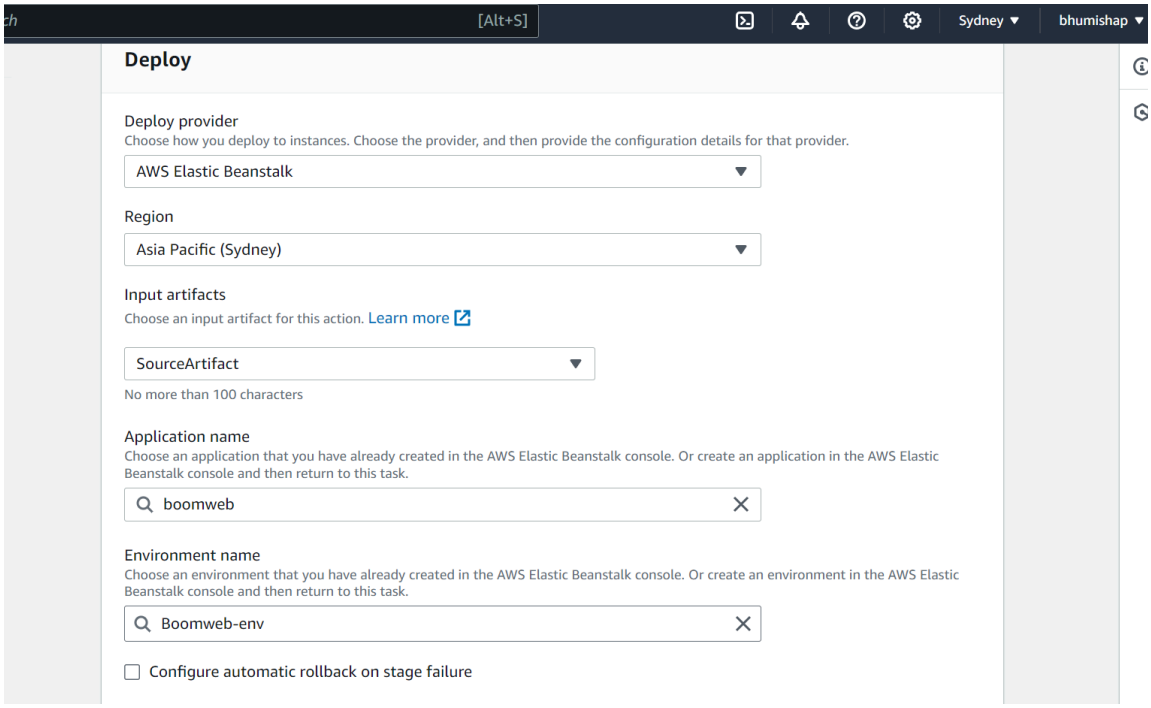
Step 11: Select the repository that you had forked to your GitHub. After that select the branch on which the files are present (default is Master).



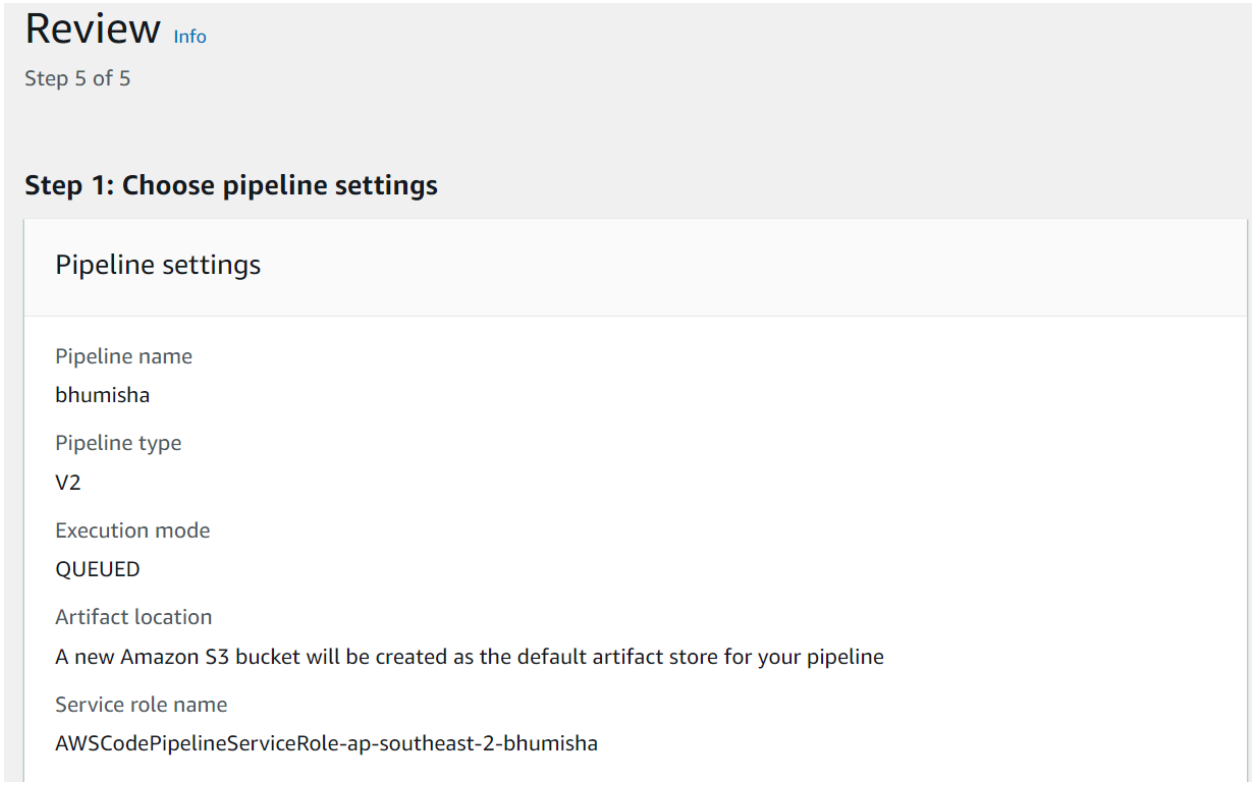
Step 12: Set the Trigger type as no filter. This would allow it to the website to update as soon as some change is made in the github.

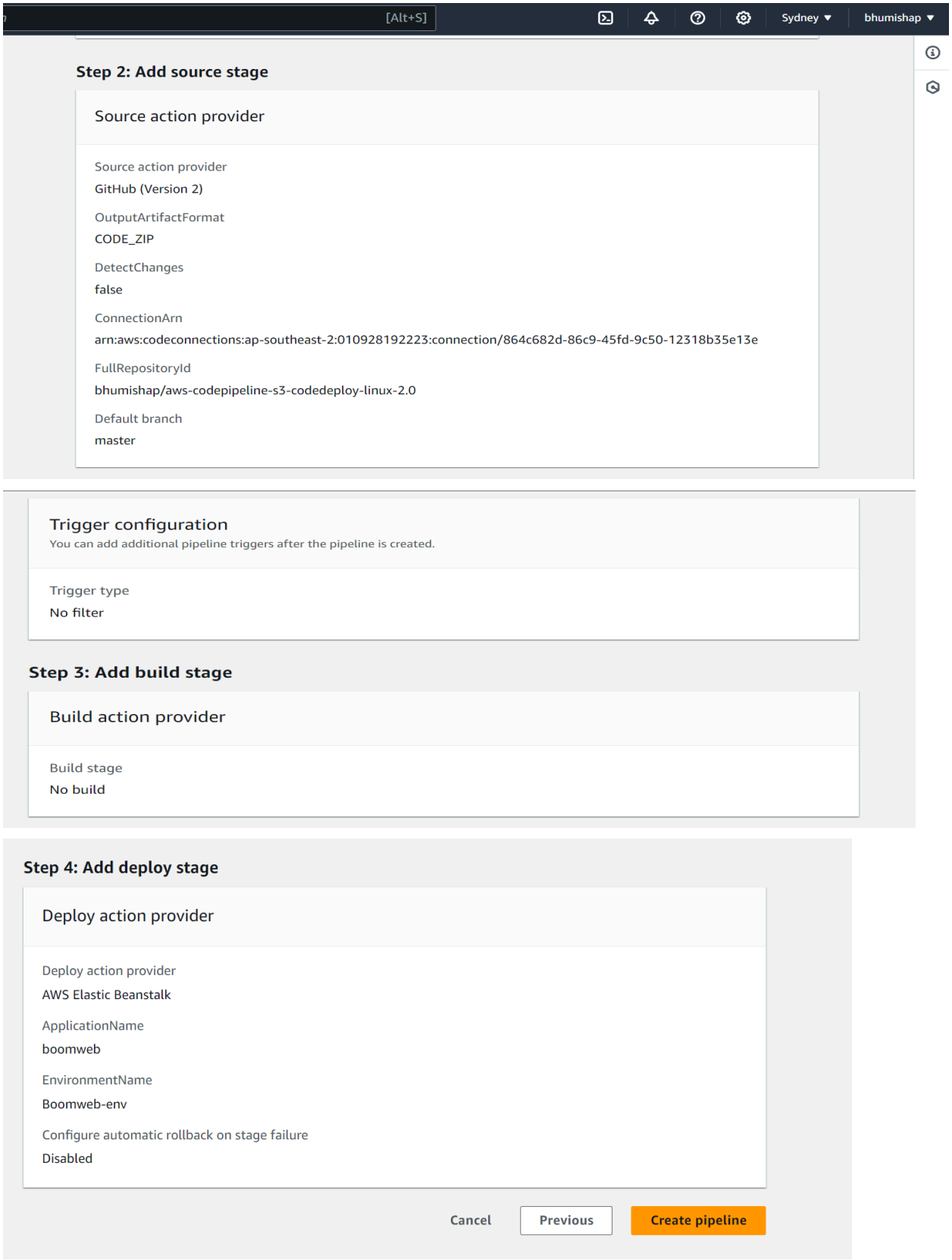


Step 13: Skip the build stage and go to Deploy. Select the deploy provider as AWS Elastic Beanstalk and Input Artifact as SourceArtifact. The application name would be the name of your Elastic Beanstalk. Then click on next.

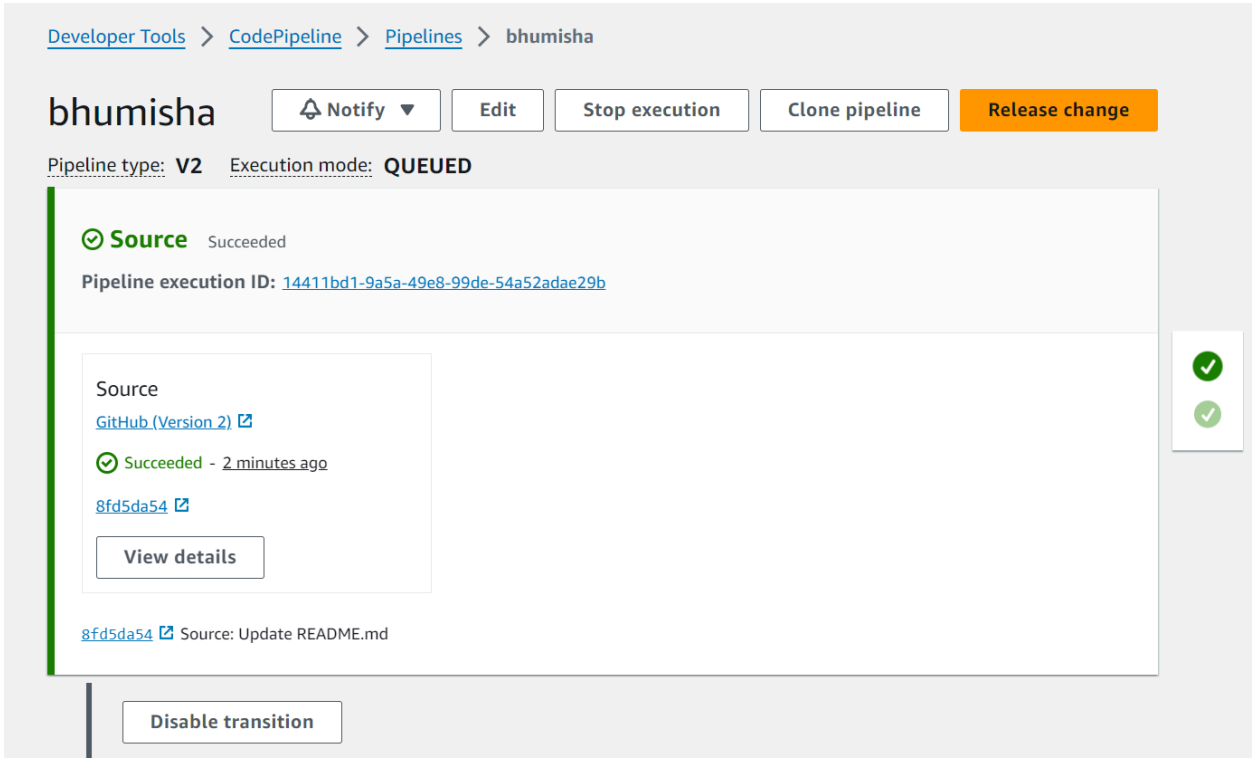


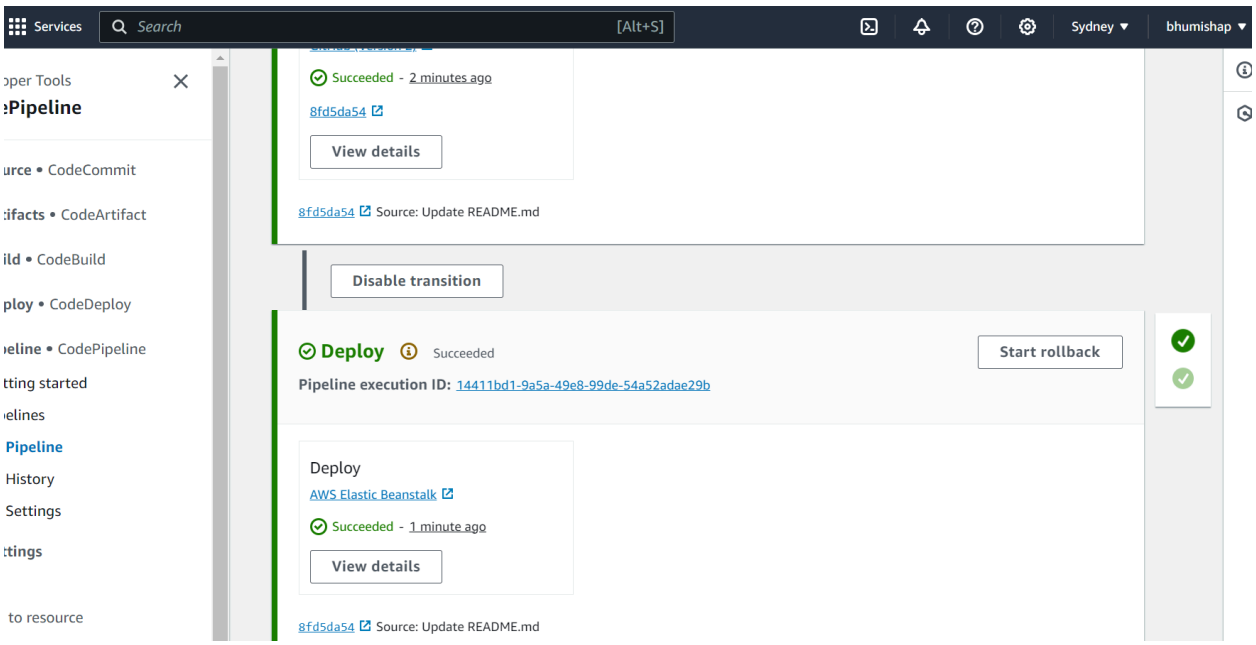
Step 14: Check all the information and click on create Pipeline.



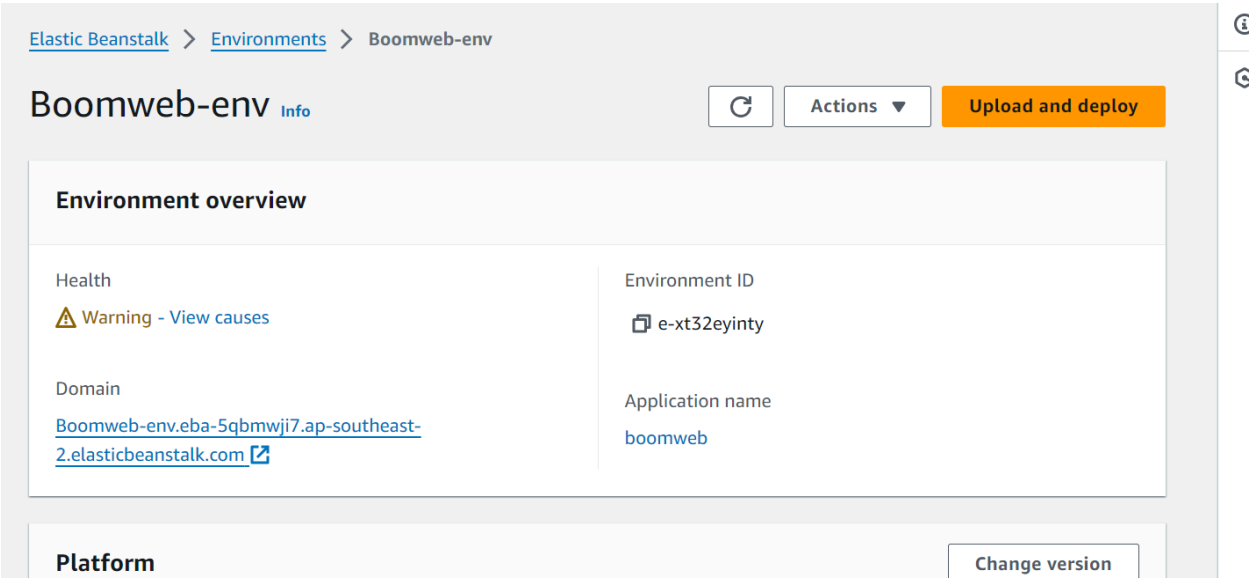

[Alt+S]
Sydney
bhumishap
Step 2: Add source stage
Source action provider
Source action provider
GitHub (Version 2)
OutputArtifactFormat
CODE_ZIP
DetectChanges
false
ConnectionArn
arn:aws:codeconnections:ap-southeast-2:010928192223:connection/864c682d-86c9-45fd-9c50-12318b35e13e
FullRepositoryId
bhumishap/aws-codepipeline-s3-codedeploy-linux-2.0
Default branch
master
Trigger configuration
You can add additional pipeline triggers after the pipeline is created.
Trigger type
No filter
Step 3: Add build stage
Build action provider
Build stage
No build
Step 4: Add deploy stage
Deploy action provider
Deploy action provider
AWS Elastic Beanstalk
ApplicationName
boomweb
EnvironmentName
Boomweb-env
Configure automatic rollback on stage failure
Disabled
Cancel
Previous
Create pipeline

Step 15: If the pipeline is successfully deployed, this screen comes up where the source is set up and then it is transitioned to deploy.

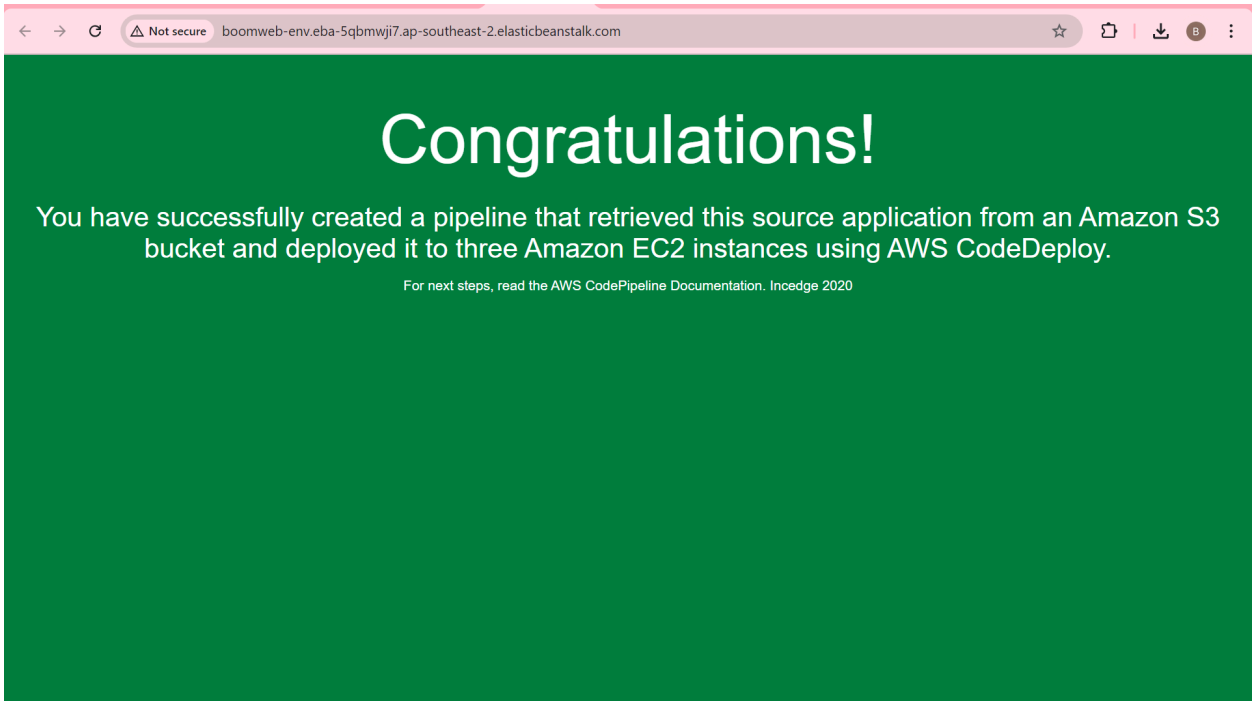




Step 16: In a few minutes the website will get hosted successfully. Then click on the url present over the environment created on Elastic Beanstalk.



Step 17: This will successfully show the sample website hosted.

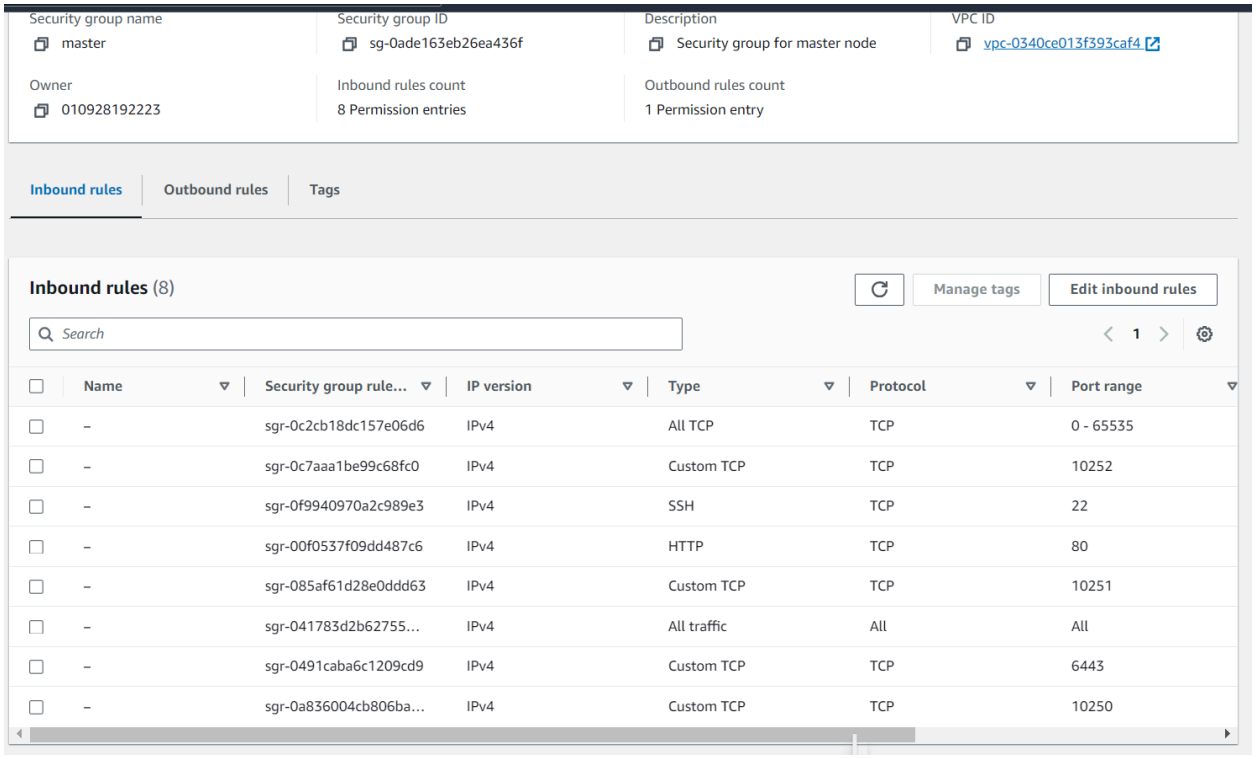


If you can see this, that means that you successfully created an automated software using CodePipeline.

EXPERIMENT NO.3

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

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



Security group name	Security group ID	Description	VPC ID
master	sg-0ade163eb26ea436f	Security group for master node	vpc-0340ce013f393caf4

Owner	Inbound rules count	Outbound rules count
010928192223	8 Permission entries	1 Permission entry

Inbound rules | Outbound rules | Tags

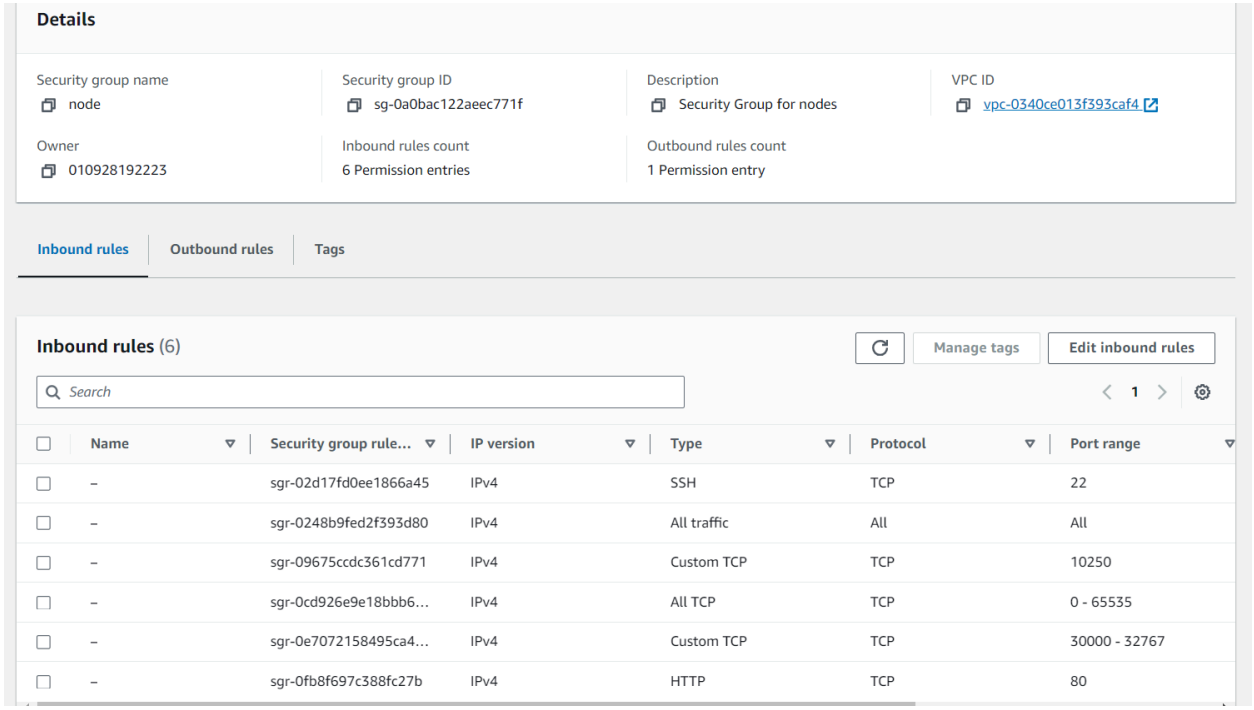
Inbound rules (8)

Manage tags | Edit inbound rules

Search

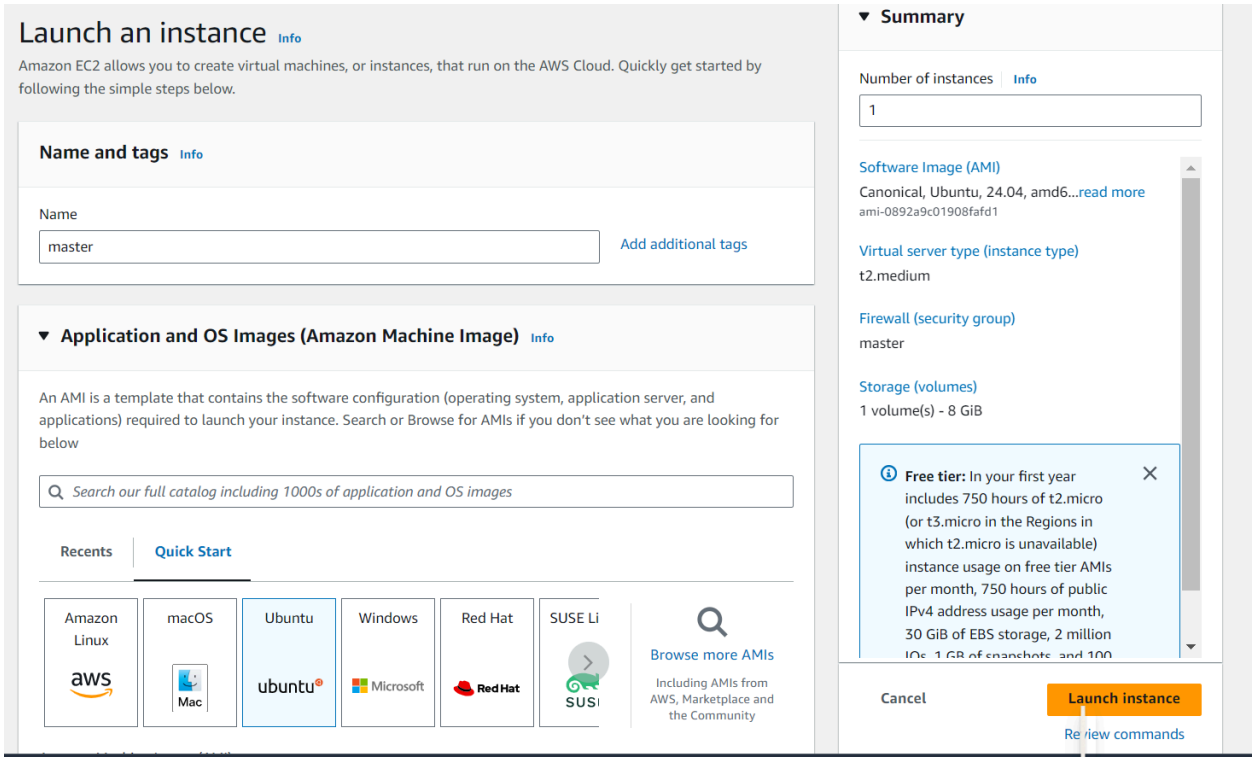
1

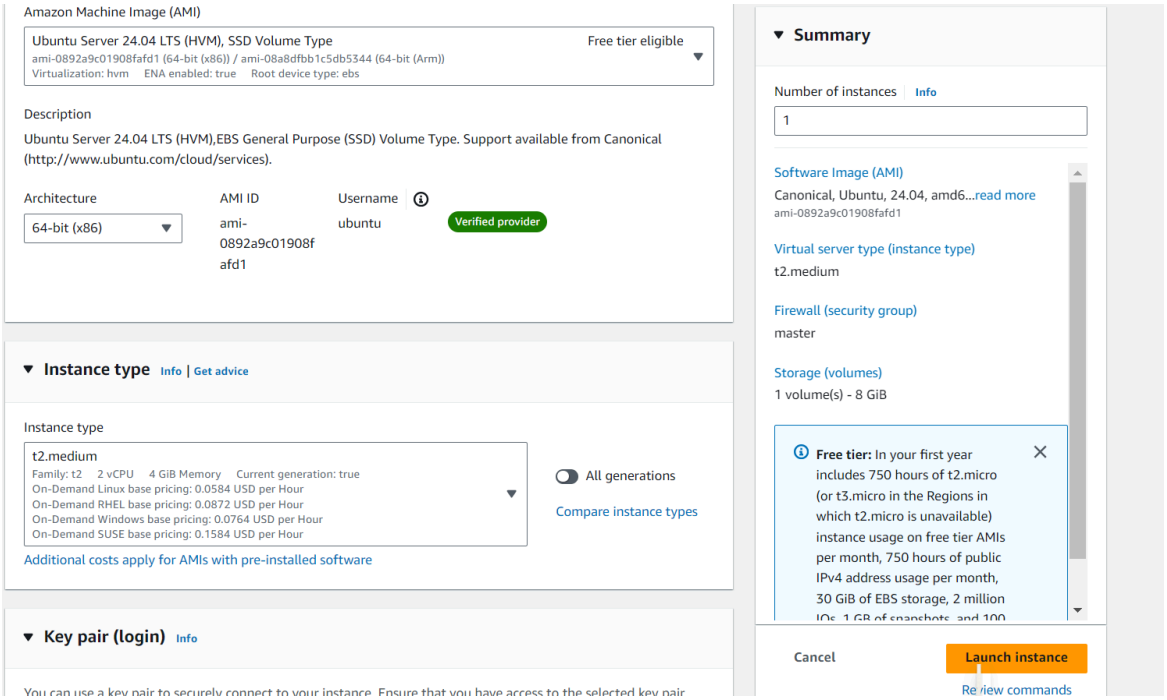
Name	Security group rule...	IP version	Type	Protocol	Port range
–	sgr-0c2cb18dc157e06d6	IPv4	All TCP	TCP	0 - 65535
–	sgr-0c7aaa1be99c68fc0	IPv4	Custom TCP	TCP	10252
–	sgr-0f9940970a2c989e3	IPv4	SSH	TCP	22
–	sgr-00f0537f09dd487c6	IPv4	HTTP	TCP	80
–	sgr-085af61d28e0ddd63	IPv4	Custom TCP	TCP	10251
–	sgr-041783d2b62755...	IPv4	All traffic	All	All
–	sgr-0491caba6c1209cd9	IPv4	Custom TCP	TCP	6443
–	sgr-0a836004cb806ba...	IPv4	Custom TCP	TCP	10250

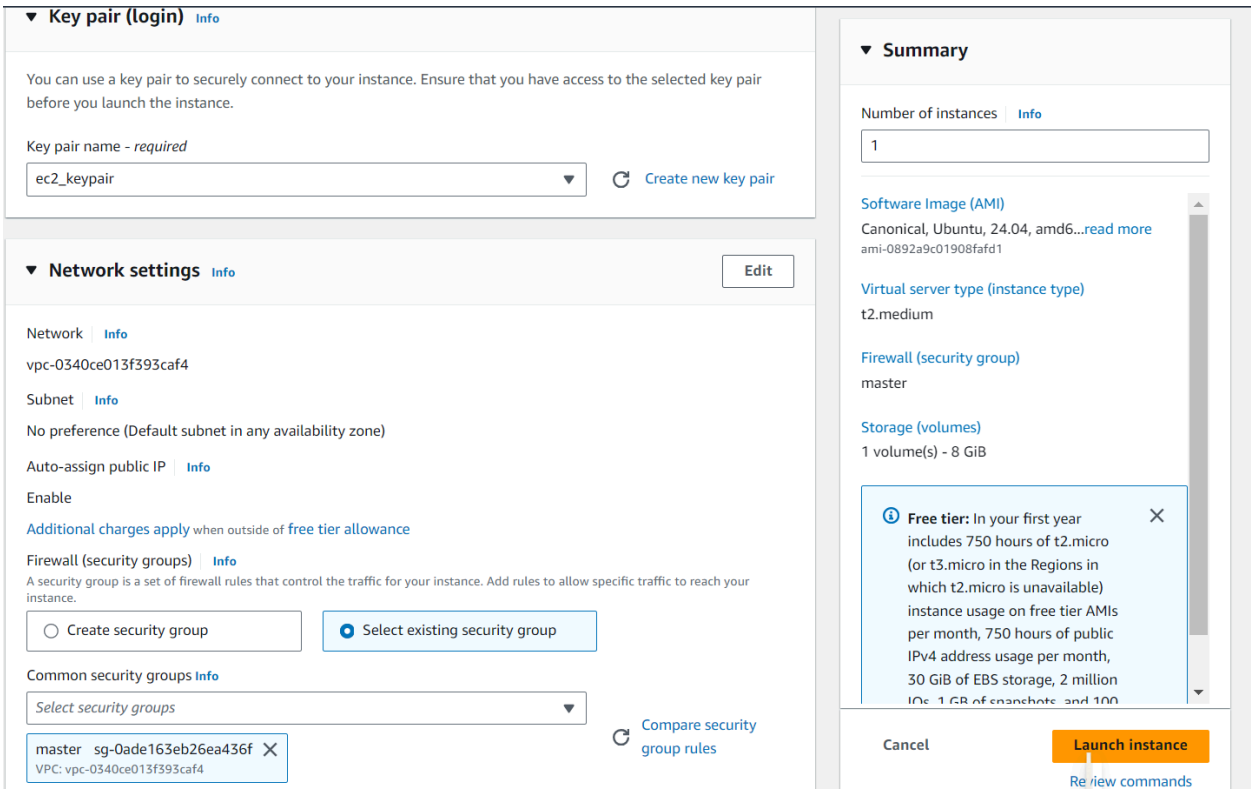


1. Log in to your AWS Academy/personal account and launch 3 new Ec2 Instances. Select Ubuntu as AMI and t2.medium as Instance Type and create a key of type RSA with .pem extension and move the downloaded key to the new folder.We can use 3 Different keys or 1 common key also.

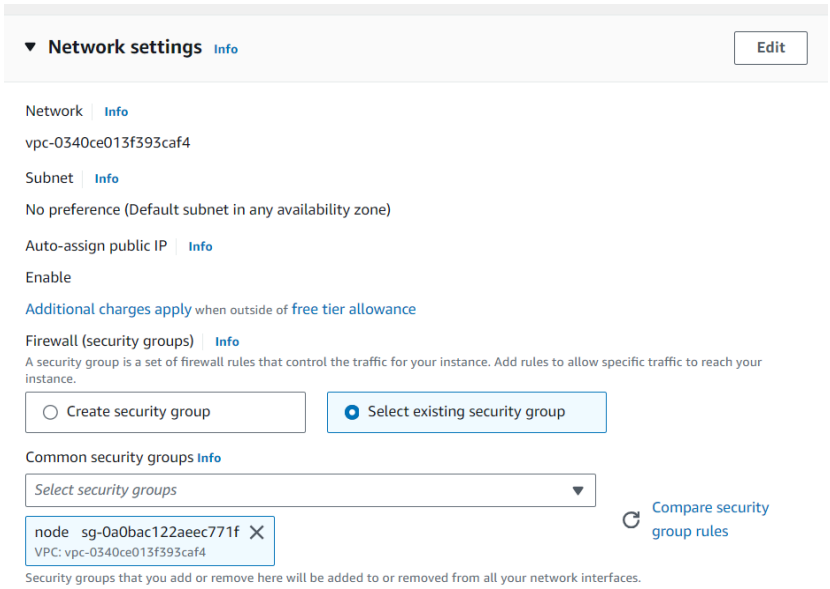
Master:



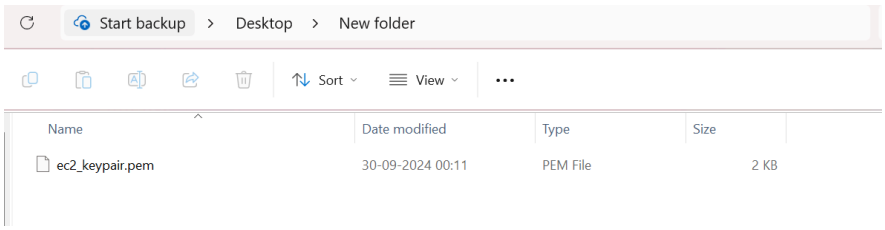




Do Same for 2 Nodes and use security groups of Node for that.

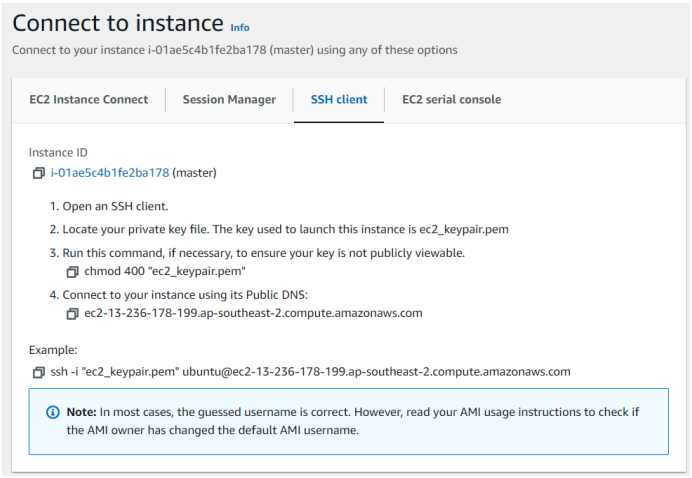


2. After creating the instances click on Connect & connect all 3 instances and navigate to SSH Client.



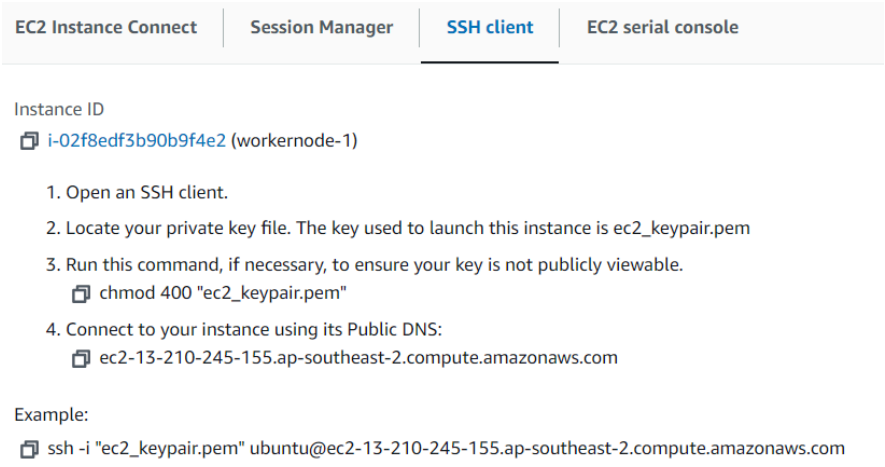
3. Now open the folder in the terminal 3 times for Master, Node1& Node 2 where our .pem key is stored. Then execute the ssh command.
For example: ssh -i "ec2_keypair.pem" ubuntu@ec2-13-236-178-199.ap-southeast-2.compute.amazonaws.com

Master:




```
PS C:\Users\bhumi> cd "C:\Users\bhumi\OneDrive\Desktop\New folder"
PS C:\Users\bhumi\OneDrive\Desktop\New folder> ssh -i "ec2_keypair.pem" ubuntu@ec2-13-236-178-199.ap-southeast-2.compute.amazonaws.com
The authenticity of host 'ec2-13-236-178-199.ap-southeast-2.compute.amazonaws.com (13.236.178.199)' can't be established.
ED25519 key fingerprint is SHA256:wumJngKU446JdujGf/vlyRzXRAKpxR+vAGODmZRXBW4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-13-236-178-199.ap-southeast-2.compute.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
 * Support:        https://ubuntu.com/pro

 System information as of Sun Sep 29 18:53:56 UTC 2024

  System load:  0.08              Processes:             115
  Usage of /:   22.8% of 6.71GB   Users logged in:       0
  Memory usage: 6%                IPv4 address for enX0: 172.31.7.184
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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


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


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

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

To run a command as administrator (user "root"), use "sudo <command>".
```

Node 1:



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


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


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

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

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

Node 2:

Instance ID

i-0630b310934847c73 (workernode-2)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is ec2_keypair.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   chmod 400 "ec2_keypair.pem"
4. Connect to your instance using its Public DNS:
   ec2-3-25-239-89.ap-southeast-2.compute.amazonaws.com

Example:

ssh -i "ec2_keypair.pem" ubuntu@ec2-3-25-239-89.ap-southeast-2.compute.amazonaws.com

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


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


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

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

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

Thus the connection is successful.

4. Run on Master,Node 1,and Node 2 the below commands to install and setup Docker in Master, Node1, and Node2.
   curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
   curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee

/etc/apt/trusted.gpg.d/docker.gpg > /dev/null
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable”

```
ubuntu@ip-172-31-7-184:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee
/etc/apt/trusted.gpg.d/docker.gpg > /dev/null
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable"
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
-----BEGIN PGP PUBLIC KEY BLOCK-----
```

```
-----END PGP PUBLIC KEY BLOCK-----
-bash: /etc/apt/trusted.gpg.d/docker.gpg: No such file or directory
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Hit:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Get:5 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [15.3 kB]
Get:8 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:9 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:10 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:11 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:12 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:13 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:14 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:15 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [535 kB]
Get:16 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [130 kB]
Get:17 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [8652 B]
Get:18 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [377 kB]
Get:19 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [156 kB]
Get:20 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [45.0 kB]
Get:21 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [14.8 kB]
Get:22 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [353 kB]
Get:23 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [68.1 kB]
Get:24 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 c-n-f Metadata [424 B]
Get:25 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [14.4 kB]
Get:26 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [3608 B]
Get:27 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [212 B]
Get:28 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [532 B]
Get:29 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:30 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [112 B]
Get:31 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [10.6 kB]
Get:32 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [10.8 kB]
Get:33 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [17.6 kB]
```

```
Get:34 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1104 B]
Get:35 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:36 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:37 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:38 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:39 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [380 kB]
Get:40 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [82.9 kB]
Get:41 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4560 B]
Get:42 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [272 kB]
Get:43 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [115 kB]
Get:44 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:45 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.3 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [353 kB]
Get:47 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [68.1 kB]
Get:48 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [428 B]
Get:49 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [10.9 kB]
Get:50 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [2808 B]
Get:51 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:52 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [344 B]
Fetched 29.1 MB in 5s (5748 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION s
ection in apt-key(8) for details.
```

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

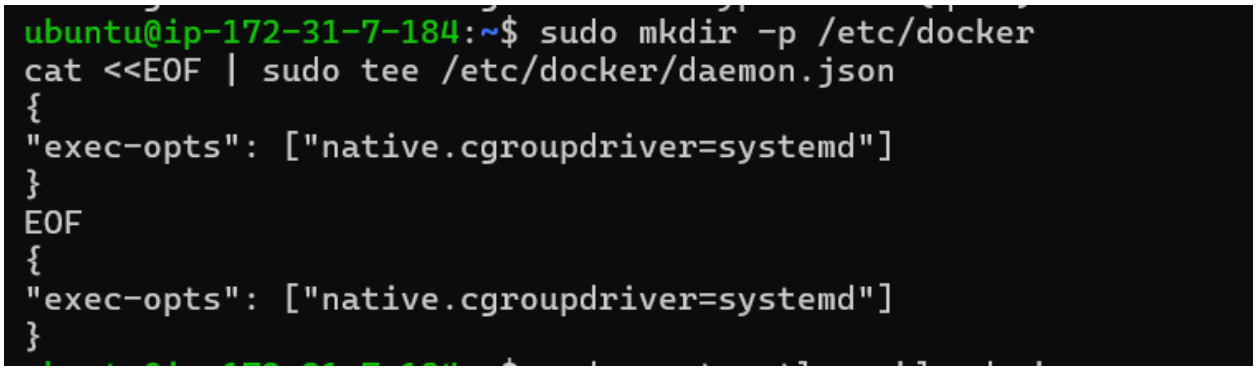
```
ubuntu@ip-172-31-7-184:~$ sudo apt-get update
sudo apt-get install -y docker-ce
Hit:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION s
ection in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0
  pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7
  libslirp0 pigz slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 143 not upgraded.
Need to get 123 MB of archives.
After this operation, 442 MB of additional disk space will be used.
Get:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libltdl7 amd64 2.4.7-7build1 [40.3 kB]
Get:3 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libslirp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
Get:4 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 slirp4netns amd64 1.2.1-1build2 [34.9 kB]
Get:5 https://download.docker.com/linux/ubuntu noble/stable amd64 containerd.io amd64 1.7.22-1 [29.5 MB]
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-buildx-plugin amd64 0.17.1-1~ubuntu.24.04~noble [30.3 MB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-cli amd64 5:27.3.1-1~ubuntu.24.04~noble [15.0 MB]
Get:8 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce amd64 5:27.3.1-1~ubuntu.24.04~noble [25.6 MB]
Get:9 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-rootless-extras amd64 5:27.3.1-1~ubuntu.24.04~noble [9588 kB]
Get:10 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-compose-plugin amd64 2.29.7-1~ubuntu.24.04~noble [12.7 MB]
Fetched 123 MB in 2s (75.3 MB/s)
```

```
Selecting previously unselected package pigz.
(Reading database ... 67741 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.8-1_amd64.deb ...
Unpacking pigz (2.8-1) ...
Selecting previously unselected package containerd.io.
Preparing to unpack .../1-containerd.io_1.7.22-1_amd64.deb ...
Unpacking containerd.io (1.7.22-1) ...
Selecting previously unselected package docker-buildx-plugin.
Preparing to unpack .../2-docker-buildx-plugin_0.17.1-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-buildx-plugin (0.17.1-1~ubuntu.24.04~noble) ...
Selecting previously unselected package docker-ce-cli.
Preparing to unpack .../3-docker-ce-cli_5%3a27.3.1-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-ce-cli (5:27.3.1-1~ubuntu.24.04~noble) ...
Selecting previously unselected package docker-ce.
Preparing to unpack .../4-docker-ce_5%3a27.3.1-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...
Selecting previously unselected package docker-ce-rootless-extras.
Preparing to unpack .../5-docker-ce-rootless-extras_5%3a27.3.1-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-ce-rootless-extras (5:27.3.1-1~ubuntu.24.04~noble) ...
Selecting previously unselected package docker-compose-plugin.
Preparing to unpack .../6-docker-compose-plugin_2.29.7-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-compose-plugin (2.29.7-1~ubuntu.24.04~noble) ...
Selecting previously unselected package libltdl7:amd64.
Preparing to unpack .../7-libltdl7_2.4.7-7build1_amd64.deb ...
```

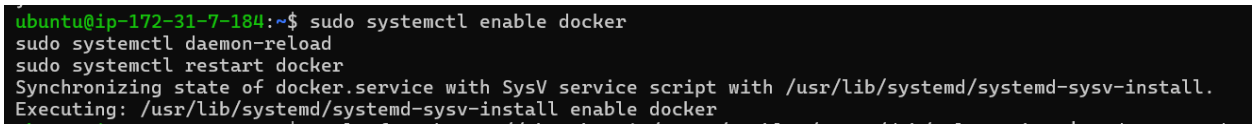
```
Unpacking slirp4netns (1.2.1-1build2) ...
Setting up docker-buildx-plugin (0.17.1-1~ubuntu.24.04~noble) ...
Setting up containerd.io (1.7.22-1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /usr/lib/systemd/system/containerd.service.
Setting up docker-compose-plugin (2.29.7-1~ubuntu.24.04~noble) ...
Setting up libltdl7:amd64 (2.4.7-7build1) ...
Setting up docker-ce-cli (5:27.3.1-1~ubuntu.24.04~noble) ...
Setting up libslirp0:amd64 (4.7.0-1ubuntu3) ...
Setting up pigz (2.8-1) ...
Setting up docker-ce-rootless-extras (5:27.3.1-1~ubuntu.24.04~noble) ...
Setting up slirp4netns (1.2.1-1build2) ...
Setting up docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

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

sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json

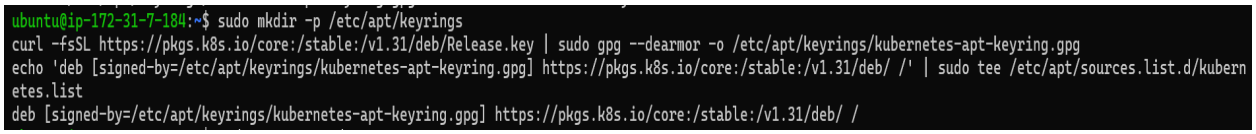
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF



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



5. Run the below command to install Kubernets.
   curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
   echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list



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

```
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
ubuntu@ip-172-31-7-184:~$ sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
Hit:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease [1186 B]
Hit:6 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  Packages [4865 B]
Fetched 6051 B in 1s (10.9 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION s
ection in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  conntrack cri-tools kubeadm kubectl kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 143 not upgraded.
Need to get 87.4 MB of archives.
After this operation, 314 MB of additional disk space will be used.
Get:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 conntrack amd64 1:1.4.8-1ubuntu1 [37.9 kB]
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  cri-tools 1.31.1-1.1 [15.7 MB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubeadm 1.31.1-1.1 [11.4 MB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubectl 1.31.1-1.1 [11.2 MB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubernetes-cni 1.5.1-1.1 [33.9 MB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubelet 1.31.1-1.1 [15.2 MB]
```

```
Preparing to unpack .../3-kubectl_1.31.1-1.1_amd64.deb ...
Unpacking kubectl (1.31.1-1.1) ...
Selecting previously unselected package kubernetes-cni.
Preparing to unpack .../4-kubernetes-cni_1.5.1-1.1_amd64.deb ...
Unpacking kubernetes-cni (1.5.1-1.1) ...
Selecting previously unselected package kubelet.
Preparing to unpack .../5-kubelet_1.31.1-1.1_amd64.deb ...
Unpacking kubelet (1.31.1-1.1) ...
Setting up conntrack (1:1.4.8-1ubuntu1) ...
Setting up kubectl (1.31.1-1.1) ...
Setting up cri-tools (1.31.1-1.1) ...
Setting up kubernetes-cni (1.5.1-1.1) ...
Setting up kubeadm (1.31.1-1.1) ...
Setting up kubelet (1.31.1-1.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
```

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

```
ubuntu@ip-172-31-7-184:~$ sudo systemctl enable --now kubelet
sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz
  slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 143 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
Get:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 MB]
Fetched 47.2 MB in 1s (61.8 MB/s)
(Reading database ... 68064 files and directories currently installed.)
Removing docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...
Removing containerd.io (1.7.22-1) ...
Selecting previously unselected package runc.
(Reading database ... 68044 files and directories currently installed.)
```

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

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

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

sudo mkdir -p /etc/containerd

sudo containerd config default | sudo tee /etc/containerd/config.toml

```
ubuntu@ip-172-31-7-184:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

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

[grpc]
  address = "/run/containerd/containerd.sock"
  gid = 0
  max_recv_message_size = 16777216
  max_send_message_size = 16777216
  tcp_address = ""
  tcp_tls_ca = ""
  tcp_tls_cert = ""
  tcp_tls_key = ""
  uid = 0
```

```
  [stream_processors."io.containerd.ocicrypt.decoder.v1.tar"]
    accepts = ["application/vnd.oci.image.layer.v1.tar+encrypted"]
    args = ["--decryption-keys-path", "/etc/containerd/ocicrypt/keys"]
    env = ["OCICRYPT_KEYPROVIDER_CONFIG=/etc/containerd/ocicrypt/ocicrypt_keyprovider.conf"]
    path = "ctd-decoder"
    returns = "application/vnd.oci.image.layer.v1.tar"

  [stream_processors."io.containerd.ocicrypt.decoder.v1.tar.gzip"]
    accepts = ["application/vnd.oci.image.layer.v1.tar+gzip+encrypted"]
    args = ["--decryption-keys-path", "/etc/containerd/ocicrypt/keys"]
    env = ["OCICRYPT_KEYPROVIDER_CONFIG=/etc/containerd/ocicrypt/ocicrypt_keyprovider.conf"]
    path = "ctd-decoder"
    returns = "application/vnd.oci.image.layer.v1.tar+gzip"

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

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

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

```
ubuntu@ip-172-31-7-184:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd
● containerd.service - containerd container runtime
     Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
     Active: active (running) since Sun 2024-09-29 19:01:59 UTC; 268ms ago
       Docs: https://containerd.io
   Main PID: 4582 (containerd)
      Tasks: 7
     Memory: 13.3M (peak: 13.7M)
        CPU: 77ms
     CGroup: /system.slice/containerd.service
             └─4582 /usr/bin/containerd

Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453685323Z" level=info msg="Start subscribi>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453745481Z" level=info msg="Start recoverin>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453808448Z" level=info msg="Start event mon>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453816198Z" level=info msg=serving... addre>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453830731Z" level=info msg="Start snapshots>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453844510Z" level=info msg="Start cni netwo>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453854312Z" level=info msg="Start streaming>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453861743Z" level=info msg=serving... addre>
Sep 29 19:01:59 ip-172-31-7-184 containerd[4582]: time="2024-09-29T19:01:59.453945451Z" level=info msg="containerd succ>
Sep 29 19:01:59 ip-172-31-7-184 systemd[1]: Started containerd.service - containerd container runtime.
```

sudo apt-get install -y socat

```
ubuntu@ip-172-31-7-184:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz
  slirp4netns
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 143 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.0.0-4build3 [374 kB]
Fetched 374 kB in 0s (22.0 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68108 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.0-4build3_amd64.deb ...
Unpacking socat (1.8.0.0-4build3) ...
Setting up socat (1.8.0.0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

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

6. Initialize the Kubecluster .Now Perform this Command only for Master.
   sudo kubeadm init --pod-network-cidr=10.244.0.0/16

```
ubuntu@ip-172-31-7-184:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W0929 19:02:58.714702    4783 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent 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-172-31-7-184 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.
local] and IPs [10.96.0.1 172.31.7.184]
[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-172-31-7-184 localhost] and IPs [172.31.7.184 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-7-184 localhost] and IPs [172.31.7.184 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file
[kubeconfig] Writing "scheduler.conf" kubeconfig file
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"
[control-plane] Using manifest folder "/etc/kubernetes/manifests"
[control-plane] Creating static Pod manifest for "kube-apiserver"
```

```
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 172.31.7.184:6443 --token asa7n9.0mkjob1gsfy3xuzy \
        --discovery-token-ca-cert-hash sha256:569daba7cee31b6f3c954325f206ce87c8d3fa2fa739e5f66641ddea8d1bf13c
```

Run this command on master and also copy and save the Join command from above.
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

```
ubuntu@ip-172-31-7-184:~$   mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

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

```
ubuntu@ip-172-31-7-184:~$ kubectl get nodes
NAME              STATUS     ROLES           AGE   VERSION
ip-172-31-7-184   NotReady   control-plane   98s   v1.31.1
```

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

sudo kubeadm join 172.31.7.184:6443 --token asa7n9.0mkjob1gsfy3xuzy \
--discovery-token-ca-cert-hash
sha256:569daba7cee31b6f3c954325f206ce87c8d3fa2fa739e5f66641ddea8d1bf13c

Node 1:

```
ubuntu@ip-172-31-3-88:~$ sudo kubeadm join 172.31.7.184:6443 --token asa7n9.0mkjob1gsfy3xuzy \
        --discovery-token-ca-cert-hash sha256:569daba7cee31b6f3c954325f206ce87c8d3fa2fa739e5f66641ddea8d1bf13c
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 500.662824ms
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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

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

Node 2:

```
ubuntu@ip-172-31-7-177:~$  sudo kubeadm join 172.31.7.184:6443 --token asa7n9.0mkjob1gsfy3xuzy \
        --discovery-token-ca-cert-hash sha256:569daba7cee31b6f3c954325f206ce87c8d3fa2fa739e5f66641ddea8d1bf13c
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 1.00113109s
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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

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

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

```
ubuntu@ip-172-31-7-184:~$ kubectl get nodes
NAME              STATUS     ROLES           AGE     VERSION
ip-172-31-3-88    NotReady   <none>          44s     v1.31.1
ip-172-31-7-177   NotReady   <none>          17s     v1.31.1
ip-172-31-7-184   NotReady   control-plane   4m42s   v1.31.1
```

10. Since Status is NotReady we have to add a network plugin. And also we have to give the name to the nodes.
    kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml

```
ubuntu@ip-172-31-7-184:~$ kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
poddisruptionbudget.policy/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
serviceaccount/calico-node created
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgppeers.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/blockaffinities.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/caliconodestatuses.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/clusterinformations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/felixconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworksets.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/hostendpoints.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamblocks.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamconfigs.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamhandles.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippools.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipreservations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/kubecontrollersconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networksets.crd.projectcalico.org created
clusterrole.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrole.rbac.authorization.k8s.io/calico-node created
clusterrolebinding.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
daemonset.apps/calico-node created
deployment.apps/calico-kube-controllers created
```

    sudo systemctl status kubelet

```
ubuntu@ip-172-31-7-184:~$ sudo systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
     Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; preset: enabled)
    Drop-In: /usr/lib/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: active (running) since Sun 2024-09-29 19:04:03 UTC; 5min ago
       Docs: https://kubernetes.io/docs/
   Main PID: 5478 (kubelet)
      Tasks: 10 (limit: 4676)
     Memory: 32.3M (peak: 32.8M)
        CPU: 6.520s
     CGroup: /system.slice/kubelet.service
             └─5478 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/ku>

Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454092    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454110    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454149    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454163    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454180    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454207    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:24 ip-172-31-7-184 kubelet[5478]: I0929 19:09:24.454222    5478 reconciler_common.go:245] "operationExecut>
Sep 29 19:09:29 ip-172-31-7-184 kubelet[5478]: E0929 19:09:29.041596    5478 kubelet.go:2902] "Container runtime networ>
Sep 29 19:09:34 ip-172-31-7-184 kubelet[5478]: E0929 19:09:34.042222    5478 kubelet.go:2902] "Container runtime networ>
Sep 29 19:09:35 ip-172-31-7-184 kubelet[5478]: I0929 19:09:35.871944    5478 scope.go:117] "RemoveContainer" containerI>
```

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

```
ubuntu@ip-172-31-7-184:~$ kubectl get nodes -o wide
NAME              STATUS   ROLES           AGE    VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-3-88    Ready    <none>          2m2s   v1.31.1   172.31.3.88     <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-7-177   Ready    <none>          95s    v1.31.1   172.31.7.177    <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-7-184   Ready    control-plane   6m     v1.31.1   172.31.7.184    <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
```

Now to Rename run this command kubectl label node ip-172-31-18-135 kubernetes.io/role=worker

Rename to Node 1:kubectl label node ip-172-31-3-88 kubernetes.io/role=Worker-Node1

Rename to Node 2:kubectl label node ip-172-31-7-177kubernetes.io/role=Worker-Node2

```
ubuntu@ip-172-31-7-184:~$ kubectl label node ip-172-31-3-88 kubernetes.io/role=Worker-Node1
node/ip-172-31-3-88 labeled
ubuntu@ip-172-31-7-184:~$ kubectl label node ip-172-31-7-177 kubernetes.io/role=Worker-Node2
node/ip-172-31-7-177 labeled
```

11. Now run kubectl get nodes

```
ubuntu@ip-172-31-7-184:~$ kubectl get nodes
NAME              STATUS   ROLES           AGE     VERSION
ip-172-31-3-88    Ready    Worker-Node1    4m22s   v1.31.1
ip-172-31-7-177   Ready    Worker-Node2    3m55s   v1.31.1
ip-172-31-7-184   Ready    control-plane   8m20s   v1.31.1
ubuntu@ip-172-31-7-184:~$
```

Hence we can see we have Successfully connected Node 1 and Node 2 to the Master.

Conclusion:

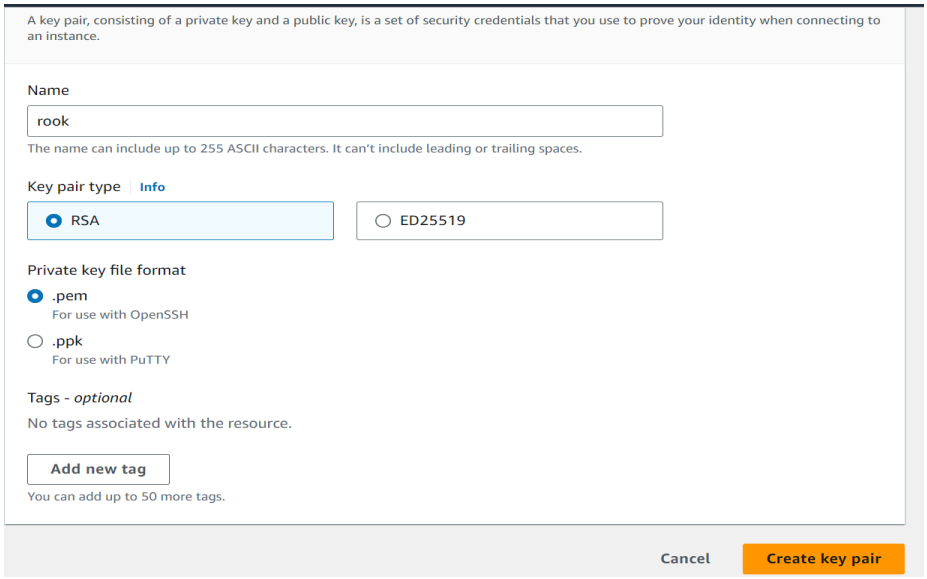
In this experiment, we successfully set up a Kubernetes cluster with one master and two worker nodes on AWS EC2 instances. After installing Docker, Kubernetes tools (kubelet, kubeadm, kubectl), and containerd on all nodes, the master node was initialized and the worker nodes were joined to the cluster. Initially, the nodes were in the NotReady state, which was resolved by installing the Calico network plugin. We also labeled the nodes with appropriate roles (control-plane and worker). The cluster became fully functional with all nodes in the Ready state, demonstrating the successful configuration and orchestration of Kubernetes.

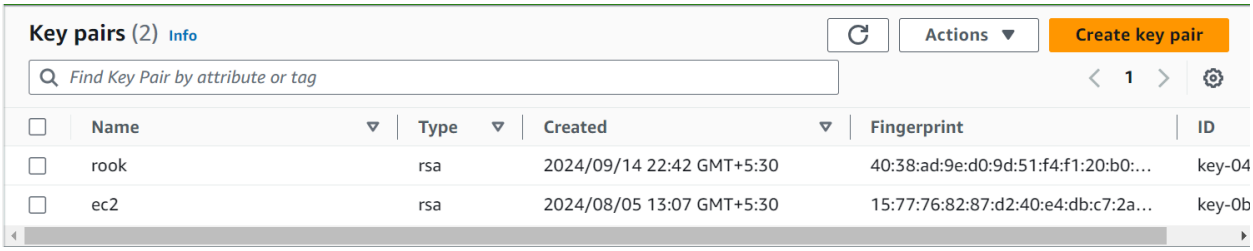
EXPERIMENT NO. 4

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

Steps:

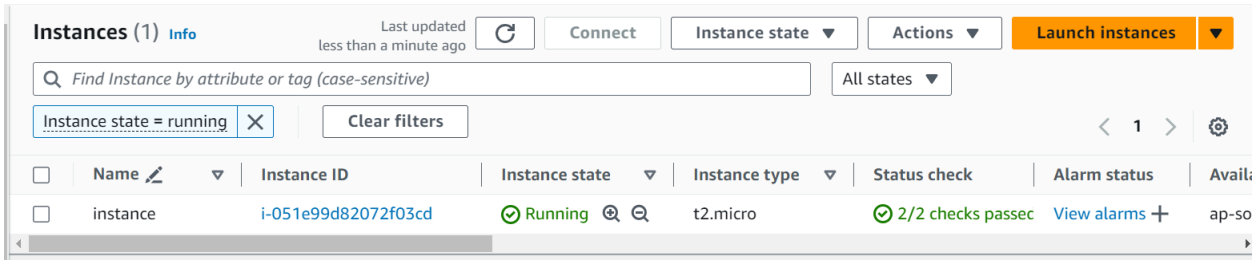
1. Create a key pair.



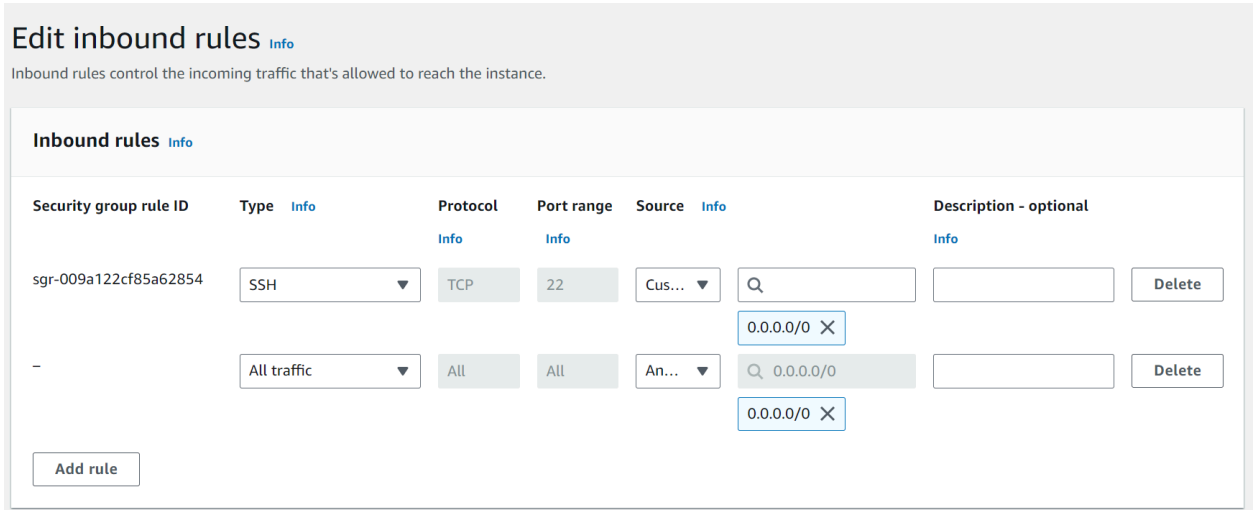


The .pem file will be downloaded on your machine and will be required in the further steps.

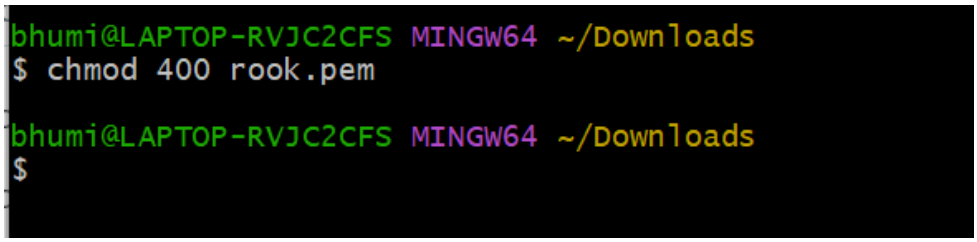
2. Now we will create an EC2 Ubuntu instance. Select the key pair which you just created while creating this instance.



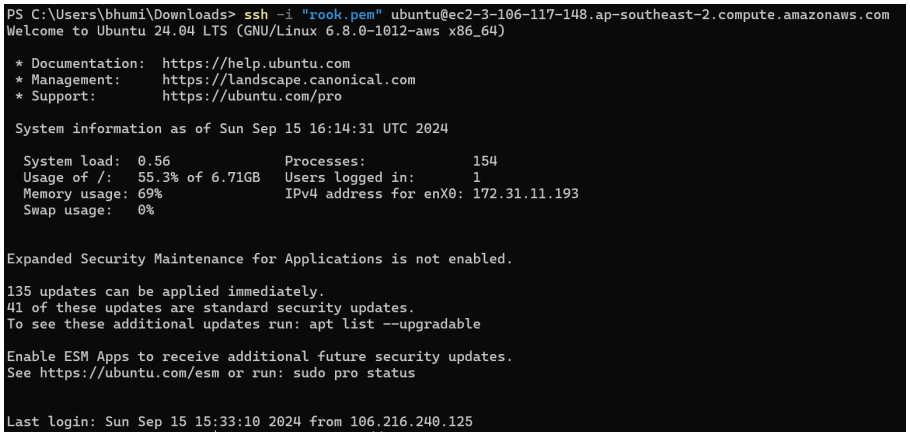
3. Now edit the inbound rules to allow ssh.



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



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.



6. Docker installation:
   We will be installing docker by using “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!
```

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

```
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.
```

8. Kubernetes installation:
   Search kubeadm installation on your browser and scroll down and select red hat-based 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

```
ubuntu@ip-172-31-11-193:~$ sudo swapoff -a
ubuntu@ip-172-31-11-193:~$ echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf
net.bridge.bridge-nf-call-iptables=1
ubuntu@ip-172-31-11-193:~$ sudo sysctl -p
net.bridge.bridge-nf-call-iptables = 1
ubuntu@ip-172-31-11-193:~$
```

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
```

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

```
ubuntu@ip-172-31-11-193:~$    mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

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

```
ubuntu@ip-172-31-11-193:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-f
lannel.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
```

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

```
ubuntu@ip-172-31-11-193:~$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
```

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

```
ubuntu@ip-172-31-11-193:~$ kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-49tlj   0/1     Pending   0          13s
nginx-deployment-d556bf558-qjqbg   0/1     Pending   0          13s
```

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.

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.

```
ubuntu@ip-172-31-11-193:~$ kubectl get pods
NAME                     READY   STATUS    RESTARTS   AGE
nginx-68cf7659df-zx7vp   1/1     Running   0          19s
```

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

```
ubuntu@ip-172-31-11-193:~$ kubectl port-forward $POD_NAME 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
Handling connection for 8080
```

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

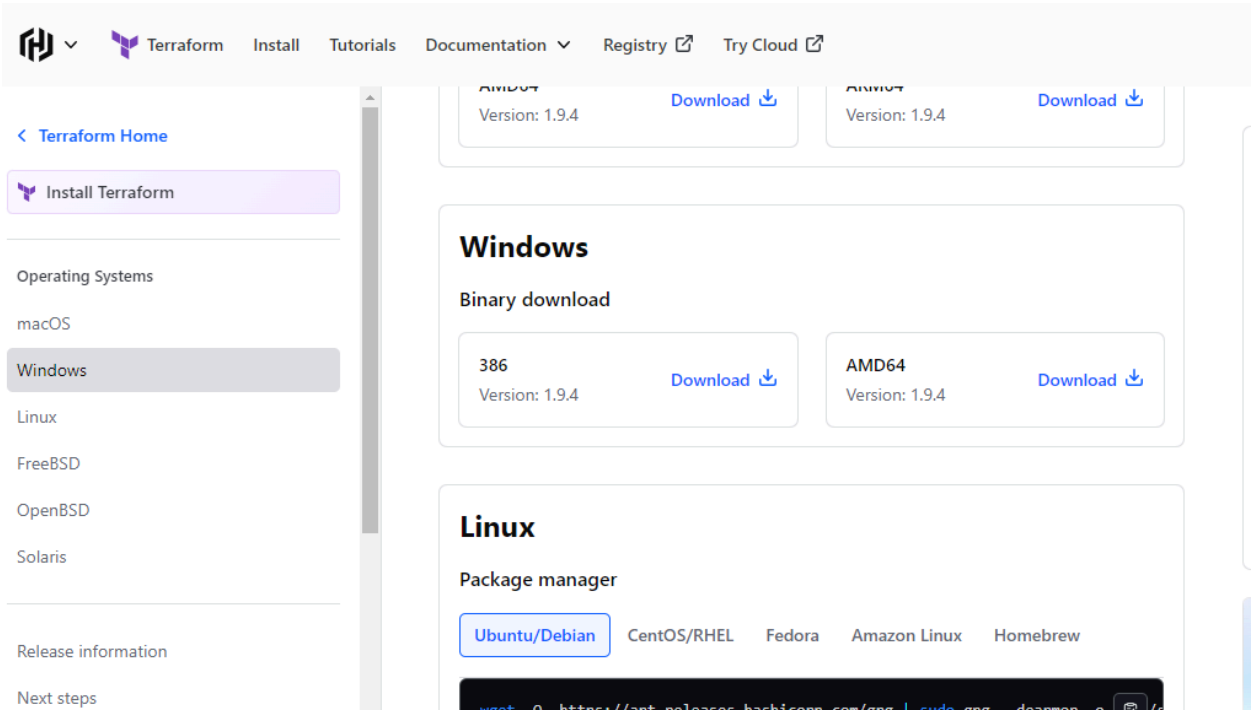
```
ubuntu@ip-172-31-11-193:~$ curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Sun, 15 Sep 2024 16:14:46 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 04 Dec 2018 14:44:49 GMT
Connection: keep-alive
ETag: "5c0692e1-264"
Accept-Ranges: bytes
```

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.

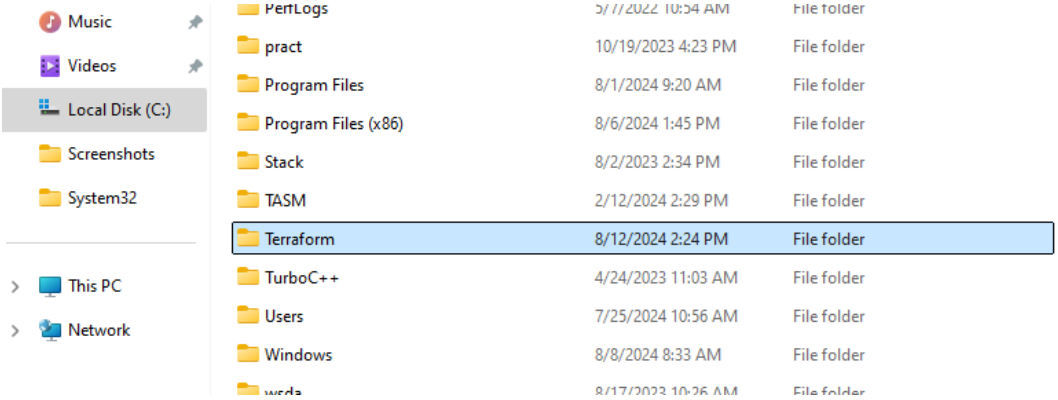
EXPERIMENT NO. 5

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

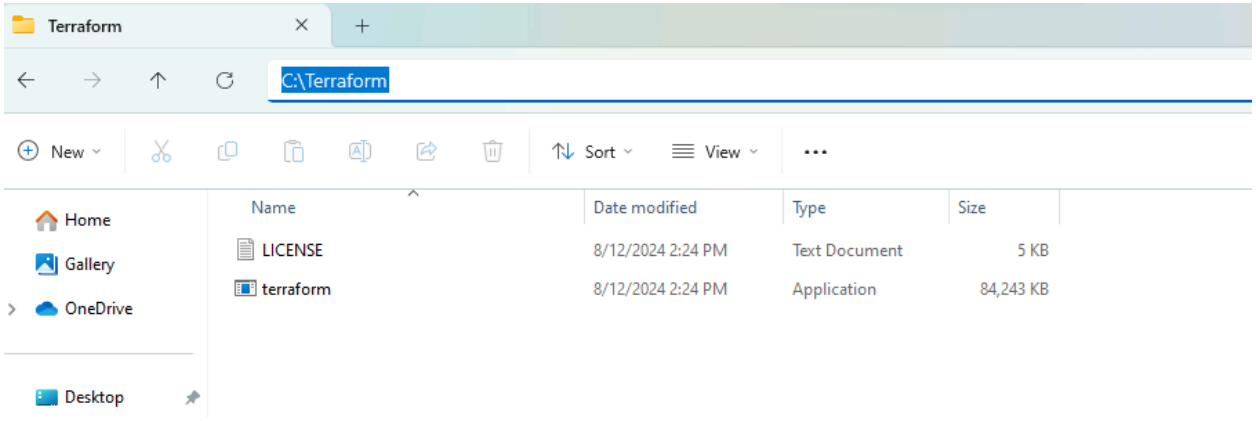
Step 1: Navigate to hashicorp.com page and download the AMD64 file for Terraform. Install it for Windows.



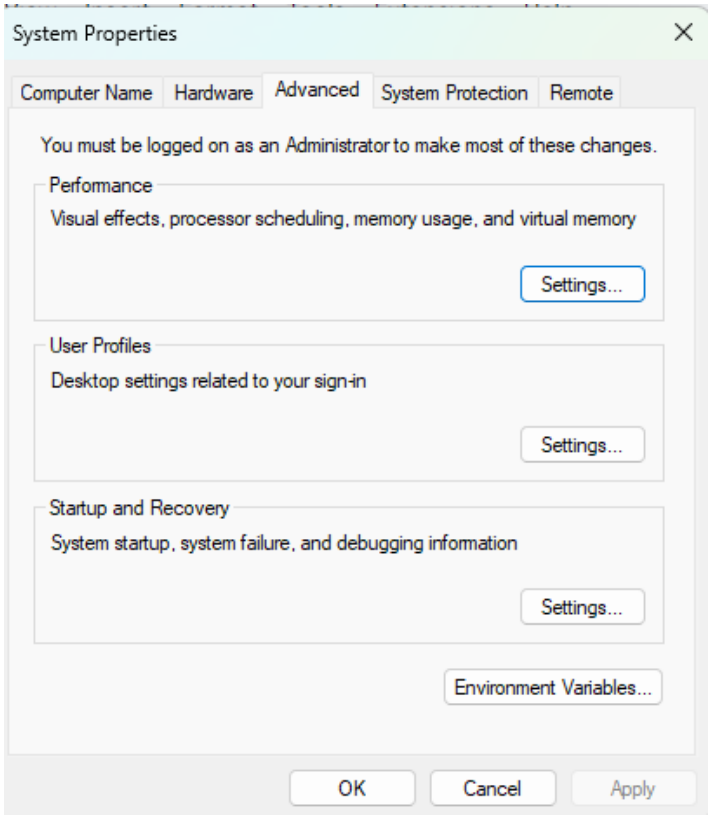
Step 2: Create a new folder inside your C drive named Terraform for your convenience. We would need it for setting up our path variable inside system variables.

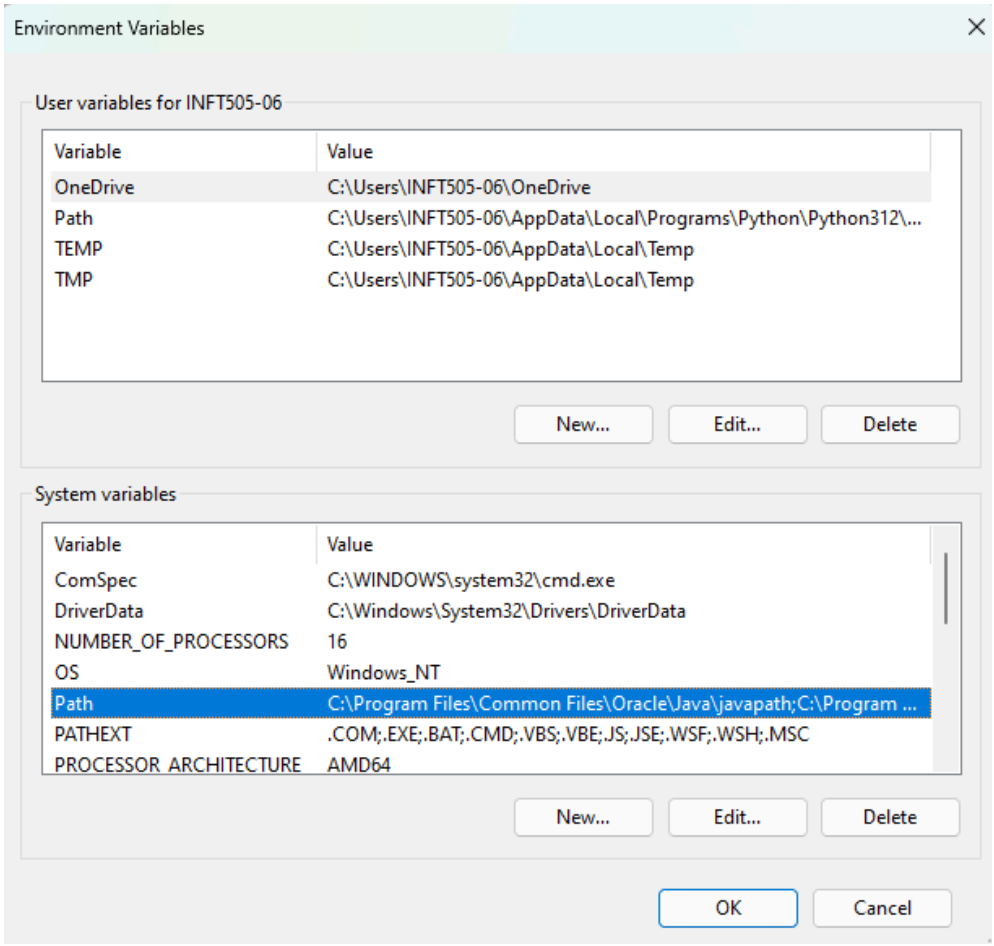


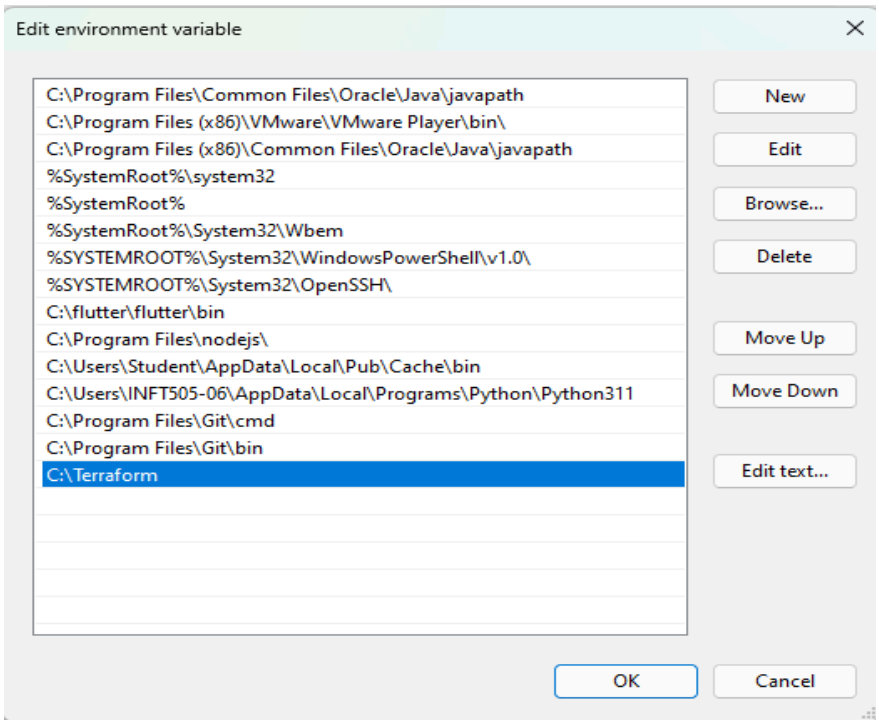
Step 3: Ensure that you have the following files appearing in your newly made Terraform folder in your local C drive.



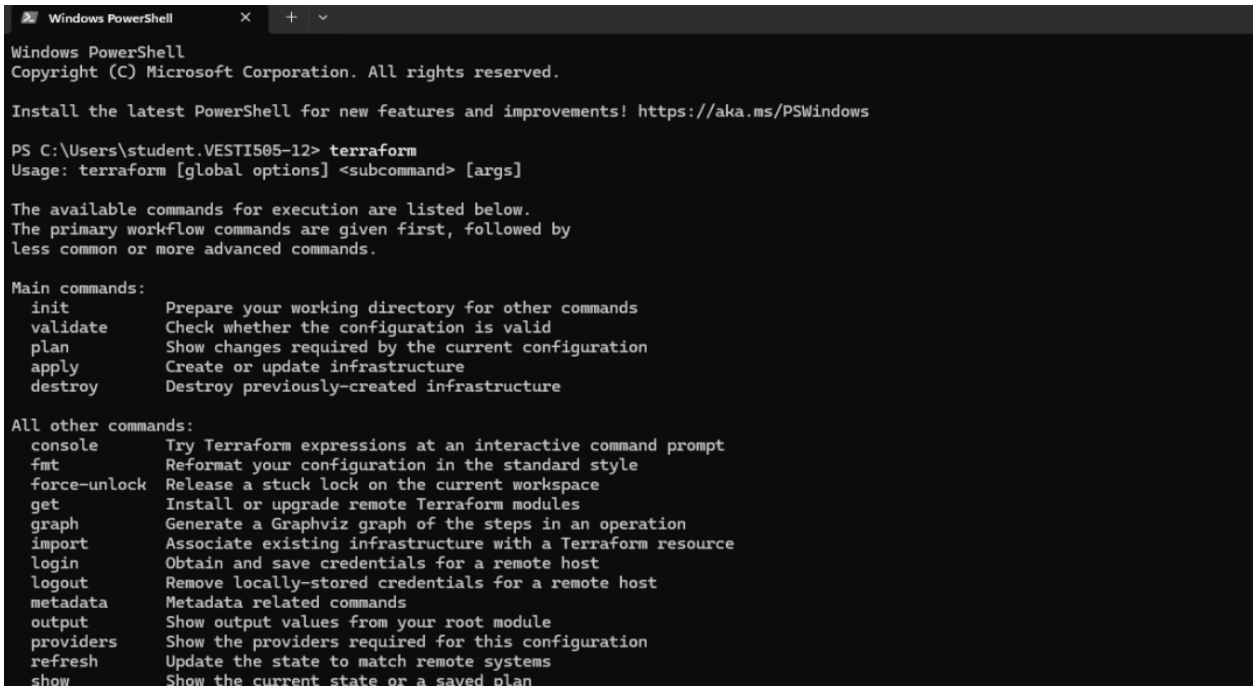
Step 4: Add the path of your Terraform folder in the PATH variable in the system environment variables inside settings.




Environment Variables
User variables for INFT505-06
Variable
Value
OneDrive
C:\Users\INFT505-06\OneDrive
Path
C:\Users\INFT505-06\AppData\Local\Programs\Python\Python312\...
TEMP
C:\Users\INFT505-06\AppData\Local\Temp
TMP
C:\Users\INFT505-06\AppData\Local\Temp
New...
Edit...
Delete
System variables
Variable
Value
ComSpec
C:\WINDOWS\system32\cmd.exe
DriverData
C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS
16
OS
Windows_NT
Path
C:\Program Files\Common Files\Oracle\Java\javapath;C:\Program ...
PATHEXT
.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE
AMD64
New...
Edit...
Delete
OK
Cancel


Edit environment variable
C:\Program Files\Common Files\Oracle\Java\javapath
C:\Program Files (x86)\VMware\VMware Player\bin\
C:\Program Files (x86)\Common Files\Oracle\Java\javapath
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\flutter\flutter\bin
C:\Program Files\nodejs\
C:\Users\Student\AppData\Local\Pub\Cache\bin
C:\Users\INFT505-06\AppData\Local\Programs\Python\Python311
C:\Program Files\Git\cmd
C:\Program Files\Git\bin
C:\Terraform
New
Edit
Browse...
Delete
Move Up
Move Down
Edit text...
OK
Cancel

Step 5: After opening Windows PowerShell, type the command ‘terraform’. If it is properly extracted and path is added to system variables, then following is the result we would be able to see. It will show us numerous other commands and options that we can try out so as to work with Terraform.



EXPERIMENT NO. 6

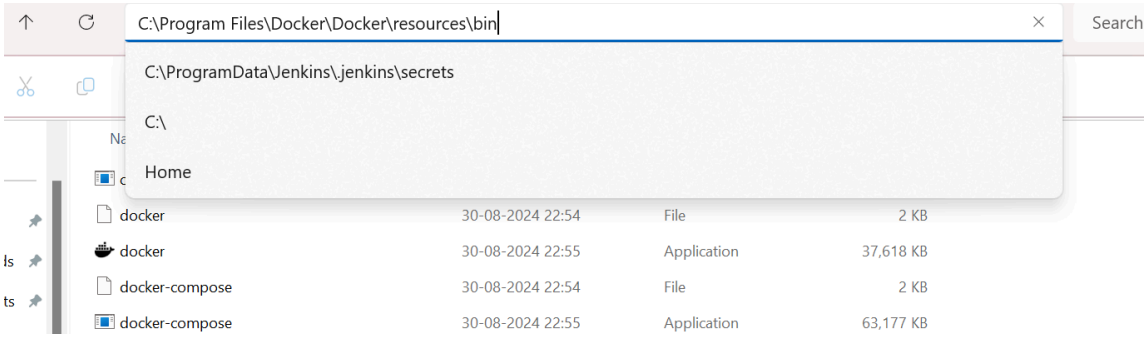
Aim: To Build, change, and destroy AWS / GCP /Microsoft Azure/ DigitalOcean infrastructure Using Terraform. (S3 bucket or Docker) fdp.

Step 1: For this experiment, you need to install docker on your computer. Go to https://www.docker.com/ and download the file according to the OS you have. Open the file and start the installation. Once installed, open your terminal and run ‘docker’ command. If this is your output, then docker is installed successfully.

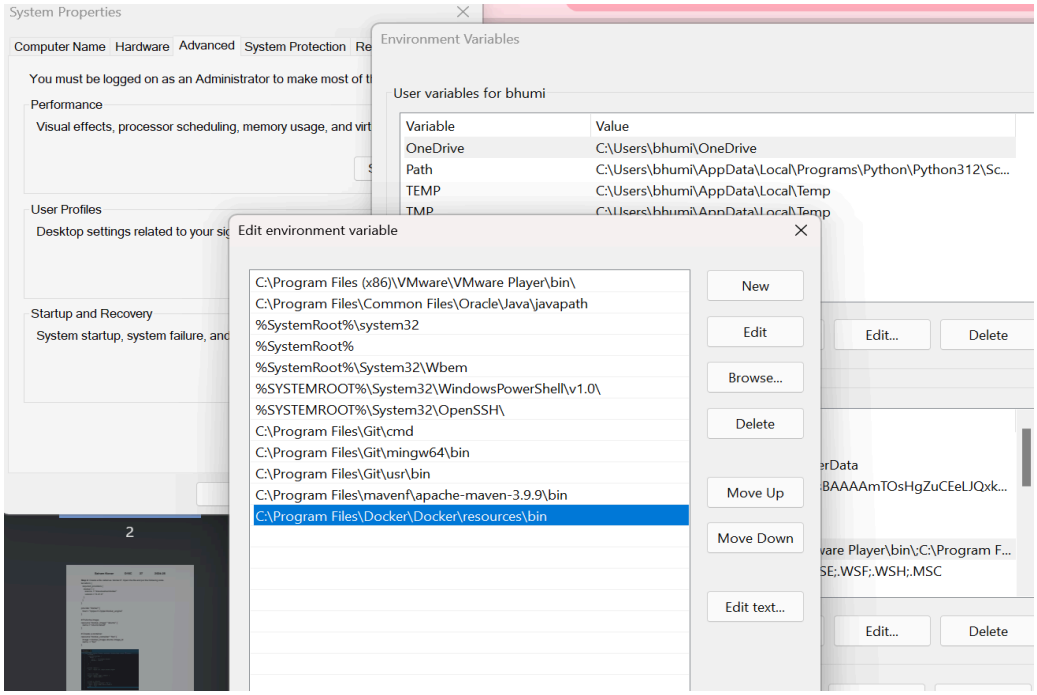
```
C:\Users\bhumi>docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

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

Management Commands:
  builder     Manage builds
  buildx*     Docker Buildx
  compose*    Docker Compose
  container   Manage containers
  context     Manage contexts
  debug*      Get a shell into any image or container
  desktop*    Docker Desktop commands (Alpha)
  dev*        Docker Dev Environments
```

If you det an error like ‘docker is not an internal or external command’, you need to add the bin path of docker to your environment variables.

Go to File Explorer, and follow this path: C drive → Program Files → Docker → Docker → Resources → bin. Copy this path by clicking on the bar having the path and using shortcut CTRL + C



Open ‘Edit the System Environment Variables’ on your system. Click on Environment Variables. Now, check for a ‘Path’ variable under System variables, if it exists, click on it, then click on edit. Else, click on New and add the variable ‘Path’. If the variable existed, click on Edit, then on New. This will give you a text box. Paste the path you copied here and click on ok until you close all the tabs.

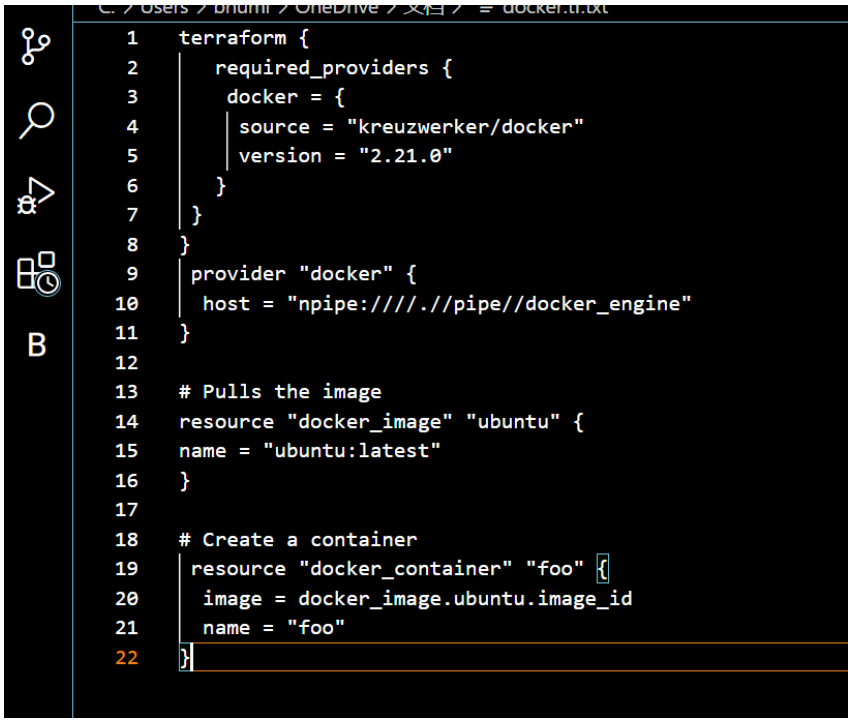


Now run the docker command again and the output would appear.
Alternatively, you could also run ‘docker –version’ to check whether docker is started on the terminal.

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

Step 2: Create a file called ‘docker.tf’. Open the file and put the following code.

```
terraform {
  required_providers {
    docker = {
      source = "kreuzwerker/docker"
      version = "2.21.0"
    }
  }
}
provider "docker" {
  host = "npipe:////.//pipe//docker_engine"
}

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

# Create a container resource
"docker_container" "foo" {
image = docker_image.ubuntu.image_id
name = "foo"
}
```



Step 3: Open the folder where the docker.tf is present on your terminal. Execute the command ‘terrafom init’. This will initialize terraform in the directory.

```
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: Run the command ‘terraform plan’. This creates an execution plan and lets you overview changes that are going to happen in your infrastructure.

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

Terraform will perform the following actions:

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

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

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

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

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

  Enter a value: yes

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

Docker images, Before Executing Apply step:

```
C:\Users\2022k\OneDrive\Desktop\Docker\Terraform Scripts>docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```

Docker images, After Executing Apply step:

```
C:\Users\2022k\OneDrive\Desktop\Docker\Terraform Scripts>docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    edbfe74c41f8   3 weeks ago   78.1MB
```

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

```
C:\Users\2022k\OneDrive\Desktop\Docker\Terraform Scripts>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]

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_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, 1 to destroy.

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

  Enter a value: yes

docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image.ubuntu: Destruction complete after 1s

Destroy complete! Resources: 1 destroyed.
```

Docker images After Executing Destroy step.

```
C:\Users\2022k\OneDrive\Desktop\Docker\Terraform Scripts>docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```

EXPERIMENT NO. 7

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

Prerequisites:

1) Docker

Run docker -v command.

Use this command to check if docker is installed and running on your system.

```
C:\Users\bhumi>docker -v
Docker version 27.2.0, build 3ab4256
```

2) Install SonarQube image Command:

docker pull sonarqube

This command helps you to install an image of SonarQube that can be used on the local system without actually installing the SonarQube installer.

```
C:\Users\bhumi>docker pull sonarqube
Using default tag: latest
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
docker.io/library/sonarqube:latest
```

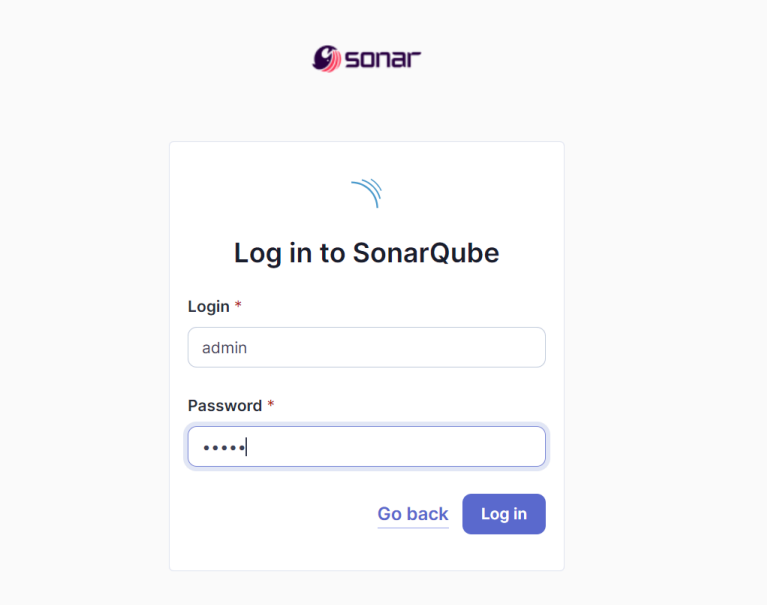
3) Keep jenkins installed on your system.

Experiment Steps:

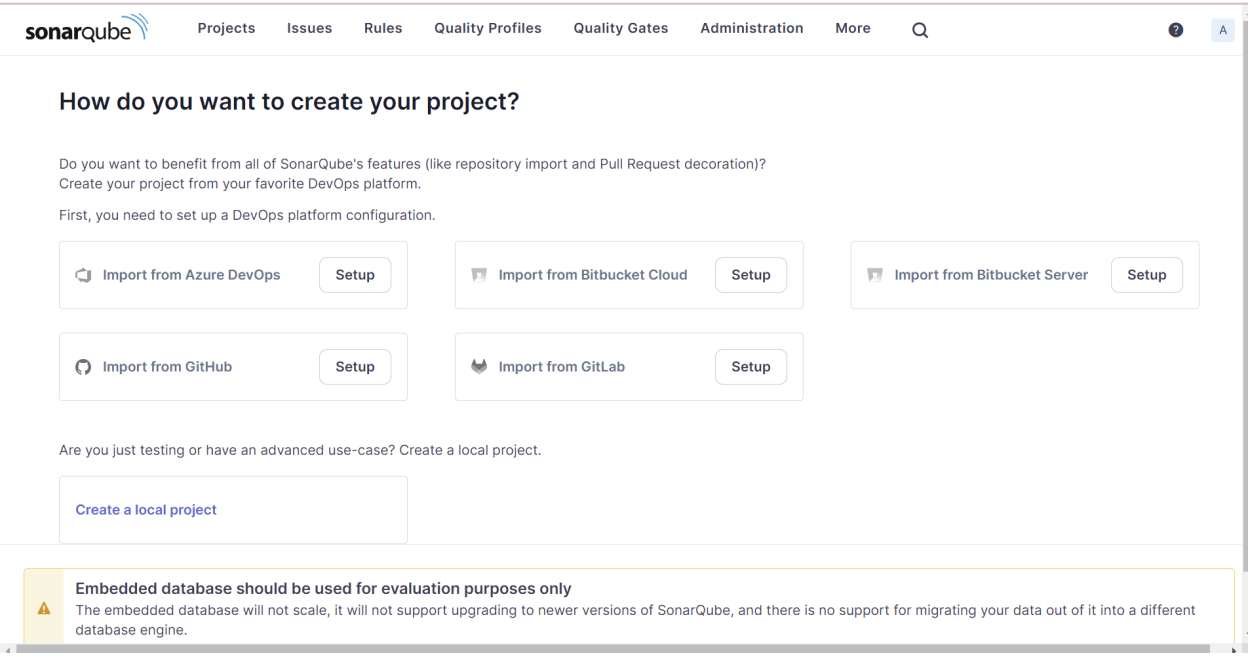
1. Run SonarQube in a Docker container using this command -
   docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

```
C:\Users\bhumi>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
7401befce9e7a7248c0e5648a2913e99c3843cce08e91d403e5af3a1479a151e
```

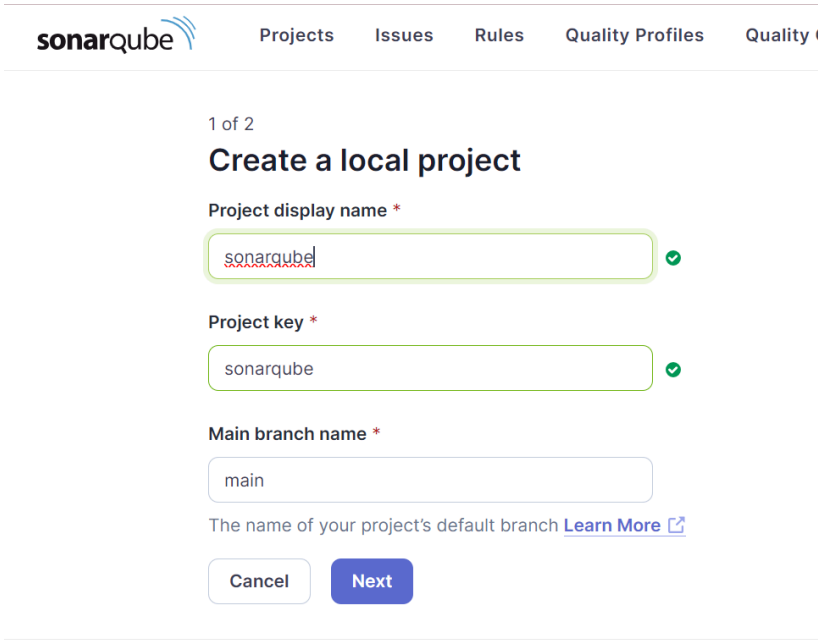
2. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.
3. Login to SonarQube using username admin and password admin.



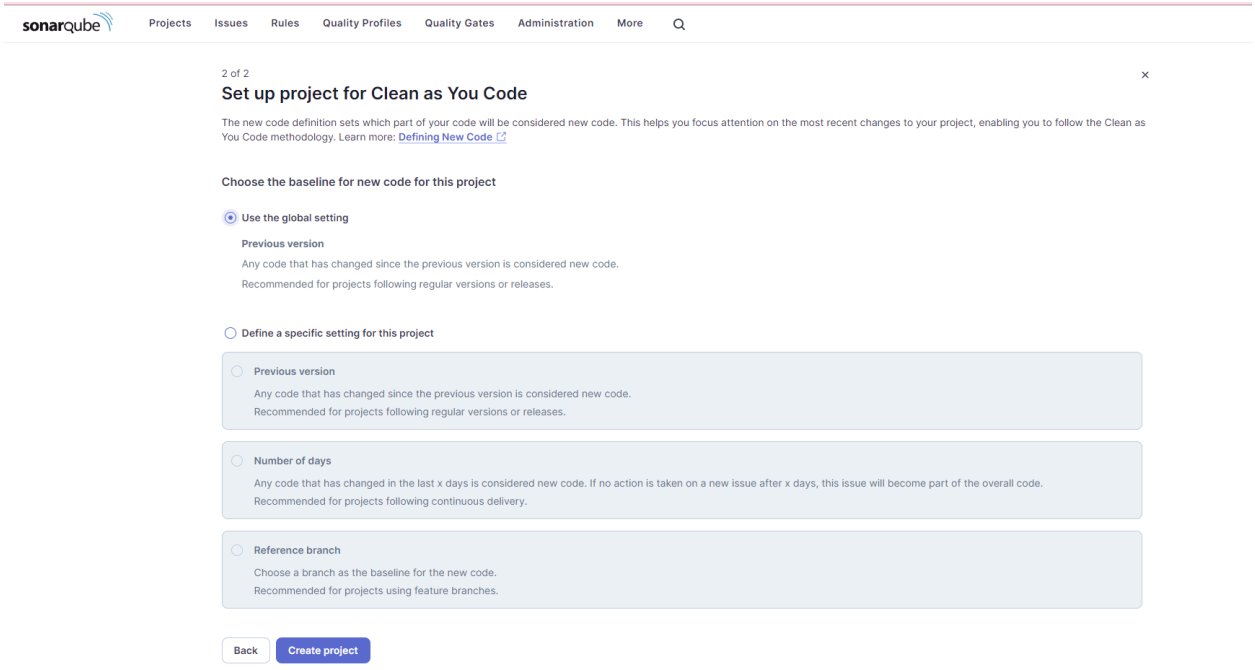
4. Create a manual project in SonarQube with the name sonarqube.



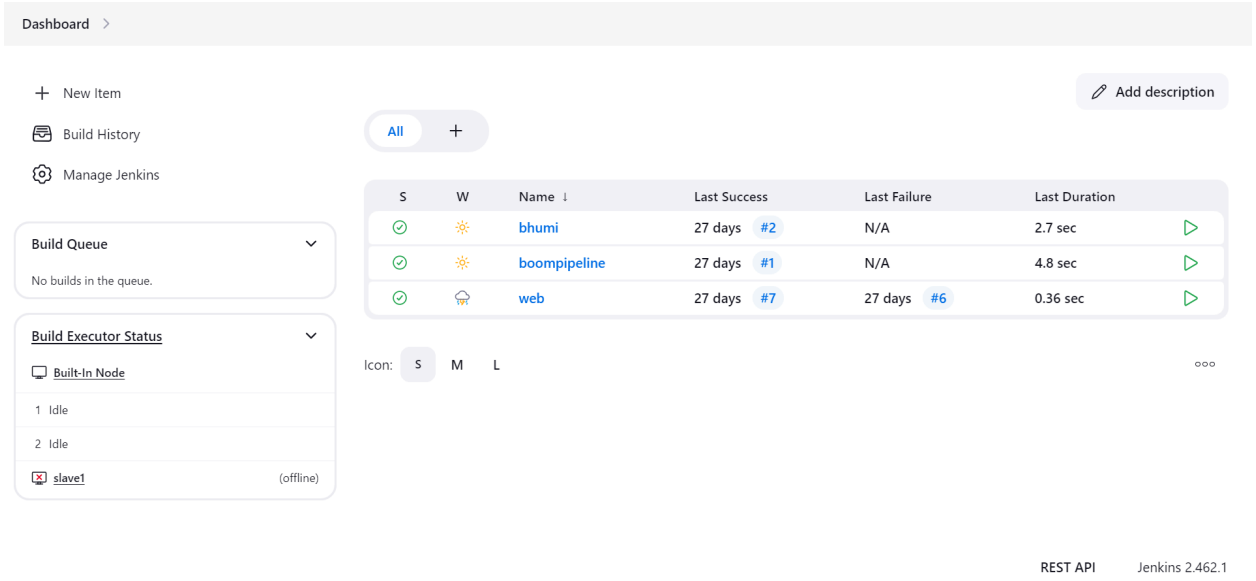
Click on Create a Local Project.
Here, I have given the name ‘sonarqube’. This will be your project key as well. We’ll use it later.
Keep the branch main only and click on next.



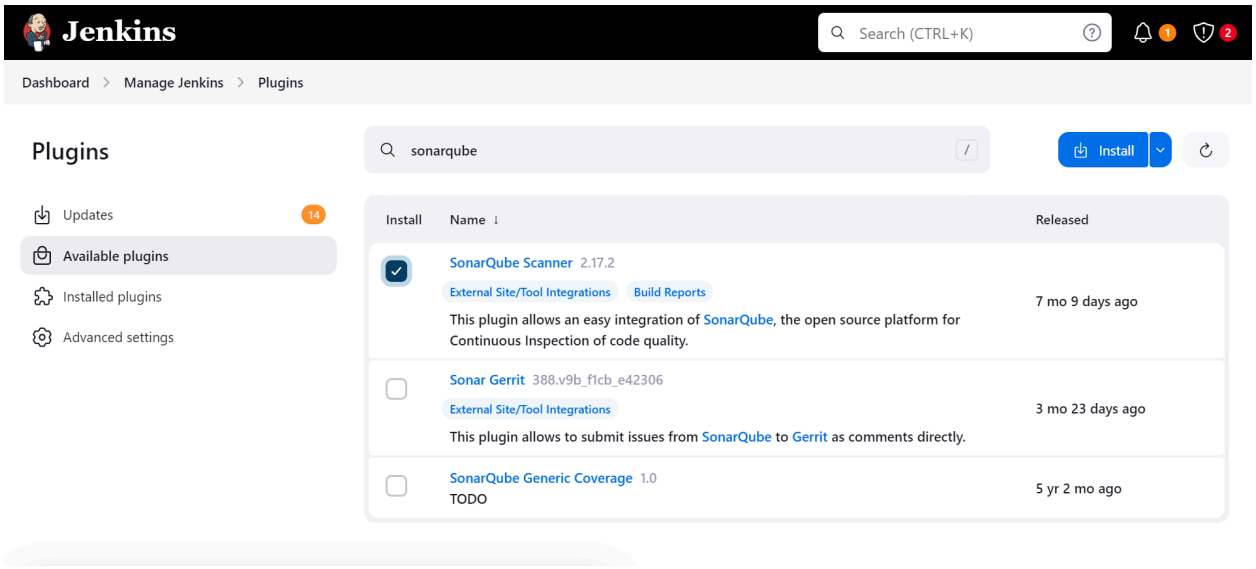
Set up the project as required and click on create.



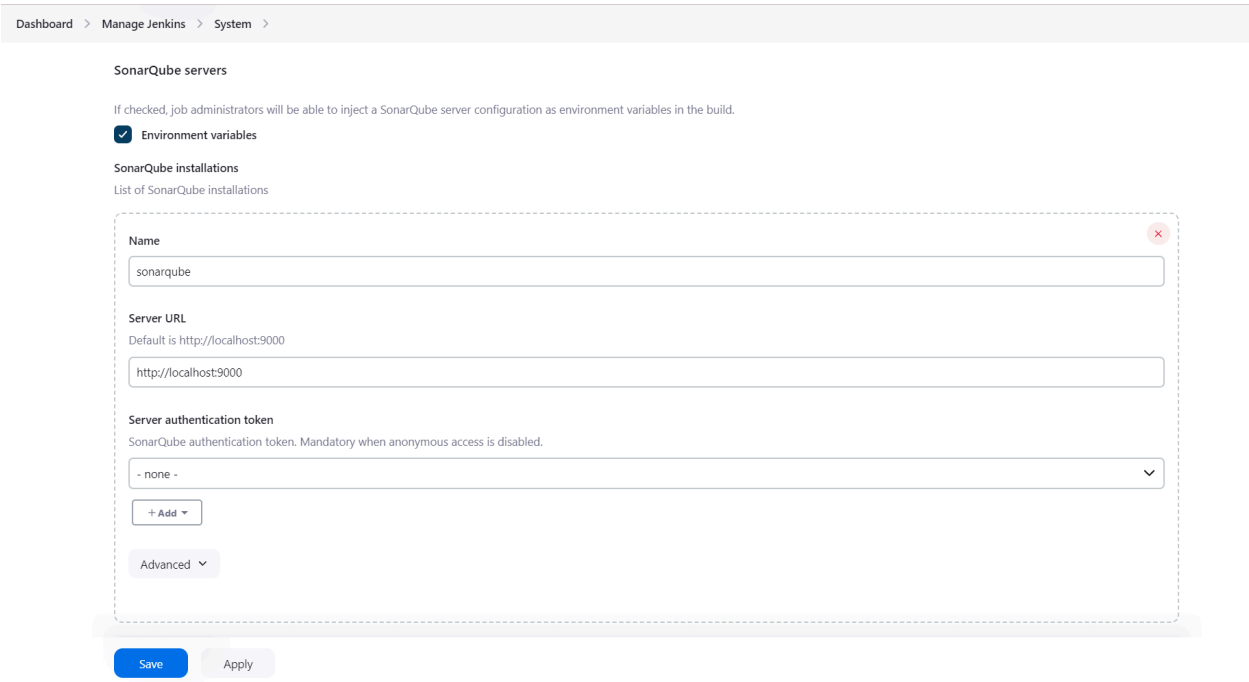
5. Open Jenkins on whichever port it is installed. (http://loaclhost:<port number>).



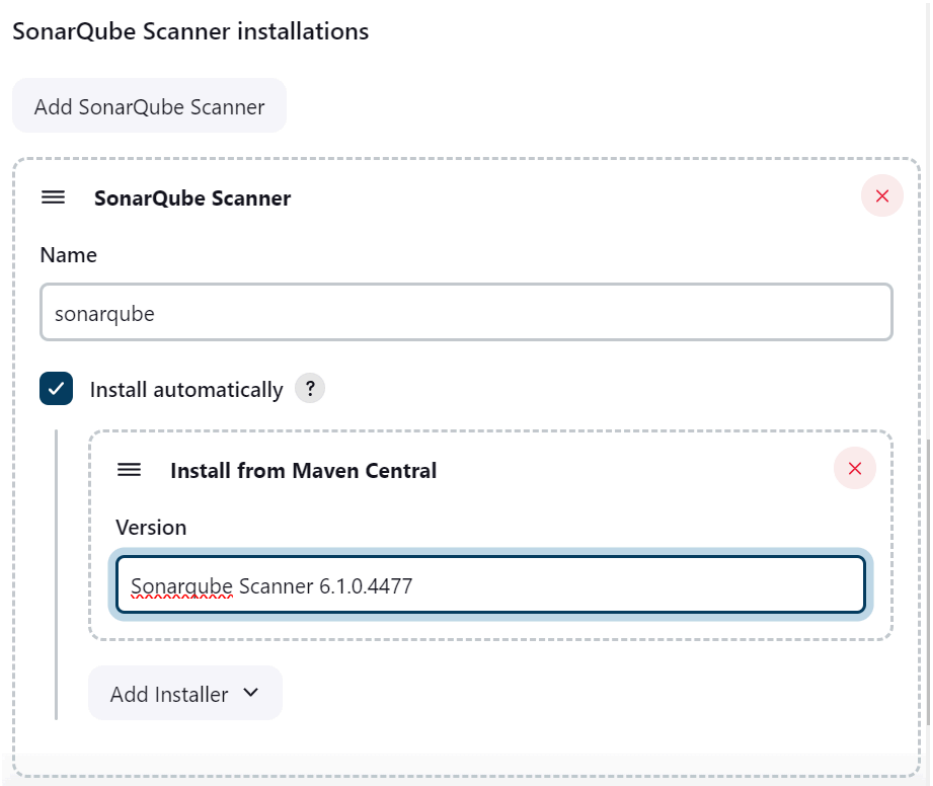
6. Go to Manage jenkins → Plugins → Available Plugins
Search for Sonarqube Scanner for Jenkins and install it.



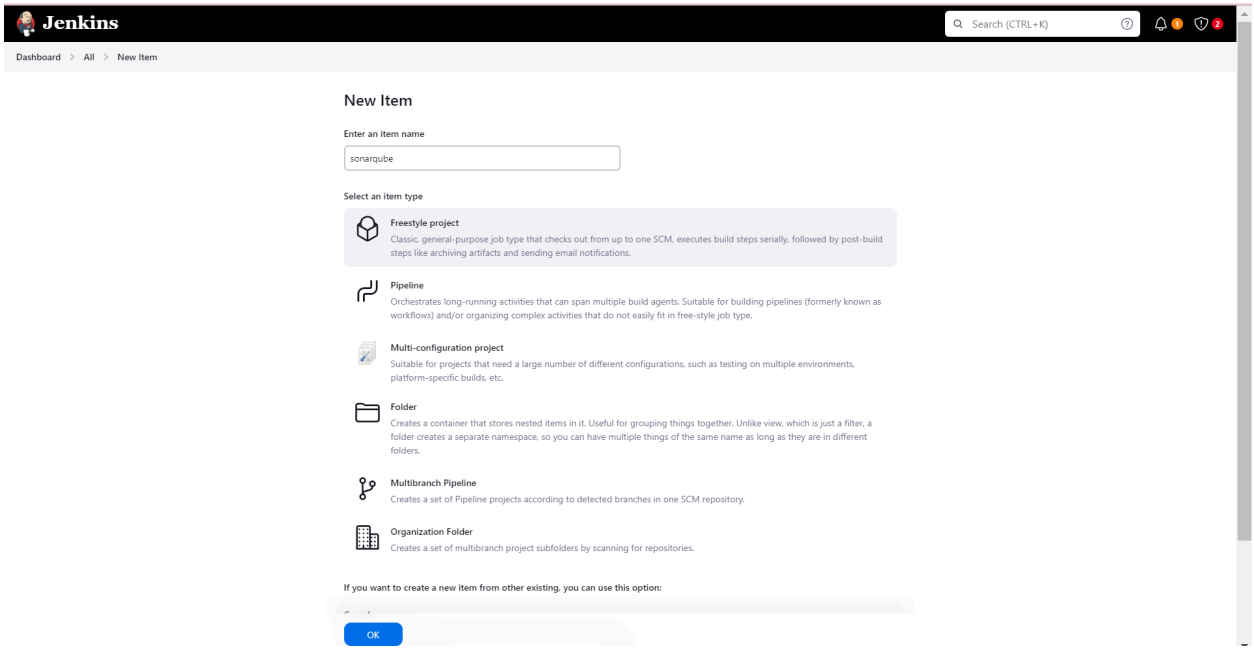
7. Under Jenkins ‘Configure System’, look for SonarQube Servers and enter the details.
I have named the server ‘sonarqube’ and added the server url for jenkins.
Enter the Server Authentication token if needed.



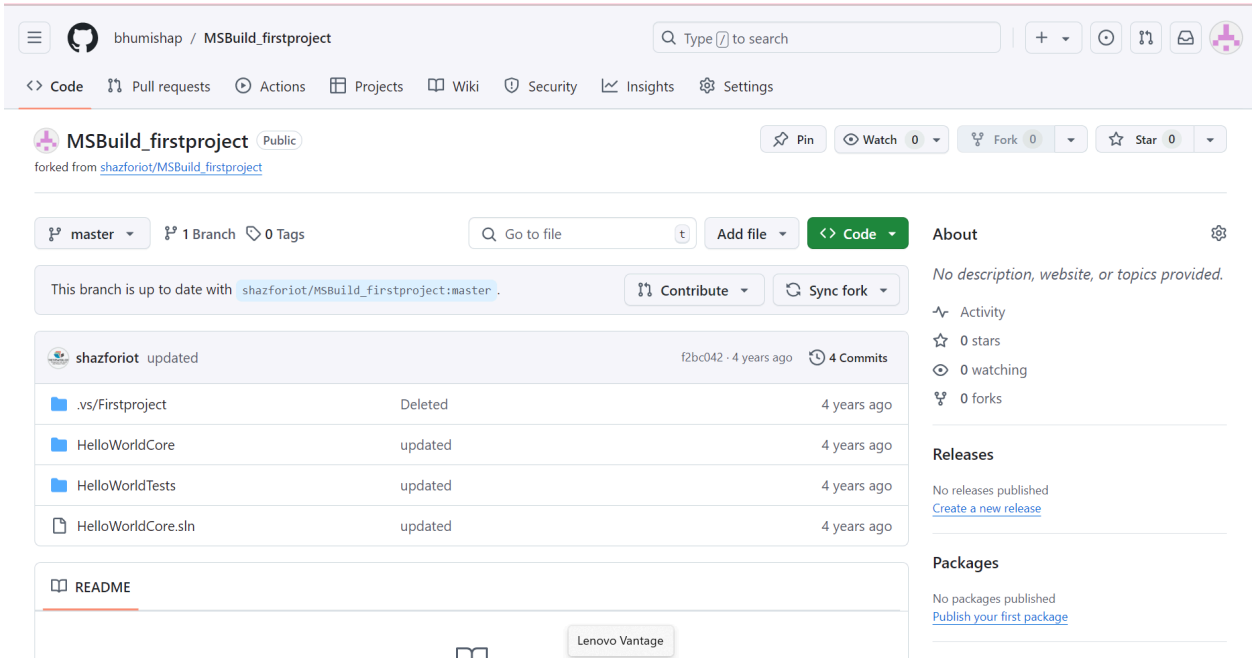
8. Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.



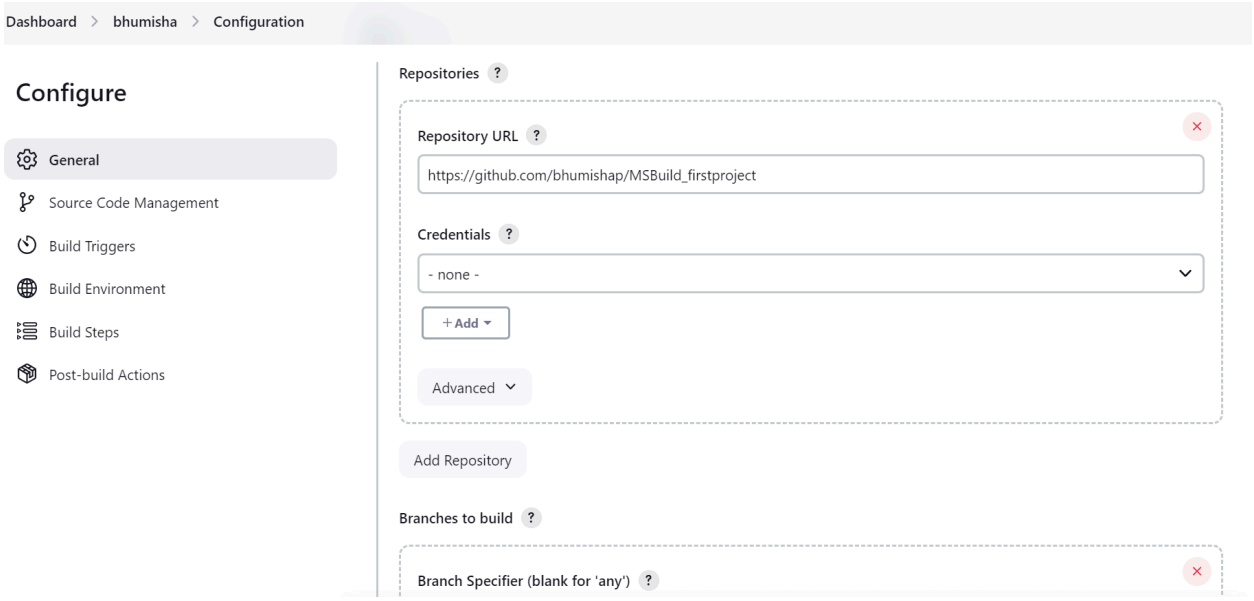
9. After configuration, create a New Item → choose a freestyle project and name your project. Here, I have given the ‘sonarqube’, then click on OK.



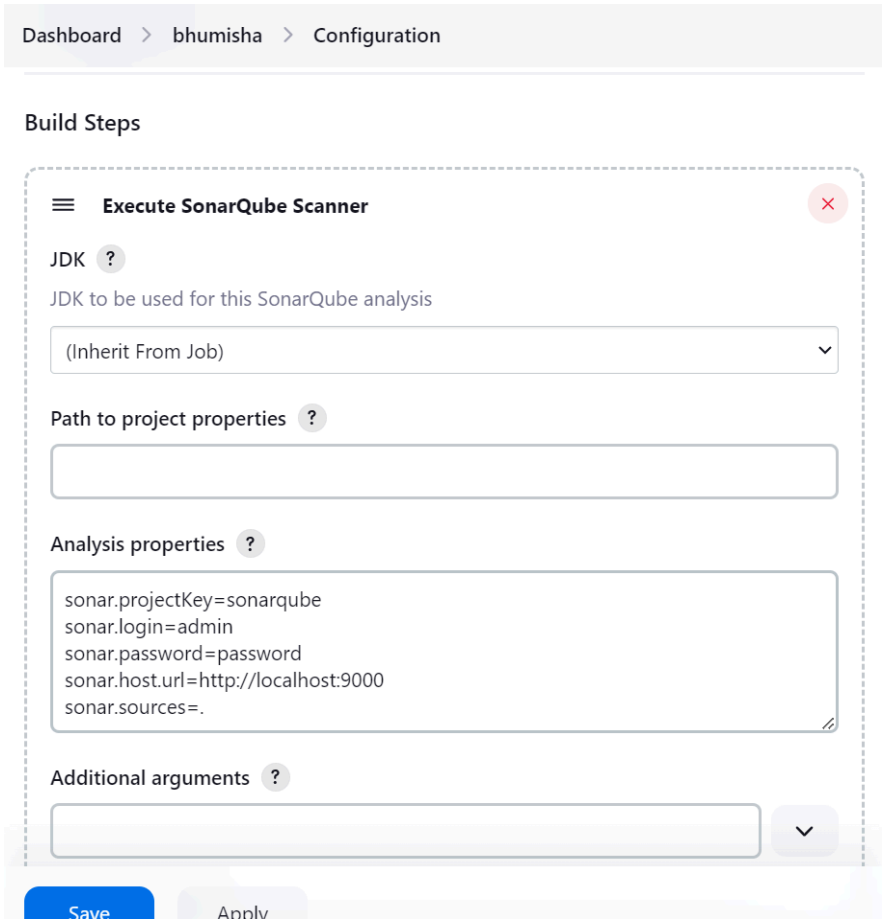
10. Choose the github repository https://github.com/shazforiot/MSBuild_firstproject.
    It is a sample hello-world project with no vulnerabilities and issues, just to test the integration.
    Fork this repository.



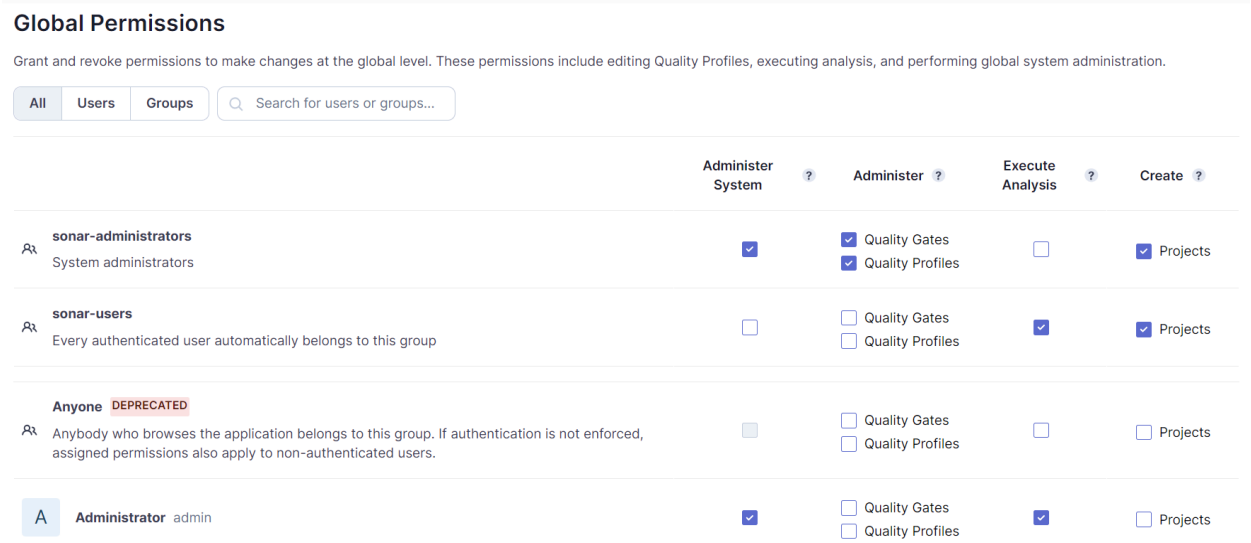
11. Enable git and add the repository you forked.



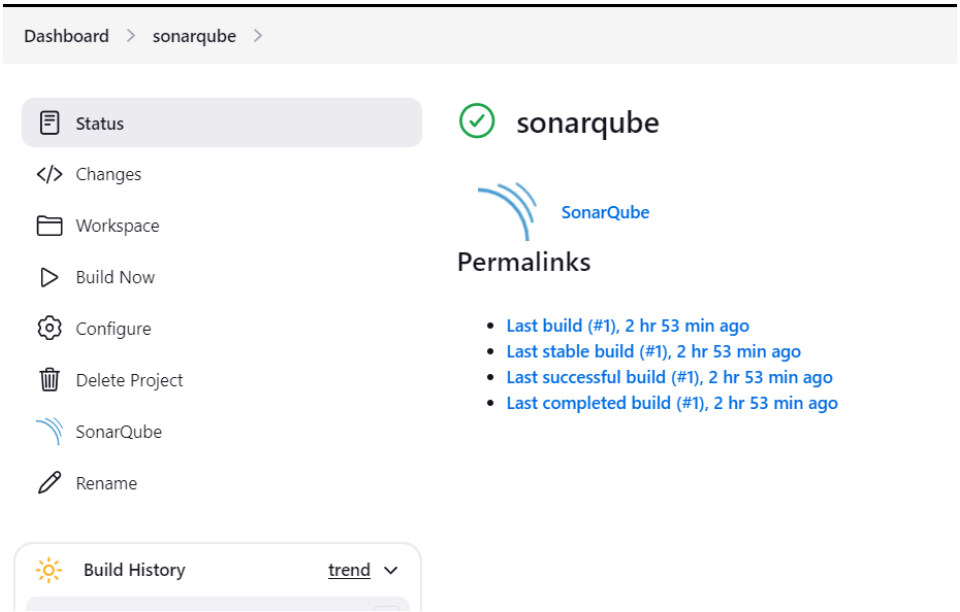
12. Under Build Steps, select Execute Sonarqube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.



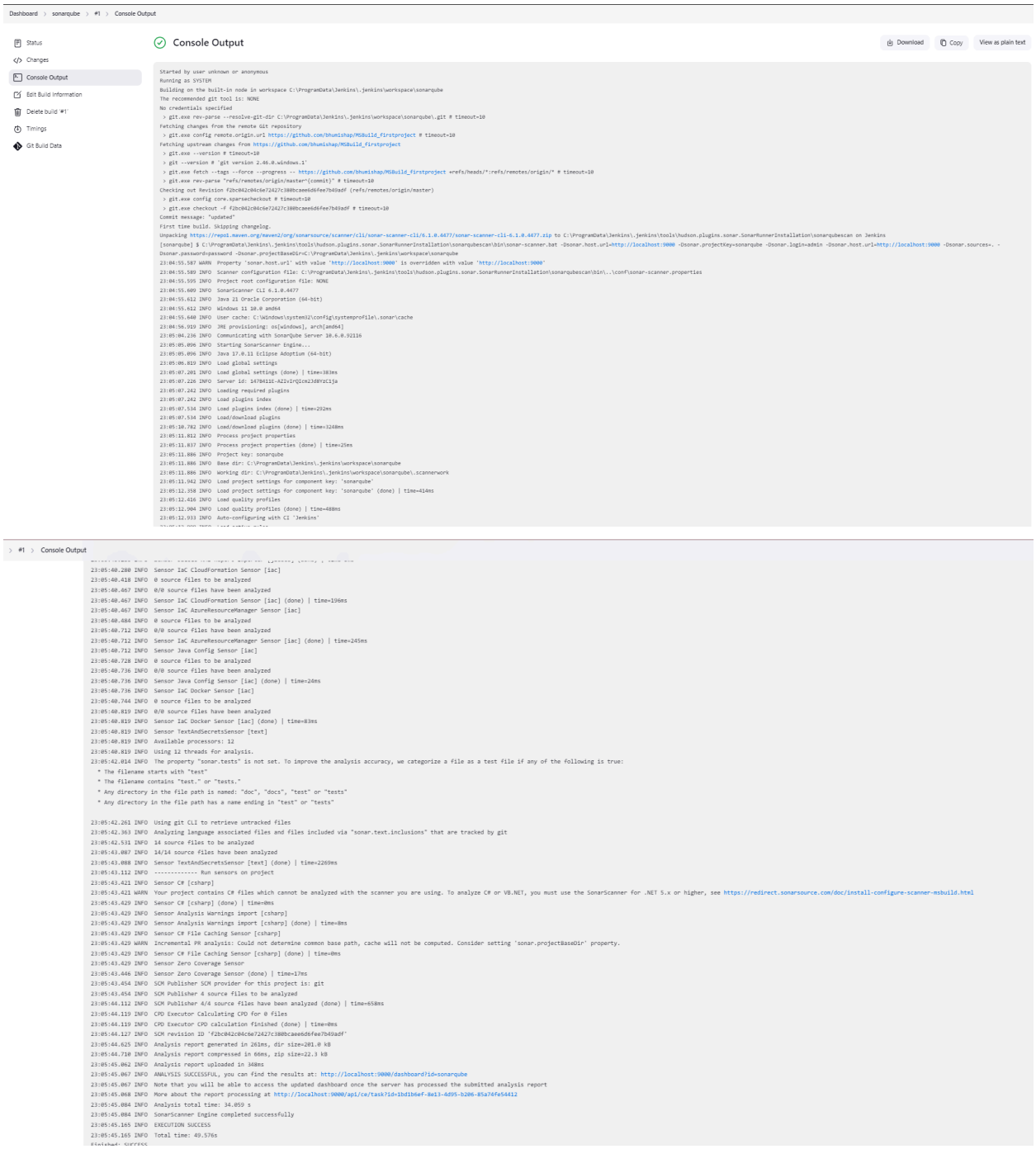
13. Then click on save.Go to http://localhost:9000/<user_name>/permissions and allow Execute Permissions to the Admin user.



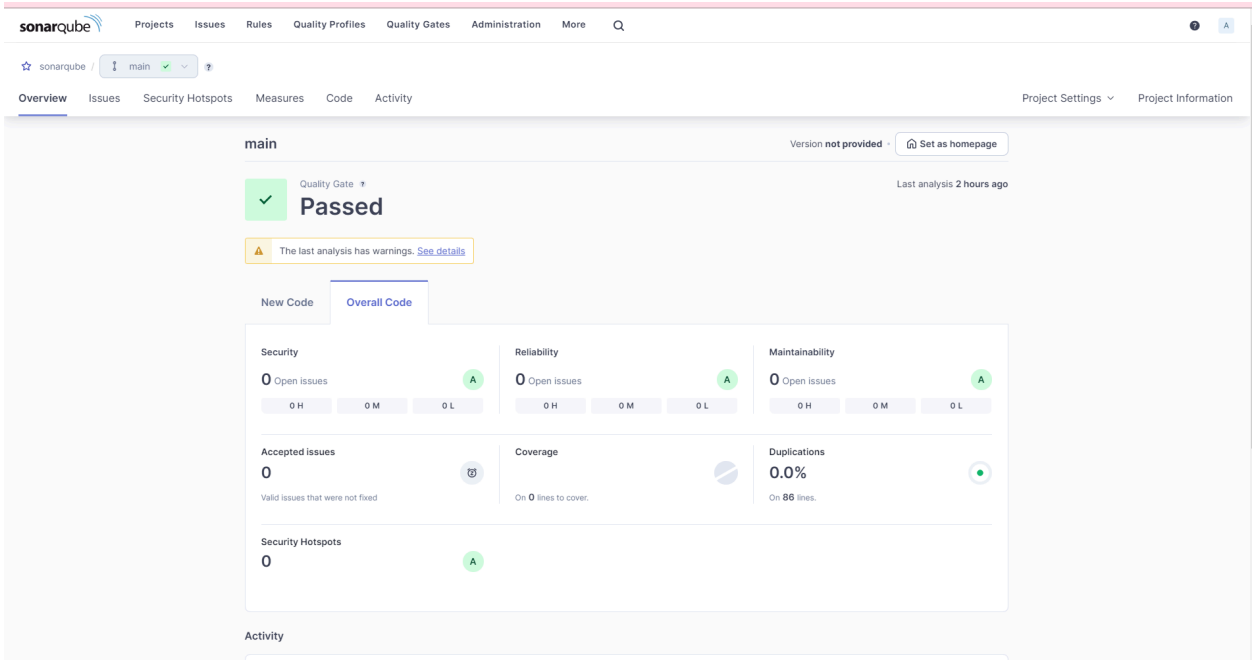
14. Go back to jenkins. Go to the job you had just built and click on Build Now.

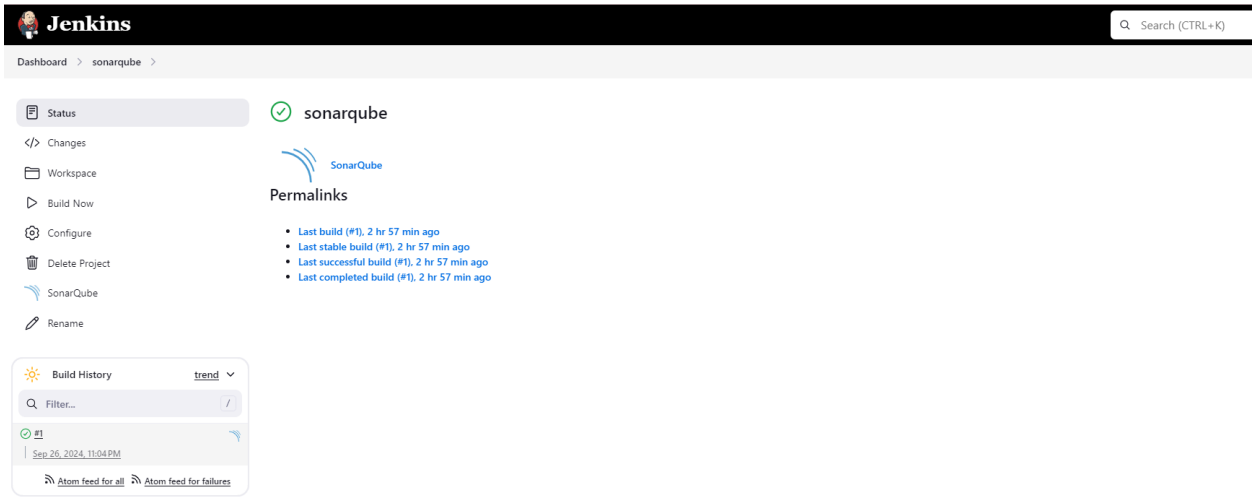


15. Check the console Output.



16. Once the build is complete, go back to SonarQube and check the project linked.





Conclusion:

In this experiment, we explored how to perform Static Application Security Testing (SAST) in Jenkins using SonarQube. Instead of installing SonarQube directly on the local system, we utilized a Docker image for a more streamlined setup. After configuring Jenkins with the necessary SonarQube settings, we integrated it with a GitHub repository containing code for analysis. Upon building the project, SonarQube successfully scanned the code, confirming that no errors were present. This process demonstrated how Jenkins and SonarQube can be effectively used together for automated code quality and security analysis.

EXPERIMENT NO. 8

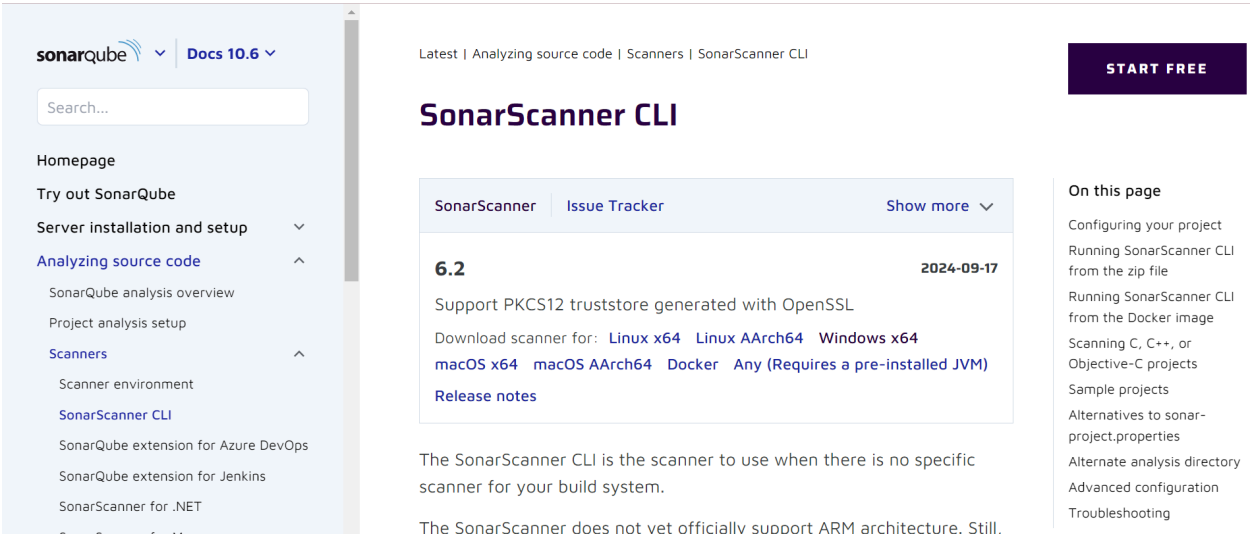
Aim: Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web / Java / Python application.

Prerequisites:

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

```
C:\Users\bhumi>docker -v
Docker version 27.2.0, build 3ab4256

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

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

- Download sonar scanner



Extract the downloaded zip file in a folder.

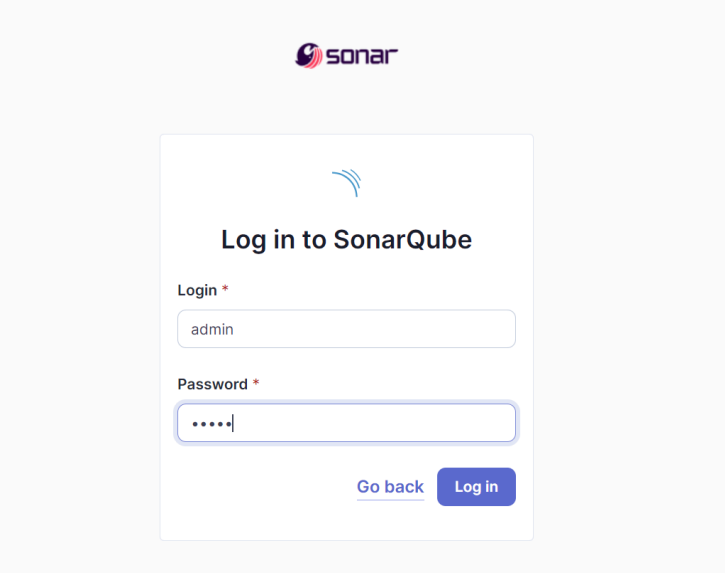
Steps to create a Jenkins CI/CD Pipeline and use SonarQube to perform SAST

1. Run SonarQube image

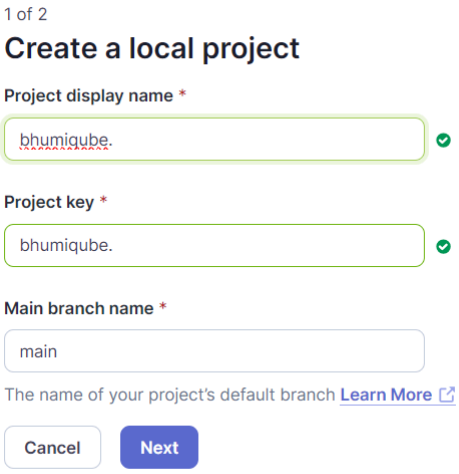
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
This command will run the SonarQube image that was just installed using docker.

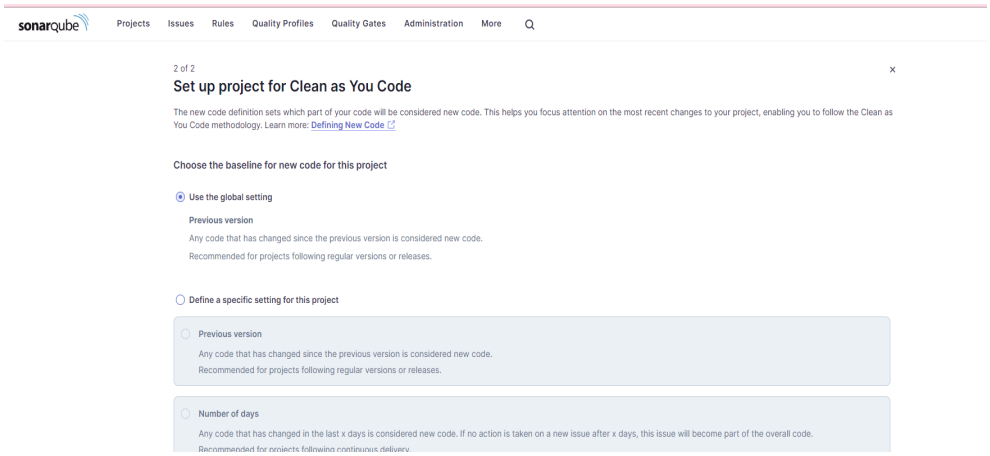
```
C:\Users\bhumi>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
7401befce9e7a7248c0e5648a2913e99c3843cce08e91d403e5af3a1479a151e
```

2. Once the SonarQube image is started, you can go to http://localhost:9000 to find the SonarQube that has started.
3. Login to SonarQube using username admin and password admin.

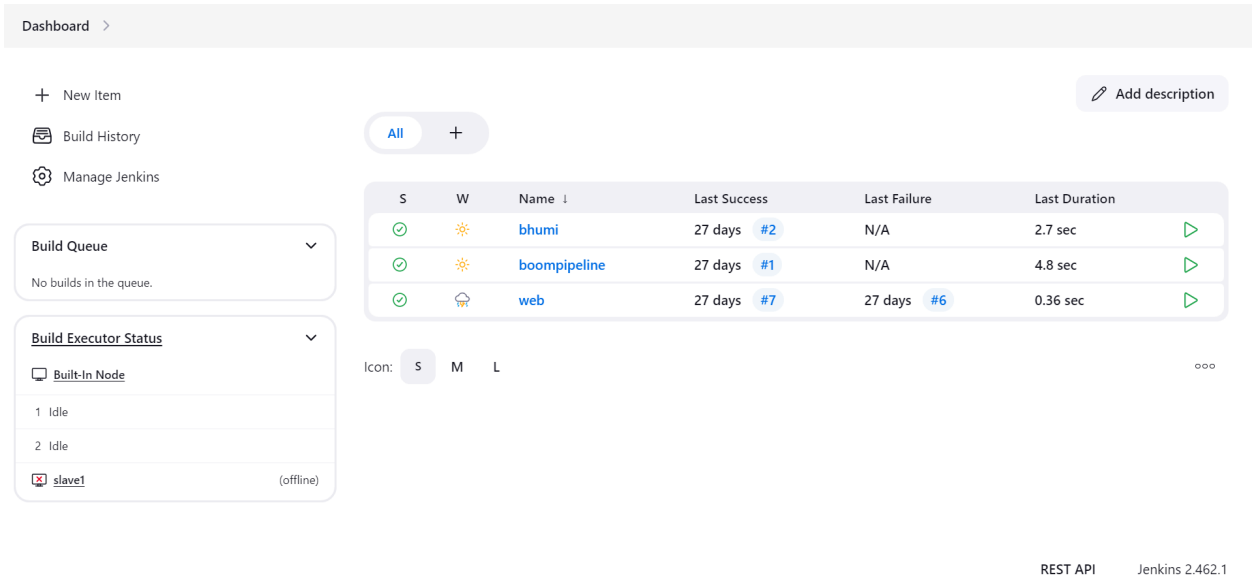


4. Create a local project in SonarQube and enter a name.
Here I have given the name ‘bhumiqube’ which is also the project key.

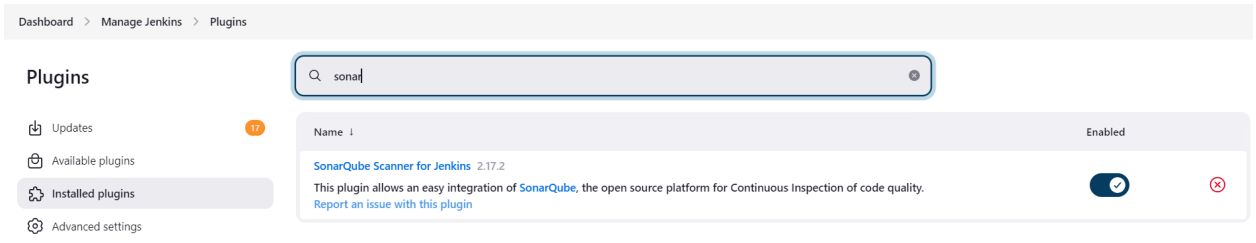




5. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.

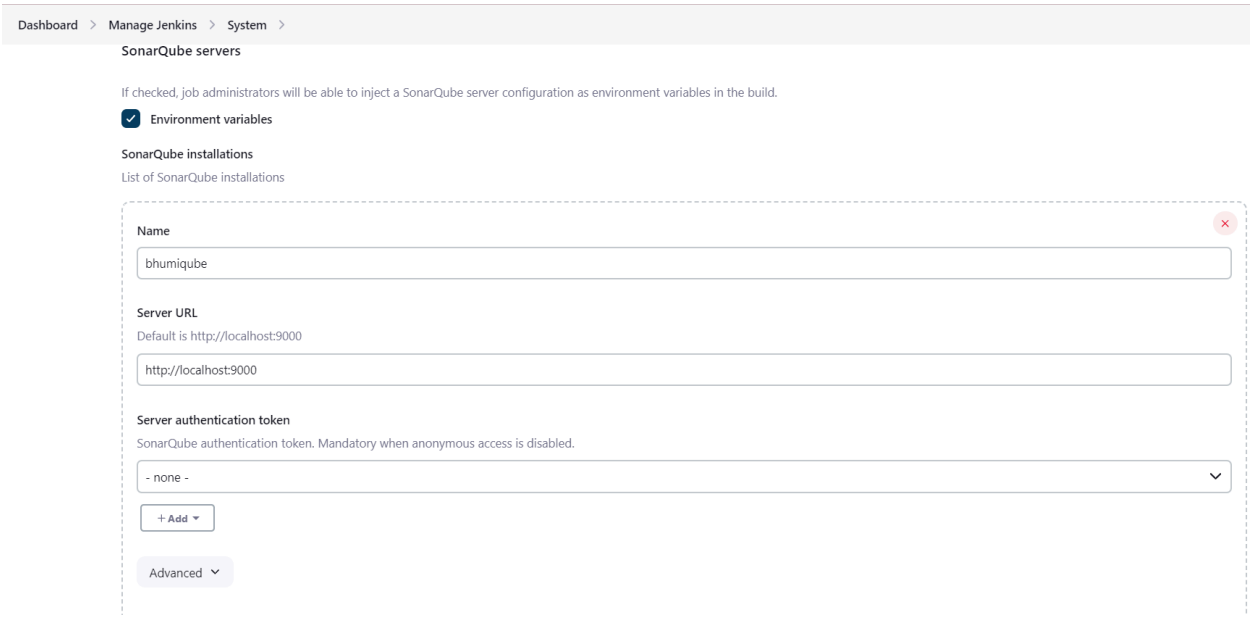


6. Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.(we already installed it for exp 7 so you can skip)

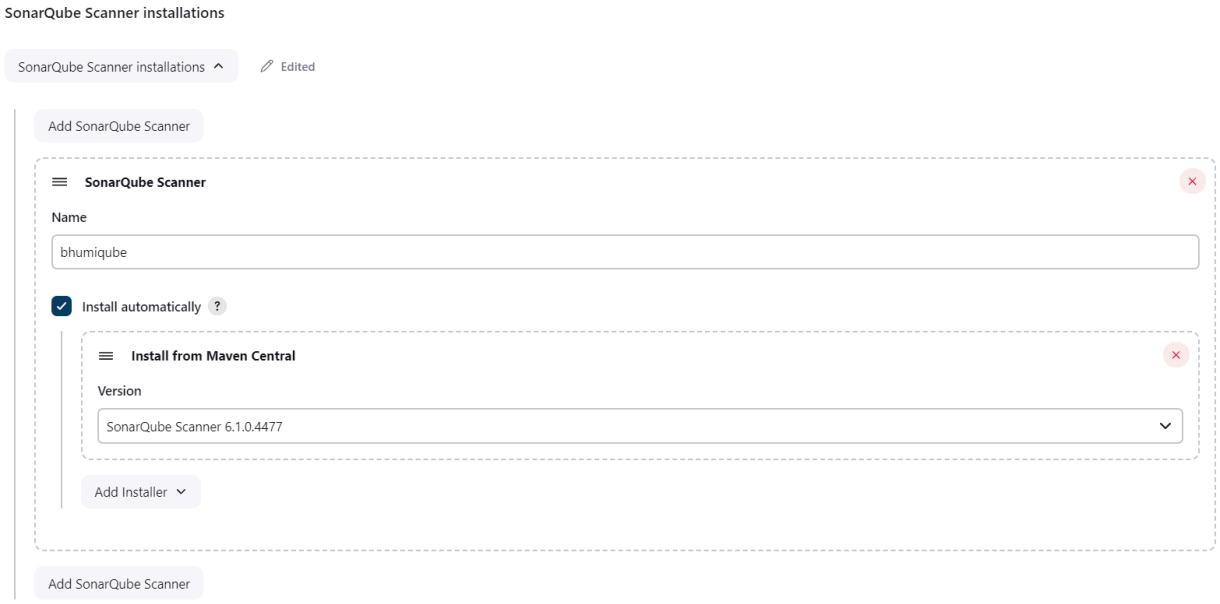


7. Under Jenkins ‘Manage Jenkins’ then go to ‘system’, scroll and look for SonarQube Servers and enter the details.

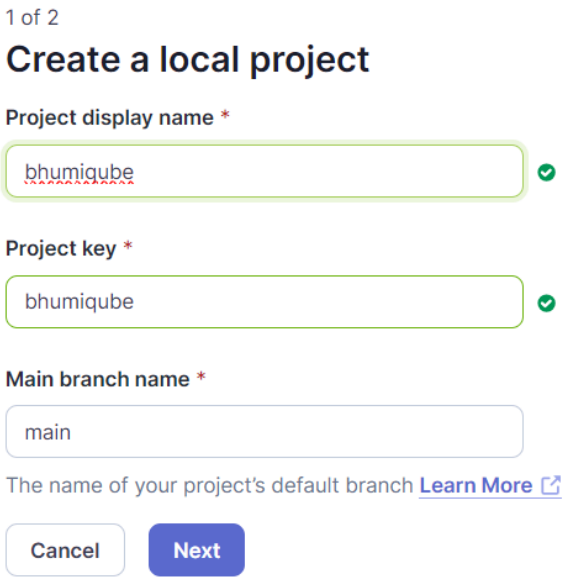
In SonarQube installations: Under Name add <project name of sonarqube> which is ‘bhumiqube’ for me. In Server URL Default is http://localhost:9000.



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



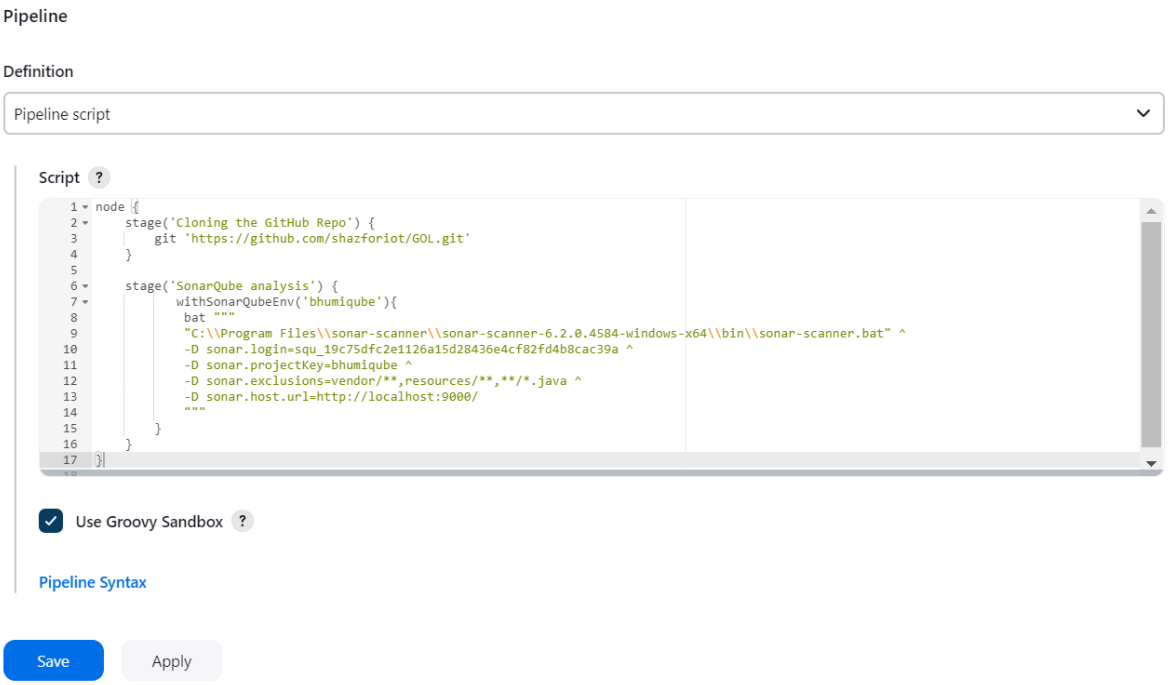
9. After configuration, create a New Item → choose a pipeline project.



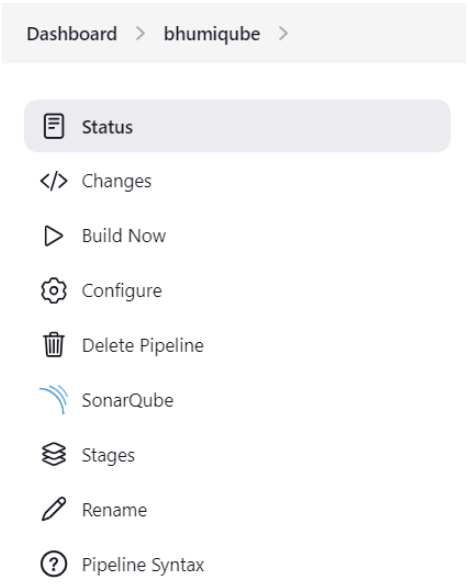
10. Under Pipeline script, enter the following:

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

    stage('SonarQube analysis') {
          withSonarQubeEnv('bhumiqube'){
           bat """
           "C:\\Program
Files\\sonar-scanner\\sonar-scanner-6.2.0.4584-windows-x64\\bin\\sonar-scanne
r.bat" ^
           -D sonar.login=squ_19c75dfc2e1126a15d28436e4cf82fd4b8cac39a ^
           -D sonar.projectKey=bhumiqube ^
           -D sonar.exclusions=vendor/**,resources/**,**/*.java ^
           -D sonar.host.url=http://localhost:9000/
           """
        }
    }
}
```

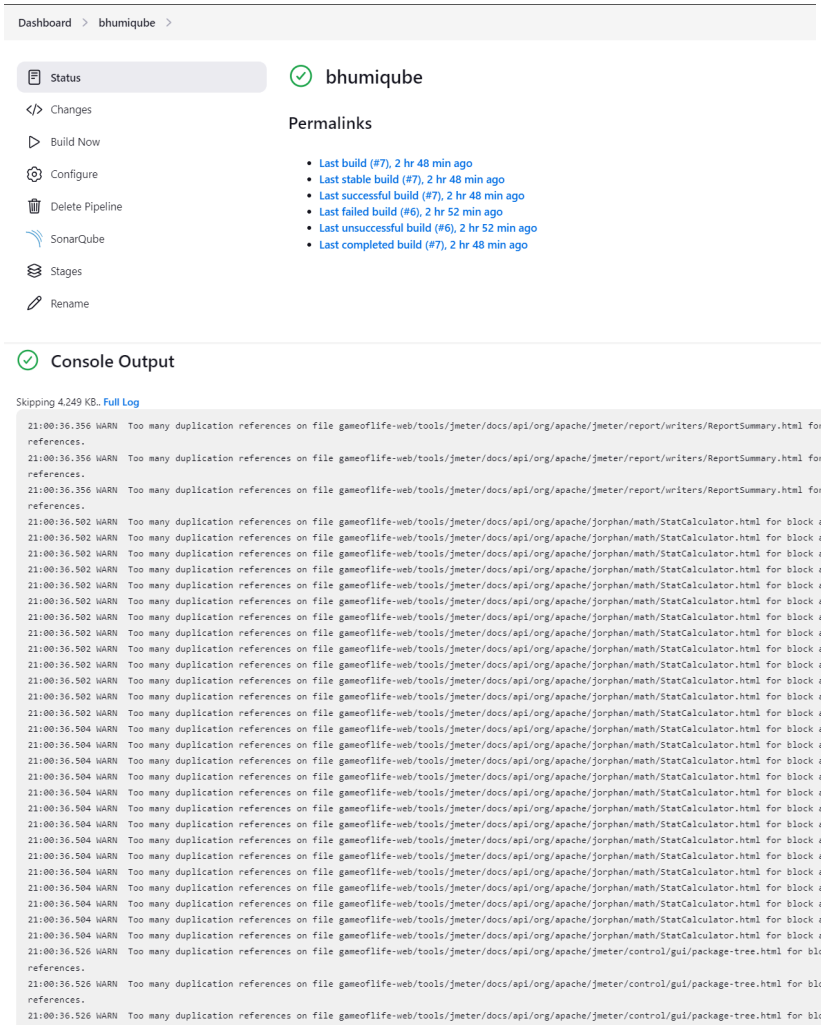
It is a java sample project which has a lot of repetitions and issues that will be detected by SonarQube.



11. Go to the job you had just built and click on Build Now.

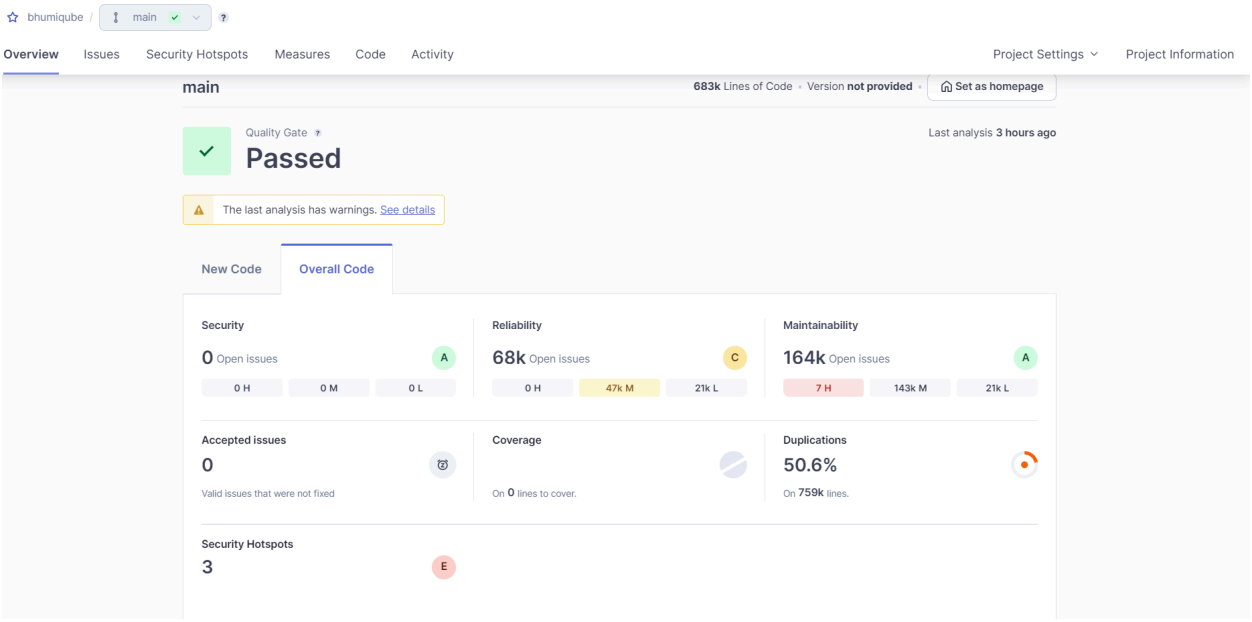


12. Once it is built, check the console output.



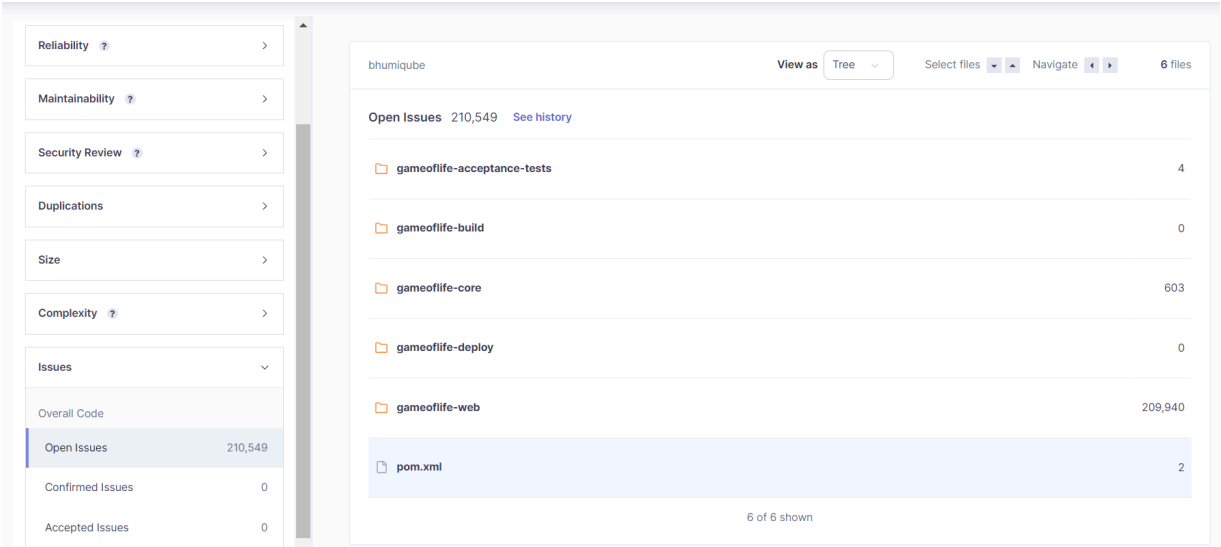

```
21:00:42.643 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/ldap/control/gui/LdapExtTestSamplerGui.html for
block at line 155. Keep only the first 100 references.
21:00:42.643 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/ldap/control/gui/LdapExtTestSamplerGui.html for
block at line 515. Keep only the first 100 references.
21:00:42.643 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/ldap/control/gui/LdapExtTestSamplerGui.html for
block at line 768. Keep only the first 100 references.
21:00:42.643 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/ldap/control/gui/LdapExtTestSamplerGui.html for
block at line 714. Keep only the first 100 references.
21:00:42.643 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/ldap/control/gui/LdapExtTestSamplerGui.html for
block at line 668. Keep only the first 100 references.
21:00:42.643 INFO  CPD Executor CPD calculation finished (done) | time=187786ms
21:00:42.660 INFO  SCM revision ID 'ba799ba7e1b576f04a4612322b0412c5e6e1e5e4'
21:02:50.519 INFO  Analysis report generated in 5916ms, dir size=127.2 MB
21:03:11.285 INFO  Analysis report compressed in 20750ms, zip size=29.6 MB
21:03:11.715 INFO  Analysis report uploaded in 430ms
21:03:11.717 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=bhumiqube
21:03:11.717 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
21:03:11.717 INFO  More about the report processing at http://localhost:9000/api/ce/task?id=2bf5a9f3-919a-4725-94d5-474d4773e610
21:03:26.163 INFO  Analysis total time: 16:37.760 s
21:03:26.163 INFO  SonarScanner Engine completed successfully
21:03:26.916 INFO  EXECUTION SUCCESS
21:03:26.916 INFO  Total time: 16:42.318s
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

13. Once the build is complete, go back to SonarQube and check the project linked.

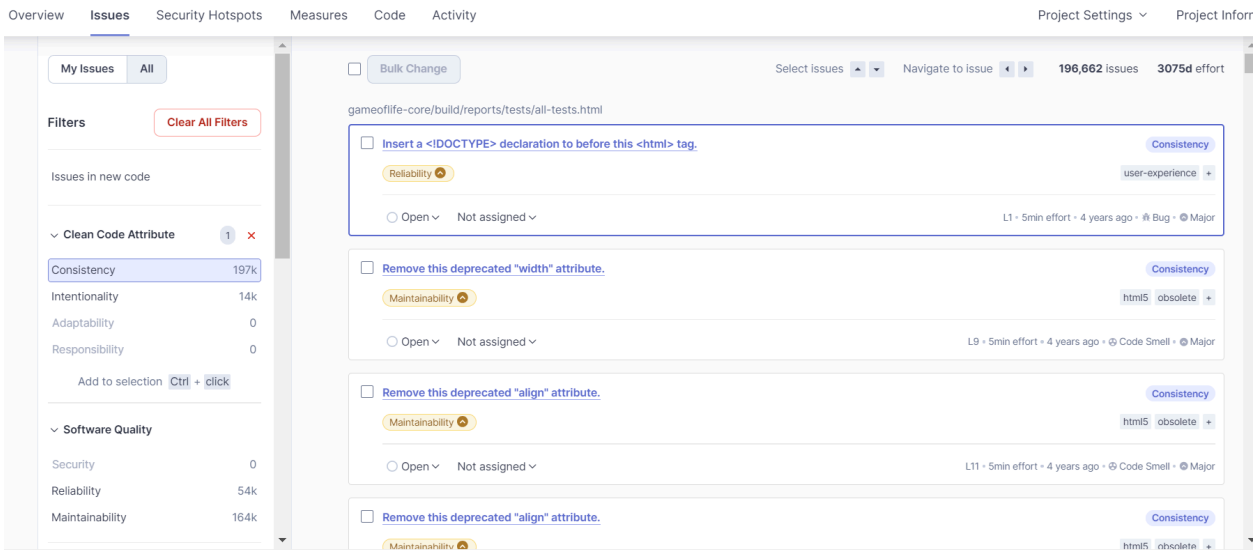


Under different tabs, check all the issues with the code.

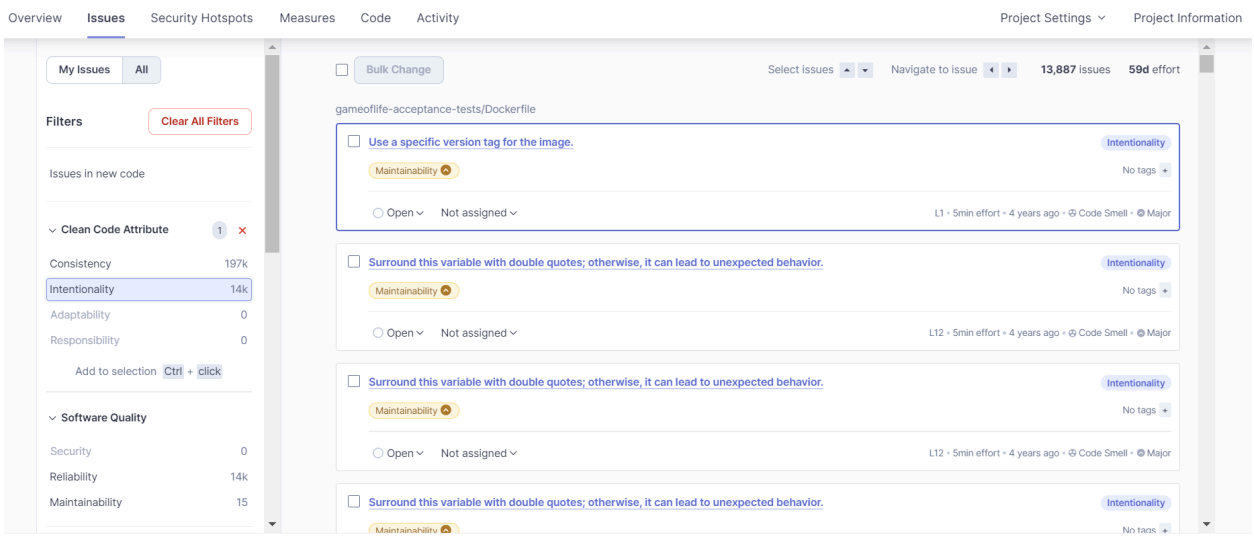
- Code Problems



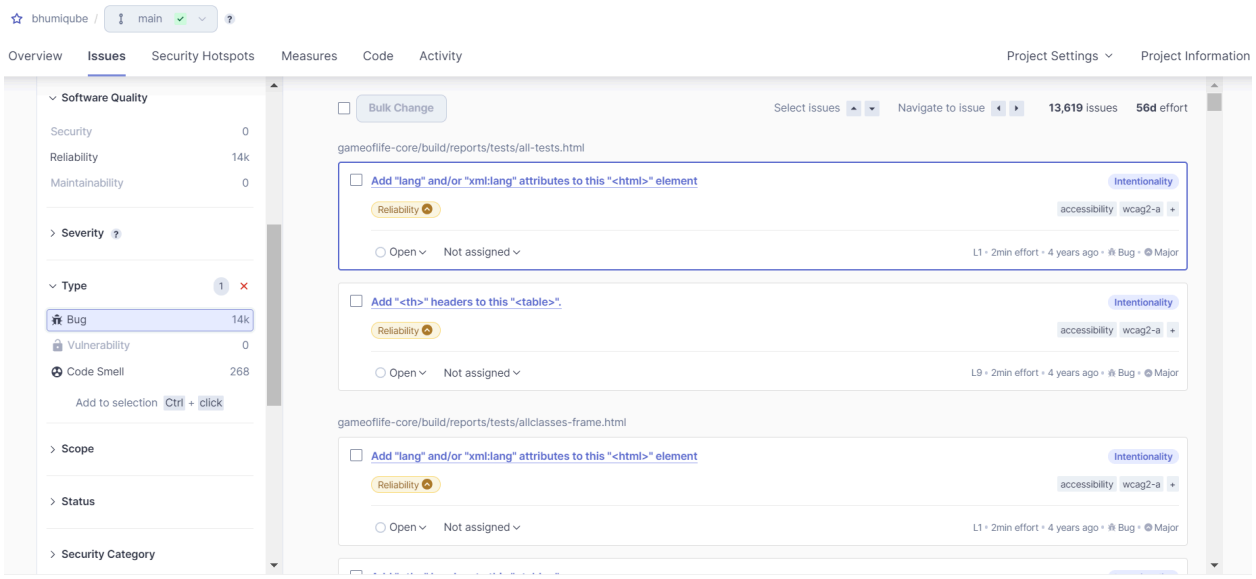
- Consistency



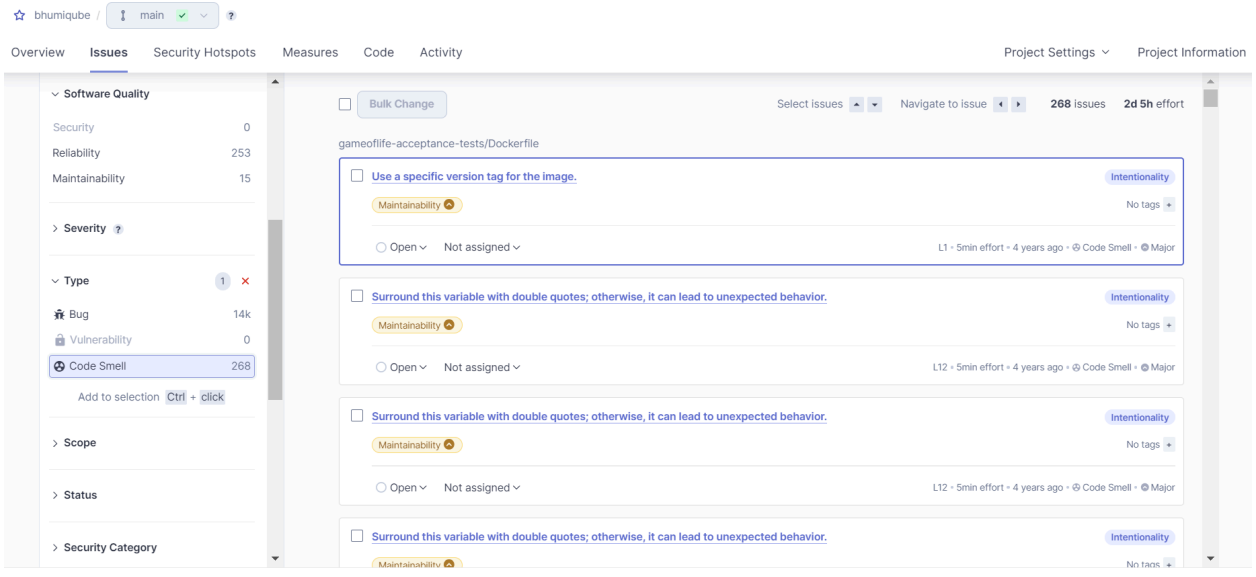
- Intentionality



- Bugs

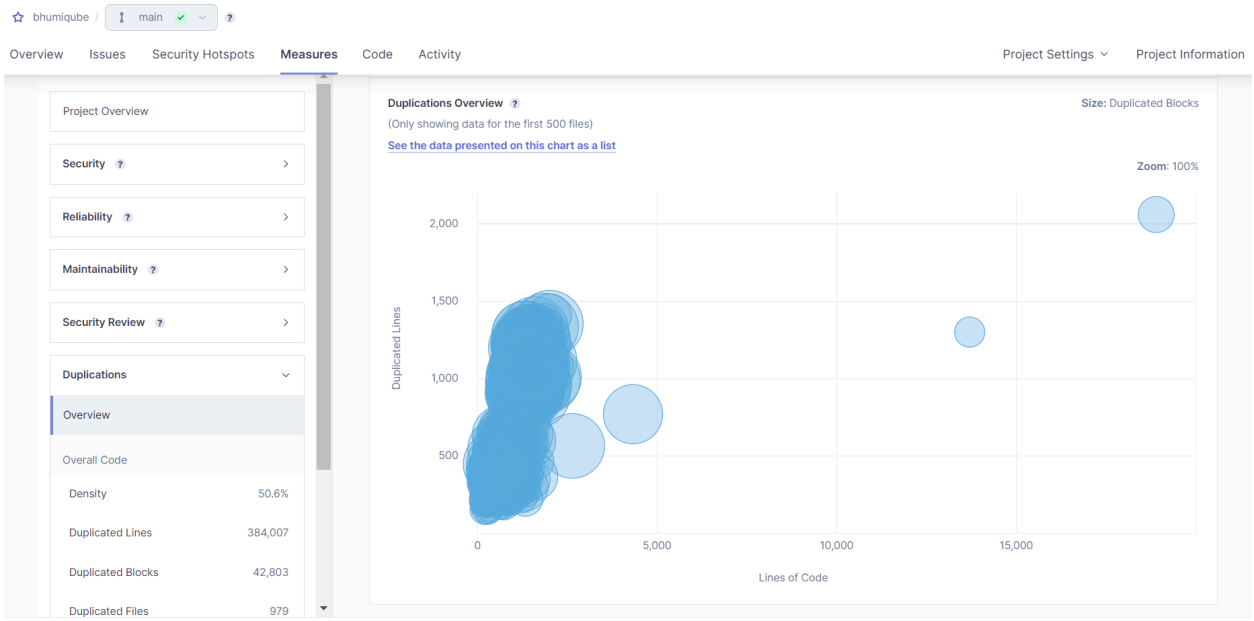


- Code Smells



Embedded database should be used for evaluation purposes only

- Duplications



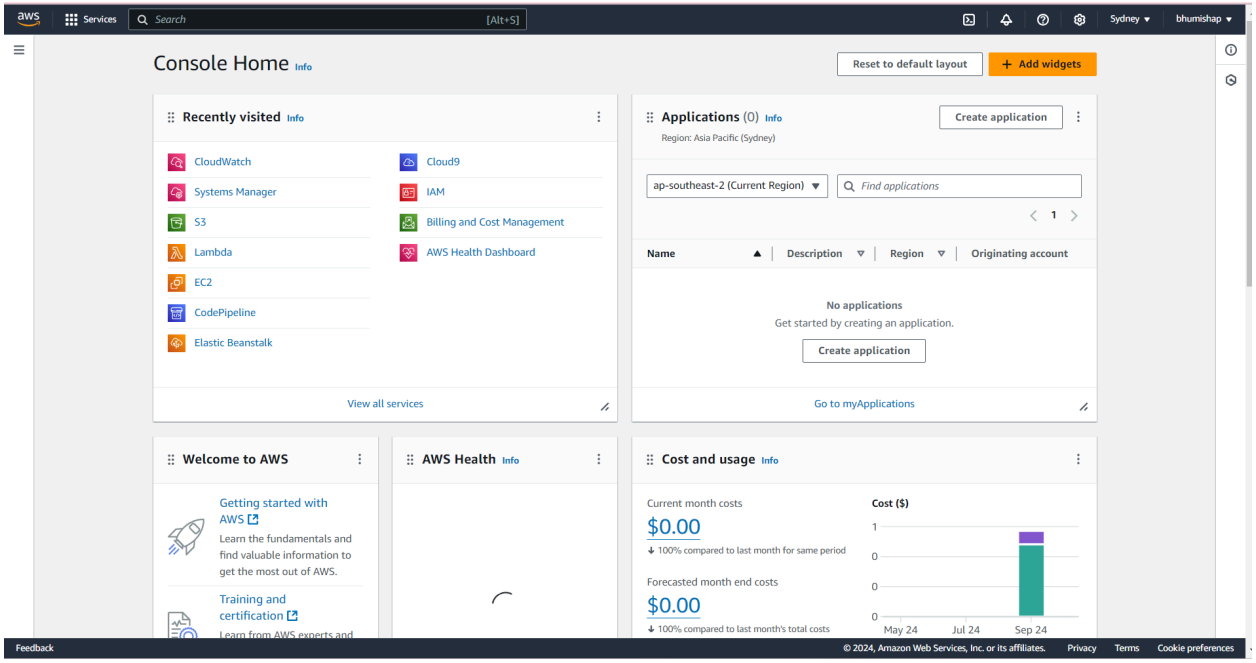
Conclusion:

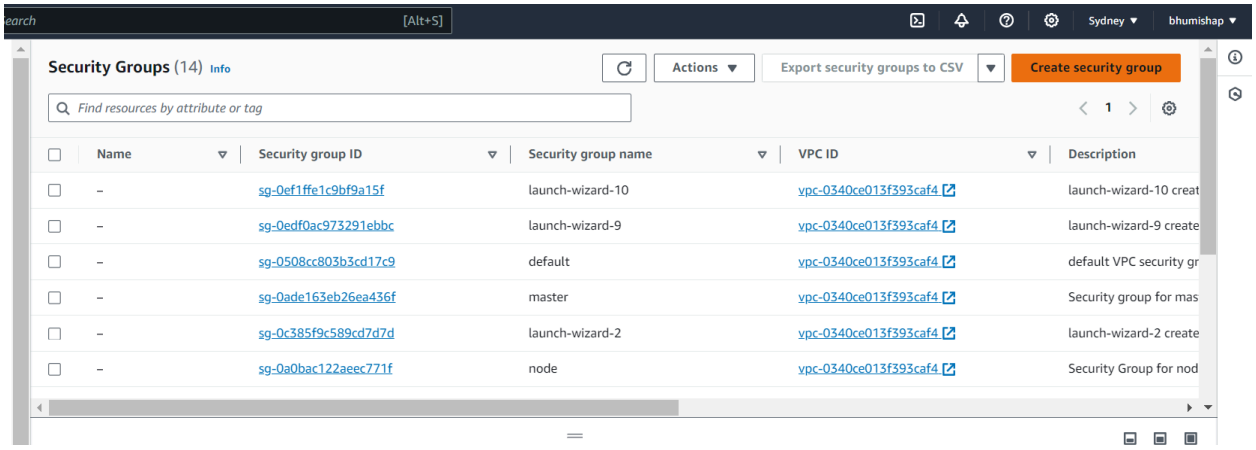
In this experiment, we successfully integrated Jenkins with SonarQube to automate continuous monitoring of code quality within our CI/CD pipeline. The process involved deploying SonarQube using Docker, setting up a project for code analysis, and configuring Jenkins with the SonarQube Scanner plugin. After configuring the tools and providing the SonarQube server details, we developed a Jenkins pipeline that automatically clones code from GitHub and runs static analysis. This integration helps us identify bugs, code smells, and security vulnerabilities throughout the development process, ensuring better code quality and smoother development workflows.

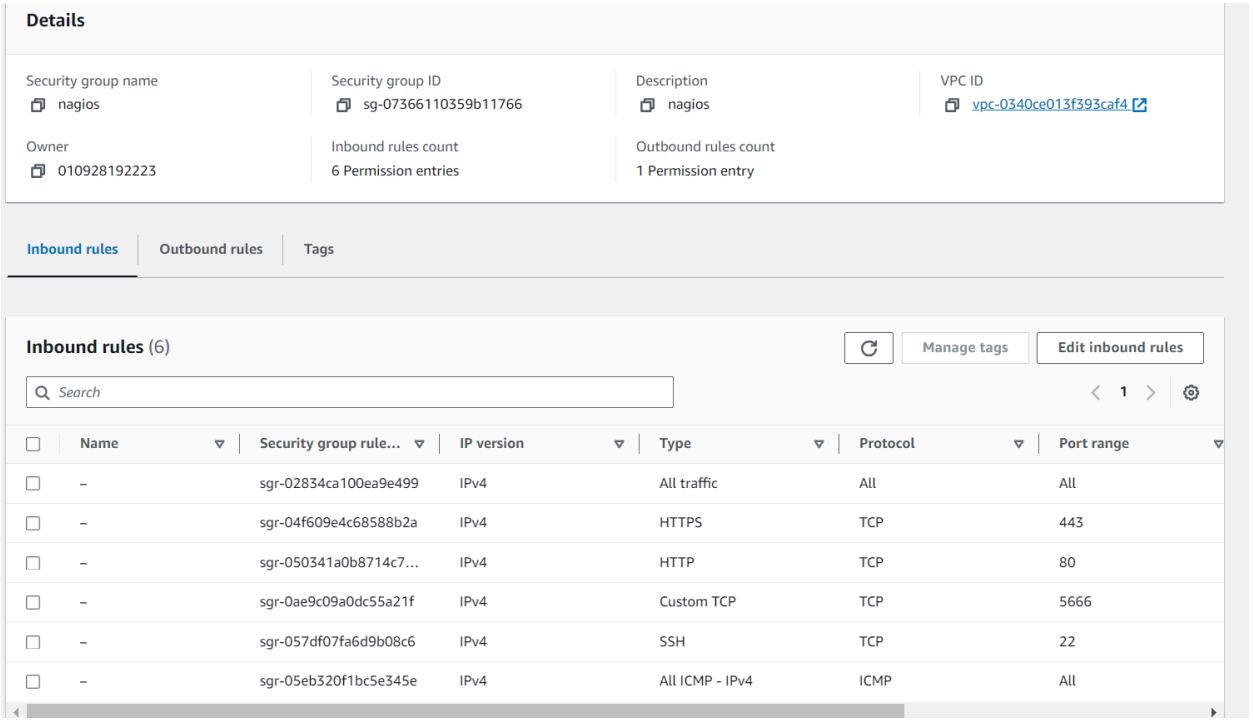
EXPERIMENT NO. 9

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

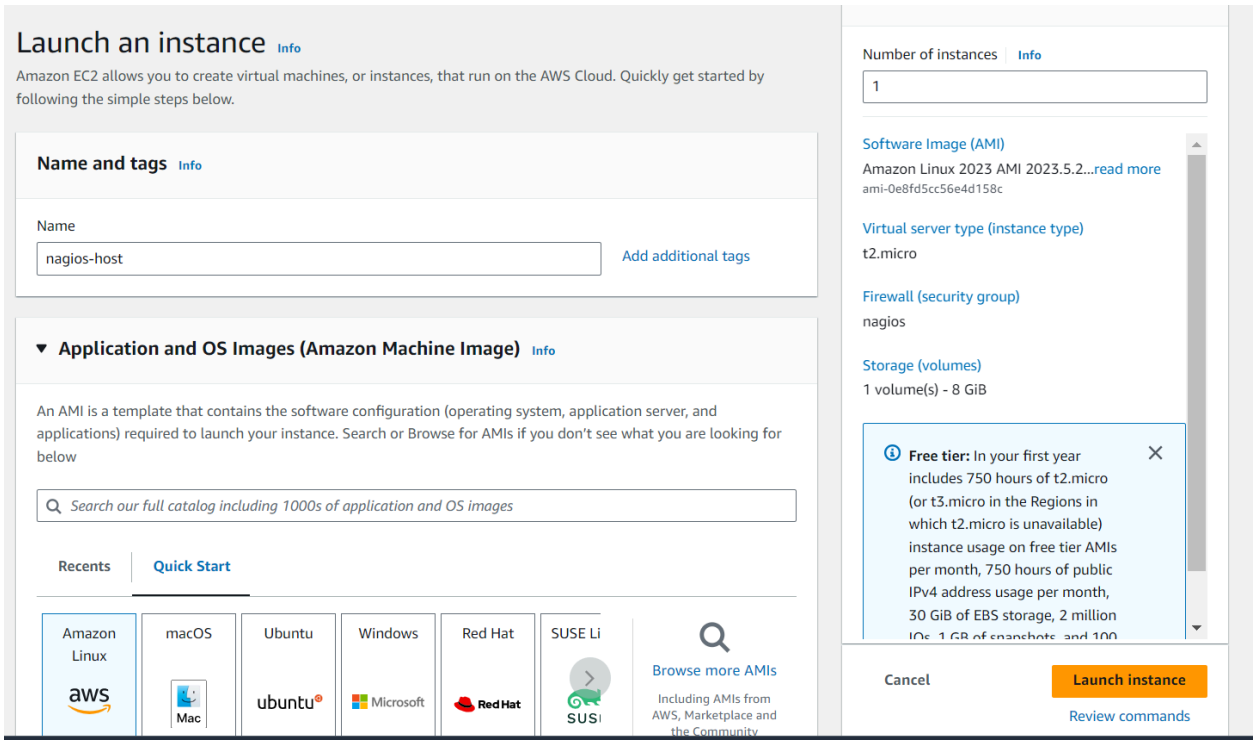
1. Login to your AWS account Personal / Academy. Click on EC2 instance then click on Create Security Group. Give the name as Nagios and any description and add the following inbound rules.





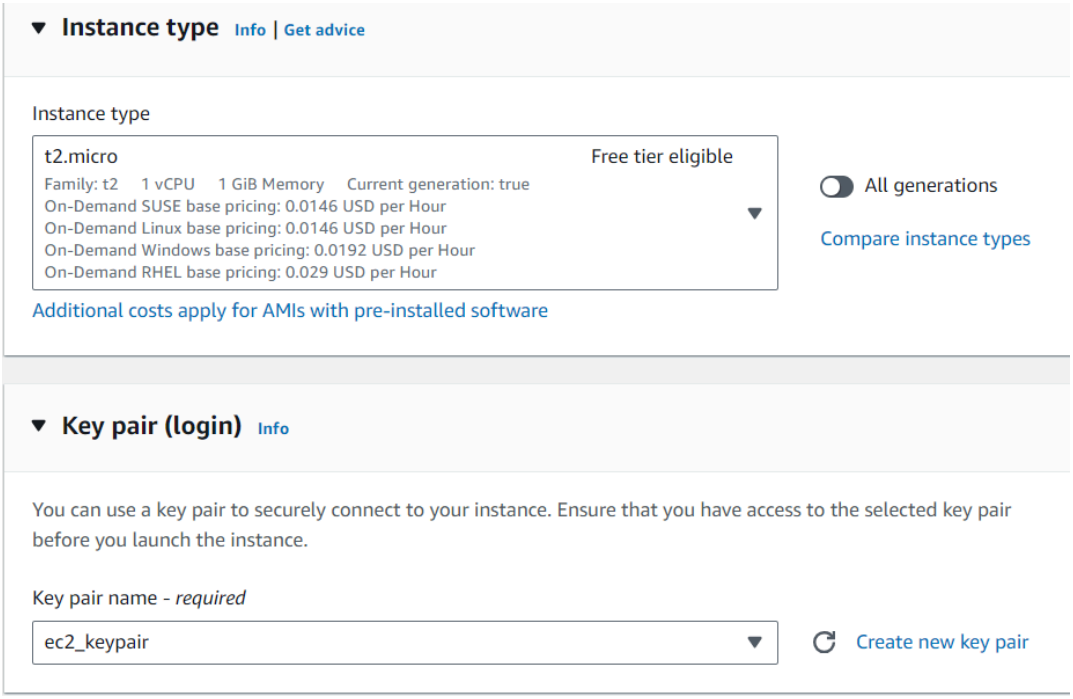


2. Now Create a new EC2 instance. Name: nagios-host, AMI: Amazon Linux, Instance Type: t2.micro.

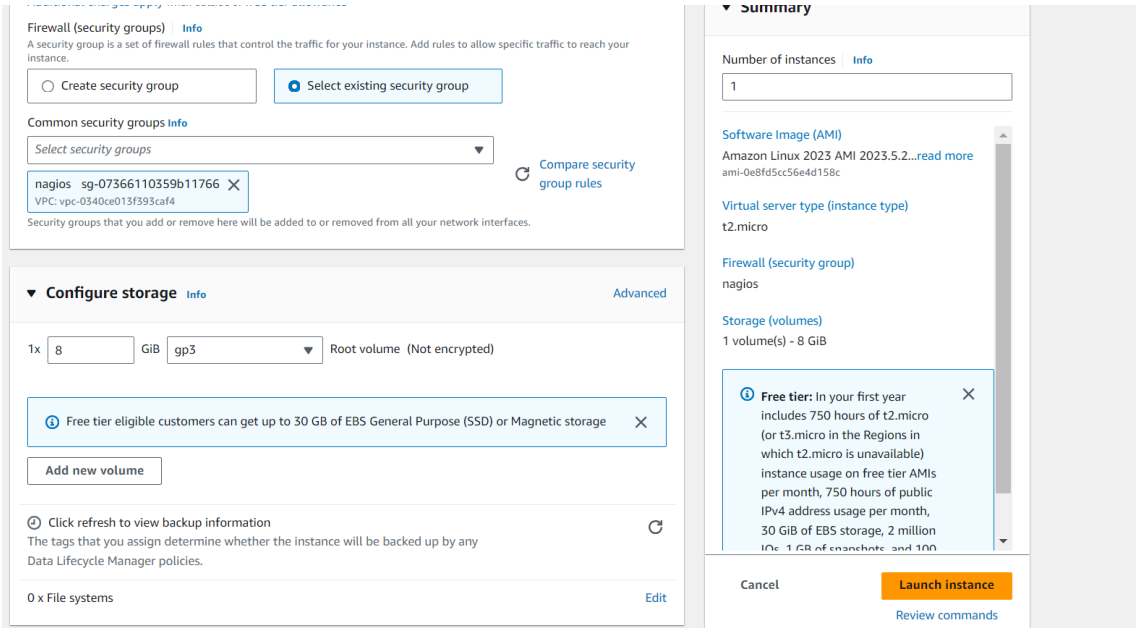


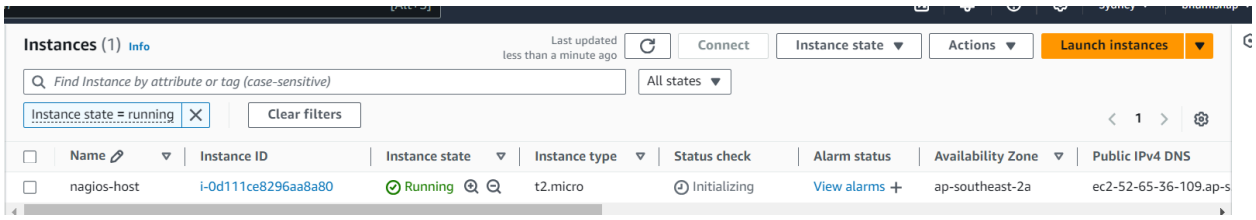
For Key pair : Click on create key and make key of type RSA with extension .pem . Key will be downloaded to your local machine. Now select that key in the key pair, if you already have a key

with type RSA and extension .pem no need to create a new key but you must have that key downloaded.

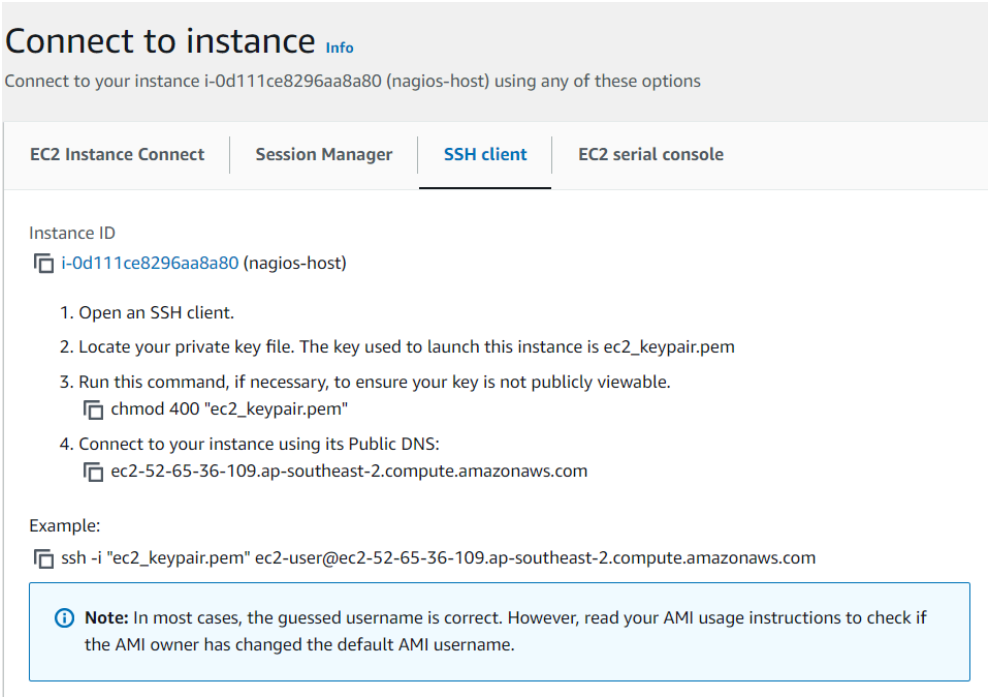


Select the Existing Security Group and select the Security Group we have created in Step 1



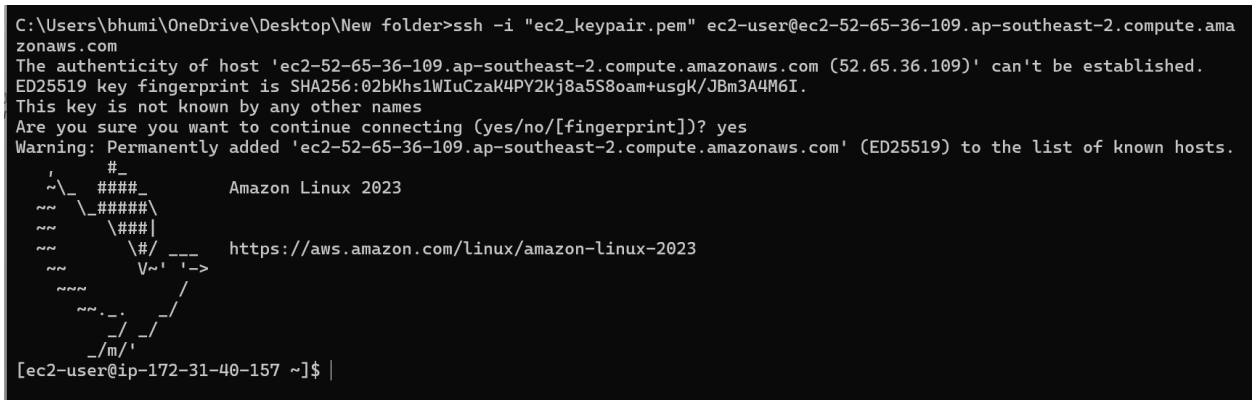


3. Now After creating the EC2 Instance click on connect and then copy the command which is given as example in the SSH Client section .



Now open the terminal in the folder where your key(RSA key with .pem) is located.and paste that copied command

Successfully connected to the instance.



4. Now Run the following command to make a new user and set the password.
sudo adduser -m nagios

sudo passwd nagios

```
[ec2-user@ip-172-31-40-157 ~]$ 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:
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-40-157 ~]$
```

5. Now Run the following command to make a new user group.
   sudo groupadd nagcmd
   sudo usermod -a -G nagcmd nagios
   sudo usermod -a -G nagcmd apache

```
[ec2-user@ip-172-31-40-157 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-40-157 ~]$ sudo usermod -a -G nagcmd nagios
[ec2-user@ip-172-31-40-157 ~]$ sudo usermod -a -G nagcmd apache
```

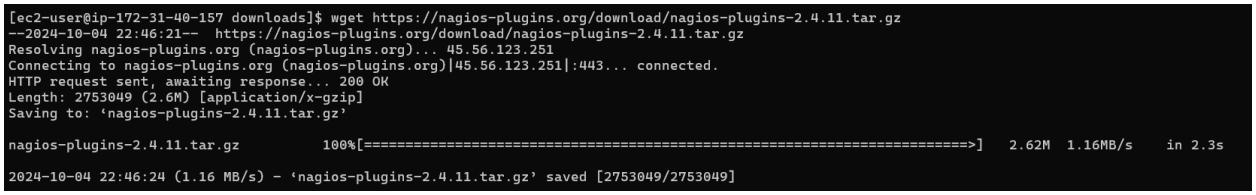
6. Now make a new directory and go to that directory.
   mkdir ~/downloads
   cd ~/downloads

```
[ec2-user@ip-172-31-40-157 ~]$ mkdir ~/downloads
cd ~/downloads
[ec2-user@ip-172-31-40-157 downloads]$
```

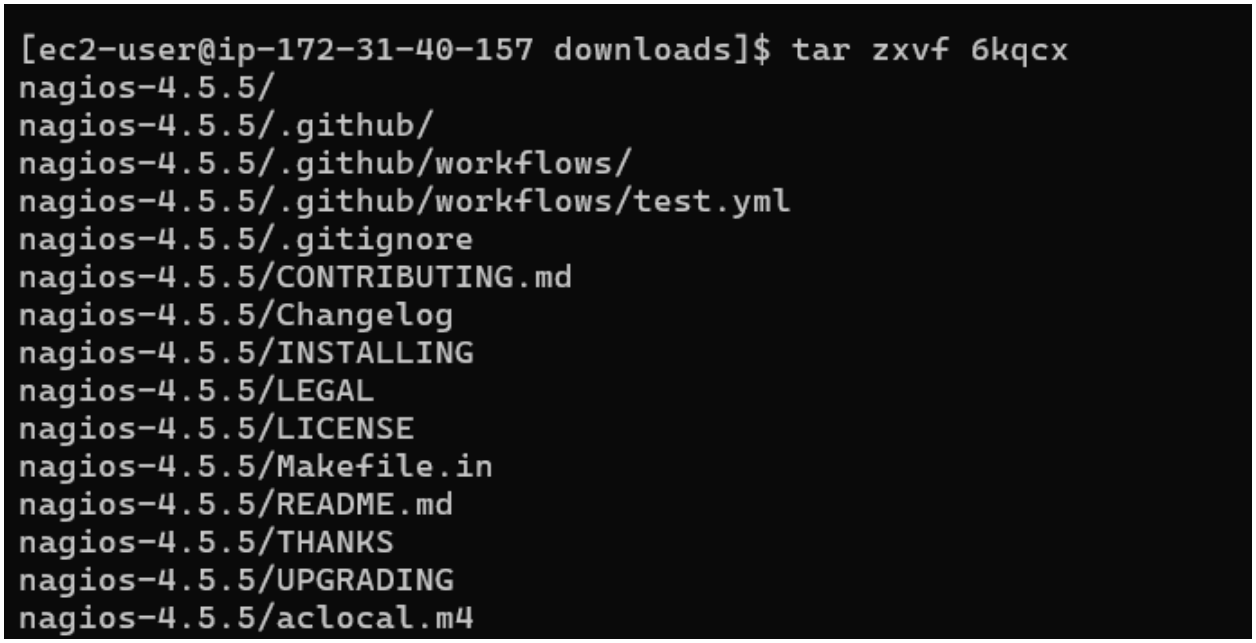
7. Now to download the Nagios 4.5.5 and Nagios-plugins 2.4.11 run the following commands respectively.
   wget https://go.nagios.org/l/975333/2024-09-17/6kqcx

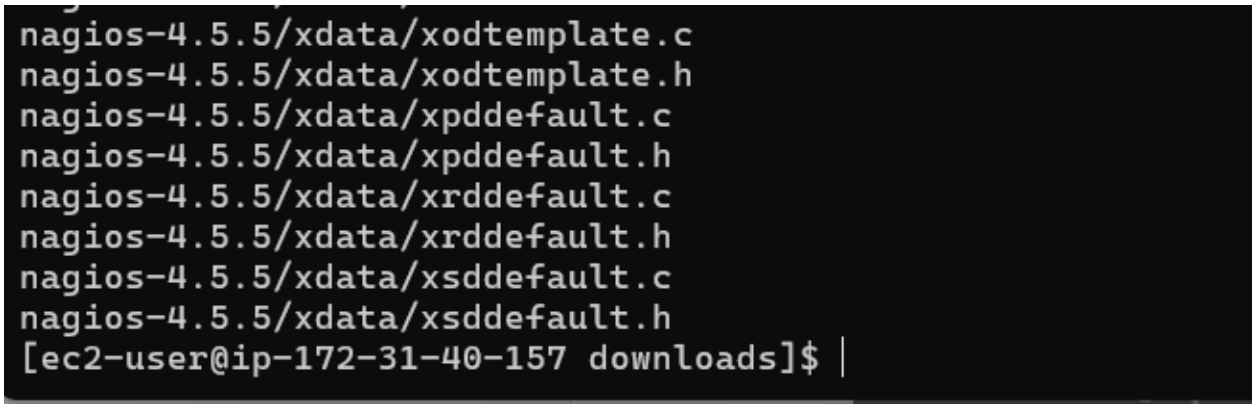
```
[ec2-user@ip-172-31-40-157 downloads]$ wget https://go.nagios.org/l/975333/2024-09-17/6kqcx
--2024-10-04 22:45:39--  https://go.nagios.org/l/975333/2024-09-17/6kqcx
Resolving go.nagios.org (go.nagios.org)... 52.54.96.194, 18.208.125.13, 3.215.172.219, ...
Connecting to go.nagios.org (go.nagios.org)|52.54.96.194|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: http://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8 [following]
--2024-10-04 22:45:40--  http://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8
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|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8 [following]
--2024-10-04 22:45:40--  https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: '6kqcx'

6kqcx                       100%[=========================================================================>]   1.97M  1.10MB/s    in 1.8s

2024-10-04 22:45:43 (1.10 MB/s) - '6kqcx' saved [2065473/2065473]
```

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

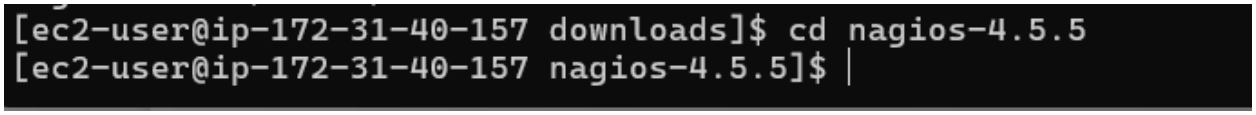


8. Now to extract the files from the downloaded Nagios 4.5.5 run the following command.
   tar zxvf 6kqcx
   (Replace 6kqcx with your saved file name of Nagios 4.5.5 refer above screenshot)





9. Now change the directory to nagios-4.5.5
   cd nagios-4.5.5



10. Now run the following command to configure.
    ./configure --with-command-group=nagcmd

```
[ec2-user@ip-172-31-40-157 nagios-4.5.5]$ ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether the compiler supports GNU C... yes
checking for asprintf... yes
checking for vasprintf... yes
checking for sigaction... yes
checking for C99 vsnprintf... yes
checking for library containing getservbyname... none required
checking for library containing connect... none required
checking for initgroups... yes
checking for setenv... yes
checking for strdup... yes
checking for strstr... yes
checking for strtoul... yes
checking for unsetenv... yes
checking for type of socket size... size_t
checking for Kerberos include files... configure: WARNING: could not find include files
checking for pkg-config... pkg-config
checking for SSL headers... configure: error: Cannot find ssl headers
[ec2-user@ip-172-31-40-157 nagios-4.5.5]$
```

At the end we have found the error of cannot find ssl header

```
checking for pkg-config... pkg-config
checking for SSL headers... configure: error: Cannot find ssl headers
```

So run the following command to install ssl.
sudo yum install openssl-devel

```
[ec2-user@ip-172-31-40-157 nagios-4.5.5]$ sudo yum install openssl-devel
Last metadata expiration check: 0:16:22 ago on Fri Oct  4 22:35:57 2024.
Dependencies resolved.
================================================================================
 Package              Architecture          Version                    Repository                  Size
================================================================================
Installing:
 openssl-devel        x86_64                1:3.0.8-1.amzn2023.0.14    amazonlinux                 3.0 M

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

Total download size: 3.0 M
Installed size: 4.7 M
Is this ok [y/N]: y
Downloading Packages:
openssl-devel-3.0.8-1.amzn2023.0.14.x86_64.rpm                  31 MB/s | 3.0 MB     00:00
--------------------------------------------------------------------------------
Total                                                           21 MB/s | 3.0 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                            1/1
  Installing       : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64               1/1
  Running scriptlet: openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64               1/1
  Verifying        : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64               1/1

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

Complete!
```

Now rerun the command
./configure --with-command-group=nagcmd

```
[ec2-user@ip-172-31-41-231 nagios-4.5.5]$ ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether the compiler supports GNU C... yes
checking whether gcc accepts -g... yes
checking for gcc option to enable C11 features... none needed
```

```
*** Support Notes *****************************************

If you have questions about configuring or running Nagios,
please make sure that you:

     - Look at the sample config files
     - Read the documentation on the Nagios Library at:
          https://library.nagios.com

before you post a question to one of the mailing lists.
Also make sure to include pertinent information that could
help others help you.  This might include:

     - What version of Nagios you are using
     - What version of the plugins you are using
     - Relevant snippets from your config files
     - Relevant error messages from the Nagios log file

For more information on obtaining support for Nagios, visit:

       https://support.nagios.com

************************************************************

Enjoy.
```

11. Now run the following commands to setup the Nagios.
    sudo make install

```
[ec2-user@ip-172-31-40-157 nagios-4.5.5]$ sudo make install
cd ./base && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagios /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagiostats /usr/local/nagios/bin
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5/base'
cd ./cgi && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
make install-basic
make[2]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/sbin
for file in *.cgi; do \
       /usr/bin/install -c -s -m 775 -o nagios -g nagios $file /usr/local/nagios/sbin; \
done
make[2]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
cd ./html && make install
```

```
*** Exfoliation theme installed ***
NOTE: Use 'make install-classicui' to revert to classic Nagios theme

make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5'
make install-basic
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/libexec
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/var
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/var/archives
/usr/bin/install -c -m 775 -o nagios -g nagcmd -d /usr/local/nagios/var/spool/checkresults
chmod g+s /usr/local/nagios/var/spool/checkresults

*** Main program, CGIs and HTML files installed ***

You can continue with installing Nagios as follows (type 'make'
without any arguments for a list of all possible options):

  make install-init
     - This installs the init script in /lib/systemd/system

  make install-commandmode
     - This installs and configures permissions on the
       directory for holding the external command file

  make install-config
     - This installs sample config files in /usr/local/nagios/etc

make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5'
[ec2-user@ip-172-31-40-157 nagios-4.5.5]$
```

sudo make install-init

```
/usr/bin/install -c -m 755 -d -o root -g root /lib/systemd/system
/usr/bin/install -c -m 755 -o root -g root startup/default-service /lib/systemd/system/nagios.service
[ec2-user@ip-172-31-40-157 nagios-4.5.5]$
```

sudo make install-config

```
[ec2-user@ip-172-31-40-157 nagios-4.5.5]$ sudo make install-config
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc/objects
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/nagios.cfg /usr/local/nagios/etc/nagios.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/cgi.cfg /usr/local/nagios/etc/cgi.cfg
/usr/bin/install -c -b -m 660 -o nagios -g nagios sample-config/resource.cfg /usr/local/nagios/etc/resource.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/templates.cfg /usr/local/nagios/etc/objects/templates.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/commands.cfg /usr/local/nagios/etc/objects/commands.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/contacts.cfg /usr/local/nagios/etc/objects/contacts.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/timeperiods.cfg /usr/local/nagios/etc/objects/timeperiods.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/localhost.cfg /usr/local/nagios/etc/objects/localhost.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/windows.cfg /usr/local/nagios/etc/objects/windows.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/printer.cfg /usr/local/nagios/etc/objects/printer.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/switch.cfg /usr/local/nagios/etc/objects/switch.cfg

*** Config files installed ***

Remember, these are *SAMPLE* config files.  You'll need to read
the documentation for more information on how to actually define
services, hosts, etc. to fit your particular needs.
```

sudo make install-webconf

```
[ec2-user@ip-172-31-40-157 nagios-4.5.5]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios.conf
if [ 0 -eq 1 ]; then \
        ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagios.conf; \
fi

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

12. To set the password
    sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin

```
[ec2-user@ip-172-31-40-157 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-172-31-40-157 nagios-4.5.5]$
```

13. Now restart the httpd service and to do so, run the following command.
    sudo service httpd restart

```
[ec2-user@ip-172-31-40-157 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-40-157 nagios-4.5.5]$
```

14. Now to extract the files from the downloaded Nagios plugin 2.4.11 run the following command first change the directory.
    cd ~/downloads
    tar zxvf nagios-plugins-2.4.11.tar.gz

```
[ec2-user@ip-172-31-40-157 nagios-4.5.5]$ cd ~/downloads
[ec2-user@ip-172-31-40-157 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/plugins-root/t/check_dhcp.t
nagios-plugins-2.4.11/plugins-root/t/check_icmp.t
nagios-plugins-2.4.11/po/
nagios-plugins-2.4.11/po/Makefile.in.in
nagios-plugins-2.4.11/po/remove-potcdate.sin
nagios-plugins-2.4.11/po/Makevars
nagios-plugins-2.4.11/po/POTFILES.in
nagios-plugins-2.4.11/po/fr.po
nagios-plugins-2.4.11/po/de.po
nagios-plugins-2.4.11/po/fr.gmo
nagios-plugins-2.4.11/po/de.gmo
nagios-plugins-2.4.11/po/nagios-plugins.pot
nagios-plugins-2.4.11/po/stamp-po
nagios-plugins-2.4.11/po/ChangeLog
nagios-plugins-2.4.11/po/LINGUAS
nagios-plugins-2.4.11/release
[ec2-user@ip-172-31-40-157 downloads]$
```

15. Now change the directory to nagios-plugins-2.4.11 and run the config command to configure.
    cd nagios-plugins-2.4.11
    ./configure --with-nagios-user=nagios --with-nagios-group=nagios

```
[ec2-user@ip-172-31-40-157 downloads]$ cd nagios-plugins-2.4.11
[ec2-user@ip-172-31-40-157 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
```

```
config.status: creating perlmods/Makefile
config.status: creating test.pl
config.status: creating pkg/solaris/pkginfo
config.status: creating po/Makefile.in
config.status: creating config.h
config.status: config.h is unchanged
config.status: executing depfiles commands
config.status: executing libtool commands
config.status: executing po-directories commands
config.status: creating po/POTFILES
config.status: creating po/Makefile
[ec2-user@ip-172-31-40-157 nagios-plugins-2.4.11]$
```

16. Run the following commands to check nagios and start it. sudo chkconfig --add nagios

```
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

Things look okay - No serious problems were detected during the pre-flight check
[ec2-user@ip-172-31-40-157 nagios-plugins-2.4.11]$
```

cd sudo service nagios start

```
[ec2-user@ip-172-31-40-157 nagios-plugins-2.4.11]$ cd
[ec2-user@ip-172-31-40-157 ~]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-40-157 ~]$
```

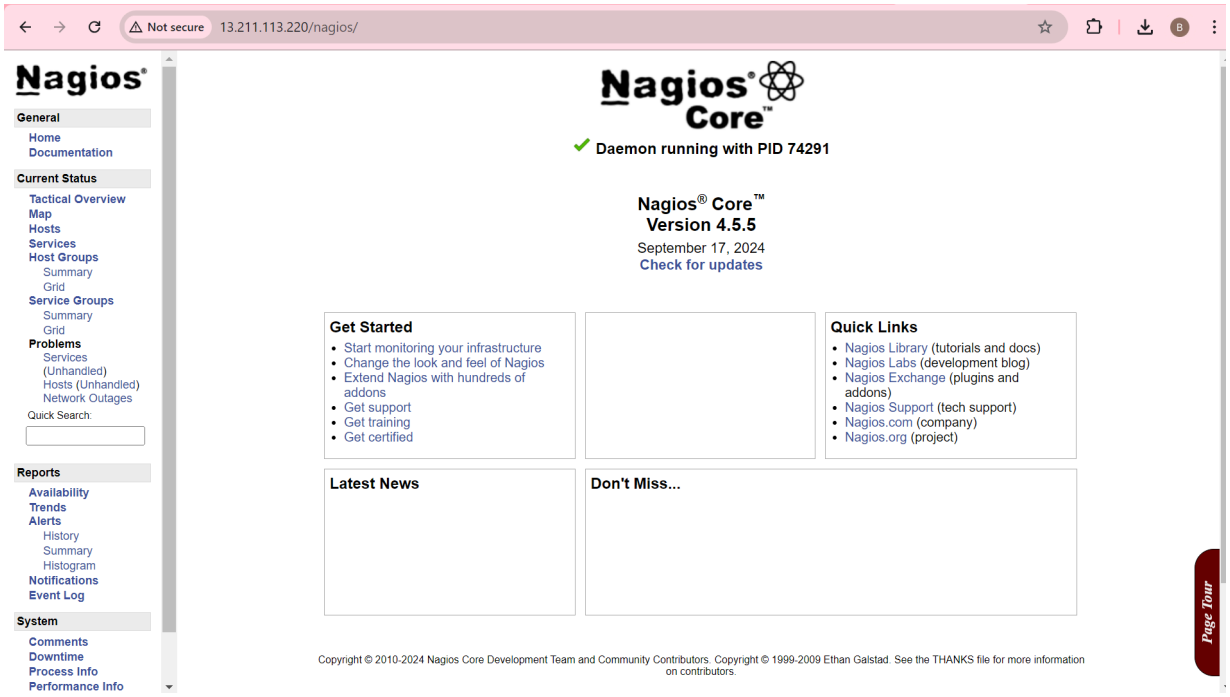
sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

```
[ec2-user@ip-172-31-40-157 ~]$ 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.
```

sudo systemctl status nagios

```
[ec2-user@ip-172-31-40-157 ~]$ sudo systemctl status nagios
● nagios.service - Nagios Core Monitoring
     Loaded: loaded (/etc/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Fri 2024-10-04 23:20:25 UTC; 5min ago
   Main PID: 57764 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 4.1M
        CPU: 59ms
     CGroup: /system.slice/nagios.service
             ├─57764 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg
             ├─57765 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─57766 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─57767 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─57768 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─57769 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg

Oct 04 23:20:25 ip-172-31-40-157.ap-southeast-2.compute.internal nagios[57764]: wproc: Registry request: name=Core Worker 57765;pid=57765
Oct 04 23:20:25 ip-172-31-40-157.ap-southeast-2.compute.internal nagios[57764]: wproc: Registry request: name=Core Worker 57768;pid=57768
Oct 04 23:20:25 ip-172-31-40-157.ap-southeast-2.compute.internal nagios[57764]: wproc: Registry request: name=Core Worker 57768;pid=57768
Oct 04 23:20:25 ip-172-31-40-157.ap-southeast-2.compute.internal nagios[57764]: wproc: Registry request: name=Core Worker 57767;pid=57767
Oct 04 23:20:25 ip-172-31-40-157.ap-southeast-2.compute.internal nagios[57764]: wproc: Registry request: name=Core Worker 57767;pid=57767
Oct 04 23:20:25 ip-172-31-40-157.ap-southeast-2.compute.internal nagios[57764]: Successfully launched command file worker with pid 57769
Oct 04 23:20:25 ip-172-31-40-157.ap-southeast-2.compute.internal nagios[57764]: Successfully launched command file worker with pid 57769
Oct 04 23:20:55 ip-172-31-40-157.ap-southeast-2.compute.internal nagios[57764]: HOST ALERT: localhost;DOWN;SOFT;8;(No output on stdout) stderr: execvp(/usr/local/nagios
Oct 04 23:21:55 ip-172-31-40-157.ap-southeast-2.compute.internal nagios[57764]: HOST ALERT: localhost;DOWN;SOFT;9;(No output on stdout) stderr: execvp(/usr/local/nagios
Oct 04 23:22:55 ip-172-31-40-157.ap-southeast-2.compute.internal nagios[57764]: HOST ALERT: localhost;DOWN;HARD;10;(No output on stdout) stderr: execvp(/usr/local/nagio
[ec2-user@ip-172-31-40-157 ~]$
```

17. We can see we have successfully launched Nagios now .
    Open http:///nagios/ here it is http://<ip address>/nagios and we can see the running web page of nagios.



Conclusion: In this experiment, we have set up the Nagios core with plugins on Amazon Linux. Which will help us to understand Continuous monitoring and Installation. It is important to note that whatever set of rules we have added in step 1 are very important for this experiment.

EXPERIMENT NO. 10

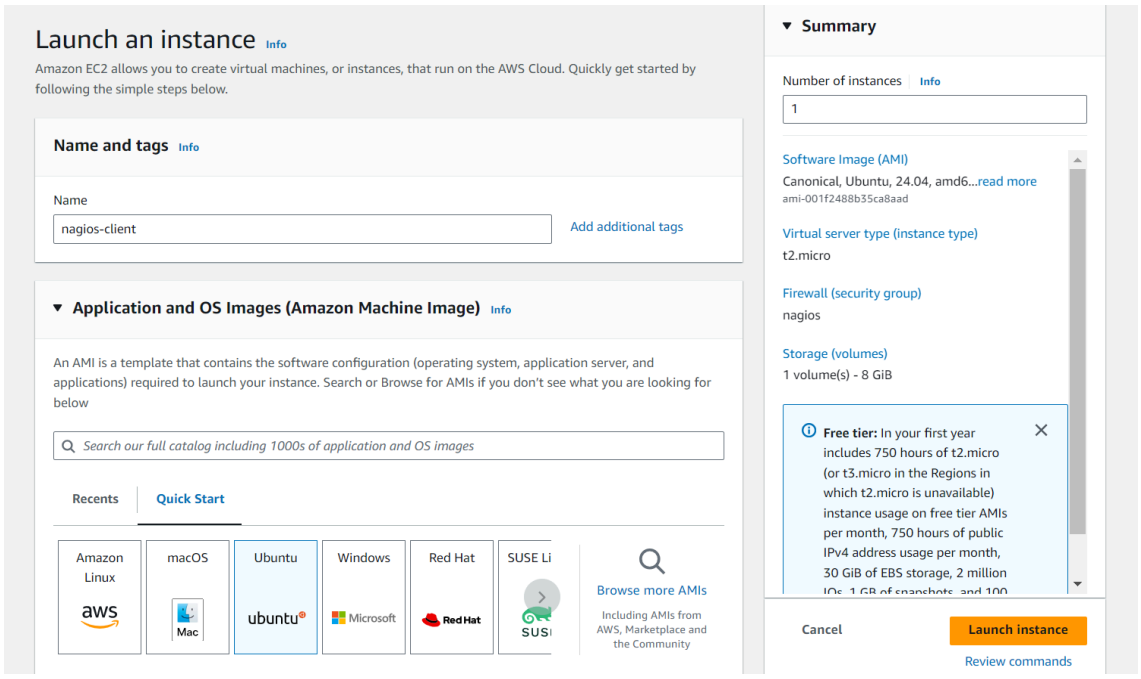
Aim: To perform Port, Service monitoring, and Windows/Linux server monitoring using Nagios.

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

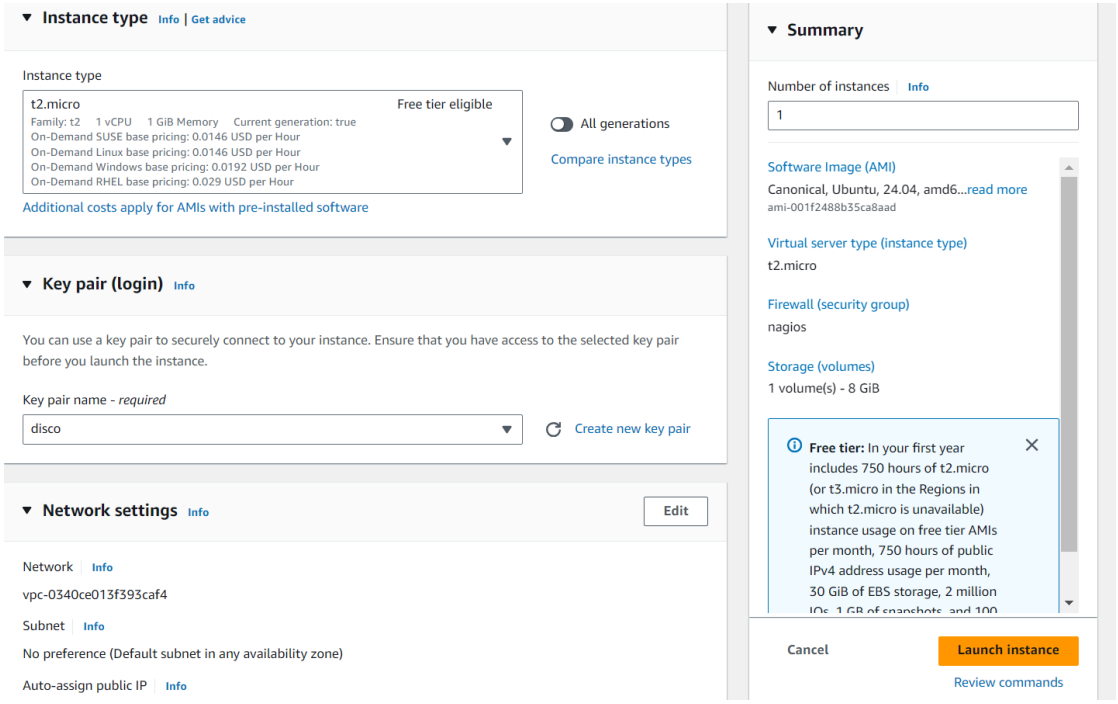
```
[ec2-user@ip-172-31-13-224 ~]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/etc/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Sat 2024-10-05 19:22:57 UTC; 2min 1s ago
   Main PID: 74867 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 5.6M
        CPU: 233ms
     CGroup: /system.slice/nagios.service
             ├─74867 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg
             ├─74868 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─74869 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─74870 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─74871 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─74872 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg

Oct 05 19:22:57 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[74867]: wproc: Registry request: name=Core Worker 74869;pid=>
Oct 05 19:22:57 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[74867]: wproc: Registry request: name=Core Worker 74868;pid=>
Oct 05 19:22:57 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[74867]: Successfully launched command file worker with pid 7>
Oct 05 19:22:57 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[74867]: Successfully launched command file worker with pid 7>
Oct 05 19:22:57 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[74867]: HOST ALERT: localhost;DOWN;SOFT;1;(No output on stdo>
Oct 05 19:23:34 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[74867]: SERVICE ALERT: localhost;Current Load;CRITICAL;HARD;>
Oct 05 19:23:57 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[74867]: HOST ALERT: localhost;DOWN;SOFT;2;(No output on stdo>
Oct 05 19:24:12 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[74867]: SERVICE ALERT: localhost;Current Users;CRITICAL;HARD>
Oct 05 19:24:49 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[74867]: SERVICE ALERT: localhost;HTTP;CRITICAL;HARD;1;(No ou>
Oct 05 19:24:57 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[74867]: HOST ALERT: localhost;DOWN;SOFT;3;(No output on stdo>
lines 1-25/25 (END)
```

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

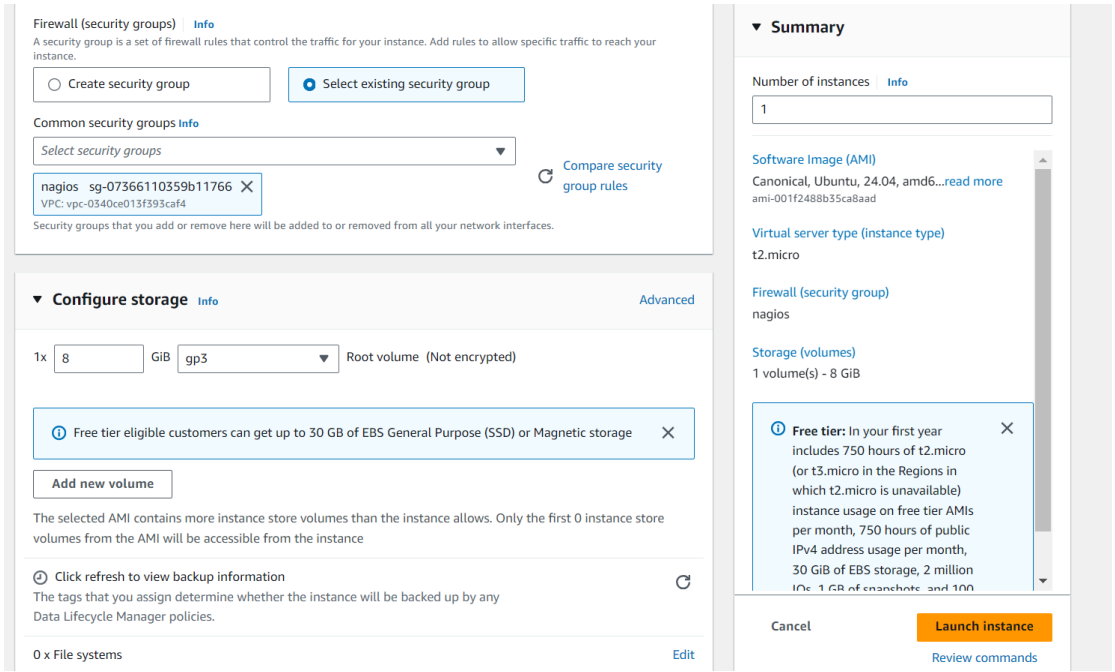
2. Now Create a new EC2 instance. Name: Nagios-client, AMI: Ubuntu Instance Type: t2.micro.



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 the key pair if you already have a key with type RSA and extension .pem no need to create a 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).



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\bhumi> cd "C:\Users\bhumi\OneDrive\Desktop\New folder"
PS C:\Users\bhumi\OneDrive\Desktop\New folder> ssh -i "disco.pem" ubuntu@ec2-3-25-84-91.ap-southeast-2.compute.amazonaws.com
The authenticity of host 'ec2-3-25-84-91.ap-southeast-2.compute.amazonaws.com (3.25.84.91)' can't be established.
ED25519 key fingerprint is SHA256:zIXx3ATmWCr9U4e6inijfig4vG+Bji+Xm8lz0cDdHCc.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-25-84-91.ap-southeast-2.compute.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1016-aws x86_64)

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

 System information as of Sat Oct  5 18:51:05 UTC 2024

  System load:  0.0               Processes:             105
  Usage of /:   23.0% of 6.71GB   Users logged in:       0
  Memory usage: 20%               IPv4 address for enX0: 172.31.5.17
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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


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

Last login: Sat Oct  5 18:26:23 2024 from 13.239.158.3
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

Successfully connected to the instance.

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

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

```
[ec2-user@ip-172-31-13-224 ~]$ ps -ef | grep nagios
nagios     74867       1  0 19:22 ?        00:00:00 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg
nagios     74868   74867  0 19:22 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     74869   74867  0 19:22 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     74870   74867  0 19:22 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     74871   74867  0 19:22 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     74872   74867  0 19:22 ?        00:00:00 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg
ec2-user   75110   75013  0 19:26 pts/1    00:00:00 grep --color=auto nagios
[ec2-user@ip-172-31-13-224 ~]$
```

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-172-31-13-224 ~]$ sudo su
[root@ip-172-31-13-224 ec2-user]# mkdir /usr/local/nagios/etc/objects/monitorhosts
[root@ip-172-31-13-224 ec2-user]# mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-13-224 ec2-user]#
```

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-172-31-13-224 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhost
s/linuxserver.cfg
[root@ip-172-31-13-224 ec2-user]#
```

7. Open linuxserver.cfg using nano and make the following changes everywhere in the file.
   Change hostname to linuxserver.
   Change address to the public IP of your Linux client.
   Set hostgroup_name to linux-servers1.

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

```
[root@ip-172-31-13-224 ec2-user]# sudo sed -i 's/^ *host_name.*/    host_name       linuxserver/' /usr/local/nagios/etc/objects/moni
torhosts/linuxhosts/linuxserver.cfg
sudo sed -i 's/^ *address.*/    address         3.25.70.155/' /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
sudo sed -i 's/^ *hostgroup_name.*/    hostgroup_name   linux-servers1/' /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxs
erver.cfg
[root@ip-172-31-13-224 ec2-user]# nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
[root@ip-172-31-13-224 ec2-user]# nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
```

```
##############################################################################
#
# 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         3.25.70.155
}
```

8. Now update the Nagios config file .Add the following line in the file.

   Line to add : cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/
   Run the command : nano /usr/local/nagios/etc/nagios.cfg

```
# 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/
```

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

```
[root@ip-172-31-13-224 ec2-user]# /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

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

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

Running pre-flight check on configuration data...

Checking objects...
        Checked 16 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
```

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

```
[root@ip-172-31-13-224 ec2-user]# service nagios restart
Redirecting to /bin/systemctl restart nagios.service
[root@ip-172-31-13-224 ec2-user]#
```

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-172-31-5-17:~$ sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins
Hit:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:5 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:6 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:8 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:9 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:10 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:11 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:12 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:13 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [537 kB]
Get:14 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [132 kB]
Get:15 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [8860 B]
Get:16 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [384 kB]
Get:17 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [159 kB]
```

```
Creating config file /etc/nagios-plugins/config/snmp.cfg with new version
Setting up monitoring-plugins (2.3.5-1ubuntu3) ...
Setting up libldb2:amd64 (2:2.8.0+samba4.19.5+dfsg-4ubuntu9) ...
Setting up libavahi-client3:amd64 (0.8-13ubuntu6) ...
Setting up samba-libs:amd64 (2:4.19.5+dfsg-4ubuntu9) ...
Setting up python3-ldb (2:2.8.0+samba4.19.5+dfsg-4ubuntu9) ...
Setting up samba-dsdb-modules:amd64 (2:4.19.5+dfsg-4ubuntu9) ...
Setting up libsmbclient0:amd64 (2:4.19.5+dfsg-4ubuntu9) ...
Setting up libcups2t64:amd64 (2.4.7-1.2ubuntu7.3) ...
Setting up python3-samba (2:4.19.5+dfsg-4ubuntu9) ...
Setting up smbclient (2:4.19.5+dfsg-4ubuntu9) ...
Setting up samba-common-bin (2:4.19.5+dfsg-4ubuntu9) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.3) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-5-17:~$
```

12. Open nrpe.cfg file to make changes.Under allowed_hosts, add your nagios host IP address.
    sudo nano /etc/nagios/nrpe.cfg

```
# 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,::1,3.25.70.155
```

13. Now restart the NRPE server by this command.
    sudo systemctl restart nagios-nrpe-server

```
ubuntu@ip-172-31-5-17:~$ sudo systemctl restart nagios-nrpe-server
ubuntu@ip-172-31-5-17:~$
```

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 activate it.
    sudo systemctl status nagios

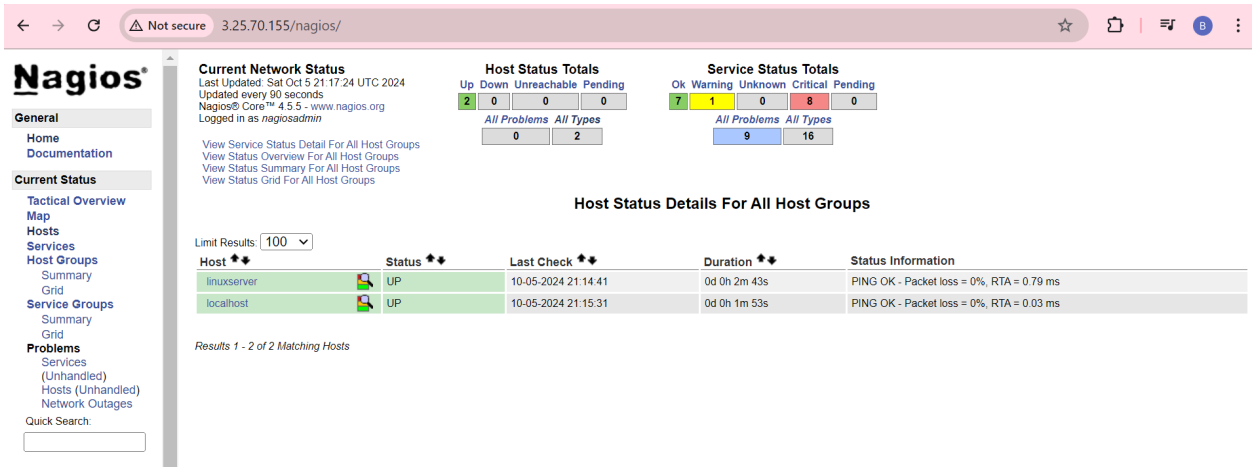
```
[root@ip-172-31-13-224 ec2-user]# sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/etc/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Sat 2024-10-05 19:42:04 UTC; 22min ago
   Main PID: 76185 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 4.1M
        CPU: 252ms
     CGroup: /system.slice/nagios.service
             ├─76185 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg
             ├─76186 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─76187 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─76188 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─76189 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─76190 /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg

Oct 05 19:45:11 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[76185]: SERVICE ALERT: linuxserver;Root Partition;CRITICAL;HARD;1;(No o>
Oct 05 19:45:49 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[76185]: SERVICE ALERT: linuxserver;SSH;CRITICAL;HARD;1;(No output on st>
Oct 05 19:46:04 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[76185]: HOST ALERT: linuxserver;DOWN;SOFT;5;(No output on stdout) stder>
Oct 05 19:46:26 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[76185]: SERVICE ALERT: linuxserver;Swap Usage;CRITICAL;HARD;1;(No outpu>
Oct 05 19:47:04 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[76185]: SERVICE ALERT: linuxserver;Total Processes;CRITICAL;HARD;1;(No >
Oct 05 19:47:04 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[76185]: HOST ALERT: linuxserver;DOWN;SOFT;6;(No output on stdout) stder>
Oct 05 19:48:04 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[76185]: HOST ALERT: linuxserver;DOWN;SOFT;7;(No output on stdout) stder>
Oct 05 19:49:04 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[76185]: HOST ALERT: linuxserver;DOWN;SOFT;8;(No output on stdout) stder>
Oct 05 19:50:04 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[76185]: HOST ALERT: linuxserver;DOWN;SOFT;9;(No output on stdout) stder>
Oct 05 19:51:04 ip-172-31-13-224.ap-southeast-2.compute.internal nagios[76185]: HOST ALERT: linuxserver;DOWN;HARD;10;(No output on stdout) stde>
lines 1-25/25 (END)
```

sudo systemctl status httpd
sudo systemctl start httpd
sudo systemctl enable httpd

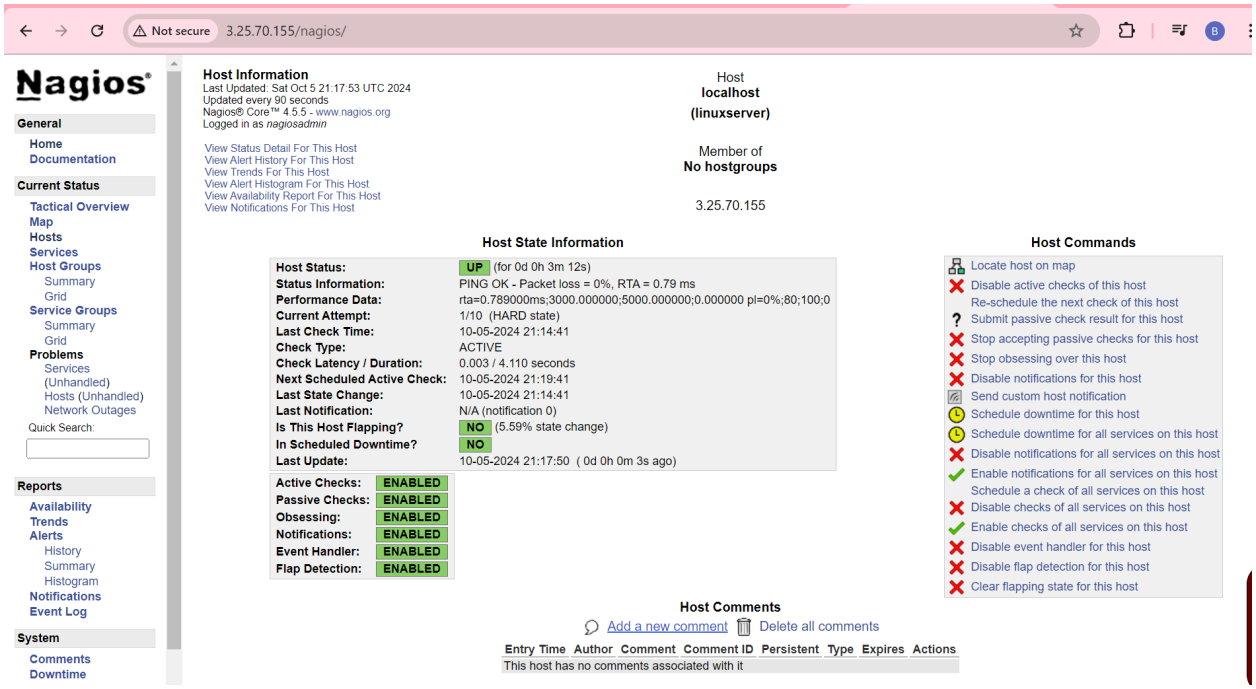
```
[root@ip-172-31-13-224 ec2-user]# sudo systemctl status httpd
sudo systemctl start httpd
sudo systemctl enable httpd
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
    Drop-In: /usr/lib/systemd/system/httpd.service.d
             └─php-fpm.conf
     Active: active (running) since Sat 2024-10-05 19:12:34 UTC; 51min ago
       Docs: man:httpd.service(8)
   Main PID: 49647 (httpd)
     Status: "Total requests: 24; Idle/Busy workers 100/0;Requests/sec: 0.00772; Bytes served/sec:  52 B/sec"
      Tasks: 230 (limit: 1112)
     Memory: 21.5M
        CPU: 1.806s
     CGroup: /system.slice/httpd.service
             ├─49647 /usr/sbin/httpd -DFOREGROUND
             ├─49654 /usr/sbin/httpd -DFOREGROUND
             ├─49655 /usr/sbin/httpd -DFOREGROUND
             ├─49656 /usr/sbin/httpd -DFOREGROUND
             ├─49657 /usr/sbin/httpd -DFOREGROUND
             └─77500 /usr/sbin/httpd -DFOREGROUND

Oct 05 19:12:34 ip-172-31-13-224.ap-southeast-2.compute.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...
Oct 05 19:12:34 ip-172-31-13-224.ap-southeast-2.compute.internal systemd[1]: Started httpd.service - The Apache HTTP Server.
Oct 05 19:12:34 ip-172-31-13-224.ap-southeast-2.compute.internal httpd[49647]: Server configured, listening on: port 80
[root@ip-172-31-13-224 ec2-user]#
```

15. Now to check Nagios dashboard go to http://<nagios-host-public-ip>/nagios.

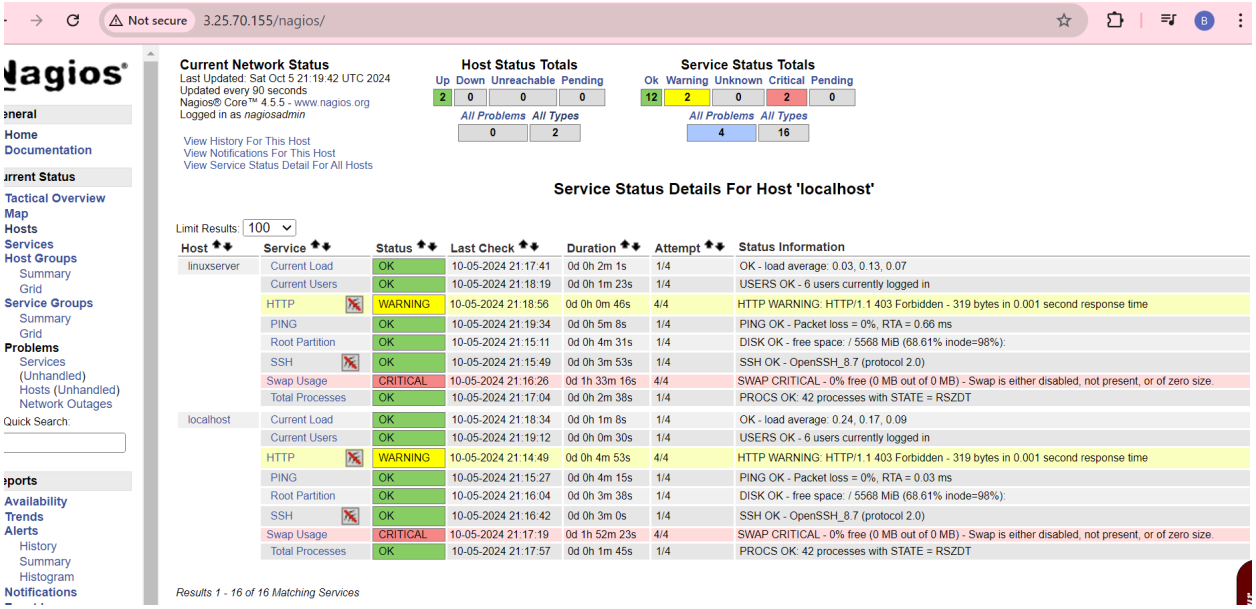
Now Click on Hosts from left side panel



We can see our linuxserver now click on it and we can see the host information.



Current Network Status



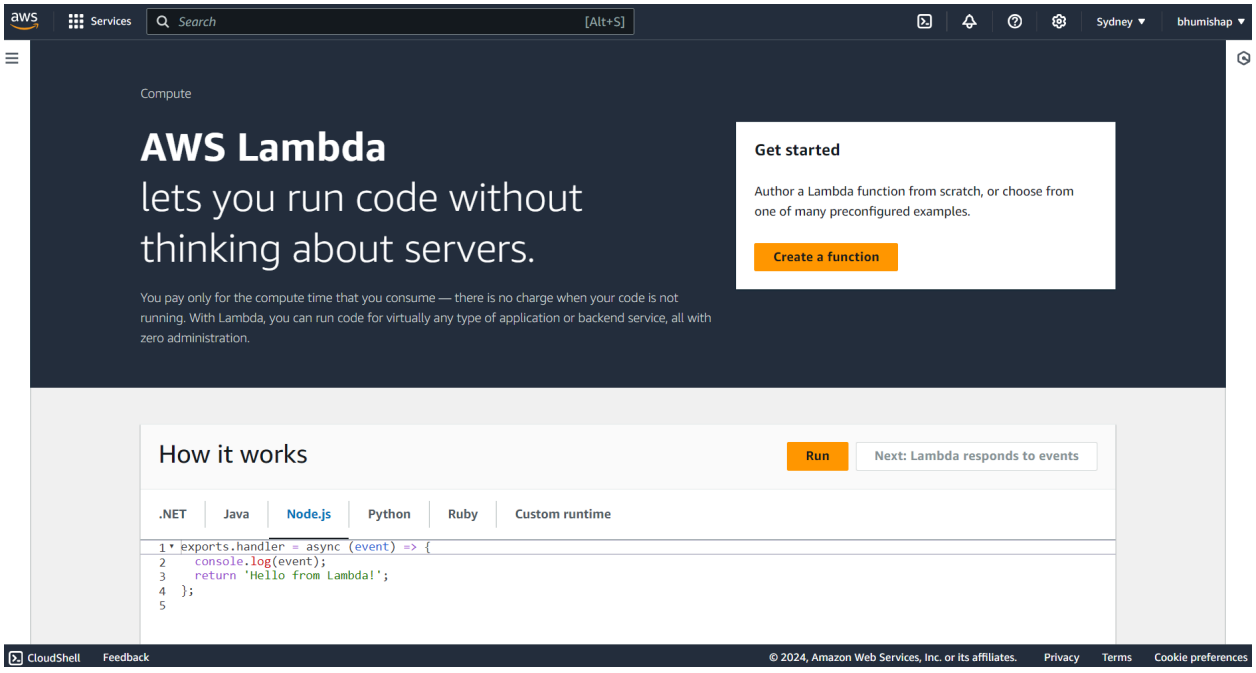
Conclusion: In conclusion, the experiment focused on monitoring ports, services, and a Linux server using Nagios. Through the step-by-step process, we successfully configured Nagios to monitor essential network services on the Linux server. By setting up both the Nagios host and client, we were able to track system performance, ensure service availability, and monitor key metrics like CPU and memory usage.

EXPERIMENT NO. 11

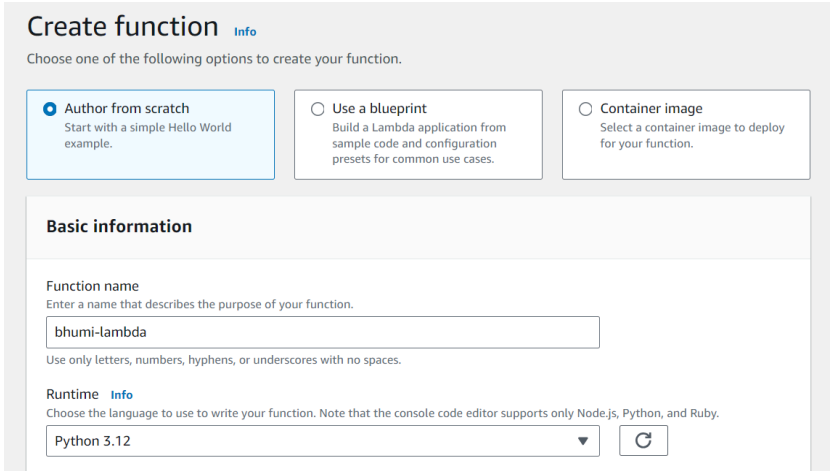
Aim: To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

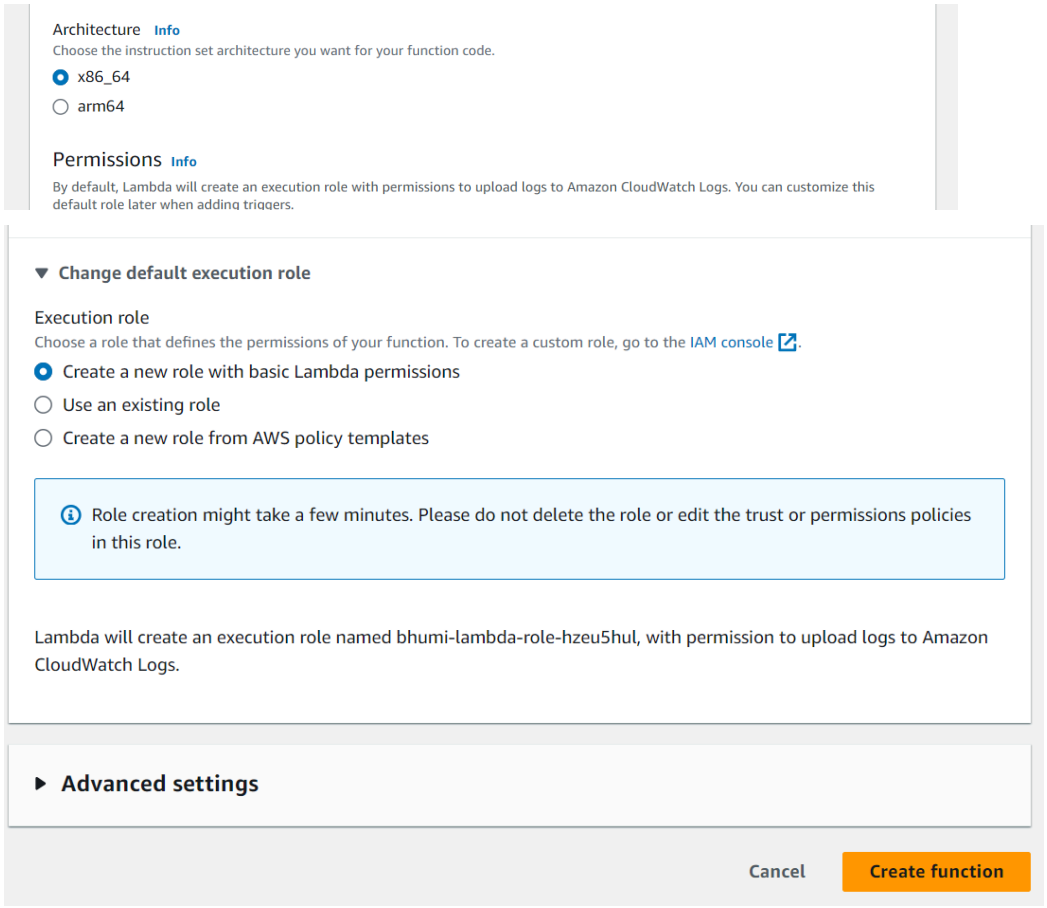
Steps To create the lambda function:

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

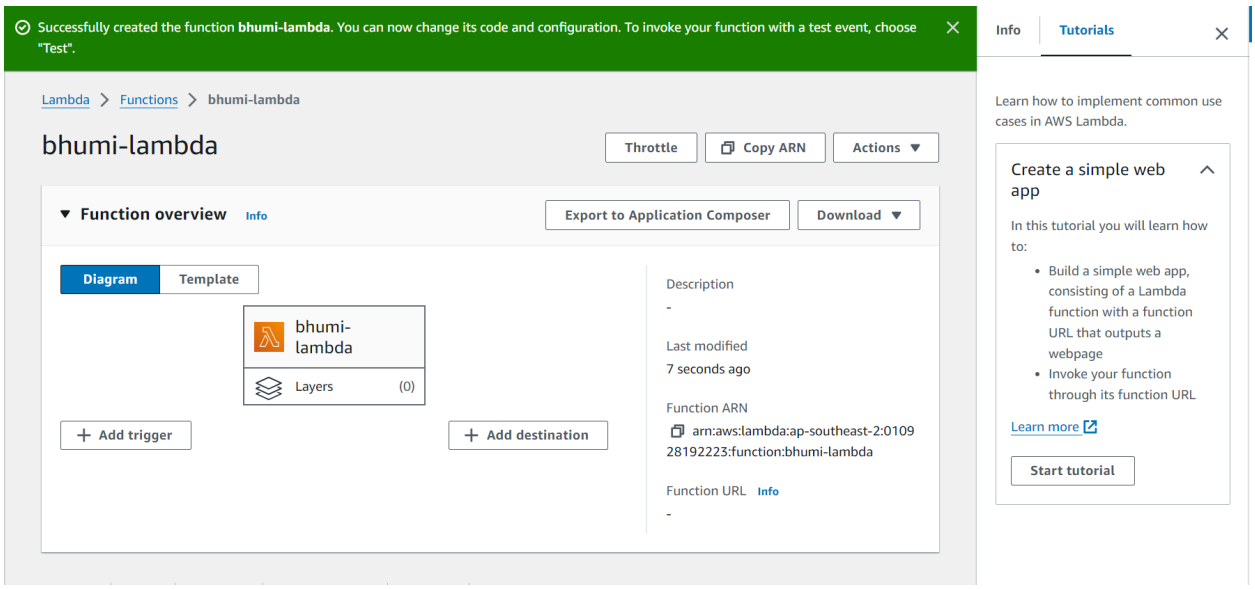


2. Now Give a name to your Lambda function.
   Select the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby. So we will select Python 3.12, Architecture as x86, and Execution role to Create a new role with basic Lambda permissions

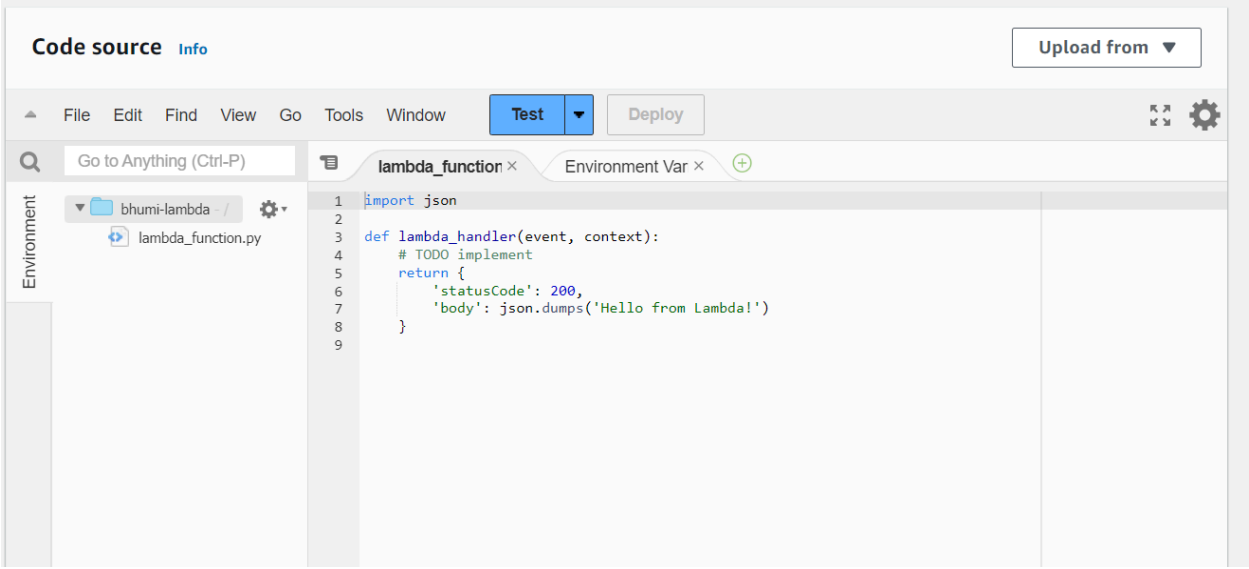




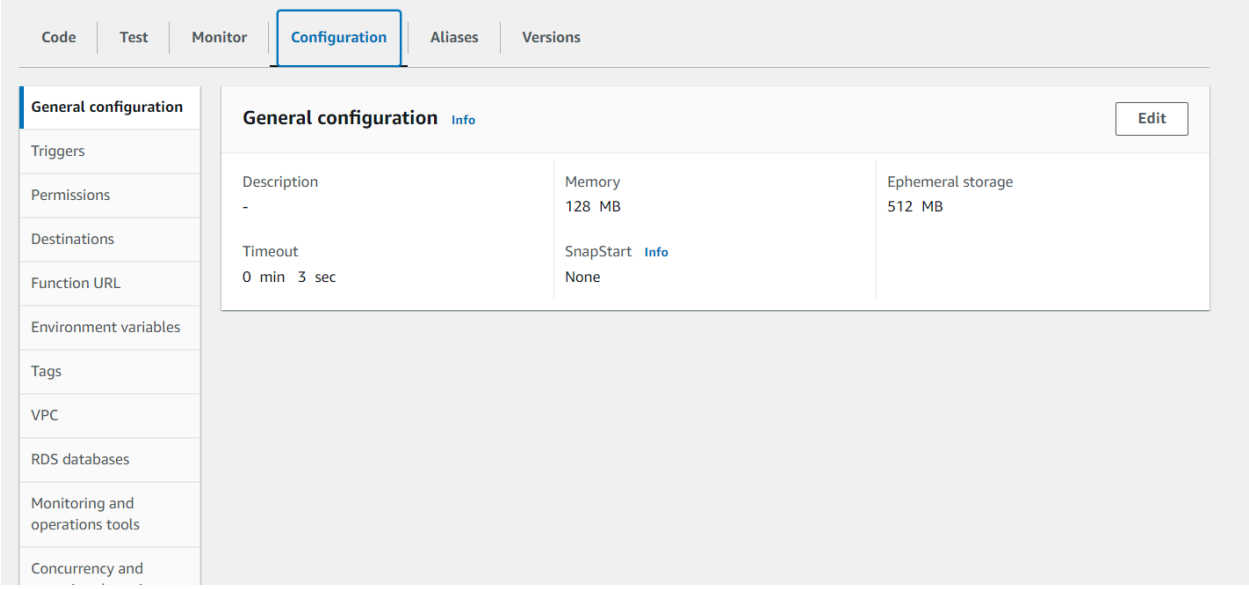
Thus, we have successfully created a lambda function named bhumi-lambda.



This is how the function is displayed.

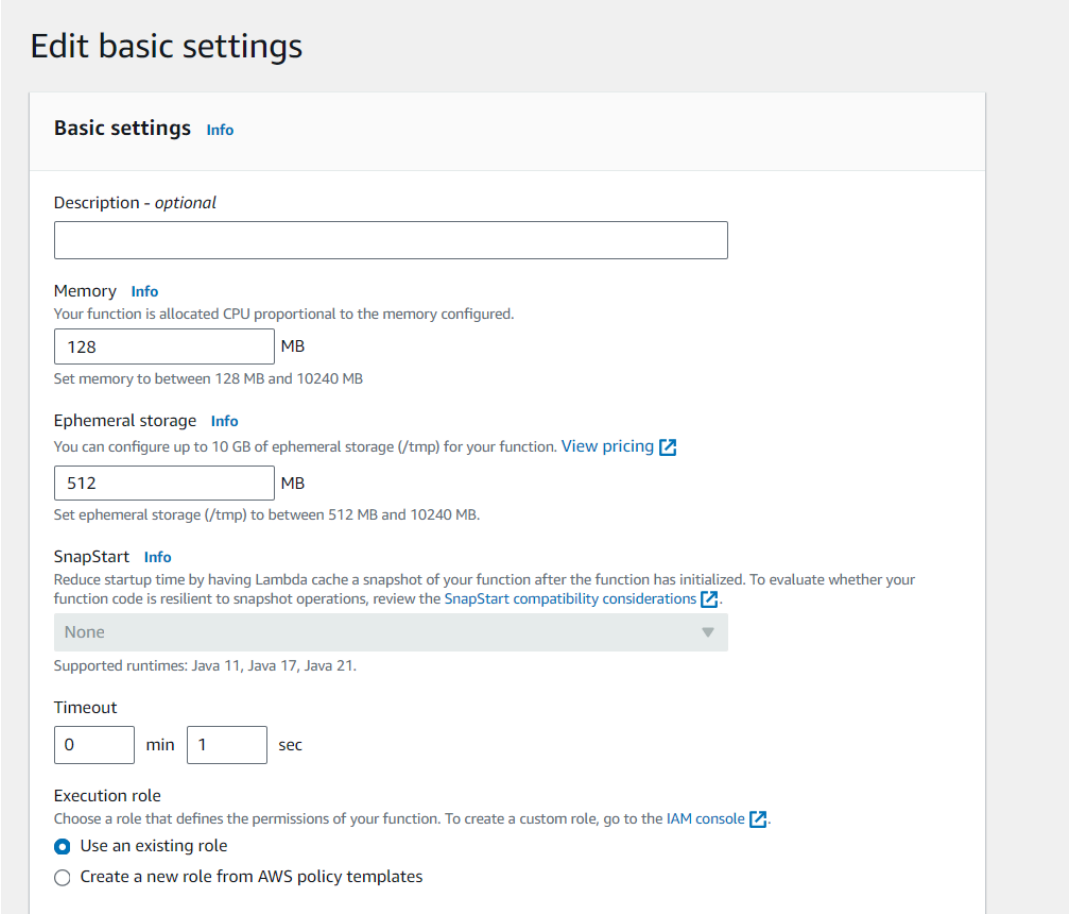


Go to the Configuration tab to see the general configuration of our function.

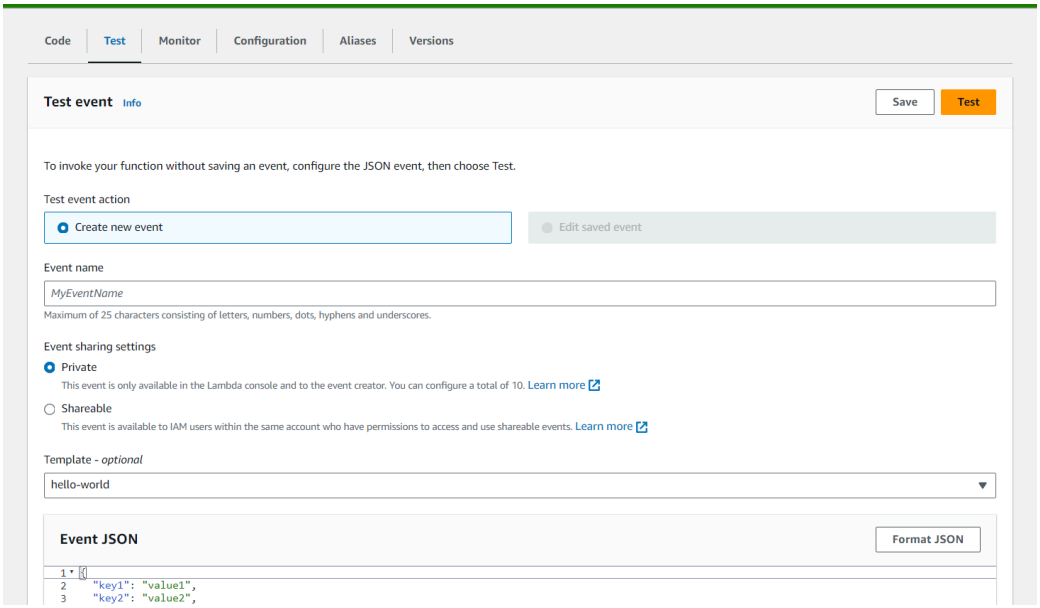


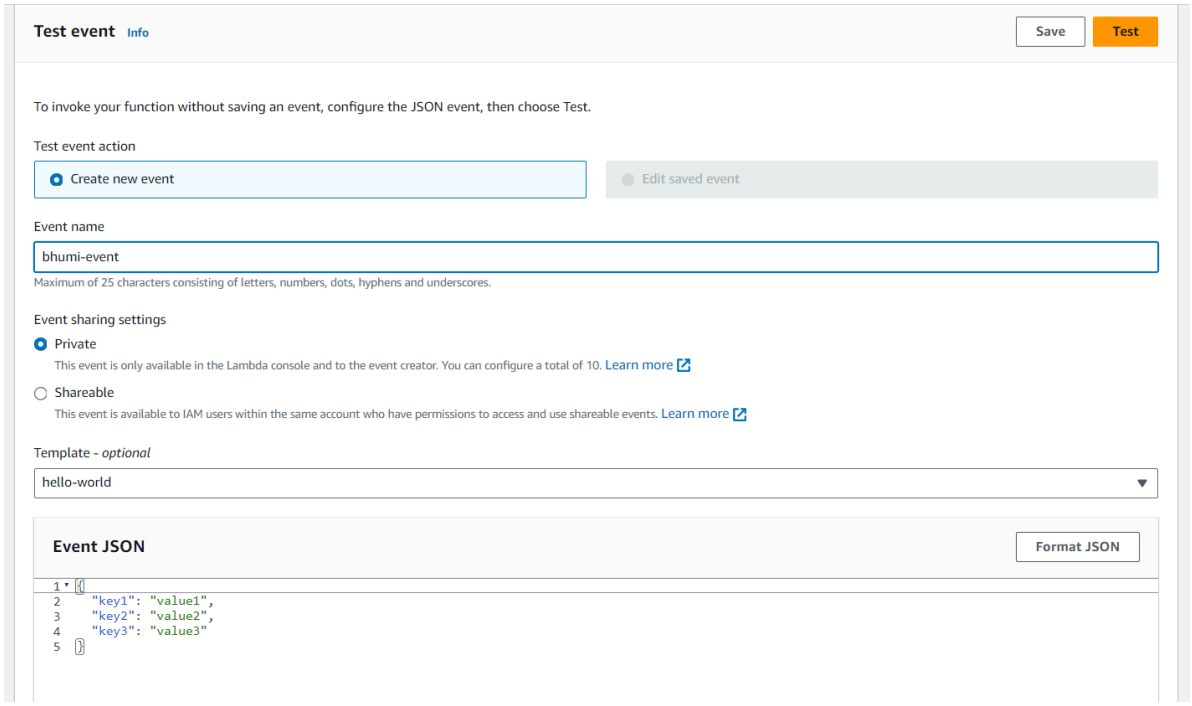
We want to edit the timeout time and the rest can be kept the default.
Here, we can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.

To edit, we go to the Edit button seen in the above screenshot.

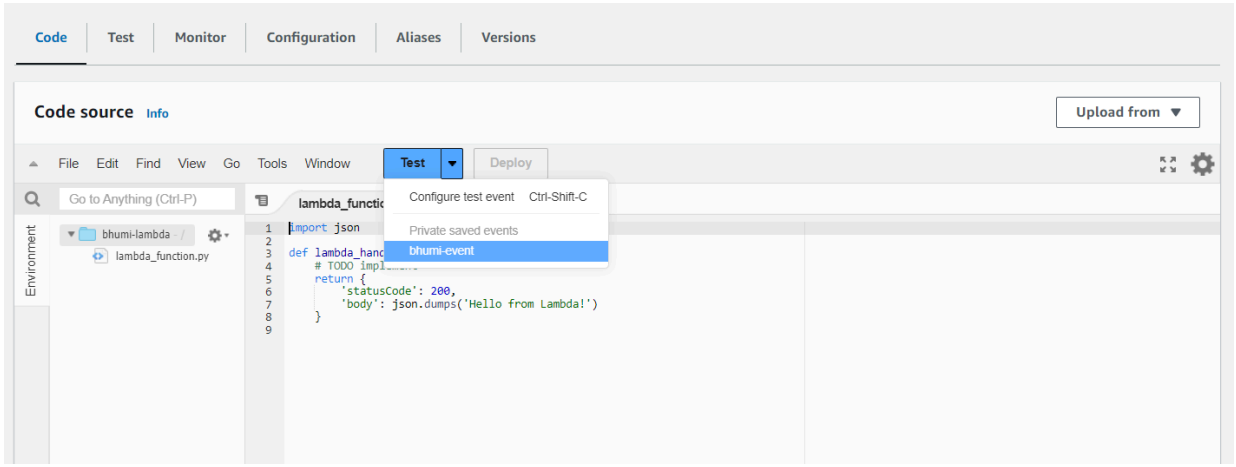


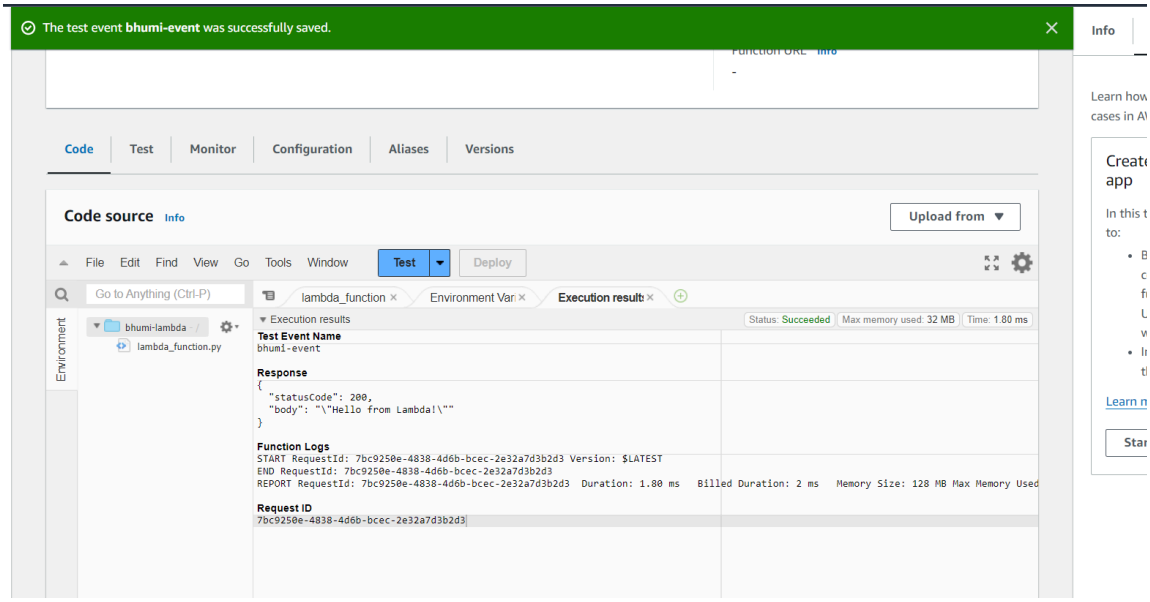
3. Now Click on the Test tab then select *Create a new event,* give a name to the event and select Event Sharing to private and select hello-world template.



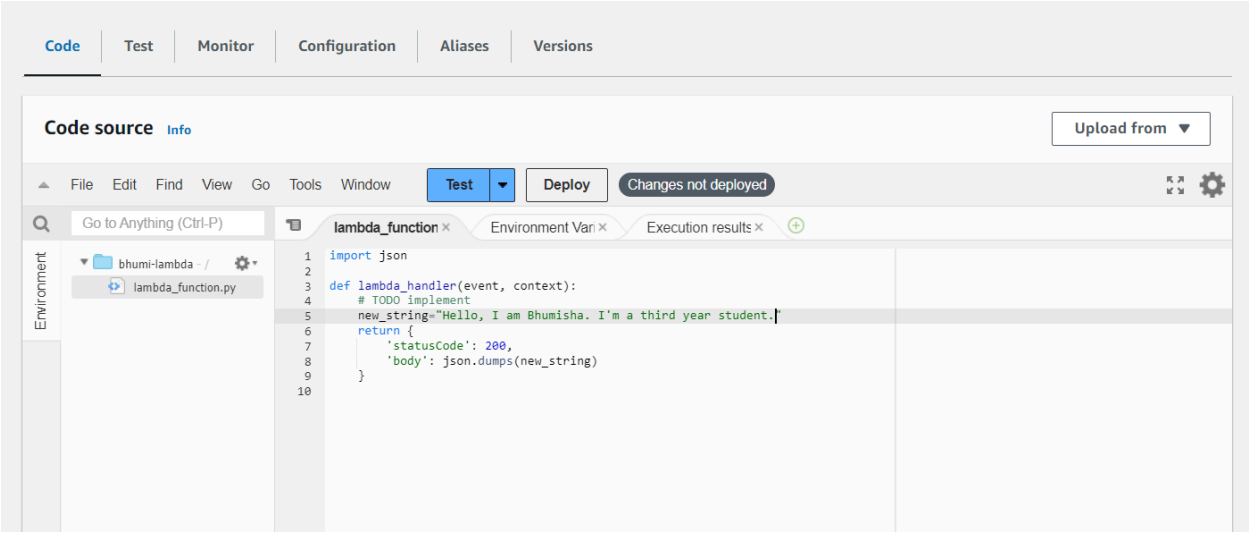


4. Now in the Code section, select the created event from the dropdown of test then click on test . You will see the below output.

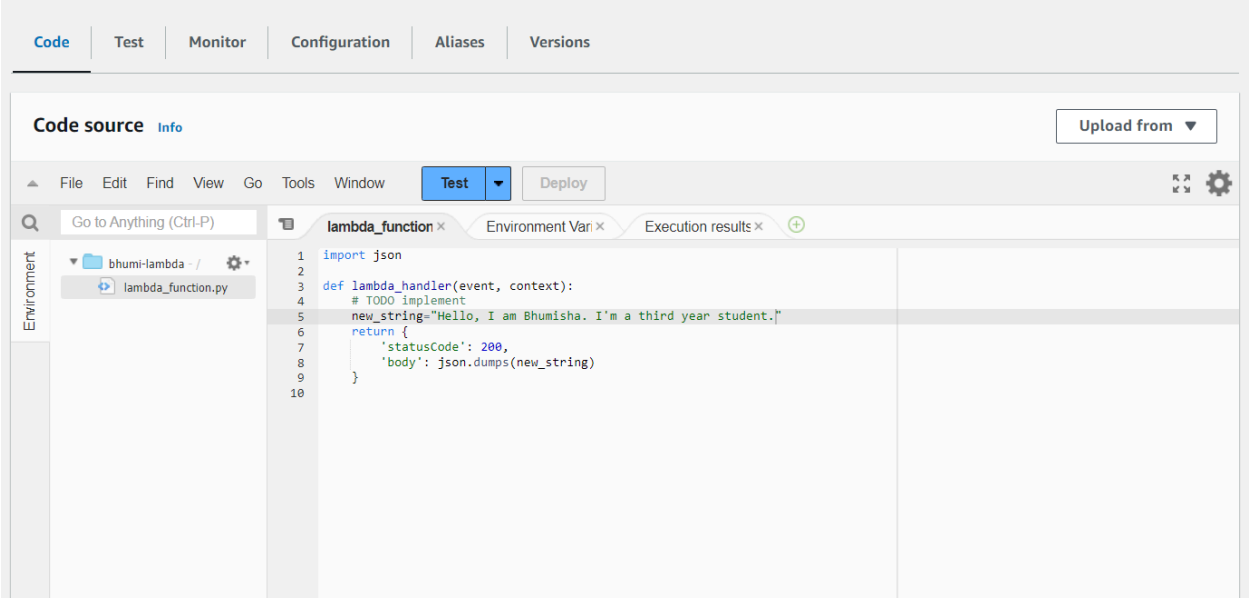




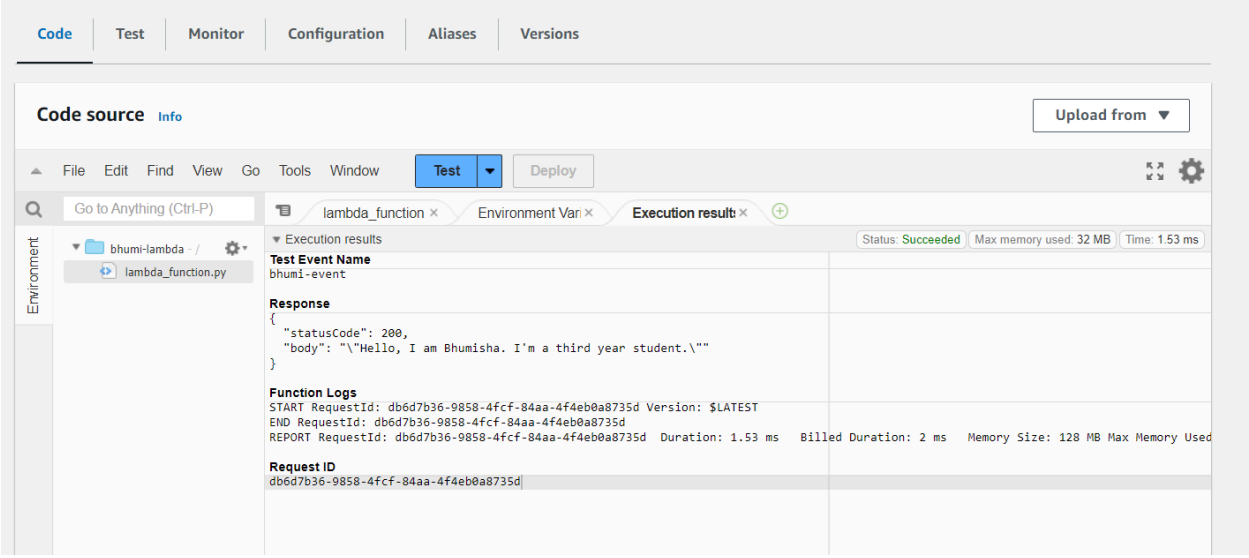
5. You can edit your lambda function code. I have changed the code to display the new string.



Now ctrl+s to save and click on deploy to deploy the changes.



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



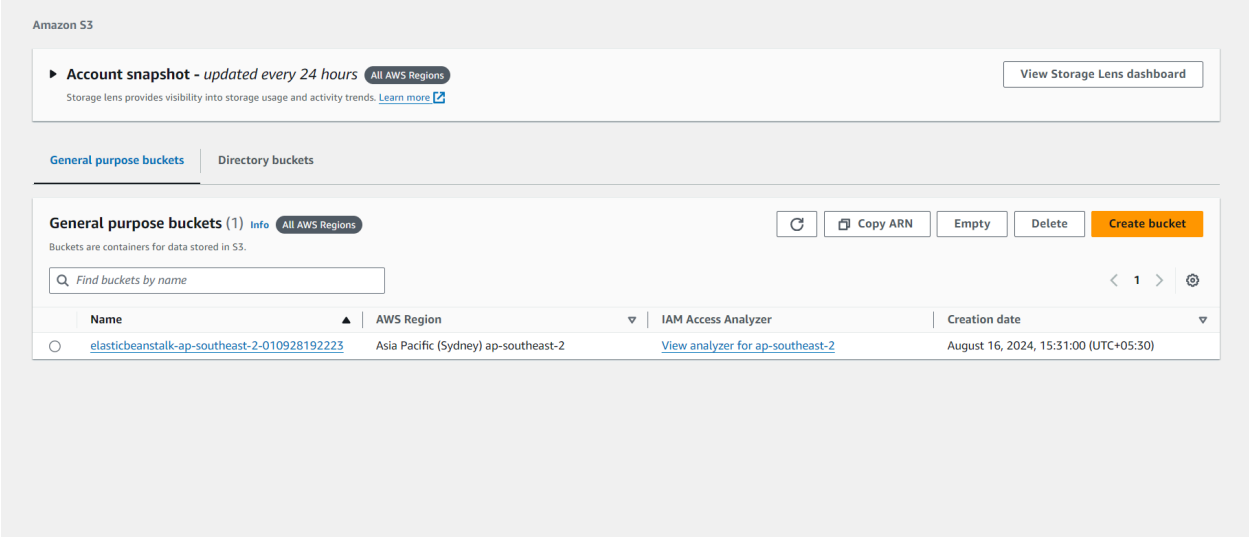
Conclusion: In this experiment, we successfully created an AWS Lambda function and walked through its essential steps. After setting up the function with Python, we configured the basic settings, including adjusting the timeout to 1 second. We then created a test event, deployed the function, and validated the output. Additionally, we modified the Lambda function's code and redeployed it to observe the changes in real-time.

This practical experience demonstrated the simplicity and flexibility of AWS Lambda in creating serverless applications, allowing you to focus on code while AWS manages the infrastructure and scaling

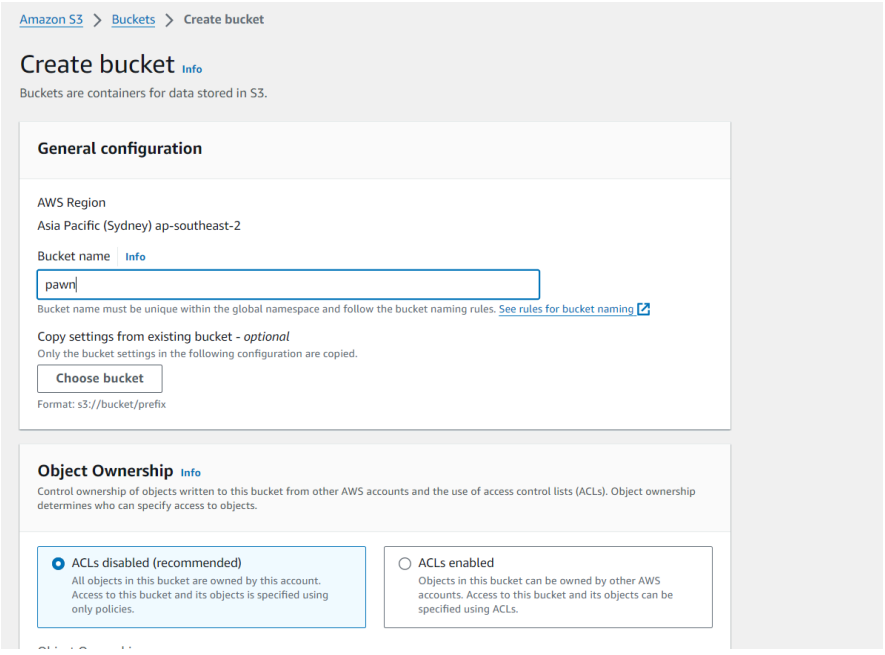
EXPERIMENT NO. 12

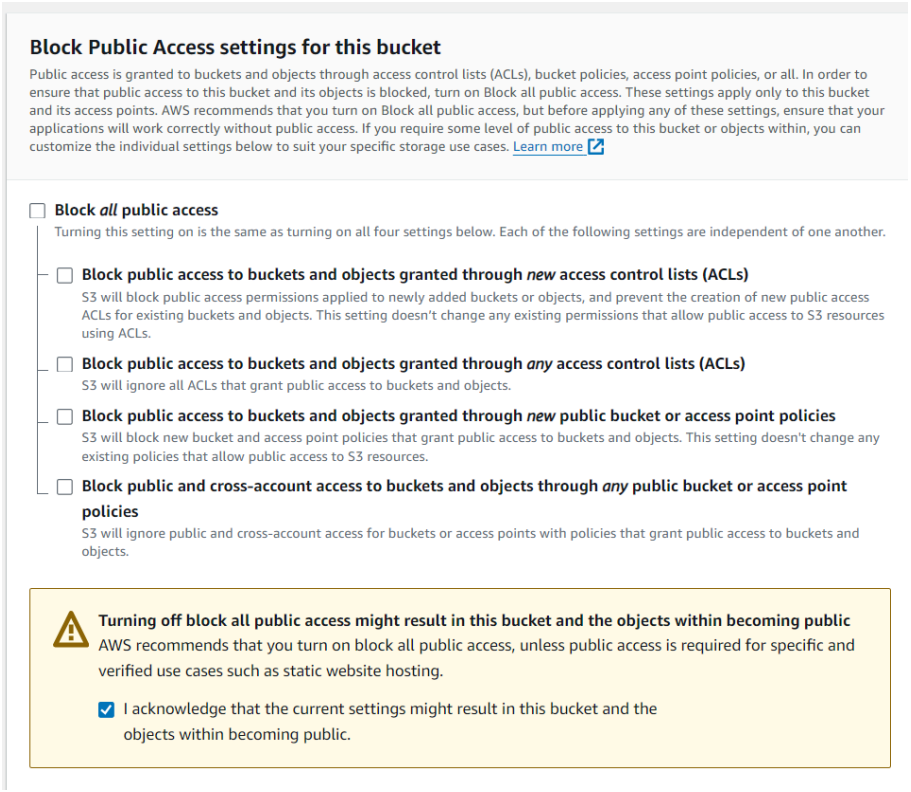
Aim: To create a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3.

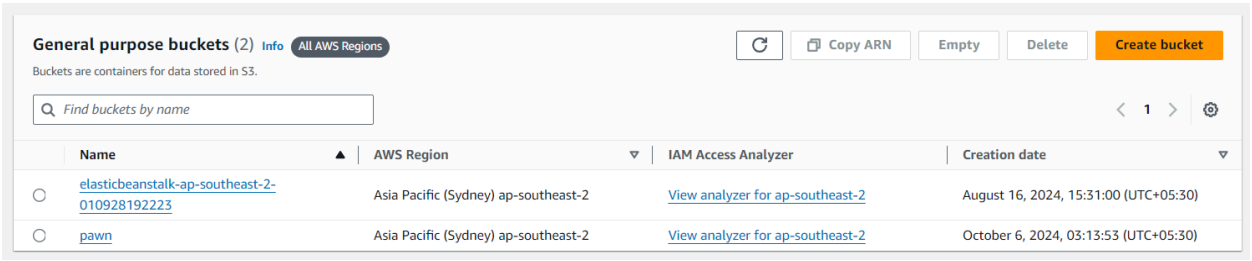
1. Login to your AWS Personal account. Now open S3 from services and click on create S3 bucket.



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

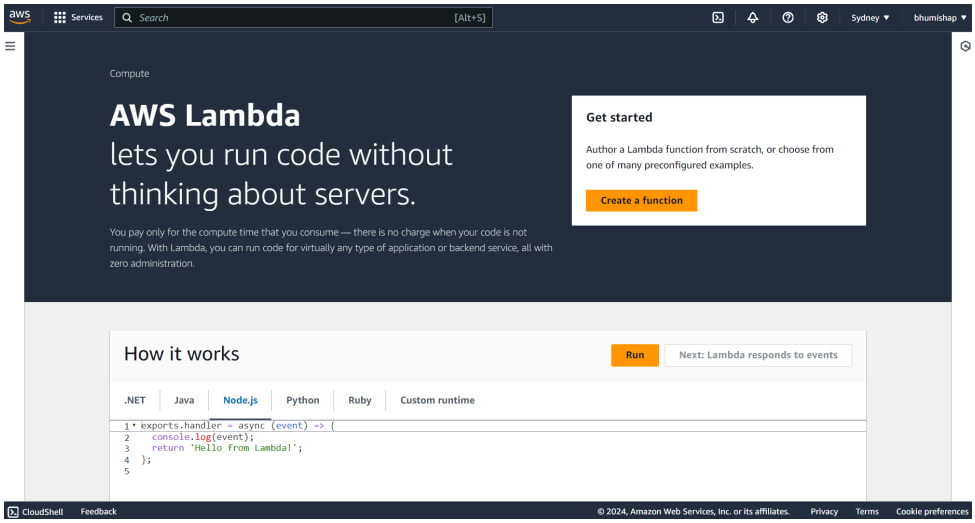




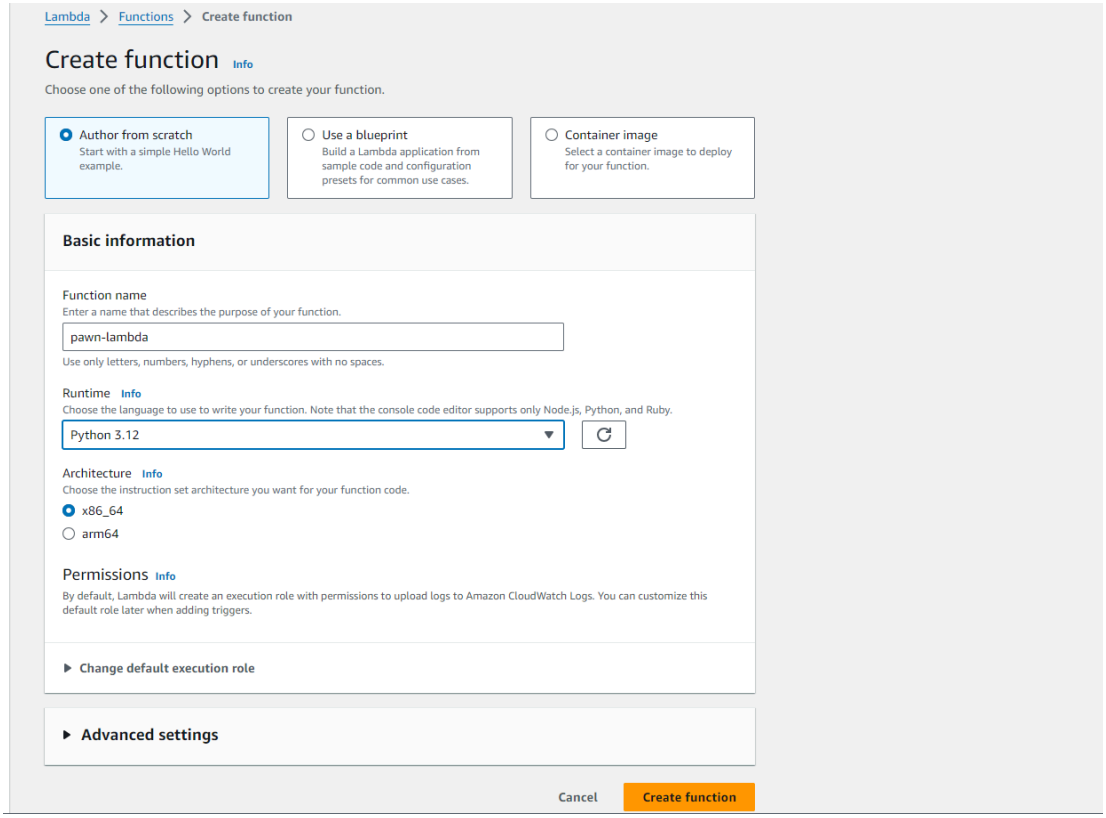


Thus, we have created a bucket named *pawn*.

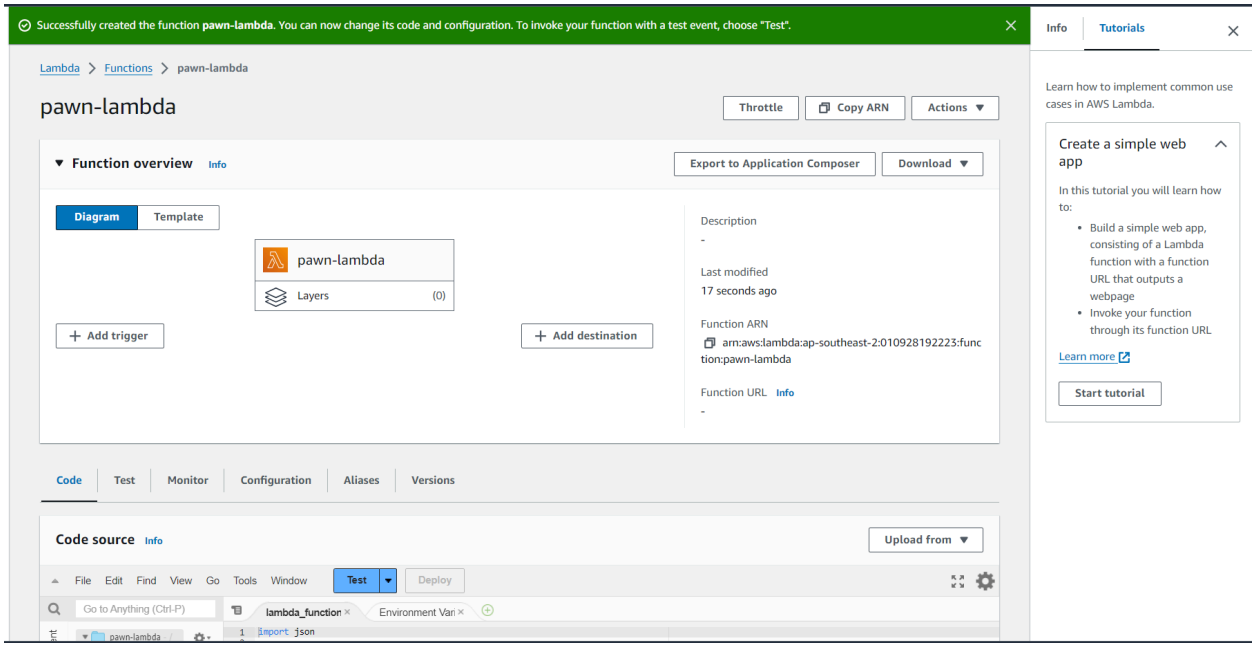
3. Open lambda console and click on the create function button.



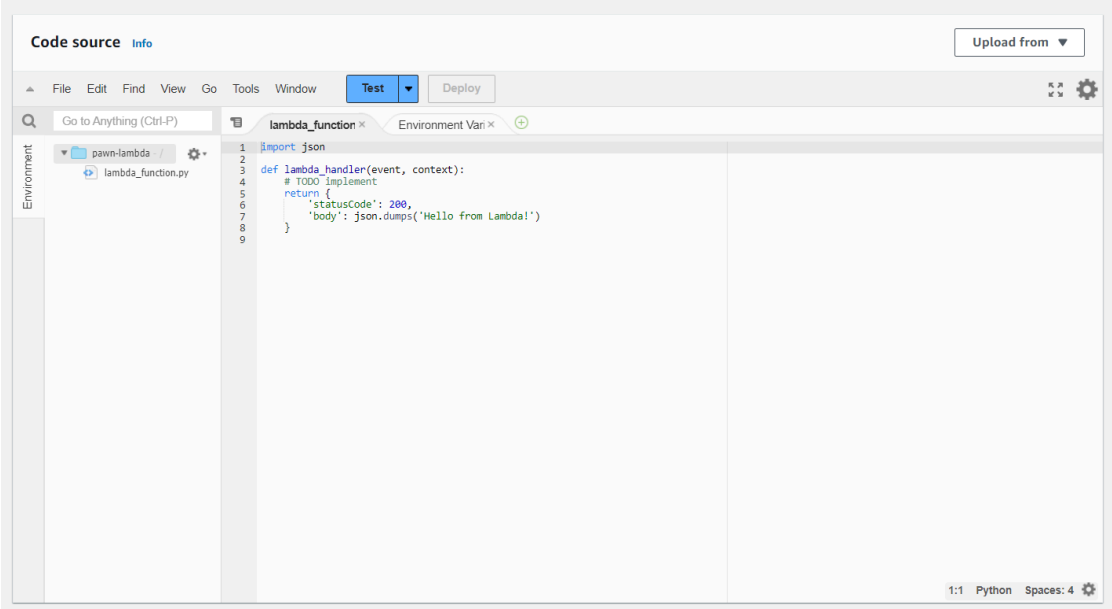
4. Now Give a name to your Lambda function, Select the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby. So will select Python 3.12, Architecture as x86 and Exceution role to Create a new role with basic Lambda permissions



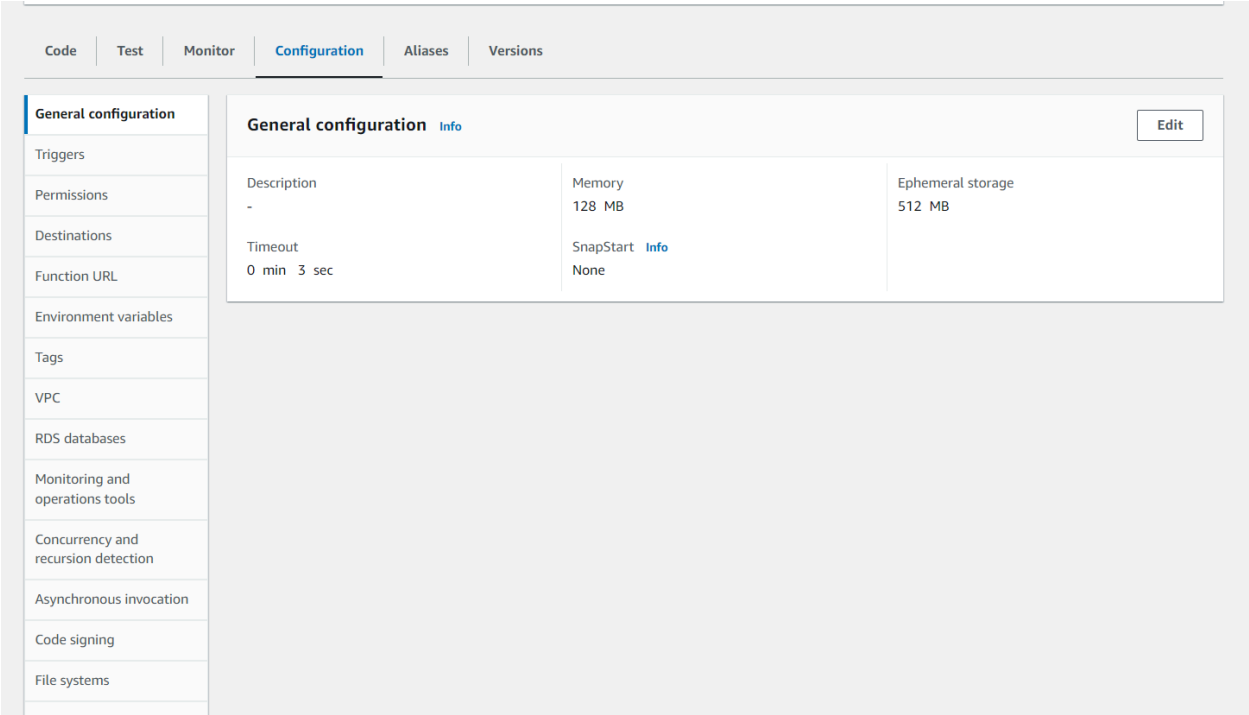
Thus, we have successfully created a lambda function named *pawn-lambda*.



This is how the function is displayed.

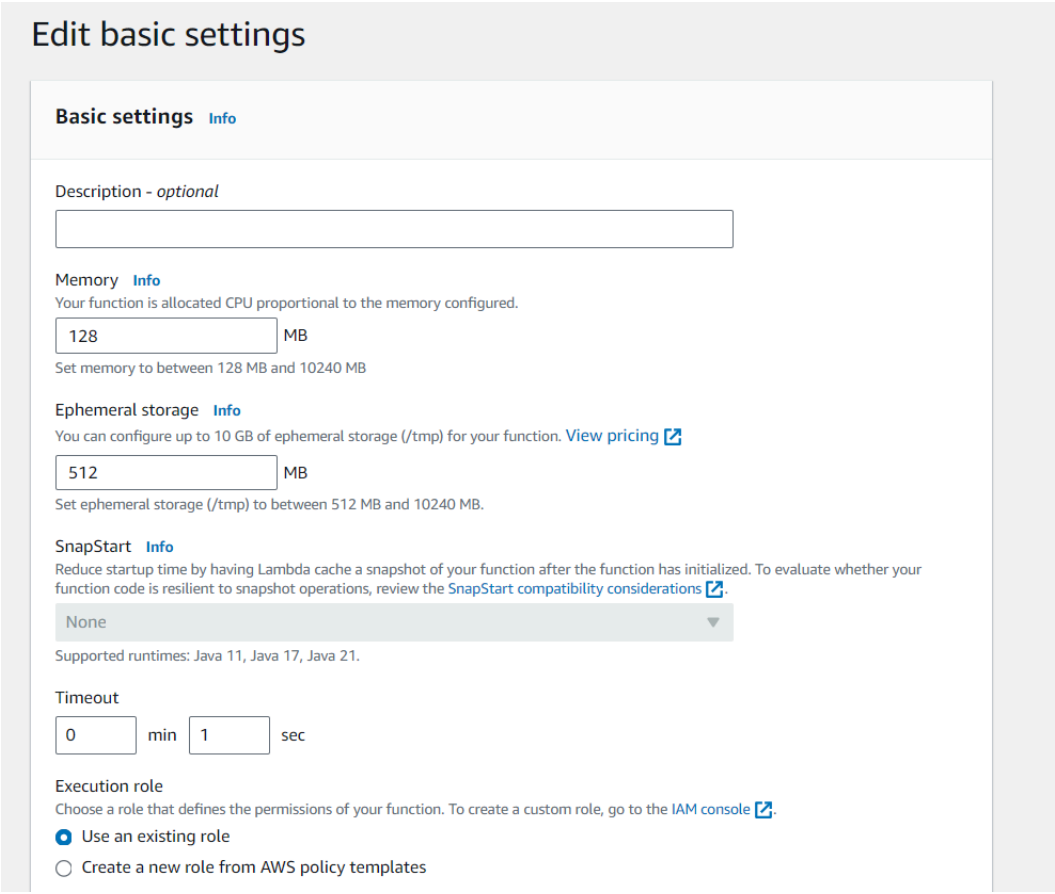


Go to the Configuration tab to see the general configuration of our function.

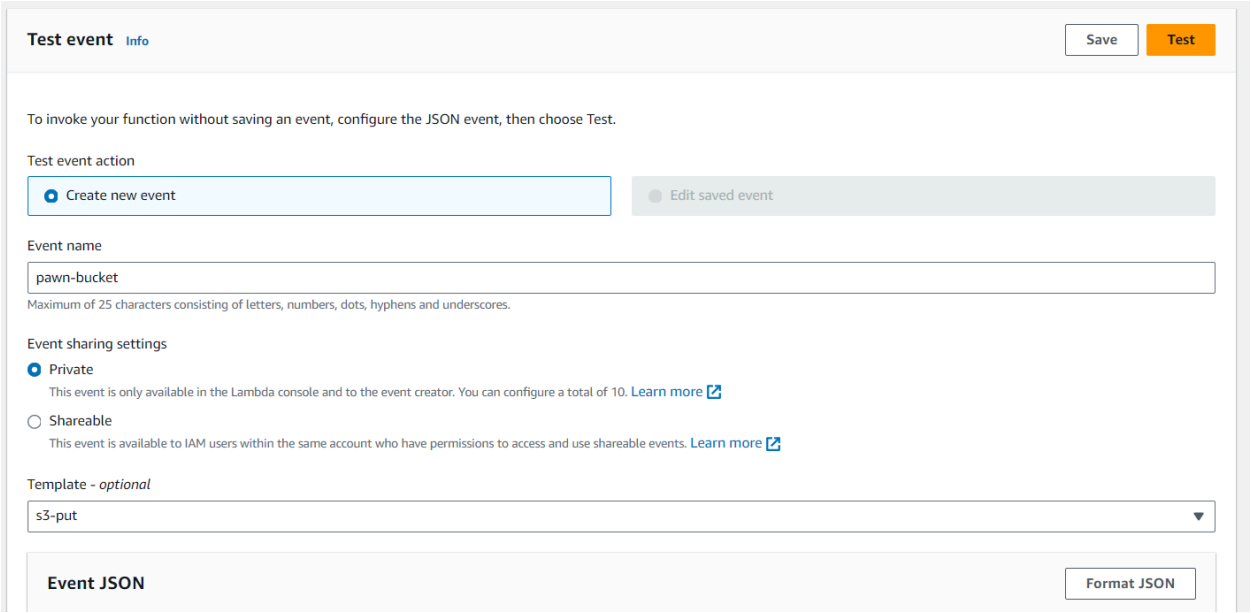


We want to edit the timeout time and the rest can be kept the default.
Here, we can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.

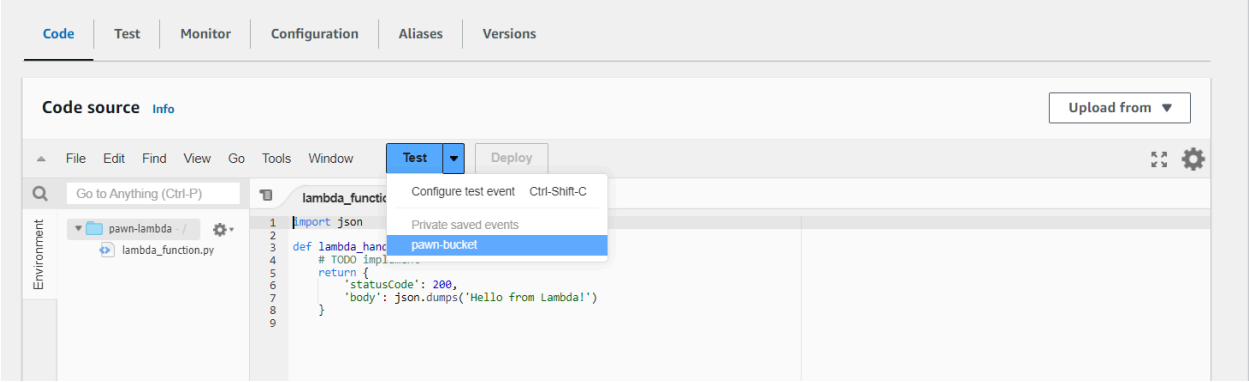
To edit, we go to the Edit button seen in the above screenshot.



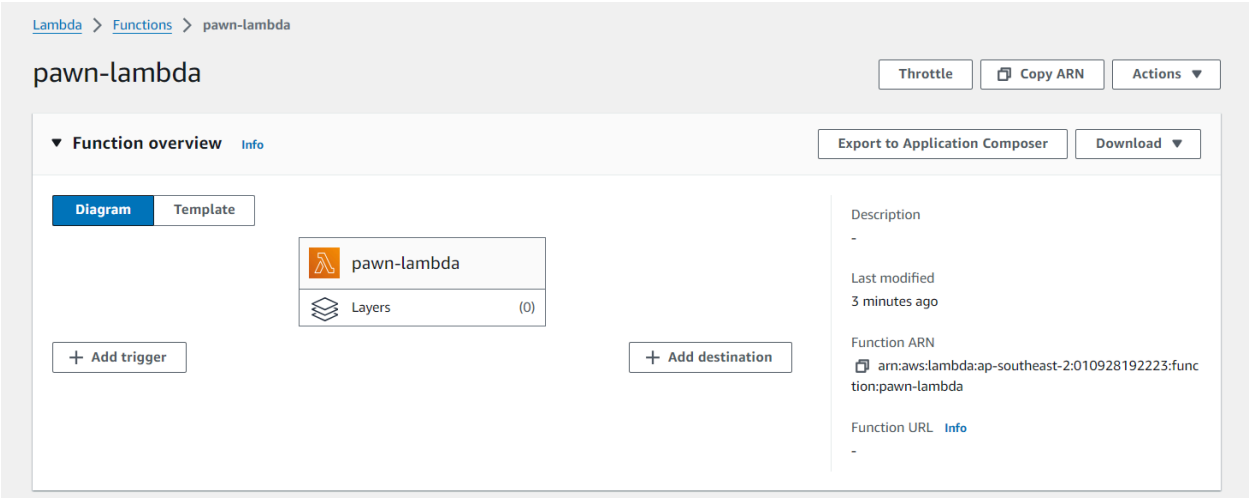
5. Now Click on the Test tab then select Create a new event, give a name to the event and select Event Sharing to private, and select s3 put template.



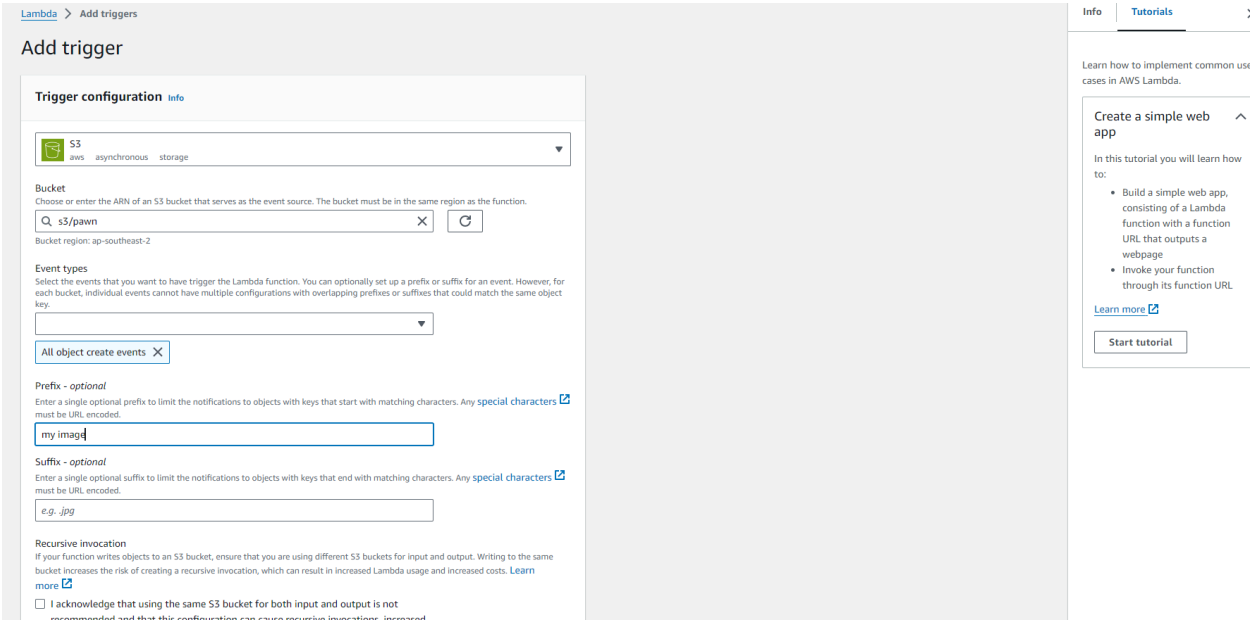
6. Now in the Code section select the created event from the dropdown.

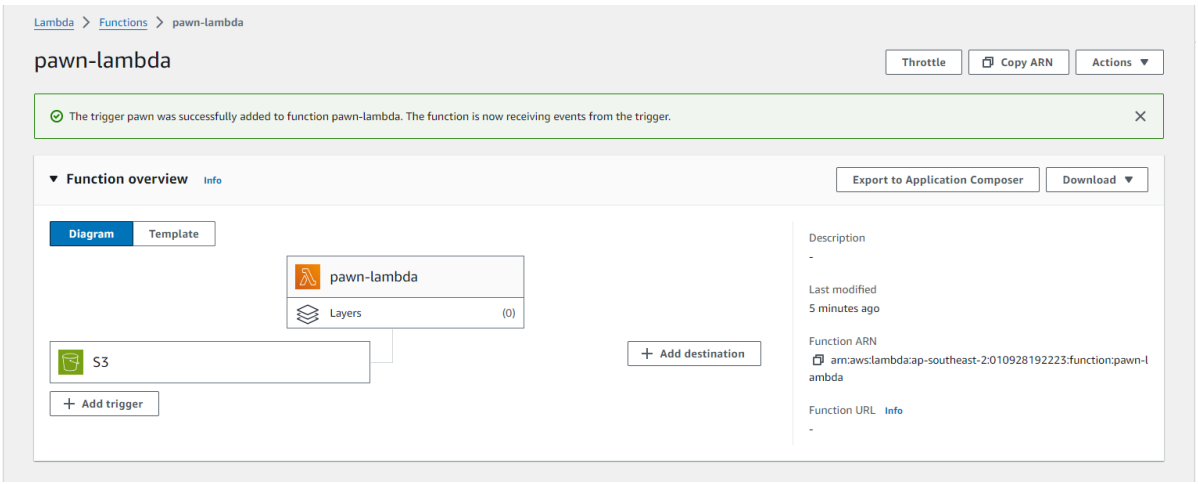


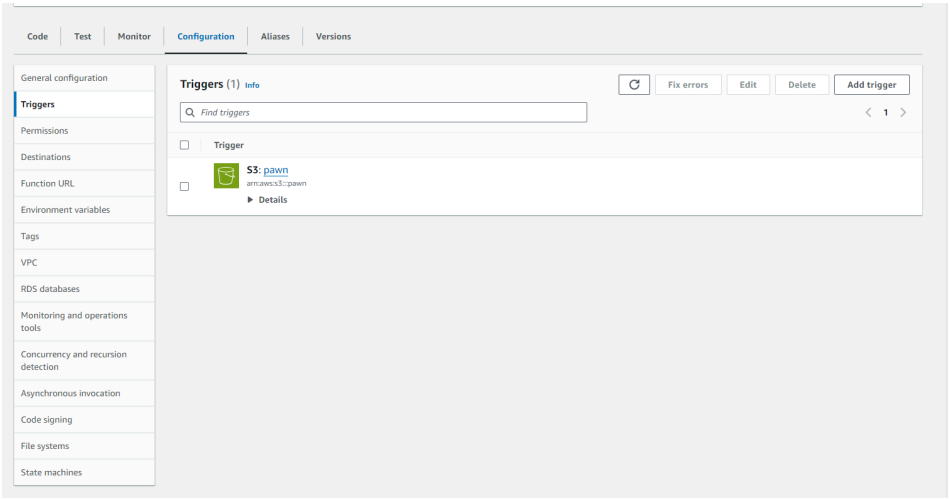
7. Now In the Lambda function click on add trigger.



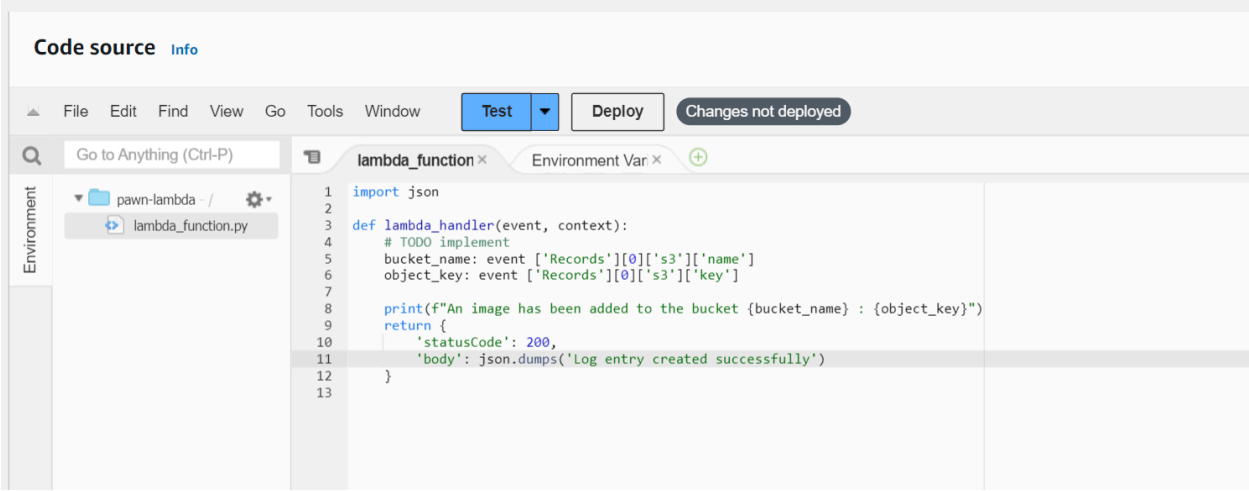
Now select the source as S3 then select the bucket name from the dropdown, keep other things to default and also you can add prefix to image

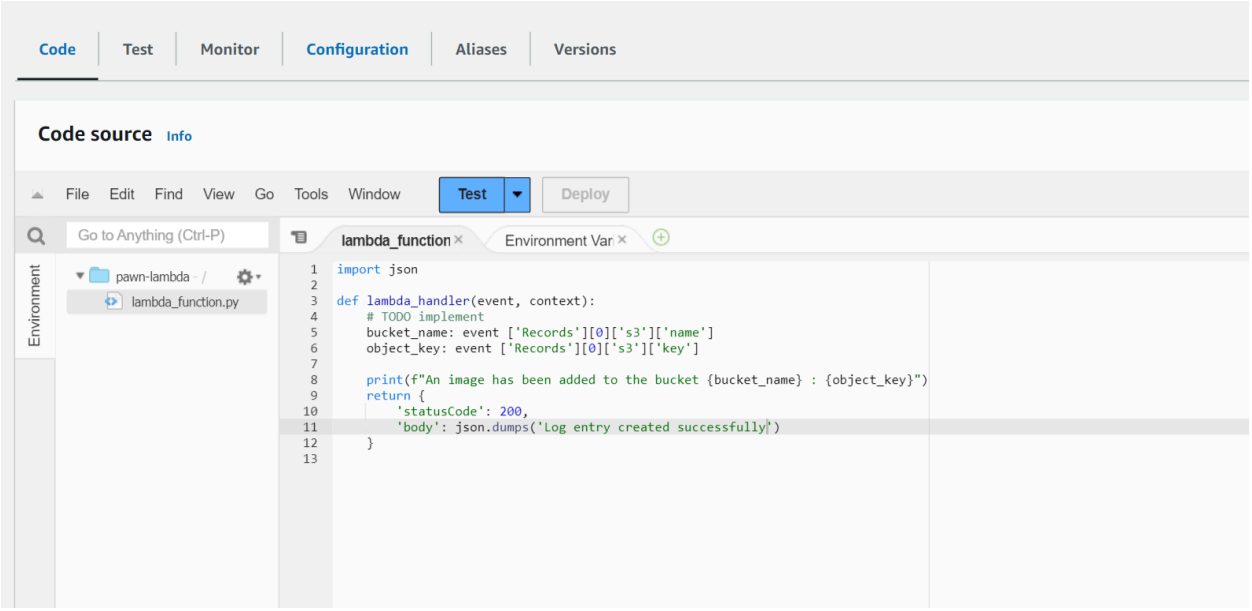




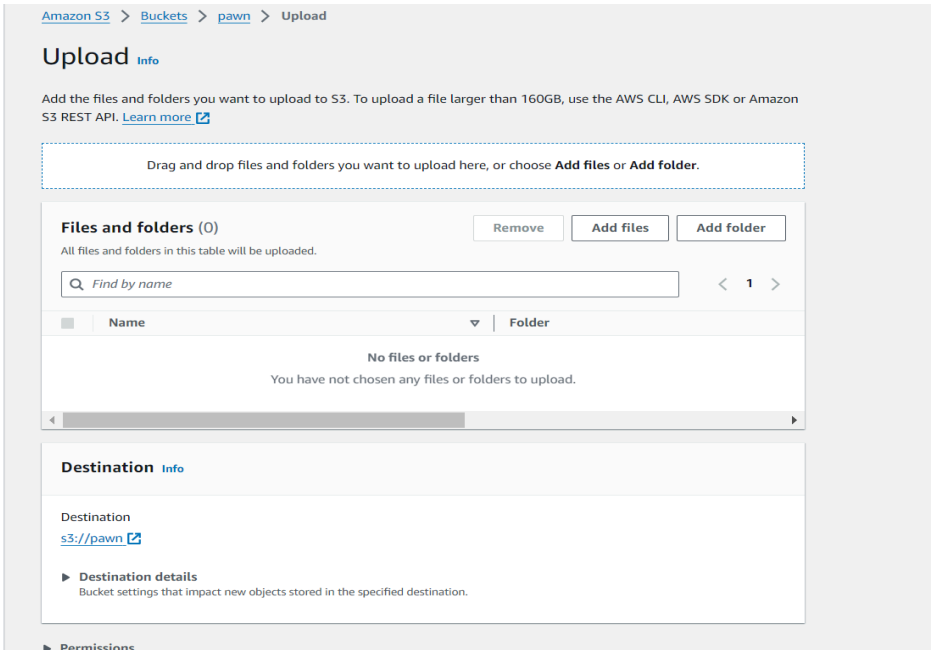


8. Now Write code that logs a message like “An Image has been added” when triggered. Save the file and click on deploy.





9. Now upload any image to the bucket.



Upload Info

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more

Drag and drop files and folders you want to upload here, or choose **Add files** or **Add folder**.

Files and folders (1 Total, 77.3 KB)

Remove | Add files | Add folder

All files and folders in this table will be uploaded.

Find by name

1

Name	Folder
Screenshot 2024-09-10 220423.png	-

Destination Info

Destination

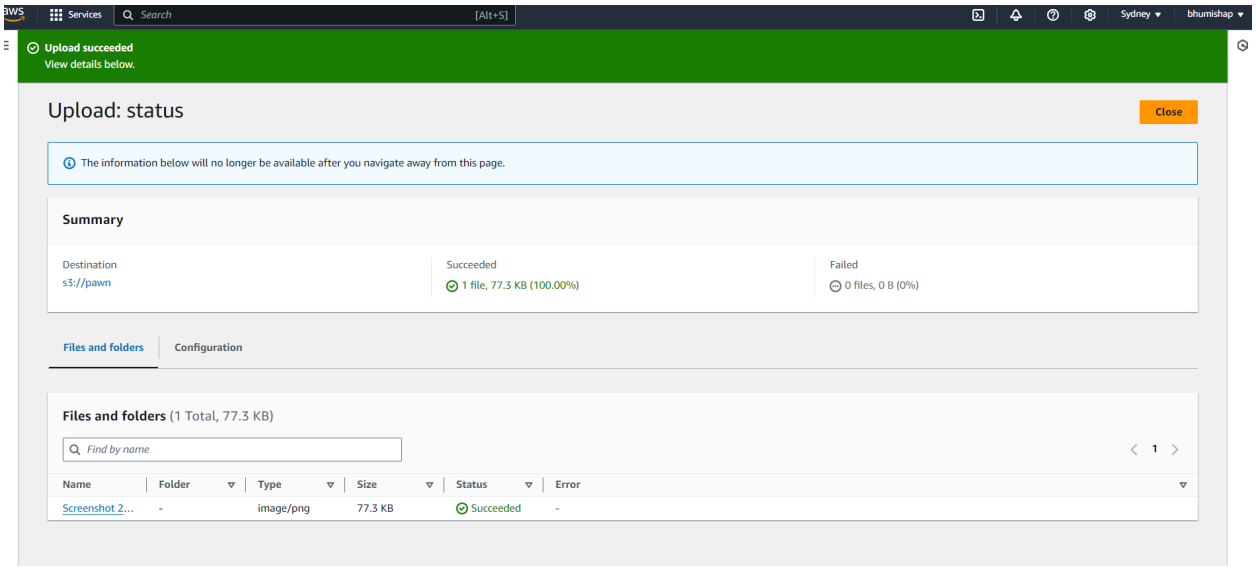
s3://pawn

▶ **Destination details**

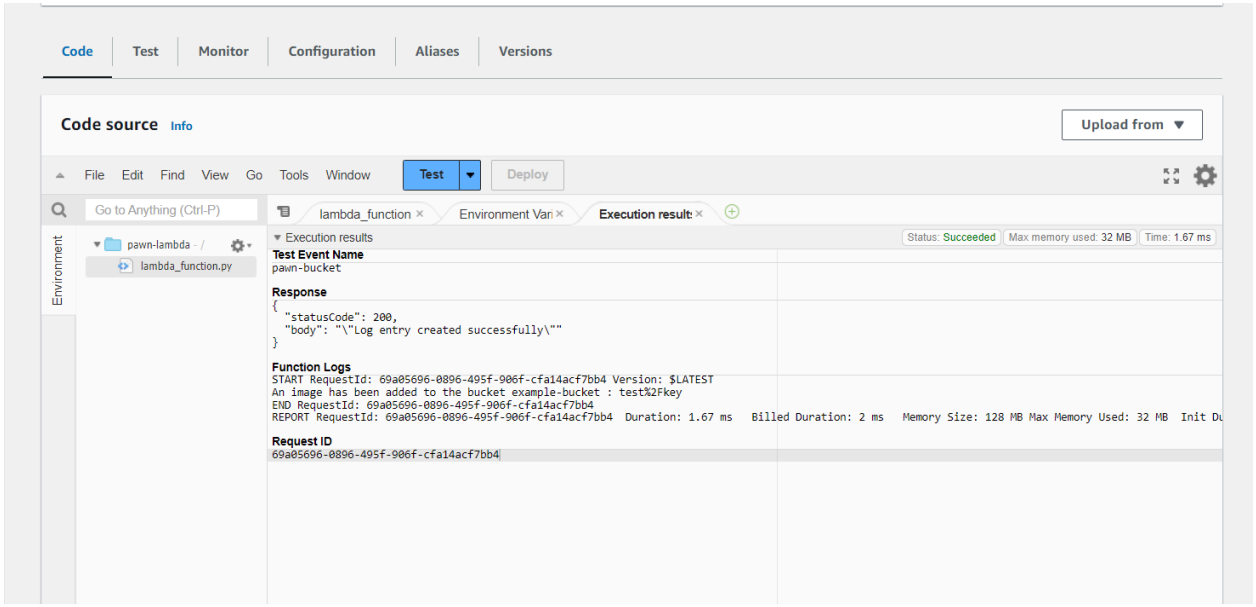
Bucket settings that impact new objects stored in the specified destination.

▶ **Permissions**

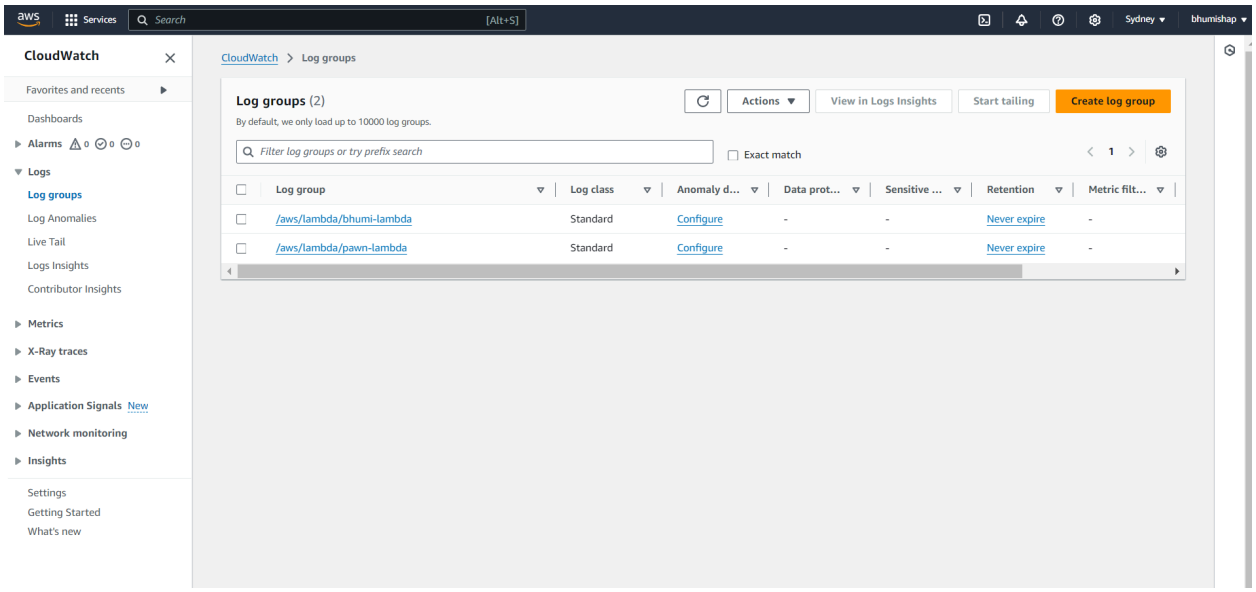
Grant public access and access to other AWS accounts.

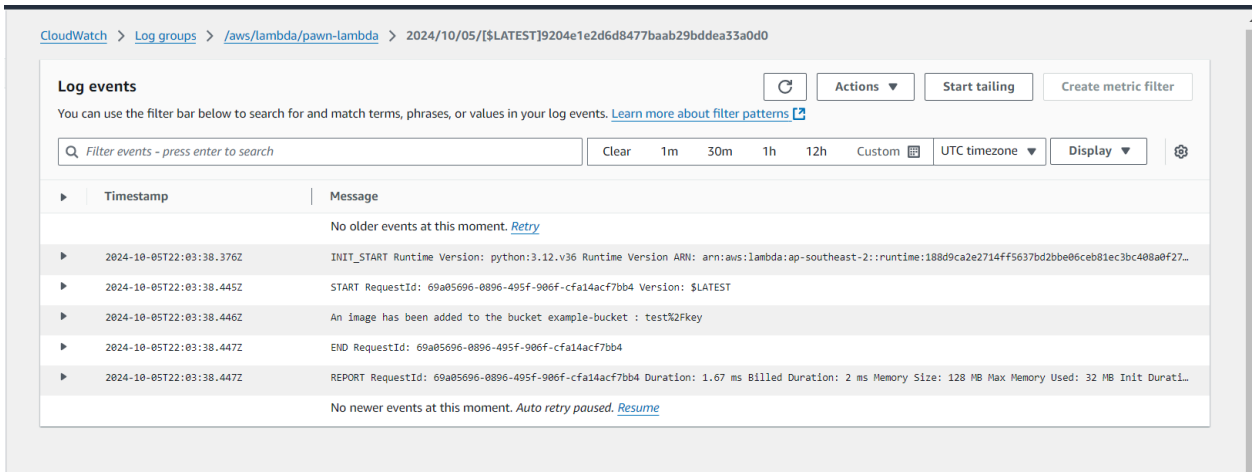


10. Now click on the test in lambda to check whether it is giving log when image is added to S3.



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





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