Experiment 1:

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

1. Find a web server that supports PHP and MYSQL

XAMPP is an easy to install Apache distribution containing MariaDB, PHP, and Perl. Just download and start the installer. It's that easy. Installers created using InstallBuilder.

XAMPP for **Windows** 8.0.30, 8.1.25 & 8.2.12

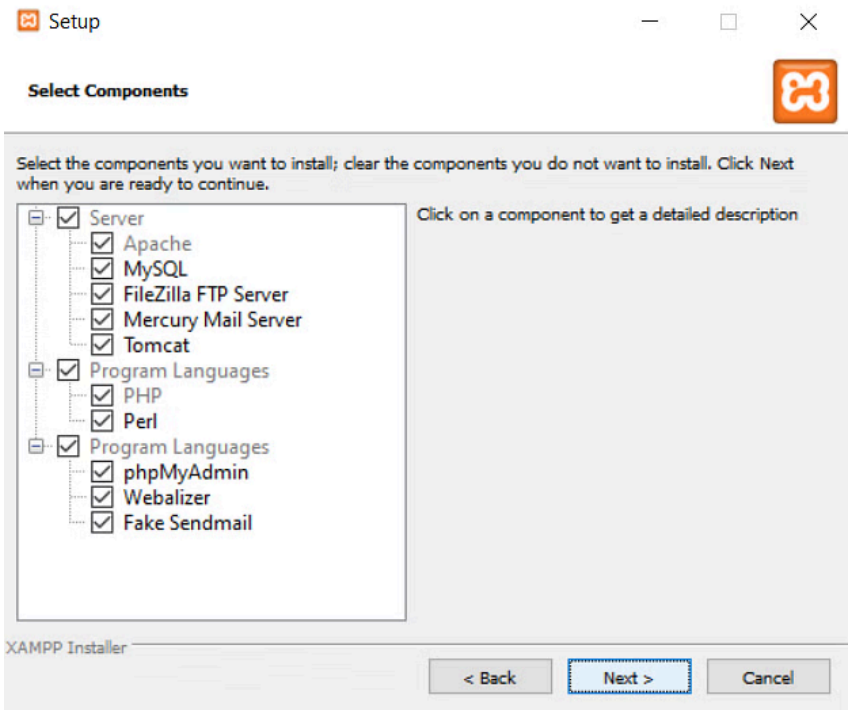
Version		Checksum			Size
8.0.30 / PHP 8.0.30	What's Included?	md5	sha1	Download (64 bit)	144 Mb
8.1.25 / PHP 8.1.25	What's Included?	md5	sha1	Download (64 bit)	148 Mb
8.2.12 / PHP 8.2.12	What's Included?	md5	sha1	Download (64 bit)	149 Mb

Requirements More Downloads »

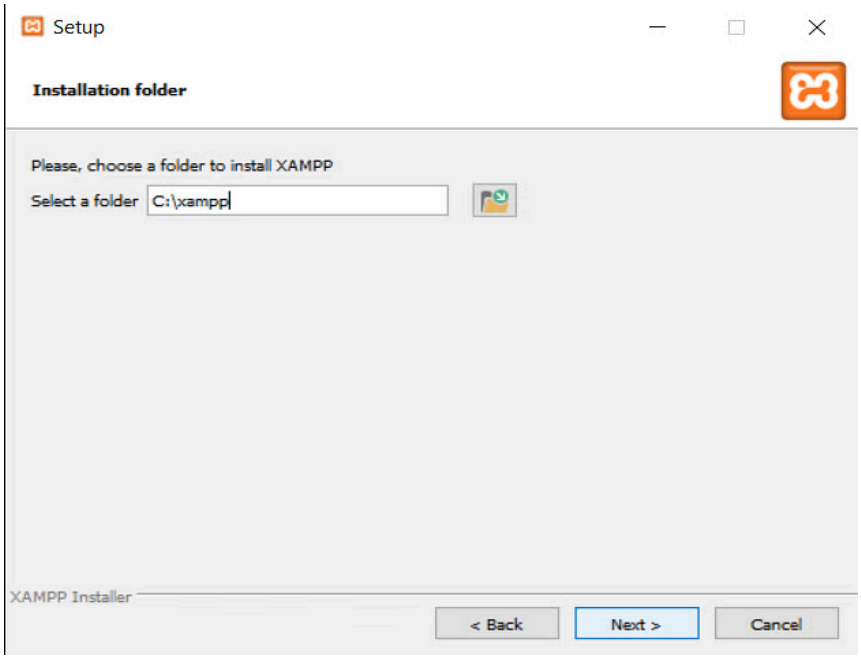
Windows XP or 2003 are not supported. You can download a compatible version of XAMPP for these platforms here.

2. Download the latest version of Xampp and MYSQL DB

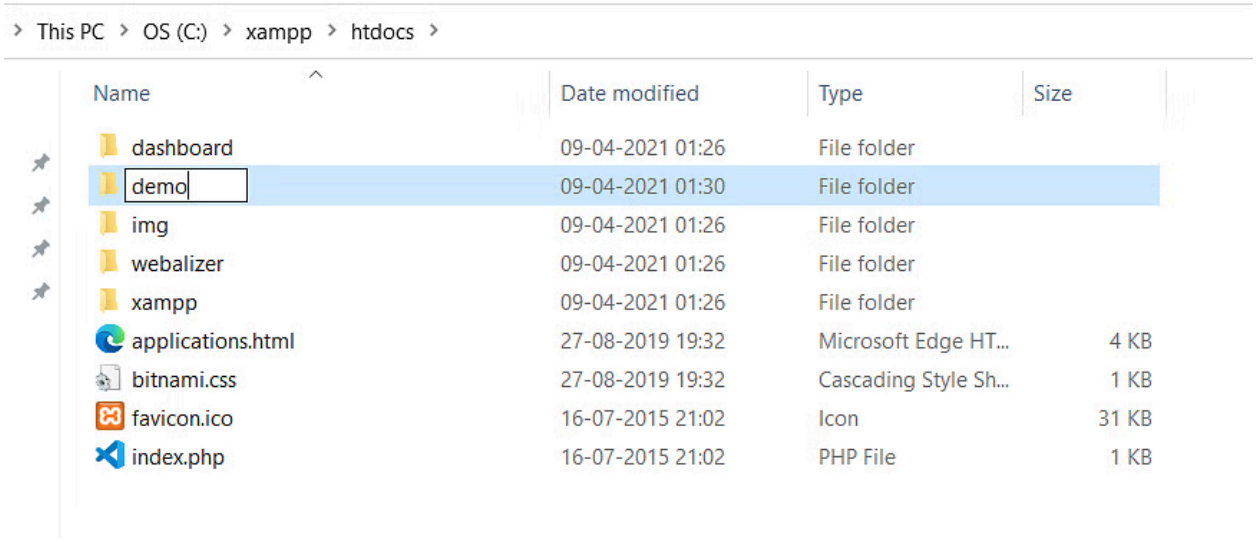
3.Once downloaded setup and begin the installation process and, in the “Select Components” section, select all the required components



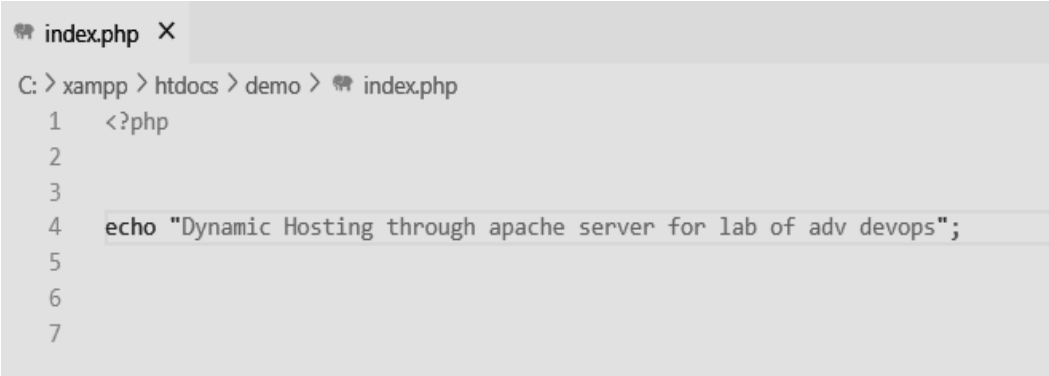
4. Keep the default directory “C:\xampp” and click on “next” to complete the installation.



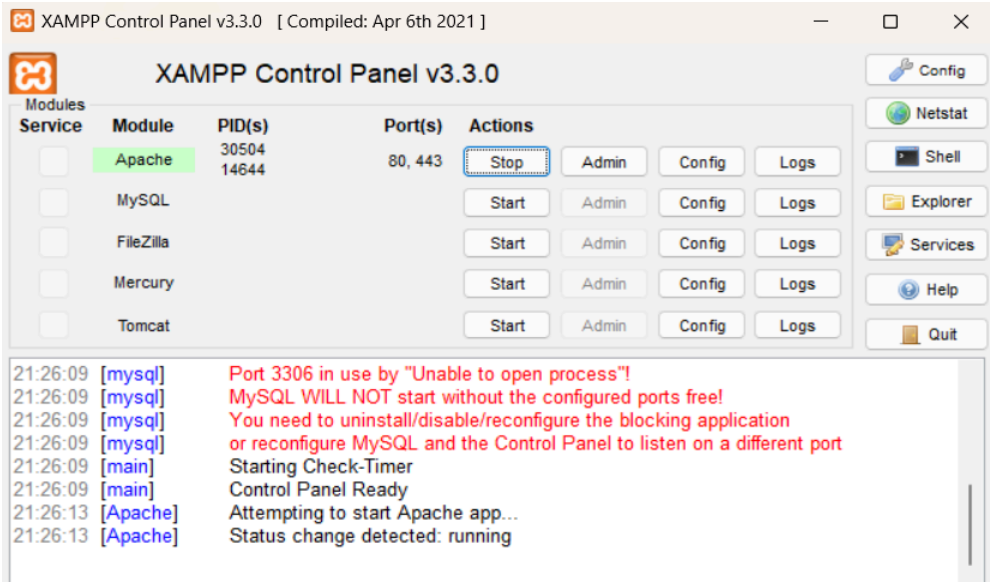
5. Go to “C:\xampp\htdocs” and inside it, create a folder called demo (or any name you want)



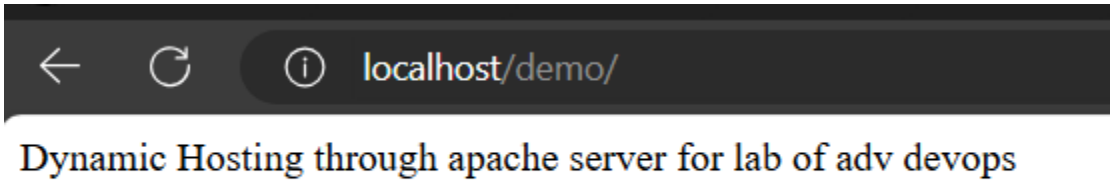
6.Open the folder and create a file name “index.php” and write the required script



7.To see the script output, open the XAMPP control panel and start Apache to host the local web server, where our script will be running.

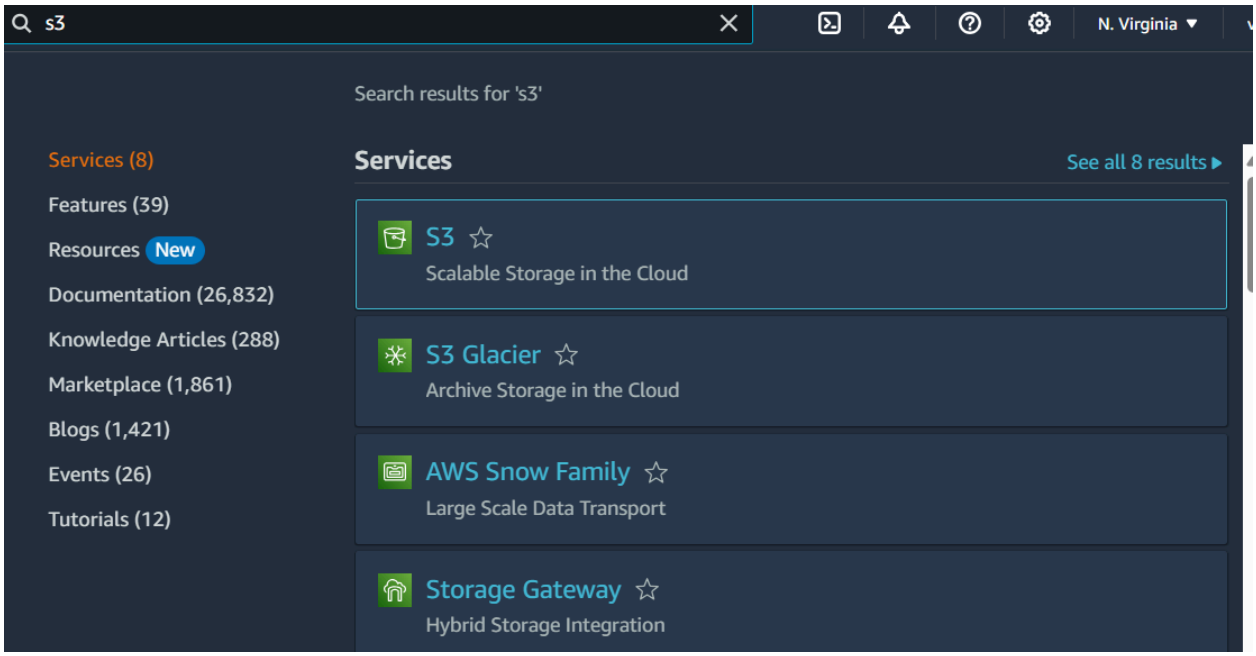


8. Navigate to your browser and type in “localhost/demo/” in the address bar to view the output.

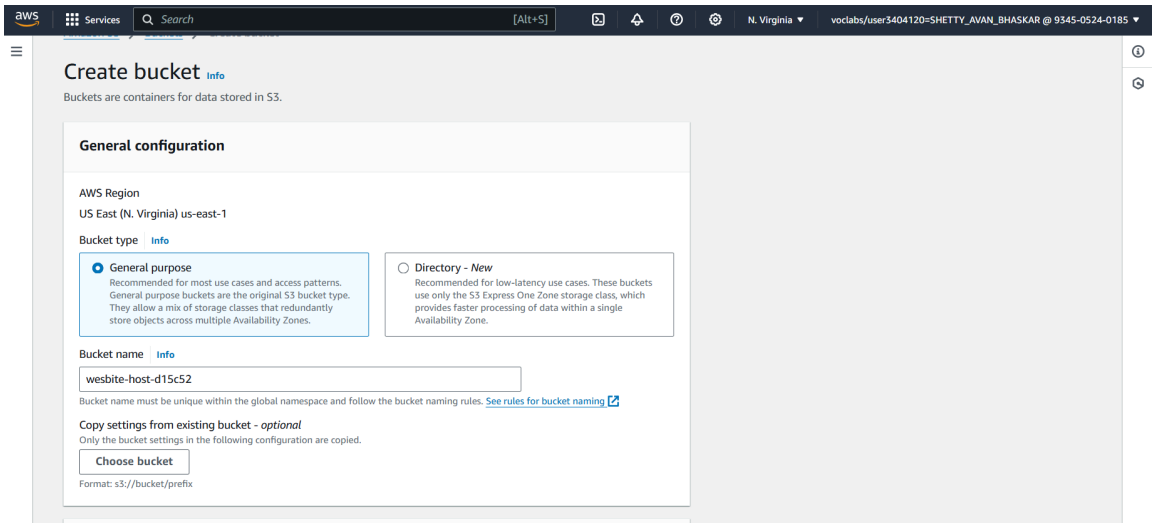


AWS S3 BUCKET:

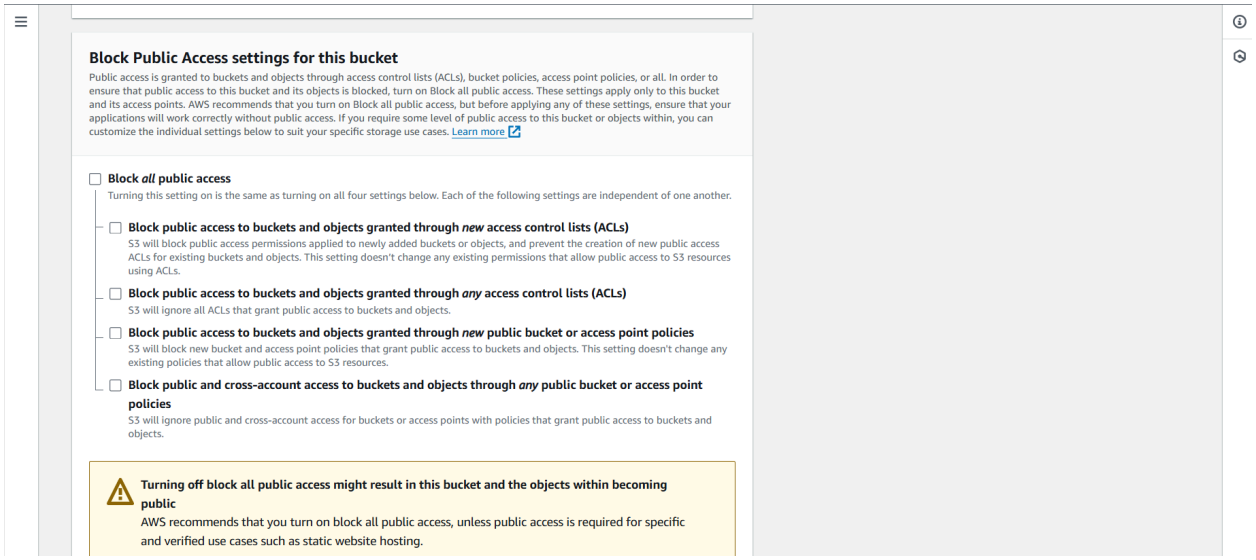
1. Open the aws Console and search for S3 bucket



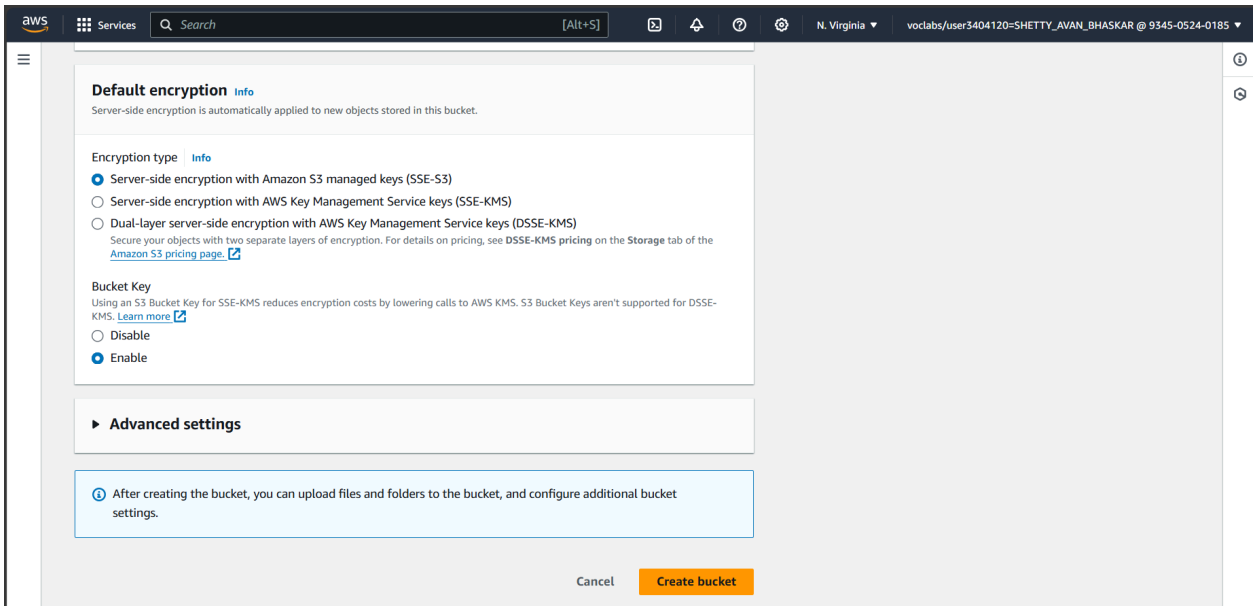
2. Now create a bucket of <-*your name*-> mine as 'website-host-d15c-52'



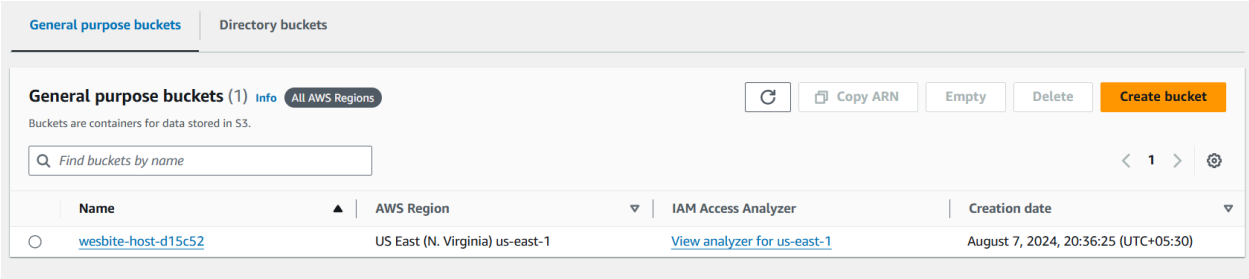
3. Scroll down and uncheck the Block of public access (*as we are making our static website available for all users in public domain*)



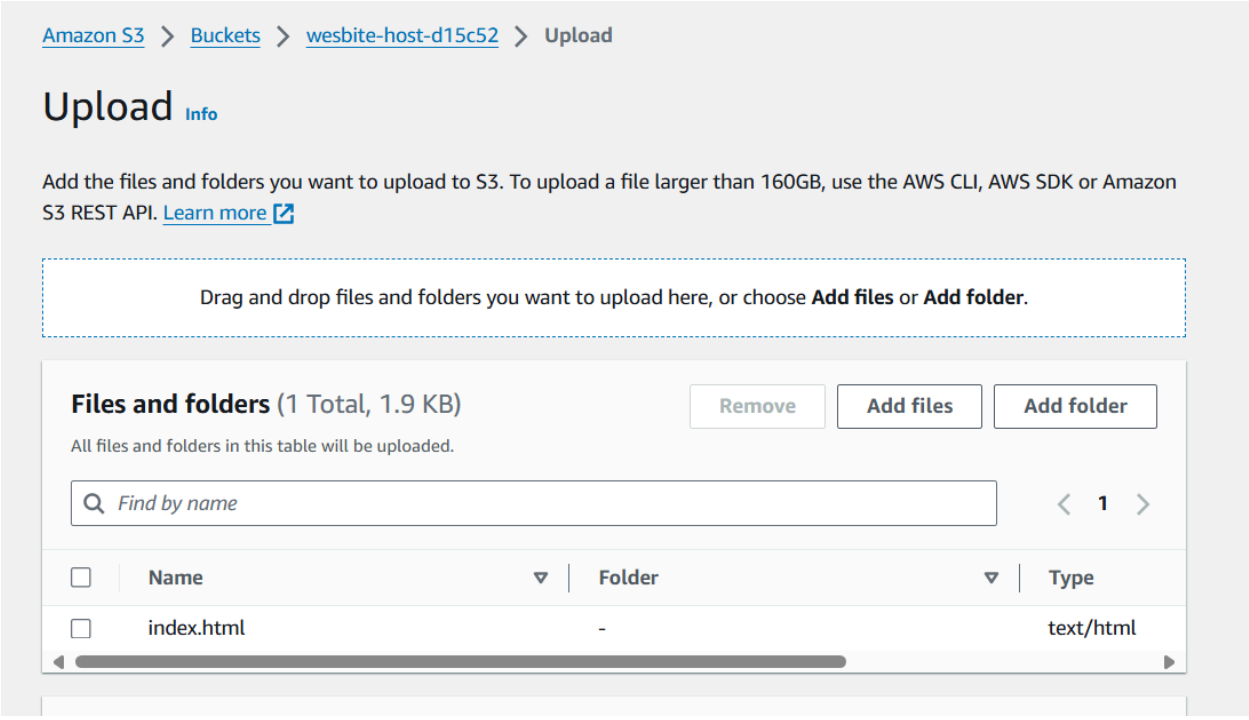
4. Click on Create bucket option

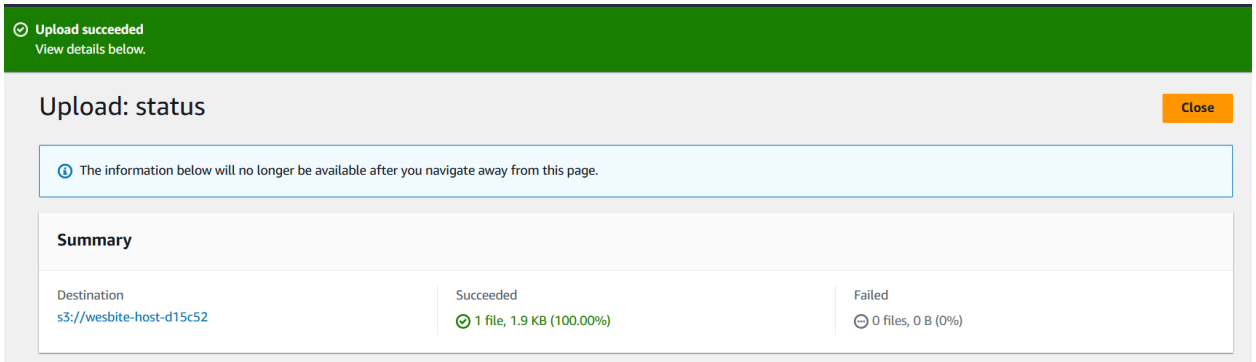


5. You go back to your buckets page and see the bucket has been created

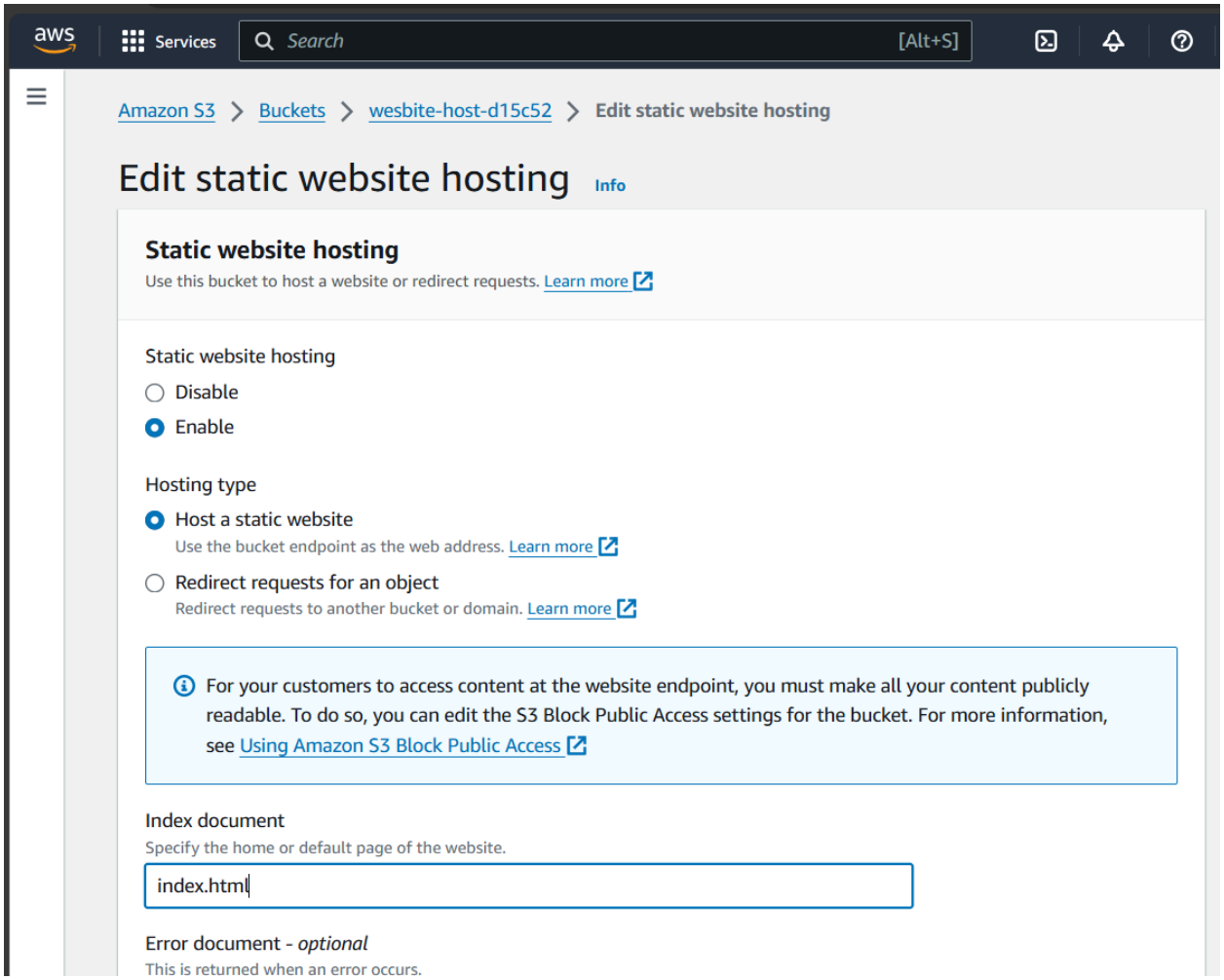


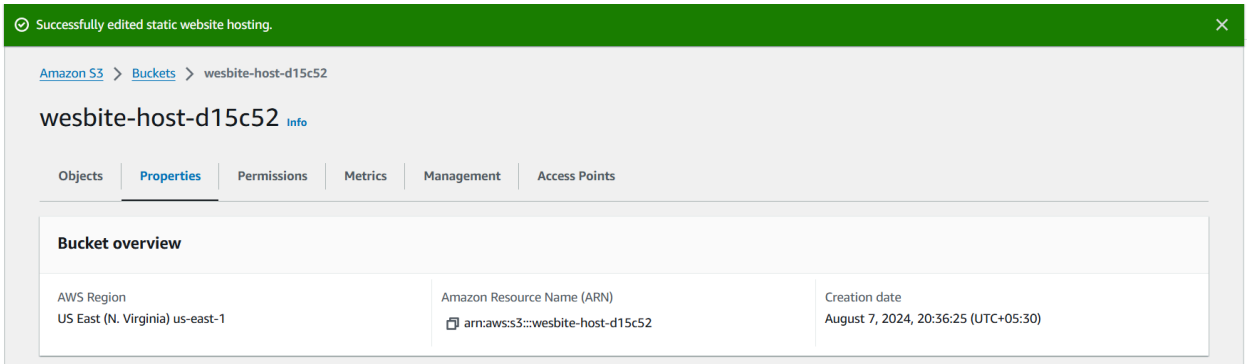
6. Now upload your static website folder or files, including any images if available. Once uploaded, you will see the message 'Upload successful'



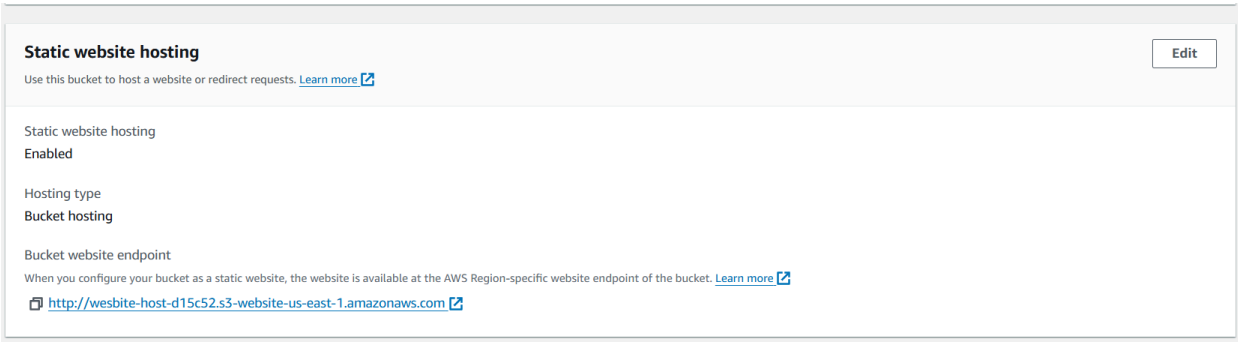


7. Go to Properties and when you scroll at the bottom click edit static website hosting
Enable -> host a static website -> input the name of your file uploaded and click save





8. With this your link for the static website through S3 bucket will be generated

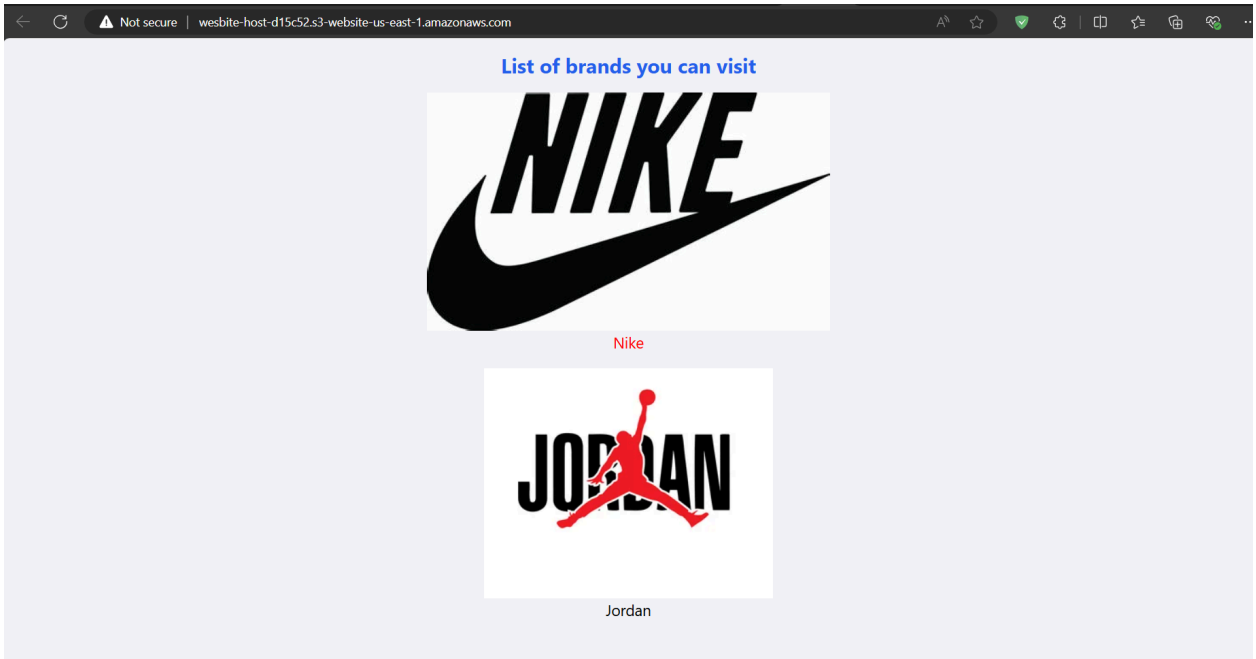


9. Now go to the Permissions and edit the bucket policy by adding this

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::wesbite-host-d15c52/*"
    }
  ]
}
```

Just change the arn according to your bucket name and click save

10. Once edited the bucket policy click on the link in the generated back in step 8 and your website is hosted through aws S3 bucket



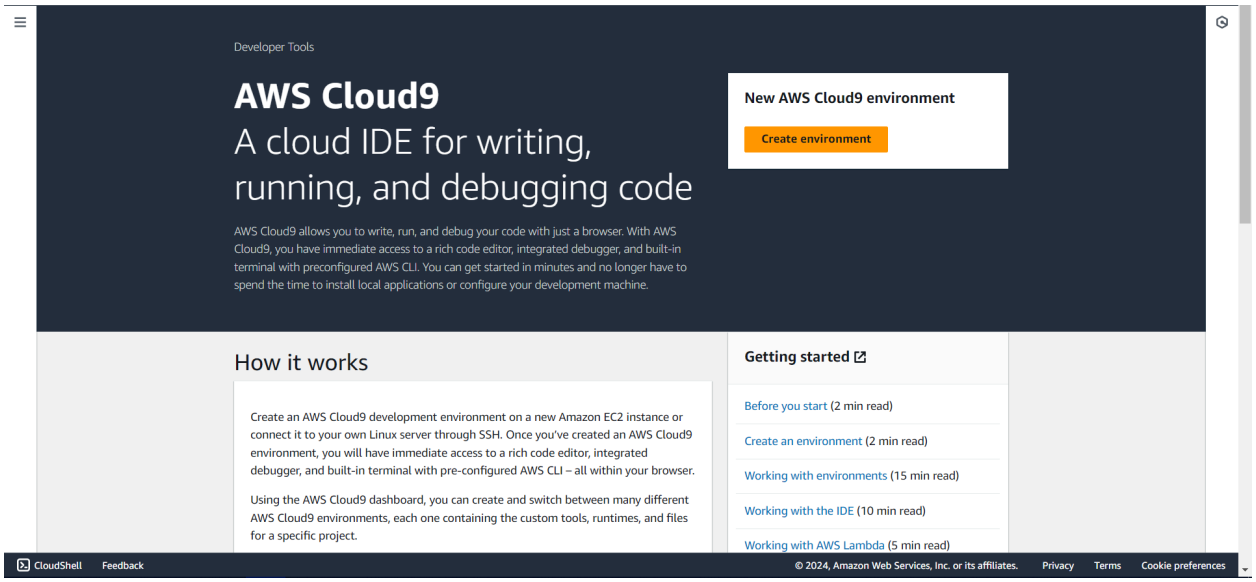
Conclusion :

Developing and hosting a website on a local VM provides full control over the environment but requires manual setup and maintenance. In contrast, hosting the website on an S3 bucket is simpler and more efficient, leveraging the cloud's auto-scaling, durability, and cost-effectiveness. With S3, there is no need for server management, and the website becomes globally accessible with built-in redundancy. This experiment demonstrated the flexibility of local development alongside the benefits of using cloud services like S3 for hosting, making cloud a more scalable and reliable solution for web hosting.

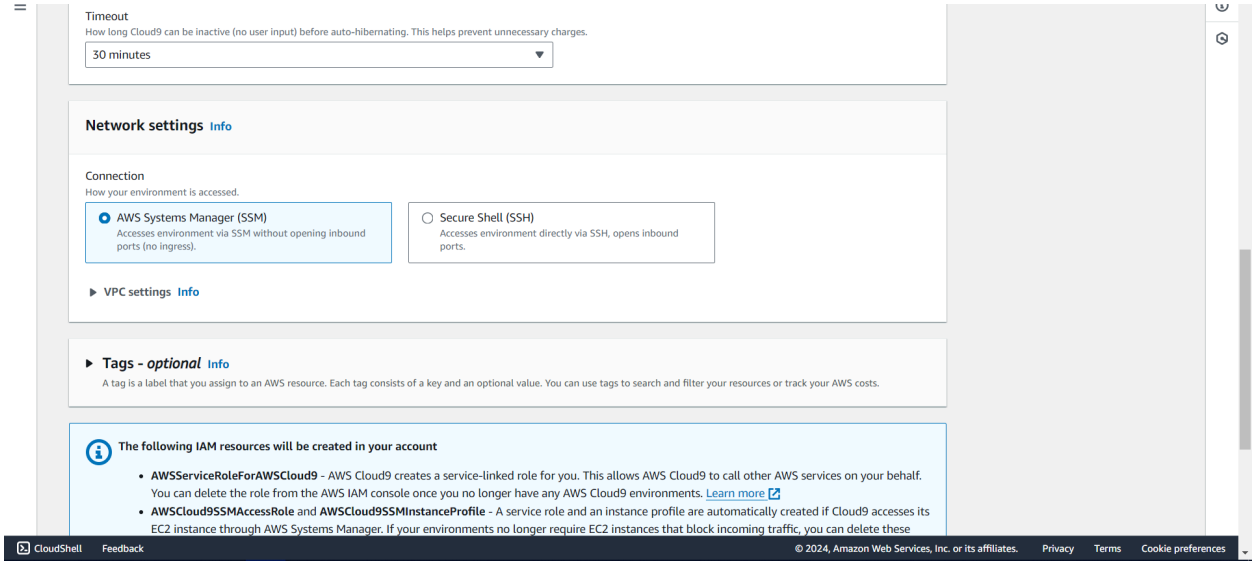
Cloud9 Infrastructure:

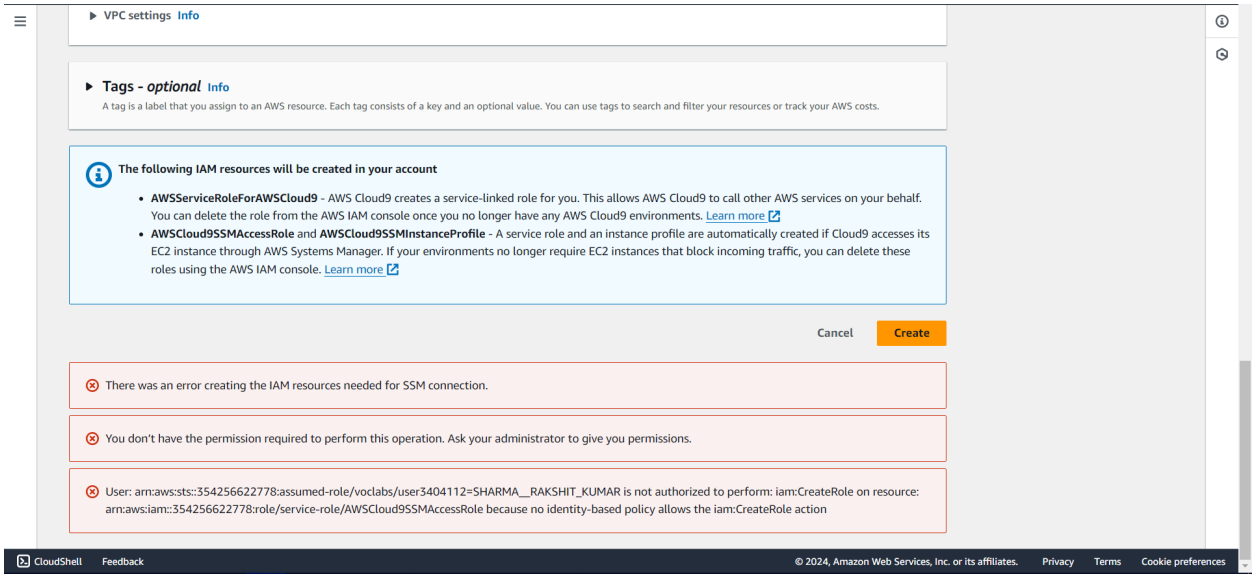
Aim : To understand the benefits of Cloud Infrastructure and Setup AWS Cloud9 IDE, Launch AWS Cloud9 IDE and Perform Collaboration Demonstration.

1. Open the AWS account and search for Cloud9. Click on create environment.

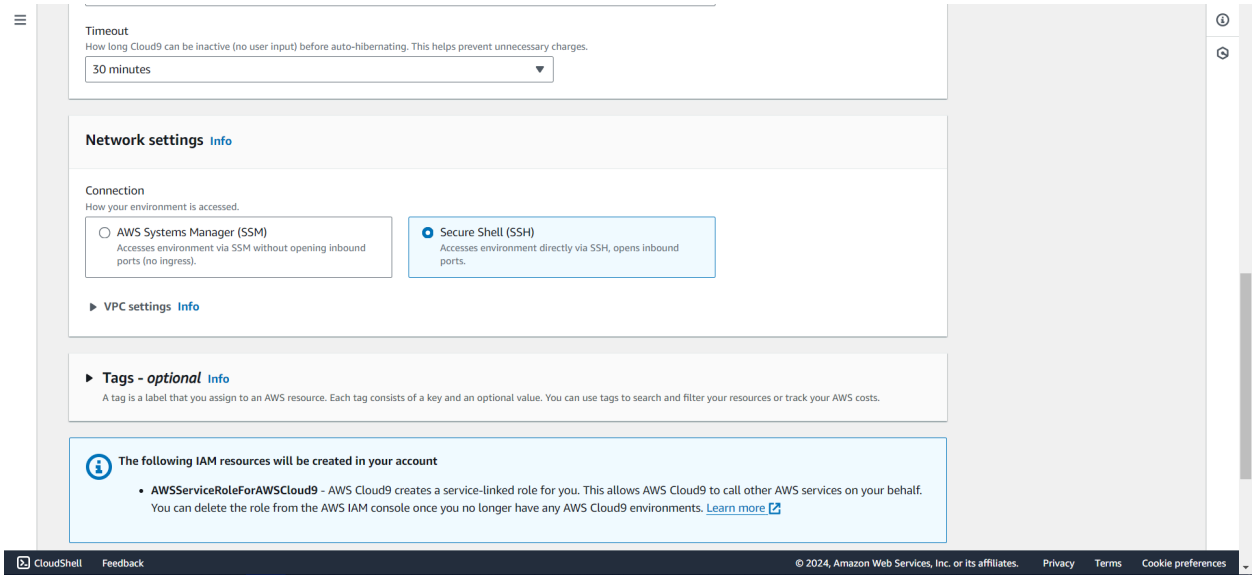


2. Enter the name and other required configuration for creating an environment. In network settings, using the AWS system manager gives an error while creating the environment

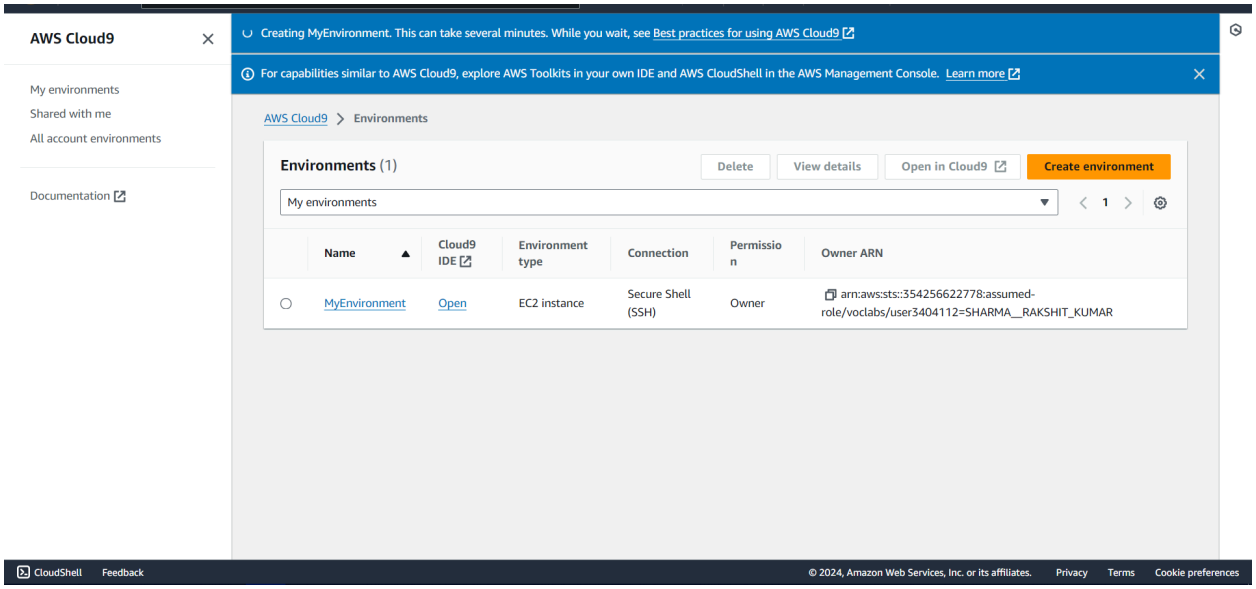




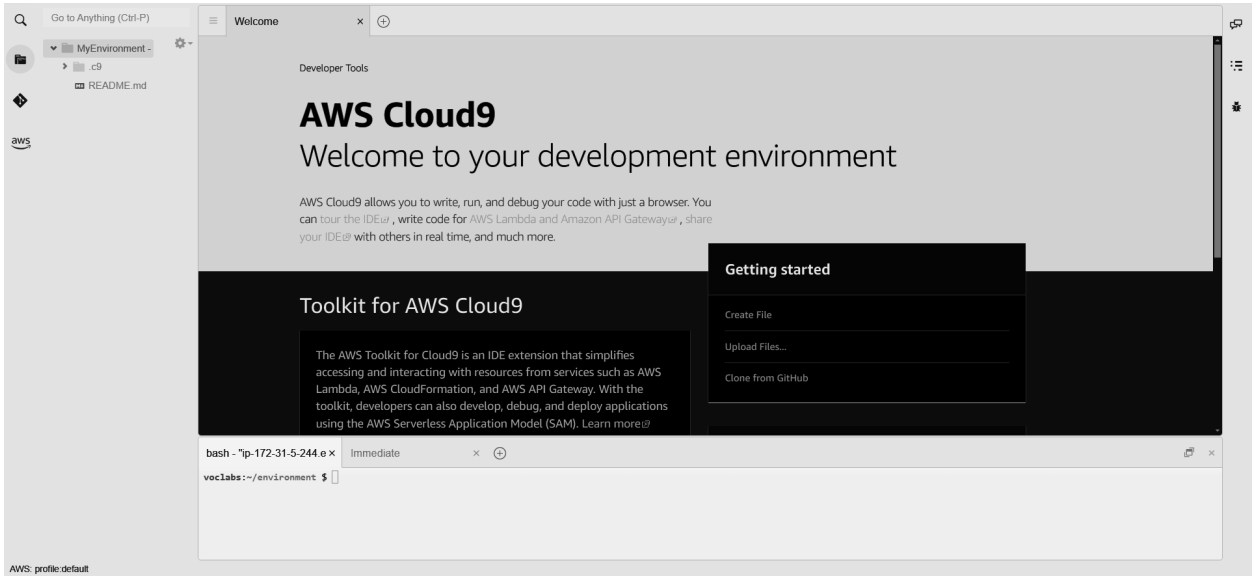
3. Use the Secure Shell option in Network settings.



4. Once the configuration is complete, click on create environment to create a Cloud9 environment.

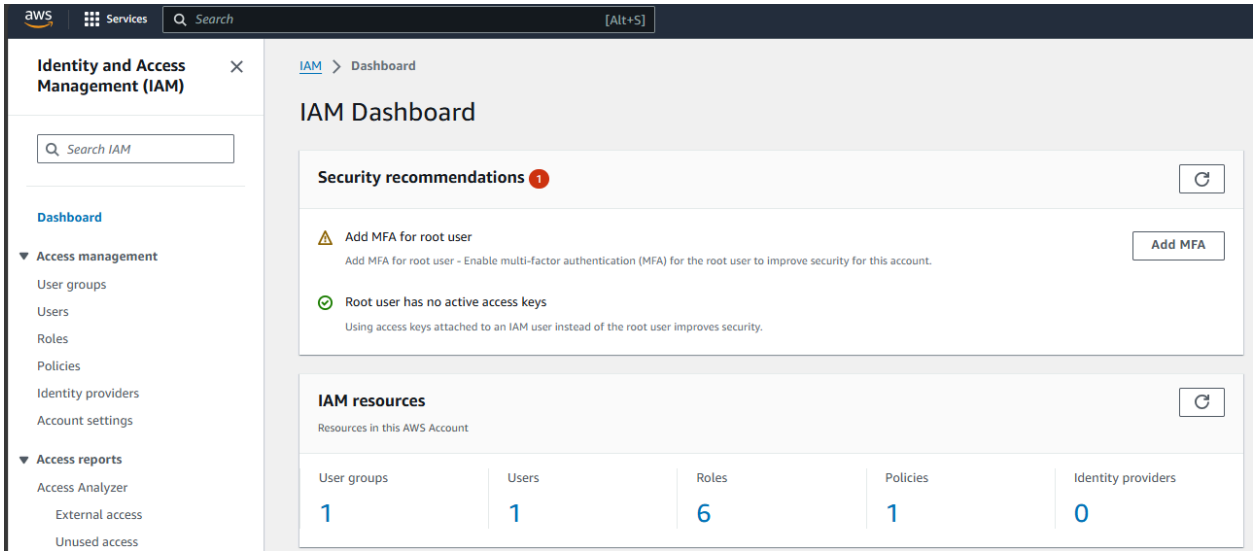


5. Cloud9 Environment.is opened when u click on the environment name

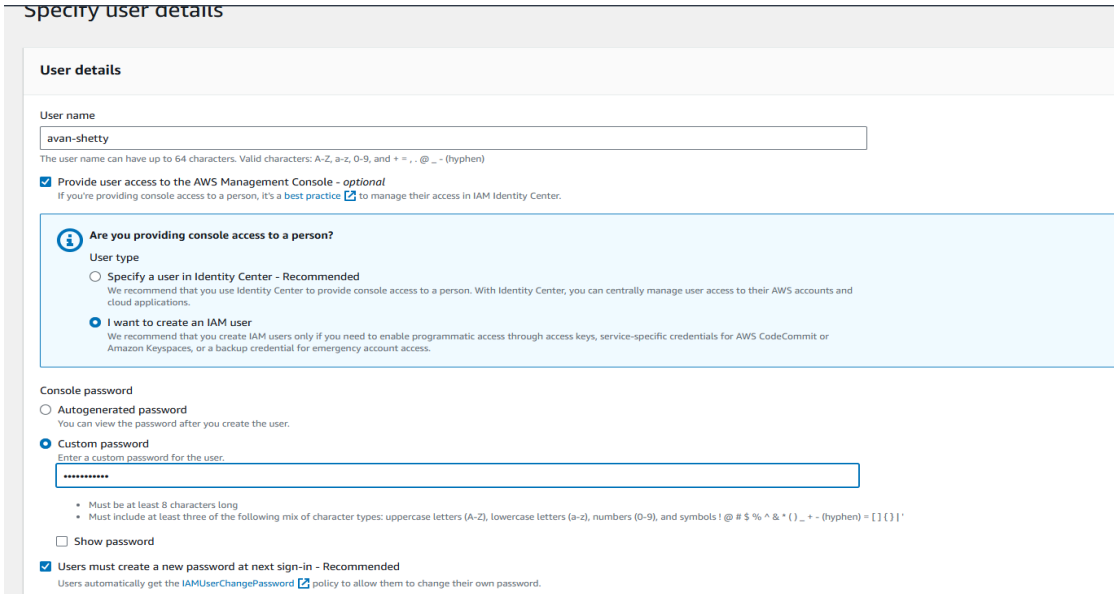


IAM user creation steps

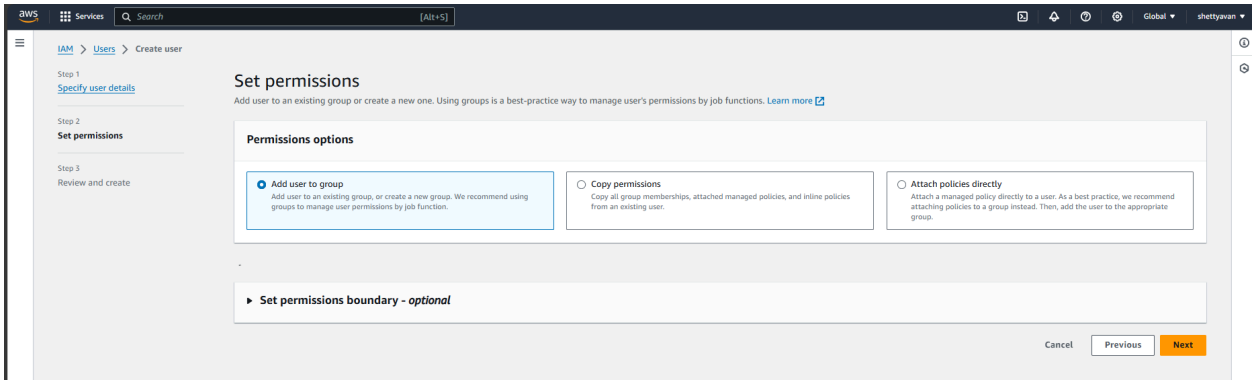
1. Open the aws account and search for IAM service.



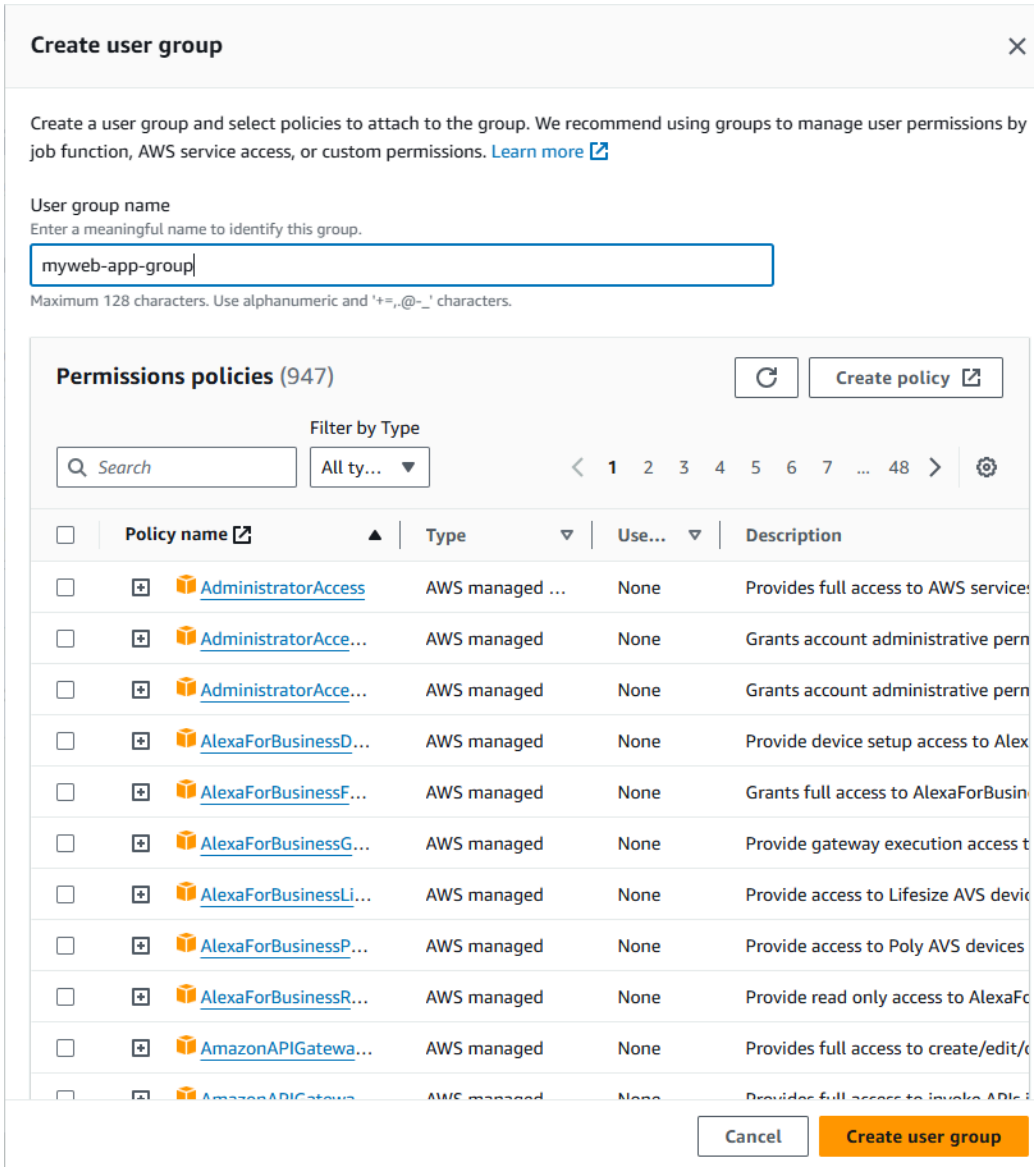
2. Select the users option from the left panel and click on create user button.Give the user name, select the "provide user access" checkbox. Also select the option "I want to create an IAM user". Otherwise we will have to enable the Identity center and specify a user there.
Add your custom password



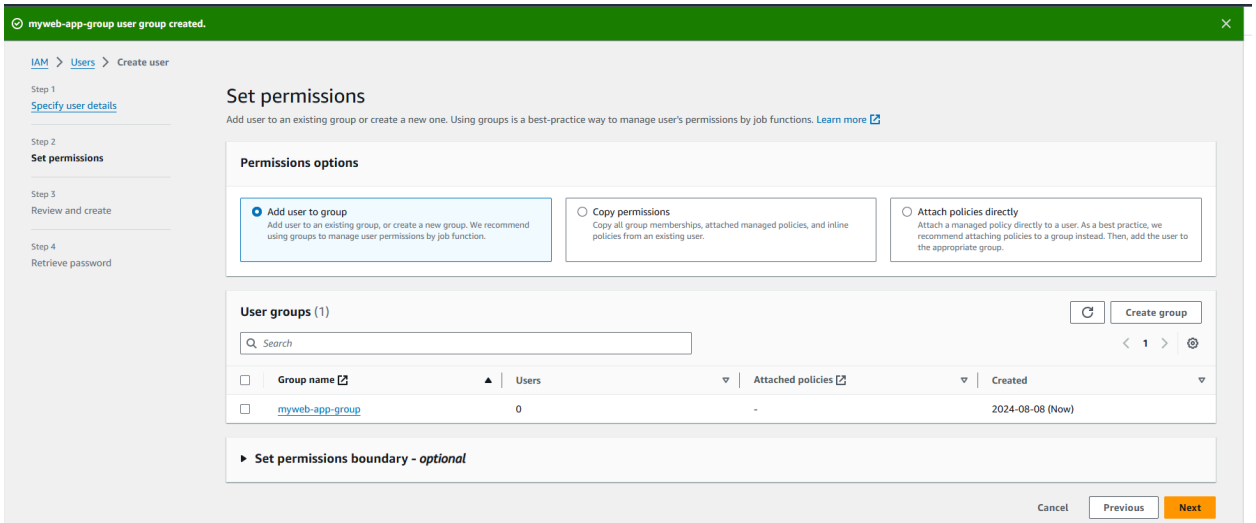
3. Click the add user option if you don't have an existing user group



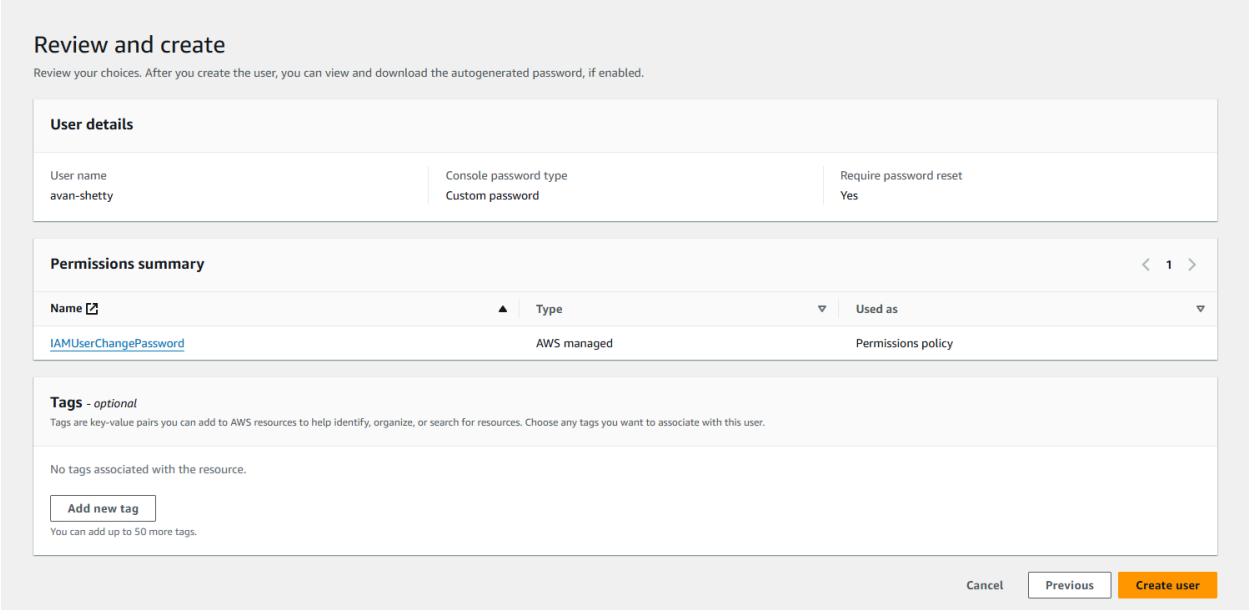
4. Give a name to your user group and check the policies if required any



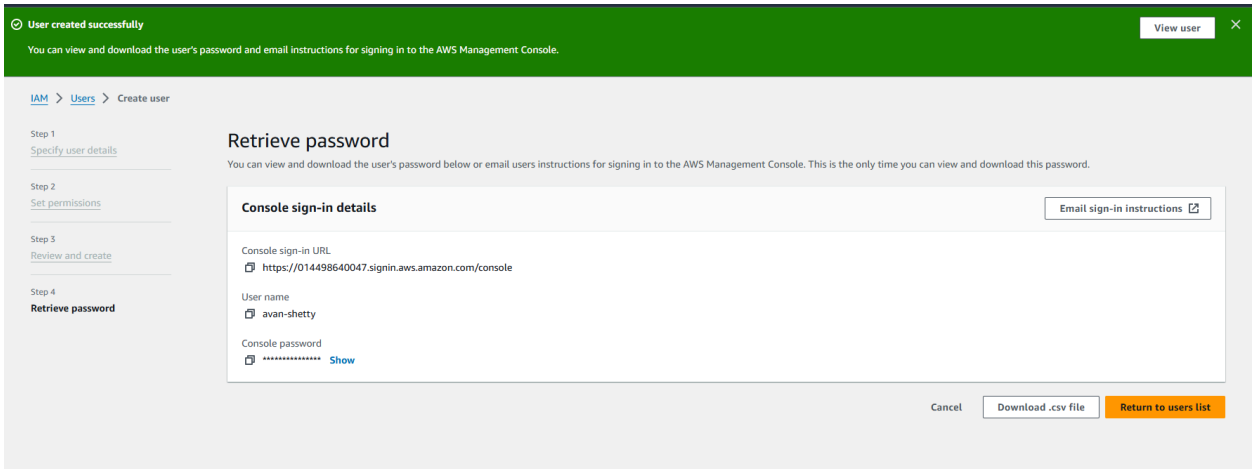
5. Once the user group is created select the name and click next to create your user



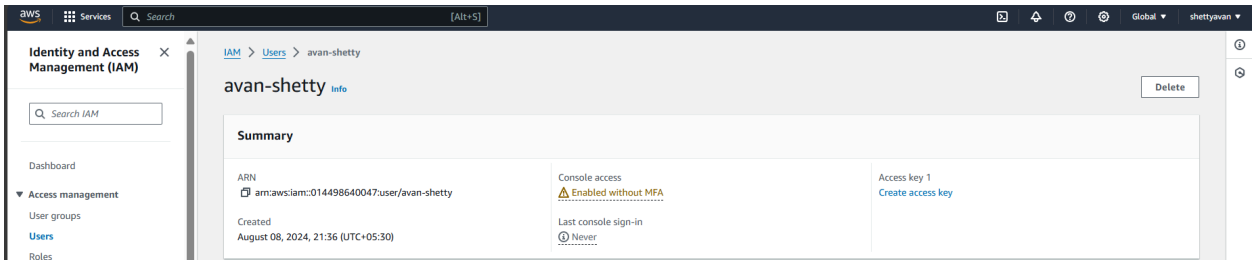
6. Review the configuration details and check if you have missed any steps and then click on 'Create user' button



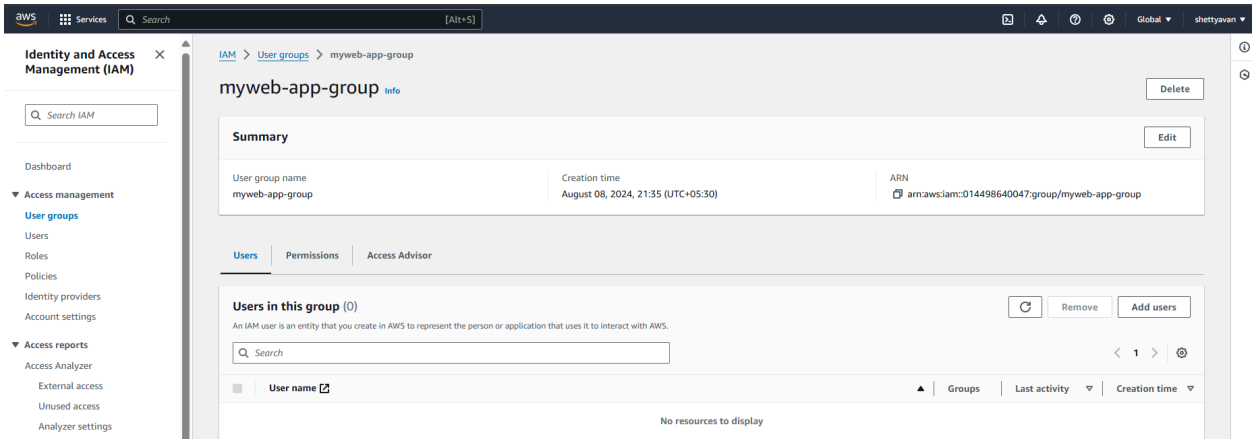
7. You will see the “user created successfully” message and incase you need then store your password by downloading the csv file



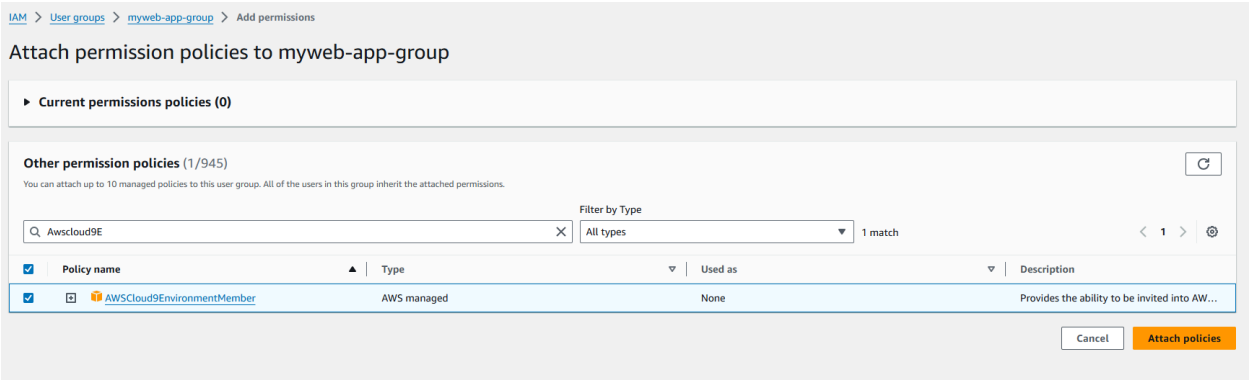
8. Click the users tab and you will see the user is created under IAM service. Also this service does not charge any price but this feature is not available in academy lab



9. After creation of the user, go to the user groups tab and select the group in which the user has been added. Navigate to the permissions tab. Click on add permissions and attach policy.



10. Search for the “AWSCloud9EnvironmentMember” policy and attach it.

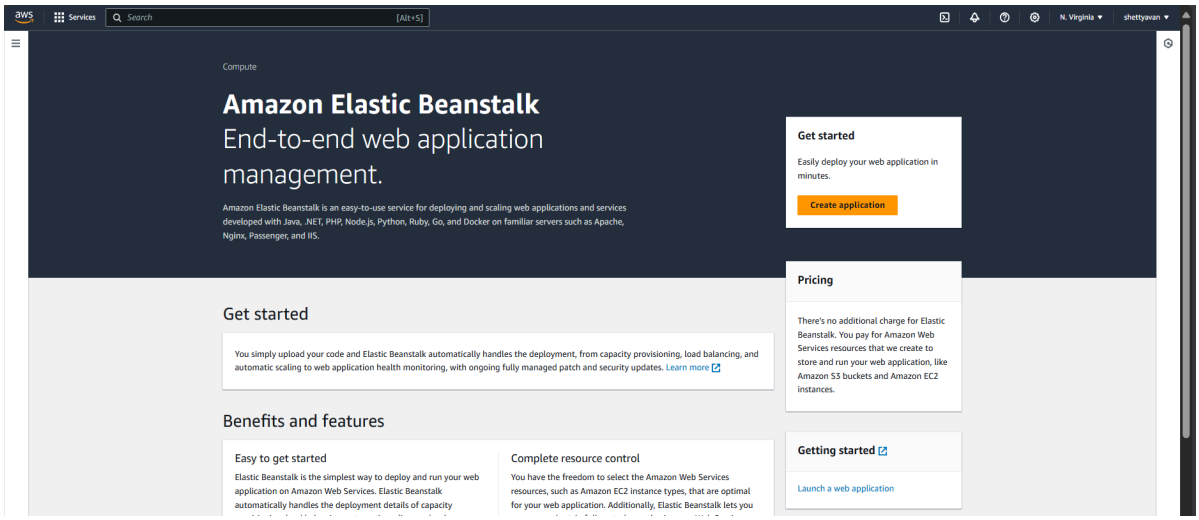


Conclusion:

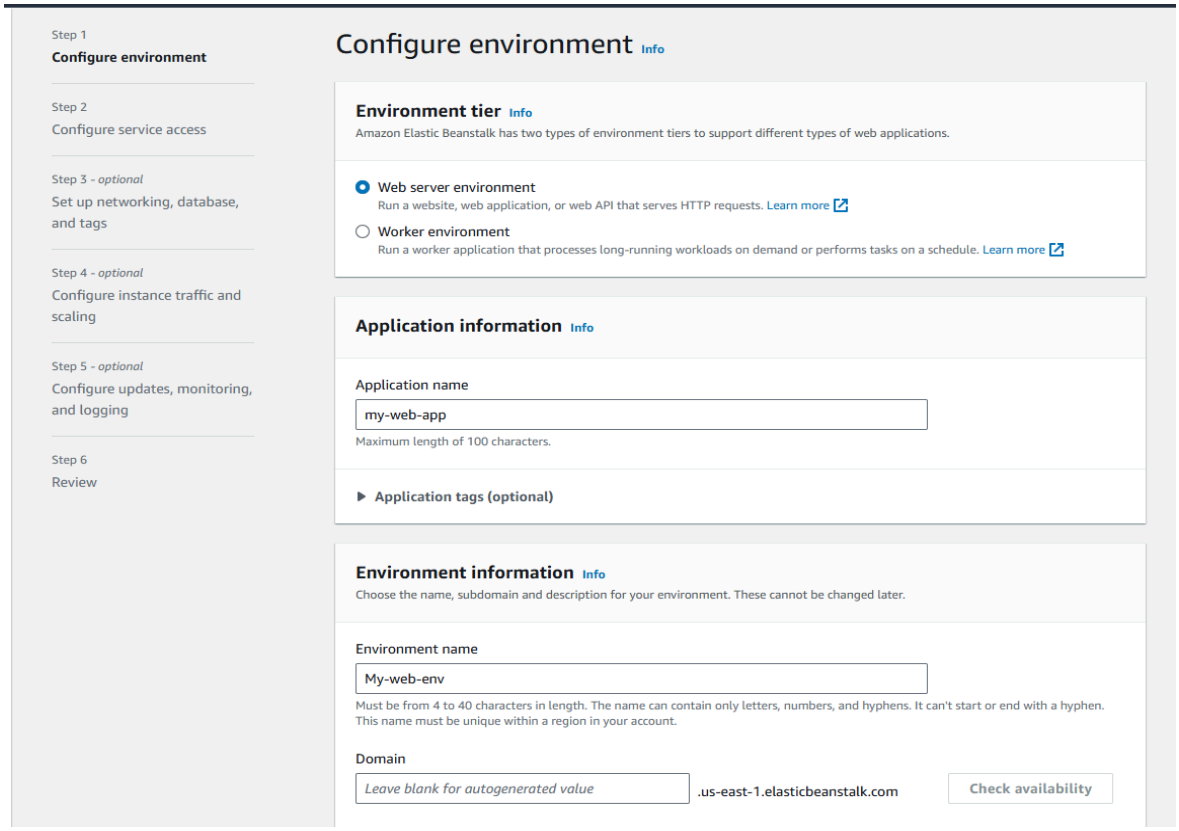
I explored the advantages of cloud infrastructure, particularly focusing on AWS Cloud9. Also set up the Cloud9 IDE and performed a collaboration demonstration to highlight the real-time coding and teamwork benefits.By performing a collaboration demonstration, I saw how multiple developers can work on the same project in real-time, improving teamwork, speed, and productivity. This IDE simplifies cloud-based development, making collaboration seamless and efficient without needing local setups

Experiment 2:
Aim : To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.

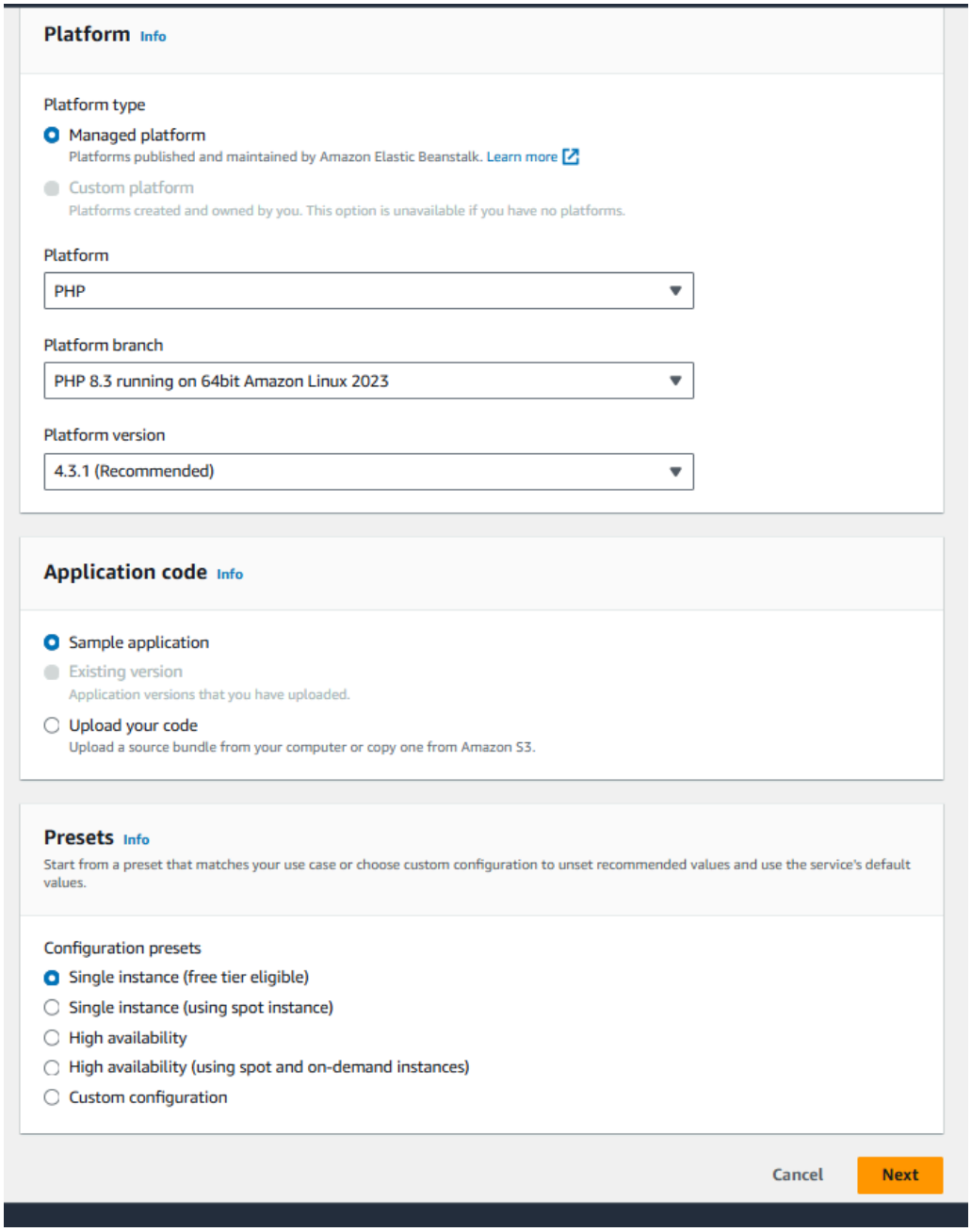
1. Open the aws console and then search Elastic Beanstalk (Opens a dashboard as seen below)



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

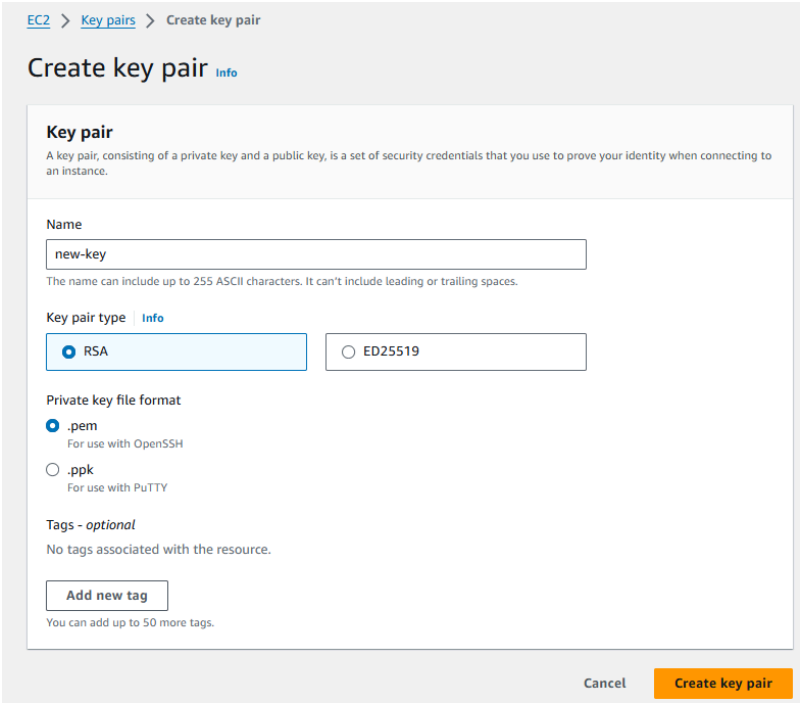


3. Choose PHP from the drop-down menu and click next

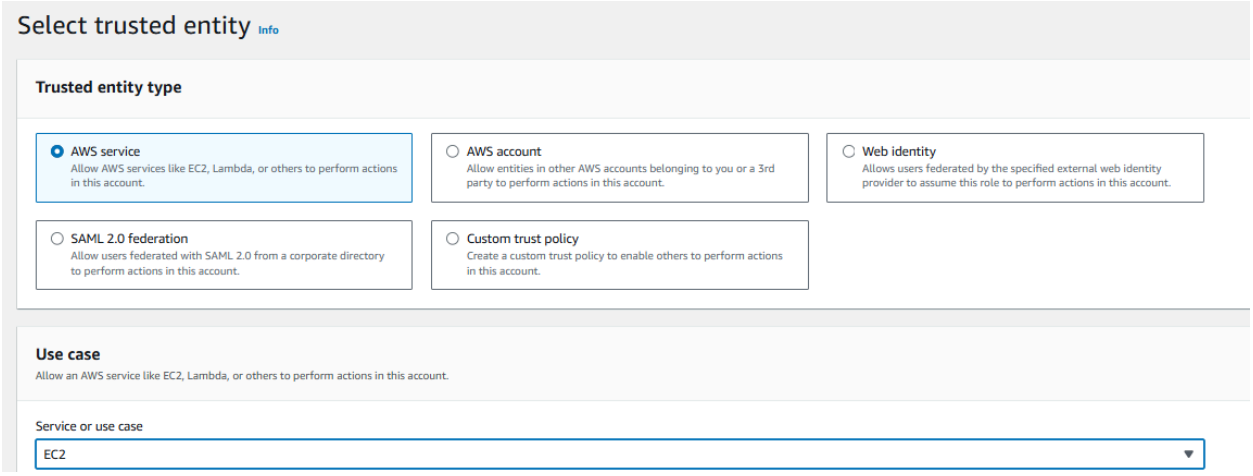


4. In this step you have to create a key pair

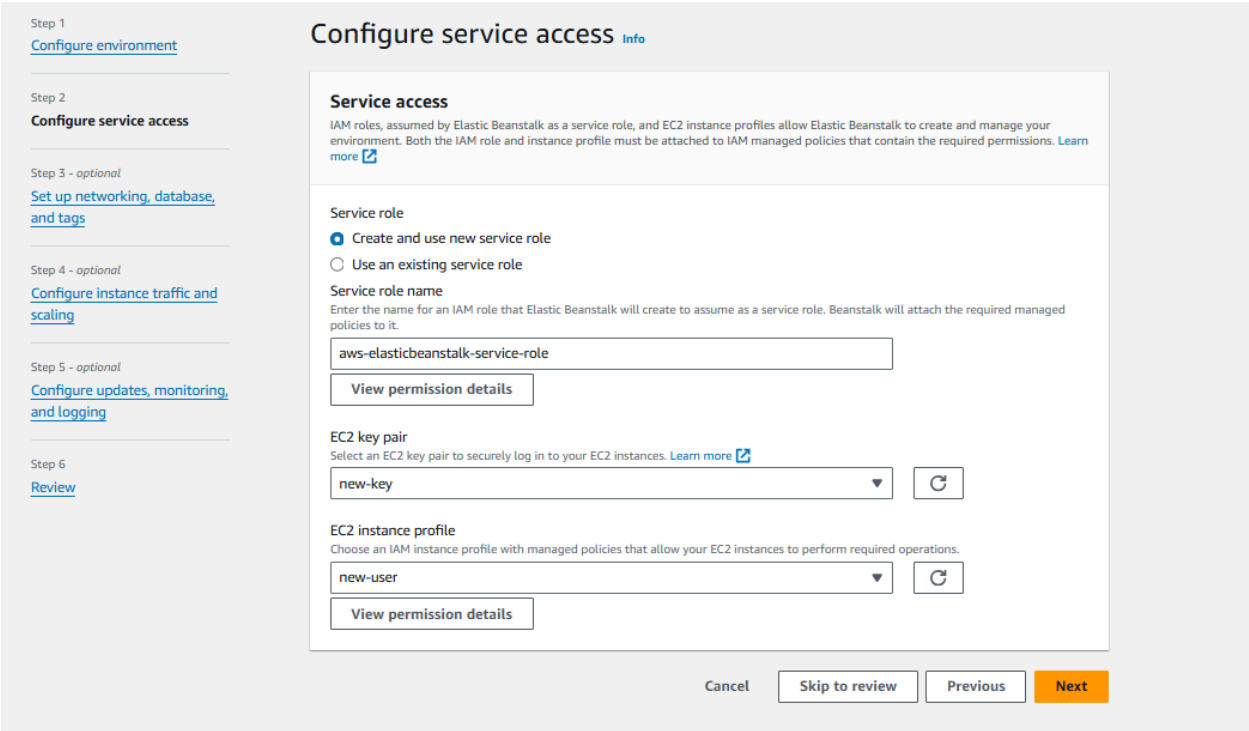
a. Go to EC2 instance tab and from the let panel create a key pair. As the key pair might be useful for the further process



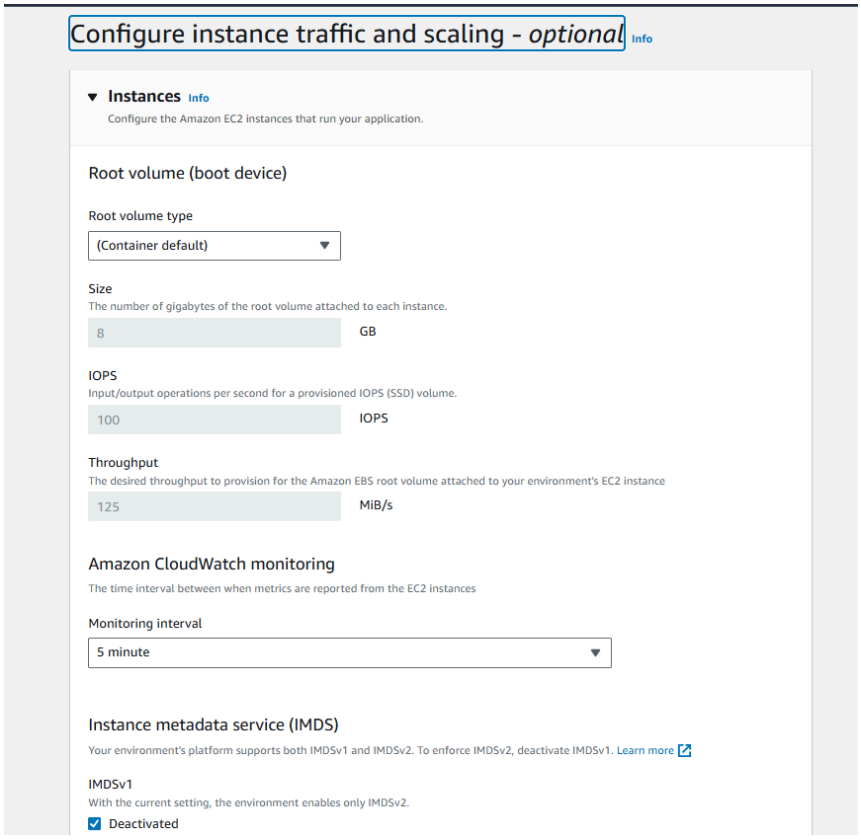
b. In the same fashion go to IAM and then under roles section click create role and then Select AWS Service and under instances select EC2. Your EC2 Instance role is created

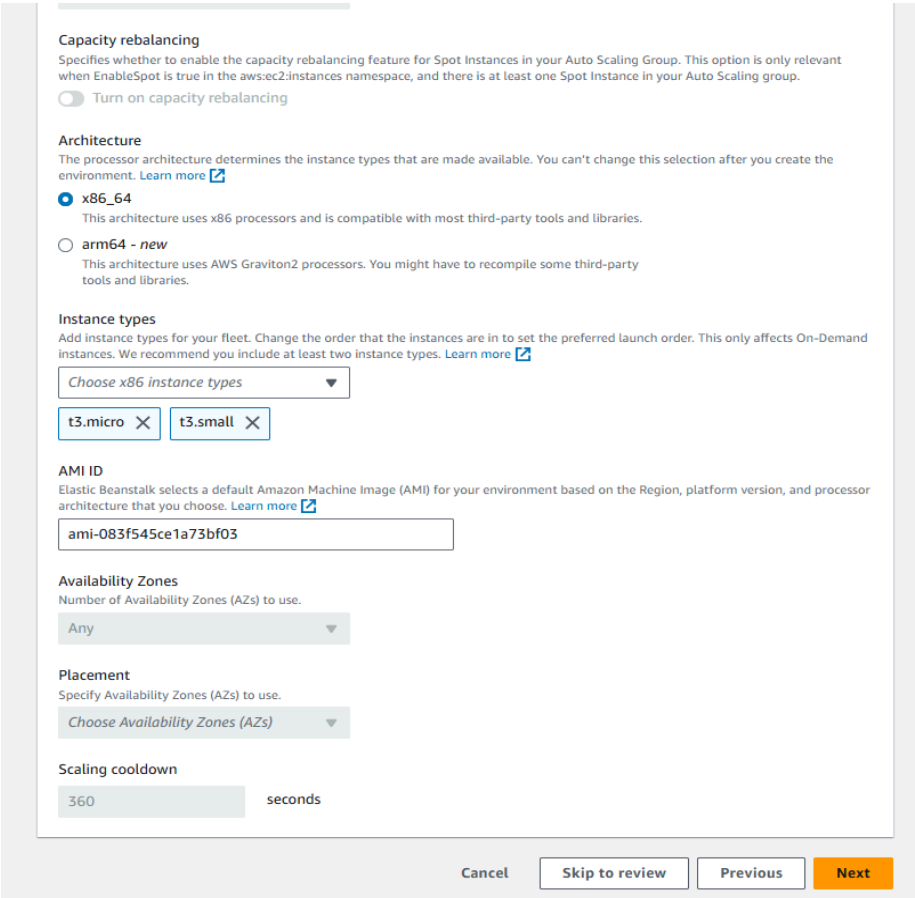


5. Now comeback to Elastic Beanstalk page and from the drop down menu select the newly created key pair and instance profile



6. No changes further in the Configure instance traffic section





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

Configure updates, monitoring, and logging - *optional* Info

Monitoring Info

Health reporting

Enhanced health reporting provides free real-time application and operating system monitoring of the instances and other resources in your environment. The **EnvironmentHealth** custom metric is provided free with enhanced health reporting. Additional charges apply for each custom metric. For more information, see Amazon CloudWatch Pricing

System

- ○ Basic
- ◉ Enhanced

CloudWatch Custom Metrics - Instance

Choose metrics

CloudWatch Custom Metrics - Environment

Choose metrics

Health event streaming to CloudWatch Logs

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

Log streaming

☐ Activated (standard CloudWatch charges apply.)

Retention

7

Lifecycle

Keep logs after terminating environment

Managed platform updates Info

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

Amazon X-Ray

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

X-Ray daemon

(service charges may apply.)

☐ Activated

S3 log storage

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

Rotate logs

(standard S3 charges apply.)

☐ Activated

Instance log streaming to CloudWatch logs

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

Log streaming

(standard CloudWatch charges apply.)

☐ Activated

Retention

7

Lifecycle

Keep logs after terminating envir...

Environment properties

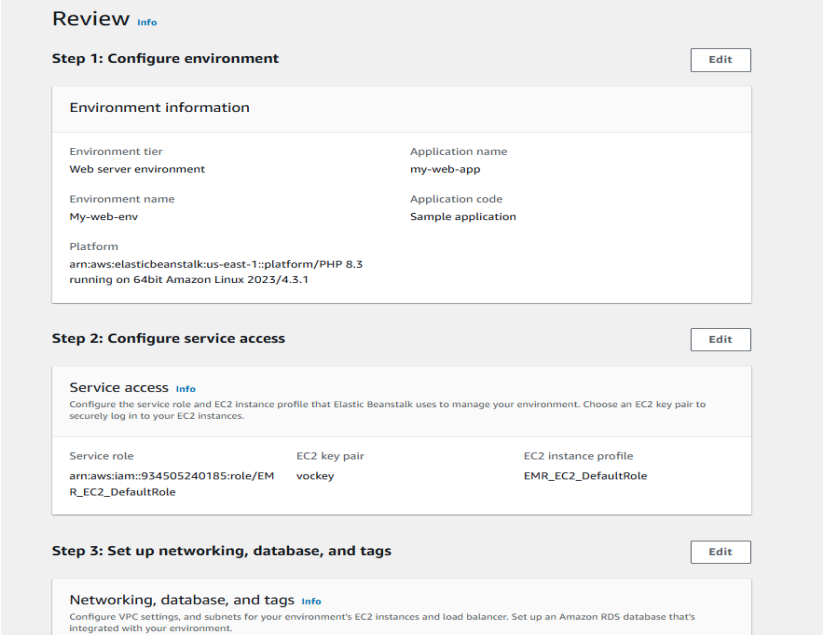
The following properties are passed in the application as environment properties. Learn more

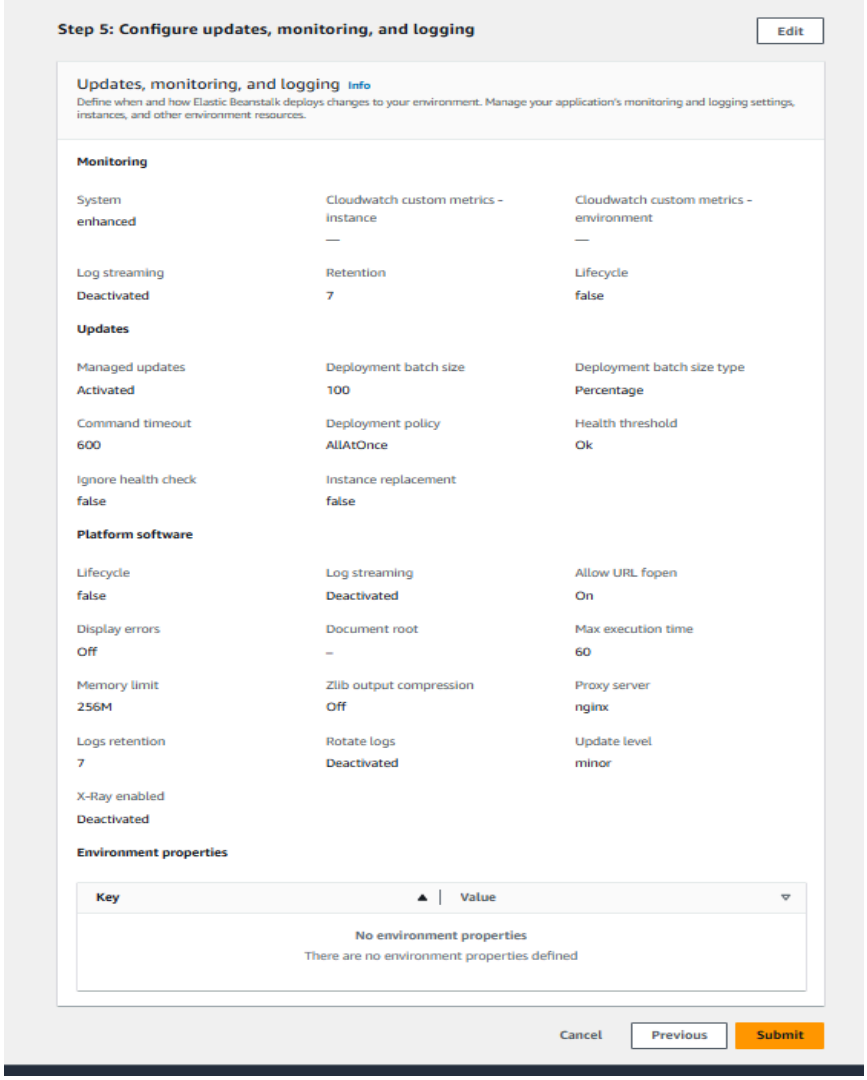
No environment properties have been configured.

Add environment property

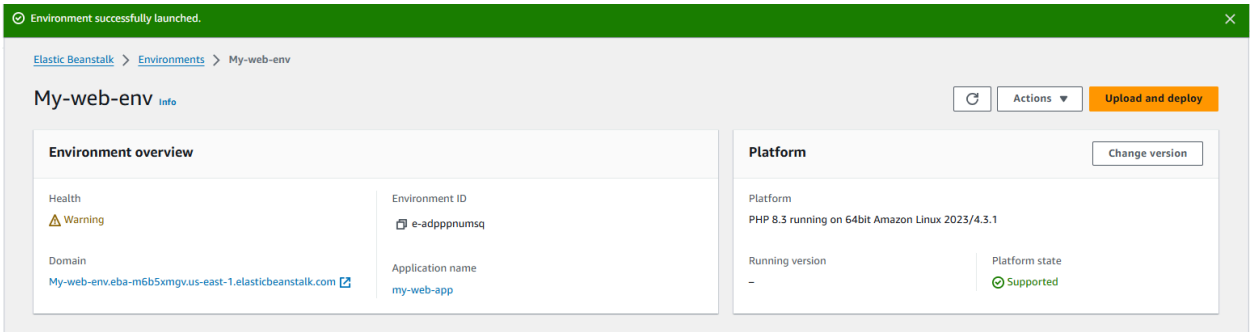
Cancel Previous Next

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





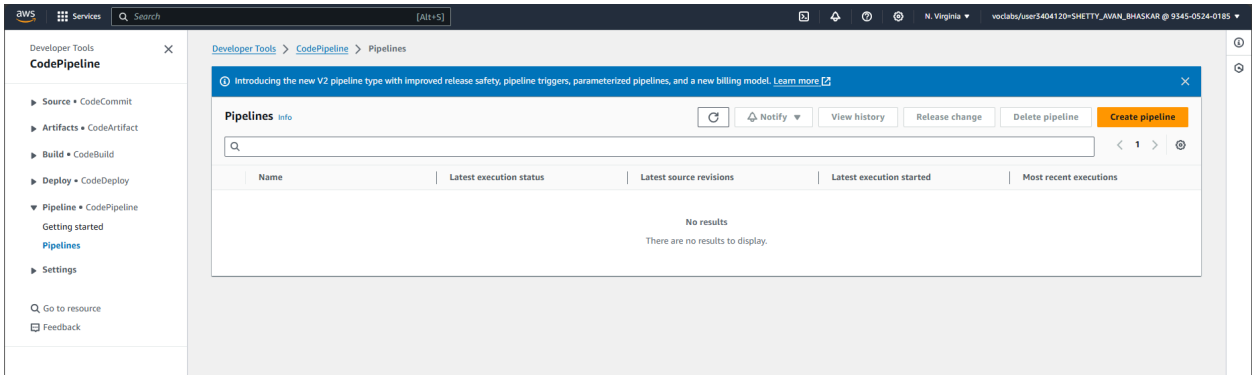
9. Your sample environment is created for you to deploy your application. By default, it creates an EC2 instance, a security group, an Auto Scaling group, an Amazon S3 Bucket, Amazon CloudWatch alarms and a domain name for your Application.



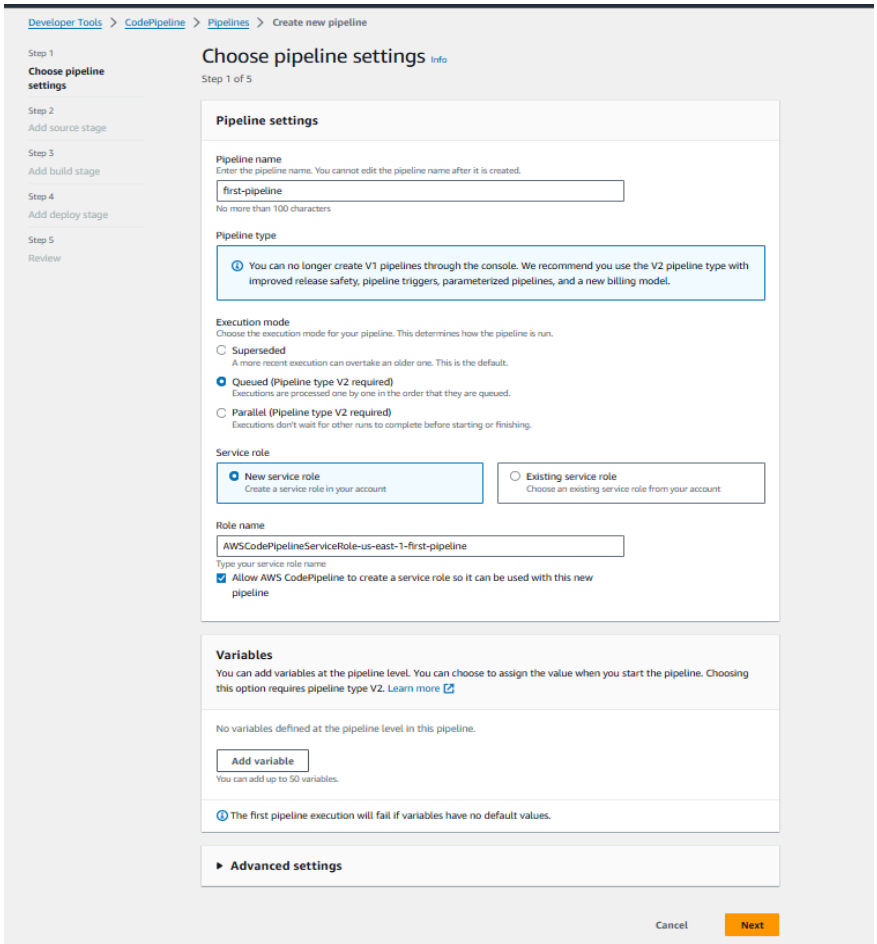
Pipeline creation:

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

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

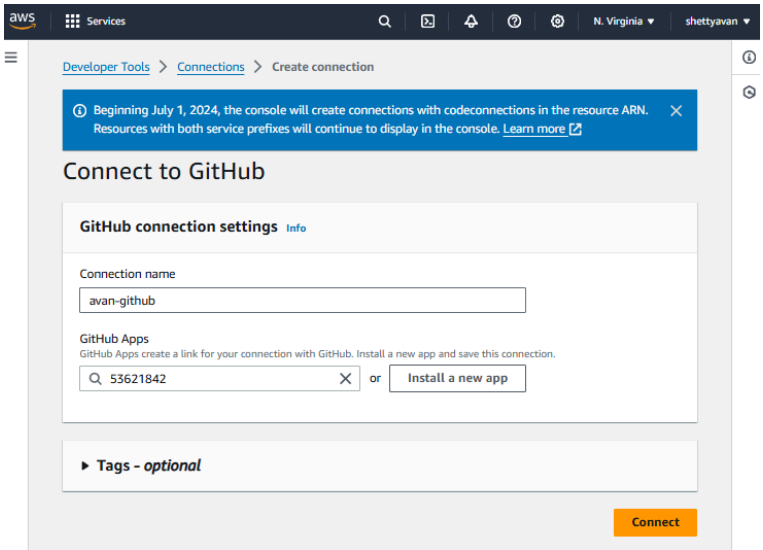


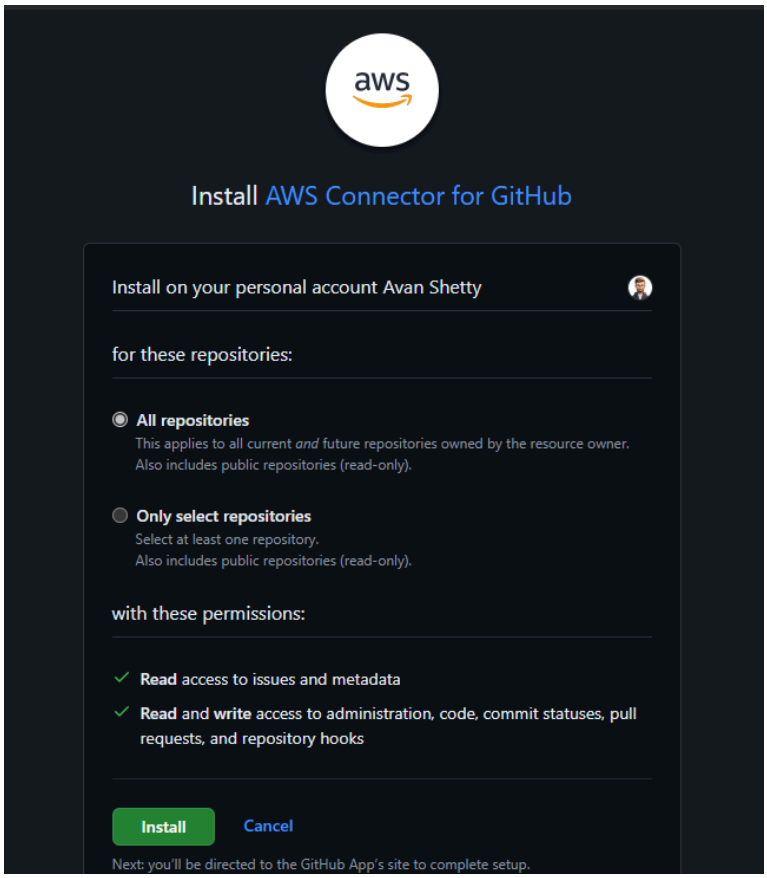
3. Name your pipeline and select the desired service role



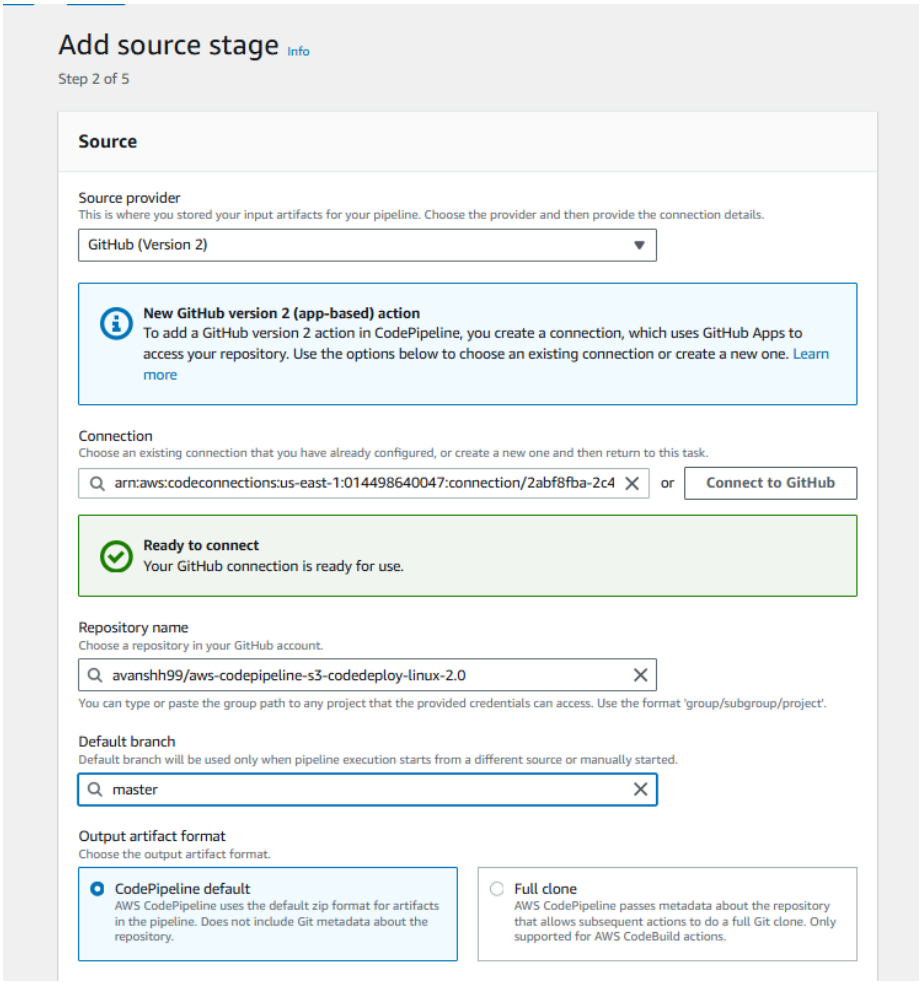
4. In the source stage select Github v2 as the provider and then connect your github connect so that the pipeline can access the forked source code .

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

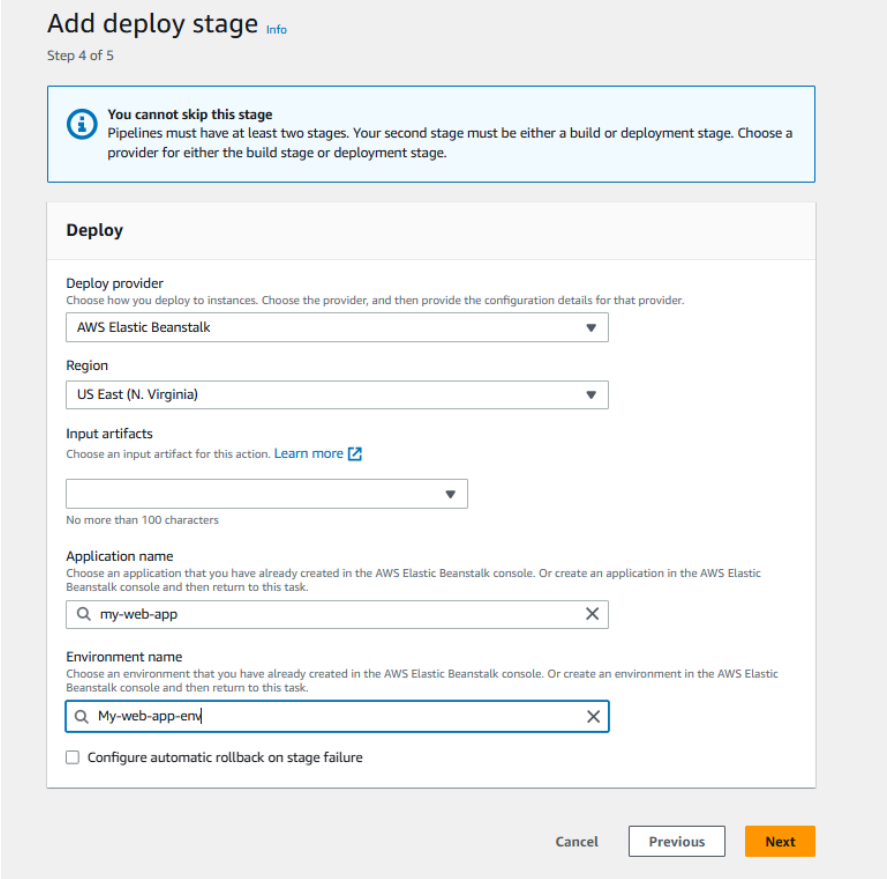




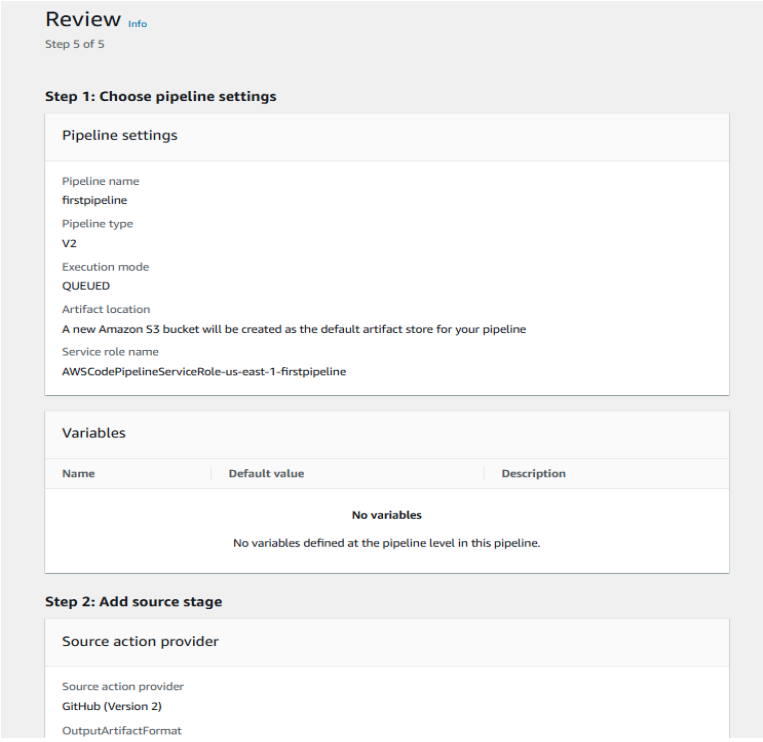
5. Once the connection is established from the drop down menu select the repository and the branch

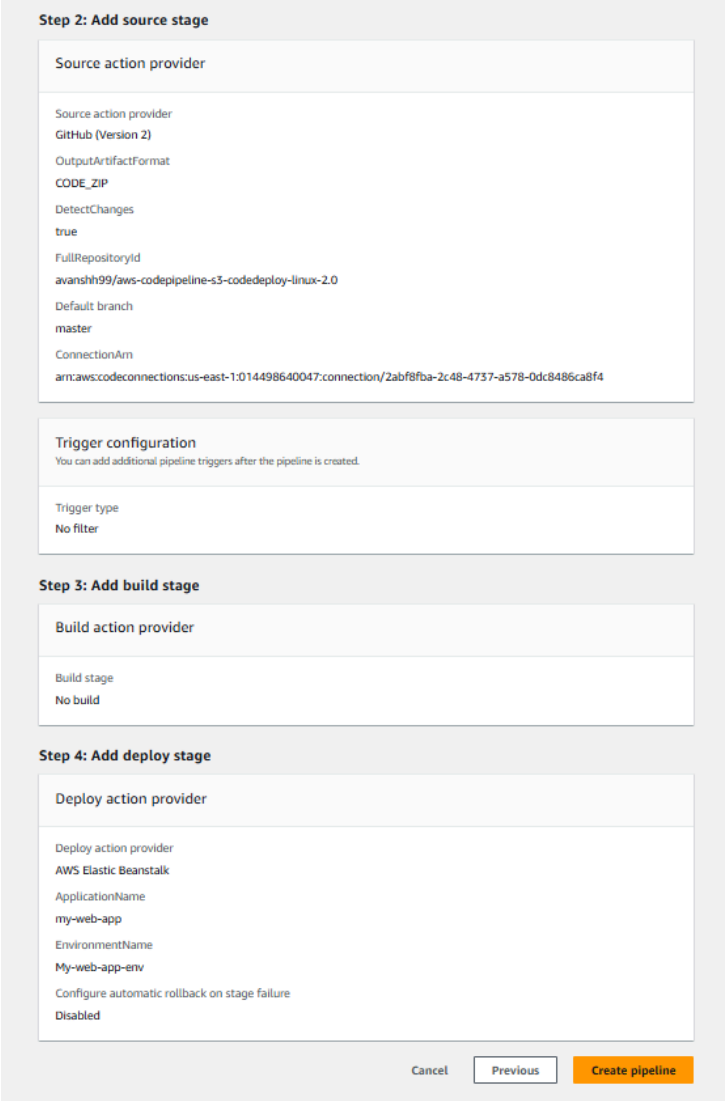


6. Skip the build stage part as we are not plugging in any build provider and in choose Beanstalk as the Deploy Provider, same region as the Bucket and Beanstalk, name and environment name.

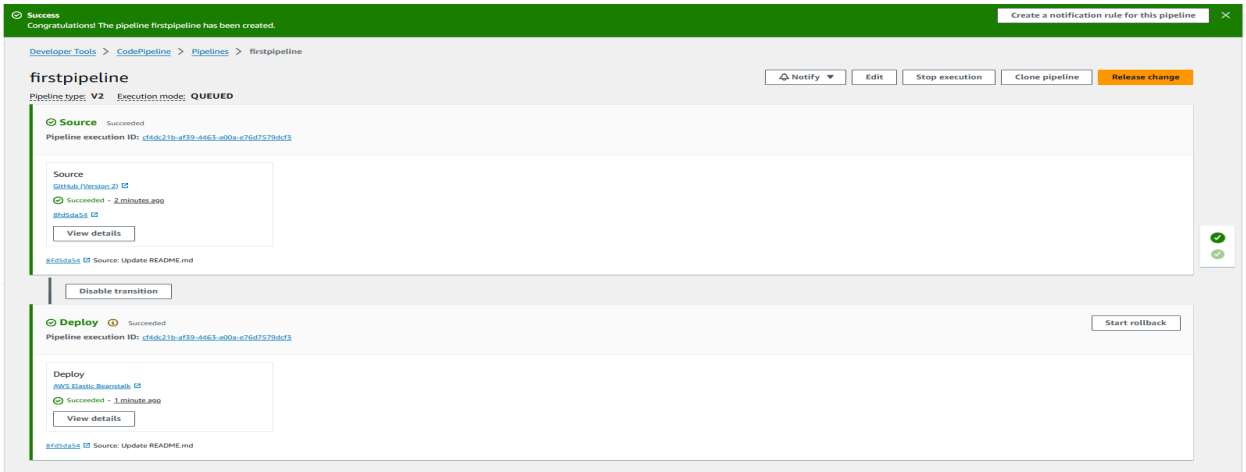


7. Review the settings and click on create pipeline

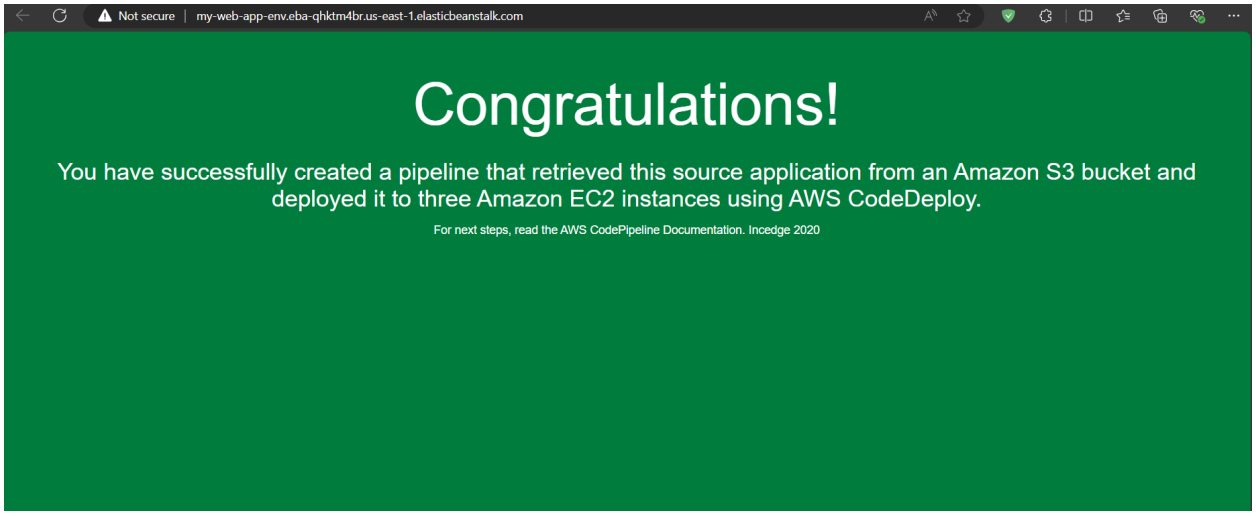




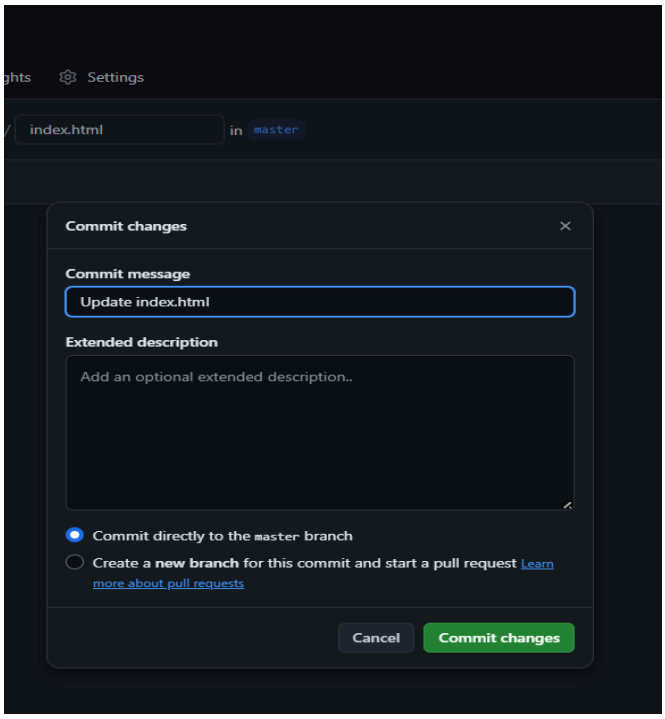
8. Once the Successfully created message appears, your pipeline is created.Then go ahead and check the URL provided in the EBS environment.



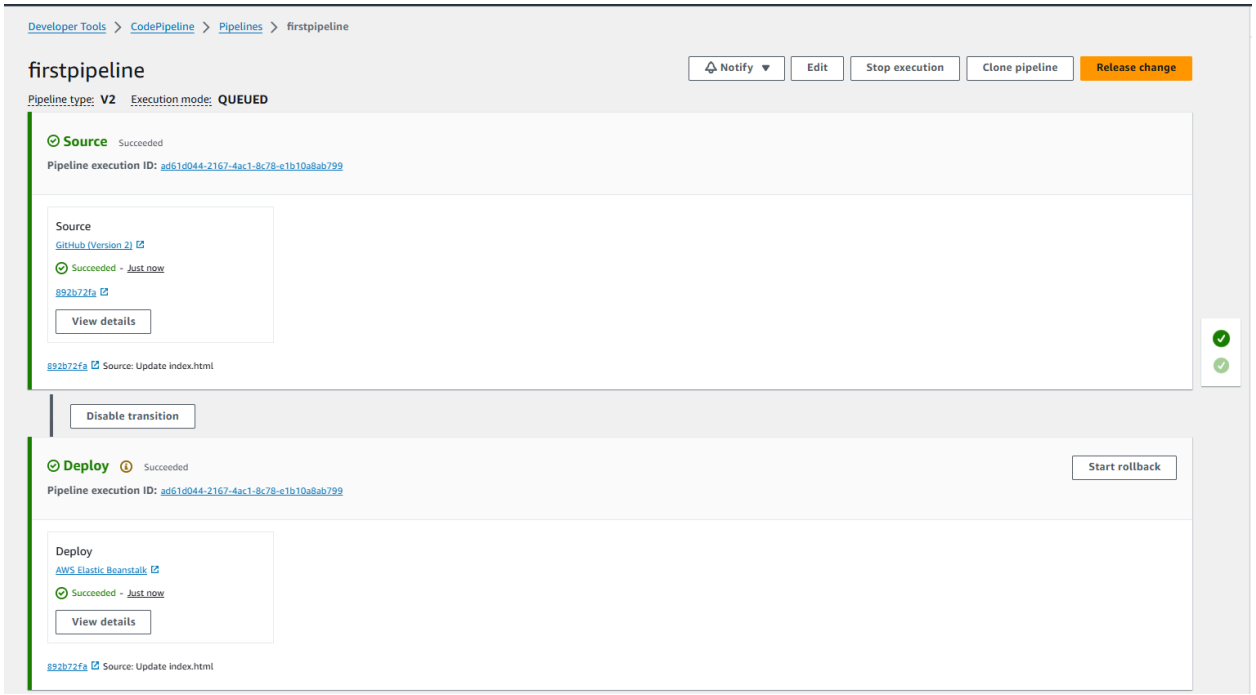
9. This is the website hosted from that forked repo in our beanstalk environment

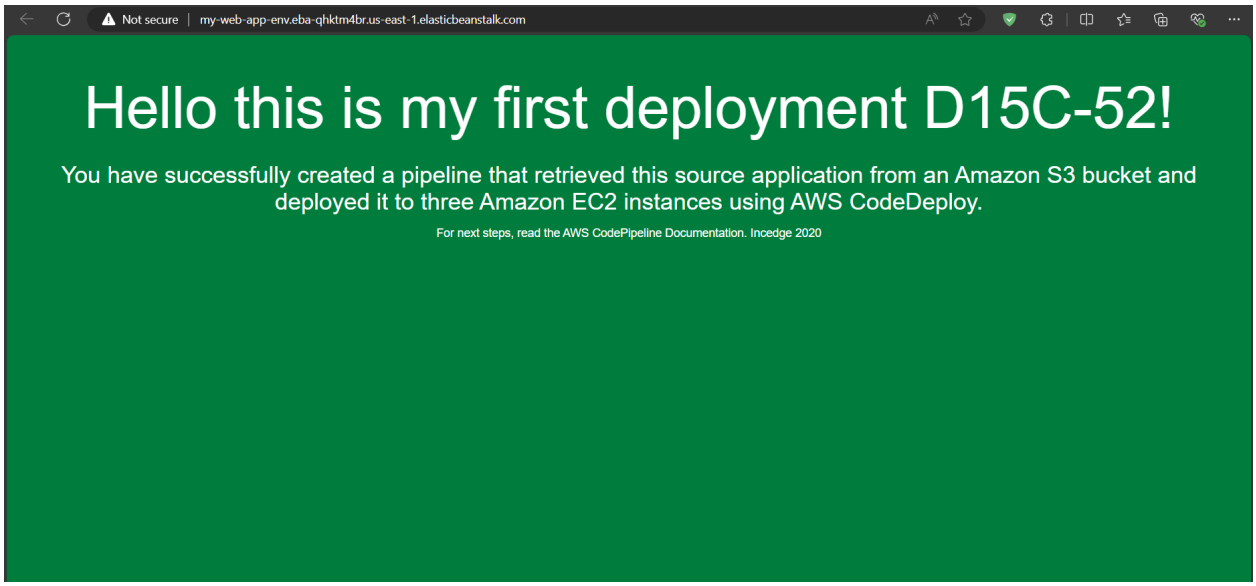


10. Go to the repository and make the changes in the index.html file and commit them



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

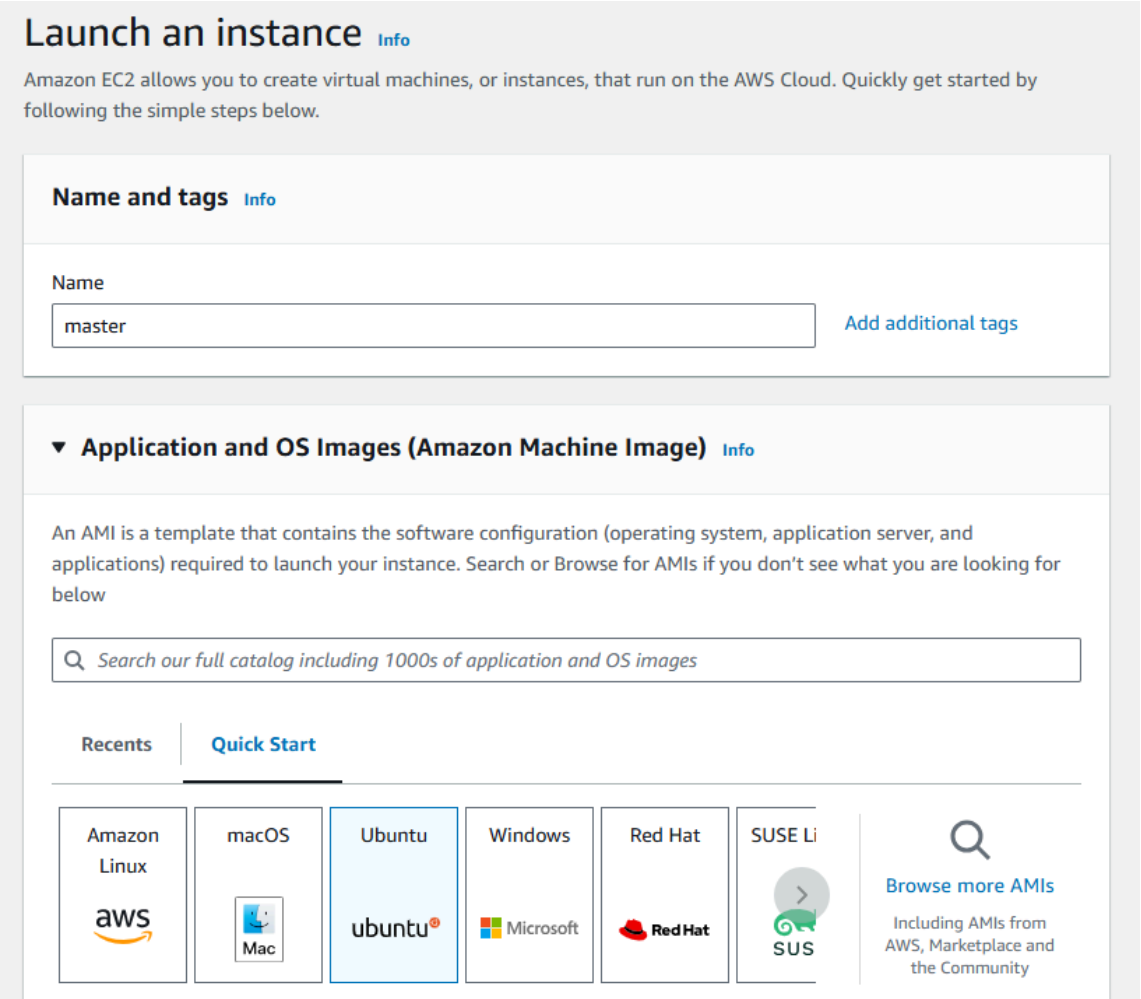

Developer Tools > CodePipeline > Pipelines > firstpipeline
firstpipeline
Notify
Edit
Stop execution
Clone pipeline
Release change
Pipeline type: V2
Execution mode: QUEUED
Source Succeeded
Pipeline execution ID: ad61d044-2167-4ac1-8c78-e1b10a8ab799
Source
GitHub (Version 2)
Succeeded - Just now
892b72fa
View details
892b72fa Source: Update index.html
Disable transition
Deploy Succeeded
Start rollback
Pipeline execution ID: ad61d044-2167-4ac1-8c78-e1b10a8ab799
Deploy
AWS Elastic Beanstalk
Succeeded - Just now
View details
892b72fa Source: Update index.html


Not secure | my-web-app-env.eba-qhktm4br.us-east-1.elasticbeanstalk.com
Hello this is my first deployment D15C-52!
You have successfully created a pipeline that retrieved this source application from an Amazon S3 bucket and deployed it to three Amazon EC2 instances using AWS CodeDeploy.
For next steps, read the AWS CodePipeline Documentation. Incedge 2020

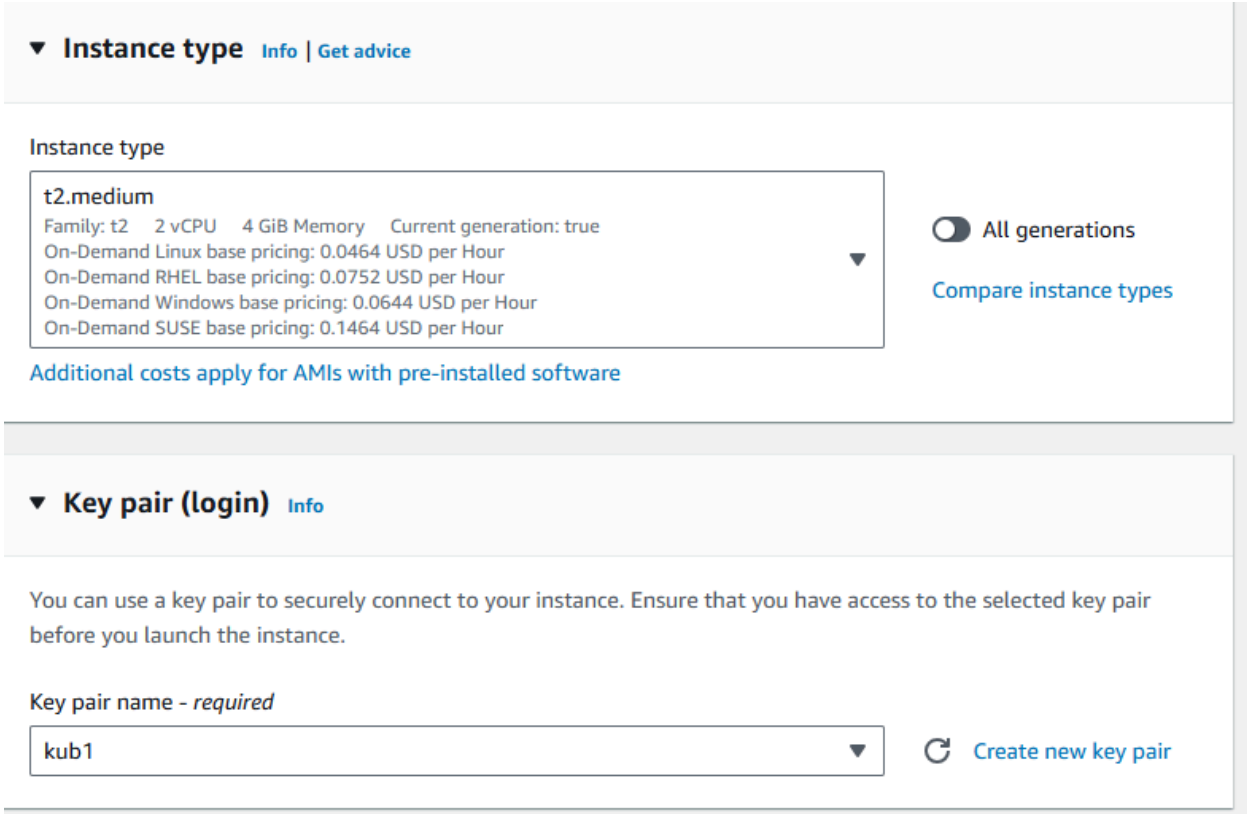
Experiment 3:

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

1. Create 3 EC-2 instances with all running on Ubuntu as OS with inbound SSH allowed and the proper key pair

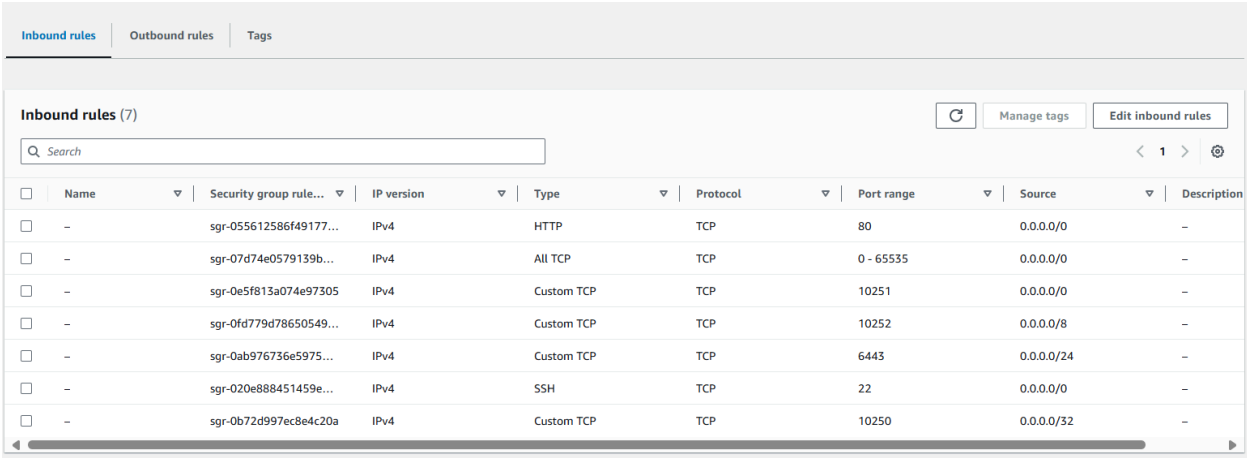


To efficient run kubernetes cluster select instance type of at least t2.medium as kubernetes recommends at least 2 vCPU to run smoothly



Also edit the inbound rules accordingly for the master as well as node/worker machines

Master :



Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
–	sgr-055612586f49177...	IPv4	HTTP	TCP	80	0.0.0.0/0	–
–	sgr-07d74e0579139b...	IPv4	All TCP	TCP	0 - 65535	0.0.0.0/0	–
–	sgr-0e5f813a074e97305	IPv4	Custom TCP	TCP	10251	0.0.0.0/0	–
–	sgr-0fd779d78650549...	IPv4	Custom TCP	TCP	10252	0.0.0.0/8	–
–	sgr-0ab976736e5975...	IPv4	Custom TCP	TCP	6443	0.0.0.0/24	–
–	sgr-020e888451459e...	IPv4	SSH	TCP	22	0.0.0.0/0	–
–	sgr-0b72d997ec8e4c20a	IPv4	Custom TCP	TCP	10250	0.0.0.0/32	–

Worker 1 and Worker 2:

Inbound rules (6)

Manage tags | Edit inbound rules

Search

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
–	sgr-0be6fa8163e229558	IPv4	Custom TCP	TCP	30000 - 32767	0.0.0.0/0	–
–	sgr-06d2c234c61af508e	IPv4	Custom TCP	TCP	10250	0.0.0.0/0	–
–	sgr-0e5853c91cde89d87	IPv4	HTTP	TCP	80	0.0.0.0/0	–
–	sgr-031248dbc56ff4687	IPv4	SSH	TCP	22	0.0.0.0/0	–
–	sgr-0bc0b5370fdf55a4f	IPv4	All traffic	All	All	0.0.0.0/0	–
–	sgr-0ddccad44425bb2...	IPv4	All TCP	TCP	0 - 65535	0.0.0.0/0	–

In this way create 3 instances namely master, worker-1 and worker-2

Instances (3) Info

Last updated less than a minute ago | Connect | Instance state | Actions | Launch instances

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

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6
master	i-0604dbbc26a0a4f01	Running	t2.medium	Initializing	View alarms +	us-east-1a	ec2-54-85-79-186.com...	54.85.79.186	–	–
worker-1	i-05548bd7fe0a7292f	Running	t2.medium	Initializing	View alarms +	us-east-1a	ec2-54-196-211-209.co...	54.196.211.209	–	–
worker-2	i-007a15dade39c85b0	Running	t2.medium	Initializing	View alarms +	us-east-1a	ec2-18-209-62-85.com...	18.209.62.85	–	–

2. SSH into all 3 machines each in separate terminal

   a. You can do it through the aws console directly

```
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 Wed Oct  2 08:10:57 UTC 2024

  System load:  0.09               Processes:             138
  Usage of /:   65.1% of 6.71GB    Users logged in:       1
  Memory usage: 19%                IPv4 address for enX0: 172.31.93.13
  Swap usage:   0%

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

   https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

6 updates can be applied immediately.
5 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

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


Last login: Wed Oct  2 07:24:25 2024 from 103.250.98.143
ubuntu@ip-172-31-93-13:~$
```

Or

b. Locate your key from the Downloads folder and open it in cmd and paste this command

ssh -i <-your-key->.pem ec2-user<ip-address of instance>

```
PS C:\Users\Avan> cd Downloads
PS C:\Users\Avan\Downloads> ssh -i "kub1.pem" ubuntu@ec2-44-201-103-71.compute-1.amazonaws.
com
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 Wed Oct  2 07:24:24 UTC 2024

  System load:  0.08              Processes:             117
  Usage of /:   23.1% of 6.71GB   Users logged in:       0
  Memory usage: 5%                IPv4 address for enX0: 172.31.93.13
  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


Last login: Wed Oct  2 07:22:47 2024 from 103.250.98.143
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

With this you can continue your commands through local terminal

3. From now on, until mentioned, perform these steps on all 3 machines.

Install Docker

sudo yum install docker -y

```
[ec2-user@ip-172-31-31-212 ~]$ sudo yum install docker -y
Last metadata expiration check: 0:33:43 ago on Thu Sep 12 13:11:13 2024.
Dependencies resolved.
================================================================================
 Package                    Architecture                    Version
================================================================================
Installing:
 docker                     x86_64                          25.0.6-1.amzn2023.0.2
Installing dependencies:
 containerd                 x86_64                          1.7.20-1.amzn2023.0.1
 iptables-libs              x86_64                          1.8.8-3.amzn2023.0.2
 iptables-nft               x86_64                          1.8.8-3.amzn2023.0.2
 libcgroup                  x86_64                          3.0-1.amzn2023.0.1
 libnetfilter_conntrack     x86_64                          1.0.8-2.amzn2023.0.2
 libnfnetlink               x86_64                          1.0.1-19.amzn2023.0.2
 libnftnl                   x86_64                          1.2.2-2.amzn2023.0.2
 pigz                       x86_64                          2.5-1.amzn2023.0.3
 runc                       x86_64                          1.1.13-1.amzn2023.0.1

Transaction Summary
```

Then, configure cgroup in a daemon.json file by using following commands

- cd /etc/docker

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

```
[ec2-user@ip-172-31-20-75 ~]$ cd /etc/docker
[ec2-user@ip-172-31-20-75 docker]$ cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"],
"log-driver": "json-file",
"log-opts": {
"max-size": "100m"
},
"storage-driver": "overlay2"
}
EOF
{
"exec-opts": ["native.cgroupdriver=systemd"],
"log-driver": "json-file",
"log-opts": {
"max-size": "100m"
},
"storage-driver": "overlay2"
}
[ec2-user@ip-172-31-20-75 docker]$ ls
daemon.json  kubectl
```

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

```
[ec2-user@ip-172-31-31-212 docker]$ sudo systemctl enable docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
[ec2-user@ip-172-31-31-212 docker]$ sudo systemctl daemon-reload
[ec2-user@ip-172-31-31-212 docker]$ sudo systemctl restart docker
[ec2-user@ip-172-31-31-212 docker]$ docker -v
Docker version 25.0.5, build 5dc9bcc
[ec2-user@ip-172-31-31-212 docker]$
```

4. **Install Kubernetes on all 3 machines**

Run the below command to install Kubernetes.

curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

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

```
ubuntu@ip-172-31-93-13:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.ke
y | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
ubuntu@ip-172-31-93-13:~$ 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/kubern
etes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stab
le:/v1.31/deb/ /
ubuntu@ip-172-31-93-13:~$ sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:5 https://download.docker.com/linux/ubuntu noble InRelease
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/de
b  InRelease [1186 B]
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/de
b  Packages [4865 B]
Fetched 6051 B in 1s (10.6 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 section in apt-key(8) for d
etails.
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
```

```
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.
```

Run this command to install the kubelet server

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

```
ubuntu@ip-172-31-93-13:~$ sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:5 https://download.docker.com/linux/ubuntu noble InRelease
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/de
b  InRelease [1186 B]
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/de
b  Packages [4865 B]
Fetched 6051 B in 1s (10.6 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 section in apt-key(8) for d
etails.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools kubernetes-cni
```

Run the below commands to containerize the server

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

```
ubuntu@ip-172-31-93-13:~$ 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 6 not upgraded.
```

Now the next command initializes the containerd configuration file with default settings, creating the config.toml file, which will be used by containerd to manage containers

```
ubuntu@ip-172-31-93-13:~$ 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

[metrics]
  address = ""
  grpc_histogram = false
```

```
[proxy_plugins]

[stream_processors]

  [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
```

Restarting and running the container with these commands

- sudo systemctl restart containerd

- sudo systemctl enable containerd
- sudo systemctl status containerd

```
ubuntu@ip-172-31-93-13:~$ 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; p>
     Active: active (running) since Wed 2024-10-02 07:41:44 UTC; 207ms ago
       Docs: https://containerd.io
   Main PID: 4835 (containerd)
      Tasks: 7
     Memory: 13.7M (peak: 14.1M)
        CPU: 64ms
     CGroup: /system.slice/containerd.service
             └─4835 /usr/bin/containerd

Oct 02 07:41:44 ip-172-31-93-13 containerd[4835]: time="2024-10-02T07:41:44>
Oct 02 07:41:44 ip-172-31-93-13 containerd[4835]: time="2024-10-02T07:41:44>
Oct 02 07:41:44 ip-172-31-93-13 containerd[4835]: time="2024-10-02T07:41:44>
Oct 02 07:41:44 ip-172-31-93-13 containerd[4835]: time="2024-10-02T07:41:44>
Oct 02 07:41:44 ip-172-31-93-13 containerd[4835]: time="2024-10-02T07:41:44>
Oct 02 07:41:44 ip-172-31-93-13 containerd[4835]: time="2024-10-02T07:41:44>
Oct 02 07:41:44 ip-172-31-93-13 containerd[4835]: time="2024-10-02T07:41:44>
Oct 02 07:41:44 ip-172-31-93-13 containerd[4835]: time="2024-10-02T07:41:44>
Oct 02 07:41:44 ip-172-31-93-13 systemd[1]: Started containerd.service - co>
Oct 02 07:41:44 ip-172-31-93-13 containerd[4835]: time="2024-10-02T07:41:44>
```

Below command :

- **sudo apt-get install -y socat**

installs the socat utility, a versatile networking tool used for bidirectional data transfer between two endpoints, commonly utilized in Kubernetes for port forwarding and networking tasks.

```
ubuntu@ip-172-31-93-13:~$ 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 6 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.0.0-4
build3 [374 kB]
Fetched 374 kB in 0s (18.0 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68203 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.
```

5. **Perform this ONLY on the Master machine**

   Initialize kubernetes by typing below command

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

```
ubuntu@ip-172-31-93-13:~$ 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 connec
tion
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W1002 07:42:59.444277    5045 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 reco
mmended 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-172-31-93-13 kubernetes kubernet
es.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1
172.31.93.13]
[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-172-31-93-13 localhost] and IP
s [172.31.93.13 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-93-13 localhost] and IPs
[172.31.93.13 127.0.0.1 ::1]
```

```
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 172.31.93.13:6443 --token v6qmnw.lhucizuo02h2fkwo \
        --discovery-token-ca-cert-hash sha256:78b5edd6f35bd7763eb06d22cdfed85728c12119f083a
11e434b96fba530da0c
```

Copy the mkdir and chown commands from the top and execute them

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

Copy this join link and save it in clipboard (copy from your output as it different for each master instance)

Example :

```
sudo kubeadm join 172.31.93.13:6443 --token v6qmnw.lhucizuo02h2fkwo \
        --discovery-token-ca-cert-hash
sha256:78b5edd6f35bd7763eb06d22cdfed85728c12119f083a11e434b96fba530da0c
```

```
ubuntu@ip-172-31-84-116:~$ sudo kubeadm join 172.31.93.13:6443 --token v6qmnw.lhucizuo02h2fkwo \
--discovery-token-ca-cert-hash sha256:78b5edd6f35bd7763eb06d22cdfed85728c12119f083a11e434b96fba530da0c
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 501.194701ms
[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.
```

Running the kubectl get nodes to check the cluster connection

```
ubuntu@ip-172-31-93-13:~$ kubectl get nodes
NAME                STATUS     ROLES           AGE     VERSION
ip-172-31-84-116    NotReady   <none>          39s     v1.31.1
ip-172-31-91-194    NotReady   <none>          35s     v1.31.1
ip-172-31-93-13     NotReady   control-plane   3m59s   v1.31.1
```

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-172-31-93-13:~$ 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
```

6. Now check the status for kubelet server
   sudo systemctl status kubelet

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

```
ubuntu@ip-172-31-93-13:~$ kubectl get nodes -o wide
NAME                STATUS   ROLES           AGE     VERSION   INTERNAL-IP
  EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-84-116    Ready    Node1           4m52s   v1.31.1   172.31.84.116
  <none>        Ubuntu 24.04.1 LTS   6.8.0-1016-aws   containerd://1.7.12
ip-172-31-91-194    Ready    Node2           4m48s   v1.31.1   172.31.91.194
  <none>        Ubuntu 24.04.1 LTS   6.8.0-1016-aws   containerd://1.7.12
ip-172-31-93-13     Ready    control-plane   8m12s   v1.31.1   172.31.93.13
  <none>        Ubuntu 24.04.1 LTS   6.8.0-1016-aws   containerd://1.7.12
```

8. Renaming the master and worker nodes
   kubectl label node ip-172-31-84-116 kubernetes.io/role=Node1
   (Similarly for your own ip you can rename the nodes and master)

```
ubuntu@ip-172-31-93-13:~$ kubectl label node ip-172-31-84-116 kubernetes.io/
role=Node1
node/ip-172-31-84-116 labeled
```

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

```
ubuntu@ip-172-31-93-13:~$ kubectl get nodes -o wide
NAME                STATUS   ROLES           AGE     VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION   C
ONTAINER-RUNTIME
ip-172-31-84-116    Ready    Node1           4m52s   v1.31.1   172.31.84.116   <none>        Ubuntu 24.04.1 LTS   6.8.0-1016-aws   c
ontainerd://1.7.12
ip-172-31-91-194    Ready    Node2           4m48s   v1.31.1   172.31.91.194   <none>        Ubuntu 24.04.1 LTS   6.8.0-1016-aws   c
ontainerd://1.7.12
ip-172-31-93-13     Ready    control-plane   8m12s   v1.31.1   172.31.93.13    <none>        Ubuntu 24.04.1 LTS   6.8.0-1016-aws   c
ontainerd://1.7.12
```

```
ubuntu@ip-172-31-93-13:~$ kubectl get nodes
NAME                STATUS   ROLES           AGE     VERSION
ip-172-31-84-116    Ready    Node1           5m7s    v1.31.1
ip-172-31-91-194    Ready    Node2           5m3s    v1.31.1
ip-172-31-93-13     Ready    control-plane   8m27s   v1.31.1
```

x

Conclusion :

In this experiment, we aimed to deploy Kubernetes in Docker, connecting a master node with two worker nodes. We encountered several challenges, starting with SSH inbound rule misconfigurations, which were resolved by enabling the correct rules for secure access. It became clear that using **t2.medium** or **t3** instances was essential for sufficient resources to run Kubernetes smoothly. Despite these adjustments, the worker nodes failed to join the cluster. Although the master node was ready, the issue seemed to stem from improper configuration of the worker nodes' kubelet or a networking problem, such as the worker nodes being unable to reach the master node's API server. This could be due to incorrect firewall settings, missing API server certificates, or failure in configuring the kubeadm join process on the worker nodes.

Experiment 4:

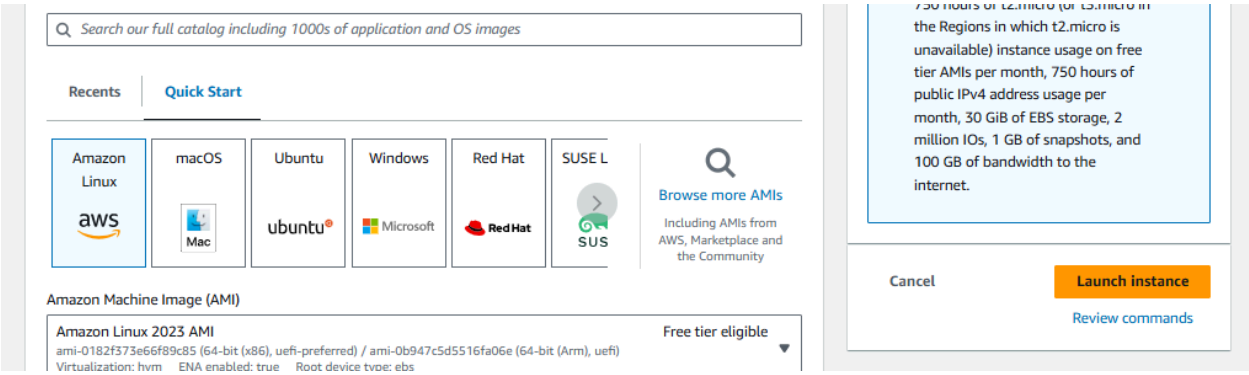
Aim: To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

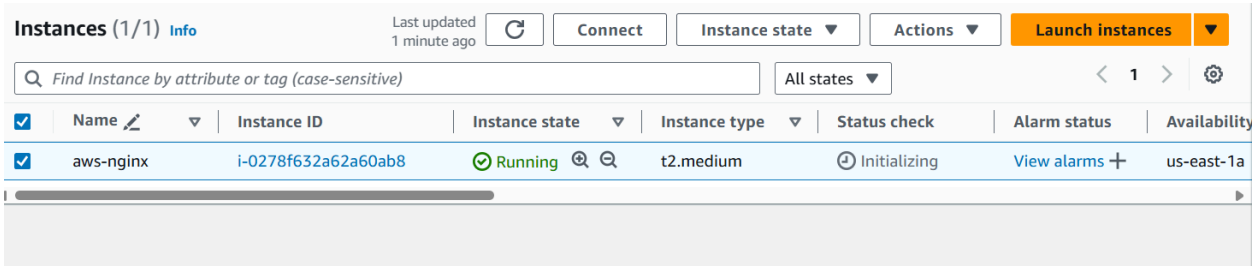
STEPS:

#Creating EC2 instance:

1. Create 1 EC-2 instances with all running on Amazon Linux as OS with inbound SSH allowed and the proper key

- Select Amazon linux as OS image
- To efficient run kubernetes cluster select instance type of at least t2.medium as kubernetes recommends at least 2 vCPU to run smoothly





2. **ssh connection checking through the local terminal with the help of key**

```
C:\Users\Avan>ssh -i "C:\Users\Avan\Downloads\kub1.pem" ec2-user@3.85.237.93
The authenticity of host '3.85.237.93 (3.85.237.93)' can't be established.
ED25519 key fingerprint is SHA256:Nz2iC26abFyhATf/8i4F0IgWmDoxTXBbzY9/NkMwYyM.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.85.237.93' (ED25519) to the list of known hosts.
       #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Thu Sep 12 13:11:49 2024 from 18.206.107.29
```

3. **Install Docker**

- sudo dnf update
- sudo dnf install docker
- sudo systemctl enable docker
- sudo systemctl start docker

```
[ec2-user@ip-172-31-16-177 ~]$ sudo dnf update
sudo dnf install docker
sudo systemctl enable docker
sudo systemctl start docker
Last metadata expiration check: 0:01:07 ago on Sat Sep 14 11:26:29 2024.
Dependencies resolved.
Nothing to do.
Complete!
Last metadata expiration check: 0:01:08 ago on Sat Sep 14 11:26:29 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
```

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

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
  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
  pigz-2.5-1.amzn2023.0.3.x86_64                runc-1.1.13-1.amzn2023.0.1.x86_64

Complete!
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
```

To test whether docker is successfully running, use command

- **sudo docker run hello-world**

```
[ec2-user@ip-172-31-16-177 ~]$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:91fb4b041da273d5a3273b6d587d62d518300a6ad268b28628f74997b93171b2
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.
```

Then, configure cgroup in a daemon.json file.This allows kubernetes to manage host more efficiently

- cd /etc/docker

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

```
[ec2-user@ip-172-31-16-177 ~]$ cd /etc/docker
[ec2-user@ip-172-31-16-177 docker]$ cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
[ec2-user@ip-172-31-16-177 docker]$ sudo systemctl daemon-reload
sudo systemctl restart docker
[ec2-user@ip-172-31-16-177 docker]$
```

4. **Install Kubernetes**
   SELinux needs to be disabled before configuring kubelet

- sudo setenforce 0
- sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config

Now we need to install kubectl

Set up repository:

- cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
  [kubernetes]
  name=Kubernetes
  baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
  enabled=1
  gpgcheck=1
  gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
  EOF

- sudo yum install -y kubectl

```
[ec2-user@ip-172-31-16-177 docker]$ sudo yum update
Kubernetes
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-16-177 docker]$ sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
Last metadata expiration check: 0:00:12 ago on Sat Sep 14 11:53:54 2024.
Dependencies resolved.
================================================================================================================
 Package                                          Architecture                        Version
================================================================================================================
Installing:
 kubeadm                                          x86_64                              1.30.5-150500.1.1
 kubectl                                          x86_64                              1.30.5-150500.1.1
 kubelet                                          x86_64                              1.30.5-150500.1.1
Installing dependencies:
 conntrack-tools                                  x86_64                              1.4.6-2.amzn2023.0.2
 cri-tools                                        x86_64                              1.30.1-150500.1.1
 kubernetes-cni                                   x86_64                              1.4.0-150500.1.1
 libnetfilter_cthelper                            x86_64                              1.0.0-21.amzn2023.0.2
 libnetfilter_cttimeout                           x86_64                              1.0.0-19.amzn2023.0.2
 libnetfilter_queue                               x86_64                              1.0.5-2.amzn2023.0.2

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

Checking the version

```
[ec2-user@ip-172-31-16-177 docker]$ kubectl version
Client Version: v1.30.5
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
```

We have installed **successfully** installed kubernetes .After installing Kubernetes, we need to configure internet options to allow bridging.

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

```
[ec2-user@ip-172-31-16-177 docker]$ sudo swapoff -a
echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
net.bridge.bridge-nf-call-iptables=1
net.bridge.bridge-nf-call-iptables = 1
```

- **Disable SELINUX**

Type the following command

- **sudo nano /etc/selinux/config**

and set the value of **SELINUX=disabled** instead of **SELINUX=permissive**

Save the file by pressing ctrl+o then press enter then press ctrl+x

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
# See also:
# https://docs.fedoraproject.org/en-US/quick-docs/getting-started-with-selinux/#getting-st
#
# NOTE: In earlier Fedora kernel builds, SELINUX=disabled would also
# fully disable SELinux during boot. If you need a system with SELinux
# fully disabled instead of SELinux running with no policy loaded, you
# need to pass selinux=0 to the kernel command line. You can use grubby
# to persistently set the bootloader to boot with selinux=0:
#
#    grubby --update-kernel ALL --args selinux=0
#
# To revert back to SELinux enabled:
#
#    grubby --update-kernel ALL --remove-args selinux
#
SELINUX=disabled
# SELINUXTYPE= can take one of these three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

Then reboot the system using

- **sudo reboot**

After rebooting we need to make ssh connection with machine after it gets disconnected

```
[ec2-user@ip-172-31-16-177 docker]$ sudo reboot

Broadcast message from root@localhost on pts/1 (Sat 2024-09-14 11:58:50 UTC):

The system will reboot now!
```

Now if we type command

- **sestatus** (*then it show disabled*)

```
[ec2-user@ip-172-31-16-177 docker]$ sestatus
SELinux status:                 disabled
[ec2-user@ip-172-31-16-177 docker]$
```

5. **Initialize the Kubecluster**

Install packages socat and iproute-tc and conntrack to avoid prelight errors

- sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all

```
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 172.31.16.177:6443 --token z7ersu.jg6wsvrvkywtavqd \
        --discovery-token-ca-cert-hash sha256:c3f37da241eae40a745ab6c8ab9f631c88778927d75ebeb8f37ce1a6f911c8cb
```

Copy the mkdir and chown commands from the top and execute them

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

Restart the kubelet server

- sudo systemctl restart kubelet

Then, add a common networking plugin called flannel as mentioned in the code.

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

```
[ec2-user@ip-172-31-16-177 docker]$   mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config
[ec2-user@ip-172-31-16-177 docker]$ sudo systemctl restart kubelet
[ec2-user@ip-172-31-16-177 docker]$ kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
error: flag needs an argument: 'f' in -f
See 'kubectl apply --help' for usage.
-bash: https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml: No such file or directory
[ec2-user@ip-172-31-16-177 docker]$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
```

Now to see your nodes are running type this command

- **kubectl get nodes**

```
[ec2-user@ip-172-31-16-177 docker]$ kubectl get nodes
NAME                              STATUS   ROLES           AGE     VERSION
ip-172-31-16-177.ec2.internal     Ready    control-plane   6m23s   v1.30.5
```

(***If you face the connectivity is not established/ refused try this command for 2-3 times and wait for a while the nodes will be set to status ready and if not happening then restart the kubelet server again)***

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

```
[ec2-user@ip-172-31-16-177 docker]$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
```

Use 'kubectl get pods' to verify if the deployment was properly created and the pod is working correctly.

```
[ec2-user@ip-172-31-16-177 docker]$ kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
nginx-deployment-77d8468669-9kkmf  0/1     Pending   0          29s
nginx-deployment-77d8468669-zcdsc  0/1     Pending   0          29s
[ec2-user@ip-172-31-16-177 docker]$
```

As we can see our pods are in pending state we need to make pods untainted
Type kubectl get nodes to see name of node

```
[ec2-user@ip-172-31-16-177 docker]$ kubectl get nodes
NAME                              STATUS   ROLES           AGE     VERSION
ip-172-31-16-177.ec2.internal     Ready    control-plane   6m23s   v1.30.5
```

Copy the name of the node (ip-172-31-16-177.ec2.internal)
Then type command

- **kubectl taint nodes ip-172-31-16-177.ec2.internal <u>node-role.kubernetes.io/control-plane-</u>**

```
[ec2-user@ip-172-31-16-177 docker]$ kubectl taint nodes ip-172-31-16-177.ec2.internal node-role.kubernetes.io/control-plane-
node/ip-172-31-16-177.ec2.internal untainted
```

After executing above command, check again status of pods if still pending then restart kubelet wait for sometime and let the server load everything and check by running the pods command

```
[ec2-user@ip-172-31-16-177 docker]$ kubectl get pods
NAME                                READY   STATUS    RESTARTS        AGE
nginx-deployment-77d8468669-9kkmf   1/1     Running   3 (3m5s ago)    14m
nginx-deployment-77d8468669-zcdsc   1/1     Running   3 (50s ago)     14m
[ec2-user@ip-172-31-16-177 docker]$
```

As we can see our pods are running and status is set

6. **Deployment of Nginx Server**

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

- kubectl port-forward <POD_NAME> 8080:80

Note: if you are getting connection refused error then restart kubelet server

```
[ec2-user@ip-172-31-16-177 docker]$ kubectl port-forward nginx-deployment-77d8468669-9kkmf 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
```

As port forwarding is active so we cannot type other commands.
Open new terminal window and make ssh connection to same machine OR we can open instance of same machine in new browser tab
And type command

- **curl --head http://127.0.0.1:8080**

```
[ec2-user@ip-172-31-16-177 ~]$ curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Sat, 14 Sep 2024 12:28:13 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
```

Response status 200 (OK) indicates that our nginx server is running successfully on kubernetes

Conclusion:

The deployment was completed successfully, but I ran into connection issues shortly after. Even though I tried troubleshooting through the terminal, the problem persisted, forcing me to restart the server several times before the issue was resolved. The pods created were not running because nodes were tainted so we had to make them untainted. During this process, I gained valuable insights into deployment installation, understanding both the setup and the challenges that can arise with Kubernetes. It was a bit frustrating at times, but the experience taught me how critical it is to troubleshoot effectively and manage the deployment environment. Ultimately, and the deployment was fully functional.

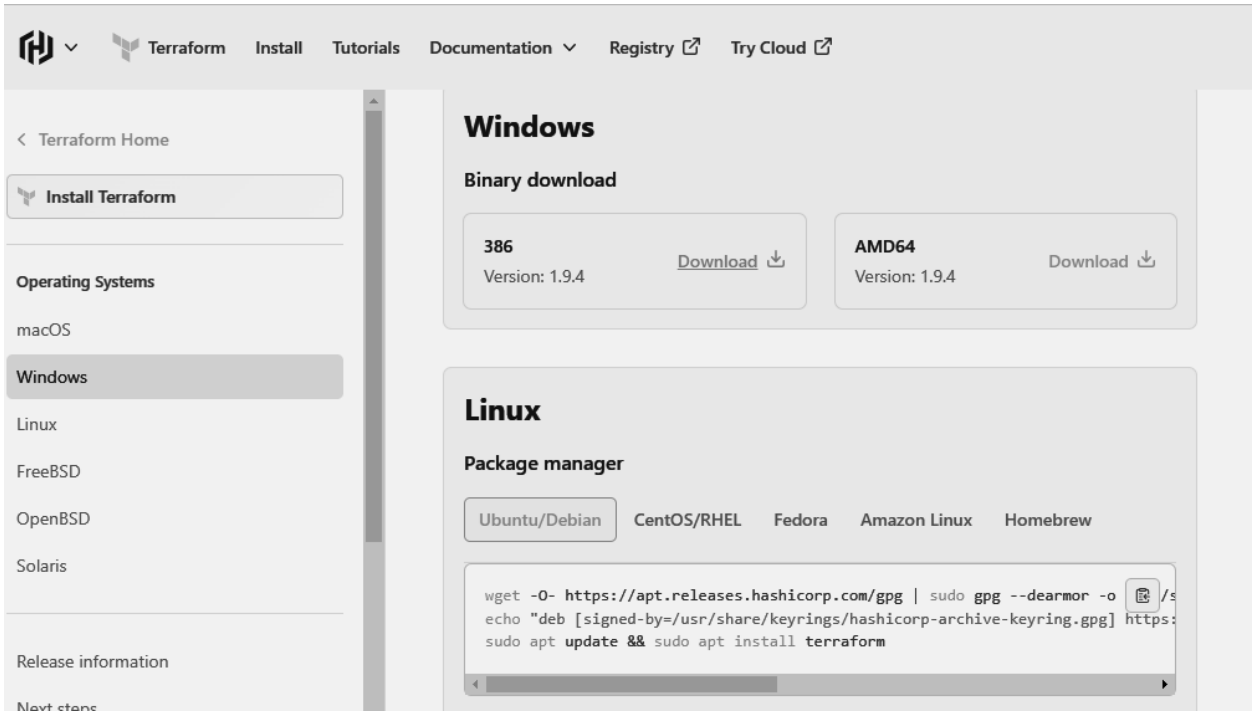
Experiment 5 :

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

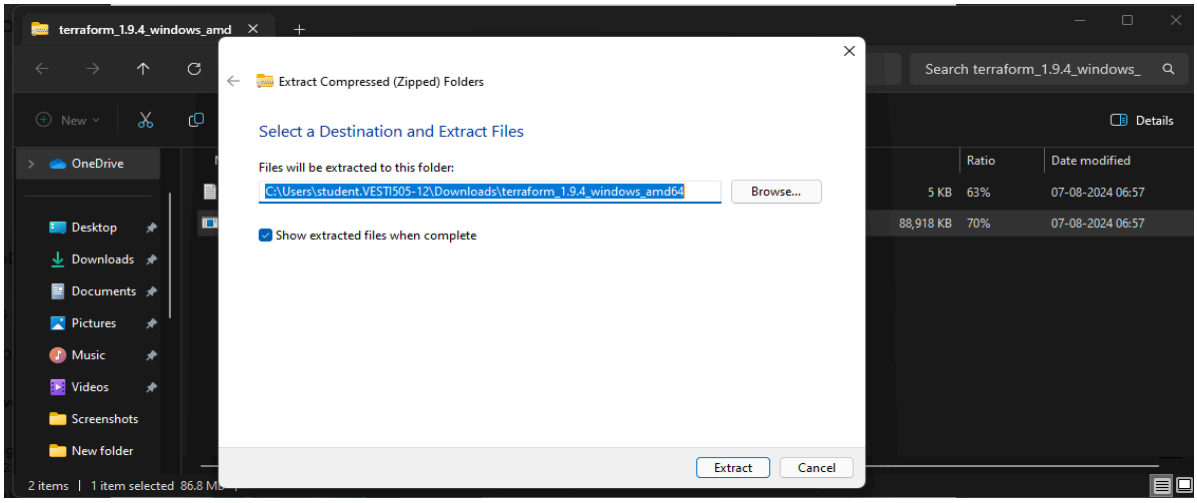
Installation and Configuration of Terraform in Windows

1: To install Terraform, First Download the Terraform Cli Utility for windows from terraforms official website: :https://www.terraform.io/downloads.html

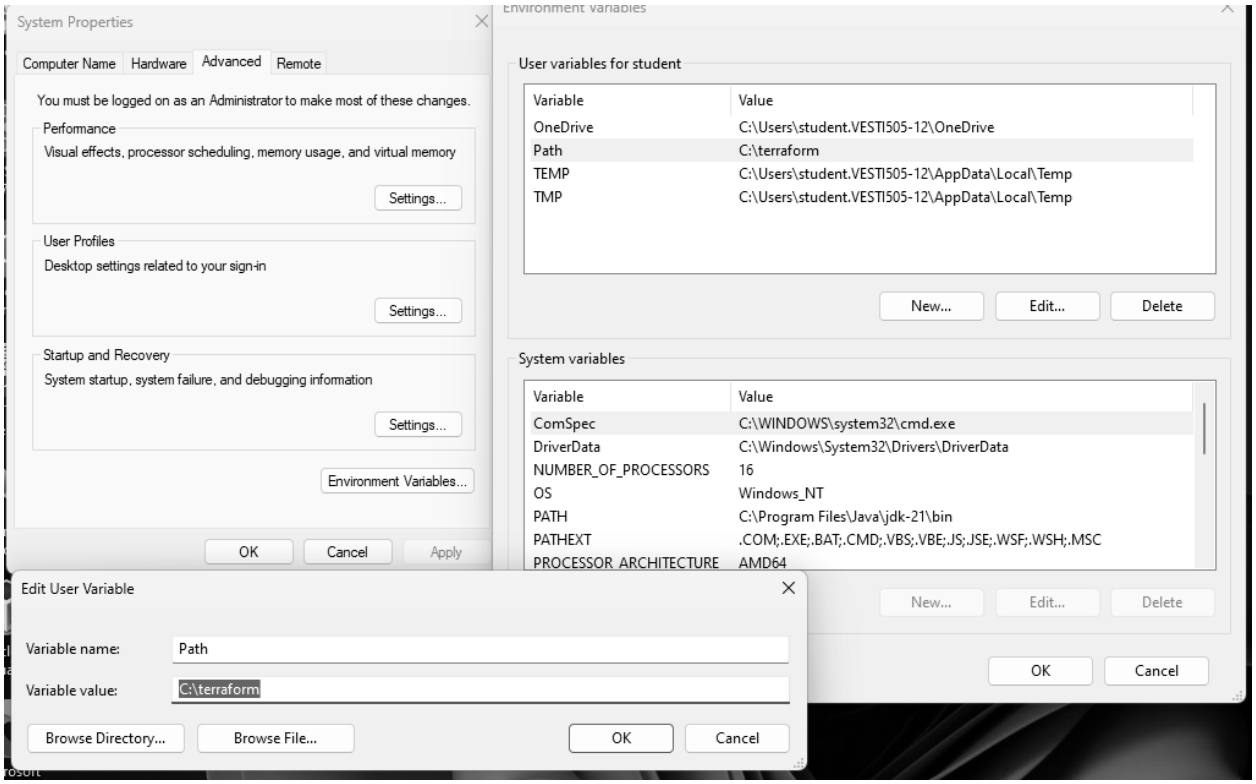
Select the Operating System Windows followed by either 32 bit or 64 bit based on your OS type.



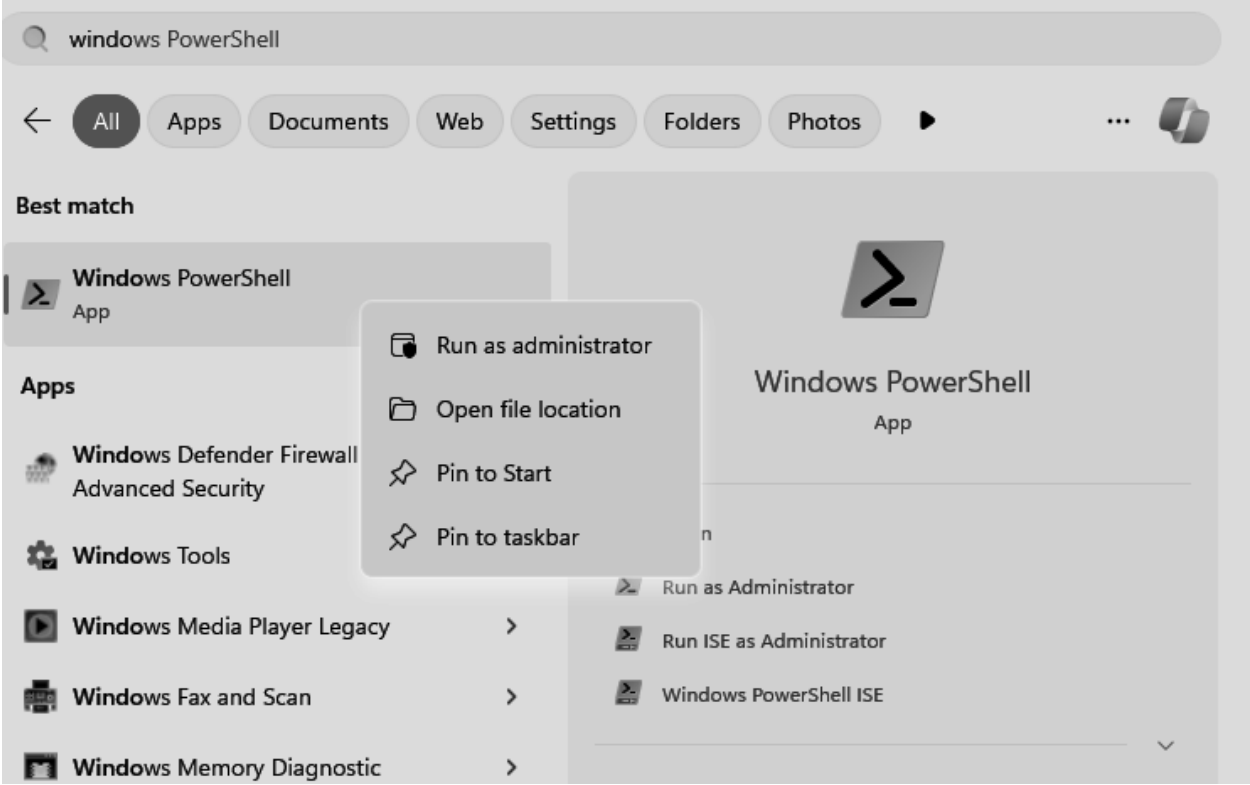
2. Extract the downloaded setup file Terraform.exe in C:\Terraformdirectory



3: Set the System path for Terraform in Environment Variables



4: Open PowerShell with Admin Access



5 : Open Terraform in PowerShell and check its functionality

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

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

PS C:\Users\student.VESTI505-12> terraform
Usage: terraform [global options] <subcommand> [args]

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

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

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

Experiment 6:

Aim : Exp 6 To Build, change, and destroy AWS / GCP /Microsoft Azure/ DigitalOcean infrastructure Using Terraform.(S3 bucket or Docker)

Creating the docker image using terraform

1: Check the docker version and functionality if its not downloaded you can download it from https://www.docker.com/

```
C:\Users\Avan\Desktop>docker --version
Docker version 27.0.3, build 7d4bcd8
```

```
C:\Users\Avan\Desktop>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
```

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

2: Firstly create a new folder named **'Docker'** in the **'TerraformScripts'** folder. Then create a new docker.tf file using Atom editor (or you can use vscode) and write the following contents into it to create a Ubuntu Linux container.

```
terraform {
  required_providers {
```

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

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

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

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

docker.tf

Docker > docker.tf

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

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

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

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

3: Execute **Terraform Init** command to initialize the resources (*Make sure you are in the Docker directory before executing the command*)

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

Terraform has been successfully initialized!

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

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

4. Execute **Terraform plan** to see the available resources

```
C:\Users\Avan\Desktop\TerraformScripts\Docker>terraform plan

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

Terraform will perform the following actions:

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

```
  # 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.
```

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

```
C:\Users\Avan\Desktop\TerraformScripts\Docker>terraform apply
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=23364bd2196c264de0ff6af7150449ffea25e1a9f7553ac6955b95679547526e]

Note: Objects have changed outside of Terraform

Terraform detected the following changes made outside of Terraform since the last "terraform apply" which may have affected this plan:

  # docker_image.ubuntu has been deleted
  - resource "docker_image" "ubuntu" {
      id          = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest"
    - image_id    = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
      name        = "ubuntu:latest"
      # (2 unchanged attributes hidden)
    }


Unless you have made equivalent changes to your configuration, or ignored the relevant attributes using ignore_changes, the following plan may include actions
to undo or respond to these changes.

─────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────

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

Terraform will perform the following actions:

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

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

      + healthcheck (known after apply)

      + labels (known after apply)
    }

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

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

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

  Enter a value: yes

docker_image.ubuntu: Creating...
docker_image.ubuntu: Still creating... [10s elapsed]
docker_image.ubuntu: Creation complete after 14s [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Creating...
docker_container.foo: Creation complete after 1s [id=7e9ee45651ce4057af54ab9bb95ef29084eb679a0f4114e88a5e0887c6480b8c]

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

6. Docker images before executing this command

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

Docker images after the execution of command

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

7. Executing the command **terraform providers** a provider is a plugin that allows terraform to interact with APIs of external services, enabling to manage the resources offered by those services.

```
C:\Users\Student\Desktop\TerraformsScripts\Docker>terraform providers

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

Providers required by state:

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

8. Running the command **terraform validate** to check if the configuration is valid

```
C:\Users\Student\Desktop\TerraformsScripts\Docker>terraform validate
Success! The configuration is valid.
```

9. The **terraform state list** is used to display a list of resources managed by Terraform within the current state file

```
C:\Users\Student\Desktop\TerraformsScripts\Docker>terraform state list
docker_container.foo
docker_image.ubuntu
```

10. Execute **Terraform destroy** to delete the configuration, which will automatically delete the Ubuntu Container.

```
C:\Users\Avan\Desktop\TerraformScripts\Docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=7e9ee45651ce4057af54ab9bb95ef29084eb679a0f4114e88a5e0887c6480b8c]

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

Terraform will perform the following actions:

  # docker_container.foo will be destroyed
  - resource "docker_container" "foo" {
      - attach            = false -> null
      - command           = [
          - "sleep",
          - "3600",
        ] -> null
      - cpu_shares        = 0 -> null
      - dns               = [] -> null
      - dns_opts          = [] -> null
      - dns_search        = [] -> null
      - entrypoint        = [] -> null
      - env               = [] -> null
      - gateway           = "172.17.0.1" -> null
      - group_add         = [] -> null
      - hostname          = "7e9ee45651ce" -> null
      - id                = "7e9ee45651ce4057af54ab9bb95ef29084eb679a0f4114e88a5e0887c6480b8c" -> null
      - image             = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
      - init              = false -> null
      - ip_address        = "172.17.0.2" -> null
      - ip_prefix_length  = 16 -> null
      - ipc_mode          = "private" -> null
      - links             = [] -> null
      - log_driver        = "json-file" -> null
      - log_opts          = {} -> null
      - logs              = false -> null
      - max_retry_count   = 0 -> null
      - memory            = 0 -> null
      - memory_swap       = 0 -> null
      - must_run          = true -> null
      - name              = "foo" -> null
```

```
      - stop_timeout      = 0 -> null
      - storage_opts      = {} -> null
      - sysctls           = {} -> null
      - tmpfs             = {} -> null
      - tty               = false -> null
        # (8 unchanged attributes hidden)
    }

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

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

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

  Enter a value: yes

docker_container.foo: Destroying... [id=7e9ee45651ce4057af54ab9bb95ef29084eb679a0f4114e88a5e0887c6480b8c]
docker_container.foo: Destruction complete after 0s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image.ubuntu: Destruction complete after 1s

Destroy complete! Resources: 2 destroyed.
```

11. Docker images after the destroy command execution

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

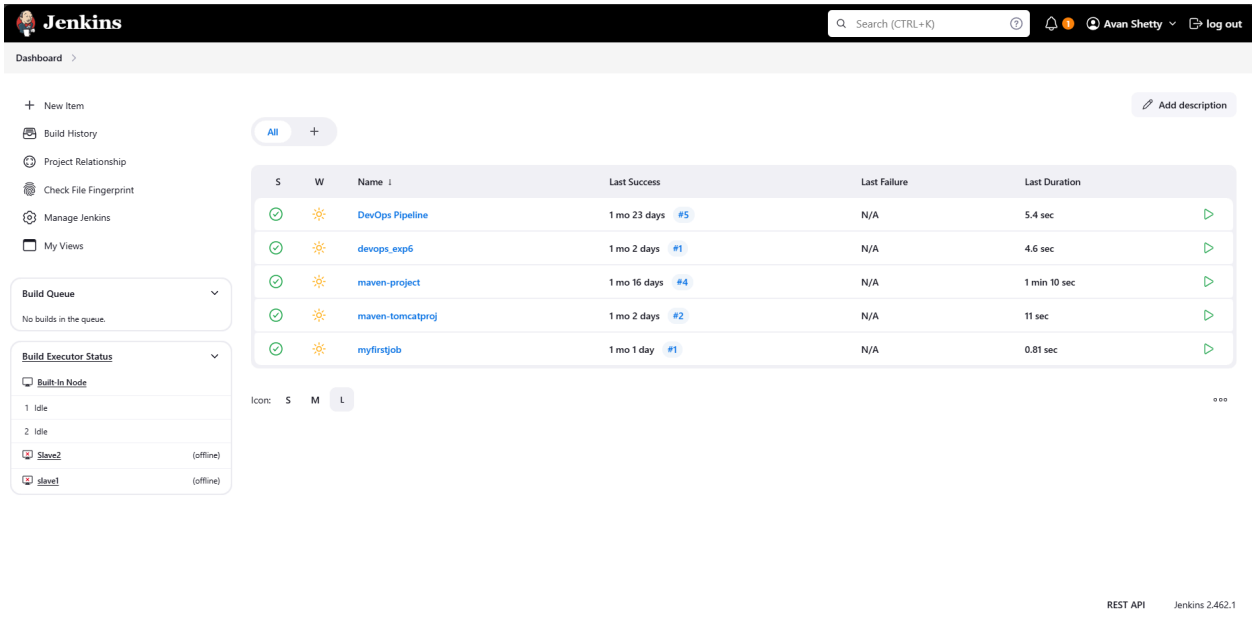
Conclusion : In this way the docker and terraform commands were learned thoroughly and how the images are a class of the container(instance) through which the production can happen and the applications can be deployed without any errors.

Experiment 7:

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

Prerequisites:
- Jenkins installed (Java JDK required)
- Docker Installed (for SonarQube)

Steps to integrate Jenkins with SonarQube
1. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.



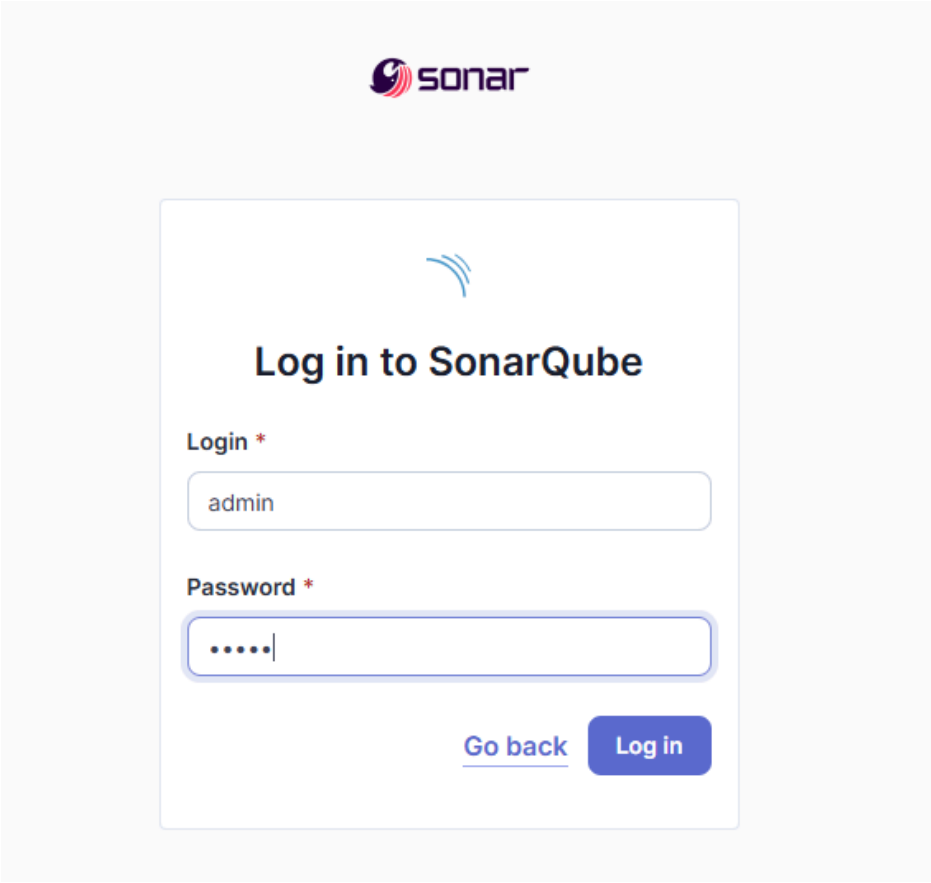
SonarQube installed successfully

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

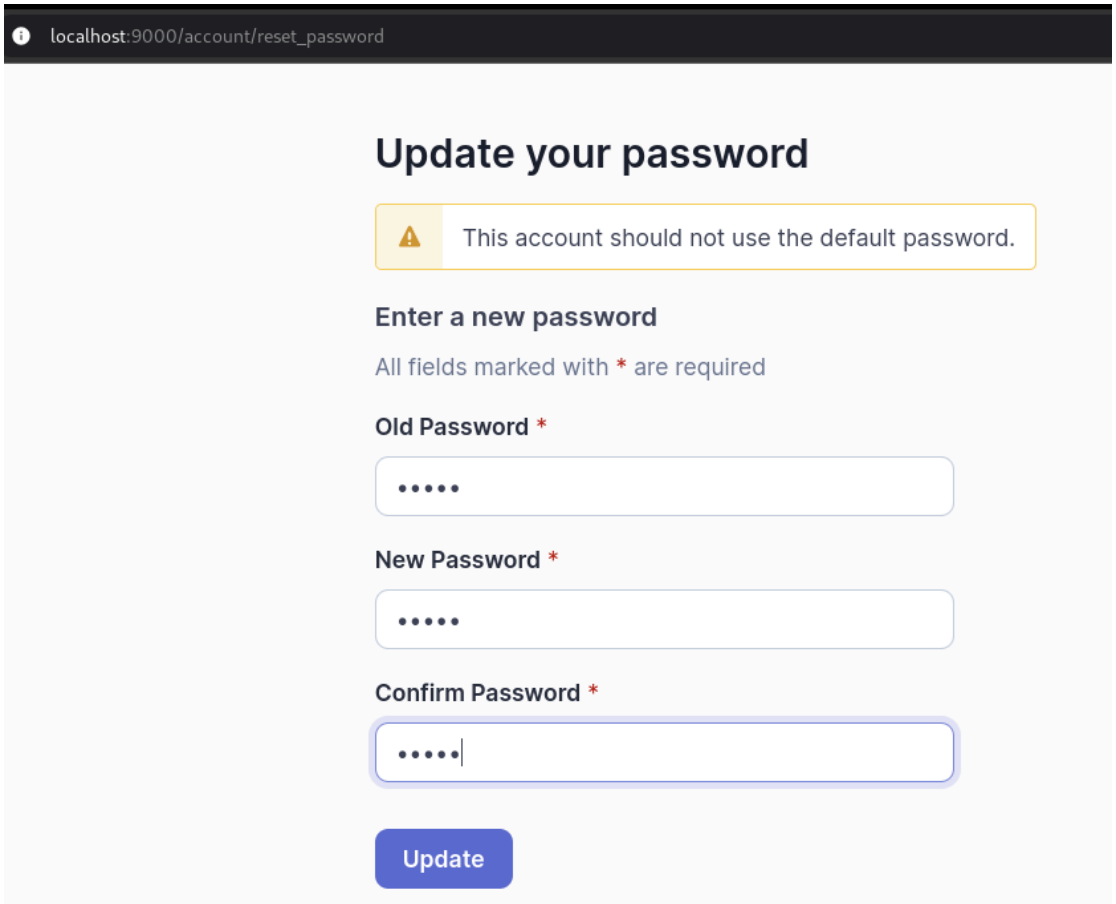
```
PS C:\Users\Avan> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube

7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
422f06f09c5a01e27ae36bc3b1cd882d1f6cc22de12345703a83b7c4beb20cf7
PS C:\Users\Avan> docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED          STATUS          PORTS                    NAM
ES
422f06f09c5a   sonarqube:latest   "/opt/sonarqube/dock…"   41 seconds ago   Up 38 seconds   0.0.0.0:9000->9000/tcp   son
arqube
PS C:\Users\Avan>
```

Our jenkins is running on port 8080

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



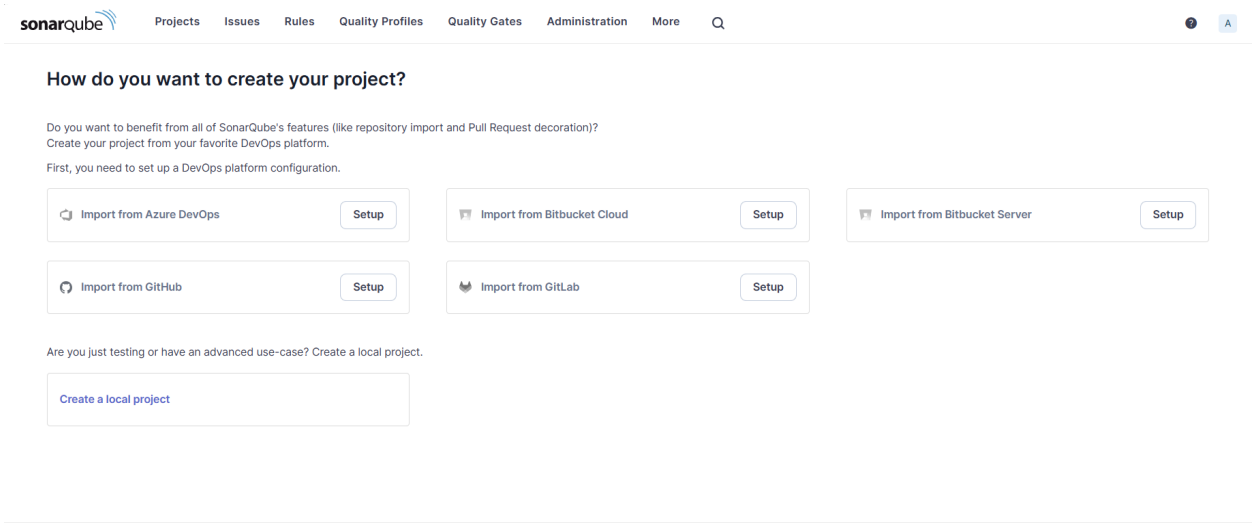
4. Login to SonarQube using username admin and password admin.

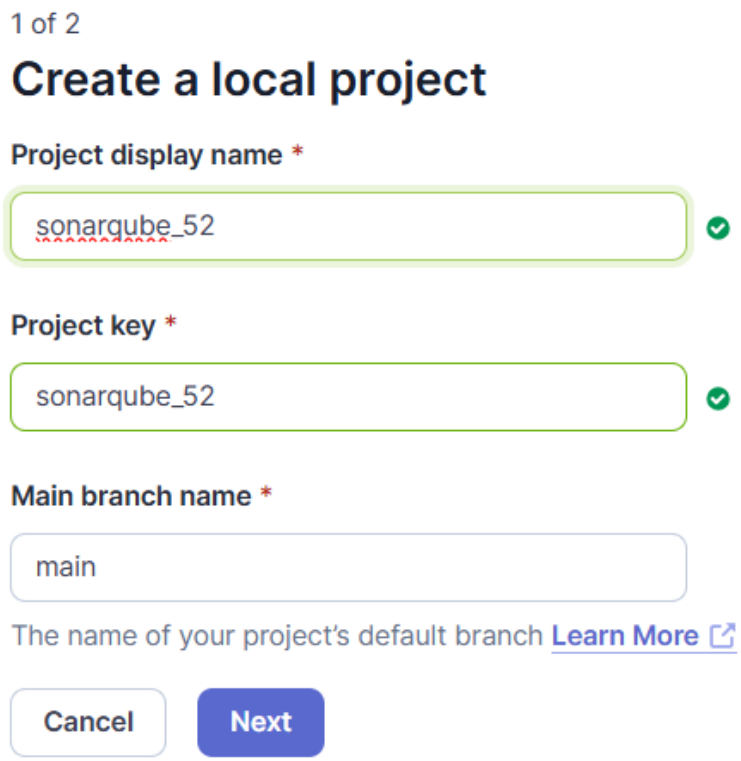


After logging, we have to change default password

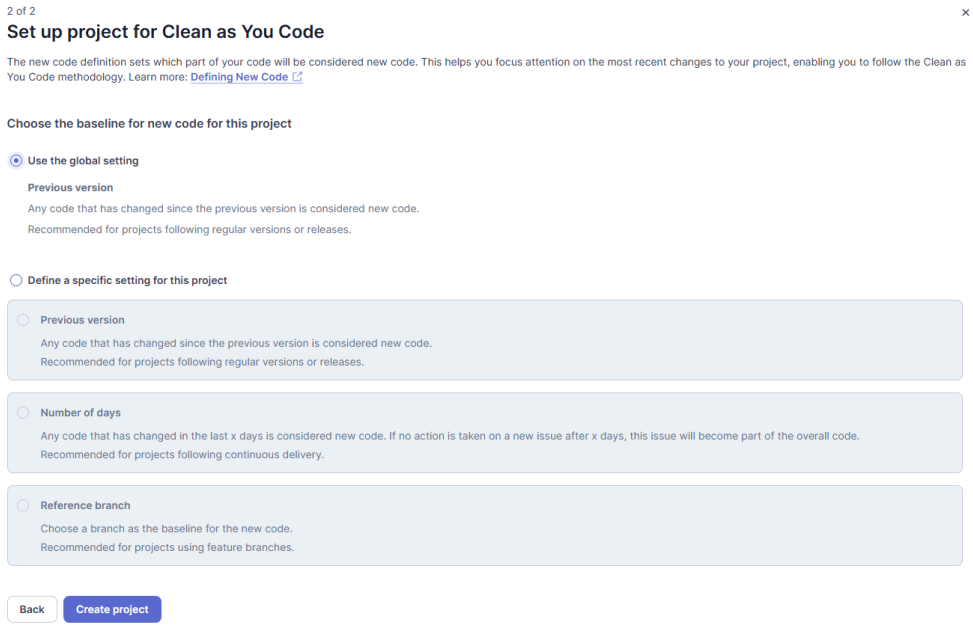
5. Create a manual project in SonarQube with the name sonarqube

Click on **create a local project** on dashboard

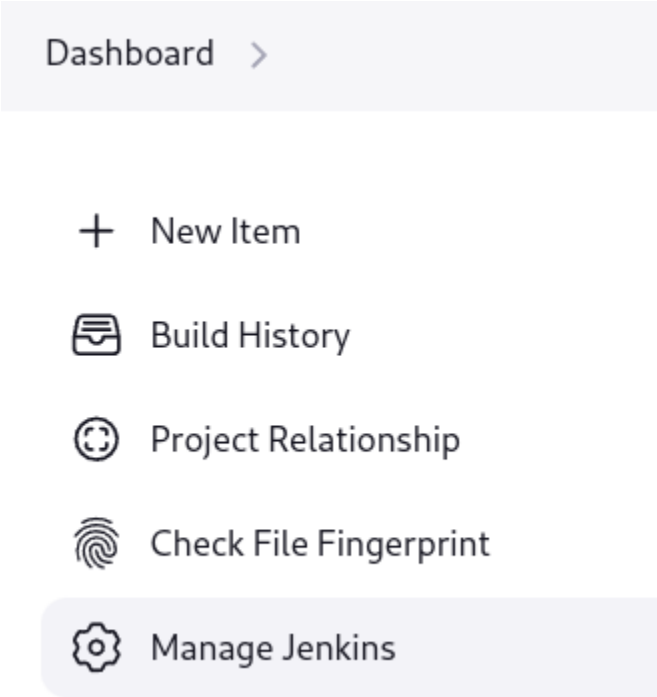




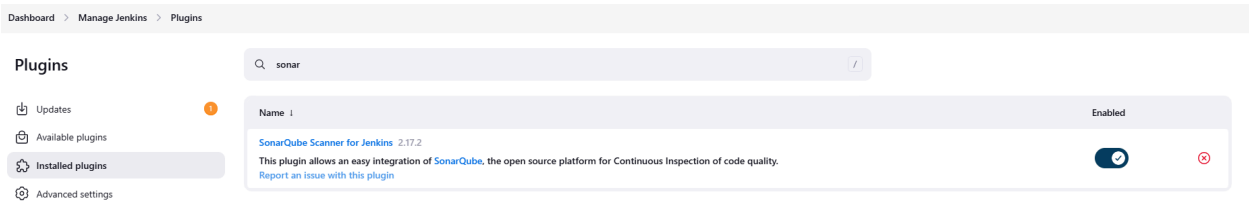
Set the baseline as global setting so that you don't need to make changes continuously every time



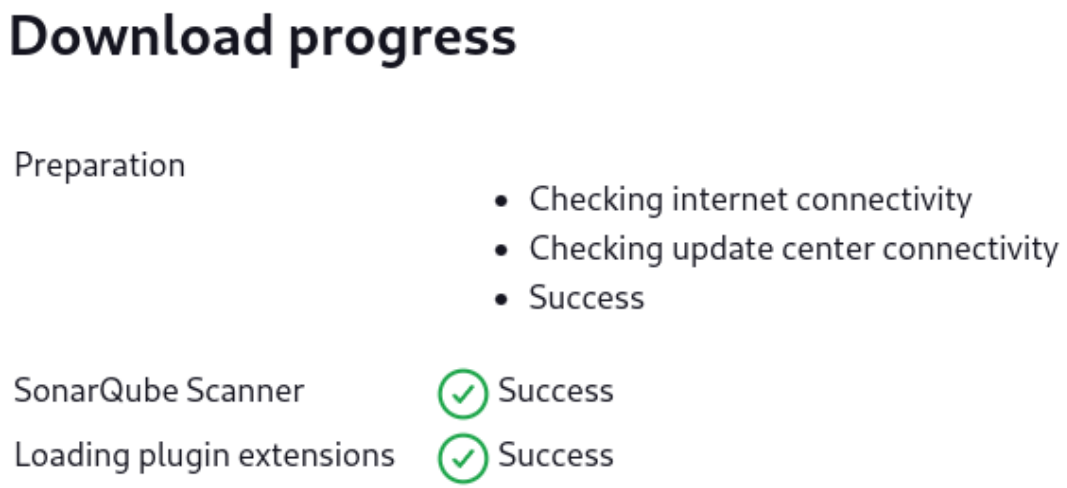
6. After setting project in sonarqube, go to **Jenkins Dashboard**



7. Go to Manage Jenkins and search for SonarQube Scanner in Plugins settings and install it.

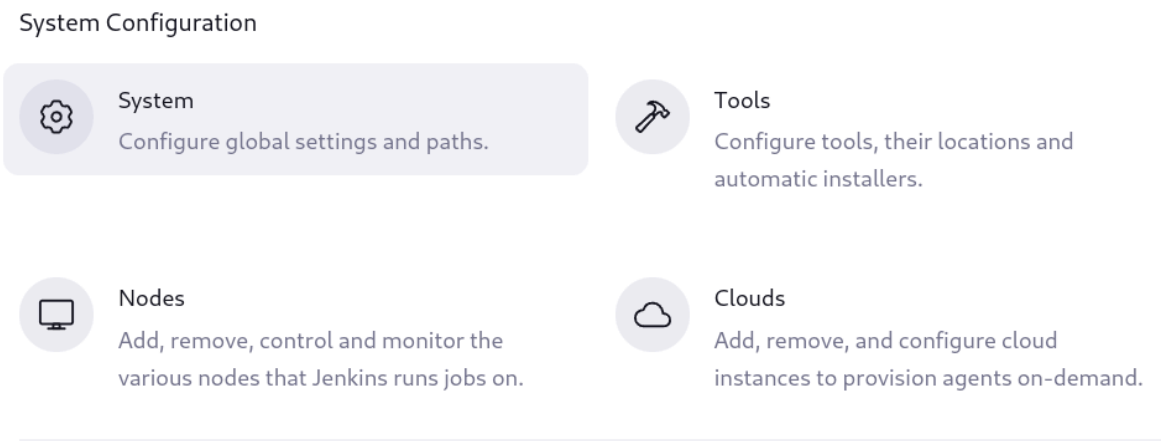


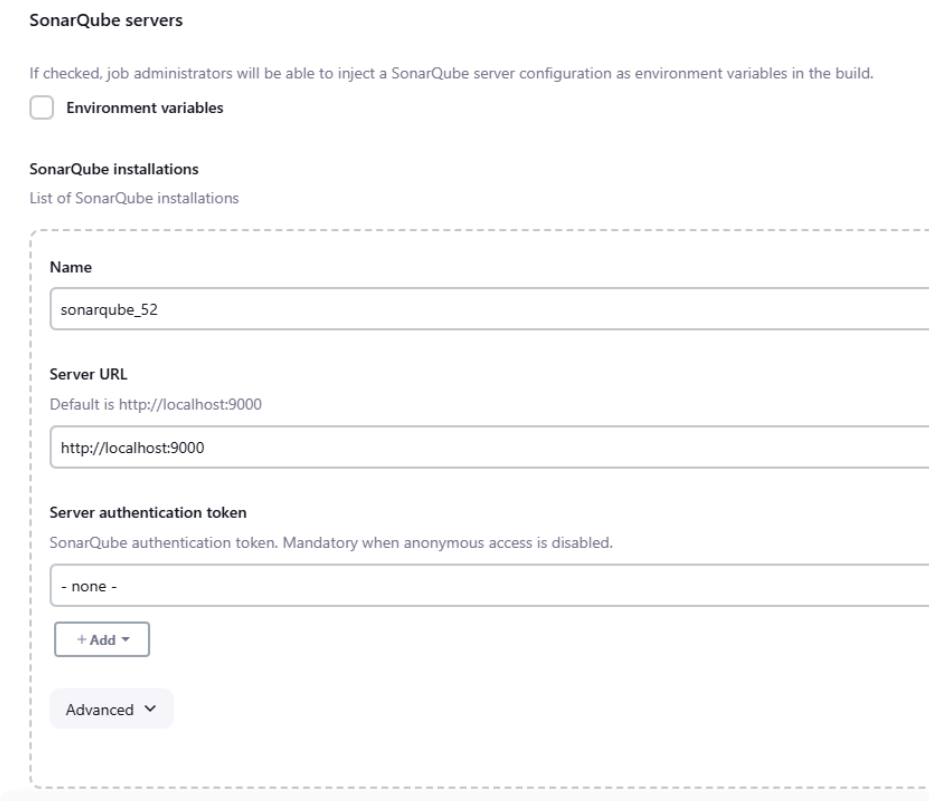
Our installation is in progress wait for it to download and install packages



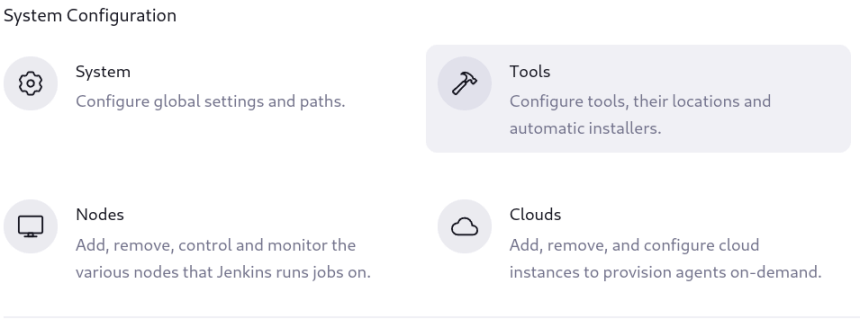
Plugin installed successfully

8. Under Jenkins dashboard 'Configure System', look for SonarQube Servers and enter the details.





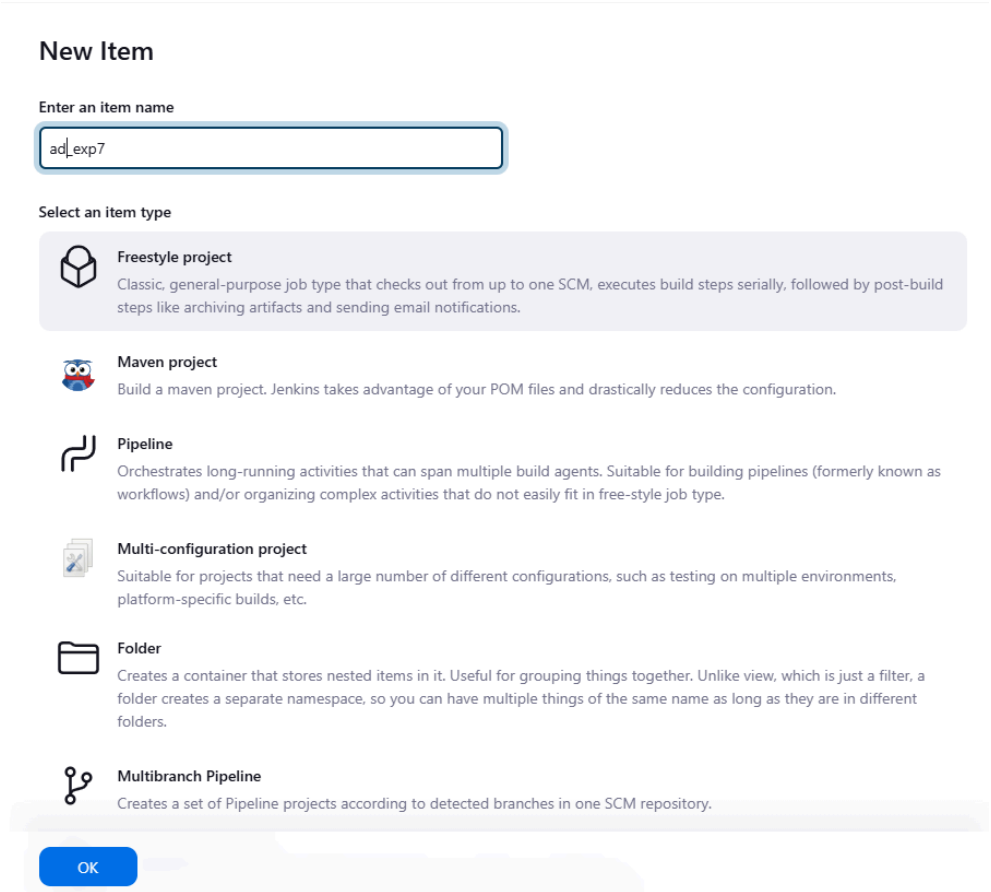
9. Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.



10. Click on **Add SonarQube Scanner**

Select Latest version and save configuration

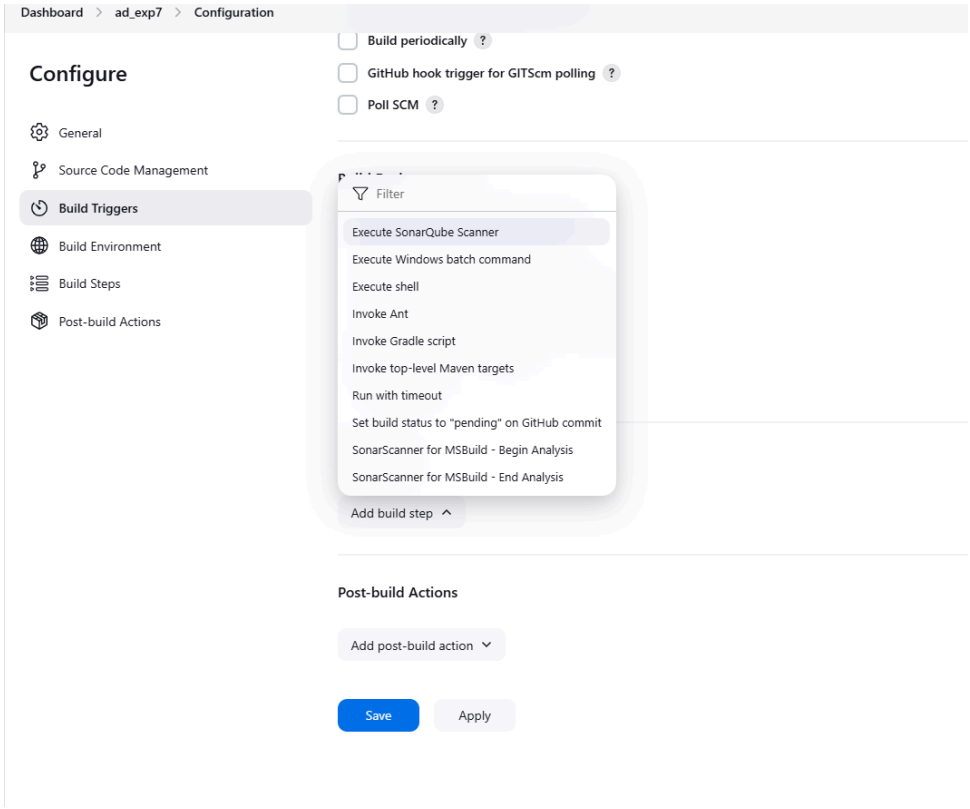
11. After the configuration, create a New Item in Jenkins, choose a freestyle project.



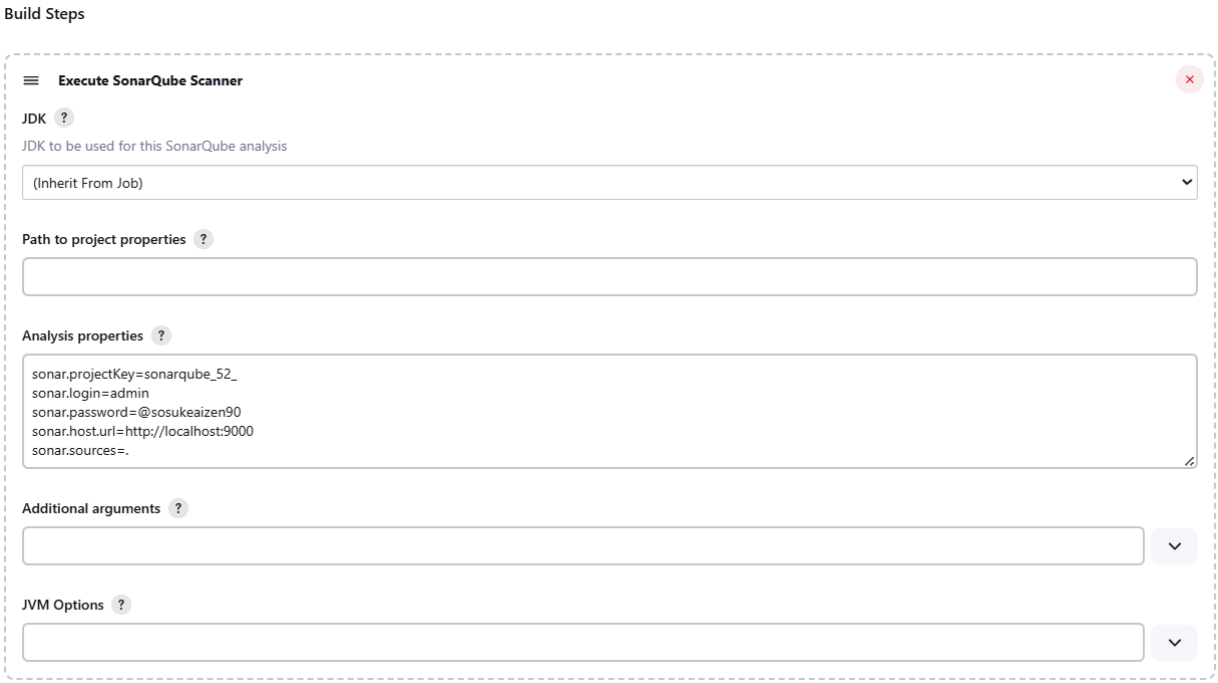
12. Choose this GitHub repository in Source Code Management. https://github.com/shazforiot/MSBuild_firstproject.git It is a sample hello-world project with no vulnerabilities and issues, just to test the integration

13. Under Build-> Execute SonarQube Scanner ->Click on add build steps

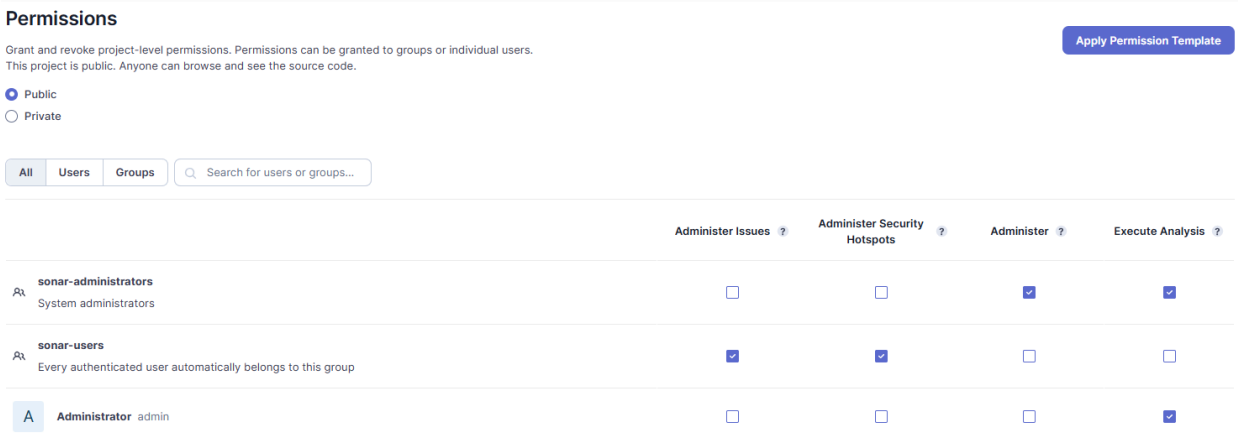
Then click on **Execute SonarQube Scanner**



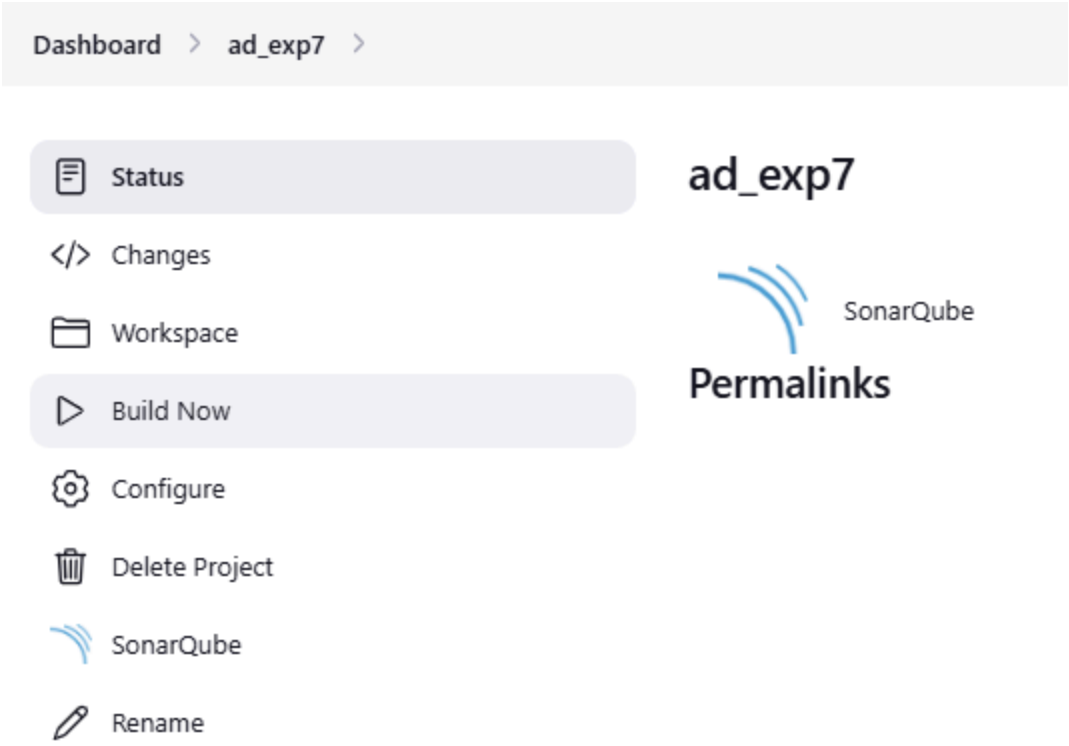
14. Mention the SonarQube Project Key, Login, Password, Source path and Host URL in Analysis properties



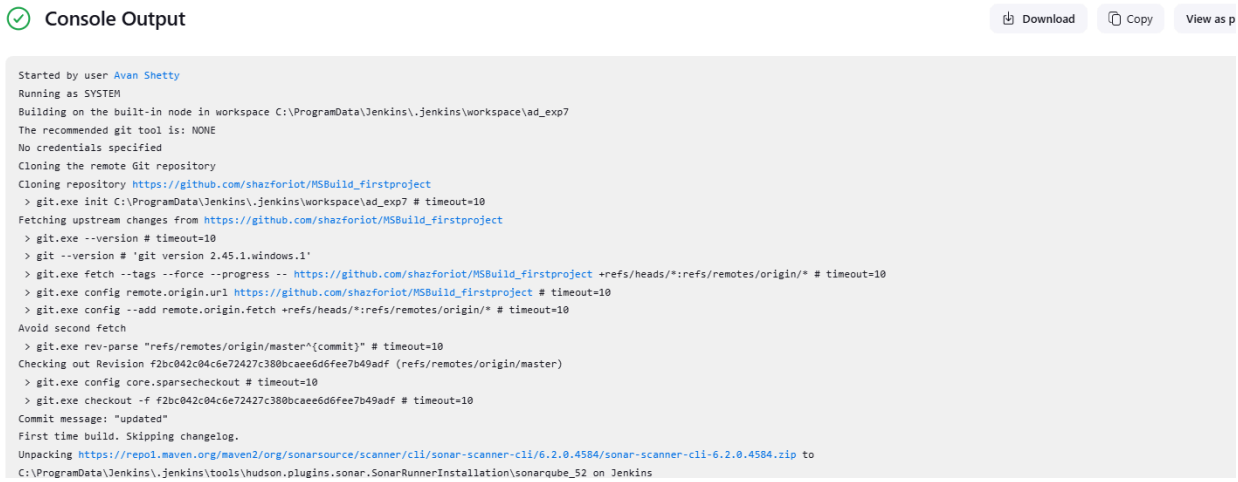
15. Go to http://localhost:9000/project_roles?id=<project_key> and allow Execute Permissions to the Admin user.



Run The Build.



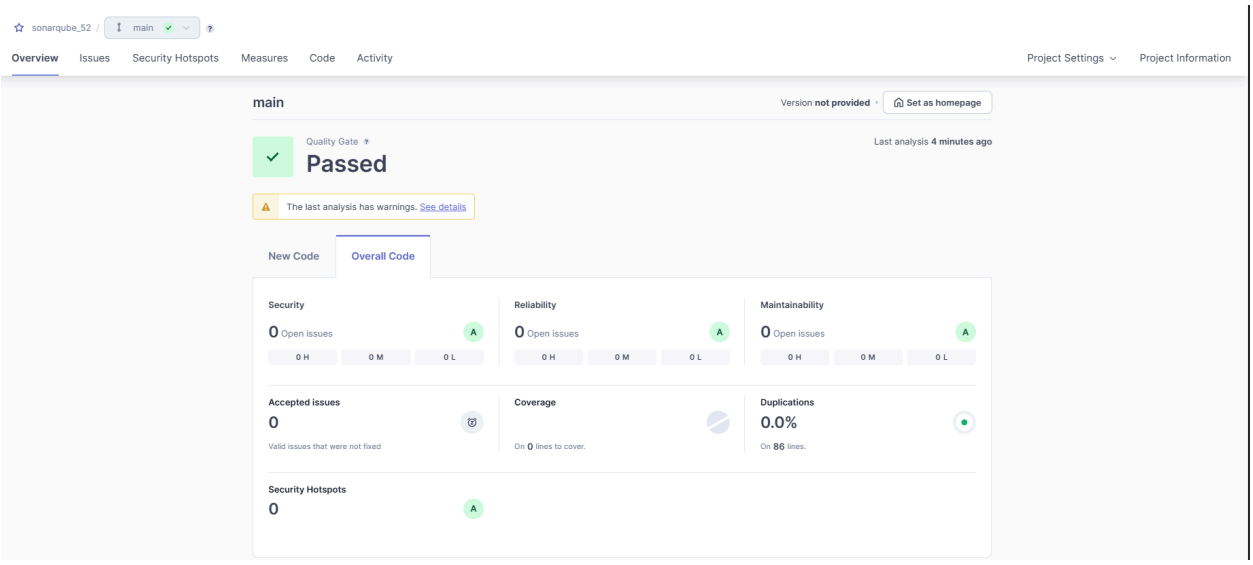
16. Check the console output



In the end if you view as success then the build is successful

```
23:17:56.019 WARN  Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir' property.
23:17:56.019 INFO  Sensor C# File Caching Sensor [csharp] (done) | time=1ms
23:17:56.019 INFO  Sensor Zero Coverage Sensor
23:17:56.042 INFO  Sensor Zero Coverage Sensor (done) | time=23ms
23:17:56.046 INFO  SCM Publisher SCM provider for this project is: git
23:17:56.049 INFO  SCM Publisher 4 source files to be analyzed
23:17:56.712 INFO  SCM Publisher 4/4 source files have been analyzed (done) | time=664ms
23:17:56.717 INFO  CPD Executor Calculating CPD for 0 files
23:17:56.718 INFO  CPD Executor CPD calculation finished (done) | time=2ms
23:17:56.730 INFO  SCM revision ID 'f2bc042c04c6e72427c380bcaee6d6fee7b49adf'
23:17:57.120 INFO  Analysis report generated in 155ms, dir size=201.0 kB
23:17:57.199 INFO  Analysis report compressed in 66ms, zip size=22.4 kB
23:17:58.420 INFO  Analysis report uploaded in 1218ms
23:17:58.423 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube_52
23:17:58.423 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
23:17:58.424 INFO  More about the report processing at http://localhost:9000/api/ce/task?id=37f427d8-0f91-42ba-930a-88b5ac66894c
23:17:58.451 INFO  Analysis total time: 58.656 s
23:17:58.456 INFO  SonarScanner Engine completed successfully
23:17:58.601 INFO  EXECUTION SUCCESS
23:17:58.707 INFO  Total time: 1:31.437s
Finished: SUCCESS
```

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



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

Conclusion:

During the experiment, I successfully set up SonarQube using Docker and created a new project in SonarQube. I integrated it with Jenkins by creating a freestyle project linked to a Git repository for code analysis. After configuring Jenkins with the correct settings and granting necessary permissions, the project ran successfully, with all tests passing in SonarQube.However, there were some challenges, such as ensuring precise SonarQube commands and entering correct credentials. Security access needed to be configured properly, or the build would fail. Additionally, it was crucial to have the JDK installed and the path correctly set in the environment variables.

Experiment 8:

Aim: Create a Jenkins CI/CD 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.

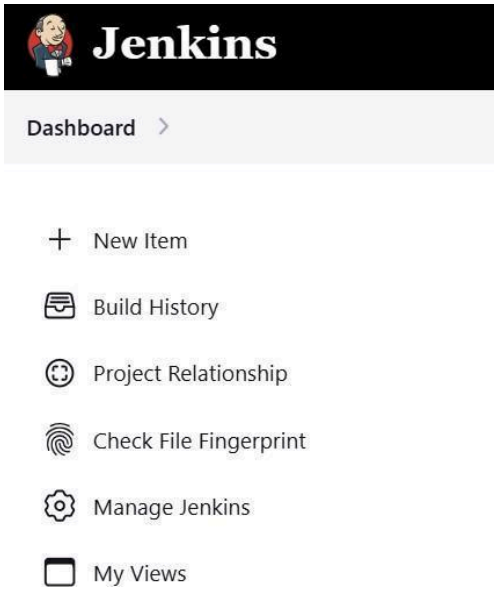
Integrating Jenkins with SonarQube:

Prerequisites:

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

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

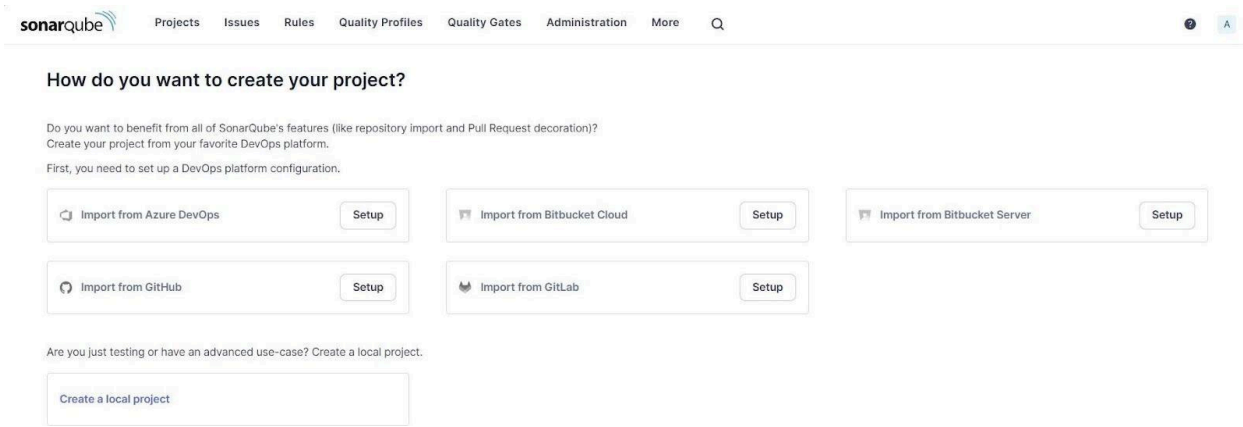
1. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.



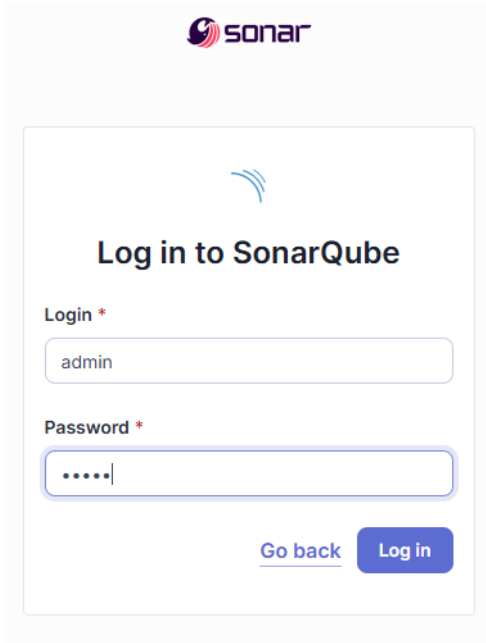
2. Run SonarQube in a Docker container using this command

```
PS C:\Users\Avan> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube

7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
422f06f09c5a01e27ae36bc3b1cd882d1f6cc22de12345703a83b7c4beb20cf7
PS C:\Users\Avan> docker ps
CONTAINER ID   IMAGE             COMMAND                  CREATED          STATUS          PORTS                    NAM
ES
422f06f09c5a   sonarqube:latest   "/opt/sonarqube/dock…"   41 seconds ago   Up 38 seconds   0.0.0.0:9000->9000/tcp   son
arqube
PS C:\Users\Avan>
```

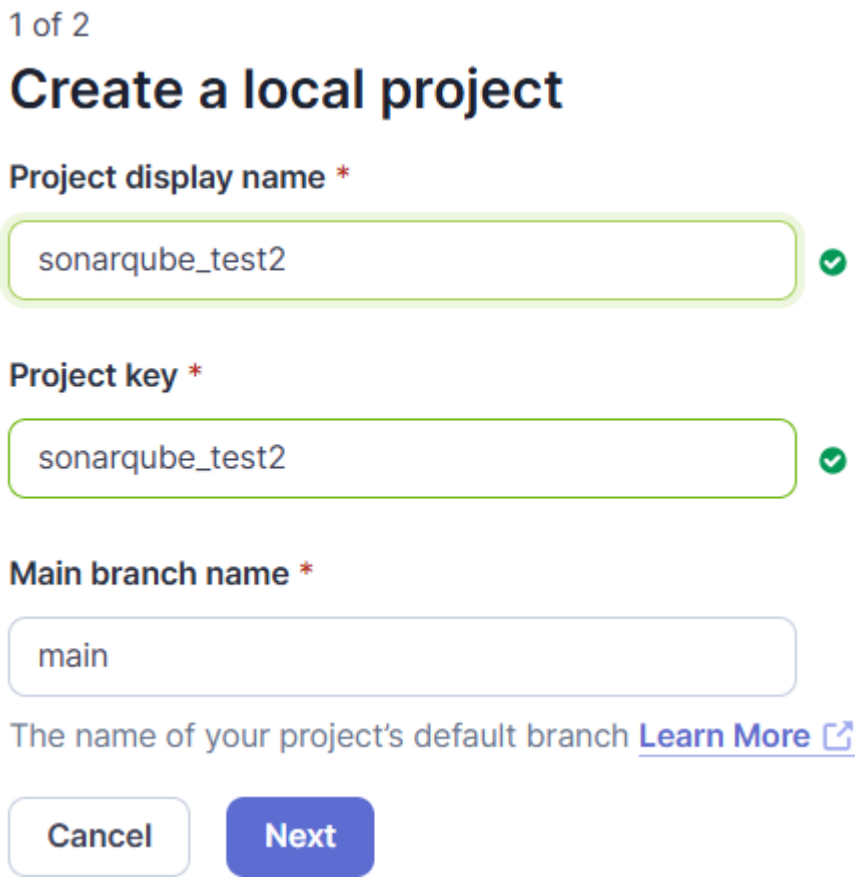
3. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.



4. Login to SonarQube using your username and password

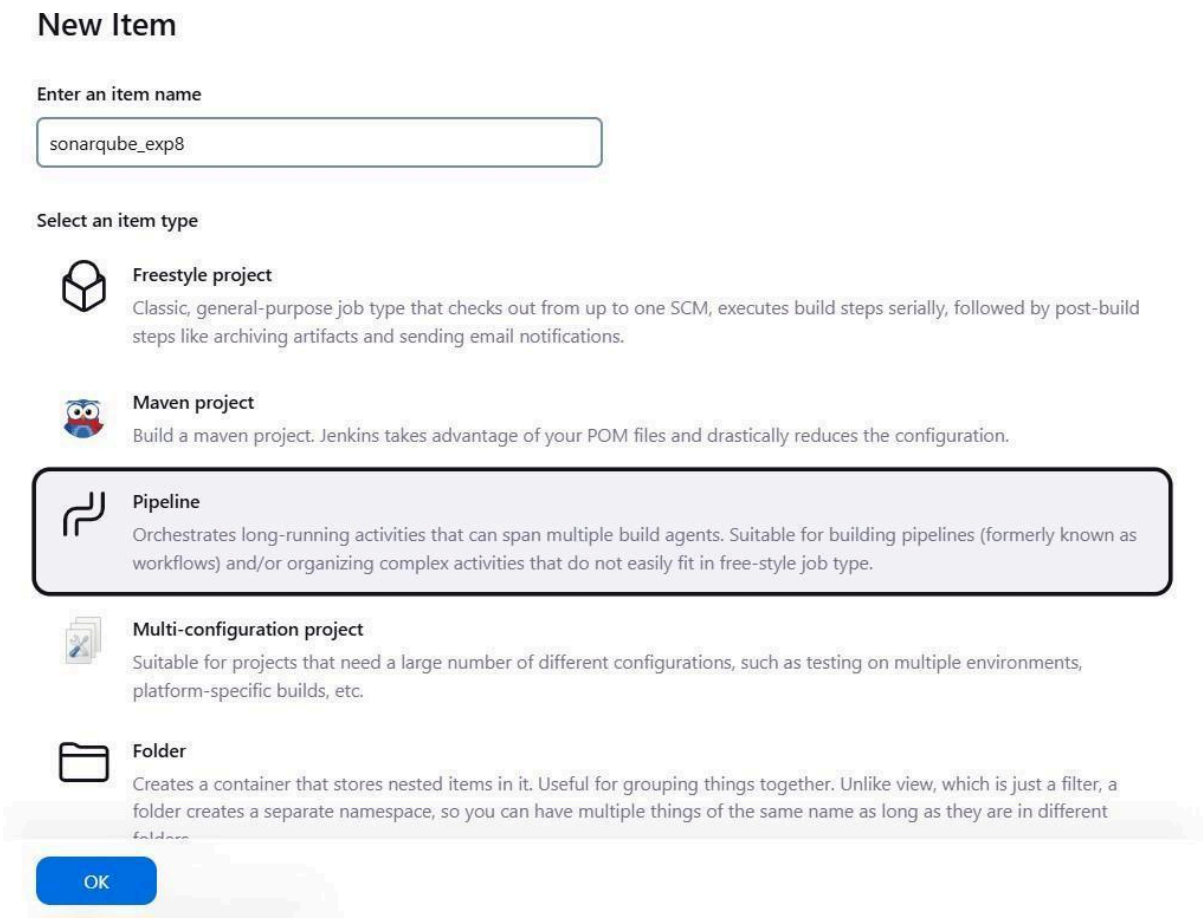


5. Create a manual project in SonarQube with the name sonarqube-test2



Setup the project and come back to Jenkins Dashboard.

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



7. Under Pipeline Script, enter the following -

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

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

SonarQube.

8. Install sonar-scanner

Now we need to install **sonar-scanner** binary to perform code analysis
To do so, go to
(https://binaries.sonarsource.com/Distribution/sonar-scanner-cli/sonar-scanner-cli-6.2.0.4584-windows-x64.zip for windows OS)
For windows path would be
C:\\Users\<USER_NAME>\Downloads\sonar-scanner-cli-6.2.0.4584-windows-x64\bin\sonar-s c anner
Now we need to update the required details in the pipeline script as :

```
node {
   stage('Cloning the GitHub Repo') {
      git 'https://github.com/shazforiot/GOL.git'
   }
   stage('SonarQube analysis') {
      withSonarQubeEnv('sonarqube_52')
      {
         bat """

C:\\Users\\Avan\\Downloads\\sonar-scanner-cli-6.2.0.4584-windows-x64\\sonar-scanner-6.2.0.4
5 84-windows-x64\\bin\\sonar-scanner.bat ^
         -Dsonar.login=admin ^
         -Dsonar.password=@sosukeaizen90 ^
         -Dsonar.projectKey=sonarqube_test2 ^
         -Dsonar.exclusions=vendor/*,resources/**,**/*.java ^
         -Dsonar.host.url=http://127.0.0.1:9000/
         """
      }
   }
}
```

Definition

Pipeline script

Script

```
node {
    stage('Cloning the GitHub Repo') {
        git 'https://github.com/shazforiot/GOL.git'
    }
    stage('SonarQube analysis') {
        withSonarQubeEnv('sonarqube_52') {
            bat """
            C:\\Users\\Avan\\Downloads\\sonar-scanner-cli-6.2.0.4584-windows-x64\\sonar-scanner-6.2.0.4584-windows-x64\\bin\\sonar-scanner.bat ^
            -Dsonar.login=admin ^
            -Dsonar.password=@sosukeaizen90 ^
            -Dsonar.projectKey=sonarqube_test2 ^
            -Dsonar.exclusions=vendor/*,resources/**,**/*.java ^
            -Dsonar.host.url=http://127.0.0.1:9000/
            """
        }
    }
}
```

Use Groovy Sandbox

Pipeline Syntax

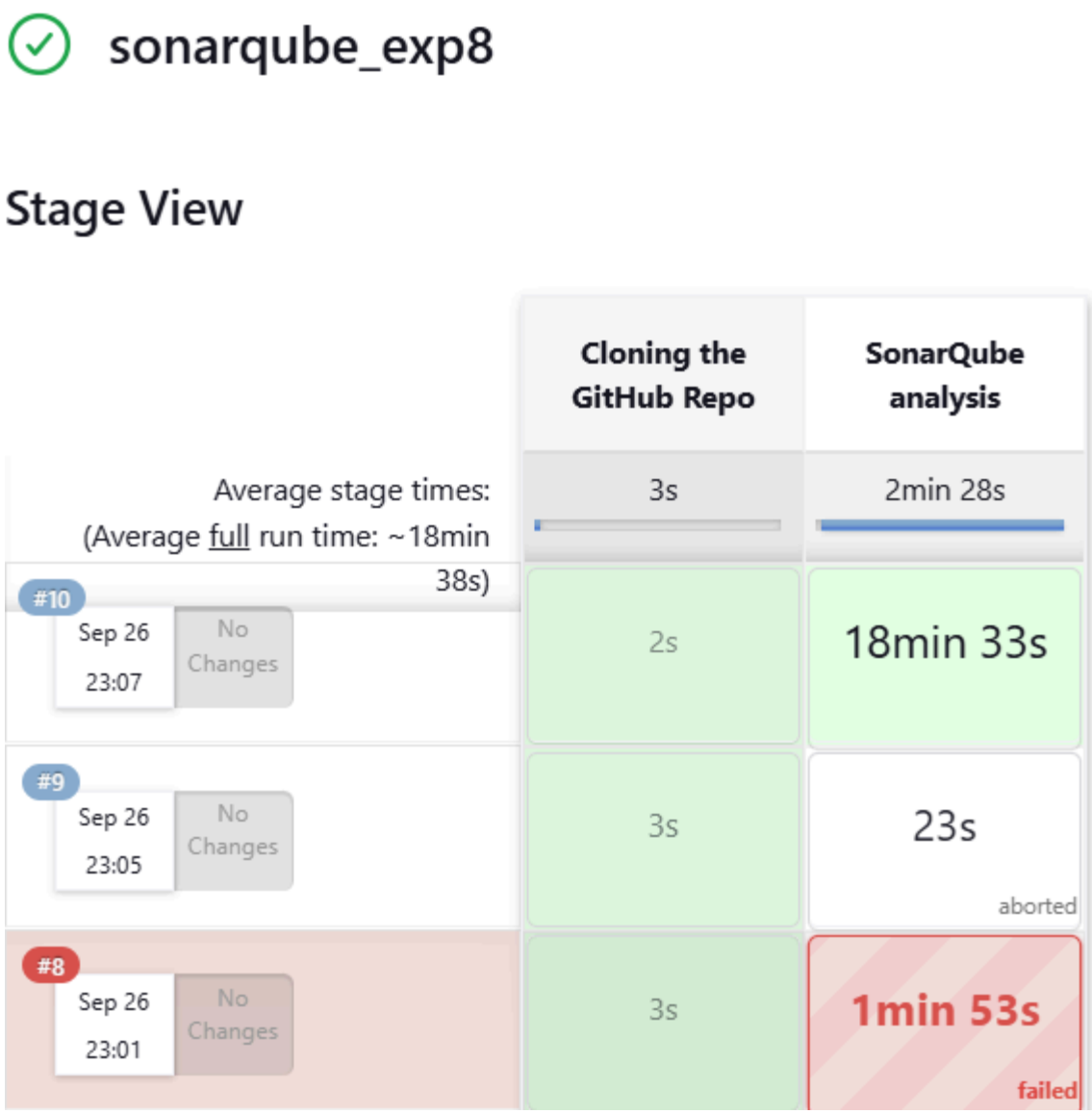
Save Apply

9. Run the build

Dashboard > SonarQube-exp8 >

- Status
- Changes
- Build Now
- Configure
- Delete Pipeline
- Full Stage View
- Stages
- Rename
- Pipeline Syntax

Check the status of Build



As we can see the SonarQube analysis is completed

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

Console Output

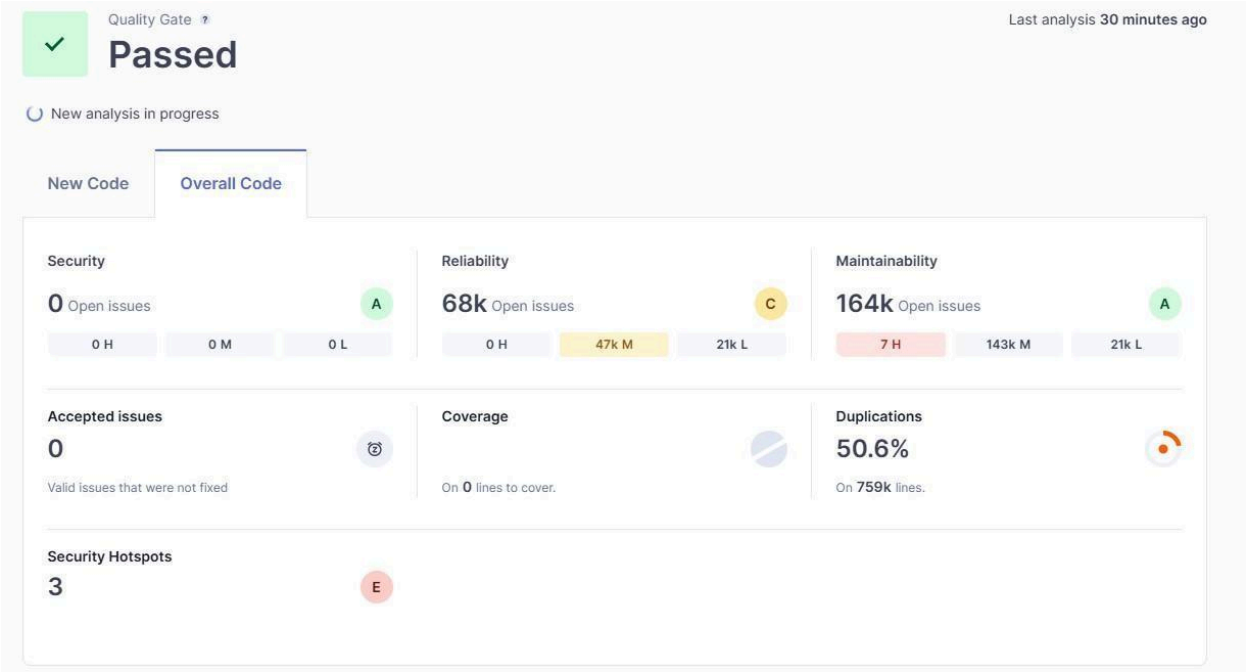
Download Copy View as plain text

Skipping 4,250 KB.. Full Log

```
23:21:54.344 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/AssertionVisualizer.html for block at line
870. Keep only the first 100 references.
23:21:54.344 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/AssertionVisualizer.html for block at line
480. Keep only the first 100 references.
23:21:54.344 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/AssertionVisualizer.html for block at line
760. Keep only the first 100 references.
23:21:54.344 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/AssertionVisualizer.html for block at line
480. Keep only the first 100 references.
23:21:54.344 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/AssertionVisualizer.html for block at line
496. Keep only the first 100 references.
23:21:54.344 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/AssertionVisualizer.html for block at line
481. Keep only the first 100 references.
23:21:54.344 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/AssertionVisualizer.html for block at line
527. Keep only the first 100 references.
23:21:54.344 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/AssertionVisualizer.html for block at line
794. Keep only the first 100 references.
23:21:54.344 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/AssertionVisualizer.html for block at line
496. Keep only the first 100 references.
23:21:54.345 WARN  Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/AssertionVisualizer.html for block at line
773. Keep only the first 100 references.
```

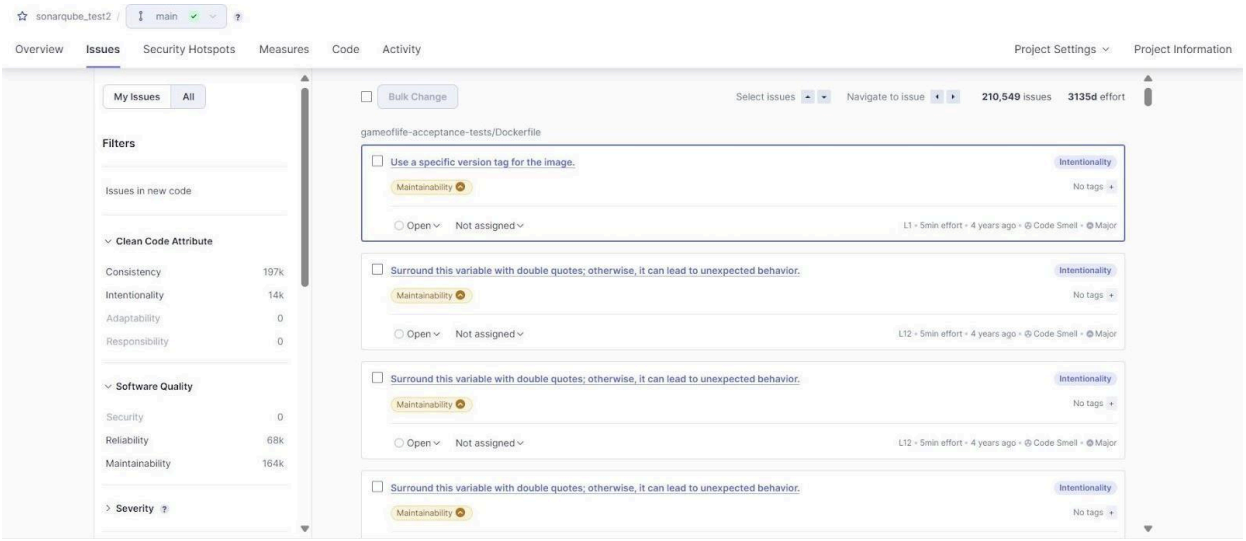
```
Keep only the first 100 references.
23:21:59.444 INFO  CPD Executor CPD calculation finished (done) | time=208061ms
23:22:00.219 INFO  SCM revision ID 'ba799ba7e1b576f04a4612322b0412c5e6e1e5e4'
23:24:27.580 INFO  Analysis report generated in 9097ms, dir size=127.2 MB
23:24:53.548 INFO  Analysis report compressed in 25938ms, zip size=29.6 MB
23:25:07.200 INFO  Analysis report uploaded in 13534ms
23:25:07.366 INFO  ANALYSIS SUCCESSFUL, you can find the results at: http://127.0.0.1:9000/dashboard?id=sonarqube_test2
23:25:07.367 INFO  Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
23:25:07.369 INFO  More about the report processing at http://127.0.0.1:9000/api/ce/task?id=06b00655-446a-4b9c-93cd-ae0f4bac4715
23:26:06.547 INFO  Analysis total time: 18:11.651 s
23:26:06.671 INFO  SonarScanner Engine completed successfully
23:26:07.988 INFO  EXECUTION SUCCESS
23:26:08.197 INFO  Total time: 18:21.790s
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

11. After that, check the project in SonarQube.

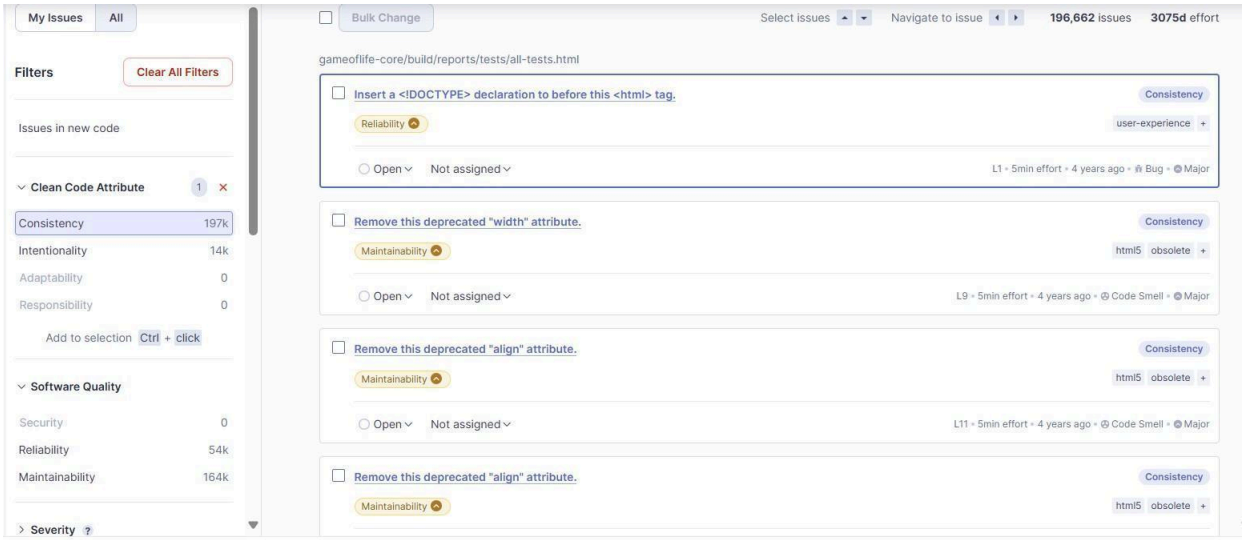


Under different tabs, check all different issues with the code

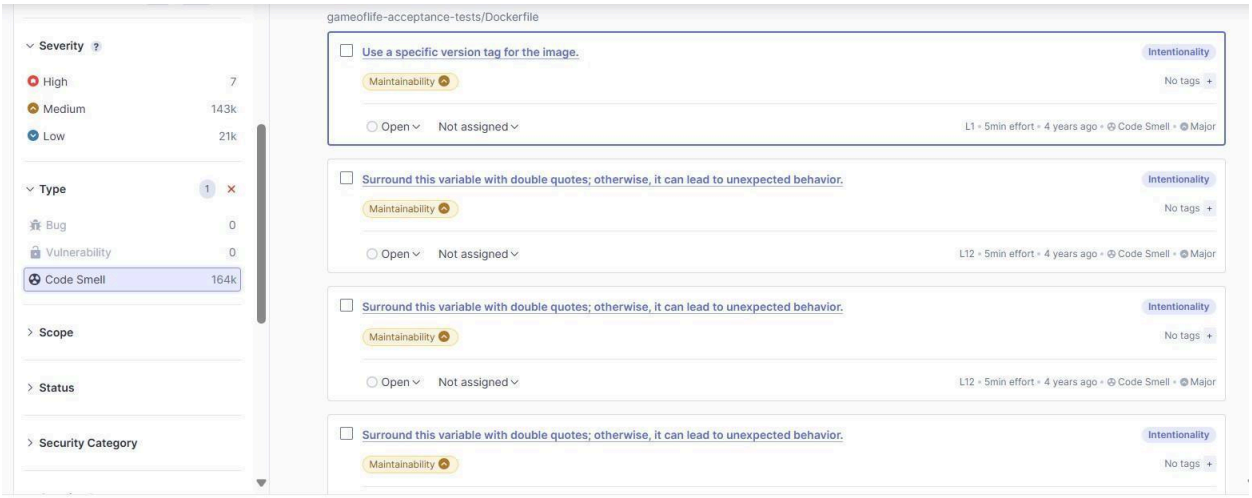
12. **Code Problems-**



Code Smells



Bugs



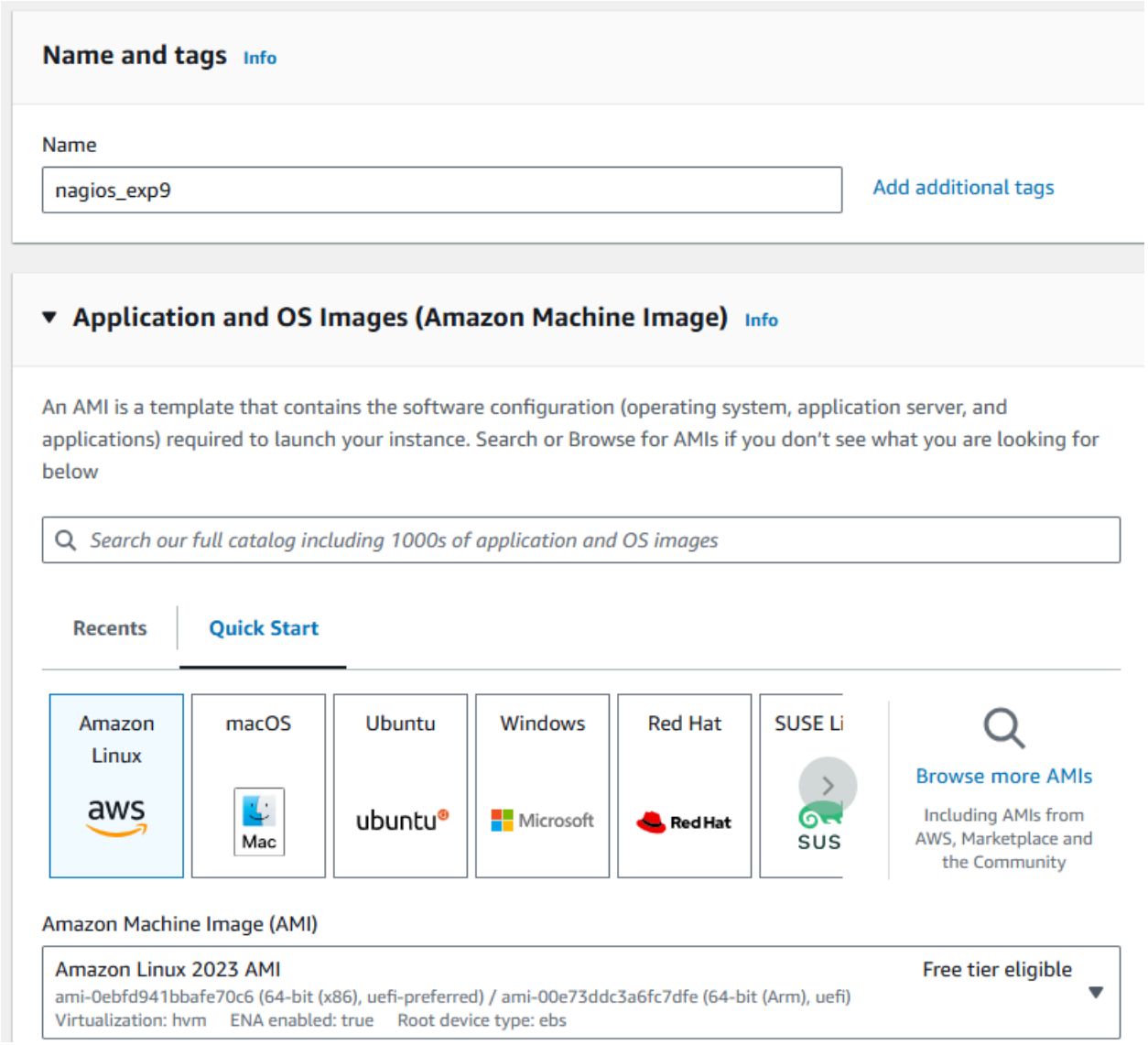
Conclusion: We began the experiment by creating a new project in SonarQube and setting up a new pipeline in Jenkins with the proper configuration of the pipeline script. Then we installed the Sonar Scanner CLI, enabling Jenkins to perform code analysis on the Git repository. The pipeline can also be configured to use the installed Sonar Scanner plugin instead of the locally installed binary. Eventually, the pipeline ran successfully, with all tests passing in SonarQube.However, it took me 10 attempts in the Jenkins console to get the build right. The script had to be written in a particular format, ensuring that the credentials were added correctly without any errors. Additionally, SonarQube takes a little time to pass the tests and complete the analysis.

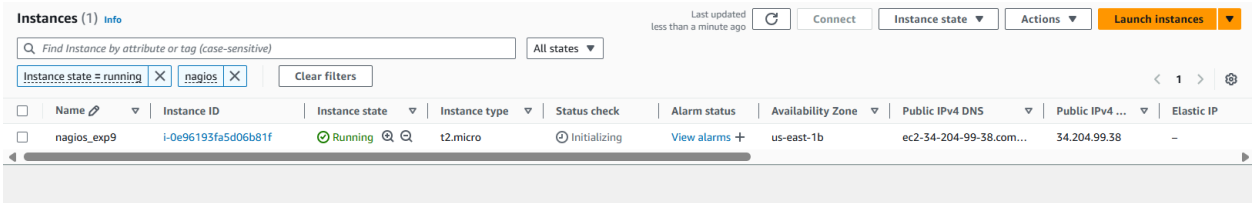
Experiment 9:

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

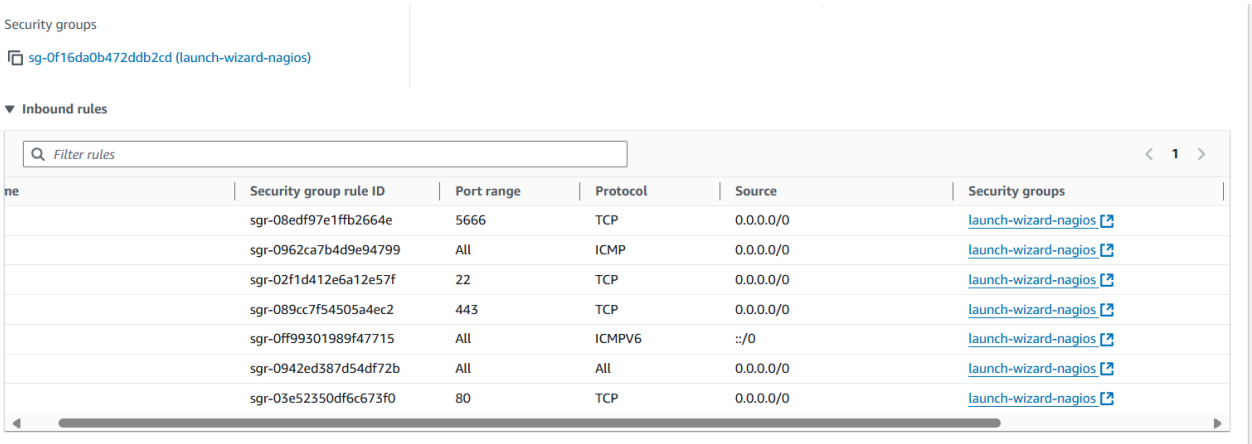
Steps to install and configure Nagios in EC2 instance

1. Create an Amazon Linux EC-2 instance and select either existing key pair or create new

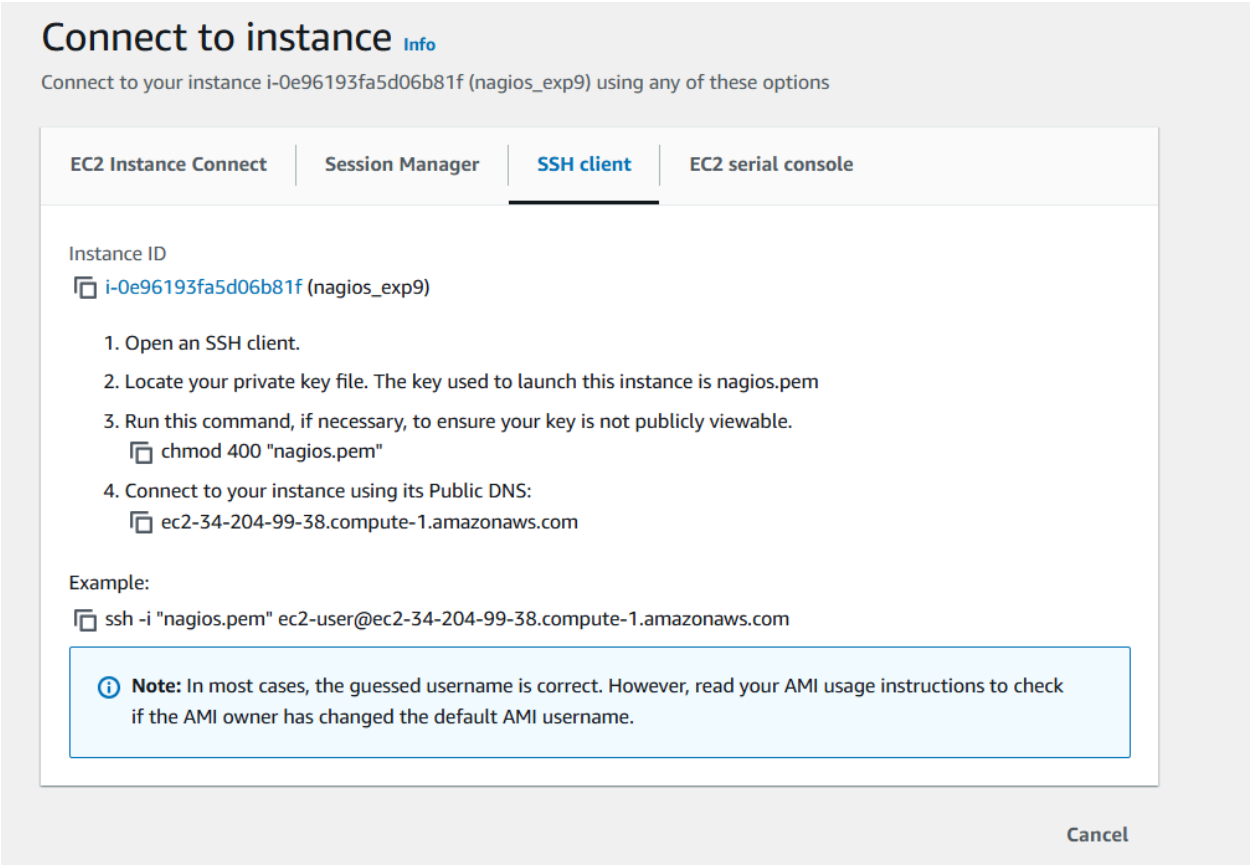




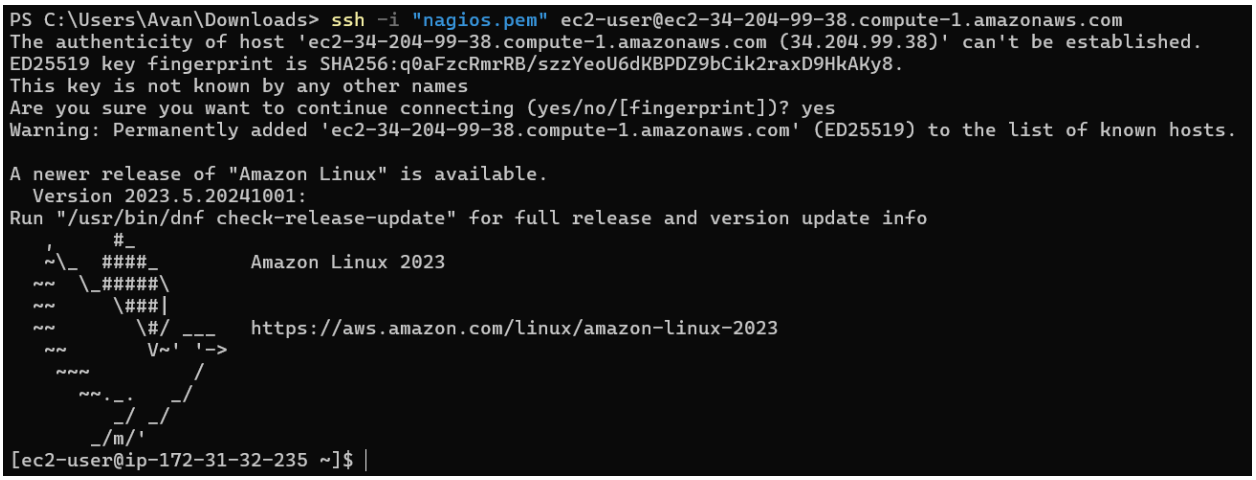
2. Under Security Group, make sure HTTP, HTTPS, SSH, ICMP are open from everywhere.



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



Connection Successfully installed



4. Now, we need to install necessary packages that are required to run nagios properly

sudo yum install httpd php

```
[ec2-user@ip-172-31-32-235 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:02:20 ago on Thu Oct  3 06:48:23 2024.
Dependencies resolved.
================================================================================================================================
 Package                        Architecture       Version                                  Repository                 Size
================================================================================================================================
Installing:
 httpd                          x86_64             2.4.62-1.amzn2023                        amazonlinux                48 k
 php8.3                         x86_64             8.3.10-1.amzn2023.0.1                    amazonlinux                10 k
Installing dependencies:
 apr                            x86_64             1.7.2-2.amzn2023.0.2                     amazonlinux               129 k
 apr-util                       x86_64             1.6.3-1.amzn2023.0.1                     amazonlinux                98 k
 generic-logos-httpd            noarch             18.0.0-12.amzn2023.0.3                   amazonlinux                19 k
 httpd-core                     x86_64             2.4.62-1.amzn2023                        amazonlinux               1.4 M
 httpd-filesystem               noarch             2.4.62-1.amzn2023                        amazonlinux                14 k
 httpd-tools                    x86_64             2.4.62-1.amzn2023                        amazonlinux                81 k
 libbrotli                      x86_64             1.0.9-4.amzn2023.0.2                     amazonlinux               315 k
 libsodium                      x86_64             1.0.19-4.amzn2023                        amazonlinux               176 k
 libxslt                        x86_64             1.1.34-5.amzn2023.0.2                    amazonlinux               241 k
 mailcap                        noarch             2.1.49-3.amzn2023.0.3                    amazonlinux                33 k
 nginx-filesystem               noarch             1:1.24.0-1.amzn2023.0.4                  amazonlinux               9.8 k
 php8.3-cli                     x86_64             8.3.10-1.amzn2023.0.1                    amazonlinux               3.7 M
 php8.3-common                  x86_64             8.3.10-1.amzn2023.0.1                    amazonlinux               737 k
 php8.3-process                 x86_64             8.3.10-1.amzn2023.0.1                    amazonlinux                45 k
 php8.3-xml                     x86_64             8.3.10-1.amzn2023.0.1                    amazonlinux               154 k
```

```
  Available Versions:

  Version 2023.5.20241001:
    Run the following command to upgrade to 2023.5.20241001:

      dnf upgrade --releasever=2023.5.20241001

    Release notes:
     https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.5.20241001.html

================================================================================================================================

Installed:
  apr-1.7.2-2.amzn2023.0.2.x86_64                          apr-util-1.6.3-1.amzn2023.0.1.x86_64
  apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64             generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch
  httpd-2.4.62-1.amzn2023.x86_64                           httpd-core-2.4.62-1.amzn2023.x86_64
  httpd-filesystem-2.4.62-1.amzn2023.noarch                httpd-tools-2.4.62-1.amzn2023.x86_64
  libbrotli-1.0.9-4.amzn2023.0.2.x86_64                    libsodium-1.0.19-4.amzn2023.x86_64
  libxslt-1.1.34-5.amzn2023.0.2.x86_64                     mailcap-2.1.49-3.amzn2023.0.3.noarch
  mod_http2-2.0.27-1.amzn2023.0.3.x86_64                   mod_lua-2.4.62-1.amzn2023.x86_64
  nginx-filesystem-1:1.24.0-1.amzn2023.0.4.noarch          php8.3-8.3.10-1.amzn2023.0.1.x86_64
  php8.3-cli-8.3.10-1.amzn2023.0.1.x86_64                  php8.3-common-8.3.10-1.amzn2023.0.1.x86_64
  php8.3-fpm-8.3.10-1.amzn2023.0.1.x86_64                  php8.3-mbstring-8.3.10-1.amzn2023.0.1.x86_64
  php8.3-opcache-8.3.10-1.amzn2023.0.1.x86_64              php8.3-pdo-8.3.10-1.amzn2023.0.1.x86_64
  php8.3-process-8.3.10-1.amzn2023.0.1.x86_64              php8.3-sodium-8.3.10-1.amzn2023.0.1.x86_64
  php8.3-xml-8.3.10-1.amzn2023.0.1.x86_64
```

sudo yum install gcc glibc glibc-common

```
[ec2-user@ip-172-31-32-235 ~]$ sudo yum install gcc glibc glibc-common
Last metadata expiration check: 0:03:23 ago on Thu Oct  3 06:48:23 2024.
Package glibc-2.34-52.amzn2023.0.11.x86_64 is already installed.
Package glibc-common-2.34-52.amzn2023.0.11.x86_64 is already installed.
Dependencies resolved.
================================================================================================================================
 Package                        Architecture       Version                                  Repository                 Size
================================================================================================================================
Installing:
 gcc                            x86_64             11.4.1-2.amzn2023.0.2                    amazonlinux                32 M
Installing dependencies:
 annobin-docs                   noarch             10.93-1.amzn2023.0.1                     amazonlinux                92 k
 annobin-plugin-gcc             x86_64             10.93-1.amzn2023.0.1                     amazonlinux               887 k
 cpp                            x86_64             11.4.1-2.amzn2023.0.2                    amazonlinux                10 M
 gc                             x86_64             8.0.4-5.amzn2023.0.2                     amazonlinux               105 k
 glibc-devel                    x86_64             2.34-52.amzn2023.0.11                    amazonlinux                27 k
 glibc-headers-x86              noarch             2.34-52.amzn2023.0.11                    amazonlinux               427 k
 guile22                        x86_64             2.2.7-2.amzn2023.0.3                     amazonlinux               6.4 M
 kernel-headers                 x86_64             6.1.109-118.189.amzn2023                 amazonlinux               1.4 M
 libmpc                         x86_64             1.2.1-2.amzn2023.0.2                     amazonlinux                62 k
 libtool-ltdl                   x86_64             2.4.7-1.amzn2023.0.3                     amazonlinux                38 k
 libxcrypt-devel                x86_64             4.4.33-7.amzn2023                        amazonlinux                32 k
 make                           x86_64             1:4.3-5.amzn2023.0.2                     amazonlinux               534 k
```

```
WARNING:
  A newer release of "Amazon Linux" is available.

  Available Versions:

  Version 2023.5.20241001:
    Run the following command to upgrade to 2023.5.20241001:

      dnf upgrade --releasever=2023.5.20241001

    Release notes:
     https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.5.20241001.html

================================================================================================

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

Complete!
```

sudo yum install gd gd-devel

```
[ec2-user@ip-172-31-32-235 ~]$ sudo yum install gd gd-devel
Last metadata expiration check: 0:04:23 ago on Thu Oct  3 06:48:23 2024.
Dependencies resolved.
================================================================================================
 Package                          Architecture     Version                          Repository          Size
================================================================================================
Installing:
 gd                               x86_64           2.3.3-5.amzn2023.0.3             amazonlinux        139 k
 gd-devel                         x86_64           2.3.3-5.amzn2023.0.3             amazonlinux         38 k
Installing dependencies:
 brotli                           x86_64           1.0.9-4.amzn2023.0.2             amazonlinux        314 k
 brotli-devel                     x86_64           1.0.9-4.amzn2023.0.2             amazonlinux         31 k
 bzip2-devel                      x86_64           1.0.8-6.amzn2023.0.2             amazonlinux        214 k
 cairo                            x86_64           1.17.6-2.amzn2023.0.1            amazonlinux        684 k
 cmake-filesystem                 x86_64           3.22.2-1.amzn2023.0.4            amazonlinux         16 k
 fontconfig                       x86_64           2.13.94-2.amzn2023.0.2           amazonlinux        273 k
 fontconfig-devel                 x86_64           2.13.94-2.amzn2023.0.2           amazonlinux        128 k
 fonts-filesystem                 noarch           1:2.0.5-12.amzn2023.0.2          amazonlinux        9.5 k
 freetype                         x86_64           2.13.2-5.amzn2023.0.1            amazonlinux        423 k
 freetype-devel                   x86_64           2.13.2-5.amzn2023.0.1            amazonlinux        912 k
 glib2-devel                      x86_64           2.74.7-689.amzn2023.0.2          amazonlinux        486 k
 google-noto-fonts-common         noarch           20201206-2.amzn2023.0.2          amazonlinux         15 k
 google-noto-sans-vf-fonts        noarch           20201206-2.amzn2023.0.2          amazonlinux        492 k
 graphite2                        x86_64           1.3.14-7.amzn2023.0.2            amazonlinux         97 k
 graphite2-devel                  x86_64           1.3.14-7.amzn2023.0.2            amazonlinux         21 k
 harfbuzz                         x86_64           7.0.0-2.amzn2023.0.1             amazonlinux        868 k
 harfbuzz-devel                   x86_64           7.0.0-2.amzn2023.0.1             amazonlinux        404 k
 harfbuzz-icu                     x86_64           7.0.0-2.amzn2023.0.1             amazonlinux         18 k
 jbigkit-libs                     x86_64           2.1-21.amzn2023.0.2              amazonlinux         54 k
```

```
Installed:
  brotli-1.0.9-4.amzn2023.0.2.x86_64                        brotli-devel-1.0.9-4.amzn2023.0.2.x86_64
  bzip2-devel-1.0.8-6.amzn2023.0.2.x86_64                   cairo-1.17.6-2.amzn2023.0.1.x86_64
  cmake-filesystem-3.22.2-1.amzn2023.0.4.x86_64             fontconfig-2.13.94-2.amzn2023.0.2.x86_64
  fontconfig-devel-2.13.94-2.amzn2023.0.2.x86_64            fonts-filesystem-1:2.0.5-12.amzn2023.0.2.noarch
  freetype-2.13.2-5.amzn2023.0.1.x86_64                     freetype-devel-2.13.2-5.amzn2023.0.1.x86_64
  gd-2.3.3-5.amzn2023.0.3.x86_64                            gd-devel-2.3.3-5.amzn2023.0.3.x86_64
  glib2-devel-2.74.7-689.amzn2023.0.2.x86_64                google-noto-fonts-common-20201206-2.amzn2023.0.2.noarch
  google-noto-sans-vf-fonts-20201206-2.amzn2023.0.2.noarch  graphite2-1.3.14-7.amzn2023.0.2.x86_64
  graphite2-devel-1.3.14-7.amzn2023.0.2.x86_64              harfbuzz-7.0.0-2.amzn2023.0.1.x86_64
  harfbuzz-devel-7.0.0-2.amzn2023.0.1.x86_64                harfbuzz-icu-7.0.0-2.amzn2023.0.1.x86_64
  jbigkit-libs-2.1-21.amzn2023.0.2.x86_64                   langpacks-core-font-en-3.0-21.amzn2023.0.4.noarch
  libICE-1.0.10-6.amzn2023.0.2.x86_64                       libSM-1.2.3-8.amzn2023.0.2.x86_64
  libX11-1.7.2-3.amzn2023.0.4.x86_64                        libX11-common-1.7.2-3.amzn2023.0.4.noarch
```

5. Create a new Nagios User with its password. You'll have to enter the password twice for confirmation. This is require as to give separate permissions for nagios

sudo adduser -m nagios

sudo passwd nagios

```
[ec2-user@ip-172-31-32-235 ~]$ 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.
```

6. Create a new user group and use these commands so that you don't have to use sudo for Apache and Nagios

sudo groupadd nagcmd
sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache

```
[ec2-user@ip-172-31-32-235 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-32-235 ~]$ sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache
[ec2-user@ip-172-31-32-235 ~]$
```

7. Create a new directory for Nagios downloads

mkdir ~/downloads
cd ~/downloads

```
[ec2-user@ip-172-31-32-235 ~]$ mkdir ~/downloads
cd ~/downloads
[ec2-user@ip-172-31-32-235 downloads]$
```

8. Use wget to download the installation source zip files.

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

```
[ec2-user@ip-172-31-32-235 downloads]$ wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
--2024-10-03 06:57:29--  https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2600:3c00::f03c:92ff:fef7:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: 'nagios-4.5.5.tar.gz'

nagios-4.5.5.tar.gz             100%[=====================================================>]   1.97M  5.21MB/s    in 0.4s

2024-10-03 06:57:29 (5.21 MB/s) - 'nagios-4.5.5.tar.gz' saved [2065473/2065473]
```

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

```
[ec2-user@ip-172-31-32-235 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
--2024-10-03 06:57:45--  https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2753049 (2.6M) [application/x-gzip]
Saving to: 'nagios-plugins-2.4.11.tar.gz'

nagios-plugins-2.4.11.tar.gz     100%[=================================================>]   2.62M  5.29MB/s    in 0.5s

2024-10-03 06:57:46 (5.29 MB/s) - 'nagios-plugins-2.4.11.tar.gz' saved [2753049/2753049]
```

9. Use tar to unzip the downloaded archive and change to that directory.

tar zxvf nagios-4.5.5.tar.gz

```
[ec2-user@ip-172-31-32-235 downloads]$ tar zxvf nagios-4.5.5.tar.gz
nagios-4.5.5/
nagios-4.5.5/.github/
nagios-4.5.5/.github/workflows/
nagios-4.5.5/.github/workflows/test.yml
nagios-4.5.5/.gitignore
nagios-4.5.5/CONTRIBUTING.md
nagios-4.5.5/Changelog
nagios-4.5.5/INSTALLING
nagios-4.5.5/LEGAL
nagios-4.5.5/LICENSE
nagios-4.5.5/Makefile.in
nagios-4.5.5/README.md
nagios-4.5.5/THANKS
nagios-4.5.5/UPGRADING
nagios-4.5.5/aclocal.m4
nagios-4.5.5/autoconf-macros/
nagios-4.5.5/autoconf-macros/.gitignore
nagios-4.5.5/autoconf-macros/CHANGELOG.md
nagios-4.5.5/autoconf-macros/LICENSE
nagios-4.5.5/autoconf-macros/LICENSE.md
nagios-4.5.5/autoconf-macros/README.md
nagios-4.5.5/autoconf-macros/add_group_user
nagios-4.5.5/autoconf-macros/ax_nagios_get_distrib
nagios-4.5.5/autoconf-macros/ax_nagios_get_files
nagios-4.5.5/autoconf-macros/ax_nagios_get_inetd
nagios-4.5.5/autoconf-macros/ax_nagios_get_init
nagios-4.5.5/autoconf-macros/ax_nagios_get_os
```

10. Navigate to the extracted folder and Run the configuration script with the same group name you previously created.

cd nagios-4.5.5 (check your own version using nagios - v or ls)

```
[ec2-user@ip-172-31-32-235 downloads]$ ls
nagios-4.5.5  nagios-4.5.5.tar.gz  nagios-plugins-2.4.11.tar.gz
[ec2-user@ip-172-31-32-235 downloads]$
```

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

```
[ec2-user@ip-172-31-32-235 downloads]$ cd nagios-4.5.5
[ec2-user@ip-172-31-32-235 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
checking whether make sets $(MAKE)... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking for sys/wait.h that is POSIX.1 compatible... yes
checking for stdio.h... yes
checking for stdlib.h... yes
checking for string.h... yes
```

```
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-172-31-32-235 nagios-4.5.5]$
```

We got error, because ssl headers library is not installed

It can be installed using **sudo yum install openssl-devel**

```
[ec2-user@ip-172-31-32-235 nagios-4.5.5]$ sudo yum install openssl-devel
Last metadata expiration check: 0:54:32 ago on Thu Oct  3 06:48:23 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                                   12 MB/s | 3.0 MB     00:00
--------------------------------------------------------------------------------------------------------------------------------
Total                                                                            10 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
```

Now rerun

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

```
*** Configuration summary for nagios 4.5.5 2024-09-17 ***:

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

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


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

11. Compile the source code.

make all

```
*** 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.

/usr/bin/install -c -m 775 -o nagios -g nagcmd -d /usr/local/nagios/var/rw
chmod g+s /usr/local/nagios/var/rw

*** External command directory configured ***


[ec2-user@ip-172-31-32-235 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. Install binaries, init script and sample config files. Lastly, set permissions on the external command directory.

sudo make install

sudo make install-init

sudo make install-config

sudo make install-commandmode

```
[ec2-user@ip-172-31-32-235 nagios-4.5.5]$ sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode
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
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/html'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/media
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/stylesheets
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/contexthelp
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/docs
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/docs/images
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/js
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/images
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/images/logos
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/includes
```

13. Edit the config file and change the email address so that we can receive timely alerts about the status of our system.

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

```
  GNU nano 5.8                    /usr/local/nagios/etc/objects/contacts.cfg                    Modified
#
#       You don't need to keep these definitions in a separate file from your
#       other object definitions.  This has been done just to make things
#       easier to understand.
#
###############################################################################



###############################################################################
#
# CONTACTS
#
###############################################################################

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

define contact {

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

And change email with your email

14. Configure the web interface. This is used to set up web server configuration of nagios dashboard.

sudo make install-webconf

```
[ec2-user@ip-172-31-32-235 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 ***
```

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

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

```
[ec2-user@ip-172-31-32-235 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
```

16. Restart Apache

sudo service httpd restart

```
[ec2-user@ip-172-31-32-235 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-32-235 nagios-4.5.5]$
```

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

cd ~/downloads

tar zxvf nagios-plugins-2.4.11.tar.gz

```
[ec2-user@ip-172-31-32-235 nagios-4.5.5]$ cd ~/downloads
tar zxvf nagios-plugins-2.4.11.tar.gz
nagios-plugins-2.4.11/
nagios-plugins-2.4.11/build-aux/
nagios-plugins-2.4.11/build-aux/compile
nagios-plugins-2.4.11/build-aux/config.guess
nagios-plugins-2.4.11/build-aux/config.rpath
nagios-plugins-2.4.11/build-aux/config.sub
nagios-plugins-2.4.11/build-aux/install-sh
nagios-plugins-2.4.11/build-aux/ltmain.sh
nagios-plugins-2.4.11/build-aux/missing
nagios-plugins-2.4.11/build-aux/mkinstalldirs
nagios-plugins-2.4.11/build-aux/depcomp
nagios-plugins-2.4.11/build-aux/snippet/
nagios-plugins-2.4.11/build-aux/snippet/_Noreturn.h
nagios-plugins-2.4.11/build-aux/snippet/arg-nonnull.h
nagios-plugins-2.4.11/build-aux/snippet/c++defs.h
nagios-plugins-2.4.11/build-aux/snippet/warn-on-use.h
nagios-plugins-2.4.11/build-aux/test-driver
nagios-plugins-2.4.11/config_test/
nagios-plugins-2.4.11/config_test/Makefile
nagios-plugins-2.4.11/config_test/run_tests
nagios-plugins-2.4.11/config_test/child_test.c
nagios-plugins-2.4.11/gl/
nagios-plugins-2.4.11/gl/m4/
nagios-plugins-2.4.11/gl/m4/00gnulib.m4
nagios-plugins-2.4.11/gl/m4/absolute-header.m4
nagios-plugins-2.4.11/gl/m4/alloca.m4
nagios-plugins-2.4.11/gl/m4/arpa_inet_h.m4
```

18. Compile and install plugins

cd nagios-plugins-2.4.11

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

```
[ec2-user@ip-172-31-32-235 downloads]$ cd nagios-plugins-2.4.11
[ec2-user@ip-172-31-32-235 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
```

19. Start Nagios and Add Nagios to the list of system services

sudo chkconfig --add nagios

sudo chkconfig nagios on

```
[ec2-user@ip-172-31-32-235 nagios-plugins-2.4.11]$ sudo chkconfig --add nagios
sudo chkconfig nagios on
error reading information on service nagios: No such file or directory
Note: Forwarding request to 'systemctl enable nagios.service'.
Created symlink /etc/systemd/system/multi-user.target.wants/nagios.service → /usr/lib/systemd/system/nagios.service.
[ec2-user@ip-172-31-32-235 nagios-plugins-2.4.11]$
```

Verify the sample configuration files

```
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
```

As we can see no errors were detected

sudo service nagios start

```
[ec2-user@ip-172-31-32-235 nagios-plugins-2.4.11]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-32-235 nagios-plugins-2.4.11]$
```

20. Check the status of Nagios

sudo systemctl status nagios

```
[ec2-user@ip-172-31-32-235 nagios-plugins-2.4.11]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-32-235 nagios-plugins-2.4.11]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Fri 2024-10-04 03:42:22 UTC; 1min 5s ago
       Docs: https://www.nagios.org/documentation
    Process: 20075 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 20076 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 20077 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 5.9M
        CPU: 92ms
     CGroup: /system.slice/nagios.service
             ├─20077 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─20078 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─20079 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─20080 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─20081 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─20082 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully initialized
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: qh: core query handler registered
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: qh: echo service query handler registered
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: qh: help for the query handler registered
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: wproc: Successfully registered manager as @wproc with query handler
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: wproc: Registry request: name=Core Worker 20079;pid=20079
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: wproc: Registry request: name=Core Worker 20080;pid=20080
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: wproc: Registry request: name=Core Worker 20081;pid=20081
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: wproc: Registry request: name=Core Worker 20078;pid=20078
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: Successfully launched command file worker with pid 20082
```

The nagios service is running and working normally

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

Enter username as **nagiosadmin** and password which we had set previously

22. After entering the correct credentials, you will see the Home page of Nagios.

Now we can see system logs using nagios

Current Network Status
Last Updated: Fri Oct 4 14:20:53 UTC 2024
Updated every 90 seconds
Nagios® Core™ 4.5.5 - www.nagios.org
Logged in as *nagiosadmin*

View History For all hosts
View Notifications For All Hosts
View Host Status Detail For All Hosts

Host Status Totals

Up	Down	Unreachable	Pending
1	0	0	0

All Problems	All Types
0	1

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
6	1	0	1	0

All Problems	All Types
2	8

Service Status Details For All Hosts

Limit Results: 100

Host	Service	Status	Last Check	Duration	Attempt	Status Information
localhost	Current Load	OK	10-04-2024 14:15:58	0d 0h 14m 55s	1/4	OK - load average: 0.00, 0.00, 0.00
	Current Users	OK	10-04-2024 14:16:36	0d 0h 14m 17s	1/4	USERS OK - 1 users currently logged in
	HTTP	WARNING	10-04-2024 14:20:13	0d 0h 10m 40s	4/4	HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes in 0.001 second response time
	PING	OK	10-04-2024 14:17:51	0d 0h 13m 2s	1/4	PING OK - Packet loss = 0%, RTA = 0.03 ms
	Root Partition	OK	10-04-2024 14:18:28	0d 0h 12m 25s	1/4	DISK OK - free space: / 6031 MiB (74.31% inode=98%):
	SSH	OK	10-04-2024 14:19:06	0d 0h 11m 47s	1/4	SSH OK - OpenSSH_8.7 (protocol 2.0)
	Swap Usage	CRITICAL	10-04-2024 14:17:43	0d 0h 8m 10s	4/4	SWAP CRITICAL - 0% free (0 MB out of 0 MB) - Swap is either disabled, not present, or of zero size.
	Total Processes	OK	10-04-2024 14:20:21	0d 0h 10m 32s	1/4	PROCS OK: 37 processes with STATE = RSZDT

Results 1 - 8 of 8 Matching Services

Conclusion:

The successful installation and configuration of Nagios Core, Nagios Plugins, and NRPE on the EC2 instance marks a significant step towards establishing comprehensive system monitoring capabilities. The ability to effectively track and analyze system performance, resource utilization, and potential issues is crucial for maintaining operational efficiency and preventing downtime. In the course of experiment various issues regarding the nagios version and the ssl error was found and that was fixed by installing the openssl library. Apart from that we got to learn the basic fundamentals of nagios and how the plugins are configured in the system or the host machine. At the end Nagios server was hosted on the ec2 instance with the public ip.

Experiment 10:

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

Prerequisites:
Nagios Server running on Amazon Linux Machine.

STEPS:

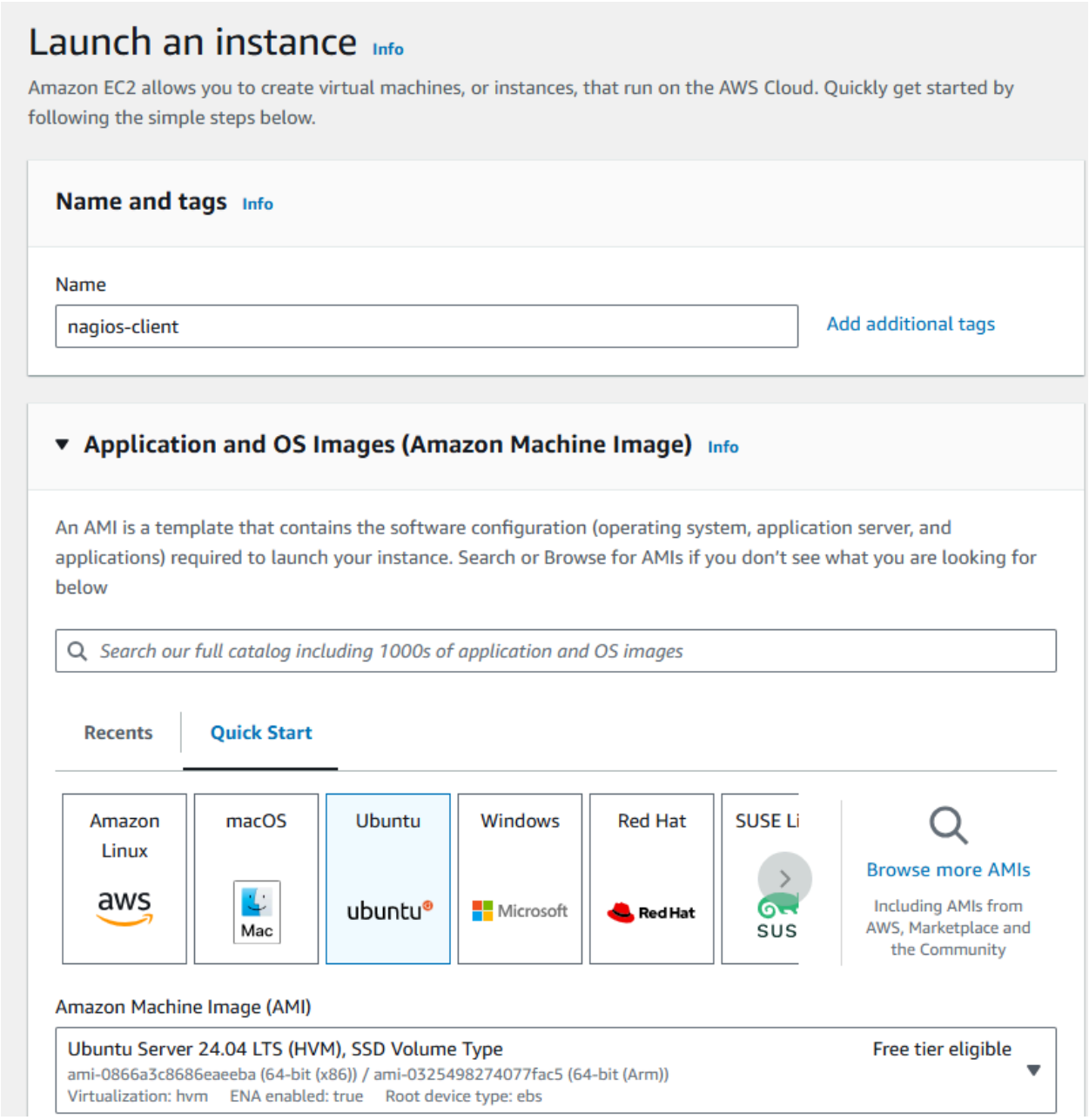
1. To Confirm Nagios is running on the server side Perform the following command on your Amazon Linux Machine (Nagios-host).

   sudo systemctl status nagios

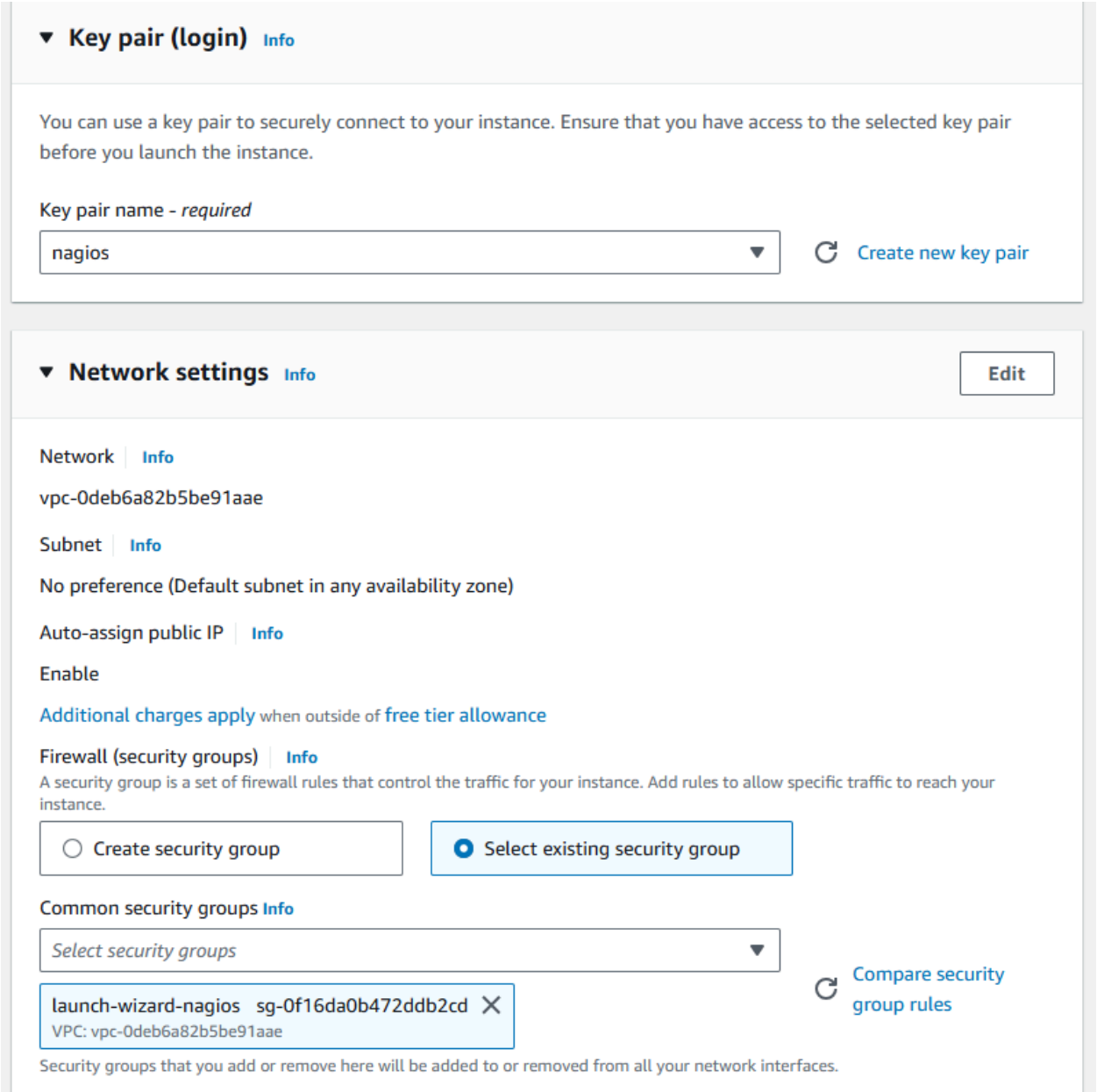
```
[ec2-user@ip-172-31-32-235 nagios-plugins-2.4.11]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-32-235 nagios-plugins-2.4.11]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Fri 2024-10-04 03:42:22 UTC; 1min 5s ago
       Docs: https://www.nagios.org/documentation
    Process: 20075 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 20076 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 20077 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 5.9M
        CPU: 92ms
     CGroup: /system.slice/nagios.service
             ├─20077 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─20078 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─20079 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─20080 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─20081 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─20082 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully initialized
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: qh: core query handler registered
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: qh: echo service query handler registered
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: qh: help for the query handler registered
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: wproc: Successfully registered manager as @wproc with query handler
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: wproc: Registry request: name=Core Worker 20079;pid=20079
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: wproc: Registry request: name=Core Worker 20080;pid=20080
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: wproc: Registry request: name=Core Worker 20081;pid=20081
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: wproc: Registry request: name=Core Worker 20078;pid=20078
Oct 04 03:42:22 ip-172-31-32-235.ec2.internal nagios[20077]: Successfully launched command file worker with pid 20082
```

   As nagios service on main machine is running, we can proceed further

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

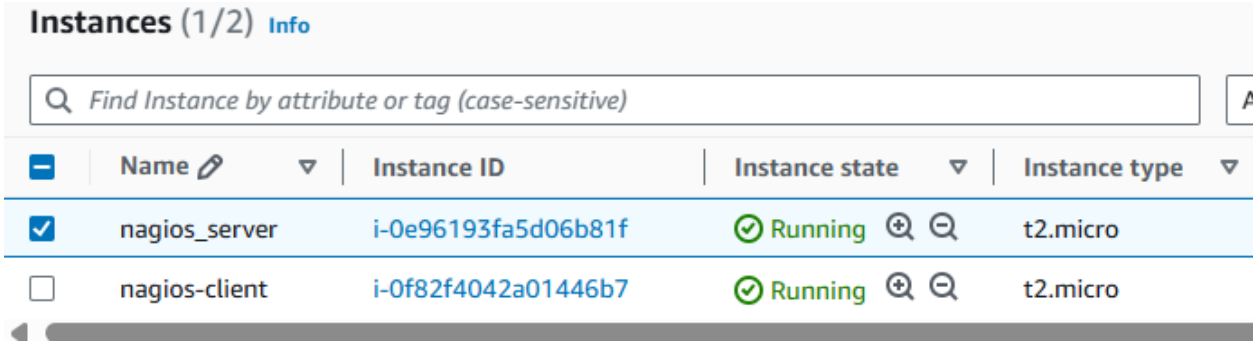


Select the existing security group and choose the one configured in Experiment 9, or the same security group used for the Nagios server (Nagios-host).



Launched successfully message appears:

You can see both the instances required for this experiment



After creating the EC2 instance, click on "Connect," copy the SSH command from the SSH Client section, open the terminal in the folder containing your RSA key (.pem), and paste the command.

```
PS C:\Users\Avan\Downloads> ssh -i "nagios.pem" ubuntu@ec2-18-206-185-207.compute-1.amazonaws.com
The authenticity of host 'ec2-18-206-185-207.compute-1.amazonaws.com (18.206.185.207)' can't be established.
ED25519 key fingerprint is SHA256:HGCTF4Wul1HL3RyrMxyq6ea/z7UeVTLxcXAi0rcw50Q.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-18-206-185-207.compute-1.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 Fri Oct  4 04:09:51 UTC 2024

  System load:  0.31               Processes:             107
  Usage of /:   22.9% of 6.71GB    Users logged in:       0
  Memory usage: 22%                IPv4 address for enX0: 172.31.36.170
  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 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.
```

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-172-31-32-235 ~]$ ps -ef | grep nagios
ec2-user    2860    2353  0 03:31 pts/0    00:00:00 journalctl -xeu nagios.service
nagios     20077       1  0 03:42 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios     20078   20077  0 03:42 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     20079   20077  0 03:42 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     20080   20077  0 03:42 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     20081   20077  0 03:42 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios     20082   20077  0 03:42 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
ec2-user   21798    2353  0 04:12 pts/0    00:00:00 grep --color=auto nagios
[ec2-user@ip-172-31-32-235 ~]$
```

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-172-31-32-235 ~]$ sudo su
[root@ip-172-31-32-235 ec2-user]# mkdir /usr/local/nagios/etc/objects/monitorhosts
[root@ip-172-31-32-235 ec2-user]# mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
```

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-172-31-32-235 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxho
sts/linuxserver.cfg
[root@ip-172-31-32-235 ec2-user]#
```

7. Open linuxserver.cfg using nano and make the following changes in all Positions everywhere in file

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

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

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



# OBJECT CONFIGURATION FILE(S)
# These are the object configuration files in which you define hosts,
# host groups, contacts, contact groups, services, etc.
# You can split your object definitions across several config files
# if you wish (as shown below), or keep them all in a single config file.

# You can specify individual object config files as shown below:
cfg_file=/usr/local/nagios/etc/objects/commands.cfg
cfg_file=/usr/local/nagios/etc/objects/contacts.cfg
cfg_file=/usr/local/nagios/etc/objects/timeperiods.cfg
cfg_file=/usr/local/nagios/etc/objects/templates.cfg
cfg_file=/usr/local/nagios/etc/objects/monitorhosts/
# Definitions for monitoring the local (Linux) host
cfg_file=/usr/local/nagios/etc/objects/localhost.cfg

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

^G Help      ^O Write Out   ^W Where Is   ^K Cut       ^T Execute
^X Exit      ^R Read File   ^\ Replace    ^U Paste     ^J Justify
```

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

```
  GNU nano 5.8                    /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg                    Modifie
# 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                 127.31.36.170
}



###############################################################################
#
# HOST GROUP DEFINITION
#
###############################################################################

# Define an optional hostgroup for Linux machines

define hostgroup {

    hostgroup_name          linux-servers1           ; The name of the hostgroup
    alias                   Linux Servers           ; Long name of the group
    members                 localhost               ; Comma separated list of hosts that belong to this group
}


###############################################################################

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

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/
nano /usr/local/nagios/etc/nagios.cfg

```
# 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_file=/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-172-31-32-235 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...
Warning: Duplicate definition found for service 'HTTP' on host 'localhost' (config file '/usr/local/na
Warning: Duplicate definition found for service 'SSH' on host 'localhost' (config file '/usr/local/nag
Warning: Duplicate definition found for service 'Swap Usage' on host 'localhost' (config file '/usr/lo
Warning: Duplicate definition found for service 'Current Load' on host 'localhost' (config file '/usr/
Warning: Duplicate definition found for service 'Total Processes' on host 'localhost' (config file '/u
Warning: Duplicate definition found for service 'Current Users' on host 'localhost' (config file '/usr
Warning: Duplicate definition found for service 'Root Partition' on host 'localhost' (config file '/us
Warning: Duplicate definition found for service 'PING' on host 'localhost' (config file '/usr/local/na
   Read object config files okay...

Running pre-flight check on configuration data...

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

Total Warnings: 0
Total Errors:   0
```

We got no errors and warnings for current configuration

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

```
[root@ip-172-31-32-235 ec2-user]# service nagios restart
Redirecting to /bin/systemctl restart nagios.service
[root@ip-172-31-32-235 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-172-31-36-170:~$ sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [382 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [537 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [132 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [8860 B]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [384 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [159 kB]
Get:19 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [83.9 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [45.0 kB]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [14.9 kB]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [14.4 kB]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [3608 B]
Get:24 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [212 B]
Get:25 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [532 B]
Get:26 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:27 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [112 B]
```

```
Creating config file /etc/nagios-plugins/config/nt.cfg with new version

Creating config file /etc/nagios-plugins/config/pgsql.cfg with new version

Creating config file /etc/nagios-plugins/config/radius.cfg with new version

Creating config file /etc/nagios-plugins/config/rpc-nfs.cfg with new version

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-172-31-36-170:~$
```

12. Now restart the NRPE server by this command.

sudo systemctl restart nagios-nrpe-server

```
[ec2-user@ip-172-31-32-235 ~]$ sudo nano /etc/nagios/nrpe.cfg

[ec2-user@ip-172-31-32-235 ~]$ sudo systemctl restart nagios-nrpe-server
```

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

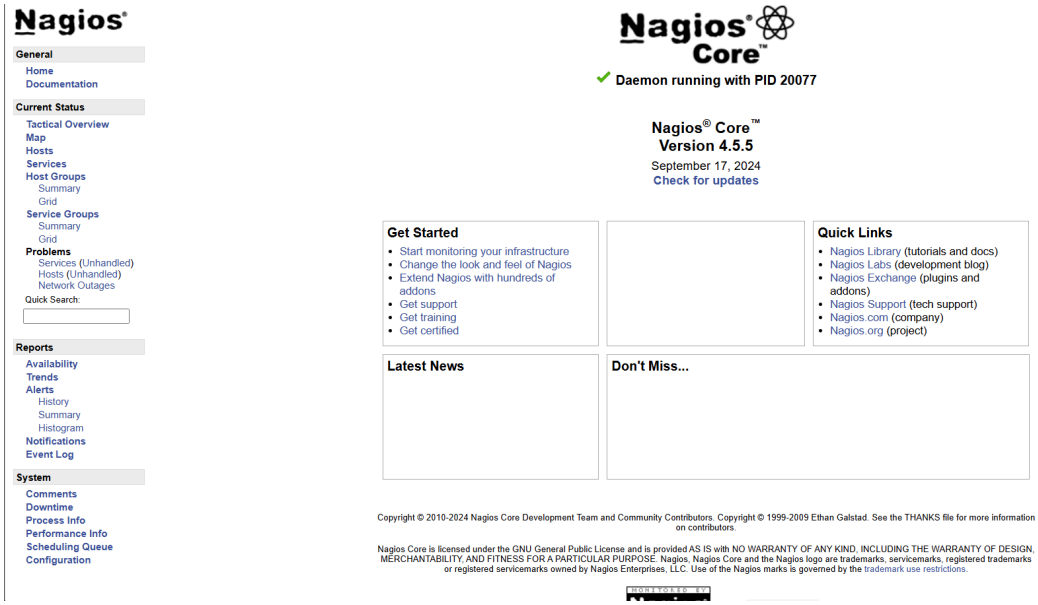
sudo systemctl status nagios

sudo systemctl status httpd

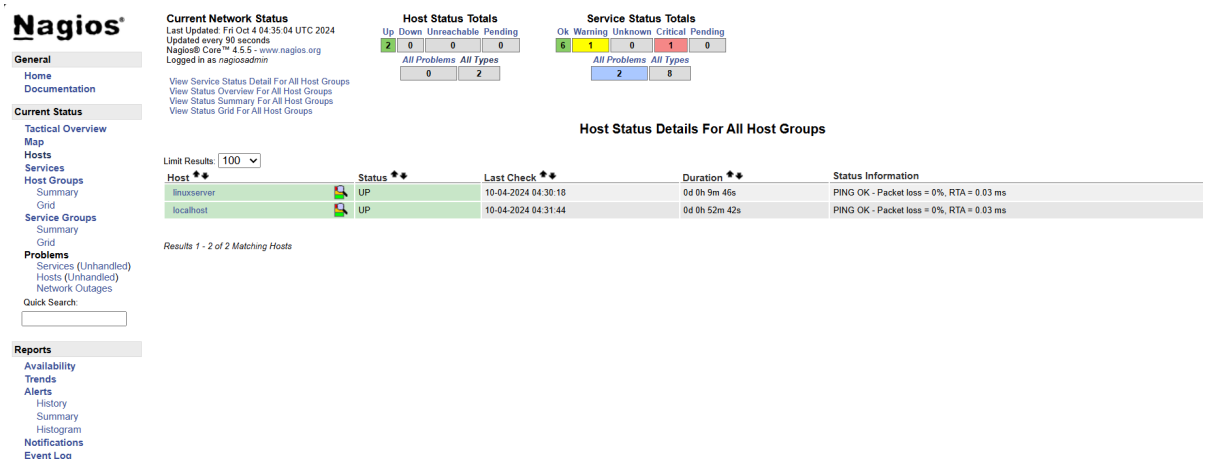
```
[root@ip-172-31-32-235 ec2-user]# sudo systemctl status httpd
sudo systemctl start httpd
sudo systemctl enable httpd
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
    Drop-In: /usr/lib/systemd/system/httpd.service.d
             └─php-fpm.conf
     Active: active (running) since Fri 2024-10-04 03:39:13 UTC; 55min ago
       Docs: man:httpd.service(8)
   Main PID: 5507 (httpd)
     Status: "Total requests: 51; Idle/Busy workers 100/0;Requests/sec: 0.0154; Bytes served/sec:  58 B/sec"
      Tasks: 177 (limit: 1112)
     Memory: 19.1M
        CPU: 1.983s
     CGroup: /system.slice/httpd.service
             ├─5507 /usr/sbin/httpd -DFOREGROUND
             ├─5509 /usr/sbin/httpd -DFOREGROUND
             ├─5510 /usr/sbin/httpd -DFOREGROUND
             ├─5511 /usr/sbin/httpd -DFOREGROUND
             └─5512 /usr/sbin/httpd -DFOREGROUND

Oct 04 03:39:12 ip-172-31-32-235.ec2.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...
Oct 04 03:39:13 ip-172-31-32-235.ec2.internal systemd[1]: Started httpd.service - The Apache HTTP Server.
Oct 04 03:39:13 ip-172-31-32-235.ec2.internal httpd[5507]: Server configured, listening on: port 80
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[root@ip-172-31-32-235 ec2-user]# Z
```

Both nagios and httpd service is running fine on host system

14. Now to check Nagios dashboard go to http://<nagios host ip>/nagios
Or you can go to the instance id and click on the Public Ip <-link-> and Nagios server gets loaded
Enter username as nagiosadmin and password which you set in Exp 9 if prompted.



Now Click on Hosts from left side panel



Our nagios client is showing up on nagios host dashboard

Host status :

Host Information
Last Updated: Fri Oct 4 04:35:25 UTC 2024
Updated every 90 seconds
Nagios® Core™ 4.5.5 - www.nagios.org
Logged in as *nagiosadmin*

View Status Detail For This Host
View Alert History For This Host
View Trends For This Host
View Alert Histogram For This Host
View Availability Report For This Host
View Notifications For This Host

Host
localhost
(linuxserver)

Member of
No hostgroups

127.31.36.170

Host State Information

Host Status:	UP (for 0d 0h 10m 7s)
Status Information:	PING OK - Packet loss = 0%, RTA = 0.03 ms
Performance Data:	rta=0.034000ms;3000.000000;5000.000000;0.000000 pl=0%;80;100;0
Current Attempt:	1/10 (HARD state)
Last Check Time:	10-04-2024 04:30:18
Check Type:	ACTIVE
Check Latency / Duration:	0.000 / 4.159 seconds
Next Scheduled Active Check:	10-04-2024 04:35:18
Last State Change:	10-04-2024 04:25:18
Last Notification:	N/A (notification 0)
Is This Host Flapping?	NO (0.00% state change)
In Scheduled Downtime?	NO
Last Update:	10-04-2024 04:35:17 (0d 0h 0m 8s ago)

Active Checks:	ENABLED
Passive Checks:	ENABLED
Obsessing:	ENABLED
Notifications:	ENABLED
Event Handler:	ENABLED
Flap Detection:	ENABLED

Host Comments
Add a new comment Delete all comments

Entry Time	Author	Comment	Comment ID	Persistent	Type	Expires	Actions
This host has no comments associated with it							

Service Status Details for Host localhost:

Current Network Status
Last Updated: Fri Oct 4 04:37:03 UTC 2024
Updated every 90 seconds
Nagios® Core™ 4.5.5 - www.nagios.org
Logged in as *nagiosadmin*

View History For This Host
View Notifications For This Host
View Service Status Detail For All Hosts

Host Status Totals

Up	Down	Unreachable	Pending
2	0	0	0

All Problems	All Types
0	2

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
6	1	0	1	0

All Problems	All Types
2	8

Service Status Details For Host 'localhost'

Limit Results: 100

Host	Service	Status	Last Check	Duration	Attempt	Status Information
localhost	Current Load	OK	10-04-2024 04:32:59	0d 0h 54m 4s	1/4	OK - load average: 0.00, 0.02, 0.00
	Current Users	OK	10-04-2024 04:33:37	0d 0h 53m 26s	1/4	USERS OK - 2 users currently logged in
	HTTP	WARNING	10-04-2024 04:32:14	0d 0h 49m 49s	4/4	HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes in 0.000 second response time
	PING	OK	10-04-2024 04:34:52	0d 0h 52m 11s	1/4	PING OK - Packet loss = 0%, RTA = 0.04 ms
	Root Partition	OK	10-04-2024 04:35:29	0d 0h 51m 34s	1/4	DISK OK - free space: / 5584 MiB (68.81% inode=98%):
	SSH	OK	10-04-2024 04:36:07	0d 0h 50m 56s	1/4	SSH OK - OpenSSH_8.7 (protocol 2.0)
	Swap Usage	CRITICAL	10-04-2024 04:34:44	0d 0h 47m 19s	4/4	SWAP CRITICAL - 0% free (0 MB out of 0 MB) - Swap is either disabled, not present, or of zero size.
	Total Processes	OK	10-04-2024 04:32:22	0d 0h 49m 41s	1/4	PROCS OK: 39 processes with STATE = RSZDT

Results 1 - 8 of 8 Matching Services

Here these details indicate that the client machine is running with :

Current Load: OK (load average is low).
Current Users: OK (2 users are logged in).
HTTP: Warning (HTTP 403 Forbidden, indicating a potential access permission issue).
PING: OK (no packet loss, response time is very low).
Swap Usage: Critical (swap is either 0 or disabled).
Total Processes: OK (39 processes are running as expected).

Conclusion:

In this experiment, we focused on monitoring ports, services, and a Linux server using Nagios. By following a structured approach, we successfully configured Nagios to monitor critical network services on the Linux server. Through the setup of both the Nagios host and client, we effectively tracked system performance, ensured service availability, and monitored key metrics such as CPU and memory usage.Additionally, the configuration of Nagios files, particularly defining the host name and correctly setting the IP address of the server, played a crucial role. This ensures proper identification and communication between the Nagios host and the monitored server, allowing accurate monitoring of services. Properly configuring these details is essential for the efficient operation and management of network resources.

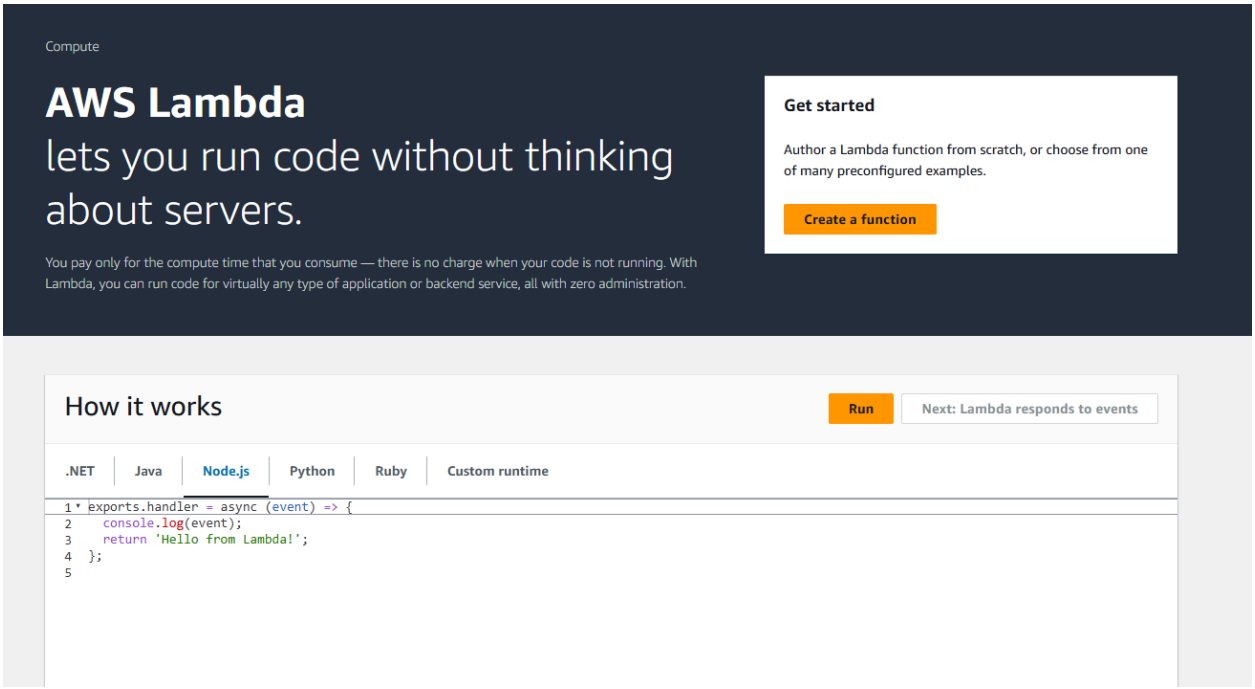
Experiment 11:

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

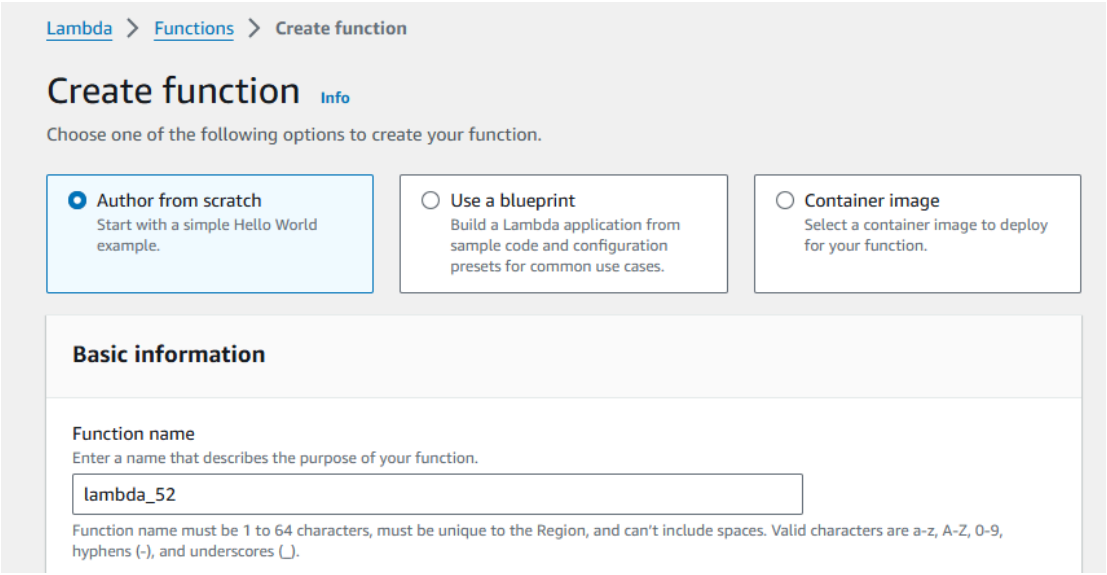
Steps:

1. Go to AWS ACADEMY.
2. **Create the lambda function:**

Firstly, Search lambda, then Open lambda and then click on create function button.

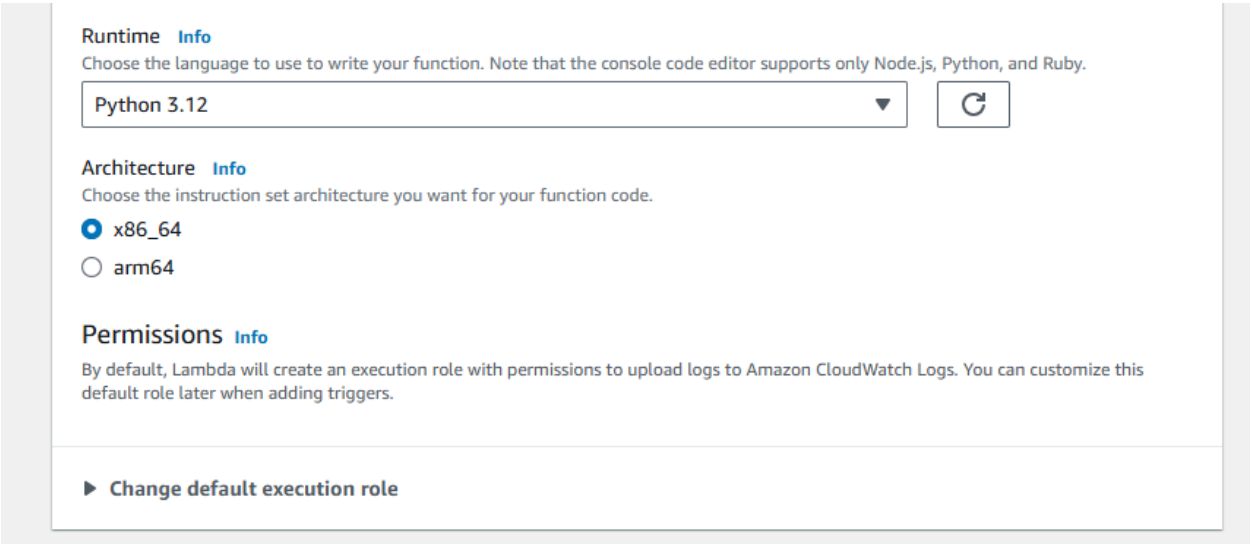


3. Now Give a name to your Lambda function,

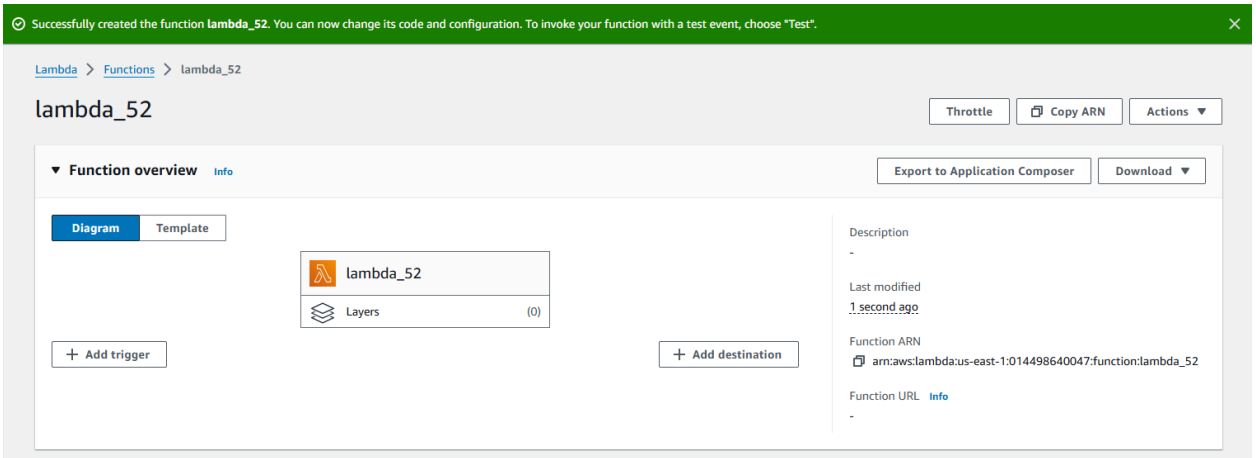


4. Select the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby. So will select Python 3.12, Architecture as x86, and Exceution role to

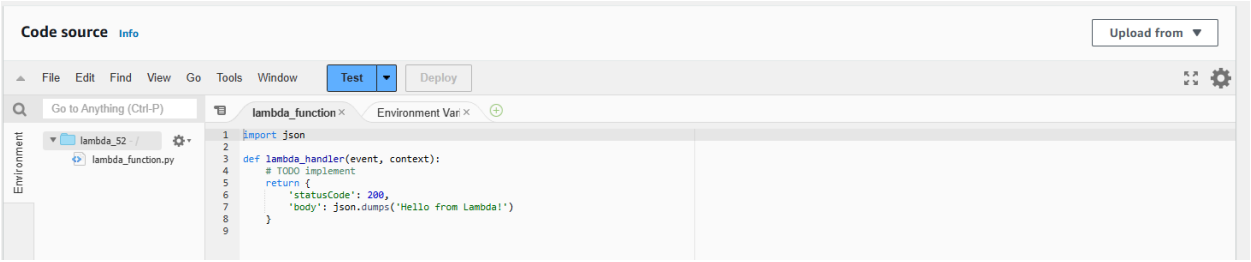
Create a new role with basic Lambda permissions.



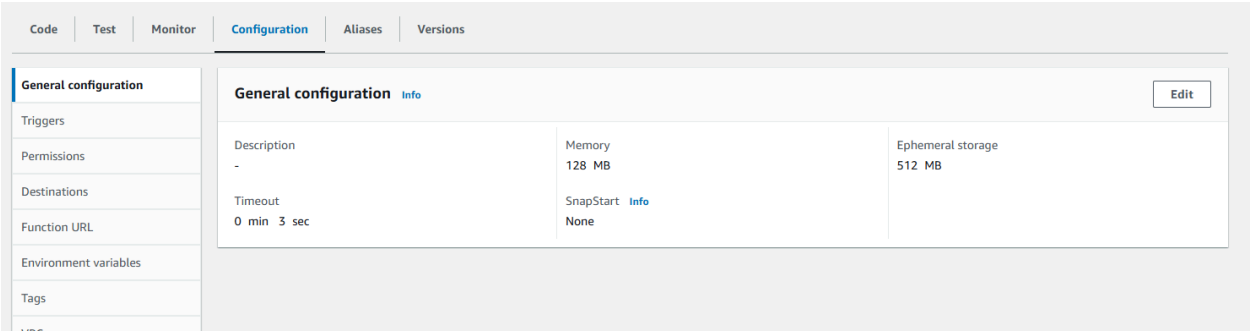
Thus the Lambda function was created successfully.



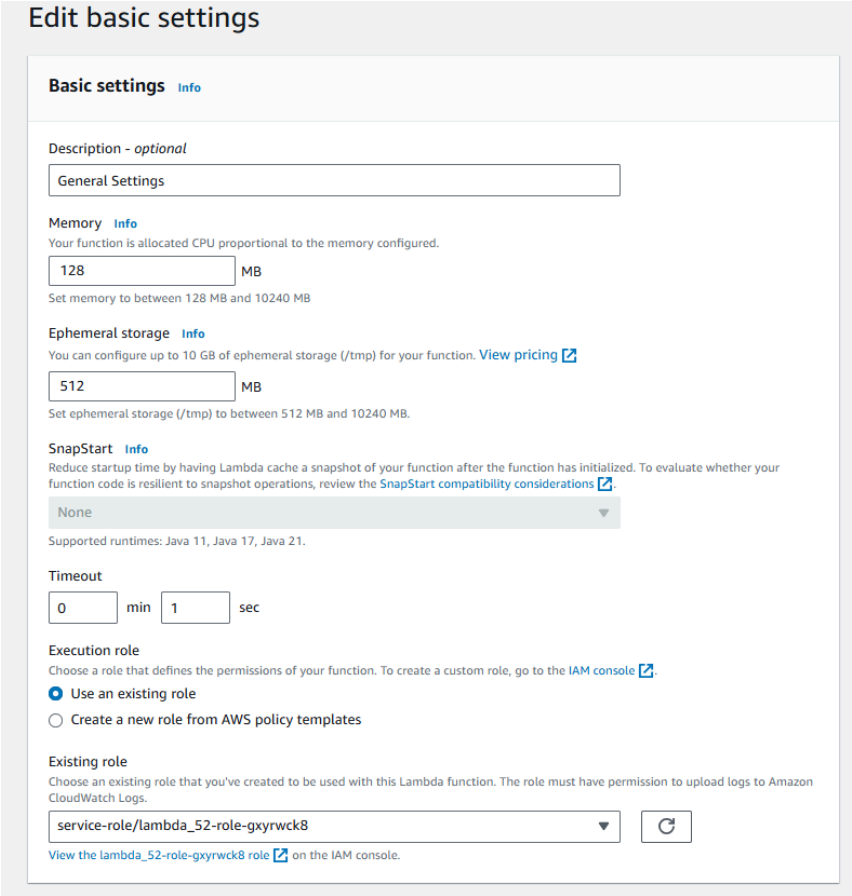
5. Code Source Section



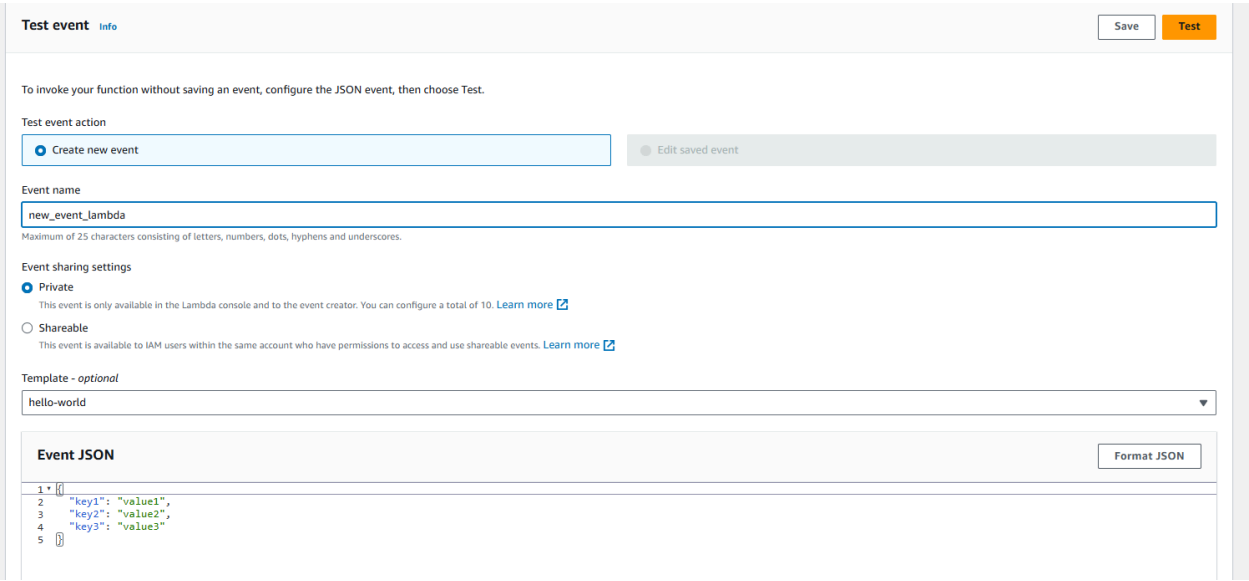
So now to edit the basic settings, go to configuration then click on edit.



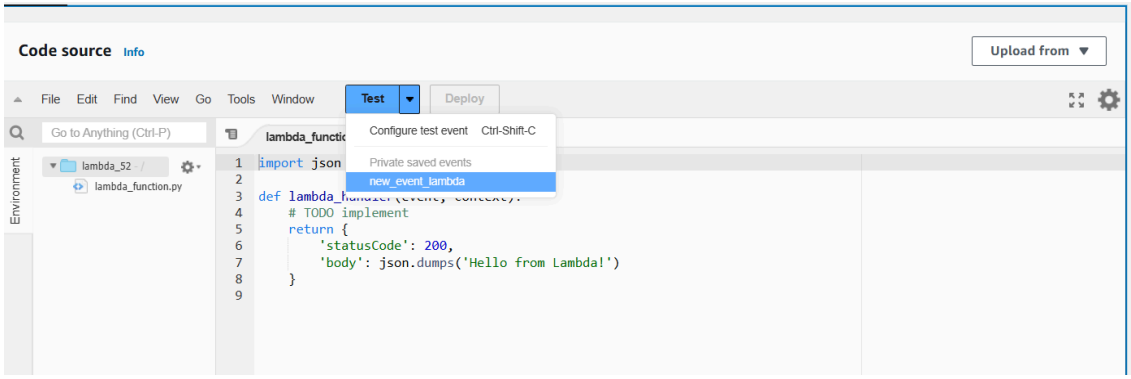
Now enter a description and change Memory and Timeout.Here, I've changed the Timeout period to 1 sec.



6. Now Click on the Test tab then select Create a new event, give a name to the event here i have given name as "new_event_lambda" and then select event sharing to private, and select hello-world template.



7. **Testing & Deployment:** Now In Code section select the created event (our_event) from the dropdown of test ,then click on test .

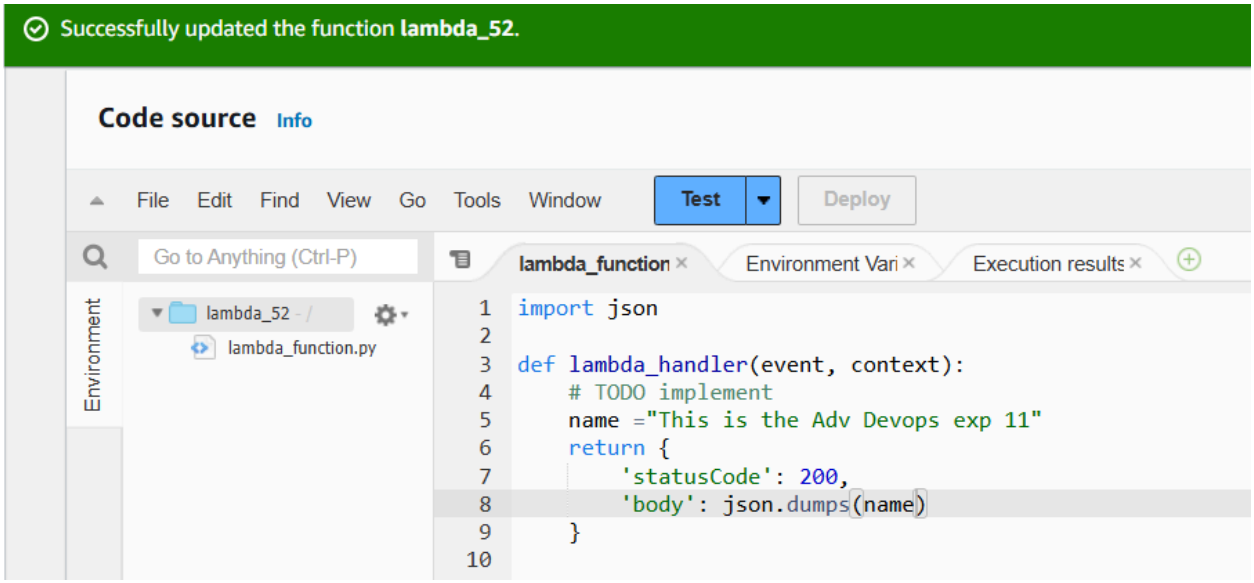


8. Now you will see the following output.

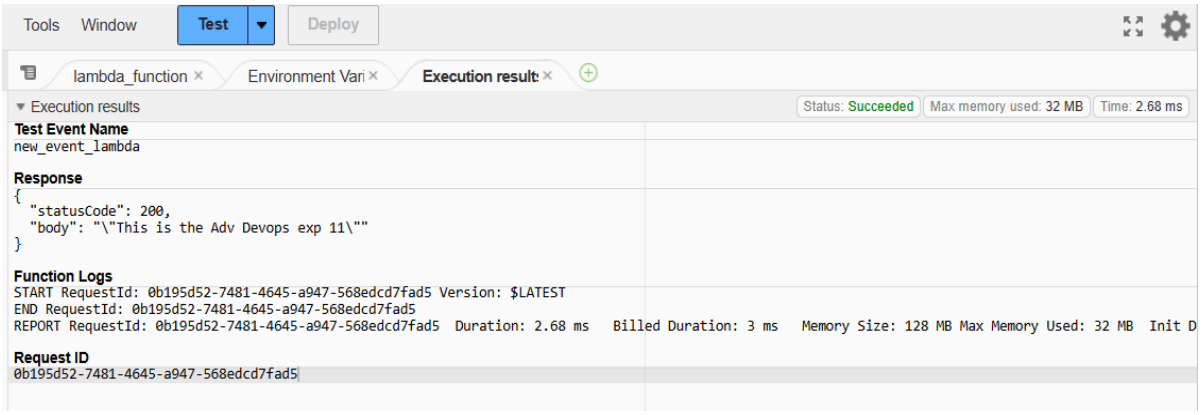
9. **Editing the given code into our own by adding a string or you can add anything**

You can edit your lambda function code.Here I have created a new string name “new_string” and assigned a string to it.

Now save it by ctrl+s and then click finally on deploy to deploy the changes.



10. **Testing and redeploying changes** Now click on the test and observe the output. Thus Output gives status code 200. This deployment is done successfully.



Conclusion:

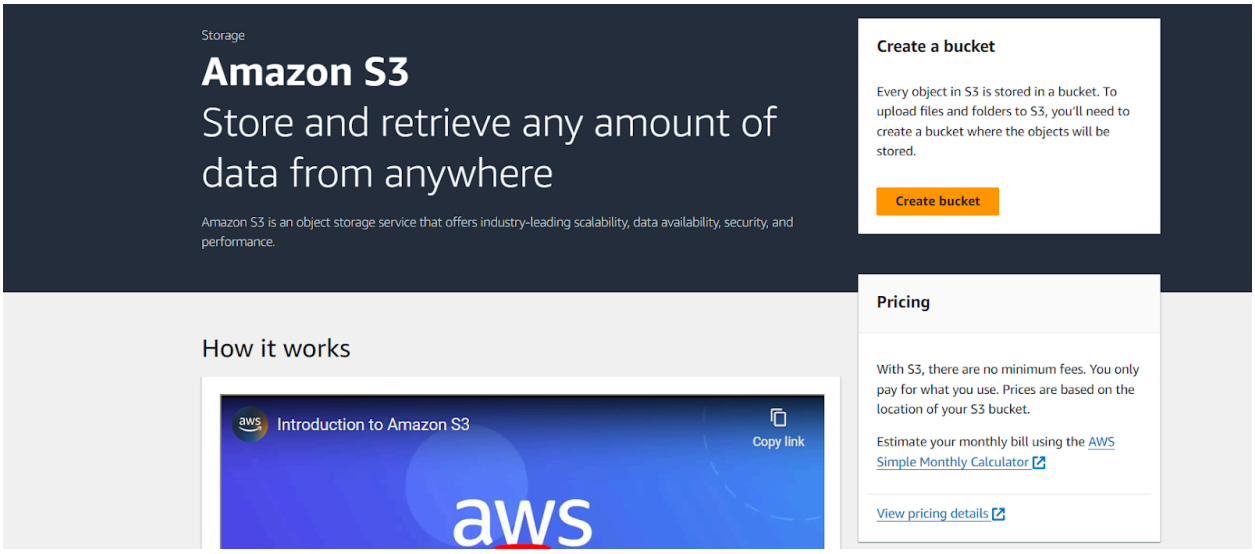
In this experiment, we successfully created an AWS Lambda function using Python as the chosen language. After configuring the basic settings, we adjusted the timeout to 1 second, tested the function, and deployed it. The deployment was successful. We then modified the Lambda function's code and redeployed it to observe real-time changes. This process highlighted the ease of using AWS Lambda for building serverless applications. However, one issue we encountered was that, while we initially added the code source in Python, we could have chosen other supported languages as well.

Experiment 12:

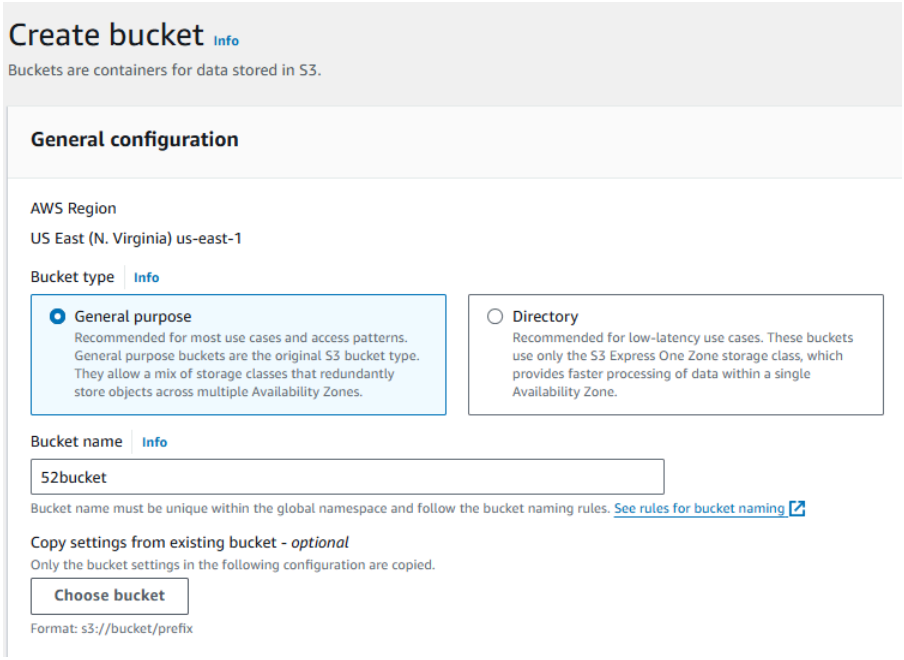
Aim: To create a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3

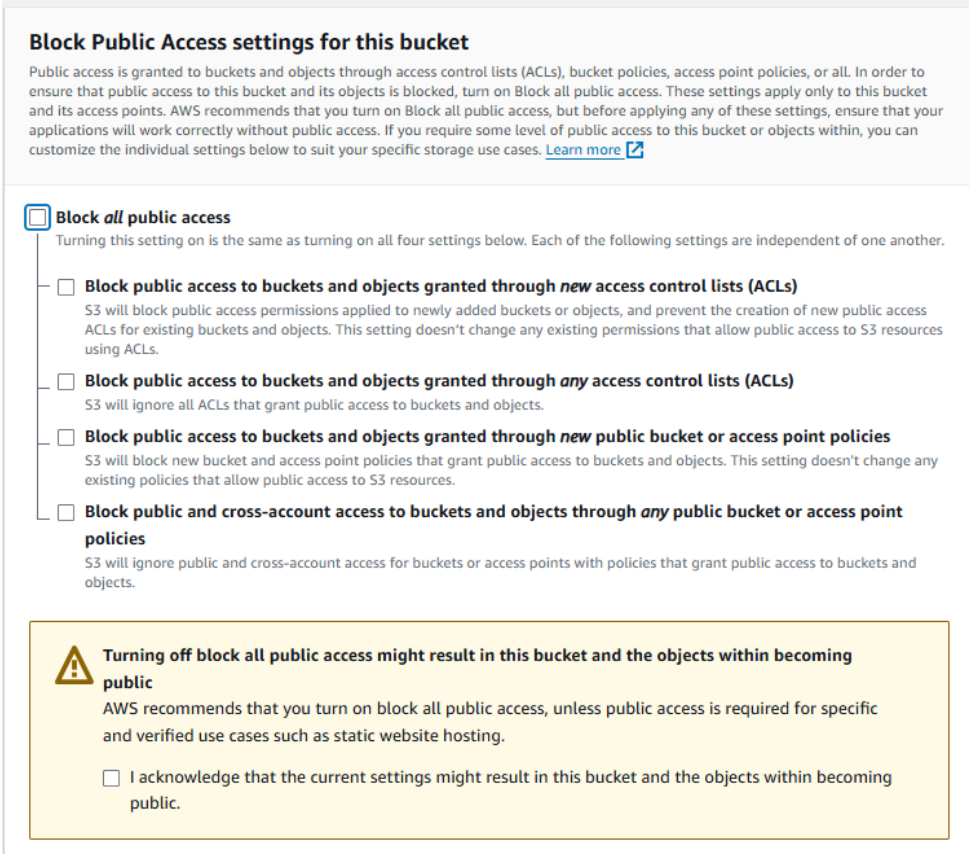
Steps :

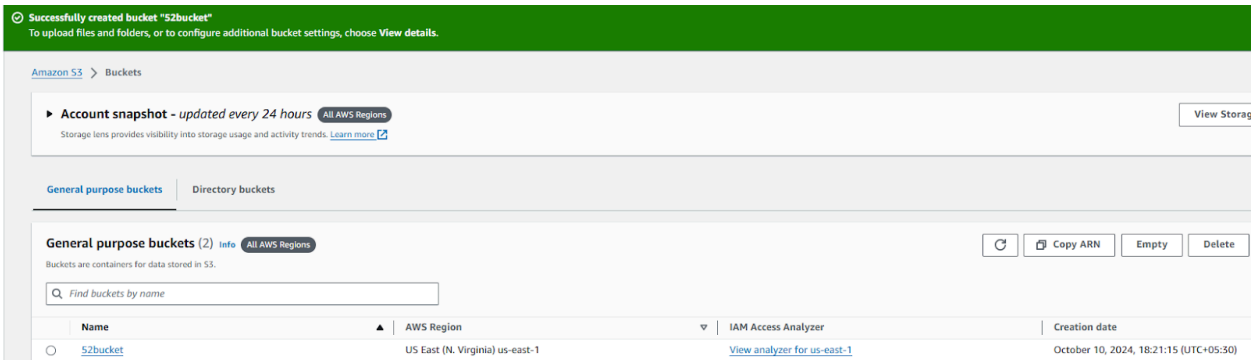
1: Login to your AWS Personal account. Now open S3 from services and click on create S3 bucket.



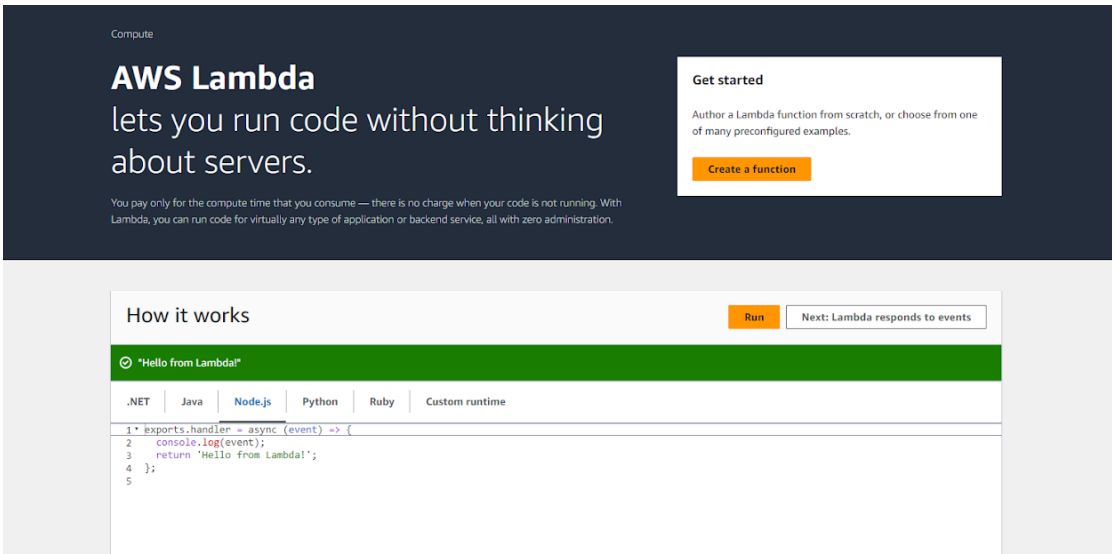
2: Now Give a name to the Bucket, select general purpose project and deselect the Block public access and keep other this to default.



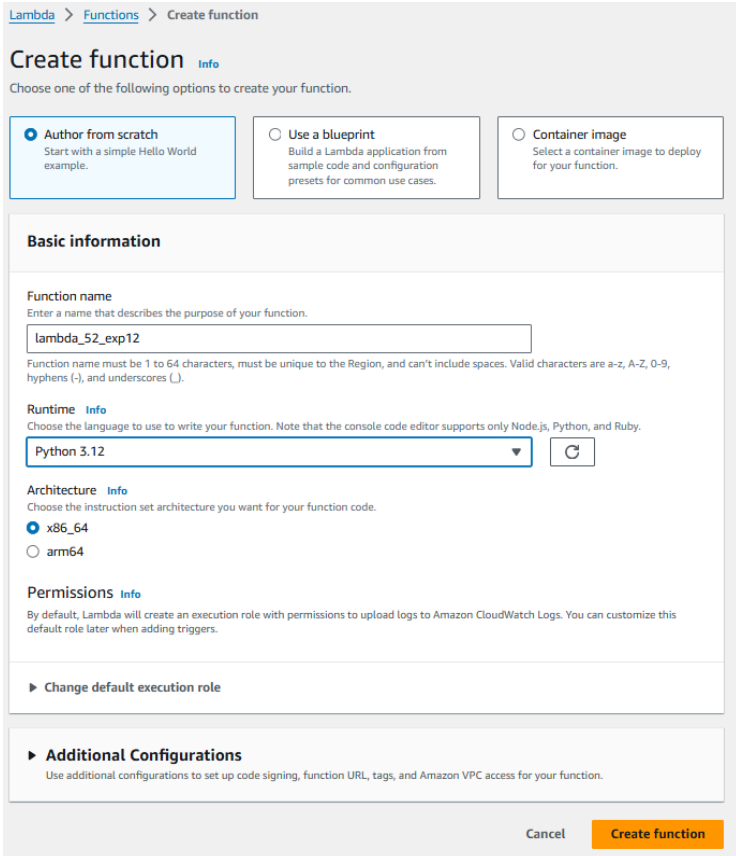




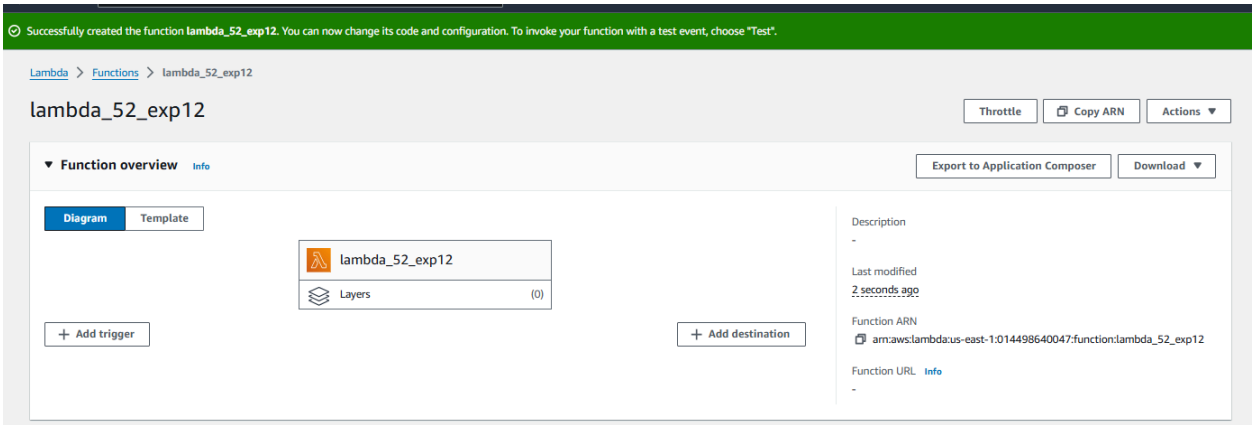
3. Search and Open lambda console and click on create function button.



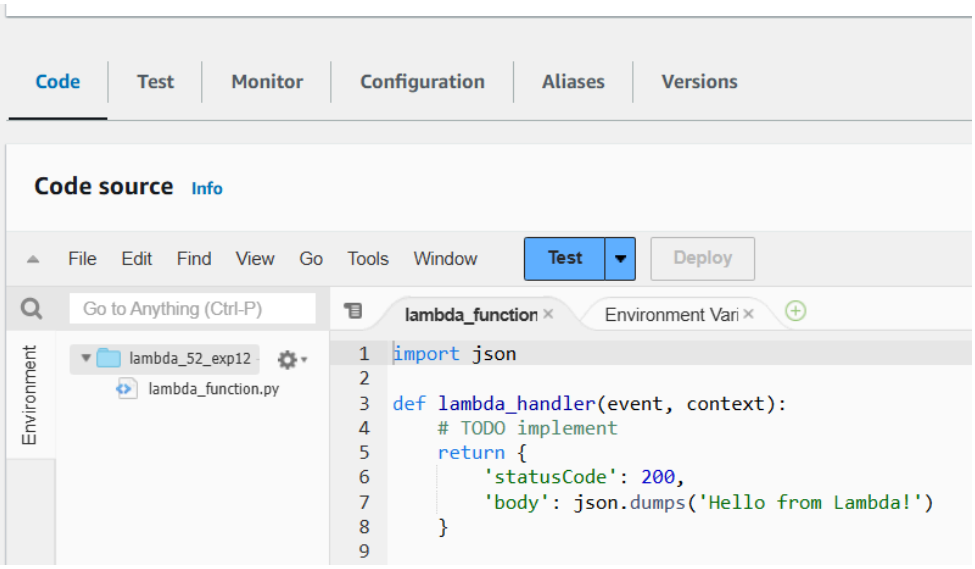
4. Now Give a name to your Lambda function, Select the language to write your function.Here I have chosen python3.12, Architecture as x86, and Execution role to Create a new role with basic Lambda permissions.Note that the console code editor supports only Node.js, Python, and Ruby



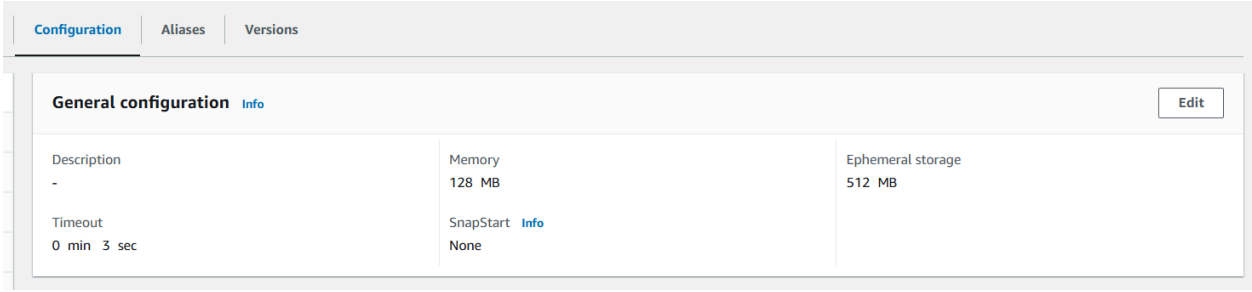
5. The Lambda function was created successfully.



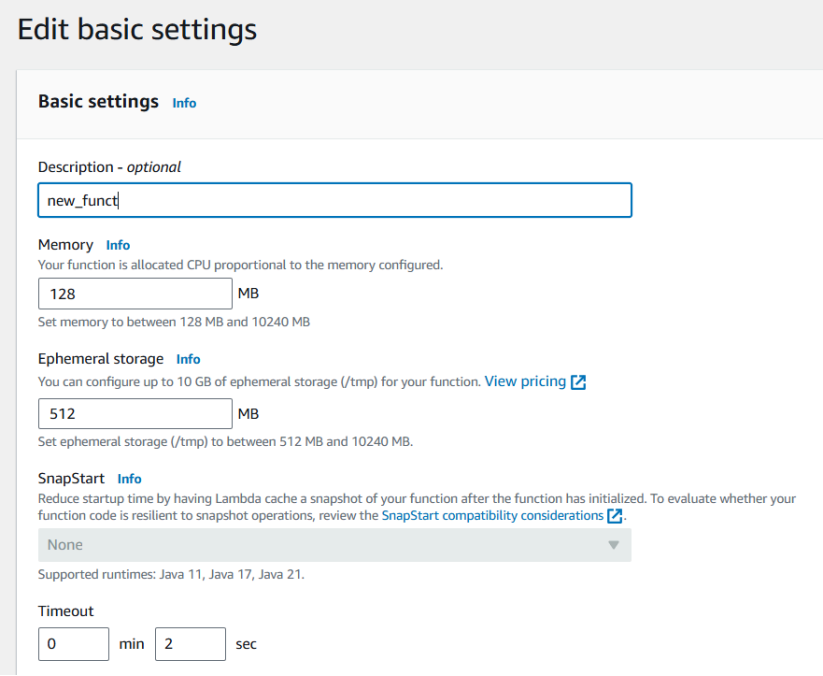
6. Then Go into the code section.You will see some default code there.



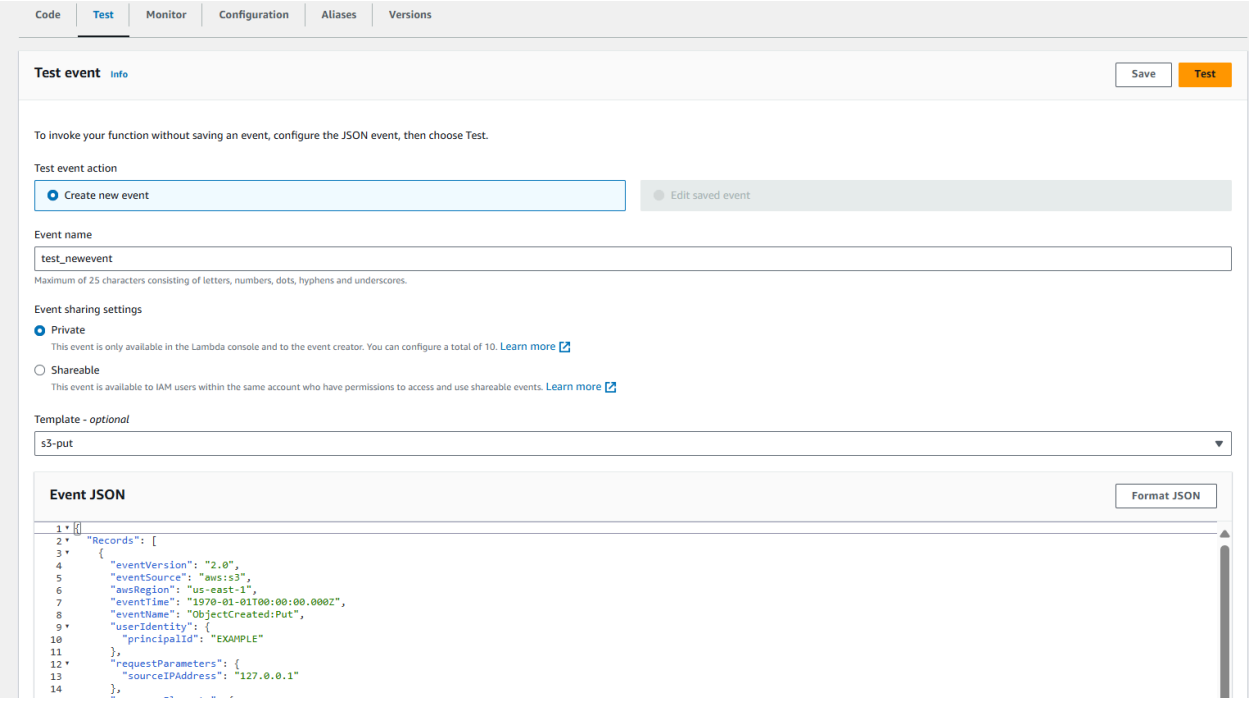
7. To Edit the basic settings go to configuration then click on edit setting.



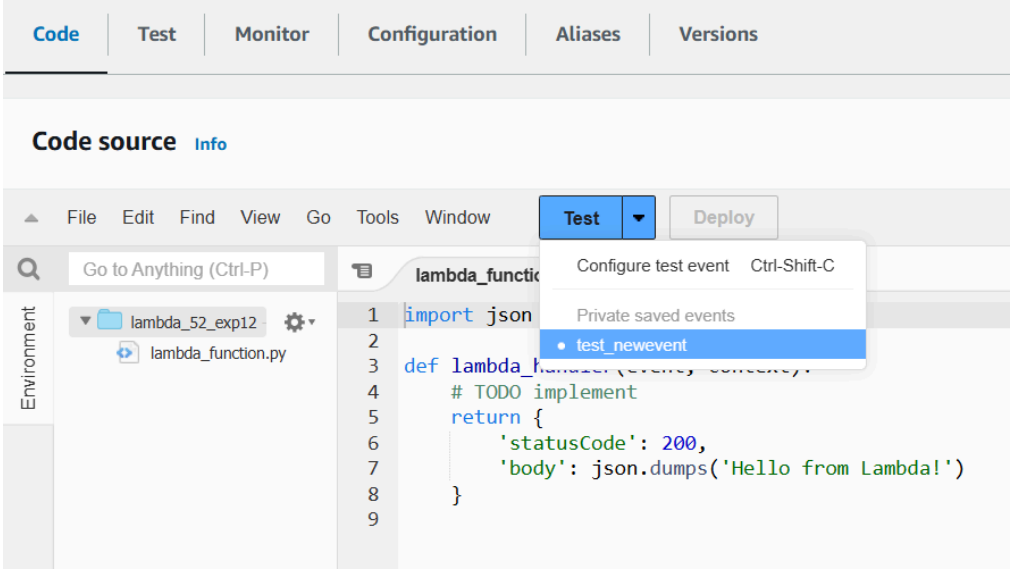
8. Here,enter a description which is optional and change Memory and Timeout. I've changed the Timeout period to 2 sec.



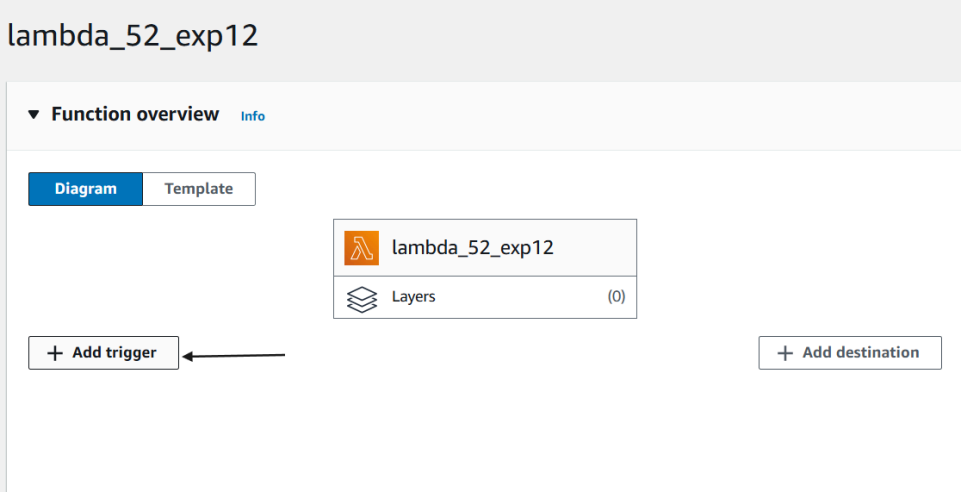
9. Now Click on the Test then select Create a new event, give a name to the event.Here I have given name as 'test_newevent' and then select Event Sharing to private, and select s3 put template.



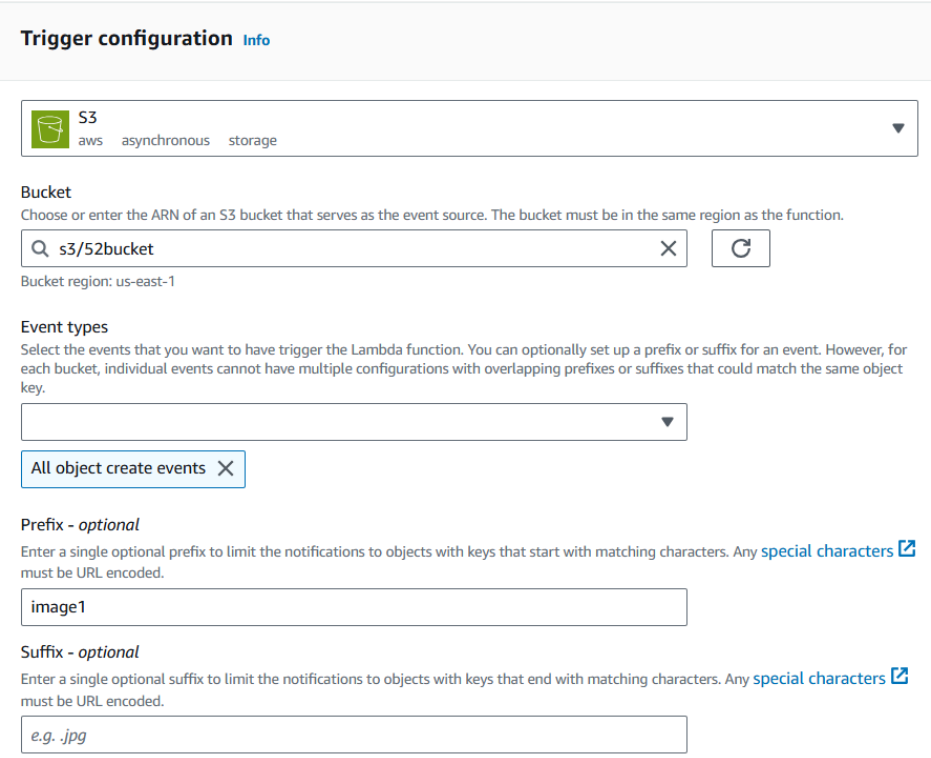
10. Now go to the Code section.Then click on the Test dropdown icon and select the event which we have created now(‘test_newevent’).



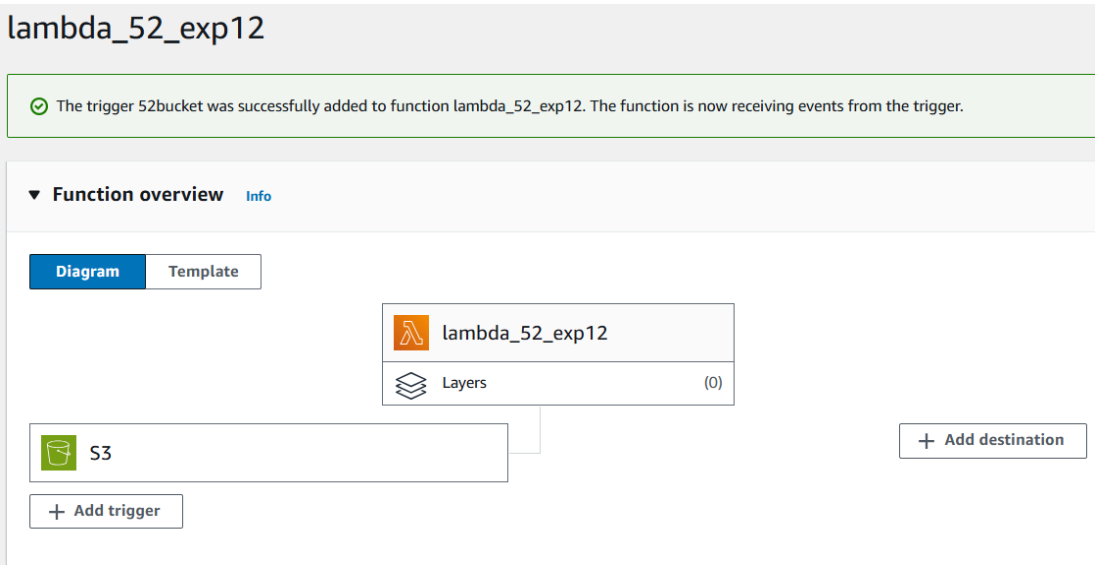
11. Now go into the Lambda function and then click on add tigger.


lambda_52_exp12
Function overview
Info
Diagram
Template
lambda_52_exp12
Layers
(0)
Add trigger
Add destination

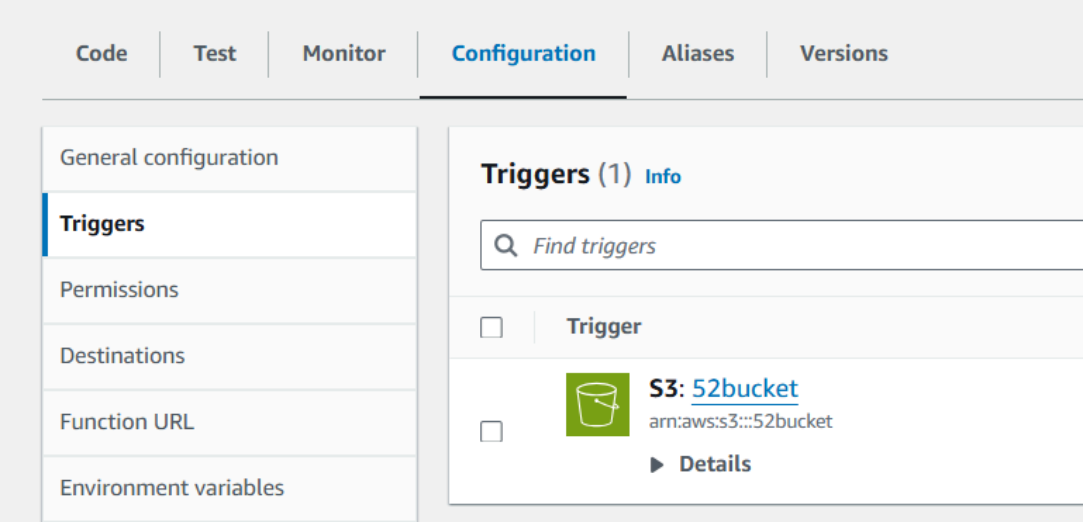
12. Now in the Trigger information.Select the source as S3.Then select the bucket which we have created now (52bucket{in my case}), keep other things default and also you can add prefix to image.



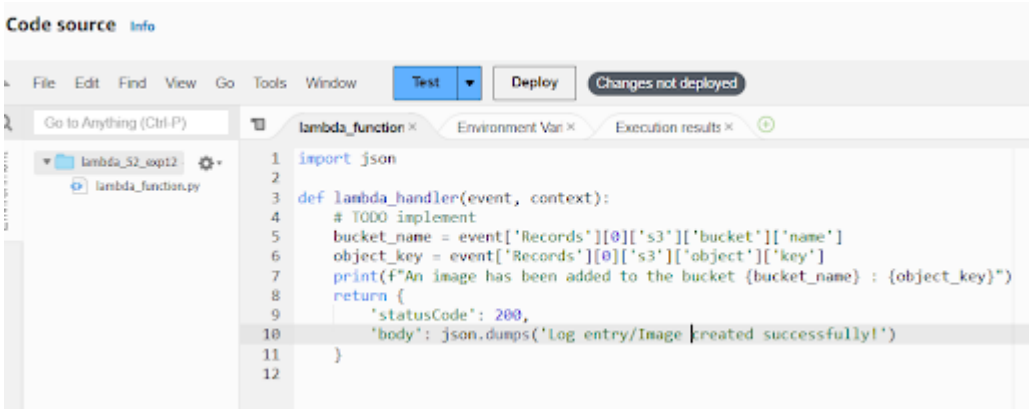
13. Thus, **trigger** is created successfully.



14. You can also check it in the configuration section.

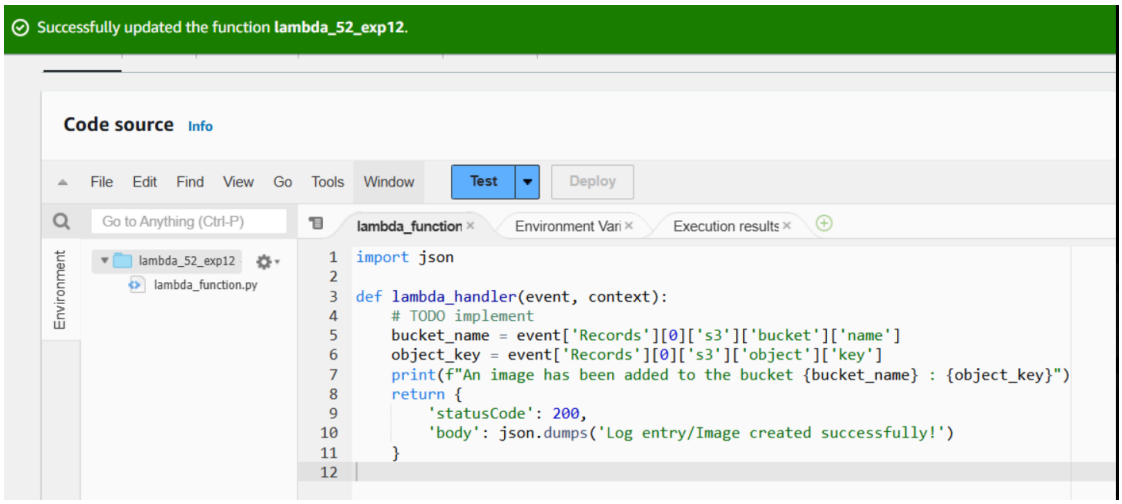


15. Now write a code which logs a message "Log entry/image created successfully" when triggered.

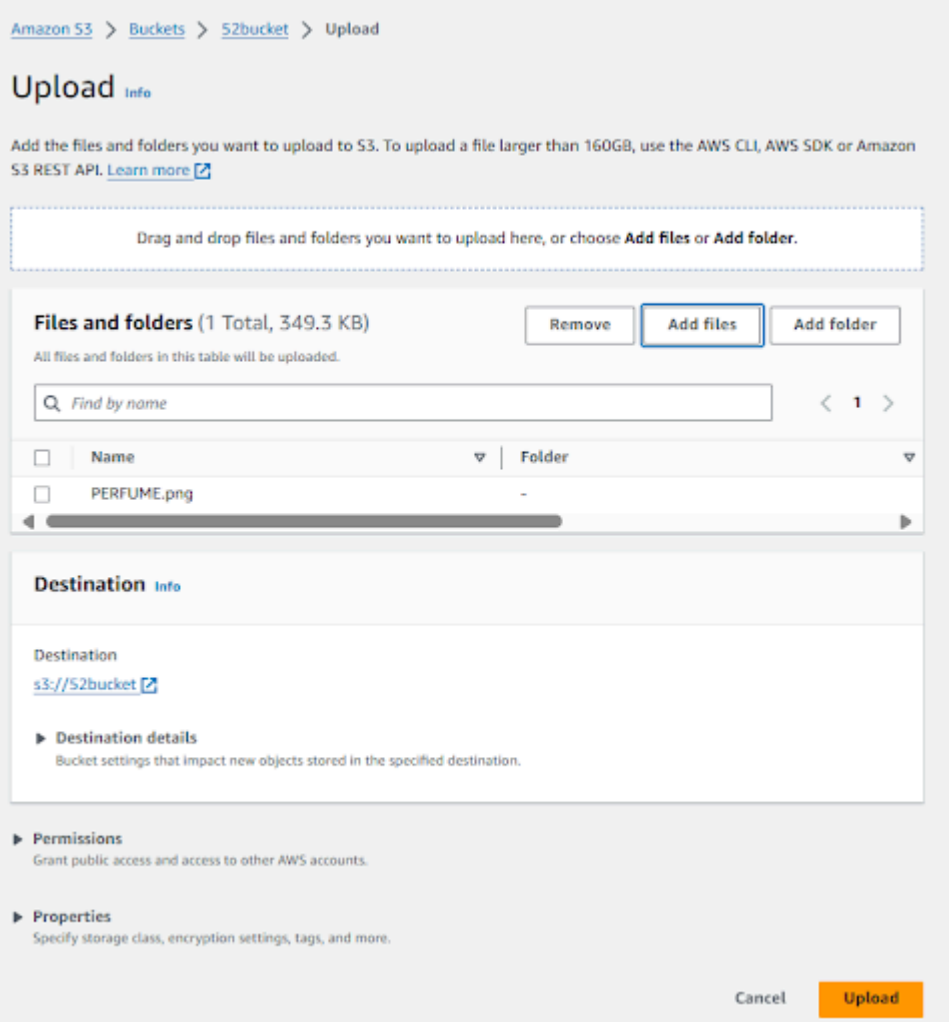


Here changes are not deployed.

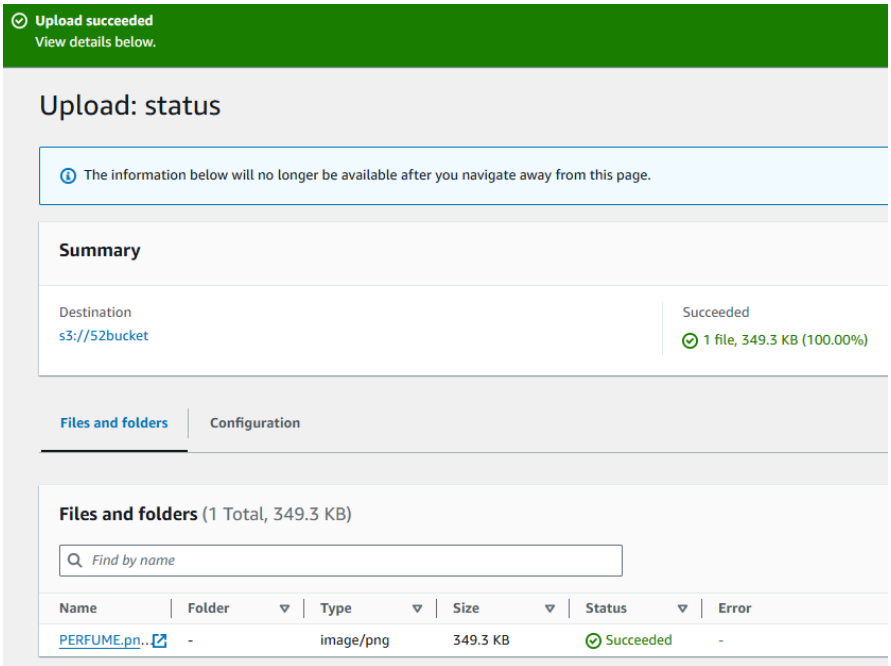
16. So now,Save the file by ctrl+s and then click on deploy.



17. Go to S3 bucket,and there upload **any image** to the bucket.



18. Thus the image was uploaded successfully



19. Now goto lambda function.Then click on test.This will give you log about the image that we have uploaded in S3 bucket.

lambda_function × Environment Vari × Execution result: ×

Execution results

Test Event Name

```
test_newevent
```

Response

```
{
  "statusCode": 200,
  "body": "\"Log entry/Image created successfully!\""
}
```

Function Logs

```
START RequestId: ece2072a-bb6a-4cc3-9065-2594e7d22abc Version: $LATEST
An image has been added to the bucket example-bucket : test%2Fkey
END RequestId: ece2072a-bb6a-4cc3-9065-2594e7d22abc
REPORT RequestId: ece2072a-bb6a-4cc3-9065-2594e7d22abc  Duration: 1.65 ms   Billed Duration: 2 ms   Memory Size: 128 MB Max Memory Used: 32 MB
```

Request ID

```
ece2072a-bb6a-4cc3-9065-2594e7d22abc
```

(In response, It gives status 200 and also the message "Log entry/image created successfully" and also contains function Logs)

20. Now go to cloudwatch.Then go into log groups.Inside that you will get the lambda function name that we have created click on it. Here, you will get a detailed log of events.

CloudWatch > Log groups > /aws/lambda/lambda_52_exp12 > 2024/10/10/[$LATEST]a153468b1a574e0a8512ea7220531432

Log events

Actions ▾ | Start tailing | Create metric filter

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

Filter events - press enter to search | Clear | 1m | 30m | 1h | 12h | Custom | UTC timezone ▾ | Display ▾

Timestamp	Message
	No older events at this moment. Retry
2024-10-10T13:25:24.595Z	INIT_START Runtime Version: python:3.12.v36 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:188d9ca2e2714ff5637bd2bbe06ceb81ec3bc408a0f277dab104c14cd814b081
2024-10-10T13:25:24.688Z	START RequestId: 5072b72b-1982-4ec5-b1c4-dde4705cd4a7 Version: $LATEST
2024-10-10T13:25:24.689Z	An image has been added to the bucket example-bucket : test%2Fkey
2024-10-10T13:25:24.704Z	END RequestId: 5072b72b-1982-4ec5-b1c4-dde4705cd4a7
2024-10-10T13:25:24.704Z	REPORT RequestId: 5072b72b-1982-4ec5-b1c4-dde4705cd4a7 Duration: 1.97 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 32 MB Init Duration: 90.56 ms
2024-10-10T13:26:11.981Z	START RequestId: 80470e6d-fce1-48af-ab9a-42ca8000ae4b Version: $LATEST
2024-10-10T13:26:11.982Z	An image has been added to the bucket example-bucket : test%2Fkey
2024-10-10T13:26:11.984Z	END RequestId: 80470e6d-fce1-48af-ab9a-42ca8000ae4b
2024-10-10T13:26:11.984Z	REPORT RequestId: 80470e6d-fce1-48af-ab9a-42ca8000ae4b Duration: 1.48 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 32 MB
2024-10-10T13:26:31.220Z	START RequestId: ece2072a-bb6a-4cc3-9065-2594e7d22abc Version: $LATEST
2024-10-10T13:26:31.221Z	An image has been added to the bucket example-bucket : test%2Fkey
2024-10-10T13:26:31.224Z	END RequestId: ece2072a-bb6a-4cc3-9065-2594e7d22abc
2024-10-10T13:26:31.224Z	REPORT RequestId: ece2072a-bb6a-4cc3-9065-2594e7d22abc Duration: 1.65 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 32 MB
	No newer events at this moment. *Auto retry paused.* Resume

Conclusion:

Through this project, I successfully set up a Lambda function and an S3 bucket. After configuring the settings, like adding a description and setting a 1-second timeout, I created a test event named 'test_newevent' and deployed the function without any issues. I also linked the Lambda function to the S3 bucket via a trigger and added a print statement in the code. After uploading an image to the bucket and redeploying, the function returned a status code of 200 with the expected message, and CloudWatch logs captured the entire process.This practical taught me how to effectively manage AWS Lambda settings, set up and test triggers, and monitor logs using CloudWatch. I also gained a deeper understanding of integrating S3 with Lambda to automate tasks.