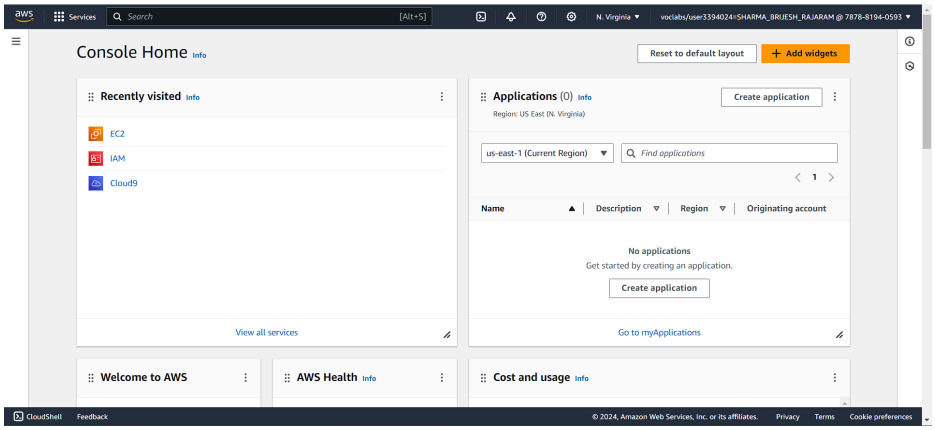
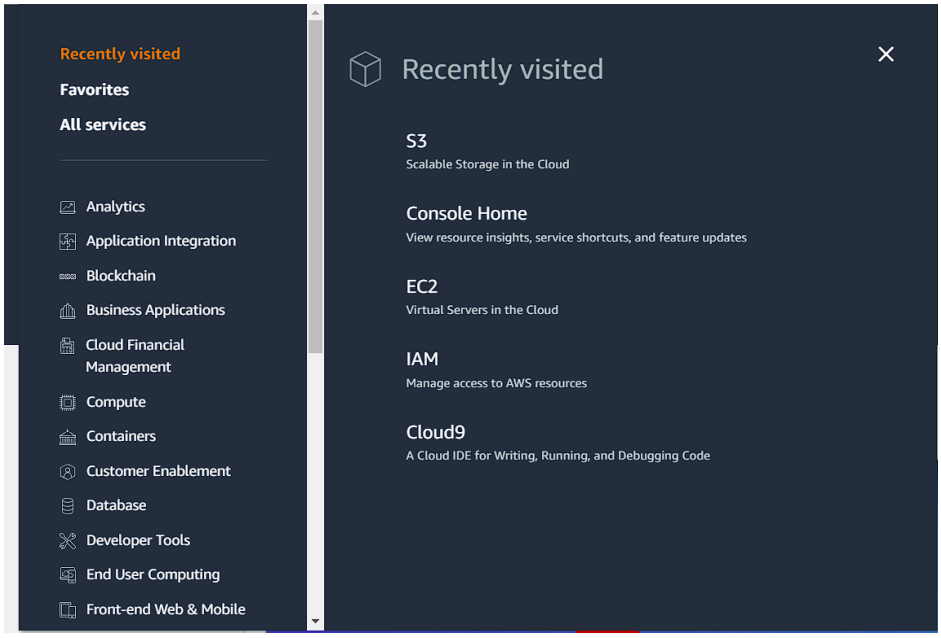
Hosting a static website on Amazon Web Services (S3)

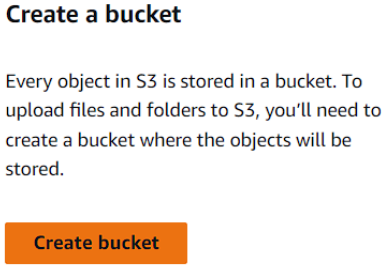
1) Open the AWS console home



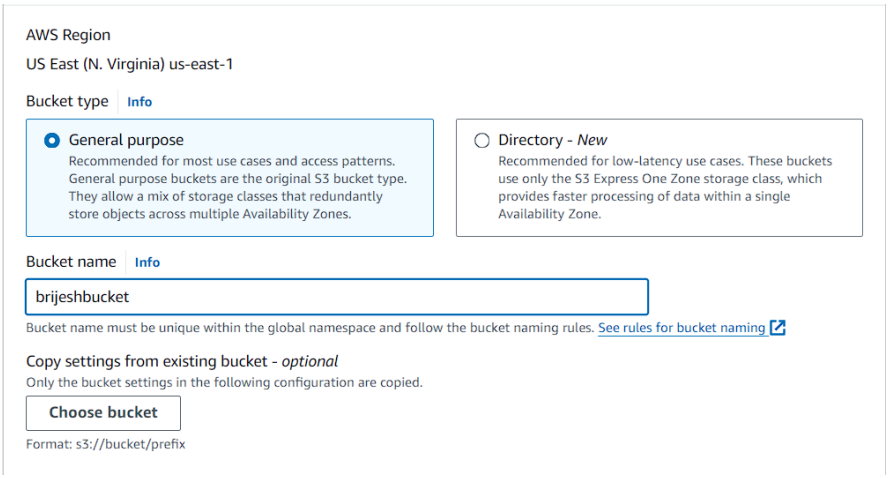
2) Navigate to the S3 to host the website



3) On S3, click on create bucket



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



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

Object Ownership Info

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

- (•) ACLs disabled (recommended)
  All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.
- ( ) ACLs enabled
  Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership

Bucket owner enforced

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

- [ ] **Block all public access**
  Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.
  - [ ] **Block public access to buckets and objects granted through *new* access control lists (ACLs)**
    S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
  - [ ] **Block public access to buckets and objects granted through *any* access control lists (ACLs)**
    S3 will ignore all ACLs that grant public access to buckets and objects.
  - [ ] **Block public access to buckets and objects granted through *new* public bucket or access point policies**
    S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
  - [ ] **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
    S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. Learn more

Bucket Versioning

- (•) Disable
- ( ) Enable

Tags - *optional* (0)

You can use bucket tags to track storage costs and organize buckets. Learn more

No tags associated with this bucket.

Add tag

Default encryption Info

Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type | Info

- (•) Server-side encryption with Amazon S3 managed keys (SSE-S3)
- ( ) Server-side encryption with AWS Key Management Service keys (SSE-KMS)
- ( ) Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)
  Secure your objects with two separate layers of encryption. For details on pricing, see DSSE-KMS pricing on the Storage tab of the Amazon S3 pricing page.

Bucket Key

Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. Learn more

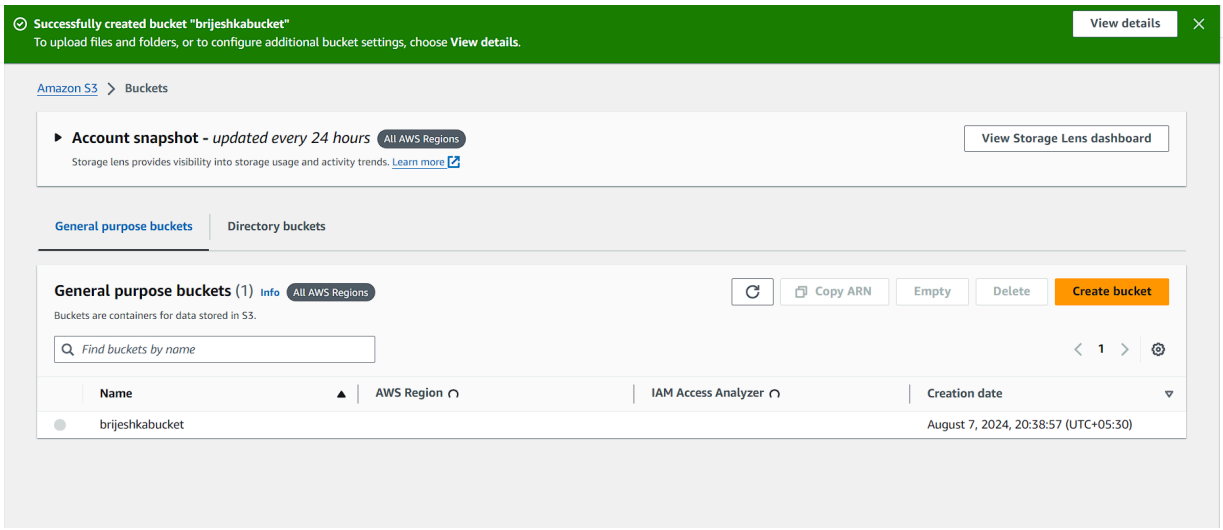
- ( ) Disable
- (•) Enable

▸ **Advanced settings**

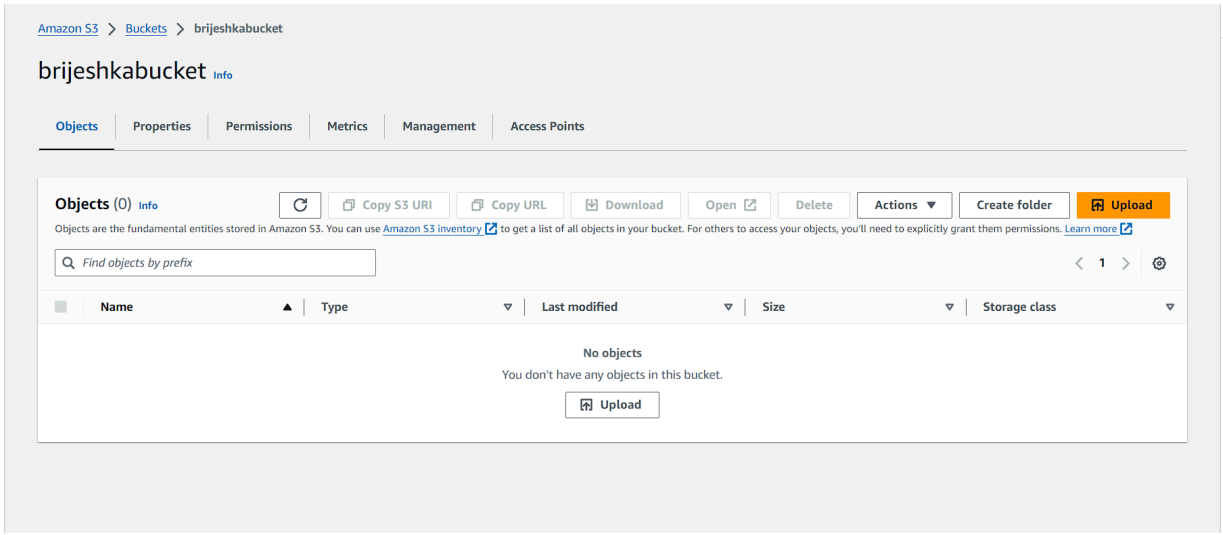
After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

Cancel | Create bucket

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

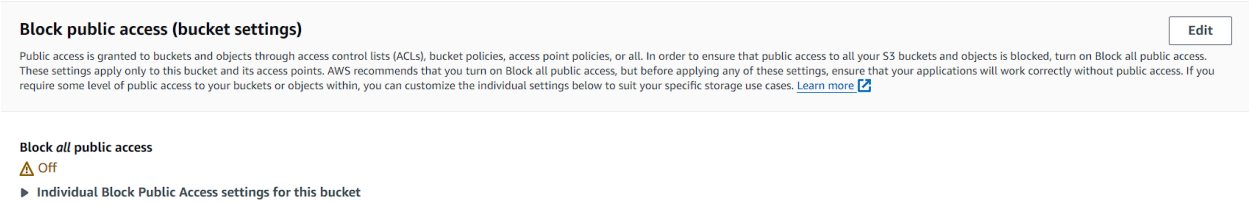


7) Go on Permissions tab and check for Block public access

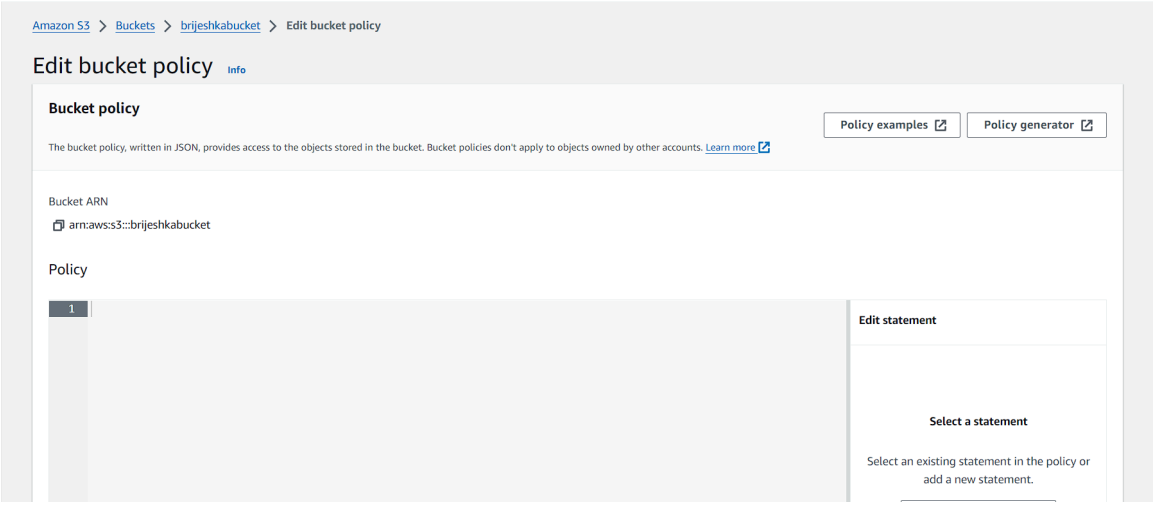


8) Block public access is default on, we need to uncheck it to ensure the hosted website is public.

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



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

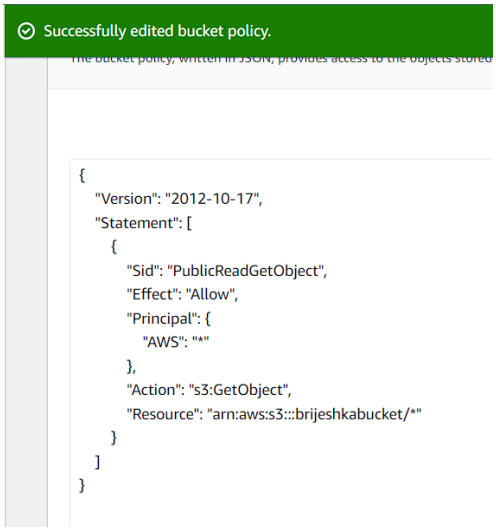


11) Fill the following policy in the empty policy space. Ensure that you change the name of the bucket in Resource with the name of your bucket.

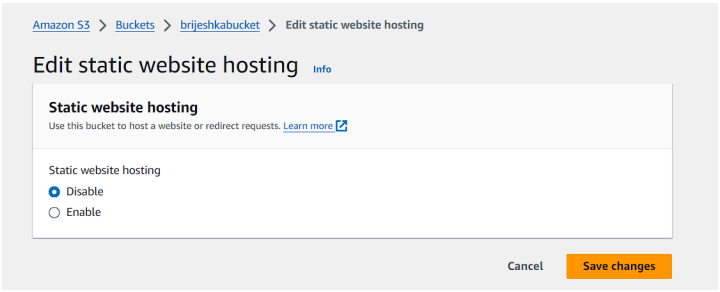
Policy

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

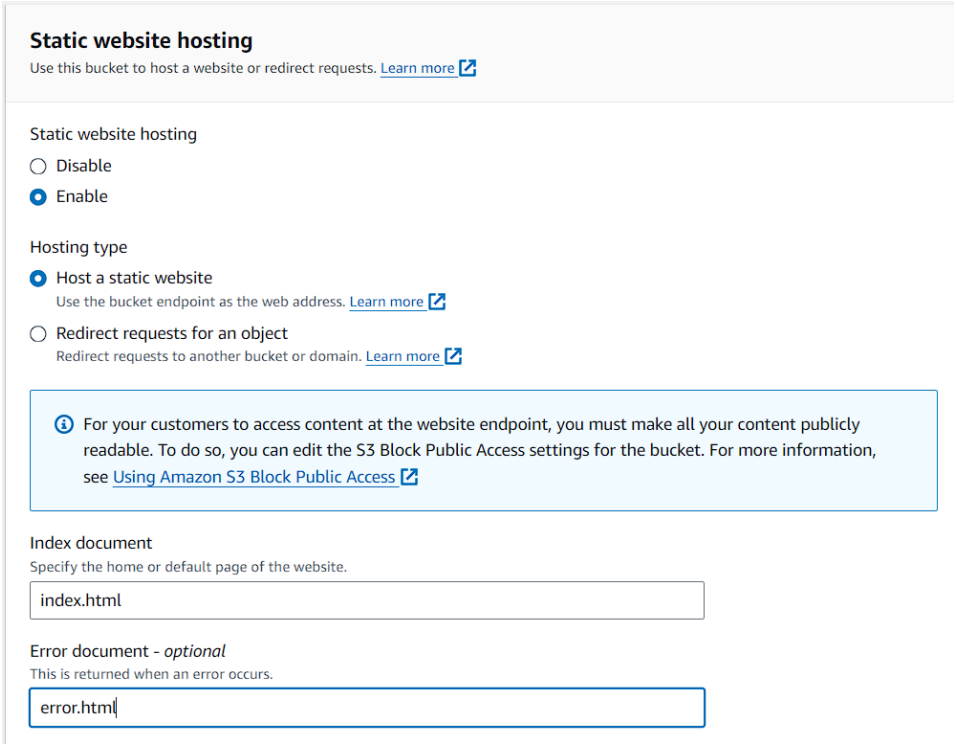
12) After saving the changes, you will see a message.



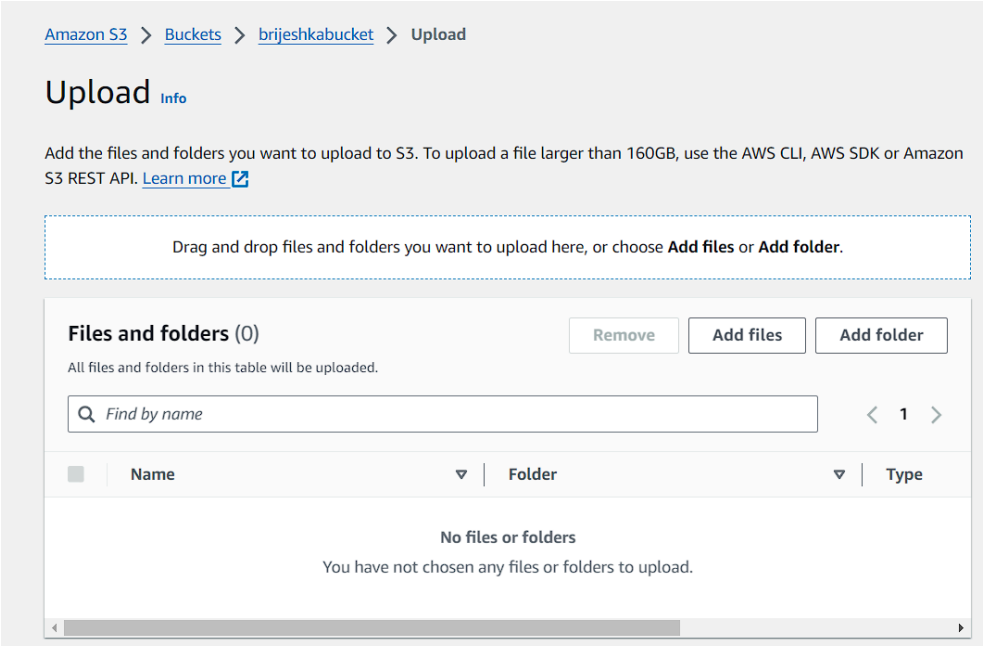
13) Go to the edit static website hosting in the properties tab to use bucket to host websites.



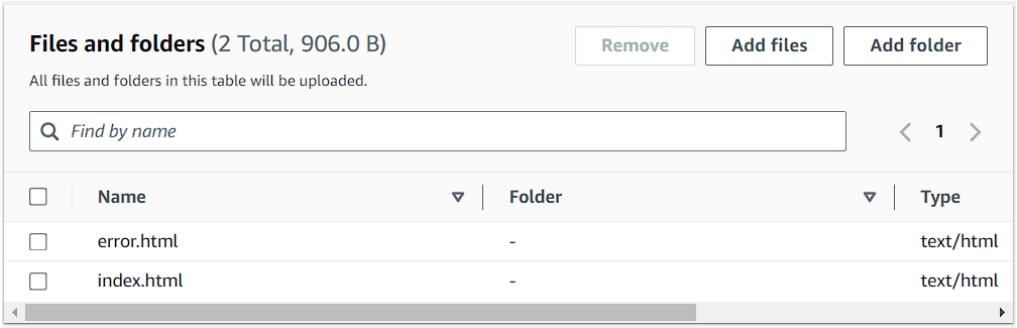
14) Check the options as shown below, and add the names of the file.



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



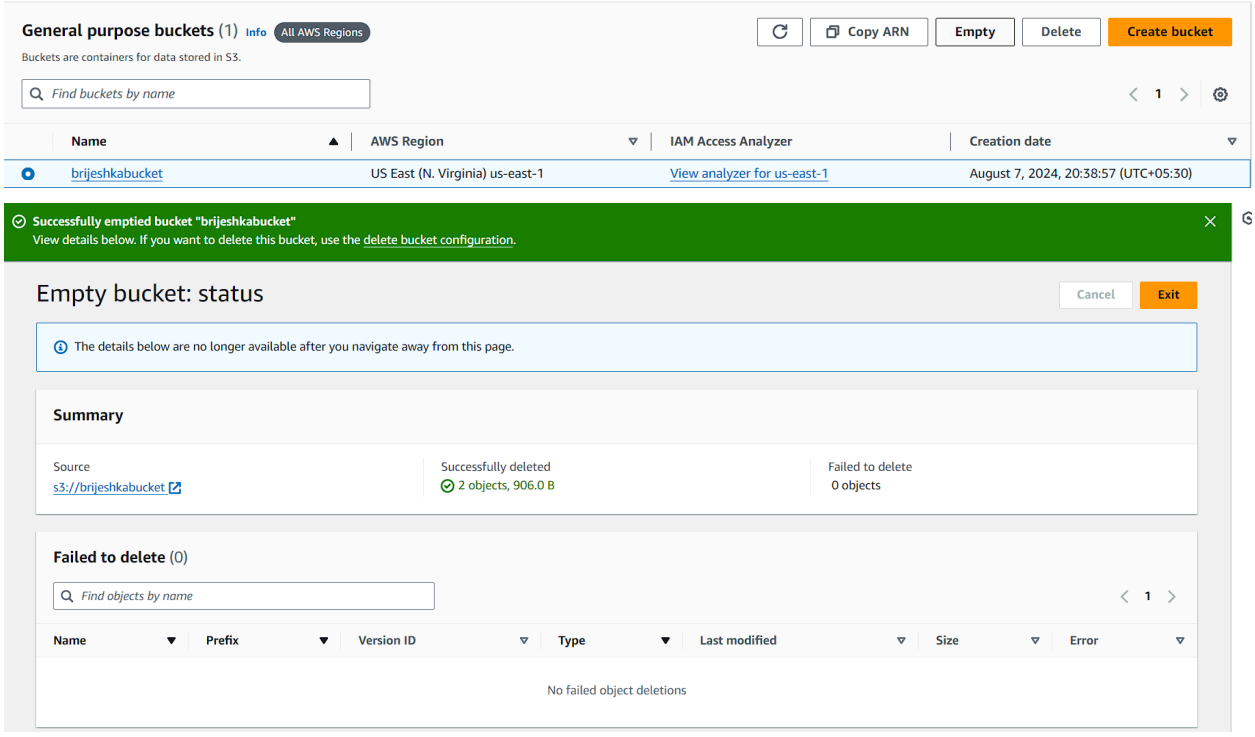
16) Uploaded files will be visible after successful upload.



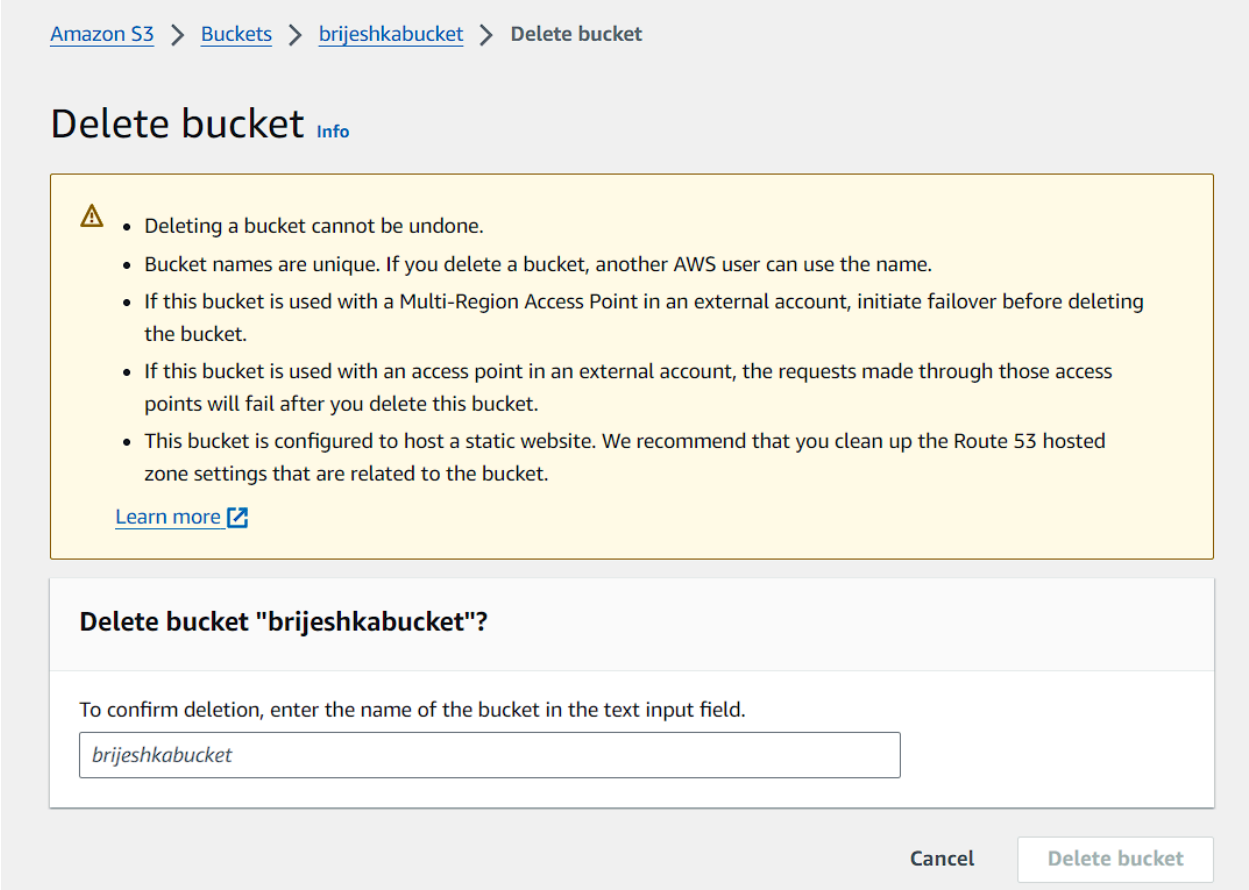
17) Get the link for the hosted website in the properties tab at the bottom.

18) The hosted website using AWS S3.

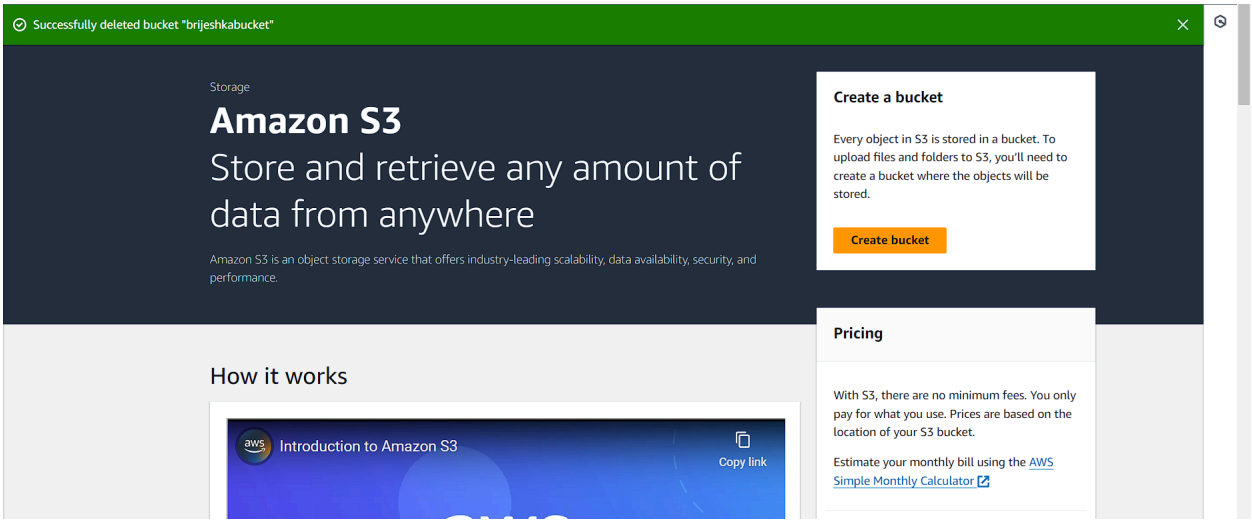
19) To terminate the S3 bucket, first empty the bucket by selecting the files and clicking on Empty.



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

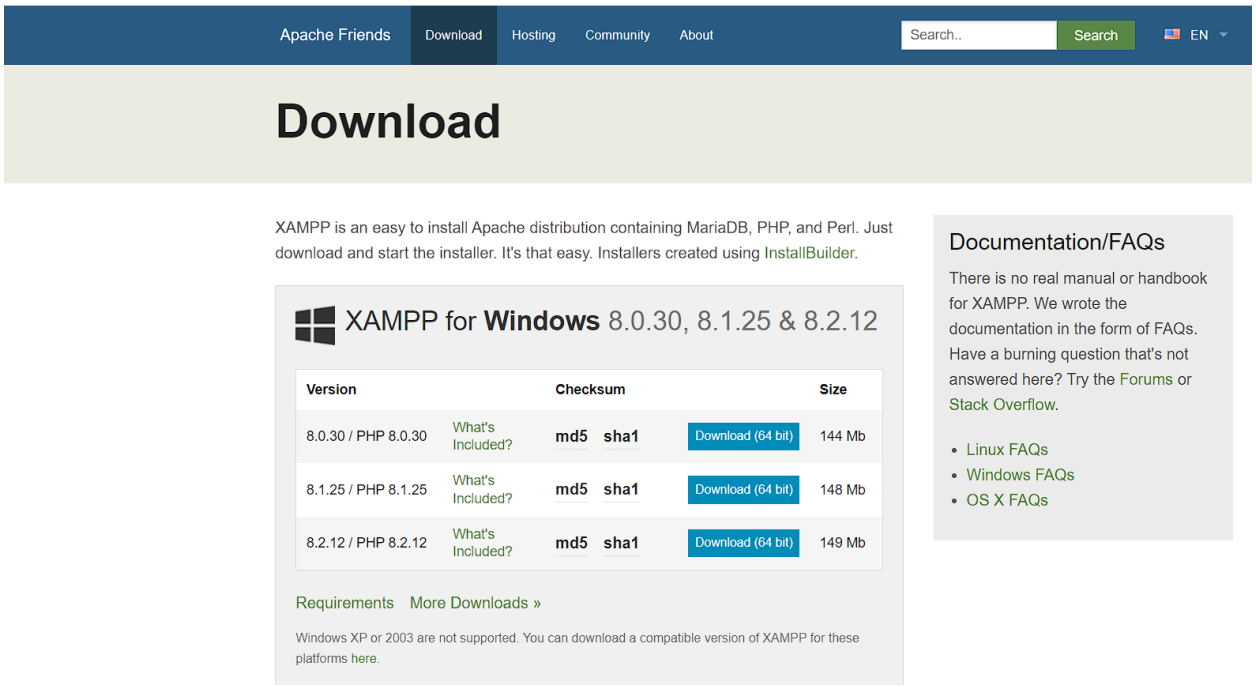


21) After deleting the bucket, a message will appear.

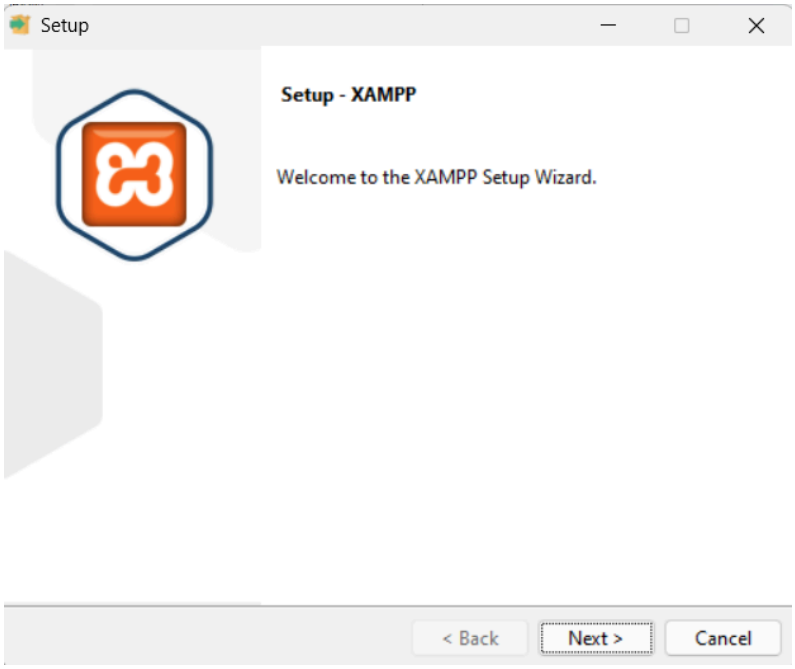


XAMPP Hosting

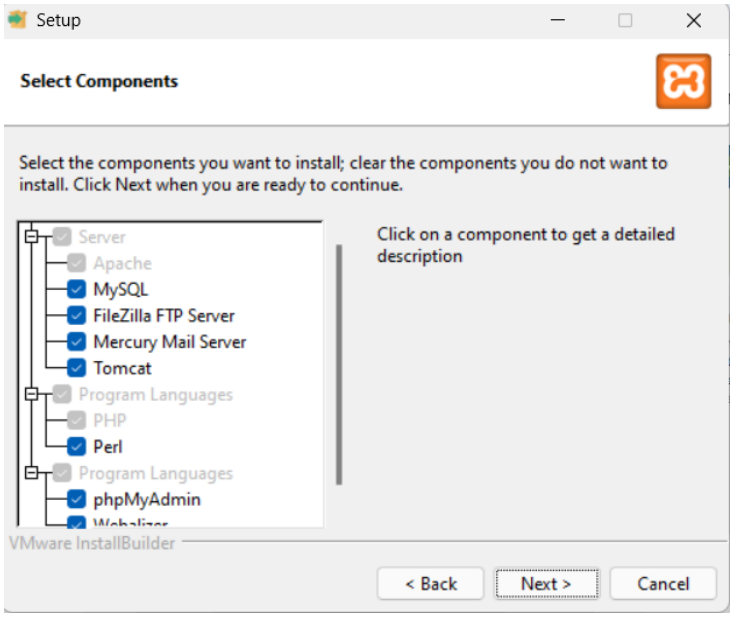
1) Search for XAMPP download and navigate to the xampp official website and click on download as per your system.



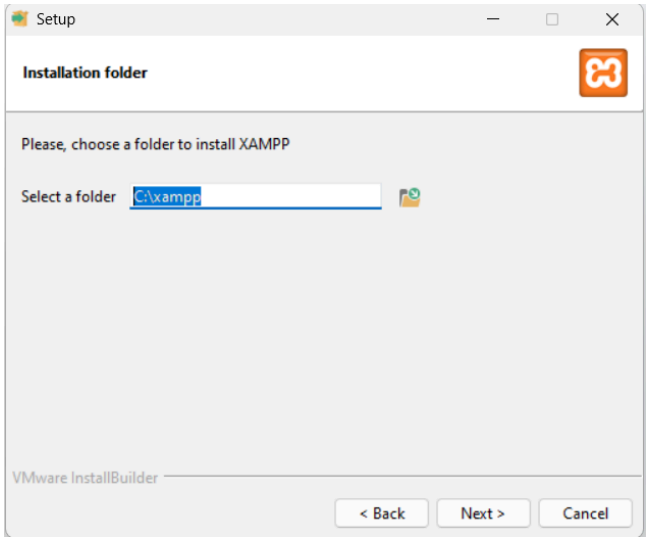
2) Xampp installer will be installed in the system.
3) Window asking to give permission will appear. Click pn ‘Yes’.
4) A window will appear for setup. Click on ‘Next’.



5) Keep the default settings and click on Next.



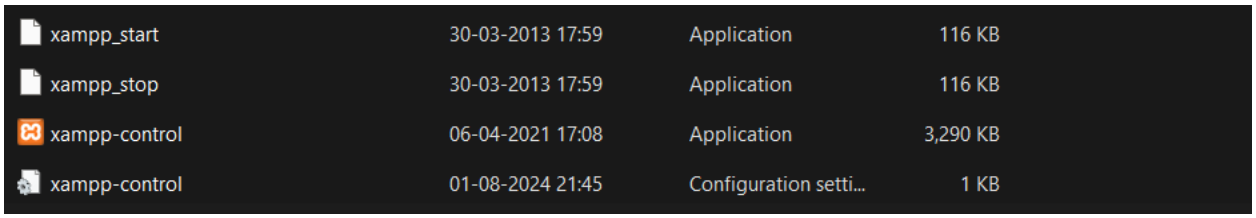
6) Choose the folder path.



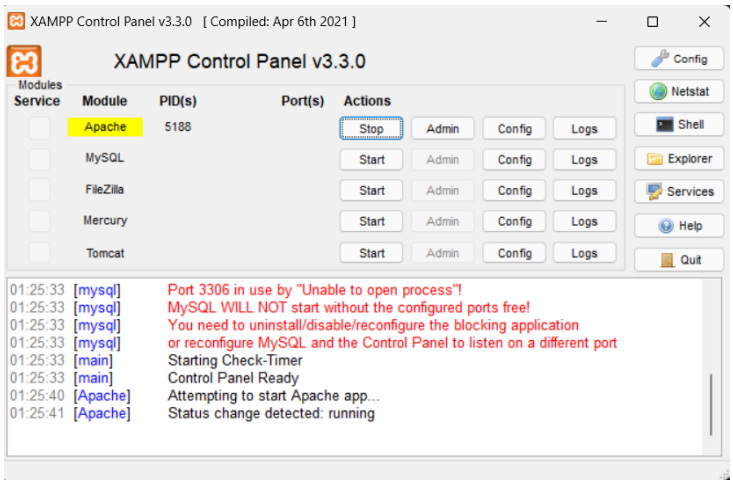
7) Then the XAMPP installation will be done.
8) Locate the folder and then locate the ‘htdocs’ folder in the xampp folder.

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

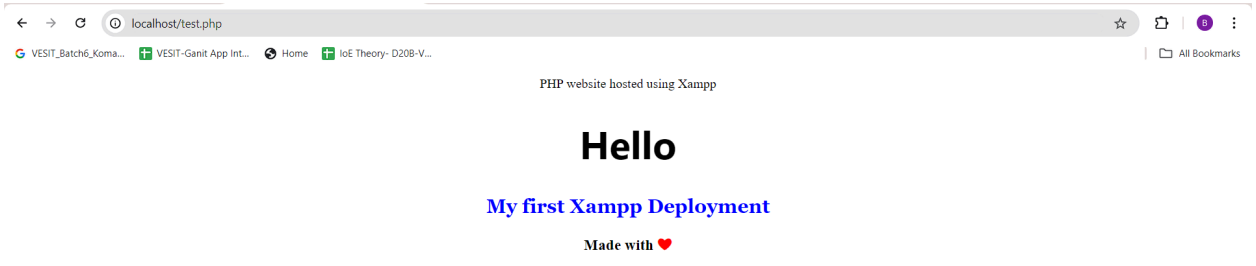
10) Now go to the xampp contol panel in the xampp folder.



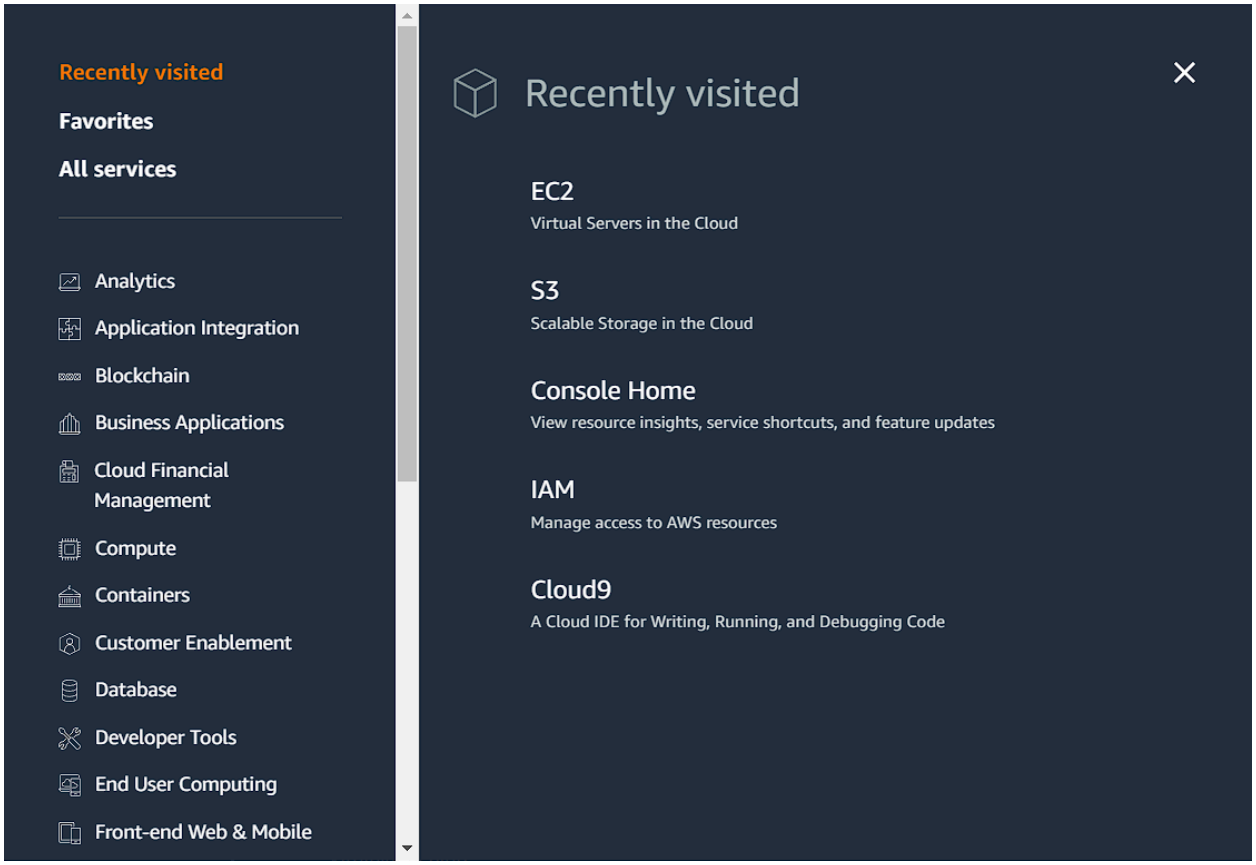
11) Start the Apache server.

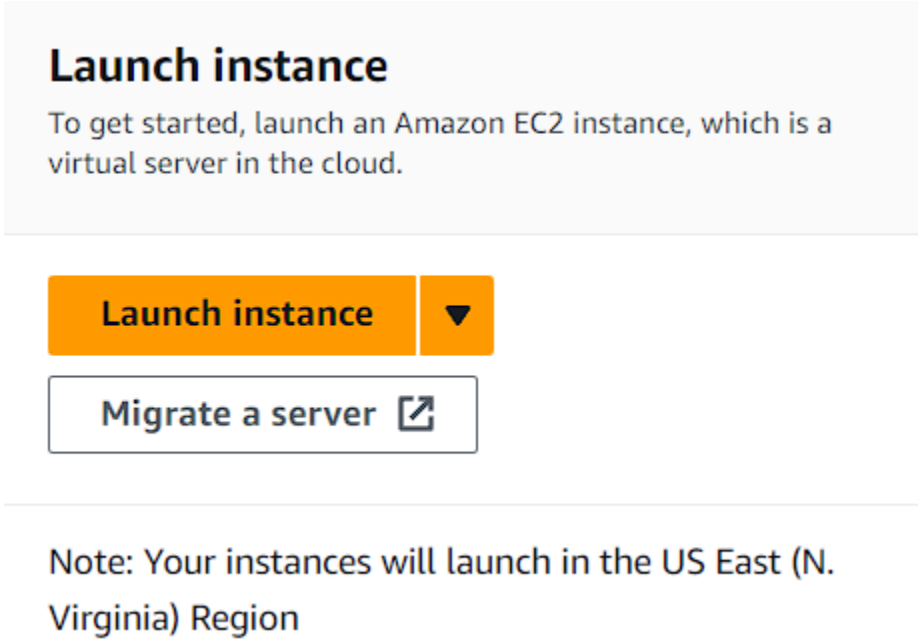


12) After strating the service, got to “localhost/file_name”, then the output window will appear.



EC2 Instance





Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name

AWS Test Server

Add additional tags

Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Recents | Quick Start

Amazon Linux | macOS | Ubuntu | Windows | Red Hat | SUSE Li

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type — Free tier eligible

ami-04a81a99f5ec58529 (64-bit (x86)) / ami-0c14ff330901e49ff (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Architecture: 64-bit (x86)

AMI ID: ami-04a81a99f5ec58529 Verified provider

Instance type Info | Get advice

Instance type

t2.micro — Free tier eligible

Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0162 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour
On-Demand RHEL base pricing: 0.026 USD per Hour
On-Demand Linux base pricing: 0.0116 USD per Hour

All generations

Compare instance types

Additional costs apply for AMIs with pre-installed software

Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

vockey

Create new key pair

Network settings Info

Edit

Network | Info

vpc-0531204c9e29f6332

Subnet | Info

No preference (Default subnet in any availability zone)

Auto-assign public IP | Info

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) | Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

- (•) Create security group
- ( ) Select existing security group

We'll create a new security group called '**launch-wizard-2**' with the following rules:

- [x] Allow SSH traffic from
  Helps you connect to your instance
  Anywhere
  0.0.0.0/0
- [ ] Allow HTTPS traffic from the internet
  To set up an endpoint, for example when creating a web server
- [ ] Allow HTTP traffic from the internet
  To set up an endpoint, for example when creating a web server

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only. ×

Configure storage Info

Advanced

1x 8 GiB gp3 Root volume (Not encrypted)

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage

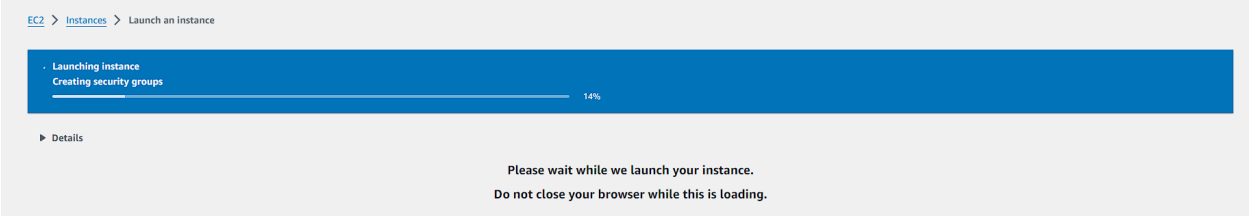
Add new volume

The selected AMI contains more instance store volumes than the instance allows. Only the first 0 instance store volumes from the AMI will be accessible from the instance

Click refresh to view backup information

The tags that you assign determine whether the instance will be backed up by any Data Lifecycle Manager policies.

0 x File systems Edit



EC2 > Instances > i-033e8bf30d3d5ed91 > Connect to instance

Connect to instance Info

Connect to your instance i-033e8bf30d3d5ed91 (AWS Test Server) using any of these options

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

Port 22 (SSH) is open to all IPv4 addresses
Port 22 (SSH) is currently open to all IPv4 addresses, indicated by **0.0.0.0/0** in the inbound rule in your security group. For increased security, consider restricting access to only the EC2 Instance Connect service IP addresses for your Region: 18.206.107.24/29. Learn more.

Instance ID
i-033e8bf30d3d5ed91 (AWS Test Server)

Connection Type

- (selected) Connect using EC2 Instance Connect
  Connect using the EC2 Instance Connect browser-based client, with a public IPv4 address.
- ( ) Connect using EC2 Instance Connect Endpoint
  Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.

Public IP address
35.171.89.89

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

ubuntu

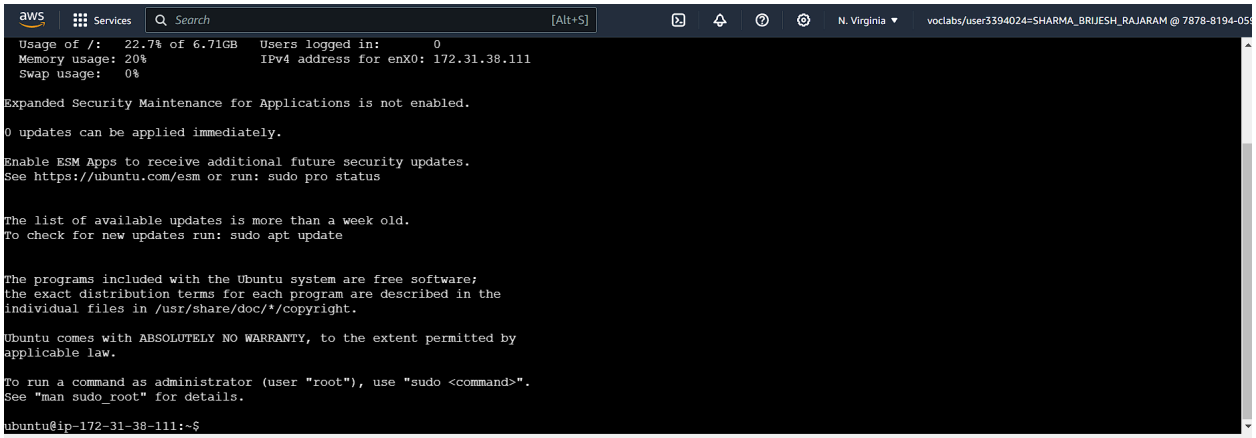
Note: In most cases, the default username, ubuntu, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

You have insufficient IAM permissions to connect to an instance using EC2 Instance Connect
To connect to an instance via EC2 Instance Connect, you must have an attached IAM policy that grants the following permissions:

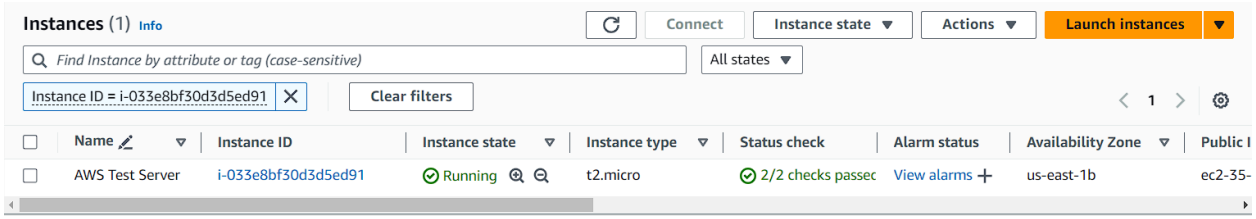
- `ec2-instance-connect:SendSSHPublicKey`
- `ec2:DescribeInstances`

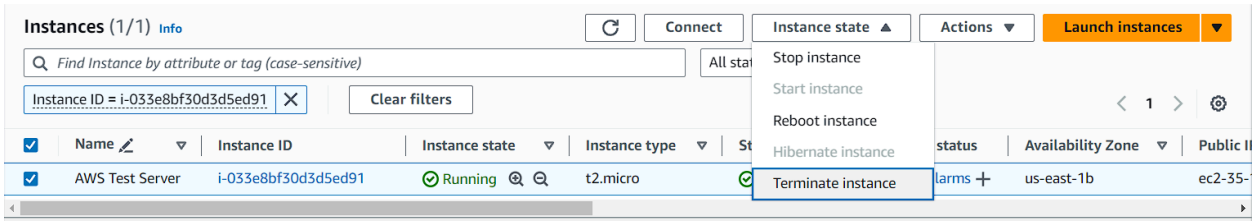
Consider restricting access to specific EC2 instances using `ec2:osuser` condition, or specific resource tag.
Visit IAM Console to verify if you have above permissions.
For more information about IAM policy examples, see Grant IAM permissions for EC2 Instance Connect.

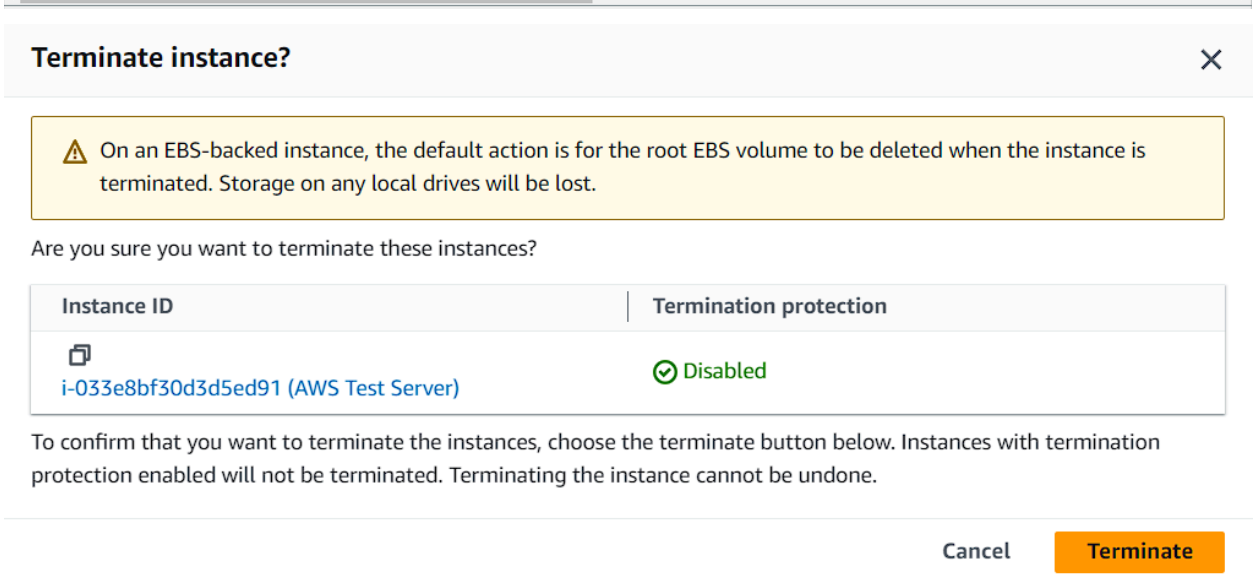
Cancel | Connect

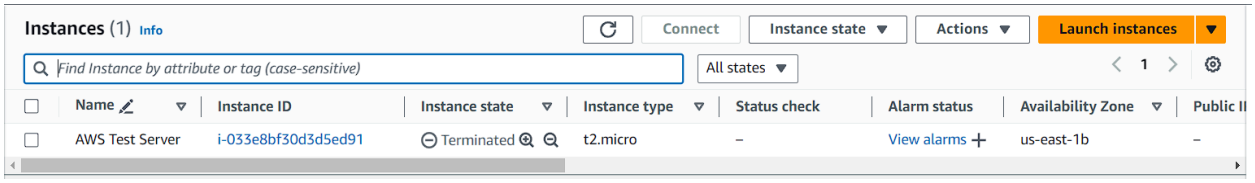



```
ubuntu@ip-172-31-38-111:~$ ping www.google.com
PING www.google.com (142.251.167.106) 56(84) bytes of data.
64 bytes from ww-in-f106.1e100.net (142.251.167.106): icmp_seq=1 ttl=58 time=2.22 ms
64 bytes from ww-in-f106.1e100.net (142.251.167.106): icmp_seq=2 ttl=58 time=2.24 ms
64 bytes from ww-in-f106.1e100.net (142.251.167.106): icmp_seq=3 ttl=58 time=2.26 ms
64 bytes from ww-in-f106.1e100.net (142.251.167.106): icmp_seq=4 ttl=58 time=2.38 ms
64 bytes from ww-in-f106.1e100.net (142.251.167.106): icmp_seq=5 ttl=58 time=2.31 ms
64 bytes from ww-in-f106.1e100.net (142.251.167.106): icmp_seq=6 ttl=58 time=2.26 ms
^C
--- www.google.com ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5007ms
rtt min/avg/max/mdev = 2.220/2.276/2.376/0.052 ms
ubuntu@ip-172-31-38-111:~$ vmstat
procs -----------memory---------- ---swap-- -----io---- -system-- -------cpu-------
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st gu
 2  0      0 504660  18052 288972    0    0   381   298  141    1  2  1 95  1  2  0
ubuntu@ip-172-31-38-111:~$ df
Filesystem     1K-blocks    Used Available Use% Mounted on
/dev/root        7034376 1609612   5408380  23% /
tmpfs             490212       0    490212   0% /dev/shm
tmpfs             196088     868    195220   1% /run
tmpfs               5120       0      5120   0% /run/lock
/dev/xvda16       901520   76972    761420  10% /boot
/dev/xvda15       106832    6246    100586   6% /boot/efi
tmpfs              98040      12     98028   1% /run/user/1000
ubuntu@ip-172-31-38-111:~$ mkdir test
ubuntu@ip-172-31-38-111:~$ touch e.txt
ubuntu@ip-172-31-38-111:~$ ls
e.txt  test
ubuntu@ip-172-31-38-111:~$ history
    1  ping
    2  ping www.google.com
    3  vmstat
    4  df
    5  mkdir test
    6  touch e.txt
    7  ls
    8  history
```


Instances (1) Info
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states
Instance ID = i-033e8bf30d3d5ed91
Clear filters
1
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public I
AWS Test Server
i-033e8bf30d3d5ed91
Running
t2.micro
2/2 checks passec
View alarms
us-east-1b
ec2-35-

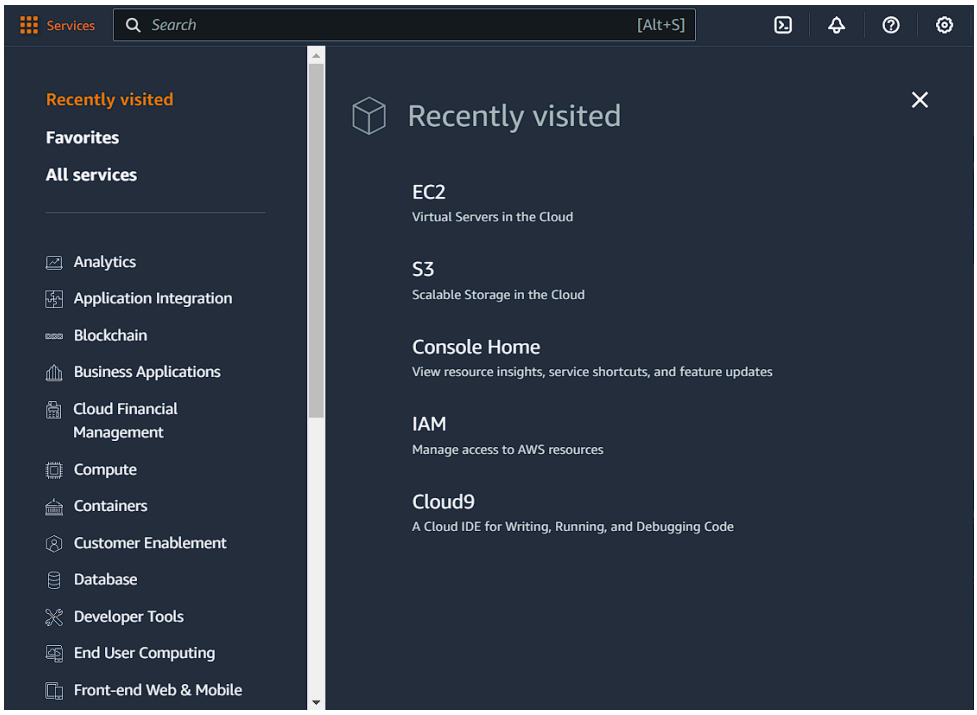

Instances (1/1) Info
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
Instance ID = i-033e8bf30d3d5ed91
Clear filters
1
Stop instance
Start instance
Reboot instance
Hibernate instance
Terminate instance
Name
Instance ID
Instance state
Instance type
Availability Zone
AWS Test Server
i-033e8bf30d3d5ed91
Running
t2.micro
us-east-1b


Terminate instance?
On an EBS-backed instance, the default action is for the root EBS volume to be deleted when the instance is terminated. Storage on any local drives will be lost.
Are you sure you want to terminate these instances?
Instance ID
Termination protection
i-033e8bf30d3d5ed91 (AWS Test Server)
Disabled
To confirm that you want to terminate the instances, choose the terminate button below. Instances with termination protection enabled will not be terminated. Terminating the instance cannot be undone.
Cancel
Terminate

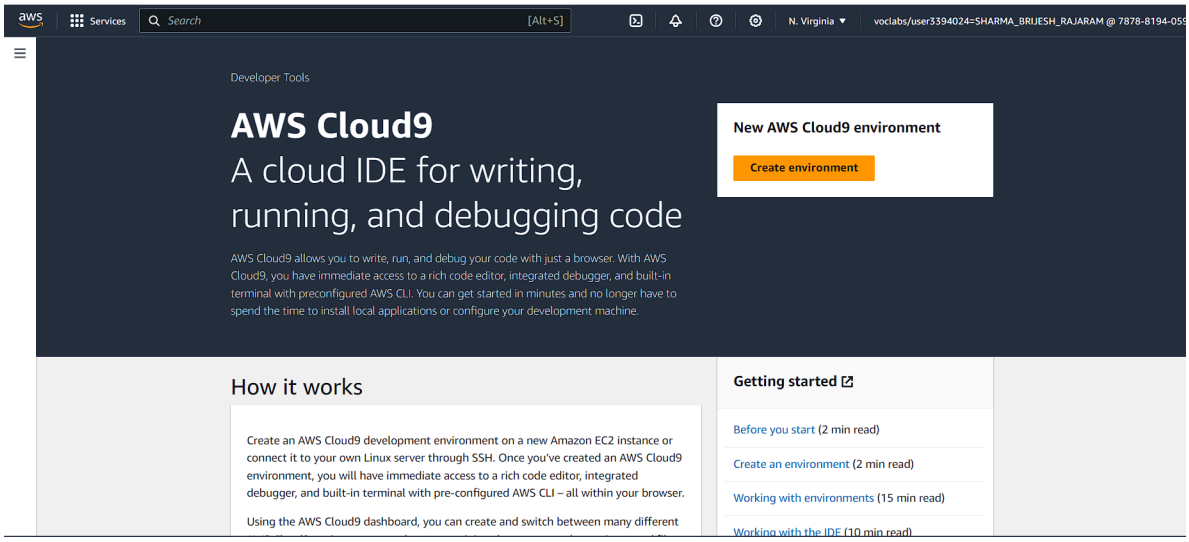

Instances (1) Info
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states
1
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public I
AWS Test Server
i-033e8bf30d3d5ed91
Terminated
t2.micro
View alarms
us-east-1b

Cloud 9 - IDE

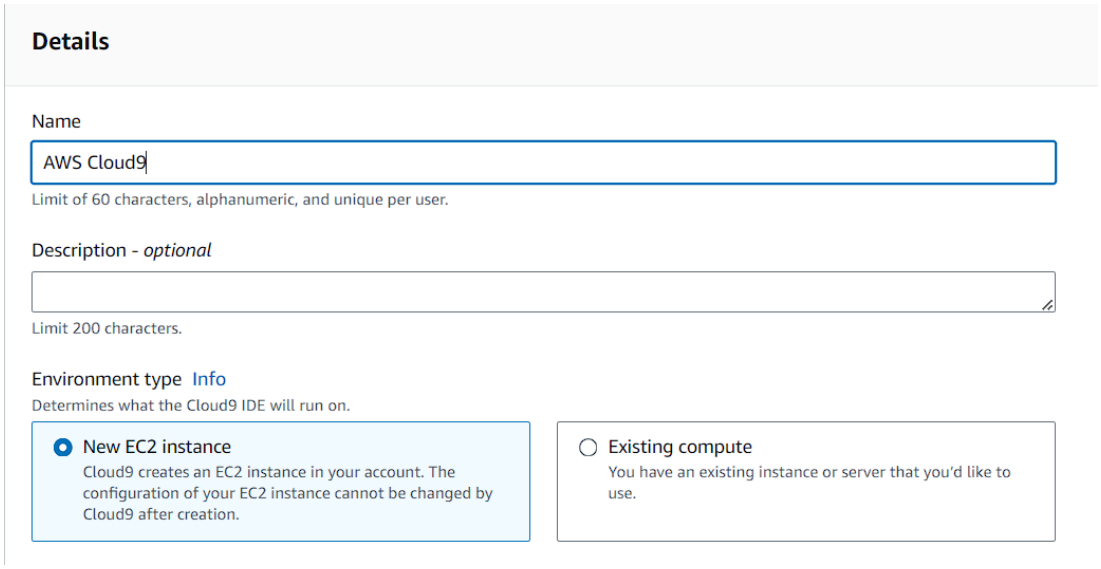
1) Navigate to developer tools -> Cloud9 and start creating Cloud9 environment.



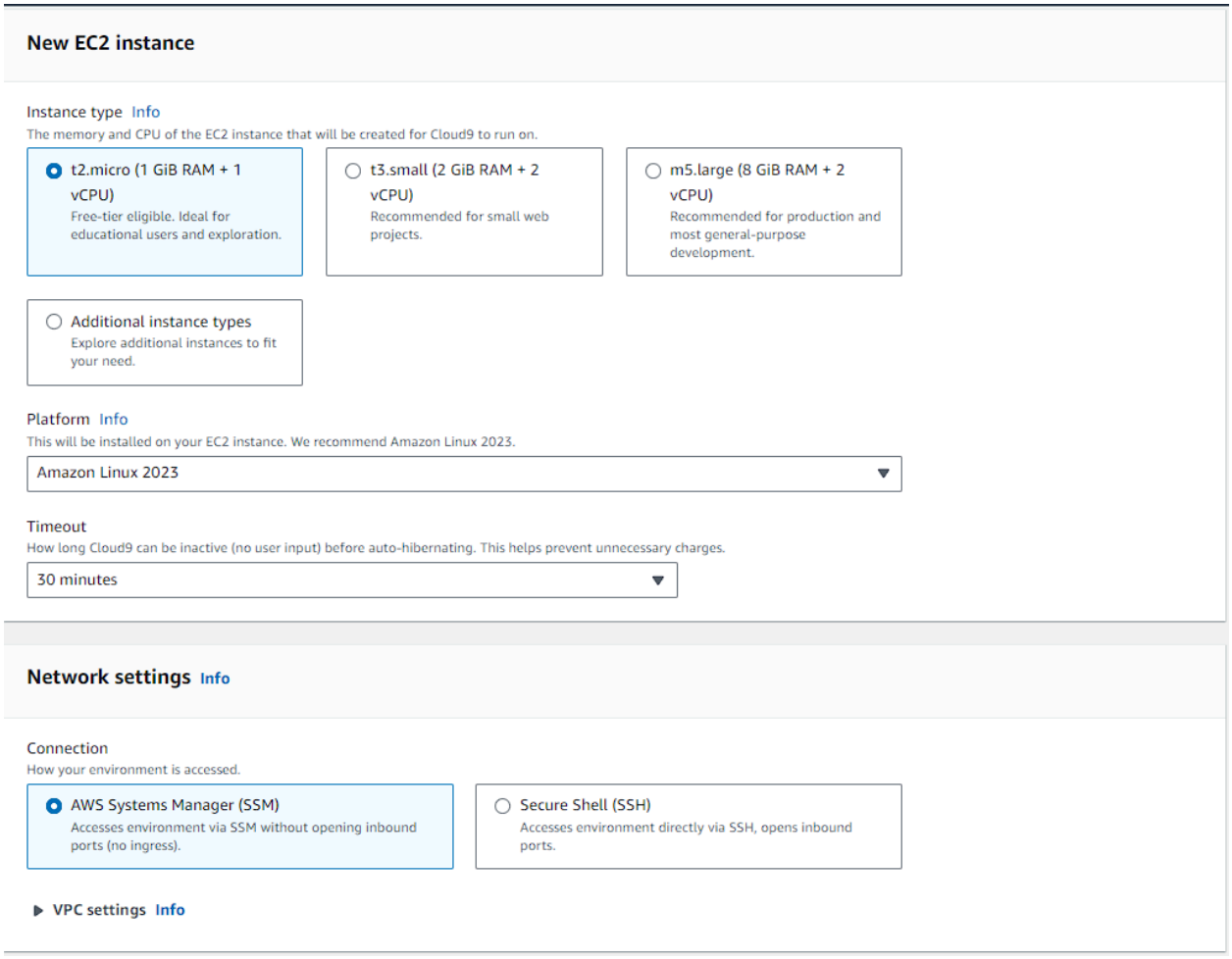
2) Click on Create Enviornment and start creating the environment



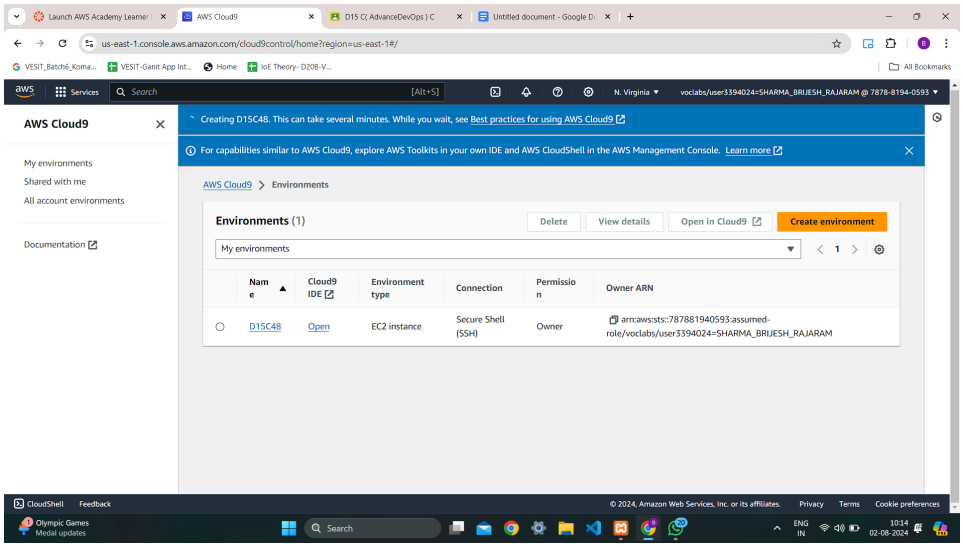
3) Name the environment and select new EC2 instance.

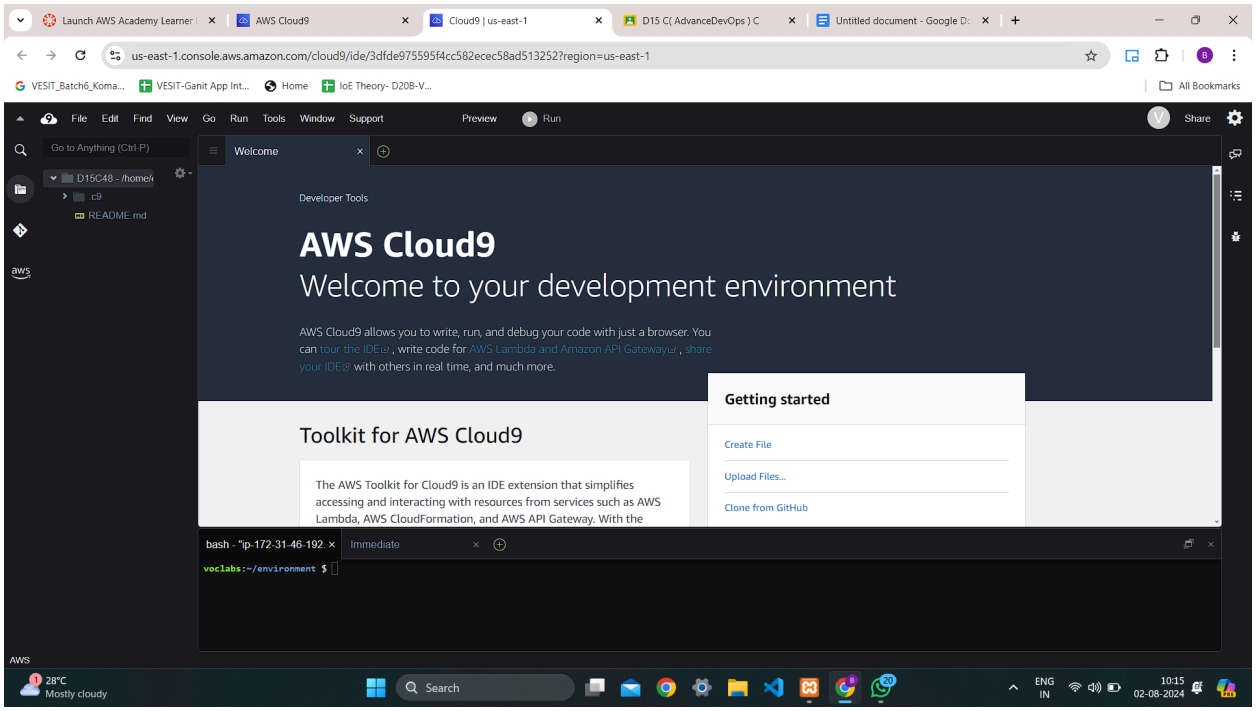


4) Keep the options default and proceed

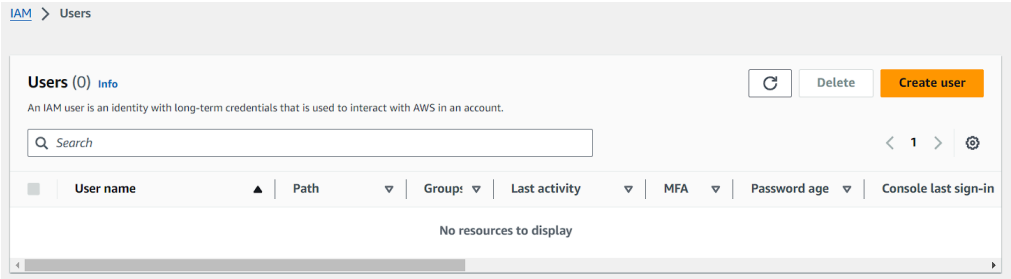


5) Environment created successfully.

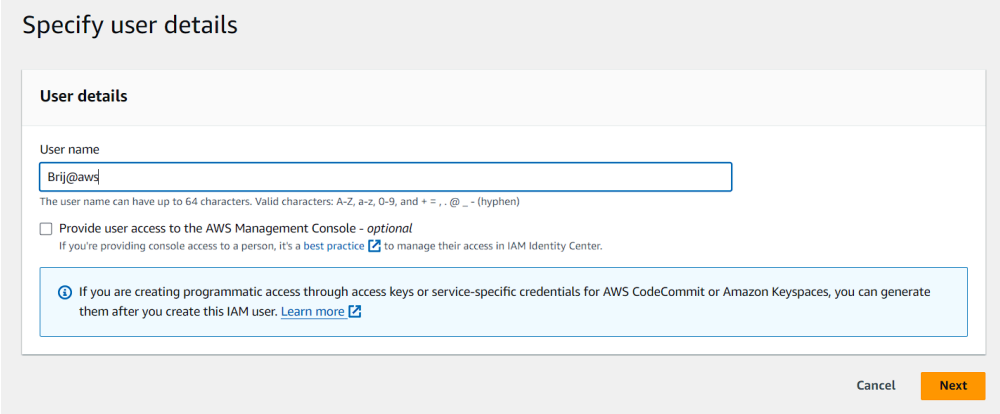




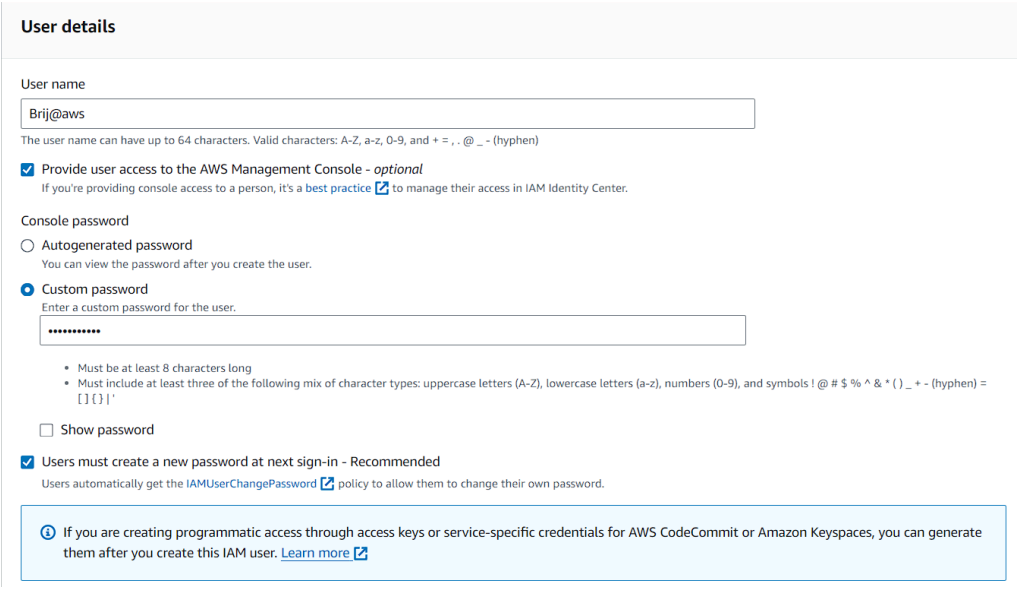
6) Create user using the IAM.



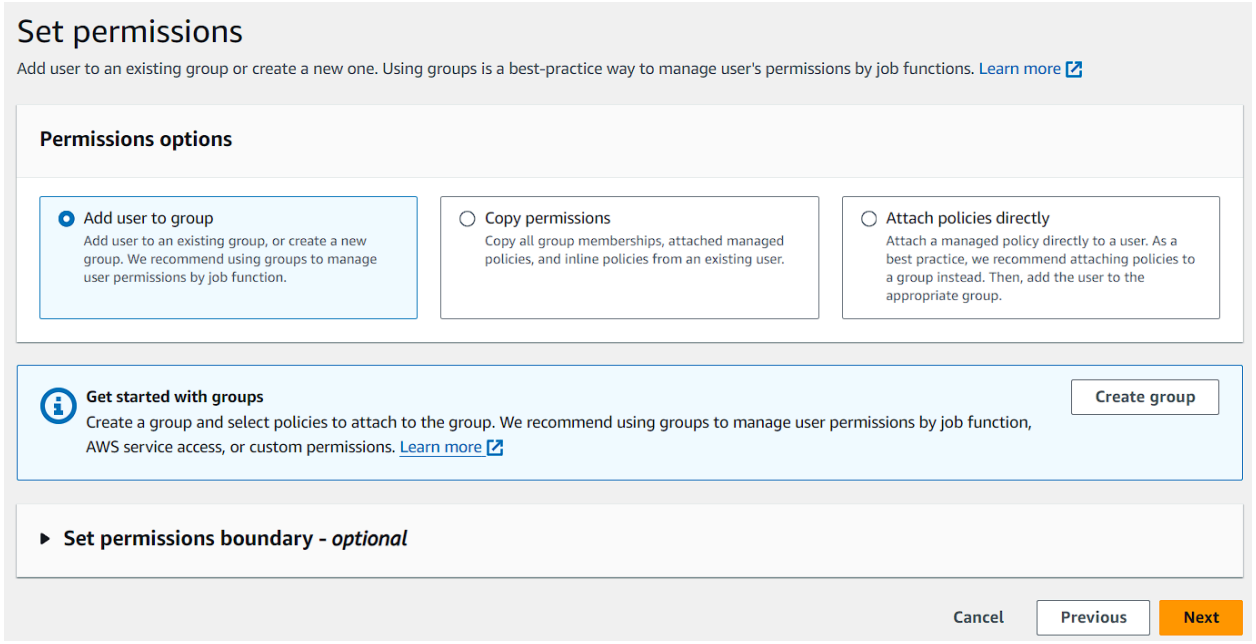
7) Add the username



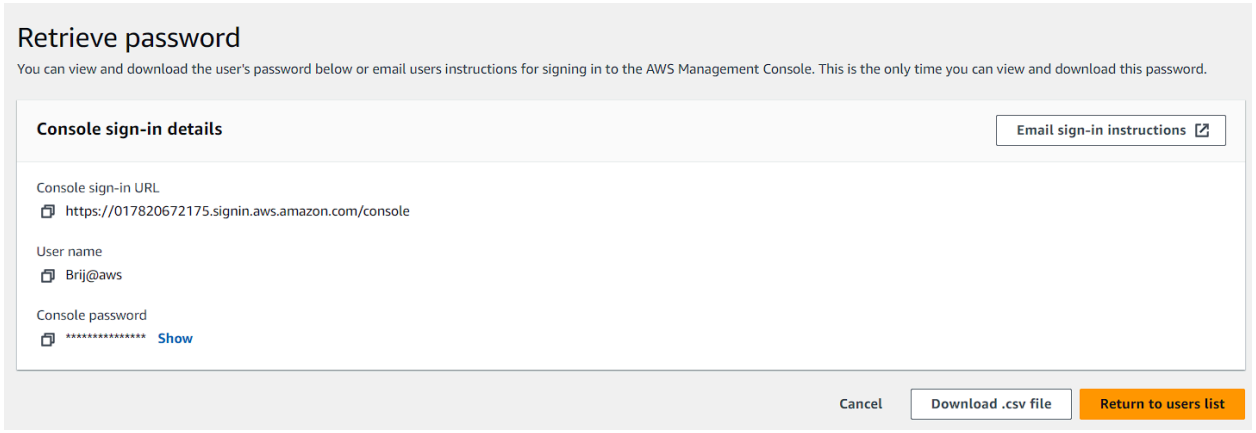
8) Add the remaining user details and provide access to the AWS Management Console



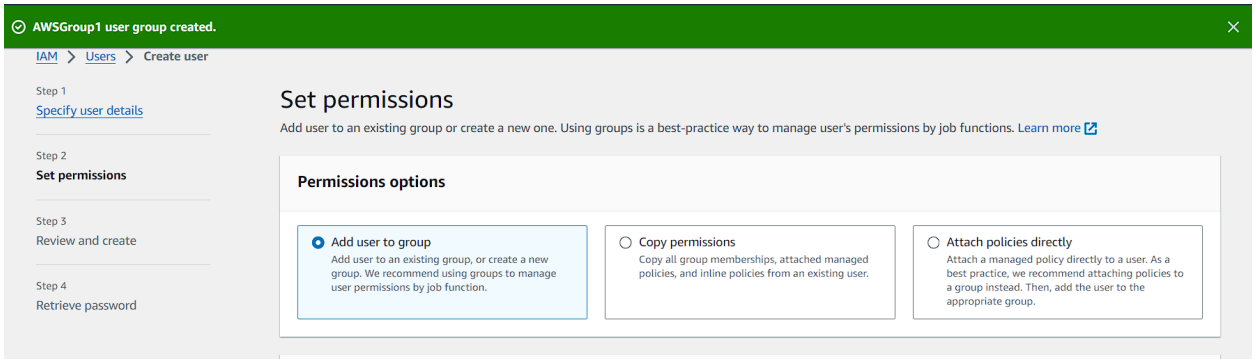
9) User created successfully and can be added to user groups.



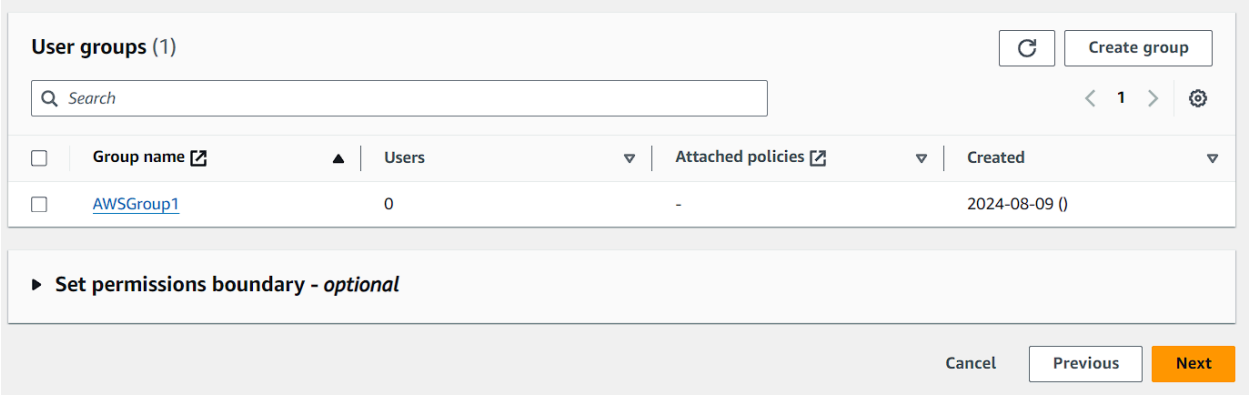
10) User credentials can be downloaded.



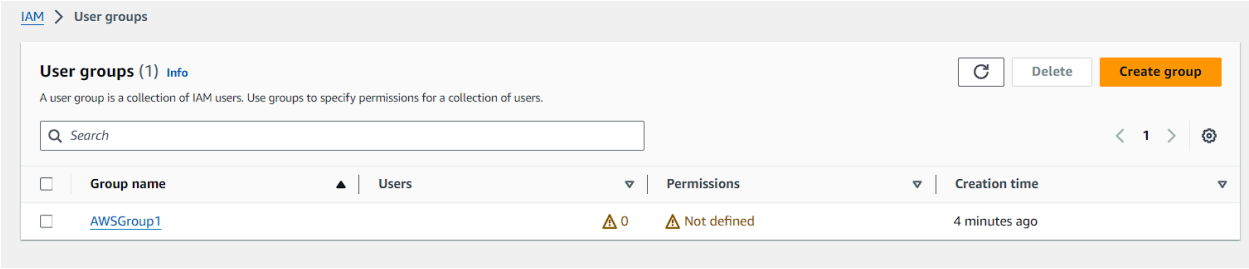
11) Add user to group if group exists else create a new group.



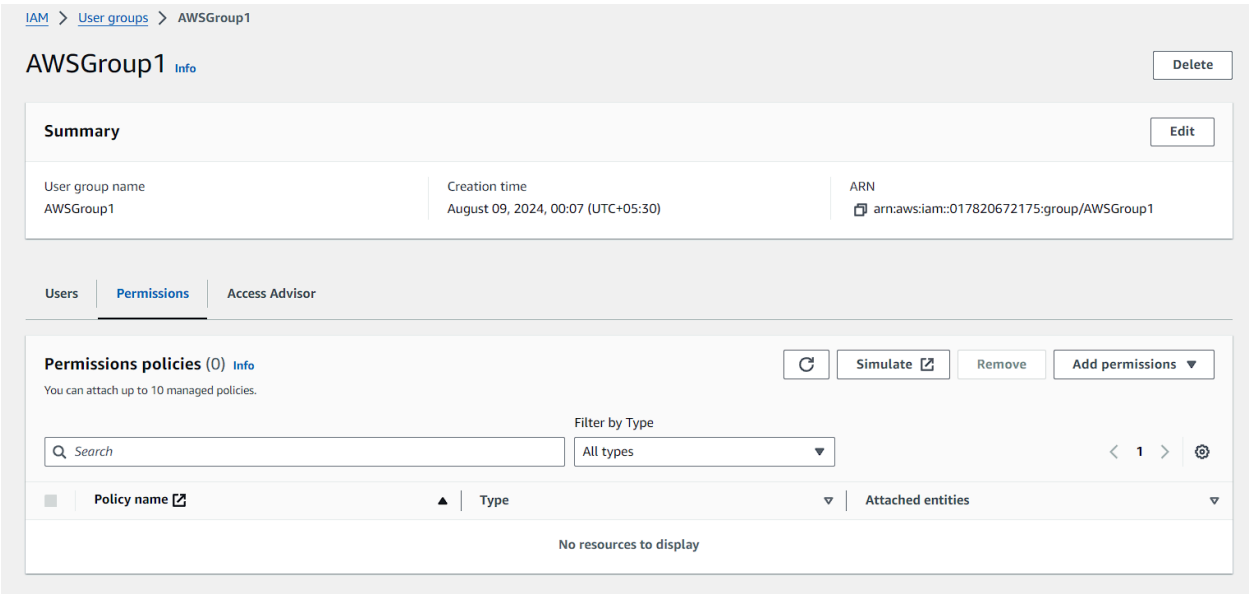
12) Write the user group name and proceed.



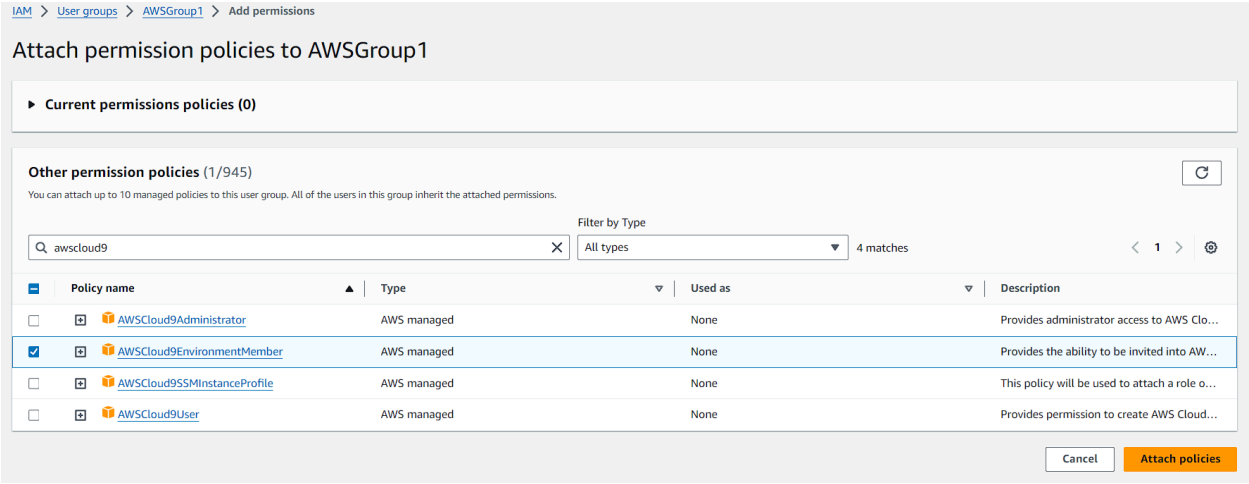
13) Click on *group_name.*



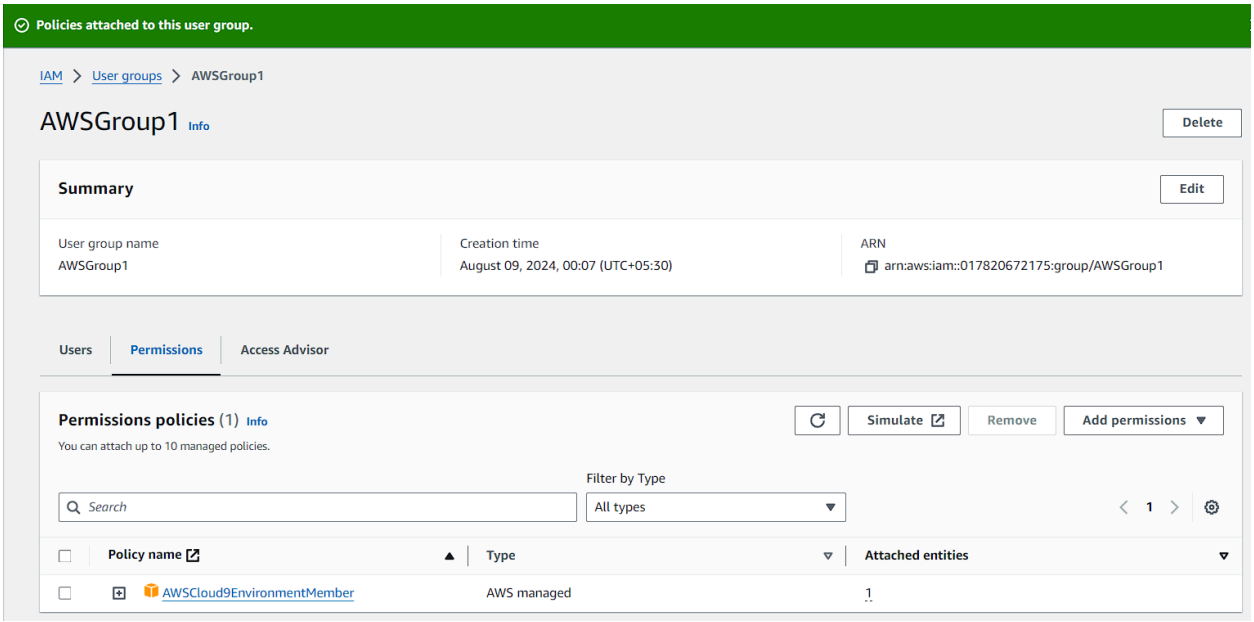
14) Go to Add permissions and click on Add Permissions



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

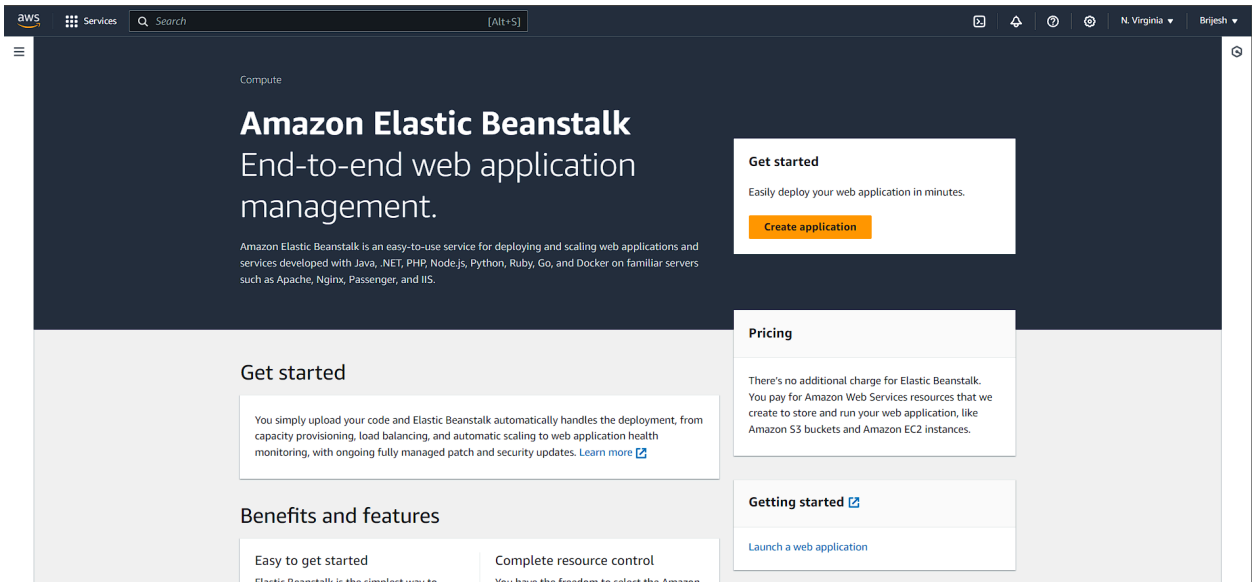


16) User group is created successfully.

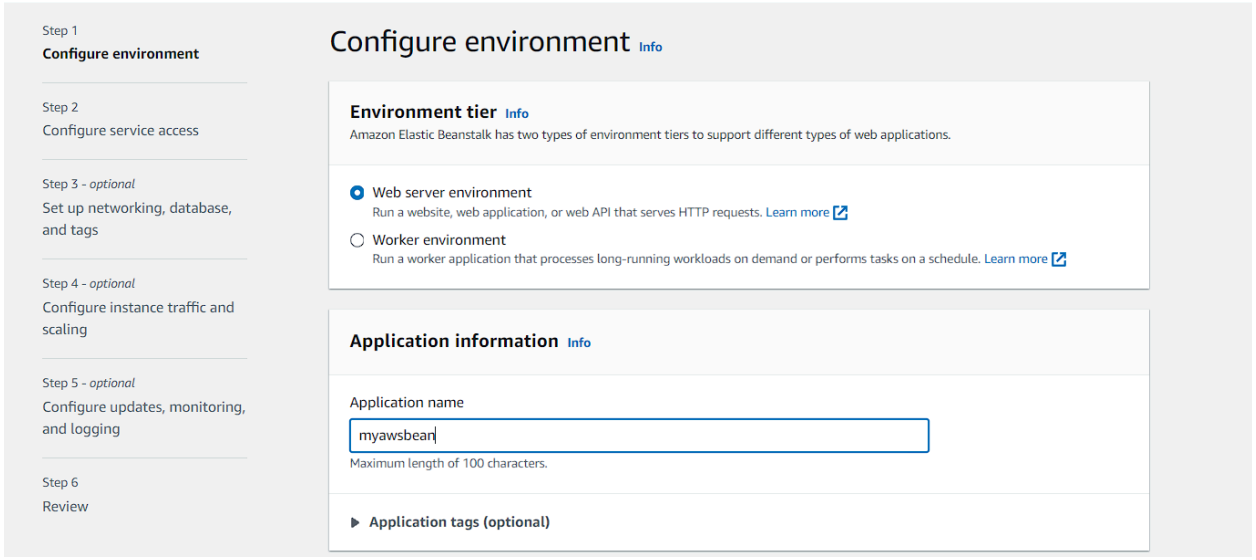


Aim : To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy

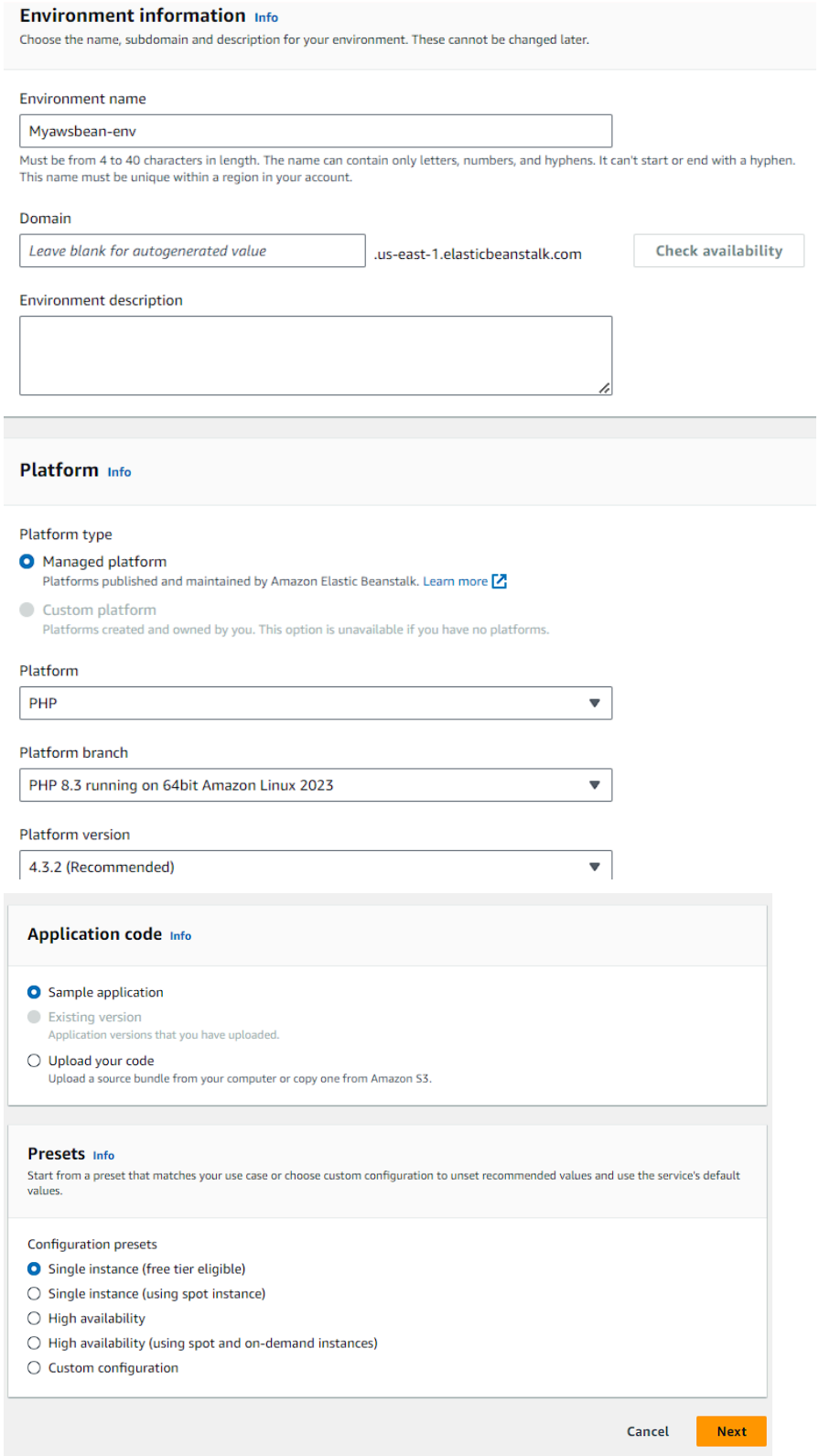
1) Open the aws console and then search Elastic Beanst



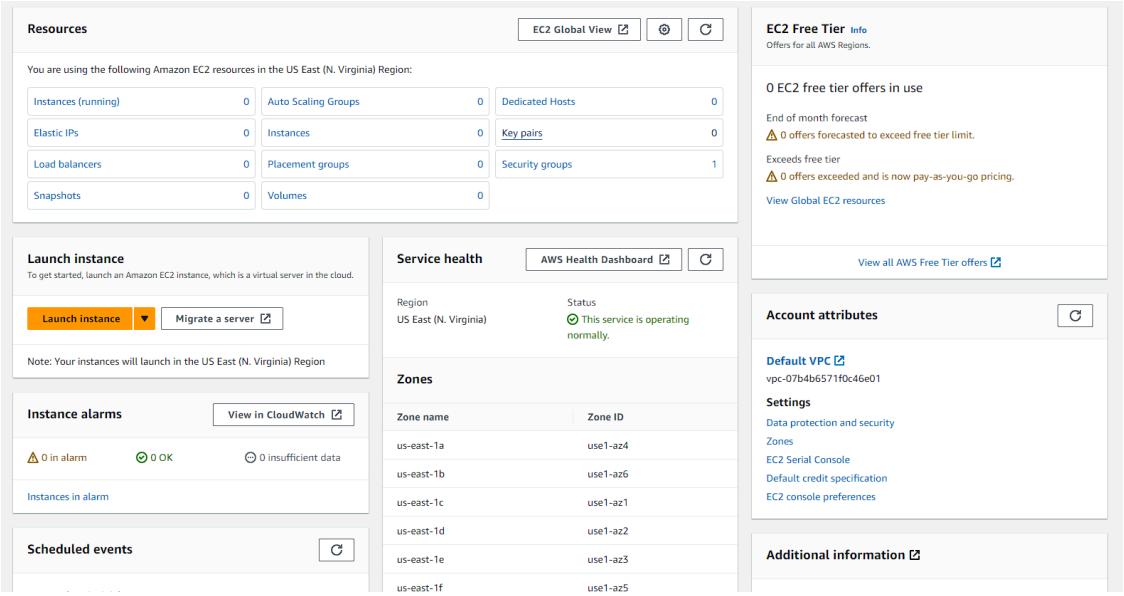
2) Click on create application and configure the environment by adding your application name.



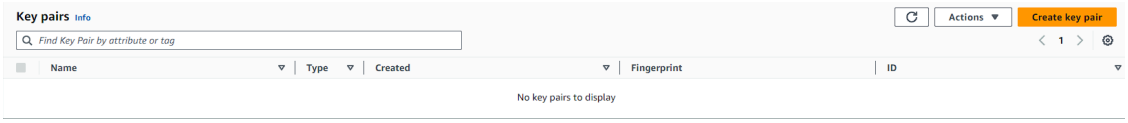
3) Select the environment as PHP and other options as default and click on next.



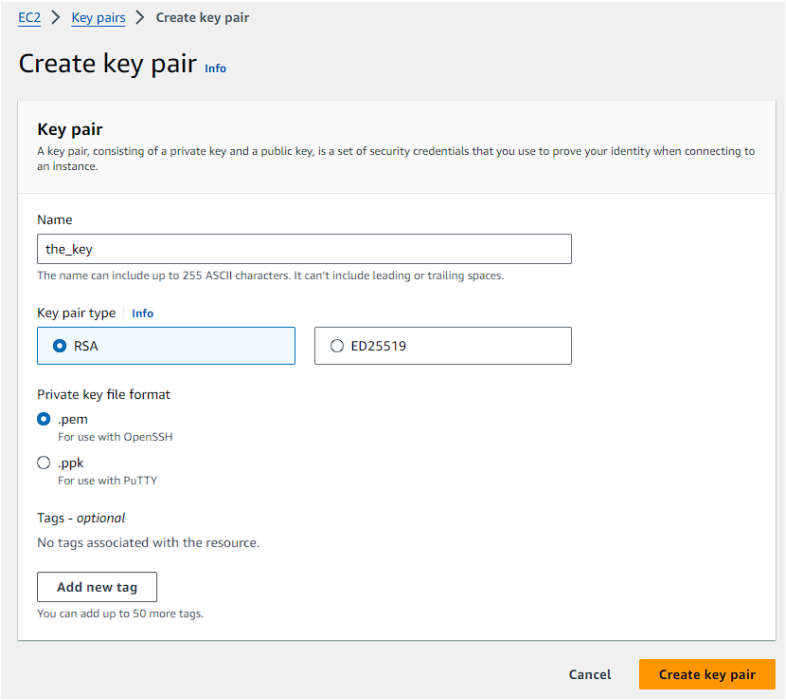
4) After clicking on Next for creating Elastic Beanstalk, we need key-pair that will be require for deployment. Go to EC2 Instance and click on Key Pairs.



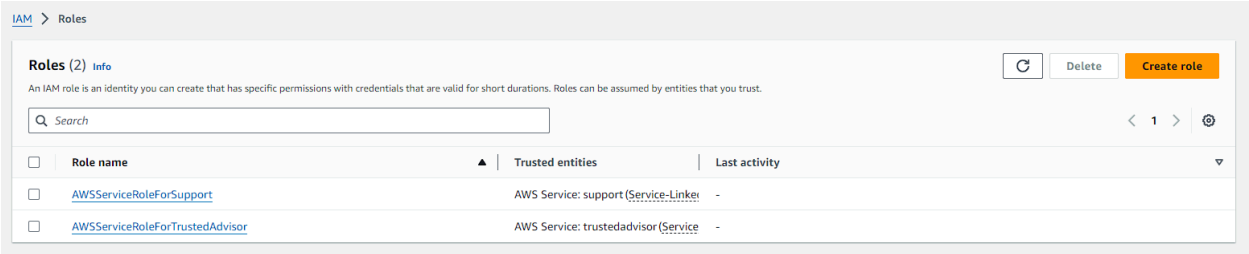
5) Then click on Create key pair



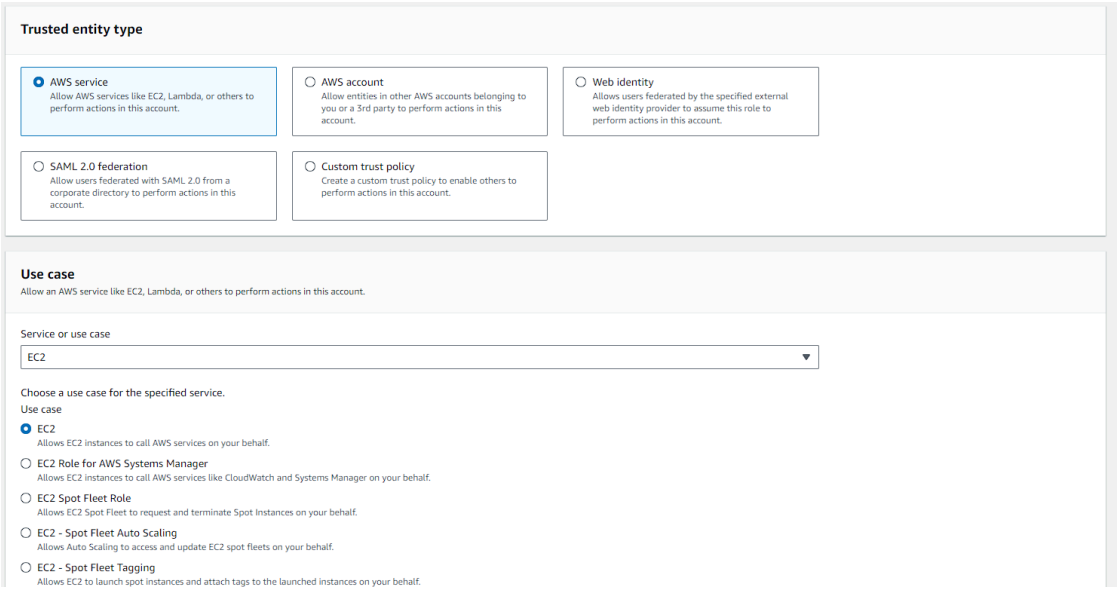
6) Input the name of the key-pair and select pem as file format and click on Create key pair.



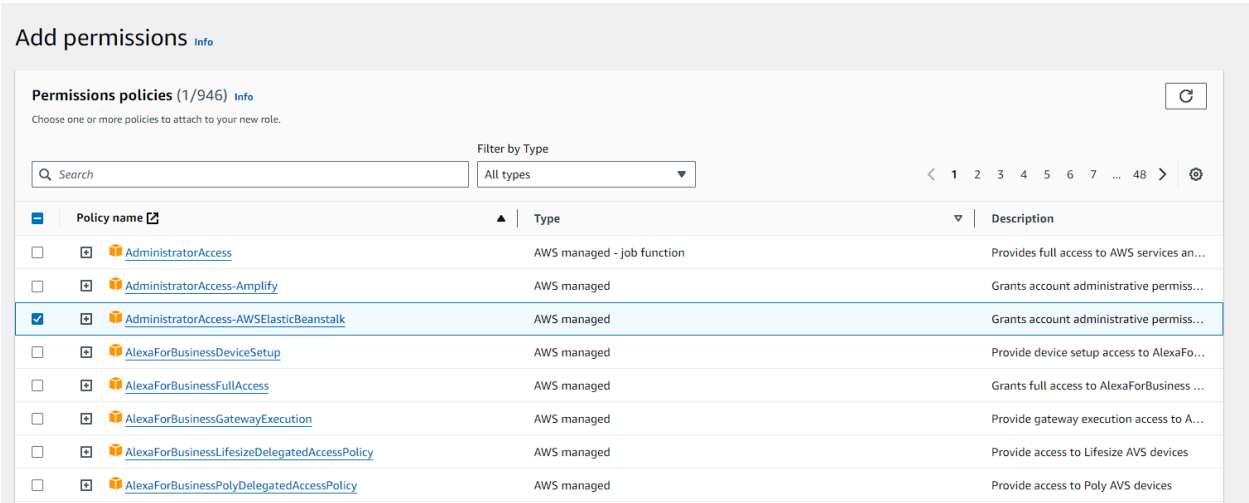
7) After creating key pair, open new tab and go to IAM to create a role that will be used to build Codepipeline. Click on Create role.



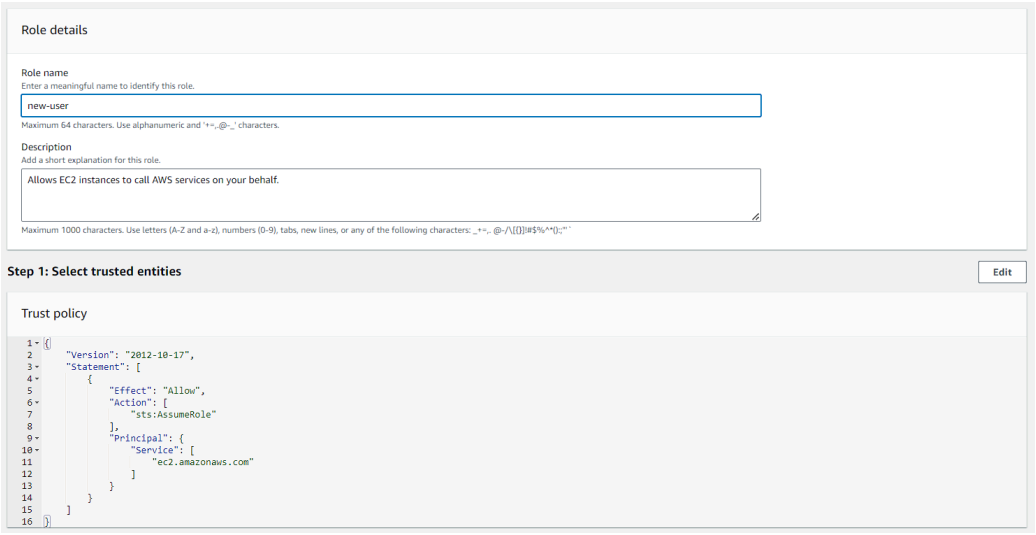
8) Select AWS service as Trusted Entity type and EC2 as service.



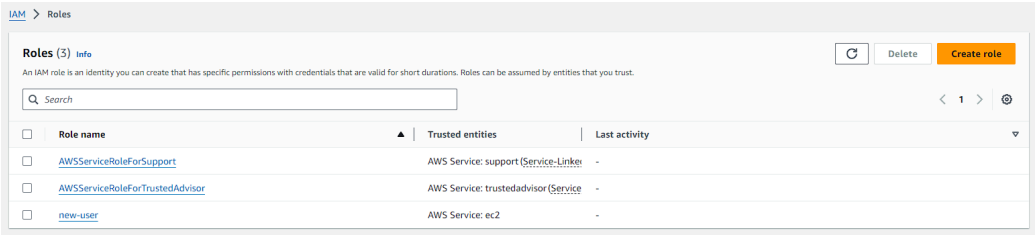
9) Choose AdministratorAccess-AWSElasticBeanstalk as Policy and click on Next.



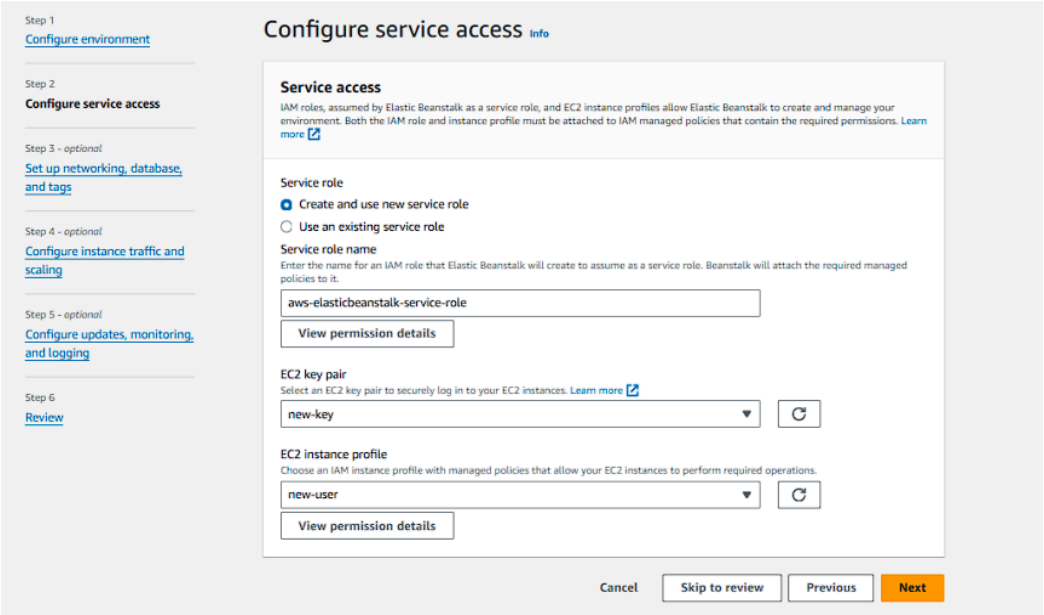
10) Name the role and keep other as default.

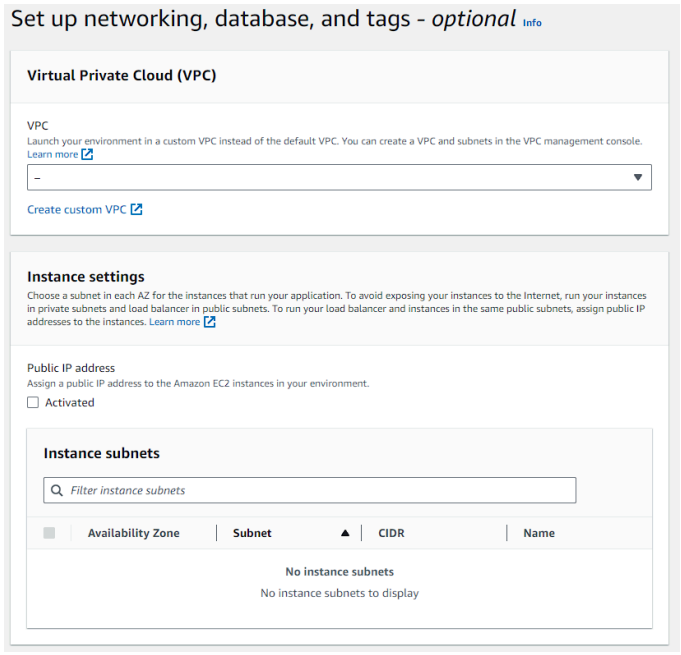


11) The role is created successfully.

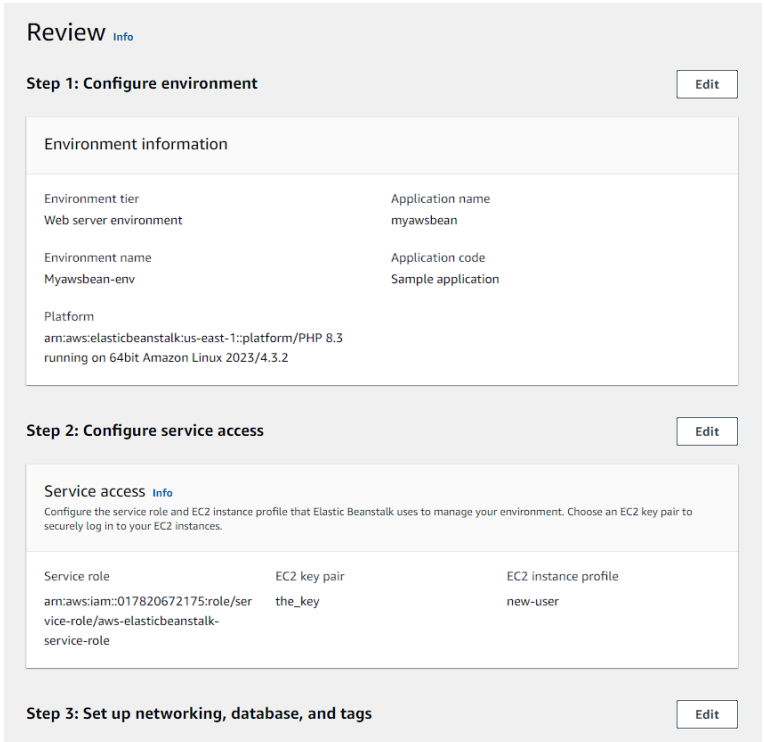


12) Now move to the tab where Elastic Beanstalk was opened and from the drop down menu select the newly created key pair and instance profile. Now let everything be default.

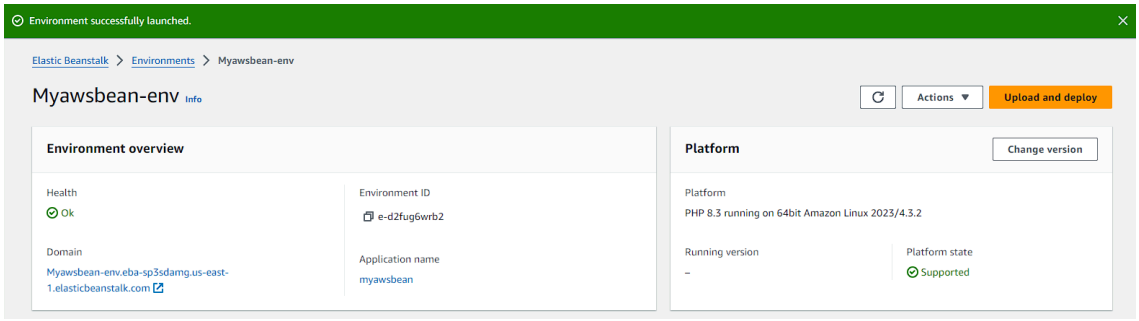




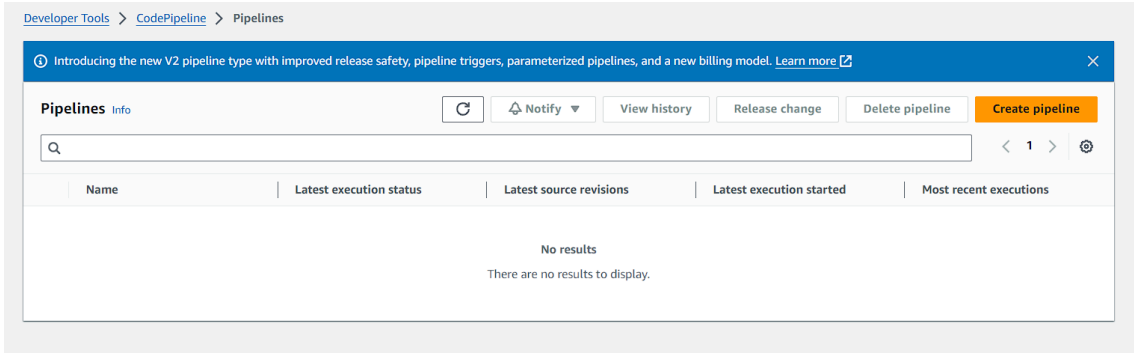
13) Review the changes and click on Create.



14) 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.



15) Now, we need to make a CodePipeline. Go to CodePipeline and click on Create Pipeline.



16) Name the pipeline and select the service role as below and click on Next.

Choose pipeline settings Info

Step 1 of 5

Pipeline settings

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

mypipeline

No more than 100 characters

Pipeline type

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

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

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

Service role

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

Role name

AWSCodePipelineServiceRole-us-east-1-mypipeline

Type your service role name

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

Variables

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

No variables defined at the pipeline level in this pipeline.

Add variable

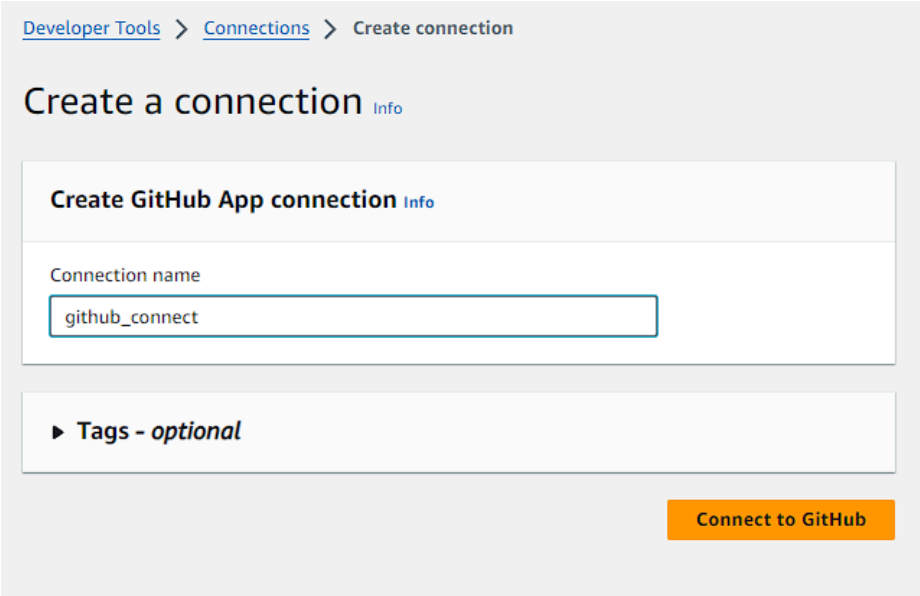
You can add up to 50 variables.

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

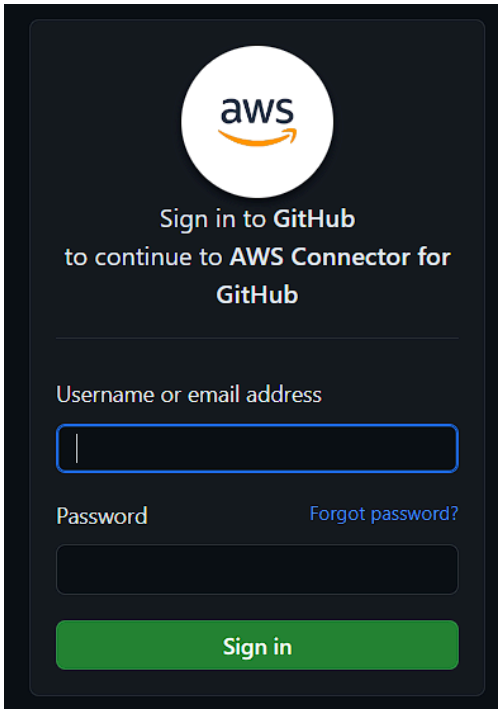
Advanced settings

Cancel Next

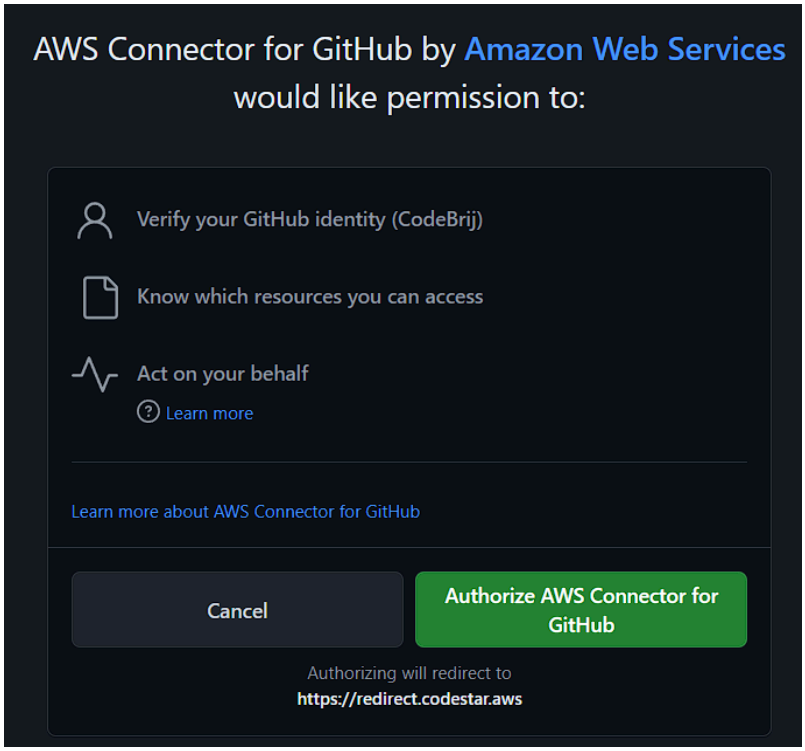
17) In the source stage select Github v2 as the provider and then connect your github connect so that the pipeline can access the forked source code. Name the connection.



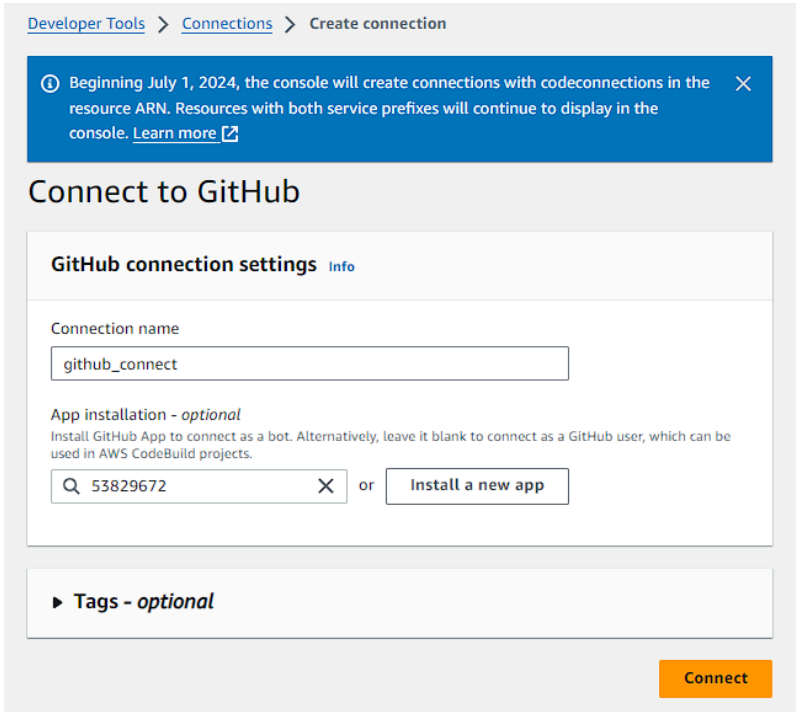
18) Signin to GitHub to connect with AWS.



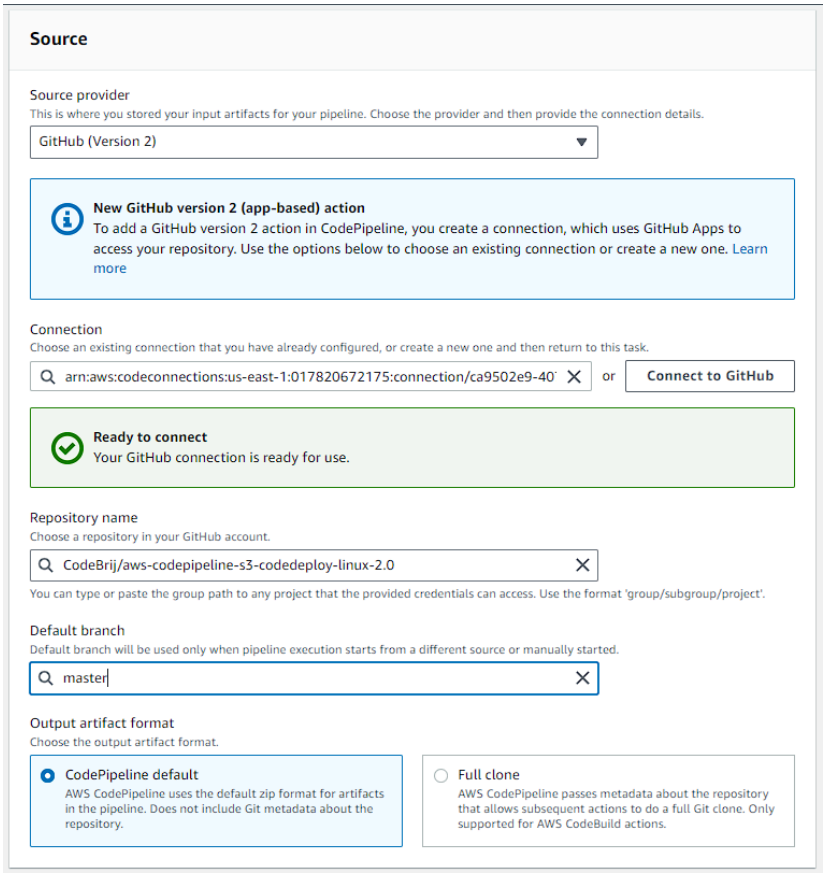
19) Authorize AWS Connector for GitHub.



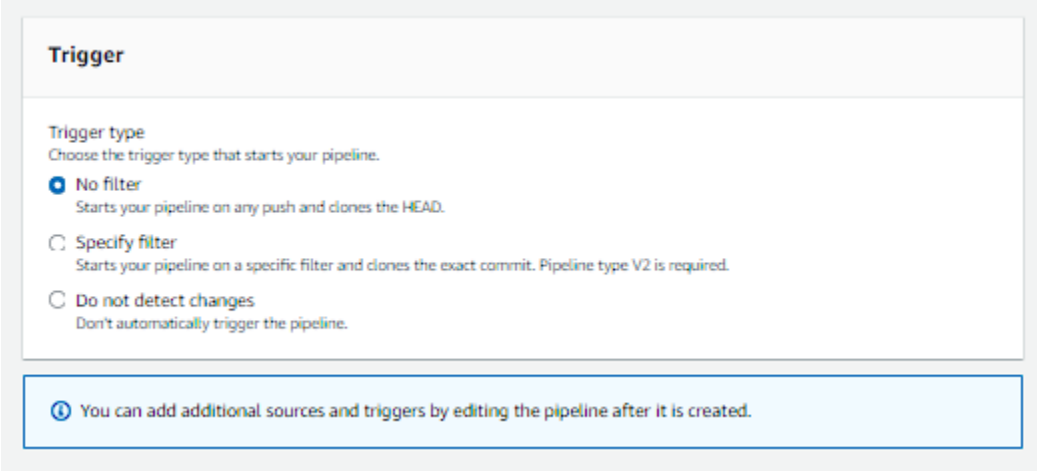
20) We need to install the GitHub connector.



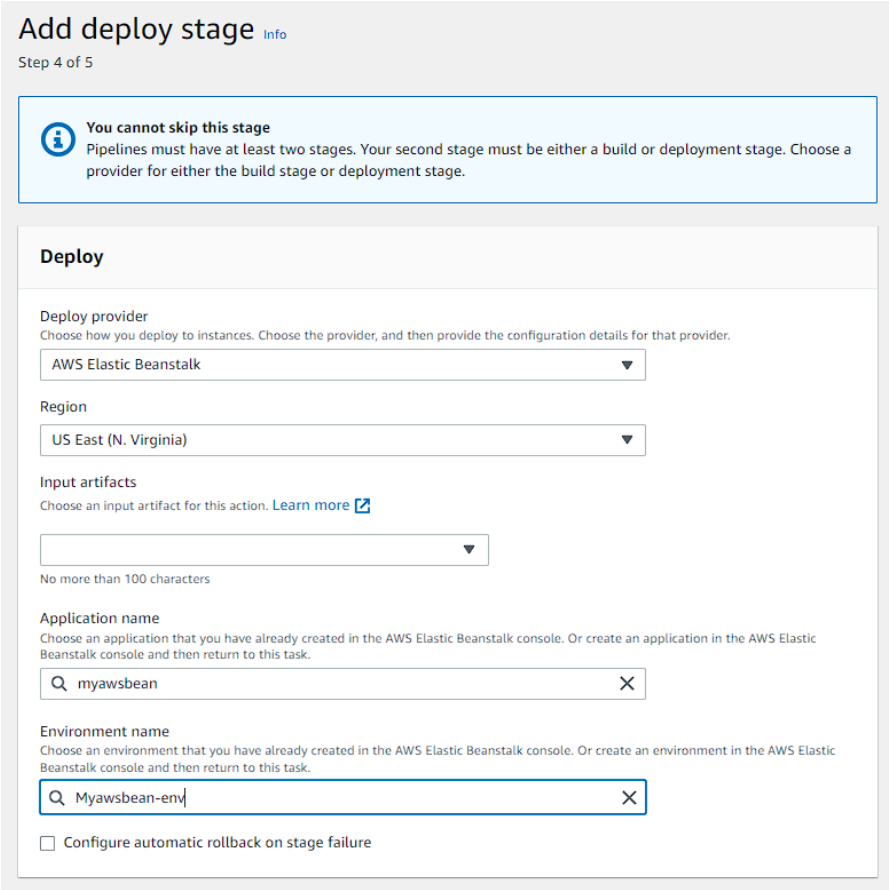
21) Now, select the repository and the branch to be deployed.



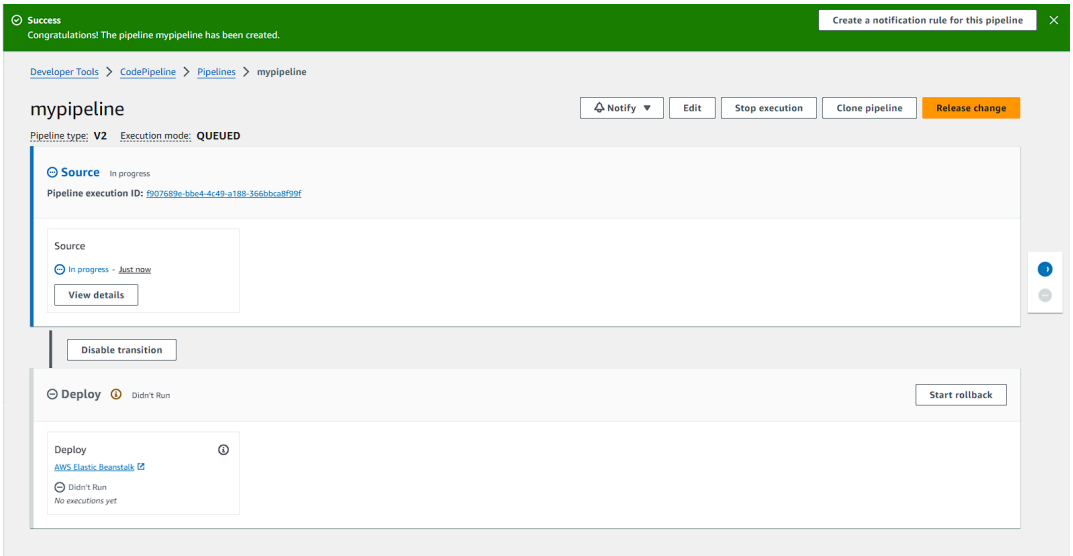
22) Select No fliter in Trigger.



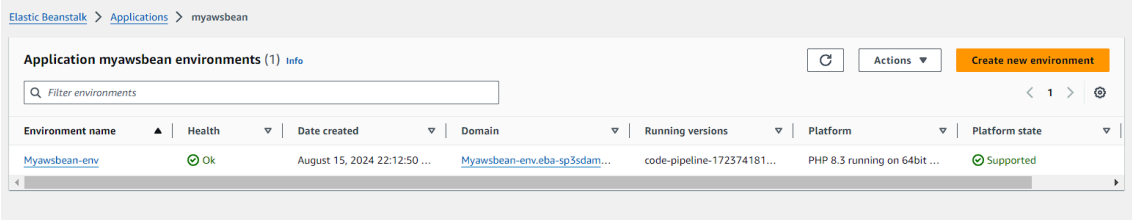
23) In deploy stage add application name as environment name. Then review the settings and click on Create pipeline.



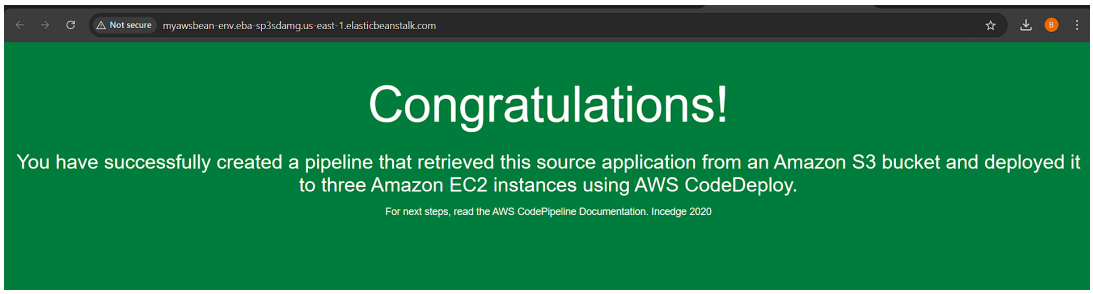
24) The pipeline is ready and the provided repo is deployed successfully.



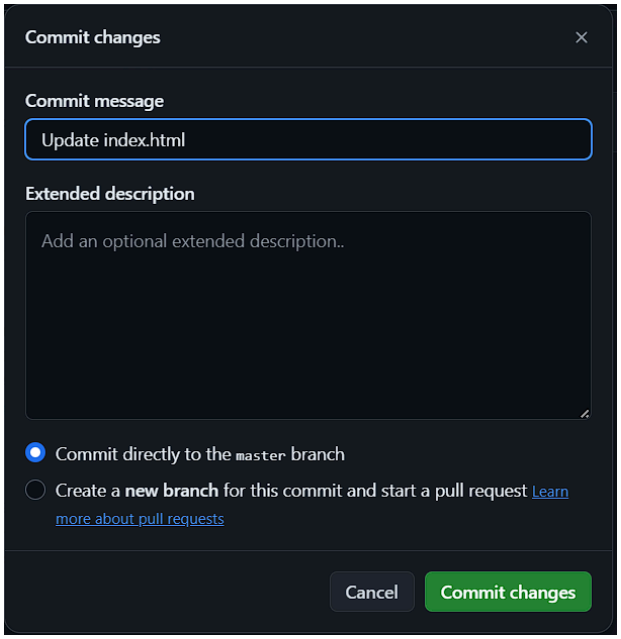
25) Go to Elastic Beanstalk and from DOmain open the hosted site.

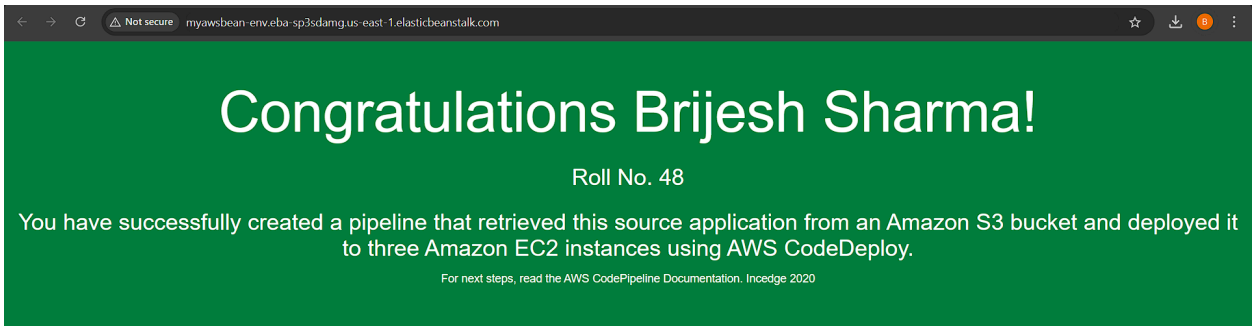


26) Hosted site from the github repository.



27) Make some changes in the index.html and reload.



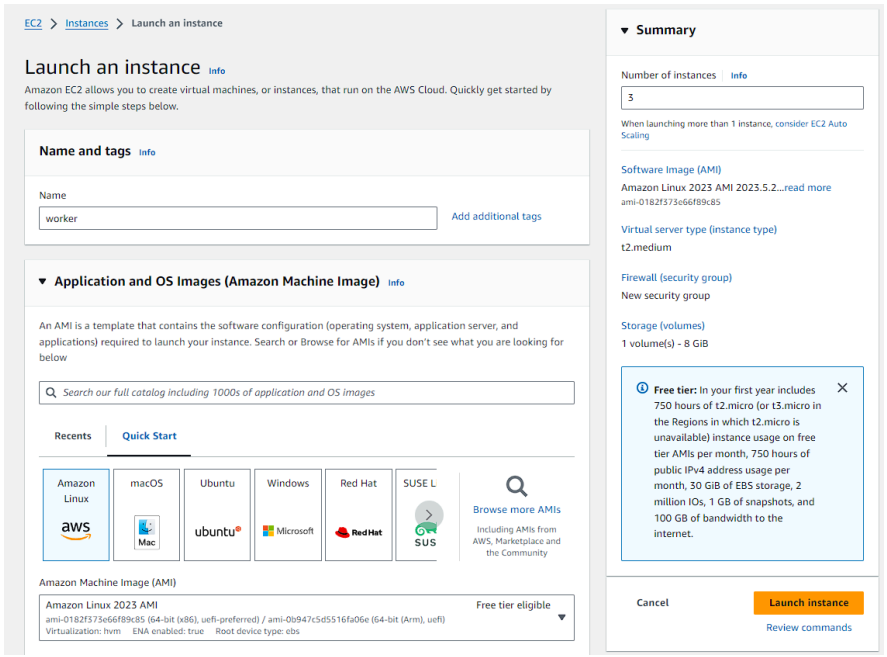


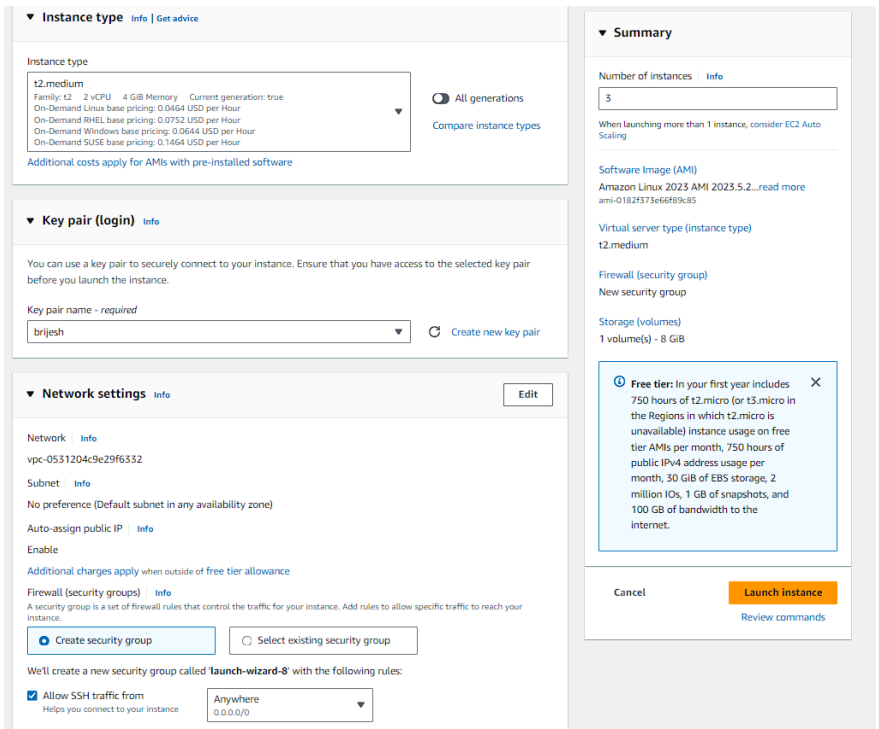
Name - Brijesh Sharma **Div - D15C** **Roll No. 48**

Experiment - 3

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

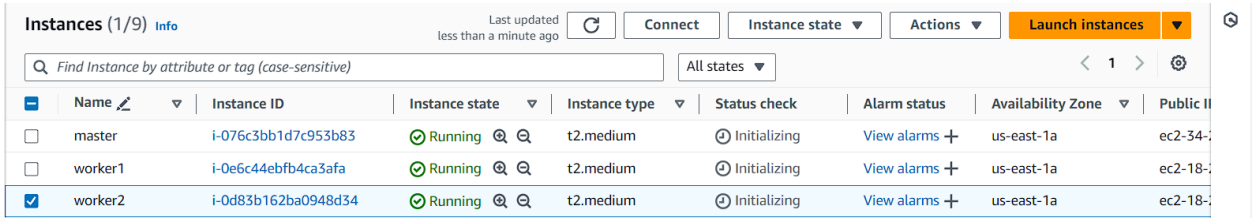
1) Create 3 EC-2 instances with all running on Amazon Linux as OS with inbound SSH allowed.
2) To efficient run kubernetes cluster, select instance type of at least t2.medium as kubernetes recommends at least 2 vCPU to run smoothly



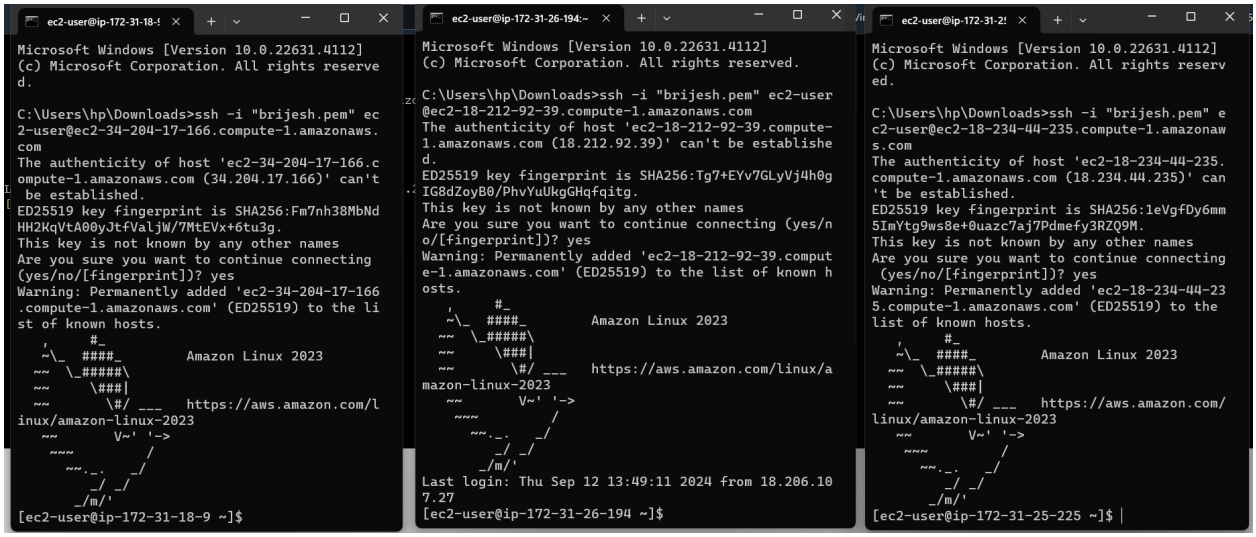


Name - Brijesh Sharma **Div - D15C** **Roll No. 48**

3) Three instance are ready - master, worker1, and worker2.



4) Connect the instances to the local terminal using the SSH client.



5) Run the following commands on all the machines.

Install Docker

a) sudo yum install docker -y

```
[ec2-user@ip-172-31-25-225 ~]$ sudo yum install docker -y
Last metadata expiration check: 0:06:08 ago on Thu Sep 12 13:45:25 2024.
Dependencies resolved.
================================================================================
 Package                 Architecture  Version                    Repository     Size
================================================================================
Installing:
 docker                  x86_64        25.0.6-1.amzn2023.0.2      amazonlinux    44 M
Installing dependencies:
 containerd              x86_64        1.7.20-1.amzn2023.0.1      amazonlinux    35 M
 iptables-libs           x86_64        1.8.8-3.amzn2023.0.2       amazonlinux   401 k
 iptables-nft            x86_64        1.8.8-3.amzn2023.0.2       amazonlinux   183 k
 libcgroup               x86_64        3.0-1.amzn2023.0.1         amazonlinux    75 k
 libnetfilter_conntrack  x86_64        1.0.8-2.amzn2023.0.2       amazonlinux    58 k
 libnfnetlink            x86_64        1.0.1-19.amzn2023.0.2      amazonlinux    30 k
 libnftnl                x86_64        1.2.2-2.amzn2023.0.2       amazonlinux    84 k
 pigz                    x86_64        2.5-1.amzn2023.0.3         amazonlinux    83 k
 runc                    x86_64        1.1.13-1.amzn2023.0.1      amazonlinux   3.2 M

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

Total download size: 84 M
Installed size: 317 M
Downloading Packages:
(1/10): iptables-libs-1.8.8-3.amzn2023.0.2.x86_64.rpm   3.6 MB/s | 401 kB     00:00
```

b) Then, configure cgroup in a daemon.json file by using following commands. This allows kubernetes to manage host more efficiently -

- cd /etc/docker

Run the scripts below -

```
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-172-31-18-9 ~]$ cd /etc/docker
[ec2-user@ip-172-31-18-9 docker]$
```

```
[ec2-user@ip-172-31-18-9 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-172-31-18-9 docker]$
```

c) After configuring restart docker service service :

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

```
[ec2-user@ip-172-31-18-9 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-172-31-18-9 docker]$
```

```
[ec2-user@ip-172-31-18-9 docker]$ sudo systemctl daemon-reload
sudo systemctl restart docker
docker -v
Docker version 25.0.5, build 5dc9bcc
[ec2-user@ip-172-31-18-9 docker]$
```

Name - Brijesh Sharma **Div - D15C** **Roll No. 48**

Install Kubernetes

a) SELinux needs to be disabled before configuring kubelet to avoid interference with kubernetes api server

- sudo setenforce 0
- sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
- Add kubernetes repository (paste in terminal)

  cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
  [kubernetes]
  name=Kubernetes
  baseurl=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/
  enabled=1
  gpgcheck=1
  gpgkey=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/repodata/repomd.xml.key
  exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
  EOF

```
[ec2-user@ip-172-31-18-9 docker]$ sudo setenforce 0
[ec2-user@ip-172-31-18-9 docker]$ sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
[ec2-user@ip-172-31-18-9 docker]$ cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.30/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
[ec2-user@ip-172-31-18-9 docker]$
```

b) sudo yum update

```
[ec2-user@ip-172-31-25-225 docker]$ sudo yum update
Kubernetes
Dependencies resolved.
Nothing to do.
Complete!
```

c) sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes

```
[ec2-user@ip-172-31-18-9 docker]$ sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
Last metadata expiration check: 0:00:54 ago on Thu Sep 12 14:50:28 2024.
Dependencies resolved.
================================================================================================================================================
 Package                                 Architecture              Version                                 Repository                        Size
================================================================================================================================================
Installing:
 kubeadm                                 x86_64                    1.30.5-150500.1.1                       kubernetes                        10 M
 kubectl                                 x86_64                    1.30.5-150500.1.1                       kubernetes                        10 M
 kubelet                                 x86_64                    1.30.5-150500.1.1                       kubernetes                        17 M
Installing dependencies:
 conntrack-tools                         x86_64                    1.4.6-2.amzn2023.0.2                    amazonlinux                      208 k
 cri-tools                               x86_64                    1.30.1-150500.1.1                       kubernetes                       8.6 M
 kubernetes-cni                          x86_64                    1.4.0-150500.1.1                        kubernetes                       6.7 M
 libnetfilter_cthelper                   x86_64                    1.0.0-21.amzn2023.0.2                   amazonlinux                       24 k
 libnetfilter_cttimeout                  x86_64                    1.0.0-19.amzn2023.0.2                   amazonlinux                       24 k
 libnetfilter_queue                      x86_64                    1.0.5-2.amzn2023.0.2                    amazonlinux                       30 k

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

Total download size: 53 M
Installed size: 292 M
Downloading Packages:
(1/9): conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64.rpm                                                  3.1 MB/s | 208 kB     00:00
(2/9): libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64.rpm                                           332 kB/s |  24 kB     00:00
```

d) 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-172-31-25-225 docker]$ sudo swapoff -a
[ec2-user@ip-172-31-25-225 docker]$ echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf
net.bridge.bridge-nf-call-iptables=1
[ec2-user@ip-172-31-25-225 docker]$ sudo sysctl -p
net.bridge.bridge-nf-call-iptables = 1
[ec2-user@ip-172-31-25-225 docker]$
```

6) To perform only on Master machine

a) Initialize kubernetes by typing below command
sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all

```
[ec2-user@ip-172-31-18-9 docker]$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all
I0912 14:55:34.563710   30027 version.go:256] remote version is much newer: v1.31.0; falling back to: stable-1.30
[init] Using Kubernetes version: v1.30.4
[preflight] Running pre-flight checks
        [WARNING FileExisting-socat]: socat not found in system path
        [WARNING FileExisting-tc]: tc not found in system path
        [WARNING Service-Kubelet]: kubelet service is not enabled, please run 'systemctl enable kubelet.service'
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
W0912 14:55:34.796553   30027 checks.go:844] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that used b
y kubeadm.It is recommended to use "registry.k8s.io/pause:3.9" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-18-9.ec2.internal kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cl
uster.local] and IPs [10.96.0.1 172.31.18.9]
[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
```

b) Copy this join link

kubeadm join 172.31.22.128:6443 --token 2nzclk.1ek0i93tsqnednb9 \
--discovery-token-ca-cert-hash
sha256:e7c55b0579b7e928431704c459e9c9c521c4af034e3d346f3418e1afc672928d

c) Run the below command -

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

d) Then, add a common networking plugin called flammel file as mentioned in the code.
kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml

```
[ec2-user@ip-172-31-22-128 docker]$ kubectl apply -f https://raw.githubuserconte
nt.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
[ec2-user@ip-172-31-22-128 docker]$ kubectl get pods
No resources found in default namespace.
```

e) Check the created pod using this command
- kubectl get pods

7) To perform on both worker machine -

a) sudo yum install iproute-tc socat -y

```
[ec2-user@ip-172-31-27-40 ~]$ sudo yum install iproute-tc socat -y
Last metadata expiration check: 0:11:39 ago on Sat Sep 14 12:44:32 2024.
Dependencies resolved.
================================================================================ Package         Architecture
================================================================================Installing:
 iproute-tc       x86_64       5.10.0-2.amzn2023.0.5         amazonlinux       455 k
 socat            x86_64       1.7.4.2-1.amzn2023.0.2        amazonlinux       303 k

Transaction Summary
================================================================================Install  2 Packages

Total download size: 758 k
Installed size: 2.0 M
Downloading Packages:
(1/2): socat-1.7.4.2-1.amzn2023.0.2.x86_64.rpm       4.5 MB/s | 303 kB     00:00
(2/2): iproute-tc-5.10.0-2.amzn2023.0.5.x86_64.rpm   6.1 MB/s | 455 kB     00:00
--------------------------------------------------------------------------------Total
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : socat-1.7.4.2-1.amzn2023.0.2.x86_64                    1/2
  Installing       : iproute-tc-5.10.0-2.amzn2023.0.5.x86_64                2/2
  Running scriptlet: iproute-tc-5.10.0-2.amzn2023.0.5.x86_64                2/2
  Verifying        : iproute-tc-5.10.0-2.amzn2023.0.5.x86_64                1/2
  Verifying        : socat-1.7.4.2-1.amzn2023.0.2.x86_64                    2/2

Installed:
  iproute-tc-5.10.0-2.amzn2023.0.5.x86_64     socat-1.7.4.2-1.amzn2023.0.2.x86_64

Complete!
```

b) sudo systemctl enable kubelet

```
[ec2-user@ip-172-31-17-38 ~]$ sudo systemctl enable kubelet
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service → /u
sr/lib/systemd/system/kubelet.service.
```

c) sudo systemctl restart kubelet
d) kubeadm join 172.31.22.128:6443 --token 2nzclk.1ek0i93tsqnednb9 \
--discovery-token-ca-cert-hash
sha256:e7c55b0579b7e928431704c459e9c9c521c4af034e3d346f3418e1afc672928d

```
[ec2-user@ip-172-31-17-38 ~]$ sudo kubeadm join 172.31.22.128:6443 --token 2nzcl
k.1ek0i93tsqnednb9 --discovery-token-ca-cert-hash sha256:e7c55b0579b7e928431704c
459e9c9c521c4af034e3d346f3418e1afc672928d
[preflight] Running pre-flight checks
error execution phase preflight: couldn't validate the identity of the API Serve
r: failed to request the cluster-info ConfigMap: Get "https://172.31.22.128:6443
/api/v1/namespaces/kube-public/configmaps/cluster-info?timeout=10s": context dea
dline exceeded
To see the stack trace of this error execute with --v=5 or higher
```

If it gives error or refusing connection, restart the master server using sudo systemctl restart kubelet and try to connect.

With the help of command the worker nodes are connected master node and is ready to do task assigned by master node.

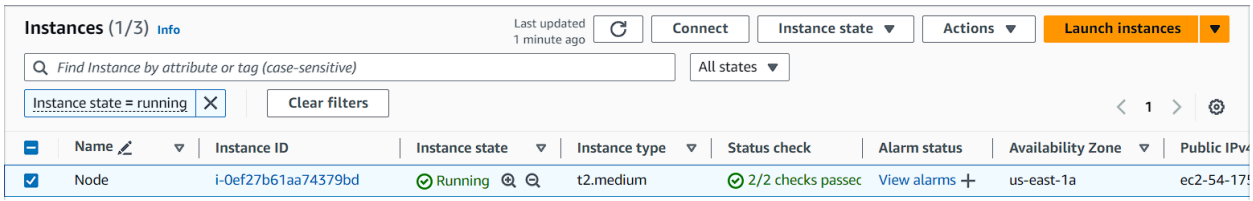
Conclusion -
In this experiment, we connected master nodes from Kubernetes to the worker nodes successfully First, we created the instances and connected with Kubernetes, while installing and configuring, there were some packages, that were needed to be installed separately. Even while connecting the nodes, error occurs and hence, system needs to be restarted and connected properly..

Name - Brijesh Sharma **Div - D15C** **Roll No. 48**

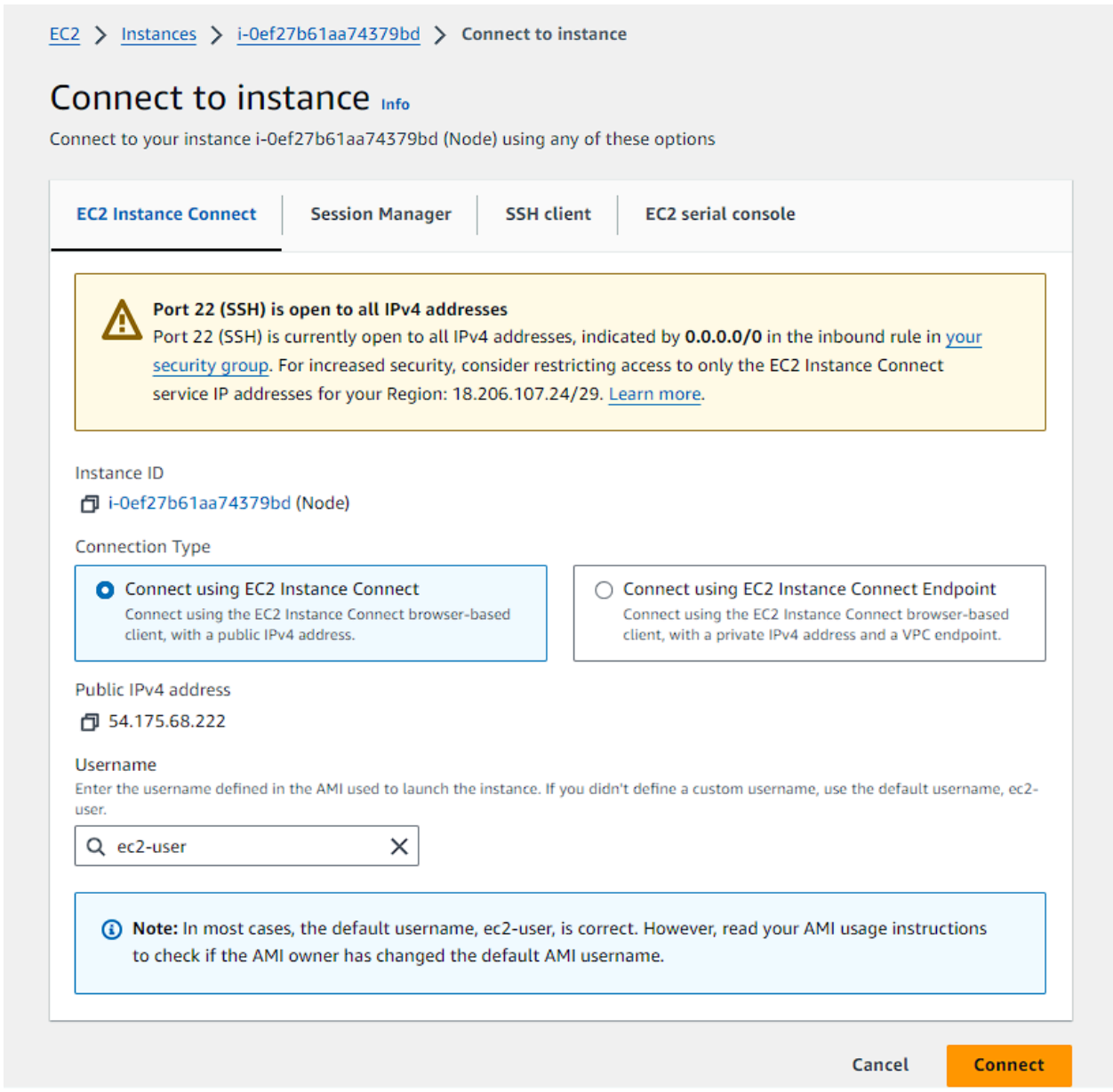
Experiment - 4

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

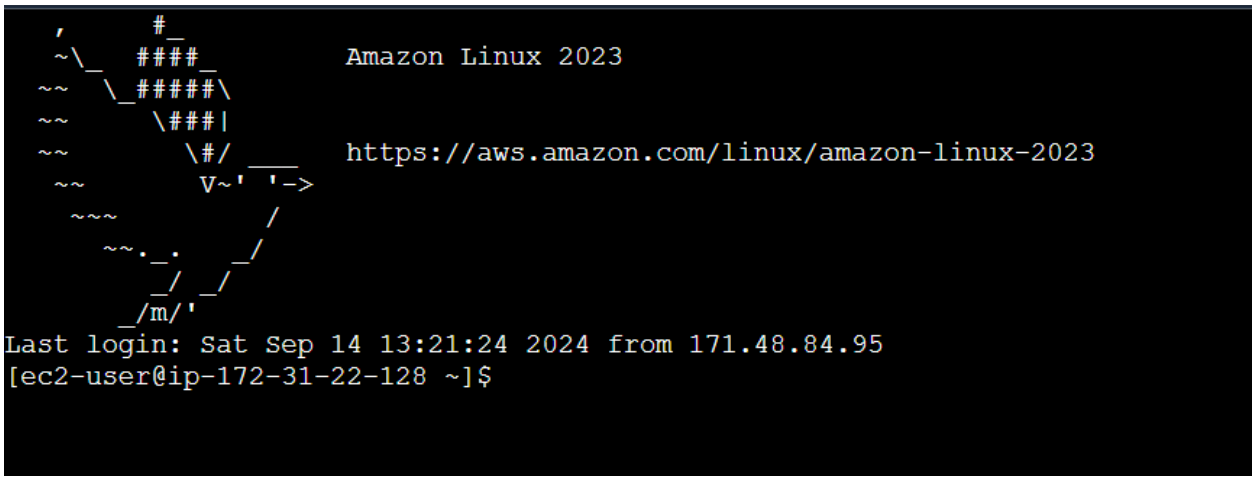
1) Create an EC2 instance and allow SSH traffic to connect.



2) Select the instance and click on Connect and go to Connect.

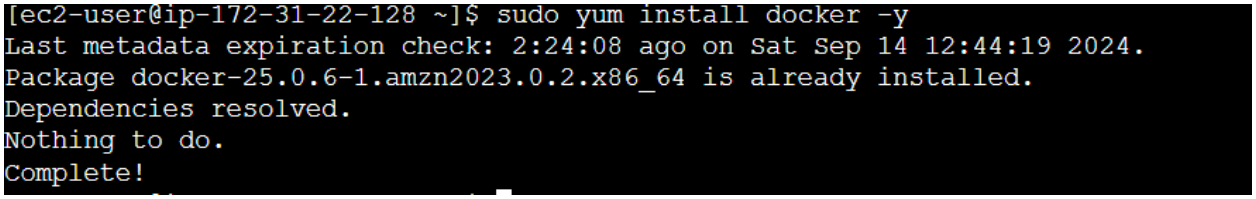


3) EC2 server is connected.



4) To install docker run the following command:

sudo yum install docker -y



5) Configure cgroup in daemon.json file using the following commands

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-172-31-30-107 ~]$ cd /etc/docker
[ec2-user@ip-172-31-30-107 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"
}
```

6) Run the following command after this:
   sudo systemctl enable docker
   sudo systemctl daemon-reload
   sudo systemctl restart docker

```
[ec2-user@ip-172-31-30-107 docker]$ sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
```

7) Check for installation of docker by docker -v command.

```
[ec2-user@ip-172-31-30-107 docker]$ docker -v
Docker version 25.0.5, build 5dc9bcc
```

Install Kubernetes

8) Disable SELinux before configuring kubelet
   - sudo setenforce 0
   - sudo sed -i 's/^SELINUX-enforcing$/SELINUX=permissive/' /etc/selinux/config

```
[ec2-user@ip-172-31-30-107 docker]$ sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
```

9) Add kubernetes 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
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF

10) Run the commands to update and install kubernetes packages
sudo yum update
sudo yum install -y kubelet kubeadm kubectl --disableexcludes kubernetes

```
[ec2-user@ip-172-31-31-241 docker]$ sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
Last metadata expiration check: 0:00:46 ago on Sat Sep 14 16:39:07 2024.
Dependencies resolved.
================================================================================
 Package                 Architecture  Version                 Repository    Size
================================================================================
Installing:
 kubeadm                 x86_64        1.31.1-150500.1.1       kubernetes    11 M
 kubectl                 x86_64        1.31.1-150500.1.1       kubernetes    11 M
 kubelet                 x86_64        1.31.1-150500.1.1       kubernetes    15 M
Installing dependencies:
 conntrack-tools         x86_64        1.4.6-2.amzn2023.0.2    amazonlinux  208 k
 cri-tools               x86_64        1.31.1-150500.1.1       kubernetes   6.9 M
 kubernetes-cni          x86_64        1.5.1-150500.1.1        kubernetes   7.1 M
 libnetfilter_cthelper   x86_64        1.0.0-21.amzn2023.0.2   amazonlinux   24 k
 libnetfilter_cttimeout  x86_64        1.0.0-19.amzn2023.0.2   amazonlinux   24 k
 libnetfilter_queue      x86_64        1.0.5-2.amzn2023.0.2    amazonlinux   30 k

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

Total download size: 51 M
Installed size: 269 M
Downloading Packages:
(1/9): libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64.rpm  412 kB/s |  24 kB  00:00
```

11) 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-172-31-31-241 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
```

12) Initialize the kubecluster
sudo kubeadm init --pod-network-cidr-10.244.0.0/16

```
[ec2-user@ip-172-31-31-241 docker]$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
        [WARNING FileExisting-socat]: socat not found in system path
        [WARNING FileExisting-tc]: tc not found in system path
        [WARNING Service-Kubelet]: kubelet service is not enabled, please run 'systemctl enable kubelet.service'
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W0914 16:43:42.619918   27909 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that used by kubeadm.It is recommended to use "registry.k8s.io/pause:3.10" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-31-241.ec2.internal kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.31.241]
[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-172-31-31-241.ec2.internal localhost] and IPs [172.31.31.241 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-31-241.ec2.internal localhost] and IPs [172.31.31.241 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
```

13) Copy the Join Command-

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

kubeadm join 172.31.31.241:6443 --token x6zuwa.bafjhtm4fqxp4yi8 \
        --discovery-token-ca-cert-hash sha256:ad0a2b0eb8318975f42be705f750f923adc01bd102ebd7d93401df81429ef5a5
```

kubeadm join 172.31.31.241:6443 --token x6zuwa.bafjhtm4fqxp4yi8 \
--discovery-token-ca-cert-hash
sha256:ad0a2b0eb8318975f42be705f750f923adc01bd102ebd7d93401df81429ef5a5

14) Run the following commands

```
[ec2-user@ip-172-31-31-241 docker]$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
cp: overwrite '/home/ec2-user/.kube/config'? yes
```

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

```
c2-user@ip-172-31-20-245 dockekubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
mespace/kube-flannel created
usterrole.rbac.authorization.k8s.io/flannel created
usterrolebinding.rbac.authorization.k8s.io/flannel created
rviceaccount/flannel created
nfigmap/kube-flannel-cfg created
emonset.apps/kube-flannel-ds created
c2-user@ip-172-31-20-245 docker]$
```

16) Check for creation of pods

```
[ec2-user@ip-172-31-20-245 ~]$ kubectl get pods
NAME    READY   STATUS    RESTARTS   AGE
nginx   0/1     Pending   0          80s
```

17) To change the state from pending to running, use the following command kubectl describe pod nginx
This command will help to describe the pods it gives reason for failure as it shows the untolerated taints which need to be untainted.

```
Containers:
  nginx:
    Image:        nginx:1.14.2
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-dmncs (ro)
Conditions:
  Type           Status
  PodScheduled   False
Volumes:
  kube-api-access-dmncs:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
QoS Class:                   BestEffort
Node-Selectors:              <none>
Tolerations:                 node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type     Reason            Age    From               Message
  ----     ------            ----   ----               -------
  Warning  FailedScheduling  2m47s  default-scheduler  0/1 nodes are available: 1 node(s) had untolerated taint {node-role.kubernetes.
io/control-plane: }. preemption: 0/1 nodes are available: 1 Preemption is not helpful for scheduling.
```

```
[ec2-user@ip-172-31-20-245 docker]$ kubectl taint nodes ip-172-31-20-245.ec2.internal node-role.kubernetes.io/control-plane-
node/ip-172-31-20-245.ec2.internal untainted
```

18) Check the status of pods.

```
[ec2-user@ip-172-31-20-245 docker]$ kubectl get pods
NAME    READY   STATUS    RESTARTS   AGE
nginx   1/1     Running   0          4m3s
```

```
[ec2-user@ip-172-31-20-245 docker]$ kubectl port-forward nginx 8081:80
Forwarding from 127.0.0.1:8081 -> 80
Forwarding from [::1]:8081 -> 80
```

Conclusion:

We successfully configured Kubernetes environment on Amazon Linux EC2 instance. We installed Docker and adjusted its settings to use systemd for cgroup management. Then, we installed Kubernetes. We desabled SELinux, and added the Kubernetes repository. Then we installed the necessary components. Then added a network plugin and deployed on Nginx server. We also addressed issues related to pod scheduling and port forwarding, ensuring the Nginx pod.

Brijesh Sharma Div:D15C Roll No:48

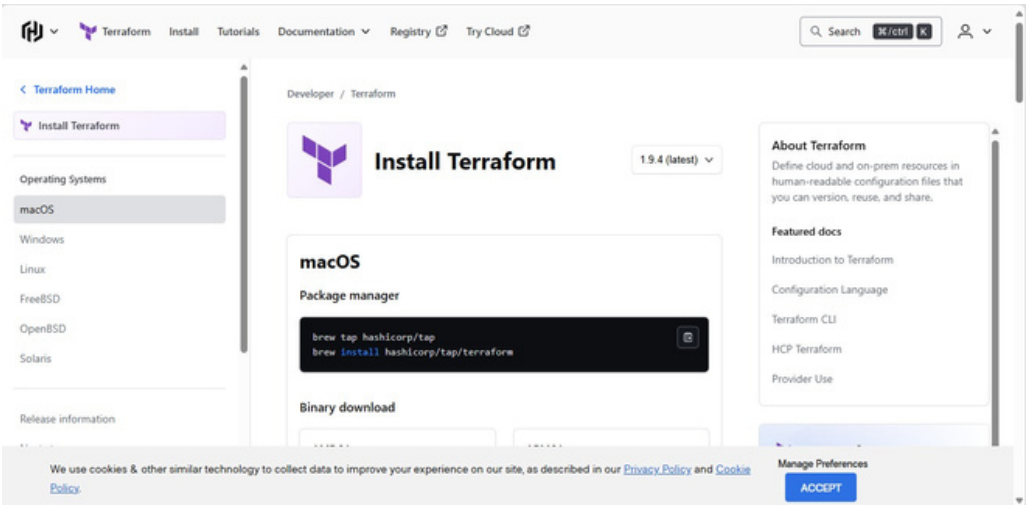
Experiment No: 5

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

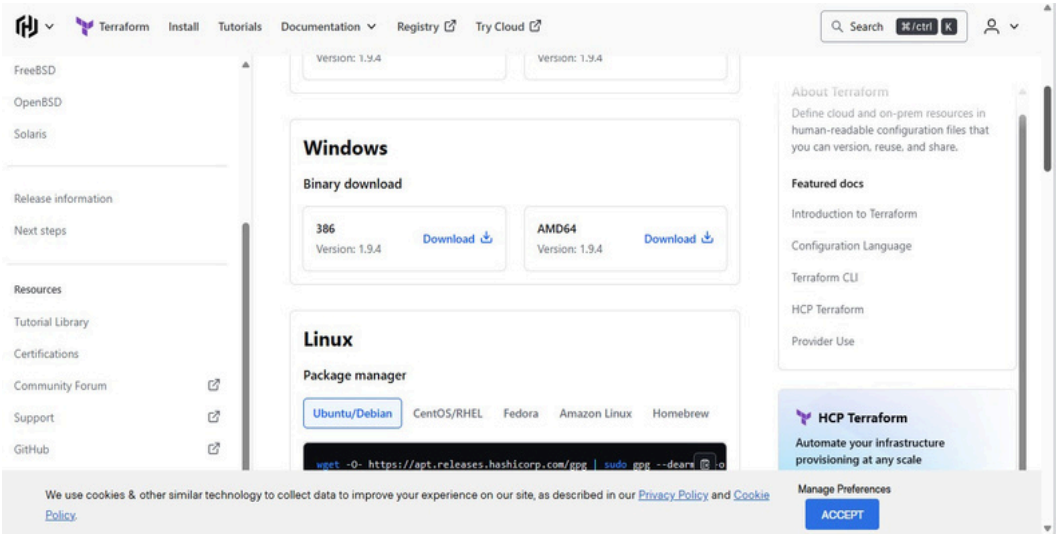
Installation and Configuration Steps of Terraform in Windows:

Step 1: Download terraform

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

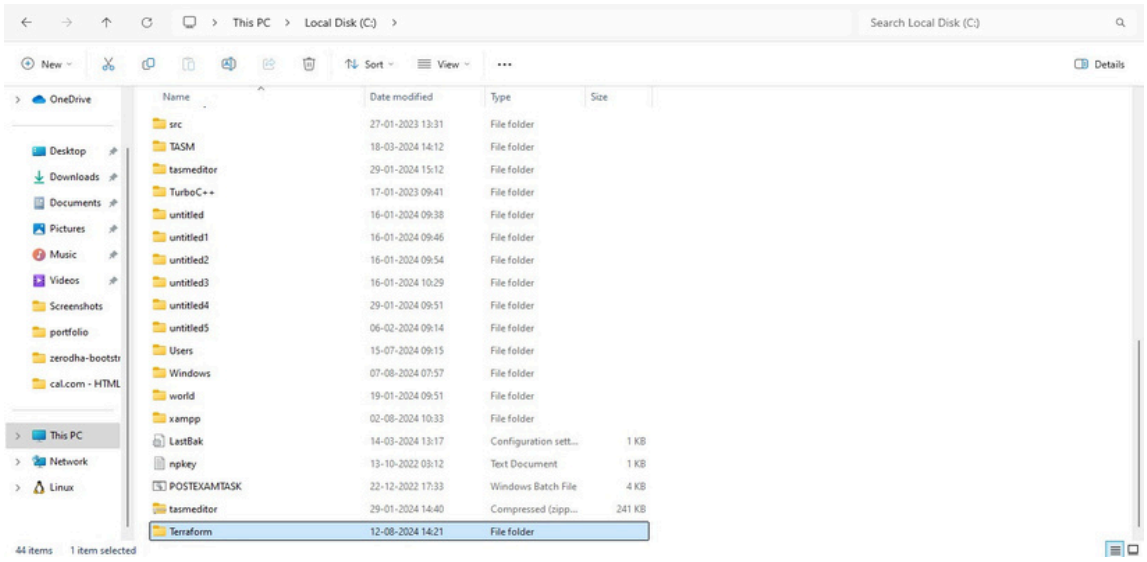


2)Select the Operating System Windows followed by either 32 bit or 64 bit based on your OS type.

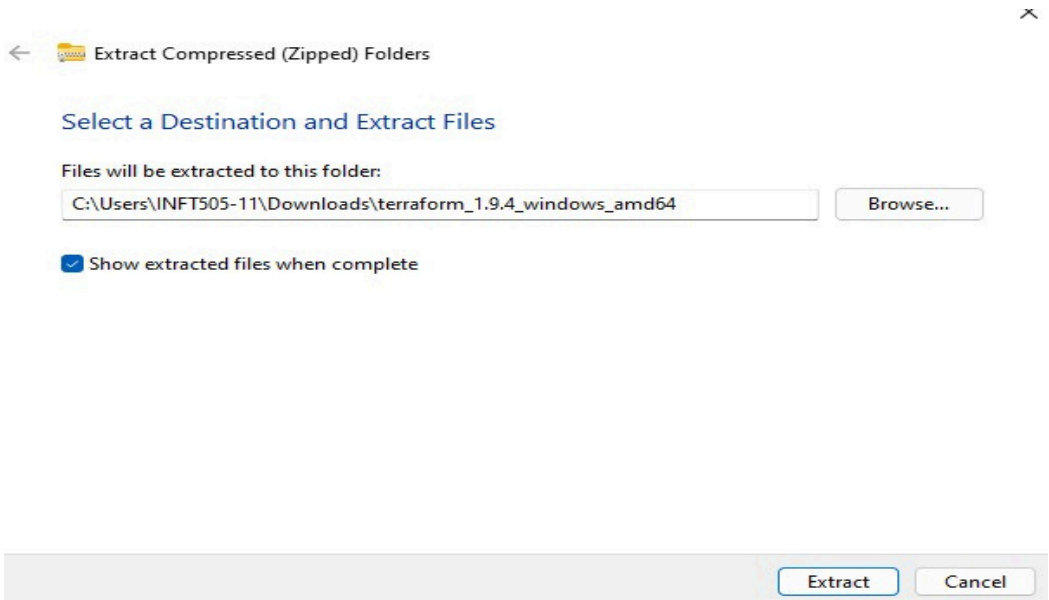


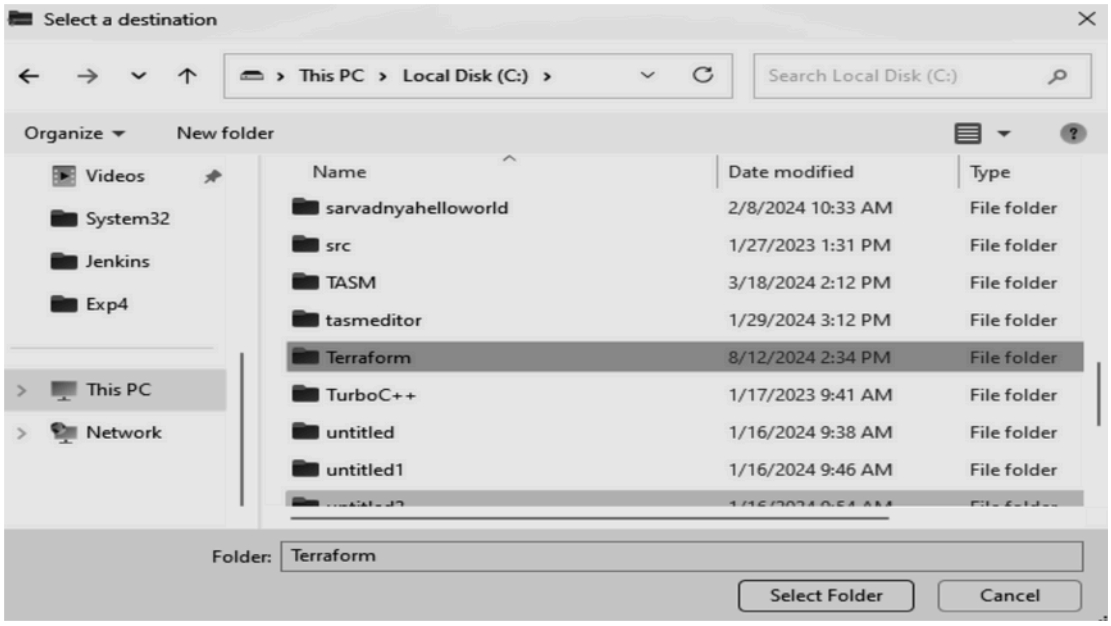
Step 2: Extract the downloaded setup file Terraform.exe in C:\Terraform Directory

1) Go to file manager -> Click on This PC -> move to C drive and create a new folder for example named terraform

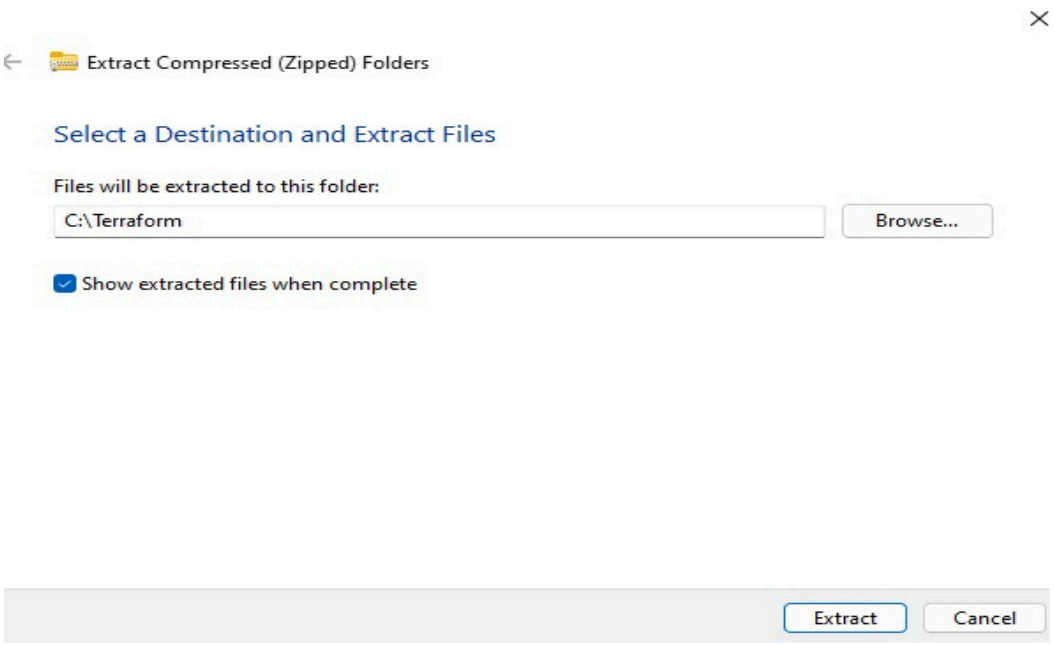


2) Got to location where main file is downloaded , right click on the file name and click on extract all while clicking on extract all it displays the location of the file to be saved , after extraction initially the destination of file to be extracted is not the folder which we have created earlier in previous steps , so to get the exact folder click on Browse and select it.

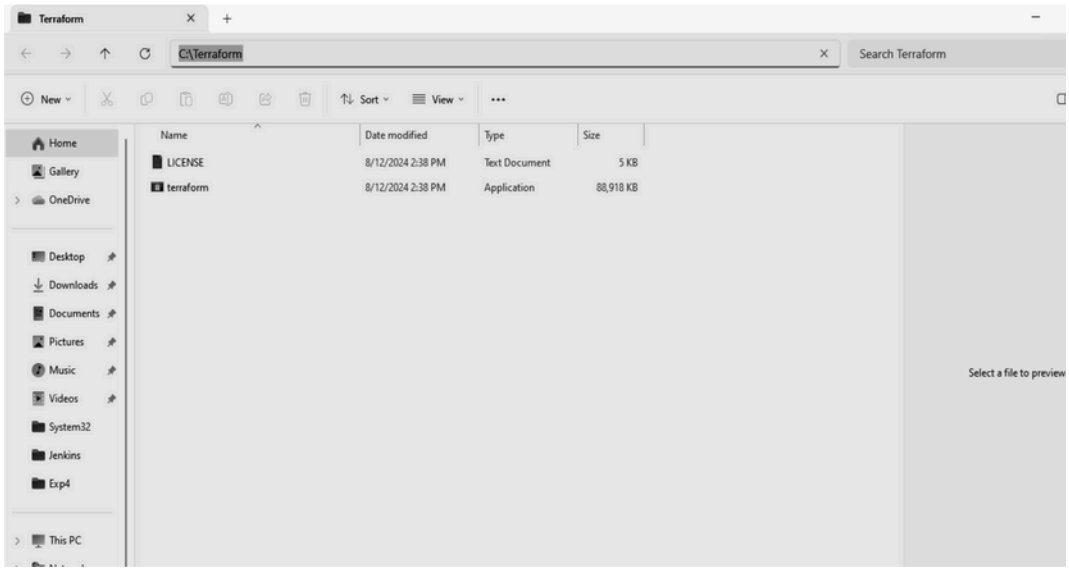




3) Click on extract Button.

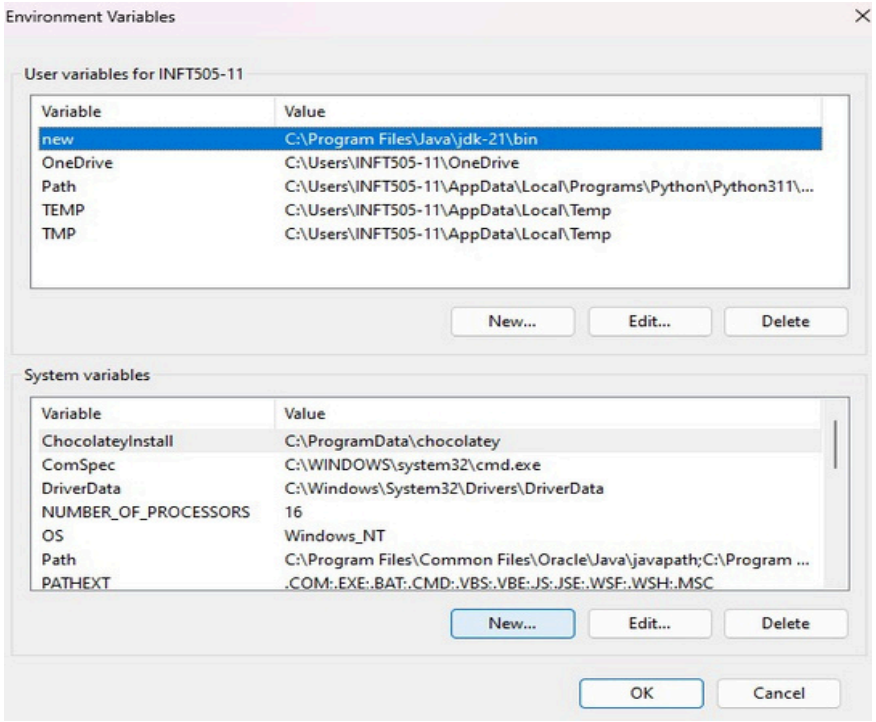


4) After extracting it will redirect to file manager copy the path which will be used for the further steps

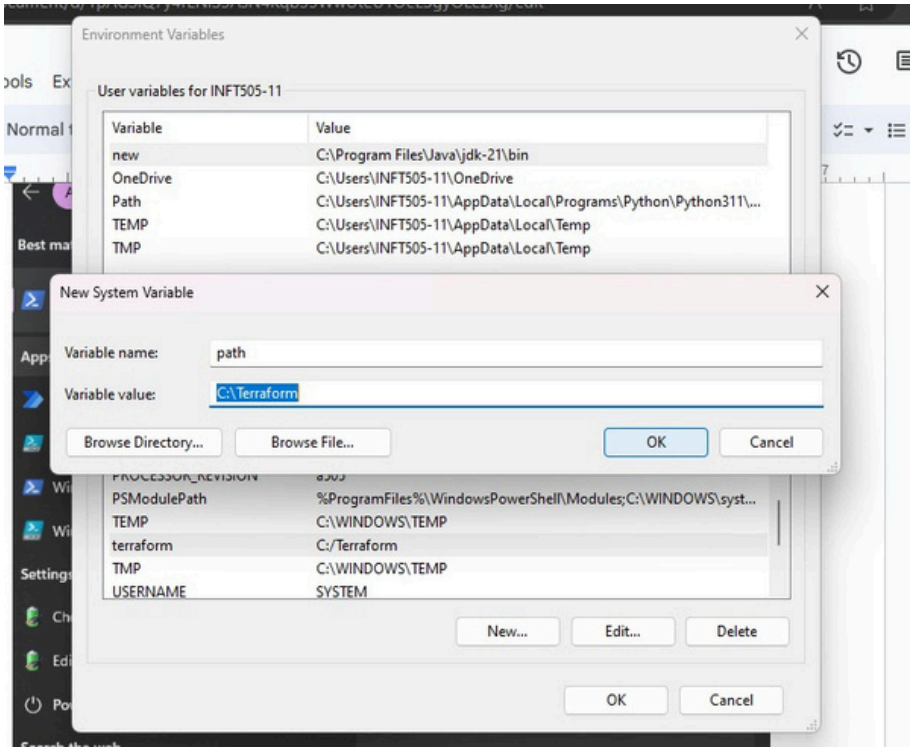


Step 3: Set the System path for Terraform in Environment Variables

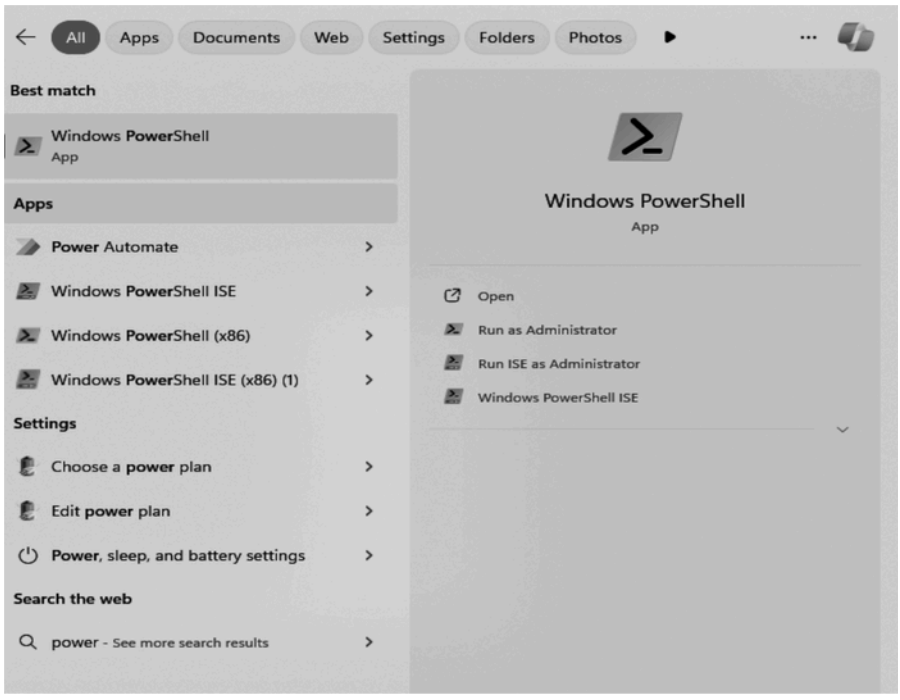
1) Search Environment Variable from search area in the Taskbar then Click on New under System Variables



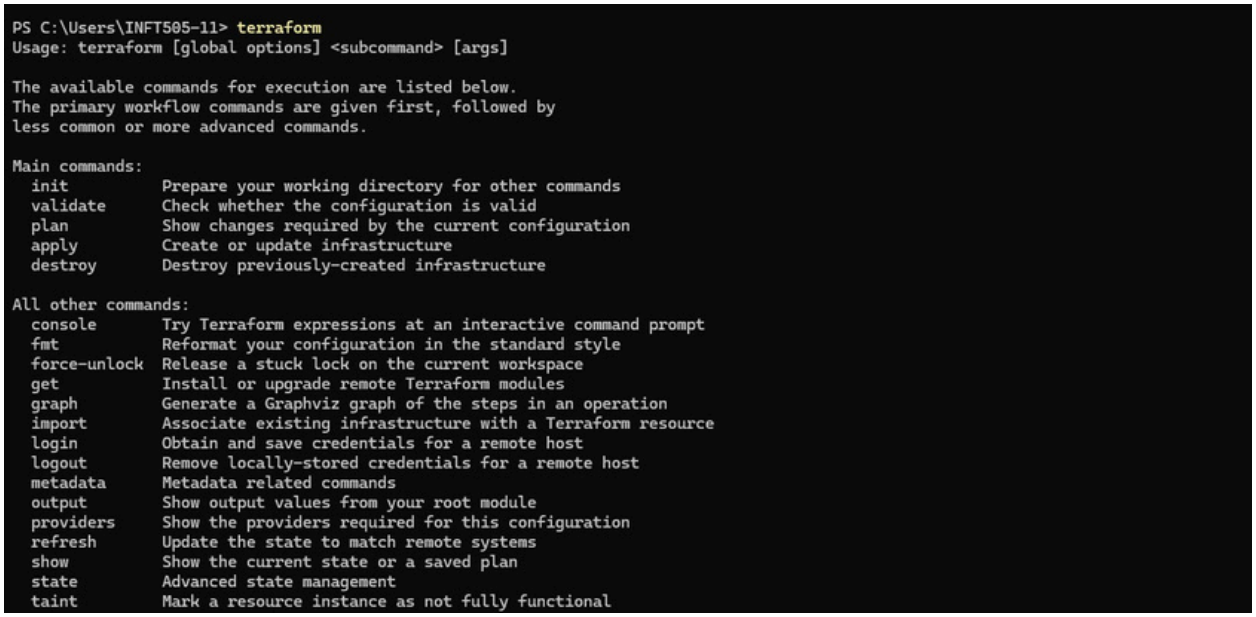
2) Set the Variable name as path and paste the location of the folder which we copied in previous step after extraction process as the variable value .Click on Ok



Step 4: Open PowerShell with Admin Access



Step 5 : Open Terraform in PowerShell and check its functionality



If the case of any Errors, then please recheck or set the path of Terraform in Environment variable again.

Name: Brijesh Sharma **Div: D15C** **Roll No. 48**

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

1) Check the docker version and functionality if its not downloaded you can download it from https://www.docker.com/
2) Use *docker --version* command to check for the version of docker.

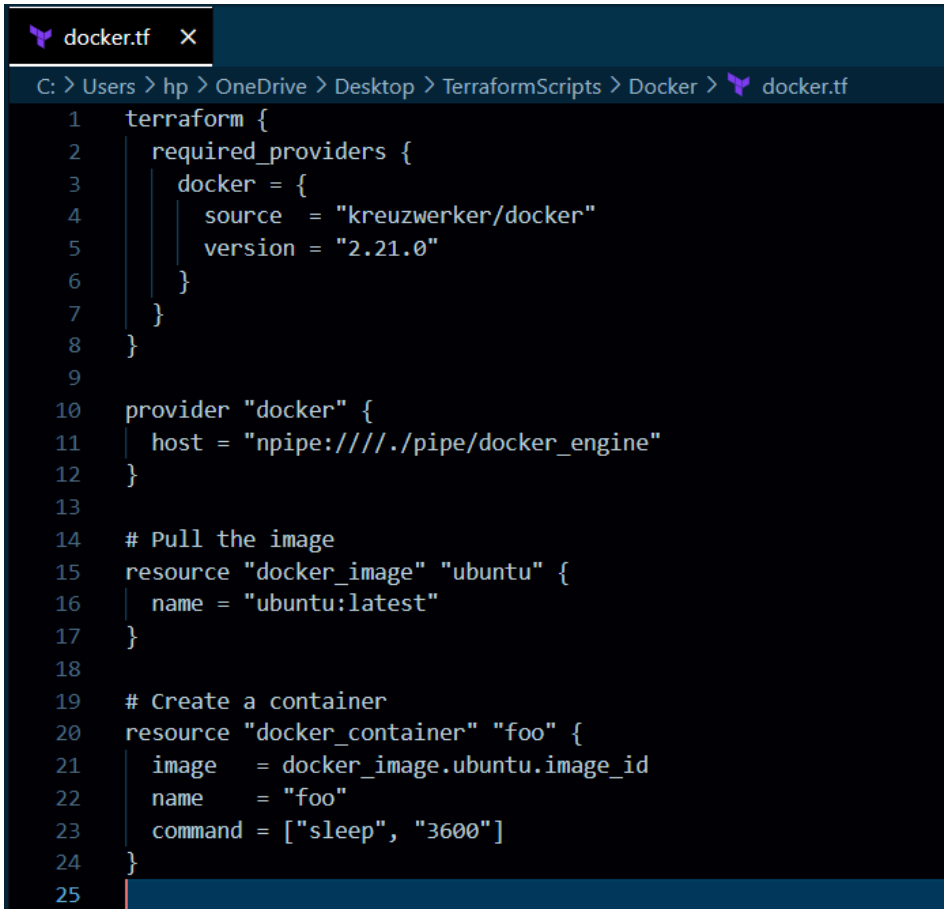
```
PS C:\Users\hp> docker --version
Docker version 27.0.3, build 7d4bcd8
```

```
PS C:\Users\hp> docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

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

Management Commands:
  builder     Manage builds
```

3) Create a folder named **'Terraform Scripts**' in which we save our different types of scripts which will be further used in this experiment.
4) Create a new docker.tf file using an IDE 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"]
}
```



5) Execute *terraform init* command to initialize the resources (Make sure you are in the Docker directory before executing the command)

```
PS C:\Users\hp\OneDrive\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.
```

6) Execute *terraform plan* to see the available resources if the build fails, there may be a chance that the docker engine is not running. Go to Docker Desktop and start the engine.

```
PS C:\Users\hp\OneDrive\Desktop\TerraformScripts\Docker> terraform plan

Planning failed. Terraform encountered an error while generating this plan.


  Error: Error pinging Docker server: error during connect: This error may indicate that the docker daemon is not running.: Get "http://%2F%2F.%2Fpipe%2Fdo
cker_engine/_ping": open //./pipe/docker_engine: The system cannot find the file specified.

    with provider["registry.terraform.io/kreuzwerker/docker"],
    on docker.tf line 10, in provider "docker":
    10: provider "docker" {
```

7) Again, run the terraform plan.

```
PS C:\Users\hp\OneDrive\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)
```

8) Execute *terraform apply* to apply the configuration, which will automatically create and run the Ubuntu Linux container based on our configuration.

```
PS C:\Users\hp\OneDrive\Desktop\TerraformScripts\Docker> terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with
he 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
```

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

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

  Enter a value: yes

docker_image.ubuntu: Creating...
docker_image.ubuntu: Still creating... [10s elapsed]
docker_image.ubuntu: Still creating... [20s elapsed]
docker_image.ubuntu: Still creating... [30s elapsed]
docker_image.ubuntu: Still creating... [40s elapsed]
docker_image.ubuntu: Still creating... [50s elapsed]
docker_image.ubuntu: Still creating... [1m0s elapsed]
docker_image.ubuntu: Still creating... [1m10s elapsed]
docker_image.ubuntu: Creation complete after 1m11s [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad
761c2598aubuntu:latest]
docker_container.foo: Creating...
docker_container.foo: Still creating... [10s elapsed]
docker_container.foo: Still creating... [20s elapsed]
docker_container.foo: Still creating... [30s elapsed]
docker_container.foo: Creation complete after 34s [id=0461fcf8f00556f61c080a147e0d0b33687fc09868abbad5c3e8205f6f6b033
1]
```

9) Docker images before executing this command

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

Docker images after executing this command

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

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

```
PS C:\Users\hp\OneDrive\Desktop\TerraformScripts\Docker> terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=0461fcf8f00556f61c080a147e0d0b33687fc09868abbad5c3e8205f6f6b0331]

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          = "0461fcf8f005" -> null
      - id                = "0461fcf8f00556f61c080a147e0d0b33687fc09868abbad5c3e8205f6f6b0331" -> 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
```

```
  # 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=0461fcf8f00556f61c080a147e0d0b33687fc09868abbad5c3e8205f6f6b0331]
docker_container.foo: Still destroying... [id=0461fcf8f00556f61c080a147e0d0b33687fc09868abbad5c3e8205f6f6b0331, 10s elapsed]
docker_container.foo: Still destroying... [id=0461fcf8f00556f61c080a147e0d0b33687fc09868abbad5c3e8205f6f6b0331, 20s elapsed]
docker_container.foo: Still destroying... [id=0461fcf8f00556f61c080a147e0d0b33687fc09868abbad5c3e8205f6f6b0331, 30s elapsed]
docker_container.foo: Still destroying... [id=0461fcf8f00556f61c080a147e0d0b33687fc09868abbad5c3e8205f6f6b0331, 40s elapsed]
docker_container.foo: Destruction complete after 44s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image.ubuntu: Destruction complete after 7s

Destroy complete! Resources: 2 destroyed.
```

11) Docker images after the destroy command execution

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

Name - Brijesh Sharma Div:D15C Roll No.48

Experiment 7

Aim:

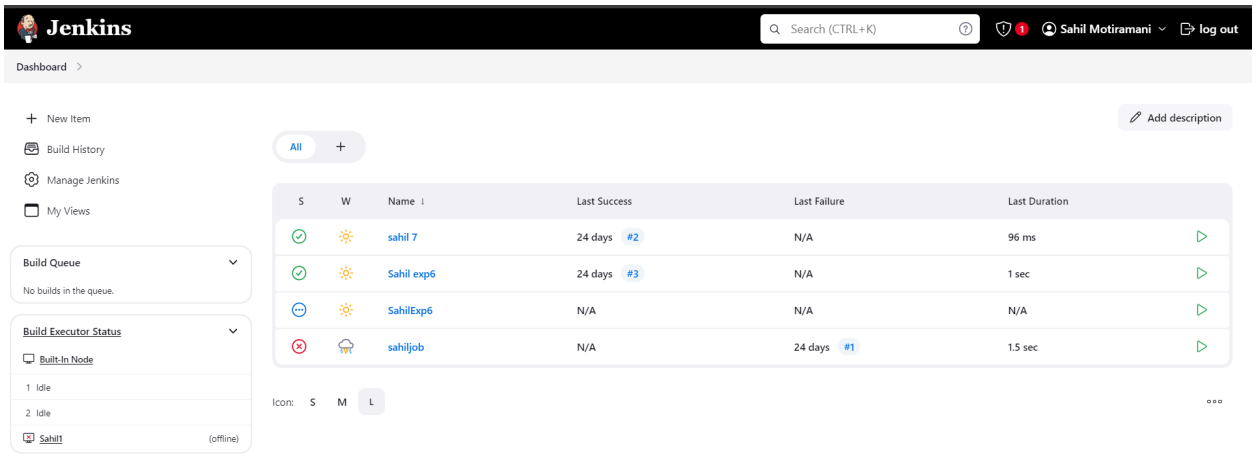
To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

Integrating Jenkins with SonarQube:

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

Steps to integrate Jenkins with SonarQube
Prerequisites: Make sure you have docker and jenkins installed.
Run **docker -v** to check the docker installation.
Run

1) Open up Jenkins Dashboard on localhost, port 8090 or whichever port it is at for you.



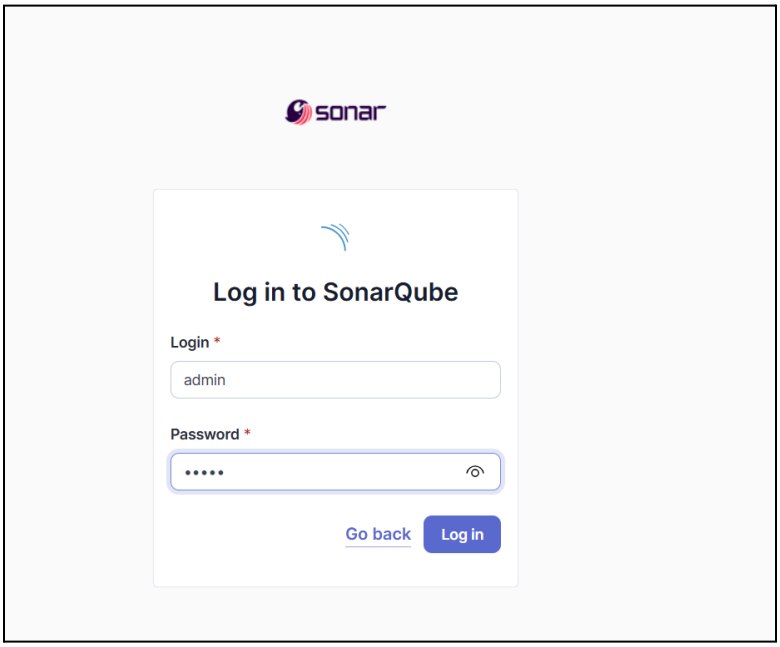
2) Run SonarQube in a Docker container using this command -

docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

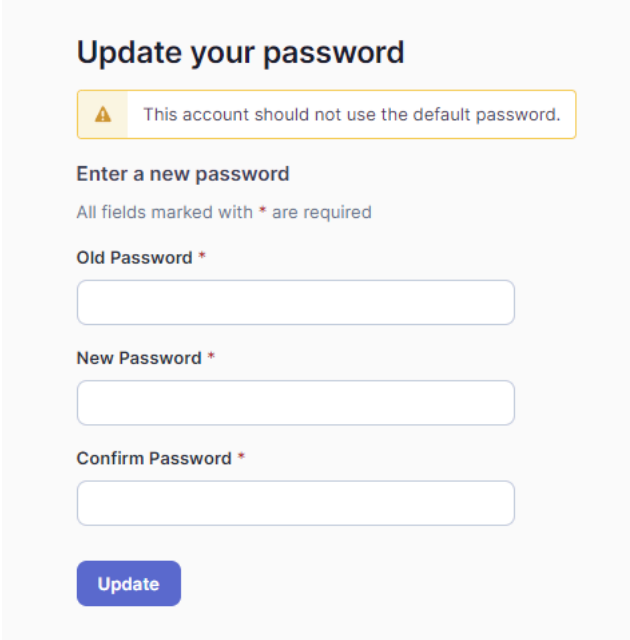
----------Warning: run below command only once

```
C:\Users\Lenovo>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
47f6db8dbf2ed99dbe304bc0ebdf47b9d4144c4e4add42055ba44ce231058272
```

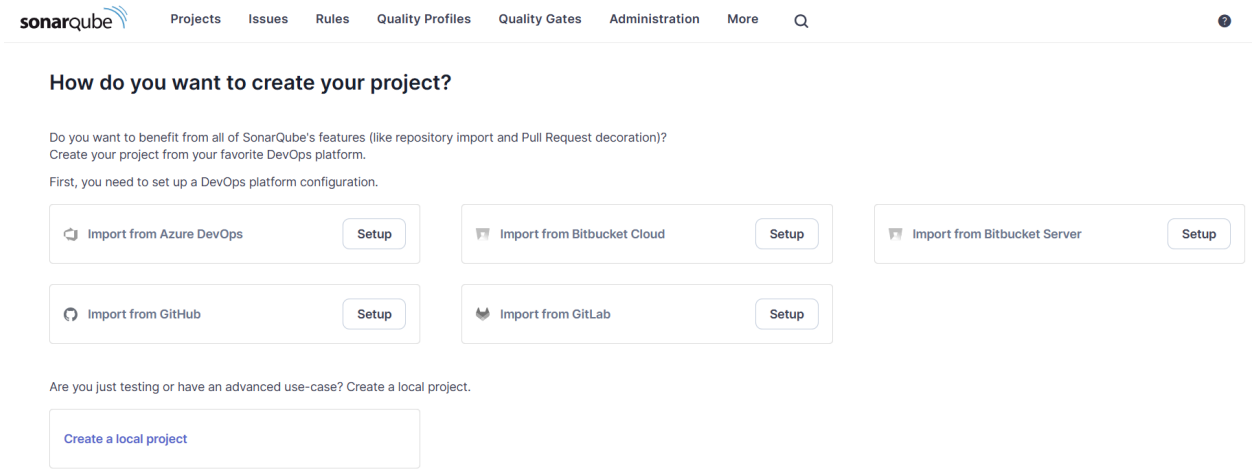
3) Once the container is up and running, you can check the status of SonarQube at localhost port 9000.



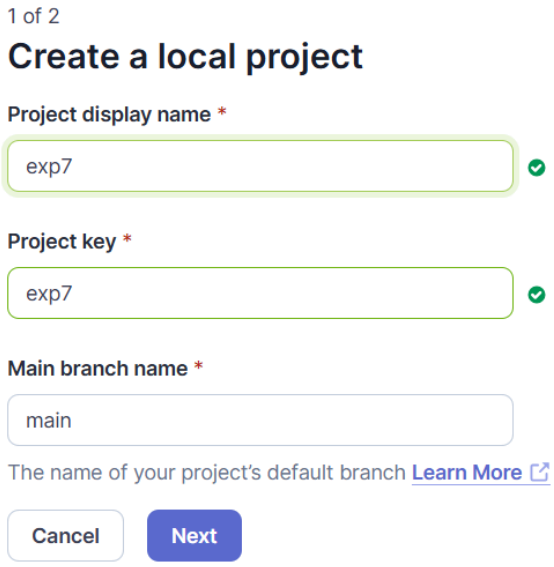
4) Login to SonarQube using username - *admin* and password - *admin*. (do change the password as you cannot use the default one)



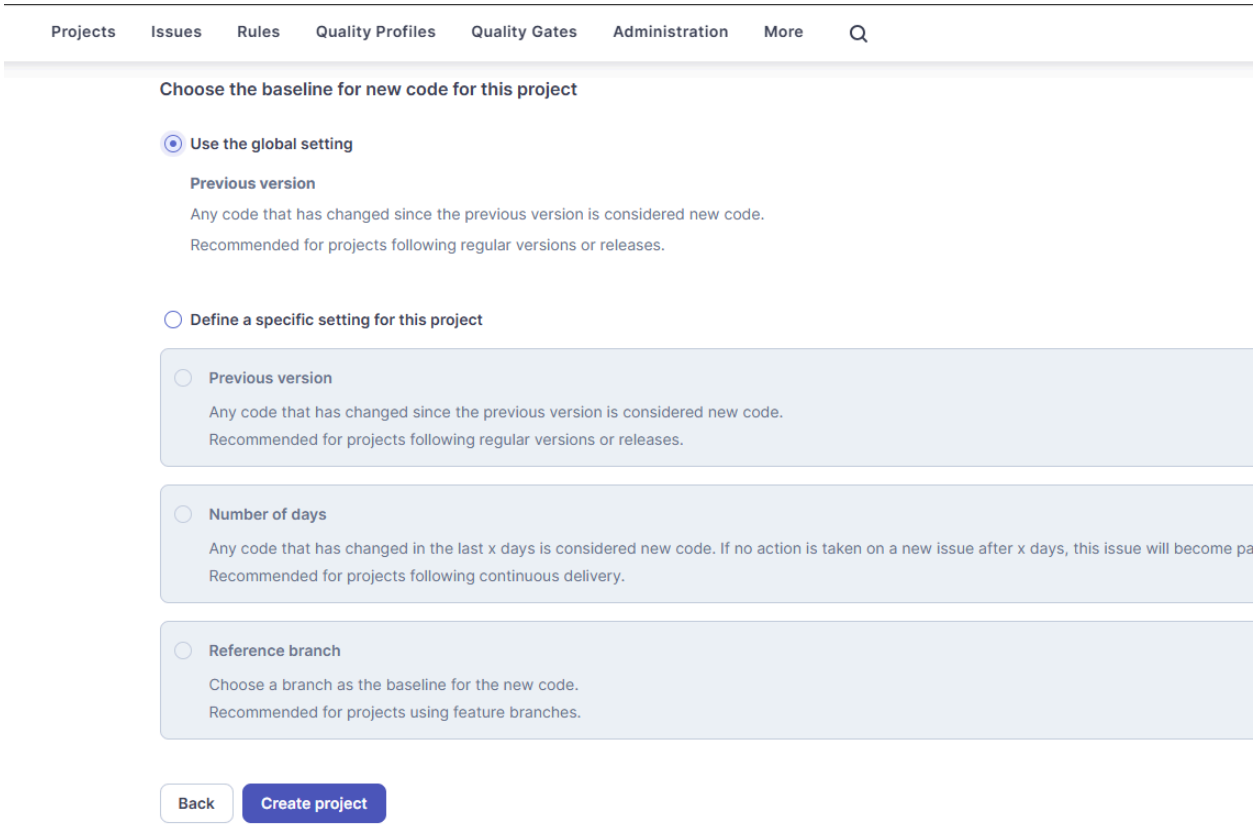
Name - Brijesh Sharma Div:D15C Roll No.48



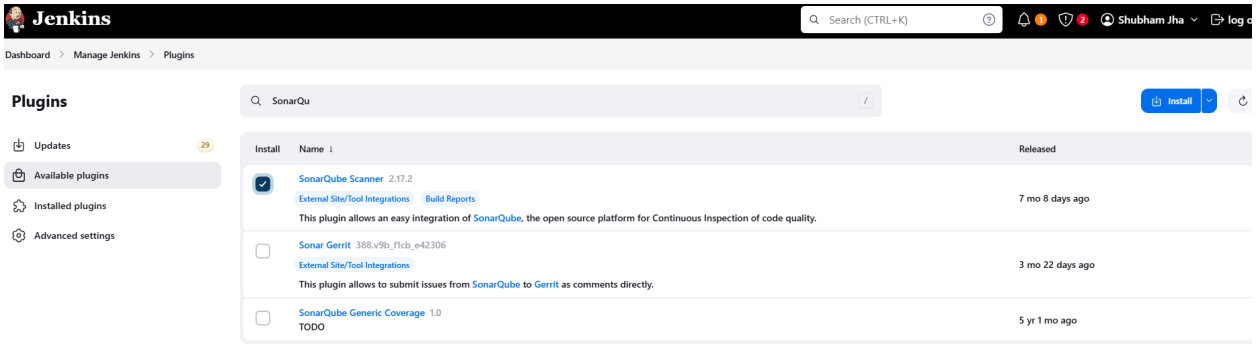
5) Create a manual project in SonarQube with the name sonarqube (Click on create local project)



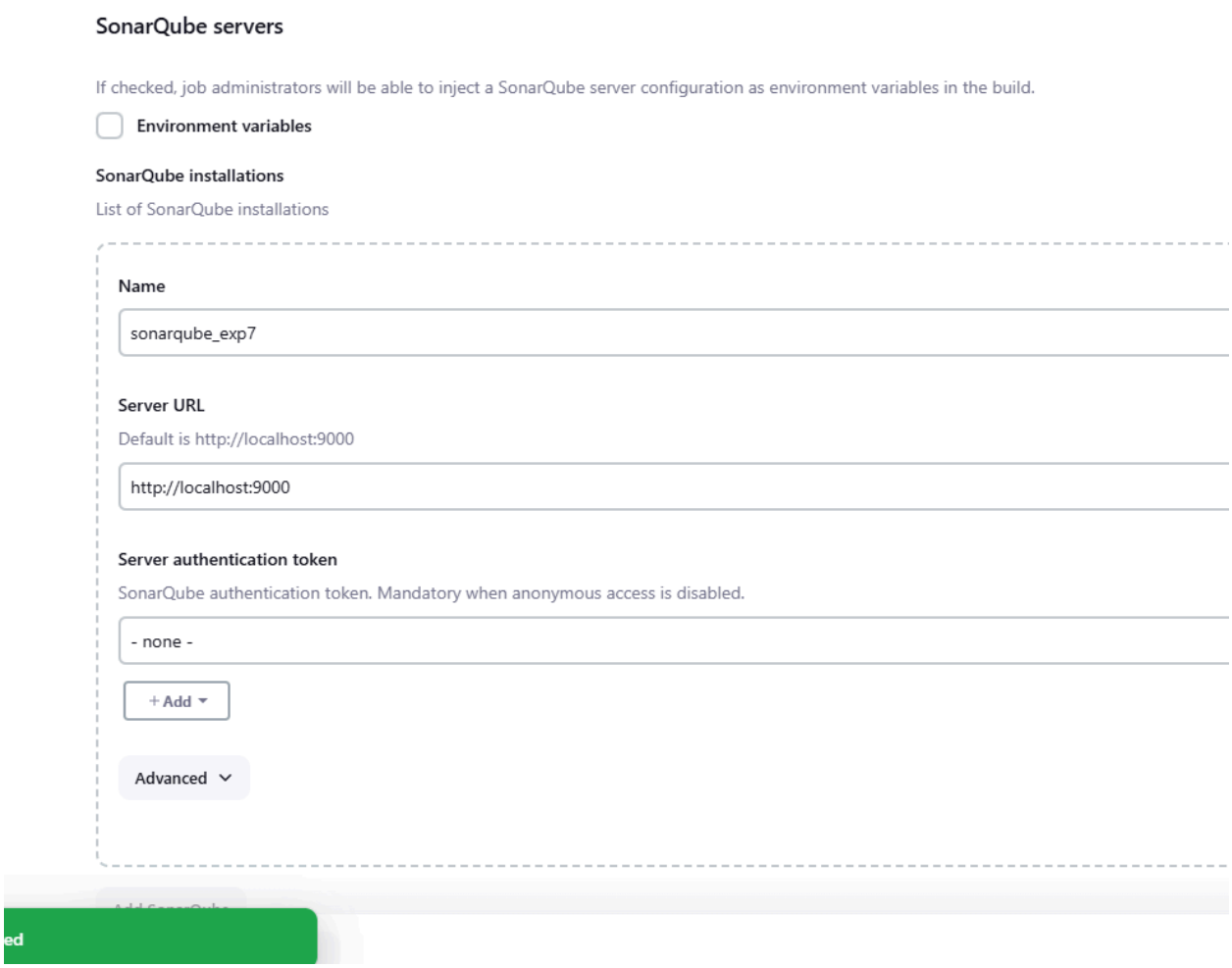
6) Setup the project and come back to Jenkins Dashboard.



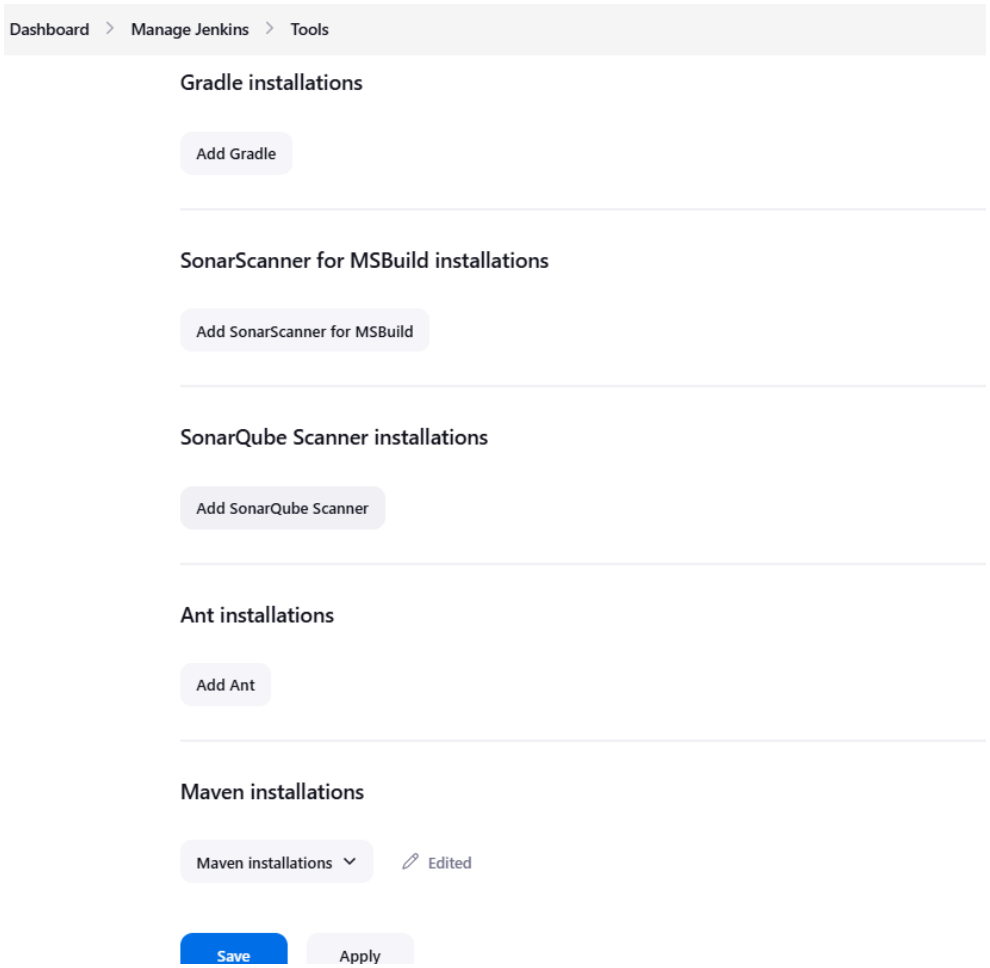
7) Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.



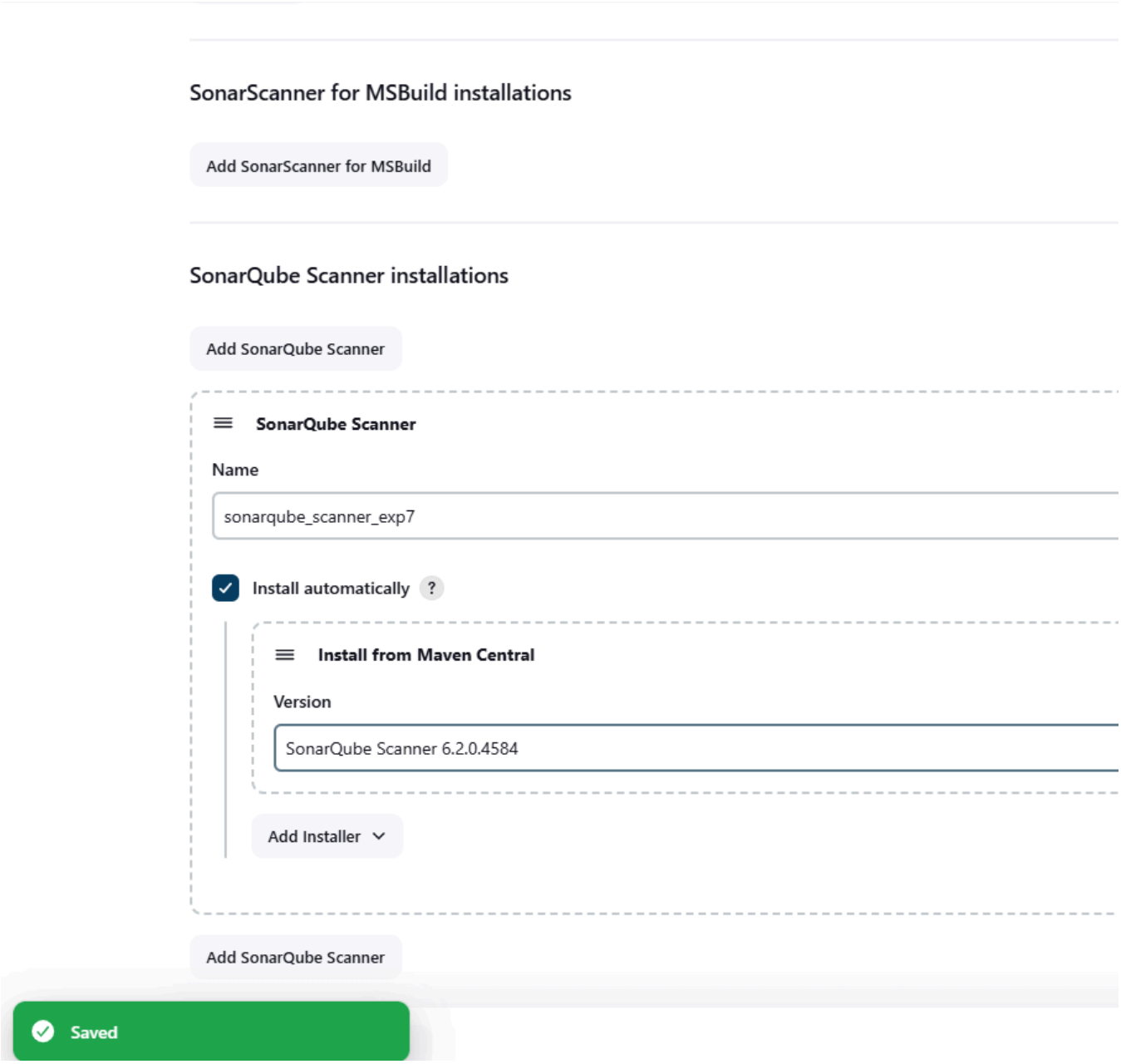
8) Under Jenkins ‘Manage Jenkins’ then go to ‘system’, scroll and look for **SonarQube Servers** and click on **add SonarQube** and then enter the details.
Enter the Server Authentication token if needed.(I didn’t do it)
In SonarQube installations: Under **Name** add <project name of sonarqube> for me its sonarqube_exp7
In **Server URL** Default is **http://localhost:9000**



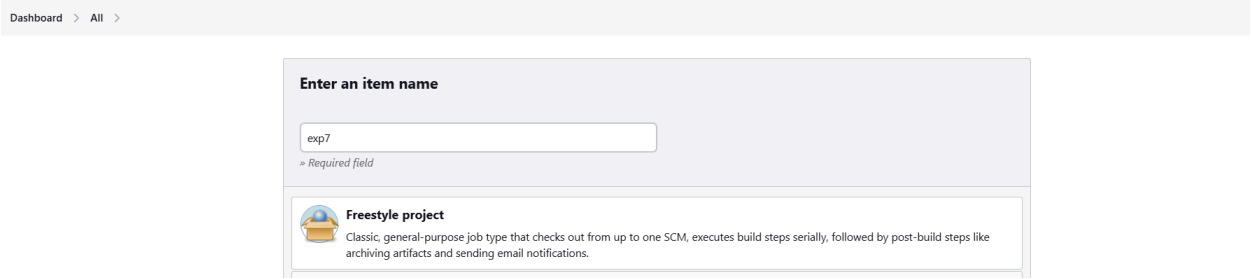
9) Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.
Dashboard > Manage Jenkins > Tools



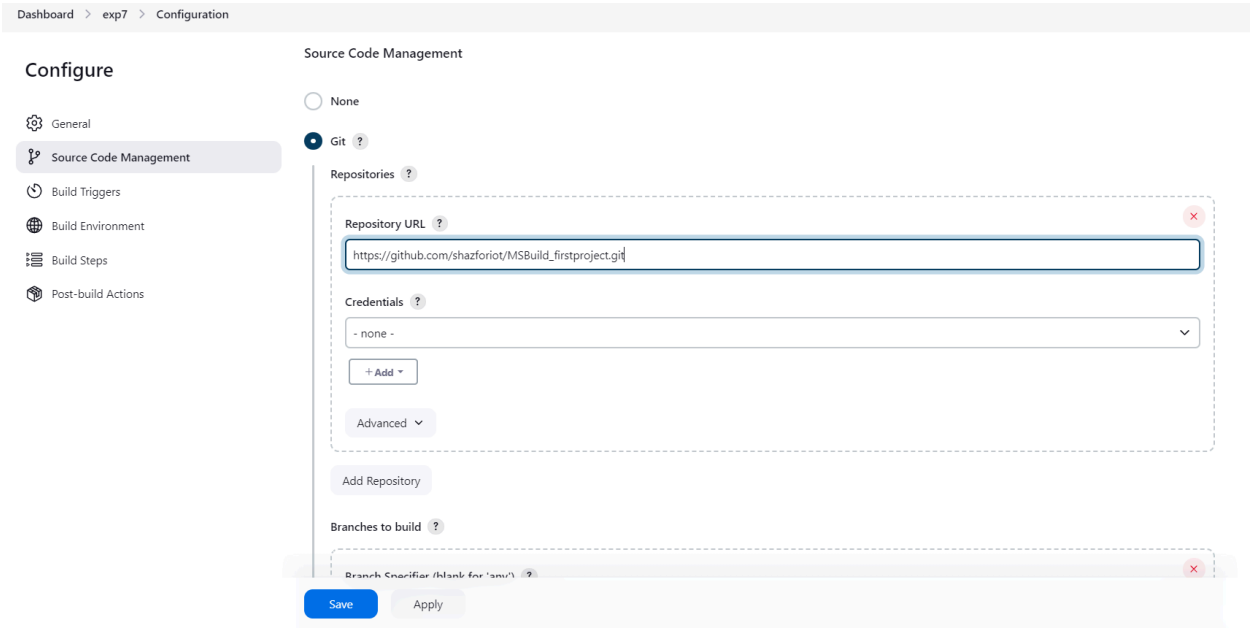
Click on **Add SonarQube Scanner** .
Check the “Install automatically” option. → Under name write any name as identifier →
Check the “Install automatically” option.



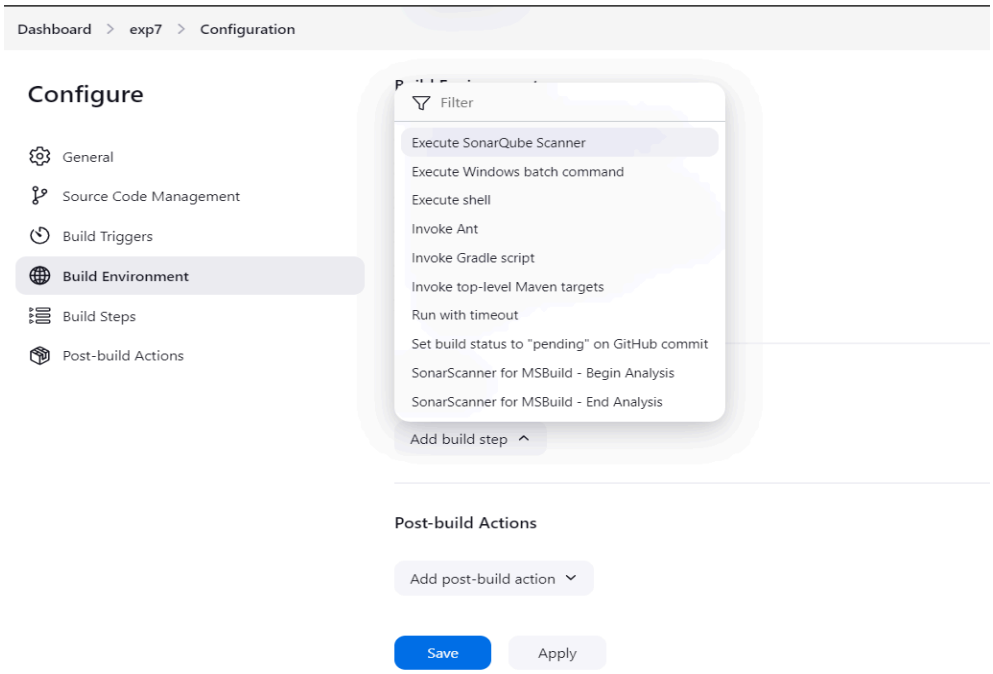
10) After the configuration, create a New Item in Jenkins, choose a freestyle project.



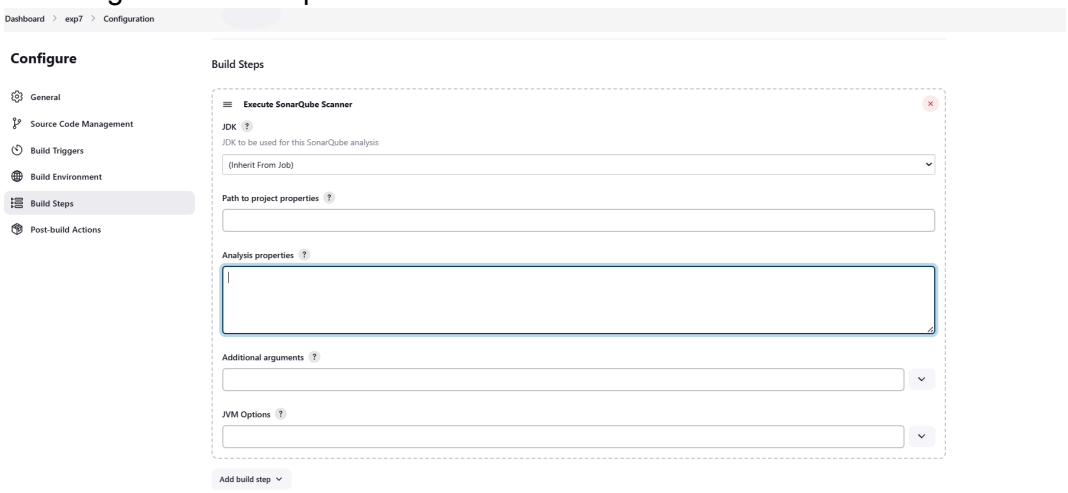
11) Choose this GitHub repository in Source Code Management.
https://github.com/shazforiot/MSBuild_firstproject
It is a sample hello-world project with no vulnerabilities and issues, just to test the integration.



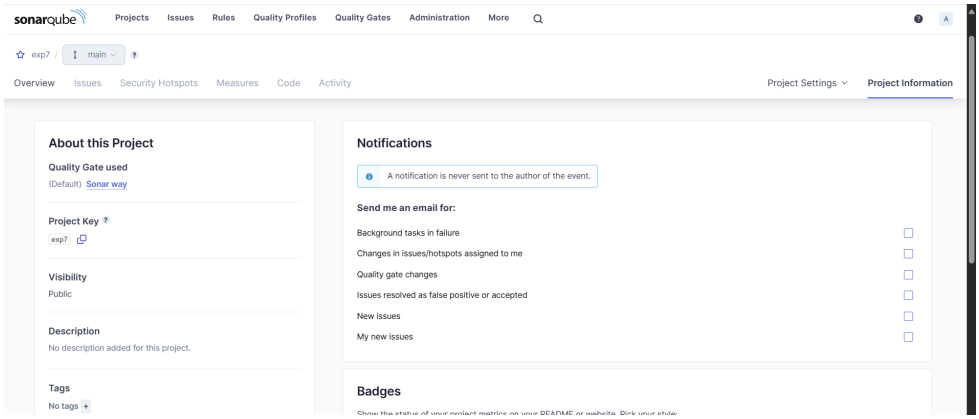
12) Under **Select project → Configuration → Build steps → Execute SonarQube Scanner**, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.



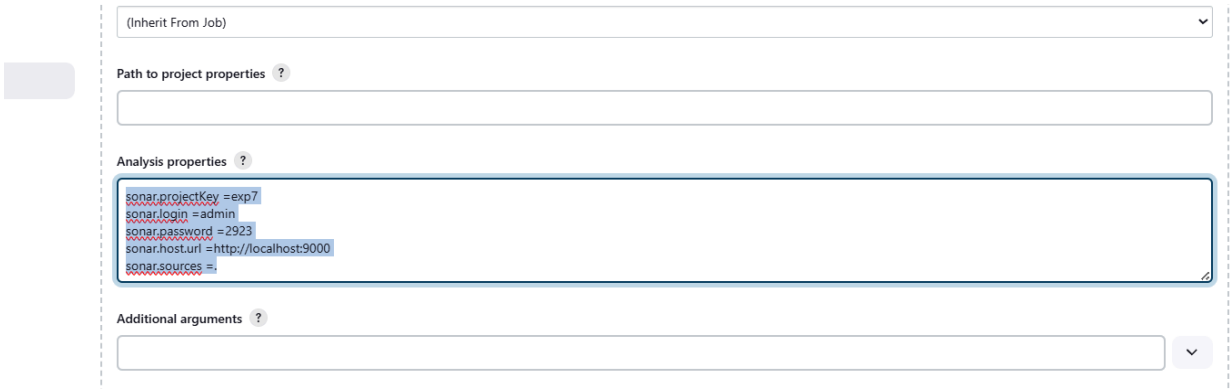
Following window will open -



Open sonarQube again and go to Project Information appearing in the right side. Click on it and you can copy the project key from About the Project Section.

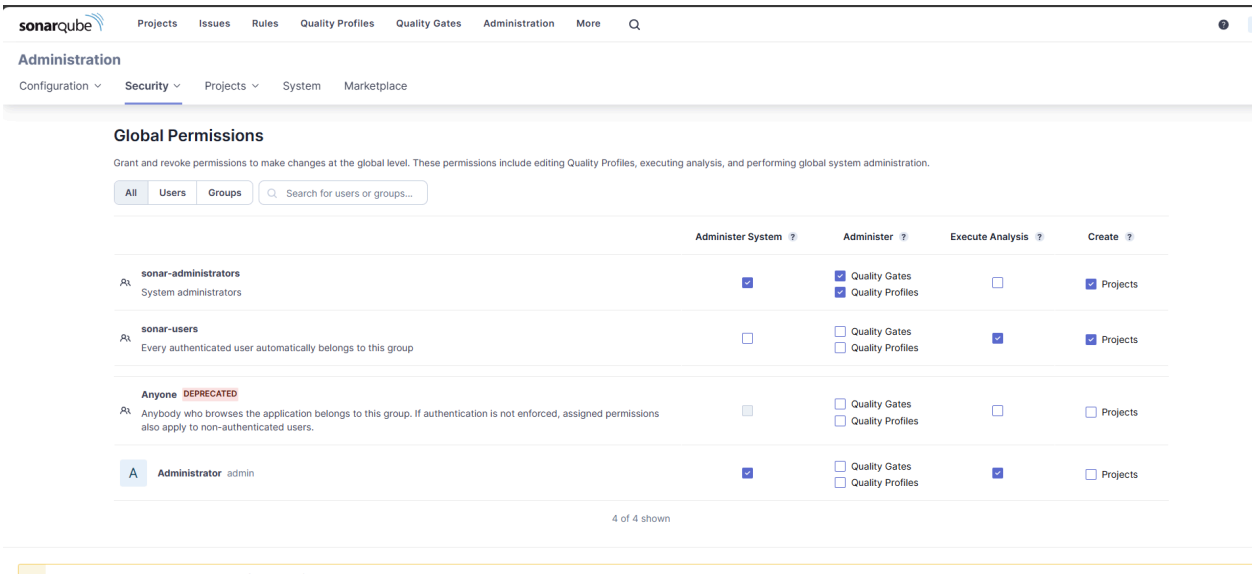


Use this key in place of <projectKey> in the following code
sonar.projectKey =<projectKey>
sonar.login =admin
sonar.password =<yourpassword for sonar qube>
sonar.host.url =http://localhost:9000
sonar.sources =

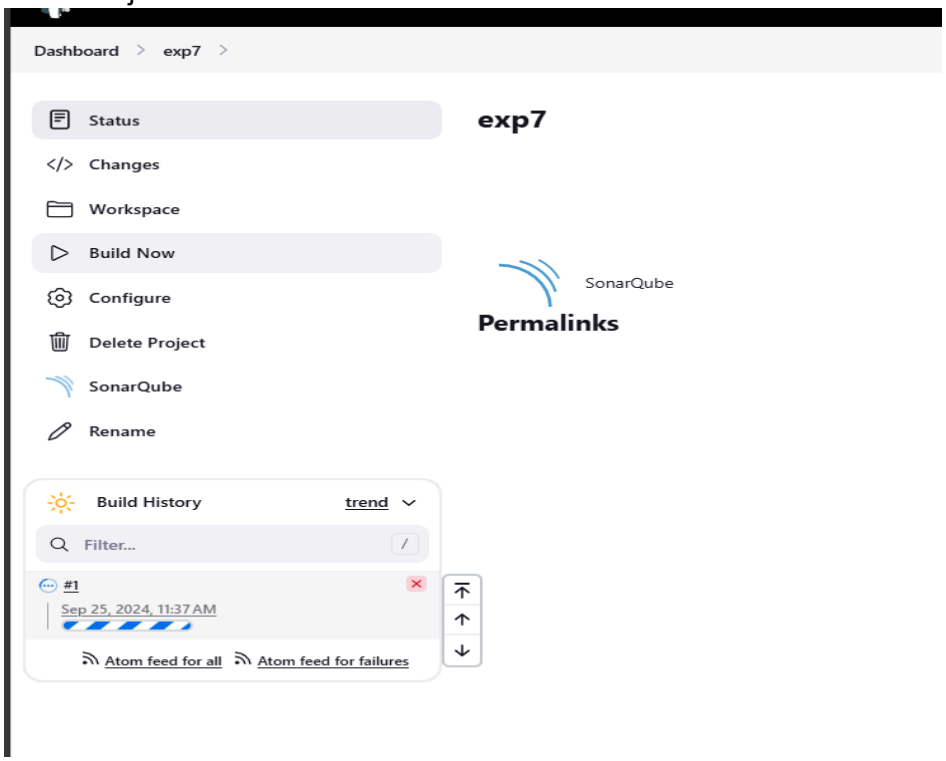


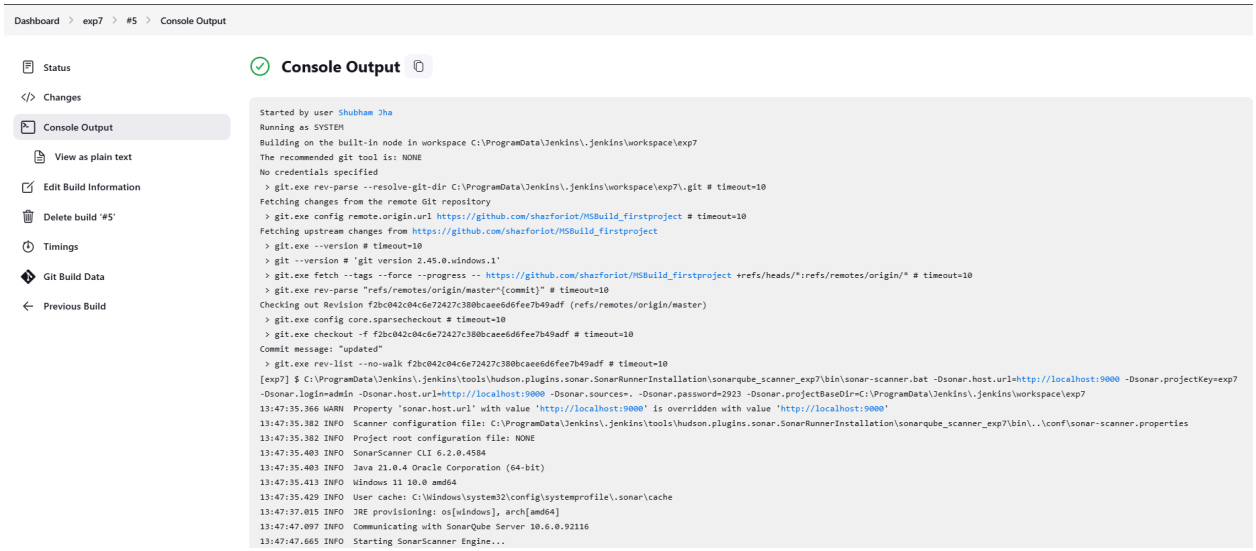
Apply and save.

13) Go to sonarQube and go to administration → Security (dropdown) → Global Permissions.
See the administrator below and check the boxes as checked below..



12. Go to jenkins and click build:





Conclusion:

In this project, we successfully integrated Jenkins with SonarQube to establish a robust automated static application security testing (SAST) pipeline. The setup involved deploying SonarQube using Docker, ensuring smooth container orchestration and efficient resource management. A key component was configuring Jenkins with the appropriate SonarQube plugins, authentication mechanisms, and linking it to a GitHub repository for continuous integration.

One of the challenges was configuring Docker on the Jenkins environment, which required resolving networking issues between the Docker containers and ensuring that the SonarQube server was reachable from Jenkins. Additionally, setting up secure authentication between Jenkins and SonarQube involved troubleshooting token-based authentication and resolving environment path issues, particularly with the **JAVA_HOME** setup for the SonarQube scanner.

After overcoming these obstacles, I integrated the SonarQube scanner as a build step, allowing for continuous code analysis. This setup provided automated detection of code vulnerabilities, code smells, and quality issues. It helped ensure that any new commits triggered immediate analysis, generating detailed reports and promoting continuous improvement in code quality.

Name:Brijesh Sharma **Roll No: 48** **Division: D15C**

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

Theory:

What is SAST?

Static Application Security Testing (SAST) is a methodology that analyzes source code to identify security vulnerabilities before compilation. It is often referred to as white box testing and helps developers detect issues early in the software development lifecycle (SDLC).

Problems SAST Solves

1. **Early Detection**: Identifies vulnerabilities in the initial development stages, reducing later risks.
2. **Real-Time Feedback**: Provides immediate insights, allowing developers to fix issues before moving forward.
3. **Code Navigation**: Offers visual representations of vulnerabilities for easier code understanding.
4. **Guidance on Fixes**: Suggests specific remediation steps without requiring deep security expertise.
5. **Comprehensive Coverage**: Analyzes the entire codebase quickly, outperforming manual reviews.
6. **Regular Scanning**: Ensures continuous security assessment through scheduled scans during builds or releases.

Importance of SAST

- **Resource Efficiency**: Automates code reviews, addressing the resource gap between developers and security staff.
- **Speed**: Processes millions of lines of code in minutes, identifying critical vulnerabilities.
- **Proactive Security**: Integrates security into the development process, preventing vulnerabilities from being overlooked.

What is a CI/CD Pipeline?

A **CI/CD Pipeline** refers to Continuous Integration and Continuous Delivery, automating software development tasks. It includes stages such as coding, building, testing, and deploying, ensuring each step is completed sequentially for efficient releases.

What is SonarQube?

SonarQube is an open-source platform for continuous code quality inspection. It performs static code analysis to generate reports on bugs, vulnerabilities, and code duplications across various programming languages.

Name:Brijesh Sharma **Roll No: 48** **Division: D15C**

Benefits of SonarQube

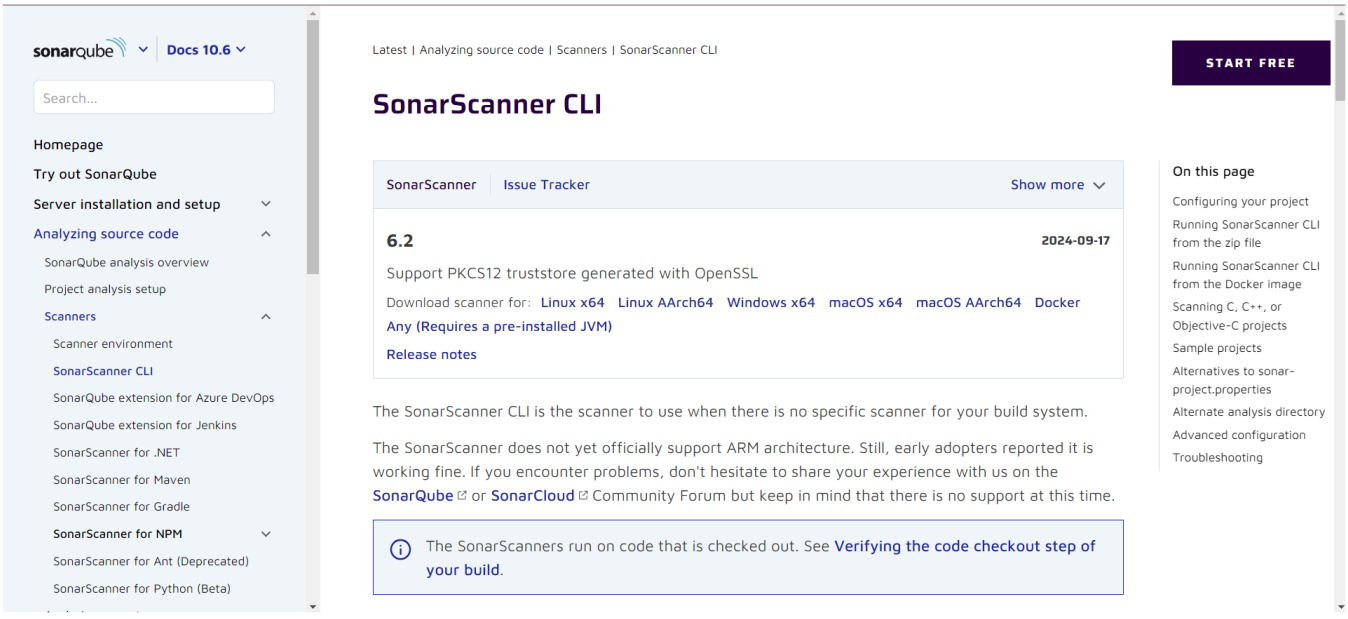
- **Sustainability**: Optimizes application lifecycle by reducing complexity and vulnerabilities.
- **Increased Productivity**: Minimizes maintenance efforts and costs.
- **Quality Control**: Integrates code quality checks into development.
- **Error Detection**: Alerts developers to fix issues before release.
- **Scalability**: Supports multiple projects without restrictions.
- **Skill Enhancement**: Provides regular feedback to improve developer skills.

Prerequisites:

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

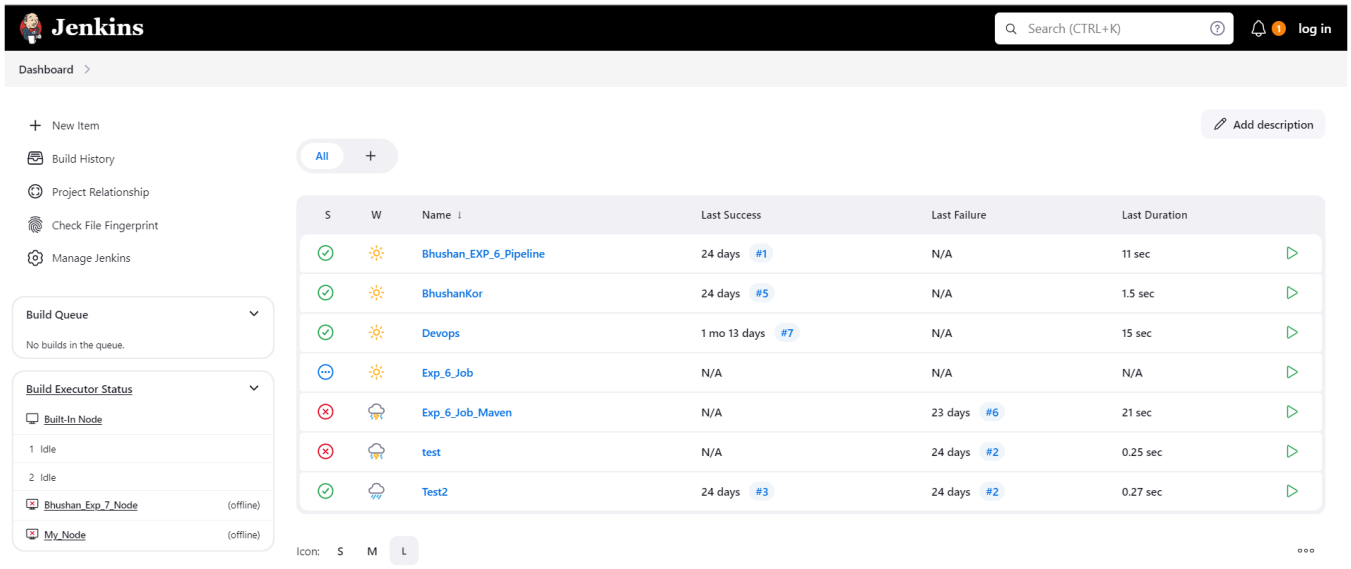
Download The SonarQube CLI according to your system :

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



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

Name:Brijesh Sharma **Roll No: 48** **Division: D15C**

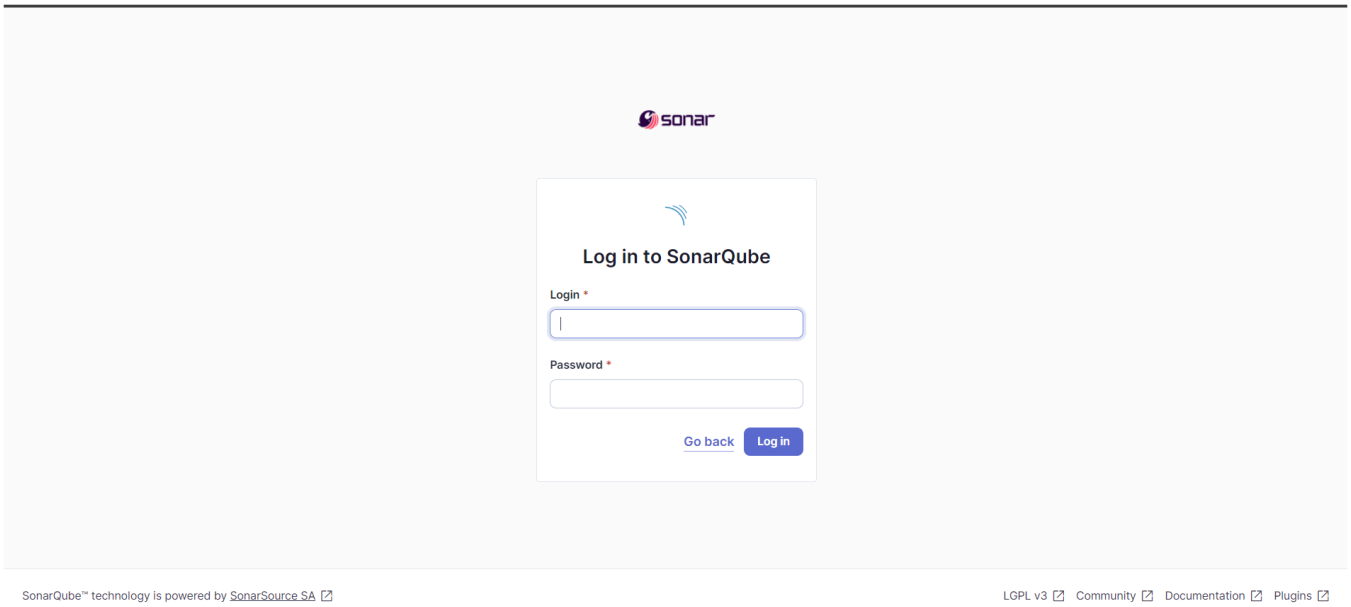


Step 2: Run SonarQube in a Docker container using this command
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

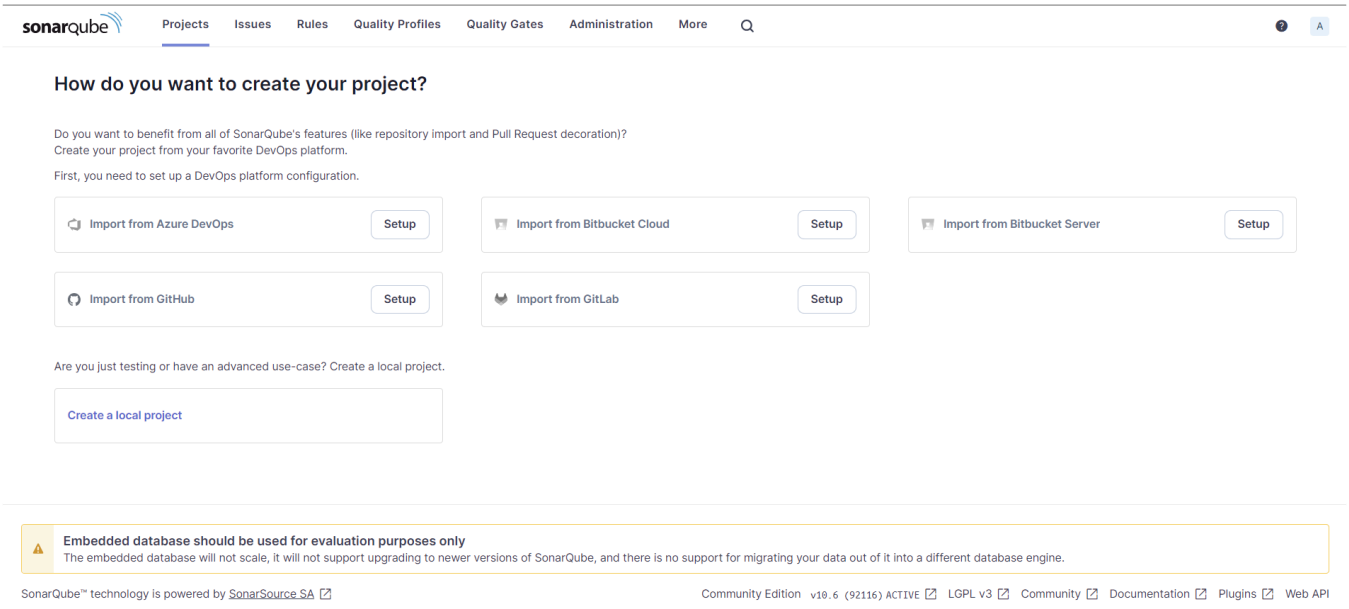
```
Microsoft Windows [Version 10.0.22621.4169]
(c) Microsoft Corporation. All rights reserved.

C:\Users\INFT505-11>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:late
st
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
d72b183b1866cea7ecdb976a63dfe521172c307eb45eace7b769f726f0bbf989

C:\Users\INFT505-11>
```

Name:Brijesh Sharma **Roll No: 48** **Division: D15C**

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

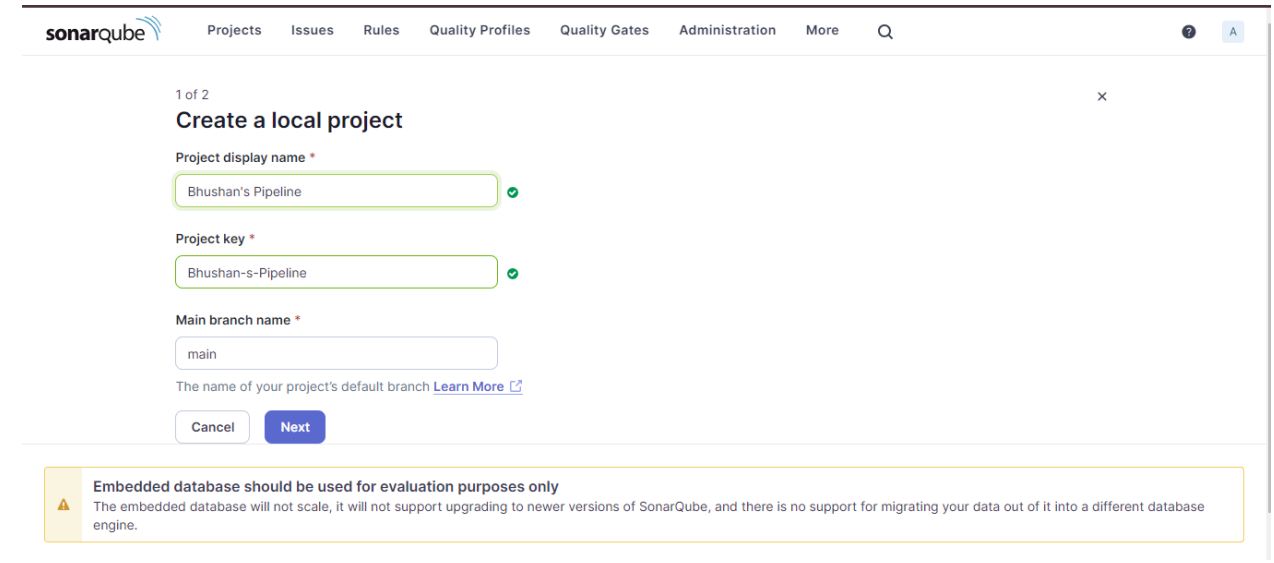


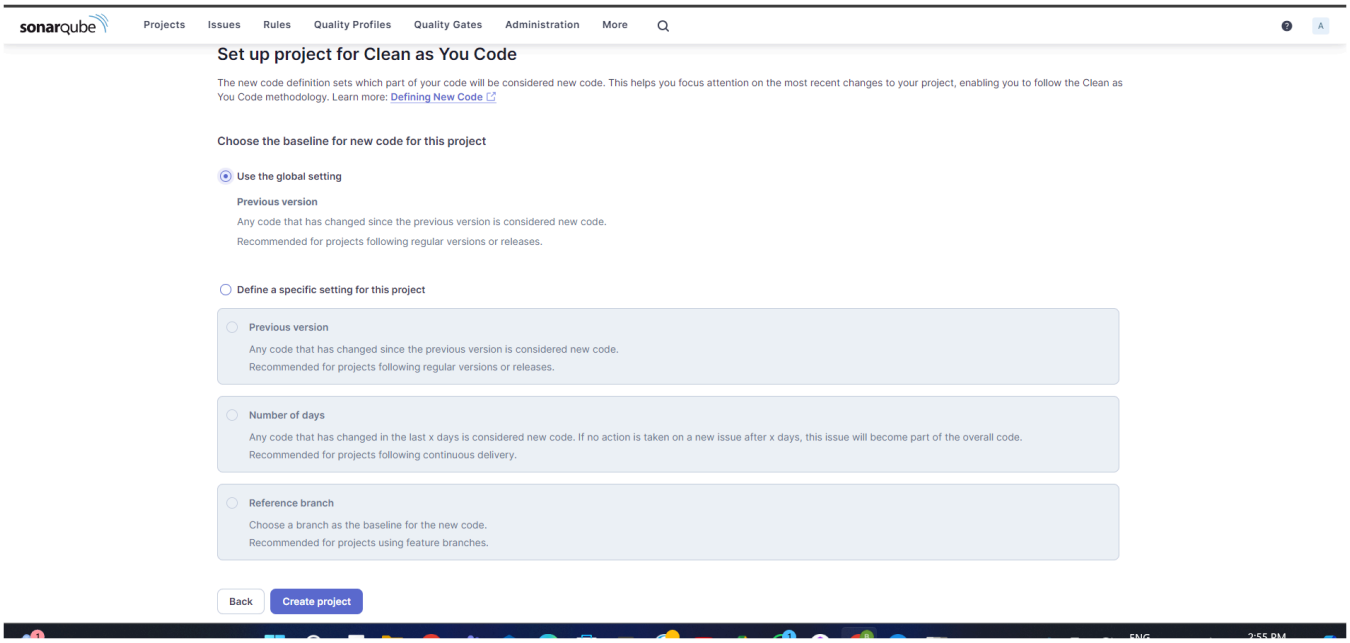
Start 4: Login to SonarQube using username admin and password admin.



Name:Brijesh Sharma **Roll No: 48** **Division: D15C**

Step 5: Create a manual project in SonarQube with any Name

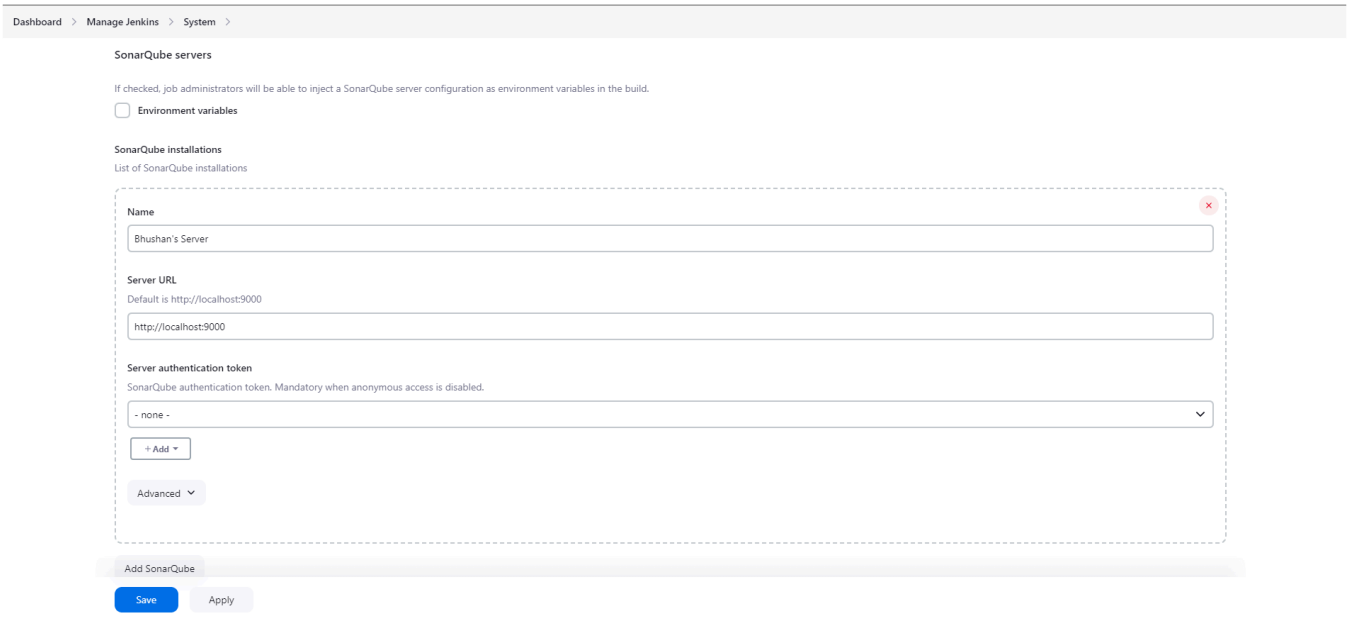




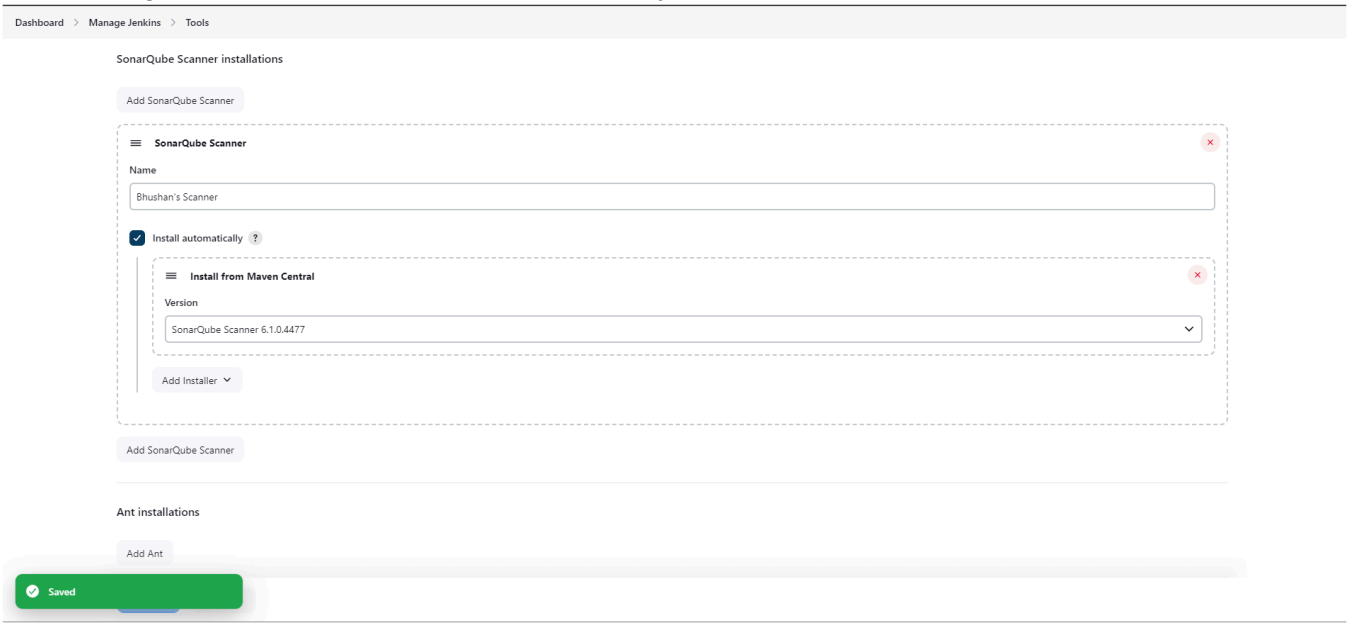
Name:Brijesh Sharma **Roll No: 48** **Division: D15C**

Step 6:Setup the project and come back to Jenkins Dashboard.
Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.

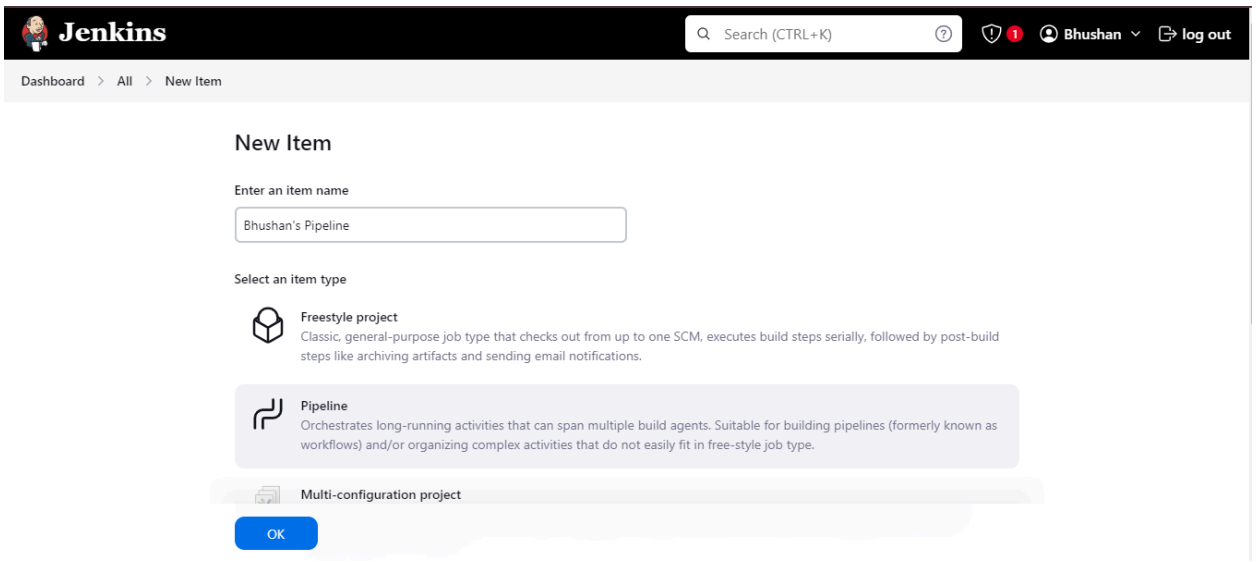
Step 7: Under Jenkins ‘Configure System’, look for SonarQube Servers and enter the details. Enter the Server Authentication token if needed.



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

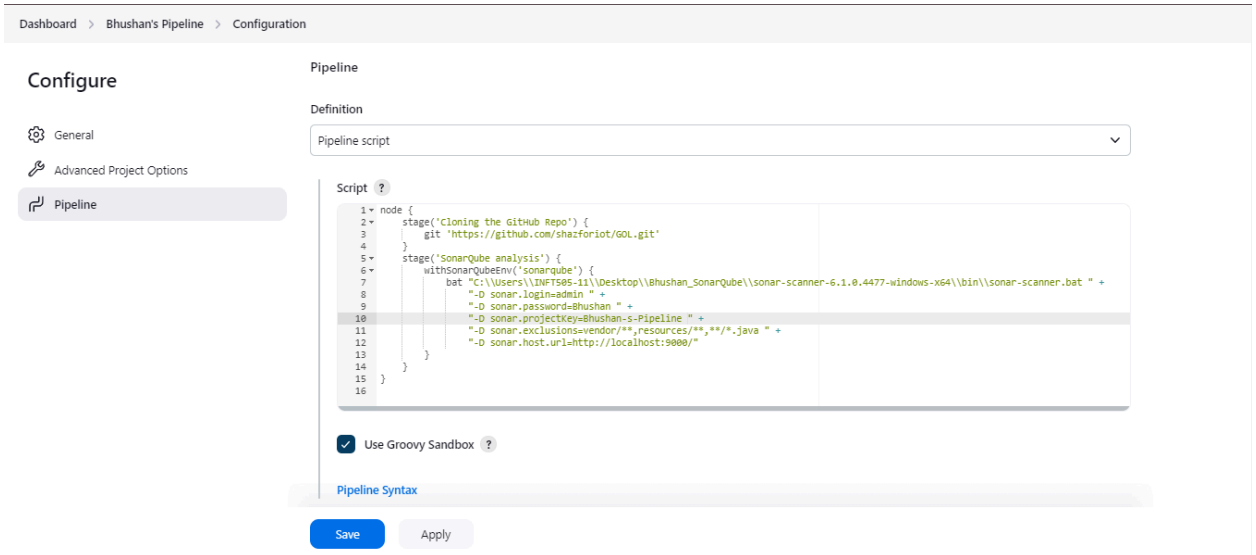


Step 9: Now click on the new item and select the pipeline project and give name.



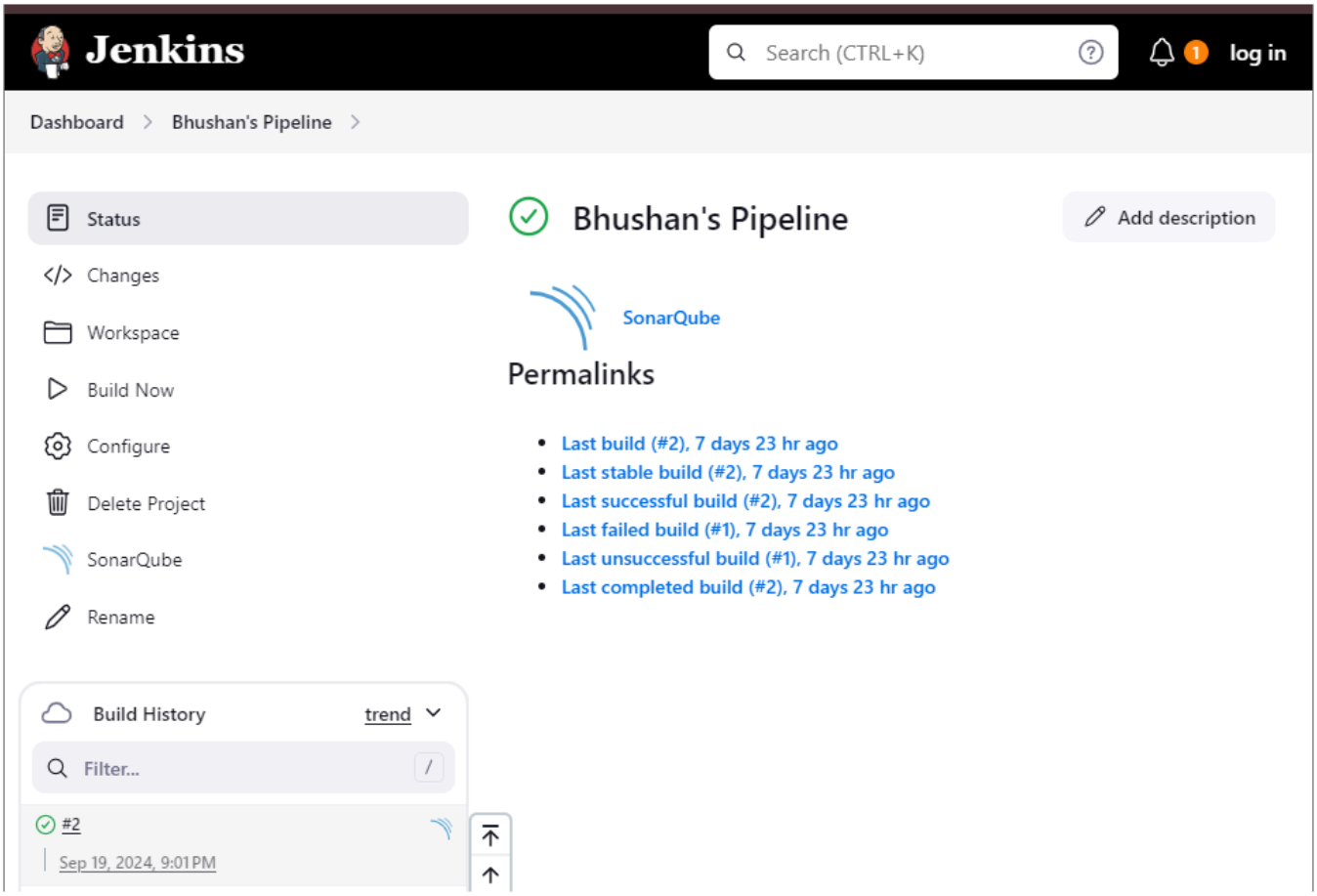
Step 10: Under the scripts add the following script

```
node {
        stage('Cloning the GitHub Repo'){
                git 'https://github.com/shazforiot/GOL.git'
        }
        stage('SonarQube analysis') {
                withSonarQubeEnv('sonarqube') {
                        bat  ''<your bin file location of SonarQube CLI>"+
                                "-D sonar.login=<your user name>"+
                                "-D sonar.password=<your password>"+
                                "-D sonar.projectkey=<your projectkey>"+
                                "-D sonar.exclusions=vendor/**,resources/**,**/*.java "+
                                "-D sonar.host.url=http://localhost:9000/"+
                }
        }

}
```

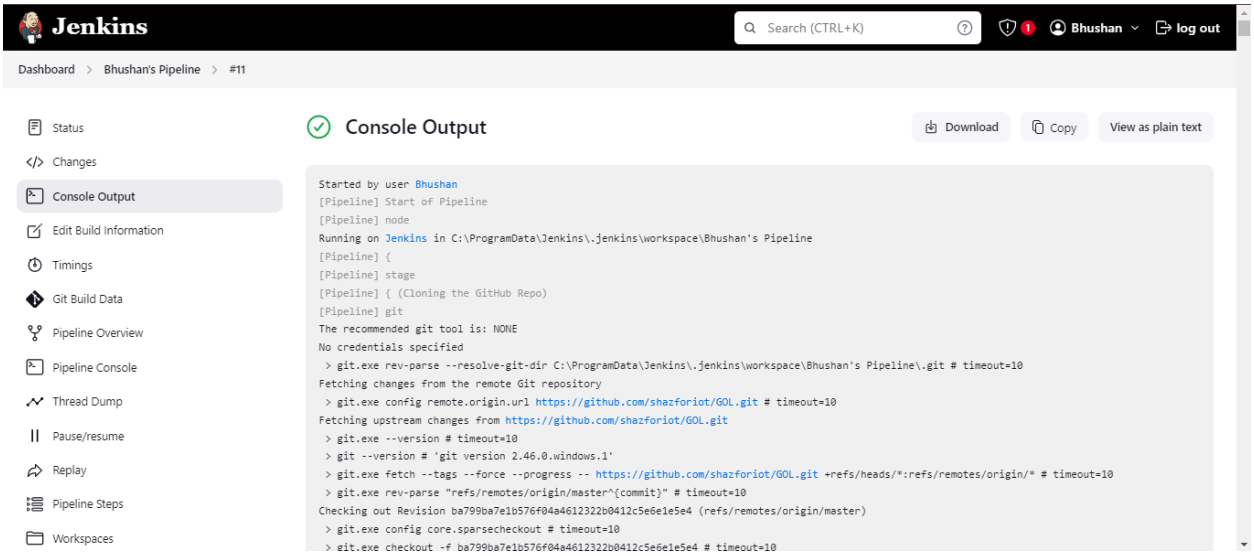


It is a java sample project which has a lot of repetitions and issues that will be detected by SonarQube.

Step 11: Go back to jenkins. Go to the job you had just built and click on Build Now.

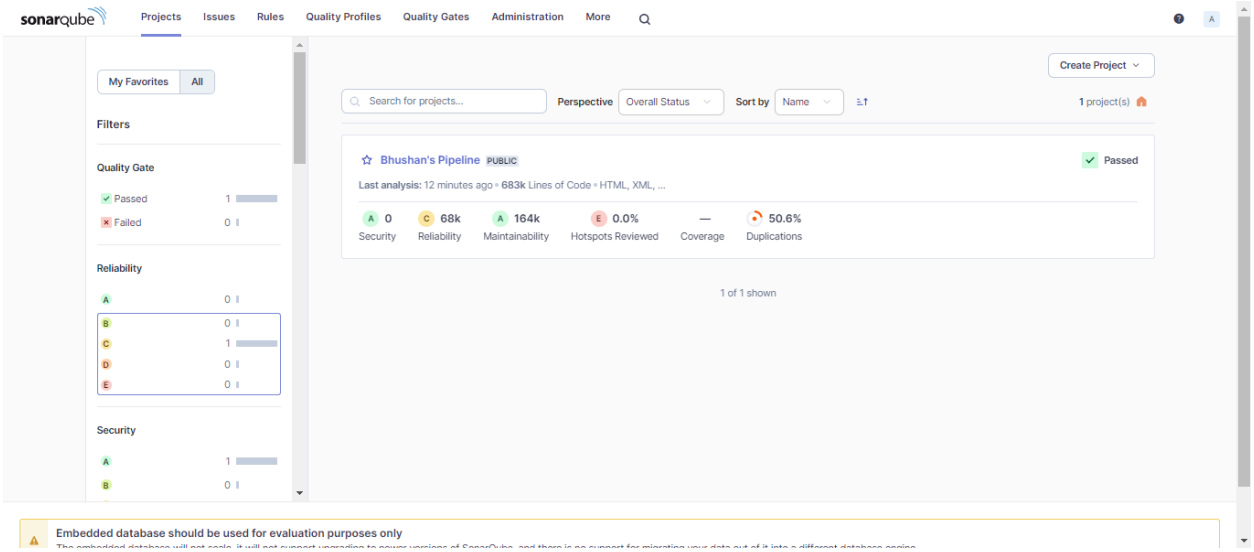


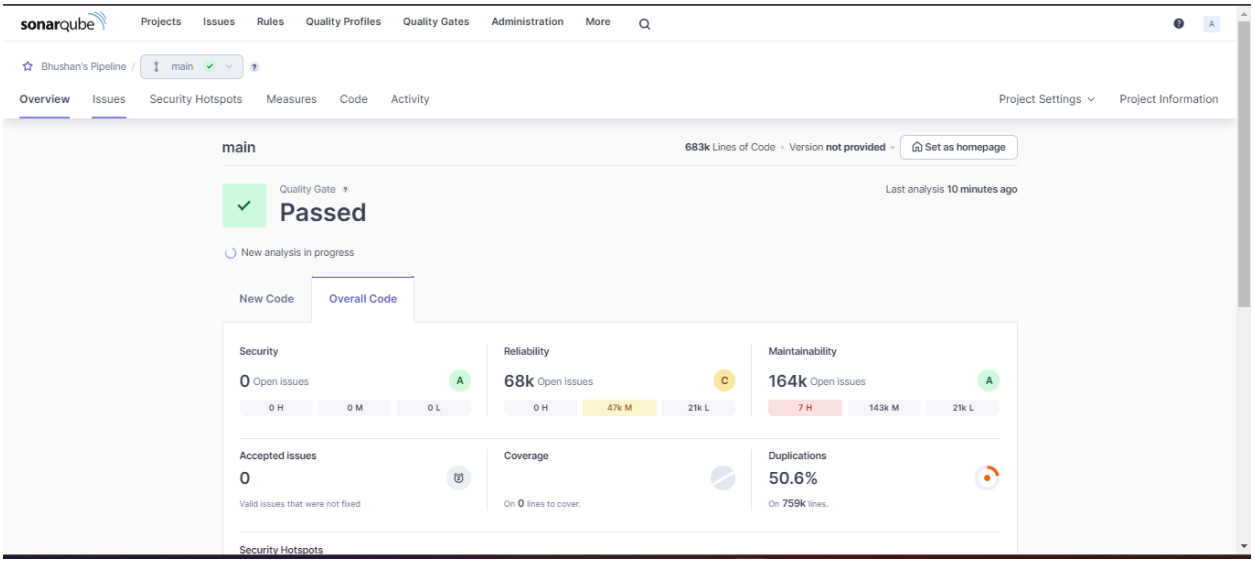
Step 12: Check the console output.



Name:Brijesh Sharma **Roll No: 48** **Division: D15C**

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

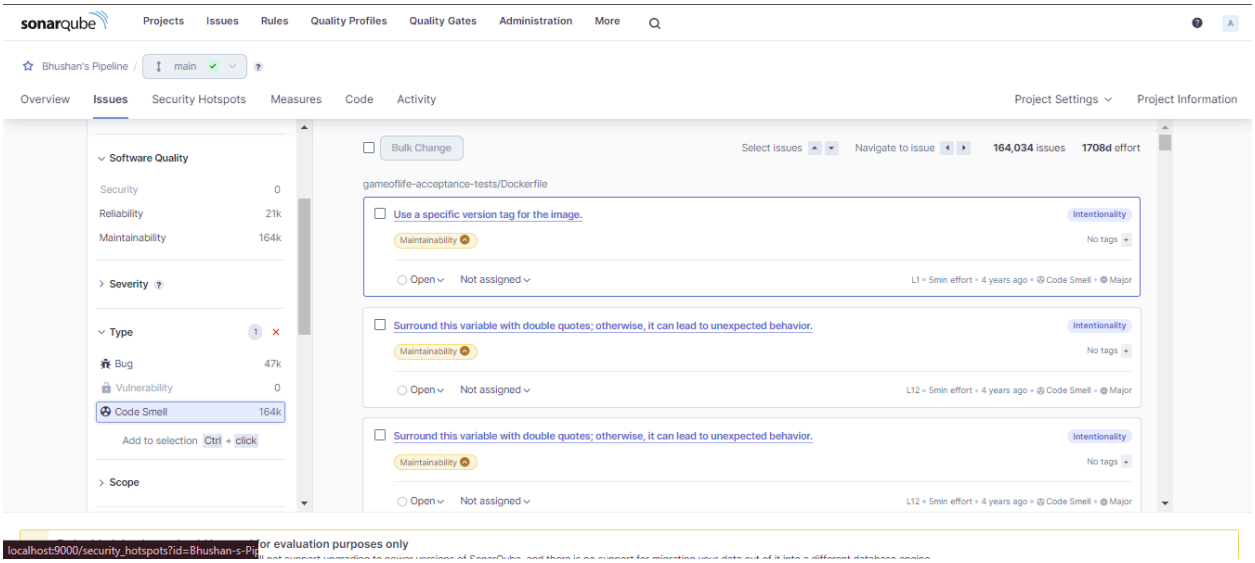




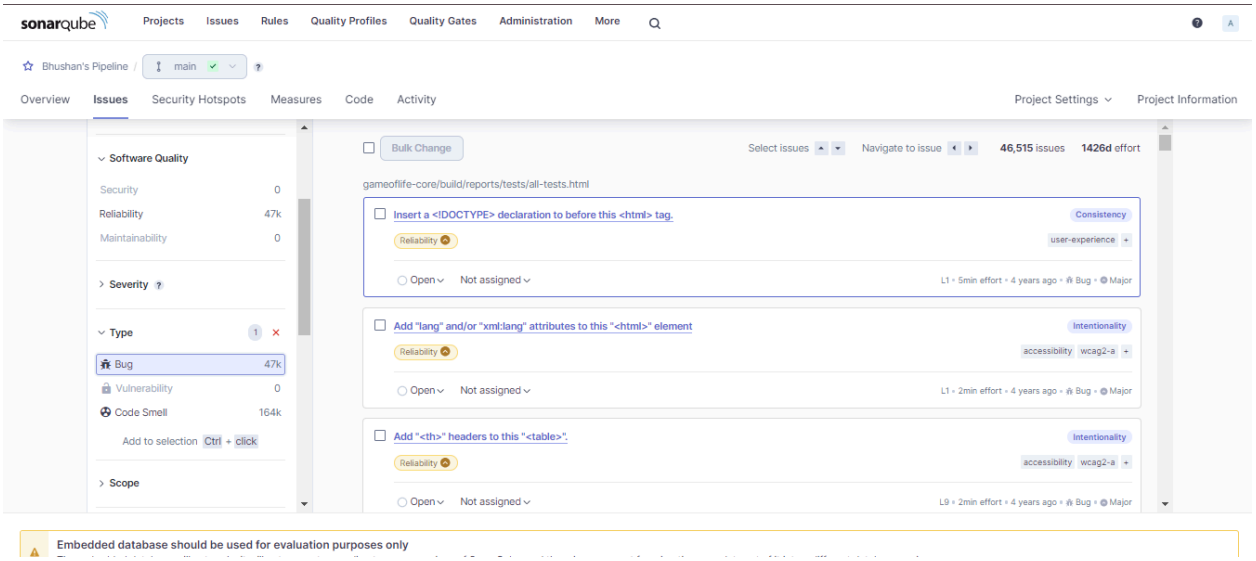
Name:Brijesh Sharma **Roll No: 48** **Division: D15C**

Click on Issues and see the different Issues.

Codesmell:

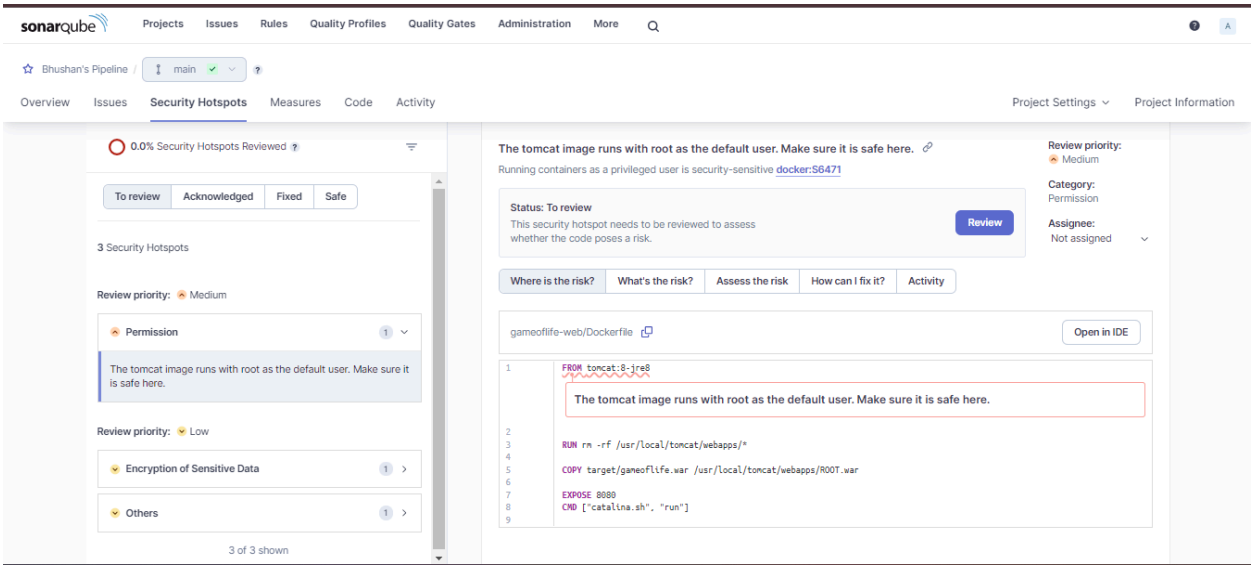


Bugs:

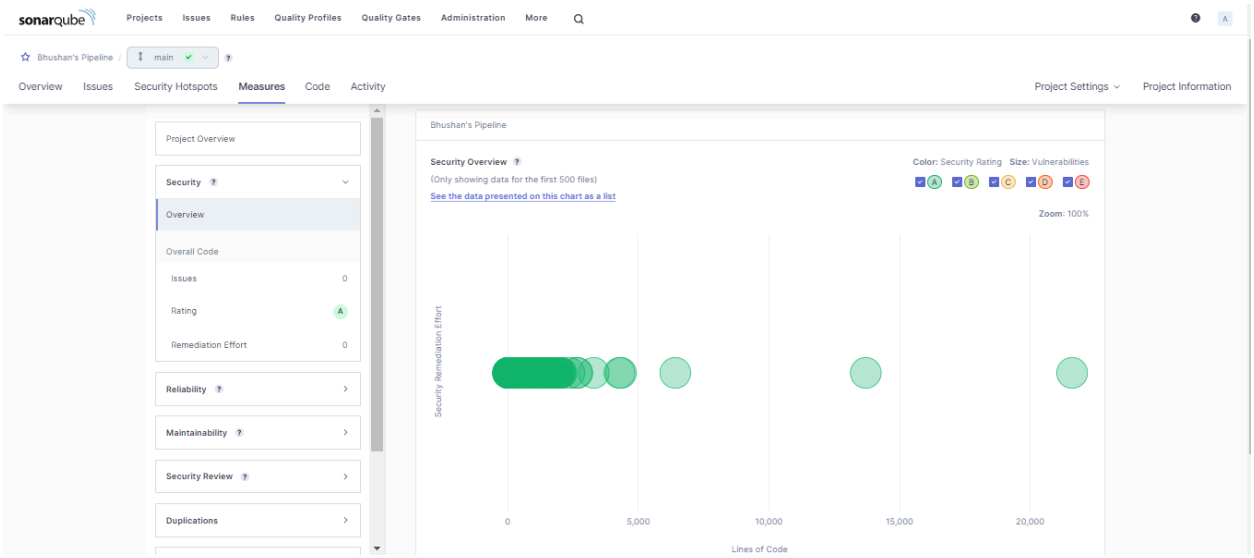


Name:Brijesh Sharma **Roll No: 48** **Division: D15C**

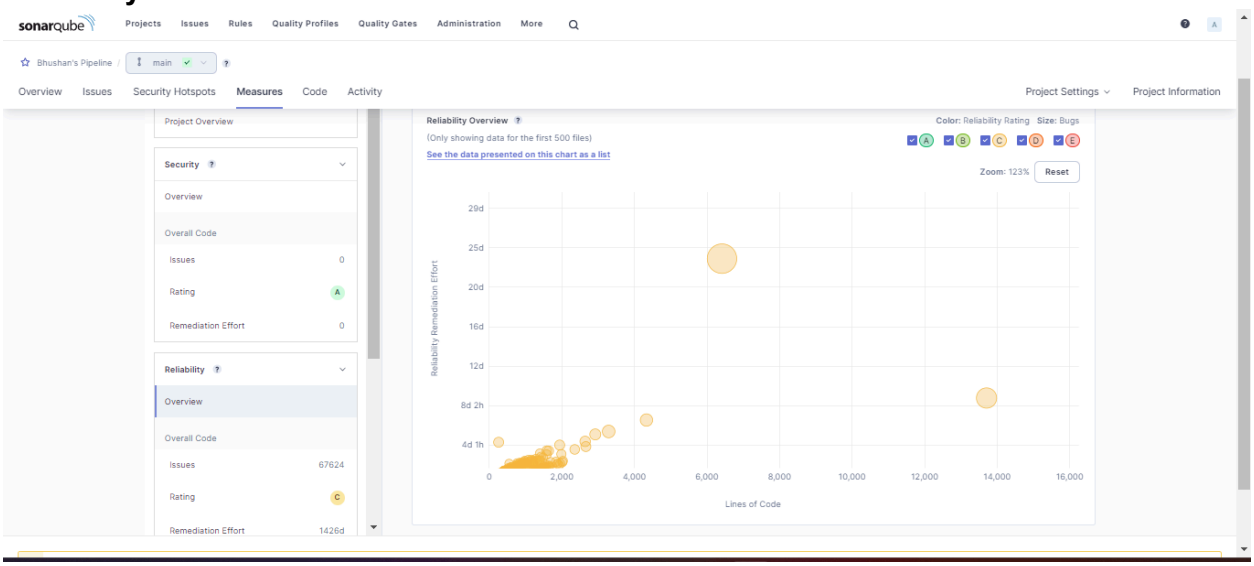
Click on Security hotspots and see the different Security hotspots.



Click on Measures and see the Measures in the form of Graphs.
Security:

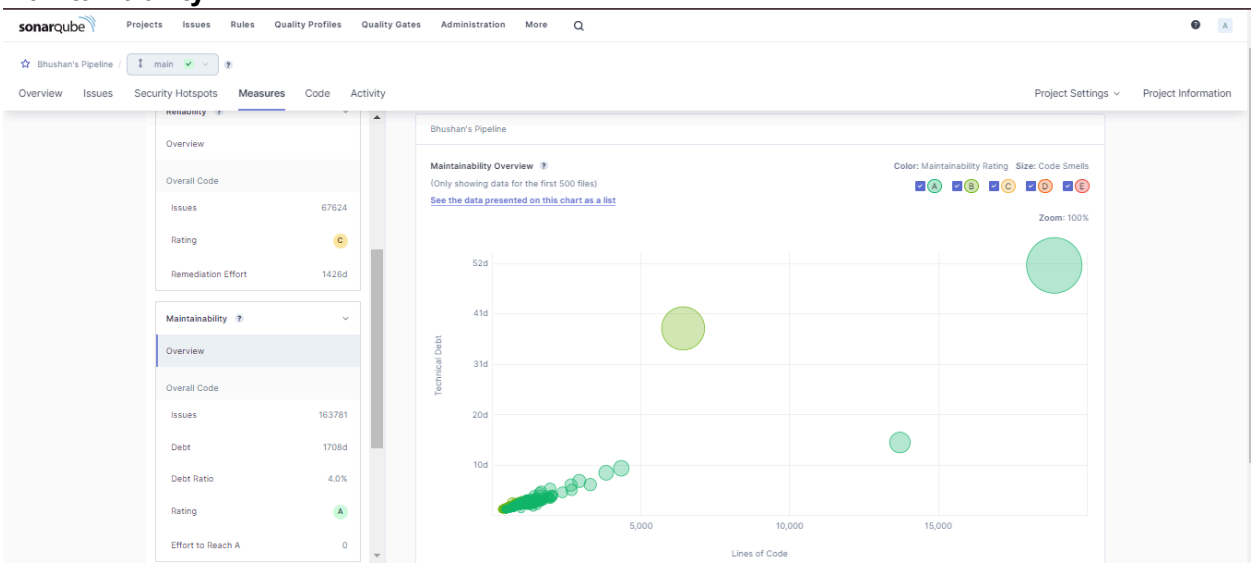


Reliability:

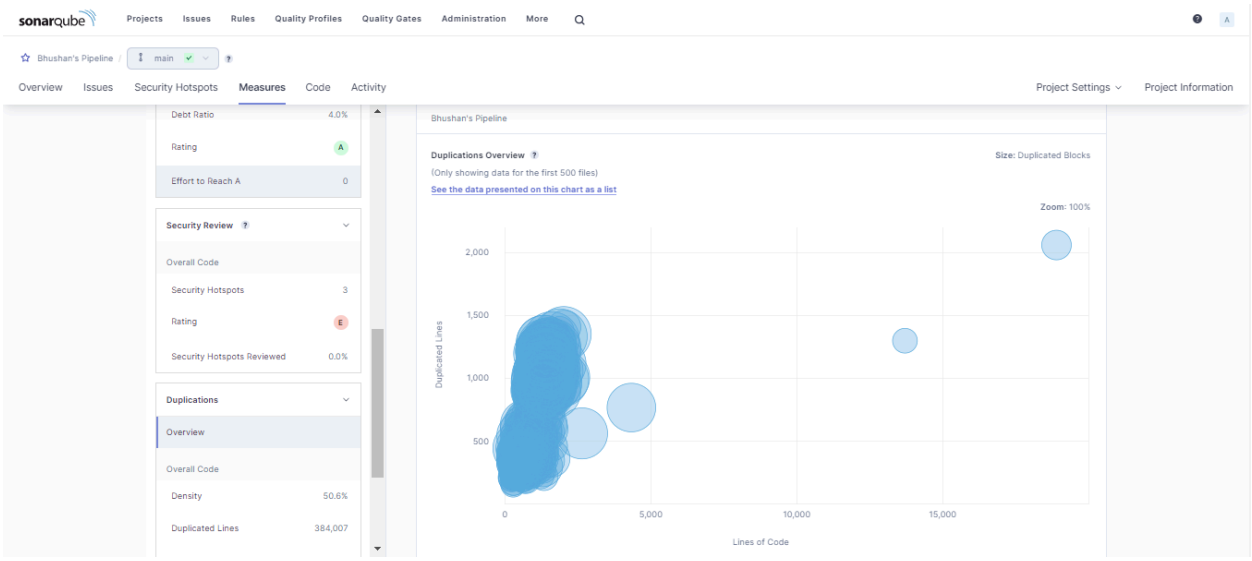


Name:Brijesh Sharma **Roll No: 48** **Division: D15C**

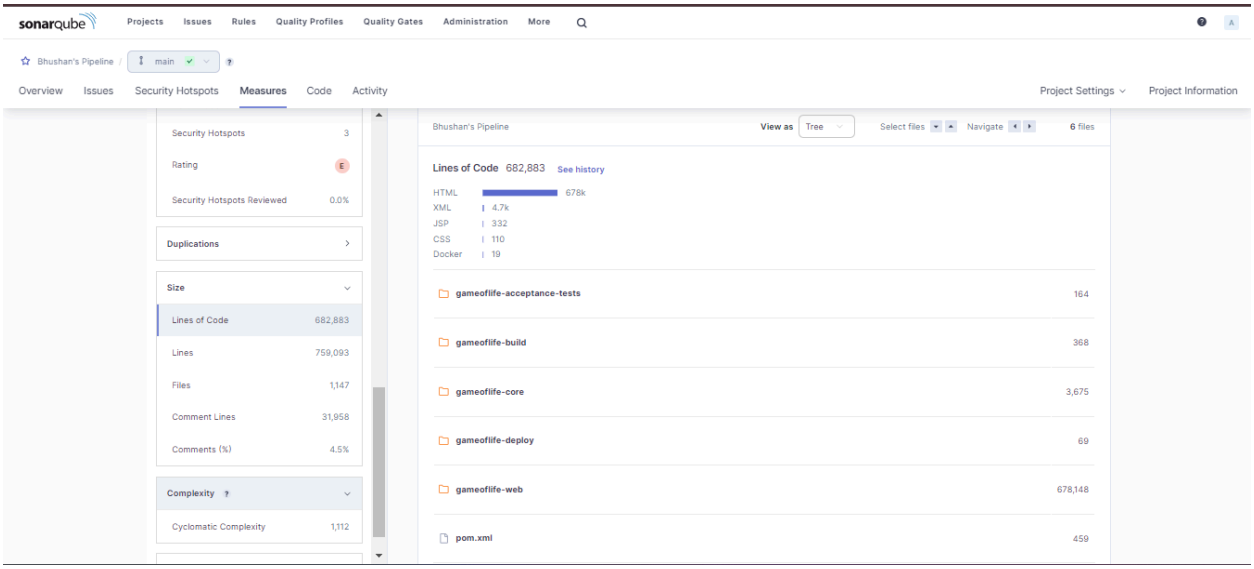
Maintainability:



Duplication:

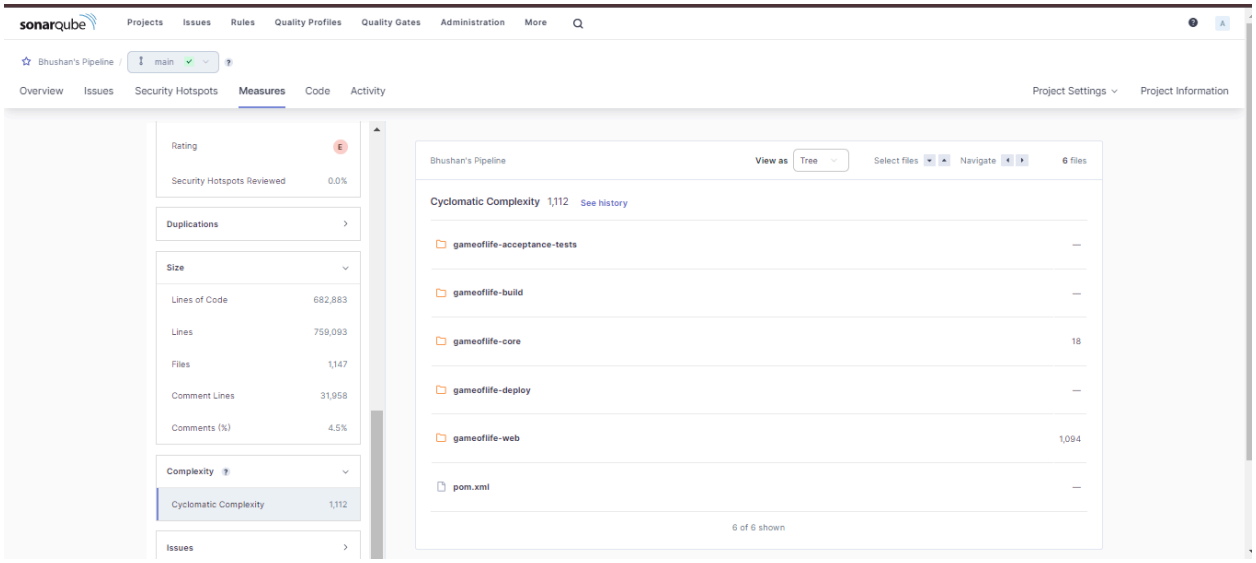


Sizes:

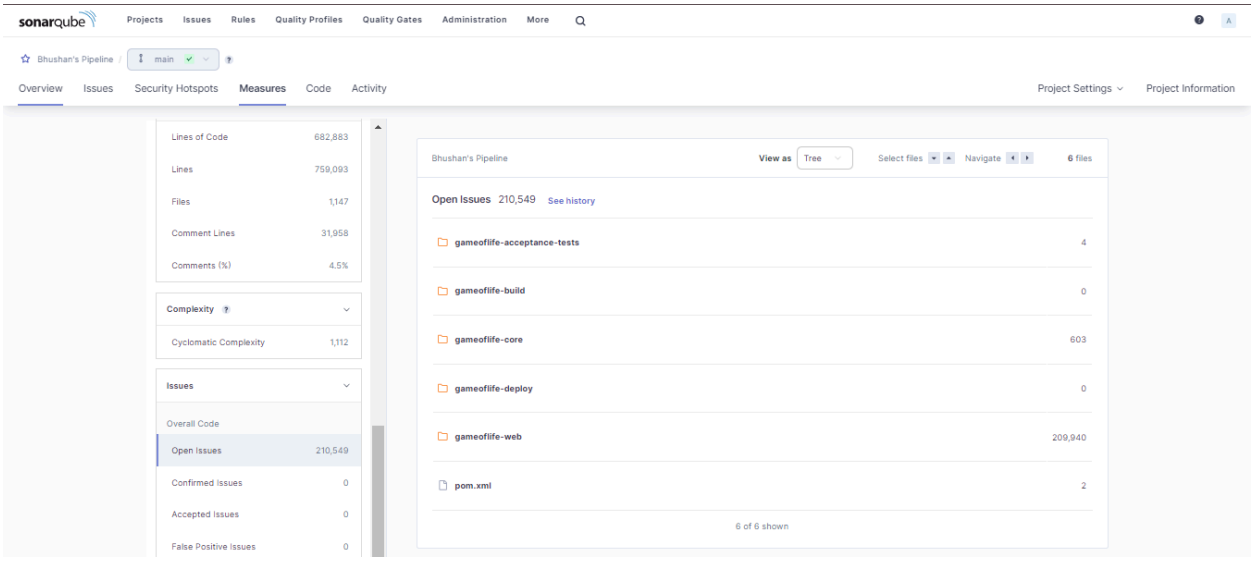


Name:Brijesh Sharma **Roll No: 48** **Division: D15C**

Complexity:



Issues:



Conclusion: In this experiment, we performed static analysis of a code using Jenkins CI/CD Pipeline with SonarQune analysis. A pipeline project is to be created which is given a pipeline script. This script contains all the information needed for the project to run the SonarQube analysis. After the necessary configurations are made on Jenkins, the Jenkins project is built. The code provided in this experiment contains lots of errors, bugs, duplications which can be checked on the SonarQube project linked with this build.It streamlines the process of detecting errors in the code.

Experiment - 9

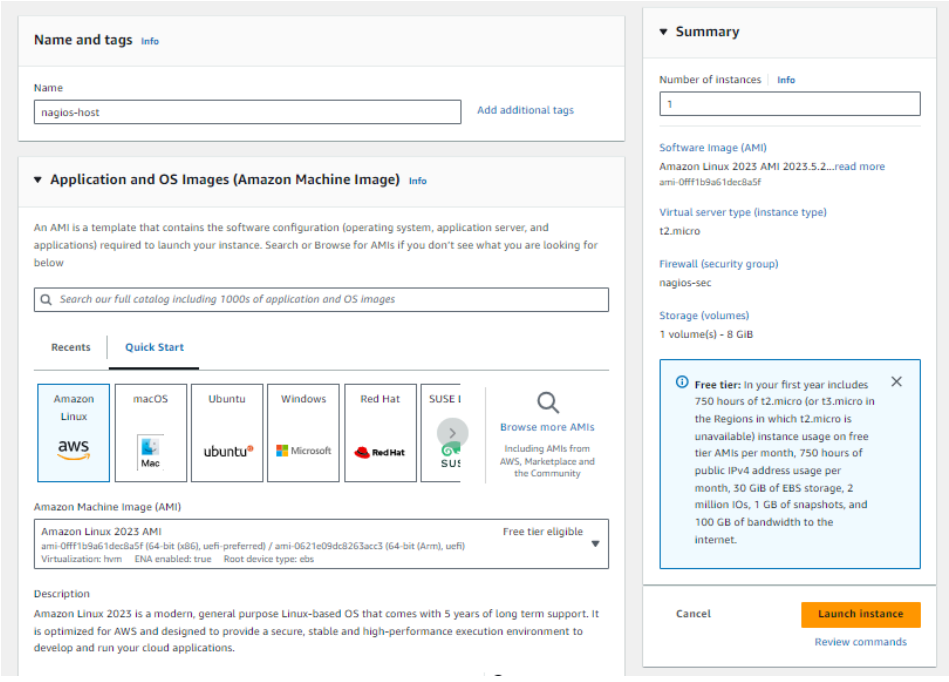
Aim: To Understand Continuous monitoring and Installation and configuration of Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine.

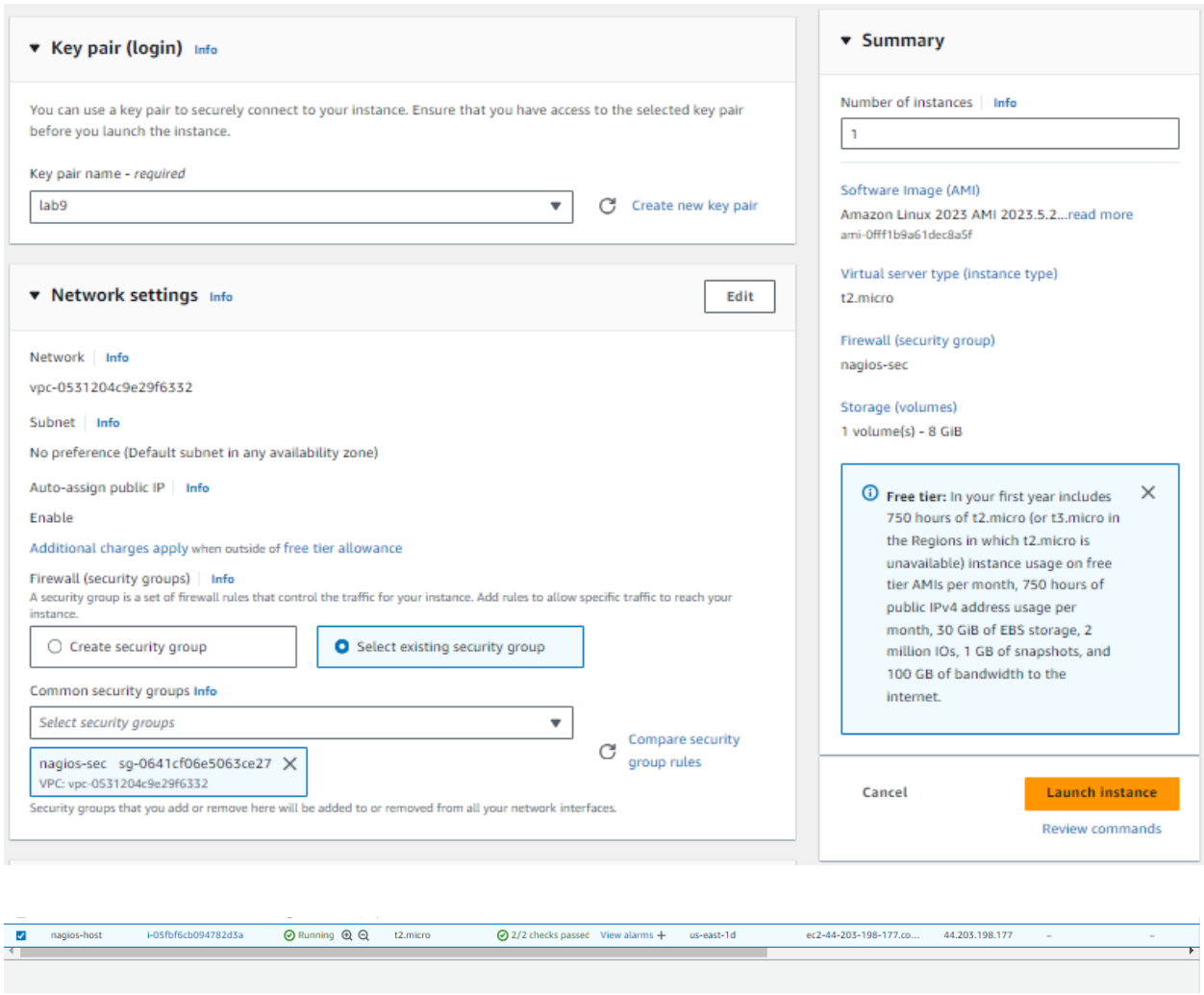
Theory:

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

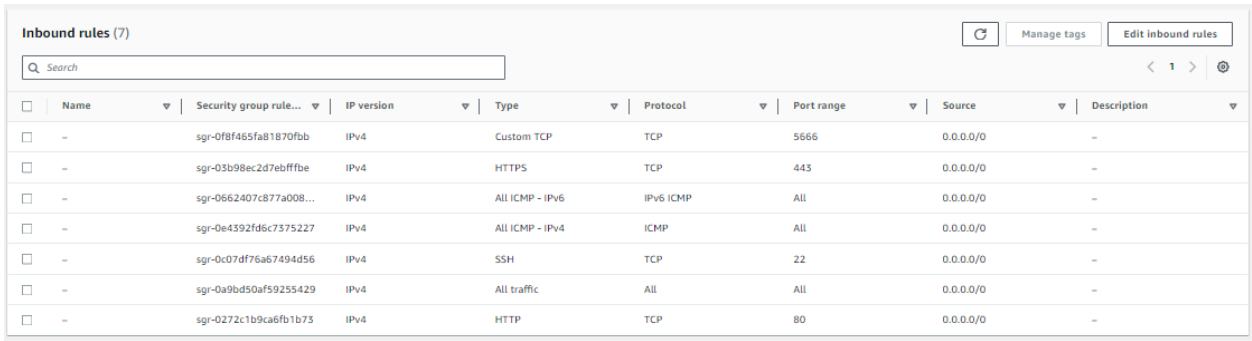
Prerequisites: AWS Free Tier

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



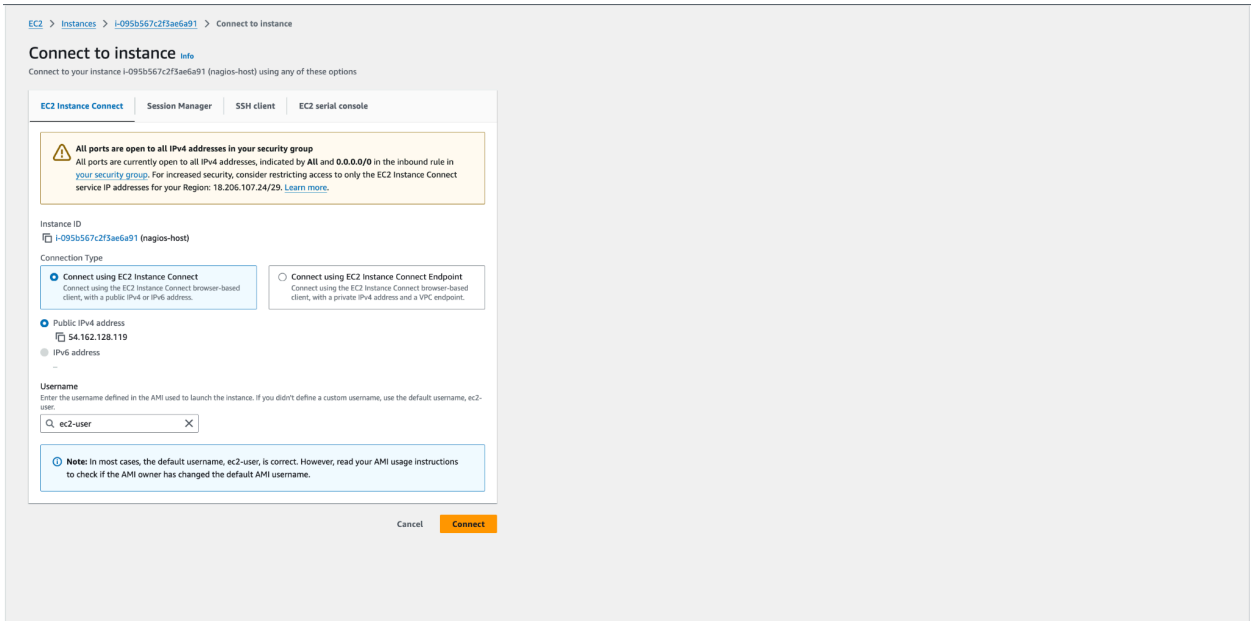


2. Under Security Group, make sure HTTP, HTTPS, SSH, ICMP are open from everywhere.



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

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




```
[ec2-user@ip-172-31-34-108 ~]$ sudo yum update
Last metadata expiration check: 0:04:33 ago on Fri Oct  4 04:35:02 2024.
Dependencies resolved.
Nothing to do.
Complete!
```

sudo yum update

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

sudo yum install httpd php

```
[ec2-user@ip-172-31-91-91 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:19:29 ago on Sun Sep 29 06:56:15 2024.
Dependencies resolved.
================================================================================================================
 Package                    Architecture       Version                                 Repository          Size
================================================================================================================
Installing:
 httpd                      x86_64             2.4.62-1.amzn2023                       amazonlinux         48 k
 php8.3                     x86_64             8.3.10-1.amzn2023.0.1                   amazonlinux         10 k
Installing dependencies:
 apr                        x86_64             1.7.2-2.amzn2023.0.2                    amazonlinux        129 k
 apr-util                   x86_64             1.6.3-1.amzn2023.0.1                    amazonlinux         98 k
 generic-logos-httpd        noarch             18.0.0-12.amzn2023.0.3                  amazonlinux         19 k
 httpd-core                 x86_64             2.4.62-1.amzn2023                       amazonlinux        1.4 M
 httpd-filesystem           noarch             2.4.62-1.amzn2023                       amazonlinux         14 k
 httpd-tools                x86_64             2.4.62-1.amzn2023                       amazonlinux         81 k
 libbrotli                  x86_64             1.0.9-4.amzn2023.0.2                    amazonlinux        315 k
 libsodium                  x86_64             1.0.19-4.amzn2023                       amazonlinux        176 k
 libxslt                    x86_64             1.1.34-5.amzn2023.0.2                   amazonlinux        241 k
 mailcap                    noarch             2.1.49-3.amzn2023.0.3                   amazonlinux         33 k
 nginx-filesystem           noarch             1:1.24.0-1.amzn2023.0.4                 amazonlinux        9.8 k
 php8.3-cli                 x86_64             8.3.10-1.amzn2023.0.1                   amazonlinux        3.7 M
 php8.3-common              x86_64             8.3.10-1.amzn2023.0.1                   amazonlinux        737 k
 php8.3-process             x86_64             8.3.10-1.amzn2023.0.1                   amazonlinux         45 k
 php8.3-xml                 x86_64             8.3.10-1.amzn2023.0.1                   amazonlinux        154 k
Installing weak dependencies:
 apr-util-openssl           x86_64             1.6.3-1.amzn2023.0.1                    amazonlinux         17 k
 mod_http2                  x86_64             2.0.27-1.amzn2023.0.3                   amazonlinux        166 k
 mod_lua                    x86_64             2.4.62-1.amzn2023                       amazonlinux         61 k
 php8.3-fpm                 x86_64             8.3.10-1.amzn2023.0.1                   amazonlinux        1.9 M
 php8.3-mbstring            x86_64             8.3.10-1.amzn2023.0.1                   amazonlinux        528 k
 php8.3-opcache             x86_64             8.3.10-1.amzn2023.0.1                   amazonlinux        379 k
 php8.3-pdo                 x86_64             8.3.10-1.amzn2023.0.1                   amazonlinux         89 k
 php8.3-sodium              x86_64             8.3.10-1.amzn2023.0.1                   amazonlinux         41 k

Transaction Summary
```

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

sudo yum install gcc glibc glibc-common

```
[ec2-user@ip-172-31-34-108 ~]$ sudo yum install gcc glibc glibc-common
Last metadata expiration check: 0:06:01 ago on Fri Oct  4 04:35:02 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             Arch    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
```

```
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-172-31-34-108 ~]$ sudo yum install gd gd-devel
Last metadata expiration check: 0:06:19 ago on Fri Oct  4 04:35:02 2024.
Dependencies resolved.
================================================================================
 Package                  Arch   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
 langpacks-core-font-en   noarch 3.0-21.amzn2023.0.4        amazonlinux  10 k
 libICE                   x86_64 1.0.10-6.amzn2023.0.2      amazonlinux  71 k
 libSM                    x86_64 1.2.3-8.amzn2023.0.2       amazonlinux  42 k
 libX11                   x86_64 1.7.2-3.amzn2023.0.4       amazonlinux 657 k
```

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

Complete!
```

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

sudo adduser -m nagios
sudo passwd nagios
(password : *hello123)*

```
[ec2-user@ip-172-31-34-108 ~]$ sudo adduser -m nagios
sudo passwd nagios
Changing password for user nagios.
New password:
[ec2-user@ip-172-31-34-108 ~]$ sudo passwd nagios
Changing password for user nagios.
New password:
BAD PASSWORD: The password fails the dictionary check - it is based on a dic
tionary word
Retype new password:
passwd: all authentication tokens updated successfully.
```

6. Create a new user group
sudo groupadd nagcmd

```
[ec2-user@ip-172-31-34-108 ~]$ sudo groupadd nagcmd
```

7. Use these commands so that you don't have to use sudo for Apache and Nagios
sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache

```
[ec2-user@ip-172-31-34-108 ~]$ sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache
```

8. Create a new directory for Nagios downloads
mkdir ~/downloads
cd ~/downloads

```
[ec2-user@ip-172-31-34-108 ~]$ mkdir ~/downloads
cd ~/downloads
```

9. Use wget to download the source zip files.
wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz

```
[ec2-user@ip-172-31-34-108 downloads]$ wget https://assets.nagios.com/downlo
ads/nagioscore/releases/nagios-4.5.5.tar.gz
--2024-10-04 04:42:39--  https://assets.nagios.com/downloads/nagioscore/rele
ases/nagios-4.5.5.tar.gz
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2600:3c00::
f03c:92ff:fef7:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... con
nected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: 'nagios-4.5.5.tar.gz'

nagios-4.5.5.tar.g 100%[===============>]   1.97M  5.31MB/s    in 0.4s

2024-10-04 04:42:39 (5.31 MB/s) - 'nagios-4.5.5.tar.gz' saved [2065473/20654
73]
```

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

```
[ec2-user@ip-172-31-34-108 downloads]$ wget https://nagios-plugins.org/downl
oad/nagios-plugins-2.4.11.tar.gz
--2024-10-04 04:42:47--  https://nagios-plugins.org/download/nagios-plugins-
2.4.11.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 2753049 (2.6M) [application/x-gzip]
Saving to: ‘nagios-plugins-2.4.11.tar.gz’

nagios-plugins-2.4 100%[=================>]   2.62M  5.79MB/s    in 0.5s

2024-10-04 04:42:48 (5.79 MB/s) - ‘nagios-plugins-2.4.11.tar.gz’ saved [2753
049/2753049]

[ec2-user@ip-172-31-34-108 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
```

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

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

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

Here we get an error

```
[ec2-user@ip-172-31-34-108 downloads]$ ./configure --with-command-group=nagc
md
-bash: ./configure: No such file or directory
```

Solution

Navigate to nagios folder in downloads
cd nagios-4.5.5

```
[ec2-user@ip-172-31-34-108 downloads]$ cd nagios-4.5.5
```

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

Steps:
sudo yum install openssl-devel

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

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

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

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

```
[ec2-user@ip-172-31-34-108 nagios-4.5.5]$ ./configure --with-command-group=n
agcmd
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
```

12. Compile the source code.
make all

```
[ec2-user@ip-172-31-34-108 nagios-4.5.5]$ make all
cd ./base && make
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o nagios.o ./nagios.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o broker.o broker.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o nebmods.o nebmods.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o ../common/shared.o ../common/shared.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o query-handler.o query-handler.c
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o workers.o workers.c
In function ‘get_wproc_list’,
    inlined from ‘get_worker’ at workers.c:277:12:
workers.c:253:17: warning: ‘%s’ directive argument is null [-Wformat-overflo
w=]
  253 |                 log_debug_info(DEBUGL_CHECKS, 1, "Found specialized
worker(s) for '%s'", (slash && *slash != '/') ? slash : cmd_name);
      |                 ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
gcc -Wall -I.. -I. -I../lib -I../include  -I../include -I..  -g -O2  -DHAVE_
CONFIG_H -DNSCORE -c -o checks.o checks.c
```

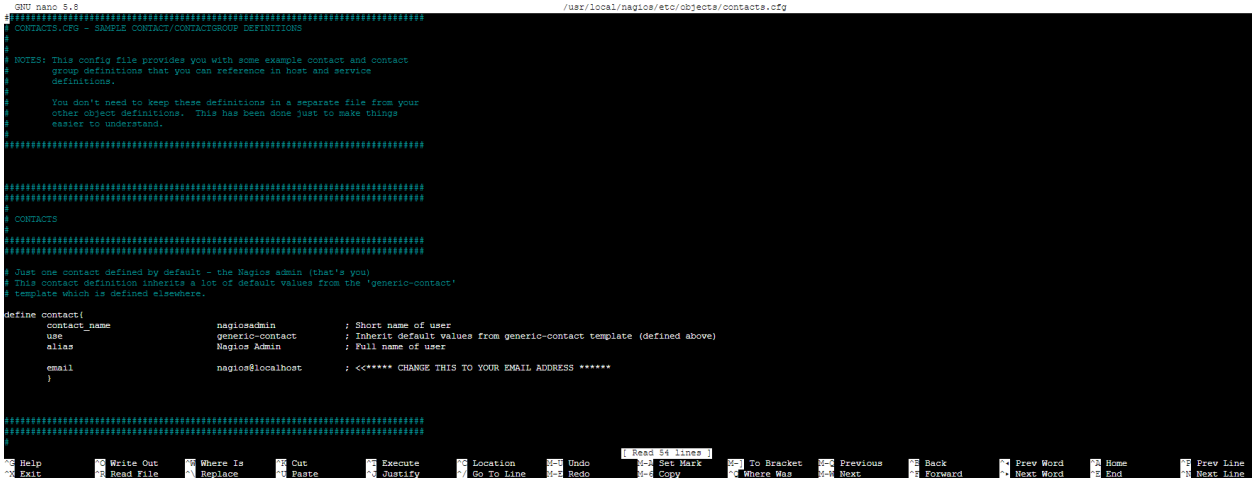
13. Install binaries, init script and sample config files. Lastly, set permissions on the external command directory.
sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode

```
[ec2-user@ip-172-31-34-108 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/b
in
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagiostats /usr/local/nagi
os/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/n
agios/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/me
dia
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/st
ylesheets
```

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

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

```
[ec2-user@ip-172-31-34-108 nagios-4.5.5]$ sudo nano /usr/local/nagios/etc/ob
jects/contacts.cfg
```



In this page, change email address:

```
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.brijesh.sharma@ves.ac.in  ; <<***** CHANGE THIS TO YOUR EMAIL ADDRESS ******
        }
```

15. Configure the web interface.

sudo make install-webconf

```
[ec2-user@ip-172-31-34-108 nagios-4.5.5]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios
.conf
if [ 0 -eq 1 ]; then \
        ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagio
s.conf; \
fi

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

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

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

```
[ec2-user@ip-172-31-34-108 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
```

Password: *hello123*

17. Restart Apache

sudo service httpd restart

```
[ec2-user@ip-172-31-34-108 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
```

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

cd ~/downloads

tar zxvf nagios-plugins-2.4.11.tar.gz

```
[ec2-user@ip-172-31-34-108 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
```

19. Compile and install plugins

cd nagios-plugins-2.4.11

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

```
[ec2-user@ip-172-31-34-108 downloads]$ cd nagios-plugins-2.4.11
./configure --with-nagios-user=nagios --with-nagios-group=nagios
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether to enable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether gcc understands -c and -o together... yes
checking whether make supports the include directive... yes (GNU style)
```

make
sudo make install

```
[ec2-user@ip-172-31-34-108 nagios-plugins-2.4.11]$ make
sudo make install
make  all-recursive
make[1]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
Making all in gl
make[2]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/
gl'
rm -f alloca.h-t alloca.h && \
{ echo '/* DO NOT EDIT! GENERATED AUTOMATICALLY! */'; \
  cat ./alloca.in.h; \
} > alloca.h-t && \
mv -f alloca.h-t alloca.h
rm -f c++defs.h-t c++defs.h && \
sed -n -e '/_GL_CXXDEFS/,$p' \
  < ../build-aux/snippet/c++defs.h \
  > c++defs.h-t && \
mv c++defs.h-t c++defs.h
rm -f warn-on-use.h-t warn-on-use.h && \
sed -n -e '/^.ifndef/,$p' \
```

```
fi
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/p
o'
make[1]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
make[2]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
```

20. Start Nagios

Add Nagios to the list of system services

sudo chkconfig --add nagios

sudo chkconfig nagios on

```
[ec2-user@ip-172-31-34-108 nagios-plugins-2.4.11]$ sudo chkconfig --add nagi
os
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.
```

Verify the sample configuration files

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

```
[ec2-user@ip-172-31-34-108 nagios-plugins-2.4.11]$ sudo /usr/local/nagios/bin/nagios -v /us
r/local/nagios/etc/nagios.cfg

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

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

Running pre-flight check on configuration data...

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

Total Warnings: 0
Total Errors:   0
```

sudo service nagios start

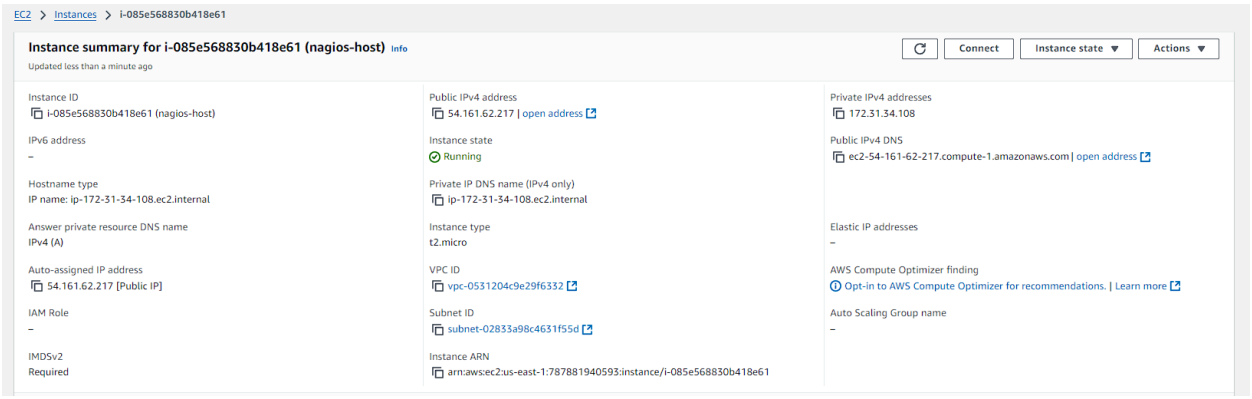
```
[ec2-user@ip-172-31-34-108 nagios-plugins-2.4.11]$ sudo service nagios startRedirecting to
/bin/systemctl start nagios.service
```

21. Check the status of Nagios

sudo systemctl status nagios

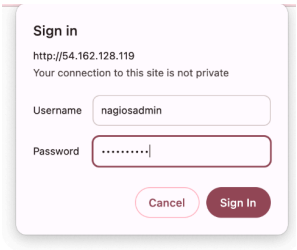
```
[ec2-user@ip-172-31-34-108 nagios-plugins-2.4.11]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; prese>     Active: active (runni
ng) since Fri 2024-10-04 04:48:01 UTC; 6s ago
       Docs: https://www.nagios.org/documentation
    Process: 63465 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/>    Process: 63466 ExecSta
rt=/usr/local/nagios/bin/nagios -d /usr/local/nag>   Main PID: 63467 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 5.6M
        CPU: 79ms
     CGroup: /system.slice/nagios.service
             ├─63467 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/>             ├─63468 /usr/
local/nagios/bin/nagios --worker /usr/local/nagio>             ├─63469 /usr/local/nagios/bin/nagios --
worker /usr/local/nagio>             ├─63470 /usr/local/nagios/bin/nagios --worker /usr/local/nagio>
           ├─63471 /usr/local/nagios/bin/nagios --worker /usr/local/nagio>             └─63472 /usr/lo
cal/nagios/bin/nagios -d /usr/local/nagios/etc/>
Oct 04 04:48:01 ip-172-31-34-108.ec2.internal nagios[63467]: qh: Socket '/u>Oct 04 04:48:01 ip-172-31-
34-108.ec2.internal nagios[63467]: qh: core query>Oct 04 04:48:01 ip-172-31-34-108.ec2.internal nagios
[63467]: qh: echo servi>Oct 04 04:48:01 ip-172-31-34-108.ec2.internal nagios[63467]: qh: help for t>Oc
t 04 04:48:01 ip-172-31-34-108.ec2.internal nagios[63467]: wproc: Success>Oct 04 04:48:01 ip-172-31-34
-108.ec2.internal nagios[63467]: wproc: Registr>Oct 04 04:48:01 ip-172-31-34-108.ec2.internal nagios[6
3467]: wproc: Registr>Oct 04 04:48:01 ip-172-31-34-108.ec2.internal nagios[63467]: wproc: Registr>Oct
04 04:48:01 ip-172-31-34-108.ec2.internal nagios[63467]: wproc: Registr>Oct 04 04:48:01 ip-172-31-34-1
08.ec2.internal nagios[63467]: Successfully l>lines 1-28/28 (END)
[ec2-user@ip-172-31-34-108 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; prese>     Active: active (runni
ng) since Fri 2024-10-04 04:48:01 UTC; 30s ago
       Docs: https://www.nagios.org/documentation
    Process: 63465 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/>    Process: 63466 ExecSta
rt=/usr/local/nagios/bin/nagios -d /usr/local/nag>   Main PID: 63467 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 5.6M
        CPU: 82ms
     CGroup: /system.slice/nagios.service
             ├─63467 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/>             ├─63468 /usr/
local/nagios/bin/nagios --worker /usr/local/nagio>             ├─63469 /usr/local/nagios/bin/nagios --
worker /usr/local/nagio>             ├─63470 /usr/local/nagios/bin/nagios --worker /usr/local/nagio>
           ├─63471 /usr/local/nagios/bin/nagios --worker /usr/local/nagio>             └─63472 /usr/lo
cal/nagios/bin/nagios -d /usr/local/nagios/etc/>
Oct 04 04:48:01 ip-172-31-34-108.ec2.internal nagios[63467]: qh: Socket '/u>Oct 04 04:48:01 ip-172-31-
34-108.ec2.internal nagios[63467]: qh: core query>Oct 04 04:48:01 ip-172-31-34-108.ec2.internal nagios
[63467]: qh: echo servi>Oct 04 04:48:01 ip-172-31-34-108.ec2.internal nagios[63467]: qh: help for t>Oc
t 04 04:48:01 ip-172-31-34-108.ec2.internal nagios[63467]: wproc: Success>Oct 04 04:48:01 ip-172-31-34
-108.ec2.internal nagios[63467]: wproc: Registr>Oct 04 04:48:01 ip-172-31-34-108.ec2.internal nagios[6
3467]: wproc: Registr>Oct 04 04:48:01 ip-172-31-34-108.ec2.internal nagios[63467]: wproc: Registr>Oct
04 04:48:01 ip-172-31-34-108.ec2.internal nagios[63467]: wproc: Registr>Oct 04 04:48:01 ip-172-31-34-1
08.ec2.internal nagios[63467]: Successfully l>lines 1-28/28 (END)client_loop: send disconnect: Connect
ion reset
```

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

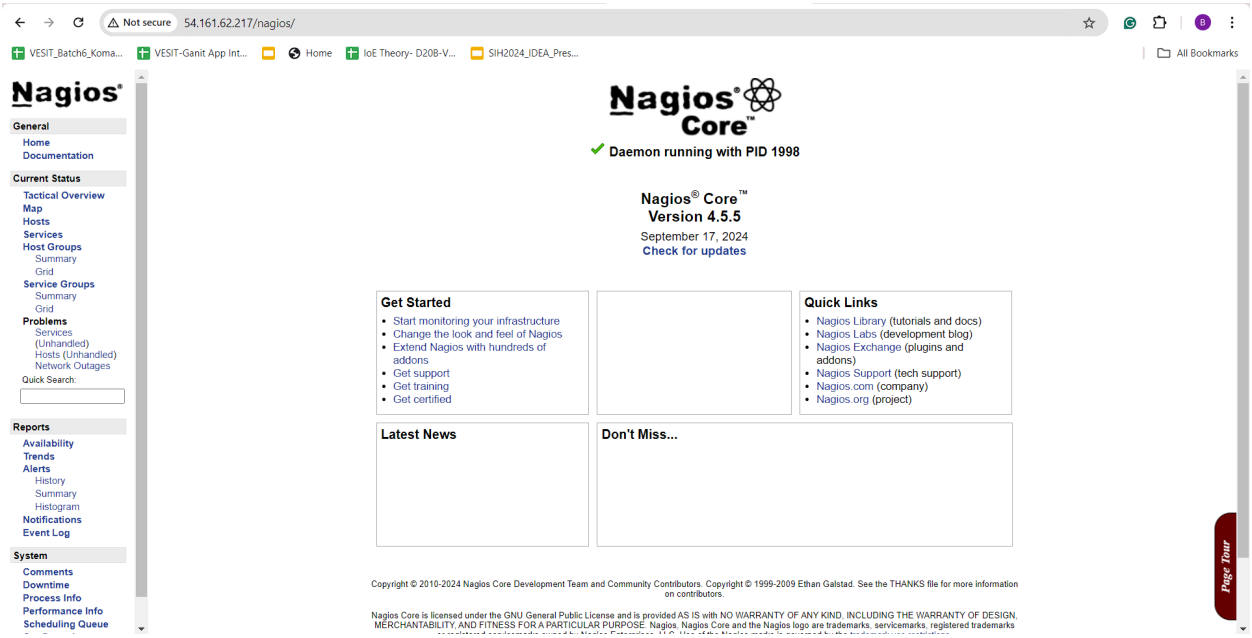


23. Open up your browser and look for http://<your_public_ip_address>/nagios

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



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



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

Conclusion:

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

Brijesh R. Sharma **Division: D15C** **Roll No:48**

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

Theory:

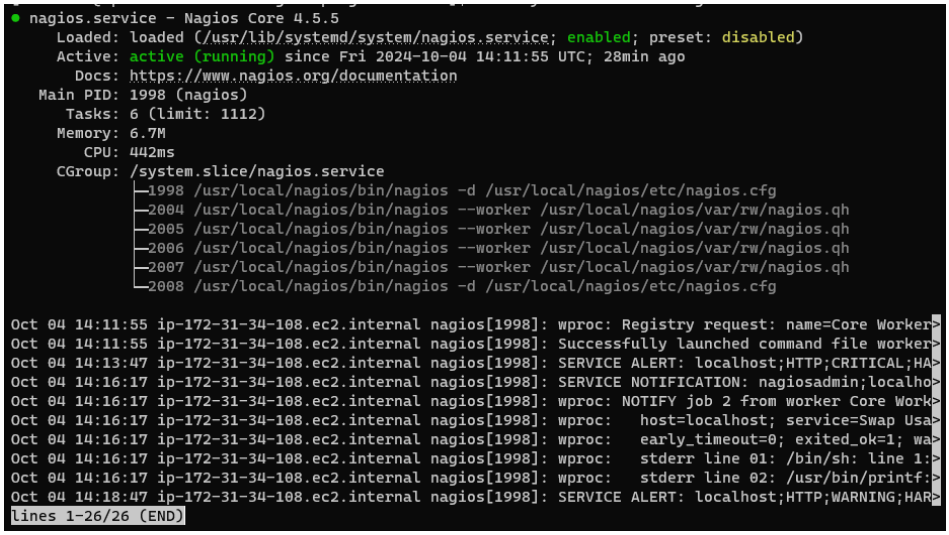
Port and Service Monitoring
Port and service monitoring in Nagios involves checking the availability and responsiveness of network services running on specific ports. This ensures that critical services (like HTTP, FTP, or SSH) are operational. Nagios uses plugins to ping the ports and verify whether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

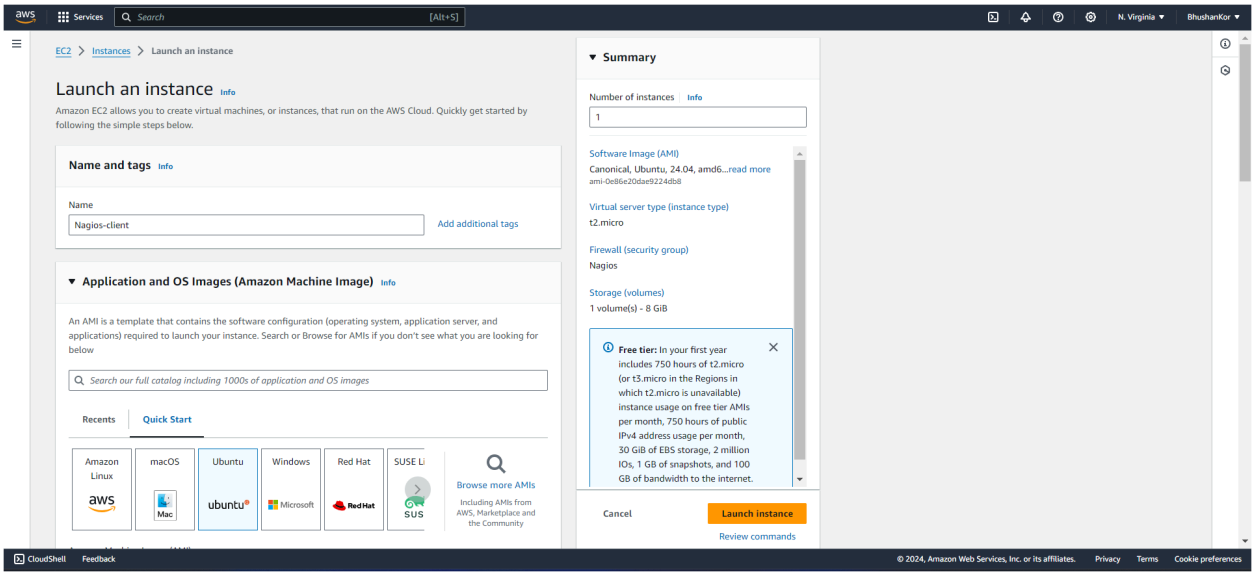
Windows/Linux Server Monitoring
Windows/Linux server monitoring with Nagios entails tracking the performance and health of servers running these operating systems. It includes monitoring metrics such as CPU usage, memory consumption, disk space, and system logs. Nagios employs various plugins to gather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

Prerequisites:
AWS Academy or Personal account.
Nagios Server running on Amazon Linux Machine. (Refer Experiment No 9)

Monitoring Using Nagios:

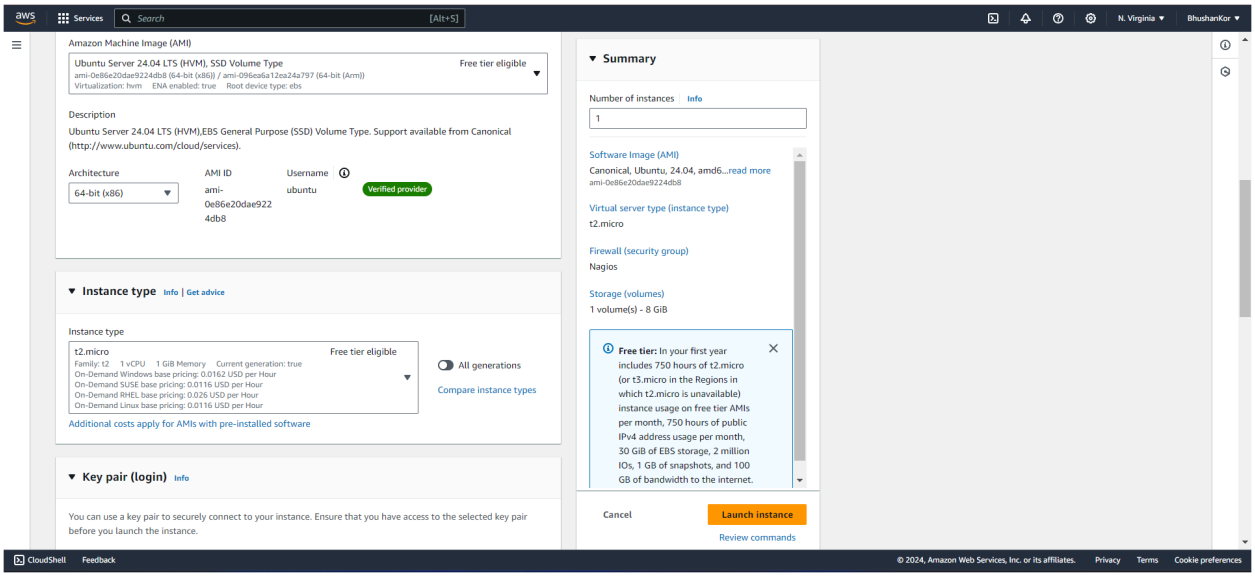
Step 1: To Confirm Nagios is running on the server side Perform the following command on your Amazon Linux Machine (Nagios-host).
sudo systemctl status nagios
You can now proceed if you get the above message/output.
Step 2: Now Create a new EC2 instance. Name: Nagios-client, AMI: Ubuntu Instance Type: t2.micro.



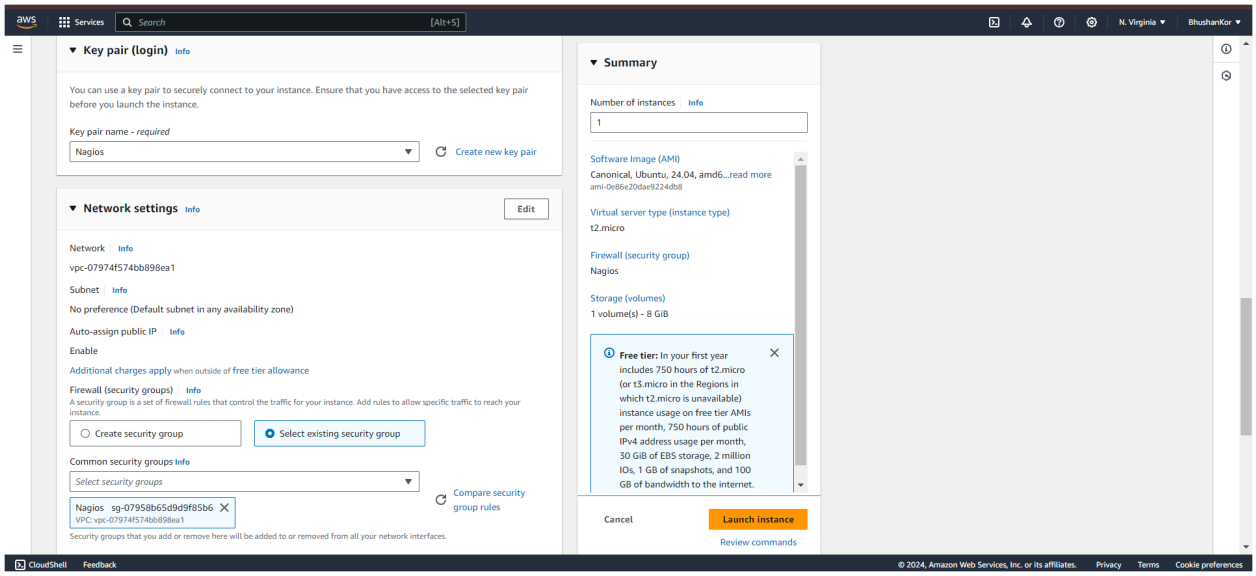


For Key pair : Click on create key and make key of type RSA with extension .pem . Key will be downloaded to your local machine.

Now select that key in key pair if you already have key with type RSA and extension .pem no need to create new key but you must have that key downloaded.



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



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

Successfully connected to the instance.

```
PS C:\Users\hp\Downloads\lab09> ssh -i "lab09.pem" ec2-user@ec2-54-172-217-167.compute-1.amazonaws.com
The authenticity of host 'ec2-54-172-217-167.compute-1.amazonaws.com (54.172.217.167)' can't be established.
ED25519 key fingerprint is SHA256:LBNecYQvwiUAFbRZ4UavnvY9vHuBhVKS/K3DVKJ9Jn4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-172-217-167.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
       #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-38-4 ~]$
```

Now perform all the commands on the Nagios-host till step 10
Step 4: Now on the server Nagios-host run the following command.
ps -ef | grep nagios

```
[ec2-user@ip-172-31-34-108 ~]$ ps -ef | grep nagios
nagios      1998       1  0 14:11 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios      2004    1998  0 14:11 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      2005    1998  0 14:11 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      2006    1998  0 14:11 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      2007    1998  0 14:11 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios      2008    1998  0 14:11 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
root       16967    2370  0 14:20 pts/0    00:00:00 sudo systemctl status nagios
root       16970   16967  0 14:20 pts/1    00:00:00 sudo systemctl status nagios
root       16971   16970  0 14:20 pts/1    00:00:00 systemctl status nagios
root       18221   18132  0 14:40 pts/2    00:00:00 sudo systemctl status nagios
root       18223   18221  0 14:40 pts/3    00:00:00 sudo systemctl status nagios
root       18224   18223  0 14:40 pts/3    00:00:00 systemctl status nagios
ec2-user   19275   19251  0 14:59 pts/4    00:00:00 grep --color=auto nagios
```

Step 5: Now Become root user and create root directories.

sudo su

mkdir /usr/local/nagios/etc/objects/monitorhosts

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

```
[ec2-user@ip-172-31-34-108 ~]$ sudo su
mkdir /usr/local/nagios/etc/objects/monitorhosts
mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-34-108 ec2-user]#
```

```
[root@ip-172-31-34-108 ec2-user]# mkdir /usr/local/nagios/etc/objects/monitorhosts
[root@ip-172-31-34-108 ec2-user]# mkdir /usr/local/nagios/etc/objects/monitorhosts/linu
xhosts
```

Step 6: Copy the sample localhost.cfg to linuxhost.cfg by running the following command.(Below command should come in one line see screenshot below)

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

```
[root@ip-172-31-34-108 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/l
ocal/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
[root@ip-172-31-34-108 ec2-user]#
```

Step 7: Open linuxserver.cfg using nano and make the following changes in all positions?everywhere in file.

Change `hostname` to **linuxserver**.

Change `address` to the public IP of your Linux client.

Set `hostgroup_name` to **linux-servers1**.

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

```
define host {

    use                     linux-server            ; Name of host template to use
                                                    ; This host definition will inheri>
                                                    ; in (or inherited by) the linux-s>
    host_name               linuxserver
    alias                   localhost
    address                 54.172.217.167
}
```

```
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 th>
```

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

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

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

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

```
[root@ip-172-31-34-108 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/loca
l/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg', starting on line 152)
Warning: Duplicate definition found for service 'SSH' on host 'localhost' (config file '/usr/local
/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg', starting on line 138)
Warning: Duplicate definition found for service 'Swap Usage' on host 'localhost' (config file '/us
r/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg', starting on line 125)
Warning: Duplicate definition found for service 'Current Load' on host 'localhost' (config file '/
usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg', starting on line 112)
Warning: Duplicate definition found for service 'Total Processes' on host 'localhost' (config file
 '/usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg', starting on line 100)
Warning: Duplicate definition found for service 'Current Users' on host 'localhost' (config file '
/usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg', starting on line 86)
Warning: Duplicate definition found for service 'Root Partition' on host 'localhost' (config file
'/usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg', starting on line 72)
Warning: Duplicate definition found for service 'PING' on host 'localhost' (config file '/usr/loca
l/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg', starting on line 58)
   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
```

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

```
[root@ip-172-31-34-108 ec2-user]# service nagios restart
Redirecting to /bin/systemctl restart nagios.service
[root@ip-172-31-34-108 ec2-user]#
```

Step 11: Now Go to the Nagios-client ssh terminal and update and install the packages by running the following command.

sudo apt update -y

sudo apt install gcc -y

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

```
ubuntu@ip-172-31-47-124:~$ 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 InReleas
e
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-backport
s InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [1
26 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe
 amd64 Packages [15.0 MB]
Get:6 http://security.ubuntu.com/ubuntu noble-security/main amd64 P
ackages [382 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main Transla
tion-en [83.9 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main amd64 c
-n-f Metadata [4704 B]
Get:9 http://security.ubuntu.com/ubuntu noble-security/universe amd
64 Packages [277 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/univers
e Translation-en [5982 kB]
Get:11 http://security.ubuntu.com/ubuntu noble-security/universe Tr
anslation-en [117 kB]
Get:12 http://security.ubuntu.com/ubuntu noble-security/universe am
d64 Components [8632 B]
Get:13 http://security.ubuntu.com/ubuntu noble-security/universe am
d64 c-n-f Metadata [10.4 kB]
Get:14 http://security.ubuntu.com/ubuntu noble-security/multiverse
```

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

```
  GNU nano 7.2              /etc/nagios/nrpe.cfg
#
# NOTE: This option is ignored if NRPE is running under either ine>

nrpe_user=nagios



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

nrpe_group=nagios



# ALLOWED HOST ADDRESSES
# This is an optional comma-delimited list of IP address or hostna>
# that are allowed to talk to the NRPE daemon. Network addresses w>
# (i.e. 192.168.1.0/24) are also supported. Hostname wildcards are>
# supported.
#
# Note: The daemon only does rudimentary checking of the client's >
# address.  I would highly recommend adding entries in your /etc/h>
# file to allow only the specified host to connect to the port
# you are running this daemon on.
#
# NOTE: This option is ignored if NRPE is running under either ine>

allowed_hosts=127.0.0.1,::1,54.161.62.217
```

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

```
ubuntu@ip-172-31-47-124:~$ sudo systemctl restart nagios-nrpe-serve
r
```

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

sudo systemctl status nagios

```
[ec2-user@ip-172-31-34-108 ~]$ 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 15:39:38 UTC; 14min ago
       Docs: https://www.nagios.org/documentation
    Process: 21775 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg >
    Process: 21776 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (co>
   Main PID: 21781 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 4.1M
        CPU: 244ms
     CGroup: /system.slice/nagios.service
             ├─21781 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─21782 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─21783 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─21784 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─21785 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─21790 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 04 15:39:38 ip-172-31-34-108.ec2.internal nagios[21781]: Warning: Duplicate definition found >
Oct 04 15:39:38 ip-172-31-34-108.ec2.internal nagios[21781]: Warning: Duplicate definition found >
Oct 04 15:39:38 ip-172-31-34-108.ec2.internal nagios[21781]: Warning: Duplicate definition found >
Oct 04 15:39:38 ip-172-31-34-108.ec2.internal nagios[21781]: Warning: Duplicate definition found >
Oct 04 15:39:38 ip-172-31-34-108.ec2.internal nagios[21781]: Warning: Duplicate definition found >
Oct 04 15:39:38 ip-172-31-34-108.ec2.internal nagios[21781]: Warning: Duplicate definition found >
Oct 04 15:39:38 ip-172-31-34-108.ec2.internal nagios[21781]: Warning: Duplicate definition found >
Oct 04 15:39:38 ip-172-31-34-108.ec2.internal nagios[21781]: Warning: Duplicate definition found >
Oct 04 15:39:38 ip-172-31-34-108.ec2.internal nagios[21781]: Successfully launched command file w>
Oct 04 15:40:20 ip-172-31-34-108.ec2.internal nagios[21781]: HOST ALERT: linuxserver;UP;SOFT;1;PI>
lines 1-28/28 (END)
```

sudo systemctl status httpd

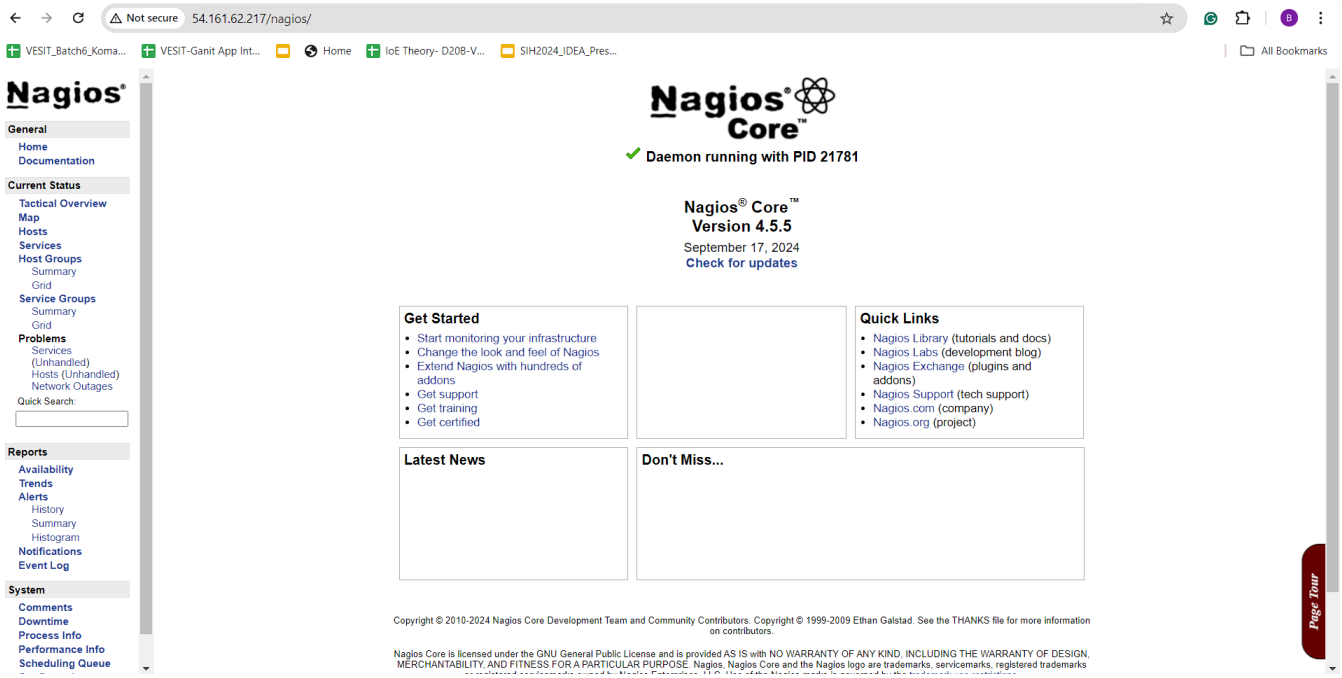
```
[ec2-user@ip-172-31-34-108 ~]$ sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
    Drop-In: /usr/lib/systemd/system/httpd.service.d
             └─php-fpm.conf
     Active: active (running) since Fri 2024-10-04 14:18:18 UTC; 1h 37min ago
       Docs: man:httpd.service(8)
   Main PID: 2495 (httpd)
     Status: "Total requests: 48; Idle/Busy workers 100/0;Requests/sec: 0.0082; Bytes served/sec:>
      Tasks: 230 (limit: 1112)
     Memory: 24.0M
        CPU: 3.655s
     CGroup: /system.slice/httpd.service
             ├─ 2495 /usr/sbin/httpd -DFOREGROUND
             ├─ 2543 /usr/sbin/httpd -DFOREGROUND
             ├─ 2544 /usr/sbin/httpd -DFOREGROUND
             ├─ 2545 /usr/sbin/httpd -DFOREGROUND
             ├─ 2546 /usr/sbin/httpd -DFOREGROUND
             └─20138 /usr/sbin/httpd -DFOREGROUND

Oct 04 14:18:18 ip-172-31-34-108.ec2.internal systemd[1]: Starting httpd.service - The Apache HTT>
Oct 04 14:18:18 ip-172-31-34-108.ec2.internal systemd[1]: Started httpd.service - The Apache HTTP>
Oct 04 14:18:18 ip-172-31-34-108.ec2.internal httpd[2495]: Server configured, listening on: port >
lines 1-22/22 (END)
```

sudo systemctl start httpd

sudo systemctl enable httpd

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

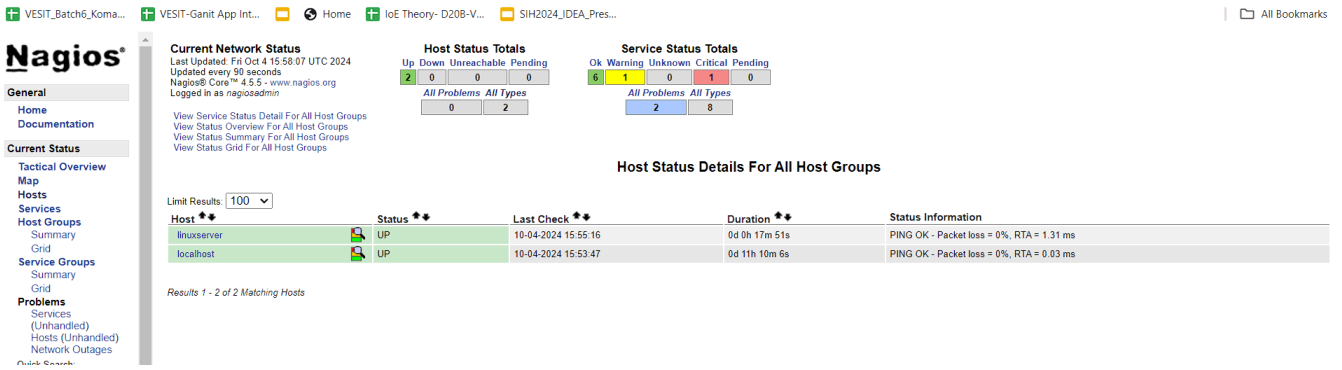
Brijesh R. Sharma **Division: D15C** **Roll No:48**



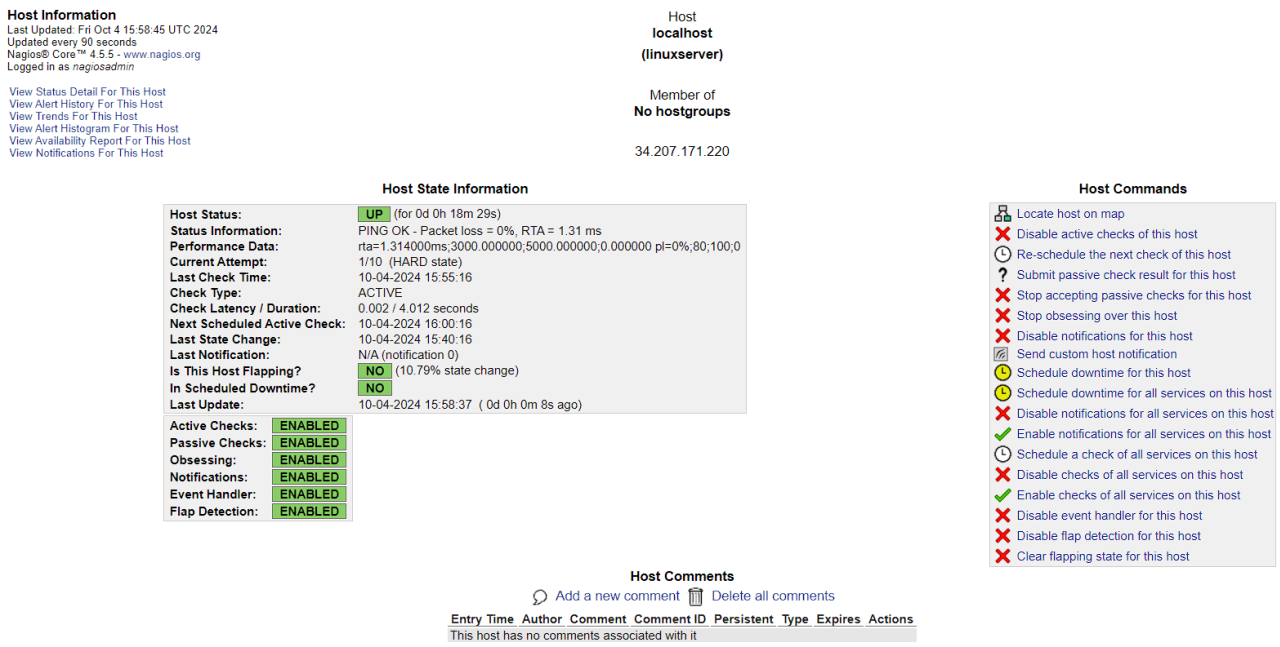
Now Click on Hosts from left side panel

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

Current Network Status



Brijesh R. Sharma **Division: D15C** **Roll No:48**



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

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

Theory:

AWS Lambda

A fully managed, serverless computing service where you run code without provisioning or managing servers. Lambda automatically scales your application based on the number of incoming requests or events, ensuring efficient resource utilization. You are only charged for the time your code is running, with no upfront cost, making it cost-effective for on-demand workloads.

Lambda Workflow

- **Create a Function**: Write the function code and define its handler (entry point). You can use the AWS Console, CLI, or upload a deployment package.
- **Set Event Sources**: Define how the function is triggered (e.g., when an object is uploaded to S3 or a DynamoDB table is updated).
- **Execution**: When triggered, Lambda runs your function, executes the logic, and automatically scales to handle the incoming event volume.
- **Scaling and Concurrency**: Lambda scales automatically by launching more instances of the function to handle simultaneous invocations. There are also options for configuring **reserved concurrency** to manage traffic.
- **Monitoring and Logging**: Lambda integrates with Amazon CloudWatch for logging and monitoring. Logs for each invocation are sent to CloudWatch, allowing you to track performance and troubleshoot errors.

AWS Lambda Functions

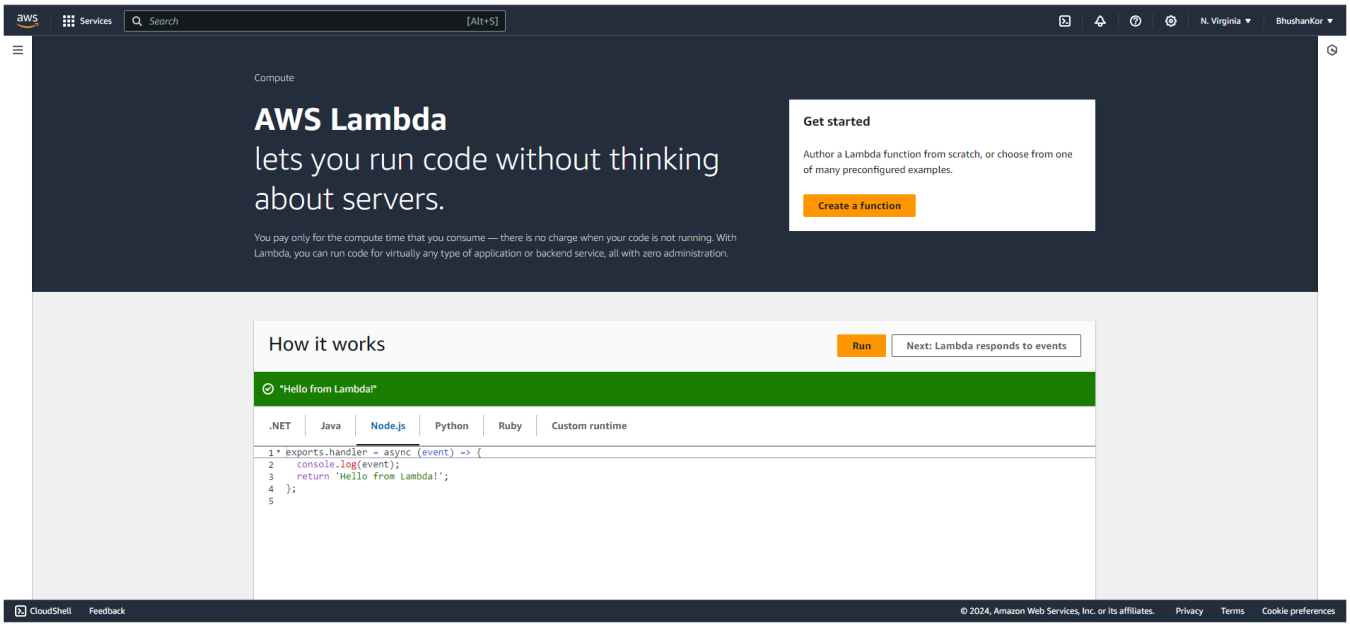
- **Python**: Great for quick development with its rich standard library and support for lightweight tasks.
- **Java**: Typically used for more complex, compute-intensive tasks. While it's robust, cold start times can be higher.
- **Node.js**: Excellent for I/O-bound tasks like handling APIs or streaming data, with fast startup times and efficient memory usage.

Prerequisites: AWS Personal/Academy Account

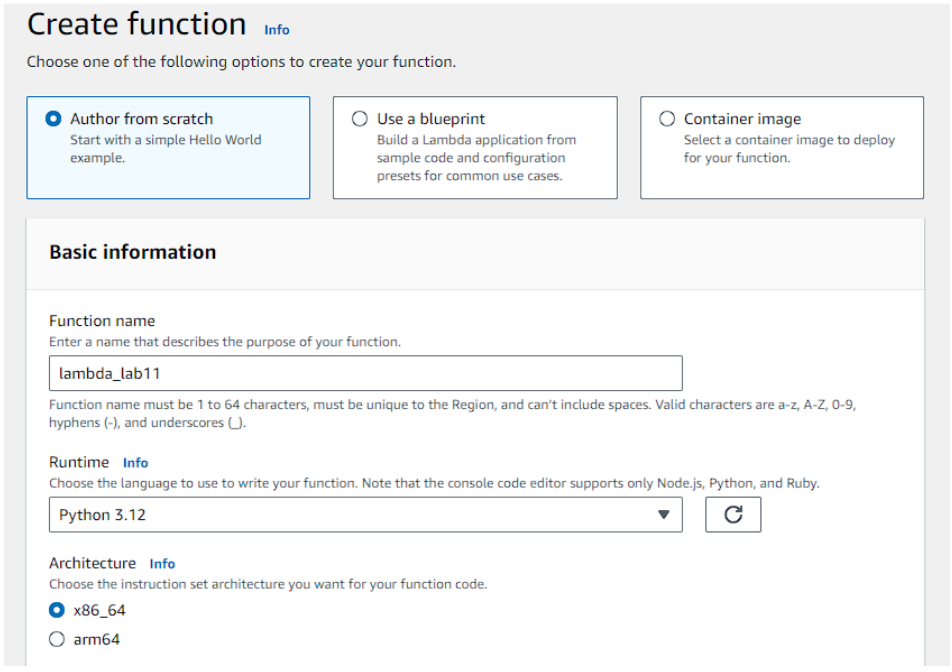
Brijesh Sharma **Roll No. 48** **Div: D15C**

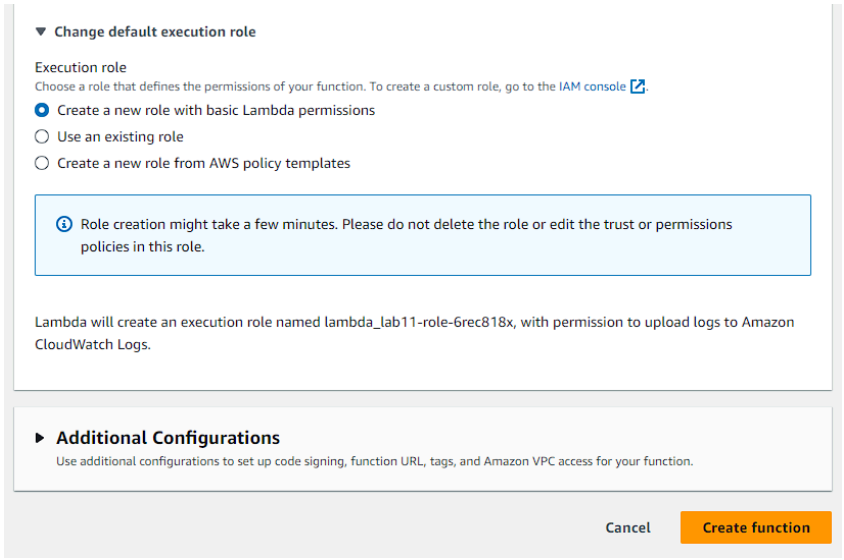
Steps To create the lambda function:

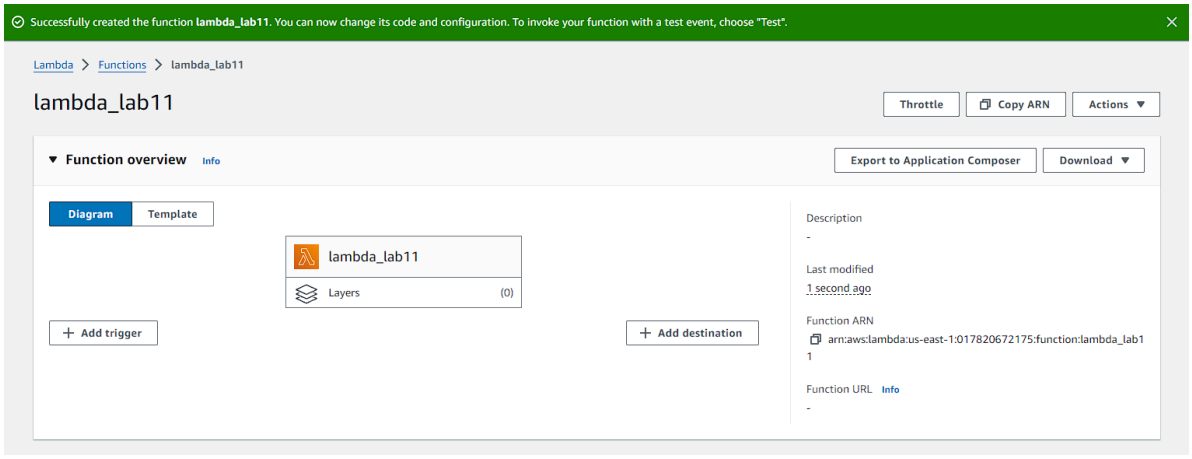
Step 1: Login to your AWS Personal/Academy Accout.Open lambda andclick on create function button.

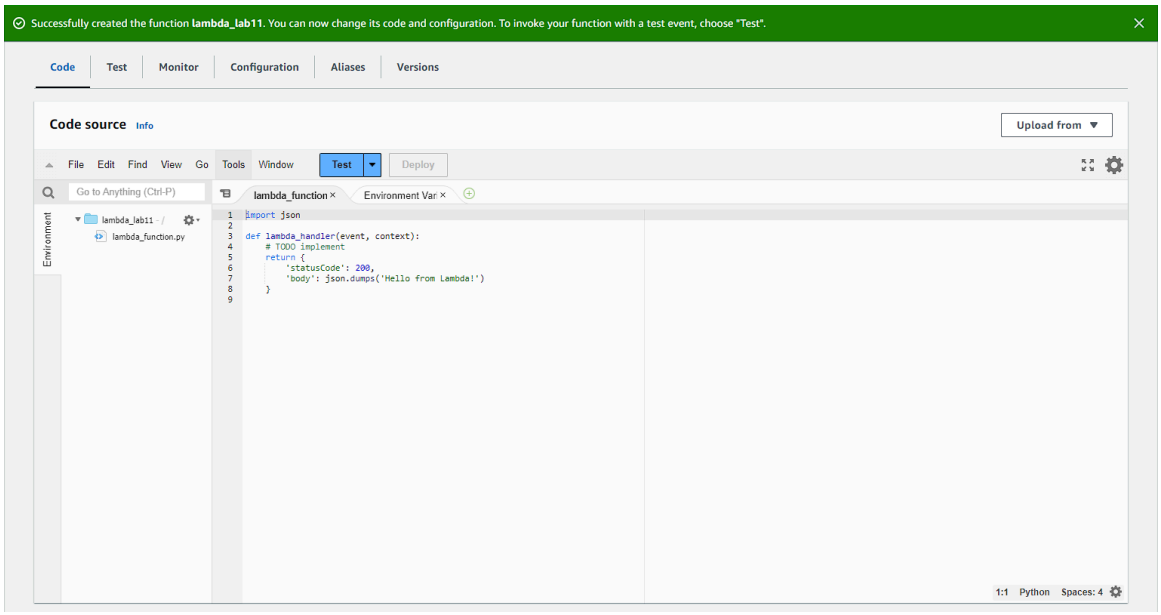


Step 2: Now Give a name to your Lambda function, Select the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby. So will select Python 3.12 , Architecture as x86, and Exceution role to Create a new role with basic Lambda permissions.

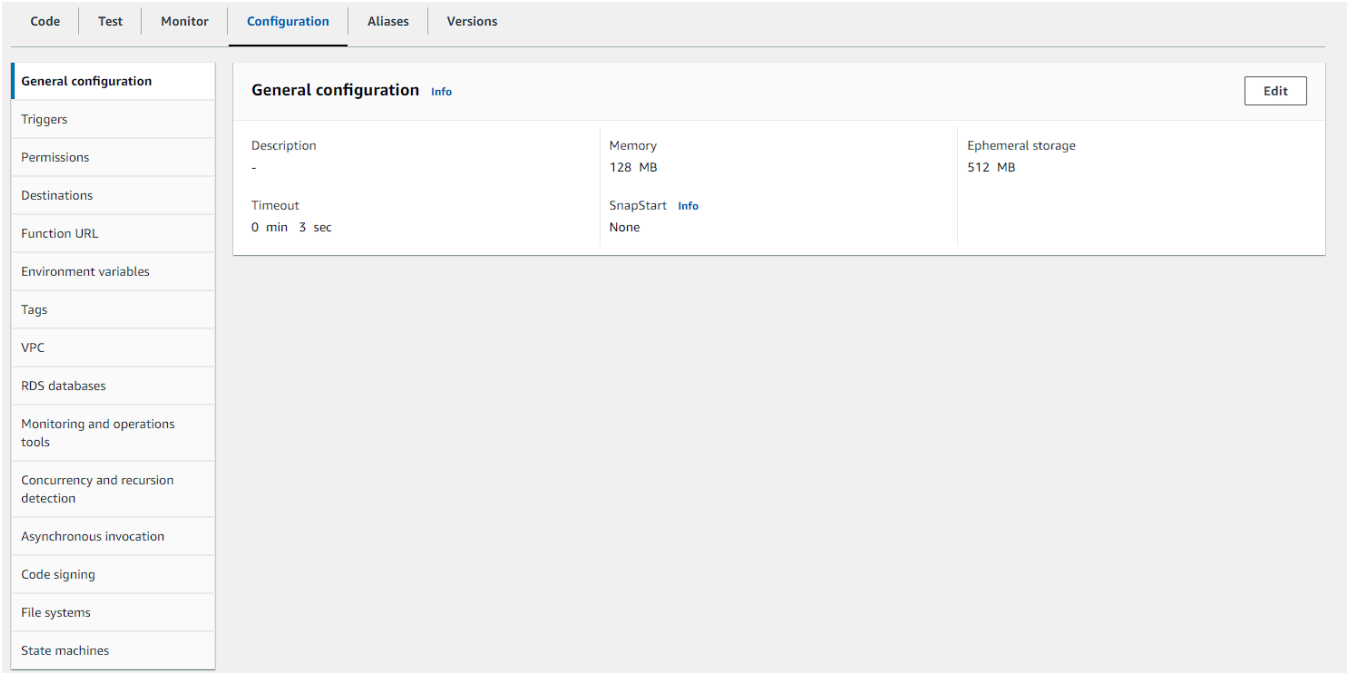



Change default execution role
Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
Create a new role with basic Lambda permissions
Use an existing role
Create a new role from AWS policy templates
Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.
Lambda will create an execution role named lambda_lab11-role-6rec818x, with permission to upload logs to Amazon CloudWatch Logs.
Additional Configurations
Use additional configurations to set up code signing, function URL, tags, and Amazon VPC access for your function.
Cancel
Create function

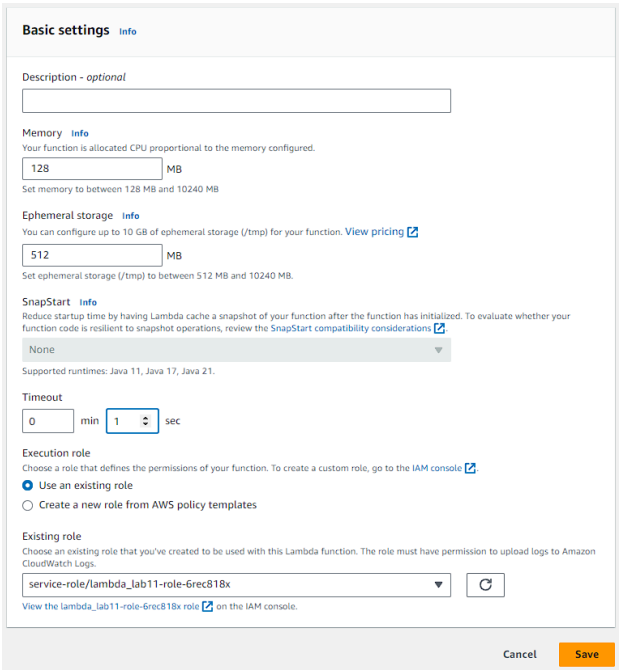

Successfully created the function lambda_lab11. You can now change its code and configuration. To invoke your function with a test event, choose "Test".
Lambda > Functions > lambda_lab11
lambda_lab11
Throttle
Copy ARN
Actions
Function overview
Info
Export to Application Composer
Download
Diagram
Template
lambda_lab11
Layers
(0)
Add trigger
Add destination
Description
-
Last modified
1 second ago
Function ARN
arn:aws:lambda:us-east-1:017820672175:function:lambda_lab11
Function URL
Info
-


Successfully created the function lambda_lab11. You can now change its code and configuration. To invoke your function with a test event, choose "Test".
Code
Test
Monitor
Configuration
Aliases
Versions
Code source
Info
Upload from
File
Edit
Find
View
Go
Tools
Window
Test
Deploy
Go to Anything (Ctrl-P)
lambda_function
Environment Var
Environment
lambda_lab11
lambda_function.py
import json
def lambda_handler(event, context):
# TODO implement
return {
'statusCode': 200,
'body': json.dumps('Hello from Lambda!')
}
1:1
Python
Spaces: 4

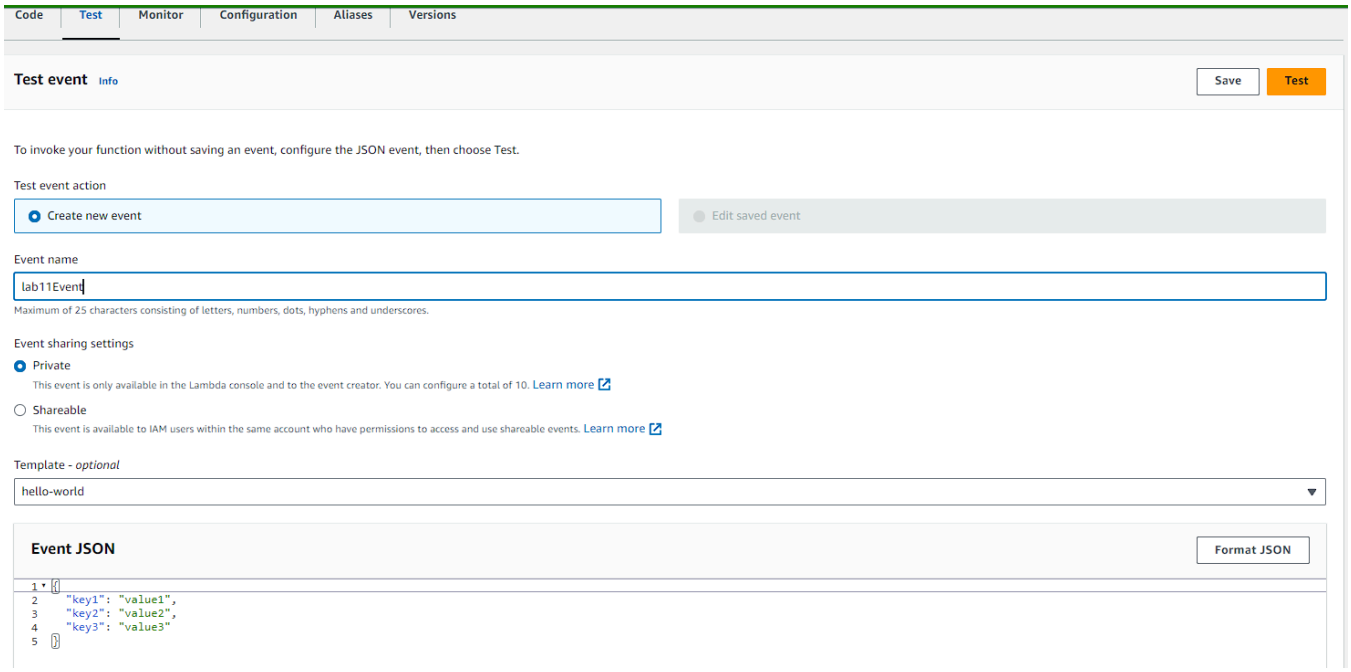
So See or Edit the basic settings go to configuration then click on edit general setting.



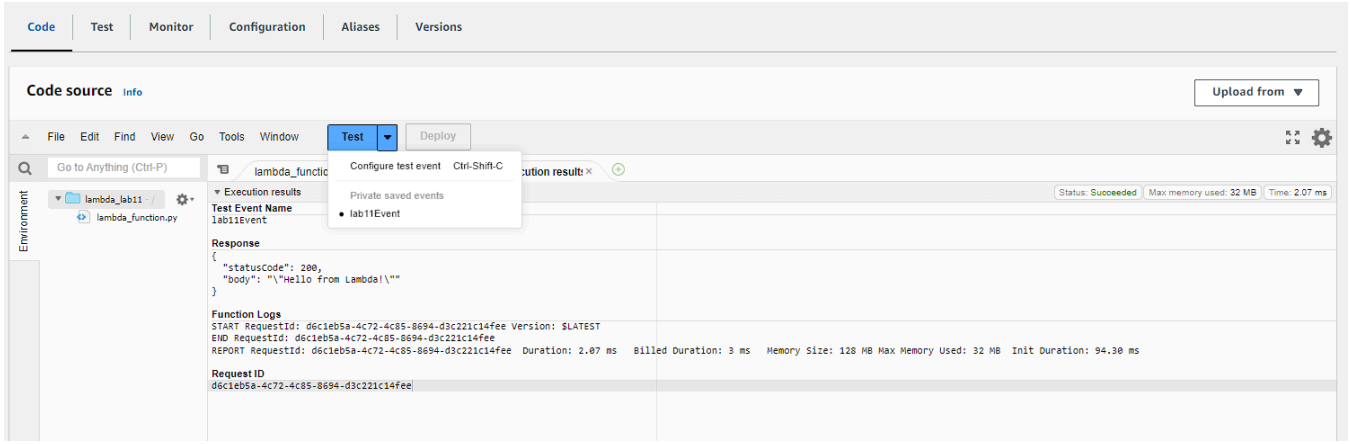
Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.



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



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



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

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

Conclusion: In this experiment, we successfully created an AWS Lambda function and walked through its essential steps. After setting up the function with Python, we configured the basic settings, including adjusting the timeout to 1 second. We then created a test event, deployed the function, and validated the output. Additionally, we modified the Lambda function's code and redeployed it to observe the changes in real-time.This practical experience demonstrated the simplicity and flexibility of AWS Lambda in creating serverless applications, allowing you to focus on code while AWS manages the infrastructure and scaling.

Brijesh R Sharma **Roll No. 48** **D15C**

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

Theory:

AWS Lambda and S3 Integration:

AWS Lambda allows you to execute code in response to various events, including those triggered by Amazon S3. When an object is added to an S3 bucket, it can trigger a Lambda function to execute, allowing for event-driven processing without managing servers.

Workflow:

1. Create an S3 Bucket:

○ First, create an S3 bucket that will store the objects. This bucket will act as the trigger source for the Lambda function.

2. Create the Lambda Function:

○ Set up a new Lambda function using AWS Lambda’s console. You can choose a runtime environment like Python, Node.js, or Java.
○ Write code that logs a message like “An Image has been added” when triggered.

3. Set Up Permissions:

○ Ensure that the Lambda function has the necessary permissions to access S3. You can do this by attaching an IAM role with policies that allow reading from the bucket and writing logs to CloudWatch.

4. Configure S3 Trigger:

○ Link the S3 bucket to the Lambda function by setting up a trigger. Specify that the function should be triggered when an object is created in the bucket (e.g., when an image is uploaded).

5. Test the Setup:

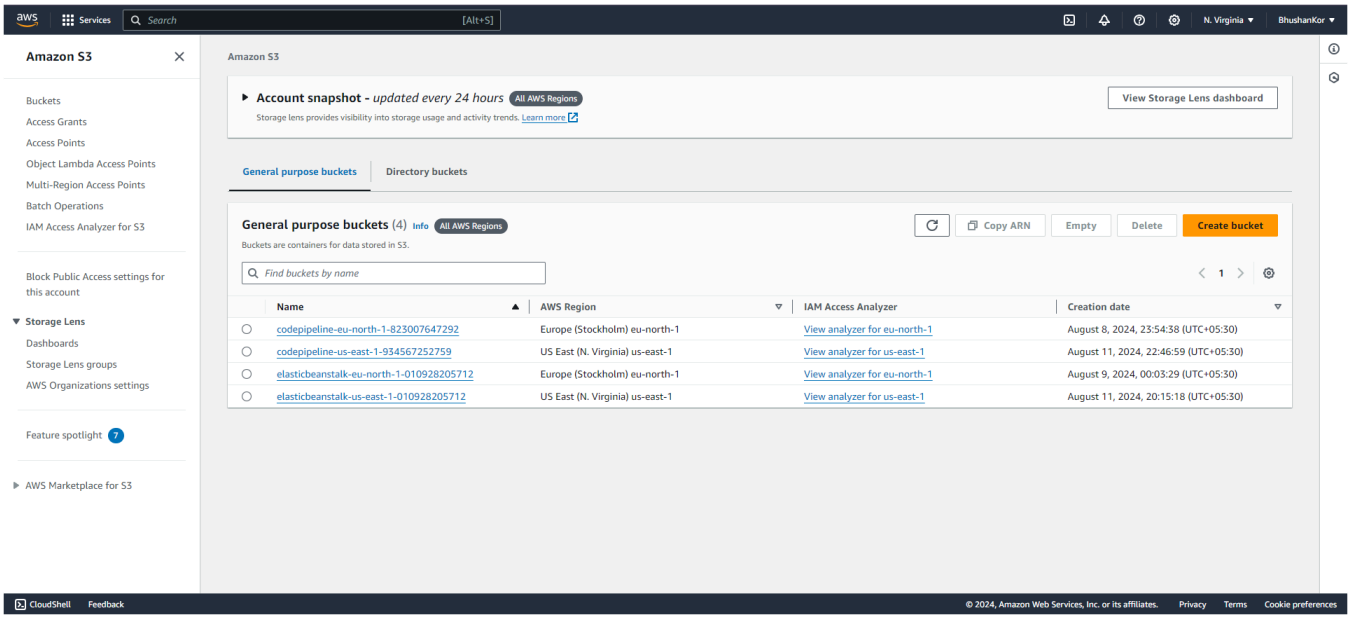
○ Upload an object (e.g., an image) to the S3 bucket to test the trigger. The Lambda function should execute and log the message “An Image has been added” in AWS CloudWatch Logs.

Prerequisites: AWS Personal Account

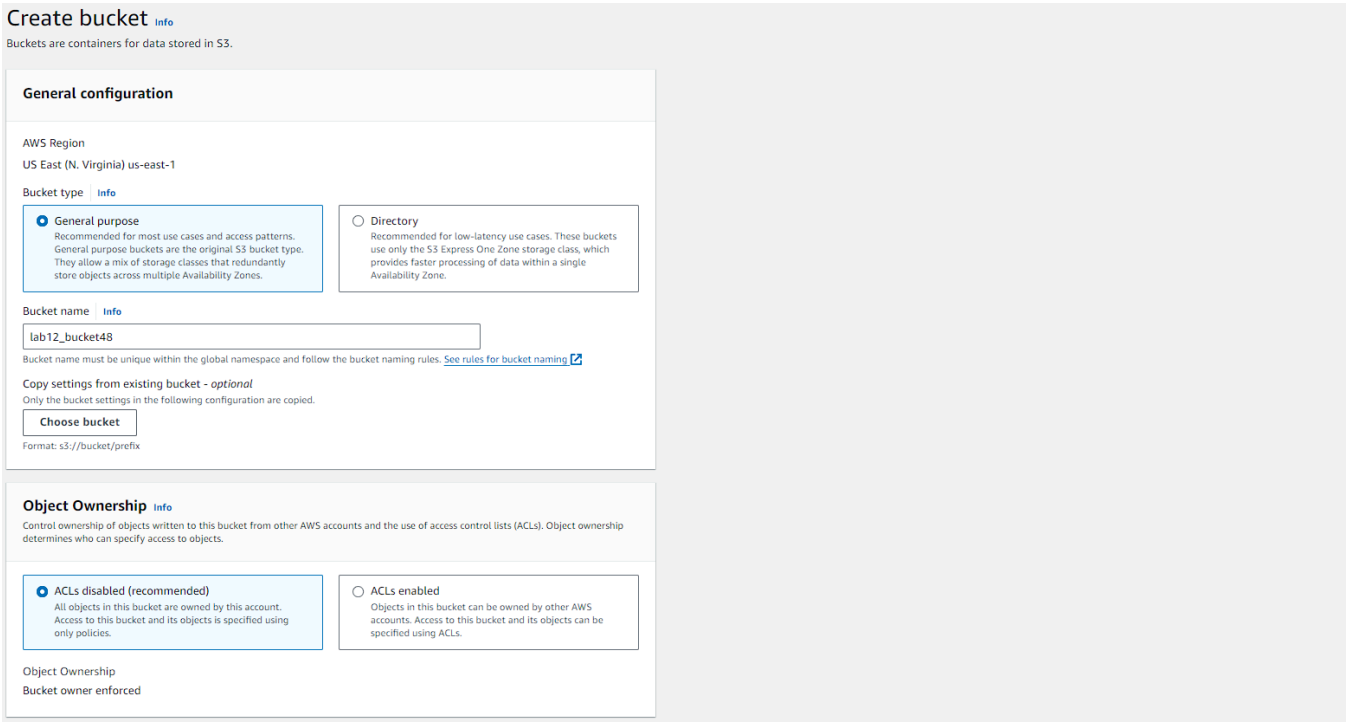
Brijesh R Sharma **Roll No. 48** **D15C**

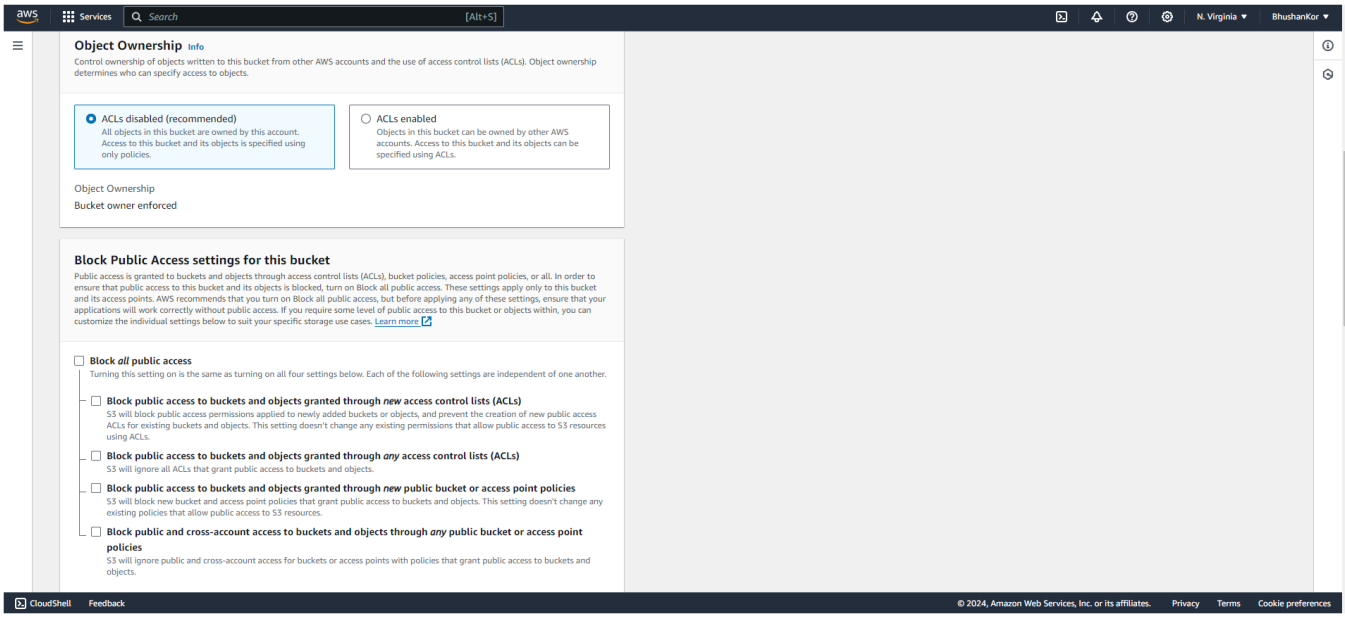
Steps To create the lambda function:

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



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





Block Public Access settings for this bucket

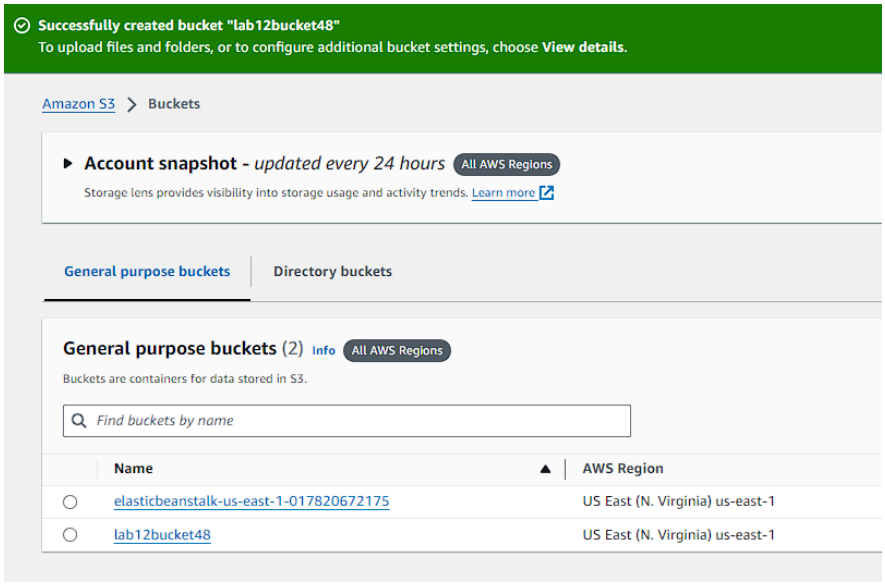
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

- ☐ **Block *all* public access**
  Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.
  - ☐ **Block public access to buckets and objects granted through *new* access control lists (ACLs)**
    S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
  - ☐ **Block public access to buckets and objects granted through *any* access control lists (ACLs)**
    S3 will ignore all ACLs that grant public access to buckets and objects.
  - ☐ **Block public access to buckets and objects granted through *new* public bucket or access point policies**
    S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
  - ☐ **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
    S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

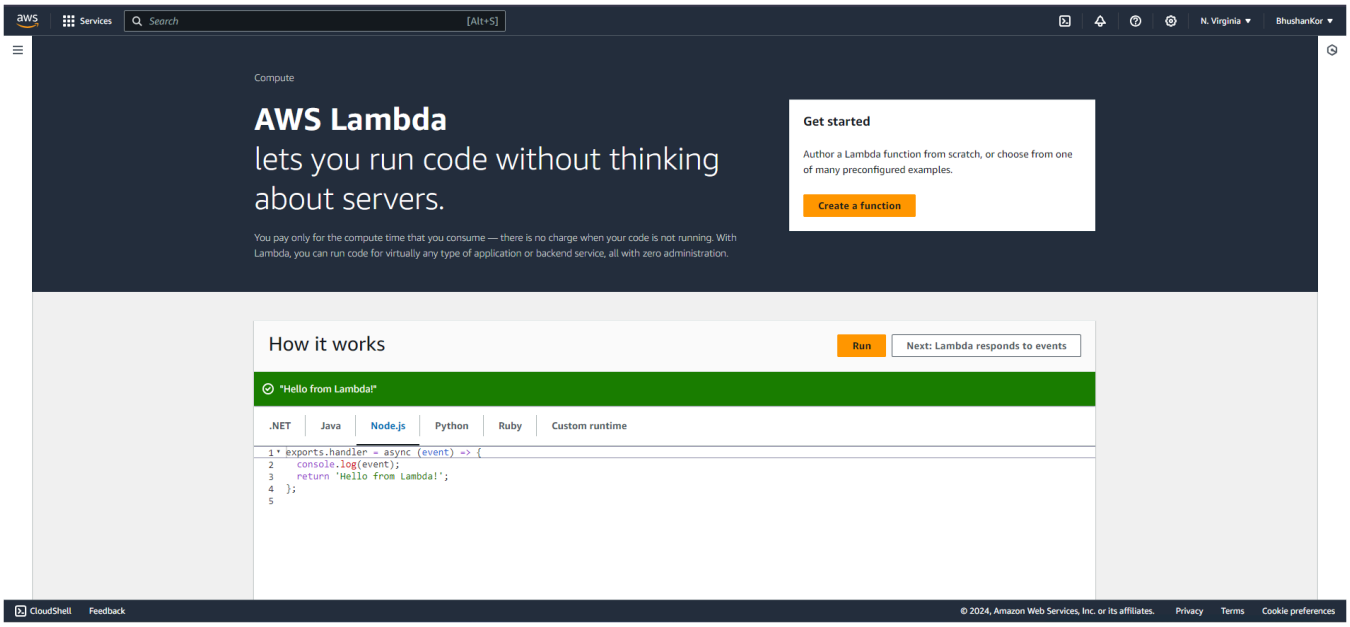
⚠ **Turning off block all public access might result in this bucket and the objects within becoming public**

AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.

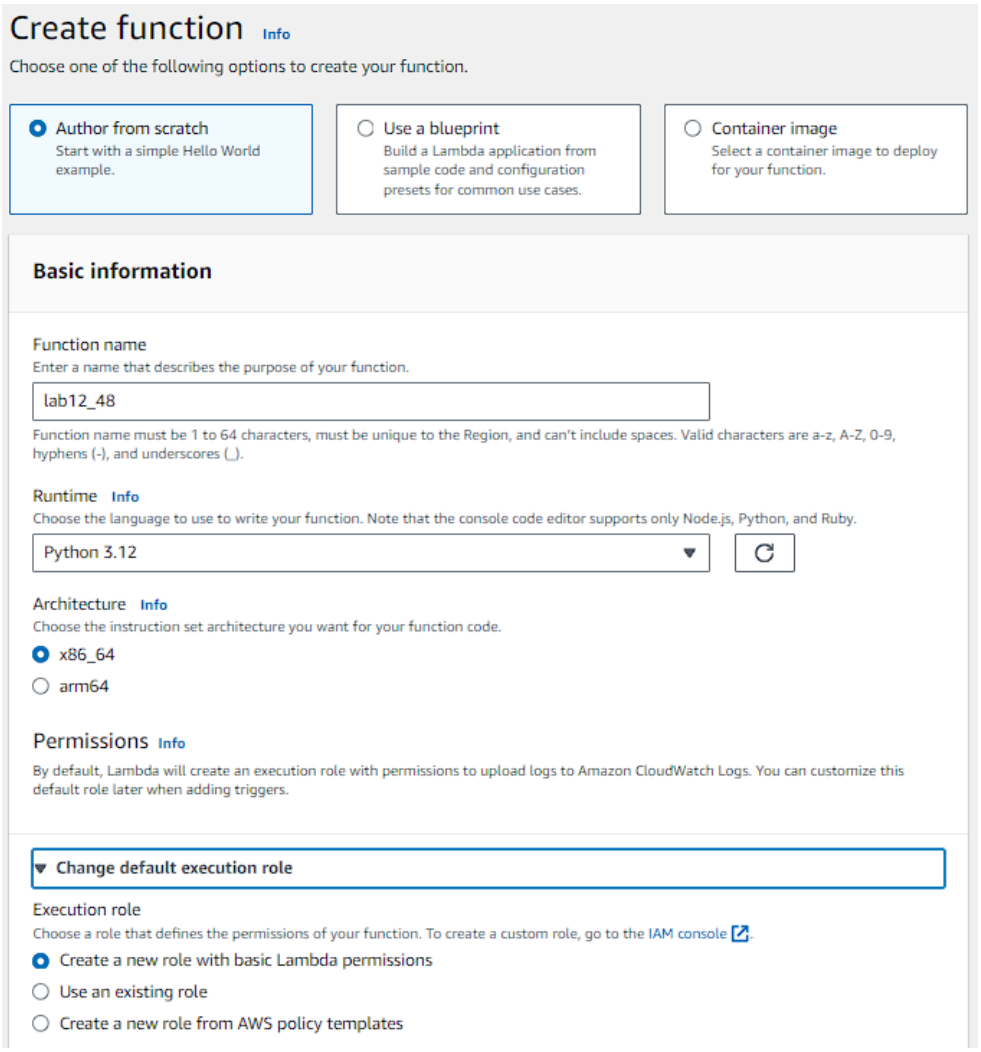
☐ I acknowledge that the current settings might result in this bucket and the objects within becoming public.

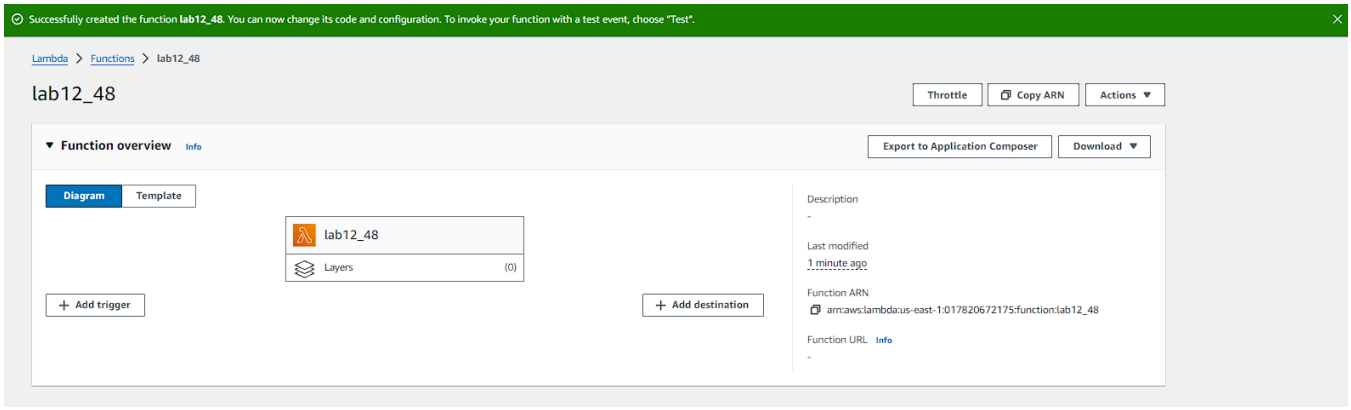


Step 3: Open lambda console and click on create function button.

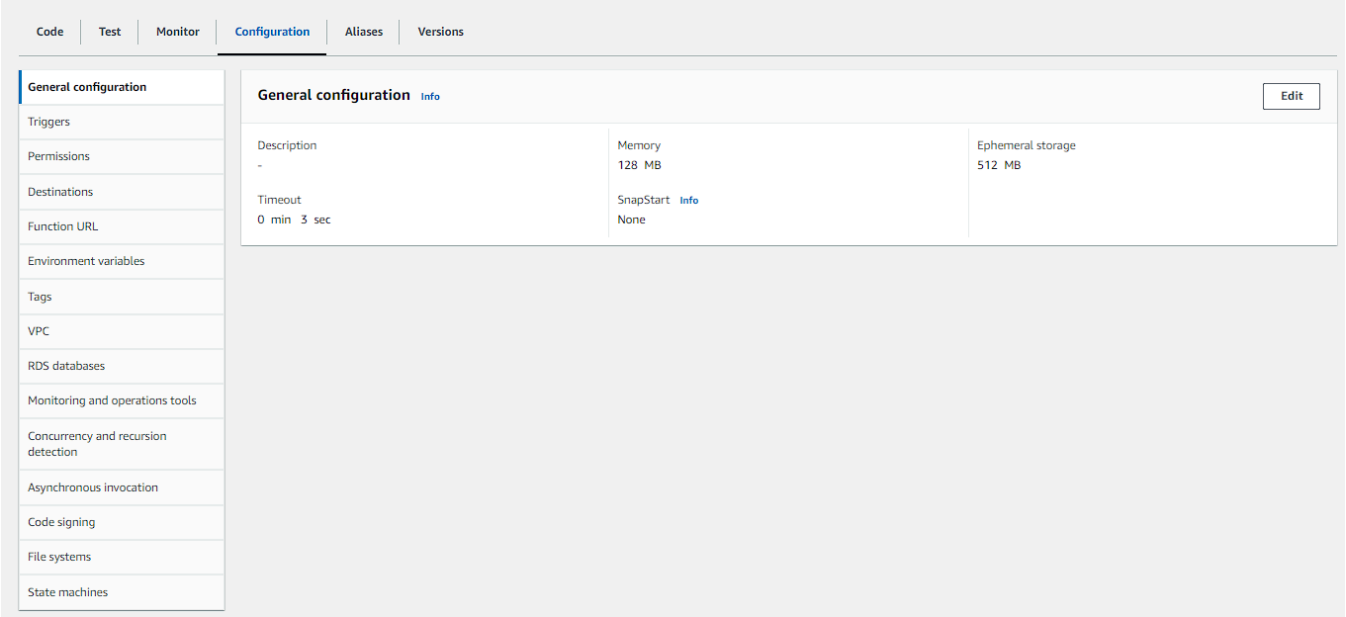


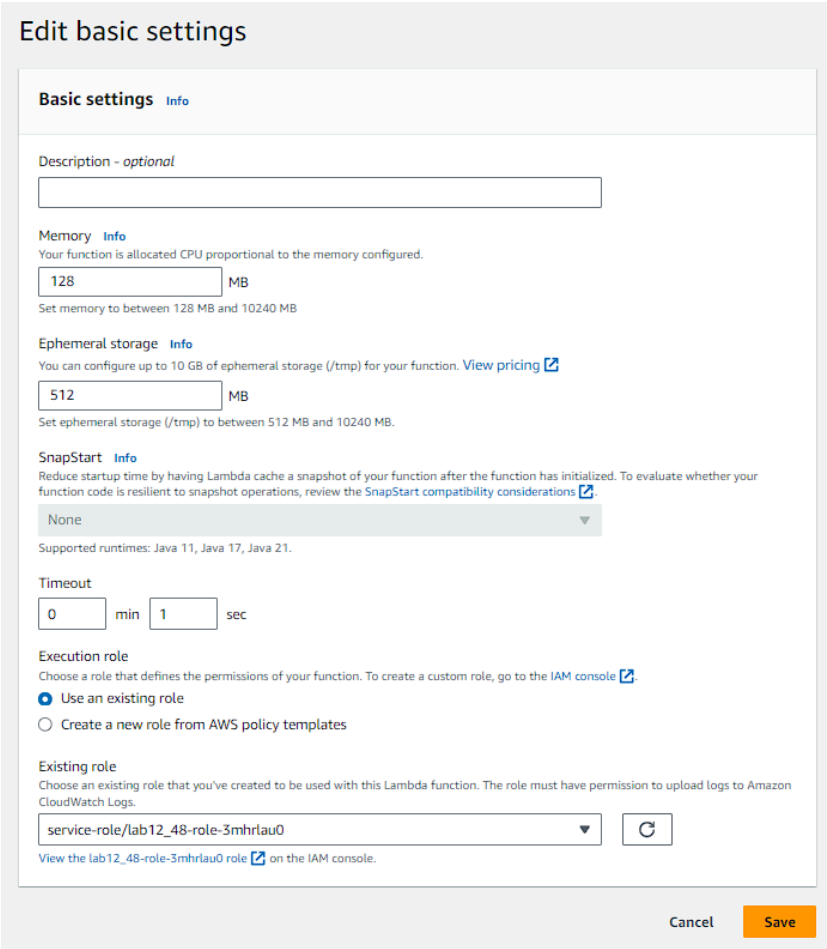
Step 4: Now Give a name to your Lambda function, Select the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby. So will select Python 3.12 , Architecture as x86, and Exceution role to Create a new role with basic Lambda permissions.





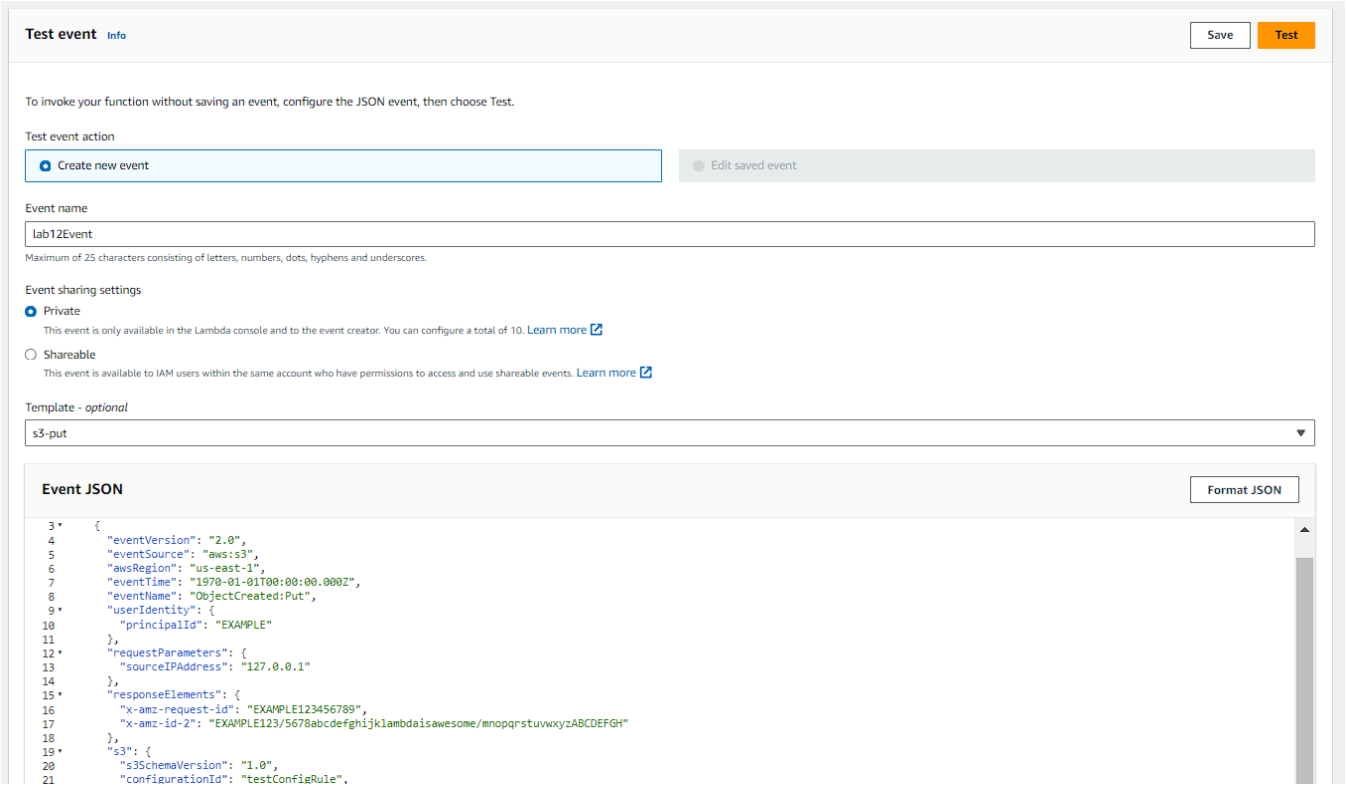
So See or Edit the basic settings go to configuration then click on edit general setting.



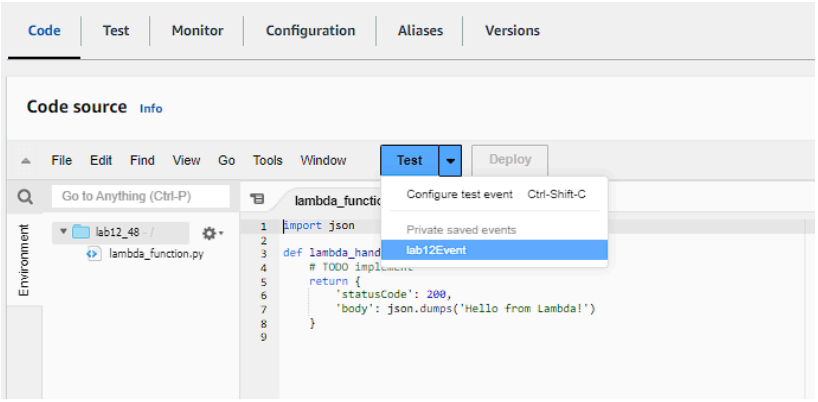


Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.

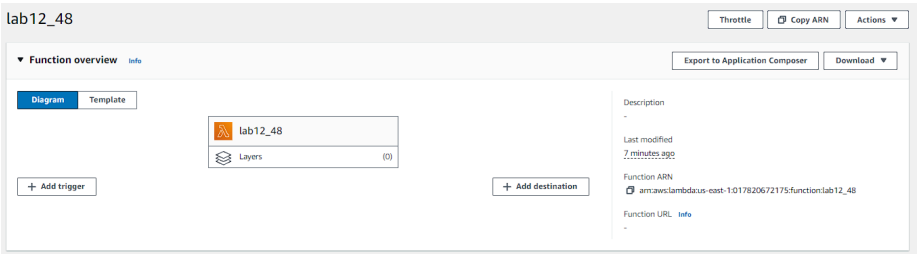
Step 5: Now Click on the Test tab then select Create a new event, give a name to the event and select Event Sharing to private, and select s3 put template.



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



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



Add trigger

Trigger configuration Info

S3
aws asynchronous storage

Bucket
Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.

s3/lab12bucket48

Bucket region: us-east-1

Event types
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.

All object create events

Prefix - *optional*
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters. Any special characters must be URL encoded.

image

Suffix - *optional*
Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters. Any special characters must be URL encoded.

e.g. .jpg

Recursive invocation
If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. Learn more

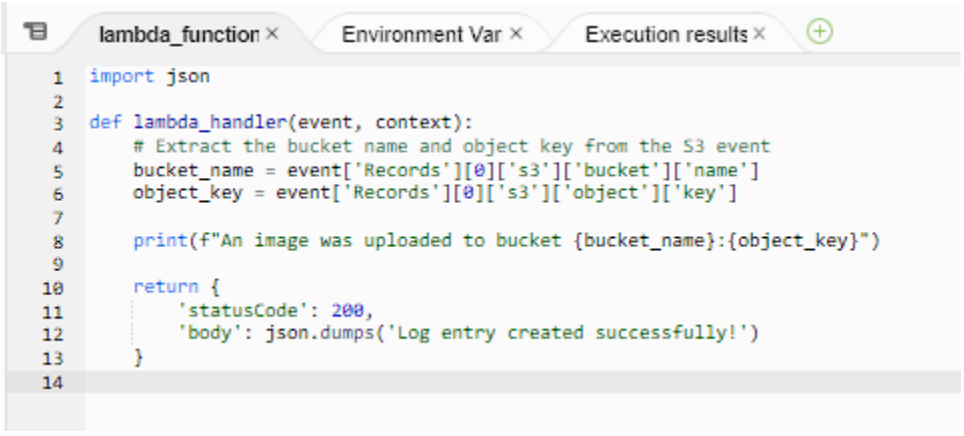
- [x] I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.

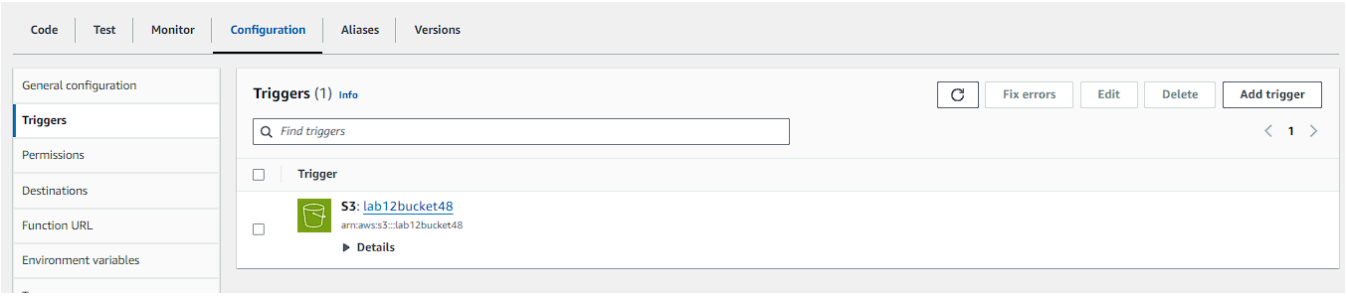
Lambda will add the necessary permissions for AWS S3 to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.

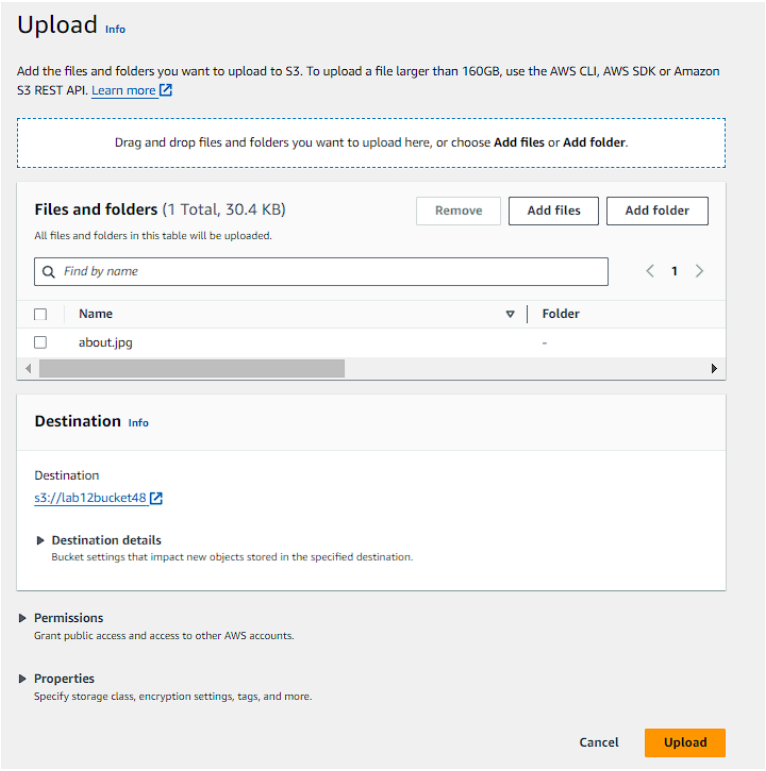
Cancel Add

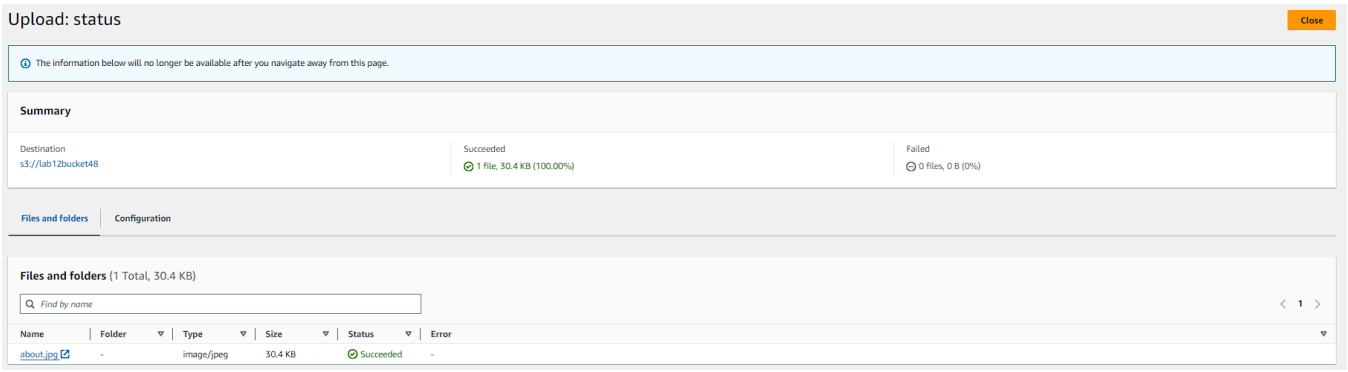
Step 8: Now Write code that logs a message like “An Image has been added” when triggered. Save the file and click on deploy.

Step 9: Now upload any image to the bucket.

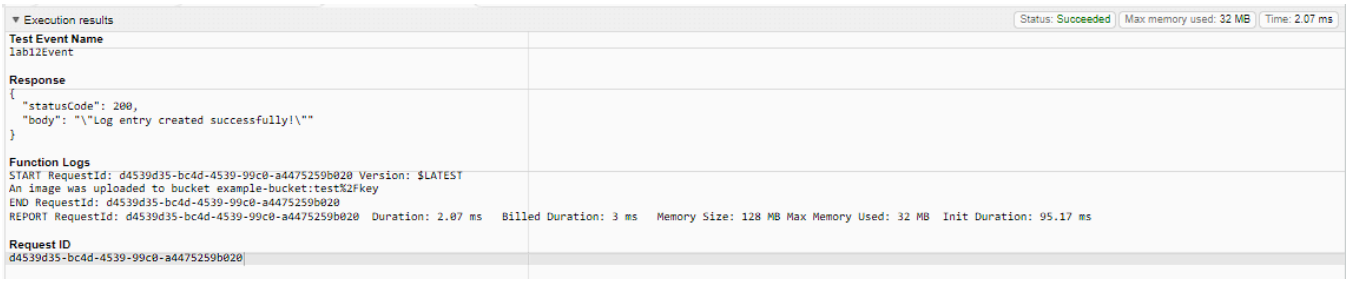




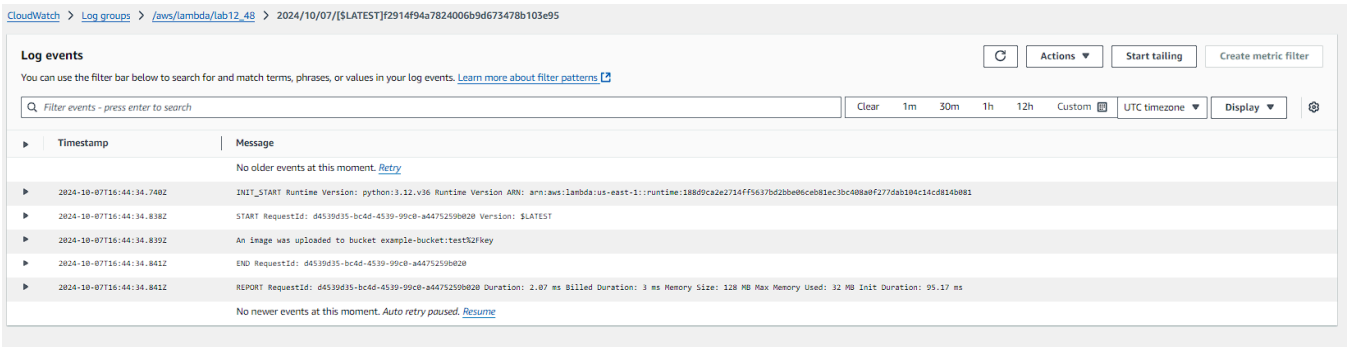




Step 10: Now to click on test in lambda to check whether it is giving log when image is added to S3.



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



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