

Static Hosting [Exp 1(a)]

Aim: To develop a website and host it on:-

1. Local machine or virtual machine.
2. Amazon S3 bucket.

Steps:-

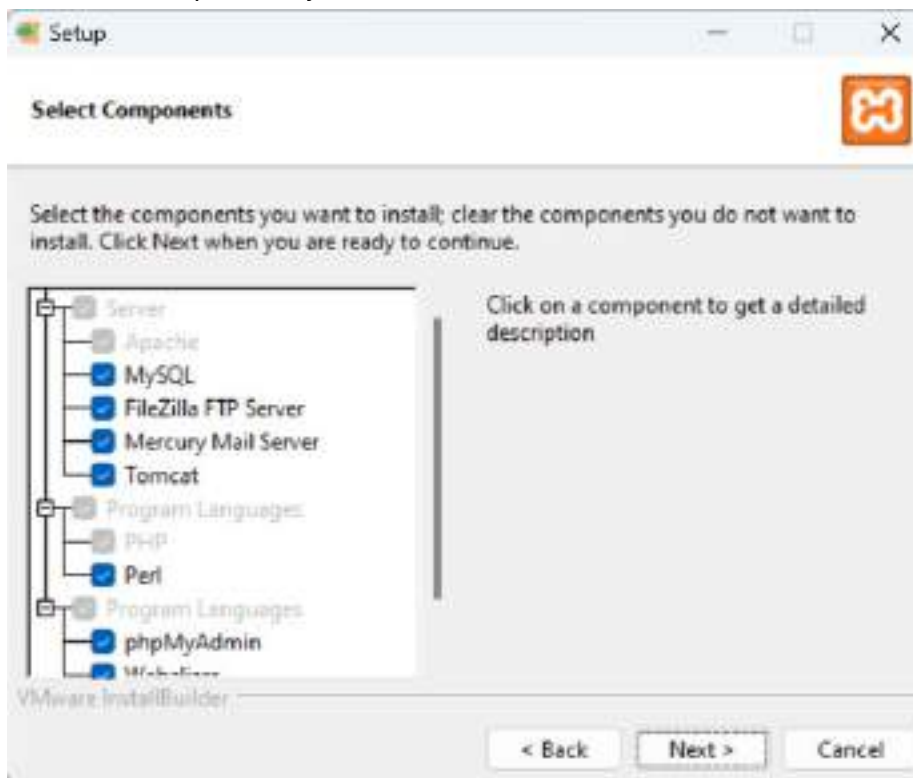
- On local server using XAMPP:-

Step 1: Install XAMPP.

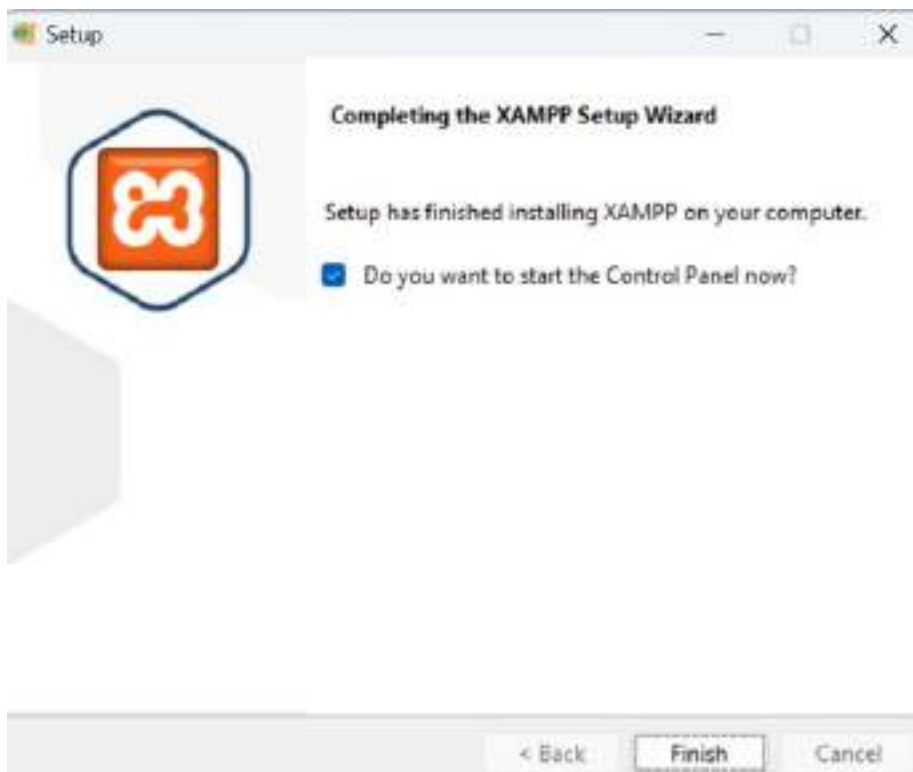
1. Begin the setup process.



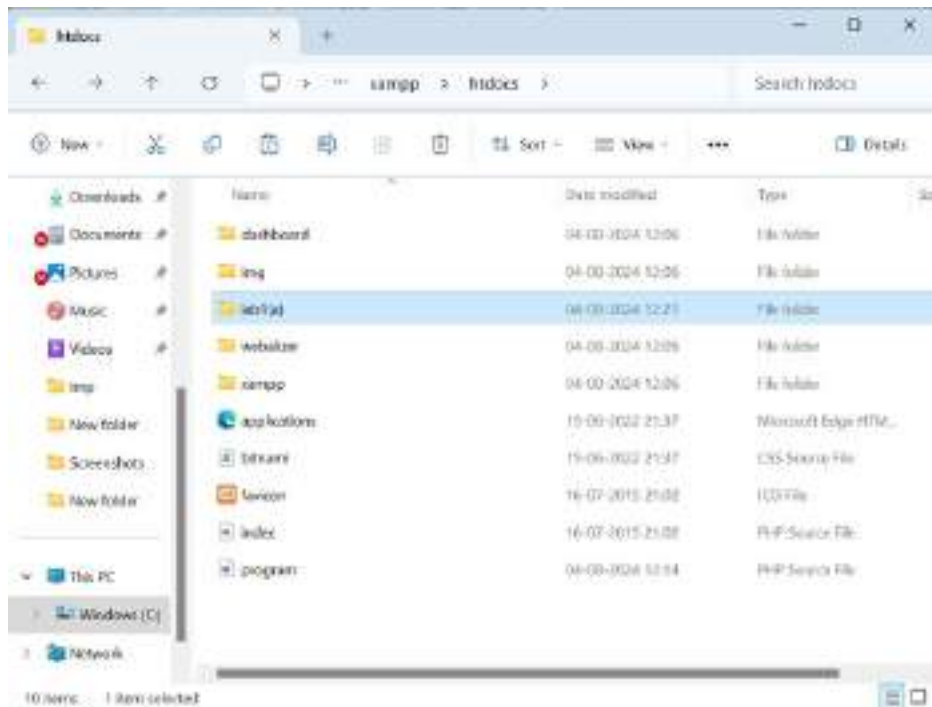
2. Select the components you want to install and click Next.



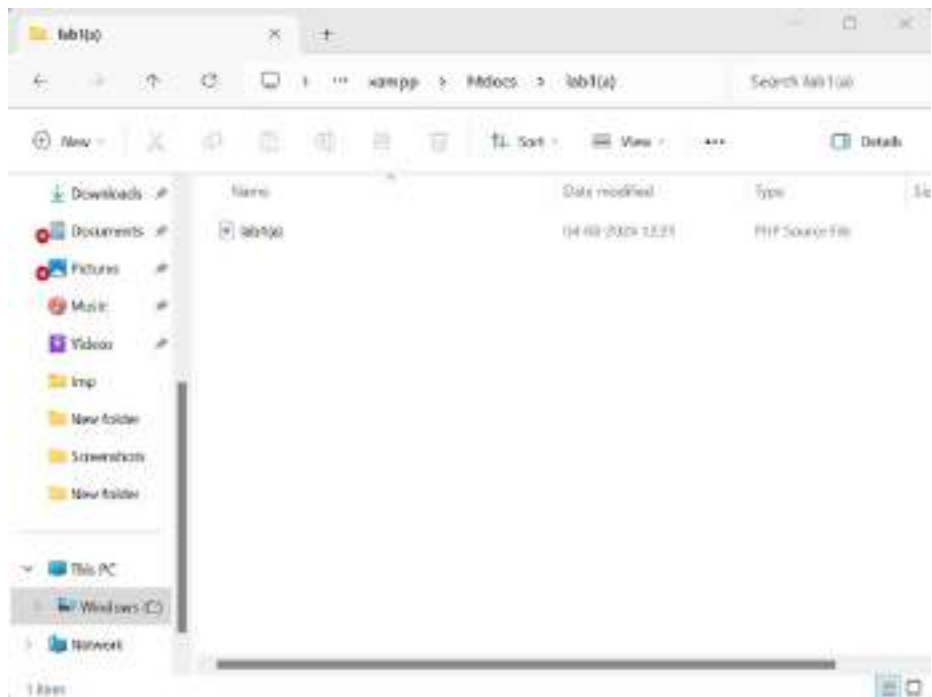
3. Go through the subsequent installation steps (keeping the default options) and finish the installation.



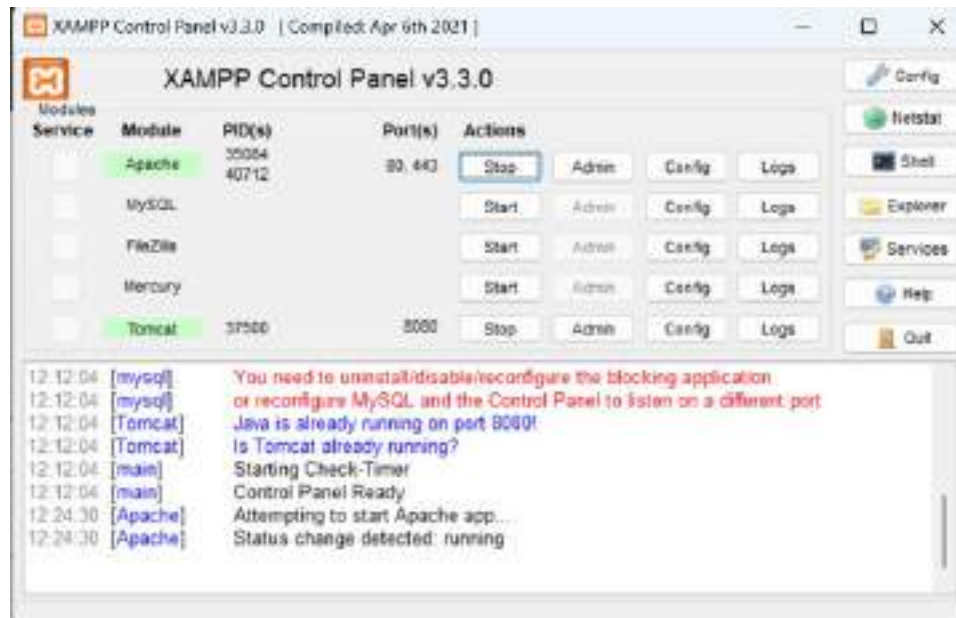
Step 2: Inside the htdocs folder inside of the xampp folder create a new folder.



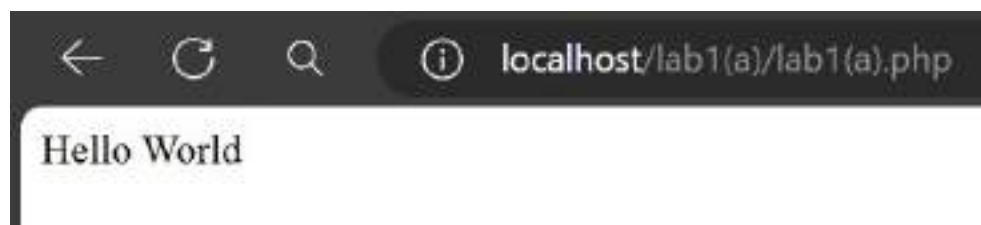
Step 3: Create a file (with extension: .php) that is to be hosted on the local server.



Step 4: Open the XAMPP Control Panel and start the Apache service.

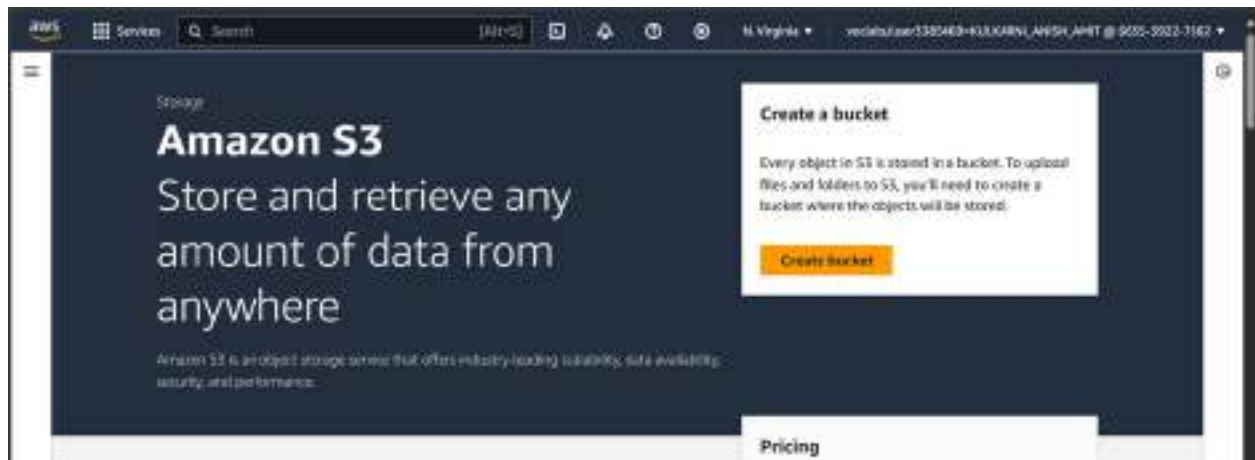


Step 5: Type 'localhost/YOUR_FILENAME.php' into your web browser. This will open your website on your browser.

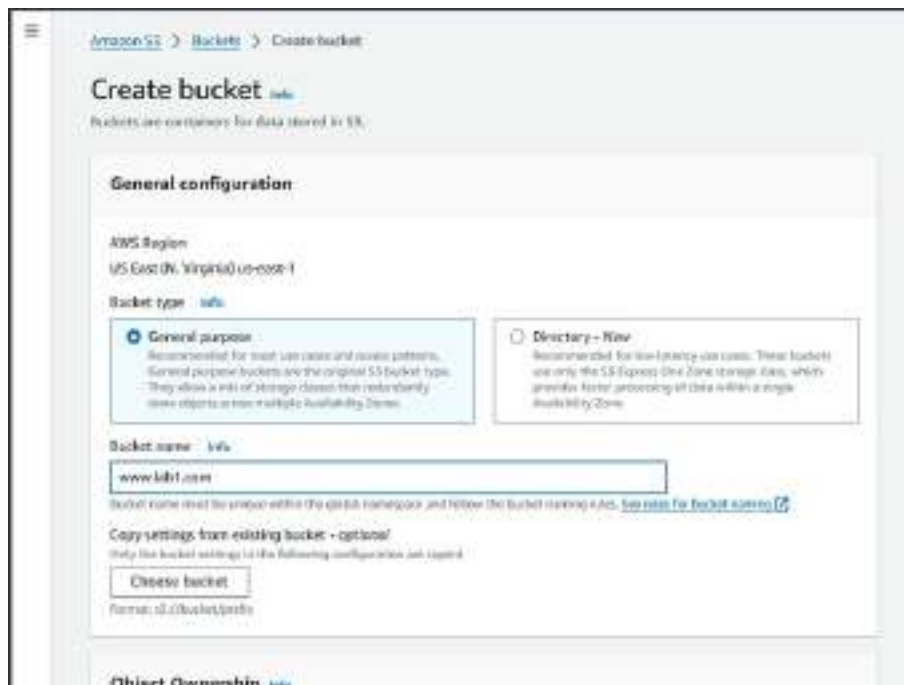


● OnAWS3:-

Step 1: Login into your AWS account, choose the S3 option in the 'Services' tab and click on 'Create Bucket'.



Step 2: Give a name to your bucket and keeping the others options default, click on 'Create Bucket'.



Step 3: In the properties of your bucket, navigate to the 'Edit static website hosting' tab and click on 'Enable'. Also, in the 'Index document' and 'Error document' boxes, enter the names of your index and error websites respectively.

Edit static website hosting [info](#)

Static website hosting

Use this bucket to host a website or redirect requests. [Learn more](#)

Static website hosting

☐ Disable

☒ Enable

Hosting type

☒ Host a static website

Use the bucket endpoint as the web address. [Learn more](#)

☐ Redirect requests for an object

Redirect requests to another bucket or domain. [Learn more](#)

For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see [Using Amazon S3 Block Public Access](#).

Index document

Specify the name of default page of the website.

index.html

Error document - optional

This is returned when a server error occurs.

error.html

Static website hosting

Use this bucket to host a website or redirect requests. [Learn more](#)

Static website hosting

Enabled

Hosting type

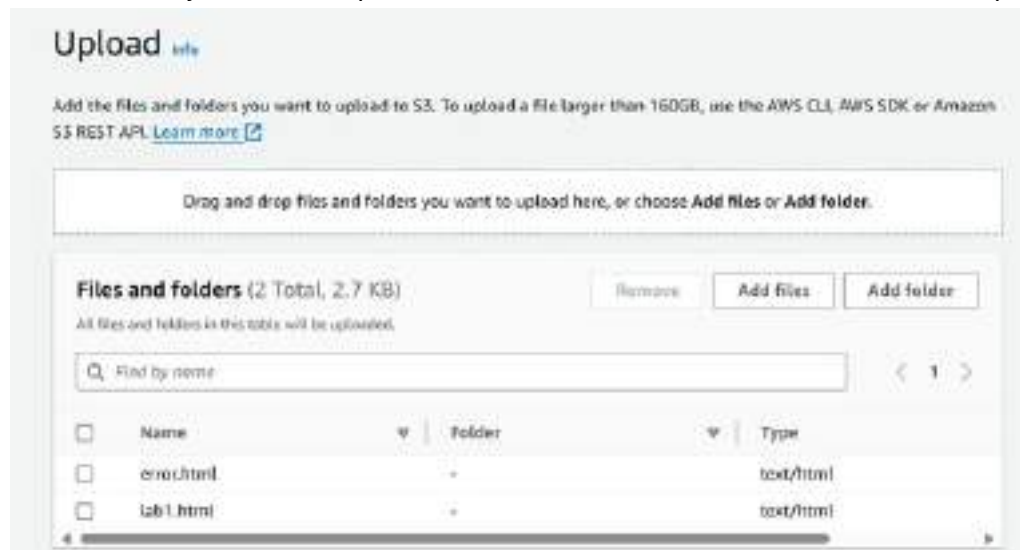
Bucket hosting

Bucket website endpoint

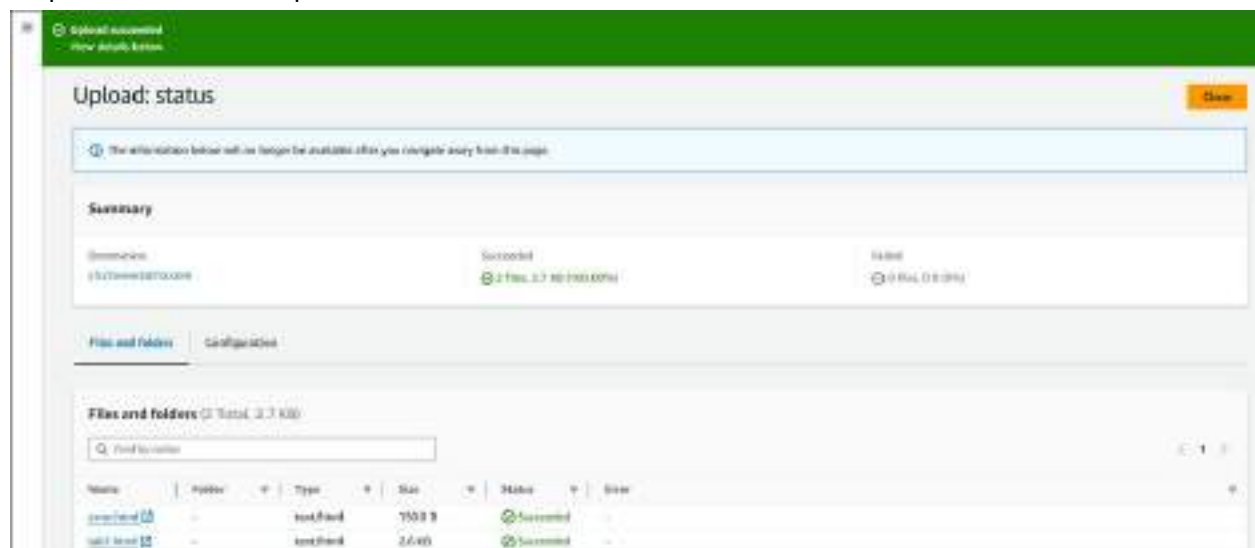
When you configure your bucket as a static website, the website is available at the AWS region-specific website endpoint of the bucket. [Learn more](#)

☒ <http://www.lairia.com.s3-website-us-east-1.amazonaws.com>

Step 4: In your bucket, click on 'Upload' option, click on 'Add files' option and select the websites that you want to upload (index and error websites). Then, click on 'Upload'.



Step 5: Your files are uploaded.



Now, as the contents of the bucket are not yet available to the public, the website does not get displayed and an error message is displayed.

Step 6: To make it public, go to permissions tab, go to 'Block public access' and click on edit. Uncheck the 'Block all public access' checkbox and click on save changes.

Edit Block public access (bucket settings) [info](#)

Block public access (bucket settings)

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 all your S3 buckets and 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 your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

☒ **Block all public access**

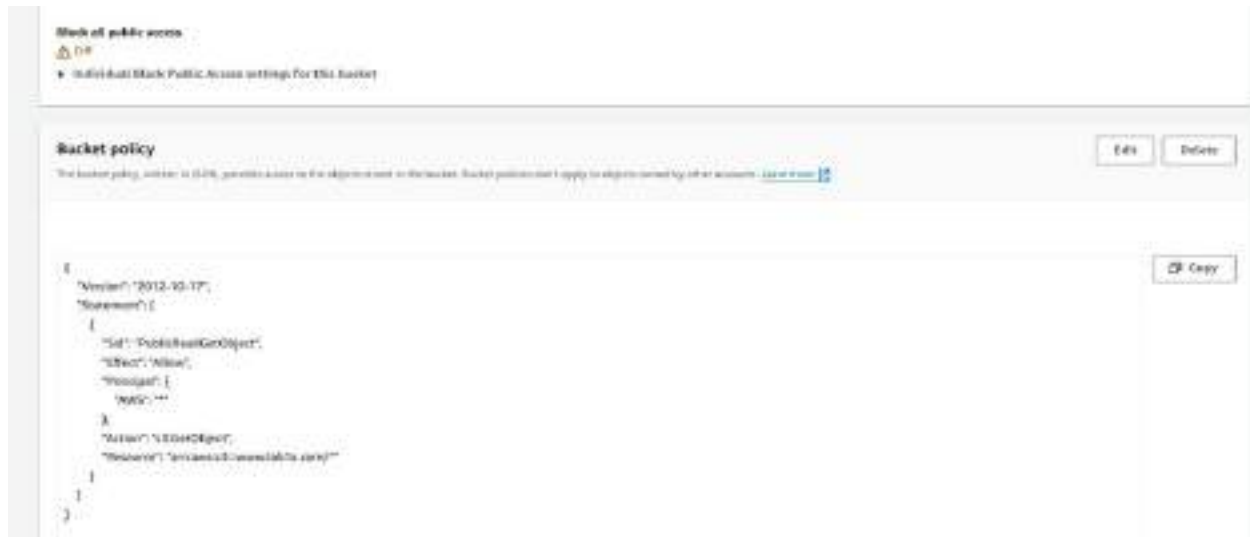
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- ☐ **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.

[Cancel](#) [Save changes](#)

Step 7: Go to 'Bucket policy' tab and click edit. Paste the following code snippet into the text box (replace 'YOUR-BUCKET-NAME-HERE' with the name of your bucket and save changes).

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

Step 8: Click on 'Bucket website endpoint' in the 'Static website hosting' tab in your bucket. Your website is visible.



Conclusion: In this experiment, we learned how to host a static website both on a local server using XAMPP and on AWS S3. By setting up XAMPP, we were able to deploy a local web server and access the hosted site through the browser using localhost. Additionally, we configured an S3 bucket for static website hosting on AWS, where we uploaded files, enabled public access, and applied the necessary bucket policies to

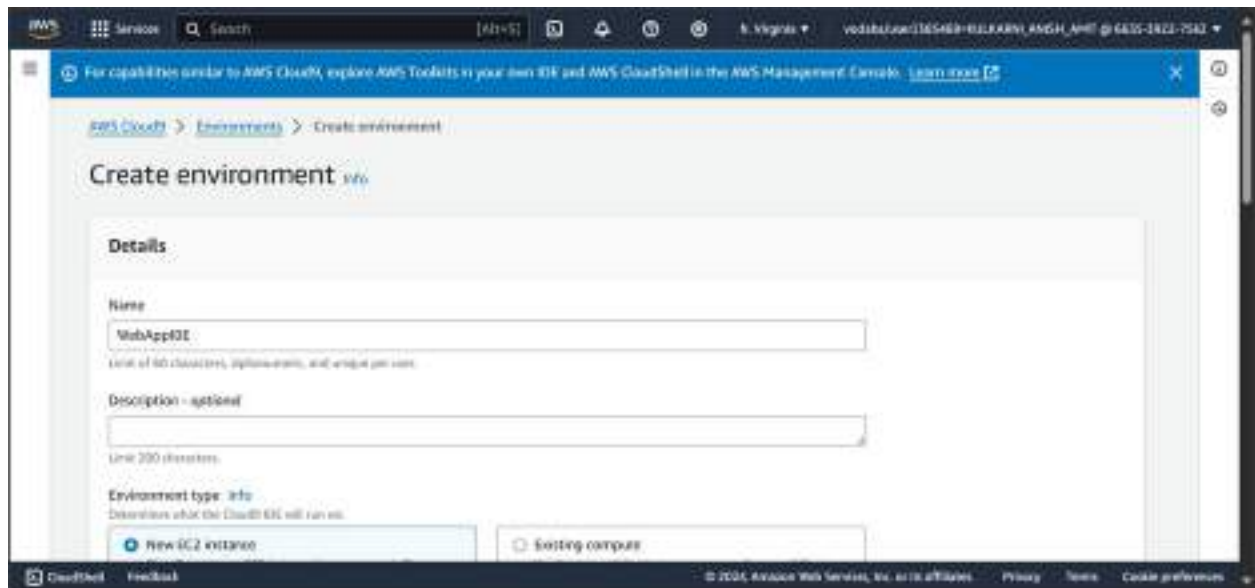
make the website accessible. This hands-on process illustrated the differences between local and cloud hosting, emphasizing the ease of deployment and configuration flexibility that each method offers.

Cloud9 setup, launch and collaboration [Exp 1(b)]

Aim: To understand the benefits of Cloud infrastructure and setup AWS Cloud9 IDE, launch AWS Cloud9 IDE and perform collaboration demonstration.

Steps:-

Step 1: Log into your AWS account and navigate to Cloud9 and click on 'Create environment' option. Give your environment a name.



Step 2: Keep all settings as default and click on Next.

New EC2 instance

Instance type [info](#)
The memory and CPU of the EC2 instance that will be created for Cloud9 to run on.

☒ **t2.micro** (1 GiB RAM + 1 vCPU)
Free-tier eligible, ideal for educational uses and exploration.

☐ t3.micro (2 GiB RAM + 2 vCPU)
Recommended for small web projects.

☐ m5.large (8 GiB RAM + 2 vCPU)
Recommended for production and most general-purpose development.

☐ **Additional instance types**
Explore additional instances to fit your need.

Platform [info](#)
This will be installed on your EC2 instance. We recommend Amazon Linux 2023.

Amazon Linux 2023

Timeout
How long Cloud9 can be inactive (no user input) before auto-terminating. This helps prevent unnecessary charges.

30 minutes

CloudShell Feedback

© 2024 Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Network settings [info](#)

Connection
How your environment is accessed.

☐ AWS Systems Manager (SSM)
Accesses environment via SSM without opening inbound ports (no internet).

☒ **Secure Shell (SSH)**
Accesses environment directly via SSH, opens inbound ports.

VPC settings [info](#)

Tags - optional [info](#)
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

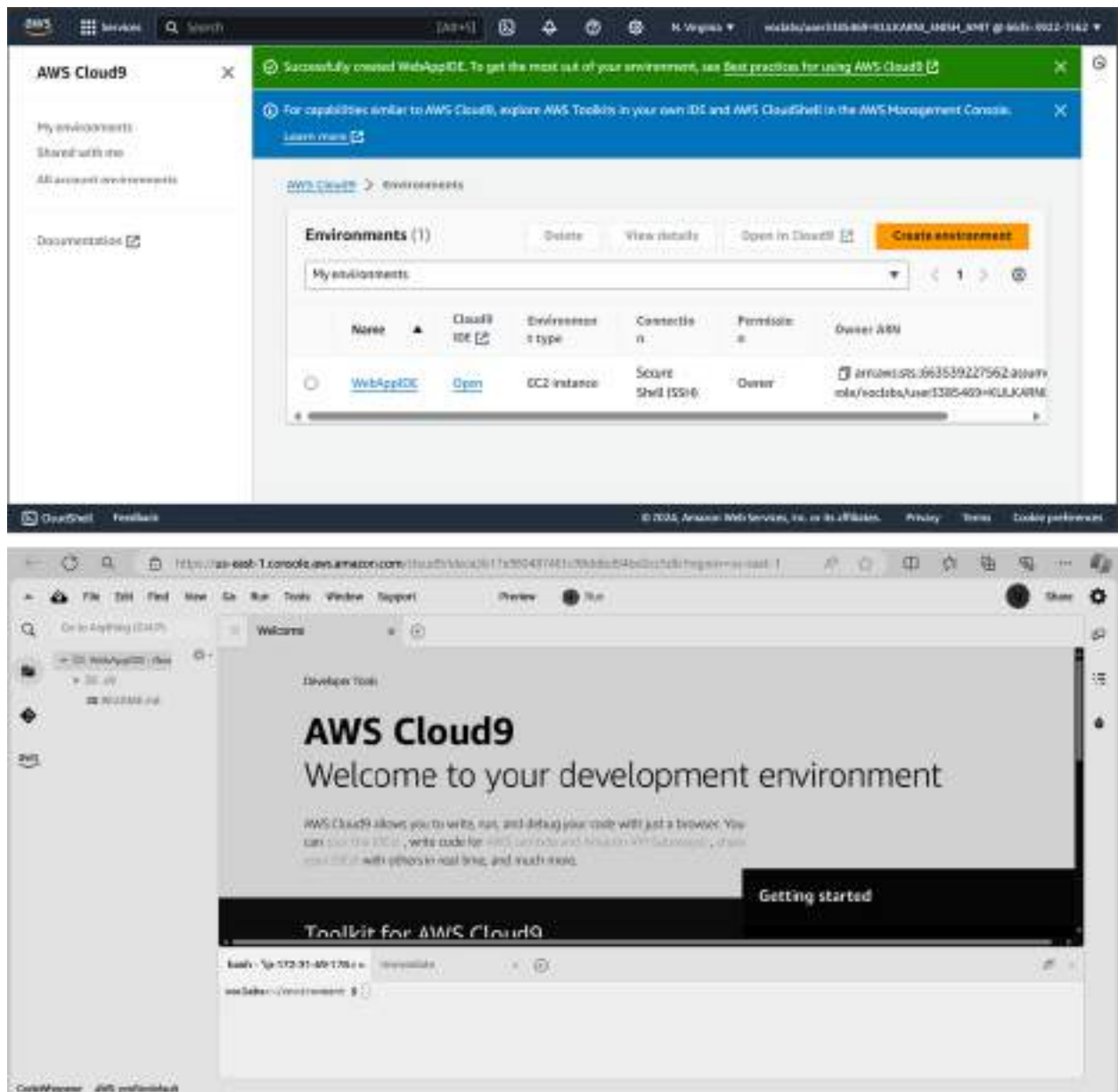
The following IAM resources will be created in your account:

- AWSServiceRoleForAWSCloud9** - AWS Cloud9 creates a service-linked role for you. This allows AWS Cloud9 to call other AWS services on your behalf. You can delete this role from the AWS IAM console once you no longer have any AWS Cloud9 environments. [Learn more](#)

CloudShell Feedback

© 2024 Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 3: Review your environment options and click on 'Create environment'. Your environment is created.



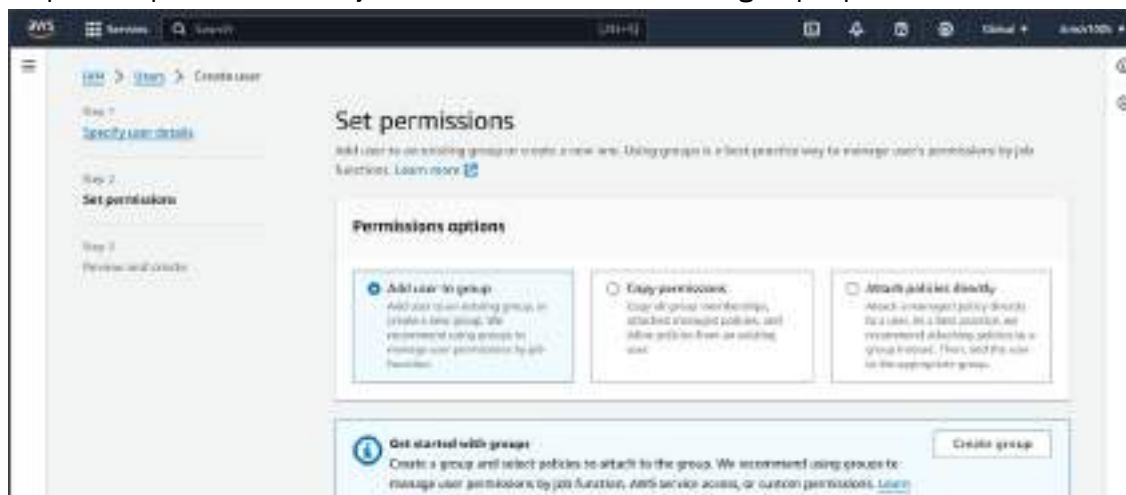
Step 4: Navigate to IAM (Identity and Access Management), click on 'Users' tab and click on 'Create User'.



Step 5: Give a name to your user and click on 'Next'.



Step 6: Set permissions for your user and click on 'Create group' option.



Step 7: Give a name to your user group and click on 'Create user group'.

Create user group

Create a user group and select policies to attach to the group. We recommend using groups to manage user permissions by job function, AWS service access, or custom permissions. [Learn more](#)

User group name
Enter a meaningful name to identify this group.

group1

Maximum 128 characters. Use alphanumeric and '+', '@', '-' characters.

Permissions policies (947)

Filter by Type
All ty... ▼

< 1 2 3 4 5 6 7 ... 48 > ⚙

<input type="checkbox"/>	Policy name	Type	Use...	Description
<input type="checkbox"/>	AdministratorAccess	AWS managed ...	None	Provides full access to AWS service
<input type="checkbox"/>	AdministratorAcce...	AWS managed	None	Grants account administrative perm

Cancel

Create user group

Step 8: Review your user options and click on 'Create user'.

Services and details

Step 1: Set permissions

Step 2: Review and create

REVIEW AND CREATE

Review your choices. After you create the user, you can view and download the welcome email (pending if enabled).

User details

User name	Choose password type	Require password reset
user1	None	No

Permissions summary

None selected

Name	Type	Used as
No resources		

Tags - optional

Tags are key-value pairs you can add to help organize resources, or search for resources. Choose any tags you want to associate with this user.

The tags associated with this resource are:

+ add new tag

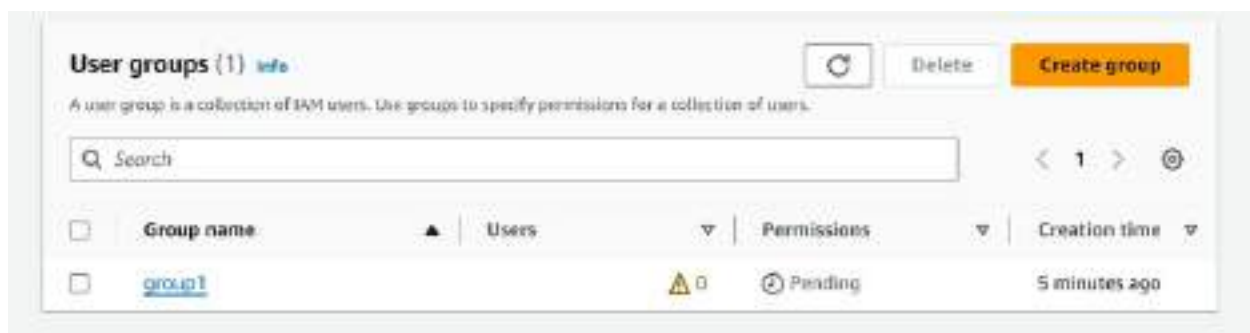
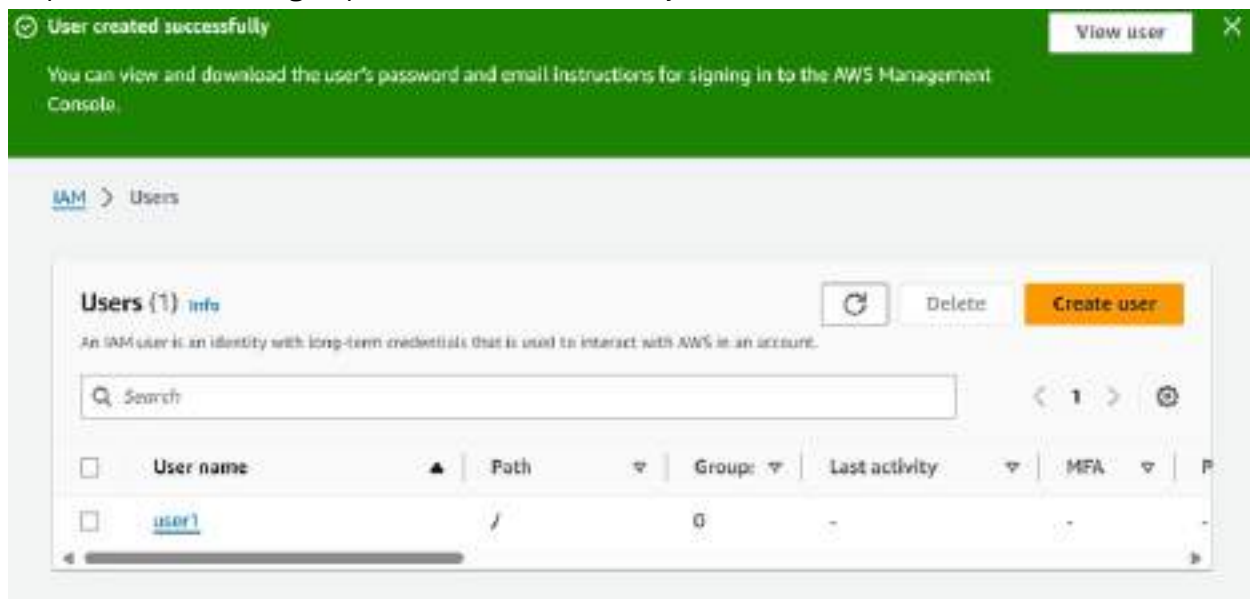
Maximum of 50 tags

Cancel

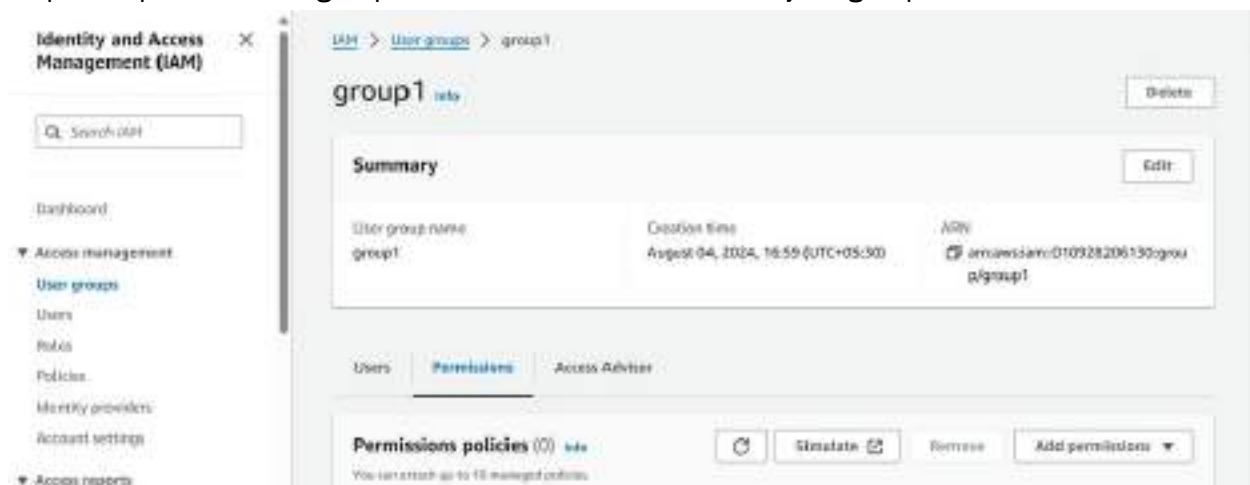
Previous

Create user

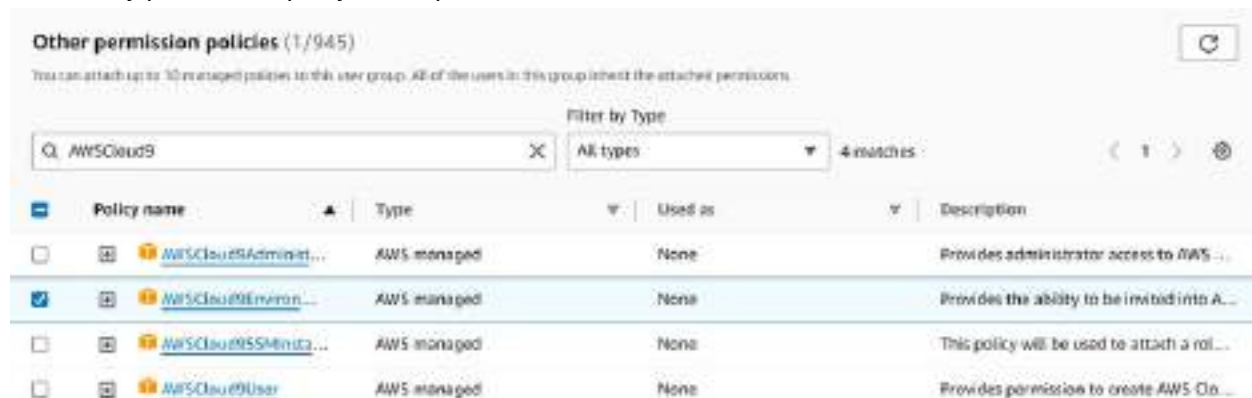
Step 9: User and User group are created successfully.



Step 10: Open the 'User groups' tab and click on the name of your group.

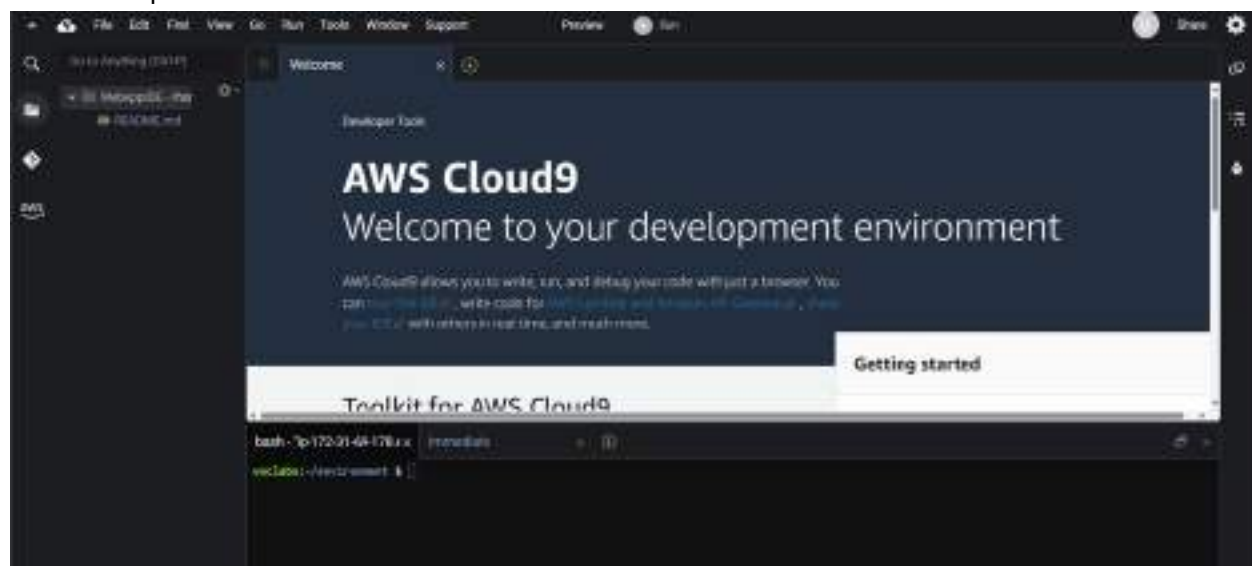


Step 11: Go to permissions and click on 'Add permissions'. Then, click on 'Attach policies' and attach any policies as per your requirement.

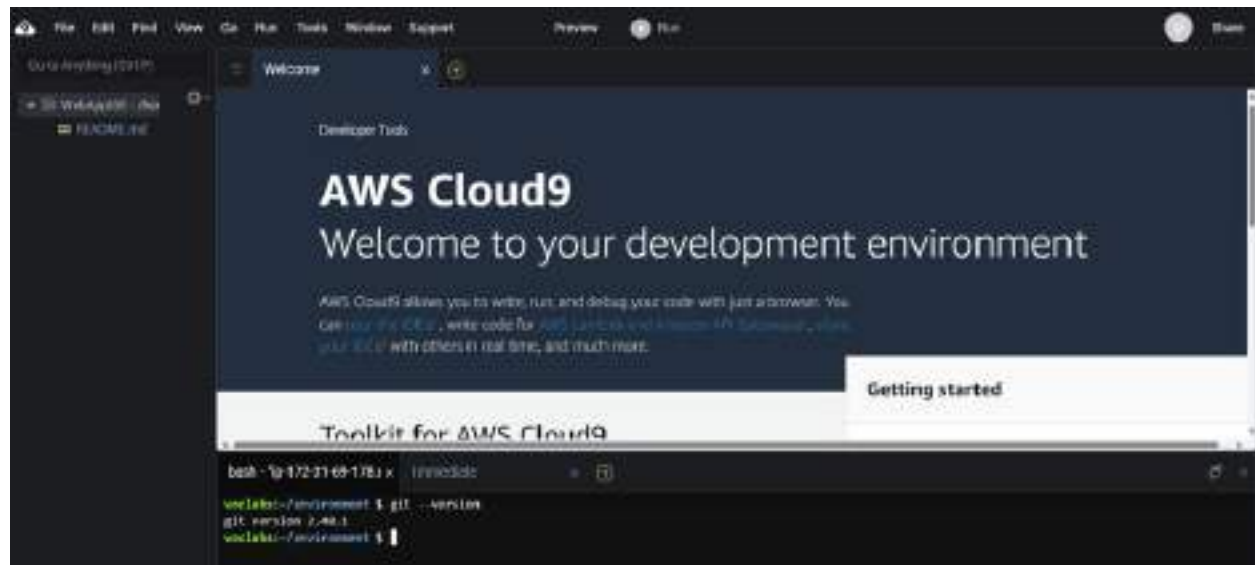


This attaches the policies to your user group.

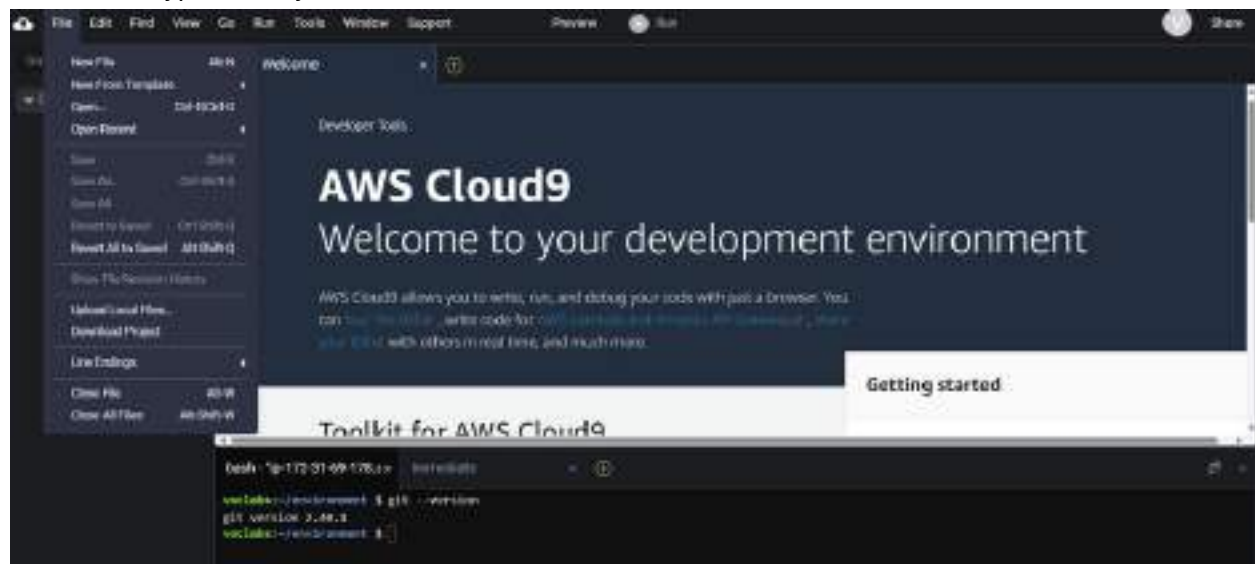
Step 12: To work on the Cloud9 IDE interface, enter commands into the command console which occupies the bottom one-third of the screen.



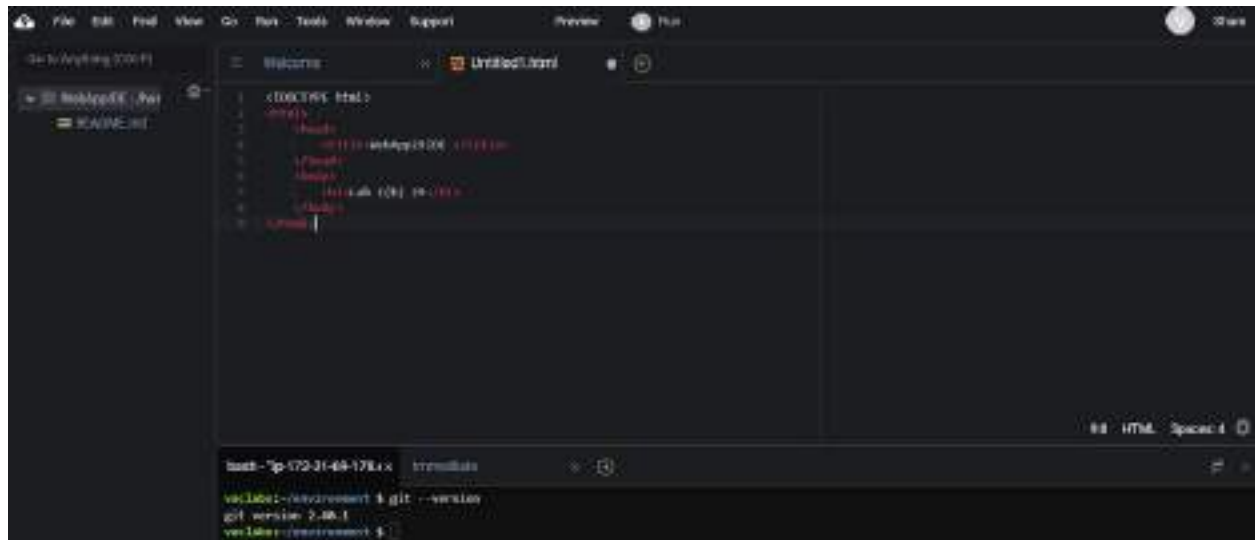
For example, entering the command 'git --version' produces the following output:-



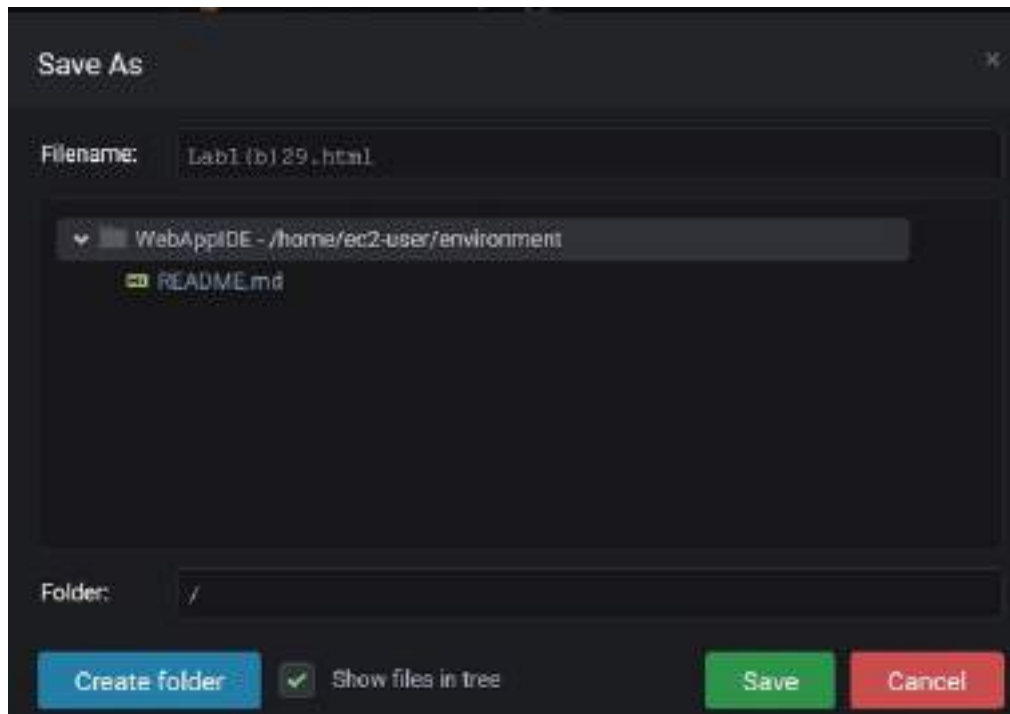
Step13: To add a file, click on 'File' in the top left corner, then click on 'New From Template' and choose the type of file you want to create.



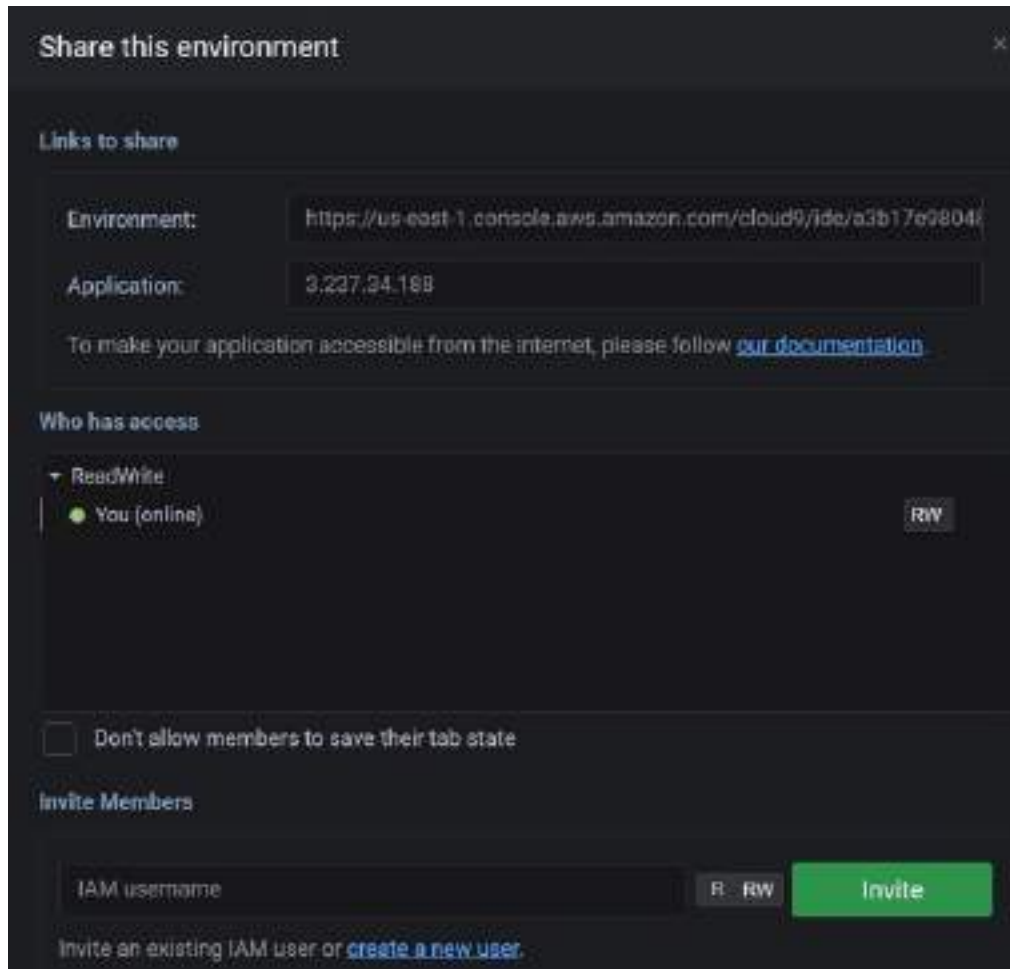
For this example, we create an HTML file. This gives a basic HTML code template on the coding IDE.



Step 14: After creating your file, click on 'File' in the top left corner and click on 'Save' from the drop-down menu.

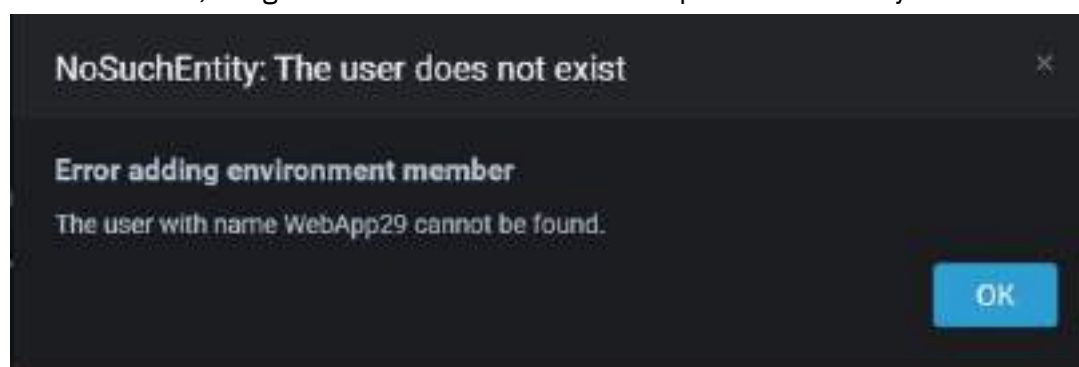


Step 15: After sharing, click on 'Share' in the top-right corner. Then, enter the username that you had entered during creating your IAM user.



The screenshot shows the 'Share this environment' dialog box. It has a title bar with a close button. Below the title is a section 'Links to share' containing two input fields: 'Environment' with the URL 'https://us-east-1.console.aws.amazon.com/cloud9/ide/a3b17e9804...' and 'Application' with the IP address '3.237.34.188'. Below these is a note: 'To make your application accessible from the internet, please follow [our documentation](#)'. The next section is 'Who has access', which shows a dropdown menu set to 'ReadWrite' and a list of users with 'You (online)' and a 'RW' role. Below this is a checkbox labeled 'Don't allow members to save their tab state'. The final section is 'Invite Members', which has an input field for 'IAM username', a dropdown for role set to 'R. RW', and a green 'Invite' button. At the bottom, it says 'Invite an existing IAM user or [create a new user](#)'.

Doing the above prompts the following error message to pop-up. This error occurs because while the above Cloud9 activities have been performed on AWS academy, it doesn't allow IAM users to be created on the academy account. On the other hand, while IAM users can be created on your personal account, Cloud9 services have been made inaccessible on personal accounts. Thus, integration of Cloud9 and IAM is not possible currently.



Conclusion: This experiment highlights how to set up and work with AWS Cloud9, although account limitations on AWS Academy prevent full collaboration demonstration. By understanding the steps and limitations, users can better leverage cloud infrastructure for development and teamwork when fully functional accounts are available.

Experiment No. 2

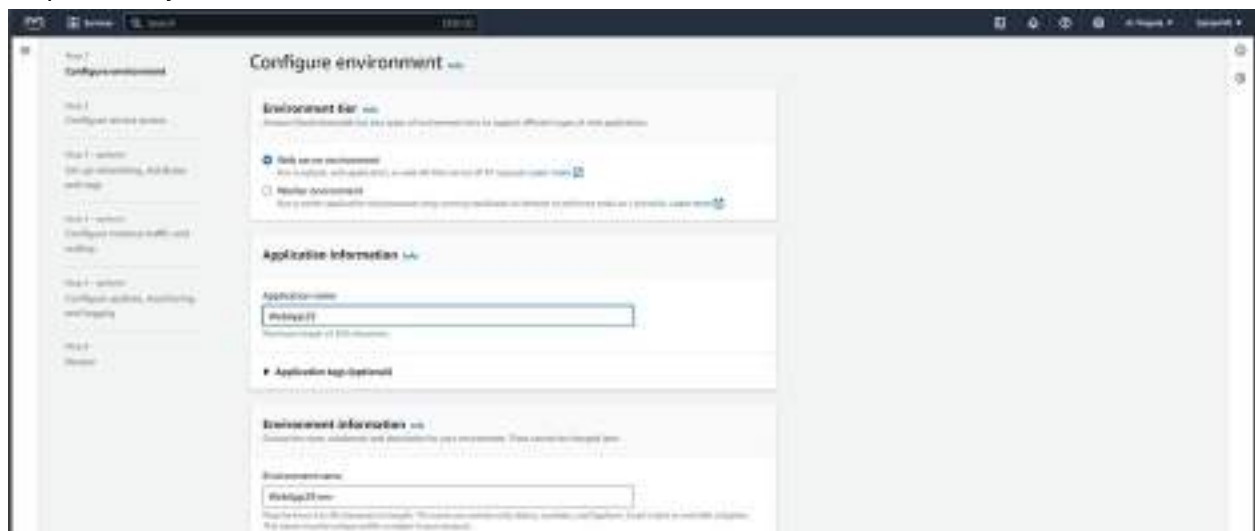
Aim: To build your application using AWS Codebuild and deploy on S3/SEBS using AWS CodePipeline, deploy a sample application on an EC2 instance using AWS CodeDeploy.

Steps:-

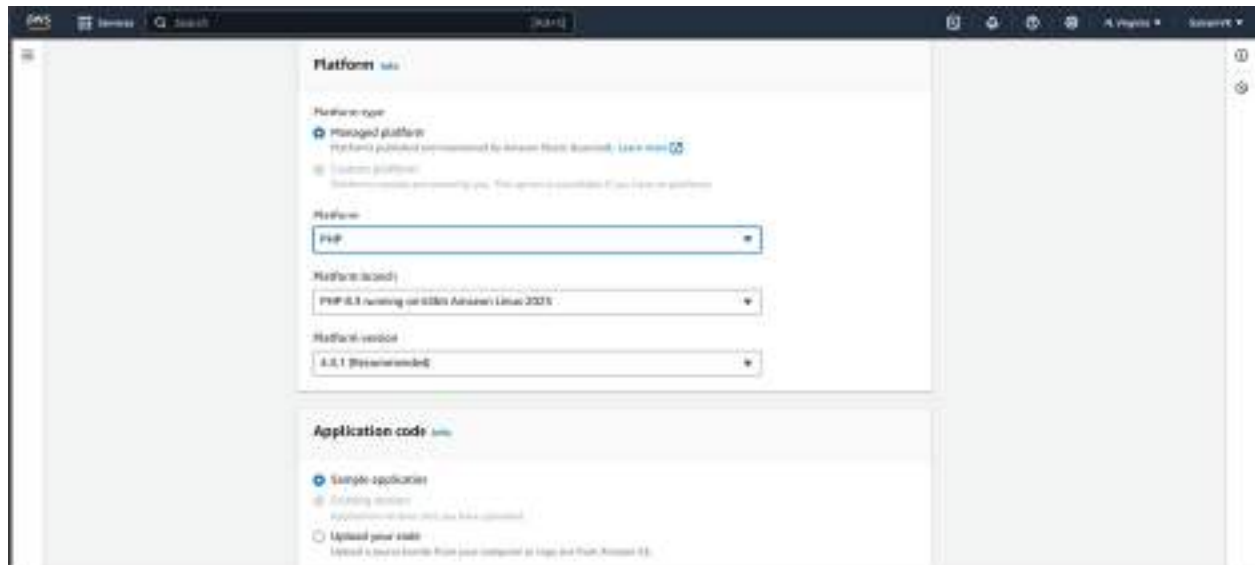
Step 1: Log into AWS and navigate to 'Elastic Beanstalk'. Then, click on 'Create application'.



Step 2: Give your environment a name.



Step 3: In the 'Platform' drop-down box, choose PHP. Keep all other settings as default and click on 'Next'.



The screenshot shows the 'Platform' configuration page in the AWS SageMaker console. The 'Platform type' is set to 'Managed platform'. The 'Platform' dropdown is set to 'PHP'. The 'Platform (legacy)' dropdown is set to 'PHP 8.1 running on Amazon Linux 2019'. The 'Platform version' dropdown is set to '8.1.1 (Recommended)'. The 'Application code' section shows 'Sample applications' as the selected option.

Platform [Info](#)

Platform type

- ☒ **Managed platform**
Managed platform is managed by Amazon SageMaker. [Learn more](#)
- ☐ Custom platform
Custom platform is managed by you. This option is available if you have a custom platform.

Platform

PHP

Platform (legacy)

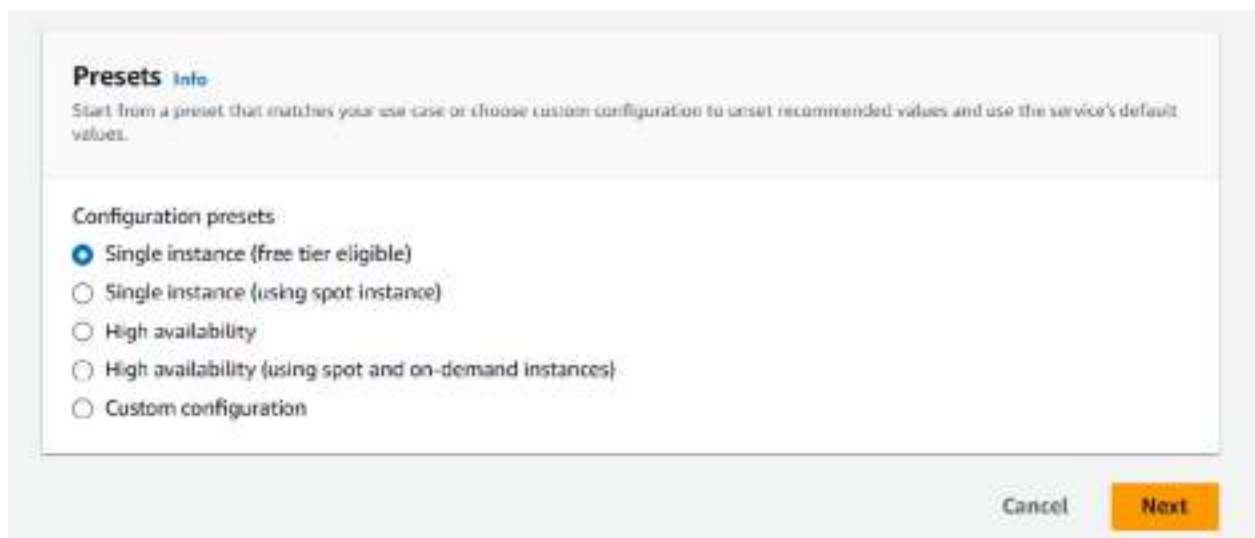
PHP 8.1 running on Amazon Linux 2019

Platform version

8.1.1 (Recommended)

Application code [Info](#)

- ☒ **Sample applications**
Sample applications are pre-built applications that you can use to get started with SageMaker. [Learn more](#)
- ☐ Upload your code
Upload your code to SageMaker. [Learn more](#)



The screenshot shows the 'Presets' configuration page in the AWS SageMaker console. The 'Configuration presets' section has five options: 'Single instance (free tier eligible)' (selected), 'Single instance (using spot instance)', 'High availability', 'High availability (using spot and on-demand instances)', and 'Custom configuration'.

Presets [Info](#)

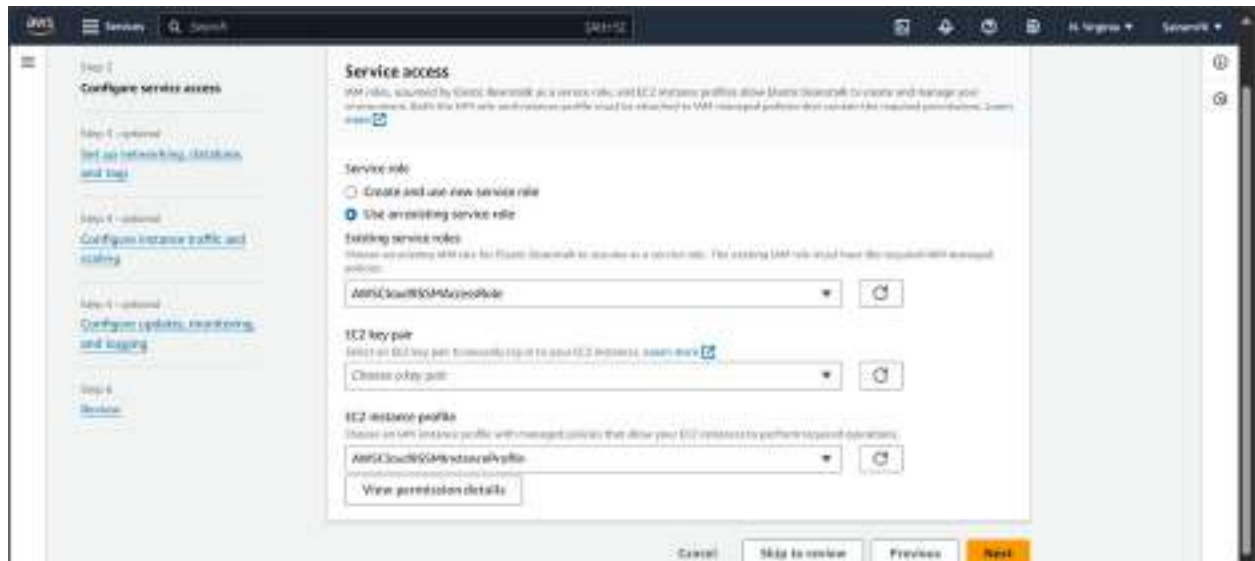
Start from a preset that matches your use case or choose custom configuration to unset recommended values and use the service's default values.

Configuration presets

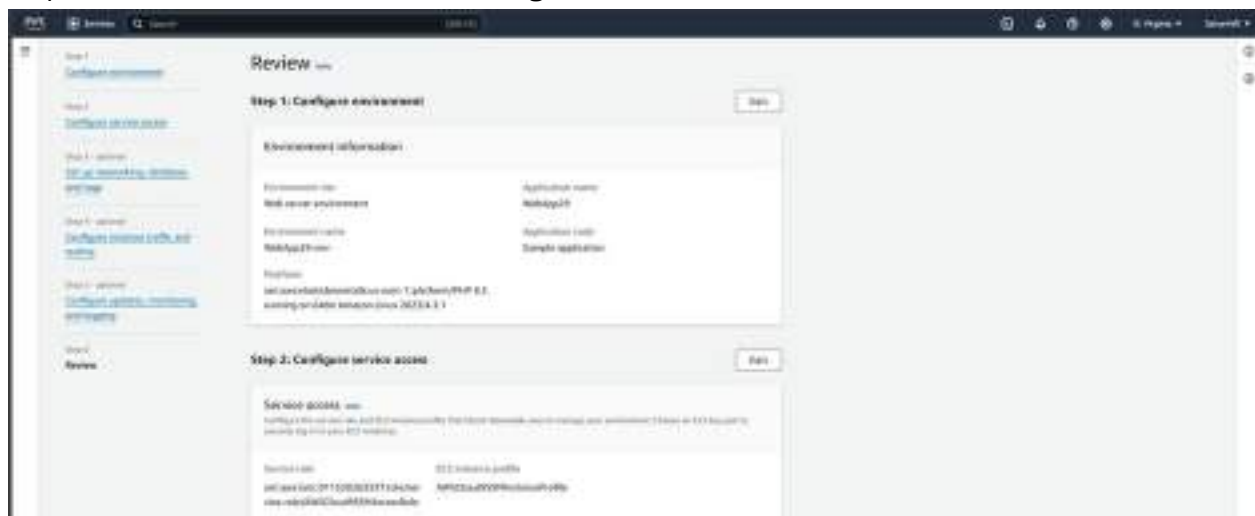
- ☒ **Single instance (free tier eligible)**
- ☐ Single instance (using spot instance)
- ☐ High availability
- ☐ High availability (using spot and on-demand instances)
- ☐ Custom configuration

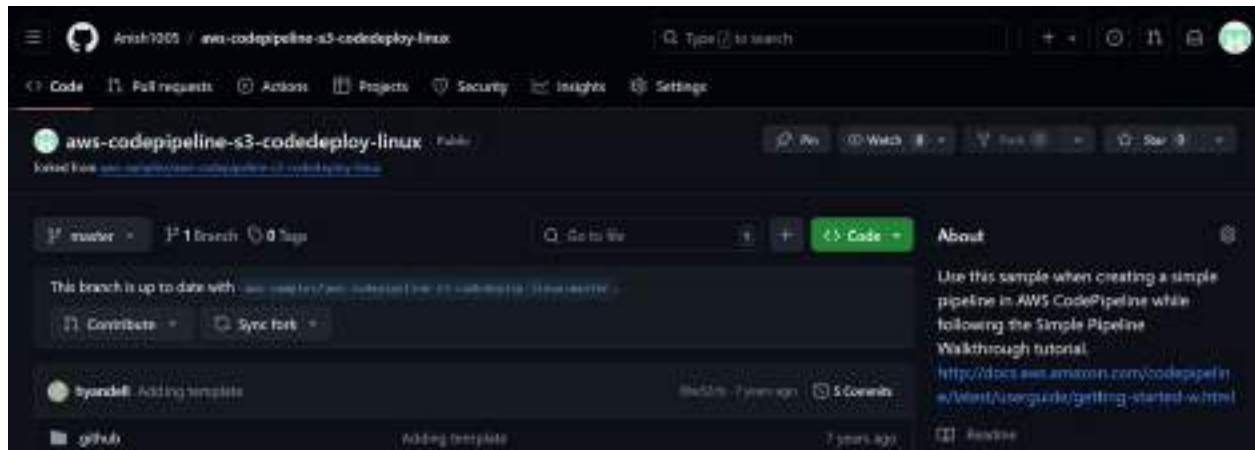
[Cancel](#) [Next](#)

Step 4: In the 'Service access' section, choose 'AWSCloud9AAMAccessRole' in 'Existing service roles' and 'AWSCloud9SSMInstanceProfile' in 'EC2 instance profile'.

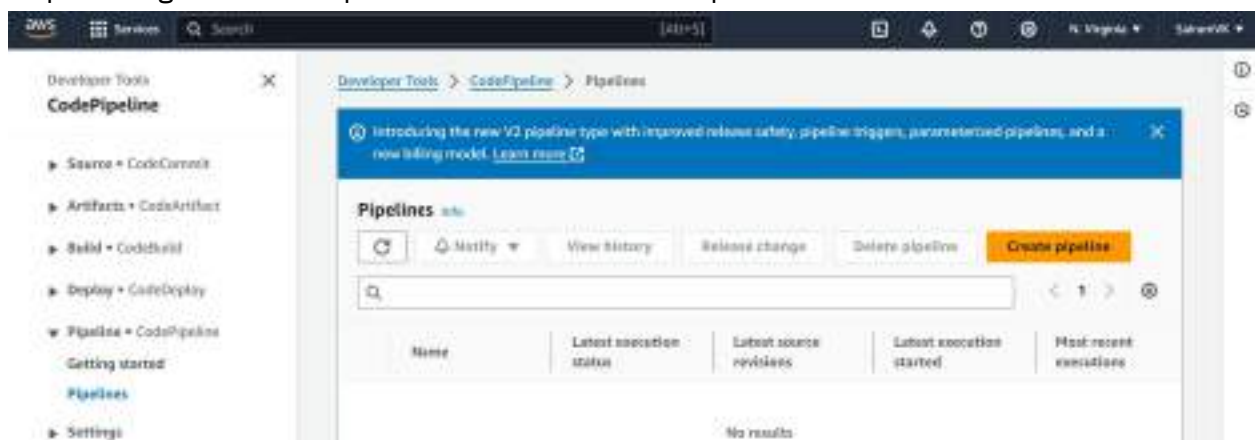


Step 5: Review all the environment settings and click on 'Submit'.

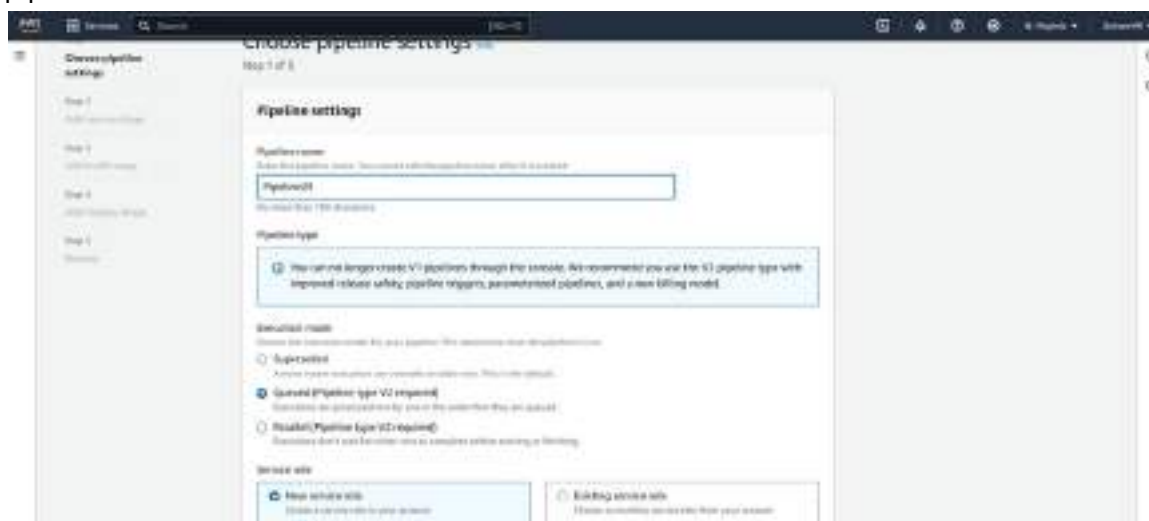




Step 7: Navigate to CodePipeline and click on 'Create Pipeline'.

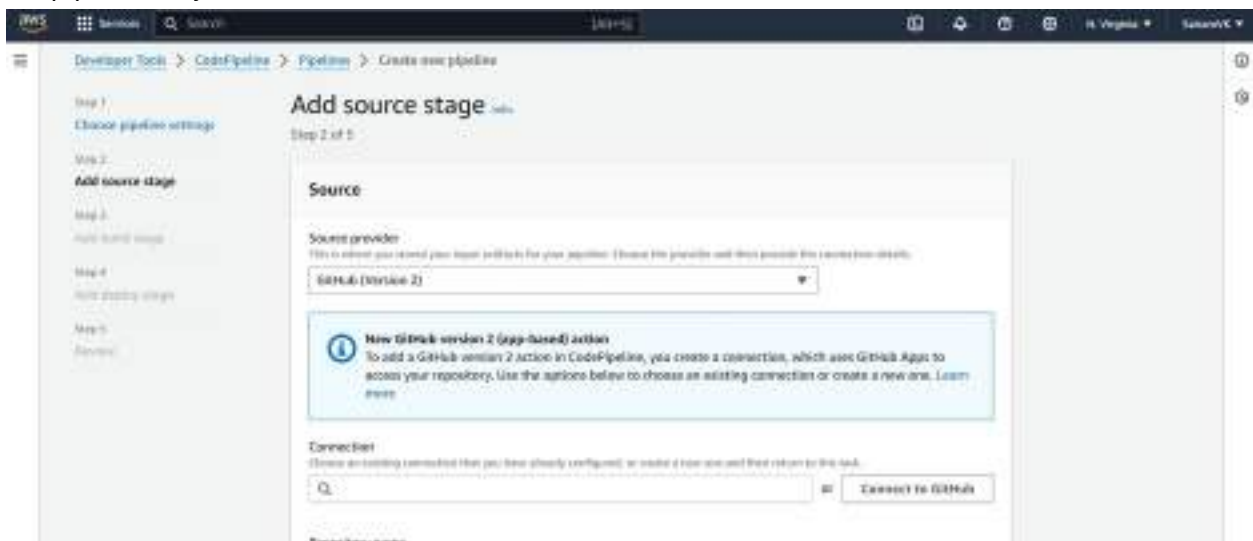


Step 8: Give your pipeline a name. A new service role is also created with the name of the pipeline.





Step 9: Select Github (Version 2) as source provider and click on 'Connect to Github' to connect the pipeline to your Github.

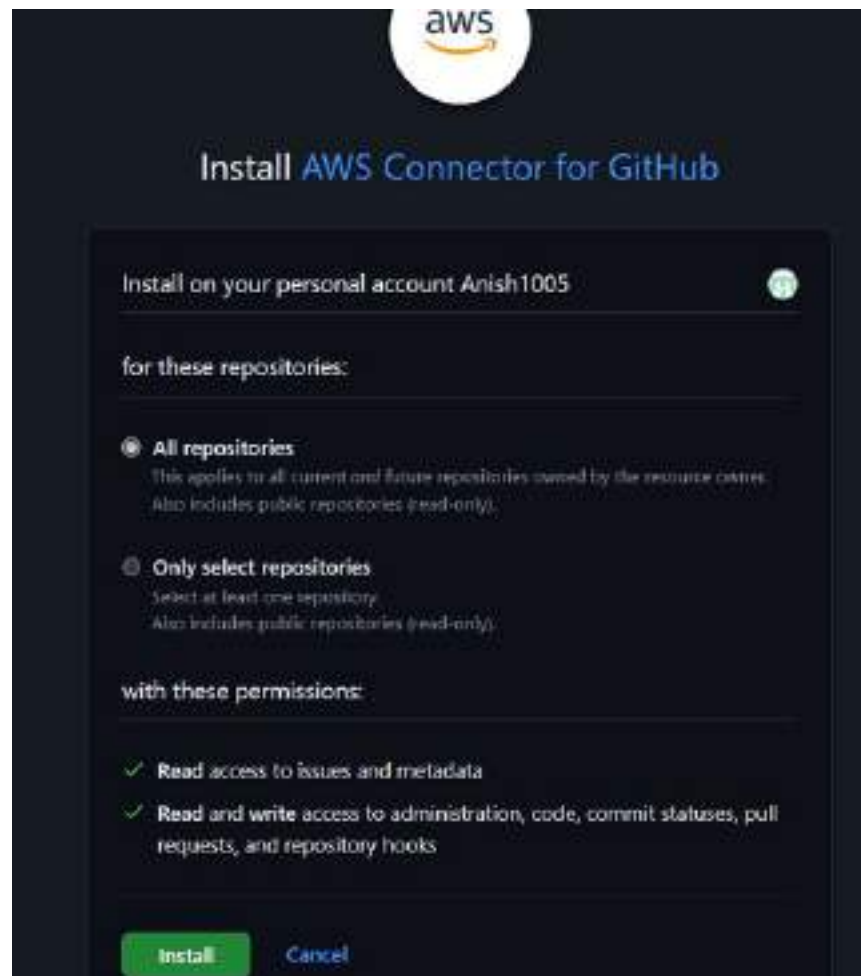


Step 10: Give your connection a name and click on 'Connect to Github'. Then, either provide a link for connection or install the app if it is your first time.

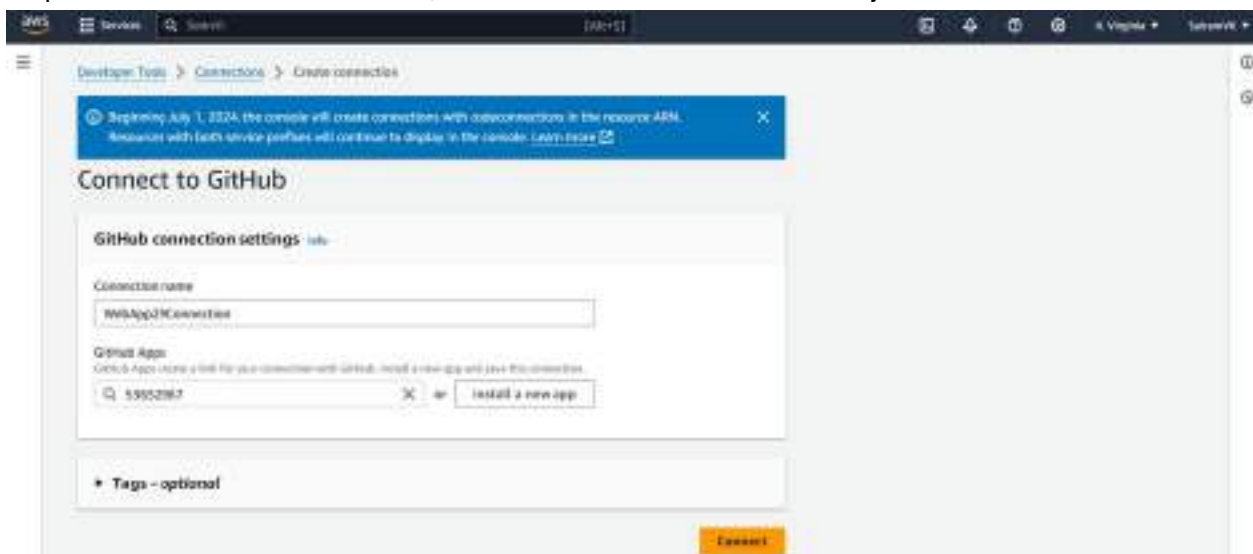
This screenshot shows the 'Create a connection' page in the AWS IAM console. The breadcrumb trail is 'Developer Tools > Connections > Create connection'. The page title is 'Create a connection'. Below this, there is a section titled 'Create GitHub App connection'. It contains a text input field for 'Connection name' with the value 'WebApp20Connection'. Below the input field is a section for 'Tags - optional' with a plus icon and a text input field. At the bottom right, there is an orange button labeled 'Connect to GitHub'.

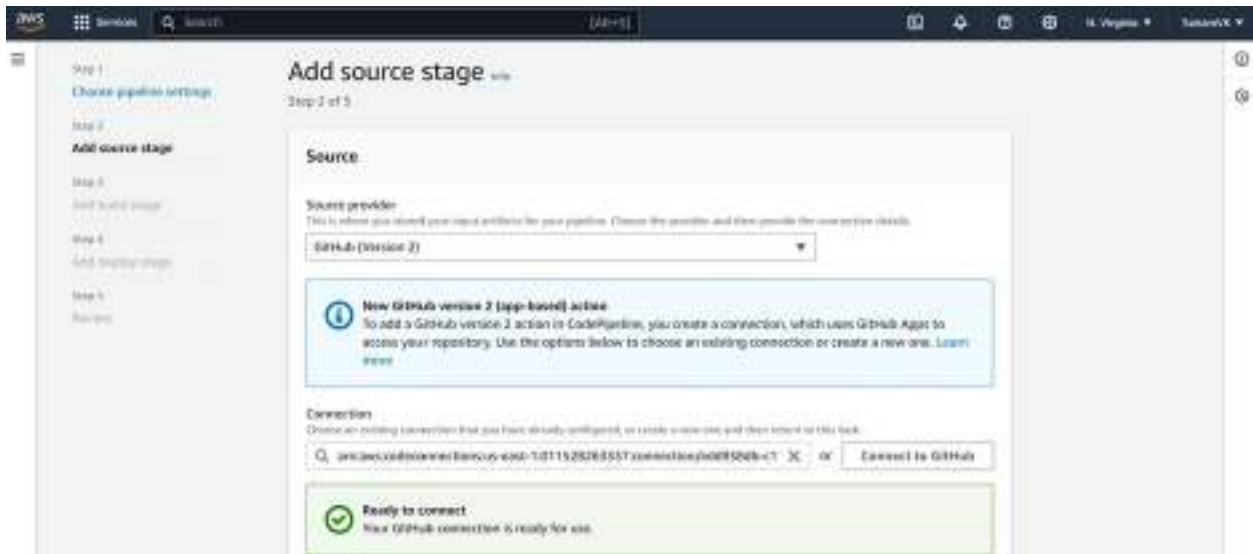
This screenshot shows the 'Connect to GitHub' page in the AWS IAM console. The breadcrumb trail is 'Developer Tools > Connections > Create connection'. At the top, there is a blue banner with a warning icon and text: 'Beginning July 1, 2022, if a console will create connections with GitHub connections in the console API. Users can with both console profiles will continue to display in the console. Learn more'. Below the banner, the page title is 'Connect to GitHub'. There is a section titled 'GitHub connection settings' which contains a text input field for 'Connection name' with the value 'WebApp20Connection'. Below this, there is a section for 'GitHub App' with a text input field for 'GitHub App ID' and a button labeled 'Install a new app'. At the bottom right, there is an orange button labeled 'Connect'.

Step 11: Install AWS Connector for Github.



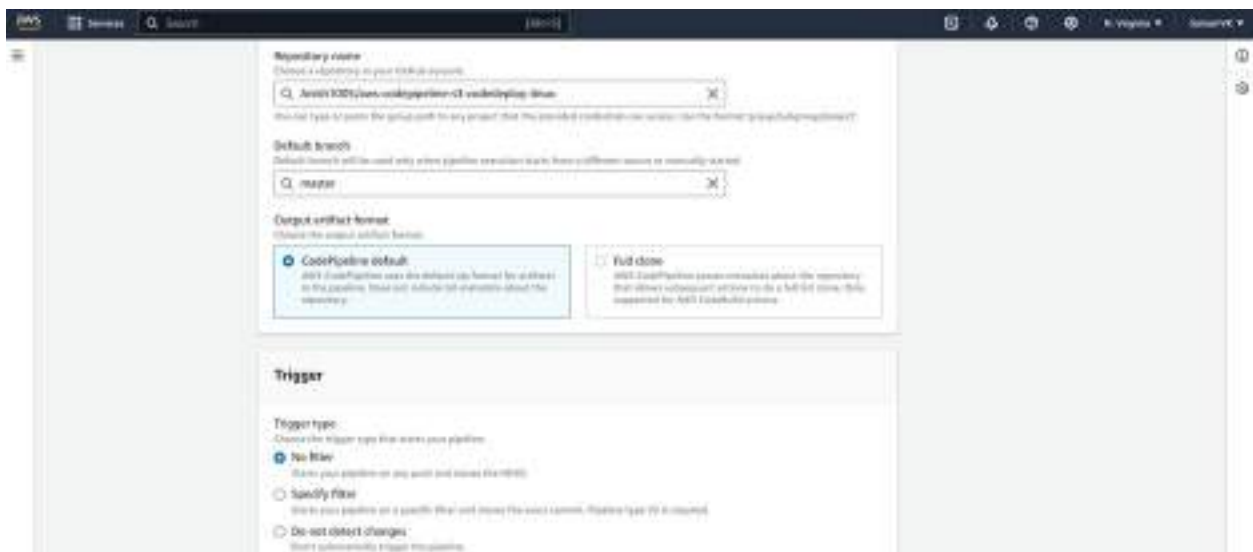
Step 12: Once installation is done, the text field is filled automatically. Click on 'Connect'.





AWS shows that the Github connection is ready to use.

Step 13: Select the repository that you forked the sample code to be deployed to and choose the branch on which the files are present ('master' is set as default). Also set the trigger type as 'no filter'.



Step 14: Skip the build stage and go to deploy stage.



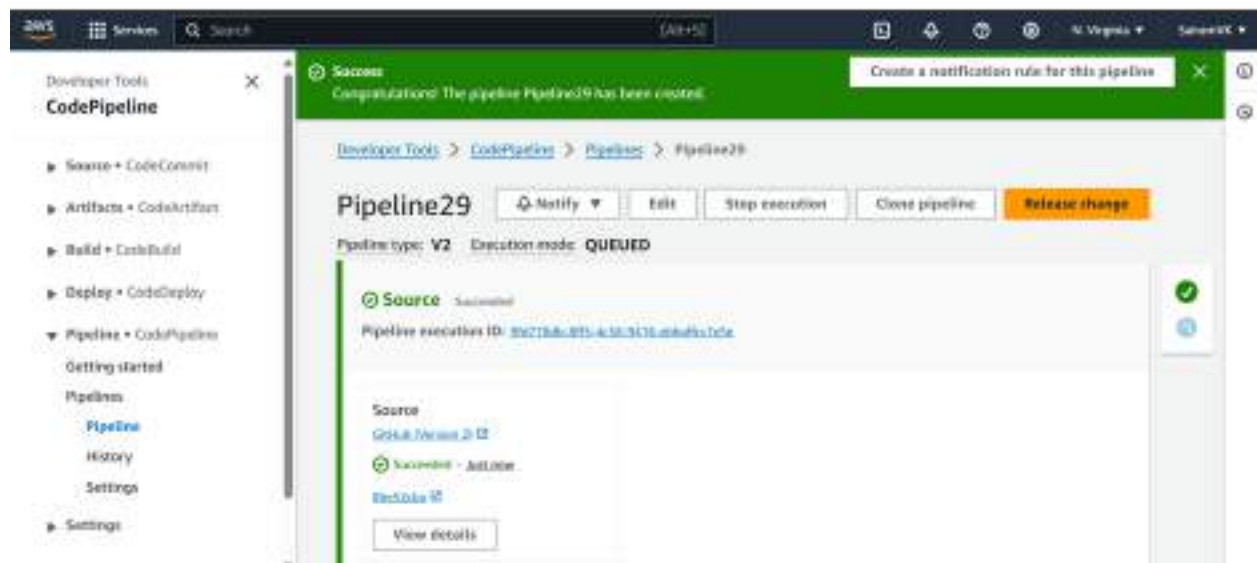
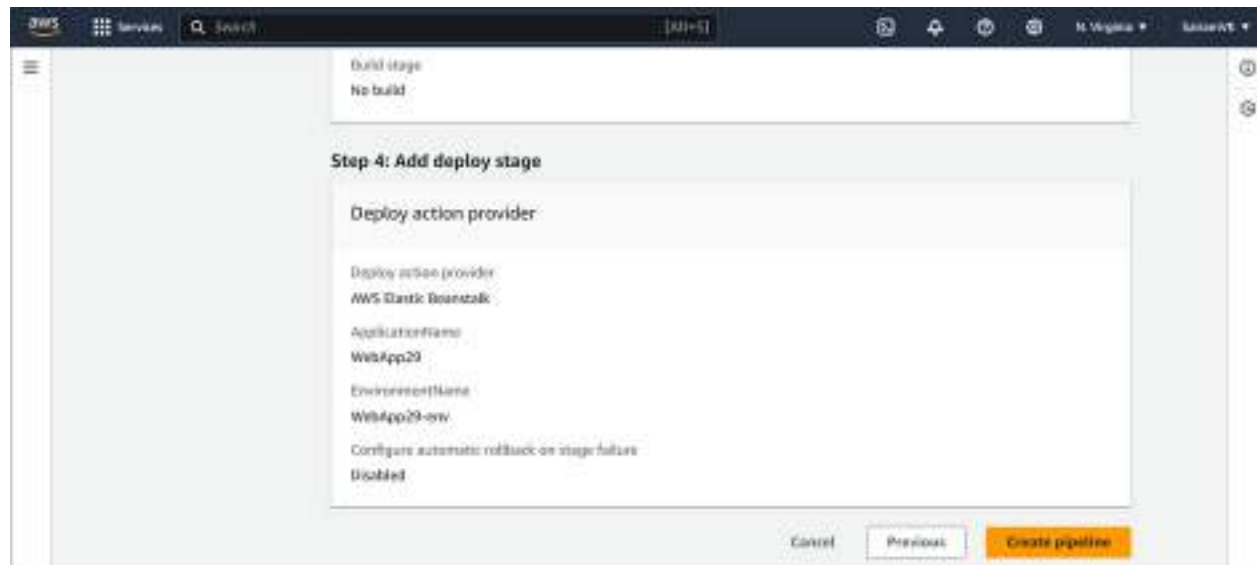
The screenshot shows the AWS CodePipeline console interface. The breadcrumb navigation at the top reads: **Developer Tools** > **CodePipeline** > **Pipelines** > **Create new pipeline**. On the left, a sidebar lists the steps of the pipeline: **Step 1: Choose pipeline settings**, **Step 2: Add source stage**, **Step 3: Add build stage** (which is the current step), **Step 4: Add deploy stage**, and **Step 5: Review**. The main content area is titled **Add build stage** with a subtitle **Step 3 of 5**. Inside this area, there is a section titled **Build - optional** with a sub-section **Build provider** and a description: "This is the root of your build project. Provide build artifact details like operating system, build spec file, and output file names." Below this is a dropdown menu. At the bottom right of the main content area, there are four buttons: **Cancel**, **Previous**, **Skip build stage**, and **Next** (which is highlighted in orange).

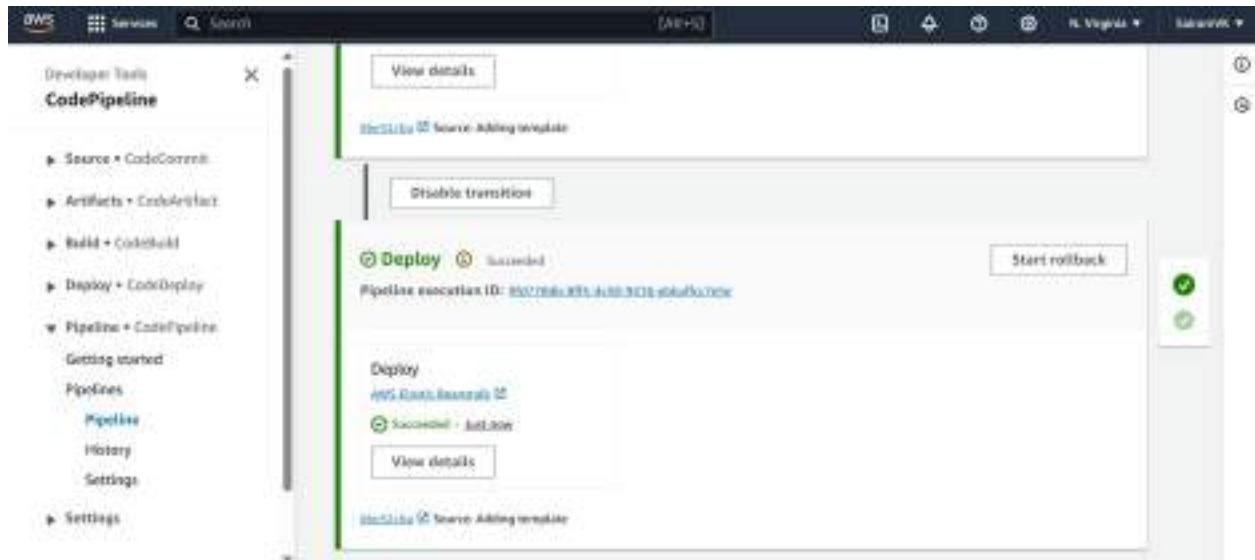
Step 15: Choose 'AWS Elastic Beanstalk' as deploy provider and input artifacts as 'SourceArtifact'. Then, enter the names of your application and environment.



The screenshot shows the AWS CodePipeline console interface for the **Add deploy stage** step. The breadcrumb navigation at the top is: **Developer Tools** > **CodePipeline** > **Pipelines** > **Create new pipeline**. The left sidebar shows the pipeline steps: **Step 1: Choose pipeline settings**, **Step 2: Add source stage**, **Step 3: Add build stage**, and **Step 4: Add deploy stage** (which is the current step). The main content area is titled **Add deploy stage** with a subtitle **Step 4 of 5**. The main section is titled **Deploy** and contains several fields: **Deploy provider** (with a description: "Choose how your deploy is performed. Choose this provider, and then provide the configuration details for this provider.") set to **AWS Elastic Beanstalk**, **Region** set to **US East (N. Virginia)**, **Input artifacts** (with a description: "Choose an input artifact for this action. Learn more.") set to **SourceArtifact**, **Application name** (with a description: "Choose an application that you have directly connected to the AWS Elastic Beanstalk console. Or create an application in the AWS Elastic Beanstalk console and then return to this page.") set to **CL-WebApp01**, and **Environment name** (with a description: "Choose an environment that you have already created in the AWS Elastic Beanstalk console. Or create an environment in the AWS Elastic Beanstalk console and then return to this page.") set to **CL-WebApp01-env**. There is also an unchecked checkbox labeled **Configure automatic rollback on stage failure**. At the bottom right, there are three buttons: **Cancel**, **Previous**, and **Next** (which is highlighted in orange).

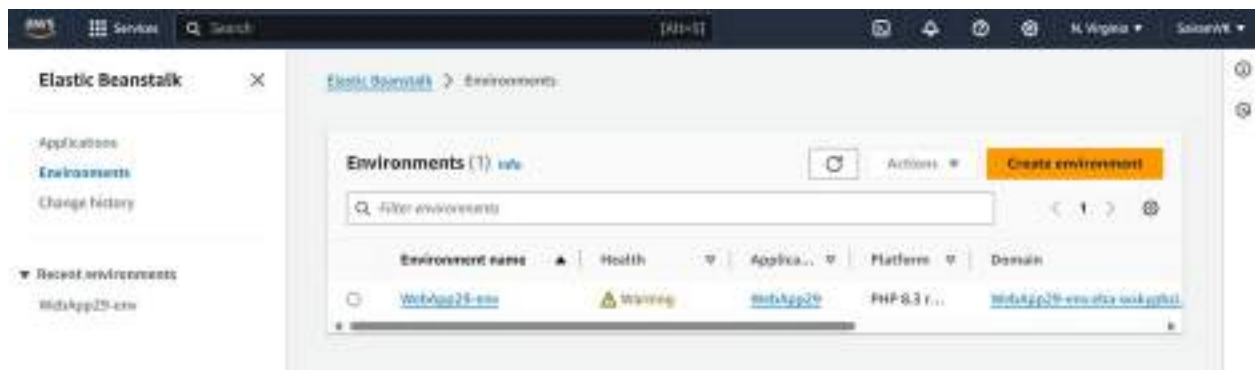
Step 16: Review all pipeline information and click on 'Create pipeline'.



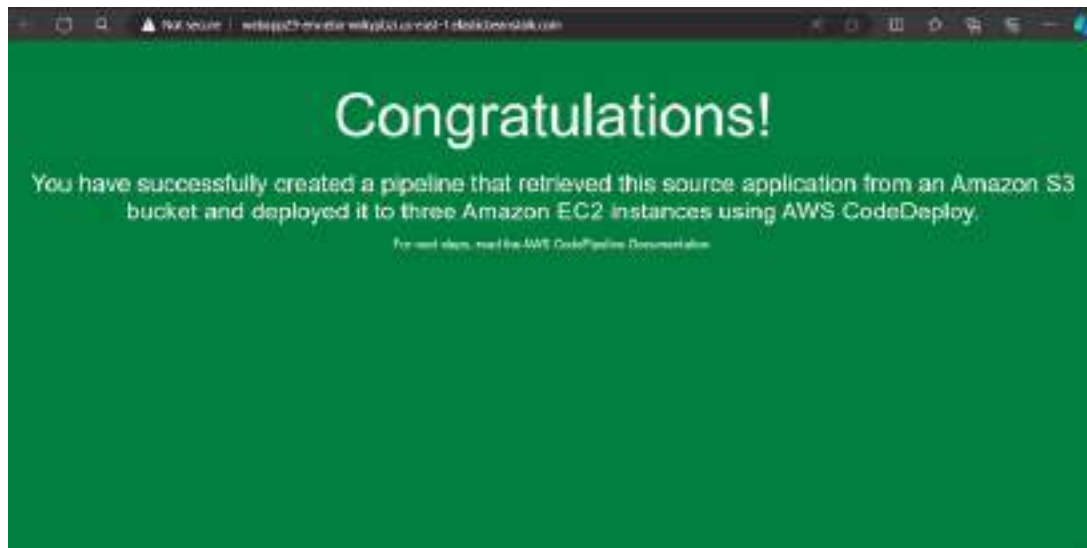


The pipeline is created and deployment is complete.

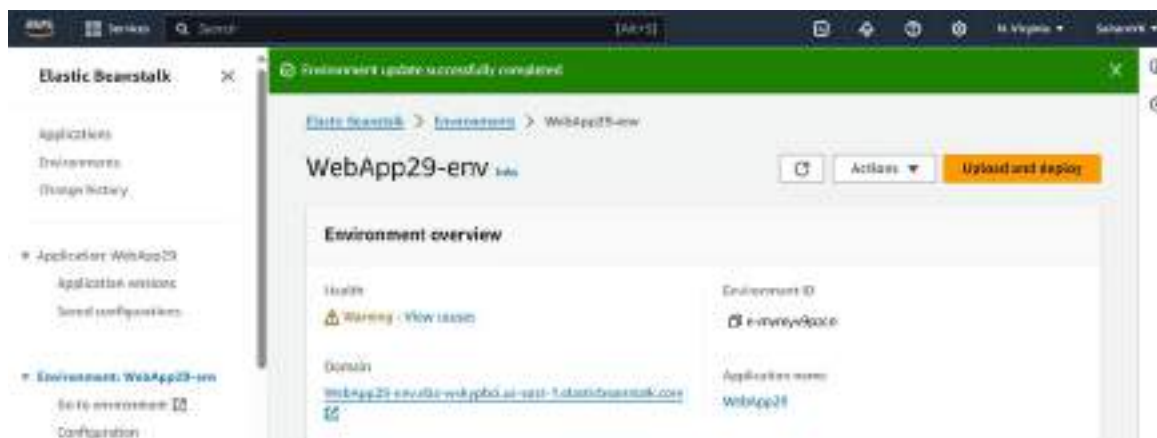
Step 17: Clicking on 'AWS Elastic Beanstalk' under 'Deploy' redirects you to the environments screen of Elastic Beanstalk. Click on the link under 'Domain'.



Step 18: The sample website is displayed showing that the website was successfully hosted.



Step 19: Making changes to the index file in the github will update the website and changes made to the website are visible.



Conclusion: This process showcases the ability to automate the entire lifecycle of building, deploying, and managing updates for an application using AWS services. By utilizing AWS CodeBuild, AWS CodePipeline, and Elastic Beanstalk, you streamline the deployment of a sample application and ensure continuous delivery. Additionally, by integrating GitHub, any changes to the source code are automatically deployed, making the process efficient and scalable. This approach simplifies application management, reduces manual intervention, and enhances collaboration, making it an ideal choice for modern cloud-based software development.

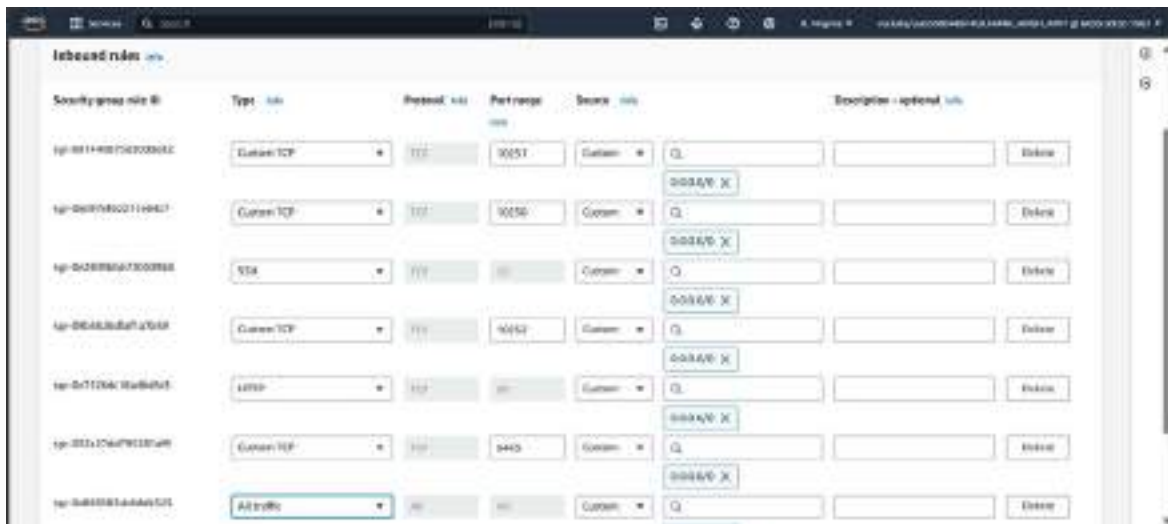
Experiment 3

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

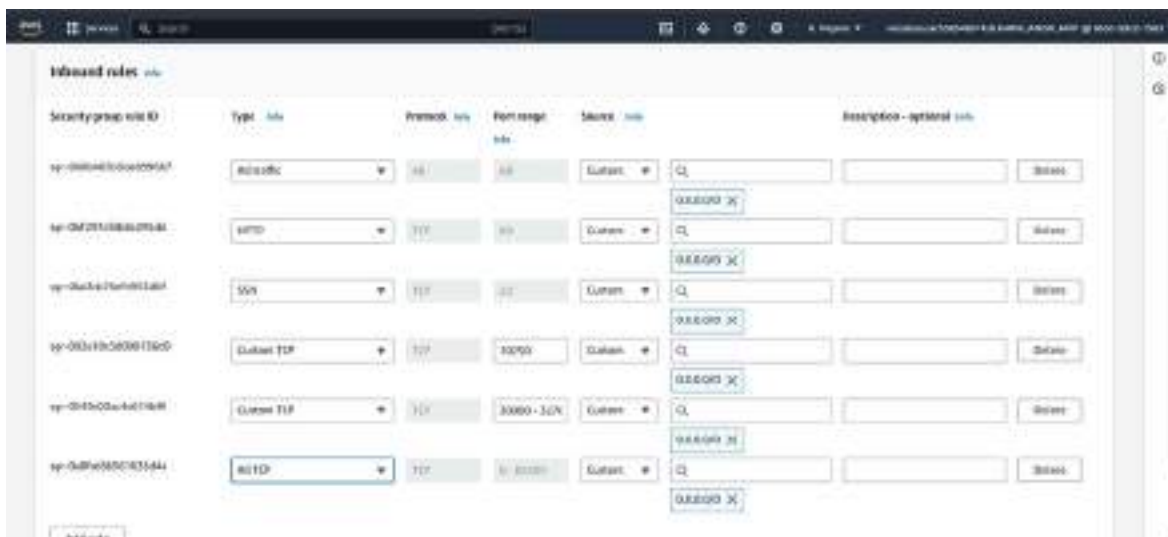
Steps:

Step 1: Create separate security groups for the master and worker nodes and add the following inbound rules. To do so, click on 'Security groups' in the left sidebar and click on 'Edit inbound rules'.

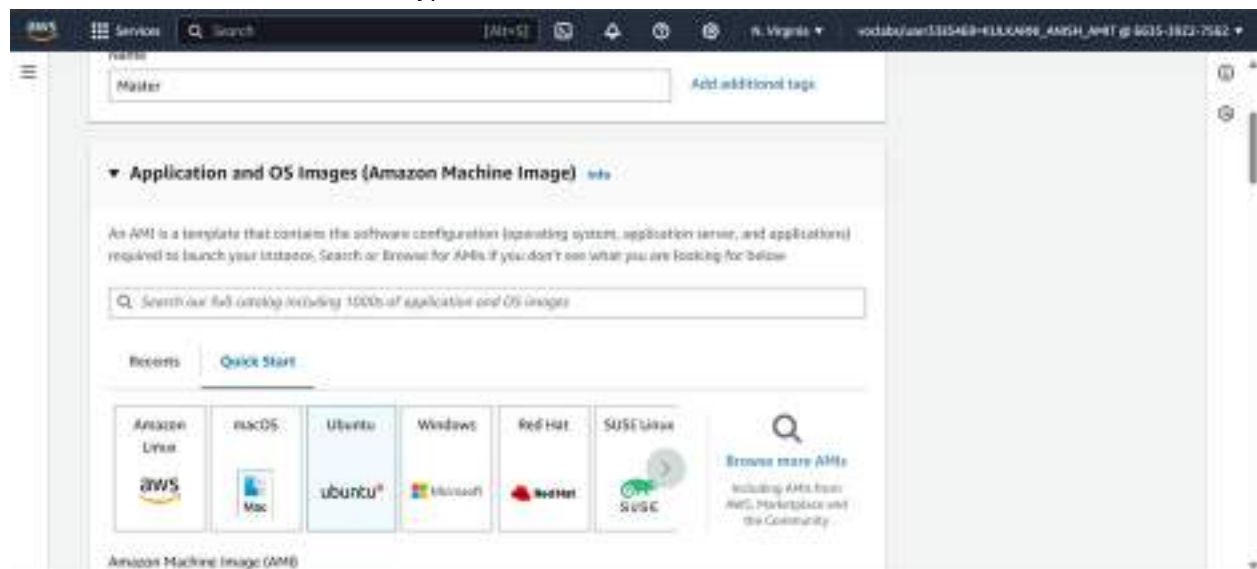
● Master:-



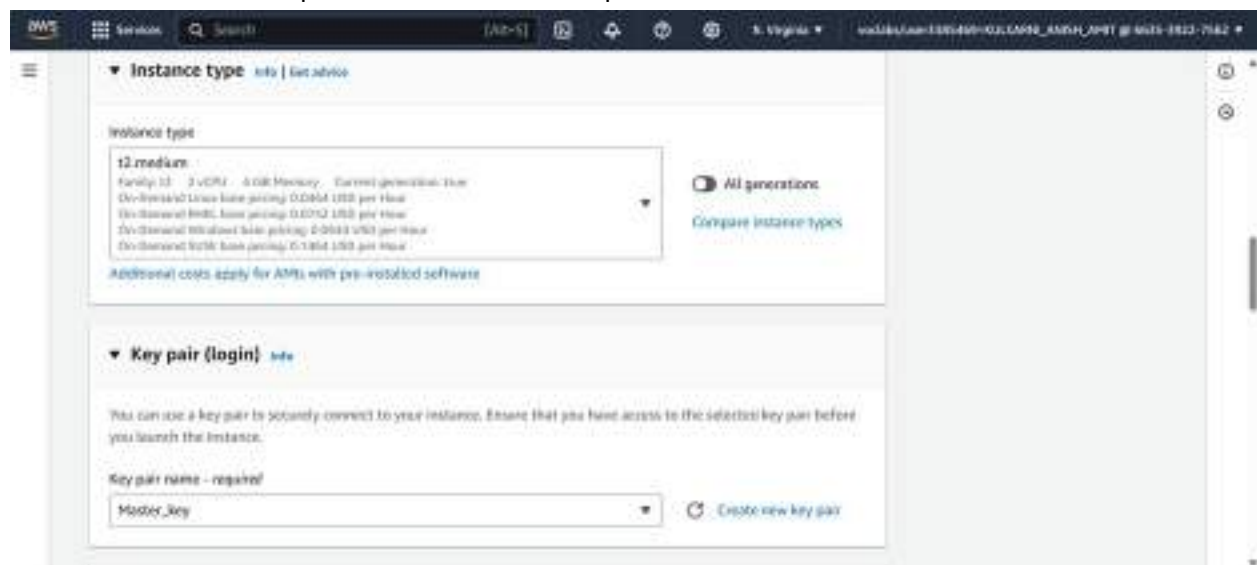
● Nodes:-



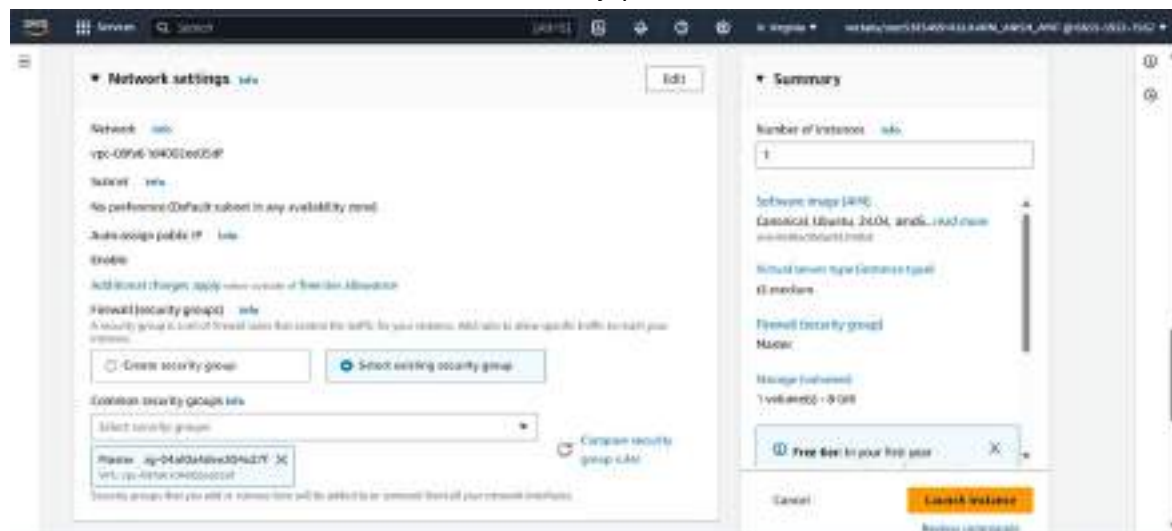
Step 2: Create 3 new EC2 instances in AWS (1 master and 2 nodes). Make sure that you choose Ubuntu as the instance type.



Under 'Instance type', choose t2.medium because the default (t2.micro) does not provide the sufficient resources required to execute this experiment.

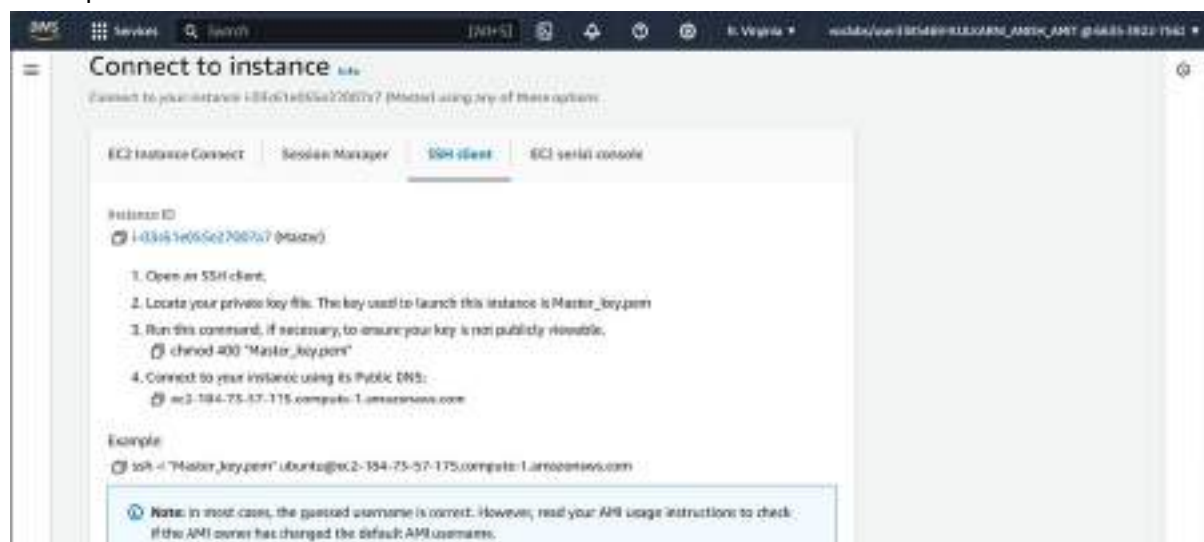


Create 2 separate key pairs (one for the master and one for the other 2 nodes). Under 'Network settings', click on 'Select existing security group and for the master, choose the Master key-pair and for the other 2 nodes, choose the Node key-pair.



All the instances are created as above.

Step 3: Now, we must connect each instance to a SSH. To do so, click on an instance and click on 'Connect'. Next, navigate to the 'SSH client' section and copy the command under the 'Example' section.



Connect to instance [info](#)

Connect to your instance `i-09d921009056d3876` (Node1) using any of these options:

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

Instance ID
`i-09d921009056d3876` (Node1)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is `Node_key.pem`.
3. Run this command, if necessary, to ensure your key is not publicly viewable.
`chmod 400 "Node_key.pem"`
4. Connect to your instance using its Public DNS:
`ec2-54-87-81-118.compute-1.amazonaws.com`

Example:
`ssh -i "Node_key.pem" ubuntu@ec2-54-87-81-118.compute-1.amazonaws.com`

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

Connect to instance [info](#)

Connect to your instance `i-04e88cd275b9f8415` (Node2) using any of these options:

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

Instance ID
`i-04e88cd275b9f8415` (Node2)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is `Node_key.pem`.
3. Run this command, if necessary, to ensure your key is not publicly viewable.
`chmod 400 "Node_key.pem"`
4. Connect to your instance using its Public DNS:
`ec2-54-166-66-100.compute-1.amazonaws.com`

Example:
`ssh -i "Node_key.pem" ubuntu@ec2-54-166-66-100.compute-1.amazonaws.com`

Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check

Step 4: Navigate to the folder where the .pem files for the key-pairs (master and nodes) have been downloaded and open the folder in terminal. Then paste the command copied in Step 3 in the terminal.

```
ubuntu@ip-172-31-84-88:~$  
System information as of Wed Sep 25 16:48:10 UTC 2024  
  
System load: 0.0          Processes:              113  
Usage of /:  22.7% of 6.71GB Users logged in:        0  
Memory usage: 5%          IPv4 address for eno33: 172.31.84.88  
Swap usage:  0%  
  
Expanded Security Maintenance for Applications is not enabled.  
0 updates can be applied immediately.  
  
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
ubuntu@ip-172-31-84-88:~$
```

```
ubuntu@ip-172-31-87-41:~$  
System information as of Wed Sep 25 16:49:44 UTC 2024  
  
System load: 0.0          Processes:              112  
Usage of /:  22.7% of 6.71GB Users logged in:        8  
Memory usage: 5%          IPv4 address for eno33: 172.31.87.41  
Swap usage:  0%  
  
Expanded Security Maintenance for Applications is not enabled.  
0 updates can be applied immediately.  
  
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
ubuntu@ip-172-31-87-41:~$
```



```
ubuntu@ip-172-31-86-164:~$  
System information as of Mon Sep 25 16:58:44 UTC 2024  
  
System load: 0.8          Processes:              115  
Usage of /: 22.8% of 6.71GB Users logged in:         0  
Memory usage: 8%         IPv4 address for eno0: 172.31.86.164  
Swap usage: 0%  
  
Expanded Security Maintenance for Applications is not enabled.  
  
0 updates can be applied immediately.  
  
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/*copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
ubuntu@ip-172-31-86-164:~$
```

Thus, the master and 2 nodes are now connected to the SSH.

Step 5: Run the following command in Master, Node1 and Node2 to install and setup docker in all 3:-

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

```
ubuntu@ip-172-31-86-164:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -  
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null  
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"  
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).  
OK  
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'  
Description:  
Archive for codename: noble components: stable  
More info: https://download.docker.com/linux/ubuntu  
Adding repository.  
Press [ENTER] to continue or Ctrl-C to cancel.  
Adding deb entry to /etc/apt/sources.list.d/archive.uri=https_download_docker_com_linux_ubuntu-noble.list  
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive.uri=https_download_docker_com_linux_ubuntu-noble.list  
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease  
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]  
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]  
Get:4 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]  
Get:5 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]  
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 kB]  
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [15.3 kB]  
Get:8 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [377 kB]  
Get:9 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [101.6 kB]  
Get:10 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4528 B]  
Get:11 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [278 kB]  
Get:12 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [111 kB]  
Get:13 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]  
Get:14 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [19.1 kB]  
Get:15 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [353 kB]  
Get:16 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [68.1 kB]
```

```

Get:36 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [353 kB]
Get:37 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [68.1 kB]
Get:38 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 c-n-f Metadata [424 B]
Get:39 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [14.4 kB]
Get:40 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [3668 B]
Get:41 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [212 B]
Get:42 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [512 B]
Get:43 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [288 B]
Get:44 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [112 B]
Get:45 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [19.6 kB]
Get:46 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [19.8 kB]
Get:47 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [19.6 kB]
Get:48 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1164 B]
Get:49 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:50 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:51 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:52 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Fetched 29.1 MB in 4s (6951 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
ubuntu@ip-172-31-86-114:~$

```

Step 6: Execute the following command:-

sudo apt-get update

sudo apt-get install -y docker-ce

```

ubuntu@ip-172-31-86-114:~$ sudo apt-get update
sudo apt-get install -y docker-ce
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containers.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libbtrfs2 libsllp0
  pigz sllp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containers.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libbtrfs2
  libsllp0 pigz sllp4netns
0 upgraded, 10 newly installed, 0 to remove and 142 not upgraded.
Need to get 123 MB of archives.
After this operation, 442 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.0-1 [65.6 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libbtrfs2 amd64 2.4.7-7build1 [48.3 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libsllp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 sllp4netns amd64 1.2.1-1build2 [34.9 kB]
Setting up docker-ce (5:27.3.1-1ubuntu.24.04-noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service + /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket + /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-86-114:~$

```

Step 7: Execute the following command:-

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

```
ubuntu@ip-172-31-86-164:~$ sudo mkdir -p /etc/docker

# Use this block to create and write into the daemon.json file
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
```

Step 8: Execute the following command:-

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

```
ubuntu@ip-172-31-90-99:~$ sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
```

Step 9: Execute the following command to install Kubernetes on all 3 machines:-

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

```
ubuntu@ip-172-31-87-41:~$ curl -fsSL https://pkgs.k8s.io/core:stable/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:stable/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:stable/v1.31/deb/ /
```


Step 10: Execute the following command:-

```
sudo apt-get update
```

```
sudo apt-get install -y kubelet kubeadm kubectl
```

```
sudo apt-mark hold kubelet kubeadm kubectl
```

```
ubuntu@ip-172-31-94-99:~$ sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv/kubernetes:/core:/stable:/v1.31/deb InRelease [1186 B]
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv/kubernetes:/core:/stable:/v1.31/deb Packages [4865 B]
Fetched 6951 B in 1s (18.7 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  contrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  contrack cri-tools kubeadm kubectl kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 142 not upgraded.
Need to get 87.4 MB of archives.
After this operation, 314 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 contrack amd64 1:1.4.8-1ubuntu1 [37.9 kB]
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv/kubernetes:/core:/stable:/v1.31/deb cri-tools 1.31.1-1.1 [18.7
```

```
Processing triggers for man-db (2.12.0-4build2) ...
```

```
Scanning processes...
```

```
Scanning linux images...
```

```
Running kernel seems to be up-to-date.
```

```
No services need to be restarted.
```

```
No containers need to be restarted.
```

```
No user sessions are running outdated binaries.
```

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

```
kubelet set on hold.
```

```
kubeadm set on hold.
```

```
kubectl set on hold.
```

```
ubuntu@ip-172-31-94-99:~$
```

Step 11: Execute the following command:-

sudo systemctl enable --now kubelet

sudo apt-get install -y containerd

```
ubuntu@ip-172-31-94-99:~$ sudo systemctl enable --now kubelet
sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libltdl0 pigz
  xlixp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 142 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8899 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 MB]
Fetched 47.2 MB in 1s (80.9 MB/s)
(Reading database ... 68044 files and directories currently installed.)
Removing docker-ce (5:27.3.1-1-ubuntu.24.04-noble) ...
Removing containerd.io (1.7.22-1) ...
Selecting previously unselected package runc.
(Reading database ... 68044 files and directories currently installed.)
Preparing to unpack .../runc_1.1.12-0ubuntu3.1_amd64.deb ...
Unpacking runc (1.1.12-0ubuntu3.1) ...
```

Scanning processes...

Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

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

ubuntu@ip-172-31-94-99:~\$

Step 12: Execute the following command:-

`sudo mkdir -p /etc/containerd`

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

```
ubuntu@ip-172-31-90-99:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

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

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

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

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

[ttrpc]
  address = ""
  gid = 0
  uid = 0
ubuntu@ip-172-31-90-99:~$
```

Step 13: Execute the following command:-

sudo systemctl restart containerd

sudo systemctl enable containerd

sudo systemctl status containerd

```
ubuntu@ip-172-31-94-99:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd
● containerd.service - containerd container runtime
   Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
   Active: active (running) since Wed 2024-09-25 18:15:47 UTC; 277ms ago
     Docs: https://containerd.io
   Main PID: 6255 (containerd)
      Tasks: 7
   Memory: 13.6M (peak: 14.2M)
      CPU: 68ms
   CGroup: /system.slice/containerd.service
           └─6255 /usr/bin/containerd

Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549371367Z" level=info msg=serving... address=0.0.0.0
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549434451Z" level=info msg="Start subscribing to events"
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549438769Z" level=info msg=serving... address=0.0.0.0
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549478289Z" level=info msg="Start recovering state"
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549534459Z" level=info msg="Start event monitoring"
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549550069Z" level=info msg="Start snapshotter"
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549568056Z" level=info msg="Start cni network"
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549567883Z" level=info msg="Start streaming"
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549628324Z" level=info msg="containerd successfully started"
Sep 25 18:15:47 ip-172-31-94-99 systemd[1]: Started containerd.service - containerd container runtime.
```

Step 14: Execute the following command:-

sudo apt-get install -y socat

```
ubuntu@ip-172-31-94-99:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libsdlpp8 pigz
  slirp4netns
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 142 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.0.8-4build3 [374 kB]
Fetched 374 kB in 0s (13.8 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68108 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.8-4build3_amd64.deb ...
Unpacking socat (1.8.0.8-4build3) ...
Setting up socat (1.8.0.8-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.
```

No user sessions are running outdated binaries.

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

Step 15: Now, we must initialize the kubernetes cluster. To do so, run the following command on the Master machine only:-

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

```
ubuntu@ip-172-31-94-99:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.11.8
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection.
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W0925 18:21:53.860369 6698 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:3.18" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/ssl"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-94-99.kubernetes.kubernetes.default.kubernetes.default.svc.kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.94.99]
[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-172-31-94-99.localhost] and IPs [172.31.94.99 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-94-99.localhost] and IPs [172.31.94.99 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file

[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

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

Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/

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

kubeadm join 172.31.94.99:6443 --token tc3id6.v5ar4lyvt5pgi3or \
--discovery-token-ca-cert-hash sha256:5e6f566ee1e3d82e939085ef1aaa178f56679b1d8ef72eb131f55be91876069
```

Step 16: Copy the following part of the output of Step 15 and run it in the terminal:-

`mkdir -p $HOME/.kube`

`sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config`

`sudo chown $(id -u):$(id -g) $HOME/.kube/config`

Also, copy the join command from the output of Step 15 for future steps.

```
ubuntu@ip-172-31-94-99:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```


Step 17: Check if the kubernetes cluster has been initialized correctly using the 'kubectl get nodes' command.

```
ubuntu@ip-172-31-94-99:~$ kubectl get nodes
NAME                STATUS    ROLES    AGE   VERSION
ip-172-31-94-99     NotReady control-plane 2m53s v1.31.1
```

We see that the cluster has been initialized without any issues and for now, only the Master machine is a part of the cluster.

Step 18: Run the join command that was previously copied only in the Node machines. This allows the Node machines to join the kubernetes cluster as well.

● Node1:-

```
ubuntu@ip-172-31-87-41:~$ sudo kubeadm join 172.31.94.99:6443 --token tc2id6.v5nr4lyvt5pgi3or \
--discovery-token-ca-cert-hash sha256:5ef2566ee1e34d82e039485e51aaa179f58639b1d4fef2cb131f55be51826469
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 1.081283532s
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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

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

● Node2:-

```
ubuntu@ip-172-31-86-164:~$ sudo kubeadm join 172.31.94.99:6443 --token tc2id6.v5nr4lyvt5pgi3or \
--discovery-token-ca-cert-hash sha256:5ef2566ee1e34d82e039485e51aaa179f58639b1d4fef2cb131f55be51826469
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 508.72345ms
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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

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

Step 19: Run the 'kubectl get nodes' command on the Master machine again to check if the Node machines have successfully joined the cluster.

```
ubuntu@ip-172-31-94-99:~$ kubectl get nodes
NAME                STATUS    ROLES    AGE   VERSION
ip-172-31-86-164     NotReady <none>    34s   v1.31.1
ip-172-31-87-41      NotReady <none>    73s   v1.31.1
ip-172-31-94-99     NotReady control-plane 6m17s v1.31.1
```

But, it is observed that the status of all the nodes is 'NotReady'.

Step 20: We must add a network plugin to change the status of the nodes to 'Ready':-
kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml

```
ubuntu@ip-172-31-94-99:~$ kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
poddisruptionbudget.policy/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
serviceaccount/calico-node created
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgppeers.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/blockaffinities.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/caliconodestatuses.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/clusterinformations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/felixconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworksets.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/hostendpoints.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamblocks.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamconfigs.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamhandles.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippools.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippreservations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/kubecontrollersconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networksets.crd.projectcalico.org created
clusterrole.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrole.rbac.authorization.k8s.io/calico-node created
clusterrolebinding.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
daemonset.apps/calico-node created
deployment.apps/calico-kube-controllers created
```

Step 21: Execute the following command:-
sudo systemctl status kubelet

```
ubuntu@ip-172-31-94-99:~$ sudo systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
   Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; preset: enabled)
   Drop-In: /usr/lib/systemd/system/kubelet.service.d
            └─10-kubeadm.conf
   Active: active (running) since Wed 2024-09-25 18:22:14 UTC; 8min ago
     Docs: https://kubernetes.io/docs/
   Main PID: 7366 (kubelet)
    Tasks: 10 (limit: 4676)
   Memory: 33.6M (peak: 34.1M)
      CPU: 8.441s
   CGroup: /system.slice/kubelet.service
            └─7366 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kube

Sep 25 18:38:29 ip-172-31-94-99 kubelet[7366]: I0925 18:38:29.856287 T366 pod_container_deletor.go:89] "Container not
Sep 25 18:38:29 ip-172-31-94-99 kubelet[7366]: I0925 18:38:29.856234 T366 scope.go:117] "RemoveContainer" containerID
Sep 25 18:38:29 ip-172-31-94-99 kubelet[7366]: I0925 18:38:29.857797 T366 scope.go:117] "RemoveContainer" containerID
Sep 25 18:38:29 ip-172-31-94-99 kubelet[7366]: I0925 18:38:29.869340 T366 scope.go:117] "RemoveContainer" containerID
Sep 25 18:38:29 ip-172-31-94-99 kubelet[7366]: I0925 18:38:29.876450 T366 scope.go:117] "RemoveContainer" containerID
Sep 25 18:38:29 ip-172-31-94-99 kubelet[7366]: I0925 18:38:29.877260 T366 scope.go:117] "RemoveContainer" containerID
Sep 25 18:38:33 ip-172-31-94-99 kubelet[7366]: E0925 18:38:33.872874 T366 pod_workers.go:1301] "Error syncing pod, s
Sep 25 18:38:39 ip-172-31-94-99 kubelet[7366]: I0925 18:38:39.167638 T366 scope.go:117] "RemoveContainer" containerID
Sep 25 18:38:39 ip-172-31-94-99 kubelet[7366]: E0925 18:38:39.167749 T366 pod_workers.go:1301] "Error syncing pod, s
Sep 25 18:38:39 ip-172-31-94-99 kubelet[7366]: I0925 18:38:39.882131 T366 scope.go:117] "RemoveContainer" containerID
```

Step 22: Run the 'kubectl get nodes' to check if the status of the nodes have been successfully updated or not.

```
ubuntu@ip-172-31-94-99:~$ kubectl get nodes
NAME                 STATUS    ROLES                  AGE      VERSION
ip-172-31-86-164     Ready     <none>                 3m37s    v1.31.1
ip-172-31-87-41      Ready     <none>                 4m16s    v1.31.1
ip-172-31-94-99      Ready     control-plane          9m20s    v1.31.1
```

Step 23: To rename the nodes to their actual names instead of their IP address, run the following on the Master machine:-

```
kubectl label node ip-172-31-87-41 kubernetes.io/role=Node1
```

```
kubectl label node ip-172-31-86-164 kubernetes.io/role=Node2
```

```
ubuntu@ip-172-31-94-99:~$ kubectl label node ip-172-31-87-41 kubernetes.io/role=Node1
node/ip-172-31-87-41 labeled
ubuntu@ip-172-31-94-99:~$ kubectl label node ip-172-31-86-164 kubernetes.io/role=Node2
node/ip-172-31-86-164 labeled
```

Step 24: Run the 'kubectl get nodes' to check if the 'Roles' of the nodes have been successfully updated or not.

```
ubuntu@ip-172-31-94-99:~$ kubectl get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
ip-172-31-86-164	Ready	Node2	7m58s	v1.31.1
ip-172-31-87-41	Ready	Node1	8m37s	v1.31.1
ip-172-31-94-99	Ready	control-plane	13m	v1.31.1

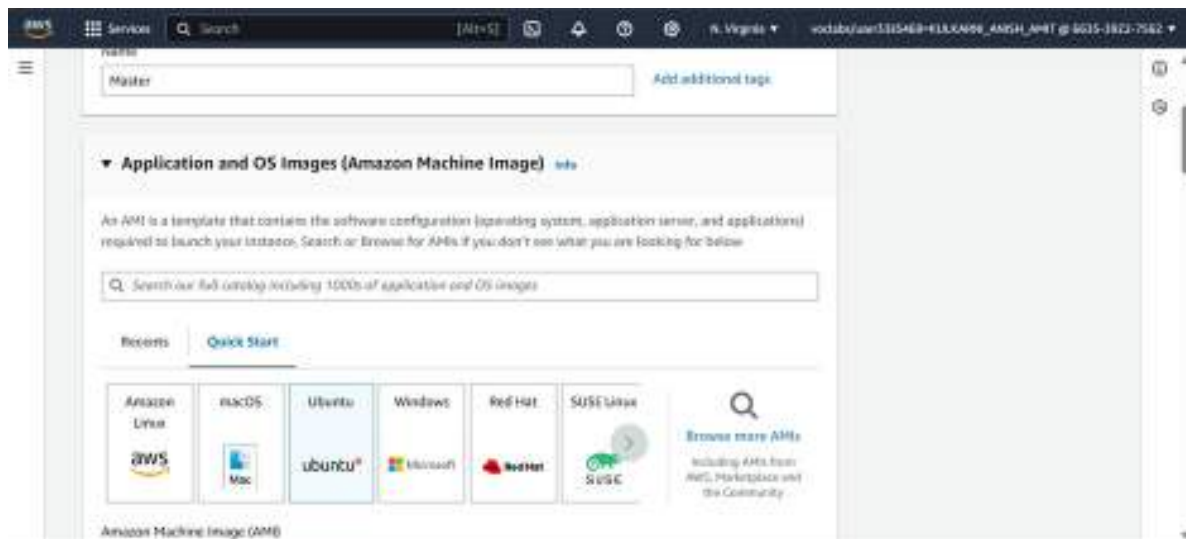
Conclusion: In this experiment, we learned about the Kubernetes cluster architecture and how to install and spin up a Kubernetes cluster on Linux Machines/Cloud Platforms. First, we created 3 EC2 Ubuntu instances on AWS (1 master and 2 worker nodes) and connected them to our local terminal using SSH. Then, we installed and configured docker and kubernetes on all machines and added the master node to a kubernetes cluster. Then, we used the 'join' command to add the worker nodes to the cluster as well. At the end, we have a kubernetes cluster with all 3 machines with 'Ready' status.

Experiment 4

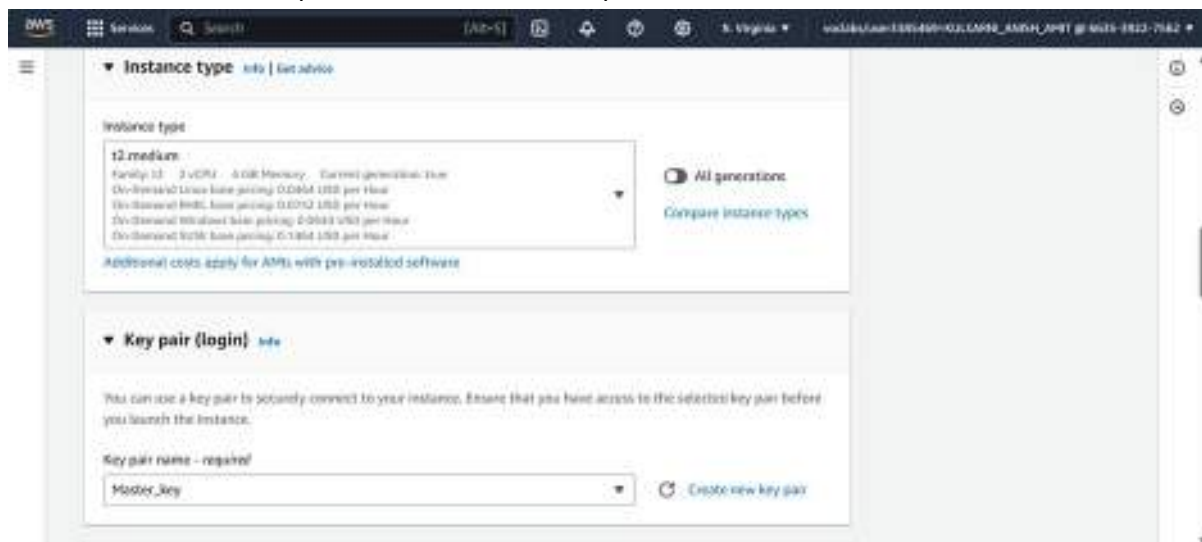
Aim: To install Kubectl and execute Kubectl command to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

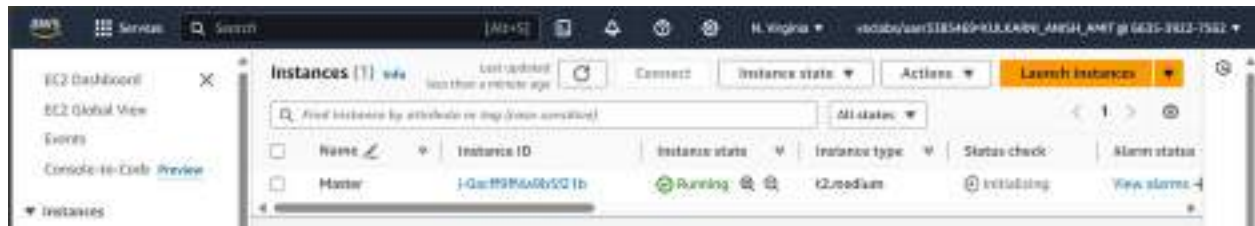
Steps:

Step 1: Create a new EC2 instance in AWS. Make sure that you choose Ubuntu as the instance type.

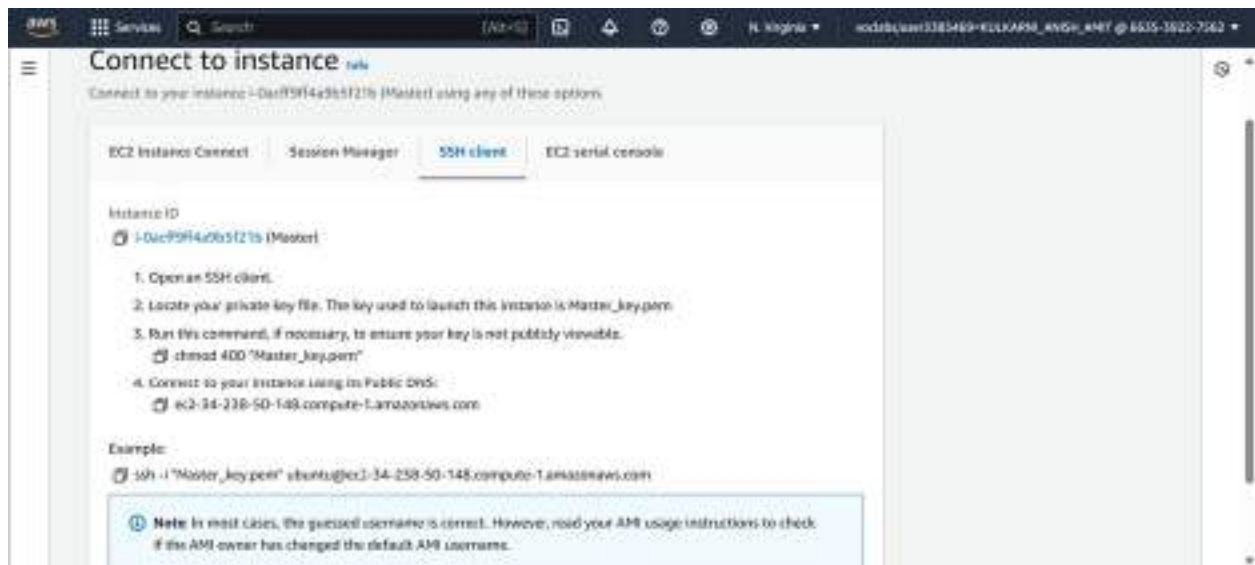


Under 'Instance type', choose t2.medium because the default (t2.micro) does not provide the sufficient resources required to execute this experiment.





Step 2: Now, we must connect the instance to a SSH. To do so, click on the instance and click on 'Connect'. Next, navigate to the 'SSH client' section and copy the command under the 'Example' section.



Step 3: Navigate to the folder where the .pem file for the Master key-pair has been downloaded and open the folder in terminal. Then paste the command copied in Step 2 in the terminal.

```
ubuntu@ip-172-31-94-127: ~$ ssh -i "Master_key.pem" ubuntu@ec2-34-238-58-148.compute-1.amazonaws.com
PS C:\Users\anish\Downloads> ssh -i "Master_key.pem" ubuntu@ec2-34-238-58-148.compute-1.amazonaws.com
The authenticity of host 'ec2-34-238-58-148.compute-1.amazonaws.com (34.238.58.148)' can't be established.
ED25519 key fingerprint is SHA256:A8tEafucdELWwW+SLCL56fSjVG8hjT08vPXhtySIL4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-34-238-58-148.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

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

System information as of Wed Sep 25 10:44:47 UTC 2024

System load: 0.2                Processes:              117
Usage of /:  22.8% of 6.71GB    Users logged in:       0
Memory usage: 6%                IPv4 address for eni0: 172.31.94.127
Swap usage:  0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

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

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

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

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

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

```
ubuntu@ip-172-31-94-127: ~$
```

Step 4: Run the following command to install and setup docker:-

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee
```

```
/etc/apt/trusted.gpg.d/docker.gpg > /dev/null
```

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

```
ubuntu@ip-172-31-94-127:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -  
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null  
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"  
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).  
OK  
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble:stable'  
Description:  
Archive for codename: noble components: stable  
More info: https://download.docker.com/linux/ubuntu  
Adding repository.  
Press [ENTER] to continue or Ctrl-C to cancel.  
Adding deb entry to /etc/apt/sources.list.d/archive_uri=https.download.docker.com.linux.ubuntu-noble.list  
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri=https.download.docker.com.linux.ubuntu-noble.list  
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease  
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]  
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]  
Get:4 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]  
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.8 MB]  
Get:6 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]  
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [15.3 kB]  
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]  
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]  
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [391 kB]  
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]  
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]  
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]  
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]  
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [631 kB]  
  
Get:47 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [68.1 kB]  
Get:48 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [428 B]  
Get:49 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [18.9 kB]  
Get:50 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [2898 B]  
Get:51 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [268 B]  
Get:52 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [344 B]  
Fetched 29.1 MB in 4s (7834 kB/s)  
Reading package lists... Done  
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.  
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.  
ubuntu@ip-172-31-94-127:~$
```

Step 5: Execute the following command:-

`sudo apt-get update`

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

```
ubuntu@ip-172-31-94-127:~$ sudo apt-get update
sudo apt-get install -y docker-ce
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0
  pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7
  libslirp0 pigz slirp4netns
8 upgraded, 10 newly installed, 0 to remove and 142 not upgraded.
Need to get 123 MB of archives.
After this operation, 682 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libltdl7 amd64 2.4.7-7build1 [49.3 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libslirp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
```

Scanning processes...

Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

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

ubuntu@ip-172-31-94-127:~\$

Step 6: Execute the following command:-

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

```
ubuntu@ip-172-31-94-127:~$ sudo mkdir -p /etc/docker

# Use this block to create and write into the daemon.json file
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
```

Step 7: Execute the following command:-

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

```
ubuntu@ip-172-31-94-127:~$ sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
```

Step 8: Run the following command to install and setup kubernetes:-

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

```
ubuntu@ip-172-31-94-127:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
```

Step 9: Execute the following command:-

sudo apt-get update

sudo apt-get install -y kubelet kubeadm kubectl

sudo apt-mark hold kubelet kubeadm kubectl

```
ubuntu@ip-172-31-94-127:~$ sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:5 https://download.docker.com/linux/ubuntu noble InRelease
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb InRelease [1186 B]
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb Packages [4865 B]
Fetched 6051 B in 1s (12.1 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  contrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  contrack cri-tools kubeadm kubectl kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 142 not upgraded.
Need to get 57.4 MB of archives.
After this operation, 314 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 contrack amd64 1:1.4.8-1ubuntu1 [37.9 kB]
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb cri-tools 1.31.1-1.1 [15.7 MB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb kubeadm 1.31.1-1.1 [11.4 MB]
Setting up kubeadm (1.31.1-1.1) ...
Setting up kubelet (1.31.1-1.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
ubuntu@ip-172-31-94-127:~$
```

Step 10: Execute the following command:-

sudo systemctl enable --now kubelet

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

```
ubuntu@ip-172-31-04-127:~$ sudo systemctl enable --now kubelet
sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.8
[preflight] Running pre-flight checks
W0925 19:58:23.457259: 4286 checks.go:1888] [preflight] WARNING: Couldn't create the interface used for talking to the
container runtime: failed to create new CRI runtime service: validate service connection: validate CRI v1 runtime API f
or endpoint "unix:///var/run/containerd/containerd.sock": rpc error: code = Unimplemented desc = unknown service runtime
.v1.RuntimeService
[WARNING FileExisting-socat]: socat not found in system path
[preflight] Pulling images required for setting up a Kubernetes cluster.
[preflight] This might take a minute or two, depending on the speed of your internet connection.
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
error execution phase preflight: [preflight] Some fatal errors occurred:
failed to create new CRI runtime service: validate service connection: validate CRI v1 runtime API for endpoint "unix://
/var/run/containerd/containerd.sock": rpc error: code = Unimplemented desc = unknown service runtime.v1.RuntimeService[p
reflight] If you know what you are doing, you can make a check non-fatal with '--ignore-preflight-errors...'
To see the stack trace of this error execute with --v=5 or higher
```

But, we encounter the above error on running the commands. Thus, a few more commands need to be run in order to solve the error.

Step 11: Execute the following command:-

sudo apt-get install -y containerd

```
ubuntu@ip-172-31-04-127:~$ sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libtdb1 libtirpc pigz
  slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 142 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu3.1 [38.0 M
B]
Fetched 47.2 MB in 1s (63.2 MB/s)
(Reading database ... 68664 files and directories currently installed.)
Removing docker-ce (5:27.1.1-1ubuntu.24.04-noble) ...
Removing containerd.io (1.7.12-1) ...
Selecting previously unselected package runc.
(Reading database ... 68644 files and directories currently installed.)
Preparing to unpack .../runc_1.1.12-0ubuntu3.1_amd64.deb ...
Unpacking runc (1.1.12-0ubuntu3.1) ...
Selecting previously unselected package containerd.
```

```
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-04-127:~$ |
```

Step 12: Execute the following command:-

sudo mkdir -p /etc/containerd

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

```
ubuntu@ip-172-31-94-127:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2
```

```
[cgroup]
  path = ""
```

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

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

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

```
[timeouts]
  "io.containerd.timeout.bolt.open" = "0s"
  "io.containerd.timeout.metrics.shinstats" = "1s"
  "io.containerd.timeout.shim.cleanup" = "5s"
  "io.containerd.timeout.shim.load" = "5s"
  "io.containerd.timeout.shim.shutdown" = "3s"
  "io.containerd.timeout.task.state" = "2s"
```

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

```
ubuntu@ip-172-31-94-127:~$
```


Step 13: Execute the following command:-

sudo systemctl restart containerd

sudo systemctl enable containerd

sudo systemctl status containerd

```
ubuntu@ip-172-31-94-127:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd
● containerd.service - containerd container runtime
   Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
   Active: active (running) since Wed 2024-09-25 20:01:08 UTC; 229ms ago
     Docs: https://containerd.io
   Main PID: 4658 (containerd)
      Tasks: 7
    Memory: 13.4M (peak: 14.1M)
       CPU: 57ms
    CGroup: /system.slice/containerd.service
           └─4658 /usr/bin/containerd

Sep 25 20:01:08 ip-172-31-94-127 containerd[4658]: time="2024-09-25T20:01:08.945083945Z" level=error msg="failed to load
Sep 25 20:01:08 ip-172-31-94-127 containerd[4658]: time="2024-09-25T20:01:08.945806710Z" level=info msg="Start subscrib
Sep 25 20:01:08 ip-172-31-94-127 containerd[4658]: time="2024-09-25T20:01:08.945848268Z" level=info msg="Start recoveri
Sep 25 20:01:08 ip-172-31-94-127 containerd[4658]: time="2024-09-25T20:01:08.945896866Z" level=info msg="Start event ms
Sep 25 20:01:08 ip-172-31-94-127 containerd[4658]: time="2024-09-25T20:01:08.945905090Z" level=info msg="Start snapshot
Sep 25 20:01:08 ip-172-31-94-127 containerd[4658]: time="2024-09-25T20:01:08.945913873Z" level=info msg="Start cni netw
Sep 25 20:01:08 ip-172-31-94-127 containerd[4658]: time="2024-09-25T20:01:08.945928246Z" level=info msg="Start streamin
Sep 25 20:01:08 ip-172-31-94-127 containerd[4658]: time="2024-09-25T20:01:08.945912786Z" level=info msg="serving... addr
Sep 25 20:01:08 ip-172-31-94-127 containerd[4658]: time="2024-09-25T20:01:08.946023718Z" level=info msg="serving... addr
Sep 25 20:01:08 ip-172-31-94-127 containerd[4658]: time="2024-09-25T20:01:08.947101438Z" level=info msg="containerd suc
ubuntu@ip-172-31-94-127:~$
```

Step 14: Execute the following command:-

sudo apt-get install -y socat

```
ubuntu@ip-172-31-94-127:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp8 pigz
  slirp4netns
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 142 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.8.0-4build3 [374 kB]
Fetched 374 kB in 0s (17.0 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68108 files and directories currently installed.)
Preparing to unpack .../socat_1.8.8.0-4build3_amd64.deb ...
Unpacking socat (1.8.8.0-4build3) ...
Setting up socat (1.8.8.0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.
```

Now, the error that was encountered in Step 10 should be fixed.

Step 15: Initialise the kubernetes cluster. To do so, execute the following command:-
sudo kubeadm init --pod-network-cidr=10.244.0.0/16

```
ubuntu@ip-172-31-94-127:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W0925 20:04:25.033576 4946 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/ssl"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-94-127 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.94.127]
[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-172-31-94-127 localhost] and IPs [172.31.94.127 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-94-127 localhost] and IPs [172.31.94.127 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file

[bootstrap-token] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Token
[bootstrap-token] Configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

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

Alternatively, if you are the root user, you can run:

export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run 'kubectl apply -f [podnetwork].yaml' with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/

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

kubeadm join 172.31.94.127:6443 --token luj1sh.9tuddvu59wbxrv4 \
--discovery-token-ca-cert-hash sha256:82da7c0275290ee835fa49156f1b059216030c3f1ac6501184759c6da82277f1
ubuntu@ip-172-31-94-127:~$
```

Step 16: Copy the following part of the output of Step 15 and run it in the terminal:-
mkdir -p \$HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf \$HOME/.kube/config
sudo chown \$(id -u):\$(id -g) \$HOME/.kube/config
Also, copy the join command from the output of Step 15 for future steps.

```
ubuntu@ip-172-31-94-127:~$ mkdir -p $HOME/.kube
ubuntu@ip-172-31-94-127:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
ubuntu@ip-172-31-94-127:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-94-127:~$
```

Step 17: Add a common networking plugin called Flannel to the kubernetes cluster:-

kubectl apply -f

<https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml>

```
ubuntu@ip-172-31-94-127:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
```

Step 18: The kubernetes cluster has been created and initialized. Now, we can deploy our nginx server on the cluster. To apply this deployment file, use the following command to create a deployment:-

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

```
ubuntu@ip-172-31-94-127:~$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
```

Step 19: Check the status of the kubernetes cluster using 'kubectl get pods' command.

```
ubuntu@ip-172-31-94-127:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-cr6j8    0/1     Pending   0           36s
nginx-deployment-d556bf558-q5kng    0/1     Pending   0           36s
```

Step 20: Now, to access the kubernetes pod running the nginx application, run the following command:-

POD_NAME=\$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
kubectl port-forward \$POD_NAME 8080:80

```
ubuntu@ip-172-31-94-127:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
kubectl port-forward $POD_NAME 8080:80
error: unable to forward port because pod is not running. Current status=Pending
```

But, we see that an error is encountered because the status of our pod is 'Pending' and not 'Running'. To do so, we must untaint the tainted nodes.

Step 21: Execute the following command:-

kubectl taint nodes --all node-role.kubernetes.io/control-plane-

```
ubuntu@ip-172-31-94-127:~$ kubectl taint nodes --all node-role.kubernetes.io/control-plane-
node/ip-172-31-94-127 untainted
```

This changes the status of the pod to 'Running'.

Step 22: Run 'kubectl get nodes' to check on the status of the kubernetes cluster.

```
ubuntu@ip-172-31-94-127:~$ kubectl get nodes
NAME                STATUS    ROLES                  AGE     VERSION
ip-172-31-94-127    Ready     control-plane          9m38s   v1.31.1
```


Step 23: Run the 'kubectl get pods' command again to check if the status of our pod has been updated to 'Running' or not.

```
ubuntu@ip-172-31-94-127:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-cr6j8    1/1     Running   0           6m50s
nginx-deployment-d556bf558-q5kng    1/1     Running   0           6m50s
```

Step 24: Execute the following command again:-

POD_NAME=\$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
kubectl port-forward \$POD_NAME 8080:80

```
ubuntu@ip-172-31-94-127:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
kubectl port-forward $POD_NAME 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
Handling connection for 8080
```

Step 25: To verify the deployment, open a new terminal (SSH) for your EC2 instance.

```
P5 C:\Users\anish\Downloads> ssh -i "Master_key.pem" ubuntu@ec2-34-238-56-140.compute-1.amazonaws.com
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1812-aws x86_64)

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

System information as of Wed Sep 25 20:22:38 UTC 2024

System load:  0.8               Processes:    154
Usage of /:   55.4% of 6.71GB   Users logged in: 1
Memory usage: 19%              IPv4 address for enx0: 172.31.94.127
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

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

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

Last login: Wed Sep 25 19:44:48 2024 from 182.48.216.266
```

Step 26: Run the following command to check if the nginx server is running or not:-

```
ubuntu@ip-172-31-94-127:~$ curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Wed, 25 Sep 2024 20:23:09 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 04 Dec 2018 14:44:49 GMT
Connection: keep-alive
ETag: "5c0692e1-264"
Accept-Ranges: bytes
```

As the response says '200 OK' along with the server name, we can conclude that we successfully deployed our nginx server on our EC2 instance.

Conclusion: In this experiment, we learned how to install kubectl and execute kubectl commands to manage the Kubernetes cluster and deploy our first Kubernetes application. First, we created an EC2 Ubuntu instance on AWS and connected it to our local terminal using SSH. Then, we installed and configured Docker and Kubernetes on the machine and added the machine to the kubernetes cluster. Next, we deployed the Flannel networking plugin and nginx server on our cluster and forwarded port 8080 on our local machine to port 80 on the specified pod to deploy and run the nginx server on our EC2 instance.

Experiment No. 5

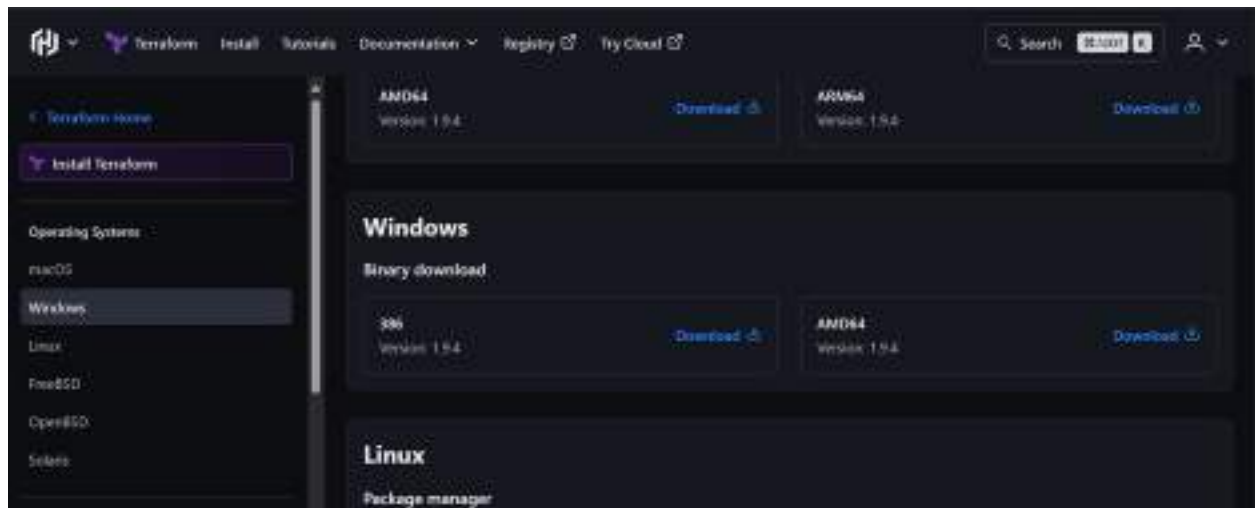
Aim: To understand Terraform lifecycle, core concepts/terminologies and install it on a Linux machine and Windows.

Steps:-

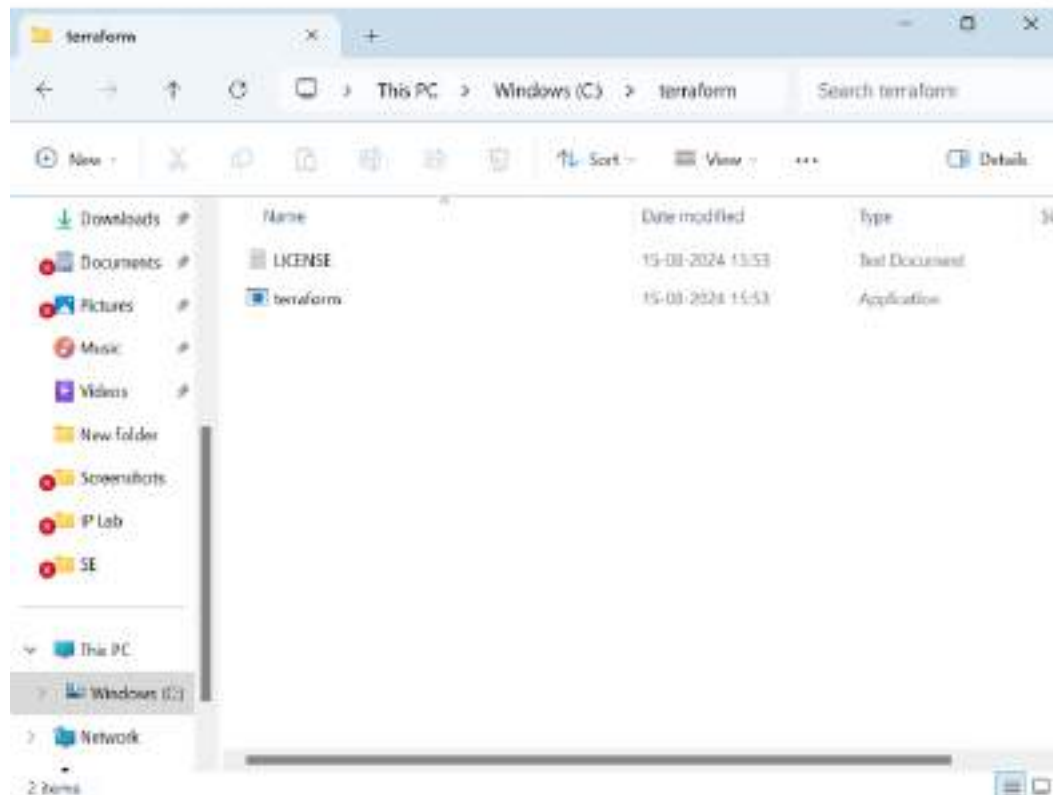
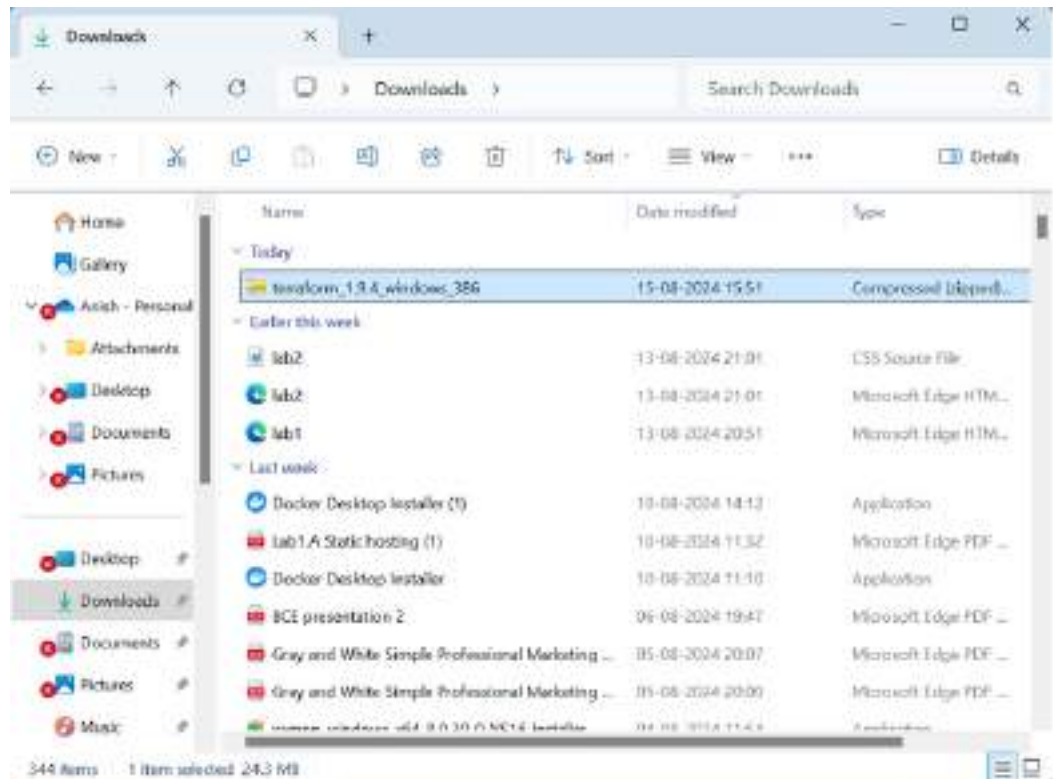
Step 1: Install Terraform from Terraform's official website:

<https://www.terraform.io/downloads.html>.

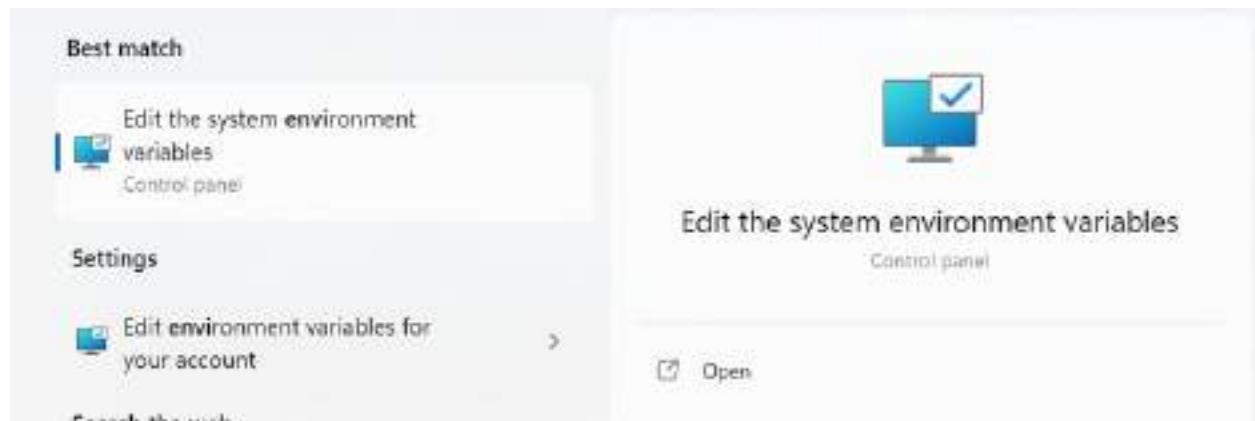
Select your operating system followed by either 32bit or 64bit depending on your OS type.



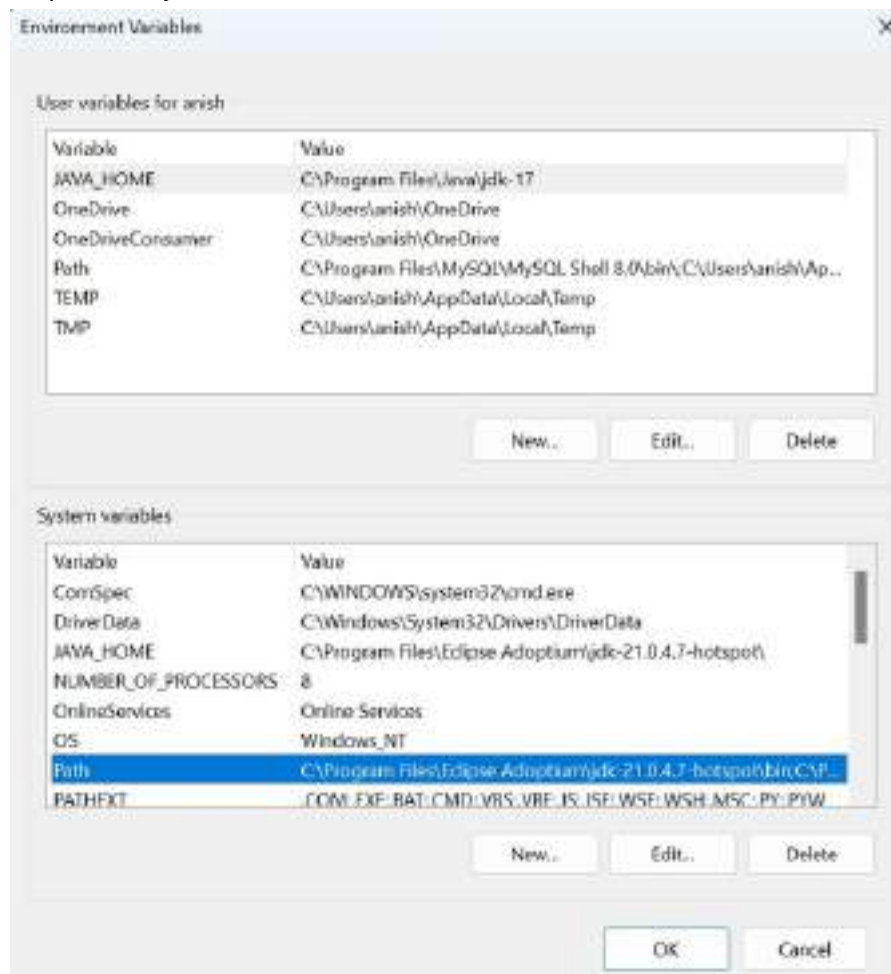
Step 2: Once the download is complete, navigate to the downloaded zipped terraform folder and extract the folder (including the setup file Terraform.exe) into 'C:\Terraform' directory.



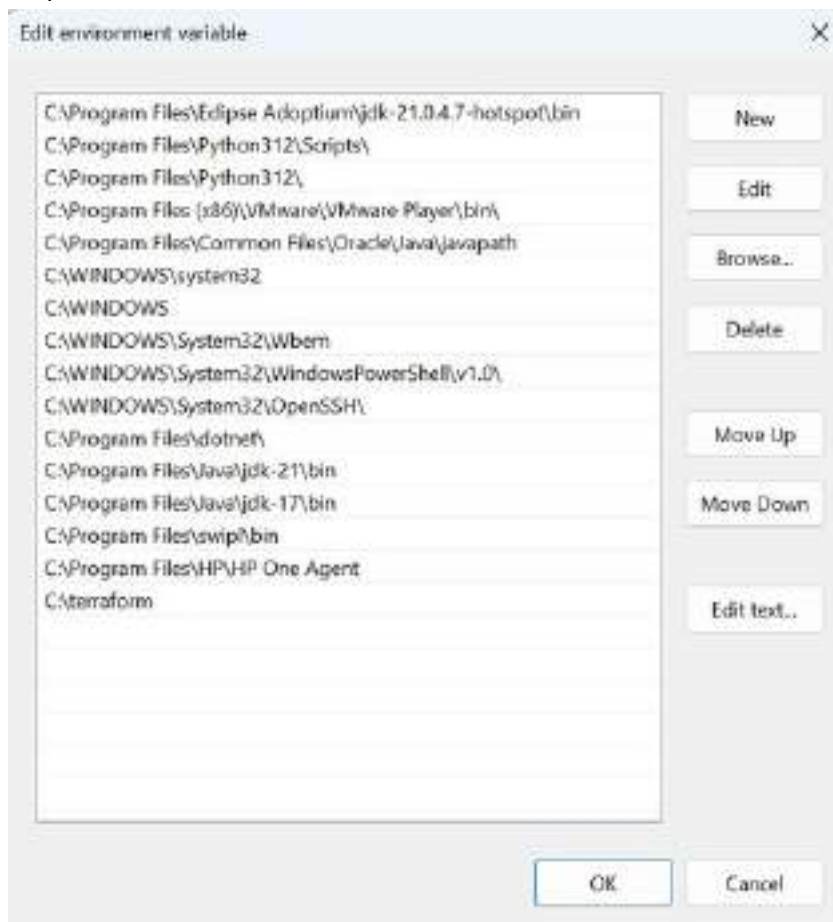
Step 3: Navigate to 'Edit the system environment variables' in your device and click on the 'Environment variables' button.



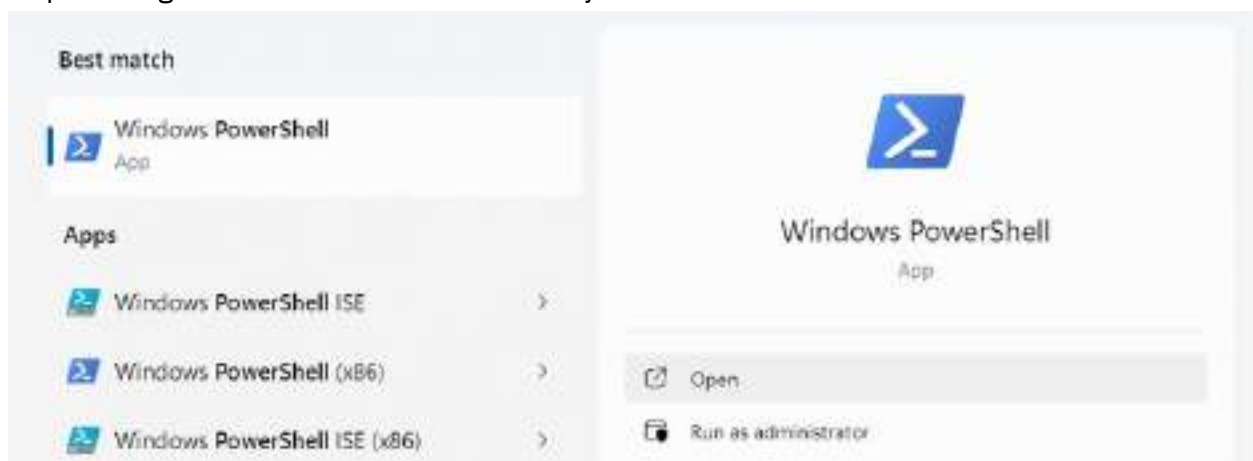
Step 4: In 'System variables', click on 'Path' and then click on the 'Edit' button.



Step 5: Click on 'New' and enter 'C:\terraform' and click on 'OK'.



Step 6: Navigate to 'Windows PowerShell' on your device and click on 'Run as administrator'.



Step 7: Open Terraform in PowerShell. This displays all the main commands, other commands and global options. This being your output means that you have successfully installed Terraform on your system.

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

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

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

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

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

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

Conclusion: The installation of Terraform on a Windows machine involves downloading the appropriate version, setting up the Terraform folder on the system, and configuring the system's environment variables to recognize Terraform commands. By adding the folder to the system path and verifying the installation via PowerShell, users can effectively utilize Terraform for infrastructure automation. This setup is essential for integrating Terraform into the system's commandline, enabling smooth execution of Terraform commands for managing cloud infrastructure.

Experiment 6

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

Steps:-

Step 1: Install Docker Desktop from its official website at <https://www.docker.com/> and check docker's functionality by using the 'docker' and 'docker --version' commands in Powershell.

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

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

PS C:\Users\anish> docker

Usage: docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

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

Management Commands:
builder       Manage builds
buildx        Docker Buildx
compose       Docker Compose
container     Manage containers
context       Manage contexts
debug         Get a shell into any image or container
desktop       Docker Desktop commands (Alpha)
dev           Docker Dev Environments
extensions    Manage Docker extensions
feedback      Provide feedback, right in your terminal!
image         Manage images
init          Creates Docker-related starter files for your project
manifest      Manage Docker image manifests and manifest lists
network       Manage networks
plugin        Manage plugins
sbom          View the packaged-based Software Bill Of Materials (SBOM) for an image
scout         Docker Scout
system        Manage Docker
trust         Manage trust on Docker images
volume        Manage volumes

Swarm Commands:
swarm         Manage Swarm

Commands:
attach        Attach local standard input, output, and error streams to a running container
commit        Create a new image from a container's changes
cp            Copy files/folders between a container and the local filesystem
create        Create a new container
diff          Inspect changes to files or directories on a container's filesystem
events        Get real time events from the server
export        Export a container's filesystem as a tar archive
```



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

Step 2: Create a folder named 'Terraform Scripts'. Create a folder named 'Docker' inside of the 'Terraform Scripts' folder and create a new file named docker.tf in this folder. Write the following in the 'docker.tf' file (creates a container):-

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

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

resource "docker_image" "ubuntu" {
  name = "ubuntu:latest"
}

resource "docker_container" "foo" {
  image = docker_image.ubuntu.image_id
  name = "foo"
}
```

```
Welcome  docker.tf X
docker > docker.tf > resource "docker_container" "foo"
1  terraform {
2    required_providers {
3      docker = {
4        source = "kreuzwerker/docker"
5        version = "2.21.0"
6      }
7    }
8  }
9
10 provider "docker" {
11   host = "npipe:////./pipe/docker_engine"
12 }
13
14 # Pulls the image
15 resource "docker_image" "ubuntu" {
16   name = "ubuntu:latest"
17 }
18
19 # Create a container
20 resource "docker_container" "foo" {
21   image = docker_image.ubuntu.image_id
22   name = "foo"
23 }
```

Step 3: Execute 'terraform init' command in Powershell. This command initializes a Terraform working directory by downloading necessary plugins and setting up the backend for state management.

```
C:\Users\anish\OneDrive\Desktop\Terraform Scripts\docker>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding kreuzwerker/docker versions matching "2.21.0"...
- Installing kreuzwerker/docker v2.21.0...
- Installed kreuzwerker/docker v2.21.0 (self-signed, key ID BD080C4571C6184C)
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.

C:\Users\anish\OneDrive\Desktop\Terraform Scripts\docker>
```

Step 4: Execute 'terraform plan' command to generate and display an execution plan, showing what actions Terraform will take to achieve the desired infrastructure state without making any actual changes.

```
C:\Users\anish\OneDrive\Desktop\Terraform Scripts\docker>terraform plan

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

Terraform will perform the following actions:

# docker_container.foo will be created
+ resource "docker_container" "foo" {
  + attach      = false
  + bridge      = [known after apply]
  + command     = [known after apply]
  + container_logs = [known after apply]
  + entrypoint  = [known after apply]
  + env         = [known after apply]
  + exit_code   = [known after apply]
  + gateway     = [known after apply]
  + hostname    = [known after apply]
  + id          = [known after apply]
  + image       = [known after apply]
  + init        = [known after apply]
  + ip_address  = [known after apply]
  + init_prefix_length = [known after apply]
  + ipc_mode    = [known after apply]
  + log_driver  = [known after apply]
  + logs        = false
  + mount_nas   = true
  + name        = "foo"
  + network_data = [known after apply]
  + read_only   = false
  + remove_volumes = true
  + restart     = "no"
  + rm          = false
  + runtime     = [known after apply]
  + security_opts = [known after apply]
  + size_bytes  = [known after apply]
  + start       = true

  + stdin_open = false
  + stop_signal = [known after apply]
  + stop_timeout = [known after apply]
  + tty         = false

  + healthcheck [known after apply]

  + labels [known after apply]
}

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

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

```
Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run
"terraform apply" now.
```

Step 5: Execute 'terraform apply' command to execute the changes outlined in the Terraform plan, creating, updating, or deleting resources in the infrastructure.

On executing the command we see that the following error occurs:-

```
Error: container exited immediately

with docker_container.foo,
on docker.tf line 20, in resource "docker_container" "foo":
20: resource "docker_container" "foo" {

C:\Users\anish\OneDrive\Desktop\Terraform Scripts\docker>
```

This error occurs because the container's entry command or process gets completed too quickly, causing the container to stop running. To fix the error, add the following lines of code at the end of the docker.tf file.

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

Now, executing the 'terraform apply' command gives the following output:-

```
C:\Users\anish\OneDrive\Desktop\Terraform Scripts\docker>terraform apply
docker_image.ubuntu: Refreshing state... [id=sha256:ed8-fe79641f8a3581ce543e137cf28ea0dd82e6df8c9d66519b64ae761c2598aubuntu:latest]

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",
    + "infinity",
  ]
  + 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        = "sha256:ed8fe79641f8a3581ce543e137cf28ea0dd82e6df8c9d66519b64ae761c2598a"
  + init         = (known after apply)
  + ip_address    = (known after apply)
  + ip_prefix_length = (known after apply)
  + ip_mode       = (known after apply)
  + log_driver    = (known after apply)
  + logs         = false
  + mount_run    = true
  + name         = "foo"
  + network_data  = (known after apply)
  + read_only     = false
  + remove_volumes = true
  + restart      = "no"
  + rm           = false
  + runtime       = (known after apply)
}
```

```

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

+ healthcheck (known after apply)

+ labels (known after apply)
}

Plan: 1 to add, 0 to change, 0 to destroy.

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

Enter a value: yes

docker_container.foo: Creating...
docker_container.foo: Creation complete after 1s [id=83348de94d5fd1a9a5ad8cf681c97682b4bd75d797432d383147cf62839d3049]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

```

Executing the 'terraform apply' command executes the Terraform plan and creates a docker image. This can be seen as such:-

● Dockerimagesbeforeexecuting'terraformapply':-

```

C:\Users\anish\OneDrive\Desktop\Terraform Scripts\docker>docker images
REPOSITORY      TAG          IMAGE ID       CREATED        SIZE
C:\Users\anish\OneDrive\Desktop\Terraform Scripts\docker>

```

● Dockerimagesafterexecuting'terraformapply':-

```

C:\Users\anish\OneDrive\Desktop\Terraform Scripts\docker>docker images
REPOSITORY      TAG          IMAGE ID       CREATED        SIZE
ubuntu          latest       edbfe74c41f8   2 weeks ago    78.1MB

```

Step 6: Execute the 'terraform destroy' command to remove all the infrastructure resources that Terraform previously created, effectively tearing down the environment. This automatically deletes the docker image that was created in the previous step.

```
C:\Users\anish\OneDrive\Desktop\Terraform_Scripts\docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbf74c41f8a3581ce542e137cf28ea84dd93e6df8c9d66519b6ae761c2598a:latest]
docker_container.foo: Refreshing state... [id=sha256:9a8b4d1a9a5ad8df881e07462b4bd75d70712d385157cf463839d3819]

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    = []
  console     = []
  "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   = "08348de9ad5f" -> null
  id         = "08348de9ad5f41a8a5a3d4f031e97692b2e75c78742d3d301167f6c239d3648" -> null
  image      = "sha256:edbf74c41f8a3581ce542e137cf28ea84dd93e6df8c9d66519b6ae761c2598a" -> null
  restart    = false -> null
  ip_address = "172.17.0.2" -> null
  ip_prefix_length = 16 -> null
  ipc_mode   = "private" -> null
  links      = [] -> null
  log_driver = "json-file" -> null
  log_opts   = {} -> null
  logs       = false -> null
  max_retry_count = 0 -> null
  memory     = 0 -> null
  memory_swap = 0 -> null
  must_run   = true -> null
  name       = "foo" -> null
}
```

```
network_data = {
  gateway = "172.17.0.1"
  global_ip_v4_prefix_length = 0
  ip_address = "172.17.0.2"
  ip_prefix_length = 16
  network_name = "bridge"
  # {} unchanged attributes hidden
}
} -> null
network_mode = "bridge" -> null
privileged = false -> null
publish_all_ports = false -> null
read_only = false -> null
remove_volumes = true -> null
restart = "no" -> null
rm = false -> null
runtime = "runc" -> null
security_opts = [] -> null
shm_size = 64 -> null
start = true -> null
stdin_open = false -> null
stop_timeout = 0 -> null
storage_opts = {} -> null
sysctl = {} -> null
tmpfs = {} -> null
tty = false -> null
# {} unchanged attributes hidden
}

# docker_image.ubuntu will be destroyed
resource "docker_image" "ubuntu" {
  id = "sha256:edbf74c41f8a3581ce542e137cf28ea84dd93e6df8c9d66519b6ae761c2598a:latest" -> null
  image_id = "sha256:edbf74c41f8a3581ce542e137cf28ea84dd93e6df8c9d66519b6ae761c2598a" -> null
  latest = "sha256:edbf74c41f8a3581ce542e137cf28ea84dd93e6df8c9d66519b6ae761c2598a" -> null
  name = "ubuntu:latest" -> null
  repo_digest = "ubuntu@sha256:8a77d66f4f72ebf3c9efa4bcf662779b13720982a8038c1609ef04e14a9ab62ee" -> 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=sha256:9a8b4d1a9a5ad8df881e07462b4bd75d70712d385157cf463839d3819]
docker_container.foo: Destruction complete after 1s
docker_image.ubuntu: Destroying... [id=sha256:edbf74c41f8a3581ce542e137cf28ea84dd93e6df8c9d66519b6ae761c2598a:latest]
docker_image.ubuntu: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
```


Docker images after executing the 'terraform destroy' command:-

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

Conclusion: This experiment demonstrates how to build, modify, and destroy infrastructure using Terraform by interacting with Docker containers. By creating a Terraform configuration file (docker.tf), initializing Terraform in the directory, and applying changes, users can pull Docker images and manage containers efficiently. The key commands—terraform init, terraform plan, terraform apply, and terraform destroy—enable infrastructure automation. This setup showcases Terraform's capability to manage Docker containers, making it a versatile tool for multi-cloud and containerized infrastructure management.

Experiment 7

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

Steps:

Step 1: Install a SonarQube image by running the 'docker pull sonarqube' command on your terminal. This allows for a Sonarqube image to be used on a local machine without having to install the SonarQube application.

```
PS C:\Users\anish\OneDrive\Desktop\Adv DevOps 7> docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
7478e8ac0f23: Pull complete
98a925ab929a: Pull complete
7d9a34308537: Pull complete
88338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecd
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest

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

Step 2: Execute the following command:

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

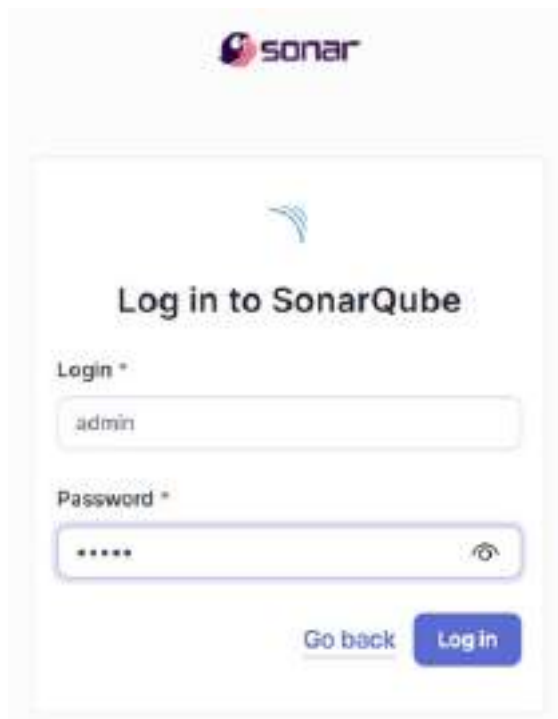
This command will run the SonarQube image that was just installed using docker.

```
PS C:\Users\anish\OneDrive\Desktop\Adv DevOps 7> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
dce67335999a42d81ec64d7ef0c3e3c16cc7ed36d87000033121ae1344f4fb
```

Step 3: Go to <http://localhost:9000> on your browser and check if SonarQube is starting or not.



Step 4: On the login page, enter 'Login' as admin and 'Password' as admin to log in initially. It then asks you to change the password to a password of your choice. Do the same and proceed to the next step.



Step 5: On the SonarQube dashboard, click on 'Create a local project'.



Step 6: Create a local project by entering the project name and key and click on 'Next'.

1 of 2

Create a local project

Project display name *

Project key *

Main branch name *

The name of your project's default branch. [Learn More](#)

Step 7: Set up your project and click on 'Create project'.

2 of 2

Set up project for Clean as You Code

The new code definition tells which part of your code will be considered new code. This helps you focus attention on the most recent changes to your project, enabling you to follow the Clean as You Code methodology. Learn more: [Defining New Code](#)

Choose the baseline for new code for this project

☒ Use the global setting

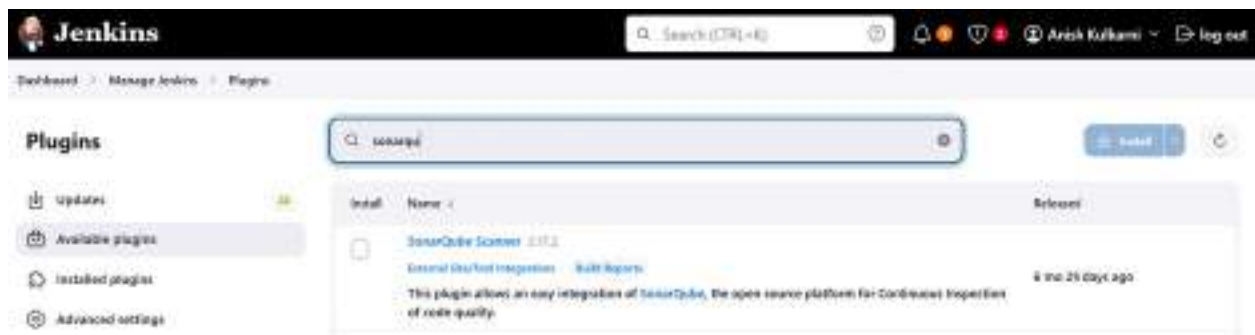
Previous version
Any code that has changed since the previous version is considered new code.
Recommended for projects following regular versions or releases.

☐ Define a specific setting for this project

☐ Previous version
Any code that has changed since the previous version is considered new code.
Recommended for projects following regular versions or releases.

☐ Number of days

Step 8: Navigate to your Jenkins server (on whichever port it has been installed), click on 'Manage Jenkins', click on 'Plugins' and search for the 'SonarQube Scanner' plugin and install it.



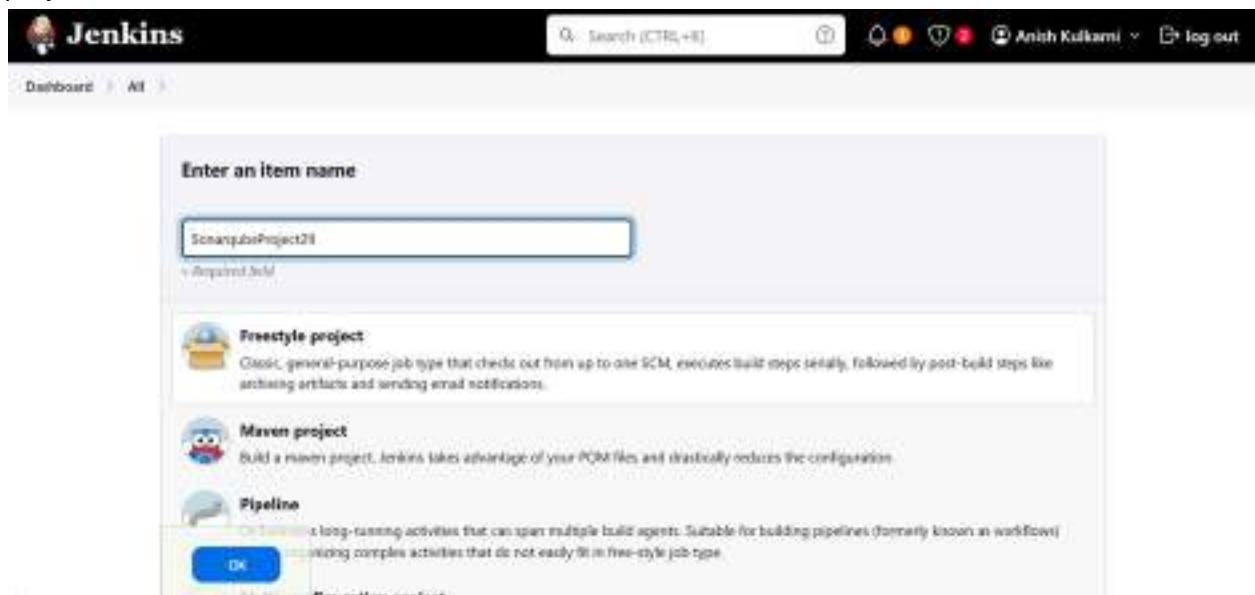
Step 9: Under 'Manage Jenkins', click on System. Under the 'Sonarqube installations' section, add a server and add a server authentication token if needed.



Step 10: Under 'Manage Jenkins', click on 'Tools'. Under the 'SonarQube Scanner installations' section, give your scanner a name, choose the latest version and click on 'Install automatically'.



Step 11: Create a new Jenkins project by giving it a name and ensure that it is a freestyle project.



Step 12: In 'Source Code Management' section, choose 'Git' and enter the following repository URL:-

https://github.com/shazforiot/MSBuild_firstproject

The above is a sample hello-world project with no vulnerabilities.

Source Code Management

☐ None

☒ Git

Repositories

Repository URL

Credentials

+ Add +

Advanced

Save Apply

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

Build Steps

Execute SonarQube Scanner

JDK

JDK to be used for this SonarQube analysis

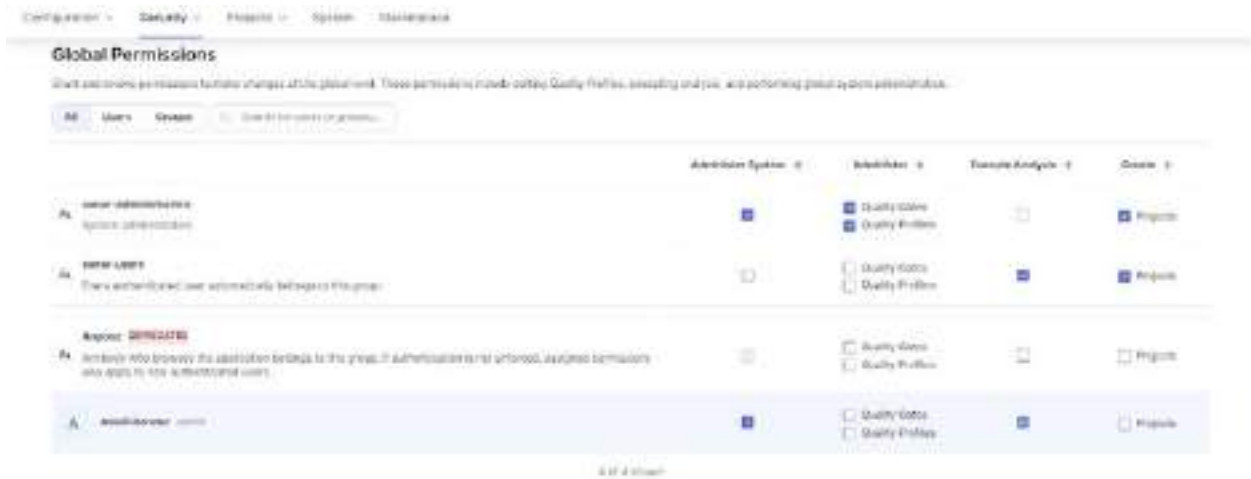
Path to project properties

Analysis properties

Additional arguments

Save Apply

Step 14: On your browser, go to http://localhost:<port_number>/admin/permissions and check the 'Execute Analysis' checkbox for Administrator. This gives the required permissions to the user for the analysis stage on SonarQube.



Step 15: Navigate to your Jenkins project and click on 'Build Now'.

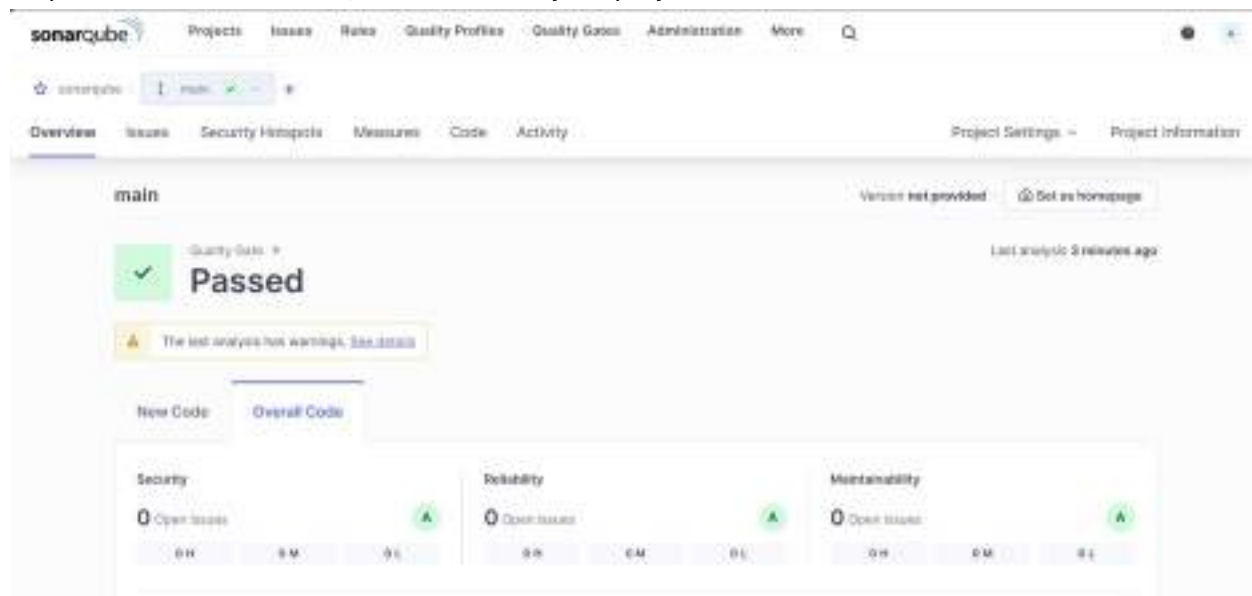


Step 16: Once the build is successful, check the console output.

```
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\jenkins\workspace\SonarqubeProject29
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\jenkins\workspace\SonarqubeProject29\.git # timeout=10
Fetching changes from the remote git repository
> git.exe config remote.origin.url http://github.com/shazforiot/MSBuild_FirstProject.git # timeout=10
Fetching upstream changes from http://github.com/shazforiot/MSBuild_FirstProject.git
> git.exe --version # timeout=10
> git --version # 'git version 2.46.0.windows.1'
> git.exe fetch --tags --force --progress -- http://github.com/shazforiot/MSBuild_FirstProject.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c0e72427c300bcae6d6fee7b49adf (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f f2bc042c04c0e72427c300bcae6d6fee7b49adf # timeout=30
Commit message: "updated"
> git.exe rev-list --no-walk f2bc042c04c0e72427c300bcae6d6fee7b49adf # timeout=10
[SonarqubeProject29] $ C:\ProgramData\Jenkins\jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\scanner29\bin\sonar-
scanner.bat -Dsonar.host.url=http://localhost:9000 -Dsonar.projectKey=sonarqube -Dsonar.host.url=http://localhost:9000 -
Dsonar.login=admin -Dsonar.sources=. -Dsonar.password=admin@15H2004 -
Dsonar.projectBaseDir=C:\ProgramData\Jenkins\jenkins\workspace\SonarqubeProject29
16:13:33.210 WARN Property 'sonar.host.url' with value 'http://localhost:9000' is overridden with value 'http://localhost:9000'
16:13:33.210 INFO Scanner configuration file:
C:\ProgramData\Jenkins\jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\scanner29\bin\..\conf\sonar-scanner.properties
16:13:33.246 INFO Project root configuration file: NONE

16:14:12.509 INFO Sensor C# [csharp] (done) | time=2ms
16:14:12.509 INFO Sensor Analysis Warnings Import [csharp]
16:14:12.509 INFO Sensor Analysis Warnings Import [csharp] (done) | time=0ms
16:14:12.509 INFO Sensor C# File Caching Sensor [csharp]
16:14:12.509 WARN Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting
'sonar.projectBaseDir' property.
16:14:12.509 INFO Sensor C# File Caching Sensor [csharp] (done) | time=0ms
16:14:12.509 INFO Sensor Zero Coverage Sensor
16:14:12.555 INFO Sensor Zero Coverage Sensor (done) | time=0ms
16:14:12.555 INFO SCM Publisher SCM provider for this project is git
16:14:12.567 INFO SCM Publisher 4 source files to be analyzed
16:14:13.100 INFO SCM Publisher 4/4 source files have been analyzed (done) | time=413ms
16:14:13.104 INFO CPD Executor Calculating CPD for 0 files
16:14:13.104 INFO CPD Executor CPD calculation finished (done) | time=0ms
16:14:13.101 INFO SCM revision ID 'f2bc042c04c0e72427c300bcae6d6fee7b49adf'
16:14:13.474 INFO Analysis report generated in 337ms, dir size=201.0 kB
16:14:13.522 INFO Analysis report compressed in 30ms, zip size=22.2 kB
16:14:13.750 INFO Analysis report uploaded in 233ms
16:14:13.759 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube-
16:14:13.759 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis
report
16:14:13.759 INFO Here about the report processing at http://localhost:9000/api/ci/task?id=f009ec1-0662-4115-a011-10cc01b3c001
16:14:13.787 INFO Analysis total time: 25.432 s
16:14:13.789 INFO SonarScanner Engine completed successfully
16:14:13.802 INFO EXECUTION SUCCESS
16:14:13.802 INFO Total time: 48.006s
Finished: SUCCESS
```

Step 17: Go back to SonarQube and check your project.



Conclusion: In this experiment, we learned how to integrate Jenkins SAST to SonarQube. We first used a docker image of SonarQube in order to avoid having to install it on our system. Next, SonarQube was configured and a SonarQube project was created. Then, required configurations were done on Jenkins and a Jenkins freestyle project was created which contained links to a code file from a Github repository and also our SonarQube project. When the Jenkins project was built, the SonarQube project displayed that there were no issues with the code in the Jenkins project.

Experiment 8

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

Steps:

Step 1: Install the SonarScanner CLI from the following link:

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



Once download is complete, extract the downloaded files into a folder.

Step 2: Install a SonarQube image by running the 'docker pull sonarqube' command on your terminal. This allows for a Sonarqube image to be used on a local machine without having to install the SonarQube application.

```
PS C:\Users\anish\OneDrive\Desktop\Adv DevOps 7> docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
7478e8ac0f23: Pull complete
80a925ab929a: Pull complete
7d9a34368537: Pull complete
89338217a4ab: Pull complete
3a5fd5c7e384: Pull complete
7b87d0fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb788ef54: Pull complete
Digest: sha256:72e9feec71242af83faf60f95a40d5a3bb2822a6c3b2cda8568790f3d31aeece
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest

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

Step 3: Execute the following command:

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

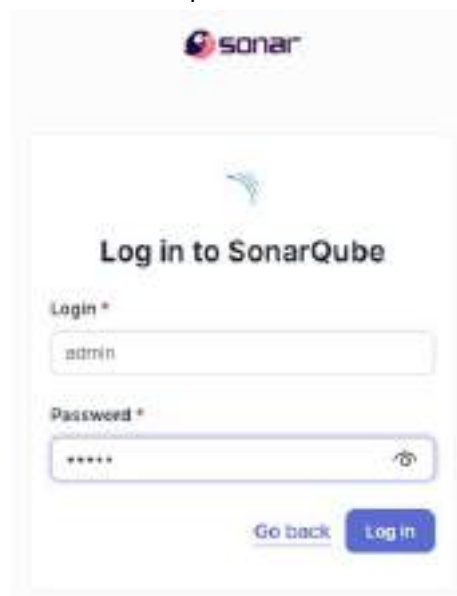
This command will run the SonarQube image that was just installed using docker.

```
PS C:\Users\anish\OneDrive\Desktop\Adv DevOps 7> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
dce6733909e42d81ec64d3ef9c5e3e2c36cc7ed3dd87680033121ae1346f4fb
```

Step 4: Go to <http://localhost:9000> on your browser and check if SonarQube is starting or not.



Step 5: On the login page, enter 'Login' as admin and 'Password' as admin to log in initially. It then asks you to change the password to a password of your choice. Do the same and proceed to the next step.



Step 6: On the SonarQube dashboard, click on 'Create a local project'.



Step 7: Create a local project by entering the project name and key and click on 'Next'.

1 of 2

Create a local project

Project display name *

Project key *

Main branch name *

The name of your project's default branch [Learn More](#)

Step 8: Set up your project and click on 'Create project'.

2 of 2

Set up project for Clean as You Code

The new code definition tells which part of your code will be considered new code. This helps you focus attention on the most recent changes to your project, enabling you to follow the Clean as You Code methodology. [Learn more: Defining New Code](#)

Choose the baseline for new code for this project:

☒ Use the global setting

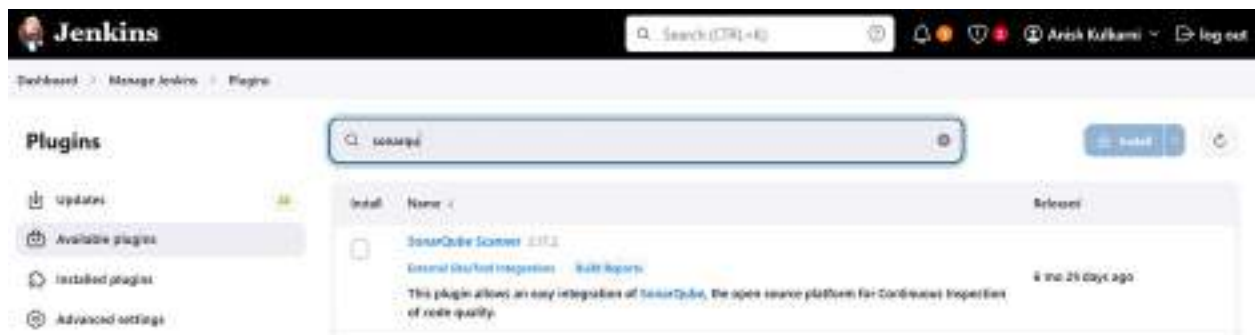
Previous version
Any code that has changed since the previous version is considered new code.
Recommended for projects following regular versions or releases.

☐ Define a specific setting for this project

☐ Previous version
Any code that has changed since the previous version is considered new code.
Recommended for projects following regular versions or releases.

☐ Number of days

Step 9: Navigate to your Jenkins server (on whichever port it has been installed), click on 'Manage Jenkins', click on 'Plugins' and search for the 'SonarQube Scanner' plugin and install it.



Step 10: Under 'Manage Jenkins', click on System. Under the 'Sonarqube installations' section, add a server and add a server authentication token if needed.

SonarQube installations

List of SonarQube installations

Name

sonarqube29

Server URL

Default is http://localhost:9000

http://localhost:9000

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

- none -

+ Add +

Save

Apply

Step 11: Under 'Manage Jenkins', click on 'Tools'. Under the 'SonarQube Scanner installations' section, give your scanner a name, choose the latest version and click on 'Install automatically'.

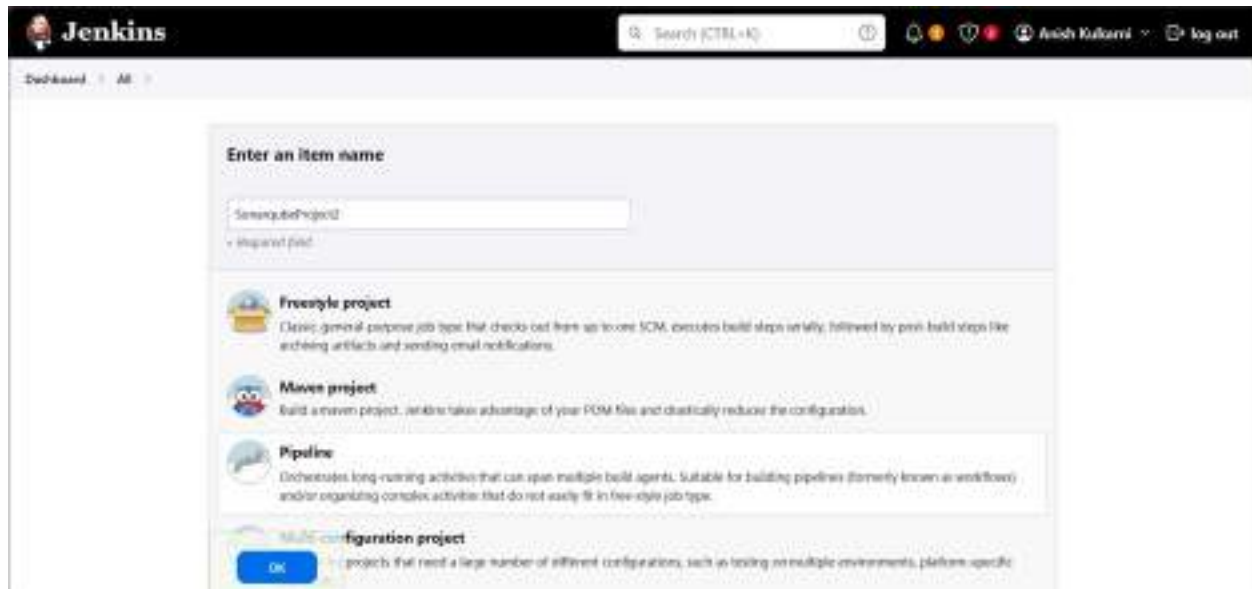


The screenshot shows the 'Add SonarQube Scanner' configuration page in Jenkins. At the top, there's a header 'SonarQube Scanner installations' with an 'Edit all' link. Below it, a form titled 'Add SonarQube Scanner' contains the following fields:

- Name:** A text input field containing 'SCANNER02'.
- Install automatically:** A checked checkbox with a help icon.
- Install from Maven Central:** A sub-section containing:
 - Version:** A dropdown menu showing 'SonarQube Scanner 6.10.0.8577'.
 - Add Installer:** A button.

At the bottom of the form are two buttons: 'Save' (in blue) and 'Apply' (in grey).

Step 12: Create a new Jenkins project by giving it a name and ensure that it is a pipeline project.



The screenshot shows the Jenkins 'Create new item' page. The top navigation bar includes the Jenkins logo, a search bar, and user information. The main content area has a header 'Enter an item name' with a text input field containing 'SonarqubeProject2'. Below this, there are four project type options, each with an icon and a description:

- Freestyle project:** Classic general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
- Maven project:** Build a maven project, Jenkins takes advantage of your POM file and drastically reduces the configuration.
- Pipeline:** Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Configuration project:** Projects that need a large number of different configurations, such as testing on multiple environments, platform-specific.

At the bottom left of the project type list is a blue 'OK' button.

Step 13: Under the 'Pipeline Script' section, enter the following:-

```
node {  
    stage('Cloning the GitHub Repo') {  
        git 'https://github.com/shazforiot/GOL.git'  
    }  
  
    stage('SonarQube analysis') {  
        withSonarQubeEnv('sonarqube29') {  
            bat """  
            <PATH_TO_SONARSCANNER_FOLDER>\\bin\\sonar-scanner.bat ^  
            -D sonar.login=<SONARQUBE_LOGIN> ^  
            -D sonar.password=<SONARQUBE_PASSWORD> ^  
            -D sonar.projectKey=<PROJECT_KEY> ^  
            -D sonar.exclusions=vendor/**,resources/**,**/*.java ^  
            -D sonar.host.url=http://localhost:9000/  
            """  
        }  
    }  
}
```



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

Step 14: Go back to Jenkins, navigate to your Jenkins project and click on 'Build Now'.

Stage View



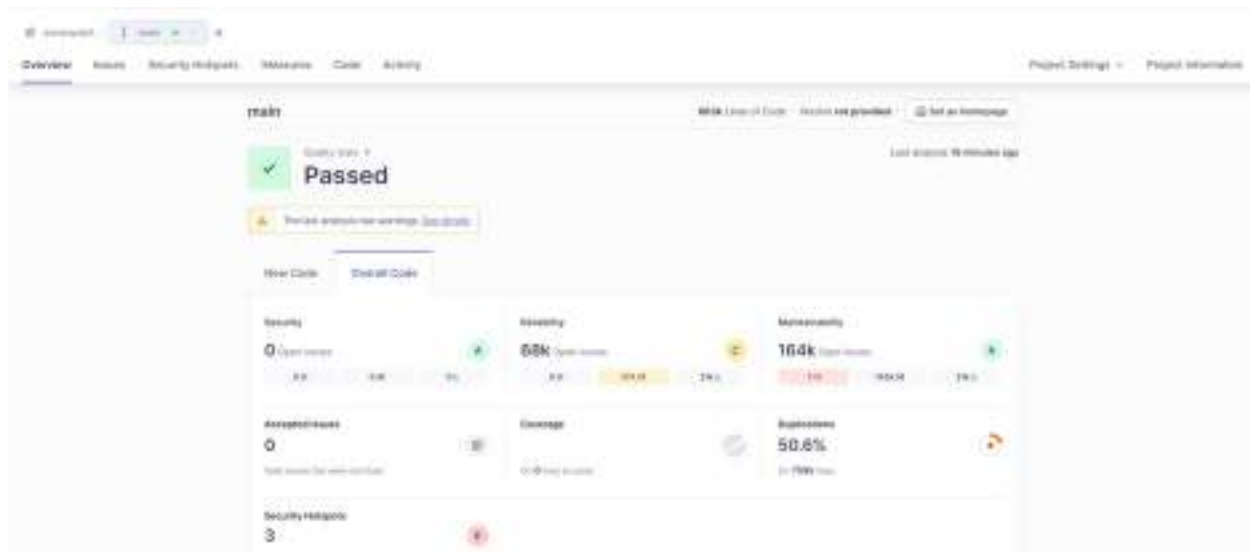
Step 15: Once build is successfully completed, check the console output.

```

Dashboard > SonargraphProject2 > #30
20:28:17.898 WARN too many duplication references in file gsmc01ife-
webtools/jmeter/docs/api/engr/apache/jmeter/protocol/http/sampler/HTTPSamplerBase.html for block at line 4737. keep only the first 100
references.
20:28:17.898 WARN too many duplication references in file gsmc01ife-
webtools/jmeter/docs/api/engr/apache/jmeter/protocol/http/sampler/HTTPSamplerBase.html for block at line 45. keep only the first 100
references.
20:28:17.904 INFO O96 Executor O96 calculation finished (done) : time=12123ms
20:28:18.898 INFO SON revision is 'b49bba5400979f84a412220882170e6e6ed'
20:27:04.340 INFO Analysis report generated in 4318ms, dir size=127.3 KB
20:27:06.181 INFO Analysis report compressed in 3088ms, zip size=28.4 KB
20:28:06.003 INFO Analysis report uploaded in 400ms
20:28:06.525 INFO NOW READY SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonargraph
20:28:06.525 INFO Note that you will be able to access the updated Dashboard once the server has processed the submitted analysis
report
20:28:06.825 INFO Here about the report processing at: http://localhost:9000/webhook?id=sonargraph-son-2020-09-15-09-05-30-054
20:28:06.994 INFO Analysis total time: 41112ms
20:28:26.826 INFO SonarScanner engine completed successfully
20:28:27.340 INFO EXECUTION SUCCESS
20:28:27.934 INFO Total time: 15108200ms
[Pipeline] }
[Pipeline] // withMaven-jar
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] end of pipeline
Finished: SUCCESS

```

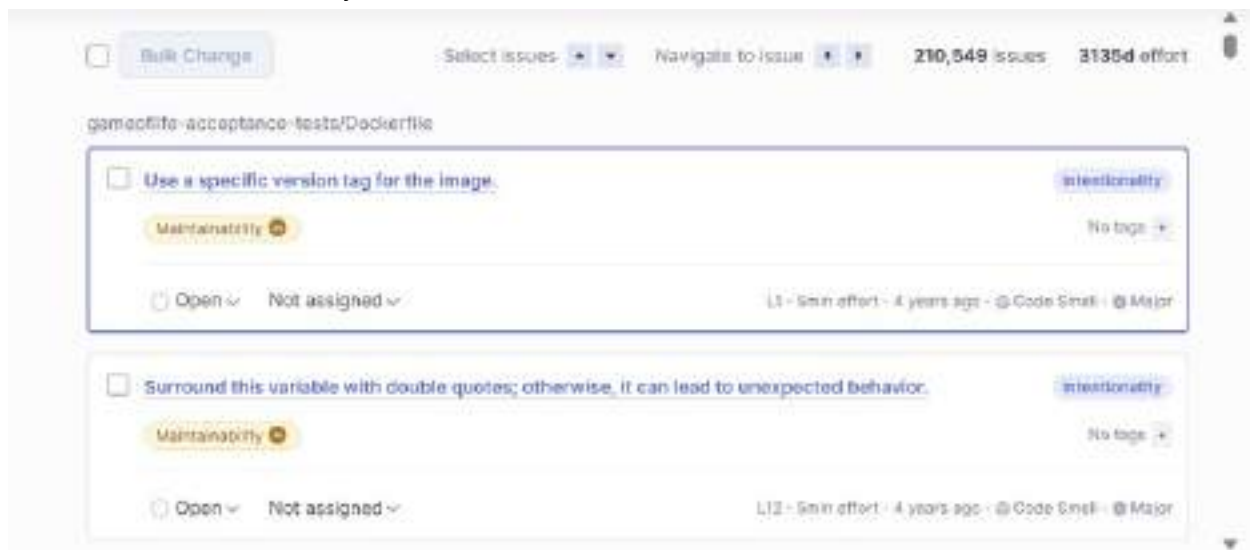
Step 16: Go back to SonarQube and check your project.



Step 17: Check the different types of issues with the code:-

● Codeproblems:-

● Intentionality:-



● Consistency:-

This screenshot displays three accessibility issues from a tool like Lighthouse. Each issue is a checkbox item with a title, a category tag, a sub-category tag, and a status. The first two issues are identical: 'Remove this deprecated "align" attribute' with a 'Consistency' tag, 'html5 - obsolete' sub-tag, and a status of 'Open' and 'Not assigned'. The third issue is 'Remove this deprecated "size" attribute' with a 'Consistency' tag, 'html5 - obsolete' sub-tag, and a status of 'Open' and 'Not assigned'. All three issues are marked as 'Major' and have a '3min effort' estimate.

- ☐ Remove this deprecated "align" attribute. Consistency
html5 - obsolete
Open Not assigned L11 - 3min effort - 4 years ago - @ Code Smell - @ Major
- ☐ Remove this deprecated "align" attribute. Consistency
html5 - obsolete
Open Not assigned L13 - 3min effort - 4 years ago - @ Code Smell - @ Major
- ☐ Remove this deprecated "size" attribute. Consistency
html5 - obsolete
Open Not assigned L12 - 3min effort - 4 years ago - @ Code Smell - @ Major

● Bugs and Code Smells:-

This screenshot displays five accessibility issues. The first two are bugs: 'Add "lang" and/or "xml:lang" attributes to this "<html>" element' (Intentionality, accessibility, wcag2-a, Bug, Major, 2min effort) and 'Insert a <!DOCTYPE> declaration to before this <html> tag.' (Consistency, user-experience, Bug, Major, 5min effort). The next two are code smells: 'Remove this deprecated "valign" attribute' (Consistency, html5 - obsolete, Code Smell, Major, 5min effort) and 'Remove this deprecated "name" attribute.' (Consistency, html5 - obsolete, Code Smell, Major, 5min effort). The fifth issue is a general guideline: 'Anchors must have content and the content must be accessible by a screen reader.' (Consistency). All issues are marked as 'Open' and 'Not assigned'.

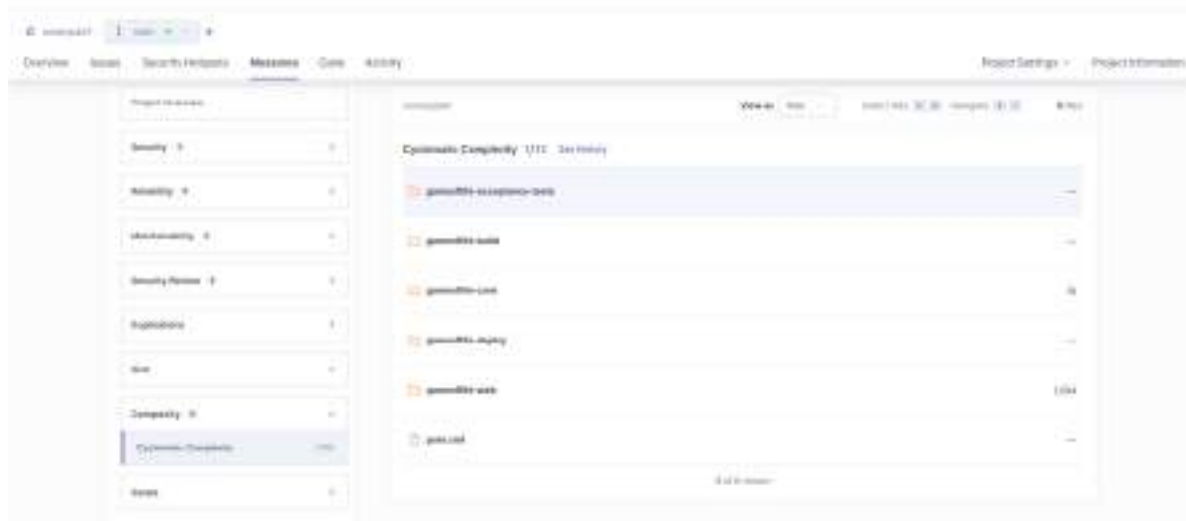
gameoflife-console/build/reports/tests/allclasses-frame.html

- ☐ Add "lang" and/or "xml:lang" attributes to this "<html>" element. Intentionality
accessibility - wcag2-a
Open Not assigned L1 - 2min effort - 4 years ago - @ Bug - @ Major
- ☐ Insert a <!DOCTYPE> declaration to before this <html> tag. Consistency
user-experience
Open Not assigned L1 - 5min effort - 4 years ago - @ Bug - @ Major
- ☐ Remove this deprecated "valign" attribute. Consistency
html5 - obsolete
Open Not assigned L627 - 5min effort - 4 years ago - @ Code Smell - @ Major
- ☐ Remove this deprecated "name" attribute. Consistency
html5 - obsolete
Open Not assigned L438 - 5min effort - 4 years ago - @ Code Smell - @ Major
- ☐ Anchors must have content and the content must be accessible by a screen reader. Consistency

● Duplications:-



● Cyclomatic complexities:-



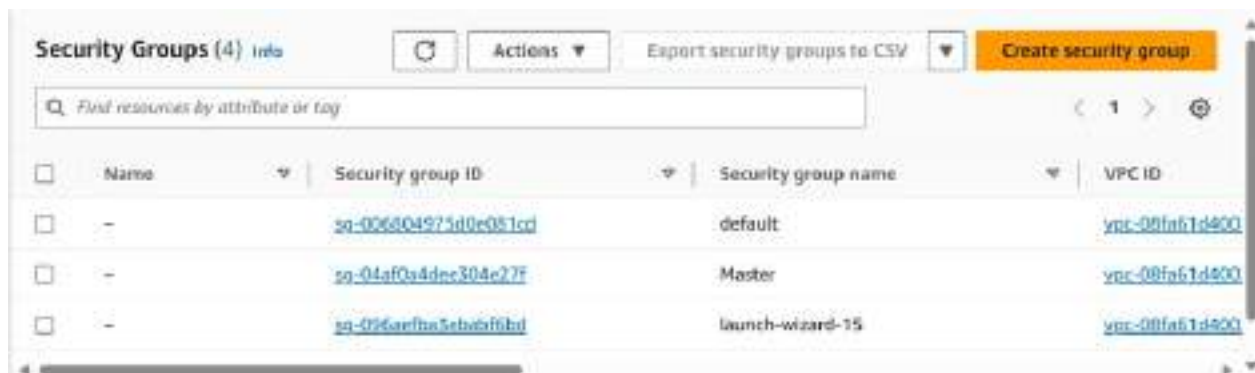
Conclusion: In this experiment, we learned how to create a Jenkins CI/CD Pipeline with SonarQube integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Java application. A pipeline project is created in Jenkins and a pipeline script contains the link to the Java application on which the SonarQube analysis is to be done. Then the pipeline project is configured as per needs and built. The SonarQube project linked to the pipeline project then successfully does the SonarQube analysis and points out all the issues, bugs, duplications etc in the pipeline project.

Experiment 9

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

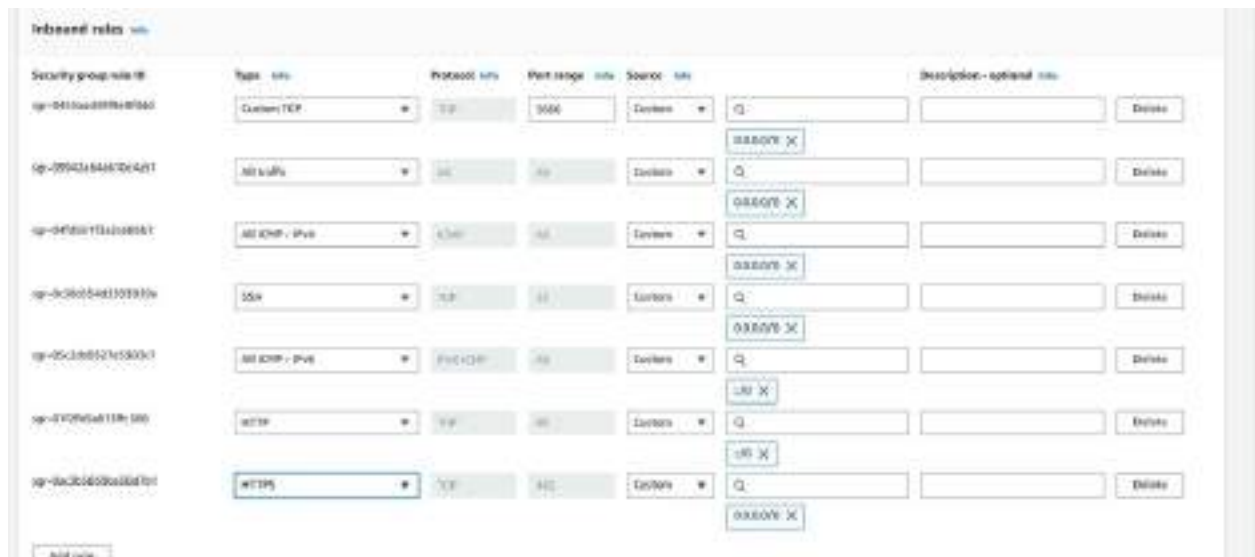
Steps:

Step 1: Navigate to the EC2 section on your AWS console using the 'Services' section. Then, from the options in the left-side panel, click on 'Security groups'. Next, click on 'Create security group'.

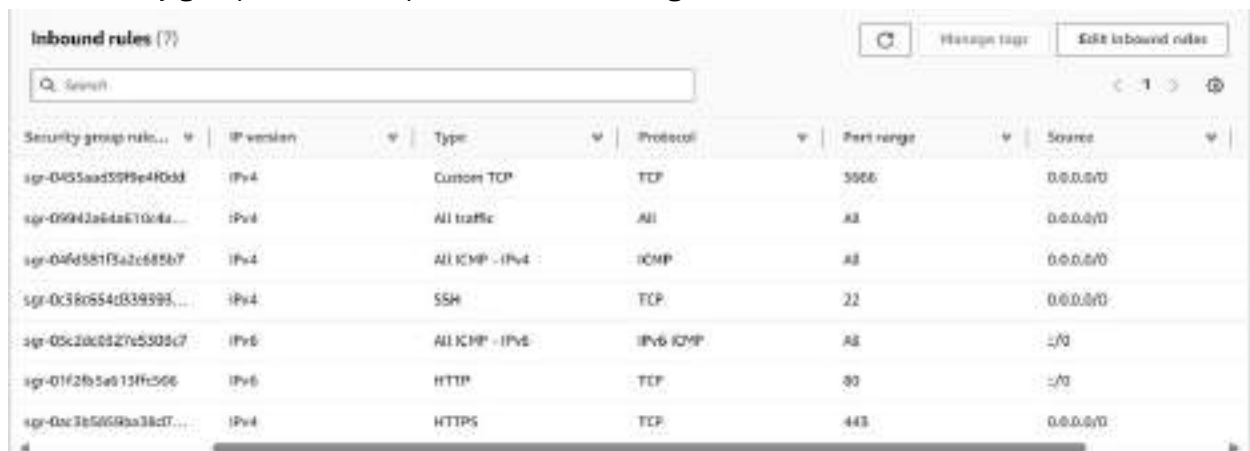


Give your security group a name (here, the name is launch-wizard-15) and then in the 'Inbound rules' section, click on 'Edit'. Then, click on add rules, and add the rules for the following protocols:

HTTP, All ICMP - IPv6, HTTPS, All traffic, Custom TCP (Port 5666), All ICMP - IPv4



Your security group with the required inbound rules gets created as such:-



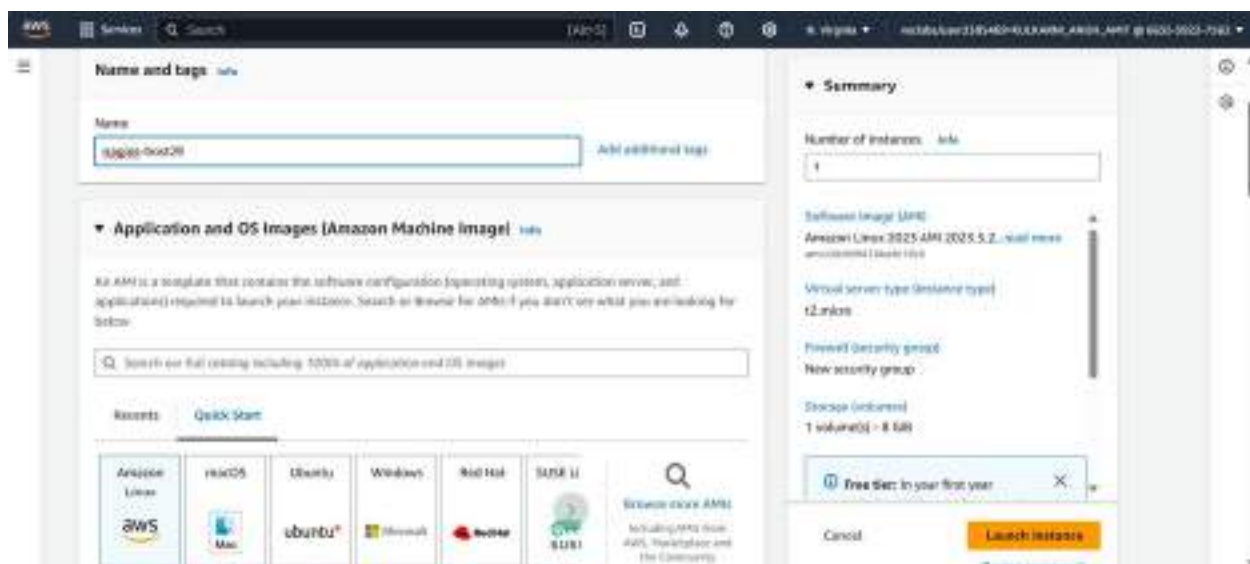
The screenshot shows the 'Inbound rules' section of an AWS Security Group. It contains a table with 7 rules. The columns are: Security group rule ID, IP version, Type, Protocol, Port range, and Source. The rules are as follows:

Security group rule ID	IP version	Type	Protocol	Port range	Source
sg-0455aad39f9e4f0dd	IPv4	Custom TCP	TCP	3388	0.0.0.0/0
sg-099d2a64a610cfa...	IPv4	All traffic	All	All	0.0.0.0/0
sg-04fd581f3a2c685b7	IPv4	All ICMP - IPv4	ICMP	All	0.0.0.0/0
sg-0c38c54c8395993...	IPv4	SSH	TCP	22	0.0.0.0/0
sg-05c2dc0327c5305c7	IPv6	All ICMP - IPv6	IPv6 ICMP	All	::/0
sg-01f2b5a013ff566	IPv6	HTTP	TCP	80	::/0
sg-0ac3e5669ba38c7...	IPv4	HTTPS	TCP	443	0.0.0.0/0

Step 2: Navigate to the EC2 section and click on 'Launch instances'.



Give your instance a name, choose 'Amazon Linux' as the instance type, insert the key pair for which you have the .pem file available in the 'Key pair' section, choose the security group that you created in Step 1 in the 'Network settings' section, keep all other options as default and click on 'Launch instance'.



Connect to instance [Info](#)

Connect to your instance i-031e03391a2b1030 (Amazon Linux 2) using any of these options:

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

Instance ID
i-031e03391a2b1030 (Amazon Linux 2)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is AdvDevopsLab.pem.
3. Run this command, if necessary, to ensure your key is not publicly available:
`ssh-keygen -f AdvDevopsLab.pem`
4. Connect to your instance using its Public DNS:
`ssh -i AdvDevopsLab.pem ec2-user@ec2-3-64-16-157.compute-1.amazonaws.com`

Example:
`ssh -i "AdvDevopsLab.pem" ec2-user@ec2-3-64-16-157.compute-1.amazonaws.com`

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

[illegible]

Step 5: First, run the following command:-
`sudo yum update`
This command will check for any updates for the YUM library.

```
[ec2-user@ip-172-31-88-33 ~]$ sudo yum update
Last metadata expiration check: 0:02:17 ago on Sun Sep 29 10:22:03 2024.
Dependencies resolved.
Nothing to do.
Complete!
```


Step 6: Run the command:

sudo yum install httpd php

This installs an Apache server and a PHP on your instance.

```
[ec2-user@ip-172-31-88-33 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:04:36 ago on Sun Sep 29 10:22:03 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        18 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            10.0.0-12.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
libssl1                                x86_64            1.1.34-5.amzn2023.0.2    amazonlinux        241 k
mod_ssl                                noarch            2.2.49-3.amzn2023.0.3    amazonlinux        33 k
nginx-filesystem                       noarch            1:1.24.0-1.amzn2023.0.4    amazonlinux        9.8 k

Installed:
apr-1.7.2-2.amzn2023.0.2.x86_64
apr-util-1.6.3-1.amzn2023.0.1.x86_64
httpd-2.4.62-1.amzn2023.x86_64
httpd-filesystem-2.4.62-1.amzn2023.noarch
libbrotli-1.0.9-4.amzn2023.0.2.x86_64
libssl1-1.1.34-5.amzn2023.0.2.x86_64
mod_http2-2.0.27-1.amzn2023.0.3.x86_64
nginx-filesystem-1:1.24.0-1.amzn2023.0.4.noarch
php8.3-cli-8.3.10-1.amzn2023.0.1.x86_64
php8.3-fpm-8.3.10-1.amzn2023.0.1.x86_64
php8.3-opcache-8.3.10-1.amzn2023.0.1.x86_64
php8.3-process-8.3.10-1.amzn2023.0.1.x86_64
php8.3-xsl-8.3.10-1.amzn2023.0.1.x86_64
apr-util-1.6.3-1.amzn2023.0.1.x86_64
generic-logos-httpd-10.0.0-12.amzn2023.0.3.noarch
httpd-core-2.4.62-1.amzn2023.x86_64
httpd-tools-2.4.62-1.amzn2023.x86_64
libsodium-1.0.19-4.amzn2023.x86_64
mod_ssl-2.2.49-3.amzn2023.0.3.noarch
mod_lua-2.4.62-1.amzn2023.x86_64
php8.3-8.3.10-1.amzn2023.0.1.x86_64
php8.3-common-8.3.10-1.amzn2023.0.1.x86_64
php8.3-embed-8.3.10-1.amzn2023.0.1.x86_64
php8.3-pdo-8.3.10-1.amzn2023.0.1.x86_64
php8.3-sodium-8.3.10-1.amzn2023.0.1.x86_64

Complete!
```

Step 7: Run the command:

sudo yum install gcc glibc glibc-common

This installs the C/C++ compiler (GCC) along with the necessary C libraries required for compiling and running C programs.

```
[ec2-user@ip-172-31-88-33 ~]$ sudo yum install gcc glibc glibc-common
Last metadata expiration check: 0:05:41 ago on Sun Sep 29 10:22:03 2024.
Package glibc-2.34-52.amzn2023.0.11.x86_64 is already installed.
Package glibc-common-2.34-52.amzn2023.0.11.x86_64 is already installed.
Dependencies resolved.
=====
Package                                Architecture      Version           Repository        Size
=====
Installing:
gcc                                     x86_64            11.4.1-2.amzn2023.0.2    amazonlinux        32 M
Installing dependencies:
annobin-docs                           noarch            10.93-1.amzn2023.0.1    amazonlinux        92 k
annobin-plugin-gcc                     x86_64            10.93-1.amzn2023.0.1    amazonlinux        887 k
cpp                                     x86_64            11.4.1-2.amzn2023.0.2    amazonlinux        10 M
gc                                       x86_64            8.0.4-5.amzn2023.0.2    amazonlinux        166 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
glibc22                                 x86_64            2.2.7-2.amzn2023.0.3    amazonlinux        6.4 M
kernel-headers                          x86_64            6.1.109-118.109.amzn2023    amazonlinux        1.4 M
libgcc                                  x86_64            1.2.1-2.amzn2023.0.2    amazonlinux        62 k
libtool-ltdl                            x86_64            2.4.7-1.amzn2023.0.3    amazonlinux        38 k
libxcrypt-devel                         x86_64            4.4.33-7.amzn2023      amazonlinux        32 k
make                                    x86_64            1:4.3-5.amzn2023.0.2    amazonlinux        534 k

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

```

Installed:
  annobin-does-10.93-1.amzn2023.0.1.noarch
  cpp-11.4.1-2.amzn2023.0.2.x86_64
  gcc-11.4.1-2.amzn2023.0.2.x86_64
  glibc-headers-x86-2.34-52.amzn2023.0.11.noarch
  kernel-headers-6.1.109-118.189.amzn2023.x86_64
  libtool-ltdl-2.4.7-1.amzn2023.0.3.x86_64
  make-1:4.3-5.amzn2023.0.2.x86_64
  annobin-plugin-gcc-10.93-1.amzn2023.0.1.x86_64
  gc-8.0.4-5.amzn2023.0.2.x86_64
  glibc-devel-2.34-52.amzn2023.0.11.x86_64
  guile22-2.2.9-2.amzn2023.0.3.x86_64
  libipc-1.2.1-2.amzn2023.0.2.x86_64
  libxcrypt-devel-4.4.33-7.amzn2023.x86_64

Complete!
[ec2-user@ip-172-31-88-33 ~]$

```

Step 8: Run the command:
 sudo yum install gd gd-devel

```

[ec2-user@ip-172-31-88-33 ~]$ sudo yum install gd gd-devel
Last metadata expiration check: 0:06:51 ago on Sun Sep 29 10:22:03 2024.
Dependencies resolved.
=====
Package                                Architecture      Version            Repository          Size
=====
Installing:
  gd                                    x86_64            2.3.3-5.amzn2023.0.3  amazonlinux        139 k
  gd-devel                             x86_64            2.3.3-5.amzn2023.0.3  amazonlinux         38 k
Installing dependencies:
  brotli                                x86_64            1.0.9-4.amzn2023.0.2  amazonlinux        314 k
  brotli-devel                          x86_64            1.0.9-4.amzn2023.0.2  amazonlinux         31 k
  brotli-devel                          x86_64            1.0.9-6.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        120 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
  libffi-devel-3.4.4-1.amzn2023.0.1.x86_64
  libicu-devel-67.1-7.amzn2023.0.3.x86_64
  libjpeg-turbo-devel-2.1.4-2.amzn2023.0.5.x86_64
  libpng-2:1.6.37-10.amzn2023.0.6.x86_64
  libselinux-devel-3.4-5.amzn2023.0.2.x86_64
  libtiff-4.4.0-4.amzn2023.0.18.x86_64
  libwebp-1.2.4-1.amzn2023.0.6.x86_64
  libxcb-1.13.1-7.amzn2023.0.2.x86_64
  libxml2-devel-2.10.4-1.amzn2023.0.6.x86_64
  pcre2-utf16-10.40-1.amzn2023.0.3.x86_64
  pixman-0.40.0-3.amzn2023.0.3.x86_64
  xsl-common-0.6.3-56.amzn2023.0.2.noarch
  xz-devel-5.2.5-9.amzn2023.0.2.x86_64
  libicu-67.1-7.amzn2023.0.3.x86_64
  libjpeg-turbo-2.1.4-2.amzn2023.0.5.x86_64
  libmount-devel-2.37.4-1.amzn2023.0.4.x86_64
  libpng-devel-2:1.6.37-10.amzn2023.0.6.x86_64
  libsepol-devel-3.4-3.amzn2023.0.3.x86_64
  libtiff-devel-4.4.0-4.amzn2023.0.18.x86_64
  libwebp-devel-1.2.4-1.amzn2023.0.6.x86_64
  libxcb-devel-1.13.1-7.amzn2023.0.2.x86_64
  pcre2-devel-10.40-1.amzn2023.0.3.x86_64
  pcre2-utf32-10.40-1.amzn2023.0.3.x86_64
  sysprof-capture-devel-3.60.1-2.amzn2023.0.2.x86_64
  xorg-x11-proto-devel-2021.4-1.amzn2023.0.2.noarch
  zlib-devel-1.2.11-33.amzn2023.0.5.x86_64

Complete!
[ec2-user@ip-172-31-88-33 ~]$

```

Step 9: Run the commands:
 sudo adduser -m nagios
 sudo passwd nagios
 This creates a user named 'nagios', ensures it has a home directory and sets up a password for it.

```

[ec2-user@ip-172-31-88-33 ~]$ sudo adduser -m nagios
sudo passwd nagios
Changing password for user nagios.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.

```

Step 10: Create a user group named 'nagcmd' to execute nagios commands.

sudo groupadd nagcmd

```
[ec2-user@ip-172-31-88-33 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-88-33 ~]$ |
```

Step 11: Add users apache and nagios to this user group.

sudo usermod -a -G nagcmd nagios

sudo usermod -a -G nagcmd apache

```
[ec2-user@ip-172-31-88-33 ~]$ sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache
[ec2-user@ip-172-31-88-33 ~]$ |
```

Step 12: mkdir ~/downloads

cd ~/downloads

This creates a directory named 'downloads', to store the files of the nagios server that are downloaded.

```
[ec2-user@ip-172-31-88-33 ~]$ mkdir ~/downloads
cd ~/downloads
[ec2-user@ip-172-31-88-33 downloads]$ |
```

Step 13: wget <https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz>

The above command installs the latest version of nagios-core.

```
[ec2-user@ip-172-31-88-33 downloads]$ wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
--2024-09-29 10:33:56-- https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.128, 2600:3c00::f03c:92ff:fef7:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.128|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: 'nagios-4.5.5.tar.gz'

nagios-4.5.5.tar.gz 100%[=====] 1.97M 8.29MB/s in 0.2s
2024-09-29 10:33:57 (8.29 MB/s) - 'nagios-4.5.5.tar.gz' saved [2065473/2065473]
```

Step 14: wget <https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz>

The above command installs the latest version of nagios-plugins.

```
[ec2-user@ip-172-31-88-33 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
--2024-09-29 10:34:29-- https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2753049 (2.6M) [application/x-gzip]
Saving to: 'nagios-plugins-2.4.11.tar.gz'

nagios-plugins-2.4.11.tar.gz 100%[=====] 2.62M 9.99MB/s in 0.3s
2024-09-29 10:34:30 (9.99 MB/s) - 'nagios-plugins-2.4.11.tar.gz' saved [2753049/2753049]
```

Step 15: tar zxvf nagios-4.5.5.tar.gz

This extracts the nagios-core files into the same directory using the tar command.

```
[ec2-user@ip-172-31-88-33 downloads]$ tar zxvf nagios-4.5.5.tar.gz
nagios-4.5.5/
nagios-4.5.5/.github/
nagios-4.5.5/.github/workflows/
nagios-4.5.5/.github/workflows/test.yml
nagios-4.5.5/.gitignore
nagios-4.5.5/CONTRIBUTING.md
nagios-4.5.5/ChangeLog
nagios-4.5.5/INSTALLING
nagios-4.5.5/LICENSE
nagios-4.5.5/Makefile.in
nagios-4.5.5/README.md
nagios-4.5.5/TIPS
nagios-4.5.5/UPGRADING
nagios-4.5.5/xdata/
nagios-4.5.5/xdata/.gitignore
nagios-4.5.5/xdata/Makefile.in
nagios-4.5.5/xdata/xcddefault.c
nagios-4.5.5/xdata/xcddefault.h
nagios-4.5.5/xdata/xodtemplate.c
nagios-4.5.5/xdata/xodtemplate.h
nagios-4.5.5/xdata/xpddefault.c
nagios-4.5.5/xdata/xpddefault.h
nagios-4.5.5/xdata/xrddefault.c
nagios-4.5.5/xdata/xrddefault.h
nagios-4.5.5/xdata/xsddefault.c
nagios-4.5.5/xdata/xsddefault.h
```

```
nagios-4.5.5/xdata/.gitignore
nagios-4.5.5/xdata/Makefile.in
nagios-4.5.5/xdata/xcddefault.c
nagios-4.5.5/xdata/xcddefault.h
nagios-4.5.5/xdata/xodtemplate.c
nagios-4.5.5/xdata/xodtemplate.h
nagios-4.5.5/xdata/xpddefault.c
nagios-4.5.5/xdata/xpddefault.h
nagios-4.5.5/xdata/xrddefault.c
nagios-4.5.5/xdata/xrddefault.h
nagios-4.5.5/xdata/xsddefault.c
nagios-4.5.5/xdata/xsddefault.h
[ec2-user@ip-172-31-88-33 downloads]$ |
```

Step 16: ./configure --with-command-group=nagcmd

This command ensures that Nagios uses a specific group (in this case, nagcmd) for executing external commands.

```
[ec2-user@ip-172-31-88-33 downloads]$ ./configure --with-command-group=nagcmd
-bash: ./configure: No such file or directory
```

But, we encounter an error as we weren't in the correct directory.

Use 'ls' command to find the correct directory.

```
[ec2-user@ip-172-31-88-33 downloads]$ ls
nagios-4.5.5  nagios-4.5.5.tar.gz  nagios-plugins-2.4.11.tar.gz
```

Use cd to change directory to the correct directory. Then, run the './configure --with-command-group=nagcmd' command again.

```
[ec2-user@ip-172-31-88-33 downloads]$ cd nagios-4.5.5
[ec2-user@ip-172-31-88-33 nagios-4.5.5]$ ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether the compiler supports GNU C... yes
checking whether gcc accepts -g... yes
checking for gcc option to enable C11 features... none needed
checking whether make sets $(MAKE)... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking for sys/wait.h that is POSIX.1 compatible... yes
checking for stdio.h... yes
checking for stdlib.h... yes
checking for string.h... yes
checking for inttypes.h... yes
checking for unistd.h... yes
checking for unsetenv... yes
checking for type of socket size... size_t
checking for Kerberos include files... configure: WARNING: could not find include files
checking for pkg-config... pkg-config
checking for SSL headers... configure: error: Cannot find ssl headers
```

Another error occurs which says that ssl headers cannot be found.

To fix the above error, run the 'sudo yum install openssl-devel' command.

```
[ec2-user@ip-172-31-88-33 nagios-4.5.5]$ sudo yum install openssl-devel
Last metadata expiration check: 0:35:17 ago on Sun Sep 29 10:22:03 2024.
Dependencies resolved.
=====
Package                                Architecture      Version           Size
=====
Installing:
openssl-devel                          x86_64            1:3.0.8-1.amzn2023.0.14 3.0 M
Transaction Summary
=====
Install 1 Package

Total download size: 3.0 M
Installed size: 4.7 M
Is this ok [y/N]: y
Downloading Packages:
openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64.rpm                                26 MB/s | 3.0 MB  00:00
Total
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      : 1/1
  Installing     : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64 1/1
  Running scriptlet: openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64 1/1
  Verifying      : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64 1/1
```

```
Installed:
  openssl-devel-1:1.0.8-1.amzn2023.0.14.x86_64

Complete!
```

Then, run the './configure --with-command-group=nagcmd' command again.

```
[ec2-user@ip-172-31-88-33 nagios-4.5.5]$ ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether the compiler supports GNU C... yes
checking whether gcc accepts -g... yes
checking for gcc option to enable C11 features... none needed
checking whether make sets $(MAKE)... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking for sys/wait.h that is POSIX.1 compatible... yes
checking for stdio.h... yes
checking for stdlib.h... yes
checking for string.h... yes
checking for inttypes.h... yes
checking forstdint.h... yes
```

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

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

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


```
sudo make install-commandmode
```

```
[ec2-user@ip-172-31-88-33 nagios-4.5.5]$
```

```
[ec2-user@ip-172-31-00-13 nagios-4.5.5]$ sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode
cd ./base && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagios /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagiosstats /usr/local/nagios/bin
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5/base'
cd ./cgi && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
make install-basic
make[2]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/sbin
/usr/bin/install -c -m 664 -o nagios -g nagios sample-config/template-object/switch.cfg /usr/local/nagios/etc/objects/switch.cfg

*** Config files installed ***

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

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

*** External command directory configured ***
```

Step 18: We need to update the email linked with this server to our email for it to send notifications (if any needed).

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

```
GNU nano 2.8 /usr/local/nagios/etc/objects/contacts.cfg Modified

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

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

define contact {

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

}

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

Help:  Write Out  Where Is  Cut      Execute  Location  Undo     Set Mark
Exit:  Read File  Replace  Paste   Justify  Go-To Line Redo     Copy
```

In the email section, enter your email address. Then, 'Write out' your file and 'Exit'.

Step 19: sudo make install-webconf

This installs the necessary configuration files for the Nagios web interface.

```
[ec2-user@ip-172-31-88-33 nagios-4.5.5]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios.conf
if [ 0 -eq 1 ]; then \
    ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagios.conf; \
fi

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

Step 20: sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin

This creates a user named 'nagiosadmin' to access the nagios web interface. Create a password and keep it in mind as it will be required in the future steps.

```
[ec2-user@ip-172-31-88-33 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
```

Step 21: Restart the apache server to apply all the recent configurations.

sudo service httpd restart

```
[ec2-user@ip-172-31-88-33 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
```

Step 22: cd ~/downloads

tar xzvf nagios-plugins-2.4.11.tar.gz

This changes the directory to the 'downloads' directory and extracts the files for nagios-plugins.

```
[ec2-user@ip-172-31-88-33 downloads]$ tar xzvf nagios-plugins-2.4.11.tar.gz
nagios-plugins-2.4.11/
nagios-plugins-2.4.11/build-was/
nagios-plugins-2.4.11/build-was/compile
nagios-plugins-2.4.11/build-was/config.guess
nagios-plugins-2.4.11/build-was/config.rpath
nagios-plugins-2.4.11/build-was/config.sub
nagios-plugins-2.4.11/build-was/install-sh
nagios-plugins-2.4.11/build-was/ltmain.sh
nagios-plugins-2.4.11/build-was/missing
nagios-plugins-2.4.11/build-was/mkinstalldirs
nagios-plugins-2.4.11/build-was/depcomp
nagios-plugins-2.4.11/build-was/snippet/
nagios-plugins-2.4.11/build-was/snippet/_Noreturn.h
nagios-plugins-2.4.11/build-was/snippet/arg-nonnull.h
nagios-plugins-2.4.11/build-was/snippet/c++defs.h
nagios-plugins-2.4.11/build-was/snippet/warn-on-use.h
nagios-plugins-2.4.11/build-was/test-driver
nagios-plugins-2.4.11/config_test/
nagios-plugins-2.4.11/config_test/Makefile
```

```
nagios-plugins-2.4.11/po/
nagios-plugins-2.4.11/po/Makefile.in.in
nagios-plugins-2.4.11/po/remove-potdate.sh
nagios-plugins-2.4.11/po/Makevars
nagios-plugins-2.4.11/po/POTFILES.in
nagios-plugins-2.4.11/po/fr.po
nagios-plugins-2.4.11/po/de.po
nagios-plugins-2.4.11/po/fr.gmo
nagios-plugins-2.4.11/po/de.gmo
nagios-plugins-2.4.11/po/nagios-plugins.pot
nagios-plugins-2.4.11/po/stamp-po
nagios-plugins-2.4.11/po/ChangeLog
nagios-plugins-2.4.11/po/LINGUAS
nagios-plugins-2.4.11/release
```

Step 23: cd nagios-plugins-2.4.11

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

This installs the configurations for the nagios-plugins files.

```
config.status: creating test.pl
config.status: creating pkg/solaris/pkginfo
config.status: creating po/Makefile.in
config.status: creating config.h
config.status: config.h is unchanged
config.status: executing depfiles commands
config.status: executing libtool commands
config.status: executing po-directories commands
config.status: creating po/POTFILES
config.status: creating po/Makefile
[ec2-user@ip-172-31-88-33 nagios-plugins-2.4.11]$ |
```

Step 24: Next, we must compile all components of this software according to the instructions in the Makefile. To do so, use the following commands:

make

sudo make install

```
installing de.gmo as /usr/local/nagios/share/locale/de/LC_MESSAGES/nagios-plugins.mo
if test "nagios-plugins" = "gettext-tools"; then \
  /usr/bin/mkdir -p /usr/local/nagios/share/gettext/po; \
  for file in Makefile.in remove-potcdate.sin Makevars.template; do \
    /usr/bin/install -c -o nagios -g nagios -n 644 ./${file} \
      /usr/local/nagios/share/gettext/po/${file}; \
  done; \
  for file in Makevars; do \
    rm -f /usr/local/nagios/share/gettext/po/${file}; \
  done; \
else \
  : ; \
fi
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/po'
make[1]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
make[2]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
[ec2-user@ip-172-31-88-33 nagios-plugins-2.4.11]$ |
```

Step 25: sudo chkconfig --add nagios

sudo chkconfig nagios on

This registers the Nagios service with the system ensuring that it can manage the server status.

```
[ec2-user@ip-172-31-88-33 nagios-plugins-2.4.11]$ sudo chkconfig --add nagios
sudo chkconfig nagios on
error reading information on service nagios: No such file or directory
Note: Forwarding request to 'systemctl enable nagios.service'.
Created symlink /etc/systemd/system/multi-user.target.wants/nagios.service → /usr/lib/systemd/system/nagios.service.
[ec2-user@ip-172-31-88-33 nagios-plugins-2.4.11]$ |
```

Step 26: `sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg`

This command checks and verifies that the sample configuration files has no errors.

```
[ec2-user@ip-172-31-88-33 nagios-plugins-2.4.11]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
Nagios Core 4.5.5
Copyright (c) 2000-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 service escalations.
Checking for circular paths...
  Checked 1 hosts
  Checked 0 service dependencies
  Checked 0 host dependencies
  Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors: 0

Things look okay - No serious problems were detected during the pre-flight check
[ec2-user@ip-172-31-88-33 nagios-plugins-2.4.11]$ |
```

Step 27: `sudo service nagios start`

This starts the Nagios service.

```
[ec2-user@ip-172-31-88-33 nagios-plugins-2.4.11]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-88-33 nagios-plugins-2.4.11]$ |
```



Step 28: `sudo systemctl status nagios`

This checks the status of Nagios. Ensure that it is 'active(running)'.

```
[ec2-user@ip-172-31-88-33 nagios-plugins-2.4.11]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
   Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
   Active: active (running) since Sun 2024-09-29 11:25:40 UTC; 2min 3s ago
     Docs: https://www.nagios.org/documentation
  Process: 67487 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
  Process: 67488 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
 Main PID: 67489 (nagios)
    Tasks: 6 (limit: 1112)
   Memory: 6.1M
      CPU: 122ms
   CGroup: /system.slice/nagios.service
           └─67489 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             └─67490 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
               └─67491 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                 └─67492 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                   └─67493 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                     └─67494 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 29 11:25:40 ip-172-31-88-33.ec2.internal nagios[67489]: qh: core query handler registered
Sep 29 11:25:40 ip-172-31-88-33.ec2.internal nagios[67489]: qh: echo service query handler registered
Sep 29 11:25:40 ip-172-31-88-33.ec2.internal nagios[67489]: qh: help for the query handler registered
Sep 29 11:25:40 ip-172-31-88-33.ec2.internal nagios[67489]: wproc: Successfully registered manager as @wproc with query
Sep 29 11:25:40 ip-172-31-88-33.ec2.internal nagios[67489]: wproc: Registry request: name=Core Worker 67492;pid=67492
Sep 29 11:25:40 ip-172-31-88-33.ec2.internal nagios[67489]: wproc: Registry request: name=Core Worker 67493;pid=67493
Sep 29 11:25:40 ip-172-31-88-33.ec2.internal nagios[67489]: wproc: Registry request: name=Core Worker 67491;pid=67491
Sep 29 11:25:40 ip-172-31-88-33.ec2.internal nagios[67489]: wproc: Registry request: name=Core Worker 67490;pid=67490
Sep 29 11:25:41 ip-172-31-88-33.ec2.internal nagios[67489]: Successfully launched command file worker with pid 67494
Sep 29 11:27:32 ip-172-31-88-33.ec2.internal nagios[67489]: SERVICE ALERT: localhost:HTTP:WARNING:SOFT:1:HTTP WARNING:
lines 1-28/28 (END)
```

Step 29: Navigate to your EC2 instance and copy the public IPv4 address.



The screenshot shows the AWS Management Console interface. On the left, there is a navigation menu with options like 'EC2 Dashboard', 'EC2 Global View', 'Events', 'Console-to-Code', 'Instances', 'Instance Types', 'Launch Templates', 'Spot Requests', 'Savings Plans', 'Reserved Instances', 'Dedicated Hosts', 'Capacity Reservations', and 'Images'. The main content area displays the 'Instance summary' for the instance 'i-03d0c63994db6400b (nagios-host29)'. The instance is in the 'Running' state. The public IPv4 address is '5.88.19.157' and the private IPv4 address is '172.31.88.33'. The instance type is 't2.micro'. The console also shows the 'Instance state' and 'Actions' buttons.

Step 30: In the address bar, enter 'http://<publicipaddress>/nagios'.



The above page is visible.

Conclusion: In the above experiment, we learned how to install and configure Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine. We created an EC2 Linux instance with the required security rules. Then, we installed the latest versions of nagios-core and nagios-plugins and configured them to ensure that they contained no errors. Once the setup was complete, we hosted the Nagios server and accessed the Nagios dashboard by pasting the public IPv4 address of our instance in the browser.

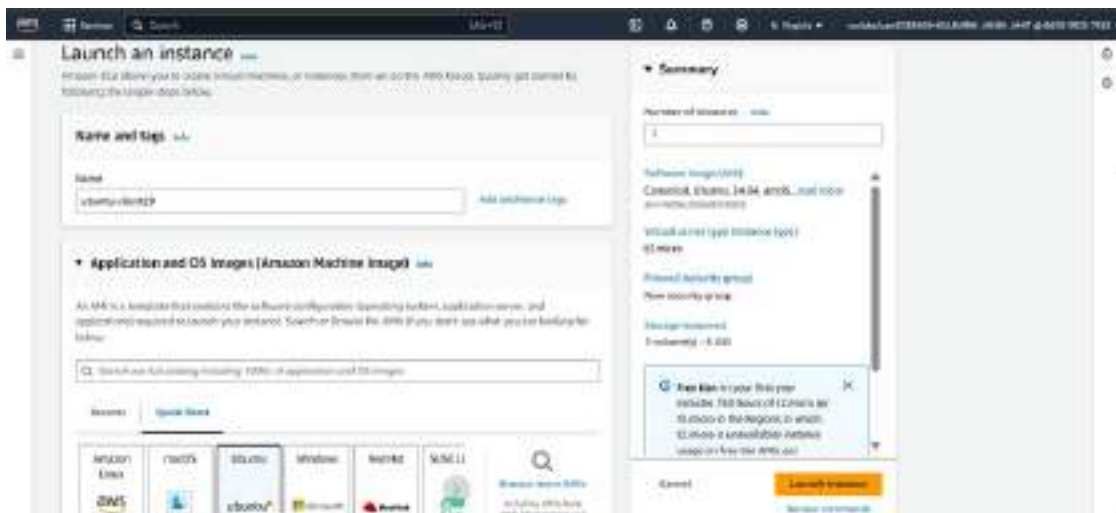
Experiment 10

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

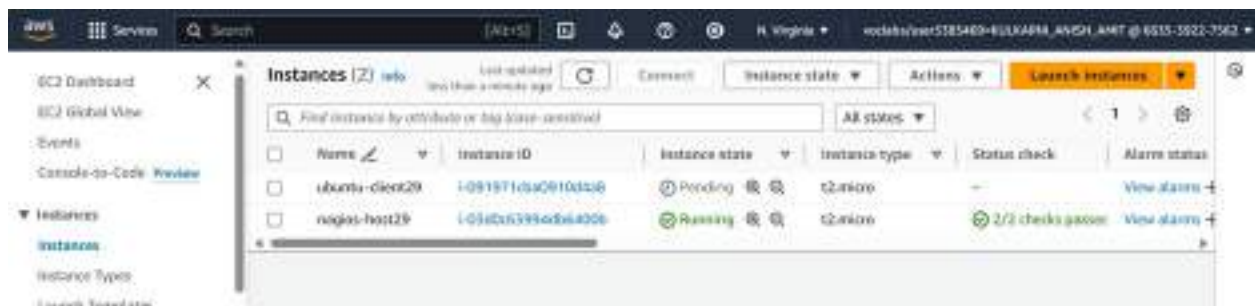
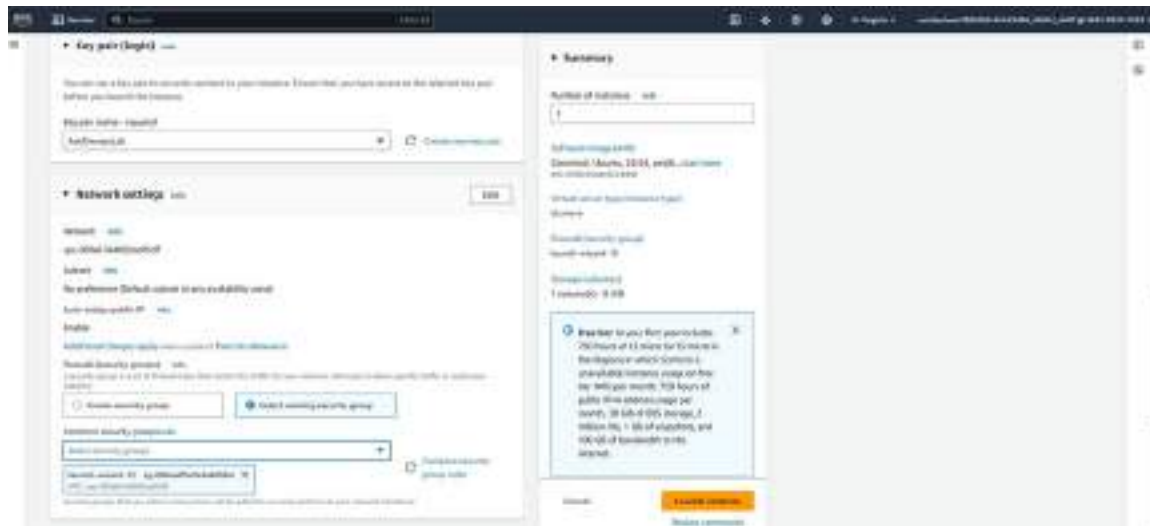
Prerequisites: An Amazon Linux instance with nagios (nagios-server) is already set up.

Steps:

Step 1: Navigate to EC2 on the AWS console using the 'Services' section and click on 'Create instance'. Give your instance a name and choose 'Ubuntu' as the instance type.



Ensure that you choose the same key pair and security group for the Ubuntu client instance as you did for the Nagios host instance. Then, click on 'Create instance'.



Your Ubuntu client instance gets created along with the Nagios host instance.

Step 2: Click on the instance ID of your nagios-server instance and click on 'Connect'. Then, click on 'SSH client' and copy the command under 'Example'. Then, open the terminal in the folder where the .pem file for your instance's key pair is located and paste the SSH command that you just copied. This connects your instance to your local terminal using SSH.

Step 3: `ps -ef | grep nagios`

Run the above command on the nagios-host instance. This verifies whether the nagios service is running or not.

```
[root@ip-172-31-88-33 ec2-user]# ps -ef | grep nagios
nagios 67489 1 0 11:25 ? 00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios 67498 67489 0 11:25 ? 00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios
nagios 67491 67489 0 11:25 ? 00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios
nagios 67492 67489 0 11:25 ? 00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios
nagios 67493 67489 0 11:25 ? 00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios
nagios 67494 67489 0 11:25 ? 00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
root 69887 68853 0 11:51 pts/1 00:00:00 grep --color=auto nagios
```

Step 4: sudo su

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

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

This makes you the root user and creates two folders with the above paths.

```
[root@ip-172-31-88-33 ec2-user]# sudo su
mkdir /usr/local/nagios/etc/objects/monitorhosts
mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-88-33 ec2-user]#
```

Step 5: We need to create a config file in this folder. So, copy the contents of the existing localhost config to the new file 'linuxserver.cfg'.

cp /usr/local/nagios/etc/objects/localhost.cfg

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

```
[root@ip-172-31-88-33 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
```

Step 6: We need to make some changes in this config file. Open it using nano editor:-

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

1. Change hostname and alias from 'hostname' to 'linuxserver'.
2. Change address to the public ip address of the ubuntu-client instance.

```
#####
4
5 HOST DEFINITION
6
#####

# Define a host for the local machine

define host {

    use                linux-server          ; Name of host template to use
                                           ; This host definition will inherit all variables that are defined
                                           ; in (or inherited by) the linux-server host template definition.

    host_name          linuxserver
    alias              linuxserver
    address            52.91.181.68
}

#####
7
8 #####

```

Change hostgroup_name to 'linux-servers1'.

```
define hostgroup {

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

```

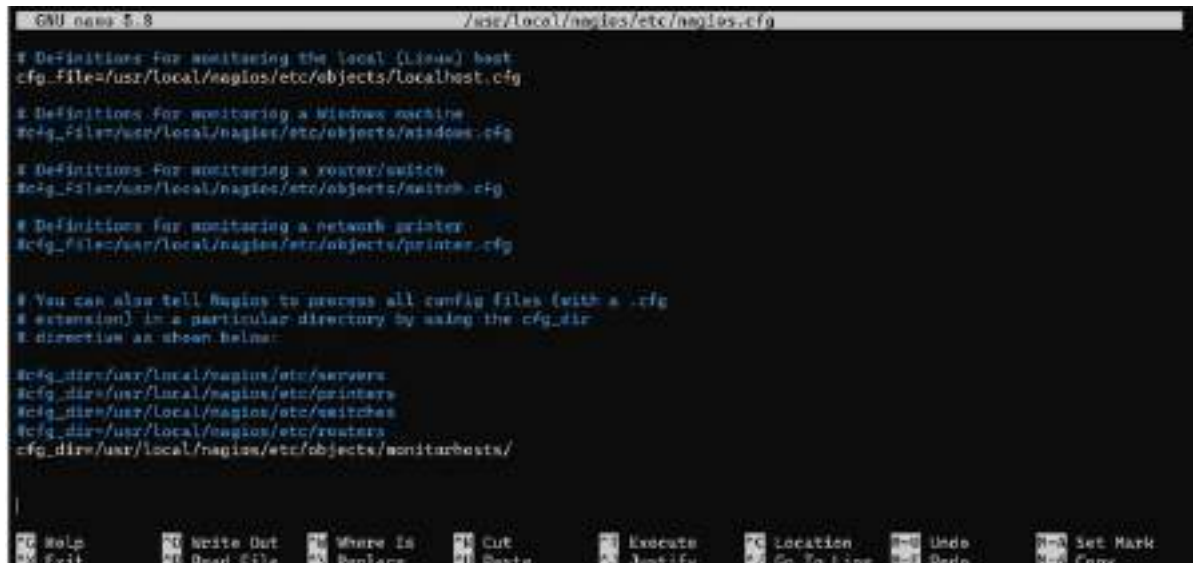
Change all the subsequent occurrences of hostname in the file from 'localhost' to linuxserver'.

Step 7: Open the Nagios config file using the following command:

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

Then, add the following line to the config file:

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



```
GNU nano 2.8 /usr/local/nagios/etc/nagios.cfg

# Definitions for monitoring the local (Linux) host
cfg_file=/usr/local/nagios/etc/objects/localhost.cfg

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

# Definitions for monitoring a router/switch
cfg_file=/usr/local/nagios/etc/objects/switch.cfg

# Definitions for monitoring a network printer
cfg_file=/usr/local/nagios/etc/objects/printer.cfg

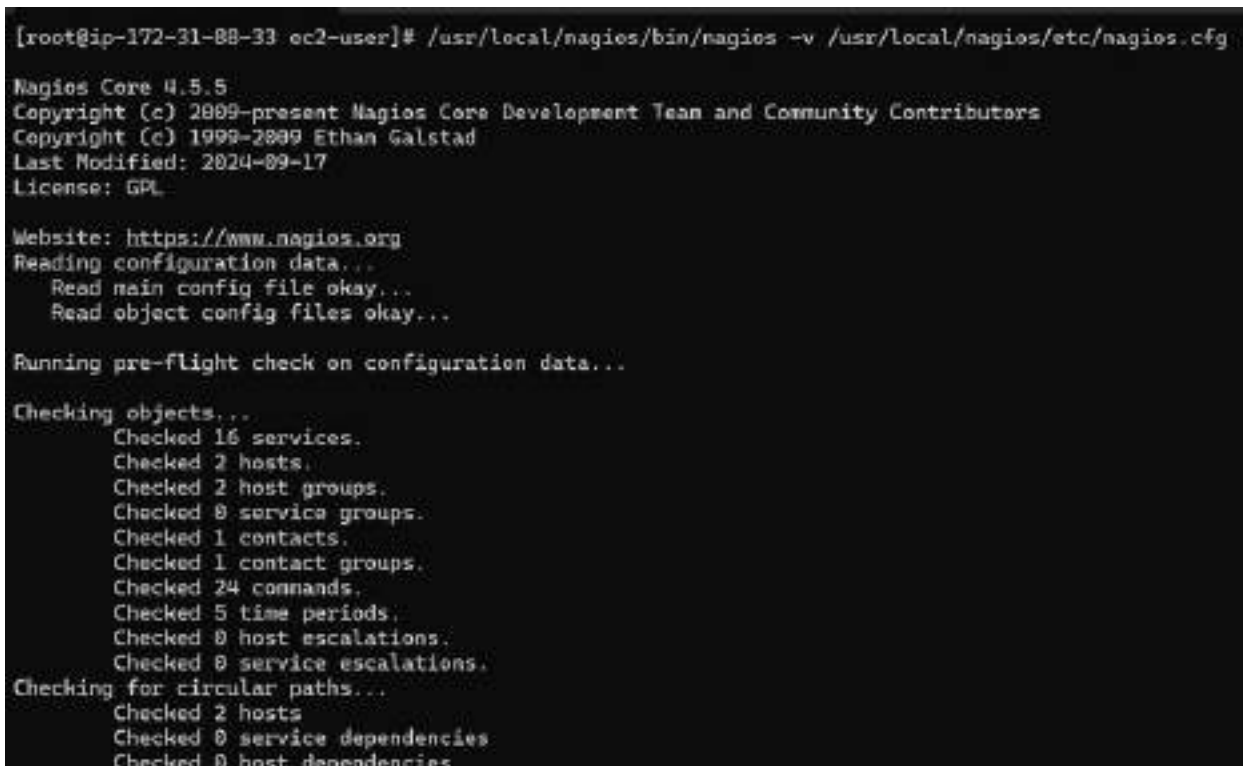
# You can also tell Nagios to process all config files (with a .cfg
# extension) in a particular directory by using the cfg_dir
# directive as shown below:

cfg_dir=/usr/local/nagios/etc/servers
cfg_dir=/usr/local/nagios/etc/printers
cfg_dir=/usr/local/nagios/etc/switches
cfg_dir=/usr/local/nagios/etc/routers
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/

^M Help      ^O Write Out ^W Where Is  ^C Cut       ^X Execute  ^G Location ^U Undo     ^M Set Mark
^M Exit      ^R Read File ^E Replace   ^V Paste    ^_ Justify   ^_ Go To Line ^_ Undo    ^_ Copy
```

Step 8: Now we verify the configuration files and check that they contain no errors using the following command:

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



```
[root@ip-172-31-88-33 ec2-user]# /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

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

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

Running pre-flight check on configuration data...

Checking objects...
  Checked 16 services.
  Checked 2 hosts.
  Checked 2 host groups.
  Checked 0 service groups.
  Checked 1 contacts.
  Checked 1 contact groups.
  Checked 24 commands.
  Checked 5 time periods.
  Checked 0 host escalations.
  Checked 0 service escalations.
Checking for circular paths...
  Checked 2 hosts
  Checked 0 service dependencies
  Checked 0 host dependencies
```

```

    Checked 0 host dependencies
    Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors: 0

Things look okay - No serious problems were detected during the pre-flight check
[root@ip-172-31-88-33 ec2-user]#

```

Step 9: Once the files are verified and it is confirmed that there are no errors, we must restart the server.

service nagios restart

```

[root@ip-172-31-88-33 ec2-user]# service nagios restart
Redirecting to /bin/systemctl restart nagios.service

```

Step 10: systemctl status nagios

Using the above command, we check the status of the nagios server and ensure that it is active (running).

```

[root@ip-172-31-88-33 ec2-user]# systemctl status nagios
● nagios.service - Nagios Core 4.3.5
   Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
   Active: active (running) since Sun 2024-09-29 12:11:40 UTC; 1min 12s ago
     Docs: https://www.nagios.org/documentation
   Process: 70244 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0)
   Process: 70245 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 70246 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 4.0M
        CPU: 38ms
   CGroup: /system.slice/nagios.service
           └─70246 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             70247 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             70248 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             70249 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             70250 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             70251 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: qh: core query handler registered
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: qh: echo service query handler registered
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: qh: help for the query handler registered
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: wproc: Successfully registered manager as wproc with query
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: wproc: Registry request: name=Core Worker 70250;pid=70250
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: wproc: Registry request: name=Core Worker 70249;pid=70249
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: wproc: Registry request: name=Core Worker 70248;pid=70248
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: wproc: Registry request: name=Core Worker 70247;pid=70247
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: Successfully launched command file worker with pid 70251

```


Step 11: Connect your ubuntu-client instance to your local terminal using SSH in the same way as you connected the nagios-host instance to your local terminal using SSH (follow Step 2)

```
P5 C:\Users\anish\Downloads> ssh -i "AdvDevopsLab.pem" ubuntu@ec2-52-91-101-68.compute-1.amazonaws.com
The authenticity of host 'ec2-52-91-101-68.compute-1.amazonaws.com (52.91.101.68)' can't be established.
ED25519 key fingerprint is SHA256:Z6cgJrMFcPl55xJ9EzJHr831t1bYaGlx6Mtu/PKusPw.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-52-91-101-68.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

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

System information as of Sun Sep 29 12:15:18 UTC 2024

System load:  0.0               Processes:            105
Usage of /:   22.7% of 6.71GB   Users logged in:     0
Memory usage: 19%              IPv4 address for enx0: 172.31.94.199
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

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

Step 12: On your ubuntu-client instance, run the following commands:-

```
sudo apt update -y
```

```
sudo apt install gcc -y
```

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

The above commands check for any new updates and then install gcc, Nagios NRPE server and Nagios plugins.

```
ubuntu@ip-172-31-94-199:~$ sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [388 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [110 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [535 kB]
Get:15 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [82.9 kB]
```

```

Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...

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

No containers need to be restarted.

User sessions running outdated binaries:
ubuntu @ session #4: sshd[1021,1132]
ubuntu @ user manager service: systemd[1027]

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-94-199:~$ |

```

Step 13: Run the following command:

```
sudo nano /etc/nagios/nrpe.cfg
```

The above command opens the NRPE config file. Here, we need to add the public IP address of our host nagios-host instance to the NRPE configuration file.

Under `allowed_hosts`, add the nagios-host public IPv4 address.

```

#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
nrpe_group=nagios

# ALLOWED HOST ADDRESSES
# This is an optional comma-delimited list of IP address or hostnames
# that are allowed to talk to the NRPE daemon. Network addresses with a bit mask
# (i.e. 192.168.1.0/24) are also supported. Hostname wildcards are not currently
# supported.
#
# Note: The daemon only does rudimentary checking of the client's IP
# address. I would highly recommend adding entries in your /etc/hosts.allow
# file to allow only the specified host to connect to the port
# you are running this daemon on.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
allowed_hosts=127.0.0.1,3.84.19.157

```

Step 14: Navigate to the Nagios dashboard. Click on 'hosts'. We see that linuxserver has been added as a host.

Nagios - 194.19.157.nagios

Current Network Status
Last Updated: Sun Sep 28 12:21:35 UTC 2024
Updated every 30 seconds
Nagios Core 4.5.5 - www.nagios.org
Logged in as nagios@nagios

Host Status Totals
Up: 1, Down: 0, Unreachable: 0, Pending: 0
All Problems: 0, All Types: 0

Service Status Totals
OK: 12, Warning: 1, Unknown: 0, Critical: 3, Pending: 0
All Problems: 4, All Types: 10

Host Status Details For All Host Groups

Host	Status	Last Check	Next Check	Status Information
linuxserver	UP	09-28-2024 12:21:35	09-28-2024 12:21:35	PING OK - Packet loss = 0%, RTT = 1.22 ms
linuxserver2	UP	09-28-2024 12:21:35	09-28-2024 12:21:35	PING OK - Packet loss = 0%, RTT = 0.24 ms

Click on 'linuxserver'. Here, we can access all information about the 'linuxserver' host.

Nagios - 194.19.157.nagios

Host Information
Last Updated: Sun Sep 29 10:23:19 UTC 2024
Updated every 30 seconds
Nagios Core 4.5.5 - www.nagios.org
Logged in as nagios@nagios

Host State Information

Host Status: UP (for 0h 2m 11m 41s)
Status Information: PING OK - Packet loss = 0%, RTT = 1.22 ms
Performance Data: rx=1.222900,tx=0.00000,drop=0.00000,packets=0.00000,packets/s=0.00000
Current Attempt: 1/10 (HARD reset)
Last Check Time: 09-29-2024 10:21:02
Check Type: ACTIVE
Check Latency / Duration: 0.805 / 4.122 seconds
Next Scheduled Active Check: 09-29-2024 10:28:00
Last State Change: 09-29-2024 10:11:40
Last Notification: N/A (notification off)
Is This Host Flapping? NO (0.00% state changes)
Is Scheduled Downtime? NO
Last Update: 09-29-2024 10:23:19 (1h 2m 11m 41s ago)

Host Commands

- Locate host on map
- Disable active checks of this host
- Re-check the host state of this host
- Schedule passive check of this host
- Stop accepting passive checks for this host
- Stop monitoring over this host
- Disable notifications for this host
- Send custom fatal notification
- Schedule downtime for this host
- Schedule downtime for all services on this host
- Disable notifications for all services on this host
- Create notifications for all services on this host
- Schedule a check of all services on this host
- Disable checks of all services on this host
- Create checks of all services on this host
- Disable event handler for this host
- Create log-rotation for this host
- Create flapping state for this host

Click on 'Services'. Here, we can see all the services that are being monitored by 'linuxserver'.



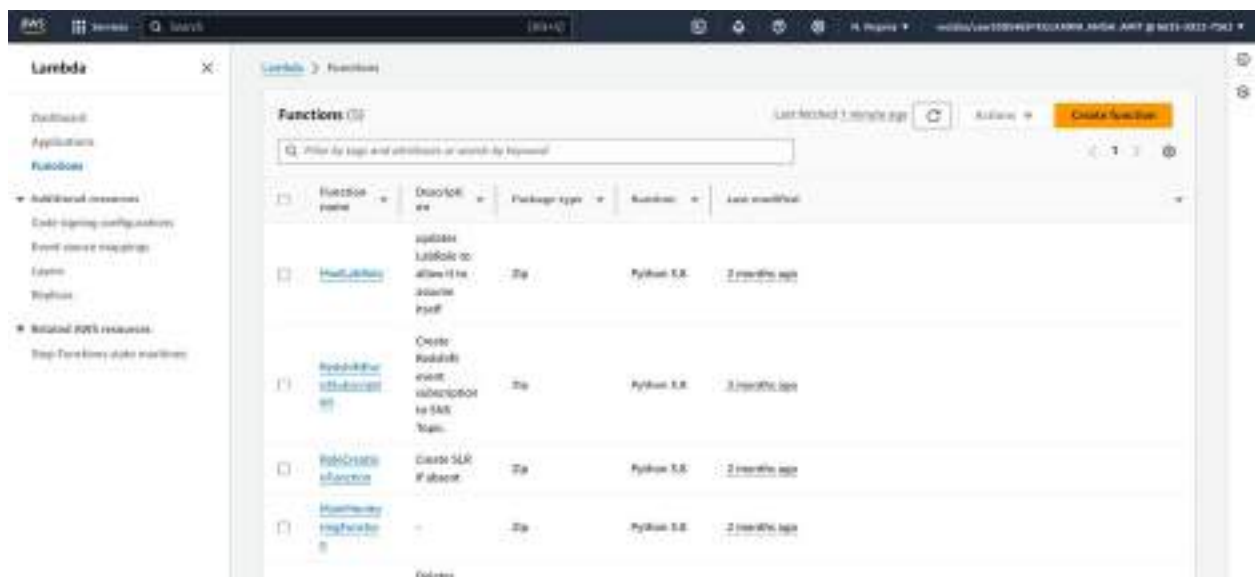
Conclusion: In this experiment, we learned how to perform port, service monitoring, Windows/Linux server monitoring using Nagios. To do so, we needed a nagios-host EC2 Linux instance which was used to host the Nagios server and dashboard. We created an Ubuntu client instance to connect to the host. We set up some configurations on the Linux instance and added the public IP address of the Ubuntu instance in it. We also set up some configurations on the Ubuntu client instance and added the IP address of the Linux server instance in it. Then, we made sure to add the Linux server instance as a 'allowed host' for the Ubuntu client instance. After restarting the NRPE server, we can see the 'linuxserver' host added.

Experiment 11

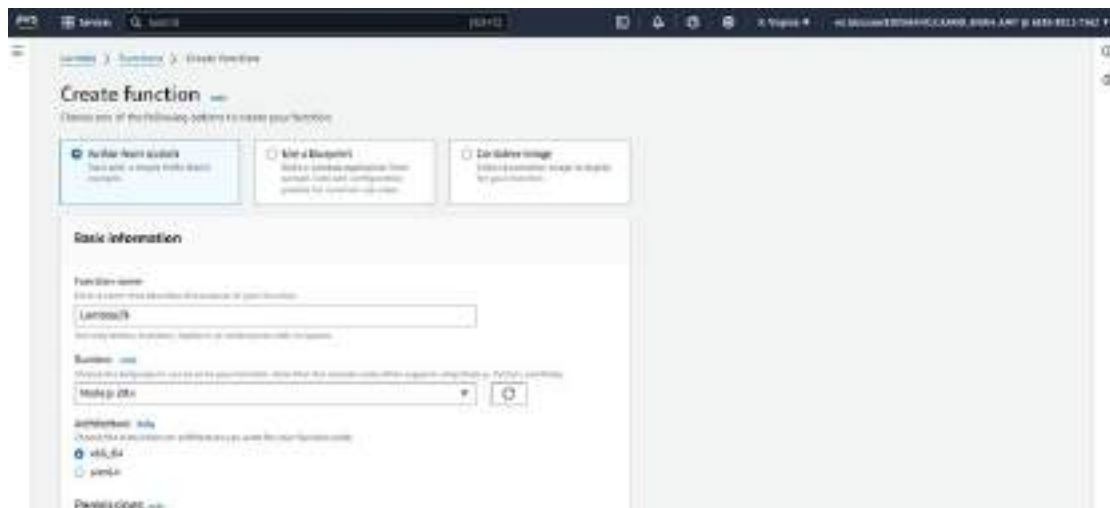
Aim: To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

Steps:

Step 1: On your AWS console, click on 'Lambda' in the services section and click on 'Create function'.



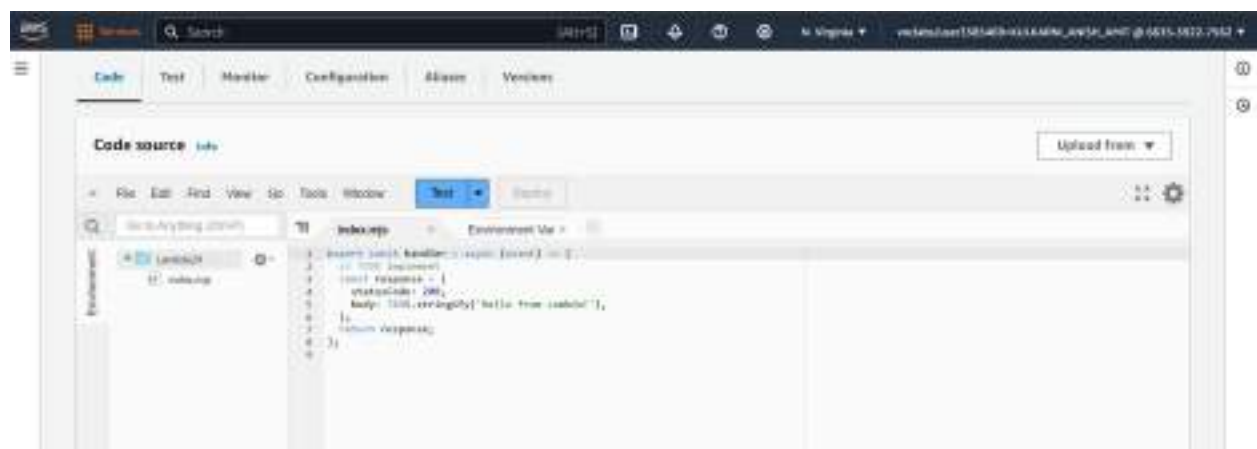
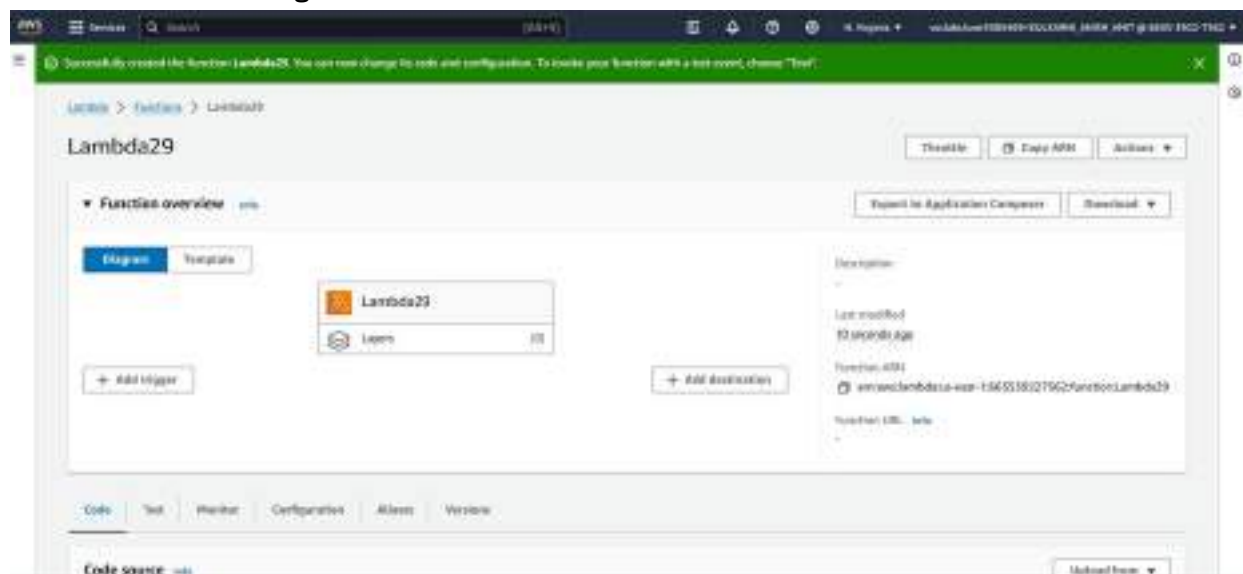
Step 2: Give your Lambda function a name. Select the language to use to write your function (Node.js is the default and what we will use in this experiment). Keep other options as default.



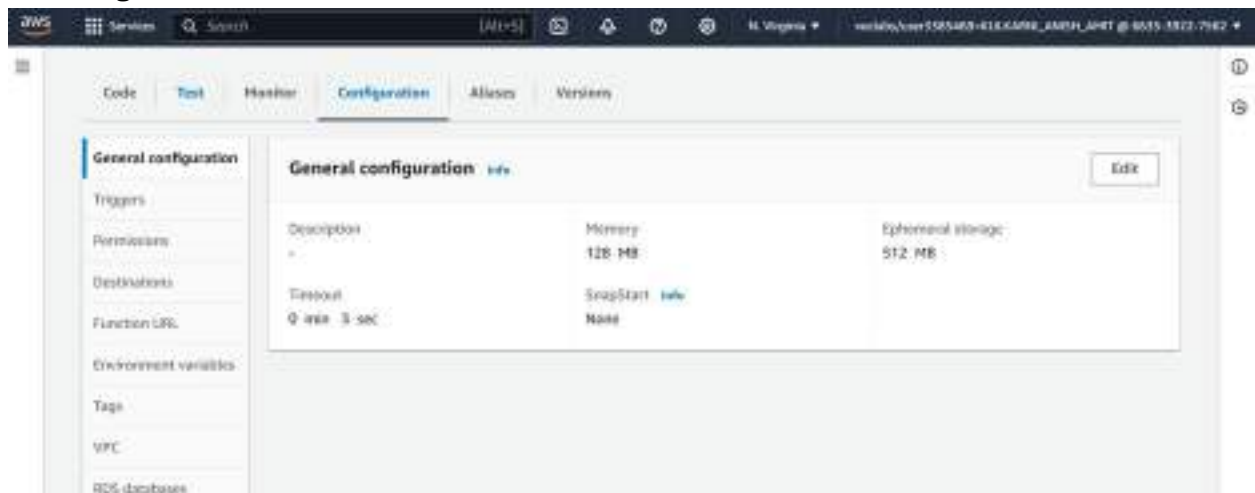


Under 'Execution role', choose 'Use an existing role' and then choose LabRole. Then, click on 'Create function'.

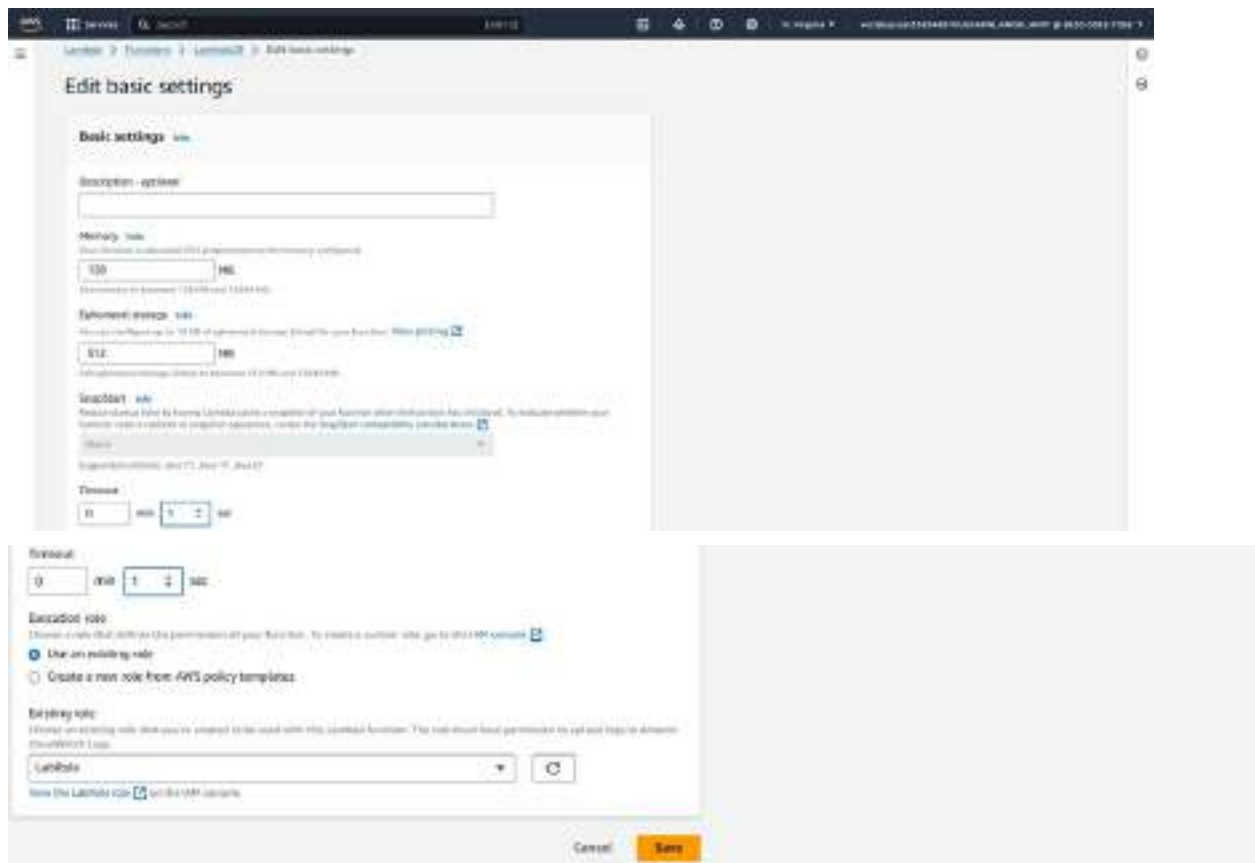
Your Lambda function gets created.



Step 3: The general configuration of the function is visible in the 'Configuration' tab. To change the configuration, click on 'Edit'.

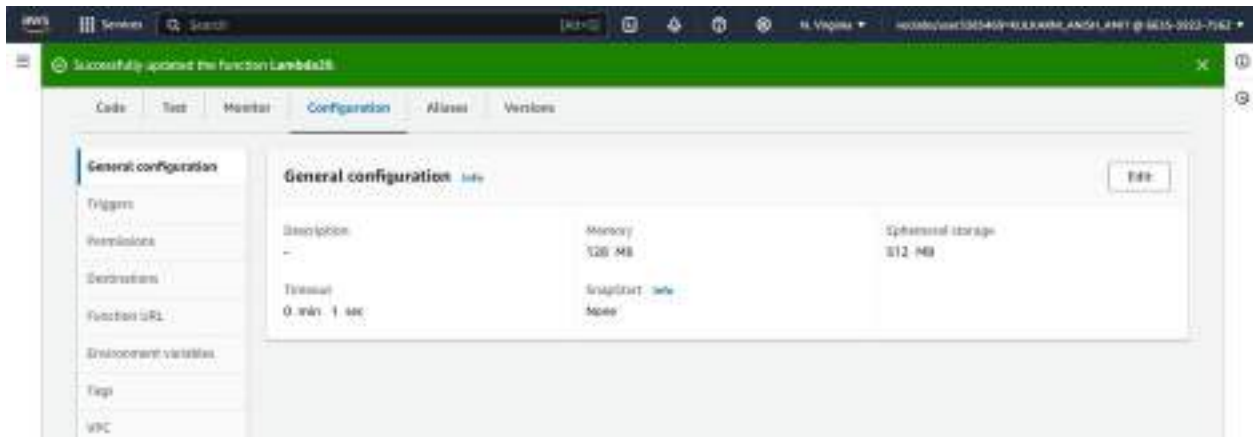


You can change the various parameters of the configuration as per your needs. Here, we can change the 'Timeout' period to 1 second as it's sufficient for our function for now. 'Timeout' is the time for which a function can be running before it gets forcibly terminated.



After making the required changes, click on 'Save'.

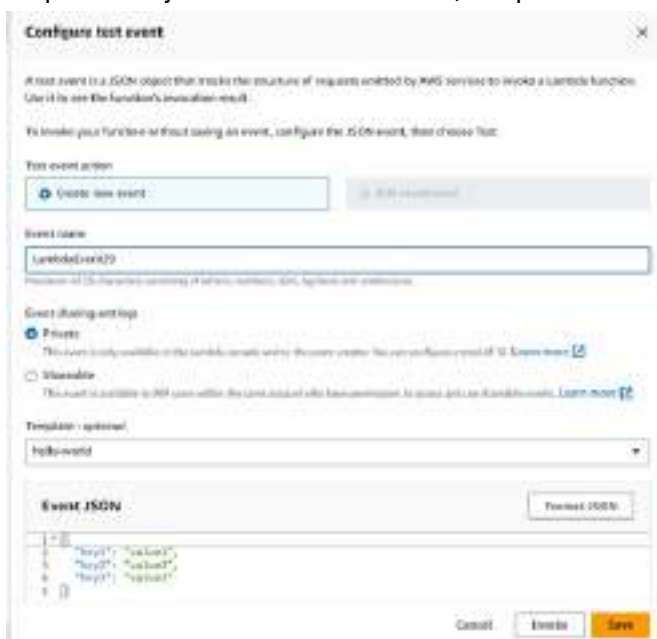
The changes in the general configuration are visible in the function.



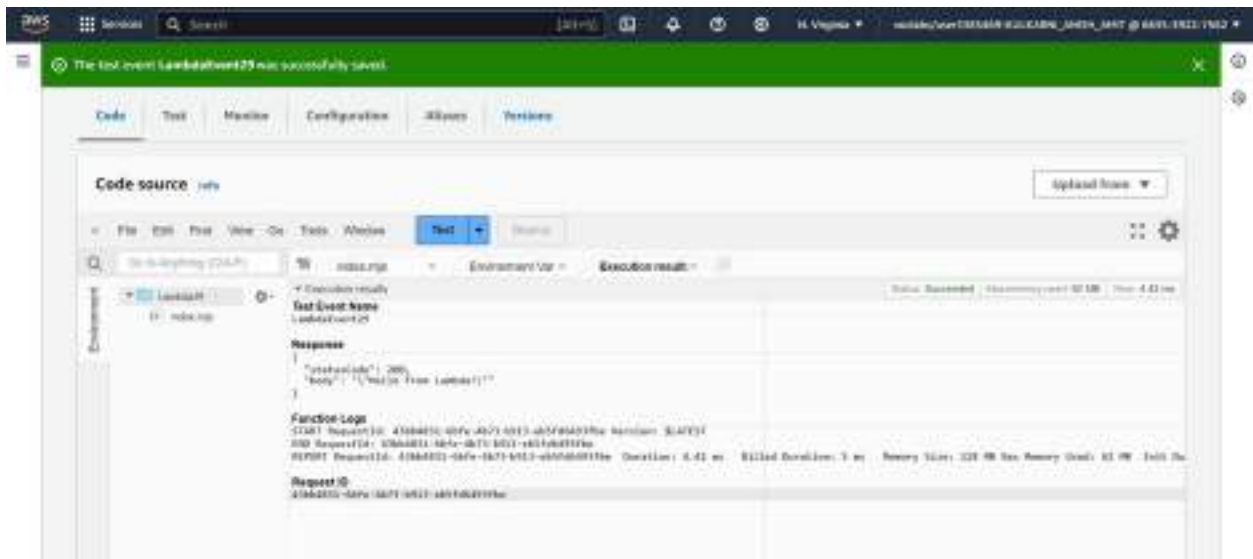
Step 4: In the 'Code source' section, click on the arrow next to the 'Test' button and click on 'Configure test event'.



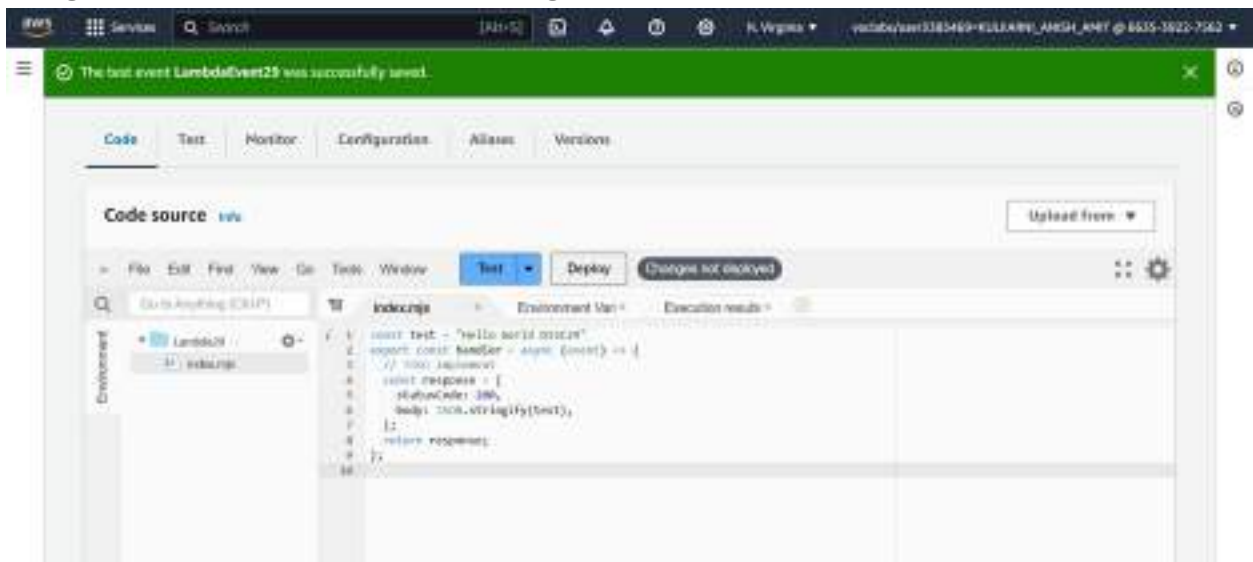
Step 5: Give your test event a name, keep all other options as default and click on 'Save'.



Step 6: Click on the 'Test' button. The following output appears.

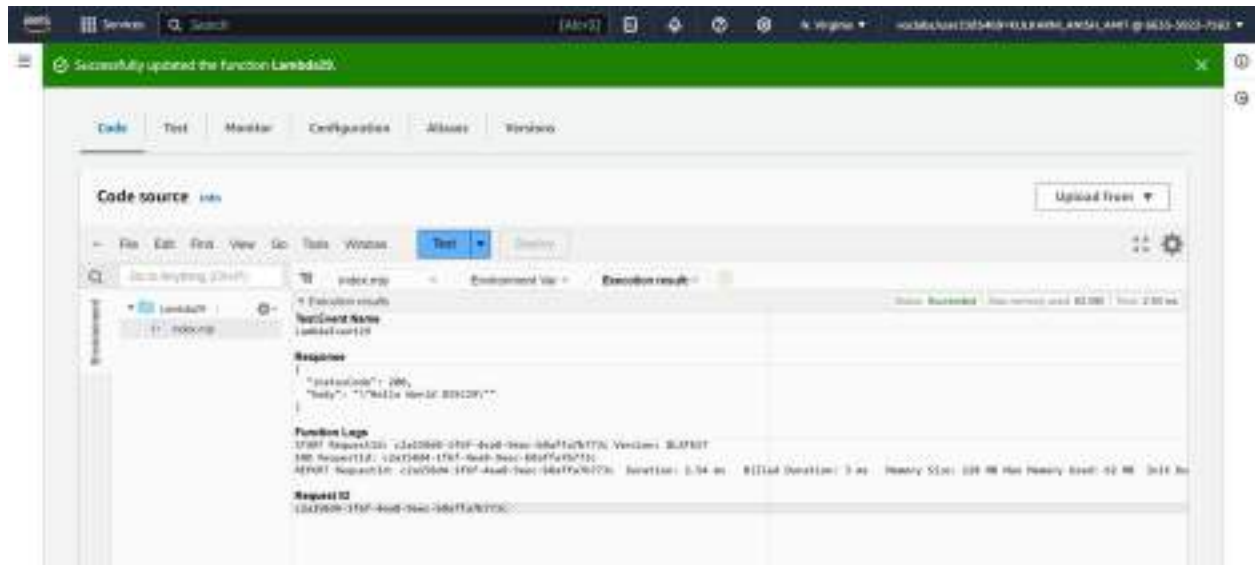


Step 7: You can make changes in the code to observe the difference in the output. Here, we change the code to display a different string as such:-



Once the changes are made, click on 'Deploy'.

Step 8: Click on 'Test' and observe how the output after the changes differs from the output before the changes.



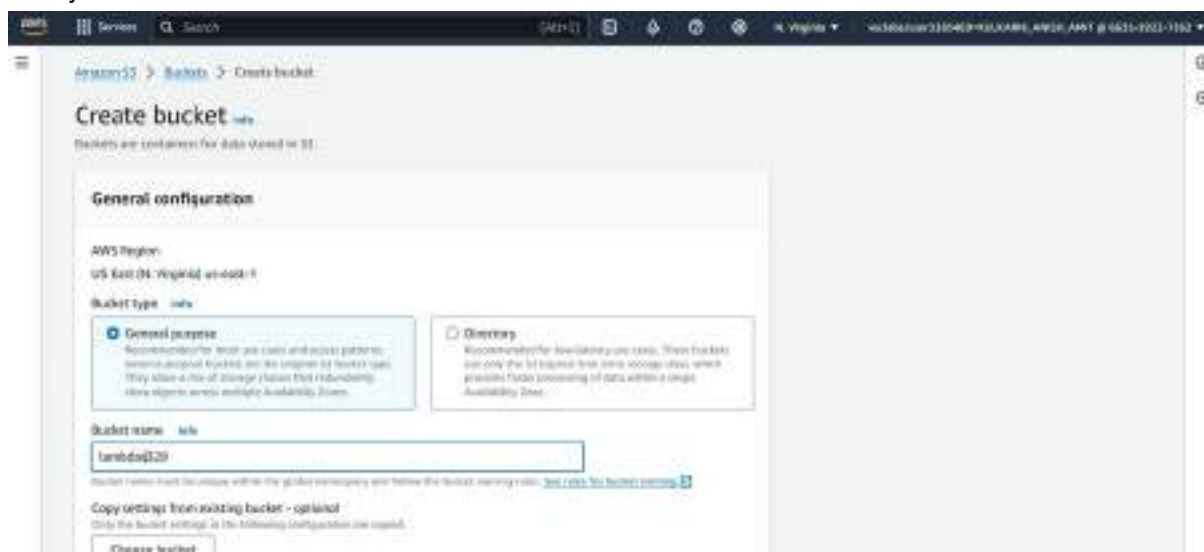
Conclusion: In this experiment, we understood the working of AWS Lambda service by creating and configuring a Lambda function using Node.js. We learned how to set up a Lambda function, change its configurations and test its functioning by creating test events for the function. From the output of the tests, we learn about the working of the Lambda function and how changes in its configuration affects its functionality and outputs.

Experiment 12

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

Steps:

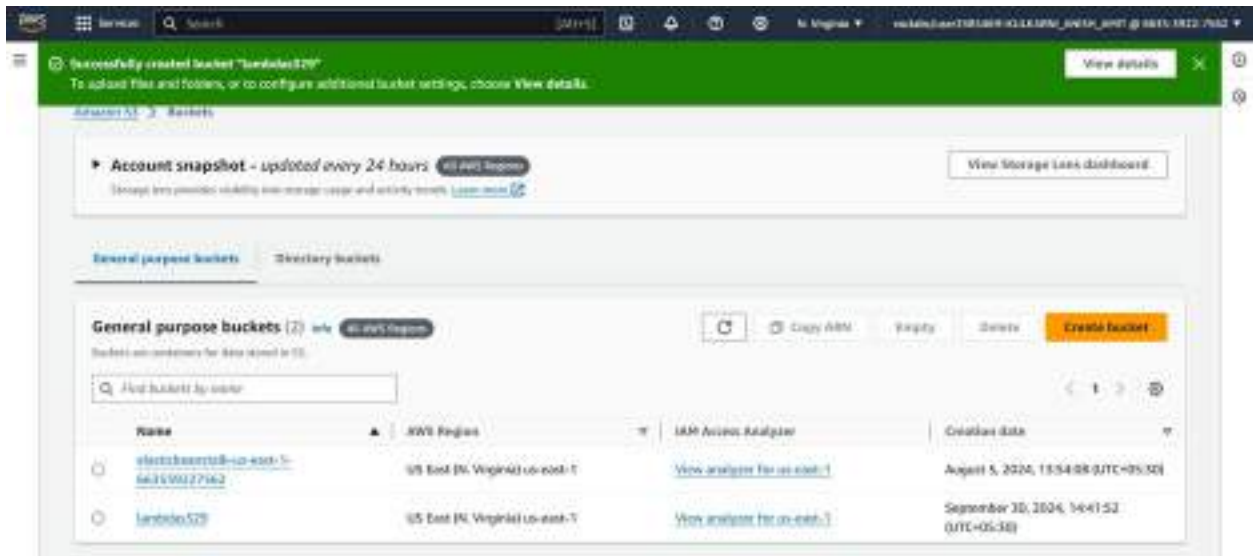
Step 1: On your AWS console, click on ‘S3’ in the services section and click on ‘Create bucket’. Give your bucket a name.



Uncheck the ‘Block all public access’ box.



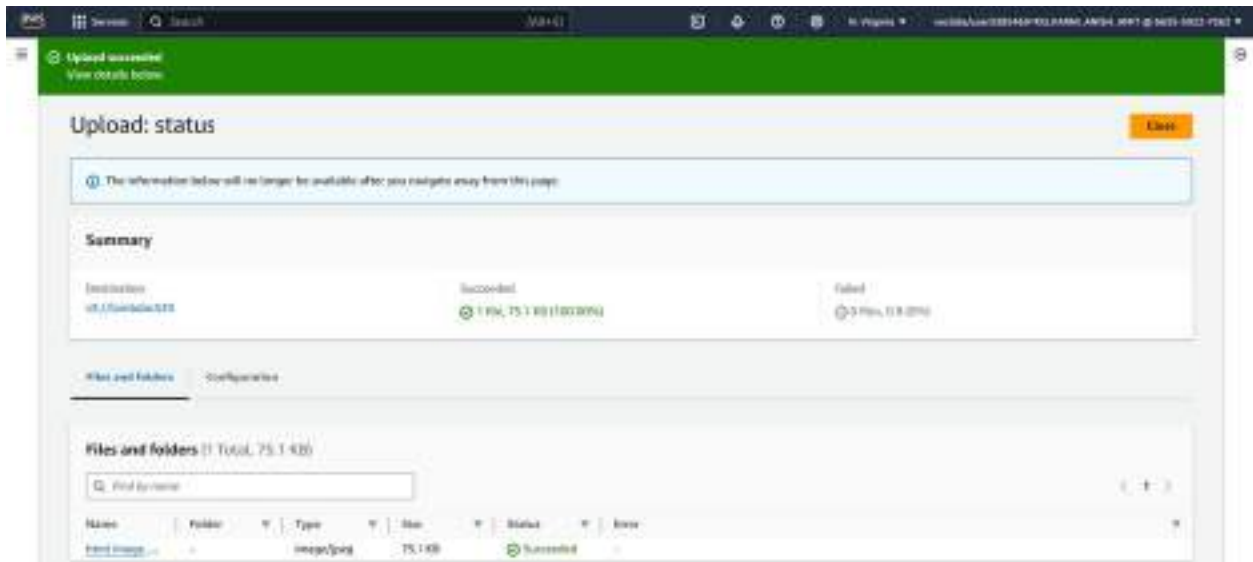
Keep all other options as default and click on ‘Create bucket’.



Your bucket is created.

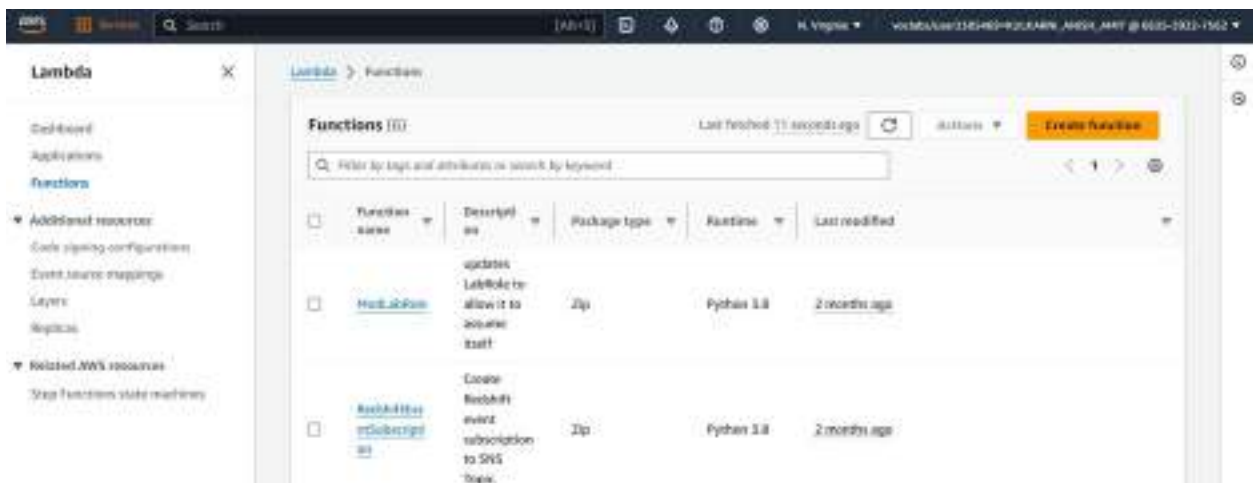
Step 2: Upload an image onto your S3 bucket by clicking on your S3 bucket, clicking on 'Upload', clicking on 'Add files', navigating to your image and selecting it.



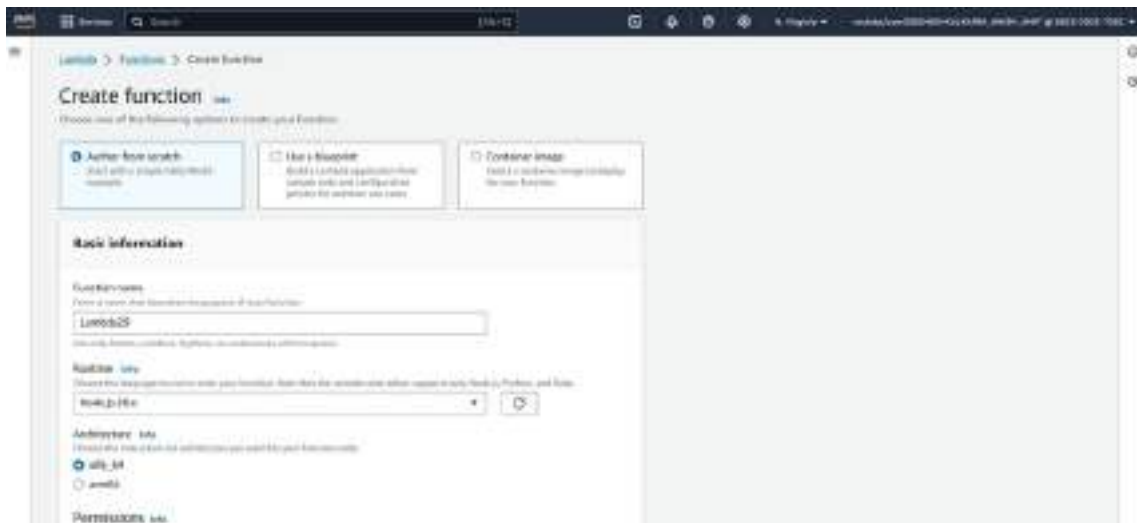


Your image gets uploaded onto the S3 bucket.

Step 3: Navigate to the AWS Lambda console using the 'Services' section. Click on 'Create function'.



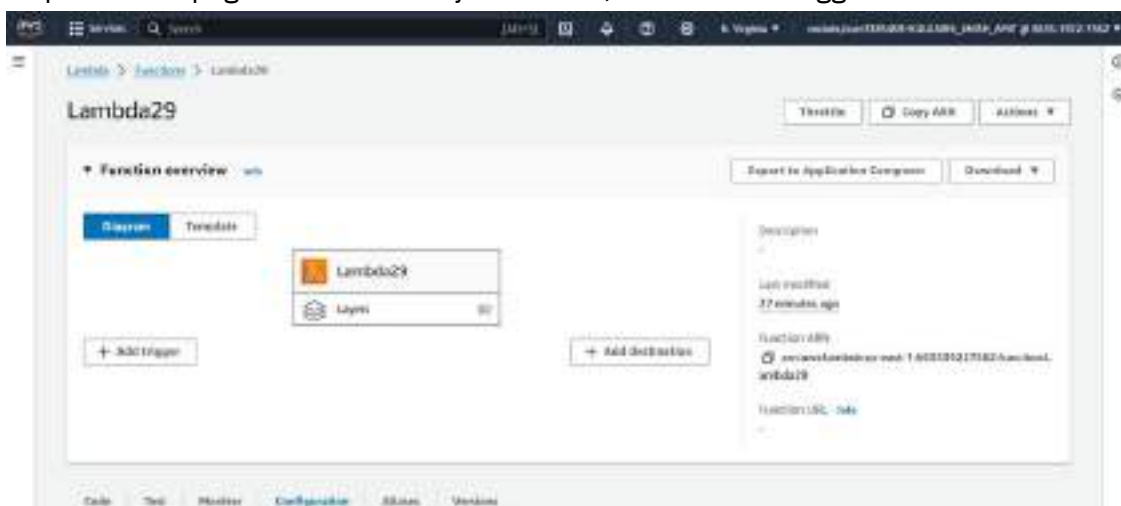
Step 4: Give your function a name and keep other settings as default.



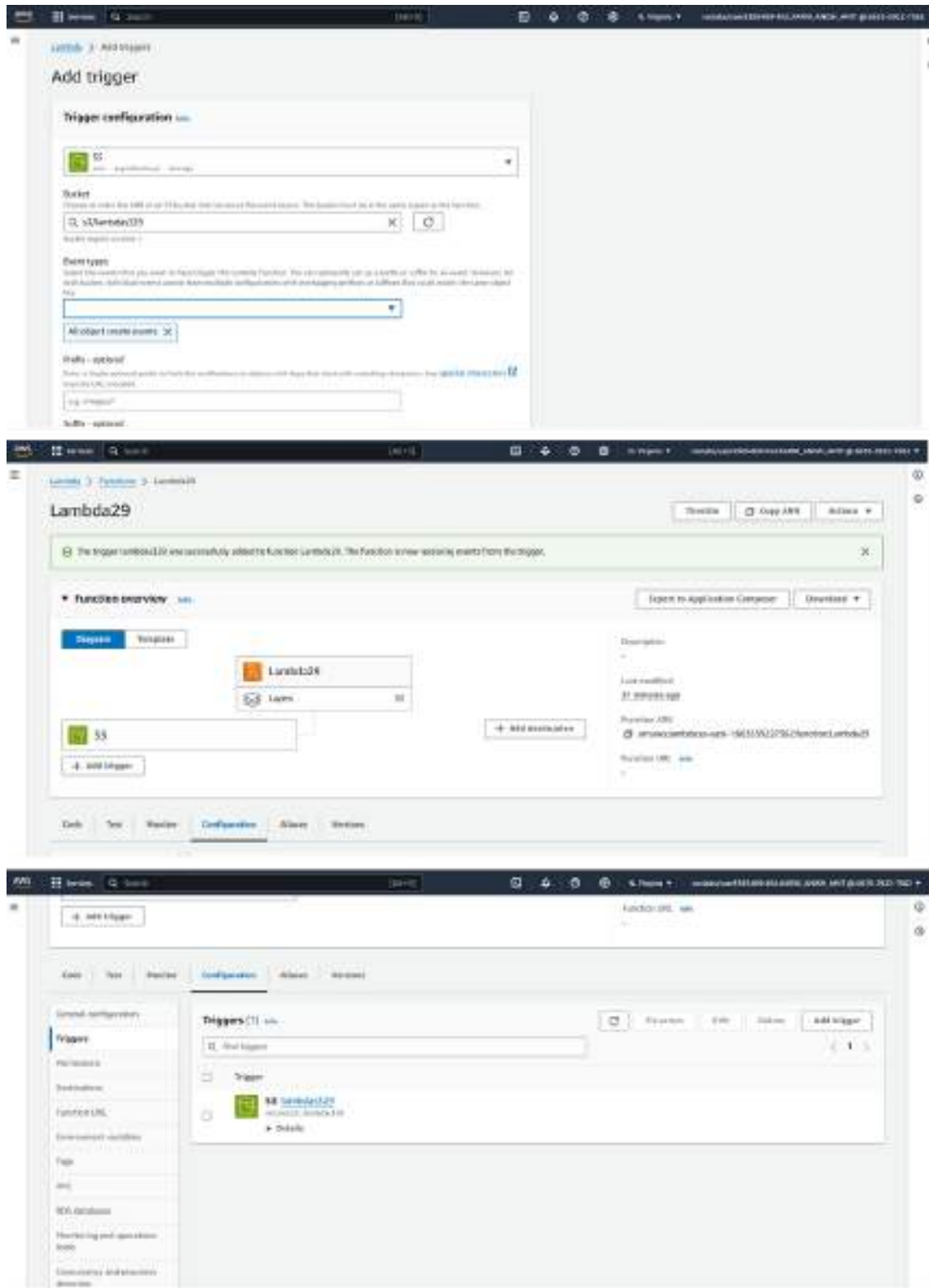
Under 'Execution role', choose 'Use an existing role' and in the dropdown box below, choose 'LabRole'. Then, click on 'Create function'. Your function gets created.



Step 5: On the page of the function you created, click on 'Add trigger'.



Step 6: Choose 'Trigger configuration' as S3 and select the name of your bucket in the dropdown box below it. Keep other options as default and click on 'Add'.



The trigger gets successfully added to your function.

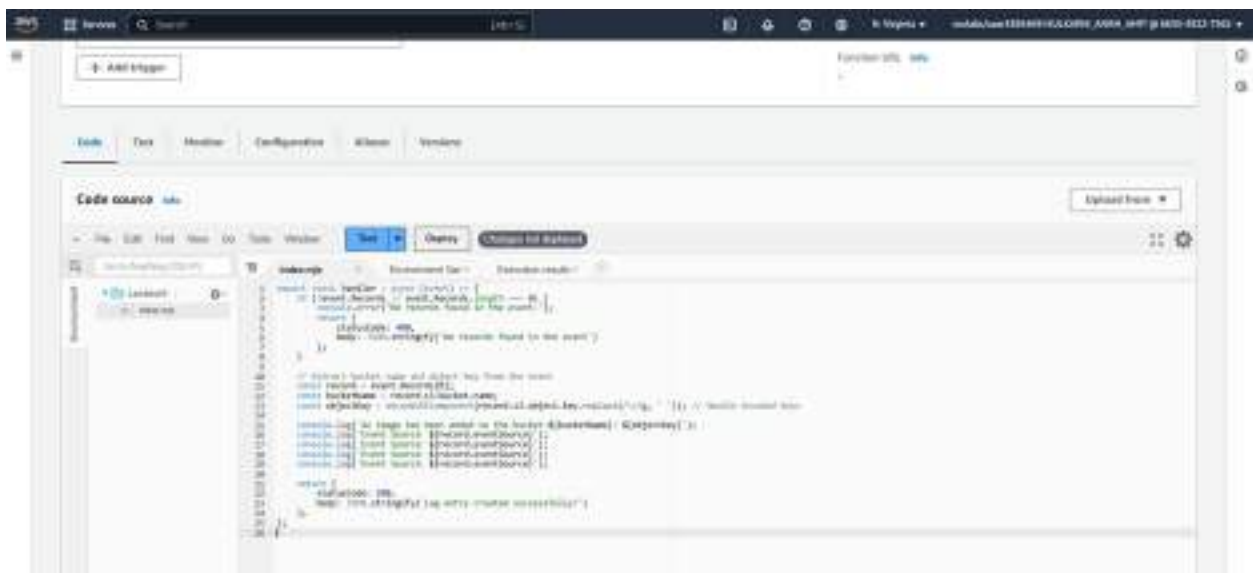
Step 7: In the 'Code source' section of your function, paste the following javascript code instead of the existing code:-

```
export const handler = async (event) => {  
  if (!event.Records || event.Records.length === 0) {  
    console.error("No records found in the event.");  
    return {  
      statusCode: 400,  
      body: JSON.stringify('No records found in the event')  
    };  
  }  
}
```

```
// Extract bucket name and object key from the event  
const record = event.Records[0];  
const bucketName = record.s3.bucket.name;  
const objectKey = decodeURIComponent(record.s3.object.key.replace(/\+/g, ' ')); // Handle  
encoded keys
```

```
console.log(`An image has been added to the bucket ${bucketName}: ${objectKey}`);  
console.log(`Event Source: ${record.eventSource}`); console.log(`Event Source:  
${record.eventSource}`); console.log(`Event Source: ${record.eventSource}`);  
console.log(`Event Source: ${record.eventSource}`);
```

```
return {  
  statusCode: 200,  
  body: JSON.stringify('Log entry created successfully!')  
};  
};
```



Step 8: Click on the arrow next to the 'Test' button and click on 'Configure test event'. In the popup box that appears, if you have an existing event, enter the name of your event or create a new event and in the 'Event JSON' section, paste the following code:-

```
{
  "Records": [
    {
      "eventVersion": "2.0",
      "eventSource": "aws:s3",
      "awsRegion": "us-east-1",
      "eventTime": "1970-01-01T00:00:00.000Z",
      "eventName": "ObjectCreated:Put",
      "userIdentity": {
        "principalId": "EXAMPLE"
      },
      "requestParameters": {
        "sourceIPAddress": "127.0.0.1"
      },
      "responseElements": {
        "x-amz-request-id": "EXAMPLE123456789",
        "x-amz-id-2":
"EXAMPLE123/5678abcdefghijklmbdaisawesome/mnopqrstuvwxyzABCDEFGH"
      },
      "s3": {
        "s3SchemaVersion": "1.0",
        "configurationId": "testConfigRule",
        "bucket": {
          "name": "example-bucket",
          "ownerIdentity": {
            "principalId": "EXAMPLE"
          },
          "arn": "arn:aws:s3:::example-bucket"
        },
        "object": {
          "key": "test%2Fkey",
          "size": 1024,
          "eTag": "0123456789abcdef0123456789abcdef",
          "sequencer": "0A1B2C3D4E5F678901"
        }
      }
    }
  ]
}
```


[illegible]

Conclusion: In this experiment, we learned how to create a Lambda function which logs “An image has been added” once we add an image to our specific S3 bucket. We first created an S3 bucket and uploaded an image to it. We then created a Lambda function and added an S3 trigger to it and selected the S3 bucket we created. Then, we configured the ‘Code section’ of our Lambda function and a test event for our Lambda function. On running the test event, we observed that it logged important information about the event such as the bucket name and object key and also verified that an image had been added to the S3 bucket. Also, a log of our Lambda function was also created which confirmed that the image had been successfully added to the bucket.