

Experiment No: 1 (A)

AIM: To develop a website and host it on local machine or virtual machine and Amazon S3 Bucket

STATIC HOSTING :

1) On local server (XAMPP)

Step 1: Install XAMPP from <https://www.apachefriends.org/>

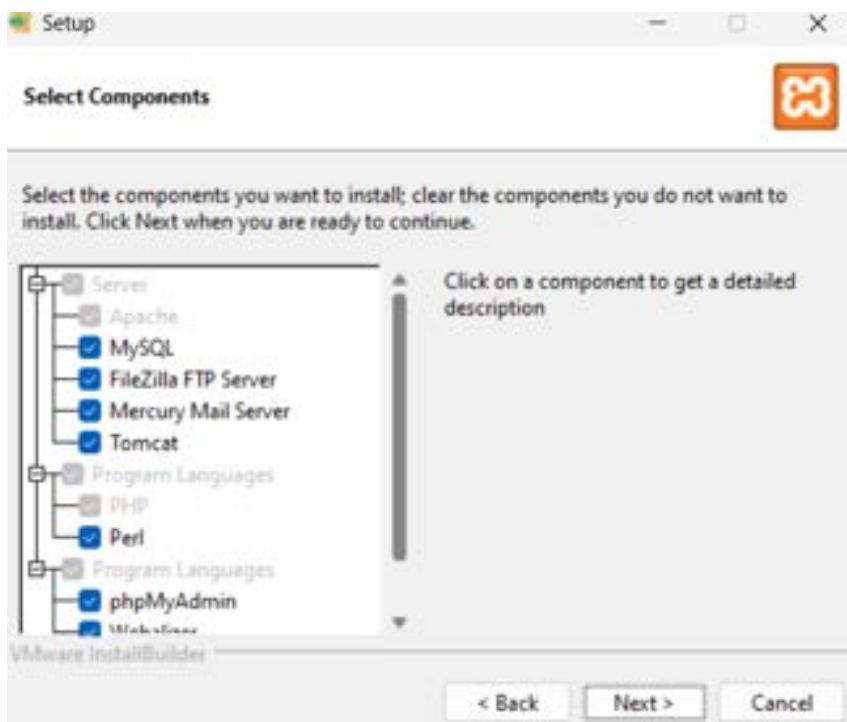
a) Select your OS. It will automatically start downloading



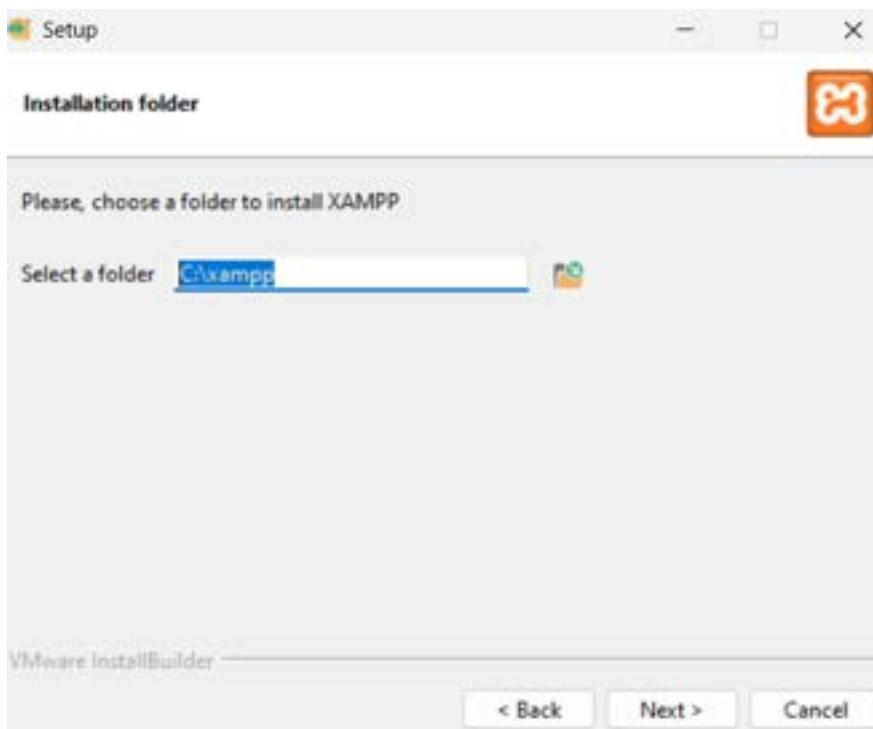
b) Open the setup file. Click on Next



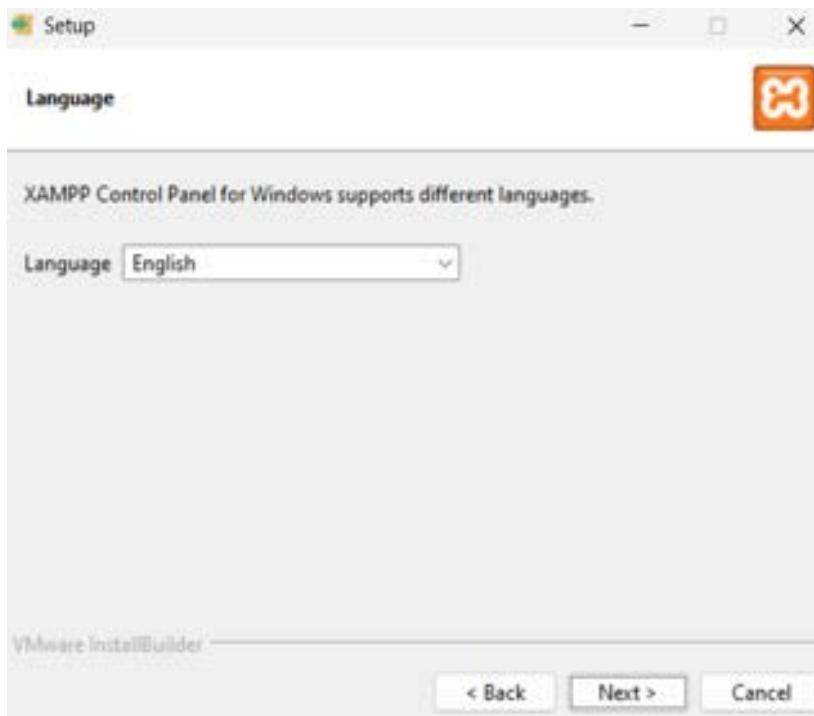
c) Select all the required components and then click on next



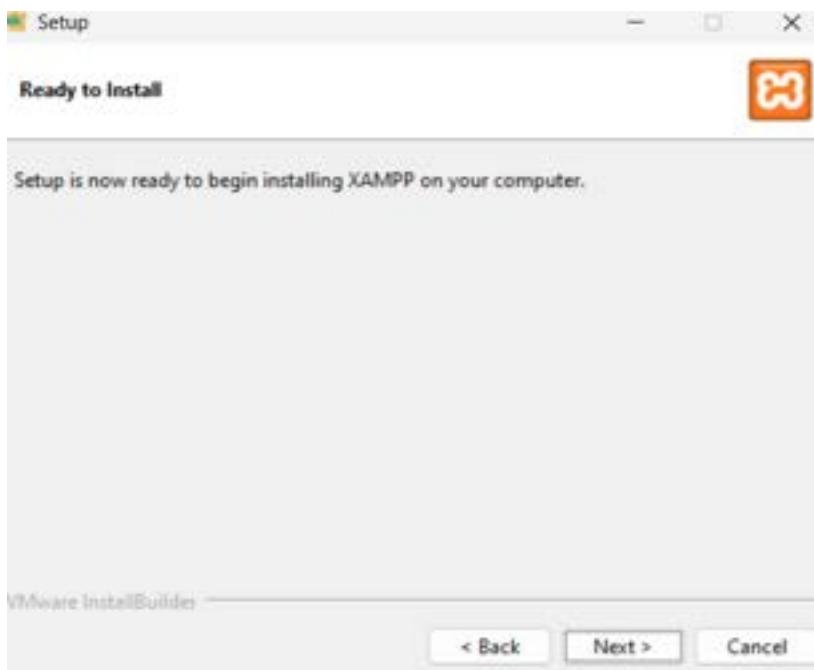
d) Choose the folder to install XAMPP in. Make sure the folder is empty. Click on next



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



f) Now it will ask user for final confirmation for installing thus Click on Next



g) Wait until unpacking of all files is done



h) The installation is complete. Click on Finish



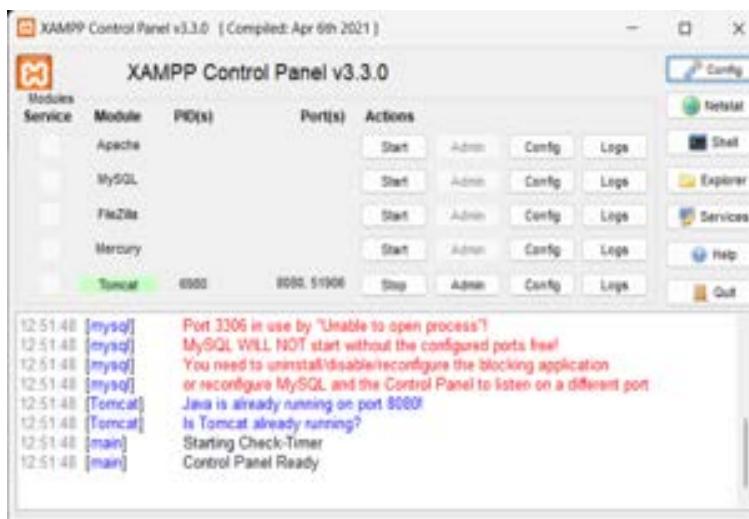
Step 2: Setup a file that is to be hosted on the server. Make sure the file has extension .php

Name	Date modified	Type	Size
index	04-08-2024 18:02	HTML File	3 KB
index2	11-08-2024 12:57	PHP Source File	3 KB

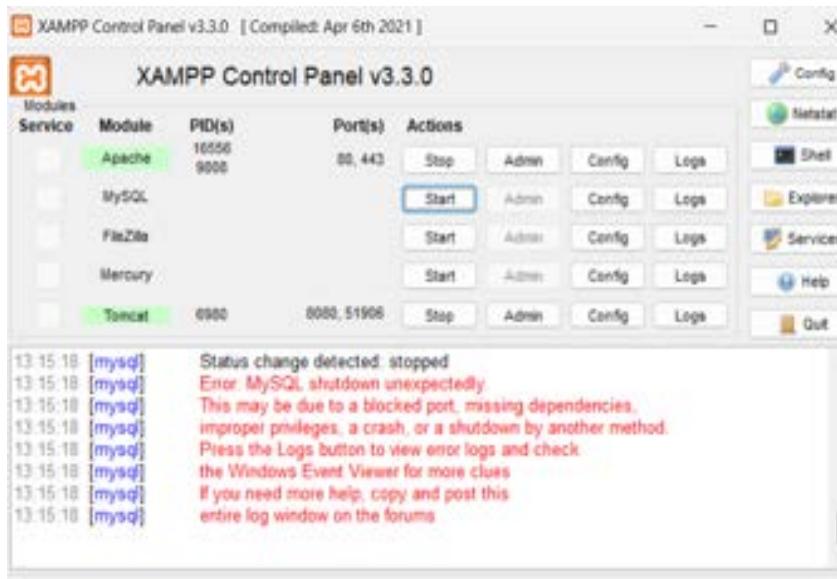
Step 3: Go to the directory where XAMPP was installed. Go to htdocs folder. Place your folder in this directory i.e Paste the index2.php here

Name	Date modified	Type	Size
dashboard	11-08-2024 12:45	File folder	
img	11-08-2024 12:45	File folder	
webalizer	11-08-2024 12:44	File folder	
xampp	11-08-2024 12:45	File folder	
applications	15-06-2022 21:37	HTML File	4 KB
bitnami	15-06-2022 21:37	CSS Source File	1 KB
favicon	16-07-2015 21:02	ICO File	31 KB
index	16-07-2015 21:02	PHP Source File	1 KB
index2	11-08-2024 12:57	PHP Source File	3 KB

Step 4: Open XAMPP Control Panel, start the Apache service (Required) and mySQL service (if needed)



Click on Start button corresponding to Apache



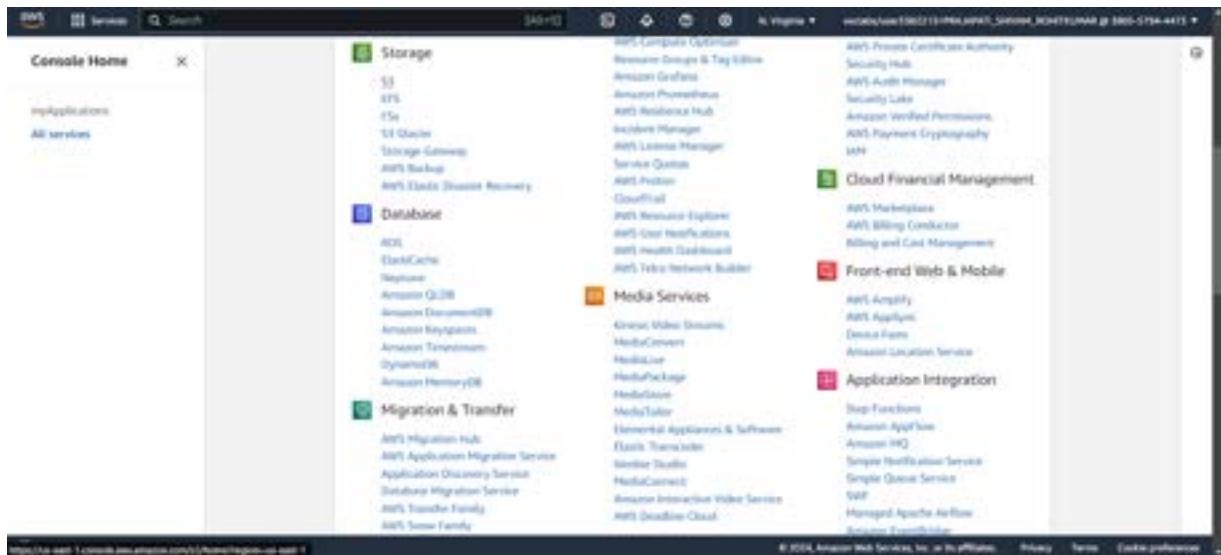
Step 5: Open your web browser. Type localhost/index2.php in the browser . This will open your website on your browser.

The screenshot shows a web browser window with the URL "localhost/index2.php". The page content includes:

- A large heading "Advanced Devops" with a sub-section "Advanced Devops" below it.
- A "Shopping list" section with the following items:
 - A. milk
 - B. eggs
 - C. oil
 - D. flour
 - 5kg flour
 - 1kg
 - 2kg
 - 1kg
 - E. oil
- A sidebar with a list of names:
 - Akash Pragji
 - Gaurav Joshi
 - Sandesh Vaidya
 - Akash Vaidya
 - Aditya Dholay
- Links for "Gaurav" and "Notes".

2) On AWS S3

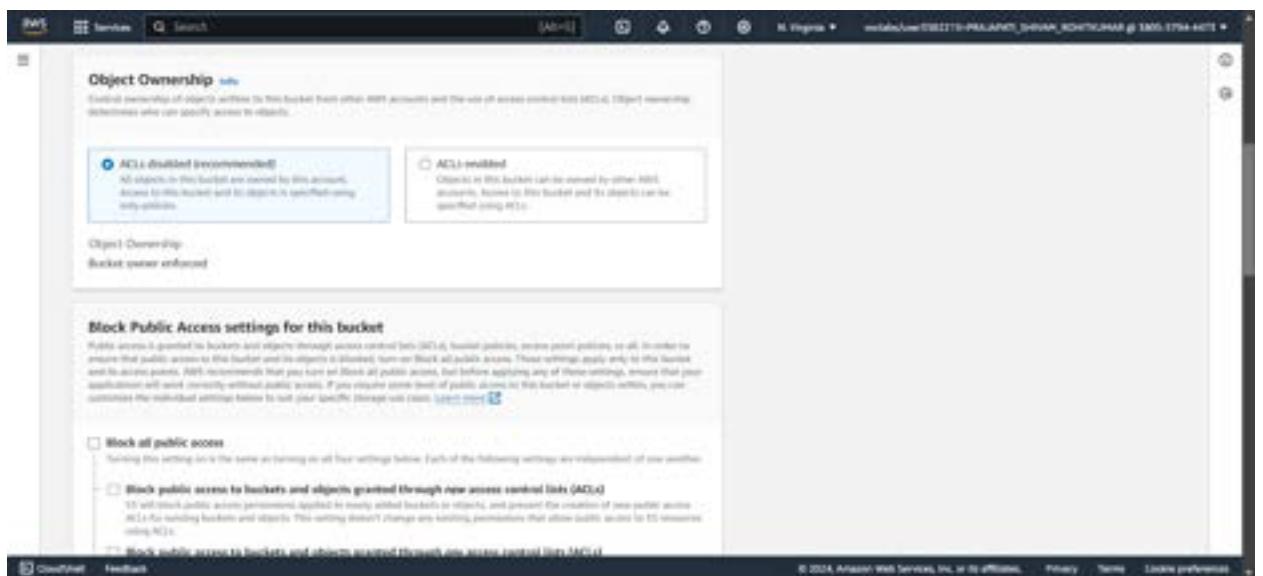
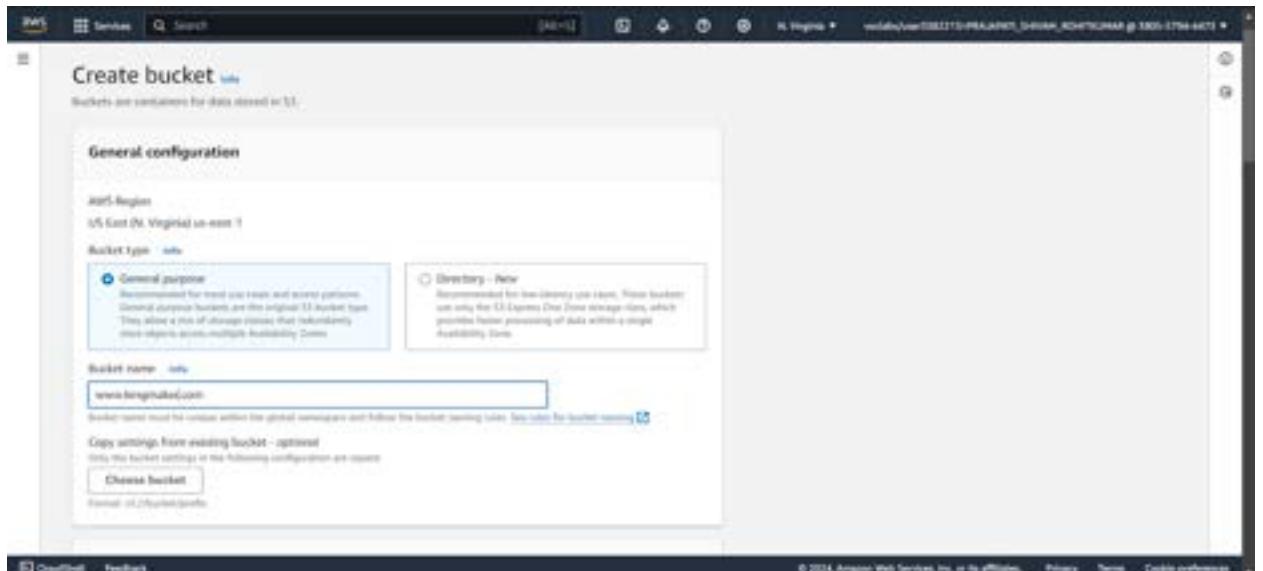
Step 1: Login to your AWS account. Go to services and open S3.



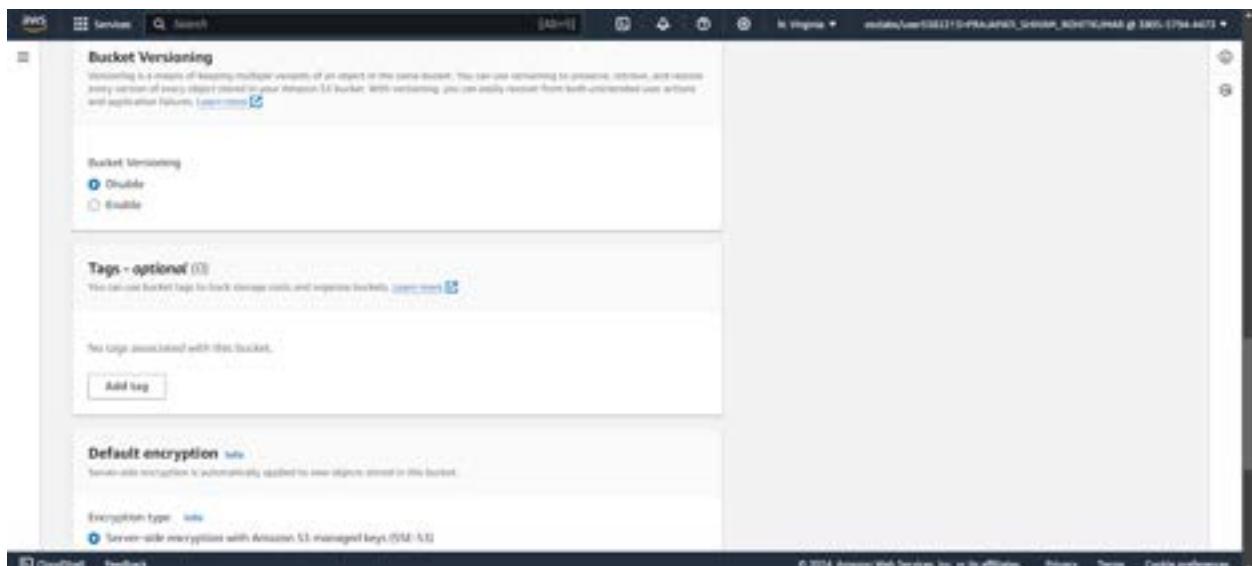
Step 2: Click on Create Bucket

The screenshot shows the Amazon S3 service page. At the top, there's a prominent 'Create a bucket' button. To the left, there's a section titled 'How it works' featuring a video thumbnail about 'Introduction to Amazon S3'. To the right, there's a 'Pricing' section stating that there are no minimum fees and providing a link to estimate monthly costs. Below that is a 'Resources' section with links to the 'User guide' and 'Sample Monthly Estimate'.

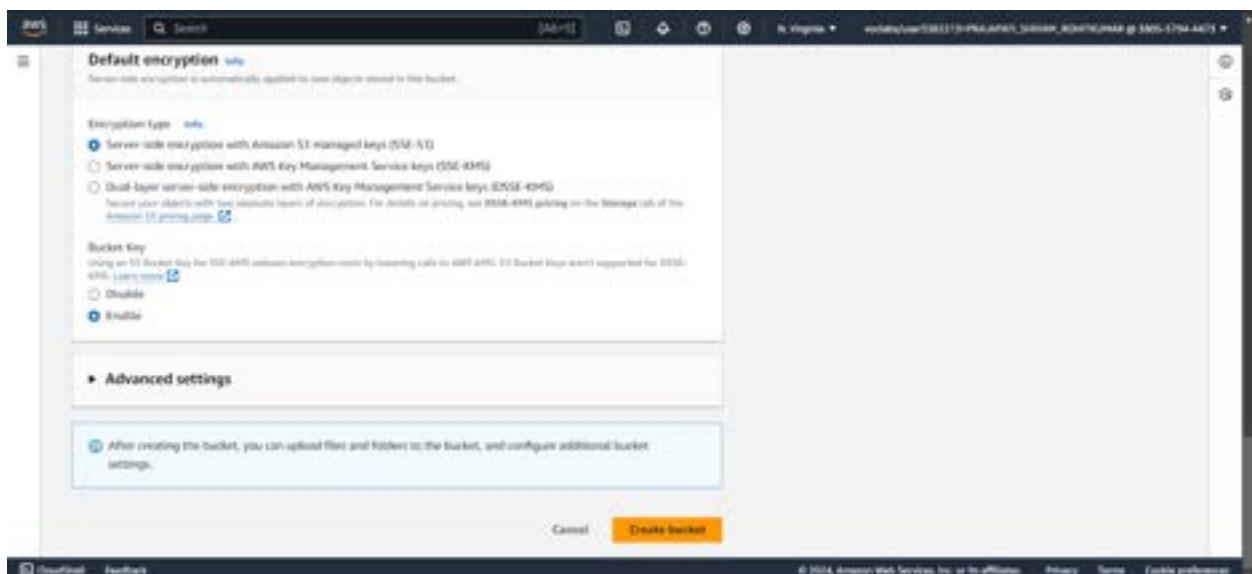
Step 3: Give a name to your bucket, keeping other options default like bucket type as general purpose , scroll down and click on Create Bucket



Uncheck the box under Block Public Access settings if it is checked because we want bucket to be publicly accessed



Keep the Bucket Versioning disabled as we dont want variants of an object to the same bucket.



Click on Create Bucket

Step 4: Click on the name of your bucket and goto Properties.

Successfully created bucket "www.kingmaker.com". To upload files and folders, or to configure additional bucket settings, choose View details.

Amazon S3 > Buckets

Account snapshot - updated every 24 hours [View details](#)

General purpose buckets [View Storage Lens dashboard](#)

General purpose buckets [View All Buckets](#)

Buckets are containers for data stored in S3.

Find buckets by name:

Name	AWS Region	S3 API Access Analyzer	Creation date
www.kingmaker.com	US East (N. Virginia) us-east-1	View analyzer for us-east-1	August 4, 2024, 17:55:21 UTC+05:30
www.mywebsite.com	US East (N. Virginia) us-east-1	View analyzer for us-east-1	July 28, 2024, 10:09:18 UTC+05:30

Copy S3A | Empty | Delete | Create bucket

« 1 »

Amazon S3 > Buckets > www.kingmaker.com

www.kingmaker.com [Edit](#)

Objects Properties Permissions Metrics Management Access Points

Bucket overview

ARN: arn:aws:s3:::www.kingmaker.com

Resource Name (ARN): [arn:aws:s3:::www.kingmaker.com](#)

Creation date: August 4, 2024, 17:55:21 UTC+05:30

Bucket Versioning [Edit](#)

Versioning is a means of keeping multiple versions of an object in the same bucket. You can now synchronize its behavior, metadata, and content across versions of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from multi-committed user actions and application failures. [Learn more](#)

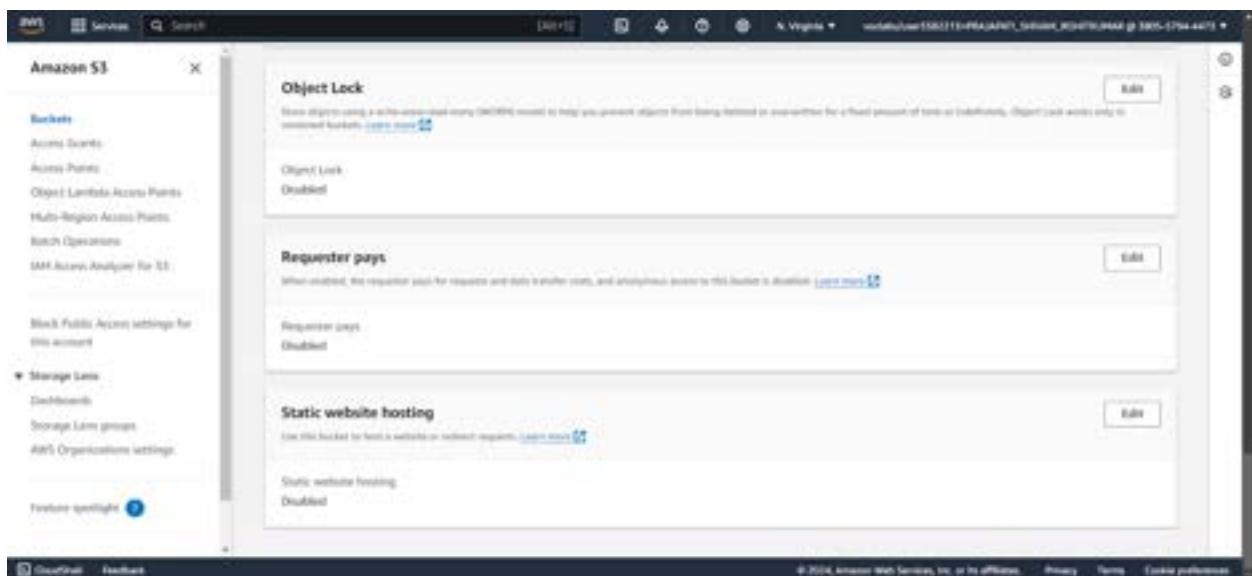
Bucket Versioning: **Disabled**

Multi-factor authentication (MFA) delete: An additional layer of security that requires multi-factor authentication for changing Bucket Versioning settings and permanently deleting object versions. To modify MFA delete settings, see the IAM User MFA, or the Amazon S3 REST API. [Learn more](#)

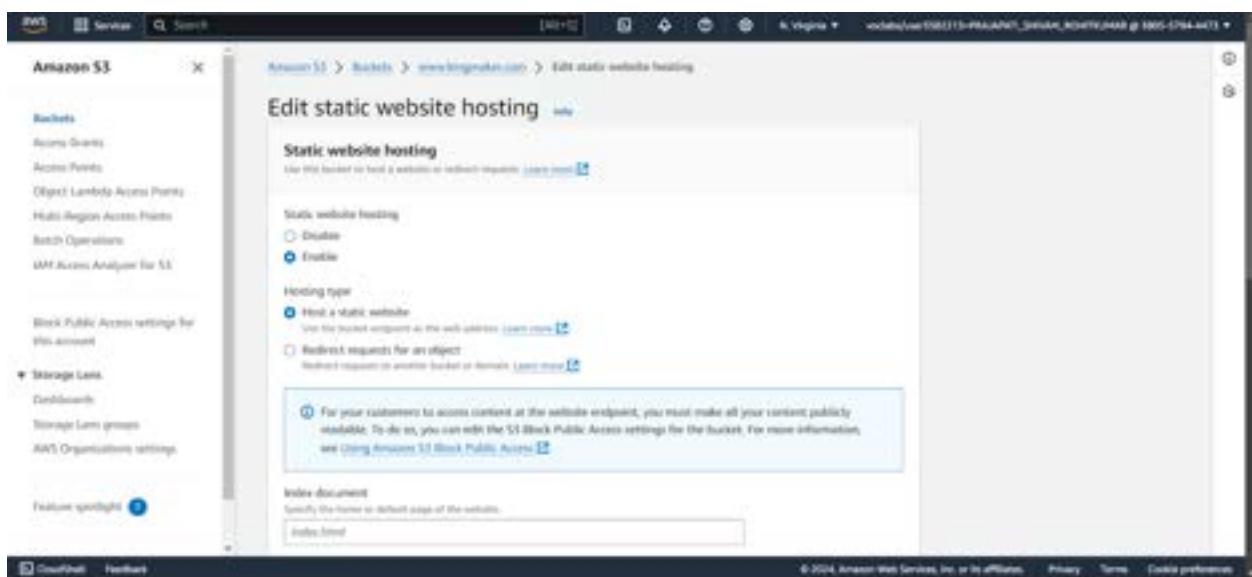
Disabled

© 2024 Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

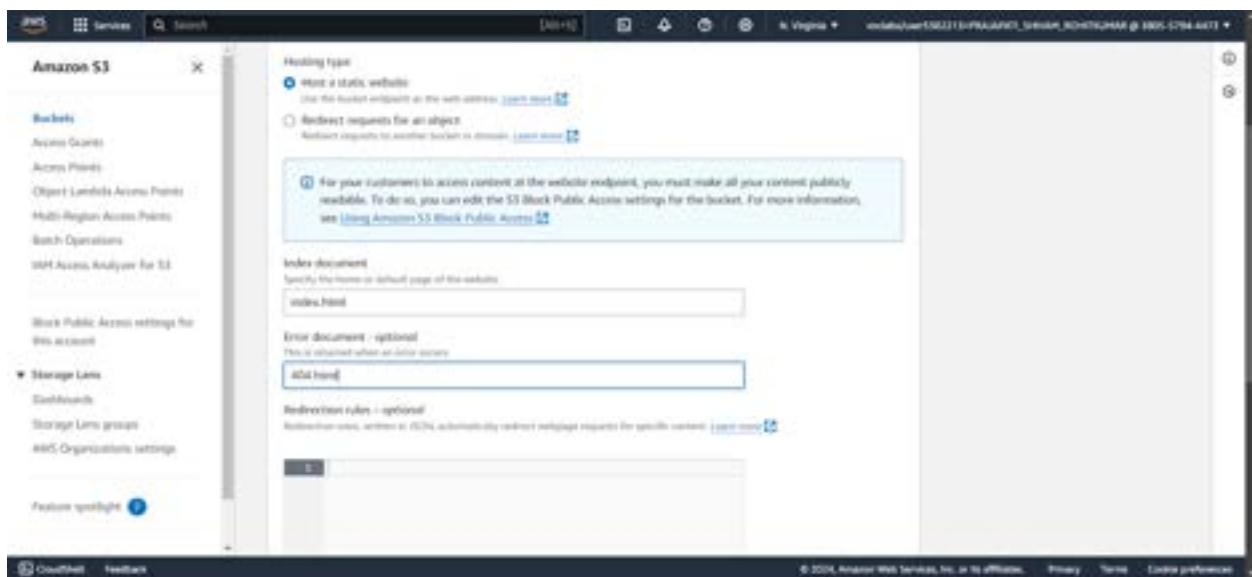
Step 5: Scroll down till you find Static website hosting, click on edit



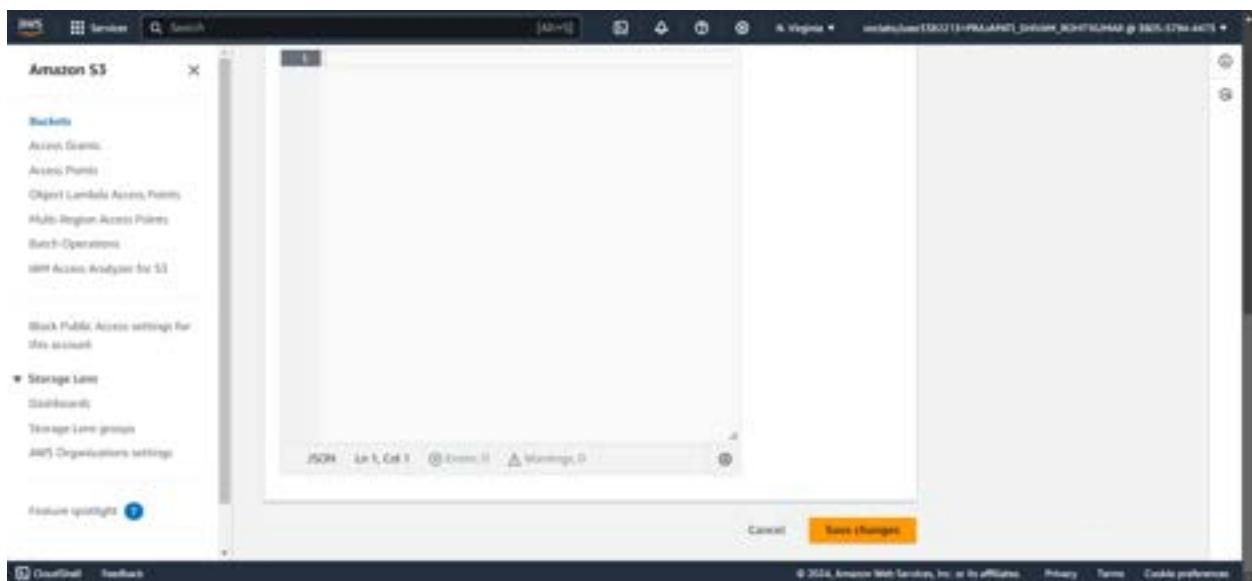
Step 6: Click on Enable static website hosting because we want to host our static website to the bucket

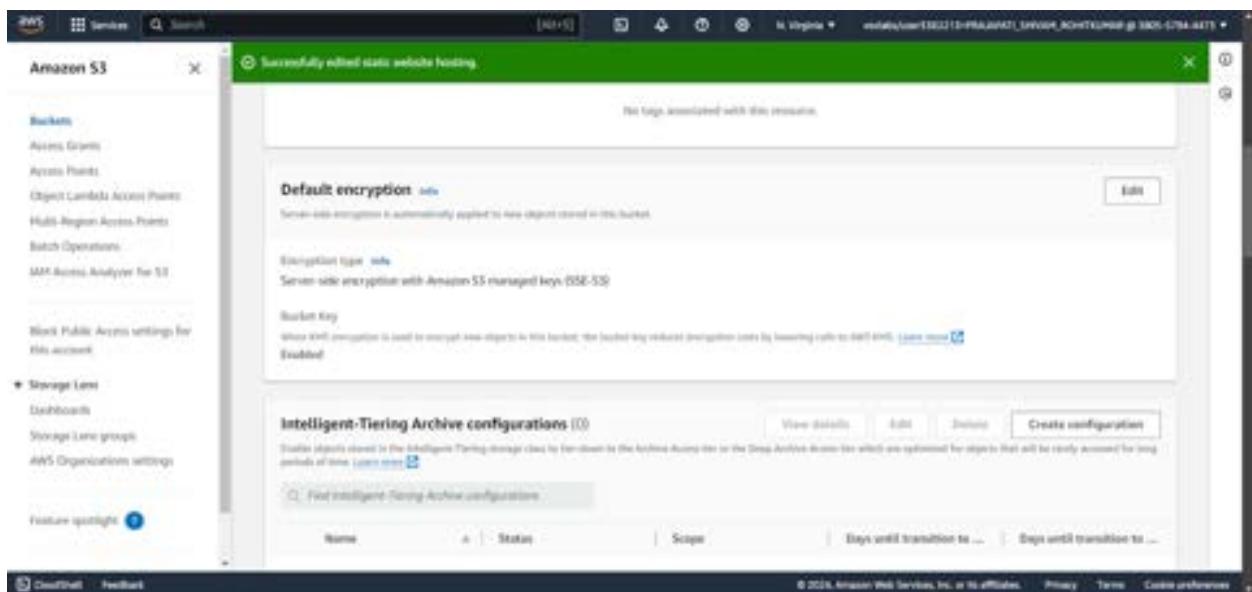


Step 7: Write the name of your document which you wanted to host on AWS from your local folder and in error document, give name as 404.html. Save your changes.

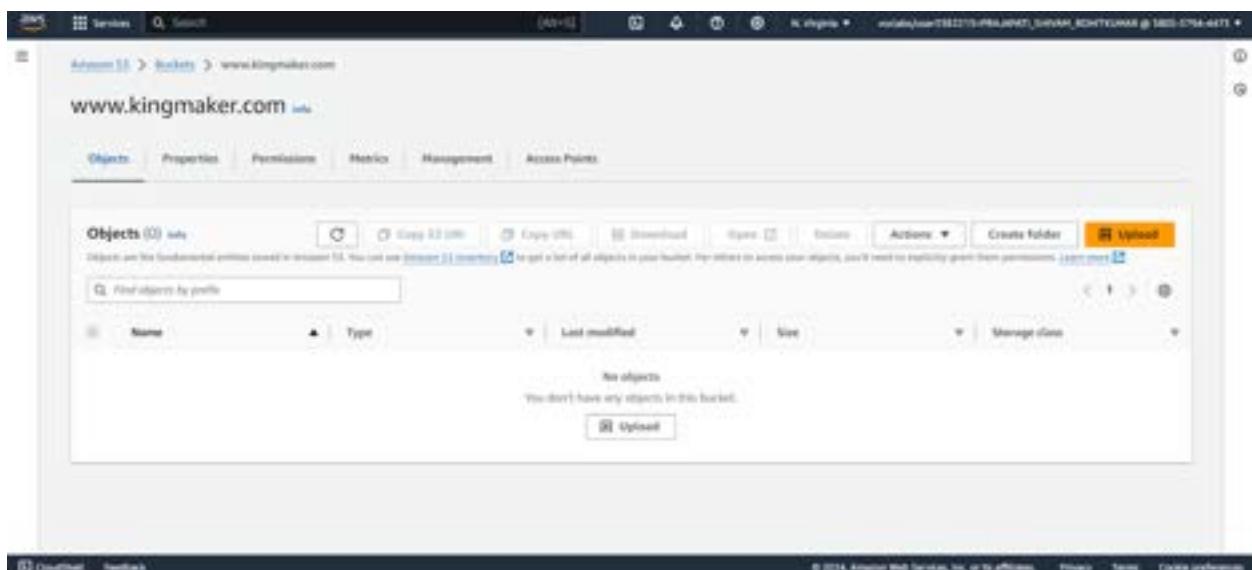


Step 8: Click on Save Changes

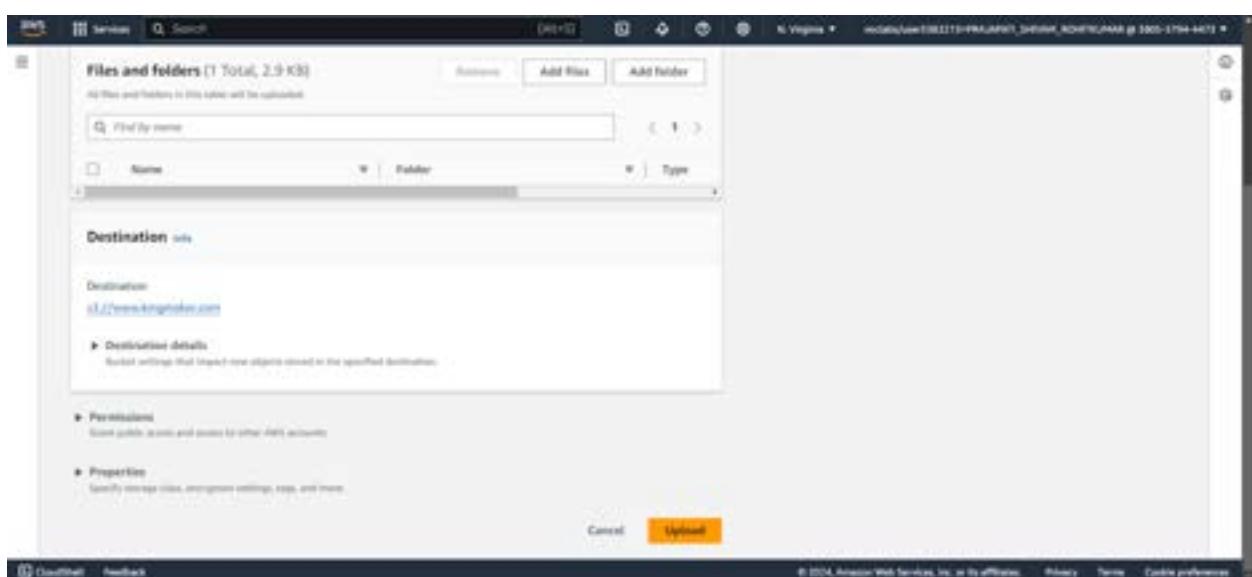
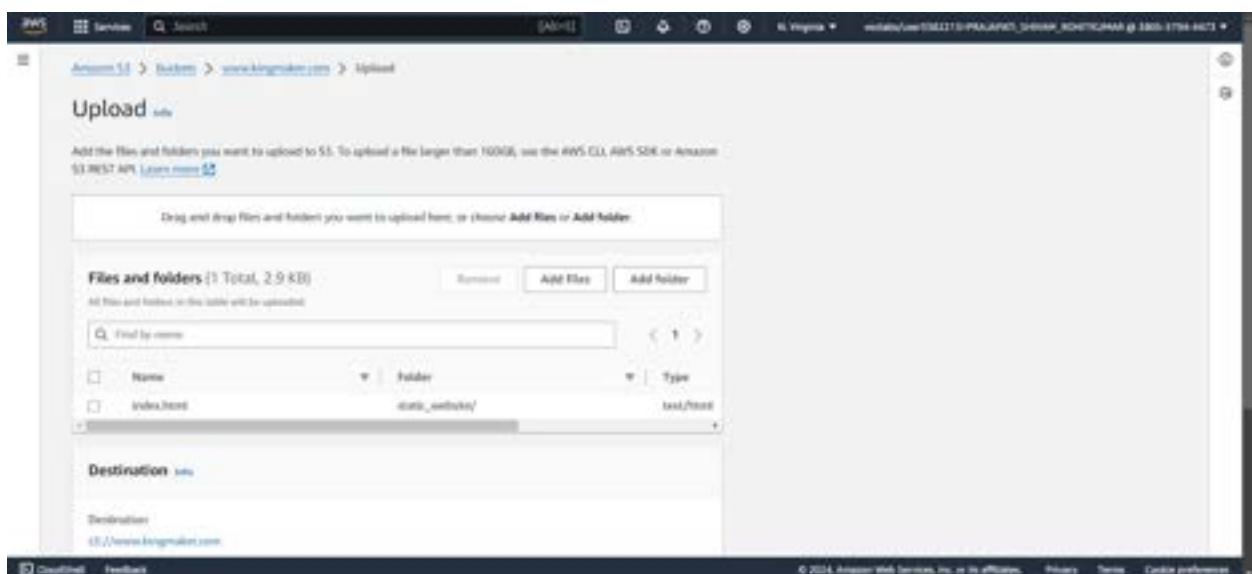




Step 9: Go to Objects tab and click on upload file to upload file to your bucket



Step 10: Click on Add files. Add all the files you want to upload. Then scroll down and click on Upload



Click on Upload

The screenshot shows the AWS S3 Objects upload successful page. At the top, there's a green banner with the message "Upload succeeded" and a link to "View details". Below the banner, a note says "The information below will no longer be available after you navigate away from this page." The main section is titled "Summary" and includes a table with two rows: "Destination" (s3://www.kingmaker.com) and "Status" (1 File, 2.9 KB (0%)). Below this is a navigation bar with "Files and folders" and "Configuration" tabs, where "Files and folders" is selected. A table titled "Files and folders [1 Total, 2.9 KB]" lists one item: "index.html" (static website, txt/html, 2.9 KB, Status: Succeeded). At the bottom, there are "Dashboard" and "Feedback" links, and a copyright notice: "© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences".

Step11: This will take you to the Objects screen. Switch to Properties, scroll down to Static web hosting. There you would find the link (Bucket website endpoint) to your website.

The screenshot shows the AWS S3 Bucket Properties page for a bucket named "www.kingmaker.com". On the left, there's a sidebar with "Buckets", "Access Grants", "Access Points", "Object Lambda Access Points", "Multi-Region Access Points", "Batch Operations", "VPC Access Analyzer for S3", "Block Public Access settings for this account", "Storage Lens" (with "Dashboards", "Storage Lens groups", and "AWS Organizations settings" sub-links), and a "Feature spotlight" section. The main content area has two sections: "Requester pays" (disabled) and "Static website hosting". Under "Static website hosting", it says "Use this bucket to host a website or redirect requests." Below this are fields for "Hosting type" (Bucket hosting) and "Bucket website endpoint". The endpoint is listed as "When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket." Below this is a link to "Learn more" and a URL: "http://www.kingmaker.com (www.kingmaker.com)" with a copy icon. At the bottom, there are "Edit" buttons for both sections, and a copyright notice: "© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences".

Step12: Open the link. It will show a 403 forbidden error screen as the contents of the bucket are not available for the public users. To change this, go to Permissions tab, go to Block public access and click on edit

403 Forbidden

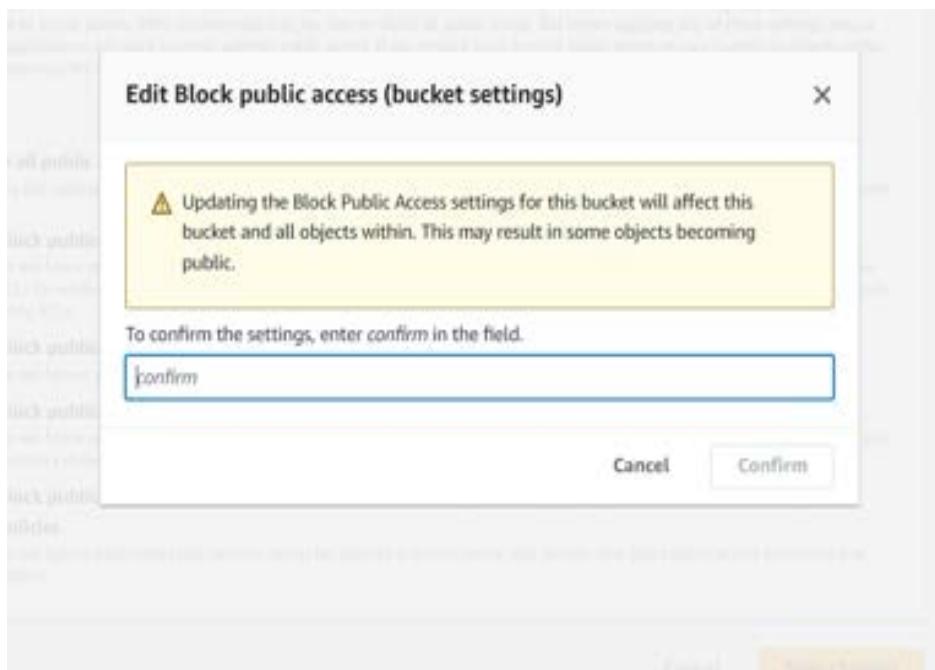
- Code: AccessDenied
- Message: Access Denied
- RequestId: 8TQ4EGP4TK06MVPB
- HostId: hF+ToadQuoCuDM8H+iFRsXdA28TGp+xikYbjb4CICS/t+3it4ihA/tvgA1Xrlxo+JL5AhkT6hJs=

An Error Occurred While Attempting to Retrieve a Custom Error Document

- Code: AccessDenied
- Message: Access Denied

Step 13: Uncheck the Block all public access checkbox and click on save changes

The screenshot shows the AWS S3 console. On the left, there's a sidebar with options like Buckets, Access Grants, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, and Block Public Access settings for this account. The main area shows the bucket 'www.kingmaker.com'. At the top, there are tabs for Objects, Properties, Permissions (which is selected), Metrics, Management, and Access Points. Below the tabs, there's a 'Permissions overview' section with a note about Access Logging. Under the 'Block public access (bucket settings)' section, there's a checkbox labeled 'Block all public access' which is currently set to 'Off'. A note below it says 'Individual Block Public Access settings for this bucket'. At the bottom of the page, there are buttons for 'Save changes' and 'Cancel'.



Type confirm in the given field and click on confirm button as shown above

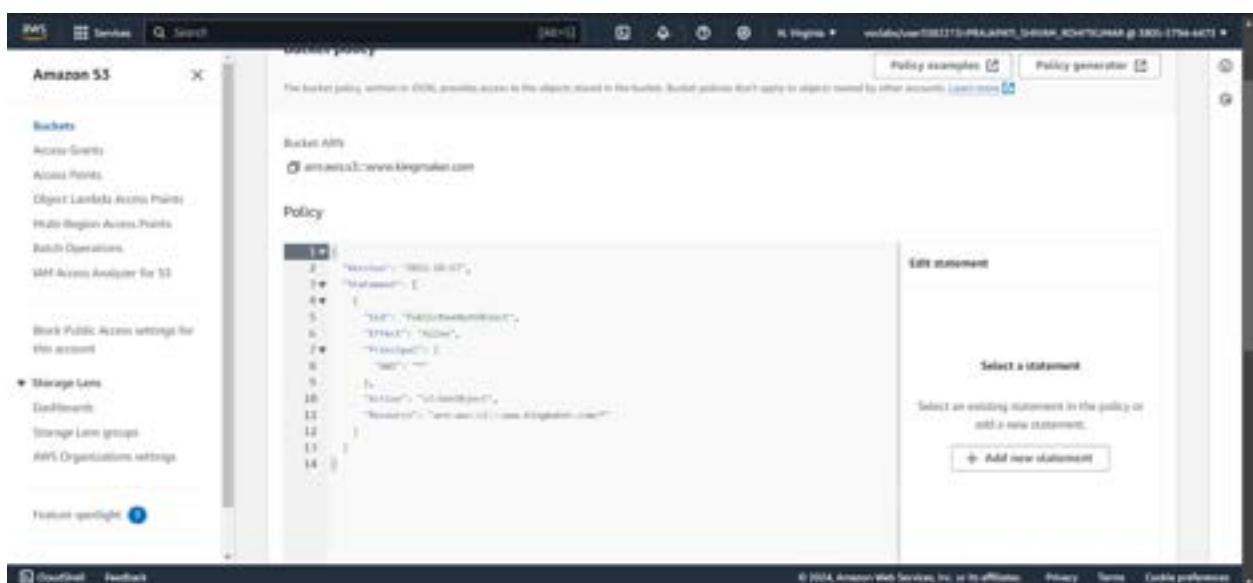
Step 14: Successfully Changed the Settings

The screenshot shows the AWS S3 console. On the left, there's a sidebar with options like 'Amazon S3', 'Services', 'Search', 'Bucket', 'Access Grants', 'Access Points', 'Object Lambda Access Points', 'Multi-Region Access Points', 'Batch Operations', 'AWS Access Analyzer for S3', 'Block Public Access settings for this account', 'Storage Lens', 'DataLens', 'Storage Lens groups', 'AWS Organizations settings', and 'Feature spotlight'. The main area has a green banner at the top that says 'Successfully edited Block Public Access settings for this bucket.' Below it, the URL 'www.kingmaker.com' is shown. The 'Permissions' tab is active. Under 'Permissions overview', there's a section for 'Access findings' which says 'Access findings are provided by AWS server access analyzer. Learn more about how AWS protects findings with View analyzer for my assets.' Below that is the 'Block public access (bucket settings)' section. It says 'Public access is granted to buckets and objects through access-control lists (ACLs), bucket policies, account-level policies, or off. In order to remove that public access to all your S3 buckets and objects is blocked, turn on Block all public access.' It also says '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 with public access. If you require controlled public access to your buckets or objects within your account, you can customize the individual settings below to suit your specific storage use cases.' There's a 'Edit' button next to the settings. At the bottom, there are links for 'Feedback', '© 2024, Amazon Web Services, Inc. or its affiliates.', 'Privacy', 'Terms', and 'Cookie preferences'.

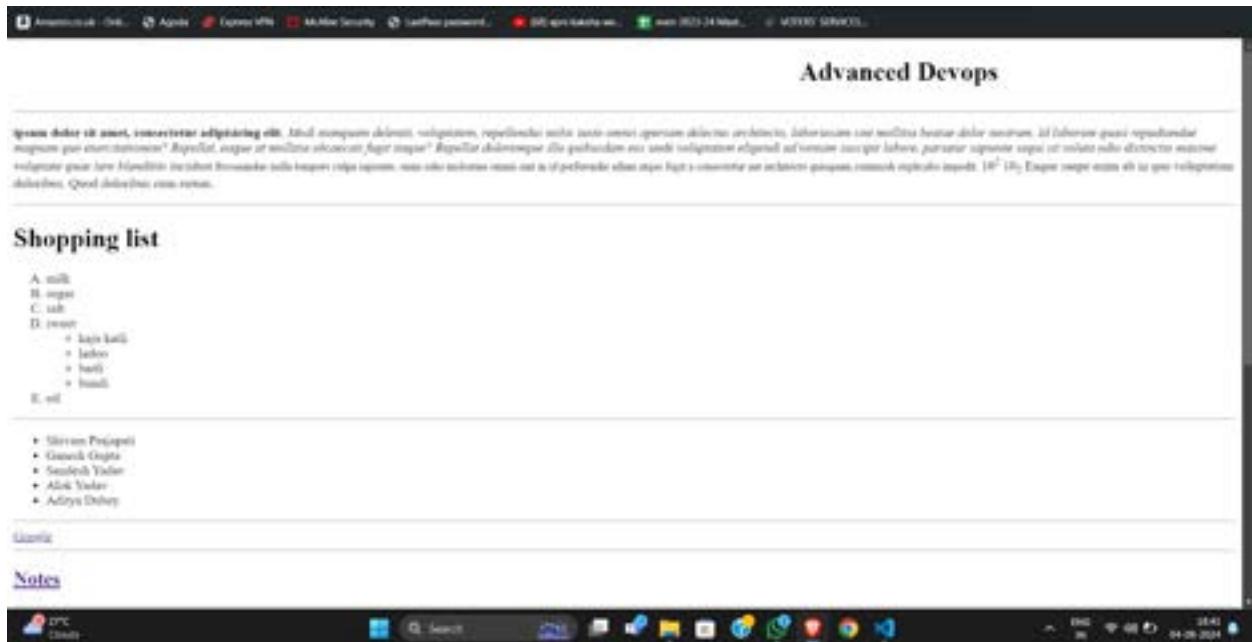
Step 15: Scroll down to bucket policy and click edit and paste the code from given Github Link
<https://gist.github.com/Savjee/b4b3a21d143a30e7dc07>

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

Paste this code snippet in the policy textarea. Replace YOUR-BUCKET-NAME-HERE with the name you have given to your bucket. Save the changes



Step 16: Now reload the website. You can see your website



CONCLUSION:

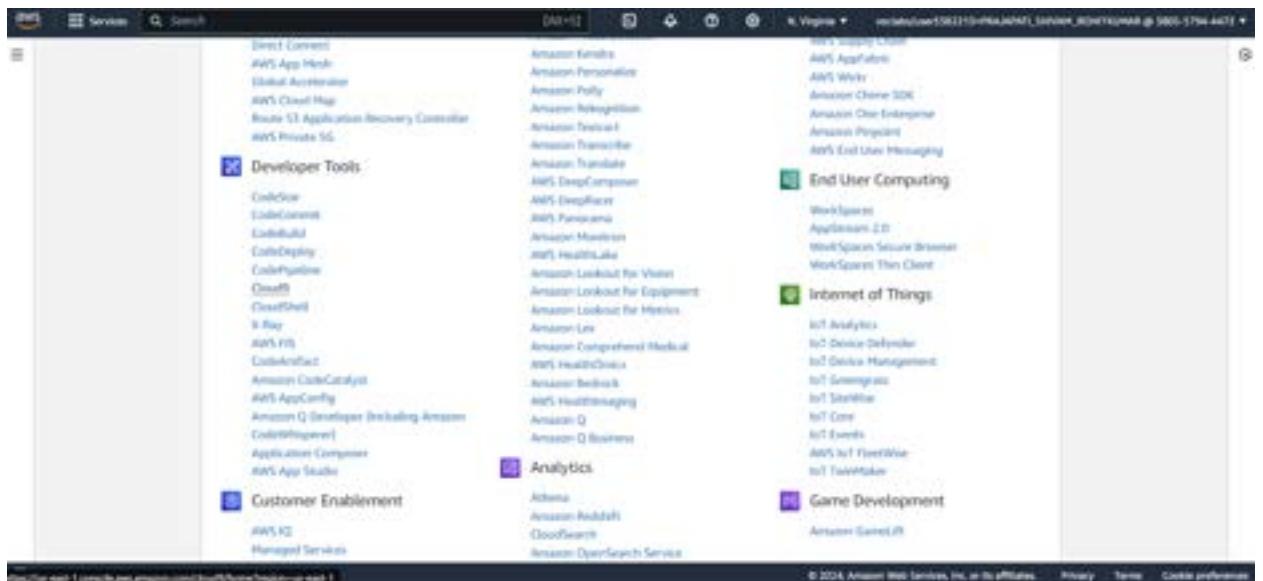
In this experiment we have hosted the local website on the XAMPP server. The user can access to his website by using the browser then XAMPP server gives the responses to the users in the form of a website which was locally hosted to XAMPP server.

Experiment No: 1(B)

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

Step 1: Set up a Cloud9 environment.

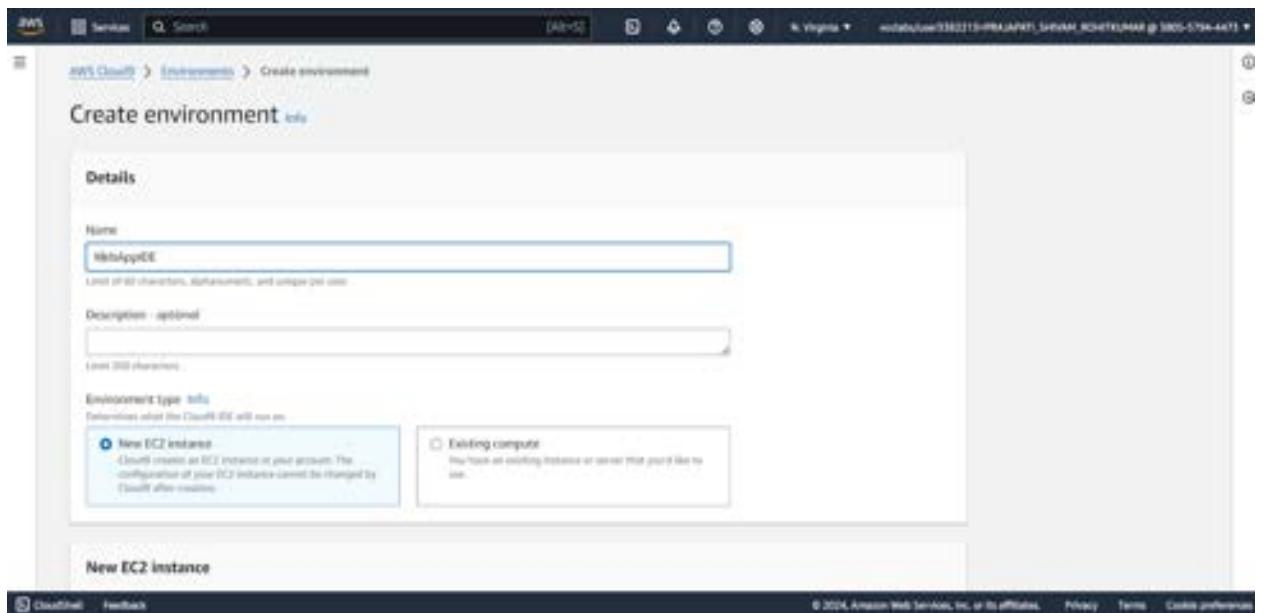
- 1) Go to Cloud9 services under developers tool in All services



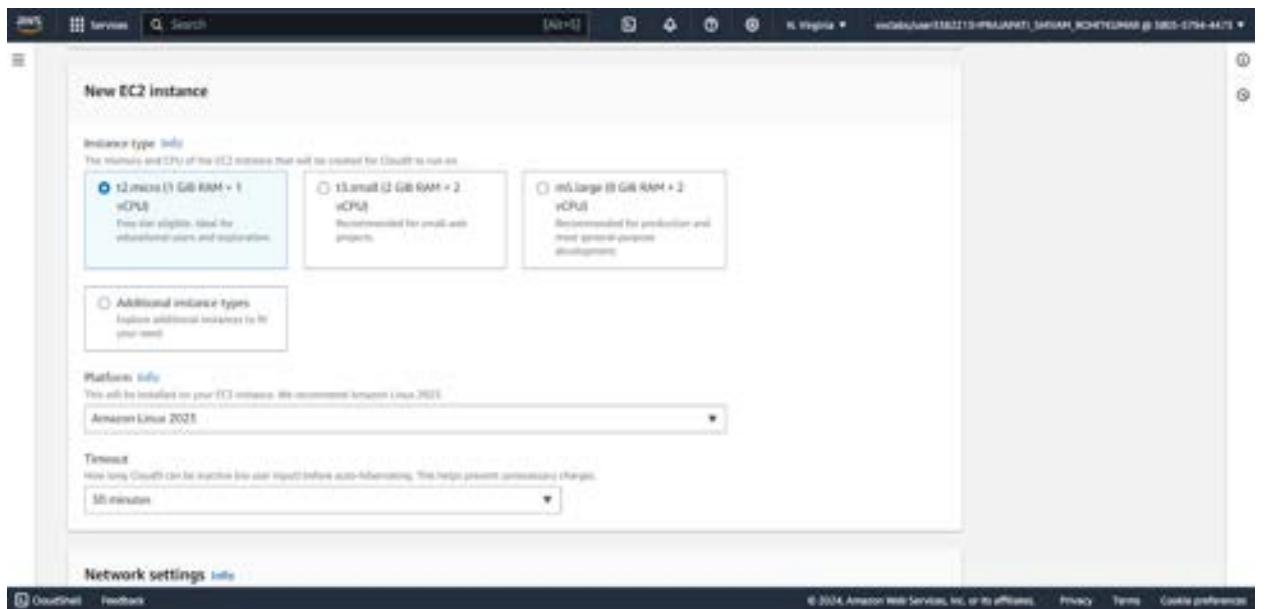
- 2) Click on create environment



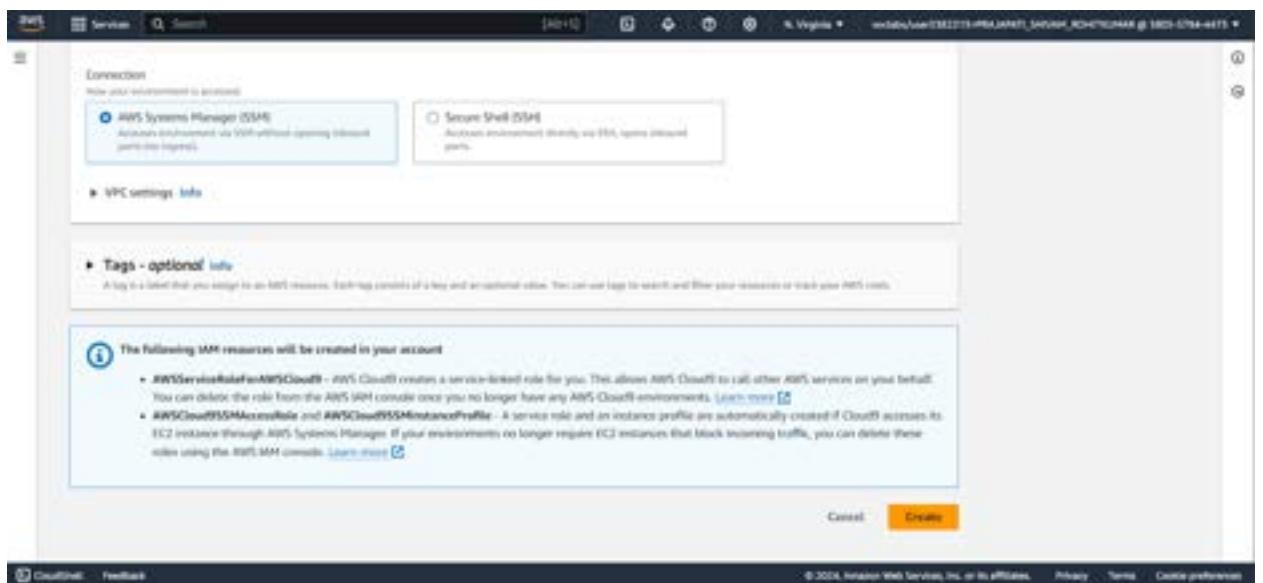
- 3) Give the name to your Environment ,keeping the other settings as default like environment type should be New EC2 instance



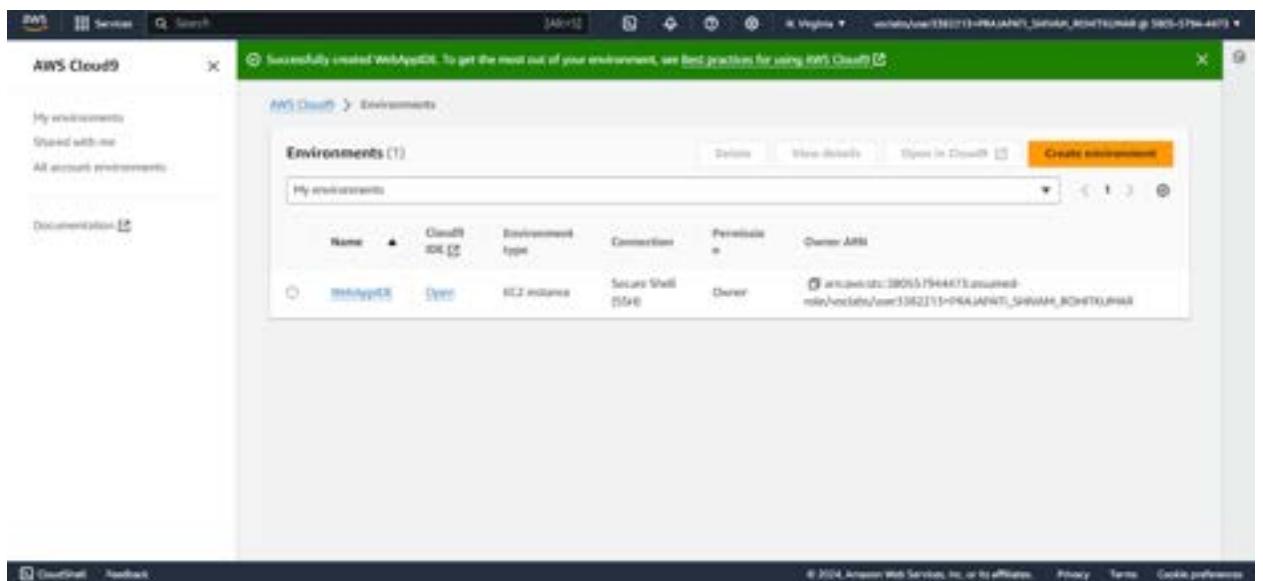
- 4) Select the correct platform type as shown below and keep the others details as default like instance type as t2.micro which gives the user 1GB RAM + 1 Virtual CPU



5) Click on SSH under connection type in network settings if we go for AWS Manager(SSM) then it won't allow to create an environment then click on Create



6) Successfully created the environment so now click on open



Step 2: Creating IAM user.

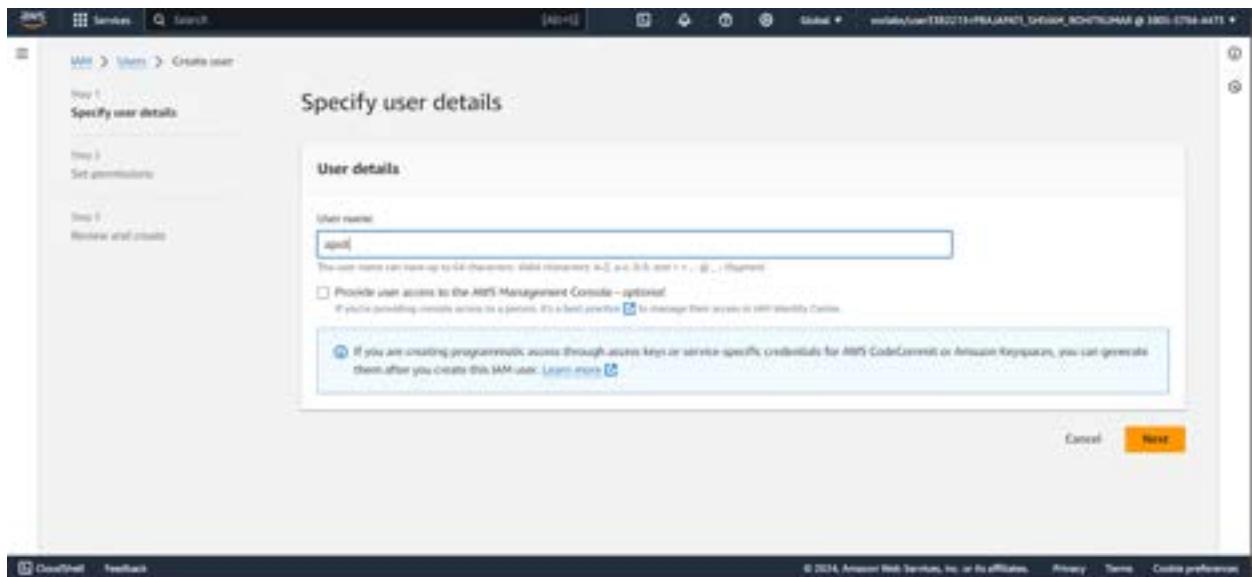
- 1) Search IAM on the services search bar and open it. Click on Create User

The screenshot shows the AWS Identity and Access Management (IAM) dashboard. On the left, there's a sidebar with navigation links for Dashboard, User groups, Users, Roles, Policies, Identity providers, and Account settings. Under 'Access management', there are links for Access Analyzer, External access, Unused access, Analysis settings, and Credential report. The main area is titled 'IAM Dashboard' and contains a summary box with the following counts: User groups (0), Users (0), Roles (20), Policies (4), and Identity providers (0). Below this, there's a 'What's new' section with four bullet points about AWS IAM Access Analyzer and AWS IAM Roles Anywhere. To the right, there's a 'AWS Account' section with details like Account ID (500015794473), Account Alias (Create), and a sign-in URL (https://signin.aws.amazon.com/console). At the bottom, there's a 'Tools' section with a 'Policy simulator' link. The footer includes copyright information for 2024 and links for Privacy, Terms, and Cookies.

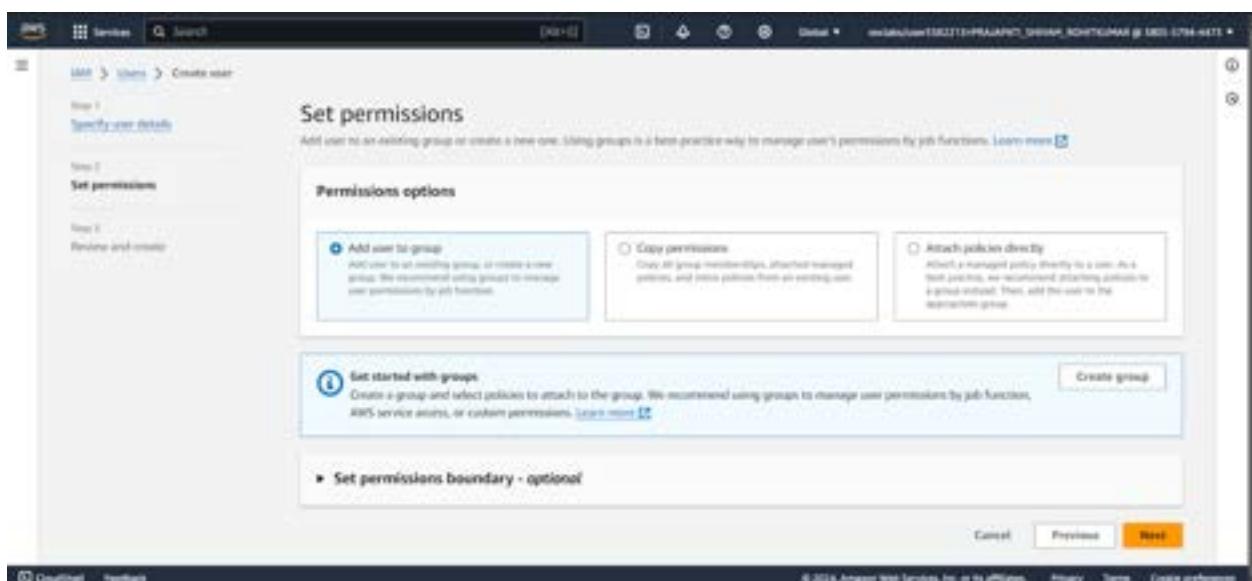
- 2) Click on the create user

The screenshot shows the 'Users' page under the IAM service. The left sidebar has the same navigation as the previous dashboard. The main area is titled 'Users [0] info' and contains a table header with columns: User name, Path, Group, Last activity, HSM, Password last change, and Consists last sign-in. A note below the table says 'No resources to display'. The footer is identical to the previous screenshot, showing 2024 copyright and various links.

3) Write the name of the user you want to add and click on next



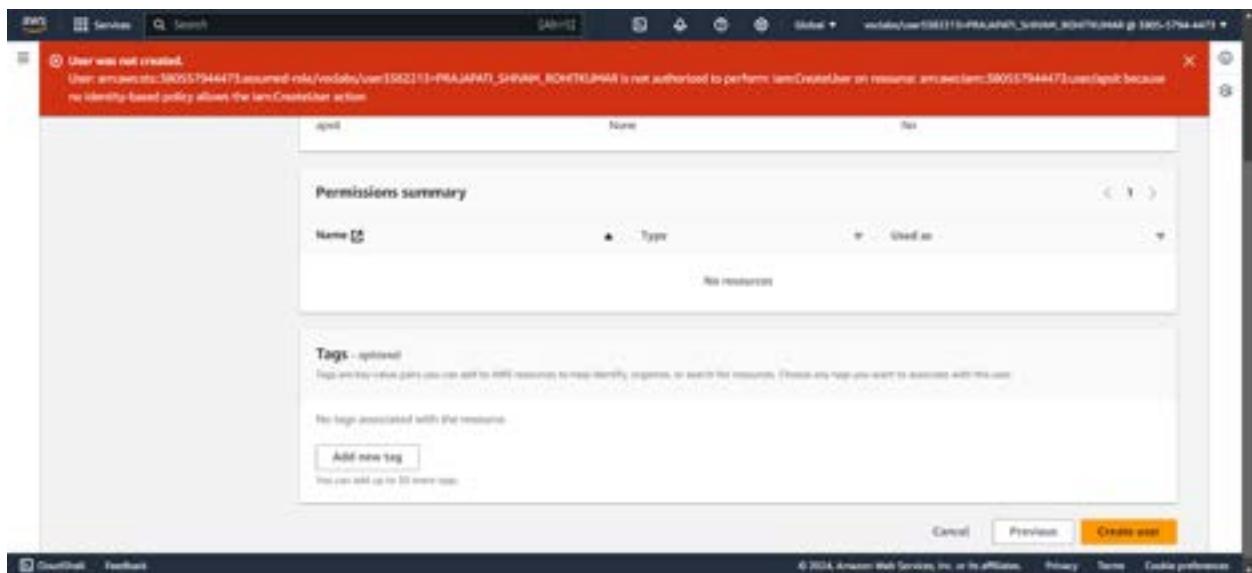
4) Select add User to Group. If there are no user groups on your accounts, you will have to create one. Click on Create Group. Click on the drop down menu of the set permissions boundary



5) Click on the checkbox and search for cloud9 under permissions policies ,click on next

Policy name	Type	Attached entities
AWSCloud9Administrator	AWS-managed	0
AWSCloud9EntitlementNumber	AWS-managed	0
AWSCloud9RootIdentity	AWS-managed	0
AWSCloud9UserAccessProfile	AWS-managed	0
AWSCloud9Key	AWS-managed	0

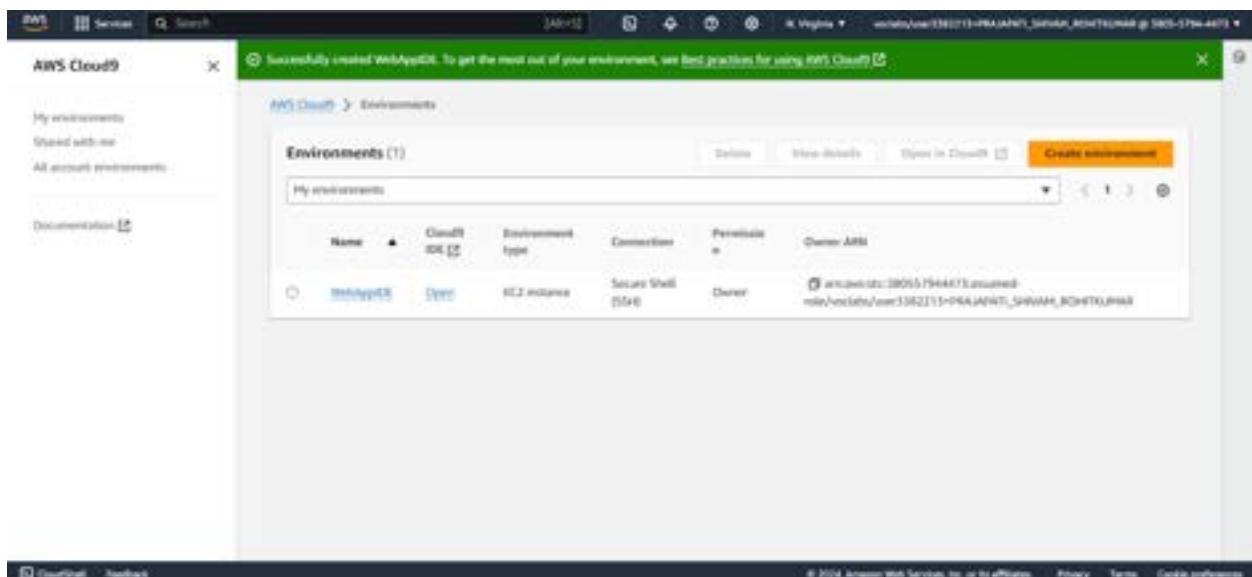
6) Scroll down and click on create user



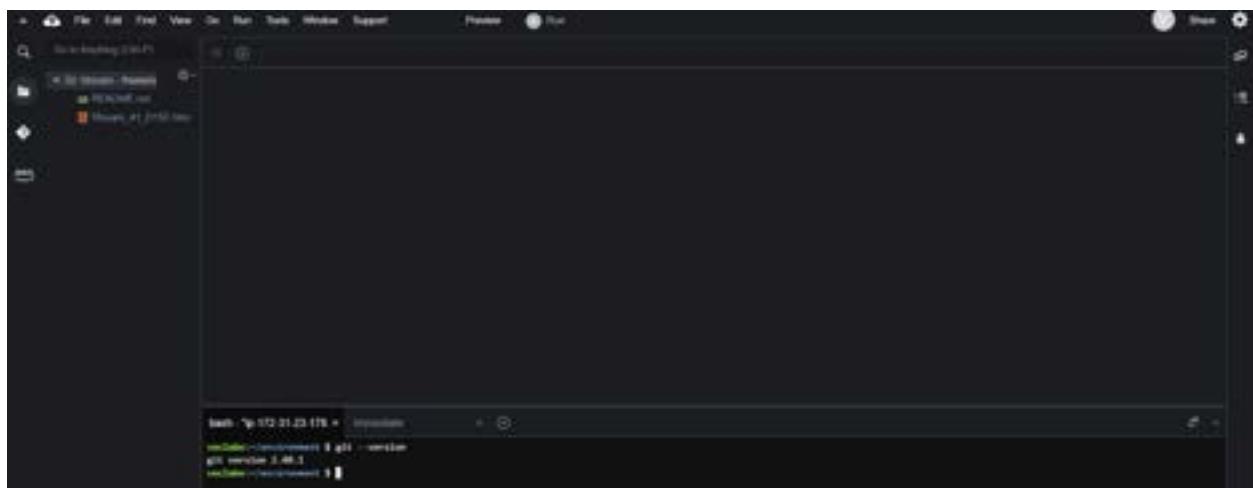
When we go to add user to a group, the AWS Academy account throws an error as we do not have the permissions to create a group. So we have to use our personal AWS account for this part.

Step 3: Working on Cloud9 IDE

- 1) Go to Cloud9 services. Click on Open under Cloud9 IDE

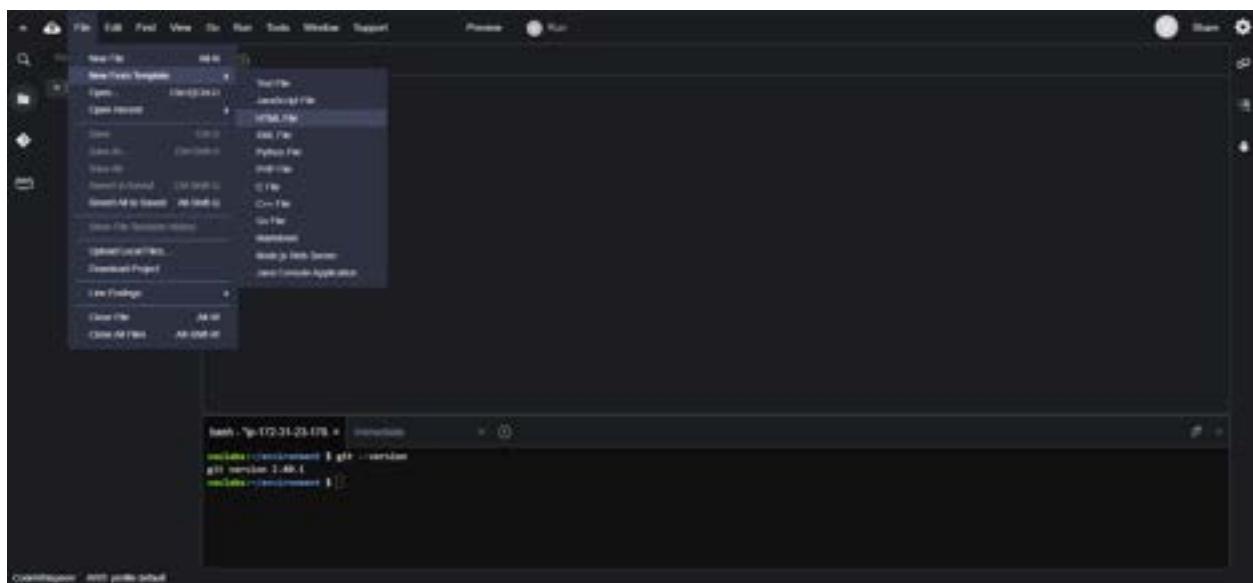


2) This is the Cloud9 IDE interface. The major part of the screen is the coding IDE. There is a command console just below it. For example, the command git --version is run.



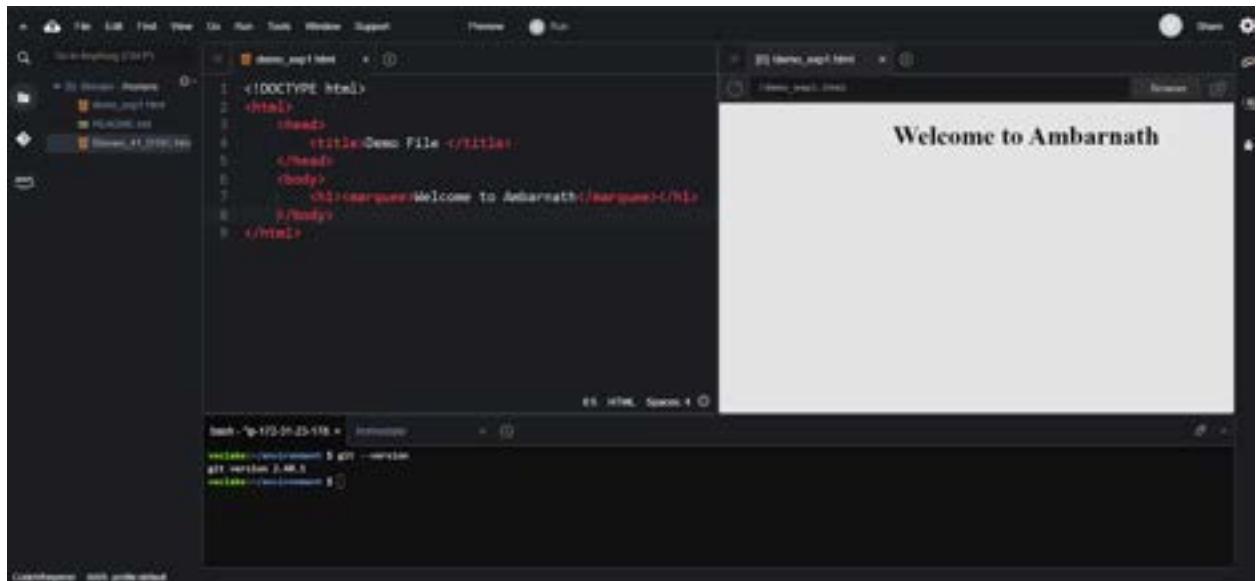
A screenshot of the Cloud9 IDE interface. The top menu bar includes File, Edit, Find, View, Sir, Run, Tools, Webview, Support, Profile, and Run. On the left is a sidebar with 'File Explorer (VS Code)' showing a project named 'Project1' with files 'index.html' and 'index.js'. The main workspace is dark-themed. At the bottom is a terminal window titled 'Terminal' with the command 'git --version' entered and its output displayed: 'git version 2.46.1'.

3) To add a file, click on file. For this experiment, we are to add an HTML file. So go to File → New From Template → HTML file. This gives a basic HTML template on the coding IDE

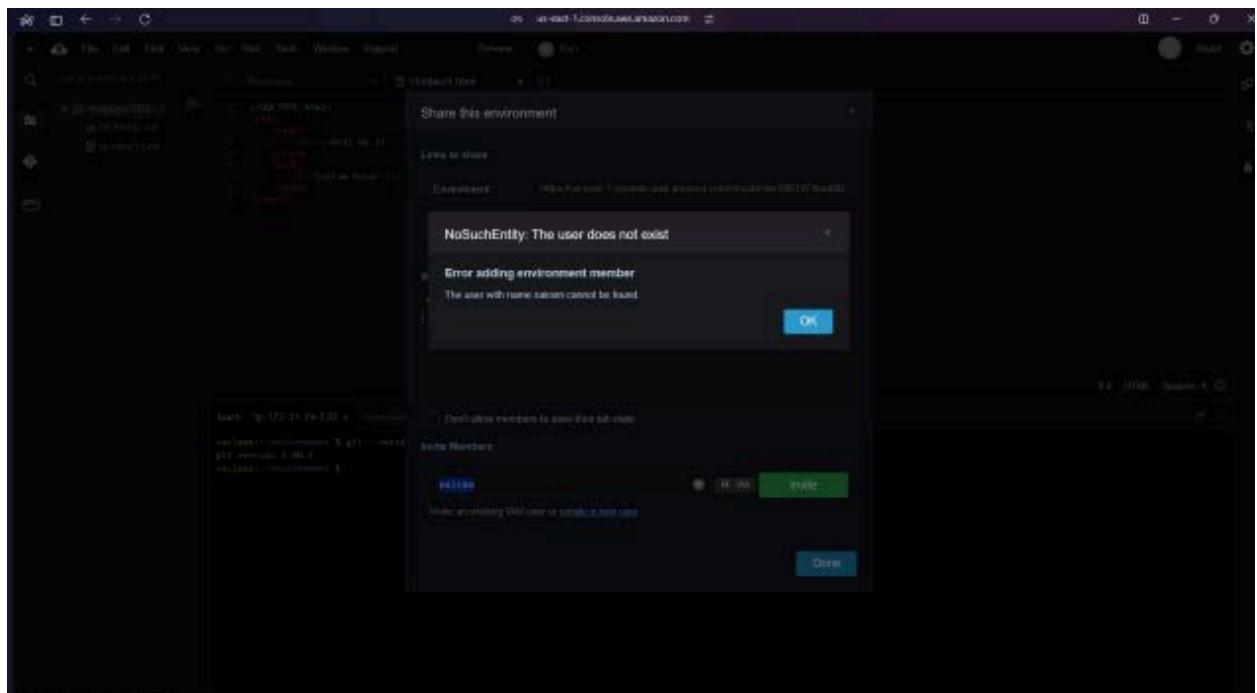


A screenshot of the Cloud9 IDE interface. The top menu bar includes File, Edit, Find, View, Sir, Run, Tools, Webview, Support, Profile, and Run. A context menu is open over the 'File' menu item, with 'New From Template' highlighted. Other options in the menu include New File, Open, Open Recent, Save, Save As, Save All, Recent Editors, Smart Move Editor, Close File, Recent Editors, Upload Local Files, Download Project, Use Folders, Close File, Close All Tabs, and Exit. The main workspace is dark-themed. At the bottom is a terminal window titled 'Terminal' with the command 'git --version' entered and its output displayed: 'git version 2.46.1'.

4) Make a basic website on the HTML template and save it.



After saving, on the toolbar towards far right, click on Share. Then put the username that you had put during creating IAM user.



Here, it gives an error as Cloud9 was created on the academy account where creating an IAM group is not available, meanwhile on the personal account, the services of Cloud9 have been deprecated. So currently, it is not possible to integrate the cloud9 and IAM parts of the experiment.

CONCLUSION:

In this part of the experiment we have hosted the website on the AWS cloud using the S3 services .For hosting of the website on the local machine or on the cloud does not require to use the personal account compulsorily ,In S3 service we need to create a bucket and then store our local file/folder to it. After doing required configuration we can globally access our website

Experiment No : 3

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

PREREQUISITES:

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

Master Node Security Group:

The screenshot shows the AWS CloudFormation interface for creating a new stack. The current step is 'Create security group'. The 'Basic details' section includes a 'Security group name' field with 'Master-node' and a 'Description' field with 'Security group for master node'. A 'VPC' field is set to 'vpc-0f0ff0f22f0a0a2e'. The 'Inbound rules' section is expanded, showing a table with eight rows of rules. Each row has columns for 'Type' (All traffic), 'Protocol' (tcp), 'Port range' (22, 22, 22, 22, 22, 22, 22, 22), and 'Source' (0.0.0.0/0). The 'Description' column for each rule is empty. The bottom of the screen shows the AWS navigation bar and footer.

The screenshot shows the 'Inbound rules' table for the 'Master-node' security group. The table has eight rows, each representing a rule. The columns are: Type (All traffic), Protocol (tcp), Port range (22, 22, 22, 22, 22, 22, 22, 22), and Source (0.0.0.0/0). The 'Description' column for all rules is empty. The bottom of the screen shows the AWS navigation bar and footer.

Type	Protocol	Port range	Source	Description
All traffic	tcp	22	0.0.0.0/0	
All traffic	tcp	22	0.0.0.0/0	
Custom TCP	tcp	2222	0.0.0.0/0	
Custom TCP	tcp	2222	0.0.0.0/0	
Custom TCP	tcp	2222	0.0.0.0/0	
All TCP	tcp	443	0.0.0.0/0	
Custom TCP	tcp	443	0.0.0.0/0	

Worker Node Security Group :

Create security group

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name [Info](#)
Node-Worker
Name cannot be edited after creation.

Description [Info](#)
Security group for worker node

VPC [Info](#)
vpc-04f4adfb001dca5c08

Inbound rules [Info](#)

Type	Protocol	Port range	Source	Description - optional
All traffic	All	80	Anywhere	
SSH	TCP	22	Anywhere	
Custom TCP	TCP	10240	Anywhere	
All TCP	TCP	0-8888	Anywhere	
Custom TCP	TCP	30000 - 52757	Anywhere	
HTTP	TCP	80	Anywhere	

Inbound rules [Info](#)

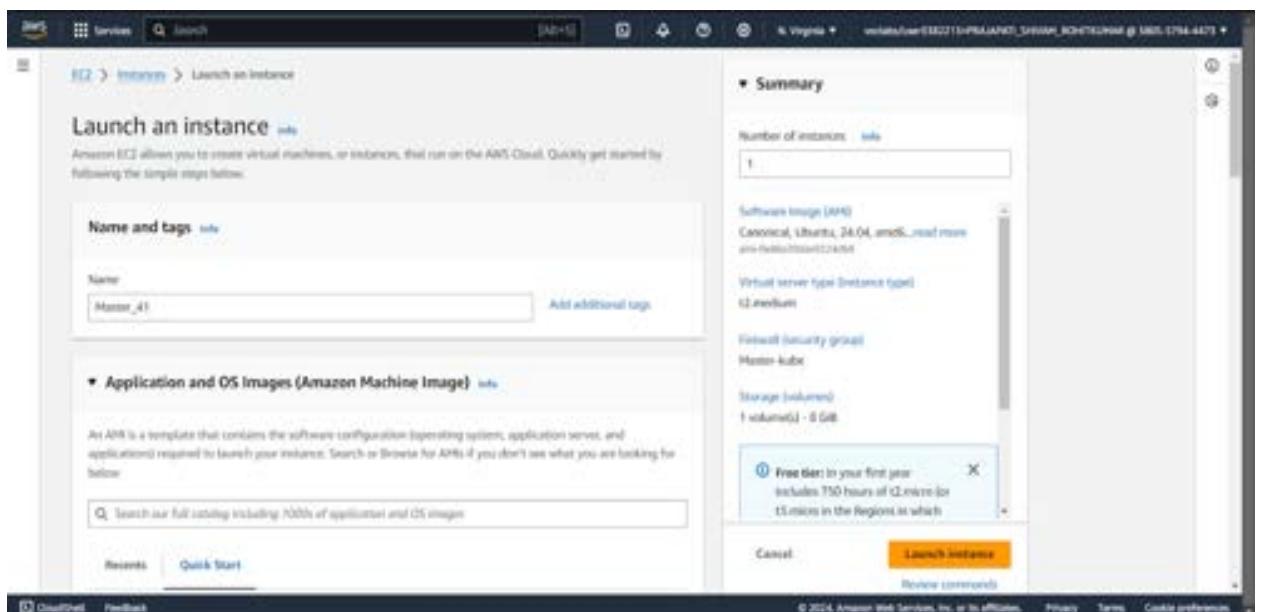
Type	Protocol	Port range	Source	Description - optional
All traffic	All	80	Anywhere	
SSH	TCP	22	Anywhere	
Custom TCP	TCP	10240	Anywhere	
All TCP	TCP	0-8888	Anywhere	
Custom TCP	TCP	30000 - 52757	Anywhere	
HTTP	TCP	80	Anywhere	

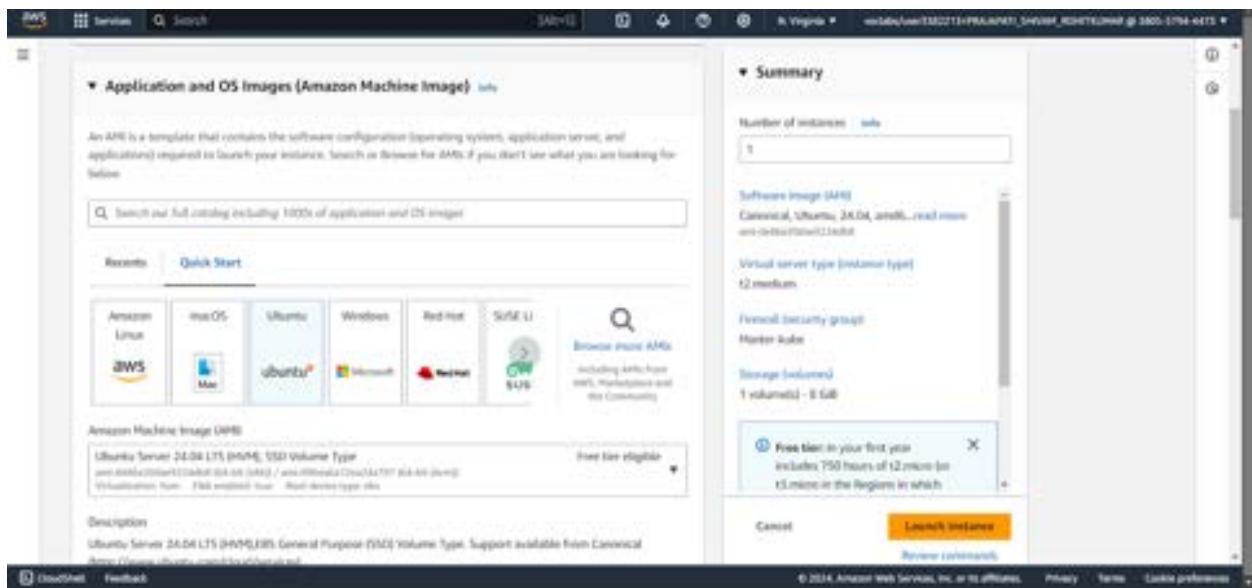
Added Required Inbound rules for worker nodes

Step 1: Access your AWS Academy or personal account and create **three new EC2 instances**. For the AMI, choose **Ubuntu** and set the **instance type to t2.medium**. Generate an RSA key in .pem format and move the downloaded key to a new folder for eg: **Newfolder**; you can either use three separate keys or one shared key for all instances ,in my case I have made three different keys .

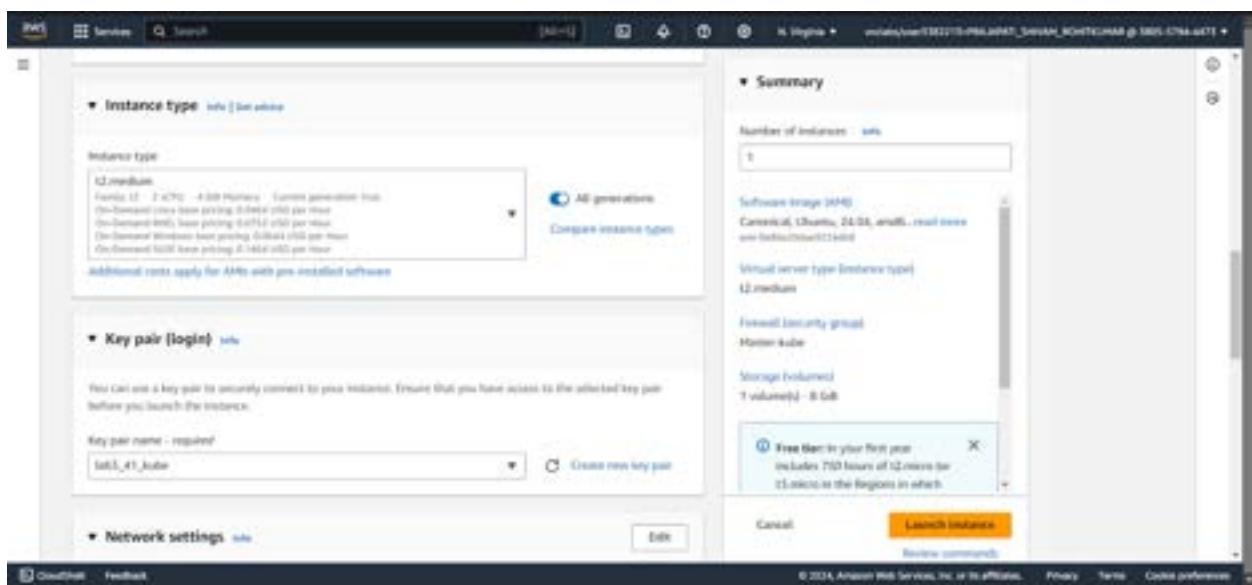
It's essential to select t2.medium, as it includes at least 2 CPUs, which are required for this setup.. Additionally, make sure to select security groups from the ones already available i.e master node and worker node will select their own security groups respectively

Master Node:

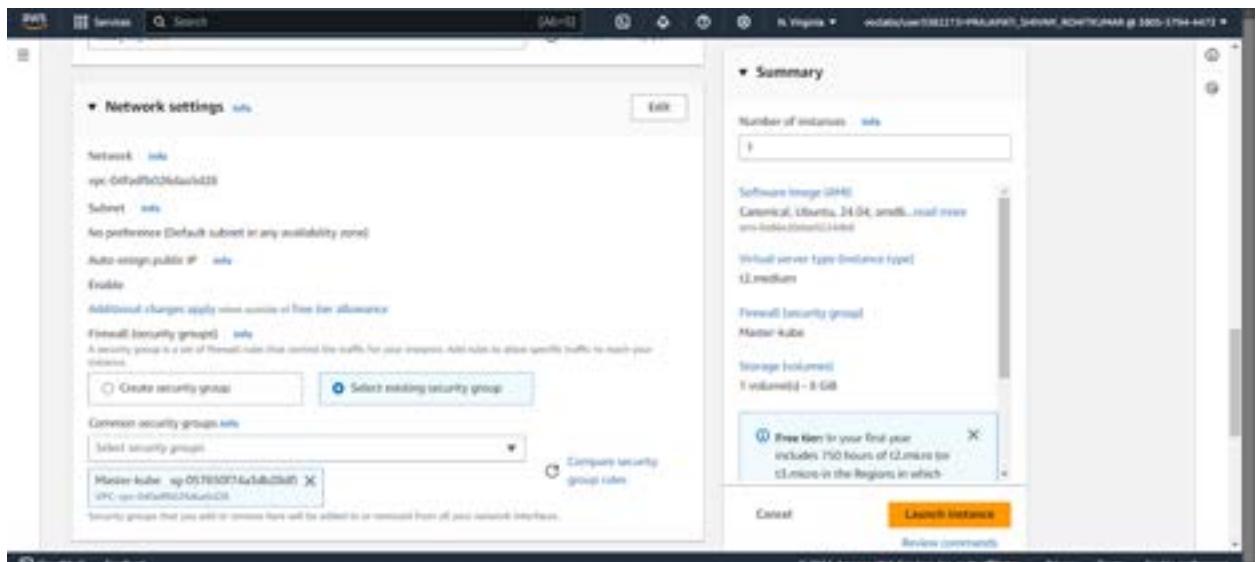




AMI as Ubuntu

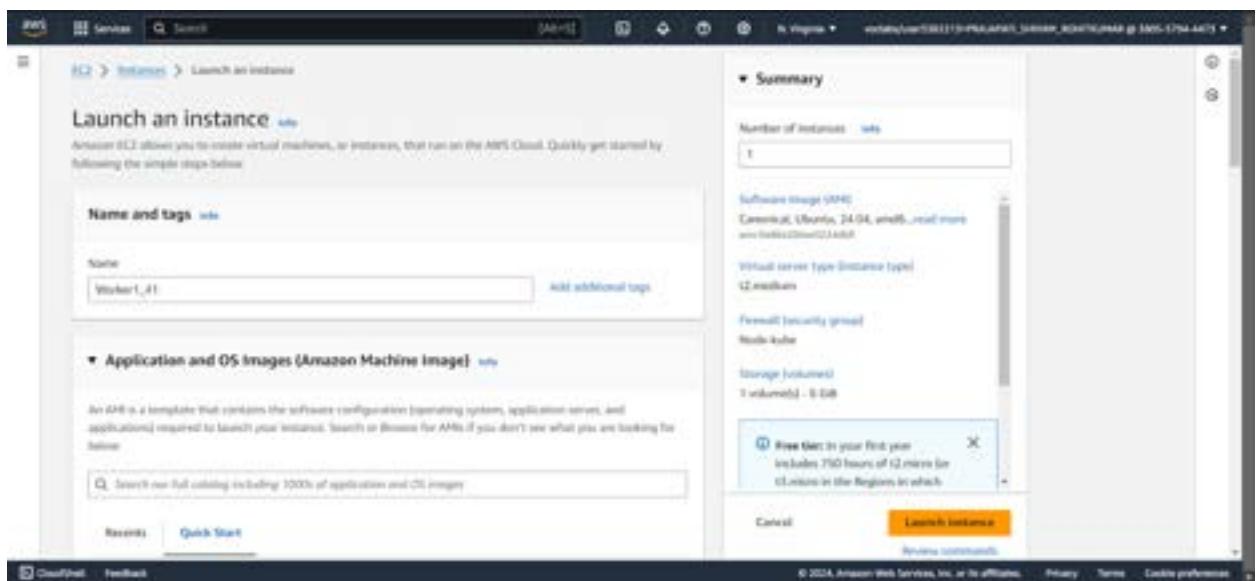


Selected Appropriate Instance type and created key pair login

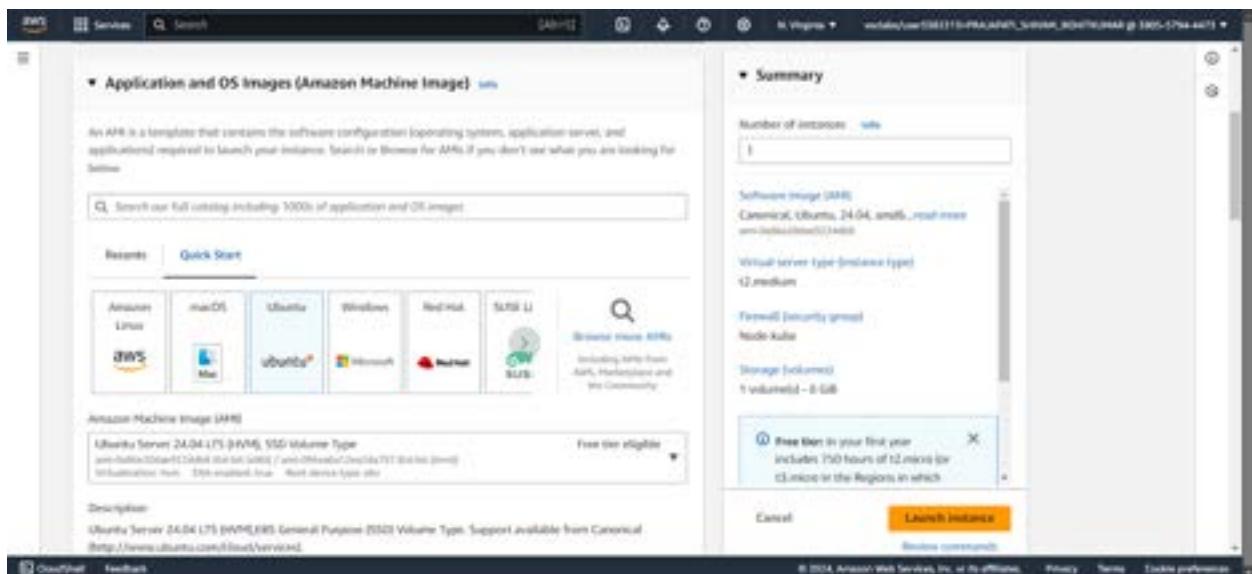


Selected Required Security Group

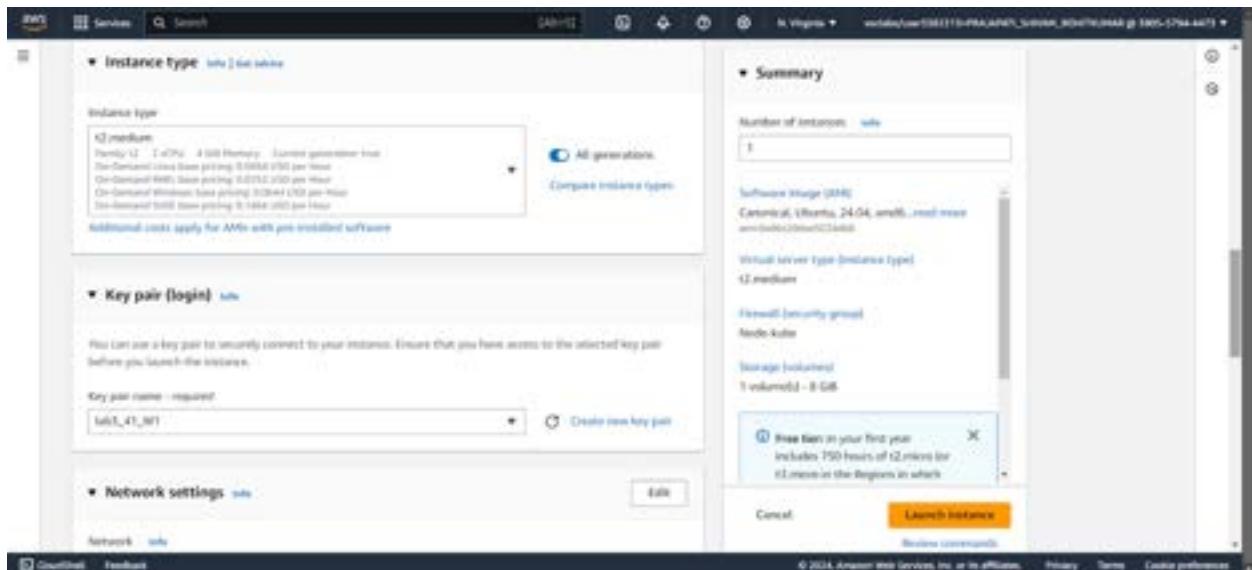
Worker Node 1:



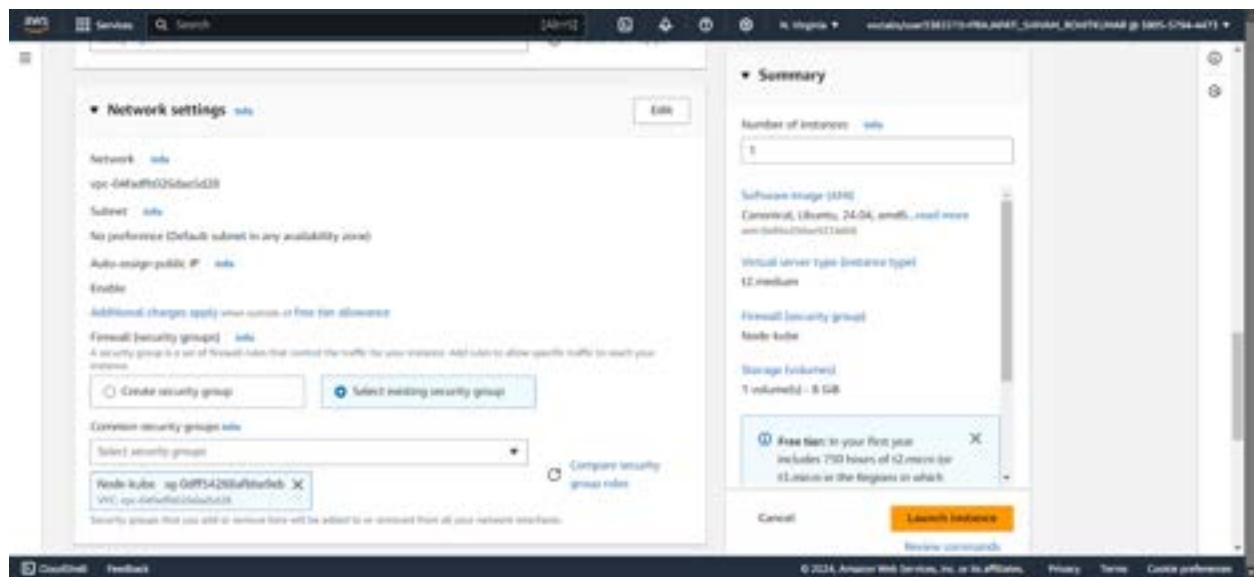
Give a name to your instance .



AMI as Ubuntu



Created the Key pair login



Selected required Security Group

Worker Node 2: Create Worker2_41 instance similarly to previous worker node created .Give Name to your Instance, Select ubuntu as Ami, t2.medium as instance type, create key-pair login as Lab3_41_W2 (you can create of your own) , Select proper security group

Instances are :

<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type
<input type="checkbox"/>	Worker1_41	i-0817bcdcf6a277c1	Running	t2.medium
<input type="checkbox"/>	Master_41	i-04d83d2955a09bc03	Running	t2.medium
<input type="checkbox"/>	Worker2_41	i-0fe66de19378bcaa1	Running	t2.medium

Step 2: After creating the instances click on Connect for all the instances one by one and navigate to the SSH Client section .

<input type="checkbox"/>	Name ↴	Instance ID	Instance state	Instance type
<input type="checkbox"/>	Worker1_41	i-0817bcdcf6a277c1	Running ⓘ ⓘ	t2.medium
<input type="checkbox"/>	Master_41	i-04d83d2955a09bc03	Running ⓘ ⓘ	t2.medium
<input type="checkbox"/>	Worker2_41	i-0fe66de19378bcaa1	Running ⓘ ⓘ	t2.medium

Master Node:

The screenshot shows the 'Connect to instance' page for an AWS EC2 instance named 'Master_41'. The instance ID is i-04d83d2955a09bc03. The 'SSH client' tab is selected. The page provides instructions for connecting via SSH, including steps to open an SSH client, locate the private key file (lab3_41_kube.pem), run chmod 400 on it, and connect using the Public DNS (ec2-54-158-48-115.compute-1.amazonaws.com). An example command is shown: ssh -i "lab3_41_kube.pem" ubuntu@ec2-54-158-48-115.compute-1.amazonaws.com. A note at the bottom states: '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.'

Step 3: Now open the folder in the terminal 3 times by right clicking on it for Master, Node1 & Node 2 where our .pem key is stored and paste the Example command (starting with ssh -i) from the ssh client section for instance in the terminal as shown above.

This will basically make our terminal to do remote login on our ec2 instance via SSH

■ Mini_Project	08-03-2024 11:15	File folder
■ Newfolder	29-09-2024 06:19	File folder
■ React_Scrimba	21-09-2024 21:16	File folder

MasterNode:

EC2 > Instances > i-04d83d2955a09bc03 > Connect to instance

Connect to instance info

Connect to your instance i-04d83d2955a09bc03 (Master_41) using any of these options

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

instance ID
 i-04d83d2955a09bc03 (Master_41)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is lab3_41_kube.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable:
 chmod 400 "lab3_41_kube.pem"
4. Connect to your instance using its Public DNS:
 ec2-54-158-48-115.compute-1.amazonaws.com

Example:
 ssh -i "lab3_41_kube.pem" ubuntu@ec2-54-158-48-115.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.

Successful Connection:

```

ubuntu@ip-172-31-84-76: ~ %
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/pro

System information as of Sun Sep 29 00:57:09 UTC 2024
System load: 0.0 Processes: 116
Usage of /: 22.9% of 6.73GB Users logged in: 0
Memory usage: 5% IPv4 address for enx0: 172.31.84.76
Swap usage: 0%

Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.

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

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

Last login: Sun Sep 29 00:55:44 2024 From 183.87.29.122
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-84-76: ~ $

```

Worker Node 1:

The screenshot shows the AWS EC2 Instances page with the instance `i-0817bcdcf6a277c1` selected. The **Connect to instance** section is open, displaying four connection methods: EC2 Instance Connect, Session Manager, SSH client (selected), and EC2 serial console. The **Instance ID** is listed as `i-0817bcdcf6a277c1 (Worker1_41)`. Below it, a numbered list of steps for using an SSH client is provided:

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is `lab3_41_W1.pem`.
3. Run this command, if necessary, to ensure your key is not publicly viewable:
`chmod 400 "lab3_41_W1.pem"`
4. Connect to your instance using its Public DNS:
`ec2-3-82-160-230.compute-1.amazonaws.com`

An **Example:** section shows the command to run: `ssh -i "lab3_41_W1.pem" ubuntu@ec2-3-82-160-230.compute-1.amazonaws.com`. A note in a callout box states: **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.

Successful Connection:

```
ubuntu@ip-172-31-93-130:~ % + =
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 00:58:10 UTC 2024

System load: 0.82      Processes:          116
Usage of /: 22.9% of 6.71GB  Users logged in:     0
Memory usage: 5%
Swap usage:  0%
IPv4 address for enx0: 172.31.93.130

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

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

Last login: Sun Sep 29 00:56:27 2024 from 103.87.29.122
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-93-130:~$ |
```

Worker Node 2:

EC2 > Instances > i-0fe66de19378bcaa1 > Connect to instance

Connect to instance Info

Connect to your instance i-0fe66de19378bcaa1 (Worker2_41) using any of these options.

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

Instance ID: [i-0fe66de19378bcaa1 \(Worker2_41\)](#)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is lab3_41_W2.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
[chmod 400 "lab3_41_W2.pem"](#)
4. Connect to your instance using its Public DNS:
[ec2-3-93-79-152.compute-1.amazonaws.com](#)

Example:
[ssh -i "lab3_41_W2.pem" ubuntu@ec2-3-93-79-152.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.

Successful Connection:

```
ubuntu@ip-172-31-93-235:~ % + . ~
System information as of Sun Sep 29 00:59:06 UTC 2024
System load: 0.0          Processes:      113
Usage of /:  22.0% of 6.71GB  Users logged in:    0
Memory usage: 6%
Swap usage:  0%
IPv4 address for enx0: 172.31.93.235

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-172-31-93-235:~ |
```

Step 4: Run on Master,Worker Node 1, and Worker Node 2 the below commands to install and setup Docker in Master,Worker Node 1, and Worker Node 2.

```
a) 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-172-31-84-76:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee
/etc/apt/trusted.gpg.d/docker.gpg > /dev/null
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable"
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
-----BEGIN PGP PUBLIC KEY BLOCK-----
mQINBFit2io8EADhWpZB/wvZ6hUTiX0wQHXMAlaFHCPh9hAtr4Fly2+0YdbtMuth
lqqmp028AqyP0PRFVMtSYMbjuQuu5byyKR01BbqYhuS3jtqQmljZ/bJvXqnmiVXh
38UuLa+z077PxxyQhu5BbqntTPQMfiyqEiU+BKbq2WmANUkQf+1AmZY/Iru0Xbnq
L4C1+g38vFmXQ99npCaxEjaNRVYf0S8QcixNzHUYnb6emjlANyEVlZzeqo7XWl7
Urw5inawTSzWnvjtEjj4nJL8NsLwscpLPQUhTQ+78bQXAmAmeHCUtQIVvvWXqw9N
cmhh4HgeQscQHYgOJjjDVfoY5MucvglbIgCqfzAHW9jxaRL4qbMZj+b1XoePEtht
ku4BiQNIx5P87fNWz2lgaRL5Z4P0XDD0ZTLiQ/E158j9kp4bnWRCJW8ly+a+f8ocodo
vZz+Doif+y4D5ZGrL4XecIQP/Lv5ufyf+k0tl/94VFYYJ0LeAv8M92KdgDkhTcTD
G7cBTikvEKENUq4Bb3aQ64N0ZQW7fVjfoKwEzD0qPE72Pa45jrZzvUFxFspdiNk2t2
XYukHjlxxEgBdC/J3cMMNRE1F4NC43ApfV1Y7/hTeOnmDuDYwv9/obABt016Yljj
q5rdkymPF4JF8mXUW5eCN1vAfHxeg9ZWehb7tQmGxXnw9M+z6hWw6ahmnARAQAB
tCtEb2NrZXIxUmVsZWFrZSAoQ8UgZGVikSA8ZG9ja2VvQGRvY2tlci5jb20+iQ13
BBMBCgAhBQJYrefAAhsvBQsJCAcDBRUKCQgLBRYCwAAhH8AheAAhJEI2BgDw0
```

```
Get:28 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [532 B]
Get:29 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:30 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [112 B]
Get:31 https://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [18.6 kB]
Get:32 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [10.8 kB]
Get:33 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [17.6 kB]
Get:34 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1104 B]
Get:35 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:36 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:37 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:38 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:39 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [388 kB]
Get:40 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [82.9 kB]
Get:41 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4560 B]
Get:42 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [272 kB]
Get:43 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [115 kB]
Get:44 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:45 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.3 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [153 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 [2888 B]
Get:51 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:52 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [344 B]
Fetched 29.1 MB in 4s (6810 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
ubuntu@ip-172-31-84-76:~$ |
```

b) sudo apt-get update

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

```
ubuntu@ip-172-31-88-76:~$ sudo apt-get update
[sudo] password for ubuntu:
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for 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-rootless-extras docker-compose-plugin libltdl7
    libslirp0 pigz slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 143 not upgraded.
Need to get 123 MB of archives.
After this operation, 442 MB of additional disk space will be used.
Get:1 http://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 [40.3 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble/main amd64 libslirp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble/universe amd64 slirp4netns amd64 1.2.1-1build2 [34.9 kB]
Get:5 https://download.docker.com/linux/ubuntu/noble/stable amd64 containerd.io amd64 1.7.22-1 [29.5 MB]
```

```
ubuntu@ip-172-31-88-76:~$ 
Unpacking slirp4netns (1.2.1-1build2) ...
Setting up docker-buildx-plugin (0.17.1-1-ubuntu.24.04-noble) ...
Setting up containerd.io (1.7.22-1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /usr/lib/systemd/system/containerd.service.
Setting up docker-compose-plugin (2.29.7-1-ubuntu.24.04-noble) ...
Setting up libltdl7:amd64 (2.4.7-7build1) ...
Setting up docker-ce-cli (5:27.3.1-1-ubuntu.24.04-noble) ...
Setting up libslirp0:amd64 (4.7.0-1ubuntu3) ...
Setting up pigz (2.8-1) ...
Setting up docker-ce-rootless-extras (5:27.3.1-1-ubuntu.24.04-noble) ...
Setting up slirp4netns (1.2.1-1build2) ...
Setting up docker-ce (5:27.3.1-1-ubuntu.24.04-noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu0.2) ...
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-172-31-88-76:~$ |
```

c) sudo mkdir -p /etc/docker

```
cat <<EOF | sudo tee /etc/docker/daemon.json
```

```
{
```

```
"exec-opts": ["native.cgroupdriver=systemd"]
```

```
}
```

EOF

```
ubuntu@ip-172-31-84-76:~$ sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-84-76:~$ |
```

- d) **sudo systemctl enable docker**
sudo systemctl daemon-reload
sudo systemctl restart docker

```
ubuntu@ip-172-31-84-76:~$ 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
ubuntu@ip-172-31-84-76:~$ |
```

Step 5: Run the below command to **install Kubernetes** on all the three terminals one by one

- a) **sudo rm -f /etc/apt/sources.list.d/kubernetes.list**

```
ubuntu@ip-172-31-90-144:~$ sudo rm -f /etc/apt/sources.list.d/kubernetes.list
```

- b) **sudo apt-get update**

```
ubuntu@ip-172-31-90-144:~$ sudo apt-get update
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
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored
  in apt-key(8) for details.
```

c) sudo apt-get install -y apt-transport-https ca-certificates curl gpg

```
ubuntu@ip-172-31-90-144:~$ sudo apt-get install -y apt-transport-https ca-certificates curl gpg
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203).
ca-certificates set to manually installed.
gpg is already the newest version (2.4.4-2ubuntu17).
gpg set to manually installed.
The following NEW packages will be installed:
```

```
Fetched 904 kB in 0s (29.7 MB/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 68007 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.7.14build2_all.deb ...
Unpacking apt-transport-https (2.7.14build2) ...
Preparing to unpack .../curl_8.5.0-2ubuntu10.4_amd64.deb ...
Unpacking curl (8.5.0-2ubuntu10.4) over (8.5.0-2ubuntu10.1) ...
Preparing to unpack .../libcurl4t64_8.5.0-2ubuntu10.4_amd64.deb ...
Unpacking libcurl4t64:amd64 (8.5.0-2ubuntu10.4) over (8.5.0-2ubuntu10.1) ...
Preparing to unpack .../libcurl3t64-gnutls_8.5.0-2ubuntu10.4_amd64.deb ...
Unpacking libcurl3t64-gnutls:amd64 (8.5.0-2ubuntu10.4) over (8.5.0-2ubuntu10.1) ...
Setting up apt-transport-https (2.7.14build2) ...
Setting up libcurl4t64:amd64 (8.5.0-2ubuntu10.4) ...
Setting up libcurl3t64-gnutls:amd64 (8.5.0-2ubuntu10.4) ...
Setting up curl (8.5.0-2ubuntu10.4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...
systemctl restart packagekit.service

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.
```

d) sudo mkdir -p -m 755 /etc/apt/keyrings

```
ubuntu@ip-172-31-90-144:~$ sudo mkdir -p -m 755 /etc/apt/keyrings
```

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

```
ubuntu@ip-172-31-90-144:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key
ng.gpg
File '/etc/apt/keyrings/kubernetes-apt-keyring.gpg' exists. Overwrite? (y/N) y
```

g) 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-172-31-90-144:~$ 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
```

f) sudo apt-get update

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

```
ubuntu@ip-172-31-90-144:~$ 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 https://download.docker.com/linux/ubuntu noble InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv/:/kubernetes:/core:/stable:/v1.31/deb/ InRelease [1186 B]
Hit:6 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv/:/kubernetes:/core:/stable:/v1.31/deb/ Packages [4865 B]
Fetched 0B in 1s (9987 B/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg)
  This keyring is deprecated. Consider using gpg --keyring instead.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  conntrack cri-tools kubeadm kubectl kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 140 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 conntrack amd64 1:1.8.8-1ubuntu1 [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 kB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv/:/kubernetes:/core:/stable:/v1.31/deb kubeadm 1.31.1-1.1 [11.4 kB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv/:/kubernetes:/core:/stable:/v1.31/deb kubectl 1.31.1-1.1 [11.2 kB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv/:/kubernetes:/core:/stable:/v1.31/deb kubernetes-cni 1.5.1-1.1 [33.9 kB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv/:/kubernetes:/core:/stable:/v1.31/deb kubelet 1.31.1-1.1 [15.2 kB]
Fetched 87.4 MB in 1s (70.6 MB/s)
Selecting previously unselected package conntrack.
```

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.

g) sudo systemctl enable --now kubelet

```
ubuntu@ip-172-31-84-76:~$ sudo systemctl enable --now kubelet
```

h) sudo apt-get install -y containerd

```
ubuntu@ip-172-31-84-76:~$ sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz
  slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 148 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~Ubuntu3.1 [8599 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12~Ubuntu4.1 [38.6 kB]
Fetched 47.2 MB in 1s (83.3 MB/s)
(Reading database ... 68048 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 ... 68048 files and directories currently installed.)
Preparing to unpack .../runc_1.1.12~Ubuntu3.1_amd64.deb ...
Unpacking runc (1.1.12~Ubuntu3.1) ...
Selecting previously unselected package containerd.
(Reading database ... 68048 files and directories currently installed.)
Preparing to unpack .../containerd_1.7.12~Ubuntu4.1_amd64.deb ...
Unpacking containerd (1.7.12~Ubuntu4.1) ...
Setting up runc (1.1.12~Ubuntu3.1) ...
Setting up containerd (1.7.12~Ubuntu4.1) ...
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.
```

i) sudo mkdir -p /etc/containerd

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

```
ubuntu@ip-172-31-90-144:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
path = ""

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

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

[metrics]
address = ""
grpc_histogram = false

[plugins]
```

j) sudo systemctl restart containerd

sudo systemctl enable containerd

sudo systemctl status containerd

```
ubuntu@ip-172-31-84-76:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd
● containerd.service - containerd container runtime
   Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
   Active: active (running) since Sun 2024-09-29 01:34:39 UTC; 229ms ago
     Docs: https://containerd.io
 Main PID: 5324 (containerd)
    Tasks: 8
   Memory: 13.1M (peak: 14.3M)
      CPU: 59ms
     CGroup: /system.slice/containerd.service
             └─5324 /usr/bin/containerd

Sep 29 01:34:39 ip-172-31-84-76 containerd[5324]: time="2024-09-29T01:34:39.934118179Z" level=info msg="Start subscribing to events"
Sep 29 01:34:39 ip-172-31-84-76 containerd[5324]: time="2024-09-29T01:34:39.934140135Z" level=info msg="serving... address=:2375"
Sep 29 01:34:39 ip-172-31-84-76 containerd[5324]: time="2024-09-29T01:34:39.934165273Z" level=info msg="Start recovering"
Sep 29 01:34:39 ip-172-31-84-76 containerd[5324]: time="2024-09-29T01:34:39.934187499Z" level=info msg="serving... address=:2375"
Sep 29 01:34:39 ip-172-31-84-76 containerd[5324]: time="2024-09-29T01:34:39.934214170Z" level=info msg="Start event monitoring"
Sep 29 01:34:39 ip-172-31-84-76 containerd[5324]: time="2024-09-29T01:34:39.934226888Z" level=info msg="Start snapshots"
Sep 29 01:34:39 ip-172-31-84-76 containerd[5324]: time="2024-09-29T01:34:39.934234364Z" level=info msg="Start cni interface"
Sep 29 01:34:39 ip-172-31-84-76 containerd[5324]: time="2024-09-29T01:34:39.934240935Z" level=info msg="Start streaming"
Sep 29 01:34:39 ip-172-31-84-76 systemd[1]: Started containerd.service - containerd container runtime.
Sep 29 01:34:39 ip-172-31-84-76 containerd[5324]: time="2024-09-29T01:34:39.936194445Z" level=info msg="containerd successfully started"
ubuntu@ip-172-31-84-76:~$ sudo apt-get install -y socat
```

```
ubuntu@ip-172-31-84-76:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz
  slirp4netns
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 149 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.0-0~4build3 [374 kB]
Fetched 374 kB in 0s (15.2 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68112 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0-0~4build3_amd64.deb ...
Unpacking socat (1.8.0-0~4build3) ...
Setting up socat (1.8.0-0~4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

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

```
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-172-31-84-76:~$ |
```

Step 6: Set up the Kubernetes cluster by running the following command exclusively on the master node:

sudo kubeadm init --pod-network-cidr=10.244.0.0/16. This command initializes the Kubernetes control plane and specifies the pod network range to be used within the cluster i.e kubeadm will initialize the kubernetes cluster and cidr is Classless Inter Domain Range which decides the range of network range of the pods

```
ubuntu@ip-172-31-84-76:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W0929 01:52:20.346395   6244 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that used by kubeadm. It is recommended to use "registry.k8s.io/pause:3.10" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-84-76 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.84.76]
[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-172-31-84-76 localhost] and IPs [172.31.84.76 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-84-76 localhost] and IPs [172.31.84.76 127.0.0.1 ::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
```

```
ubuntu@ip-172-31-84-76:~* - ~
certificate credentials
[bootstrap-token] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Books trap 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 172.31.84.76:6443 --token 6zxtbp.go2qmmge82ih7w9t \
  --discovery-token-ca-cert-hash sha256:6b96e72828e4e3b98fcba53d2b2f41f3c3ee9a813155c2dc9d2dda313bd2bc88
ubuntu@ip-172-31-84-76:~$
```

Run this command on master:

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

These commands create a .kube directory in your home folder, copy the Kubernetes admin configuration file into it, and change the ownership of that configuration file to the current user to ensure proper access.

```
ubuntu@ip-172-31-84-76:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-84-76:~$ |
```

Step 7: Now Run the command **kubectl get nodes** to see the nodes before executing Join command on nodes. kubectl is a command-line tool used to interact with and manage Kubernetes clusters. It allows users to deploy applications, inspect and manage cluster resources, and view logs, among other tasks, by sending commands to the Kubernetes API server.

```
ubuntu@ip-172-31-84-76:~$ kubectl get nodes
NAME           STATUS      ROLES   AGE      VERSION
ip-172-31-84-76   NotReady   control-plane   6m42s   v1.31.1
ubuntu@ip-172-31-84-76:~$ |
```

Step 8: Now Run the following command on Node 1 and Node 2 to Join to master. For this from the kubeadm init command output for the master node copy and paste this on both the node

```
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 172.31.84.76:6443 --token 6zxtbp.go2qmmge82ih7w9t \
  --discovery-token-ca-cert-hash sha256:6b96e72828e4e3b98fc453d2b2f41f3c3ee9a013155c2dc9d2dda313bd2bc88
ubuntu@ip-172-31-84-76:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-84-76:~$ kubectl get nodes
NAME           STATUS      ROLES   AGE      VERSION
ip-172-31-84-76   NotReady   control-plane   6m42s   v1.31.1
ubuntu@ip-172-31-84-76:~$ |
```

Copy and paste: sudo kubeadm join 172.31.84.76:6443 --token
6zxtbp.go2qmwge82ih7w9t \
--discovery-token-ca-cert-hash
sha256:6b96e72028e4e3b98fcb453d2b2f41f3c3ee9a013155c2dc9d2dda313bd2bc88

Worker Node 1:

```
ubuntu@ip-172-31-93-130:~$ sudo kubeadm join 172.31.84.76:6443 --token 6zxtbp.go2qmwge82ih7w9t \
--discovery-token-ca-cert-hash sha256:6b96e72028e4e3b98fcb453d2b2f41f3c3ee9a013155c2dc9d2dda313bd2bc88
[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 1.0001865005s
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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

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

Worker Node 2:

```
ubuntu@ip-172-31-93-235:~$ sudo kubeadm join 172.31.84.76:6443 --token 6zxtbp.go2qmwge82ih7w9t \
--discovery-token-ca-cert-hash sha256:6b96e72028e4e3b98fcb453d2b2f41f3c3ee9a013155c2dc9d2dda313bd2bc88
[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 1.0000833351s
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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

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

Step 9: Run the command **kubectl get nodes** to see the nodes after executing Join command on nodes.

```
ubuntu@ip-172-31-84-76:~$ kubectl get nodes
NAME           STATUS    ROLES      AGE     VERSION
ip-172-31-84-76  NotReady  control-plane  30m    v1.31.1
ip-172-31-93-130  NotReady  <none>       4m23s   v1.31.1
ip-172-31-93-235  NotReady  <none>       37s    v1.31.1
ubuntu@ip-172-31-84-76:~$ |
```

Step 10: Since Status is NotReady we have to add a network plugin. And also we have to give the name to the nodes.

kubectl apply -f <https://docs.projectcalico.org/manifests/calico.yaml> . This command applies the Calico network plugin configuration, which enables networking capabilities in the Kubernetes cluster . It enhances the effective communication between pods in network

```
ubuntu@ip-172-31-84-76:~$ kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
poddisruptionbudget.policy/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
serviceaccount/calico-node created
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgppeers.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/blockaffinities.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/caliconodestatuses.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/clusterinformations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/felixconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworksets.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/hostendpoints.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamblocks.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamconfigs.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamhandles.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippools.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipreservations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/kubecontrollersconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networksets.crd.projectcalico.org created
clusterrole.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrole.rbac.authorization.k8s.io/calico-node created
clusterrolebinding.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
daemonset.apps/calico-node created
deployment.apps/calico-kube-controllers created
ubuntu@ip-172-31-84-76:~$ |
```

Now perform **sudo systemctl status kubelet**

```
ubuntu@ip-172-31-84-76:~$ sudo systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
   Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; preset: enabled)
   Drop-In: /usr/lib/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: active (running) since Sun 2024-09-29 01:52:38 UTC; 35min ago
       Docs: https://kubernetes.io/docs/
 Main PID: 6918 (kubelet)
    Tasks: 10 (limit: 4676)
   Memory: 33.6M (peak: 34.0M)
      CPU: 26.546s
     CGroup: /system.slice/kubelet.service
             └─6918 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/ssl/ca.crt --cert-dir=/etc/kubernetes/ssl --v=2 --log-dir=/var/log/kubelet --log-timestamp=true --allow-privileged=true --node-ip=172.31.84.76 --node-name=kubelet --pod-infra-container-image=k8s.gcr.io/pause:3.6 --pod-infra-service-account=kubelet --pod-infra-service-account-namespace=kubelet

Sep 29 02:27:45 ip-172-31-84-76 kubelet[6918]:          : unknown
Sep 29 02:27:45 ip-172-31-84-76 kubelet[6918]: > pod="kube-system/coredns-7c65d6fc9-7nrwx" podUID="1b227dfc-524d-4b11-9e6f-4a34832a3420" runtimeID="6918" log.go:32] "StopPod Sandbox from runtime"
Sep 29 02:27:45 ip-172-31-84-76 kubelet[6918]:           rpc error: code = Unknown desc = failed to stop container "ldaf0988-423d-4a8b-9a20-3a43a510458a"
Sep 29 02:27:45 ip-172-31-84-76 kubelet[6918]:          : unknown
Sep 29 02:27:45 ip-172-31-84-76 kubelet[6918]: > podSandboxID="8ce61ac68e9c48cd2ac2ac734ef78143faf196ccf04c67427bc23a9" runtimeID="6918" log.go:32] "StopPod Sandbox from runtime"
Sep 29 02:27:45 ip-172-31-84-76 kubelet[6918]: E0929 02:27:45.874271 6918 kuberuntime_manager.go:1479] "Failed to stop pod"
Sep 29 02:27:45 ip-172-31-84-76 kubelet[6918]: E0929 02:27:45.968413 6918 kubelet.go:1865] "KillPod Failed" err="*[fa]"
Sep 29 02:27:49 ip-172-31-84-76 kubelet[6918]: E0929 02:27:49.223548 6918 scope.go:117] "RemoveContainer" containerID="ldaf0988-423d-4a8b-9a20-3a43a510458a"
Sep 29 02:27:49 ip-172-31-84-76 kubelet[6918]: E0929 02:27:49.223664 6918 pod_workers.go:1381] "Error syncing pod, skipping"
ubuntu@ip-172-31-84-76:~$ |
```

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

```
ubuntu@ip-172-31-84-76:~$ kubectl get nodes -o wide
NAME           STATUS   ROLES      AGE    VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE        KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-84-76   Ready    control-plane   39m   v1.31.1   172.31.84.76   <none>       Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-93-130  Ready    <none>      12m   v1.31.1   172.31.93.130  <none>       Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-93-235  Ready    <none>      8m45s  v1.31.1   172.31.93.235  <none>       Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ubuntu@ip-172-31-84-76:~$
```

Now to Rename run this command

Rename to Worker_Node1_41: `kubectl label node ip-172-31-93-138 kubernetes.io/role=Worker_Node1_41`

Rename to Worker_Node2_41 : `kubectl label node ip-172-31-93-235 kubernetes.io/role=Worker_Node2_41`

Make Sure to give Correct corresponding IP while renaming which can be seen from previous output also.

```
ubuntu@ip-172-31-84-76:~$ kubectl label node ip-172-31-93-138 kubernetes.io/role=Worker_Node1_41
node/ip-172-31-93-138 labeled
ubuntu@ip-172-31-84-76:~$ kubectl label node ip-172-31-93-235 kubernetes.io/role=Worker_Node2_41
node/ip-172-31-93-235 labeled
ubuntu@ip-172-31-84-76:~$ kubectl get nodes -o wide
NAME           STATUS   ROLES      AGE    VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE        KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-84-76   Ready    control-plane   43m   v1.31.1   172.31.84.76   <none>       Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-93-138  Ready    Worker_Node1_41  16m   v1.31.1   172.31.93.138  <none>       Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-93-235  Ready    Worker_Node2_41  12m   v1.31.1   172.31.93.235  <none>       Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ubuntu@ip-172-31-84-76:~$
```

Step 11: Run command **kubectl get nodes -o wide** . And Hence we can see we have Successfully connected both the Worker Nodes to the Master Node.

```
ubuntu@ip-172-31-84-76:~$ kubectl get nodes -o wide
NAME           STATUS   ROLES      AGE    VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE        KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-84-76   Ready    control-plane   43m   v1.31.1   172.31.84.76   <none>       Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-93-138  Ready    Worker_Node1_41  16m   v1.31.1   172.31.93.138  <none>       Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-93-235  Ready    Worker_Node2_41  12m   v1.31.1   172.31.93.235  <none>       Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ubuntu@ip-172-31-84-76:~$
```

CONCLUSION:

In this experiment, we established a Kubernetes cluster on AWS EC2 instances, comprising one master node and two worker nodes. After setting up Docker and the required Kubernetes tools (kubelet, kubeadm, kubectl, and containerd) on all nodes, we initialized the master and incorporated the worker nodes into the cluster. Initially, the nodes displayed a NotReady status, which we corrected by installing the Calico network plugin. We also tagged the nodes with their respective roles (control-plane and worker). In the end, all nodes changed to the Ready state, showcasing that we successfully set up and managed the Kubernetes cluster.

Experiment No: 4

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

STEPS:

Step 1: Access your AWS Academy or personal account and initiate the launch of a new EC2 instance. Select **Ubuntu as the Amazon Machine Image (AMI)** and choose **t2.medium** for the instance type. Generate an RSA key with a .pem file extension, and relocate the downloaded key to the designated folder. **Note:** Since at least 2 CPUs are needed, ensure you select t2.medium

	Name	Instance ID	Instance state	Instance type
	Exp4_41	i-0ecfe3f98ef7f2b3e	Running	t2.medium

Step 2: Once the instance is created, click on the option to connect to the instance and go to the SSH client section. Now open the folder in the terminal where our .pem key is stored and paste the Example command (starting with ssh -i) in the terminal.

Instances (1/1) Info					Last updated 1 minute ago	Connect
<input type="text"/> Find Instance by attribute or tag (case-sensitive)					All states	Filter
<input checked="" type="checkbox"/> Name		Instance ID	Instance state	Instance type	Actions	
<input checked="" type="checkbox"/>	Exp4_41	i-0ecfe3f98ef7f2b3e	Running	t2.medium	Edit	

```

Ubuntu@ip-172-31-90-144 ~ % + v

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\praja\OneDrive\Desktop\NewFolder> ssh -i "Experiment No 4.pem" ubuntu@ec2-44-204-48-238.compute-1.amazonaws.com
The authenticity of host 'ec2-44-204-48-238.compute-1.amazonaws.com (44.204.48.238)' can't be established.
ED25519 key fingerprint is SHA256:Yx7Wvnt0ENhq3Cx013bDrmyuNH3aA9wKPh8fOKwG5tY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-44-204-48-238.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

```

Step 3: Execute the following commands to **install and set up 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-172-31-90-144:~$ 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"

```

```

Get:40 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [82.9 kB]
Get:41 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4560 B]
Get:42 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [272 kB]
Get:43 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [115 kB]
Get:44 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:45 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.3 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [353 kB]
Get:47 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [68.1 kB]
Get:48 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [428 B]
Get:49 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [10.9 kB]
Get:50 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [2808 B]
Get:51 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:52 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [344 B]
Fetched 29.1 MB in 4s (7650 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring
/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
ubuntu@ip-172-31-90-144:~$ 

```

b) sudo apt-get update

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

```
ubuntu@ip-172-31-90-144:~$ 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: Key is stored
/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

```
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-172-31-90-144:~$ |
```

c) sudo mkdir -p /etc/docker

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

```
ubuntu@ip-172-31-90-144:~$ sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-90-144:~$ |
```

- d) **sudo systemctl enable docker**
sudo systemctl daemon-reload
sudo systemctl restart docker

```
ubuntu@ip-172-31-90-144:~$ 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
ubuntu@ip-172-31-90-144:~$ |
```

Step 4: Execute the following command to **install Kubernetes**.

- a) **sudo rm -f /etc/apt/sources.list.d/kubernetes.list**

```
ubuntu@ip-172-31-90-144:~$ sudo rm -f /etc/apt/sources.list.d/kubernetes.list
```

- b) **sudo apt-get update**

```
ubuntu@ip-172-31-90-144:~$ sudo apt-get update
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
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored
  in apt-key(8) for details.
```

- c) **sudo apt-get install -y apt-transport-https ca-certificates curl gpg**

```
ubuntu@ip-172-31-90-144:~$ sudo apt-get install -y apt-transport-https ca-certificates curl gpg
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240820).
ca-certificates set to manually installed.
gpg is already the newest version (2.4.4-2ubuntu17).
gpg set to manually installed.
The following NEW packages will be installed:
```

```
Fetched 904 kB in 0s (29.7 MB/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 68007 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.7.14build2_all.deb ...
Unpacking apt-transport-https (2.7.14build2) ...
Preparing to unpack .../curl_8.5.0-2ubuntu10.4_amd64.deb ...
Unpacking curl (8.5.0-2ubuntu10.4) over (8.5.0-2ubuntu10.1) ...
Preparing to unpack .../libcurl4t64_8.5.0-2ubuntu18.4_amd64.deb ...
Unpacking libcurl4t64:amd64 (8.5.0-2ubuntu10.4) over (8.5.0-2ubuntu10.1) ...
Preparing to unpack .../libcurl3t64-gnutls_8.5.0-2ubuntu18.4_amd64.deb ...
Unpacking libcurl3t64-gnutls:amd64 (8.5.0-2ubuntu18.4) over (8.5.0-2ubuntu18.1) ...
Setting up apt-transport-https (2.7.14build2) ...
Setting up libcurl4t64:amd64 (8.5.0-2ubuntu10.4) ...
Setting up libcurl3t64-gnutls:amd64 (8.5.0-2ubuntu18.4) ...
Setting up curl (8.5.0-2ubuntu10.4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...
  systemctl restart packagekit.service

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.
```

- d) sudo mkdir -p -m 755 /etc/apt/keyrings

```
ubuntu@ip-172-31-90-144:~$ sudo mkdir -p -m 755 /etc/apt/keyrings
```

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

```
ubuntu@ip-172-31-90-144:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key
ng.gpg
File '/etc/apt/keyrings/kubernetes-apt-keyring.gpg' exists. Overwrite? (y/N) y
```

- f) 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-172-31-90-144:~$ echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/ 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 / '
```

g) sudo apt-get update

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

```
ubuntu@ip-172-31-90-144:~$ 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 https://download.docker.com/linux/ubuntu noble InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv/:kubernetes:/core:/stable:/v1.31/deb InRelease [1186 B]
Hit:6 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv/:kubernetes:/core:/stable:/v1.31/deb Packages [4865 B]
Fetched 4051 B in 1s (9987 B/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg).
action in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  conntrack cri-tools kubeadm kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 100 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 conntrack amd64 1:1.4.0-1ubuntu1 [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 kB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv/:kubernetes:/core:/stable:/v1.31/deb kubeadm 1.31.1-1.1 [11.4 kB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv/:kubernetes:/core:/stable:/v1.31/deb kubectl 1.31.1-1.1 [11.2 kB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv/:kubernetes:/core:/stable:/v1.31/deb kubernetes-cni 1.5.1-1.1 [33.9 kB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv/:kubernetes:/core:/stable:/v1.31/deb kubelet 1.31.1-1.1 [15.2 kB]
Fetched 87.4 MB in 1s (79.6 MB/s)
Selecting previously unselected package conntrack.
```

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.

h) sudo systemctl enable --now kubelet

```
ubuntu@ip-172-31-90-144:~$ sudo systemctl enable --now kubelet
ubuntu@ip-172-31-90-144:~$ |
```

i) **sudo kubeadm init --pod-network-cidr=10.244.0.0/16**

```
ubuntu@ip-172-31-90-144:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
W0928 04:27:54.973362    7321 checks.go:1080] [preflight] WARNING: Couldn't create new CRI runtime service: validate service connection: validate CRI v1 runtime API: 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: rpc error: code = Unimplemented desc = unknown service runtime.v1.RuntimeService
with '--ignore-preflight-errors=...'.
To see the stack trace of this error execute with --v=5 or higher
```

Here we got an error so to resolve this we use NEXT FOLLOWING COMMAND

j) **sudo apt-get install -y containerd**

```
ubuntu@ip-172-31-90-144:~$ 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
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 140 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 containerd-all 0.10.6-0ubuntu1.20.04.1 [47.2 MB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd 0.10.6-0ubuntu1.20.04.1 [14.0 kB]
Fetched 47.2 MB in 1s (72.8 MB/s)
(Reading database ... 68068 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 ... 68048 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.
Preparing to unpack .../containerd_1.7.12-0ubuntu4.1_amd64.deb ...
Unpacking containerd (1.7.12-0ubuntu4.1) ...
Setting up runc (1.1.12-0ubuntu3.1) ...
Setting up containerd (1.7.12-0ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.
```

k) sudo mkdir -p /etc/containerd

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

```
ubuntu@ip-172-31-90-144:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
path = ""

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

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

[metrics]
address = ""
grpc_histogram = false

[plugins]
```

l) sudo systemctl restart containerd

sudo systemctl enable containerd

```
ubuntu@ip-172-31-90-144:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
ubuntu@ip-172-31-90-144:~$ |
```

m) sudo systemctl status containerd

```
ubuntu@ip-172-31-98-144:~$ sudo systemctl status containerd
● containerd.service - containerd container runtime
  Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
  Active: active (running) since Sat 2024-09-28 04:36:39 UTC; 2min 15s ago
    Docs: https://containerd.io
   Main PID: 8881 (containerd)
     Tasks: 8
    Memory: 13.5M (peak: 14.1M)
      CPU: 321ms
     CGroup: /system.slice/containerd.service
             └─8881 /usr/bin/containerd

Sep 28 04:36:39 ip-172-31-98-144 containerd[8881]: time="2024-09-28T04:36:39.268667478Z" level=info msg=
Sep 28 04:36:39 ip-172-31-98-144 containerd[8881]: time="2024-09-28T04:36:39.260698020Z" level=info msg=
Sep 28 04:36:39 ip-172-31-98-144 containerd[8881]: time="2024-09-28T04:36:39.260771896Z" level=info msg=
Sep 28 04:36:39 ip-172-31-98-144 containerd[8881]: time="2024-09-28T04:36:39.260798416Z" level=info msg=
Sep 28 04:36:39 ip-172-31-98-144 containerd[8881]: time="2024-09-28T04:36:39.260839996Z" level=info msg=
Sep 28 04:36:39 ip-172-31-98-144 containerd[8881]: time="2024-09-28T04:36:39.2608855673Z" level=info msg=
Sep 28 04:36:39 ip-172-31-98-144 containerd[8881]: time="2024-09-28T04:36:39.260863382Z" level=info msg=
Sep 28 04:36:39 ip-172-31-98-144 containerd[8881]: time="2024-09-28T04:36:39.260869817Z" level=info msg=
Sep 28 04:36:39 ip-172-31-98-144 containerd[8881]: time="2024-09-28T04:36:39.261264494Z" level=info msg=
Sep 28 04:36:39 ip-172-31-98-144 systemd[1]: Started containerd.service - containerd container runtime.
Lines 1-21/21 (END)
```

n) sudo apt-get install -y socat

```
ubuntu@ip-172-31-98-144:~$ 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-...
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 140 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 soca
Fetched 374 kB in 0s (16.3 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68112 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0-0-4build3_amd64.deb ...
Unpacking socat (1.8.0-0-4build3) ...
Setting up socat (1.8.0-0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-98-144:~$ |
```

Step 5: Initialize the Kubecluster.

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

```
ubuntu@ip-172-31-90-144:~$ 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
[preflight] You can also perform this action beforehand using 'kubeadm config images
W0928 04:46:10.278292     8652 checks.go:846] detected that the sandbox image "regist
used by kubeadm. It is recommended to use "registry.k8s.io/pause:3.10" as the CRI san
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-90-144 kubernetes
.local] and IPs [10.96.0.1 172.31.90.144]
[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-172-31-90-144 localhost]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-90-144 localhost]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file
[kubeconfig] Writing "scheduler.conf" kubeconfig file
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"
[control-plane] Using manifest folder "/etc/kubernetes/manifests"
[control-plane] Creating static Pod manifest for "kube-apiserver"
[control-plane] Creating static Pod manifest for "kube-controller-manager"
[control-plane] Creating static Pod manifest for "kube-scheduler"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubele
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Starting the kubelet
```

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 172.31.90.144:6443 --token 75sp2s.4moocvobfpkism9q \
    --discovery-token-ca-cert-hash sha256:42855a37f6c6446da3abb79740e05317a4aadc79d228cf7df4339bd90fd6f19d
ubuntu@ip-172-31-90-144:~$ |
```

Step 6: Copy the mkdir and chown commands from the top and execute them.

```
mkdir -p $HOME/.kube
```

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

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

```
ubuntu@ip-172-31-90-144:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-90-144:~$ |
```

Step 7: Add a common networking plugin called **flannel** as mentioned in the code.

Flannel is a simple and widely used networking plugin for Kubernetes that facilitates pod-to-pod communication across a cluster.

```
kubectl apply -f
```

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

```
ubuntu@ip-172-31-90-144:~$ 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
ubuntu@ip-172-31-90-144:~$ |
```

Step 8: With the cluster now running, we can deploy our **Nginx server** onto it. Run the following command to apply the deployment configuration and set up the Nginx deployment.

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

```
ubuntu@ip-172-31-90-144:~$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
ubuntu@ip-172-31-90-144:~$ |
```

kubectl get pods

NAME	READY	STATUS	RESTARTS	AGE
nginx-deployment-d556bf558-6r7bm	0/1	Pending	0	76s
nginx-deployment-d556bf558-bv4vz	0/1	Pending	0	76s

```
ubuntu@ip-172-31-90-144:~$ kubectl get pods
NAME                   READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-6r7bm   0/1     Pending   0          76s
nginx-deployment-d556bf558-bv4vz   0/1     Pending   0          76s
ubuntu@ip-172-31-90-144:~$ |
```

```
ubuntu@ip-172-31-90-144:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath=".items[0].metadata.name")
ubuntu@ip-172-31-90-144:~$ |
```

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

```
ubuntu@ip-172-31-90-144:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath=".items[0].metadata.name")
ubuntu@ip-172-31-90-144:~$ |
ubuntu@ip-172-31-90-144:~$ kubectl port-forward $POD_NAME 8080:80
error: unable to forward port because pod is not running. Current status=Pending
ubuntu@ip-172-31-90-144:~$ |
```

Note: If you encounter an error where the pod status shows as "Pending," follow these steps. Run the commands below to resolve the issue, and then re-run the previous two commands.

**kubectl taint nodes --all
node-role.kubernetes.io/control-plane-node/ip-172-31-20-171 untainted**

```
ubuntu@ip-172-31-90-144:~$ kubectl taint nodes --all node-role.kubernetes.io/control-plane-
node/ip-172-31-90-144 untainted
ubuntu@ip-172-31-90-144:~$ |
```

kubectl get nodes

```
ubuntu@ip-172-31-90-144:~$ kubectl get nodes
NAME           STATUS      ROLES      AGE      VERSION
ip-172-31-90-144   Ready   control-plane   40m    v1.31.1
ubuntu@ip-172-31-90-144:~$ |
```

kubectl get pods

```
ubuntu@ip-172-31-90-144:~$ kubectl get pods
NAME                  READY   STATUS      RESTARTS   AGE
nginx-deployment-d556bf558-6r7bm   1/1     Running   0          27m
nginx-deployment-d556bf558-bv4vz   1/1     Running   0          27m
ubuntu@ip-172-31-90-144:~$ |
```

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

```
ubuntu@ip-172-31-90-144:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath=".items[0].metadata.name")  
ubuntu@ip-172-31-90-144:~$ |  
  
ubuntu@ip-172-31-90-144:~$ 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  
  
|
```

Step 9 : Check if your deployment is working. Open a new terminal and connect to your EC2 instance using SSH. Then, run the curl command to see if the Nginx server is up and running.

```
Ubuntu@ip-172-31-90-144 ~ + -  
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\praj\OneDrive\Desktop\NewFolder> ssh -i "Experiment No 4.pem" ubuntu@ec2-44-284-48-238.compute-1.amazonaws.com  
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 Sat Sep 28 05:42:45 UTC 2024  
  
System load: 0.05 Processes: 153  
Usage of /: 55.6% of 6.71GB Users logged in: 1  
Memory usage: 19% IPv4 address for enX0: 172.31.90.144  
Swap usage: 0%  
  
* Ubuntu Pro delivers the most comprehensive open source security and  
compliance features.  
  
https://ubuntu.com/aws/pro  
  
Expanded Security Maintenance for Applications is not enabled.  
  
142 updates can be applied immediately.  
38 of these updates are standard security updates.  
To see these additional updates run: apt list --upgradable
```

curl --head <http://127.0.0.1:8080>

```
ubuntu@ip-172-31-90-144:~$ curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Sat, 28 Sep 2024 05:44:34 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

ubuntu@ip-172-31-90-144:~$ |
```

After handling connecton

```
ubuntu@ip-172-31-90-144:~$ 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
|
```

If you get a **200 OK** response and see the Nginx server name, your deployment was successful.

CONCLUSION:

In this experiment, we successfully installed Kubernetes on an EC2 instance and deployed an Nginx server using various Kubectl commands. During the process, we faced two main problems. The first issue was that the Kubernetes pod was stuck in a pending state. We fixed this by removing the control-plane taint with the command **kubectl taint nodes --all**, which allowed the pod to schedule correctly. The second problem was a missing containerd runtime, which we resolved by installing and starting the containerd service. To make sure our Kubernetes setup had enough resources, we used a t2.medium EC2 instance with 2 virtual CPUs. Overall, these steps led to a successful deployment of the Nginx server, showing how effective Kubernetes can be in a cloud environment.

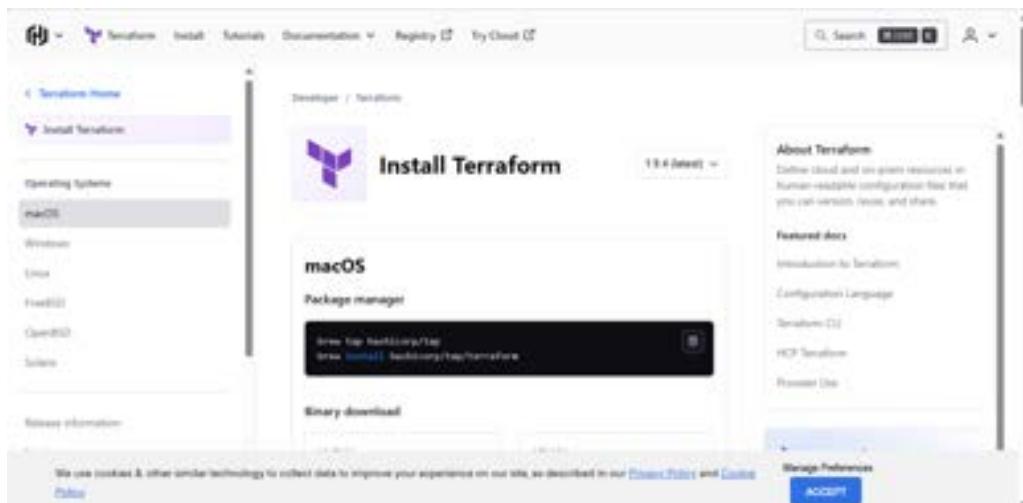
Experiment No: 5

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

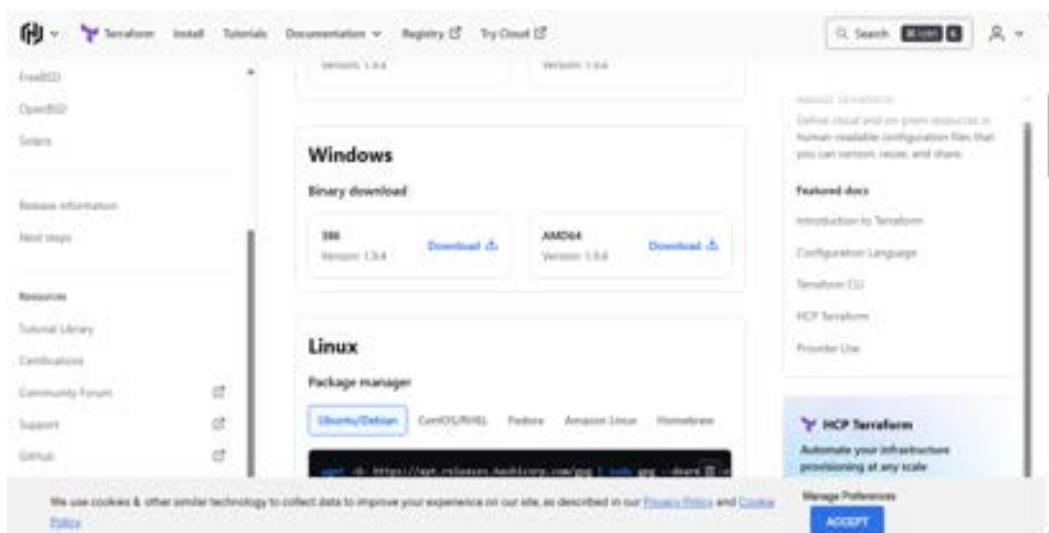
Installation and Configuration Steps of Terraform in Windows:

Step 1: Download terraform

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

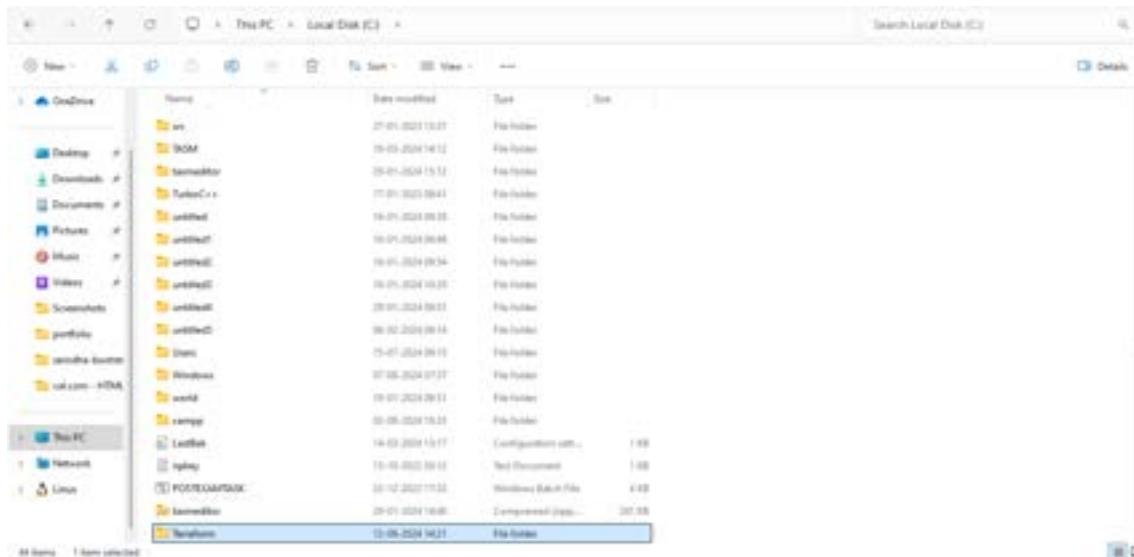


- 2) Select the Operating System Windows followed by either 32 bit or 64 bit based on your OS type.

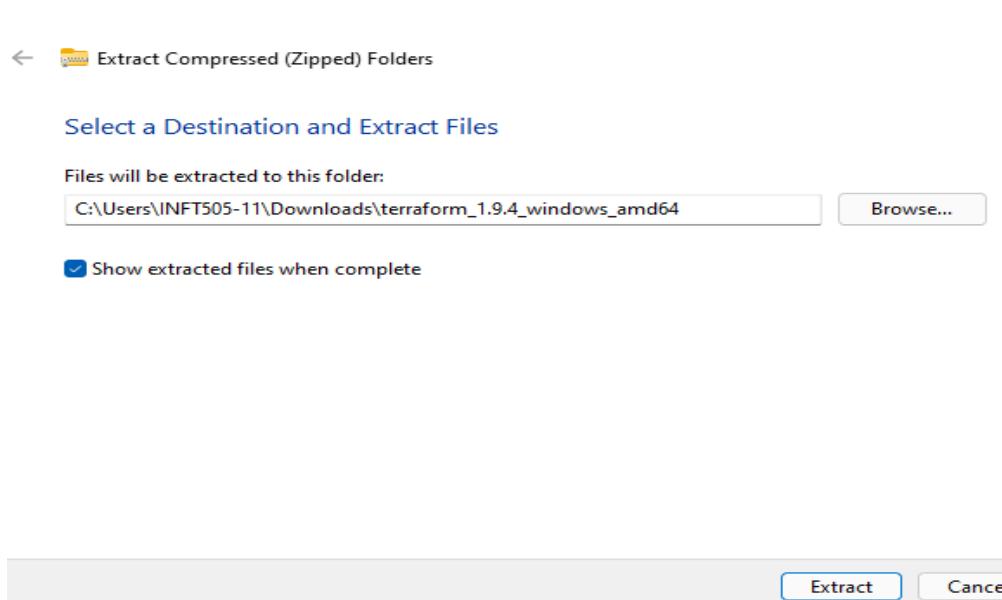


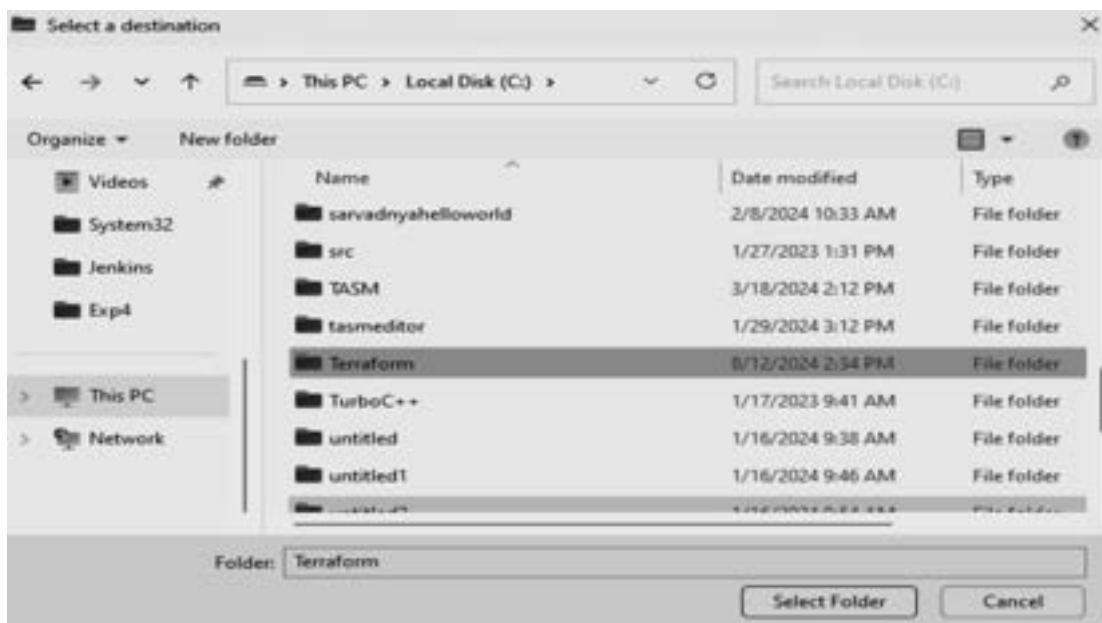
Step 2: Extract the downloaded setup file Terraform.exe in C:\Terraform Directory

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

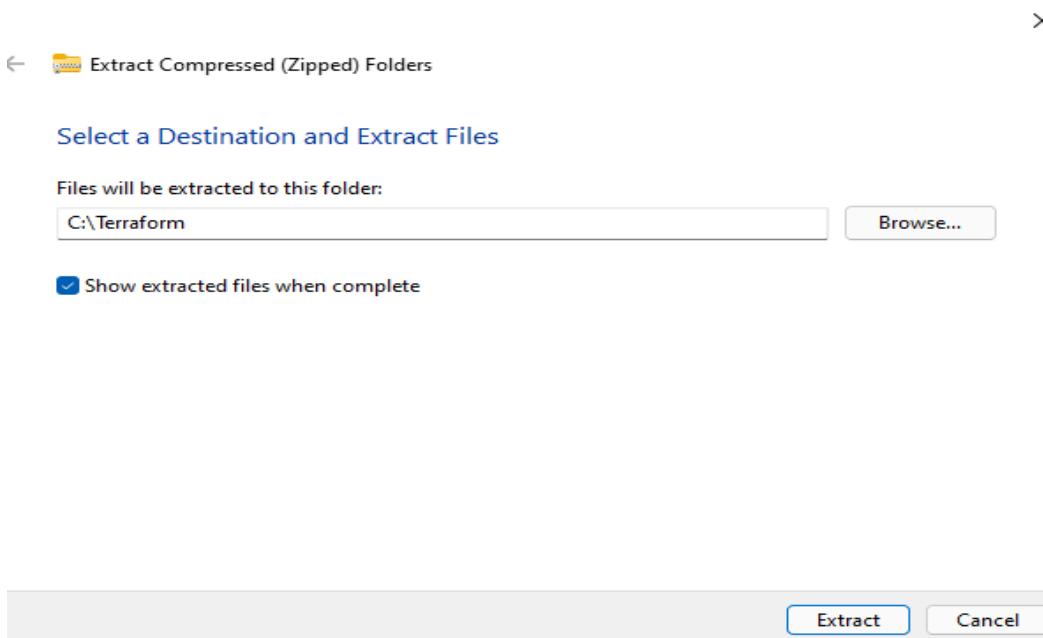


2) Got to location where main file is downloaded , right click on the file name and click on extract all while clicking on extract all it displays the location of the file to be saved , after extraction initially the destination of file to be extracted is not the folder which we have created earlier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

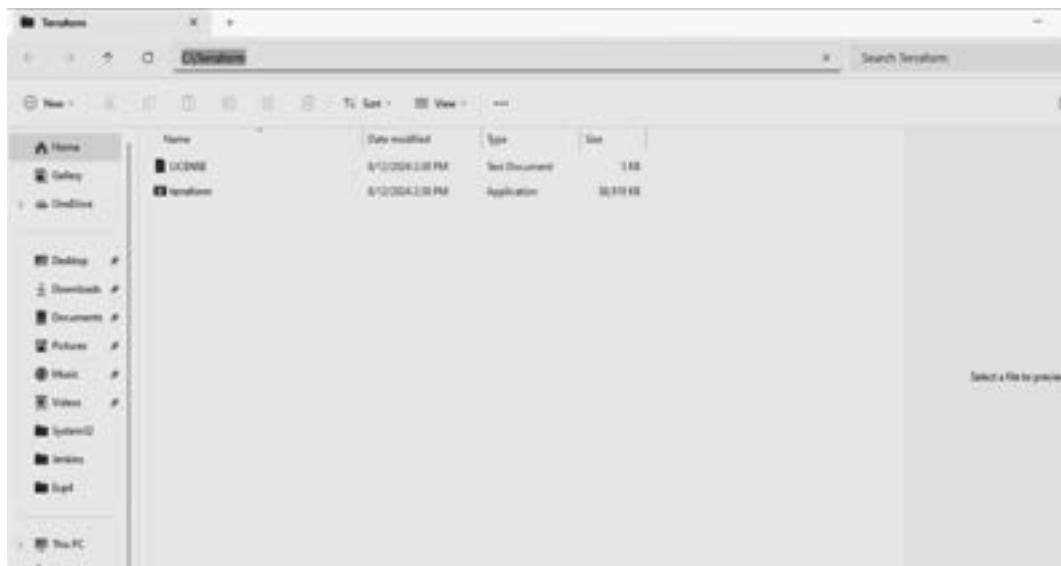




3) Click on extract Button.

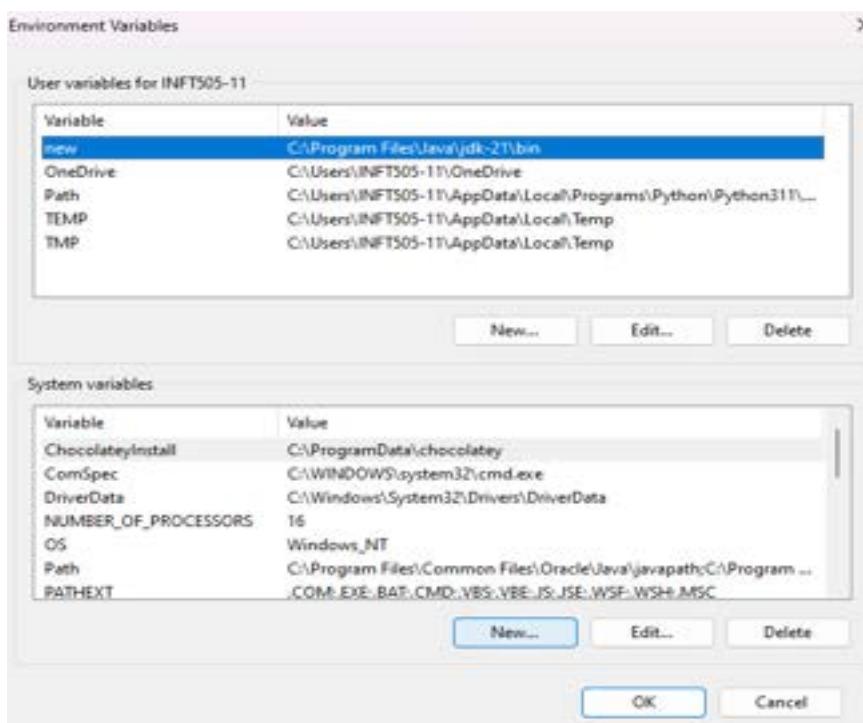


4) After extracting it will redirect to file manager copy the path which will be used for the further steps

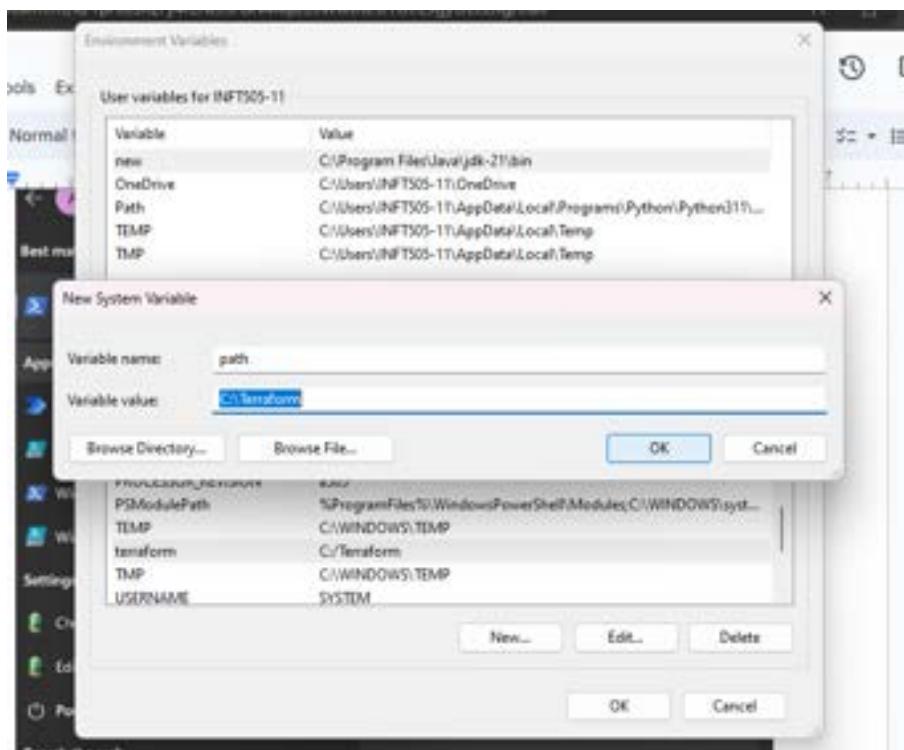


Step 3: Set the System path for Terraform in Environment Variables

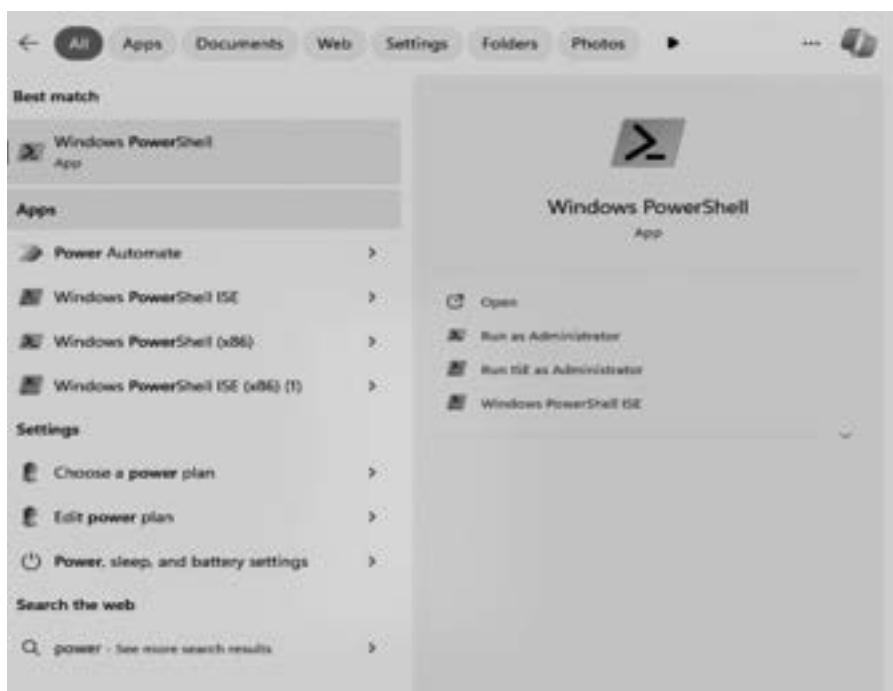
1) Search Environment Variable from search area in the Taskbar then Click on New under System Variables



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



Step 4: Open PowerShell with Admin Access



Step 5 : Open Terraform in PowerShell and check its functionality

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

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

CONCLUSION:

So This experiment confirms that the terraform can be installed correctly on a local machine by following the standard installation steps, By downloading the appropriate package and verifying The installation ensures that the Terraform is set up and operating correctly.This experiment underscores the importance of each installation step to avoid the potential issues in using Terraform.

Experiment No: 6

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

A. Creating docker image using terraform

Step 1: In this experiment ,we need to install docker on our local Machine. Go to <https://www.docker.com/> and download the file according to the OS you have.

Open the file and start the installation. After Installation, open your terminal and run ‘docker’ command. If this is your output, then docker is installed successfully .

```
PS C:\Users\praja> 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
```

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

Step 2: Create a Folder named ‘Docker’ in the ‘TerraformScripts’ folder. Then create a new docker.tf file using Atom editor if you are using linux or else use VS code in windows and write the following contents into it to create a Ubuntu Linux container.

Script:

```
terraform
{ required_providers
{docker = {
```

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

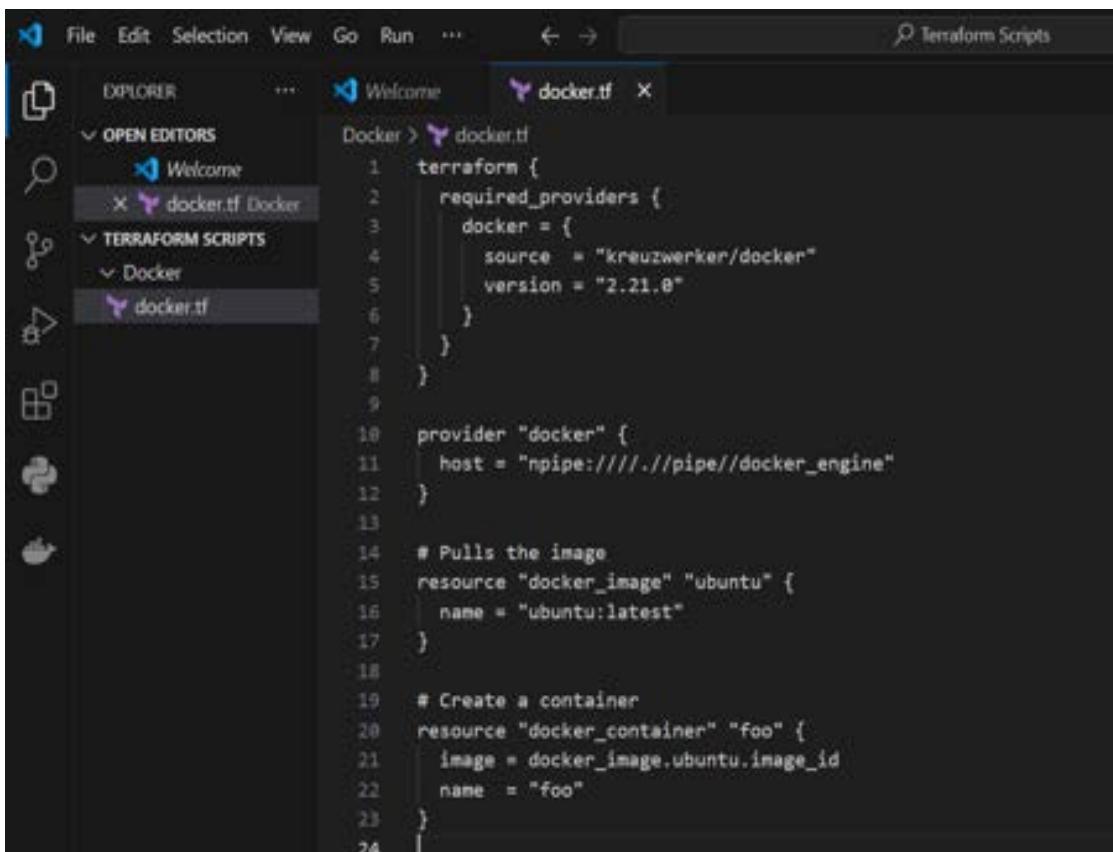
}

}

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

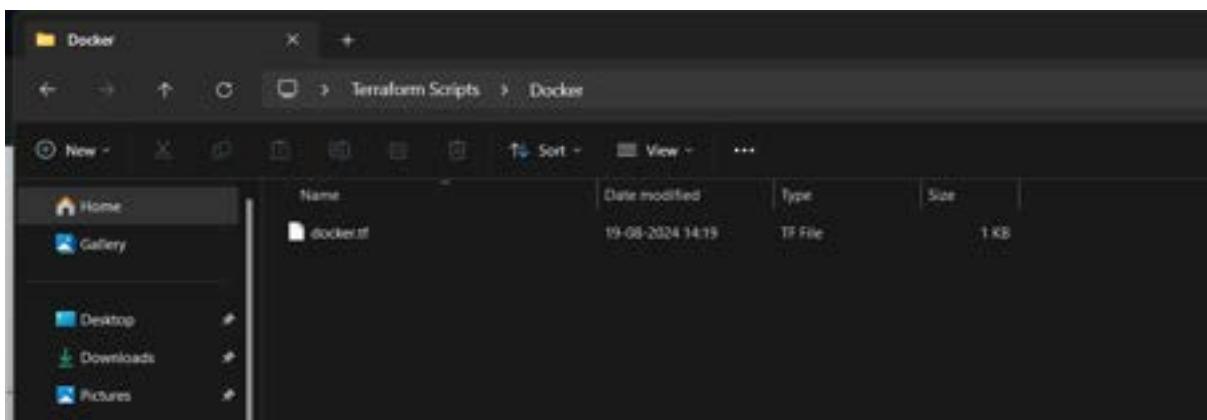
# Pulls the image
resource
  "docker_image"
  "ubuntu" {name =
"ubuntu:latest"
}

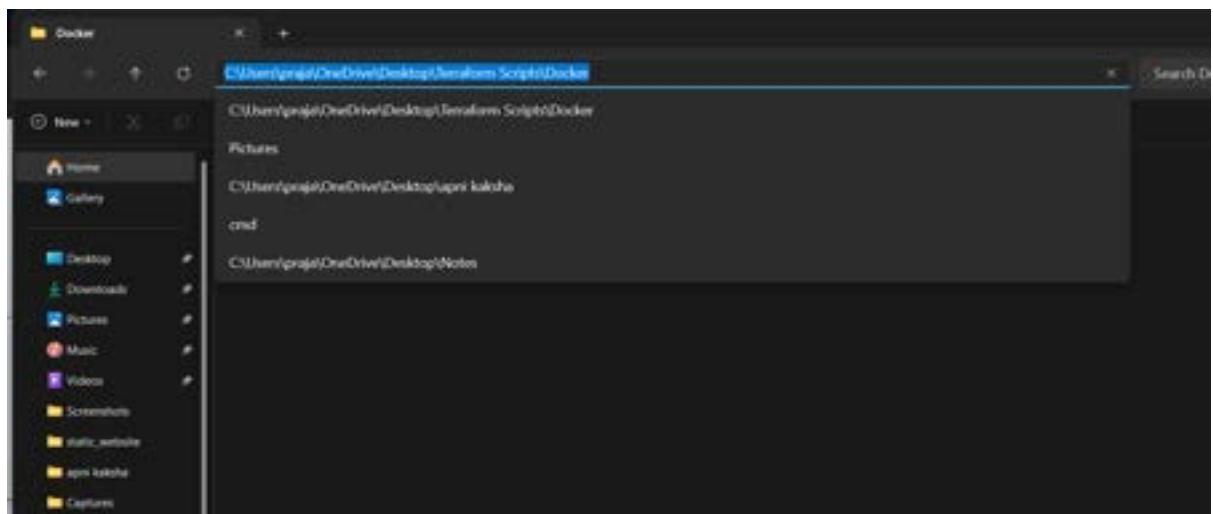
# Create a container
resource
  "docker_container"
  "foo" {image =
docker_image.ubuntu.imag
e_idname = "foo"
}
```



```
1 terraform {  
2   required_providers {  
3     docker = {  
4       source  = "Kreuzwerker/docker"  
5       version = "2.21.8"  
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 }  
24 }
```

Step 3: Execute Terraform Init command to initialize the resources .Now for this go to file manager ->Open Terraform script folder then open Docker folder ->Click on the path of these folder and type cmd this will open Command Prompt window to initialize it in our directory.





```
C:\Windows\system32\cmd.exe + - Microsoft Windows [Version 10.0.22631.4037] (c) Microsoft Corporation. All rights reserved. C:\Users\praja\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 B00880C4571C6164C) 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\praja\OneDrive\Desktop\Terraform Scripts\Docker>
```

Step 4: Execute Terraform plan to see the available resources.

This command helps to get an execution plan and lets us overview changes that are going to happen in your infrastructure.

```
C:\Users\praja\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)
  + ip_prefix_length = (known after apply)
  + ipc_mode         = (known after apply)
  + log_driver        = (known after apply)
  + logs              = false
  + must_run          = true

  + shm_size          = (known after apply)
  + start              = true
  + stdio_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.

C:\Users\praja\OneDrive\Desktop\TerraForm Scripts\Docker>
```

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

```
C:\Users\praja\OneDrive\Desktop\Terraform Scripts\Decker>terraform apply

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

Terraform will perform the following actions:

  # docker_container.foo will be created
+ resource "docker_container" "foo" {
    attach           = false
    bridge           = (known after apply)
    command          = (known after apply)
    container_logs   = (known after apply)
    entrypoint        = (known after apply)
    env              = (known after apply)
    exit_code         = (known after apply)
    gateway          = (known after apply)
    hostname         = (known after apply)
    id               = (known after apply)
    image             = (known after apply)
    init              = (known after apply)
    ip_address        = (known after apply)
    ip_prefix_length = (known after apply)
    ipc_mode          = (known after apply)
    log_driver        = (known after apply)
    logs              = false
    must_run          = true
    name              = "foo"
}
```

This will ask You to enter a value so Type “Yes”.

```
+ id      = (known after apply)
+ image_id = (known after apply)
+ latest   = (known after apply)
+ name     = "ubuntu:latest"
+ output   = (known after apply)
+ repo_digest = (known after apply)
}

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

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

Enter a value: yes

docker_image.ubuntu: Creating...
docker_image.ubuntu: Creation complete after 7s [id=sha256:edbfe74c41f8a2581ce542e137cf28ea84dd83e6df8c9d66519b6ad761c25
98ubuntu:latest]
docker_container.foo: Creating...

Error: container exited immediately

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

The script that we are using is going to throw an error. Error: container exited immediately .This is because the script used is way too small or took a lot less time to execute. To fix this, we add a

line to the code. ‘Command = [“sleep”, “infinity”]’. This line of code lets docker know to keep the program in sleep mode for an infinite amount of time so that the output can be observed rather than stopping after running immediately.

Do the following changes in the last line of the code as follows to solve the error

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

Now run the command again

```
C:\Users\praja\OneDrive\Desktop>Terraform Scripts\ Docker>terraform apply
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe70c41f8a3581ce542e137cf28ea84dd83e6df8c9d66519b6ad761c25
tu: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)
```

```
± C:\Users\praja\OneDrive\Desktop> + -
  + restart                         = "on"
  + 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=978fd330ac1cbf3873e164845ecf73e2645ec2820941fb36c629b5db2314494b]

apply complete! Resources: 1 added, 0 changed, 0 destroyed.
C:\Users\praja\OneDrive\Desktop\Terraform Scripts\ Docker>
```

Docker images, Before Executing Apply step:

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
docker/welcome-to-docker	latest	c1f619b6477e	9 months ago	18.6MB

Docker images, After Executing Apply step:

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
ubuntu	latest	edbfe74c41f8	2 weeks ago	78.1MB
docker/welcome-to-docker	latest	c1f619b6477e	9 months ago	18.6MB

Step 6: Now the image is created, if we have to destroy it. For this, we use the '*terraform destroy*' command. Again, this command will ask for a prompt to enter yes, as a confirmation to destroy the image we created.

Type Yes.

```
C:\Users\praja\OneDrive\Desktop\Terraform Scripts\Docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd93e6df8c9d66519b6ad761c2598a8buntu:latest]
docker_container.foo: Refreshing state... [id=978fd338ac1cbf3873e16f845ecd73e2645ec28209f1fb16c629b5db2314494b]

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

Terraform will perform the following actions:

  # docker_container.foo will be destroyed
  - resource "docker_container" "foo" {
      - attach                  = false => null
      - command                 =
          - "sleep"
          - "infinity"
      ] => null
      - cpu_shares              = 8 => null
      - dns                      = [] => null
      - dns_opts                 = [] => null
      - dns_search               = [] => null
      - entrypoint               = [] => null
      - env                      = [] => null
      - gateway                  = "172.17.0.1" => null
      - group_add                = [] => null
      - hostname                 = "978fd338ac1cbf3873e16f845ecd73e2645ec28209f1fb16c629b5db2314494b" => null
      - id                       = "978fd338ac1cbf3873e16f845ecd73e2645ec28209f1fb16c629b5db2314494b" => null
    }
```

```

- tty          = false -> null
# (8 unchanged attributes hidden)
}

# docker_image/ubuntu will be destroyed
resource "docker_image" "ubuntu" {
  - id      = "sha256:edbfe74c41f8a3501ce542e137cf28ea84dd03e6df8c9d66519b6ad761c2598a" -> null
  - image_id = "sha256:edbfe74c41f8a3501ce542e137cf28ea84dd03e6df8c9d66519b6ad761c2598a" -> null
  - latest    = "sha256:edbfe74c41f8a3501ce542e137cf28ea84dd03e6df8c9d66519b6ad761c2598a" -> null
  - name      = "ubuntu:latest" -> null
  - repo_digest = "ubuntu@sha256:8a37d68f4f73ebf3d@efafbcf66379bf3728962a8838616888f04e34a9ab63ee" -> 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=978fd330ac1cbf3873e16f845ecd73e2645ec28209f1fb16c629b5db2314494b]
docker_container.foo: Destruction complete after 0s
docker_image/ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea84dd03e6df8c9d66519b6ad761c2598a]ubuntu:latest
docker_image/ubuntu: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.

C:\Users\praja\OneDrive\Desktop\Terraform Scripts\Docker>

```

Docker images After Executing Destroy step

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
docker/welcome-to-docker	latest	clf619b6477e	9 months ago	18.6MB

Thus we have Successfully created the Docker image using terraform in this experiment and have also destroyed it

CONCLUSION:

The experiment successfully demonstrate the use of Terraform to automate the deployment and management of docker container .By creating the Terraform Script, initializing the environment and executing necessary commands we were able to deploy on Ubuntu Linux Container which was deleted at the last part of the experiment

Experiment No :7

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

PREREQUISITES:

1) Docker :

Run **docker -v command**. We use this command to check if docker is installed and running on your system.

```
C:\Users\praja>docker -v
Docker version 27.0.3, build 7d4bcd8
```

2) Install SonarQube Image:

The command **docker pull sonarqube** downloads a SonarQube image from Docker's online repository. This image lets you run SonarQube on your system using Docker without needing to install the full SonarQube software manually. It's like getting a ready-to-use version of SonarQube that can be started with Docker.

```
C:\Users\praja>docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
98a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a48d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest

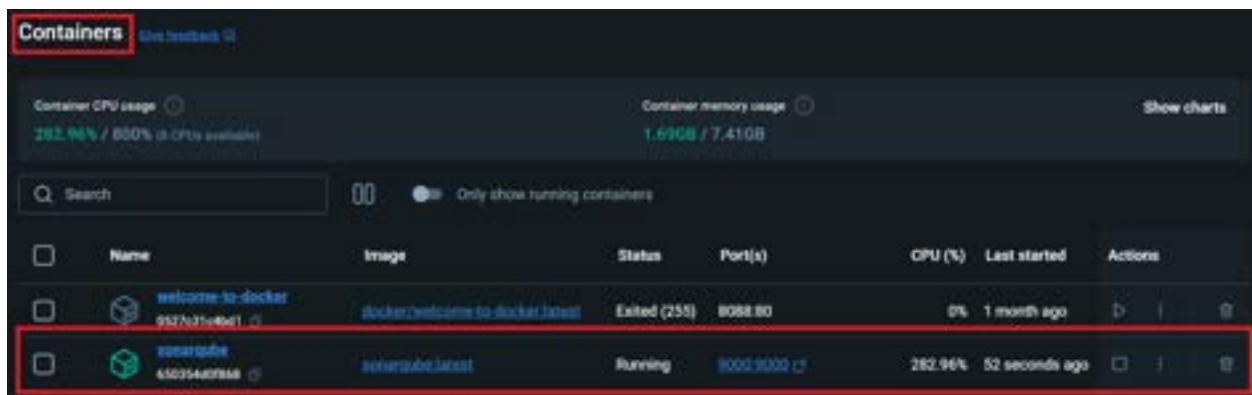
What's next:
  View a summary of image vulnerabilities and recommendations + docker scout quickview sonarqube
```

3) Make sure **Jenkins** is already installed on your system before starting the process. Jenkins will be used to automate tasks, like running SonarQube for code analysis. If Jenkins isn't installed yet, you can download and set it up from the official Jenkins website.

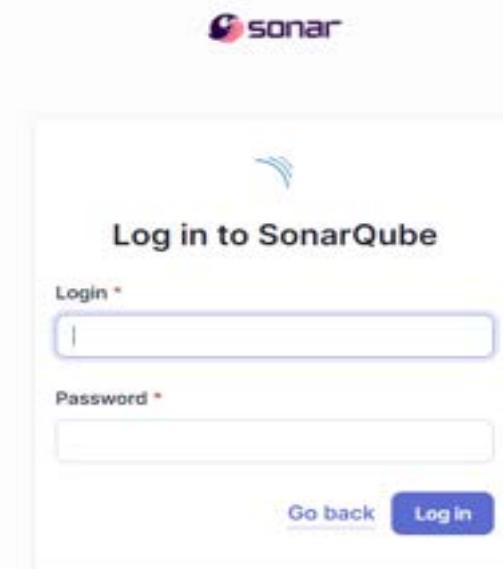
STEPS:

Step1: The command `docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest` starts SonarQube in the background on port 9000 using Docker, allowing you to access it at <http://localhost:9000>

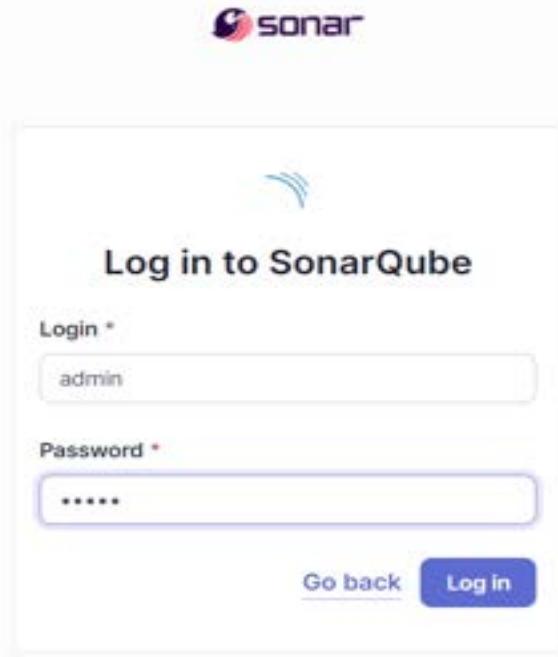
```
C:\Users\praja>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
656354d0f868ae4ad2d800426680076c604eb09f29b10d4a251aee70f51ce907
C:\Users\praja>
```



Step2: After starting the SonarQube image, open your browser and go to <http://localhost:9000> to access SonarQube.



Step 3: On the SonarQube login page, use the default credentials: **Username: admin** , **Password: admin**. After logging in, you'll be prompted to change the password. Set a new password and make sure to remember it.

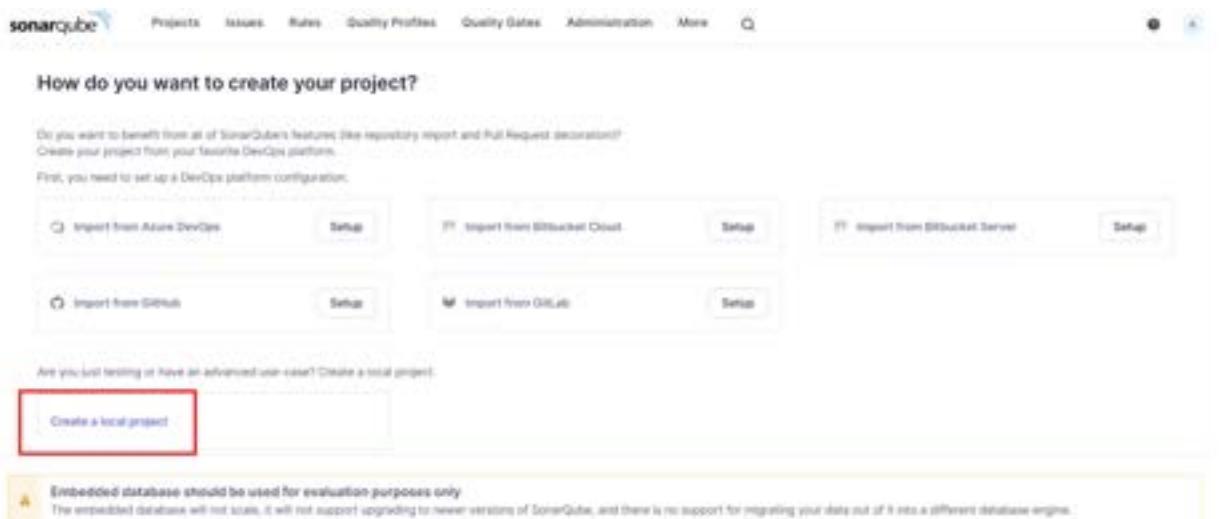


Click on Log in

A screenshot of the "Update your password" page. At the top, there's a warning message in a yellow box: "⚠ This account should not use the default password.". Below this, there are three input fields: "Old Password *", "New Password *", and "Confirm Password *". Each field contains several asterisks. At the bottom of the form is a blue "Update" button.

Click on Update .

Step 4: After changing the password, you will be directed to this screen. Click on **Create a Local Project**.



Step 5: Give your project , a display name and project key

The screenshot shows the 'Create a local project' step 1 of 2. The form has fields for 'Project display name' (41_Shivam) and 'Project key' (41_Shivam). There's also a field for 'Main branch name' (main) and a note about the default branch. At the bottom are 'Cancel' and 'Next' buttons. A yellow warning box at the bottom states: '⚠️ Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.' The footer includes links for SonarQube technology, community editions, documentation, and web API.

Step 6: Configure the project by providing the necessary settings like choosing the baseline for the new code for the project , then click **Create** to finalize the setup.

sonarqube

Projects Issues Rules Quality Profiles Quality Gates Administration More

3 of 2 Set up project for Clean as You Code

The new code definition sets 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

Any code that has changed in the last x days is considered new code. If no action is taken on a new issue after x days, this issue will become part of the overall code.
Recommended for projects following continuous delivery.

Back **Create project**

Scroll Down

sonarqube

Projects Issues Rules Quality Profiles Quality Gates Administration More

3 of 2 Set up project for Clean as You Code

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

Any code that has changed in the last x days is considered new code. If no action is taken on a new issue after x days, this issue will become part of the overall code.
Recommended for projects following continuous delivery.

Reference branch

Choose a branch as the baseline for the new code.
Recommended for projects using feature branches.

Back **Create project**

Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

Click on Create project

Step 7: Open Jenkins by going to **http://localhost:<port_number>** in your browser, replacing <port_number> with the specific port Jenkins is running on.

S	W	Name	Last Success	Last Failure	Last Duration
		AT_SonarQube	N/A	N/A	N/A
		Sonarqube Pipeline	1 min 21 days ago	N/A	1.1 sec
		Pipeline Pipeline	N/A	N/A	N/A
		Pipeline Pipeline	1 min 2 days ago	N/A	0.17 sec
		SonarQube AT	N/A	N/A	N/A
		sonarqube_AT_fall	N/A	1 day 12 hr ago	1.8 sec
		sonarqube_AT_fall	N/A	1 day 8 hr ago	2.0 sec
		sonarqube_AT_fall_AT	1 day 11 hr ago	1 day 10 hr ago	2.5 sec
		sonarqube_fall_AT	1 day 11 hr ago	1 day 10 hr ago	2.5 sec

Step 8: Now go to Manage Jenkins then go for Plugins followed by Available plugins search for **Sonarqube Scanner** where we are going to install it as a plugin.

New version of Jenkins (2.462.2) is available for download (changed). [Or Upgrade Automatically](#)

Manage Jenkins

System Configuration

- System**: Configure global settings and paths.
- Tools**: Configure tools, their locations, and automate toolpaths.
- Nodes**: Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
- Clouds**: Add, remove, and configure cloud instances to provision agents on demand.

Plugins: Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

Appearance: Configure the look and feel of Jenkins.

Plugins

Available plugins

SonarQube Scanner

Install Name: SonarQube Scanner 2.17.2

External Issue Integration - Build Reports

This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.

Released 7 mo 4 days ago

Click on Available Plugins.

Plugins

Available plugins

SonarQube Scanner

Install Name: SonarQube Scanner 2.17.2

External Issue Integration - Build Reports

This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.

Released 7 mo 4 days ago

Search in the Search bar the required Plugin Name and click on Install.

Plugins

Download progress

Preparation

- Checking internet connectivity
- Checking update center connectivity
- Success

SonarQube Scanner

Success

Loading plugin resources

Success

→ Go back to the top page
(you can start using the installed plugin right away)

→ Restart Jenkins when installation is complete and no jobs are running

Plugin Installed Successfully.

Step 9: In Jenkins, go to **Manage Jenkins → System**, then find **SonarQube servers**. Add a new server, and if required, include the authentication token for secure access

The screenshot shows the Jenkins dashboard. On the left sidebar, under 'Manage Jenkins', the 'System' option is highlighted with a red box. The main content area displays a table of pipelines: 'DevOps pipeline' (Last Success: 1 hour 18 mins, Last Failure: N/A, Last Duration: 1.3 sec), 'DevOps Pipeline' (N/A), and 'Pipeline DevOps' (Last Success: 29 days, Last Failure: N/A, Last Duration: 0.79 sec). Below the table are navigation links for 'Item', 'S', 'M', and 'L'.

Go to system

The screenshot shows the 'Manage Jenkins' page. The 'System' configuration option is highlighted with a red box. A message at the top indicates a new Jenkins version (2.612.2) is available for download. On the right, there are buttons for 'On Upgrade Automatically' and 'Downgrade to 2.492.3'. To the right of the 'System' box, there are sections for 'Tools' (Configure tools, their locations and automatic installers) and 'Plugins' (Add, remove, disable or enable plugins that can extend the functionality of Jenkins).

The screenshot shows the Jenkins 'System' configuration page. At the top, there is a navigation bar with links for 'Dashboard', 'Manage Jenkins', and 'System'. On the right side of the header, there is a search bar labeled 'Search (CTRL+E)' and a user profile icon with the name 'Shivam Prajapati' and a 'log out' button.

The main content area is titled 'System'. It contains two sections:

- Home directory:** A text input field containing the path 'c:\ProgramData\jenkins\jobs'. Below it is a help icon (info symbol).
- System Message:** A text input field with placeholder text: 'This message will be displayed at the top of the Jenkins main page. This can be useful for posting notifications to your users.' Below it is a help icon.

Below these sections are two buttons: 'Permit Review' and 'Review'. Underneath is a section for '# of executors' with a dropdown menu set to '1'. At the bottom of the page are 'Save' and 'Apply' buttons.

Scroll Down

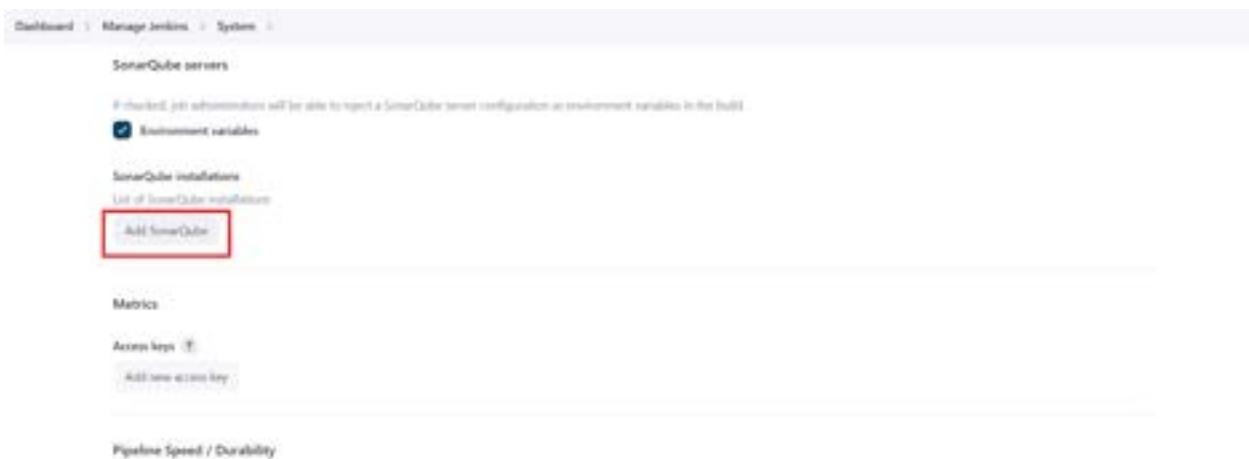
The screenshot shows the Jenkins 'System' configuration page, continuing from the previous one. The navigation bar and user profile are identical.

The main content area is titled 'System' and contains several sections:

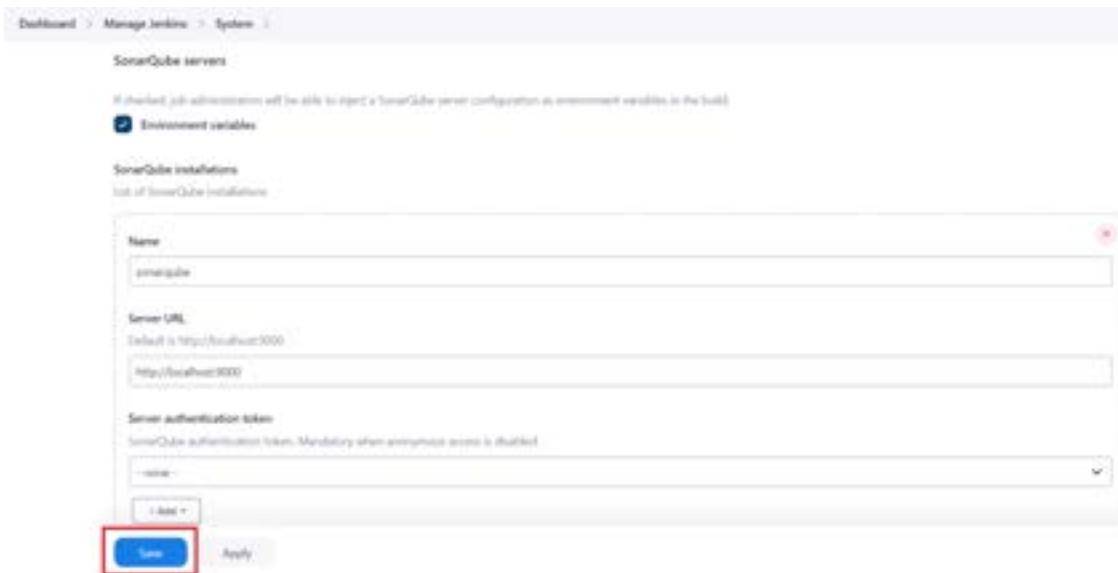
- SonarQube servers:** A section with a checkbox for 'If checked, jenkins administrators will be able to inject a SonarQube server configuration as environment variables in the build.' An unchecked checkbox for 'Environment variables' is also present.
- SonarQube installations:** A section with a link 'List of SonarQube installations' and a 'Add SonarQube' button.
- Metrics:** A section with a 'Access keys' link and a 'Add new access key' button.
- Pipeline Speed / Durability:** A section with a 'Default Speed / Durability Level' dropdown menu set to 'None: use pipeline default (maximum survivability/durability but slowest)'. Below it is a help icon.

At the bottom of the page are 'Save' and 'Apply' buttons.

Select Environment Variable and Click on Add Sonar Qube button in order to Add SonarQube Server to Jenkin

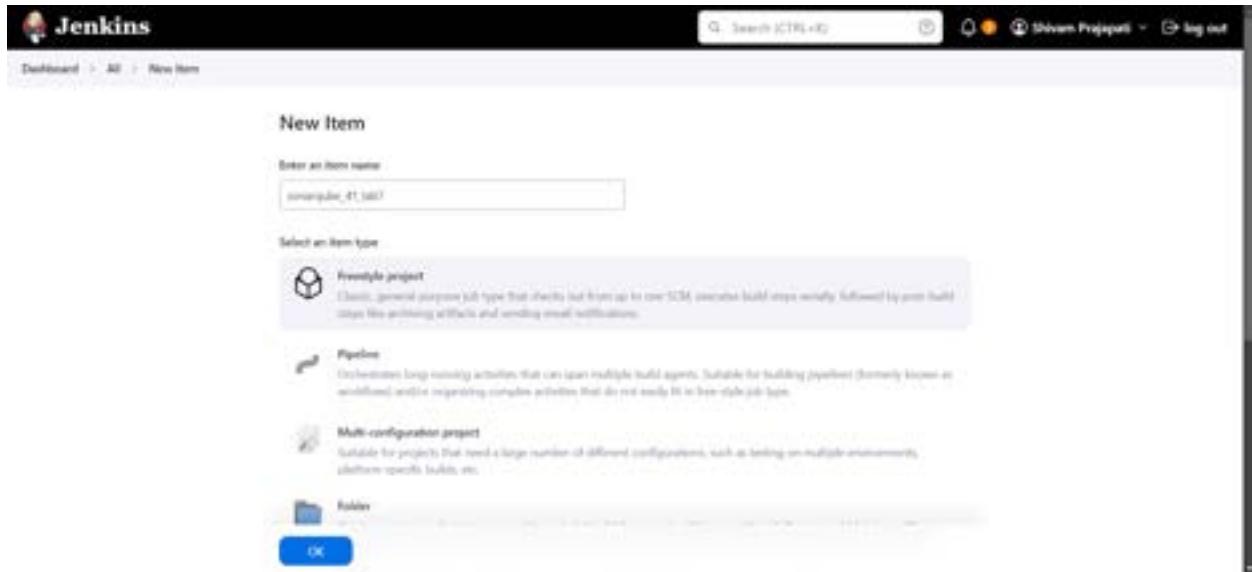


Do the required entries as shown below



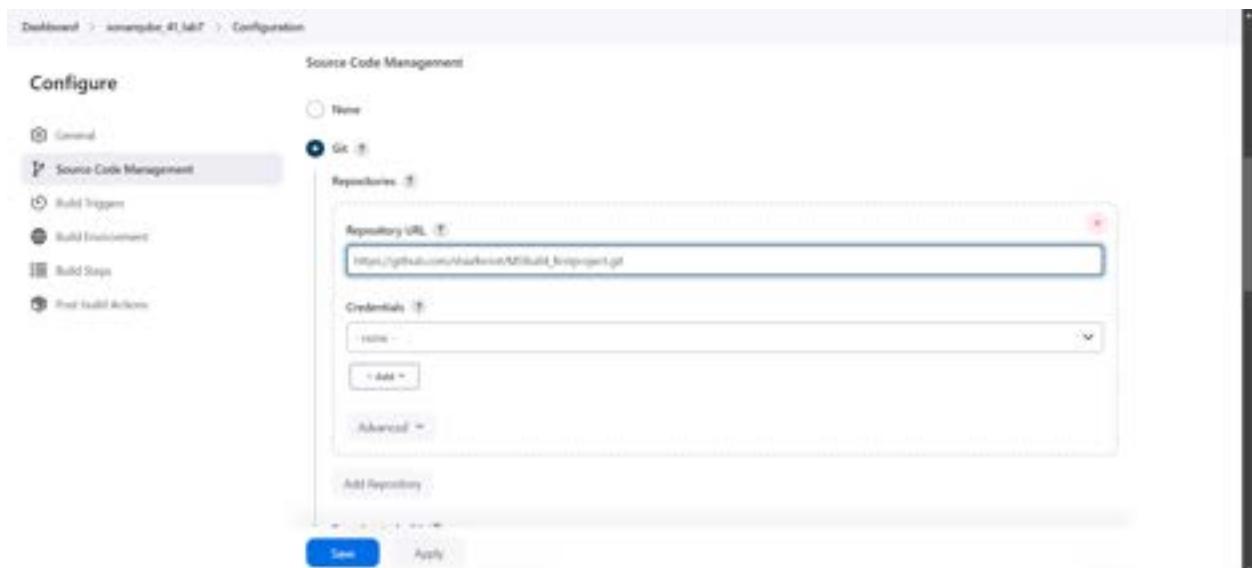
Click on save

Step 10: After configuration, create a New Item → choose a freestyle project.

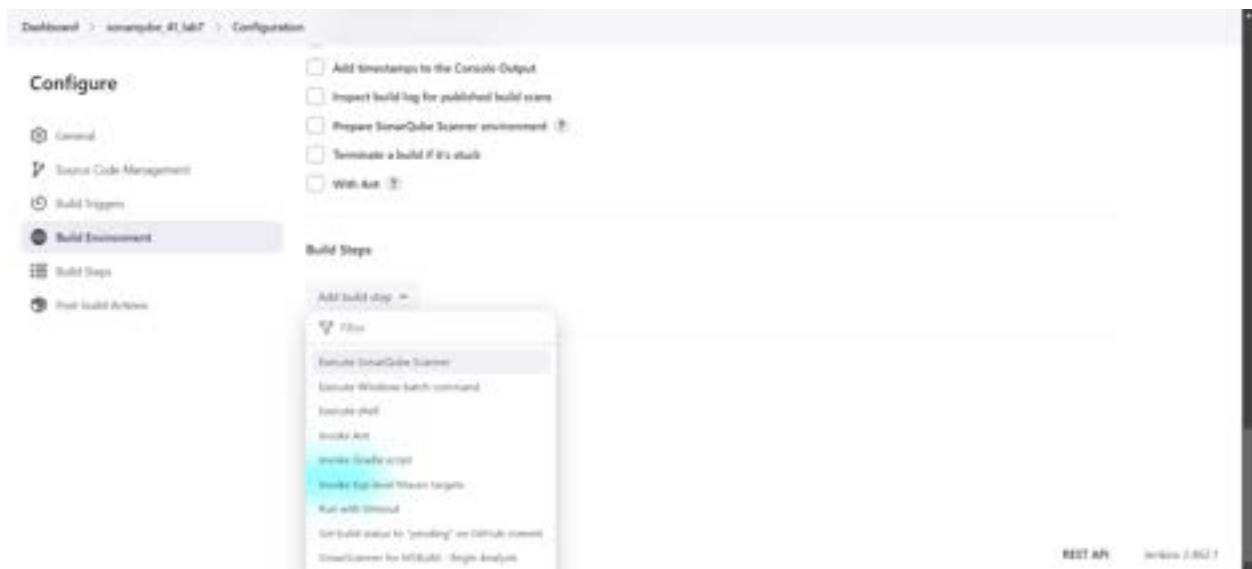


Step 11: Use this github repository in Source Code Management.

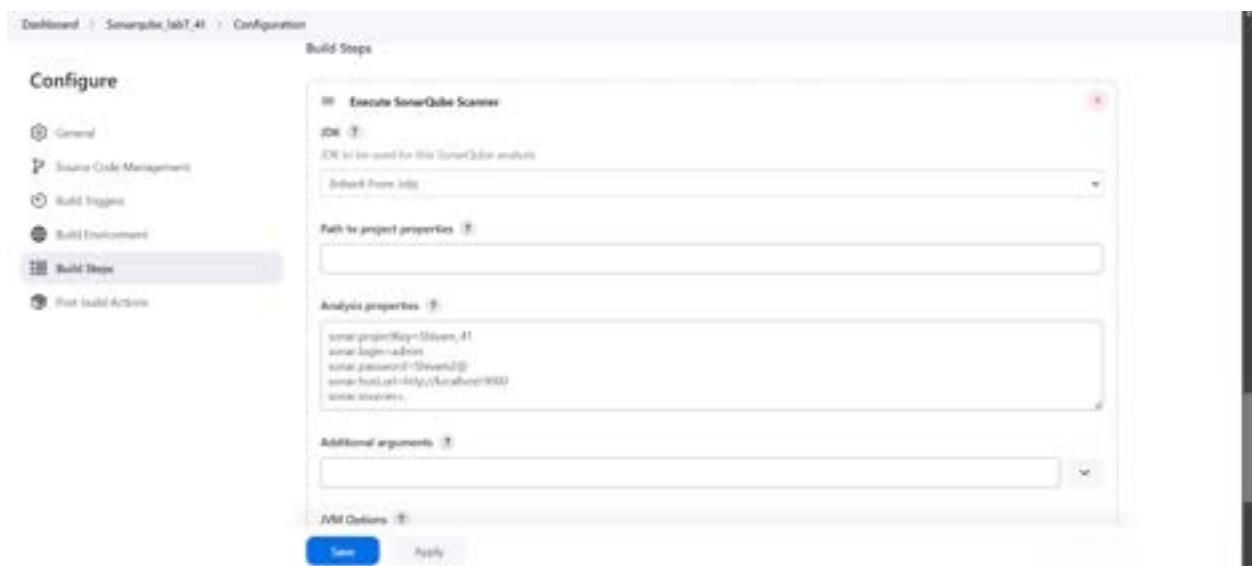
https://github.com/shazforiot/MSBuild_firstproject. It is a sample hello-world project with no vulnerabilities.



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



Click on execute sonar scanner





Step 13: Now, you need to grant the local user (here admin user) permissions to Execute the Analysis stage on SonarQube. For this, go to http://localhost:<port_number>/admin/permissions and check the ‘Execute Analysis’ checkbox under Administrator.

User Group	Administrator	Execute Analysis	Create	Projects
sonar-administrators	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Quality Gates <input checked="" type="checkbox"/> Quality Profiles	<input type="checkbox"/>	<input checked="" type="checkbox"/>
sonar-users	<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Administrator admin	<input checked="" type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Anyone <small>DEPRECATED</small>	<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input type="checkbox"/>	<input type="checkbox"/>

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

The screenshot shows the Jenkins interface for the 'Sonarqube_Lab7_41' job. The left sidebar contains links for Status, Changes, Workspace, Build Now, Configure, Delete Project, SonarQube, and Sensors. The main content area has a green checkmark icon and the text 'Sonarqube_Lab7_41'. Below it is the SonarQube logo and the word 'Permalinks'. A bulleted list shows the last seven builds: Last build (#40), Last stable build (#40), Last successful build (#40), Last failed build (#40), Last unsuccessful build (#40), and Last completed build (#40). At the bottom, there's a 'Build History' section with a dropdown set to 'Trend' and a search bar containing 'true'. It lists two builds: #40 (green circle) from Jul 23, 2013, 5:11 PM and #39 (red circle) from Jul 23, 2013, 3:26 PM.

Check the Console Output

The screenshot shows the Jenkins interface with the 'Console Output' tab selected. The main content area displays the build log for the job 'Sonatype-SNAPSHOT-41'. The log includes several git pull commands, a warning about a missing dependency, and a successful build message.

```
Started by user 'Minas Project'.
Building on the build-in node 'sonatype' (Jenkins Sonatype (Sonatype-SNAPSHOT-41))
The recommended git tool is: hg
No credentials specified
+ C:\Program Files\Git\cmd\git.exe pull -r C:\ProgramData\jenkins\jobs\jenkins.sonatype.SNAPSHOT-41\src\trunk.git
Fetching changes from the remote Git repository
+ C:\Program Files\Git\cmd\git.exe pull -r https://github.com/minas/SNAPSHOT-41.git +refs/heads/*:refs/remotes/origin/*
+ C:\Program Files\Git\cmd\git.exe config remote.origin.url https://github.com/minas/SNAPSHOT-41.git
+ C:\Program Files\Git\cmd\git.exe config remote.origin.fetch +refs/heads/*:refs/remotes/origin/*
+ C:\Program Files\Git\cmd\git.exe config remote.origin.url https://github.com/minas/SNAPSHOT-41.git
+ C:\Program Files\Git\cmd\git.exe config --get-url .git origin.SNAPSHOT-41
+ C:\Program Files\Git\cmd\git.exe fetch --progress -- https://github.com/minas/SNAPSHOT-41.git +refs/heads/*:refs/remotes/origin/*
+ C:\Program Files\Git\cmd\git.exe rev-parse --is-inside-work-tree > /dev/null
Checking out Revision 0b2e0d00c02d300000000000000000000000000 (refs/remotes/origin/SNAPSHOT-41)
+ C:\Program Files\Git\cmd\git.exe config core.sparsecheckout true
+ C:\Program Files\Git\cmd\git.exe checkout -f 0b2e0d00c02d300000000000000000000000000
Commit message: "update"
+ C:\Program Files\Git\cmd\git.exe add -u
+ C:\Program Files\Git\cmd\git.exe commit -m "update"
[SNAPSHOT-41:41] $ C:\ProgramData\jenkins\tools\plugins\sonar\sonar-runner\bin\sonar-scanner.bat
Scanner home: C:\ProgramData\jenkins\tools\plugins\sonar\sonar-runner\bin\sonar-scanner
Scanner version: 3.0.0.1020
Scanner configuration file: C:\ProgramData\jenkins\tools\plugins\sonar\sonar-runner\bin\sonar-scanner
Scanner logs: C:\ProgramData\jenkins\tools\plugins\sonar\sonar-runner\bin\sonar-scanner.log
```

```

Dashboard > Sonarqube Job 41 > 441 > Console Output
00:31:55.499 2020: Sensor Analysis warnings report [sensors] (done) | time=4ms
00:31:55.500 2020: Sensor Of File Caching Sensor [sensors]
00:31:55.500 2020: Incremental PR analysis: Could not determine issues from path, cache will not be updated. Consider setting "sonar-projectCache" property.
00:31:55.500 2020: Sensor Of File Caching Sensor [sensors] (done) | time=4ms
00:31:55.500 2020: Sensor Deep Coverage Sensor
00:31:55.500 2020: Sensor Deep Coverage Sensor [sensors] (done) | time=4ms
00:31:55.500 2020: Sonar Publisher SON provider for this project is: git
00:31:55.500 2020: SON Publisher A source file is to be analyzed
00:31:55.501 2020: SON Publisher A/A source files have been analyzed (done) | time=1ms
00:31:55.502 2020: CPS Escalator CPS calculation finished (done) | time=0ms
00:31:55.502 2020: SON Publisher ID: 'com.sonarsource.git:SonarQubeJob41'
00:31:55.509 2020: Analysis report generated in 21ms, size: 424+493.6 kB
00:31:55.510 2020: Analysis report compressed in 40ms, zip size:22.3 kB
00:31:56.001 2020: Analysis report released in 50ms
00:31:56.002 2020: ANALYSIS SONCOMPL, you can find the results at: http://localhost:9000/analysis/10-01-0041
00:31:56.002 2020: Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
00:31:56.003 2020: More about the report processing at: http://localhost:9000/doc/api/Analyses.html#analyze
00:31:56.017 2020: Analysis total time: 25.732 s
00:31:56.019 2020: SonarScanner engine completed successfully.
00:31:56.210 2020: ANALYSIS SONCOMPL
00:31:56.210 2020: Total time: 26.790s
Finished: 00:31:56

```

REST API Jenkins Job 2.1

Step 15: Once the build is complete, go back to SonarQube and check the project linked

The screenshot shows the SonarQube dashboard for a project named 'SonarJob'. The top navigation bar includes 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', 'Administration', and 'More'. Below the navigation is a search bar. The main content area features a large green 'Passed' status indicator. A yellow warning message states: 'The last analysis has warnings: 200 details'. The dashboard is divided into several sections: 'Recent Cycle' and 'Overall Code'. Under 'Recent Cycle', there are three cards: 'Security' (0 open issues, A grade), 'Reliability' (0 open issues, A grade), and 'Maintainability' (0 open issues, A grade). Under 'Overall Code', there are three cards: 'Accepted issues' (0), 'Coverage' (0.0% - 0-99.9%), and 'Implementations' (0.0%). At the bottom, there is a 'Security Hotspots' section with a count of 0.

CONCLUSION:

In this experiment, we have learned how to perform Jenkins SAST using SonarQube. For this, we used a docker image of SonarQube so as to not install it locally on our system. After installing the required configurations on Jenkins, using acoe from a gihub repository, we analyze its code using SonarQube. Once we build the project, we can see that the SonarQube project displays that the code has no errors.

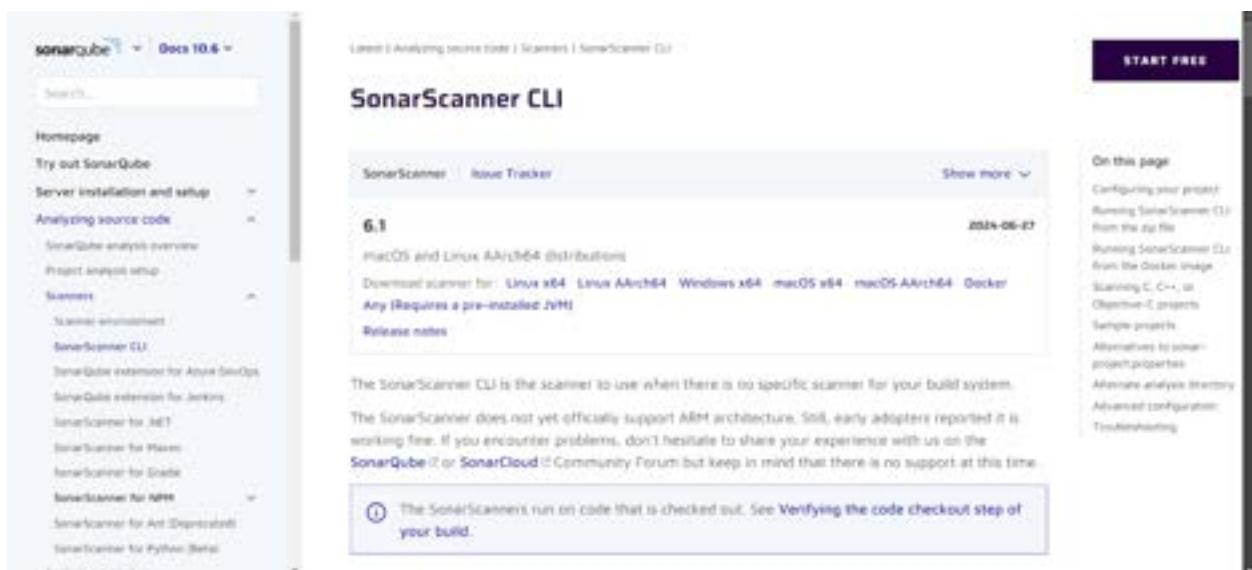
Experiment No: 8

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

PREREQUISITES:

Step 1: Download sonar scanner

<https://docs.sonarsource.com/sonarqube/latest/analyzing-source-code/scanners/sonarscanner/> . Visit this link and download the sonarqube scanner CLI



Extract the downloaded zip file in a folder.

Step 2: Docker Run **docker -v** command .If docker is not installed so install it

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

Step 3: Install sonarqube image Command: **docker pull sonarqube**

```
C:\Users\Student>docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
762bedf4b1b7: Pull complete
95f9bd9906fa: Pull complete
a32d681e6b99: Pull complete
aabdd0a18314: Pull complete
5161e45ecd8d: Pull complete
aeb0020dfa06: Pull complete
01548d361aea: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:bb444c58c1e04d8a147a3bb12af941c57e0100a5b21d10e599384d59b
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest

What's next:
    View a summary of image vulnerabilities and recommendations → docker

C:\Users\Student>
C:\Users\Student>
```

Step 4: Keep Jenkins installed on your system.

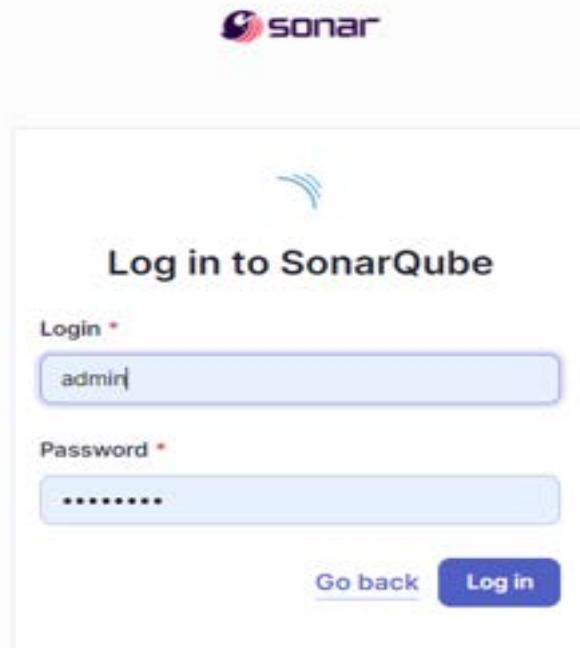
EXPERIMENT STEPS:

Step1: Run SonarQube image **docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest**. This command will run the SonarQube image that was just installed using docker.

```
C:\Users\Student>
C:\Users\Student>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
83330c33cd961d8d659f362c5f62c6cd1ff87f31ec99da134350b9b419370561

C:\Users\Student>
```

Step 2: Once the SonarQube image is started, you can go to **http://localhost:9000** to find the SonarQube that has started



Step 3: On this interface, login with **username = ‘admin’** and **password = ‘admin’**. Once logged in successfully, SonarQube will ask you to reset this password. Reset it and remember this password.

A screenshot of the SonarQube password update interface. The title is "Update your password". A yellow warning box contains the text "This account should not use the default password.". Below the warning are three input fields: "Old Password", "New Password", and "Confirm Password". At the bottom is a blue "Update" button.

Step 4: After changing the password, you will be directed to this screen. Click on **Create a Local Project**. Give the project a display name and project key

The screenshot shows the SonarQube web interface. At the top, there are navigation tabs: Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and More. A search bar is located at the top right. On the left, there's a sidebar with sections for My Favorites, Filters, Quality Gates (Passed: 1,000, Failed: 0), Reliability (A: 1,000, B: 0, C: 0, D: 0, E: 0), and Security (A: 1,000, B: 0). In the center, there's a project summary for 'D_41_Shivam PUBLIC'. It says 'Project's default branch is not analyzed yet. Configure analysis'. Below it is another project summary for 'D_SHIVAM_41 PUBLIC' with the note 'Last analysis: 4 hours ago' and 'The inspection of this project is empty'. At the bottom right of the main area, there's a green checkmark icon with the word 'Passed'. A large 'Create Project' button is visible on the right side of the interface.

Click on Create Project

1 of 2

Create a local project

Project display name *

SonarQube_Lab8



Project key *

SonarQube_Lab8



Main branch name *

main

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

[Cancel](#)

[Next](#)

Set up the project as required and click on create.

In the Step 2 while creating the project,Sonarqube ask you regarding which code should be considered as the new code for examining it .

The screenshot shows the 'Set up project for Clean as You Code' configuration page. At the top, there is a note: 'The new code definition sets which part of your code will be considered new code. This helps you focus attention on the most recent changes to your project, avoiding you to follow the Clean as You Code methodology. Learn more: [Defining New Code](#)'.

Below this, there is a section titled '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 releases 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 releases or releases.
 - Number of days:

Any code that has changed in the last x days is considered new code. If no action is taken on a new issue after x days, this issue will become part of the overall code.
Recommended for projects following continuous delivery.
 - Reference branch:

Choose a branch as the baseline for the new code.
Recommended for projects using feature branches.

At the bottom of the page, there are two buttons: 'Back' and 'Create project'. A warning message in a yellow box states: '⚠️ Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.' Below the message, it says 'SonarQube™ technology is licensed by QuliceSoftware Ltd.' and provides links for 'Community Edition', '4.x.x (latest) Artifex', '1.0.0+0', 'Community', 'Documentation', 'Plugins', and 'Web API'.

Click on Create

The screenshot shows the 'Analysis Method' configuration page in SonarQube. It displays several options for how to analyze a repository:

- With Jenkins**: Jenkins CI integration.
- With GitHub Actions**: GitHub Actions integration.
- With Bitbucket Pipelines**: Bitbucket Pipelines integration.
- With Jenkins CI**: Jenkins CI integration.
- With Azure Pipelines**: Azure Pipelines integration.
- Other CI**: Other Continuous Integration systems integration.

Below these options, there is a note about using Jenkins for building or evaluating code bases. A warning message states: "Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine."

Project is created

Step 5: Open **Jenkins** on whichever port it is installed. (<http://localhost:>). Go to the new item

The screenshot shows the Jenkins dashboard. On the left, there is a sidebar with navigation links such as 'New Item', 'Build History', 'Manage Jenkins', 'My View', 'Build Queue', 'Build Executor Status', 'Build Status', 'Jobs', and 'Builds'. The 'New Item' link is highlighted with a red box. The main area displays a table of build history for the project '41_SonarQube':

S	W	Name	Last Success	Last Failure	Last Duration	D
1	41_SonarQube	Not yet	N/A	N/A	0:0:0	
2	Change pipeline!	From 9 days ago	N/A	N/A	1:3 sec	
3	DevOps_Pipeline	Not yet	N/A	N/A	0:0:0	
4	Pipeline_DevOps	From 9 days ago	N/A	N/A	0:0:0	
5	SonarQube_41	Not yet	N/A	N/A	0:0:0	
6	sonarqube_41.job7	Not yet	4 hr 32 min ago	N/A	3:8 sec	
7	sonarqube.job7_41	4 hr 36 min ago	4 hr 32 min ago	4 hr 32 min ago	21 sec	

Step 6: Go to manage jenkins →available plugins then Search for **Sonarqube Scanner** for Jenkins and install it

Sonarqube search results:

- SonarQube Scanner 2.17.0**
Internal Jenkins Integration: Build Reports
This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.
- Sonar Gerrit 202.0.0**
Internal Jenkins Integration
This plugin allows to submit issues from SonarQube to Gerrit as comments directly.
- SonarQube Generic Coverage 1.0**
TODO

Step 7: Now, go to Manage Jenkins → System. Under Sonarqube servers, add a server. Add server authentication token if needed.

System → SonarQube servers

Add SonarQube server

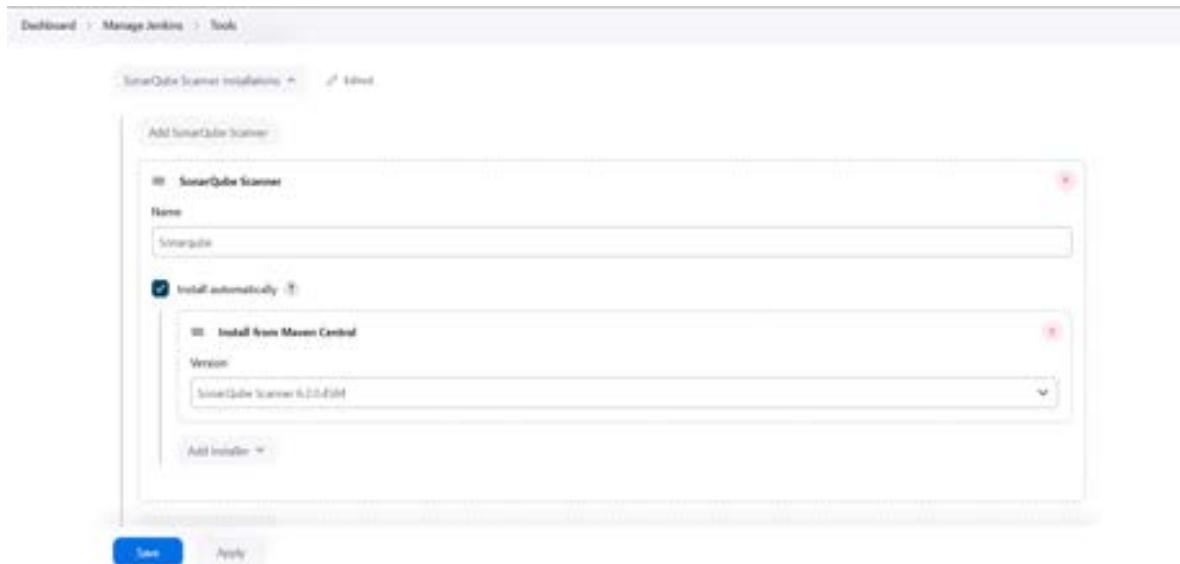
Server URL: http://localhost:9000

Server authentication token: (Leave empty if anonymous access is selected)

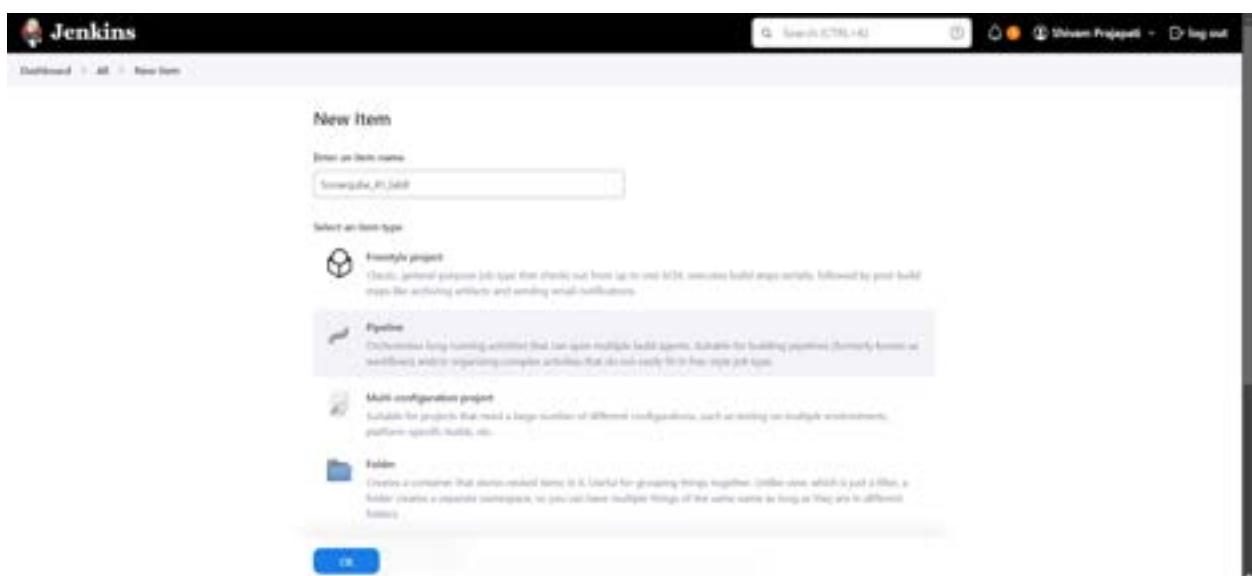
Advanced:

Save **Apply**

Step 8: Go to Manage Jenkins → Tools. Go to SonarQube scanner, choose the latest configuration and choose to install automatically.



Step 9: After configuring, click on **New Item** and select **Pipeline Project**



Step 10: Under Pipeline script, enter the following:

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

```
}

stage('SonarQube analysis') {
    withSonarQubeEnv('sonarqube') {
        bat """
            C:\\\\Users\\\\praja\\\\Downloads\\\\SonarqubeCLI\\\\sonar-scanner-6.1.0.4477-windows-x64\\\\bin\\\\sonar-scanner.bat ^
                -D sonar.login=admin ^
                -D sonar.password=Shivam2@ ^
                -D sonar.projectKey=SonarQube_Lab8 ^
                -D sonar.exclusions=vendor/**,resources/**, **/*.java ^
                -D sonar.host.url=http://localhost:9000/
            """
    }
}
```



Click on save.



This is a Java sample project with many repetitive sections and coding issues that SonarQube will be able to detect during analysis.

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

Stage View



The problem was C:\windows\system32 was not there so we need to add in our environment variable .

Now Check the console output once

Successfully BUILD

Step 12: After the build is finished, return to SonarQube and review the linked project in detail.

The screenshot displays the SonarQube web interface. At the top, there's a navigation bar with links for 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', 'Administration', and 'More'. A search bar and a 'Create Project' button are also present. On the left, there's a sidebar with 'My Favorites' (empty), 'Filters', and sections for 'Quality Gates' (Passed: 2, Failed: 0) and 'Reliability' (with a legend from green to red). The main content area shows two projects:

- Dr_Shivam_A1_HABCD**: Last analysis 1 day ago. Status: Passed. Summary: The main branch of this project is empty.
- Dr_SonarQube_Lab01_PUBLIC**: Last analysis 10 minutes ago. Lines of Code: 883k. Metrics: Security (D), Reliability (B), Maintainability (A), Duplication (0.0%), Coverage (50.6%).

Below these, there's a note: "Emitted statistics should be used for evaluation purposes only".

In the bottom section, the 'main' project is selected. It shows a summary card with a green checkmark and the word 'Passed'. Below it, there are tabs for 'New Code' and 'Overall Code'. The 'Overall Code' tab is active, displaying metrics for Security (0 Open Issues), Reliability (68k Open Issues), Maintainability (164k Open Issues), Accepted Issues (0), Coverage (50.6%), and Duplication (50.6%). There's also a 'Security Hotspots' section with a count of 0.

Under different options on the navbar , we can check all the issues with the code.

UNDER ISSUES:

1) Consistency

The screenshot shows the SonarQube web interface under the 'Issues' tab. On the left, a sidebar displays various quality profiles and their counts: Consistency (107), Intentionality (144), Acceptability (0), Responsibility (0), and Software Quality (0). The main panel lists several consistency-related issues with their assigned status (Open or Not assigned) and severity (Information).

- Issue 1: Insert a <DOCTYPE> declaration to before this <html> tag. Status: Open, Assigned to: Not assigned.
- Issue 2: Remove this deprecated "width" attribute. Status: Open, Assigned to: Not assigned.
- Issue 3: Remove this deprecated "align" attribute. Status: Open, Assigned to: Not assigned.

At the bottom of the page, a note states: "Embedded database should be used for evaluation purposes only".

2) Reliability

The screenshot shows the SonarQube web interface under the 'Issues' tab. The sidebar displays: Acceptability (0), Responsibility (0), Software Quality (0), Security (0), Reliability (109), and Maintainability (11). The main panel lists reliability-related issues with their assigned status and severity.

- Issue 1: Add "long" and/or "min-long" attributes to this "checkbox" element. Status: Open, Assigned to: Not assigned.
- Issue 2: Add "checkbox" listeners to this "checkbox". Status: Open, Assigned to: Not assigned.
- Issue 3: Add "long" and/or "min-long" attributes to this "select" element. Status: Open, Assigned to: Not assigned.

At the bottom of the page, a note states: "Embedded database should be used for evaluation purposes only".

3) Maintainability

The screenshot shows the SonarQube interface for the project "SonarQube.Lab". The left sidebar displays navigation links: Overview, Issues (selected), Security Hotspots, Measures, Code, and Activity. The main panel shows a list of issues under the "Maintainability" category. One specific issue is highlighted:

- Issue Detail:** `generalFile-incorrect-maintainability`
- Description:** `Use a specific version tag for the image.`
- Severity:** `Info`
- Type:** `Maintainability`
- Code Snippet:** `generalFile-incorrect-maintainability.js`
- Message:** `Add the "M", "S", or "C" keyword to this declaration of "prop" to make it explicit.`
- Impact:** `Info`
- Assignee:** `Not assigned`
- Time Left:** `4 years ago`
- Cost Estim:** `0 hours`
- Effort:** `0 hours`

A yellow warning bar at the bottom states: "Embedded database should be used for evaluation purposes only".

4) Severity

The screenshot shows the SonarQube interface for the project "SonarQube.Lab". The left sidebar displays navigation links: Overview, Issues (selected), Security Hotspots, Measures, Code, and Activity. The main panel shows a list of issues under the "Severity" category. One specific issue is highlighted:

- Issue Detail:** `generalFile-incorrect-severity`
- Description:** `Add the "M", "S", or "C" keyword to this declaration of "prop" to make it explicit.`
- Severity:** `Info`
- Type:** `Maintainability`
- Code Snippet:** `generalFile-incorrect-severity.js`
- Message:** `Add the "M", "S", or "C" keyword to this declaration of "prop" to make it explicit.`
- Impact:** `Info`
- Assignee:** `Not assigned`
- Time Left:** `4 years ago`
- Cost Estim:** `0 hours`
- Effort:** `0 hours`

A yellow warning bar at the bottom states: "Embedded database should be used for evaluation purposes only".

UNDER SECURITY HOTSPOT:

The screenshot shows the SonarQube interface for a project named "SonarQube". The "Security Hotspots" tab is selected. A prominent red circle indicates 6.0% Security Hotspots Reviewed. A specific hotspot is highlighted, showing the following details:

- Severity:** Critical
- Description:** The Docker image runs with root as the default user. Make sure it is safe here.
- Review priority:** Medium
- Status:** No review
- Code Snippet:**

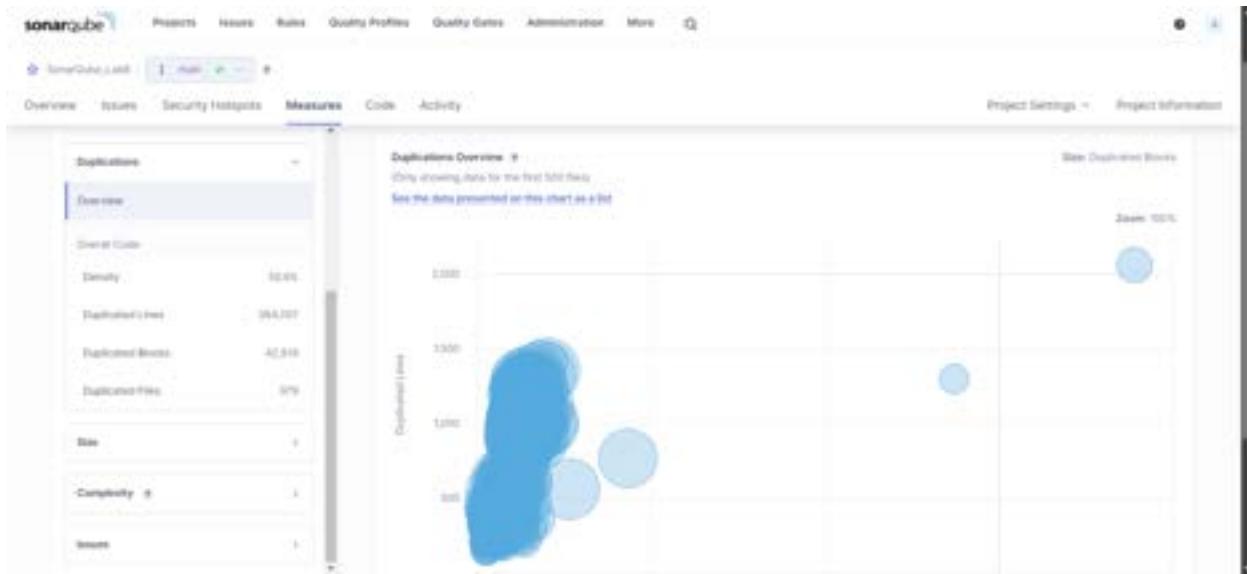
```
FROM node:12-alpine
USER root
CMD ["node", "app.js"]
```
- Review Buttons:** Review, To review, Automated, Fixed, Safe

UNDER MEASURES:

The screenshot shows the SonarQube interface for a project named "SonarQube". The "Measures" tab is selected. On the left, a sidebar lists various measures with their counts: Security (8), Readability (8), Maintainability (8), Security Review (8), Duplications (7), Size (8), and Complexity (8). The main area displays a chart titled "SonarQube.LHS" with the following data:

Measure	Value
Size	33.0%
Complexity	41.0%
Readability	55.0%

Below the chart, there is a note: "Only showing data for the first 1000 files. See the data presented on this chart as a list".



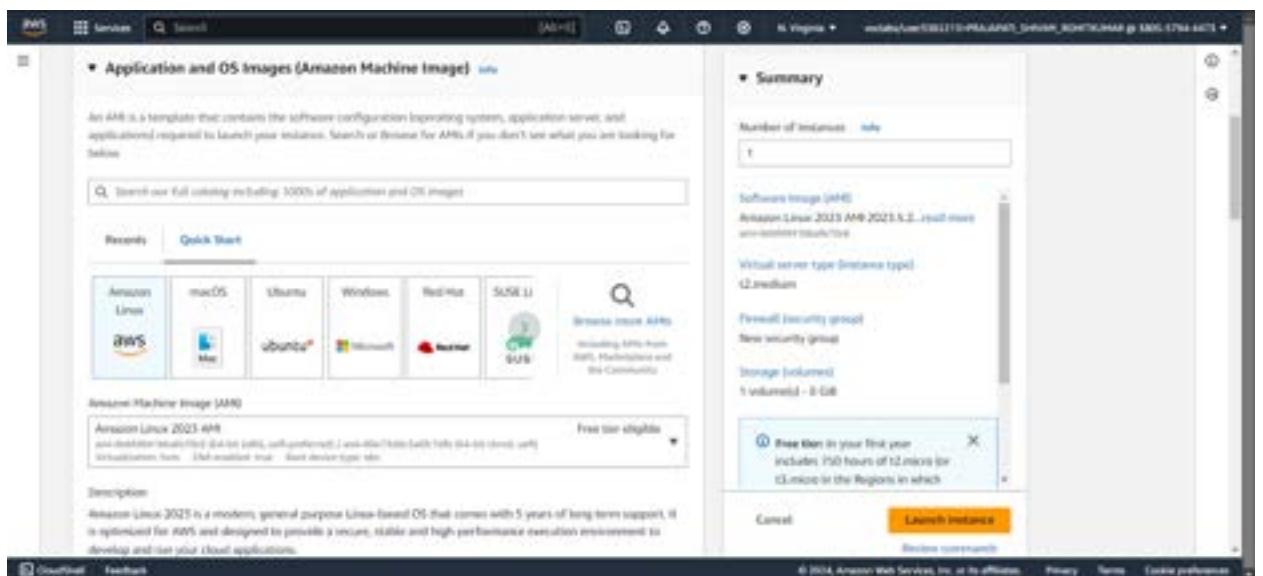
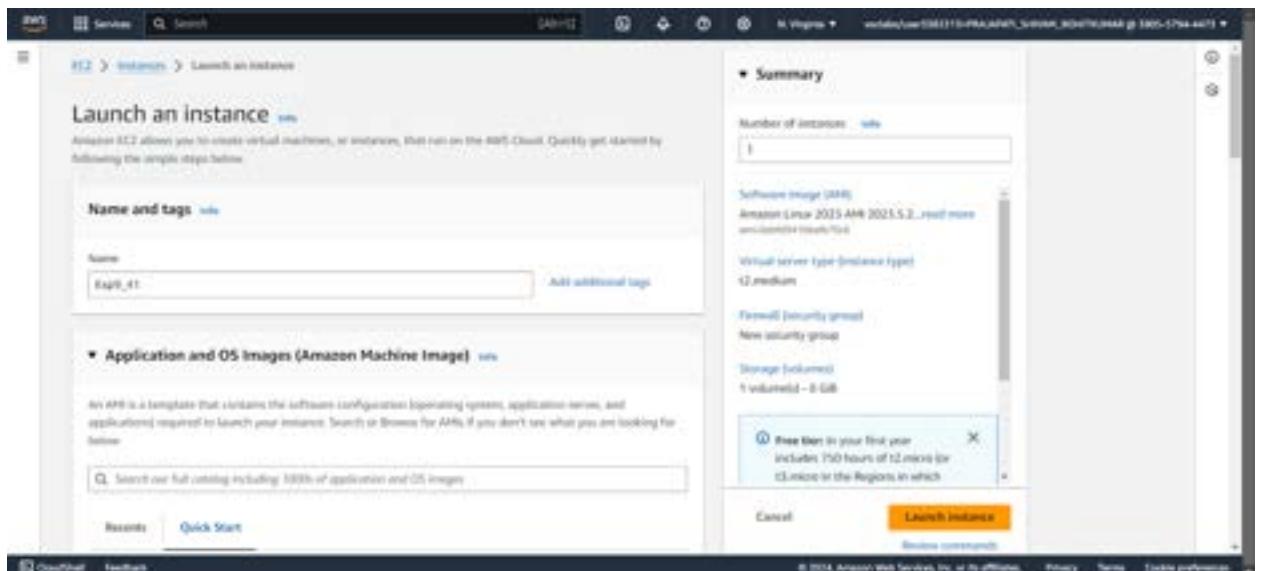
CONCLUSION:

In this experiment, we demonstrated how to perform static code analysis using Jenkins CI/CD Pipeline with SonarQube integration. We created a pipeline project with a specific script that contains all the instructions necessary to run SonarQube analysis. After configuring Jenkins appropriately, we built the project. The analyzed code in this experiment had several issues, such as errors, bugs, and duplications, all of which were detected and displayed in the linked SonarQube project.

Experiment No: 9

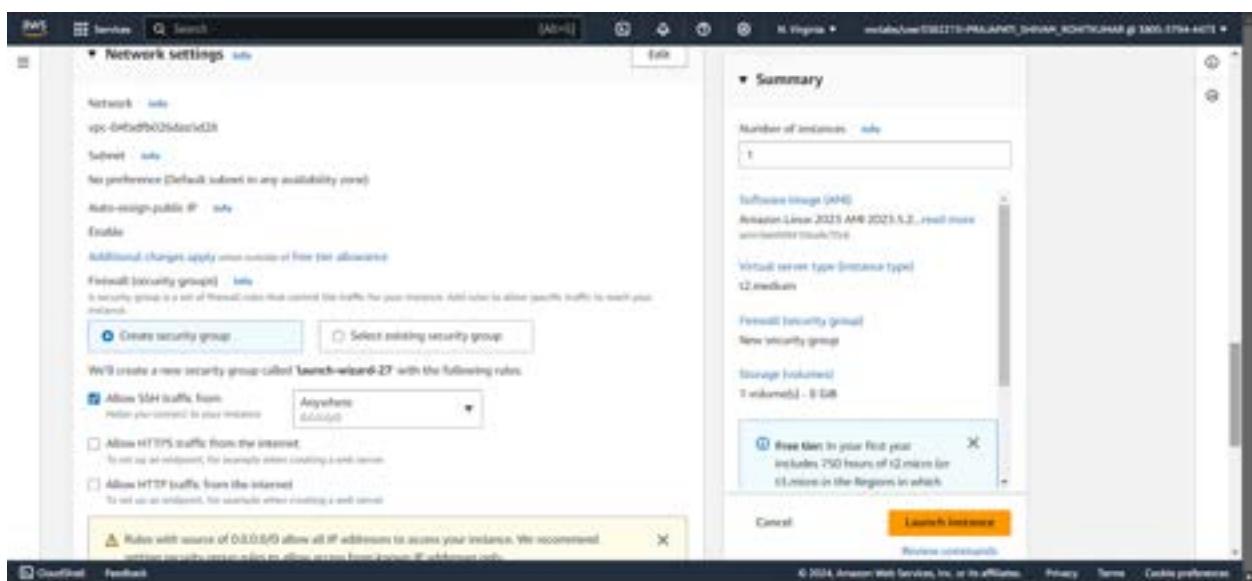
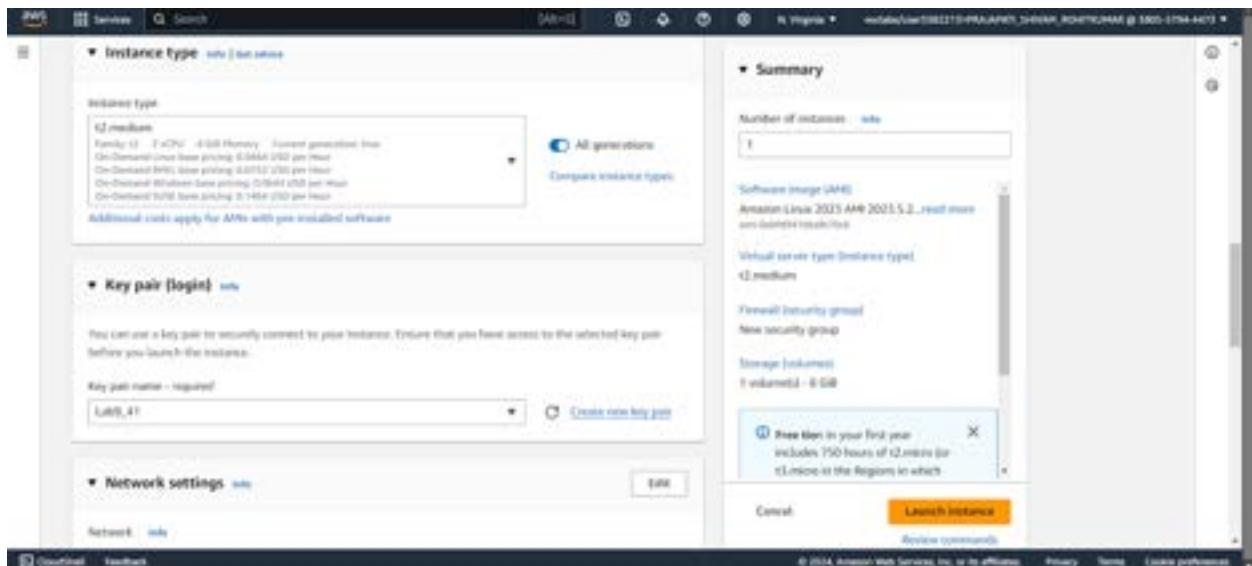
AIM: To Understand Continuous monitoring and Installation and configuration of Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine.

Step 1: Sign in to your AWS account. Look for EC2 in the services list. Open it and click on "Create Instance."



Select The OS Image as Amazon Linux.

Step 2: If you haven't created a private key or a .pem file yet, click on "Create a key pair." Otherwise, choose the key pair you created earlier. (Be sure to remember where the .pem file for that key is located on your system.) In my case I have created a new one.



AWS will create a security group for this instance. Keep the name of that instance saved.

Instance:



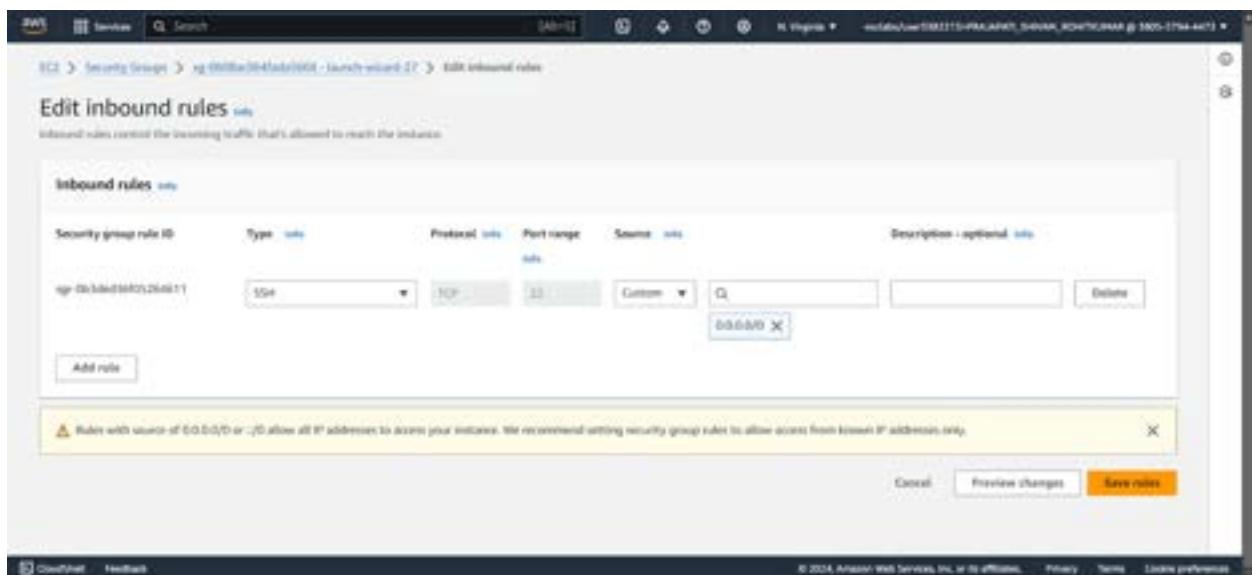
Step3: After you create the instance, click on "Security Groups" in the left sidebar. Look for the security group that corresponds to your instance, and then click on the security group ID for that group. (in my case launch-wizard-27 is the latest one.)

Name	Security group ID	Security group name	VPC ID	Description
-	sg-060Bac064fada5668	launch-wizard-27	vpc-04fafffb03e0a228	Launch-wiz...
-	sg-042109e664d31c39	launch-wizard-22	vpc-04fafffb03e0a228	Launch-wiz...
-	sg-0673af81123ec0948	launch-wizard-21	vpc-04fafffb03e0a228	Launch-wiz...
-	sg-01a80791a059a030	launch-wizard-18	vpc-04fafffb03e0a228	Launch-wiz...
-	sg-08ff7e0f03a7a1a62	launch-wizard-7	vpc-04fafffb03e0a228	Launch-wiz...
-	sg-08ff7e0f123007e630	launch-wizard-4	vpc-04fafffb03e0a228	Launch-wiz...
-	sg-0125af4f788d98527	Permit	vpc-04fafffb03e0a228	Security gr...
-	sg-0322f011239ddca22	launch-wizard-8	vpc-04fafffb03e0a228	Launch-wiz...

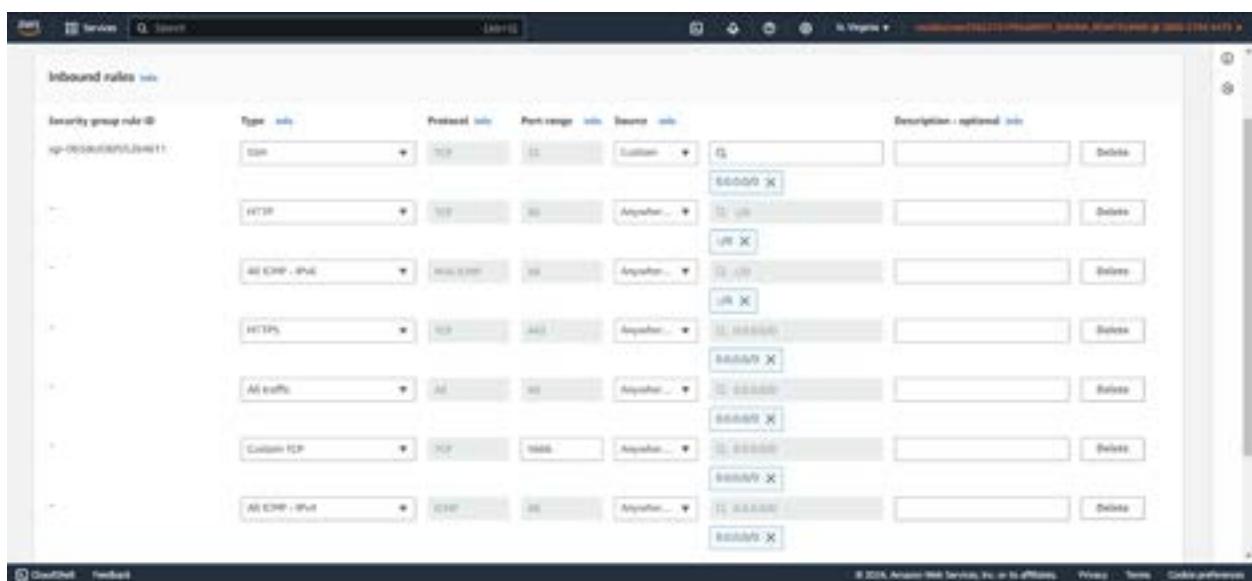
Click on Id

Name	Security group rule...	IP version	Type	Protocol	Port range
-	-	-	-	-	-

Click on the edit inbound rules



Next, click on "Add rules" and set up rules for the following protocols: HTTP, All ICMP (IPv6), HTTPS, All traffic, Custom TCP (Port 5666), and All ICMP (IPv4).



Click on save. This will add all the inbound rules to the security group.

The screenshot shows the AWS Security Groups console. The top navigation bar includes 'AWS Services' and 'Search'. The left sidebar has sections for EC2 Dashboard, EC2 Global View, Events, Connect-to-Code, Instances, Images, Elastic Block Store, and more. The main content area displays the 'sg-0608ac064fada5668 - launch-wizard-27' security group. The 'Details' tab is selected, showing the security group name, ID, description, owner, and various counts. Below this is the 'Inbound rules' tab, which lists several rules with columns for Name, Security group rule..., IP version, Type, Protocol, Port range, and Source.

Name	Security group rule...	IP version	Type	Protocol	Port range	Source
sg-0608ac064fada5668	Inbound	IPv4	Custom TCP	TCP	3000	0.0.0.0/0
sg-0608ac064fada5668	Inbound	IPv4	HTTP	TCP	80	0.0.0.0/0
sg-0608ac064fada5668	Inbound	IPv4	All ICMP - IPv4	ICMP	40-49	0.0.0.0/0
sg-0608ac064fada5668	Inbound	IPv4	SSH	TCP	22	0.0.0.0/0
sg-0608ac064fada5668	Inbound	IPv4	HTTPS	TCP	443	0.0.0.0/0
sg-0608ac064fada5668	Inbound	IPv4	All ICMP - IPv6	ICMP	40-49	0.0.0.0/0
sg-0608ac064fada5668	Inbound	IPv6	All traffic	All	All	0.0.0.0/0

Step 4: Return to the instances screen and click on the instance ID of your instance. Then, click on "Connect."

The screenshot shows the AWS Instances (EC2) console. The top navigation bar includes 'AWS Services' and 'Search'. The left sidebar has sections for EC2 Dashboard, EC2 Global View, Events, Connect-to-Code, Instances, Images, Elastic Block Store, and more. The main content area displays a list of instances. One instance, 'Launch-wizard-27', is highlighted in green and has a 'Connect' button next to it. Other instances listed include '41_Workshop1', '41_Workshop2', '41_Host1', '41_Host2', '41_Host3', '41_Host4', '41_Host5', '41_Host6', '41_Host7', '41_Host8', and '41_Host9'. Each instance row contains columns for Name, Instance ID, Instance state, Instance type, Status check, Instance status, Availability Zone, Public IPv4 (IP), and Private IP (IP).

Click on "SSH client" and copy the example command provided.



Step 5: Now, we need to connect our local terminal to the instance using SSH. Open the terminal where your private key file (.pem) is located by actually going to the folder which has the .pem file, paste the copied SSH command, and run it.

```
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\praja\OneDrive\Desktop\NewFolder> ssh -i "Lab9_41.pem" ec2-user@ec2-52-91-78-149.compute-1.amazonaws.com
The authenticity of host 'ec2-52-91-78-149.compute-1.amazonaws.com (52.91.78.149)' can't be established.
ED25519 key fingerprint is SHA256:Xyho0mkRy0vaad7JNbpHyyW9VnFJ7QIUTuLBrehVayc.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-52-91-78-149.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
[ec2-user@ip-172-31-84-149 ~]$ |
```

Step 6: Now we start working on this terminal. First run the command **sudo yum update**. This command will check for any updates for the YUM library to ensure that all libraries are up to date with the latest features and security fixes

```
[ec2-user@ip-172-31-84-149 ~]$ sudo yum update
Last metadata expiration check: 0:30:14 ago on Sat Sep 28 07:51:31 2024.
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-84-149 ~]$ |
```

Step 7: We are going to set up a web server software application called **Apache** and a programming language called **PHP** in this instance . To do this, run this command in your terminal

sudo yum install httpd php

```
[ec2-user@ip-172-31-84-149 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:31:34 ago on Sat Sep 28 07:51:31 2024.
Dependencies resolved.
=====
 Package           Architecture Version       Repository      Size
=====
Installing:
 httpd            x86_64      2.4.62-1.amzn2023   amazonlinux   48 k
 php8_3           x86_64      8.3.10-1.amzn2023.0.1  amazonlinux   10 k
Installing dependencies:
 apr              x86_64      1.7.2-2.amzn2023.0.2  amazonlinux   129 k
 apr-util         x86_64      1.6.3-1.amzn2023.0.1  amazonlinux   98 k
 generic-logos-httd  noarch    18.0.0-12.amzn2023.0.3  amazonlinux   19 k
 httpd-core       x86_64      2.4.62-1.amzn2023   amazonlinux   1.4 M
 httpd-filesystem noarch    2.4.62-1.amzn2023   amazonlinux   14 k
 httpd-tools      x86_64      2.4.62-1.amzn2023   amazonlinux   81 k
 libbrotli        x86_64      1.0.9-4.amzn2023.0.2  amazonlinux   315 k
 libsodium        x86_64      1.0.19-4.amzn2023   amazonlinux   176 k
 libxml2          x86_64      1.1.34-5.amzn2023.0.2  amazonlinux   241 k
 mailcap          noarch    2.1.49-3.amzn2023.0.3  amazonlinux   33 k
 nginx-filesystem noarch    1:1.24.0-1.amzn2023.0.4  amazonlinux   9.8 k
 php8_3-cli       x86_64      8.3.10-1.amzn2023.0.1  amazonlinux   3.7 M
 php8_3-common    x86_64      8.3.10-1.amzn2023.0.1  amazonlinux   737 k
 php8_3-process   x86_64      8.3.10-1.amzn2023.0.1  amazonlinux   45 k
 php8_3-xsl       x86_64      8.3.10-1.amzn2023.0.1  amazonlinux   154 k
Installing weak dependencies:
 apr-util-openssl x86_64      1.6.3-1.amzn2023.0.1  amazonlinux   17 k
 mod_http2        x86_64      2.0.27-1.amzn2023.0.3  amazonlinux   166 k
 mod_lua          x86_64      2.4.62-1.amzn2023   amazonlinux   61 k
=====
Installed:
 apr-1.7.2-2.amzn2023.0.2.x86_64
 apr-util-openssl-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
 libxml2-1.1.34-5.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-httd-18.0.0-12.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
mailcap-2.1.49-3.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-mbstring-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!
[ec2-user@ip-172-31-84-149 ~]$ |
```

Step 8: Now, we will **install the GCC compiler**, which is used for compiling and running C and C++ programs, along with the essential C libraries. To do this, enter the following command: **sudo yum install gcc glibc glibc-common**

```
[ec2-user@ip-172-31-84-149 ~]$ sudo yum install gcc glibc glibc-common
Last metadata expiration check: 0:34:28 ago on Sat Sep 28 07:51:31 2024.
Package glibc-2.34-52.amzn2023.0.11.x86_64 is already installed.
Package glibc-common-2.34-52.amzn2023.0.11.x86_64 is already installed.
Dependencies resolved.
=====
 Package           Architecture   Version        Repository      Size
=====
Installing:
 gcc              x86_64        11.4.1-2.amzn2023.0.2      amazonlinux    32 M
Installing dependencies:
 annobin-docs      noarch       10.93-1.amzn2023.0.1      amazonlinux    92 k
 annobin-plugin-gcc x86_64        10.93-1.amzn2023.0.1      amazonlinux    887 k
 cpp              x86_64        11.4.1-2.amzn2023.0.2      amazonlinux    10 M
 gc               x86_64        8.0.4-5.amzn2023.0.2      amazonlinux    105 k
 glibc-devel       x86_64        2.34-52.amzn2023.0.11     amazonlinux    27 k
 glibc-headers-x86 x86_64        2.34-52.amzn2023.0.11     amazonlinux    427 k
 guile22          x86_64        2.2.7-2.amzn2023.0.3      amazonlinux    6.4 M
 kernel-headers    x86_64        6.1.109-118.189.amzn2023 amazonlinux    1.4 M
 libmpc           x86_64        1.2.1-2.amzn2023.0.2      amazonlinux    62 k
 libtool-ltdl     x86_64        2.4.7-1.amzn2023.0.3      amazonlinux    38 k
 libxcrypt-devel   x86_64        4.4.33-7.amzn2023      amazonlinux    32 k
 make             x86_64        1:4.3-5.amzn2023.0.2      amazonlinux    534 k
=====
Transaction Summary
=====
Install 13 Packages

Total download size: 52 M
```

```
Installed:
 annobin-docs-10.93-1.amzn2023.0.1.noarch
 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

Complete!
[ec2-user@ip-172-31-84-149 ~]$ |
```

Step 9: Next, we need to **install the GD library**, along with its development tools. This library helps with creating and manipulating images. For that, run this command **sudo yum install gd gd-devel**

```
[ec2-user@ip-172-31-84-149 ~]$ sudo yum install gd gd-devel
Last metadata expiration check: 0:36:28 ago on Sat Sep 28 07:51:31 2024.
Dependencies resolved.
=====
 Package           Architecture   Version        Repository      Size
=====
Installing:
 gd              x86_64        2.3.3-8.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    310 k
 brootl-devel    x86_64        1.0.9-4.amzn2023.0.2      amazonlinux    31 k
 bztp2-devel     x86_64        1.0.8-6.amzn2023.0.2      amazonlinux    210 k
 cairo           x86_64        1.17.6-2.amzn2023.0.1      amazonlinux    684 k
 cimage-fs        x86_64        3.22.2-1.amzn2023.0.4      amazonlinux    16 k
 fontconfig       x86_64        2.13.94-2.amzn2023.0.2      amazonlinux    273 k
 fontconfig-devel x86_64        2.13.94-2.amzn2023.0.2      amazonlinux    128 k
 fonts-fs         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
```

```

libjpeg-turbo-devel-2.1.4-2.amzn2023.0.5.x86_64
libpng-2.1.6.37-10.amzn2023.0.6.x86_64
libsdl2-devel-3.0-5.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-0.49.0-3.amzn2023.0.3.x86_64
xml-common-0.6.3-56.amzn2023.0.2.noarch
xz-devel-5.2.5-9.amzn2023.0.2.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-3.40.1-2.amzn2023.0.2.x86_64
xorg-x11proto-devel-2021.4-1.amzn2023.0.2.noarch
zlib-devel-1.2.11-33.amzn2023.0.5.x86_64

Complete!
[ec2-user@ip-172-31-84-149 ~]$ |

```

Step 10: Now, we create a user called '**nagios**' and make sure that it has a home directory, and set up a password for it.

sudo adduser -m nagios

sudo passwd nagios

```

[ec2-user@ip-172-31-84-149 ~]$ sudo adduser -m nagios
sudo passwd nagios
Changing password for user nagios.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
Sorry, passwords do not match.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-84-149 ~]$ |

```

Password :Shivam2@

Step 11: Now, we need to create a user group named **nagcmd**, which will be used to execute Nagios commands. To do this, run the following command: **sudo groupadd nagcmd**

```

[ec2-user@ip-172-31-84-149 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-84-149 ~]$ |

```

Step 12: Next, we'll add the users **apache** and **nagios** to the **nagcmd** group. This allows them to execute Nagios commands.

sudo usermod -a -G nagcmd nagios

sudo usermod -a -G nagcmd apache

```
[ec2-user@ip-172-31-84-149 ~]$ sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache
[ec2-user@ip-172-31-84-149 ~]$ |
```

Step 13: We'll create a directory called **downloads** to store the files related to the Nagios server that we download.

mkdir ~/downloads

cd ~/downloads

```
[ec2-user@ip-172-31-84-149 ~]$ mkdir ~/downloads
cd ~/downloads
[ec2-user@ip-172-31-84-149 downloads]$ |
```

Step 14: Now we need to install the latest versions of nagios-core and nagios-plugins. Go to the respective websites and check whether a better version is available. If newer versions are available, then right click on the download button → Copy link address. Paste this link address in place of the current link in command. If not run these commands.

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

```
[ec2-user@ip-172-31-84-149 downloads]$ wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
--2024-09-28 08:37:22-- https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2608:3c00::f03c:92ff:feff:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: 'nagios-4.5.5.tar.gz'

nagios-4.5.5.tar.gz      100%[=====] 1.97M  5.07MB/s   in 0.4s
2024-09-28 08:37:23 (5.07 MB/s) - 'nagios-4.5.5.tar.gz' saved [2065473/2065473]
[ec2-user@ip-172-31-84-149 downloads]$ |
```

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

```
[ec2-user@ip-172-31-84-149 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
--2024-09-28 08:38:31-- 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  7.16MB/s   in 0.4s
2024-09-28 08:38:32 (7.16 MB/s) - 'nagios-plugins-2.4.11.tar.gz' saved [2753049/2753049]
[ec2-user@ip-172-31-84-149 downloads]$ |
```

Step 15: Now, we need to extract the Nagios Core file into the same directory. We can do this using the tar command.

```
tar zxvf nagios-4.5.5.tar.gz
```

The screenshot shows a terminal window with the following output:

```
[ec2-user@ip-172-31-84-149 downloads]$ tar zxvf nagios-4.5.5.tar.gz
nagios-4.5.5/
nagios-4.5.5/.github/
nagios-4.5.5/.github/workflows/
nagios-4.5.5/.github/workflows/test.yml
nagios-4.5.5/.gitignore
nagios-4.5.5/CONTRIBUTING.md
nagios-4.5.5/Changelog
nagios-4.5.5/INSTALLING
nagios-4.5.5/LEGAL
nagios-4.5.5/LICENSE
nagios-4.5.5/Makefile.in
nagios-4.5.5/worker/Makefile.in
nagios-4.5.5/worker/ping/
nagios-4.5.5/worker/ping/.gitignore
nagios-4.5.5/worker/ping/Makefile.in
nagios-4.5.5/worker/ping/worker-ping.c
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
[ec2-user@ip-172-31-84-149 downloads]$ |
```

Step16: Now, we need to ensure that Nagios uses the nagcmd group for executing external commands.

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

```
[ec2-user@ip-172-31-84-149 downloads]$ ./configure --with-command-group=nagcmd
-bash: ./configure: No such file or directory
[ec2-user@ip-172-31-84-149 downloads]$ |
```

An error was encountered here: `./configure`: no such path or directory . So Navigate to the nagios-4.5.5 folder in downloads. (version could vary)

ls :

```
[ec2-user@ip-172-31-84-149 downloads]$ ls
nagios-4.5.5  nagios-4.5.5.tar.gz  nagios-plugins-2.4.11.tar.gz
[ec2-user@ip-172-31-84-149 downloads]$ |
```

- `cd nagios-4.5.5` (use the version shown by your `ls` command)

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

Another error could be Cannot find SSL headers. To solve this, we need to install OpenSSL Dev Library : `sudo yum install openssl-devel`

```
Last metadata expiration check: 8:58:10 ago on Sat Sep 28 07:51:31 2024.
Dependencies resolved.
=====
| Package           | Repository | Arch          | Version        | Size   |
| ======           | ======     | ======       | ======       | ===== |
| Installing:      |            |              |              |        |
|   openssl-devel  | amazonlinux | 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-3.0.8-1.amzn2023.0.14.x86_64.rpm          18 MB/s | 3.0 MB  00:00
Total                                         14 MB/s | 3.0 MB  00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
```

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

```
[ec2-user@ip-172-31-84-149 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 stdint.h... yes
checking for strings.h... yes
checking for sys/stat.h... yes
checking for sys/types.h... yes
```

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

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

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

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

[ec2-user@ip-172-31-84-149 nagios-4.5.5]$ |
```

Step 17: Next, we need to compile all the components of the software based on the instructions in the Makefile. For that, use this command: **make all** Then, **sudo make install**

sudo make install-init

sudo make install-config

sudo make install-commandmode

```
[ec2-user@ip-172-31-84-149 nagios-4.5.5]$ make all
cd ./base && make
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o nagios.o ./nagios.c
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o broker.o broker.c
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o netmods.o netmods.c
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o ../common/shared.o ./common/shared.c
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o query-handler.o query-handler.c
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o workers.o workers.c
In function 'get_wproc_list':
  inlined from 'get_worker' at workers.c:277:12:
workers.c:253:17: warning: 'ss' directive argument is null [-Wformat-overflow]
  253 |           log_debug_info(DNSMq_CHECKS, 1, "Found specialized workers() for \"%s\", (class %s %s) in %s\n"
) | stash | cmd_name); |
|
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o checks.o checks.c
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o config.o config.c
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o commands.o commands.c
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o events.o events.c
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o flapping.o flapping.c
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o logging.o logging.c
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o macros-base.o ../common/macros.c
gcc -Wall -I.. -I.. -I./lib -I../include -I../include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o netutils.o netutils.c
```

*** Support Notes *****

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

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

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

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

For more information on obtaining support for Nagios, visit:

<https://support.nagios.com>

Enjoy.

```
[ec2-user@ip-172-31-84-149 nagios-4.5.5]$ |
```

```
[ec2-user@ip-172-31-84-149 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
for file in *.cgi; do \
    /usr/bin/install -c -s -m 775 -o nagios -g nagios $file /usr/local/nagios/sbin; \
done
make[2]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
cd ./html && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/html'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/media
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/stylesheets
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/contexthelp
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/docs
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/docs/images
```

*** Config files installed ***

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

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

*** External command directory configured ***

```
[ec2-user@ip-172-31-84-149 nagios-4.5.5]$ |
```

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**

```
[ec2-user@ip-172-31-84-149 nagios-4.5.5]$ sudo nano /usr/local/nagios/etc/objects/contacts.cfg|
```

```

GNU nano 5.8                               /usr/local/nagios/etc/objects/contacts.cfg
=====
# CONTACTS.CFG - SAMPLE CONTACT/CONTACTGROUP DEFINITIONS
#
#
# NOTES: This config file provides you with some example contact and contact
# group definitions that you can reference in host and service
# definitions.
#
# You don't need to keep these definitions in a separate file from your
# other object definitions. This has been done just to make things
# easier to understand.
#
#
# CONTACTS
#
#
# Just one contact defined by default - the Nagios admin (that's you)
# This contact definition inherits a lot of default values from the
# 'generic-contact' template which is defined elsewhere.

[ Read 51 lines ]
  ⌂ Help   ⌂ Write Out   ⌂ Where Is   ⌂ Cut   ⌂ Execute   ⌂ Location   ⌂ Undo   ⌂ Set Mark
  ⌂ Exit   ⌂ Read File   ⌂ Replace   ⌂ Paste   ⌂ Justify   ⌂ Go To Line   ⌂ Redo   ⌂ Copy

```

Here, change the email under 'define contact{}' to your email address

```

GNU nano 5.8                               /usr/local/nagios/etc/objects/contacts.cfg
=====
# CONTACTS
#
#
# Just one contact defined by default - the Nagios admin (that's you)
# This contact definition inherits a lot of default values from the
# 'generic-contact' template which is defined elsewhere.

define contact {
    contact_name      nagiosadmin          : Short name of user
    use               generic-contact       : Inherit default values from generic-contact template (defined above)
    alias             Nagios Admin        : Full name of user
    email             2922.shivam.prajapati@es.ac.in  ***** CHANGE THIS TO YOUR EMAIL ADDRESS *****
}

[ Read 51 lines ]
  ⌂ Help   ⌂ Write Out   ⌂ Where Is   ⌂ Cut   ⌂ Execute   ⌂ Location   ⌂ Undo   ⌂ Set Mark
  ⌂ Exit   ⌂ Read File   ⌂ Replace   ⌂ Paste   ⌂ Justify   ⌂ Go To Line   ⌂ Redo   ⌂ Copy

```

To save this use the following shortcut sequence **CTRL+O**→**Enter**→**CTRL+X**.

CTRL+O: Overwrite the existing file with edited file

CTRL+X: Exit nano editor

Step 19: We need to install the necessary configuration files for the Nagios web interface.
sudo make install-webconf

```
[ec2-user@ip-172-31-84-149 nagios-4.5.5]$ [ec2-user@ip-172-31-84-149 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 [ $? -eq 1 ]; then \
    ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagios.conf; \
fi
*** Nagios/Apache conf file installed ***
[ec2-user@ip-172-31-84-149 nagios-4.5.5]$ |
```

Step 20: Now we need to create a user to access the Nagios web interface. For that, run this command to create a user named '**nagiosadmin**'. Keep this username and password saved as it is needed to login to the web interface. **sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin**

```
[ec2-user@ip-172-31-84-149 nagios-4.5.5]$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
[ec2-user@ip-172-31-84-149 nagios-4.5.5]$ |
```

Kingmaker is the password

Step 21: Now, we need to restart the Apache server to apply all the recent configurations. Use this command: **sudo service httpd restart**

```
[ec2-user@ip-172-31-84-149 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-84-149 nagios-4.5.5]$ |
```

Step 22: Now we go back to the downloads folder and extract the files of nagios plugin.
cd ~/downloads
tar zxvf nagios-plugins-2.4.11.tar.gz (Version may vary)

```
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-84-149 nagios-4.5.5]$ cd ~/downloads
[ec2-user@ip-172-31-84-149 downloads]$ |
```

```
[ec2-user@ip-172-31-84-149 downloads]$ tar zxvf nagios-plugins-2.4.11.tar.gz
nagios-plugins-2.4.11/
nagios-plugins-2.4.11/build-aux/
nagios-plugins-2.4.11/build-aux/compile
nagios-plugins-2.4.11/build-aux/config.guess
nagios-plugins-2.4.11/build-aux/config.rpath
nagios-plugins-2.4.11/build-aux/config.sub
nagios-plugins-2.4.11/build-aux/install-sh
nagios-plugins-2.4.11/build-aux/ltmain.sh
nagios-plugins-2.4.11/build-aux/missing
nagios-plugins-2.4.11/build-aux/mkinstalldirs
nagios-plugins-2.4.11/build-aux/depcomp
nagios-plugins-2.4.11/build-aux/snippet/
nagios-plugins-2.4.11/build-aux/snippet/_Noreturn.h
nagios-plugins-2.4.11/build-aux/snippet/arg-nonnull.h
nagios-plugins-2.4.11/build-aux/snippet/c++defs.h
nagios-plugins-2.4.11/build-aux/snippet/warn-on-use.h
nagios-plugins-2.4.11/build-aux/test-driver
```

Step 23: Again, we need to install the configurations for these files.

`cd nagios-plugins-2.4.11` (version may vary)

```
[ec2-user@ip-172-31-84-149 downloads]$ cd nagios-plugins-2.4.11
[ec2-user@ip-172-31-84-149 nagios-plugins-2.4.11]$ |
```

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

```
[ec2-user@ip-172-31-84-149 nagios-plugins-2.4.11]$ ./configure --with-nagios-user=nagios --with-nagios-group=nagios
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether to enable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C99... none needed
checking whether gcc understands -c and -o together... yes
checking whether make supports the include directive... yes (GNU style)
checking dependency style of gcc... gcc3
checking how to run the C preprocessor... gcc -E
checking for grep that handles long lines and -e... /usr/bin/grep
checking for egrep... /usr/bin/grep -E
```

Step 24: We need to compile all the components of this software based on the instructions in the Makefile.

make**sudo make install**

```
[ec2-user@ip-172-31-84-149 nagios-plugins-2.4.11]$ make
make all-recursive
make[1]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
Making all in gl
make[2]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/gl'
make all-recursive
make[3]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/gl'
make[4]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/gl'
make[4]: Nothing to be done for 'all-am'.
```

```
[ec2-user@ip-172-31-84-149 nagios-plugins-2.4.11]$ sudo make install
Making install in gl
make[1]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/gl'
make install-recursive
make[2]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/gl'
make[3]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/gl'
make[4]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/gl'
if test yes = no; then \
  case 'linux-gnu' in \
    darwin[56]*) \
      need_charset_alias=true ;; \
    darwin* | cygwin* | mingw* | pw32* | cegcc*) \
      need_charset_alias=false ;; \
  *) \
    need_charset_alias=true ;; \
esac ; \
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-172-31-84-149 nagios-plugins-2.4.11]$ |
```

Step 25: We need to register the Nagios service with the system to enable it to manage the server status

sudo chkconfig --add nagios**sudo chkconfig nagios on**

```
[ec2-user@ip-172-31-84-149 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.
```

Step 26: We need to verify the Nagios configuration for any syntax errors or issues before starting or restarting the Nagios service.

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

```
[ec2-user@ip-172-31-84-149 nagios-plugins-2.4.11]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
Nagios Core 4.5.5
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2024-09-17
License: GPL

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

Running pre-flight check on configuration data...

Checking objects...
  Checked 8 services.
  Checked 1 hosts.
  Checked 1 host groups.
  Checked 0 service groups.
  Checked 1 contacts.
  Checked 1 contact groups.
  Checked 24 commands.
  Checked 5 time periods.
  Checked 0 host escalations.
  Checked 0 service escalations.

Checking for circular paths...
```

```
sudo service nagios start
```

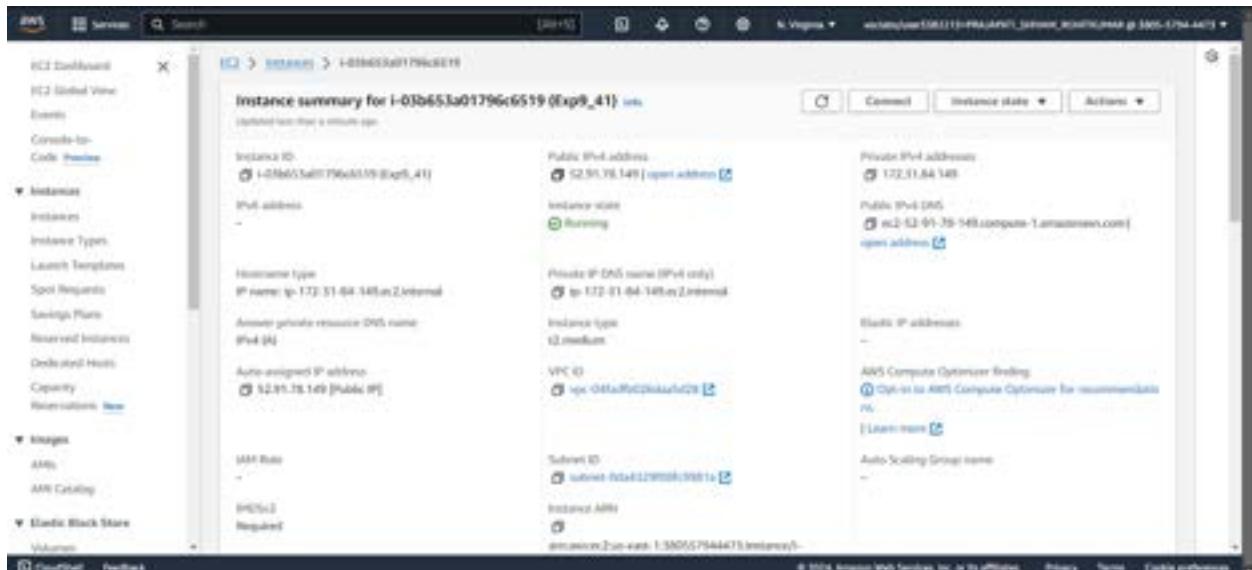
```
[ec2-user@ip-172-31-84-149 nagios-plugins-2.4.11]$ cd
[ec2-user@ip-172-31-84-149 ~]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-84-149 ~]$ |
```

Step 27: Check the status of the nagios.

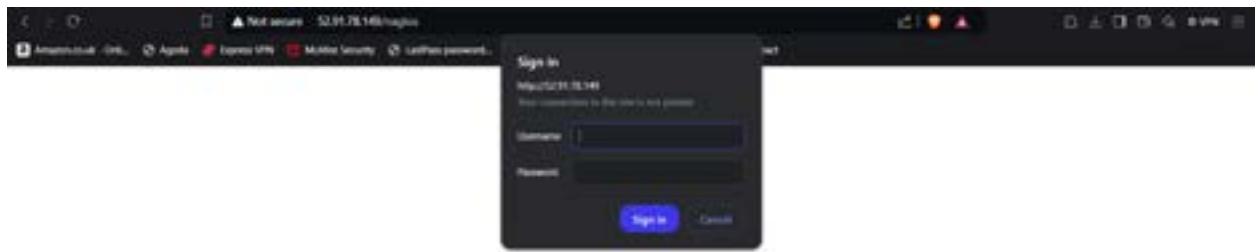
```
[ec2-user@ip-172-31-84-149 ~]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
   Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
   Active: active (running) since Sat 2024-09-28 09:51:48 UTC; 43s ago
     Docs: https://www.nagios.org/documentation
 Process: 67663 ExecStart=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
 Process: 67664 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
 Main PID: 67665 (nagios)
   Tasks: 6 (limit: 4658)
     Memory: 5.8M
        CPU: 87ms
      CGroup: /system.slice/nagios.service
              └─67665 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
                  ├─67666 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                  ├─67667 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                  ├─67668 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                  ├─67669 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                  └─67670 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 28 09:51:48 ip-172-31-84-149.ec2.internal nagios[67665]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successful
Sep 28 09:51:48 ip-172-31-84-149.ec2.internal nagios[67665]: qh: core query handler registered
Sep 28 09:51:48 ip-172-31-84-149.ec2.internal nagios[67665]: qh: echo service query handler registered
Sep 28 09:51:48 ip-172-31-84-149.ec2.internal nagios[67665]: qh: help for the query handler registered
Sep 28 09:51:48 ip-172-31-84-149.ec2.internal nagios[67665]: mproc: Successfully registered manager as 8mproc with quer...
Sep 28 09:51:48 ip-172-31-84-149.ec2.internal nagios[67665]: mproc: Registry request: name=Core Worker 67669;pid=67669
Sep 28 09:51:48 ip-172-31-84-149.ec2.internal nagios[67665]: mproc: Registry request: name=Core Worker 67666;pid=67666
Sep 28 09:51:48 ip-172-31-84-149.ec2.internal nagios[67665]: mproc: Registry request: name=Core Worker 67667;pid=67667
Sep 28 09:51:48 ip-172-31-84-149.ec2.internal nagios[67665]: mproc: Registry request: name=Core Worker 67668;pid=67668
Sep 28 09:51:49 ip-172-31-84-149.ec2.internal nagios[67665]: Successfully launched command file worker with pid 67670
[ec2-user@ip-172-31-84-149 ~]$
```

Step 28: Go back to EC2 Console and copy the Public IP address of this instance. Open up your browser and look for http://<your_public_ip_address>/nagios



[http://52.91.78.149/nagios.](http://52.91.78.149/nagios)



Enter **username as nagiosadmin** and **password as Kingmaker**.

Step 29: After entering the correct credentials, you will see this page



CONCLUSION:

In this experiment, we have learned how to install and configure Nagios Core, Nagios Plugins, and NRPE on a Linux machine. We used an Amazon Linux OS instance with the necessary security rules in place. It's important to ensure that the links for Nagios Core and Nagios Plugins are up to date (when using wget). After extracting and configuring these files, we should check for any issues before starting the server. Once everything is set up, we can start the Nagios server. By using the public IP address of the EC2 instance, we can access the Nagios dashboard by navigating to that IP followed by /nagios.

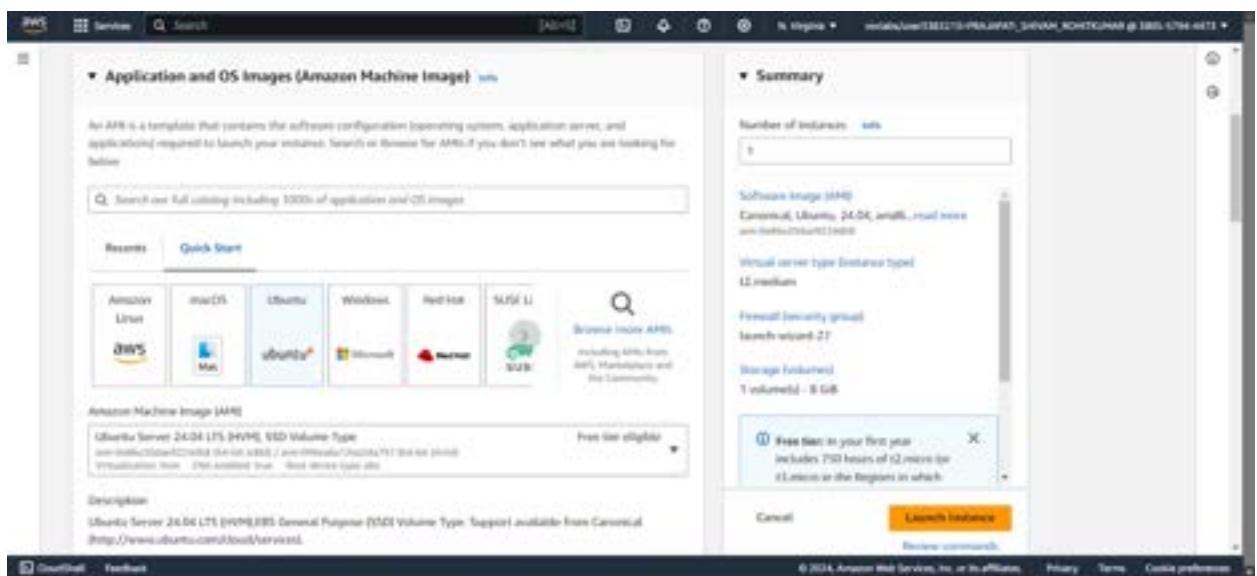
Experiment No:10

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

PREREQUISITES : We should have an Amazon Linux instance with nagios already set up.

Step 1: Set up ubuntu instance

- 1) Log in to your AWS account. Look for EC2 in the services menu. Open the interface and select Create Instance.



Select The OS Image as Ubuntu.

- 2) Ensure that you choose the same private key you created for the Amazon Linux instance. Additionally, select the same security group that you configured for the Linux instance.

The screenshots show the AWS CloudFormation console interface for creating a new instance. Both screenshots include a summary panel on the right with the following details:

- Number of instances:** 1
- Software image (AMI):** Canonical, Ubuntu, 24.04, amd64, root device: /dev/sda1 (ami-0f1aef0c0246aef22)
- Virtual server type (instance type):** t2.medium
- Firewall (security group):** Launch-wizard-23
- Storage (volume):** 1 volume(s) - 8 GiB
- Note:** Free tier in your first year includes 750 hours of t2.micro for t2.micro in the Regions in which you launch.

In the top screenshot, the 'Instance type' section is shown with 't2.medium' selected. In the bottom screenshot, the 'Network settings' section is shown with the 'Select existing security group' dropdown set to 'Launch-wizard-23'.

Instance is:



- 3) Now return to the instances screen. Click on the instance ID of your instance, then select Connect. Click on SSH client and copy the example command. Next, we need to connect our local OS terminal to the instance using SSH. To do this, open the terminal where the private key file (.pem) is stored. Paste the copied SSH command and execute it.

EC2 > Instances > i-0d74a72c6a0429781 > Connect to instance

Connect to instance Info

Connect to your instance i-0d74a72c6a0429781 (nagios_client_41Lab10) using any of these options:

- [EC2 Instance Connect](#)
- [Session Manager](#)
- [SSH client](#) Selected
- [EC2 serial console](#)

Instance ID:
 i-0d74a72c6a0429781 (nagios_client_41Lab10)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is Lab9_41.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
 chmod 400 "Lab9_41.pem"
4. Connect to your instance using its Public DNS:
 ec2-3-86-39-170.compute-1.amazonaws.com

Example:
 ssh -i "Lab9_41.pem" ubuntu@ec2-3-86-39-170.compute-1.amazonaws.com

i 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.

Copy the example command

```
ubuntu@ip-172-31-86-92: ~
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\praja\OneDrive\Desktop\NewFolder> ssh -i "Lab9_41.pem" ubuntu@ec2-3-86-39-170.compute-1.amazonaws.com
The authenticity of host 'ec2-3-86-39-170.compute-1.amazonaws.com (3.86.39.170)' can't be established.
ED25519 key fingerprint is SHA256:JPMeH3iHhSxolnMo0981B3xhjC8bD9+I1MuUgbyF4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-86-39-170.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 Sat Sep 28 11:36:50 UTC 2024

System load: 0.8          Processes:           112
Usage of /: 22.7% of 6.71GB   Users logged in:     0
Memory usage: 5%           IPv4 address for enx0: 172.31.86.92
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
```

Successfully connected the instance via SSH

Step 2: On Nagios Host machine (Linux) execute the following which we have already created as a prerequisites:

- 1) We need to verify whether the nagios service is running or not. For that, run this command : **ps -ef | grep nagios**

```
[ec2-user@ip-172-31-84-149 ~]$ ps -ef | grep nagios
nagios  67665     1  0 09:51 ?    00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios  67666  67665  0 09:51 ?    00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.oh
nagios  67667  67665  0 09:51 ?    00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.oh
nagios  67668  67665  0 09:51 ?    00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.oh
nagios  67669  67665  0 09:51 ?    00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.oh
nagios  67670  67665  0 09:51 ?    00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.oh
[ec2-user  78887   3276  0 10:48 pts/0    00:00:00 grep --color=auto nagios
[ec2-user@ip-172-31-84-149 ~]$ |
```

- 2) Next, switch to the root user and create a directory at the path '/usr/local/nagios/etc/objects/monitorhosts/linuxhosts'.

sudo su

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

```
[ec2-user@ip-172-31-84-149 ~]$ sudo su
mkdir /usr/local/nagios/etc/objects/monitorhosts
mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-84-149 ec2-user]# |
```

- 3) We need to create a configuration file in this directory. To do this, copy the contents of the existing localhost configuration into the new file named 'linuxserver.cfg'.

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

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

```
[root@ip-172-31-84-149 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
cp: cannot create regular file '/usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg': No such file or directory
```

So make the second directory again and run the cp command

```
[root@ip-172-31-84-149 ec2-user]# mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-84-149 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
[root@ip-172-31-84-149 ec2-user]# |
```

We need to make some changes in this config file. Open it using a nano editor.

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

```
[root@ip-172-31-84-149 ec2-user]# nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
```

Change **hostname** and **alias** to **linuxserver**. Change address to **public ip address of client instance** (Ubuntu instance)

```
# Define a host for the local machine

define host {

    use             linux-server

    host_name       linuxserver
    alias           linuxserver
    address         3.86.39.170
}
```

Change hostgroup_name to **linux-servers1**

```
# Define an optional hostgroup for Linux machines

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

#####
# SERVICE DEFINITIONS
#####
#####
```

Change the **occurrences of hostname** further in the document from **localhost** to **linuxserver**

Now, we need to edit the nagios configuration file to add this directory. Run this command

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

```
[root@ip-172-31-84-149 ec2-user]# nano /usr/local/nagios/etc/nagios.cfg
```

and add the following line **cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/**

```

nano 3.0 -w
# host groups, contacts, contact groups, services, etc.
# You can split your object definitions across several config files
# (if you wish to do so below), or keep them all in a single config file.

# You can specify individual object config files as shown below:
cfg_file/usr/local/nagios/etc/objects/commands.cfg
cfg_file/usr/local/nagios/etc/objects/contacts.cfg
cfg_file/usr/local/nagios/etc/objects/hostgroups.cfg
cfg_file/usr/local/nagios/etc/objects/templates.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/windows.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/objects/monitors/
cfg_dir/usr/local/nagios/etc/objects/monitors

# OBJECT CACHE FILE
# This option determines where object definitions are cached after
# they have been read from disk.

[[ Help Exit ] [ Write Out ] [ Read File ] [ Where Is ] [ Replace ] [ Cut Paste ] [ Execute Justify ] [ Go To Line ] [ Undo Redo ] [ Set Mark ] [ Copy ] [ To Bracket ] [ Where Was ]]
```

Now we verify the configuration files. **/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg**

```

[root@ip-172-31-84-149 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 8 service groups.
  Checked 1 contacts.
  Checked 1 contact groups.
```

```

Checking objects...
    Checked 16 services.
    Checked 2 hosts.
    Checked 2 host groups.
    Checked 0 service groups.
    Checked 1 contacts.
    Checked 1 contact groups.
    Checked 24 commands.
    Checked 5 time periods.
    Checked 0 host escalations.
    Checked 0 service escalations.
Checking for circular paths...
    Checked 2 hosts
    Checked 0 service dependencies
    Checked 0 host dependencies
    Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors: 0

Things look okay - No serious problems were detected during the pre-flight check
[root@ip-172-31-84-149 ec2-user]#

```

Once the files are verified, we need to restart the server: **service nagios restart**

```

Things look okay - No serious problems were detected during
[root@ip-172-31-84-149 ec2-user]# service nagios restart
Redirecting to /bin/systemctl restart nagios.service
[root@ip-172-31-84-149 ec2-user]# |

```

```

[root@ip-172-31-84-149 ec2-user]# service nagios restart
Redirecting to /bin/systemctl restart nagios.service
[root@ip-172-31-84-149 ec2-user]# sudo systemctl status nagios
* nagios.service - Nagios Core 4.5.3
   Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
   Active: active (running) since Sat 2024-09-28 11:38:35 UTC; 3min 55s ago
     Docs: https://www.nagios.org/documentation
  Process: 73417 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (codenamed, status=0/SUCCESS)
  Process: 73418 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (codenamed, status=0/SUCCESS)
 Main PID: 73419 (nagios)
   Tasks: 6 (limit: 4888)
    Memory: 4.2M
      CPU: 11ms
     CGroup: /system.slice/nagios.service
             ├─73419 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─73420 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/www/nagios
             ├─73421 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/www/nagios
             ├─73422 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/www/nagios
             ├─73423 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/www/nagios
             └─73425 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 28 11:38:31 ip-172-31-84-149.ec2.internal nagios[73459]: apnrc: Successfully registered manager as apnrc with query handler
Sep 28 11:38:31 ip-172-31-84-149.ec2.internal systemd[1]: Started nagios.service - Nagios Core 4.5.3.
Sep 28 11:38:31 ip-172-31-84-149.ec2.internal nagios[73459]: apnrc: Registry request: nseidCore Worker 73423;pid=77423
Sep 28 11:38:31 ip-172-31-84-149.ec2.internal nagios[73459]: apnrc: Registry request: nseidCore Worker 73421;pid=77421
Sep 28 11:38:31 ip-172-31-84-149.ec2.internal nagios[73459]: apnrc: Registry request: nseidCore Worker 73420;pid=77420
Sep 28 11:38:31 ip-172-31-84-149.ec2.internal nagios[73459]: apnrc: Registry request: nseidCore Worker 73422;pid=77422
Sep 28 11:38:31 ip-172-31-84-149.ec2.internal nagios[73459]: Successfully launched command file worker with pid 73425
Sep 28 11:38:31 ip-172-31-84-149.ec2.internal nagios[73459]: SERVICE ALERT: linuxserver;HTTP;CRITICAL;SOFT;1;connect to address 3.88.39.178 and port 80; Cr
Sep 28 11:38:31 ip-172-31-84-149.ec2.internal nagios[73459]: SERVICE ALERT: linuxserver;HTTP;CRITICAL;SOFT;2;connect to address 3.88.39.178 and port 80; Cr
Sep 28 11:38:31 ip-172-31-84-149.ec2.internal nagios[73459]: SERVICE ALERT: linuxserver;HTTP;CRITICAL;SOFT;3;connect to address 3.88.39.178 and port 80; Cr
1 lines, 1-28/21 (3.02s)
```

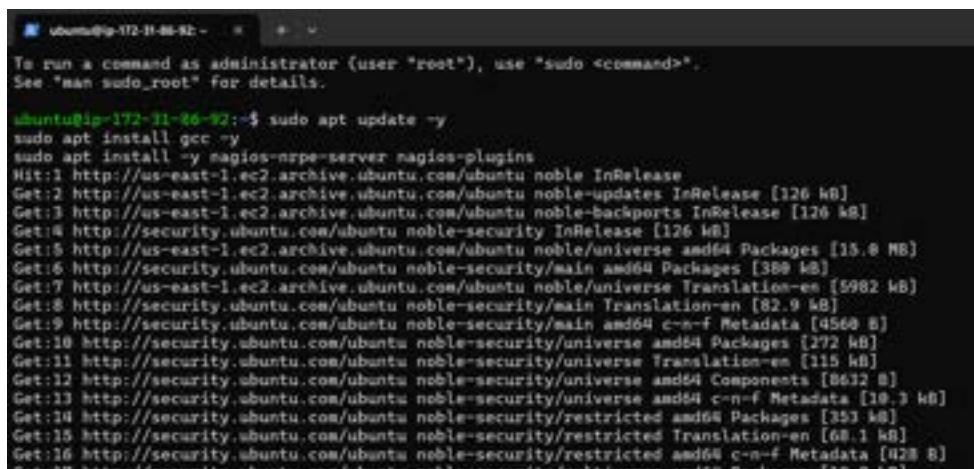
Step 3: Execute the following on Nagios Client machine (Ubuntu)

- 1) First, check for any available updates, and then proceed to install gcc, the Nagios NRPE server, and Nagios plugins.

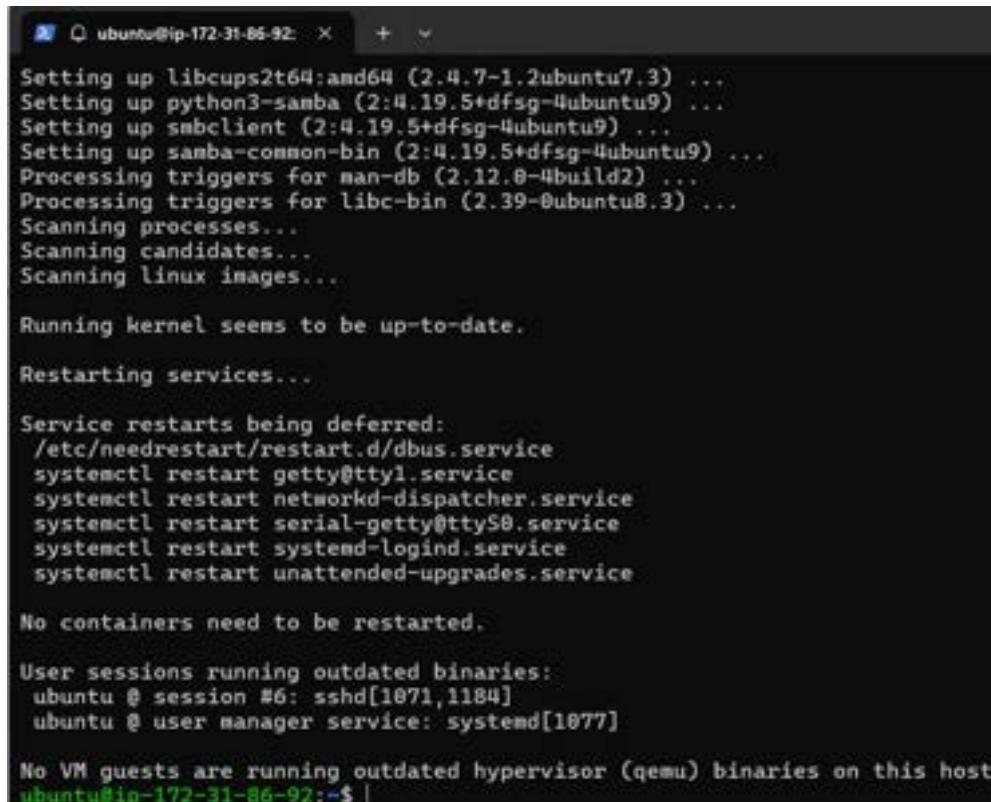
```
sudo apt update -y
```

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

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



```
ubuntu@ip-172-31-86-92:~$ sudo apt update -y
[sudo] password for ubuntu:
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease [126 kB]
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://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [380 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [82.9 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4568 B]
Get:10 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [272 kB]
Get:11 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [315 kB]
Get:12 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8612 B]
Get:13 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.3 kB]
Get:14 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [353 kB]
Get:15 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [68.1 kB]
Get:16 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [828 B]
Get:17 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [10.3 kB]
```



```
Setting up libcurl2t64:amd64 (2.4.7-1.2ubuntu7.3) ...
Setting up python3-samba (2:4.19.5+dfsg-4ubuntu9) ...
Setting up smbcclient (2:4.19.5+dfsg-4ubuntu9) ...
Setting up samba-common-bin (2:4.19.5+dfsg-4ubuntu9) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.3) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...

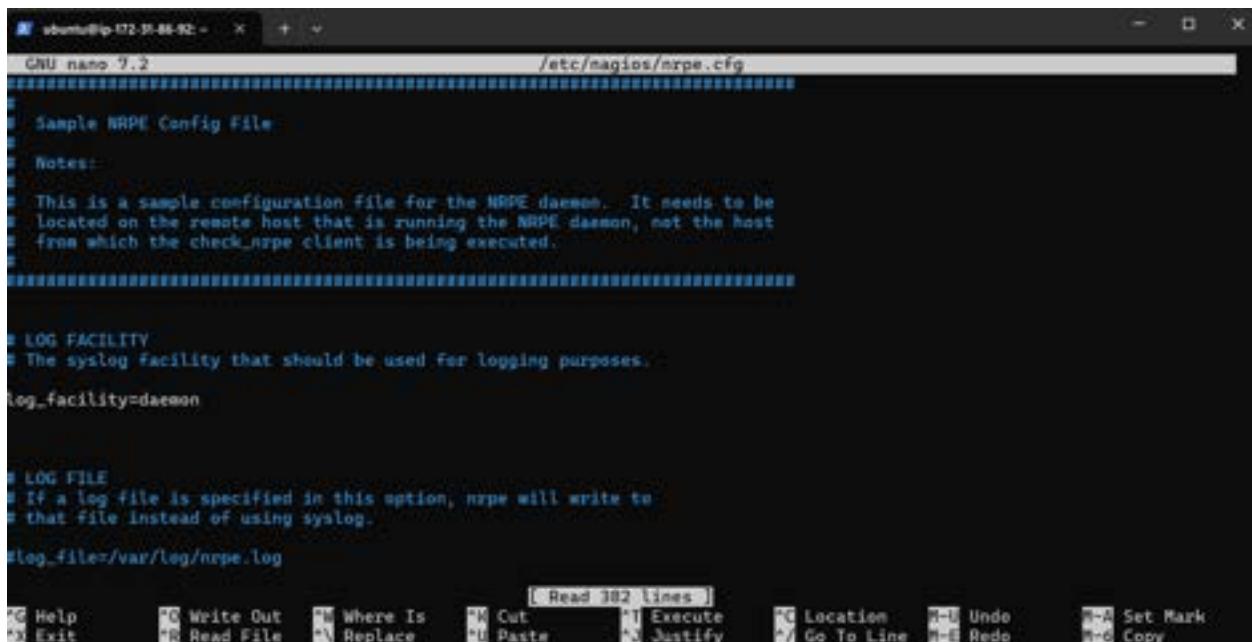
Service restarts being deferred:
/etc/needrestart/restart.d/dbus.service
systemctl restart getty@tty1.service
systemctl restart networkd-dispatcher.service
systemctl restart serial-getty@ttyS0.service
systemctl restart systemd-logind.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

User sessions running outdated binaries:
ubuntu @ session #6: sshd[1071,1184]
ubuntu @ user manager service: systemd[1077]

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-86-92:~$ |
```

- 2) We need to include the public IP address of our Nagios host machine (Linux) in the NRPE configuration file. **sudo nano /etc/nagios/nrpe.cfg**



```

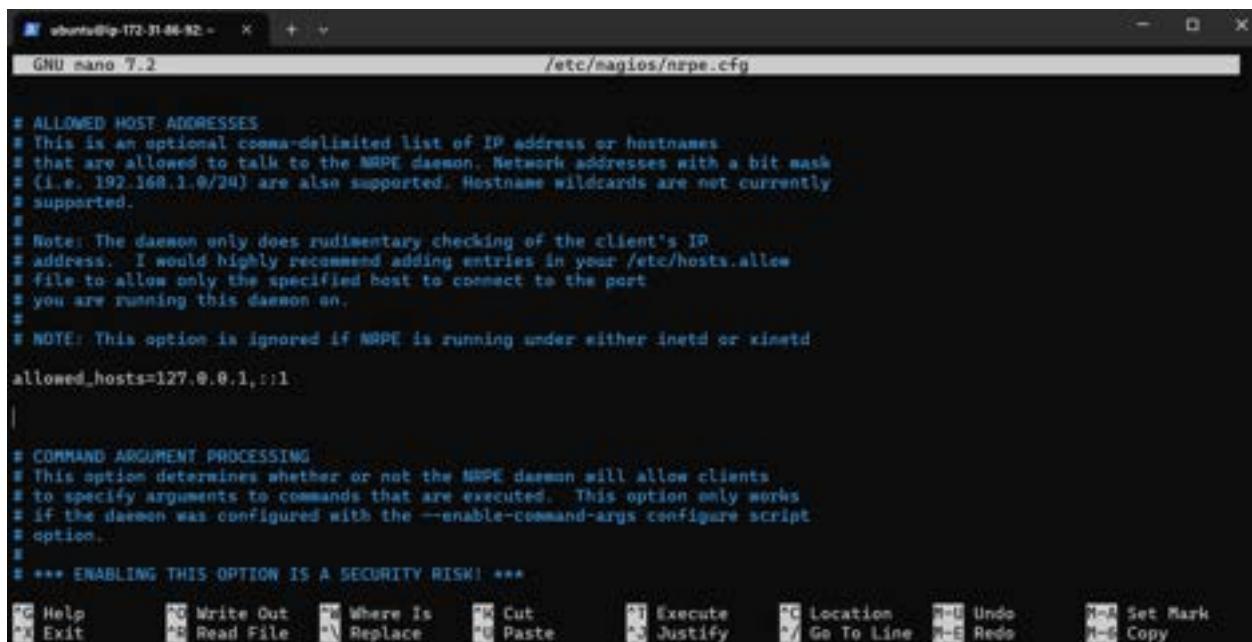
ubuntu@ip-172-31-88-92: ~ % + v
GNU nano 7.2                               /etc/nagios/nrpe.cfg
=====
# Sample NRPE Config File.
#
# Notes:
#
# This is a sample configuration file for the NRPE daemon. It needs to be
# located on the remote host that is running the NRPE daemon, not the host
# from which the check_nrpe client is being executed.
#
# LOG FACILITY
# The syslog facility that should be used for logging purposes.
log_facility=daemon

# LOG FILE
# If a log file is specified in this option, nrpe will write to
# that file instead of using syslog.
log_file=/var/log/nrpe.log

[ Read 302 lines ]
G Help      W Write Out   M Where Is    C Cut        E Execute   L Location   U Undo     S Set Mark
E Exit      R Read File   R Replace    P Paste      J Justify   G Go To Line  D Redo     C Copy

```

Under allowed_hosts, add the nagios host ip address (public)



```

ubuntu@ip-172-31-88-92: ~ % + v
GNU nano 7.2                               /etc/nagios/nrpe.cfg
=====
# ALLOWED HOST ADDRESSES.
# This is an optional comma-delimited list of IP address or hostnames
# that are allowed to talk to the NRPE daemon. Network addresses with a bit mask
# (i.e. 192.168.1.0/24) are also supported. Hostname wildcards are not currently
# supported.
#
# Note: The daemon only does rudimentary checking of the client's IP
# address. I would highly recommend adding entries in your /etc/hosts.allow
# file to allow only the specified host to connect to the port
# you are running this daemon on.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
allowed_hosts=127.0.0.1,::1

#
# COMMAND ARGUMENT PROCESSING
# This option determines whether or not the NRPE daemon will allow clients
# to specify arguments to commands that are executed. This option only works
# if the daemon was configured with the --enable-command-args configure script
# option.
#
# *** ENABLING THIS OPTION IS A SECURITY RISK! ***
[ Read 302 lines ]
G Help      W Write Out   M Where Is    C Cut        E Execute   L Location   U Undo     S Set Mark
E Exit      R Read File   R Replace    P Paste      J Justify   G Go To Line  D Redo     C Copy

```

Step 4: Check the Nagios Dashboard. Go to Nagios dashboard, click on hosts. Here, we can see that the linuxserver is also added as a host.

Click on linuxserver. we can check all the information about linuxserver host.

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

The screenshot shows the Nagios monitoring interface. At the top, there are three status summary boxes: 'Current Network Status' (green), 'Host Status Totals' (green), and 'Service Status Totals' (yellow). Below these is a large table titled 'Service Status Details For All Hosts'. The table has columns for Host, Service, Status, Last Check, Duration, and Attempt. The table lists various services across different hosts, with some entries in yellow indicating a warning state. The right side of the table provides detailed status information for each entry. On the left, there's a sidebar with links for General, Current Status, Services, Problems, Reports, and System. The main content area shows 'Result 1 - 10 of 10 Matching Services'.

CONCLUSION:

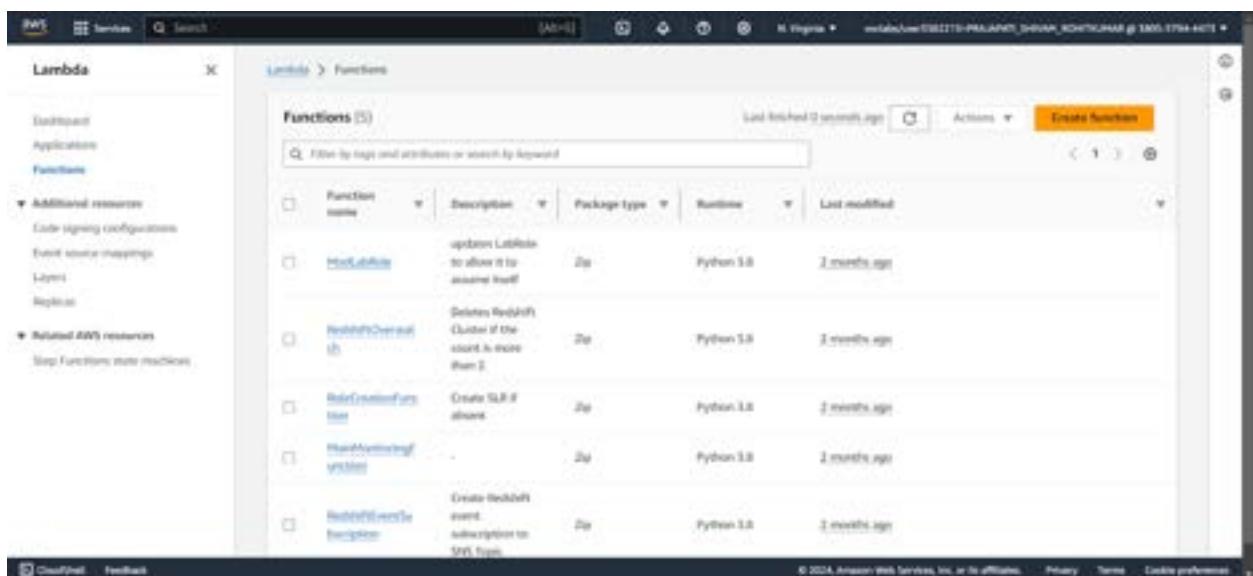
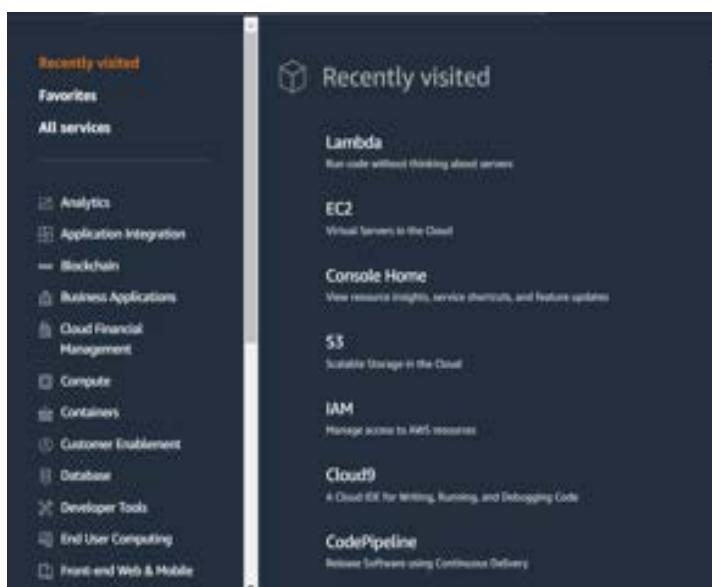
In this experiment, we learned to conduct port service monitoring and server monitoring using Nagios. To do this, we require a Linux instance to host the Nagios dashboard and a separate Ubuntu instance linked as a second host. We need to configure the Linux instance and include the IP address of the Ubuntu instance. Subsequently, we must replicate the initial setup from the Linux instance on the Ubuntu instance by adding the IP address of the Linux instance to the list of allowed hosts. After restarting the NRPE server, we should see the 'linux server' host listed.

Experiment No: 11

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

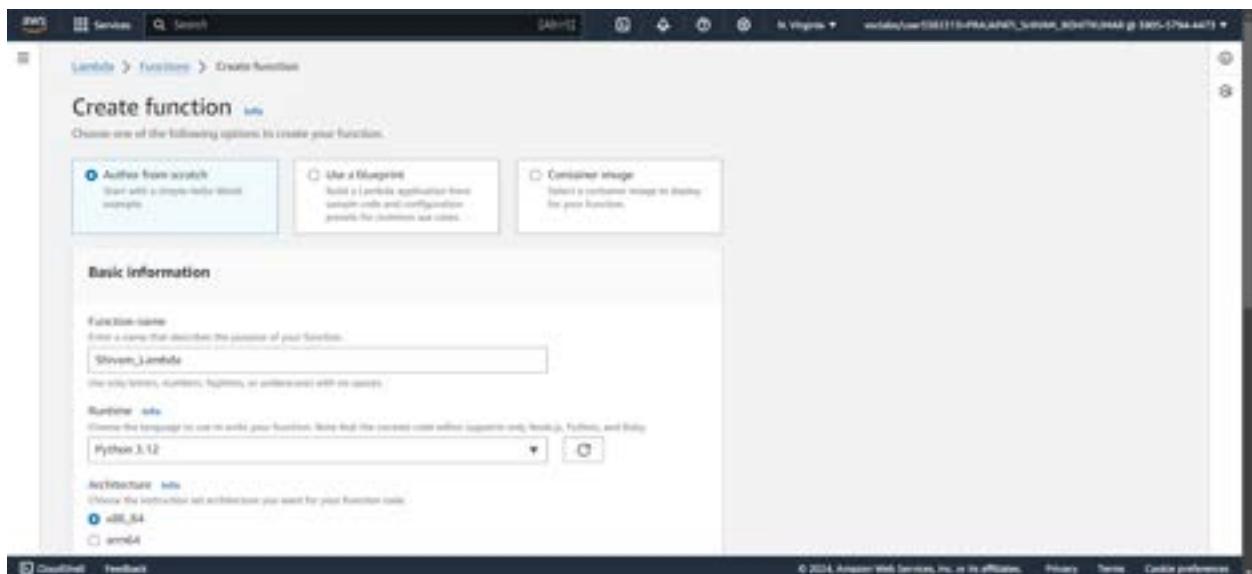
CREATION OF LAMBDA FUNCTION:

Step1: Log in to your AWS Personal or Academy account. Navigate to Lambda, then select the 'Create Function' button



Step 2: Give your Lambda function a name and choose a programming language. The code editor only supports Node.js, Python, and Ruby, so in my case I have chosen **Python 3.12**. Set the **architecture to x86**. For the execution role, select 'Use an existing role,' then pick '**Lab role**' from the dropdown menu under existing roles .

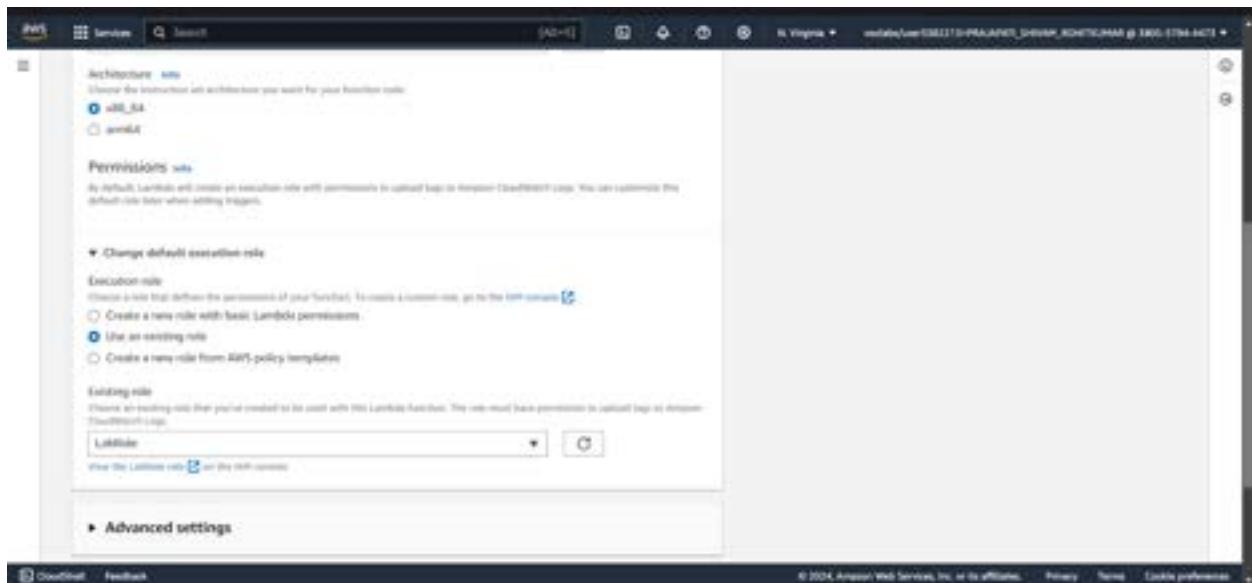
(This is because the Lab role already has the permissions needed for Lambda to run properly, so you don't need to create a new role from scratch. It's a quicker and more convenient option)



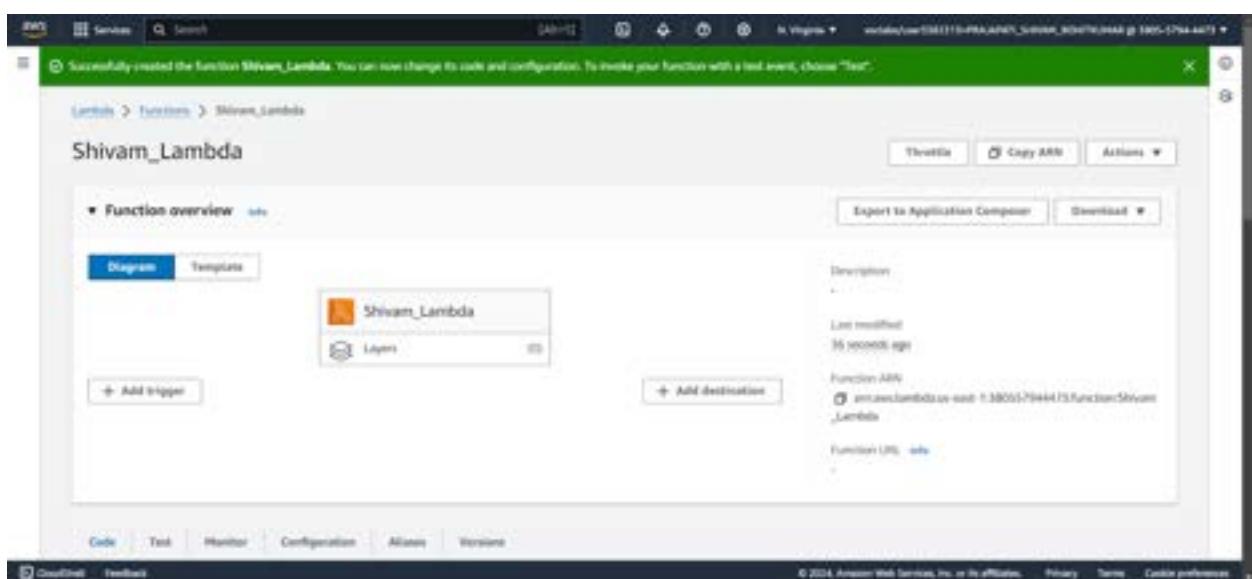
Give the function name and select required language for lambda function



Architecture will be x86_64

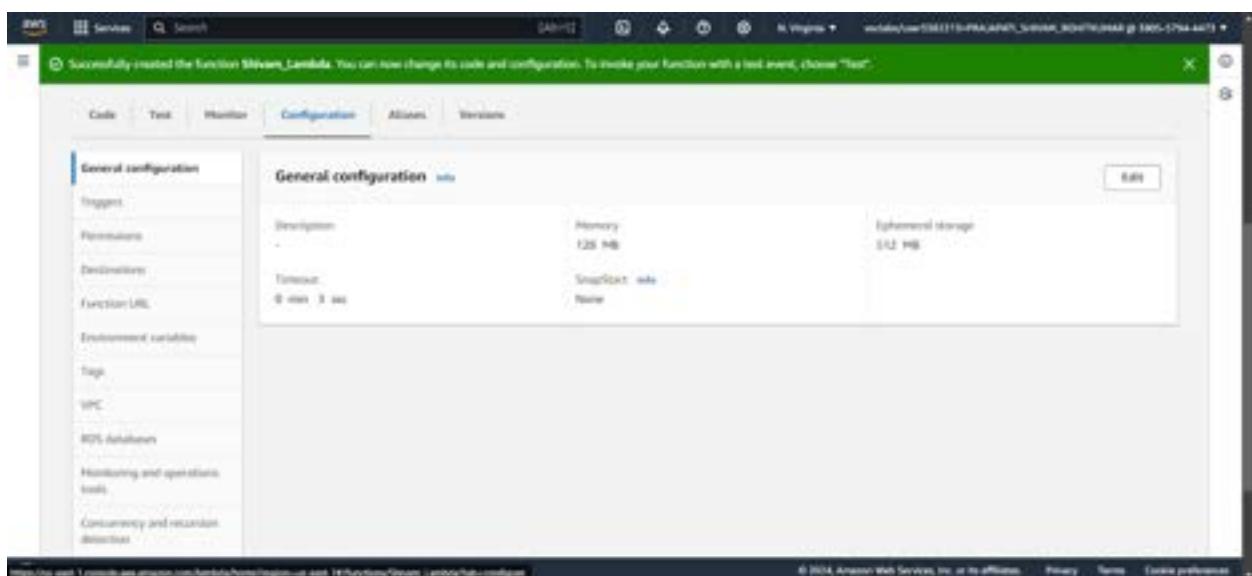


Select proper Execution role

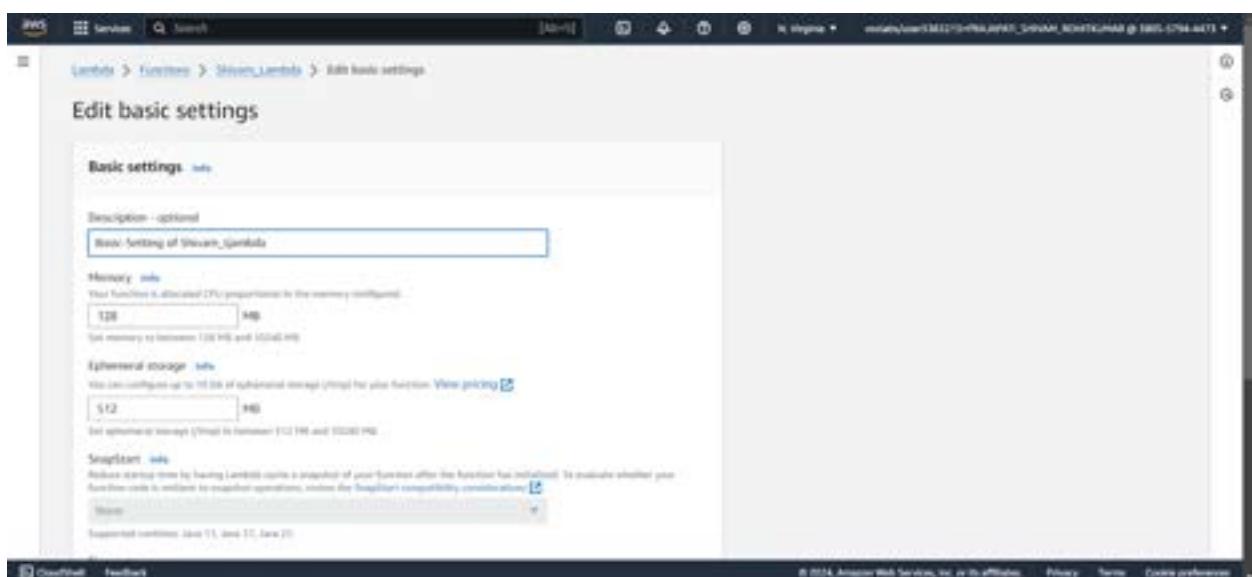


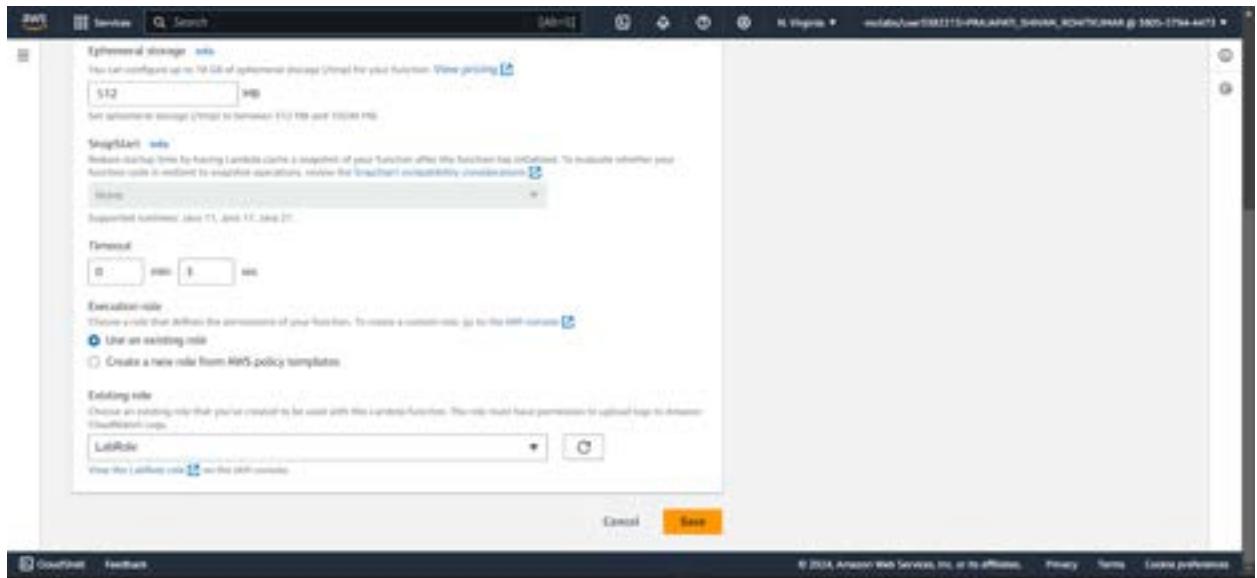
Successfully created Lambda function

Step 3: To view or change the basic settings, go to the 'Configuration' tab and click 'Edit' under 'General settings.' (THIS STEP IS OPTIONAL)



You can add a description and adjust the memory and timeout settings. I've changed the timeout to 1 second, as that's enough for now.

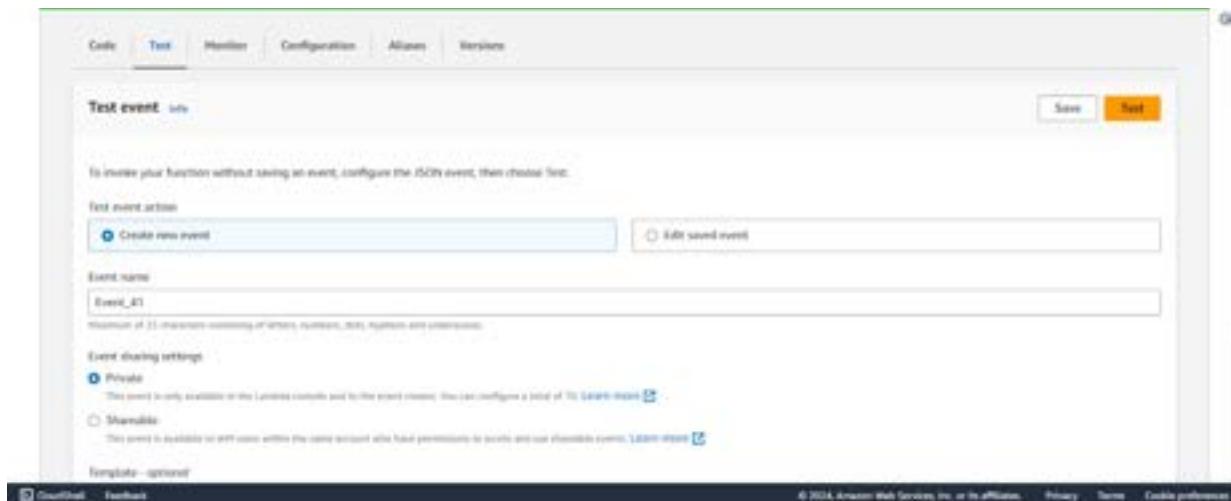


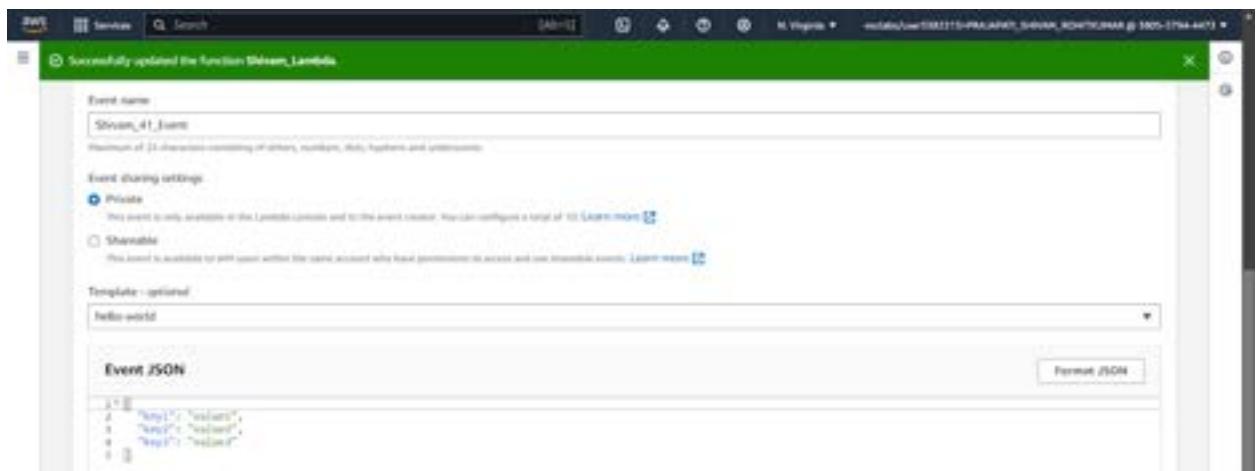


Click on Save

Step 4: Go to the 'Test' tab and click 'Create a new event.' Give the event a name, set 'Event Sharing' to private, and choose the 'hello-world' template.

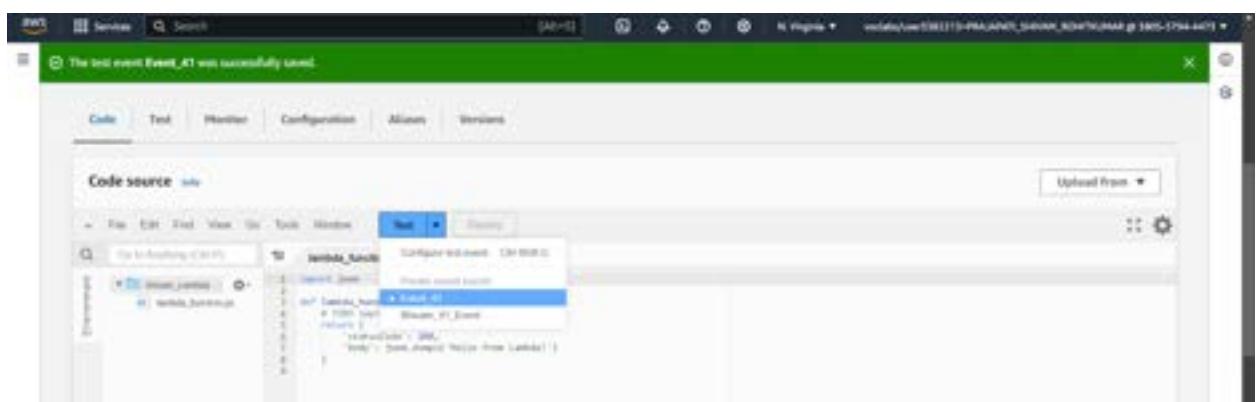
We basically create a new event to test and verify your Lambda function; setting Event Sharing to private keeps it secure and choosing the "hello-world" template provides a simple structure for testing without complex inputs.



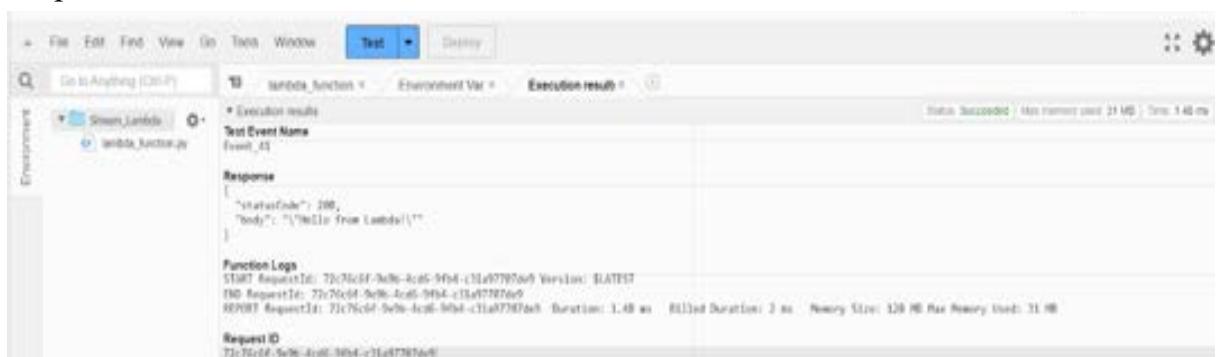


Click on save button above

Step 5: In the Code section, select the event you created from the dropdown menu under 'Test,' then click 'Test.' You should see the output below."

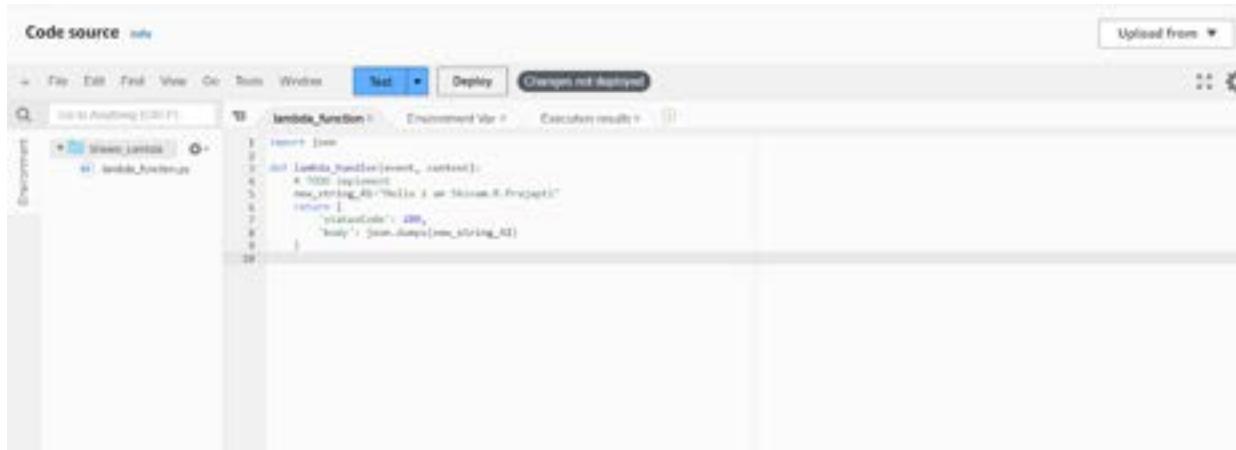


Output :

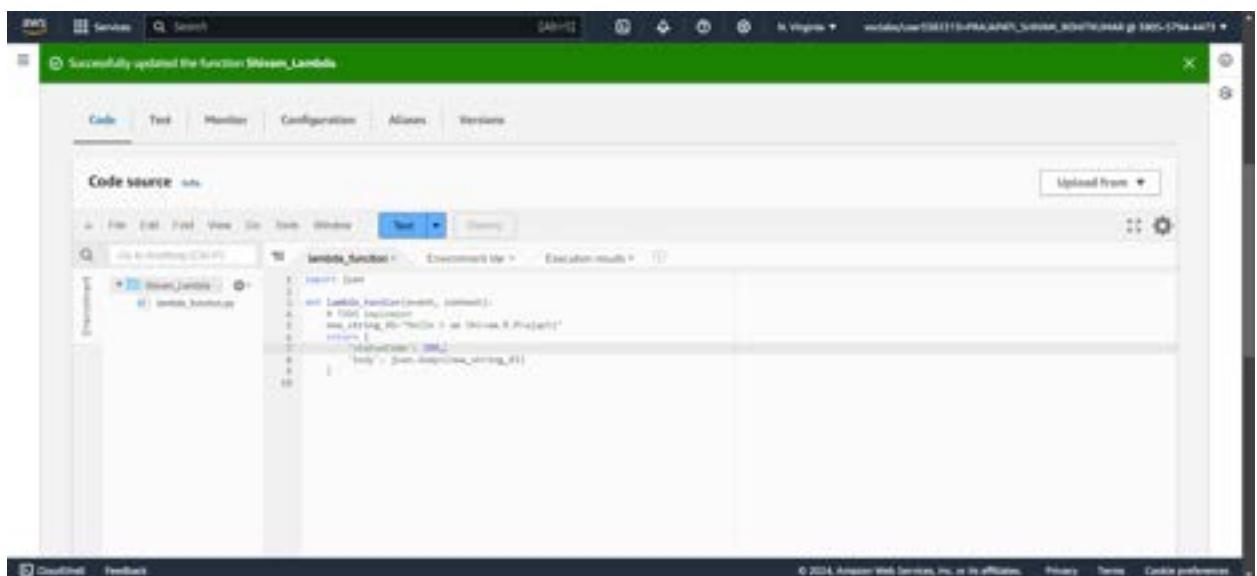


You select the created event to run the specific test you set up, and clicking 'Test' executes your Lambda function to check if it works as expected and produces the desired output.

Step 6: You can edit your lambda function code. I have changed the code to display the new String. After Changing save it by Control + S and click on Deploy . Make sure you have internet connectivity while deploying or else it will show failed deployment



```
Code source info
File Edit Find View Go Window Test Deploy Check for updates Upload from ...
Q lambda_function Environment Var Execution results
lambda_function
1 import json
2
3 def Lambda_handler(event, context):
4     # 1000 payload
5     new_string = "Hello I am Shivam.R.Prajapati"
6     return {
7         "statusCode": 200,
8         "body": json.dumps(new_string)
9     }
10
```



Successfully updated the function: Shivas_Lambda

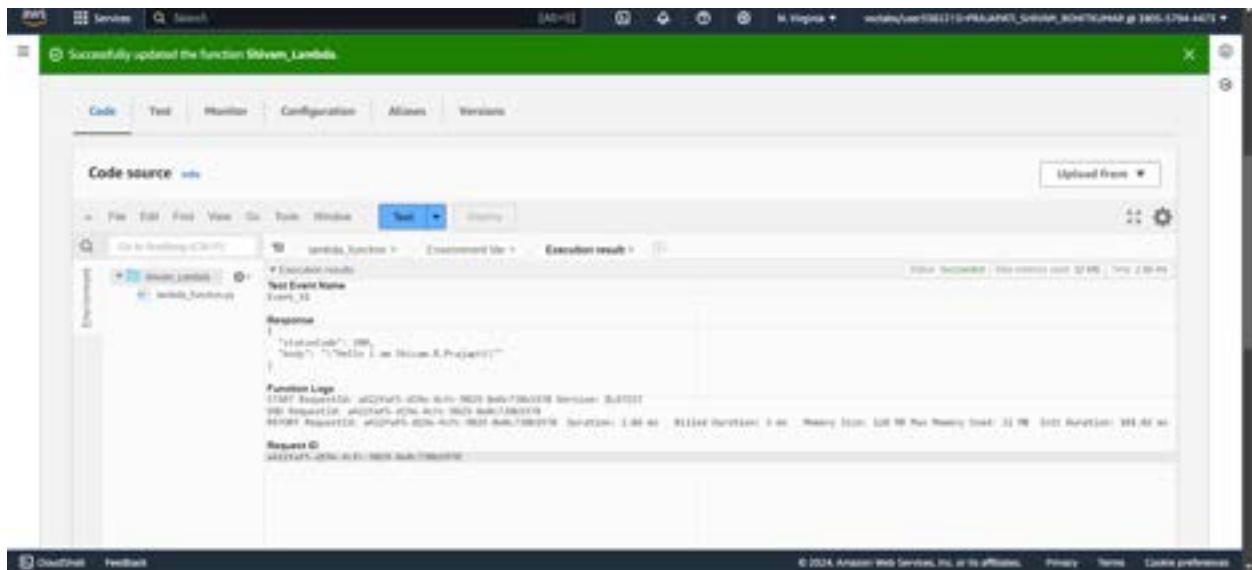
Code Test Monitor Configuration Aliases Versions

Code source [info](#) [Upload from](#) [...](#)

```
Code source info
File Edit Find View Go Window Test Deploy Check for updates Upload from ...
Q lambda_function Environment Var Execution results
lambda_function
1 import json
2
3 def Lambda_handler(event, context):
4     # 1000 payload
5     new_string = "Hello I am Shivam.R.Prajapati"
6     return {
7         "statusCode": 200,
8         "body": json.dumps(new_string)
9     }
10
```

Successfully changed the function.

Step 7: Click on 'Test' to see the output. You'll get a status code of **200** which means "OK" and indicates that the request was successful, your string output, and the function logs, showing that it was deployed successfully.



CONCLUSION:

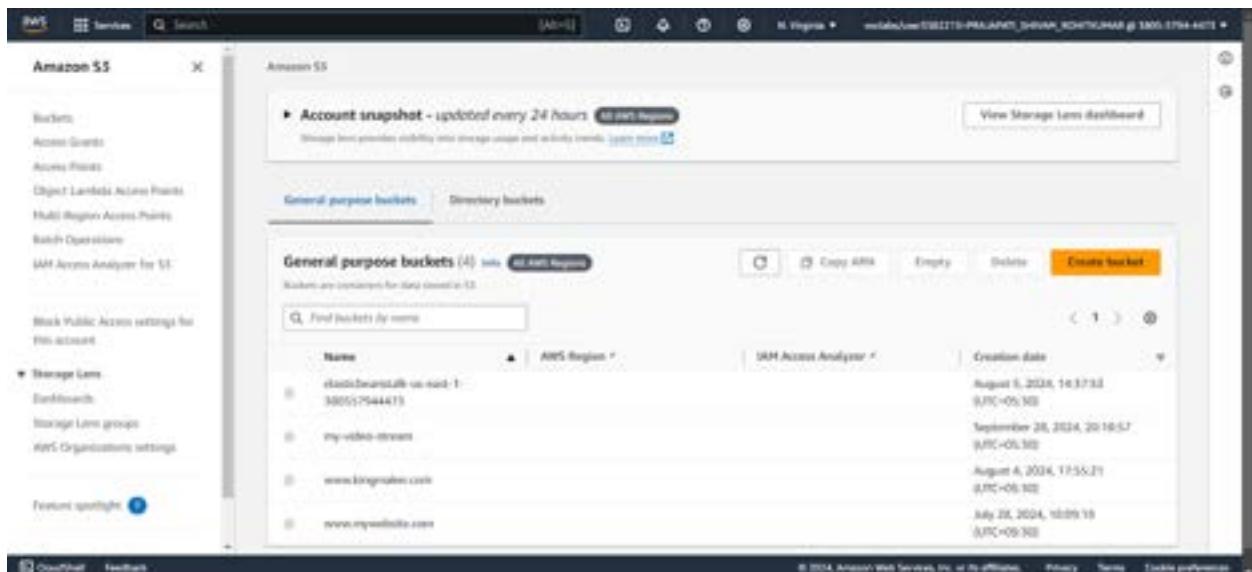
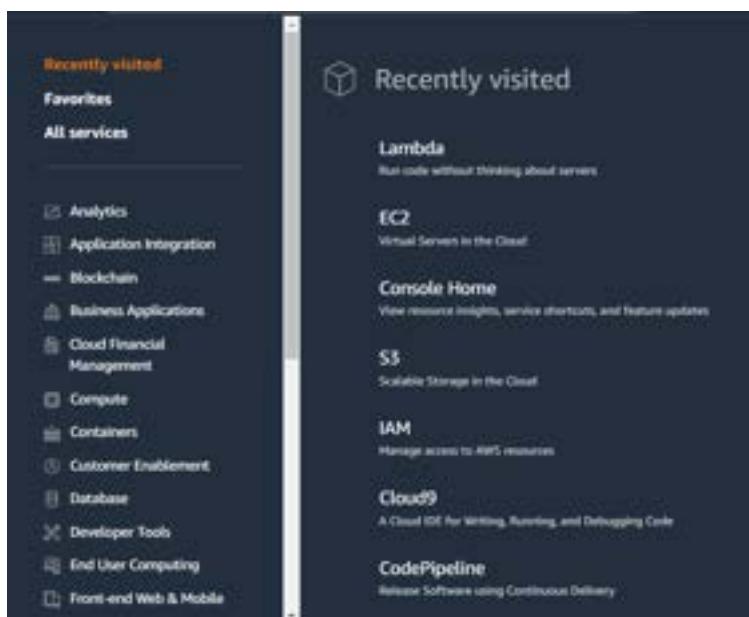
In this experiment, we successfully created an AWS Lambda function and followed the important steps involved. First, we set up the function using Python and adjusted the timeout setting to 1 second. Then, we created a test event to see how the function works and checked the output to ensure it was correct. We also modified the function's code and redeployed it to see the changes in real-time. So Lambda Function allows you to concentrate on writing code while AWS manages the infrastructure and automatically scales the service as needed. This makes it easier to develop and run applications without worrying about server management.

Experiment No :12

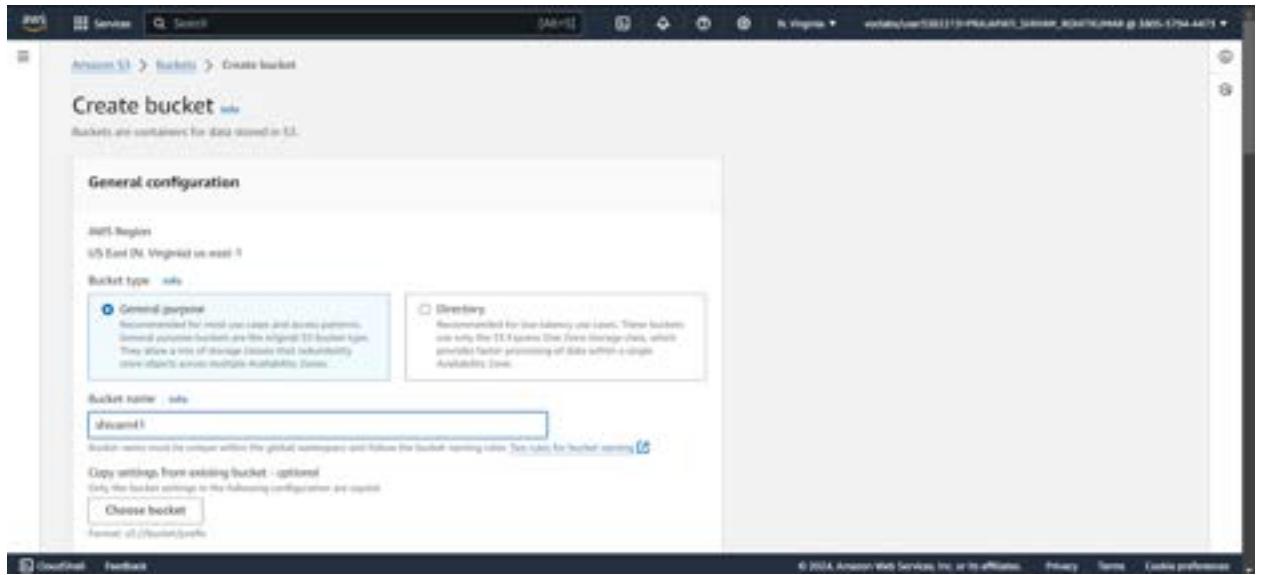
AIM : To create a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3

CREATING LAMBDA FUNCTION :

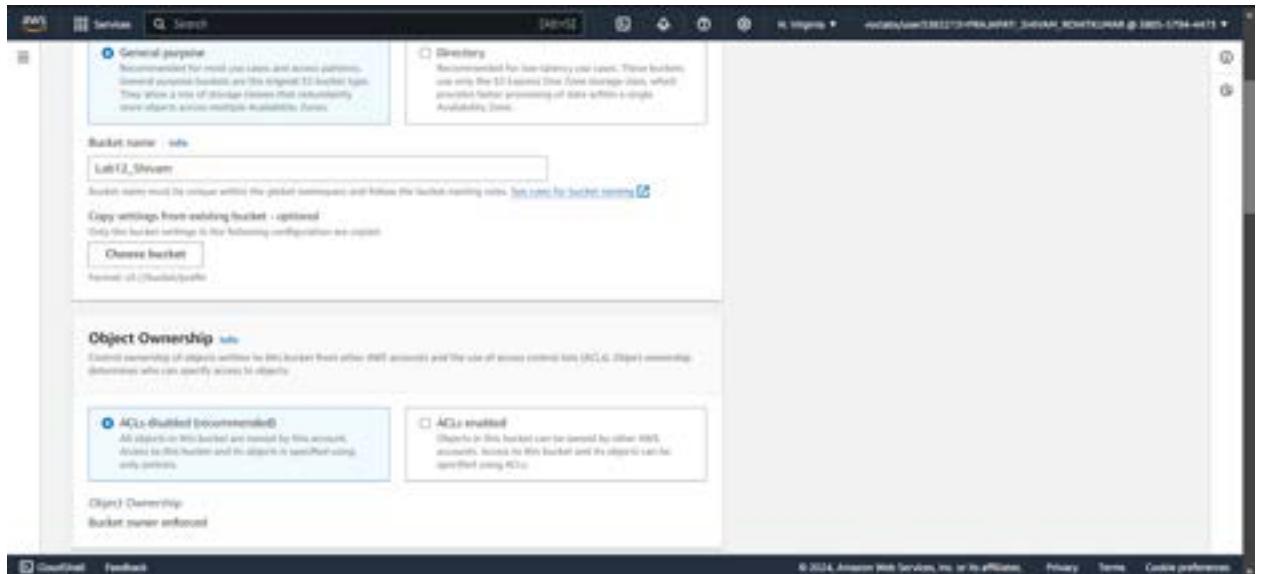
Step 1: Log in to your AWS Personal account. Then go to S3 in the services menu and click on "Create S3 Bucket."



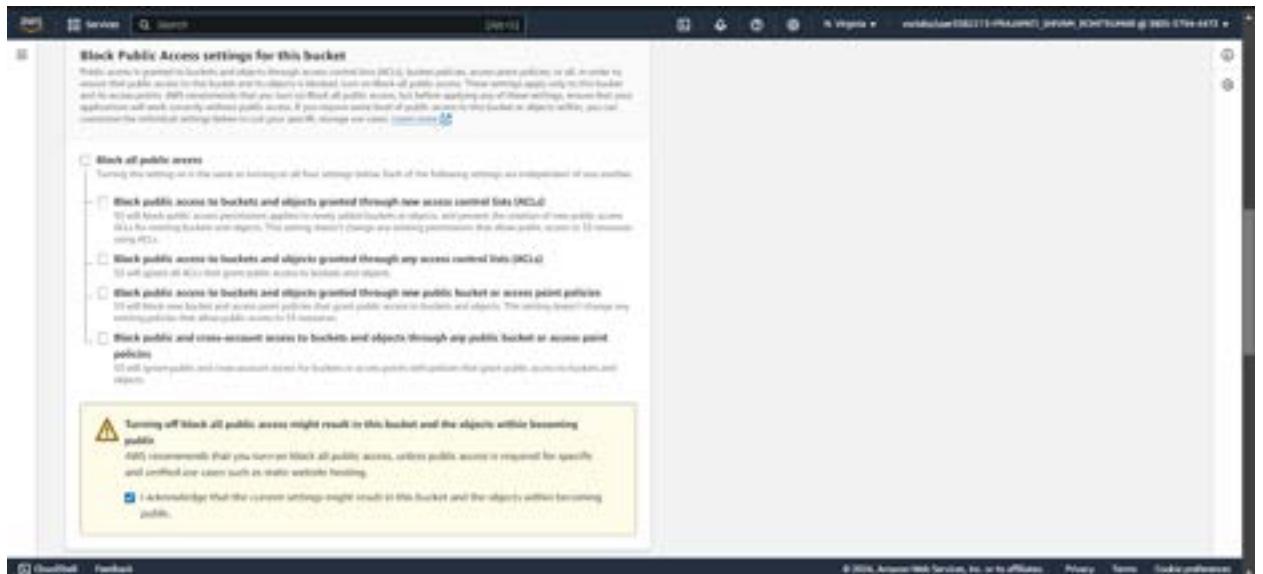
Step 2: Give your bucket a name, select "General purpose project," then uncheck "Block public access." Keep the other settings as they are.



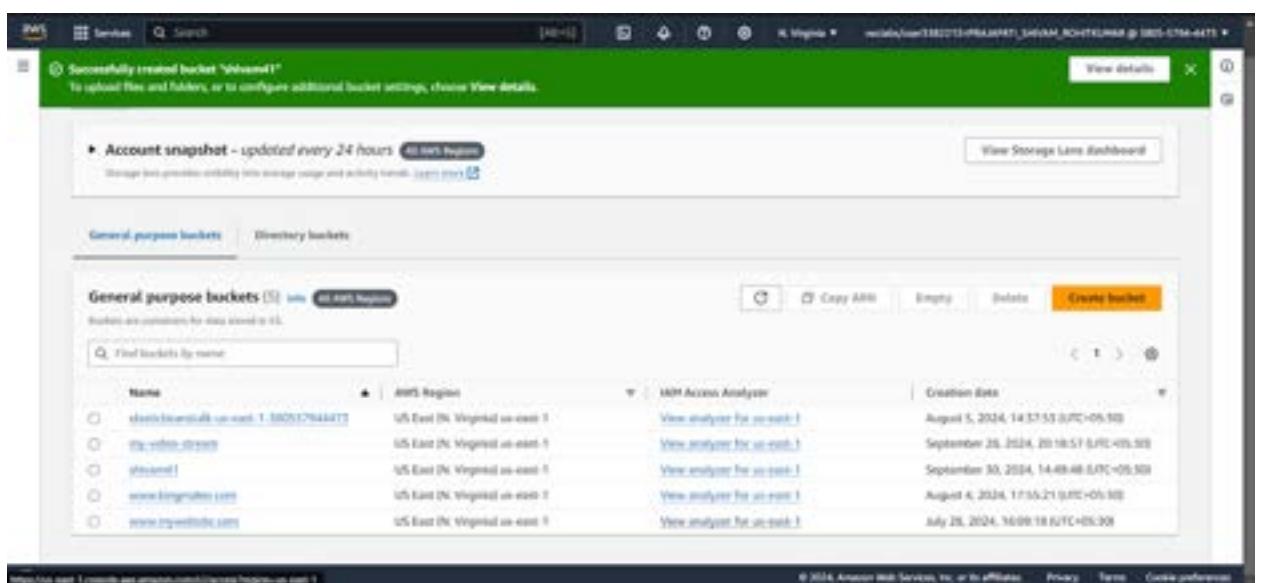
Given proper bucket Name



Selected Appropriate object Ownership

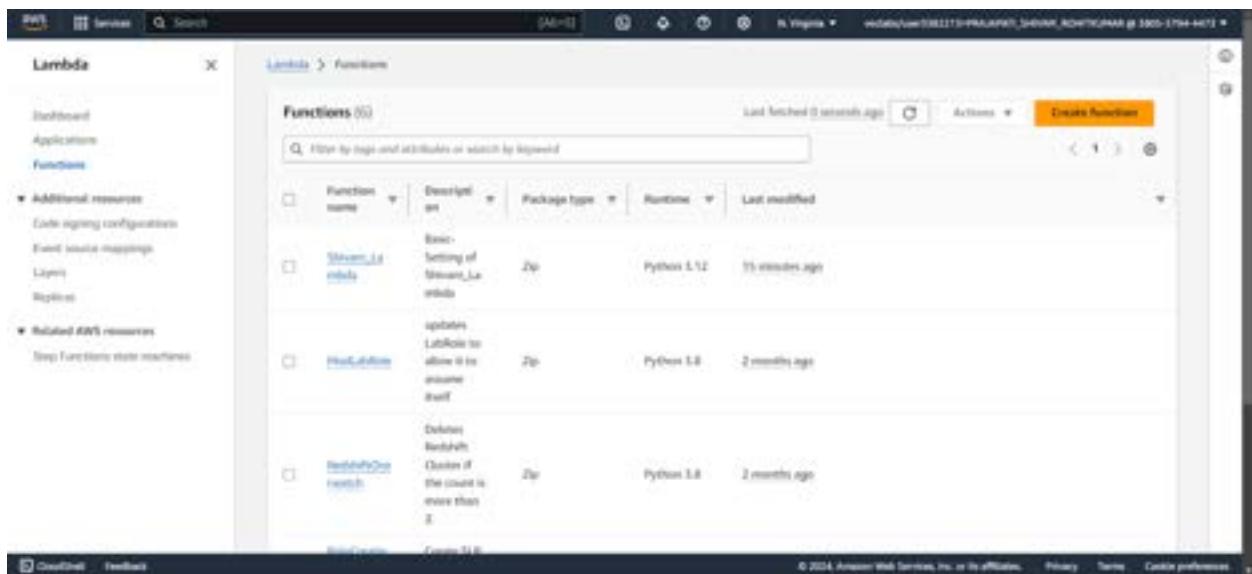
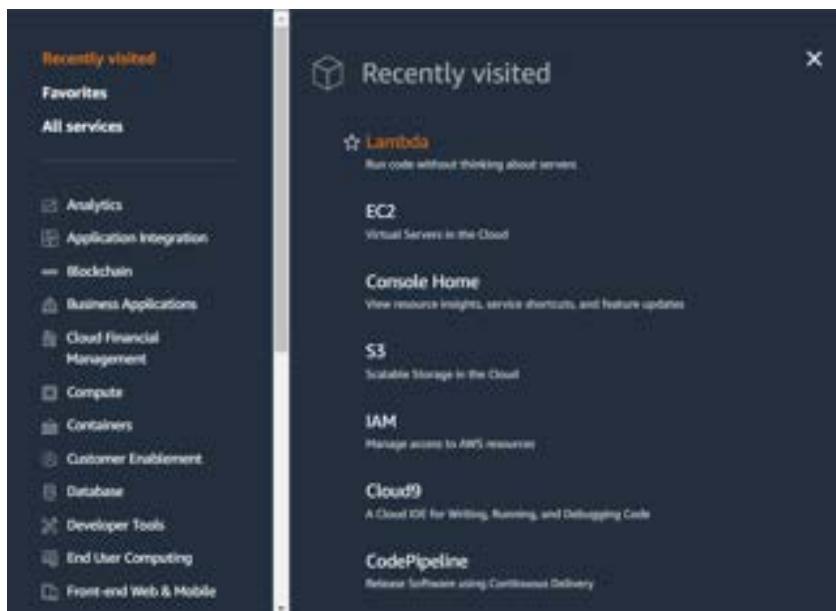


Unchecked the block all public access checkbox and checked the lower checkbox.



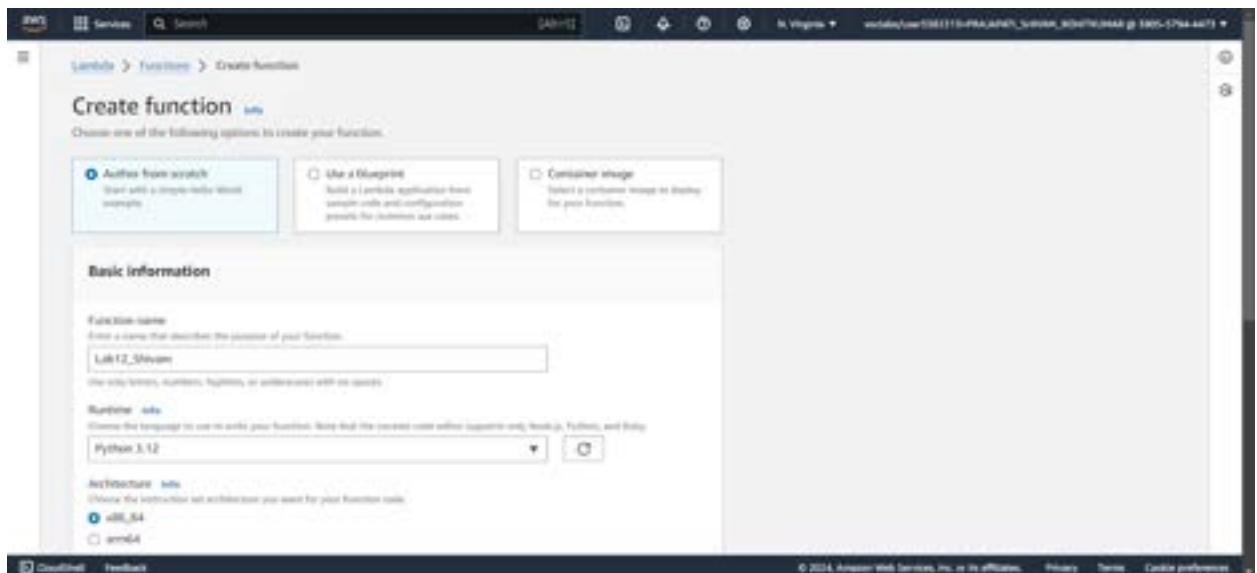
Successfully created the bucket

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

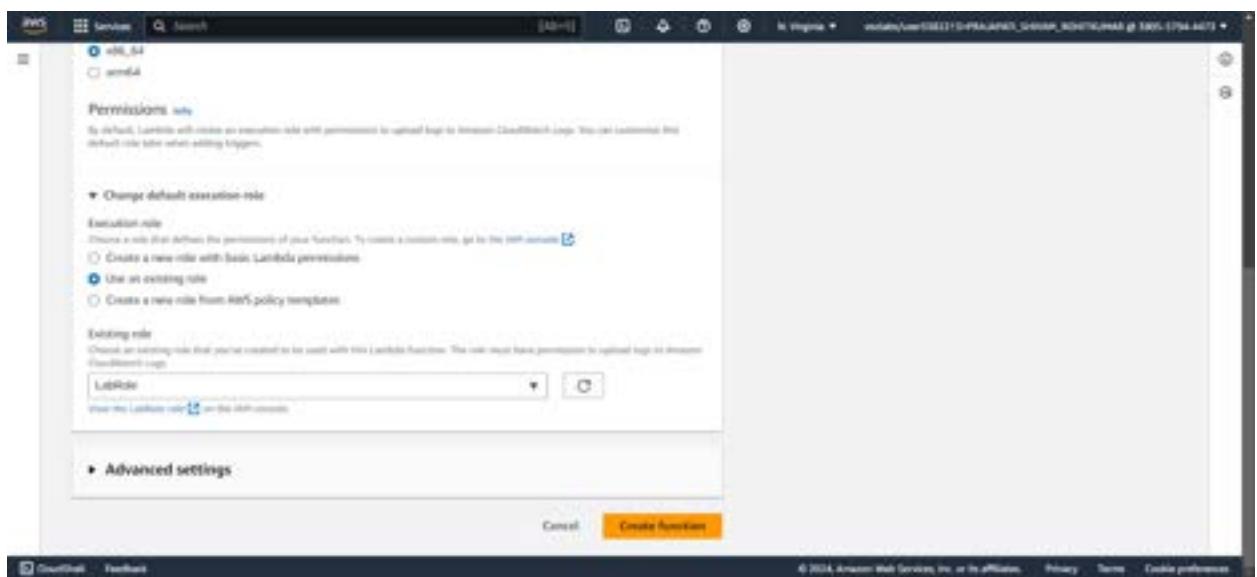


Step 4: Give your Lambda function a name and choose a programming language. The code editor only supports Node.js, Python, and Ruby, so in my case I have chosen **Python 3.12**. Set the **architecture to x86**. For the execution role, select '**Use an existing role**', then pick '**Lab role**' from the dropdown menu under existing roles .

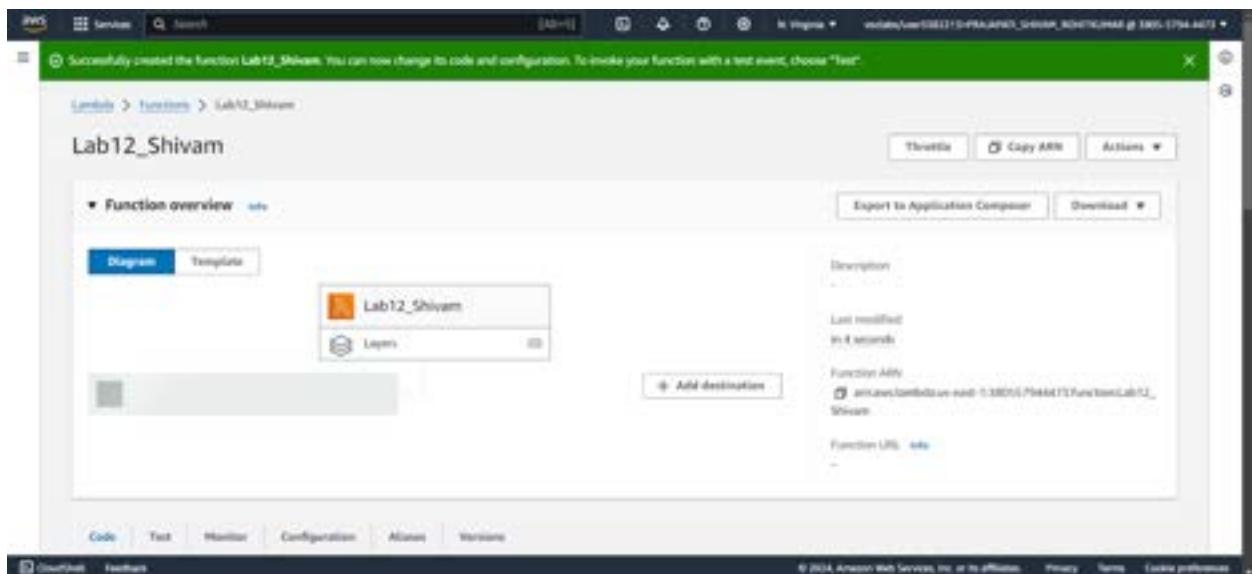
(This is because the Lab role already has the permissions needed for Lambda to run properly, so you don't need to create a new role from scratch. It's a quicker and more convenient option)



Given proper function name and selected language for Lambda function

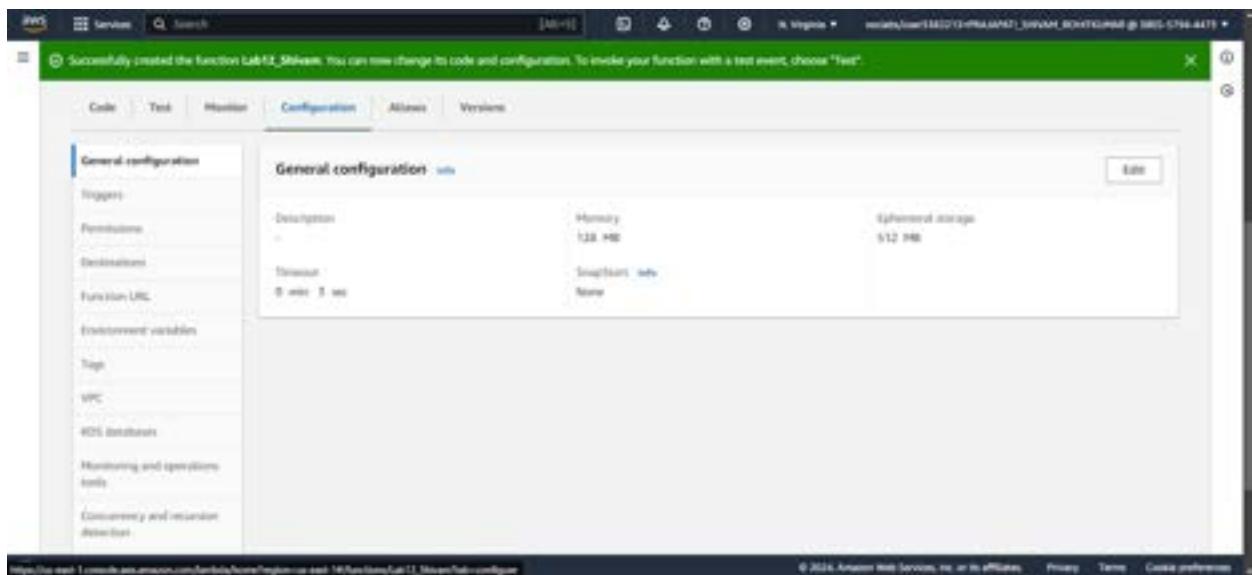


Selected Appropriate Execution role

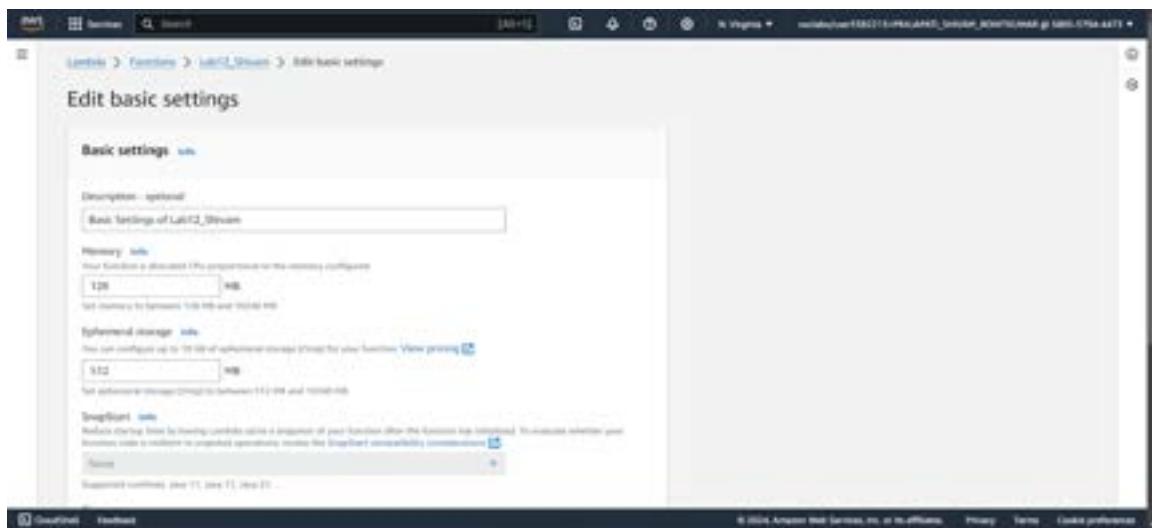


Successfully created the Lambda function.

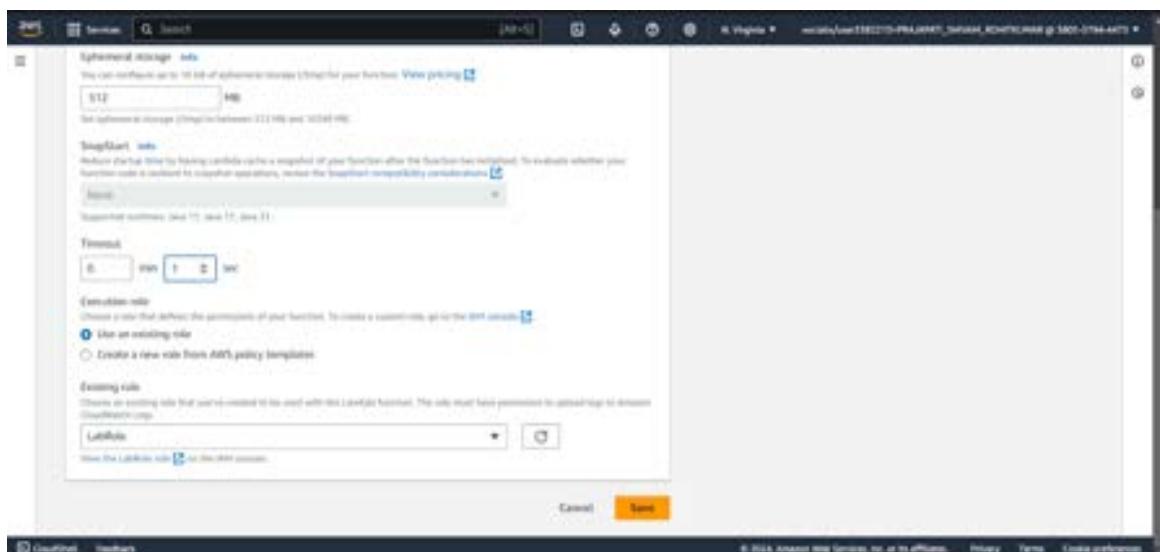
Step 5: To view or change the basic settings, go to the 'Configuration' tab and click 'Edit' under 'General settings.' (THIS STEP IS OPTIONAL)



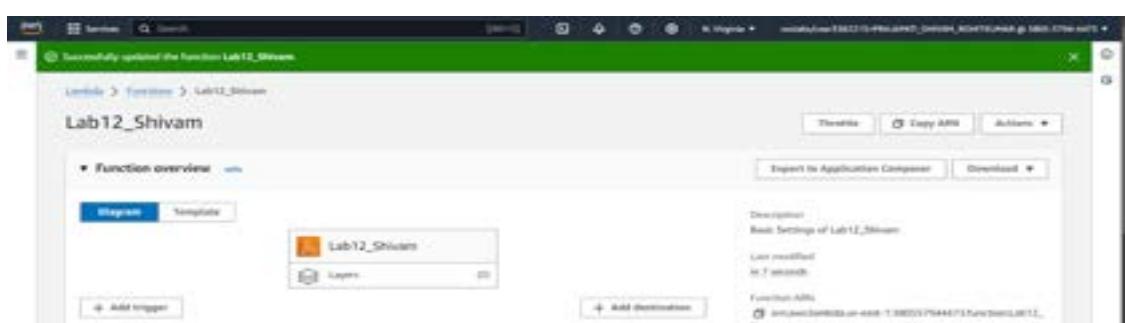
You can add a description and adjust the memory and timeout settings. I've changed the timeout to 1 second, as that's enough for now.



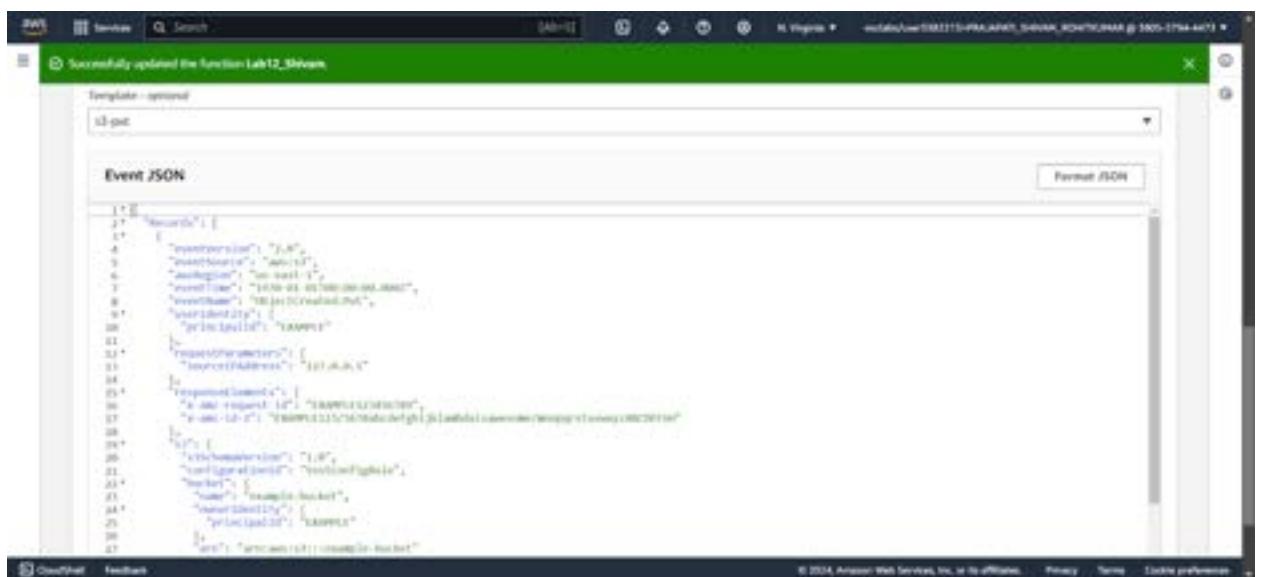
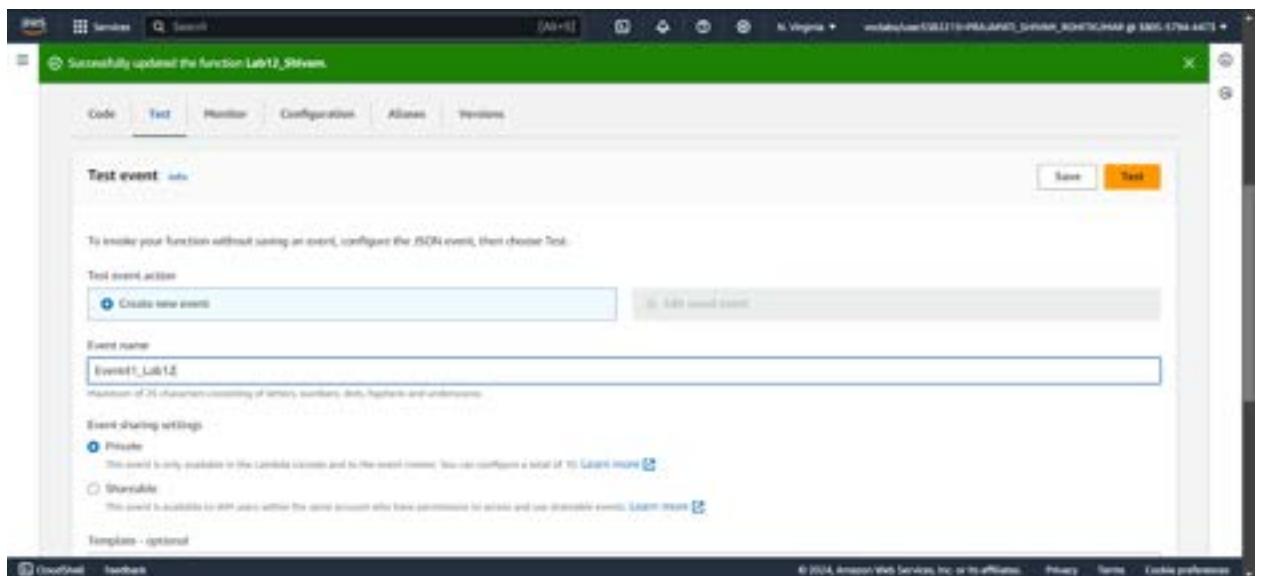
Given some description for your settings



Click on Save.

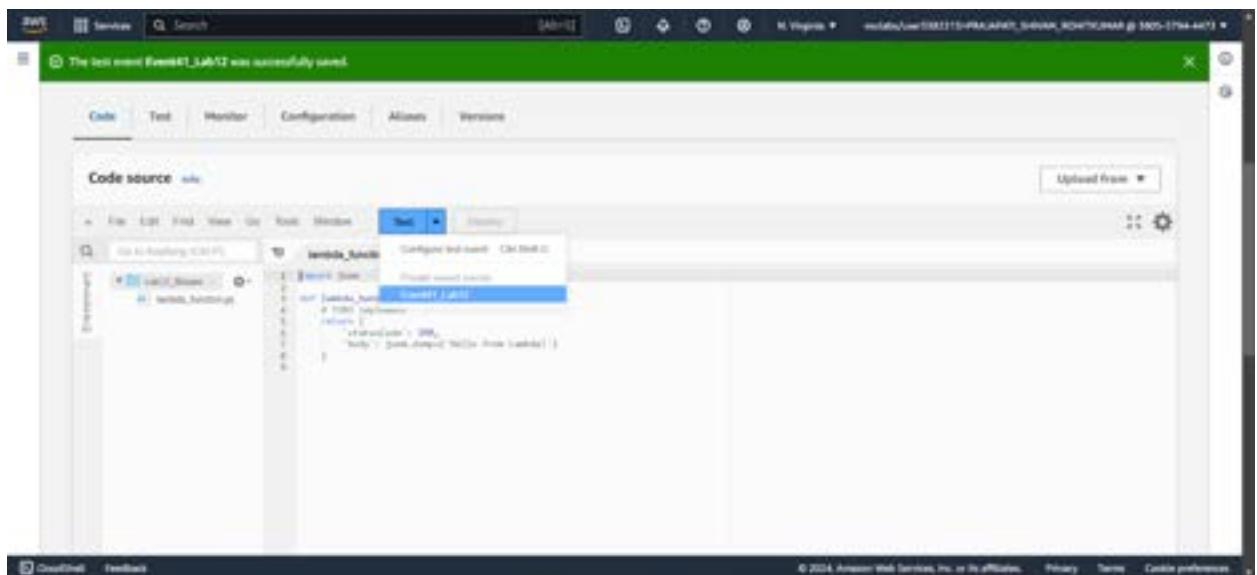


Step 6: Click on the "Test" tab, then select "Create a new event." Give the event a name, set "Event Sharing" to private, and choose the "S3 Put" template. S3 (Simple Storage Service) template allows you to test your Lambda function specifically for events related to uploading files to an S3 bucket.

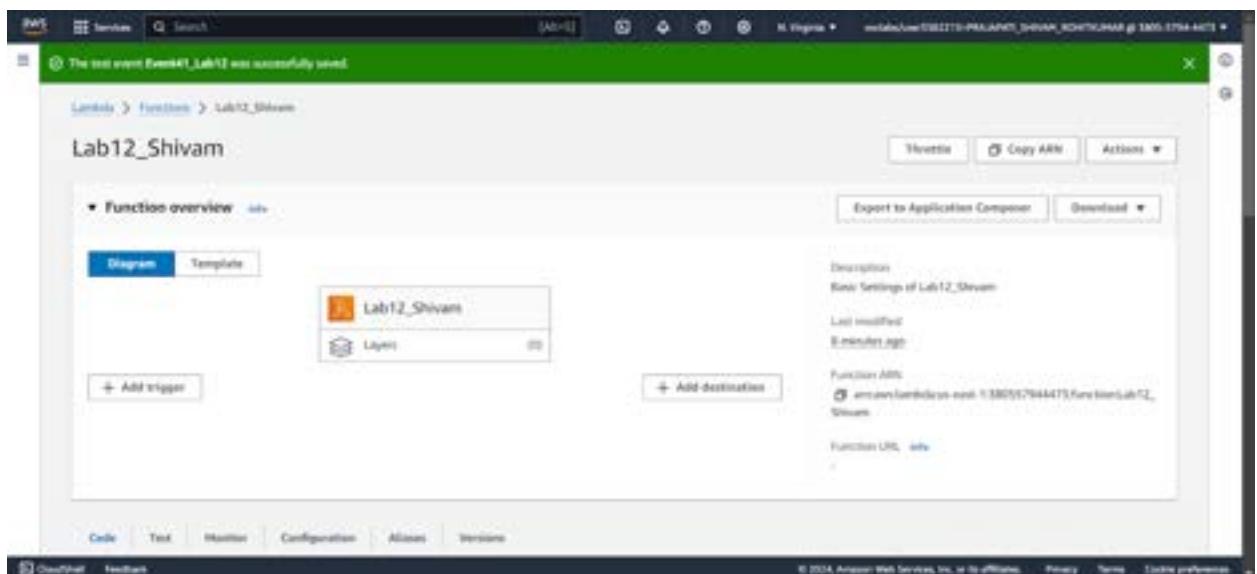


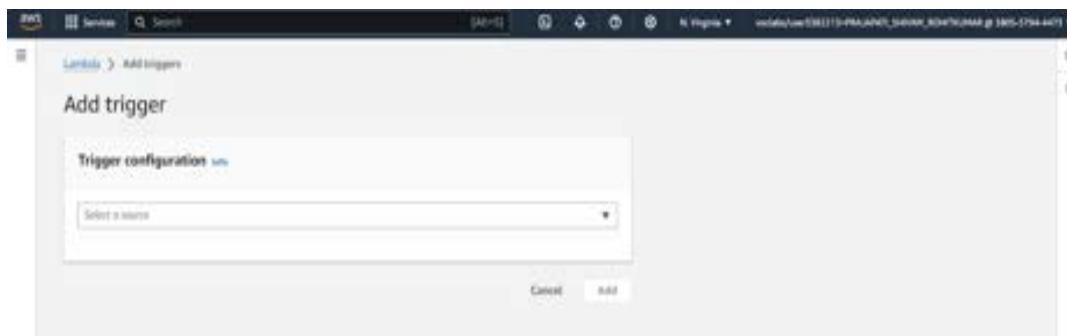
Event Jason code will be automatically generated once S3 -put is selected.

Step 7: Now In the Code section select the created event from the dropdown .

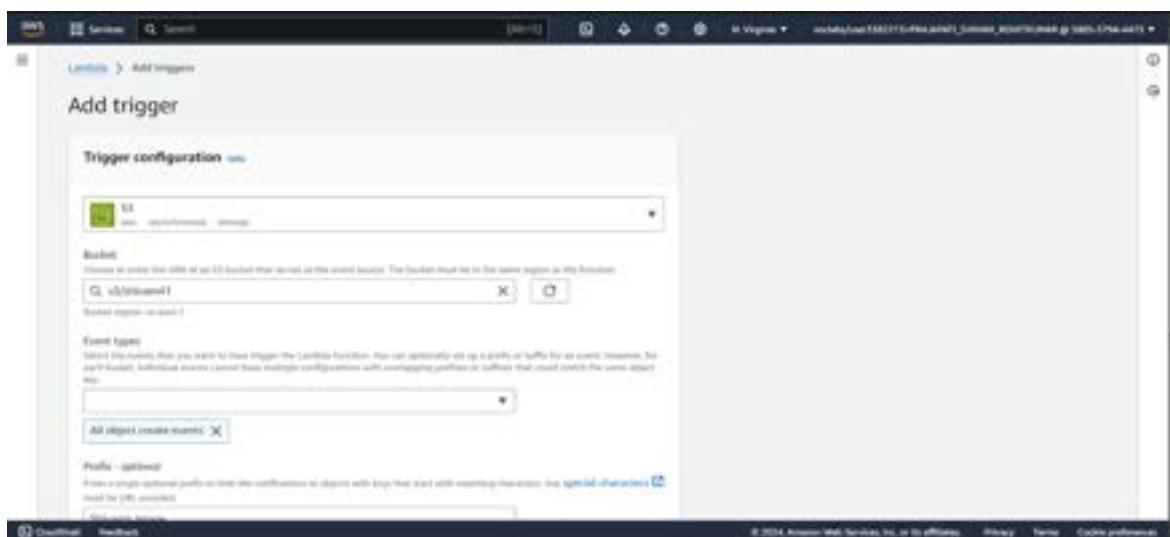
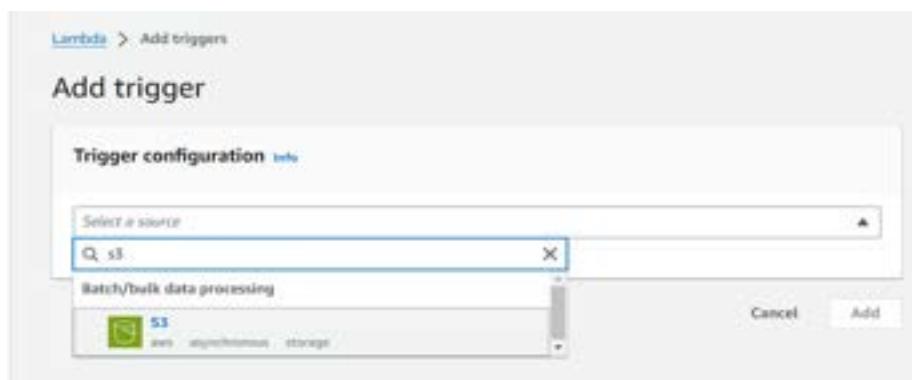


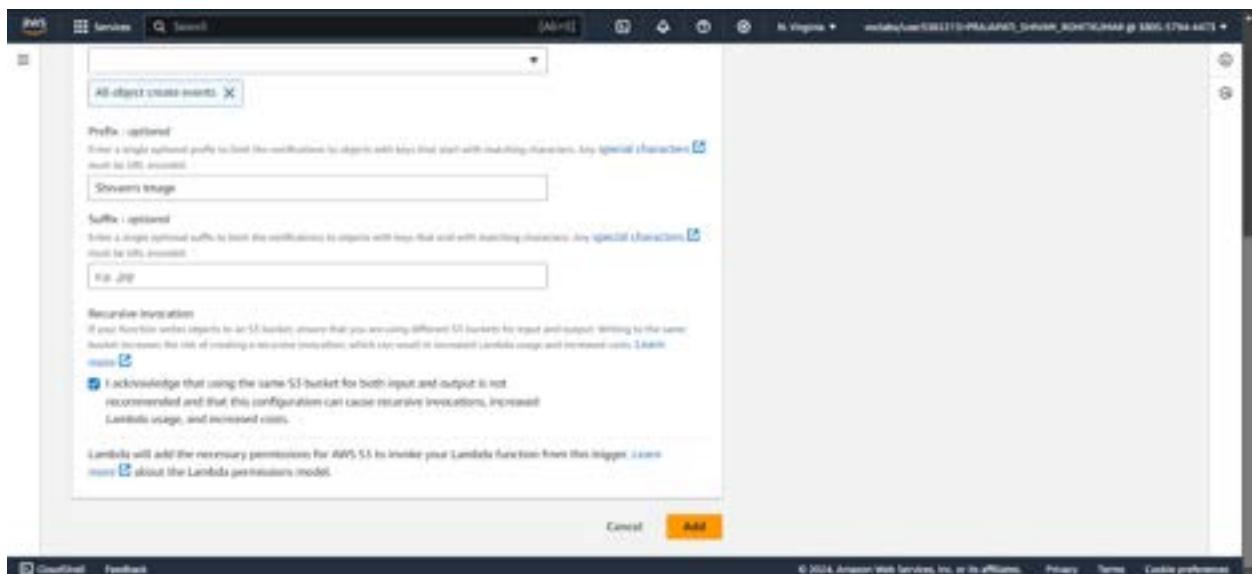
Step 8: In the Lambda function, click on "Add Trigger." Adding a trigger allows your Lambda function to automatically run in response to specific events such as uploads to an S3 bucket



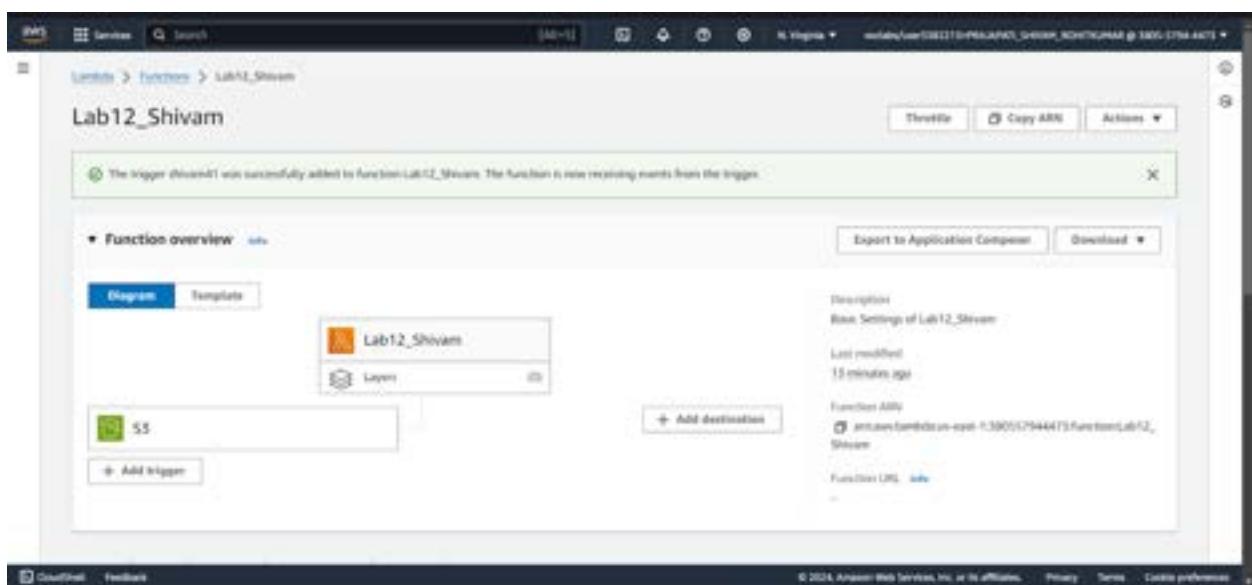


Now select the source as S3, then choose the bucket name from the dropdown menu. Keep the other settings as default, and you can also add a prefix for the image if you want. A prefix for an image (or any file) in S3 is a string that you can use to organize or filter files within a bucket. It acts like a folder name, helping to categorize your files.

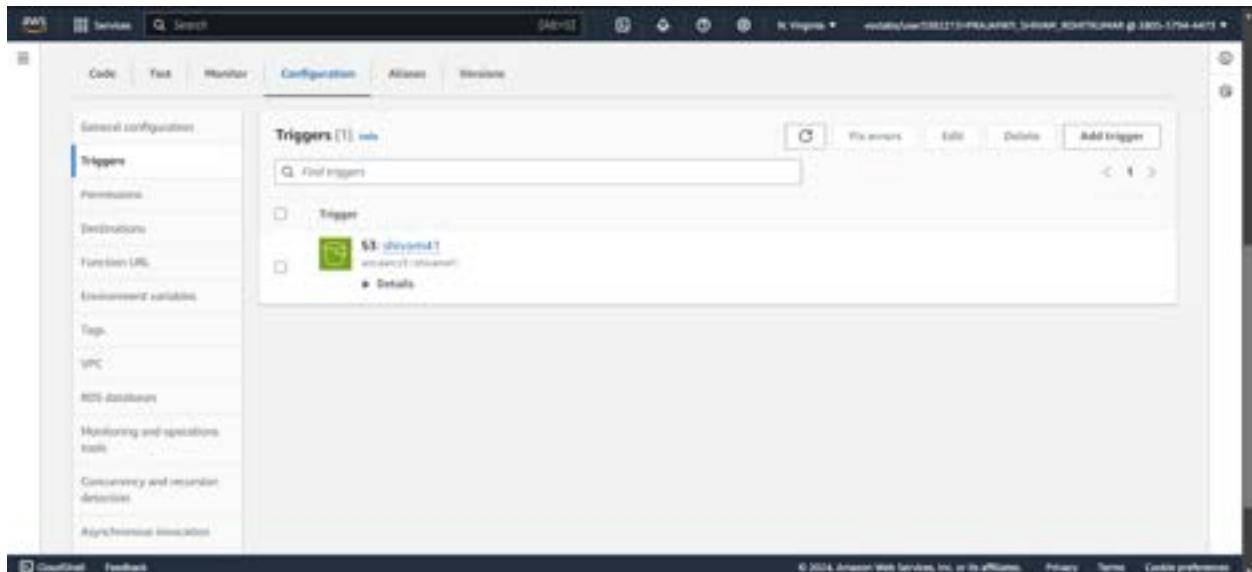




Click on save



Successfully triggered the event



Step 9: Now Write code that logs a message like “An Image has been added” when triggered. Save the file and click on deploy.

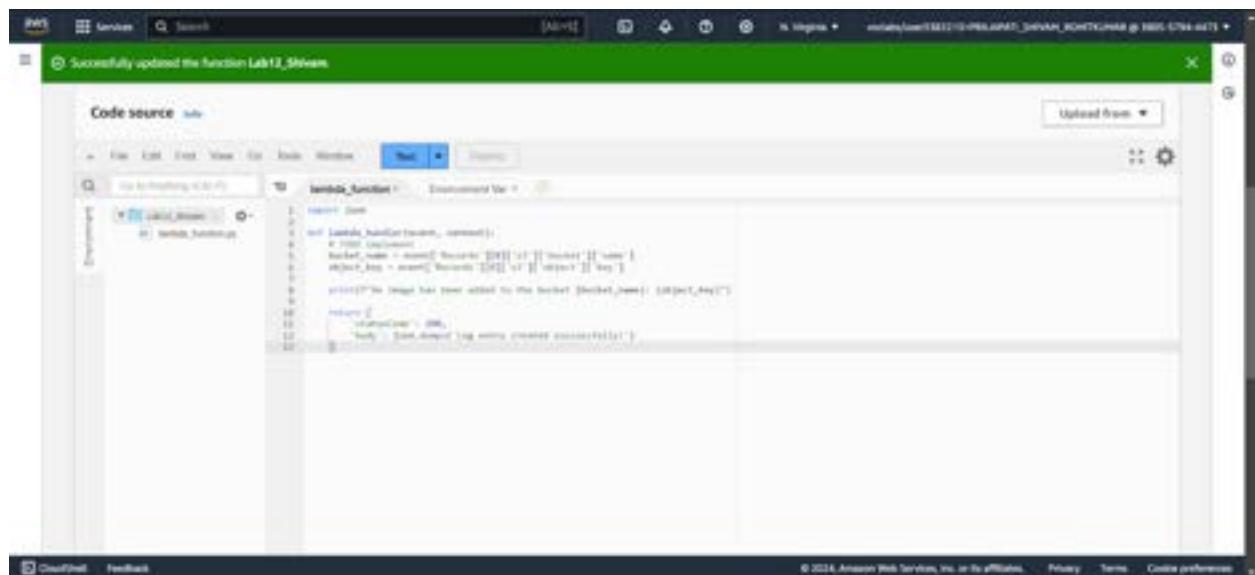
```
import json
def lambda_handler(event, context):
    # TODO implement
    bucket_name = event['Records'][0]['s3']['bucket']['name']
    object_key = event['Records'][0]['s3']['object']['key']

    print(f"An image has been added to the bucket {bucket_name}: {object_key}")

    return {
        'statusCode': 200,
        'body': json.dumps('Log entry created successfully!')
    }
```

```
1 import json
2
3 def lambda_handler(event, context):
4     # TODO implement
5     bucket_name = event['Records'][0]['s3']['bucket']['name']
6     object_key = event['Records'][0]['s3']['object']['key']
7
8     print(f"An image has been added to the bucket {bucket_name}: {object_key}")
9
10    return {
11        'statusCode': 200,
12        'body': json.dumps('Log entry created successfully!')
13    }
```

Save it by Control + S and deploy



The screenshot shows the AWS Lambda function configuration page. At the top, a green banner says "Successfully updated the function Lambda_Shivam". Below it, the "Code source" tab is selected, showing a Python file named "lambda_function.py". The code in the file is:

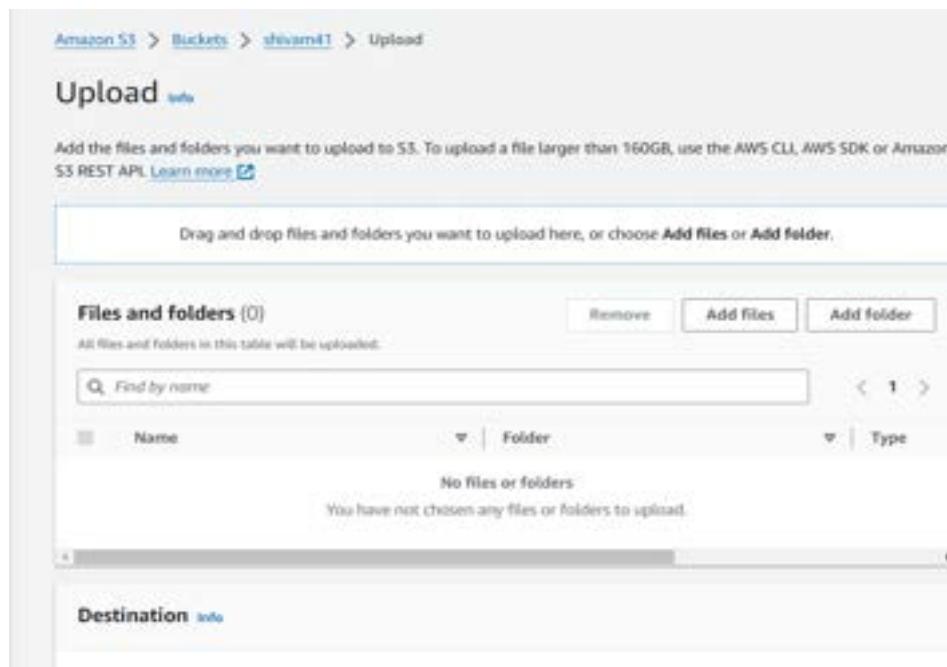
```
import json

def lambda_handler(event, context):
    # Get the object from the event and show its content type
    bucket_name = event['Records'][0]['s3']['bucket']['name']
    object_key = event['Records'][0]['s3']['object']['key']

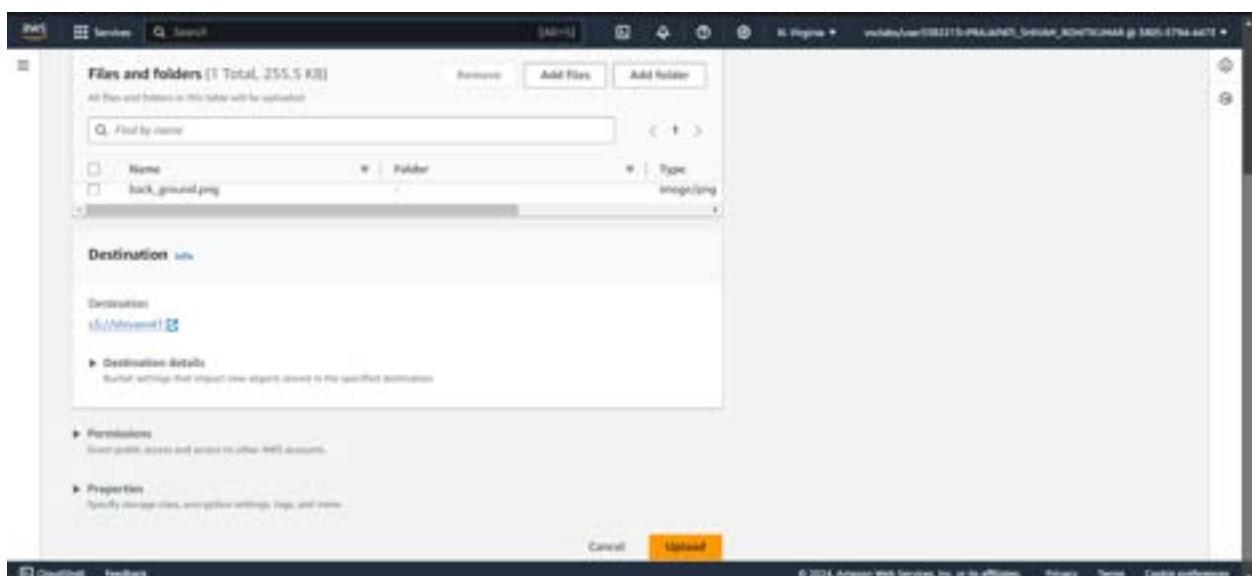
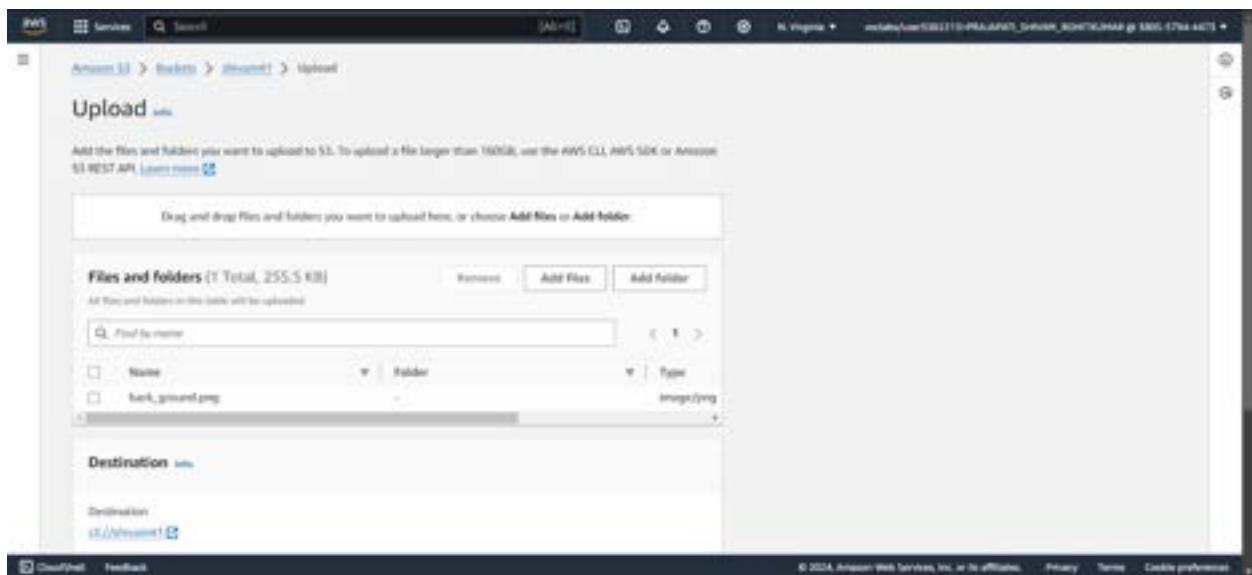
    print(f"An image has been added to the bucket {bucket_name} with key {object_key}")

    return {
        'statusCode': 200,
        'body': json.dumps('Log entry created successfully')
    }
```

Step 10: Now we will upload any image to the bucket

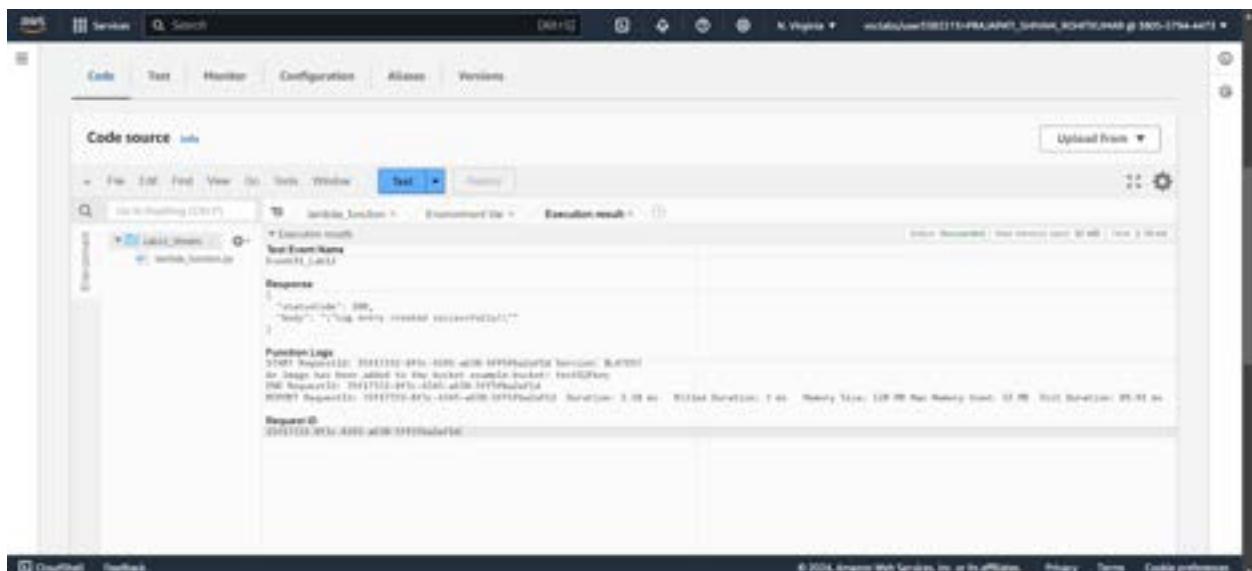


Click on add file where you can upload any image of your choice in your bucket

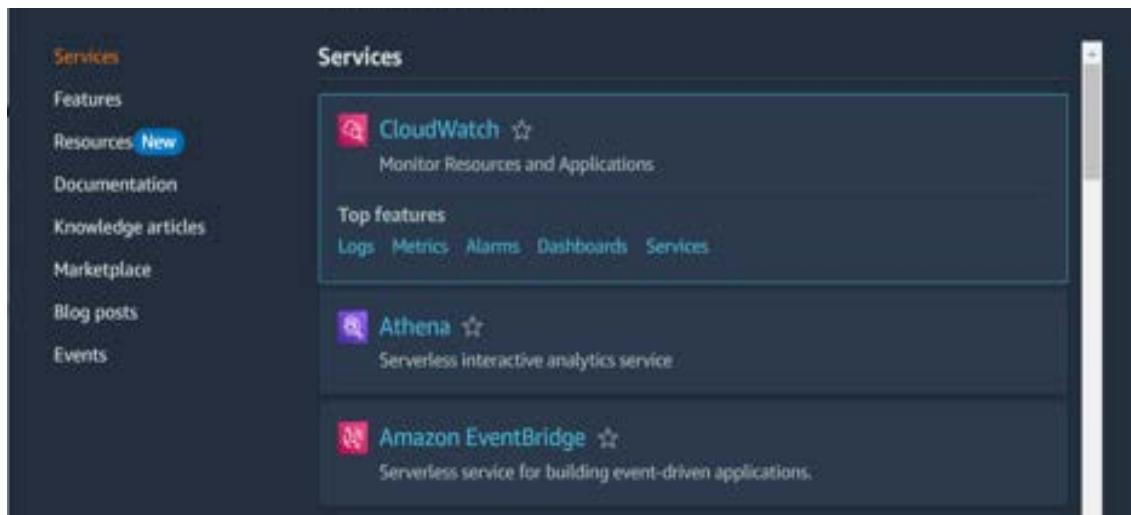


Click on Upload

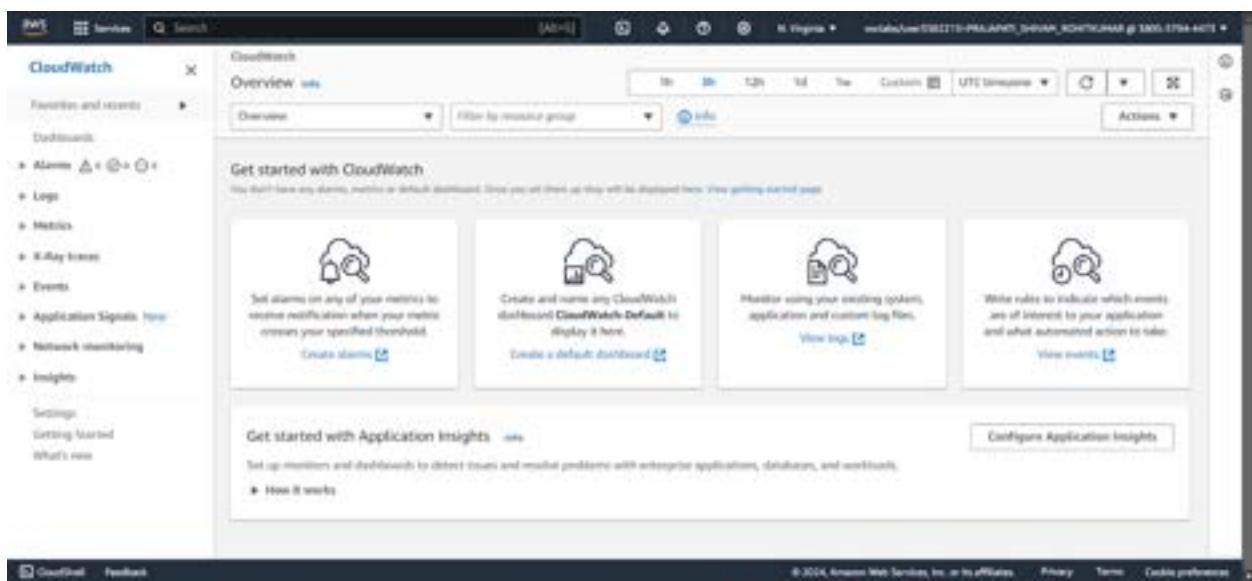
Step 11: Now click on "Test" in Lambda to see if it logs the activity when an image is added to S3.



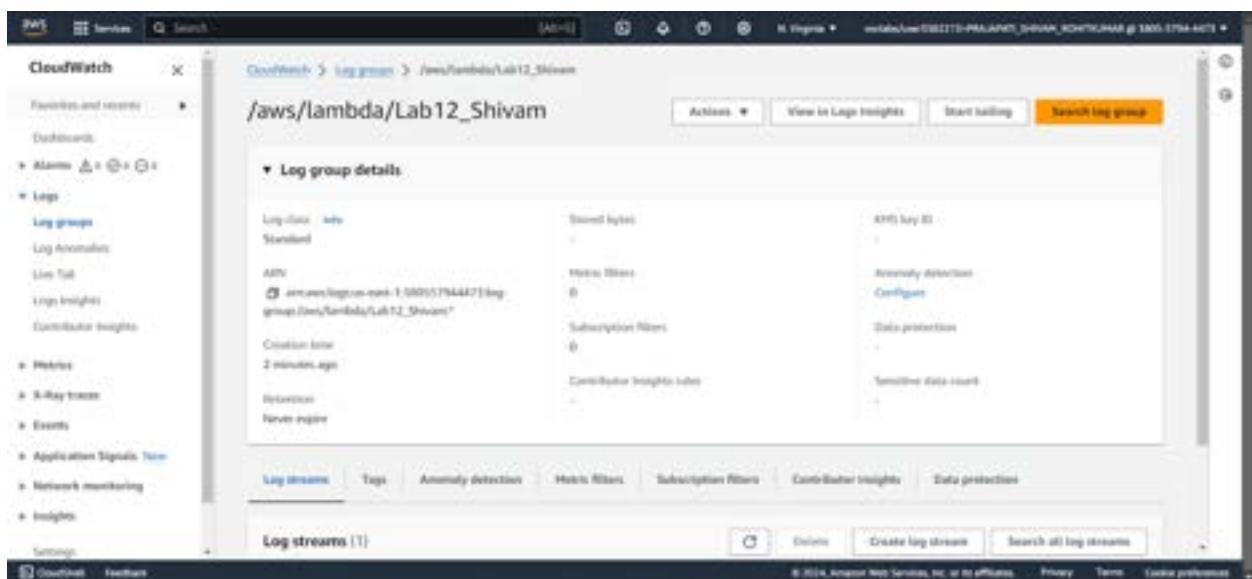
Step 12: Now let's check the logs on CloudWatch. Go to the "Monitor" section and click on "View CloudWatch Logs".



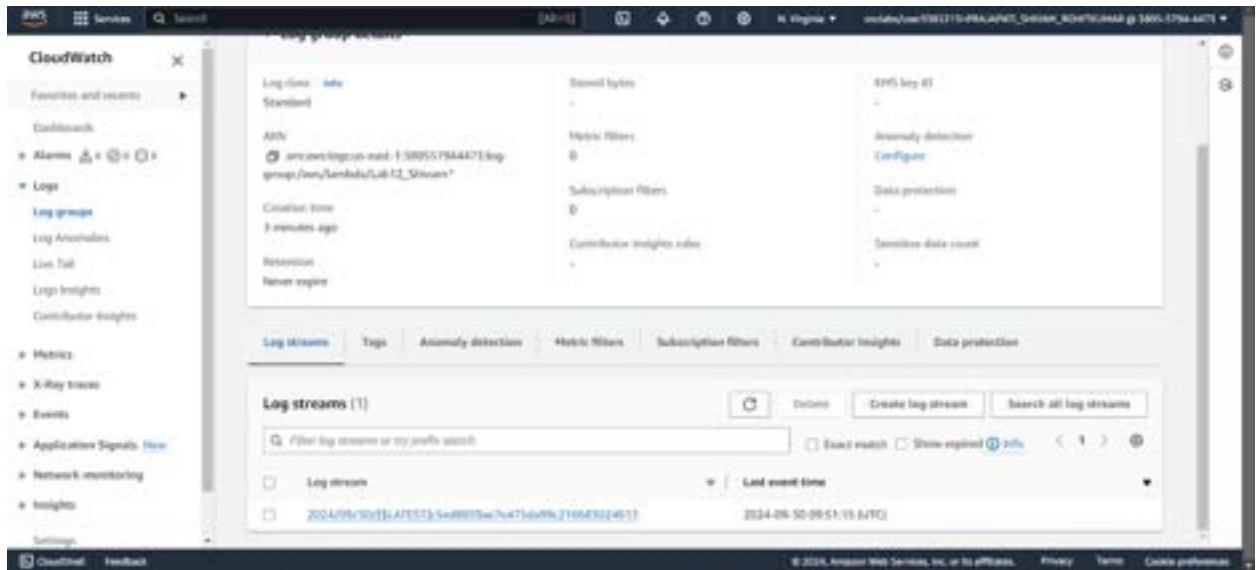
Click on the CloudWatch:



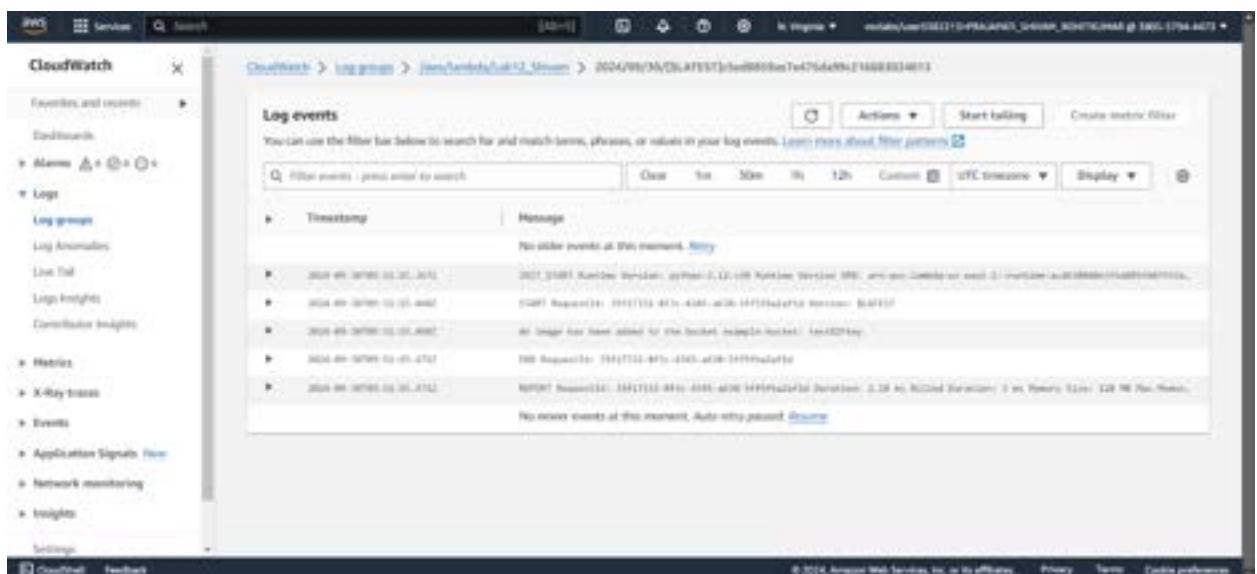
Click on the logs:



Click on the log group:



Scrolled down and click on the log stream:



CONCLUSION:

In this experiment, we successfully created an AWS Lambda function that logs a message when an image is uploaded to an S3 bucket. It's important to choose the S3 Put template for the event; otherwise, the code will give an error. The function was triggered correctly when files were uploaded to S3, showing that Lambda's event-driven design works well. This experiment showed how Lambda can respond to S3 events and how to fix common problems with the event setup.