

Static Hosting [Exp 1(a)]

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

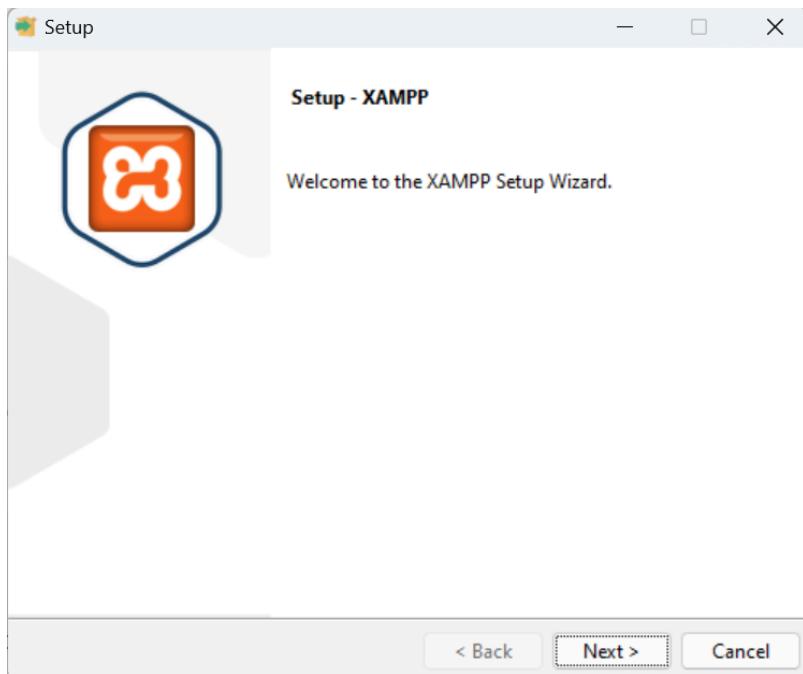
1. Local machine or virtual machine.
2. Amazon S3 bucket.

Steps:-

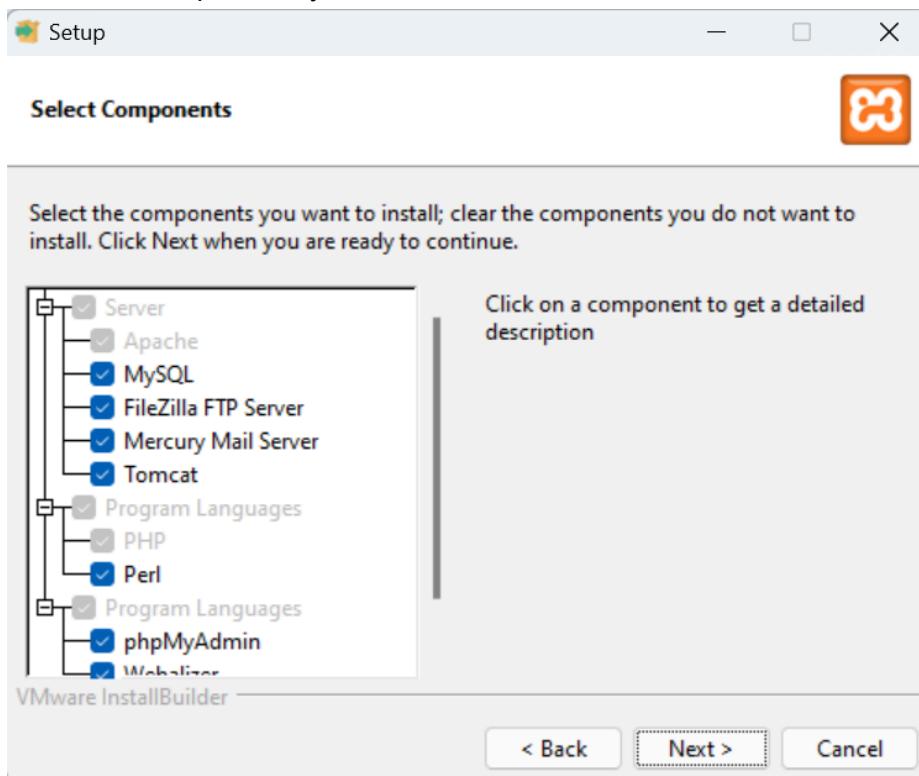
- On local server using XAMPP:-

Step 1: Install XAMPP.

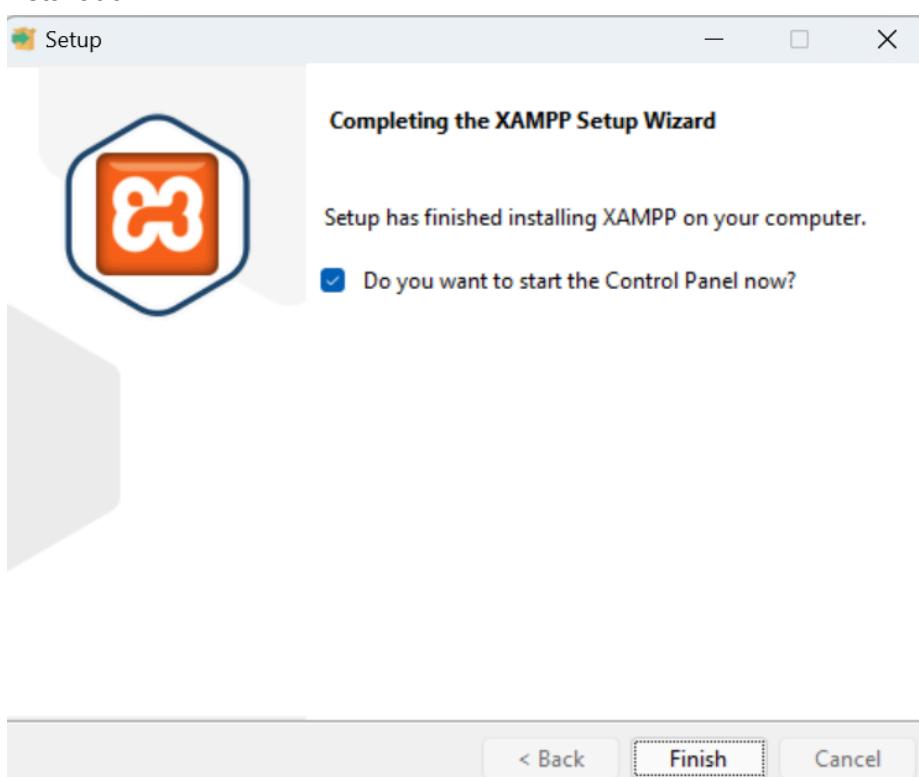
1. Begin the setup process.



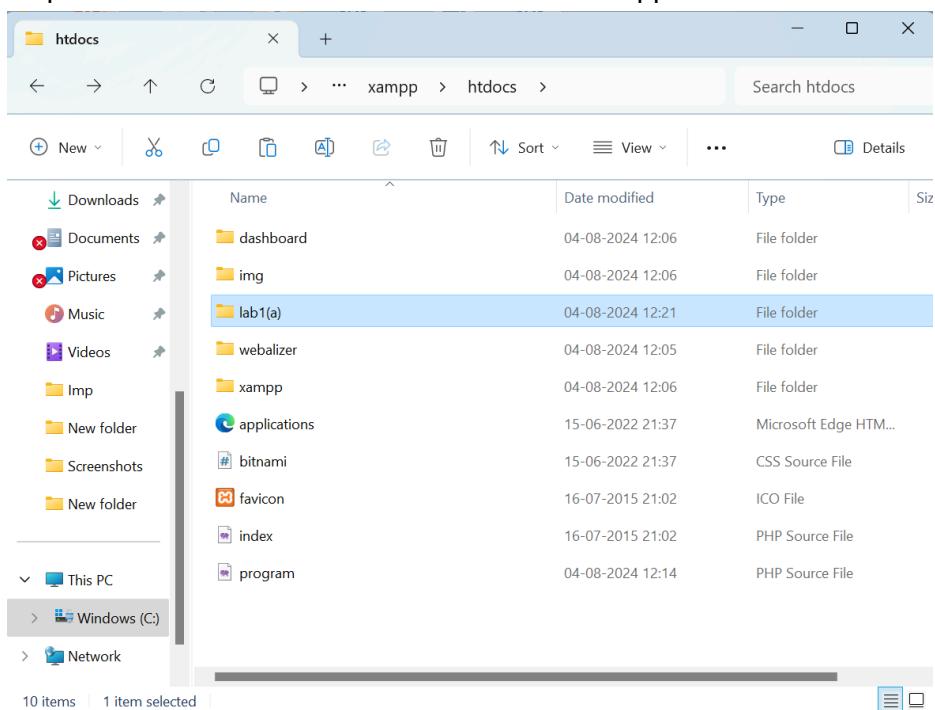
2. Select the components you want to install and click Next.



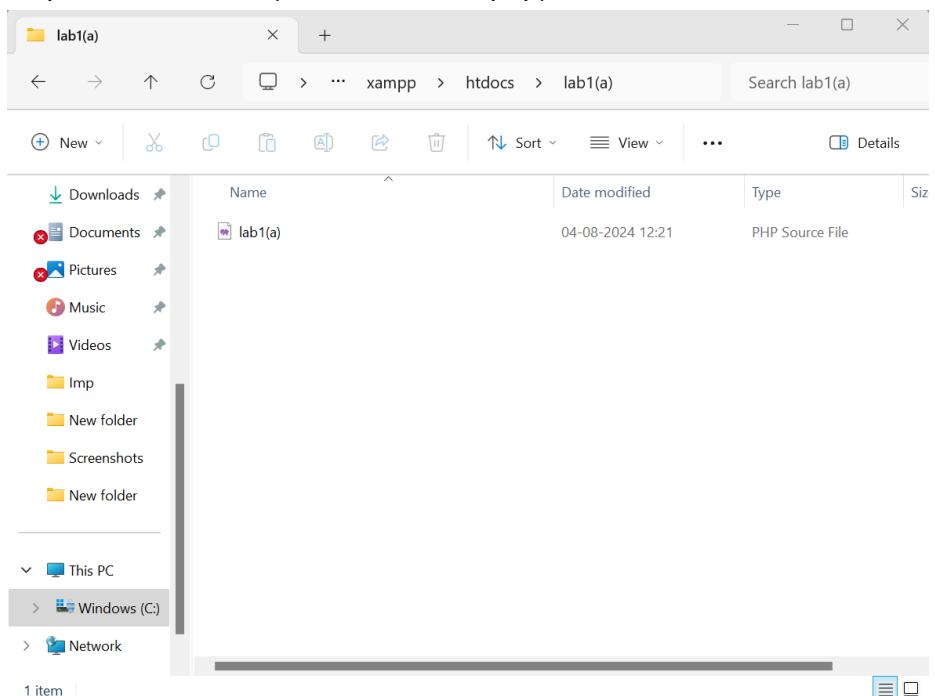
3. Go through the subsequent installation steps (keeping the default options) and finish the installation.



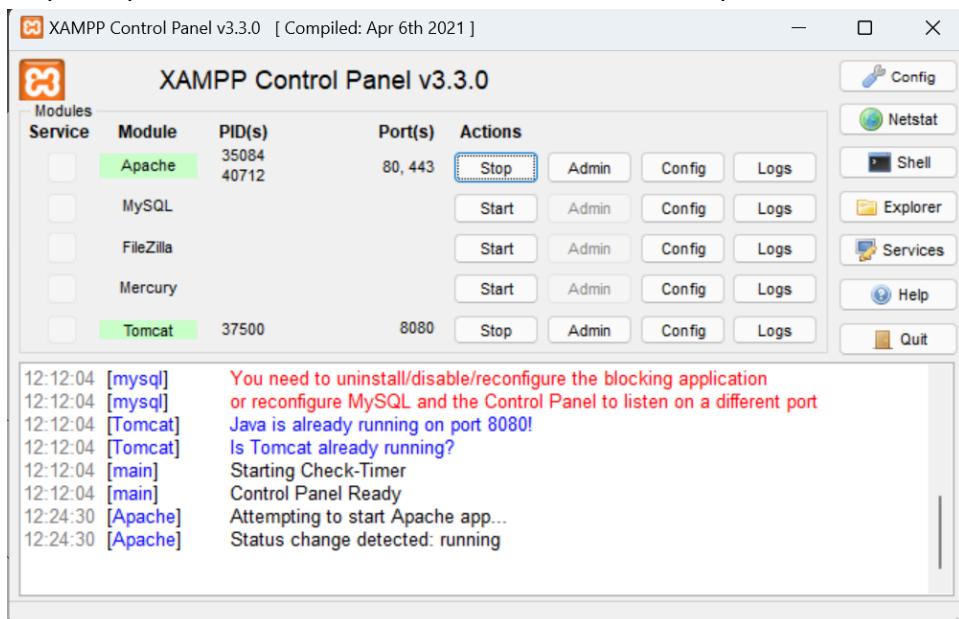
Step 2: Inside the htdocs folder inside of the xampp folder create a new folder.



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



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



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

The browser address bar shows the URL `localhost/lab1(a)/`. The page content is:

Index of /lab1(a)

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
Parent Directory		-	
lab1(a).php	2024-08-04 12:21	31	

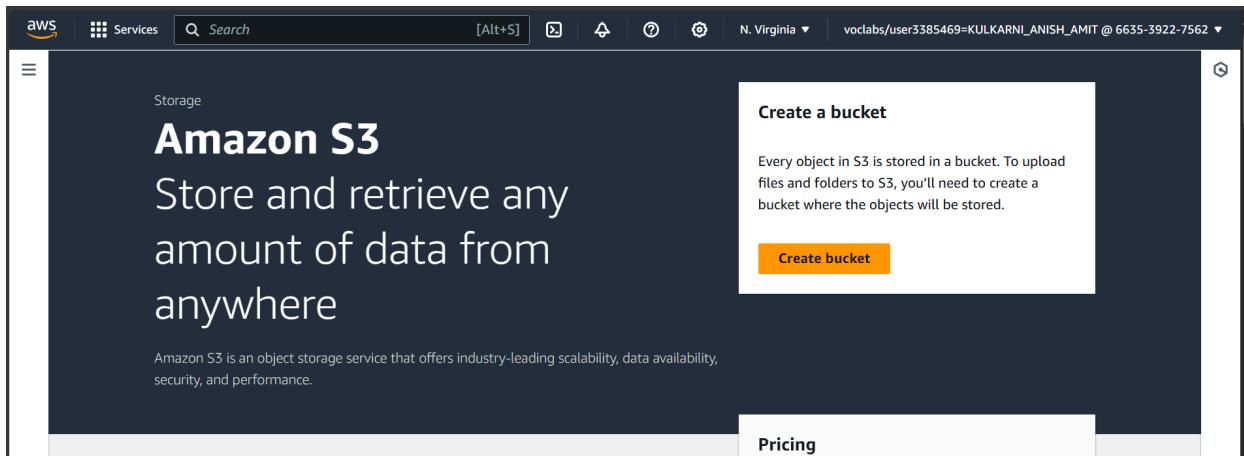
Apache/2.4.58 (Win64) OpenSSL/3.1.3 PHP/8.0.30 Server at localhost Port 80

The browser address bar shows the URL `localhost/lab1(a)/lab1(a).php`. The page content is:

Hello World

- On AWS S3:-

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



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

A screenshot of the 'Create bucket' configuration page. The top navigation bar shows 'Amazon S3 > Buckets > Create bucket'. The main section is titled 'Create bucket' with an 'Info' link. It shows 'General configuration' settings. Under 'AWS Region', 'US East (N. Virginia) us-east-1' is selected. Under 'Bucket type', 'General purpose' is selected (indicated by a blue border). There are two options: 'General purpose' (selected) and 'Directory - New'. The 'Bucket name' field contains 'www.lab1.com'. Below it, a note says 'Bucket name must be unique within the global namespace and follow the bucket naming rules.' with a link 'See rules for bucket naming'. A 'Copy settings from existing bucket - optional' section is present, with a 'Choose bucket' button and a note about prefix format. At the bottom, there's an 'Object Ownership' section with an 'Info' link.

Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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

Edit static website hosting Info

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

Static website hosting
 Disable
 Enable

Hosting type
 Host a static website
Use the bucket endpoint as the web address. [Learn more](#)
 Redirect requests for an object
Redirect requests to another bucket or domain. [Learn more](#)

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

Index document
Specify the home or default page of the website.

Error document - optional
This is returned when an error occurs.

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

Static website hosting
Enabled

Hosting type
Bucket hosting

Bucket website endpoint
When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. [Learn more](#)
<http://www.lab1a.com.s3-website-us-east-1.amazonaws.com>

Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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

The screenshot shows the 'Upload' interface in the AWS S3 console. At the top, there's a message: 'Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. [Learn more](#)'.

Below this is a large dashed box with the instruction: 'Drag and drop files and folders you want to upload here, or choose Add files or Add folder.'

Underneath is a table titled 'Files and folders (2 Total, 2.7 KB)'. It lists two files:

<input type="checkbox"/>	Name	Folder	Type
<input type="checkbox"/>	error.html	-	text/html
<input type="checkbox"/>	lab1.html	-	text/html

Buttons at the top of the table area include 'Remove', 'Add files', and 'Add folder'. A search bar and navigation arrows are also present.

Step 5: Your files are uploaded.

The screenshot shows the 'Upload: status' page. At the top, a green bar indicates 'Upload succeeded' with a link to 'View details below.'

The main area is titled 'Upload: status' with a 'Close' button. It contains a message: 'The information below will no longer be available after you navigate away from this page.'

A 'Summary' section shows the destination as 's3://www.lab1a.com' and the results:

Succeeded	Failed
2 files, 2.7 KB (100.00%)	0 files, 0 B (0%)

Below this is a navigation bar with 'Files and folders' and 'Configuration' tabs, with 'Files and folders' being active.

The 'Files and folders' section shows the same list as the previous screenshot:

Name	Folder	Type	Size	Status	Error
error.html	-	text/html	150.0 B	Succeeded	-
lab1.html	-	text/html	2.6 KB	Succeeded	-

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

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

Block public access (bucket settings) Info

Block all public access

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

- Block public access to buckets and objects granted through *new* access control lists (ACLs)**
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- Block public access to buckets and objects granted through *any* access control lists (ACLs)**
S3 will ignore all ACLs that grant public access to buckets and objects.
- Block public access to buckets and objects granted through *new* public bucket or access point policies**
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Cancel Save changes

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

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

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "PublicReadGetObject",  
            "Effect": "Allow",  
            "Principal": {"AWS": "*"},  
            "Action": "s3:GetObject",  
            "Resource": "arn:aws:s3:::www.lab1a.com/*"  
        }  
    ]  
}
```

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

Hello World!

This is a lab1(a) website

Error Message

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

Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

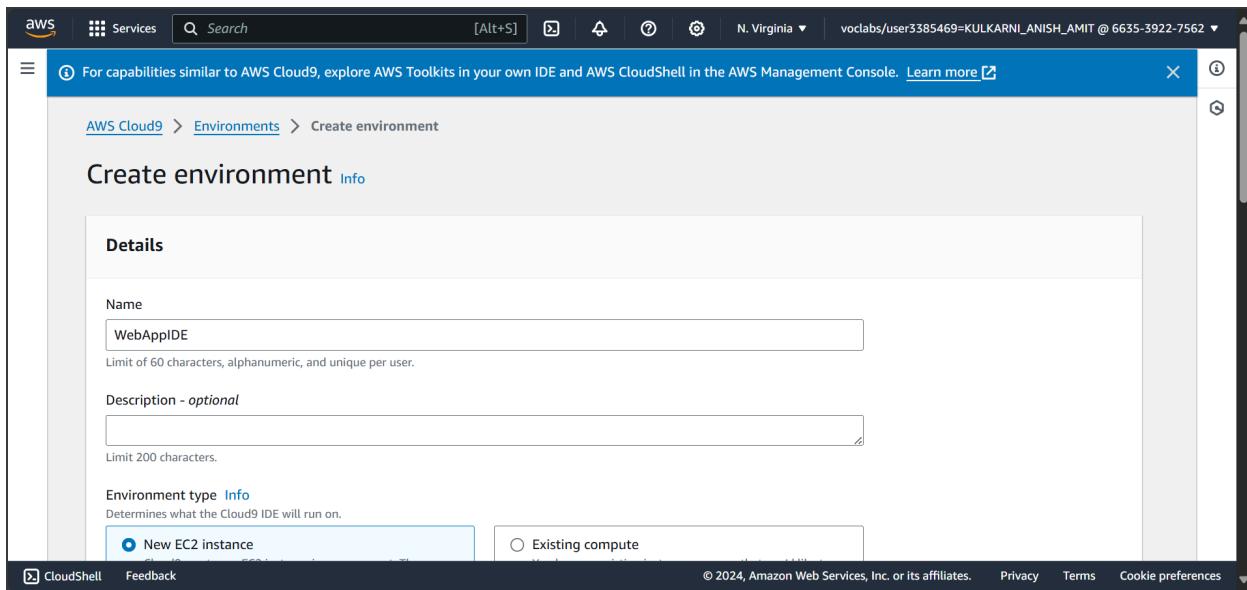
make the website accessible. This hands-on process illustrated the differences between local and cloud hosting, emphasizing the ease of deployment and configuration flexibility that each method offers.

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

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

Steps:-

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



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

The screenshot shows the 'New EC2 instance' configuration page. Under 'Instance type Info', the 't2.micro (1 GiB RAM + 1 vCPU)' option is selected. It is described as 'Free-tier eligible. Ideal for educational users and exploration.' Other options like 't3.small' and 'm5.large' are also listed. Below this, there's a link to 'Additional instance types' for exploring more options. Under 'Platform Info', 'Amazon Linux 2023' is chosen. A 'Timeout' dropdown set to '30 minutes' is shown. At the bottom, standard AWS navigation links for CloudShell, Feedback, Privacy, Terms, and Cookie preferences are visible.

The screenshot shows the 'Network settings' configuration page. Under 'Connection', the 'Secure Shell (SSH)' option is selected, indicated by a blue outline. It is described as 'Accesses environment directly via SSH, opens inbound ports.' Other options like 'AWS Systems Manager (SSM)' are shown without a selection outline. Below this, there's a section for 'VPC settings' and 'Tags - optional'. A note at the bottom states: 'The following IAM resources will be created in your account' and lists 'AWS Service Role for AWS Cloud9' with a brief description and a 'Learn more' link. The bottom of the page includes standard AWS navigation links.

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

The screenshot shows the AWS Cloud9 environments page. At the top, a green success message reads: "Successfully created WebAppIDE. To get the most out of your environment, see Best practices for using AWS Cloud9." Below this, a blue banner provides information about AWS Toolkits and CloudShell. The main table lists one environment:

Name	Cloud9 IDE	Environment type	Connection	Permission	Owner ARN
WebAppIDE	Open	EC2 instance	Secure Shell (SSH)	Owner	arn:aws:sts::663539227562:assumed-role/voclabs/user3385469=KULKARNI

The screenshot shows the AWS Cloud9 IDE interface. The browser title is "Welcome" and the URL is "https://us-east-1.console.aws.amazon.com/cloud9/ide/a3b17e980487461c98ddbd04bd2cc5db?region=us-east-1". The interface includes a navigation bar with File, Edit, Find, View, Go, Run, Tools, Window, Support, Preview, and Run buttons. On the left, there's a sidebar with "Go to Anything (Ctrl-P)" and a file tree showing "WebAppIDE - hor" with files ".c9" and "README.md". The main area displays the "AWS Cloud9" logo and the text "Welcome to your development environment". Below this, it says "AWS Cloud9 allows you to write, run, and debug your code with just a browser. You can tour the IDE, write code for AWS Lambda and Amazon API Gateway, share your IDE with others in real time, and much more." A "Getting started" button is visible. At the bottom, there's a terminal window titled "bash - *ip-172-31-69-178.6x" with the command "voclabs:~/environment \$".

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

The screenshot shows the AWS IAM service interface. On the left, there's a navigation pane titled "Identity and Access Management (IAM)" with options like "Dashboard", "Access management", "User groups", "Users" (which is selected and highlighted in blue), "Roles", and "Policies". The main content area is titled "Users (0) Info" and contains a message: "An IAM user is an identity with long-term credentials that is used to interact with AWS in an account." Below this is a search bar and a table header with columns: "User name", "Path", "Group", "Last activity", "MFA", and "Pas". A message at the bottom of the table says "No resources to display".

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

This screenshot shows the "Specify user details" step of the IAM user creation wizard. The left sidebar lists steps: Step 1 "Specify user details" (selected), Step 2 "Set permissions", and Step 3 "Review and create". The main panel is titled "User details" and contains a "User name" field with "user1" typed in. Below it is a note: "The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and +, -, @, _ (hyphen)". There's also an optional checkbox for "Provide user access to the AWS Management Console - optional" with a note: "If you're providing console access to a person, it's a best practice [link] to manage their access in IAM Identity Center". A callout box provides information about generating programmatic access: "If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. [Learn more]". At the bottom right are "Cancel" and "Next" buttons.

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

This screenshot shows the "Set permissions" step of the IAM user creation wizard. The left sidebar shows Step 1 "Specify user details" (selected) and Step 2 "Set permissions". The main panel is titled "Permissions options" and contains three radio button choices: "Add user to group" (selected), "Copy permissions", and "Attach policies directly". A callout box at the bottom left says: "Get started with groups. Create a group and select policies to attach to the group. We recommend using groups to manage user permissions by job function, AWS service access, or custom permissions. [Learn more]". At the bottom right is a "Create group" button.

Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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

Create user group

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

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

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

Permissions policies (947)

Filter by Type
Search All ty... ▾

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

Cancel **Create user group**

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

REVIEW AND CREATE

Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.

User details

User name user1	Console password type None	Require password reset No
--------------------	-------------------------------	------------------------------

Permissions summary

Name	Type	Used as
No resources		

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

No tags associated with the resource.

Add new tag
You can add up to 50 more tags.

Cancel Previous **Create user**

Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

Step 9: User and User group are created successfully.

The screenshot shows two separate IAM management pages. The top part is the 'Users' page, which displays a single user named 'user1'. A success message at the top indicates 'User created successfully'. The bottom part is the 'User groups' page, which displays a single group named 'group1'. Both pages include search bars, navigation buttons, and standard AWS UI elements like 'Create user' and 'Create group' buttons.

Users (1) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

User name	Path	Group:	Last activity	MFA
user1	/	0	-	-

User groups (1) Info

A user group is a collection of IAM users. Use groups to specify permissions for a collection of users.

Group name	Users	Permissions	Creation time
group1	⚠ 0	Pending	5 minutes ago

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

This screenshot shows the detailed view of the 'group1' user group. The left sidebar shows the IAM navigation menu with 'User groups' selected. The main content area displays the group's summary information, including its name, creation time, and ARN. Below this, there are tabs for 'Users', 'Permissions' (which is currently selected), and 'Access Advisor'. At the bottom, there is a section for managing permissions policies.

Identity and Access Management (IAM)

group1 Info

Summary

User group name group1	Creation time August 04, 2024, 16:59 (UTC+05:30)	ARN arn:aws:iam::010928206130:group/group1
---------------------------	---	---

Permissions

Permissions policies (0) Info

You can attach up to 10 managed policies.

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

Other permission policies (1/945)

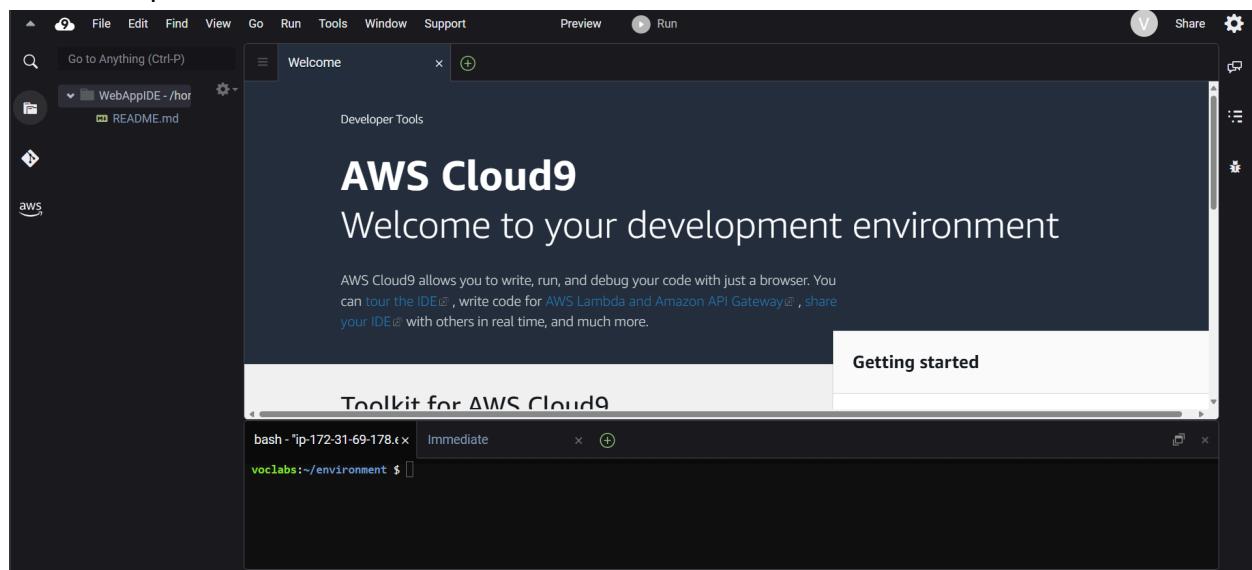
You can attach up to 10 managed policies to this user group. All of the users in this group inherit the attached permissions.

Filter by Type

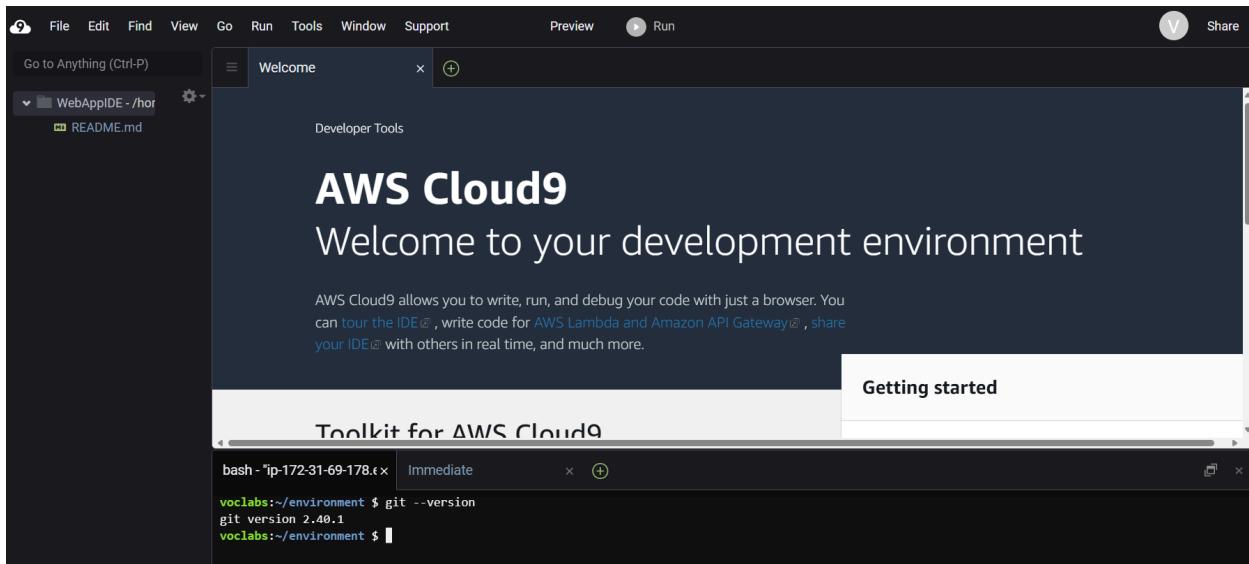
Policy name	Type	Used as	Description
AWSCloud9Admin	AWS managed	None	Provides administrator access to AWS ...
AWSCloud9Environ	AWS managed	None	Provides the ability to be invited into A...
AWSCloud9SSMInsta	AWS managed	None	This policy will be used to attach a rol...
AWSCloud9User	AWS managed	None	Provides permission to create AWS Clo...

This attaches the policies to your user group.

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



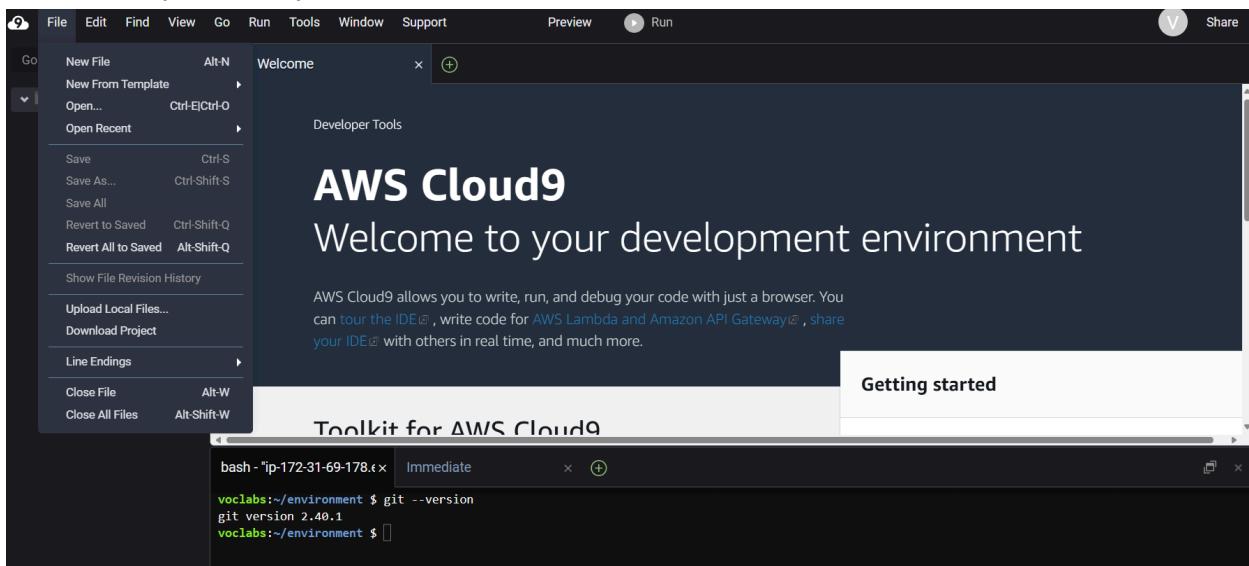
For example, entering the command 'git --version' produces the following output:-



The screenshot shows the AWS Cloud9 IDE interface. The top navigation bar includes File, Edit, Find, View, Go, Run, Tools, Window, Support, Preview, and Run buttons. A 'Share' button is in the top right. The left sidebar shows a file tree with 'WebAppIDE - /home' and 'README.md'. The main area displays the AWS Cloud9 welcome screen with the title 'AWS Cloud9' and subtitle 'Welcome to your development environment'. Below this, a message states: 'AWS Cloud9 allows you to write, run, and debug your code with just a browser. You can tour the IDE, write code for AWS Lambda and Amazon API Gateway, share your IDE with others in real time, and much more.' A 'Getting started' section is visible. At the bottom, a terminal window titled 'Toolkit for AWS Cloud9' shows the command 'git --version' being run, with the output 'git version 2.40.1' displayed.

```
bash -*ip-172-31-69-178.t* x Immediate x +  
vocabs:~/environment $ git --version  
git version 2.40.1  
vocabs:~/environment $
```

Step 13: To add a file, click on 'File' in the top left corner, then click on 'New From Template' and choose the type of file you want to create.



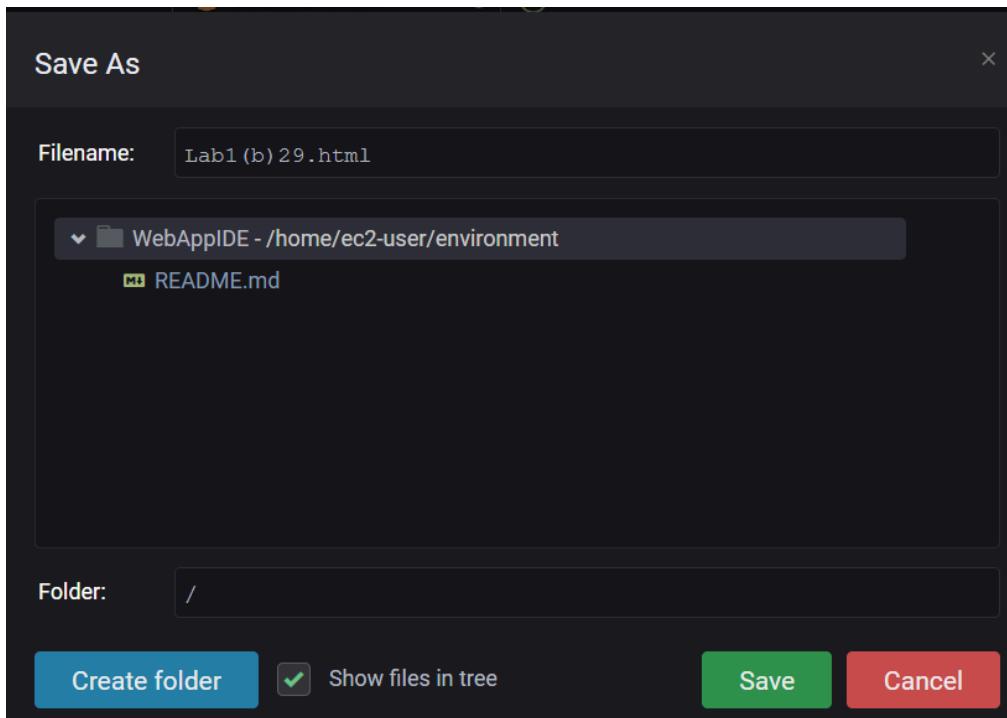
For this example, we create an HTML file. This gives a basic HTML code template on the coding IDE.

The screenshot shows a dark-themed IDE interface. At the top, there's a menu bar with File, Edit, Find, View, Go, Run, Tools, Window, Support, Preview, and Run. Below the menu is a toolbar with a gear icon, a search bar labeled 'Go to Anything (Ctrl+P)', and a share icon. The main workspace has a title bar 'Welcome' and a tab 'Untitled1.html'. The code editor contains the following HTML:

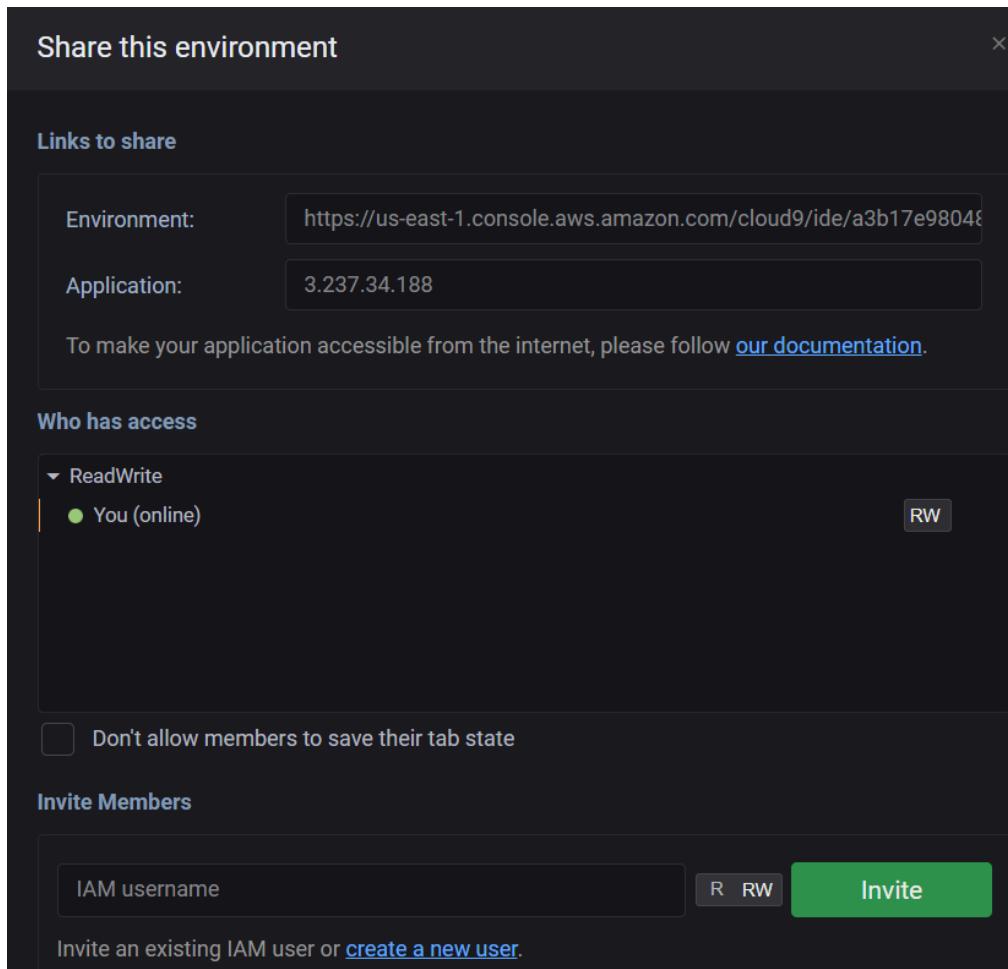
```
<!DOCTYPE html>
<html>
  <head>
    <title>WebApp29IDE </title>
  </head>
  <body>
    <h1>Lab 1(b) 29</h1>
  </body>
</html>
```

Below the code editor is a terminal window titled 'bash - *ip-172-31-69-178.tx'. It shows the command 'git --version' being run and the output 'git version 2.40.1'. The status bar at the bottom right indicates '9:8 HTML Spaces: 4'.

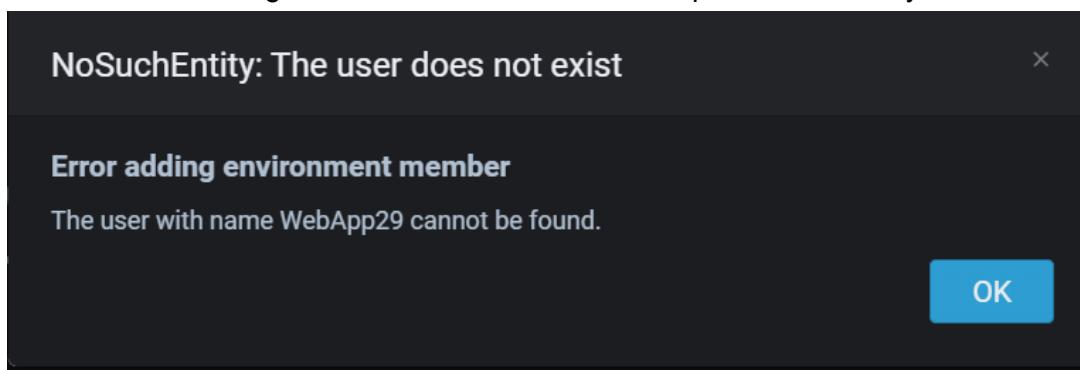
Step 14: After creating your file, click on 'File' in the top left corner and click on 'Save' from the drop-down menu.



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



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



Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

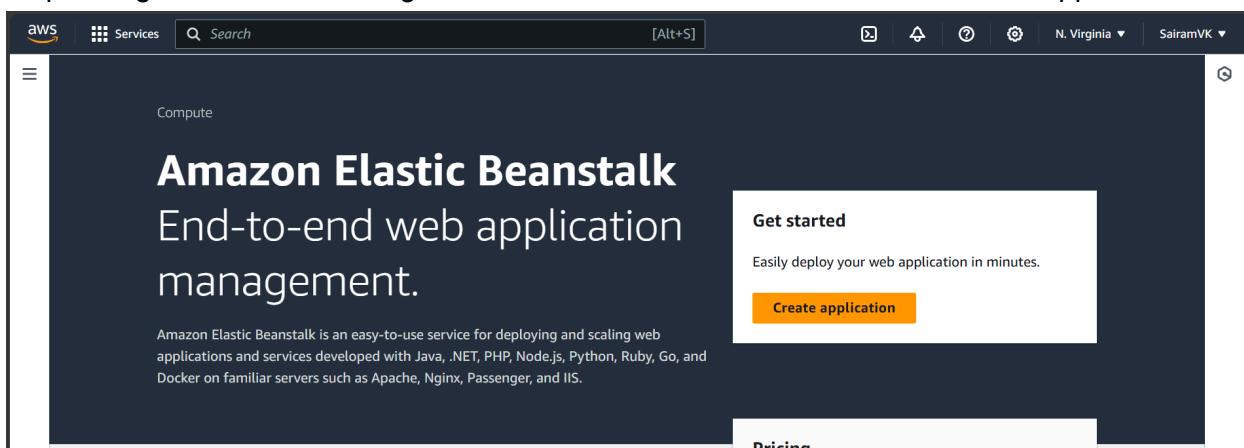
Conclusion: This experiment highlights how to set up and work with AWS Cloud9, although account limitations on AWS Academy prevent full collaboration demonstration. By understanding the steps and limitations, users can better leverage cloud infrastructure for development and teamwork when fully functional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

Experiment No. 2

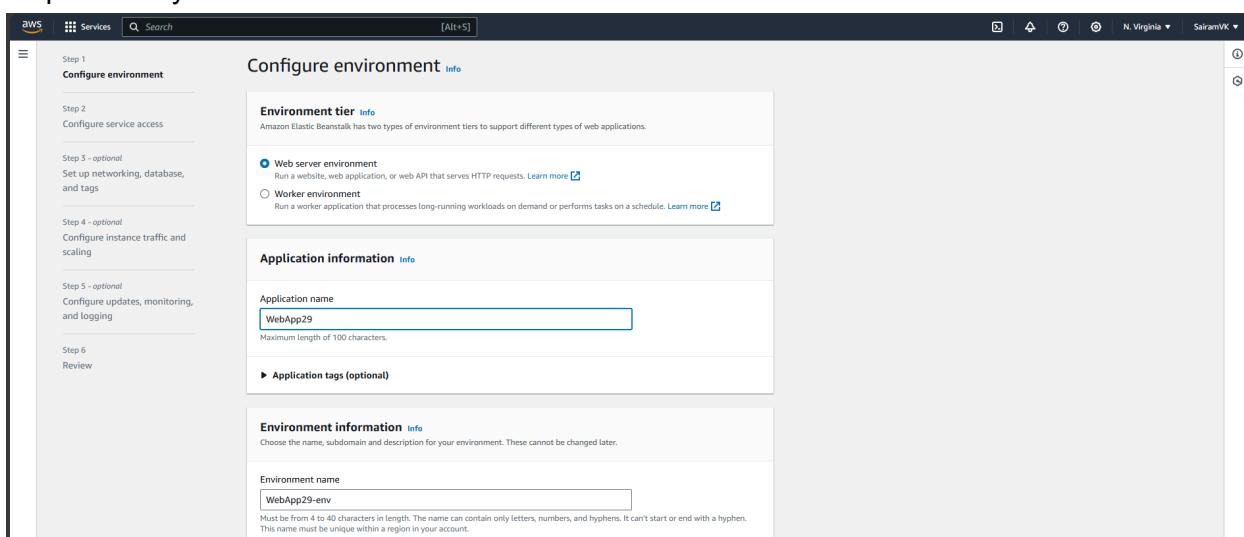
Aim: To build your application using AWS Codebuild and deploy on S3/SEBS using AWS CodePipeline, deploy a sample application on an EC2 instance using AWS CodeDeploy.

Steps:-

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



Step 2: Give your environment a name.



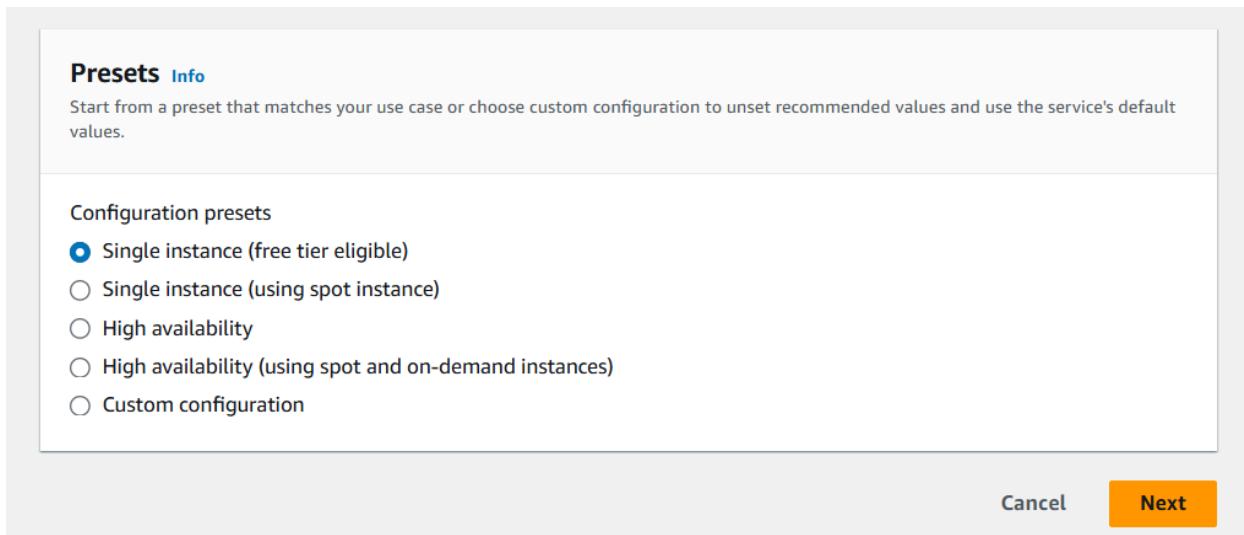
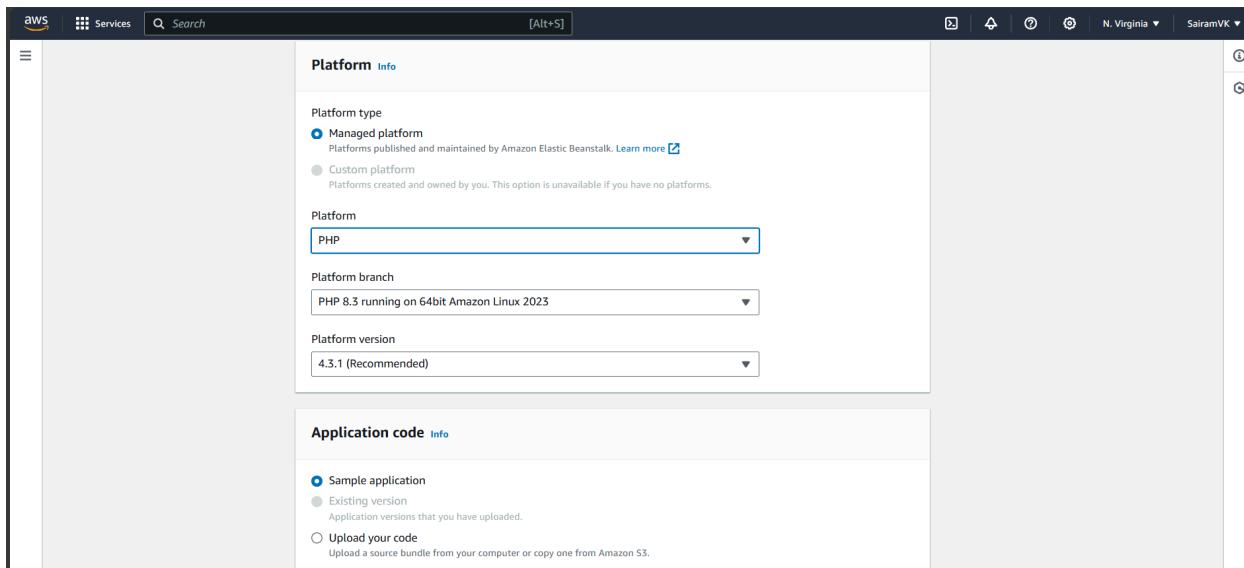
Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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



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

Service access

IAM roles, assumed by Elastic Beanstalk as a service role, and EC2 instance profiles allow Elastic Beanstalk to create and manage your environment. Both the IAM role and instance profile must be attached to IAM managed policies that contain the required permissions. [Learn more](#)

Service role

Create and use new service role
 Use an existing service role

Existing service roles

Choose an existing IAM role for Elastic Beanstalk to assume as a service role. The existing IAM role must have the required IAM managed policies.

AWSCloud9SSMAccessRole

EC2 key pair

Select an EC2 key pair to securely log in to your EC2 instances. [Learn more](#)

Choose a key pair

EC2 instance profile

Choose an IAM instance profile with managed policies that allow your EC2 instances to perform required operations.

AWSCloud9SSMInstanceProfile

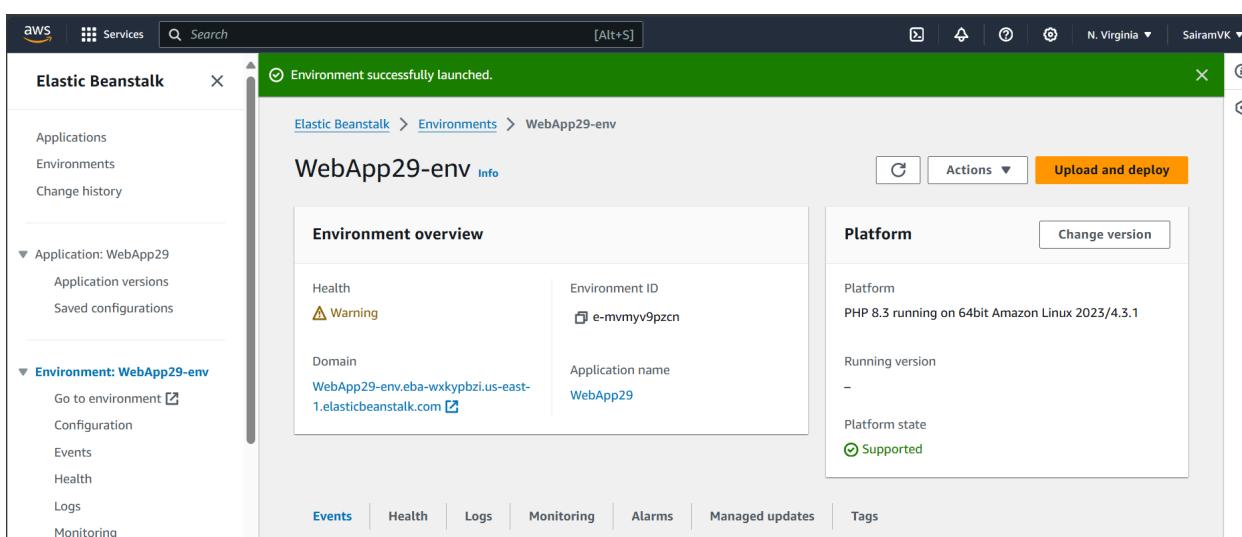
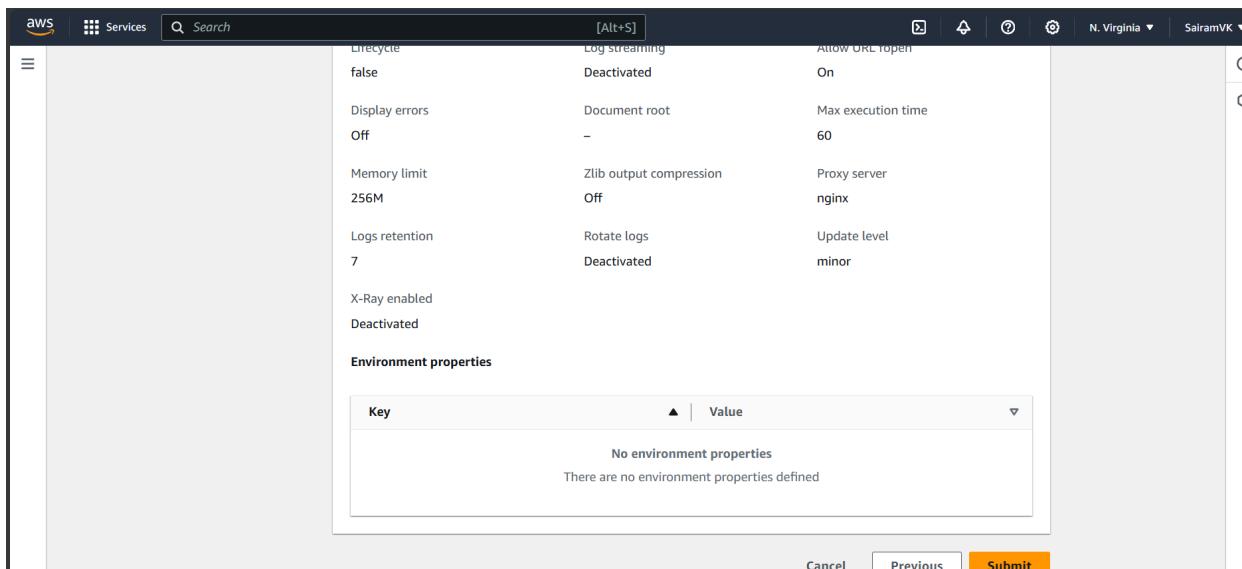
[View permission details](#)

Cancel | Skip to review | Previous | **Next**

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

Environment information	
Environment tier	Application name
Web server environment	WebApp29
Environment name	Application code
WebApp29-env	Sample application
Platform	
arn:aws:elasticbeanstalk:us-east-1::platform/PHP 8.3 running on 64bit Amazon Linux 2023/4.3.1	

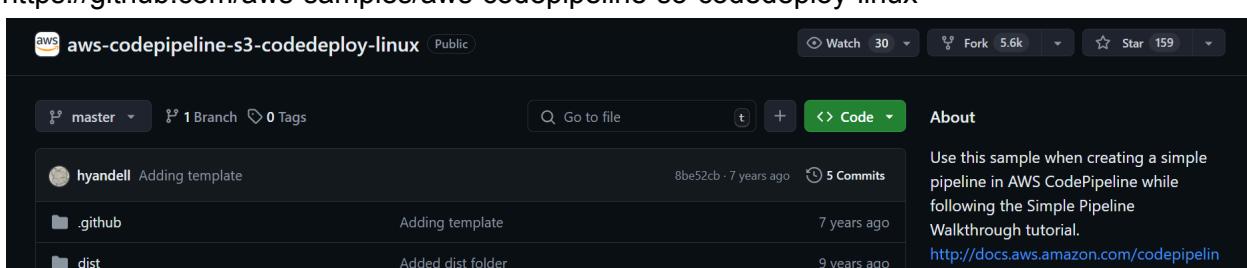
Service access info	
Configure the service role and EC2 instance profile that Elastic Beanstalk uses to manage your environment. Choose an EC2 key pair to securely log in to your EC2 instances.	
Service role	EC2 instance profile
arn:aws:iam::011528263337:role/service-role/AWSCloud9SSMAccessRole	AWSCloud9SSMInstanceProfile



The environment gets created.

Step 6: Go to the github link below and fork the repository into your personal github in order to get the sample code for deploying a file on AWS CodePipeline.

<https://github.com/aws-samples/aws-codepipeline-s3-codedeploy-linux>

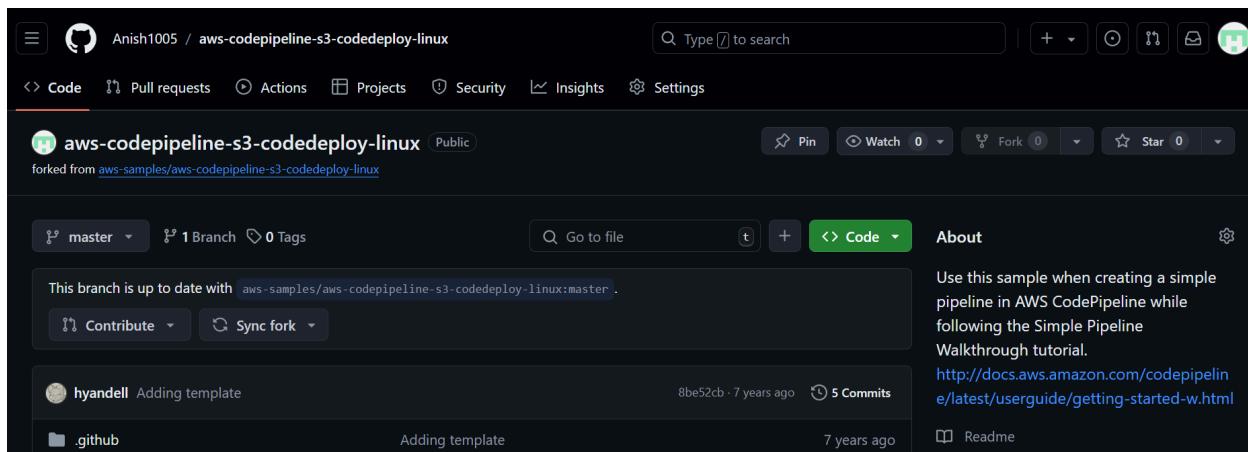


Name: Anish Kulkarni

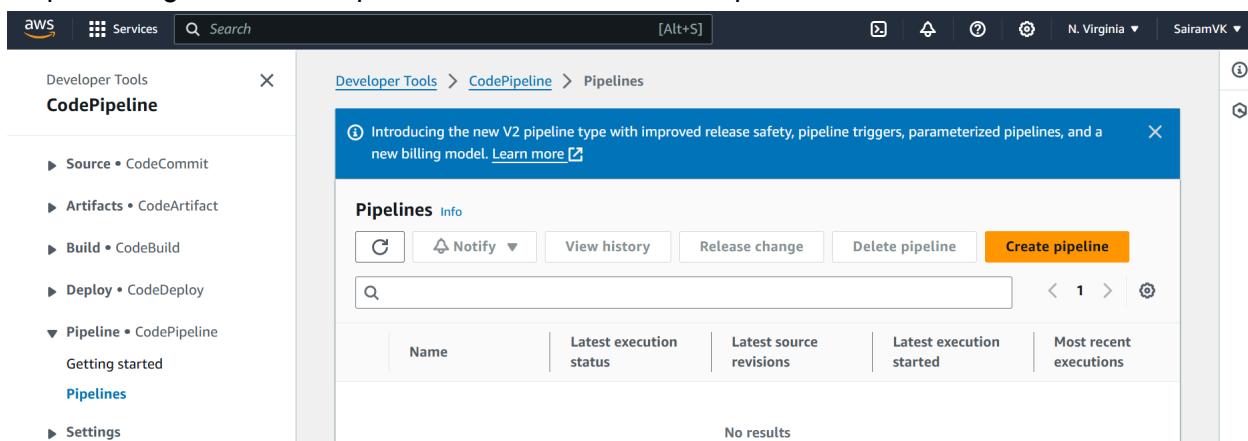
Roll No.: 29

Class: D15C

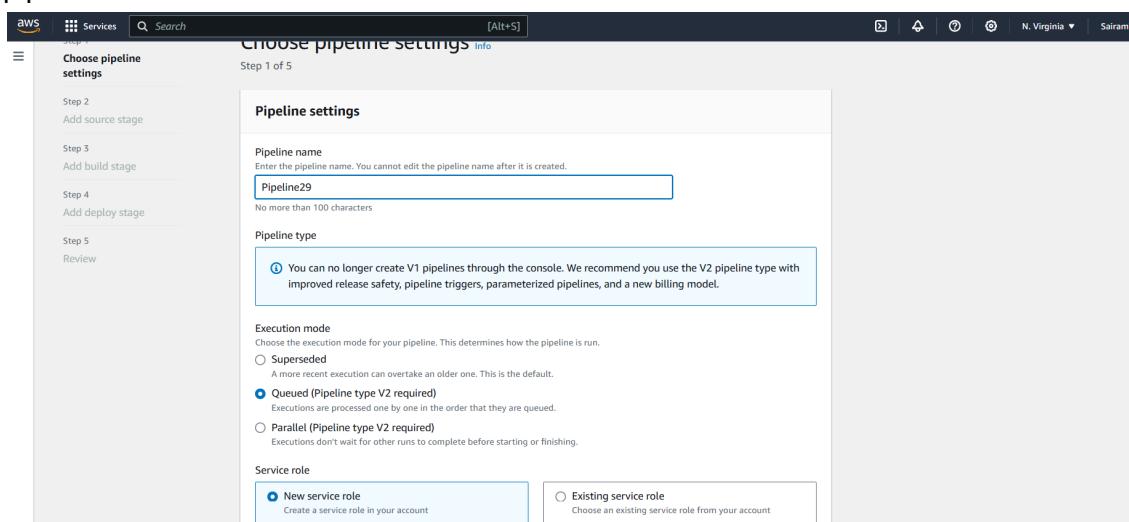
AY: 2024-25

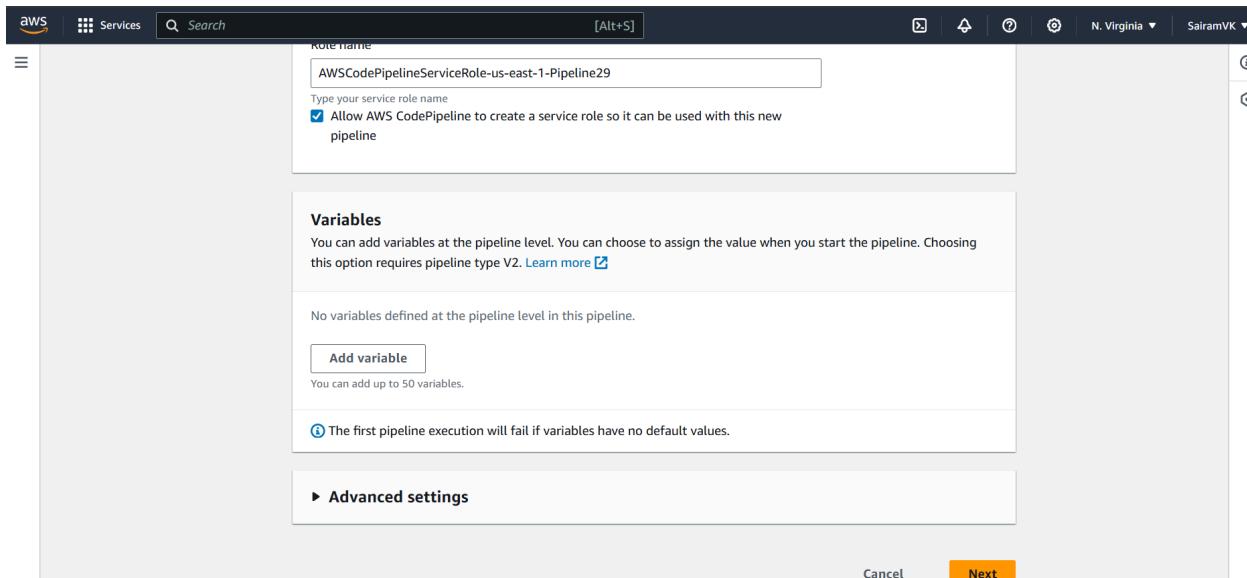


Step 7: Navigate to CodePipeline and click on 'Create Pipeline'.

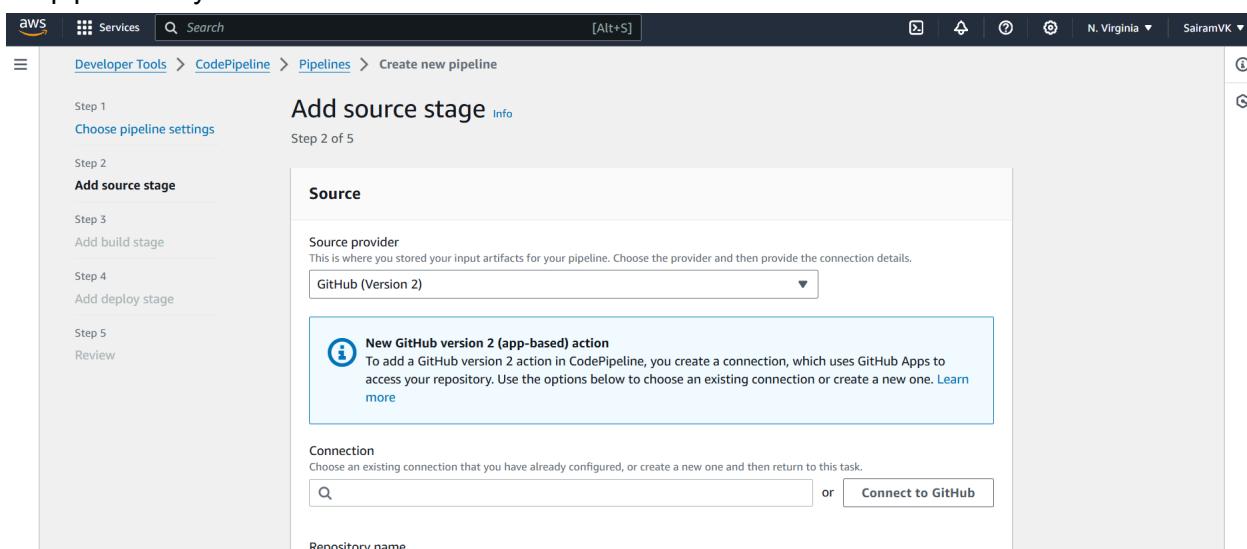


Step 8: Give your pipeline a name. A new service role is also created with the name of the pipeline.





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



Name: Anish Kulkarni

Roll No.: 29

Class: D15C

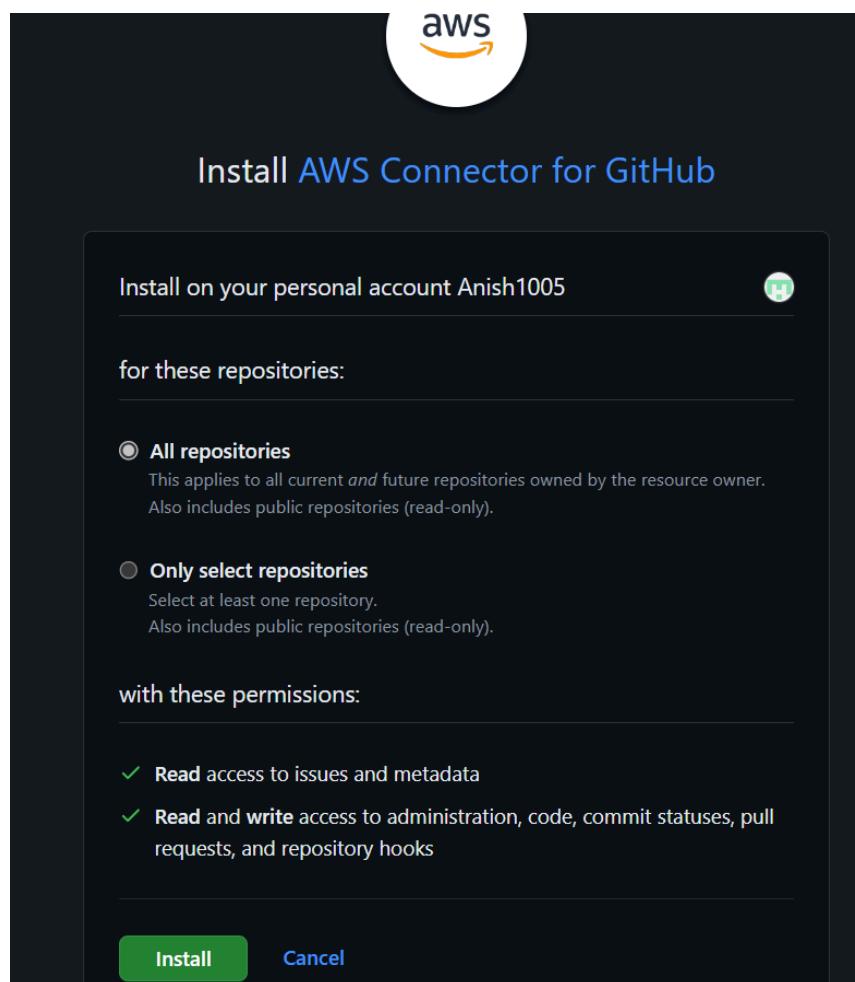
AY: 2024-25

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

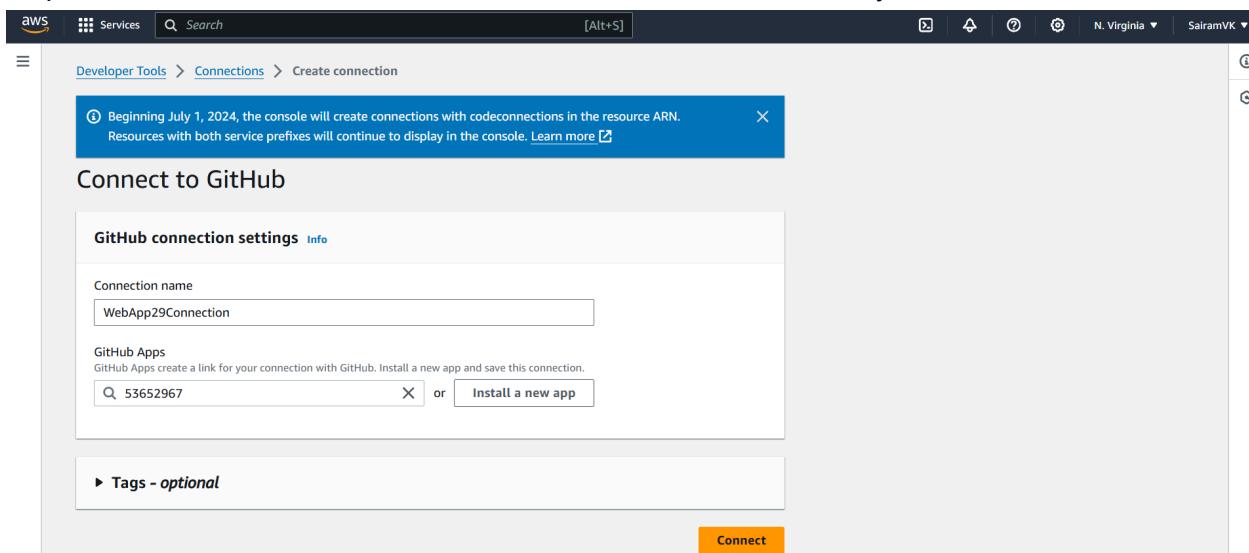
The screenshot shows the 'Create a connection' page for a GitHub App connection. The 'Connection name' field contains 'WebApp29Connection'. A 'Tags - optional' section is present, and a prominent orange 'Connect to GitHub' button is at the bottom right.

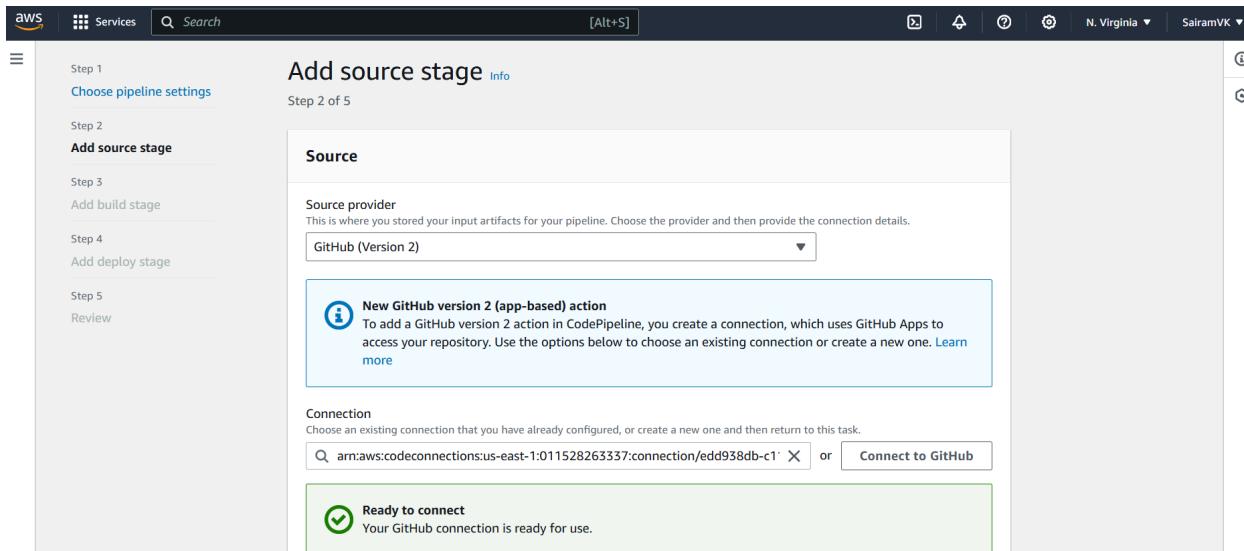
The screenshot shows the 'Create to GitHub' page. It displays a message about code connections and service prefixes. The 'GitHub connection settings' section includes a 'Connection name' field with 'WebApp29Connection', a 'GitHub Apps' section with a search bar and 'Install a new app' button, and a 'Tags - optional' section. A 'Connect' button is located at the bottom right.

Step 11: Install AWS Connector for Github.



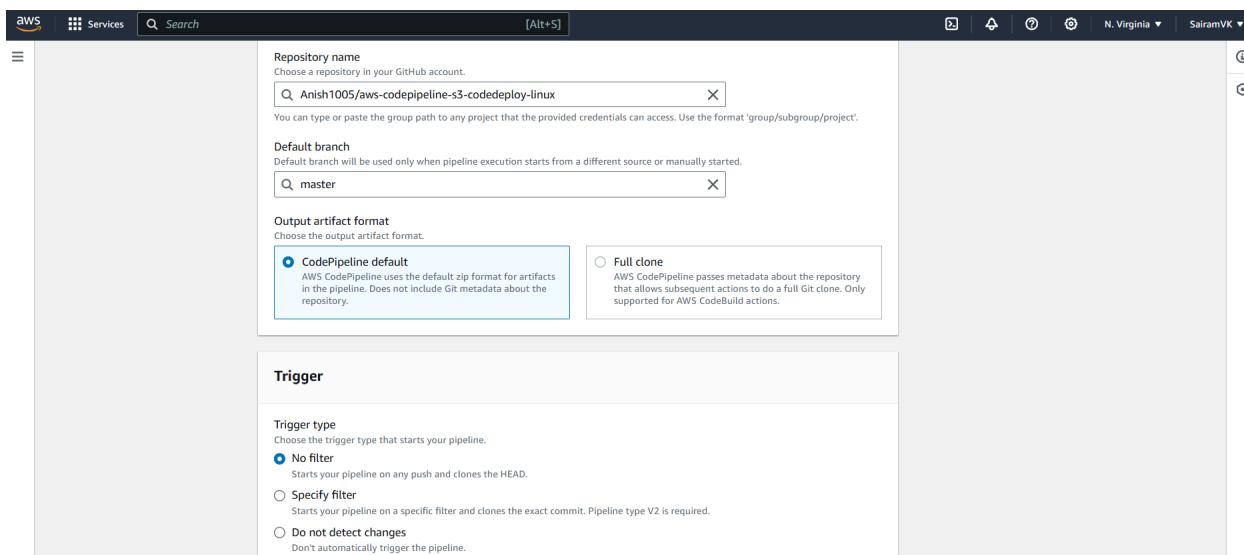
Step 12: Once installation is done, the text field is filled automatically. Click on 'Connect'.





AWS shows that the Github connection is ready to use.

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



Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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

The screenshot shows the AWS CodePipeline 'Create new pipeline' wizard at Step 3 of 5. The current step is 'Add build stage'. On the left sidebar, the steps are listed: Step 1 (Choose pipeline settings), Step 2 (Add source stage), Step 3 (Add build stage), Step 4 (Add deploy stage), and Step 5 (Review). The main panel title is 'Add build stage' with an 'Info' link. Below it, it says 'Step 3 of 5'. A section titled 'Build - optional' contains a 'Build provider' field with a dropdown menu. At the bottom right are 'Cancel', 'Previous', 'Skip build stage' (which is highlighted in orange), and 'Next' buttons.

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

The screenshot shows the AWS CodePipeline 'Create new pipeline' wizard at Step 4 of 5. The current step is 'Add deploy stage'. On the left sidebar, the steps are listed: Step 4 (Add deploy stage), Step 5 (Review). The main panel title is 'Deploy' with a 'Info' link. It shows the 'Deploy provider' set to 'AWS Elastic Beanstalk', 'Region' set to 'US East (N. Virginia)', and 'Input artifacts' set to 'SourceArtifact'. Under 'Application name', the application 'WebApp29' is selected. Under 'Environment name', the environment 'WebApp29-env' is selected. A checkbox for 'Configure automatic rollback on stage failure' is present. At the bottom right are 'Cancel', 'Previous', and 'Next' buttons.

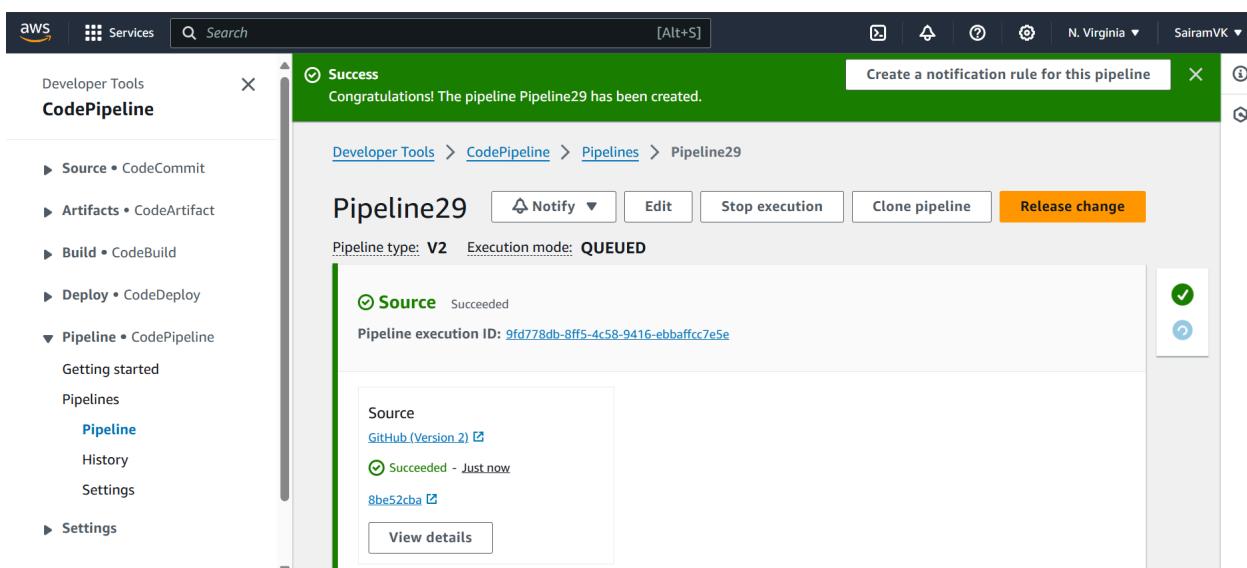
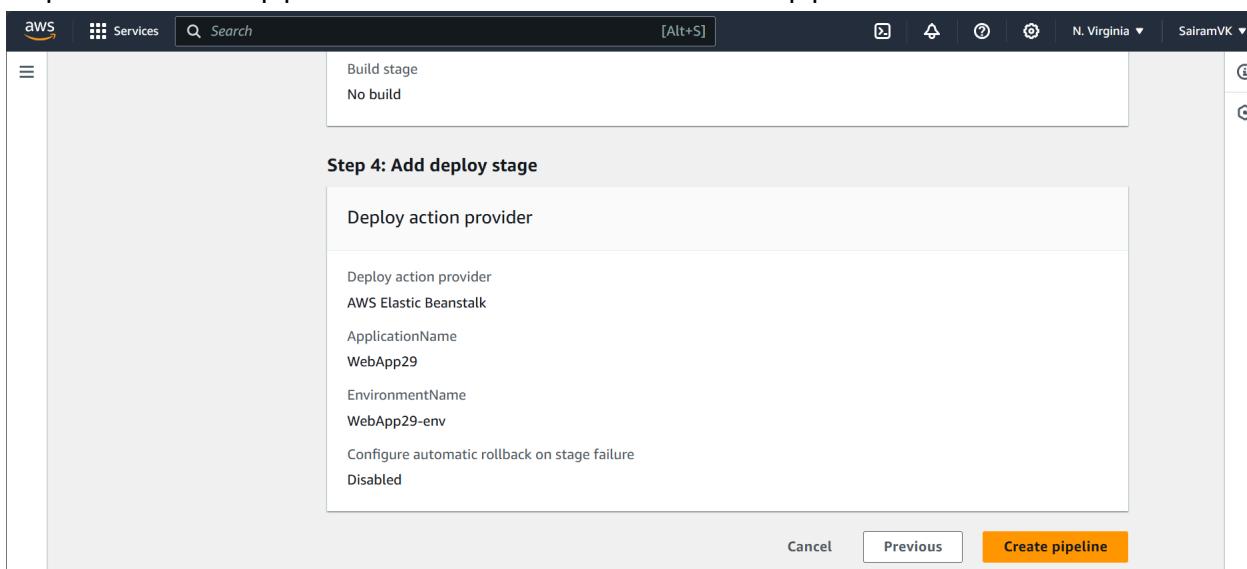
Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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

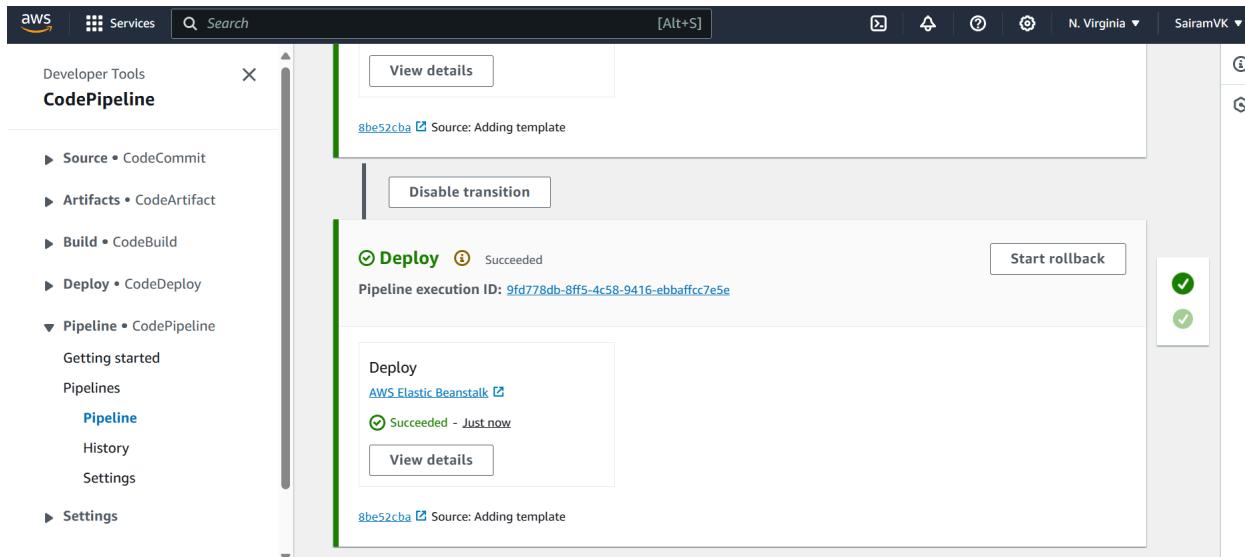


Name: Anish Kulkarni

Roll No.: 29

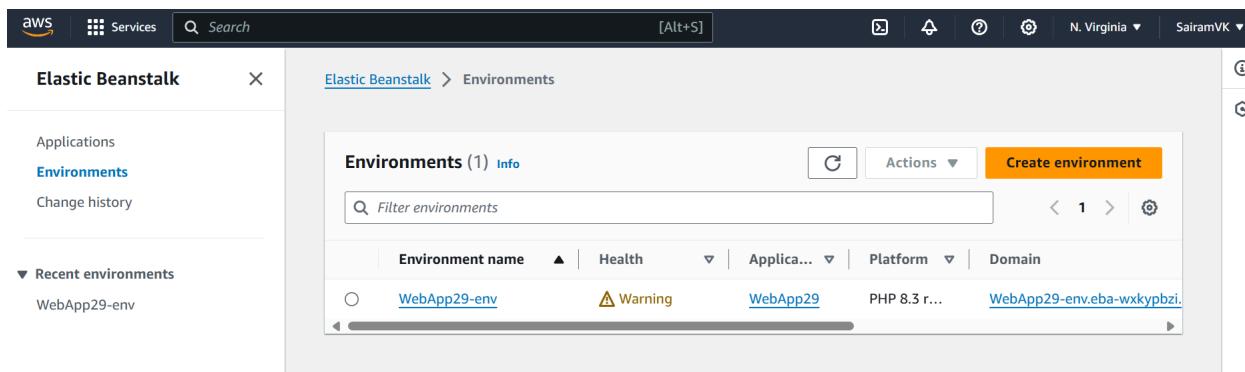
Class: D15C

AY: 2024-25



The pipeline is created and deployment is complete.

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



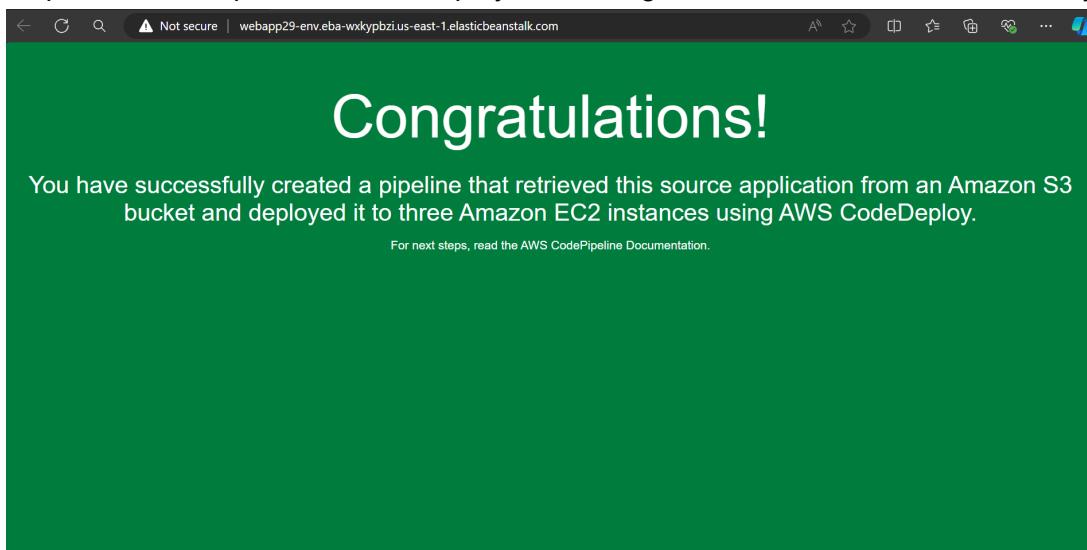
Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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



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

A screenshot of the AWS Elastic Beanstalk console. The left sidebar shows 'Elastic Beanstalk' with sections for Applications, Environments, Change history, Application: WebApp29 (with sub-options for Application versions and Saved configurations), and Environment: WebApp29-env (with sub-options for Go to environment and Configuration). The main content area shows an environment named 'WebApp29-env' with an 'Info' button. A green banner at the top says 'Environment update successfully completed.' Below it is an 'Actions' dropdown and an orange 'Upload and deploy' button. The 'Environment overview' section displays 'Health' (with a warning icon and a 'View causes' link), 'Domain' (WebApp29-env.eba-wxkypbzi.us-east-1.elasticbeanstalk.com), 'Environment ID' (e-mvmyv9pcn), and 'Application name' (WebApp29).

A screenshot of the AWS CloudWatch Pipeline execution details. At the top, it shows a green 'Deploy' status with an 'Succeeded' message and a timestamp '3 minutes ago'. It also shows a 'Pipeline execution ID: f91a7fd1-eace-44ff-88f4-8422e562398a'. There is a 'Start rollback' button. Below this, there's a 'Deploy' section with a 'AWS Elastic Beanstalk' link and a green checkmark indicating success. A 'View details' button is present. At the bottom, it shows a link 'e2507183' with a note 'Source: Update index.html'.

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

Experiment 3

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

Steps:

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

- Master:-

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-081449875d3038e82	Custom TCP	TCP	10251	Custom	<input type="text"/> Delete
sgr-06d97ef62211e84c7	Custom TCP	TCP	10250	Custom	<input type="text"/> Delete
sgr-0e2809bb673050968	SSH	TCP	22	Custom	<input type="text"/> Delete
sgr-09b3626dfaf1a7b59	Custom TCP	TCP	10252	Custom	<input type="text"/> Delete
sgr-0e7326dc18ad6d8c3	HTTP	TCP	80	Custom	<input type="text"/> Delete
sgr-032a27dd795281af9	Custom TCP	TCP	6443	Custom	<input type="text"/> Delete
sgr-0a898585cb6ddb525	All traffic	All	All	Custom	<input type="text"/> Delete

- Nodes:-

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-060b465b5da8996b7	All traffic	All	All	Custom	<input type="text"/> Delete
sgr-0bf297cfdb8a295dd	HTTP	TCP	80	Custom	<input type="text"/> Delete
sgr-06a3cb76ef4953d0f	SSH	TCP	22	Custom	<input type="text"/> Delete
sgr-002e10c3d000136d9	Custom TCP	TCP	10250	Custom	<input type="text"/> Delete
sgr-0349c03ac4e614bf0	Custom TCP	TCP	30000 - 3276	Custom	<input type="text"/> Delete
sgr-0a8fce8b561635d4a	All TCP	TCP	0 - 65535	Custom	<input type="text"/> Delete

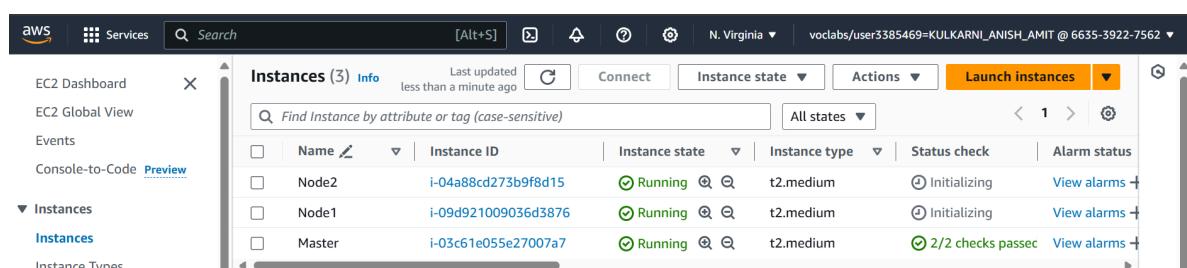
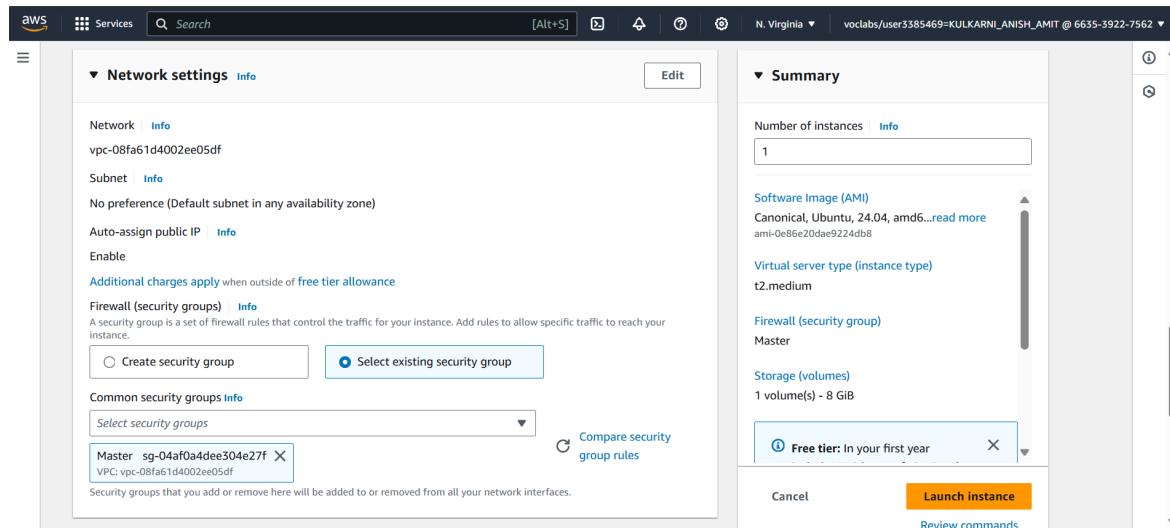
Step 2: Create 3 new EC2 instances in AWS (1 master and 2 nodes). Make sure that you choose Ubuntu as the instance type.

The screenshot shows the AWS Lambda console. At the top, there is a search bar and a 'Create Function' button. Below it, a table lists existing functions: 'Master' (selected), 'HelloWorld', 'LambdaTest', and 'lambdafunction'. The 'Master' row has a 'Edit' button. On the right, there are tabs for 'Overview', 'Code', 'Logs', and 'Metrics'. The 'Logs' tab is currently selected, showing a log entry: 'Function execution started'. The 'Code' tab shows the code for the 'Master' function, which is a simple 'Hello World' example.

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

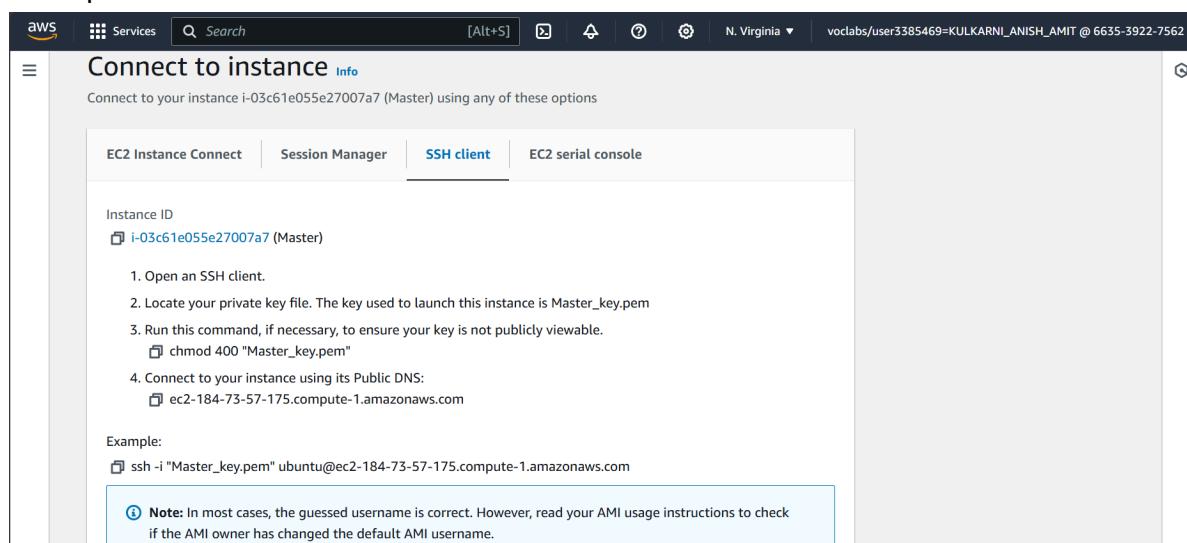
The screenshot shows the AWS EC2 instance creation wizard. The first step is 'Configure Instance Details'. It shows the 'Instance type' section where 't2.medium' is selected. Below it, there is a note: 'Additional costs apply for AMIs with pre-installed software'. To the right, there is a 'Compare instance types' link and a 'All generations' toggle. The second step is 'Configure Volume'. It shows the 'Volume Type' dropdown set to 'Standard' and a 'Create volume' button. The third step is 'Configure Network & Security'. It shows the 'Subnet' dropdown set to 'My Subnet' and a 'Next Step' button.

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



All the instances are created as above.

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



Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

The screenshot shows the AWS CloudWatch Metrics interface. At the top, there's a navigation bar with the AWS logo, a 'Services' dropdown, a search bar, and account information: 'N. Virginia' and 'voclabs/user3385469=KULKARNI_ANISH_AMIT @ 6635-3922-7562'. Below the navigation bar, the main title is 'Connect to instance [Info](#)'. A sub-instruction says 'Connect to your instance i-09d921009036d3876 (Node1) using any of these options'. There are four tabs: 'EC2 Instance Connect', 'Session Manager', 'SSH client' (which is selected), and 'EC2 serial console'. Under the 'SSH client' tab, the 'Instance ID' is listed as 'i-09d921009036d3876 (Node1)'. Below it is a numbered list of steps: 1. Open an SSH client. 2. Locate your private key file. The key used to launch this instance is 'Node_key.pem'. 3. Run this command, if necessary, to ensure your key is not publicly viewable.
chmod 400 "Node_key.pem" 4. Connect to your instance using its Public DNS:
ec2-54-87-91-118.compute-1.amazonaws.com. An example command is provided: ssh -i "Node_key.pem" ubuntu@ec2-54-87-91-118.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.'

The screenshot shows the AWS CloudWatch Metrics interface. At the top, there's a navigation bar with the AWS logo, a 'Services' dropdown, a search bar, and account information: 'N. Virginia' and 'voclabs/user3385469=KULKARNI_ANISH_AMIT @ 6635-3922-7562'. Below the navigation bar, the main title is 'Connect to instance [Info](#)'. A sub-instruction says 'Connect to your instance i-04a88cd273b9f8d15 (Node2) using any of these options'. There are four tabs: 'EC2 Instance Connect', 'Session Manager', 'SSH client' (which is selected), and 'EC2 serial console'. Under the 'SSH client' tab, the 'Instance ID' is listed as 'i-04a88cd273b9f8d15 (Node2)'. Below it is a numbered list of steps: 1. Open an SSH client. 2. Locate your private key file. The key used to launch this instance is 'Node_key.pem'. 3. Run this command, if necessary, to ensure your key is not publicly viewable.
chmod 400 "Node_key.pem" 4. Connect to your instance using its Public DNS:
ec2-54-166-66-100.compute-1.amazonaws.com. An example command is provided: ssh -i "Node_key.pem" ubuntu@ec2-54-166-66-100.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 4: Navigate to the folder where the .pem files for the key-pairs (master and nodes) have been downloaded and open the folder in terminal. Then paste the command copied in Step 3 in the terminal.

```
ubuntu@ip-172-31-94-99:~ System information as of Wed Sep 25 16:48:10 UTC 2024
System load: 0.0      Processes:        113
Usage of /: 22.7% of 6.71GB  Users logged in:    0
Memory usage: 5%          IPv4 address for enX0: 172.31.94.99
Swap usage:  0%
Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

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

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

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

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

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

```
ubuntu@ip-172-31-87-41:~ System information as of Wed Sep 25 16:49:44 UTC 2024
System load: 0.0      Processes:        112
Usage of /: 22.7% of 6.71GB  Users logged in:    0
Memory usage: 5%          IPv4 address for enX0: 172.31.87.41
Swap usage:  0%
Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

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

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

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

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

ubuntu@ip-172-31-87-41:~$ |
```

```

ubuntu@ip-172-31-86-164:~ + | 
System information as of Wed Sep 25 16:50:44 UTC 2024
System load: 0.0 Processes: 115
Usage of /: 22.8% of 6.71GB Users logged in: 0
Memory usage: 8% IPv4 address for enX0: 172.31.86.164
Swap usage: 0%
Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

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

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

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

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

ubuntu@ip-172-31-86-164:~$ |

```

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

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

```

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

```

```

ubuntu@ip-172-31-86-164:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [15.3 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [377 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [81.6 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4528 B]
Get:11 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [270 kB]
Get:12 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [113 kB]
Get:13 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:14 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.1 kB]
Get:15 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [353 kB]
Get:16 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [68.1 kB]

```

```

Get:36 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [353 kB]
Get:37 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [68.1 kB]
Get:38 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 c-n-f Metadata [424 B]
Get:39 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [14.4 kB]
Get:40 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [3608 B]
Get:41 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [212 B]
Get:42 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [532 B]
Get:43 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:44 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [112 B]
Get:45 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [10.6 kB]
Get:46 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [10.8 kB]
Get:47 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [17.6 kB]
Get:48 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1104 B]
Get:49 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [216 B]
Get:50 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:51 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:52 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Fetched 29.1 MB in 4s (6952 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
ubuntu@ip-172-31-86-164:~$ |

```

Step 6: Execute the following command:-

sudo apt-get update

sudo apt-get install -y docker-ce

```

ubuntu@ip-172-31-94-99:~$ sudo apt-get update
sudo apt-get install -y docker-ce
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0
    pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-rootless-extras docker-compose-plugin libltdl7
    libslirp0 pigz slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 142 not upgraded.
Need to get 123 MB of archives.
After this operation, 442 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/main amd64 libltdl7 amd64 2.4.7-7build1 [40.3 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/main amd64 libslirp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/universe amd64 slirp4netns amd64 1.2.1-1build2 [34.9 kB]

```

```

Setting up docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-94-99:~$ |

```

Step 7: Execute the following command:-

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

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

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

Step 8: Execute the following command:-

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

```
ubuntu@ip-172-31-94-99:~$ sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
```

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

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

Step 10: Execute the following command:-

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

```
ubuntu@ip-172-31-94-99:~$ sudo apt-get update  
sudo apt-get install -y kubelet kubeadm kubectl  
sudo apt-mark hold kubelet kubeadm kubectl  
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease  
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease  
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease  
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease  
Hit:6 http://security.ubuntu.com/ubuntu noble-security InRelease  
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb InRelease [1186 B]  
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb Packages [4865 B]  
Fetched 6051 B in 1s (10.7 kB/s)  
Reading package lists... Done  
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.  
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following additional packages will be installed:  
  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 142 not upgraded.  
Need to get 87.4 MB of archives.  
After this operation, 314 MB of additional disk space will be used.  
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/main amd64 conntrack amd64 1:1.4.8-1ubuntu1 [37.9 kB]  
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb cri-tools 1.31.1-1.1 [15.7
```

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

```
Scanning processes...
```

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

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

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

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

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

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

```
kubelet set on hold.
```

```
kubeadm set on hold.
```

```
kubectl set on hold.
```

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

Step 11: Execute the following command:-

sudo systemctl enable --now kubelet

sudo apt-get install -y containerd

```
ubuntu@ip-172-31-94-99:~$ sudo systemctl enable --now kubelet
sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 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 142 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 M]
B]
Fetched 47.2 MB in 1s (80.9 MB/s)
(Reading database ... 68064 files and directories currently installed.)
Removing docker-ce (5:27.3.1-1~ubuntu.24.04-noble) ...
Removing containerd.io (1.7.22-1) ...
Selecting previously unselected package runc.
(Reading database ... 68044 files and directories currently installed.)
Preparing to unpack .../runc_1.1.12-0ubuntu3.1_amd64.deb ...
Unpacking runc (1.1.12-0ubuntu3.1) ...
```

Scanning processes...

Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

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

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

Step 12: Execute the following command:-

sudo mkdir -p /etc/containerd

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

```
ubuntu@ip-172-31-94-99:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

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

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

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

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

[ttrpc]
  address = ""
  gid = 0
  uid = 0
ubuntu@ip-172-31-94-99:~$
```

Step 13: Execute the following command:-

sudo systemctl restart containerd

sudo systemctl enable containerd

sudo systemctl status containerd

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

Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549371367Z" level=info msg=serving...
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549434451Z" level=info msg="Start subscribi...
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549438760Z" level=info msg=serving...
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549478280Z" level=info msg="Start recoverin...
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549534454Z" level=info msg="Start event mon...
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549550069Z" level=info msg="Start snapshots...
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549560056Z" level=info msg="Start cni netwo...
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549567803Z" level=info msg="Start streaming...
Sep 25 18:15:47 ip-172-31-94-99 containerd[6255]: time="2024-09-25T18:15:47.549620324Z" level=info msg="containerd succ...
Sep 25 18:15:47 ip-172-31-94-99 systemd[1]: Started containerd.service - containerd container runtime.
```

Step 14: Execute the following command:-

sudo apt-get install -y socat

```
ubuntu@ip-172-31-94-99:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 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 142 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble amd64 socat amd64 1.8.0.0-4build3 [374 kB]
Fetched 374 kB in 0s (13.8 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68108 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.0-4build3_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.

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

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

```
ubuntu@ip-172-31-94-99:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W0925 18:21:53.860369   6690 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-172-31-94-99 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.94.99]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-94-99 localhost] and IPs [172.31.94.99 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-94-99 localhost] and IPs [172.31.94.99 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file

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

Your Kubernetes control-plane has initialized successfully!

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

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

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

export KUBECONFIG=/etc/kubernetes/admin.conf

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

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

kubeadm join 172.31.94.99:6443 --token tc2id6.v5nr4lyvt5pgi3or \
  --discovery-token-ca-cert-hash sha256-5ef2566ee1e34d82e039485e51aaa179ef58639b1d4fef2cb131f55be51026469
```

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

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

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

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

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

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

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

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

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

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

- Node1:-

```
ubuntu@ip-172-31-87-41:~$ sudo kubeadm join 172.31.94.99:6443 --token tc2id6.v5nr4lyvt5pgi3or \
--discovery-token-ca-cert-hash sha256:5ef2566ee1e34d82e039485e51aaa179f58639b1d4fef2cb131f55be51026469
[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 1.001283532s
[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.
```

- Node2:-

```
ubuntu@ip-172-31-86-164:~$ sudo kubeadm join 172.31.94.99:6443 --token tc2id6.v5nr4lyvt5pgi3or \
--discovery-token-ca-cert-hash sha256:5ef2566ee1e34d82e039485e51aaa179f58639b1d4fef2cb131f55be51026469
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 500.72345ms
[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 19: Run the ‘kubectl get nodes’ command on the Master machine again to check if the Node machines have successfully joined the cluster.

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

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

Step 20: We must add a network plugin to change the status of the nodes to 'Ready':-

```
kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
```

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

Step 21: Execute the following command:-

```
sudo systemctl status kubelet
```

```
ubuntu@ip-172-31-94-99:~$ sudo systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
   Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; preset: enabled)
   Drop-In: /usr/lib/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: active (running) since Wed 2024-09-25 18:22:14 UTC; 8min ago
       Docs: https://kubernetes.io/docs/
   Main PID: 7366 (kubelet)
     Tasks: 10 (limit: 4676)
    Memory: 33.6M (peak: 34.1M)
      CPU: 8.441s
     CGroup: /system.slice/kubelet.service
             └─7366 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/ku
Sep 25 18:30:29 ip-172-31-94-99 kubelet[7366]: I0925 18:30:29.856207    7366 pod_container_deletor.go:80] "Container no
Sep 25 18:30:29 ip-172-31-94-99 kubelet[7366]: I0925 18:30:29.856234    7366 scope.go:117] "RemoveContainer" containerID:>
Sep 25 18:30:29 ip-172-31-94-99 kubelet[7366]: I0925 18:30:29.857797    7366 scope.go:117] "RemoveContainer" containerID:>
Sep 25 18:30:29 ip-172-31-94-99 kubelet[7366]: I0925 18:30:29.869340    7366 scope.go:117] "RemoveContainer" containerID:>
Sep 25 18:30:29 ip-172-31-94-99 kubelet[7366]: I0925 18:30:29.876450    7366 scope.go:117] "RemoveContainer" containerID:>
Sep 25 18:30:33 ip-172-31-94-99 kubelet[7366]: I0925 18:30:33.872760    7366 scope.go:117] "RemoveContainer" containerID:>
Sep 25 18:30:33 ip-172-31-94-99 kubelet[7366]: E0925 18:30:33.872874    7366 pod_workers.go:1301] "Error syncing pod, s>
Sep 25 18:30:39 ip-172-31-94-99 kubelet[7366]: I0925 18:30:39.167638    7366 scope.go:117] "RemoveContainer" containerID:>
Sep 25 18:30:39 ip-172-31-94-99 kubelet[7366]: I0925 18:30:39.167749    7366 pod_workers.go:1301] "Error syncing pod, s>
Sep 25 18:30:39 ip-172-31-94-99 kubelet[7366]: I0925 18:30:39.882133    7366 scope.go:117] "RemoveContainer" containerID:>
ubuntu@ip-172-31-94-99:~$
```

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

NAME	STATUS	ROLES	AGE	VERSION
ip-172-31-86-164	Ready	<none>	3m37s	v1.31.1
ip-172-31-87-41	Ready	<none>	4m16s	v1.31.1
ip-172-31-94-99	Ready	control-plane	9m20s	v1.31.1

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

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

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

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

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

```
ubuntu@ip-172-31-94-99:~$ kubectl get nodes
NAME           STATUS    ROLES      AGE     VERSION
ip-172-31-86-164   Ready    Node2      7m58s   v1.31.1
ip-172-31-87-41   Ready    Node1      8m37s   v1.31.1
ip-172-31-94-99   Ready    control-plane 13m    v1.31.1
```

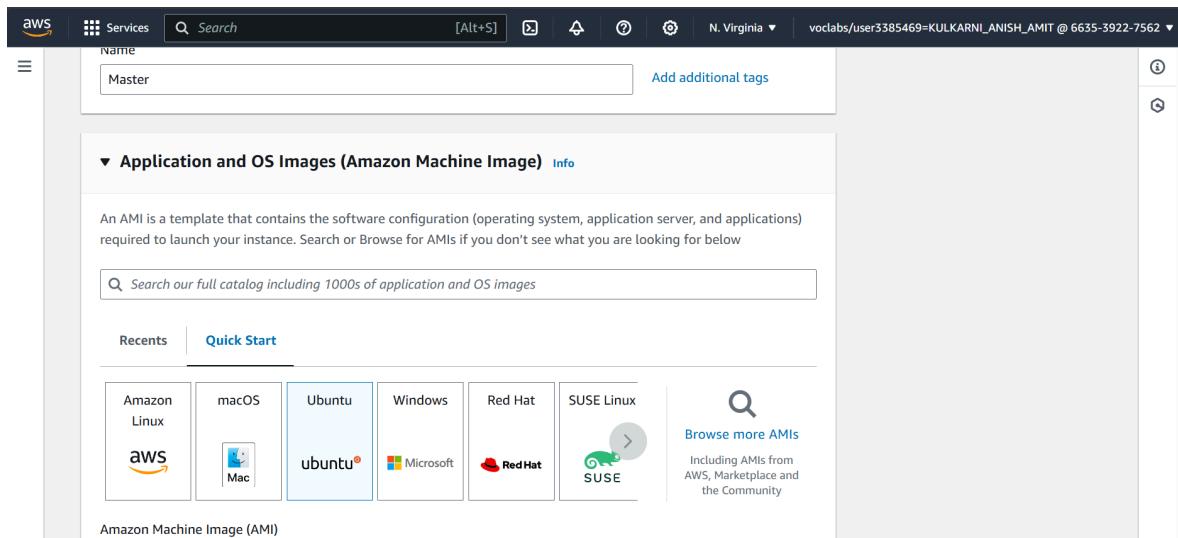
Conclusion: In this experiment, we learned about the Kubernetes cluster architecture and how to install and spin up a Kubernetes cluster on Linux Machines/Cloud Platforms. First, we created 3 EC2 Ubuntu instances on AWS (1 master and 2 worker nodes) and connected them to our local terminal using SSH.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

Experiment 4

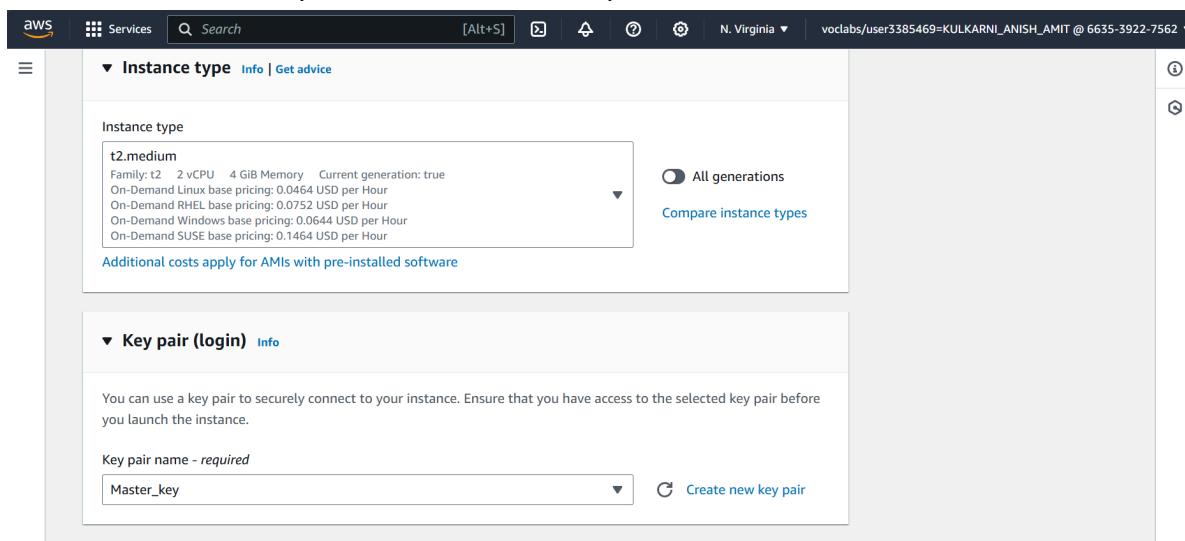
Aim: To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

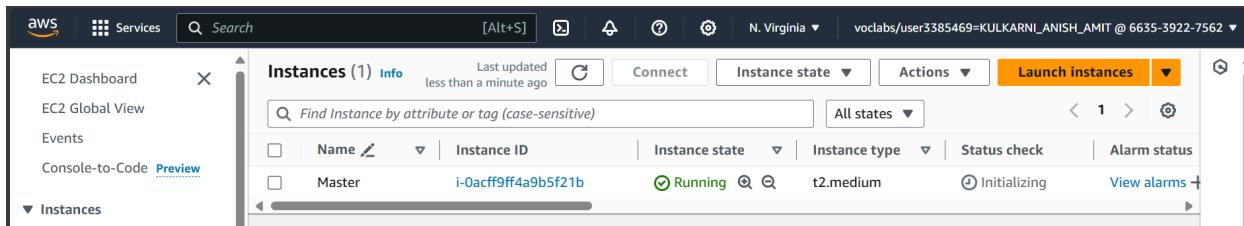
Steps:

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

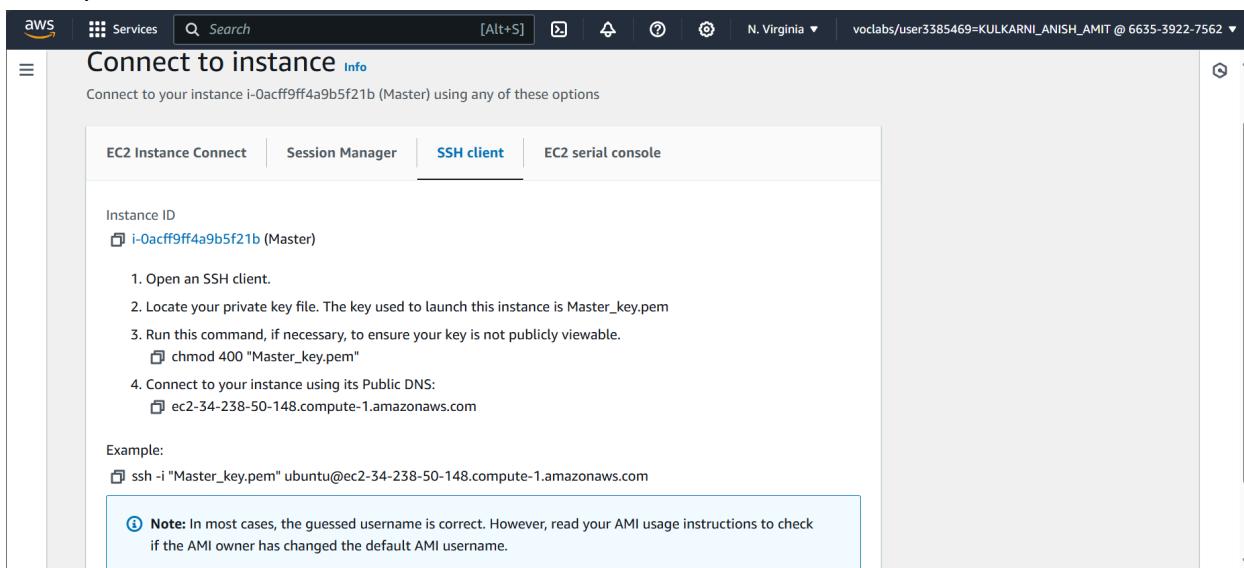


Under 'Instance type', choose t2.medium because the default (t2.micro) does not provide the sufficient resources required to execute this experiment.





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



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

```
PS C:\Users\anish\Downloads> ssh -i "Master_key.pem" ubuntu@ec2-34-238-50-148.compute-1.amazonaws.com
The authenticity of host 'ec2-34-238-50-148.compute-1.amazonaws.com (34.238.50.148)' can't be established.
ED25519 key fingerprint is SHA256:A8tEafucdELWuW++SLCL56FSjVGBhjT08vPXhtySIL4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-34-238-50-148.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

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

System information as of Wed Sep 25 19:44:47 UTC 2024

 System load:  0.2          Processes:      117
 Usage of /:   22.8% of 6.71GB  Users logged in:    0
 Memory usage: 6%           IPv4 address for enX0: 172.31.94.127
 Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

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

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

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

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

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

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

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

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

```
ubuntu@ip-172-31-94-127:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Get:5 https://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 https://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [15.3 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [531 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 (7834 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
ubuntu@ip-172-31-94-127:~$
```

Step 5: Execute the following command:-

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

```
ubuntu@ip-172-31-94-127:~$ sudo apt-get update
sudo apt-get install -y docker-ce
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0
  pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7
  libslirp0 pigz slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 142 not upgraded.
Need to get 123 MB of archives.
After this operation, 442 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [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]
```

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

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

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

Step 6: Execute the following command:-

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

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

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

Step 7: Execute the following command:-

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

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

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

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

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

Step 9: Execute the following command:-

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

```
ubuntu@ip-172-31-94-127:~$ sudo apt-get update  
sudo apt-get install -y kubelet kubeadm kubectl  
sudo apt-mark hold kubelet kubeadm kubectl  
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease  
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease  
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease  
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease  
Hit:5 https://download.docker.com/linux/ubuntu noble InRelease  
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb InRelease [1186 B]  
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb Packages [4865 B]  
Fetched 6051 B in 1s (12.1 kB/s)  
Reading package lists... Done  
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.  
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following additional packages will be installed:  
  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 142 not upgraded.  
Need to get 87.4 MB of archives.  
After this operation, 314 MB of additional disk space will be used.  
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/main amd64 conntrack amd64 1:1.4.8-1ubuntu1 [37.9 kB]  
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb cri-tools 1.31.1-1.1 [15.7 MB]  
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb kubeadm 1.31.1-1.1 [11.4 M]  
Setting up kubeadm (1.31.1-1.1) ...  
Setting up kubelet (1.31.1-1.1) ...  
Processing triggers for man-db (2.12.0-4build2) ...  
Scanning processes...  
Scanning linux images...  
  
Running kernel seems to be up-to-date.  
  
No services need to be restarted.  
  
No containers need to be restarted.  
  
No user sessions are running outdated binaries.  
  
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
kubelet set on hold.  
kubeadm set on hold.  
kubectl set on hold.  
ubuntu@ip-172-31-94-127:~$ |
```

Step 10: Execute the following command:-

```
sudo systemctl enable --now kubelet
sudo kubeadm init --pod-network-cidr=10.244.0.0/16
```

```
ubuntu@ip-172-31-94-127:~$ sudo systemctl enable --now kubelet
sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
W0925 19:58:21.457250      4200 checks.go:1080] [preflight] WARNING: Couldn't create the interface used for talking to the
container runtime: failed to create new CRI runtime service: validate service connection: validate CRI v1 runtime API f
or endpoint "unix:///var/run/containerd/containerd.sock": rpc error: code = Unimplemented desc = unknown service runtime
.v1.RuntimeService
          [WARNING FileExisting-socat]: socat not found in system path
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
error execution phase preflight: [preflight] Some fatal errors occurred:
failed to create new CRI runtime service: validate service connection: validate CRI v1 runtime API for endpoint "unix://
/var/run/containerd/containerd.sock": rpc error: code = Unimplemented desc = unknown service runtime.v1.RuntimeService[p
reflight] If you know what you are doing, you can make a check non-fatal with '--ignore-preflight-errors=...'
To see the stack trace of this error execute with --v=5 or higher
```

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

Step 11: Execute the following command:-

```
sudo apt-get install -y containerd
```

```
ubuntu@ip-172-31-94-127:~$ sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin 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 142 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 M
B]
Fetched 47.2 MB in 1s (63.2 MB/s)
(Reading database ... 68064 files and directories currently installed.)
Removing docker-ce (5:27.3.1-1~ubuntu.24.04-noble) ...
Removing containerd.io (1.7.22-1) ...
Selecting previously unselected package runc.
(Reading database ... 68044 files and directories currently installed.)
Preparing to unpack .../runc_1.1.12-0ubuntu3.1_amd64.deb ...
Unpacking runc (1.1.12-0ubuntu3.1) ...
Selecting previously unselected package containerd.
```

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

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

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

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

Step 12: Execute the following command:-

sudo mkdir -p /etc/containerd

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

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

[cgroup]
  path = ""

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

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

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

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

[ttrpc]
  address = ""
  gid = 0
  uid = 0
ubuntu@ip-172-31-94-127:~$ |
```

Step 13: Execute the following command:-

sudo systemctl restart containerd

sudo systemctl enable containerd

sudo systemctl status containerd

```
ubuntu@ip-172-31-94-127:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd
● containerd.service - containerd container runtime
   Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
   Active: active (running) since Wed 2024-09-25 20:01:08 UTC; 229ms ago
     Docs: https://containerd.io
     Main PID: 4650 (containerd)
        Tasks: 7
       Memory: 13.4M (peak: 14.1M)
         CPU: 57ms
        CGroup: /system.slice/containerd.service
                  └─4650 /usr/bin/containerd

Sep 25 20:01:08 ip-172-31-94-127 containerd[4650]: time="2024-09-25T20:01:08.945603464Z" level=error msg="failed to load configuration file /etc/containerd/config.toml: failed to open file: permission denied"
Sep 25 20:01:08 ip-172-31-94-127 containerd[4650]: time="2024-09-25T20:01:08.945800710Z" level=info msg="Start subscriber"
Sep 25 20:01:08 ip-172-31-94-127 containerd[4650]: time="2024-09-25T20:01:08.945848260Z" level=info msg="Start recovery"
Sep 25 20:01:08 ip-172-31-94-127 containerd[4650]: time="2024-09-25T20:01:08.945895860Z" level=info msg="Start event manager"
Sep 25 20:01:08 ip-172-31-94-127 containerd[4650]: time="2024-09-25T20:01:08.945905096Z" level=info msg="Start snapshotter"
Sep 25 20:01:08 ip-172-31-94-127 containerd[4650]: time="2024-09-25T20:01:08.945913073Z" level=info msg="Start cni network"
Sep 25 20:01:08 ip-172-31-94-127 containerd[4650]: time="2024-09-25T20:01:08.945920246Z" level=info msg="Start streaming"
Sep 25 20:01:08 ip-172-31-94-127 containerd[4650]: time="2024-09-25T20:01:08.945812786Z" level=info msg="serving... address"
Sep 25 20:01:08 ip-172-31-94-127 containerd[4650]: time="2024-09-25T20:01:08.946023718Z" level=info msg="serving... address"
Sep 25 20:01:08 ip-172-31-94-127 containerd[4650]: time="2024-09-25T20:01:08.947101430Z" level=info msg="containerd successfully started"
ubuntu@ip-172-31-94-127:~$
```

Step 14: Execute the following command:-

sudo apt-get install -y socat

```
ubuntu@ip-172-31-94-127:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 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 142 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.0.0-4build3 [374 kB]
Fetched 374 kB in 0s (17.0 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68108 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.0-4build3_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.
```

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

Step 15: Initialise the kubernetes cluster. To do so, execute the following command:-

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

```
ubuntu@ip-172-31-94-127:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W0925 20:04:25.033576    4946 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that used by kubeadm. It is recommended to use "registry.k8s.io/pause:3.10" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-94-127 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.94.127]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-94-127 localhost] and IPs [172.31.94.127 127.0.0.1 :1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-94-127 localhost] and IPs [172.31.94.127 127.0.0.1 :1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file

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

Your Kubernetes control-plane has initialized successfully!

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

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

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

export KUBECONFIG=/etc/kubernetes/admin.conf

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

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

kubeadm join 172.31.94.127:6443 --token lv3lsh.7tuddvu59m0xzzv4 \
  --discovery-token-ca-cert-hash sha256:82da7c0275294ee855fa49156f1b059216830c3f1ac6501184759c6da82277f1
ubuntu@ip-172-31-94-127:~$ |
```

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

mkdir -p \$HOME/.kube

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

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

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

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

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

kubectl apply -f

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

!

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

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

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

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

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

NAME	READY	STATUS	RESTARTS	AGE
nginx-deployment-d556bf558-cr6j8	0/1	Pending	0	36s
nginx-deployment-d556bf558-q5kng	0/1	Pending	0	36s

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

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

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

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

Step 21: Execute the following command:-

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

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

This changes the status of the pod to ‘Running’.

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

NAME	STATUS	ROLES	AGE	VERSION
ip-172-31-94-127	Ready	control-plane	9m38s	v1.31.1

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

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

Step 24: Execute the following command again:-

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

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

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

```
PS C:\Users\anish\Downloads> ssh -i "Master_key.pem" ubuntu@ec2-34-238-50-148.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 Wed Sep 25 20:22:30 UTC 2024

System load: 0.0           Processes:          154
Usage of /: 55.4% of 6.71GB  Users logged in:    1
Memory usage: 19%           IPv4 address for enX0: 172.31.94.127
Swap usage:  0%

Expanded Security Maintenance for Applications is not enabled.

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

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

Last login: Wed Sep 25 19:44:48 2024 from 182.48.210.206
```

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

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

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

Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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

Experiment No. 5

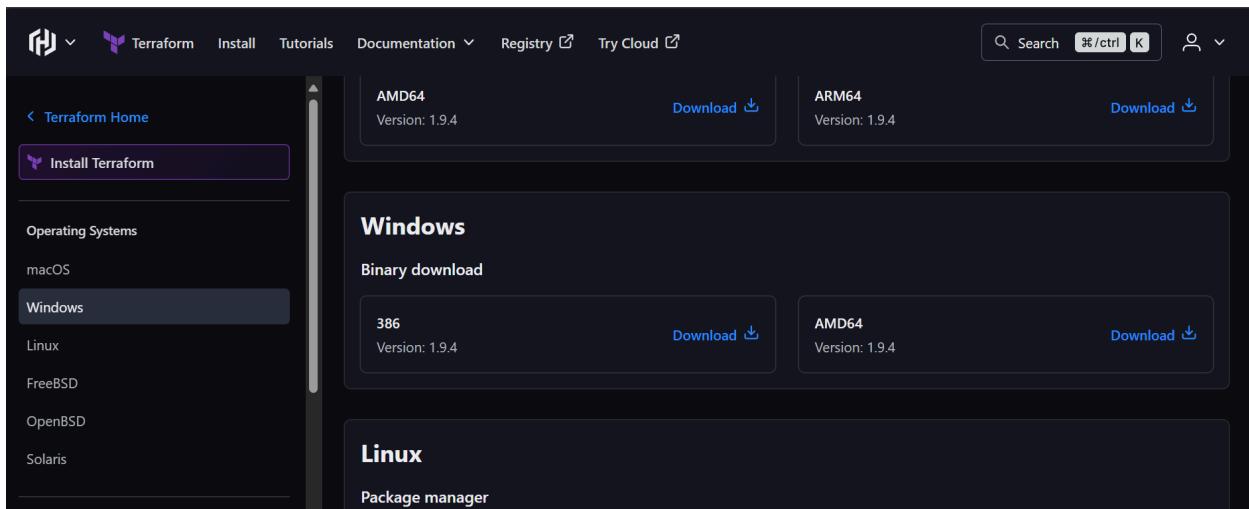
Aim: To understand Terraform lifecycle, core concepts/terminologies and install it on a Linux machine and Windows.

Steps:-

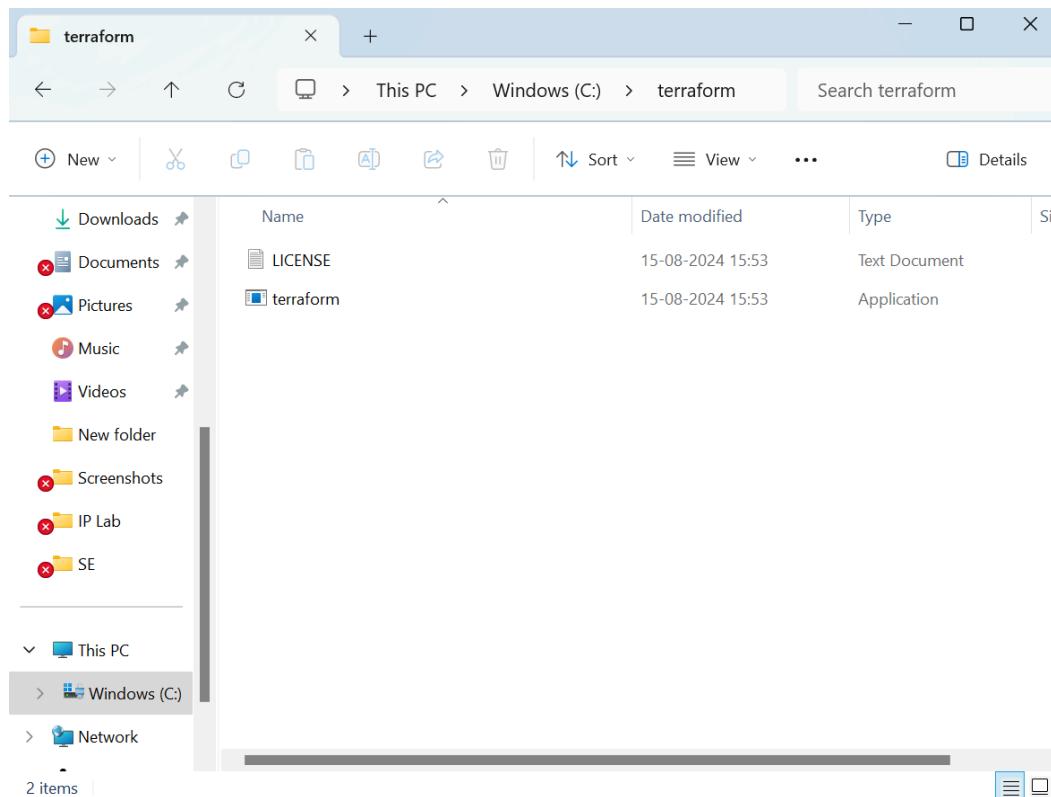
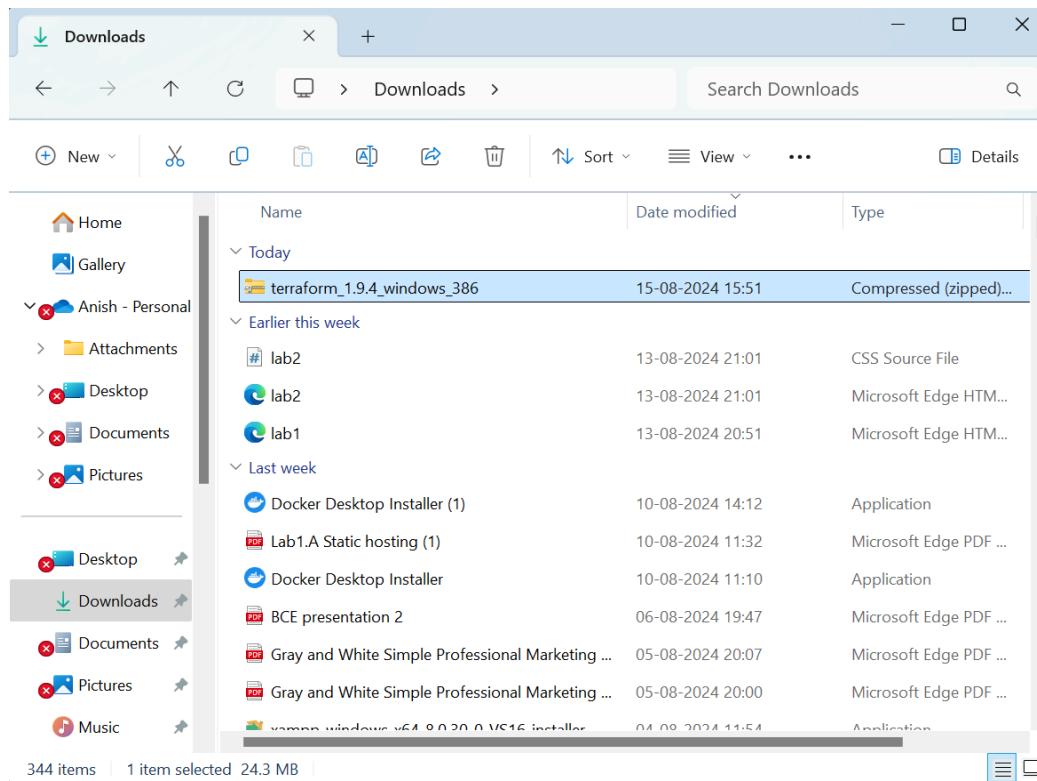
Step 1: Install Terraform from Terraform's official website:

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

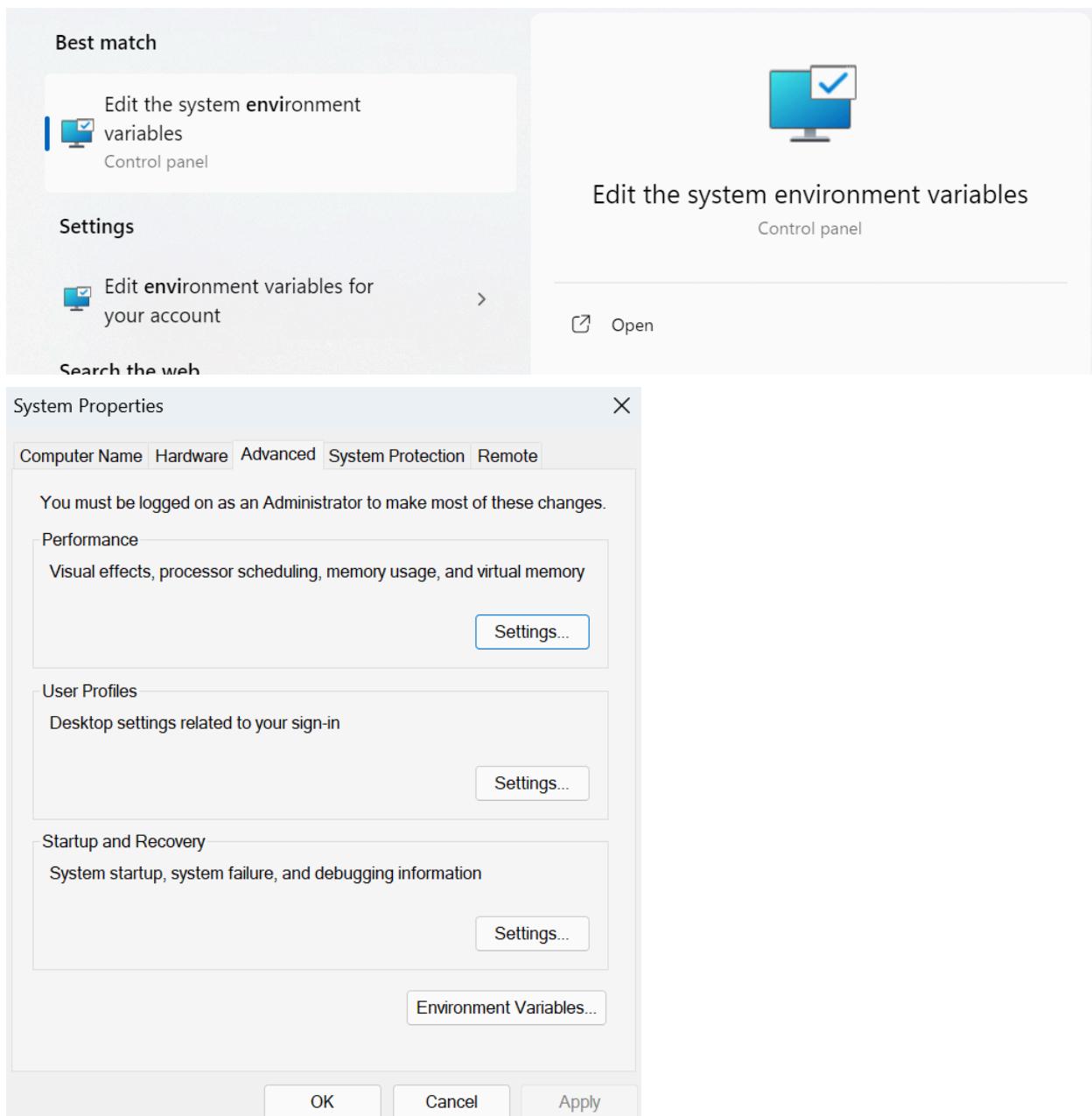
Select your operating system followed by either 32bit or 64bit depending on your OS type.



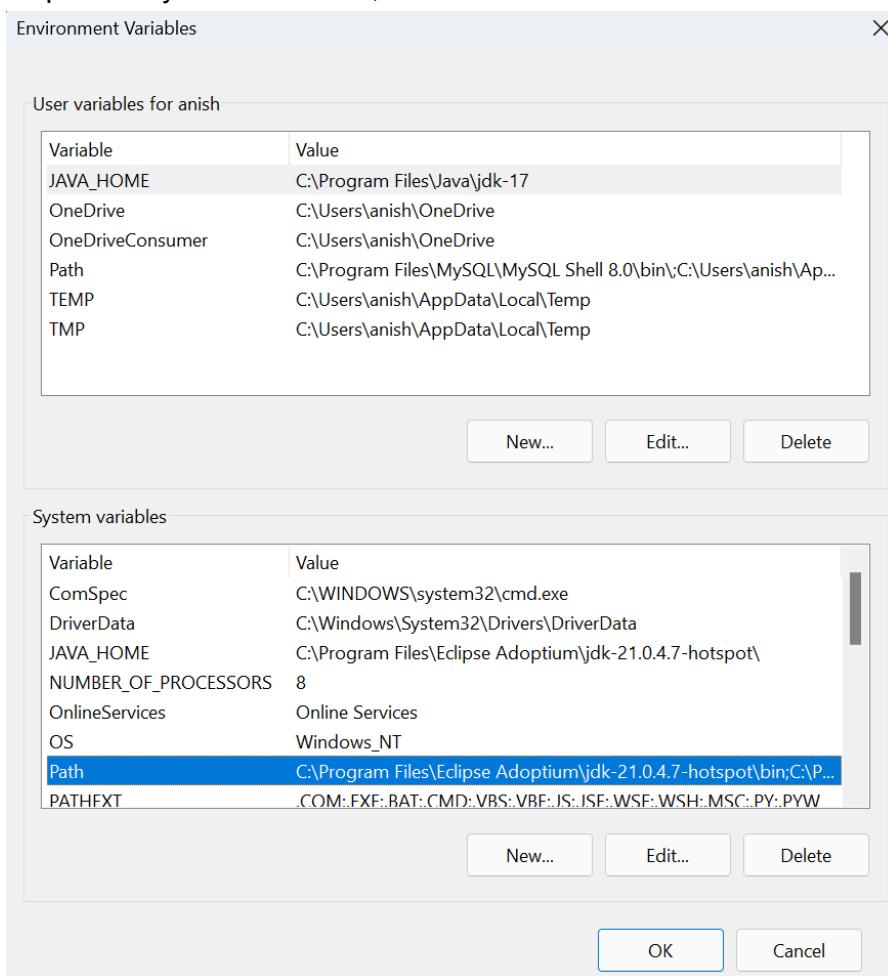
Step 2: Once the download is complete, navigate to the downloaded zipped terraform folder and extract the folder (including the setup file Terraform.exe) into 'C:\Terraform' directory.



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



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



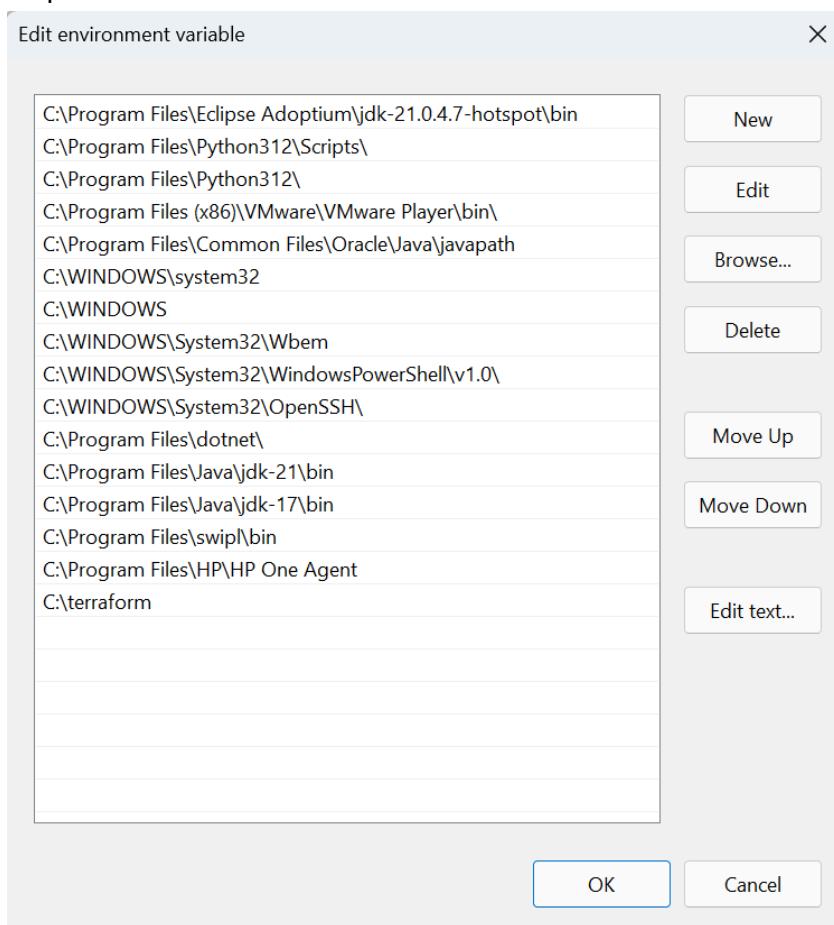
Name: Anish Kulkarni

Roll No.: 29

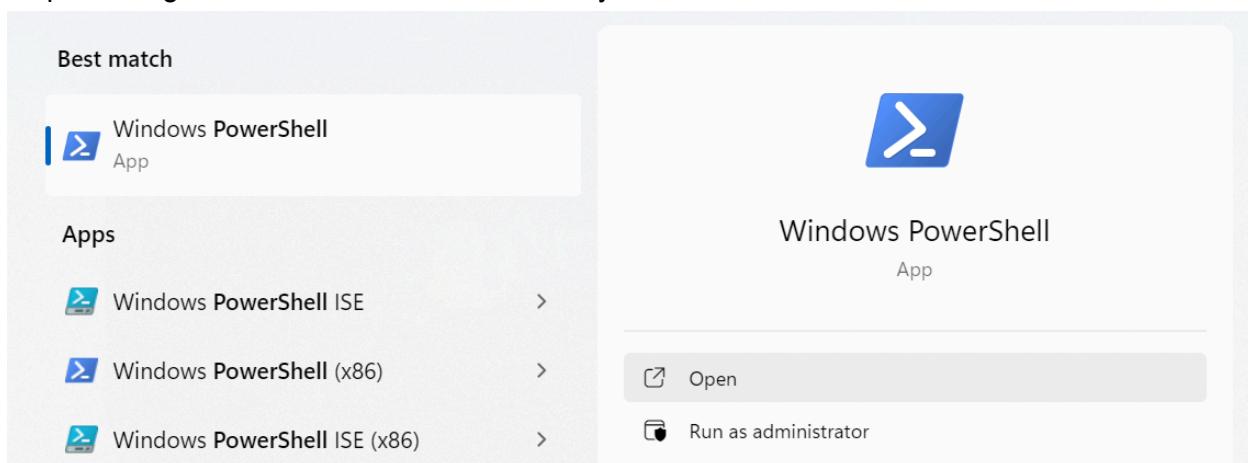
Class: D15C

AY: 2024-25

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



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



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

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

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

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

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

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

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

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

Experiment 6

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

Steps:-

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

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

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

PS C:\Users\anish> docker
Usage: docker [OPTIONS] COMMAND
      A self-sufficient runtime for containers

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

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

Swarm Commands:
  swarm   Manage Swarm

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

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

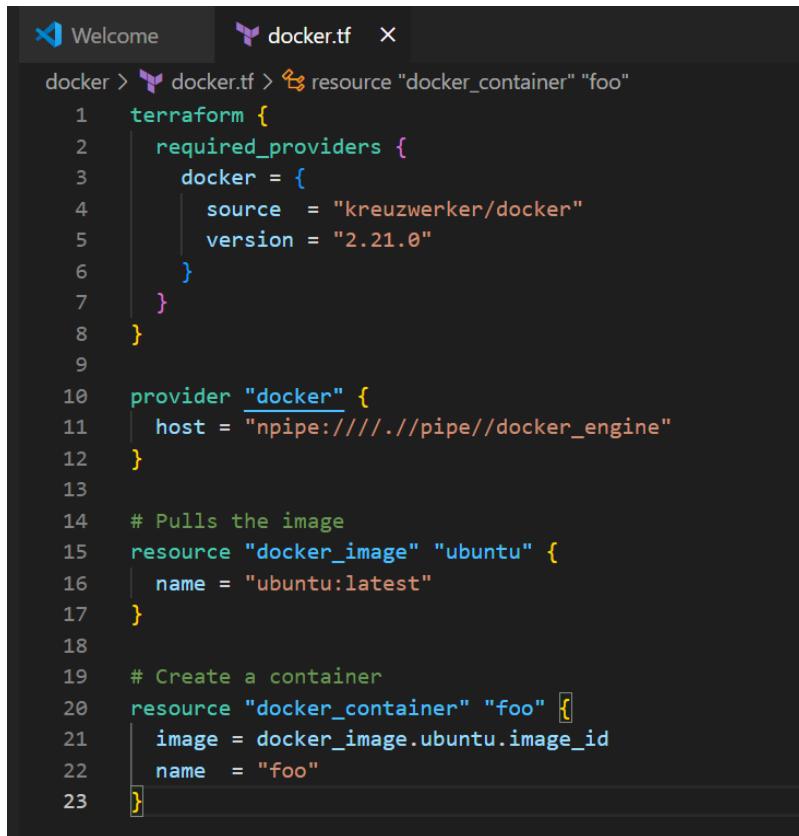
Step 2: Create a folder named ‘Terraform Scripts’. Create a folder named ‘Docker’ inside of the ‘Terraform Scripts’ folder and create a new file named docker.tf in this folder. Write the following in the ‘docker.tf’ file (creates a container):-

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

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

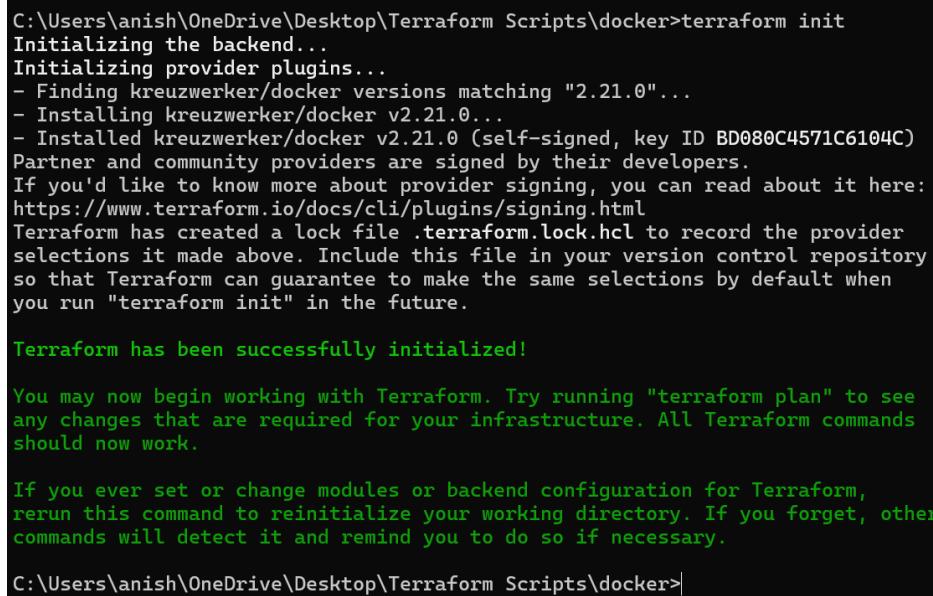
resource "docker_image" "ubuntu" {
  name = "ubuntu:latest"
}

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



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

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



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

Terraform has been successfully initialized!

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

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

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

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

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

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

Terraform will perform the following actions:

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

    + stdin_open          = false
    + stop_signal          = (known after apply)
    + stop_timeout          = (known after apply)
    + tty                  = false

    + healthcheck (known after apply)

    + labels (known after apply)
}

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

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

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

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

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

```
Error: container exited immediately

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

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

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

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

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

```
C:\Users\anish\OneDrive\Desktop\Terraform Scripts\docker>terraform apply
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create
Terraform will perform the following actions:

# docker_container.foo will be created
+ resource "docker_container" "foo" {
  + attach          = false
  + bridge          = (known after apply)
  + command         = [
    + "sleep",
    + "infinity",
  ]
  + container_logs = (known after apply)
  + entrypoint      = (known after apply)
  + env             = (known after apply)
  + exit_code       = (known after apply)
  + gateway         = (known after apply)
  + hostname        = (known after apply)
  + id              = (known after apply)
  + image           = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a"
  + init            = (known after apply)
  + ip_address      = (known after apply)
  + ip_prefix_length = (known after apply)
  + ipc_mode        = (known after apply)
  + log_driver      = (known after apply)
  + logs            = false
  + must_run        = true
  + name            = "foo"
  + network_data    = (known after apply)
  + read_only       = false
  + remove_volumes = true
  + restart         = "no"
  + rm              = false
  + runtime         = (known after apply)
```

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

+ 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.

Enter a value: yes

docker_container.foo: Creating...
docker_container.foo: Creation complete after 1s [id=08340de94d5fd1a9a5ad8df081c97602b4bd75d707432d303147cf62839d3049]

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

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

- Docker images before executing ‘terraform apply’:-

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

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

- Docker images after executing ‘terraform apply’:-

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

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

```
C:\Users\anish\OneDrive\Desktop\Terraform Scripts\docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=08340de94d5fd1a9a5ad8df081c97602b4bd75d707432d303147cf62839d3049]

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       = 0 -> null
    - dns              = [] -> null
    - dns_opts         = [] -> null
    - dns_search       = [] -> null
    - entrypoint       = [] -> null
    - env              = [] -> null
    - gateway          = "172.17.0.1" -> null
    - group_add        = [] -> null
    - hostname         = "08340de94d5fd1a9a5ad8df081c97602b4bd75d707432d303147cf62839d3049" -> null
    - id               = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
    - image             = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a"
    - init              = false -> null
    - ip_address        = "172.17.0.2" -> null
    - ip_prefix_length = 16 -> null
    - ipc_mode          = "private" -> null
    - links             = [] -> null
    - log_driver         = "json-file" -> null
    - log_opts           = {} -> null
    - logs              = false -> null
    - max_retry_count   = 0 -> null
    - memory             = 0 -> null
    - memory_swap        = 0 -> null
    - must_run           = true -> null
    - name              = "foo" -> null
}

- network_data      = [
    - {
        - gateway          = "172.17.0.1"
        - global_ipv6_prefix_length = 0
        - ip_address        = "172.17.0.2"
        - ip_prefix_length  = 16
        - network_name      = "bridge"
        # (2 unchanged attributes hidden)
    },
    ] -> null
- network_mode      = "bridge" -> null
- privileged         = false -> null
- publish_all_ports = false -> null
- read_only          = false -> null
- remove_volumes    = true -> null
- restart            = "no" -> null
- rm                 = false -> null
- runtime             = "runc" -> null
- security_opts      = [] -> null
- shm_size            = 64 -> null
- start              = true -> null
- stdin_open          = false -> null
- stop_timeout        = 0 -> null
- storage_opts        = {} -> null
- sysctls             = {} -> null
- tmpfs               = {} -> null
- tty                 = false -> null
    # (8 unchanged attributes hidden)
}

# docker_image.ubuntu will be destroyed
- resource "docker_image" "ubuntu" {
    - id               = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest" -> null
    - image_id         = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
    - latest            = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
    - name              = "ubuntu:latest" -> null
    - repo_digest       = "ubuntu@sha256:a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e34a9ab63ee" -> 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=08340de94d5fd1a9a5ad8df081c97602b4bd75d707432d303147cf62839d3049]
docker_container.foo: Destruction complete after 1s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image.ubuntu: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
```

Docker images after executing the ‘terraform destroy’ command:-

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

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

Experiment 7

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

Steps:

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

```
PS C:\Users\anish\OneDrive\Desktop\Adv DevOps 7> docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest

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

Step 2: Execute the following command:

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

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

```
PS C:\Users\anish\OneDrive\Desktop\Adv DevOps 7> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000
0 sonarqube:latest
dce67335909e42d81ec64d3ef0c5e5e2c36cc7ed36d87088033121ae1544f4fb
```

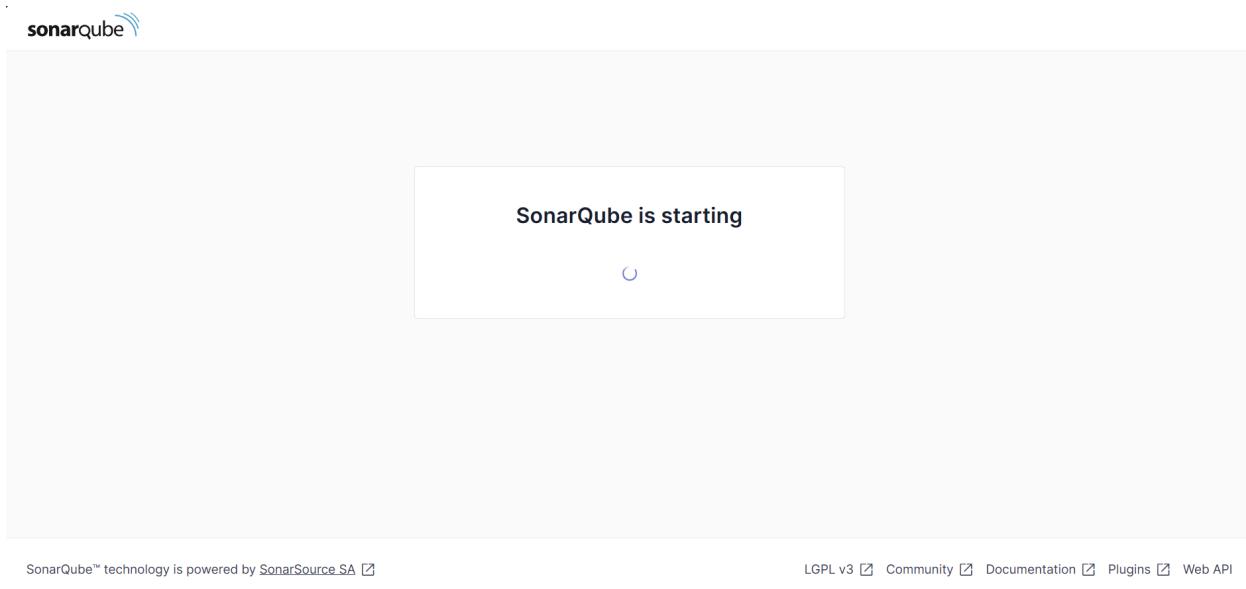
Name: Anish Kulkarni

Roll No.: 29

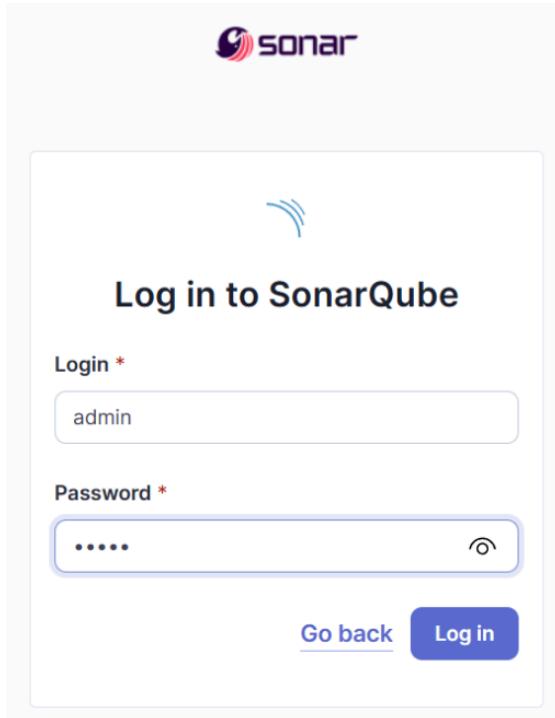
Class: D15C

AY: 2024-25

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



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



Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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

The screenshot shows the SonarQube dashboard with the 'Create a local project' button highlighted. Below it, there are sections for importing from various platforms like Azure DevOps, Bitbucket Cloud, Bitbucket Server, GitHub, and GitLab, each with a 'Setup' button. A note at the bottom says 'Are you just testing or have an advanced use-case? Create a local project.' with a corresponding button.

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

1 of 2

Create a local project

Project display name *

sonarqubetest29



Project key *

sonarqubetest29



Main branch name *

main

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

[Cancel](#)

[Next](#)

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

2 of 2

Set up project for Clean as You Code

The new code definition 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

Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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

The screenshot shows the Jenkins Plugins page. The search bar at the top contains the text "sonarqu". A list of plugins is displayed, with "SonarQube Scanner" being the first result. The plugin details show it is version 2.17.2, released 6 months and 29 days ago. It is described as allowing an easy integration of SonarQube for continuous inspection of code quality. There is an "Install" button next to the plugin entry.

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

The screenshot shows the Jenkins System configuration page under the ‘SonarQube installations’ section. It includes fields for 'Name' (set to 'sonarqube29'), 'Server URL' (set to 'http://localhost:9000'), and 'Server authentication token' (a dropdown menu currently showing '- none -'). At the bottom are 'Save' and 'Apply' buttons.

Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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

The screenshot shows the 'SonarQube Scanner installations' configuration page. At the top, there's a header with 'SonarQube Scanner installations ^' and 'Edited'. Below it, a button 'Add SonarQube Scanner' is visible. A section titled 'SonarQube Scanner' contains a 'Name' field with 'scanner29' and a checked 'Install automatically' checkbox. A nested section 'Install from Maven Central' shows a 'Version' field with 'SonarQube Scanner 6.1.0.4477'. At the bottom are 'Save' and 'Apply' buttons.

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

The screenshot shows the 'Create New Item' dialog in Jenkins. The title bar says 'Jenkins'. The main area has a heading 'Enter an item name' with a text input field containing 'SonarqubeProject29'. Below it, a note says '» Required field'. Three project types are listed: 'Freestyle project' (selected), 'Maven project', and 'Pipeline'. Each type has a description and an 'OK' button. The 'Freestyle project' description includes: 'Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.' The 'Maven project' description includes: 'Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.' The 'Pipeline' description includes: 'Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) organizing complex activities that do not easily fit in free-style job type.'

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

https://github.com/shazforiot/MSBuild_firstproject

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

Source Code Management

None

Git [?](#)

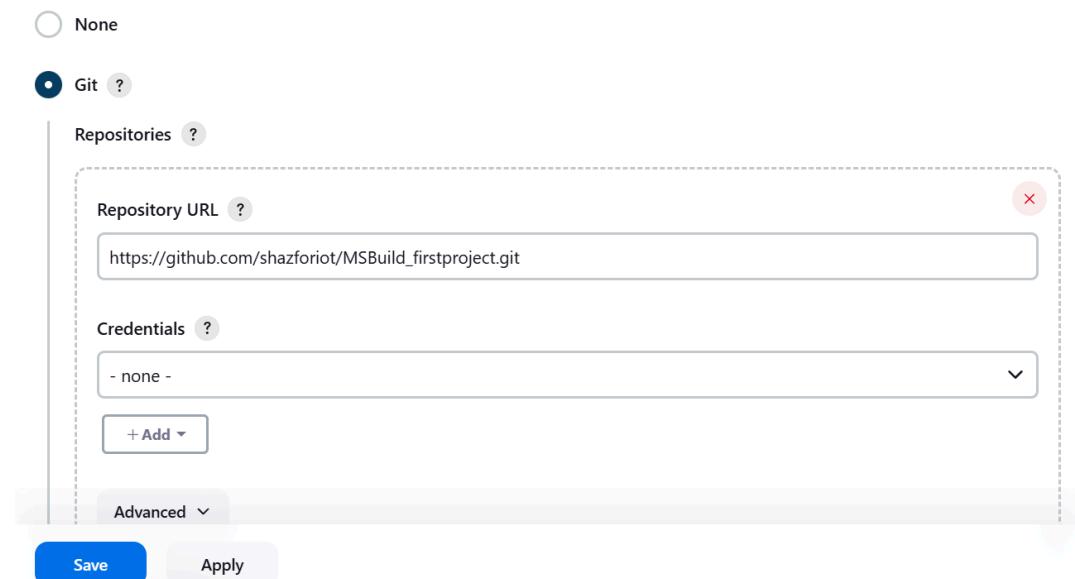
Repositories [?](#)

Repository URL [?](#)

Credentials [?](#)
- none -
[+ Add](#)

Advanced [▼](#)

[Save](#) [Apply](#)



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

Build Steps

Execute SonarQube Scanner

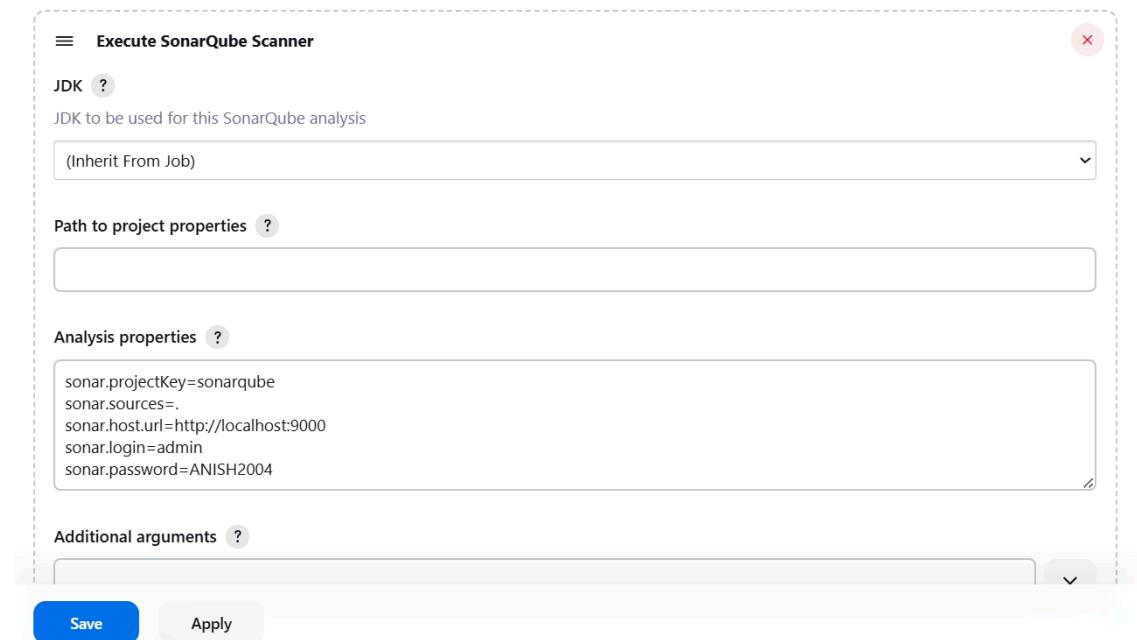
JDK [?](#)
JDK to be used for this SonarQube analysis

Path to project properties [?](#)

Analysis properties [?](#)

Additional arguments [?](#)

[Save](#) [Apply](#)



Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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

The screenshot shows the 'Global Permissions' section of the SonarQube administration interface. At the top, there are tabs for Configuration, Security, Projects, System, and Marketplace. Below that is a 'Global Permissions' header with a sub-header: 'Grant and revoke permissions to make changes at the global level. These permissions include editing Quality Profiles, executing analysis, and performing global system administration.' There are three tabs: All, Users, and Groups, with Groups selected. A search bar says 'Search for users or groups...'. The main table lists four entries: 'sonar-administrators' (System administrators), 'sonar-users' (Every authenticated user automatically belongs to this group), 'Anyone DEPRECATED' (Anybody who browses the application belongs to this group. If authentication is not enforced, assigned permissions also apply to non-authenticated users.), and 'Administrator admin'. For each entry, there are four checkboxes under 'Permissions': 'Administer System', 'Administer Quality Gates', 'Execute Analysis', and 'Create Projects'. The 'Execute Analysis' checkbox is checked for the 'Administrator' group. At the bottom, it says '4 of 4 shown'.

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

The screenshot shows the Jenkins project page for 'SonarqubeProject29'. The left sidebar has links: Status (highlighted), Changes, Workspace, Build Now (highlighted), Configure, Delete Project, SonarQube, Rename, and Build History. The main area shows the Jenkins logo and the project name 'SonarqubeProject29'. It includes a 'Permalinks' section with a list of build history items: Last build (#9), Last stable build (#9), Last successful build (#9), Last failed build (#8), Last unsuccessful build (#8), and Last completed build (#9). To the right are buttons for 'Add description' and 'Disable Project'.

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

```
Started by user Anish Kulkarni
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\.jenkins\workspace\SonarqubeProject29
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\SonarqubeProject29\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url http://github.com/shazforiot/MSBuild_firstproject.git # timeout=10
Fetching upstream changes from http://github.com/shazforiot/MSBuild_firstproject.git
> git.exe --version # timeout=10
> git --version # 'git version 2.46.0.windows.1'
> git.exe fetch --tags --force --progress -- http://github.com/shazforiot/MSBuild_firstproject.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
Commit message: "updated"
> git.exe rev-list --no-walk f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
[SonarqubeProject29] $ C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\scanner29\bin\sonar-
scanner.bat -Dsonar.host.url=http://localhost:9000 -Dsonar.projectKey=sonarqube -Dsonar.host.url=http://localhost:9000 -
Dsonar.login=admin -Dsonar.sources=. -Dsonar.password=ANISH2004 -
Dsonar.projectBaseDir=C:\ProgramData\Jenkins\.jenkins\workspace\SonarqubeProject29
16:13:33.220 WARN Property 'sonar.host.url' with value 'http://localhost:9000' is overridden with value 'http://localhost:9000'
16:13:33.236 INFO Scanner configuration file:
C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\scanner29\bin\..\conf\sonar-scanner.properties
16:13:33.246 INFO Project root configuration file: NONE

16:14:12.549 INFO Sensor C# [csharp] (done) | time=2ms
16:14:12.549 INFO Sensor Analysis Warnings import [csharp]
16:14:12.549 INFO Sensor Analysis Warnings import [csharp] (done) | time=0ms
16:14:12.549 INFO Sensor C# File Caching Sensor [csharp]
16:14:12.549 WARN Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting
'sonar.projectBaseDir' property.
16:14:12.549 INFO Sensor C# File Caching Sensor [csharp] (done) | time=0ms
16:14:12.549 INFO Sensor Zero Coverage Sensor
16:14:12.555 INFO Sensor Zero Coverage Sensor (done) | time=6ms
16:14:12.555 INFO SCM Publisher SCM provider for this project is: git
16:14:12.567 INFO SCM Publisher 4 source files to be analyzed
16:14:13.180 INFO SCM Publisher 4/4 source files have been analyzed (done) | time=613ms
16:14:13.184 INFO CPD Executor Calculating CPD for 0 files
16:14:13.184 INFO CPD Executor CPD calculation finished (done) | time=0ms
16:14:13.191 INFO SCM revision ID 'f2bc042c04c6e72427c380bcaee6d6fee7b49adf'
16:14:13.474 INFO Analysis report generated in 117ms, dir size=201.0 kB
16:14:13.522 INFO Analysis report compressed in 39ms, zip size=22.2 kB
16:14:13.756 INFO Analysis report uploaded in 231ms
16:14:13.759 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube
16:14:13.759 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis
report
16:14:13.759 INFO More about the report processing at http://localhost:9000/api/ce/task?id=fe09e0c2-e06a-411d-ae21-10cce1b3c081
16:14:13.787 INFO Analysis total time: 25.432 s
16:14:13.789 INFO SonarScanner Engine completed successfully
16:14:13.842 INFO EXECUTION SUCCESS
16:14:13.842 INFO Total time: 40.606s
Finished: SUCCESS
```

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

The screenshot shows the SonarQube interface for the 'main' project. At the top, there's a navigation bar with links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a search bar. Below the navigation is a breadcrumb trail showing the project path: sonarqube / main. The main content area has tabs for Overview, Issues, Security Hotspots, Measures, Code, and Activity. On the right, there are Project Settings and Project Information options. The central part of the page displays a large green box indicating a 'Passed' status for the Quality Gate. It also shows a warning message: 'The last analysis has warnings. See details'. Below this, there are two tabs: 'New Code' (selected) and 'Overall Code'. Under these tabs, there are three main sections: Security, Reliability, and Maintainability, each with a summary of open issues and severity counts (0 H, 0 M, 0 L). A 'Set as homepage' button is located in the top right corner of the main content area.

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

Experiment 8

Aim: Create a Jenkins 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.

Steps:

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

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

The screenshot shows a web browser displaying the SonarScanner CLI documentation. The URL in the address bar is <https://docs.sonarsource.com/sonarqube/latest/analyzing-source-code/scanners/sonarscanner/>. The page title is "SonarScanner CLI". On the left, there's a sidebar with navigation links for SonarQube, Server installation and setup, Analyzing source code, Scanners, and SonarScanner CLI. The main content area shows a table for version 6.1, which was released on 2024-06-27. It lists macOS and Linux AArch64 distributions. Below the table, there are sections for "Release notes" and "Scanner environment". To the right, there's a sidebar titled "On this page" with links to various documentation topics like "Configuring your project" and "Running SonarScanner CLI from the zip file". A "START FREE" button is visible at the top right.

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

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

```
PS C:\Users\anish\OneDrive\Desktop\Adv DevOps 7> docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest

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

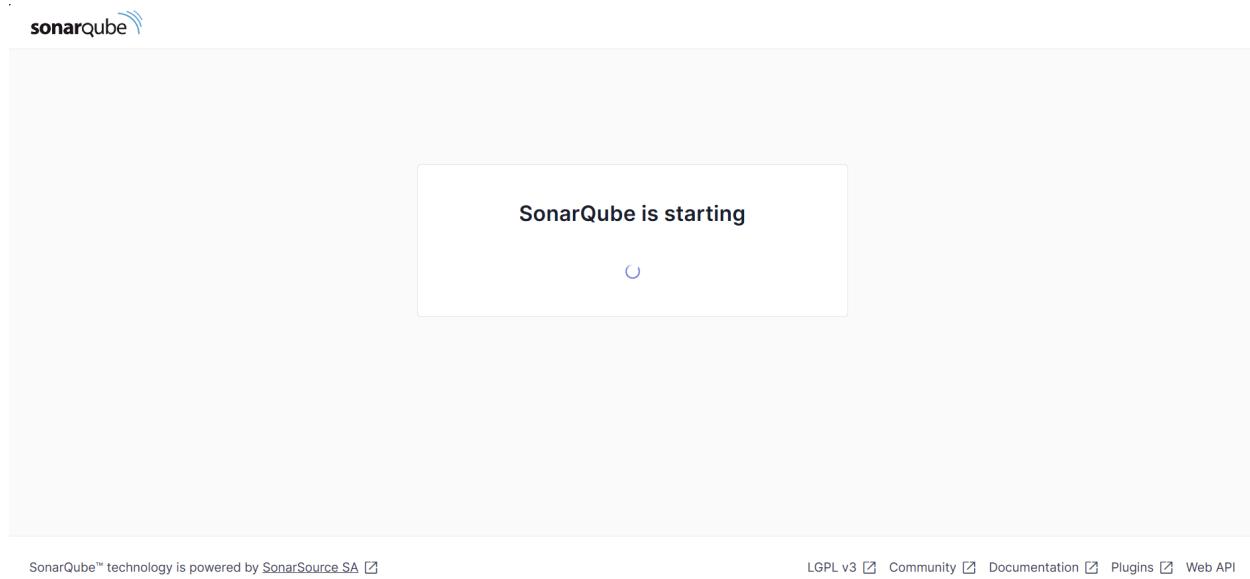
Step 3: Execute the following command:

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

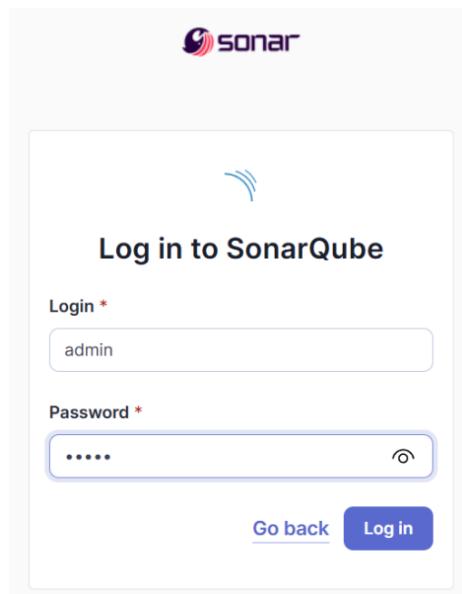
This command will run the SonarQube image that was just installed using docker.

```
PS C:\Users\anish\OneDrive\Desktop\Adv DevOps 7> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
dce67335909e42d81ec64d3ef0c5e5e2c36cc7ed36d87088033121ae1544f4fb
```

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



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



Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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

The screenshot shows the SonarQube dashboard with the 'Create a local project' option selected. It displays several import methods: Import from Azure DevOps, Import from Bitbucket Cloud, Import from Bitbucket Server, Import from GitHub, and Import from GitLab. Below these, there's a section for testing or advanced use-cases with a 'Create a local project' button.

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

1 of 2

Create a local project

Project display name *

sonarqube1

Project key *

sonarqube1

Main branch name *

main

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

Cancel

Next

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

2 of 2

Set up project for Clean as You Code

The new code definition 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

Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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

The screenshot shows the Jenkins 'Plugins' page. In the top navigation bar, there are icons for notifications (1), security (2), and user 'Anish Kulkarni'. The main search bar contains the text 'sonarqu'. Below the search bar, there is a list of available plugins. One plugin is highlighted: 'SonarQube Scanner 2.17.2'. The description below the plugin name states: 'This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.' To the right of the plugin listing, there is a timestamp '6 mo 29 days ago'. On the left side of the page, there are tabs for 'Updates' (22), 'Available plugins' (selected), 'Installed plugins', and 'Advanced settings'.

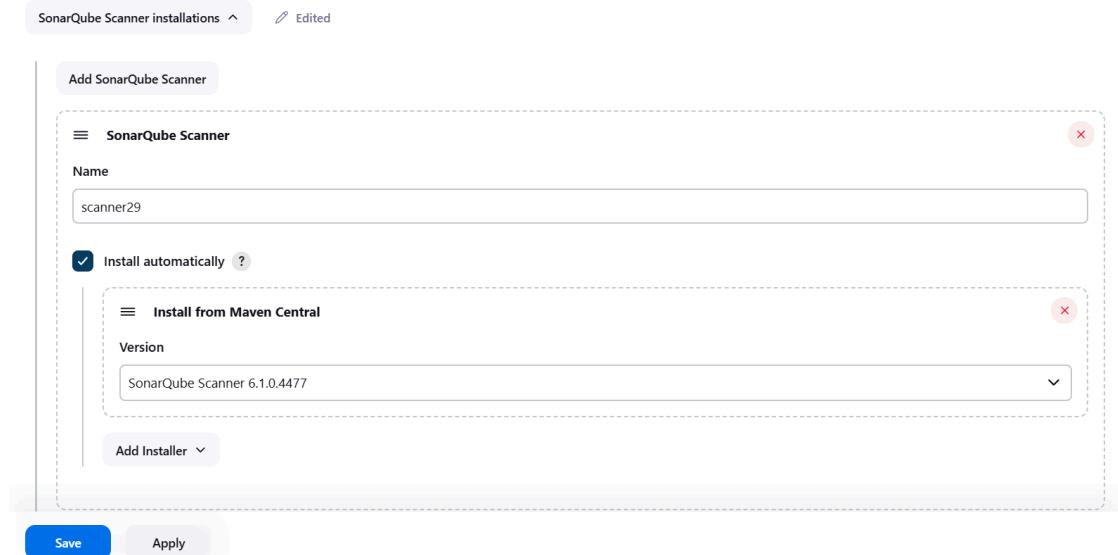
Step 10: Under ‘Manage Jenkins’, click on System. Under the ‘Sonarqube installations’ section, add a server and add a server authentication token if needed.

The screenshot shows the Jenkins 'System' configuration page under 'SonarQube installations'. The page title is 'List of SonarQube installations'. There is a single entry for 'sonarqube29'. The configuration fields for this entry include:

- Name:** sonarqube29
- Server URL:** http://localhost:9000 (Default is http://localhost:9000)
- Server authentication token:** A dropdown menu shows '- none -'. Below it is a button '+ Add ▾'.

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

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



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

Jenkins

Search (CTRL+K)

Anish Kulkarni

Dashboard > All >

Enter an item name

SonarqubeProject2

» Required field

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Use for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific etc.

OK

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

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

    stage('SonarQube analysis') {
        withSonarQubeEnv('sonarqube29') {
            bat """
                <PATH_TO SONARSCANNER FOLDER>\bin\sonar-scanner.bat ^
                -D sonar.login=<SONARQUBE_LOGIN> ^
                -D sonar.password=<SONARQUBE_PASSWORD> ^
                -D sonar.projectKey=<PROJECT_KEY> ^
                -D sonar.exclusions=vendor/**,resources/**, */*.java ^
                -D sonar.host.url=http://localhost:9000/
            """
        }
    }
}
```

Script ?

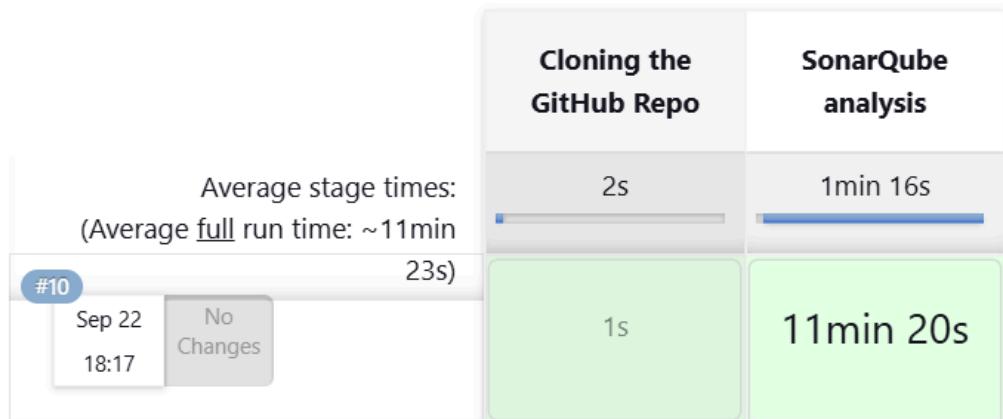
```
1 node {
2     stage('Cloning the GitHub Repo') {
3         git 'https://github.com/shazforiot/GOL.git'
4     }
5     stage('SonarQube analysis') {
6         withSonarQubeEnv('sonarqube29') {
7             bat """
8                 C:\\Users\\anish\\OneDrive\\Desktop\\sonar-scanner-6.1.0.4477-windows-x64\\bin\\sonar-scanner.bat ^
9                 -D sonar.login=admin ^
10                -D sonar.password=ANISH2004 ^
11                -D sonar.projectKey=sonarqube1 ^
12                -D sonar.exclusions=vendor/**,resources/**, */*.java \
13                -D sonar.host.url=http://localhost:9000/
14            """
15        }
16    }
17 }
```

try sample Pipeline... ▾

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

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

Stage View



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

```

Dashboard > SonarqubeProject2 > #10
18:26:17.936 WARN Too many duplication references on file gameoflife-
web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/sampler/HTTPSamplerBase.html for block at line 4757. Keep only the first
100 references.
18:26:17.936 WARN Too many duplication references on file gameoflife-
web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/sampler/HTTPSamplerBase.html for block at line 75. Keep only the first 100
references.
18:26:17.938 INFO CPD Executor CPD calculation finished (done) | time=121215ms
18:26:18.048 INFO SCM revision ID 'ba799ba7e1b576f04a461232b0412c5e6e1e5e4'
18:27:45.140 INFO Analysis report generated in 4310ms, dir size=127.2 MB
18:27:56.101 INFO Analysis report compressed in 10948ms, zip size=29.6 MB
18:28:00.915 INFO Analysis report uploaded in 4800ms
18:28:00.925 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube1
18:28:00.925 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis
report
18:28:00.925 INFO More about the report processing at http://localhost:9000/api/ce/task?id=2ed70e70-9e8c-438c-b080-f9fa3c343654
18:28:26.594 INFO Analysis total time: 11:12.471 s
18:28:26.626 INFO SonarScanner Engine completed successfully
18:28:27.349 INFO EXECUTION SUCCESS
18:28:27.534 INFO Total time: 11:18.248s
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```

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

The screenshot shows the SonarQube 'main' project overview. The Quality Gate is green with a checkmark and the word 'Passed'. Below it, there's a yellow warning box stating 'The last analysis has warnings. See details'. The dashboard displays various metrics: Security (0 Open issues), Reliability (68k Open issues), Maintainability (164k open issues), Accepted issues (0), Coverage (On 0 lines to cover), and Duplications (50.6% on 759k lines).

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

- Code problems:-

- Intentionality:-

The screenshot shows the SonarQube issue list for the 'gameoflife-acceptance-tests/Dockerfile' file. There are two main issues listed under the 'Intentionality' category:

- Use a specific version tag for the image.** (Maintainability, Open)
- Surround this variable with double quotes; otherwise, it can lead to unexpected behavior.** (Maintainability, Open)

Both issues are marked as 'Open' and have a 'Not assigned' status. The first issue was reported 4 years ago with L1 effort and is a Major code smell. The second issue was reported 4 years ago with L12 effort and is also a Major code smell.

- Consistency:-

The screenshot displays three separate code analysis results, each showing a checkbox for a task, the task name, a 'Consistency' button, a 'Maintainability' button, status dropdowns ('Open', 'Not assigned'), and metadata ('L11', '5min effort', '4 years ago', 'Code Smell', 'Major').

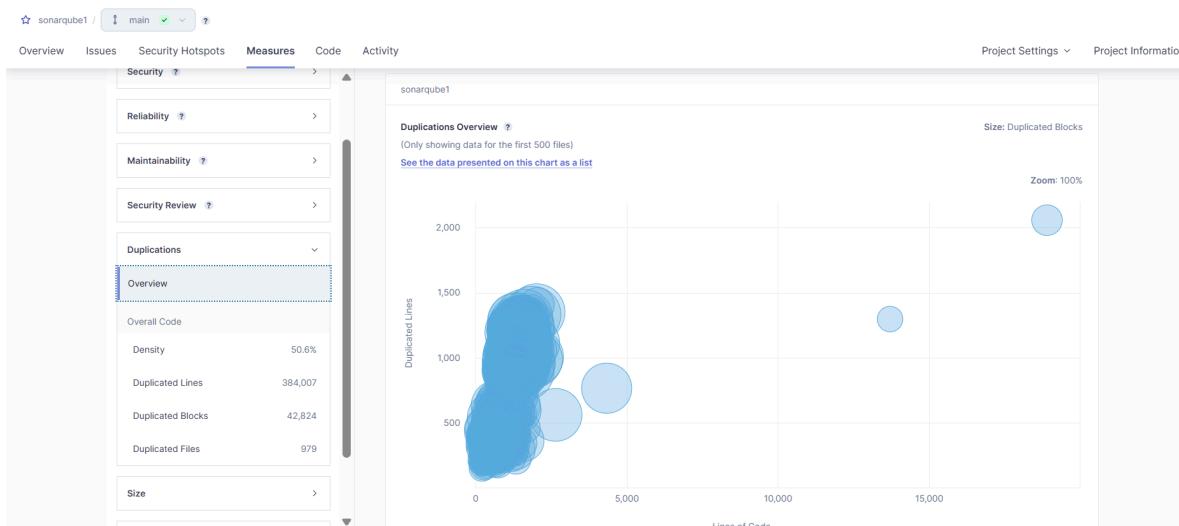
- Remove this deprecated "align" attribute. Consistency
- Remove this deprecated "align" attribute. Consistency
- Remove this deprecated "size" attribute. Consistency

- Bugs and Code Smells:-

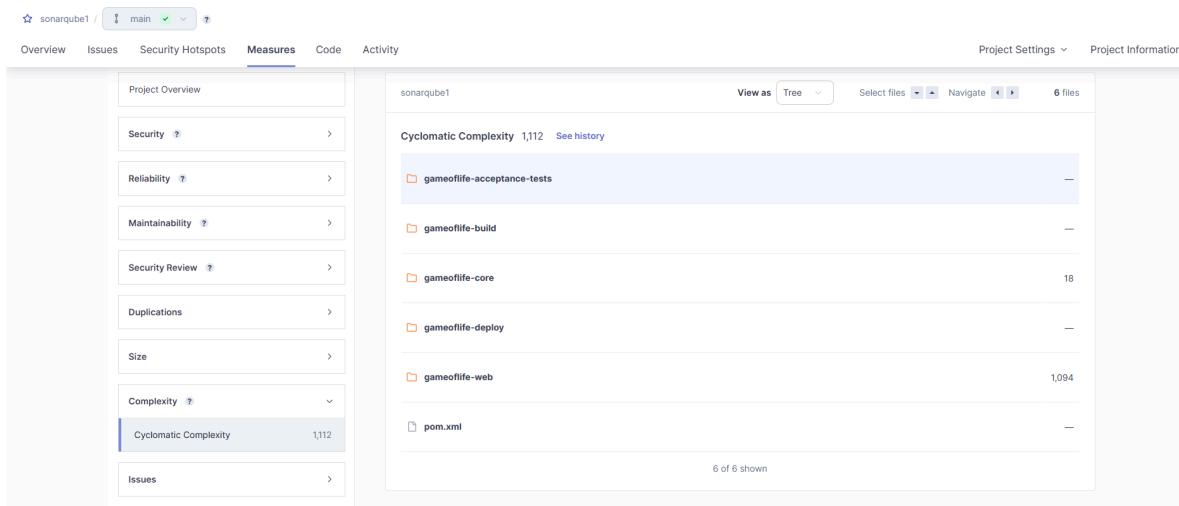
The screenshot displays several code analysis results, each showing a checkbox for a task, the task name, a 'Consistency' or 'Intentionality' button, a 'Reliability' button, status dropdowns ('Open', 'Not assigned'), and metadata ('L1', '2min effort', '4 years ago', 'Bug', 'Major').

- Add "lang" and/or "xml:lang" attributes to this "<html>" element Intentionality
- Insert a <!DOCTYPE> declaration to before this <html> tag. Consistency
- Remove this deprecated "valign" attribute. Consistency
- Remove this deprecated "name" attribute. Consistency
- Anchors must have content and the content must be accessible by a screen reader. Consistency

- Duplications:-



- Cyclomatic complexities:-



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

Experiment 9

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

Steps:

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

Security Groups (4) Info					Actions ▾	Export security groups to CSV	Create security group
<input type="text"/> Find resources by attribute or tag					< 1 >	⚙️	
	Name	Security group ID	Security group name	VPC ID			
<input type="checkbox"/>	-	sg-006804975d0e081cd	default	vpc-08fa61d400			
<input type="checkbox"/>	-	sg-04af0a4dee304e27f	Master	vpc-08fa61d400			
<input type="checkbox"/>	-	sg-096aefba3ebabf6bd	launch-wizard-15	vpc-08fa61d400			

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

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

Inbound rules Info							
Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info		
sgr-0455aad39f9e4f0dd	Custom TCP	TCP	5666	Custom	<input type="text"/> 0.0.0.0/0 X	Delete	
sgr-09942a64a610c4a51	All traffic	All	All	Custom	<input type="text"/> 0.0.0.0/0 X	Delete	
sgr-04fd581f3a2c685b7	All ICMP - IPv4	ICMP	All	Custom	<input type="text"/> 0.0.0.0/0 X	Delete	
sgr-0c38c654d3393939a	SSH	TCP	22	Custom	<input type="text"/> 0.0.0.0/0 X	Delete	
sgr-05c2dc0327e5303c7	All ICMP - IPv6	IPv6 ICMP	All	Custom	<input type="text"/> ::/0 X	Delete	
sgr-01f2fb5a613ff506	HTTP	TCP	80	Custom	<input type="text"/> ::/0 X	Delete	
sgr-0ac3b5859ba38d7b1	HTTPS	TCP	443	Custom	<input type="text"/> 0.0.0.0/0 X	Delete	

[Add rule](#)

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

Inbound rules (7)						
Security group rule...	IP version	Type	Protocol	Port range	Source	
sgr-0455aad39f9e4f0dd	IPv4	Custom TCP	TCP	5666	0.0.0.0/0	
sgr-09942a64a610c4a...	IPv4	All traffic	All	All	0.0.0.0/0	
sgr-04fd581f3a2c685b7	IPv4	All ICMP - IPv4	ICMP	All	0.0.0.0/0	
sgr-0c38c654d339393...	IPv4	SSH	TCP	22	0.0.0.0/0	
sgr-05c2dc0327e5303c7	IPv6	All ICMP - IPv6	IPv6 ICMP	All	::/0	
sgr-01f2fb5a613ff506	IPv6	HTTP	TCP	80	::/0	
sgr-0ac3b5859ba38d7...	IPv4	HTTPS	TCP	443	0.0.0.0/0	

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

The screenshot shows the AWS EC2 Instances page. The left sidebar has 'Instances' expanded, with 'Instances' selected. The main area displays a message: 'No instances' and 'You do not have any instances in this region'. Below this is a large orange 'Launch instances' button. The top navigation bar includes the AWS logo, services menu, search bar, and user information.

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

The screenshot shows the 'Create New Instance' wizard. Step 1: 'Name and tags'. It shows a 'Name' field with 'nagios-host29' and a 'Summary' section with 'Number of instances: 1'. Step 2: 'Application and OS Images (Amazon Machine Image)'. It shows a search bar and a list of AMIs including Amazon Linux, macOS, Ubuntu, Windows, Red Hat, and SUSE Linux. Step 3: 'Security Groups'. It shows a summary of the chosen security group and a note about free tier. Step 4: 'Review and Launch'. It shows the 'Launch instance' button. The top navigation bar is visible at the top.

Key pair (login) [Info](#)

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

Key pair name - *required*

AdvDevopsLab [Edit](#) [Create new key pair](#)

Network settings [Info](#) [Edit](#)

Network [Info](#)
vpc-08fa61d4002ee05df

Subnet [Info](#)
No preference (Default subnet in any availability zone)

Auto-assign public IP [Info](#)
Enable
Additional charges apply when outside of [free tier allowance](#)

Firewall (security groups) [Info](#)
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group Select existing security group

Common security groups [Info](#)
Select security groups

launch-wizard-15 sg-096ae9ba3ebabf6bd X
VPC: vpc-08fa61d4002ee05df

Compare security group rules

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

[Cancel](#) [Launch instance](#) [Review commands](#)

Instances (1) [Info](#) Last updated less than a minute ago [Connect](#) [Actions](#) [Launch instances](#)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
nagios-host29	i-03d0c63994db6400b	Running	t2.micro	-	View alarms

Your instance gets created.

Step 3: Click on the instance ID of your instance and click on ‘Connect’. Then, click on ‘SSH client’ and copy the command under ‘Example’.

The screenshot shows the AWS EC2 Connect interface for an instance with ID i-03d0c63994db6400b. The 'SSH client' tab is active. The page provides instructions for connecting via SSH, including steps like opening an SSH client, locating a private key file, running a command to ensure the key is not publicly viewable, and connecting using the Public DNS. It also shows an example command: ssh -i "AdvDevopsLab.pem" ec2-user@ec2-3-84-19-157.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 4: Now, we need to connect our local terminal to the instance using SSH. To do so, open the terminal in the folder where the .pem file for your instance's key pair is located and paste the SSH command that you copied in Step 3.

```
PS C:\Users\anish\Downloads> ssh -i "AdvDevopsLab.pem" ec2-user@ec2-3-84-19-157.compute-1.amazonaws.com
The authenticity of host 'ec2-3-84-19-157.compute-1.amazonaws.com (3.84.19.157)' can't be established.
ED25519 key fingerprint is SHA256:a+QXqvw5WaiVmzblX2AJvzqISc12vkiVvsUMxtmoug.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-84-19-157.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

#_
~\_ ##### Amazon Linux 2023
~~ \_\#\#\#\#
~~ \#\#\#
~~ \#/ ___ https://aws.amazon.com/linux/amazon-linux-2023
~~ \~' '->
~~ _/
~~ _/ _/
~~ _/ _/
[ec2-user@ip-172-31-88-33 ~]$ |
```

This connects your instance to your local terminal using SSH.

Step 5: First, run the following command:-

`sudo yum update`

This command will check for any updates for the YUM library.

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

Step 6: Run the command:

```
sudo yum install httpd php
```

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

```
[ec2-user@ip-172-31-88-33 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:04:36 ago on Sun Sep 29 10:22:03 2024.
Dependencies resolved.
=====
==== Packages =====
=====
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-httdp 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
libssl             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
=====
==== 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
libssl-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-xml-8.3.10-1.amzn2023.0.1.x86_64
=====
apr-util-1.6.3-1.amzn2023.0.1.x86_64
generic-logos-httdp-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!
```

Step 7: Run the command:

```
sudo yum install gcc glibc glibc-common
```

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

```
[ec2-user@ip-172-31-88-33 ~]$ sudo yum install gcc glibc glibc-common
Last metadata expiration check: 0:05:41 ago on Sun Sep 29 10:22:03 2024.
Package glibc-2.34-52.amzn2023.0.11.x86_64 is already installed.
Package glibc-common-2.34-52.amzn2023.0.11.x86_64 is already installed.
Dependencies resolved.
=====
          Package           Architecture     Version      Repository  Size
=====
Installing:
  gcc                  x86_64        11.4.1-2.amzn2023.0.2    amazonlinux  32 M
Installing dependencies:
  annobin-docs          noarch       10.93-1.amzn2023.0.1    amazonlinux  92 k
  annobin-plugin-gcc    x86_64        10.93-1.amzn2023.0.1    amazonlinux  887 k
  cpp                  x86_64        11.4.1-2.amzn2023.0.2    amazonlinux  10 M
  gc                   x86_64        8.0.4-5.amzn2023.0.2    amazonlinux  105 k
  glibc-devel           x86_64        2.34-52.amzn2023.0.11   amazonlinux  27 k
  glibc-headers-x86    noarch       2.34-52.amzn2023.0.11   amazonlinux  427 k
  guile22              x86_64        2.2.7-2.amzn2023.0.3    amazonlinux  6.4 M
  kernel-headers        x86_64        6.1.109-118.189.amzn2023 amazonlinux  1.4 M
  libmpc               x86_64        1.2.1-2.amzn2023.0.2    amazonlinux  62 k
  libtool-ltdl          x86_64        2.4.7-1.amzn2023.0.3    amazonlinux  38 k
  libxcrypt-devel       x86_64        4.4.33-7.amzn2023      amazonlinux  32 k
  make                 x86_64        1:4.3-5.amzn2023.0.2    amazonlinux  534 k

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

```
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-172-31-88-33 ~]$ |
```

Step 8: Run the command:

```
sudo yum install gd gd-devel
```

```
[ec2-user@ip-172-31-88-33 ~]$ sudo yum install gd gd-devel
Last metadata expiration check: 0:06:51 ago on Sun Sep 29 10:22:03 2024.
Dependencies resolved.
=====
Package           Architecture Version      Repository  Size
=====
Installing:
  gd                  x86_64    2.3.3-5.amzn2023.0.3   amazonlinux 139 k
  gd-devel            x86_64    2.3.3-5.amzn2023.0.3   amazonlinux 38 k
Installing dependencies:
  brotli              x86_64    1.0.9-4.amzn2023.0.2   amazonlinux 314 k
  brotli-devel        x86_64    1.0.9-4.amzn2023.0.2   amazonlinux 31 k
  bzip2-devel         x86_64    1.0.8-6.amzn2023.0.2   amazonlinux 214 k
  cairo               x86_64    1.17.6-2.amzn2023.0.1   amazonlinux 684 k
  cmake-filesystem    x86_64    3.22.2-1.amzn2023.0.4   amazonlinux 16 k
  fontconfig          x86_64    2.13.94-2.amzn2023.0.2   amazonlinux 273 k
  fontconfig-devel    x86_64    2.13.94-2.amzn2023.0.2   amazonlinux 128 k
  fonts-filesystem   noarch   1:2.0.5-12.amzn2023.0.2   amazonlinux 9.5 k
  freetype             x86_64    2.13.2-5.amzn2023.0.1   amazonlinux 423 k
  freetype-devel      x86_64    2.13.2-5.amzn2023.0.1   amazonlinux 912 k
  libffi-devel-3.4.4-1.amzn2023.0.1.x86_64
  libicu-devel-67.1-7.amzn2023.0.3.x86_64
  libjpeg-turbo-devel-2.1.4-2.amzn2023.0.5.x86_64
  libpng-2:1.6.37-10.amzn2023.0.6.x86_64
  libselinux-devel-3.4-5.amzn2023.0.2.x86_64
  libtiff-4.4.0-4.amzn2023.0.18.x86_64
  libwebp-1.2.4-1.amzn2023.0.6.x86_64
  libxcb-1.13.1-7.amzn2023.0.2.x86_64
  libxml2-devel-2.10.4-1.amzn2023.0.6.x86_64
  pcre2-utf16-10.40-1.amzn2023.0.3.x86_64
  pixman-0.40.0-3.amzn2023.0.3.x86_64
  xml-common-0.6.3-56.amzn2023.0.2.noarch
  xz-devel-5.2.5-9.amzn2023.0.2.x86_64
  libicu-67.1-7.amzn2023.0.3.x86_64
  libjpeg-turbo-2.1.4-2.amzn2023.0.5.x86_64
  libmount-devel-2.37.4-1.amzn2023.0.4.x86_64
  libpng-devel-2:1.6.37-10.amzn2023.0.6.x86_64
  libsepol-devel-3.4-3.amzn2023.0.3.x86_64
  libtiff-devel-4.4.0-4.amzn2023.0.18.x86_64
  libwebp-devel-1.2.4-1.amzn2023.0.6.x86_64
  libxcb-devel-1.13.1-7.amzn2023.0.2.x86_64
  pcre2-devel-10.40-1.amzn2023.0.3.x86_64
  pcre2-utf32-10.40-1.amzn2023.0.3.x86_64
  sysprof-capture-devel-3.40.1-2.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-172-31-88-33 ~]$ |
```

Step 9: Run the commands:

```
sudo adduser -m nagios
```

```
sudo passwd nagios
```

This creates a user named 'nagios', ensures it has a home directory and sets up a password for it.

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

Step 10: Create a user group named ‘nagcmd’ to execute nagios commands.
 sudo groupadd nagcmd

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

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

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

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

Step 12: mkdir ~/downloads

```
cd ~/downloads
```

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

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

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

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

```
[ec2-user@ip-172-31-88-33 downloads]$ wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
--2024-09-29 10:33:56-- https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2600:3c00::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  8.29MB/s    in 0.2s

2024-09-29 10:33:57 (8.29 MB/s) - 'nagios-4.5.5.tar.gz' saved [2065473/2065473]
```

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

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

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

nagios-plugins-2.4.11.tar.gz 100%[=====]  2.62M  9.99MB/s    in 0.3s

2024-09-29 10:34:30 (9.99 MB/s) - 'nagios-plugins-2.4.11.tar.gz' saved [2753049/2753049]
```

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

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

```
[ec2-user@ip-172-31-88-33 downloads]$ tar zxvf nagios-4.5.5.tar.gz
nagios-4.5.5/
nagios-4.5.5/.github/
nagios-4.5.5/.github/workflows/
nagios-4.5.5/.github/workflows/test.yml
nagios-4.5.5/.gitignore
nagios-4.5.5/CONTRIBUTING.md
nagios-4.5.5/Changelog
nagios-4.5.5/INSTALLING
nagios-4.5.5/LEGAL
nagios-4.5.5/LICENSE
nagios-4.5.5/Makefile.in
nagios-4.5.5/README.md
nagios-4.5.5/THANKS
nagios-4.5.5/UPGRADING
nagios-4.5.5/aclocal.m4
nagios-4.5.5/autoconf-macros/
nagios-4.5.5/autoconf-macros/.gitignore
nagios-4.5.5/autoconf-macros/CHANGELOG.md
nagios-4.5.5/autoconf-macros/LICENSE
nagios-4.5.5/autoconf-macros/LICENSE.md
nagios-4.5.5/autoconf-macros/README.md
nagios-4.5.5/autoconf-macros/_dd_group_user
```

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

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

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

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

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

Use 'ls' command to find the correct directory.

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

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

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

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

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

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

Total download size: 3.0 M
Installed size: 4.7 M
Is this ok [y/N]: y
Downloading Packages:
openssl-devel-3.0.8-1.amzn2023.0.14.x86_64.rpm          26 MB/s | 3.0 MB   00:00
=====
Total                                         16 MB/s | 3.0 MB   00:00

Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing          :
  Installing         : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64          1/1
  Running scriptlet: openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64          1/1
  Verifying          : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64          1/1
```

```
Installed:  
  openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64  
Complete!
```

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

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

```
General Options:  
-----  
  Nagios executable: nagios  
  Nagios user/group: nagios,nagios  
  Command user/group: nagios,nagcmd  
    Event Broker: yes  
  Install ${prefix}: /usr/local/nagios  
  Install ${includedir}: /usr/local/nagios/include/nagios  
    Lock file: /run/nagios.lock  
  Check result directory: /usr/local/nagios/var/spool/checkresults  
    Init directory: /lib/systemd/system  
  Apache conf.d directory: /etc/httpd/conf.d  
    Mail program: /bin/mail  
    Host OS: linux-gnu  
  IOBroker Method: epoll  
  
Web Interface Options:  
-----  
    HTML URL: http://localhost/nagios/  
    CGI URL: http://localhost/nagios/cgi-bin/  
Traceroute (used by WAP): /usr/bin/traceroute
```

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

Step 17: Next, we must compile all components of this software according to the instructions in the Makefile. To do so, use the following command:

`make all`

Then,

`sudo make install`

`sudo make install-init`

`sudo make install-config`

`sudo make install-commandmode`

```
[ec2-user@ip-172-31-88-33 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./lib -I./include -I./include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o nagios.o ./nagios.c
gcc -Wall -I.. -I./lib -I./include -I./include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o broker.o broker.c
gcc -Wall -I.. -I./lib -I./include -I./include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o nebmods.o nebmods.c
gcc -Wall -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./lib -I./include -I./include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o query-handler.o quer
y-handler.c
gcc -Wall -I.. -I./lib -I./include -I./include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o workers.o workers.c
In function 'get_wproc_list',
  inlined from 'get_worker' at workers.c:277:12:
workers.c:253:17: warning: '%s' directive argument is null [-Wformat-overflow=]
  253 |         log_debug_info(DEBUGL_CHECKS, 1, "Found specialized worker(s) for '%s'", (slash && *slash != '/'
 ) ? slash : cmd_name);
|           ^
=====
gcc -Wall -I.. -I./lib -I./include -I./include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o checks.o checks.c
gcc -Wall -I.. -I./lib -I./include -I./include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o config.o config.c
gcc -Wall -I.. -I./lib -I./include -I./include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o commands.o commands.
c
gcc -Wall -I.. -I./lib -I./include -I./include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o events.o events.c
gcc -Wall -I.. -I./lib -I./include -I./include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o flapping.o flapping.
c
gcc -Wall -I.. -I./lib -I./include -I./include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o logging.o logging.c
gcc -Wall -I.. -I./lib -I./include -I./include -I.. -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o macros.o macros.c
=====
```

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-172-31-88-33 nagios-4.5.5]$ |
```

```
[ec2-user@ip-172-31-88-33 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 -b -m 664 -o nagios -g nagios sample-config/template-object/switc.cfg /usr/local/nagios/etc/objects \
    /switch.cfg
done
*** 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 ***
```

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

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

```
GNU nano 5.8                               /usr/local/nagios/etc/objects/contacts.cfg                         Modified

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

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

define contact {

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

}

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

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

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

Step 19: sudo make install-webconf

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

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

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

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

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

```
[ec2-user@ip-172-31-88-33 nagios-4.5.5]$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
```

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

sudo service httpd restart

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

Step 22: cd ~/downloads

tar zxvf nagios-plugins-2.4.11.tar.gz

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

```
[ec2-user@ip-172-31-88-33 downloads]$ tar 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/lmain.sh
nagios-plugins-2.4.11/build-aux/missing
nagios-plugins-2.4.11/build-aux/mkinstalldirs
nagios-plugins-2.4.11/build-aux/depcomp
nagios-plugins-2.4.11/build-aux/snippet/
nagios-plugins-2.4.11/build-aux/snippet/_Noreturn.h
nagios-plugins-2.4.11/build-aux/snippet/arg-nonnull.h
nagios-plugins-2.4.11/build-aux/snippet/c++defs.h
nagios-plugins-2.4.11/build-aux/snippet/warn-on-use.h
nagios-plugins-2.4.11/build-aux/test-driver
nagios-plugins-2.4.11/config_test/
nagios-plugins-2.4.11/config_test/Makefile
```

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

Step 23: cd nagios-plugins-2.4.11

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

This installs the configurations for the nagios-plugins files.

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

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

```
make
```

```
sudo make install
```

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

Step 25: sudo chkconfig --add nagios

```
sudo chkconfig nagios on
```

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

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

Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

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

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

```
[ec2-user@ip-172-31-88-33 nagios-plugins-2.4.11]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.5.5
Copyright (c) 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 service escalations.

Checking for circular paths...
  Checked 1 hosts
  Checked 0 service dependencies
  Checked 0 host dependencies
  Checked 5 timeperiods

Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors: 0

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

Step 27: sudo service nagios start

This starts the Nagios service.

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

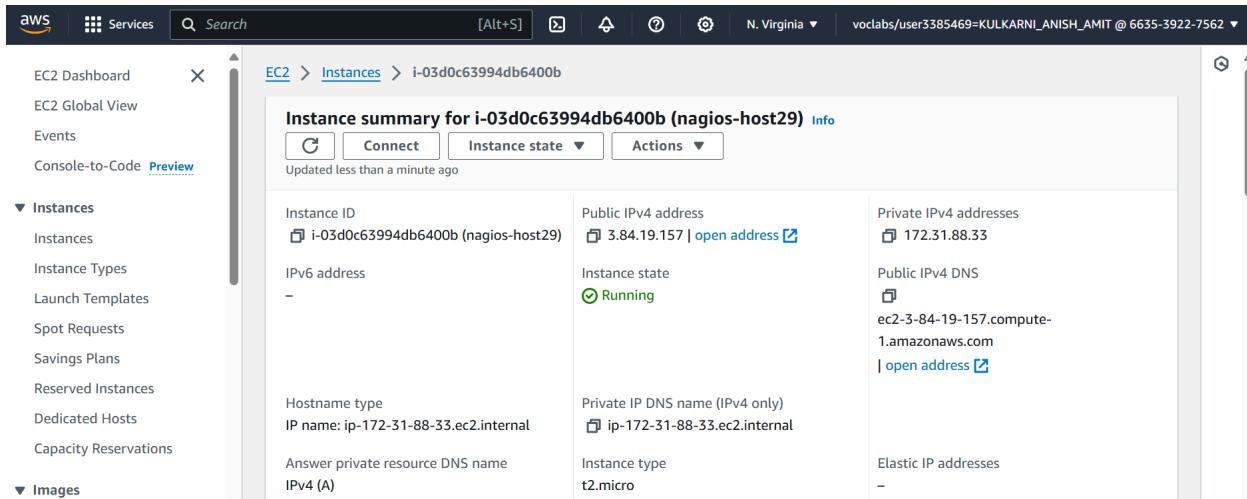
Step 28: sudo systemctl status nagios

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

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

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

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



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

The screenshot shows the Nagios Core dashboard. At the top right, it displays "Nagios® Core™ Version 4.5.5" and the date "September 17, 2024". A green checkmark indicates "Daemon running with PID 67489". On the left, a sidebar menu includes sections for General (Home, Documentation), Current Status (Tactical Overview, Map, Hosts, Services, Host Groups, Service Groups, Problems, Reports, System), and Reports (Availability, Trends, Alerts, History, Summary, Histogram, Notifications, Event Log). The main content area features four boxes: "Get Started" (bullet points: Start monitoring your infrastructure, Change the look and feel of Nagios, Extend Nagios with hundreds of addons, Get support, Get training, Get certified), "Latest News" (empty), "Don't Miss..." (empty), and "Quick Links" (bullet points: Nagios Library (tutorials and docs), Nagios Labs (development blog), Nagios Exchange (plugins and addons), Nagios Support (tech support), Nagios.com (company), Nagios.org (project)). A "Page Tour" button is located on the right side of the dashboard.

The above page is visible.

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

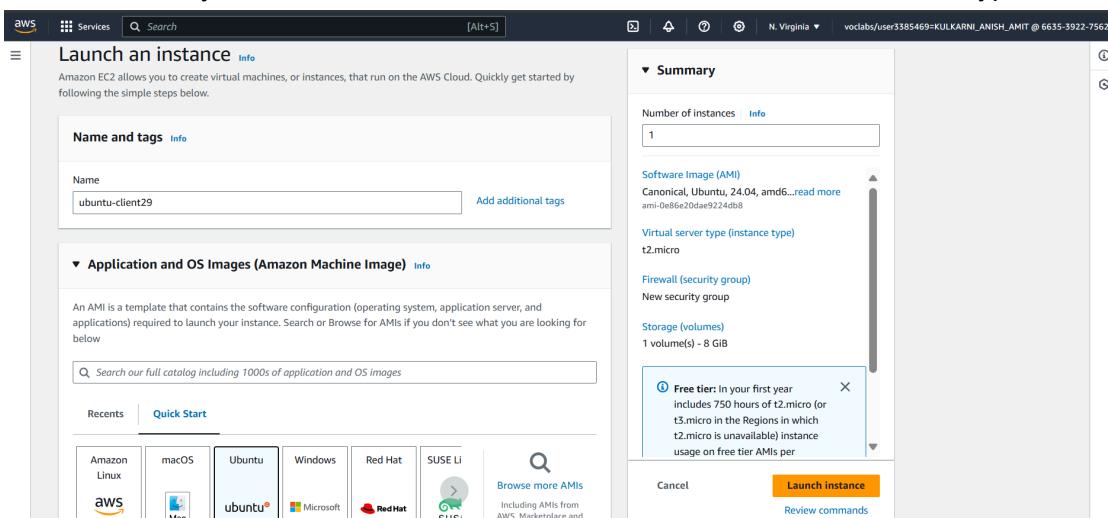
Experiment 10

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

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

Steps:

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



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

The screenshot shows the AWS EC2 'Key pair (login)' configuration page. It includes fields for 'Key pair name - required' (Ad/DevopsLab), 'Network' (vpc-08fa61d4002ee05df), 'Subnet' (No preference), 'Firewall (security group)' (Create security group or Select existing security group), and 'Storage (volumes)' (1 volume(s) - 8 GiB). A tooltip for the 'Free tier' is displayed, stating: 'Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOPS, 1 GB of snapshots, and 100 GiB of bandwidth to the internet.' A 'Launch instance' button is at the bottom.

The screenshot shows the AWS EC2 'Instances' page with two instances listed: 'ubuntu-client29' (Pending) and 'nagios-host29' (Running). The 'Launch instances' button is visible at the top right of the main content area.

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

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

Step 3: ps -ef | grep nagios

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

```
[root@ip-172-31-88-33 ec2-user]# ps -ef | grep nagios
nagios  67489      1  0 11:25 ?
          00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios  67490  67489  0 11:25 ?
          00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.sock
nagios  67491  67489  0 11:25 ?
          00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.sock
nagios  67492  67489  0 11:25 ?
          00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.sock
nagios  67493  67489  0 11:25 ?
          00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.sock
nagios  67494  67489  0 11:25 ?
          00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
root   69067  68853  0 11:51 pts/1
          00:00:00 grep --color=auto nagios
[root@ip-172-31-88-33 ec2-user]# |
```

Step 4: sudo su

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

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

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

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

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

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

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

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

Step 6: We need to make some changes in this config file. Open it using nano editor:-
nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

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

```
#####
#
# HOST DEFINITION
#
#####

# Define a host for the local machine

define host {

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

    host_name    linuxserver
    alias        linuxserver
    address      52.91.101.68

}

#####

#
# HOST GROUP DEFINITION
#
```

Change hostgroup_name to ‘linux-servers1’.

```
define hostgroup {

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

}
```

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

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

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

Then, add the following line to the config file:

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

```
GNU nano 5.8                               /usr/local/nagios/etc/nagios.cfg

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

# Definitions for monitoring a Windows machine
#cfg_file=/usr/local/nagios/etc/objects/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/servers
#cfg_dir=/usr/local/nagios/etc/printers
#cfg_dir=/usr/local/nagios/etc/switches
#cfg_dir=/usr/local/nagios/etc/routers
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/
```

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

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

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

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

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

Running pre-flight check on configuration data...

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

```

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

Total Warnings: 0
Total Errors: 0

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

```

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

```
service nagios restart
```

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

Step 10: systemctl status nagios

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

```
[root@ip-172-31-88-33 ec2-user]# systemctl status nagios
● nagios.service - Nagios Core 4.5.5
    Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
    Active: active (running) since Sun 2024-09-29 12:11:40 UTC; 1min 12s ago
      Docs: https://www.nagios.org/documentation
   Process: 70244 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0)
   Process: 70245 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SU>
 Main PID: 70246 (nagios)
   Tasks: 6 (limit: 1112)
  Memory: 4.0M
    CPU: 38ms
   CGroup: /system.slice/nagios.service
           ├─70246 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
           ├─70247 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─70248 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─70249 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─70250 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           └─70251 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

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

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

```
PS C:\Users\anish\Downloads> ssh -i "AdvDevopsLab.pem" ubuntu@ec2-52-91-101-68.compute-1.amazonaws.com
The authenticity of host 'ec2-52-91-101-68.compute-1.amazonaws.com (52.91.101.68)' can't be established.
ED25519 key fingerprint is SHA256:Z6cgJrMFcPL5SxJ9EzJKrB3lt1bYaG1x6Ntu/PKumPw.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-52-91-101-68.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

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

System information as of Sun Sep 29 12:15:10 UTC 2024

System load:  0.0          Processes:           105
Usage of /:   22.7% of 6.71GB  Users logged in:     0
Memory usage: 19%          IPv4 address for enX0: 172.31.94.199
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

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

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

`sudo apt update -y`

`sudo apt install gcc -y`

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

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

```
ubuntu@ip-172-31-94-199:~$ sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [380 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [535 kB]
Get:15 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [82.9 kB]
Get:16 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4560 B]
```

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

Running kernel seems to be up-to-date.

Restarting services...

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

No containers need to be restarted.

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

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

Step 13: Run the following command:

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

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

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

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

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

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

Host Status Details For All Host Groups

Host	Status	Last Check	Duration	Status Information
linuxserver	UP	09-29-2024 12:21:02	0d 0h 9m 55s	PING OK - Packet loss = 0%, RTA = 1.02 ms
localhost	UP	09-29-2024 12:20:02	0d 0h 55m 54s	PING OK - Packet loss = 0%, RTA = 0.04 ms

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

Host Information

Host
linuxserver
(linuxserver)

Member of
linux-servers1

52.91.101.68

Host State Information

Host Status:	UP (for 0d 0h 11m 41s)
Status Information:	PING OK - Packet loss = 0%, RTA = 1.02 ms
Performance Data:	rta=1.023000ms;3000.000000;5000.000000;0.000000 pl=0%;80;100;0
Current Attempt:	1/10 (HARD state)
Last Check Time:	09-29-2024 12:21:02
Check Type:	ACTIVE
Check Latency / Duration:	0.000 / 4.122 seconds
Next Scheduled Active Check:	09-29-2024 12:26:02
Last State Change:	09-29-2024 12:11:40
Last Notification:	N/A (notification 0)
Is This Host Flapping?	NO (0.00% state change)
In Scheduled Downtime?	NO
Last Update:	09-29-2024 12:23:19 (0d 0h 0m 2s ago)
Active Checks:	ENABLED
Passive Checks:	ENABLED
Obsessing:	ENABLED
Notifications:	ENABLED
Event Handler:	ENABLED
Flap Detection:	ENABLED

Host Commands

- Locate host on map
- Disable active checks of this host
- Re-schedule the next check of this host
- Submit passive check result for this host
- Stop accepting passive checks for this host
- Stop obsessing over this host
- Disable notifications for this host
- Send custom host notification
- Schedule downtime for this host
- Schedule downtime for all services on this host
- Disable notifications for all services on this host
- Enable notifications for all services on this host
- Schedule a check of all services on this host
- Disable checks of all services on this host
- Enable checks of all services on this host
- Disable event handler for this host
- Disable flap detection for this host
- Clear flapping state for this host

Host Comments

Add a new comment Delete all comments

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

The screenshot shows the Nagios web interface with the following sections:

- Current Network Status:** Last Updated: Sun Sep 29 12:24:32 UTC 2024. Updated every 90 seconds. Nagios® Core™ 4.5.5 - www.nagios.org. Logged in as nagiosadmin.
- Host Status Totals:** Up: 2, Down: 0, Unreachable: 0, Pending: 0. All Problems: 12, All Types: 16.
- Service Status Totals:** Ok: 12, Warning: 1, Unknown: 0, Critical: 3, Pending: 0. All Problems: 4, All Types: 16.
- Service Status Details For All Hosts:** A table listing services for hosts 'linuxserver' and 'localhost'. The table includes columns: Host, Service, Status, Last Check, Duration, Attempt, and Status Information.

Host	Service	Status	Last Check	Duration	Attempt	Status Information
linuxserver	Current Load	OK	09-29-2024 12:22:17	0d 0h 12m 15s	1/4	OK - load average: 0.00, 0.00, 0.00
	Current Users	OK	09-29-2024 12:22:55	0d 0h 11m 37s	1/4	USERS OK - 5 users currently logged in
	HTTP	CRITICAL	09-29-2024 12:21:32	0d 0h 8m 0s	4/4	connect to address 52.91.101.68 and port 80: Connection refused
	PING	OK	09-29-2024 12:24:10	0d 0h 10m 22s	1/4	PING OK - Packet loss = 0%, RTA = 0.74 ms
	Root Partition	OK	09-29-2024 12:19:47	0d 0h 9m 45s	1/4	DISK OK - free space: / 6107 MB (75.24% inode=98%)
	SSH	OK	09-29-2024 12:20:25	0d 0h 9m 7s	1/4	SSH OK - OpenSSH_9.6p1 Ubuntu-Subuntu13.4 (protocol 2.0)
localhost	Swap Usage	CRITICAL	09-29-2024 12:24:02	0d 0h 5m 30s	4/4	SWAP CRITICAL - 0% free (0 MB out of 0 MB) - Swap is either disabled, not present, or of zero size.
	Total Processes	OK	09-29-2024 12:21:40	0d 0h 7m 52s	1/4	PROCS OK - 43 processes with STATE = RSZDT
	Current Load	OK	09-29-2024 12:21:17	0d 0h 58m 15s	1/4	OK - load average: 0.00, 0.00, 0.00
	Current Users	OK	09-29-2024 12:21:56	0d 0h 57m 37s	1/4	USERS OK - 5 users currently logged in
	HTTP	WARNING	09-29-2024 12:20:30	0d 0h 54m 1s	4/4	HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes in 0.001 second response time
	PING	OK	09-29-2024 12:23:10	0d 0h 56m 22s	1/4	PING OK - Packet loss = 0%, RTA = 0.05 ms
	Root Partition	OK	09-29-2024 12:23:47	0d 0h 55m 45s	1/4	DISK OK - free space: / 6107 MB (75.24% inode=98%)
	SSH	OK	09-29-2024 12:24:25	0d 0h 55m 7s	1/4	SSH OK - OpenSSH_8.7 (protocol 2.0)
	Swap Usage	CRITICAL	09-29-2024 12:23:02	0d 0h 51m 30s	4/4	SWAP CRITICAL - 0% free (0 MB out of 0 MB) - Swap is either disabled, not present, or of zero size.
	Total Processes	OK	09-29-2024 12:20:40	0d 0h 53m 52s	1/4	PROCS OK - 43 processes with STATE = RSZDT

Results 1 - 16 of 16 Matching Services

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

Experiment 11

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

Steps:

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

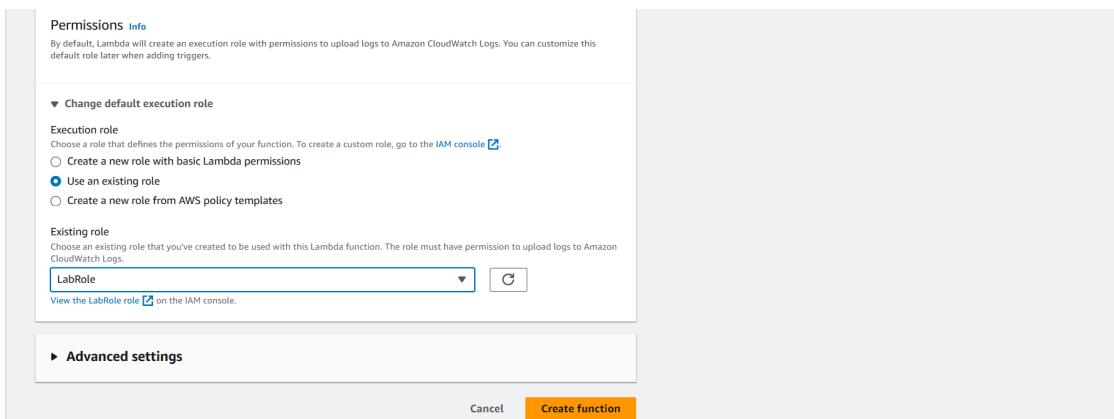
The screenshot shows the AWS Lambda service in the AWS Management Console. The left sidebar has 'Functions' selected under the 'Lambda' category. The main area displays a table of existing Lambda functions with columns for Function name, Description, Package type, Runtime, and Last modified. The functions listed are:

- ModLabRole: updates LabRole to allow it to assume itself. (Zip, Python 3.8, 2 months ago)
- RedshiftEventSubscription: Create Redshift event subscription to SNS Topic. (Zip, Python 3.8, 2 months ago)
- RoleCreationFunction: Create SLR if absent. (Zip, Python 3.8, 2 months ago)
- MainMonitoringFunction: - (Zip, Python 3.8, 2 months ago)

A 'Create function' button is visible at the top right of the table.

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

The screenshot shows the 'Create function' wizard in the AWS Lambda service. The 'Basic information' step is active. The 'Author from scratch' option is selected, and the function name is set to 'Lambda29'. The runtime is set to 'Node.js 20.x' and the architecture to 'x86_64'. Other options like 'Use a blueprint' and 'Container image' are also shown.



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

Your Lambda function gets created.

Successfully created the function Lambda29. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

Lambda > Functions > Lambda29

Lambda29

Throttle Copy ARN Actions ▾

Function overview [Info](#)

Description

Last modified 10 seconds ago

Function ARN arn:aws:lambda:us-east-1:663539227562:function:Lambda29

Function URL [Info](#)

Code Test Monitor Configuration Aliases Versions

Code source [Info](#)

Upload from ▾

Code Test Monitor Configuration Aliases Versions

Code source [Info](#)

File Edit Find View Go Tools Window Test Deploy

index.mjs Environment Var

```
1 export const handler = async (event) => {
2 // TODO implement...
3 const response = {
4 statusCode: 200,
5 body: JSON.stringify('Hello from Lambda!'),
6 };
7 return response;
8 };
9 
```

Upload from ▾

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

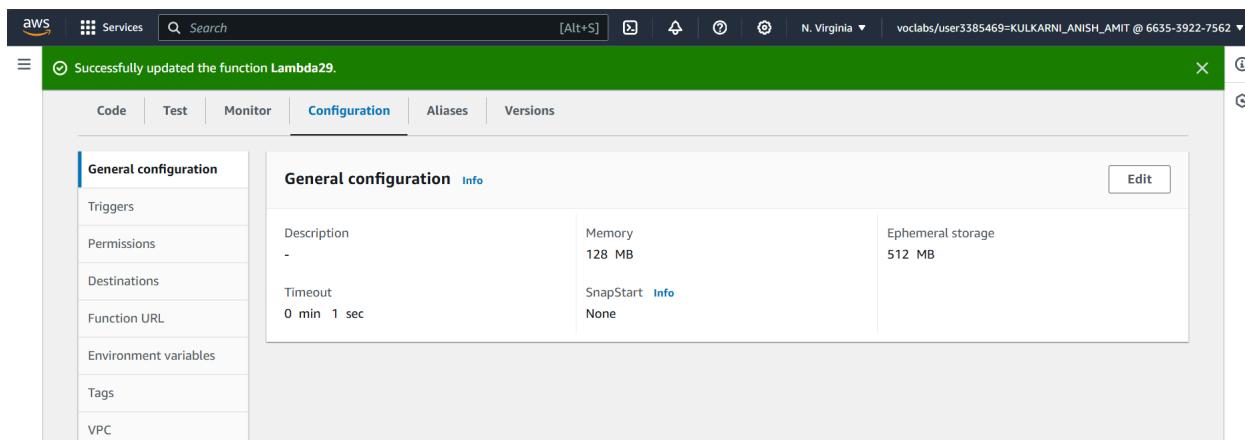
The screenshot shows the AWS Lambda 'Configuration' tab. On the left, a sidebar lists options: General configuration, Triggers, Permissions, Destinations, Function URL, Environment variables, Tags, VPC, and RDS databases. The 'General configuration' section is selected. In the main area, there's a table with two rows. The first row contains 'Description' (empty), 'Memory' (128 MB), and 'Ephemeral storage' (512 MB). The second row contains 'Timeout' (0 min 3 sec) and 'SnapStart' (None). A blue box highlights the 'Timeout' input field.

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

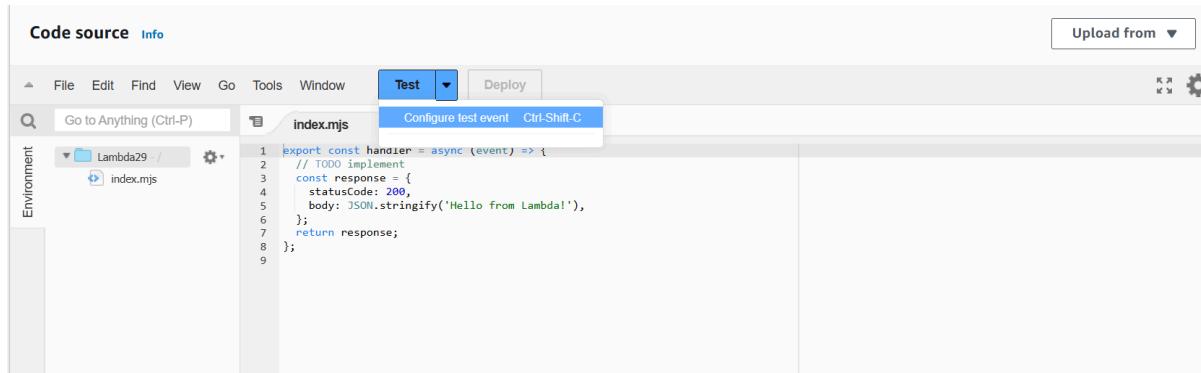
The screenshot shows the 'Edit basic settings' page for a Lambda function. Under the 'Basic settings' tab, the 'Timeout' field is set to '0 min 1 sec'. Other settings shown include Memory (128 MB), Ephemeral storage (512 MB), and SnapStart (None). The 'Execution role' section shows 'Use an existing role' selected with 'LabRole' chosen. At the bottom, there are 'Cancel' and 'Save' buttons, with 'Save' being highlighted.

After making the required changes, click on ‘Save’.

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



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



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

Configure test event

A test event is a JSON object that mocks the structure of requests emitted by AWS services to invoke a Lambda function. Use it to see the function's invocation result.

To invoke your function without saving an event, configure the JSON event, then choose Test.

Test event action

Create new event Edit saved event

Event name

Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Event sharing settings

Private

This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)

Shareable

This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

Template - optional

Event JSON

```

1 [ ]
2   "key1": "value1",
3   "key2": "value2",
4   "key3": "value3"
5 []

```

[Format JSON](#)

Cancel [Invoke](#) **Save**

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

The screenshot shows the AWS Lambda function configuration page. The 'Code' tab is selected. In the center, there's a code editor with a file named 'index.mjs'. Below the editor, the 'Execution result' section displays the following details:

- Test Event Name:** LambdaEvent29
- Status:** Succeeded
- Max memory used:** 62 MB
- Time:** 4.42 ms
- Response:**

```
{
  "statusCode": 200,
  "body": "\"Hello from Lambda!\""
}
```
- Function Logs:**

```
START RequestId: 43bb4831-6bfe-4b73-b913-eb5fd6493fbe Version: $LATEST
END RequestId: 43bb4831-6bfe-4b73-b913-eb5fd6493fbe
REPORT RequestId: 43bb4831-6bfe-4b73-b913-eb5fd6493fbe Duration: 4.42 ms Billed Duration: 5 ms Memory Size: 128 MB Max Memory Used: 62 MB Init Duration: 0 ms
```
- Request ID:** 43bb4831-6bfe-4b73-b913-eb5fd6493fbe

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

The screenshot shows the AWS Lambda function configuration page. The 'Code' tab is selected. In the code editor, the 'index.mjs' file has been modified:

```

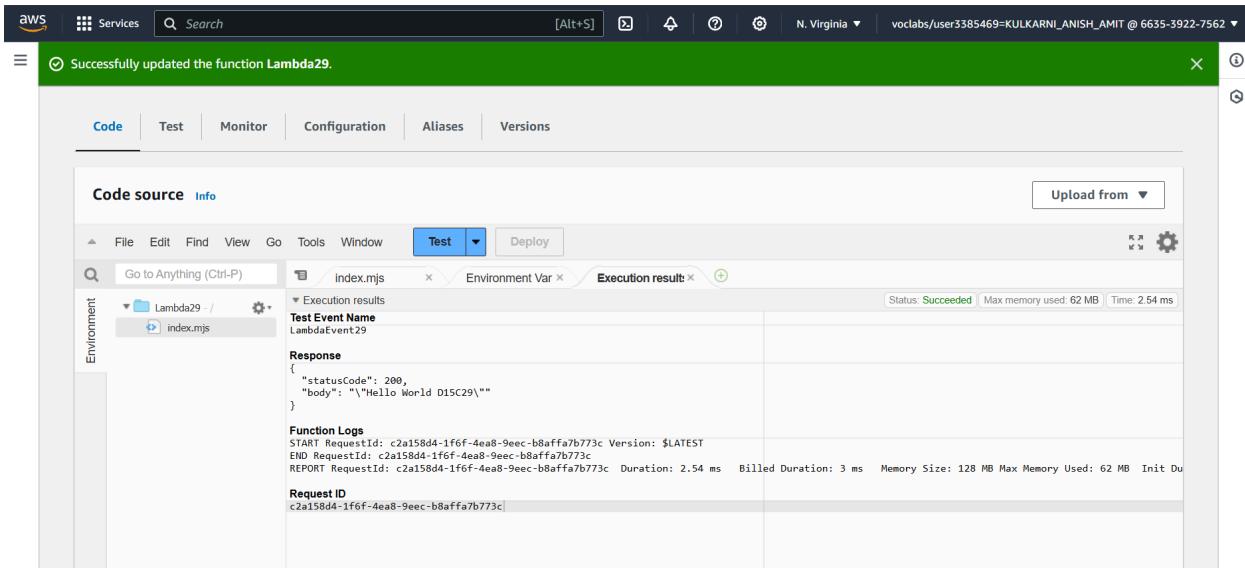
1 const test = "Hello World DISC29"
2 export const handler = async (event) => {
3   // TODO implement
4   const response = {
5     statusCode: 200,
6     body: JSON.stringify(test),
7   };
8   return response;
9 };
10

```

The 'Test' button is highlighted in blue at the top of the interface. The status bar at the bottom indicates 'Changes not deployed'.

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

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



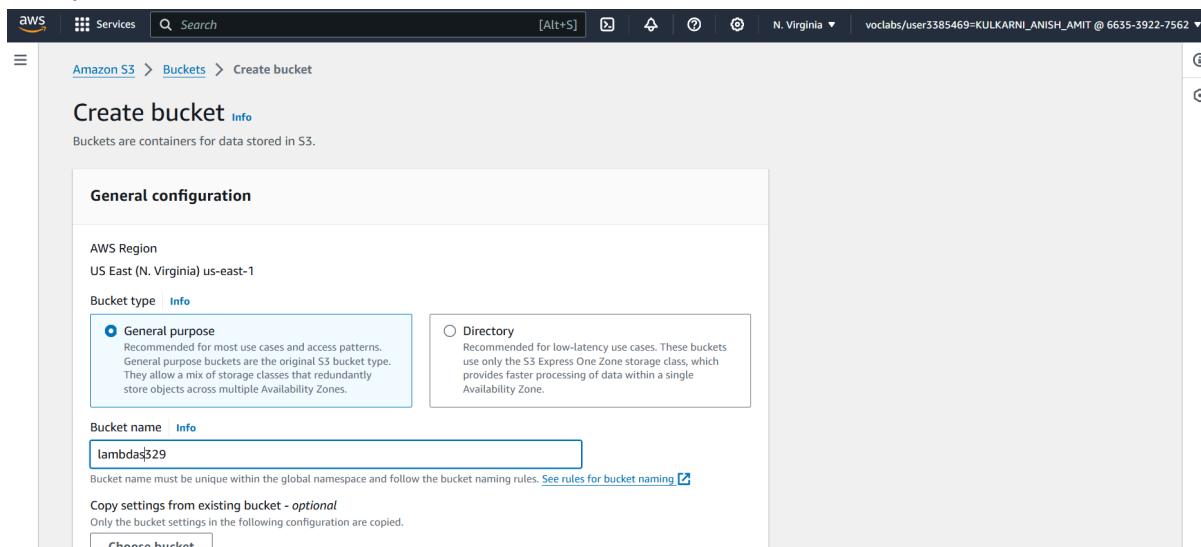
Conclusion: In this experiment, we understood the working of AWS Lambda service by creating and configuring a Lambda function using Node.js. We learned how to set up a Lambda function, change its configurations and test its functioning by creating test events for the function. From the output of the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

Experiment 12

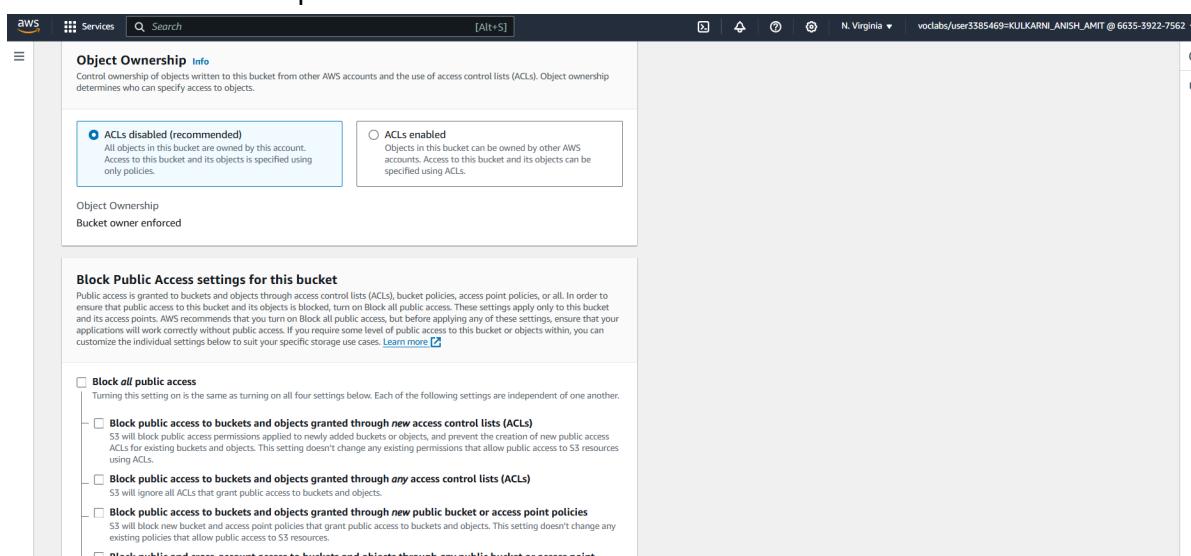
Aim: To create a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3.

Steps:

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



Uncheck the ‘Block all public access’ box.



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

Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

The screenshot shows the AWS S3 service page. At the top, a green banner indicates that a bucket named "lambdas329" has been successfully created. Below this, there's an "Account snapshot" section with an "updated every 24 hours" message and a "View Storage Lens dashboard" button. Under the "General purpose buckets" tab, a table lists two buckets: "elasticbeanstalk-us-east-1" and "lambdas329". Both buckets are located in "US East (N. Virginia) us-east-1". The "lambdas329" bucket was created on September 30, 2024, at 14:41:52 UTC+05:30.

Name	AWS Region	IAM Access Analyzer	Creation date
elasticbeanstalk-us-east-1 663539227562	US East (N. Virginia) us-east-1	View analyzer for us-east-1	August 5, 2024, 13:54:08 (UTC+05:30)
lambdas329	US East (N. Virginia) us-east-1	View analyzer for us-east-1	September 30, 2024, 14:41:52 (UTC+05:30)

Your bucket is created.

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

The screenshot shows the "Upload" interface for the "lambdas329" bucket. It includes a large central area for dragging and dropping files, and a "Files and folders" table showing one file: "html image 1.jpg". The "Destination" section shows the target path as "s3://lambdas329".

Name	Folder	Type
html image 1.jpg	-	image/jpg

Name: Anish Kulkarni

Roll No.: 29

Class: D15C

AY: 2024-25

The screenshot shows the AWS S3 console with a green header bar indicating 'Upload succeeded'. Below it, a summary table shows one file uploaded to 's3://lambdas329' with a size of 75.1 KB and a status of 'Succeeded'. A table below lists the uploaded file: 'html image...' (image/jpeg, 75.1 KB, Succeeded).

Files and folders (1 Total, 75.1 KB)					
<input type="text"/> Find by name					
Name	Folder	Type	Size	Status	Error
html image...	-	image/jpeg	75.1 KB	Succeeded	-

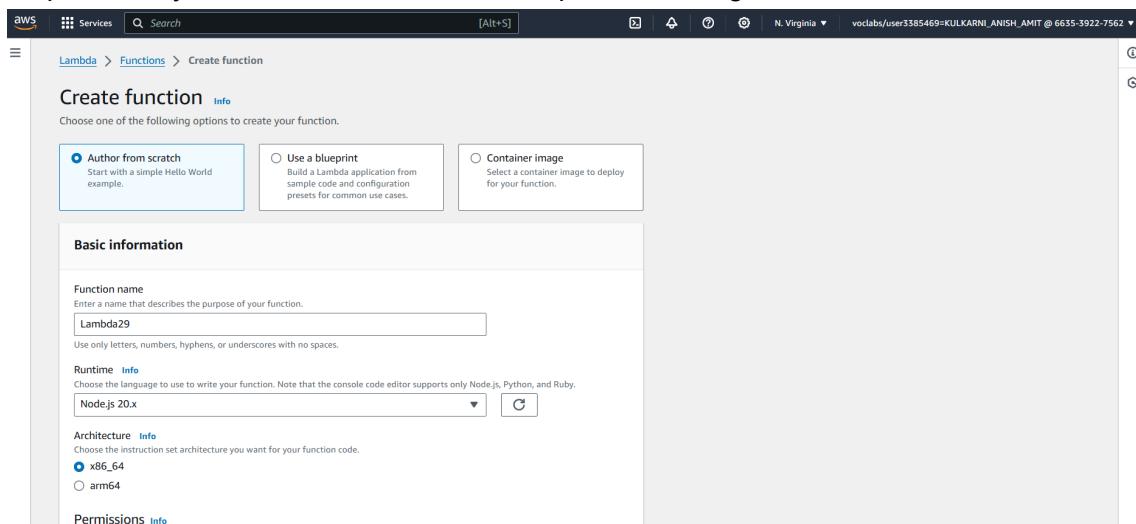
Your image gets uploaded onto the S3 bucket.

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

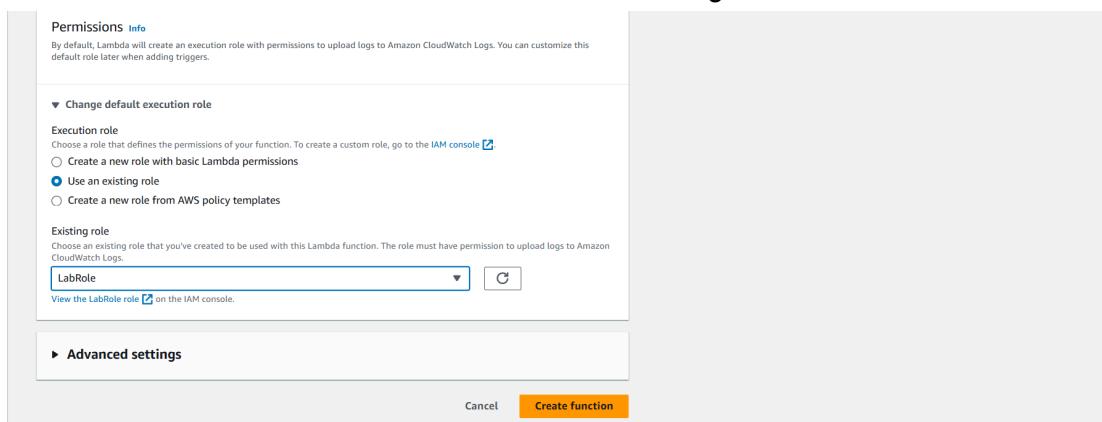
The screenshot shows the AWS Lambda console under the 'Functions' section. It displays a list of six functions, including 'ModLabRole' and 'RedshiftEventSubscription'. The 'ModLabRole' function is highlighted.

Function name	Description	Package type	Runtime	Last modified
ModLabRole	updates LabRole to allow it to assume itself	Zip	Python 3.8	2 months ago
RedshiftEventSubscription	Create Redshift event subscription to SNS Topic.	Zip	Python 3.8	2 months ago

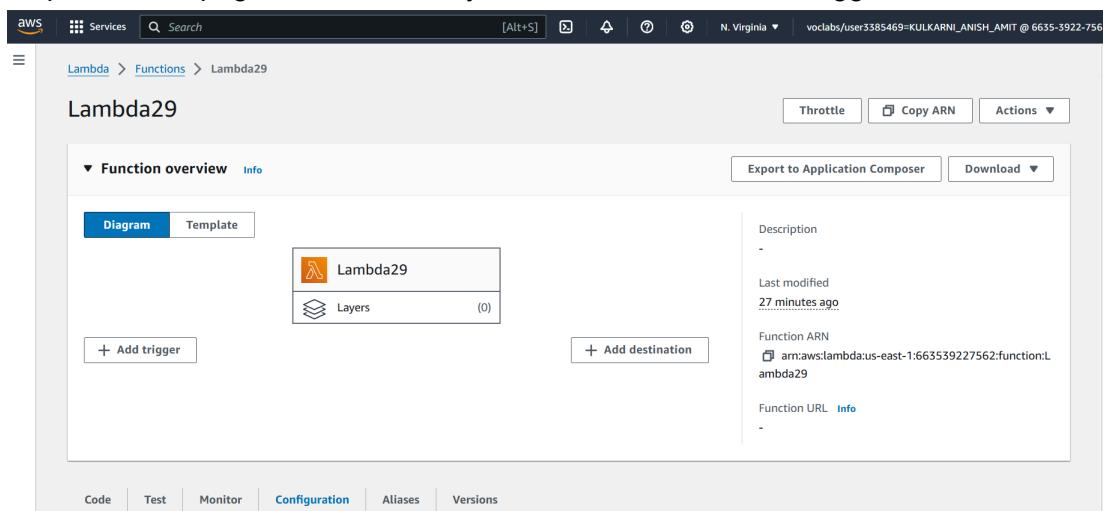
Step 4: Give your function a name and keep other settings as default.



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



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



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

The screenshots illustrate the steps to add an S3 trigger to a Lambda function:

- Screenshot 1: Trigger configuration (Info)**
Shows the 'Trigger configuration' step. The 'Bucket' dropdown is set to 'S3'. The 'Bucket' input field contains 's3://lambdas329'. The 'Event types' section has 'All object create events' selected. The 'Prefix - optional' and 'Suffix - optional' fields are empty.
- Screenshot 2: Lambda29 Function Overview**
Shows the Lambda function 'Lambda29' with its configuration tab selected. It lists an 'S3' trigger associated with the bucket 'lambdas329'. The 'Function URL' is shown as 'arn:aws:lambda:us-east-1:663539227562:function:Lambda29'.
- Screenshot 3: Lambda29 Configuration Triggers**
Shows the 'Triggers' section of the Lambda function configuration. It lists one trigger named 'S3: lambdas329' which points to the bucket 'lambdas329'.

The trigger gets successfully added to your function.

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

```

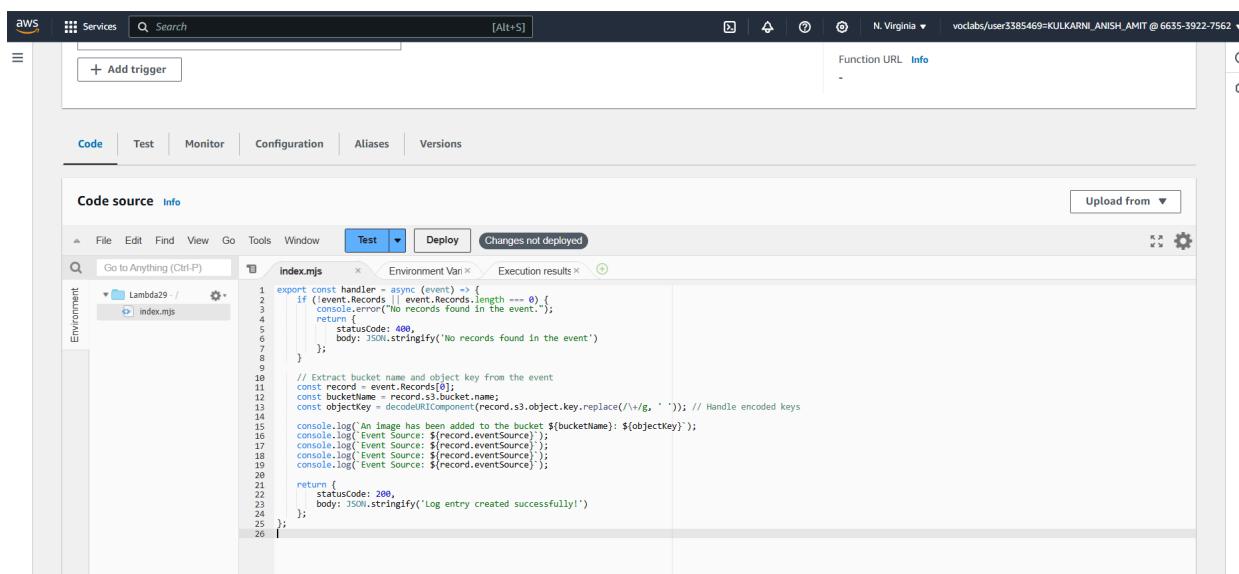
export const handler = async (event) => {
    if (!event.Records || event.Records.length === 0) {
        console.error("No records found in the event.");
        return {
            statusCode: 400,
            body: JSON.stringify('No records found in the event')
        };
    }

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

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

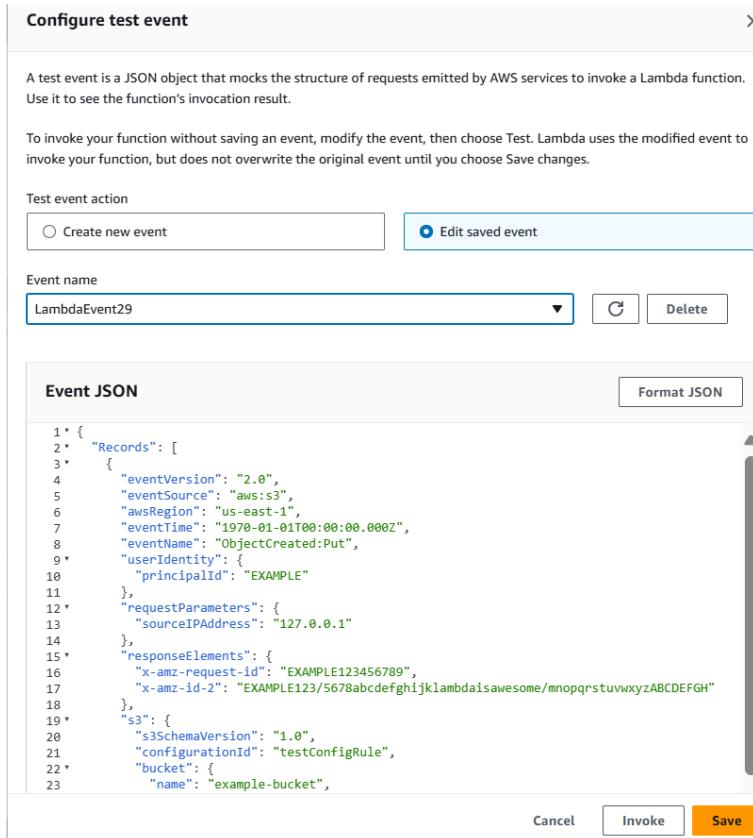
    return {
        statusCode: 200,
        body: JSON.stringify('Log entry created successfully!')
    };
}

```



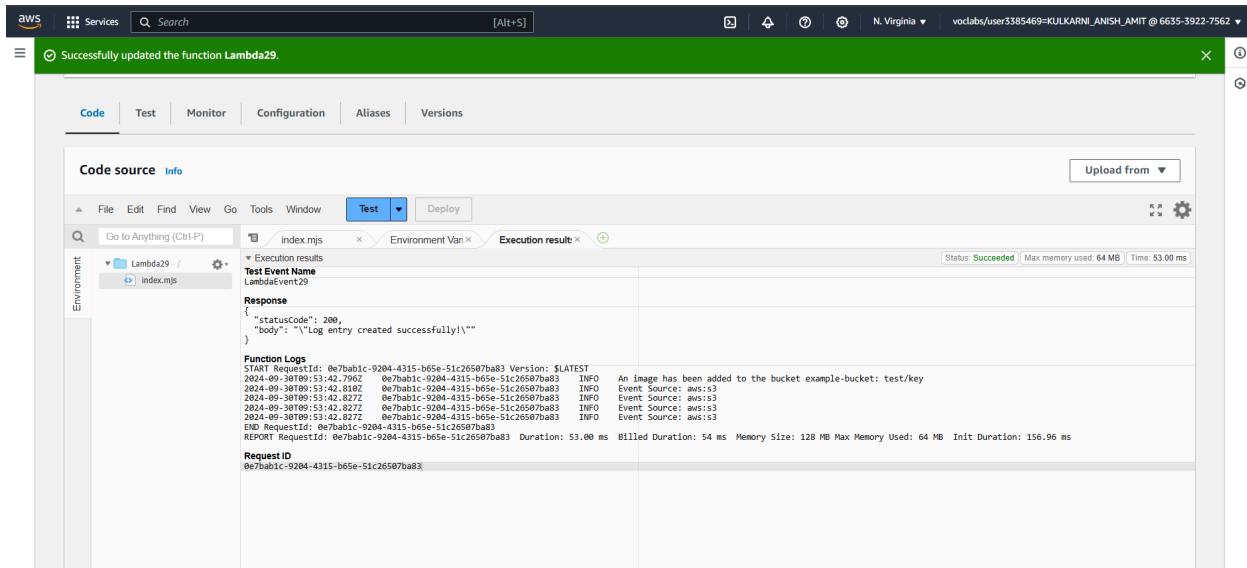
Step 8: Click on the arrow next to the 'Test' button and click on 'Configure test event'. In the popup box that appears, if you have an existing event, enter the name of your event or create a new event and in the 'Event JSON' section, paste the following code:-

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



Then, click on 'Save'. Your function gets successfully updated.

Step 9: Click on 'Deploy' and after deployment is successful, click on 'Test'.



Running the test gives the above output which displays that 'An Image has been added to the bucket' and that the log entry was successfully created.

Log events			Actions	Start tailing	Create metric filter					
Filter events - press enter to search		Clear	1m	30m	1h	12h	Custom	UTC timezone	Display	⋮
▶	Timestamp	Message								
		There are older events to load. Load more								
▶	2024-09-20T09:24:48.172Z	INIT_START Runtime Version: nodejs18.x35 Runtime Version ARN: arn:aws:lambda:us-east-1:111111111111:runtimeId:028af070644e00705701cf912080cc3dc3996012644e2717Fcc6f4267107022								
▶	2024-09-20T09:24:48.312Z	START RequestID: 01723939-7200-4210-a850-432105575405 Version: \$LATEST								
▶	2024-09-20T09:24:48.314Z	2024-09-20T09:24:48.314Z 01723939-7200-4210-a850-432105575405 INFO An image has been added to the bucket example-bucket: test/key								
▶	2024-09-20T09:24:48.314Z	2024-09-20T09:24:48.314Z 01723939-7200-4210-a850-432105575405 INFO Event Source: aws:s3								
▶	2024-09-20T09:24:48.934Z	2024-09-20T09:24:48.934Z 01723939-7200-4210-a850-432105575405 INFO Event Source: aws:s3								
▶	2024-09-20T09:24:48.934Z	2024-09-20T09:24:48.934Z 01723939-7200-4210-a850-432105575405 INFO Event source: aws:s3								
▶	2024-09-20T09:24:48.934Z	2024-09-20T09:24:48.934Z 01723939-7200-4210-a850-432105575405 INFO Event source: aws:s3								
▶	2024-09-20T09:24:48.934Z	END RequestId: 01723939-7200-4210-a850-432105575405 Duration: 20.56 ms Billed Duration: 20 ms Memory Size: 128 MB Max Memory Used: 64 MB Init Duration: 147.32 ms								
▶	2024-09-20T09:24:48.934Z	REPORT RequestId: 01723939-7200-4210-a850-432105575405 Duration: 20.56 ms Billed Duration: 20 ms Memory Size: 128 MB Max Memory Used: 64 MB Init Duration: 147.32 ms								
		No newer events at this moment. Auto retry paused. Resume								

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