

## Experiment 1(a)

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

- i) local machine or virtual machine
- ii) Amazon S3 Bucket

### **Static Hosting:**

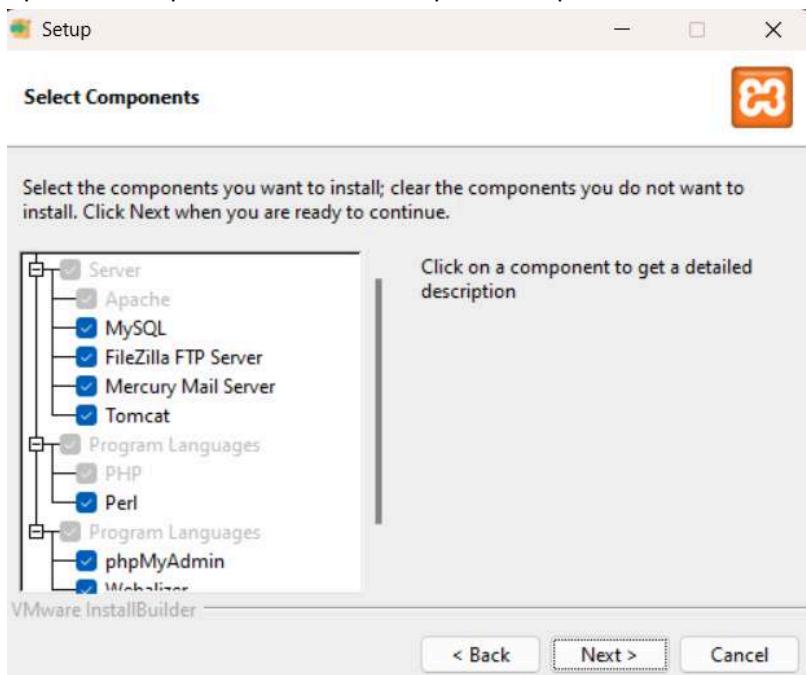
#### **1) On local server (XAMPP)**

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

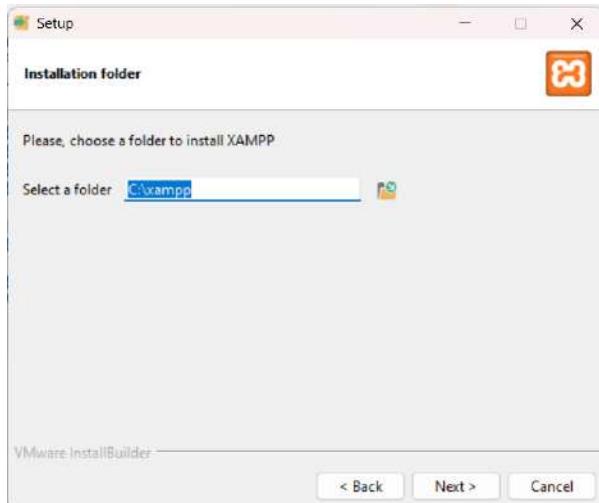
- 1) Select your OS. It will automatically start downloading.



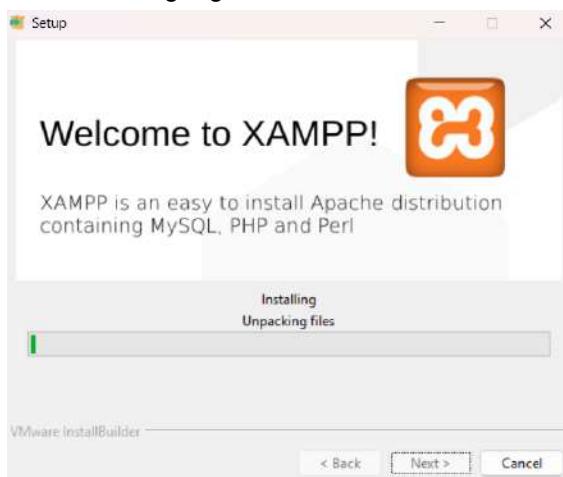
- 2) Open the setup file. Select all the required components and click next.



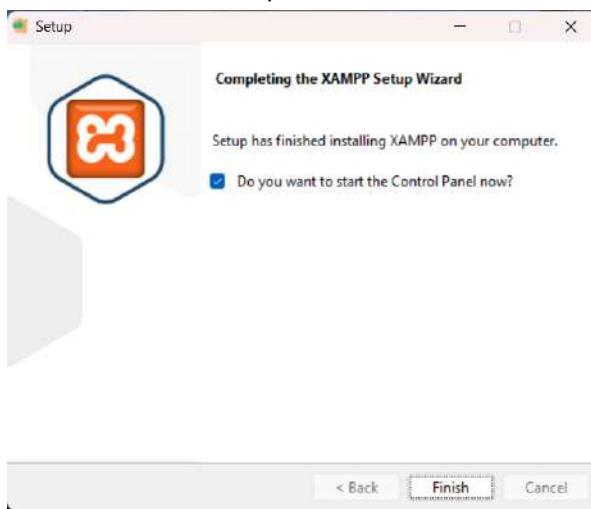
- 3) Choose the folder to install XAMPP in. Make sure the folder is empty. Click next



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



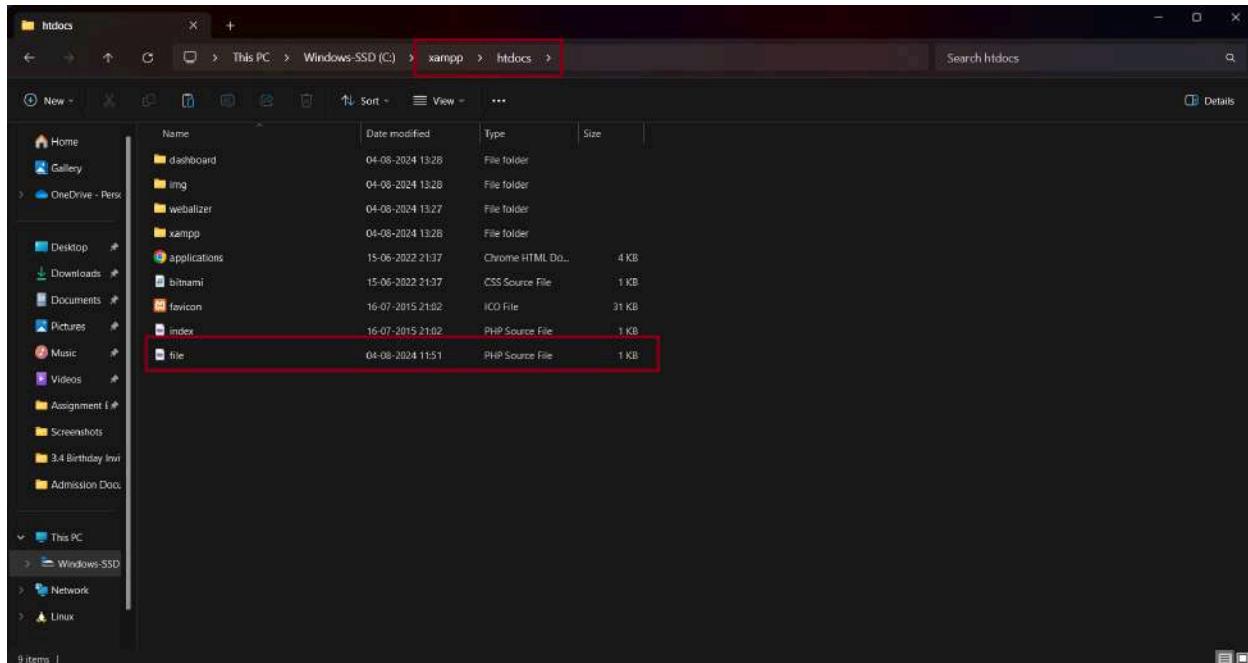
- 5) The installation is complete. Click Finish



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

Name	Date modified	Type	Size
<b>▼ Today</b>			
index - Copy	04-08-2024 11:51	PHP Source File	1 KB

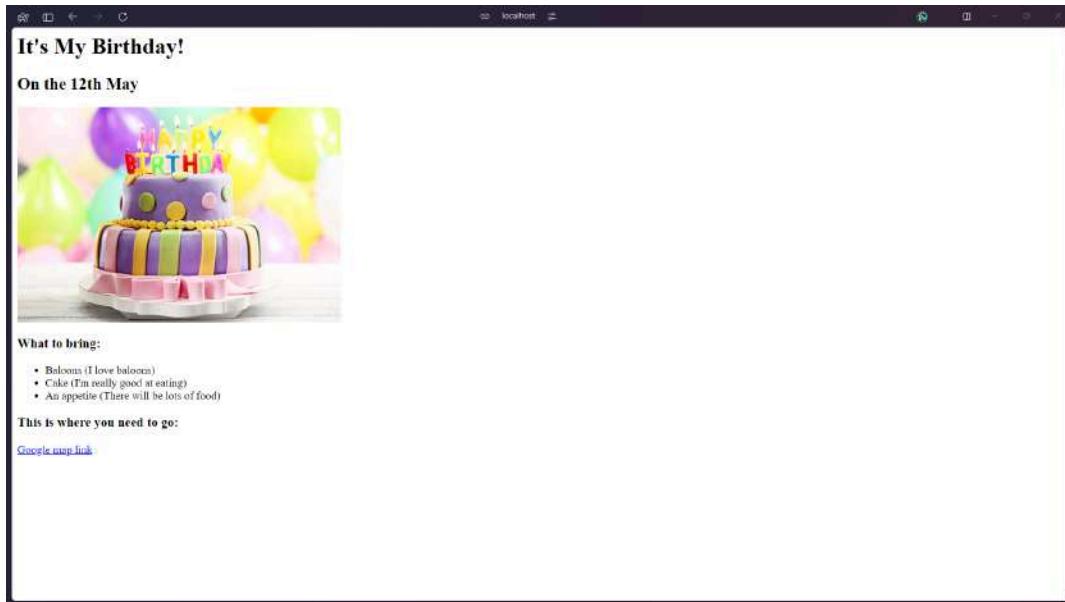
**Step 3:** Go to the directory where XAMPP was installed. Go to **htdocs** folder. Place your folder in this directory.



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

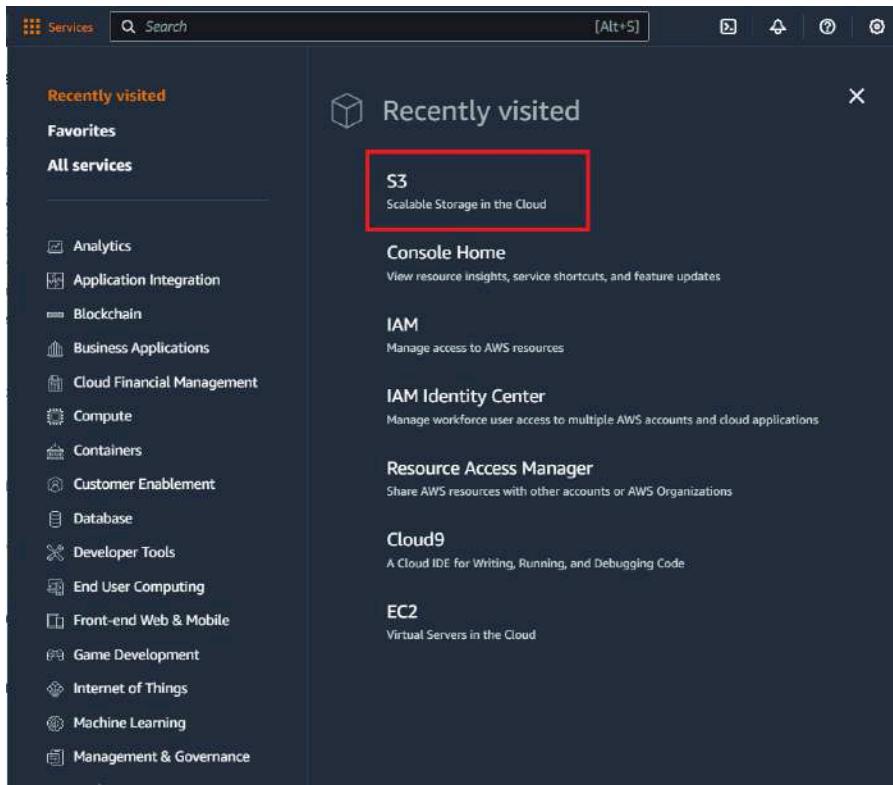


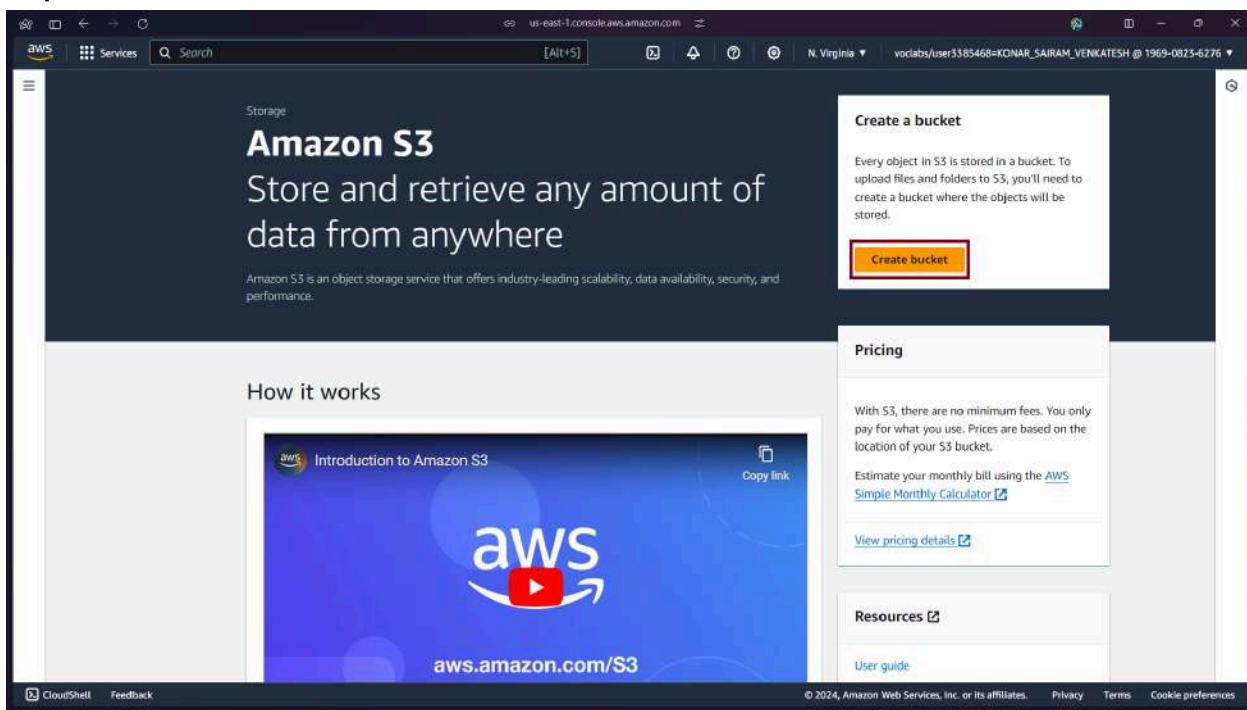
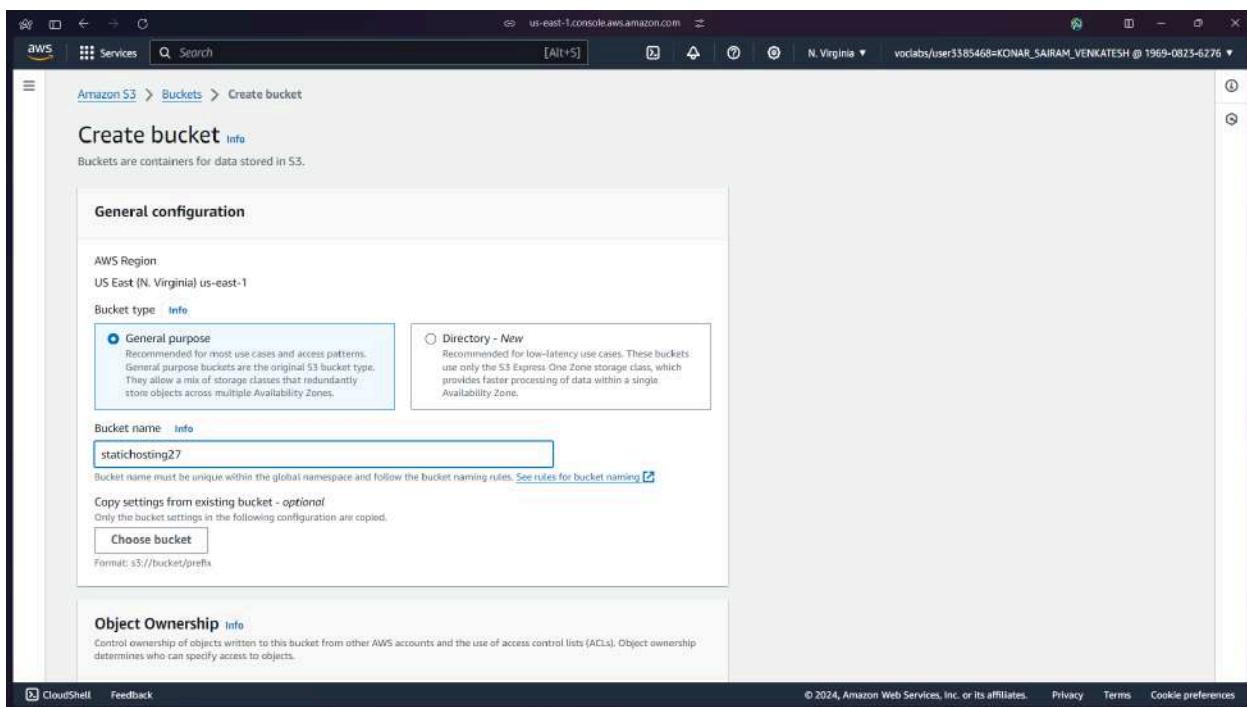
**Step 5:** Open your web browser. Type localhost/YOUR\_FILENAME.php. This will open your website on your browser.



## 2) AWS S3

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



**Step 2:** Click on Create Bucket.**Step 3:** Give a name to your bucket, keeping other options default, scroll down and click on Create Bucket.

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

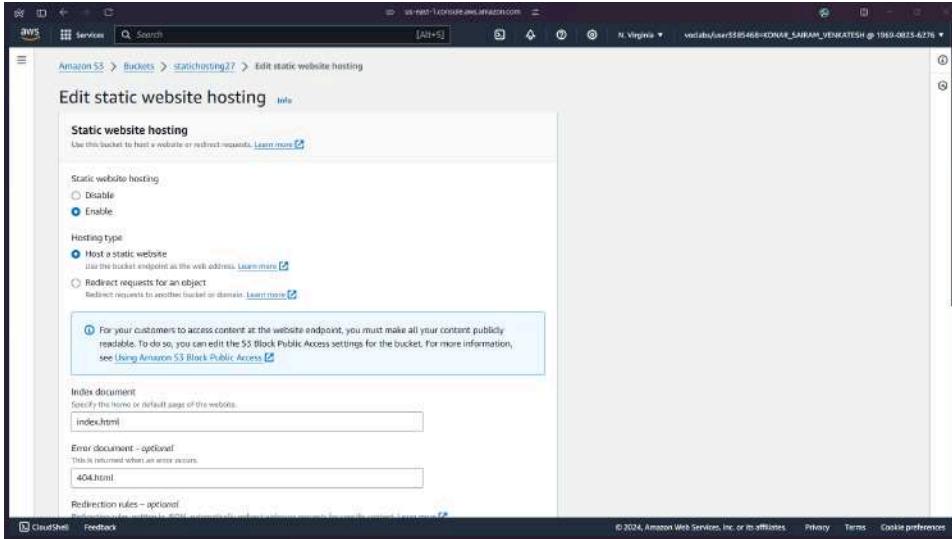
The screenshot shows the AWS S3 Buckets page. At the top, there's an account snapshot message: "Account snapshot - updated every 24 hours". Below it, there are tabs for "General purpose buckets" and "Directory buckets", with "General purpose buckets" selected. A search bar says "Find buckets by name". A table lists one bucket: "statichosting27" (Info), US East (N. Virginia) (Region), us-east-1 (Location), and a creation date of August 4, 2024, 15:30:03 (UTC+05:30). Buttons for "Copy ARN", "Empty", "Delete", and "Create bucket" are at the top right of the table.

The screenshot shows the "Properties" tab of the "statichosting27" bucket properties page. It has tabs for "Objects" (selected), "Properties", "Permissions", "Metrics", "Management", and "Access Points". Under "Objects", there's a table with columns for Name, Type, Last modified, Size, and Storage class. A message says "No objects" and "You don't have any objects in this bucket". A large orange "Upload" button is at the bottom.

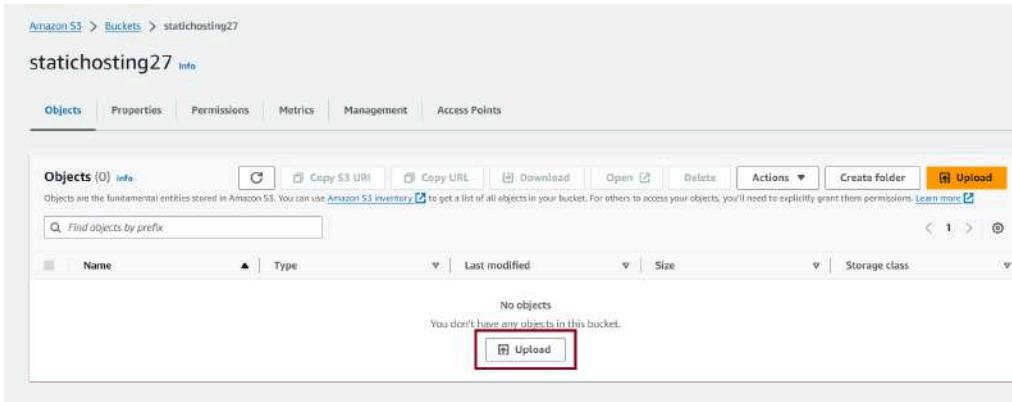
**Step 5:** Scroll down till you find Static website hosting, click on edit

The screenshot shows the "Properties" tab of the "statichosting27" bucket properties page. It includes sections for "Transfer acceleration", "Object Lock", "Requester pays", and "Static website hosting". The "Static website hosting" section has an "Edit" button highlighted with a red box. A note below it says "For this feature to take effect on all new requests, Learn more [Link]".

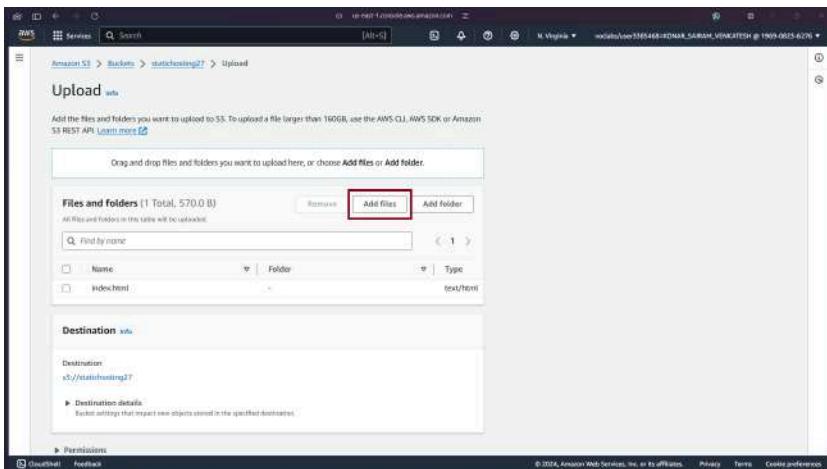
**Step 6:** Enable static website hosting, in Index document, write the name of your document and in error document, give name as 404.html. Save your changes.



**Step 7:** Go to Objects tab and click on upload file.



**Step 8:** Click on Add files. Add all the files you want to upload. Then scroll down and click on Upload.



**Step 9:** This will take you to the Objects screen. Switch to Properties and scroll down to Static Website Hosting. There you would find the link (Bucket website endpoint) to your website.

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://statichosting27.s3-website-us-east-1.amazonaws.com>

**Step 10:** Open the link. It will show a 403 forbidden error screen as the contents of the bucket are not available for public users. To change this, go to Permissions tab, go to Block public access, and click on edit.

403 Forbidden

- Code: AccessDenied
- Message: Access Denied
- RequestId: 8TQ4EGP4TK06MVPB
- HostId: hF+ToadQUoCuDM8H+iFRsXdA28TGp+xikYb|b4CICS/t+3it4ihA/tvgA1Xrlxo+JL5AhkT6hJs=

An Error Occurred While Attempting to Retrieve a Custom Error Document

- Code: AccessDenied
- Message: Access Denied

**Step 11:** Uncheck the Block all public access checkbox and click on save changes.

Amazon S3 > Buckets > statichosting27 > Edit Block public access (bucket settings)

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

**Block public access (bucket settings)**

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its contents. If you require some level of public access, do not turn on Block all public access, but instead, applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

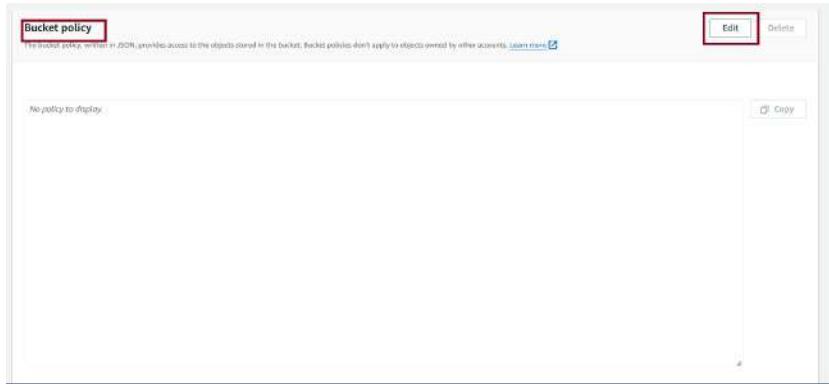
**Block all public access**

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

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

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

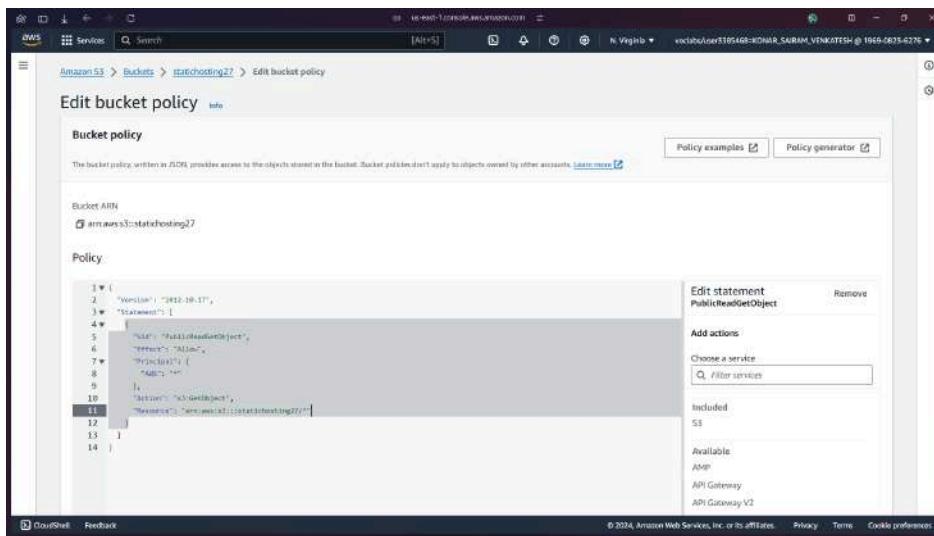
**Step 12:** Scroll down to bucket policy and click edit



**Step 13:**

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

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



**Step 14:** Now reload the website. You can see your website



### Conclusion:

In this experiment, we learned how to host a static website both on a local server using XAMPP and on AWS S3. By setting up XAMPP, we were able to deploy a local web server and access the hosted site through the browser using localhost. Additionally, we configured an S3 bucket for static website hosting on AWS, where we uploaded files, enabled public access, and applied the necessary bucket policies to make the website accessible. This hands-on process illustrated the differences between local and cloud hosting, emphasizing the ease of deployment and configuration flexibility that each method offers.

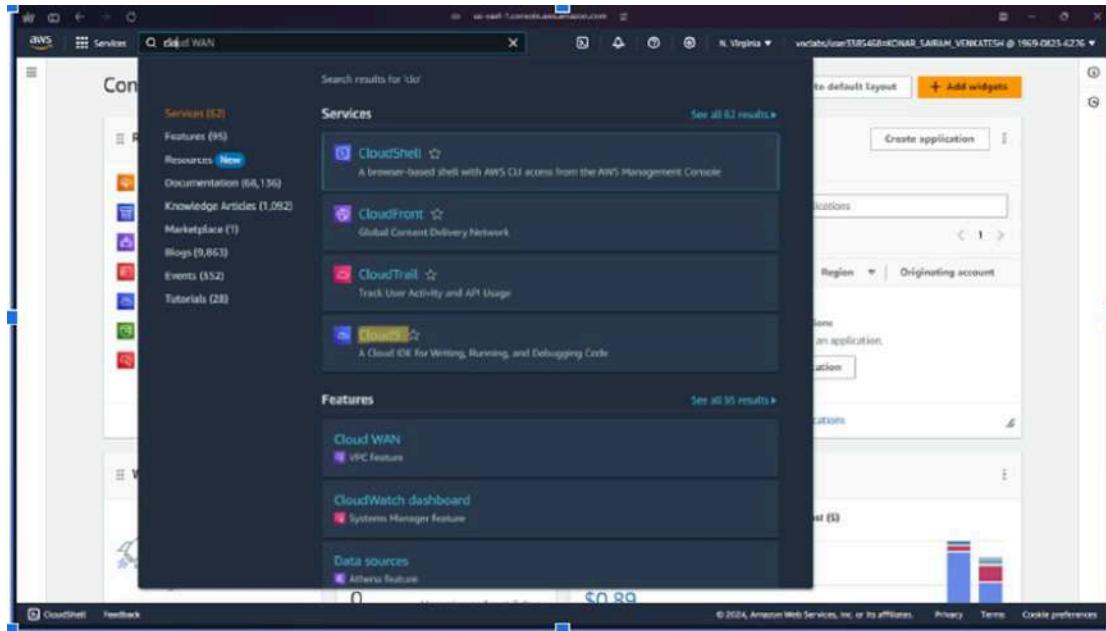
## Experiment 1(b)

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

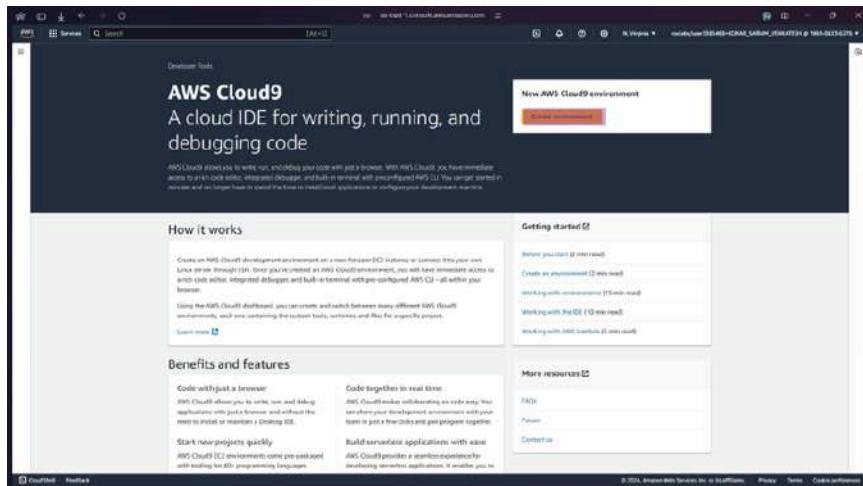
### Steps:

#### Step 1: Set up Cloud9 environment.

- 1) Search Cloud9 in the services tab and open it



- 2) Click on Create Environment



- 3) Give a name to your Cloud9 Environment. You can add a description if needed.

AWS Cloud9 > Environments > Create environment

### Create environment Info

**Details**

Name  Limit of 60 characters, alphanumeric, and unique per user.

Description - optional

Limit 200 characters.

Environment type Info  
Determines what the Cloud9 IDE will run on.

New EC2 Instance  
Cloud9 creates an EC2 instance in your account. The configuration of your EC2 instance cannot be changed by Cloud9 after creation.

Existing compute  
You have an existing instance or server that you'd like to use.

**New EC2 instance**

Instance type Info  
The memory and CPU of the EC2 instance that will be created for Cloud9 to run on.

t2.micro (1 GB RAM + 1 vCPU)  
Free tier eligible. Ideal for educational users and exploration.

- 4) Select the option new EC2 instance if you do not have one ready for the environment. Give the specifications of that EC2 instance ahead.

Services  Search [Alt+S]

Environment type Info  
Determines what the Cloud9 IDE will run on.

New EC2 instance  
Cloud9 creates an EC2 instance in your account. The configuration of your EC2 instance cannot be changed by Cloud9 after creation.

Existing compute  
You have an existing instance or server that you'd like to use.

**New EC2 instance**

Instance type Info  
The memory and CPU of the EC2 instance that will be created for Cloud9 to run on.

t2.micro (1 GB RAM + 1 vCPU)  
Free tier eligible. Ideal for educational users and exploration.

t3.small (2 GB RAM + 2 vCPUs)  
Recommended for small web projects.

m5.large (8 GB RAM + 2 vCPUs)  
Recommended for production and most general-purpose development.

Additional instance types  
Explore additional instances to fit your need.

Platform Info  
This will be installed on your EC2 instance. We recommend Amazon Linux 2023.

Amazon Linux 2023

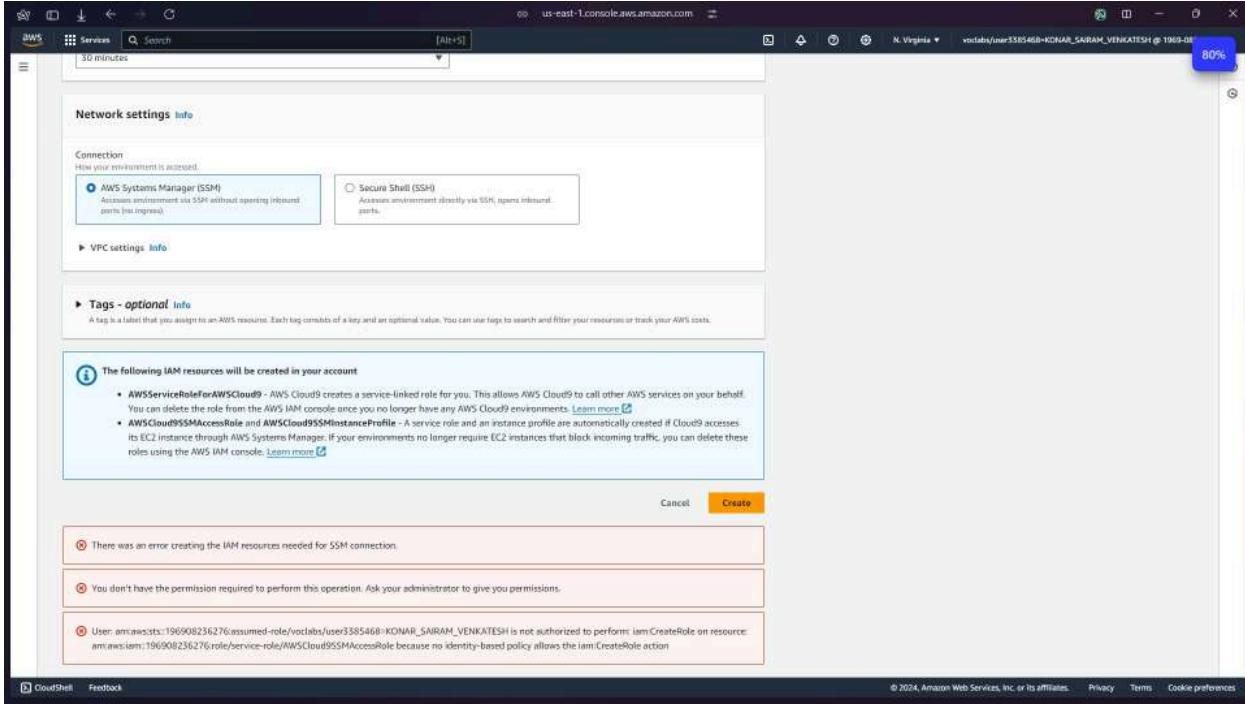
Timeout  
How long Cloud9 can be inactive (no user input) before auto-hibernating. This helps prevent unnecessary charges.

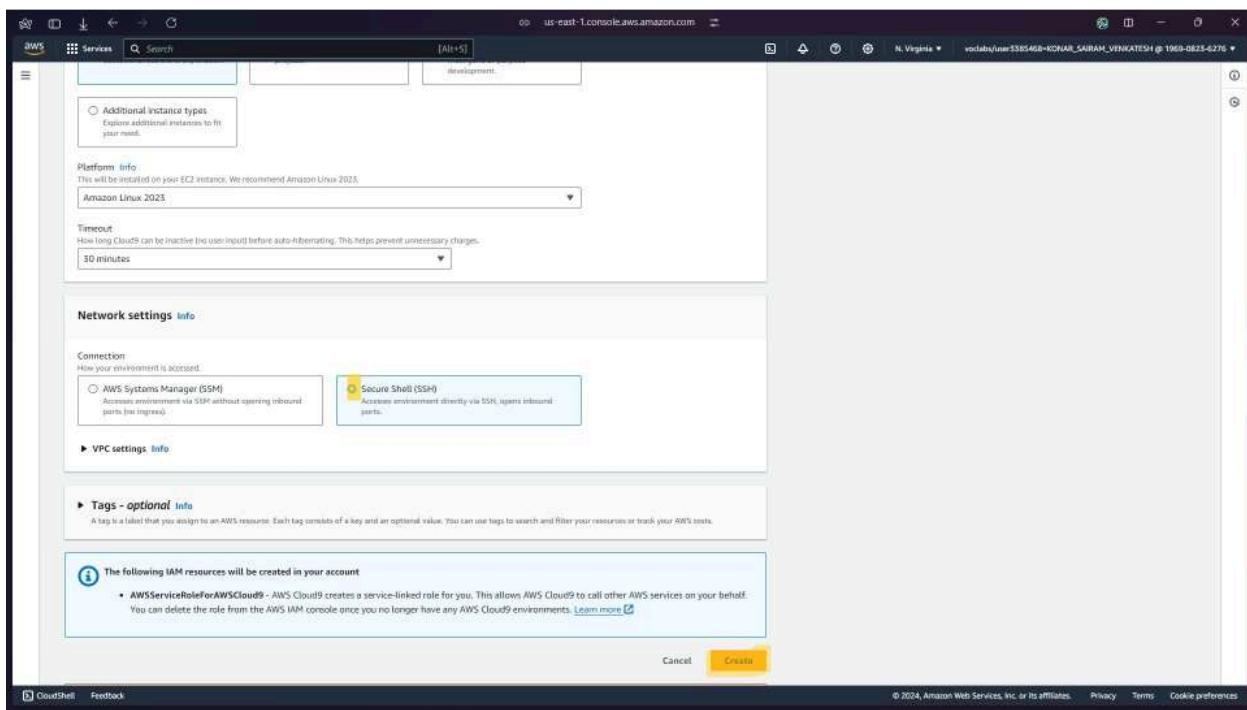
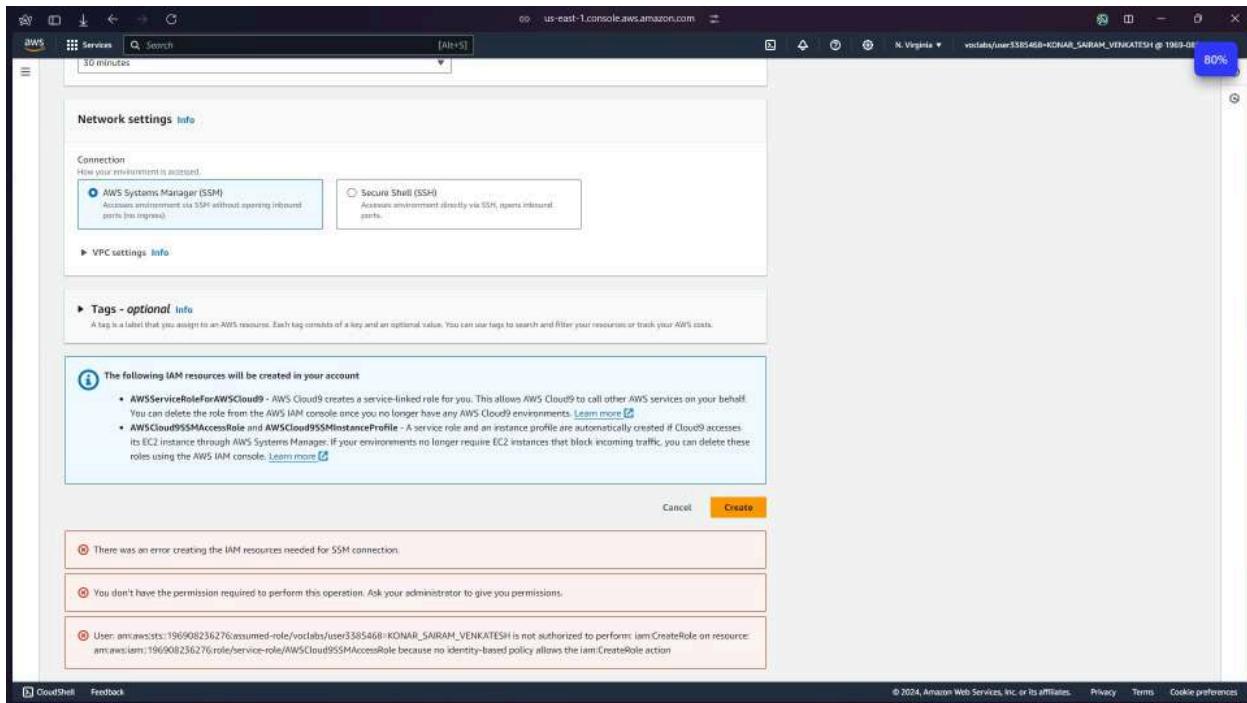
30 minutes

**Network settings Info**

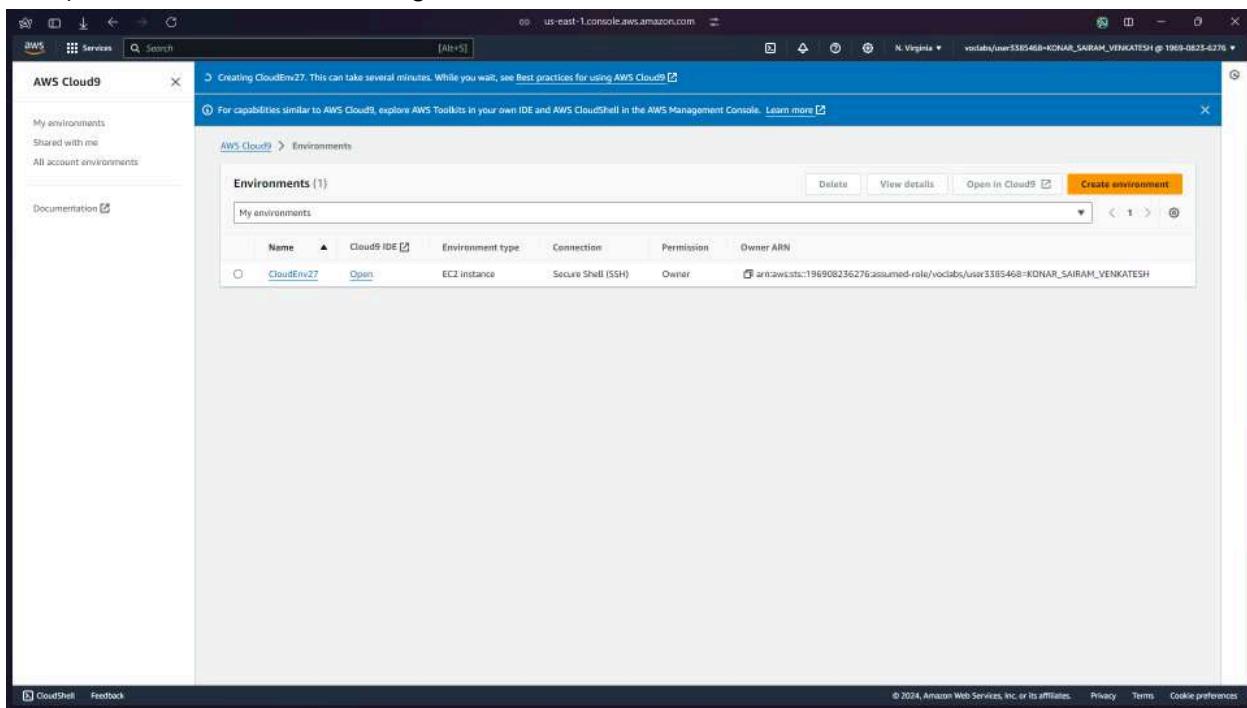
Connection  
How your environment is accessed.

- 5) On the AWS Academy account, if we select AWS System Manager (SSM) in Network settings, it gives an error as the account does not have permissions to use the setting. So we select Secure Shell (SSH). After that click on Create.



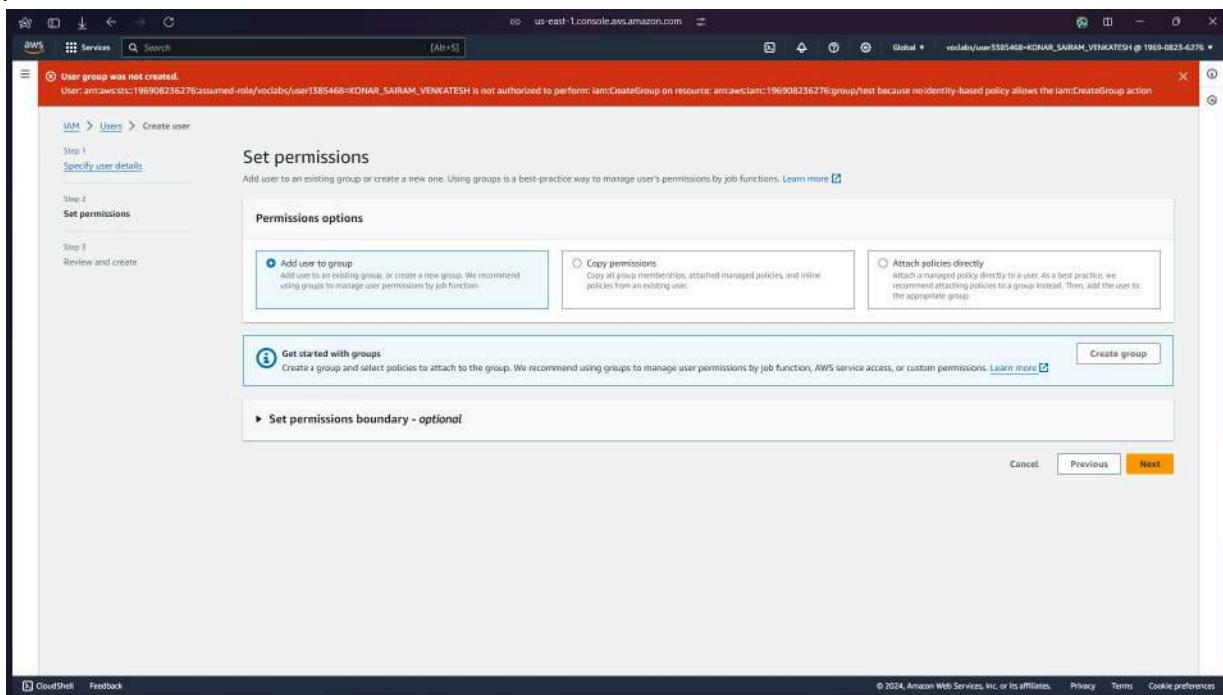


## 6) The environment is being created.

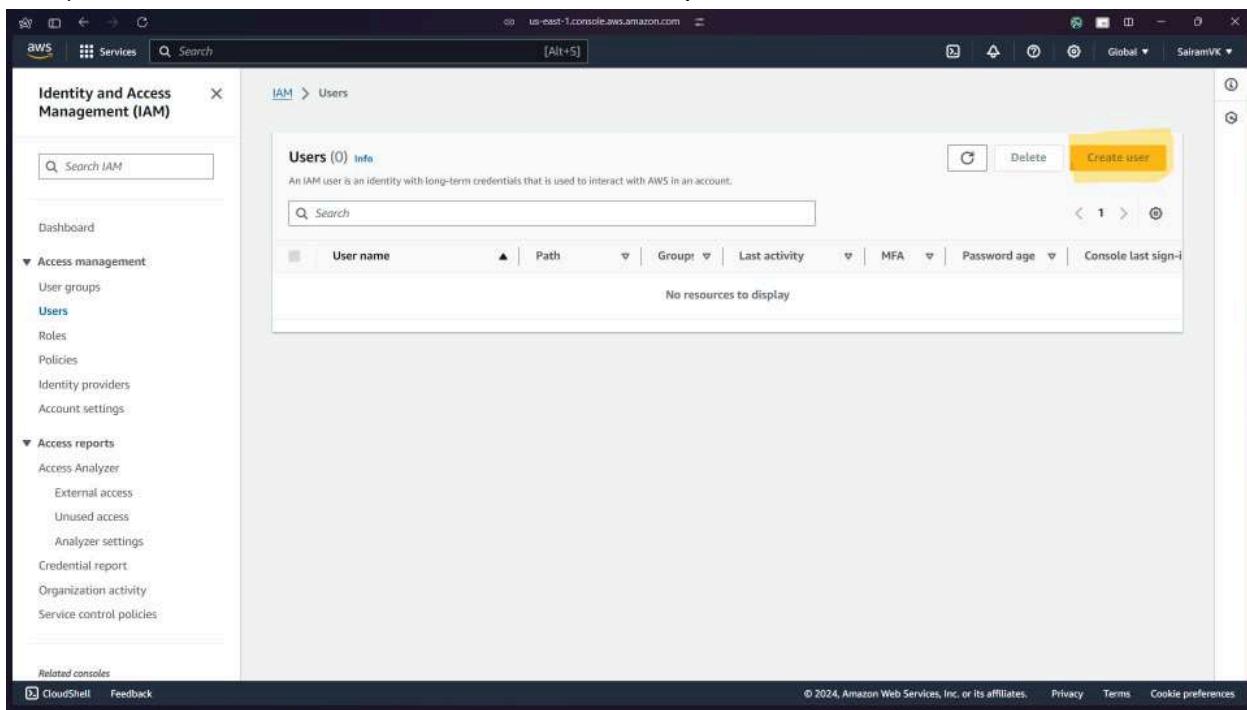


## Step 2: Creating IAM user.

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

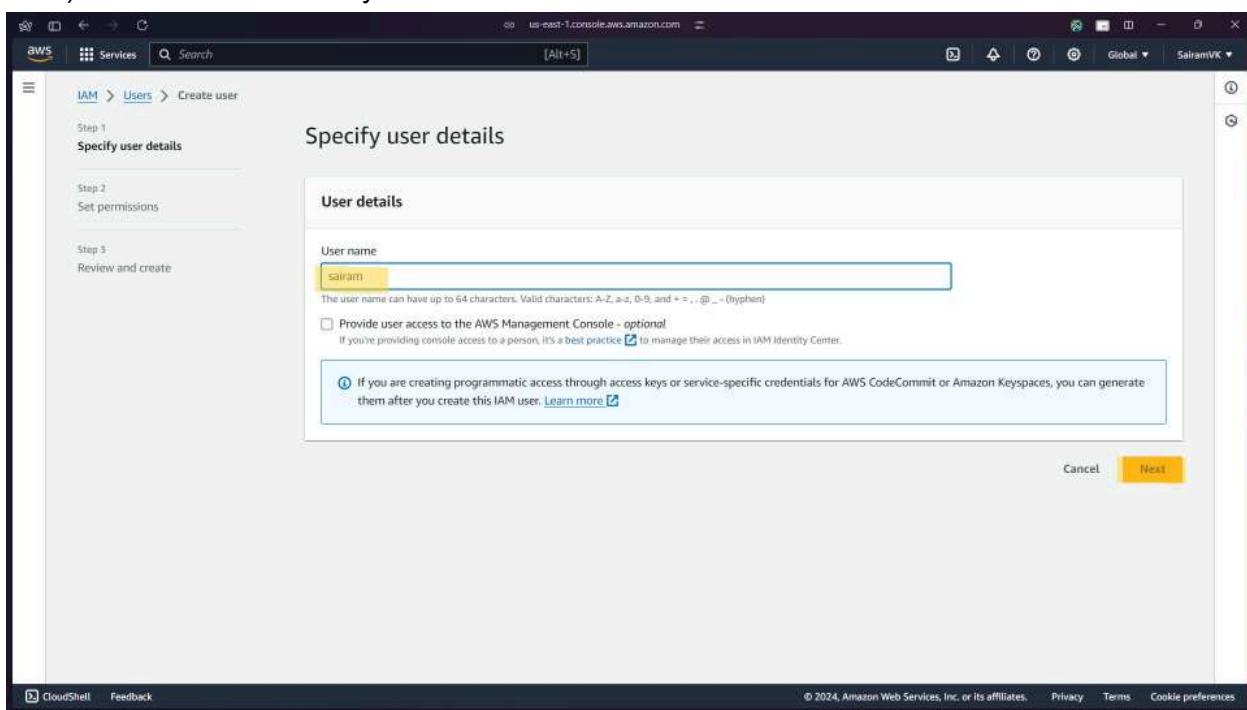


- 1) Search IAM on the services search bar and open it. Click on Create User.



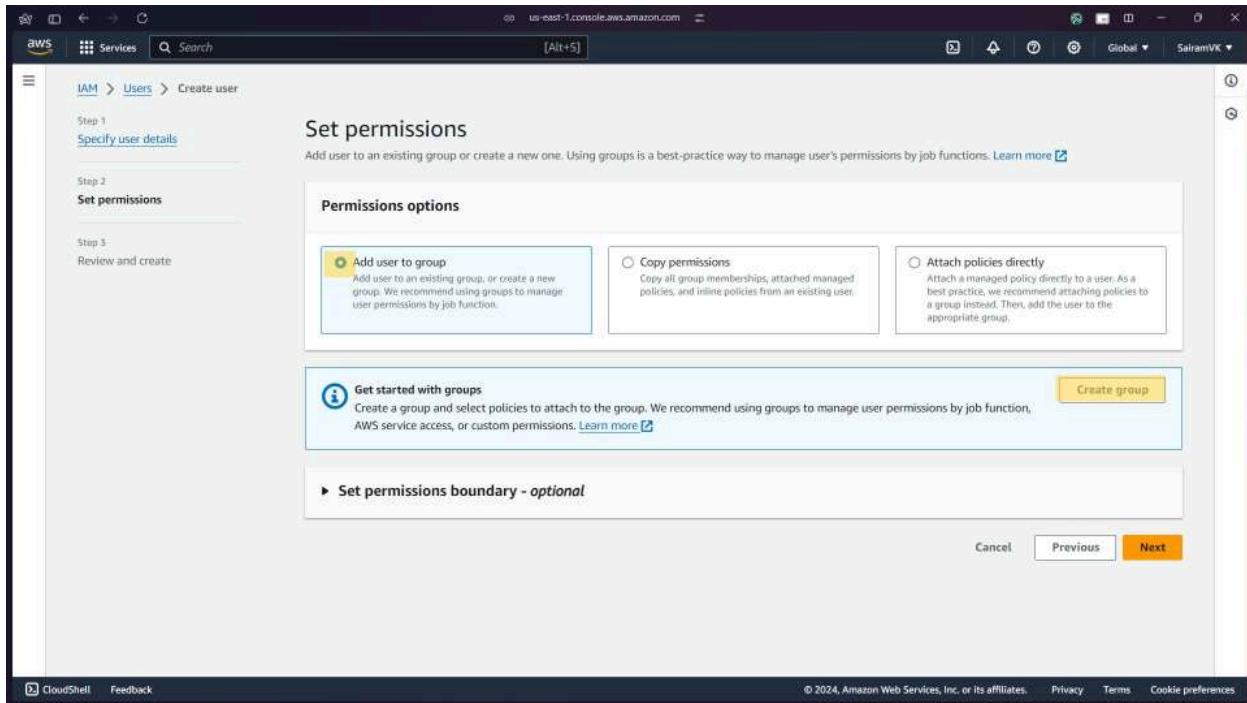
The screenshot shows the AWS Identity and Access Management (IAM) service in the AWS Management Console. The left sidebar is collapsed, and the main area displays the 'Users' page. At the top right, there are buttons for 'Create user' (highlighted with a yellow box), 'Delete', and other actions. Below the buttons, there is a search bar and a table header with columns: 'User name', 'Path', 'Group', 'Last activity', 'MFA', 'Password age', and 'Console last sign-in'. A message below the table states 'No resources to display'. On the far left, a detailed sidebar menu for IAM is visible, including sections for Access management, Access reports, and Related consoles. The bottom of the screen includes standard AWS navigation links like CloudShell and Feedback, along with copyright information for 2024.

- 2) Give a username to your user and click Next.

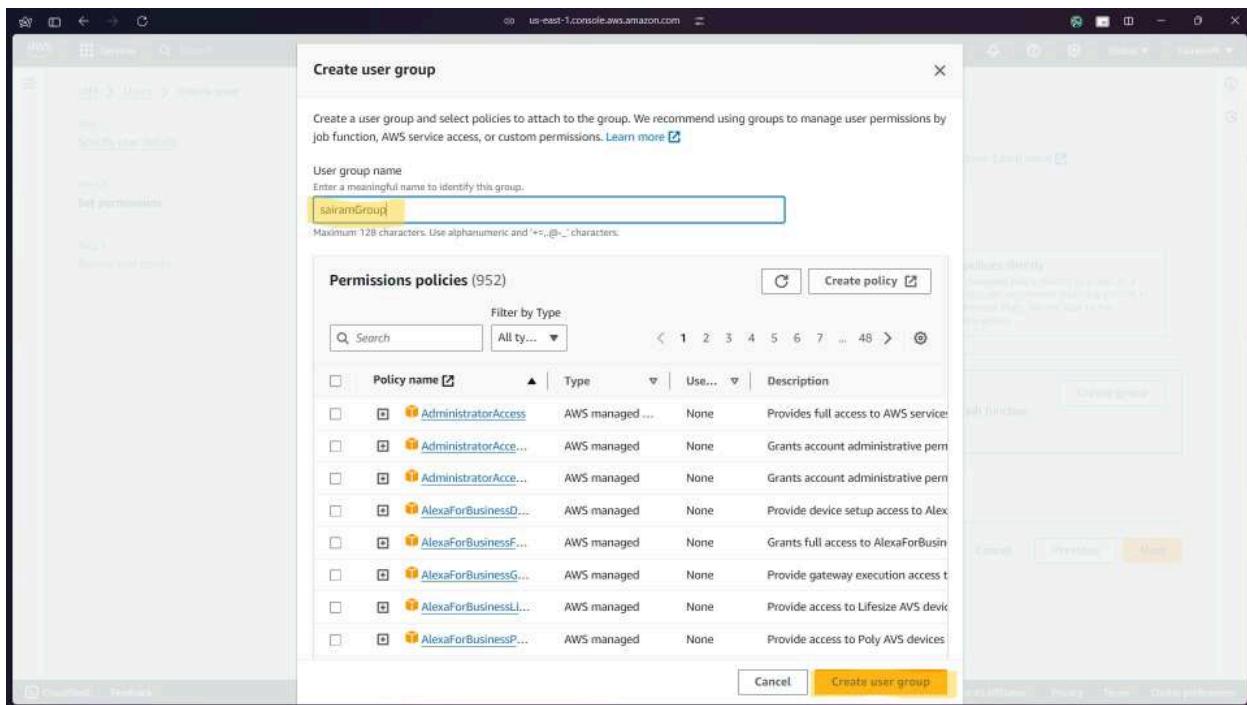


The screenshot shows the 'Specify user details' step of the 'Create user' wizard in the AWS Management Console. The left sidebar shows the steps: Step 1 (Specify user details, highlighted with a yellow box), Step 2 (Set permissions), and Step 3 (Review and create). The main area is titled 'Specify user details' and contains a 'User details' section. In the 'User name' field, the value 'sairam' is entered and highlighted with a yellow box. Below the field, a note specifies that the user name can have up to 64 characters and lists valid characters. There is also an optional checkbox for providing AWS Management Console access. A callout box provides instructions for generating programmatic access keys. At the bottom right, there are 'Cancel' and 'Next' buttons, with 'Next' also highlighted with a yellow box.

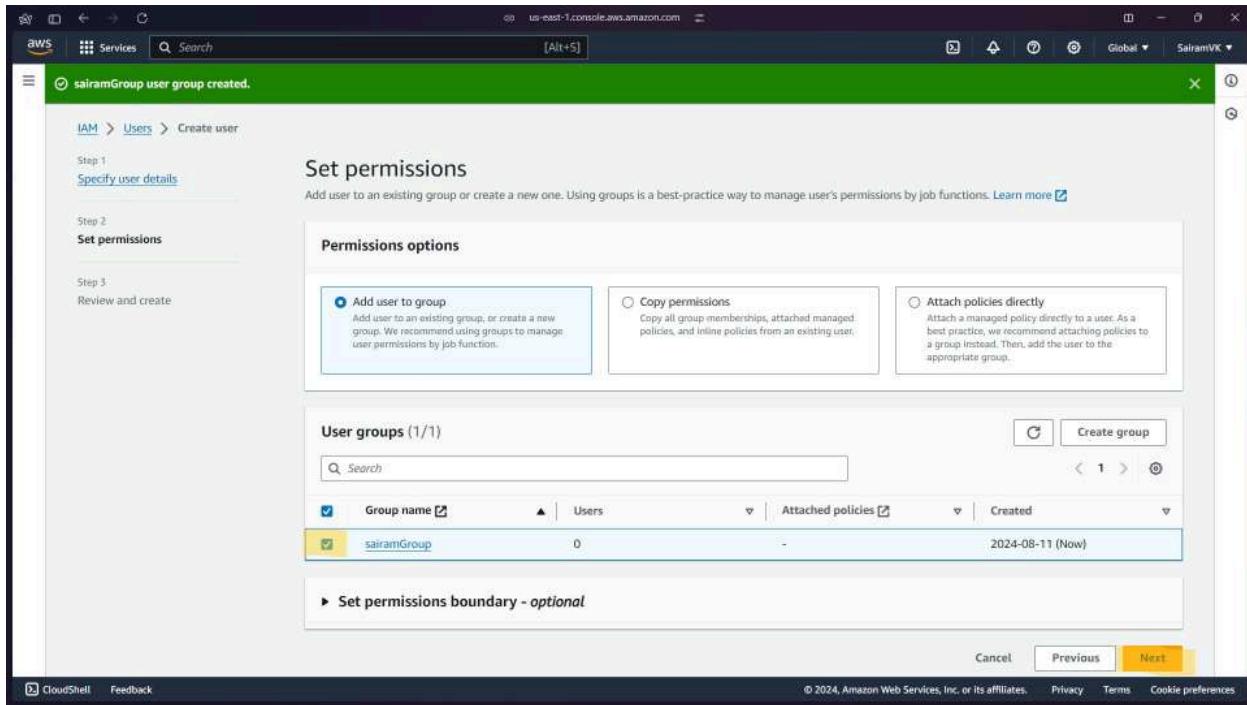
- 3) Select add User to Group. If there are no user groups on your accounts, you will have to create one. Click on Create Group.



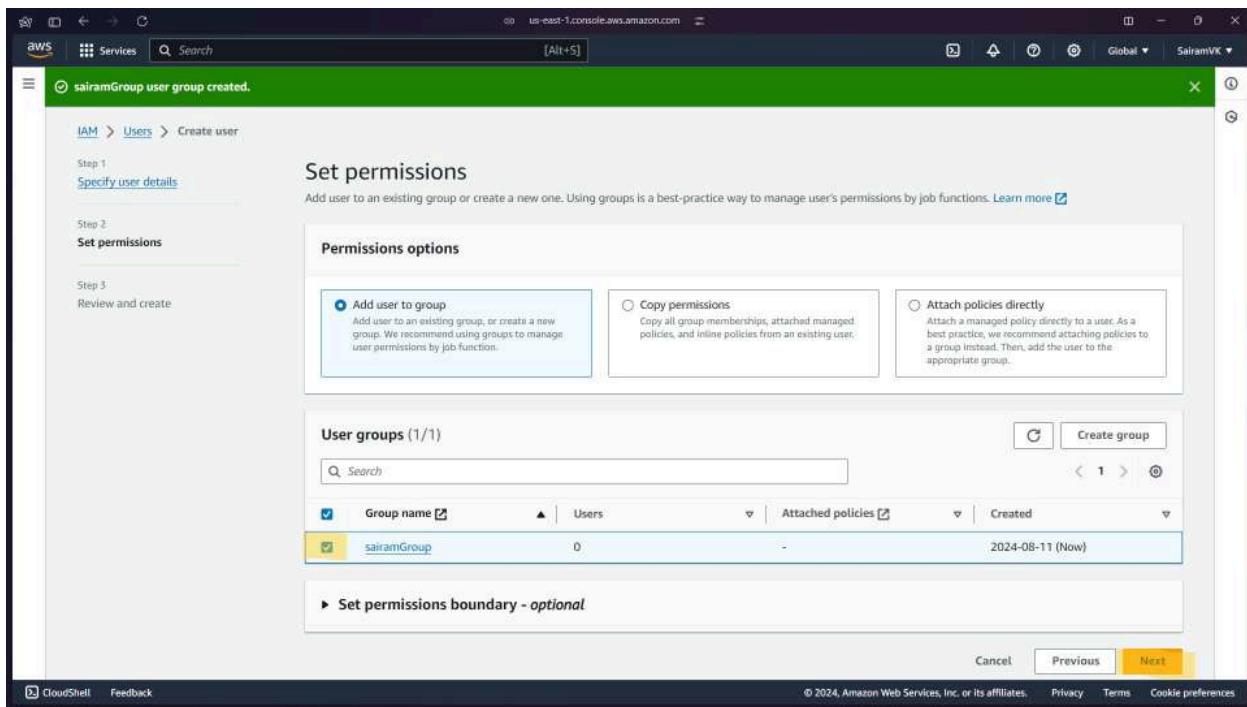
- 4) Give a name to your user group. Then click on Create User Group.



- 5) The group is created and shown under the groups area, select the checkbox. Then click Next.



- 6) Review all the Information, then click on Create user.



7) Open User Groups tab from the left side option. Click on the name of your group.

The screenshot shows the AWS IAM User Groups page. On the left sidebar, under 'Access management', 'User groups' is selected. In the main content area, there is a table titled 'User groups (1)'. The table has columns for 'Group name', 'Users', 'Permissions', and 'Creation time'. A single row is listed for 'sairamGroup', which was created 'Now' and has 'Not defined' permissions. There are buttons for 'Delete' and 'Create group' at the top right of the table.

8) Go to permissions and click on Add permissions. Click on Attach Policies.

The screenshot shows the 'sairamGroup' page under the 'Permissions' tab. At the top right, there is a 'Add permissions' button with a dropdown menu. The 'Attach policies' option is selected. Below this, there is a table for 'Permissions policies (0)' with columns for 'Policy name', 'Type', and 'Attached entities'. A search bar and a 'Filter by Type' dropdown are also present. The message 'No resources to display' is shown at the bottom.

9) Search for AWSCloud9EnvironmentMember, select it and click on Attach policies

Policy name	Type	Used as	Description
AWSCloud9Administrator	AWS managed	None	Provides administrator access to AWS ...
<b>AWSCloud9EnvironmentMember</b>	AWS managed	None	Provides the ability to be invited into ...
AWSCloud9SSMInstanceProfile	AWS managed	Permissions policy (1)	This policy will be used to attach a rol...
AWSCloud9User	AWS managed	None	Provides permission to create AWS Clo...

10) The policies have been attached.

Policy name	Type	Attached entities
<b>AWSCloud9EnvironmentMember</b>	AWS managed	1

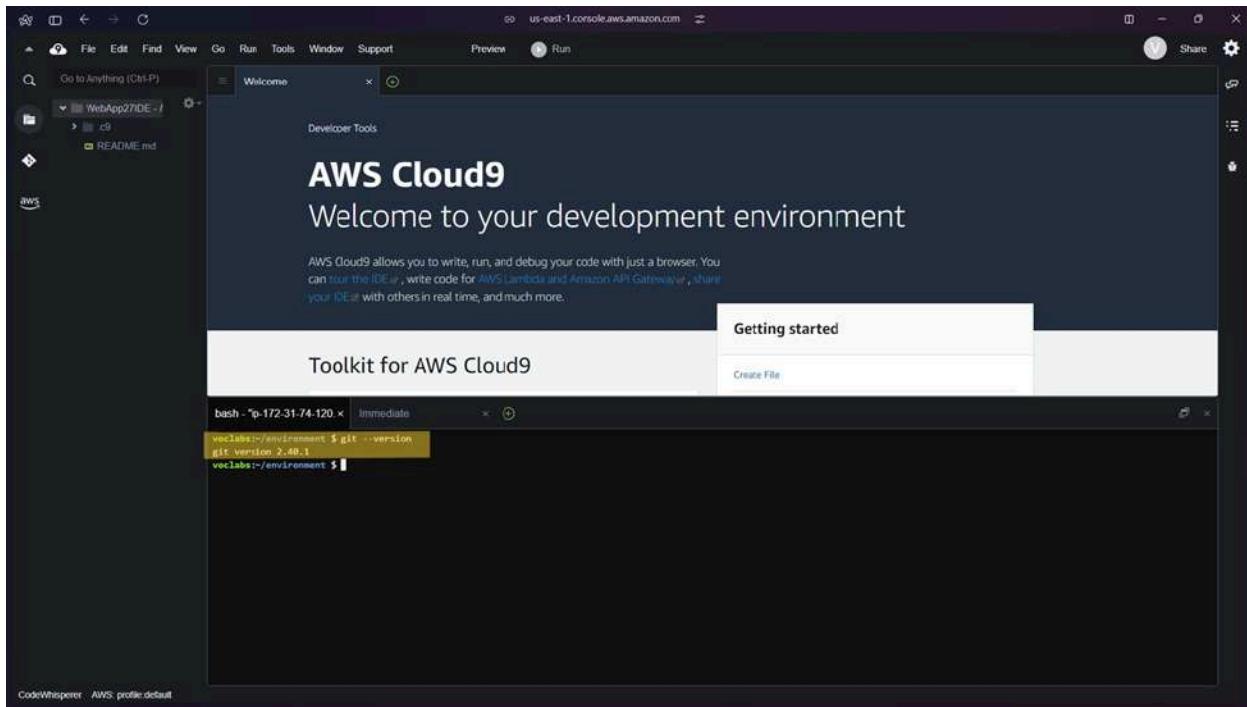
### Step 3: Working on Cloud9 IDE

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

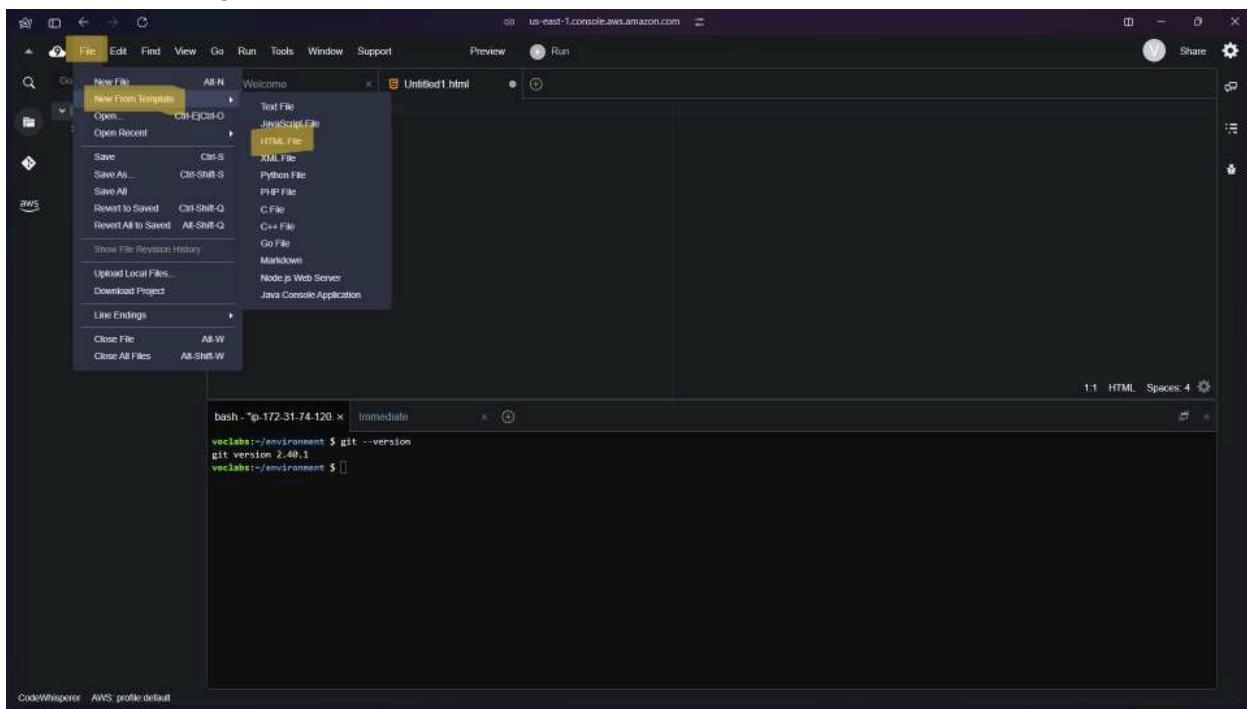
The screenshot shows the AWS Cloud9 Services console. In the top navigation bar, 'Cloud9 IDE' is highlighted. Below it, the 'Environments' section displays a table with one row. The row contains the environment name 'WebApp27IDE', its status 'Cloud9 IDE [Open]', 'Environment type: EC2 Instance', 'Connection: Secure Shell (SSH)', 'Permission: Owner', and 'Owner ARN'. A yellow box highlights the 'Open' button next to the environment name. At the bottom of the page, there are links for 'CloudShell' and 'Feedback'.

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

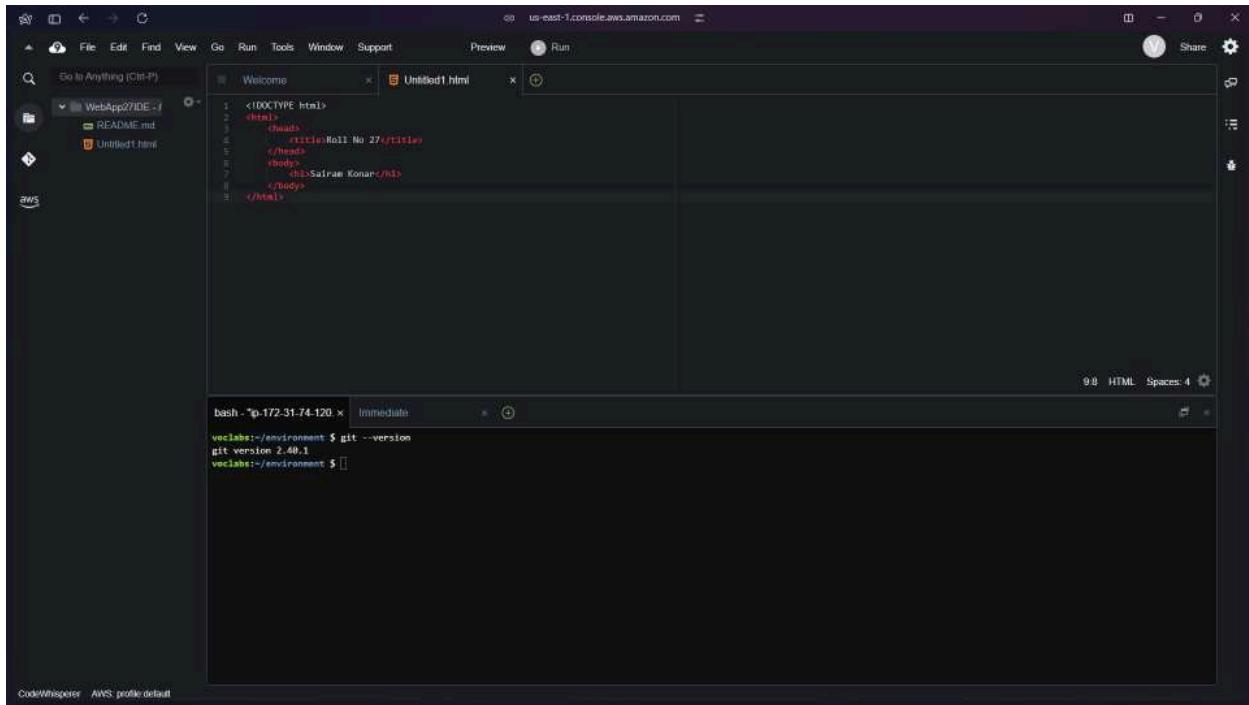
The screenshot shows the AWS Cloud9 IDE interface. The main area is a code editor with a dark theme, displaying a file named 'README.md'. To the left is a sidebar with a file tree showing 'WebApp27IDE - j' and 'c9'. On the right, there are two panels: 'Getting started' with options like 'Create File', 'Upload Files...', and 'Clone from GitHub'; and 'Configure AWS Cloud9' with a 'Main Theme' dropdown set to 'soft-dark'. At the bottom of the screen is a terminal window titled 'bash - \*ip-172-31-74-120 \*' showing the command 'vocabs@environment \$'. The status bar at the bottom left indicates 'CodeWhisperer AWS profile default'.



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



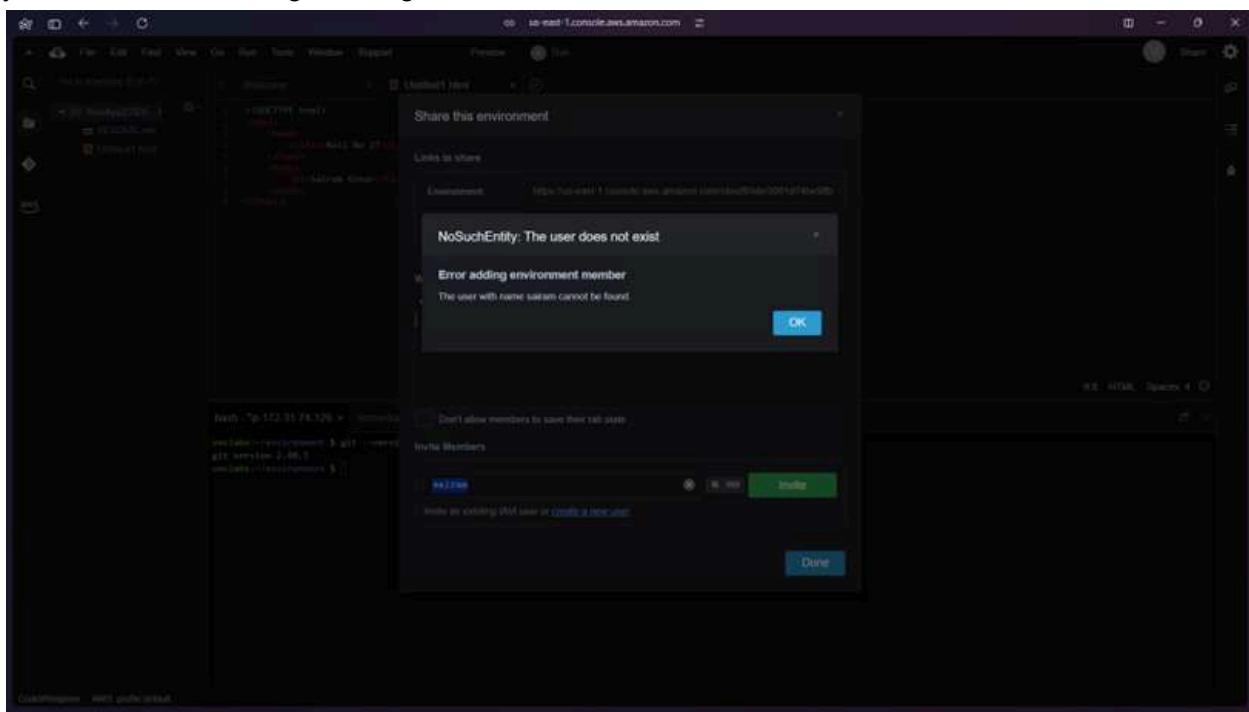
- 4) Make a basic website on the HTML template and save it.



```
<!DOCTYPE html>
<html>
<head>
<title>Roll No 27</title>
</head>
<body>
<h1>Sairam Konar</h1>
</body>
</html>
```

```
bash ->git --version
git version 2.40.1
bash ->
```

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



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

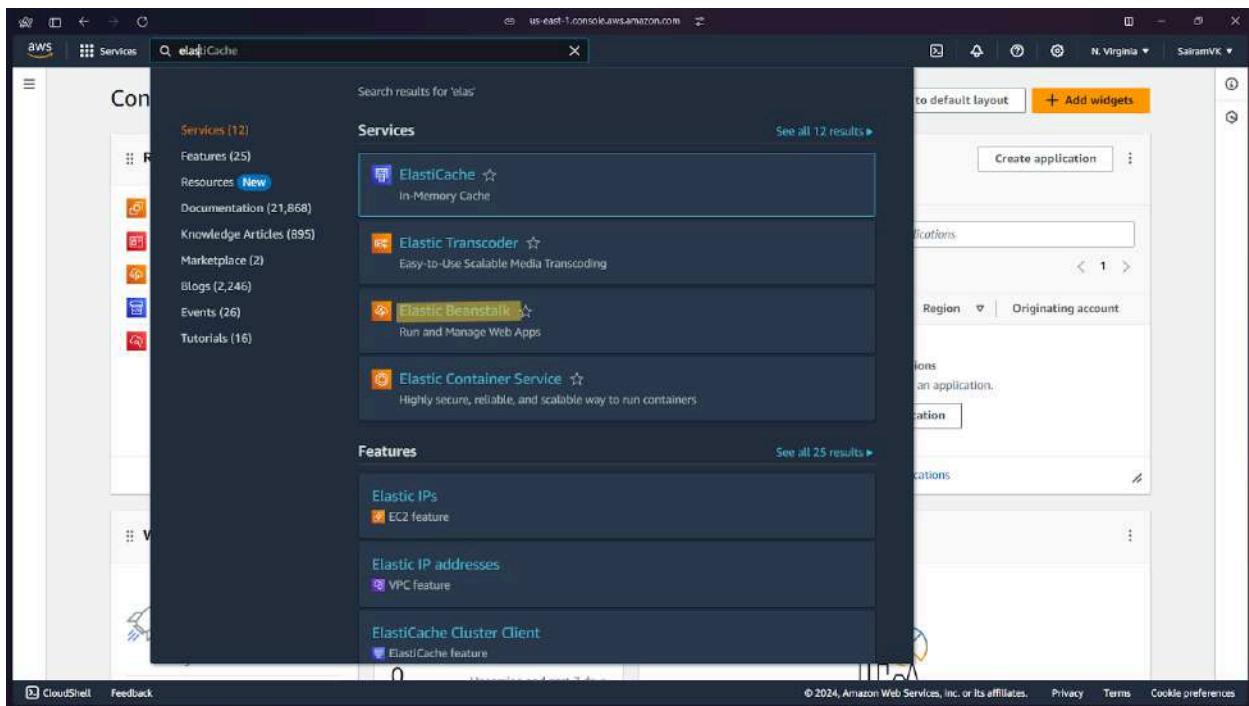
**Conclusion:**

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.

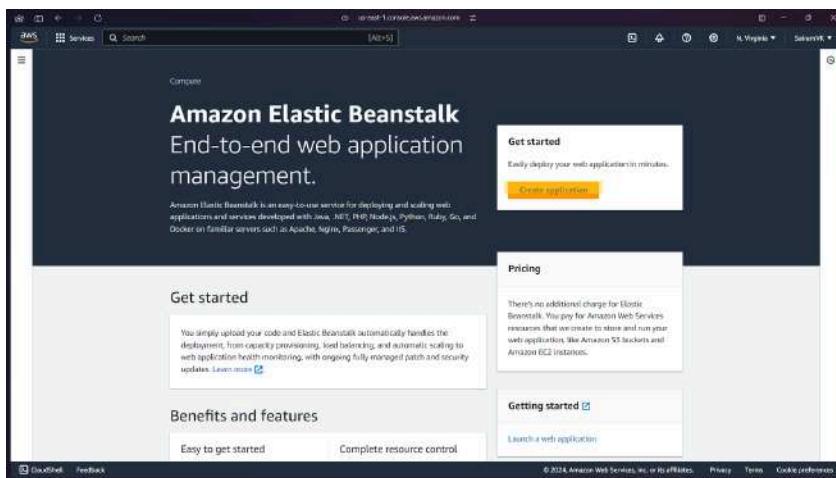
## Experiment 2

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

**Step 1:** Login to your AWS console. Search for Elastic Beanstalk in the searchbar near services.



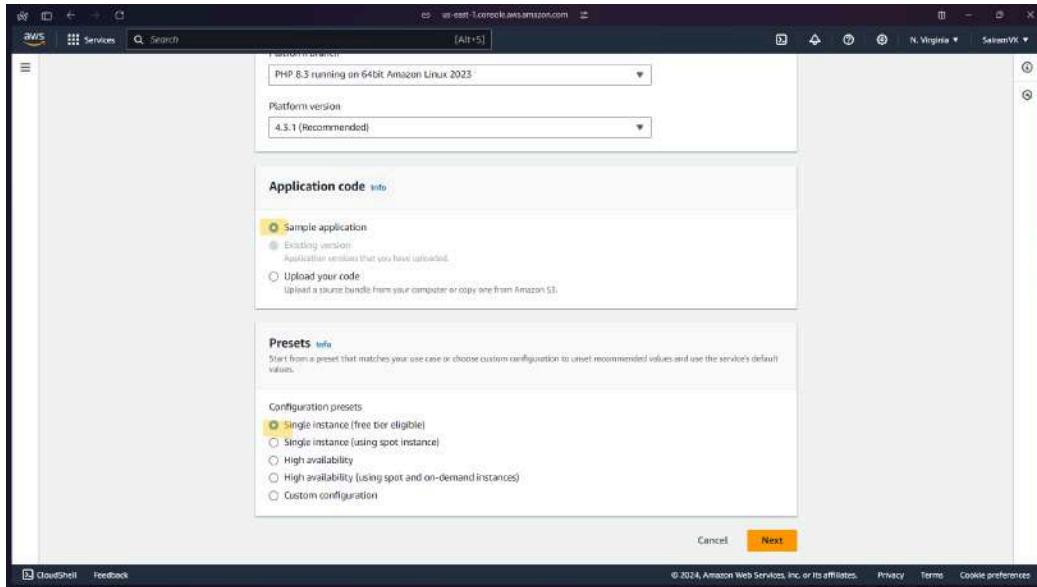
**Step 2:** Go to Elastic Beanstalk and click on Create Application



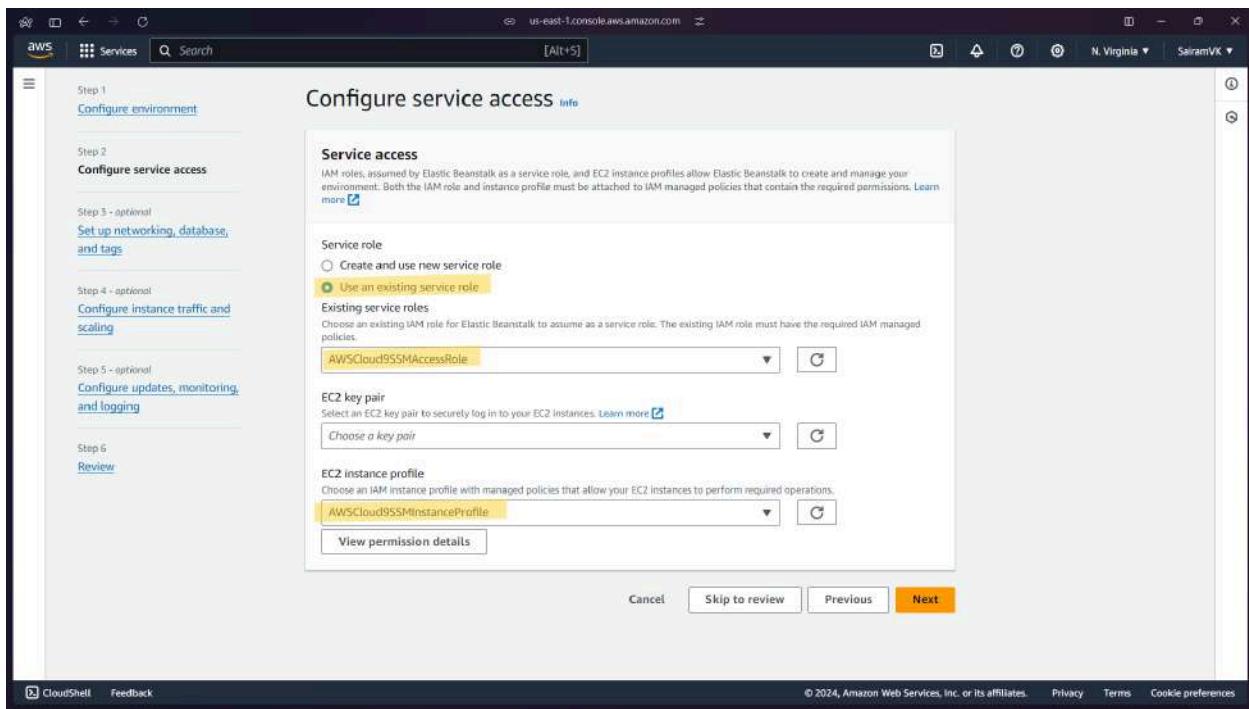
**Step 3:** Enter the name of your application. Scroll down and in the platform, select platform as PHP. Keep the application code as Sample Application. Set the instance to single instance. Click on NEXT.

The screenshot shows the 'Configure environment' step of the AWS Elastic Beanstalk setup wizard. The application name is 'MyWebApp27'. The environment tier is set to 'Web server environment'. The environment name is 'MyWebApp27-env'.

The screenshot shows the 'Platform' configuration step. The platform type is 'Managed platform' (PHP). The platform version is '4.3.1 (Recommended)'.



**Step 4:** Use an existing service role and choose whatever service role is available on your account.



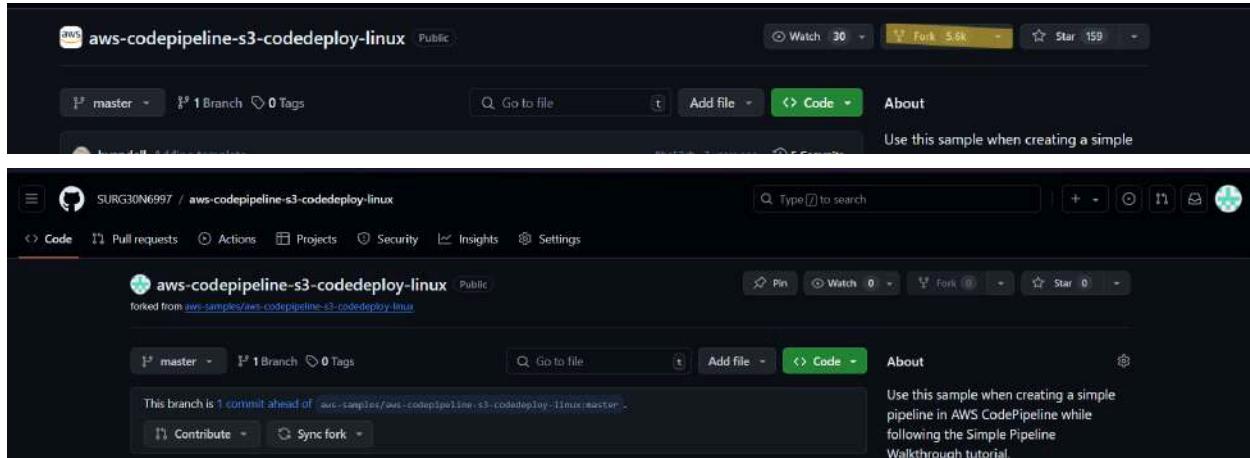
**Step 5:** Review the settings that you have set up for your application and submit your application.

The screenshot shows the 'Review' step of the configuration wizard. It displays the environment information and service access configurations. The environment tier is set to 'Web server environment' with the application name 'MyWebApp27'. The environment name is 'MyWebApp27-env' and the application code is 'Sample application'. The platform is specified as 'arn:aws:elasticbeanstalk:us-east-1::platform/PHP 8.3' running on '64bit Amazon Linux 2023/4.3.1'. The service role is 'arn:aws:iam::011528263337:role/service-role/AWSCloud9SSMAccessRole' and the EC2 instance profile is 'AWSCloud9SSMInstanceProfile'. The sidebar on the left lists optional steps: networking, database, and tags; instance traffic and scaling; and updates, monitoring, and logging.

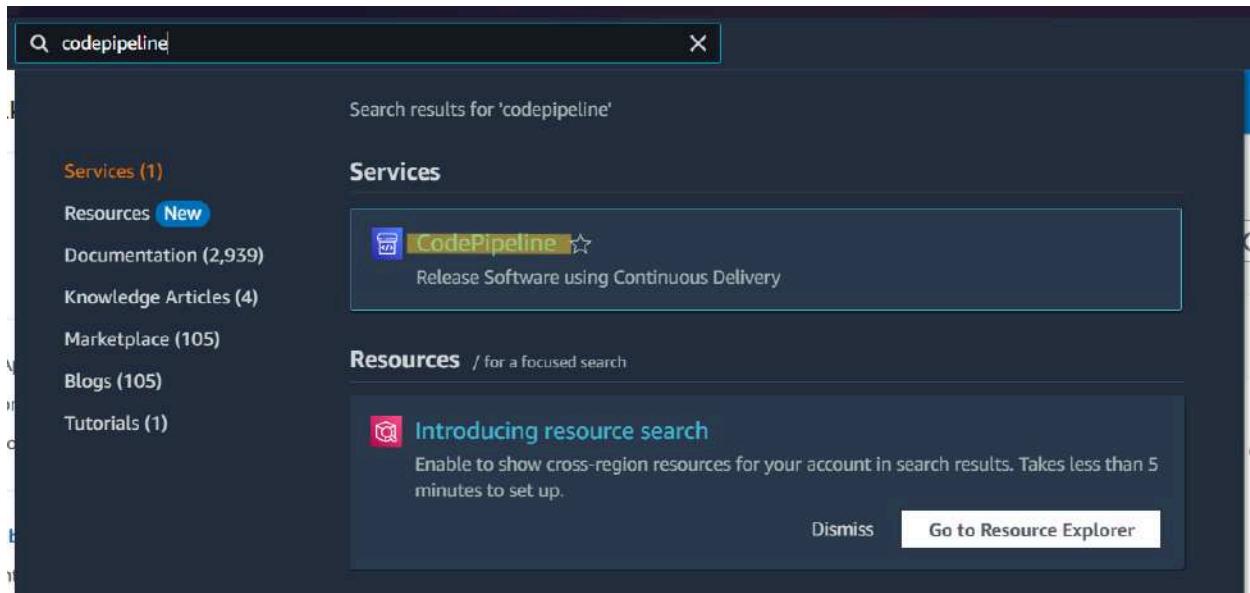
The screenshot shows the 'Review' step of the configuration wizard. It displays the service access configurations. The service role is 'arn:aws:iam::011528263337:role/service-role/AWSCloud9SSMAccessRole' and the EC2 instance profile is 'AWSCloud9SSMInstanceProfile'. The sidebar on the left lists optional steps: networking, database, and tags; instance traffic and scaling; and updates, monitoring, and logging.

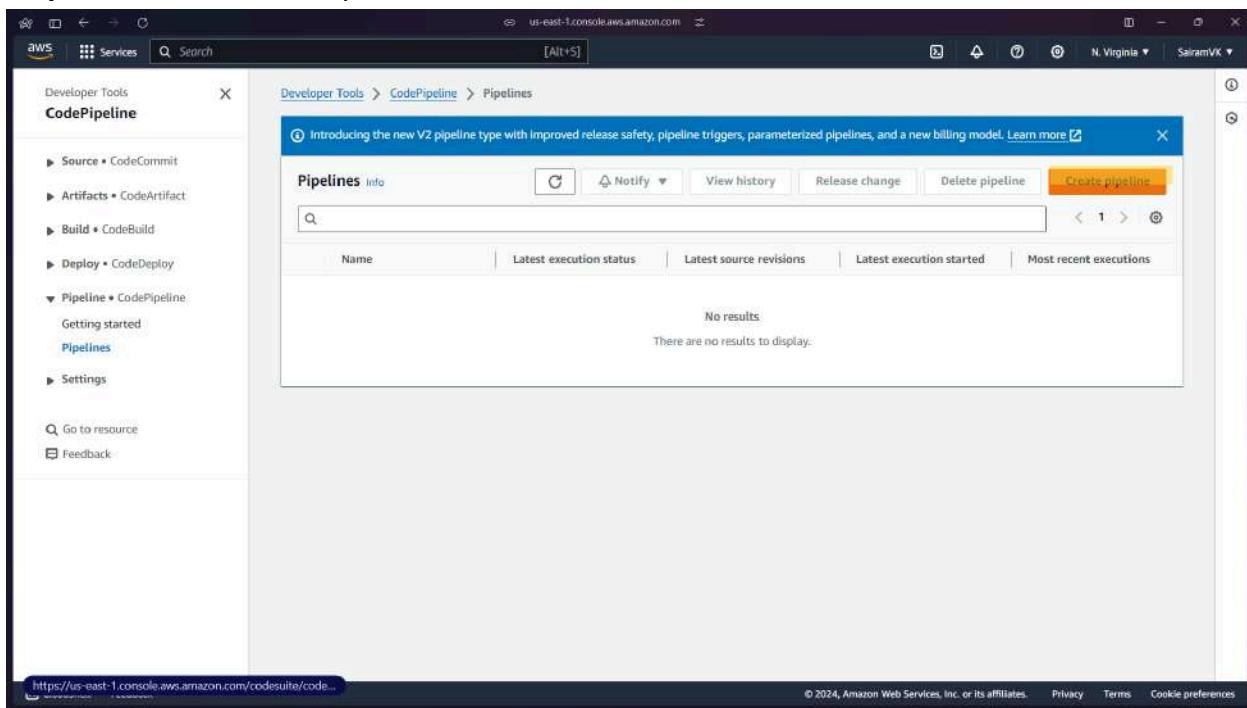
**Step 6:** Go to the github link below. This is a github with a sample code for deploying a file on AWS CodePipeline. Fork this repository into your personal github.

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

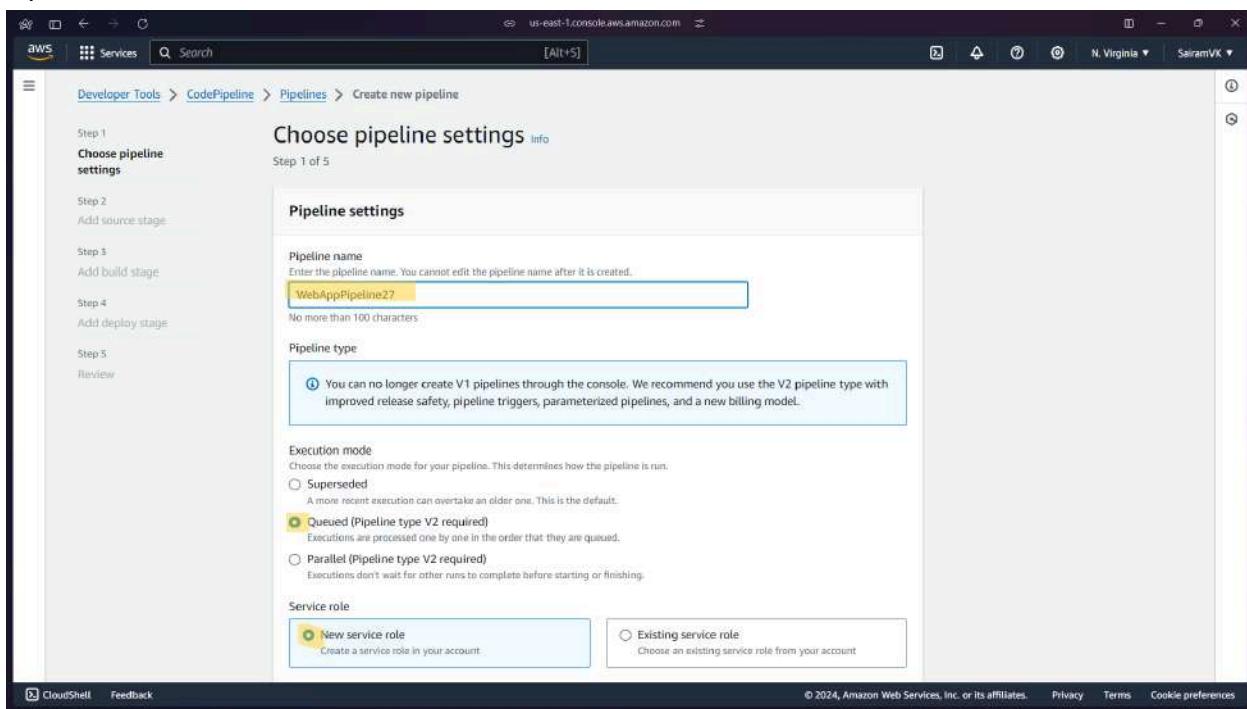


**Step 7:** Search CodePipeline in the services tab and click on it.

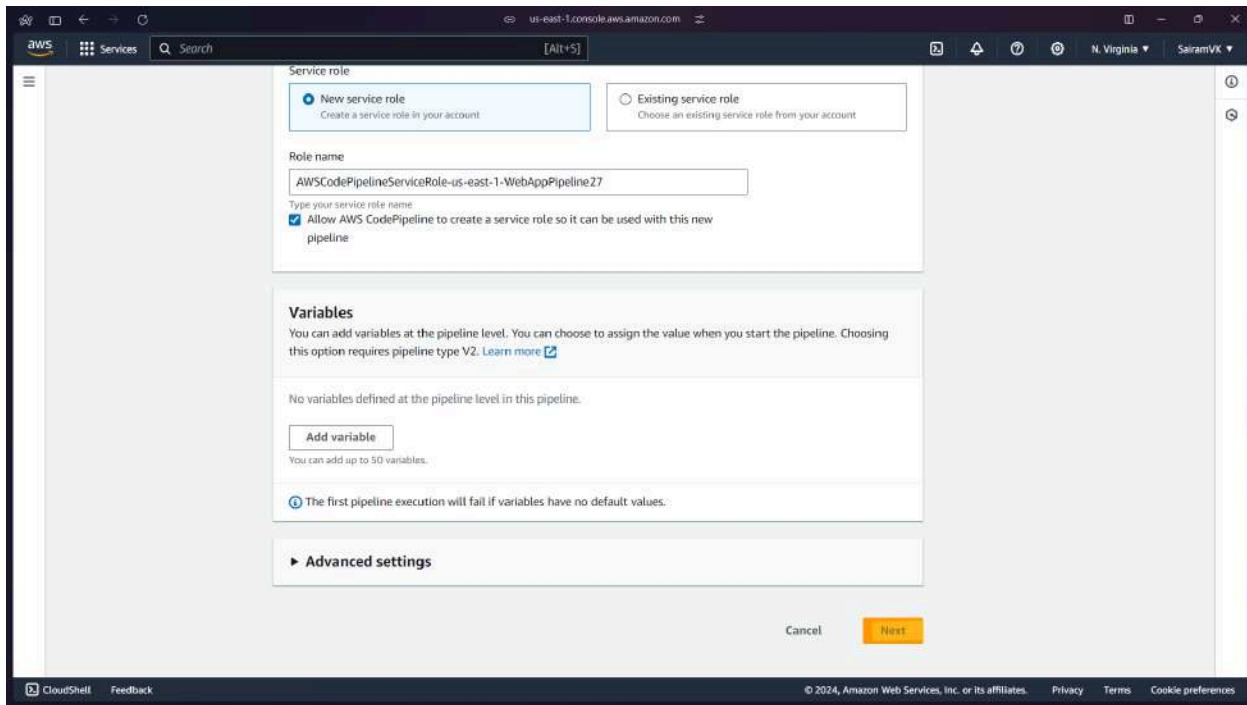


**Step 8:** Click on Create Pipeline.

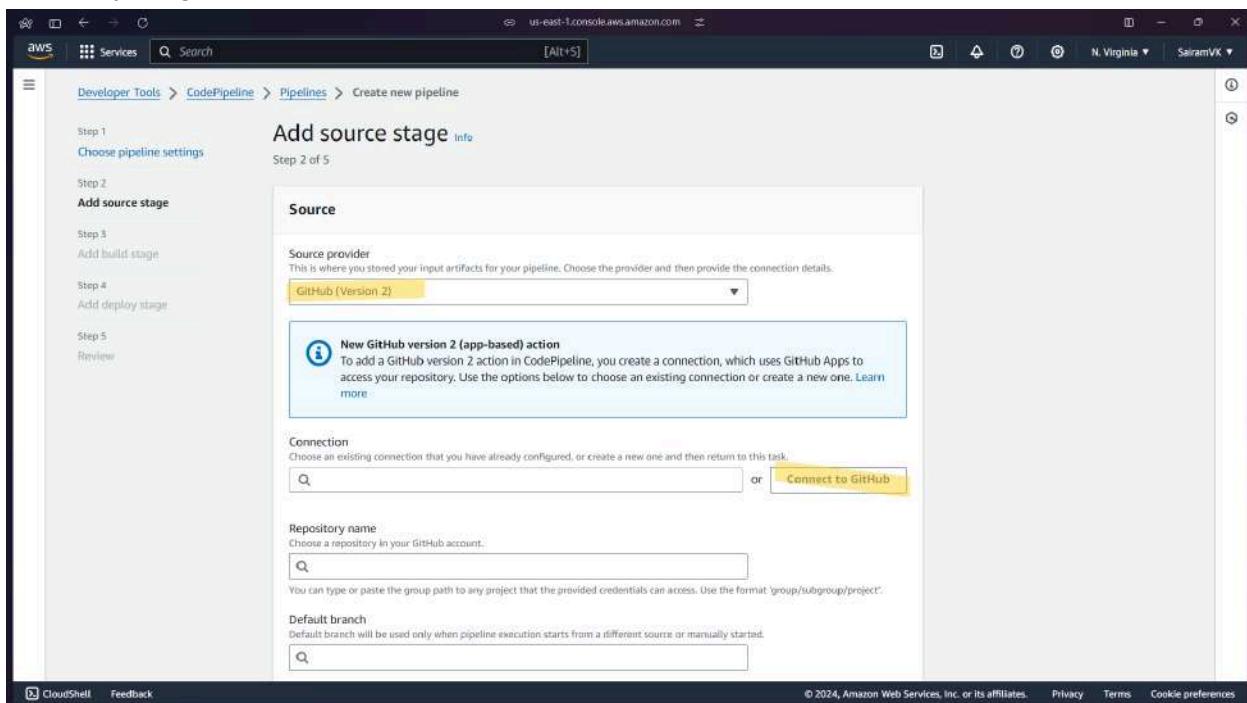
The screenshot shows the AWS CodePipeline Pipelines page. On the left, there's a sidebar with 'CodePipeline' selected under 'Developer Tools'. The main area has a heading 'Introducing the new V2 pipeline type with improved release safety, pipeline triggers, parameterized pipelines, and a new billing model.' Below it is a table with columns: Name, Latest execution status, Latest source revisions, Latest execution started, and Most recent executions. A message 'No results' and 'There are no results to display.' is centered. At the top right of the main area is a blue button labeled 'Create pipeline'.

**Step 9:** Give a name to your Pipeline. A new service role would be created with the name of the Pipeline.

The screenshot shows the 'Choose pipeline settings' step of the pipeline creation wizard. On the left, a sidebar lists steps: Step 1 (selected), Step 2, Step 3, Step 4, Step 5, and Review. The main area has a title 'Choose pipeline settings' and a sub-section 'Pipeline settings'. It shows a 'Pipeline name' field containing 'WebAppPipeline27' and a note 'Enter the pipeline name. You cannot edit the pipeline name after it is created. No more than 100 characters'. Below this is a 'Pipeline type' section with a note: 'You can no longer create V1 pipelines through the console. We recommend you use the V2 pipeline type with improved release safety, pipeline triggers, parameterized pipelines, and a new billing model.' Under 'Execution mode', 'Queued (Pipeline type V2 required)' is selected. Under 'Service role', 'New service role' is selected. At the bottom are buttons for 'CloudShell', 'Feedback', 'Privacy', 'Terms', and 'Cookie preferences'.



**Step 10:** Select a source provider (as Github (Version 2)). Click on Connect to Github to connect your github.



**Step 11:** Give a name to your GitHub app Connection and click on Connect. This will give you a prompt to either to select a GitHub app or install a new app. If this is your first time, click on Install a new app.

The image contains two screenshots of the AWS Developer Tools Connections interface, illustrating the process of creating a GitHub connection.

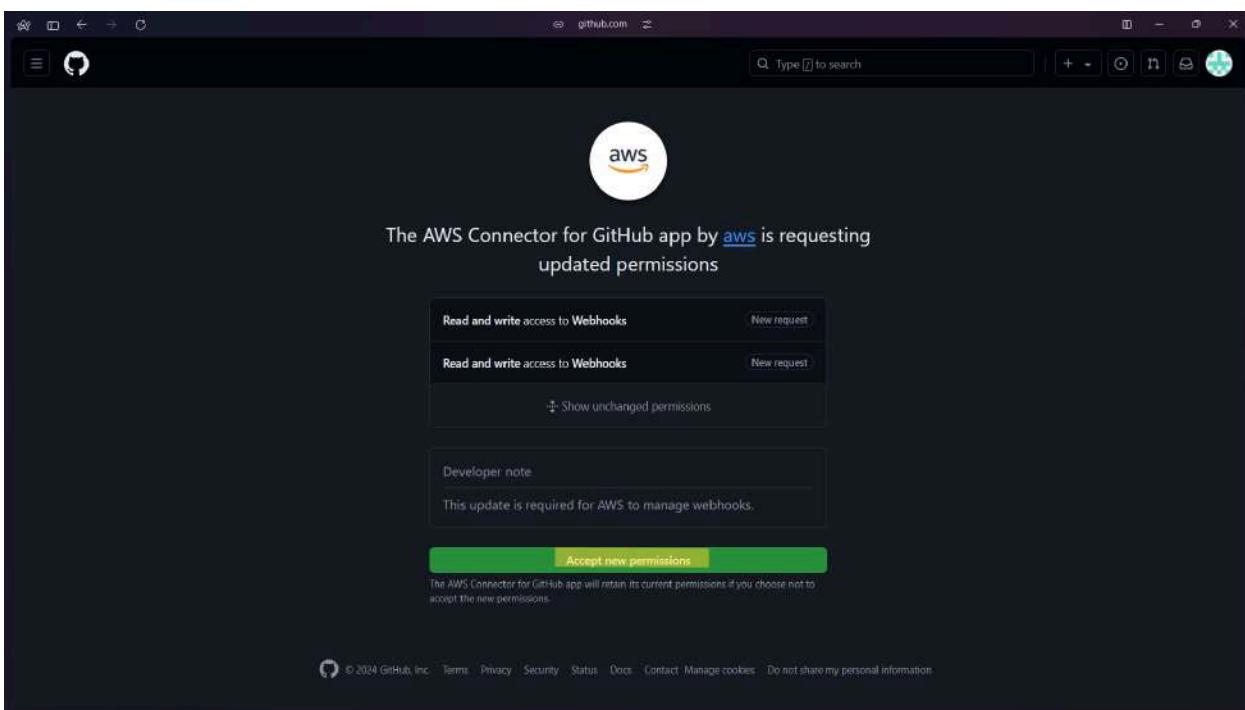
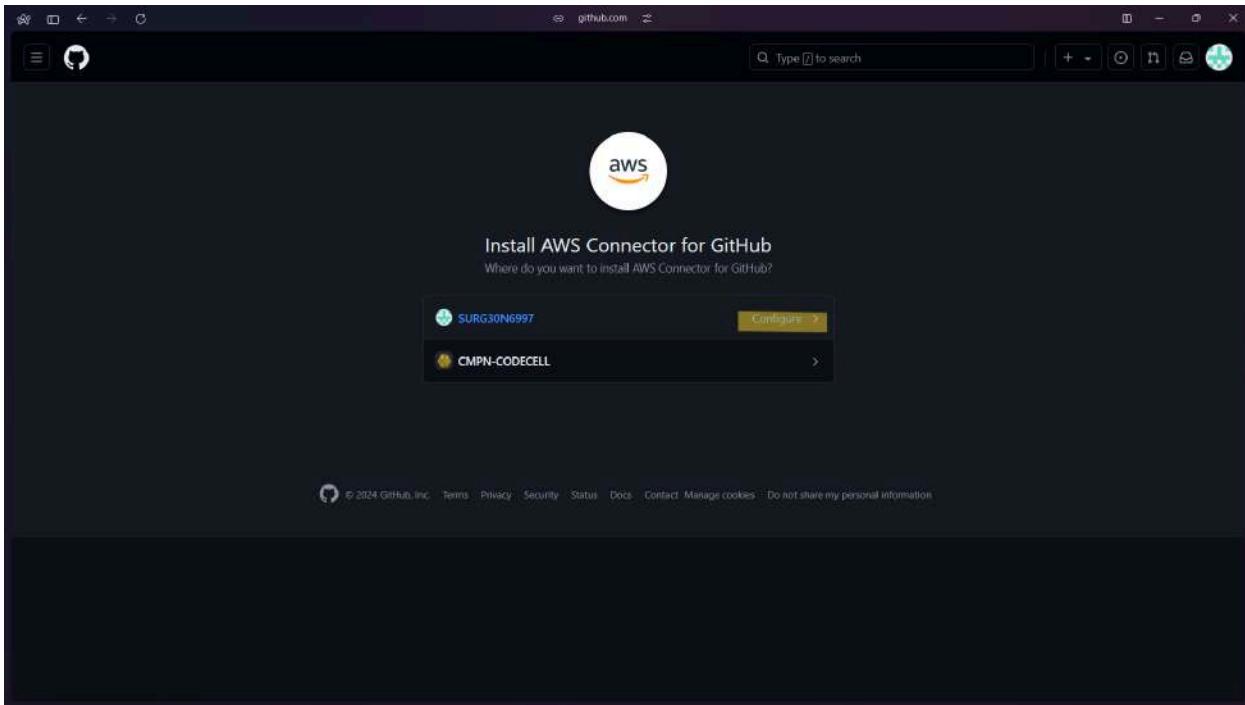
**Screenshot 1: Create a connection - Step 1**

This screenshot shows the initial step of creating a GitHub connection. The "Connection name" field is filled with "MyGithub". The "Connect to GitHub" button is visible at the bottom right.

**Screenshot 2: Create a connection - Step 2**

This screenshot shows the continuation of the connection setup. A blue banner at the top provides information about the transition to ARN-based connections. The "GitHub connection settings" section shows the "Connection name" as "MyGithub". The "GitHub Apps" section includes a search bar and a "Install a new app" button. The "Connect" button is located at the bottom right.

**Step 12:** This will direct you to install AWS Connector on your GitHub. Install it to your account and give it its permissions.



**Step 13:** After the app is set up, it gives the number in the text field. Click on Connect. After clicking on connect, the link is shown in the connection field and AWS shows that GitHub connection is ready to use.

The image contains two screenshots of the AWS CodePipeline console interface, illustrating the steps to set up a GitHub connection.

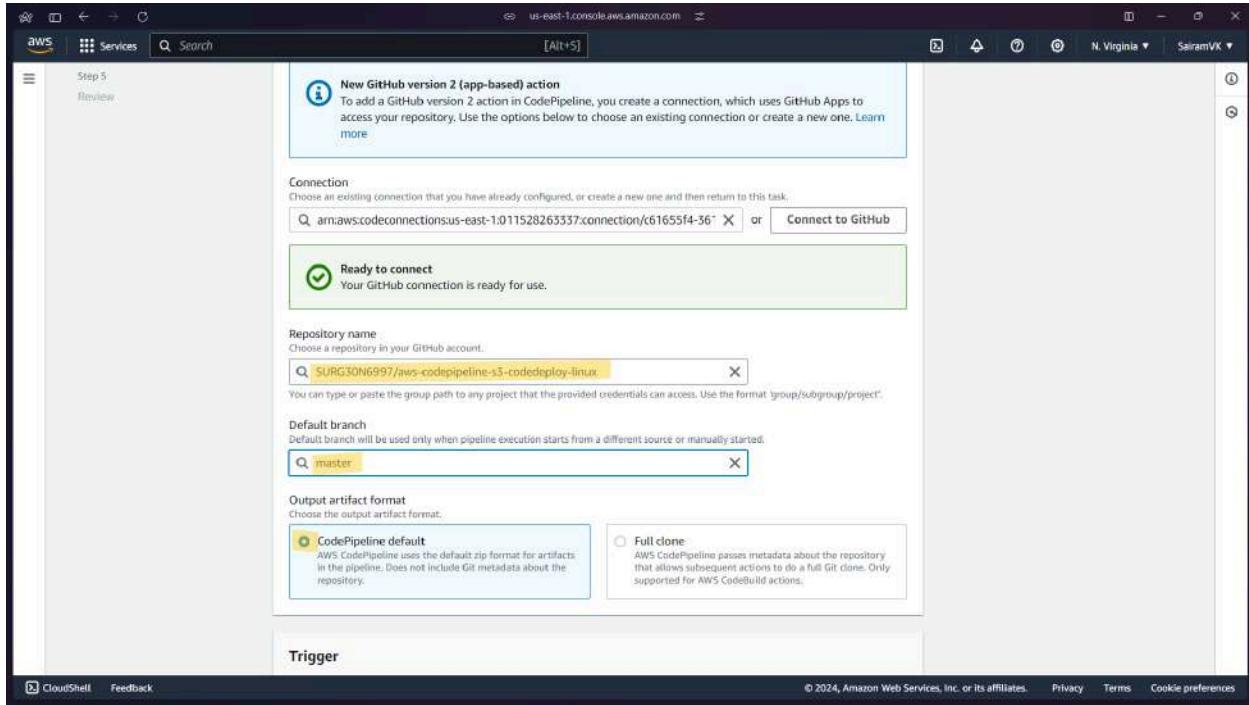
**Screenshot 1: Create Connection - Step 1**

This screenshot shows the "Create connection" step in the AWS Lambda developer tools. The "GitHub connection settings" section is visible, with a "Connection name" input field containing "MyGithub". Below it, a "GitHub Apps" section shows a search bar with "53469209" and a "Install a new app" button. A "Tags - optional" section is present, and a prominent orange "Connect" button is at the bottom right.

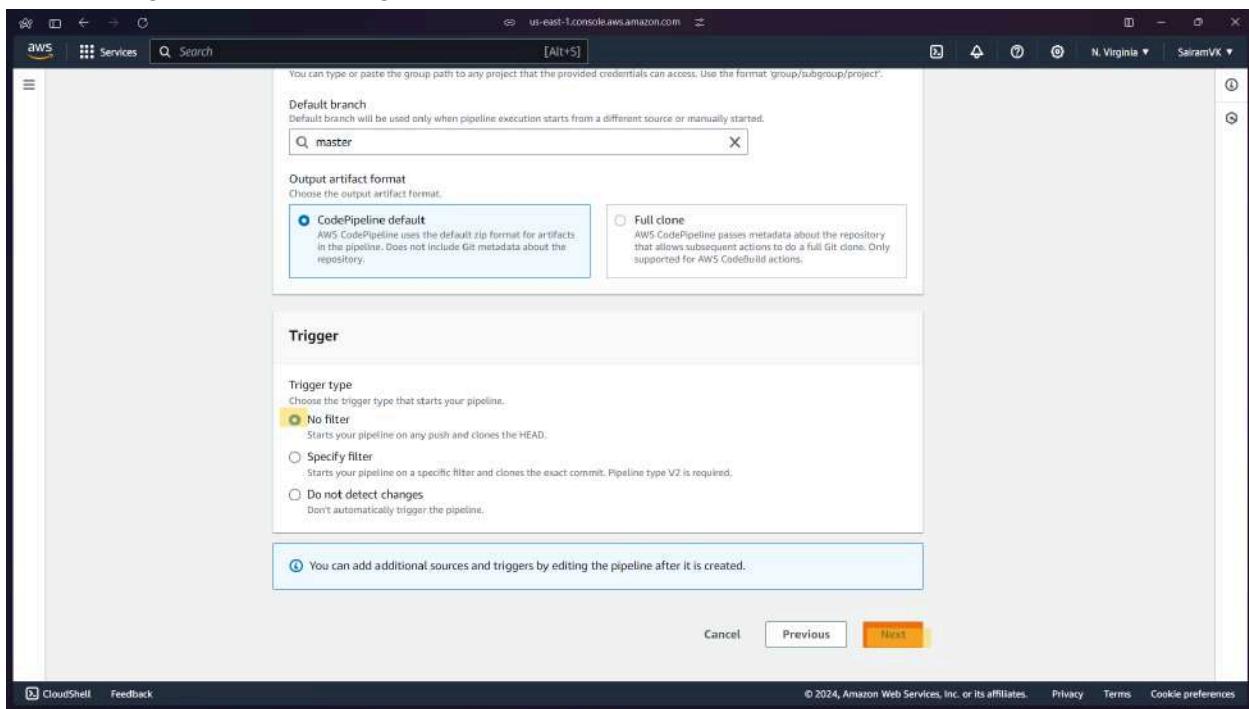
**Screenshot 2: Create Pipeline - Step 4**

This screenshot shows the "Add source stage" step in the AWS CodePipeline pipeline creation wizard. The "Source" provider dropdown is set to "GitHub (Version 2)". A callout box highlights the "New GitHub version 2 (app-based) action" information. The "Connection" section shows a search bar with "arn:aws:codeconnections:us-east-1:011528263337:connection/c61655f4-3d5c-45e0-833a-1234567890ab" and a "Connect to GitHub" button. A green success message states "Ready to connect" with a checkmark icon. Below are fields for "Repository name" (with a placeholder "Choose a repository in your GitHub account."), "Default branch" (with a placeholder "Default branch will be used only when pipeline execution starts from a different source or manually started."), and "Output artifact format" (with options "CodePipeline default" and "Full clone").

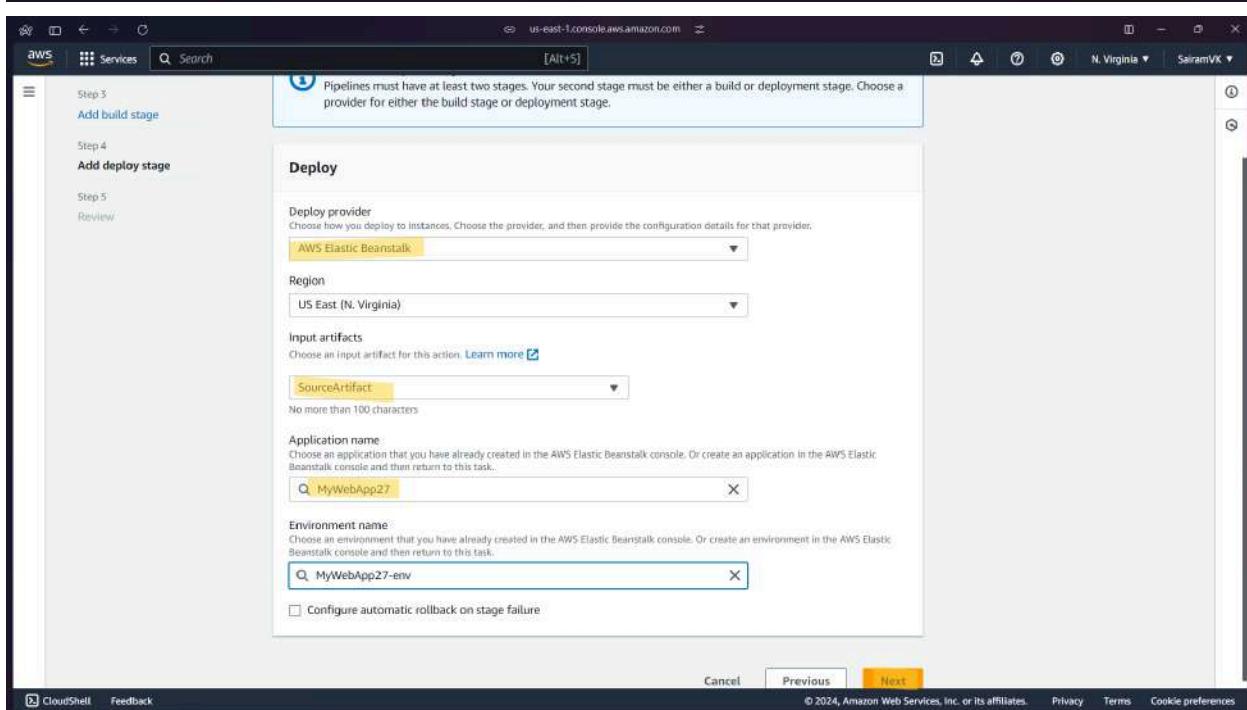
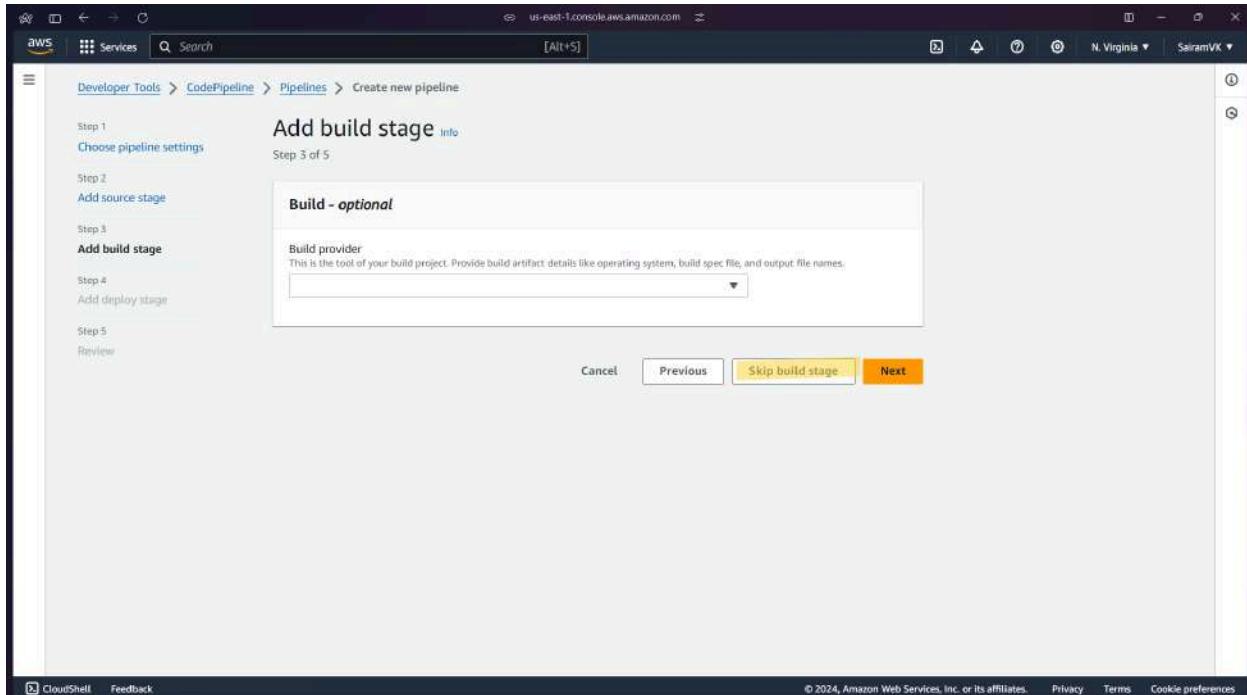
**Step 14:** Select the repository that you had forked to your GitHub. After that select the branch on which the files are present (default is Master).



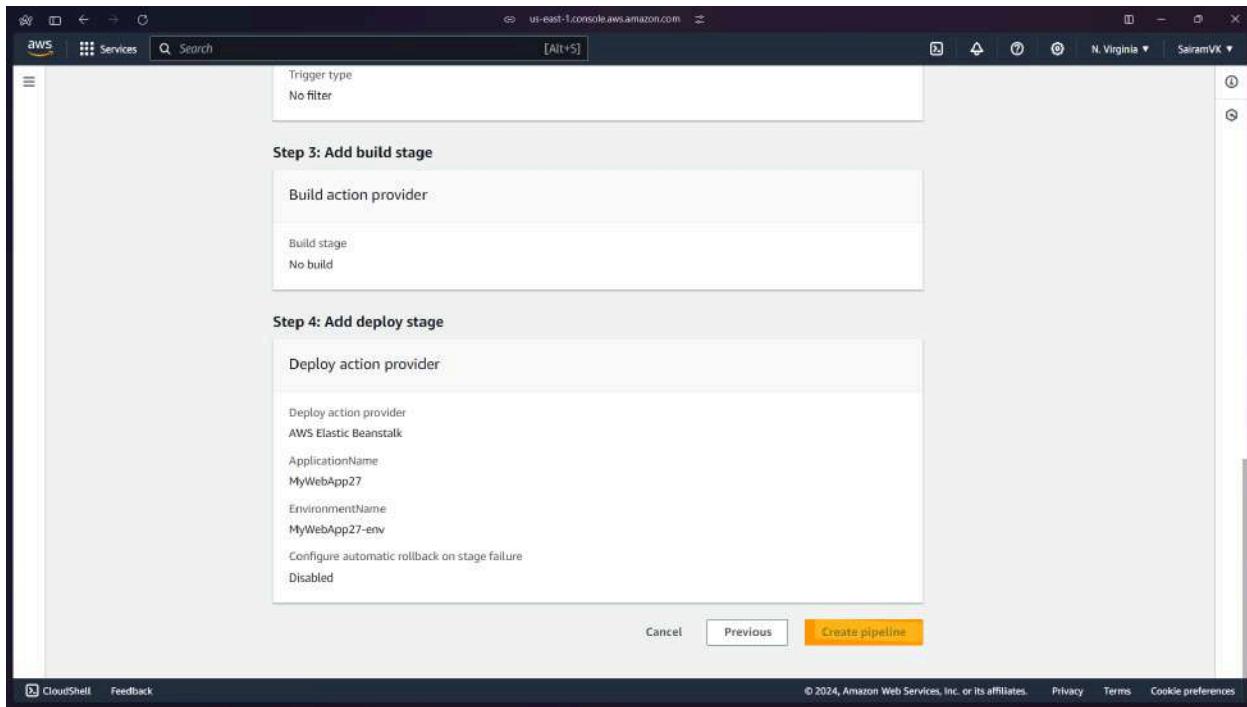
**Step 15:** Set the Trigger type as no filter. This would allow it to the website to update as soon as some change is made in the github.



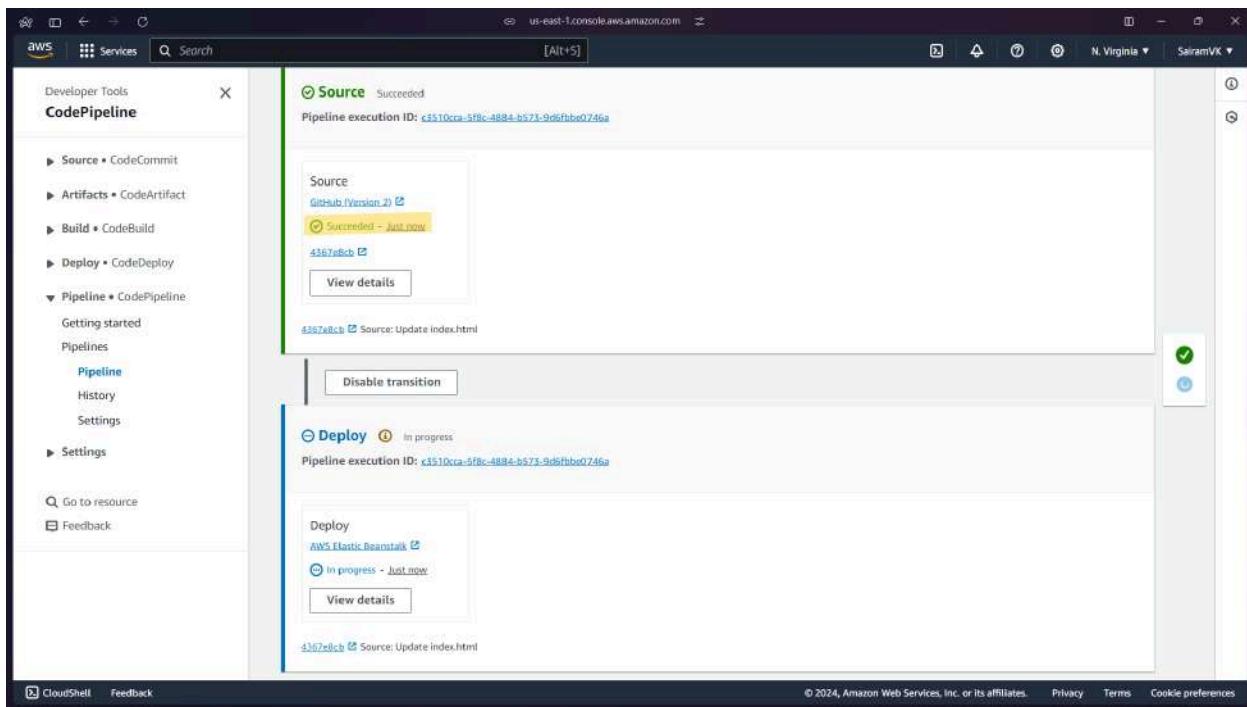
**Step 16:** Skip the build stage and go to Deploy. Select the deploy provider as AWS Elastic Beanstalk and Input Artifact as SourceArtifact. The application name would be the name of your Elastic Beanstalk. Then click on next.



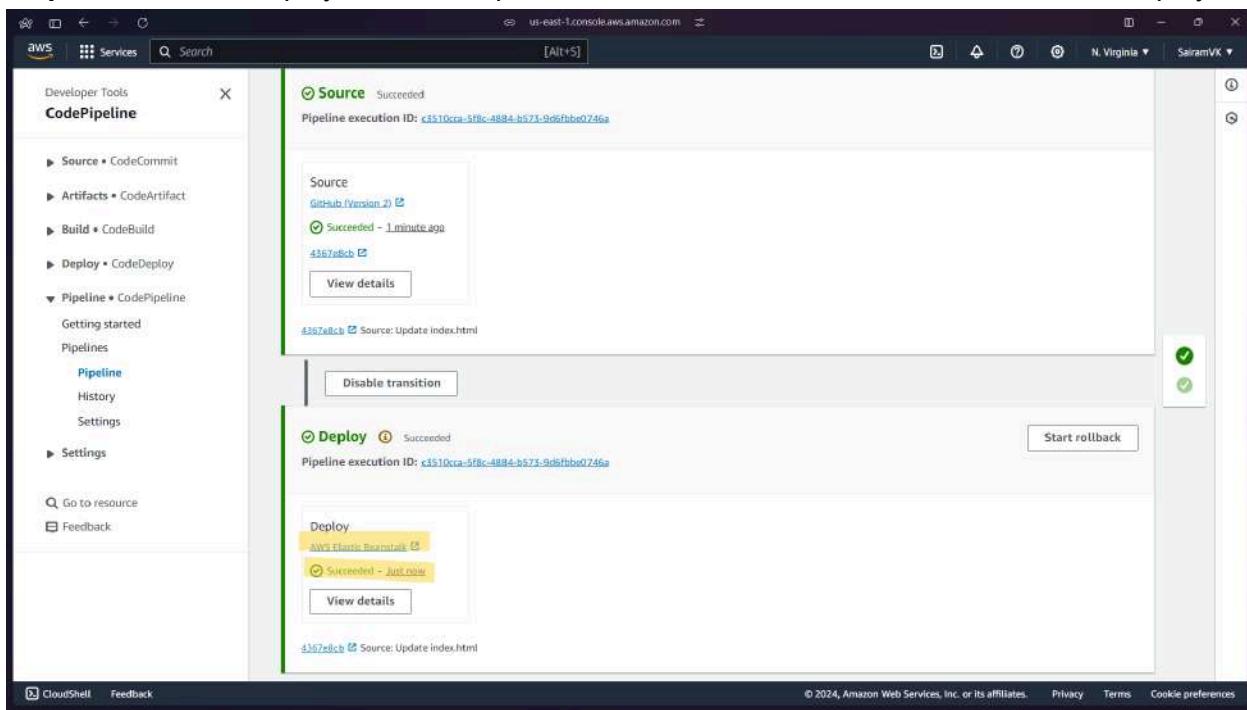
**Step 17:** Check all the information and click on create Pipeline.



**Step 18:** If the pipeline is successfully deployed, this screen comes up where the source is set up and then it is transitioned to deploy.

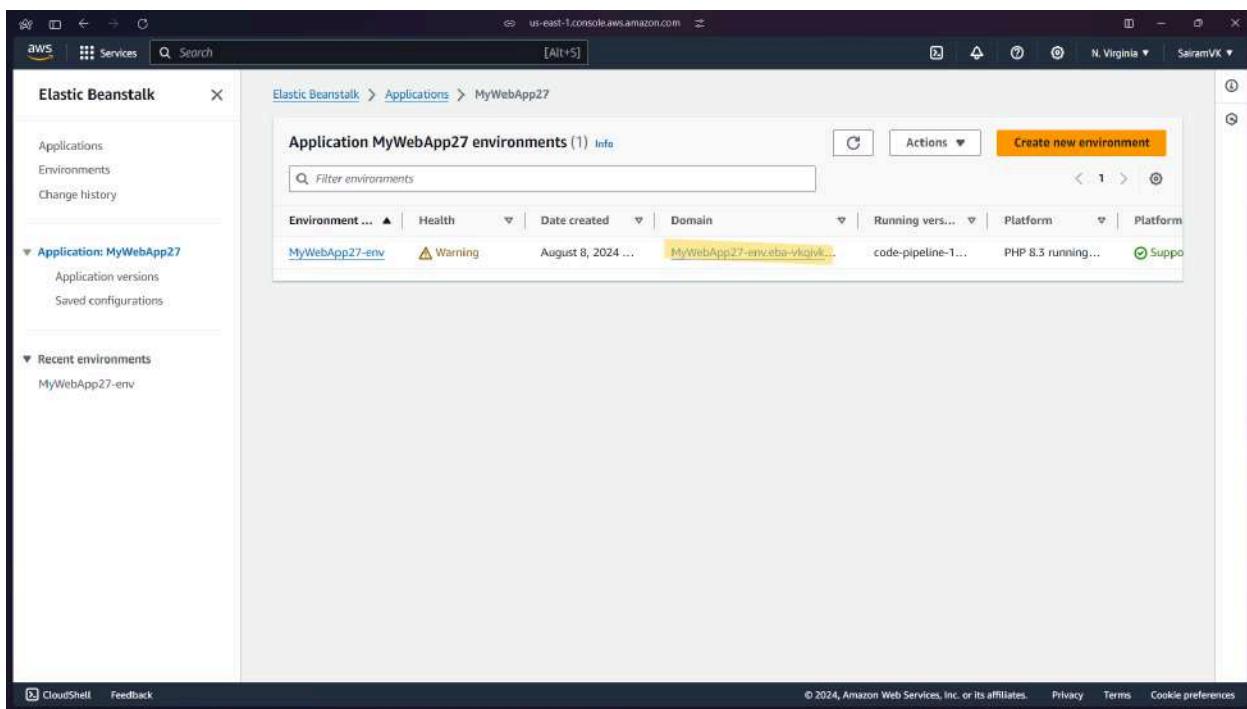


**Step 19:** Once the deployment is complete, click on the AWS Elastic Beanstalk under Deploy.



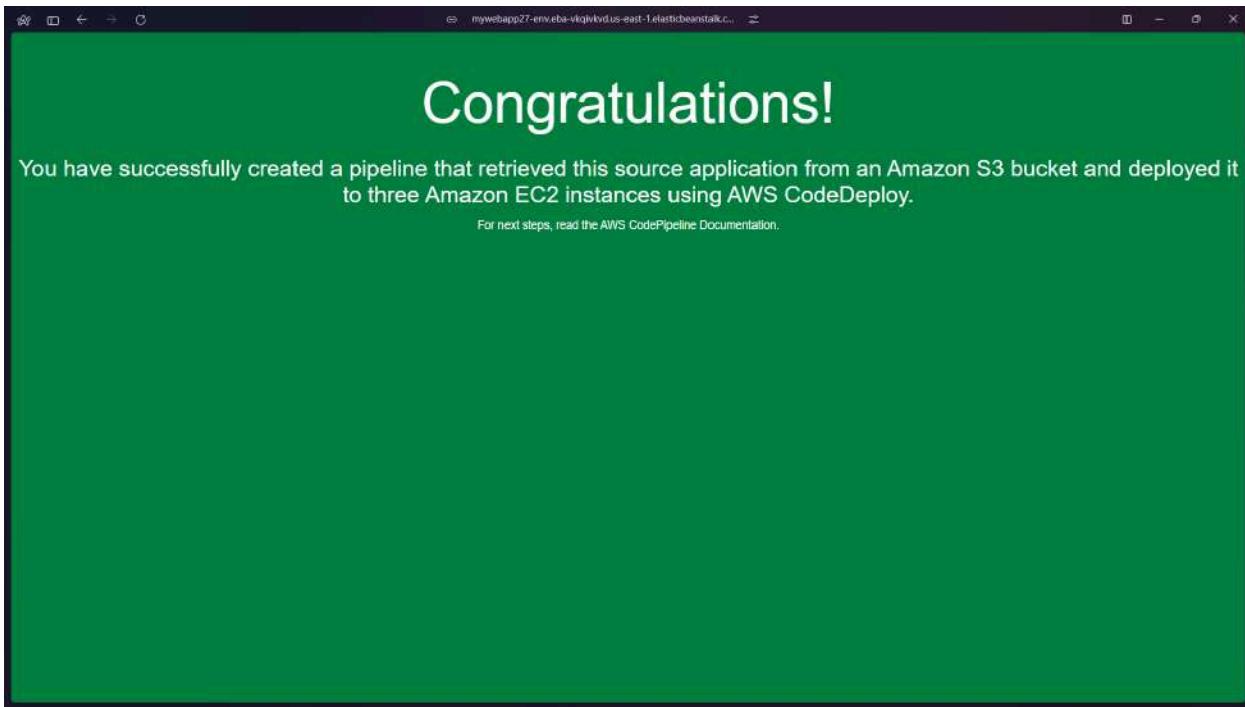
The screenshot shows the AWS CodePipeline console. On the left, there's a sidebar with 'Developer Tools' and 'CodePipeline' selected. Under 'CodePipeline', there are sections for Source, Artifacts, Build, Deploy, Pipeline, History, Settings, and Settings. Below that are links for 'Go to resource' and 'Feedback'. The main area has two tabs: 'Source' (Succeeded) and 'Deploy' (Succeeded). The 'Source' tab shows a GitHub source provider with a status of 'Succeeded - 1 minute ago'. The 'Deploy' tab shows an 'AWS Elastic Beanstalk' provider with a status of 'Succeeded - just now'. There are 'View details' buttons for both. A 'Start rollback' button is also visible in the Deploy section. The URL in the browser is 'us-east-1.console.aws.amazon.com'.

**Step 20:** This will redirect you to the application screen of Elastic Beanstalk. Click on the link shown under Domain.



The screenshot shows the AWS Elastic Beanstalk Applications screen. The left sidebar has 'Elastic Beanstalk' selected, with options for Applications, Environments, Change history, Application: MyWebApp27 (with sub-options for Application versions and Saved configurations), and Recent environments (MyWebApp27-env). The main area shows a table for 'Application MyWebApp27 environments (1) info'. It lists one environment: 'MyWebApp27-env' with a warning icon, created on August 8, 2024, running version 'MyWebApp27-env.eba-vkqjvkh...', platform 'code-pipeline-1...', and PHP 8.3. There are 'Actions' and 'Create new environment' buttons at the top right. The URL in the browser is 'us-east-1.console.aws.amazon.com'.

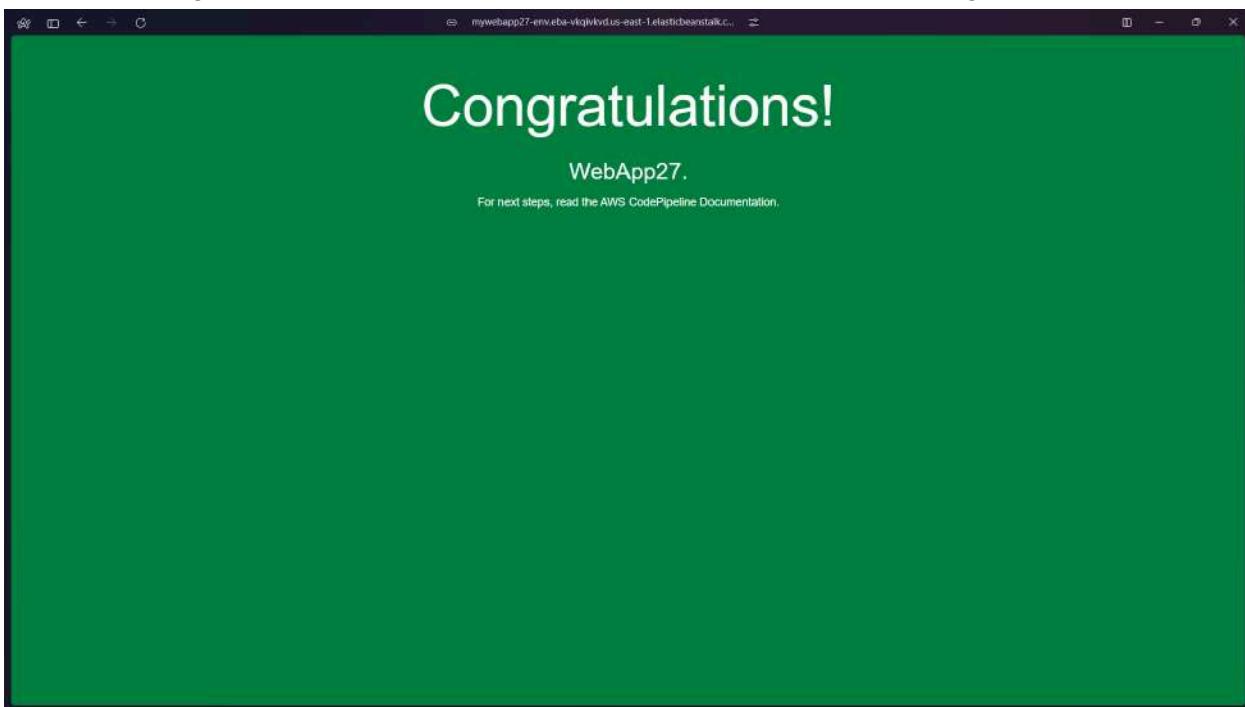
**Step 21:** This will successfully show the sample website hosted.



**Step 22:** Now, we make some changes to the index.html file in the github.

For eg: If you make some changes to the <h2> tag.

Once the changes are committed, when the website is refreshed, the changes can be seen.



**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.

### Prerequisites:

Create 2 security groups by navigating to Search → Security Groups.

The screenshot shows the AWS Cloud9 IDE interface. A search bar at the top contains the text 'security'. Below it, the search results for 'Security' are displayed under the 'Services' section. The results include 'Security Lake', 'Security Hub', 'Detective', and 'Amazon Inspector'. Under the 'Features' section, there are two entries: 'Security groups' (EC2 Feature) and 'Security groups' (VPC feature). The 'Security groups' (EC2 Feature) entry is highlighted with a blue border. On the right side of the interface, there is a sidebar with various AWS services like EC2 Dashboard, Events, CloudWatch Logs, Instances, Images, and Elastic Block Store. At the bottom, there are links for 'Inbound rules' and 'Outbound rules'.

### Group 1: Master

This screenshot shows the 'Inbound rules' configuration for a security group named 'Group 1: Master'. The table lists nine rules:

Type	Protocol	Port range	Source	Description - optional	Action
HTTP	TCP	80	Anywhere-1...		Delete
All traffic	All	All	Anywhere-1...		Delete
Custom TCP	TCP	6443	Anywhere-1...		Delete
Custom TCP	TCP	10251	Anywhere-1...		Delete
Custom TCP	TCP	10250	Anywhere-1...		Delete
All TCP	TCP	0-65535	Anywhere-1...		Delete
Custom TCP	TCP	10252	Anywhere-1...		Delete
SSH	TCP	22	Anywhere-1...		Delete

At the bottom left, there is a button labeled 'Add rule'.

## Group 2: Nodes

The screenshot shows the 'Inbound rules' section of the AWS CloudWatch Metrics interface. It lists several rules:

- All traffic: Protocol: All, Port range: All, Source: Anywhere(-), Description: optional.
- SSH: Protocol: TCP, Port range: 22, Source: Custom, Description: optional.
- Custom TCP: Protocol: TCP, Port range: 10250, Source: Anywhere(-), Description: optional.
- All TCP: Protocol: TCP, Port range: 0 - 65535, Source: Anywhere(-), Description: optional.
- Custom TCP: Protocol: TCP, Port range: 30000 - 32767, Source: Anywhere(-), Description: optional.
- HTTP: Protocol: TCP, Port range: 80, Source: Anywhere(-), Description: optional.

At the bottom left is a 'Add rule' button.

## Step 1: Set Up EC2 Instances.

- 1) Set up 3 EC2 instances called master, node1, node2  
Select Amazon Linux as the OS image.

The screenshot shows the AWS Lambda console with a new function named 'HelloWorld' being created. The code editor contains the following Python code:

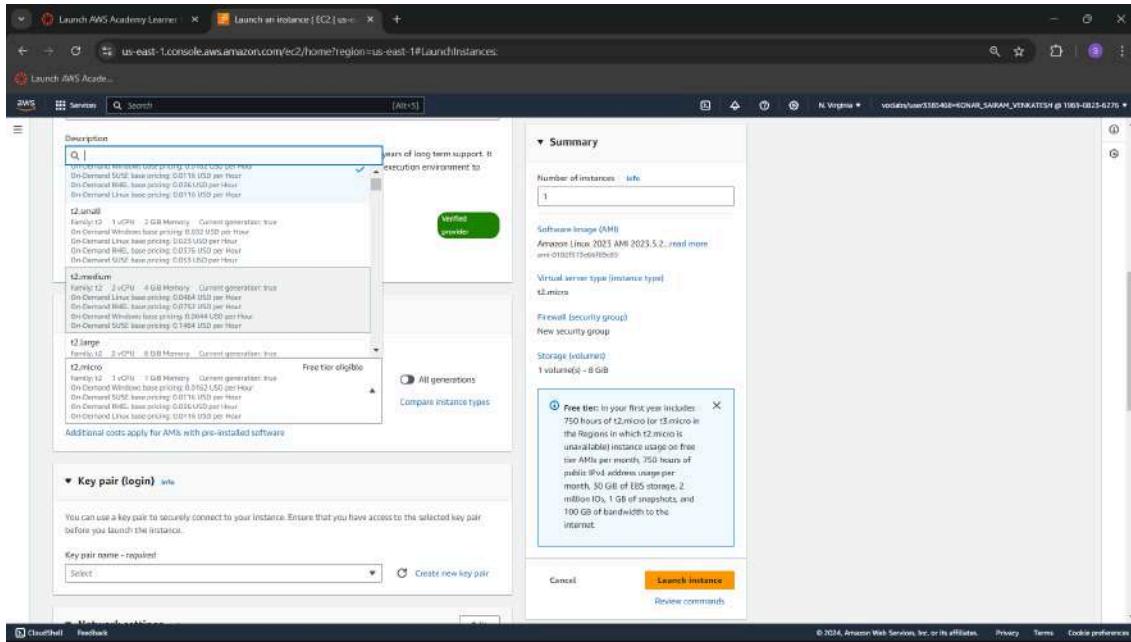
```
def lambda_handler(event, context):
    """This function takes a single string and returns it in a structured response object.
    """
    # Create a response.
    # The response must have a "body" key and a "statusCode" key.
    # Other keys like "headers", "isBase64Encoded", etc., are optional.
    response = {
        "statusCode": 200,
        "body": "Hello from Lambda"
    }

    # Return the response.
    return response
```

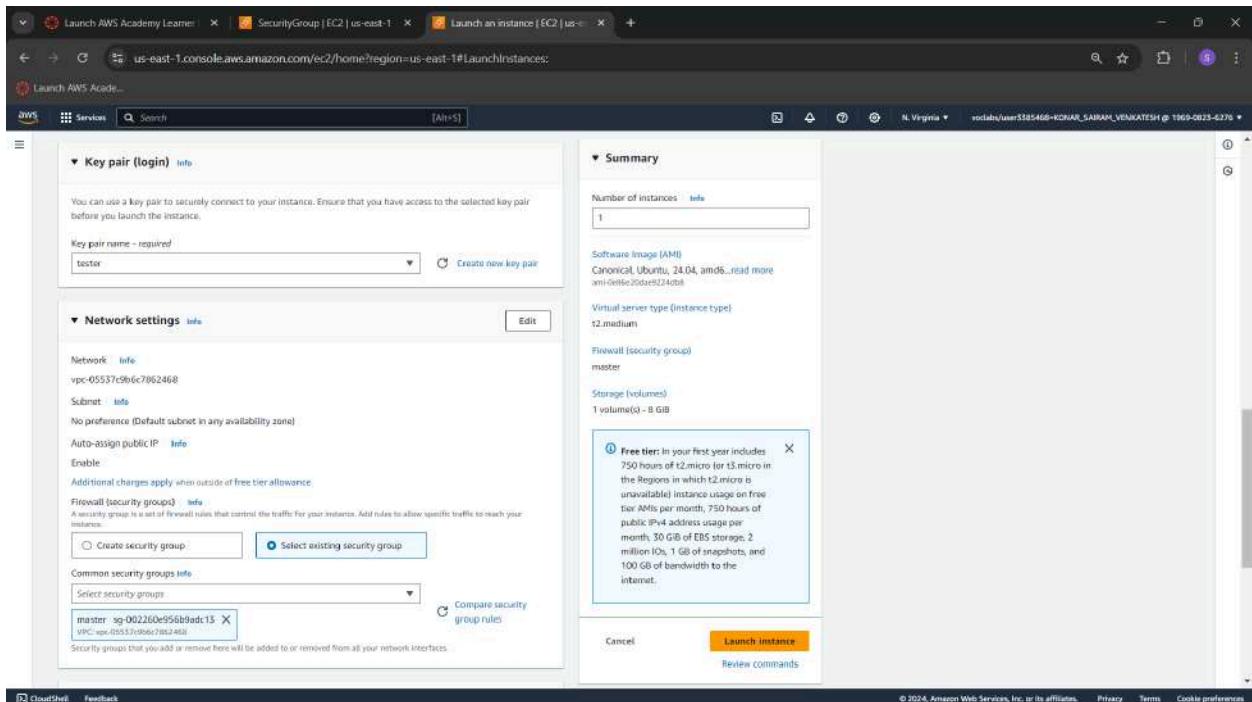
The configuration tabs show:

- Runtime:** Python 3.9
- Handler:** lambda\_function.lambda\_handler
- Role:** Lambda execution role
- Environment:** None
- Code:** Python 3.9 (1 file)
- Test:** None

**IMPORTANT:** The default instance type and free one provided by AWS is t2.micro, which provides only 1CPU and 1 GiB of memory. For running Kubernetes, a minimum of 2 CPUs and 2GiB of RAM is required, hence change **t2.micro** to **t2.medium**.



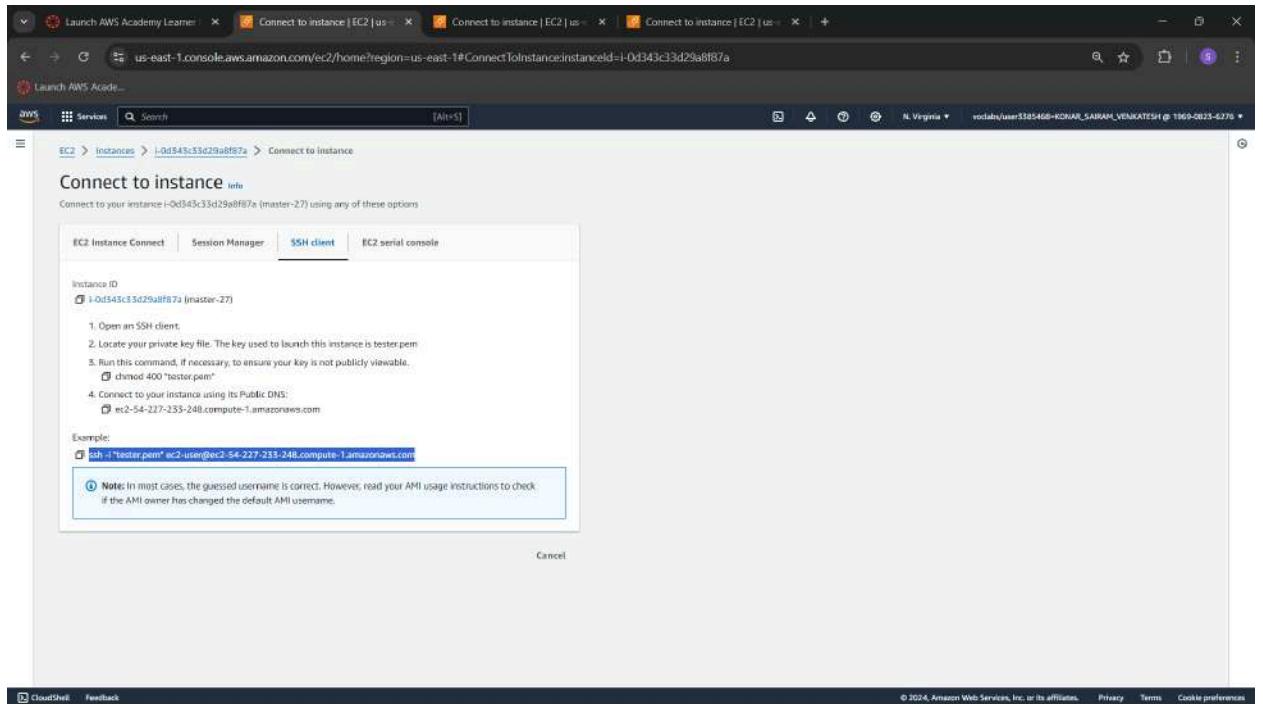
- 2) Create a key pair as you need the .pem file (private key file) on your system.
- 3) Click on 'Select existing security group' and select the master group for master instance, node group for both node instances.



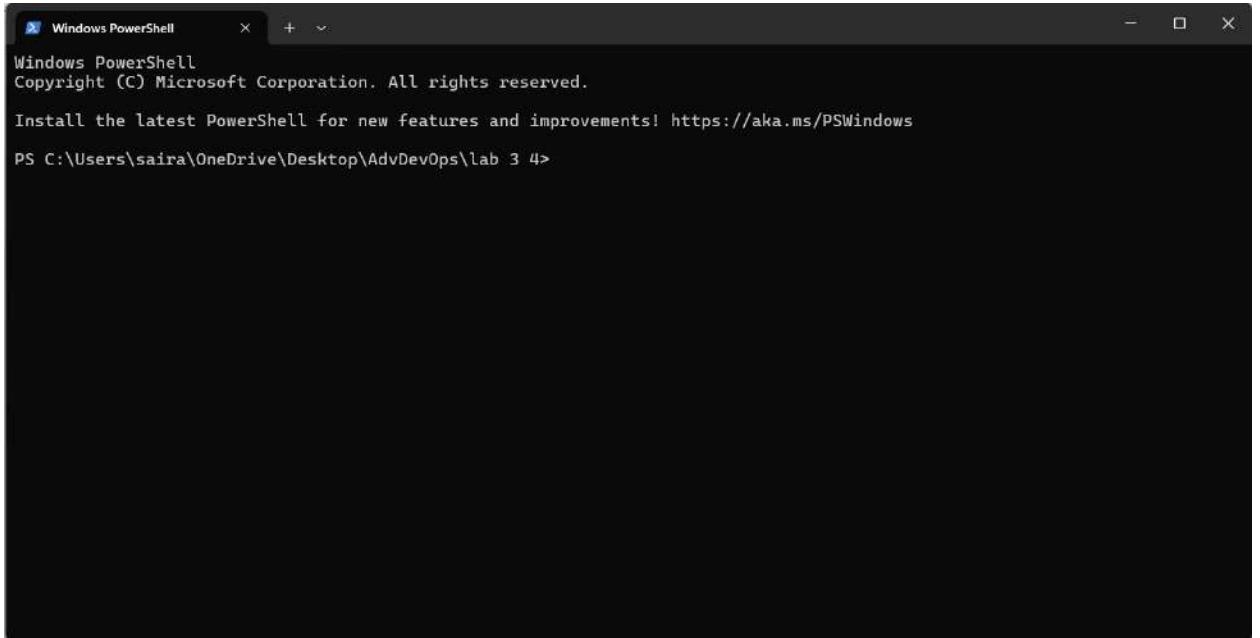
- 4) These are the instances that are created. Click on the Instance ID of master.

Instances (1/6) <a href="#">Info</a>		Last updated less than a minute ago	<a href="#">Connect</a>	Instance state <a href="#">▼</a>	Actions <a href="#">▼</a>	Launch instances <a href="#">▼</a>				
	Name <a href="#">▼</a>	Instance ID	Instance state <a href="#">▼</a>	Instance type <a href="#">▼</a>	Status check <a href="#">▼</a>	Alarm status	Availability Zone <a href="#">▼</a>	Public IPv4 DNS <a href="#">▼</a>	Public IPv4 <a href="#">▼</a>	Elastic IP <a href="#">▼</a>
<input type="checkbox"/>	node-2-27	i-09b6410d60907be52	Terminated <a href="#">View details</a> <a href="#">Logs</a>	t2.medium	...	<a href="#">View alarms</a> <a href="#">+</a>	us-east-1b	-	-	...
<input type="checkbox"/>	master27	i-03c6ec2e755806d08	Terminated <a href="#">View details</a> <a href="#">Logs</a>	t2.medium	...	<a href="#">View alarms</a> <a href="#">+</a>	us-east-1b	-	-	...
<input type="checkbox"/>	node-1-27	i-00a4bf5ab72137c96	Terminated <a href="#">View details</a> <a href="#">Logs</a>	t2.medium	...	<a href="#">View alarms</a> <a href="#">+</a>	us-east-1b	-	-	...
<input type="checkbox"/>	node2_27	i-0305ad251d191e2fe	Running <a href="#">View details</a> <a href="#">Logs</a>	t2.medium	...	<a href="#">View alarms</a> <a href="#">+</a>	us-east-1b	ec2-44-203-69-112.co...	44.203.69.112	...
<input checked="" type="checkbox"/>	master-27	i-0d343c33d29a8f07a	Running <a href="#">View details</a> <a href="#">Logs</a>	t2.medium	<a href="#">2/2 checks passed</a>	<a href="#">View alarms</a> <a href="#">+</a>	us-east-1b	ec2-54-227-233-248.co...	54.227.233.248	...
<input type="checkbox"/>	node1_27	i-019b032e030a20826	Running <a href="#">View details</a> <a href="#">Logs</a>	t2.medium	...	<a href="#">View alarms</a> <a href="#">+</a>	us-east-1b	ec2-3-89-90-121.comp...	3.89.90.121	...

- 5) Click on Connect. This directs you to a connect dashboard. Click on SSH client, you get a SSH command. Use this command to access the instance terminal on your local system.



- 6) Go to the folder where your private key file (.pem file) is installed. Right click → Open in terminal.



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

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

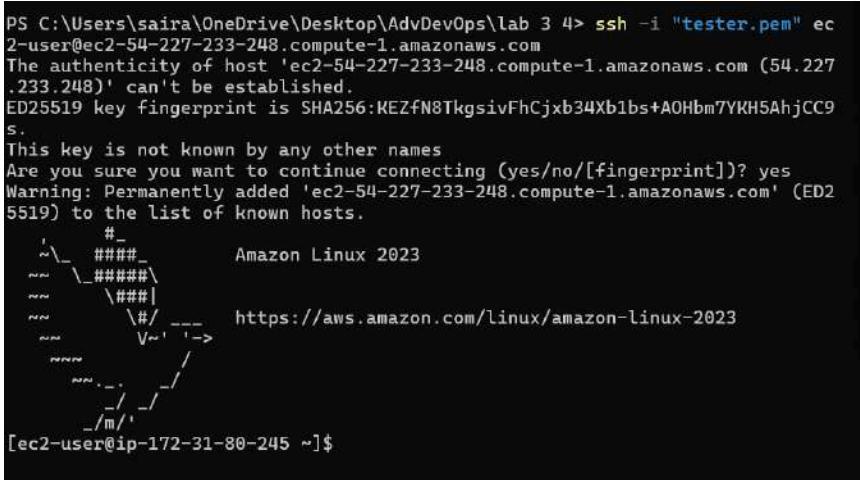
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab 3 4>
```

Paste the SSH command here and run it.

You might get the error of UNPROTECTED KEY FILE. This is because the .pem access is with all users of the system. Run the following commands to change the access to only the current user.

- `icacls "C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab 3 4\tester.pem" /inheritance:r`
- `icacls "C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab 3 4\tester.pem" /grant:r "%USERNAME%:F"`

Now rerun the SSH command.



```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab 3 4> ssh -i "tester.pem" ec2-user@ec2-54-227-233-248.compute-1.amazonaws.com
The authenticity of host 'ec2-54-227-233-248.compute-1.amazonaws.com (54.227.233.248)' can't be established.
ED25519 key fingerprint is SHA256:KEZfN8TkgsivFhCjxb34Xb1bs+A0Hbm7YKH5AhjCC9s.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-227-233-248.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
               _#
              ~\_ #####_      Amazon Linux 2023
             ~~ \#####\_
             ~~  \###|
             ~~   \#/ ___ https://aws.amazon.com/linux/amazon-linux-2023
             ~~    V~' '-->
             ~~~   _/_/
             ~~~  _/_/
             ~~~ /_/
             ~~~ /m/
[ec2-user@ip-172-31-80-245 ~]$
```

- 7) After doing the above steps, the terminal for all 3 nodes can be seen.

The image shows three separate terminal windows, each representing a different EC2 instance. Each window displays the following text:

```

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

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

PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab 3 4> ssh -i "tester.pem" ec2-user@ec2-54-227-233-248.compute-1.amazonaws.com
The authenticity of host 'ec2-54-227-233-248.compute-1.amazonaws.com (54.227.233.248)' can't be established.
ED25519 key fingerprint is SHA256:KEZfN8TkgsviHcJxb34Xb1bs+A0Hbe7YKHSAhjCC9s.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-227-233-248.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-172-31-80-245 ~]$
```

Below this, another terminal window shows a similar process for a different instance:

```

2-user@ec2-3-89-90-121.compute-1.amazonaws.com
The authenticity of host 'ec2-3-89-90-121.compute-1.amazonaws.com (3.89.90.121)' can't be established.
ED25519 key fingerprint is SHA256:THchr7pke1QtAjoA0Gl91aSd+fUCu8b2D/rW3jizY0o.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-89-90-121.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-172-31-86-11 ~]$
```

Finally, a third terminal window shows the same process for a third instance:

```

2-user@ec2-44-203-69-112.compute-1.amazonaws.com
The authenticity of host 'ec2-44-203-69-112.compute-1.amazonaws.com (44.203.69.112)' can't be established.
ED25519 key fingerprint is SHA256:+951lZ7Q193UkcmhX/ErjNsNLcxSubctHM5yrCSe6E.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-44-203-69-112.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-172-31-95-98 ~]$
```

## PERFORM THE FOLLOWING STEPS ON ALL 3 MACHINES

### Step 2: Installation of Docker

- 1) Use command

'sudo su'

This allows you to act as the root user of the terminal

```

/_m/
[ec2-user@ip-172-31-80-245 ~]$ sudo su
[root@ip-172-31-80-245 ec2-user]#
```

- 2) We can install docker using YUM(Yellowdog Updater, Modified). Use the command  
 'yum install docker -y'

```
[root@ip-172-31-80-245 ec2-user]# yum install docker -y
Last metadata expiration check: 0:05:22 ago on Thu Sep 26 03:53:56 2024.
Dependencies resolved.
=====
 Package           Arch    Version        Repository      Size
=====
Installing:
 docker            x86_64  25.0.6-1.amzn2023.0.2  amazonlinux   44 M
Installing dependencies:
 containerd         x86_64  1.7.20-1.amzn2023.0.1   amazonlinux   35 M
 iptables-libs     x86_64  1.8.8-3.amzn2023.0.2  amazonlinux  401 k
 iptables-nft      x86_64  1.8.8-3.amzn2023.0.2  amazonlinux  183 k
 libcgroup          x86_64  3.0-1.amzn2023.0.1   amazonlinux   75 k
 libnetfilter_conntrack x86_64  1.0.8-2.amzn2023.0.2  amazonlinux   58 k
 libnfnetwork      x86_64  1.0.1-19.amzn2023.0.2  amazonlinux  30 k
 libnftnl           x86_64  1.2.2-2.amzn2023.0.2  amazonlinux   84 k
 pigz              x86_64  2.5-1.amzn2023.0.3   amazonlinux   83 k
 runc              x86_64  1.1.13-1.amzn2023.0.1  amazonlinux   3.2 M

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

```
Installed:
 containerd-1.7.20-1.amzn2023.0.1.x86_64
 docker-25.0.6-1.amzn2023.0.2.x86_64
 iptables-libs-1.8.8-3.amzn2023.0.2.x86_64
 iptables-nft-1.8.8-3.amzn2023.0.2.x86_64
 libcgroup-3.0-1.amzn2023.0.1.x86_64
 libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64
 libnfnetwork-1.0.1-19.amzn2023.0.2.x86_64
 libnftnl-1.2.2-2.amzn2023.0.2.x86_64
 pigz-2.5-1.amzn2023.0.3.x86_64
 runc-1.1.13-1.amzn2023.0.1.x86_64
```

```
Complete!
[root@ip-172-31-80-245 ec2-user]# |
```

- 3) Now, configure a daemon.json file by using the following chain of commands.

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

- sudo systemctl daemon-reload
- sudo systemctl restart docker

```
[root@ip-172-31-80-245 ec2-user]# cd /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opt": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
EOF
sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opt": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service →
/usr/lib/systemd/system/docker.service.
[root@ip-172-31-80-245 docker]#
```

### Step 3: Installing Kubernetes

- 1) For installing kubernetes, we will be using kubeadm, a framework used for creating kubernetes clusters using command line.

<https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/>

The following will be visible when you visit the website.

The screenshot shows a web browser displaying the Kubernetes documentation for installing kubeadm. The URL in the address bar is <https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/>. The page content includes:

- A sidebar on the left with navigation links: Documentation, Getting started, Production environment, Container Runtimes, Installing Kubernetes with deployment tools, Bootstrapping clusters with kubeadm, and a detailed section on **Installing kubeadm**.
- A main content area with two tabs: **Debian-based distributions** and **Red Hat-based distributions**.
- Sub-sections for **Without a package manager** and **With a package manager**.
- Step 1: Set SELinux to **permissive** mode. It provides a shell command:
 

```
# Set SELinux to permissive mode (effectively disabling it)
sudo setenforce 0
sudo sed -i '/^SELINUX=enforcing$/i SELINUX=permissive' /etc/selinux/config
```
- Caution:**
  - Setting SELinux in permissive mode by running `setenforce 0` and `sed ...` effectively disables it. This is required to allow containers to access the host filesystem; for example, some cluster network plugins require that. You have to do this until SELinux support is improved in the kubelet.
  - You can leave SELinux enabled if you know how to configure it but it may require settings that are not supported by kubeadm.
- Step 2: Add the Kubernetes **yum** repository. The `include` parameter in the repository definition ensures that the packages related to Kubernetes are not upgraded upon running `yum update`, as there's a special procedure that must be followed for upgrading.
- A sidebar on the right with options: Edit this page, Create child page, Create documentation issue, Print entire section, and links to Before you begin, Verify the MAC address and product UUID, Check network adapters, Check required ports, Swap configuration, Installing a container runtime, Installing kubeadm, kubelet and kubectl, Configuring a cgroup driver, Troubleshooting, and What's next.

- 2) Select **red hat-based distributions** as amazon linux is based on red hat.

```
sudo setenforce 0
```

→ sets SELinux to permissive mode

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

→ edits the SELinux configuration file (/etc/selinux/config) to make the change persistent across reboots. If not used, SELinux reverts to enforcing mode after reboot.

Setting SELinux to permissive mode during Kubernetes installation prevents permission-related issues with container runtimes and components that may not function correctly under SELinux's enforcing policies.

Run the following commands:

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

This command is a repository script to create a Kubernetes repository

```
/usr/lib/systemd/system/docker.service
[root@ip-172-31-80-245 docker]# sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
[root@ip-172-31-80-245 docker]# cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repo/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repo/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
[root@ip-172-31-80-245 docker]#
```

- `yum repolist`

This command shows the repositories created on the machine.

```
[root@ip-172-31-80-245 docker]# yum repolist
repo id          repo name
amazonlinux      Amazon Linux 2023 repository
kernel-livepatch Amazon Linux 2023 Kernel Livepatch repository
kubernetes       Kubernetes
[root@ip-172-31-80-245 docker]#
```

Next step is to install kubelet, kubeadm, kubectl

- `sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes`

```
[root@ip-172-31-80-245 docker]# sudo yum install -y kubelet kubeadm kubectl
--disableexcludes=kubernetes
Kubernetes                               65 kB/s | 9.4 kB     00:00
Dependencies resolved.
=====
 Package           Arch   Version        Repository    Size
=====
 Installing:
  kubeadm          x86_64  1.31.1-150500.1.1  kubernetes   11 M
  kubectl          x86_64  1.31.1-150500.1.1  kubernetes   11 M
  kubelet          x86_64  1.31.1-150500.1.1  kubernetes   15 M
 Installing dependencies:
  conntrack-tools  x86_64  1.4.6-2.amzn2023.0.2  amazonlinux  208 k
  cri-tools        x86_64  1.31.1-150500.1.1  kubernetes   6.9 M
  kubernetes-cni   x86_64  1.5.1-150500.1.1  kubernetes   7.1 M
  libnetfilter_cthelper x86_64  1.0.0-21.amzn2023.0.2  amazonlinux  24 k
  libnetfilter_cttimeout  x86_64  1.0.0-19.amzn2023.0.2  amazonlinux  24 k
  libnetfilter_queue x86_64  1.0.5-2.amzn2023.0.2  amazonlinux  30 k
 Transaction Summary
=====
```

```
Installed:
  conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64
  cri-tools-1.31.1-150500.1.1.x86_64
  kubeadm-1.31.1-150500.1.1.x86_64
  kubectl-1.31.1-150500.1.1.x86_64
  kubelet-1.31.1-150500.1.1.x86_64
  kubernetes-cni-1.5.1-150500.1.1.x86_64
  libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64
  libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64
  libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64

Complete!
[root@ip-172-31-80-245 docker]# |
```

Now, we need to enable the kubelet service. Run the command

- `sudo systemctl enable --now kubelet`

```
[root@ip-172-31-80-245 docker]# sudo systemctl enable --now kubelet
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service
→ /usr/lib/systemd/system/kubelet.service.
[root@ip-172-31-80-245 docker]#
```

## PERFORM THE FOLLOWING ON ONLY THE MASTER MACHINE

- 1) Firstly, we need to initialize kubernetes. For this, run the command:

- kubeadm init

```
[root@ip-172-31-21-124 ~]# kubeadm init
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[WARNING FileExisting-socat]: socat not found in system path
[preflight] Verifying system call whitelisting for kubelet
[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'
[0913 17:19:05.142482 28614 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.6" of the container runtime is inconsistent with that used by kubeadm. It is recommended to use "registry.k8s.io/pause:3.10" as the CRT sandbox image.
[cores] Using cert-dir folder "/etc/kubernetes/pki"
[cores] Generating "ca" certificate and key
[cores] Generating "apiserver" certificate and key
[cores] apiserver serving cert is signed for DNS names [ip-172-31-21-124.ec2.internal kubernetes.kubernetes.default kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.21.124]
[cores] Generating "apiserver-kubelet-client" certificate and key
[cores] Generating "front-proxy-ca" certificate and key
[cores] Generating "front-proxy-client" certificate and key
[cores] Generating "etcd/ca" certificate and key
[cores] Generating "etcd/server" certificate and key
[cores] etcd/server serving cert is signed for DNS names [ip-172-31-21-124.ec2.internal localhost] and IPs [172.31.21.124 127.0.0.1 ::1]
[cores] etcd/peers serving cert is signed for DNS names [ip-172-31-21-124.ec2.internal localhost] and IPs [172.31.21.124 127.0.0.1 ::1]
[cores] Generating "etcd/healthcheck-client" certificate and key
[cores] Generating "apiserver-etcd-client" certificate and key
[cores] Generating "apiserver-ingress-client" certificate and key
[kubeconfig] Using existing config folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file
[kubeconfig] Writing "scheduler.conf" kubeconfig file
[etcd] creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"
[control-plane] Using manifest folder "/etc/kubernetes/manifests"

[api-check] Waiting for a healthy API server. This can take up to 4m0s
[api-check] The API server is healthy after 5.002506689s
[upload-config] Storing the configuration used in ConfigMap "kubeadm-config" in the "kube-system" Namespace
[kubebundle] Creating a ConfigMap "kubebundle-config" in namespace kube-system with the configuration for the kubebundles in the cluster
[upload-certs] Skipping phase. Please see --upload-certs
[mark-control-plane] Marking the node ip-172-31-21-124.ec2.internal as control-plane by adding the labels: {node-role.kubernetes.io/control-plane node.kubernetes.io/exclude-from-external-load-balancers}
[mark-control-plane] Marking the node ip-172-31-21-124.ec2.internal as control-plane by adding the taints {node-role.kubernetes.io/control-plane:NoSchedule}
[bootstrapping] Using token: 76rlql.qpkq9uvv8t9oagg4
[bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Roles
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to get nodes
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certificate credentials
[bootstrap-token] Configured RBAC rules to allow the etcdapprover 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
[bootstrapping] Finalized Update: "/etc/kubernetes/kubebundle-config" to point to a rotatable kubebundle 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 172.31.21.124:6443 --token 76rlql.qpkq9uvv8t9oagg4 \
    --discovery-token-ca-cert-hash sha256:a03f677cdcc93d202c5f89e2370eb7231d212211bac6f816cddcb547b7f00a452
[root@ip-172-31-21-124 ~]#
```

From the output, we will receive a command that is used to link the nodes to the master.

Copy it and save it somewhere local.

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

```
kubeadm join 172.31.80.245:6443 --token g7dyjj.r62w19oc05n7k4kc \
    --discovery-token-ca-cert-hash sha256:2975de445064ffc56887b7b0ef79d7
1-7ec611bc0-20520016fc-522616725-
```

- 2) From the output, we receive the following commands:

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

Run these commands.

```
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
```

- 3) To check whether nodes are connected, run the command

- kubectl get nodes

This output shows only master is connected right now.

```
[root@ip-172-31-80-245 docker]# kubectl get nodes
NAME           STATUS    ROLES      AGE   VERSION
ip-172-31-80-245.ec2.internal   NotReady  control-plane  112s  v1.31.1
[root@ip-172-31-80-245 docker]#
```

## PERFORM THE FOLLOWING ONLY ON THE NODE MACHINES

Use the command that you had copied before and use them on the node machines.

The image displays two separate terminal sessions, each showing the execution of the `kubeadm init --node` command on a specific node machine. Both sessions show the process of writing kubelet configuration files, starting the kubelet service, and performing a TLS Bootstrap. They also indicate that the node has joined the cluster successfully after sending a certificate signing request to the API server.

```
root@ip-172-31-86-11:/etc/do ~ + - X
em get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/conf
ig.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/li
b/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/heal
thz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 501.57413ms
[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 recei
ved.
* The Kubelet was informed of the new secure connection details.

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

[root@ip-172-31-86-11 docker]#
```

```
root@ip-172-31-95-90:/etc/dc ~ + - X
em get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/conf
ig.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/li
b/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/heal
thz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 1.001297846s
[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 recei
ved.
* The Kubelet was informed of the new secure connection details.

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

[root@ip-172-31-95-90 docker]#
```

NOW GO BACK TO THE MASTER MACHINE AND RERUN 'kubectl get nodes'

```
[root@ip-172-31-80-245 docker]# kubectl get nodes
NAME           STATUS   ROLES      AGE   VERSION
ip-172-31-80-245.ec2.internal   NotReady   control-plane   2m18s   v1.31.1
ip-172-31-86-11.ec2.internal   NotReady   <none>        9s     v1.31.1
ip-172-31-95-90.ec2.internal   NotReady   <none>        2s     v1.31.1
[root@ip-172-31-80-245 docker]# |
```

As we see, the status of the nodes are <NOT READY>. To change it, we need to install a network CNI plugin.

**Use the following command only on the master machine::**

kubectl apply -f

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

Now run ‘kubectl get nodes’ again.

```
[root@ip-172-31-80-245 docker]# kubectl get nodes
NAME           STATUS   ROLES      AGE   VERSION
ip-172-31-80-245.ec2.internal   Ready    control-plane   9m33s   v1.31.1
ip-172-31-86-11.ec2.internal   Ready    <none>        7m24s   v1.31.1
ip-172-31-95-90.ec2.internal   Ready    <none>        7m17s   v1.31.1
[root@ip-172-31-80-245 docker]# |
```

### Conclusion:

In this experiment, we have learned how to create kubernetes clusters on a linux terminal, how the ssh command works on a local terminal and what requirements are necessary to create kubernetes clusters. We have used many command inline tools of Kubernetes like kubecmd, kubectl to set up the clusters and work with docker container to perform the connection of the nodes with master machine.

## Experiment 4

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

Prerequisites:

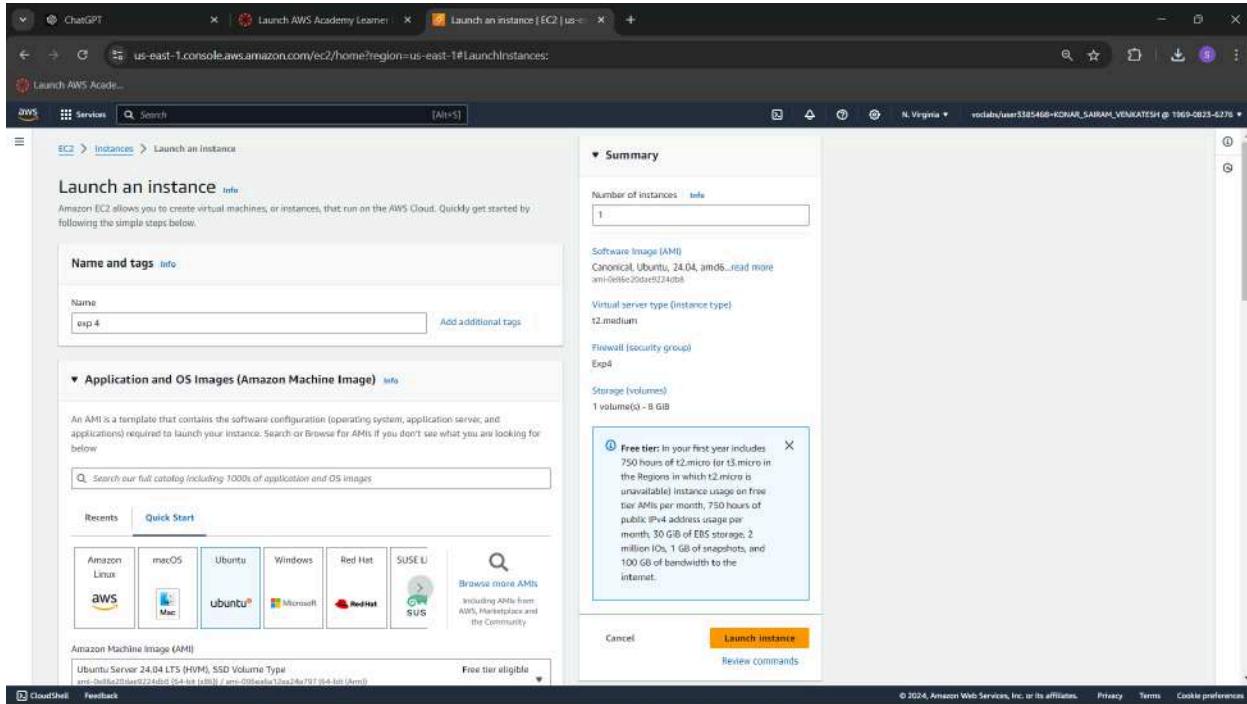
Create a security group by navigating to Search → Security Groups.

The screenshot shows the AWS CloudWatch Metrics search interface. A search bar at the top contains the query "Security". Below the search bar, there are two main sections: "Services" and "Features". Under "Services", there are four items: "Security Lake", "Security Hub", "Detective", and "Amazon Inspector". Under "Features", there is one item: "Security groups". On the right side of the interface, there is a list of EC2 instances with columns for "Alarm status", "Availability Zone", and "Public IP". One instance, "us-east-1b", is highlighted.

The screenshot shows the "Create security group" wizard in the AWS EC2 console. The first step, "Basic details", is completed with the security group name "Exp4" and a description "Allows SSH access to developers". The VPC dropdown is set to "vpc-05337c86c7863468". The second step, "Inbound rules", shows two rules defined: one for SSH (Protocol TCP, Port range 22, Source Anywhere) and one for Custom TCP (Protocol TCP, Port range 6443, Source Anywhere). Both rules have a "Delete" button next to them. At the bottom of the page, there are links for "cloudShell" and "Feedback".

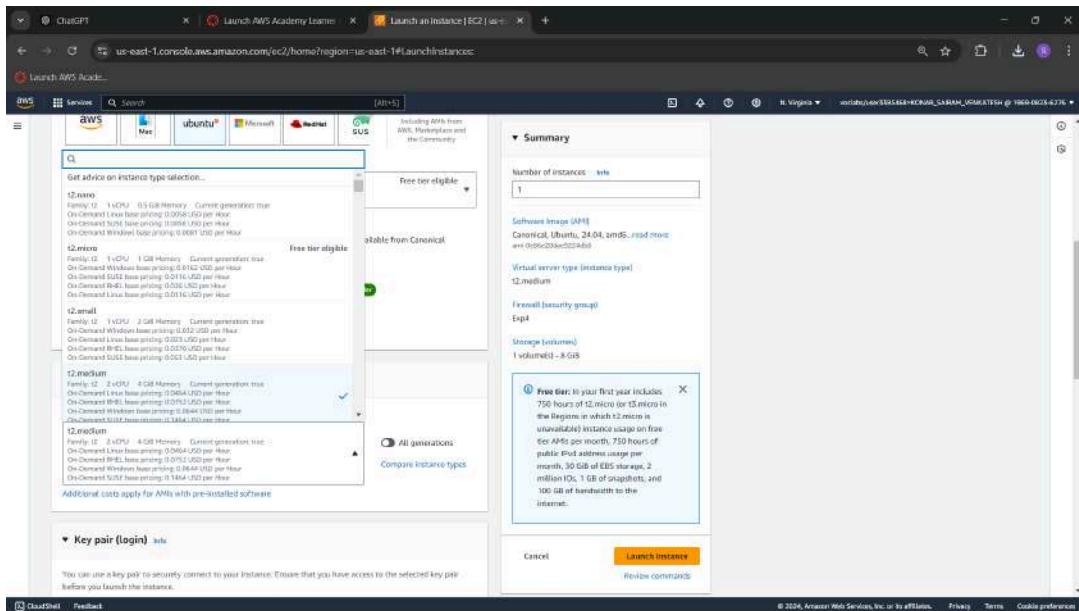
## Step 1: Creation of instances.

- 1) Go to the AWS Dashboard. In Services, search for EC2. Open it. Click on Launch instance,

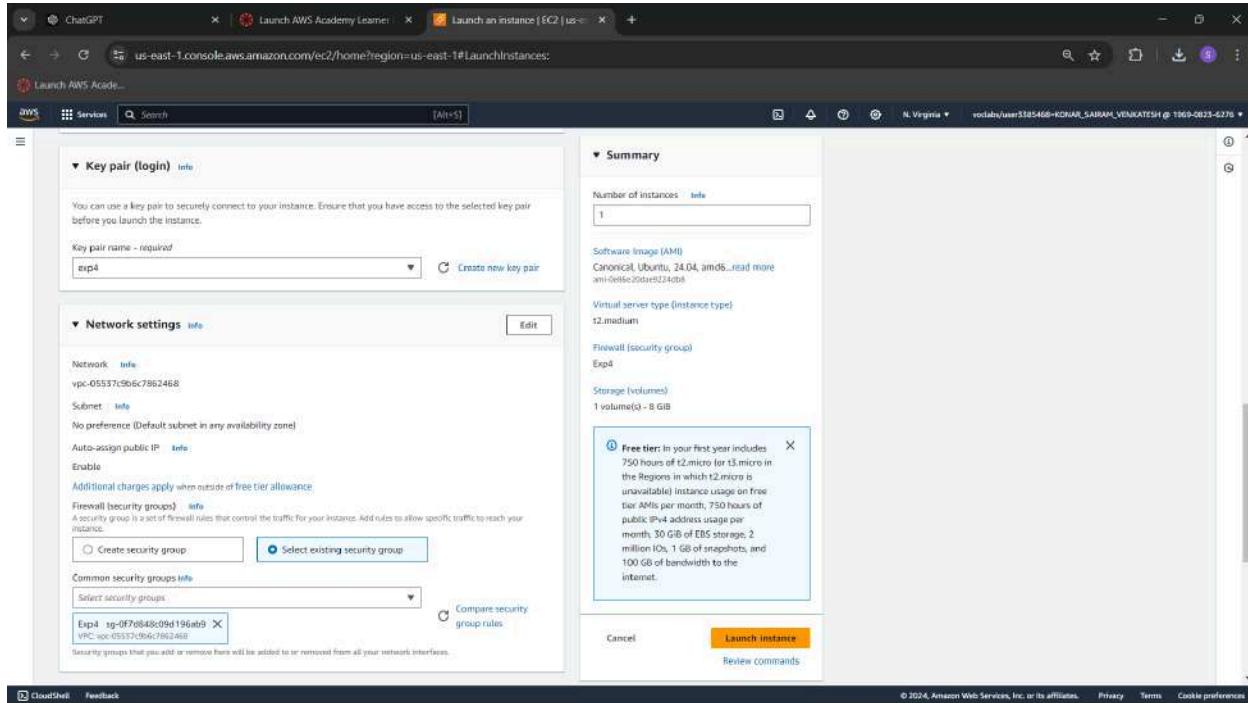


- 2) Select ubuntu 22.04 as you OS image.

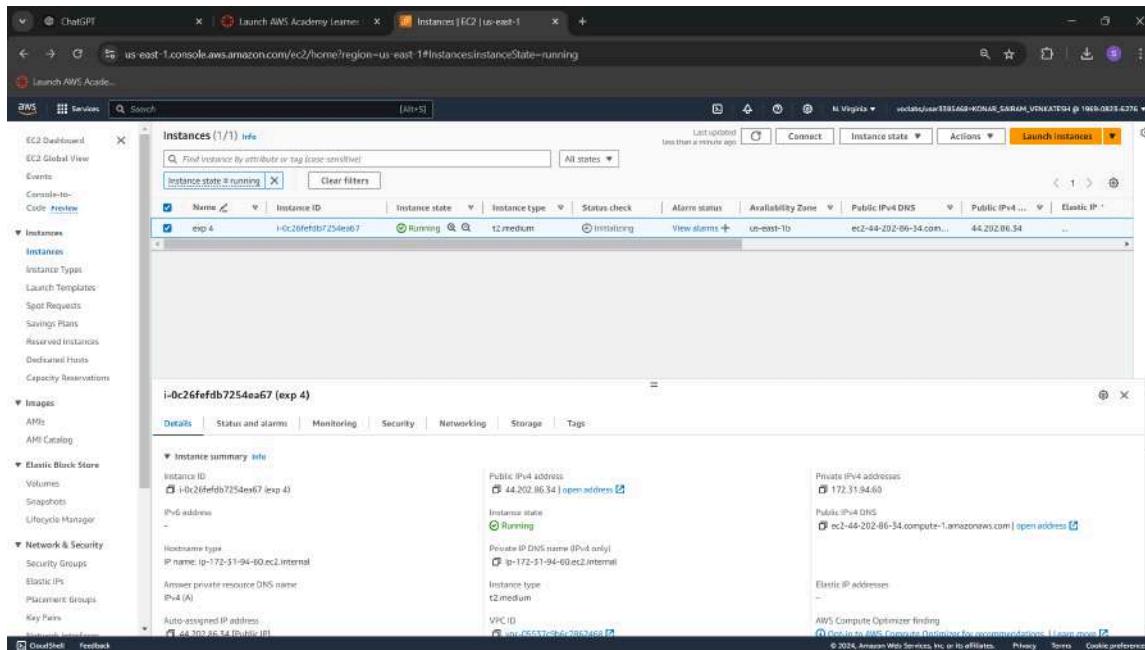
**IMPORTANT:** The default instance type and free one provided by AWS is t2.micro, which provides only 1CPU and 1 GiB of memory. For running Kubernetes, a minimum of 2 CPUs and 2GiB of RAM is required, hence change **t2.micro** to **t2.medium**.



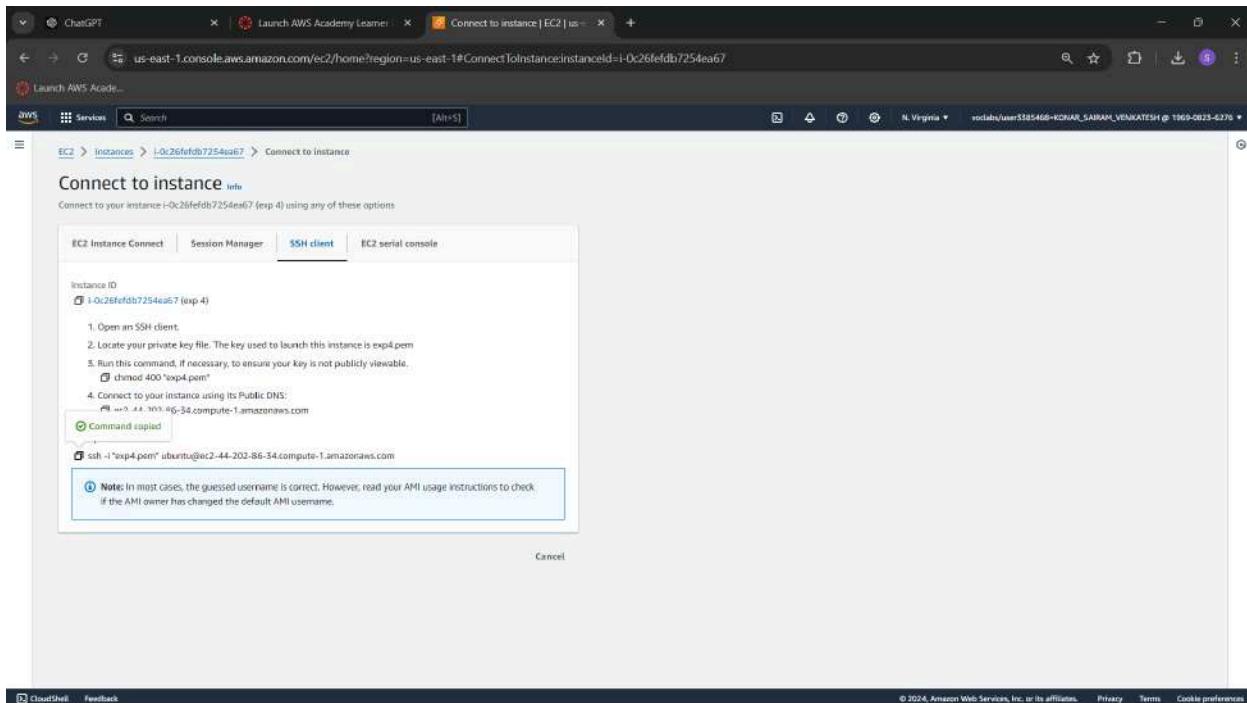
- 3) Create a key pair as you need the .pem file (private key file) on your system.
- 4) Click on 'Select existing security group' and select the exp4 group for the instance.



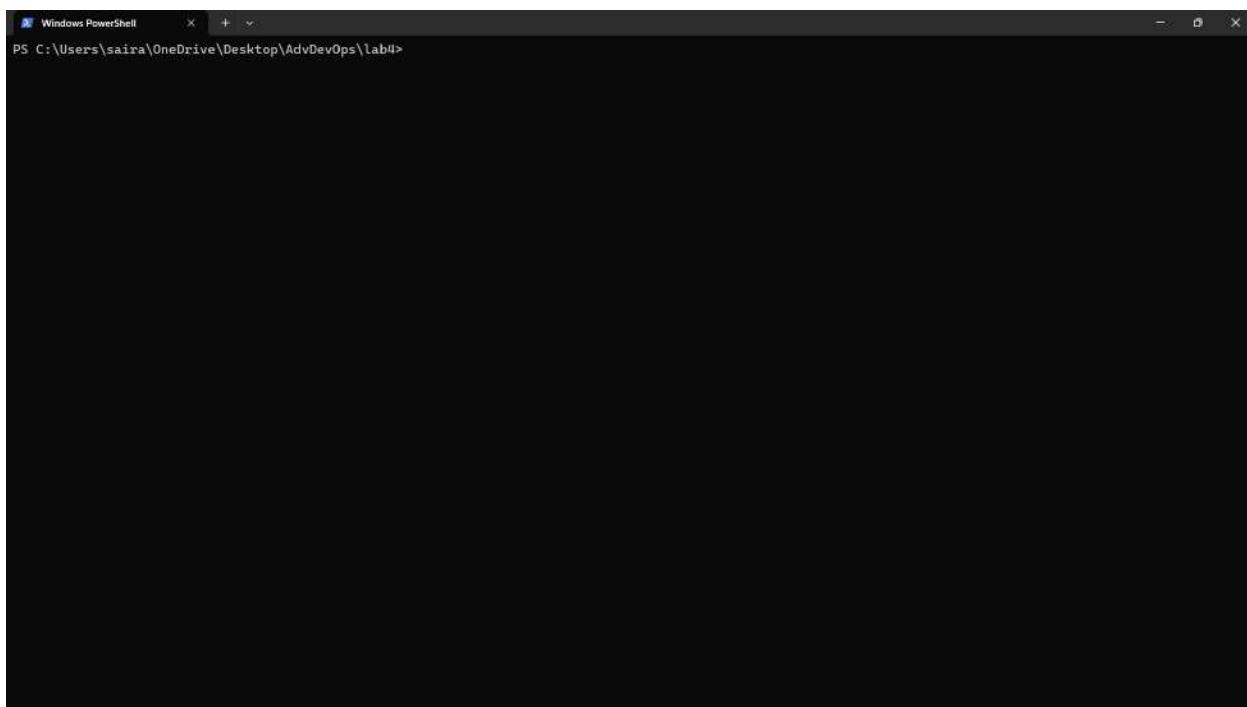
- 5) The instance is created. Click on the instance id, then click on connect.



- 6) Click on Connect. This directs you to a connect dashboard. Click on SSH client, you get a SSH command. Use this command to access the instance terminal on your local system.



- 7) Go to the folder where your private key file (.pem file) is installed. Right click → Open in terminal.



Paste the SSH command here and run it.

You might get the error of UNPROTECTED KEY FILE. This is because the .pem access is with all users of the system. Run the following commands to change the access to only the current user.

- icacls "C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab 3 4\tester.pem" /inheritance:r
- icacls "C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab 3 4\tester.pem" /grant:"%USERNAME%:F"

Now rerun the SSH command.

```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab4> ssh -i "exp4.pem" ubuntu@ec2-44-203-88-254.compute-1.amazonaws.com
The authenticity of host 'ec2-44-203-88-254.compute-1.amazonaws.com (44.203.88.254)' can't be established.
ED25519 key fingerprint is SHA256:4a09wFPIj0+zy4hJHl4dpRt+zJg0kyYRFICt3/gbBI.
This host key is known by the following other names/addresses:
  C:\Users\saira/.ssh/known_hosts:47: ec2-44-202-86-34.compute-1.amazonaws.com
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-44-203-88-254.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 Mon Sep 30 05:19:42 UTC 2024

System load: 1.08      Processes:          137
Usage of /: 55.5% of 6.71GB  Users logged in:   0
Memory usage: 13%        IPv4 address for enX0: 172.31.94.60
Swap usage:  0%

Expanded Security Maintenance for Applications is not enabled.

147 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: Mon Sep 30 04:42:33 2024 from 103.197.221.172
ubuntu@ip-172-31-94-60:~$ |
```

## Step 2: Setup Docker

- 1) We have to install and setup Docker. Run these commands
  - 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-172-31-94-60:~$ 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
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
ubuntu@ip-172-31-94-60:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
W: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has an unsupported filetype.
ubuntu@ip-172-31-94-60:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null
ubuntu@ip-172-31-94-60:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
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 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
```

```
Get:45 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [68.1 kB]
Get:46 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:47 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [428 B]
Get:48 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [10.9 kB]
Get:49 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [2808 B]
Get:50 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 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 (7836 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-172-31-94-60:~$ |
```

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

```
ubuntu@ip-172-31-94-60:~$ 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 https://download.docker.com/linux/ubuntu noble InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
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.
Setting up docker-ce (5:27.3.1-1~ubuntu.24.04-noble) ...
Setting up dockerd (1.2.1-1build2) ...
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-172-31-94-60:~$ |
```

- sudo mkdir -p /etc/docker
- cat <<EOF | sudo tee /etc/docker/daemon.json
 

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

```
ubuntu@ip-172-31-94-60:~$ sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-94-60:~$ |
```

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

```
ubuntu@ip-172-31-94-60:~$ 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 3: Set up 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-172-31-94-60:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
ubuntu@ip-172-31-94-60:~$ 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-172-31-94-60:~$ |
```

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

```
ubuntu@ip-172-31-94-60:~$ 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
```

```
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-172-31-94-60:~$ |
```

#### Step 4: Initialise the kubecluster

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

```
ubuntu@ip-172-31-94-60:~$ 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
W0930 04:29:33.758696    4337 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.
```

Here, we encounter an error as a few of the dependencies for running the command are not installed. So, run the following commands

- sudo apt-get install -y containerd

```
ubuntu@ip-172-31-94-60:~$ 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 143 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
```

```
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-172-31-94-60:~$ |
```

- sudo mkdir -p /etc/containerd
- sudo containerd config default | sudo tee /etc/containerd/config.toml

```
ubuntu@ip-172-31-94-60:~$ 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
```

```
[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-172-31-94-60:~$
```

- sudo systemctl restart containerd
- sudo systemctl enable containerd
- sudo systemctl status containerd

```
ubuntu@ip-172-31-94-60:~$ 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 Mon 2024-09-30 04:32:57 UTC; 200ms ago
       Docs: https://containerd.io
      Main PID: 4809 (containerd)
        Tasks: 7
         Memory: 13.4M (peak: 13.8M)
            CPU: 51ms
           CGroup: /system.slice/containerd.service
                   └─4809 /usr/bin/containerd

Sep 30 04:32:57 ip-172-31-94-68 containerd[4809]: time="2024-09-30T04:32:57.321902576Z" level=info msg="serving..." address=/run/containerd/containerd.sock
Sep 30 04:32:57 ip-172-31-94-68 containerd[4809]: time="2024-09-30T04:32:57.321928176Z" level=info msg="Start subscribing containerd event"
Sep 30 04:32:57 ip-172-31-94-68 containerd[4809]: time="2024-09-30T04:32:57.321986090Z" level=info msg="Start recovering state"
Sep 30 04:32:57 ip-172-31-94-68 containerd[4809]: time="2024-09-30T04:32:57.322036817Z" level=info msg="Start event monitor"
Sep 30 04:32:57 ip-172-31-94-68 containerd[4809]: time="2024-09-30T04:32:57.322068916Z" level=info msg="Start snapshots syncer"
Sep 30 04:32:57 ip-172-31-94-68 containerd[4809]: time="2024-09-30T04:32:57.322081044Z" level=info msg="Start cni network conf syncer for default"
Sep 30 04:32:57 ip-172-31-94-68 containerd[4809]: time="2024-09-30T04:32:57.322092347Z" level=info msg="Start streaming server"
Sep 30 04:32:57 ip-172-31-94-68 containerd[4809]: time="2024-09-30T04:32:57.321933210Z" level=info msg="serving..." address=/run/containerd/containerd.sock
Sep 30 04:32:57 ip-172-31-94-68 containerd[4809]: time="2024-09-30T04:32:57.322165177Z" level=info msg="containerd successfully booted in 0.020892s"
Sep 30 04:32:57 ip-172-31-94-68 systemd[1]: Started containerd.service - containerd container runtime.
ubuntu@ip-172-31-94-60:~$ |
```

- sudo apt-get install -y socat

```
ubuntu@ip-172-31-94-60:~$ 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 143 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 (11.6 kB/s)
Selecting previously unselected package socat.
(Reading database ... 68188 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.0-4build3_amd64.deb ...
Unpacking socat (1.8.0.0-4build3) ...
Setting up socat (1.8.0.0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-94-60:~$ |
```

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

```
ubuntu@ip-172-31-94-60:~$ 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'
W0930 04:34:21.946260      5034 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-172-31-94-68 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.94.68]
[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-172-31-94-68 localhost] and IPs [172.31.94.68 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-94-68 localhost] and IPs [172.31.94.68 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
```

```
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 172.31.94.60:6443 --token bhzgoy.c4u84a2hc4bf7u19 \
    --discovery-token-ca-cert-hash sha256:9d14667afdeed67b83ac64ae48deb6b315e62070260dd78c7d18b1db4c94b195
ubuntu@ip-172-31-94-60:~$ |
```

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

```
ubuntu@ip-172-31-94-60:~$ 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
ubuntu@ip-172-31-94-60:~$ |
```

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

```
ubuntu@ip-172-31-94-60:~$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
ubuntu@ip-172-31-94-60:~$ |
```

## Step 5: Connect Ngix server to pod.

- kubectl get pods

```
ubuntu@ip-172-31-94-60:~$ kubectl get pods
NAME                      READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-4cbnz   0/1     Pending   0          28s
nginx-deployment-d556bf558-zk989   0/1     Pending   0          28s
ubuntu@ip-172-31-94-60:~$ |
```

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

```
ubuntu@ip-172-31-94-60:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
ubuntu@ip-172-31-94-60:~$ kubectl port-forward $POD_NAME 8080:80
error: unable to forward port because pod is not running. Current status=Pending
ubuntu@ip-172-31-94-60:~$ |
```

Here, another error is generated as the pods are not ready. To make them ready, we need to untaint them. Run the following commands.

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

```
ubuntu@ip-172-31-94-60:~$ kubectl taint nodes --all node-role.kubernetes.io/control-plane:NoSchedule-
node/ip-172-31-94-60 untainted
ubuntu@ip-172-31-94-60:~$ kubectl get nodes
NAME           STATUS    ROLES      AGE     VERSION
ip-172-31-94-60   Ready    control-plane   5m36s   v1.31.1
ubuntu@ip-172-31-94-60:~$ |
```

- kubectl get pods

```
ubuntu@ip-172-31-94-60:~$ kubectl get pods
NAME                  READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-4cbnz   1/1     Running   0          4m36s
nginx-deployment-d556bf558-zk989   1/1     Running   0          4m36s
ubuntu@ip-172-31-94-60:~$ |
```

Now rerun 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-172-31-94-60:~$ 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
|
```

Open another terminal and connect ssh to that terminal as well.

```
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\saira> cd '.\OneDrive\Desktop\AdvDevOps\lab4'
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab4> ssh -i "exp4.pem" ubuntu@ec2-44-202-86-34.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 Mon Sep 30 04:42:33 UTC 2024

 System load:  0.12      Processes:          153
 Usage of /:   55.3% of 6.71GB   Users logged in:     1
 Memory usage: 19%           IPv4 address for enX0: 172.31.94.60
 Swap usage:   0%
 
Expanded Security Maintenance for Applications is not enabled.

147 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: Mon Sep 30 04:22:55 2024 from 103.197.221.172
ubuntu@ip-172-31-94-60:~$ |
```

Run the command curl --head <http://127.0.0.1:8080>

```
ubuntu@ip-172-31-94-60:~$ curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Mon, 30 Sep 2024 04:43:48 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
```

If it shows status code of 200 OK, it means that the deployment of nginx server was successful.

Go back to the previous terminal. After running curl, the output for port forward looks like this

```
ubuntu@ip-172-31-94-60:~$ 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
|
```

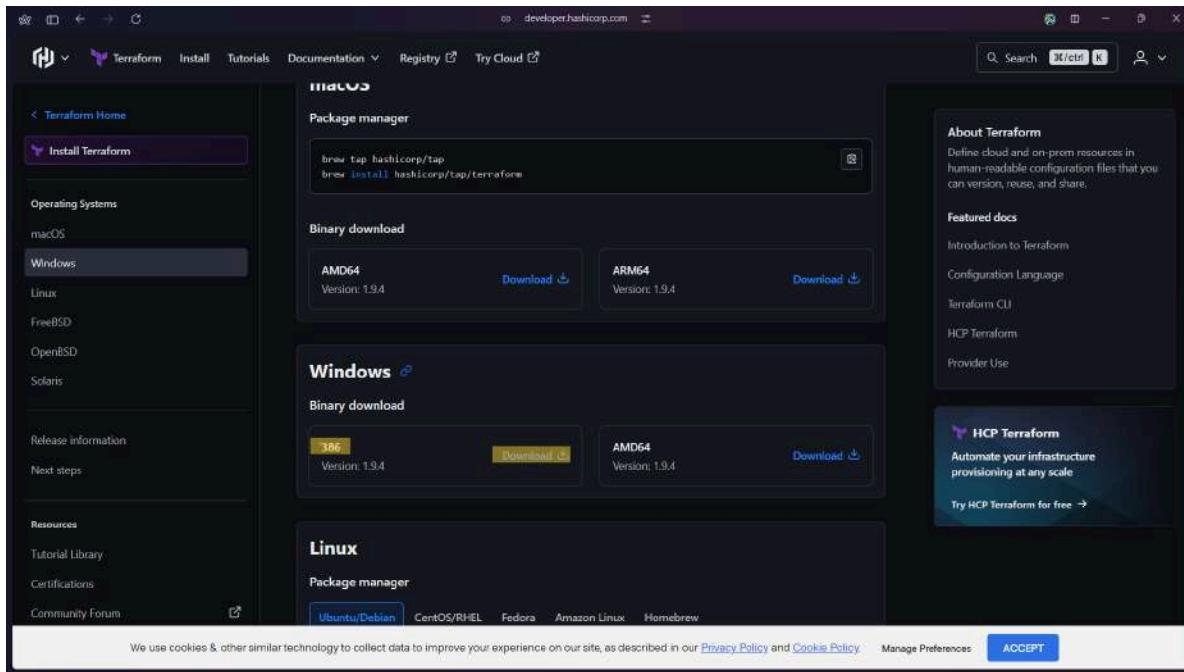
### Conclusion:

In this experiment, we have successfully installed kubectl and deployed our first Kubernetes Application. After setting up the instance, we connect it to our local terminal using SSH. The application we deployed was an nginx server. To deploy this, we had to install flannel, a common networking plugin. There were a few errors. One being while initializing the cluster, which was solved by installing the missing dependencies. The second one was when we checked the pods that were created. These pods were in the not ready state as they had some issues. To solve this, we had to untaint them. Once all the errors were resolved, we run the port command to initiate the connection. When we run the curl command, we get a OK status. Hence the deployment was complete.

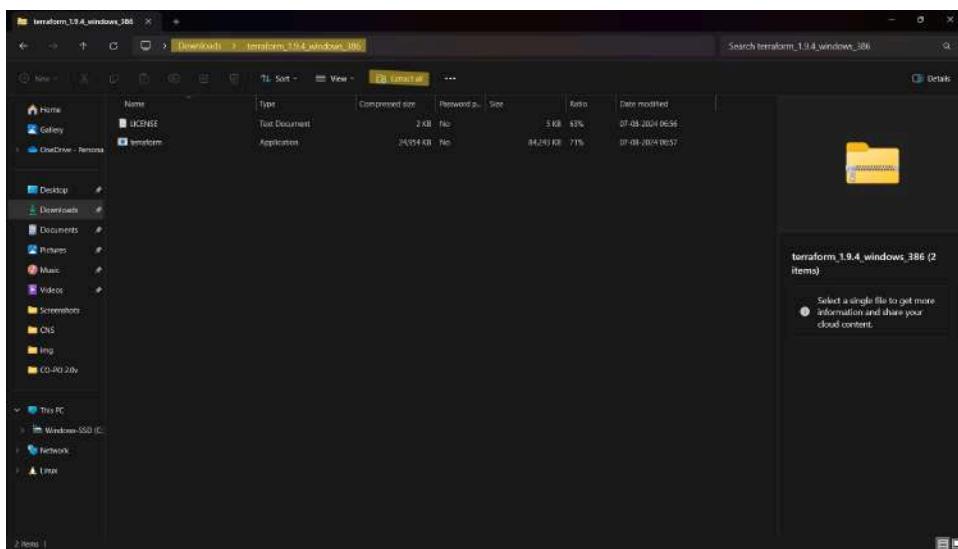
## Experiment 5

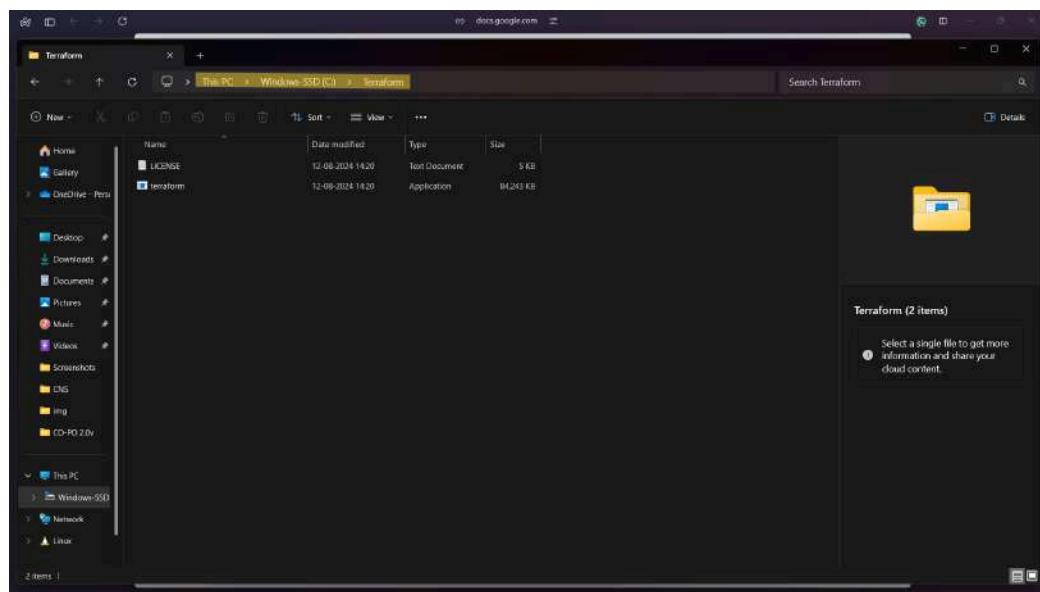
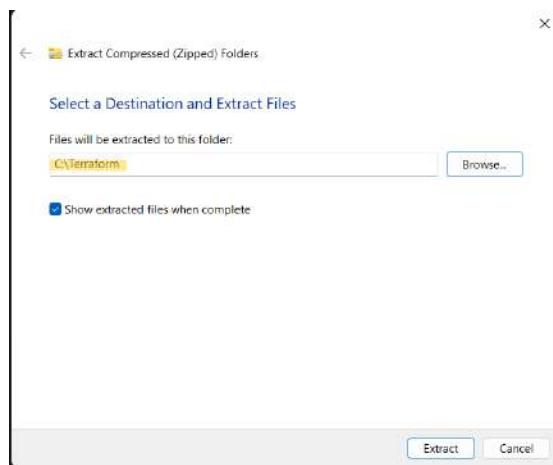
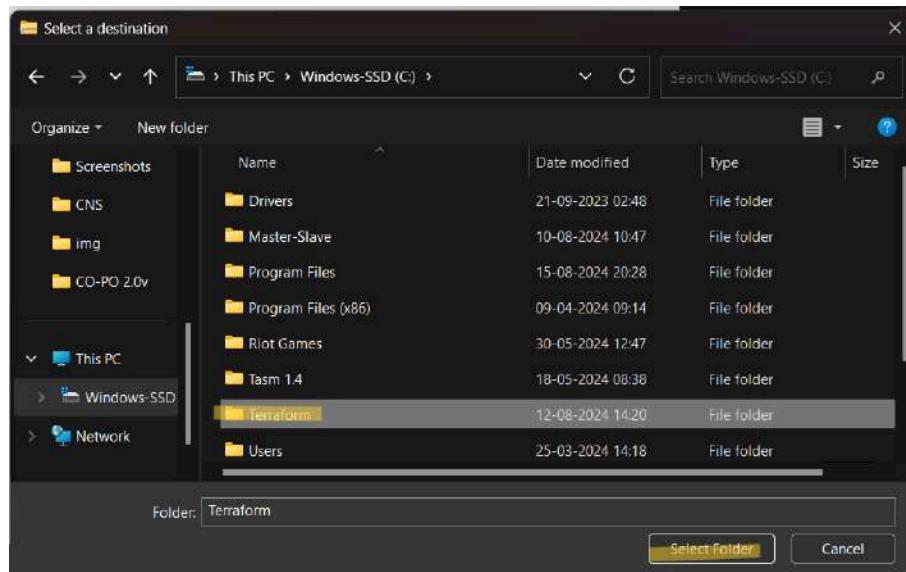
**Aim:** To understand terraform lifecycle, core concepts/terminologies and install it on a Linux Machine and Windows.

**Step 1:** Visit <https://developer.hashicorp.com/terraform/install>. Scroll down to the operating system on which you want to install Terraform. For Windows, select 386 option.

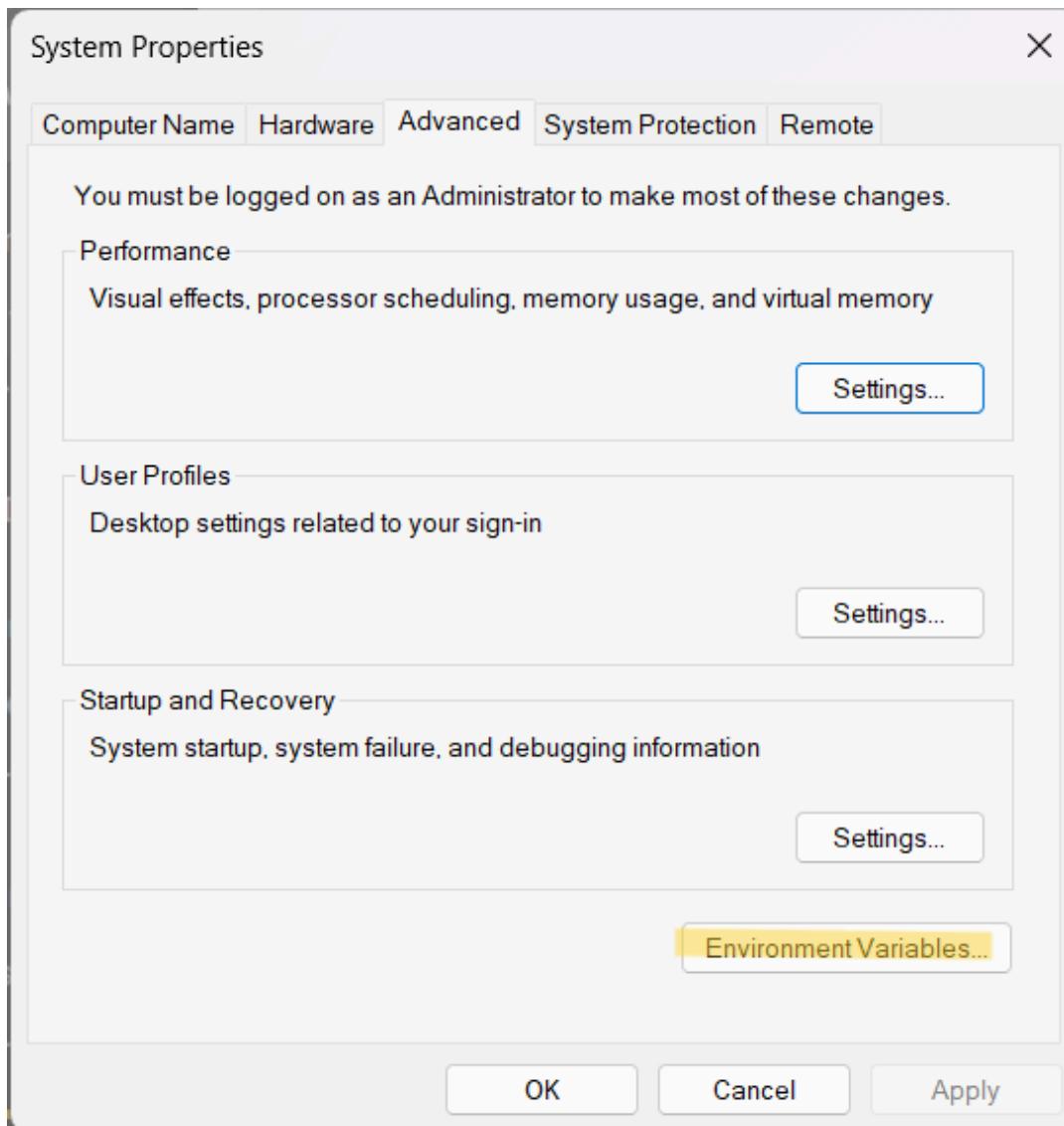


**Step 2:** A zip file gets downloaded to your system. Create a folder on your C-drive named as Terraform. Go to the zip you downloaded, click on extract all, and select the C-drive file we just created. This will extract the Folder in C:\Terraform.



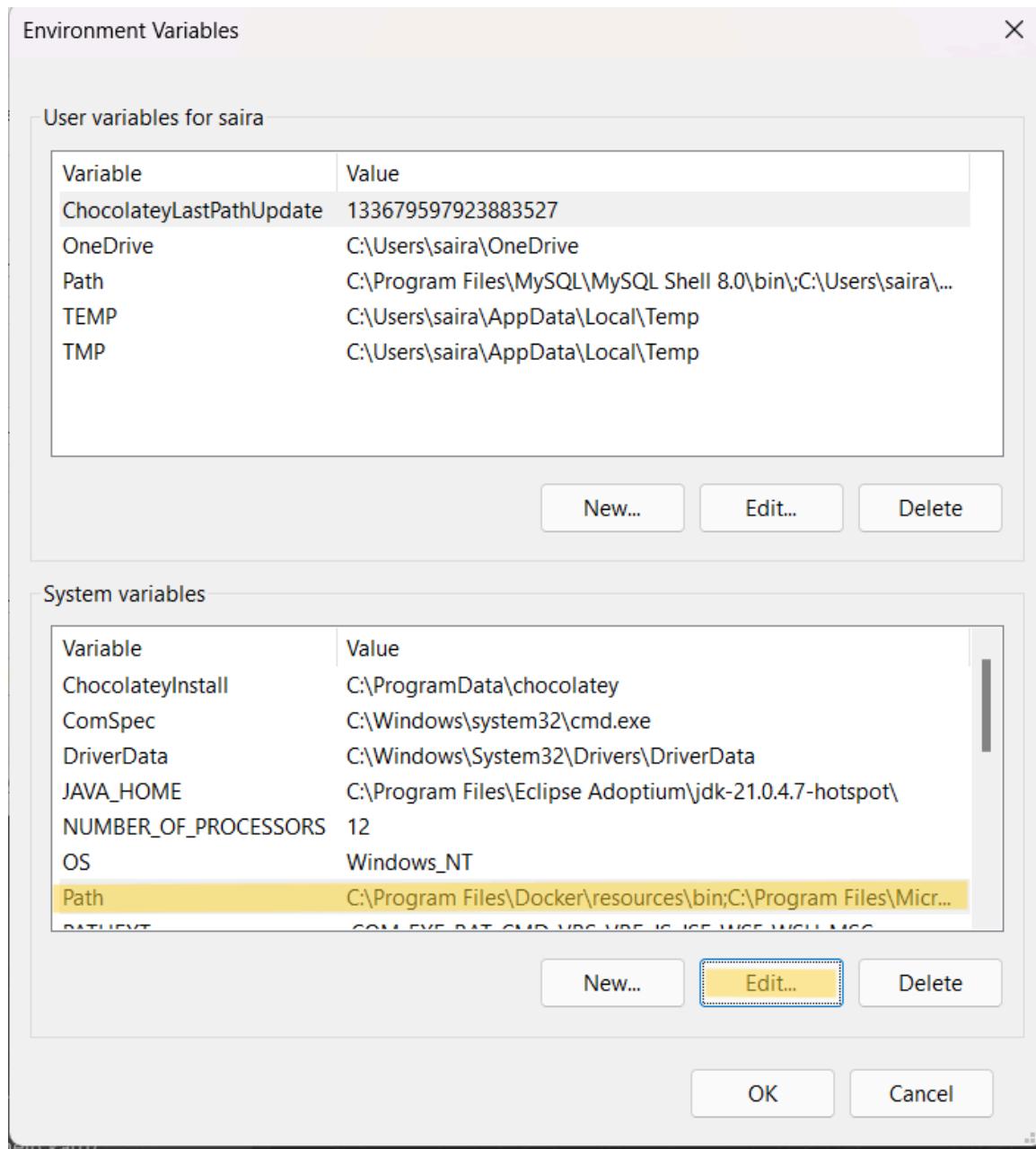


**Step 3:** After extracting the files, we need to add the folder path as a User System variable so that the commands could be used in the command prompt or powershell. For this, search Edit the system environment variables on you system. Open it, click on Environment Variables.

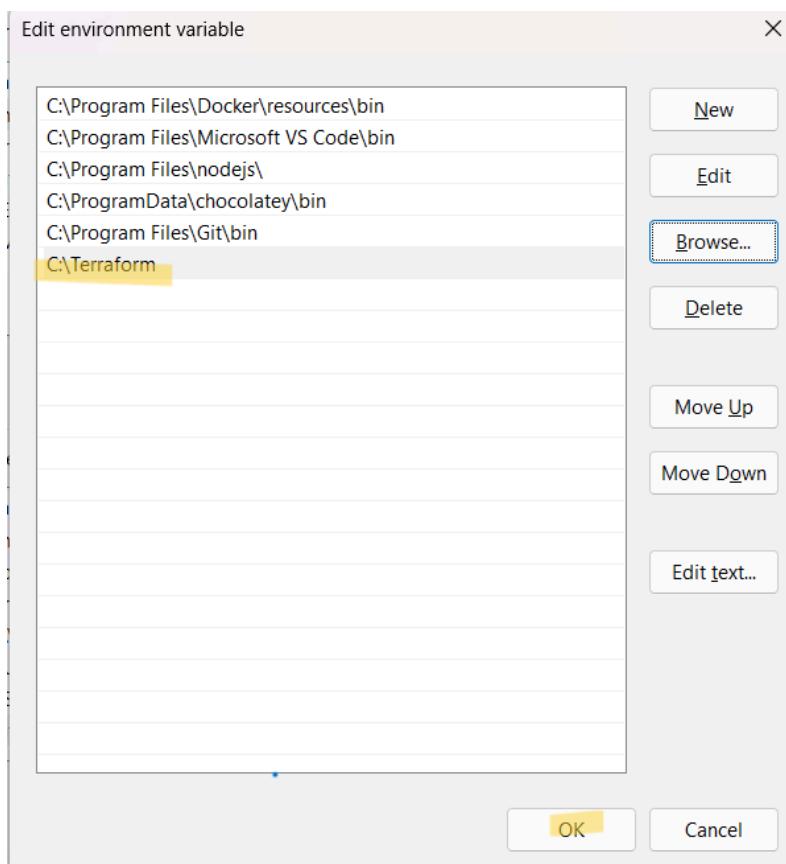
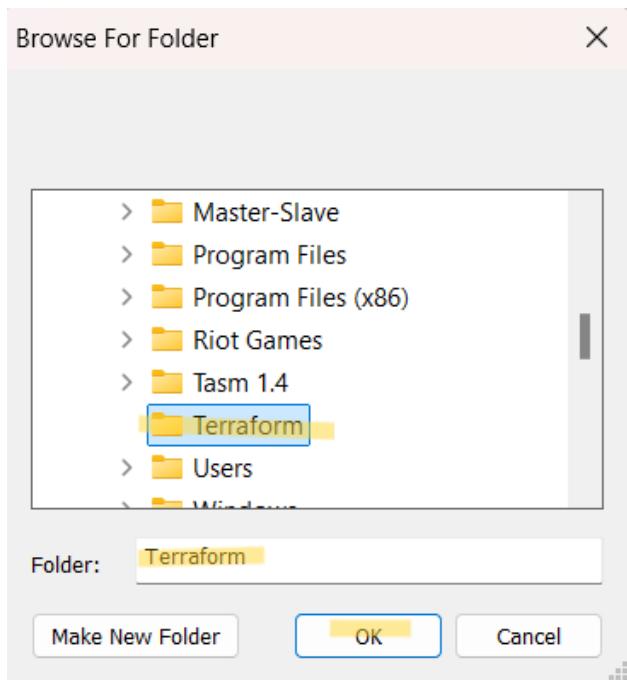


**Step 4:** This opens up an Environment Variables window. Here, under System Variables:

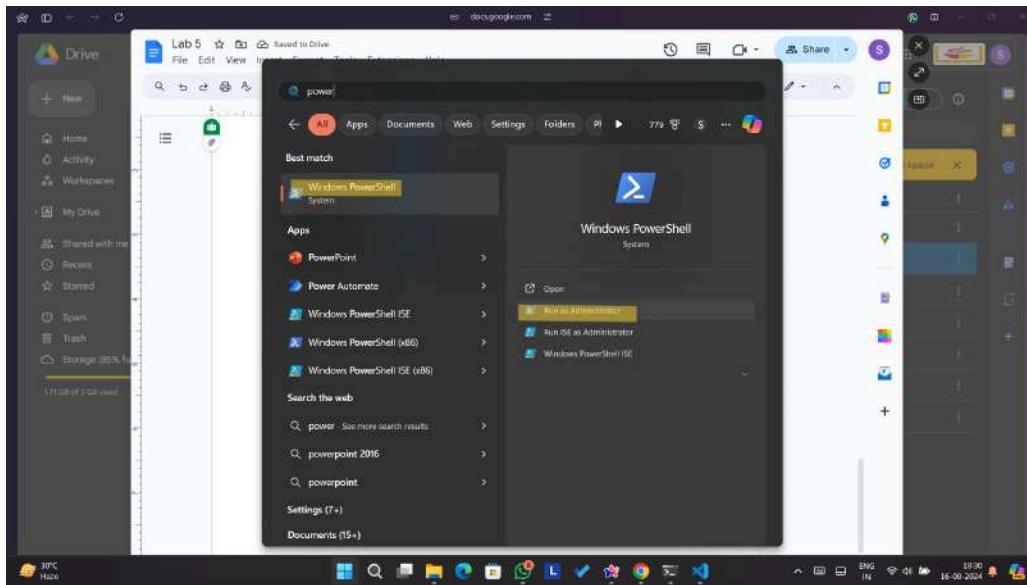
- 1) If a variable called Path does not exist, create the variable.
- 2) If Path variable exists, click on it; this will give you an option to edit the variable. Click on Edit.



**Step 5:** This gives you a screen to add a path to the existing paths. Click on Browse, search for the Terraform folder (C:\Terraform) and click on Ok. This adds the link and it can be seen in the list of paths. Click on Ok.



**Step 6:** Click on OK until you exit from all the Environment Variable screens. Now go to your search and search for Windows Powershell. Run this as an administrator.



**Step 7:** As the powershell opens, run the command `terraform`. This would give you all the main commands, other commands, and global options. If this is your output, then you have successfully installed Terraform on your system.

```
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
```

```
Global options (use these before the subcommand, if any):
  -chdir=DIR      Switch to a different working directory before executing the
                  given subcommand.
  -help           Show this help output, or the help for a specified subcommand.
  -version        An alias for the "version" subcommand.
```

**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.

**Step 1:** For this experiment, you need to install docker on your computer. Go to <https://www.docker.com/> and download the file according to the OS you have.

Open the file and start the installation.

Once installed, open your terminal and run ‘docker’ command.

If this is your output, then docker is installed successfully.

```
Microsoft Windows [Version 10.0.22631.3880]
(c) Microsoft Corporation. All rights reserved.

C:\Users\saira>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

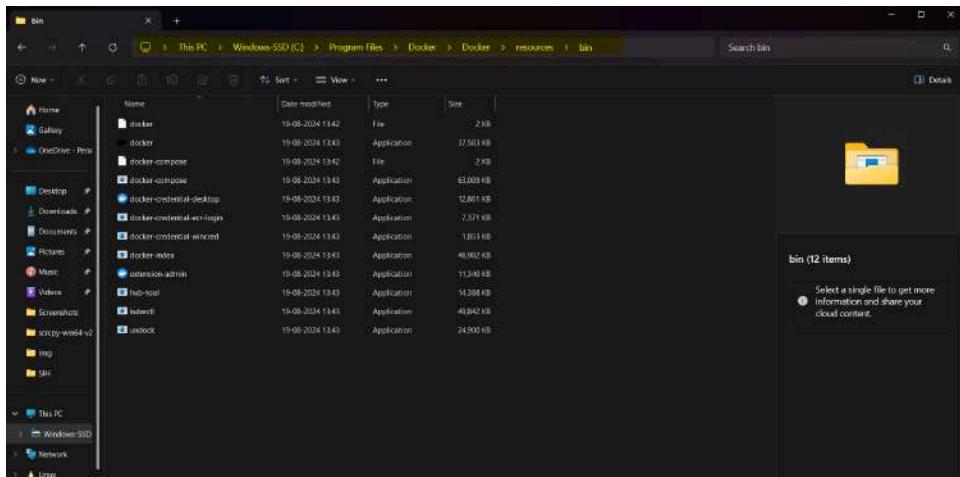
Management Commands:
  builder  Manage builds
  buildx*  Docker Buildx
  compose* Docker Compose
  container Manage containers
  context   Manage contexts
  debug*   Get a shell into any image or container
  desktop* Docker Desktop commands (Alpha)
  dev*    Docker Dev Environments
  extension* Manages 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
  history  Show the history of an image
  import   Import the contents from a tarball to create a filesystem image
  inspect  Return low-level information on Docker objects
  kill     Kill one or more running containers
  load    Load an image from a tar archive or STDIN
  logs    Fetch the logs of a container
  pause   Pause all processes within one or more containers
  port    List port mappings or a specific mapping for the container
  rename  Rename a container
  restart Restart one or more containers
  rm     Remove one or more containers
```

If you get an error like 'docker is not an internal or external command', you need to add the bin path of docker to your environment variables.

Go to File Explorer, and follow this path: C drive → Program Files → Docker → Docker → Resources → bin. Copy this path by clicking on the bar having the path and using shortcut CTRL + C.



Open 'Edit the System Environment Variables' on your system. Click on Environment Variables. Now, check for a 'Path' variable under System variables, if it exists, click on it, then click on edit. Else, click on New and add the variable 'Path'.

If the variable existed, click on Edit, then on New. This will give you a text box. Paste the path you copied here and click on ok until you close all the tabs.

Now run the docker command again and the output would appear.

Alternatively, you could also run 'docker --version' to check whether docker is started on terminal.

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

**Step 2:** Create a file called as 'docker.tf'. Open the file and put the following code.

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

```

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

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

# Create a container
resource "docker_container" "foo" {
  image = docker_image.ubuntu.image_id
  name  = "foo"
}

```

```

docker.tf
C: > Users > saira > OneDrive > Desktop > AdvDevOps > Terraform Scripts > docker > docker.tf
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 }
24

```

**Step 3:** Open the folder where the docker.tf is present on your terminal. Execute the command 'terraform init'. This will initialize terraform in the directory.

```

C:\Users\saira\OneDrive\Desktop\AdvDevOps\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...
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.

```

**Step 4:** Run the command ‘terraform plan’. This creates an execution plan and lets you overview changes that are going to happen in your infrastructure.

```
C:\Users\saira\OneDrive\Desktop\AdvDevOps\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
- destroy
~ modify

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)
  shn_size          = (known after apply)
  start             = true
  stdio_open         = false
  stop_signal        = (known after apply)
  stop_timeout       = (known after apply)
  tty               = false

  + healthcheck (known after apply)
  + labels (known after apply)
}

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

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

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

**Step 5:** Next, run command ‘terraform apply’. This command will carry out the changes that were to be made when ‘terrafrom plan’ command was executed.

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

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)
  shn_size          = (known after apply)
  start             = true
  stdio_open         = false
```

```

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

+ healthcheck (known after apply)
+ labels (known after apply)
}

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

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

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

Enter a value: yes

docker_image.ubuntu: Creating...
docker_image.ubuntu: Creation complete after 8s [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Creating...

Error: container exited immediately
with docker_container.foo,
on docker.tf line 20, in resource "docker_container" "foo":
```

The script that we are using is going to throw an error.

**Error: container exited immediately**

This is because the script used is way too small or took a lot less time to execute. To fix this, we add a line to the code. ‘Command = [“sleep”, “infinity”]’.

This line of code lets docker know to keep the program in sleep mode for an infinite amount of time so that the output can be observed rather than stopping after running immediately.

Now rerun the ‘terraform apply’ code. It will ask you to enter yes to execute it. Type yes. The code gets executed and the image is formed.

```

C:\Users\saira\OneDrive\Desktop\AdvDevOps\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:cdbfc74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a"
  + 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
+ stdio_open     = false
+ stop_signal    = "(Known after apply)"
+ stop_timeout   = "(Known after apply)"
+ tty            = false

+ healthcheck (known after apply)
+ labels (known after apply)

}

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

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

Enter a value: yes

docker_container.foo: Creating...
docker_container.foo: Creation complete after 1s [id=4baef3d214e59ae23d73cdef4193903a60d94f4d4b0f2ff8c2131891b7a652fca]

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

```

Using ‘docker images’ command, you can check the images that are present in docker.

‘docker images’ before ‘terraform apply’ is executed

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

‘docker images’ command after ‘terraform apply’ is executed

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

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

```

C:\Users\saira\OneDrive\Desktop\AdvDevOps\Terraform Scripts\docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542a137cf28a04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=4baef3d214e59ae23d73cdef4193903a60d94f4d4b0f2ff8c2131891b7a652fca]

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                = "4baef3d214e59" -> null
  - id                      = "4baef3d214e59ae23d73cdef4193903a60d94f4d4b0f2ff8c2131891b7a652fca" -> null
  - image                    = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
  - init                     = false -> null
  - ip_address               = "172.17.0.2" -> null
  - ip_prefix_length         = 16 -> null
  - ipc_mode                = "private" -> null
  - links                   = [] -> null
  - log_driver               = "json-file" -> null
  - log_opts                 = {} -> null
  - logs                     = false -> null
  - max_retry_count          = 0 -> null
}
```

```

  - ip_prefix_length      = 16
  - network_name          = "bridge"
    # (2 unchanged attributes hidden)
}
] => null
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@sha256:8a37d68f4f73ebf3d4efafbcf56379bf3728902a80386168008f94e34a9ab63ee" -> 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=4ba59ae23d73cdef4193903a60d94f4d4b0f2ff8c2131891b7a652fc]
docker_container.foo: Destruction complete after 0s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image.ubuntu: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.

```

Run the ‘docker images’ command again to check whether the image is destroyed or not.

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

Thus, we have created an image on docker using terraform and destroyed it.

### 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.

Prerequisites:

- 1) Docker

Run docker -v command.g

Use this command to check if docker is installed and running on your system.

```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab7> docker -v
Docker version 27.0.3, build 7d4bcd8
```

- 2) Install SonarQube image

Command: docker pull sonarqube

This command helps you to install an image of SonarQube that can be used on the local system without actually installing the SonarQube installer.

```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab7> docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
762bedf4b1b7: Pull complete
95f9bd9906fa: Pull complete
a32d681e6b99: Pull complete
aabdd0a18314: Pull complete
5161e45ecd8d: Pull complete
aeb0020dfa06: Pull complete
01548d361aea: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:bb444c58c1e04d8a147a3bb12af941c57e0100a5b21d10e599384d59bed36c86
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest

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

- 3) Keep jenkins installed on your system.

Experiment Steps:

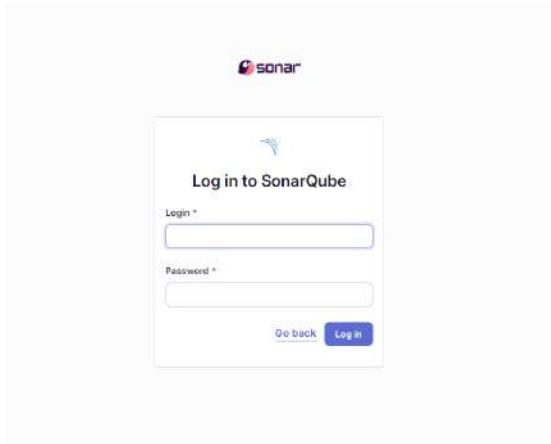
Step 1: Run SonarQube image

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

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

```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab7> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=t
36ff8a656bd28857ba9a28bf2bb0174099ae3232a9fc9ba2766d46f0c14d08a6
```

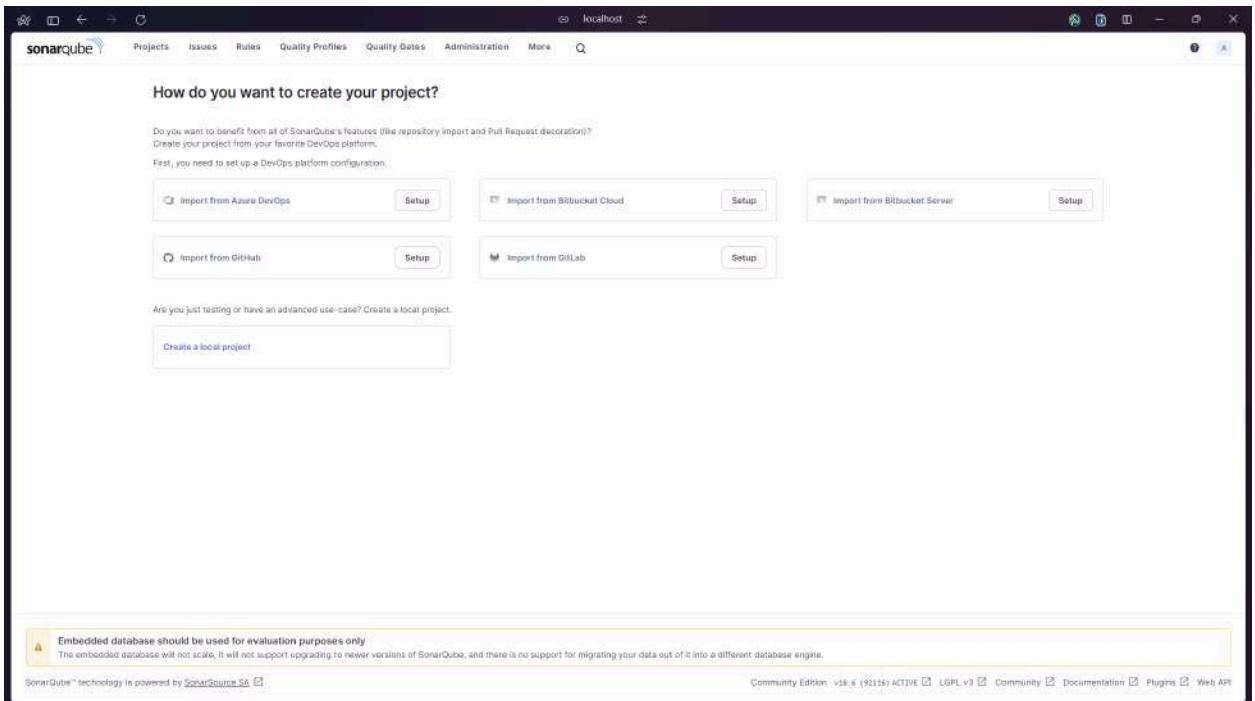
Step 2: Once the SonarQube image is started, you can go to <http://localhost:9000> to find the SonarQube that has started



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

A screenshot of a "Update your password" form. At the top, it says "Update your password". Below that is a yellow warning box with an exclamation mark and the text "This account should not use the default password." The form contains three input fields: "Enter a new password", "Old Password \*", and "New Password \*". There is also a field for "Confirm Password \*". At the bottom is a blue "Update" button.

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



Give the project a display name and project key

1 of 2

## Create a local project

Project display name \*

Project key \*

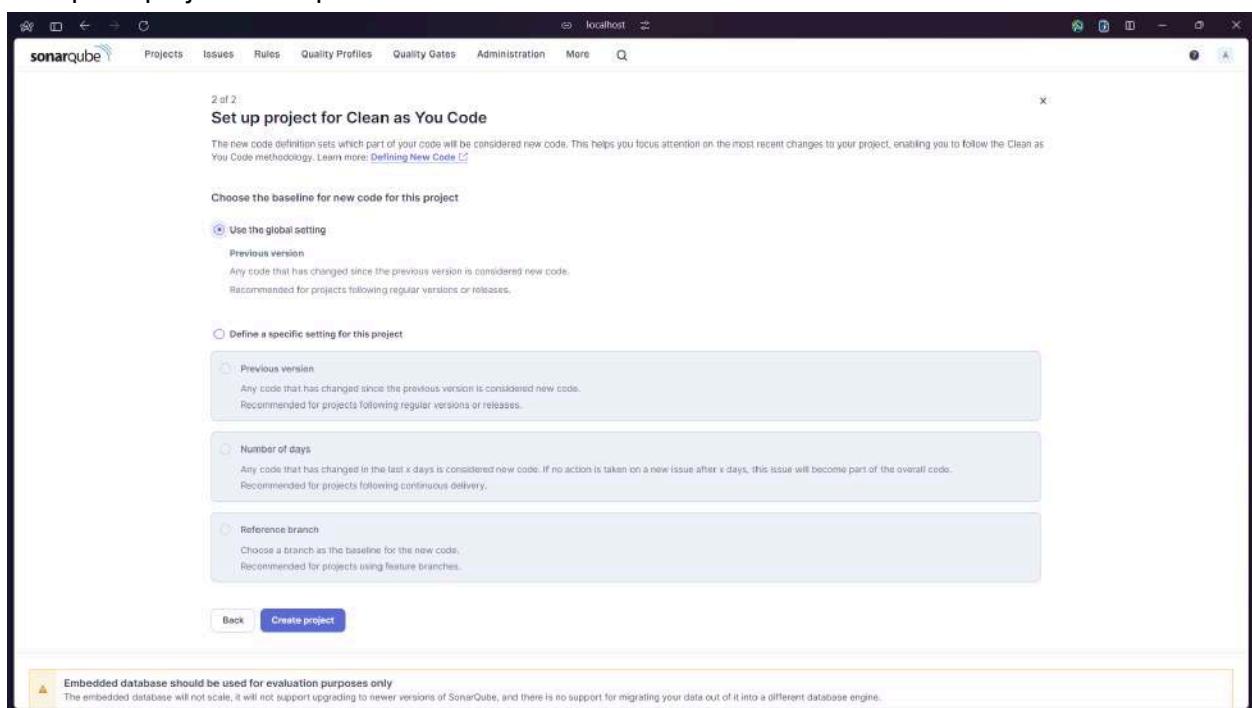
 

Main branch name \*

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

CancelNext

Set up the project as required and click on create.



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

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

[Back](#) [Create project](#)

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

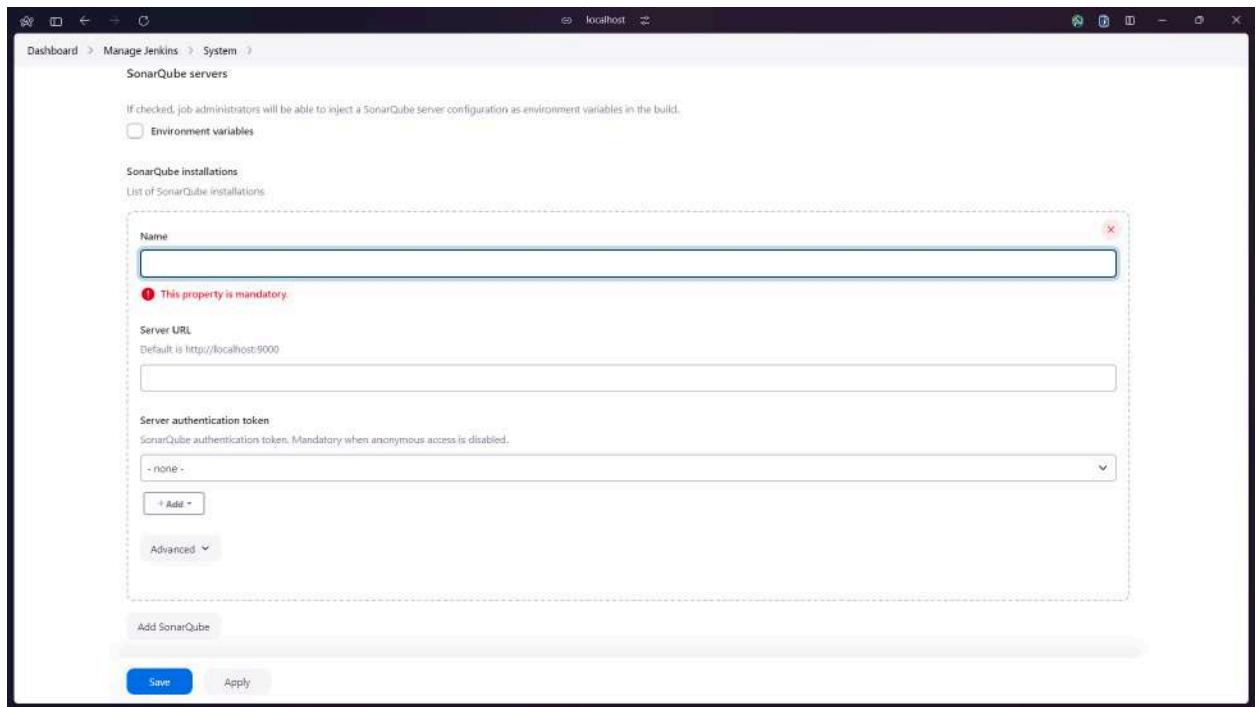
Step 5: Open Jenkins on whichever port it is installed. ([http://localhost:<port\\_number>](http://localhost:<port_number>)).

The screenshot shows the Jenkins dashboard. At the top, there's a search bar with 'Search (CTRL+K)' and a user dropdown for 'Sairam Konar'. Below the header is a 'Dashboard' section with a 'New Item' button and a list of recent projects: 'lmao', 'myNewJob', and 'myPipelineProject'. To the right of this is a table showing build statistics: Last Success, Last Failure, and Last Duration. Below the dashboard are sections for 'Build Queue' (empty) and 'Build Executor Status' (listing 1 idle, 2 idle, and 1 slave node). At the bottom right, there are links for 'REST API' and 'Jenkins 2.462.1'.

Step 6: Go to manage jenkins → Search for Sonarqube Scanner for Jenkins and install it.

The screenshot shows the Jenkins 'Manage Jenkins > Plugins' page. The left sidebar has tabs for 'Updates', 'Available plugins' (which is selected), 'Installed plugins', and 'Advanced settings'. A search bar at the top contains 'sonarqube'. The main area lists the 'SonarQube Scanner' plugin, version 2.17.2, under the 'Available plugins' tab. It has a brief description and was released 6 months and 29 days ago. Below it are other plugins: 'Sonar Gerrit' and 'SonarQube Generic Coverage'. A red banner at the bottom indicates an error: 'There were errors checking the update sites: UnknownHostException: updates.jenkins.io'.

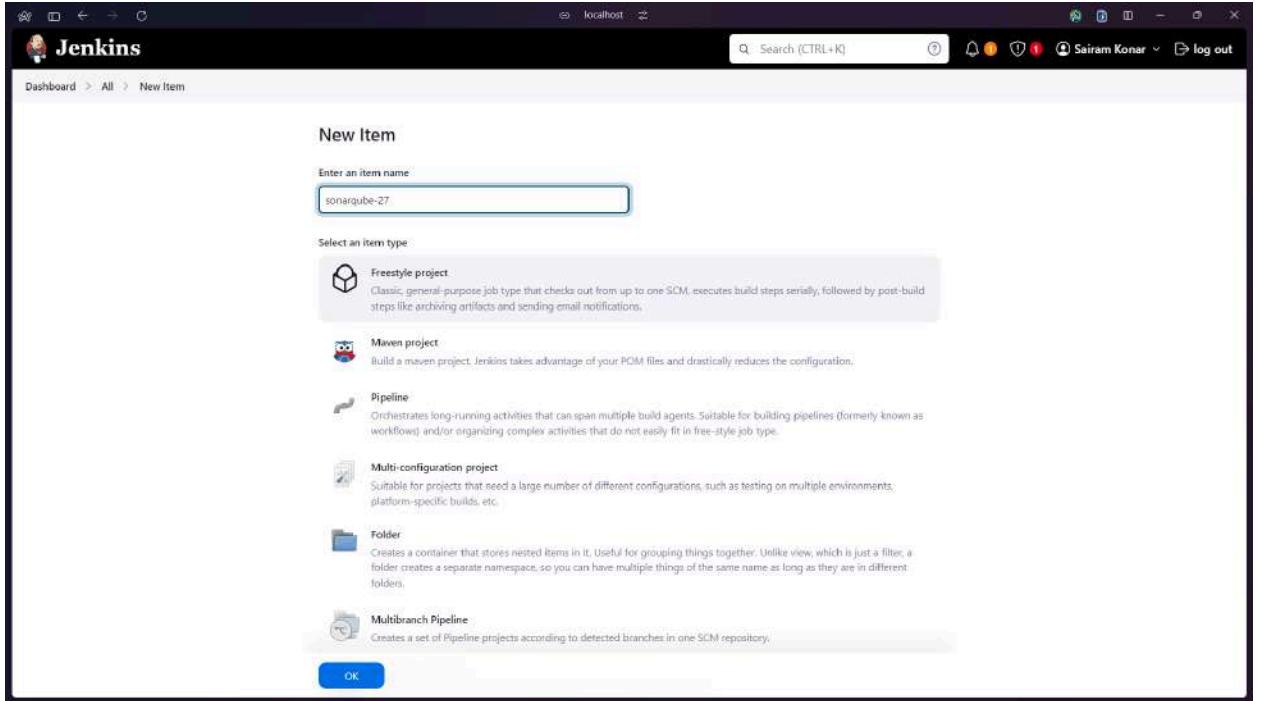
Step 7: Now, go to Manage Jenkins → System. Under Sonarqube servers, add a server. Add server authentication token if needed.



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



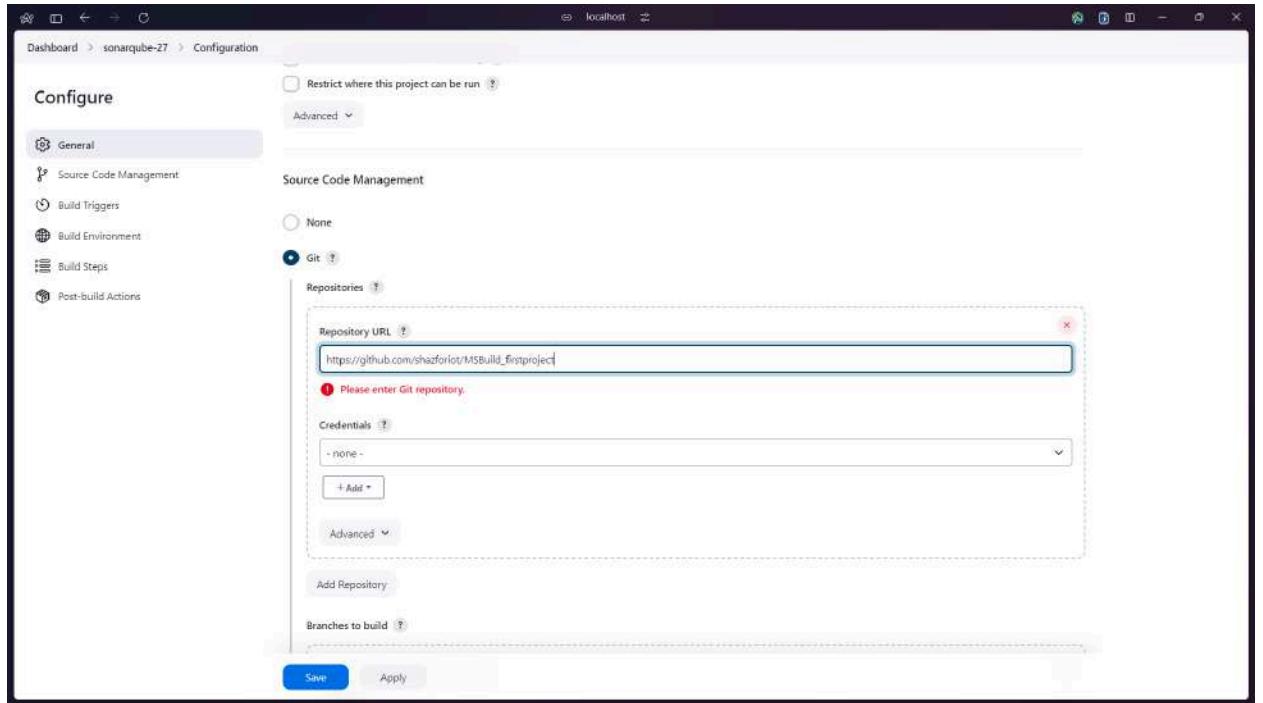
Step 9: After configuration, create a New Item → choose a freestyle project.



Step 10: Use this github repository in Source Code Management.

[https://github.com/shazforiot/MSBuild\\_firstproject](https://github.com/shazforiot/MSBuild_firstproject)

It is a sample hello-world project with no vulnerabilities.



Step 11: Under Build Steps, enter SonarQube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.

A screenshot of the Jenkins Build Step configuration for 'Execute SonarQube Scanner'. It includes fields for 'JDK', 'Path to project properties', 'Analysis properties' (containing 'sonar.projectKey=sonarqube', 'sonar.login=admin', 'sonar.password=123456', 'sonar.host.url=http://localhost:9000', and 'sonar.sources='), 'Additional arguments', and 'JVM Options'.

Step 12: Now, you need to grant the local user (here admin user) permissions to Execute the Analysis stage on SoanrQube. For this, go to [http://localhost:<port\\_number>/admin/permissions](http://localhost:<port_number>/admin/permissions) and check the 'Execute Analysis'

checkbox under Administrator.

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

4 of 4 shown

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

SonarQube™ technology is powered by SonarSource SA [\[link\]](#)

Community Edition: v18.6 (92116) ACTIVE [\[link\]](#) LGPL v3 [\[link\]](#) Community [\[link\]](#) Documentation [\[link\]](#) Plugins [\[link\]](#) Web API [\[link\]](#)

Step 13: Go back to jenkins. Go to the job you had just built and click on Build Now.

Dashboard > sonarqube-27 >

- Status
- </> Changes
- Workspace
- ▷ Build Now
- ⚙ Configure
- trash Delete Project
- SonarQube
- pencil Rename

**sonarqube-27**

 SonarQube

**Permalinks**

- Last build (#1), 4 hr 18 min ago
- Last stable build (#1), 4 hr 18 min ago
- Last successful build (#1), 4 hr 18 min ago
- Last completed build (#1), 4 hr 18 min ago

Check the console Output

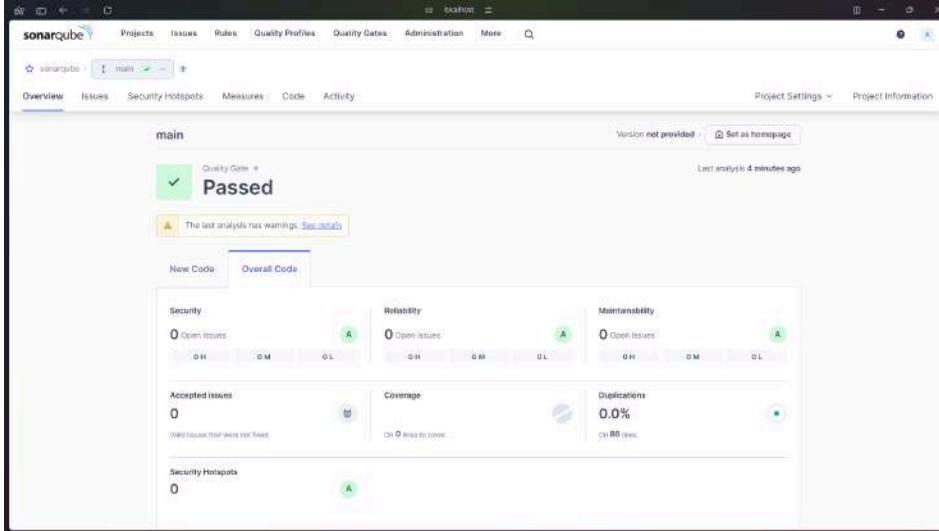
```

Started by user Sairam Konar
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\jenkins\workspace\sonarqube-27
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\jenkins\workspace\sonarqube-27\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/sharfuriot/MSBuild_FirstProject # timeout=10
Fetching upstream changes from https://github.com/sharfuriot/MSBuild_FirstProject
> git.exe --version # timeout=10
> git.exe rev-parse --tag --force --progress -- https://github.com/sharfuriot/MSBuild_FirstProject +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse --depth=1000caee6d6fee7b49adf # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcacee6d6fee7b49adf (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f f2bc042c04c6e72427c380bcacee6d6fee7b49adf # timeout=10
Commit message: "updated"
First time build. Skipping changelog.
[sonarqube-27] $ C:\ProgramData\Jenkins\jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube_lab\bin\sonar-scanner.bat -Dsonar.projectKey=sonarqube -Dsonar.login=admin -Dsonar.host.url=http://localhost:9000 -Dsonar.sources=. -Dsonar.password=123456 -Dsonar.projectBaseDir=C:\ProgramData\Jenkins\jenkins\workspace\sonarqube-27
09:24:56.227 INFO Scanner configuration file: C:\ProgramData\Jenkins\jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube_lab\bin..\conf\sonar-scanner.properties
09:24:56.231 INFO Project root configuration file: NONE
analyzer has been disabled for C:\ProgramData\Jenkins\jenkins\workspace\sonarqube-27

09:25:27.920 WARN Your project contains C# files which cannot be analyzed with the scanner you are using. To analyze C# or VB.NET, you must use the SonarScanner for .NET 5.x or higher, see https://redirect.sonarsource.com/doc/install-configure-scanner-asbuild.html
09:25:27.931 INFO Sensor C# [csharp] (done) | time=1ms
09:25:27.931 INFO Sensor Analysis Warnings import [csharp]
09:25:27.933 INFO Sensor Analysis Warnings import [csharp] (done) | time=2ms
09:25:27.934 INFO Sensor C# File Caching Sensor [csharp]
09:25:27.935 WARN Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir' property.
09:25:27.935 INFO Sensor C# File Caching Sensor [csharp] (done) | time=1ms
09:25:27.935 INFO Sensor Zero Coverage Sensor
09:25:27.952 INFO Sensor Zero Coverage Sensor (done) | time=18ms
09:25:27.956 INFO SCM Publisher SCM provider for this project is: git
09:25:27.958 INFO SCM Publisher 4 source files to be analyzed
09:25:28.360 INFO SCM Publisher 4/A source files have been analyzed (done) | time=501ms
09:25:28.363 INFO CPD Executor Calculating CPD for 0 files
09:25:28.363 INFO CPD Executor CPD calculation finished (done) | time=0es
09:25:28.366 INFO SCM revision ID: f2bc042c04c6e72427c380bcacee6d6fee7b49adf
09:25:28.697 INFO Analysis report generated in 119ms, dir size=201.0 kB
09:25:28.754 INFO Analysis report compressed in 41ms, zip size=22.4 kB
09:25:28.977 INFO Analysis report uploaded in 22ms
09:25:28.988 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube
09:25:28.988 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
09:25:28.988 INFO More about the report processing at http://localhost:9000/api/ca/task?id=288a1357-211b-46ce-aaab-ab463fb70929
09:25:28.996 INFO Analysis total time: 22.822 s
09:25:28.997 INFO SonarScanner Engine completed successfully
09:25:29.013 INFO EXECUTION SUCCESS
09:25:29.031 INFO Total time: 32.809s
Finished: SUCCESS

```

Step 14: Once the build is complete, go back to SonarQube and check the project linked.



## Conclusion:

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

## **Experiment 8**

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

### Prerequisites:

#### Step 1: Download sonar scanner

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

The screenshot shows the 'SonarScanner CLI' page from the SonarQube documentation. The left sidebar contains navigation links for 'Homepage', 'Scanners', 'SonarScanner CLI', and other project analysis setup options. The main content area displays version 6.1 details, including supported distributions (macOS and Linux AArch64), download links for various platforms, and release notes. A note states that SonarScanners run on checked-out code. Below this is a 'Configuring your project' section with instructions to create a configuration file named 'sonar-project.properties'. The right sidebar lists related topics like 'Configuring your project', 'Running SonarScanner CLI from the zip file', and 'Scanning C, C++, or Objective-C projects'.

Extract the downloaded zip file in a folder.

- 1) Docker

Run docker -v command.

```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab7> docker -v
Docker version 27.0.3, build 7d4bcd8
```

- 2) Install sonarqube image

Command: docker pull sonarqube

```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab7> docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
762bedf4b1b7: Pull complete
95f9bd9906fa: Pull complete
a32d681e6b99: Pull complete
aabdd0a18314: Pull complete
5161e45ecd8d: Pull complete
aeb0020dfa06: Pull complete
01548d361aea: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:bb444c58c1e04d8a147a3bb12af941c57e0100a5b21d10e599384d59bed36c86
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest

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

- 3) Keep Jenkins installed on your system.

### Experiment Steps:

Step 1) Run SonarQube image

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

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

```
PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab7> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
36ff8a656bd28857ba9a28bf2bb0174099ae3232a9fc9ba2766d46f0c14d08a6
```

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



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

**Update your password**

⚠️ This account should not use the default password.

**Enter a new password**  
All fields marked with \* are required

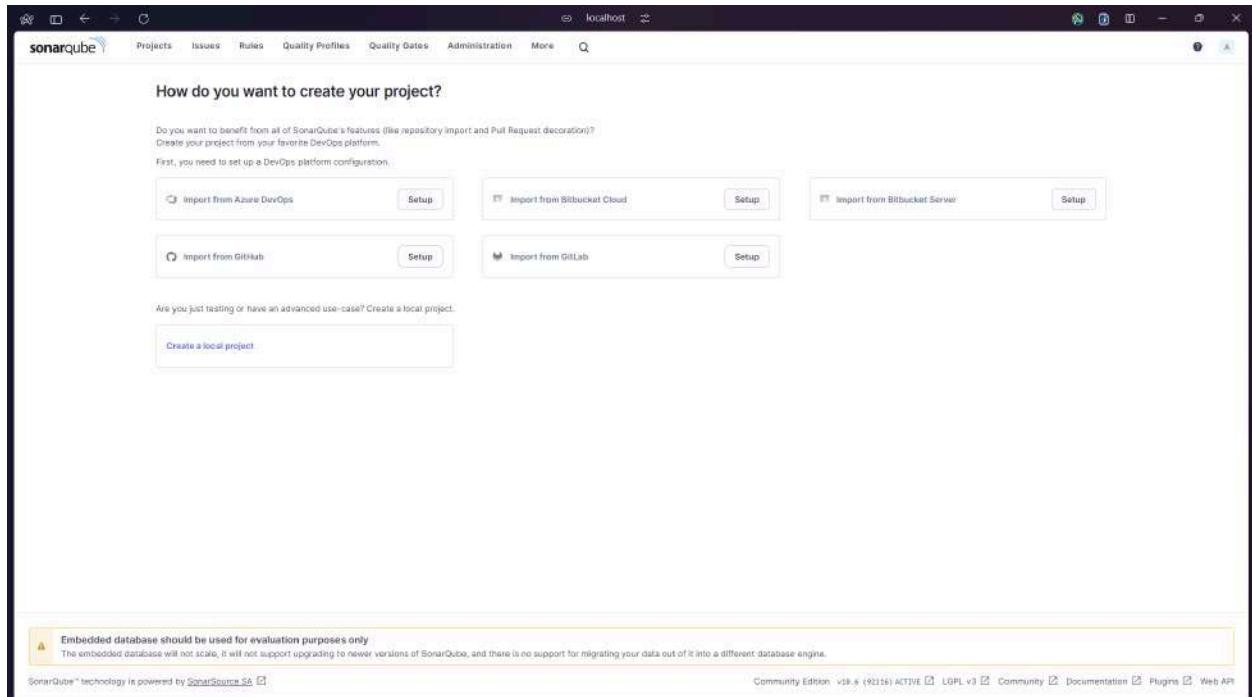
**Old Password \***

**New Password \***

**Confirm Password \***

**Update**

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



Give the project a display name and project key

1 of 2

## Create a local project

Project display name \*

Project key \*

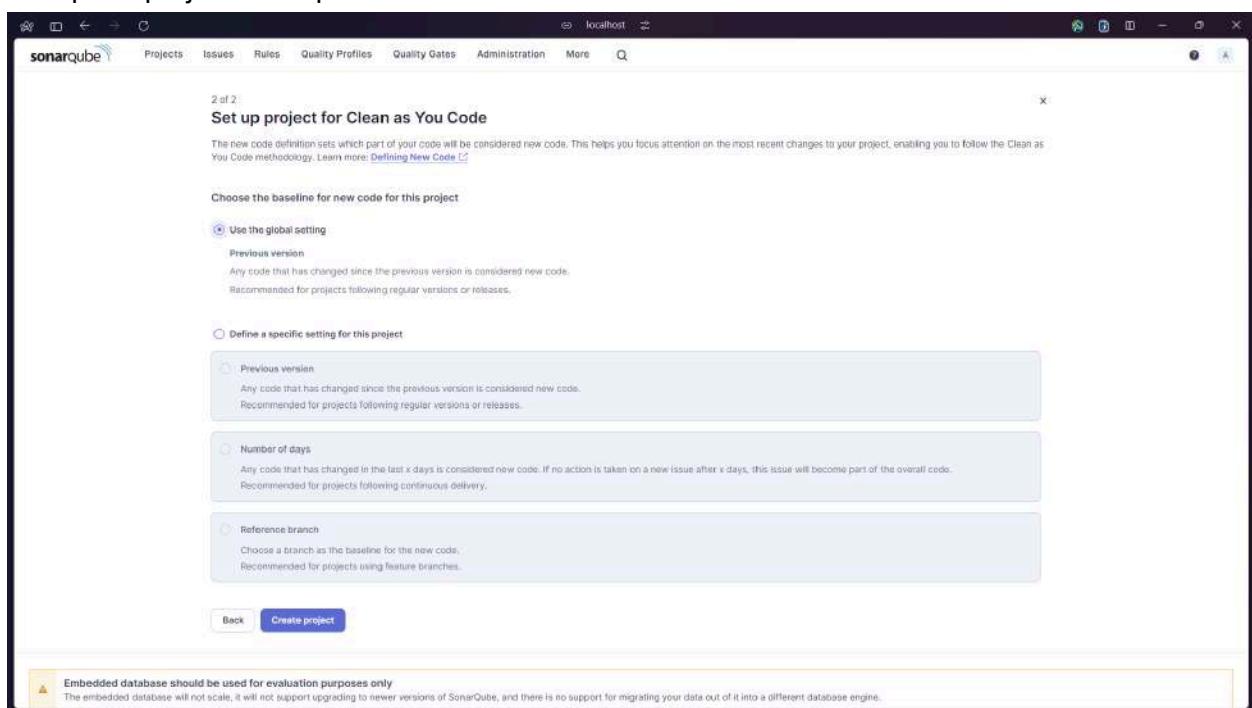
 

Main branch name \*

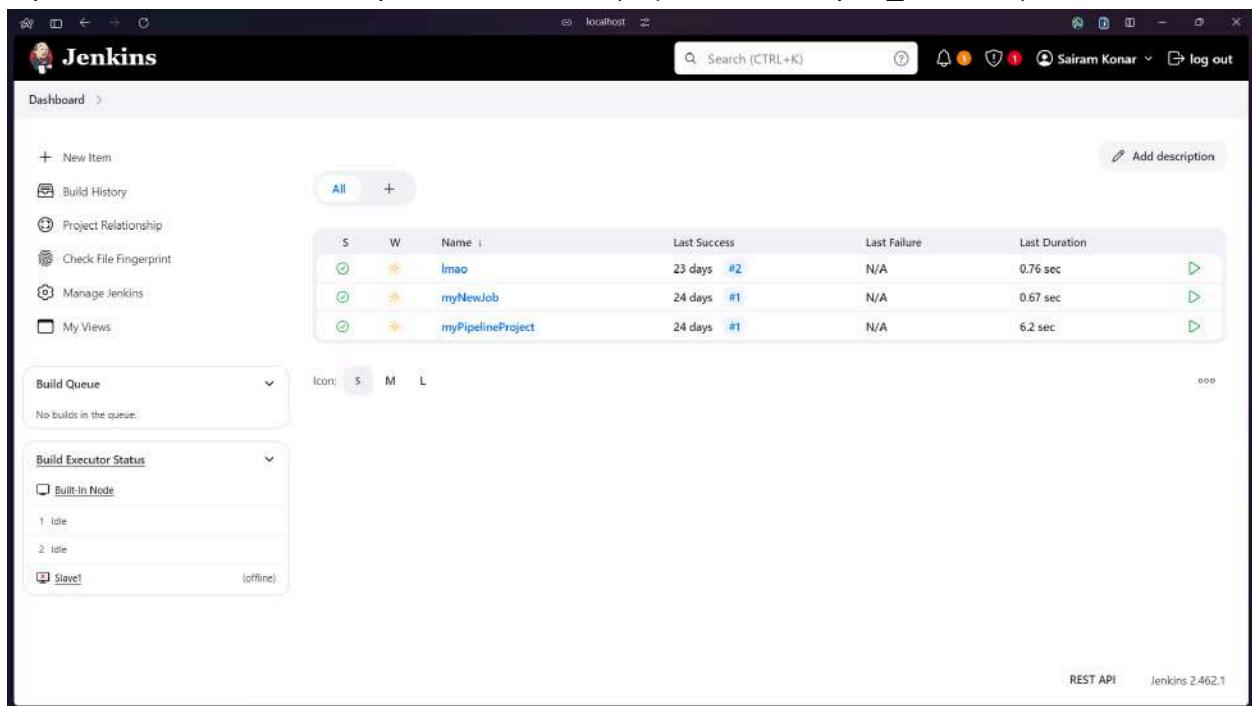
The name of your project's default branch [Learn More](#) 

CancelNext

Set up the project as required and click on create.

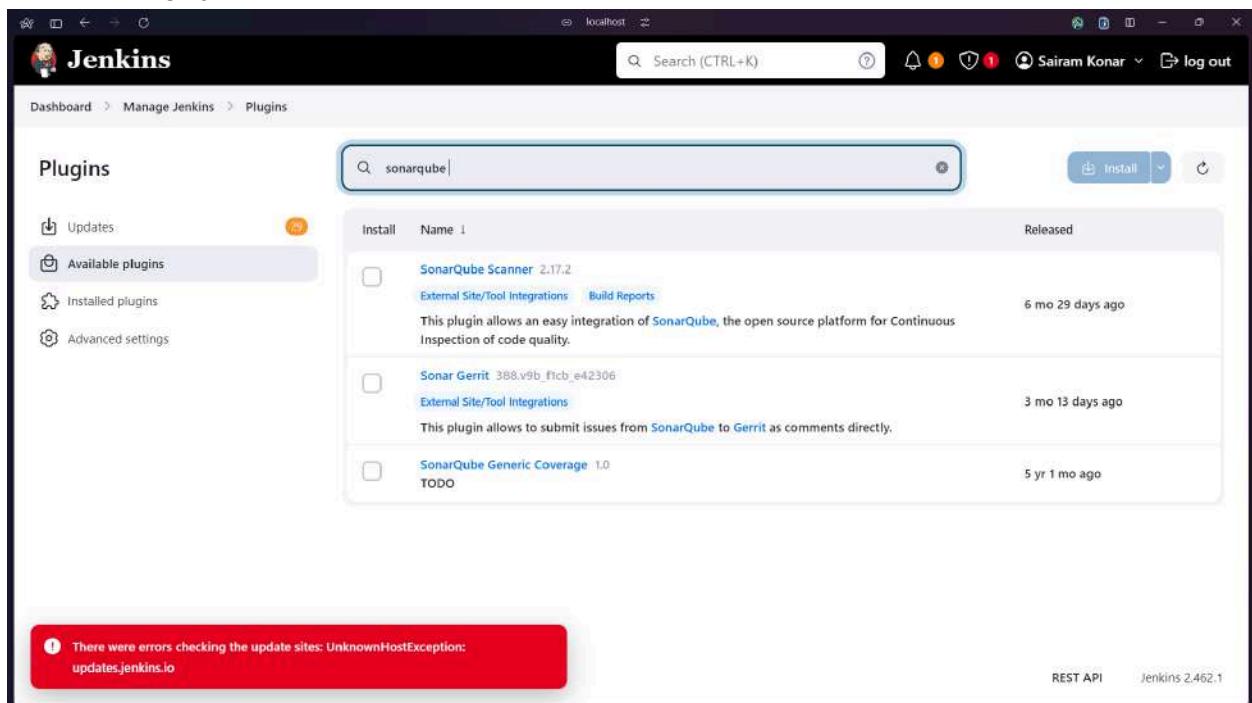


Step 5) Open Jenkins on whichever port it is installed. ([http://localhost:<port\\_number>](http://localhost:<port_number>)).



The screenshot shows the Jenkins dashboard. At the top, there's a search bar with 'Search (CTRL+K)' and a user dropdown for 'Sairam Konar'. Below the header is a 'Dashboard' section with a 'New Item' button and links for 'Build History', 'Project Relationship', 'Check File Fingerprint', 'Manage Jenkins', and 'My Views'. A table lists three projects: 'lmao' (last success 23 days ago), 'myNewJob' (last success 24 days ago), and 'myPipelineProject' (last success 24 days ago). Below the dashboard are sections for 'Build Queue' (empty) and 'Build Executor Status' (one idle node and one slave node labeled '(offline)'). At the bottom right, there are 'REST API' and 'Jenkins 2.462.1' links.

Step 6) Go to manage jenkins → Search for Sonarqube Scanner for Jenkins and install it.

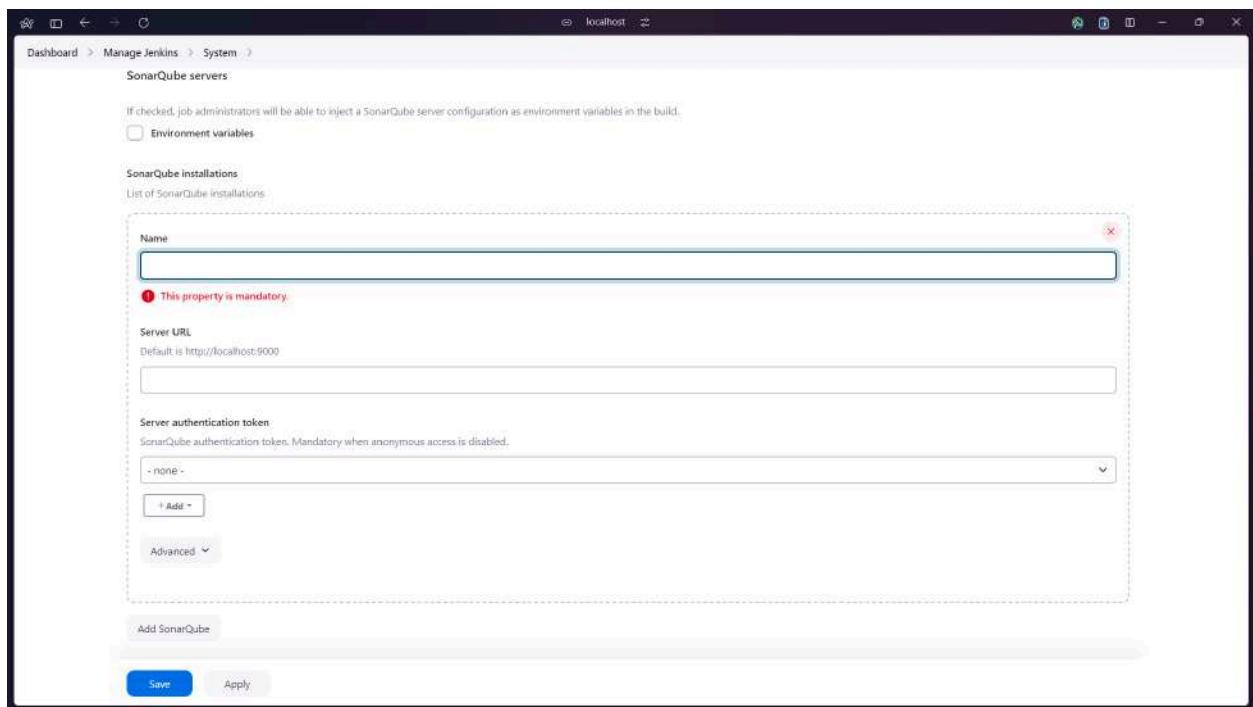


The screenshot shows the Jenkins 'Manage Jenkins > Plugins' page. The left sidebar has tabs for 'Updates', 'Available plugins' (which is selected), 'Installed plugins', and 'Advanced settings'. A search bar at the top contains 'sonarqube'. The main area lists three plugins under the 'Install' tab:

Name	Released
SonarQube Scanner 2.17.2	6 mo 29 days ago
Sonar Gerrit 388.v9_b_f1cb_e42306	3 mo 13 days ago
SonarQube Generic Coverage 1.0	5 yr 1 mo ago

A red banner at the bottom left indicates an error: 'There were errors checking the update sites: UnknownHostException: updates.jenkins.io'. At the bottom right are 'REST API' and 'Jenkins 2.462.1' links.

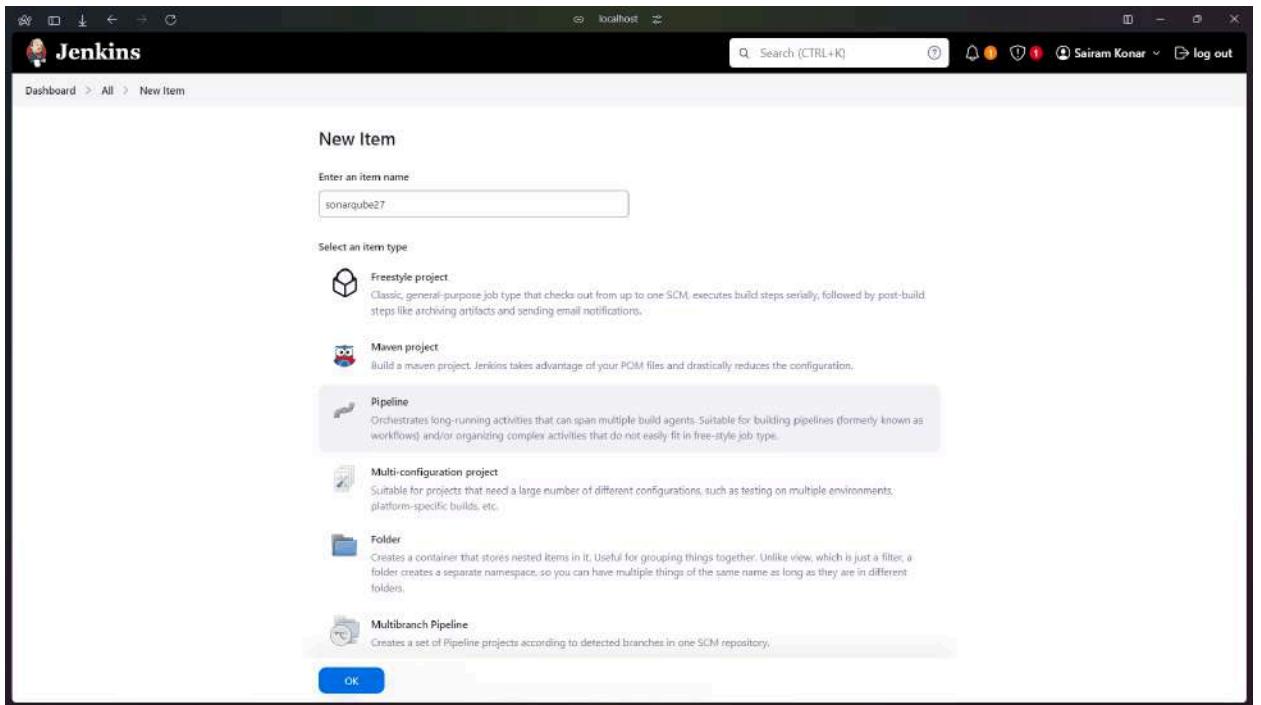
Step 7) Now, go to Manage Jenkins → System. Under Sonarqube servers, add a server. Add server authentication token if needed.



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



Step 9) After configuration, create a New Item → choose a pipeline project.



Step 10) Under Pipeline script, enter the following:

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

    stage('SonarQube analysis') {
        withSonarQubeEnv('sonarqube lab') {
            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/
            """
        }
    }
}
```

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

Definition

Pipeline script

```

Script ?  

1 * node {  

2 *   stage('Cloning the GitHub Repo') {  

3 *     git 'https://github.com/shazforiot/GOL.git'  

4 *   }  

5 *  

6 *   stage('SonarQube analysis') {  

7 *     withSonarQubeEnv('sonarqube lab') {  

8 *       bat ""  

9 *         C:\\\\Users\\\\saira\\\\Downloads\\\\sonar-scanner-cli-6.1.0.4477-windows-x64\\\\sonar-scanner-6.1.0.4477-windows-x64\\\\bin\\\\sonar-scanner.bat ^  

10 *           -D sonar.login=admin ^  

11 *           -D sonar.password=123456 ^  

12 *           -D sonar.projectKey=sonarqube27 ^  

13 *           -D sonar.exclusions=vendor/**,resources/**,**/*.java ^  

14 *           -D sonar.host.url=http://localhost:9000/  

15 *       }  

16 *     }  

17 *   }  

18 * }

```

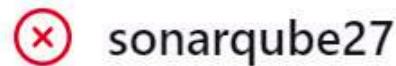
try sample Pipeline... ▾

Use Groovy Sandbox ?

Pipeline Syntax

Save Apply

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



## Stage View



## Permalinks

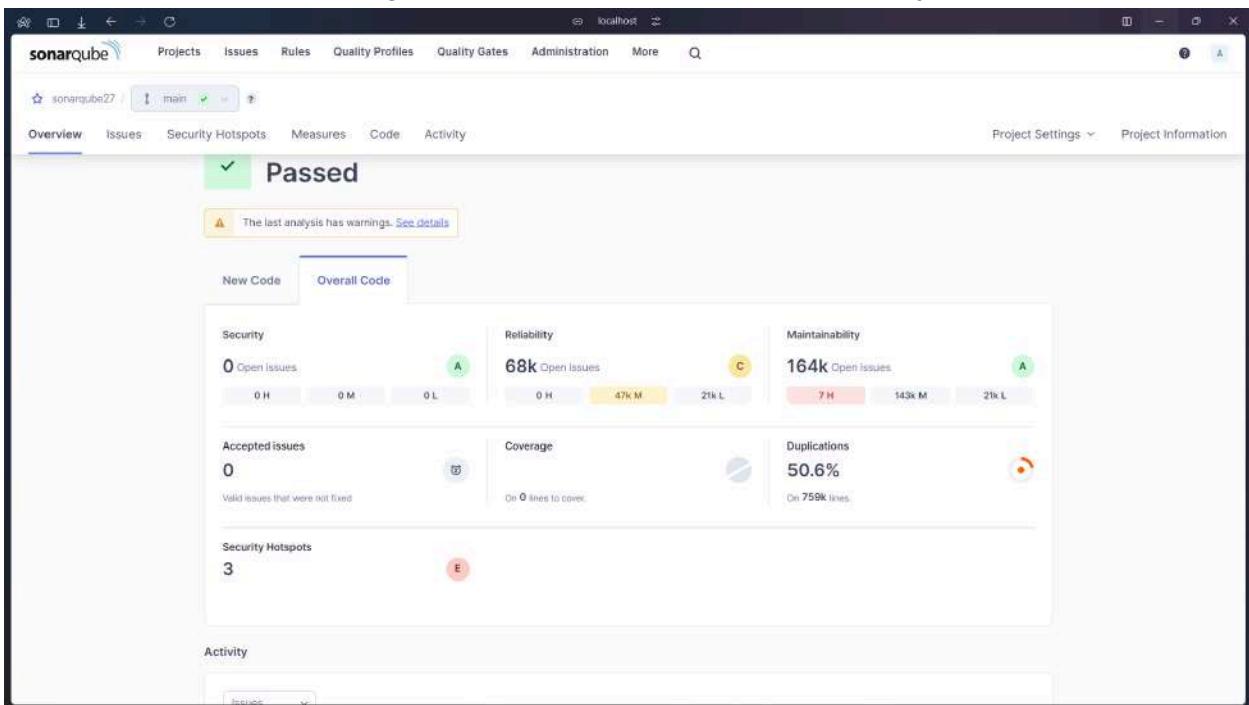
Check the console output once

```

first 100 references.
09:36:08.013 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/action/Cut.html for block at line 75. Keep only the first 100 references.
09:36:08.029 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/jdbc/config/package-summary.html for block at line 39. Keep only the first 100 references.
09:36:08.029 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/jdbc/config/package-summary.html for block at line 40. Keep only the first 100 references.
09:36:08.054 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/util/accesslog/Generator.html for block at line 40. Keep only the first 100 references.
09:36:08.054 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/util/accesslog/Generator.html for block at line 41. Keep only the first 100 references.
09:36:08.055 INFO CPD Executor CPD calculation finished (done) | time=163405ms
09:36:08.069 INFO SCM revision ID 'ba799ba7e1b576f04a612322b0412c5e6e1e5e4'
09:38:20.750 INFO Analysis report generated in 4390ms, dir size=127.2 MB
09:38:45.857 INFO Analysis report compressed in 25089ms, zip size=29.6 MB
09:38:46.532 INFO Analysis report uploaded in 675ms
09:38:46.533 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube27
09:38:46.533 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
09:38:46.533 INFO More about the report processing at http://localhost:9000/api/ce/task?id=46576333-chde-4277-89d7-471ee554de32
09:39:00.038 INFO Analysis total time: 15:24.256 s
09:39:00.041 INFO SonarScanner Engine completed successfully
09:39:00.010 INFO EXECUTION SUCCESS
09:39:00.811 INFO Total time: 15:29.301s
[Pipeline] }
[Pipeline] // withSonarQubeEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```

Step 12) Once the build is complete, go back to SonarQube and check the project linked.



Under different tabs, check all the issues with the code.

- Code Problems
- Consistency

gameoflife-core/build/reports/tests/all-tests.html

- Insert a <!DOCTYPE> declaration to before this <html> tag. Consistency  
Reliability ●  
Open ▾ Not assigned ▾ L1 • 5min effort • 4 years ago • ⚙ Bug • ⚙ Major
- Remove this deprecated "width" attribute. Consistency  
Maintainability ●  
HTML5 obsolete HTML5 obsolete  
Open ▾ Not assigned ▾ L9 • 5min effort • 4 years ago • ⚙ Code Smell • ⚙ Major
- Remove this deprecated "align" attribute. Consistency  
Maintainability ●  
HTML5 obsolete HTML5 obsolete  
Open ▾ Not assigned ▾ L11 • 5min effort • 4 years ago • ⚙ Code Smell • ⚙ Major
- Remove this deprecated "align" attribute. Consistency  
Maintainability ●  
HTML5 obsolete HTML5 obsolete  
Open ▾ Not assigned ▾ L11 • 5min effort • 4 years ago • ⚙ Code Smell • ⚙ Major

for evaluation purposes only  
It's not supported upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

## ● Intentionality

gameoflife-acceptance-tests/Dockerfile

- Use a specific version tag for the image. Intentionality  
Maintainability ●  
No tags No tags  
Open ▾ Not assigned ▾ L1 • 5min effort • 4 years ago • ⚙ Code Smell • ⚙ Major
- Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. Intentionality  
Maintainability ●  
No tags No tags  
Open ▾ Not assigned ▾ L12 • 5min effort • 4 years ago • ⚙ Code Smell • ⚙ Major
- Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. Intentionality  
Maintainability ●  
No tags No tags  
Open ▾ Not assigned ▾ L12 • 5min effort • 4 years ago • ⚙ Code Smell • ⚙ Major
- Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. Intentionality  
Maintainability ●  
No tags No tags  
Open ▾ Not assigned ▾ L12 • 5min effort • 4 years ago • ⚙ Code Smell • ⚙ Major

or evaluation purposes only  
It's not supported upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

- Bugs

The screenshot shows the SonarQube interface for reviewing bugs. At the top, there are navigation links for 'Bulk Change', 'Select issues', 'Navigate to issue', and status indicators for '13,619 issues' and '56d effort'. Below this, two sections of bugs are listed:

- gameoflife-core/build/reports/tests/html-tests.html**
  - Add "lang" and/or "xml:lang" attributes to this "<html>" element. Intentionality accessibility wcag2-a
  - Add "<th>" headers to this "<table>". Intentionality accessibility wcag2-a
- gameoflife-core/build/reports/tests/allclasses-frame.html**
  - Add "lang" and/or "xml:lang" attributes to this "<html>" element. Intentionality accessibility wcag2-a
  - Add "<th>" headers to this "<table>". Intentionality accessibility wcag2-a

A note at the bottom states: 'for evaluation purposes only' and 'it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.'

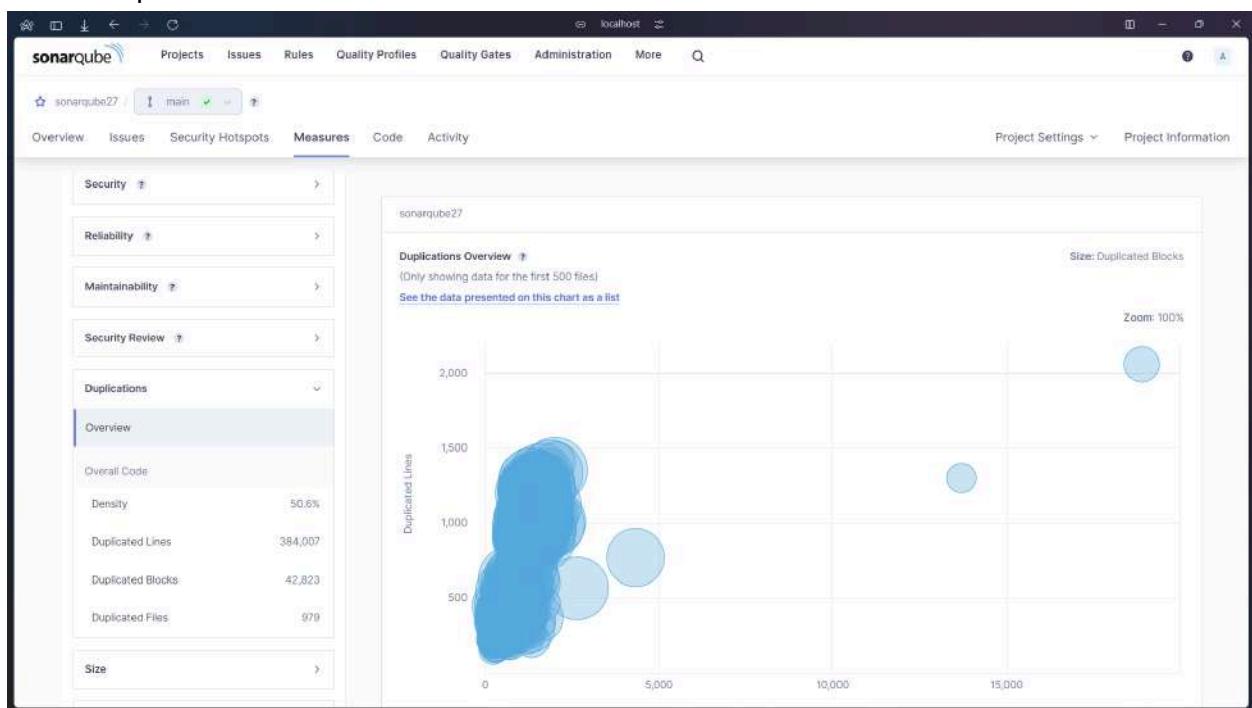
- Code Smells

The screenshot shows the SonarQube interface for reviewing code smells. At the top, there are navigation links for 'Bulk Change', 'Select issues', 'Navigate to issue', and status indicators for '268 issues' and '2d 5h effort'. Below this, three sections of code smells are listed:

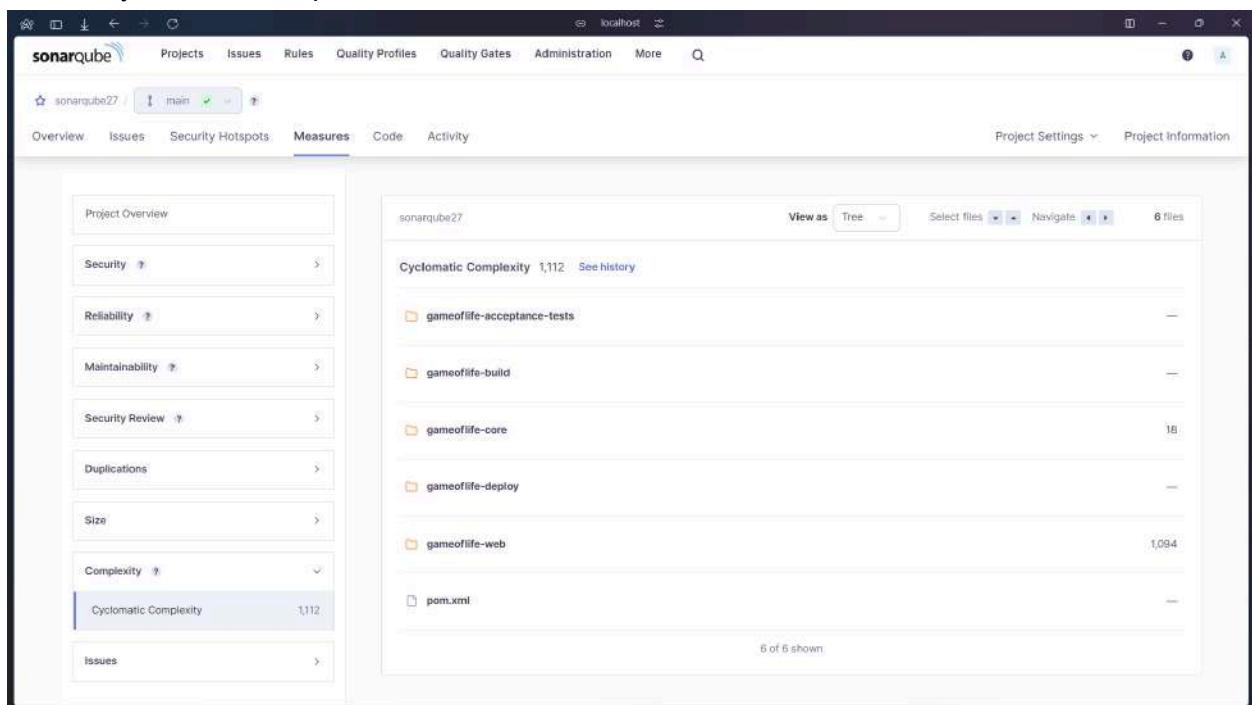
- gameoflife-acceptance-tests/Dockerfile**
  - Use a specific version tag for the image. Intentionality No tags
  - Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. Intentionality No tags
  - Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. Intentionality No tags
  - Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. Intentionality No tags

A note at the bottom states: 'for evaluation purposes only' and 'it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.'

- Duplications



- Cyclomatic Complexities

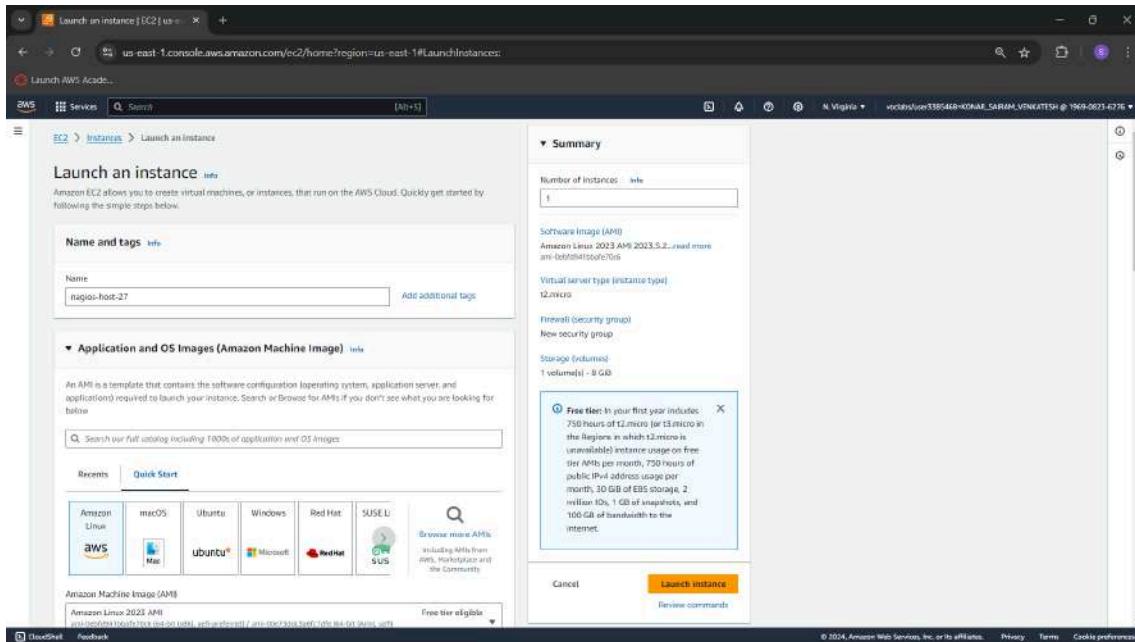


**Conclusion:** In this experiment, we have learned how to perform static analysis of a code using Jenkins CI/CD Pipeline with SonarQune analysis. A pipeline project is to be created which is given a pipeline script. This script contains all the information needed for the project to run the SonarQube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

## Experiment 9

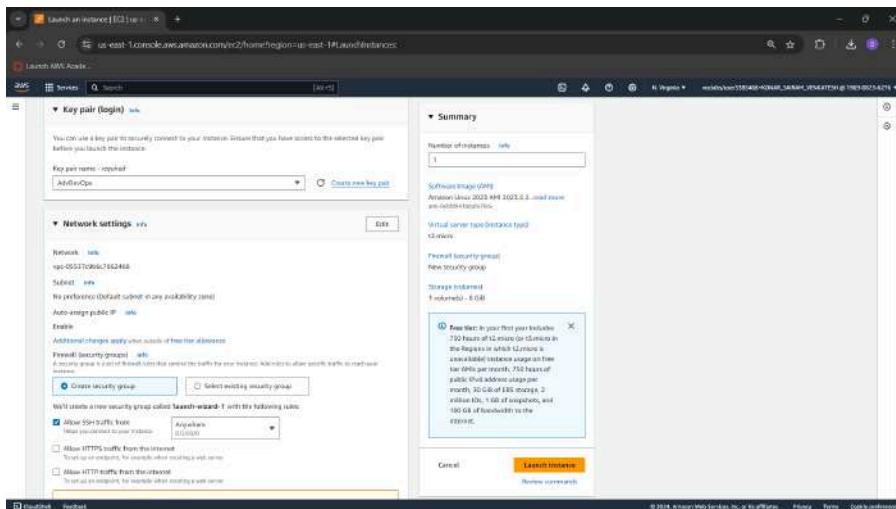
**Aim:** To Understand Continuous monitoring and Installation and configuration of Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine.

Step 1: Login to your AWS account. Search for EC2 on services. Open the interface and click on Create Instance.



Select The OS Image as Amazon Linux.

Step 2: If you do not have a private key created or a .pem file created, click on create a key pair. Else select the key pair that you had created before. (Make sure you know where the .pem file for that key is present on your system)



AWS will create a security group for this instance. Keep the name of that instance saved.  
 Step 3: After creating the instance, click on Security Groups from the left side pane. Find the security group that was created for your instance. Click on the instance ID for that group.

The screenshot shows the AWS Management Console with the URL [us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#SecurityGroups](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#SecurityGroups). The left sidebar is collapsed. The main content area displays a table titled "Security Groups (4) info". The table has columns for Name, Security group ID, Security group name, VPC ID, Description, and Owner. The data is as follows:

Name	Security group ID	Security group name	VPC ID	Description	Owner
-	sg-02ff30815ef37a6f2	launch-wizard-1	vpc-05537c9b6c786246b	launch-wizard-1 created 2024-09-23T...	196908236276
-	sg-0eb010ebcd236d5a	default	vpc-05537c9b6c786246b	default VPC security group	196908236276
-	sg-002266c95eb9a4c13	master	vpc-05537c9b6c786246b	master node group	196908236276
-	sg-027f0c1b77adear23	node	vpc-05537c9b6c786246b	rules for node	196908236276

Here, click on Edit Inbound Rules.

The screenshot shows the AWS Management Console with the URL [us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#SecurityGroup:groupId=sg-02ff30815ef37a6f2](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#SecurityGroup:groupId=sg-02ff30815ef37a6f2). The left sidebar is collapsed. The main content area shows the "Details" section for the security group "sg-02ff30815ef37a6f2 - launch-wizard-1". It includes fields for Security group name, Security group ID, Description, and VPC ID. Below this, the "Inbound rules" tab is selected, showing a table with one rule:

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
-	sgr-06fa5340b1e8b301d	IPv4	SSH	TCP	22	0.0.0.0/0	

Now, click on add rules, and add teh rules for the following protocols:

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

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

Step 4: Now come back to the instances screen. Click on the instance ID of your instance. Then click on Connect.

Click on SSH client. Copy the example command.

Step 5: Now, we have to connect our local OS terminal to the instance using SSH. For this, Open terminal wher the private key file is located (.pem) Paste the copied SSH command and run it.

```
C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab 9>ssh -i "AdvDevOps.pem" ec2-user@ec2-3-89-111-169.compute-1.amazonaws.com
  _#_
 /_###_      Amazon Linux 2023
 \###|
  #/ __> https://aws.amazon.com/linux/amazon-linux-2023
   V-__/
    __/ /
   __/_/ /
  __/_/ /
 /m/_/ /
 [ec2-user@ip-172-31-90-224 ~]$
```

Step 6: Now we start working on this terminal. First run the command  
sudo yum update

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

```
[ec2-user@ip-172-31-83-157 ~]$ sudo yum update
Last metadata expiration check: 0:04:50 ago on Sat Sep 28 03:46:46 2024.
Dependencies resolved.
Nothing to do.
Complete!
```

Step 7: We are going to install an Apache server and a PHP on this instance. For that, run this command.

sudo yum install httpd php

```
[ec2-user@ip-172-31-83-157 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:04:50 ago on Sat Sep 28 03:46:46 2024.
Dependencies resolved.
=====
Package           Architecture Version       Repository  Size
=====
Installing:
httpd            x86_64      2.4.62-1.amzn2023.0.2      amazonlinux 48 k
php8.3           x86_64      8.3.10-1.amzn2023.0.1      amazonlinux 10 k
=====
Installing dependencies:
apr              x86_64      1.7.2-2.amzn2023.0.2      amazonlinux 129 k
apr-util         x86_64      1.6.3-1.amzn2023.0.1      amazonlinux 98 k
generic-logos-httpd noarch     18.0.0-12.amzn2023.0.3      amazonlinux 19 k
httpd-core       x86_64      2.4.62-1.amzn2023          amazonlinux 1.4 M
httpd-filesystem noarch     2.4.62-1.amzn2023          amazonlinux 14 k
httpd-tools      x86_64      2.4.62-1.amzn2023          amazonlinux 81 k
libbrotli        x86_64      1.0.9-4.amzn2023.0.2      amazonlinux 315 k
libsodium        x86_64      1.0.19-4.amzn2023          amazonlinux 176 k
libxml2          x86_64      1.1.34-5.amzn2023.0.2      amazonlinux 241 k
mailcap          noarch     2.1.49-3.amzn2023.0.3      amazonlinux 33 k
nginx-filesystem noarch     1:1.24.0-1.amzn2023.0.4      amazonlinux 9.8 k
php8.3-cli       x86_64      8.3.10-1.amzn2023.0.1      amazonlinux 3.7 M
php8.3-common    x86_64      8.3.10-1.amzn2023.0.1      amazonlinux 737 k
=====
Installed:
apr-1.7.2-2.amzn2023.0.2.x86_64      apr-util-1.6.3-1.amzn2023.0.1.x86_64
generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch  httpd-2.4.62-1.amzn2023.x86_64
httpd-filesystem-2.4.62-1.amzn2023.noarch      httpd-tools-2.4.62-1.amzn2023.x86_64
libsodium-1.0.19-4.amzn2023.x86_64          libxml2-1.1.34-5.amzn2023.0.2.x86_64
mod_http2-2.8.27-1.amzn2023.0.3.x86_64     mod_lua-2.4.62-1.amzn2023.x86_64
php8.3-8.3.10-1.amzn2023.0.1.x86_64        php8.3-cli-8.3.10-1.amzn2023.0.1.x86_64
php8.3-fpm-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-process-8.3.10-1.amzn2023.0.1.x86_64
php8.3-xml-8.3.10-1.amzn2023.0.1.x86_64    php8.3-sodium-8.3.10-1.amzn2023.0.1.x86_64
=====
apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64
httpd-core-2.4.62-1.amzn2023.x86_64
libbrotli-1.0.9-4.amzn2023.0.2.x86_64
mailcap-2.1.49-3.amzn2023.0.3.noarch
nginx-filesystem-1:1.24.0-1.amzn2023.0.4.noarch
php8.3-common-8.3.10-1.amzn2023.0.1.x86_64
php8.3-ocache-8.3.10-1.amzn2023.0.1.x86_64
php8.3-process-8.3.10-1.amzn2023.0.1.x86_64
php8.3-sodium-8.3.10-1.amzn2023.0.1.x86_64
=====
Complete!
[ec2-user@ip-172-31-83-157 ~]$ sudo yum install gcc glibc glibc-common
```

Step 8: Next we install C/C++ compiler (GCC) along with the necessary C libraries required for compiling and running C programs. Use the following command.

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

```
[ec2-user@ip-172-31-83-157 ~]$ sudo yum install gcc glibc glibc-common
Last metadata expiration check: 0:05:58 ago on Sat Sep 28 03:46:46 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
  libmpc-devel                          x86_64       2.4.7-1.amzn2023.0.3      amazonlinux    38 k
  libtalloc-ltdl                        x86_64       1.1.23-7.amzn2023.0.1     amazonlinux    32 k
  libcrypt-devel                       x86_64
                                             0.1.1-1.amzn2023.0.1

Installed:
  annobin-docs-10.93-1.amzn2023.0.1.noarch
  gcc-8.0.4-5.amzn2023.0.2.x86_64
  glibc-headers-x86-2.34-52.amzn2023.0.11.noarch
  libmpc-1.2.1-2.amzn2023.0.2.x86_64
  make-4.3-5.amzn2023.0.2.x86_64

Complete!
[ec2-user@ip-172-31-83-157 ~]$
```

Step 9: We would also need GD library and its development tools. For that, run this command

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

```
[ec2-user@ip-172-31-83-157 ~]$ sudo yum install gd gd-devel
Last metadata expiration check: 0:11:36 ago on Sat Sep 28 03:46:46 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-fs                            x86_64       3.2.1-2.amzn2023.0.4      amazonlinux    16 k
  fontconfig                          x86_64       2.13.94-2.amzn2023.0.2     amazonlinux    273 k
  fontconfig-devel                     x86_64       2.13.94-2.amzn2023.0.2     amazonlinux    128 k
  fonts-fs                            noarch      1:2.0.5-12.amzn2023.0.2   amazonlinux    9.5 k
  freetype                            x86_64       2.13.2-5.amzn2023.0.1     amazonlinux    423 k
  freetype-devel                      x86_64       2.13.2-5.amzn2023.0.1     amazonlinux    912 k
  glib2-devel                          x86_64       2.74.7-689.amzn2023.0.2   amazonlinux    486 k

Installed:
  brotli-1.0.9-4.amzn2023.0.2.x86_64
  bzip2-devel-1.0.8-6.amzn2023.0.2.x86_64
  cmake-fs-3.22.2-1.amzn2023.0.4.x86_64
  fontconfig-devel-2.13.94-2.amzn2023.0.2.x86_64
  freetype-2.13.2-5.amzn2023.0.1.x86_64
  gd-2.3.3-5.amzn2023.0.3.x86_64
  glib2-devel-2.74.7-689.amzn2023.0.2.x86_64
  google-noto-sans-vf-fonts-20201206-2.amzn2023.0.2.noarch
  graphite2-devel-1.3.14-7.amzn2023.0.2.x86_64
  harfbuzz-icu-7.0.0-2.amzn2023.0.1.x86_64
  jbigkit-libs-2.1-21.amzn2023.0.2.x86_64
  libICE-1.0.10-6.amzn2023.0.2.x86_64
  libX11-1.7.2-3.amzn2023.0.4.x86_64
  libX11-devel-1.7.2-3.amzn2023.0.4.x86_64
  libXau-1.0.9-6.amzn2023.0.2.x86_64
  libXext-1.3.4-6.amzn2023.0.2.x86_64
  libXpm-devel-3.5.15-2.amzn2023.0.3.x86_64
  libXt-1.2.0-4.amzn2023.0.2.x86_64
  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-4.0.4.amzn2023.0.18.x86_64
  libwebrtc-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

Complete!
[ec2-user@ip-172-31-83-157 ~]$
```

```
[ec2-user@ip-172-31-83-157 ~]$ sudo yum install gd gd-devel
Last metadata expiration check: 0:11:36 ago on Sat Sep 28 03:46:46 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-fs                            x86_64       3.2.1-2.amzn2023.0.4      amazonlinux    16 k
  fontconfig                          x86_64       2.13.94-2.amzn2023.0.2     amazonlinux    273 k
  fontconfig-devel                     x86_64       2.13.94-2.amzn2023.0.2     amazonlinux    128 k
  fonts-fs                            noarch      1:2.0.5-12.amzn2023.0.2   amazonlinux    9.5 k
  freetype                            x86_64       2.13.2-5.amzn2023.0.1     amazonlinux    423 k
  freetype-devel                      x86_64       2.13.2-5.amzn2023.0.1     amazonlinux    912 k
  glib2-devel                          x86_64       2.74.7-689.amzn2023.0.2   amazonlinux    486 k
  glib2-devel-1.0.9-4.amzn2023.0.2.x86_64
  gd-devel-2.3.3-5.amzn2023.0.3.x86_64
  google-noto-fonts-common-20201206-2.amzn2023.0.2.noarch
  graphite2-1.3.14-7.amzn2023.0.2.x86_64
  harfbuzz-7.0.0-2.amzn2023.0.1.x86_64
  harfbuzz-icu-7.0.0-2.amzn2023.0.1.x86_64
  langpacks-core-font-en-3.6-21.amzn2023.0.4.noarch
  libSM-1.2.3-8.amzn2023.0.2.x86_64
  libX11-1.7.2-3.amzn2023.0.4.noarch
  libX11-common-1.7.2-3.amzn2023.0.4.noarch
  libX11-xcb-1.7.2-3.amzn2023.0.4.x86_64
  libXau-devel-1.0.9-6.amzn2023.0.2.x86_64
  libXpm-3.5.15-2.amzn2023.0.3.x86_64
  libXrender-0.9.10-14.amzn2023.0.2.x86_64
  libblkid-devel-2.37.4-1.amzn2023.0.4.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
  libsepolicy-3.4-3.amzn2023.0.3.x86_64
  libtiff-devel-4.0.4-1.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-172-31-83-157 ~]$
```

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

```
sudo adduser -m nagios
```

```
sudo passwd nagios
```

```
[ec2-user@ip-172-31-83-157 ~]$ sudo adduser -m nagios
sudo passwd nagios
Changing password for user nagios.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
```

Step 11: Create a user group called as 'nagcmd' to execute nagios commands.

```
sudo groupadd nagcmd
```

```
[ec2-user@ip-172-31-83-157 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-83-157 ~]$ |
```

Step 12: Add users apache and nagios to this user group.

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

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

```
[ec2-user@ip-172-31-83-157 ~]$ sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache
[ec2-user@ip-172-31-83-157 ~]$
```

Step 13: We create a directory downloads, to store the files of nagios server that are downloaded.

```
mkdir ~/downloads
```

```
cd ~/downloads
```

```
[ec2-user@ip-172-31-83-157 ~]$ mkdir ~/downloads
cd ~/downloads
[ec2-user@ip-172-31-83-157 downloads]$ |
```

Step 14: Now we need to install the latest versions of nagios-core and nagios-plugins. Go to the respective websites and check whether a better version is available. If newer versions are available, then right click on the download button → Copy link address.

Paste this link address in place of the current link in command.

If not run these commands.

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

```
[ec2-user@ip-172-31-83-157 downloads]$ wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
--2024-09-28 04:04:23-- 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:fe7:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: 'nagios-4.5.5.tar.gz'

nagios-4.5.5.tar.gz          100%[=====] 1.97M 5.36MB/s   in 0.4s

2024-09-28 04:04:24 (5.36 MB/s) - 'nagios-4.5.5.tar.gz' saved [2065473/2065473]

[ec2-user@ip-172-31-83-157 downloads]$
```

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

```
[ec2-user@ip-172-31-83-157 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
--2024-09-28 04:06:15-- https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2753049 (2.6M) [application/x-gzip]
Saving to: 'nagios-plugins-2.4.11.tar.gz'

nagios-plugins-2.4.11.tar.gz      100%[=====] 2.62M 5.22MB/s   in 0.5s

2024-09-28 04:06:16 (5.22 MB/s) - 'nagios-plugins-2.4.11.tar.gz' saved [2753049/2753049]

[ec2-user@ip-172-31-83-157 downloads]$ |
```

Step 15: Now, we need to extract nagios-core file into the same directory. For that, we will use tar command.

tar zxvf nagios-4.5.5.tar.gz

```
[ec2-user@ip-172-31-83-157 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/xdata/.gitignore
nagios-4.5.5/xdata/Makefile.in
nagios-4.5.5/xdata/xcdddefault.c
nagios-4.5.5/xdata/xcdddefault.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-172-31-83-157 downloads]$
```

Step 16: We need to ensure that Nagios uses a specific group (in this case, `nagcmd`) for executing external commands.

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

An **error** could be encountered here: `./configure: no such path or directory`

Solution: Navigate to the nagios-4.5.5 folder in downloads. (version could vary)

Steps: ls

```
[ec2-user@ip-172-31-83-157 downloads]$ ls
nagios-4.5.5  nagios-4.5.5.tar.gz  nagios-plugins-2.4.11.tar.gz
```

- `cd nagios-4.5.5` (use the version shown by your `ls` command)
- `./configure --with-command-group=nagcmd`

Another **error** could be **Cannot find SSL headers.**

To solve this, we need to install OpenSSL Dev Library

Steps:

sudo yum install openssl-devel

```
[ec2-user@ip-172-31-83-157 nagios-4.5.5]$ sudo yum install openssl-devel
Last metadata expiration check: 0:21:59 ago on Sat Sep 28 03:46:46 2024.
Dependencies resolved.
=====
== Package                                     Repository          Architecture      Size   Version
=====
Installing:
openssl-devel                                    amazonlinux        x86_64           3.6 M  1:3.0.8-1.amzn2023.0.14

Transaction Summary
=====
Install 1 Package

Total download size: 3.6 M
Installed size: 4.7 M
Is this ok (y/n): y
```

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

```
[ec2-user@ip-172-31-83-157 nagios-4.5.5]$ ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether the compiler supports GNU C... yes
checking whether gcc accepts -g... yes
checking for gcc option to enable C11 features... none needed
checking whether make sets $(MAKE)... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking for sys/wait.h that is POSIX.1 compatible... yes
checking for stdio.h... yes
checking for stdlib.h... yes
checking for string.h... yes
checking for inttypes.h... yes
checking for stdint.h... yes
checking for strings.h... yes
checking for sys/stat.h... yes
```

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

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

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

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

[ec2-user@ip-172-31-83-157 nagios-4.5.5]\$

Step 17: We need to compile all components of this software according to the instruction in the Makefile. For that, use this command:

`make all`

Then,

```
sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode
```

```
[ec2-user@ip-172-31-83-157 nagios-4.5.5]$ sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode
cd ./base && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagios /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagiostats /usr/local/nagios/bin
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5/base'
cd ./cgi && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
make install-basic
make[2]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/sbin
```

```
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5'
/usr/bin/install -c -m 755 -d -o root -g root /lib/systemd/system
/usr/bin/install -c -m 755 -o root -g root startup/default-service /lib/systemd/system/nagios.service
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc/objects
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/nagios.cfg /usr/local/nagios/etc/nagios.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/cgi.cgi /usr/local/nagios/etc/cgi.cgi
/usr/bin/install -c -b -m 660 -o nagios -g nagios sample-config/resource.cfg /usr/local/nagios/etc/resource.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/templates.cfg /usr/local/nagios/etc/objects/templates.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/commands.cfg /usr/local/nagios/etc/objects/commands.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/contacts.cfg /usr/local/nagios/etc/objects/contacts.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/timeperiods.cfg /usr/local/nagios/etc/objects/timeperiods.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/localhost.cfg /usr/local/nagios/etc/objects/localhost.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/windows.cfg /usr/local/nagios/etc/objects/windows.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/printer.cfg /usr/local/nagios/etc/objects/printer.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/swtch.cfg /usr/local/nagios/etc/objects/swtch.cfg

*** Config files installed ***

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

/usr/bin/install -c -m 775 -o nagios -g 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-CFG - SAMPLE CONTACT/CONTACTGROUP DEFINITIONS

# NOTES: This config file provides you with some example contact and contact
# group definitions that you can reference in host and service
# definitions.

# You don't need to keep these definitions in a separate file from your
# other object definitions. This has been done just to make things
# easier to understand.

# =====

# CONTACTS
# =====

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

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.Sairam.Konar@ves.ac.in; <***** CHANGE THIS TO YOUR EMAIL ADDRESS ****>
}

# =====

# CONTACT GROUPS
# =====

# We only have one contact in this single configuration file, so there is
# no need to create more than one contact group.

define contactgroup {
    contactgroup_name   admins
    alias              Nagios Administrators
    members            nagiosadmin
}

```

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

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

CTRL+O: Overwrite the existing file with edited file

CTRL+X: Exit nano editor.

Step 19: We need to install the necessary configuration files for the Nagios web interface. Run the following command.

sudo make install-webconf

```

[ec2-user@ip-172-31-83-157 nagios-4.5.5]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios.conf
if [ $? -eq 1 ]; then \
    ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagios.conf; \
fi

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

[ec2-user@ip-172-31-83-157 nagios-4.5.5]$

```

Step 20: Now we need to setup a user to access this nagios web interface. So we run this command to create a user called 'nagiosadmin'.

Keep this username and password saved as it is needed to login to the web interface.

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

```
[ec2-user@ip-172-31-83-157 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-172-31-83-157 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-83-157 nagios-4.5.5]$
```

Step 22: Now we go back to the downloads folder and extract the files of nagios plugin.

```
cd ~/downloads
```

```
tar zxvf nagios-plugins-2.4.11.tar.gz (Version may vary)
```

```
[ec2-user@ip-172-31-83-157 downloads]$ tar zxvf nagios-plugins-2.4.11.tar.gz
nagios-plugins-2.4.11/
nagios-plugins-2.4.11/build-aux/
nagios-plugins-2.4.11/build-aux/compile
nagios-plugins-2.4.11/build-aux/config.guess
nagios-plugins-2.4.11/build-aux/config.rpath
nagios-plugins-2.4.11/build-aux/config.sub
nagios-plugins-2.4.11/build-aux/install-sh
nagios-plugins-2.4.11/build-aux/ltmain.sh
nagios-plugins-2.4.11/build-aux/missing
nagios-plugins-2.4.11/build-aux/mkinstalldirs
nagios-plugins-2.4.11/build-aux/depcomp
nagios-plugins-2.4.11/build-aux/snippet/
nagios-plugins-2.4.11/build-aux/snippet/_Noreturn.h
nagios-plugins-2.4.11/build-aux/snippet/arg-nonnull.h
nagios-plugins-2.4.11/build-aux/snippet/c++defs.h
nagios-plugins-2.4.11/build-aux/snippet/warn-on-use.h
nagios-plugins-2.4.11/build-aux/test-driver
nagios-plugins-2.4.11/config_test/
nagios-plugins-2.4.11/config_test/Makefile
nagios-plugins-2.4.11/config_test/run_tests
nagios-plugins-2.4.11/config_test/child_test.c
nagios-plugins-2.4.11/gl/
nagios-plugins-2.4.11/gl/m4/
```

```
nagios-plugins-2.4.11/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
[ec2-user@ip-172-31-83-157 downloads]$ |
```

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

```
cd nagios-plugins-2.4.11 (version may vary)
./configure --with-nagios-user=nagios --with-nagios-group=nagios
```

```
[ec2-user@ip-172-31-83-157 nagios-plugins-2.4.11]$ ./configure --with-nagios-user=nagios --with-nagios-group=nagios
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether to enable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables
config.status: creating plugins-scripts/utils.sh
config.status: creating perlmods/Makefile
config.status: creating test.pl
config.status: creating pkg/solaris/pkginfo
config.status: creating po/Makefile.in
config.status: creating config.h
config.status: config.h is unchanged
config.status: executing depfiles commands
config.status: executing libtool commands
config.status: executing po-directories commands
config.status: creating po/POTFILES
config.status: creating po/Makefile
[ec2-user@ip-172-31-83-157 nagios-plugins-2.4.11]$ |
```

Step 24: We need to compile all components of this software according to the instruction in the Makefile. For that, use the commands:

```
make
sudo make install
```

```
+1
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-172-31-83-157 nagios-plugins-2.4.11]$ |
```

Step 25: We need to register the Nagios service with the system, which would make it able to manage the server status. So run the following commands

```
sudo chkconfig --add nagios
sudo chkconfig nagios on
```

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

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

```
[ec2-user@ip-172-31-90-224 nagios-plugins-2.0.3]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
Nagios Core 4.0.8
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 08-12-2014
License: GPL

Website: http://www.nagios.org
Reading configuration data...
Error in configuration file '/usr/local/nagios/etc/nagios.cfg' - Line 452 (Check result path '/usr/local/nagios/var/spool/checkresults' is not a valid directory)
  Error processing main config file!

[ec2-user@ip-172-31-90-224 nagios-plugins-2.0.3]$ sudo mkdir -p /usr/local/nagios/var/spool/checkresults
[ec2-user@ip-172-31-90-224 nagios-plugins-2.0.3]$ sudo chown nagios:nagios /usr/local/nagios/var/spool/checkresults
sudo chmod 775 /usr/local/nagios/var/spool/checkresults
[ec2-user@ip-172-31-90-224 nagios-plugins-2.0.3]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.0.8
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 08-12-2014
License: GPL
```

Error: Error in configuration file '/usr/local/nagios/etc/nagios.cfg' - Line 452 (Check result path '/usr/local/nagios/var/spool/checkresults' is not a valid directory)

It is an error in processing main config file!

Solution: Create the missing directory, set the permissions, verify it.

- sudo mkdir -p /usr/local/nagios/var/spool/checkresults (Create)
- sudo chown nagios:nagios /usr/local/nagios/var/spool/checkresults
- sudo chmod 775 /usr/local/nagios/var/spool/checkresults (permissions)

Now rerun the command

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

```
Created symlink /etc/systemd/system/multi-user.target.wants/nagios.service → /usr/lib/systemd/system/nagios.service.
[ec2-user@ip-172-31-83-157 nagios-plugins-2.4.11]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

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

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

Running pre-flight check on configuration data...

Checking objects...
  Checked 8 services.
  Checked 1 hosts.
  Checked 1 host groups.
  Checked 0 service groups.
  Checked 1 contacts.
  Checked 1 contact groups.
  Checked 24 commands.
  Checked 5 time periods.
  Checked 0 host escalations.
  Checked 0 service escalations.
Checking for circular paths...
  Checked 1 hosts
  Checked 0 service dependencies
  Checked 0 host dependencies
  Checked 5 timperiods
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-172-31-83-157 nagios-plugins-2.4.11]$ |
```

## Step 27:

sudo service nagios start

```
[ec2-user@ip-172-31-83-157 nagios-plugins-2.4.11]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-83-157 nagios-plugins-2.4.11]$ |
```

sudo systemctl status nagios

```
[ec2-user@ip-172-31-90-224 nagios-plugins-2.0.3]$ sudo systemctl restart nagios
Failed to restart nagios service: Unit nagios service not found.
[ec2-user@ip-172-31-90-224 nagios-plugins-2.0.3]$ sudo service nagios start
Reloading systemctl: [ OK ]
Starting nagios (via systemctl): [ OK ]
[ec2-user@ip-172-31-90-224 nagios-plugins-2.0.3]$ sudo systemctl status nagios
● nagios.service - LSB: Starts and stops the Nagios monitoring server
   Loaded: loaded (/etc/rc.d/init.d/nagios; generated)
     Active: active (running) since Mon 2024-09-23 09:31:32 UTC; 3min 4s ago
       Docs: man:systemd-sysv-generator(8)
     Process: 66885 ExecStart=/etc/rc.d/init.d/nagios start (code=exited, status=0/SUCCESS)
      Tasks: 6 (limit: 1112)
     Memory: 2.4M
        CPU: 85ms
       CGroup: /system.slice/nagios.service
           ├─66907 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
           ├─66909 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─66910 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─66911 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─66912 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           └─66913 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 23 09:33:21 ip-172-31-90-224.ec2.internal nagios[66907]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmppeg2SN' for writing status data: Permission denied
Sep 23 09:33:24 ip-172-31-90-224.ec2.internal nagios[66907]: SERVICE ALERT: localhost,HTTP:WARNING;SOFT,1,HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes in 0.001 seconds
Sep 23 09:33:31 ip-172-31-90-224.ec2.internal nagios[66907]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmpSMct9' for writing status data: Permission denied
Sep 23 09:33:41 ip-172-31-90-224.ec2.internal nagios[66907]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tpuSNlk1' for writing status data: Permission denied
Sep 23 09:33:51 ip-172-31-90-224.ec2.internal nagios[66907]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmpYe9EWB' for writing status data: Permission denied
Sep 23 09:34:01 ip-172-31-90-224.ec2.internal nagios[66907]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmpVoHut' for writing status data: Permission denied
Sep 23 09:34:11 ip-172-31-90-224.ec2.internal nagios[66907]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tpp0mP0' for writing status data: Permission denied
Sep 23 09:34:21 ip-172-31-90-224.ec2.internal nagios[66907]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tphUEU19' for writing status data: Permission denied
Sep 23 09:34:29 ip-172-31-90-224.ec2.internal nagios[66907]: SERVICE ALERT: localhost,HTTP:WARNING;SOFT,2,HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes in 0.001 seconds
Sep 23 09:34:31 ip-172-31-90-224.ec2.internal nagios[66907]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmpppy1CO9' for writing status data: Permission denied
Lines 1-26/26 (END)
[ec2-user@ip-172-31-90-224 nagios-plugins-2.0.3]$ ls -ld /usr/local/nagios/var
```

Error: Sep 23 09:34:31 ip-172-31-90-224.ec2.internal nagios[66907]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmpppy1CO9' for writing status data: Permission denied

Solution:

- ls -ld /usr/local/nagios/var
- sudo chown -R nagios:nagios /usr/local/nagios/var
- sudo chmod -R 755 /usr/local/nagios/var
- sudo systemctl restart nagios

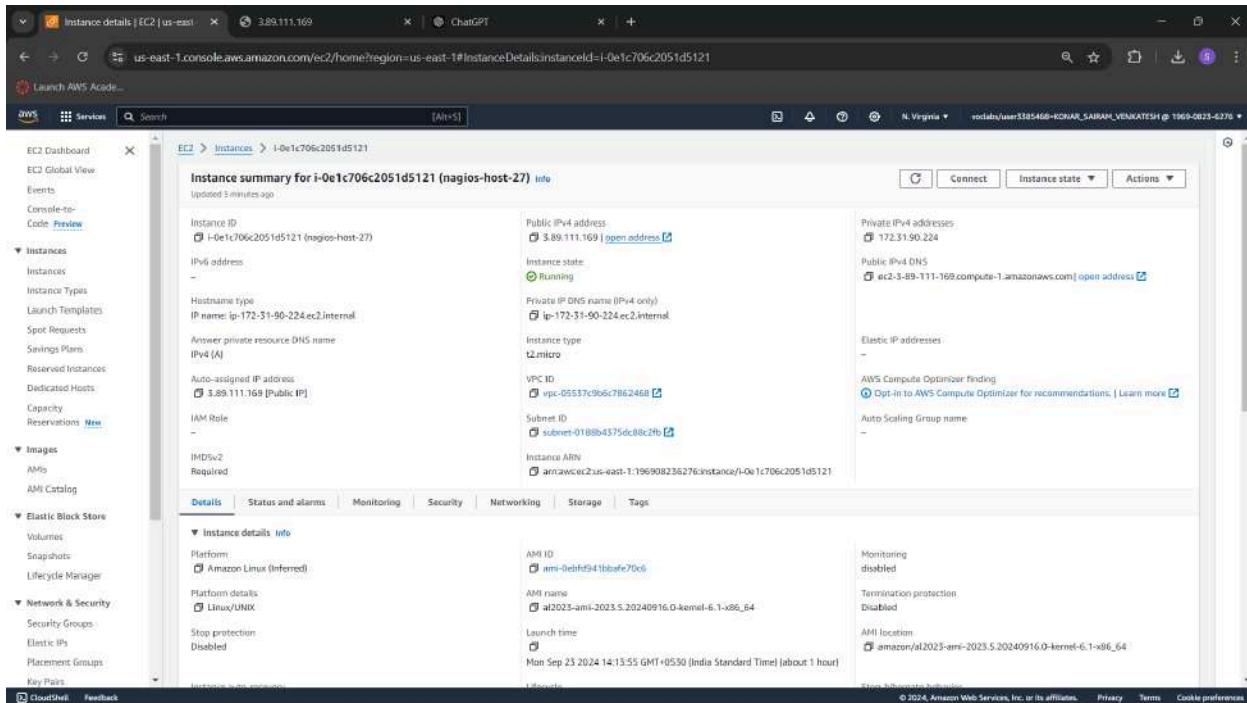
Rerun the command

sudo systemctl status nagios

```
[ec2-user@ip-172-31-83-157 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 Sat 2024-09-28 04:18:59 UTC; 12s ago
       Docs: https://www.nagios.org/documentation
   Process: 66082 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
  Process: 66083 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
 Main PID: 66084 (nagios)
   Tasks: 6 (limit: 1112)
  Memory: 8.6K
    CPU: 77ms
   CGroup: /system.slice/nagios.service
           ├─66084 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
           ├─66085 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─66086 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─66087 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─66088 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           └─66089 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 28 04:18:59 ip-172-31-83-157.ec2.internal nagios[66084]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully initialized
Sep 28 04:18:59 ip-172-31-83-157.ec2.internal nagios[66084]: qh: core query handler registered
Sep 28 04:18:59 ip-172-31-83-157.ec2.internal nagios[66084]: qh: echo service query handler registered
Sep 28 04:18:59 ip-172-31-83-157.ec2.internal nagios[66084]: qh: help for the query handler registered
Sep 28 04:18:59 ip-172-31-83-157.ec2.internal nagios[66084]: wproc: Successfully registered manager as @wproc with query handler
Sep 28 04:18:59 ip-172-31-83-157.ec2.internal nagios[66084]: wproc: Registry request: name=Core Worker 66087;pid=66087
Sep 28 04:18:59 ip-172-31-83-157.ec2.internal nagios[66084]: wproc: Registry request: name=Core Worker 66088;pid=66088
Sep 28 04:18:59 ip-172-31-83-157.ec2.internal nagios[66084]: wproc: Registry request: name=Core Worker 66086;pid=66086
Sep 28 04:18:59 ip-172-31-83-157.ec2.internal nagios[66084]: wproc: Registry request: name=Core Worker 66085;pid=66085
Sep 28 04:18:59 ip-172-31-83-157.ec2.internal nagios[66084]: Successfully launched command file worker with pid 66089
[ec2-user@ip-172-31-83-157 nagios-plugins-2.4.11]$
```

Step 28: Now, go to EC2 instance and click on instance id. Then, click on the copy icon just before the public IP address on public IP.



Step 29: Open a new tab. In the address bar type <http://<publicipaddress>/nagios>.  
This would be in the output

The screenshot shows a web browser window with the URL <http://54.210.81.106/nagios/>. The main content is the Nagios Core dashboard. At the top right, it says "Nagios® Core™ Version 4.5.5" and "September 17, 2024". A green checkmark indicates "Daemon running with PID 66054". On the left, there's a sidebar with links for General, Current Status, Problems, Reports, and System. The "Current Status" section is active. It includes a "Get Started" box with links to monitoring infrastructure, changing look, extending Nagios, getting support, training, and certification. There are also "Latest News" and "Don't Miss..." boxes, both of which are currently empty. On the right, there's a "Quick Links" box with links to Nagios Library, Labs, Exchange, Support, and the official websites. A "Page Tour" button is visible on the far right.

### Conclusion:

In this experiment, we have learned how to install and configure Nagios Core, Nagios Plugins and NRPE on a Linux machine. We are using an Amazon Linux OS instance configured with the need security rules. We need to make sure that the Nagios-core and Nagios-plugins links that are used are the ones which are up-to date (wget commands). It is needed to extract and configure these files so that no issues are detected while starting the server. Once all the setup is complete, we can start the nagios server. Using the public IP address of the EC2 instance, we can access the nagios dashboard by navigating that IP to nagios.

## Experiment 10

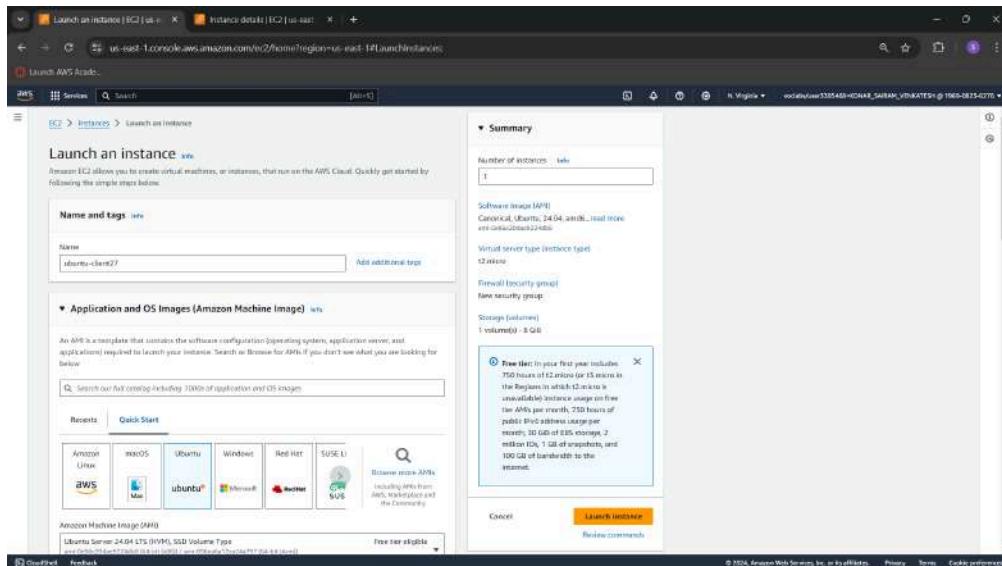
Aim: To perform Port, Service monitoring, Windows/Linux server monitoring using Nagios.

Prerequisites:

- 1) An Amazon Linux instance with nagios already set up.

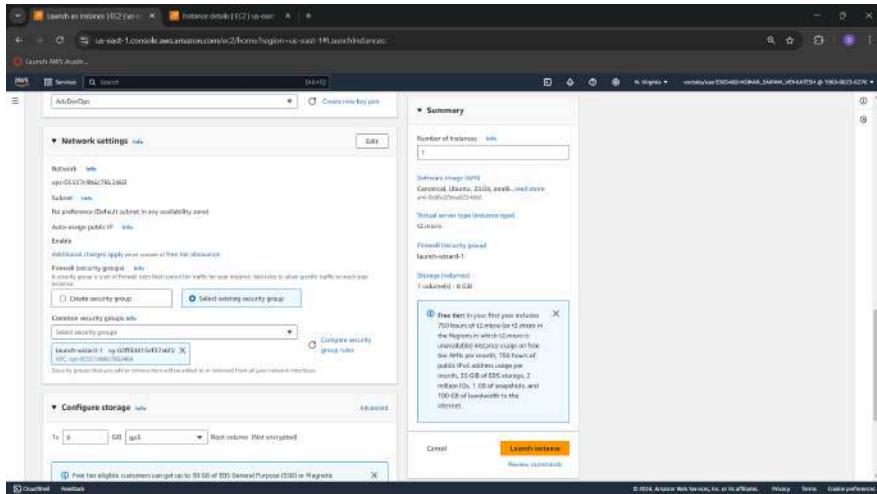
Step 1: Set up ubuntu instance

- 1) Login to your AWS account. Search for EC2 on services. Open the interface and click on Create Instance.



Select The OS Image as Ubuntu.

- 2) Make sure to select the same private key that you created for the Amazon Linux instance. Also select the same security group as you created for the Linux instance.



- Now come back to the instances screen. Click on the instance ID of your instance. Then click on Connect. Click on SSH client. Copy the example command. Now, we have to connect our local OS terminal to the instance using SSH. For this, open terminal where the private key file is located (.pem). Paste the copied SSH command and run it.

## Step 2: Execute the following on Nagios Host machine (Linux)

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

```
[ec2-user@ip-172-31-83-157 ~]$ ps -ef | grep nagios
nagios   66054      1  0 04:18 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios   66055  66054  0 04:18 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.gh
nagios   66056  66055  0 04:18 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.gh
nagios   66057  66054  0 04:18 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.gh
nagios   66058  66054  0 04:18 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.gh
nagios   66059  66054  0 04:18 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
ec2-user  66758  66657  0 04:29 pts/0    00:00:00 grep --color=auto nagios
[ec2-user@ip-172-31-83-157 ~]$ |
```

- Now, make yourself as the root user, and create a folder with the path '/usr/local/nagios/etc/objects/monitorhosts/linuxhosts'  
**sudo su**  
**mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts**

```
[ec2-user@ip-172-31-83-157 ~]$ sudo su
mkdir /usr/local/nagios/etc/objects/monitorhosts
mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-83-157 ec2-user]# |
```

- 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
```

- 4) 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

**Change hostname and alias to linuxserver**

**Change address to public ip address of client instance (Ubuntu instance)**

```
# Define a host for the local machine

define host{
    use                 linux-server ; Name of host template
                        ; This host definition is (or inherited)
    host_name           linuxserver
    alias               linuxserver
    address             32.226.136.73
}
```

**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         localhost     ; Comma separated list of hosts that belong to this hostgroup
}
```

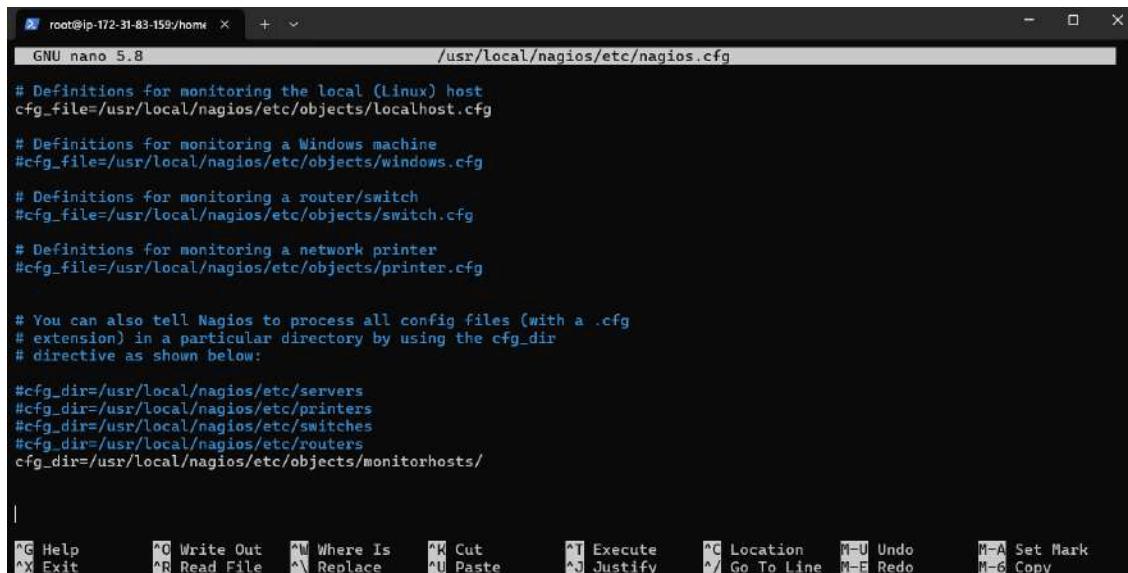
**Change the occurrences of hostname further in the document from localhost to linuxserver**

- 5) Now, we need to edit the nagios configuration file to add this directory.

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

Run this command and add the following line

**cfg\_dir=/usr/local/nagios/etc/objects/monitorhosts/**



```
root@ip-172-31-83-159:~% nano /usr/local/nagios/etc/nagios.cfg
GNU nano 5.8

# 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/
```

- 6) Now we verify the configuration files.

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

```
[root@ip-172-31-83-157 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 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-172-31-83-157 ec2-user]# |
```

- 7) Once the files are verified, we need to restart the server.

```
service nagios restart
```

```
[root@ip-172-31-83-159 nagios-plugins-2.0.3]# service nagios restart
Restarting nagios (via systemctl): [ OK ]
[root@ip-172-31-83-159 nagios-plugins-2.0.3]# |
```

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

- 1) First, we check for any new updates, then we install gcc, nagios nrpe server and nagios plugins.

```
sudo apt update -y
```

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

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

```
ubuntu@ip-172-31-81-89:~$ 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://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
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 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]
```

```
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[1495,1569]
ubuntu @ user manager service: systemd[1500]
```

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

```
ubuntu@ip-172-31-81-89:~$ |
```

- 2) We need to add the public IP address of our host Nagios machine (Linux) to the nrpe configuration file.

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

Under allowed\_hosts, add the nagios host ip address (public)

```
GNU nano 7.2                                         /etc/nagios/nrpe.cfg *
# You can either supply a username or a UID.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
nrpe_user=nagios

# NRPE GROUP
# This determines the effective group that the NRPE daemon should run as.
# You can either supply a group name or a GID.
#
# 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=127.0.0.1,54.210.81.106

# COMMAND ARGUMENT PROCESSING
# This option determines whether or not the NRPE daemon will allow clients
# to specify arguments to commands that are executed. This option only works
# if the daemon was configured with the --enable-command-args configure script
# option.

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

#### Step 4: Check the Nagios Dashboard

- 1) Go to Nagios dashboard, click on hosts.

Here, we can see that the linuxserver is also added as a host.

**Current Network Status**  
Last Updated: Sat Sep 20 04:42:09 UTC 2024  
Updated every 90 seconds  
Region: us-east-1 - www.nagios.org

**Host Status Totals**  
Up: 2 Down: 0 Unreachable: 0 Pending: 0  
All Problems: 0 All Types: 0

**Service Status Totals**  
OK: 1 Warning: 0 Unknown: 0 Critical: 2 Pending: 3  
All Problems: 3 All Types: 16

**Host Status Details For All Host Groups**

Host	Status	Last Check	Duration	Status Information
localhost	UP	09-20-2024 04:42:16	0d 0h 2m 15s	PING OK - Packet loss = 0%, RTA = 1.15 ms
linuxserver	UP	09-20-2024 04:38:21	0d 0h 24m 0s	PING OK - Packet loss = 0%, RTA = 0.03 ms

Results 1 - 2 of 2 Matching Hosts

2) Click on linuxserver. Here, we can check all the information about linuxserver host.

**Host Information**  
Last Updated: Sat Sep 20 04:43:37 UTC 2024  
Updated every 90 seconds  
Nagios Core™ 4.5.5 - www.nagios.org  
Logged in as nagiosadmin

**Host**  
**linuxserver**  
(linuxserver)  
Member of  
No hostgroups  
IP: 107.22.153.120

**Host State Information**

Host Status:	UP (0d 0h 0m 313s)
Status Information:	PING OK - Packet loss = 0%, RTA = 1.15 ms.
Performance Data:	rta=1.15100ms,3000.000000,5000.000000,0.000000,p1=0%;80;100,0
Previous Attempt:	1/10 (HARD state)
Last Check Time:	09-20-2024 04:42:16
Check Type:	ACTIVE
Check Latency / Duration:	0.00000 / 0.03 seconds
Next Scheduled Active Check:	09-20-2024 04:47:16
Last State Change:	09-20-2024 04:40:24
Last Notification:	N/A (modification 0)
Is This Host Flapping?	NO (0.00% state change)
In Scheduled Downtime?	NO
Last Update:	09-20-2024 04:43:33 ( 0d 0h 0m 4s ago)

**Host Commands**

- Locate host on map
- Disable active checks for 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

Entry Time Author Comment Comment ID Persistent Type Expires Actions

This host has no comments associated with it.

- 3) 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 details:

- Current Network Status:**
  - Last Updated: Sat Sep 28 04:44:19 UTC 2024
  - Up: 2, Down: 0, Unreachable: 0, Pending: 0
  - All Problems: 16, All Types: 3
- Service Status Details For All Hosts:**

Host	Service	Status	Last Check	Duration	Attempt	Status Information
linuxserver	Current Load	OK	09-26-2024 04:41:01	0d 0h 3m 48s+	1/4	OK - load average: 0.00, 0.01, 0.00
	Current Users	OK	09-26-2024 04:43:39	0d 0h 3m 48s+	1/4	USERS OK - 2 users currently logged in
HTTP	PING	CRITICAL	09-26-2024 04:43:15	0d 0h 1m 54s	2/4	connect to address 197.22.153.120 and port 80: Connection refused
	PING	OK	09-26-2024 04:42:54	0d 0h 3m 48s+	1/4	PING OK - Packet loss = 0%, RTA = 1.11 ms
Root Partition	SSH	OK	09-26-2024 04:43:31	0d 0h 3m 48s+	1/4	DISK OK - free space / 6116 MB (75.35% inode=98%)
	Swap Usage	PENDING	N/A	0d 0h 3m 48s+	1/4	Service check scheduled for Sat Sep 28 04:44:09 UTC 2024
Total Processes	SSH	PENDING	N/A	0d 0h 3m 48s+	1/4	Service check scheduled for Sat Sep 28 04:44:46 UTC 2024
	Total Processes	PENDING	N/A	0d 0h 3m 48s+	1/4	Service check scheduled for Sat Sep 28 04:45:24 UTC 2024
localhost	Current Load	OK	09-26-2024 04:39:36	0d 0h 24m 34s	1/4	OK - load average: 0.00, 0.02, 0.00
	Current Users	OK	09-26-2024 04:40:14	0d 0h 23m 56s	1/4	USERS OK - 2 users currently logged in
HTTP	PING	WARNING	09-26-2024 04:43:51	0d 0h 20m 19s	4/4	HTTP WARNING: HTTP/1.1 405 Forbidden - 319 bytes in 0.00 second response time
	PING	OK	09-26-2024 04:41:29	0d 0h 22m 41s	1/4	PING OK - Packet loss = 0%, RTA = 0.03 ms
Root Partition	SSH	OK	09-26-2024 04:42:06	0d 0h 22m 4s	1/4	DISK OK - free space / 6116 MB (75.35% inode=98%)
	Swap Usage	OK	09-26-2024 04:42:44	0d 0h 21m 20s	1/4	SSH OK - OpenSSH_8.7 (protocol 2.0)
Total Processes	Swap Usage	CRITICAL	09-26-2024 04:41:21	0d 0h 17m 49s	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	09-26-2024 04:43:59	0d 0h 20m 11s	1/4	PROCS: 37 processes with STATE = R/SZDT

In this case, we have monitored -

Servers: 1 linux server

Services: swap

Ports: 22, 80 (ssh, http)

Processes: User status, Current load, total processes, root partition, etc.

## Conclusion:

In this experiment, we learned to perform port service monitoring and server monitoring using Nagios. For this, we need the Linux instance used to host the Nagios dashboard and server. Also, we would need an Ubuntu instance which would be linked to a second host. We need to set up some configurations on the Linux instance and add the IP address of the Ubuntu instance. After that, we need to make the same initial setup on the ubuntu instance as the Linux instance. Add the IP address of linux instance in allowed hosts. After restarting the NRPE server, we can see the 'linuxserver' host added.

## Experiment 11

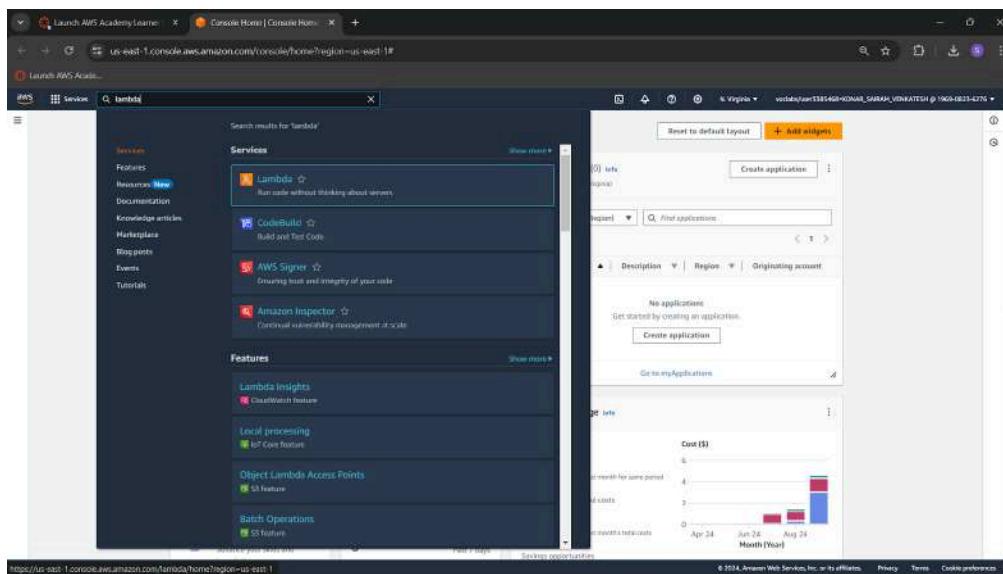
**Aim:** To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

**Prerequisites:**

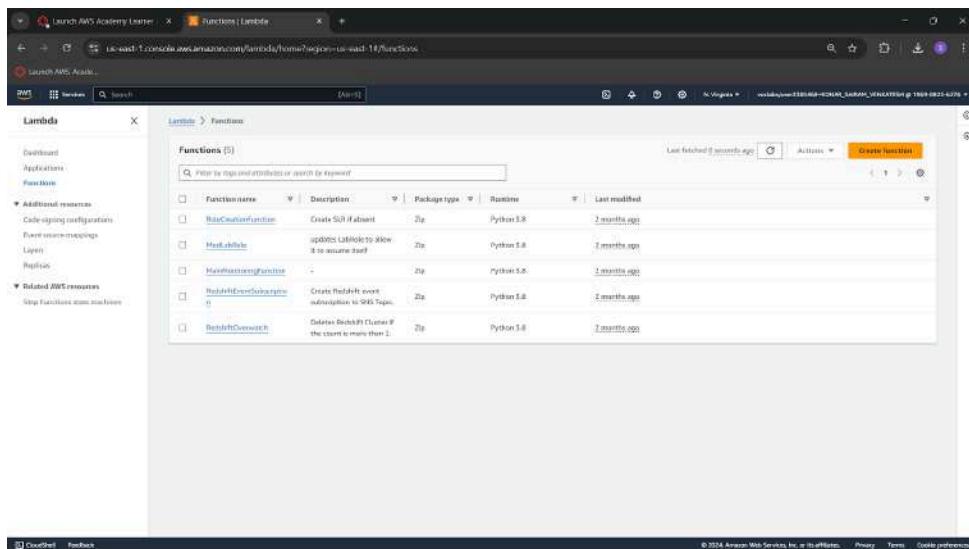
- 1) AWS account (academy recommended)

### Step 1: Set up AWS Lambda Function

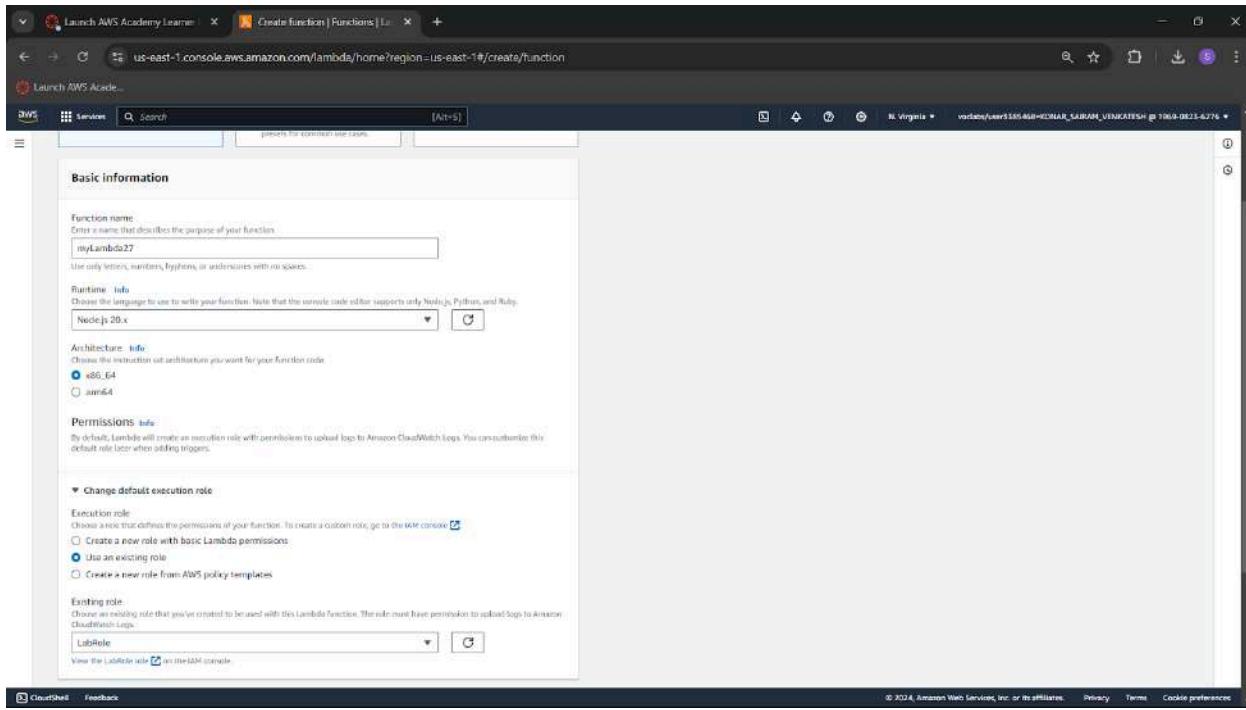
- 1) Search for Lambda in the services tab. Click on it once found.



- 2) Click on create functions.



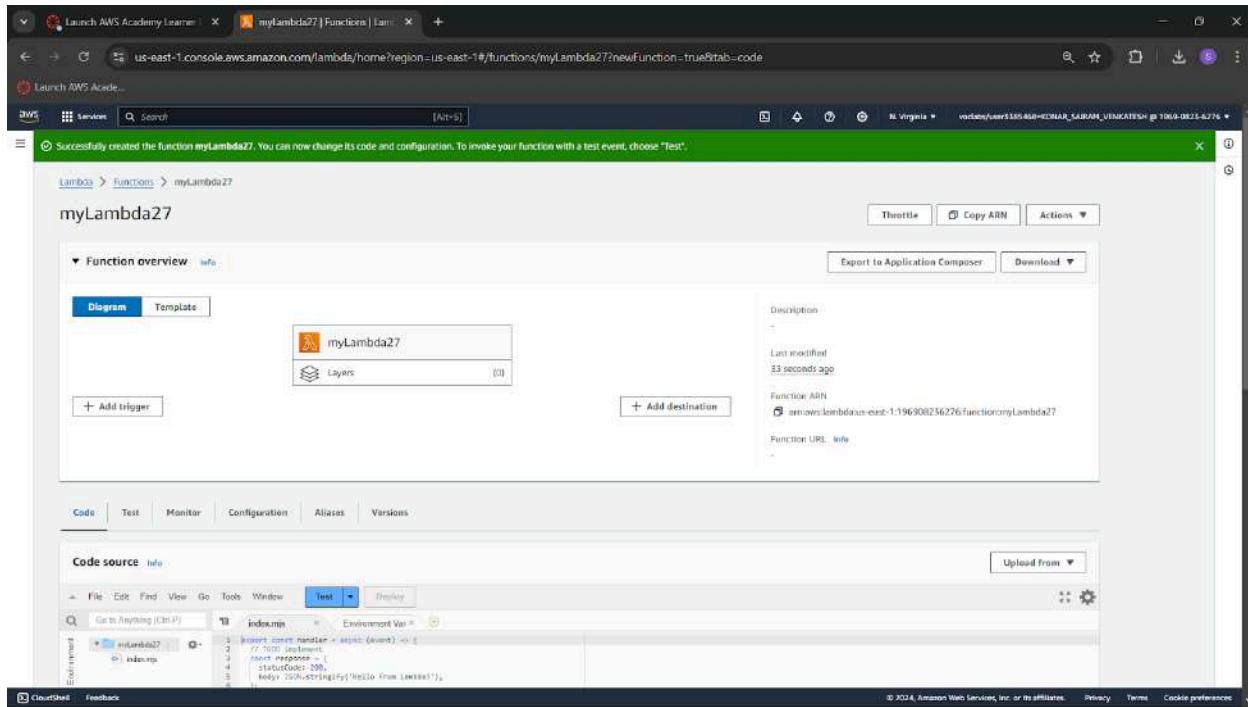
- 3) Give a name to your Lambda function. Select the runtime as Node.js 20.x (You can also use python). Select the architecture as x86\_64. Set the default execution role as LabRole if you are doing this on your academy account. (Use an existing role → LabRole)



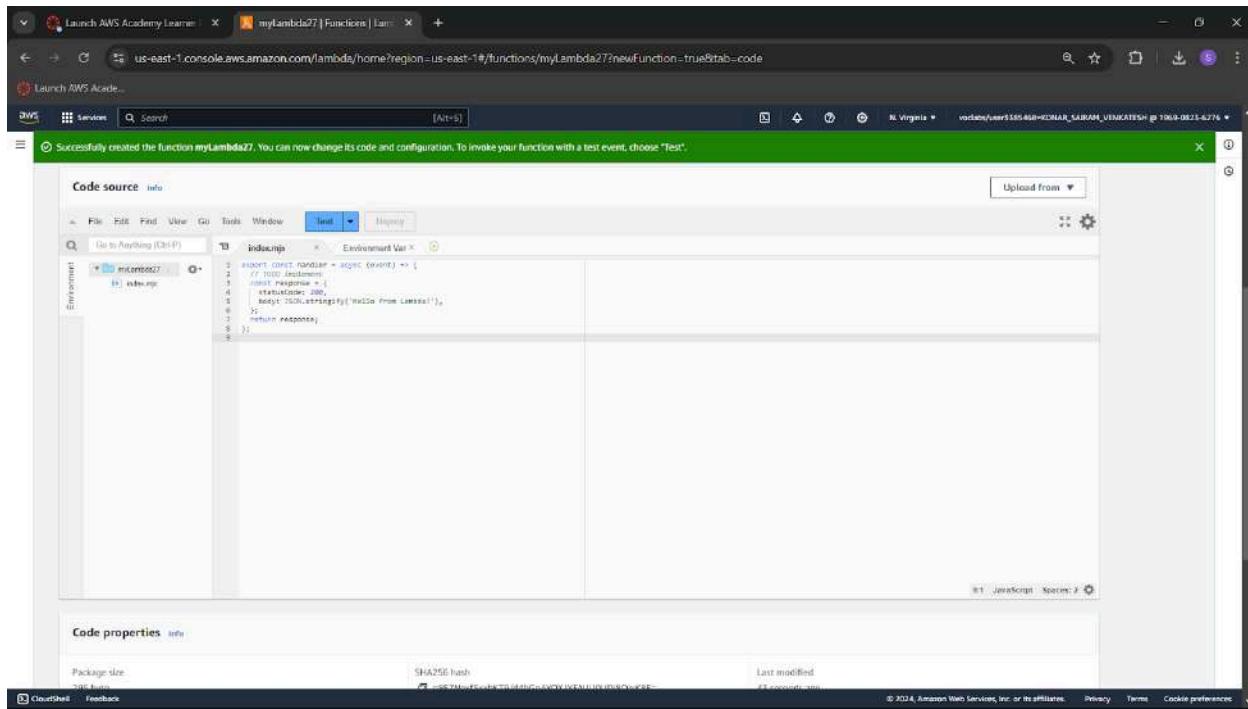
- 4) Once the function is created, click on the name of the function (myLambda27 in my case).

Functions (7)						
	Function name	Description	Package type	Runtime	Last modified	Actions
<input type="checkbox"/>	myLambda27	-	Zip	Node.js 20.x	5 days ago	<a href="#">Edit</a>
<input type="checkbox"/>	RoleCreationFunction	Create S3If absent	Zip	Python 3.8	2 months ago	<a href="#">Edit</a>
<input type="checkbox"/>	ModLabRole	updates LabRole to allow it to assume itself	Zip	Python 3.8	2 months ago	<a href="#">Edit</a>
<input type="checkbox"/>	myLambda27_12	-	Zip	Python 3.12	5 days ago	<a href="#">Edit</a>
<input type="checkbox"/>	MainMonitoringFunction	-	Zip	Python 3.8	2 months ago	<a href="#">Edit</a>
<input type="checkbox"/>	RedshiftEventSubscription	Create Redshift event subscription to SNS Topic:	Zip	Python 3.8	2 months ago	<a href="#">Edit</a>
<input type="checkbox"/>	RedshiftOverwatch	Deletes Redshift Cluster if the count is more than ?.	Zip	Python 3.8	2 months ago	<a href="#">Edit</a>

5) This is the dashboard of our lambda function.

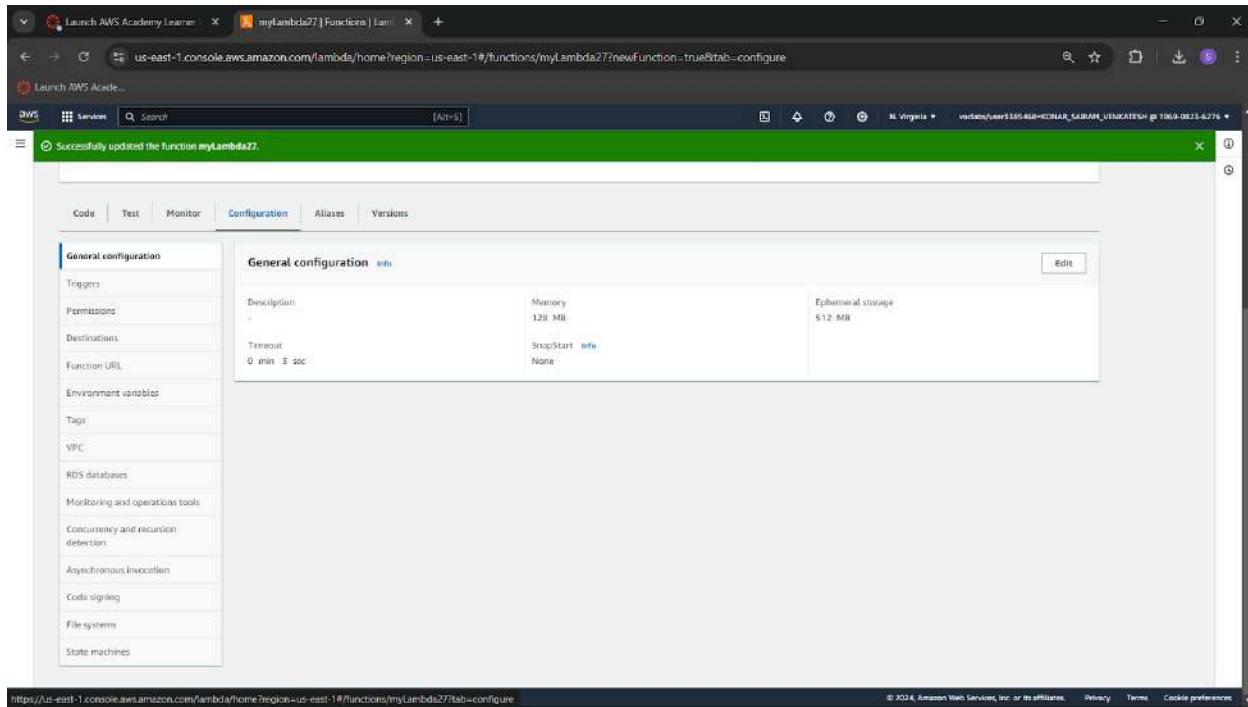


6) This function has the following default code, which is used to print "Hello form Lambda!".

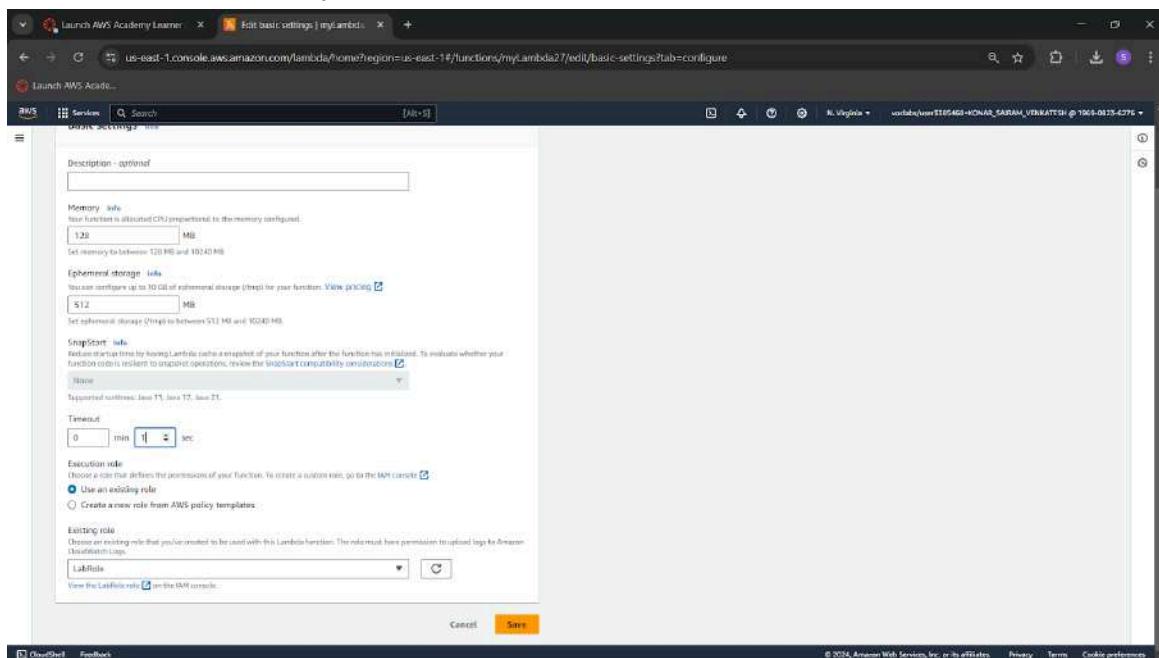


### Step 3: Set up configurations and test events

- Just above the test code, you would find Configuration, click on it. Then click on Edit.



- Here, change the Timeout to 1 sec. This is the time for which the function can be running before it is forcibly terminated.



3) We can see the executed changes.

The screenshot shows the AWS Lambda function configuration page for 'myLambda27'. The 'Configuration' tab is selected. In the 'General configuration' section, the 'Description' field is empty, 'Memory' is set to 128 MB, 'Ephemeral storage' is set to 512 MB, and 'Timeout' is set to 0 min. 1 sec. The 'Edit' button is visible. On the left sidebar, under 'General configuration', the 'Triggers' section is expanded, showing an 'index.json' trigger. Other sections like 'Permissions', 'Destinations', 'Function URL', etc., are listed but not expanded.

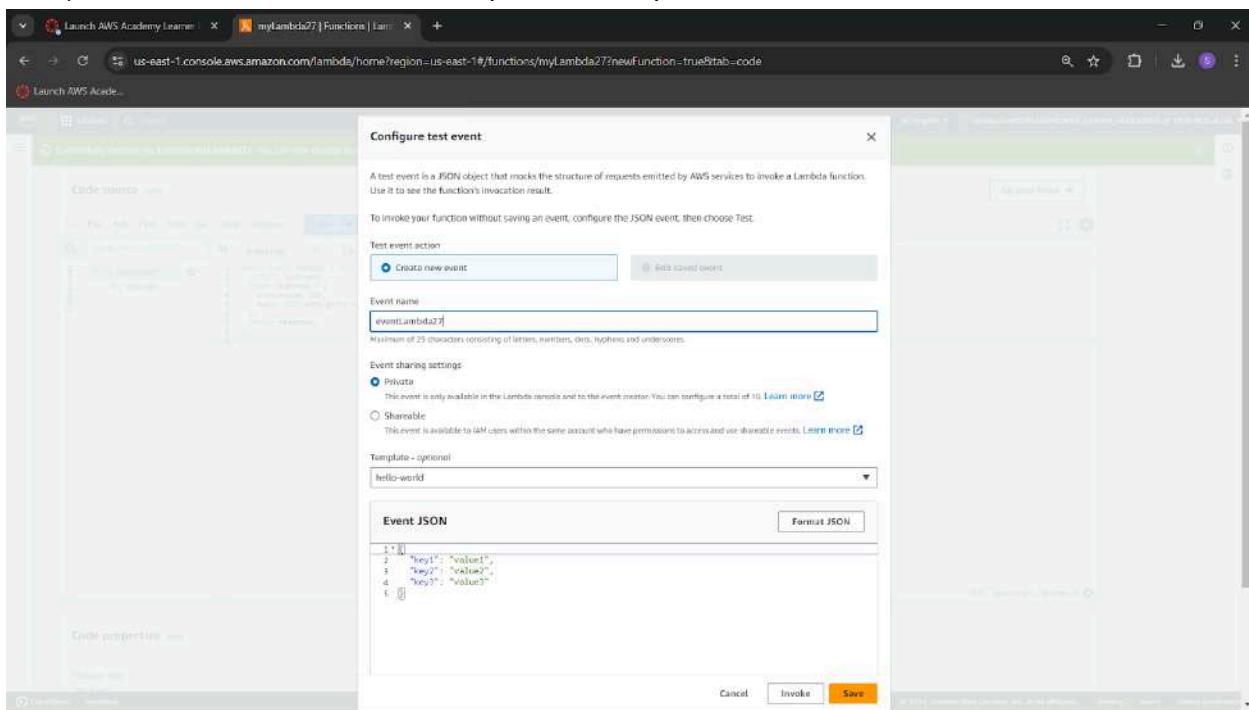
4) Switch back to the code tab. Click on the dropdown arrow near test. Then select configure test event.

The screenshot shows the AWS Lambda code editor for 'myLambda27'. A green banner at the top states: 'Successfully created the function myLambda27. You can now change its code and configuration. To invoke your function with a test event, choose "Test".' The 'Code source' tab is selected, showing the 'index.js' file with the following code:

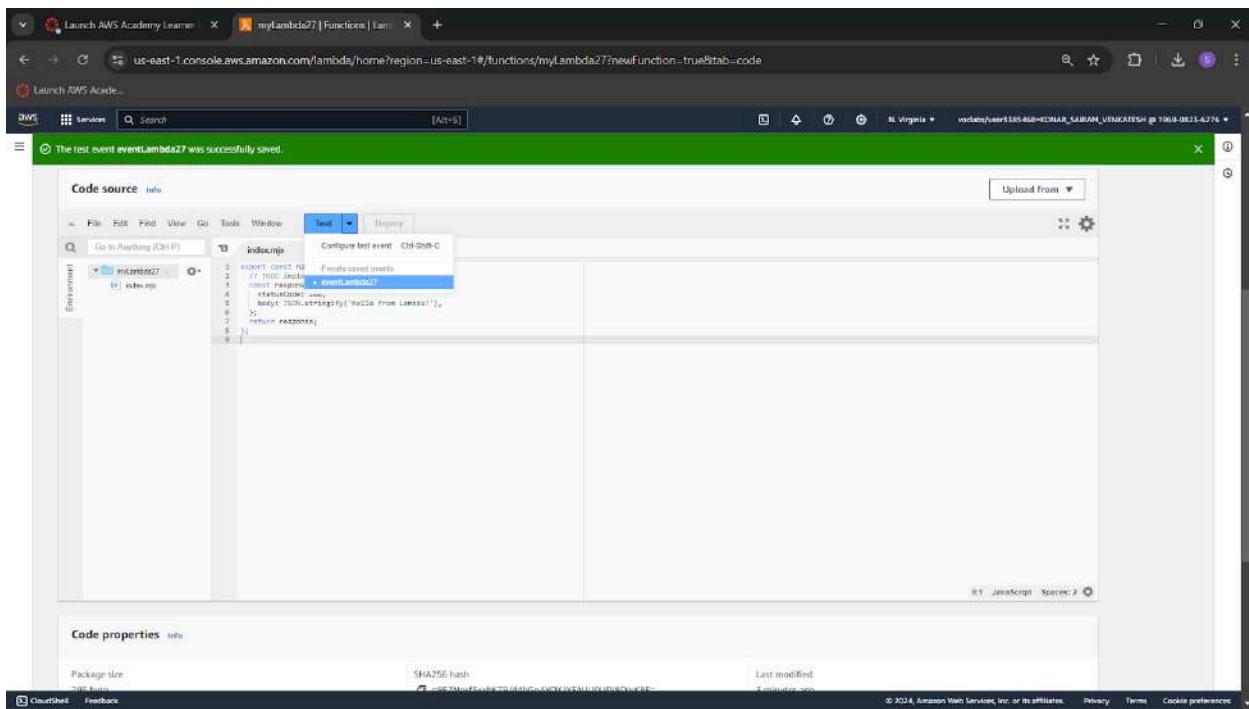
```
1 // Import AWS Lambda functions package
2 const AWS = require('aws-lambda');
3
4 // Create an API endpoint
5 const response = {
6   statusCode: 200,
7   body: JSON.stringify({HelloFromLambda!})
8 };
9
10 module.exports.handler = (event) => {
11   return response;
12 };
```

The 'Test' dropdown menu is open, with 'Configure test event' highlighted. The 'Code properties' section at the bottom shows the package size as 1.4 KB and the SHA256 hash as 2f557345c13e11111111111111111111. The status bar indicates the last modified time was 3 minutes ago.

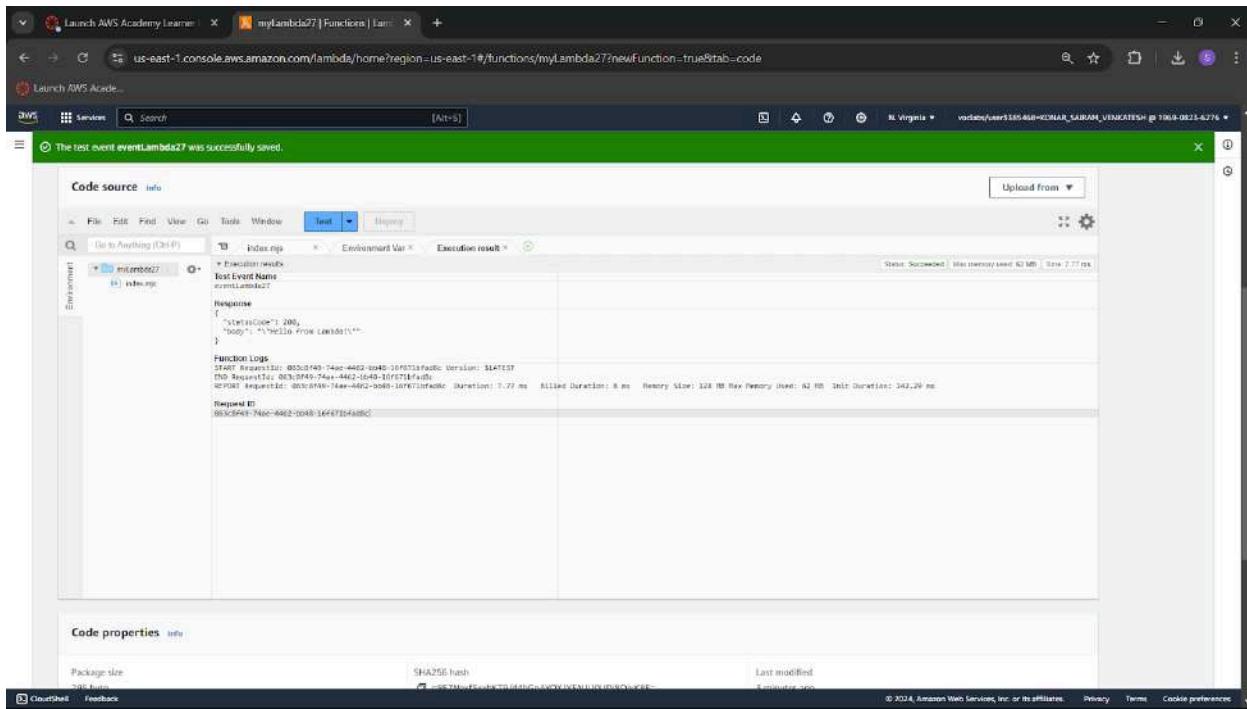
- 5) Here, create a new event, keep the other options default and save the event.



- 6) Now, again click on the dropdown. This time, select the event you have created. Then, click on TEST.



7) We can see the expected output for the sample code.



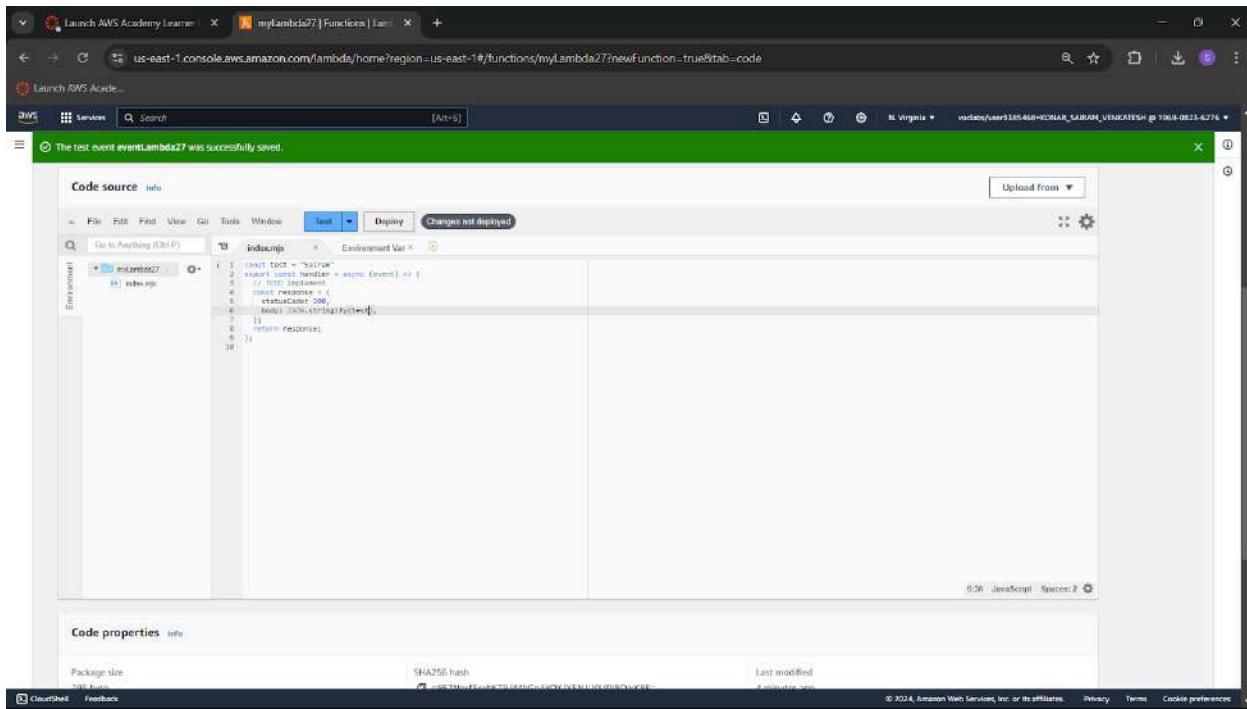
The screenshot shows the AWS Lambda function editor for the function 'myLambda27'. The 'Code source' tab is selected, displaying the code for 'index.js'. The 'Environment' sidebar shows the test event 'eventLambda27'. The 'Execution result' panel at the bottom right shows the following output:

```
Response
{
  "statusCode": 200,
  "body": "Hello from Lambda!"
}

Function Logs
START RequestId: 063c946f-74ec-46d2-a94c-1097cafa0dc Version: $LATEST
2024-04-17T10:54:45+00:00[UTC] [INFO] Lambda@Layer-1: /var/task/index.main.js:1:1
2024-04-17T10:54:45+00:00[UTC] [INFO] RequestId: 063c946f-74ec-46d2-a94c-1097cafa0dc Duration: 7.77 ms Billed Duration: 8 ms Memory Size: 128 MB Max Memory Used: 62 MB Init Duration: 142.29 ms
Request ID
063c946f-74ec-46d2-a94c-1644770480C
```

The 'Code properties' section at the bottom shows the package size, SHA256 hash, and last modified date.

8) For a test, declare a string and call it in line 6. After making the changes click on deploy.



The screenshot shows the AWS Lambda function editor for the function 'myLambda27'. The 'Code source' tab is selected, displaying the code for 'index.js'. The 'Environment' sidebar shows the test event 'eventLambda27'. The 'Deployment' button at the top is highlighted in blue, indicating that changes have been made but not yet deployed. The code in 'index.js' has been modified:1 const text = "sairam";
2 exports.handler = async (event) => {
3 // TODO implement
4 const response = {
5 statusCode: 200,
6 body: JSON.stringify(text)
7 };
8 return response;
9 }

The 'Code properties' section at the bottom shows the package size, SHA256 hash, and last modified date.

- 9) Run the test. We can see that the string we declared has come in the output.

The screenshot shows the AWS Lambda console interface. The top navigation bar includes tabs for 'Launch AWS Academy', 'myLambda27 | Functions', and '+'. Below the navigation is a search bar and a 'Services' dropdown. A green banner at the top states 'Successfully updated the function myLambda27.' The main area is titled 'Code source' and contains the code for 'index.js'. The code defines a Lambda function named 'myLambda27' that returns a JSON response with a status code of 200 and a body of 'Hello world'. Below the code editor is a 'Function logs' section showing a single log entry for a recent execution. The log entry details the start time (2024-06-21T14:45:44.820Z), end time (2024-06-21T14:45:44.820Z), duration (21 ms), memory usage (128 MB), and initialization time (175.51 ms). The log message itself is empty. At the bottom of the page, there's a 'Code properties' section with details like package size (2.4 MB), SHA256 hash, and last modified time (5 minutes ago). The footer includes links for CloudShell, Feedback, and AWS terms and conditions.

## Conclusion:

In this experiment, we successfully explored the AWS Lambda service by creating and configuring Lambda functions using Python, Java, or Node.js. We learned how to set up a Lambda function, modify its configuration (such as adjusting the timeout), and test the function with custom events. Through this process, we observed how Lambda handles executions, including managing timeouts and returning expected outputs based on the code changes.

## Experiment 12

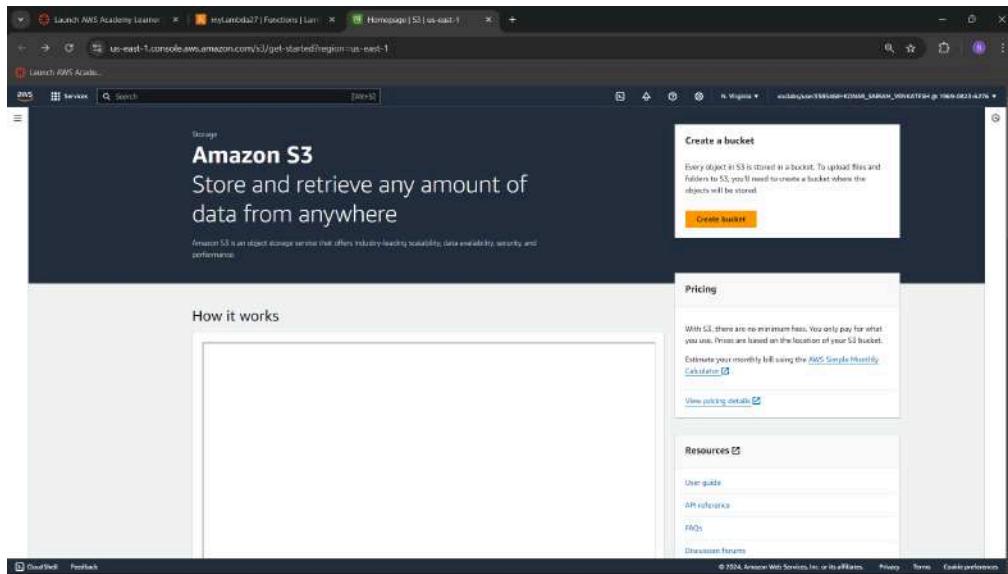
**Aim:** To create a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3.

**Prerequisites:**

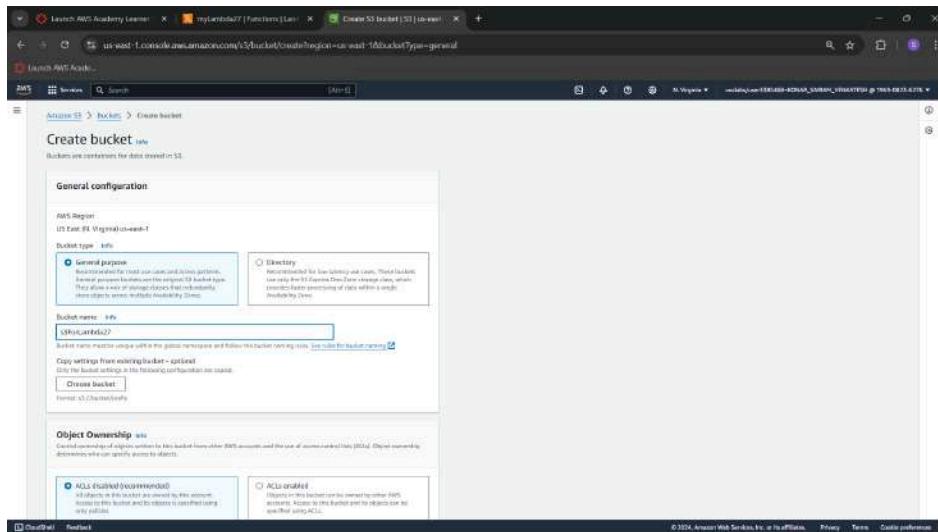
- 1) AWS account (academy preferable)
- 2) Lambda function (created in the previous experiment).

**Step 1: Create a s3 bucket.**

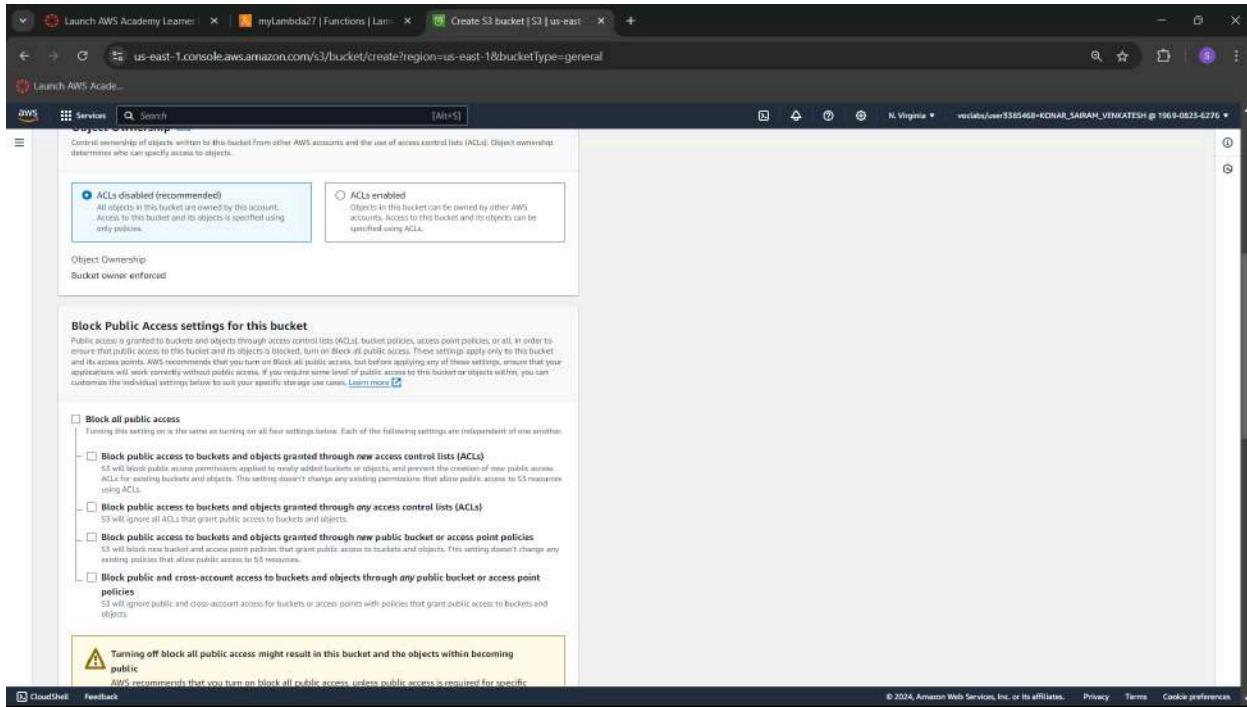
- 1) Search for S3 bucket in the services search. Then click on create bucket.



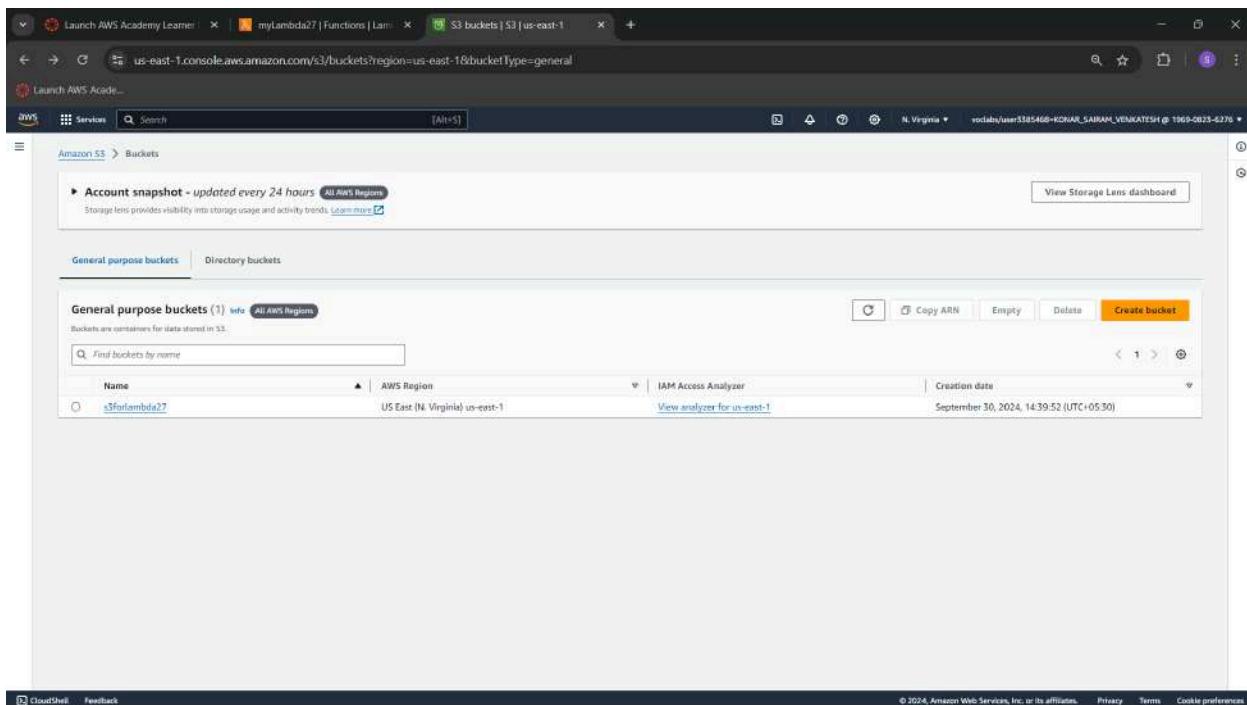
- 2) Keep the bucket as a general purpose bucket. Give a name to your bucket.



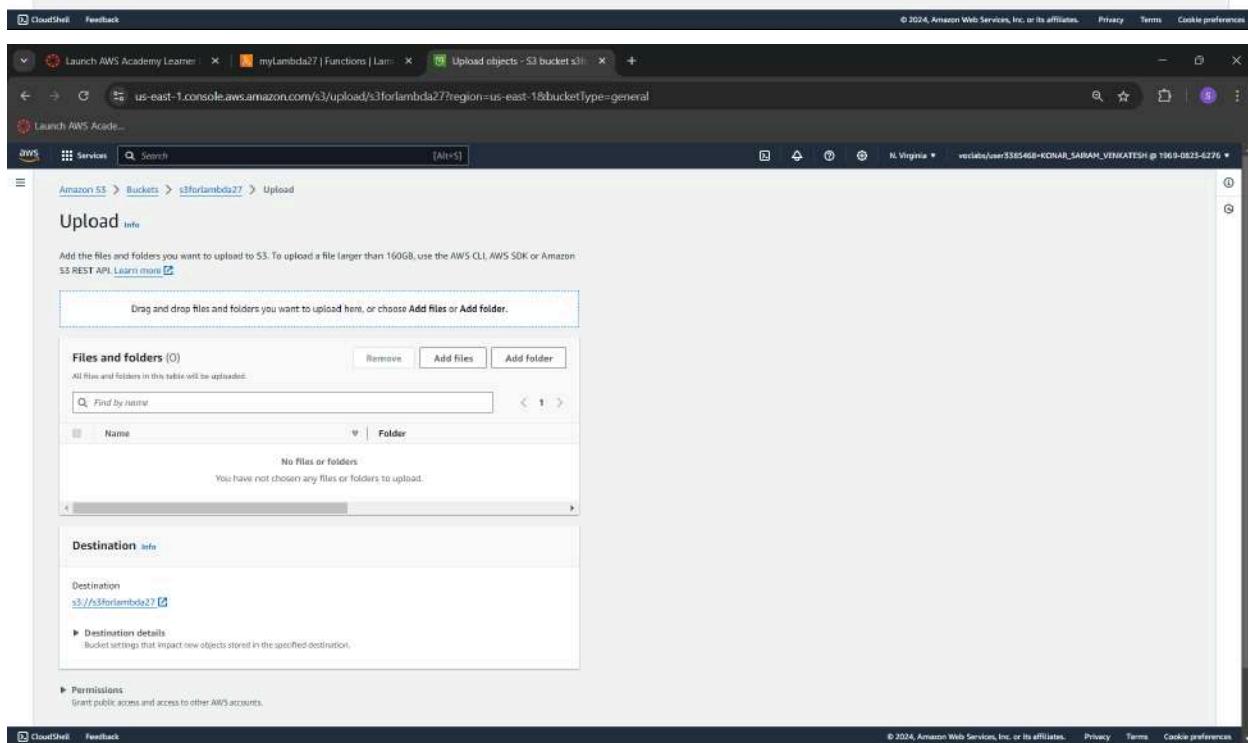
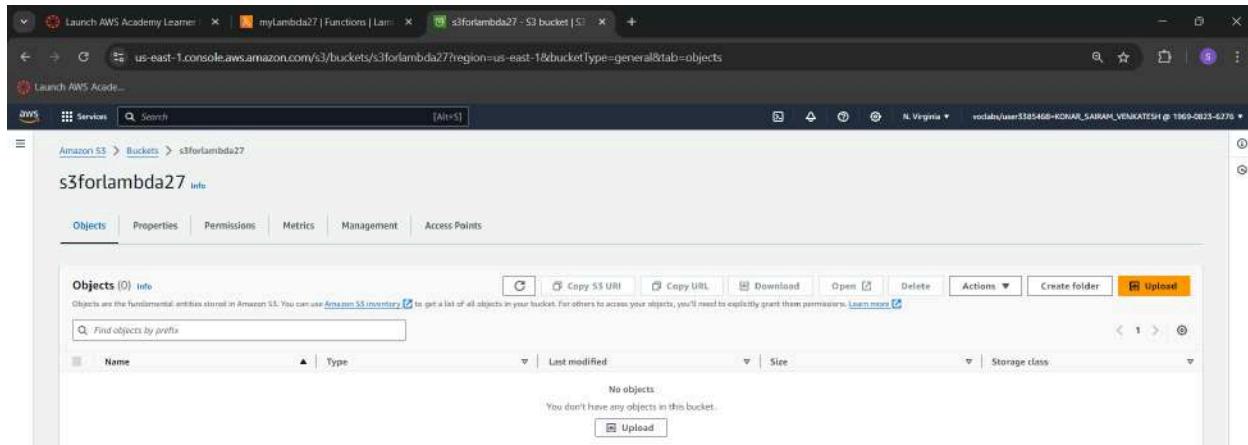
3) Uncheck block all public access.

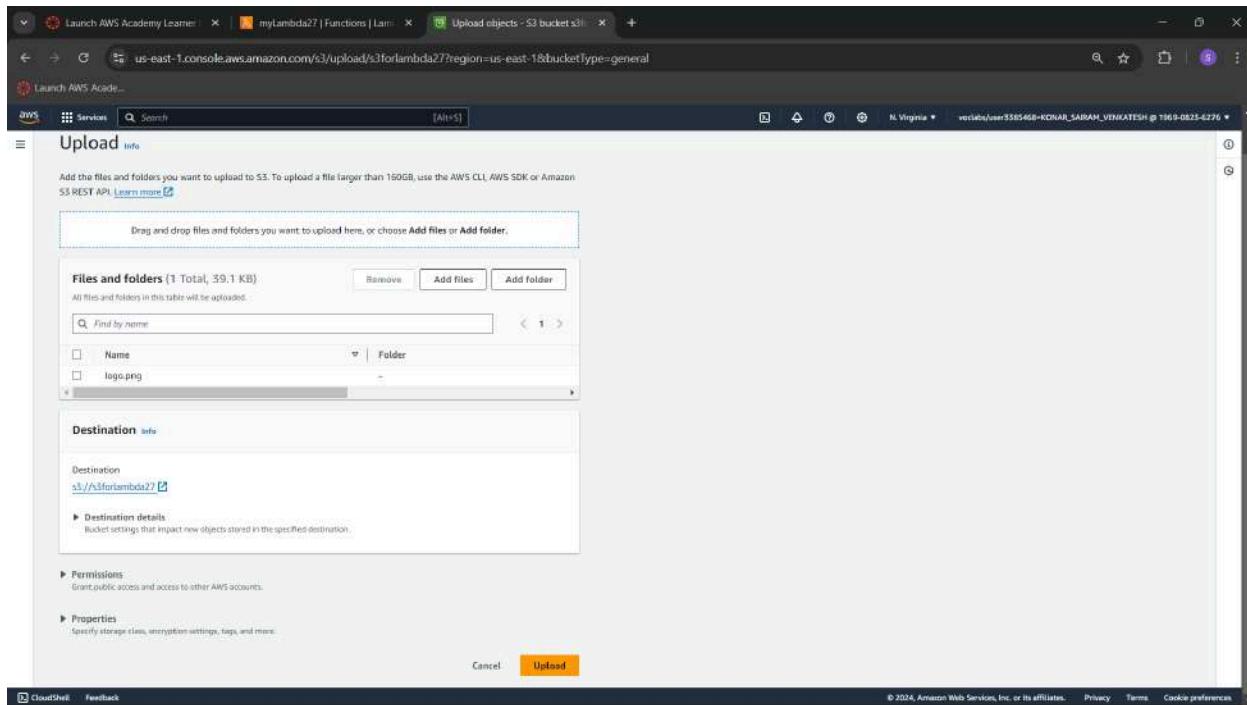


4) Keeping all other options same, click on create. This would create your bucket. Now click on the name of the bucket.

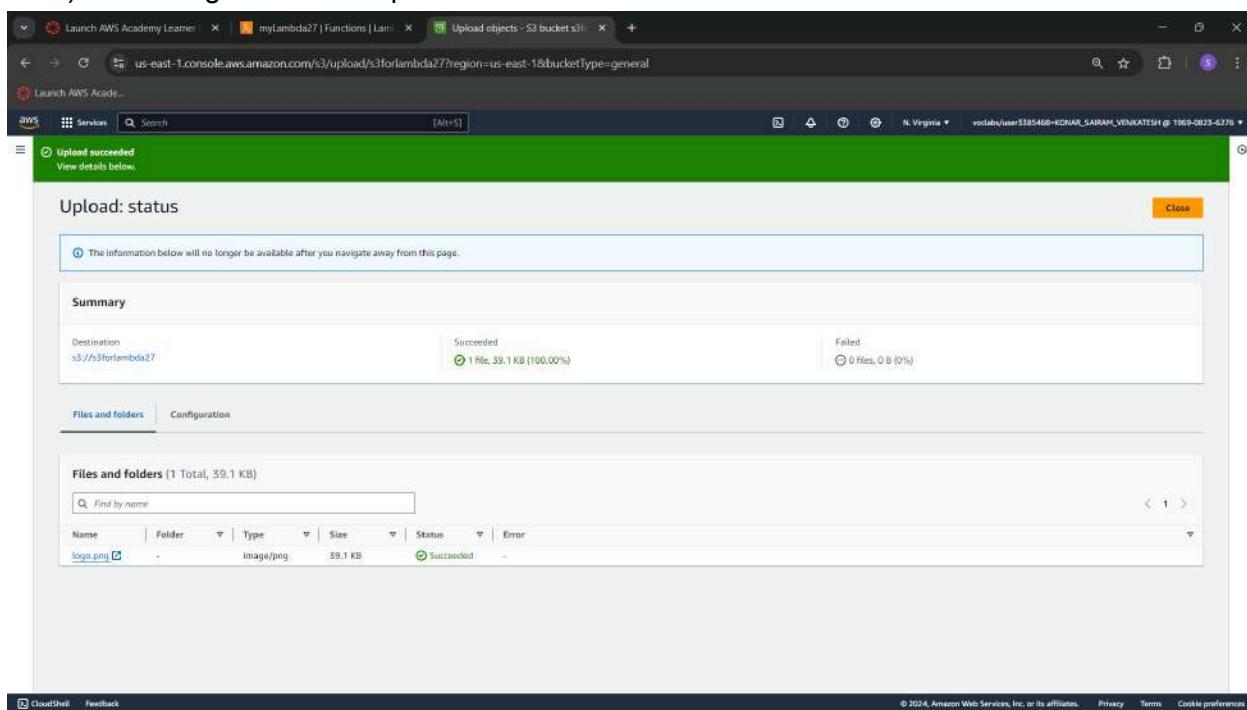


- 5) Here, click on upload, then add files. Select any image that you want to upload in the bucket and click on upload.



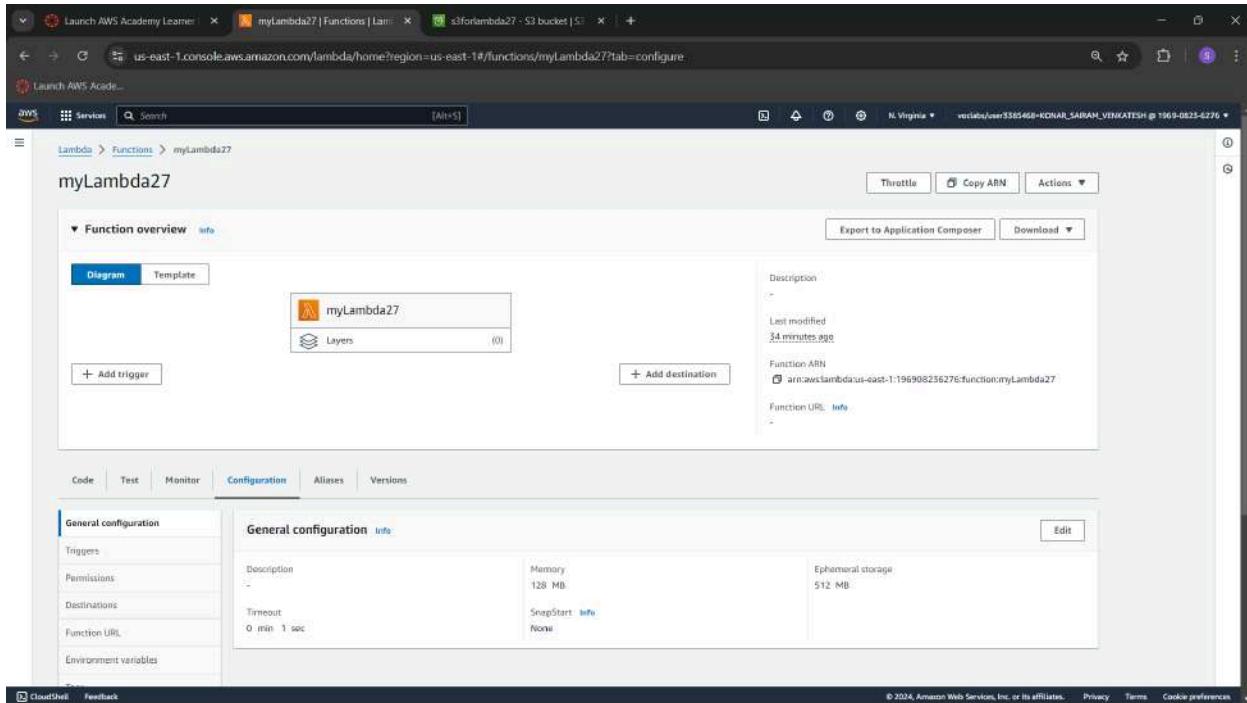


6) The image has been uploaded to the bucket.

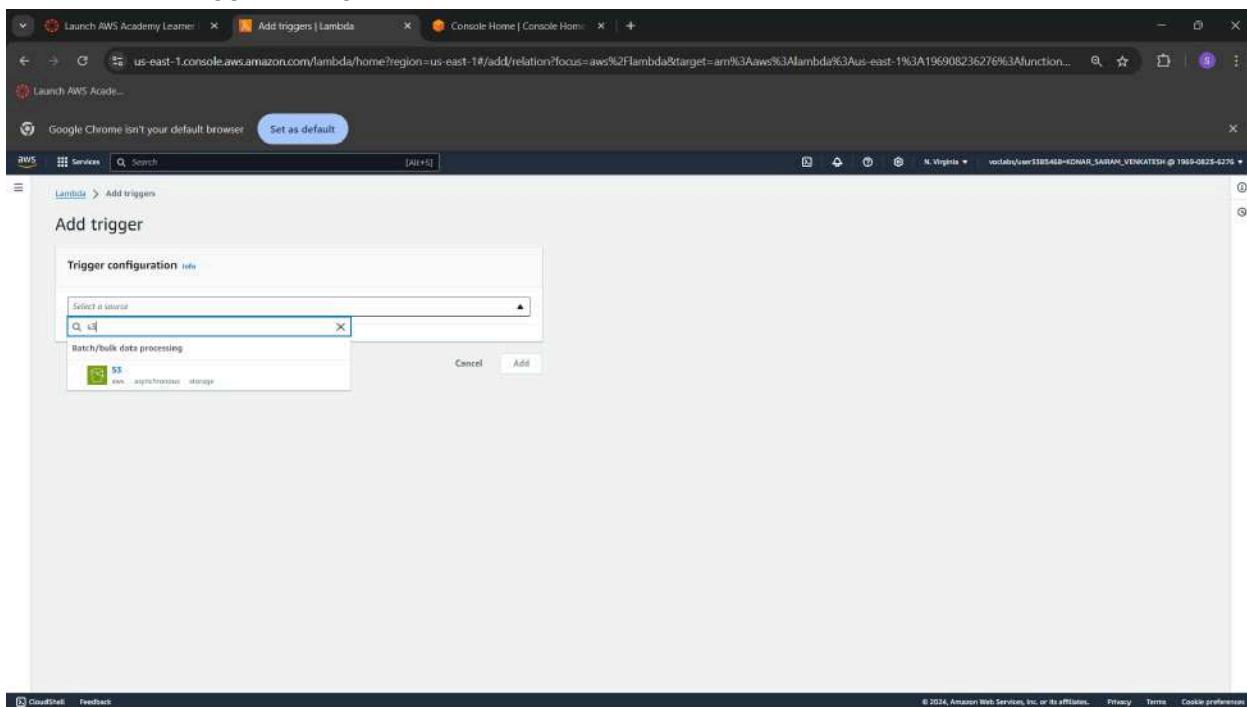


## Step 2: Configure Lambda function

- 1) Go to the lambda function you had created before. (Services → Lambda → Click on name of function). Here, click on add trigger.



- 2) Under trigger configuration, search for S3 and select it.



- 3) Here, select teh S3 bucket you created for this experiment. Acknowledge the condition given by AWS. then click on Add. This will add the S3 bucket trigger to your function.

The screenshot shows two consecutive screenshots of the AWS Lambda console. The top screenshot displays the 'Trigger configuration' step of adding triggers to a Lambda function named 'myLambda27'. It shows the selection of an S3 bucket named 's3forlambda27' and the choice of 'All object create events' as the event type. The bottom screenshot shows the 'Function overview' page for 'myLambda27', where the newly added S3 trigger is listed under the 'Triggers' section. A success message indicates that the trigger was successfully added.

**Trigger configuration - Info**

**S3** Bucket: s3forlambda27 Event types: All object create events Prefix - optional: a.g. /images/ Suffix - optional: a.g. .jpg Recursive invocation: I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.

**Add**

**myLambda27**

The trigger s3forlambda27 was successfully added to function myLambda27. The function is now receiving events from the trigger.

**Configuration**

**Triggers (1)**

**Trigger**

**s3: s3forlambda27**

- 4) Scroll down to the code section of the function. Add the following javascript code to the code area by replacing 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!')
  };
}

```

This code checks for records in the event, extracts the bucket name and object key, logs the details, and returns a success message if an image is added to the bucket.

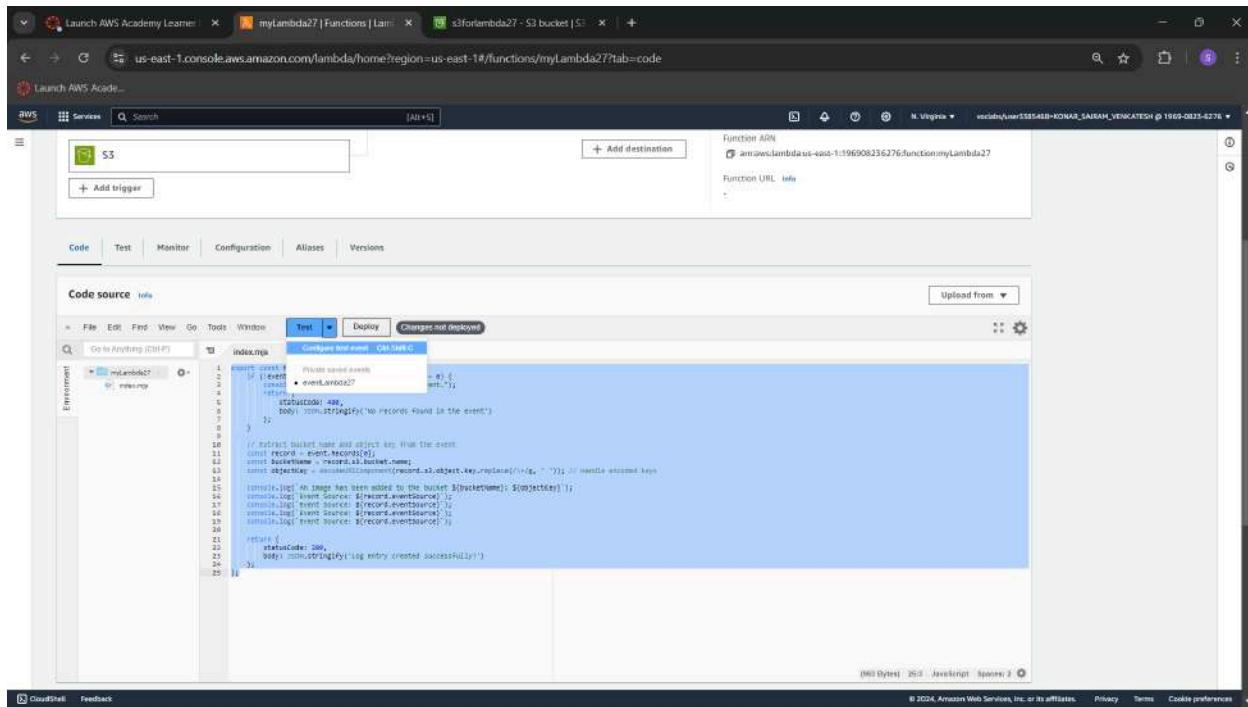
The screenshot shows the AWS Lambda function editor interface. The 'Code source' tab is selected. The code area contains the provided JavaScript function 'handler'. The code is syntax-highlighted and shows the logic for handling S3 events, extracting bucket and object names, logging them, and returning a success response. The browser status bar at the bottom indicates '1441 bytes | 2h3 | Javascript | spaces | 2'.

```

Code source Info
File Edit Find View Go Tools Window Test Deploy Changes not deployed
index.jsx Environment
Go to Anything (Ctrl+P)
index.jsx Environment Var. 
1 export const handler = async (event) => {
2   if (!event.Records || event.Records.length === 0) {
3     console.error("No records found in the event.");
4     return {
5       statusCode: 400,
6       body: JSON.stringify('No records found in the event')
7     };
8   }
9
10  // Extract bucket name and object key from the event
11  const record = event.Records[0];
12  const bucketName = record.s3.bucket.name;
13  const objectKey = decodeURIComponent(record.s3.object.key.replace(/\+/g, ' ')); // Handle
14  encoded keys
15
16  console.log(`An image has been added to the bucket ${bucketName}: ${objectKey}`);
17  console.log(`Event Source: ${record.eventSource}`);
18  console.log(`Event Source: ${record.eventSource}`);
19  console.log(`Event Source: ${record.eventSource}`);
20  console.log(`Event Source: ${record.eventSource}`);
21
22  return {
23    statusCode: 200,
24    body: JSON.stringify('Log entry created successfully!')
25  };
26

```

- 5) Now, click on the dropdown near test, then click on configure test event.



- 6) Here, select edit saved event. Select the event taht you had created before. Under Event JSON, paste the following code.

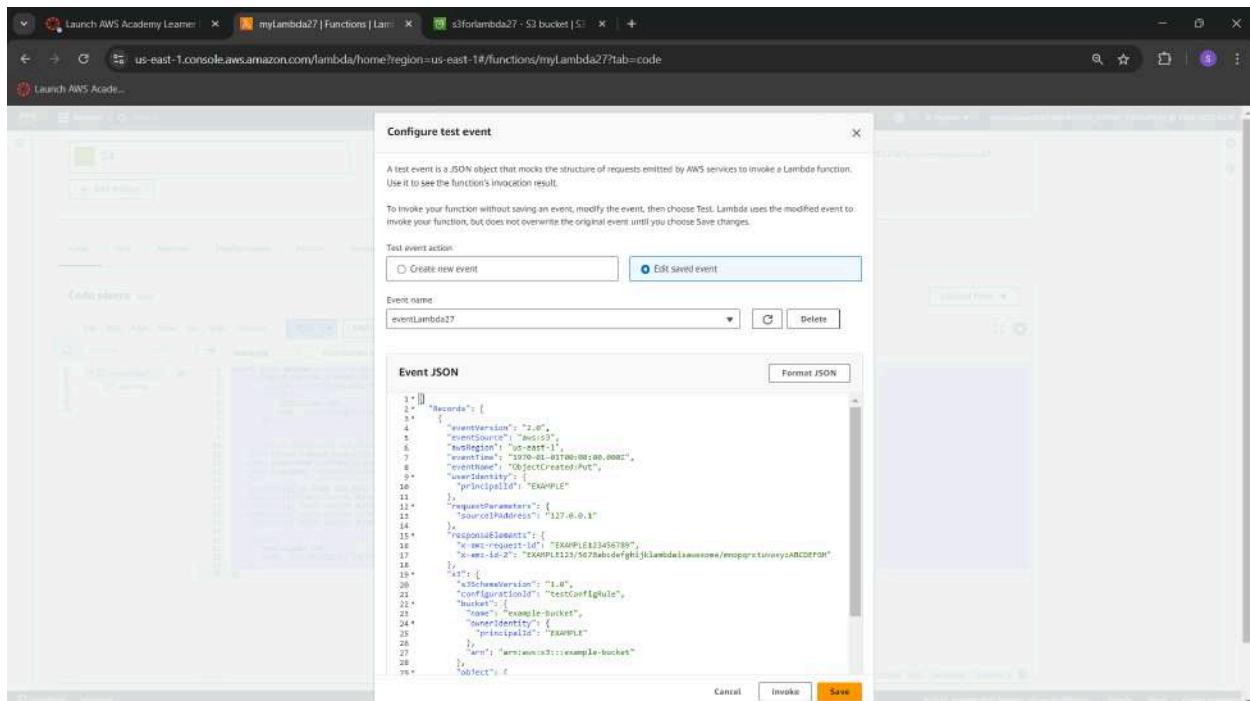
```
{
  "Records": [
    {
      "eventVersion": "2.0",
      "eventSource": "aws:s3",
      "awsRegion": "us-east-1",
      "eventTime": "1970-01-01T00:00:00.000Z",
      "eventName": "ObjectCreated:Put",
      "userIdentity": {
        "principalId": "EXAMPLE"
      },
      "requestParameters": {
        "sourceIPAddress": "127.0.0.1"
      },
      "responseElements": {
        "x-amz-request-id": "EXAMPLE123456789",
        "x-amz-id-2": "EXAMPLE123456789ABCDEF0"
      },
      "s3": {
        "s3SchemaVersion": "1.0",
        "configurationId": "testConfigRule",
        "bucket": {
          "name": "myLambda27"
        },
        "objectKey": "testObjectKey"
      }
    }
  ]
}
```

```

"bucket": {
    "name": "example-bucket",
    "ownerIdentity": {
        "principalId": "EXAMPLE"
    },
    "arn": "arn:aws:s3:::example-bucket"
},
"object": {
    "key": "test%2Fkey",
    "size": 1024,
    "eTag": "0123456789abcdef0123456789abcdef",
    "sequencer": "0A1B2C3D4E5F678901"
}
}
]
}
}

```

This JSON structure represents an S3 event notification triggered when an object is uploaded to an S3 bucket. It contains details about the event, including the bucket name (example-bucket), the object key (test/key), and metadata like the object's size, the event source (aws:s3), and the event time.



Save the changes. Then deploy the code changes by clicking on deploy.

- 7) After deploying, click on Test. The console output shows that 'an image has been added to the bucket'

The JSON response shows that the log entry was created successfully.

The screenshot shows the AWS Lambda Test interface. The 'Test' tab is selected. In the 'Execution results' section, there is a log entry:

```

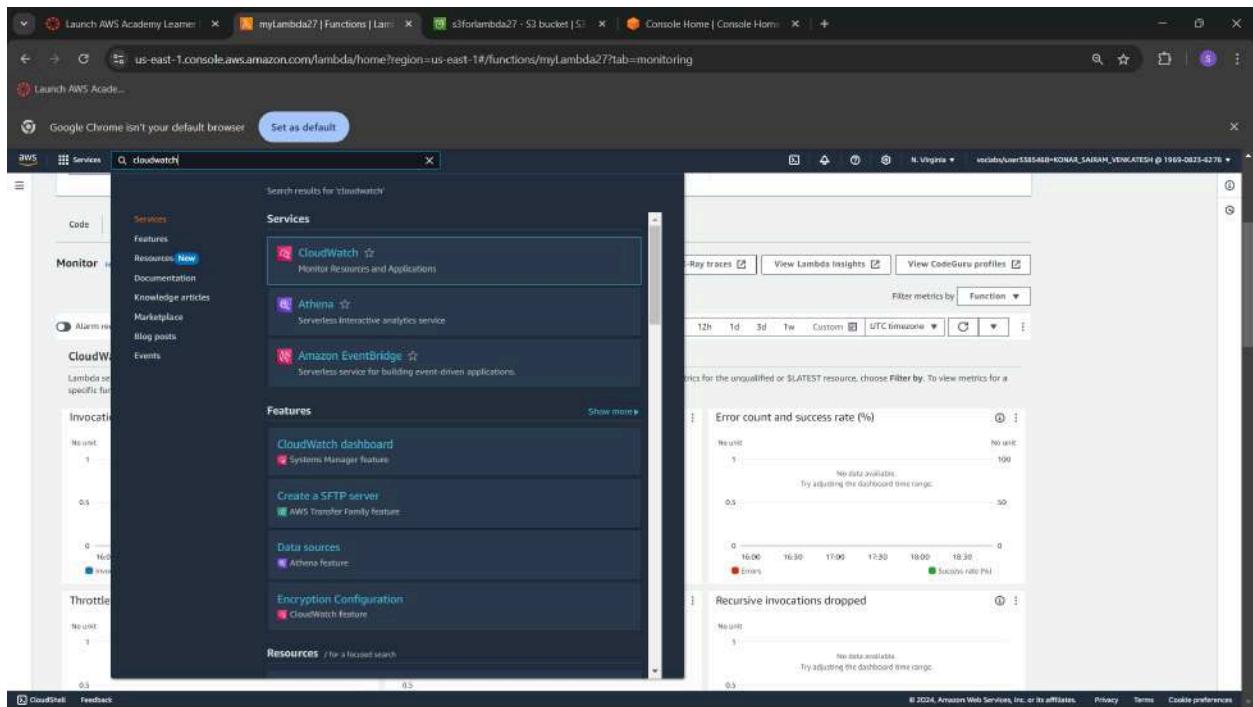
Test Event Name: event1aa0d027
Response:
{
  "statusCode": 200,
  "body": "Log entry created successfully\\\""
}

Function Log:
START RequestId: 01723939-7200-421b-a550-432105575405 version: $LATEST
2024-09-30T09:24:40,324Z 01723939-7200-421b-a550-432105575405 INFO An image has been added to the bucket example-bucket: test/key
2024-09-30T09:24:40,334Z 01723939-7200-421b-a550-432105575405 INFO Event source: aws:s3
END RequestId: 01723939-7200-421b-a550-432105575405
REPORT RequestID: 01723939-7200-421b-a550-432105575405 Duration: 29.56 ms. Allocated Duration: 30 ms. Memory size: 128 MB Max Memory Used: 64 MB. Init Duration: 147.32 ms
Request ID: 01723939-7200-421b-a550-432105575405

```

### Step 3: Check the logs

- 1) To check the logs explicitly, search for CloudWatch on services and open it in a new tab.



- 2) Here, Click on Logs → Log Groups. Select the log that has the lambda function name you just ran.

The screenshot shows the AWS CloudWatch Log Groups page. The left sidebar navigation includes 'Logs' under 'CloudWatch'. The main content area displays a table titled 'Log groups (5)'. The table columns are: Log group, Log class, Anomaly detection, Data protection, Sensitive data count, Retention, Metric filters, and Contributor Insights. The log groups listed are: /aws/lambda/RedshiftEventSubscription, /aws/lambda/RedshiftEventSubscription, /aws/lambda/RoleCreationFunction, /aws/lambda/MyLambda27, and /aws/lambda/myLambda27\_12. All log groups are of type 'Standard' and have 'Configure' under 'Anomaly detection', 'Data protection', and 'Metric filters'. The 'Retention' column shows 'Never expire' for all. The 'Contributor Insights' column shows a small icon for each group.

- 3) Here, under Log streams, select the log stream you want to check.

The screenshot shows the AWS CloudWatch Log group details page for the log group '/aws/lambda/myLambda27'. The left sidebar navigation includes 'Logs' under 'CloudWatch'. The main content area shows 'Log group details' for '/aws/lambda/myLambda27'. It includes sections for Log class (Standard), ARN, Creation time (5 days ago), Retention (Never expire), and various monitoring and compliance settings like KMS key ID, Metrics filters, Subscription filters, and Contributor insights rules. Below this, the 'Log streams' section is visible, showing a table with four log streams. The columns are: Log stream, Last event time, and a checkbox column. The log streams are: 2024/10/05/[\$LATEST]9e54081d397ae42e5b6052d571fb99fb2 (Last event time: 2024-10-05 18:55:19 UTC), 2024/09/30/[\$LATEST]290165e7477441a085774157e5f6712 (Last event time: 2024-09-30 09:24:40 UTC), 2024/09/30/[\$LATEST]1566755a128fe4c1d9605c7d8895d2599 (Last event time: 2024-09-30 08:54:34 UTC), and 2024/09/30/[\$LATEST]ef49207f960543f7b7d197e1f230da (Last event time: 2024-09-30 08:51:55 UTC). There are buttons for Actions, View in Logs Insights, Start tailing, and Create log stream.

4) Here again, we can see that 'An image has been added to the bucket'.

The screenshot shows the AWS CloudWatch Log Events interface. The left sidebar navigation includes 'CloudWatch', 'Logs', 'Log groups', 'Metrics', 'X-Ray traces', 'Events', 'Application Signals', 'Network monitoring', and 'Insights'. The main content area displays 'Log events' for the log group '/aws/lambda/myLambda27'. A search bar at the top allows filtering by timestamp or message. Below the search bar are buttons for 'Actions', 'Start tailing', and 'Create metric filter'. The log entries list several events, all timestamped around 2024-09-20T09:24:48.174Z:

- 2024-09-20T09:24:48.174Z INFO START runtime.Version: /nodejs12.x Runtime.Version: 2024.09.20T09:24:48.174Z FunctionArn: arn:aws:lambda:us-east-1:123456789012:function:myLambda27
- 2024-09-20T09:24:48.174Z START RequestId: 01723355-7200-4210-8958-432186579405 Version: \$LATEST
- 2024-09-20T09:24:48.174Z 2024-09-20T09:24:48.174Z INFO An image has been added to the bucket example-bucket: test/key
- 2024-09-20T09:24:48.174Z 2024-09-20T09:24:48.174Z INFO Event source: s3:object
- 2024-09-20T09:24:48.174Z 2024-09-20T09:24:48.174Z INFO EventID: 01723355-7200-4210-8958-432186579405 INFO event\_source: s3:object
- 2024-09-20T09:24:48.174Z 2024-09-20T09:24:48.174Z INFO RequestID: 01723355-7200-4210-8958-432186579405 REPORT REQUESTID: 01723355-7200-4210-8958-432186579405 Duration: 29.56 ms killed Duration: 29 ms MemorySize: 102 MB MaxMemoryUsed: 94 MB InitDuration: 147.52 ms

Below the log entries, a note states: 'No newer events at this moment. Auto-verify paused. [Resume](#)'.

## Conclusion:

In this experiment, we successfully created a Lambda function that logs a message when an image is added to a specific S3 bucket. By configuring an S3 bucket trigger for the Lambda function, we demonstrated how AWS services can work together to automate tasks. The function logged important details about the event, including the bucket name and object key. After deploying the function and testing with a sample event, we verified the logs in CloudWatch, confirming that the Lambda function correctly detected and logged the addition of an image to the bucket. This experiment highlighted the seamless integration between AWS Lambda and S3 for event-driven processes.