

Static Hosting:

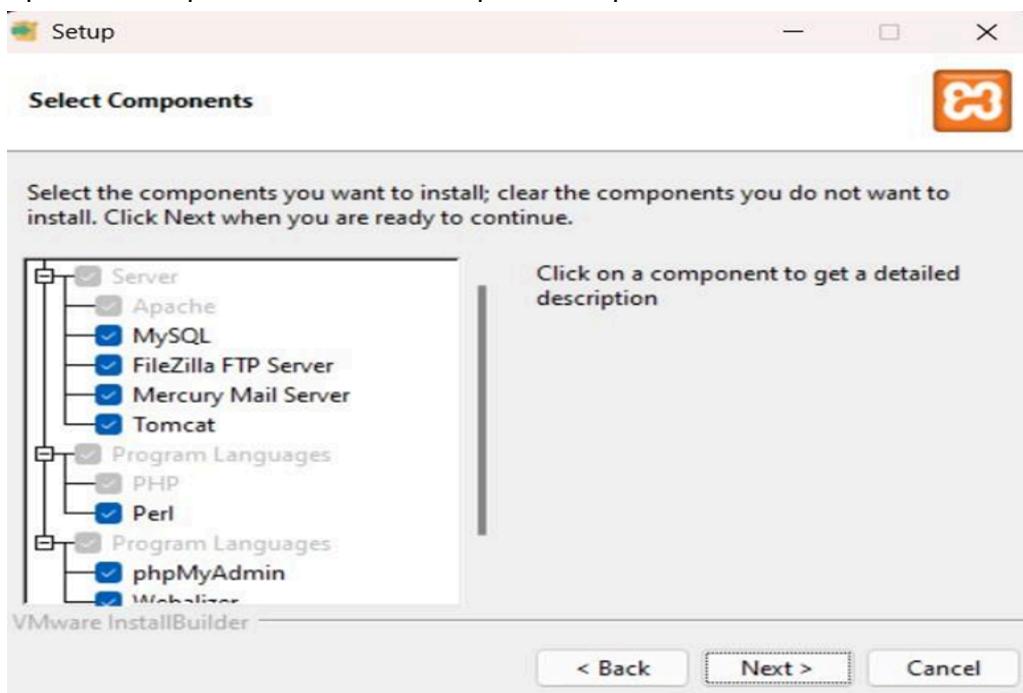
1) On local server (XAMPP)

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

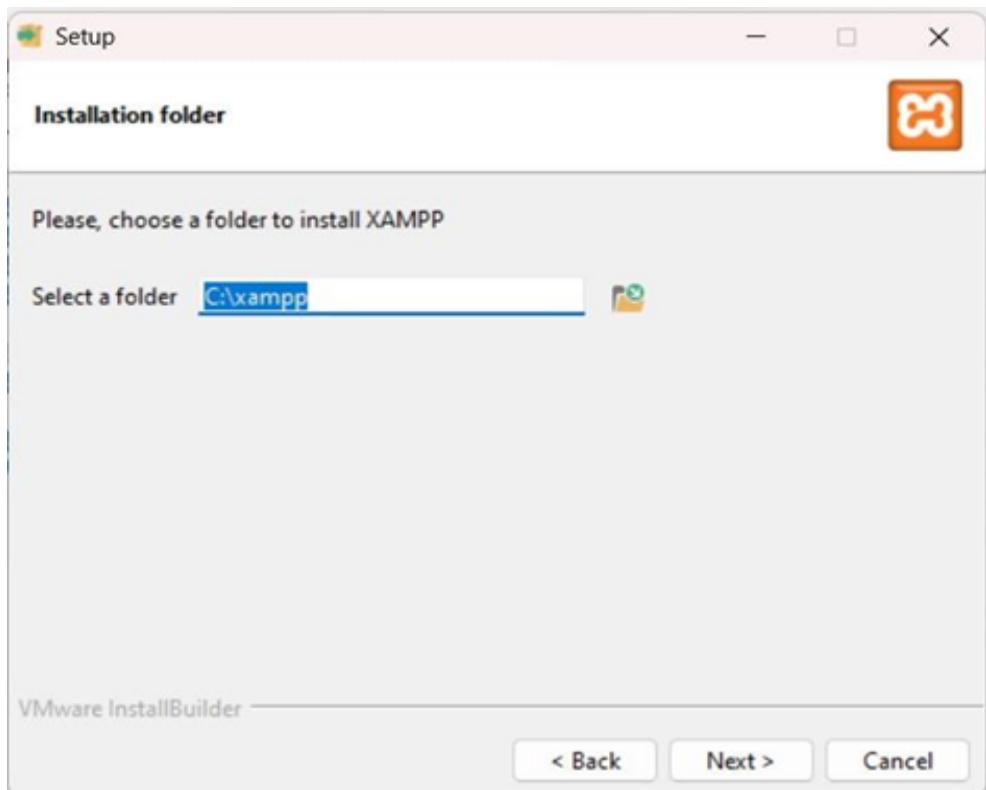
- 1) Select your OS. It will automatically start downloading.



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



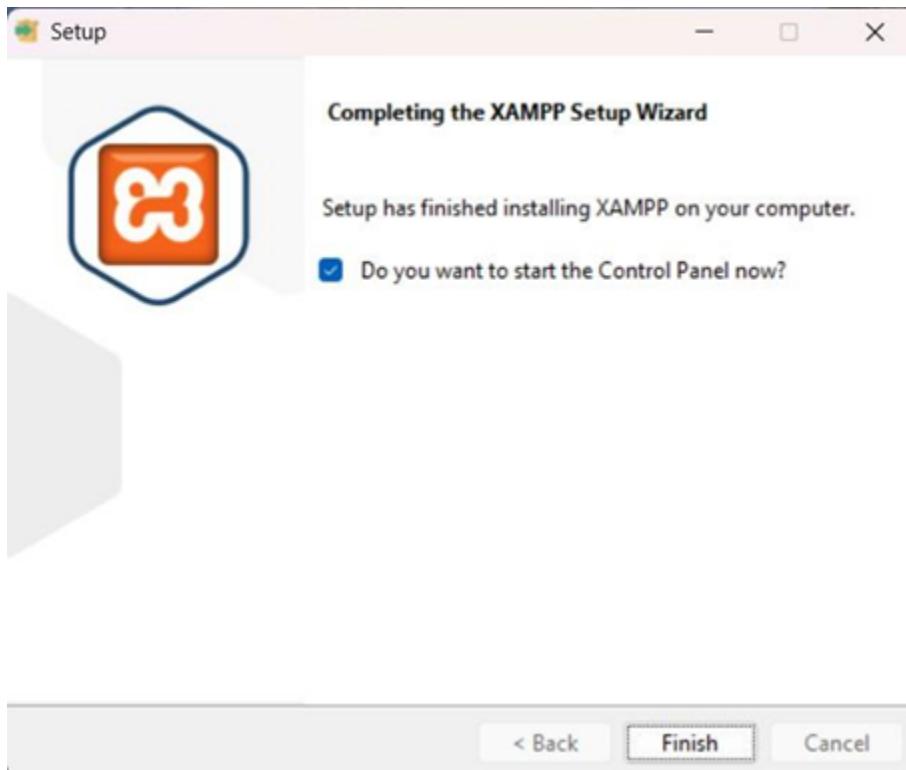
- 3) Choose the folder to install XAMPP in. Make sure the folder is empty. Click next



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



- 5) The installation is complete. Click Finish



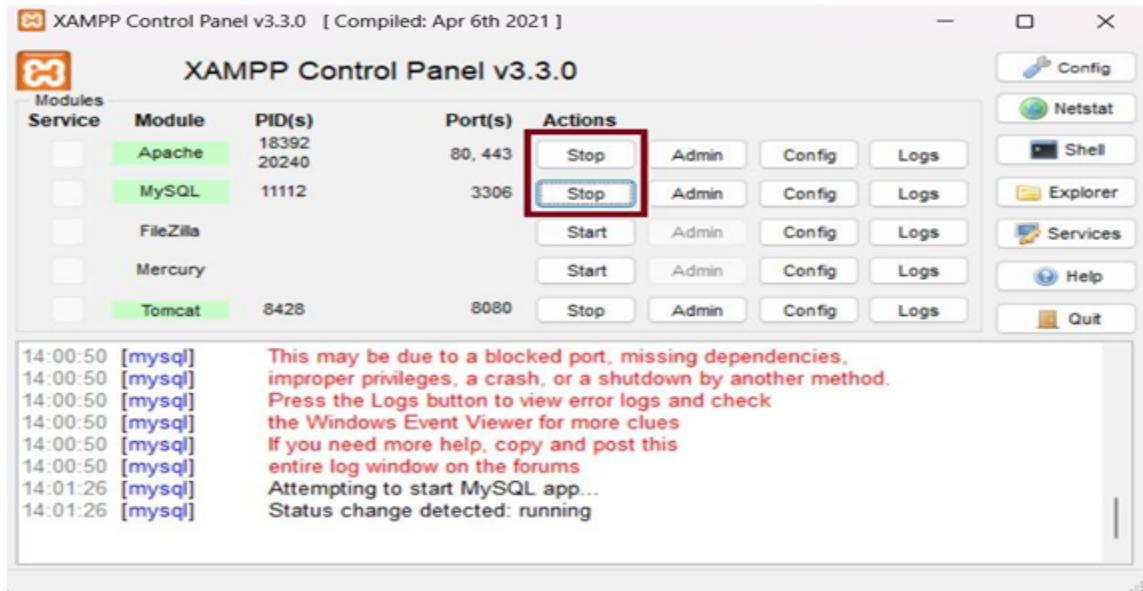
Step 2: Setup a file that is to be hosted on the server. Make sure the file has extension .php

test1	06-08-2024 22:48	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.

Name	Date modified	Type	Size
dashboard	06-08-2024 20:42	File folder	
img	06-08-2024 20:42	File folder	
webalizer	06-08-2024 20:42	File folder	
xampp	06-08-2024 22:44	File folder	
applications	15-06-2022 21:37	Chrome HTML Do...	4 KB
bitnami	15-06-2022 21:37	CSS Source File	1 KB
favicon.ico	16-07-2015 21:02	ICO File	31 KB
index	16-07-2015 21:02	PHP Source File	1 KB
test1	06-08-2024 22:48	PHP Source File	1 KB
text	06-08-2024 22:23	PHP Source File	1 KB

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

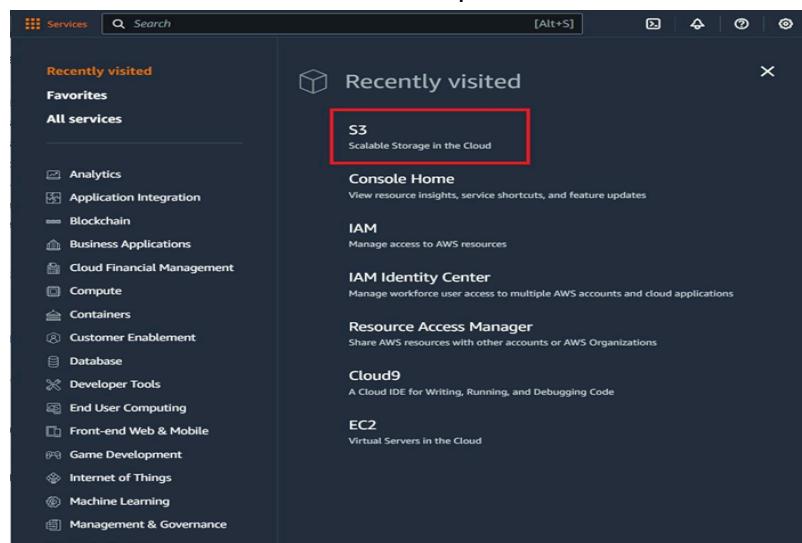


Step 5: Open your web browser. Type localhost/YOUR_FILENAME.php. This will open your website on your browser.

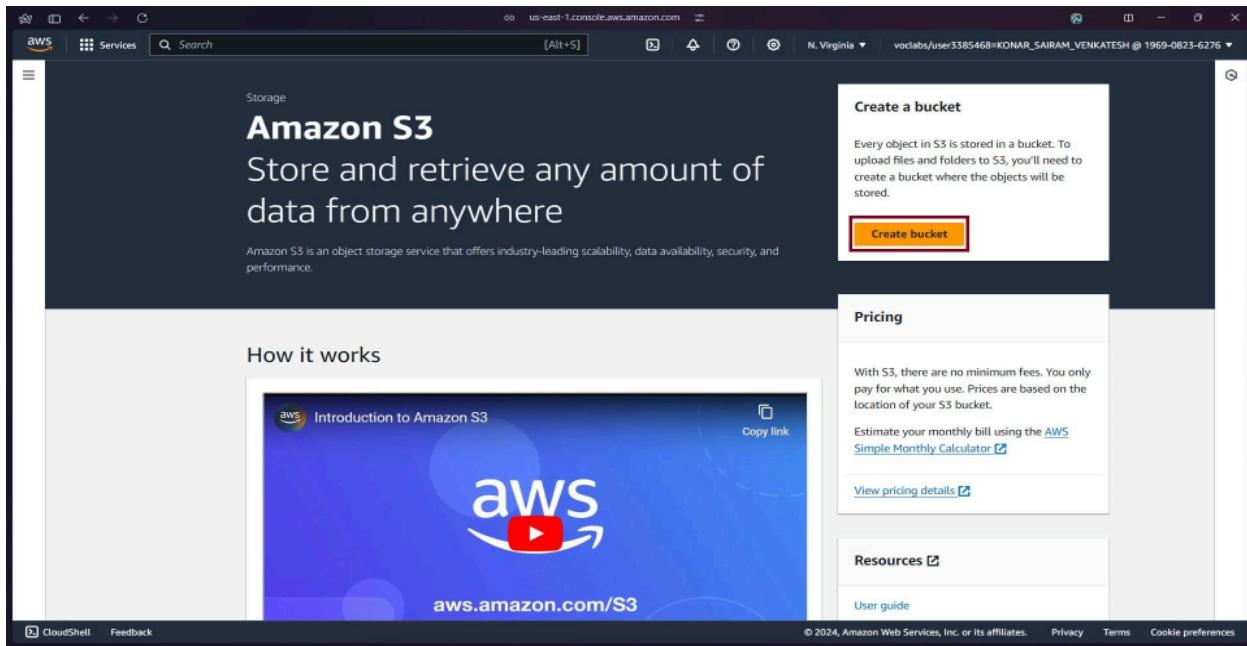


2) AWS S3

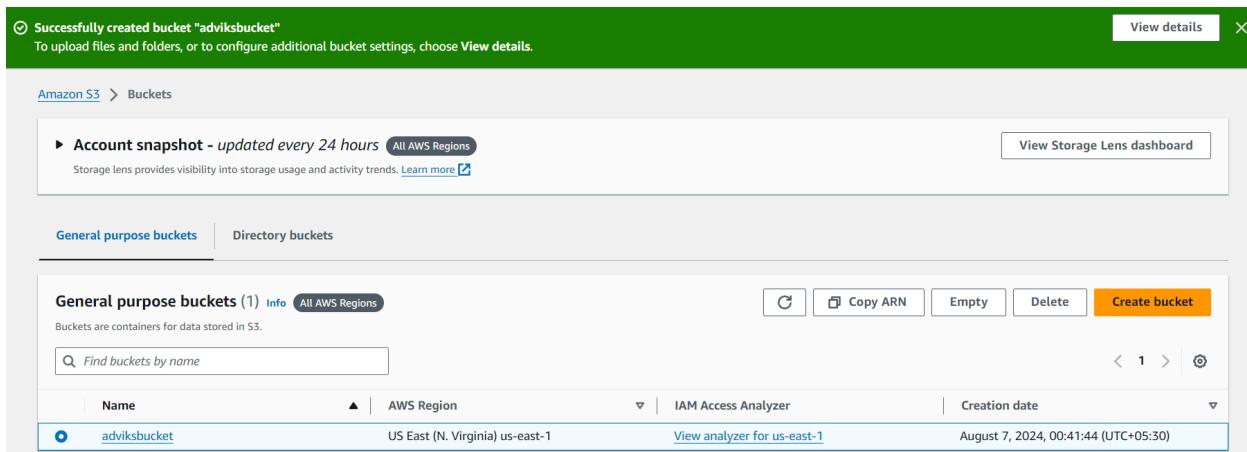
Step 1: Login to your AWS account. Go to services and open S3.



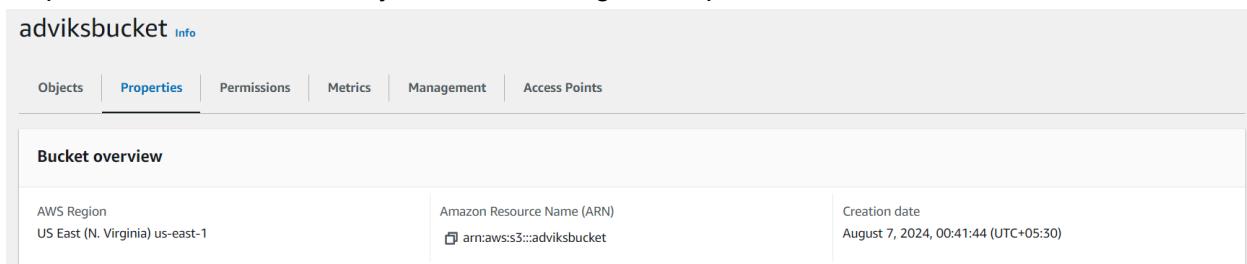
Step 2: Click on Create Bucket



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



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

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

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.

Hosting type

Host a static website
Use the bucket endpoint as the web address. [Learn more](#)

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

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

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

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

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

Amazon S3 > Buckets > statichosting27

statichosting27 [Info](#)

[Objects](#) [Properties](#) [Permissions](#) [Metrics](#) [Management](#) [Access Points](#)

Objects (0) [Info](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

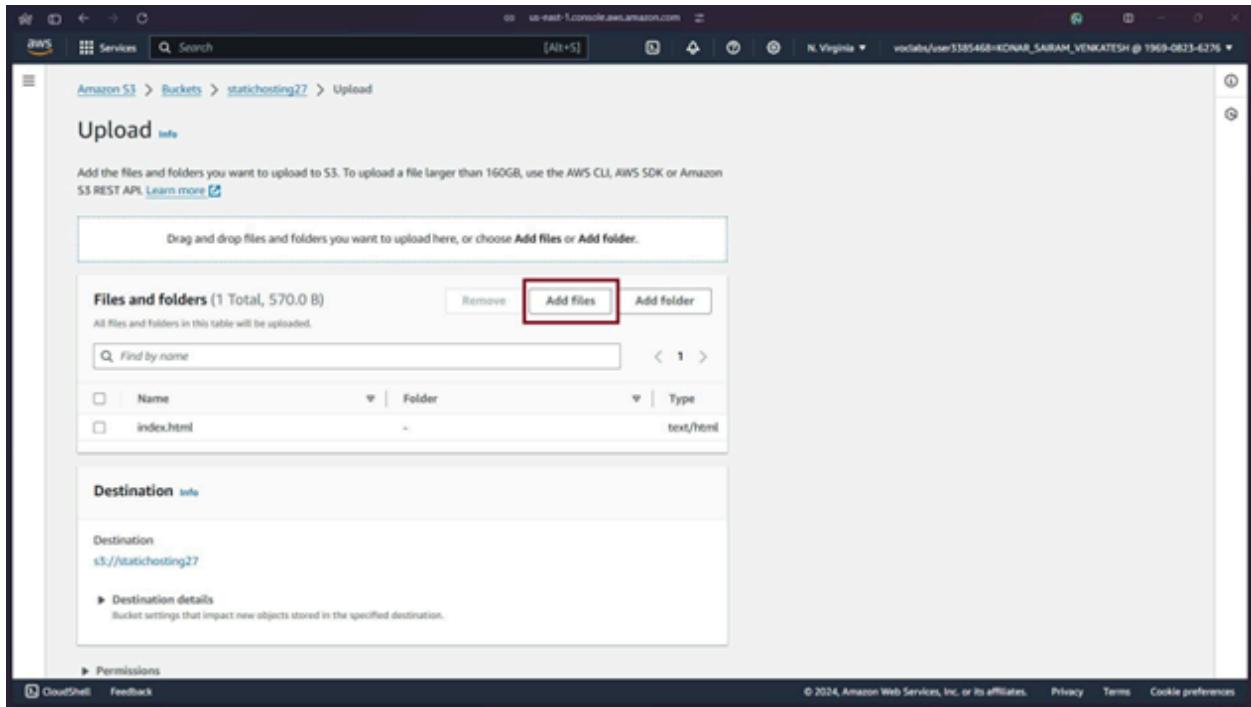
[Actions](#) [Create folder](#) [Upload](#)

Name	Type	Last modified	Size	Storage class
No objects				

You don't have any objects in this bucket.

[Upload](#)

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, scroll down to Static web 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://adviksbucket.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 the public users. To change this, go to Permissions tab, go to Block public access and click on edit

403 Forbidden

- Code: AccessDenied
- Message: Access Denied
- RequestId: QFWZ19ZNK2CSHN24
- HostId: 5tvUKU1Jr0bR7xeXCiKQ2O89qapq//OQxXenu/o9umWWNH2blrLSZ11Sa3h0JMqG3CLAkZkB8U=

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 access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

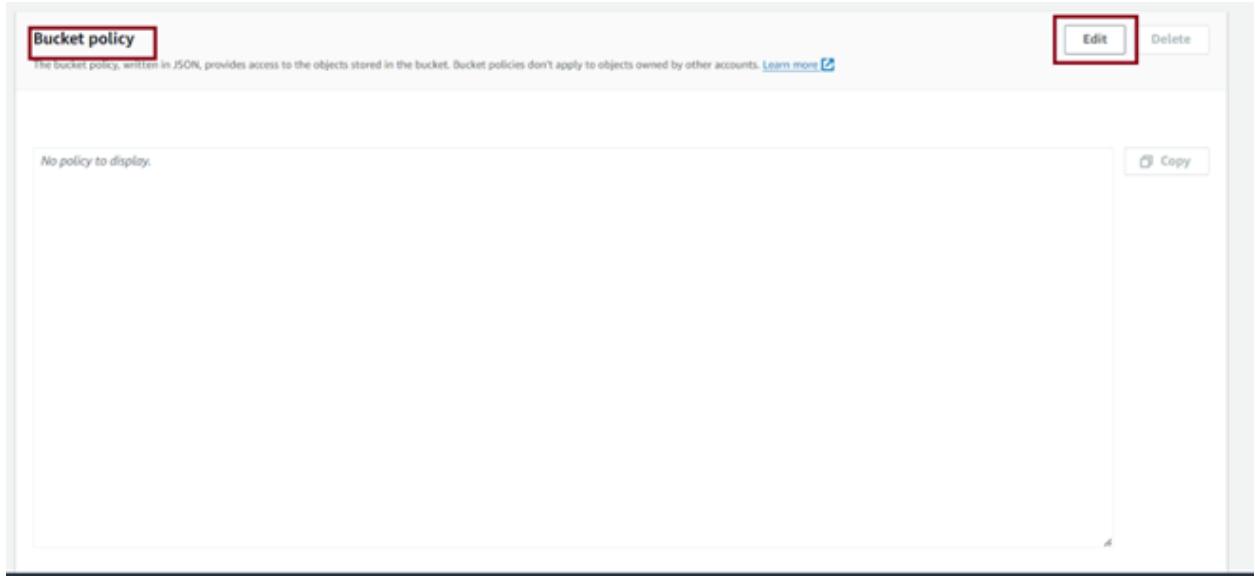
Block all public access

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

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

Cancel **Save changes**

Step 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/*"  
    }  
  ]  
}
```

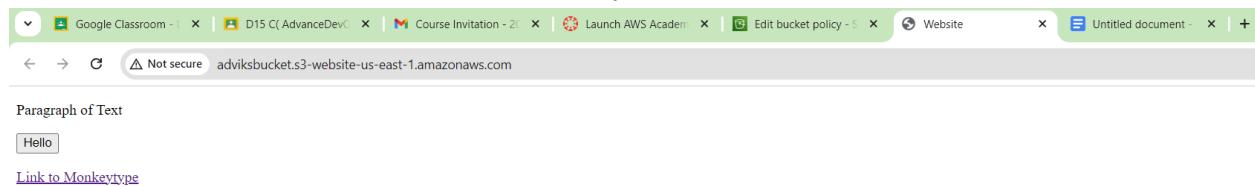
Paste this code snippet in the policy textarea. Replace YOUR-BUCKET-NAME-HERE with the name you have given to your bucket. Save the changes.

Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bu

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

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



Name:Advik Hegde

Div/Roll No. : D15C/15

Exp 1b: Cloud9 Setup and Launch, Collaboration demonstration by creation of IAM groups and users.

Open your AWS account and search for Cloud9 service inside Developer tools. Create a new Cloud9 environment by filling in the required details. Make sure you use an EC2 instance to create your environment.

Create environment Info

Details

Name
AdviksCloud9
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.

<https://us-east-1.console.aws.amazon.com/cloud9control/home?region=us-east-1#/create/>

New EC2 instance

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

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

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

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

Additional instance types
Explore additional instances to fit your need.

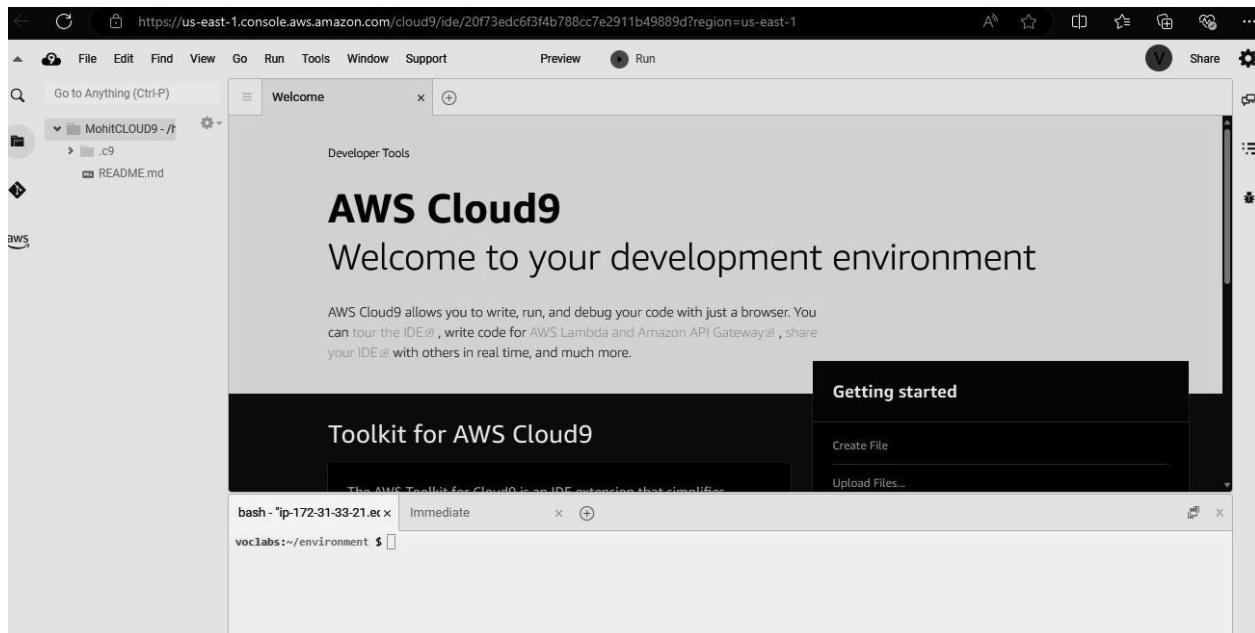
Platform Info
This will be installed on your EC2 instance. We recommend Amazon Linux 2023.

Amazon Linux 2023

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

30 minutes

We have successfully set up and launched our Cloud9 environment. Over here, we can build and develop programs as per our desire. We are also allowed to collaborate with multiple other users and access shared resources.



Moving on, we are supposed to create a new user. Give a suitable name to the user and decide the password for the same.

Specify user details

User details

User name
The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

Provide user access to the AWS Management Console - optional
If you're providing console access to a person, it's a best practice [to manage their access in IAM Identity Center](#).

Console password
 Autogenerated password
You can view the password after you create the user.
 Custom password
Enter a custom password for the user.

- Must be at least 8 characters long
- Must include at least three of the following mix of character types: uppercase letters (A-Z), lowercase letters (a-z), numbers (0-9), and symbols ! @ # \$ % ^ & * () _ + - (hyphen) = [] { } | `

Show password

Users must create a new password at next sign-in - Recommended
Users automatically get the [IAMUserChangePassword](#) policy to allow them to change their own password.

Create user group

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

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

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

Permissions policies (951)

Filter by Type				
<input type="text"/>	All ty... ▾	< 1 2 3 4 5 6 7 ... 48 >	<input type="button" value="Create policy"/>	
<input type="checkbox"/> Policy name	Type	Use...	Description	
<input type="checkbox"/> AdministratorAccess	AWS managed ...	Permis...	Provides full access to AWS services	
<input type="checkbox"/> AdministratorAcce...	AWS managed	None	Grants account administrative perm	
<input type="checkbox"/> AdministratorAcce...	AWS managed	None	Grants account administrative perm	

Similarly, create a new group and provide a suitable name for them. Include the IAM users in this group together for our convenience, that is, to provide similar kinds of permissions to the entire group rather than an individual user.

https://us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#users/create

MSBCLLOUD9 user group created.

[Review and create](#)

Step 4
Retrieve password

Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.

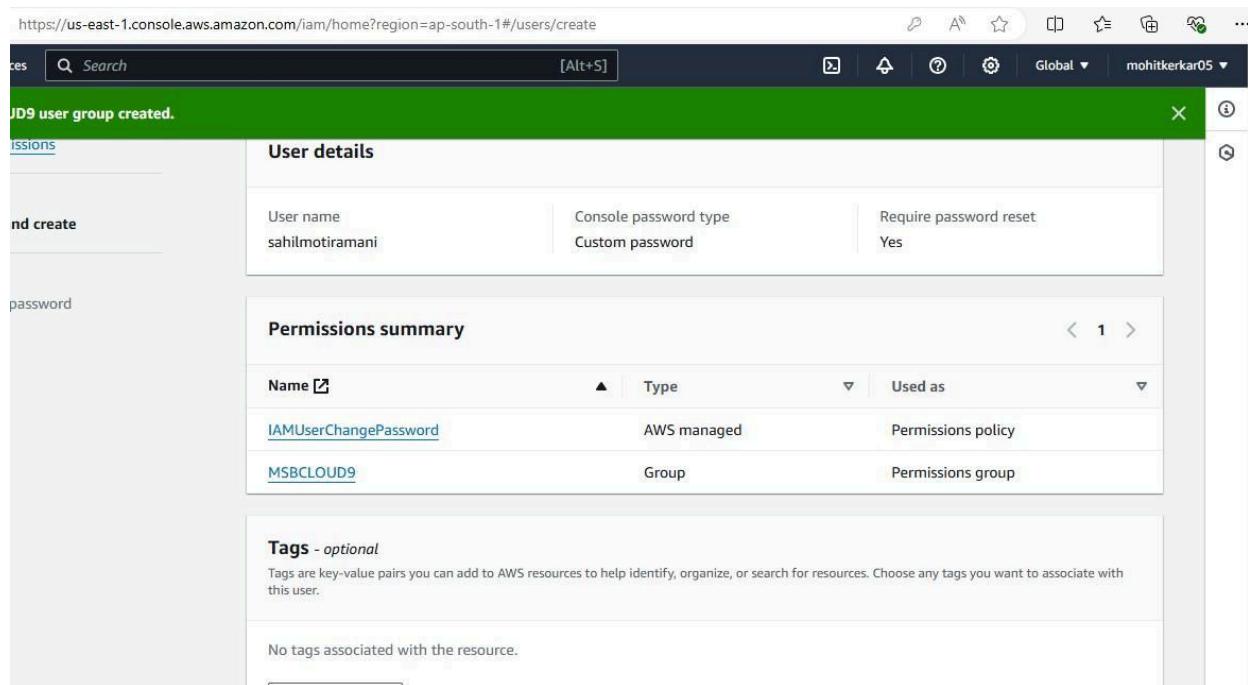
Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

User groups (1/1)

Filter by Type				
<input type="checkbox"/> Group name	Users	Attached policies	Created	
<input checked="" type="checkbox"/> MSBCLLOUD9	0	-	2024-07-29 (Now)	

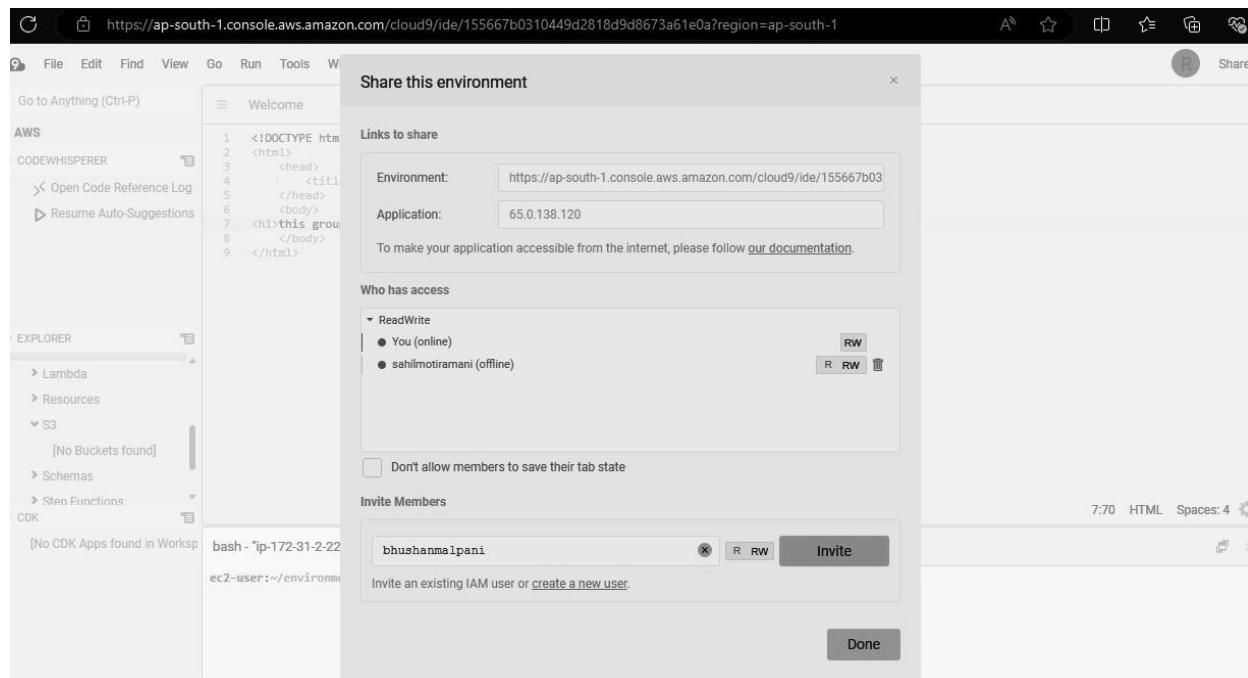
Set permissions boundary - optional

The user has successfully been created i.e. There is a custom-made username and a password for the IAM user.

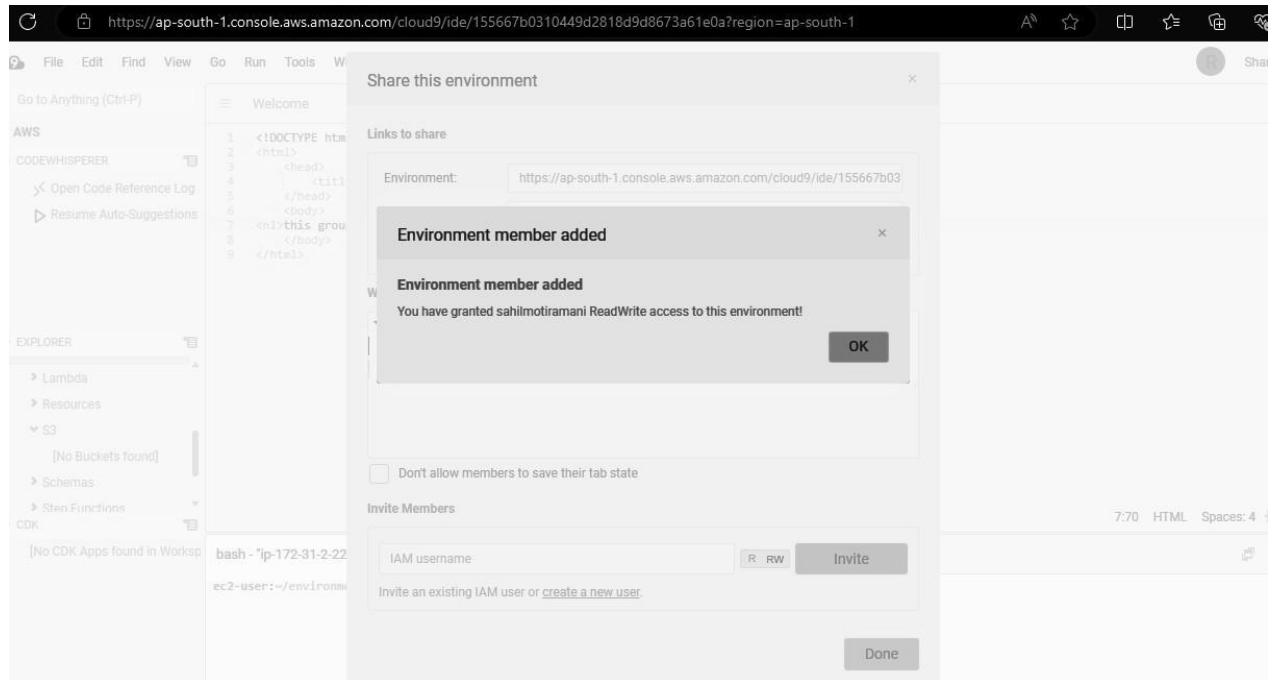


A screenshot of the AWS IAM User Creation page. At the top, a green banner says "JD9 user group created." Below it, the "User details" section shows a user named "sahilmotiramani" with a "Custom password" type and "Yes" for "Require password reset". The "Permissions summary" section lists two entries: "IAMUserChangePassword" (AWS managed, Permissions policy) and "MSBCLOUD9" (Group, Permissions group). The "Tags - optional" section indicates "No tags associated with the resource."

Go back to the cloud9 environment. Click on the share this environment option so as to allow other collaborators to access your environment. Include your newly made IAM user in this environment and enable Read/Write permissions for it.



A screenshot of the AWS Cloud9 "Share this environment" dialog. It shows "Links to share" with an "Environment" link to "https://ap-south-1.console.aws.amazon.com/cloud9/ide/155667b0310449d2818d9d8673a61e0a?region=ap-south-1" and an "Application" link to "65.0.138.120". Under "Who has access", "You (online)" and "sahilmotiramani (offline)" are listed with "ReadWrite" permissions. A checkbox for "Don't allow members to save their tab state" is unchecked. In the "Invite Members" section, "bhushanmalpani" is listed with "R RW" permissions, and an "Invite" button is present. A "Done" button is at the bottom right.



Further, we are supposed to login from another browser using the credentials of the IAM user, to access the shared cloud9 environment with us.

These steps could not be completed because Cloud9 services have been disrupted and there is no access to the IAM user from the remote login.

Name:Advik Hegde

Div/Roll No. : D15C/15

Exp 02:To Build Your Application using AWS Code Build and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS Code Deploy.

Step 1: Create our Elastic Beanstalk Environment

Login into your AWS account and navigate to services. Search for Elastic Beanstalk service and click on create application. Give your application a suitable name. For the platform, select PHP. The rest of the configuration settings are to be kept as default.

Application information Info

Application name
AdviksBeanstalk
Maximum length of 100 characters.

► Application tags (optional)

Environment information Info

Choose the name, subdomain and description for your environment. These cannot be changed later.

Environment name
AdviksBeanstalk-env
Must be from 4 to 40 characters in length. The name can contain only letters, numbers, and hyphens. It can't start or end with a hyphen. This name must be unique within a region in your account.

The screenshot shows the 'Create Environment' wizard in the AWS Elastic Beanstalk console. The 'Application information' step is completed with the application name 'AdviksBeanstalk'. The 'Environment information' step is also completed with the environment name 'AdviksBeanstalk-env'. The 'Platform' dropdown is set to 'PHP'. The 'Application code' section shows the 'Sample application' option selected. The browser address bar shows the URL: <https://ap-south-1.console.aws.amazon.com/elasticbeanstalk/home?region=ap-south-1#/create-environment?applicationName=AdviksBeanstalk>.

Now, while creating the environment, we are asked to provide an IAM role with the necessary EC2 permissions. We are supposed to make sure that we have made an existing IAM role with the following set of permissions:

1. AWSElasticBeanstalkWebTier
2. AWSElasticBeanstalkWorkerTier
3. AWSElasticBeanstalkMulticontainerDocker

We can skip the steps to follow after the initial few steps mentioned above and move straight to review the settings of our environment. After reviewing everything properly, our environment can successfully be created.

The screenshot shows the AWS Elastic Beanstalk console. The left sidebar shows the navigation path: AWS Services > Elastic Beanstalk > Environments > AdviksBeanstalk-env-2. The main content area displays the environment overview for 'AdviksBeanstalk-env-2'. It includes sections for Environment overview (Health: Warning, Domain: advik.ap-south-1.elasticbeanstalk.com), Platform (Platform: PHP 8.3 running on 64bit Amazon Linux 2023/4.3.2, Running version: -), and Events (10). A success message at the top says 'Environment successfully launched.'

Step 2: Fork the required repository onto our github account

The repository to be forked is - imoisharma/aws-codepipeline-s3-codeddeploy-linux-2.0

The screenshot shows the GitHub repository page for 'aws-codepipeline-s3-codeddeploy-linux-2.0' owned by 'imoisharma'. The repository is public and has 20 commits. A large 'Fork' button is visible in the top right corner. The repository details include a description: 'Use this sample when creating a simple pipeline in AWS CodePipeline while following the Simple Pipeline Walkthrough tutorial.' The commit history lists several files like README.md, .github, dist, scripts, CODE_OF_CONDUCT.md, CONTRIBUTING.md, LICENSE, and README.md.

The screenshot shows the GitHub forked repository page for 'aws-codepipeline-s3-codeddeploy-linux-2.0' owned by 'AdvikHegde-VES'. It is a fork of the original repository from 'imoisharma'. The repository details and commit history are identical to the original, showing the same files and commit history. A note at the top of the repository page states: 'This branch is up to date with imoisharma/aws-codepipeline-s3-codeddeploy-linux-2.0:master.'

This step is necessary for the execution of the steps to follow. It will be helpful in the creation of a pipeline.

Step 3: Creation of the Pipeline

Navigate to Codepipeline inside Developer Tools. Give a suitable name to the pipeline you want to create.

The screenshot shows the 'Choose pipeline settings' step of the AWS CodePipeline creation wizard. On the left, a sidebar lists steps from 'Step 1 Choose pipeline settings' to 'Step 5 Review'. The main area is titled 'Pipeline settings'. It shows the 'Pipeline name' field containing 'AdviksPipeline'. A note below it says 'No more than 100 characters'. Under 'Pipeline type', a note says '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.' The 'Execution mode' section shows 'Queued (Pipeline type V2 required)' selected, with a note explaining that executions are processed one by one.

And click on next ...

The screenshot shows the 'Add source stage' step of the AWS CodePipeline creation wizard. The sidebar shows steps from 'Step 2 Add source stage' to 'Step 5 Review'. The main area is titled 'Source'. It shows the 'Source provider' dropdown set to 'GitHub (Version 2)'. A note below it says 'New GitHub version 2 (app-based action)'. The 'Connection' section has a search bar and a 'Connecting' button. The 'Repository name' section has a search bar. The 'Default branch' section has a search bar.

Step 4: GitHub connection

In this step, we are supposed to create a GitHub connection and add our existing repository over here i.e. the one we forked earlier.

Source provider

This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.

GitHub (Version 2) ▾



New GitHub version 2 (app-based) action

To add a GitHub version 2 action in CodePipeline, you create a connection, which uses GitHub Apps to access your repository. Use the options below to choose an existing connection or create a new one. [Learn more](#)

Connection

Choose an existing connection that you have already configured, or create a new one and then return to this task.



arn:aws:codeconnections:ap-south-1:221082173765:connection/479335fc-9 X

or

[Connect to GitHub](#)



Ready to connect

Your GitHub connection is ready for use.

Repository name

Choose a repository in your GitHub account.



AdvikHegde-VES/aws-codepipeline-s3-codedeploy-linux-2.0 X

You can type or paste the group path to any project that the provided credentials can access. Use the format 'group/subgroup/project'.

Default branch

Default branch will be used only when pipeline execution starts from a different source or manually started.



master X

We are supposed to enter our GitHub username so as to proceed towards making the connection.

aws Services More ▾

Developer Tools > ... > Create connection

Create a connection Info

Create GitHub App connection Info

Connection name

▶ Tags - *optional*

Connect to GitHub

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

Now to finalize our connection, we are to install an application which connects AWS to our GitHub account and repository.

Post the establishment of the connection, this is the message that is displayed. We can further select the branch of our repository that we want to connect.

Source provider

This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.

GitHub (Version 2) ▾



New GitHub version 2 (app-based) action

To add a GitHub version 2 action in CodePipeline, you create a connection, which uses GitHub Apps to access your repository. Use the options below to choose an existing connection or create a new one. [Learn more](#)

Connection

Choose an existing connection that you have already configured, or create a new one and then return to this task.

arn:aws:codeconnections:ap-south-1:221082173765:connection/479335fc-9 X or [Connect to GitHub](#)



Ready to connect

Your GitHub connection is ready for use.

Repository name

Choose a repository in your GitHub account.

AdvikHegde-VES/aws-codepipeline-s3-codedeploy-linux-2.0 X

You can type or paste the group path to any project that the provided credentials can access. Use the format 'group/subgroup/project'.

Default branch

Default branch will be used only when pipeline execution starts from a different source or manually started.

master X

Step 5: Deployment stage:

We are expected to skip the build stage and move towards the deployment step. In the deployment step we are supposed to choose the Elastic Beanstalk application and the environment that we created earlier and proceed with our pipeline creation.

Deploy

Deploy provider
Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.

AWS Elastic Beanstalk ▾

Region
Asia Pacific (Mumbai) ▾

Input artifacts
Choose an input artifact for this action. [Learn more](#)

No more than 100 characters ▾

Application name
Choose an application that you have already created in the AWS Elastic Beanstalk console. Or create an application in the AWS Elastic Beanstalk console and then return to this task.

AdviksBeanStalk X

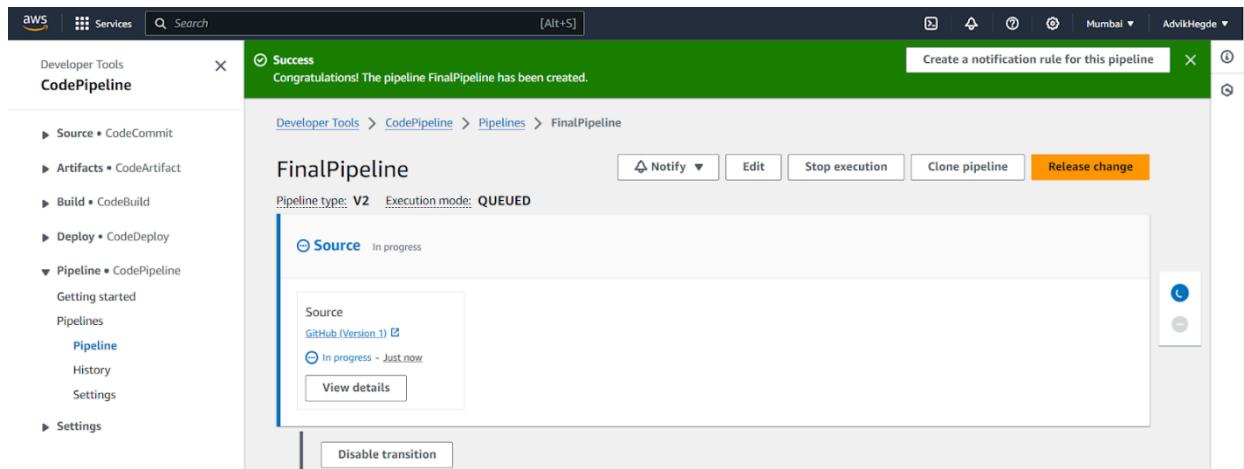
Environment name
Choose an environment that you have already created in the AWS Elastic Beanstalk console. Or create an environment in the AWS Elastic Beanstalk console and then return to this task.

AdviksBeanstalk-env-2 X

Configure automatic rollback on stage failure

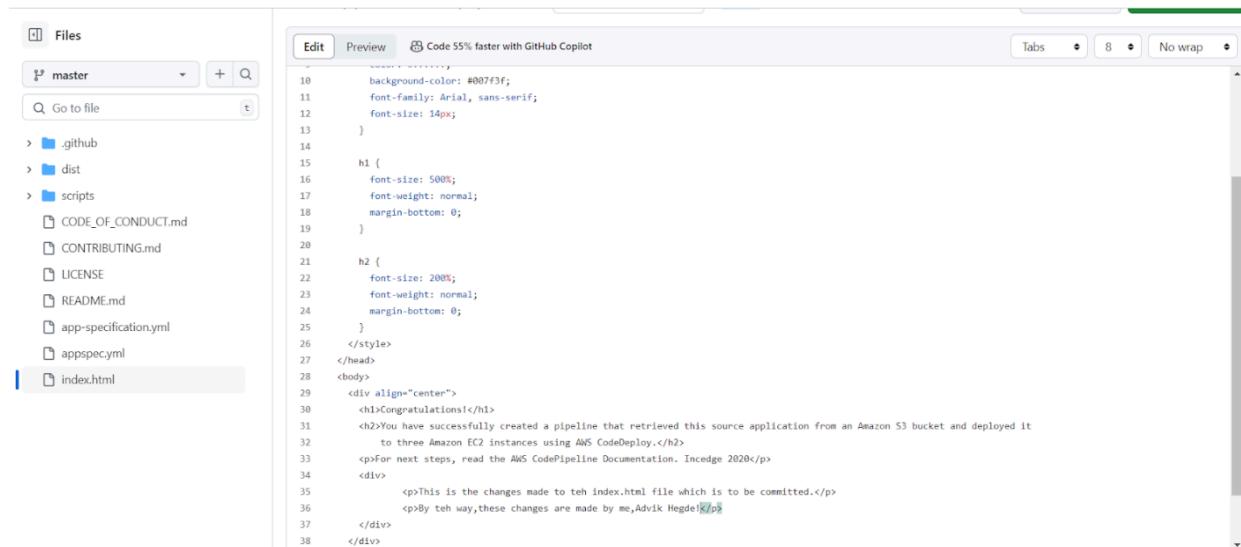
Cancel Previous Next

Step 6: Post deployment stage: When all the stages run successfully, this is what is displayed onto the screen. It shows us that our application and our environment have successfully been deployed using a dedicated pipeline created.



Step 7: Committing changes to your GitHub code

Now, we will go to our forked repository and make some changes to the index.html file. On making the desired changes, we are supposed to commit those changes on our forked repository. Write a good commit message so as to recognize it when it appears on the pipeline.



```
background-color: #007f3f;
font-family: Arial, sans-serif;
font-size: 14px;
}

h1 {
    font-size: 500px;
    font-weight: normal;
    margin-bottom: 0;
}

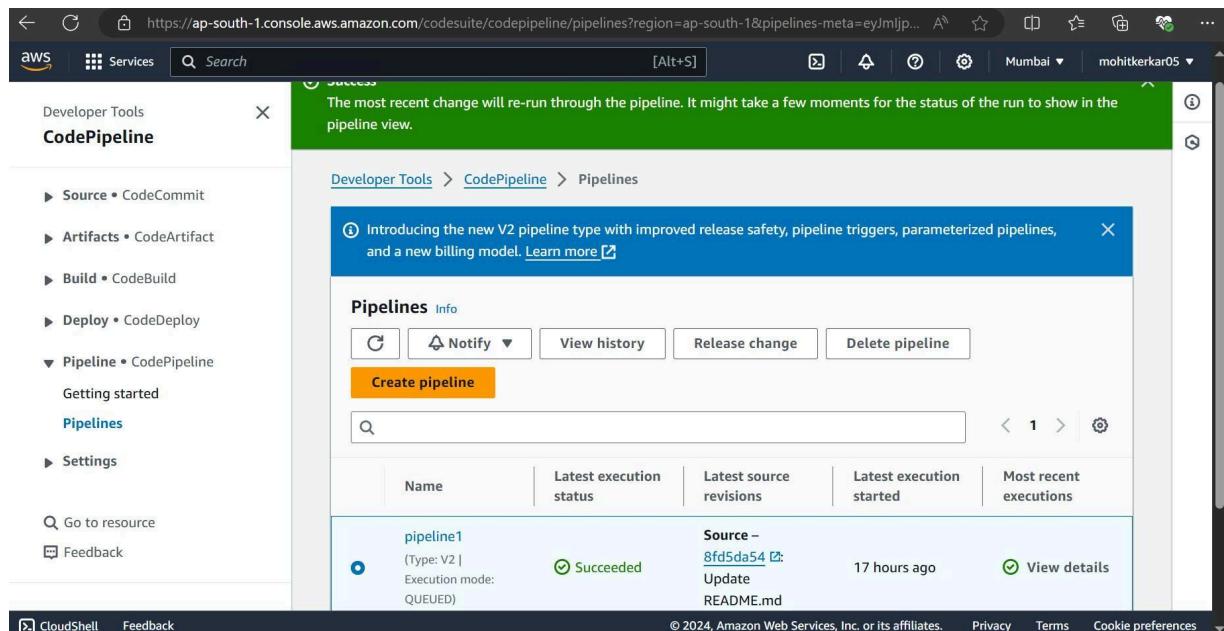
h2 {
    font-size: 200px;
    font-weight: normal;
    margin-bottom: 0;
}

</style>
</head>
<body>
<div align="center">
<h1>Congratulations!</h1>
<h2>You have successfully created a pipeline that retrieved this source application from an Amazon S3 bucket and deployed it to three Amazon EC2 instances using AWS CodeDeploy.</h2>
<p>For next steps, read the AWS CodePipeline Documentation. Incide 2020</p>
<div>
<p>This is the changes made to teh index.html file which is to be committed.</p>
<p>By teh way,these changes are made by me,Advik Hegde</p>
</div>
</div>

```

Step 8: Apply the newly made changes in index.html onto our pipeline

Come back to the Codepipeline section and select the pipeline through which we successfully created and deployed our application. Click on the release change option to apply the latest changes/commits from our GitHub repository to our pipeline.



The most recent change will re-run through the pipeline. It might take a few moments for the status of the run to show in the pipeline view.

Developer Tools > CodePipeline > Pipelines

Introducing the new V2 pipeline type with improved release safety, pipeline triggers, parameterized pipelines, and a new billing model. [Learn more](#)

Name	Latest execution status	Latest source revisions	Latest execution started	Most recent executions
pipeline1 (Type: V2 Execution mode: QUEUED)	Succeeded	Source - 8fd5da54 Update README.md	17 hours ago	View details

Once the changes have been applied, we see the commit message that we wrote for the latest commit on our repository being reflected on our pipeline. Over here, it would be seen somewhere near the bottom of the image that is attached. “Update index.html” was the latest commit message in the GitHub repository.

The screenshot shows the AWS CodePipeline console. At the top, there are two green success notifications: one stating 'Congratulations! The pipeline FinalPipeline has been created.' and another stating 'The most recent change will re-run through the pipeline. It might take a few moments for the status of the run to show in the pipeline view.' Below these, the pipeline's navigation path is shown: Developer Tools > CodePipeline > Pipelines > FinalPipeline. The pipeline itself is titled 'FinalPipeline' and is set to 'Pipeline type: V2' with 'Execution mode: QUEUED'. A green 'Source' step is listed, which has succeeded and is associated with a GitHub commit ID: 3cf895ae. A 'View details' button is available for this step. To the right of the pipeline list, there are buttons for 'Notify', 'Edit', 'Stop execution', 'Clone pipeline', and a prominent orange 'Release change' button. A sidebar on the right contains a green checkmark icon and a blue refresh/circular arrow icon.

Step 9: Open the Domain of our Elastic Beanstalk environment

Now, we navigate back to our Elastic Beanstalk environment and open the environment domain of our deployed application.

The text in this image is clearly distinguishable from the earlier website’s text meaning that the changes that we made to our code in index.html has successfully been applied to the website that we deployed.



Advanced DevOps Lab

Experiment No:3

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

Theory:

Container-based microservices architectures have profoundly changed the way development and operations teams test and deploy modern software. Containers help companies modernize by making it easier to scale and deploy applications, but containers have also introduced new challenges and more complexity by creating an entirely new infrastructure ecosystem.

Large and small software companies alike are now deploying thousands of container instances daily, and that's a complexity of scale they have to manage. So how do they do it?

Enter the age of Kubernetes.

Originally developed by Google, Kubernetes is an open-source container orchestration platform designed to automate the deployment, scaling, and management of containerized applications. In fact, Kubernetes has established itself as the defacto standard for container orchestration and is the flagship project of the Cloud Native Computing Foundation (CNCF), backed by key players like Google, AWS, Microsoft, IBM, Intel, Cisco, and Red Hat.

Kubernetes makes it easy to deploy and operate applications in a microservice architecture. It does so by creating an abstraction layer on top of a group of hosts so that development teams can deploy their applications and let Kubernetes manage the following activities:

- Controlling resource consumption by application or team
- Evenly spreading application load across a hosting infrastructure
- Automatically load balancing requests across the different instances of an application
- Monitoring resource consumption and resource limits to automatically stop applications from consuming too many resources and restarting the applications again
- Moving an application instance from one host to another if there is a shortage of resources in a host, or if the host dies
- Automatically leveraging additional resources made available when a new host is added to the cluster
- Easily performing canary deployments and rollbacks

Steps:

1. Create 3 EC2 Ubuntu Instances on AWS.

(Name 1 as Master, the other 2 as worker-1 and worker-2)

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with options like EC2 Dashboard, EC2 Global View, Events, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, and Reserved Instances. The main area is titled "Instances (1/3) Info" and shows a table of three instances. The columns include Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IP. The instances listed are: node-2 (Running, t2.micro, Initializing, ap-south-1b, ec2-43-20-123-45), master (Running, t2.micro, 2/2 checks passed, ap-south-1b, ec2-35-15-123-45), and node-1 (Running, t2.micro, 2/2 checks passed, ap-south-1b, ec2-13-23-123-45).

2. Edit the Security Group Inbound Rules to allow SSH

The screenshot shows the AWS Security Groups Inbound rules page. It lists three rules under the "Inbound rules" section. Each rule has columns for Security group rule ID, Type, Protocol, Port range, Source, and Description - optional. The rules are: 1. Security group rule ID sgr-01b5ed431d6cab19e, Type SSH, Protocol TCP, Port range 22, Source 0.0.0.0/0, Description - optional. 2. Security group rule ID sgr-0ca2edd1eff92bd48, Type HTTP, Protocol TCP, Port range 80, Source 0.0.0.0/0, Description - optional. 3. Security group rule ID sgr-06652627bdaeacbf0, Type HTTPS, Protocol TCP, Port range 443, Source 0.0.0.0/0, Description - optional. At the bottom left, there's a "Add rule" button.

3. SSH into all 3 machines

a. You can do it through the aws console directly

or

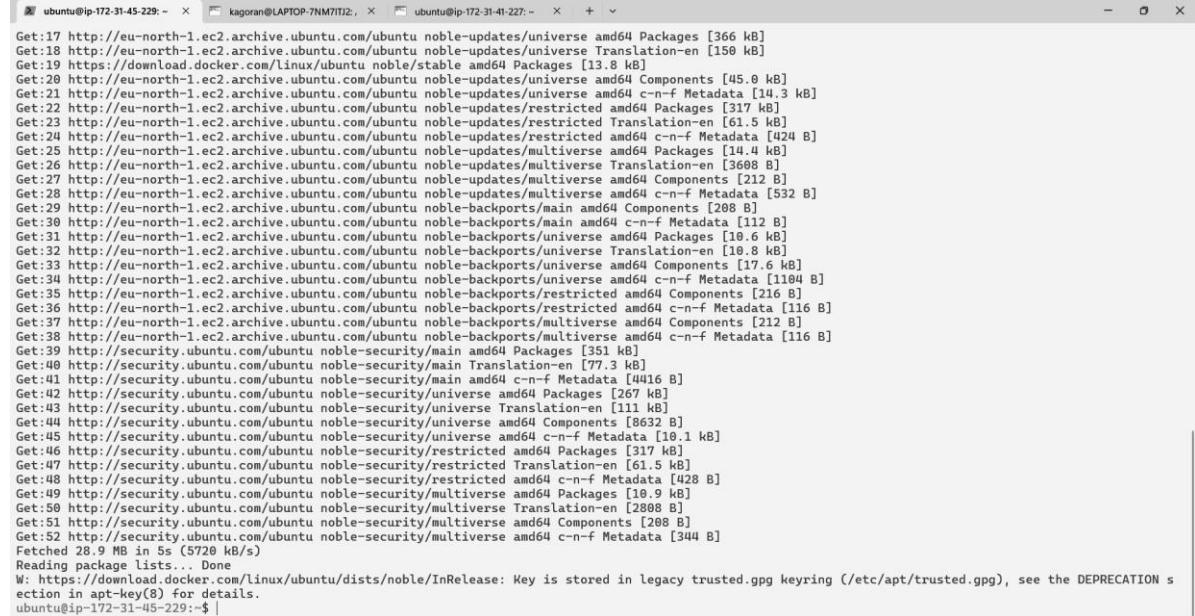
b. Locate your key from the Downloads folder and open it in cmd and paste this command

`ssh -i <-your-key->.pem ec2-user<ip-address of instance>`

3. From now on, until mentioned, perform these steps on all 3 machines.

Install Docker

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
${lsb_release -cs} stable" sudo apt-get update
sudo apt-get install -y docker-ce
```



```
ubuntu@ip-172-31-45-229:~$ kagoran@LAPTOP-7NM7ITIZ:~$ ubuntu@ip-172-31-41-227:~$ + x
Get:17 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [366 kB]
Get:18 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [150 kB]
Get:19 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [13.8 kB]
Get:20 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [45.0 kB]
Get:21 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [14.3 kB]
Get:22 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [237 kB]
Get:23 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [61.5 kB]
Get:24 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 c-n-f Metadata [424 B]
Get:25 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [14.4 kB]
Get:26 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [3608 B]
Get:27 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [212 B]
Get:28 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [532 B]
Get:29 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:30 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [112 B]
Get:31 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [10.6 kB]
Get:32 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [10.8 kB]
Get:33 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [17.6 kB]
Get:34 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1104 B]
Get:35 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:36 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:37 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:38 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:39 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [351 kB]
Get:40 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [77.3 kB]
Get:41 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4416 B]
Get:42 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [267 kB]
Get:43 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [111 kB]
Get:44 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:45 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.1 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [317 kB]
Get:47 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [61.5 kB]
Get:48 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [428 B]
Get:49 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [10.9 kB]
Get:50 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [2808 B]
Get:51 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:52 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [344 B]
Fetched 28.9 MB in 5s (5720 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
ubuntu@ip-172-31-45-229:$
```

Then, configure cgroup in a daemon.json file.

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

Install Kubernetes on all 3 machines

```
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
cat << EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list deb
https://apt.kubernetes.io/ kubernetes-xenial main EOF sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl

ubuntu@ip-172-31-40-255:~$ # Add Kubernetes GPG key
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -

# Add Kubernetes repository
sudo tee /etc/apt/sources.list.d/kubernetes.list <<EOF
deb https://apt.kubernetes.io/ kubernetes-xenial main
EOF

# Update package list
sudo apt-get update

# Install kubelet, kubeadm, and kubectl
sudo apt-get install -y kubelet kubeadm kubectl

# Hold the versions of Kubernetes components
sudo apt-mark hold kubelet kubeadm kubectl
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
deb https://apt.kubernetes.io/ kubernetes-xenial main
Hit:1 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Ign:6 https://packages.cloud.google.com/apt kubernetes-xenial InRelease
Err:7 https://packages.cloud.google.com/apt kubernetes-xenial Release
```

After installing Kubernetes, we need to configure internet options to allow bridging.

```
sudo swapoff -a
echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf sudo sysctl -p

ubuntu@ip-172-31-45-229:~$ # Disable swap
sudo swapoff -a

# Allow bridging for iptables
echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf

# Apply sysctl changes
sudo sysctl -p
net.bridge.bridge-nf-call-iptables=1
net.bridge.bridge-nf-call-iptables = 1
ubuntu@ip-172-31-45-229:~$ █
```

4. Perform this **ONLY** on the Master machine

Initialize the Kubecluster

```
sudo kubeadm init --pod-network-cidr=10.244.0.0/16
--ignore-preflight-errors=all
```

```
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 172.31.45.229:6443 --token s9zq75.bsi7js5f62ridulc \
    --discovery-token-ca-cert-hash sha256:91eae090fdd49337bf70d5bf7478e60bc85820d0996651871129a082db6fa8f1
ubuntu@ip-172-31-45-229:~$ █
```

Copy the join command and keep it in a notepad, we'll need it later.

Copy the mkdir and chown commands from the top and execute them

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

Then, add a common networking plugin called flammel file as mentioned in the code.

```
kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
```

```
ubuntu@ip-172-31-45-229:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
```

Check the created pod using this command

Now, keep a watch on all nodes using the following command

```
watch kubectl get nodes
```

5. Perform this **ONLY on the worker machines**

```
sudo kubeadm join <ip> --token <token> \
--discovery-token-ca-cert-hash <hash>
```

Now, notice the changes on the master terminal

```
Every 2.0s: kubectl get nodes                                         ip-172-31-45-229: Sat Sep 14 12:19:42 20
24
NAME           STATUS    ROLES      AGE   VERSION
ip-172-31-45-229   Ready    control-plane   28m   v1.31.1
```

That's it, we now have a Kubernetes cluster running across 3 AWS EC2 Instances. This cluster can be used to further deploy applications and their loads being distributed across these machines.

Conclusion:

In this setup, we established a Kubernetes cluster utilizing three AWS EC2 instances, with Docker and Kubernetes components successfully deployed on each. While the master node is operational and the initial setup was completed, the worker nodes have experienced difficulties joining the cluster, which appear to be related to configuration or networking issues. To finalize the cluster setup and ensure its functionality, additional troubleshooting on the worker nodes is required to address these connectivity problems. Once these issues are resolved, the cluster will be fully functional, enabling efficient management and scaling of containerized applications across the instances.

Advanced DevOps Lab

Experiment 4

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

Theory:

Originally developed by Google, Kubernetes is an open-source container orchestration platform designed to automate the deployment, scaling, and management of containerized applications. In fact, Kubernetes has established itself as the de facto standard for container orchestration and is the flagship project of the Cloud Native Computing Foundation (CNCF), backed by key players like Google, AWS, Microsoft, IBM, Intel, Cisco, and Red Hat.

Kubernetes Deployment

A Kubernetes Deployment is used to tell Kubernetes how to create or modify instances of the pods that hold a containerized application. Deployments can scale the number of replica pods, enable the rollout of updated code in a controlled manner, or roll back to an earlier deployment version if necessary.

Steps:

1. Create an EC2 Ubuntu Instance on AWS.

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
<input type="checkbox"/>	Master	i-02e41e4eb63bbe589	Running	t3.micro	3/3 checks passed		eu-north-1b

2. Edit the Security Group Inbound Rules to allow SSH

Inbound rules							
Security group rule ID	Type	Protocol	Port range	Source	Description - optional		
sgr-01b5ed431d6cab19e	SSH	TCP	22	Custom	0.0.0.0/0		
sgr-0ca2edd1eff92bd48	HTTP	TCP	80	Custom	0.0.0.0/0		
sgr-06652627bdaecabf0	HTTPS	TCP	443	Custom	0.0.0.0/0		

3. SSH into the machine

```
ssh -i <keyname>.pem ubuntu@<public_ip_address>
```

4. Install Docker

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable" sudo apt-get update
sudo apt-get install -y docker-ce
```

```
Get:17 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [366 kB]
Get:18 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [150 kB]
Get:19 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [13.8 kB]
Get:20 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [45.0 kB]
Get:21 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [14.3 kB]
Get:22 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [317 kB]
Get:23 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [61.5 kB]
Get:24 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 c-n-f Metadata [424 kB]
Get:25 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [14.4 kB]
Get:26 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [3608 B]
Get:27 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [212 kB]
Get:28 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [532 B]
Get:29 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:30 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [112 B]
Get:31 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [10.6 kB]
Get:32 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [10.8 kB]
Get:33 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [17.6 kB]
Get:34 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1104 B]
Get:35 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 kB]
Get:36 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:37 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 kB]
Get:38 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:39 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [351 kB]
Get:40 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [77.3 kB]
Get:41 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4416 B]
Get:42 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [267 kB]
Get:43 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [111 kB]
Get:44 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:45 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.1 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [317 kB]
Get:47 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [61.5 kB]
Get:48 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [428 B]
Get:49 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [10.9 kB]
Get:50 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [2808 B]
Get:51 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:52 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [344 B]
Fetched 28.9 MB in 5s (5720 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
ubuntu@ip-172-31-45-229:~$
```

Then, configure cgroup in a daemon.json file.

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

5. Install Kubernetes

```
sudo apt-get update
# apt-transport-https may be a dummy package; if so, you can skip that package
sudo apt-get install -y apt-transport-https ca-certificates curl gpg
# If the directory `/etc/apt/keyrings` does not exist, it should be created before the curl command, read
the note below.
# sudo mkdir -p -m 755 /etc/apt/keyrings
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg
--dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg # This
overwrites any existing configuration in
/etc/apt/sources.list.d/kubernetes.list
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 sudo apt-get
update
sudo apt-get install -y kubelet kubeadm kubectl sudo apt-mark
hold kubelet kubeadm kubectl
sudo systemctl enable --now kubelet
```

```

ubuntu@ip-172-31-40-255:~$ # Add Kubernetes GPG key
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -

# Add Kubernetes repository
sudo tee /etc/apt/sources.list.d/kubernetes.list <<EOF
deb https://apt.kubernetes.io/ kubernetes-xenial main
EOF

# Update package list
sudo apt-get update

# Install kubelet, kubeadm, and kubectl
sudo apt-get install -y kubelet kubeadm kubectl

# Hold the versions of Kubernetes components
sudo apt-mark hold kubelet kubeadm kubectl
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
deb https://apt.kubernetes.io/ kubernetes-xenial main
Hit:1 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Ign:6 https://packages.cloud.google.com/apt kubernetes-xenial InRelease
Err:7 https://packages.cloud.google.com/apt kubernetes-xenial Release

```

After installing Kubernetes, we need to configure internet options to allow bridging.

```

sudo swapoff -a
echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a
/etc/sysctl.conf sudo
sysctl -p

```

```

ubuntu@ip-172-31-45-229:~$ # Disable swap
sudo swapoff -a

# Allow bridging for iptables
echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf

# Apply sysctl changes
sudo sysctl -p
net.bridge.bridge-nf-call-iptables=1
net.bridge.bridge-nf-call-iptables = 1
ubuntu@ip-172-31-45-229:~$ █

```

6. Initialize the Kubecluster

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

```
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 172.31.45.229:6443 --token s9zq75.bsi7js5f62ridulc \  
--discovery-token-ca-cert-hash sha256:91eae090fdd49337bf70d5bf7478e60bc85820d0996651871129a082db6fa8f1  
ubuntu@ip-172-31-45-229:~$ █
```

Copy the mkdir and chown commands from the top and execute them

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

Then, add a common networking plugin called flannel as mentioned in the code.

```
kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/  
kube-flannel.yml
```

```
ubuntu@ip-172-31-45-229:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml  
namespace/kube-flannel created  
clusterrole.rbac.authorization.k8s.io/flannel created  
clusterrolebinding.rbac.authorization.k8s.io/flannel created  
serviceaccount/flannel created  
configmap/kube-flannel-cfg created  
daemonset.apps/kube-flannel-ds created
```

7. Now that the cluster is up and running, we can deploy our nginx server on this cluster.

Apply this deployment file using this command to create a deployment

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

```
ubuntu@ip-172-31-45-229:~$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml  
deployment.apps/nginx-deployment created  
ubuntu@ip-172-31-45-229:~$ █
```

Use ‘kubectl get pods’ to verify if the deployment was properly created and the pod is working correctly.

```
ubuntu@ip-172-31-45-229:~$ kubectl get pods  
NAME                      READY   STATUS    RESTARTS   AGE  
nginx-deployment-d556bf558-krhbv   0/1     Pending   0          2m29s  
nginx-deployment-d556bf558-mhlm2   0/1     Pending   0          2m29s  
ubuntu@ip-172-31-45-229:~$ █
```

```
Next up, create a name alias for this pod. POD_NAME=$(kubectl  
get pods -l app=nginx -o jsonpath=".items[0].metadata.name")
```

8. Lastly, port forward the deployment to your localhost so that you can view it.

```
kubectl port-forward $POD_NAME 8080:80
```

9. Verify your deployment

Open up a new terminal and ssh to your EC2 instance.

Then, use this curl command to check if the Nginx server is running.

```
curl --head http://127.0.0.1:8080
```

```
ubuntu@ip-172-31-45-229:~$ curl --head http://127.0.0.1:8080  
HTTP/1.1 200 OK  
Server: nginx/1.18.0  
Date: Sat, 14 Sep 2024 7:20:53 GMT  
Content-Type: text/html  
Content-Length: 612  
Connection: keep-alive  
ETag: "5c0692e1-265"  
Accept-Ranges: bytes
```

If the response is 200 OK and you can see the Nginx server name, your deployment was successful.

We have successfully deployed our Nginx server on our EC2 instance.

Conclusion:

In this project, we effectively set up Kubernetes and Docker on an AWS EC2 Ubuntu instance, fine-tuned the necessary configurations, and initiated a Kubernetes cluster. We then deployed an Nginx server via a Kubernetes Deployment and utilized the Flannel networking plugin to enable pod communication. By checking the status of the pods and using port forwarding, we were able to access the Nginx server from our local environment. The successful `200 OK` response from the `curl` command verified the successful deployment. This exercise highlighted key Kubernetes functionalities, including cluster setup, application deployment, and operational validation, demonstrating Kubernetes' capability to manage and orchestrate containerized applications with efficiency.

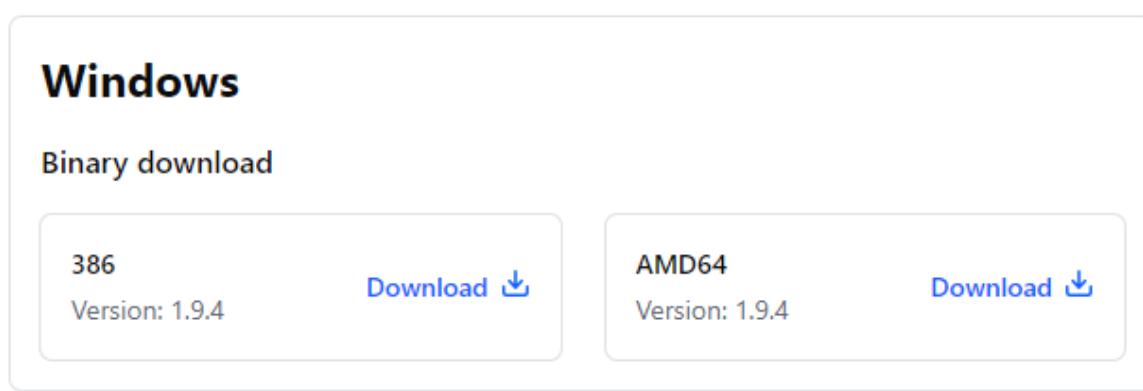
Name:Advik Hegde

Div/Roll No.:D15C/15

Exp 05:To Understand Terraform lifecycle,core concepts,technologies and install it on a Linux or Windows machine.

Installation and Configuration of Terraform in Windows

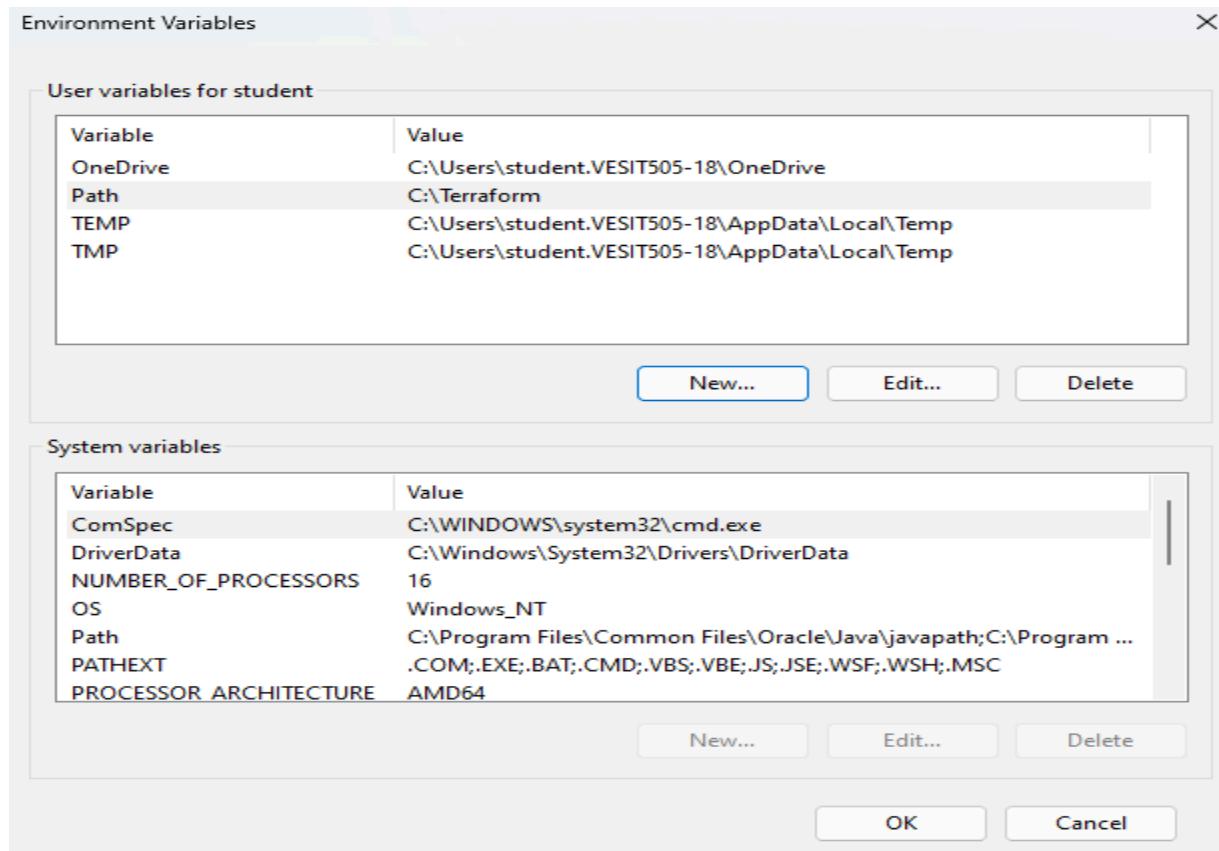
1: To install Terraform, First Download the Terraform Cli Utility for windows from terraforms official website: <https://www.terraform.io/downloads.html> Select the Operating System Windows followed by either 32 bit or 64 bit based on your OS type.



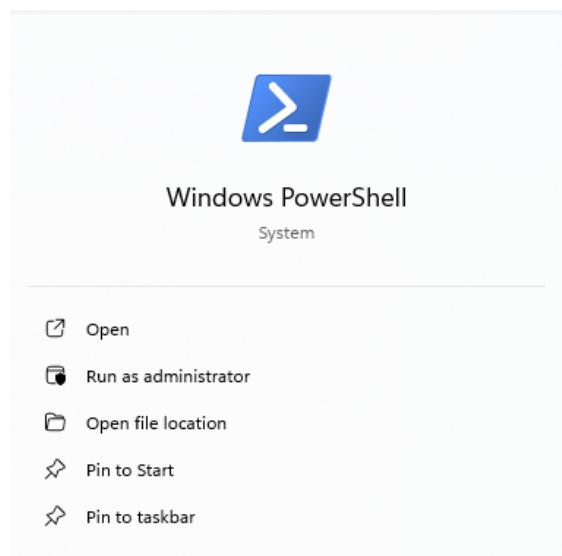
2. Extract the downloaded setup file Terraform.exe in C:\Terraformdirectory.

C:\ > Downloads > terraform_1.9.4_darwin_arm64							Search terraform_1.9.4_darwin_ar	🔍
				Sort	View	Extract all	...	Details
Name	Type	Compressed size	Password ...	Size	Ratio	Date modified		
LICENSE	Text Document	2 KB	No	5 KB	63%	07-08-2024 06:57		
terraform	File	25,119 KB	No	85,795 KB	71%	07-08-2024 06:57		

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



4: Open PowerShell with Admin Access.



5 : Open Terraform in PowerShell and check its functionality.

```
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\student.VESIT505-18> 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.
PS C:\Users\student.VESIT505-18> |
```

Experiment No : 6

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

Implementation

A. Creating docker image using terraform

Prerequisite:

- 1) Download and Install Docker Desktop from <https://www.docker.com/>

Step 1: Check the docker functionality

```
PS C:\Windows\system32> docker --version
Docker version 27.0.3, build 7d4bcd8
PS C:\Windows\system32> 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
  checkpoint  Manage checkpoints
  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
```

Now, create a folder named ‘Terraform Scripts’ in which we save our different types of scripts which will be further used in this experiment.

Step 2: Firstly create a new folder named ‘Docker’ in the ‘TerraformScripts’ folder. Then create a new docker.tf file using Atom editor and write the following contents into it to create a Ubuntu Linux container.

Script:

terraform

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

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

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

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

```

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 }

```

Step 3: Execute Terraform Init command to initialize the resources

```

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

```

Terraform has been successfully initialized!

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

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

PS D:\TerraformScripts\Docker> ■

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

```
PS D:\TerraformScripts\Docker> terraform plan

Terraform used the selected providers to generate the following execution plan.
+ create

Terraform will perform the following actions:

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

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

```
PS D:\TerraformScripts\Docker> terraform apply

Terraform used the selected providers to generate the following execution plan.
+ create

Terraform will perform the following actions:

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

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

Docker images, After Executing Apply step:

```
PS D:\TerraformScripts\Docker> docker images
REPOSITORY      TAG          IMAGE ID      CREATED        SIZE
ubuntu          latest       edbfe74c41f8  2 weeks ago   78.1MB
PS D:\TerraformScripts\Docker> ■
```

Step 6: Execute Terraform destroy to delete the configuration, which will automatically delete the Ubuntu Container.

```
PS D:\TerraformScripts\Docker> terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=df0818fda9652d036abe76b261c69c8177df5c55da3b32310d6f09b66a654482]

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         = [
    "/bin/sh",
    "-c",
    "while true; do sleep 1000; done",
  ] -> 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        = "df0818fda965" -> null
  id              = "df0818fda9652d036abe76b261c69c8177df5c55da3b32310d6f09b66a654482" -> 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
  memory          = 0 -> null
  memory_swap     = 0 -> null
  must_run         = true -> null
  name            = "foo" -> null
  network_data    = [
```

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=df0818fda9652d036abe76b261c69c8177df5c55da3b32310d6f09b66a654482]
docker_container.foo: Destruction complete after 1s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image.ubuntu: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
```

Docker images After Executing Destroy step

```
PS D:\TerraformScripts\Docker> docker images
REPOSITORY      TAG          IMAGE ID      CREATED        SIZE
PS D:\TerraformScripts\Docker>
```

Name : Advik Ashok Hegde

Div/Roll No. : D15C/15

Experiment - 7 :

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

Integrating Jenkins with SonarQube (Prerequisites):

- **Jenkins installed** : To perform this experiment,it is necessary to have jenkins pre installed and up and running at some port like 8080.
- **Docker Installed (for SonarQube)** : It is also a requirement to have docker installed.

Steps to integrate Jenkins with SonarQube

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

STEP:2. Run SonarQube in a Docker container using this command -

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

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

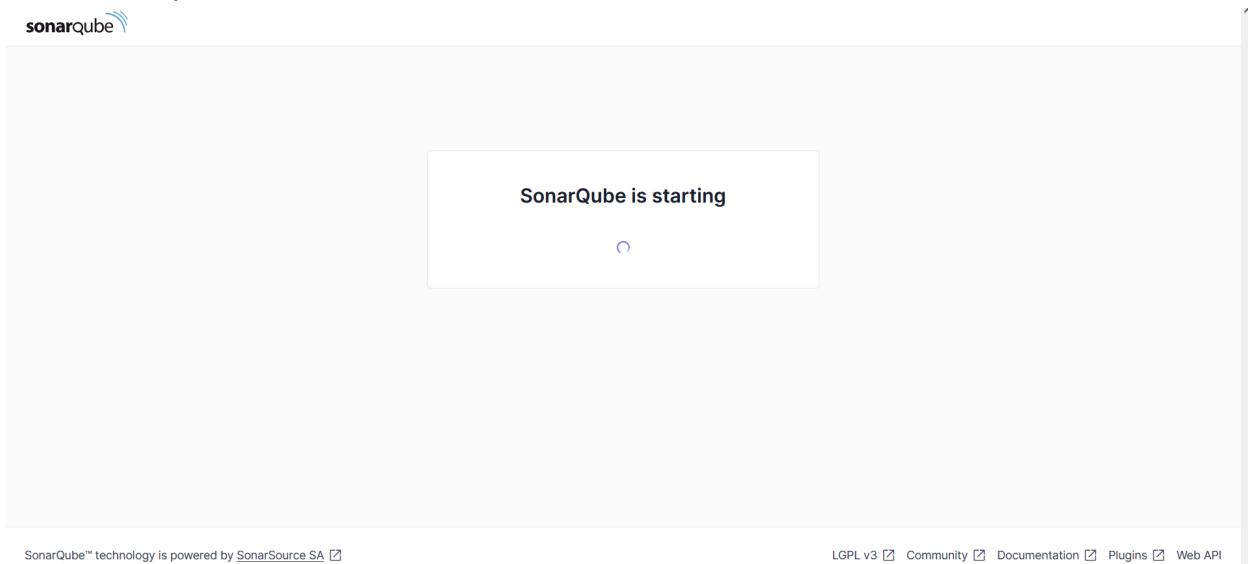
```
C:\Users\ADMIN>docker run -d --name sonarqube -p 9000:9000 sonarqube
dca969bc6baf139bf8d1d7517fd76b0ce90b18db59a88a8bf6f694a648c2d084
```

Run the given command only once and replace the name 'sonarqube' with any name that you prefer.In my case I have not changed the name.

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

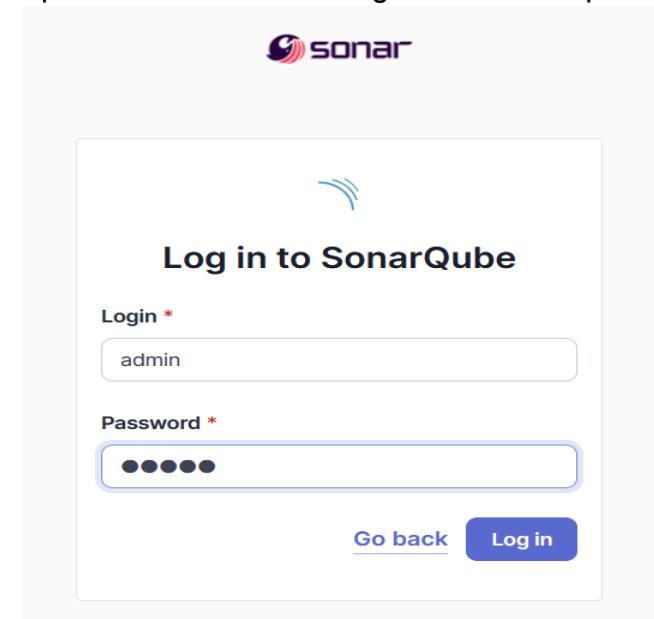
Note: Run the above command only once as once it is run, the sonarqube container is created. Running the same command again will result in an error as the container with that name will already have been created.

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



If you want to access this sonarqube dashboard again once you have closed this tab, you do not have to run the command again, rather, just open your docker and within it run the container that was created and then access the port 9000.

STEP: 4. Login to SonarQube using username `admin` and password `admin`. This is the default username and password for the first log-in into sonarqube.



After Logging in,you will be asked to reset the password,create a new personal password and store it for future use.

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:5. Create a manual project in SonarQube with the name **sonarqube-test**. Again,you can set the name as per your wish,I have decided to name the project as sonarqube-test.

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 ↗](#)

Cancel

Next

After completing the process of setting up the project,come back to the Jenkins Dashboard.

Go to Manage Jenkins and search for SonarQube Scanner for Jenkins. It is an available plugin that you can search for and then install.

The screenshot shows the Jenkins 'Plugins' page. In the top navigation bar, 'Manage Jenkins' is selected. Below it, the 'Plugins' section is active. A search bar at the top right contains the text 'sonarqube'. The search results list three items:

Install	Name	Released
<input checked="" type="checkbox"/>	SonarQube Scanner 2.17.2 External Site/Tool Integrations Build Reports	7 mo 8 days ago
<input type="checkbox"/>	Sonar Gerrit 388.v9b_f1cb_e42306 External Site/Tool Integrations	3 mo 22 days ago
<input type="checkbox"/>	SonarQube Generic Coverage 1.0 TODO	5 yr 1 mo ago

STEP : 6 . Under Jenkins 'Configure System', look for SonarQube Installations and enter the details. Provide the name for the project and enter the url based on the port that is used to run sonarqube which in my case is port **9000**.

The screenshot shows the 'Configure System' page under 'System Configuration'. The 'SonarQube installations' section is expanded. It includes fields for 'Name' (set to 'sonarqube') and 'Server URL' (set to 'http://localhost:9000'). There is also a 'Server authentication token' dropdown set to '- none -' and an 'Advanced' button.

STEP : 7 . Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.

Add SonarQube Scanner

☰ SonarQube Scanner

Name
sonarqube

Install automatically ?

☰ Install from Maven Central

Version
SonarQube Scanner 6.2.0.4584

Add Installer ▾

Add SonarQube Scanner

STEP : 8. After the configuration, create a New Item in Jenkins,choose a freestyle project.

New Item

Enter an item name

SonarQube

Select an item type



Freestyle project

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



Maven project

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



Pipeline

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



Multi-configuration project

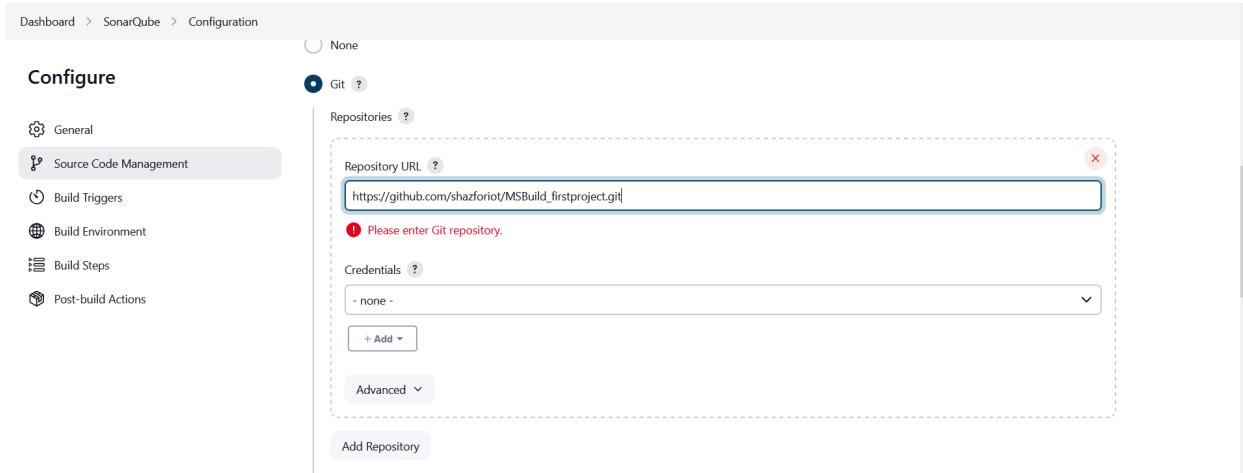
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

OK

STEP : 9. Choose this GitHub repository in Source Code Management.

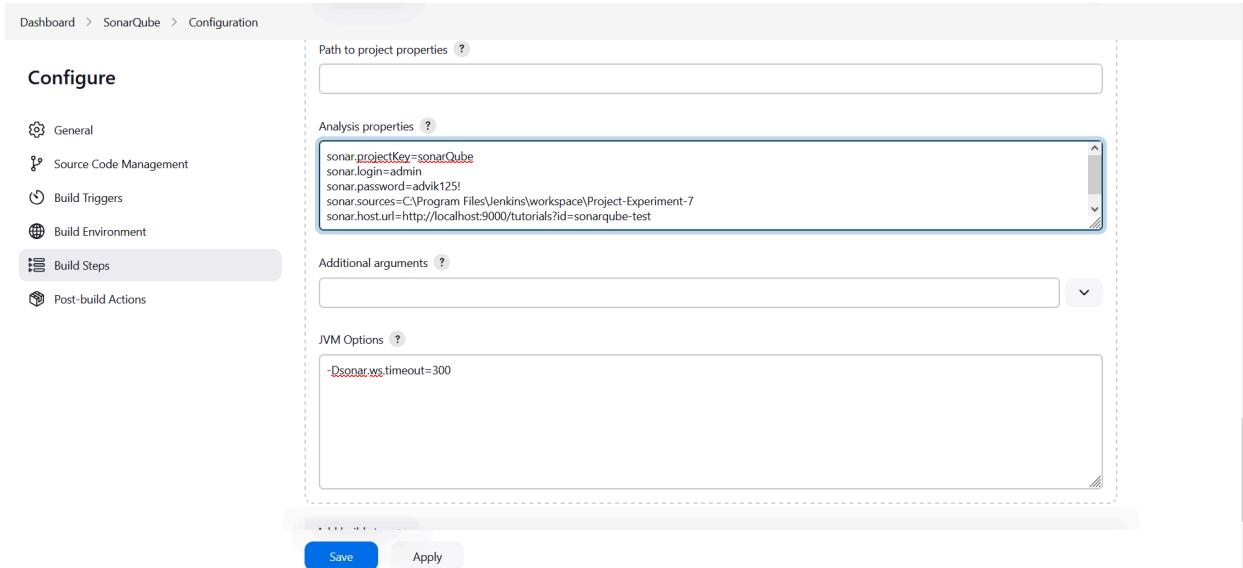
https://github.com/shazforiot/MSBuild_firstproject.git

It is a sample hello-world project with no vulnerabilities and issues, just to test the integration.



STEP : 10. Under Build-> Execute SonarQube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.

- Project-Key:sonarQube
- Login:admin
- Password:advik125!
- Host URL : http://localhost9000/



Note : Carefully enter the values for the Project Key,Login,Password,Source path and Host URL as any error will result in a **Failed Build Attempt**.

STEP : 11. Go to http://localhost:9000/<user_name>/permissions and allow Execute Permissions to the Admin user. In my case ,the user_name will be replaced with 'admin'.

The screenshot shows the SonarQube administration interface under the 'Global Permissions' section. It lists users and groups with their assigned permissions. The 'sonar-administrators' group has 'Administrator System' and 'Execute Analysis' checked, along with 'Quality Gates' and 'Quality Profiles'. The 'sonar-users' group has 'Administrator System' and 'Execute Analysis' checked, along with 'Quality Gates' and 'Quality Profiles'. The 'Anyone' group has 'Administrator System' and 'Execute Analysis' checked, along with 'Quality Gates' and 'Quality Profiles'. The 'Administrator admin' user has 'Administrator System' and 'Execute Analysis' checked, along with 'Quality Gates' and 'Quality Profiles'.

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

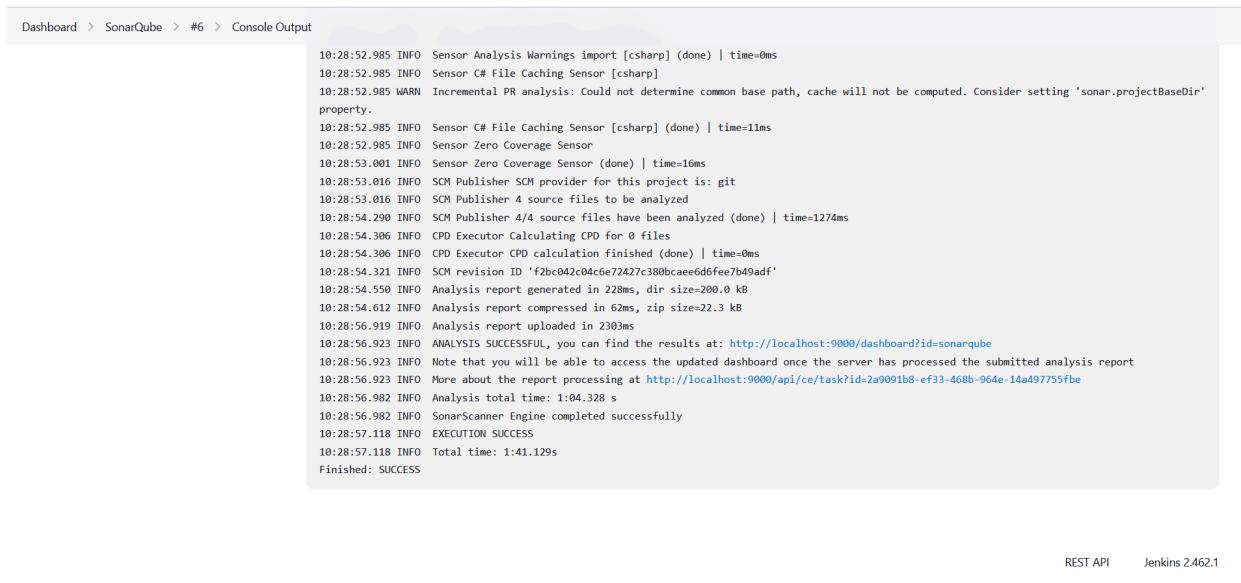
STEP : 12. Run The Build.

The screenshot shows the Jenkins dashboard with a 'SonarQube' job selected. The job configuration includes the following steps:

- Status
- </> Changes
- Workspace
- Build Now (highlighted)
- Configure
- Delete Project
- SonarQube
- Rename

The 'SonarQube' icon is also present on the right side of the dashboard.

Once the build is complete, check the console output to determine whether the build was a success or a failure and identify the error in case of any errors.

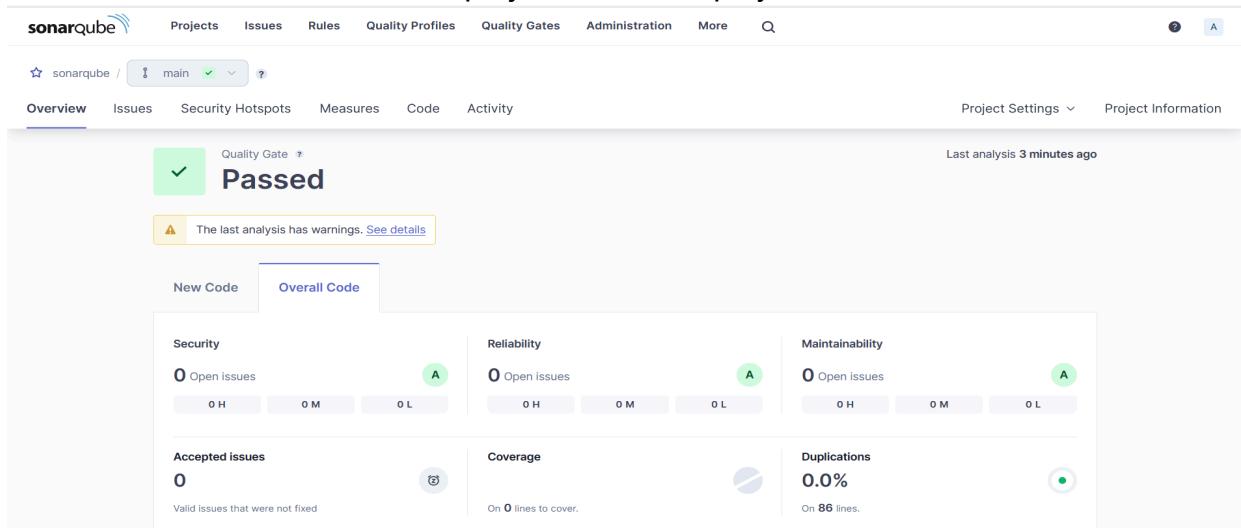


```
Dashboard > SonarQube > #6 > Console Output

10:28:52.985 INFO Sensor Analysis Warnings import [csharp] (done) | time=0ms
10:28:52.985 INFO Sensor C# File Caching Sensor [csharp]
10:28:52.985 WARN Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir' property.
10:28:52.985 INFO Sensor C# File Caching Sensor [csharp] (done) | time=11ms
10:28:52.985 INFO Sensor Zero Coverage Sensor
10:28:53.001 INFO Sensor Zero Coverage Sensor (done) | time=16ms
10:28:53.016 INFO SCM Publisher SCM provider for this project is: git
10:28:53.016 INFO SCM Publisher 4 source files to be analyzed
10:28:54.290 INFO SCM Publisher 4/4 source files have been analyzed (done) | time=1274ms
10:28:54.306 INFO CPD Executor Calculating CPD for 0 files
10:28:54.306 INFO CPD Executor CPD calculation finished (done) | time=0ms
10:28:54.321 INFO SCM revision ID 'f2bc042c04ce5e72427c380bcae0d6fee7b49adf'
10:28:54.358 INFO Analysis report generated in 228ms, dir size=200.0 kB
10:28:54.612 INFO Analysis report compressed in 62ms, zip size=22.3 kB
10:28:56.019 INFO Analysis report uploaded in 230ms
10:28:56.923 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube
10:28:56.923 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
10:28:56.923 INFO More about the report processing at http://localhost:9000/api/ce/task?id=2a9091b8-ef33-468b-964e-14a497755fbe
10:28:56.982 INFO Analysis total time: 1:04.328 s
10:28:56.982 INFO SonarScanner Engine completed successfully
10:28:57.118 INFO EXECUTION SUCCESS
10:28:57.118 INFO Total time: 1:41.129s
Finished: SUCCESS
```

REST API Jenkins 2.462.1

STEP : 13. Once the build is complete and successful, check the project in SonarQube. The overview of the project should display Passed.



The screenshot shows the SonarQube project overview for the 'main' branch. The top navigation bar includes 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', 'Administration', 'More', and a search bar. Below the navigation is a breadcrumb trail: 'sonarqube / main'. The main content area features a large green 'Passed' status indicator with a checkmark icon. A yellow warning message states: 'The last analysis has warnings. See details'. The overview is divided into several sections: 'New Code' and 'Overall Code' (selected), 'Security' (0 open issues, A grade), 'Reliability' (0 open issues, A grade), 'Maintainability' (0 open issues, A grade), 'Accepted issues' (0 valid issues), 'Coverage' (0 lines to cover), and 'Duplications' (0.0% on 86 lines). The bottom right corner indicates the last analysis was 3 minutes ago.

Conclusion : In this experiment, we integrated Jenkins with SonarQube using Docker to automate code quality analysis. SonarQube, deployed via Docker, efficiently performed static code checks, while Jenkins orchestrated the process through a pipeline that triggered analysis after every code update. This setup ensured continuous monitoring of code quality, providing immediate feedback on potential issues such as bugs and code smells. Docker simplified the management of SonarQube, making the entire process more efficient. Overall, this integration streamlined the workflow and enhanced code quality through automated analysis.

Experiment - 8 :

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

Integrating Jenkins with SonarQube (Prerequisites)

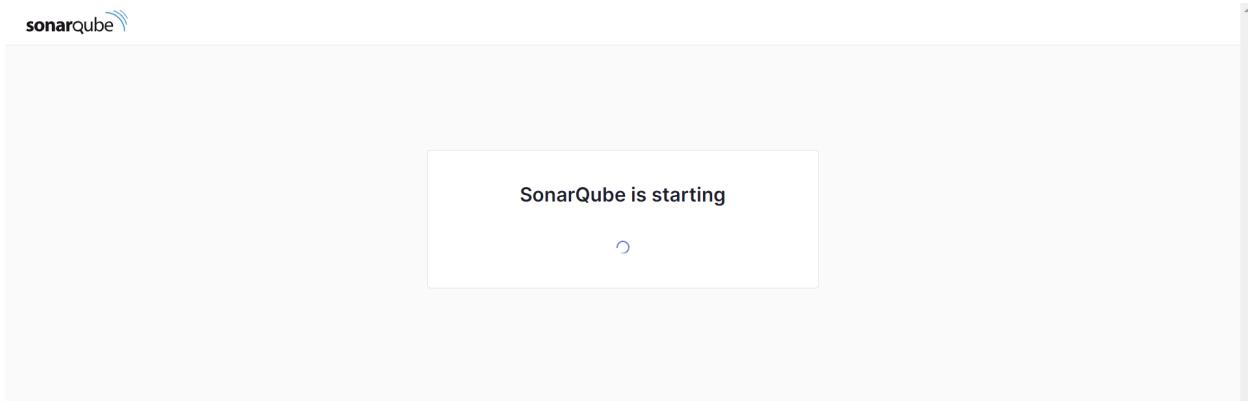
- Jenkins installed
- Docker Installed (for SonarQube)
- SonarQube Docker Image

Steps to create a Jenkins CI/CD Pipeline and use SonarQube to perform SAST

1. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.
2. Run SonarQube in a Docker container using this command -

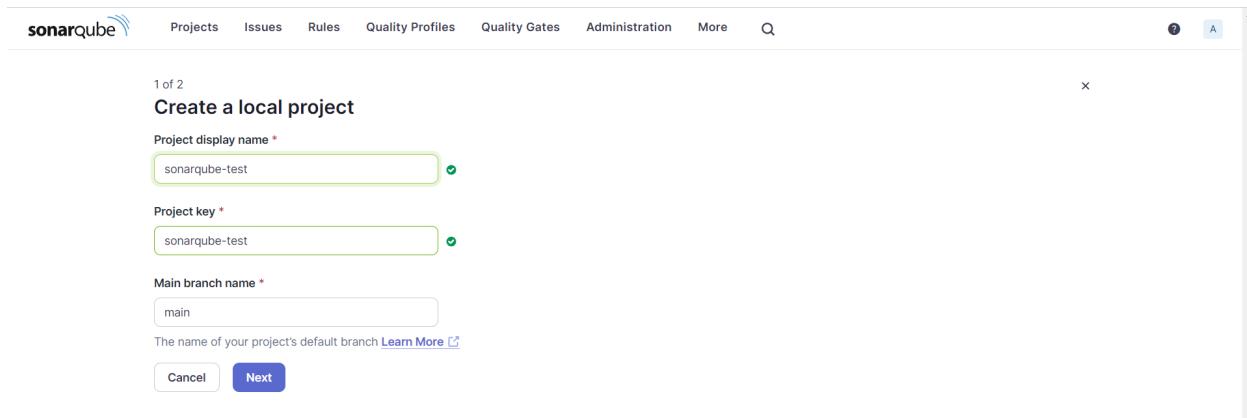
```
C:\Users\ADMIN>docker run -d --name sonarqube-2 -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest  
9f2e3f25d6fc6dac6175fc2feabc967ff04003b0f90b9e2a58bbc7870f093c50  
C:\Users\ADMIN>
```

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



4. Login to SonarQube using username *admin* and password which we reset to *advi125!*

5. Create a manual project in SonarQube with the name **sonarqube-test** .



The screenshot shows the 'Create a local project' dialog in SonarQube. The 'Project display name' field contains 'sonarqube-test'. The 'Project key' field also contains 'sonarqube-test'. The 'Main branch name' field contains 'main'. At the bottom, there are 'Cancel' and 'Next' buttons.

Setup the project and come back to Jenkins Dashboard.

6. Create a New Item in Jenkins, choose **Pipeline**.

7. Under Pipeline Script, enter the following -

```
node {  
    stage('Cloning the GitHub Repo') {  
        git 'https://github.com/shazforiot/GOL.git'  
    }  
    stage('SonarQube analysis') {  
        withSonarQubeEnv('sonarqube') {  
            bat """  
            sh C:/ProgramData/Jenkins/.jenkins/tools/hudson.plugins.sonar.  
            SonarRunnerInstallation/sonarqube/bin/sonar-scanner.bat ^  
            -D sonar.login=<admin> ^  
            -D sonar.password=<advik125!> ^  
            -D sonar.projectKey=<sonarqube-test> ^  
            -D sonar.exclusions=vendor/**,resources/**,*/*.java ^  
            -D sonar.host.url=http://127.0.0.1:9000/"  
            -D sonar.branch.name=main  
        }  
    }  
}
```

Dashboard > SonarQube-2 > Configuration

Configure

Pipeline

Definition

Pipeline script

```
Script ?  
try sample Pipeline...  
  
1 ~ node {  
2 ~   stage('Cloning the GitHub Repo') {  
3 ~     git 'https://github.com/shazforiot/GOL.git'  
4 ~   }  
5 ~   stage('SonarQube analysis') {  
6 ~     withSonarQubeEnv('sonarqube') {  
7 ~       bat ""^  
8 ~       sh "C:/ProgramData/Jenkins/tools/hudson.plugins.sonar.SonarRunnerInstallation/sonarqube/bin/sonar-scanner.bat ^  
9 ~       -D sonar.login=cadm1n ^  
10 ~      -D sonar.password=cadm1k1251> ^  
11 ~      -D sonar.projectKey=sonarqube-test> ^  
12 ~      -D sonar.exclusions=<resources>/**, **/*.java ^  
13 ~      -D sonar.host.url=http://127.0.0.1:9080/  
14 ~    }  
15 ~  }  
16 ~ }  
17 }
```

Use Groovy Sandbox ?

Pipeline Syntax

Save Apply

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

8. Run The Build.

Jenkins

Dashboard > SonarQube-2 >

Status Changes Build Now Configure Delete Pipeline Full Stage View Stages Rename Pipeline Syntax

SonarQube-2

Stage View

No data available. This Pipeline has not yet run.

Permalinks

9. Check the console output once the build is complete.

```
Dashboard > SonarQube-2 > #4
01:38:50.321 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/util/TextAreaCellRenderer.html
for block at line 75. Keep only the first 100 references.
01:38:50.321 INFO CPD Executor CPD calculation finished (done) | time=238591ms
01:38:50.413 INFO SCM revision ID 'ba799ba7eb576f04a4612322b0d412c5e6e1e5e4'
01:38:58.184 INFO Analysis report generated in 5253ms, dir size=127.2 MB
01:39:17.257 INFO Analysis report compressed in 19081ms, zip size=29.6 MB
01:39:28.472 INFO Analysis report uploaded in 11215ms
01:39:28.488 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://127.0.0.1:9000/dashboard?id=sonarqube-test
01:39:28.488 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
01:39:28.488 INFO More about the report processing at http://127.0.0.1:9000/api/ce/task?id=add9b398-1d74-4d8a-81ac-9b210f0e0b94
01:39:48.048 INFO Analysis total time: 11:58.619 s
01:39:48.086 INFO SonarScanner Engine completed successfully
01:39:49.184 INFO EXECUTION SUCCESS
01:39:49.314 INFO Total time: 12:15.607s
[Pipeline]
[Pipeline] // withSonarQubeEnv
[Pipeline] // stage
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

10. After that, check the project in SonarQube.

The screenshot shows the SonarQube main dashboard for the 'main' project. At the top, there's a navigation bar with links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a search bar. Below the navigation is a breadcrumb trail: sonarqube-test / main. A dropdown menu next to the breadcrumb shows 'main' is the active branch. The main content area displays a summary card for the 'main' branch, indicating 683k Lines of Code, Version not provided, and a 'Passed' status with a green checkmark. It also shows the last analysis was 22 minutes ago. Below this, there are several cards: Security (0 Open issues), Reliability (68k Open issues), Maintainability (164k Open issues), Accepted Issues (0), Coverage (0 lines to cover), and Duplications (50.6% on 759k lines). On the left, there are tabs for Overview, Issues, Security Hotspots, Measures, Code, and Activity. The 'Issues' tab is currently selected.

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

11. Code Problems - Bugs :

The screenshot shows the SonarQube Issues tab for the 'main' project. The left sidebar has filters for Issues in new code, Clean Code Attribute, Software Quality, Severity, and Type (with 'Bug' selected). The main panel lists two bugs: 'Add "lang" and/or "xml:lang" attributes to this "<html>" element' and 'Insert a <!DOCTYPE> declaration to before this <html> tag.'. Both bugs are marked as 'Reliability' issues, 'Open', and 'Not assigned'. The right side of the interface includes buttons for Bulk Change, Select Issues, and Navigate to issue, along with statistics: 21,573 issues and 661d effort. A warning message at the bottom states: 'Embedded database should be used for evaluation purposes only. The embedded database will not scale. It will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.'

Code smells :

The screenshot shows the SonarQube interface for the project 'sonarqube-test'. The 'Issues' tab is selected. On the left, a sidebar lists categories like 'Clean Code Attribute', 'Software Quality', and 'Severity'. Under 'Type', 'Code Smell' is selected. The main area displays three specific code smell issues:

- Remove this deprecated "width" attribute.** (Consistency, Maintainability) - L9 - 5min effort - 4 years ago - Code Smell - Major
- Remove this deprecated "align" attribute.** (Consistency, Maintainability) - L11 - 5min effort - 4 years ago - Code Smell - Major
- Remove this deprecated "align" attribute.** (Consistency) - L9 - 5min effort - 4 years ago - Code Smell - Major

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

Clean Code Consistency issues:

The screenshot shows the SonarQube interface for the project 'sonarqube-test'. The 'Issues' tab is selected. On the left, a sidebar shows filters and a section for 'Clean Code Attribute' under 'Consistency' (91k). The main area displays two consistency issues:

- Insert a <!DOCTYPE> declaration to before this <html> tag.** (Reliability) - L1 - 5min effort - 4 years ago - Bug - Major
- Remove this deprecated "width" attribute.** (Consistency, Maintainability) - L9 - 5min effort - 4 years ago - Code Smell - Major

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

Reliability Issues:

The screenshot shows the SonarQube interface for the project 'sonarqube-test'. The 'Issues' tab is selected. In the left sidebar, under 'Software Quality', 'Reliability' is selected, showing 31k issues. The main panel displays several reliability-related code smells:

- Add "lang" and/or "xml:lang" attributes to this <html> element. (Intentionality: accessibility, wcag2-a)
- Insert a <!DOCTYPE> declaration to before this <html> tag. (Consistency: user-experience)
- Anchors must have content and the content must be accessible by a screen reader. (Consistency)

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

Maintainability Issues:

The screenshot shows the SonarQube interface for the project 'sonarqube-test'. The 'Issues' tab is selected. In the left sidebar, under 'Software Quality', 'Maintainability' is selected, showing 76k issues. The main panel displays several maintainability-related code smells:

- Remove this deprecated "width" attribute. (Consistency: html5, obsolete)
- Remove this deprecated "align" attribute. (Consistency: html5, obsolete)
- Remove this deprecated "align" attribute. (Consistency)

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

Conclusion:

In this experiment, we set up a Jenkins CI/CD pipeline integrated with SonarQube to automate static analysis on a sample application. Jenkins was configured to trigger builds and run SonarQube's analysis with every code change, detecting bugs, code smells, and security vulnerabilities. This pipeline provided continuous monitoring and ensured early detection of issues, improving code quality and security. The experiment showcased how integrating CI/CD pipelines with SonarQube enhances development efficiency and ensures better, more reliable software.

Adv DevOps Exp 09

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

Theory:

What is Nagios?

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

Nagios is used for continuous monitoring of systems, applications, service and business processes in a DevOps culture.

Why We Need Nagios tool?

Here are the important reasons to use Nagios monitoring tool:

- Detects all types of network or server issues
- Helps you to find the root cause of the problem which allows you to get the permanent solution to the problem
- Active monitoring of your entire infrastructure and business processes
- Allows you to monitor and troubleshoot server performance issues
- Helps you to plan for infrastructure upgrades before outdated systems create failures
- You can maintain the security and availability of the service
- Automatically fix problems in a panic situation

Features of Nagios

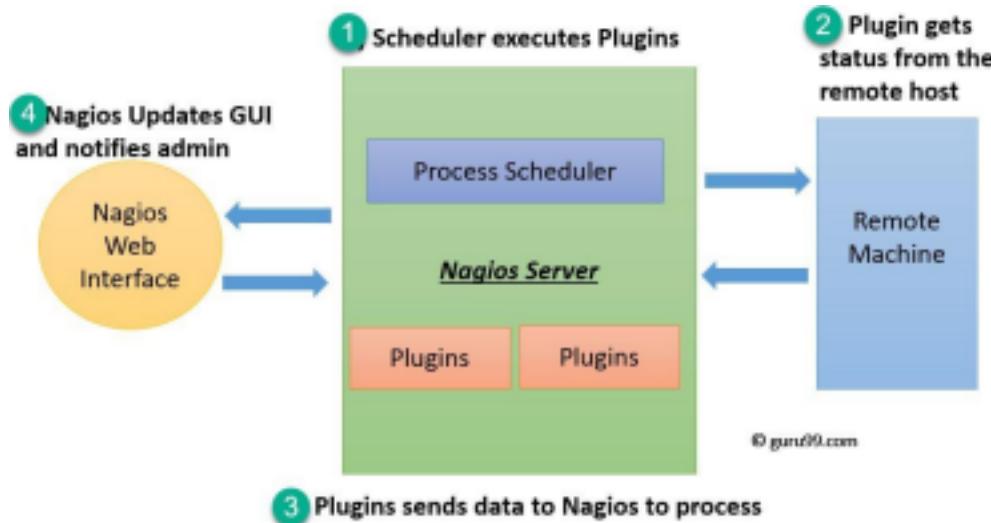
Following are the important features of Nagios monitoring tool:

- Relatively scalable, Manageable, and Secure
- Good log and database system
- Informative and attractive web interfaces
- Automatically send alerts if condition changes
- If the services are running fine, then there is no need to do check that host is an alive
- Helps you to detect network errors or server crashes
- You can troubleshoot the performance issues of the server.
- The issues, if any, can be fixed automatically as they are identified during the monitoring process
- You can monitor the entire business process and IT infrastructure with a single pass
- The product's architecture is easy to write new plugins in the language of your choice
- Nagios allows you to read its configuration from an entire directory which helps you to decide how to define individual files
- Utilizes topology to determine dependencies
- Monitor network services like HTTP, SMTP, HTTP, SNMP, FTP, SSH, POP, etc.
- Helps you to define network host hierarchy using parent hosts
- Ability to define event handlers that runs during service or host events for proactive

- problem resolution
- Support for implementing redundant monitoring hosts

Nagios Architecture

Nagios is a client-server architecture. Usually, on a network, a Nagios server is running on a host, and plugins are running on all the remote hosts which should be monitored.



1. The scheduler is a component of the server part of Nagios. It sends a signal to execute the plugins at the remote host.
2. The plugin gets the status from the remote host
3. The plugin sends the data to the process scheduler
4. The process scheduler updates the GUI and notifications are sent to admins.

Step 1: Create a security group with the required configurations

I have created a new security group with a name 'newsecurity'

EC2 > Security Groups > Create security group

Create security group Info

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, you must provide a name and optional description.

Basic details

Security group name Info

Name cannot be edited after creation.

Description Info

VPC Info

I have modified the INBOUND RULES as follows

Inbound rules Info

Type <small>Info</small>	Protocol <small>Info</small>	Port range <small>Info</small>	Source <small>Info</small>	Description - optional <small>Info</small>	Delete
HTTP	TCP	80	Anywhere	::/0 X	<input type="button" value="Delete"/>
HTTPS	TCP	443	Anywhere	0.0.0.0/0 X	<input type="button" value="Delete"/>
SSH	TCP	22	Anywhere	0.0.0.0/0 X	<input type="button" value="Delete"/>
All ICMP - IPv6	IPv6 ICMP	All	Anywhere	::/0 X	<input type="button" value="Delete"/>
All ICMP - IPv4	ICMP	All	Anywhere	0.0.0.0/0 X	<input type="button" value="Delete"/>
All traffic	All	All	Anywhere	0.0.0.0/0 X	<input type="button" value="Delete"/>
Custom TCP	TCP	5666	Anywhere	0.0.0.0/0 X	<input type="button" value="Delete"/>

Step 2: Create ec2 instance

Name it as nagios-host. Select instance type as amazon-linux and choose the already created key pair and security group

The screenshot shows the AWS Lambda console interface. A modal window is open for creating a new function. The function name is set to 'HelloWorld'. Under the 'Runtime' dropdown, 'Node.js 12.x' is selected. The 'Handler' field contains the default value 'index.handler'. The 'Role' dropdown is set to 'Lambda execution role (lambda-execution-role)'. Below the role, there is a note: 'Lambda automatically creates a role for your function. You can't edit or delete this role.' At the bottom of the modal, there is a large blue 'Create Function' button.

The screenshot shows the AWS Lambda console interface with the 'HelloWorld' function selected. On the left, the function's ARN is shown: 'arn:aws:lambda:ap-south-1:123456789012:function>HelloWorld'. The 'Code' tab is active, showing the code editor with the following code:

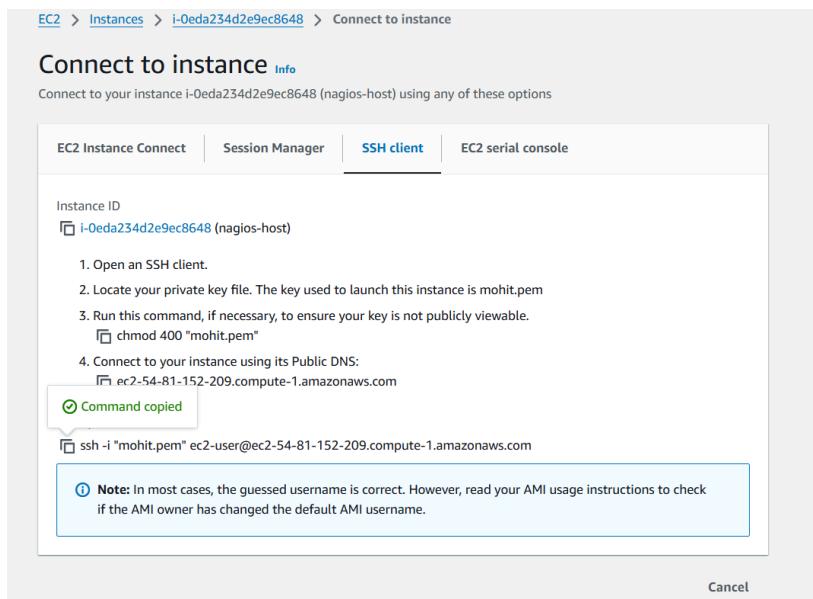
```
function handler(event, context) {
  const response = {
    statusCode: 200,
    body: "Hello from Lambda"
  };

  context.succeed(response);
}
```

The 'Logs' tab is also visible on the right side of the interface.

The screenshot shows the AWS Lambda console interface with the 'HelloWorld' function selected. On the left, the function's ARN is shown: 'arn:aws:lambda:ap-south-1:123456789012:function>HelloWorld'. The 'Logs' tab is active, showing the log stream 'HelloWorld.log' with the message 'Function started'.

Copy the given ssh command, as we will require it for logging into our nagios-host instance from our windows powershell



Step 3: Open an administrative powershell and remotely login using the above mentioned ssh command

```

> ec2-user@ip-172-31-92-249:~ 
Windows PowerShell
Copyright (c) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\WINDOWS\system32> cd C:\Users\Del1\Downloads
PS C:\Users\Del1\Downloads> ssh -i "mohit.pem" ec2-user@ec2-54-81-152-209.compute-1.amazonaws.com
, # ~\_ ##### Amazon Linux 2023
~~ \#####\ ~\###| ~~ \#/#/ https://aws.amazon.com/linux/amazon-linux-2023
~~ \#/ .-'-> ~~ V~' '-'>
~~ / ~~ / 
~~ / .-' / 
~~ / / 
Last login: Mon Sep 30 09:25:13 2024 from 125.99.93.18
, # ~\_ ##### Amazon Linux 2023
~~ \#####\ ~\###| ~~ \#/#/ https://aws.amazon.com/linux/amazon-linux-2023
~~ \#/ .-'-> ~~ V~' '-'>
~~ / ~~ / 
~~ / .-' / 
~~ / / 
Last login: Mon Sep 30 09:25:13 2024 from 125.99.93.18
[ec2-user@ip-172-31-92-249 ~]$ sudo yum update
Last metadata expiration check: 0:13:13 ago on Mon Sep 30 09:23:03 2024.
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-92-249 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:13:23 ago on Mon Sep 30 09:23:03 2024.
Package httpd-2.4.62-1.amzn2023.x86_64 is already installed.
Package php8.3-8.3.10-1.amzn2023.0.1.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!

```

And then run these commands
sudo yum update
sudo yum install httpd php

```
[ec2-user@ip-172-31-41-160 ~]$ sudo yum update
Last metadata expiration check: 0:01:37 ago on Wed Oct 2 12:28:33 2024.
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-41-160 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:01:45 ago on Wed Oct 2 12:28:33 2024.
Dependencies resolved.

=====
Package           Architecture      Version       Repository   Size
=====
Installing:
httpd            x86_64          2.4.62-1.amzn2023
php8_3           x86_64          8.3.10-1.amzn2023.0.1

Installing dependencies:
apr              x86_64          1.7.2-2.amzn2023.0.2
apr-util         x86_64          1.6.3-1.amzn2023.0.1
generic-logging-httd
httpd-core       x86_64          2.4.62-1.amzn2023
httpd-filesystem
httpd-tools      x86_64          2.4.62-1.amzn2023
libxml2          x86_64          1.0.34-4.amzn2023.0.2
libxml2-xmlns   x86_64          1.0.19-5.amzn2023
libxml2-modular x86_64          1.1.36-5.amzn2023.0.2
libxml2-xmlsec  x86_64          2.1.49-3.amzn2023.0.3
mailing          noarch         2.1.49-3.amzn2023.0.3
nginx            noarch         1.11.24-0.1.amzn2023.0.4
nginx-filesystem
nginx-mod-wsgi  x86_64          0.3.11-1.amzn2023.0.1
php8_3-common   x86_64          8.3.10-1.amzn2023.0.1
php8_3-process  x86_64          8.3.10-1.amzn2023.0.1
php8_3-xml      x86_64          8.3.10-1.amzn2023.0.1

Installing weak dependencies:
apr-util-openssl x86_64          1.6.3-1.amzn2023.0.1
mod              x86_64          2.5.22-1.amzn2023.0.3
mod_wsgi         x86_64          2.4.62-1.amzn2023
php8_3-ftp      x86_64          8.3.10-1.amzn2023.0.1
php8_3-mbstring x86_64          8.3.10-1.amzn2023.0.1
php8_3-opcache  x86_64          8.3.10-1.amzn2023.0.1
php8_3-pdo     x86_64          8.3.10-1.amzn2023.0.1
php8_3-sodium   x86_64          8.3.10-1.amzn2023.0.1

Transaction Summary
Install 25 Packages
```

sudo yum install gcc glibc glibc-common

```
[ec2-user@ip-172-31-41-160 ~]$ sudo yum install gcc glibc glibc-common
Last metadata expiration check: 0:02:02 ago on Wed Oct 2 12:28:33 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
Installing dependencies:
annobin-docs     noarch         10.93-1.amzn2023.0.1
annobin-plugin-gcc
cpp              x86_64          10.93-1.amzn2023.0.1
gc               x86_64          11.4.1-2.amzn2023.0.2
glibc-devel      x86_64          8.0.4-5.amzn2023.0.2
glibc-headers-x86
guile22         noarch         2.3.4-52.amzn2023.0.11
kernel-headers   x86_64          2.3.4-52.amzn2023.0.11
guile2          x86_64          2.2.7-2.amzn2023.0.3
kernel-headers   x86_64          6.1.109-118.189.amzn2023
libmpc           x86_64          1.2.1-2.amzn2023.0.2
libtool-ltdl    x86_64          2.4.7-1.amzn2023.0.3
libcrypt-devel   x86_64          4.4.33-7.amzn2023
make             x86_64          1:4.3-5.amzn2023.0.2

Transaction Summary
Install 13 Packages

Total download size: 52 M
Installed size: 168 M
Is this ok [y/N]: y
Downloading Packages:
(1/13): annobin-docs-10.93-1.amzn2023.0.1.noarch.rpm                                852 kB/s |  92 kB  00:00
(2/13): annobin-plugin-gcc-10.93-1.amzn2023.0.1.x86_64.rpm                          6.5 MB/s | 887 kB  00:00
(3/13): gc-8.0.4-5.amzn2023.0.2.x86_64.rpm                                         2.3 MB/s | 105 kB  00:00
(4/13): glibc-devel-2.34-52.amzn2023.0.11.x86_64.rpm                           1.1 MB/s | 27 kB  00:00
(5/13): cpp-11.4.1-2.amzn2023.0.2.x86_64.rpm                                     32 MB/s | 10 MB  00:00
(6/13): glibc-headers-x86-2.34-52.amzn2023.0.11.noarch.rpm                      2.9 MB/s | 427 kB  00:00
(7/13): kernel-headers-6.1.109-118.189.amzn2023.x86_64.rpm                         16 MB/s | 1.4 MB  00:00
(8/13): libmpc-1.2.1-2.amzn2023.0.2.x86_64.rpm                               2.1 MB/s | 62 kB  00:00
(9/13): guile22-2.2.7-2.amzn2023.0.3.x86_64.rpm                            27 MB/s | 6.4 MB  00:00
(10/13): libtool-ltdl-2.4.7-1.amzn2023.0.3.x86_64.rpm                         322 kB/s | 38 kB  00:00
(11/13): libcrypt-devel-4.4.33-7.amzn2023.x86_64.rpm                         1.4 MB/s | 32 kB  00:00
```

sudo yum install gd gd-devel

Package	Architecture	Version	Repository	Size
gd	x86_64	2.3.3-3.amzn2023.0.3	amazonlinux	139 k
gd-devel	x86_64	2.3.3-5.amzn2023.0.3	amazonlinux	38 k
Installing:				
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
brrip2-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
caja-filesystem	x86_64	3.22.2-1.amzn2023.0.4	amazonlinux	16 k
fontconfig	x86_64	2.13.94-2.amzn2023.0.2	amazonlinux	273 k
fontconfig-devel	x86_64	2.13.94-2.amzn2023.0.2	amazonlinux	128 k
fonts-filesystem	noarch	1:2.0.5-12.amzn2023.0.2	amazonlinux	9.5 k
freetype	x86_64	2.13.2-5.amzn2023.0.1	amazonlinux	423 k
freetype-devel	x86_64	2.13.2-5.amzn2023.0.1	amazonlinux	912 k
glib2-devel	x86_64	2.74.7-689.amzn2023.0.2	amazonlinux	486 k
google-noto-fonts-common	noarch	29201286-1.amzn2023.0.2	amazonlinux	15 k
google-noto-sans-vf-fonts	noarch	20201206-2.amzn2023.0.2	amazonlinux	492 k
grayscale2	x86_64	1.3.14-7.amzn2023.0.2	amazonlinux	97 k
grayscale2-devel	x86_64	1.3.14-7.amzn2023.0.2	amazonlinux	21 k
harfbuzz	x86_64	7.0.0-2.amzn2023.0.1	amazonlinux	868 k
harfbuzz-devel	x86_64	7.0.0-2.amzn2023.0.1	amazonlinux	404 k
harfbuzz-icu	x86_64	7.0.0-2.amzn2023.0.1	amazonlinux	18 k
jbigkit2s	x86_64	2.1.21.amzn2023.0.2	amazonlinux	54 k
langpacks-core-font-en	noarch	3.0-21.amzn2023.0.4	amazonlinux	10 k
libICE	x86_64	1.0.10-6.amzn2023.0.2	amazonlinux	71 k
libSM	x86_64	1.2.3-8.amzn2023.0.2	amazonlinux	42 k
libX11	x86_64	1.7.2-2.amzn2023.0.4	amazonlinux	657 k
libX11-common	noarch	1.7.2-3.amzn2023.0.4	amazonlinux	152 k
libX11-devel	x86_64	1.7.2-3.amzn2023.0.4	amazonlinux	939 k
libX11-xcb	x86_64	1.7.2-3.amzn2023.0.4	amazonlinux	12 k
libXau	x86_64	1.0.9-6.amzn2023.0.2	amazonlinux	31 k
libXau-devel	x86_64	1.0.9-6.amzn2023.0.2	amazonlinux	14 k
libXext	x86_64	1.3.4-6.amzn2023.0.2	amazonlinux	41 k
libXpm	x86_64	3.5.15-2.amzn2023.0.3	amazonlinux	65 k
libXpm-devel	x86_64	3.5.15-2.amzn2023.0.3	amazonlinux	59 k
libXrender	x86_64	0.9.10-14.amzn2023.0.2	amazonlinux	28 k
libXt	x86_64	1.2.0-4.amzn2023.0.2	amazonlinux	181 k
libblkid-devel	x86_64	2.37.4-1.amzn2023.0.4	amazonlinux	15 k

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

sudo adduser -m nagios

sudo passwd nagios

```
[ec2-user@ip-172-31-41-160 ~]$ sudo adduser -m nagios
[ec2-user@ip-172-31-41-160 ~]$ sudo passwd nagios
Changing password for user nagios.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-41-160 ~]$
```

Create a new user group & create a new directory for Nagios downloads using the following commands

sudo groupadd nagcmd

sudo usermod -a -G nagcmd nagios

sudo usermod -a -G nagcmd apache

mkdir ~/downloads

cd ~/downloads

Use **wget** to download the source zip files.

In this step, we are downloading, the latest version of nagios and the necessary plugins required to carry out the tasks of setting up a nagios server

wget <https://sourceforge.net/projects/nagios/files/latest/download>

```
[ec2-user@ip-172-31-41-160:~/downloads/nagios-4.5.5]
[ec2-user@ip-172-31-41-160 downloads]$ wget https://sourceforge.net/projects/nagios/files/latest/download
--2024-10-02 12:34:21-- https://sourceforge.net/projects/nagios/files/latest/download
Resolving sourceforge.net (sourceforge.net)... 172.64.150.145, 104.18.37.111, 2606:4700:4400::6812:256f, ...
Connecting to sourceforge.net (sourceforge.net)|172.64.150.145|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://downloads.sourceforge.net/project/nagios/nagios-4.x/nagios-4.5.5/nagios-4.5.5.tar.gz?ts=gAAAAABm_T3NmNSZPzP-6la2Tltvo0GCG7VVV7QGVH08n3tC240ehfMw7vhCoKbGHg2iIRxbmfugI10LccNfxtaoixg3jzKg3w3D0use_mirror=phoenixnapkr=[following]
--2024-10-02 12:34:21-- https://downloads.sourceforge.net/project/nagios/nagios-4.x/nagios-4.5.5/nagios-4.5.5.tar.gz?ts=gAAAAABm_T3NmNSZPzP-6la2Tltvo0GCG7VVV7QGVH08n3tC240ehfMw7vhCoKbGHg2iIRxbmfugI10LccNfxtaoixg3jzKg3w3D0use_mirror=phoenixnapkr=
Resolving downloads.sourceforge.net (downloads.sourceforge.net)|204.68.111.105|:443... connected.
Connecting to downloads.sourceforge.net (downloads.sourceforge.net)|204.68.111.105|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: 'download'

download                                         100%[=====] 1.97M 4.23MB/s in 0.5s

2024-10-02 12:34:22 (4.23 MB/s) - 'download' saved [2065473/2065473]
```

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

```
[ec2-user@ip-172-31-41-160:~/downloads/nagios-4.5.5]
[ec2-user@ip-172-31-41-160 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
--2024-10-02 12:34:46-- 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 (2.0M) [application/x-gzip]
Saving to: 'nagios-plugins-2.4.11.tar.gz'

nagios-plugins-2.4.11.tar.gz 100%[=====] 2.62M 7.48MB/s in 0.4s

2024-10-02 12:34:46 (7.48 MB/s) - 'nagios-plugins-2.4.11.tar.gz' saved [2753049/2753049]
```

```
[ec2-user@ip-172-31-92-249:~/downloads]
[ec2-user@ip-172-31-92-249 ~]$ cd ~/downloads
[ec2-user@ip-172-31-92-249 downloads]$ wget https://sourceforge.net/projects/nagios/files/latest/download
--2024-09-30 09:54:56-- https://sourceforge.net/projects/nagios/files/latest/download
Resolving sourceforge.net (sourceforge.net)... 172.64.150.145, 104.18.37.111, 2606:4700:4400::6812:256f, ...
Connecting to sourceforge.net (sourceforge.net)|172.64.150.145|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://downloads.sourceforge.net/project/nagios/nagios-4.x/nagios-4.5.5/nagios-4.5.5.tar.gz?ts=gAAAAABm-nVw9RdAvnMaShLf3gu4RXTSVxrtZ6fGxJvhVAOzpB1bPgbyzLMcDDAALgtEC1p0Kr0cgJNJ23bKktan1cJ0Vfkpg3D0use_mirror=netaactuate&r=[following]
--2024-09-30 09:54:56-- https://downloads.sourceforge.net/project/nagios/nagios-4.x/nagios-4.5.5/nagios-4.5.5.tar.gz?ts=gAAAAABm-nVw9RdAvnMaShLf3gu4RXTSVxrtZ6fGxJvhVAOzpB1bPgbyzLMcDDAALgtECl0OKrcpgJNj23bKktan1cJ0Vfkpg3D0use_mirror=netaactuate&r=
Resolving downloads.sourceforge.net (downloads.sourceforge.net)|204.68.111.105|:443... connected.
Connecting to downloads.sourceforge.net (downloads.sourceforge.net)|204.68.111.105|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: 'download'

download                                         100%[=====] 1.97M ---KB/s in 0.07s

2024-09-30 09:54:57 (29.8 MB/s) - 'download' saved [2065473/2065473]

[ec2-user@ip-172-31-92-249 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.9.tar.gz
--2024-09-30 09:56:53-- https://nagios-plugins.org/download/nagios-plugins-2.4.9.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: 2754403 (2.0M) [application/x-gzip]
Saving to: 'nagios-plugins-2.4.9.tar.gz'

nagios-plugins-2.4.9.tar.gz 100%[=====] 2.63M 7.54MB/s in 0.3s

2024-09-30 09:56:54 (7.54 MB/s) - 'nagios-plugins-2.4.9.tar.gz' saved [2754403/2754403]
```

Now, we run the next command in the following manner

tar zxvf <nagios-4.5.5 version> (for me it has gotten saved as 'download')

So i wrote tar zxvf download

```
[ec2-user@ip-172-31-41-160:~/downloads]
[ec2-user@ip-172-31-41-160 downloads]$ tar zxvf download
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/autocal.m4
nagios-4.5.5/autocom-macros/
nagios-4.5.5/autocom-macros/.gitignore
nagios-4.5.5/autocom-macros/CHANGELOG.md
nagios-4.5.5/autocom-macros/LICENSE
nagios-4.5.5/autocom-macros/LICENSE.md
nagios-4.5.5/autocom-macros/README.md
nagios-4.5.5/autocom-macros/add_group_user
nagios-4.5.5/autocom-macros/ax_nagios_get_distrib
nagios-4.5.5/autocom-macros/ax_nagios_get_files
nagios-4.5.5/autocom-macros/ax_nagios_get_inetd
nagios-4.5.5/autocom-macros/ax_nagios_get_init
nagios-4.5.5/autocom-macros/ax_nagios_get_os
nagios-4.5.5/autocom-macros/ax_nagios_get_paths
nagios-4.5.5/autocom-macros/ax_nagios_get_ssl
nagios-4.5.5/base/
nagios-4.5.5/base/.gitignore
nagios-4.5.5/base/Makefile.in
nagios-4.5.5/base/broker.c
nagios-4.5.5/base/checks.c
nagios-4.5.5/base/commands.c
nagios-4.5.5/base/config.c
nagios-4.5.5/base/events.c
nagios-4.5.5/base/flapping.c
nagios-4.5.5/base/logging.c
nagios-4.5.5/base/nagios.c
nagios-4.5.5/base/nagiosstats.c
nagios-4.5.5/base/nemodus.c
nagios-4.5.5/base/nerc.c
```

After which we are supposed to **change our directory** over there

For eg. **cd nagios-4.5.5...** depending on the version that we have downloaded

Next, Run this command (make sure that you are working inside nagios-4.x.x directory)

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

```
[ec2-user@ip-172-31-41-160:~/downloads/nagios-4.5.5]
[ec2-user@ip-172-31-41-160 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 for suffix of object files... o
checking whether the compiler supports GNU C... yes
checking whether gcc accepts -g... yes
checking for gcc option to enable C11 features... none needed
checking whether make sets ${MAKE}... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking for sys/wait.h that is POSIX.1 compatible... yes
checking for stdio.h... yes
checking for stdlib.h... yes
checking for string.h... yes
checking for inttypes.h... yes
checking for stdint.h... yes
checking for strings.h... yes
checking for sys/stat.h... yes
checking for sys/types.h... yes
checking for unistd.h... yes
checking for arpa/inet.h... yes
checking for ctype.h... yes
checking for dirent.h... yes
checking for errno.h... yes
checking for fcntl.h... yes
checking for getopt.h... yes
checking for grp.h... yes
checking for libgen.h... yes
checking for limits.h... yes
checking for math.h... yes
checking for netdb.h... yes
checking for netinet/in.h... yes
checking for pwd.h... yes
checking for regex.h... yes
checking for signal.h... yes
checking for socket.h... no
checking for stdarg.h... yes
```

```
checking for strerror... yes
checking for strtoul... yes
checking for unsetenv... yes
checking for type of socket size... size_t
checking for Kerberos include files... configure: WARNING: could not find include files
checking for pkg-config... pkg-config
checking for SSL headers... configure: error: Cannot find ssl headers
[ec2-user@ip-172-31-41-160 nagios-4.5.5]$
```

After running this command, we get an **error related to ssl header being absent**

For that purpose, we are to run the following command.

sudo yum install openssl-devel (for ssl header)

```
[ec2-user@ip-172-31-41-160 nagios-4.5.5]$ sudo yum install openssl-devel
Last metadata expiration check: 0:12:11 ago on Wed Oct 2 12:28:33 2024.
Dependencies resolved.

-----  
 Package           Architecture      Version            Repository      Size  
-----  
Installing:  
openssl-devel     x86_64          1:3.0.8-1.amzn2023.0.14      amazonlinux    3.0 M  
  
Transaction Summary  
-----  
Install 1 Package  
  
Total download size: 3.0 M  
Installed size: 4.7 M  
Is this ok [y/N]: y  
Downloading Packages:  
openssl-devel-3.0.8-1.amzn2023.0.14.x86_64.rpm   26 MB/s | 3.0 MB  00:00  
  
Total  
Running transaction check  
Transaction check succeeded.  
Running transaction test  
Transaction test succeeded.  
Running transaction  
Preparing :  
Installing  : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64  
Running scriptlet: openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64  
Verifying   : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64  
  
Installed:  
openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64  
  
Complete!
[ec2-user@ip-172-31-41-160 nagios-4.5.5]$
```

Now, Re-run `./configure --with-command-group=nagcmd`

After this, run **make all** command

```
[ec2-user@ip-172-31-41-160:~/downloads/nagios-4.5.5]$ make all
cd ./base & make
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o nagios.o ./nagios.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o broker.o broker.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o nebmods.o nebmods.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o ..//common/shared.o ..//common/shared.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o query-handler.o query-handler.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o workers.o workers.c
In function 'get_wproc_list':
inlined from 'get_workers' at workers.c:277:12:
workers.c:253:17: warning: "%s" directive argument is null [Wformat-overflow]
    log_debug_info(DEBUG_CHECKS, 1, "Found specialized worker(s) for '%s'", (slash && !slash != '/') ? slash : cmd_name);
^~~~~~
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o checks.o checks.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o config.o config.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o commands.o commands.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o events.o events.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o flapping.o flapping.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o logging.o logging.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o macros-base.o ..//common/macros.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o netutils.o netutils.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o notifications.o notifications.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o sehandlers.o sehandlers.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o utils.o utils.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o retention_base.o ./retention.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o xstatusdata.o ..//data/xsddefault.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o comments_base.o ..//common/comments.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o xcomments_base.o ..//data/xsddefault.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o objects_base.o ..//common/objects.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o xobjects_base.o ..//data/xodtemplate.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o statusdata_base.o ..//common/statusdata.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o xstatusdata_base.o ..//data/xsddefault.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o perfdata_base.o ./perfdata.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o xperfdata_base.o ..//data/xpdefaul.c
gcc -Wall -I . -I . -I ./lib -I .. -I include -I . -g -O2 -DHAVE_CONFIG_H -DNSCORE -c -o downtime_base.o ..//common/downtime.c
make -C ./lib

make[2]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/lib'
gcc -Wall -g 0 -I . -I ./include -DHAVE_CONFIG_H -c squeue.c -o squeue.o
gcc -Wall -g 0 -I . -I ./include -DHAVE_CONFIG_H -c kvvec.c -o kvvec.o
gcc -Wall -g 0 -I . -I ./include -DHAVE_CONFIG_H -c iocache.c -o iocache.o
gcc -Wall -g 0 -I . -I ./include -DHAVE_CONFIG_H -c iobroker.o
gcc -Wall -g 0 -I . -I ./include -DHAVE_CONFIG_H -c bitmap.o -o bitmap.o
gcc -Wall -g 0 -I . -I ./include -DHAVE_CONFIG_H -c dkhash.o -o dkhash.o
```

```

ec2-user@ip-172-31-41-160:~/downloads/nagios-4.5.5
on doing this. Pay particular attention to the docs on
object configuration files, as they determine what/how
things get monitored!
make install-webconf
  This installs the Apache config file for the Nagios
  web interface
make install-exfoliation
  This installs the Exfoliation theme for the Nagios
  web interface
make install-classicui
  This installs the classic theme for the Nagios
  web interface

*** Support Notes *****
If you have questions about configuring or running Nagios,
please make sure that you:
  - Look at the sample config files
  - Read the documentation on the Nagios Library at:
    https://library.nagios.com

before you post a question to one of the mailing lists.
Also make sure to include pertinent information that could
help others help you. This might include:
  - What version of Nagios you are using
  - What version of the plugins you are using
  - Relevant snippets from your config files
  - Relevant error messages from the Nagios log file

For more information on obtaining support for Nagios, visit:
  https://support.nagios.com

*****
Enjoy.

```

Run the following set of commands to ensure that
sudo make install

```

ec2-user@ip-172-31-41-160:~/downloads/nagios-4.5.5
[ec2-user@ip-172-31-41-160 nagios-4.5.5]$ sudo make install
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
for file in *.cgi; do \
    /usr/bin/install -c -s -m 775 -o nagios -g nagios $file /usr/local/nagios/sbin; \
done
make[2]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
cd ./html && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/html'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/media
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/stylesheets
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/contexthelp
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/docs
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/docs/images
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/js
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/images
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/images/logos
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/includes
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/ssi
/usr/bin/install -c -m 664 -o nagios -g nagios ./robots.txt /usr/local/nagios/share
/usr/bin/install -c -m 664 -o nagios -g nagios ./jsonquery.html /usr/local/nagios/share
rm -f /usr/local/nagios/share/index.html
rm -f /usr/local/nagios/share/main.html
rm -f /usr/local/nagios/share/side.html
rm -f /usr/local/nagios/share/map.html
rm -f /usr/local/nagios/share/rss/*
rm -f /usr/local/nagios/share/graph-header.html
rm -f /usr/local/nagios/share/histogram.html
rm -f /usr/local/nagios/share/histogram-form.html
rm -f /usr/local/nagios/share/histogram-graph.html
rm -f /usr/local/nagios/share/histogram-links.html
rm -f /usr/local/nagios/share/infobox.html
rm -f /usr/local/nagios/share/map.php

```

sudo make install-init

```
[ec2-user@ip-172-31-41-160:~/downloads/nagios-4.5.5]$ sudo make install-init
/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
```

sudo make install-config

```
[ec2-user@ip-172-31-41-160 nagios-4.5.5]$ sudo make install-config
/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/switch.cfg /usr/local/nagios/etc/objects/switch.cfg

*** Config files installed ***

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

sudo make install-webconf

```
[ec2-user@ip-172-31-41-160 nagios-4.5.5]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios.conf
if [ 0 -eq 1 ]; then \
    ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagios.conf; \
fi

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

[ec2-user@ip-172-31-41-160 nagios-4.5.5]$
```

Next, we are supposed to create a nagiosadmin account for nagios login along with password.
Specify the password twice.

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

```
[ec2-user@ip-172-31-41-160 nagios-4.5.5]$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
[ec2-user@ip-172-31-41-160 nagios-4.5.5]$
```

Restart Apache

sudo service httpd restart

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

cd ~/downloads

tar zxvf nagios-plugins-2.4.11.tar.gz

```
[ec2-user@ip-172-31-41-160:~/downloads]
[ec2-user@ip-172-31-41-160 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-41-160 nagios-4.5.5]$ cd ~/downloads
[ec2-user@ip-172-31-41-160 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/ltdmain.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
```

Compile and install plugins

cd nagios-plugins-2.4.11

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

Run the following command:

sudo chkconfig --add nagios

On running the above command

```
[ec2-user@ip-172-31-41-160 nagios-plugins-2.4.11]$ sudo chkconfig --add nagios
error reading information on service nagios: No such file or directory
```

If this is the output that one is getting, then it means that the init script is missing...

We can check this by running ls /etc/init.d/

```
[ec2-user@ip-172-31-92-249 nagios-plugins-2.4.9]$ ls /etc/init.d/
README  functions
[ec2-user@ip-172-31-92-249 nagios-plugins-2.4.9]$
```

With ls command, we must see a file named nagios, which i was not able to see

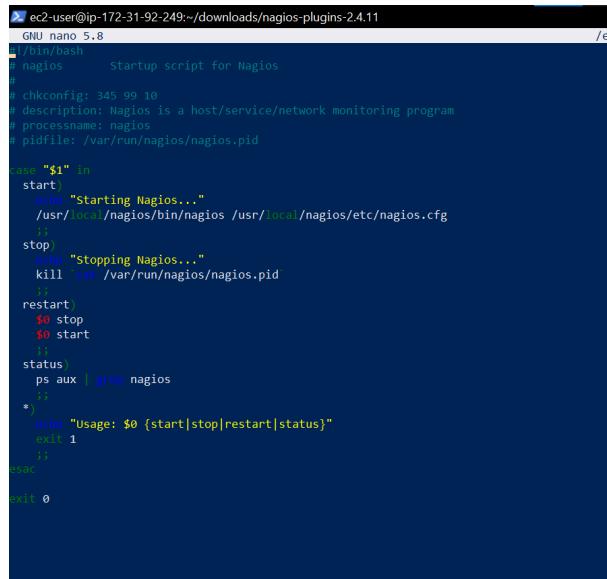
If the Init Script is Missing i.e If you don't see the nagios script in /etc/init.d/, you can create it manually. Here's how:

Run the following command:

sudo nano /etc/init.d/nagios

Within this file, paste the following script

```
#!/bin/bash
# nagios      Startup script for Nagios
#
# chkconfig: 345 99 10
# description: Nagios is a host/service/network monitoring program
# processname: nagios
# pidfile: /var/run/nagios/nagios.pid
case "$1" in
    start)
        echo "Starting Nagios..."
        /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg
        ;;
    stop)
        echo "Stopping Nagios..."
        kill `cat /var/run/nagios/nagios.pid`
        ;;
    restart)
        $0 stop
        $0 start
        ;;
    status)
        ps aux | grep nagios
        ;;
    *)
        echo "Usage: $0 {start|stop|restart|status}"
        exit 1
        ;;
esac
exit 0
```



The screenshot shows a terminal window titled 'ec2-user@ip-172-31-92-249:~/downloads/nagios-plugins-2.4.11'. The window displays the script content from the previous code block. The text is in white on a dark background, with syntax highlighting for commands like 'echo', 'kill', and 'ps'. The terminal window has a standard Linux-style interface with a title bar and a scroll bar.

```
#!/bin/bash
# nagios      Startup script for Nagios
#
# chkconfig: 345 99 10
# description: Nagios is a host/service/network monitoring program
# processname: nagios
# pidfile: /var/run/nagios/nagios.pid
case "$1" in
    start)
        echo "Starting Nagios..."
        /usr/local/nagios/bin/nagios /usr/local/nagios/etc/nagios.cfg
        ;;
    stop)
        echo "Stopping Nagios..."
        kill `cat /var/run/nagios/nagios.pid`
        ;;
    restart)
        $0 stop
        $0 start
        ;;
    status)
        ps aux | grep nagios
        ;;
    *)
        echo "Usage: $0 {start|stop|restart|status}"
        exit 1
        ;;
esac
exit 0
```

Make the Script Executable: After saving the file, run the following command to make it executable:

sudo chmod +x /etc/init.d/nagios

Run sudo chkconfig --add nagios again

And then run sudo chkconfig nagios on

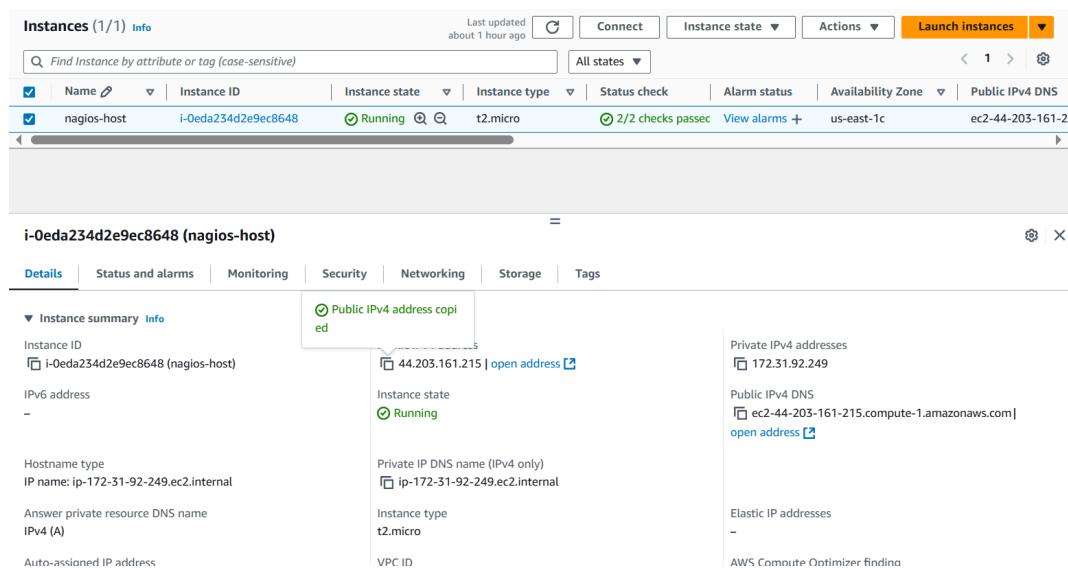
```
[ec2-user@ip-172-31-41-160 nagios-plugins-2.4.11]$ sudo nano /etc/init.d/nagios
[ec2-user@ip-172-31-41-160 nagios-plugins-2.4.11]$ sudo chmod +x /etc/init.d/nagios
[ec2-user@ip-172-31-41-160 nagios-plugins-2.4.11]$ sudo chkconfig --add nagios
[ec2-user@ip-172-31-41-160 nagios-plugins-2.4.11]$ sudo chkconfig nagios on
Note: Forwarding request to 'systemctl enable nagios.service'.
Synchronizing state of nagios.service with sysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable nagios
Created symlink /etc/systemd/system/multi-user.target.wants/nagios.service → /usr/lib/systemd/system/nagios.service.
[ec2-user@ip-172-31-41-160 nagios-plugins-2.4.11]$
```

sudo service nagios start

```
[ec2-user@ip-172-31-92-249 nagios-plugins-2.4.11]$ sudo service nagios start
Starting Nagios...
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
Nagios 4.5.5 starting... (PID=72261)
Local time is Tue Oct 01 20:59:58 UTC 2024.
wproc: Successfully registered manager as @wproc with query handler
wproc: Registry request: name=Core Worker 72265;pid=72265
wproc: Registry request: name=Core Worker 72264;pid=72264
wproc: Registry request: name=Core Worker 72263;pid=72263
wproc: Registry request: name=Core Worker 72262;pid=72262
Successfully launched command file worker with pid 72266
wproc: NOTIFY job 4 from worker Core Worker 72262 is a non-check helper but exited with return code 127
wproc: host=localhost; service=Swap Usage; contact=nagiosadmin
wproc: early_timeout=0; exited_ok=1; wait_status=32512; error_code=0;
wproc: stderr line 01: /bin/sh: line 1: /bin/mail: No such file or directory
wproc: stderr line 02: /usr/bin/printf: write error: Broken pipe
```

Get your public IPv4 address from your instance. We will require it for connecting to our nginx server



Browse for this url: http://<your_public_ip_address>/nagios

The browser may ask you for your nagios credentials which set in the earlier steps

The username is nagiosadmin and enter the password that you set earlier

The screenshot shows the Nagios Core web interface. The top navigation bar indicates the URL is 34.229.45.75/nagios and the connection is not secure. The main header features the Nagios Core logo and the text "Process running with PID 62668". On the left, a vertical sidebar contains a navigation menu with sections like General, Current Status, Service Groups, Problems, Reports, and a search bar. The main content area includes a "Get Started" section with bullet points about monitoring infrastructure, changing the look, extending with addons, and getting support. It also features a "Latest News" section and a "Don't Miss..." section. A "Quick Links" sidebar on the right provides links to Nagios Library, Labs, Exchange, Support, and the official website. A "Page Tour" button is located on the far right.

Conclusion:

In this experiment, we successfully installed and configured Nagios Core on an Amazon Linux EC2 instance, showcasing its role in continuous monitoring within a DevOps environment. We learned about user management and service configuration, emphasizing Nagios's ability to monitor systems and networks effectively. This experience laid the groundwork for enhancing infrastructure reliability and integrating advanced monitoring strategies in future projects.

Adv DevOps Exp 10

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

Monitoring Using Nagios:

Step 1: To Confirm Nagios is running on the server side Perform the following command on your Amazon Linux Machine (Nagios-host).

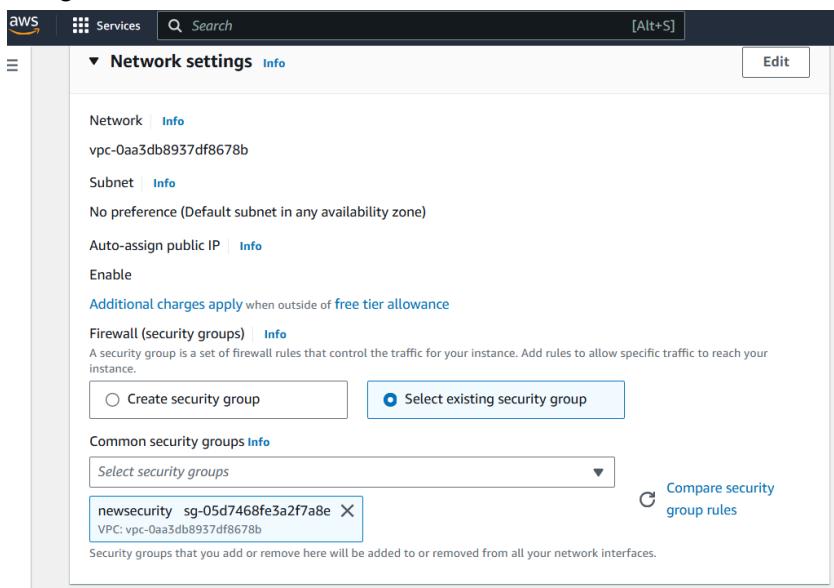
Run this command **sudo systemctl status**

```
ec2-user@ip-172-31-41-160:~$ /downloads/nagios-plugins-2.4.11
[ec2-user@ip-172-31-41-160 nagios-plugins-2.4.11]$ sudo systemctl status
● ip-172-31-41-160.ec2.internal
    State: running
      Units: 296 loaded (incl. loaded aliases)
        Jobs: 0 queued
       Failed: 0 units
     Since: Wed 2024-10-02 12:28:05 UTC; 33min ago
    Systemd: 252.23-2.amzn2023
   CGroup: /
           └─init.scope
             ├─1 /usr/lib/systemd/systemd --switched-root --system --deserialize=32
             ├─system.slice
             ├─acpid.service
             ├─acpid.slice
             ├─1938 /usr/bin/systemd-inhibit --what=handle-suspend-key:handle-hibernate-key --who=noah "--why=acpid instead" --mode=block /usr/sbin/acpid -f
             ├─2059 /usr/sbin/acpid -f
             ├─amazon-ssm-agent.service
             ├─2141 /usr/bin/amazon-ssm-agent
             ├─atd.service
             ├─2152 /usr/sbin/atd -f
             ├─audited.service
             ├─1768 /sbin/audited
             ├─chronynd.service
             ├─2175 /usr/sbin/chronynd -F 2
             ├─dbus-broker.service
             ├─1946 /usr/bin/dbus-broker-launch --scope system --audit
             ├─1954 dbus-broker --log 4 --controller 9 --machine-id ec2e4d759a3e2f6fe850b14e4cdacabe --max-bytes 536870912 --max-fds 4096 --max-matches 16384 --audit
             ├─gssproxy.service
             ├─1959 /usr/sbin/gssproxy -D
             ├─httpd.service
             ├─49553 /usr/sbin/httpd -DFOREGROUND
             ├─49555 /usr/sbin/httpd -DFOREGROUND
             ├─49556 /usr/sbin/httpd -DFOREGROUND
             ├─49557 /usr/sbin/httpd -DFOREGROUND
             ├─49558 /usr/sbin/httpd -DFOREGROUND
             ├─62800 /usr/sbin/httpd -DFOREGROUND
             └─libstoragemgmt.service
               └─1940 /usr/bin/lsm -d
```

Step 2: Before we begin,

To monitor a Linux machine, create an **Ubuntu 20.04 server** EC2 Instance in AWS.

Provide it with the **same security group** as the Nagios Host and name it 'nagios-client' alongside the host.



Key pair (login) Info

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

Key pair name - *required*

mohit ▼ ⟳ Create new key pair

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
nagios-host	i-03facef442a77494d	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-34-229-45-75
nagios-client	i-0b934b61f21351c1b	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-54-172-92-221

Step 3: TO BE DONE IN THE Nagios-host TERMINAL

In the nagios-host terminal, run this command

ps -ef | grep nagios

```
[ec2-user@ip-172-31-41-160 nagios-plugins-2.4.11]$ ps -ef | grep nagios
ec2-user 63115 2315 0 13:03 pts/0 00:00:00 grep --color=auto nagios
[ec2-user@ip-172-31-41-160 nagios-plugins-2.4.11]$ ■
```

To become a root user, run '**sudo su**' and make two directories using the following commands. If one is running these commands in windows powershell, make sure that he/she copies it line by line as powershell might make an error while interpreting multiple lines

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

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

```
[ec2-user@ip-172-31-92-249 ~]$ sudo su
[root@ip-172-31-92-249 ec2-user]# mkdir /usr/local/nagios/etc/objects/monitorhosts
[root@ip-172-31-92-249 ec2-user]# mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-92-249 ec2-user]#
```

Copy the sample localhost.cfg file to linuxhost folder. Use the following mentioned command to achieve it

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

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

Open linuxserver.cfg using nano and make the following changes. This is a conf type file in which we will have to modify the configurations in way which will help us specify the hosts and clients to be monitored

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

Changes to be made:

1. Change the hostname to linux-server (EVERYWHERE ON THE FILE)
2. Change address to the public IP address of your LINUX CLIENT.
3. Change hostgroup_name under hostgroup to linux-servers1

```
# HOST DEFINITION
#
#####
# Define a host for the local machine

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

    host_name    linux-server
    alias        localhost
    address     54.172.92.226
}

#####
# Define an optional hostgroup for Linux machines

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

IMP: Everywhere else on the file, change the hostname to linux-server instead of localhost.

Open the Nagios Config file and add the following line

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

Add the following line in the file and save

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

```
# OBJECT CONFIGURATION FILE(S)
# These are the object configuration files in which you define hosts,
# host groups, contacts, contact groups, services, etc.
# You can split your object definitions across several config files
# if you wish (as shown below), or keep them all in a single config file.

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

# Definitions for monitoring the local (Linux) host
cfg_file=/usr/local/nagios/etc/objects/localhost.cfg
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/
# Definitions for monitoring a Windows machine
#cfg_file=/usr/local/nagios/etc/objects/windows.cfg
```

Verify the configuration files by running the following command

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

```
[root@ip-172-31-41-160 nagios-plugins-2.4.11]# /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-172-31-41-160 nagios-plugins-2.4.11]#
```

You are good to go if there are no errors.

Restart the nagios service

service nagios restart

And by running sudo systemctl status nagios, we can again check whether our server is running or not

```

root@ip-172-31-41-160:/tmp/nagios-plugins-2.4.11]
[root@ip-172-31-41-160 nagios-plugins-2.4.11]# sudo systemctl restart nagios
[root@ip-172-31-41-160 nagios-plugins-2.4.11]# sudo systemctl status nagios
● nagios.service - Nagios Core 5
   Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Wed 2024-10-02 13:20:17 UTC; 7s ago
       Docs: https://www.nagios.org/documentation
   Process: 78776 ExecStart=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
  Process: 78777 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
 Main PID: 78778 (nagios)
   Tasks: 6 (limit: 1112)
      Memory: 4.0M
        CPU: 24ms
      CGroup: /system.slice/nagios.service
          └─78778 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
              ├─78779 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/nagios.gh
              ├─78780 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/nagios.gh
              ├─78781 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/nagios.gh
              ├─78782 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/nagios.gh
              └─78783 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 02 13:20:17 ip-172-31-41-160.ec2.internal nagios[78778]: qh: echo service query handler registered
Oct 02 13:20:17 ip-172-31-41-160.ec2.internal nagios[78778]: qh: help for the query handler registered
Oct 02 13:20:17 ip-172-31-41-160.ec2.internal nagios[78778]: wpoc: Successfully registered nagios @proc with query handler
Oct 02 13:20:17 ip-172-31-41-160.ec2.internal nagios[78778]: wpoc: Registry request: name=Core Worker 78782;pid=78782
Oct 02 13:20:17 ip-172-31-41-160.ec2.internal nagios[78778]: wpoc: Registry request: name=Core Worker 78781;pid=78781
Oct 02 13:20:17 ip-172-31-41-160.ec2.internal nagios[78778]: wpoc: Registry request: name=Core Worker 78780;pid=78780
Oct 02 13:20:17 ip-172-31-41-160.ec2.internal nagios[78778]: wpoc: Registry request: name=Core Worker 78779;pid=78779
Oct 02 13:20:17 ip-172-31-41-160.ec2.internal nagios[78778]: Successfully launched command file worker with pid 78783
Oct 02 13:20:21 ip-172-31-41-160.ec2.internal nagios[78778]: HOST ALERT: linux-server;UP;SOFT;1;PING OK - Packet loss = 0%, RTA = 0.93 ms
Oct 02 13:20:24 ip-172-31-41-160.ec2.internal nagios[78778]: SERVICE ALERT: localhost;HTTP;WARNING;HARD;4;HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes in 0.0
[root@ip-172-31-41-160 nagios-plugins-2.4.11]# sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
     Drop-In: /usr/lib/systemd/system/httpd.service.d
       └─php-fpm.conf
     Active: active (running) since Wed 2024-10-02 12:47:56 UTC; 33min ago
       Docs: man:httpd.service(8)
   Main PID: 49553 (httpd)
     Status: "Total requests: 26; Idle/Busy workers 100/0;Requests/sec: 0.0129; Bytes served/sec: 94 B/sec"
     Tasks: 230 (limit: 1112)
        Memory: 1.7M
          CPU: 1.416s
        CGroup: /system.slice/httpd.service
            └─49553 /usr/sbin/httpd -DFOREGROUND

```

Step 4: TO BE DONE IN THE Nagios-client TERMINAL

Now it is time to switch to the client machine.

SSH into the machine or simply use the EC2 Instance Connect feature.

```

PS C:\WINDOWS\system32> cd C:\Users\DEll\Downloads
PS C:\Users\DEll\Downloads> ssh -i "mohit.pem" ubuntu@ec2-54-172-92-226.compute-1.amazonaws.com
The authenticity of host 'ec2-54-172-92-226.compute-1.amazonaws.com (54.172.92.226)' can't be established.
ECDSA key fingerprint is SHA256:e/WkFQRuHSpjQ5hDMA0dku8msNHETN9SAgzEy53E.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-172-92-226.compute-1.amazonaws.com,54.172.92.226' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1016-aws x86_64)

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

System information as of Wed Oct  2 13:26:11 UTC 2024

System load:  0.0           Processes:          104
Usage of /:   22.8% of  6.71GB  Users logged in:    0
Memory usage: 20%           IPv4 address for enX0: 172.31.36.100
Swap usage:   0%

* Ubuntu Pro delivers the most comprehensive open source security and
  compliance features.

  https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

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

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

```

Make a package index update and install gcc, nagios-nrpe-server and the plugins. Run the following commands to achieve the same.

sudo apt update -y

sudo apt install gcc -y

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

Open nrpe.cfg file to make changes.

sudo nano /etc/nagios/nrpe.cfg

Under allowed_hosts, add your nagios host IP address like so

```
ubuntu@ip-172-31-36-100: ~
GNU nano 7.2

#
# 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=127.0.0.1,34.229.45.75

#
# 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
```

Now restart the NRPE server by this command.

sudo systemctl restart nagios-nrpe-server

```
ubuntu@ip-172-31-36-100: ~$ sudo systemctl restart nagios-nrpe-server
ubuntu@ip-172-31-36-100: ~$
```

Run the following command in the Nagios-host terminal

sudo systemctl status nagios

```
[root@ip-172-31-41-160 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 Wed 2024-10-02 13:20:17 UTC; 15min ago
     Docs: https://www.nagios.org/documentation
 Main PID: 78778 (nagios)
   Tasks: 6 (limit: 1112)
    Memory: 4.3M
       CPU: 403ms
      CGroup: /system.slice/nagios.service
              └─78778 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 02 13:22:54 ip-172-31-41-160.ec2.internal nagios[78778]: SERVICE NOTIFICATION: nagiosadmin;localhost;Swap Usage;CRITICAL;notify-service-by-email;SWAP CRITICAL - 0% free (0 MB out of 0 MB)
Oct 02 13:22:54 ip-172-31-41-160.ec2.internal nagios[78778]: wproc: NOTIFY job 3 from worker Core Worker 78782 is a non-check helper but exited with return code 127
Oct 02 13:22:54 ip-172-31-41-160.ec2.internal nagios[78778]: wproc: host=localhost; service=Swap Usage; contact=nagiosadmin
Oct 02 13:22:54 ip-172-31-41-160.ec2.internal nagios[78778]: wproc: early timeout=0; exited_ok=1; wait_status=2512; error_code=0;
Oct 02 13:22:54 ip-172-31-41-160.ec2.internal nagios[78778]: stderr line 01: /bin/sh: line 1: /bin/mail: No such file or directory
Oct 02 13:22:54 ip-172-31-41-160.ec2.internal nagios[78778]: stderr line 02: /usr/bin/printf: write error: Broken pipe
Oct 02 13:23:13 ip-172-31-41-160.ec2.internal nagios[78778]: SERVICE ALERT: linux-server;Total Processes;OK;HARD;1;PROCS OK: 37 processes with STATE = RSZDT
Oct 02 13:23:50 ip-172-31-41-160.ec2.internal nagios[78778]: SERVICE ALERT: linux-server;Current Load;OK;HARD;1;OK - load average: 0.01, 0.07, 0.04
Oct 02 13:24:28 ip-172-31-41-160.ec2.internal nagios[78778]: SERVICE ALERT: linux-server;Current Users;OK;HARD;1;USERS OK - 2 users currently logged in
Oct 02 13:24:46 ip-172-31-41-160.ec2.internal nagios[78778]: SERVICE ALERT: localhost;Current Users;OK;HARD;1;USERS OK - 2 users currently logged in
Lines 1-26/26 (END)
```

Step 5: Visiting your nagios server using your nagios-host ip address

Open up your browser and look for http://<public_ip_address_of_nagios-host>/nagios

The screenshot shows the Nagios Core 4.5.5 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 78778". The left sidebar has sections for General, Current Status (Tactical Overview, Map, Hosts, Services, Host Groups, Service Groups), and Reports (Availability, Trends, Alerts, History, Summary, Histogram, Notifications, Event Log). The main content area includes a "Get Started" section with bullet points about monitoring, a "Latest News" section, a "Don't Miss..." section, and a "Quick Links" section with links to Nagios Library, Labs, Exchange, Support, and the official websites.

Click on Hosts.

The screenshot shows the "Host Status Details For All Host Groups" table. It lists two hosts: "linux-server" and "localhost", both marked as "UP". The table columns include Host, Status, Last Check, Duration, and Status Information. The status information for both hosts indicates "PING OK - Packet loss = 0%, RTA = 0.84 ms" and "PING OK - Packet loss = 0%, RTA = 0.04 ms" respectively. The left sidebar is identical to the previous screenshot, showing the Nagios Core 4.5.5 interface.

Click on linux-server to view host information

The screenshot shows the Nagios web interface for a host named 'localhost' (linux-server). The left sidebar includes sections for General, Home, Documentation, Current Status (selected), and Reports. The 'Current Status' section displays various metrics and status indicators. The 'Host Information' panel shows the host's name, last update time, and a link to its configuration. The 'Host State Information' panel provides detailed status for the host, including its current state (UP), performance data, and check history. The 'Host Commands' panel lists various actions that can be taken for this host, such as enabling or disabling checks, scheduling downtime, or sending notifications. The bottom right corner shows a 'Page Tour' button.

We can even navigate to the services section, which explicitly mentions the status, duration, checks, information about the numerous services present on our hosts

The screenshot shows the Nagios web interface displaying service status details for all hosts. The left sidebar includes sections for General, Home, Documentation, Current Status (selected), and Reports. The 'Service Status Details For All Hosts' panel lists various services across different hosts, including their status, last check time, duration, and attempt count. A table below provides a summary of host status totals (Up, Down, Unreachable, Pending) and service status totals (Ok, Warning, Unknown, Critical, Pending). The bottom right corner shows a 'Page tour' button.

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

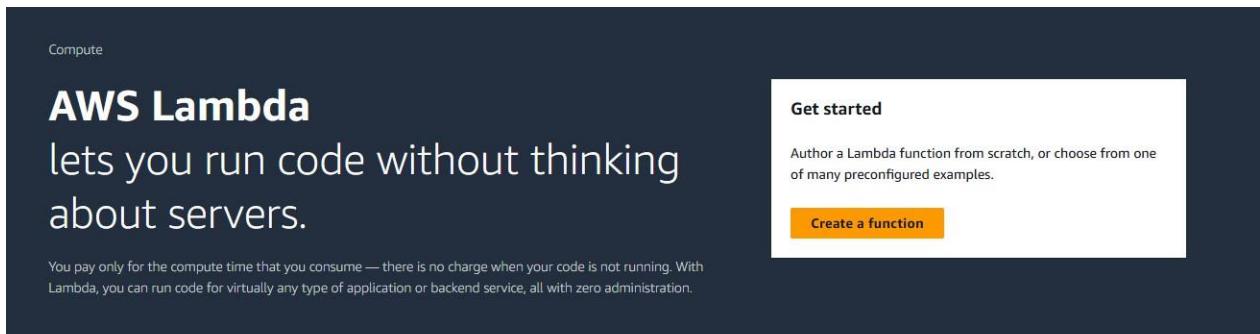
Aim: To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

Steps:

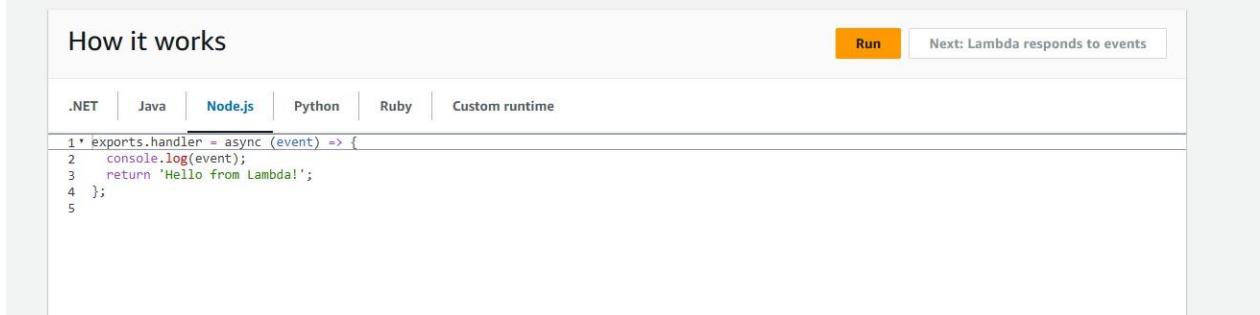
1. Go to AWS ACADEMY.

2. **Create the lambda function:**

Firstly, Search lambda, then Open lambda and then click on create function button.



The screenshot shows the AWS Lambda landing page under the Compute section. It features a large heading "AWS Lambda" with the tagline "lets you run code without thinking about servers." Below the heading, a paragraph explains that users pay only for compute time and can run code for virtually any type of application or backend service. On the right side, there is a "Get started" box with a "Create a function" button.



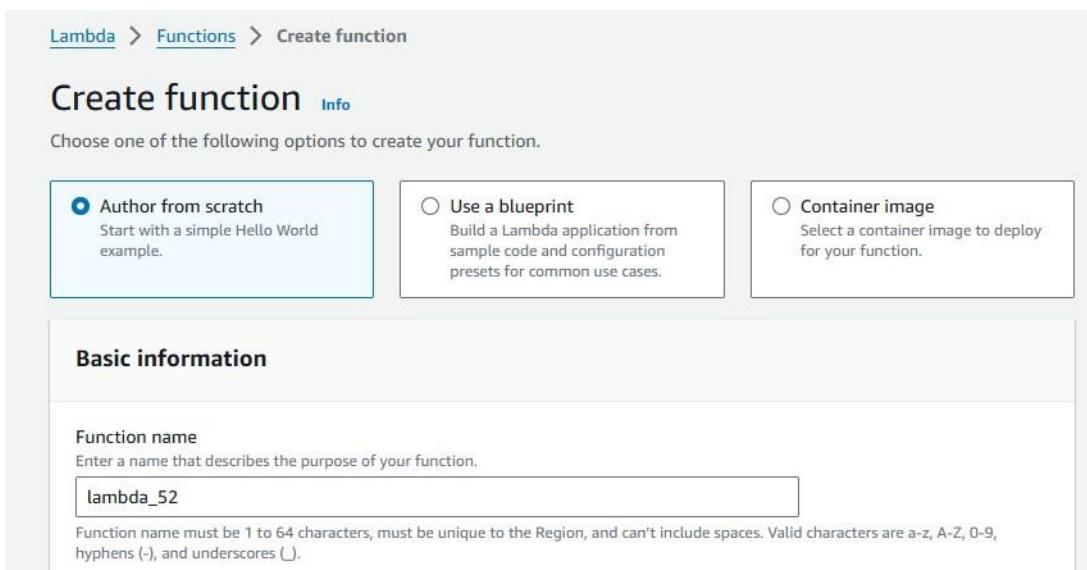
The screenshot shows the "How it works" section. It includes a "Run" button and a link to "Next: Lambda responds to events". Below this, there is a code editor showing a sample Node.js code snippet:

```

1* exports.handler = async (event) => {
2  console.log(event);
3  return 'Hello from Lambda!';
4 };
5

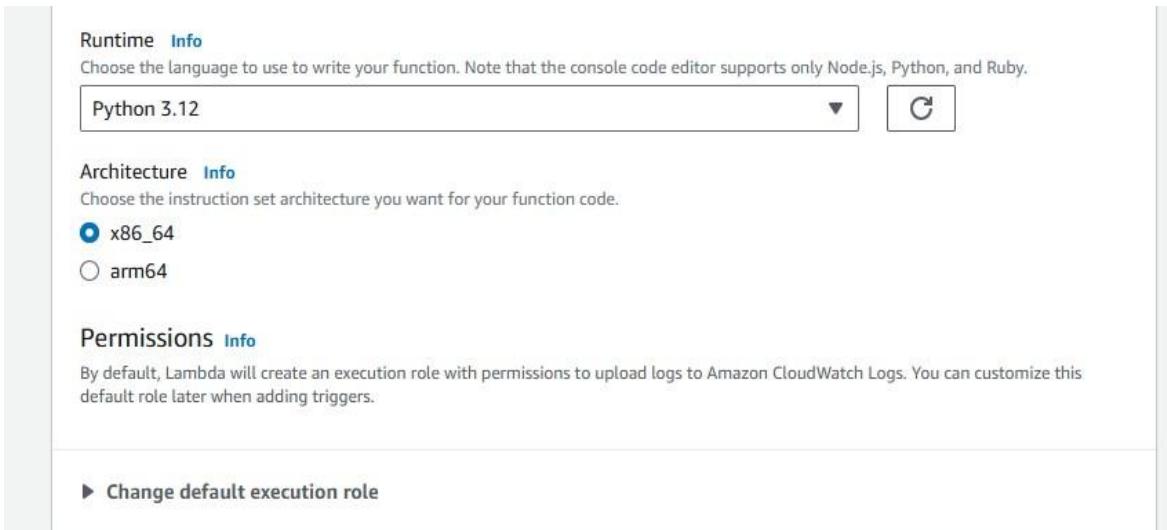
```

3. Now Give a name to your Lambda function,

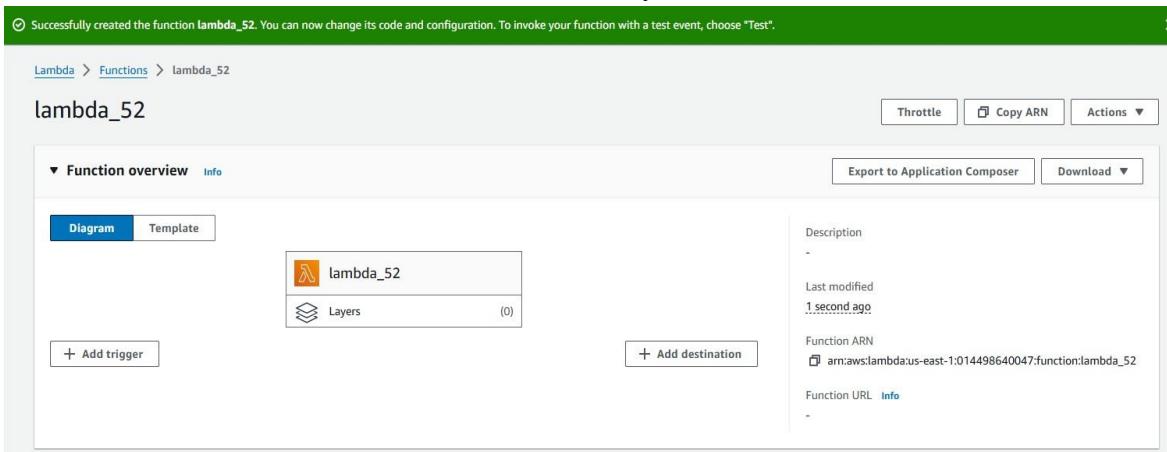


The screenshot shows the "Create function" wizard. It starts with a navigation bar: "Lambda > Functions > Create function". The main section is titled "Create function" with an "Info" link. It asks the user to choose one of three options: "Author from scratch", "Use a blueprint", or "Container image". The "Author from scratch" option is selected. Below this, there is a "Basic information" section where the "Function name" is set to "lambda_52". A note at the bottom states that the function name must be between 1 and 64 characters, unique to the region, and cannot include spaces, hyphens, or underscores.

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



Thus the Lambda function was created successfully.



5. Code Source Section



So now to edit the basic settings, go to configuration then click on edit.

The screenshot shows the AWS Lambda Configuration page. The 'Configuration' tab is selected. On the left, a sidebar lists 'General configuration', 'Triggers', 'Permissions', 'Destinations', 'Function URL', 'Environment variables', and 'Tags'. The main area displays 'General configuration' settings:

Description	Memory	Ephemeral storage
-	128 MB	512 MB
Timeout	SnapStart	None
0 min 3 sec	Info	

An 'Edit' button is located in the top right corner of the configuration table.

Now enter a description and change Memory and Timeout. Here, I've changed the Timeout period to 1 sec.

The screenshot shows the 'Edit basic settings' page for a Lambda function. The 'Basic settings' tab is selected. The configuration fields are as follows:

- Description - optional:** General Settings
- Memory:** 128 MB (Set memory to between 128 MB and 10240 MB)
- Ephemeral storage:** 512 MB (Set ephemeral storage (/tmp) to between 512 MB and 10240 MB)
- SnapStart:** None (Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the SnapStart compatibility considerations.)
- Timeout:** 0 min 1 sec
- Execution role:** Use an existing role (Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.)
- Existing role:** service-role/lambda_52-role-gxryrwck8 (Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.)

- Now Click on the Test tab then select Create a new event, give a name to the event here i have given name as “new_event_lambda” and then select event sharing to private, and select hello-world template.

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

Test event action

- Create new event
- Edit saved event

Event name

Event sharing settings

- Private
- Shareable

Template - optional

hello-world

Event JSON

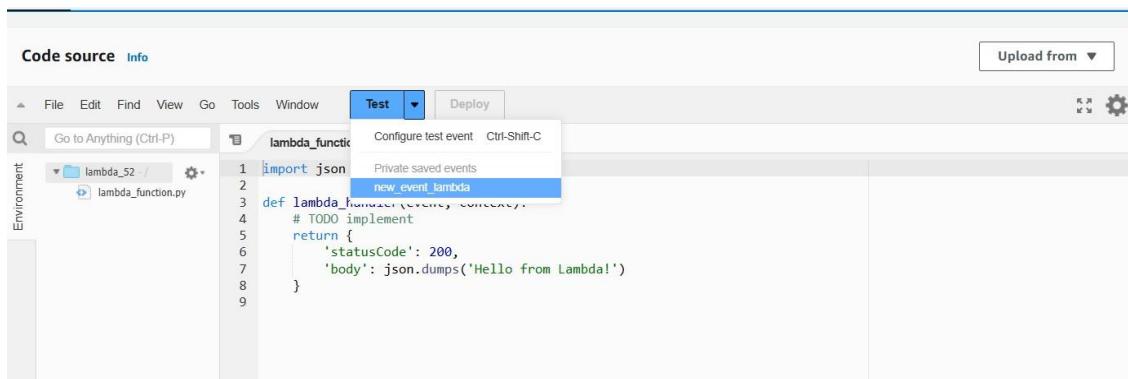
```

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

```

Format JSON

7. **Testing & Deployment:** Now In Code section select the created event (our_event) from the dropdown of test ,then click on test .



8. Now you will see the following output.

Execution results

Status: Succeeded | Max memory used: 32 MB | Time: 2.69 ms

Test Event Name
new_event_lambda

Response

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

Function Logs

```
START RequestId: 1072d581-ba0d-4d38-9db7-28086419d7f9 Version: $LATEST
END RequestId: 1072d581-ba0d-4d38-9db7-28086419d7f9
REPORT RequestId: 1072d581-ba0d-4d38-9db7-28086419d7f9 Duration: 2.69 ms Billed Duration: 3 ms Memory Size: 128 MB Max Memory Used: 32 MB Init Duration: 0 ms
```

Request ID
1072d581-ba0d-4d38-9db7-28086419d7f9

9. **Editing the given code into our own by adding a string or you can add anything**
You can edit your lambda function code. Here I have created a new string name “new_string” and

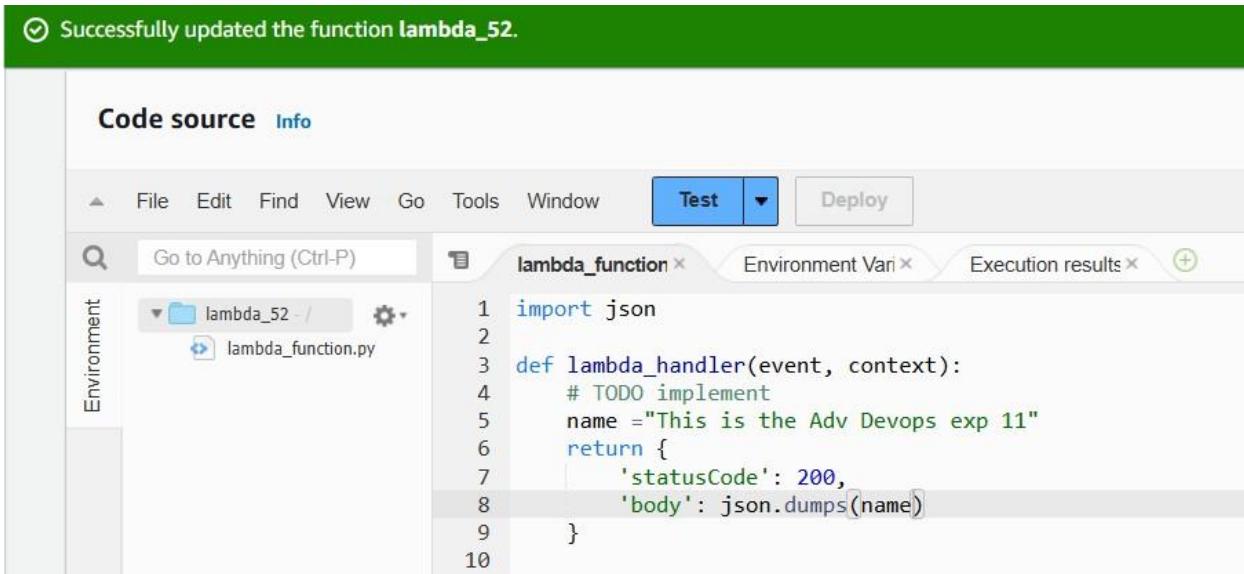
assigned a string to it.



The screenshot shows a code editor window titled "lambda_function". The file path is "/lambda_function.py". The code defines a lambda function "lambda_handler" that returns a JSON response with status code 200 and a body containing the string "This is the Adv Devops exp 11".

```
1  import json
2
3  def lambda_handler(event, context):
4      # TODO implement
5      name = "This is the Adv Devops exp 11"
6      return {
7          'statusCode': 200,
8          'body': json.dumps(name)
9      }
10
```

Now save it by *ctrl+s* and then click finally on deploy to deploy the changes.



The screenshot shows the AWS Lambda console. A green banner at the top indicates "Successfully updated the function lambda_52.". Below, the "Code source" tab is selected. The "Test" button is highlighted. The code editor shows the same lambda function code as before. The left sidebar shows the project structure with "lambda_52 - /" and "lambda_function.py".

Code source Info

File Edit Find View Go Tools Window Test Deploy

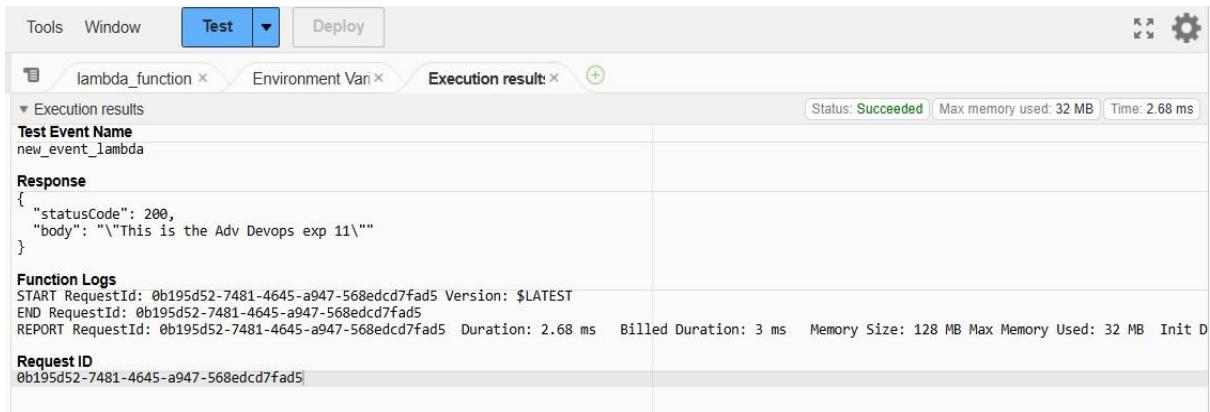
λ Go to Anything (Ctrl-P)

Environment

lambda_function Environment Vari Execution results +

```
1  import json
2
3  def lambda_handler(event, context):
4      # TODO implement
5      name = "This is the Adv Devops exp 11"
6      return {
7          'statusCode': 200,
8          'body': json.dumps(name)
9      }
10
```

10. Testing and redeploying changes Now click on the test and observe the output. Thus Output gives status code 200. This deployment is done successfully.



The screenshot shows the AWS Lambda Test interface. The top navigation bar includes 'Tools', 'Window', 'Test' (which is selected), and 'Deploy'. Below the navigation is a toolbar with icons for file operations and settings. The main area displays the 'Execution results' tab for a function named 'lambda_function'. The 'Test Event Name' is set to 'new_event_lambda'. The 'Response' section shows a JSON object with 'statusCode': 200 and 'body': '\"This is the Adv Devops exp 11\"'. The 'Function Logs' section contains log entries for 'START', 'END', and 'REPORT' requests, detailing the request ID, version, duration, and memory usage. The 'Request ID' is listed as '0b195d52-7481-4645-a947-568edcd7fad5'.

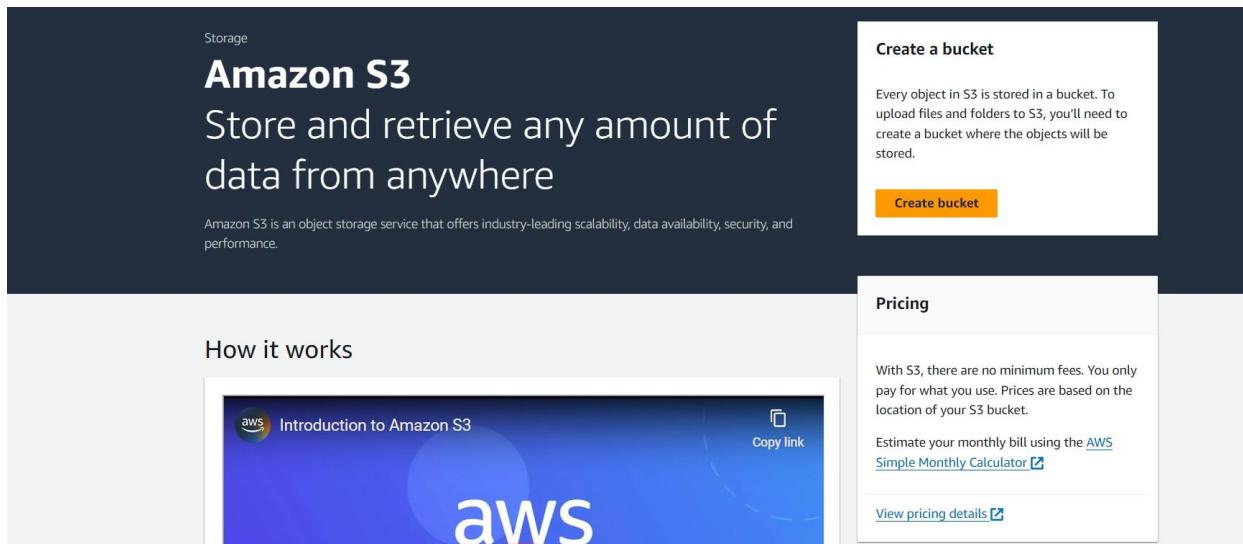
Conclusion:

In this experiment, we successfully created an AWS Lambda function using Python as the chosen language. After configuring the basic settings, we adjusted the timeout to 1 second, tested the function, and deployed it. The deployment was successful. We then modified the Lambda function's code and redeployed it to observe real-time changes. This process highlighted the ease of using AWS Lambda for building serverless applications. However, one issue we encountered was that, while we initially added the code source in Python, we could have chosen other supported languages as well.

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

Steps :

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



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

The screenshot shows the "Create bucket" configuration page. It starts with a general configuration section where the AWS Region is set to "US East (N. Virginia) us-east-1". The "Bucket type" section shows "General purpose" selected (indicated by a blue border). Other options like "Directory" are also listed. Below this, the "Bucket name" field contains "52bucket". A note below the name field states that the name must be unique and follow naming rules, with a link to "See rules for bucket naming". The "Copy settings from existing bucket - optional" section is present but empty. At the bottom, there's a "Choose bucket" button and a note about the prefix format.

Block Public Access settings for this bucket

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

Block all public access

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

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

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

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

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

Successfully created bucket "52bucket". To upload files and folders, or to configure additional bucket settings, choose [View details](#).

Amazon S3 > Buckets

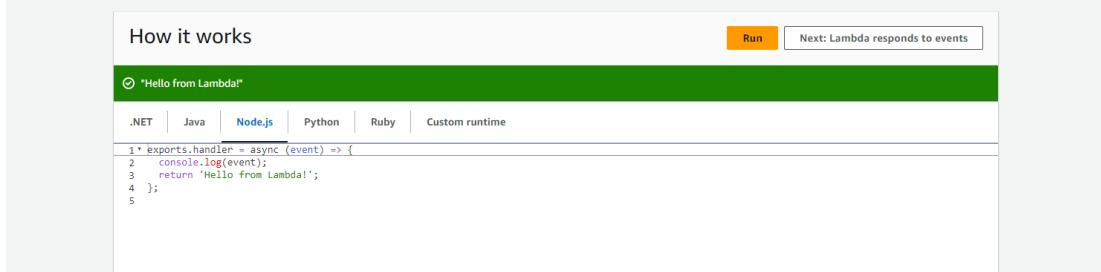
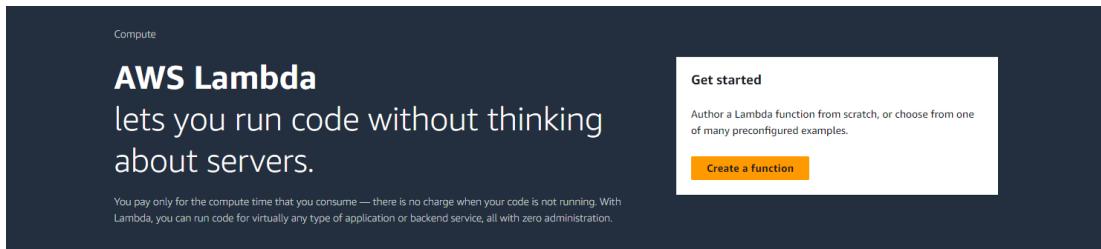
Account snapshot - updated every 24 hours [All AWS Regions](#)
Storage lens provides visibility into storage usage and activity trends. [Learn more](#)

[View Storage](#)

General purpose buckets | Directory buckets

General purpose buckets (2) Info All AWS Regions		Edit	Copy ARN	Empty	Delete
Buckets are containers for data stored in S3.					
<input type="text"/> Find buckets by name					
Name	AWS Region	IAM Access Analyzer	Creation date		
52bucket	US East (N. Virginia) us-east-1	View analyzer for us-east-1	October 10, 2024, 18:21:15 (UTC+05:30)		

3. Search and Open lambda console and click on create function button.



4. Now Give a name to your Lambda function, Select the language to write your function. Here I have chosen python3.12, Architecture as x86, and Execution role to Create a new role with basic Lambda permissions. Note that the console code editor supports only Node.js, Python, and Ruby

Lambda > Functions > Create function

Create function Info

Choose one of the following options to create your function.

- Author from scratch

Start with a simple Hello World example.
- Use a blueprint

Build a Lambda application from sample code and configuration presets for common use cases.
- Container image

Select a container image to deploy for your function.

Basic information

Function name
Enter a name that describes the purpose of your function.

Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).

Runtime Info
Choose the language to use for your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Architecture Info
Choose the instruction set architecture you want for your function code.
 x86_64
 arm64

Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

Change default execution role

Additional Configurations
Use additional configurations to set up code signing, function URL, tags, and Amazon VPC access for your function.

5. The Lambda function was created successfully.

The screenshot shows the AWS Lambda Functions overview page. At the top, a green banner indicates: "Successfully created the function lambda_52_exp12. You can now change its code and configuration. To invoke your function with a test event, choose 'Test'." Below the banner, the function name "lambda_52_exp12" is displayed. On the right, there are buttons for "Throttle", "Copy ARN", and "Actions". Under the "Function overview" section, there are tabs for "Diagram" (selected) and "Template". The diagram shows a single function box labeled "lambda_52_exp12" with a "Layers" section below it. Buttons for "+ Add trigger" and "+ Add destination" are present. On the right, there is a "Description" section with details like "Last modified 2 seconds ago", "Function ARN arn:aws:lambda:us-east-1:014498640047:function:lambda_52_exp12", and a "Function URL" link.

6. Then Go into the code section. You will see some default code there.

The screenshot shows the AWS Lambda Code source editor. The top navigation bar includes tabs for "Code" (selected), "Test", "Monitor", "Configuration", "Aliases", and "Versions". Below the tabs, the "Code source" section is selected. The interface includes a toolbar with "File", "Edit", "Find", "View", "Go", "Tools", "Window", "Test" (selected), and "Deploy". A sidebar on the left shows the "Environment" and a file tree with "lambda_52_exp12" and "lambda_function.py". The main area displays the Python code for the lambda function:

```

1 import json
2
3 def lambda_handler(event, context):
4     # TODO implement
5     return {
6         'statusCode': 200,
7         'body': json.dumps('Hello from Lambda!')
8     }
9

```

7. To Edit the basic settings go to configuration then click on edit setting.

The screenshot shows the AWS Lambda Configuration page. The top navigation bar includes tabs for "Configuration" (selected), "Aliases", and "Versions". Below the tabs, the "General configuration" section is selected. The "Edit" button is visible in the top right corner. The configuration table contains the following data:

Description	Memory	Ephemeral storage
-	128 MB	512 MB
Timeout	SnapStart	
0 min 3 sec	None	

8. Here, enter a description which is optional and change Memory and Timeout.
I've changed the Timeout period to 2 sec.

Edit basic settings

Basic settings [Info](#)

Description - *optional*
new_func1

Memory [Info](#)
Your function is allocated CPU proportional to the memory configured.
128 MB
Set memory to between 128 MB and 10240 MB.

Ephemeral storage [Info](#)
You can configure up to 10 GB of ephemeral storage (/tmp) for your function. [View pricing](#)
512 MB
Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.

SnapStart [Info](#)
Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the [SnapStart compatibility considerations](#).
None
Supported runtimes: Java 11, Java 17, Java 21.

Timeout
0 min 2 sec

9. Now Click on the Test then select Create a new event, give a name to the event. Here I have given name as ‘test_newevent’ and then select Event Sharing to private, and select s3 put template.

Code | **Test** | Monitor | Configuration | Aliases | Versions

Test event [Info](#) Save **Test**

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

Test event action Create new event Edit saved event

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

Event sharing settings Private
This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)
 Shareable
This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

Template - *optional*
s3-put

Event JSON Format JSON

```

1 * []
2 *   "Records": [
3 *     {
4 *       "eventVersion": "2.0",
5 *       "eventSource": "aws:s3",
6 *       "awsRegion": "us-east-1",
7 *       "eventTime": "1970-01-01T00:00:00.000Z",
8 *       "eventName": "ObjectCreated:Put",
9 *       "userIdentity": {
10 *         "principalId": "EXAMPLE"
11 *       },
12 *       "requestParameters": {
13 *         "sourceIPAddress": "127.0.0.1"
14 *       }
}

```

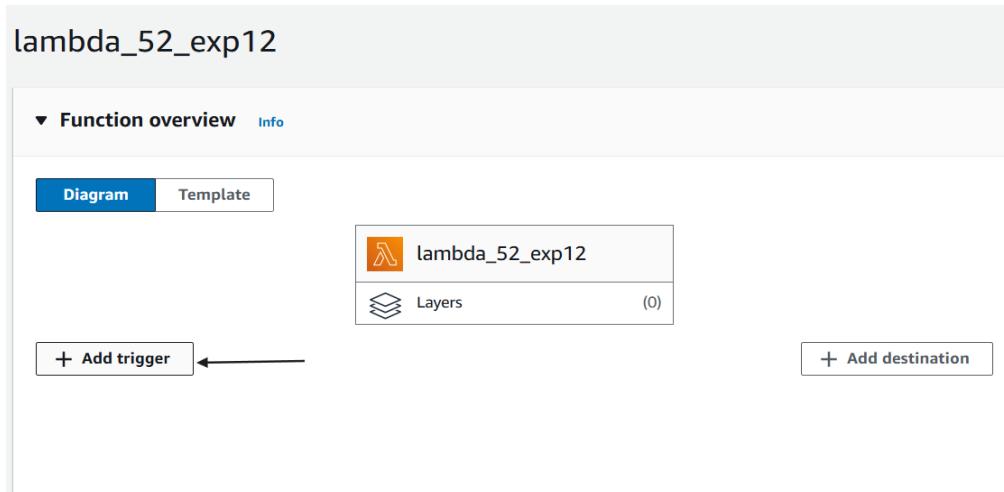
10. Now go to the Code section. Then click on the Test dropdown icon and select the event which

we have created now('test_newevent').

The screenshot shows the AWS Lambda code editor interface. At the top, there are tabs for Code, Test, Monitor, Configuration, Aliases, and Versions. The Code tab is selected. Below the tabs is a toolbar with File, Edit, Find, View, Go, Tools, Window, Test (which is currently selected), and Deploy. A dropdown menu from the Test button shows options: Configure test event (Ctrl-Shift-C), Private saved events, and test_newevent (which is highlighted). On the left, there's an Environment sidebar with a search bar 'Go to Anything (Ctrl-P)'. The main area shows a file named lambda_function.py with the following code:

```
1 import json
2
3 def lambda_handler(event, context):
4     # TODO implement
5     return {
6         'statusCode': 200,
7         'body': json.dumps('Hello from Lambda!')
8     }
9
```

11. Now go into the Lambda function and then click on add tigger.



12. Now in the Trigger information. Select the source as S3. Then select the bucket which we have created now (52bucket{in my case}), keep other things default and also you can add prefix to image.

Trigger configuration [Info](#)

S3 aws asynchronous storage

Bucket
Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.
 [X](#) [C](#)

Bucket region: us-east-1

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

All object create events [X](#)

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

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

13. Thus, **trigger** is created successfully.

lambda_52_exp12

The trigger 52bucket was successfully added to function lambda_52_exp12. The function is now receiving events from the trigger.

Function overview [Info](#)

[Diagram](#) [Template](#)

lambda_52_exp12
Layers (0)

S3 [+ Add destination](#)

[+ Add trigger](#)

14. You can also check it in the configuration section.

The screenshot shows the AWS Lambda Configuration page. The top navigation bar includes tabs for Code, Test, Monitor, Configuration (which is selected), Aliases, and Versions. On the left, a sidebar menu lists General configuration, Triggers (which is selected and highlighted in blue), Permissions, Destinations, Function URL, and Environment variables. The main content area is titled 'Triggers (1) Info' and contains a search bar labeled 'Find triggers'. Below the search bar is a list with one item: 'Trigger' followed by a green bucket icon, the name 'S3: 52bucket', and the ARN 'arn:aws:s3:::52bucket'. A 'Details' button is located to the right of the trigger entry.

15. Now write a code which logs a message “Log entry/image created successfully” when triggered.

The screenshot shows the AWS Lambda function editor. The top navigation bar includes File, Edit, Find, View, Go, Tools, Window, Test (which is selected and highlighted in blue), Deploy, and a status message 'Changes not deployed'. The main area displays the code for the 'lambda_function' file, which is part of the 'lambda_52_exp12' project. The code is as follows:

```

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

```

Here changes are not deployed.

16. So now, Save the file by ctrl+s and then click on deploy.

The screenshot shows the AWS Lambda function editor after deployment. A green success message at the top of the screen reads 'Successfully updated the function lambda_52_exp12.' The interface is identical to the previous screenshot, showing the 'lambda_function' code in the 'lambda_52_exp12' project. The status message at the top now says 'Changes deployed'.

17. Go to S3 bucket, and there upload **any image** to the bucket.

The screenshot shows the AWS S3 'Upload' interface. At the top, the navigation path is: Amazon S3 > Buckets > 52bucket > Upload. Below this, the title 'Upload' is followed by a link 'Info'. A note says: 'Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. [Learn more](#)'.

A large dashed box contains the instruction: 'Drag and drop files and folders you want to upload here, or choose Add files or Add folder.' Below this is a table titled 'Files and folders (1 Total, 349.3 KB)'. It lists one item: 'PERFUME.png'. There are buttons for 'Remove', 'Add files' (which is highlighted with a blue border), and 'Add folder'. A search bar 'Find by name' is present. The table has columns for checkboxes, Name, and Folder. The 'Name' column shows 'PERFUME.png' and the 'Folder' column shows '-'. Navigation arrows are at the bottom of the table.

The 'Destination' section shows the destination as 's3://52bucket'. It includes a 'Destination details' section with the note: 'Bucket settings that impact new objects stored in the specified destination.' Below this are sections for 'Permissions' (with the note: 'Grant public access and access to other AWS accounts.') and 'Properties' (with the note: 'Specify storage class, encryption settings, tags, and more.'). At the bottom right are 'Cancel' and 'Upload' buttons, with 'Upload' being orange.

18. Thus the image was uploaded successfully

The screenshot shows the AWS S3 'Upload: status' page. At the top, a green banner indicates 'Upload succeeded' with a link to 'View details below.' Below this, the title 'Upload: status' is displayed. A note states: 'The information below will no longer be available after you navigate away from this page.' Under the 'Summary' section, it shows the destination 's3://52bucket' and a status of 'Succeeded' with '1 file, 349.3 KB (100.00%)'. Below this, there are tabs for 'Files and folders' (which is selected) and 'Configuration'. The 'Files and folders' section shows one item: 'PERFUME.png' (image/png, 349.3 KB, Succeeded). A search bar labeled 'Find by name' is also present.

19. Now go to lambda function. Then click on test. This will give you log about the image that we have uploaded in S3 bucket.

The screenshot shows the AWS Lambda function test results. The 'Execution result' tab is selected. It displays the 'Test Event Name' as 'test_newevent', the 'Response' as a JSON object with 'statusCode': 200 and 'body': '\"Log entry/Image created successfully!\"', and the 'Function Logs' which show the execution details including RequestID, Duration, Billed Duration, Memory Size, and Max Memory Used. The Request ID shown is 'ece2072a-bb6a-4cc3-9065-2594e7d22abc'.

(In response, It gives status 200 and also the message “Log entry/image created successfully” and also contains function Logs)

20. Now go to cloudwatch. Then go into log groups. Inside that you will get the lambda function name that we have created click on it. Here, you will get a detailed log of events.

The screenshot shows the CloudWatch Log Groups interface for the AWS Lambda function 'lambda_52_exp12'. The log stream '2024/10/10/[LATEST]a153468b1a574e0a8512ea7220531432' is selected. The log events are listed in chronological order from newest to oldest. The log entries show the Lambda function starting, processing an image upload to an S3 bucket, and returning a response. The logs include detailed information such as Request IDs, AWS Lambda ARN, and execution duration.

Timestamp	Message
2024-10-10T13:25:24.595Z	INIT_START Runtime Version: python3.12.v36 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:188d9ca2e2714ff5637bd2bbe06ceb81ec3bc408a0f277dab104c14cd814b081
2024-10-10T13:25:24.688Z	START RequestId: 5072b72b-1982-4ec5-b1c4-dde4705cd4a7 Version: \$LATEST
2024-10-10T13:25:24.699Z	An image has been added to the bucket example-bucket : test52fkey
2024-10-10T13:25:24.704Z	END RequestId: 5072b72b-1982-4ec5-b1c4-dde4705cd4a7
2024-10-10T13:25:24.704Z	REPORT RequestId: 5072b72b-1982-4ec5-b1c4-dde4705cd4a7 Duration: 1.97 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 32 MB Init Duration: 90.56 ms
2024-10-10T13:26:11.881Z	START RequestId: 80470eed-fc1-48af-ab9a-42ca8000ae4b Version: \$LATEST
2024-10-10T13:26:11.982Z	An image has been added to the bucket example-bucket : test52fkey
2024-10-10T13:26:11.984Z	END RequestId: 80470eed-fc1-48af-ab9a-42ca8000ae4b
2024-10-10T13:26:11.984Z	REPORT RequestId: 80470eed-fc1-48af-ab9a-42ca8000ae4b Duration: 1.46 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 32 MB
2024-10-10T13:26:31.220Z	START RequestId: ece2072a-bb6a-4cc3-9065-2594e7d22abc Version: \$LATEST
2024-10-10T13:26:31.221Z	An image has been added to the bucket example-bucket : test52fkey
2024-10-10T13:26:31.224Z	END RequestId: ece2072a-bb6a-4cc3-9065-2594e7d22abc
2024-10-10T13:26:31.224Z	REPORT RequestId: ece2072a-bb6a-4cc3-9065-2594e7d22abc Duration: 1.65 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 32 MB

Conclusion:

Through this project, I successfully set up a Lambda function and an S3 bucket. After configuring the settings, like adding a description and setting a 1-second timeout, I created a test event named 'test_newevent' and deployed the function without any issues. I also linked the Lambda function to the S3 bucket via a trigger and added a print statement in the code. After uploading an image to the bucket and redeploying, the function returned a status code of 200 with the expected message, and CloudWatch logs captured the entire process. This practical taught me how to effectively manage AWS Lambda settings, set up and test triggers, and monitor logs using CloudWatch. I also gained a deeper understanding of integrating S3 with Lambda to automate tasks.