

# EXPERIMENT 1A

## Step 1: Log in to AWS Management Console

Go to AWS Management Console and log in with your AWS account credentials.

## Step 2: Navigate to S3 Service

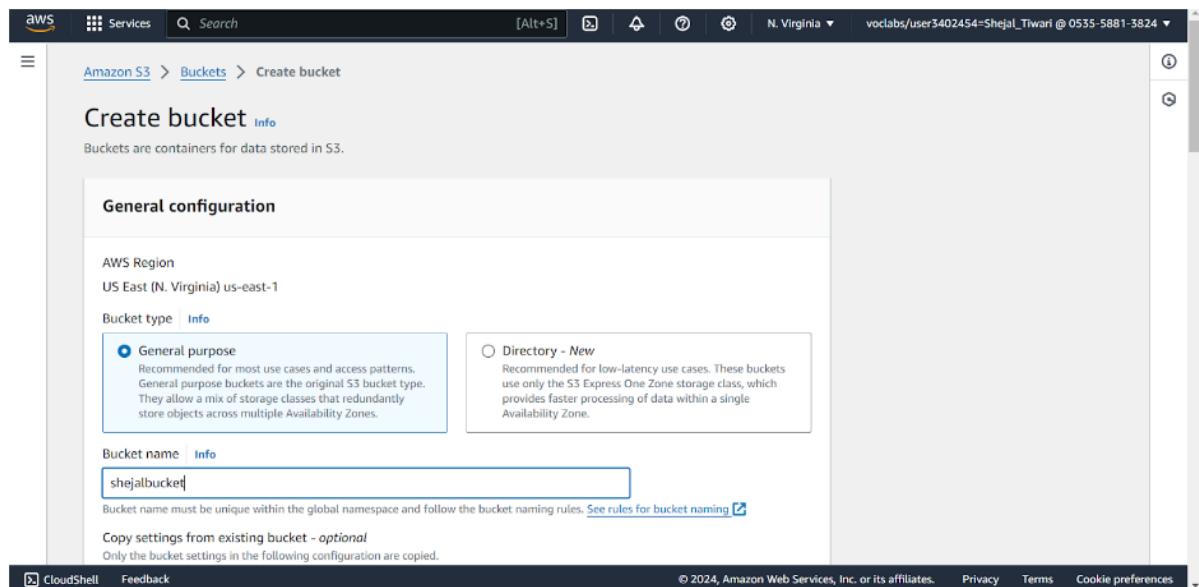
In the AWS Management Console, search for "S3" in the search bar at the top, and click on "S3" under Services.

## Step 3: Create a New Bucket

Click on the "Create bucket" button.

Fill in the details:

- Bucket Name: Enter a globally unique name for your bucket. The name must be DNS-compliant.
- AWS Region: Select the region where you want to create the bucket. Choose a region closest to you or your users for better performance.



## Step 4: Configure Bucket Options

**Object Ownership:** Decide if you want to disable or enable object ownership. By default, it's set to "Bucket owner preferred."

**Block Public Access Settings:** You can choose to block all public access to your bucket or configure specific rules. AWS recommends blocking public access unless you specifically need it to be public.

The screenshot shows the 'Permissions overview' section of the AWS S3 console. It includes a 'Block public access (bucket settings)' section where 'Block all public access' is set to 'Off'. There is also a 'Bucket policy' section with an 'Edit' button. The left sidebar lists various S3 features like Access Grants, Object Lambda Access Points, and Storage Lens.

**Bucket Versioning:** Enable versioning if you want to keep multiple versions of objects in your bucket. This is useful for backup and recovery.

**Tags:** You can add tags to your bucket to organize and track its costs.

The screenshot shows the 'Static website hosting' section of the AWS S3 console. It indicates that static website hosting is enabled and points to the bucket endpoint at <http://shejalbucket.s3-website-us-east-1.amazonaws.com>. The left sidebar shows other bucket configuration options like Object Lock and Requester pays.

## Step 5: Set Policy and Permissions

1. Set permissions based on your requirements. You can keep the default settings for private access or modify them for public or specific user access.

The screenshot shows the 'Bucket policy' section of an AWS S3 bucket configuration. It displays a JSON policy document with the following content:

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

At the top right, there are 'Edit' and 'Delete' buttons. On the far right, there is a 'Copy' button.

## Step 6: Review and Create

Review all the settings you've configured.

If everything looks good, click the "Create bucket" button.

The screenshot shows the 'Buckets' page in the AWS S3 console. A green success message at the top states: "Successfully created bucket 'shejalbucket'. To upload files and folders, or to configure additional bucket settings, choose View details." Below this, the 'Account snapshot - updated every 24 hours' is displayed. The 'General purpose buckets' tab is selected, showing a table with one item:

Name	AWS Region	IAM Access Analyzer	Creation date
shejalbucket	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	August 1, 2024, 13:30:57 (UTC+05:30)

At the bottom, there are links for CloudShell, Feedback, and various legal notices.

## Step 1: Install XAMPP

- Download XAMPP:** If you haven't already installed XAMPP, you can download it from the official website.
- Install XAMPP:** Run the installer and follow the instructions to install XAMPP on your machine.

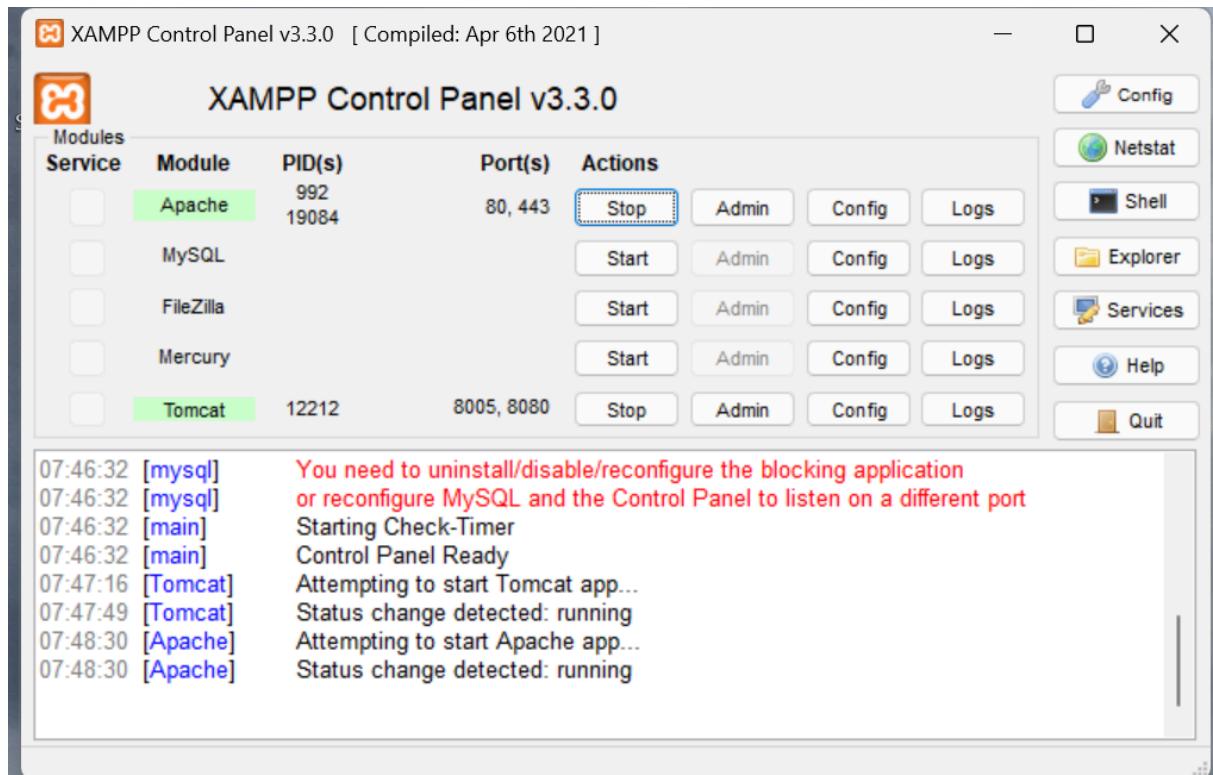
To host a PHP file on the XAMPP server, follow these steps:

## Step 1: Install XAMPP

1. **Download XAMPP:** If you haven't already installed XAMPP, you can download it from the official website.
2. **Install XAMPP:** Run the installer and follow the instructions to install XAMPP on your machine.

## Step 2: Start Apache Server

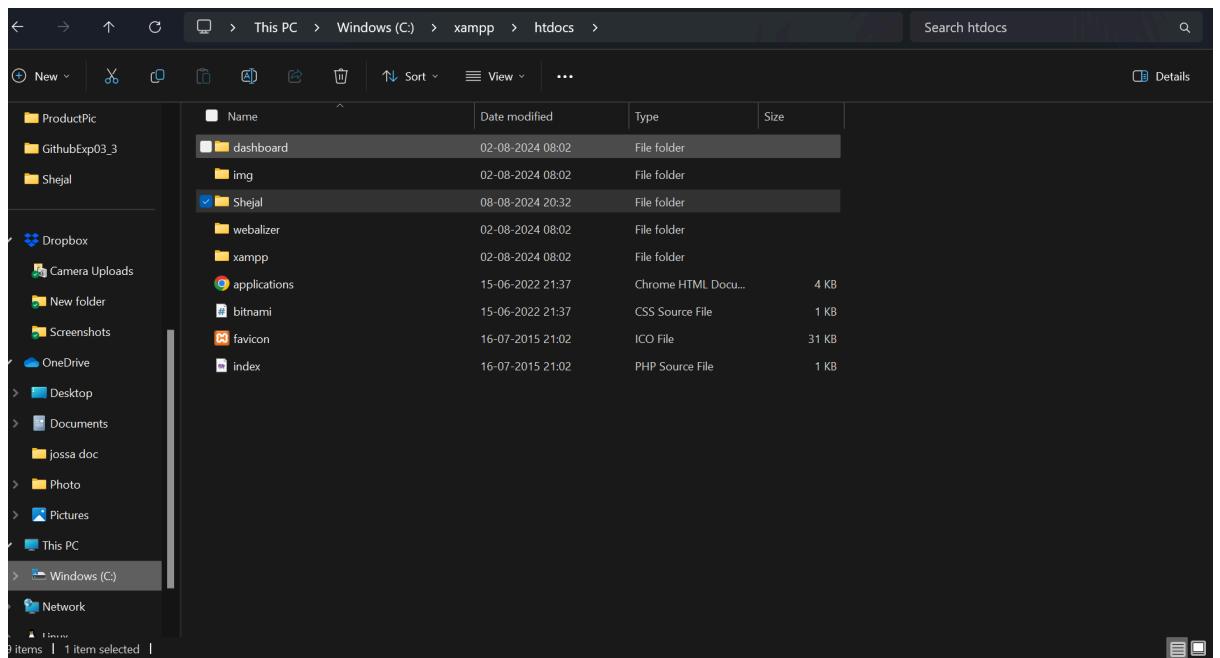
1. **Open XAMPP Control Panel:** After installation, open the XAMPP Control Panel.
2. **Start Apache:** Click the "Start" button next to Apache. The status should turn green, indicating that the Apache server is running.



3.

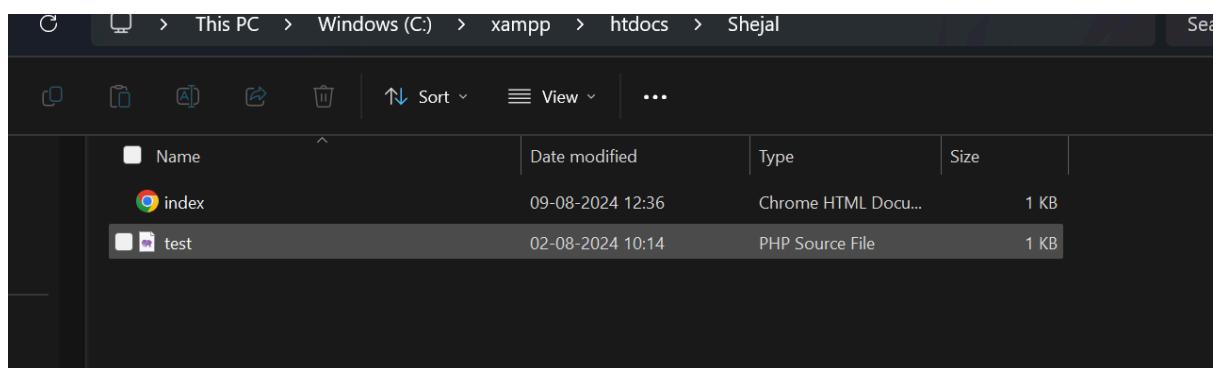
## Step 3: Place PHP Files in the htdocs Folder

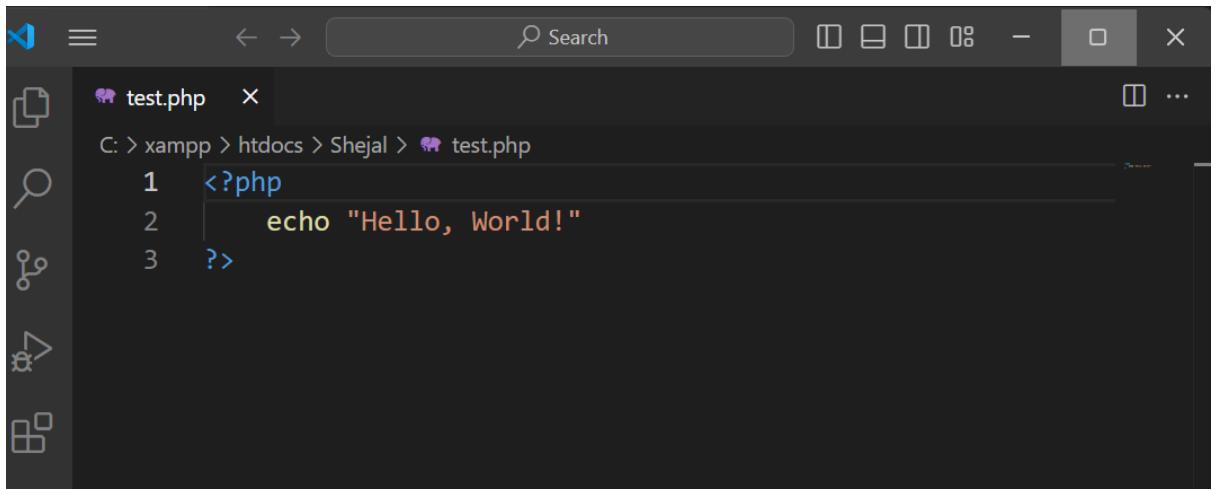
1. **Navigate to XAMPP Directory:** Go to the directory where you installed XAMPP (e.g., `C:\xampp` on Windows).
2. **Open htdocs Folder:** Inside the XAMPP folder, you'll find a folder named `htdocs`. This is the root directory where you should place your PHP files.
3. **Create a New Folder (Optional):** You can create a new folder inside `htdocs` to organize your project (e.g., `C:\xampp\htdocs\myproject`).
4. **Place PHP File:** Copy your PHP file (e.g., `index.php`) into the `htdocs` directory or your project folder.



## Step 4: Access the PHP File via Browser

- Open Web Browser:** Open any web browser.
- Access the File:** In the address bar, type the following URL:
  - If you placed the file directly in `htdocs`, use: <http://localhost/filename.php>
  - If you placed the file inside a folder, use: <http://localhost/foldername/filename.php>
  - For example, if your file is `index.php` inside `myproject` folder, the URL would be: <http://localhost/myproject/index.php>





```
C: > xampp > htdocs > Shejal > test.php
1 <?php
2 echo "Hello, World!";
3 ?>
```

## Step 5: View the Output

- The PHP file will be executed on the server, and the output will be displayed in your browser.



## Additional Tips:

- PHP Errors:** If there are errors in your PHP code, they will be displayed in the browser. You can also check the error logs in the XAMPP Control Panel under Apache > Logs > PHP Error Log.

Your PHP file is now hosted on your local XAMPP server!

## Step 7: Upload Objects (Optional)

- Once the bucket is created, you can start uploading objects to it by clicking on the bucket name and using the "Upload" button.

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with 'Amazon S3' selected under 'Buckets'. The main area shows the 'shejalbucket' with one object, 'test.php', listed. At the top right of the object list, there's an orange 'Upload' button. The bottom of the screen shows standard AWS navigation links like CloudShell and Feedback.

The screenshot shows the 'Upload' dialog box. It displays a table with one item, 'test.php', and an orange 'Upload' button at the bottom. The dialog also includes sections for 'Destination details' and 'Permissions'.

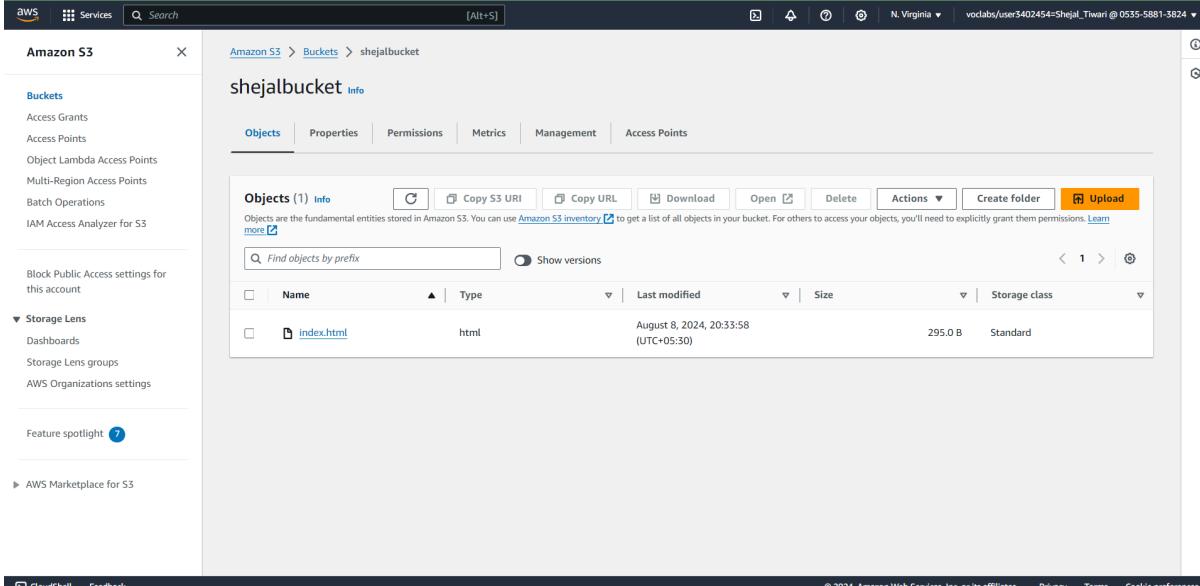
The screenshot shows the AWS S3 console. On the left, there's a sidebar with various options like Buckets, Access Grants, and Storage Lens. The main area shows a file named "test.php" in a bucket called "shejalbucket". The "Properties" tab is selected, displaying details such as Owner (awslabscOw4513773t1664266739), AWS Region (US East (N. Virginia) us-east-1), Last modified (August 2, 2024, 10:15:08 UTC+05:30), Size (35.0 B), Type (php), and Key (test.php). To the right, a "Recent download history" panel is open, showing three previous downloads: "test (2).php" (35 B, Done), "test (1).php" (35 B, 3 hours ago), and another "test.php" (35 B, 3 hours ago). A "Full download history" link is also present.

This screenshot is similar to the one above, showing the AWS S3 console for the file "test.php". However, a code editor window is overlaid on the bottom half of the screen. The code editor shows a single PHP file named "test (2).php" with the following content:

```
C: > Users > 91900 > Downloads > test (2).php
1 <?php
2 echo "Hello, World!";
3 ?>
```

The code editor interface includes tabs for File, Edit, Selection, View, Go, Run, etc., and a status bar at the bottom with various developer tools and status indicators.

In similar way it is done for index.html file



The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with various navigation links like 'Buckets', 'Access Grants', 'Access Points', etc. The main area shows a bucket named 'shejalbucket'. Under the 'Objects' tab, there is one object listed: 'index.html'. The details for this object are as follows:

Name	Type	Last modified	Size	Storage class
index.html	html	August 8, 2024, 20:33:58 (UTC+05:30)	295.0 B	Standard

Click on the URL and your website is hosted on cloud .



This is a basic HTML page.

# EXPERIMENT 1B

## Step 1: Sign in to AWS Management Console

1. Go to the [AWS Management Console](#).
2. Sign in with your AWS credentials.

## Step 2: Navigate to Cloud9

1. In the AWS Management Console, search for "Cloud9" in the search bar at the top.
2. Click on "Cloud9" under Services.

The screenshot shows the AWS Management Console interface. On the left, there is a navigation sidebar with various services like Identity and Access Management, CloudWatch Metrics, and CloudWatch Metrics Insights. The main area has a search bar at the top. Below it, a search result for "Cloud9" is displayed, showing the Cloud9 service card. The Cloud9 card includes a description: "A Cloud IDE for Writing, Running, and Debugging Code". To the right of the search result, there is a table titled "Cloud9 environments" with three entries. Each entry has columns for "Name", "MFA", "Password age", and "Console last sign-in". All three entries show "Yesterday" in all columns. At the bottom of the screenshot, the URL is https://us-east-1.console.aws.amazon.com/cloud9/home?region=us-east-1 and the page footer includes links for 2024, Privacy, Terms, and Cookie preferences.

## Step 3: Create a New Environment

1. On the Cloud9 dashboard, click on the "Create environment" button.

The screenshot shows the AWS Cloud9 developer tools interface. At the top, there's a navigation bar with the AWS logo, a 'Services' dropdown, a search bar, and user information ('N. Virginia' and 'vocabs/user3402454=Shejal\_Tiwari @ 0535-5881-3824'). Below the header, the main title 'AWS Cloud9' is displayed, followed by the subtitle 'A cloud IDE for writing, running, and debugging code'. A call-to-action button 'Create environment' is prominently shown. To the left, a section titled 'How it works' provides an overview of Cloud9's functionality. On the right, a sidebar titled 'Getting started' lists several learning resources with their respective durations: 'Before you start (2 min read)', 'Create an environment (2 min read)', 'Working with environments (15 min read)', 'Working with the IDE (10 min read)', and 'Working with AWS Lambda (5 min read)'. At the bottom of the page, there are links for 'CloudShell', 'Feedback', and legal notices ('© 2024, Amazon Web Services, Inc. or its affiliates.', 'Privacy', 'Terms', 'Cookie preferences').

## Step 4: Configure Environment Settings

### 1. Name and Description:

- **Name:** Enter a name for your environment.
- **Description:** Optionally, provide a description of the environment.

### 2. Environment Settings:

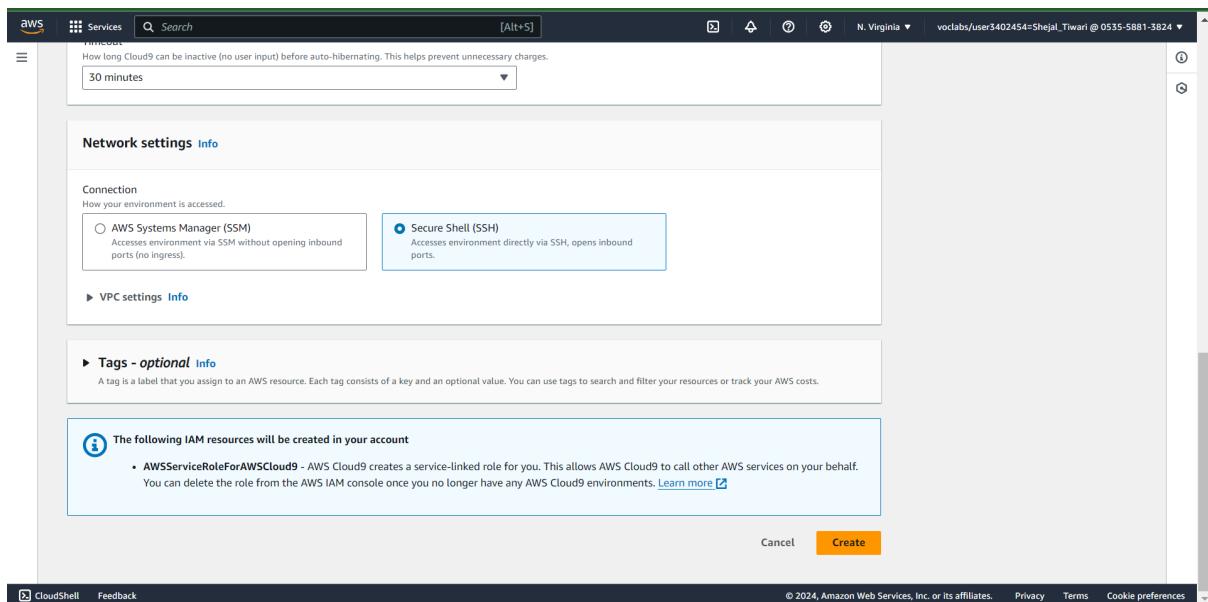
- **Environment type:** Choose one of the following:
  - **Create a new EC2 instance for environment:** AWS will automatically create an EC2 instance for you. Choose this if you want to run the environment on a new EC2 instance.
  - **Connect to an existing EC2 instance:** If you have an existing EC2 instance, you can select this option to connect Cloud9 to it.
  - **Connect to your own server via SSH:** Use this if you want to connect Cloud9 to an on-premises server or another cloud provider's server.
- **Instance type:** Select the instance type (e.g., `t2.micro` for free tier).
- **Platform:** Choose the platform that suits your development needs (e.g., Amazon Linux, Ubuntu, etc.).
- **Cost-saving setting:** Set an automatic timeout to stop the EC2 instance when it's not in use, saving costs.

The screenshot shows the 'Create environment' page in the AWS Cloud9 interface. In the 'Details' section, the 'Name' field is filled with 'ShejalTiwari'. Below it, there's an optional 'Description' field and a section for 'Environment type'. Under 'New EC2 instance', the 't2.micro (1 GiB RAM + 1 vCPU)' option is selected. Other options like 't3.small (2 GiB RAM + 2 vCPU)' and 'm5.large (8 GiB RAM + 2 vCPU)' are also shown. The page includes standard AWS navigation and footer links.

### 3. Network Settings:

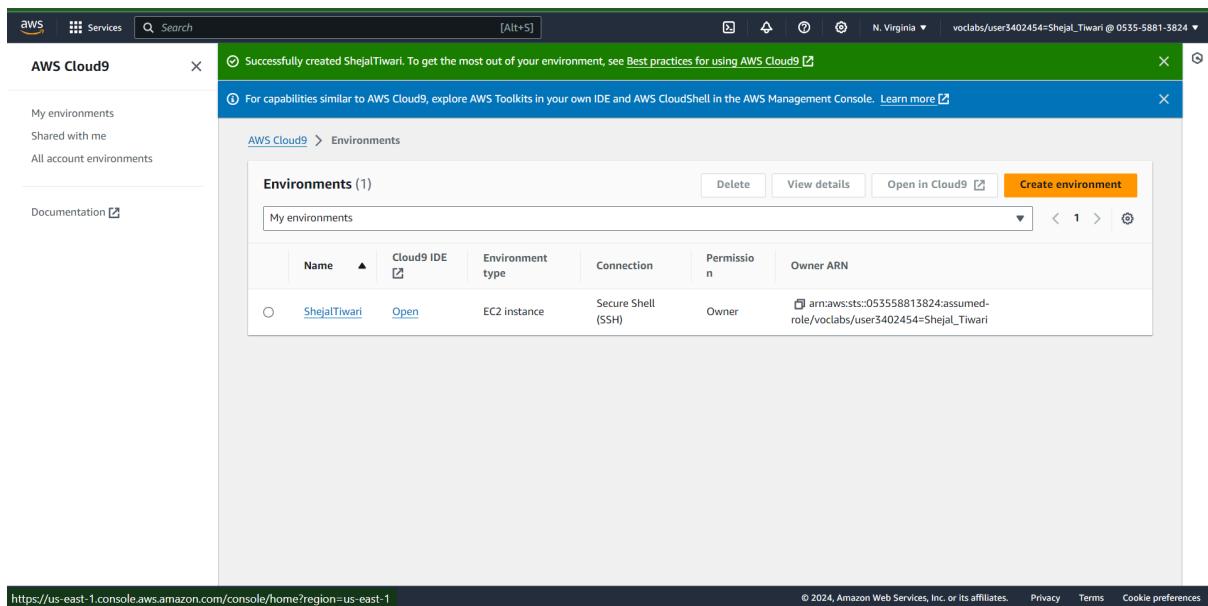
- If you selected "Create a new EC2 instance for environment," you'll need to choose a VPC (Virtual Private Cloud) and a subnet. Usually, the default options will work, but you can customize them if needed.

This screenshot shows the 'New EC2 instance' configuration page. It allows selecting an instance type: 't2.micro (1 GiB RAM + 1 vCPU)', 't3.small (2 GiB RAM + 2 vCPU)', or 'm5.large (8 GiB RAM + 2 vCPU)'. The 't2.micro' option is currently selected. Below this, there's a link to 'Additional instance types' and a 'Platform info' section where 'Amazon Linux 2023' is chosen. A 'Timeout' section specifies a 30-minute auto-hibernation period. The page also includes a 'Network settings' section and standard AWS footer links.



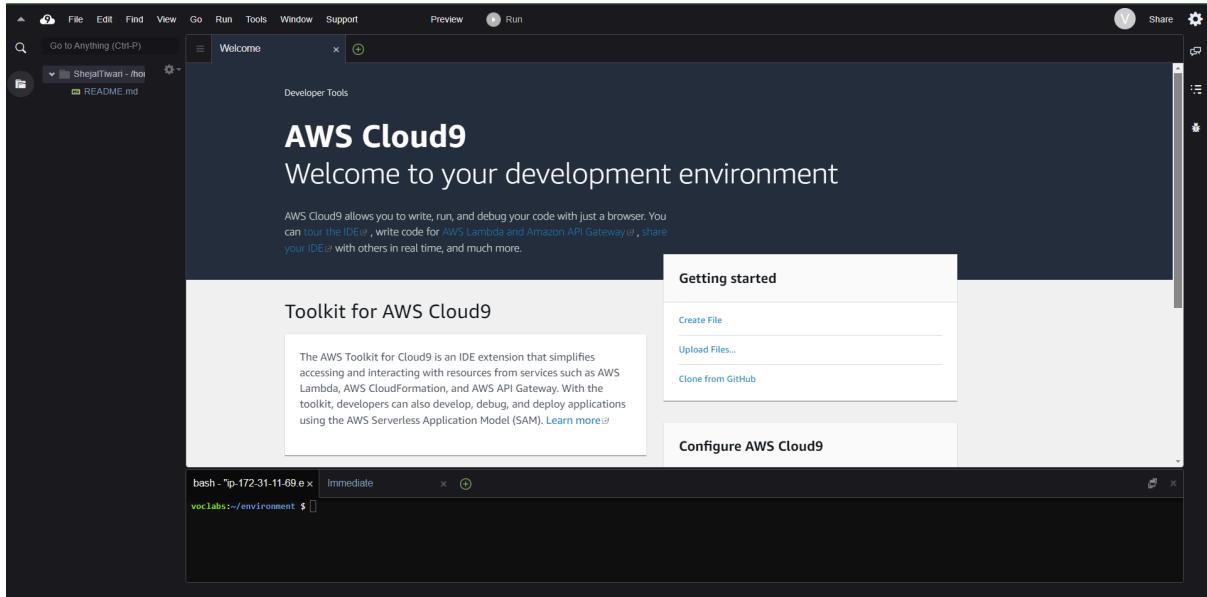
## Step 5: Review and Create

1. Review all the settings you configured.
2. If everything looks good, click on the "Create environment" button.



## Step 6: Access Your Cloud9 Environment

1. After a few moments, your Cloud9 environment will be ready, and you'll be automatically redirected to the Cloud9 IDE.
2. You can start coding immediately, using the terminal to install packages, run scripts, or manage files as you would on a local machine.



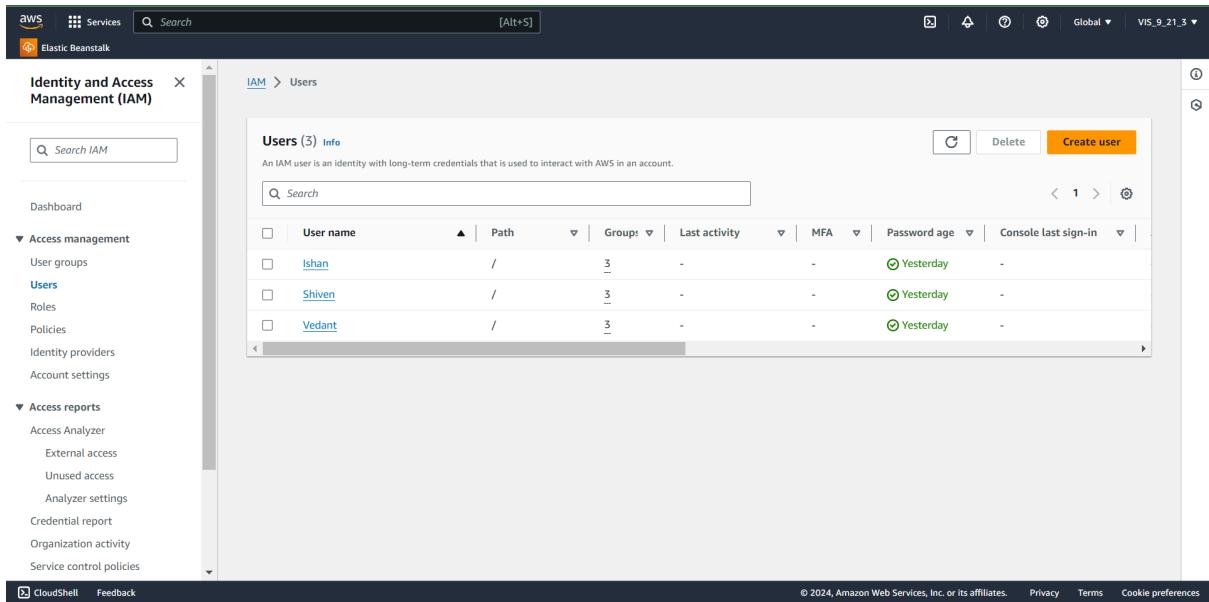
## Additional Configuration (Optional)

- **Install additional tools:** You can install additional tools and software using the terminal within Cloud9, just as you would on a regular Linux server.
- **Clone repositories:** Use Git within Cloud9 to clone repositories, manage version control, and collaborate on code.

## 2. Creating an IAM User in AWS

To create a new IAM user in AWS:

1. **Sign in to AWS Management Console:**
  - Go to the [AWS Management Console](#).
2. **Navigate to IAM:**
  - Search for "IAM" in the services menu and select it.
3. **Create a New User:**
  - Click on "Users" in the left-hand menu.
  - Click on "Add user."



The screenshot shows the AWS IAM service interface. The left sidebar has a tree view with 'Identity and Access Management (IAM)' selected. Under 'Access management', 'Users' is also selected. The main content area is titled 'Users (3) Info' and contains a table with three rows. The columns are: User name, Path, Group, Last activity, MFA, Password age, and Console last sign-in. The users listed are Ishan, Shiven, and Vedant, all of whom have 'Yesterday' listed under 'Last activity' and 'Console last sign-in'. There are buttons for 'Create user' and 'Delete' at the top right of the table. The bottom right corner of the page footer includes links for CloudShell, Feedback, Privacy, Terms, and Cookie preferences.

User name	Path	Group	Last activity	MFA	Password age	Console last sign-in
Ishan	/	3	-	-	Yesterday	-
Shiven	/	3	-	-	Yesterday	-
Vedant	/	3	-	-	Yesterday	-

### 4. Set User Details:

- **User Name:** Enter a name for the new user.
- **Access Type:**
  - **Programmatic access:** If the user needs access to AWS CLI, SDK, or API.
  - **AWS Management Console access:** If the user needs access to the AWS Management Console.

**Specify user details**

User details

User name: ShejtiTiwari123

Provide user access to the AWS Management Console - optional

**Are you providing console access to a person?**

User type:

- Specify a user in Identity Center - Recommended
- I want to create an IAM user

Console password

Autogenerated password

Custom password

Enter a custom password for the user: \*\*\*\*\*

Show password

Users must create a new password at next sign-in - Recommended

## 5. Set Permissions:

- You can attach existing policies directly, add the user to a group with predefined permissions, or copy permissions from another user.

**Set permissions**

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

**Permissions options**

- Add user to group
- Copy permissions
- Attach policies directly

**User groups (3)**

Group name	Users	Attached policies	Created
AdvanceDevOps_21_3_9	3	AWSCloud9EnvironmentMember	2024-08-07 (Yesterday)
AdvanceDevOps_3_21_9	3	AWSCloud9EnvironmentMember	2024-08-07 (Yesterday)
AdvDevOpsLab_9	3	AWSCloud9EnvironmentMember	2024-08-07 (Yesterday)

**Set permissions boundary - optional**

## 6. Tags (Optional):

- Add tags to the user account for easy identification or organization.

## 7. Review and Create:

- Review the settings and click "Create user."

The screenshot shows the AWS IAM 'Users' page. A success message at the top says 'User created successfully'. The main table lists four users:

User name	Path	Group	Last activity	MFA	Password age	Console last sign-in	Access key ID	Active key age
Ishan	/	3	-	-	Yesterday	-	-	-
ShejalTiwari123	/	0	-	-	-	-	-	-
Shiven	/	3	-	-	Yesterday	-	-	-
Vedant	/	3	-	-	Yesterday	-	-	-

## 8. Download Credentials:

- On the confirmation page, download the access key and secret key if you've enabled programmatic access. Keep them secure.

The screenshot shows the 'Create user' wizard, Step 4: Retrieve password. It displays the following information:

- Console sign-in details:**
  - Console sign-in URL: <https://O22499016110.siginin.aws.amazon.com/console>
  - User name: ShejalTiwari
  - Console password: (redacted)
- Buttons:** Cancel, Download .csv file, Return to users list.

## Creating a User Group in AWS IAM

To create a new user group in AWS IAM:

- Sign in to AWS Management Console:**
  - Go to the [AWS Management Console](#).
- Navigate to IAM:**
  - Search for "IAM" in the services menu and select it.
- Create a New Group:**
  - Click on "User groups" in the left-hand menu.
  - Click on "Create group."

The screenshot shows the 'Create user group' interface in the AWS IAM console. In the 'Name the group' section, the user has entered 'Shejalgroup'. Below it, under 'Add users to the group - Optional (4)', there is a table listing four IAM users: Ishan, ShejalTiwari123, Shiven, and Yedant. Each user row includes a checkbox, a 'User name' column, a 'Groups' column (all empty), a 'Last activity' column (all marked as 'Yesterday'), and a 'Creation time' column. At the bottom of the page, there is a large list titled 'Attach permissions policies - Optional (945)' with a note: 'You can attach up to 10 policies to this user group. All the users in this group will have permissions that are defined in the selected policies.' A search bar and a filter dropdown for 'All types' are also present.

#### 4. Set Group Name:

- Enter a name for your group in the "Group name" field.

This screenshot is identical to the one above, showing the 'Create user group' interface in the AWS IAM console. The user group 'Shejalgroup' has been named, and the same four users (Ishan, ShejalTiwari123, Shiven, Yedant) are listed in the 'Add users to the group' section. The 'Attach permissions policies' section is also visible at the bottom.

#### 5. Attach Permissions Policies:

- Select the policies you want to attach to the group. These policies define what actions the users in this group can perform.
- You can attach existing AWS managed policies, or create and attach a custom policy.

## Create user group

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

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

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

### Permissions policies (947)

Filter by Type				
<input type="text" value="Search"/>	All ty... ▾	< 1 2 3 4 5 6 7 ... 48 >		<a href="#">Create policy</a>
<input type="checkbox"/>	Policy name ▾	Type	Use...	Description
<input type="checkbox"/>	<a href="#">AdministratorAccess</a>	AWS managed ...	None	Provides full access to AWS services
<input type="checkbox"/>	<a href="#">AdministratorAcce...</a>	AWS managed	None	Grants account administrative permission
<input type="checkbox"/>	<a href="#">AdministratorAcce...</a>	AWS managed	None	Grants account administrative permission
<input type="checkbox"/>	<a href="#">AlexaForBusinessD...</a>	AWS managed	None	Provide device setup access to Alexa
<input type="checkbox"/>	<a href="#">AlexaForBusinessF...</a>	AWS managed	None	Grants full access to AlexaForBusiness
<input type="checkbox"/>	<a href="#">AlexaForBusinessG...</a>	AWS managed	None	Provide gateway execution access to Alexa
<input type="checkbox"/>	<a href="#">AlexaForBusinessLi...</a>	AWS managed	None	Provide access to Lifesize AVS devices
<input type="checkbox"/>	<a href="#">AlexaForBusinessP...</a>	AWS managed	None	Provide access to Poly AVS devices
<input type="checkbox"/>	<a href="#">AlexaForBusinessR...</a>	AWS managed	None	Provide read only access to AlexaForBusiness
<input type="checkbox"/>	<a href="#">AmazonAPIGatewa...</a>	AWS managed	None	Provides full access to create/edit/delete
<input type="checkbox"/>	<a href="#">AmazonAPIGatewa...</a>	AWS managed	None	Provides full access to invoke APIs in

[Cancel](#) [Create user group](#)

Shejalgrp user group created.

Step 1 [Specify user details](#)

Step 2 [Set permissions](#)

Step 3 [Review and create](#)

Step 4 [Retrieve password](#)

### Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

#### Permissions options

- 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 (4)

User groups (4)				
<input type="checkbox"/>	Group name ▾	Users	Attached policies ▾	Created
<input type="checkbox"/>	<a href="#">AdvanceDevOps_21_3_9</a>	3	<a href="#">AWSCloud9EnvironmentMember</a>	2024-08-07 (Yesterday)
<input type="checkbox"/>	<a href="#">AdvanceDevOps_3_21_9</a>	3	<a href="#">AWSCloud9EnvironmentMember</a>	2024-08-07 (Yesterday)
<input type="checkbox"/>	<a href="#">AdvDevOpsLab_9</a>	3	<a href="#">AWSCloud9EnvironmentMember</a>	2024-08-07 (Yesterday)
<input type="checkbox"/>	<a href="#">Shejalgrp</a>	0	-	2024-08-08 (Now)

▶ Set permissions boundary - optional

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

## 6. Review and Create:

- Review your settings and click on "Create group."

Review and create

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

**User details**

User name ShejalTiwari	Console password type Custom password	Require password reset No
---------------------------	--	------------------------------

**Permissions summary**

Name	Type	Used as
Shejalgrp	Group	Permissions group

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

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

Cancel Previous Create user

## 7. Add Users to the Group (Optional):

- After creating the group, you can add users to it by selecting the group, then going to the "Users" tab and clicking on "Add users to group."

Policies attached to this user group.

IAM > User groups > Shejalgrp

**Shejalgrp** Info

**Summary**

User group name Shejalgrp	Creation time August 08, 2024, 13:47 (UTC+0:30)	ARN arn:aws:iam:022499016110:group/Shejalgrp
------------------------------	--	---

Users (1) Permissions Access Advisor

**Permissions policies (1)** Info

You can attach up to 10 managed policies.

Policy name	Type	Attached entities
AWSCloud9EnvironmentMember	AWS managed	4

Filter by Type All types

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



## EXPERIMENT NO: 2

**Aim:** To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy. Your continuous deployment pipeline will need a target environment containing virtual servers, or Amazon EC2 instances, where it will deploy sample code. You will prepare this environment before creating the pipeline.

1. Go to the [AWS Management Console](#) and navigate to Elastic Beanstalk.

Click on **Create Application**.

Add name to your application

The screenshot shows the 'Configure environment' step of the AWS Elastic Beanstalk 'Create Application' wizard. The interface includes a navigation bar with the AWS logo, services menu, search bar, and account information (N. Virginia). The main area is titled 'Step 1 of 6' and 'Configure environment'. It contains three sections: 'Environment tier', 'Application information', and 'Environment information'. In the 'Environment tier' section, 'Web server environment' is selected. In the 'Application information' section, the 'Application name' field is filled with 'Shejal'. The 'Environment information' section shows a placeholder 'Environment name'.

The screenshot shows the 'Application Information' configuration page for an AWS Elastic Beanstalk application named 'Shejal'. The 'Application name' field contains 'Shejal'. The 'Environment name' field contains 'Shejal-env'. The 'Domain' field has the value '.us-east-1.elasticbeanstalk.com'. The 'Check availability' button is visible. The 'Environment description' field is empty.

**Application information**

Application name: Shejal

Maximum length of 100 characters.

► Application tags (optional)

**Environment information**

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

Environment name: Shejal-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.

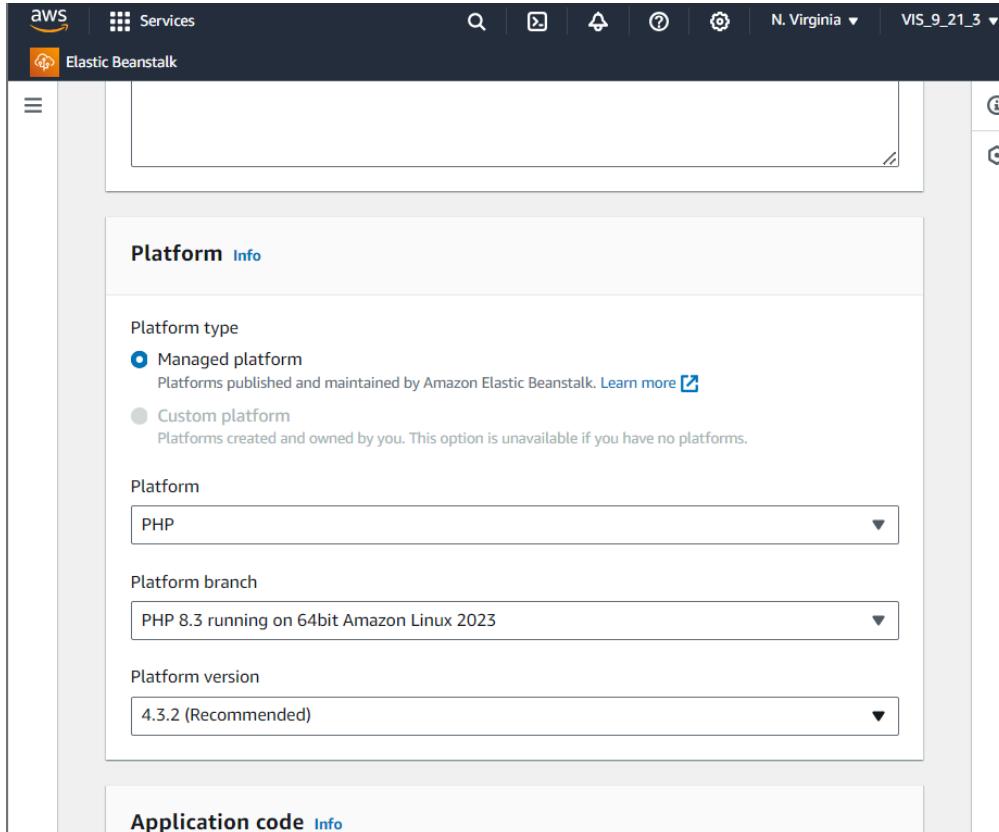
Domain: Leave blank for autogenerated value .us-east-1.elasticbeanstalk.com

Check availability

Environment description:

## 2. Configure PHP Environment:

- Choose PHP as the platform for your Elastic Beanstalk environment, ensuring compatibility with your PHP application.



3. In configure service access select “use an existing service role”

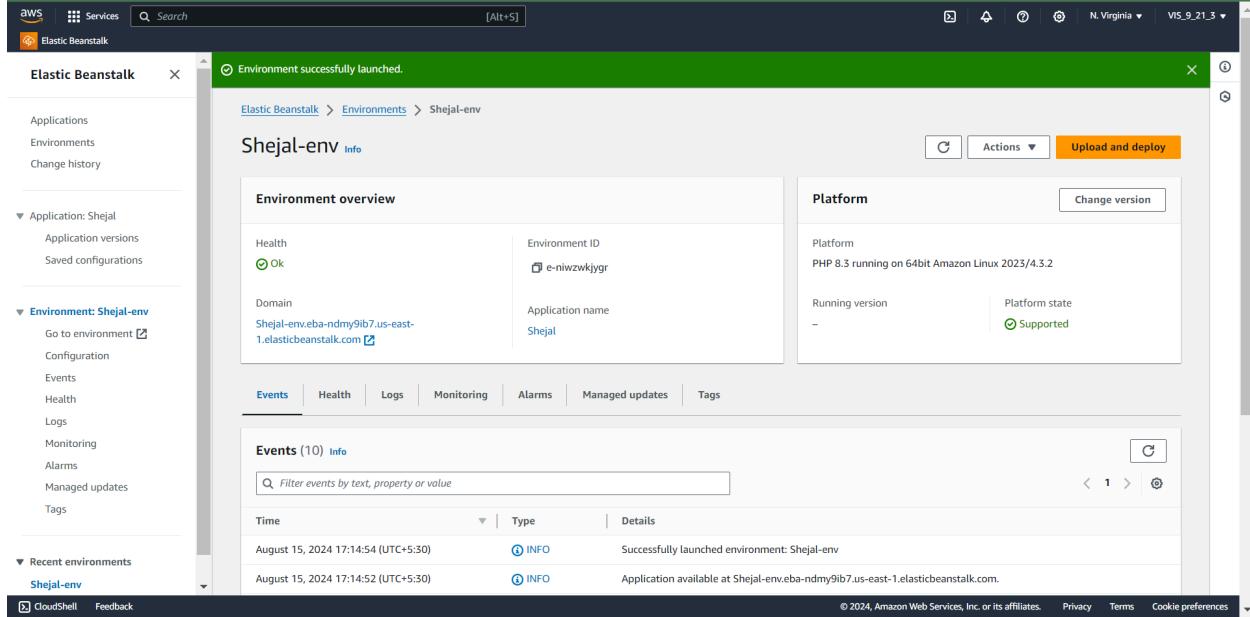
Also you need to create key and instance profile first before adding it in the below dropdown.

The screenshot shows the 'Configure service access' step of the AWS Elastic Beanstalk setup wizard. On the left, a sidebar lists steps: Step 1 (Configure environment), Step 2 (Configure service access, currently selected), Step 3 (optional: Set up networking, database, and tags), Step 4 (optional: Configure instance traffic and scaling), Step 5 (optional: Configure updates, monitoring, and logging), and Step 6 (Review). The main panel is titled 'Service access' and contains the following configuration:

- Service role:** A radio button is selected for "Use an existing service role". A dropdown menu shows "aws-elasticbeanstalk-service-role" with a clear button next to it.
- EC2 key pair:** A dropdown menu shows "shejalkey" with a clear button next to it.
- EC2 instance profile:** A dropdown menu shows "landingpagerole" with a clear button next to it.

At the bottom of the panel are buttons for "Cancel", "Skip to review", "Previous", and "Next" (which is highlighted in orange).

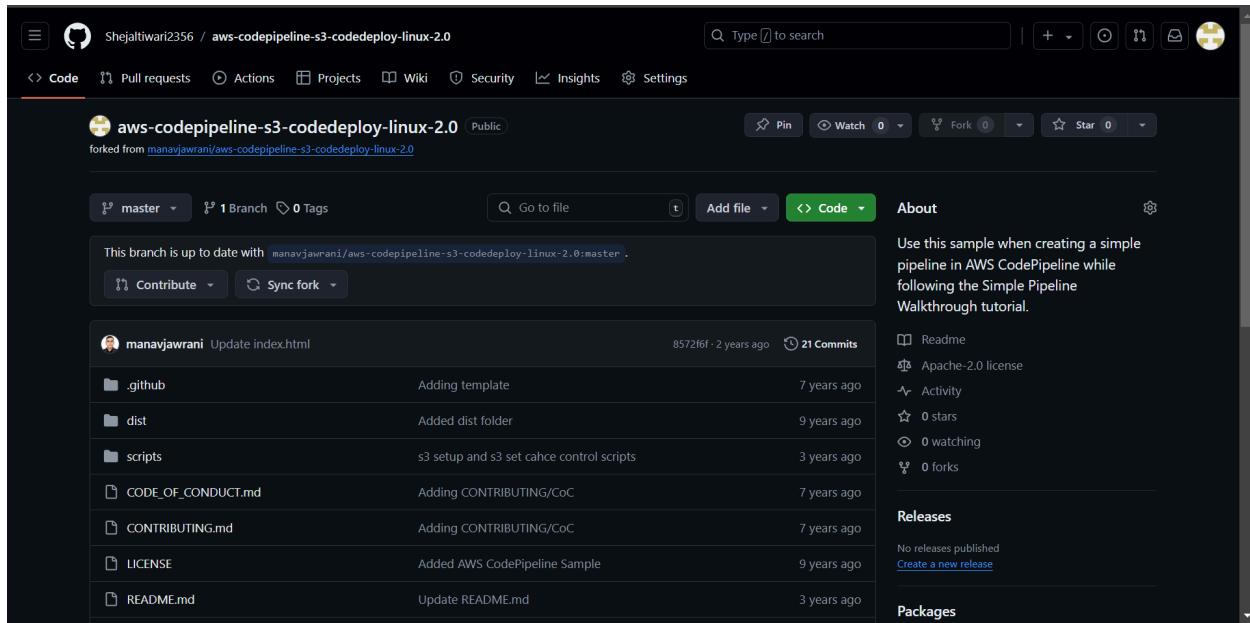
4. Directly select next to all the optional settings.  
 Your environment will be created successfully .



The screenshot shows the AWS Elastic Beanstalk console with the following details:

- Environment Overview:**
  - Health: Ok
  - Environment ID: e-niwzkwjygr
  - Domain: Shejal-env.eba-ndmy9ib7.us-east-1.elasticbeanstalk.com
  - Application name: Shejal
- Platform:**
  - Platform: PHP 8.3 running on 64bit Amazon Linux 2023/4.3.2
  - Running version: -
  - Platform state: Supported
- Events:** Shows two INFO-level events from August 15, 2024, detailing the successful launch and the application being available at the specified URL.

5. You need to fork this repository.



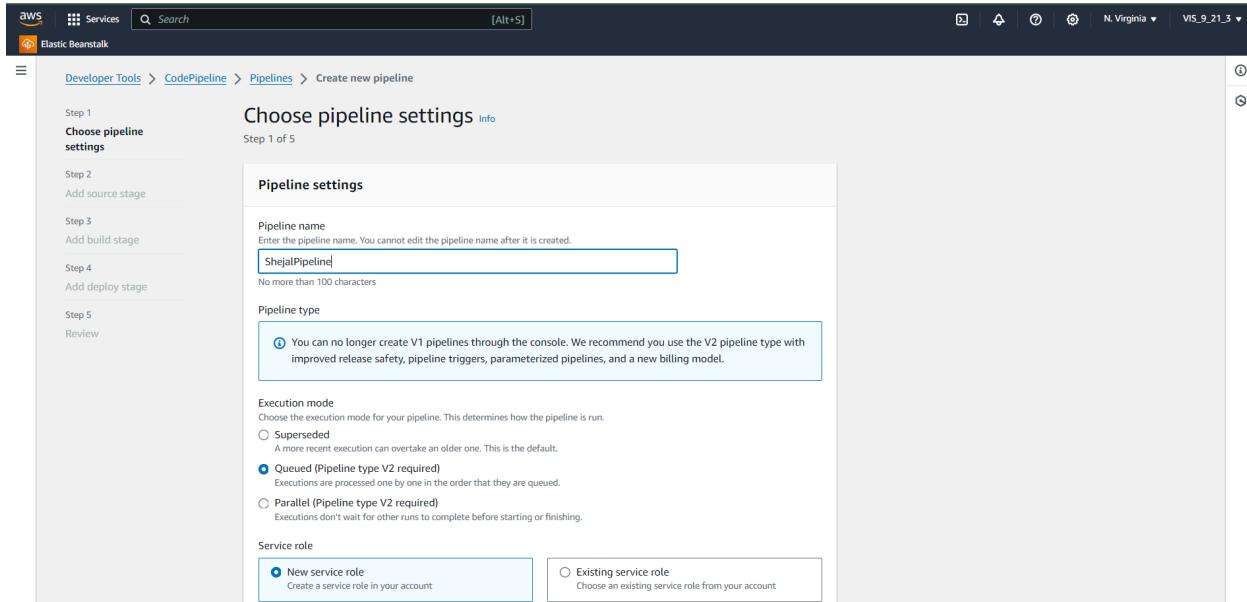
The screenshot shows a GitHub repository page for "aws-codepipeline-s3-codedeploy-linux-2.0".

- Repository Summary:**
  - Owner: Shejaltiwari2356
  - Name: aws-codepipeline-s3-codedeploy-linux-2.0
  - Description: forked from manavjawani/aws-codepipeline-s3-codedeploy-linux-2.0
  - Code tab is selected.
  - Branch: master
  - 1 Branch
  - 0 Tags
  - 21 Commits
  - 2 years ago
- Code View:**
  - This branch is up to date with manavjawani/aws-codepipeline-s3-codedeploy-linux-2.0:master.
  - Contribute and Sync fork buttons.
  - File list:
    - .github: Adding template (7 years ago)
    - dist: Added dist folder (9 years ago)
    - scripts: s3 setup and s3 set cache control scripts (3 years ago)
    - CODE\_OF\_CONDUCT.md: Adding CONTRIBUTING/CoC (7 years ago)
    - CONTRIBUTING.md: Adding CONTRIBUTING/CoC (7 years ago)
    - LICENSE: Added AWS CodePipeline Sample (9 years ago)
    - README.md: Update README.md (3 years ago)
- Right Sidebar:**
  - About: Use this sample when creating a simple pipeline in AWS CodePipeline while following the Simple Pipeline Walkthrough tutorial.
  - Readme
  - Apache-2.0 license
  - Activity
  - 0 stars
  - 0 watching
  - 0 forks
  - Releases: No releases published. Create a new release
  - Packages

## 6. Open CodePipeline:

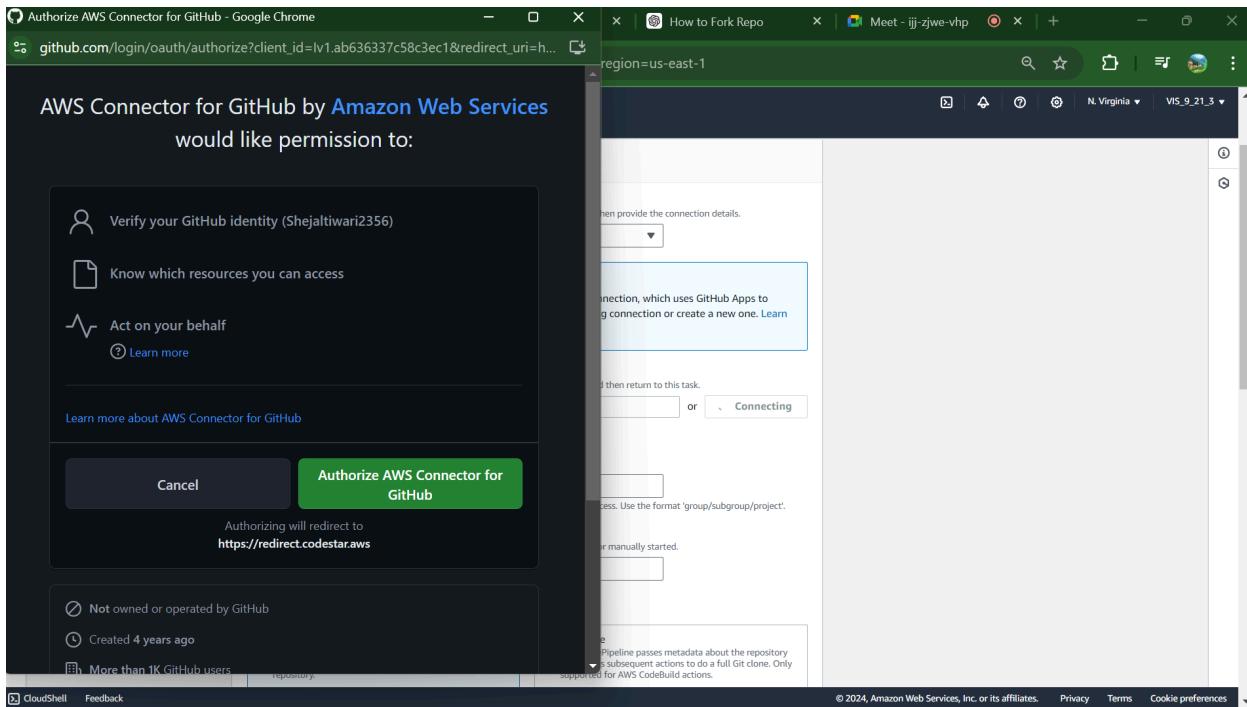
- Navigate to the AWS Management Console and open CodePipeline.
- Click "Create pipeline."

Create a codepipeline add name for your pipeline.



## 7. After selecting the github version

We need to connect it to the repository for that we will add git connect



## 9. Create a connection in your GitHub give connection name.

The screenshot shows the AWS Elastic Beanstalk Developer Tools interface. The user is in the 'Connections' section, specifically creating a new connection. The 'GitHub connection settings' form is displayed, with the 'Connection name' field set to 'ShejalConnect'. A note at the top of the form informs the user that starting in July 2024, connections will be managed via 'codeconnections' in the resource ARN. A 'Connect' button is located at the bottom right of the form.

## 10. select version 2 for your GitHub.

The screenshot shows the AWS CodePipeline 'Add source stage' step. The 'Source' provider dropdown is set to 'GitHub (Version 2)'. A note below the dropdown explains the new GitHub version 2 action, stating: '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.' A green box at the bottom left indicates 'Ready to connect' with a checkmark. The 'Repository name' and 'Default branch' fields are also visible on the right side of the screen.

11. Select branch for the repository here we have selected master branch.

This screenshot shows the configuration of a GitHub action within the AWS CodePipeline interface. The 'Connection' section lists an existing connection named 'Shejaltiwari2356/aws-codepipeline-s3-codedeploy-linux-2.0'. The 'Default branch' is set to 'master'. The 'Output artifact format' is set to 'CodePipeline default'.

12. You can skip build stage.

This screenshot shows the 'Add build stage' step in the AWS CodePipeline pipeline creation process. The 'Build provider' dropdown is empty, indicating no build provider has been selected. The 'Skip build stage' button is highlighted in orange, suggesting it is the next action to take.

## 13. To Deploy select on next.

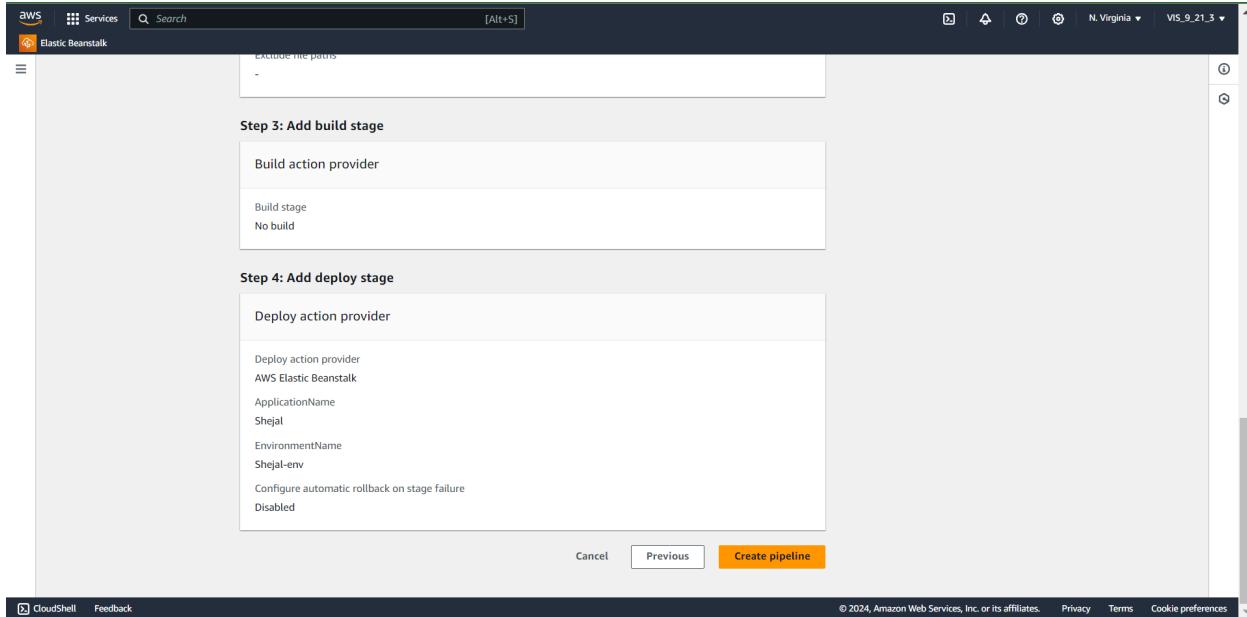
The screenshot shows the 'Deploy' step in the AWS Elastic Beanstalk console. The 'Deploy provider' dropdown is set to 'AWS Elastic Beanstalk'. The 'Region' dropdown is set to 'US East (N. Virginia)'. The 'Input artifacts' dropdown is empty. The 'Application name' field contains 'Shejal'. The 'Environment name' field contains 'Shejal-env'. A checkbox for 'Configure automatic rollback on stage failure' is unchecked. At the bottom, there are 'Cancel', 'Previous', and 'Next' buttons, with 'Next' being highlighted in orange.

## 14. Review all the settings created.

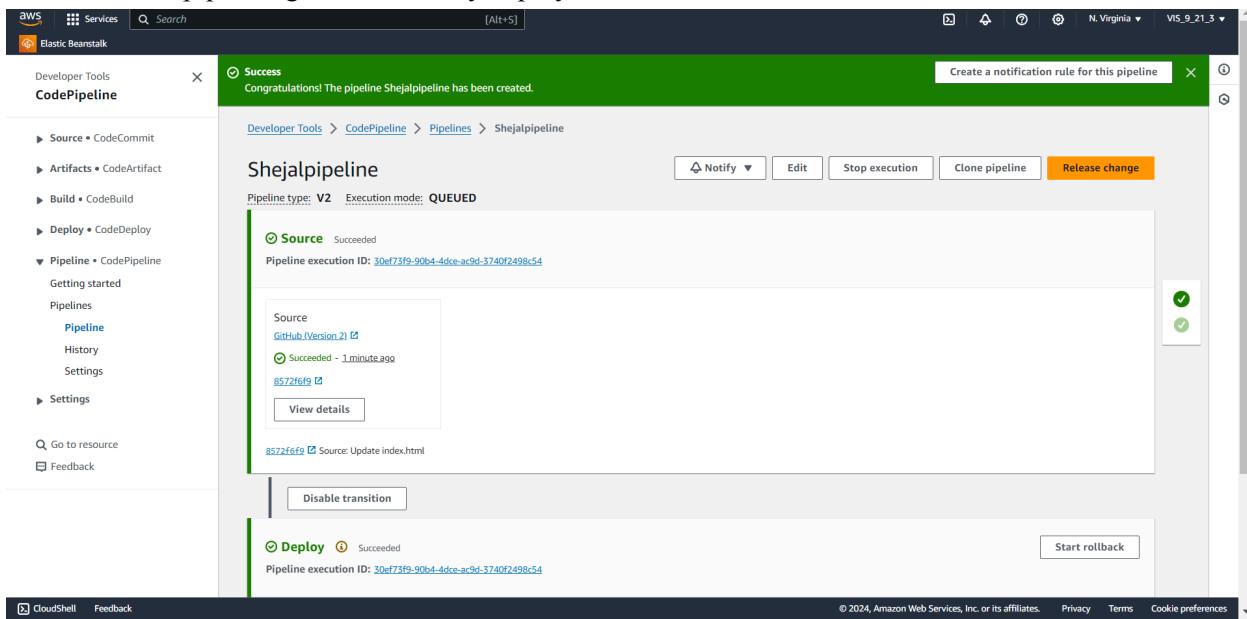
The screenshot shows the 'Step 1: Choose pipeline settings' step in the AWS CodePipeline console. The pipeline name is 'Shejalpipeline', the type is 'V2', and the execution mode is 'QUEUED'. The artifact location is 'codepipeline-us-east-1-819735269122'. The service role name is 'AWSCodePipelineServiceRole-us-east-1-Shejalpipeline'. Below this, there is a 'Variables' section which is currently empty. At the bottom, there are 'CloudShell', 'Feedback', and 'Next' buttons, with 'Next' being highlighted in orange.

Name: Shejal Nilesh Tiwari

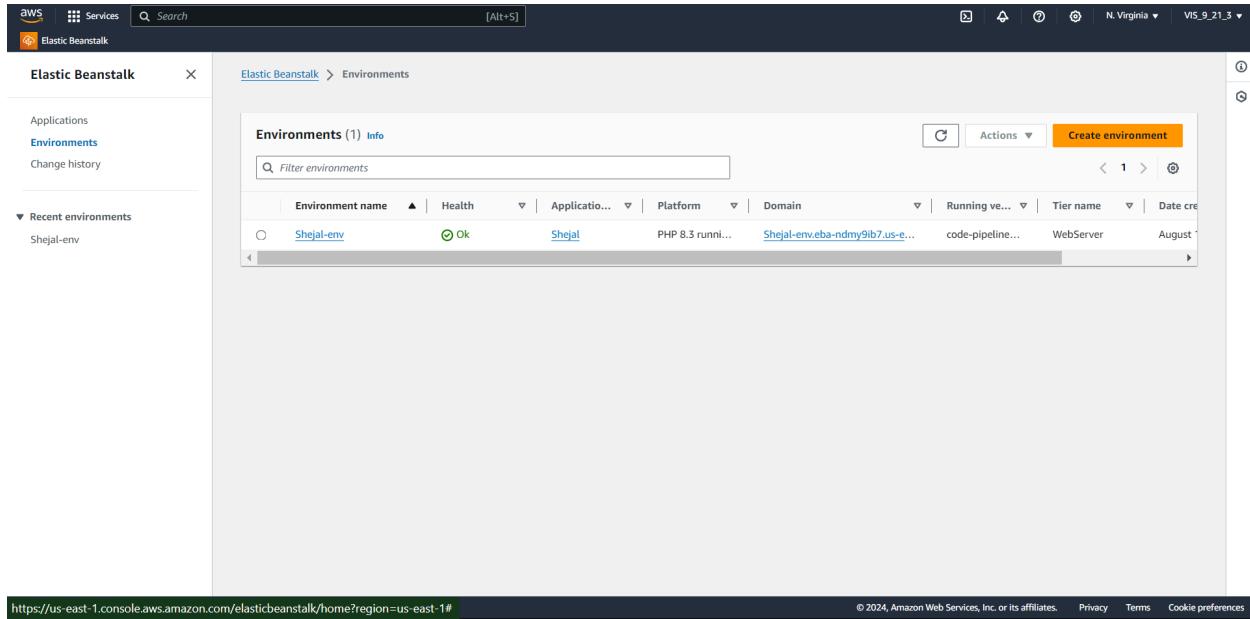
Div/Roll No: D15C/57



15. Wait till the pipeline gets successfully deployed.

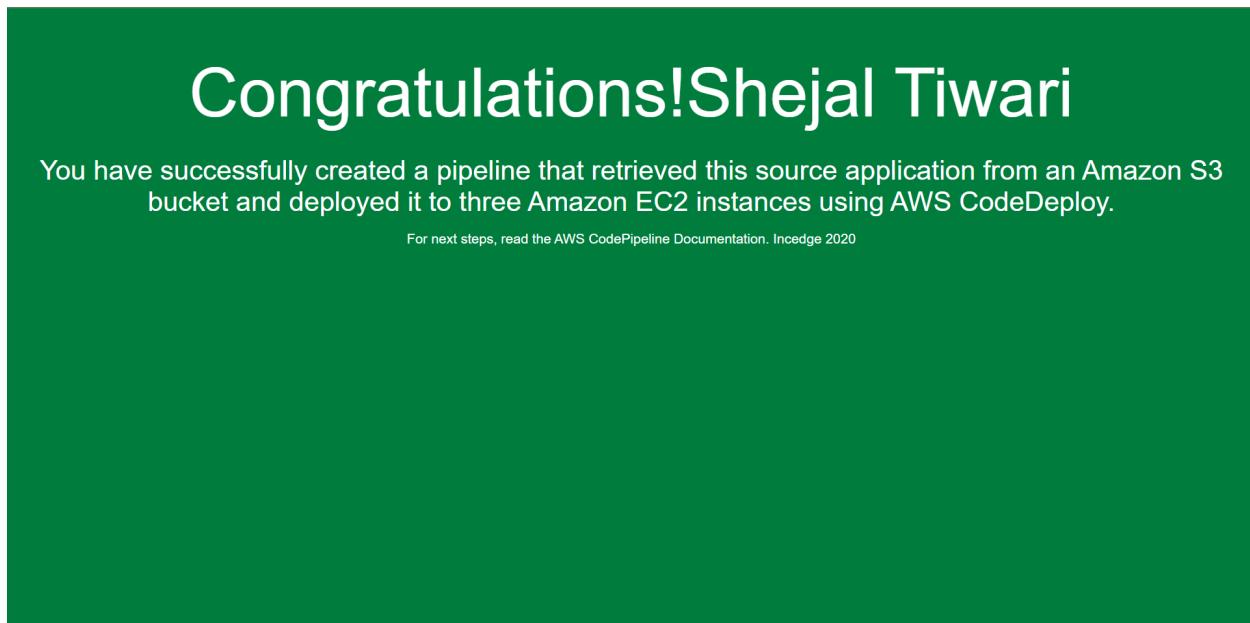


16. Go back to your environment and click on the domain to see the deployed github repository



The screenshot shows the AWS Elastic Beanstalk console. On the left, there's a sidebar with options like Applications, Environments, Change history, and Recent environments (which lists "Shejal-env"). The main area is titled "Environments (1) Info" and shows a single environment named "Shejal-env". The details for this environment include its status as "Ok", application name "Shejal", platform "PHP 8.3 running on Amazon Linux 2", domain "Shejal-env.eba-ndmy9ib7.us-east-1.amazonaws.com", tier "WebServer", and creation date "August 2024". A prominent orange button at the top right says "Create environment". At the bottom of the page, the URL "https://us-east-1.console.aws.amazon.com/elasticbeanstalk/home?region=us-east-1#" is shown along with copyright information and links for Privacy, Terms, and Cookie preferences.

17. To see the proper working of pipeline. Go to github make the changes in the code commit it. Here I have added the my name in the index.html file. The changes can be seen instantly.



## EXPERIMENT NO: 3

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

### STEPS :

#### 1. Create Instances:

We initiated the creation of three virtual machines or instances, naming them Master, Worker1, and Worker2. These instances will act as the nodes in our Kubernetes cluster.

Master	i-0767a02f53056b254	<span style="color: green;">Running</span>	<span style="color: green;">Q</span>	t3.small	<span style="color: blue;"> Initializing</span>	<a href="#">View alarms +</a>
worker-1	i-0a98404682c2bf690	<span style="color: green;">Running</span>	<span style="color: green;">Q</span>	t3.small	<span style="color: blue;"> Initializing</span>	<a href="#">View alarms +</a>
worker-2	i-0a9ea3e263873d151	<span style="color: green;">Running</span>	<span style="color: green;">Q</span>	t3.small	<span style="color: blue;"> Initializing</span>	<a href="#">View alarms +</a>

#### 2. Install Docker:

On each instance, execute the following commands to gain superuser privileges and install Docker:

```
sudo su
```

```
yum install docker -y
```

This step ensures that Docker, a container runtime, is installed on Master, Worker1, and Worker2. Docker is essential for running the containerized applications that Kubernetes will orchestrate.

```
[ec2-user@ip-172-31-93-226 ~]$ sudo su
[root@ip-172-31-93-226 ec2-user]# yum install docker -y
Last metadata expiration check: 0:07:12 ago on Fri Sep 13 11:58:42 2024.
Dependencies resolved.
=====
=====
Package          Architecture      Version       Repo
sitory          Size
=====
=====
Installing:
  docker          x86_64          25.0.6-1.amzn2023.0.2   amaz
onlinux          44 M
Installing dependencies:
  containerd      x86_64          1.7.20-1.amzn2023.0.1   amaz
onlinux          35 M
  iptables-libc  x86_64          1.8.8-3.amzn2023.0.2   amaz
onlinux          401 k
  iptables-nft   x86_64          1.8.8-3.amzn2023.0.2   amaz
onlinux          183 k
```

### 3.Start Docker:

Start the Docker service on each instance to ensure it is running.

```
systemctl start docker
```

- This command activates the Docker service on Master, Worker1, and Worker2, enabling them to run and manage containers.

```
[root@ip-172-31-93-226 ec2-user]# systemctl start docker
[root@ip-172-31-93-226 ec2-user]# █
```

### 4.Install kubeadm:

Prepare the system for Kubernetes installation by configuring SELinux and adding the Kubernetes repository:

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

### Install Kubernetes components:

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

These steps configure SELinux to run in permissive mode, add the Kubernetes repository, and install Kubernetes tools (kubelet, kubeadm, and kubectl) on Master, Worker1, and Worker2. The kubelet service is also enabled and started to ensure it runs on boot.

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

Complete!
[root@ip-172-31-84-46 ec2-user]# sudo systemctl enable --now kubelet
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service → /usr/lib/systemd/system/kubelet.service.
```

## 5. Confirm Repository:

Verify that the Kubernetes repository is correctly configured by listing all enabled repositories:

bash

Copy code

yum repo list

Execute this command on Master, Worker1, and Worker2 to confirm that the Kubernetes repository has been added and is available for package installations.

```
[root@ip-172-31-84-46 ec2-user]# yum repolist
repo id                                repo name
amazonlinux                             Amazon Linux 2023 repository
kernel-livepatch                         Amazon Linux 2023 Kernel Livepatch repository
kubernetes                             Kubernetes
[root@ip-172-31-84-46 ec2-user]#
```

## 6.Initialize kubeadm on Master Node:

On the Master node, initialize the Kubernetes cluster using kubeadm. This process sets up the Kubernetes control plane and generates commands for joining worker nodes:

Copy the commands displayed in the output of the initialization process to configure permissions and obtain the join token. This includes a join command link needed for worker nodes to connect to the master.

```
Your Kubernetes control-plane has initialized successfully!

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

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

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

export KUBECONFIG=/etc/kubernetes/admin.conf

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

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

kubeadm join 172.31.84.46:6443 --token jl06ac.t7cdzxf0x5eddmsl \
    --discovery-token-ca-cert-hash sha256:4a152b913ee4b60dc2126d55f631b86d0dafb7d58132416c4f32f0668ac553be
[root@ip-172-31-84-46 ec2-user]#
```

In the screenshot you can see the commands written in the 3rd 4th and 5th line

Copy that command , this command is used to add right permission to the user

Also copy the 7th line , here it is the credential for the user .

Also copy the last 2 lines it is a link used to join the nodes

## 7.Join Worker Nodes:

On Worker1 and Worker2, use the join command provided by the Master node to connect them to the Kubernetes cluster:

```
kubeadm join 172.31.84.46:6443 --token jl06ac.t7cdzxf0x5eddmsl \
    --discovery-token-ca-cert-hash
sha256:4a152b913ee4b60dc2126d55f631b86d0dafb7d58132416c4f32f0668ac553be
```

This command allows Worker1 and Worker2 to register themselves with the Master node and become part of the Kubernetes cluster.

Verify Nodes:

On the Master node, check the status of all nodes in the cluster to ensure that Worker1 and Worker2 have successfully joined:

```
kubectl get nodes
```

This command lists all nodes and their status within the Kubernetes cluster, helping to confirm successful node registration and connectivity.

```
[root@ip-172-31-84-46 ec2-user]# kubectl get node
NAME                  STATUS    ROLES     AGE      VERSION
ip-172-31-84-46.ec2.internal   NotReady  control-plane  56s      v1.31.1
[root@ip-172-31-84-46 ec2-user]# kubectl get node
The connection to the server 172.31.84.46:6443 was refused - did you specify the right host or port?
[root@ip-172-31-84-46 ec2-user]# kubectl get node
NAME                  STATUS    ROLES     AGE      VERSION
ip-172-31-84-46.ec2.internal   NotReady  control-plane  5m1s      v1.31.1
[root@ip-172-31-84-46 ec2-user]#
```

And this was the output when i tried connecting it with worker 1 and worker 2

```
          #_
          ##_#
          #####_
          ~~~\##_\
          ~~~ \##_|
          ~~~ \#/ _>
          ~~~ \~' _> https://aws.amazon.com/linux/amazon-linux-2023
          ~~~ _/ \
          ~~~ _/ _/
          /m/_
Last login: Fri Sep 13 12:04:33 2024 from 18.206.107.28
[ec2-user@ip-172-31-87-149 ~]$ sudo su
[root@ip-172-31-87-149 ec2-user]# kubeadm join 172.31.84.46:6443 --token qq448c.8c0zquwywz3eqt0bh sha256:49cc770e646aab704883a3d9fb30e6146fdd83e782ebd5ec3194dacee59f2257 --discovery-token-ca-cert-has
[preflight] Running pre-flight checks
  [WARNING FileExisting-socat]: socat not found in system path
  [WARNING FileExisting-tc]: tc not found in system path
```

So there was a failure in connecting the worker 1 and 2 with the master even if i provided the joining link.

The Cloud shell didn't proceed further.

## CONCLUSION :

We created and configured three instances (Master, Worker1, and Worker2), installed Docker and Kubernetes on all nodes, and initialized the Kubernetes control plane on the Master node. Despite providing the join command to Worker1 and Worker2, they failed to connect to the Master node.

**Connection Refused Errors:** The Kubernetes components are failing to connect to the API server at <https://172.31.84.46:6443>, resulting in "connection refused" errors.

**No Running Containers:** The `docker ps -a` command shows no containers, which means the Kubernetes components are not running in Docker.

**CrashLoopBackOff:** There are errors indicating that the containers for Kubernetes components are restarting repeatedly but failing to start properly.

**Network Plugin Not Ready:** The error message "Network plugin returns error: cni plugin not initialized" suggests issues with the network plugin required for Kubernetes.

## Experiment 4

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

Step 1: Set Up EC2 Instances. Select Amazon Linux with t2.medium and right ssh network configurations in the inbound rules.

The screenshot shows the AWS Lambda console interface. A modal window titled "Create New Function" is open, prompting for a function name ("HelloWorld") and runtime ("Python 3.8"). The "Code" tab is selected, displaying the "Hello World" code sample. The "Environment" tab shows environment variables: LAMBDA\_TASK\_ROOT and AWS\_LAMBDA\_FUNCTION\_NAME. The "Logs" tab shows the log stream for the function.

2) Select a key pair, to create one. Click on create Also check if the pem file got downloaded in your pc.

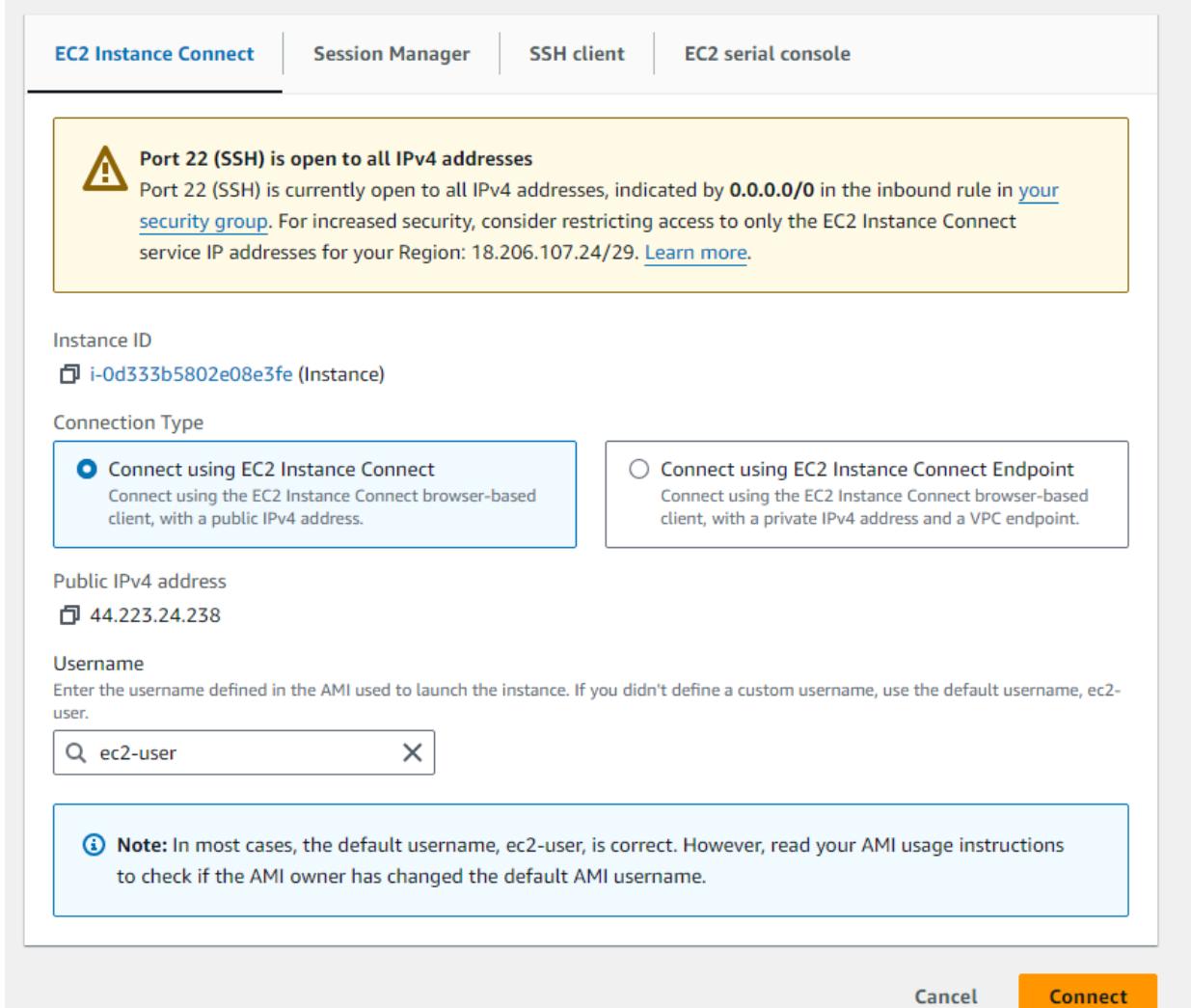
The screenshot shows the AWS Lambda console interface. A modal window titled "Create New Function" is open, prompting for a function name ("HelloWorld") and runtime ("Python 3.8"). The "Code" tab is selected, displaying the "Hello World" code sample. The "Environment" tab shows environment variables: LAMBDA\_TASK\_ROOT and AWS\_LAMBDA\_FUNCTION\_NAME. The "Logs" tab shows the log stream for the function.

When you are going to perform the command in your cmd/git bash

You need to write this command.

ssh -i <keyname> ec2-user@<public id address of your ec2 instances> after launching the instance.

3) Once created, go back to the instances page. Click on the instance id. Then, click on connect.



## Step 2: Installation of Docker

1) Use command ‘sudo su’ This allows you to act as the root user of the terminal

```
[ec2-user@ip-172-31-29-96 ~]$ sudo su  
[root@ip-172-31-29-96 ec2-user]#
```

2) We can install docker using yum. Use the command ‘yum install docker-y’

```
[root@ip-172-31-29-96 ec2-user]# yum install docker -y
Last metadata expiration check: 0:04:49 ago on Sun Sep 15 10:36:23 2024.
Dependencies resolved.
=====
Package           Architecture   Version
=====
Installing:
  docker          x86_64        25.0.6-1.amzn2023.0.2
Installing dependencies:
  containerd      x86_64        1.7.20-1.amzn2023.0.1
  iptables-libc   x86_64        1.8.8-3.amzn2023.0.2
  iptables-nft    x86_64        1.8.8-3.amzn2023.0.2
  libcgroup       x86_64        3.0-1.amzn2023.0.1
  libnetfilter_conntrack x86_64  1.0.8-2.amzn2023.0.2
  libnftnl        x86_64        1.0.1-19.amzn2023.0.2
  libnftnl        x86_64        1.2.2-2.amzn2023.0.2
  pigz            x86_64        2.5-1.amzn2023.0.3
  runc            x86_64        1.1.13-1.amzn2023.0.1

Transaction Summary
=====
Install 10 Packages

Total download size: 84 M
Installed size: 317 M
Downloading Packages:
[===(1/10): containerd-1.7.20-1.amzn2023.0.1.x86_64]
=====
  Running scriptlet: docker-25.0.6-1.amzn2023.0.2.x86_64
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.

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

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

Complete!
[root@ip-172-31-29-96 ec2-user]# ]
```

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

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

```
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opt": {
    "max-size": "100m"
  }
}
```

```
}, "  
storage-driver": "overlay2"  
}  
EOF  
sudo systemctl enable docker  
sudo systemctl daemon-reload  
sudo systemctl restart docker
```

```
[root@ip-172-31-29-96 docker]# cd /etc/docker  
cat <<EOF | sudo tee /etc/docker/daemon.json  
{  
"exec-opts": ["native.cgroupdriver=systemd"],  
"log-driver": "json-file",  
"log-opt": {  
"max-size": "100m"  
},  
"storage-driver": "overlay2"  
}  
EOF  
sudo systemctl enable docker  
sudo systemctl daemon-reload  
sudo systemctl restart docker  
{  
"exec-opts": ["native.cgroupdriver=systemd"],  
"log-driver": "json-file",  
"log-opt": {  
"max-size": "100m"  
},  
"storage-driver": "overlay2"  
}  
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/  
systemd/system/docker.service.  
[root@ip-172-31-29-96 docker]# █
```

### Step 3: Installing Kubernetes

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

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

The screenshot shows the Kubernetes documentation website. The top navigation bar includes links for Documentation, Kubernetes Blog, Training, Partners, Community, Case Studies, Versions, and English. A search bar is also present. On the left, there's a sidebar with links for Documentation, Getting started, Learning environment, Production environment (Container Runtimes, Installing Kubernetes with deployment tools, Bootstrapping clusters), and Without a package manager. The main content area is titled 'the installation guide for your desired minor version.' It shows sections for Red Hat-based distributions and Debian-based distributions, with 'Red Hat-based distributions' currently selected. Below this, there's a section for 'Without a package manager' with a sub-section for '1. Set SELinux to permissive mode:' which includes a command example: `# Set SELinux in permissive mode (effectively disabling it)`. To the right, there's a sidebar with links for 'Edit this page', 'Create child page', 'Create documentation issue', and 'Print entire section'. Further down, there's a 'Before you begin' section with links for Verify the MAC address and product UUID, Check network adapters, Check required ports, Swap configuration, and Installing a container runtime.

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

`sudo setenforce 0`

→sets SELinux to permissive mode

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

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

Run the following commands:

- `sudo setenforce 0`
- `sudo sed-i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config`

```
[root@ip-172-31-29-96 docker]# sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
```

- `cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo`  
`[kubernetes]`  
`name=Kubernetes`  
`baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/`  
`enabled=1`  
`gpgcheck=1`  
`gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repo/repodata/repomd.xml.key`  
`exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni`  
`EOF`

```
[root@ip-172-31-29-96 docker]# cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repo/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repo/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
[root@ip-172-31-29-96 docker]# ]
```

- yum repolist

This command shows the repositories created on the machine.

```
[root@ip-172-31-29-96 docker]# yum repolist
repo id                                repo name
amazonlinux                             Amazon Linux 2023 repository
kernel-livepatch                         Amazon Linux 2023 Kernel Livepatch repository
kubernetes                             Kubernetes
[root@ip-172-31-29-96 docker]# ]
```

Next step is to install kubelet, kubeadm, kubectl

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

```
=====
Installing      : kubelet-1.31.1-150500.1.1.x86_64
9/9
Running scriptlet: kubelet-1.31.1-150500.1.1.x86_64
Verifying       : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64
Verifying       : libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64
Verifying       : libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64
Verifying       : libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64
Verifying       : cri-tools-1.31.1-150500.1.1.x86_64
Verifying       : kubeadm-1.31.1-150500.1.1.x86_64
Verifying       : kubectl-1.31.1-150500.1.1.x86_64
Verifying       : kubelet-1.31.1-150500.1.1.x86_64
Verifying       : kubernetes-cni-1.5.1-150500.1.1.x86_64

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

Complete!
[root@ip-172-31-29-96 docker]# ]
```

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

- sudo systemctl enable --now kubelet

```
[root@ip-172-31-29-96 docker]# sudo systemctl enable --now kubelet
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service → /usr/lib/systemd/system/kubelet.service.
[root@ip-172-31-29-96 docker]# ]
```

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

```
[root@ip-172-31-29-96 docker]# sudo swapoff -a
echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
net.bridge.bridge-nf-call-iptables=1
net.bridge.bridge-nf-call-iptables = 1
[root@ip-172-31-29-96 docker]#
```

3) Firstly, we need to initialize kubernetes. For This, run the command:

- sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=NumCPU,Mem

```
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 172.31.29.96:6443 --token 0b6cct.1cm4p25mefy05fh1 \
    --discovery-token-ca-cert-hash sha256:ae83caa940837900b62231f4f381a06d69b4d25b0207ce5fff9a943e6757b6a8
[root@ip-172-31-29-96 docker]#
```

4) From The Output, we receive the following commands:

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

Run These Commands

```
[root@ip-172-31-29-96 docker]# mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
[root@ip-172-31-29-96 docker]#
```

5) Add a common networking plugin flannel using this command

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

```
[root@ip-172-31-29-96 docker]# kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flanneld.kube-flanneld.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
[root@ip-172-31-29-96 docker]#
```

### Step3:Deploy nginx server

- 1) Now that the cluster is set,apply the deployment file of nginx using this command

- kubectl apply-fhttps://k8s.io/examples/pods/simple-pod.yaml

```
[root@ip-172-31-29-96 docker]# kubectl apply -f https://k8s.io/examples/pods/simple-pod.yaml
pod/nginx created
[root@ip-172-31-29-96 docker]#
```

### 2) Use The Command

- kubectl get pods

To Get the list of pods in cluster.

```
[root@ip-172-31-29-96 docker]# kubectl get pods
NAME      READY     STATUS    RESTARTS   AGE
nginx    0/1      Pending    0          44s
[root@ip-172-31-29-96 docker]#
```

This output shows that the pod is in a 'PENDING' state, change it to RUNNING state, run the following commands.

- kubectl describe pod nginx: Provides Details About Your Pod

This command is used to get details about the pod and potential issues with the pod

```
[root@ip-172-31-29-96 docker]# kubectl describe pod nginx
Name:           nginx
Namespace:      default
Priority:       0
Service Account: default
Node:           <none>
Labels:          <none>
Annotations:    <none>
Status:         Pending
IP:
IPs:            <none>
Containers:
  nginx:
    Image:        nginx:1.14.2
    Port:         80/TCP
    Host Port:   0/TCP
    Environment: <none>
```

```
Events:
  Type     Reason     Age   From           Message
  ----     -----     ---   ----           -----
  Warning  FailedScheduling  2m43s  default-scheduler  0/1 nodes are available: 1 node(s) had
untolerated taint {node-role.kubernetes.io/control-plane: }. preemption: 0/1 nodes are available: 1 Preemption is not helpful for scheduling.
  Warning  FailedScheduling  1s    default-scheduler  0/1 nodes are available: 1 node(s) had
untolerated taint {node-role.kubernetes.io/control-plane: }. preemption: 0/1 nodes are available: 1 Preemption is not helpful for scheduling.
[root@ip-172-31-29-96 docker]#
```

3) From this output, we get to know that the node has some untolerated taint. To remove this, use

- kubectl taintnodes --allnode-role.kubernetes.io/control-plane:NoSchedule-

```
[root@ip-172-31-29-96 docker]# kubectl taint nodes --all node-role.kubernetes.io/control-plane:NoSchedule-
kubectl taint nodes --all node-role.kubernetes.io/control-plane:NoSchedule-
node/ip-172-31-29-96.ec2.internal untainted
```

4) Now, we check the status of the pod by running “kubectl get pods” again

```
[root@ip-172-31-29-96 docker]# kubectl get pods
NAME    READY    STATUS    RESTARTS   AGE
nginx  1/1     Running   2 (68s ago)  9m54s
[root@ip-172-31-29-96 docker]#
```

5) Now, change the port to which you want to host your server on using command

- kubectl port-forward nginx <port number you want to host on>:80

```
[root@ip-172-31-29-96 docker]# kubectl port-forward pod/nginx 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
```

6) To check whether the deployment was successful, run the command

- curl --head http://127.0.0.1:<port number given by you>

If the terminal returns a status code of 200, it means that the deployment is successful.

## Conclusion:

### Few Difficulties faced during the practical:

**1. EC2 Instance Connection Issue:** After connecting the instances it shows connection failure so with the notification in red at the top for bash so in that case try another instances if still it now works go to your git bash and connection for ssh connection with keyname and public ip address of the instance. And then proceed with the further installation of docker and kubernetes.

**2. Kubernetes Installation Issue:** Some time the commands for setting up your kubernetes repository doesn't work and goes in failure in that case try reconnecting yours instance and start again with all the commands.

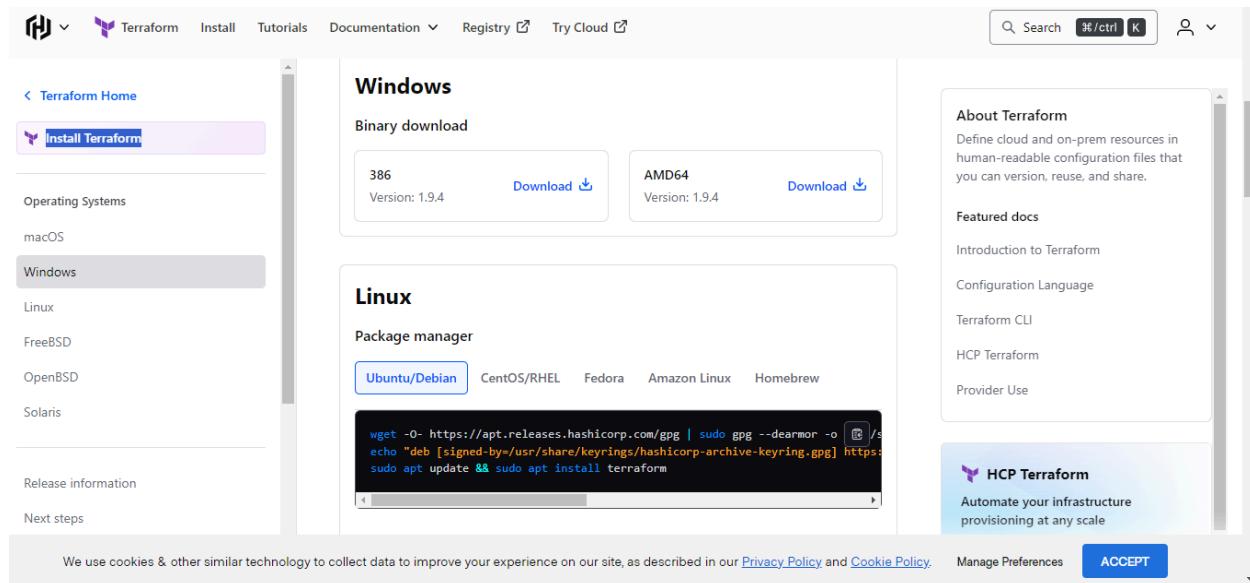
**3. Nginx Deployment Issues:** The Nginx server might not start because of network problems in Kubernetes or restrictions on the control plane that stop the pod from running therefore in this case you need to re-run the commands again and again if it shows that "have you connected to the right host ?"

## EXPERIMENT NO. 05

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

### 1.Download and Install Terraform:

- If you haven't already, download Terraform from the official website.
- Install Terraform by extracting the downloaded zip file to a directory of your choice. Note the path where Terraform executable (`terraform.exe` on Windows) is located.



## 2.Find the Terraform Executable Path:

- After installation, locate the directory where Terraform executable (`terraform.exe`) is present. For example, it could be in `C:\terraform` or any other path you chose during installation.

## 3.Set Environment Variables:

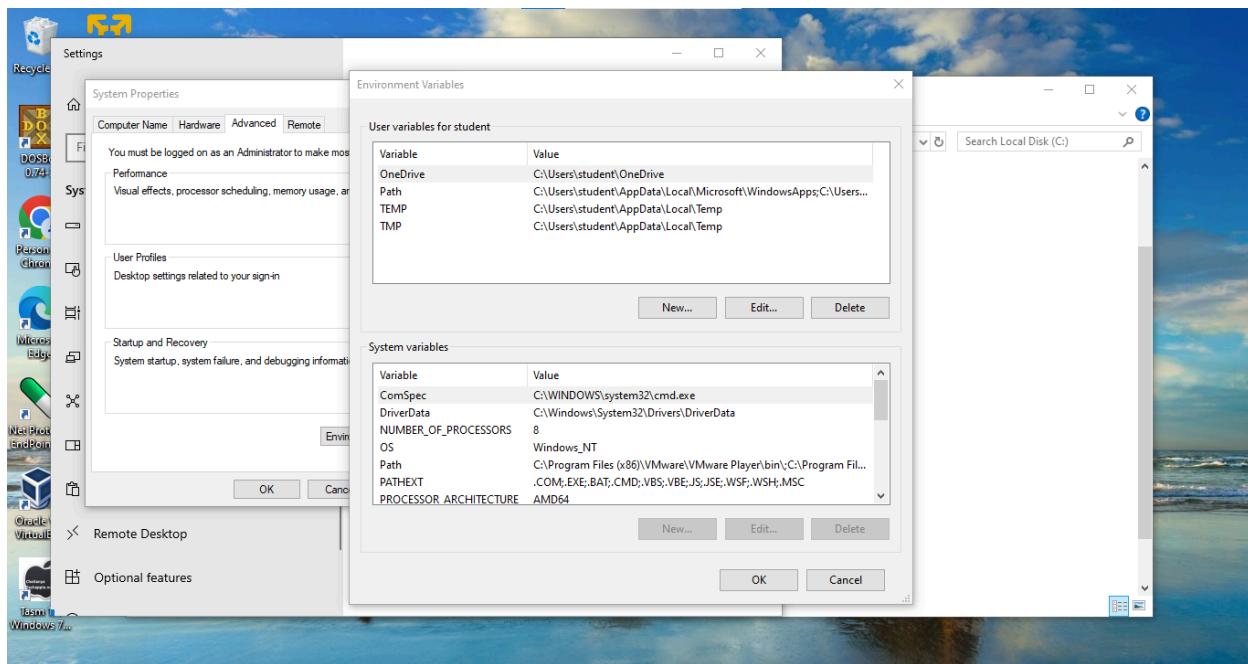
- Right-click on **This PC** or **Computer** on your desktop or in File Explorer, then click **Properties**.
- In the System window, click on **Advanced system settings** on the left-hand side.
- In the System Properties window, click on the **Environment Variables** button.

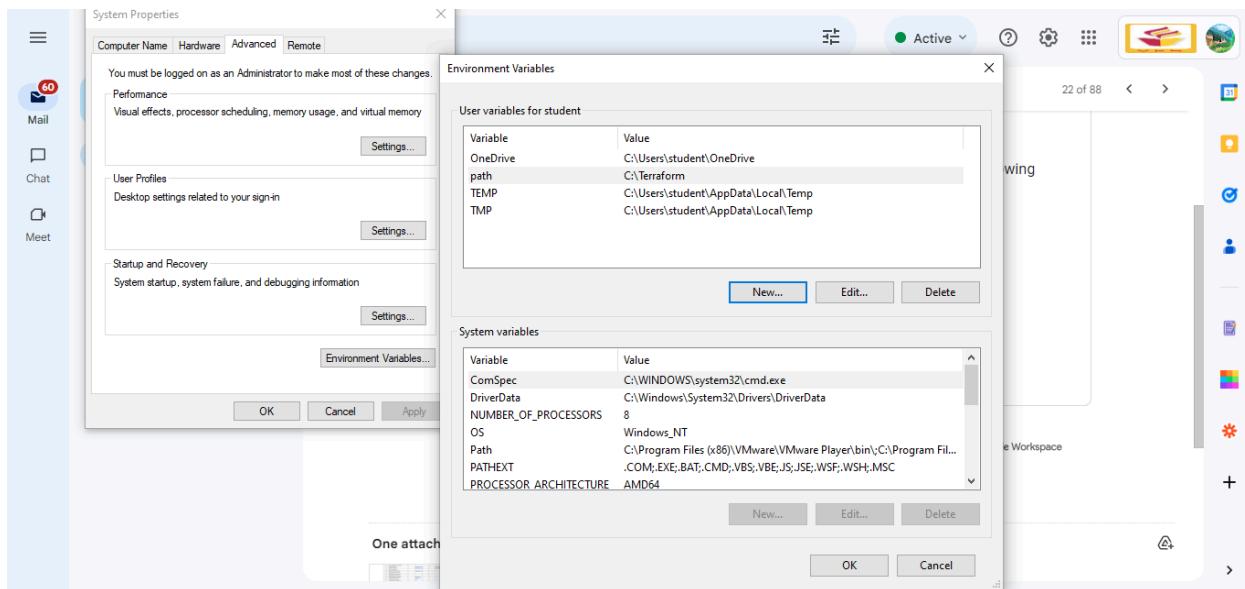
## 4.Edit System Environment Variables:

- In the Environment Variables window (System Properties), under the section **System Variables**, find and select the variable named **Path**, then click **Edit**.

## 5.Add Terraform to Path:

- In the Edit Environment Variable window, click **New** to add a new entry.
- Enter the path to the directory where Terraform executable (`terraform.exe`) is located. For example, if Terraform is installed in `C:\terraform`, add `C:\terraform` to the list.





## 6.Verify Installation:

- Open a new command prompt (to ensure it picks up the updated environment variables).
- Type `terraform --version` and press **Enter**. You should see the Terraform version information printed on the screen if the path configuration was successful.

If PowerShell is unable to recognize the `terraform` command, even though you've installed Terraform. This issue typically occurs when the directory containing the Terraform executable is not included in the `PATH` environment variable that PowerShell is using.

To resolve this issue, you need to add the directory containing the Terraform executable to your PowerShell session's `PATH` variable.

### Set the `PATH` Variable Temporarily for the Current Session:

- In PowerShell, you can set the `PATH` variable for the current session using the following command:  
`powershell`  
 Copy code  
`$env:PATH += ";C:\Terraform"`

```
PS C:\Users\student> $env:PATH += ";C:\Terraform"
>>
PS C:\Users\student> terraform --version
Terraform v1.9.4
on windows_amd64
PS C:\Users\student> ■
```

## AdvanceDevops Experiment: 6

**AIM: To Build, change, and destroy AWS /GCP/ Microsoft Azure/ Digital Ocean infrastructure using Terraform. (S3 bucket or Docker)**

**To Create docker image using terraform**

We need Download and Install Docker Desktop from <https://www.docker.com/>

**Step 1:** Check the docker functionality

```
PS C:\Users\INFT505-16> 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
```

```
PS C:\Users\INFT505-16> docker --version
Docker version 24.0.6, build ed223bc
PS C:\Users\INFT505-16> |
```

**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. Copy the Script into it. **Script:**

```
terraform {  
    required_providers {  
        docker = {  
            source = "kreuzwerker/docker"  
            version = "2.21.0"  
        }  
    }  
}  
  
provider "docker" {  
    host = "npipe://////pipe//docker_engine"  
}  
  
# Pulls the Ubuntu image  
resource "docker_image" "ubuntu" {  
    name = "ubuntu:latest"  
}  
  
# Create a container  
resource "docker_container" "foo" {  
    image = docker_image.ubuntu.image_id  
    name = "foo"  
}
```

### Step 3: Execute Terraform Init command to initialize the resources

```
C:\Users\INFT505-16>cd desktop\TerraformScripts\Docker  
C:\Users\INFT505-16\Desktop\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 BD880C4571C6104C)  
Partner and community providers are signed by their developers.  
If you'd like to know more about provider signing, you can read about it here:  
https://www.terraform.io/docs/cli/plugins/signing.html  
Terraform has created a lock file .terraform.lock.hcl to record the provider  
selections it made above. Include this file in your version control repository  
so that Terraform can guarantee to make the same selections by default when  
you run "terraform init" in the future.
```

Terraform has been successfully initialized!

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

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

### Step 4: Execute Terraform apply to apply the configuration, which will automatically create and run the Ubuntu Linux container based on our configuration. Using command : “terraform apply”

```
C:\Users\INFT505-16\Desktop\TerraformScripts\Docker>terraform apply  
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:  
+ create  
Terraform will perform the following actions:  
  
# docker_container.foo will be created  
+ resource "docker_container" "foo" {  
    + attach          = false  
    + bridge          = (known after apply)  
    + command         = [  
        + "sleep",  
        + "3600",  
    ]  
    + 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)
```

```

+ output      = (known after apply)
+ repo_digest = (known after apply)
}

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

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

Enter a value: yes

docker_image.ubuntu: Creating...
docker_image.ubuntu: STILL creating... [18s elapsed]
docker_image.ubuntu: STILL creating... [28s elapsed]
docker_image.ubuntu: STILL creating... [38s elapsed]
docker_image.ubuntu: STILL creating... [48s elapsed]
docker_image.ubuntu: STILL creating... [58s elapsed]
docker_image.ubuntu: Creation complete after 63s [id=sha256:edbfe74c41f8a3591ce542e137cf28ea84dd83e6dFbc9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Creating...
docker_container.foo: Creation complete after 2s [id=353bd8cae537e335931797ed86d2b683c682529b71c2af7d5b72F3c89eed2b11]

```

Docker images, After Executing Apply step:

```
C:\Users\INFT505-16\Desktop\TerraformScripts\Docker>docker images
REPOSITORY      TAG          IMAGE ID      CREATED        SIZE
ubuntu          latest       edbfe74c41f8  3 weeks ago   78.1MB
sonarqube       latest       3183d6818c6e  10 months ago  716MB
```

**Step 5:** Execute Terraform destroy to delete the configuration, which will automatically delete the Ubuntu Container.

```
C:\Users\INFT505-16\Desktop\TerraformScripts\Docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3591ce542e137cf28ea84dd83e6dFbc9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=353bd8cae537e335931797ed86d2b683c682529b71c2af7d5b72F3c89eed2b11]

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

Terraform will perform the following actions:

# docker_container.foo will be destroyed
- resource "docker_container" "foo" {
  - attach           = false => null
  - command         = [
    - "sleep",
    - "3600",
  ] => null
  - cpu_shares      = 0 => null
  - dns              = [] => null
  - dns_opts         = [] => null
  - dns_search       = [] => null
  - entrypoint       = [] => null
  - env              = [] => null
  - gateway          = "172.17.0.1" => null
}
```

```

# docker_image.ubuntu will be destroyed
- resource "docker_image" "ubuntu" {
    - id          = "sha256:edbfe74c41f8a3581ce542e137cf28ea84dd83e6df8c9d66519b6ad761c2598aubuntu:latest" => null
    - image_id   = "sha256:edbfe74c41f8a3581ce542e137cf28ea84dd83e6df8c9d66519b6ad761c2598a" => null
    - latest     = "sha256:edbfe74c41f8a3581ce542e137cf28ea84dd83e6df8c9d66519b6ad761c2598a" => null
    - name       = "ubuntu:latest" => null
    - repo_digest = "ubuntu@sha256:8a37d68f4f73ebf3d4efaefbf66379bf3728992a8838616888f04e34a9ab63ee" => null
}

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

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

docker_container.foo: Destroying... [id=353bd0cae537e335931797ed86d2b683c682528b71c2af7d5b72f3c09eed2b11]
docker_container.foo: Destruction complete after 0s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3581ce542e137cf28ea84dd83e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image.ubuntu: Destruction complete after 1s

Destroy complete! Resources: 2 destroyed.

```

**Step 6:** This command outputs the state or plan in a human-readable format, helping you review the details of your configuration..

```

C:\Users\INFT505-16\Desktop\TerraformScripts\Docker>terraform show
# docker_container.foo:
resource "docker_container" "foo" {
    attach      = false
    bridge      = null
    command     = [
        "sleep",
        "3600",
    ]
    cpu_set     = null
    cpu_shares  = 0
    domainname  = null
    entrypoint   = []
    env         = []
    gateway     = "172.17.0.1"
    hostname    = "353bd0cae537"
    id          = "353bd0cae537e335931797ed86d2b683c682528b71c2af7d5b72f3c09eed2b11"
    image       = "sha256:edbfe74c41f8a3581ce542e137cf28ea84dd83e6df8c9d66519b6ad761c2598a"
    init        = false
    ip_address  = "172.17.0.2"
    ip_prefix_length = 16
}

```

**Step 7:** This command generates a visual graph of your Terraform resources, which can help you understand the dependencies and relationships between them.

```

C:\Users\INFT505-16\Desktop\TerraformScripts\Docker>terraform graph
digraph G {
    rankdir = "RL";
    node [shape = rect, fontname = "sans-serif"];
    "docker_container.foo" [label="docker_container.foo"];
    "docker_image.ubuntu" [label="docker_image.ubuntu"];
    "docker_container.foo" -> "docker_image.ubuntu";
}

```

**Step 8:** This command lists all the resources tracked by the Terraform state, allowing you to see which resources have been created and are being managed by Terraform..

```
C:\Users\INFT505-16\Desktop\TerraformScripts\Docker>terraform state list  
docker_container.foo  
docker_image.ubuntu
```

```
C:\Users\INFT505-16\Desktop\TerraformScripts\Docker>
```

## AdvanceDevops Experiment 7

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

### Integrating Jenkins with SonarQube:

Windows installation

Step 1 Install JDK 1.8

Step 2 download and install jenkins

<https://www.blazemeter.com/blog/how-to-install-jenkins-on-windows>

#### Ubuntu installation

<https://www.digitalocean.com/community/tutorials/how-to-install-java-with-a-pt-on-ubuntu-20-04#installing-the-default-jre-jdk>

Step 1 Install JDK 1.8

sudo apt-get install openjdk-8-jre

sudo apt install default-jre

<https://www.digitalocean.com/community/tutorials/how-to-install-jenkins-on-ubuntu-20-04>

[Open SSH](#)

### Prerequisites:

- [Jenkins installed](#)
- [Docker Installed](#) (for SonarQube)  
(sudo apt-get install docker-ce=5.20.10.15~3-0~ubuntu-jammy  
docker-ce-cli=5:20.10.15~3-0~ubuntu-jammy containerd.io docker-compose-plugin)
- SonarQube Docker Image

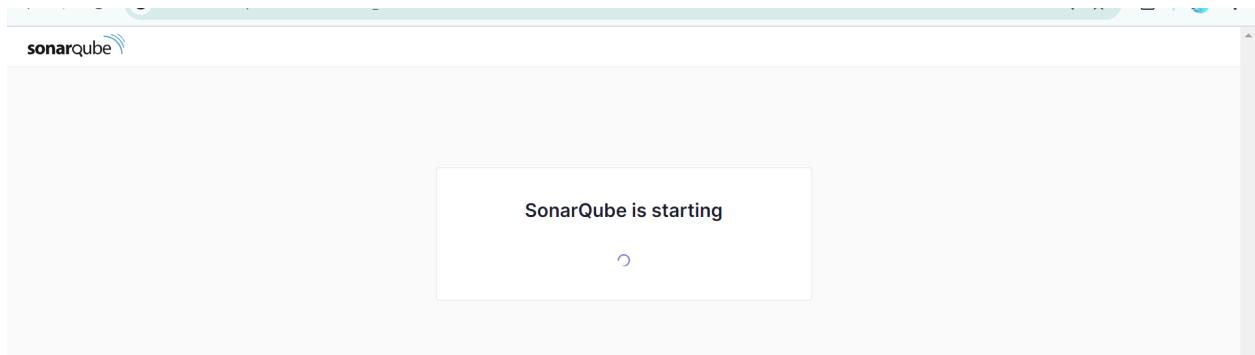
### Steps to integrate Jenkins with SonarQube

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 -

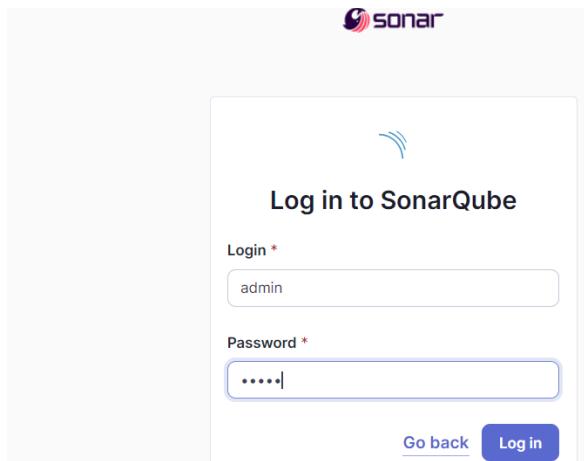
```
PS C:\Users\91900> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
8b2a833004ac39a9b009118bacac47e5808c9ec8df3f59f8657bd23fa23f48f2
```

Warning: run below command only once  
docker run -d --name sonarqube -e SONAR\_ES\_BOOTSTRAP\_CHECKS\_DISABLE=true -p 9000:9000 sonarqube:latest

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



- Login to SonarQube using username *admin* and password *admin*.



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

1 of 2

## Create a local project

Project display name \*

sonarqube-test



Project key \*

sonarqube-test



Main branch name \*

main

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

2 of 2

## Set up project for Clean as You Code

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

Choose the baseline for new code for this project

 Use the global setting

Previous version

Any code that has changed since the previous version is considered new code.

Recommended for projects following regular versions or releases.

 Define a specific setting for this project Previous version

Any code that has changed since the previous version is considered new code.

Recommended for projects following regular versions or releases.

Setup the project and come back to Jenkins Dashboard.

Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.

The screenshot shows the Jenkins Plugins page. A search bar at the top contains the text "sonar". Below the search bar, there are three tabs: "Updates" (with 34 notifications), "Available plugins" (selected), and "Installed plugins". In the "Available plugins" section, a search result for "SonarQube Scanner 2.17.2" is displayed. The plugin is categorized under "External Site/Fool Integrations" and "Build Reports". It is marked as "Released" and was last updated "7 mo 8 days ago". A brief description states: "This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality." To the right of the plugin card, there are "Install" and "Uninstall" buttons.

6. Under Jenkins ‘Configure System’, look for SonarQube Servers and enter the details.

Enter the Server Authentication token if needed.

SonarQube servers

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

Environment variables

SonarQube installations

List of SonarQube installations

Name: sonarqube

Server URL: http://localhost:9000

Server authentication token: - none -

Advanced

**Save** **Apply**

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

Add SonarScanner for MSBuild

SonarQube Scanner installations

Add SonarQube Scanner

SonarQube Scanner

Name: sonarqube

Install automatically

Install from Maven Central

Version: SonarQube Scanner 6.2.0.4584

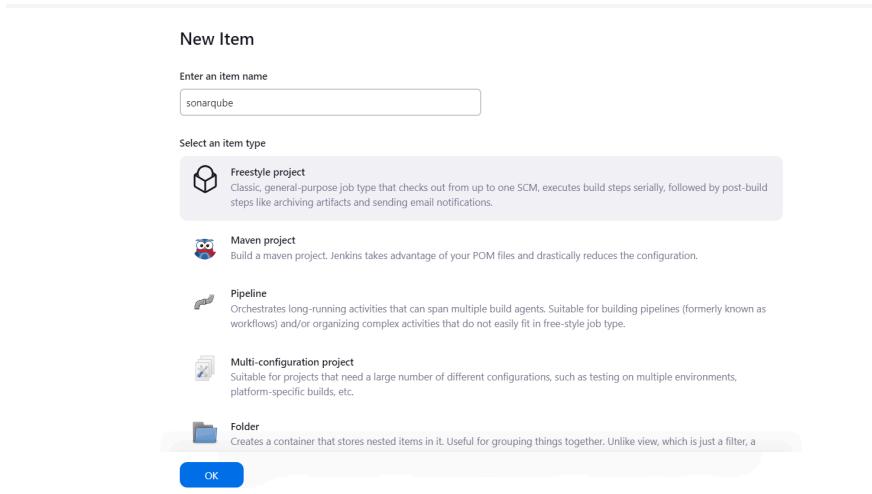
Add Installer

Add SonarQube Scanner

**Save** **Apply**

---

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



## 9. Choose this GitHub repository in Source Code Management.

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

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

test

Configure

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

None

Git

Repositories

Repository URL

Please enter Git repository.

Credentials

- none -

+ Add

Advanced

Add Repository

Branches to build

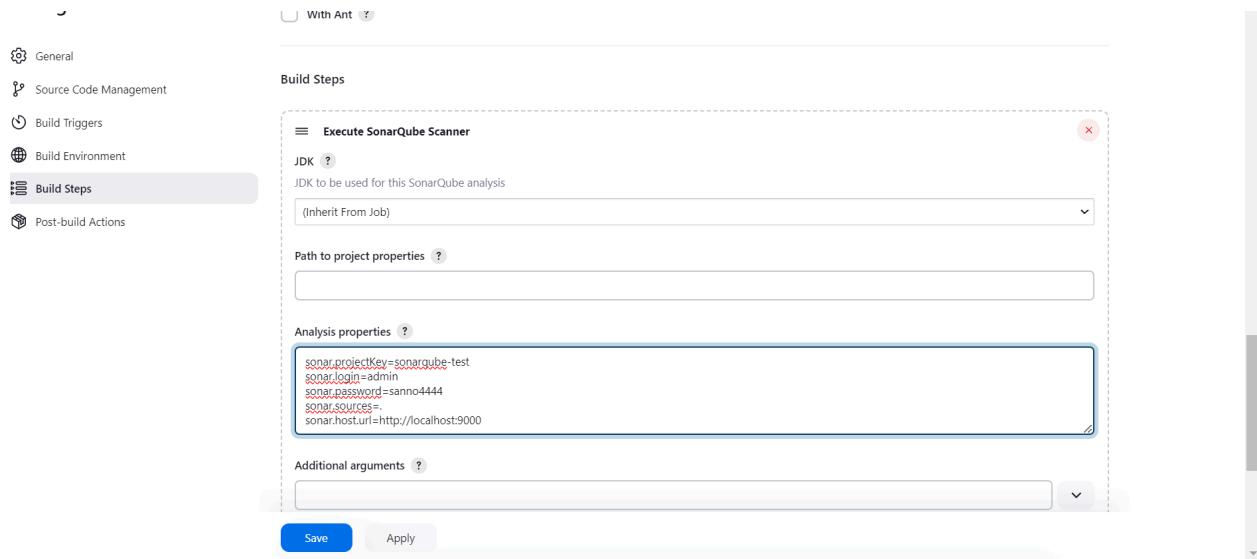
Branch Specifier (blank for 'any')

\*/\*master

Save Apply

the integration.

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



11. Go to [http://localhost:9000/<user\\_name>/permissions](http://localhost:9000/<user_name>/permissions) and allow Execute Permissions to the Admin user.

The screenshot shows the SonarQube 'Global Permissions' configuration page under the 'Administration' tab. The 'Administrator' user ('admin') has been granted the 'Execute Analysis' permission, which is highlighted in blue.

User/Group	Permissions
sonar-administrators	Administrator System, Administer, Execute Analysis (highlighted), Create
sonar-users	Administrator System, Administer, Execute Analysis, Create
Anyone DEPRECATED	Administrator System, Administer, Execute Analysis, Create
Administrator admin	Administrator System, Administer, Execute Analysis (highlighted), Create

12. Run The Build.

**sonarqube**

**Permalinks**

- Last build (#1), 8 min 33 sec ago
- Last stable build (#1), 8 min 33 sec ago
- Last successful build (#1), 8 min 33 sec ago
- Last completed build (#1), 8 min 33 sec ago

**Build History**

#1

Sep 25, 2024, 10:09 AM

Atom feed for all Atom feed for failures

Check the console output.

```

Started by user Vedant Dhone
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
  Cloning repository https://github.com/shazforiot/MSBuild_firstproject.git
    > git.exe init C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube # timeout=10
  Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject.git
    > git.exe --version # timeout=10
    > git --version # 'git' version 2.42.0.windows.2'
    > git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git +refs/heads/*:refs/remotes/origin/* # timeout=10
    > git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject.git # timeout=10
    > git.exe config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
  Avoid second fetch
    > git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
  Checking out Revision f2bc042c04c6e72427c380bcae6d6fee7b49adf (refs/remotes/origin/master)
    > git.exe config core.sparsecheckout # timeout=10
    > git.exe checkout -f f2bc042c04c6e72427c380bcae6d6fee7b49adf # timeout=10
  Commit message: "updated"
  First time build. Skipping changelog.
  Unpacking https://repo1.maven.org/maven2/org/sonarsource/scanner/cli/sonar-scanner-cli/6.2.0.4584/sonar-scanner-cli-6.2.0.4584.zip to
  C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube on Jenkins
  [sonarqube] $ C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube\bin\sonar-scanner.bat -
  Dsonar.host.url=http://localhost:9000 -Dsonar.projectKey=sonarqube-test -Dsonar.login=admin -Dsonar.host.url=http://localhost:9000 -Dsonar.sources=.
  Dsonar.password=sanno4444 -Dsonar.projectBaseDir=C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube

```

```
SonarScanner for .NET 5.x or higher, see https://redirect.sonarsource.com/doc/install-configure-scanner-msbuild.html
10:10:57.397 INFO Sensor C# [csharp] (done) | time=2ms
10:10:57.397 INFO Sensor Analysis Warnings import [csharp]
10:10:57.399 INFO Sensor Analysis Warnings import [csharp] (done) | time=4ms
10:10:57.401 INFO Sensor C# File Caching Sensor [csharp]
10:10:57.405 WARN Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir' property.
10:10:57.405 INFO Sensor C# File Caching Sensor [csharp] (done) | time=5ms
10:10:57.405 INFO Sensor Zero Coverage Sensor
10:10:57.424 INFO Sensor Zero Coverage Sensor (done) | time=19ms
10:10:57.428 INFO SCM Publisher SCM provider for this project is: git
10:10:57.430 INFO SCM Publisher 4 source files to be analyzed
10:10:58.315 INFO SCM Publisher 4/4 source files have been analyzed (done) | time=883ms
10:10:58.324 INFO CPD Executor Calculating CPD for 0 files
10:10:58.363 INFO CPD Executor CPD calculation finished (done) | time=0ms
10:10:58.372 INFO SCM revision ID 'f2bc042c04c6e72427c380bcace6d6fee7b49adf'
10:10:58.843 INFO Analysis report generated in 226ms, dir size=201.0 kB
10:10:58.903 INFO Analysis report compressed in 45ms, zip size=22.2 kB
10:10:59.397 INFO Analysis report uploaded in 491ms
10:10:59.401 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube-test
10:10:59.402 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
10:10:59.403 INFO More about the report processing at http://localhost:9000/api/ce/task?id=2620d8fe-82f7-4a3c-999f-1ed8a7b15249
10:10:59.429 INFO Analysis total time: 30.223 s
10:10:59.431 INFO SonarScanner Engine completed successfully
10:10:59.519 INFO EXECUTION SUCCESS
10:10:59.521 INFO Total time: 47.815s
Finished: SUCCESS
```

### 13. Once the build is complete, check the project in SonarQube.

The screenshot shows the SonarQube web interface. At the top, there's a navigation bar with links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a search icon. Below the navigation is a breadcrumb trail: sonarqube-test / main. The main content area is titled 'main'. It features a 'Quality Gate' section with a green checkmark and the word 'Passed'. A yellow warning box indicates 'The last analysis has warnings. See details'. Below this, there are two tabs: 'New Code' (disabled) and 'Overall Code' (selected). The 'Overall Code' section contains several cards: 'Security' (0 Open issues, A grade), 'Reliability' (0 Open issues, A grade), 'Maintainability' (0 Open issues, A grade), 'Accepted issues' (0, Valid issues that were not fixed), 'Coverage' (On 0 lines to cover), and 'Duplications' (0.0%, On 86 lines).

In this way, we have integrated Jenkins with SonarQube for SAST.

## Conclusion

**1. Docker Container Issues:** The SonarQube container might not start because your system doesn't have enough memory or processing power. SonarQube needs around 2GB of RAM to work properly, so if your system is low on resources, the container won't run.

**2. Login Problems in SonarQube:** You might have trouble logging in with the default username (admin) and password (admin). This could happen if there was a configuration issue with SonarQube or if the default password was changed during previous setups.

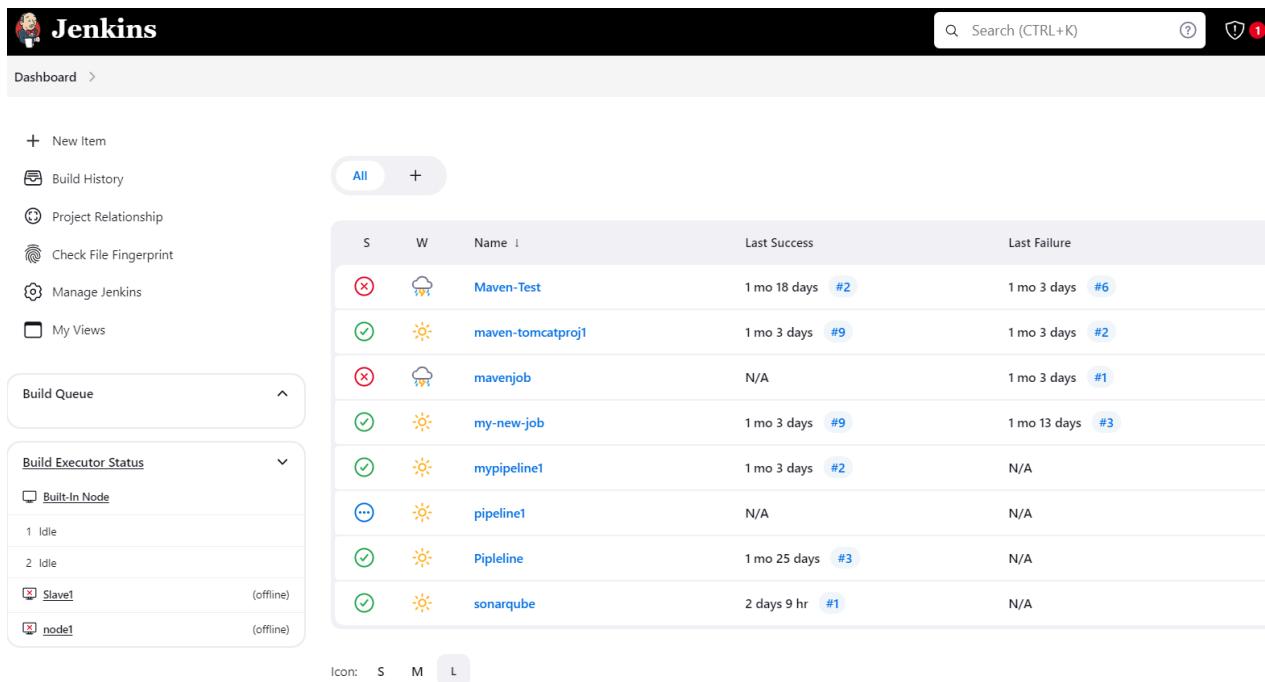
**3. Jenkins Plugin Installation Errors:** While installing the SonarQube Scanner plugin in Jenkins, you might encounter failures due to network issues or proxy settings, preventing the plugin from downloading correctly.

**4. Incorrect SonarQube Configuration in Jenkins:** While configuring SonarQube in Jenkins, entering the wrong project key, username, or password can cause the scan to fail. Ensuring accurate information is critical for a successful scan.

## EXPERIMENT 8

**Aim:** Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web Java / Python application. dive deep into this segment, let's first understand what is meant by the term 'pipeline'?

1. Open Jenkin dashboard.



The screenshot shows the Jenkins dashboard with the following interface elements:

- Header:** Jenkins logo, search bar, and notifications.
- Left Sidebar:**
  - New Item
  - Build History
  - Project Relationship
  - Check File Fingerprint
  - Manage Jenkins
  - My Views
- Build Queue:** A section showing the status of builds: 1 Idle, 2 Idle, 1 Slave (Slave1, offline), and 1 node1 (offline).
- Build Executor Status:** A section showing the status of built-in nodes: 1 Idle, 2 Idle, 1 Slave (Slave1, offline), and 1 node1 (offline).
- Central Table:** A table listing Jenkins jobs with columns: Status (S), Warning (W), Name, Last Success, and Last Failure.
 

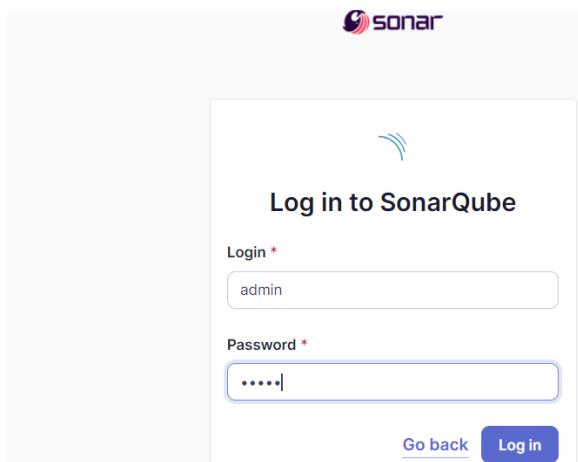
S	W	Name	Last Success	Last Failure
✗	☁️	Maven-Test	1 mo 18 days #2	1 mo 3 days #6
✓	☀️	maven-tomcatproj1	1 mo 3 days #9	1 mo 3 days #2
✗	☁️	mavenjob	N/A	1 mo 3 days #1
✓	☀️	my-new-job	1 mo 3 days #9	1 mo 13 days #3
✓	☀️	mypipeline1	1 mo 3 days #2	N/A
:(	☀️	pipeline1	N/A	N/A
✓	☀️	Pipeline	1 mo 25 days #3	N/A
✓	☀️	sonarqube	2 days 9 hr #1	N/A
- Bottom:** Icon legend (S, M, L) and a link to the Jenkins documentation.

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

```
C:\Users\91900>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
d23acccacd96c274f5f87912674ecf2d9adffff185a940c24740f44b29534485
```

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 *admin*.



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

1 of 2

## Create a local project

Project display name \*

sonarqube-2 ✓

Project key \*

sonarqube-2 ✓

Main branch name \*

main

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

[Cancel](#)

[Next](#)

Setup the project and come back to Jenkins Dashboard.

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

New Item

Enter an item name  
sonarqube14

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.
- Folder**  
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

OK

## 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') {
            sh "<PATH_TO SONARQUBE FOLDER>/bin//sonar-scanner \
                -D sonar.login=<SonarQube_USERNAME> \
                -D sonar.password=<SonarQube_PASSWORD> \
                -D sonar.projectKey=<Project_KEY> \
                -D sonar.exclusions=vendor/**,resources/**,**/*.java \
                -D sonar.host.url=http://127.0.0.1:9000/"
        }
    }
}

```

The screenshot shows the SonarQube Pipeline configuration page. The pipeline definition is as follows:

```
1 w noo {
2   stage('Cloning the GitHub Repo') {
3     git 'https://github.com/sonarforiot/Git.git'
4   }
5
6   stage('SonarQube analysis') {
7     withSonarQubeEnv('sonarqube14') {
8       C:\Users\91900\Downloads\sonar-scanner-cli-6.2.0.4584-windows-x64\b1n\sonar-scanner.bat ^
9         -DSonar.login=admin ^
10        -DSonar.host=http://localhost:9000 ^
11        -DSonar.projectKey=sonarqube14 ^
12        -DSonar.exclusions=src/main/resources/**, **/*.java ^
13        -DSonar.sources=src/main/java ^
14        -DSonar.url=http://localhost:9000/
15      }
16    }
17  }
18 }
```

Below the script, there is a checkbox for "Use Groovy Sandbox" which is checked. At the bottom are "Save" and "Apply" buttons.

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

#### 8. Run The Build.

The screenshot shows the SonarQube Status page for the project "sonarqube14". The main menu items include:

- </> Changes
- ▷ Build Now
- ⚙️ Configure
- trash Delete Pipeline
- 🔍 Full Stage View
- SonarQube
- ⠇ Stages
- ✍️ Rename
- ⓘ Pipeline Syntax

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

```

Dashboard > sonarqube14 > #11
21:22:54.863 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/BSFListener.html for block at line 75. Keep
only the first 100 references.
21:22:54.863 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/BSFListener.html for block at line 41. Keep
only the first 100 references.
21:22:54.864 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/BSFListener.html for block at line 17. Keep
only the first 100 references.
21:22:54.864 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/BSFListener.html for block at line 185. Keep
only the first 100 references.
21:22:54.864 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/BSFListener.html for block at line 185. Keep
only the first 100 references.
21:22:54.864 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/BSFListener.html for block at line 550. Keep
only the first 100 references.
21:22:54.864 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/BSFListener.html for block at line 75. Keep
only the first 100 references.
21:22:54.864 INFO CPD Executor CPD calculation finished (done) | time=443368ms
21:22:55.087 INFO SCM revision ID 'ba799be7e1b57f0da4d012322b04125ce6e1e5d4'
21:27:06.067 INFO Analysis report generated in 5834ms, dir size=127.2 MB
21:27:38.159 INFO Analysis report compressed in 30044ms, zip size=29.6 MB
21:27:52.892 INFO Analysis report uploaded in 1488ms
21:27:52.947 INFO SonarScanner completed successfully and the results at: http://127.0.0.1:9080/dashboard?id=sonarqube14
21:27:52.947 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
21:27:52.947 INFO More about the report processing at https://127.0.0.1:9080/api/cv/task?id=d45ad9df-fd29-4bd5-bfcf-78c329f005f9
21:28:36.635 INFO Analysis total time: 25:19.835 s
21:28:36.702 INFO SonarScanner Engine completed successfully
21:28:37.637 INFO EXECUTION SUCCESS
21:28:38.083 INFO Total time: 25:29.478s
[Pipeline]
[Pipeline] // withSonarQubeEnv
[Pipeline] 
[Pipeline] // stage
[Pipeline] 
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```

## 10. After that, check the project in SonarQube.

The screenshot shows the SonarQube interface for the 'main' project. The top navigation bar includes 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', 'Administration', 'More', and a search bar. The 'Issues' tab is selected. The main content area displays a summary card with a green 'Passed' status. Below this, there are six 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%). A note below 'Accepted issues' says 'Valid issues that were not fixed'. The 'Security Hotspots' section shows 3 hotspots. The overall status is '683k Lines of Code - Version not provided' with a 'Last analysis 37 minutes ago' message.

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

## 11. Code Problems - Consistency

The screenshot shows the SonarQube 'Issues' tab for the 'main' project. The left sidebar has 'My Issues' selected. The main area lists several code problems under the 'Consistency' category:

- Insert a <!DOCTYPE> declaration to before this <html> tag. (Reliability) user-experience
- Remove this deprecated "width" attribute. (Maintainability) html5 obsolete
- Remove this deprecated "align" attribute. (Maintainability) html5 obsolete
- Remove this deprecated "align" attribute. (Consistency) html5 obsolete

Each item includes a checkbox, a brief description, a severity level (e.g., Reliability, Maintainability), and a 'Details' link.

## Intentionality

The screenshot shows the SonarQube interface for the 'main' project. The 'Issues' tab is selected. On the left, the 'Filters' sidebar is open, showing a section for 'Clean Code Attribute' with 'Intentionality' selected. The main panel displays a list of issues under the 'gameoflife-acceptance-tests/Dockerfile' file. Each issue has a checkbox for 'Bulk Change'. The first issue is highlighted with a blue border and labeled 'Intentionality'. It contains the text 'Use a specific version tag for the image.' and 'Maintainability'. Below it are other issues with similar descriptions and status options ('Open', 'Not assigned').

## Severity

The screenshot shows the SonarQube interface for the 'main' project. The 'Issues' tab is selected. The 'Filters' sidebar shows 'Severity' with 'High' selected. The main panel displays a list of issues across three files: 'gameoflife-core/.../com/wakaleo/gameoflife/domain/0\_WhenYouCreateACell.html', 'gameoflife-core/.../com/wakaleo/gameoflife/domain/1\_WhenYouCreateAGrid.html', and 'gameoflife-core/.../com/wakaleo/gameoflife/domain/2\_WhenYouCreateANewUniverse.html'. Each issue has a checkbox for 'Bulk Change'. The first issue in each file is highlighted with a blue border and labeled 'Intentionality'. It contains the text 'Add the "let", "const" or "var" keyword to this declaration of "prop" to make it explicit.' and 'Maintainability'. Below it are other issues with similar descriptions and status options ('Open', 'Not assigned').

## Bugs and Code Smells

**SonarQube** Projects Issues Rules Quality Profiles Quality Gates Administration More Search

sonarqube14 / main

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

**Issues**

- Medium: 47k
- Low: 3

**Type**

- Bug: 47k
- Vulnerability: 0
- Code Smell: 164k

Add to selection Ctrl + click

**Scope**

**Status**

**Security Category**

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

Add "lang" and/or "xml:lang" attributes to this "<html>" element. Intentionality Reliability accessibility wcag2-a

Insert a <!DOCTYPE> declaration to before this <html> tag. Consistency Reliability user-experience

Add "<th>" headers to this "<table>". Intentionality

46,515 issues 1426d effort

**SonarQube** Projects Issues Rules Quality Profiles Quality Gates Administration More Search

sonarqube14 / main

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

**Issues**

- Medium: 143k
- Low: 21k

**Type**

- Bug: 47k
- Vulnerability: 0
- Code Smell: 164k

Add to selection Ctrl + click

**Scope**

**Status**

**Security Category**

**gameoflife-acceptance-tests/Dockerfile**

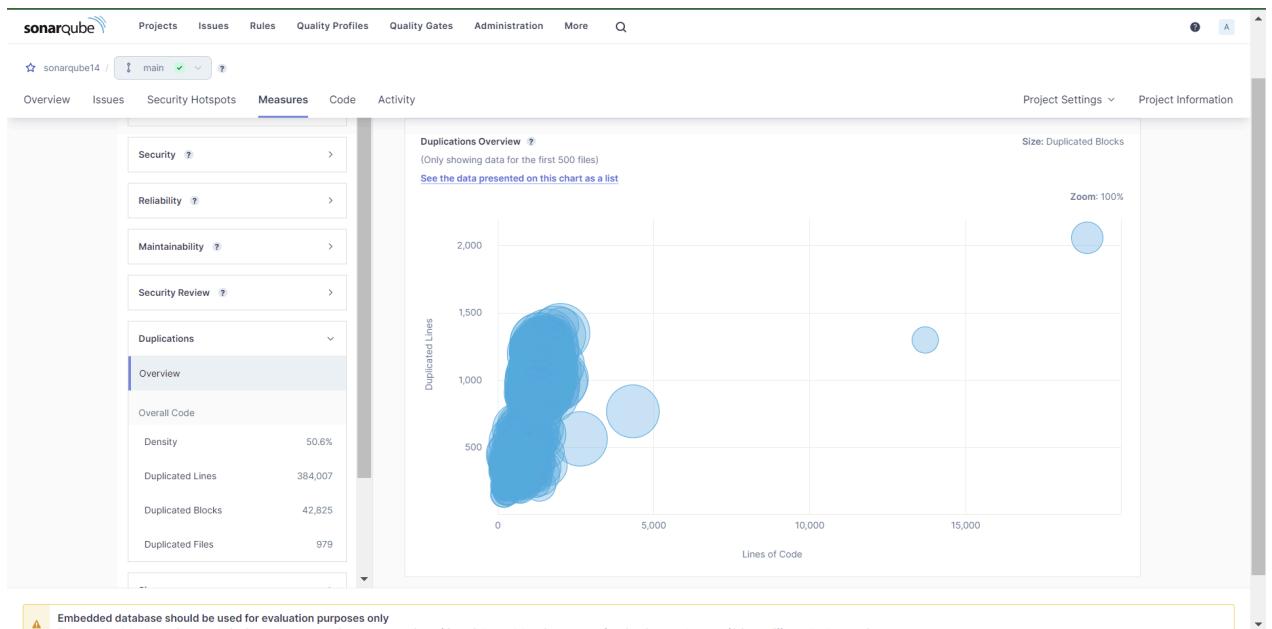
Use a specific version tag for the image. Intentionality Maintainability No tags

Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. Intentionality Maintainability No tags

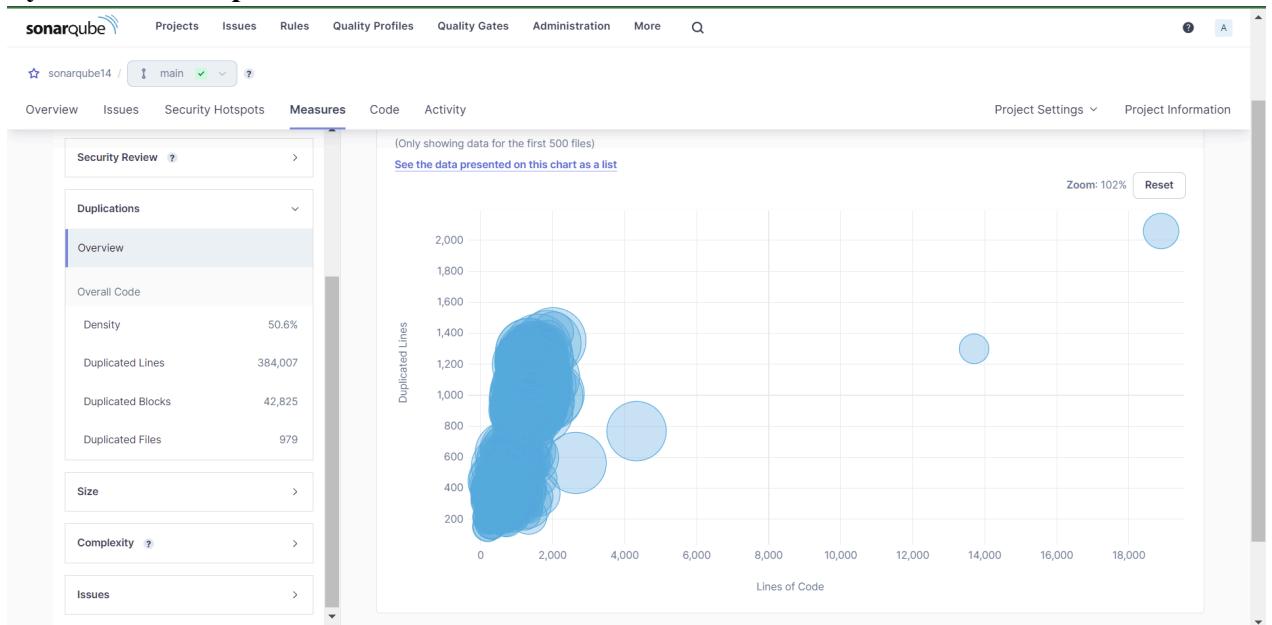
Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. Intentionality

164,034 issues 1708d effort

## Duplicates



## Cyclomatic Complexities



In this way, we have created a CI/CD Pipeline with Jenkins and integrated it with SonarQube to find issues in the code like bugs, code smells, duplicates, cyclomatic complexities, etc.

**Conclusion:** In this experiment, we successfully cloned a GitHub repository and integrated it with SonarQube for code analysis. During the analysis, SonarQube highlighted various types of program issues, including:

- **Consistency:** Code standards and formatting issues.
- **Intentionality:** Potential logical or structural errors in code.
- **Severity:** Classification of errors by criticality.
- **Duplicates:** Identification of duplicate code sections.
- **Cyclomatic Complexity:** Measurement of code complexity based on control flow.

These insights provided a comprehensive understanding of the code quality and highlighted areas that needed improvement.

### **Issues Faced:**

1. **SonarQube Scanner Path Error:** The Jenkins pipeline script was initially unable to run correctly due to Jenkins not detecting the correct path for the SonarQube Scanner. This required manual intervention to modify the script and set the proper path for the scanner bash file, allowing the pipeline to function as expected.

## Experiment No: 9

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

### 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

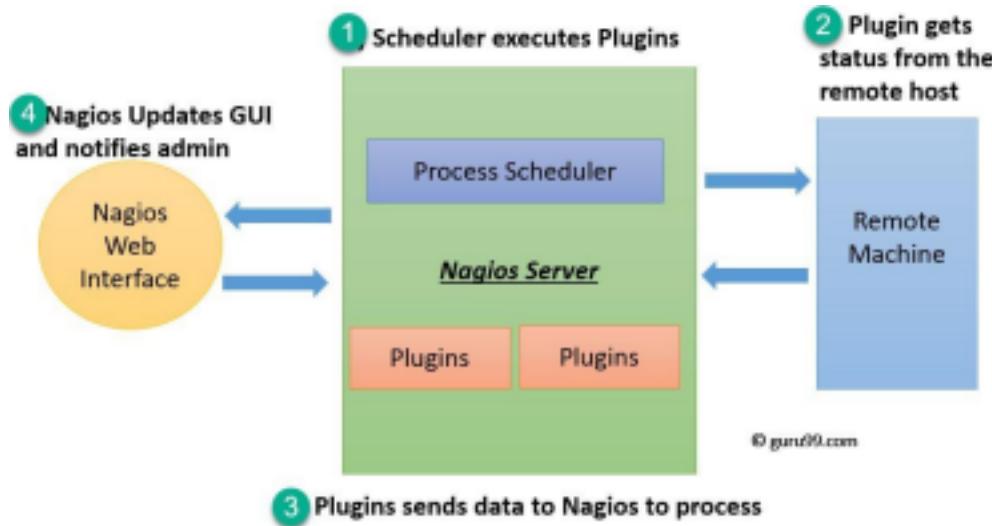
Following are the important features of Nagios monitoring tool:

- 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 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.

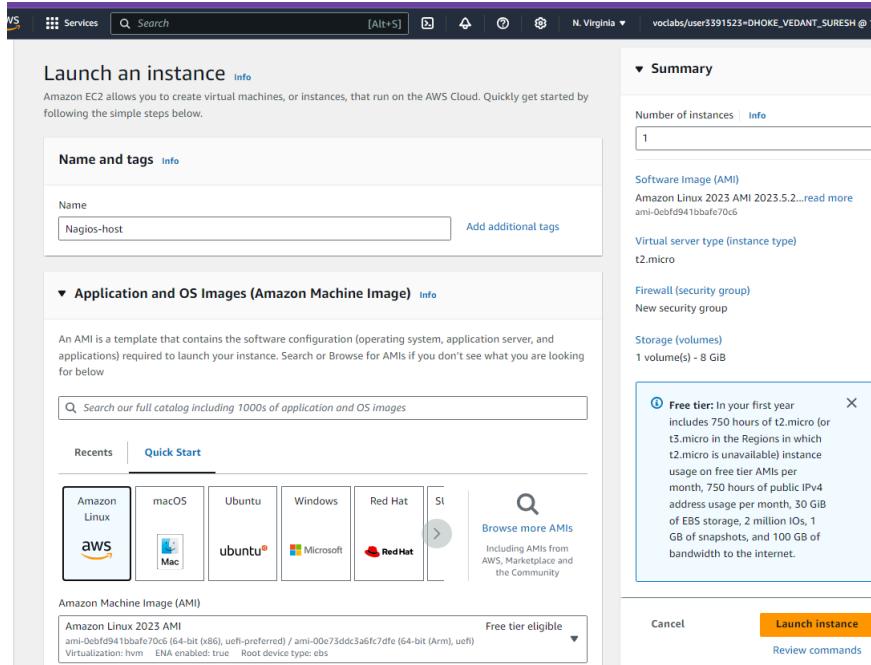
**Step 1:** Login to your AWS account Personal / Academy. Click on EC2 instance then click on Create Security Group. Give the name as Nagios and any description and add the following inbounds rules.

The screenshot shows the AWS VPC Inbound Rules configuration page. It lists eight inbound rules with the following details:

Type	Protocol	Port range	Source	Description - optional
HTTPS	TCP	443	Anywhere-IPv4	0.0.0.0/0
All traffic	All	All	Anywhere-IPv4	0.0.0.0/0
HTTP	TCP	80	Anywhere-IPv4	0.0.0.0/0
All ICMP - IPv6	IPv6 ICMP	All	Anywhere-IPv6	-/0
SSH	TCP	22	Anywhere-IPv4	0.0.0.0/0
All ICMP - IPv4	ICMP	All	Anywhere-IPv4	0.0.0.0/0
Custom TCP	TCP	5666	Anywhere-IPv4	0.0.0.0/0

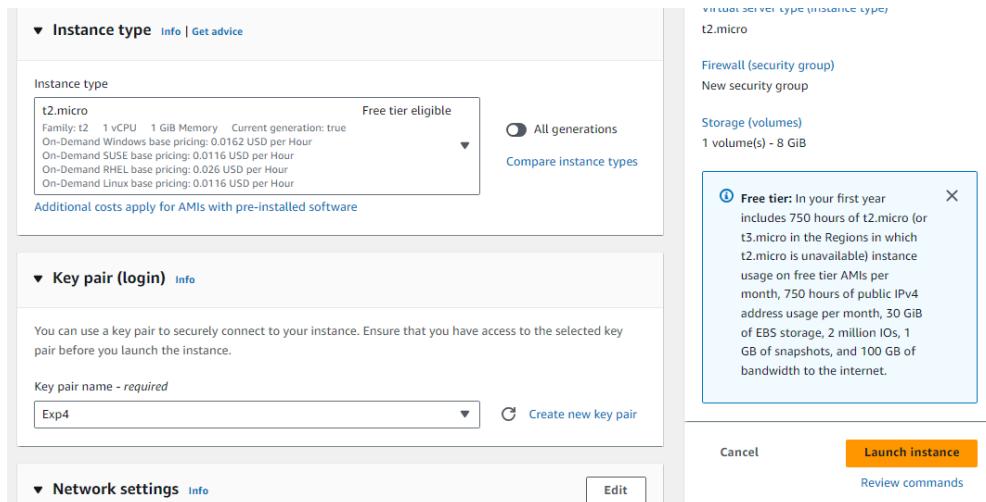
An "Add rule" button is located at the bottom left of the table.

**Step 2:** Now Create a new EC2 instance. Name: Nagios-host ,AMI: Amazon Linux, Instance Type: t2.micro.



**For Key pair :** Click on create key and make key of type RSA with extension .pem . Key will be downloaded to your local machine.

Now select that key in key pair if you already have key with type RSA and extension .pem no need to create new key but you must have that key downloaded.



Select the Existing Security Group and select the Security Group we have created in Step 1

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

[Edit](#)

Network | [Info](#)  
vpc-09b4fa6cf9c39cafb

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

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

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

Create security group     Select existing security group

Common security groups [Info](#)

Select security groups ▾

Nagios sg-0527f220d5b4a08fd X    VPC: vpc-09b4fa6cf9c39cafb

Compare security group rules

Security groups that you add or remove here will be added to or removed from all your network interfaces.

**Step 3:** Now After creating the EC2 Instance click on connect and then copy the command which is given as example in the SSH Client section .

[EC2](#) > [Instances](#) > [i-097ceff974c1f2078](#) > Connect to instance

**Connect to instance** [Info](#)

Connect to your instance i-097ceff974c1f2078 (Nagios-host) using any of these options

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

Instance ID  
 [i-097ceff974c1f2078 \(Nagios-host\)](#)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is Exp4.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.  
 chmod 400 "Exp4.pem"
4. Connect to your instance using its Public DNS:  
 [ec2-54-145-143-72.compute-1.amazonaws.com](#)

Example:  
 `ssh -i "Exp4.pem" ec2-user@ec2-54-145-143-72.compute-1.amazonaws.com`

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

Now open the terminal in the folder where your key(RSA key with .pem) is located.and paste that copied command.

Successfully connected to the instance.

```
C:\Users\LENOVO>ssh -i "Exp4.pem" ec2-user@ec2-54-210-4-52.compute-1.amazonaws.com
The authenticity of host 'ec2-54-210-4-52.compute-1.amazonaws.com (54.210.4.52)' can't be established.
ED25519 key fingerprint is SHA256:YqoGBku3mptyrGnwBwnKC060wMjlNnInQm5MPBeB1RM.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-210-4-52.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

          #
 ~\_ ####_      Amazon Linux 2023
 ~~ \#####\
 ~~  \###|
 ~~    \#/ --- https://aws.amazon.com/linux/amazon-linux-2023
 ~~     V~' '-->
 ~~~   /
 ~~~-.-
 ~-/-
 _/m/'[ec2-user@ip-172-31-38-62 ~]$ |
```

**Step 4:** Now Run the following command to make a new user.

**sudo adduser -m nagios**

**sudo passwd nagios**

```
[ec2-user@ip-172-31-38-62 ~]$ sudo adduser -m nagios
[ec2-user@ip-172-31-38-62 ~]$ sudo passwd nagios
Changing password for user nagios.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-38-62 ~]$ |
```

**Step 5:** Now Run the following command to make a new user group.

**sudo groupadd nagcmd**

**sudo usermod -a -G nagcmd nagios**

**sudo usermod -a -G nagcmd apache**

```
[ec2-user@ip-172-31-38-62 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-38-62 ~]$ sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache
[ec2-user@ip-172-31-38-62 ~]$ |
```

**Step 6:** Now make a new directory and go to that directory.

**mkdir ~/downloads**

**cd ~/downloads**

```
[ec2-user@ip-172-31-38-62 ~]$ mkdir ~/downloads
cd ~/downloads|
```

**Step 7:** Now to download the Nagios 4.5.5 and Nagios-plugins 2.4.11 run the following commands respectively.

**wget <https://go.nagios.org/l/975333/2024-09-17/6kqcx>**

```
[ec2-user@ip-172-31-38-62 downloads]$ wget https://go.nagios.org/l/975333/2024-09-17/6kqcx
--2024-10-02 07:05:23-- https://go.nagios.org/l/975333/2024-09-17/6kqcx
Resolving go.nagios.org (go.nagios.org)... 34.237.219.119, 3.92.120.28, 18.208.125.13, ...
Connecting to go.nagios.org (go.nagios.org)|34.237.219.119|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: http://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8 [following]
--2024-10-02 07:05:23-- http://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2600:3c0::f03c:92ff:fe7:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8 [following]
--2024-10-02 07:05:23-- https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: '6kqcx'

6kqcx          100%[=====] 1.97M 6.65MB/s   in 0.3s

2024-10-02 07:05:24 (6.65 MB/s) - '6kqcx' saved [2065473/2065473]

[ec2-user@ip-172-31-38-62 downloads]$ |
```

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

```
[ec2-user@ip-172-31-38-62 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
--2024-10-02 07:05:58-- https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2753049 (2.6M) [application/x-gzip]
Saving to: 'nagios-plugins-2.4.11.tar.gz'

nagios-plugins-2.4.11.t 100%[=====] 2.62M 7.32MB/s   in 0.4s

2024-10-02 07:05:59 (7.32 MB/s) - 'nagios-plugins-2.4.11.tar.gz' saved [2753049/2753049]

[ec2-user@ip-172-31-38-62 downloads]$ |
```

**Step 8:** Now to extract the files from the downloaded Nagios 4.5.5 run the following command.

**tar zxvf 6kqcx**

```
[ec2-user@ip-172-31-38-62 downloads]$ tar zxvf 6kqcx
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
--- " / E / T U A N U C
```

**Step 9:** Now change the directory to nagios-4.5.5 (Or which version you have downloaded)

```
[ec2-user@ip-172-31-38-62 downloads]$ cd nagios-4.5.5
```

**Step 10:** Now run the following command to configure.

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

```
[ec2-user@ip-172-31-38-62 nagios-4.5.5]$ ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether the compiler supports GNU C... yes
checking whether gcc accepts -g... yes
checking for gcc option to enable C11 features... none needed
checking whether make sets $(MAKE)... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking for sys/wait.h that is POSIX.1 compatible... yes
checking for stdio.h... yes
checking for stdlib.h... yes
checking for string.h... yes
checking for inttypes.h... yes
```

At the end we have found the error of cannot find ssl header .

```
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-172-31-38-62 nagios-4.5.5]$ |
```

So run following command to install ssl.

```
sudo yum install openssl-devel
```

```
[ec2-user@ip-172-31-38-62 nagios-4.5.5]$ sudo yum install openssl-devel
Last metadata expiration check: 0:10:57 ago on Wed Oct 2 06:57:40 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          21 MB/s | 3.0 MB  00:00
-----
Total                                         16 MB/s | 3.0 MB  00:00

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

Installed:
openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64

Complete!
[ec2-user@ip-172-31-38-62 nagios-4.5.5]$ |
```

Now rerun the command **./configure --with-command-group=nagcmd**

```
[ec2-user@ip-172-31-38-62 nagios-4.5.5]$ ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether the compiler supports GNU C... yes
checking whether gcc accepts -g... yes
checking for gcc option to enable C11 features... none needed
checking whether make sets $(MAKE)... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking for sys/wait.h that is POSIX.1 compatible... yes
checking for stdio.h... yes
```

```
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.

[ec2-user@ip-172-31-38-62 nagios-4.5.5]$ |
```

**Step 11:** Now run the following commands to setup the Nagios.

**sudo make install**

```
[ec2-user@ip-172-31-38-62 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 nagcmd -d /usr/local/nagios/var/spool/checkresults
chmod g+s /usr/local/nagios/var/spool/checkresults
```

\*\*\* Main program, CGIs and HTML files installed \*\*\*

You can continue with installing Nagios as follows (type 'make' without any arguments for a list of all possible options):

```
make install-init
- This installs the init script in /lib/systemd/system

make install-commandmode
- This installs and configures permissions on the
  directory for holding the external command file

make install-config
- This installs sample config files in /usr/local/nagios/etc

make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5'
[ec2-user@ip-172-31-38-62 nagios-4.5.5]$ |
```

**sudo make install-init**

```
[ec2-user@ip-172-31-38-62 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
[ec2-user@ip-172-31-38-62 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 ***
```

**sudo make install-webconf**

```
[ec2-user@ip-172-31-38-62 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-172-31-38-62 nagios-4.5.5]$ |
```

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

```
[ec2-user@ip-172-31-38-62 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-172-31-38-62 nagios-4.5.5]$ |
```

Now to restart the httpd service run the following command.

**sudo service httpd restart**

```
[ec2-user@ip-172-31-38-62 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
```

**Step 12:** Now to extract the files from the downloaded Nagios plugin 2.4.11 run the following command first change the directory.

**cd ~/downloads**

**tar zxvf nagios-plugins-2.4.11.tar.gz**

```
[ec2-user@ip-172-31-38-62 nagios-4.5.5]$ cd ~/downloads
[ec2-user@ip-172-31-38-62 downloads]$ tar zxvf nagios-plugins-2.4.11.tar.gz
nagios-plugins-2.4.11/
nagios-plugins-2.4.11/build-aux/
nagios-plugins-2.4.11/build-aux/compile
nagios-plugins-2.4.11/build-aux/config.guess
nagios-plugins-2.4.11/build-aux/config.rpath
nagios-plugins-2.4.11/build-aux/config.sub
nagios-plugins-2.4.11/build-aux/install-sh
nagios-plugins-2.4.11/build-aux/ltmain.sh
nagios-plugins-2.4.11/build-aux/missing
nagios-plugins-2.4.11/build-aux/mkinstalldirs
nagios-plugins-2.4.11/build-aux/depcomp
nagios-plugins-2.4.11/build-aux/snippet/
```

**Step 13:** Now change the directory to nagios-plugins-2.4.11 and run the config command to configure.  
**cd nagios-plugins-2.4.11**

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

```
[ec2-user@ip-172-31-38-62 nagios-plugins-2.4.11]$ ./configure --with-nagios-user=nagios --with-nagios-group=nagios
checking for a BSD-compatible install... /usr/bin/install -
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether to enable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
-----
```

**Step 14:** Run the following commands to check nagios and start it.

**sudo chkconfig --add nagios**

```
[ec2-user@ip-172-31-38-62 nagios-plugins-2.4.11]$ sudo chkconfig --add nagios
error reading information on service nagios: No such file or directory
[ec2-user@ip-172-31-38-62 nagios-plugins-2.4.11]$ |
```

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

```
[ec2-user@ip-172-31-38-62 nagios-plugins-2.4.11]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

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

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

Running pre-flight check on configuration data...

Checking objects...
  Checked 8 services.
  Checked 1 hosts.
  Checked 1 host groups.
  Checked 0 service groups.
  Checked 1 contacts.
  Checked 1 contact groups.
  Checked 24 commands.
  Checked 5 time periods.
  Checked 0 host escalations.
  Checked 0 service escalations.
Checking for circular paths...
  Checked 1 hosts
  Checked 0 service dependencies
  Checked 0 host dependencies
  Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors: 0

Things look okay - No serious problems were detected during the pre-flight check
[ec2-user@ip-172-31-38-62 nagios-plugins-2.4.11]$ |
```

**cd****sudo service nagios start**

```
[ec2-user@ip-172-31-38-62 nagios-plugins-2.4.11]$ cd
[ec2-user@ip-172-31-38-62 ~]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-38-62 ~]$ |
```

**sudo systemctl status nagios**

```
[ec2-user@ip-172-31-38-62 ~]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
   Loaded: loaded (/usr/lib/systemd/system/nagios.service; disabled; preset: disabled)
     Active: active (running) since Wed 2024-10-02 07:41:50 UTC; 5s ago
       Docs: https://www.nagios.org/documentation
   Process: 63762 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (c>
   Process: 63763 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code>
 Main PID: 63764 (nagios)
   Tasks: 6 (limit: 1112)
  Memory: 5.4M
    CPU: 74ms
   CGroup: /system.slice/nagios.service
           ├─63764 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
           ├─63765 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─63766 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─63767 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─63768 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           └─63769 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 02 07:41:50 ip-172-31-38-62.ec2.internal nagios[63764]: qh: core query handler registered
Oct 02 07:41:50 ip-172-31-38-62.ec2.internal nagios[63764]: qh: echo service query handler registered
Oct 02 07:41:50 ip-172-31-38-62.ec2.internal nagios[63764]: qh: help for the query handler registered
Oct 02 07:41:50 ip-172-31-38-62.ec2.internal nagios[63764]: wproc: Successfully registered manager
Oct 02 07:41:50 ip-172-31-38-62.ec2.internal nagios[63764]: wproc: Registry request: name=Core Work
Oct 02 07:41:50 ip-172-31-38-62.ec2.internal nagios[63764]: wproc: Registry request: name=Core Work
Oct 02 07:41:50 ip-172-31-38-62.ec2.internal nagios[63764]: wproc: Registry request: name=Core Work
Oct 02 07:41:50 ip-172-31-38-62.ec2.internal nagios[63764]: wproc: Registry request: name=Core Work
Oct 02 07:41:50 ip-172-31-38-62.ec2.internal nagios[63764]: wproc: Registry request: name=Core Work
Oct 02 07:41:51 ip-172-31-38-62.ec2.internal nagios[63764]: Successfully launched command file work
Oct 02 07:41:51 ip-172-31-38-62.ec2.internal nagios[63764]: HOST ALERT: localhost;DOWN;SOFT;1;(No op
Oct 02 07:41:51 ip-172-31-38-62.ec2.internal nagios[63764]: HOST ALERT: localhost;DOWN;SOFT;1;(No op
[lines 1-28/28 (END)]
```

**Step 15:** We can see we have successfully launched the Nagios now . Open <http://<instance public ip>/nagios/> here it is <http://54.210.4.52/nagios> we can see the running web page of nagios.

The screenshot shows the Nagios Core 4.5.5 web interface. The left sidebar contains navigation links for General, Current Status, Problems, Reports, and System. The main content area features the Nagios logo and a message indicating the daemon is running with PID 63764. It includes sections for 'Get Started' (with a bulleted list of steps), 'Latest News' (empty), 'Don't Miss...' (empty), and 'Quick Links' (with links to Nagios Library, Labs, Exchange, Support, and other resources). The bottom of the page includes copyright information and a 'Page Tour' link.

## Conclusion:

In this experiment, we successfully installed and configured Nagios Core, Nagios Plugins, and NRPE on a Linux machine within an AWS EC2 instance. The aim of continuously monitoring a remote system

was achieved by integrating Nagios with the EC2 environment and allowing web access via the Nagios dashboard. We faced several challenges that required troubleshooting:

- **Security Group Configuration:** Setting up the correct inbound rules in the AWS security group was essential but prone to mistakes. Incorrectly configured ports could block HTTP and NRPE communication, preventing access to the Nagios dashboard or monitoring checks.
- **User and Group Permissions:** There were some issues when configuring user and group permissions, especially while adding users to the nagcmd group. If the commands weren't run correctly, Nagios failed to run properly due to incorrect access rights.
- **Dependencies and Package Installation:** While installing Nagios and its plugins, we encountered dependency issues, particularly with OpenSSL. Missing packages or libraries often halted the configuration process. Resolving these involved installing required dependencies and restarting the configuration steps.

## Experiment No: 10

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

### Theory:

#### Port and Service Monitoring

Port and service monitoring in Nagios involves checking the availability and responsiveness of network services running on specific ports. This ensures that critical services (like HTTP, FTP, or SSH) are operational. Nagios uses plugins to ping the ports and verify whether services are up and responding as expected, allowing administrators to be alerted in case of outages.

#### Windows/Linux Server Monitoring

Windows/Linux server monitoring with Nagios entails tracking the performance and health of servers running these operating systems. It includes monitoring metrics such as CPU usage, memory consumption, disk space, and system logs. Nagios employs various plugins to gather data, enabling administrators to ensure optimal performance, identify potential issues, and maintain uptime across their server infrastructure.

#### Prerequisites:

AWS Academy or Personal account.

Nagios Server running on Amazon Linux Machine. (Refer Experiment No 9)

#### 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).

**sudo systemctl status nagios**

```

● nagios.service - Nagios Core 4.5.5
   Loaded: loaded (/usr/lib/systemd/system/nagios.service; disabled; preset: disabled)
   Active: active (running) since Sun 2024-10-06 10:58:43 UTC; 4s ago
     Docs: https://www.nagios.org/documentation
   Process: 62217 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Process: 62218 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
 Main PID: 62219 (nagios)
   Tasks: 6 (limit: 1112)
  Memory: 5.4M
    CPU: 74ms
   CGroup: /system.slice/nagios.service
           └─62219 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─62220 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─62221 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─62222 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─62223 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─62224 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 06 10:58:43 ip-172-31-46-196.ec2.internal nagios[62219]: qh: core query handler registered
Oct 06 10:58:43 ip-172-31-46-196.ec2.internal nagios[62219]: qh: echo service query handler registered
Oct 06 10:58:43 ip-172-31-46-196.ec2.internal nagios[62219]: qh: help for the query handler registered
Oct 06 10:58:43 ip-172-31-46-196.ec2.internal nagios[62219]: wproc: Successfully registered manager as @wproc with query handler
Oct 06 10:58:43 ip-172-31-46-196.ec2.internal nagios[62219]: wproc: Registry request: name=Core Worker 62223;pid=62223
Oct 06 10:58:43 ip-172-31-46-196.ec2.internal nagios[62219]: wproc: Registry request: name=Core Worker 62221;pid=62221
Oct 06 10:58:43 ip-172-31-46-196.ec2.internal nagios[62219]: wproc: Registry request: name=Core Worker 62222;pid=62222
Oct 06 10:58:43 ip-172-31-46-196.ec2.internal nagios[62219]: wproc: Registry request: name=Core Worker 62220;pid=62220
Oct 06 10:58:43 ip-172-31-46-196.ec2.internal nagios[62219]: Successfully launched command file worker with pid 62224
Oct 06 10:58:43 ip-172-31-46-196.ec2.internal nagios[62219]: HOST ALERT: localhost;DOWN;SOFT;1;(No output on stdout) stderr: execvp(/usr/local/nagios/libexec/nagios/cgi-bin/nagios?<*>
~

```

You can now proceed if you get the above message/output.

**Step 2:** Now Create a new EC2 instance. Name: Nagios-client, AMI: Ubuntu Instance Type: t2.micro.

The screenshot shows the AWS CloudFormation console with the following details:

- Name and tags:** The instance is named "Nagios-client".
- Application and OS Images (Amazon Machine Image):** The AMI type is set to "Ubuntu". Other options like Amazon Linux, macOS, Windows, Red Hat, and SUSE Linux are also listed.
- Outputs:** A placeholder for outputs is shown, with a note: "Including AMIs from AWS, Marketplace and the Community".

**For Key pair :** Click on create key and make key of type RSA with extension .pem . Key will be downloaded to your local machine.

The screenshot shows the 'Key pair (login)' configuration step in the AWS EC2 instance creation process. It includes a dropdown menu for selecting a key pair named 'devops' and a link to 'Create new key pair'.

Select the Existing Security Group and select the Security Group that we have created in Experiment no 9 or the same one you have used for the Nagios server (Nagios-host).

The screenshot shows the 'Network settings' configuration step. It displays network details like 'vpc-09b4fa6cf9c39caf' and subnet information. Under 'Firewall (security groups)', the 'Select existing security group' option is selected, and the 'Nagios sg-0527f220d5b4a08fd' security group is listed.

**Step 3:** Now After creating the EC2 Instance click on connect and then copy the command which is given as example in the SSH Client section .

Now open the terminal in the folder where your key(RSA key with .pem) is located. and paste that copied command.Successfully connected to the instance.

```
PS C:\Users\Vedant> ssh -i "devops.pem" ubuntu@ec2-3-81-218-241.compute-1.amazonaws.com
The authenticity of host 'ec2-3-81-218-241.compute-1.amazonaws.com (3.81.218.241)' can't be established.
ED25519 key fingerprint is SHA256:7YtdUbwFY6vK575h5DIfKqnloF220VC34blksm0Qcw.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-81-218-241.compute-1.amazonaws.com' (ED25519) 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 Fri Sep 27 08:38:26 UTC 2024

System load: 1.36 Processes: 26
Usage of /home: unknown Users Logged in: 0
Memory usage: 4% IPv4 address for eth0: 10.10.10.2
Swap usage: 0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

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

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

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

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

## Now perform all the commands on the Nagios-host till step 10

**Step 4:** Now on the server Nagios-host run the following command.

**ps -ef | grep nagios**

```
[ec2-user@ip-172-31-46-196 ~]$ ps -ef | grep nagios
nagios      2428      1  0 11:05 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagio
/etc/nagios.cfg
nagios      2430     2428  0 11:05 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local
nagios/rw/nagios.qh
nagios      2431     2428  0 11:05 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local
nagios/rw/nagios.qh
nagios      2432     2428  0 11:05 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local
nagios/rw/nagios.qh
nagios      2433     2428  0 11:05 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local
nagios/rw/nagios.qh
nagios      2437     2428  0 11:05 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagio
/etc/nagios.cfg
ec2-user    3367     3273  0 11:16 pts/0    00:00:00 grep --color=auto nagios
[ec2-user@ip-172-31-46-196 ~]$ |
```

**Step 5:** Now Become root user and create root directories.

**sudo su**

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

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

```
[ec2-user@ip-172-31-46-196 ~]$ sudo su
[root@ip-172-31-46-196 ec2-user]# mkdir /usr/local/nagios/etc/objects/monitorhosts
[root@ip-172-31-46-196 ec2-user]# mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-46-196 ec2-user]# |
```

**6:** Copy the sample localhost.cfg to linuxhost.cfg by running the following command.(Below command should come in one line see screenshot below)

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

```
[root@ip-172-31-46-196 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg  
[root@ip-172-31-46-196 ec2-user]# |
```

**Step 7:**Open linuxserver.cfg using nano and make the following changes in all positions?everywhere in file.

Change **hostname** to **linuxserver**.

Change **address** to the public IP of your Linux client.

Set **hostgroup\_name** to **linux-servers1**.

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

The screenshot shows a terminal window with two tabs. The active tab is titled 'GNU nano 5.8 /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg Modified'. The file content is a Nagios configuration snippet defining a host and a hostgroup.

```
#####
# 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 t>
                  ; in (or inherited by) the linux-server host templ>
    host_name    linux-server
    alias        localhost
    address      3.81.218.241
}

#####
# HOST GROUP DEFINITION
#
#####

# 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>
}
```

At the bottom of the terminal window, there is a menu bar with various keyboard shortcuts:

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

**Step 8:** Now update the Nagios config file .Add the following line in the file.

**Line to add : cfg\_dir=/usr/local/nagios/etc/objects/monitorhosts/**

Run the command : **nano /usr/local/nagios/etc/nagios.cfg**

```

GNU nano 5.8                               /usr/local/nagios/etc/nagios.cfg                         Modified
#####
# NAGIOS.CFG - Sample Main Config File for Nagios 4.5.5
#
# Read the documentation for more information on this configuration
# file. I've provided some comments here, but things may not be so
# clear without further explanation.
#
#####
#####

# LOG FILE
# This is the main log file where service and host events are logged
# for historical purposes. This should be the first option specified
# in the config file!!!
log_file=/usr/local/nagios/var/nagios.log

# 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.

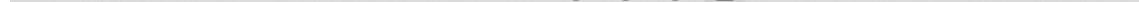
# You can specify individual object config files as shown below:
cfg_file=/usr/local/nagios/etc/objects/commands.cfg
cfg_file=/usr/local/nagios/etc/objects/contacts.cfg
cfg_file=/usr/local/nagios/etc/objects/timeperiods.cfg
cfg_file=/usr/local/nagios/etc/objects/templates.cfg
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/
# Definitions for monitoring the local (Linux) host
cfg_file=/usr/local/nagios/etc/objects/localhost.cfg

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

# Definitions for monitoring a router/switch

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

```



**Step 9:** Now Verify the configuration files by running the following commands.

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

```
[root@ip-172-31-46-196 ec2-user]# /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

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

Website: https://www.nagios.org
Reading configuration data...
  Read main config file okay...
Warning: Duplicate definition found for service 'HTTP' on host 'localhost' (config file '/usr/local/nagios.cfg', starting on line 152)
Warning: Duplicate definition found for service 'SSH' on host 'localhost' (config file '/usr/local/nagios.cfg', starting on line 138)
Warning: Duplicate definition found for service 'Swap Usage' on host 'localhost' (config file '/usr/local/nagios.cfg', starting on line 125)
Warning: Duplicate definition found for service 'Current Load' on host 'localhost' (config file '/usr/local/nagios.cfg', starting on line 112)
Warning: Duplicate definition found for service 'Total Processes' on host 'localhost' (config file '/usr/local/nagios.cfg', starting on line 100)
Warning: Duplicate definition found for service 'Current Users' on host 'localhost' (config file '/usr/local/nagios.cfg', starting on line 86)
Warning: Duplicate definition found for service 'Root Partition' on host 'localhost' (config file '/usr/local/nagios.cfg', starting on line 72)
Warning: Duplicate definition found for service 'PING' on host 'localhost' (config file '/usr/local/nagios.cfg', starting on line 58)
  Read object config files okay...

Running pre-flight check on configuration data...

Checking objects...
  Checked 8 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

[root@ip-172-31-46-196 ec2-user]#
```

**Step 10:** Now restart the services of nagios by running the following command.

**service nagios restart**

```
[root@ip-172-31-46-196 ec2-user]# service nagios restart
Redirecting to /bin/systemctl restart nagios.service
[root@ip-172-31-46-196 ec2-user]# |
```

**Step 11:** Now Go to the Nagios-client ssh terminal and update and install the packages by running the following command.

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

```
ubuntu@ip-172-31-37-150:~$ sudo apt update -y  
sudo apt install gcc -y  
sudo apt install -y nagios-nrpe-server nagios-plugins  
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease  
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]  
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]  
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]  
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]  
Get:6 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [382 kB]  
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]  
Get:8 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [83.9 kB]  
Get:9 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4704 B]  
Get:10 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [277 kB]  
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]  
Get:12 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [117 kB]  
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]  
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]  
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]  
Get:16 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]  
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]  
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]  
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [537 kB]  
Get:20 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.4 kB]  
Get:21 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [10.9 kB]  
  
Setting up python3-ldb (2:2.8.0+samba4.19.5+dfsg-4ubuntu9) ...  
Setting up samba-dsdb-modules:amd64 (2:4.19.5+dfsg-4ubuntu9) ...  
Setting up lib smbclient0:amd64 (2:4.19.5+dfsg-4ubuntu9) ...  
Setting up libcups2t64:amd64 (2.4.7-1.2ubuntu7.3) ...  
Setting up python3-samba (2:4.19.5+dfsg-4ubuntu9) ...  
Setting up smbclient (2:4.19.5+dfsg-4ubuntu9) ...  
Setting up samba-common-bin (2:4.19.5+dfsg-4ubuntu9) ...  
Processing triggers for man-db (2.12.0-4build2) ...  
Processing triggers for libc-bin (2.39-0ubuntu8.3) ...  
Scanning processes...  
Scanning linux images...  
  
Running kernel seems to be up-to-date.  
  
No services need to be restarted.  
  
No containers need to be restarted.  
  
No user sessions are running outdated binaries.  
  
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
ubuntu@ip-172-31-37-150:~$ |
```

**Step 12:** Open nrpe.cfg file to make changes.Under allowed\_hosts, add your nagios host IP address.

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

```
GNU nano 2.2                               /etc/nagios/nrpe.cfg *
```

```
# NRPE USER
# This determines the effective user that the NRPE daemon should run as.
# You can either supply a username or a UID.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
nrpe_user=nagios

# NRPE GROUP
# This determines the effective group that the NRPE daemon should run as.
# You can either supply a group name or a GID.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
nrpe_group=nagios

# ALLOWED HOST ADDRESSES
# This is an optional comma-delimited list of IP address or hostnames
# that are allowed to talk to the NRPE daemon. Network addresses with a bit mask
# (i.e. 192.168.1.0/24) are also supported. Hostname wildcards are not currently
# supported.
#
# Note: The daemon only does rudimentary checking of the client's IP
# address. I would highly recommend adding entries in your /etc/hosts.allow
# file to allow only the specified host to connect to the port
# you are running this daemon on.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
allowed_hosts=127.0.0.1,::1,3.91.89.94|
```

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

**Step 13:** Now restart the NRPE server by this command.

**sudo systemctl restart nagios-nrpe-server**

```
ubuntu@ip-172-31-37-150:~$ sudo systemctl restart nagios-nrpe-server
ubuntu@ip-172-31-37-150:~$ |
```

**Step 14:** Now again check the status of Nagios by running this command on Nagios-host and also check httpd is active and run the command to active it.

**sudo systemctl status nagios**

```
[root@ip-172-31-46-196 ec2-user]# sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
   Loaded: loaded (/usr/lib/systemd/system/nagios.service; disabled; preset: disabled)
   Active: active (running) since Sun 2024-10-06 11:37:16 UTC; 9min ago
     Docs: https://www.nagios.org/documentation
 Process: 4481 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, st>
 Process: 4482 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, statu>
Main PID: 4488 (nagios)
   Tasks: 6 (limit: 1112)
  Memory: 4.1M
     CPU: 108ms
    CGroup: /system.slice/nagios.service
        └─4488 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
          ├─4489 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
          ├─4490 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
          ├─4491 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
          ├─4492 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
          └─4497 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 06 11:37:16 ip-172-31-46-196.ec2.internal nagios[4488]: HOST ALERT: linux-server;DOWN;SOFT;1;(No output on s>
Oct 06 11:38:16 ip-172-31-46-196.ec2.internal nagios[4488]: HOST ALERT: linux-server;DOWN;SOFT;2;(No output on s>
Oct 06 11:39:16 ip-172-31-46-196.ec2.internal nagios[4488]: HOST ALERT: linux-server;DOWN;SOFT;3;(No output on s>
Oct 06 11:40:16 ip-172-31-46-196.ec2.internal nagios[4488]: HOST ALERT: linux-server;DOWN;SOFT;4;(No output on s>
Oct 06 11:41:16 ip-172-31-46-196.ec2.internal nagios[4488]: HOST ALERT: linux-server;DOWN;SOFT;5;(No output on s>
Oct 06 11:42:16 ip-172-31-46-196.ec2.internal nagios[4488]: HOST ALERT: linux-server;DOWN;SOFT;6;(No output on s>
Oct 06 11:43:16 ip-172-31-46-196.ec2.internal nagios[4488]: HOST ALERT: linux-server;DOWN;SOFT;7;(No output on s>
Oct 06 11:44:16 ip-172-31-46-196.ec2.internal nagios[4488]: HOST ALERT: linux-server;DOWN;SOFT;8;(No output on s>
Oct 06 11:45:16 ip-172-31-46-196.ec2.internal nagios[4488]: HOST ALERT: linux-server;DOWN;SOFT;9;(No output on s>
Oct 06 11:46:16 ip-172-31-46-196.ec2.internal nagios[4488]: HOST ALERT: linux-server;DOWN;HARD;10;(No output on s>
lines 1-28/28 (END)
```

**sudo systemctl status httpd**

**sudo systemctl start httpd**

**sudo systemctl enable httpd**

```
[root@ip-172-31-46-196 ec2-user]# sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
   Drop-In: /usr/lib/systemd/system/httpd.service.d
             └─php-fpm.conf
   Active: active (running) since Sun 2024-10-06 11:08:08 UTC; 42min ago
     Docs: man:httpd.service(8)
 Main PID: 2546 (httpd)
   Status: "Total requests: 48; Idle/Busy workers 100/0;Requests/sec: 0.0188; Bytes served/sec: 121 B/sec"
   Tasks: 230 (limit: 1112)
  Memory: 25.1M
     CPU: 1.834s
    CGroup: /system.slice/httpd.service
            ├─2546 /usr/sbin/httpd -DFOREGROUND
            ├─2548 /usr/sbin/httpd -DFOREGROUND
            ├─2554 /usr/sbin/httpd -DFOREGROUND
            ├─2555 /usr/sbin/httpd -DFOREGROUND
            ├─2556 /usr/sbin/httpd -DFOREGROUND
            └─2889 /usr/sbin/httpd -DFOREGROUND

Oct 06 11:08:07 ip-172-31-46-196.ec2.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...
Oct 06 11:08:08 ip-172-31-46-196.ec2.internal systemd[1]: Started httpd.service - The Apache HTTP Server.
Oct 06 11:08:08 ip-172-31-46-196.ec2.internal httpd[2546]: Server configured, listening on: port 80
[root@ip-172-31-46-196 ec2-user]# sudo systemctl start httpd
[root@ip-172-31-46-196 ec2-user]# sudo systemctl enable httpd
[root@ip-172-31-46-196 ec2-user]# |
```

**Step 15:** Now to check Nagios dashboard go to <http://<Nagios-host ip>/nagios> .

**Now Click on Hosts from left side panel**



**Current Network Status**

Last Updated: Sun Oct 6 12:06:05 UTC 2024  
Updated every 90 seconds  
Nagios® Core™ 4.5.5 - www.nagios.org  
Logged in as nagiosadmin

**Host Status Totals**

Up	Down	Unreachable	Pending
2	0	0	0

All Problems All Types

**Service Status Totals**

Ok	Warning	Unknown	Critical	Pending
6	1	0	1	0

All Problems All Types

**Host Status Details For All Host Groups**

Host	Status	Last Check	Duration	Status Information
linux-server	UP	10-06-2024 12:01:16	0d 0h 4m 49s	PING OK - Packet loss = 0%, RTA = 1.00 ms
localhost	UP	10-06-2024 12:02:49	0d 0h 7m 1s	PING OK - Packet loss = 0%, RTA = 0.03 ms

Results 1 - 2 of 2 Matching Hosts

**General**

- Home
- Documentation

**Current Status**

- Tactical Overview
- Map
- Hosts
- Services
- Host Groups
  - Summary
  - Grid
- Service Groups
  - Summary
  - Grid
- Problems
  - Services (Unhandled)
  - Hosts (Unhandled)
  - Network Outages

Quick Search:

**Reports**

- Availability
- Trends
- Alerts
- History

We can see our linuxserver now click on it we can see the host information.

**Host Information**

Last Updated: Sun Oct 6 12:07:06 UTC 2024  
Updated every 90 seconds  
Nagios® Core™ 4.5.5 - www.nagios.org  
Logged in as nagiosadmin

**Host**  
**localhost**  
(linux-server)

**Member of**  
**No hostgroups**

3.81.218.241

**Host State Information**

<b>Host Status:</b>	<b>UP</b> (for 0d 0h 5m 50s)
<b>Status Information:</b>	PING OK - Packet loss = 0%, RTA = 0.72 ms
<b>Performance Data:</b>	rts=0.721000ms;3000.000000;5000.000000;0.000000 pl=0%;80;100;0
<b>Current Attempt:</b>	1/10 (HARD state)
<b>Last Check Time:</b>	10-06-2024 12:06:16
<b>Check Type:</b>	ACTIVE
<b>Check Latency / Duration:</b>	0.000 / 4.199 seconds
<b>Next Scheduled Active Check:</b>	10-06-2024 12:11:16
<b>Last State Change:</b>	10-06-2024 12:01:16
<b>Last Notification:</b>	N/A (notification 0)
<b>Is This Host Flapping?</b>	<b>NO</b> (10.66% state change)
<b>In Scheduled Downtime?</b>	<b>NO</b>
<b>Last Update:</b>	10-06-2024 12:07:05 (0d 0h 0m 1s ago)

**Host Commands**

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

**Host Comments**

Add a new comment Delete all comments

Entry Time Author Comment Comment ID Persistent Type Expires Actions

**General**

- Home
- Documentation

**Current Status**

- Tactical Overview
- Map
- Hosts
- Services
- Host Groups
  - Summary
  - Grid
- Service Groups
  - Summary
  - Grid
- Problems
  - Services (Unhandled)
  - Hosts (Unhandled)
  - Network Outages

Quick Search:

**Reports**

- Availability
- Trends
- Alerts
- History
- Summary
- Histogram
- Notifications
- Event Log

**System**

- Comments
- Downtime

## Current Network Status

**Current Network Status:**

- Last Updated: Sun Oct 6 12:09:17 UTC 2024
- Updated every 99 seconds
- Nagios® Core™ 4.5.5 - www.nagios.org
- Logged in as nagiosadmin

**Host Status Totals:**

Up	Down	Unreachable	Pending
2	0	0	0
All Problems	All Types		
0	2		

**Service Status Totals:**

Ok	Warning	Unknown	Critical	Pending
6	1	0	1	0
All Problems	All Types			
2	8			

**Service Status Details For All Hosts**

Entries sorted by host name (descending)

Host	Service	Status	Last Check	Duration	Attempt	Status Information
localhost	Current Load	OK	10-06-2024 12:06:34	0d 0h 7m 43s	1/4	OK - load average: 0.00, 0.02, 0.00
	Current Users	OK	10-06-2024 12:07:12	0d 0h 7m 5s	1/4	USERS OK - 3 users currently logged in
	HTTP	WARNING	10-06-2024 12:07:49	0d 0h 6m 28s	4/4	HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes in 0.000 second response time
	PING	OK	10-06-2024 12:08:27	0d 0h 5m 50s	1/4	PING OK - Packet loss = 0%, RTA = 0.02 ms
	Root Partition	OK	10-06-2024 12:09:04	0d 0h 10m 13s	1/4	DISK OK - free space: / 6122 MIB (75.43% inode=98%)
	SSH	OK	10-06-2024 12:04:42	0d 0h 9m 35s	1/4	SSH OK - OpenSSH 8.7 (protocol 2.0)
	Swap Usage	CRITICAL	10-06-2024 12:05:19	0d 0h 58m 58s	4/4	SWAP CRITICAL - 0% free (0 MB out of 0 MB) - Swap is either disabled, not present, or of zero size
	Total Processes	OK	10-06-2024 12:05:57	0d 0h 8m 20s	1/4	PROCS OK: 38 processes with STATE = RSZDT

Results 1 - 8 of 8 Matching Services

## Conclusion:

In this experiment, we successfully implemented port, service, and Windows/Linux server monitoring using Nagios, but encountered a few challenges.

- **Configuration Issues:** Setting up monitoring hosts and editing files like linuxserver.cfg led to some errors in file paths and syntax, which required careful review.
- **NRPE Setup:** Configuring NRPE for remote monitoring was tricky due to firewall and permission issues, often causing connectivity problems between the Nagios host and clients.
- **Service Restarts:** Restarting Nagios and NRPE to apply changes didn't always work smoothly, with misconfigurations requiring troubleshooting.
- **Dashboard Access:** Accessing the Nagios dashboard was hindered by incorrect AWS security group rules, needing adjustments to allow proper HTTP and TCP traffic.

## Experiment No: 11

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

### Theory:

#### AWS Lambda

A fully managed, serverless computing service where you run code without provisioning or managing servers. Lambda automatically scales your application based on the number of incoming requests or events, ensuring efficient resource utilization. You are only charged for the time your code is running, with no upfront cost, making it cost-effective for on-demand workloads.

#### Lambda Workflow

- **Create a Function:** Write the function code and define its handler (entry point). You can use the AWS Console, CLI, or upload a deployment package.
- **Set Event Sources:** Define how the function is triggered (e.g., when an object is uploaded to S3 or a DynamoDB table is updated).
- **Execution:** When triggered, Lambda runs your function, executes the logic, and automatically scales to handle the incoming event volume.
- **Scaling and Concurrency:** Lambda scales automatically by launching more instances of the function to handle simultaneous invocations. There are also options for configuring **reserved concurrency** to manage traffic.
- **Monitoring and Logging:** Lambda integrates with Amazon CloudWatch for logging and monitoring. Logs for each invocation are sent to CloudWatch, allowing you to track performance and troubleshoot errors.

#### AWS Lambda Functions

- **Python:** Great for quick development with its rich standard library and support for lightweight tasks.
- **Java:** Typically used for more complex, compute-intensive tasks. While it's robust, cold start times can be higher.
- **Node.js:** Excellent for I/O-bound tasks like handling APIs or streaming data, with fast startup times and efficient memory usage.

## Steps To create the lambda function:

**Step 1:** Login to your AWS Personal/Academy Account. Open lambda and click on create function button.

**Step 2:** Now Give a name to your Lambda function, 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.

The screenshot displays the AWS Lambda service interface. On the left, a sidebar navigation includes 'Dashboard', 'Applications', and 'Functions' (which is selected). Other sections like 'Additional resources' and 'Related AWS resources' are also visible. The main content area shows a table titled 'Functions (5)' with columns for 'Function name', 'Description', 'Package type', 'Runtime', and 'Last modified'. Three functions are listed: 'ModLabRole', 'RedshiftOverwatch', and 'RoleCreationFunction'. The 'Create function' button is located at the top right of the table. Below this, a modal window titled 'Create function' is open, showing three options: 'Author from scratch', 'Use a blueprint', and 'Container image'. The 'Author from scratch' option is selected. The 'Basic information' section contains fields for 'Function name' (set to 'MY-lambda'), 'Runtime' (set to 'Python 3.12'), and 'Architecture' (set to 'x86\_64'). The status bar at the bottom indicates the date and time as '10/4/2024'.

So See or Edit the basic settings go to configuration then click on edit general setting.

Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.

The screenshot shows the configuration page for a Lambda function. It includes sections for Memory (128 MB), Ephemeral storage (512 MB), SnapStart (None), and Timeout (1 sec). The Execution role is set to 'Use an existing role' with 'service-role/MY-lambda-role-en0dja4u' selected. A 'Create' button is visible, and at the bottom are 'Cancel' and 'Save' buttons.

**Step 3:** Now Click on the Test tab then select Create a new event, give a name to the event and select Event Sharing to private, and select hello-world template.

**Step 4:** Now In Code section select the created event from the dropdown of test then click on test . You will see the below output.

The screenshot shows the AWS Lambda Test event configuration interface. At the top, a green header bar indicates "Successfully updated the function MY-lambda." Below the header, there are tabs for "Code", "Test" (which is selected), "Monitor", "Configuration", "Aliases", and "Versions". The "Test event" section contains fields for "Event name" (set to "MY-Event"), "Event sharing settings" (set to "Private"), and a "Template - optional" dropdown (set to "hello-world"). A "Event JSON" section includes a "Format JSON" button. At the bottom, a success message states "The test event MY-Event was successfully saved." The overall interface is clean and modern, typical of AWS services.

Successfully updated the function MY-lambda.

Code | Test | Monitor | Configuration | Aliases | Versions

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

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

MY-Event

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

hello-world

**Event JSON**

Format JSON

The test event MY-Event was successfully saved.

Code | Test | Monitor | Configuration | Aliases | Versions

**Code source** [Info](#)

File Edit Find View Go Tools Window **Test** Deploy

Upload from ▾

Configure test event Ctrl-Shift-C

Private saved events

• MY-Event

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

**Step 5:** You can edit your lambda function code. I have changed the code to display the new String. Now ctrl+s to save and click on deploy to deploy the changes.

```

1 import json
2
3 def lambda_handler(event, context):
4     # TODO implement
5     new_string = "Helloooo ! How are you ? "
6     return {
7         'statusCode': 200,
8         'body': json.dumps(new_string)
9     }
10

```

**Step 6:** Now click on the test and observe the output. We can see the status code 200 and your string output and function logs. On successful deployment.

Execution results

Test Event Name  
MY-Event

Response

```
{
    "statusCode": 200,
    "body": "\"Helloooo ! How are you ? \""
}
```

Function Logs

```
START RequestId: a0d9a9be-5433-4073-8d7a-dd5824ff2788 Version: $LATEST
END RequestId: a0d9a9be-5433-4073-8d7a-dd5824ff2788
REPORT RequestId: a0d9a9be-5433-4073-8d7a-dd5824ff2788 Duration: 2.29 ms Billed Duration: 3 ms Memory Size: 128 MB Max
```

Request ID  
a0d9a9be-5433-4073-8d7a-dd5824ff2788

## Conclusion:

In this experiment, we successfully created and deployed our first AWS Lambda function using Python, gaining an understanding of its workflow and capabilities. The function was executed and tested, allowing us to observe the output and logs. While the overall process was smooth, there were several challenges that we encountered:

- **Role and Permissions Configuration:** Setting up the correct execution role with the necessary

permissions was critical, as misconfigurations could prevent the function from running or accessing other AWS services. Debugging permission issues was time-consuming, especially when using new roles.

- **Timeout Issues:** Initially, the default timeout setting was insufficient for some operations. Adjusting the timeout to a value that suits the function's workload was necessary, especially for more complex operations beyond basic tasks.
- **Event Testing:** Configuring and testing events within Lambda required careful attention. Choosing the wrong template or incorrect event parameters often led to failures during the test phase, requiring adjustments to the event settings.

**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

### Theory:

#### AWS Lambda and S3 Integration:

AWS Lambda allows you to execute code in response to various events, including those triggered by Amazon S3. When an object is added to an S3 bucket, it can trigger a Lambda function to execute, allowing for event-driven processing without managing servers.

### Workflow:

#### 1. Create an S3 Bucket:

- First, create an S3 bucket that will store the objects. This bucket will act as the trigger source for the Lambda function.

#### 2. Create the Lambda Function:

- Set up a new Lambda function using AWS Lambda’s console. You can choose a runtime environment like Python, Node.js, or Java.
- Write code that logs a message like “An Image has been added” when triggered.

#### 3. Set Up Permissions:

- Ensure that the Lambda function has the necessary permissions to access S3. You can do this by attaching an IAM role with policies that allow reading from the bucket and writing logs to CloudWatch.

#### 4. Configure S3 Trigger:

- Link the S3 bucket to the Lambda function by setting up a trigger. Specify that the function should be triggered when an object is created in the bucket (e.g., when an image is uploaded).

#### 5. Test the Setup:

- Upload an object (e.g., an image) to the S3 bucket to test the trigger. The Lambda function should execute and log the message “An Image has been added” in AWS CloudWatch Logs.

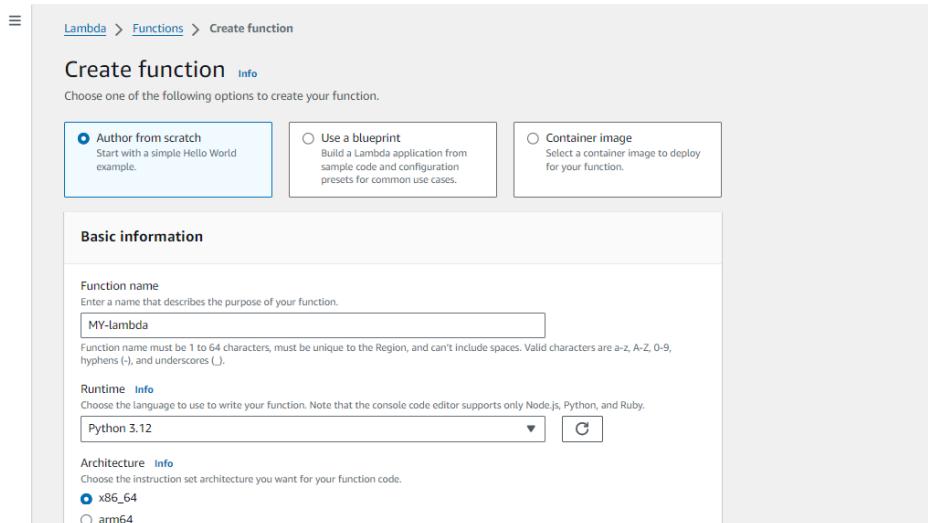
**Steps To create the lambda function:**

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

**Step 2:** I have used already created bucket v2bucket if you dont have then you can create a basic bucket for this experiment .

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

**Step 4:** Now Give a name to your Lambda function, 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.



The screenshot shows the AWS Lambda console. In the top navigation bar, 'Services' is selected. Below it, 'Elastic Beanstalk' is visible. A green success message at the top says: 'Successfully created the function MY-lambda. You can now change its code and configuration. To invoke your function with a test event, choose "Test".' The main area is titled 'MY-lambda'. It has tabs for 'Function overview' (selected) and 'Info'. Under 'Function overview', there are buttons for 'Throttle', 'Copy ARN', and 'Actions'. A 'Diagram' tab is selected, showing a single box labeled 'MY-lambda' with a 'Layers' section below it. Buttons for '+ Add trigger' and '+ Add destination' are also present. On the right, there's a 'Description' section with 'Last modified 3 seconds ago', 'Function ARN arn:aws:lambda:us-east-1:022499016110:function:MY-lambda', and a 'Function URL' link. A 'Tutorials' tab is open, showing a 'Create a simple web app' tutorial with a brief description and a 'Start tutorial' button.

The screenshot shows the 'Code source' tab for the MY-lambda function. The top navigation bar includes 'File', 'Edit', 'Find', 'View', 'Go', 'Tools', 'Window', 'Test' (selected), and 'Deploy'. The main area displays the code editor for 'lambda\_function.py'. The code is as follows:

```
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     }
```

So See or Edit the basic settings go to configuration then click on edit general setting.

The screenshot shows the AWS Lambda Configuration page. The left sidebar has a 'General configuration' section with links for Triggers, Permissions, Destinations, Function URL, Environment variables, Tags, VPC, and RDS databases. The main area is titled 'General configuration' with an 'Edit' button. It displays the following settings:

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

Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.

The screenshot shows the 'Edit basic settings' page for a Lambda function named 'Bhushan\_Lambda'. The top navigation bar includes the AWS logo, Services, a search bar, and a [Alt+S] key shortcut. The breadcrumb path is Lambda > Functions > Bhushan\_Lambda > Edit basic settings.

**Basic settings**

**Description - optional**  
Basic Settings

**Memory** 128 MB  
Your function is allocated CPU proportional to the memory configured. Set memory to between 128 MB and 10240 MB.

**Ephemeral storage** 512 MB  
You can configure up to 10 GB of ephemeral storage (/tmp) for your function. View pricing

**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  
Supported runtimes: Java 11, Java 17, Java 21.

**Execution role**  
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.

- Use an existing role
- Create a new role from AWS policy templates

**Step 5:** Now Click on the Test tab then select Create a new event, give a name to the event and select Event Sharing to private, and select s3 put 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-exp\_12

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
```

Template - optional

s3-put

Event JSON

```
{
  "Records": [
    {
      "eventversion": "2.0",
      "eventsource": "aws:s3",
      "awsregion": "us-east-1",
      "eventtime": "1970-01-01T00:00:00.000Z",
      "eventName": "ObjectCreated:Put",
      "userIdentity": {
        "principalId": "EXAMPLE"
      },
      "requestParameters": {
        "sourceIPAddress": "127.0.0.1"
      },
      "responseElements": {
        "x-amz-request-id": "EXAMPLE123456789",
        "x-amz-id-2": "EXAMPLE123/5678abcdefhijklambdaisawesome/mnopqrstuvwxyzABCDEFGH"
      },
      "s3": {
        "s3SchemaVersion": "1.0",
        "configurationId": "testConfigRule",
        "bucket": {
          "name": "example-bucket",
          "ownerIdentity": {
            "principalId": "EXAMPLE"
          }
        },
        "arn": "arn:aws:s3:::example-bucket"
      },
      "object": {
        "key": "testS3Key",
        "size": 1024,
        "eTag": "1234567890abcdef1234567890abcdef1234567890abcdef"
      }
    }
  ]
}
```

Format JSON

Learn how to implement common use cases in AWS Lambda.

Create a simple web app

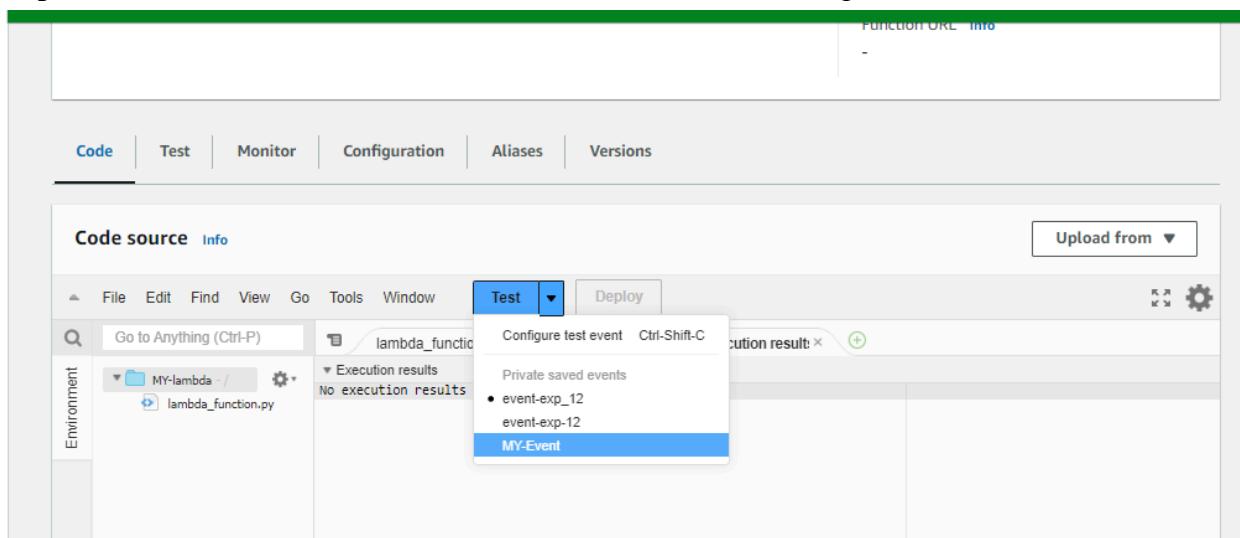
In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#)

[Start tutorial](#)

**Step 6:** Now In Code section select the created event from the dropdown .



**Step 7:** Now In the Lambda function click on add trigger.

Now select the source as S3 then select the bucket name from the dropdown, keep other things to default and also you can add prefix to

The screenshot shows the AWS Lambda Function Overview page for a function named 'MY-lambda'. The 'Diagram' tab is selected, displaying a single Lambda function icon. Below the diagram are two buttons: '+ Add trigger' and '+ Add destination'. To the right of the function name, there is a 'Description' section with a note about layers (0). Further down, the 'Last modified' time is shown as '23 minutes ago'. The 'Function ARN' is listed as 'arn:aws:lambda:us-east-1:022499016110:function:MY-lambda'. A 'Function URL' link is also present. At the bottom of the page, navigation tabs include 'Code', 'Test', 'Monitor', 'Configuration', 'Aliases', and 'Versions'.

The screenshot shows the 'Add trigger' configuration page for an S3 event source. The 'Trigger configuration' section is visible, showing the selected source as 'S3'. Under the 'Bucket' section, a dropdown menu shows 's3/v2buckets' as the chosen bucket, with 'Bucket region: us-east-1'. The 'Event types' section lists 'All object create events'. At the bottom, there is a note about 'Prefix - optional' and a field for entering a prefix, with a note that special characters must be URI encoded.

The screenshot shows the configuration dialog for adding a new trigger to a Lambda function. The trigger type selected is "All object create events" for the S3 service. A prefix "23images" has been entered. The "Recursive invocation" note is visible, along with a checkbox for acknowledging the risk of recursive invocation. A note at the bottom states that Lambda will add necessary permissions for AWS S3 to invoke the function.

**Trigger configuration:**

- Trigger type: All object create events
- S3 bucket prefix: 23images
- Recursive invocation acknowledged:
- Lambda will add necessary permissions for AWS S3 to invoke your Lambda function from this trigger.

**Step 8:** Now Write code that logs a message like “An Image has been added” when triggered. Save the file and click on deploy.

The screenshot shows the configuration page for the "MY-lambda" function. It lists one trigger named "S3: v2buckets". The right sidebar displays details such as Last modified (28 minutes ago), Function ARN (arn:aws:lambda:us-east-1:022499016110:function:MY-lambda), and Function URL (Info).

**Function Configuration:**

- Code, Test, Monitor, Configuration (selected), Aliases, Versions
- Triggers (1) Info: S3: v2buckets
- General configuration, Triggers (selected), Permissions, Destinations, Function URL, Environment

**Step 9:** Now upload any image to the bucket.

The screenshot shows the AWS S3 'Upload' interface. At the top, it says 'Amazon S3 > Buckets > v2buckets > Upload'. Below that is a large 'Upload' button with an 'Info' link. A note below the button 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](#)'.

In the center, there's a dashed box with the text '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, 155.2 KB)'. It contains one item: 'Screenshot (1).png' (image/png). There are 'Remove', 'Add files', and 'Add folder' buttons at the top of the table. A search bar and pagination controls are also present.

At the bottom, there's a 'Destination' section with a dropdown menu set to 'v2buckets'.

**Step 10:** Now to click on test in lambda to check whether it is giving log when image is added to S3.

The screenshot shows the AWS Lambda function editor. At the top, there are tabs: 'Code', 'Test', 'Monitor', 'Configuration', 'Aliases', and 'Versions'. The 'Code' tab is selected.

Below the tabs is a 'Code source' section with an 'Info' link. It has a toolbar with 'File', 'Edit', 'Find', 'View', 'Go', 'Tools', 'Window', 'Test' (which is highlighted), 'Deploy', and 'Changes not deployed'.

The main area shows a code editor with a Python script named 'lambda\_function.py'. 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
8     print(f"An Image has been added to the bucket {bucket_name} : {objecy_key}")
9
10    return {
11        'statusCode': 200,
12        'body': json.dumps('Log entry created successfully')
13    }

```

The screenshot shows the AWS Lambda Test console interface. At the top, there are tabs for 'Code source' and 'Info'. Below the tabs is a toolbar with 'File', 'Edit', 'Find', 'View', 'Go', 'Tools', 'Window', 'Test' (which is currently selected), and 'Deploy'. On the left, there's a sidebar labeled 'Environment' with a dropdown menu showing 'MY-lambda /' and 'lambda\_function.py'. The main area has tabs for 'Execution results', 'Execution result' (selected), and '+'. Under 'Execution result', it shows 'Test Event Name: event-exp\_12'. The 'Response' section contains a JSON object with 'statusCode': 200 and 'body': '\"Log entry created successfully\"'. The 'Function Logs' section displays logs from the function execution, including START and END Request IDs, Duration, Billed Duration, Memory Size, and Max Memory. The 'Request ID' is also listed.

**Step 11:** Now Lets see the log on Cloud watch. To see it go to monitor section and then click on view cloudwatch logs.

The screenshot shows the AWS CloudWatch Log Events interface. The left sidebar has sections for 'CloudWatch', 'Favorites and recents', 'Logs' (selected), 'Log groups', 'Log Anomalies', 'Live Tail', 'Logs Insights', 'Contributor Insights', 'Metrics', 'X-Ray traces', 'Events', and 'Application Signals'. The main area shows a log group path: CloudWatch > Log groups > /aws/lambda/MY-lambda > 2024/10/05/[\$.LATEST]f42c0b3426b9499ea3d2b422849c0be. It displays a table of log events with columns for 'Timestamp' and 'Message'. The messages show the function starting, an unhandled exception occurring, and the function ending.

Timestamp	Message
2024-10-05T11:53:17.811Z	INIT_START Runtime Version: python:3.12.v36 Runtime Version ARN: arn:aws:lambda:us-east-1:123456789012:function:MY-lambda Version: \$LATEST
2024-10-05T11:53:17.914Z	START RequestId: 30a151e7-098f-4574-9bf7-71dcbe15cb0a Version: \$LATEST
2024-10-05T11:53:17.916Z	LAMBDA_WARNING: Unhandled exception. The most likely cause is an issue in the function code.
2024-10-05T11:53:17.916Z	[ERROR] KeyError: 'Records' Traceback (most recent call last): File "/var/task/lambda_function.py", line 10, in lambda_handler
2024-10-05T11:53:17.935Z	END RequestId: 30a151e7-098f-4574-9bf7-71dcbe15cb0a
2024-10-05T11:53:17.935Z	REPORT RequestId: 30a151e7-098f-4574-9bf7-71dcbe15cb0a Duration: 19.19 ms Billed Duration: 19.19 ms Memory Size: 128 MB Max Memory: 128 MB
2024-10-05T11:53:31.022Z	START RequestId: ef4bf0a2-8819-4f31-bfce-00f230170210 Version: \$LATEST
2024-10-05T11:53:31.022Z	An image has been added to the bucket example-bucket. test%2Fkey

**Conclusion:** In this experiment, we successfully created an AWS Lambda function that logs a message when an image is uploaded to an S3 bucket. The function was successfully triggered by S3 object uploads, validating the functionality of Lambda's event-driven architecture. This experiment demonstrated how Lambda can efficiently respond to S3 events and how to troubleshoot common issues with event structure.

**Errors faced during the experiment :**

**It is important to configure S3 triggers properly, after changing the code its important to save the code then deploy it and then click on the test to see successful output otherwise we might see an error after clicking on the test.**