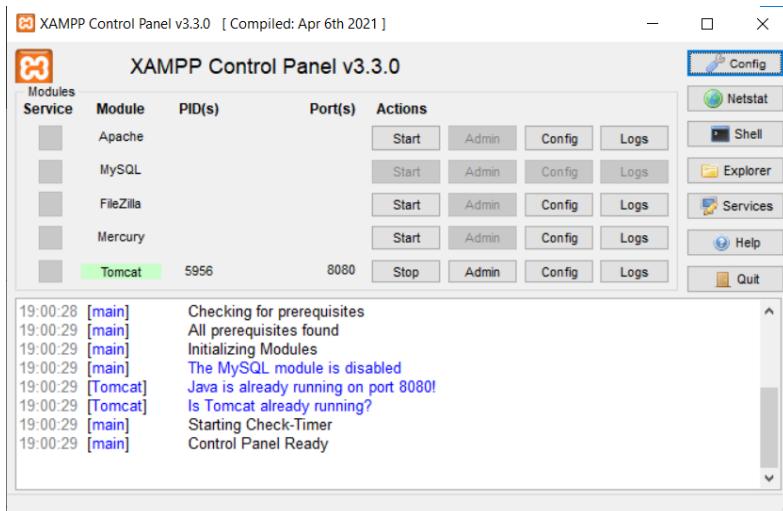


Experiment 1A

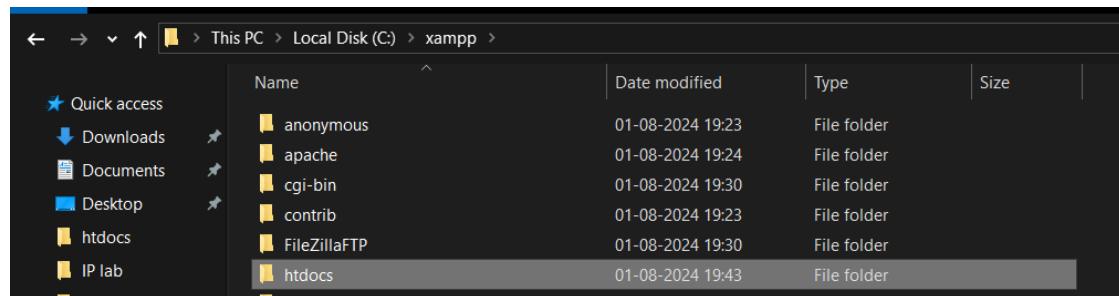
Part 1: To develop a website and host it on Xampp:

Step 1: Go to the official website of Xampp. <https://www.apachefriends.org/download.html>. Select the suitable version and complete the installation.

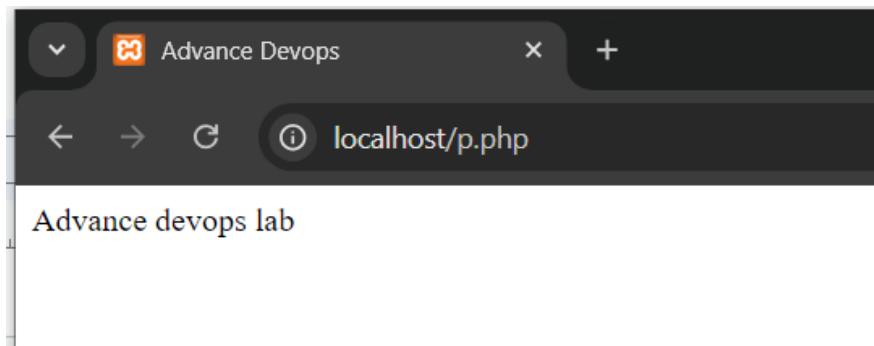
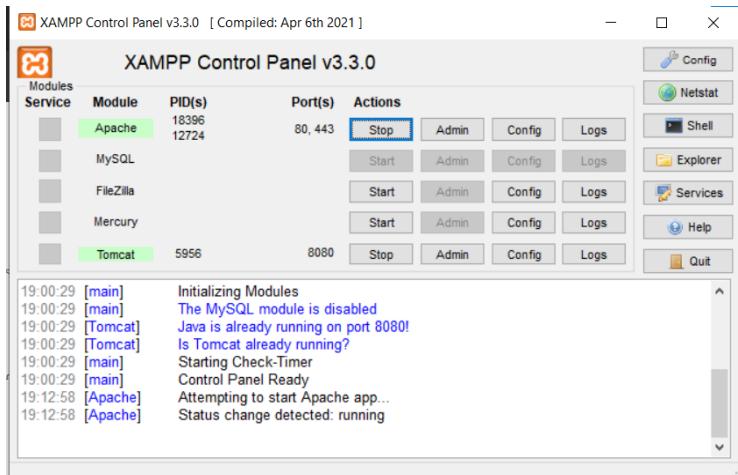
Step 2: After installation open the Xampp control panel, we need to start the Apache server to host the website.



Step 3: Setup a php project, create a php project and save it as .php file. The file should be saved in 'htdocs' folder present in the 'Xampp' folder created after installation.

A screenshot of a code editor window titled "p.php". The code is as follows:
1 <html>
2 <head>
3 <title>Advance Devops</title>
4 </head>
5 <body>
6 <?php
7 echo "Advance devops lab";
8 ?>

Step 4: Start the Xampp server and go to localhost in browser. Edit the URL such that it is localhost/filename.php or localhost/folder_name .

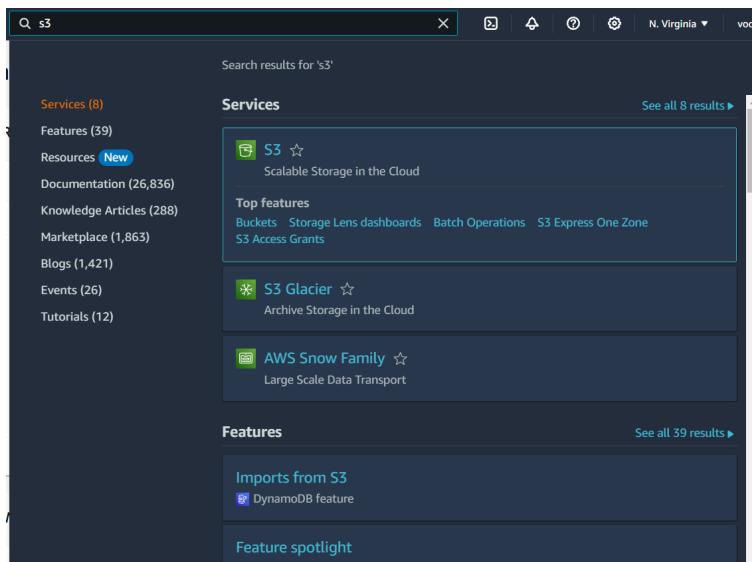


Our website has now been hosted using xampp

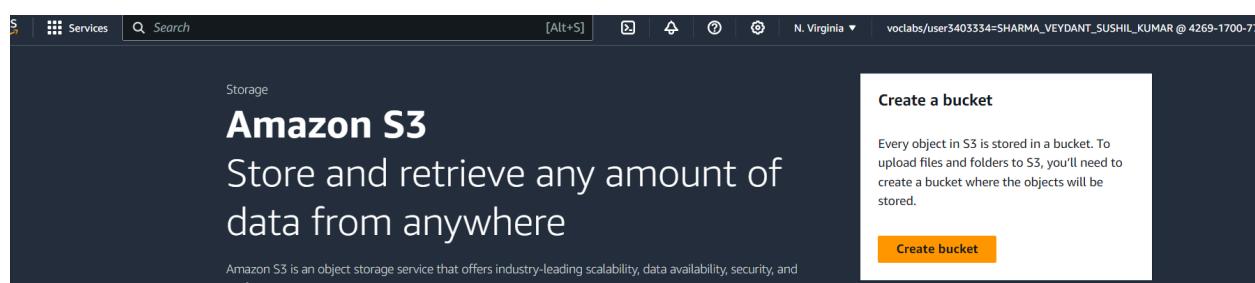
Part 2: Hosting a static website on AWS S3.

Step1: Login to AWS Academy and launch learner's lab.

Step 2: Search for the S3 service and Create a bucket.

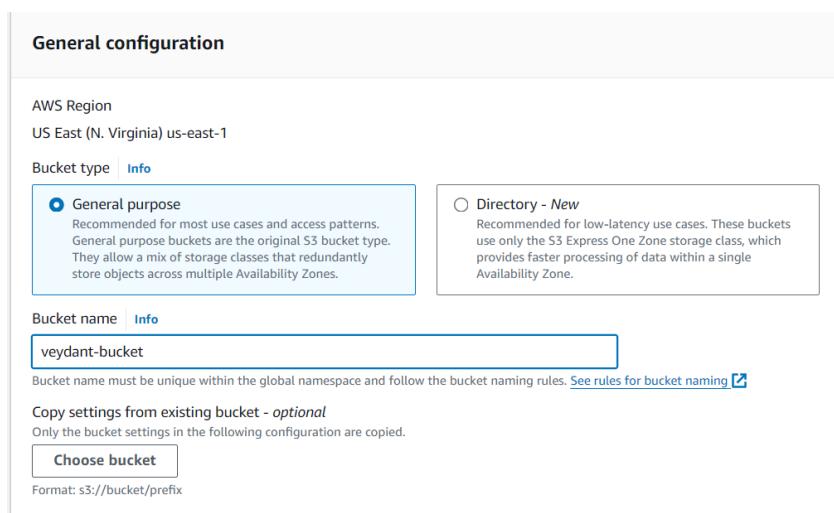


The screenshot shows the AWS search interface with the query 's3'. The top result is the 'S3' service card, which is highlighted. It displays the service name, a star icon, and a brief description: 'Scalable Storage in the Cloud'. Below the service card are sections for 'Top features' (Buckets, Storage Lens dashboards, Batch Operations, S3 Express One Zone, S3 Access Grants) and 'AWS Snow Family' (Archive Storage in the Cloud). Further down are sections for 'Features' (Imports from S3, DynamoDB feature) and 'Feature spotlight' (S3 feature).



The screenshot shows the Amazon S3 landing page. The main heading is 'Amazon S3' with the tagline 'Store and retrieve any amount of data from anywhere'. A subtext below states: 'Amazon S3 is an object storage service that offers industry-leading scalability, data availability, security, and performance.' To the right, there is a 'Create a bucket' section with a descriptive paragraph and a prominent orange 'Create bucket' button.

Step 3: Fill in the details and name your bucket. Use default settings. Uncheck the “Block all public access” box to prevent error.



This screenshot shows the 'General configuration' step of the AWS S3 bucket creation wizard. It includes fields for 'AWS Region' (set to 'US East (N. Virginia) us-east-1'), 'Bucket type' (radio buttons for 'General purpose' and 'Directory - New'), 'Bucket name' (input field containing 'veydant-bucket'), and a note about unique bucket names. There are also sections for 'Copy settings from existing bucket - optional' and a 'Choose bucket' button.

Block Public Access settings for this bucket

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

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

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

Once completed, Click 'Create Bucket'.

The screenshot shows the Amazon S3 console with a green header bar indicating "Successfully created bucket 'veydant-bucket'". Below the header, there's a message: "To upload files and folders, or to configure additional bucket settings, choose View details." The main area shows a single bucket named "veydant-bucket" under the "General purpose buckets" tab. The bucket details include: Name: veydant-bucket, AWS Region: US East (N. Virginia) us-east-1, IAM Access Analyzer: View analyzer for us-east-1, Creation date: August 8, 2024, 19:27:53 (UTC+05:30). There are buttons for "Copy ARN", "Empty", "Delete", and "Create bucket". A "View Storage Lens dashboard" button is also present.

Step 4: Open the bucket and click on upload in the objects tab.

The screenshot shows the "Objects" tab of the Amazon S3 console for the "veydant-bucket". The top navigation bar includes tabs for Objects, Properties, Permissions, Metrics, Management, and Access Points. The "Actions" menu is expanded, showing options like Copy S3 URI, Copy URL, Download, Open, Delete, Create folder, and Upload. A search bar for "Find objects by prefix" is present. The main table displays "No objects" with the note "You don't have any objects in this bucket." A large "Upload" button is located at the bottom of the table. Below the table, a dashed box allows users to "Drag and drop files and folders you want to upload here, or choose Add files or Add folder." At the bottom, a "Files and folders" section shows 2 items: "error.html" and "demo.html", both with a size of 244.0 B. Buttons for Remove, Add files, and Add folder are available for managing these files.

After adding the files click on upload files.

The screenshot shows the AWS S3 console interface. At the top, there's a green header bar indicating an 'Upload succeeded' message. Below it, the 'Summary' section shows the destination as 's3://veydant-bucket' and lists two succeeded uploads: 'error.html' and 'demo.html'. Under the 'Files and folders' tab, a table lists the uploaded files with their names, types, sizes, and statuses. Both files are marked as 'Succeeded'.

Name	Folder	Type	Size	Status	Error
error.html	-	text/html	123.0 B	Succeeded	-
demo.html	-	text/html	121.0 B	Succeeded	-

Step 5: Go to the properties tab and navigate to the “Static website hosting” section and click on edit , then click on enable.

The screenshot shows the 'Static website hosting' properties page. It includes a note to use the bucket for hosting a website or redirect requests, an 'Edit' button, and a section where 'Static website hosting' is currently set to 'Disabled'.

Specify the document/filenames and click on save changes.

The screenshot shows the detailed configuration for static website hosting. It includes sections for enabling hosting, choosing a hosting type (Host a static website), and specifying index and error documents. A note about making content publicly readable is present, along with instructions for using Amazon S3 Block Public Access.

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

Disable
 Enable

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

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

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

Step 6: Head to the Permissions tab and navigate to the “Bucket Policy” section and click on edit.

The screenshot shows the 'Bucket policy' section of the AWS S3 console. It displays a JSON policy document:

```

1▼ ( 
2    "Version": "2012-10-17",
3    "Statement": [
4      {
5        "Sid": "PublicReadGetObject",
6        "Effect": "Allow",
7        "Principal": "*",
8        "Action": "s3:GetObject",
9        "Resource": "arn:aws:s3:::veydant-bucket/*"
10      }
11    ]
12  ]
13 ]
14 )
  
```

At the top right, there are 'Edit' and 'Delete' buttons. Below the policy, there's a note: 'The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts.' A 'Learn more' link is also present.

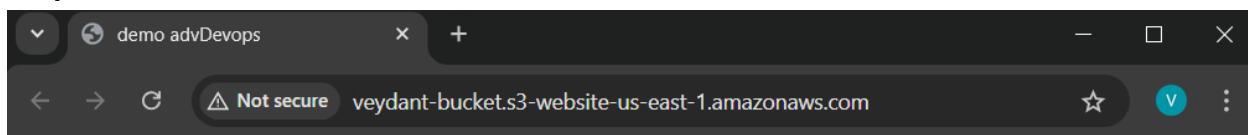
Add the bucket policy , save changes and exit.

The screenshot shows the 'Bucket policy' section of the AWS S3 console. It displays a JSON policy document with a single statement allowing public read access to all objects in the bucket. The 'Edit statement' button is visible at the top right of the policy area.

Step 7: Once the Bucket policy has been updated , navigate back to “Static website hosting”, a link will be available, click on the link to view the hosted/deployed website.

The screenshot shows the 'Static website hosting' section of the AWS S3 console. It indicates that the bucket is configured for static website hosting with the 'Enabled' option selected. The 'Hosting type' is set to 'Bucket hosting'. The 'Bucket website endpoint' is listed as 'veydant-bucket.s3-website-us-east-1.amazonaws.com'. A link to the endpoint is provided at the bottom.

Output:



this is advance devops

Experiment 1B

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

Step 1: Navigate to Cloud 9 service

The screenshot shows the AWS Cloud 9 search results. The search bar at the top contains 'cloud 9'. The results are categorized under 'Services' and 'Features'. Under 'Services', 'Cloud9' is listed as 'A Cloud IDE for Writing, Running, and Debugging Code'. Under 'Features', 'Cloud WAN', 'Namespaces', and 'Workloads' are listed. On the right side of the search results, there is a sidebar titled 'Create application' with fields for 'Name', 'Region', and 'Originating account'. The bottom of the page includes standard AWS navigation links like CloudShell, Feedback, and a footer with copyright information.

The screenshot shows the AWS Cloud9 landing page. The title is 'AWS Cloud9' with the subtitle 'A cloud IDE for writing, running, and debugging code'. Below the title, a paragraph explains that AWS Cloud9 allows you to write, run, and debug your code with just a browser. It mentions immediate access to a rich code editor, integrated debugger, and built-in terminal with preconfigured AWS CLI. A large orange button labeled 'Create environment' is prominently displayed on the right.

Step 2: Create Environment

The screenshot shows the 'Create environment' form. The top navigation bar shows 'AWS Cloud9 > Environments > Create environment'. The main section is titled 'Create environment' with a 'Details' tab selected. The 'Name' field is filled with 'vey-env'. The 'Description - optional' field is empty. The 'Environment type' section has two options: 'New EC2 instance' (selected) and 'Existing compute'. The 'New EC2 instance' option includes a note: 'Cloud9 creates an EC2 instance in your account. The configuration of your EC2 instance cannot be changed by Cloud9 after creation.' The 'Existing compute' option includes a note: 'You have an existing instance or server that you'd like to use.'

Network settings Info

Connection
How your environment is accessed.

- AWS Systems Manager (SSM)
Accesses environment via SSM without opening inbound ports (no ingress).
- Secure Shell (SSH)
Accesses environment directly via SSH, opens inbound ports.

► **VPC settings** Info

Click on “Create”

Step 3: Environment Will be created.

Successfully created vey-env. To get the most out of your environment, see [Best practices for using AWS Cloud9](#)

For capabilities similar to AWS Cloud9, explore AWS Toolkits in your own IDE and AWS CloudShell in the AWS Management Console. [Learn more](#)

[AWS Cloud9](#) > Environments

Name	Cloud9 IDE	Environment type	Connection	Permission	Owner ARN
vey-env	Open	EC2 instance	Secure Shell (SSH)	Owner	arn:aws:sts::426917007754:assumed-role/voclabs/user3403534=SHARMA_VEYDANT_SUSHIL_KUMAR

Step 4: Navigate to IAM service in AWS.

Search results for 'IAM'

Services

- IAM** ☆ Manage access to AWS resources
- IAM Identity Center** ☆ Manage workforce user access to multiple AWS accounts and cloud applications
- Resource Access Manager** ☆ Share AWS resources with other accounts or AWS Organizations
- AWS App Mesh** ☆ Easily monitor and control microservices

Features

- Groups**
- IAM feature**

Step 5: Navigate to users tab and Create new User.

The screenshot shows the AWS IAM service interface. On the left, the navigation pane is open, showing the 'Identity and Access Management (IAM)' section. Under 'Access management', the 'Users' option is selected. The main content area is titled 'Users (0) Info' and contains a message stating 'An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.' Below this is a search bar and a table header with columns: 'User name', 'Path', 'Groups', 'Last activity', 'MFA', 'Password age', and 'Console last sign-in'. A note at the bottom says 'No resources to display'.

Add name, custom password. Keep the “Useers create new password at next sign in” button checked

This screenshot shows the 'Create new IAM user' wizard at Step 3: 'Review and create'. The left sidebar lists 'Step 3: Review and create' and 'Step 4: Retrieve password'. The main form has a 'User name' field set to 'Veydant'. A note below it specifies character restrictions. A checkbox 'Provide user access to the AWS Management Console - optional' is checked. A callout box highlights the 'User type' section, where the 'I want to create an IAM user' radio button is selected. It notes that IAM users are recommended for programmatic access. The 'Console password' section shows 'Custom password' is selected, with a password entered in a masked field. A note specifies password requirements: at least 8 characters, mix of uppercase, lowercase, numbers, and symbols. A checkbox 'Users must create a new password at next sign-in - Recommended' is checked. The right sidebar shows icons for help and refresh.

This screenshot shows the 'Set permissions' step of the wizard. The left sidebar lists 'Step 1: Specify user details' and 'Step 2: Set permissions'. The main area is titled 'Set permissions' with a note about using groups or creating a new one. It features three options: 'Add user to a group' (selected), 'Copy permissions', and 'Attach policies directly'. A callout box for 'Get started with groups' suggests creating a group and selecting policies. A 'Create group' button is available. At the bottom, there's a note about setting a 'permissions boundary - optional'.

Select default settings for the rest.

User created successfully

You can view and download the user's password and email instructions for signing in to the AWS Management Console.

IAM > Users > Create user

Step 1 Specify user details

Step 2 Set permissions

Step 3 Review and create

Step 4 Retrieve password

Console sign-in details

Console sign-in URL: <https://975050293750.signin.aws.amazon.com/console>

User name: Veydant

Console password: ***** Show

Email sign-in instructions

Cancel Download .csv file Return to users list

Step 6: Navigate to User groups tab.

Identity and Access Management (IAM)

Search IAM

Dashboard

Access management

- User groups
- Users
- Roles
- Policies
- Identity providers
- Account settings

Access reports

- Access Analyzer
- External access
- Unused access
- Analyzer settings
- Credential report
- Organization activity

CloudShell Feedback

Identity and Access Management (IAM)

User groups (0) Info

No resources to display

Create group

Group name	Users	Permissions	Creation time
------------	-------	-------------	---------------

User group name
Enter a meaningful name to identify this group.
D15C_50
Maximum 128 characters. Use alphanumeric and '+,-_,@,_.' characters.

Add users to the group - Optional (1) Info

An IAM user is an entity that you create in AWS to represent the person or application that uses it to interact with AWS.

User name	Groups	Last activity	Creation time
Veydant	0	None	1 minute ago

Attach permissions policies - Optional (945) Info

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.

Filter by Type

Create a new user group.

The screenshot shows the 'User groups' page in the AWS IAM console. A green banner at the top indicates that the 'D15C_50' user group has been created. The main table lists one group: 'D15C_50'. The 'Permissions' column shows a warning icon and 'Not defined'. The 'Creation time' column shows 'Now'. The page includes a search bar, navigation buttons, and a 'Create group' button.

Step 7: Navigate to Users tab , go to Groups tab and add User to the group.

The screenshot shows the 'Veydant' user details page in the AWS IAM console. The 'Groups' tab is selected, showing '1' group member: 'D15C_50'. Other tabs include 'Permissions', 'Tags', 'Security credentials', and 'Access Advisor'. The 'Summary' section provides basic user information like ARN, console access status, and creation date.

The screenshot shows the 'Add user to groups' dialog box in the AWS IAM console. It lists the 'D15C_50' group under 'Other groups'. The 'Attached policies' column is empty. The 'Created' column shows '2024-08-08 (16 minutes ago)'. At the bottom, there is a 'Cancel' button and an orange 'Add user to group(s)' button.

The screenshot shows the AWS IAM User details page for 'Veydant'. The top navigation bar includes 'Services' and 'Search' with the AWS logo. The main title is 'User added to group D15C_50'. The left sidebar has sections for 'Access management' (User groups, Users, Roles, Policies, Identity providers, Account settings) and 'Access reports' (Access Analyzer, External access, Unused access, Analyzer settings, Credential report, Organization activity). The main content area shows 'Veydant' info with a 'Summary' section containing ARN (arn:aws:iam::975050293750:user/Veydant), Console access (Enabled without MFA), Created (August 08, 2024, 23:36 (UTC+05:30)), Last console sign-in (Never), and Access key 1 (Create access key). Below the summary are tabs for 'Permissions', 'Groups (1)', 'Tags', 'Security credentials', and 'Access Advisor'. The 'Groups (1)' tab is selected, showing 'User groups membership (1)'. It lists 'D15C_50' as a member of the group. There are 'Remove' and 'Add user to groups' buttons.

The user has been added to the the user group.

Step 8 : Navigate to the permissions tab in User group, select "AWSCloud9EnvironmentMember" and add the permissions.

The screenshot shows the AWS IAM User group details page for 'D15C_50'. The top navigation bar includes 'Services' and 'Search' with the AWS logo. The main title is 'Policies attached to this user group.' The left sidebar has sections for 'Access management' (User groups, Users, Roles, Policies, Identity providers, Account settings) and 'Access reports' (Access Analyzer, External access, Unused access, Analyzer settings, Credential report, Organization activity). The main content area shows 'D15C_50' info with a 'Summary' section containing User group name (D15C_50), Creation time (August 08, 2024, 23:20 (UTC+05:30)), and ARN (arn:aws:iam::975050293750:group/D15C_50). Below the summary are tabs for 'Users (1)', 'Permissions', and 'Access Advisor'. The 'Permissions' tab is selected, showing 'Permissions policies (1)'. It lists 'AWSCloud9EnvironmentMember' as a policy attached to the group. There are 'Edit', 'Simulate', 'Remove', and 'Add permissions' buttons.

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.

Step 1: Search for IAM , Go to Roles section and click on create a role.

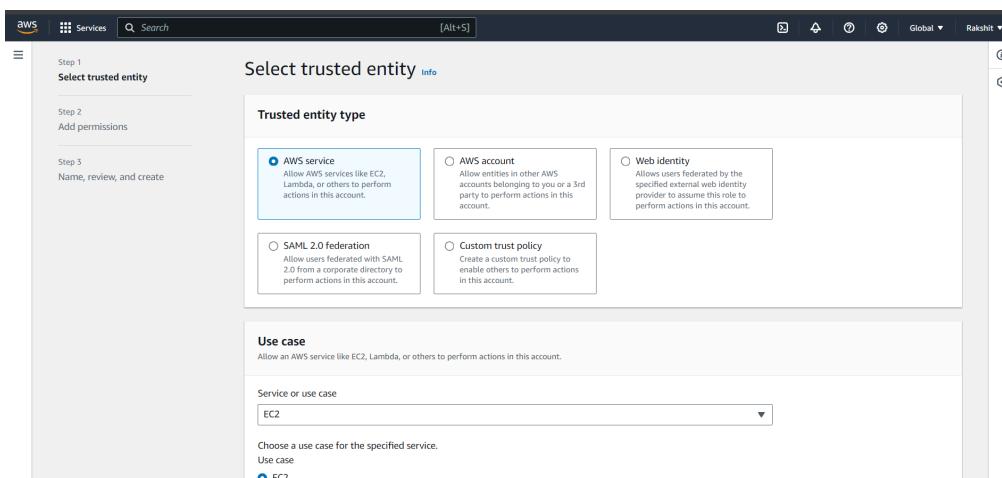
The screenshot shows the AWS search interface with the query 'IAM' entered in the search bar. The results are categorized into 'Services' and 'Features'. Under 'Services', 'IAM' is listed as 'Manage access to AWS resources'. Under 'Features', there are sections for 'Groups', 'Roles', 'Roles Anywhere', and 'Access Analyzer', all of which are 'IAM features'. The 'See all 11 results' link is visible at the top right of the 'Services' section.

Step 2: Fill the role details. Select AWS service and enable web tier and work tier permissions.

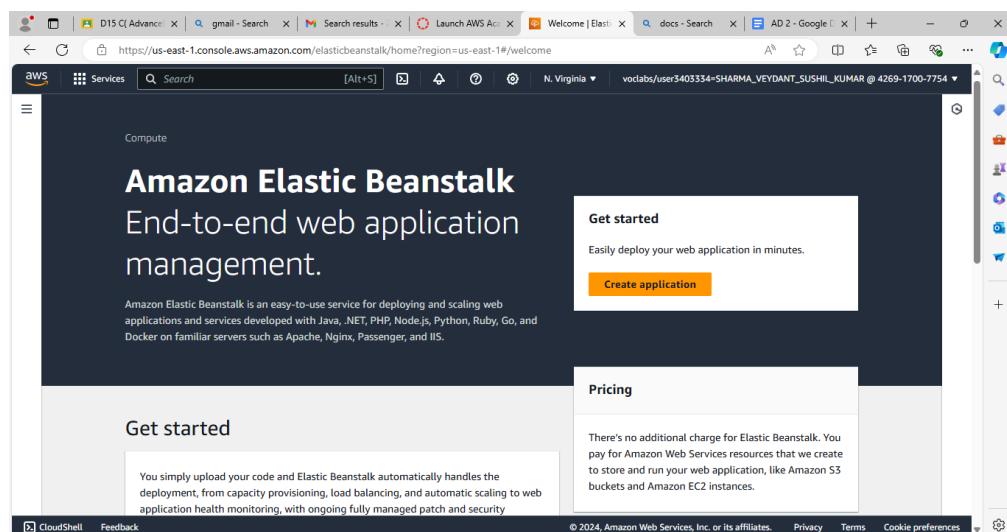
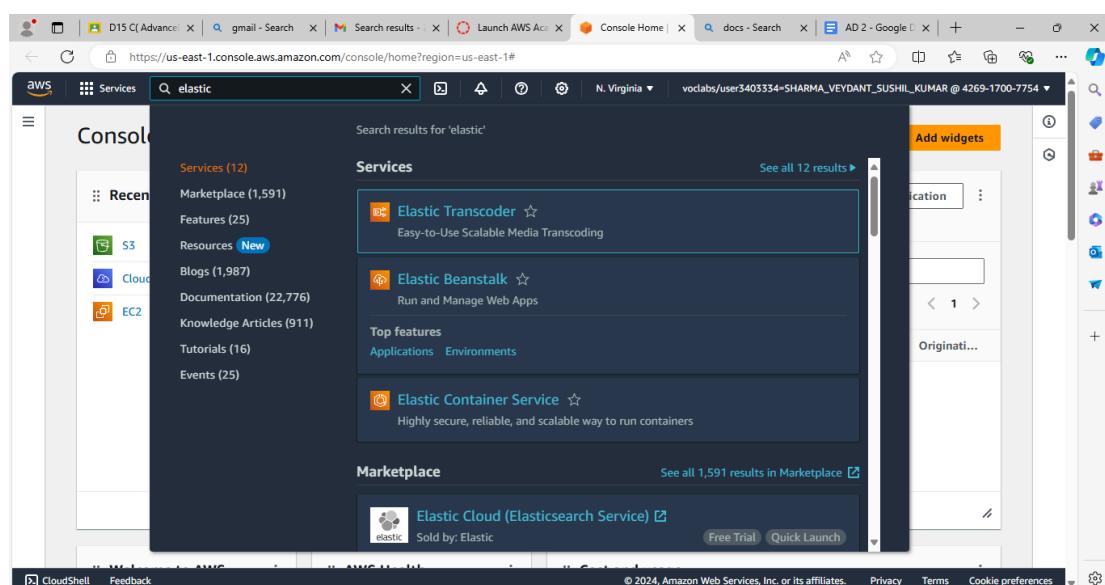
The screenshot shows the 'Create New Role' wizard. The current step is 'Role details'. The 'Role name' field contains 'elastic-bs'. The 'Description' field contains 'Allows EC2 instances to call AWS services on your behalf.' The 'Step 1: Select trusted entities' section shows a JSON trust policy:

```
1: {
2:     "Version": "2012-10-17",
3:     "Statement": [
4:         {
5:             "Effect": "Allow",
6:             "Action": [
7:                 "sts:AssumeRole"
8:             ],
9:             "Principal": [
10:                 "ec2.amazonaws.com"
11:             ]
12:         }
13:     ]
14: }
```

The 'Step 2: Add permissions' section is partially visible below.



Step 3: Search elastic beanstalk . After loading the page click on Create application.



Step 4: Create application , provide application name and environment

Application information

Application name: aws50

Environment name: Aws50-env

Domain: Leave blank for autogenerated value

Platform type: Managed platform

Select the platform.

Platform Info

Platform type: Managed platform

Platform: PHP

Platform branch: PHP 8.3 running on 64bit Amazon Linux 2023

Platform version: 4.3.1 (Recommended)

Click on 'Use existing service role' in the service menu . Assign the service role as the one create in IAM, also assign the key pair.

Configure service access

Service role: Use an existing service role

Existing service roles: elastic-bs

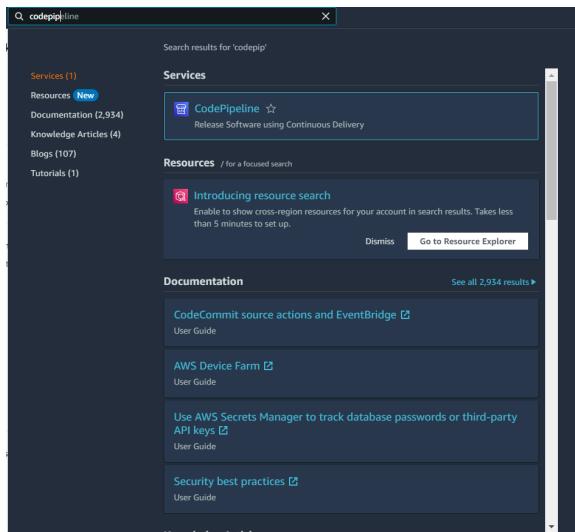
EC2 key pair: rakshit

EC2 instance profile: elastic-bs

Keep the rest of the settings as default. Wait for the environment to be created/launched successfully.

Step 5: Get a sample of the code from github , fork the repository.

Step 6: Go to codepipeline, Create new pipeline.



Step 7: Name the pipeline. Head on to the 'Source' Page.

Pipeline settings

Pipeline name
Enter the pipeline name. You cannot edit the pipeline name after it is created.
 No more than 100 characters

Pipeline type
You can no longer create V1 pipelines through the console. We recommend you use the V2 pipeline type with improved release safety, pipeline triggers, parameterized pipelines, and a new billing model.

Execution mode
Choose the execution mode for your pipeline. This determines how the pipeline is run.
 Superseded
 Queued (Pipeline type V2 required)
 Parallel (Pipeline type V2 required)

Service role
 New service role
 Create a service role in your account
 Existing service role
 Choose an existing service role from your account

Role name

 Type your service role name
 Allow AWS CodePipeline to create a service role so it can be used with this new

On the source page, select source provider as Github V2 , connect github account, and select the repository and the branch.

Source

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

GitHub (Version 2) ▾

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

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

arn:aws:codeconnections:us-east-1:975050293750:connection/eac0d1b2-a1: X or [Connect to GitHub](#)

Ready to connect
Your GitHub connection is ready for use.

Repository name
Choose a repository in your GitHub account.

v ey-droid/aws-codepipeline-s3-codedeploy-linux-2.0 X

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

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

master X

Output artifact format
Choose the output artifact format.

CodePipeline default
AWS CodePipeline uses the default zip format for artifacts in the pipeline. Does not include Git metadata about the repository.

Full clone
AWS CodePipeline passes metadata about the repository that allows subsequent actions to do a full Git clone. Only supported for AWS CodeBuild actions.

Select defaults and skip the build page.

On the deploy page , select provider as AWS Elastic Beanstalk. Select the Application and environment name, then proceed further.

Deploy

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

AWS Elastic Beanstalk ▾

Region

US East (N. Virginia) ▾

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

No more than 100 characters ▾

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

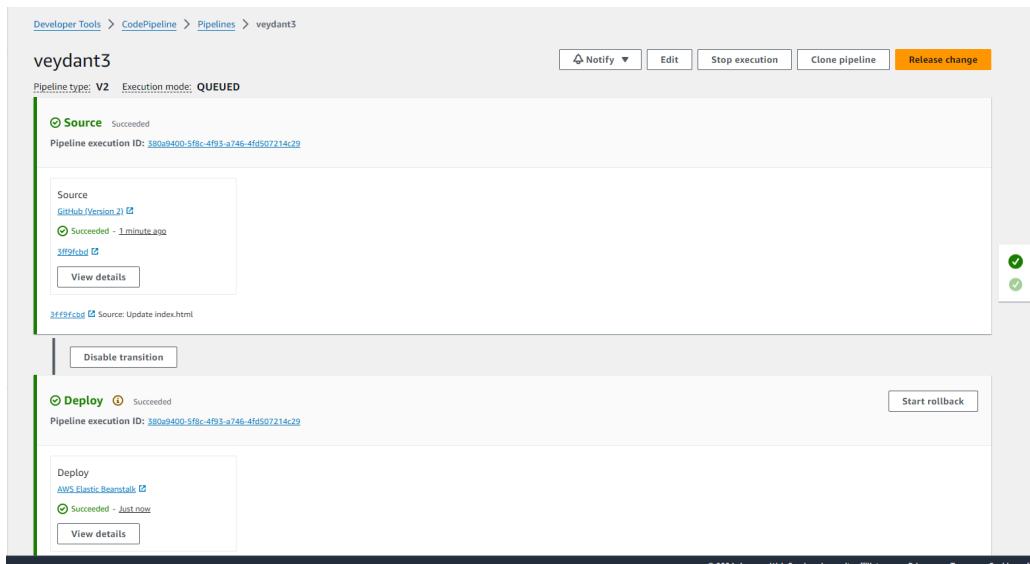
v eydant-server X

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

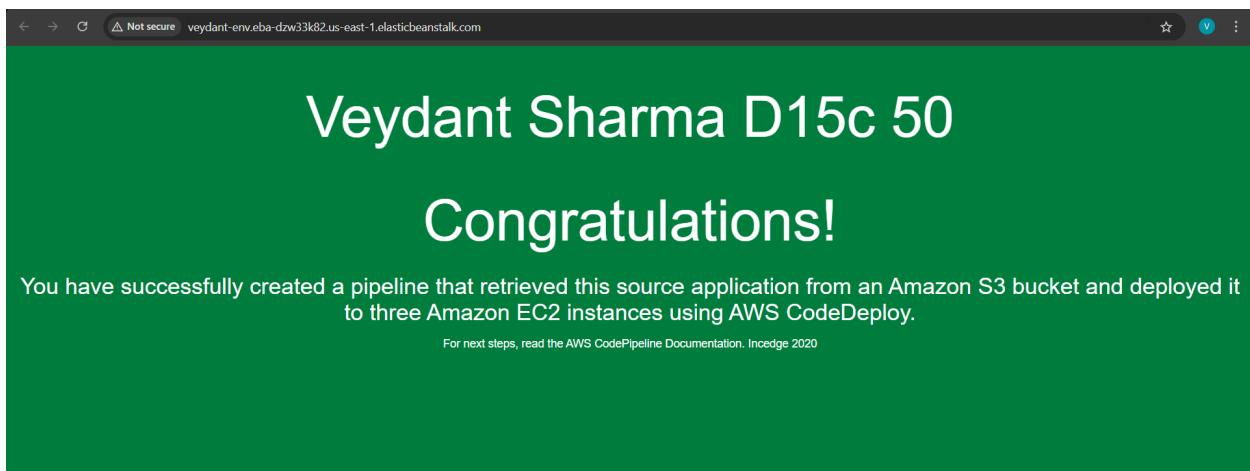
V eydant-server-env X

Configure automatic rollback on stage failure

Step 8: The pipeline will be launched in a few moments, once it is successfully created head over to elastic beanstalk environment.



Step 9: A URL will be created on the environment page, that is the hosted link, click on it.



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

Step 1: Make 3 EC2 Instances, 1 Master 2 Node instances, use Ubuntu , t2.medium adn add the security groups

Master:

Name and tags [Info](#)

Name Add additional tags

Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Search our full catalog including 1000s of application and OS images

Recent [Quick Start](#)

Amazon Linux macOS Ubuntu Windows Red Hat SUSE Linux

[Browse more AMIs](#) Including AMIs from AWS, Marketplace and the Community

Security group rule 4 (TCP, 6445) [Remove](#)

Security group rule 5 (TCP, 10251) [Remove](#)

Type Info	Protocol Info	Port range Info
Custom TCP	TCP	10251
Source type Info	Source Info	Description - optional Info
Custom	<input type="text" value="Add CIDR, prefix list or security group"/>	e.g. SSH for admin desktop

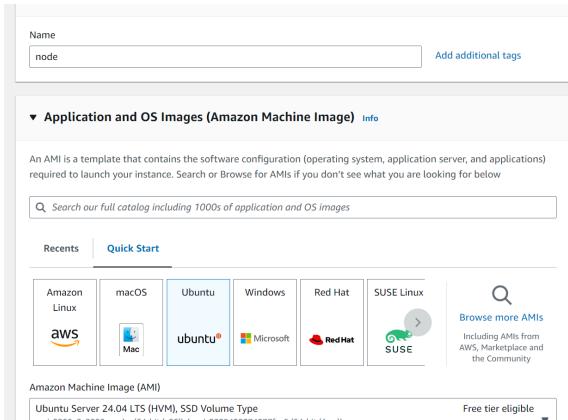
Security group rule 6 (TCP, 10250) [Remove](#)

Type Info	Protocol Info	Port range Info
Custom TCP	TCP	10250
Source type Info	Source Info	Description - optional Info
Custom	<input type="text" value="Add CIDR, prefix list or security group"/>	e.g. SSH for admin desktop

Security group rule 7 (TCP, 10252) [Remove](#)

Type Info	Protocol Info	Port range Info
Custom TCP	TCP	10252
Source type Info	Source Info	Description - optional Info
Custom	<input type="text" value="Add CIDR, prefix list or security group"/>	e.g. SSH for admin desktop

Worker:



Step 2: After creating the instances click on Connect & connect all 3 instances and navigate to SSH Client.

Instances (4) [Info](#) Last updated less than a minute ago [C](#) [Connect](#) [Instance state](#) [Actions](#) [Launch instances](#)

<input type="checkbox"/>	Name 🔗	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	nagios-host	i-06c955d252d68c106	Terminated	t2.micro	...	View alarms +
<input type="checkbox"/>	master	i-03753a87170da35bc	Running	t2.medium	2/2 checks passed	View alarms +
<input type="checkbox"/>	node	i-0b9d7d0a41c70f7f0	Running	t2.medium	Initializing	View alarms +
<input type="checkbox"/>	node	i-0c4a87fdff3eb59e0	Running	t2.medium	Initializing	View alarms +

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

Instance ID
 i-0ba8c41833c7c1f5b (AD4)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is dev.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
 chmod 400 "dev.pem"
4. Connect to your instance using its Public DNS:
 ec2-3-85-103-198.compute-1.amazonaws.com

Example:
 ssh -i "dev.pem" ec2-user@ec2-3-85-103-198.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.

Step 3: Now open the folder in the terminal 3 times for Master, Node1& Node 2 where our .pem key is stored and paste the Example command (starting with ssh -i) in the terminal.(ssh -i "Master_Ec2_Key.pem" ubuntu@ec2-54-196-129-215.compute-1.amazonaws.com) Master:

Step 4: Run on Master,Node 1, and Node 2 the below commands to install and setup Docker in Master, Node1, and Node2.

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

```
ubuntu@ip-172-31-81-239:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee
/etc/apt/trusted.gpg.d/docker.gpg > /dev/null
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable"
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
-----BEGIN PGP PUBLIC KEY BLOCK-----
mQINBFit2ioBEADhWpZ8/wvZ6hUTiX0wQHXMAlaFHcPH9hAtr4F1y2+0YdbtMuth
lqwp028AqY+PRfVmSYMbjuQuu5byKR01BbqYhuS3jtgQmljZ/bJvXqmniVXh
38UuLa+z077PxxyQhu5BbqntTPQMfiyQeiu+BKbq2WmANUKQf+1AmZY/IruOXbnq
L4C1+gJ8vfmXot99npCaxEjaNRVYf0S8QcixNzHUynb6emjLANyEVLzeqo7XKL7
UrwV5inawTSzwNvtjEej4nJL8NsLwsACPQUhTQ+7BbQXAwAmeHCUTQIvvWxqw0N
cmhh4HgeQscQHYg0JjjDvfoY8NsMucvglbIgCqfzAHW9jxmRL4qbMZj+b1XoePEtht
ku4bIQN1X5P07fNWzIgaRL5Z4POXDDZTLIQ/El58j9kp4bnWRcjW0lya+f8ocodo
vZZ+Doi+fy4D5ZGrL4XEcIQP/Lv5uFyf+kQtL/94VFYVJOLeAv8W92KdgDkhTcTD
G7c0tIkVEKNUq48b3aQ64NOZQW7fVjfokwEZd0qPE72Pa45jrZvUFxSpdiNk2tZ
XYukHjlxxEgBdC/J3cMMNRE1F4NCA3ApfV1Y7/hTeOnmDuDYwv9/obA8t016Yljj
q5rdkywPf4JF8mXUW5eCN1vAFHxeg9ZWemhBtQmGxXnw9M+z6hWwc6ahmwARAQAB
+tEh2NrZXTalImVcZWEzSAo0UuZGVikSA8ZG9ia2Vv0GRvY2t1ci5ib20+iQI3
```

```
sudo apt-get update
```

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

```
ubuntu@ip-172-31-81-239:~$ sudo apt-get update
sudo apt-get install -y docker-ce
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0
  pigz slirp4netns
Suggested packages:
  
```

```
sudo mkdir -p /etc/docker
```

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

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

```
ubuntu@ip-172-31-81-239:~$ sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
```

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

```
ubuntu@ip-172-31-81-239:~$ sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
```

Step 5: Run the below command to install Kubernets.

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

```
ubuntu@ip-172-31-81-239:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o
/etc/apt/keyrings/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list
gpg: missing argument for option "--"
-bash: /etc/apt/keyrings/kubernetes-apt-keyring.gpg: No such file or directory
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
```

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

sudo systemctl enable --now kubelet

sudo apt-get install -y containerd

```
ubuntu@ip-172-31-81-239:~$ sudo apt-get install -y apt-transport-https ca-certificates curl
curl -fsSL https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
echo "deb https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list
sudo apt-get update
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203).
ca-certificates set to manually installed.
curl is already the newest version (8.5.0-2ubuntu10.4).
curl set to manually installed.
The following NEW packages will be installed:
  apt-transport-https
0 upgraded, 1 newly installed, 0 to remove and 6 not upgraded.
Need to get 3974 B of archives.
After this operation, 35.8 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 apt-transport-https all 2.7.14build2 [3974 B]
```

sudo mkdir -p /etc/containerd

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

```
ubuntu@ip-172-31-81-239:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

[debug]
  address = ""
  format = ""
  gid = 0
  level = ""
  pid = 0
```

sudo systemctl restart containerd

sudo systemctl enable containerd

sudo systemctl status containerd

```
ubuntu@ip-172-31-81-239:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd
● containerd.service - containerd container runtime
   Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
   Active: active (running) since Thu 2024-10-03 12:03:56 UTC; 187ms ago
     Docs: https://containerd.io
     Main PID: 4388 (containerd)
       Tasks: 8
      Memory: 13.8M (peak: 14.1M)
        CPU: 47ms
       CGroup: /system.slice/containerd.service
               └─4388 /usr/bin/containerd

Oct 03 12:03:56 ip-172-31-81-239 containerd[4388]: time="2024-10-03T12:03:56.810259144Z" level=info msg="Start subscrib>
Oct 03 12:03:56 ip-172-31-81-239 containerd[4388]: time="2024-10-03T12:03:56.810436084Z" level=info msg="Start recoveri>
Oct 03 12:03:56 ip-172-31-81-239 containerd[4388]: time="2024-10-03T12:03:56.810483329Z" level=info msg=serving... addr>
Oct 03 12:03:56 ip-172-31-81-239 containerd[4388]: time="2024-10-03T12:03:56.810489545Z" level=info msg="Start event mo>
Oct 03 12:03:56 ip-172-31-81-239 containerd[4388]: time="2024-10-03T12:03:56.810583963Z" level=info msg="Start snapshot>
Oct 03 12:03:56 ip-172-31-81-239 containerd[4388]: time="2024-10-03T12:03:56.810599063Z" level=info msg="Start cni netw>
Oct 03 12:03:56 ip-172-31-81-239 containerd[4388]: time="2024-10-03T12:03:56.810605397Z" level=info msg="Start streamin>
Oct 03 12:03:56 ip-172-31-81-239 containerd[4388]: time="2024-10-03T12:03:56.810635592Z" level=info msg=serving... addr>
Oct 03 12:03:56 ip-172-31-81-239 systemd[1]: Started containerd.service - containerd container runtime.
Oct 03 12:03:56 ip-172-31-81-239 containerd[4388]: time="2024-10-03T12:03:56.813064087Z" level=info msg="containerd suc>
```

Step 6: Initialize the Kubecluster .Now Perform this Command only for Master.**sudo kubeadm init --pod-network-cidr=10.244.0.0/16**

```
ubuntu@ip-172-31-82-119:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W1003 12:37:31.290835    10158 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that used by kubeadm. It is recommended to use "registry.k8s.io/pause:3.10" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-82-119 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.82.119]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-82-119 localhost] and IPs [172.31.82.119 127.0.0.1 :1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-82-119 localhost] and IPs [172.31.82.119 127.0.0.1 ::1]
```

Run this command on master and also copy and save the Join command from above.

mkdir -p \$HOME/.kube

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

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

Step 7: Now Run the command kubectl get nodes to see the nodes before executing Join command on nodes.

ubuntu@ip-172-31-82-119:~\$ kubectl get nodes					
NAME	STATUS	ROLES	AGE	VERSION	
ip-172-31-82-119	NotReady	control-plane	82s	v1.31.1	

Step 8: Now Run the following command on Node 1 and Node 2 to Join to master.

```
ubuntu@ip-172-31-84-169:~$ sudo kubeadm join 172.31.82.119:6443 --token h4iisg.g9tdfmc88m9toefp \
--discovery-token-ca-cert-hash sha256:5b3c7cfdf8115a26f1e73b752820d2b27b84ed476b344aaa3bd1a90e4b1f2105
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 501.315929ms
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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

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

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

NAME	STATUS	ROLES	AGE	VERSION
ip-172-31-82-119	NotReady	control-plane	2m56s	v1.31.1
ip-172-31-84-169	NotReady	<none>	43s	v1.31.1
ip-172-31-87-189	NotReady	<none>	39s	v1.31.1

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

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

```
ubuntu@ip-172-31-82-119:~$ kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
poddisruptionbudget.policy/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
serviceaccount/calico-node created
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgppeers.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/blockaffinities.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/caliconodestatuses.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/clusterinformations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/felixconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworksets.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/hostendpoints.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamblocks.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamconfigs.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamhandles.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippools.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipreservations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/kubecontrollersconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networksets.crd.projectcalico.org created
clusterrole.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrole.rbac.authorization.k8s.io/calico-node created
```

```
sudo systemctl status kubelet
```

```
ubuntu@ip-172-31-82-119:~$ sudo systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
   Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; preset: enabled)
   Drop-In: /usr/lib/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: active (running) since Thu 2024-10-03 12:37:52 UTC; 4min 4s ago
       Docs: https://kubernetes.io/docs/
   Main PID: 10849 (kubelet)
     Tasks: 10 (limit: 4676)
    Memory: 32.3M (peak: 33.0M)
      CPU: 5.030s
     CGroup: /system.slice/kubelet.service
             └─10849 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/k>
Oct 03 12:41:52 ip-172-31-82-119 kubelet[10849]: E1003 12:41:52.678397 10849 kuberuntime_container.go:851] "Kill cont>
Oct 03 12:41:52 ip-172-31-82-119 kubelet[10849]:                      rpc error: code = Unknown desc = failed to kill container "233>
Oct 03 12:41:52 ip-172-31-82-119 kubelet[10849]:                      : unknown
Oct 03 12:41:52 ip-172-31-82-119 kubelet[10849]: > pod="kube-system/etcfd-ip-172-31-82-119" podUID="201366195a044a56173>
Oct 03 12:41:52 ip-172-31-82-119 kubelet[10849]: E1003 12:41:52.688038 10849 log.go:32] "StopPodSandbox from runtime >
Oct 03 12:41:52 ip-172-31-82-119 kubelet[10849]:                      rpc error: code = Unknown desc = failed to stop container "233>
Oct 03 12:41:52 ip-172-31-82-119 kubelet[10849]:                      : unknown
```

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

```
ubuntu@ip-172-31-82-119:~$ kubectl get nodes -o wide
NAME           STATUS    ROLES      AGE   VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE        KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-82-119   Ready    control-plane   4m32s   v1.31.1   172.31.82.119   <none>       Ubuntu 24.04.1 LTS   6.8.0-1
016-aws   containerd://1.17.12
ip-172-31-84-169   Ready    <none>      2m19s   v1.31.1   172.31.84.169   <none>       Ubuntu 24.04.1 LTS   6.8.0-1
016-aws   containerd://1.17.12
ip-172-31-87-189   Ready    <none>      2m15s   v1.31.1   172.31.87.189   <none>       Ubuntu 24.04.1 LTS   6.8.0-1
016-aws   containerd://1.17.12
```

Now to Rename run this command `kubectl label node ip-172-31-18-135`

kubernetes.io/role=worker Rename to Node 1:kubectl label node ip-172-31-28-117

kubernetes.io/role=Node1 Rename to Node 2:kubectl label node ip-172-31-18-135

kubernetes.io/role=Node2

Step 11: Run command `kubectl get nodes -o wide` . And Hence we can see we have

Successfully connected Node 1 and Node 2 to the Master.

```
ubuntu@ip-172-31-82-119:~$ kubectl get nodes
NAME           STATUS    ROLES      AGE     VERSION
ip-172-31-82-119   Ready    control-plane   6m52s   v1.31.1
ip-172-31-84-169   Ready    Node1       4m39s   v1.31.1
ip-172-31-87-189   Ready    Node2       4m35s   v1.31.1
```

Conclusion:

- Successfully set up a Kubernetes cluster on AWS EC2 instances.
 - Provisioned three EC2 instances: Master, Node1, and Node2, with appropriate security rules for internal communication.
 - Securely SSHed into each instance to manage installations.
 - Installed Docker as the container runtime on all instances.
 - Installed Kubernetes on each instance and initialized the cluster from the master node.

- Connected the worker nodes to the master using the kubeadm join command, forming a cohesive Kubernetes cluster.
- Implemented a network plugin to ensure proper communication between nodes.
- Verified that the nodes' status transitioned to "Ready."

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

Step 1: Login to AWS Account, Select EC2 instance, and choose instance type as t2.medium

The screenshot shows the AWS EC2 instance creation process. In the 'Name and tags' section, the name 'AD4' is entered. In the 'Application and OS Images (Amazon Machine Image)' section, the search bar contains 'Search our full catalog including 1000s of application and OS images'. Below the search bar are tabs for 'Recents' and 'Quick Start', with 'Quick Start' selected. A row of OS icons includes Amazon Linux, macOS, Ubuntu, Windows, Red Hat, and SUSE Linux. A magnifying glass icon and the text 'Browse more AMIs' are also present. In the 'Instance type' section, 't2.medium' is selected, and a detailed description of its specifications is shown. A note at the bottom states 'Additional costs apply for AMIs with pre-installed software'.

Name and tags [Info](#)

Name
AD4 [Add additional tags](#)

Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Recents | **Quick Start**

Amazon Linux | macOS | Ubuntu | Windows | Red Hat | SUSE Li [Browse more AMIs](#)

Instance type [Info](#) | [Get advice](#)

Instance type
t2.medium

Family: t2 2 vCPU 4 GiB Memory Current generation: true
On-Demand Linux base pricing: 0.0464 USD per Hour
On-Demand RHEL base pricing: 0.0752 USD per Hour
On-Demand Windows base pricing: 0.0644 USD per Hour
On-Demand SUSE base pricing: 0.1464 USD per Hour

All generations [Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

Step 2: After creating the instance click on Connect the instance and navigate to SSH Client.

Instances (1) [Info](#)

Last updated less than a minute ago

[Connect](#) [Instance state](#) [Actions](#) [Launch instances](#)

Find Instance by attribute or tag (case-sensitive)

All states

Instance ID = i-0ba8c41833c7c1f5b [X](#) [Clear filters](#)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
AD4	i-0ba8c41833c7c1f5b	Running	t2.medium	Initializing	View alarms +	us-east-1d	ec2-3-85-1

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

Instance ID
 [i-0ba8c41833c7c1f5b \(AD4\)](#)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is dev.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
 chmod 400 "dev.pem"
4. Connect to your instance using its Public DNS:
 [ec2-3-85-103-198.compute-1.amazonaws.com](#)

Example:
 ssh -i "dev.pem" ec2-user@ec2-3-85-103-198.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.

Step 3: Now open the folder in the terminal where our .pem key is stored and paste the command

Step 4: Run the below commands to install and setup Docker.

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg >
```

```
/dev/null sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
```

```
ubuntu@ip-172-31-86-43:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee
/etc/apt/trusted.gpg.d/docker.gpg > /dev/null
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable"
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBFit2ioBEADhWpZ8/wvZ6hUTiX0wQHXMAlaFHcPH9hAtr4F1y2+0YdbtMuth
lqqwp028AqyY+PRfVmSYMbjuQuu5byyKR01BbqYhuS3jtqQmljZ/bJvXqmniVXh
38UuLa+z077PxxyQhu5BbqntTPQMfiyqEiU+BKbq2WmANUKQf+1AmZY/IruOxbnq
L4C1+gJ8vfmXQt99npCaxEjaNRVYfOS8QcixNzHUYnb6emjlAnyEVlZzeqq7Xkl7
UrwV5inawTSzMNvtjEjj4nJL8NsLwscpLPQUhTQ+7BbQXAwAmeHCUTQIvvWXqw0N
cmhh4HgeQscQHYg0JjjDVfoY5MucvgIgCqfzAHW9jxmRL4qbMZj+b1XoePEht
ku4bIQN1X5P07fNWzlgARL5Z4POXDDZTLIQ/El58j9kp4bnWRCJW0ly+a+f8ocodo
vZZ+Doi+fy4D5ZGrL4XEcIQP/Lv5uFyf+kQtL/94VFYVJ0leAv8W92KdgDkhTcTD
G7c0tIkVEKNUq48b3aQ64NOZQW7FvjfoKwEZd0qPE72Pa45jrZzvUFxSpdiNk2tZ
```

```
sudo apt-get update
```

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

```
ubuntu@ip-172-31-86-43:~$ sudo apt-get update
sudo apt-get install -y docker-ce
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

```
sudo mkdir -p /etc/docker
```

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

```
no virtual guests are running on this host
ubuntu@ip-172-31-86-43:~$ sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
```

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

```
ubuntu@ip-172-31-86-43:~$ sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
```

Step 5: Run the below command to install Kubernets.

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

```
ubuntu@ip-172-31-86-43:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o
/etc/apt/keyrings/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list
gpg: missing argument for option "-o"
-bash: /etc/apt/keyrings/kubernetes-apt-keyring.gpg: No such file or directory
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
```

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

```
ubuntu@ip-172-31-86-43:~$ sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 https://security.ubuntu.com/ubuntu noble-security InRelease
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb InRelease [1186 B]
Err:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb InRelease
  The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 234654DA9A296436
Reading package lists... Done
```

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

```
ubuntu@ip-172-31-86-43:~$ sudo systemctl enable --now kubelet
sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
W1003 13:11:39.162569    5084 checks.go:1080] [preflight] WARNING: Couldn't create the interface used for talking to the
container runtime: failed to create new CRI runtime service: validate service connection: validate CRI v1 runtime API f
or endpoint "unix:///var/run/containerd/containerd.sock": rpc error: code = Unimplemented desc = unknown service runtime
.v1.RuntimeService
        [WARNING FileExisting-socat]: socat not found in system path
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
```

TO fix the error:

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

```
ubuntu@ip-172-31-86-43:~$ sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz
  slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 6 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 M]
```

sudo mkdir -p /etc/containerd

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

```
no VM guests are running, validated hypervisor compatibility binaries on this host.
ubuntu@ip-172-31-86-43:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

[debug]
  address = ""
  format = ""
```

sudo systemctl restart containerd

sudo systemctl enable containerd

sudo systemctl status containerd

```
ubuntu@ip-172-31-86-43:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd
● containerd.service - containerd container runtime
   Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)
     Active: active (running) since Thu 2024-10-03 13:12:51 UTC; 276ms ago
       Docs: https://containerd.io
   Main PID: 5462 (containerd)
      Tasks: 8
     Memory: 13.5M (peak: 14.1M)
        CPU: 54ms
      CGroup: /system.slice/containerd.service
              └─5462 /usr/bin/containerd

Oct 03 13:12:51 ip-172-31-86-43 containerd[5462]: time="2024-10-03T13:12:51.543254591Z" level=info msg="serving...
Oct 03 13:12:51 ip-172-31-86-43 containerd[5462]: time="2024-10-03T13:12:51.543291344Z" level=info msg="serving...
Oct 03 13:12:51 ip-172-31-86-43 containerd[5462]: time="2024-10-03T13:12:51.543326951Z" level=info msg="Start subscribi...
Oct 03 13:12:51 ip-172-31-86-43 containerd[5462]: time="2024-10-03T13:12:51.543358179Z" level=info msg="Start recoverin...
Oct 03 13:12:51 ip-172-31-86-43 containerd[5462]: time="2024-10-03T13:12:51.543399144Z" level=info msg="Start event mon...
Oct 03 13:12:51 ip-172-31-86-43 containerd[5462]: time="2024-10-03T13:12:51.543407458Z" level=info msg="Start snapshots...
Oct 03 13:12:51 ip-172-31-86-43 containerd[5462]: time="2024-10-03T13:12:51.543414974Z" level=info msg="Start cni netwo...
Oct 03 13:12:51 ip-172-31-86-43 containerd[5462]: time="2024-10-03T13:12:51.543422043Z" level=info msg="Start streaming...
Oct 03 13:12:51 ip-172-31-86-43 containerd[5462]: time="2024-10-03T13:12:51.543466839Z" level=info msg="containerd succ...
Oct 03 13:12:51 ip-172-31-86-43 systemd[1]: Started containerd.service - containerd container runtime.
```

sudo apt-get install -y socat

```
ubuntu@ip-172-31-86-43:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz
    slirp4netns
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 6 not upgraded.
Need to get 374 kB of additional disk space.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.0.0-4build3 [374 kB]
Fetched 374 kB in 0s (13.9 MB/s)
Selecting previously unselected package socat.
```

Step 6: Initialize the Kubecluster sudo kubeadm init --pod-network-cidr=10.244.0.0/16

```
ubuntu@ip-172-31-86-43:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[kinit] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W1003 13:13:28.336900    5640 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that used by kubeadm. It is recommended to use "registry.k8s.io/pause:3.10" as the CRI sandbox image.
```

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

mkdir -p \$HOME/.kube

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

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

```
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 172.31.86.43:6443 --token oy5zuy.78xd696vk5gr8ip8 \
    --discovery-token-ca-cert-hash sha256:cb705f4200aa1d0a890b60cedb8802c8512842a85be9e9fd527799b09995c9ca
ubuntu@ip-172-31-86-43:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

kubectl apply -f

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

```
ubuntu@ip-172-31-86-43:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
```

Step 7: we can deploy our nginx server on this cluster.

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

```
ubuntu@ip-172-31-86-43:~$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
```

kubectl get pods

NAME	READY	STATUS	RESTARTS	AGE
nginx-deployment-d556bf558-52k84	0/1	Pending	0	17s
nginx-deployment-d556bf558-d5fv5	0/1	Pending	0	17s

```
ubuntu@ip-172-31-86-43:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath=".items[0].metadata.name")
kubectl port-forward $POD_NAME 8080:80
error: unable to forward port because pod is not running. Current status=Pending
```

Note : To fix the error:

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

```
ubuntu@ip-172-31-86-43:~$ kubectl taint nodes ip-172-31-86-43 node-role.kubernetes.io/control-plane:NoSchedule-
node/ip-172-31-86-43 untainted
ubuntu@ip-172-31-86-43:~$ kubectl get nodes
NAME           STATUS   ROLES      AGE     VERSION
ip-172-31-86-43 Ready    control-plane   4m30s   v1.31.1
```

kubectl get pods

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

```
ubuntu@ip-172-31-86-43:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-52k84  1/1     Running   0          3m51s
nginx-deployment-d556bf558-d5fv5  1/1     Running   0          3m51s
ubuntu@ip-172-31-86-43:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath=".items[0].metadata.name")
kubectl port-forward $POD_NAME 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
```

Step 8: Verify your deployment Open up a new terminal and ssh to your EC2 instance. Then, use this curl command to check if the Nginx server is running.

Expanded Security Maintenance for Applications is not enabled.

6 updates can be applied immediately.

5 of these updates are standard security updates.

To see these additional updates run: apt list --upgradable

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

```
Last login: Thu Oct  3 13:05:19 2024 from 110.226.182.217
ubuntu@ip-172-31-86-43:~$ curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Thu, 03 Oct 2024 13:19:50 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 04 Dec 2018 14:44:49 GMT
Connection: keep-alive
ETag: "5c0692e1-264"
Accept-Ranges: bytes

ubuntu@ip-172-31-86-43:~$ |
```

Conclusion:

In this experiment, we,

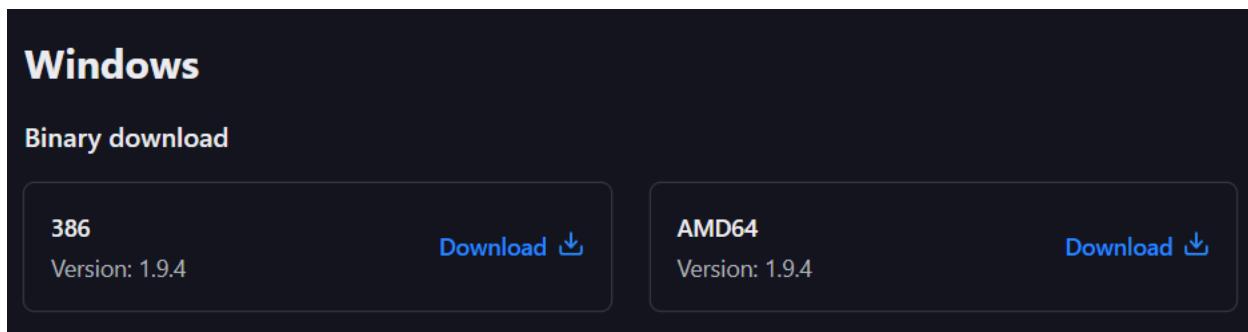
- Successfully set up a Kubernetes cluster on an AWS EC2 instance using kubectl.
- Deployed a sample nginx application to the cluster.

- Installed Docker and Kubernetes, and configured containerd to ensure proper operations.
- Initialized the Kubernetes cluster to facilitate application deployment.
- Verified the successful deployment of the nginx server using kubectl port-forward.
- Confirmed access to the application via curl.
- Gained practical experience in managing Kubernetes clusters and deploying applications on cloud infrastructure.

Experiment 5

Step 1: Download terraform

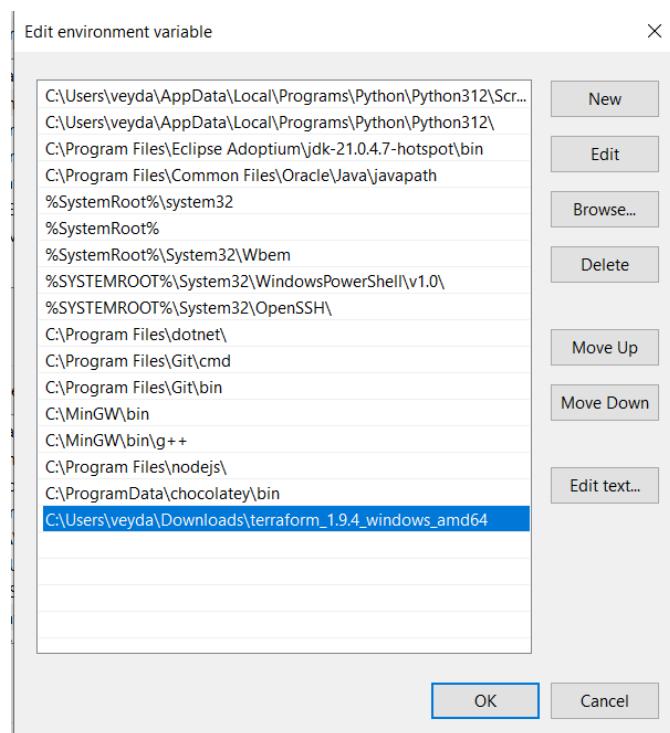
To install Terraform, First Download the Terraform Cli Utility for windows from terraforms official website



Step 2: Extract the files from the download.

Nam

Step 3: Add terraform path to the environment variable



Step 4: Check if terraform was installed by going to the command prompt and type 'terraform -v'.

```
Command Prompt
Microsoft Windows [Version 10.0.19045.4651]
(c) Microsoft Corporation. All rights reserved.

C:\Users\veyda>terraform -v
Terraform v1.9.4
on windows_amd64
```

Step 5: Open Powershell and check terraform functionality.

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

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

PS C:\Users\veyda> terraform
Usage: terraform [global options] <subcommand> [args]

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

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

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

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

Experiment 6

A. Creating docker image using terraform

Prerequisite:

Step 1: Download and Install Docker Desktop from <https://www.docker.com/>
Check the docker functionality

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

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

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

```
Command Prompt
Microsoft Windows [Version 10.0.19045.4780]
(c) Microsoft Corporation. All rights reserved.

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

```
Microsoft Windows [Version 10.0.19045.4780]
(c) Microsoft Corporation. All rights reserved.

C:\Users\veyda>docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
31e907dcc94a: Pull complete
Digest: sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e34a9ab63ee
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest

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

C:\Users\veyda>docker run -it ubuntu
root@0ea95fa5b779:/#
root@0ea95fa5b779:/#
```

Step 2: Create a folder called terra (do not use the word terraform or scripts as that may be a keyword.)and then create a folder called docker inside it.

Then create a file called docker.tf



```
  docker.tf  X
  docker.tf
  1 ~ terraform {
  2   required_providers {
  3     docker = {
  4       source  = "kreuzwerker/docker"
  5       version = "2.21.0"
  6     }
  7   }
  8 }
  9
 10 provider "docker" {
 11   host = "npipe:///./pipe/docker_engine"
 12 }
 13
 14 # Pull the image
 15 resource "docker_image" "ubuntu" {
 16   name = "ubuntu:latest"
 17 }
 18
 19 # Create a container
 20 resource "docker_container" "foo" {
 21   image = docker_image.ubuntu.image_id
 22   name  = "foo"
 23   command = ["sleep", "3600"]
 24 }
```

Step 3: Go to powershell, switch directory to where the docker.tf script was saved, then run the terraform init command.

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

Terraform has been successfully initialized!

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

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

Step 4: Execute terraform plan to view resources.

```
PS C:\Users\veyda\Desktop\terra\docker> terraform plan

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

Terraform will perform the following actions:

# docker_container.foo will be created
+ resource "docker_container" "foo" {
  + attach          = false
  + bridge          = (known after apply)
  + command         = [
    + "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)
  + ipc_mode        = (known after apply)
  + log_driver      = (known after apply)
  + logs            = false
  + must_run        = true
  + name            = "foo"
  + network_data   = (known after apply)
  + read_only       = false
  + remove_volumes = true
  + restart         = "no"
  + rm              = false
  + runtime         = (known after apply)
  + security_opts  = (known after apply)
  + shm_size        = (known after apply)
  + start           = true
  + std_in_open     = false
  + stop_signal     = (known after apply)
  + stop_timeout    = (known after apply)
  + tty              = false
  + healthcheck     = (known after apply)
}
```

Step 5: terraform show command for viewing all the state files

```
PS C:\Users\veyda\Desktop\terra\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        = "dfaafae11b3bf9d4a108b967b3c24943543a9b381fc75576cc06cd4fce85ee72"
  id              = "dfaafae11b3bf9d4a108b967b3c24943543a9b381fc75576cc06cd4fce85ee72"
  image           = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a"
  init            = false
  ip_address       = "172.17.0.2"
  ip_prefix_length = 16
  ipc_mode         = "private"
  log_driver       = "json-file"
  logs            = false
  max_retry_count = 0
  memory          = 0
  memory_swap      = 0
  must_run         = true
  name            = "foo"
  network_data    = [
    {
      gateway      = "172.17.0.1"
      global_ipv6_address = null
      global_ipv6_prefix_length = 0
      ip_address    = "172.17.0.2"
      ip_prefix_length = 16
      ipv6_gateway  = null
      network_name  = "bridge"
    },
  ],
  network_mode     = "bridge"
  pid_mode         = null
  privileged       = false
  publish_all_ports = false
  read_only        = false
  remove_volumes   = true
  restart          = "no"
  rm              = false
  runtime          = "runc"
  security_opts    = []
  shm_size         = 64
}
```

Step 6: 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”

Docker images before apply step:

```
PS C:\Users\veyda\Desktop\terra\docker> docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
PS C:\Users\veyda\Desktop\terra\docker>
```

Docker images after apply step:

```
PS C:\Users\veyda\Desktop\terra\docker> docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
ubuntu          latest    edbfe74c41f8  3 weeks ago  78.1MB
PS C:\Users\veyda\Desktop\terra\docker>
```

Step 7: terraform refresh command is helpful to sync your state file with the real-world infrastructure without applying any changes.

```
PS C:\Users\veyda\Desktop\terra\docker> terraform refresh
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=dfaaafae11b3bf9d4a108b967b3c24943543a9b381fc75576cc06cd4fce85ee72]
PS C:\Users\veyda\Desktop\terra\docker>
```

Step 8: Execute Terraform destroy to delete the configuration, which will automatically delete the Ubuntu Container.

```
PS C:\Users\veyda\Desktop\terra\docker> terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=dfaaafae11b3bf9d4a108b967b3c24943543a9b381fc75576cc06cd4fce85ee72]

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",
  ]}
```

Docker images After Executing Destroy step

```
Destroy complete! Resources: 2 destroyed.
PS C:\Users\veyda\Desktop\terra\docker> docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
PS C:\Users\veyda\Desktop\terra\docker>
```


Experiment 7

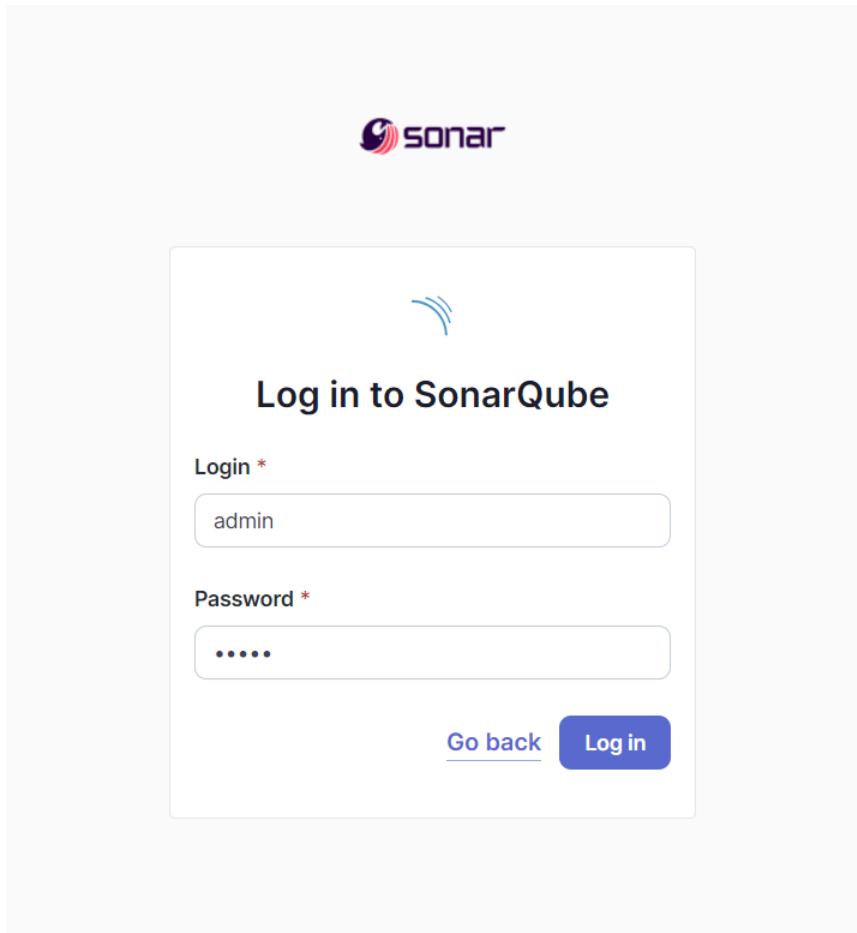
Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

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

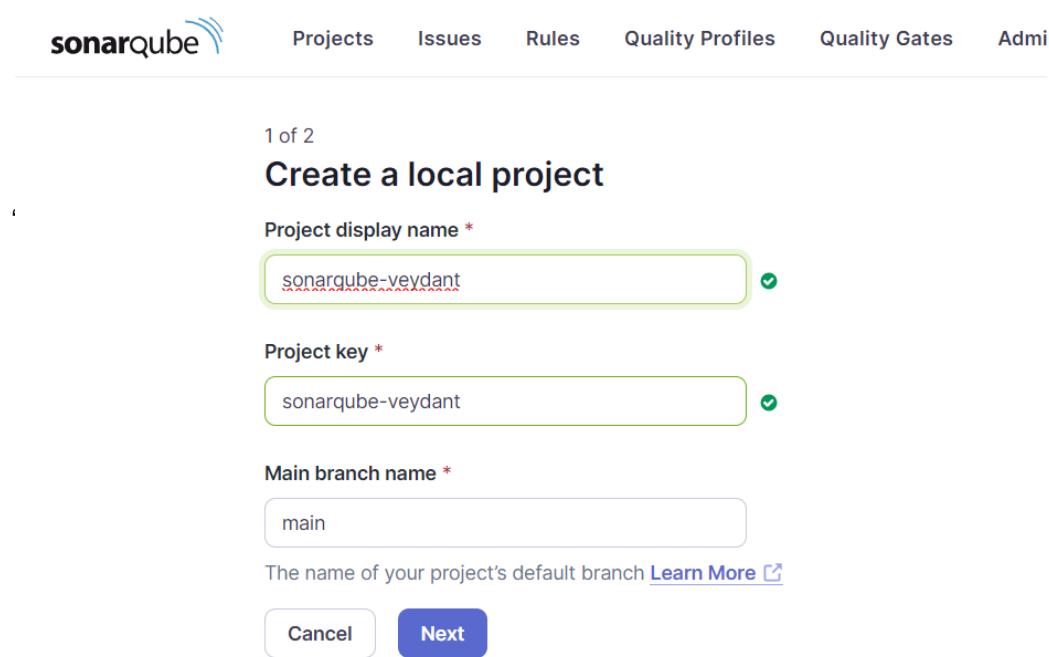
2. Run SonarQube in a Docker container using this command -

```
C:\Windows\system32>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
st
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
b6dd73afc810a20ec3d643e9a148ab9643a3b5beff2766406df21f5f54a090c1
```

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



1 of 2

Create a local project

Project display name *

 ✓

Project key *

 ✓

Main branch name *

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

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

Setup the project and come back to Jenkins Dashboard.

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

SonarQube Scanner for Jenkins 2.17.2

This plugin allows an easy integration of [SonarQube](#), the open source platform for Continuous Inspection of code quality.

[Report an issue with this plugin](#)



6. Under Jenkins 'Configure System', look for SonarQube Servers and enter the details. Enter the Server Authentication token if needed.

7. Search for SonarQube Scanner under Global Tool Configuration. Choose the

latest configuration and choose Install automatically.

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

Source Code Management

None

Git [?](#)

Repositories [?](#)

Repository URL [?](#)

Credentials [?](#)

- none -

[+ Add](#)

[Advanced](#)

[Add Repository](#)

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 and allow Execute Permissions to the Admin user.

Execute SonarQube Scanner

JDK [?](#)

JDK to be used for this SonarQube analysis

Path to project properties [?](#)

Analysis properties [?](#)

```
sonar.host.url=http://localhost:9000
sonar.projectKey=sonarqube-veydant
sonar.projectName=sonarqube-veydant
sonar.projectVersion=1.0
sonar.sources=
sonar.login=squ_3c67bb5196ad7f467c5a12b65dfb090e8769e1c
```

12. Run the build and check the output

Console Output

[Download](#)
[Copy](#)
[View as plain text](#)

```

Started by user unknown or anonymous
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube-test
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube-test\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject
> git.exe --version # timeout=10
> git --version # 'git version 2.45.2.windows.1'
> git.exe fetch --tags --force -- https://github.com/shazforiot/MSBuild_firstproject +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
Commit message: "updated"
> git.exe rev-list --no-walk f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
[sonarqube-test] $ C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube\bin\sonar-scanner.bat -Dsonar.host.url=http://localhost:9000
-Dsonar.projectKey=sonarqube-veydant -Dsonar.projectName=sonarqube-veydant -Dsonar.host.url=http://localhost:9000 -Dsonar.login=squ_3c67bbd5196ad7f467c5a12b65dfb090e8769e1c
-Dsonar.projectVersion=1.0 -Dsonar.sources=. -Dsonar.projectBaseDir=C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube-test
21:42:11.084 WARN Property 'sonar.host.url' with value 'http://localhost:9000' is overridden with value 'http://localhost:9000'
21:42:11.111 INFO Scanner configuration file: C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube\bin..\conf\sonar-scanner.properties
21:42:11.114 INFO Project root configuration file: NONE
21:42:11.182 INFO SonarScanner CLI 6.2.0.4584
21:42:11.186 INFO Java 21.0.4 Eclipse Adoptium (64-bit)
21:42:11.195 INFO Windows 10 10.0 amd64
21:42:11.257 INFO User cache: C:\Windows\system32\config\systemprofile\.sonar\cache
21:42:12.960 INFO JRE provisioning: os[windows], arch[amd64]
21:42:25.421 INFO Communicating with SonarQube Server 10.6.0.92116
21:42:27.194 INFO Starting SonarScanner Engine...
21:42:27.196 INFO Java 17.0.11 Eclipse Adoptium (64-bit)
21:42:30.344 INFO [main] INFO sonarqube scanner 6.2.0.4584

```

13. Once the build is complete check on sonarqube

The screenshot shows the SonarQube interface for the 'main' project. At the top, there's a navigation bar with links for Overview, Issues, Security Hotspots, Measures, Code, and Activity. On the right, there are Project Settings and Project Information options.

The main content area displays the following metrics:

- Quality Gate:** Passed (green checkmark icon)
- Security:** 0 Open issues (A grade)
- Reliability:** 0 Open issues (A grade)
- Maintainability:** 0 Open issues (A grade)
- Accepted issues:** 0 (G grade)
- Coverage:** On 0 lines to cover.
- Duplications:** 0.0% (A grade)
- Security Hotspots:** 0 (A grade)

A yellow warning box at the bottom left states: "⚠️ The last analysis has warnings. See details".

Conclusion

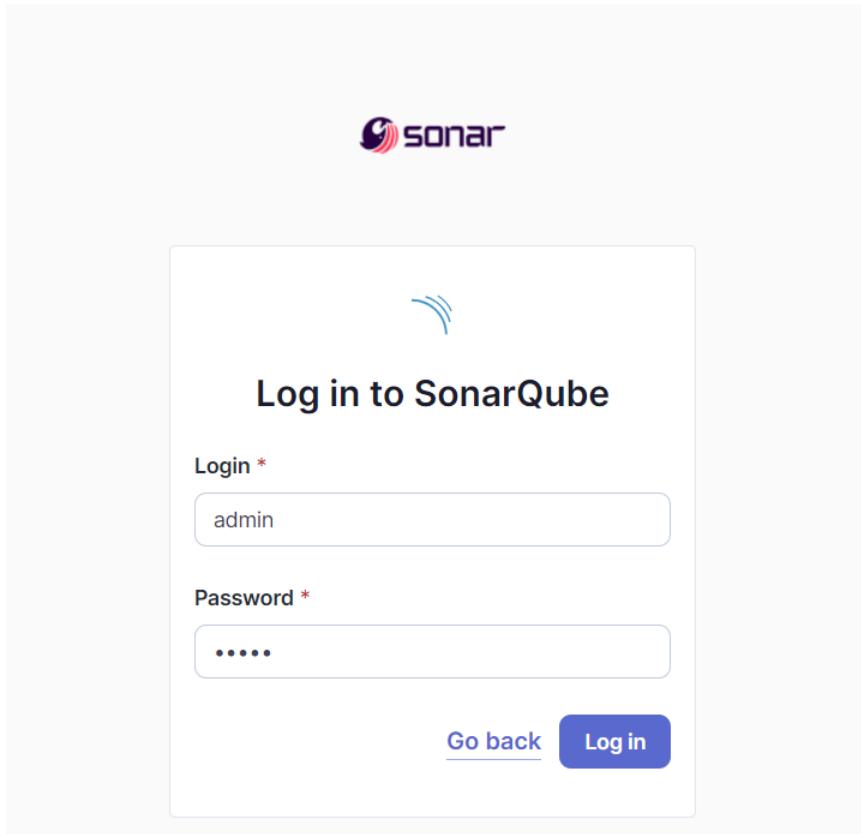
- In this experiment we worked on the sonarqube project along with jenkins.
- Project Build step issues: Issues faced were due to permissions from sonarqube project.
- The steps involved logging into SonarQube, creating a project, and configuring necessary settings within Jenkins to facilitate automated analysis of our sample GitHub repository. This integration not only enhances our ability to identify vulnerabilities early in the development lifecycle but also promotes a culture of security within our development practices.
- By integrating Jenkins with SonarQube, we established an automated framework for continuous static analysis, enhancing our CI/CD pipeline's security posture.

Experiment 8

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

```
C:\Windows\system32>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:late
st
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9fecc71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
b6dd73afc810a20ec3d643e9a148ab9643a3b5beff2766406df21f5f54a090c1
```

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



4. Create a new project

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

5. Click on Security and create a new name and token, select type as user token

6. Create a new pipeline

The screenshot shows the Jenkins 'New Item' creation dialog. At the top, it says 'Dashboard > All > New Item'. The title is 'New Item'. A search bar contains 'newpipe'. Below it, a section titled 'Select an item type' lists several options:

- Freestyle project**: Described as a classic, general-purpose job type.
- Maven project**: Described as building a Maven project.
- Pipeline**: Described as orchestrating long-running activities across multiple build agents.
- Multi-configuration project**: Described as suitable for projects with many configurations.
- Folder**: Described as creating a container for nested items.

At the bottom of the dialog is an 'OK' button.

7. Go to sonarqube website and download the SonarScanner CLI

SonarScanner CLI

SonarScanner | Issue Tracker Show more ▾

6.2 2024-09-17

Support PKCS12 truststore generated with OpenSSL

Download scanner for: [Linux x64](#) [Linux AArch64](#) [Windows x64](#) [macOS x64](#) [macOS AArch64](#) [Docker Any \(Requires a pre-installed JVM\)](#)

[Release notes](#)

Once it is downloaded, copy its path and add to the script

8. Inside the pipeline section add the following script

```
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/"
}
}
}
```

Pipeline

Definition

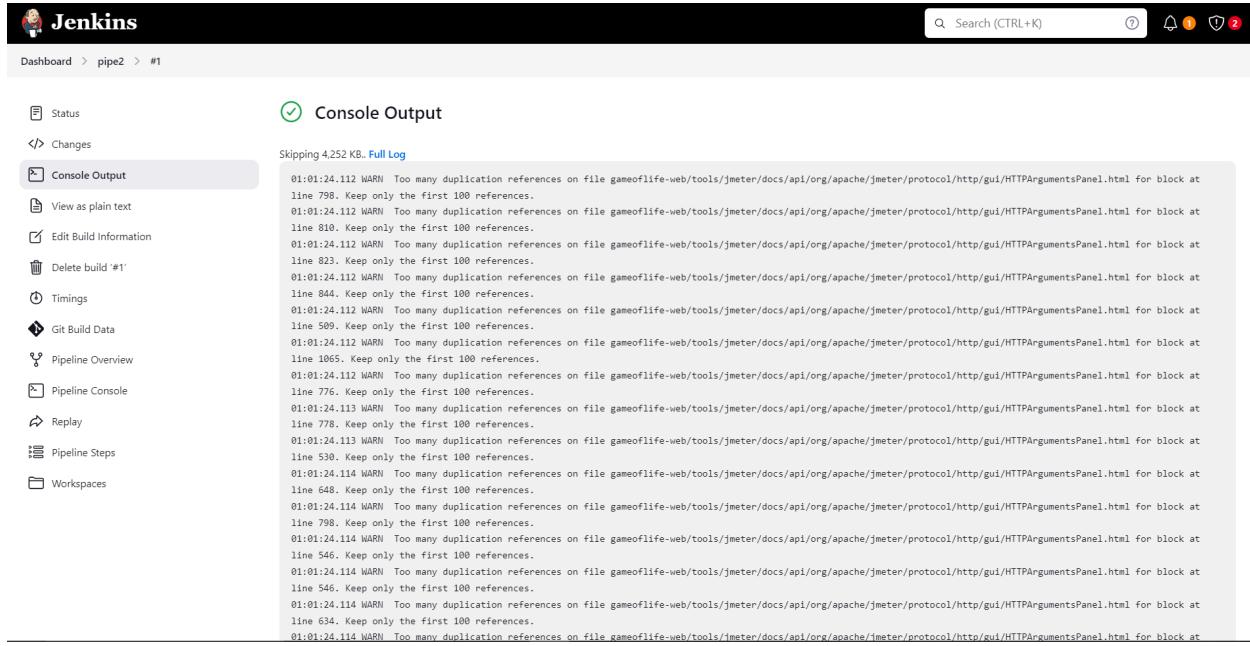
Pipeline script

```
Script ?
```

```
1 node {
2   stage('Cloning the GitHub Repo') {
3     git 'https://github.com/shazforiot/GOL.git'
4   }
5   stage('SonarQube analysis') {
6     withSonarQubeEnv('sonarqube-test') { // Use the correct name here
7       bat """
8         cd %SystemRoot%\Temp\sonar\sonar-scanner-6.2.0.4584-windows-x64\bin\sonar-scanner.bat \
9         -DSonar.login=squ_5f856d9ac21503b7232d5c635fc6363e9b4ddfd \
10        -DSonar.projectKey=sonarqube-test \
11        -DSonar.exclusions=vendor/**,resources/**,**/*.java \
12        -DSonar.host.url=http://localhost:9000/
13        """
14     }
15   }
16 }
```

Use Groovy Sandbox ?

9. Save and build the pipeline, this step may take a while. After the build is successful, check the console output

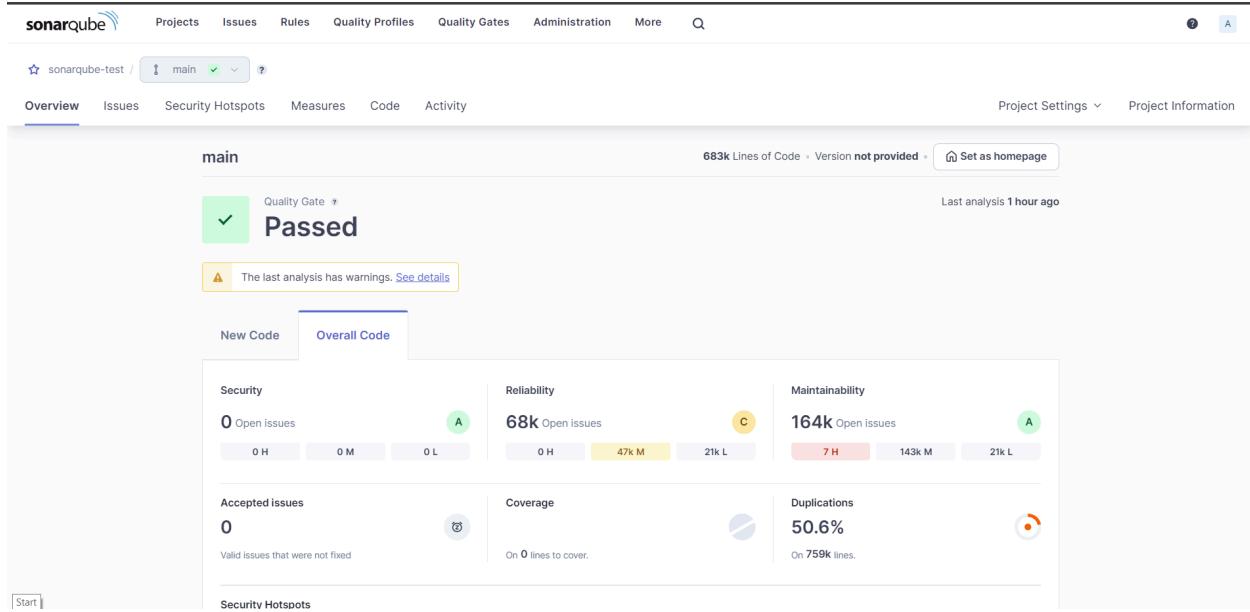


```

Skipping 4.252 KB. Full Log
01:01:24.112 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 798. Keep only the first 100 references.
01:01:24.112 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 810. Keep only the first 100 references.
01:01:24.112 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 823. Keep only the first 100 references.
01:01:24.112 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 844. Keep only the first 100 references.
01:01:24.112 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 509. Keep only the first 100 references.
01:01:24.112 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 1065. Keep only the first 100 references.
01:01:24.112 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 776. Keep only the first 100 references.
01:01:24.113 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 778. Keep only the first 100 references.
01:01:24.113 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 530. Keep only the first 100 references.
01:01:24.114 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 648. Keep only the first 100 references.
01:01:24.114 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 798. Keep only the first 100 references.
01:01:24.114 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 546. Keep only the first 100 references.
01:01:24.114 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 546. Keep only the first 100 references.
01:01:24.114 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 634. Keep only the first 100 references.
01:01:24.114 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 798. Keep only the first 100 references.

```

10. Check the sonarqube project, whether it has passed or not.



The SonarQube dashboard for the 'sonarqube-test' project shows the following details:

- Project Status:** Passed (green checkmark)
- Analysis Date:** Last analysis 1 hour ago
- Code Metrics:**
 - 683k Lines of Code - Version not provided
 - Set as homepage button
- Overall Quality Gate:** Passed
- Overall Code Health:** Overall Code tab selected
- Security:**
 - 0 Open issues (0 H, 0 M, 0 L)
- Reliability:**
 - 68k Open issues (0 H, 47k M, 21k L)
- Maintainability:**
 - 164k Open Issues (7 H, 143k M, 21k L)
- Accepted issues:** 0 (Valid issues that were not fixed)
- Coverage:** On 0 lines to cover.
- Duplications:** 50.6% (On 759k lines)

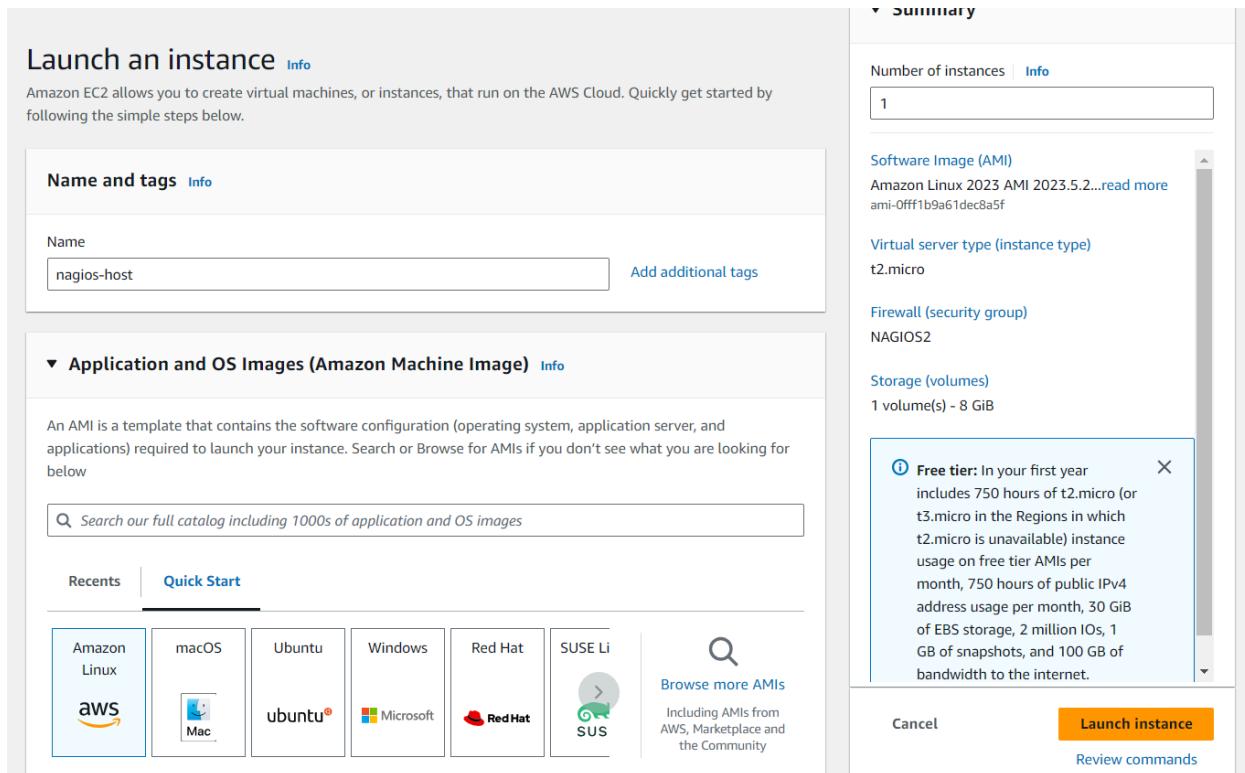
Conclusion:

- We created a sonarqube project locally
- We created a jenkins pipeline and added a script with github link
- We also added sonarqube project details like key, token etc.
- Many issues were faced with the script as the build was initially failing, later succeeded with some changes.
- The build was successful on jenkins and passed on sonarqube

Experiment 9

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

Step 1: Create an ec2 instance, select amazon linux, t2.micro , select a key name and launch instance.



Step 2 : Connect to the instance using the SSH client, and copy and paste the command on the terminal

```
ec2-user@ip-172-31-33-100:~  
microsoft Windows [Version 10.0.19045.4894]  
(c) Microsoft Corporation. All rights reserved.  
C:\Users\veyda>cd Downloads  
C:\Users\veyda\Downloads>ssh -i "nagios.pem" ec2-user@ec2-174-129-105-123.compute-1.amazonaws.com  
The authenticity of host 'ec2-174-129-105-123.compute-1.amazonaws.com (174.129.105.123)' can't be established.  
ECDSA key fingerprint is SHA256:ToI7CEUlcDMUEI7aeXSD8JcW8Vy/MienkGGAp4RR7jc.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added 'ec2-174-129-105-123.compute-1.amazonaws.com,174.129.105.123' (ECDSA) to the list of known hosts.  
, #_  
~\ ####_ Amazon Linux 2023  
~~ \####\   
~~ \###|  
~~ \#/ ____ https://aws.amazon.com/linux/amazon-linux-2023  
~~ V~' '-'>  
~~ ._. /  
~~ / /  
~/m/'  
[ec2-user@ip-172-31-33-100 ~]$
```

Step 3: First, run the following command:- sudo yum update This command will check for any updates for the YUM library.

```
./m/
[ec2-user@ip-172-31-35-21 ~]$ sudo yum update
Last metadata expiration check: 0:02:03 ago on Tue Oct  8 05:47:23 2024.
Dependencies resolved.
Nothing to do.
Complete!
```

Step 4: Run the command: sudo yum install httpd php This installs an Apache server and a PHP on your instance.

```
[ec2-user@ip-172-31-35-21 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:02:58 ago on Tue Oct  8 05:47:23 2024.
Dependencies resolved.
=====
Package           Arch      Version       Repository      Size
=====
Installing:
httpd            x86_64    2.4.62-1.amzn2023   amazonlinux    48 k
php8.3           x86_64    8.3.10-1.amzn2023.0.1  amazonlinux   10 k
Installing dependencies:
apr               x86_64    1.7.2-2.amzn2023.0.2  amazonlinux   129 k
apr-util          x86_64    1.6.3-1.amzn2023.0.1  amazonlinux   98 k
generic-logos-httpd  noarch   18.0.0-12.amzn2023.0.3  amazonlinux   19 k
httpd-core        x86_64    2.4.62-1.amzn2023   amazonlinux   1.4 M
httpd-fs           noarch   2.4.62-1.amzn2023   amazonlinux   14 k
```

Step 5: Run the command: sudo yum install gcc glibc glibc-common This installs the C/C++ compiler (GCC) along with the necessary C libraries required for compiling and running C programs.

```
[ec2-user@ip-172-31-35-21 ~]$ sudo yum install gcc glibc glibc-common
Last metadata expiration check: 0:03:21 ago on Tue Oct  8 05:47:23 2024.
Package glibc-2.34-52.amzn2023.0.11.x86_64 is already installed.
Package glibc-common-2.34-52.amzn2023.0.11.x86_64 is already installed.
Dependencies resolved.
=====
Package           Arch      Version       Repository      Size
=====
Installing:
gcc              x86_64    11.4.1-2.amzn2023.0.2  amazonlinux   32 M
Installing dependencies:
annobin-docs      noarch   10.93-1.amzn2023.0.1  amazonlinux   92 k
annobin-plugin-gcc x86_64    10.93-1.amzn2023.0.1  amazonlinux   887 k
cpp              x86_64    11.4.1-2.amzn2023.0.2  amazonlinux   10 M
gc               x86_64    8.0.4-5.amzn2023.0.2  amazonlinux   105 k
```

Step 6: Run the command: sudo yum install gd gd-devel

```
[ec2-user@ip-172-31-35-21 ~]$ sudo yum install gd gd-devel
Last metadata expiration check: 0:03:44 ago on Tue Oct  8 05:47:23 2024.
Dependencies resolved.
=====
Package           Arch      Version       Repository      Size
=====
Installing:
gd               x86_64    2.3.3-5.amzn2023.0.3  amazonlinux   139 k
gd-devel         x86_64    2.3.3-5.amzn2023.0.3  amazonlinux   38 k
Installing dependencies:
brotli           x86_64    1.0.9-4.amzn2023.0.2  amazonlinux   314 k
brotli-devel     x86_64    1.0.9-4.amzn2023.0.2  amazonlinux   31 k
bzzip2-devel     x86_64    1.0.8-6.amzn2023.0.2  amazonlinux   214 k
cairo             x86_64    1.17.6-2.amzn2023.0.1  amazonlinux   684 k
cmake-filesystem x86_64    3.22.2-1.amzn2023.0.4  amazonlinux   16 k
```

Step 7: Run the commands: sudo adduser -m nagios sudo passwd nagios

```
[ec2-user@ip-172-31-35-21 ~]$ sudo adduser -m nagios
[ec2-user@ip-172-31-35-21 ~]$ sudo passwd -m nagios
passwd: bad argument -m: unknown option
[ec2-user@ip-172-31-35-21 ~]$ sudo passwd nagios
Changing password for user nagios.
[ec2-user@ip-172-31-35-21 ~]$ New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
[ec2-user@ip-172-31-35-21 ~]$ sudo usermod -a -G nagcmd nagios
[ec2-user@ip-172-31-35-21 ~]$ sudo usermod -a -G nagcmd apache
```

Step 10: mkdir ~/downloads cd ~/downloads This creates a directory named 'downloads', to store the files of the nagios server that are downloaded

```
[ec2-user@ip-172-31-35-21 ~]$ : mkdir ~/downloads
[ec2-user@ip-172-31-35-21 ~]$ cd ~/downloads
```

Step 11: wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz The above command installs the latest version of nagios-core

Step 12: wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz The above command installs the latest version of nagios-plugins.

```
[ec2-user@ip-172-31-35-21 downloads]$ wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
--2024-10-09 05:42:49-- https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2600:3c00::f03c:92ff:fef7:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: 'nagios-4.5.5.tar.gz.1'

nagios-4.5.5.tar.gz.1      100%[=====]  1.97M  5.56MB/s   in 0.4s
2024-10-09 05:42:50 (5.56 MB/s) - 'nagios-4.5.5.tar.gz.1' saved [2065473/2065473]

[ec2-user@ip-172-31-35-21 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
--2024-10-09 05:43:04-- 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.1'

nagios-plugins-2.4.11.tar.gz.1 100%[=====]  2.62M  5.49MB/s   in 0.5s
2024-10-09 05:43:05 (5.49 MB/s) - 'nagios-plugins-2.4.11.tar.gz.1' saved [2753049/2753049]
```

Step 13:

tar zxvf nagios-4.5.5.tar.gz This extracts the nagios-core files into the same directory using the tar command.

Use 'ls' command to find the correct directory

run the 'sudo yum install openssl-devel' command.

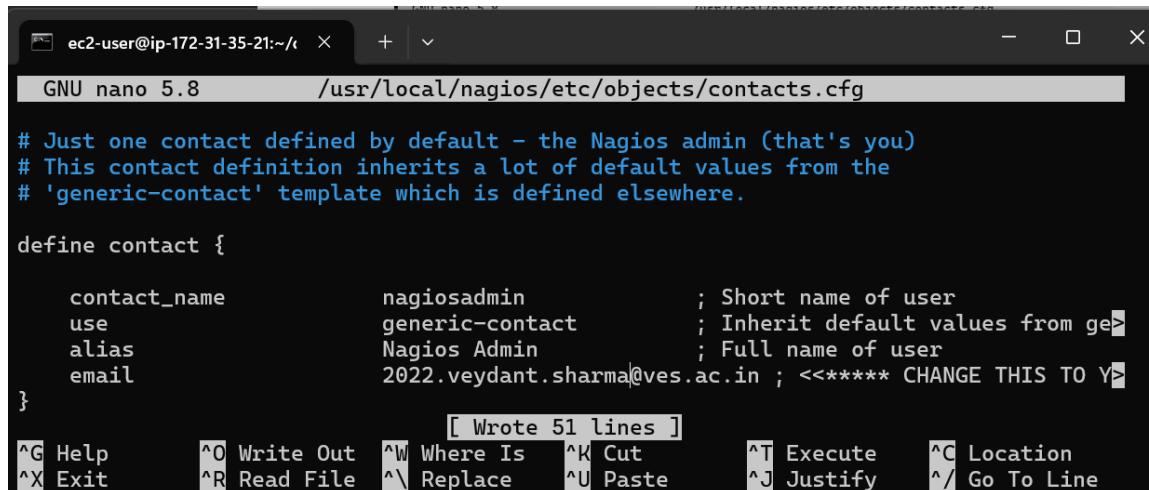
```
[ec2-user@ip-172-31-35-21 downloads]$ ls
nagios-4.5.5  nagios-4.5.5.tar.gz  nagios-4.5.5.tar.gz.1  nagios-plugins-2.4.11.tar.gz
[ec2-user@ip-172-31-35-21 downloads]$ cd nagios-4.5.5
[ec2-user@ip-172-31-35-21 nagios-4.5.5]$ sudo yum install openssl-devel
Last metadata expiration check: 23:58:21 ago on Tue Oct  8 05:47:23 2024.
Package openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-35-21 nagios-4.5.5]$ ./configure --with-command-group=nagcmd
checking for a BSD compatible install... /usr/bin/install -c
```

Step 14: Next, we must compile all components of this software according to the instructions in the Makefile. To do so, use the following command: make all Then, sudo make install sudo make install-init sudo make install-config sudo make install-commandmode

```
[ec2-user@ip-172-31-35-21 nagios-4.5.5]$ make all
cd ./base && make
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
make -C ./lib
```

```
[ec2-user@ip-172-31-35-21 nagios-4.5.5]$ sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode
cd ./base && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagios /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagiosstats /usr/local/nagios/bin
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5/base'
```

Step 15: : We need to update the email linked with this server to our email for it to send notifications (if any needed). sudo nano /usr/local/nagios/etc/objects/contacts.cfg



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

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

define contact {
    contact_name      nagiosadmin           ; Short name of user
    use               generic-contact        ; Inherit default values from ge>
    alias             Nagios Admin          ; Full name of user
    email             2022.veydant.sharma@ves.ac.in ; <<***** CHANGE THIS TO Y>
}
[Wrote 51 lines]
^G Help      ^O Write Out  ^W Where Is  ^K Cut      ^T Execute  ^C Location
^X Exit      ^R Read File   ^\ Replace   ^U Paste    ^J Justify  ^/ Go To Line
```

Step 17: sudo make install-webconf This installs the necessary configuration files for the Nagios web interface.

Step 18: sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin This creates a user named 'nagiosadmin' to access the nagios web interface. Create a password and keep it in mind as it will be required in the future steps.

Step 19: Restart the apache server to apply all the recent configurations. sudo service httpd restart

Step 20: cd ~/downloads tar zxvf nagios-plugins-2.4.11.tar.gz This changes the directory to the 'downloads' directory and extracts the files for nagios-plugins

```
[ec2-user@ip-172-31-35-21 nagios-4.5.5]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios.conf
if [ $? -eq 1 ]; then \
    ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagios.conf; \
fi

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

[ec2-user@ip-172-31-35-21 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-172-31-35-21 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-35-21 nagios-4.5.5]$ cd ~/downloads
[ec2-user@ip-172-31-35-21 downloads]$ tar zxvf nagios-plugins-2.4.11.tar.gz
nagios-plugins-2.4.11/
nagios-plugins-2.4.11/build-aux/
```

Step 21: cd nagios-plugins-2.4.11 ./configure --with-nagios-user=nagios
--with-nagios-group=nagios This installs the configurations for the nagios-plugins files.

```
[ec2-user@ip-172-31-35-21 downloads]$ cd nagios-plugins-2.4.11
[ec2-user@ip-172-31-35-21 nagios-plugins-2.4.11]$ ./configure --with-nagios-user=nagios --with-nagios-group=nagios
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether to enable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
```

Step22:: sudo chkconfig --add nagios
sudo chkconfig nagios on

```
[ec2-user@ip-172-31-35-21 nagios-plugins-2.4.11]$ sudo chkconfig --add nagios
sudo chkconfig nagios on
error reading information on service nagios: No such file or directory
Note: Forwarding request to 'systemctl enable nagios.service'.
Created symlink /etc/systemd/system/multi-user.target.wants/nagios.service → /usr/lib/systemd/system/nagios.service.
[ec2-user@ip-172-31-35-21 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...
```

```
[ec2-user@ip-172-31-35-21 downloads]$ tar zxvf nagios-4.5.5.tar.gz
nagios-4.5.5/
nagios-4.5.5/.github/
nagios-4.5.5/.github/workflows/
nagios-4.5.5/.github/workflows/test.yml
nagios-4.5.5/.gitignore
nagios-4.5.5/CONTRIBUTING.md
```

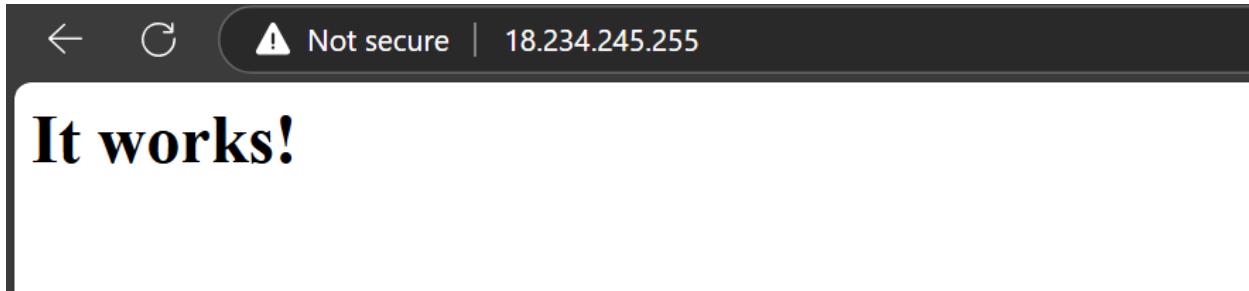
Step 23: sudo service nagios start This starts the Nagios service.

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

```
[ec2-user@ip-172-31-35-21 nagios-plugins-2.4.11]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-35-21 nagios-plugins-2.4.11]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
    Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
    Active: active (running) since Wed 2024-10-09 05:55:32 UTC; 11s ago
      Docs: https://www.nagios.org/documentation
   Process: 21100 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Process: 21101 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
 Main PID: 21102 (nagios)
    Tasks: 6 (limit: 1112)
   Memory: 5.5M
      CPU: 77ms
     CGroup: /system.slice/nagios.service
             ├─21102 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─21103 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─21104 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─21105 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─21106 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─21107 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 09 05:55:32 ip-172-31-35-21.ec2.internal nagios[21102]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully bound
Oct 09 05:55:32 ip-172-31-35-21.ec2.internal nagios[21102]: qh: core query handler registered
Oct 09 05:55:32 ip-172-31-35-21.ec2.internal nagios[21102]: qh: echo service query handler registered
Oct 09 05:55:32 ip-172-31-35-21.ec2.internal nagios[21102]: qh: help for the query handler registered
Oct 09 05:55:32 ip-172-31-35-21.ec2.internal nagios[21102]: wproc: Successfully registered manager as @wproc with query
Oct 09 05:55:32 ip-172-31-35-21.ec2.internal nagios[21102]: wproc: Registry request: name=Core Worker 21105;pid=21105
Oct 09 05:55:32 ip-172-31-35-21.ec2.internal nagios[21102]: wproc: Registry request: name=Core Worker 21104;pid=21104
Oct 09 05:55:32 ip-172-31-35-21.ec2.internal nagios[21102]: wproc: Registry request: name=Core Worker 21103;pid=21103
Oct 09 05:55:32 ip-172-31-35-21.ec2.internal nagios[21102]: wproc: Registry request: name=Core Worker 21106;pid=21106
```

Step 24: In the address bar, enter 'http://<ipv4 address>/nagios'.



A screenshot of the Nagios Core web interface. The top right corner displays the Nagios Core logo and the text 'Nagios® Core™ Version 4.5.5 September 17, 2024 Check for updates'. A green checkmark indicates that the daemon is running with PID 21102. The left sidebar contains navigation links for General, Current Status, Problems, Reports, and System. The main content area includes sections for Get Started, Latest News, and Don't Miss... . The footer contains copyright information and a note about the GNU General Public License.

Conclusion:

- In the above experiment, we learned how to install and configure Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine.
- We faced some issues with the configuration command, which were fixed with the right directory and the sudo yum install openssl-devel' command.
- Once the setup was complete, we hosted the Nagios server and accessed the Nagios dashboard by pasting the public IPv4 address of our instance in the browser

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

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

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

The screenshot shows the AWS EC2 Instances page. It displays two instances: 'Nagios1' and 'ubuntu-client', both of which are running and belong to the 't2.micro' instance type. A green banner at the top of the page indicates a successful initiation of termination for another instance. The left sidebar shows navigation options like EC2 Dashboard, EC2 Global View, Events, Console-to-Code, Instances, Instance Types, and Launch Templates.

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

Step 3: ps -ef | grep nagios Run the above command on the nagios-host instance. This verifies whether the nagios service is running or not.

```
[ec2-user@ip-172-31-35-21 ~]$ ps -ef | grep nagios
nagios    21102      1  0 05:55 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios    21103  21102  0 05:55 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagi
s.qh
nagios    21104  21102  0 05:55 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagi
s.qh
nagios    21105  21102  0 05:55 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagi
s.qh
nagios    21106  21102  0 05:55 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagi
s.qh
nagios    21107  21102  0 05:55 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
root     21110      2900  0 05:55 pts/0   00:00:00 sudo systemctl status nagios
root     21112  21110  0 05:55 pts/1   00:00:00 sudo systemctl status nagios
root     21113  21112  0 05:55 pts/1   00:00:00 systemctl status nagios
ec2-user 22304  22272  0 06:15 pts/2   00:00:00 grep --color=auto nagios
```

Step 4: sudo su mkdir -p /usr/local/nagios/etc/objects/monitorhosts mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts This makes you the root user and creates two folders with the above paths.

```
[ec2-user@ip-172-31-35-21 ~]$ sudo su
mkdir -p /usr/local/nagios/etc/objects/monitorhosts
mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-35-21 ec2-user]#
```

Step 5: We need to create a config file in this folder. So, copy the contents of the existing localhost config to the new file ‘linuxserver.cfg’. cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

```
root@ip-172-31-35-21 ec2-user]# sudo mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/
root@ip-172-31-35-21 ec2-user]# sudo cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monit
rhosts/linuxhosts/linuxserver.cf
```

Step 6: We need to make some changes in this config file. Open it using nano editor:- nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg 1. Change hostname and alias from ‘hostname’ to ‘linuxserver’. 2. Change address to the public ip address of the ubuntu-client instance.

Step 7: Once the files are verified and it is confirmed that there are no errors, we must restart the server. service nagios restart

```
[root@ip-172-31-35-21 ec2-user]# /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
Nagios Core 4.5.5
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2024-09-17
License: GPL

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

Step 8: systemctl status nagios Using the above command, we check the status of the nagios server and ensure that it is active (running).

```
[root@ip-172-31-35-21 ec2-user]# service nagios restart
Redirecting to /bin/systemctl restart nagios.service
[root@ip-172-31-35-21 ec2-user]# systemctl status nagios
● nagios.service - Nagios Core 4.5.5
    Loaded: Loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
    Active: active (running) since Wed 2024-10-09 06:32:00 UTC; 12s ago
      Docs: https://www.nagios.org/documentation
   Process: 23269 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0)
  Process: 23270 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SU>
 Main PID: 23271 (nagios)
    Tasks: 6 (limit: 1112)
   Memory: 5.4M
     CPU: 69ms
    CGroup: /system.slice/nagios.service
            └─23271 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
              ├─23272 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
              ├─23273 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
              ├─23274 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
              ├─23275 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
              └─23276 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
```

Step 9: Connect your ubuntu-client instance to your local terminal using SSH in the same way as you connected the nagios-host instance to your local terminal using SSH

Step 10: On your ubuntu-client instance, run the following commands:- sudo apt update -y sudo apt install gcc -y sudo apt install -y nagios-nrpe-server nagios-plugins The above commands check for any new updates and then install gcc, Nagios NRPE server and Nagios plugins.

```
ubuntu@ip-172-31-94-199:~$ sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [380 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [535 kB]
Get:15 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [82.9 kB]
Get:16 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [560 kB]
```

Step 11: Run the following command: sudo nano /etc/nagios/nrpe.cfg The above command opens the NRPE config file. Here, we need to add the public IP address of our host nagios-host instance to the NRPE configuration file. Under allowed_hosts, add the nagios-host public IPv4 address.

```
# Address that nrpe should bind to in case there are more than one interface
# and you do not want nrpe to bind on all interfaces.
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
18.234.245.255
#server_address=127.0.0.1

# LISTEN QUEUE SIZE
# Listen queue size (backlog) for serving incoming connections.
# You may want to increase this value under high load.
```

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

Host	Status	Last Check	Duration	Status Information
linuxserver	UP	10-09-2024 06:41:22	0d 0h 10m 30s	PING OK - Packet loss = 0%, RTA = 0.03 ms
localhost	UP	10-09-2024 06:39:54	0d 0h 46m 58s	PING OK - Packet loss = 0%, RTA = 0.03 ms

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

Host Information

Last Updated: Wed Oct 9 06:43:10 UTC 2024
Updated every 90 seconds
Nagios® Core™ 4.5.5 - www.nagios.org
Logged in as nagiosadmin

Host
linuxserver
(linuxserver)

Member of
linux-servers1

IP 127.0.0.1

Tactical Overview

- 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
- Process Info

Host Status: UP (for 0d 0h 11m 8s)
Status Information: PING OK - Packet loss = 0%, RTA = 0.03 ms
Performance Data: rta=0.032000ms;3000.000000;5000.000000;0.000000 pl=0%;80;100;0
Current Attempt: 1/10 (HARD state)
Last Check Time: 10-09-2024 06:41:22
Check Type: ACTIVE
Check Latency / Duration: 0.000 / 4.160 seconds
Next Scheduled Active Check: 10-09-2024 06:46:22
Last State Change: 10-09-2024 06:32:02
Last Notification: N/A (notification 0)
Is This Host Flapping? NO (0.00% state change)
In Scheduled Downtime? NO
Last Update: 10-09-2024 06:43:09 (0d 0h 0m 1s ago)

Host State Information

Active Checks:	ENABLED
Passive Checks:	ENABLED
Obsessing:	ENABLED
Notifications:	ENABLED
Event Handler:	ENABLED
Flap Detection:	ENABLED

Host Commands

- Locate host on map
- Disable active checks of this host
- Ré-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

This host has no comments associated with it

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

Current Network Status

Last Updated: Wed Oct 9 06:43:41 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		
0	2		

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
12	2	0	2	0
All Problems	All Types			
4	16			

Service Status Details For All Hosts

Host	Service	Status	Last Check	Duration	Attempt	Status Information
linuxserver	Current Load	OK	10-09-2024 06:42:37	0d 0h 11m 4s	1/4	OK - load average: 0.00, 0.02, 0.00
	Current Users	OK	10-09-2024 06:43:15	0d 0h 10m 26s	1/4	USERS OK - 5 users currently logged in
	HTTP	WARNING	10-09-2024 06:41:52	0d 0h 6m 49s	4/4	HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes in 0.000 second response time
	PING	OK	10-09-2024 06:39:30	0d 0h 9m 11s	1/4	PING OK - Packet loss = 0%, RTA = 0.03 ms
	Root Partition	OK	10-09-2024 06:40:07	0d 0h 8m 34s	1/4	DISK OK - free space: /6110 MB (75.29% inode=98%)
	SSH	OK	10-09-2024 06:40:45	0d 0h 7m 56s	1/4	SSH OK - OpenSSH_8.7 (protocol 2.0)
	Swap Usage	CRITICAL	10-09-2024 06:39:22	0d 0h 4m 19s	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	10-09-2024 06:42:00	0d 0h 6m 41s	1/4	PROCS OK. 40 processes with STATE = RSZDT
localhost	Current Load	OK	10-09-2024 06:41:09	0d 0h 47m 32s	1/4	OK - load average: 0.01, 0.03, 0.00
	Current Users	OK	10-09-2024 06:41:47	0d 0h 46m 54s	1/4	USERS OK - 5 users currently logged in
	HTTP	WARNING	10-09-2024 06:40:24	0d 0h 43m 17s	4/4	HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes in 0.000 second response time
	PING	OK	10-09-2024 06:43:02	0d 0h 45m 39s	1/4	PING OK - Packet loss = 0%, RTA = 0.03 ms
	Root Partition	OK	10-09-2024 06:43:38	0d 0h 45m 2s	1/4	DISK OK - free space: /6110 MB (75.29% inode=98%)
	SSH	OK	10-09-2024 06:43:17	0d 0h 44m 24s	1/4	SSH OK - OpenSSH_8.7 (protocol 2.0)
	Swap Usage	CRITICAL	10-09-2024 06:42:54	0d 0h 40m 47s	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	10-09-2024 06:40:32	0d 0h 43m 9s	1/4	PROCS OK. 40 processes with STATE = RSZDT

Results 1 - 16 of 16 Matching Services

System

- Comments
- Downtime
- Process Info
- Performance Info
- Scheduling Queue
- Configuration

Conclusion:

- We have learnt how to perform port, service monitoring using nagios.
- We added the linuxserver as host for ubuntu client.
- After restarting the NRPE server, we can see the 'linuxserver' host added.

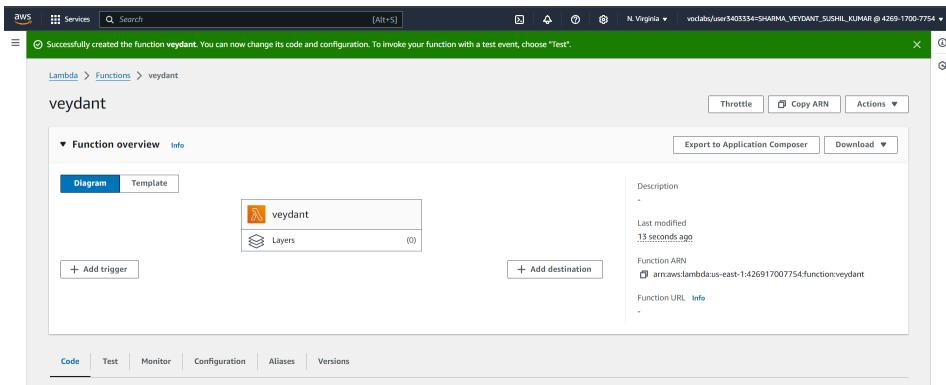
Aim: To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs

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

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

The screenshot shows the 'Create function' wizard in the AWS Lambda console. The 'Basic information' step is active. The 'Function name' field contains 'veydant'. The 'Runtime' dropdown is set to 'Node.js 20.x'. The 'Architecture' dropdown is set to 'x86_64'. The 'Permissions' section indicates that a default execution role will be created. Other tabs like 'Diagram' and 'Template' are available.

Your Lambda function gets created.



Step 3: The general configuration of the function is visible in the ‘Configuration’ tab. To change the configuration, click on ‘Edit’ Change the timeout to ‘1’. Then click on ‘Save’.

The screenshot shows the AWS Lambda Configuration page. The left sidebar lists options: General configuration, Triggers, Permissions, Destinations, Function URL, Environment variables, Tags, and VPC. The main panel displays the 'General configuration' section with the following details:

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

The screenshot shows the 'Basic settings' configuration page for a Lambda function. The fields are as follows:

- Description - optional:** (empty input field)
- Memory:** 128 MB (Set memory to between 128 MB and 10240 MB)
- Ephemeral storage:** 512 MB (Set ephemeral storage (/tmp) to between 512 MB and 10240 MB)
- SnapStart:** Info (None selected, supported runtimes: Java 11, Java 17, Java 21)
- Timeout:** 0 min 1 sec
- Execution role:** Use an existing role (radio button selected), Create a new role from AWS policy templates (radio button unselected)
- Existing role:** LabRole (dropdown menu, copy icon available)

Step4: Navigate to code source and click on the arrow near test to create test event.

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

The screenshot shows the AWS Lambda Code source editor. The code in the 'index.js' file is:

```

const test = "Hello D15C8";
export const handler = async (event) => {
  const response = {
    statusCode: 200,
    body: JSON.stringify(test),
  };
  return response;
};

```

Configure test event

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

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

Test event action

Create new event Edit saved event

Event name

lambda

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

```

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

```

Cancel Invoke Save

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

Execution results		Status: Succeeded	Max memory used: 62 M
Test Event Name	Lambda		
Response	{ "statusCode": 200, "body": "\Hello D15C50\" }		
Function Logs	START RequestId: 4cb66e32-9897-4955-a8da-327e4384600d Version: \$LATEST END RequestId: 4cb66e32-9897-4955-a8da-327e4384600d REPORT RequestId: 4cb66e32-9897-4955-a8da-327e4384600d Duration: 13.99 ms Billed Duration: 14 ms Memory Size: 128 MB Max Memory Used: 62 MB Init Duration: 143.30 ms		
Request ID	4cb66e32-9897-4955-a8da-327e4384600d		

Conclusion:

- We created a lambda function and configured using Node.js
- We setup a test event for the function.
- From the output of the tests, we learn about the working of the Lambda function and how changes in its configuration affects its functionality and outputs.

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.

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

Create bucket Info

Buckets are containers for data stored in S3.

General configuration

AWS Region
US East (N. Virginia) us-east-1

Bucket type Info

- General purpose

Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.
- Directory

Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name Info
expad12

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

Copy settings from existing bucket - optional
Only the bucket settings in the following configuration are copied.

Choose bucket

Format: s3://bucket/prefix

Uncheck the ‘Block all public access’ box.

Block Public Access settings for this bucket

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

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

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

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

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

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

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

The screenshot shows the 'Upload' interface for an S3 bucket named 'expad12'. In the 'Files and folders' section, there is one item: 'apx gp.jpg' (1 Total, 11.1 KB). Below the table are 'Remove', 'Add files', and 'Add folder' buttons. A note says 'All files and folders in this table will be uploaded.' A search bar labeled 'Find by name' is present. The 'Destination' section shows the destination as 's3://expad12'. Under 'Destination details', it says 'Bucket settings that impact new objects stored in the specified destination.' The 'Permissions' section indicates 'Grant public access and access to other AWS accounts.'

Your image gets uploaded onto the S3 bucket

The screenshot shows the 'Upload: status' interface. At the top, a green bar indicates 'Upload succeeded' with a link to 'View details below.'. Below is a summary table with three rows: Destination (s3://expad12), Succeeded (1 file, 11.1 KB (100.00%)), and Failed (0 files, 0 B (0%)). The 'Files and folders' tab is selected, showing a table with one row: 'apx gp.jpg' (image/jpeg, 11.1 KB, Status: Succeeded). A note at the top says 'The information below will no longer be available after you navigate away from this page.'

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

The screenshot shows the AWS Lambda Functions list page. At the top, there is a search bar labeled "Filter by tags and attributes or search by keyword". Below the search bar is a table titled "Functions (6)". The table has columns: "Function name", "Description", "Package type", "Runtime", and "Last modified". The "Create function" button is located at the top right of the table area.

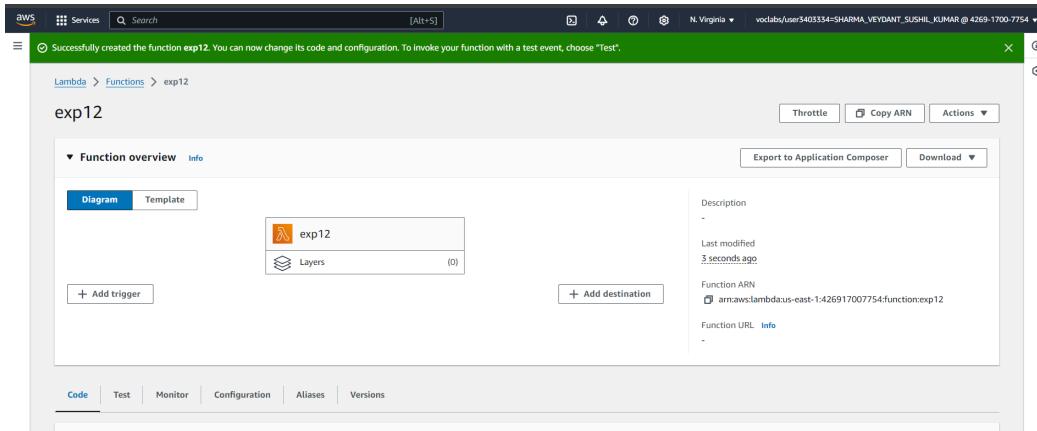
Function name	Description	Package type	Runtime	Last modified
RedshiftOverwatch	Deletes Redshift Cluster if the count is more than 2.	Zip	Python 3.8	2 months ago
RoleCreationFunction	Create SLR if absent	Zip	Python 3.8	2 months ago
RedshiftEventSubscription	Create Redshift event subscription to SNS Topic.	Zip	Python 3.8	2 months ago
MainMonitoringFunction	-	Zip	Python 3.8	2 months ago
ModLabRole	updates LabRole to allow it to assume itself	Zip	Python 3.8	2 months ago
veydant	-	Zip	Node.js 20.x	1 day ago

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

The screenshot shows the "Create function" wizard step 1. The "Function name" field contains "exp12". The "Runtime" dropdown is set to "Node.js 20.x". Other sections visible include "Architecture" (set to "x86_64"), "Permissions" (with a note about default execution role), and "Change default execution role" (with options for creating a new role, using an existing role, or creating from policy templates). The "Existing role" dropdown is set to "LabRole".

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

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



The trigger gets successfully added to your function

Trigger configuration [Info](#)

S3 aws asynchronous storage

Bucket
Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.
 [X](#) [C](#)
 Bucket region: us-east-1

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

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

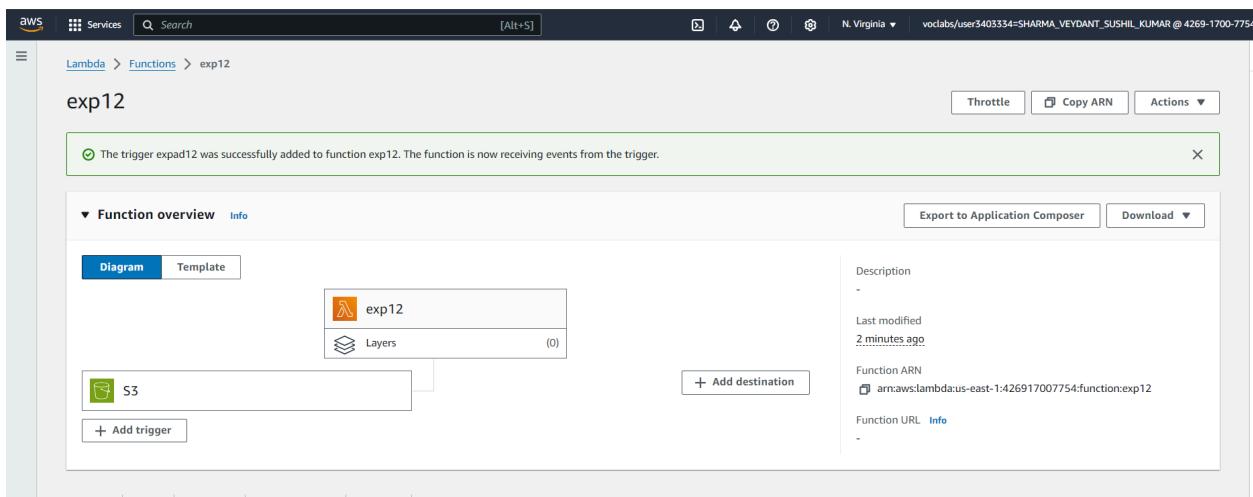
Suffix - optional
Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters. Any [special characters](#) must be URL encoded.

Recursive invocation
If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. [Learn more](#)

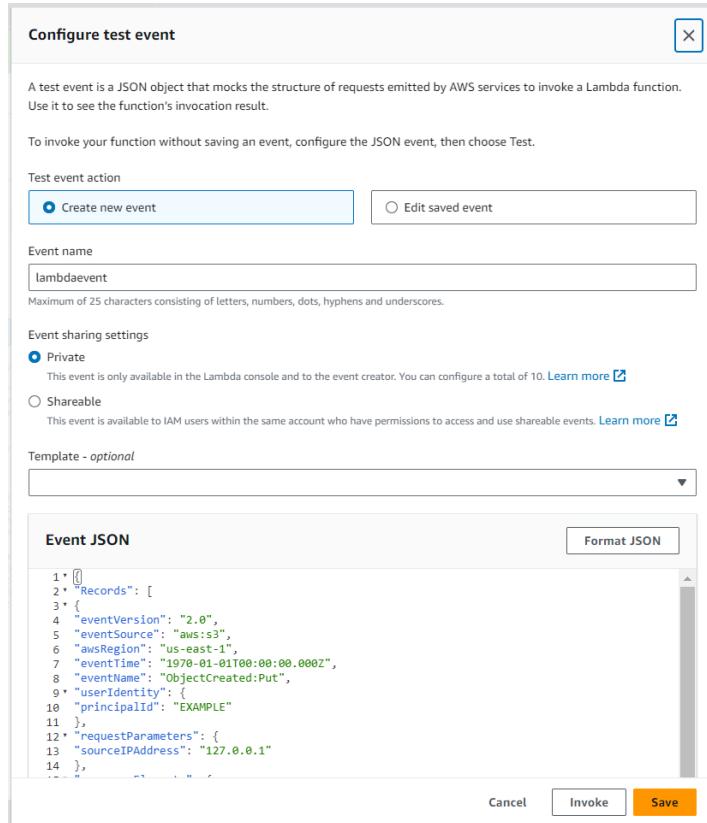
I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased

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

```
export const handler = async (event) => { if (!event.Records || event.Records.length === 0) { console.error("No records found in the event."); return { statusCode: 400, body: JSON.stringify('No records found in the event') }; } // Extract bucket name and object key from the event const record = event.Records[0]; const bucketName = record.s3.bucket.name; const objectKey = decodeURIComponent(record.s3.object.key.replace(/\+/g, ' ')); // Handle encoded keys console.log(`An image has been added to the bucket ${bucketName}: ${objectKey}`); console.log(`Event Source: ${record.eventSource}`); console.log(`Event Source: ${record.eventSource}`); console.log(`Event Source: ${record.eventSource}`); return { statusCode: 200, body: JSON.stringify('Log entry created successfully!') }; };
```



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



Step 9: Response generated after testing the code logs that the image was uploaded successfully.

```

Test event name: Lambdaeevent
Response:
{
  "statusCode": 200,
  "body": "\Log entry created successfully!\""
}
Function Logs
START RequestId: 6466cb07-530d-4873-b7d9-c5d656525014 Version: $LATEST
2024-10-10T20:01:27.106Z 6466cb07-530d-4873-b7d9-c5d656525014 INFO An image has been added to the bucket example-bucket: test/key
2024-10-10T20:01:27.129Z 6466cb07-530d-4873-b7d9-c5d656525014 INFO Event Source: aws:s3
2024-10-10T20:01:27.130Z 6466cb07-530d-4873-b7d9-c5d656525014 INFO Event Source: aws:s3
2024-10-10T20:01:27.130Z 6466cb07-530d-4873-b7d9-c5d656525014 INFO Event Source: aws:s3
2024-10-10T20:01:27.130Z 6466cb07-530d-4873-b7d9-c5d656525014 INFO Event Source: aws:s3
END RequestId: 6466cb07-530d-4873-b7d9-c5d656525014
REPORT RequestId: 6466cb07-530d-4873-b7d9-c5d656525014 Duration: 45.26 ms Billed Duration: 46 ms Memory Size: 128 MB Max Memory Used: 64 MB Init Duration: 143.23 ms
Request ID: 6466cb07-530d-4873-b7d9-c5d656525014

```

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

Timestamp	Message
No older events at this moment. Retry	
2024-10-10T15:09:08.226Z	INIT_START Runtime Version: nodejs:20.v39 Runtime Version ARN: arn:aws:lambda:us-east-1:runtime:ad9b28ae231dfc4c3325e183024ccb4d9deaa14796d98295f898140041242f7
2024-10-10T15:09:08.362Z	START RequestId: 52f08781-87fe-4178-9fa9-0d39b07cc62f Version: \$LATEST
2024-10-10T15:09:08.363Z	2024-10-10T15:09:08.363Z 52f08781-87fe-4178-9fa9-0d39b07cc62f INFO An image has been added to the bucket example-bucket: test/key
2024-10-10T15:09:08.377Z	2024-10-10T15:09:08.377Z 52f08781-87fe-4178-9fa9-0d39b07cc62f INFO Event Source: aws:s3
2024-10-10T15:09:08.394Z	2024-10-10T15:09:08.394Z 52f08781-87fe-4178-9fa9-0d39b07cc62f INFO Event Source: aws:s3
2024-10-10T15:09:08.394Z	2024-10-10T15:09:08.394Z 52f08781-87fe-4178-9fa9-0d39b07cc62f INFO Event Source: aws:s3
2024-10-10T15:09:08.394Z	2024-10-10T15:09:08.394Z 52f08781-87fe-4178-9fa9-0d39b07cc62f INFO Event Source: aws:s3
2024-10-10T15:09:08.416Z	END RequestId: 52f08781-87fe-4178-9fa9-0d39b07cc62f
2024-10-10T15:09:08.416Z	REPORT RequestId: 52f08781-87fe-4178-9fa9-0d39b07cc62f Duration: 53.40 ms Billed Duration: 54 ms Memory Size: 128 MB Max Memory Used: 64 MB Init Duration: 133.92 ms
No newer events at this moment. Auto retry paused . Resume	

Conclusion

- In this experiment we created Lambda function with S3 bucket that uploads our image.
- We configured the ‘Code section’ of our Lambda function and a test event for our Lambda function.
- A log of our function was also created showing successful operation.

Assignment 1

Q1] Use S3 bucket and host video streaming

→ Step 1: Create an S3 bucket.

i) Navigate to S3 bucket

ii) Create a new bucket

→ Click on "Create bucket"

→ Enter bucket name.

→ Choose a region

→ Create bucket

ii) Configure bucket permissions

→ Click on your bucket

→ Go to permissions tab

→ Edit block public access setting

→ Add bucket policy to allow

iii) Upload video files

→ Use console, AWS CLI or SDKs to upload files on supported formats.

→ Step 2: Set up CloudFront

i) → Open CloudFront

→ Create origin access identities and give a name.

iv) Go to create CloudFront distribution.
→ Select legacy access identities,
select identity that you created.

v) In default cache behaviour, select
redirect HTTP to HTTPS.

vi) To access, copy domain name of
distributor.

vii) Go to S3 bucket and click on the name
of the video you uploaded.

~~viii) Combine domain name and key of distributor
and video respectively to make the
final link.~~

2012-09-10 10:20:00

2012-09-10 10:20:00

2012-09-10 10:20:00

2012-09-10 10:20:00

Q2) Discuss BMW & hotstar case study using aws.

→ i) BMW case study

Objective: BMW aims to enhance connected vehicle services & improve data management.

AWS Utilization:

→ BMW leverages AWS.

→ a) Data storage & Analytics:

BMW leverages Amazon S3 for storing vast amounts of data collected from vehicles and is analyzed by AWS using Redshift or athena.

b) IoT integration:

The company employs AWS IoT core to securely connect its vehicles to the cloud facilitating real time data transfers.

c) Machine Learning:

BMW utilizes Amazon Sage maker for developing machine learning models that enhance customer experiences.

ii) Hotstar Case Study

Objective: Hotstar, a leading streaming service provider, aims to deliver high quality video content to millions of users.

AWS utilization:

a) Scalable infrastructure:

Hotstar uses EC2 instances to scale its computing resources dynamically based on demand.

b) Content delivery:

Platform uses Amazon CloudFront to distribute videos globally, ensuring low latency and high availability.

c) Data analytics:

Hotstar employs AWS analytic services to gain insights into user preferences and viewing habits.

Q3) Why Kubernetes & advantages, disadvantages of kubernetes. How adidas uses kubernetes

→ Kubernetes is an open source container orchestration platform that automates deployment, scaling and management of containerized applications.

• Advantages:

1 Scalability:

→ Kubernetes can automatically scale applications based on demand.

2 High availability:

→ Provides mechanisms for automatic restarts, replication & load balancing.

3 Resource Optimization

→ Schedules & manages containers effectively.

4 Portability:

→ Applications can run consistently across various environments.

* Disadvantages

i) Complexity!

→ Has a steep learning curve.

ii) Overhead

iii) Security Challenges

→ Misconfigurations can lead to vulnerabilities.

iv) Monitoring needs.

→ Effective monitoring is essential.

* How Adidas uses Kubernetes.

→ Adidas uses it to manage its microservices architecture enabling rapid deployment and development of application.

→ Its benefits include:

i) Support for E-commerce Operations

ii) Enhanced Collaboration

iii) Improved resource management.

Q4 What are Nagios & explain how Nagios is used in E-services?

- Nagios is an open source monitoring system that helps organizations monitor the health and performance of IT infrastructure.
- Nagios in E-services:

1 Monitoring uptime & downtime:

Nagios monitors availability of e-services, monitoring key such as websites, email servers or cloud platforms ensuring that these services stay online.

2 Performance monitoring:

It tracks the performance of E-service, monitoring key metrics.

3 Alerts and notifications:

Nagios can be configured to send notifications via email or SMS.

4 Log Monitoring

5 Custom plugins:

→ Allows installation of custom plugins for monitoring specific e-services.

Q4 What are Nagios & explain how Nagios is used in E-services?

→ Nagios is an open source monitoring system that helps organizations monitor the health and performance of IT infrastructure.

→ Nagios in E-services:

1 Monitoring uptime & downtime:

Nagios monitors availability of e-services, monitoring key such as websites, email servers or cloud platforms ensuring that these services stay online.

2 Performance monitoring:

It tracks the performance of E-services, monitoring key metrics.

3 Alerts and notifications:

Nagios can be configured to send notifications via email or SMS.

4. Log Monitoring

5. Custom plugins:

→ Allows installation of custom plugins for monitoring specified e-services.

Assignment 2

Create a rest api with a serverless framework.

Install Node.js and NPM from node.js website steps:

Install Serverless Framework

The serverless framework is a CLI tool that helps deploy serverless applications, managing resources like Lambda, API Gateway & more. Command: `npm install -g serverless`

Create a New Serverless Project

The serverless framework provides templates to scaffold a new serverless service.

Command: ~~`serverless create --template aws-nodejs --path my-rest-api`~~
~~`cd my-rest-api`~~

Configure Credentials

Command: `Aws configure`

Define API routes

Command: Modify `serverless.yml` as follows

service: my-rest-api

provider:

name: aws

runtime: nodejs14.x

region: us-east-1

stage: dev

functions:

createItem:

handler: handler.createItem

events:

- http:

path: item

method: post

getItem:

handler: handler.getItem

events:

- http:

path: item/{id}

method: get

updateItem:

handler: handler.updateItem

events:

- http:

path: item/{id}

method: put

deleteItem:

handler: handler.deleteItem

events:

- http:

path: item/{id}

method: delete

5 Write lambda function in handler.js

Deploy REST API

command: serverless deploy

Test the API

Test by using tools like Postman or curl to REST API

Monitor & Log:

For performance analysis

Case study for Sonargube

Create your own profile for Sonargube for testing project quality.

Use Sonarcloud to analyze your github code.

Install sonarlint in your java intelliJ id or eclipse.

Analyze python project with sonargube.

Analyze nodejs project with sonargube.

→ Sonargube helps developers identify issues like bugs, security etc.

1. Create personal profile
2. Sign in to sonargube
3. Connect your project

2. Sonarcloud integrates with github, offering continuous analysis.

3. Sonarlint is a plugin that offers real-time code quality checks while coding in IDE.

4. Sonargube supports python projects To analyse setup the python project with a sonar properties file.
- Run Sonarscanner
- View the detailed report

5. Analysis for node.js

- Create a sonar-project.properties file in project root
- Use sonarscanner for analysis
- Review the report.

→ At a large organization your centralized operations team may get many repetitive requests. You can use terraform to build a self serve infrastructure model that lets product teams independently manage infrastructure. You can create and use terraform modules that codify the standards for deploying and managing services in compliance with your organization's.

→ Self serve infrastructure with terraform By using terraform you can build reusable modules that encapsulate the best practice and compliance standard of your own organization. These modules can be made available to various product teams, empowering them to provision infrastructure themselves.

→ Terraform modules for standardization. Terraform modules are reusable templates that simplify the process of deploying infrastructure by codifying the standards for managing resources.

→ Integration with ticketing systems like ServiceNow.

Terraform with tools cloud on Enterprise can integrate process of generating new infrastructure requests.

- For instance, when a product team creates a ticket for infrastructure resources, cloud can automatically handle provisioning based on preapproved templates.