

Thadomal Shahani Engineering College

Bandra (W.), Mumbai- 400 050.

© CERTIFICATE ©

Certify that Mr./Miss Atharva Pundit

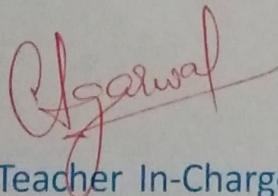
of IT Department, Semester V with

Roll No. 87 has completed a course of the necessary

experiments in the subject Advanced Devops under my

supervision in the **Thadomal Shahani Engineering College**

Laboratory in the year 2023 - 2024



Teacher In-Charge

Head of the Department

Date 21/10/23

Principal

CONTENTS

SR. NO.	EXPERIMENTS	PAGE NO.	DATE	TEACHERS SIGN.
1.	To study and perform setup of AWS Ec2 Service & launch an EC2 instance	19/7/23		
2.	To study & perform the setup of AWS cloud 9 Service & launch a Python program in cloud 9	19/7/23		
3.	To study Aws S3 service & create bucket for housing static web application	26/7/2023		
4.	To study Aws cloud pipeline & deploy web application using cloud pipeline	21/8/2023		
5.	To understand Kubernetes cluster Architecture	11/10/2023		
6.	To understand Terraform life cycle and build, change and destroy AWS infrastructure using Terraform	23/8/2023		
7.	To perform static analysis on Python program using SonarQube SAST process	13/9/2023		Rajatwal 21/10/23
8.	To understand continuous hosting using Nginx	21/8/2023		
9.	To understand Aws Lambda function & write a lambda function using Python to log an "an image has been added" message once a file is added to S3 bucket	27/9/2023		
10.	To Create a Lambda function using Python for adding data to Dynamo DB database	27/9/2023		
11.	Written Assignment 1	26/7/2023		
12.	Written Assignment 2	11/10/2023		

Assignment 1

Aim: To study EC2 service in AWS and perform the following:

1. Create an account in AWS.
2. Create a new EC2 instance and run the instance
3. Run a virtual Unix environment on the above instance.

Theory:

Creating an Amazon Web Services (AWS) EC2 (Elastic Compute Cloud) instance involves several steps. EC2 instances are virtual machines that can run a variety of operating systems. Here are the basic steps to study and create an AWS EC2 instance:

1. Sign in or create an AWS Account:

If you don't already have an AWS account, go to the [AWS website] (<https://aws.amazon.com/>) and sign up for an account.

2. Access the AWS Management Console:

Log in to the AWS Management Console using your credentials.

3. Navigate to the EC2 Dashboard:

In the AWS Management Console, go to the EC2 Dashboard. You can find it under the "Compute" section or search for "EC2."

4. Choose an AWS Region:

Select the AWS region where you want to launch your EC2 instance. AWS has regions worldwide; choose one closest to your target audience or for lower latency.

5. Launch an EC2 Instance:

Click on the "Instances" link on the EC2 Dashboard to create a new instance.

6. Select an Amazon Machine Image (AMI):

An AMI is a pre-configured image of an operating system. Choose an AMI that suits your needs, such as a Linux distribution, Windows Server, or a specific application image.

7. Choose an Instance Type:

Select the instance type that best fits your requirements in terms of CPU, memory, storage, and network performance. AWS offers various instance types optimized for different workloads.

8. Configure Instance Details:

Customize the instance settings, including the number of instances, VPC (Virtual Private Cloud), subnet, and other networking options.

9. Add Storage:

Specify the storage capacity for your instance. You can also add additional EBS (Elastic Block Store) volumes if needed.

10. Add Tags:

Assign tags to your EC2 instance to make it easier to organize and identify resources.

11. Configure Security Groups:

Security groups act as virtual firewalls for your instance. Define inbound and outbound rules to control traffic access.

12. Review and Launch:

Review your instance's configuration to ensure it's accurate. Then, click "Launch."

13. Create a Key Pair:

If you haven't created a key pair before, you'll be prompted to create one. This key pair is essential for securely connecting to your EC2 instance.

14. Launch the Instance:

Select the key pair you created and click "Launch Instances." This action will initiate the creation of your EC2 instance.

15. View Instance Details:

After your instance is launched, you can view its details, including its public IP address and other information, on the EC2 Dashboard.

16. Connect to Your EC2 Instance:

You can connect to your EC2 instance using SSH (for Linux) or Remote Desktop (for Windows). Use the key pair you created to authenticate.

17. Configure Your Instance:

Once connected, you can configure your EC2 instance according to your project or application requirements.

Snapshots:

The screenshot shows the AWS Console Home page. At the top, there's a navigation bar with the AWS logo, a search bar, and a user profile. Below the navigation bar, the main content area is titled "Console Home". It features several cards: "Recently visited" (with links to S3, Lambda, CloudWatch, DynamoDB, EC2, Billing, and Cloud9), "Welcome to AWS" (with links to Getting started with AWS, Training and certification, and What's new with AWS), "AWS Health" (showing 0 open issues, 0 scheduled changes, and 0 other notifications), "Cost and usage" (showing current month costs and top costs for the month), and "Build a solution" (with a link to Start building). At the bottom, there are links for CloudShell, Feedback, and copyright information.

The screenshot shows the AWS search results for the query "ec2". The search bar at the top contains "ec2". Below it, the results are categorized under "Services" and "Features". Under "Services", there are nine results: EC2 (Virtual Servers in the Cloud), EC2 Image Builder (A managed service to automate build, customize and deploy OS images), and AWS Compute Optimizer (Recommend optimal AWS Compute resources for your workloads). On the left, there are also links for Blogs, Documentation, Knowledge Articles, Tutorials, Events, and Marketplace. At the bottom right, there's a link to "See all 9 results".

Conclusion: Hence, we learned and implemented the steps will help us create an AWS instance.

Assignment 2

CLOUD9

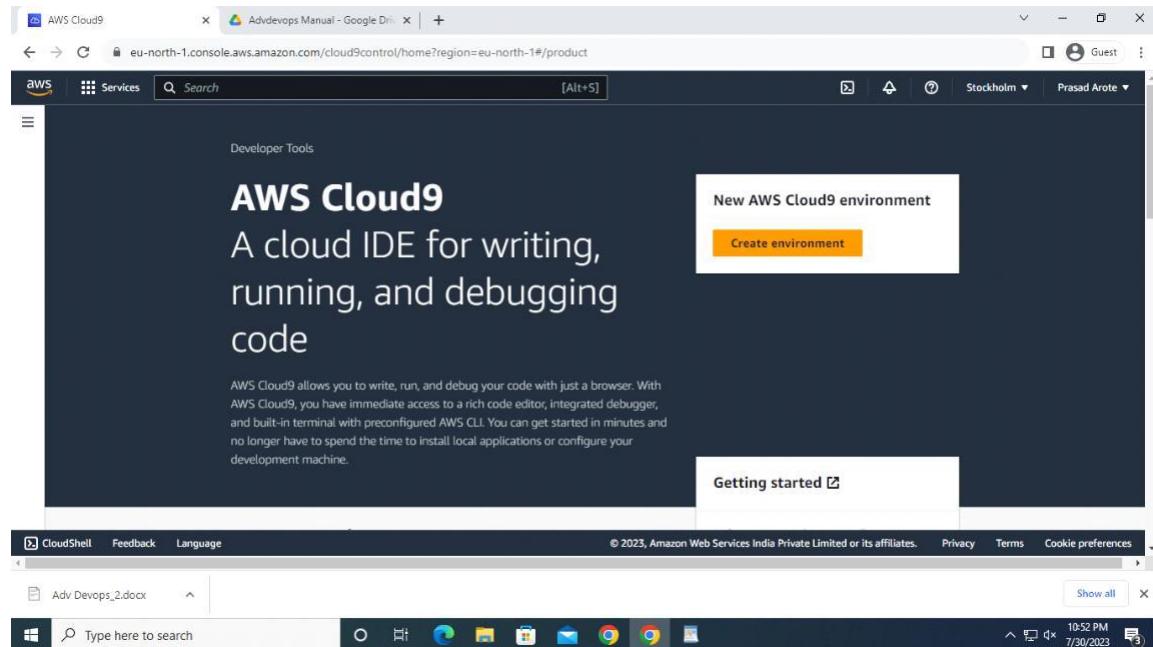
Aim: To create a Cloud9 Environment.

Theory-Cloud9 IDE is an Online IDE, published as open source from version 2.0, until version 3.0. It supports multiple programming languages, including C, C++, PHP, Ruby, Perl, Python, JavaScript with Node.js, and Go. It is written almost entirely in JavaScript, and uses Node.js on the back-end.

Steps-

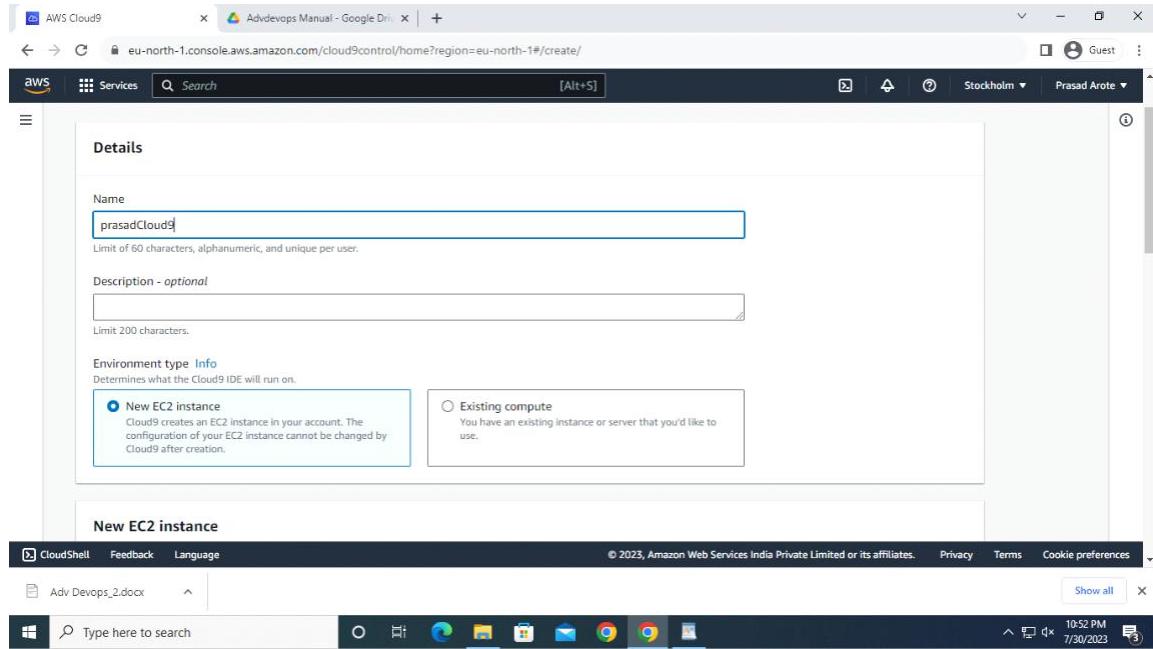
LOG IN TO YOUR AWS ACCOUNT,

SEARCH FOR CLOUD 9 IN THE SEARCH BAR

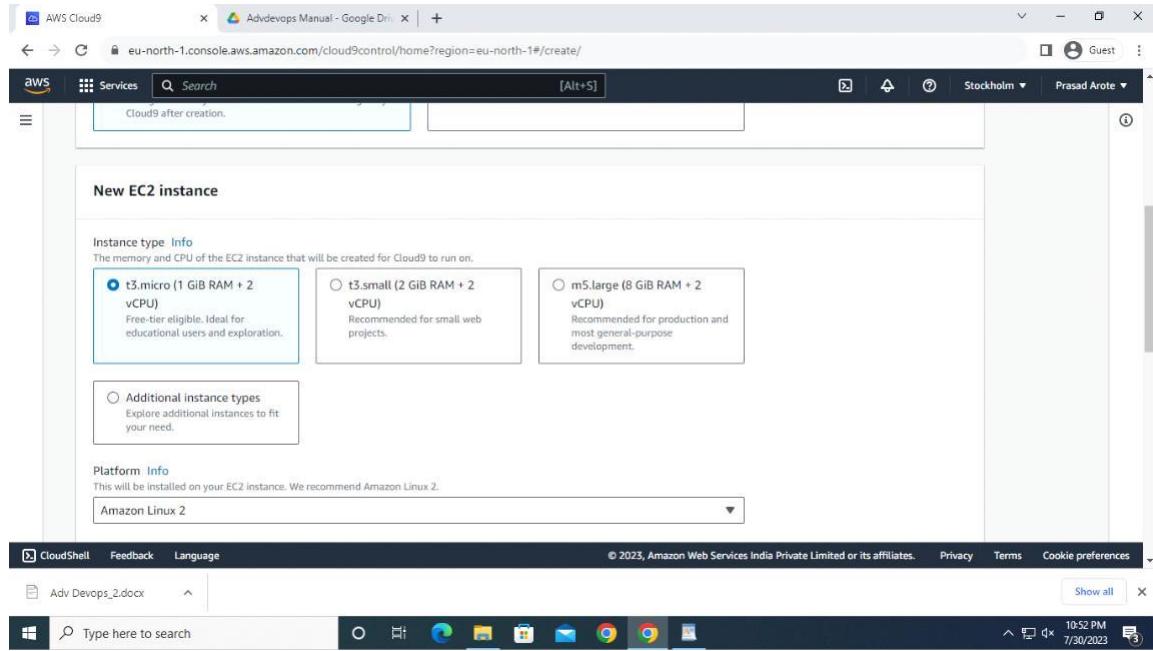


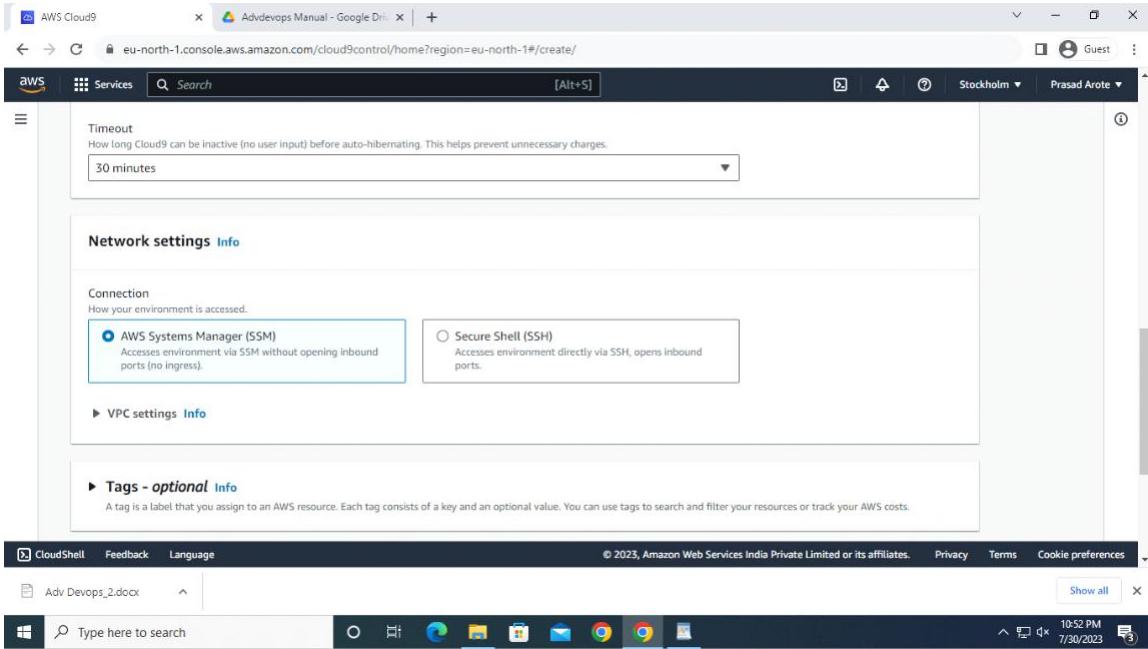
CLICK ON CREATE ENVIRONMNET,

NAME THE ENVIRONMNET

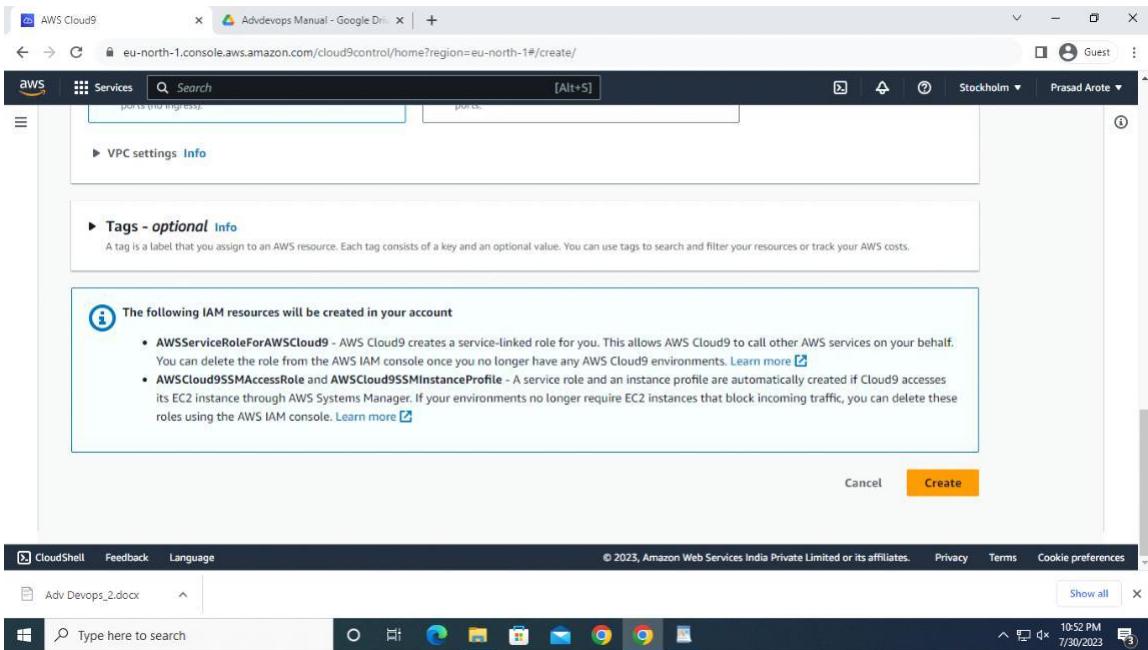


Now click on next step.





Again, click on Next Step,
Now click on Create Environment.



The screenshot shows the AWS Cloud9 interface. On the left, there's a sidebar with 'AWS Cloud9' and 'Environments'. The main area is titled 'Environments (1)' and shows a table with one row:

Name	Cloud9 IDE	Environment type	Connection	Permission	Owner ARN
prasadCloud9	Open	EC2 instance	AWS Systems Manager (SSM)	Owner	arn:aws:iam::042130962394:root

Now select any coding language and perform any operation via a code. Shown below

The screenshot shows the AWS Cloud9 IDE interface. The top navigation bar includes 'File', 'Edit', 'Find', 'View', 'Go', 'Run', 'Tools', 'Window', 'Support', 'Preview', and 'Run' buttons. The main workspace shows a file named 'hello.c' with the following code:

```
#include<stdio.h>
int main(){
    printf("Hello World");
    return 0;
}
```

Below the code editor, there's a terminal window showing the output of the program:

```
bash - ip-172-31-9-211.e x Immediate x hello.c - Stopped x
Running /home/ec2-user/environment/hello.c
Hello World
Process exited with code: 0
```

Conclusion: Hence learned and implemented steps to Create an Cloud9 environment.

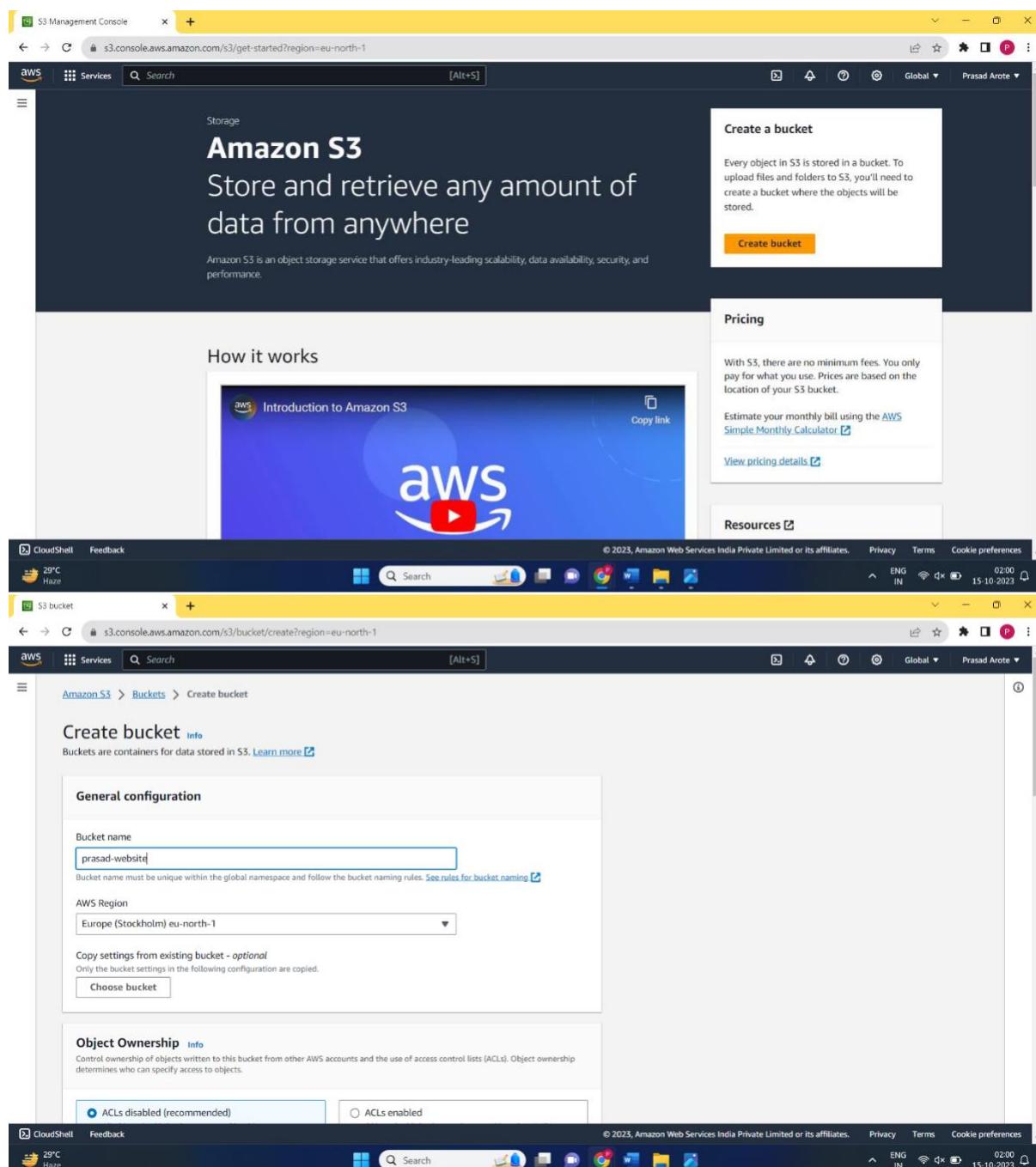
Assignment 3

AIM: To study AWS S3 service and create a bucket for hosting static web application.

LO1: To understand the fundamentals of Cloud Computing and be fully proficient with Cloud based DevOps solution deployment options to meet your business requirements.

THEORY:

1. Create a S3 bucket.



The screenshot shows the AWS S3 Management Console interface. At the top, there's a navigation bar with tabs for 'AWS Services' and 'Search'. Below the navigation bar, the main content area has a dark header with the text 'Amazon S3' and the subtext 'Store and retrieve any amount of data from anywhere'. A brief description of Amazon S3 follows. On the right side, there are three callout boxes: 'Create a bucket', 'Pricing', and 'Resources'. The 'Create a bucket' box contains instructions about buckets and a prominent orange 'Create bucket' button. The 'Pricing' box states that there are no minimum fees and provides a link to the AWS Simple Monthly Calculator. The 'Resources' box links to more resources. The bottom half of the screen shows the 'Create bucket' wizard. It has two main sections: 'General configuration' and 'Object Ownership'. In 'General configuration', the 'Bucket name' field is filled with 'prasad-website'. The 'AWS Region' dropdown is set to 'Europe (Stockholm) eu-north-1'. There's also a section for 'Copy settings from existing bucket - optional' with a 'Choose bucket' button. In 'Object Ownership', the 'ACLs disabled (recommended)' option is selected. The footer of the browser window shows standard navigation icons and system status information like battery level and date/time.

The screenshots show the AWS S3 Bucket creation process:

- Step 1: Set Bucket Access Control**
 - Block all public access:** This setting turns on four other options: Block public access to buckets and objects granted through new access control lists (ACLs), Block public access to buckets and objects granted through any access control lists (ACLs), Block public access to buckets and objects granted through new public bucket or access point policies, and Block public and cross-account access to buckets and objects through any public bucket or access point policies.
 - Warning:** Turning off all public access might result in the bucket and its objects becoming public. AWS recommends turning it off unless required for specific use cases like static website hosting.
 - Acknowledgment:** A checkbox is checked, acknowledging the potential risk.
- Step 2: Set Bucket Encryption**
 - Default encryption:** Server-side encryption is automatically applied to new objects stored in this bucket.
 - Encryption type:** Server-side encryption with Amazon S3 managed keys (SSE-S3) is selected.
 - Bucket Key:** Bucket Keys are disabled.
- Step 3: Advanced Settings**
 - After creating the bucket, you can upload files and folders to the bucket and configure additional bucket settings.

2. Upload the files of web application.

Screenshot of the AWS S3 Management Console showing the upload process for a website.

Upload Step:

- The "Files and folders (0)" section shows a message: "No files or folders. You have not chosen any files or folders to upload."
- Buttons: Remove, Add files, Add folder.

Destination Step:

- Section: Destination
- CloudShell Feedback status: 29°C Haze

File Selection Step:

- The "Files and folders (24 Total, 89.5 KB)" section lists the uploaded files:

Name	Type	Size
Bun 1.svg	image/svg+xml	865.0 B
Bun 1@2x.png	image/png	8.6 KB
Cheese.svg	image/svg+xml	619.0 B
Cheese@2x.png	image/png	1.4 KB
Lettuce.svg	image/svg+xml	629.0 B
Lettuce@2x.png	image/png	2.4 KB
Onion.svg	image/svg+xml	851.0 B
Onion@2x.png	image/png	2.8 KB
Patty.svg	image/svg+xml	659.0 B
Patty@2x.png	image/png	3.9 KB
- Buttons: Remove, Add files, Add folder.

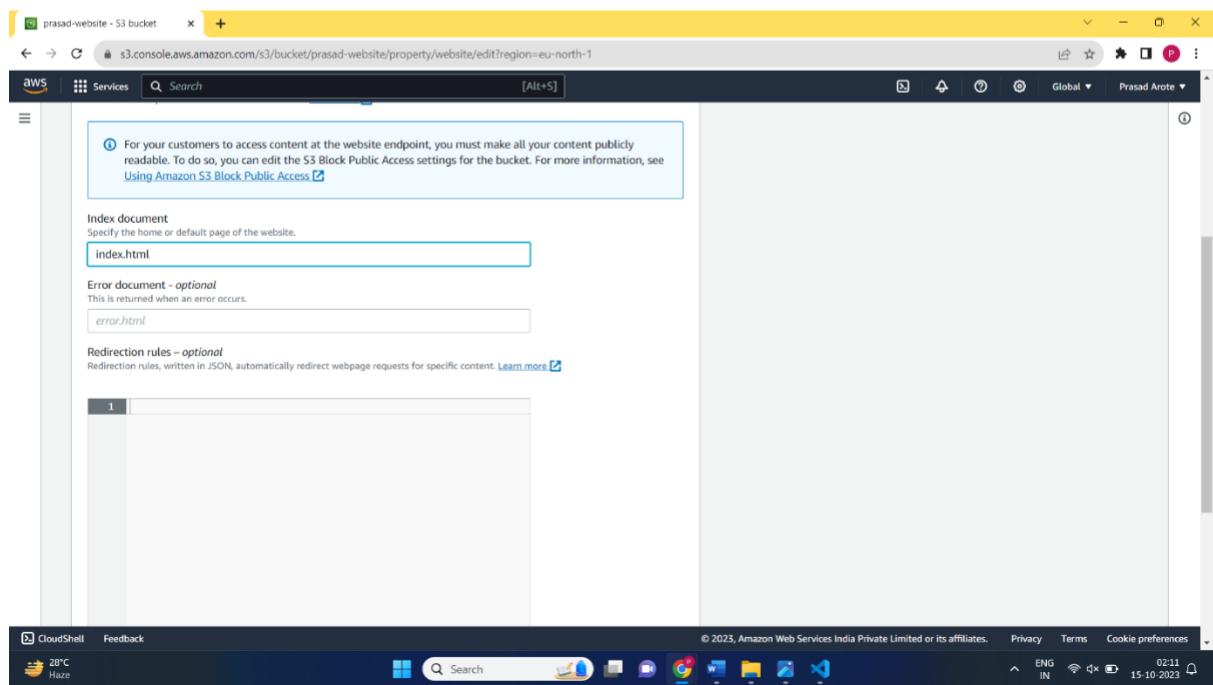
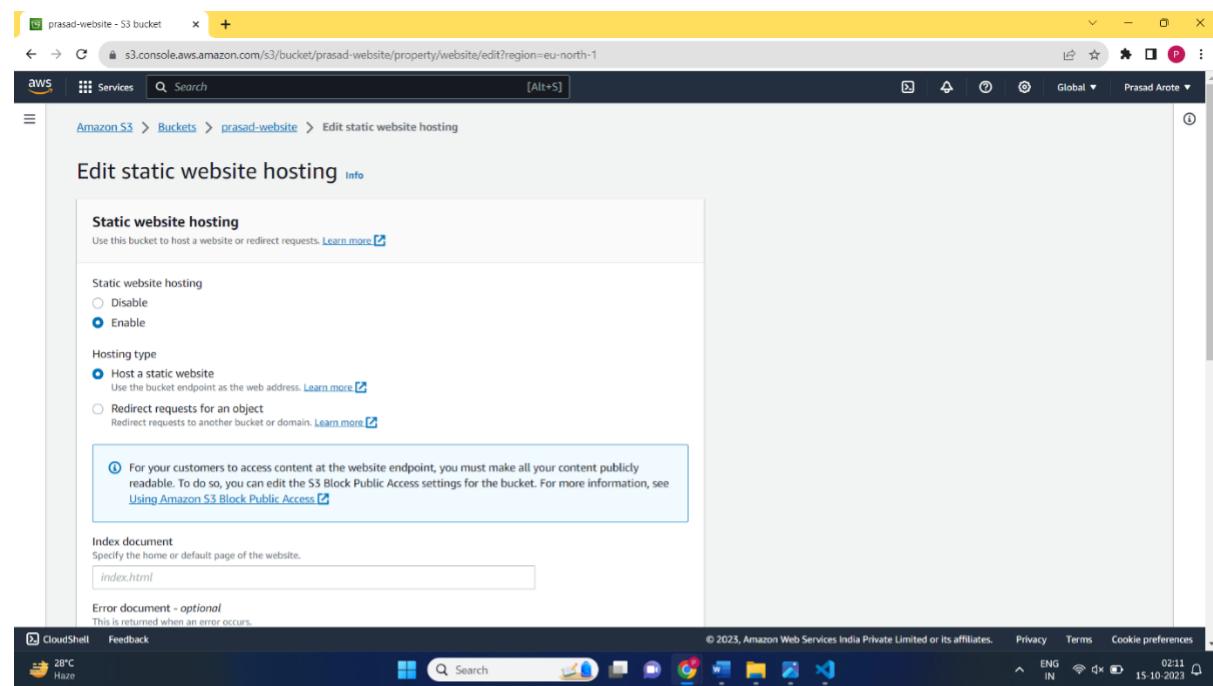
Success Step:

- Message: "Upload succeeded" with a "View details below." link.
- Summary table:

Destination	Succeeded	Failed
s3://prasad-website	24 files, 89.5 KB (100.00%)	0 files, 0 B (0%)
- Buttons: Close.
- File selection table:

Name	Folder	Type	Size	Status	Error
Bun 1.svg	-	image/svg+xml	865.0 B	Success	
Bun 1@2x.png	-	image/png	8.6 KB	Success	
Cheese.svg	-	image/svg+xml	619.0 B	Success	
Cheese@2x.png	-	image/png	1.4 KB	Success	
Lettuce.svg	-	image/svg+xml	629.0 B	Success	
Lettuce@2x.png	-	image/png	2.4 KB	Success	
Onion.svg	-	image/svg+xml	851.0 B	Success	
Onion@2x.png	-	image/png	2.8 KB	Success	
Patty.svg	-	image/svg+xml	659.0 B	Success	
Patty@2x.png	-	image/png	3.9 KB	Success	

3. Enable Static website hosting



The screenshot shows the AWS S3 console interface. The top navigation bar includes the AWS logo, Services, a search bar, and a global dropdown for 'Prasad Arote'. The main content area displays the properties of the 'prasad-website' bucket. The 'Properties' tab is selected, showing the following details:

- AWS Region:** Europe (Stockholm) eu-north-1
- Amazon Resource Name (ARN):** arn:aws:s3:::prasad-website
- Creation date:** October 15, 2023, 02:01:26 (UTC+0:30)

Below this, the 'Bucket Versioning' section is visible, indicating it is disabled. The bottom of the screen shows the Windows taskbar with various pinned icons and system status.

5. Change the Bucket Policy

The screenshot shows the 'Edit bucket policy' page for the 'prasad-website' bucket. The top navigation bar is identical to the previous screenshot. The main content area is titled 'Edit bucket policy' and contains the following information:

- Bucket policy:** A JSON-based policy that provides access to objects in the bucket. It applies to objects owned by the account and does not apply to objects owned by other accounts.
- Bucket ARN:** arn:aws:s3:::prasad-website
- Policy:** A list of statements. The first statement is selected and labeled '1'. To the right of the list is a panel titled 'Edit statement' with a 'Select a statement' dropdown and a '+ Add new statement' button.

The bottom of the screen shows the Windows taskbar with various pinned icons and system status.

AWS Policy Generator

The AWS Policy Generator is a tool that enables you to create policies that control access to Amazon Web Services (AWS) products and resources. For more information about creating policies, see Key Concepts in Using AWS Identity and Access Management. Here are sample policies.

Step 1: Select Policy Type

A Policy is a container for permissions. The different types of policies you can create are an IAM Policy, an S3 Bucket Policy, an SNS Topic Policy, a VPC Endpoint Policy, and an SQS Queue Policy.

Select Type of Policy: S3 Bucket Policy

Step 2: Add Statement(s)

A statement is the formal description of a single permission. See a description of elements that you can use in statements.

Effect: Allow (radio button selected) Deny

Principal: *

AWS Service: Amazon S3 (dropdown menu) All Services (*)

Actions: 1 Action(s) Selected (dropdown menu) All Actions (*)

Amazon Resource Name (ARN): arn:aws:s3:::prasad-website

Add Conditions (Optional)

Add Statement

You added the following statements. Click the button below to Generate a policy.

Principal(s)	Effect	Action	Resource	Conditions
*	Allow	s3:GetObject	arn:aws:s3:::prasad-website	None

Step 3: Generate Policy

A policy is a document (written in the Access Policy Language) that acts as a container for one or more statements.

Generate Policy Start Over

This AWS Policy Generator is provided for informational purposes only; you are still responsible for your use of Amazon Web Services technologies and ensuring that your use is in compliance with all applicable terms and conditions. This AWS Policy Generator is provided **as is** without warranty of any kind, whether express, implied, or statutory. This AWS Policy Generator does not modify the applicable terms and conditions governing your use of Amazon Web Services technologies.

©2010, Amazon Web Services LLC or its affiliates. All rights reserved.
An **amazon.com** company

Screenshot of the AWS Policy Generator tool showing a JSON policy document for an S3 bucket.

```

{
  "Id": "Policy1697316791653",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1697316788348",
      "Action": [
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::prasad-website/*",
      "Principal": "*"
    }
  ]
}

```

The policy allows public access to all objects in the 'prasad-website' bucket.

AWS Policy Generator

This AWS Policy Generator is provided for informational purposes only. You are still responsible for your use of Amazon Web Services technologies and ensuring compliance with all applicable terms and conditions. This AWS Policy Generator is provided as-is without warranty of any kind, whether express, implied, or statutory. This AWS Policy Generator does not modify the applicable terms and conditions governing your use of Amazon Web Services technologies.

An [amazon.com](#) company

Screenshot of the AWS S3 console showing the policy editor for the 'prasad-website' bucket.

```

1  {
2   "Id": "Policy1697316791653",
3   "Version": "2012-10-17",
4   "Statement": [
5     {
6       "Sid": "Stmt1697316788348",
7       "Action": [
8         "s3:GetObject"
9       ],
10      "Effect": "Allow",
11      "Resource": "arn:aws:s3:::prasad-website/*",
12      "Principal": "*"
13    }
14  ]
15 }

```

The policy allows public access to all objects in the 'prasad-website' bucket.

Edit statement

Select a statement

Add new statement

CloudShell Feedback © 2023, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences ENG IN 02:23 15-10-2023

The screenshot shows the AWS S3 Buckets page. At the top, there's an 'Account snapshot' section with a link to 'View Storage Lens dashboard'. Below it, a table lists one bucket:

Name	AWS Region	Access	Creation date
my-s3-web-bucket	Europe (Stockholm) eu-north-1	Bucket and objects not public	August 23, 2023, 01:22:23 (UTC-07:00)

6. Now open the link (given in the bucket below) in browser.

The screenshot shows the AWS Lambda Code Editor interface. On the left, a file tree shows a 'py-code' folder containing 'Factorial.py' and 'README.md'. The 'Factorial.py' file contains the following Python code:

```
1 """
2 Your module description
3 """
4 number = int(input("Enter a number:"))
5
6 def facto( n):
7     if(n==1 or n==0):
8         return 1
9     else:
10        return n*facto(n-1)
11
12 print("Factorial of ",number," is ",facto(number))
13
14
```

On the right, the 'Run' tab is active, showing the command 'py-code/Factorial.py'. The output window displays the execution results:

```
Enter a number:6
Factorial of 6 is 720

Process exited with code: 0
```

Conclusion:

Here we studied to host about s3 buckets and hosting projects on the bucket.

Assignment 4

Aim: To study AWS CodePipeline and deploy a web application using CodePipeline.

Theory:

AWS CodePipeline is a fully managed continuous integration and continuous delivery (CI/CD) service provided by Amazon Web Services (AWS). It enables you to automate the building, testing, and deployment of your applications or code changes, facilitating the release of software changes more quickly and reliably.

AWS CodePipeline works by creating a series of stages in a pipeline, each of which can perform various actions, such as source code retrieval, building, testing, and deployment. It can integrate with various AWS services and other tools, providing a flexible and extensible way to set up your CI/CD workflows.

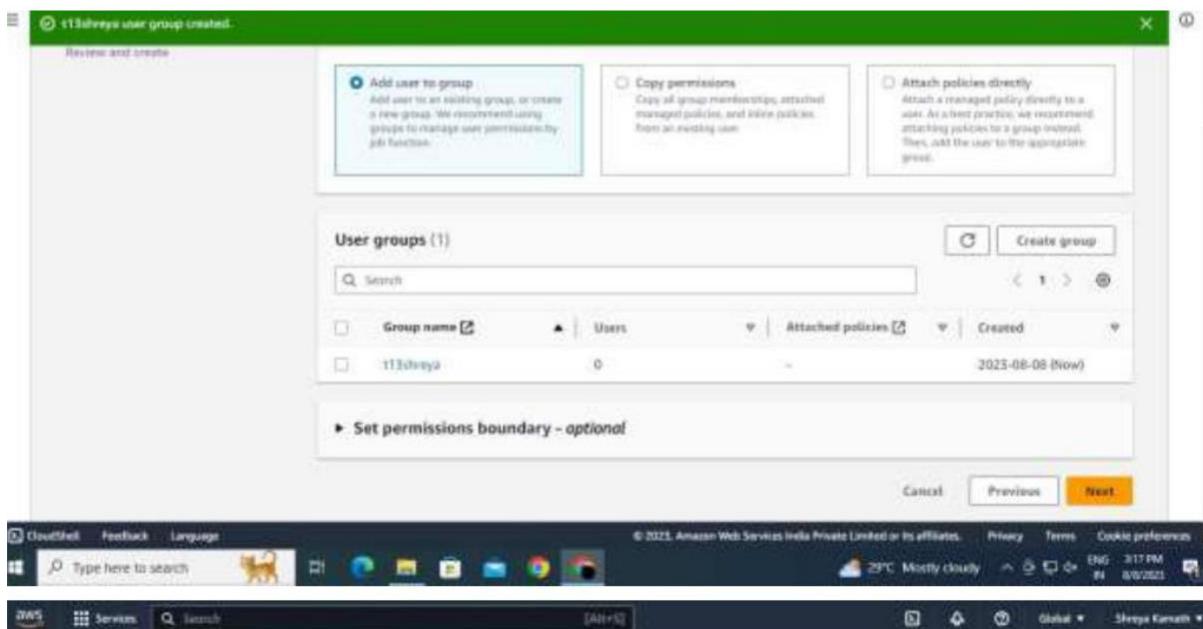
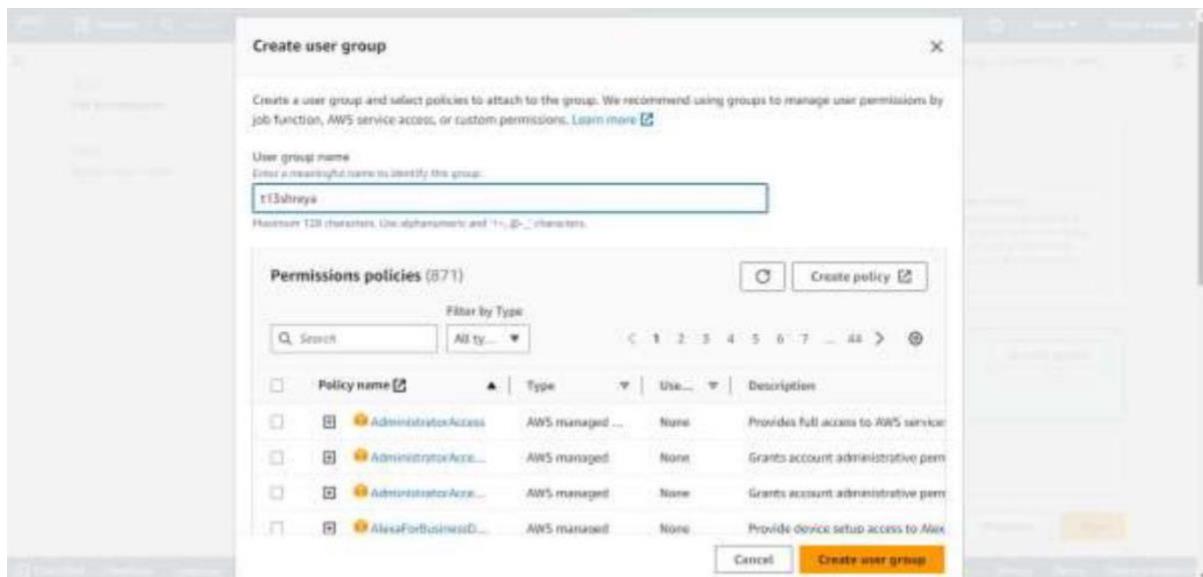
To deploy a static web application using AWS CodePipeline, you can follow these high-level steps:

1. Create an Amazon S3 bucket to host your web app files.
2. Set up an AWS CodeCommit or other source code repository.
3. Create an AWS CodePipeline.
4. Define the source provider (CodeCommit, GitHub, S3).
5. Configure a build stage using AWS CodeBuild or similar services.
6. Set up optional testing stages.
7. Configure deployment stages for the S3 bucket.
8. Grant CodePipeline permissions to access the S3 bucket.
9. Review and validate your pipeline configuration.
10. Trigger a manual or automatic pipeline execution for testing.
11. Monitor pipeline executions for success or failures.
12. Enhance your pipeline as needed for testing, monitoring, or notifications.
13. Optionally, set up a custom domain with Amazon Route 53 and CloudFront.

Output:

The screenshot shows the AWS Identity and Access Management (IAM) service interface. On the left, there's a navigation sidebar with various options like Dashboard, Access management, User groups, Users, Roles, Policies, Identity providers, Account settings, Access reports, Access analyzer, Archive rules, and Analyzers. The main area is titled "Users (0) info" and contains a sub-header: "An IAM user is an identity with long-term credentials that is used to interact with AWS X.0 accounts." Below this is a search bar labeled "Find users by username or access key". A table header is present with columns: User name, Groups, Last activity, MFA, Password age, and Active. A message at the bottom of the table says "No resources to display".

This screenshot shows the "Specify user details" step of the "Create user" wizard. The top navigation bar indicates "Step 1: Specify user details", "Step 2: Set permissions", and "Step 3: Review and create". The main form is titled "User details" and contains a "User name" field with the value "shreyakanath". Below the field is a note: "The user name may have up to 48 characters. Valid characters: a-z, p-q, 0-9, and + - _ @ . (hyphen)." There is also a checkbox for "Provide user access to the AWS Management Console - optional" with a note: "If you're granting console access to a person, it's a best practice to manage their access in IAM Identity Center." At the bottom right of the form are "Cancel" and "Next" buttons.



Screenshot 1: IAM User Group Details

The screenshot shows the AWS IAM User Groups page. A modal window titled "Users added to this group" is open, showing one user: "t13shreya". The "Summary" tab is selected. The user group name is "t13shreya", creation date is "August 08, 2023, 10:17 (UTC+05:30)", and ARN is "arn:aws:iam:538767473830:group/t13shreya". There are tabs for "Users", "Permissions", and "Access Advisor". Buttons for "Delete", "Edit", "Remove users", and "Add users" are visible.

Screenshot 2: IAM User List

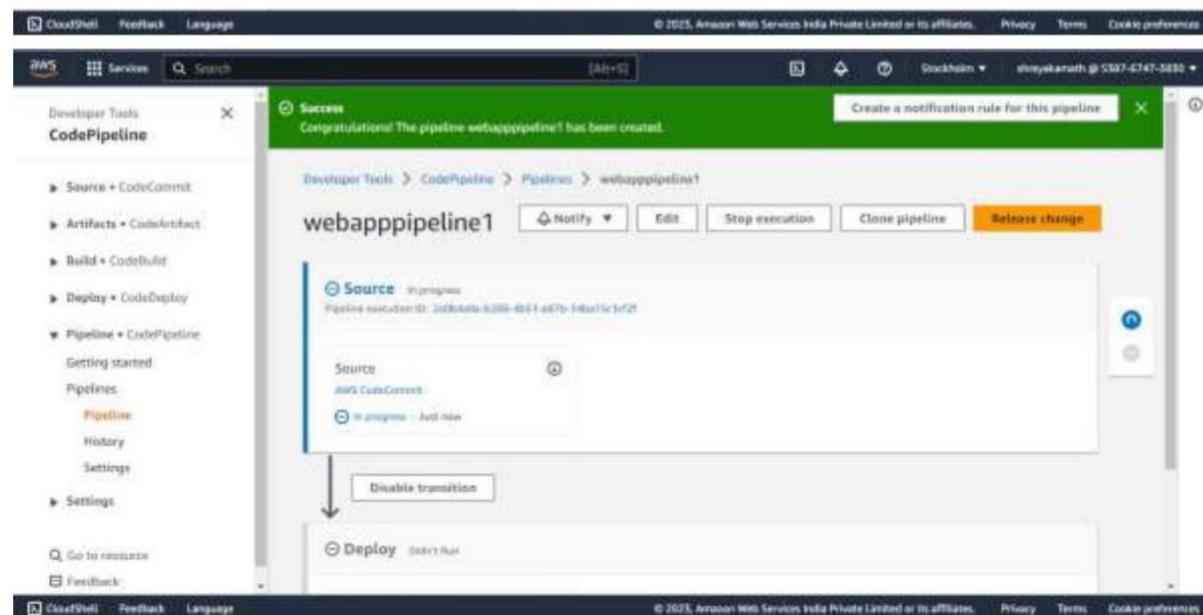
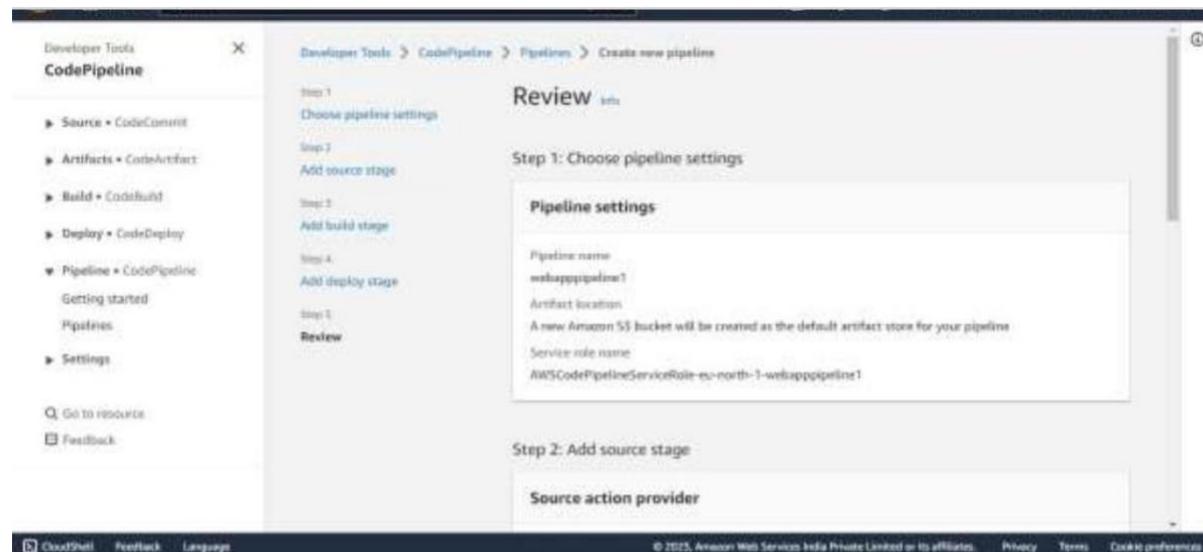
The screenshot shows the AWS IAM Users page. A modal window titled "Users (1) info" is open, showing one user: "shreyakamath". The user has the ARN "arn:aws:iam:538767473830:user/shreyakamath". The "Active" status is shown as "Yes". There are buttons for "Delete" and "Add users".

Screenshot 3: IAM User Details

The screenshot shows the AWS IAM User Details page for "shreyakamath". The "Summary" tab is selected. The ARN is "arn:aws:iam:538767473830:user/shreyakamath", console access is "Disabled", and access keys are "Not enabled". The user was created on "August 08, 2023, 10:17 (UTC+05:30)". There are tabs for "Permissions", "Groups (1)", "Tags", "Security credentials", and "Access Advisor". The "Permissions policies (0)" section indicates no policies are attached. There is a "Delete" button.

The screenshot shows the AWS Console Home page. In the top left, there's a 'Recently visited' section with links to Cloud9 and CloudWatch Metrics. To its right is a 'Welcome to AWS' section with three cards: 'Getting started with AWS' (with a rocket icon), 'Training and certification' (with a graduation cap icon), and 'What's new with AWS?' (with a lightbulb icon). At the bottom of this section is a 'View all services' link. The top navigation bar includes 'AWS', 'Services', 'Search', and user information ('Stockholm', 'sivayakarath @ 5587-6747-3830'). Below the main content is a standard Windows taskbar with icons for File Explorer, Task View, Start, Taskbar settings, and system status.

The screenshot shows the AWS CodeCommit documentation page. On the left, there's a sidebar with 'Developer Tools' and 'CodeCommit' sections, including 'Source' (selected), 'Code', 'Pull requests', 'Commits', 'Branches', 'Git tags', 'Settings', and 'Approval rule templates'. Below these are 'Artifacts', 'Build', 'Deploy', and 'Pipeline' sections. The main content area has three steps: 'Step 1: Prerequisites' (about Git clients), 'Step 2: Git credentials' (about IAM users and credentials), and 'Step 3: Clone the repository' (with a 'git clone' command and a 'Copy' button). At the bottom is an 'Additional details' section with a link to documentation. The top navigation bar and bottom taskbar are identical to the previous screenshot.



Conclusion:

Hence, we learnt the concept of Continuous Delivery/Continuous Integration and successfully deployed a static web application using AWS CodePipeline.

Assignment 5

Aim: To understand the Kubernetes Cluster Architecture.

LO2. To deploy single and multiple container applications and manage application deployments with rollouts in Kubernetes.

Theory:

Kubernetes is an open-source container management tool that automates container deployment, scaling & load balancing.

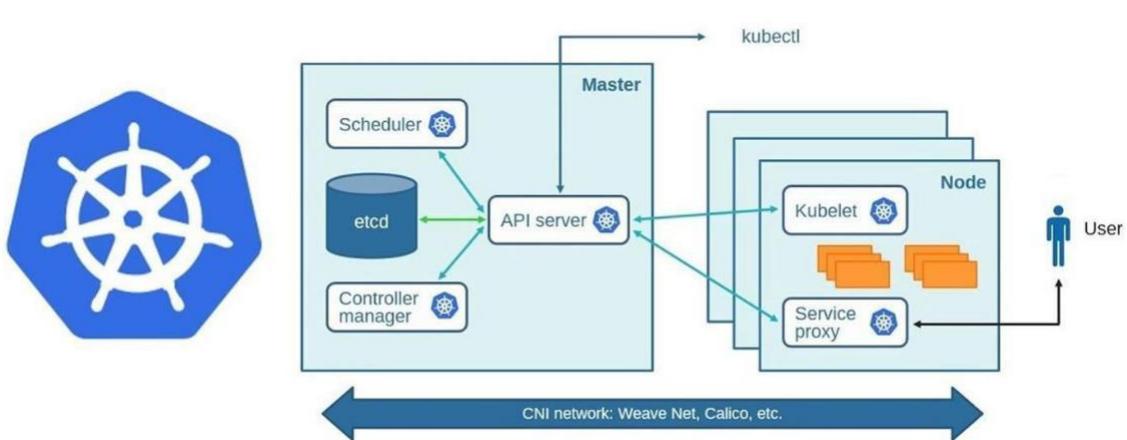
It schedules, runs, and manages isolated containers that are running on virtual/physical/cloud machines.

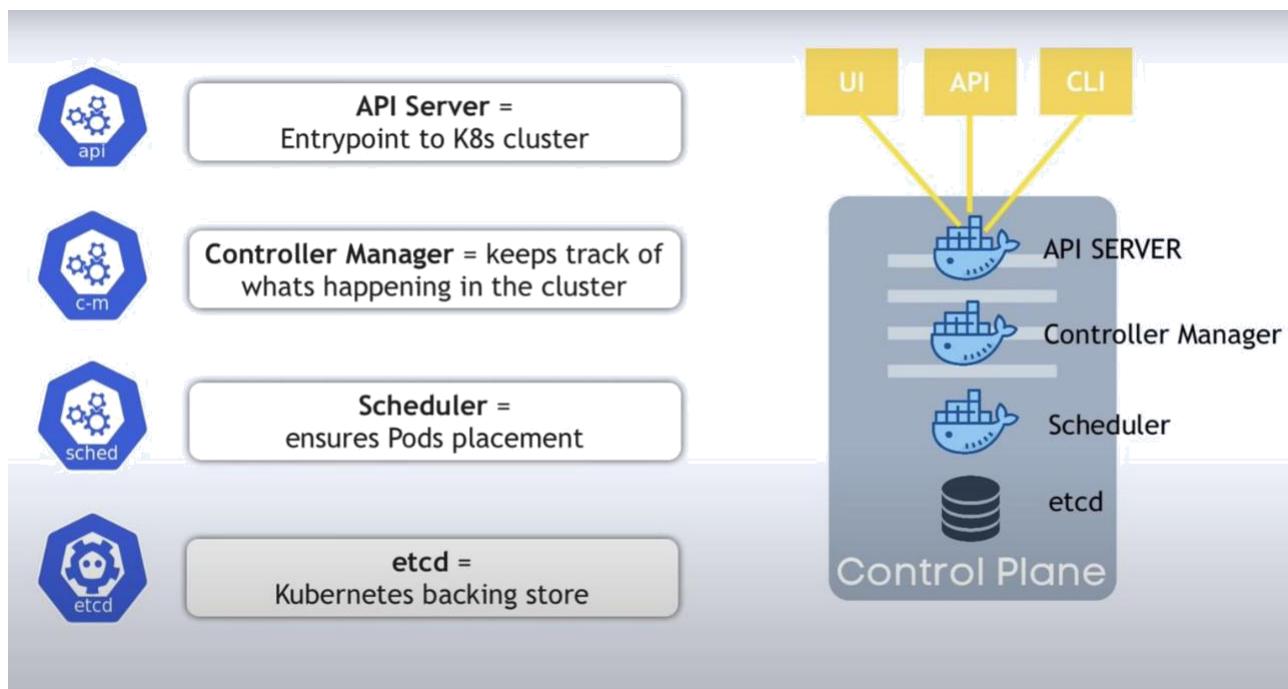
All top cloud providers support Kubernetes.

One popular name for Kubernetes is K8s.

ARCHITECTURE

Kubernetes





Working with Kubernetes

- We create a Manifest (.yml) file
- Apply those to cluster (to master) to bring it into the desired state.
- POD runs on a node, which is controlled by the master.

• Role of Master Node

- Kubernetes cluster contains containers running on Bare Metal / VM instances/cloud instances/ all mix.
- Kubernetes designates one or more of these as masters and all others as workers.
- The master is now going to run a set of K8s processes. These processes will ensure the smooth functioning of the cluster. These processes are called the 'Control Plane'.
- Can be Multi-Master for high availability.
- Master runs control plane to run cluster smoothly.

• Components of Control Plane

■ Kube-api-server → (For all communications)

- This api-server interacts directly with the user (i.e we apply .yml or .json manifest to kube-api-server)
- This kube-api-server is meant to scale automatically as per load.
- Kube-api-server is the front end of the control plane.

■ etcd

- Stores metadata and status of the cluster.
- etcd is a consistent and high-available store (key-value-store)

- Source of truth for cluster state (info about the state of the cluster)

→ **etcd has the following features**

1. Fully Replicated → The entire state is available on every node in the cluster.
2. Secure → Implements automatic TLS with optional client-certificate authentication.
3. Fast → Benchmarked at 10,000 writes per second.

■ **Kube-scheduler (action)**

- When users request the creation & management of Pods, Kube-scheduler is going to take action on these requests.
- Handles POD creation and Management.
- Kube-scheduler match/assign any node to create and run pods.
- A scheduler watches for newly created pods that have no node assigned. For every pod that the scheduler discovers, the scheduler becomes responsible for finding the best node for that pod to run.
- The scheduler gets the information for hardware configuration from configuration files and schedules the Pods on nodes accordingly.

■ **Controller-Manager**

- Make sure the actual state of the cluster matches the desired state.

→ Two possible choices for controller manager—

1. If K8s is on the cloud, then it will be a cloud controller manager.
2. If K8s is on non-cloud, then it will be kube-controller-manager.

Components on the master that runs the controller

Node Controller → For checking the cloud provider to determine if a node has been detected in the cloud after it stops responding.

Route-Controller → Responsible for setting up a network, and routes on your cloud.

Service-Controller → Responsible for load Balancers on your cloud against services of type Load Balancer.

Volume-Controller → For creating, attaching, and mounting volumes and interacting with the cloud provider to orchestrate volume.

■ **Nodes (Kubelet and Container Engine)**

- Node is going to run 3 important pieces of software/process.

Kubelet

- The agent running on the node.
- Listens to Kubernetes master (eg- Pod creation request).
- Use port 10255.
- Send success/Fail reports to master.

Container Engine

- Works with kubelet
 - Pulling images
 - Start/Stop Containers
 - Exposing containers on ports specified in the manifest.

Kube-Proxy

- Assign IP to each pod.
 - It is required to assign IP addresses to Pods (dynamic)
 - Kube-proxy runs on each node & this makes sure that each pod will get its unique IP Address.
 - These 3 components collectively consist of 'node'.

INSTALLATION:

1. Install Docker

2. Install minikube using following commands

```

Activities Terminal - prasad@prasad-VirtualBox: ~ Oct 14 22:27
prasad@prasad-VirtualBox:~$ curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
prasad@prasad-VirtualBox:~$ sudo install minikube-linux-amd64 /usr/local/bin/minikube
[sudo] password for prasad:
prasad@prasad-VirtualBox:~$ minikube start --driver=docker
minikube v1.31.2 on Ubuntu 20.04 (vbox/amd64)
Using the docker driver based on user configuration

  Exiting due to PROVIDER_DOCKER_NEWRGP: "docker version --format <no value><no value><no value>" exit status 1: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://127.0.0.1:2375/v1.24/version": dial unix /var/run/docker.sock: connect: permission denied
  Suggestion: Add your user to the 'docker' group: 'sudo usermod -aG docker $USER && newgrp docker'
  Documentation: https://docs.docker.com/engine/install/linux-postinstall/
prasad@prasad-VirtualBox:~$ sudo usermod -aG docker $USER && newgrp docker
prasad@prasad-VirtualBox:~$ minikube start --driver=docker
minikube v1.31.2 on Ubuntu 20.04 (vbox/amd64)
Using the docker driver based on user configuration

  The requested memory allocation of 1971MB does not leave room for system overhead (total system memory: 1971MB). You may face stability issues.
  Suggestion: Start minikube with less memory allocated: 'minikube start --memory=1971mb'

Using Docker driver with root privileges
Starting control plane node minikube in cluster minikube
Pulling base image ...
  Downloading Kubernetes v1.27.4 preload ...
    > preloaded-images-k8s-v18-v1...: 393.21 MB / 393.21 MB 100.00% 2.85 MiB
    > gcr.io/k8s-minikube/kicbase...: 447.62 MB / 447.62 MB 100.00% 2.99 MiB
      Creating docker container (CPUs=2, Memory=1971MB) ...

Docker is nearly out of disk space, which may cause deployments to fail! (94% of capacity). You can pass '--force' to skip this check.
Suggestion:

Try one or more of the following to free up space on the device:
  1. Run "docker system prune" to remove unused Docker data (optionally with "-a")
  2. Increase the storage allocated to Docker for Desktop by clicking on:
    Docker icon > Preferences > Resources > Disk Image Size
  3. Run "minikube ssh -- docker system prune" if using the Docker container runtime
  Related issue: https://github.com/kubernetes/minikube/issues/9024

  This container is having trouble accessing https://registry.k8s.io
  To pull new external images, you may need to configure a proxy: https://minikube.sigs.k8s.io/docs/reference/networking/proxy/
  Preparing Kubernetes v1.27.4 on Docker 24.0.4 ...
  ■ Generating certificates and keys ...
  ■ Booting up control plane ...
  ■ Configuring CNI (Container Networking Interface) ...
  ■ Using image gcr.io/k8s-minikube/storage-provisioner:v5
  Verifying Kubernetes components...
  Enabled addons: default-storageclass, storage-provisioner
  kubectl not found. If you need it, try: 'minikube kubectl -- get pods -A'
  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
prasad@prasad-VirtualBox:~$ 
prasad@prasad-VirtualBox:~$ 

Activities Terminal - prasad@prasad-VirtualBox: ~ Oct 14 22:27
prasad@prasad-VirtualBox:~$ curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
prasad@prasad-VirtualBox:~$ sudo install minikube-linux-amd64 /usr/local/bin/minikube
[sudo] password for prasad:
prasad@prasad-VirtualBox:~$ minikube start --driver=docker
minikube v1.31.2 on Ubuntu 20.04 (vbox/amd64)
Using the docker driver based on user configuration

  The requested memory allocation of 1971MB does not leave room for system overhead (total system memory: 1971MB). You may face stability issues.
  Suggestion: Start minikube with less memory allocated: 'minikube start --memory=1971mb'

Using Docker driver with root privileges
Starting control plane node minikube in cluster minikube
Pulling base image ...
  Downloading Kubernetes v1.27.4 preload ...
    > preloaded-images-k8s-v18-v1...: 393.21 MB / 393.21 MB 100.00% 2.85 MiB
    > gcr.io/k8s-minikube/kicbase...: 447.62 MB / 447.62 MB 100.00% 2.99 MiB
      Creating docker container (CPUs=2, Memory=1971MB) ...

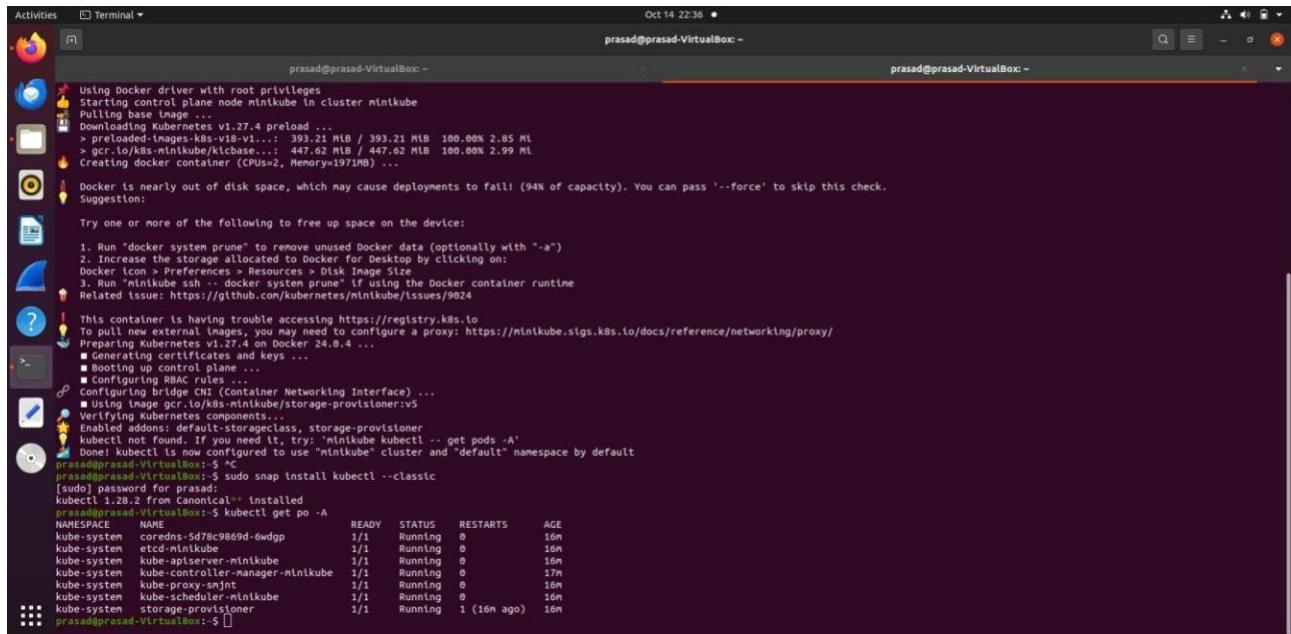
Docker is nearly out of disk space, which may cause deployments to fail! (94% of capacity). You can pass '--force' to skip this check.
Suggestion:

Try one or more of the following to free up space on the device:
  1. Run "docker system prune" to remove unused Docker data (optionally with "-a")
  2. Increase the storage allocated to Docker for Desktop by clicking on:
    Docker icon > Preferences > Resources > Disk Image Size
  3. Run "minikube ssh -- docker system prune" if using the Docker container runtime
  Related issue: https://github.com/kubernetes/minikube/issues/9024

  This container is having trouble accessing https://registry.k8s.io
  To pull new external images, you may need to configure a proxy: https://minikube.sigs.k8s.io/docs/reference/networking/proxy/
  Preparing Kubernetes v1.27.4 on Docker 24.0.4 ...
  ■ Generating certificates and keys ...
  ■ Booting up control plane ...
  ■ Configuring CNI (Container Networking Interface) ...
  ■ Using image gcr.io/k8s-minikube/storage-provisioner:v5
  Verifying Kubernetes components...
  Enabled addons: default-storageclass, storage-provisioner
  kubectl not found. If you need it, try: 'minikube kubectl -- get pods -A'
  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
prasad@prasad-VirtualBox:~$ 
prasad@prasad-VirtualBox:~$ 

```

3. Install kubectl



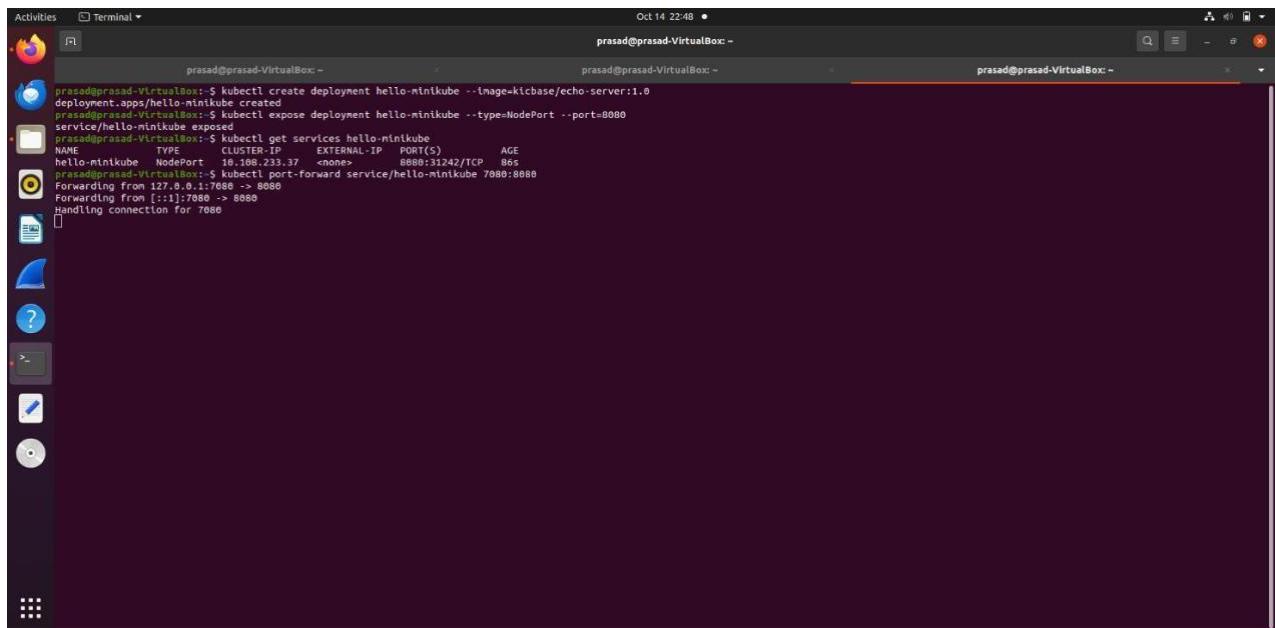
A screenshot of a Linux desktop environment showing a terminal window titled "Terminal". The terminal window has three tabs, all labeled "prasad@prasad-VirtualBox: ~". The first tab contains logs from the minikube setup process. It shows Docker driver loading, Kubernetes v1.27.4 preloading, and Docker being nearly out of disk space. The second tab shows a warning about Docker storage configuration. The third tab shows the user running "sudo snap install kubectl --classic" and then listing pods with "kubectl get po -A". The output shows several pods in the "kube-system" namespace, mostly in a "Running" state.

```
Using Docker driver with root privileges
Starting control plane node minikube in cluster minikube
Pulling base image ...
Downloading Kubernetes v1.27.4 preload ...
> gcr.io/k8s-minikube/kicbase...: 393.21 MiB / 393.21 MiB 100.0% 2.85 MiB
> gcr.io/k8s-minikube/kicbase...: 447.62 MiB / 447.62 MiB 100.0% 2.99 MiB
Creating docker container (CPUs=2, Memory=1973MiB) ...
Docker is nearly out of disk space, which may cause deployments to fail! (94% of capacity). You can pass '--force' to skip this check.
Suggestion:

Try one or more of the following to free up space on the device:
1. Run "docker system prune" to remove unused Docker data (optionally with "-a")
2. Increase the storage allocated to Docker for desktop by clicking on:
Docker icon > Preferences > Resources > Disk Image Size
3. Run "minikube ssh -- docker system prune" if using the Docker container runtime
Related issue: https://github.com/kubernetes/minikube/issues/9024

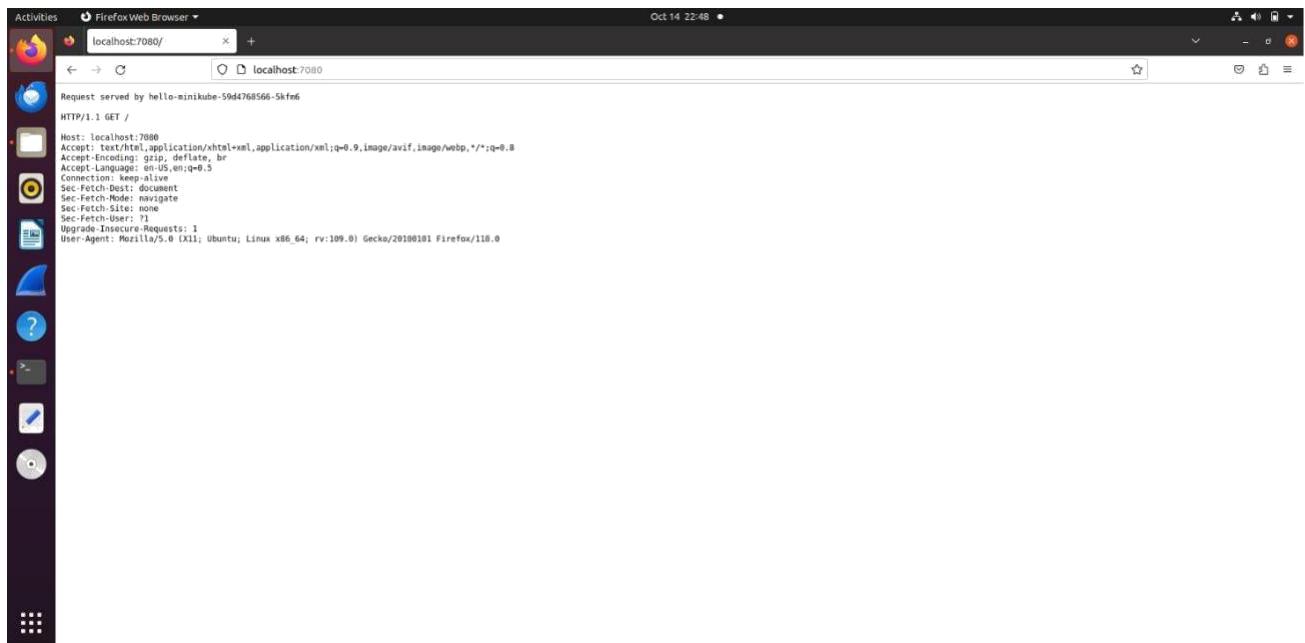
This container is having trouble accessing https://registry.k8s.io
To use new external Images, you may need to configure a proxy: https://minikube.sigs.k8s.io/docs/reference/networking/proxy/
Preparing Kubernetes v1.27.4 on Docker 24.0.4 ...
  Generating certificates and keys ...
  Booting up control plane...
  Configuring RBAC rules ...
  Configuring bridge CNI (Container Networking Interface) ...
  Using image gcr.io/k8s-minikube/storage-provisioner:v5
Verifying Kubernetes components...
kubectl not found. If you need it, try: 'minikube kubectl -- get pods -A'
Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
prasad@prasad-VirtualBox:~$ sudo snap install kubectl --classic
[sudo] password for prasad:
kubectl 1.28.2 from Canonical - Installed
prasad@prasad-VirtualBox:~$ kubectl get po -A
NAMESPACE     NAME           READY   STATUS    RESTARTS   AGE
kube-system   coredns-5d7c98e9d-gwdgp   1/1    Running   0          16m
kube-system   etcd-minikube        1/1    Running   0          16m
kube-system   kube-apiserver-minikube  1/1    Running   0          16m
kube-system   kube-controller-manager-minikube  1/1    Running   0          17m
kube-system   kube-proxy-smjnt      1/1    Running   0          16m
kube-system   kube-scheduler-minikube  1/1    Running   0          16m
kube-system   storage-provisioner   1/1    Running   1 (16m ago)  16m
prasad@prasad-VirtualBox:~$
```

4. Create a sample deployment.



A screenshot of a Linux desktop environment showing a terminal window titled "Terminal". The terminal window has three tabs, all labeled "prasad@prasad-VirtualBox: ~". The first tab shows the creation of a deployment named "hello-minikube" with the command "kubectl create deployment hello-minikube --image=kicbase/echo-server:1.0". The second tab shows exposing the deployment with "kubectl expose deployment hello-minikube --type=NodePort --port=8080", resulting in a service named "hello-minikube" being exposed. The third tab shows port forwarding from host port 7080 to service port 8080 with "kubectl port-forward service/hello-minikube 7080:8080". The output shows the connection being established between the host and the service.

```
prasad@prasad-VirtualBox:~$ kubectl create deployment hello-minikube --image=kicbase/echo-server:1.0
deployment.apps/hello-minikube created
prasad@prasad-VirtualBox:~$ kubectl expose deployment hello-minikube --type=NodePort --port=8080
service/hello-minikube exposed
prasad@prasad-VirtualBox:~$ kubectl get services hello-minikube
NAME            TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
hello-minikube  NodePort    10.108.233.37  <none>       8080:31242/TCP  86s
prasad@prasad-VirtualBox:~$ kubectl port-forward service/hello-minikube 7080:8080
Forwarding from 127.0.0.1:7080 -> 8080
Forwarding from [::1]:7080 -> 8080
Handling connection for 7080
```



CONCLUSION:

Here we studied Kubernetes cluster architecture in detail. Also we installed Kubernetes in ubuntu machine and created a sample deployment.

Assignment 6

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

LO3: To apply best practices for managing infrastructure as code environments and use terraform to define and deploy cloud infrastructure.

Theory:

Terraform is one of the most popular Infrastructure-as-code (IaC) tool, used by DevOps teams to automate infrastructure tasks. It is used to automate the provisioning of your cloud resources. Terraform is an open-source, cloud-agnostic provisioning tool developed by HashiCorp and written in GO language.

Benefits of Terraform:

- Does orchestration, not just configuration management.
- Supports multiple providers such as AWS, Azure, Oracle, GCP, and many more.
- Provide immutable infrastructure where configuration changes smoothly.
- Uses easy to understand language, HCL (HashiCorp configuration language).
- Easily portable to any other provider.

TERRAFORM LIFECYCLE

Terraform lifecycle consists of – **init**, **plan**, **apply**, and **destroy**.



1. **Terraform init** initializes the (local) Terraform environment. Usually executed only once per session.
2. **Terraform plan** compares the Terraform state with the as-is state in the cloud, build and display an execution plan. This does not change the deployment (read-only).
3. **Terraform apply** executes the plan. This potentially changes the deployment.
4. **Terraform destroy** deletes all resources that are governed by this specific terraform environment.

TERRAFORM CORE CONCEPTS/TERMINOLOGIES

1. Variables: Terraform has input and output variables, it is a key-value pair. Input variables are used as parameters to input values at run time to customize our deployments. Output variables are return values of a terraform module that can be used by other configurations.
2. Provider: Terraform users provision their infrastructure on the major cloud providers such as AWS, Azure, OCI, and others. A provider is a plugin that interacts with the various APIs required to create, update, and delete various resources.
3. Module: Any set of Terraform configuration files in a folder is a module. Every Terraform configuration has at least one module, known as its root module.
4. State: Terraform records information about what infrastructure is created in a Terraform state file. With the state file, Terraform is able to find the resources it created previously, supposed to manage and update them accordingly.
5. Resources: Cloud Providers provides various services in their offerings, they are referenced as Resources in Terraform. Terraform resources can be anything from compute instances, virtual networks to higher-level components such as DNS records. Each resource has its own attributes to define that resource.
6. Data Source: Data source performs a read-only operation. It allows data to be fetched or computed from resources/entities that are not defined or managed by Terraform or the current Terraform configuration.
7. Plan: It is one of the stages in the Terraform lifecycle where it determines what needs to be created, updated, or destroyed to move from the real/current state of the infrastructure to the desired state.
8. Apply: It is one of the stages in the Terraform lifecycle where it applies the changes real/current state of the infrastructure in order to achieve the desired state.

INSTALLATION:

1) Download Terraform

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

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

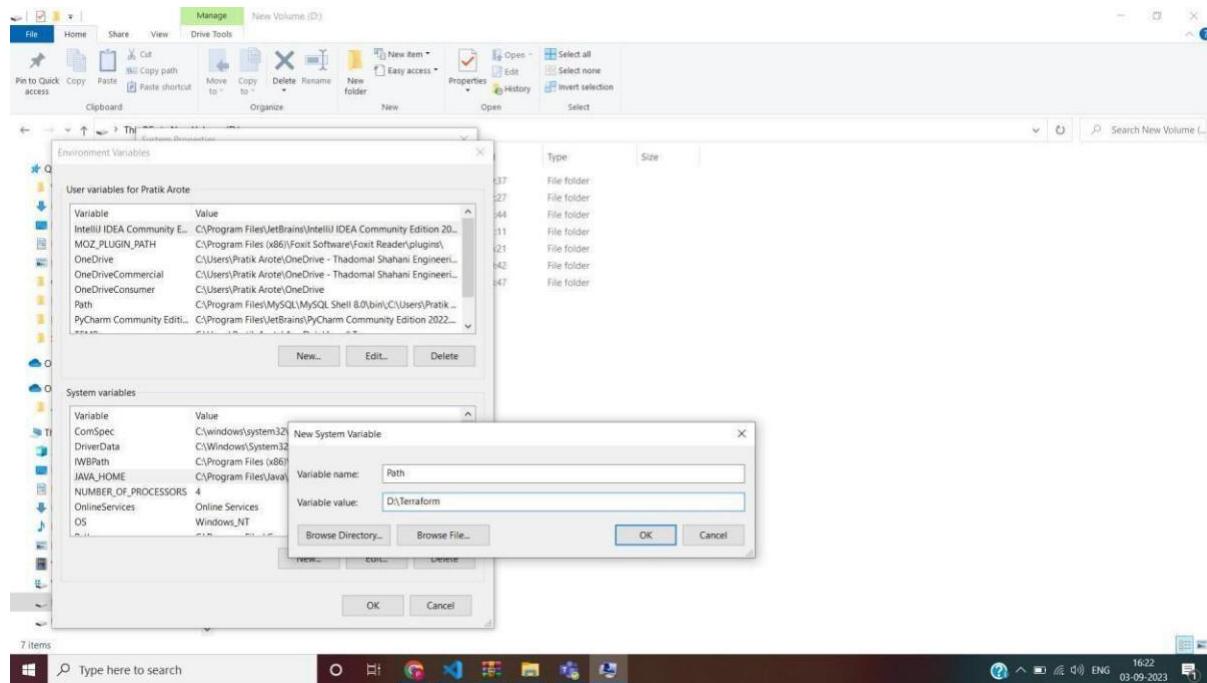
Select the Operating System Windows followed by either 32bit or 64 bit based on your OS type.

The screenshot shows the Terraform website's 'Install Terraform' section. It features a navigation bar with links like 'Terraform', 'Install', 'Tutorials', 'Documentation', 'Registry', and 'Try Cloud'. Below the navigation is a search bar and a 'Search' button. The main content area has tabs for 'Operating System' (Windows selected) and other platforms like macOS, Linux, FreeBSD, OpenBSD, and Solaris. Under 'Binary download for Windows', there are two options: '32bit' (Version: 1.5.6) and 'AMD64' (Version: 1.5.6), each with a 'Download' button. Below these are sections for 'Release information' (Changelog, Version: 1.5.6) and 'Notes'. A sidebar on the right is titled 'About Terraform' and includes links to 'Introduction to Terraform', 'Configuration Language', 'Terraform CLI', 'Terraform Cloud', and 'Provider Use'. Another sidebar titled 'Terraform Cloud' offers a free trial.

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

The screenshot shows a Windows File Explorer window with a context menu open over a zip file named 'terraform_1.5.6_windows_amd64.zip'. The menu option 'Extract here' is being selected. A 'File Extraction' dialog box is displayed, showing the 'Destination path' as 'C:\Terraform'. The 'General' tab is selected, with 'Extract and replace files' checked. Other tabs like 'Advanced' and 'Options' are visible. In the background, the desktop environment is visible with various icons and a taskbar at the bottom.

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



4. Open PowerShell with Admin Access. Open Terraform in PowerShell and check its functionality.

```

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

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

PS C:\Users\Pratik Arote> cd D:
PS D:\> 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
rm        Reformat your configuration in the standard style
force-unlock Release a stuck lock on the current workspace
get       Install or upgrade remote module dependencies
graph    Generate a dependency graph of the state 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 Show metadata 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 resources as not fully functional
test    Experimental support for module integration testing
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 D:\>

```

Conclusion:

Here, we studied the about the terraform lifecycle and terminologies/concepts of terraform and installed it on our system.

Assignment 7

Aim: To perform static analysis on Python programs using SonarQube SAST process.

LO4: To identify and remediate application vulnerabilities earlier and help integrate security in the development process using SAST Techniques.

Theory:

SonarQube:

Overview: SonarQube is an open-source platform for continuous inspection of code quality. It is used to analyze and measure code quality and security issues in a codebase.

Features:

Static Code Analysis: SonarQube scans source code to identify bugs, code smells, and security vulnerabilities.

Continuous Integration: It integrates seamlessly with CI/CD pipelines, providing automated code analysis during the development process.

Security Analysis: While it primarily focuses on code quality, it also has some security rules to catch common security issues.

Maintainability Metrics: SonarQube provides maintainability metrics and helps teams understand code complexity and maintainability.

Dashboard and Reporting: It offers dashboards and reports for tracking code quality and issues over time.

Use Case: SonarQube is used for improving code quality, maintainability, and to catch some common code security issues. It's more about general code quality and development best practices.

SAST (Static Application Security Testing):

Overview: SAST is a security testing method that analyzes source code, bytecode, or binary code for vulnerabilities without executing the application. It is primarily focused on identifying security issues and vulnerabilities in the code.

Features:

Code Scanning: SAST tools examine the source code or compiled code to identify potential security vulnerabilities, such as SQL injection, cross-site scripting, and more.

Early Detection: SAST is used early in the development process to find security issues before they can be exploited.

Language Support: SAST tools support various programming languages and frameworks.

Integration: They can be integrated into CI/CD pipelines to automatically scan code before deployment.

Use Case: SAST is used for finding and fixing security vulnerabilities in code. It helps secure applications by identifying potential security threats early in the development lifecycle.

1. INSTALL sonarqube (docker images) and sonarscanner zip file from <https://docs.sonarsource.com/sonarqube/latest/analyzing-sourcecode/scanners/sonarscanner/> and set up config file as given in docs.

```
Command Prompt

Microsoft Windows [Version 10.0.22621.2428]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Pratik Arote>docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
43f89b94cd7d: Pull complete
58431c77a77b: Pull complete
d4d8e860e672: Pull complete
637e2db99ae6: Pull complete
7de1c2853278: Pull complete
d2152ffce821: Pull complete
519cf218564f: Pull complete
Digest: sha256:c6c8096375002d4cb2ef64b89a2736ad572812a87a2917d92e7e59384b9f6f65
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest

What's Next?
View a summary of image vulnerabilities and recommendations * docker scout quickview sonarqube

C:\Users\Pratik Arote>docker pull sonarsource/sonar-scanner-cli
Using default tag: latest
latest: Pulling from sonarsource/sonar-scanner-cli
9398888236ff: Pull complete
4f4fb788ef50: Pull complete
3cd77fb28e46: Pull complete
f78b288abc31: Pull complete
Digest: sha256:494ecc3b5b6ee1625bd377b3985c4284e4f0cc165cff397885a244dee1c7d575
Status: Downloaded newer image for sonarsource/sonar-scanner-cli:latest
docker.io/sonarsource/sonar-scanner-cli:latest

What's Next?
View a summary of image vulnerabilities and recommendations * docker scout quickview sonarsource/sonar-scanner-cli
```

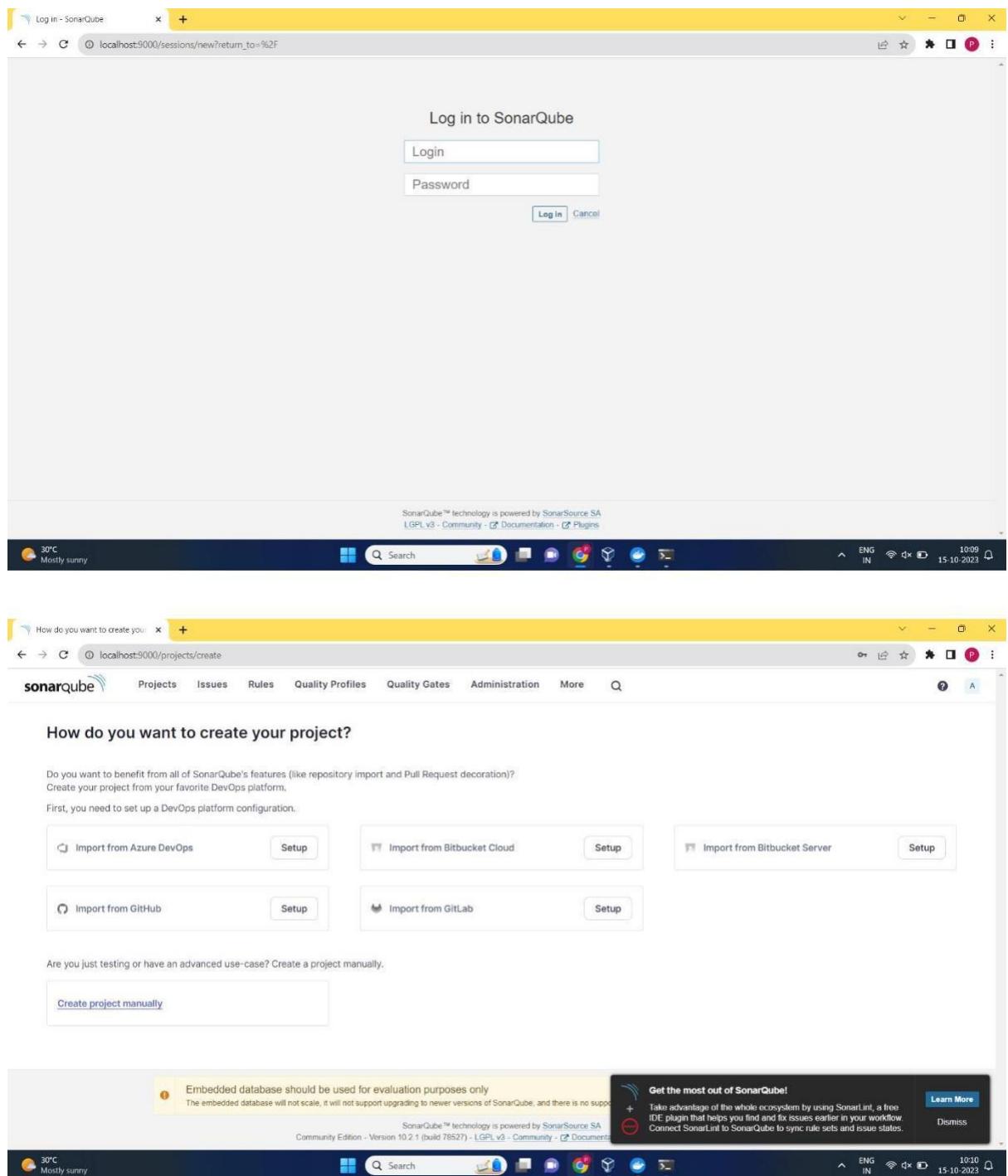
2. Spin up the container

```
C:\Users\Pratik Arote>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
f3630dbc2ffa6e5598ad922085026400a1f9f1564416b0606b5348000f6d1377

C:\Users\Pratik Arote>docker images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
sonarqube          latest   3183d6818c6e  42 hours ago  716MB
sample-web-app     latest   713c7cdaaf78  2 weeks ago   42.7MB
myimage             latest   438bb56a50a3  2 weeks ago   122MB
sonarsource/sonar-scanner-cli  latest   2f384fb1bbd5  5 weeks ago   358MB
ubuntu              latest   c6b84b685f35  8 weeks ago   77.8MB
hello-world         latest   9c7a54a9a43c  5 months ago  13.3kB

C:\Users\Pratik Arote>docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                 NAMES
f3630dbc2ffa   sonarqube:latest        "/opt/sonarqube/dock..."   27 minutes ago    Up 27 minutes    0.0.0.0:9000->9000/tcp   sonarqube
```

3. Open <http://localhost:9000> on the browser. Enter login and password both as "admin" and then set up new password.



4. Create a project

The screenshot shows the 'Create a project' page in SonarQube. The 'Project display name' is set to 'sonarPythonProgram'. The 'Project key' is also 'sonarPythonProgram'. The 'Main branch name' is 'main'. A note at the bottom says 'The name of your project's default branch [Learn More](#)'. A 'Next' button is visible.

Get the most out of SonarQube!

SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 10.2.1 (build 9627) - LGPL v3 - Community - [Documentation](#)

Dismiss

Set up project for Clean as You Code

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

Choose the baseline for new code for this project

Use the global setting

Previous version

Any code that has changed since the previous version is considered new code.
Recommended for projects following regular versions or releases.

Define a specific setting for this project

Previous version

Any code that has changed since the previous version is considered new code.
Recommended for projects following regular versions or releases.

Number of days

Any code that has changed in the last x days is considered new code. If no action is taken on a new issue code.
Recommended for projects following continuous delivery.

Get the most out of SonarQube!

Take advantage of the whole ecosystem by using SonarLint, a free IDE plugin that helps you find and fix issues earlier in your workflow. Connect SonarLint to SonarQube to sync rule sets and issue states.

Learn More

Dismiss

SonarQube - localhost:9000/tutorials?id=sonarPythonProgram1

Congratulations! Your project has been created.

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration more

sonarPythonProgram1 / main

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

Use this page to manage and set-up the way your analyses are performed.

How do you want to analyze your repository?

- With Jenkins
- With GitHub Actions
- With Bitbucket Pipelines
- With GitLab CI
- With Azure Pipelines
- Other CI

Locally

Use this for testing or advanced use-case. Other modes are recommended to help you set up your CI environment.

Embedded database should be used for evaluation purposes only

The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for SonarLint. SonarQube™ technology is powered by SonarSource SA Community Edition - Version 10.2.1 (build 78527) - LGPL v3 - Community - Documentation

Get the most out of SonarQube!

Take advantage of the whole ecosystem by using SonarLint, a free IDE plugin that helps you find and fix issues earlier in your workflow. Connect SonarLint to SonarQube to sync rule sets and issue states.

Learn More Dismiss

30°C Mostly sunny Search ENG IN 10:23 15-10-2023

5. Provide token

SonarQube - localhost:9000/tutorials?id=sonarPythonProgram1&selectedTutorial=local

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration More

sonarPythonProgram1 / main

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

We initialized your project on SonarQube, now it's up to you to launch analyses!

1 Provide a token

pythonToken1: sqp_c8922aed03df90f4cc0dfb26200772ff42a9032b

The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point in time in your [user account](#).

Continue

2 Run analysis on your project

Embedded database should be used for evaluation purposes only

The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for SonarLint. SonarQube™ technology is powered by SonarSource SA Community Edition - Version 10.2.1 (build 78527) - LGPL v3 - Community - Documentation

Get the most out of SonarQube!

Take advantage of the whole ecosystem by using SonarLint, a free IDE plugin that helps you find and fix issues earlier in your workflow. Connect SonarLint to SonarQube to sync rule sets and issue states.

Learn More Dismiss

30°C Mostly sunny Search ENG IN 10:23 15-10-2023

2 Run analysis on your project

What option best describes your build?

Maven Gradle .NET Other (for JS, TS, Go, Python, PHP, ...)

What is your OS?

Linux Windows macOS

Download and unzip the Scanner for Windows

Visit the [official documentation of the Scanner](#) to download the latest version, and add the `\bin` directory to the `%PATHE%` environment variable.

Execute the Scanner

Running a SonarQube analysis is straightforward. You just need to execute the following commands in your project's folder.

```
sonar-scanner.bat -Dsonar.projectKey=sonarPythonProgram1 -Dsonar.sources=. -Dsonar.host.url=http://localhost:9000 -Dsonar.token=sqp_c8922aed03df90f4cc0dfb26200772ff42a9032b
```

Please visit the [official documentation of the Scanner](#) for more details.

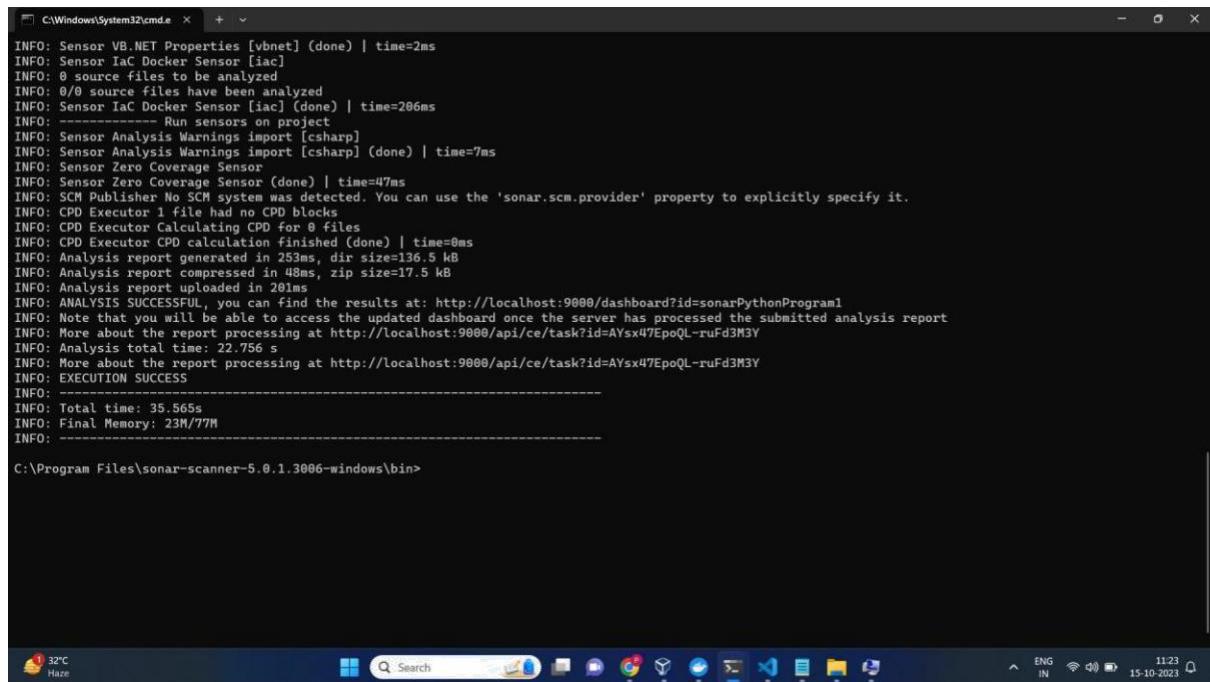


6. Enter the following command

```
C:\Windows\System32\cmd.e... + 
Microsoft Windows [Version 10.0.22621.2028]
(c) Microsoft Corporation. All rights reserved.

C:\Program Files\sonar-scanner-5.0.1.3006-windows\bin>sonar-scanner.bat -Dsonar.projectKey=sonarPythonProgram1 -Dsonar.sources=C:\Users\Pratik Arote\Desktop\sastPython -Dsonar.host.url=http://localhost:9000 -Dsonar.token=sqp_c8922aed03df90f4cc0dfb26200772ff42a9032b -Dsonar.projectBaseDir=C:\Users\Pratik Arote\Desktop\sastPython
INFO: Scanner configuration file: C:\Program Files\sonar-scanner-5.0.1.3006-windows\bin..\conf\sonar-scanner.properties
INFO: Project root configuration file: NONE
INFO: SonarScanner 5.0.1.3006
INFO: Java 17.0.7 Eclipse Adoptium (64-bit)
INFO: Windows 11 10.0 amd64
INFO: User cache: C:\Users\Pratik Arote\.sonar\cache
INFO: Analyzing on SonarQube server 18.2.1.78527
INFO: Default locale: "en_IN", source code encoding: "windows-1252" (analysis is platform dependent)
INFO: Load global settings
INFO: Load global settings (done) | time=588ms
INFO: Server id: i47B411E-AYsxoFDzoQL-ruFd2_SS
INFO: User cache: C:\Users\Pratik Arote\.sonar\cache
INFO: Load/download plugins
INFO: Load plugins index
INFO: Load plugins index (done) | time=338ms
INFO: Load/download plugins (done) | time=8251ms
INFO: Process project properties
INFO: Process project properties (done) | time=40ms
INFO: Execute project builders
INFO: Execute project builders (done) | time=7ms
INFO: Project key: sonarPythonProgram1
INFO: Base dir: C:\Users\Pratik Arote\Desktop\sastPython
INFO: Working dir: C:\Users\Pratik Arote\Desktop\sastPython\scannerwork
INFO: Load project settings for component key: 'sonarPythonProgram1'
INFO: Load project settings for component key: 'sonarPythonProgram1' (done) | time=122ms
WARN: SCM provider autodetection failed. Please use "sonar.scm.provider" to define SCM of your project, or disable the SCM Sensor in the project settings.
INFO: Load quality profiles
INFO: Load quality profiles (done) | time=597ms
INFO: Load active rules
INFO: Load active rules (done) | time=7984ms
INFO: Load analysis cache
INFO: Load analysis cache (404) | time=60ms
INFO: Load project repositories
INFO: Load project repositories (done) | time=295ms
```



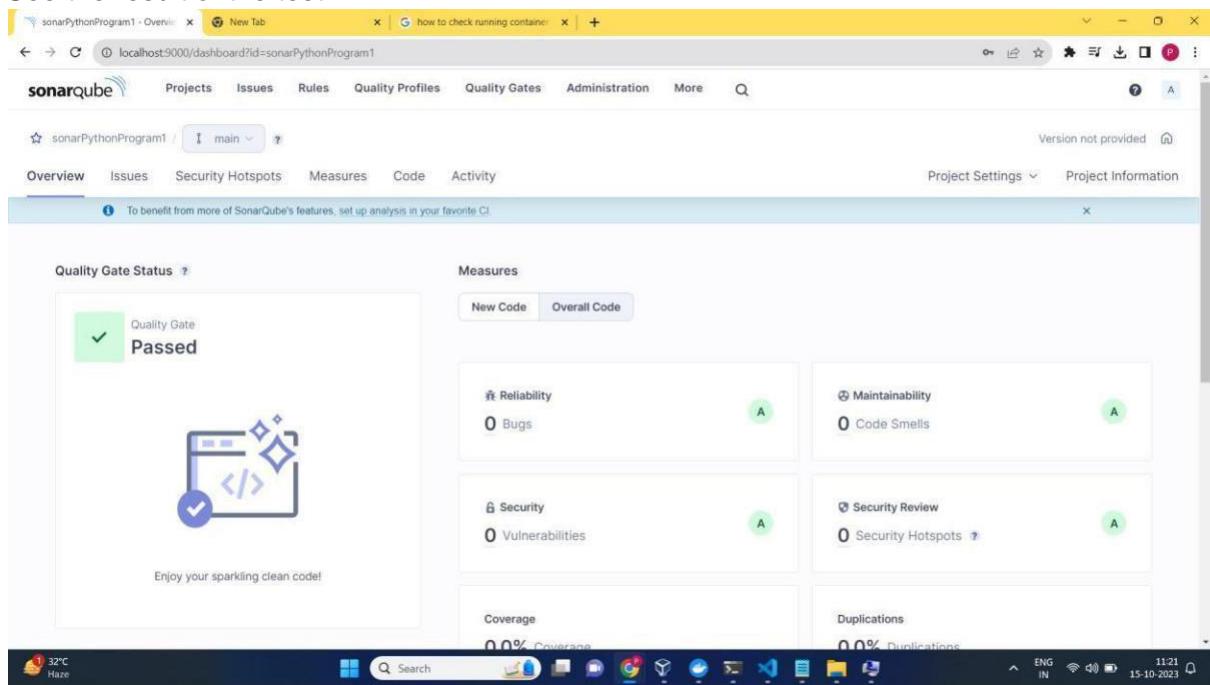


```

INFO: Sensor VB.NET Properties [vbnet] (done) | time=2ms
INFO: Sensor IaC Docker Sensor [iac]
INFO: 0 source files to be analyzed
INFO: 0/0 source files have been analyzed
INFO: Sensor IaC Docker Sensor [iac] (done) | time=206ms
INFO: -----
INFO: Run sensors on project
INFO: Sensor Analysis Warnings import [csharp]
INFO: Sensor Analysis Warnings import [csharp] (done) | time=7ms
INFO: Sensor Zero Coverage Sensor
INFO: Sensor Zero Coverage Sensor (done) | time=47ms
INFO: SCM Publisher No SCM system was detected. You can use the 'sonar.scm.provider' property to explicitly specify it.
INFO: CPD Executor 1 file had no CPD blocks
INFO: CPD Executor Calculating CPD for 0 files
INFO: CPD Executor CPD calculation finished (done) | time=0ms
INFO: Analysis report generated in 253ms, dir size=136.5 kB
INFO: Analysis report compressed in 48ms, zip size=17.5 kB
INFO: Analysis report uploaded in 201ms
INFO: ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarPythonProgram1
INFO: Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
INFO: More about the report processing at http://localhost:9000/api/ce/task?id=AYsx47EpoQL-ruFd3M3Y
INFO: Analysis total time: 22.756 s
INFO: More about the report processing at http://localhost:9000/api/ce/task?id=AYsx47EpoQL-ruFd3M3Y
INFO: EXECUTION SUCCESS
INFO: -----
INFO: Total time: 35.565s
INFO: Final Memory: 23M/77M
INFO: -----
C:\Program Files\sonar-scanner-5.0.1.3006-windows\bin>

```

7. See the result of the test



The screenshot shows the SonarQube interface for a Python project named 'sonarPythonProgram1'. The main dashboard features a large green 'Passed' badge for the Quality Gate. Below this, there are several cards providing detailed metrics: Reliability (0 Bugs, A grade), Maintainability (0 Code Smells, A grade), Security (0 Vulnerabilities, A grade), Security Review (0 Security Hotspots, A grade), Coverage (0.0% Coverage), and Duplications (0.0% Duplications). The dashboard also includes tabs for Overview, Issues, Security Hotspots, Measures, Code, and Activity, along with Project Settings and Project Information links.

Conclusion:

Here we have successfully performed static analysis of python programs.

Assignment 8

Aim: To understand Continuous Monitoring using Nagios.

LO1: To understand the fundamentals of Cloud Computing and be fully proficient with Cloud based DevOps solution deployment options to meet your business requirements.

LO5: To use Continuous Monitoring Tools to resolve any system errors (low memory, unreachable server etc.) before they have any negative impact on the business productivity.

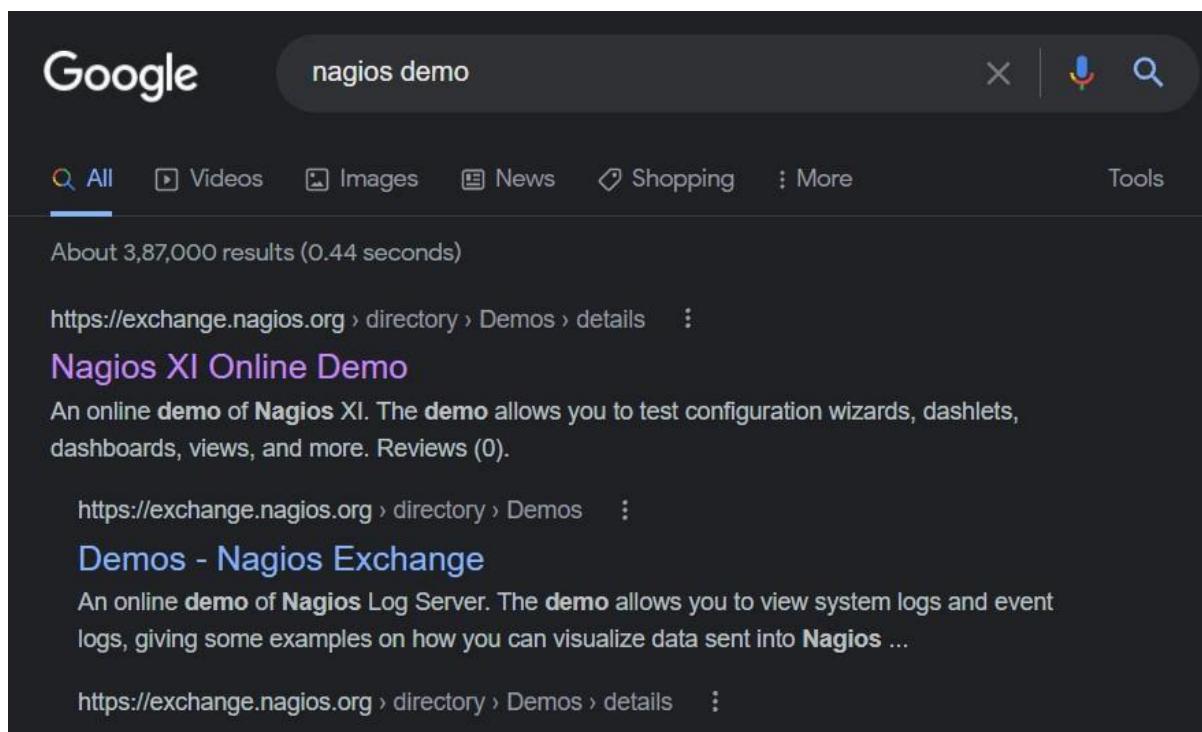
Theory:

What is Nagios and how it works?

Nagios is an open source monitoring system for computer systems. ... Nagios software runs periodic checks on critical parameters of application, network and server resources. For example, Nagios can monitor memory usage, disk usage, microprocessor load, the number of currently running processes and log files.

Steps-

Go to google.com, Search Nagios Demo



Now click on the website-

Nagios®

Network: Enterprise | Support | Library | Project | Exchange

Home Directory About

Home | Directory | Demos | Nagios XI Online Demo

Directory Tree

Nagios XI Online Demo

Submit review | Recommend | Print | Visit | Claim

Rating ★★★★★ Favoured: 0

0 votes

Owner egalstad

Website nagiosxi.demos.nagios.com

Hits 141800

Search Exchange

search... Advanced Search

Search All Sites

Go

Nagios Live Webinars

Let our experts show you how Nagios can help your organization.

Now click on login as administrator

Nagios XI Online Demo - Nagios XI Login

nagiosxi.demos.nagios.com/nagiosxi/login.php?redirect=/nagiosxi/index.php%3f&oauth=1

Nagios XI Login

Login

Username

Password

[Forgot your password?](#)

Select Language:

Nagios XI Demo System

Demo Account Options

You can access the demo with different accounts to get a different view of the monitoring system.

- Administrator Access** - An account with administrative privileges.
 Username: nagiosadmin
Password: nagiosadmin
- Read-Only User Access** - A user account that can view all hosts and services, but not make any changes or issue commands.
 Username: readonly
Password: readonly
- Advanced User Access** - An advanced user account that can see and control (schedule downtime, edit, etc) all hosts and services.
 Username: advanced
Password: advanced
- Normal User Access** - A sample "normal" user account that has rights to view only a defined subset of all hosts and services.
 Username: jdoe
Password: jdoe
- Administrator Access** - showing the dark theme.
 Username: darktheme
Password: darktheme

Demo Notes

It will have interface like this

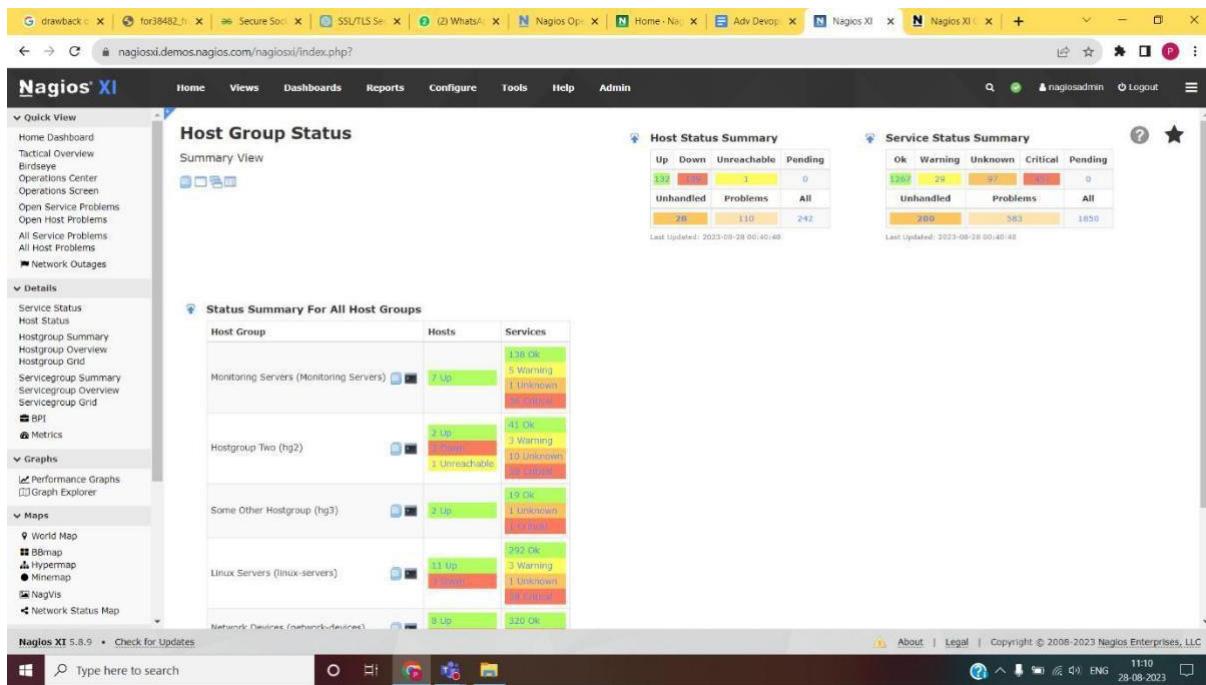
The screenshot shows the Nagios XI Home Dashboard. On the left, there's a sidebar with sections for Quick View, Details, Graphs, and Maps. The main area has three summary boxes: 'Getting Started Guide' (with tasks like changing account settings and notifications), 'Host Status Summary' (with counts for Up, Down, Unreachable, Pending, Unhandled, Problems, and All), and 'Service Status Summary' (with counts for OK, Warning, Unknown, Critical, Pending, Unhandled, Problems, and All). To the right, there's a 'We're Here To Help!' section featuring a photo of a support team member, links to Support Forum, Help Resources, Customer Ticket Support Center, and Customer Phone Support, and a 'Start Monitoring' section with links to Run a Config Wizard, Run Auto-Discovery, and Advanced Config.

Now click on Host status-

The screenshot shows the Nagios XI Host Status page. The left sidebar is identical to the Home Dashboard. The main content area displays a table titled 'Host Status' for 'All hosts'. The table includes columns for Host, Status, Duration, Attempt, Last Check, and Status Information. Each row shows the host name, its current status (Up or Down), the duration it has been in that state, the number of attempts, the last check time, and a detailed status information message. The table shows 242 total records. At the bottom, there are navigation links for About, Legal, Copyright, and a date stamp (28-08-2023).

In the above image one can see Host Status Summary and Service Status Summary also how many host are up, down and also errors in detail Now

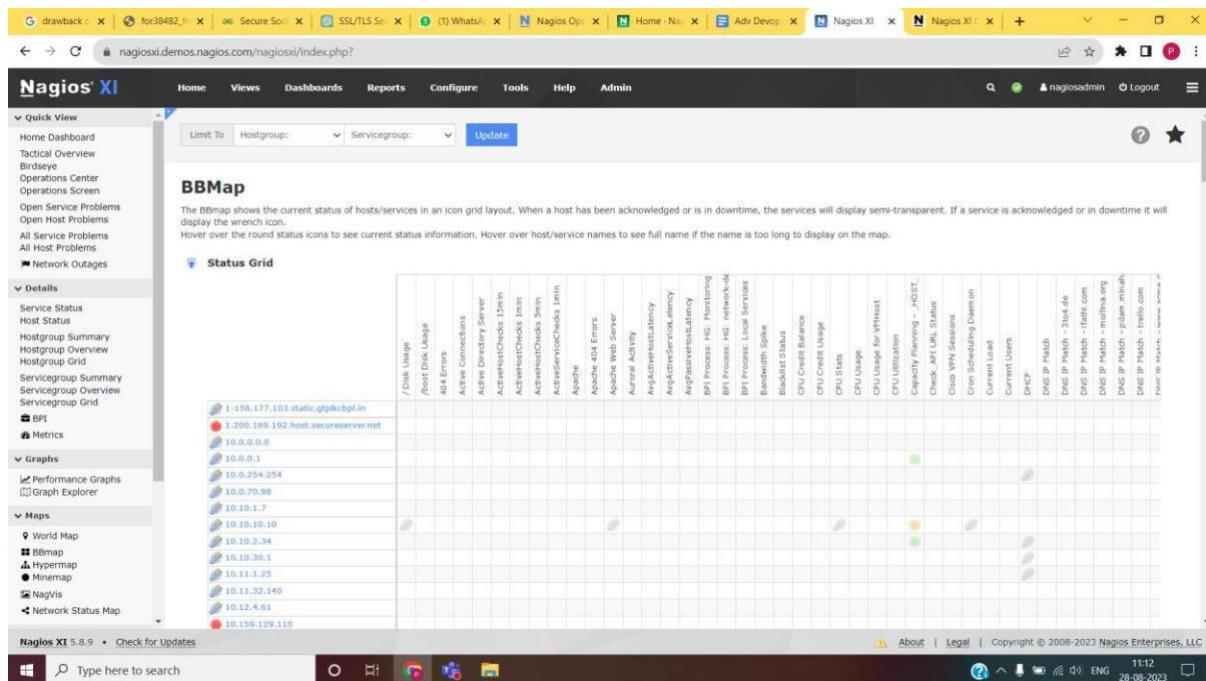
click on Host Group Status.



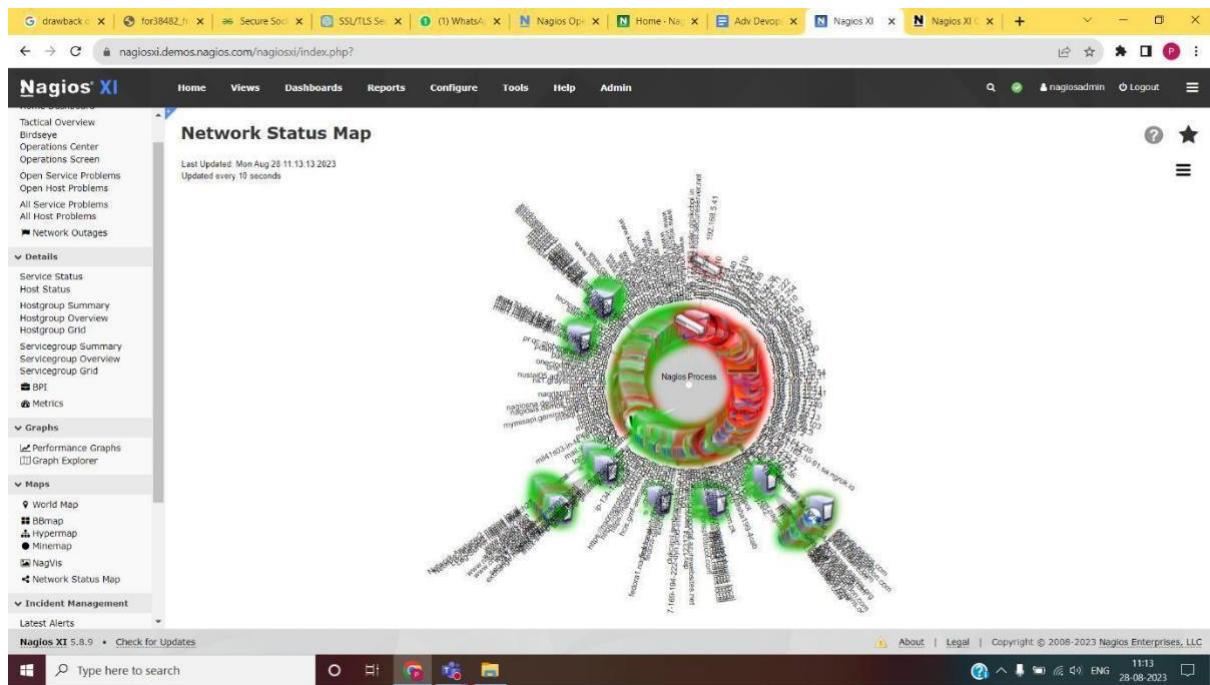
Here we can see Status Summary for All Host Groups

Now we click on BBMap

In this we can see status of following stuff in each host-



Now we have Network status map which is graphical representation of the network status



Conclusion:

Hence, we understood Nagios. It is a powerful monitoring tool, provided valuable insights into its capabilities and benefits for effective system monitoring and management.

Assignment 9

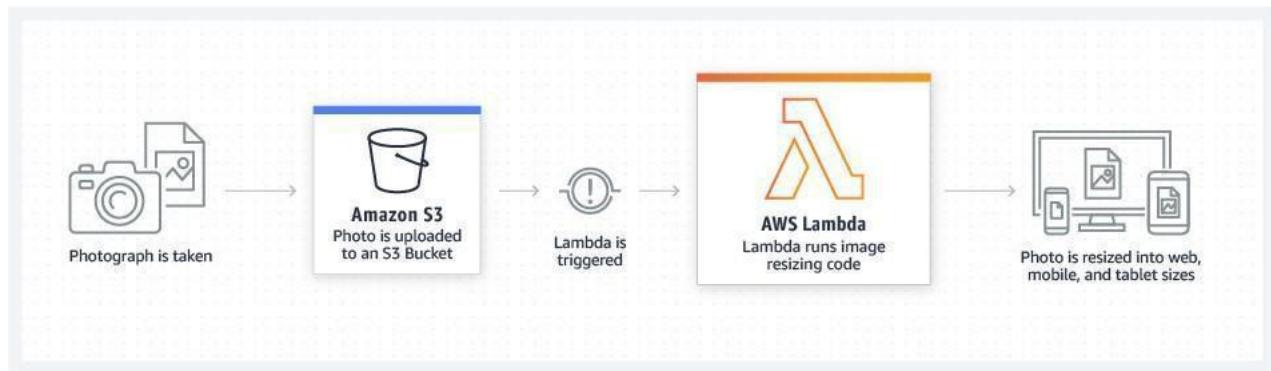
Aim: To understand AWS Lambda functions and create a Lambda function using Python to log “An Image has been added” message once a file is added to a S3 bucket.

LO6: To engineer a composition of nano services using AWS Lambda and Step Functions with the Serverless Framework.

Theory:

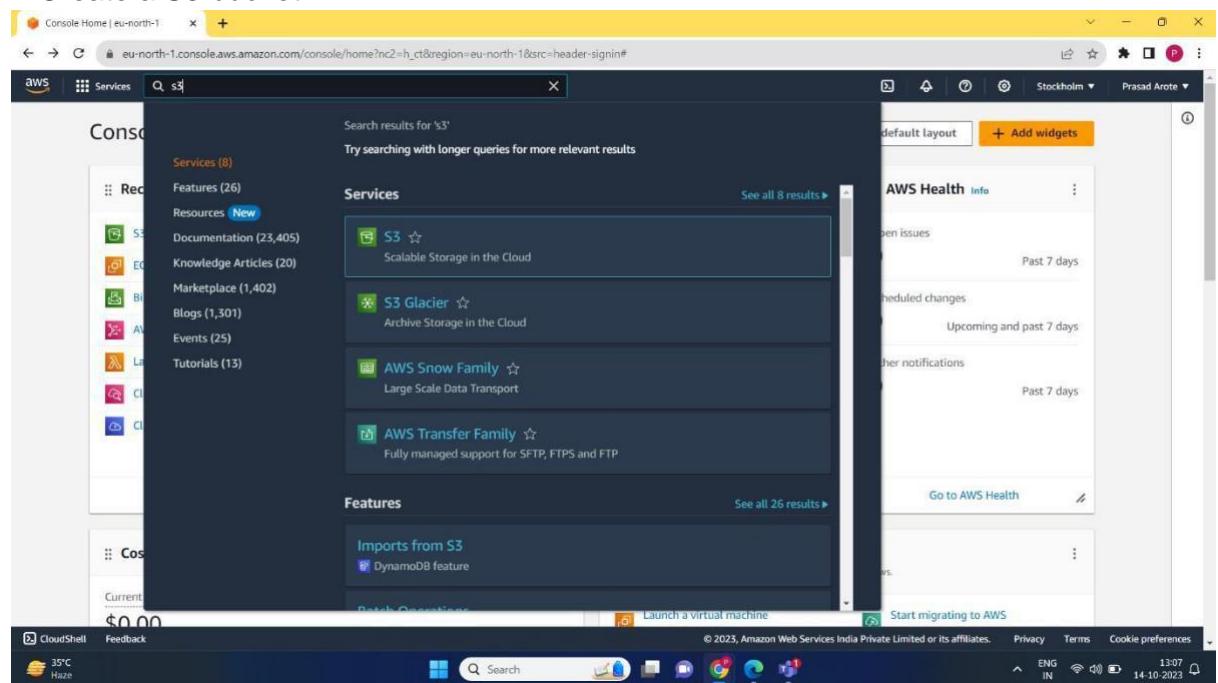
LAMBDA FUNCTION

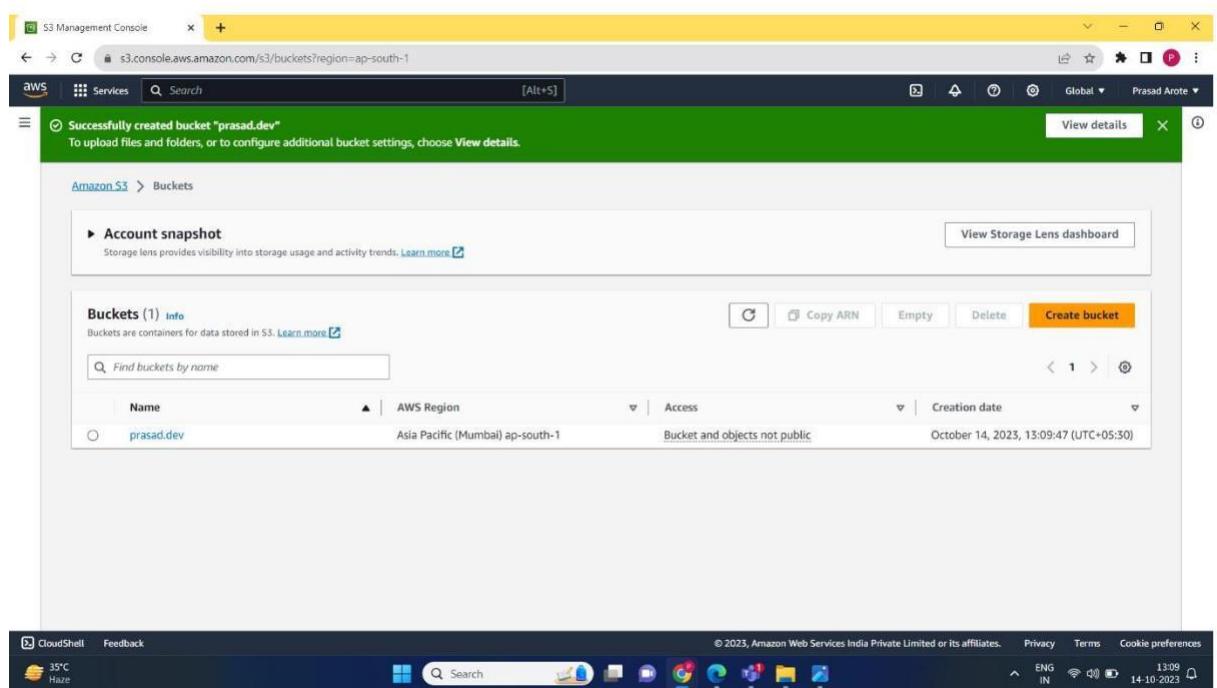
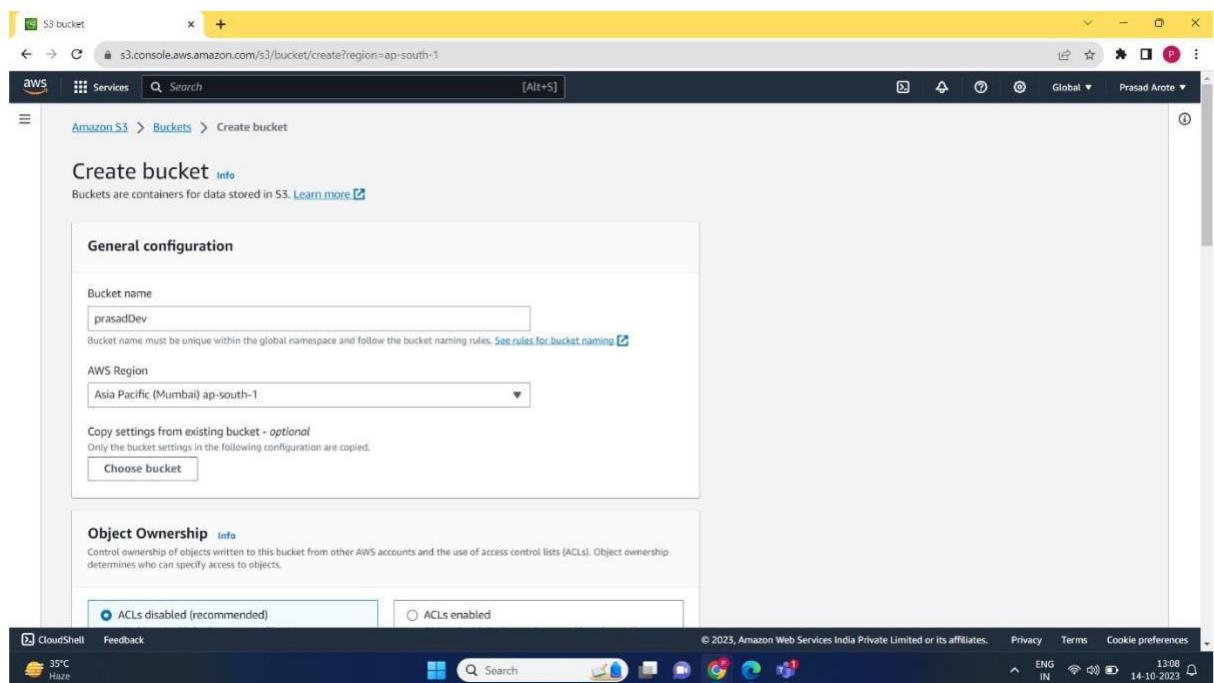
AWS Lambda is a serverless, event-driven compute service that lets you run code for virtually any type of application or backend service without provisioning or managing servers. You can trigger Lambda from over 200 AWS services and software as a service (SaaS) applications, and only pay for what you use.



Installation:

1. Create a S3 bucket

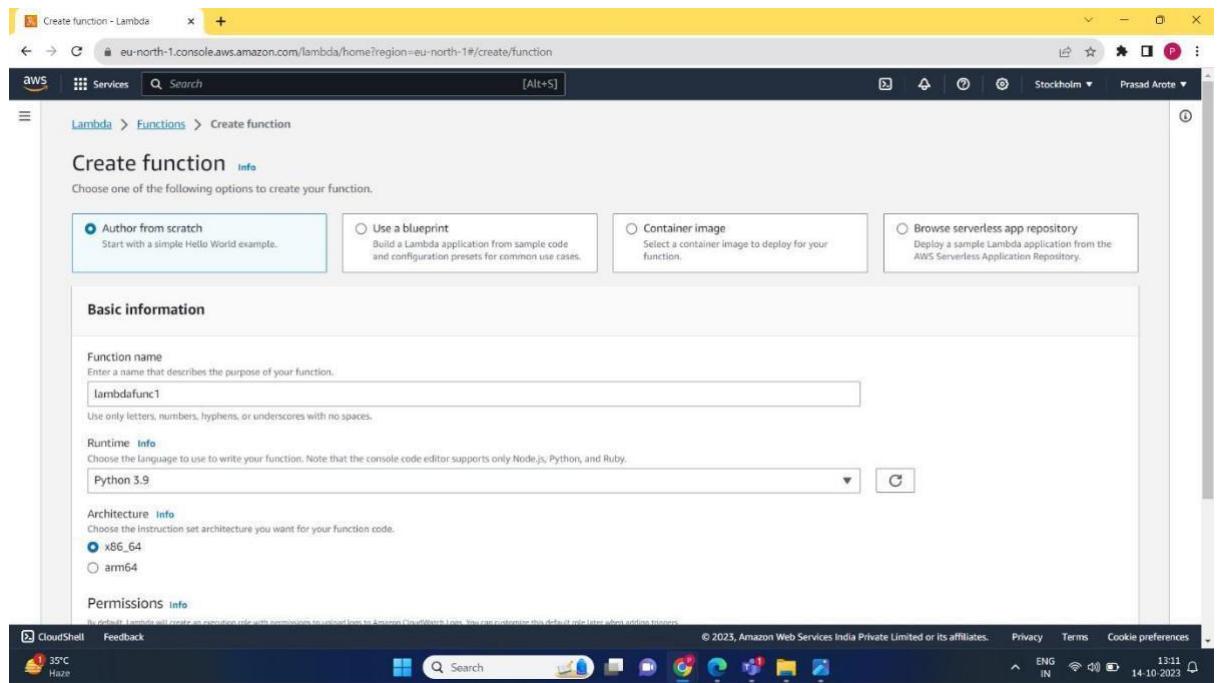




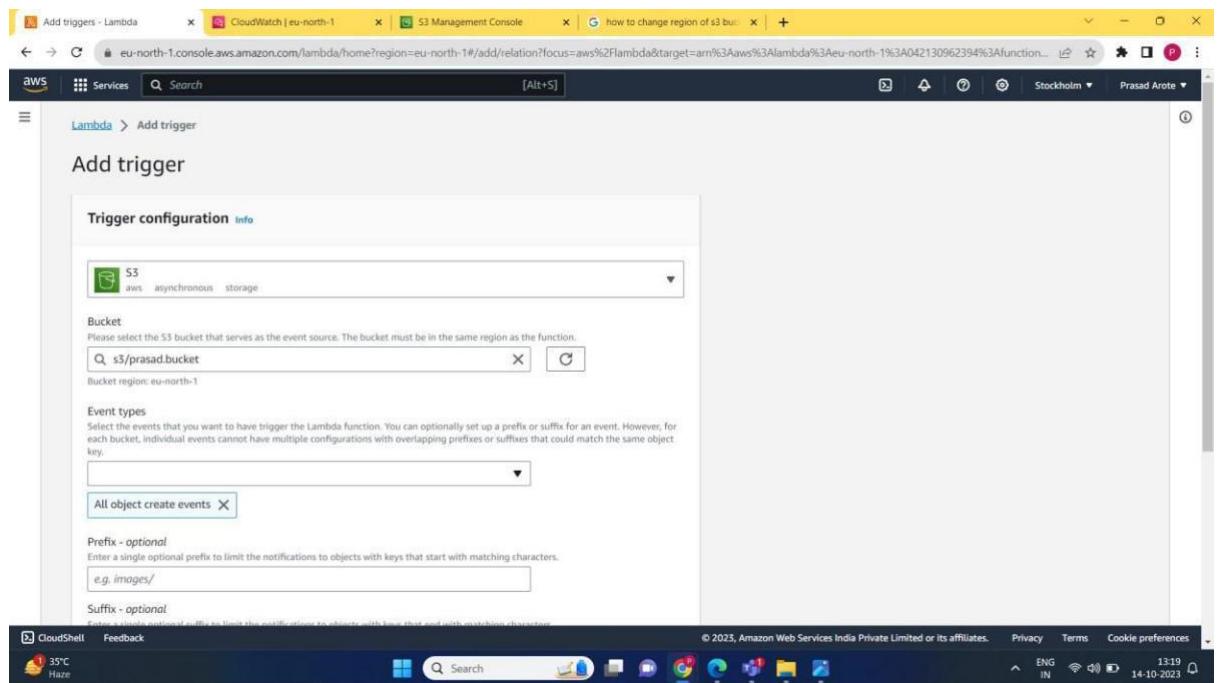
2. Create a Lambda function.

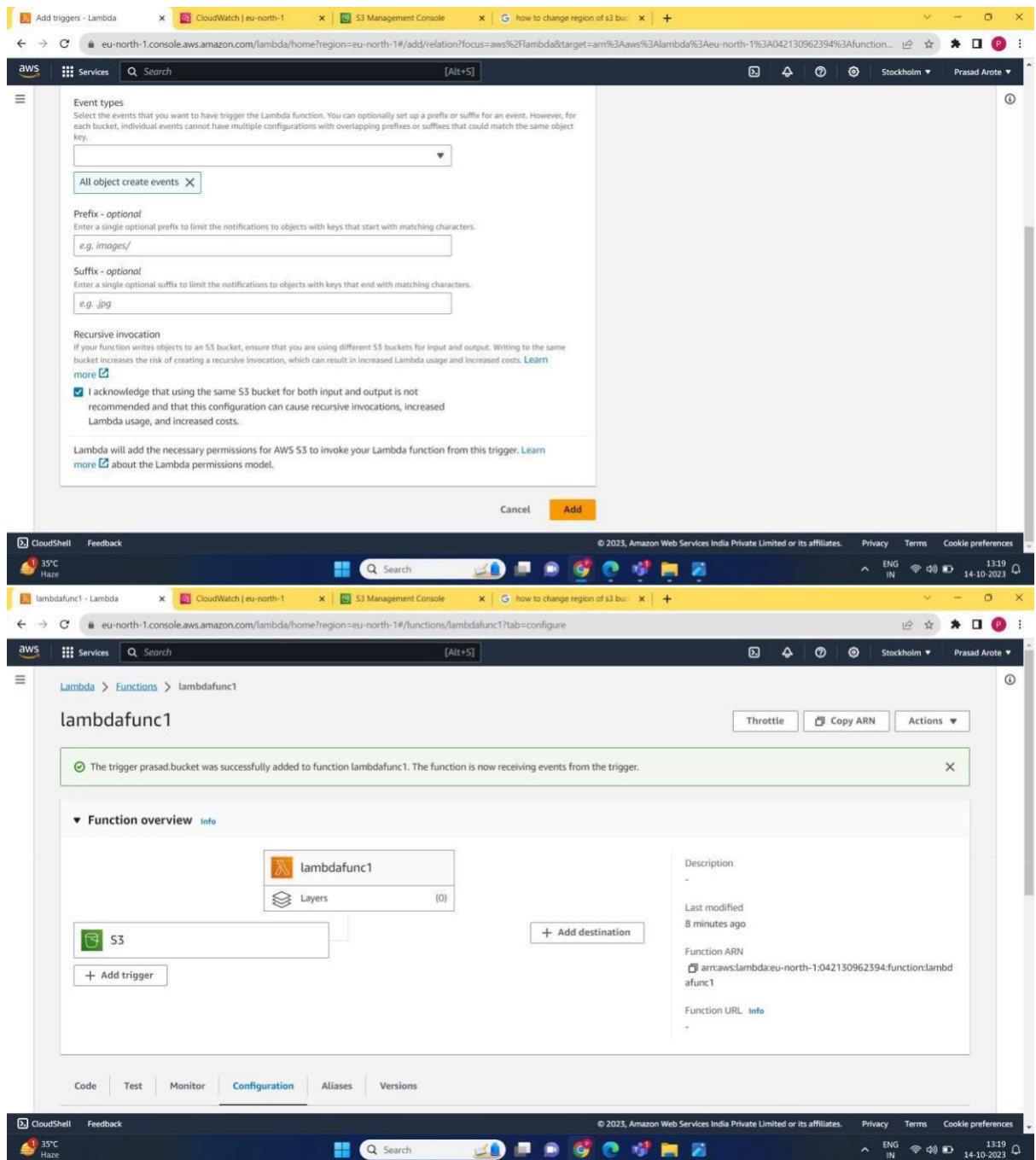
The screenshot shows the AWS S3 Management Console interface. A search bar at the top right contains the query 'lambda'. Below the search bar, the results are displayed under the heading 'Search results for "lambda"'. The results are categorized into 'Services' and 'Features'. Under 'Services', there are four items: Lambda, CodeBuild, AWS Signer, and Amazon Inspector. Under 'Features', there is one item: Local processing. On the right side of the search results, there is a sidebar with a green header that says 'Successfully created the bucket lambdafunc1'. This sidebar includes a 'View details' button, a 'Create bucket' button, and a table showing the bucket's creation date as 'October 14, 2023, 13:09:47 (UTC+05:30)'. At the bottom of the sidebar, there are buttons for 'Empty', 'Delete', and a refresh icon.

The screenshot shows the AWS Lambda console interface. The URL in the address bar is 'eu-north-1.console.aws.amazon.com/lambda/home?region=eu-north-1#functions/lambdafunc1?newFunction=true&tab=code'. The main area displays the function 'lambdafunc1'. It shows a code editor window with the title 'lambdafunc1 - Lambda'. Below the code editor, there is a 'Function overview' section. This section includes a thumbnail for the function, a 'Layers' section (which is currently empty), a 'Description' field (empty), a 'Last modified' timestamp ('3 seconds ago'), a 'Function ARN' field ('arn:aws:lambda:eu-north-1:042130962394:function:lambdafunc1'), and a 'Function URL' field. There are also buttons for 'Throttle', 'Copy ARN', and 'Actions'. At the bottom of the page, there are tabs for 'Code', 'Test', 'Monitor', 'Configuration', 'Aliases', and 'Versions'. The status bar at the bottom indicates 'CloudShell', 'Feedback', '39°C Hot weather', and the date '14-10-2023'.



3. Create a trigger





The screenshot shows the AWS S3 Management Console interface for uploading files to a bucket named 'prasad.bucket'. The 'Upload' screen is displayed, showing a central area for dragging and dropping files, a 'Files and folders' table, and a 'Destination' section.

Upload

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. [Learn more](#)

Drag and drop files and folders you want to upload here, or choose Add files or Add folder.

Files and folders (0)

All files and folders in this table will be uploaded.

Name	Folder	Type	Size
No files or folders			

You have not chosen any files or folders to upload.

Destination

Destination

s3://prasad.bucket

Destination details

Bucket settings that impact new objects stored in the specified destination.

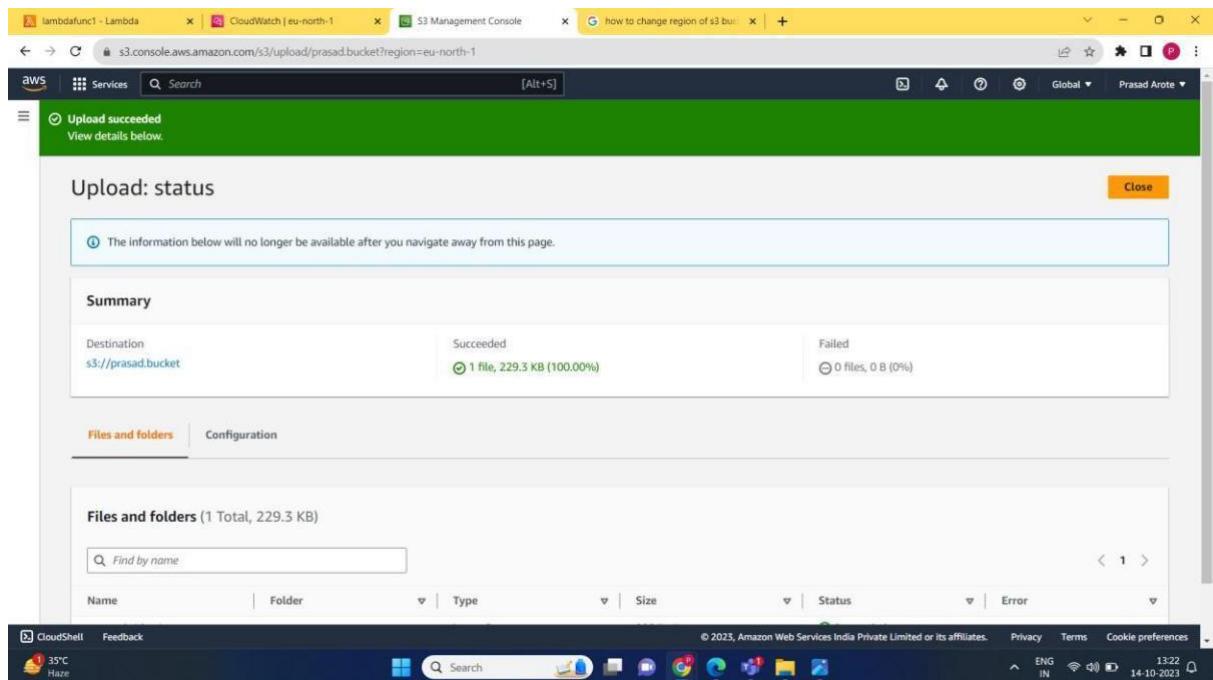
Permissions

Grant public access and access to other AWS accounts.

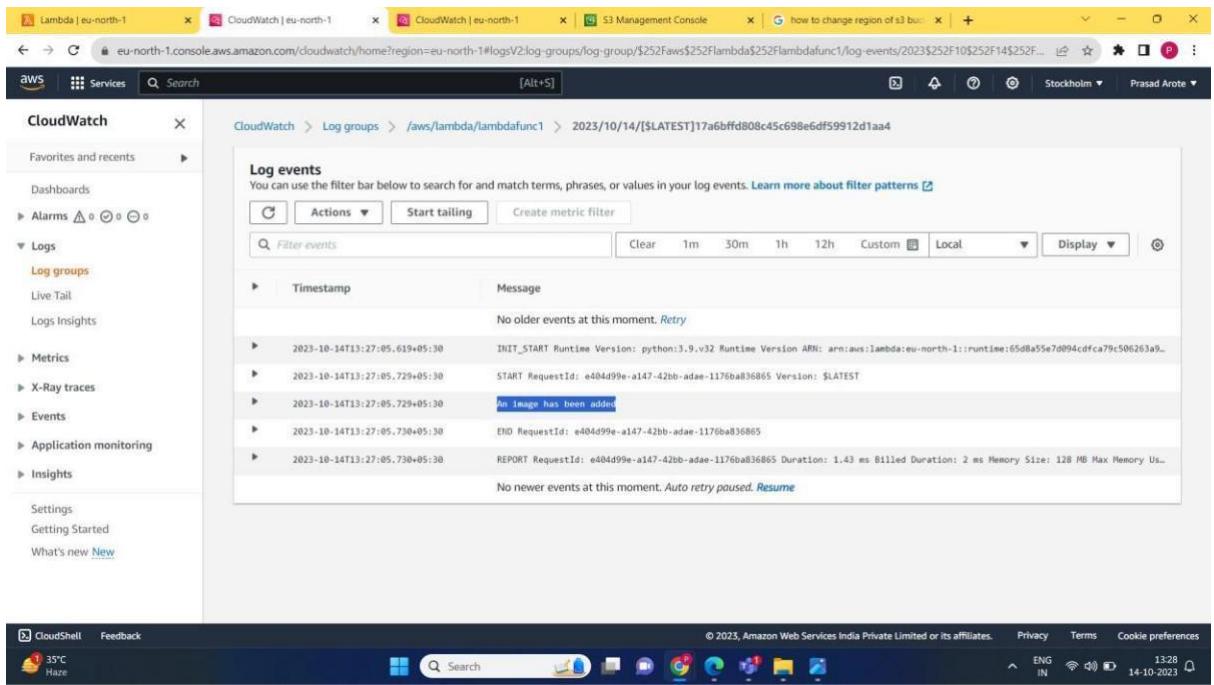
Properties

Specify storage class, encryption settings, tags, and more.

Cancel **Upload**



4. Thus we have triggered the function that logs when an image is added to S3 Bucket.



Conclusion:

We have successfully created an lambda functions that logs when an image is added in S3 bucket.

Assignment 10

Aim: To study AWS Lambda function and create the following Lambda functions using Python:

To add data to Dynamo DB database.

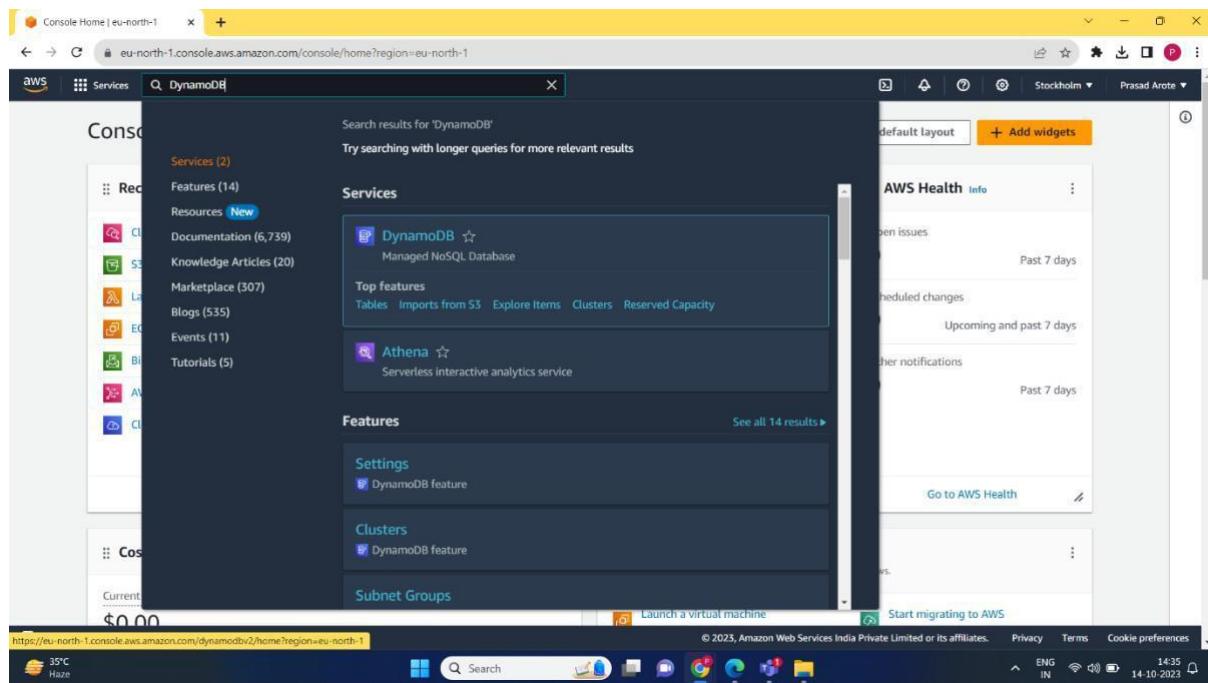
LO6: To engineer a composition of nano services using AWS Lambda and Step Functions with the serverless framework.

Theory:

DYNAMO DB

Amazon DynamoDB is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. DynamoDB lets you offload the administrative burdens of operating and scaling a distributed database so that you don't have to worry about hardware provisioning, setup and configuration, replication, software patching, or cluster scaling. DynamoDB also offers encryption at rest, which eliminates the operational burden and complexity involved in protecting sensitive data.

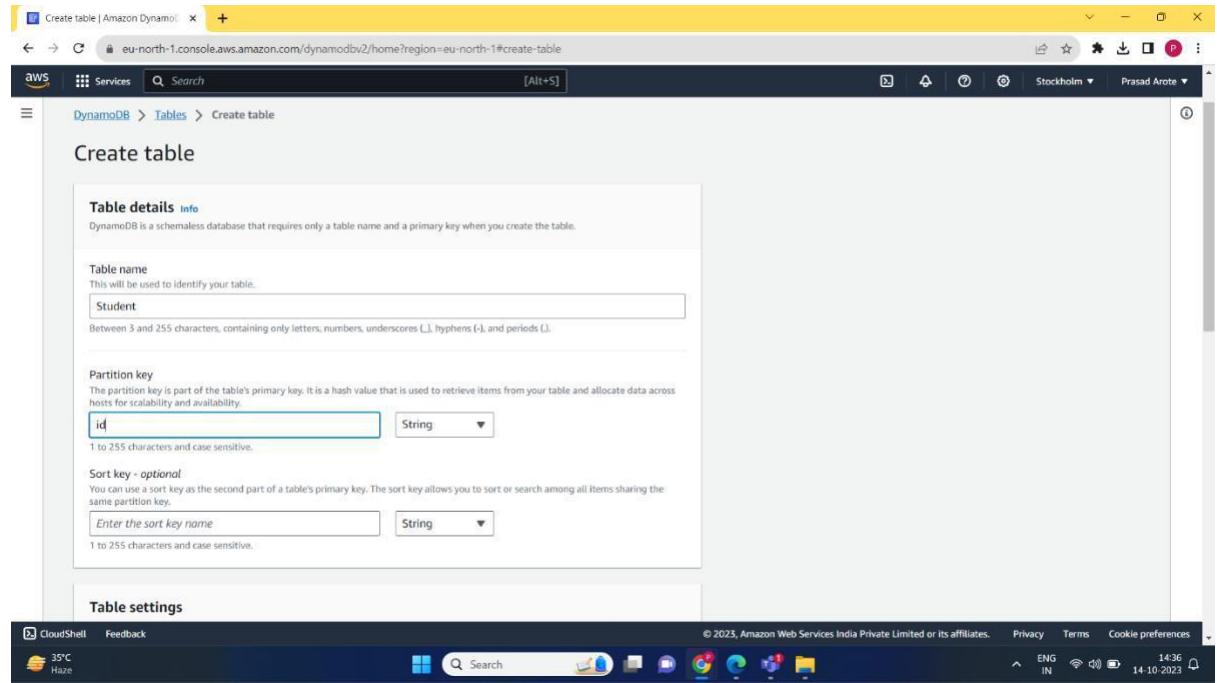
With DynamoDB, you can create database tables that can store and retrieve any amount of data and serve any level of request traffic. You can scale up or scale down your tables' throughput capacity without downtime or performance degradation. You can use the AWS Management Console to monitor resource utilization and performance metrics.



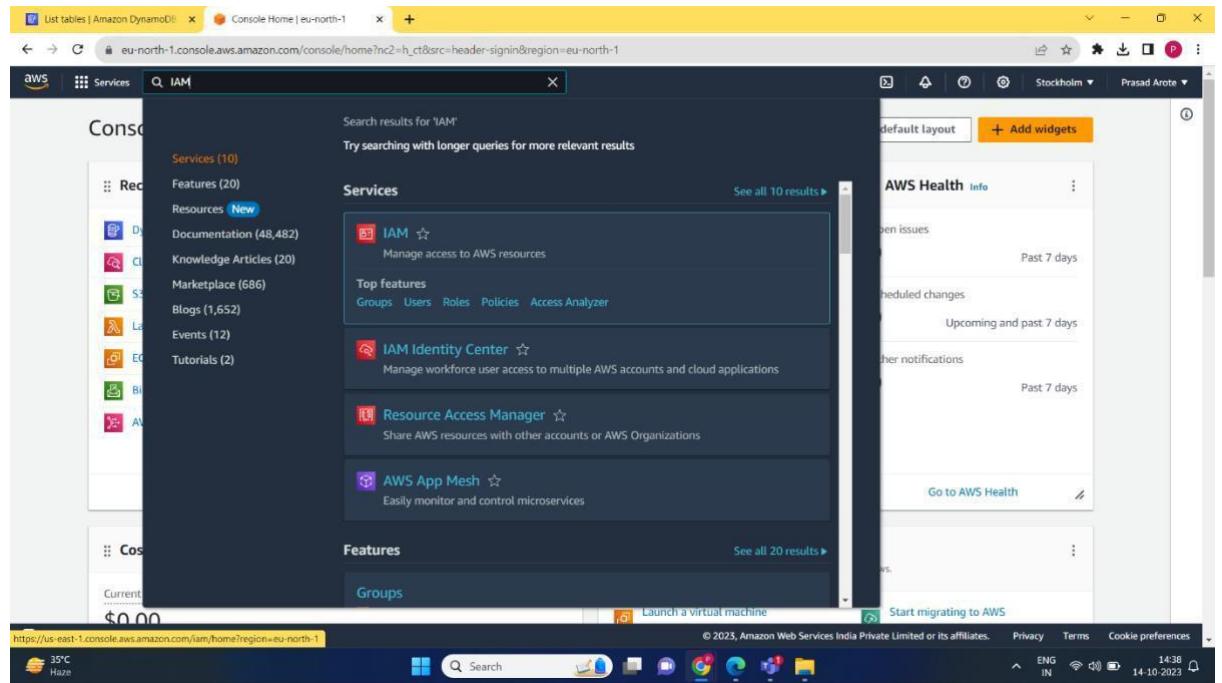
DynamoDB provides on-demand backup capability. It allows you to create full backups of your tables for long-term retention and archival for regulatory compliance needs.

STEPS:

1. Create a table



2. Create a role using IAM



3. Add permissions – AmazonDynamoFullAccess

Select trusted entity

Trusted entity type

- AWS service
- AWS account
- Web identity
- SAML 2.0 federation
- Custom trust policy

Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Service or use case

Lambda

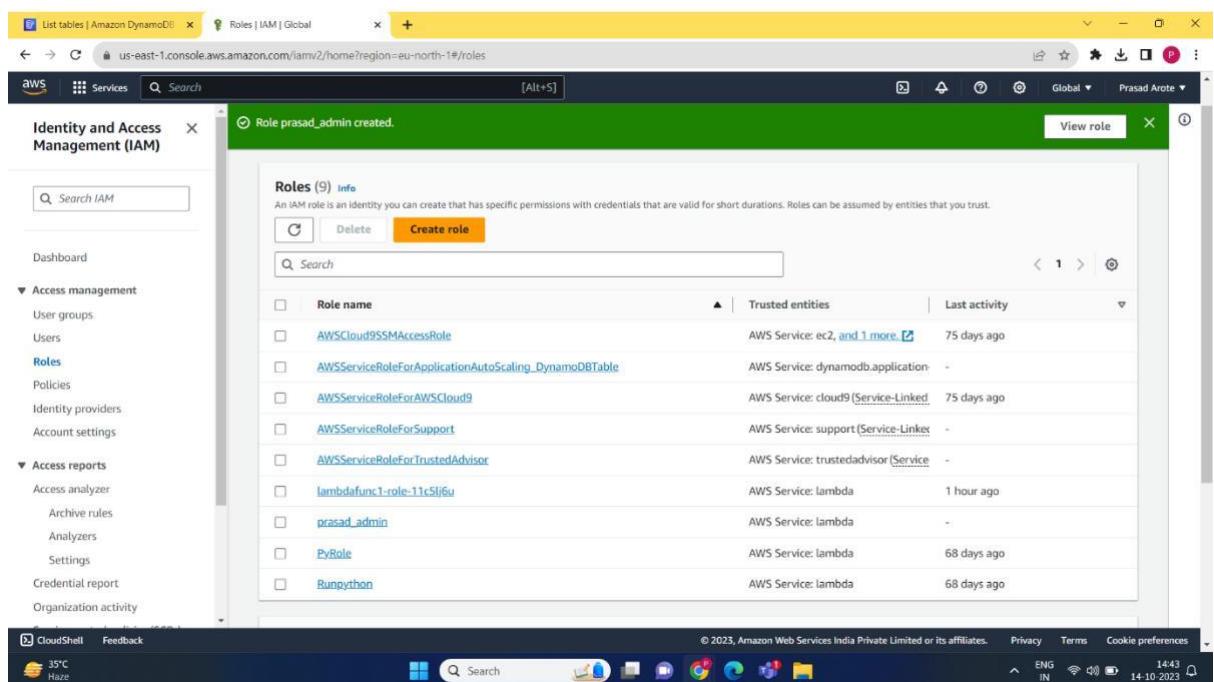
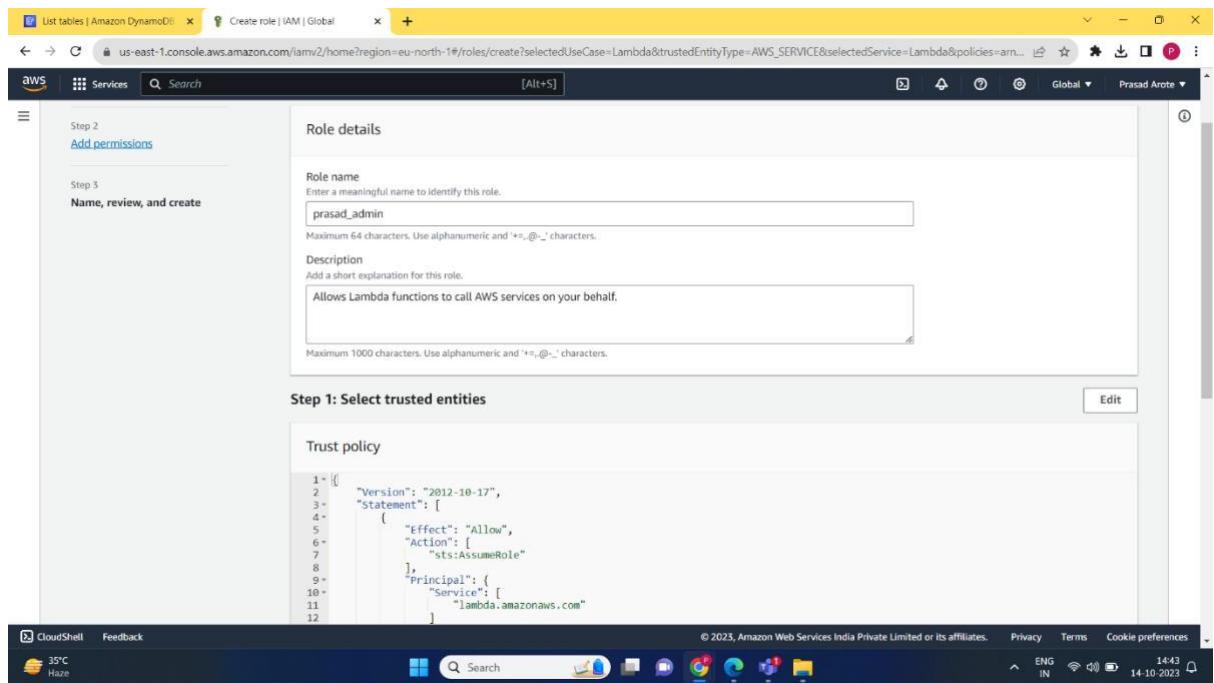
Add permissions

Permissions policies (1/887)

Policy name	Type	Description
<input checked="" type="checkbox"/> AmazonDynamoDBFullAccess	AWS managed	Provides full access to Amazon DynamoDB
<input type="checkbox"/> AmazonDynamoDBReadOnlyAccess	AWS managed	Provides read only access to Amazon Dyn...
<input type="checkbox"/> AWSLambdaDynamoDBExecutionRole	AWS managed	Provides list and read access to DynamoD...
<input type="checkbox"/> AWSLambdaInvocation-DynamoDB	AWS managed	Provides read access to DynamoDB Strea...

Set permissions boundary - optional

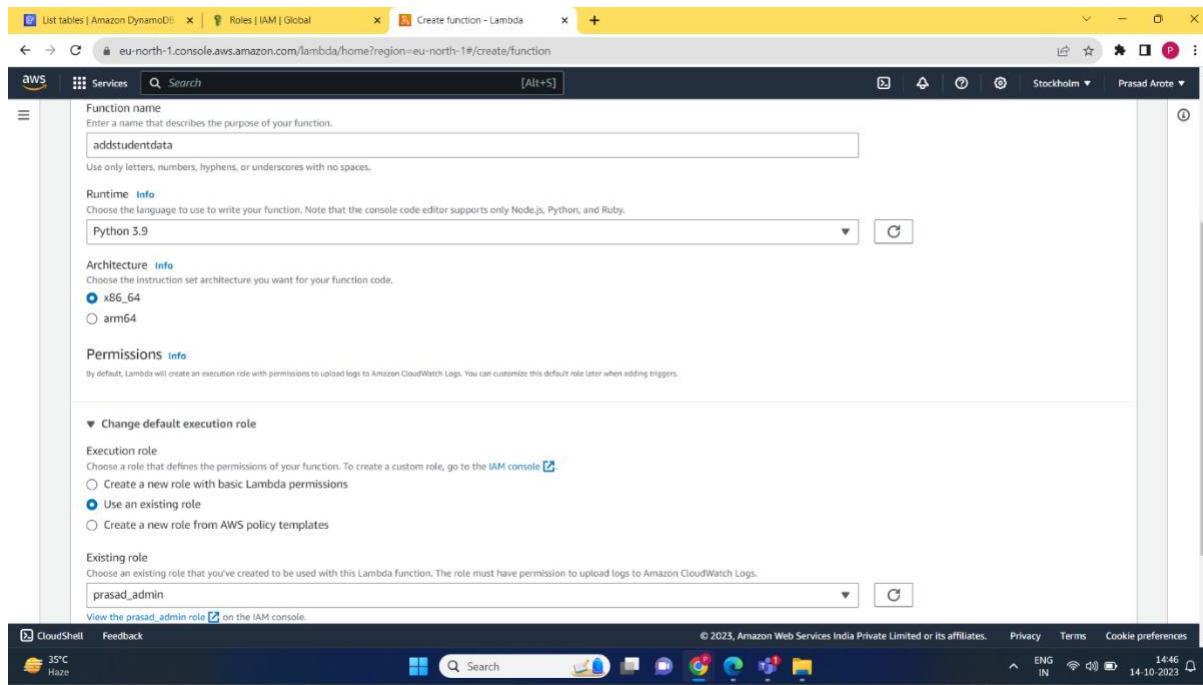
Cancel Previous Next



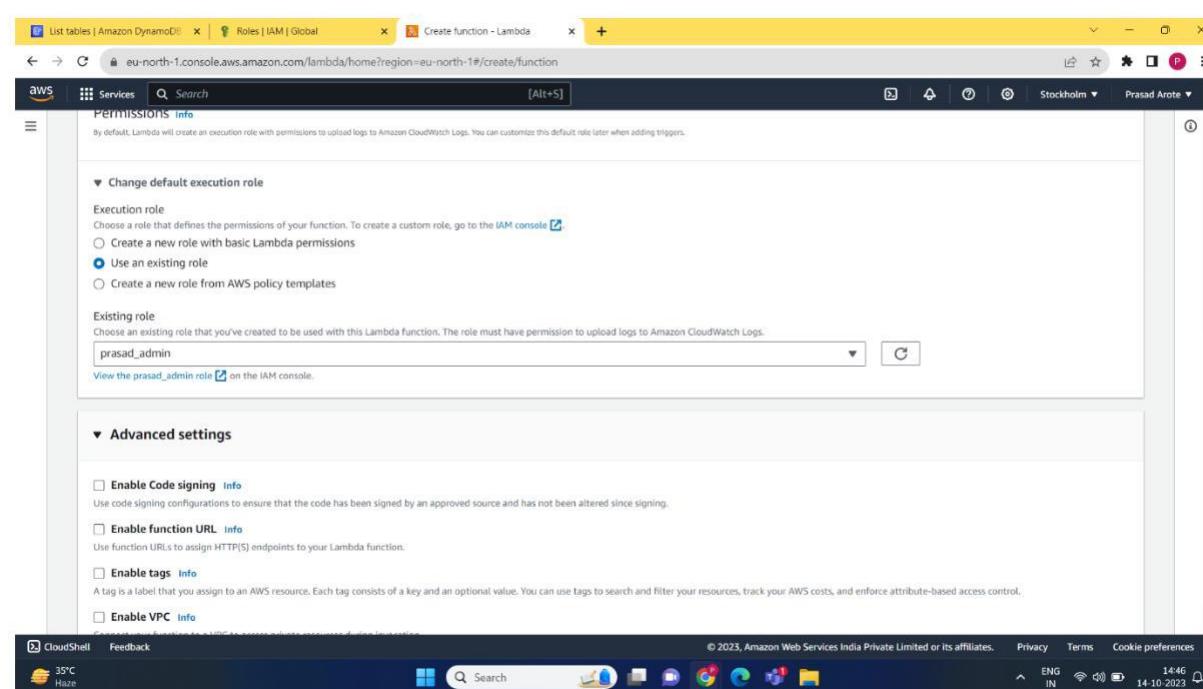
4. Create a Lambda Function

The screenshot shows the AWS Lambda search results page. The search term 'lambda' has returned 7 results. The top result is 'Lambda', described as 'Run code without thinking about servers'. Other results include 'CodeBuild', 'AWS Signer', and 'Amazon Inspector'. To the right of the search results, there is a sidebar titled 'AWS Health Info' which displays various monitoring and notification status for the last 7 days.

The screenshot shows the 'Create function' wizard. The user has chosen the 'Author from scratch' option. The 'Basic information' section requires the function name 'addstudentdata' and the runtime 'Python 3.9'. The 'Architecture' section shows 'x86_64' selected. The 'Permissions' section includes a note about Lambda's execution role. The bottom of the screen shows the AWS navigation bar and system status indicators.

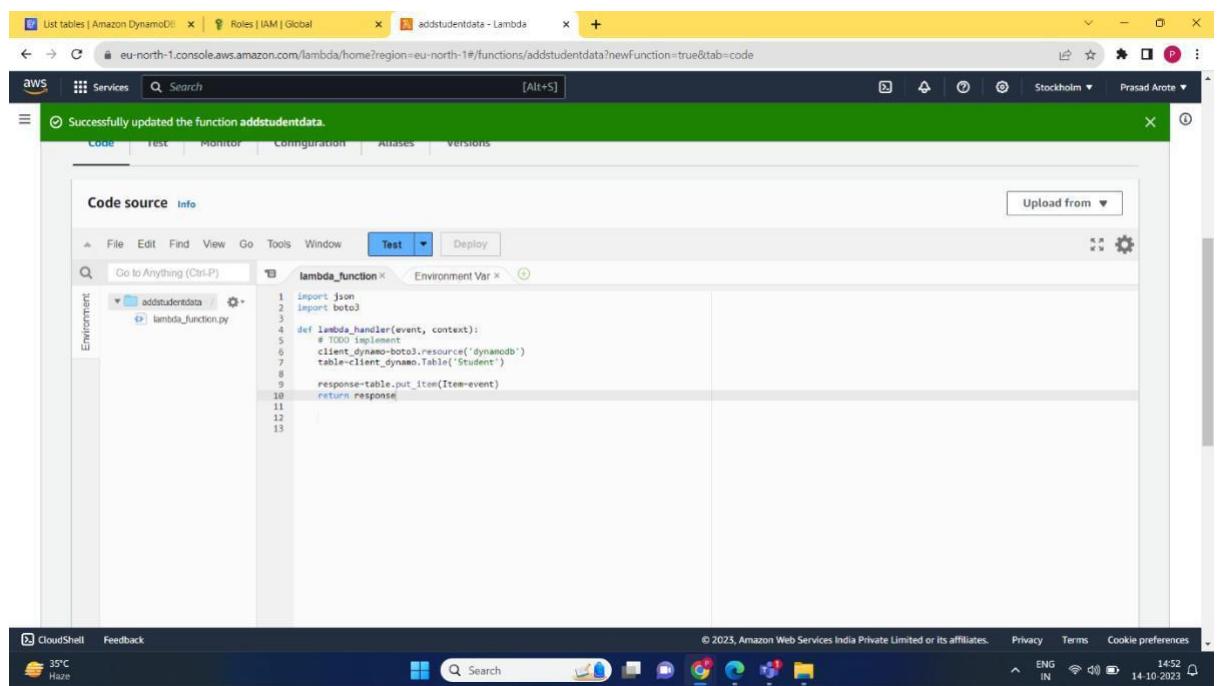


The screenshot shows the 'Create function - Lambda' wizard in the AWS Lambda console. The 'Function name' field contains 'addstudentdata'. The 'Runtime' dropdown is set to 'Python 3.9'. Under 'Architecture', 'x86_64' is selected. In the 'Permissions' section, 'Use an existing role' is chosen, and 'prasad_admin' is selected from the dropdown. The bottom part of the screenshot shows the 'Advanced settings' section, which is currently collapsed.

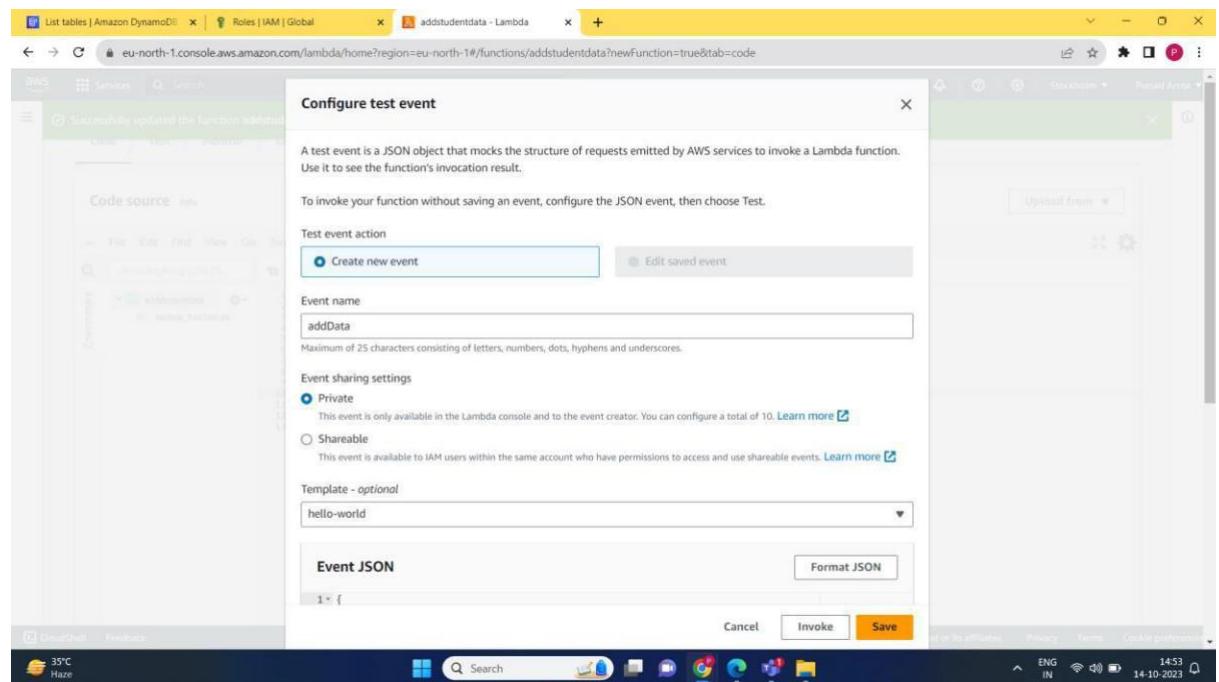


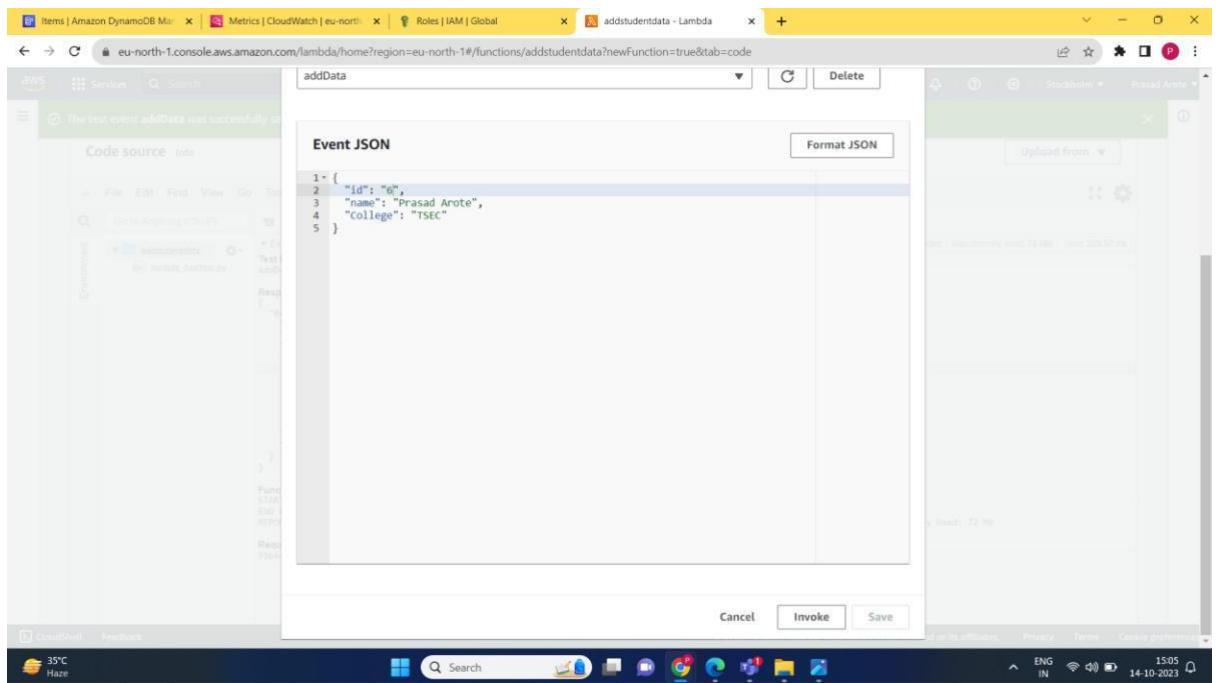
The screenshot shows the 'Advanced settings' section of the 'Create function - Lambda' wizard. It includes options for 'Enable Code signing', 'Enable function URL', 'Enable tags', and 'Enable VPC'. Each option has a corresponding 'Info' link and a description below it. The bottom part of the screenshot shows the AWS navigation bar and system status indicators.

5. Write the following code

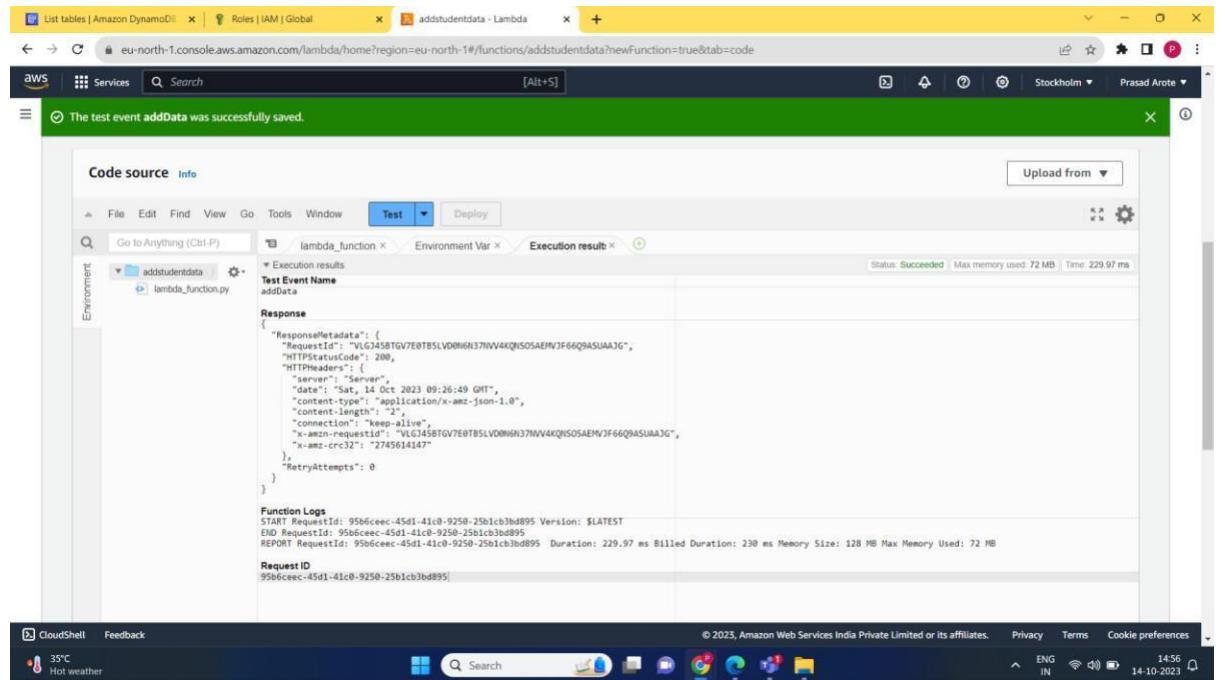


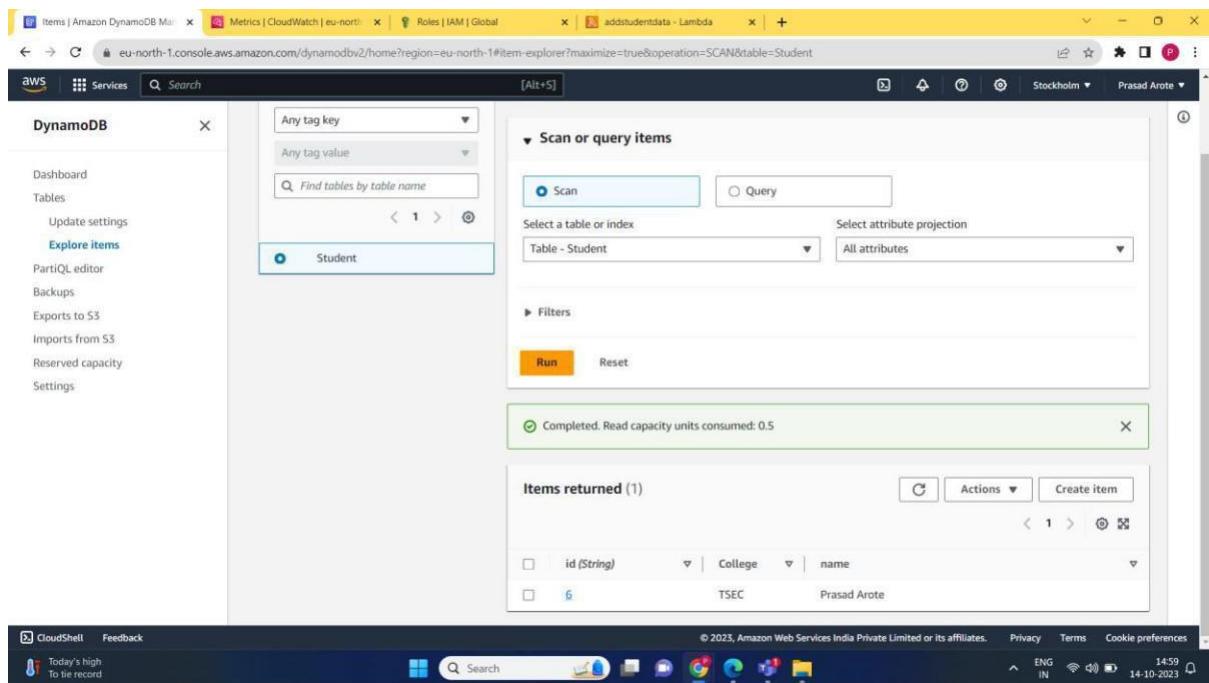
6. Configure test event and Save





- Run the test and afterwards go to the DynamoDB>Explore items> Student where you can see the record inserted using lambda function.





Conclusion:

Thus, we have successfully inserted data in DynamoDB by using a Lambda function.

Written Assignment 1

1. What security measures can be taken while using Kubernetes?

The following measures can be taken while using Kubernetes:

1. **Role-Based Access Control (RBAC):** RBAC restricts who can perform actions within a Kubernetes cluster. It defines roles and role bindings to specify what resources and operations users or service accounts can access. This prevents unauthorized access and actions within the cluster.
2. **Regular Updates:** Keeping Kubernetes and its components up to date is crucial. New releases often include security patches. Regular updates help mitigate known vulnerabilities and ensure your cluster remains secure.
3. **Network Policies:** Network policies allow you to define rules for communication between pods. By specifying which pods can communicate with each other, you can limit the attack surface and prevent unauthorized access.

- 4. Container Security Tools:** Employ container security tools like vulnerability scanners to assess the security of container images. These tools can identify and remediate vulnerabilities in the containerized applications before they are deployed.
- 5. Monitoring and Audit:** Implement monitoring and auditing solutions to track cluster activity. This helps detect and respond to suspicious or unauthorized behavior. Tools like Prometheus and Grafana can be used for monitoring, while audit logs provide insights into cluster activity.
- 6. Secrets Management:** Sensitive data like API keys, passwords, and certificates should be stored securely using Kubernetes secrets or external vaults. This prevents sensitive information from being exposed within containers or configuration files.
- 7. PodSecurityPolicies (PSP):** PSP is a Kubernetes feature that enforces security policies at the pod level. It allows you to define restrictions on privilege escalation, host access, and other security-sensitive configurations for pods.
- 8. Namespaces:** Use Kubernetes namespaces to logically isolate workloads. This provides a level of separation between different applications or teams, reducing the risk of unauthorized access or interference between them.
- 9. Admission Controllers:** Admission controllers are webhook plugins that intercept and validate requests to the Kubernetes API server. You can use them to enforce custom policies and ensure that only compliant resources are admitted to the cluster.
- 10. Container Runtime Security:** Implement container runtime security solutions like Docker Security Scanning or container runtime protection tools. These tools monitor containers at runtime for abnormal behavior, helping to detect and respond to potential threats.

2. What are the three security techniques that can be used to protect data?

Three security techniques commonly used to protect data are:

- 1. Encryption:** Encryption is the process of converting data into a secure format that can only be read by someone with the decryption key. It ensures that even if

unauthorized parties access the data, they cannot understand it without the correct key. Two common types of encryption are:

- a. Data-at-rest Encryption: Protects data when it's stored on disk or in a database.
- b. Data-in-transit Encryption: Secures data as it's transmitted between systems over networks.

1. Access Control: Access control mechanisms regulate who can access data and what actions they can perform on it. This involves setting permissions, roles, and policies to ensure that only authorized users or applications can access and manipulate data. Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC) are commonly used access control models.

2. Data Masking/Redaction: Data masking or redaction involves obscuring or replacing sensitive data with fictitious or scrambled values. This is often used in non-production environments or when sharing data with third parties. It ensures that even if someone gains access to the data, they cannot see the actual sensitive information.

3. How do you expose a service using ingress in Kubernetes?

In Kubernetes, you can expose a service using an Ingress resource. An Ingress resource is an API object that manages external access to services within your cluster. It allows you to define how HTTP and HTTPS traffic should be routed to your services based on rules and host/path-based routing.

Here's a step-by-step guide on how to expose a service using Ingress in Kubernetes:

1. Prerequisites:

- Ensure you have a running Kubernetes cluster.
- You need to have **kubectl** configured to connect to your cluster.

2. Create a Deployment and a Service:

Before exposing a service, you must have a Deployment and a Service to expose. For example:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  selector:
    matchLabels:
```

```
    app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-app-image:latest
```

```
---
```

```
apiVersion: v1
kind: Service
metadata:
  name: my-app-service
spec:
  selector:
    app: my-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

3. **Create an Ingress resource:** Create an Ingress resource that defines the rules for routing traffic to your service. Here's an example Ingress resource:

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-app-ingress
spec:
  rules:
    - host: myapp.example.com
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: my-app-service
                port:
                  number: 80
```

4. **Deploy the Ingress Resource:** Apply the Ingress resource configuration using `kubectl apply -f your-ingress.yaml`.
5. **Configure DNS:** For the host specified in your Ingress resource (in this case, `myapp.example.com`), you need to configure DNS to point to the IP address of your cluster. This step is required to route external traffic to your Ingress controller.
6. **Verify:** After the Ingress resource is created, you can verify that your service is exposed correctly by accessing `myapp.example.com` in a web browser or using tools like `curl`.

4. Which service protocols does Kubernetes ingress expose?

Kubernetes Ingress exposes services using HTTP and HTTPS protocols. An Ingress resource is primarily designed for routing and managing HTTP and HTTPS traffic to services within your Kubernetes cluster.

Here's a brief overview of the protocols Ingress can expose:

1. **HTTP:** Ingress allows you to expose services that use the HTTP protocol. You can define rules that specify how incoming HTTP requests should be routed to different services based on the host, path, or other criteria. This is the most common use case for Ingress.
2. **HTTPS:** In addition to HTTP, Ingress also supports HTTPS. You can configure TLS (Transport Layer Security) termination at the Ingress controller level to secure the traffic between the client and the Ingress controller. This is essential for securing web applications and services with encryption.
3. **HTTP/HTTPS Redirection:** Ingress resources can also be used to set up HTTP-to-HTTPS redirection or vice versa. This is often done to ensure that all traffic is encrypted over HTTPS for security purposes.
4. **TCP and UDP (In Some Cases):** While Ingress is primarily designed for HTTP and HTTPS, some Ingress controllers, like the Nginx Ingress controller, support TCP and UDP traffic routing as well. This enables you to route non-HTTP and non-HTTPS traffic to specific services based on port numbers.

The specific protocols and features supported by Ingress may vary depending on the Ingress controller you use. The choice of Ingress controller will determine the full range of features and capabilities available for routing and managing traffic in your Kubernetes cluster. Popular Ingress controllers like Nginx Ingress, Traefik, and HAProxy Ingress offer various advanced features and options for customizing traffic routing and handling.

Written Assignment 2

Q1] How to deploy a Lambda function on AWS?

Deploying a Lambda function on AWS involves several steps. Lambda is a serverless computing service that allows you to run code in response to events without having to manage servers. Here's a high-level overview of how to deploy a Lambda function:

1. Sign In to AWS Console:

- Go to the AWS Management Console (<https://aws.amazon.com/>).
- Sign in to your AWS account or create one if you don't have an account already.

2. Open Lambda Service:

- Once you're logged in, open the AWS Lambda service. You can find it in the "Compute" section or by searching for "Lambda" in the AWS services search bar.

3. Create a Lambda Function:

- Click the "Create function" button to start creating a new Lambda function.

4. Author from Scratch or Use Blueprint:

- You can choose to create the function from scratch or use a blueprint (predefined function templates) depending on your requirements.

5. Configure Function:

- You will need to configure your function with a name, runtime (e.g., Node.js, Python, Java), and execution role. The execution role defines what AWS resources your Lambda function can access.

6. Create Function:

- Click the "Create function" button to create the basic Lambda function.

7. Add Code:

- In the "Function code" section, you can either upload your code as a .zip file or edit it in the inline code editor. You can also choose to link your code from an Amazon S3 bucket.

8. Configure Trigger:

- Lambda functions are typically triggered by events. You can configure triggers, such as API Gateway, S3, CloudWatch Events, etc., depending on what should invoke your Lambda function.

9. Set Environment Variables (Optional):

- You can set environment variables for your Lambda function if it requires configuration options or secrets.

10. Configure Advanced Settings (Optional):

- You can configure other settings, including memory, timeout, VPC (Virtual Private Cloud) settings, and more in the "Advanced settings" section.

11. Review and Create Function:

- Review your function configuration to ensure it's correct, and then click the "Create function" button to create the Lambda function.

12. Test Your Function (Optional):

- You can test your function from the AWS Lambda console by configuring a test event. This helps ensure your function works as expected.

13. Deploy Your Function:

- Once you're satisfied with your function, it's effectively deployed and ready to run.

Q2] What are the deployment options for AWS Lambda?

AWS Lambda offers several deployment options, allowing you to choose the most suitable method for deploying your serverless functions. Here are the main deployment options for AWS Lambda:

1. AWS Management Console:

- You can create and deploy Lambda functions directly through the AWS Management Console. This is the most straightforward option for small-scale deployments or when you want to quickly prototype a function.

2. AWS Command Line Interface (CLI):

- The AWS CLI allows you to create and deploy Lambda functions from your local development environment or via scripting. This method is useful for automation, continuous integration, and when working with infrastructure-as-code (IaC) tools.

3. AWS Serverless Application Model (SAM):

- AWS SAM is an open-source framework for building serverless applications. It extends AWS CloudFormation to provide a simplified way to define the Amazon API Gateway APIs, AWS Lambda functions, and Amazon DynamoDB tables needed by your serverless application. It enables you to define serverless resources using a simple YAML or JSON template and deploy them using the AWS CLI.

4. AWS CloudFormation:

- AWS CloudFormation, a service for provisioning and managing AWS infrastructure using templates, allows you to define and deploy Lambda functions as part of a larger stack. This method is suitable for complex applications or environments that require a complete infrastructure-as-code approach.

5. AWS Serverless Application Repository:

- The AWS Serverless Application Repository is a service that enables sharing and deploying serverless applications built by the community.

You can discover, deploy, and manage pre-built serverless applications directly from the repository.

6. Integrated Development Environments (IDEs):

- Some integrated development environments, such as AWS Cloud9, provide Lambda deployment features, streamlining the development and deployment process.

7. Third-Party Tools and Frameworks:

- Various third-party tools and frameworks, like the Serverless Framework and the Claudia.js framework, offer more advanced deployment options and features for managing Lambda functions.

8. GitHub Actions and Other CI/CD Tools:

- You can integrate your Lambda deployment process with continuous integration and continuous deployment (CI/CD) tools like GitHub Actions, Jenkins, Travis CI, CircleCI, etc. This allows you to automate the deployment of your Lambda functions whenever changes are made to your code repository.

9. Custom Deployment Scripts and Pipelines:

- Organizations may develop custom deployment scripts and pipelines tailored to their specific requirements. These scripts can be used for more advanced deployment strategies, such as blue-green deployments or canary releases.

Q3] What are the 3 full deployment modes that can be used for AWS?

AWS offers three primary deployment modes for your applications:

1. EC2 Deployment:

- In this deployment mode, you create and manage virtual machines (EC2 instances) on AWS. You have full control over the operating system, the application stack, and all aspects of the environment.
- This mode is typically used for traditional applications that require manual management, and it's well-suited for workloads that can't be easily refactored for serverless or containerized deployment.

2. Lambda Deployment:

- AWS Lambda is the core service for serverless computing. In this mode, you deploy your application code as small, single-purpose functions that automatically scale in response to events. You don't need to manage servers or infrastructure; AWS handles scaling, patching, and maintenance.
- Serverless deployment is ideal for event-driven, stateless, and short-lived functions, such as microservices, data processing, and automation tasks.

3. ECS/EKS (Elastic Container Service / Elastic Kubernetes Service) Deployment:

- AWS ECS and EKS provide container orchestration platforms that allow you to deploy, manage, and scale containerized applications.

ECS is for running Docker containers on a managed infrastructure, while EKS is for running Kubernetes clusters.

- Container deployments are well-suited for applications that are built as container images and can be managed as microservices. They provide a balance between control and scalability, making them ideal for a wide range of workloads.

Q4] What are the 3 components of AWS Lambda?

AWS Lambda has three key components that work together to enable serverless computing and event-driven execution of code:

1. Function Code:

- The function code is the heart of a Lambda function. It contains the actual code that you want to run in response to events. Lambda supports code written in various programming languages, including Python, Node.js, Java, C#, Ruby, and more.
- You can write your code directly in the Lambda Management Console using the inline code editor, or you can package your code into a deployment package and upload it to Lambda. The code must be stateless and designed to handle a single event at a time.

2. Event Source:

- The event source is what triggers the Lambda function to execute. AWS Lambda is inherently event-driven, meaning it responds to events generated by AWS services or custom events from external sources.
- Event sources can include services like Amazon S3, Amazon DynamoDB, Amazon Kinesis, Amazon Simple Queue Service (SQS), and AWS CloudWatch Events, among others. Custom events can also be configured using Amazon API Gateway or AWS Step Functions, allowing you to connect Lambda to a wide range of AWS and non-AWS services.

3. Execution Environment:

- The execution environment is the infrastructure and resources provided by AWS to run your Lambda function. When an event occurs, AWS Lambda automatically provisions an execution environment, executes the function code, and then deallocates the environment.
- AWS Lambda manages the underlying infrastructure, including scaling, patching, and maintenance. You do not need to worry about server provisioning or resource management.
- You can configure the amount of memory allocated to your Lambda function, which in turn determines the CPU power, network bandwidth, and other resources available during execution. AWS Lambda charges based on the actual amount of memory and the execution time.
-

These three components work together to create a fully managed, event-driven compute service. When an event occurs in the event source, AWS Lambda creates

an execution environment, runs the function code, and provides the results. Lambda is designed to be highly scalable and can handle a wide range of use cases, from simple data processing tasks to complex microservices in a serverless architecture.