Thadomal Shahani Engineering College

Bandra (W.), Mumbai- 400 050.

CERTIFICATE

Certify that Mr./Miss Atharva Pandit

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

Agarwal

Teacher In-Charge

Head of the Department

Date 21/10/23.

Principal

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

CONTENTS

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.

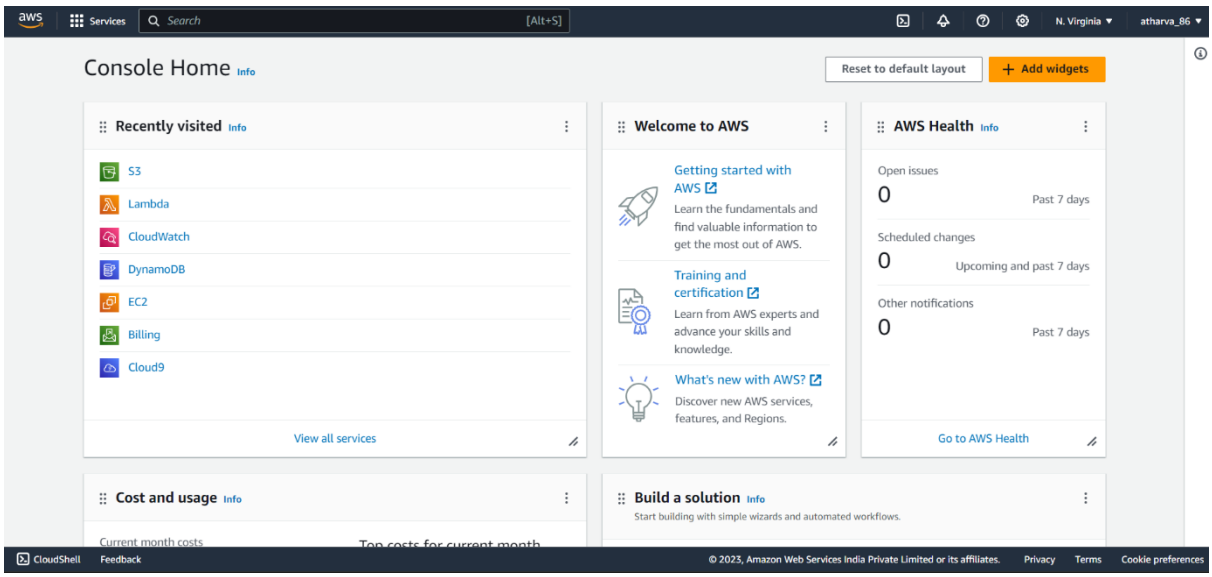
16. Connect to Your EC2 Instance:

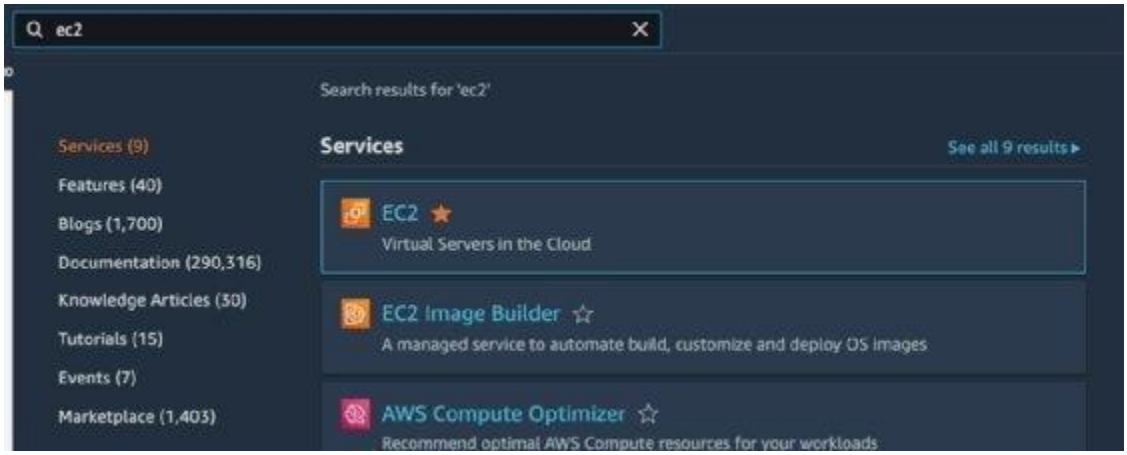
You can connect to your EC2 instance using SSH (for Linux) or Remote Desktop (for Windows). Use the key pair you created to authenticate.

17. Configure Your Instance:

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

Snapshots:





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

Assignment 2
CLOUD9

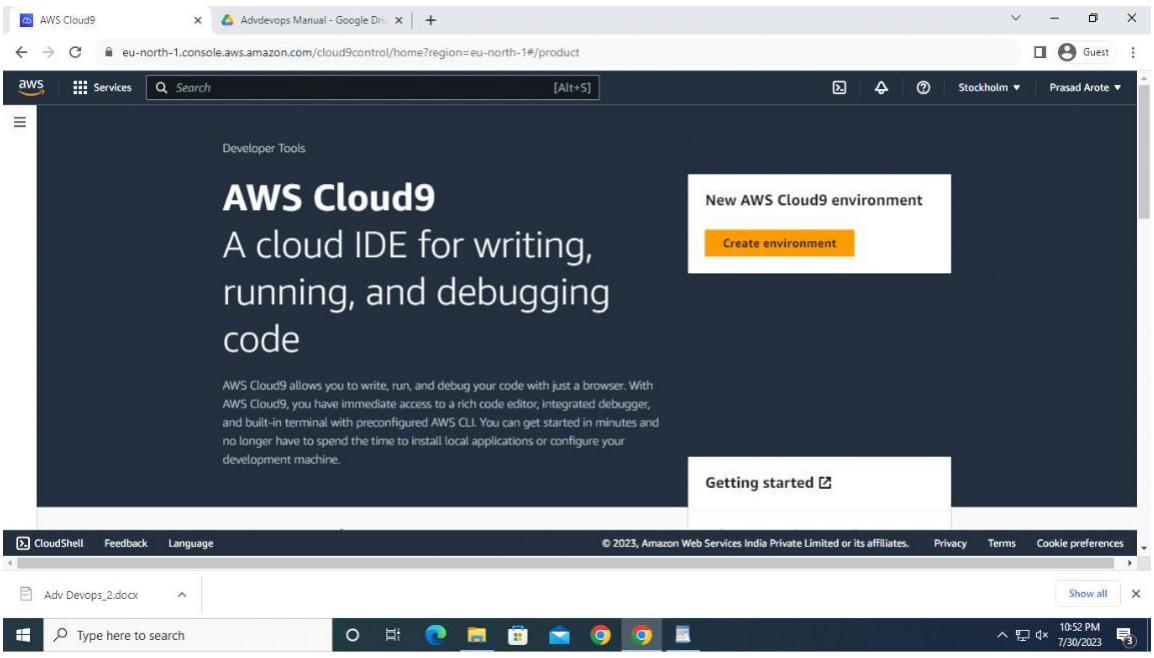
Aim: To create a Cloud9 Environment.

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-

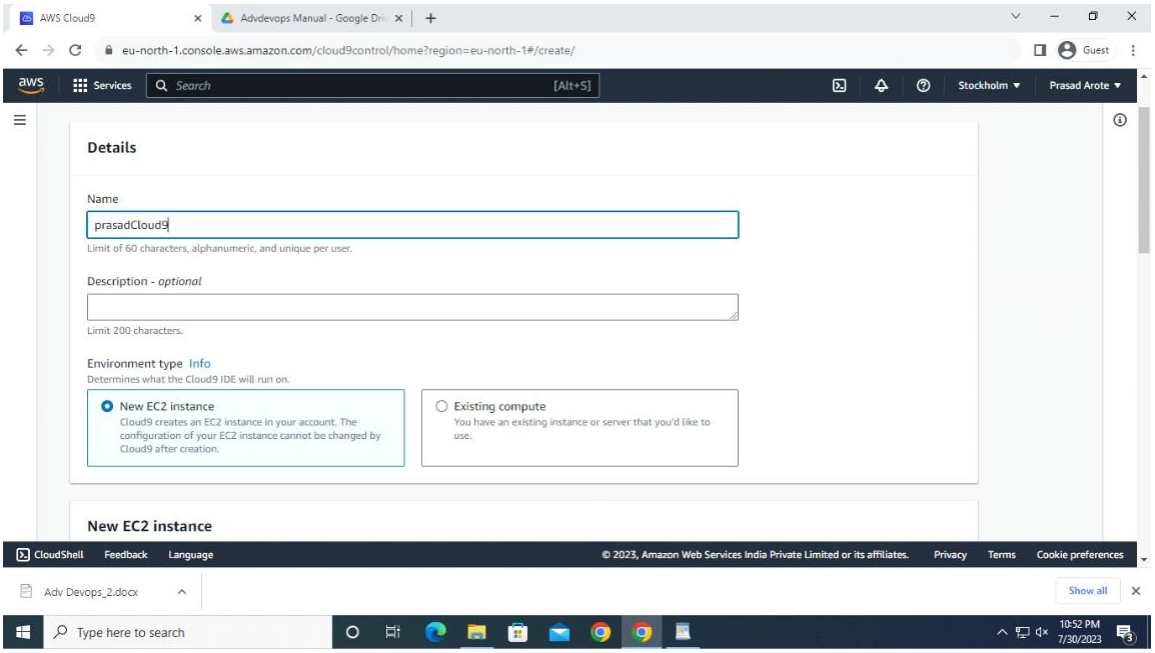
LOG IN TO YOUR AWS ACCOUNT,

SEARCH FOR CLOUD 9 IN THE SEARCH BAR

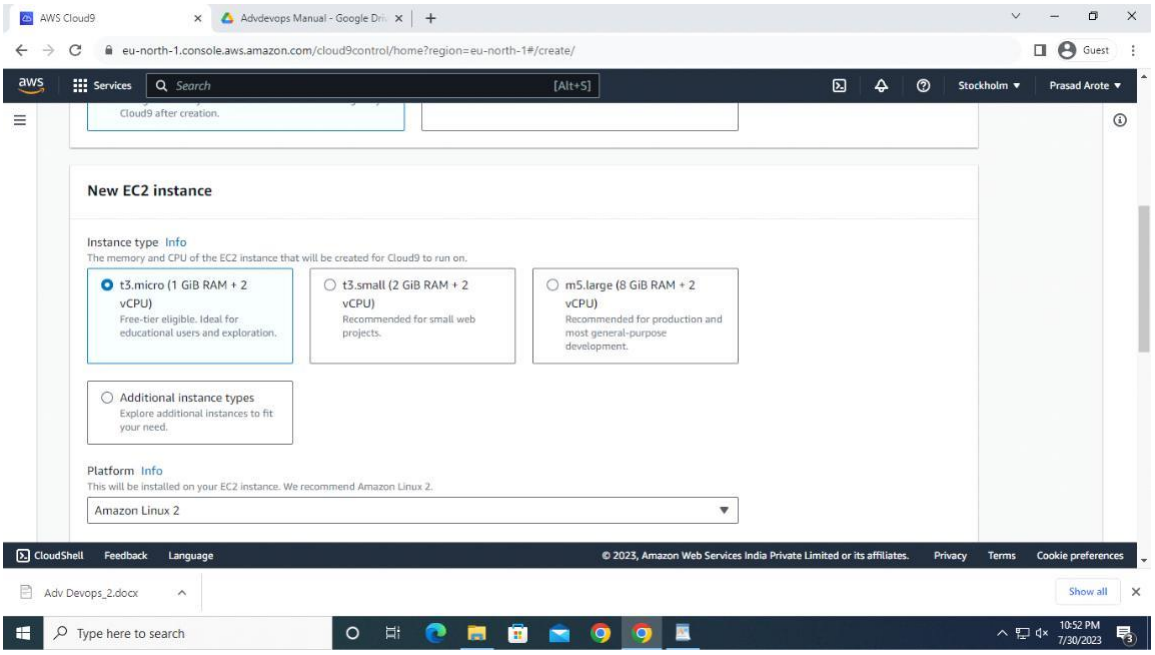


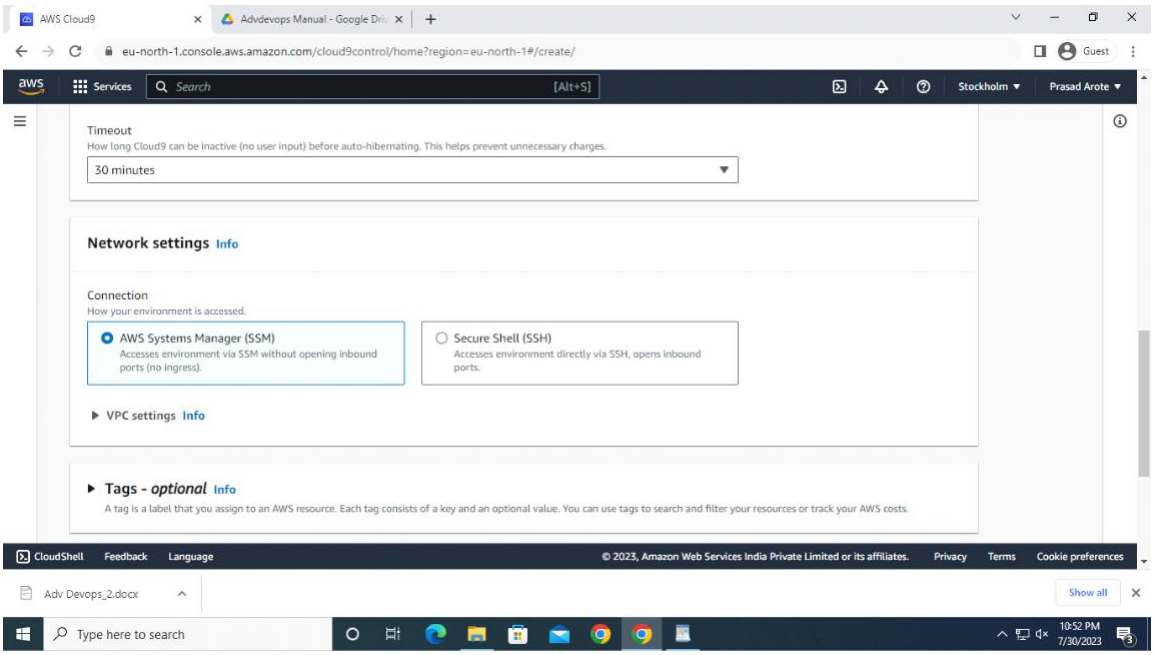
CLICK ON CREATE ENVIRONMNET,

NAME THE ENVIRONMNET

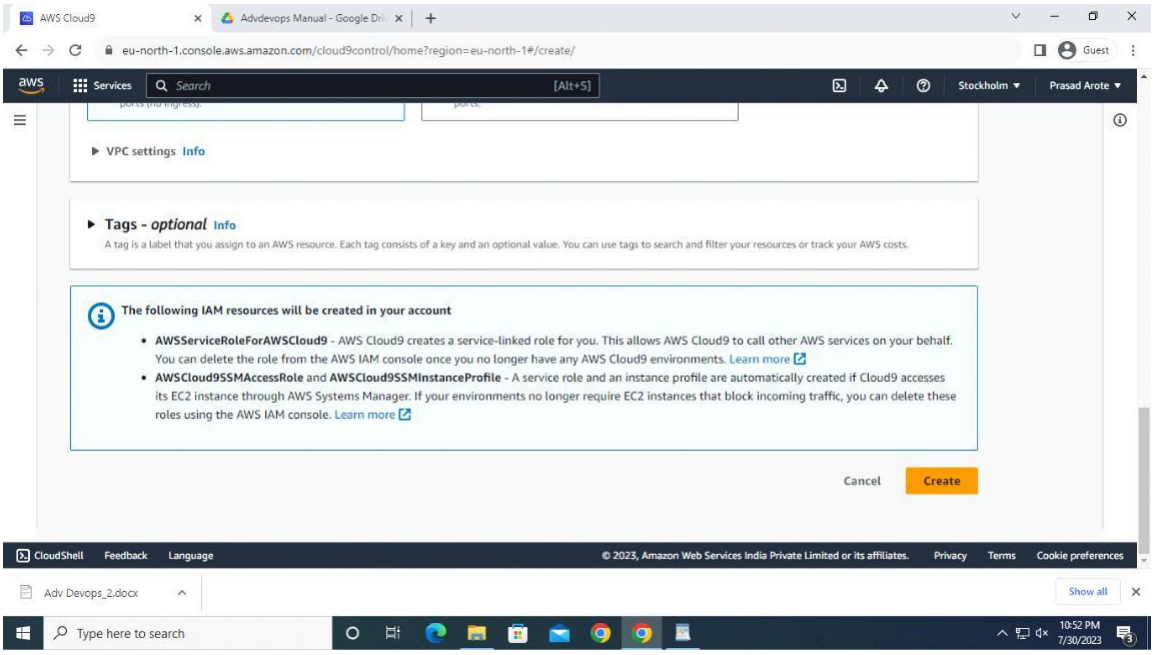


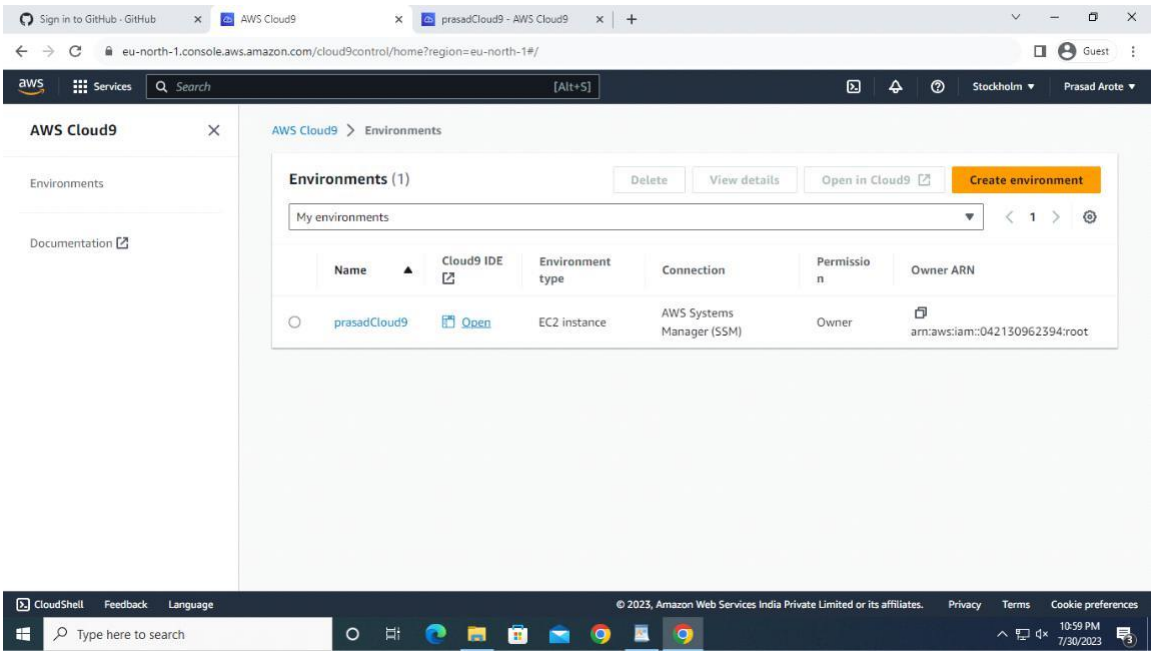
Now click on next step.



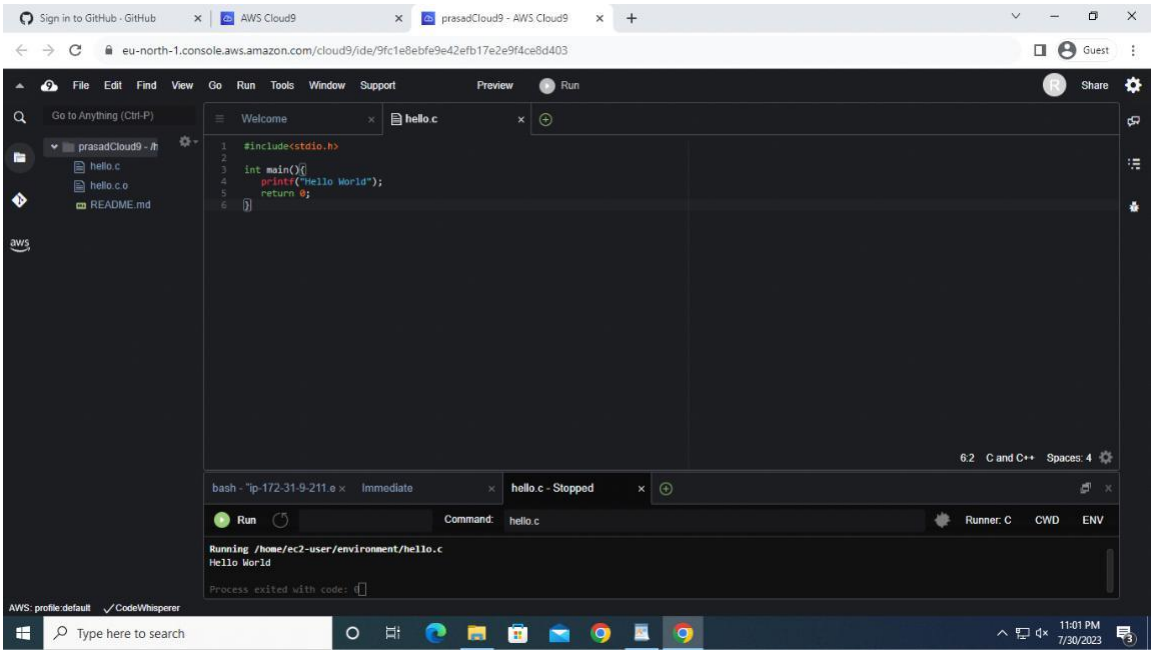


Again, click on Next Step,
Now click on Create Environment.





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



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

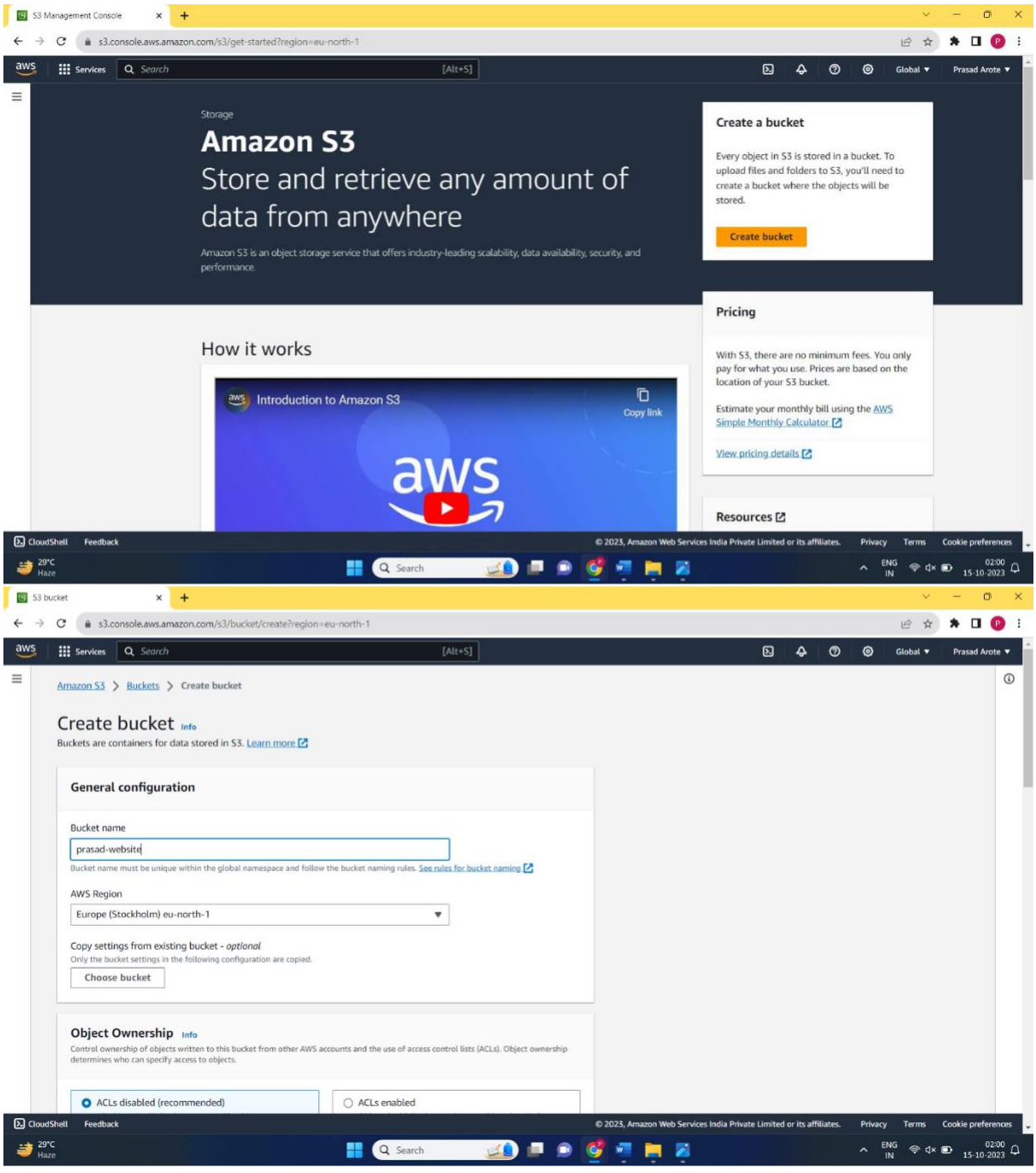
Assignment 3

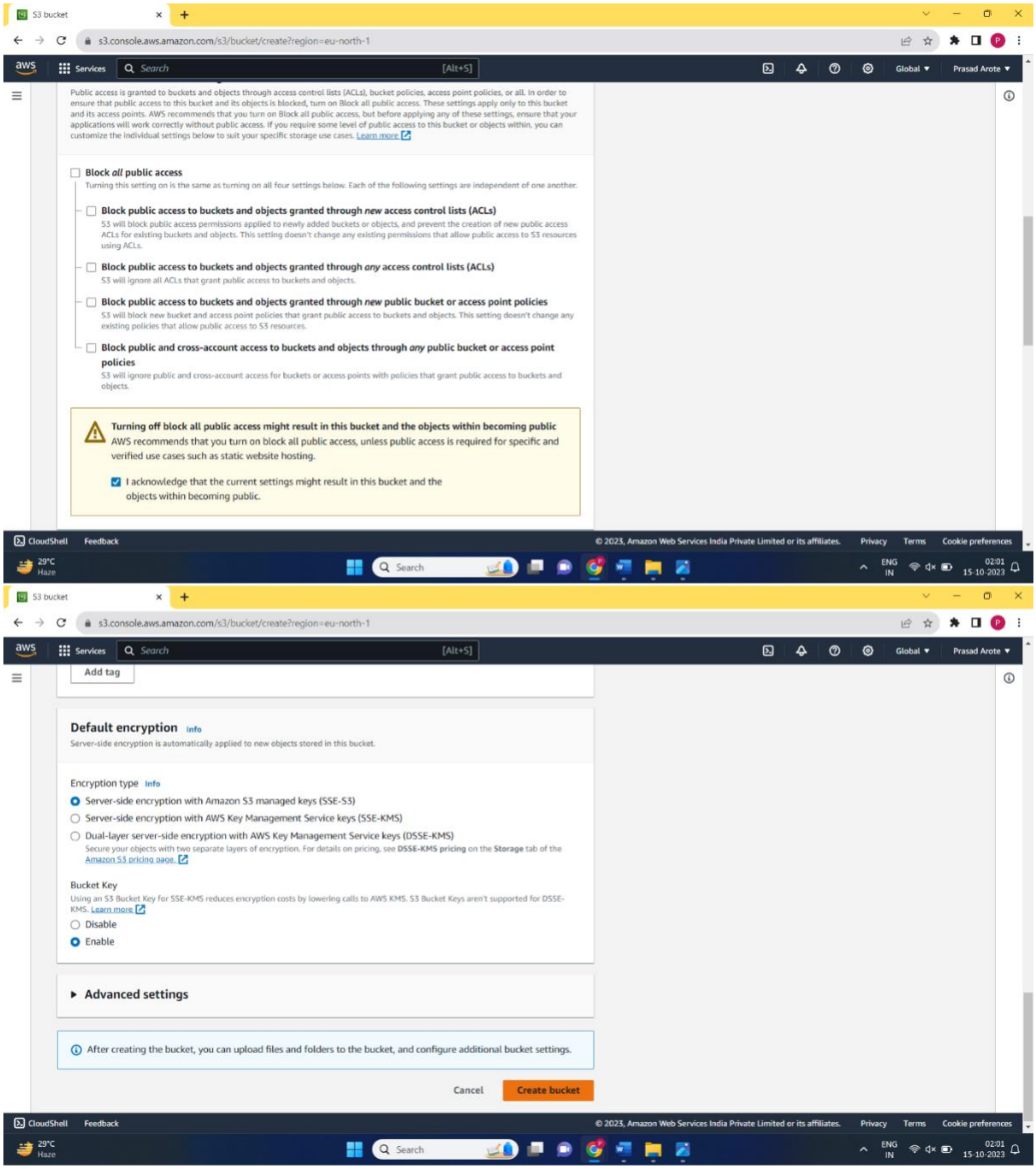
AIM: To study AWS S3 service and create a bucket for hosting static web application.

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

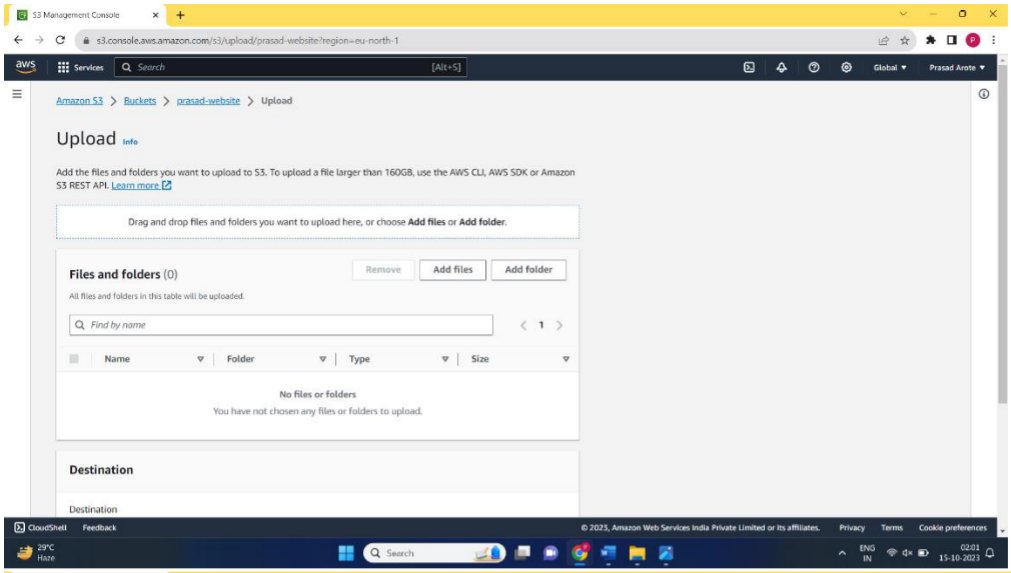
THEORY:

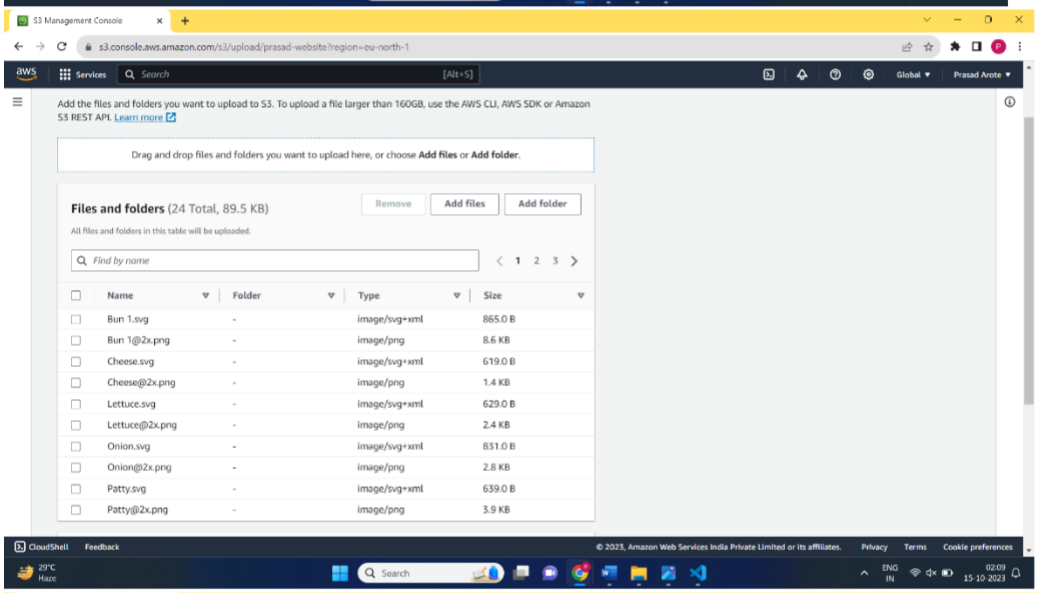
1. Create a S3 bucket.

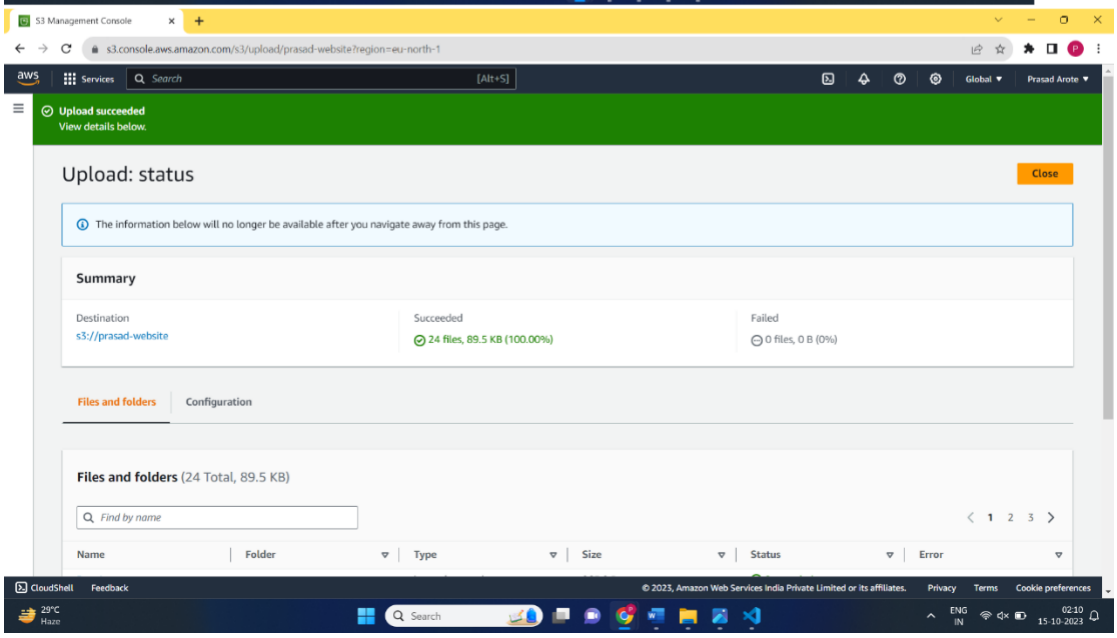




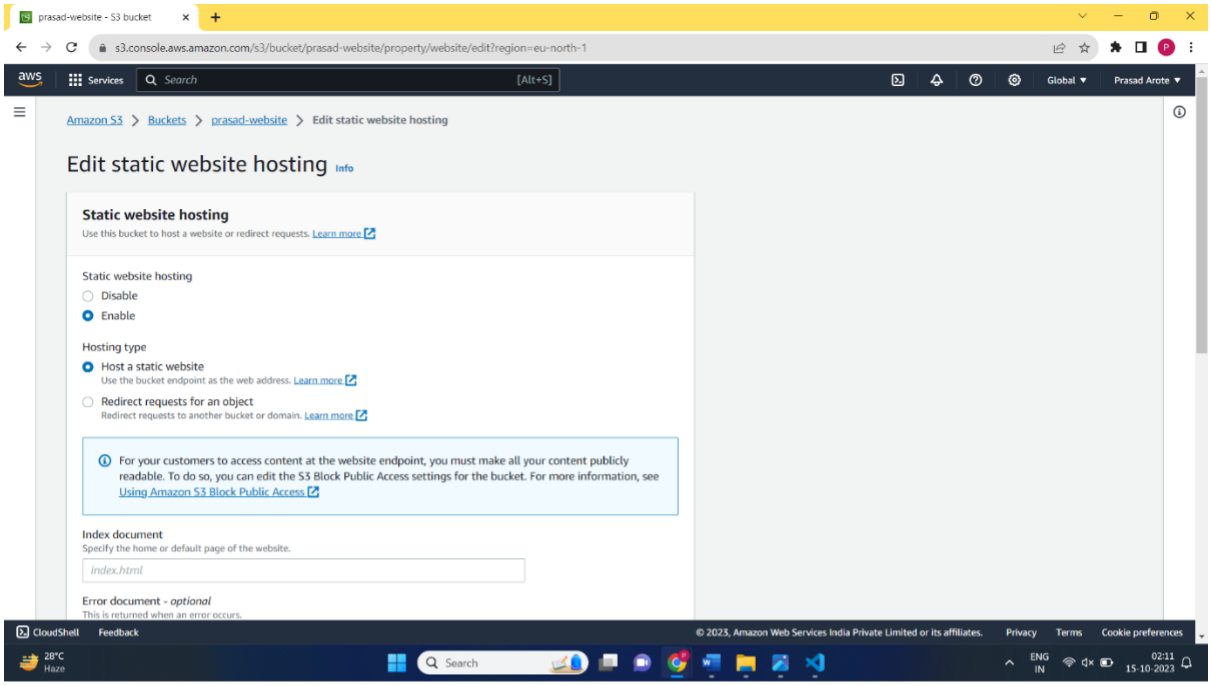
2. Upload the files of web application.

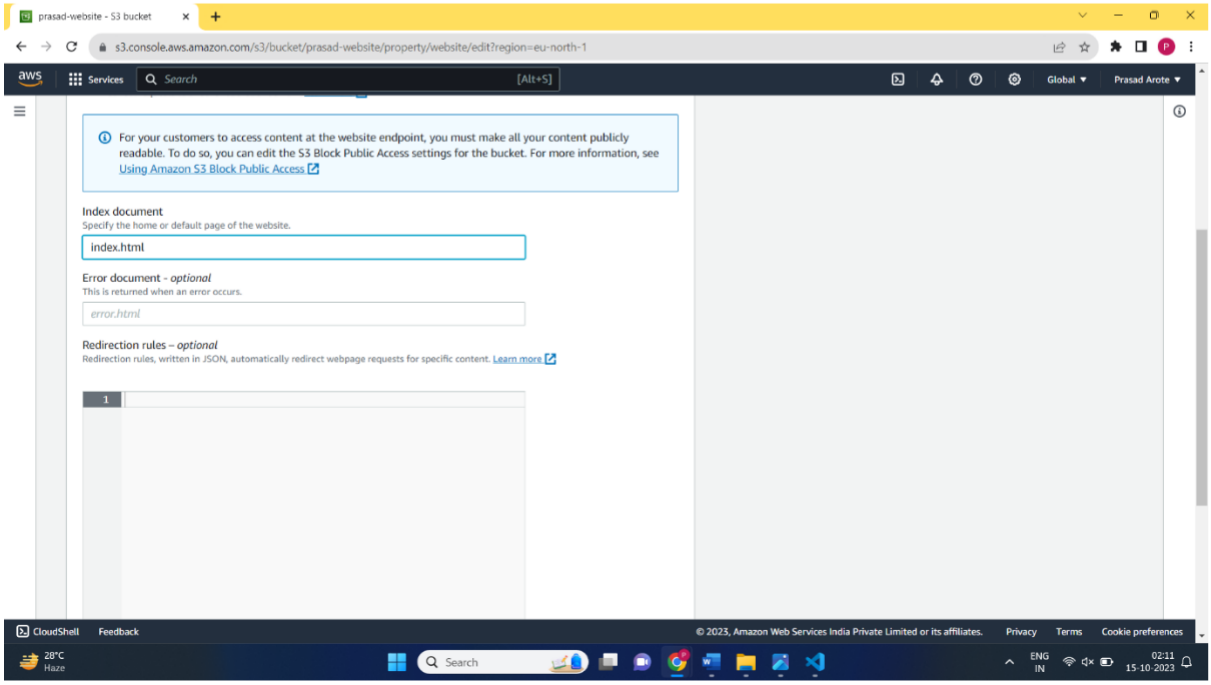

S3 Management Console
s3.console.aws.amazon.com/s3/upload/prasad-website?region=eu-north-1
Amazon S3 > Buckets > prasad-website > Upload
Upload Info
Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more
Drag and drop files and folders you want to upload here, or choose Add files or Add folder.
Files and folders (0)
Remove
Add files
Add folder
All files and folders in this table will be uploaded.
Find by name
Name
Folder
Type
Size
No files or folders
You have not chosen any files or folders to upload.
Destination
Destination
CloudShell
Feedback
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences

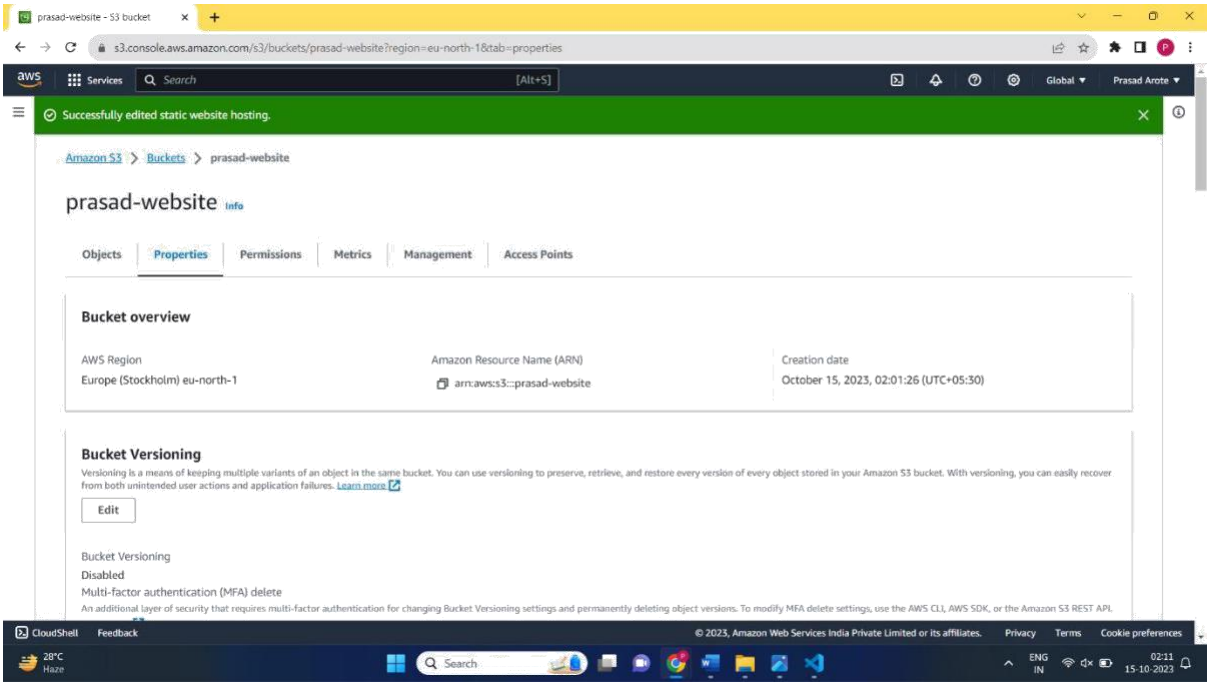

S3 Management Console
s3.console.aws.amazon.com/s3/upload/prasad-website?region=eu-north-1
Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more
Drag and drop files and folders you want to upload here, or choose Add files or Add folder.
Files and folders (24 Total, 89.5 KB)
Remove
Add files
Add folder
All files and folders in this table will be uploaded.
Find by name
Name
Folder
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 639.0 B
Patty@2x.png image/png 3.9 KB
CloudShell
Feedback
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences


S3 Management Console
s3.console.aws.amazon.com/s3/upload/prasad-website?region=eu-north-1
Upload succeeded
View details below.
Upload: status
Close
The information below will no longer be available after you navigate away from this page.
Summary
Destination
s3://prasad-website
Succeeded
24 files, 89.5 KB (100.00%)
Failed
0 files, 0 B (0%)
Files and folders
Configuration
Files and folders (24 Total, 89.5 KB)
Find by name
Name
Folder
Type
Size
Status
Error
CloudShell
Feedback
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences

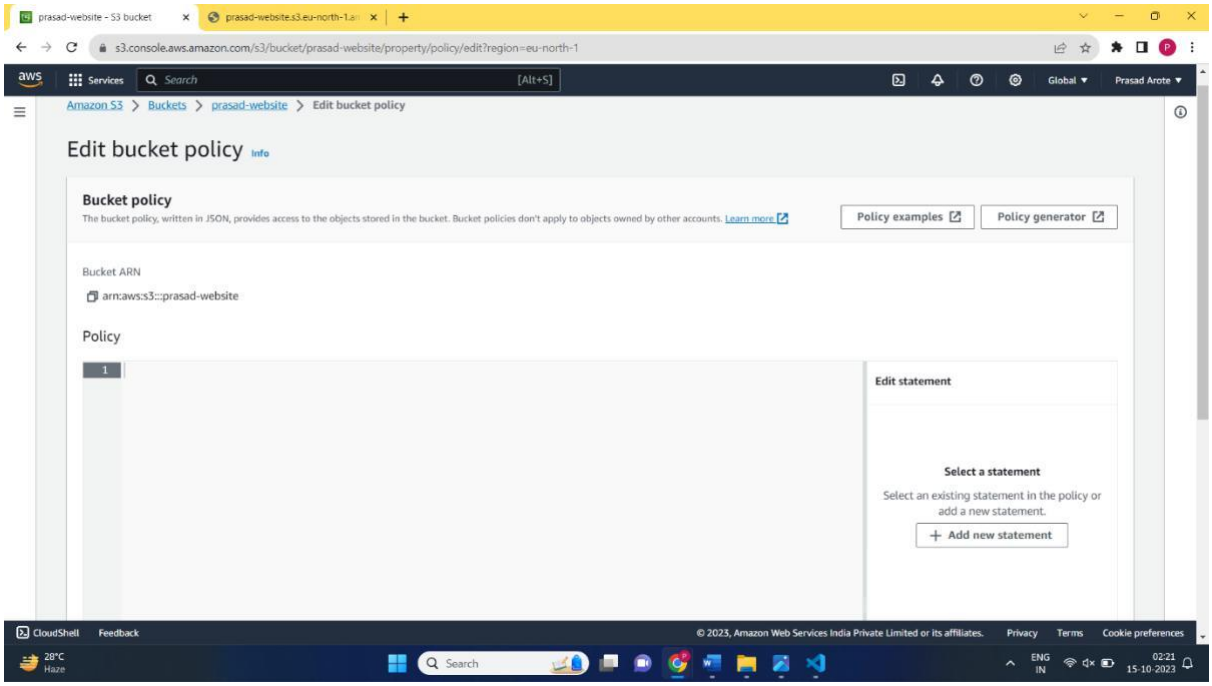
3. Enable Static website hosting

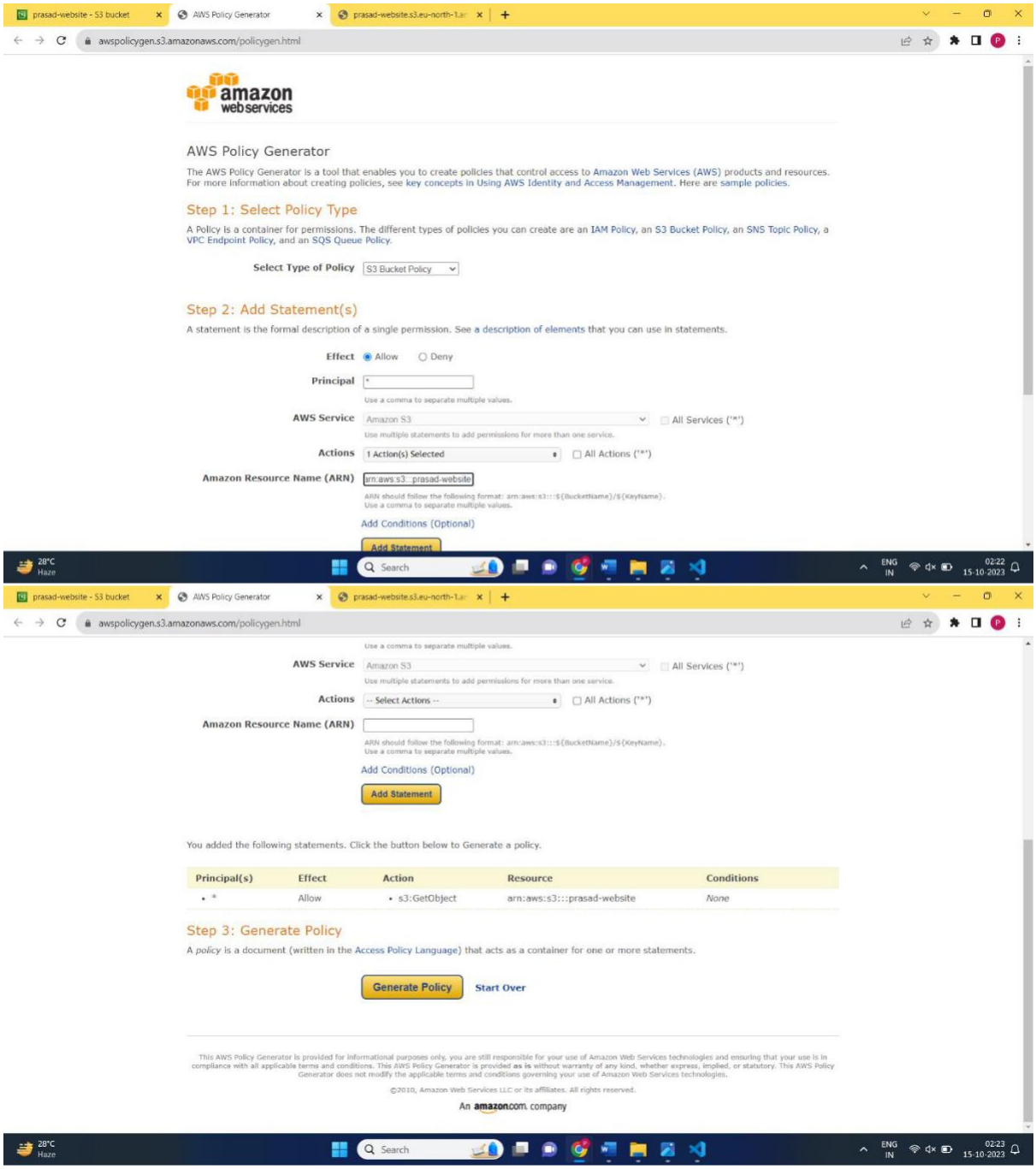


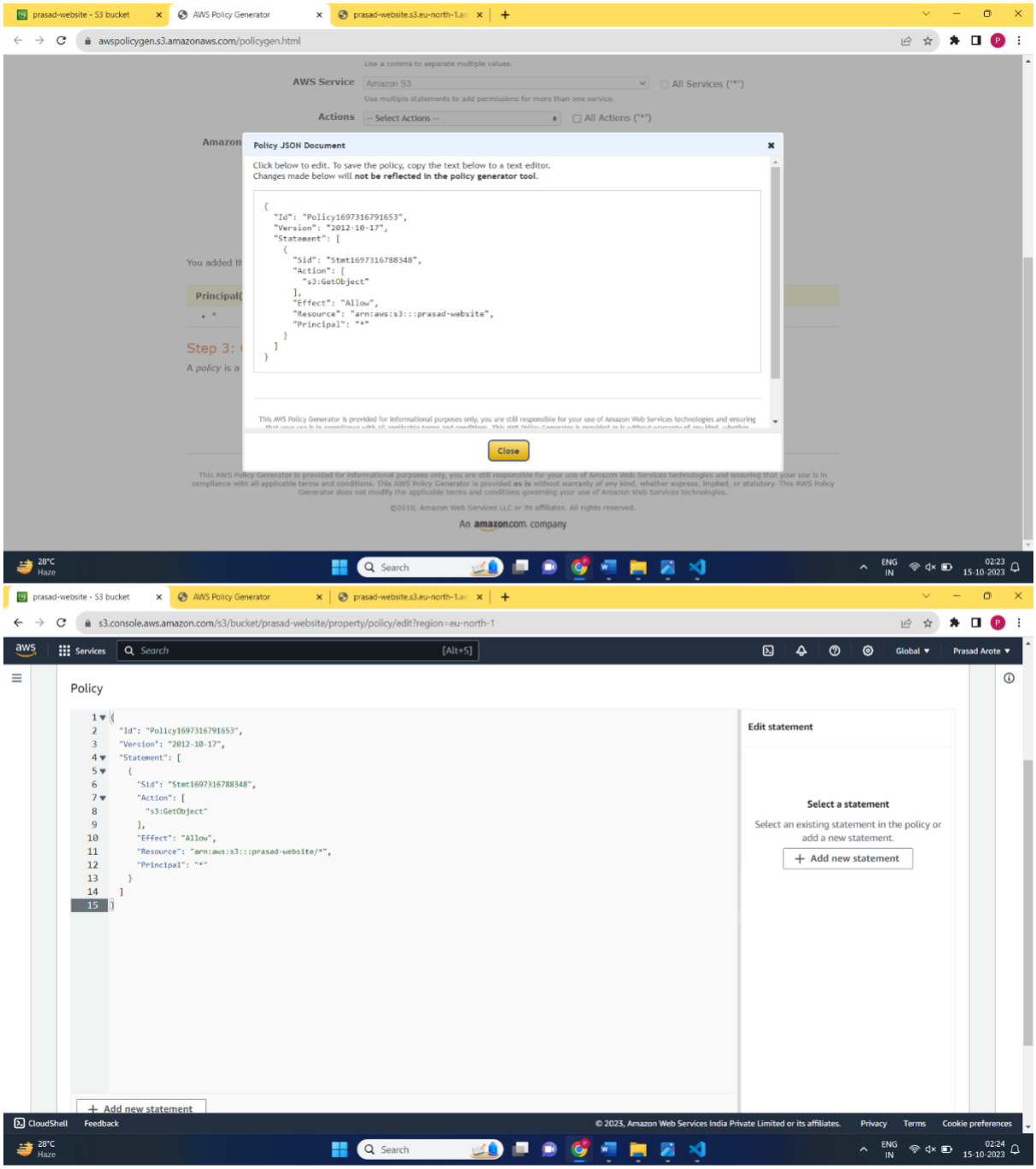




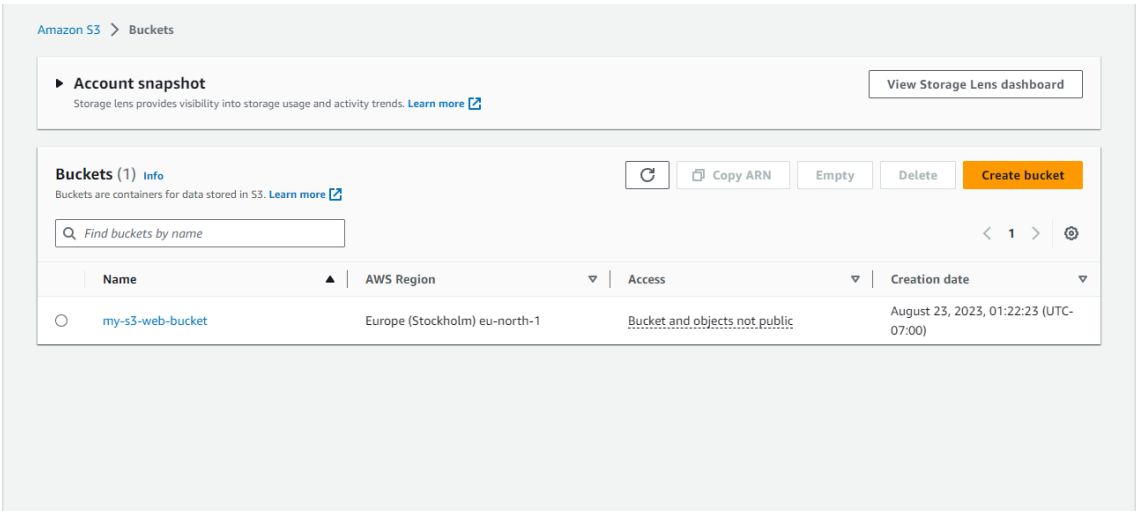
5. Change the Bucket Policy



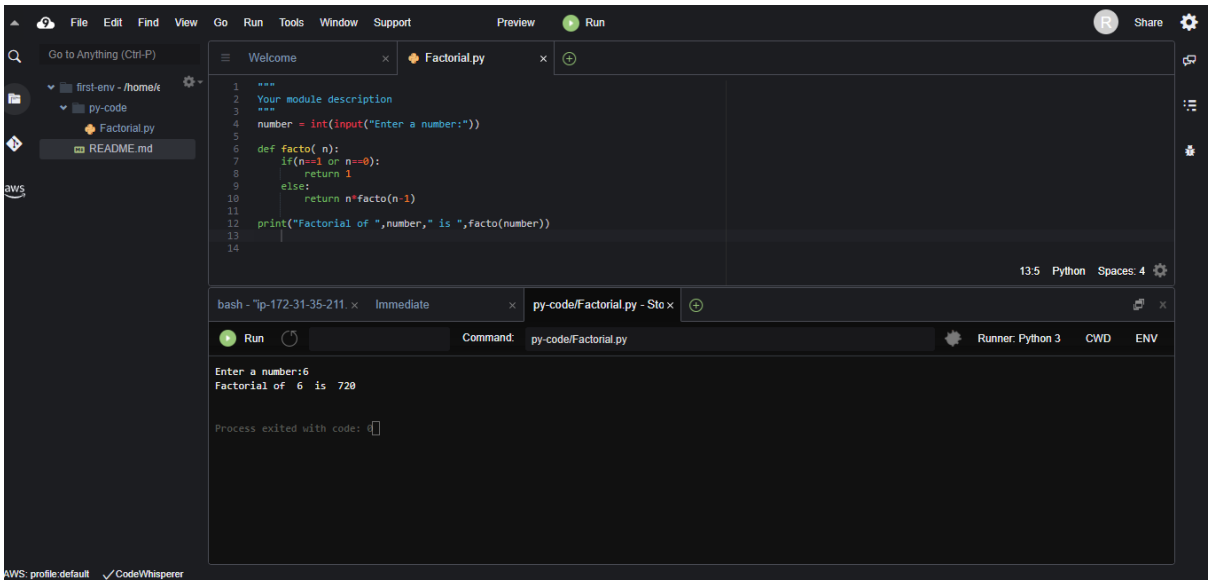

prasad-website - S3 bucket
AWS Policy Generator
awspolicygen.s3.amazonaws.com/policygen.html
amazon web services
AWS Policy Generator
The AWS Policy Generator is a tool that enables you to create policies that control access to Amazon Web Services (AWS) products and resources. For more information about 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
Deny
Principal
Use a comma to separate multiple values.
AWS Service
Amazon S3
All Services ('*')
Use multiple statements to add permissions for more than one service.
Actions
1 Action(s) Selected
All Actions ('*')
Amazon Resource Name (ARN)
arn:aws:s3:::prasad-website
ARN should follow the following format: arn:aws:s3:::${BucketName}/${KeyName}.
Use a comma to separate multiple values.
Add Conditions (Optional)
Add Statement
28°C Haze
Search
ENG IN
02:22 15-10-2023
prasad-website - S3 bucket
AWS Policy Generator
awspolicygen.s3.amazonaws.com/policygen.html
Use a comma to separate multiple values.
AWS Service
Amazon S3
All Services ('*')
Use multiple statements to add permissions for more than one service.
Actions
-- Select Actions --
All Actions ('*')
Amazon Resource Name (ARN)
ARN should follow the following format: arn:aws:s3:::${BucketName}/${KeyName}.
Use a comma to separate multiple values.
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
28°C Haze
Search
ENG IN
02:23 15-10-2023


prasad-website - S3 bucket
AWS Policy Generator
awspolicygen.s3.amazonaws.com/policygen.html
AWS Service
Amazon S3
All Services ('*')
Actions
-- Select Actions --
All Actions ('*')
Policy JSON Document
Click below to edit. To save the policy, copy the text below to a text editor.
Changes made below will not be reflected in the policy generator tool.
{
"Id": "Policy1697316791653",
"Version": "2012-10-17",
"Statement": [
{
"Sid": "Stmt1697316788348",
"Action": [
"s3:GetObject"
],
"Effect": "Allow",
"Resource": "arn:aws:s3:::prasad-website",
"Principal": "*"
}
]
}
Close
Step 3:
A policy is a
Principal(
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
28°C Haze
Search
ENG IN
02:23 15-10-2023

prasad-website - S3 bucket
AWS Policy Generator
s3.console.aws.amazon.com/s3/bucket/prasad-website/property/policy/edit?region=eu-north-1
aws
Services
Search
[Alt+S]
Global
Prasad Arote
Policy
{
"Id": "Policy1697316791653",
"Version": "2012-10-17",
"Statement": [
{
"Sid": "Stmt1697316788348",
"Action": [
"s3:GetObject"
],
"Effect": "Allow",
"Resource": "arn:aws:s3:::prasad-website/*",
"Principal": "*"
}
]
}
Edit statement
Select a statement
Select an existing statement in the policy or add a new statement.
+ Add new statement
+ Add new statement
CloudShell
Feedback
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences
28°C Haze
Search
ENG IN
02:24 15-10-2023



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



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:

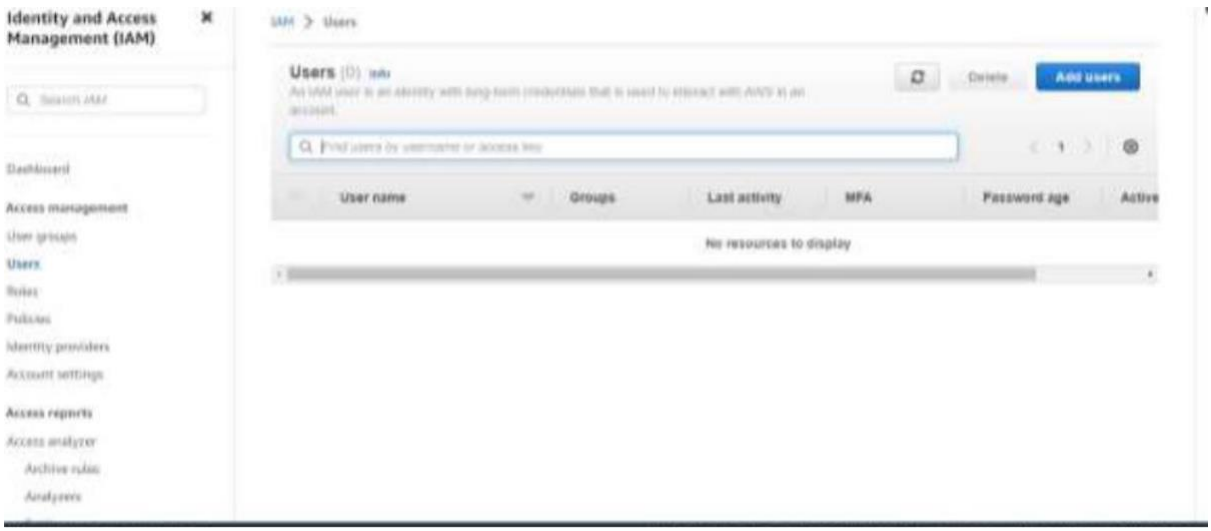
AWS CodePipeline is a fully managed continuous integration and continuous delivery (CI/CD) service provided by Amazon Web Services (AWS). It enables you to automate the building, testing, and deployment of your applications or code changes, facilitating the release of software changes more quickly and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

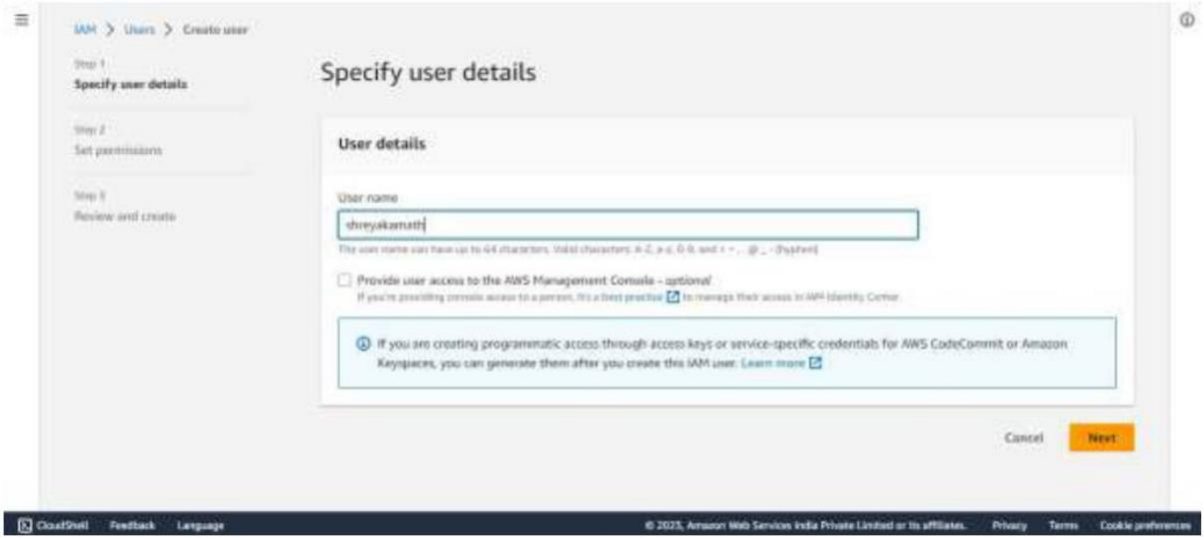
AWS CodePipeline works by creating a series of stages in a pipeline, each of which can perform various actions, such as source code retrieval, building, testing, and deployment. It can integrate with various AWS services and other tools, providing a flexible and extensible way to set up your CI/CD workflows.

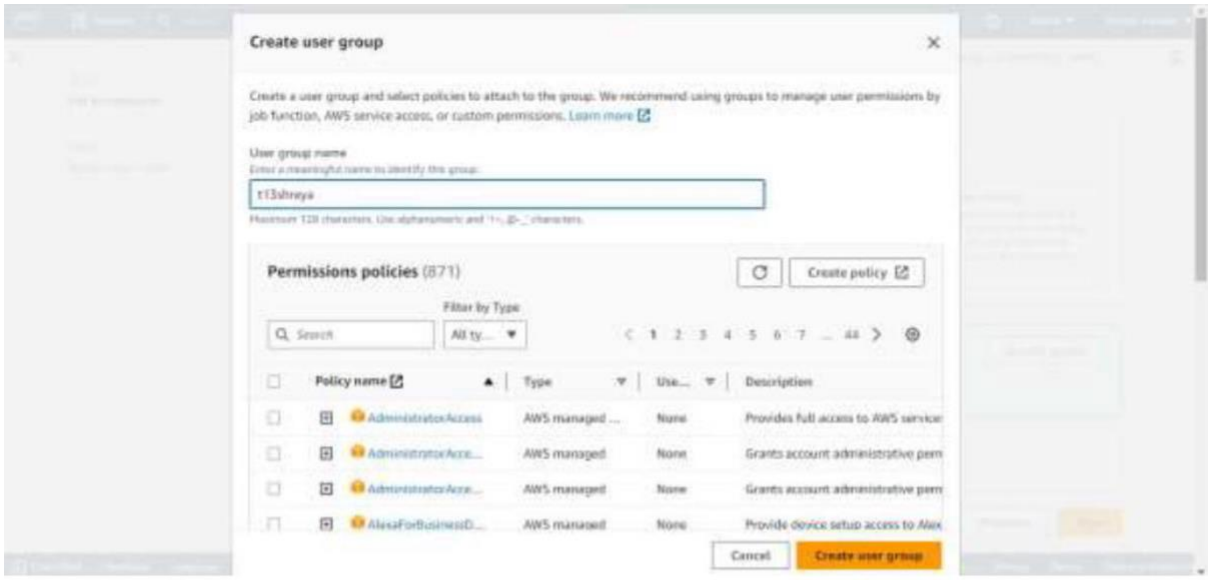
To deploy a static web application using AWS CodePipeline, you can follow these high-level steps:

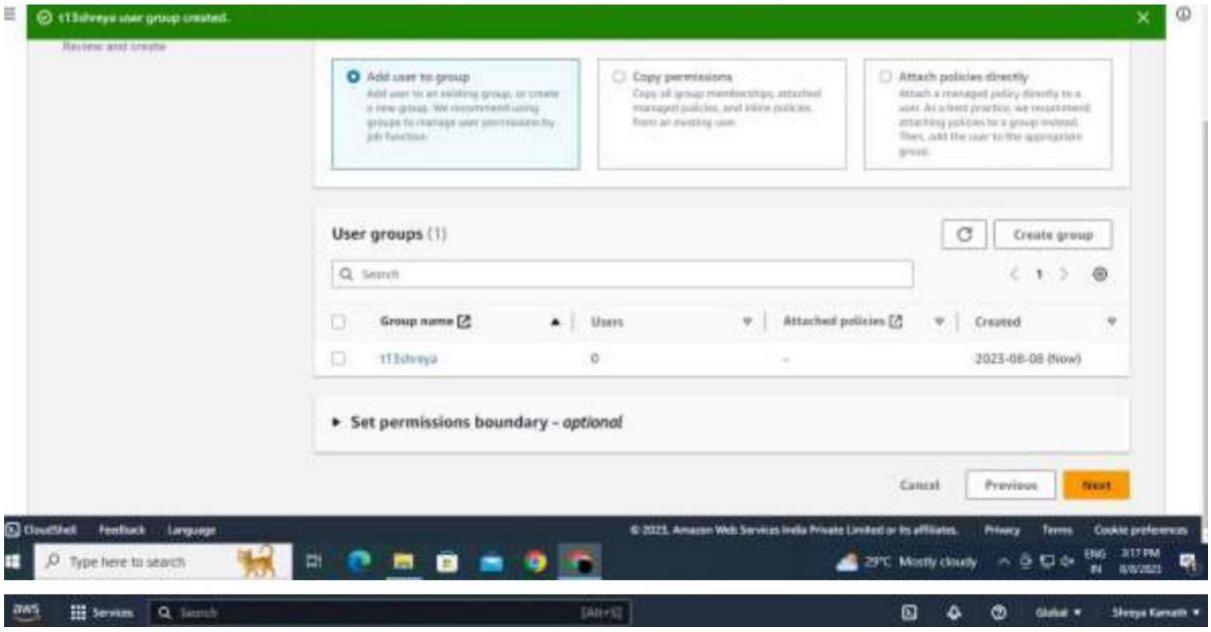
1. Create an Amazon S3 bucket to host your web app files.
2. Set up an AWS CodeCommit or other source code repository.
3. Create an AWS CodePipeline.
4. Define the source provider (CodeCommit, GitHub, S3).
5. Configure a build stage using AWS CodeBuild or similar services.
6. Set up optional testing 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.

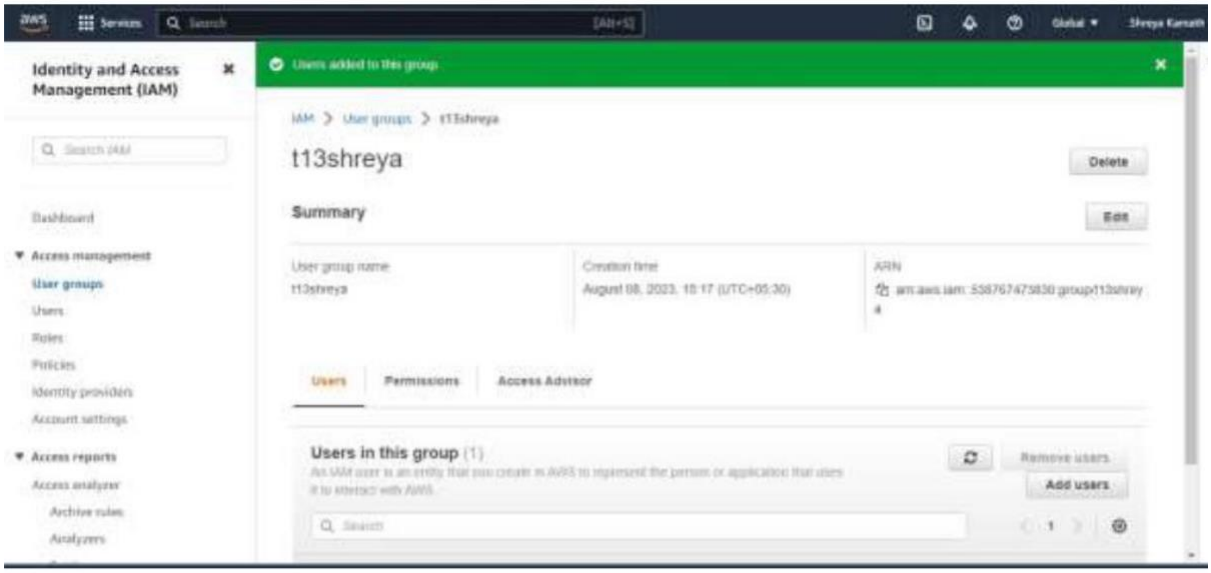
Output:

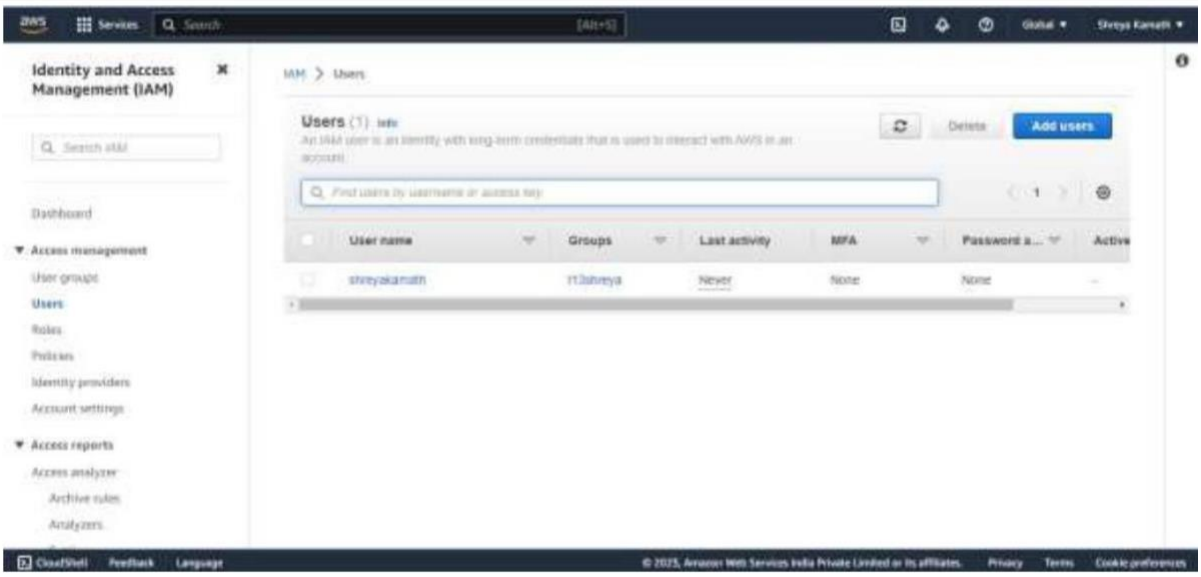

Identity and Access Management (IAM)
IAM > Users
Users (0)
Delete
Add users
User name
Groups
Last activity
MFA
Password age
Active
No resources to display
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access analyzer
Archive rules
Analyzers

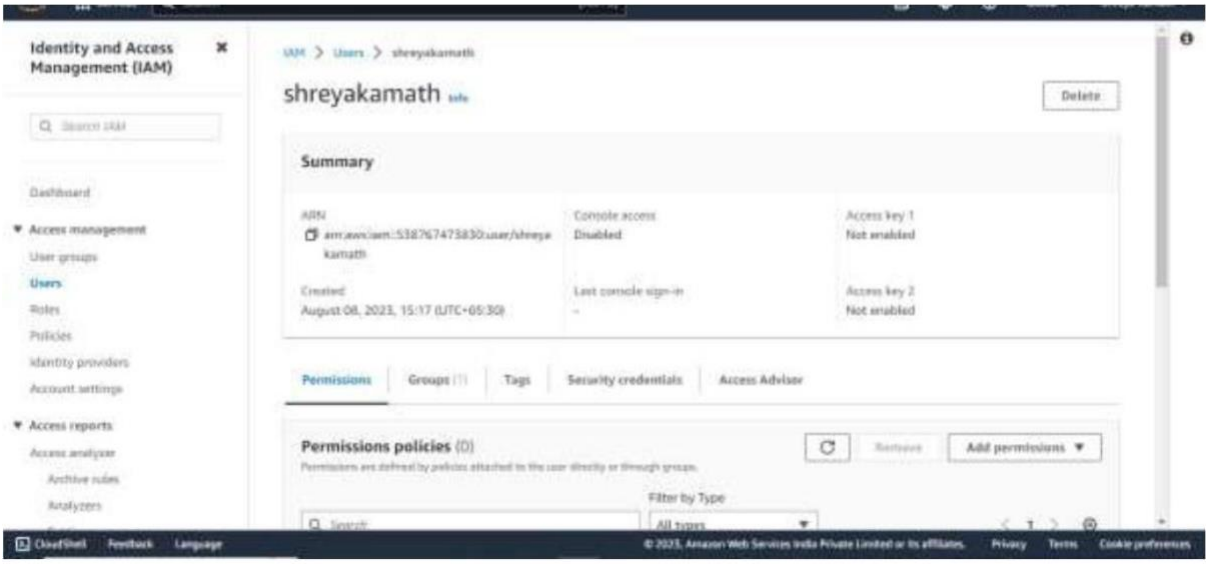

IAM > Users > Create user
Step 1
Specify user details
Step 2
Set permissions
Step 3
Review and create
Specify user details
User details
User name
shreyakamath
Provide user access to the AWS Management Console - optional
If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. Learn more
Cancel
Next
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences

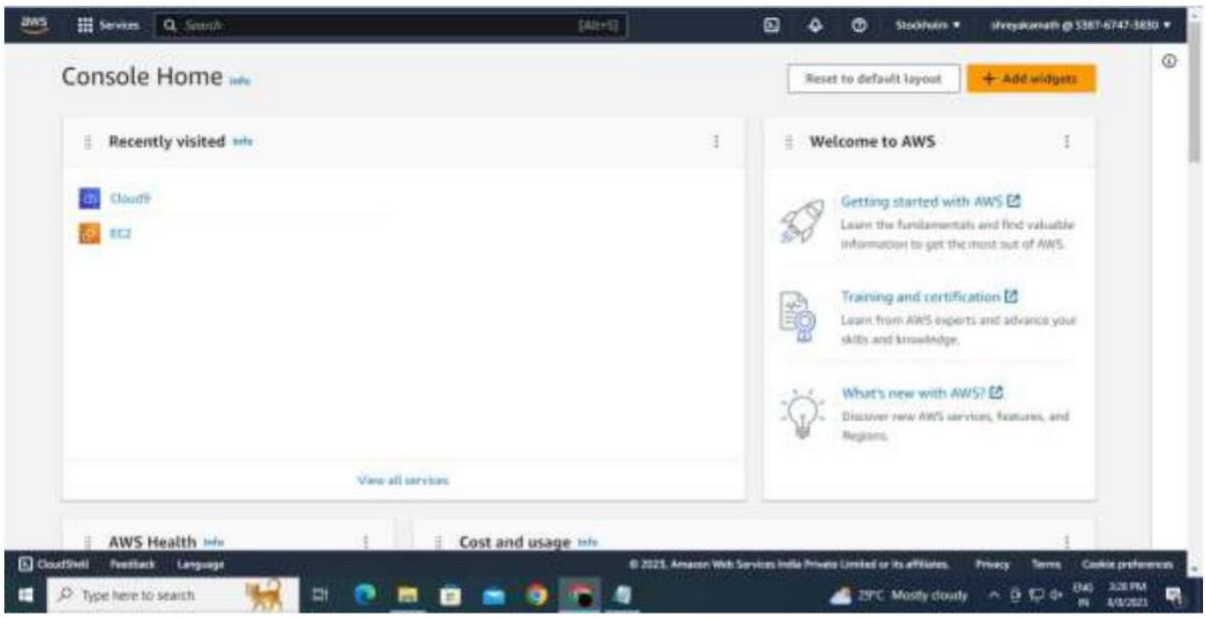

Create user group
Create a user group and select policies to attach to the group. We recommend using groups to manage user permissions by job function, AWS service access, or custom permissions. Learn more
User group name
t13shreya
Permissions policies (871)
Create policy
Filter by Type
Search
All ty...
Policy name
Type
Use...
Description
AdministratorAccess
AWS managed ...
None
Provides full access to AWS service
AWS managed
Grants account administrative perm
Cancel
Create user group

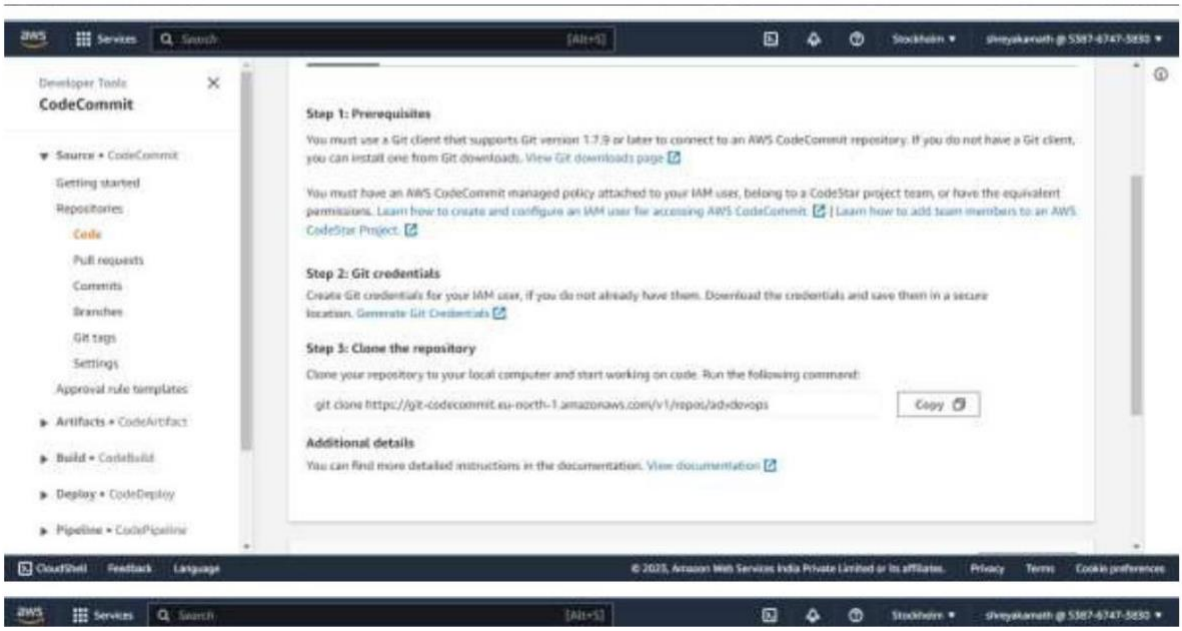

t13shreya user group created.
Review and create
Add user to group
Copy permissions
Attach policies directly
User groups (1)
Create group
Search
Group name
Users
Attached policies
Created
t13shreya
0
2023-08-08 (Now)
Set permissions boundary - optional
Cancel
Previous
Next
CloudShell
Feedback
Language
Privacy
Terms
Cookie preferences
Type here to search
29°C Mostly cloudy
Services
Search
[Alt+S]
Global
Shreya Kamath

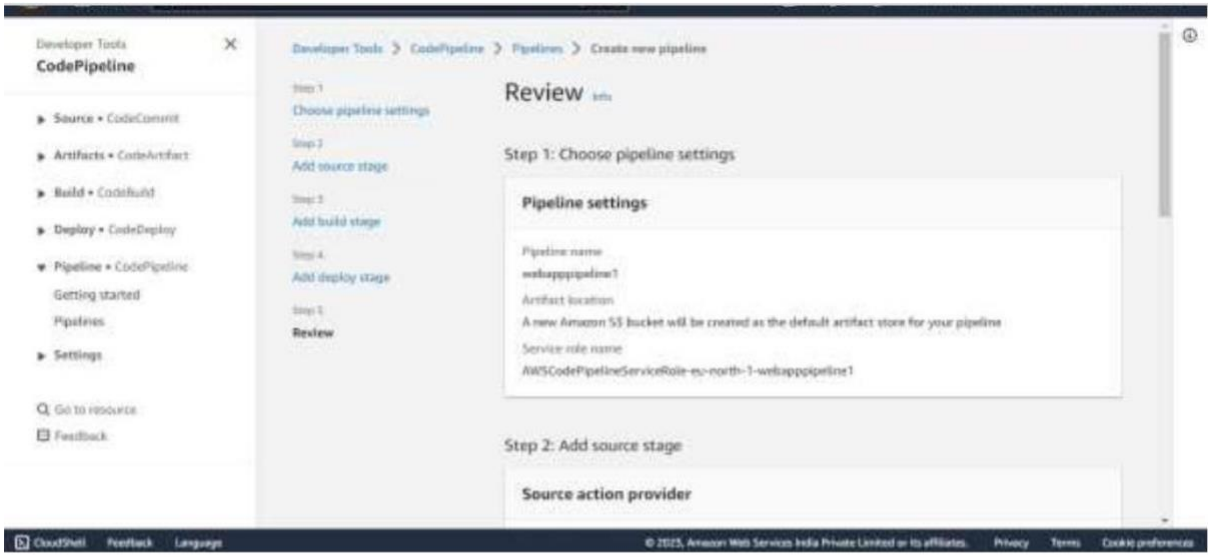

Identity and Access Management (IAM)
Users added to this group
IAM > User groups > t13shreya
t13shreya
Delete
Summary
Edit
User group name
t13shreya
Creation time
August 08, 2023, 15:17 (UTC+05:30)
ARN
Users
Permissions
Access Advisor
Users in this group (1)
Remove users
Add users

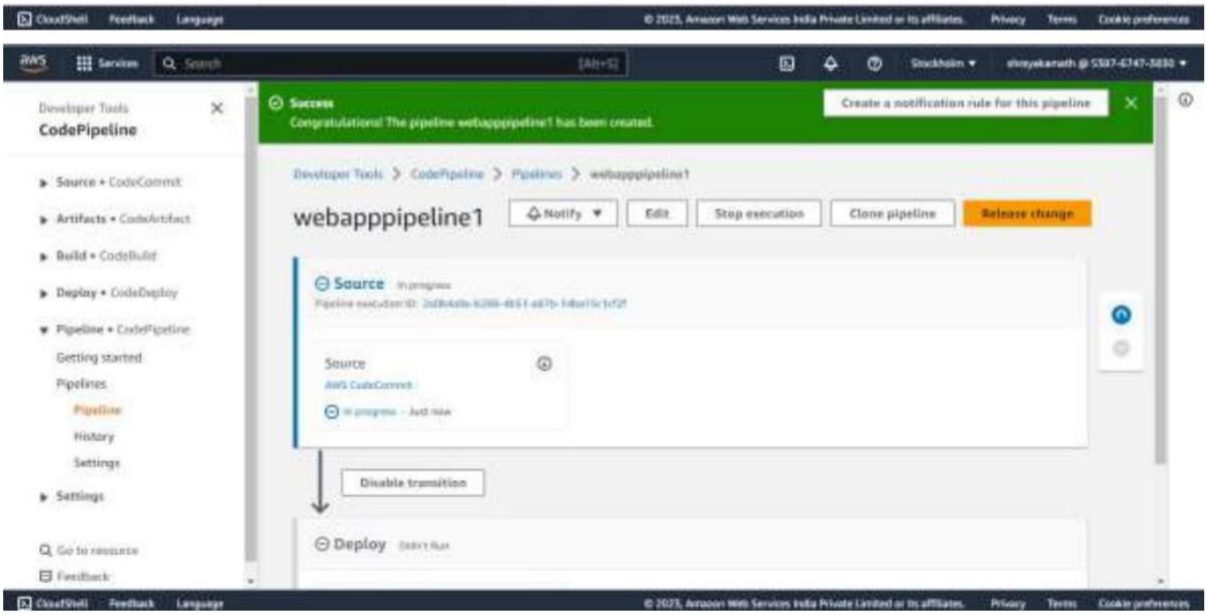

Identity and Access Management (IAM)
IAM > Users
Users (1)
Delete
Add users
User name
Groups
Last activity
MFA
Password a...
shreyakamath
t13shreya
Never
None
None


Identity and Access Management (IAM)
IAM > Users > shreyakamath
shreyakamath
Delete
Summary
ARN
Console access
Disabled
Access key 1
Not enabled
Created
August 08, 2023, 15:17 (UTC+05:30)
Access key 2
Not enabled
Permissions
Groups (1)
Tags
Security credentials
Access Advisor
Permissions policies (0)
Add permissions
Filter by Type


Console Home
Reset to default layout
+ Add widgets
Recently visited
Cloud9
EC2
View all services
Welcome to AWS
Getting started with AWS
Learn the fundamentals and find valuable information to get the most out of AWS.
Training and certification
Learn from AWS experts and advance your skills and knowledge.
What's new with AWS?
Discover new AWS services, features, and Regions.
AWS Health
Cost and usage
CloudShell
Feedback
Language
Privacy
Terms
Cookie preferences
Type here to search


Developer Tools
CodeCommit
Source • CodeCommit
Getting started
Repositories
Code
Pull requests
Commits
Branches
Git tags
Settings
Approval rule templates
Artifacts • CodeArtifact
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Step 1: Prerequisites
You must use a Git client that supports Git version 1.7.9 or later to connect to an AWS CodeCommit repository. If you do not have a Git client, you can install one from Git downloads. View Git downloads page
You must have an AWS CodeCommit managed policy attached to your IAM user, belong to a CodeStar project team, or have the equivalent permissions. Learn how to create and configure an IAM user for accessing AWS CodeCommit. | Learn how to add team members to an AWS CodeStar Project.
Step 2: Git credentials
Create Git credentials for your IAM user, if you do not already have them. Download the credentials and save them in a secure location. Generate Git Credentials
Step 3: Clone the repository
Clone your repository to your local computer and start working on code. Run the following command:
git clone https://git-codecommit.eu-north-1.amazonaws.com/v1/repos/adxdevops
Copy
Additional details
You can find more detailed instructions in the documentation. View documentation
CloudShell
Feedback
Language
Privacy
Terms
Cookie preferences





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.

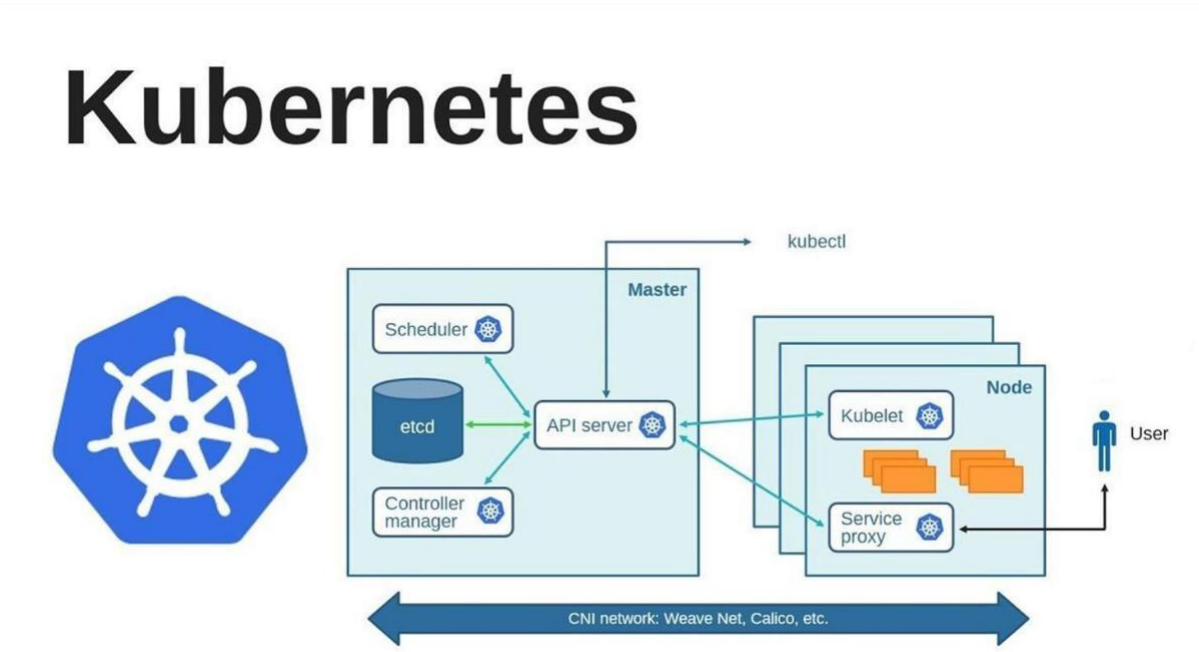
LO2. To deploy single and multiple container applications and manage application deployments with rollouts in Kubernetes.

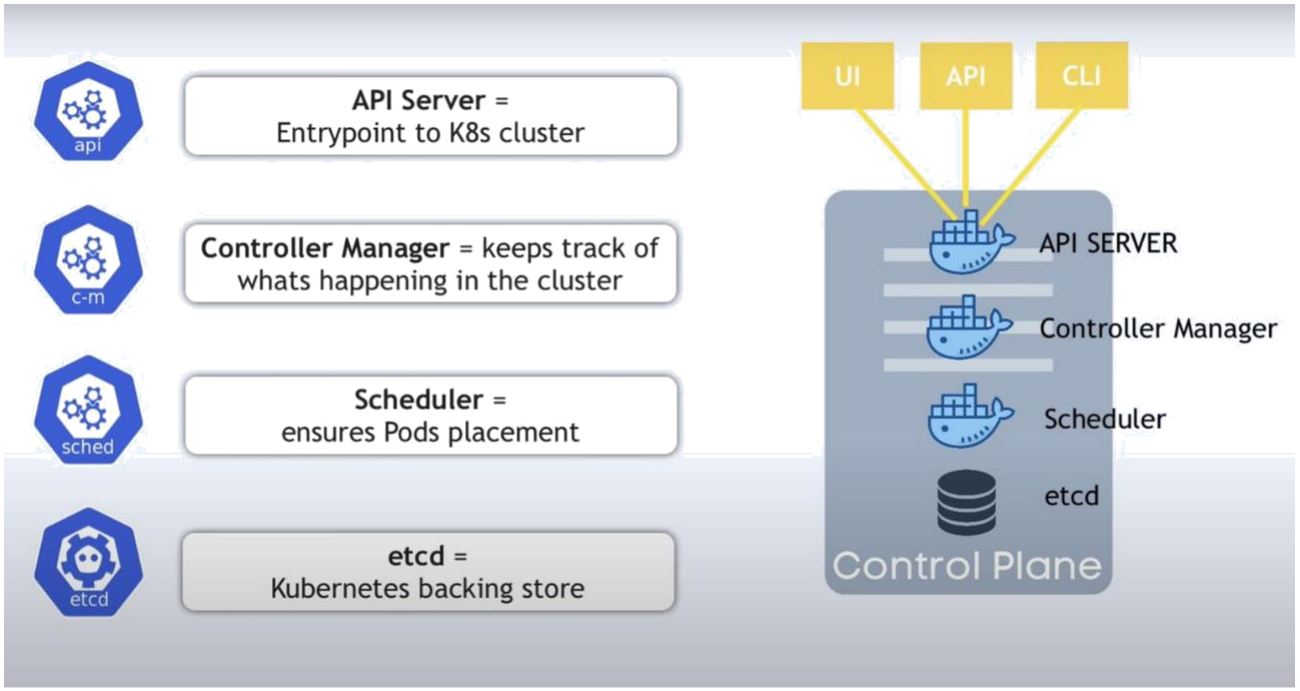
Theory:

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





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 or 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 touch 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-schedular (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

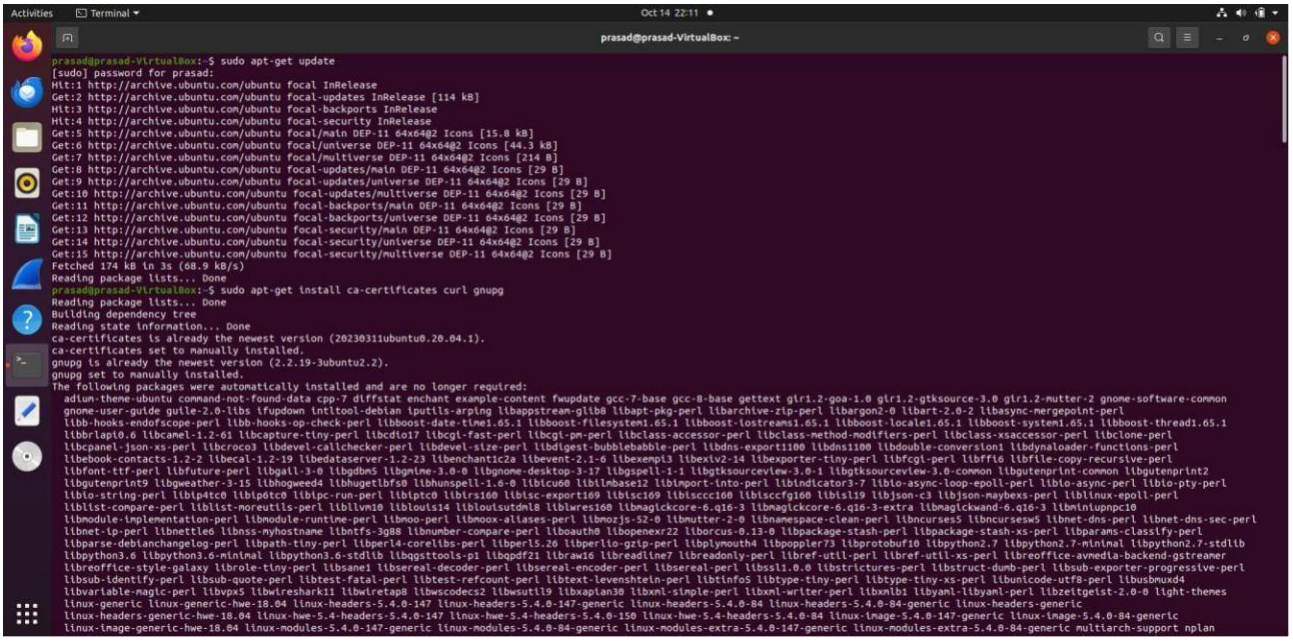
- Works with kubelet
- Pulling images
- Start/Stop Containers
- Exposing containers on ports specified in the manifest.

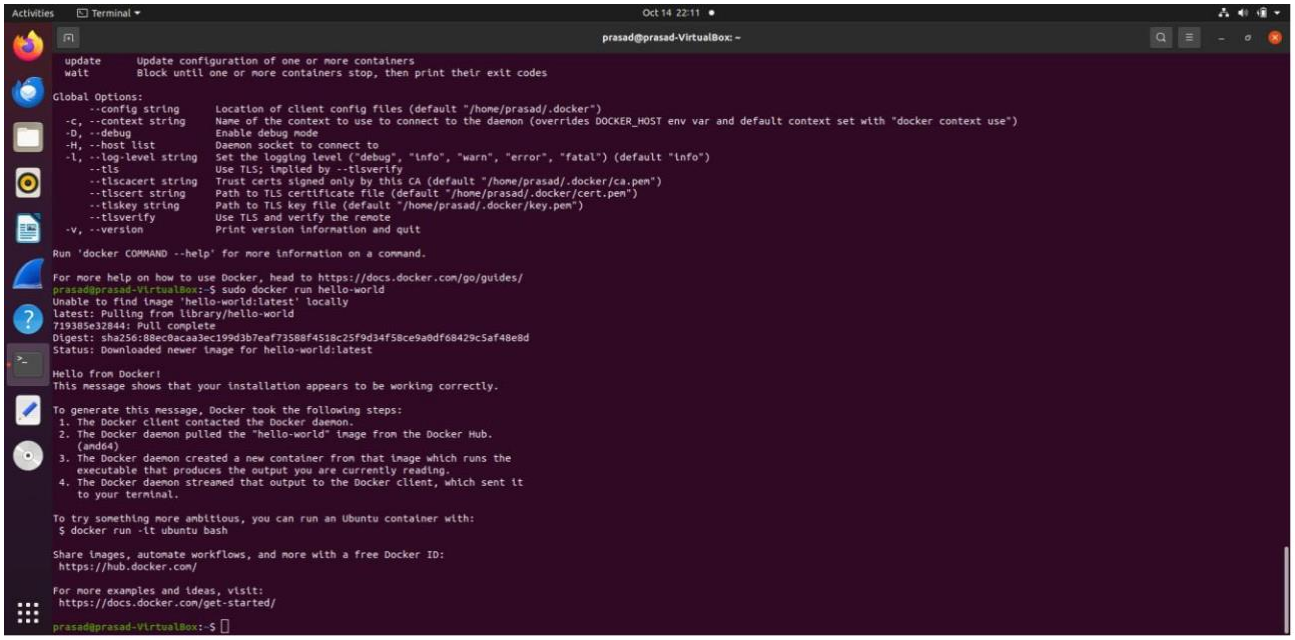
Kube-Proxy

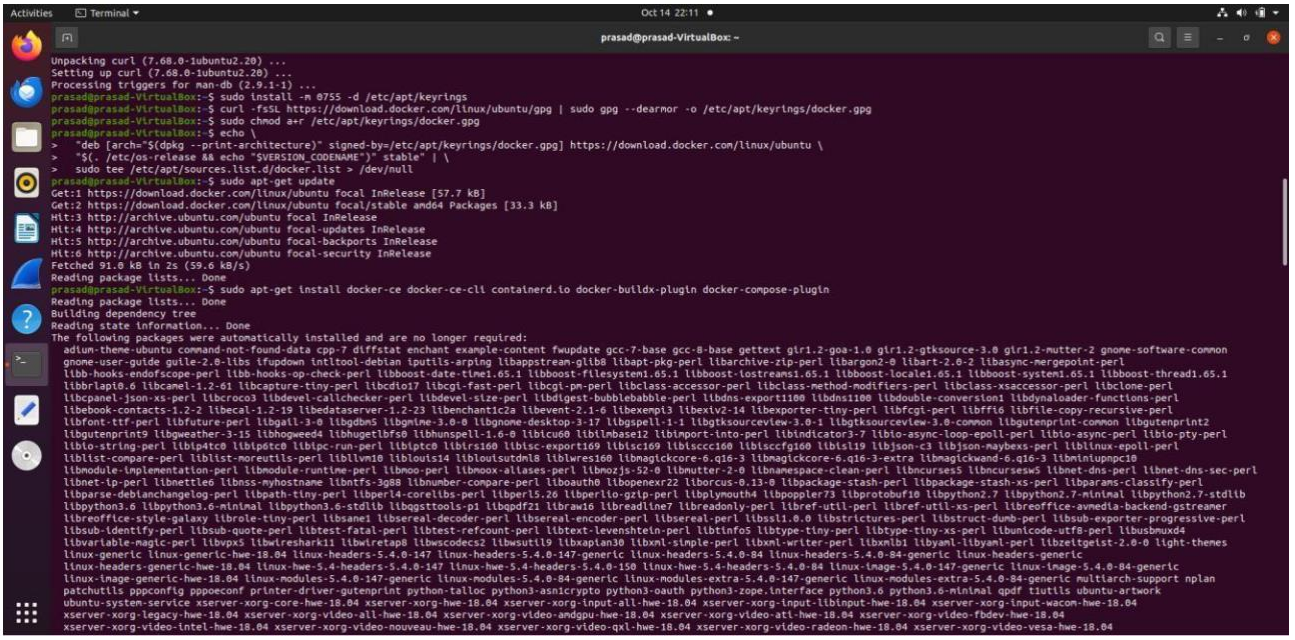
- Assign IP to each pod.
- It is required to assign IP addresses to Pods (dynamic)
- Kube-proxy runs on each node & this makes sure that each pod will get its unique IP Address.
- These 3 components collectively consist of ‘node’.

INSTALLATION:

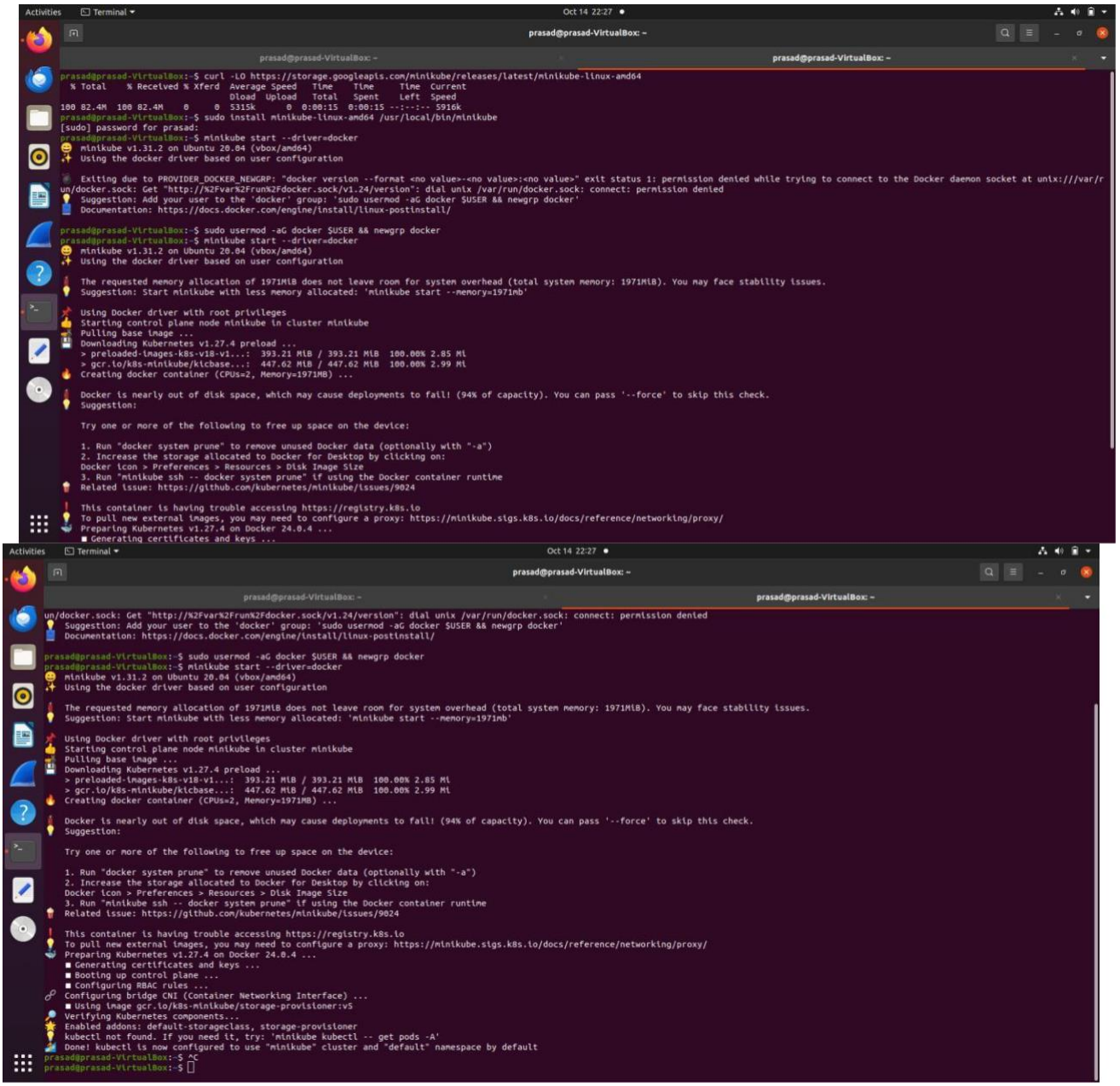
1. Install Docker



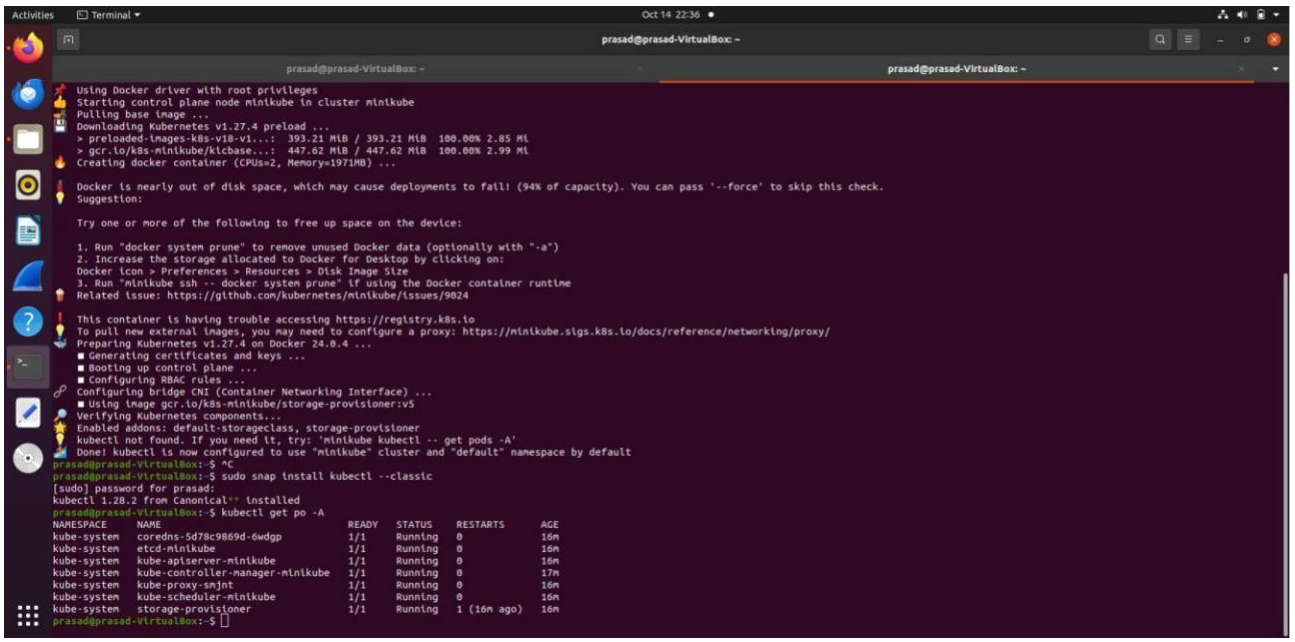




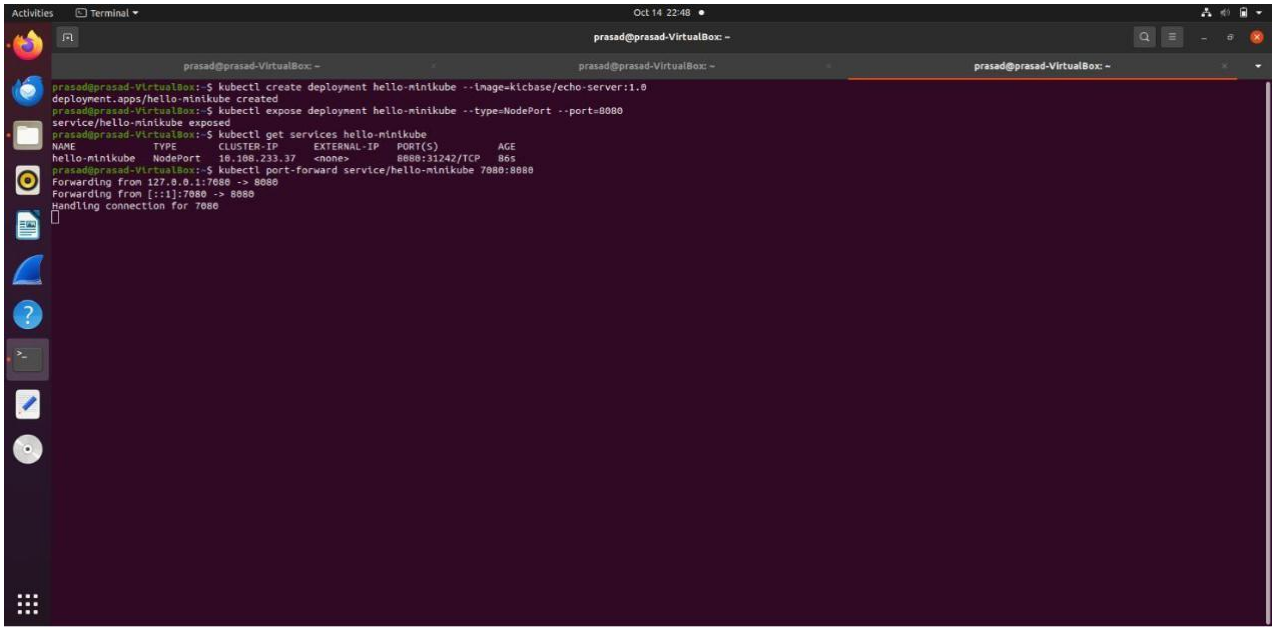
2. Install minikube using following commands

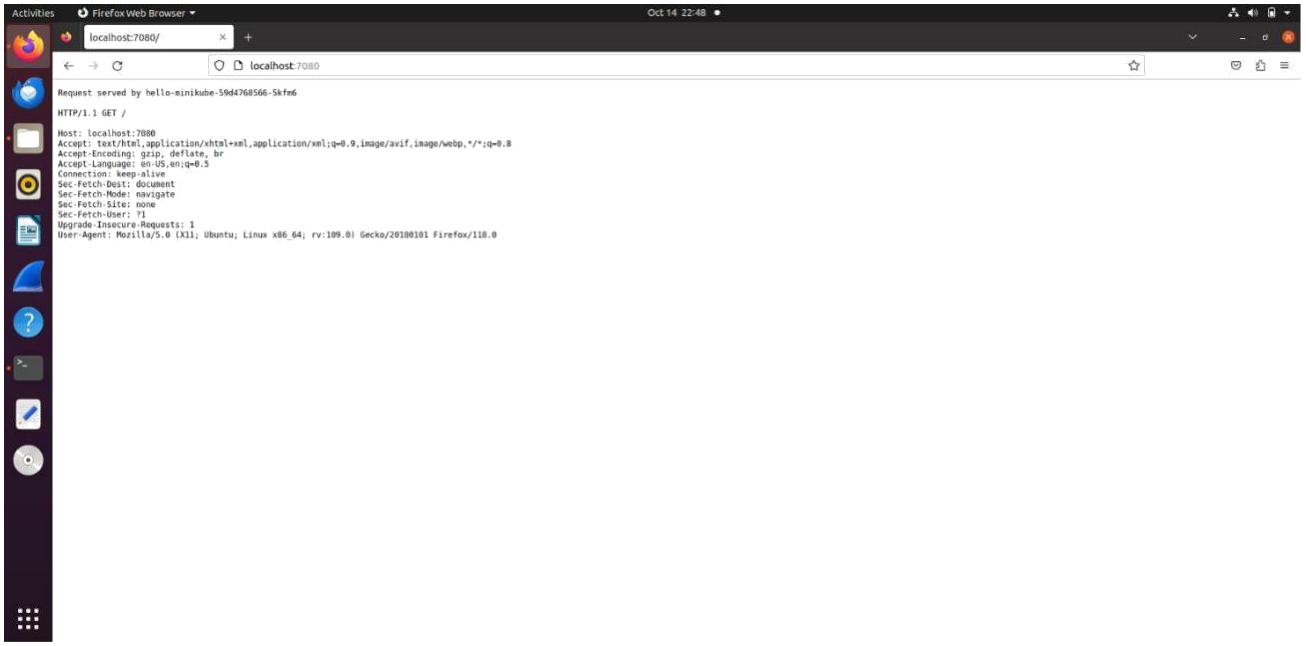


3. Install kubectl



4. Create a sample deployment.





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:

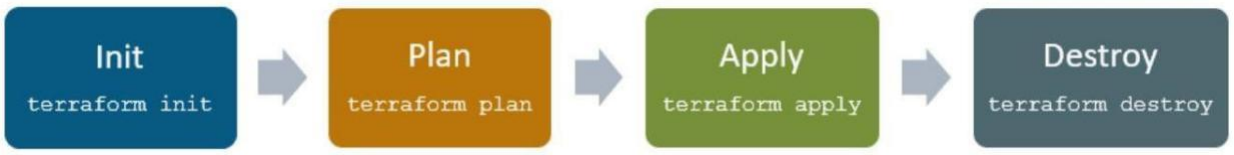
Terraform is one of the most popular Infrastructure-as-code (IaC) tool, used by DevOps teams to automate infrastructure tasks. It is used to automate the provisioning of your cloud resources. Terraform is an open-source, cloud-agnostic provisioning tool developed by HashiCorp and written in GO language.

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.

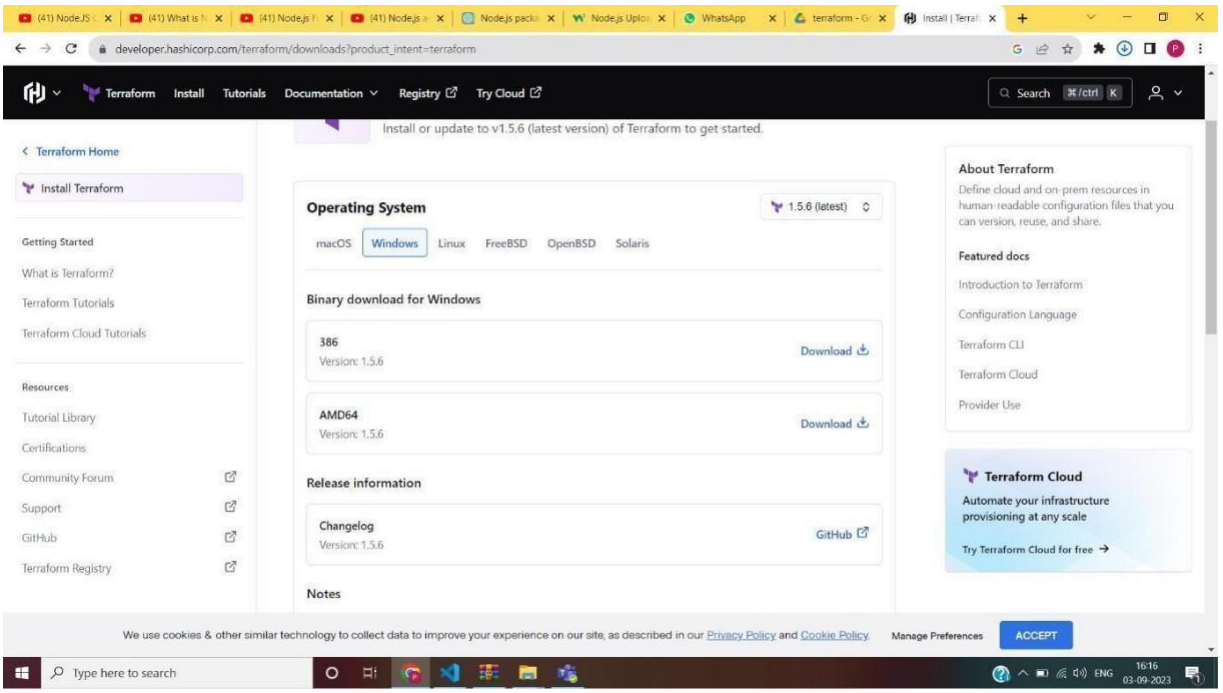
INSTALLATION:

1) Download Terraform

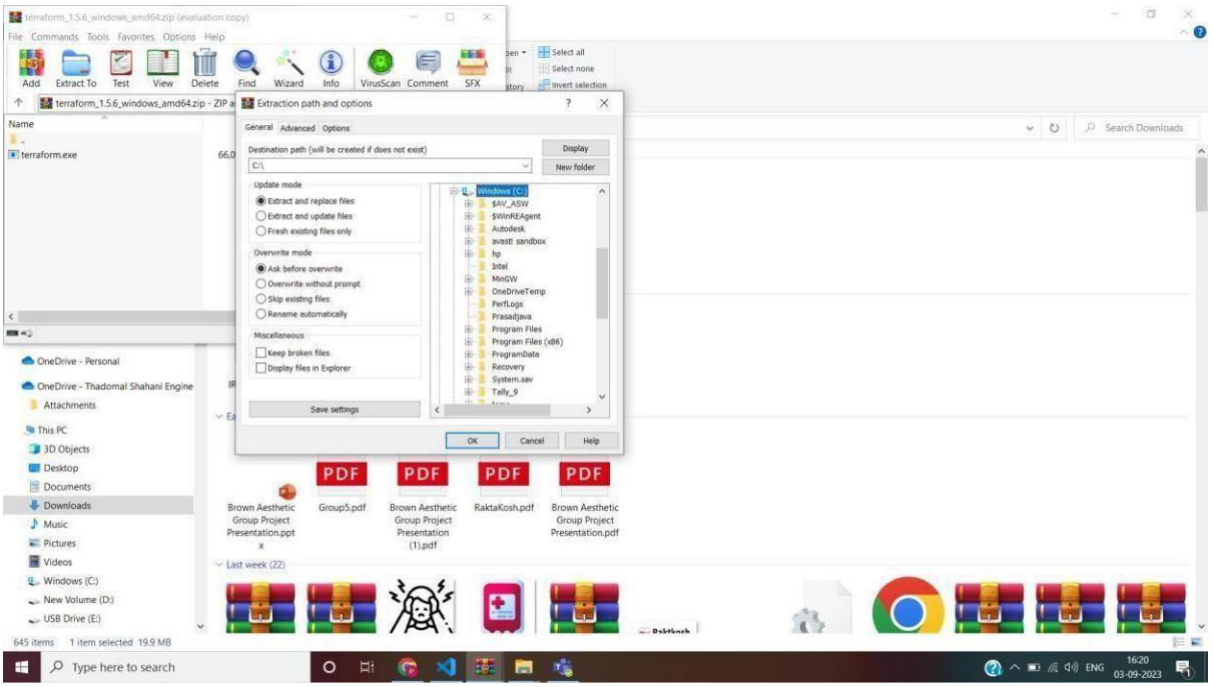
To install Terraform, First Download the Terraform Cli Utility for windows from terraforms official website

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

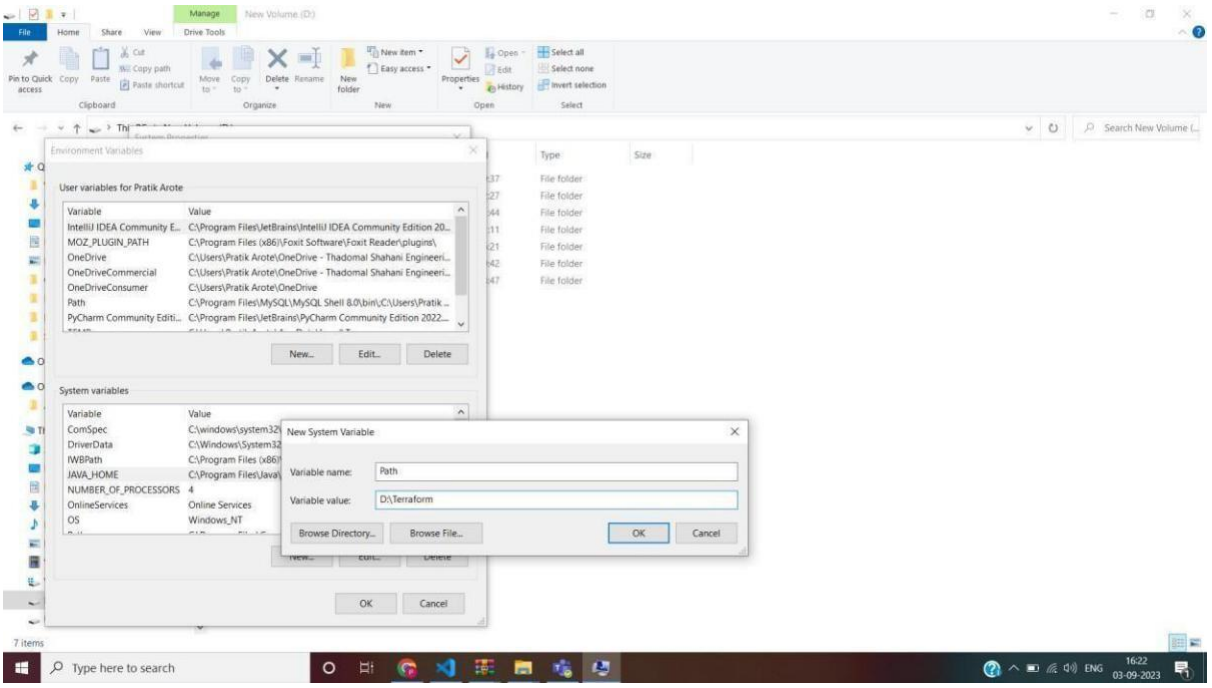
Select the Operating System Windows followed by either 32bit or 64 bit based on your OS type.



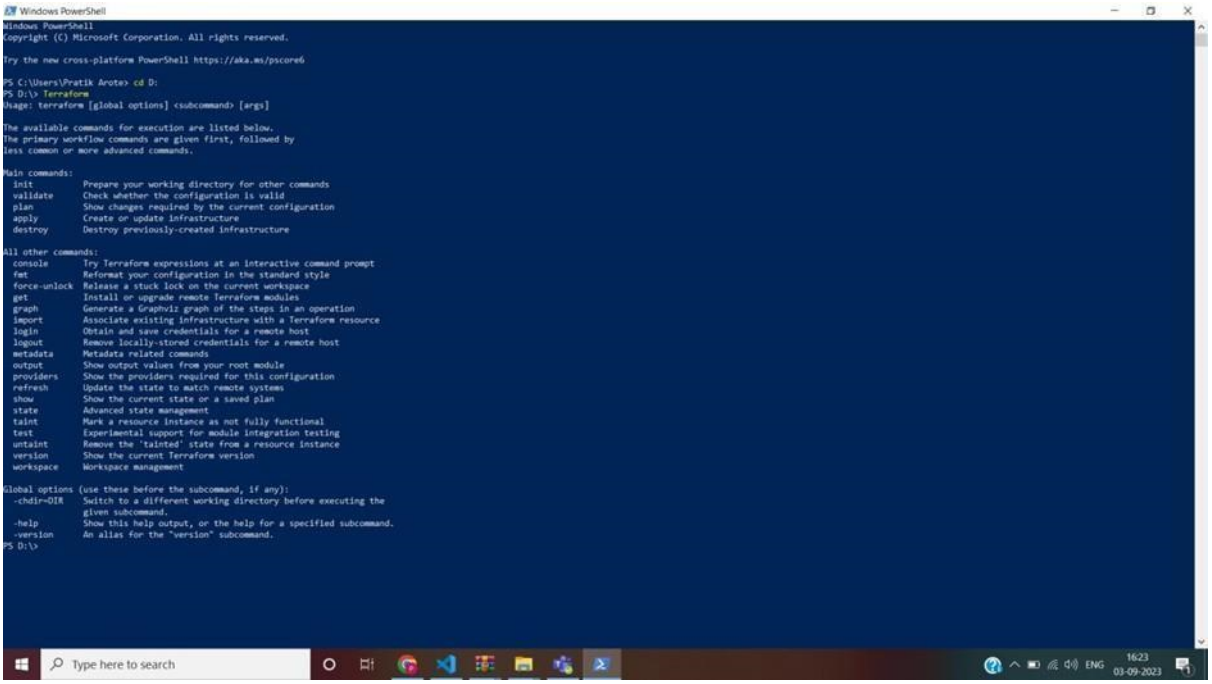
2. Extract the downloaded setup file Terraform.exe in C:\Terraform Directory



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



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



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.

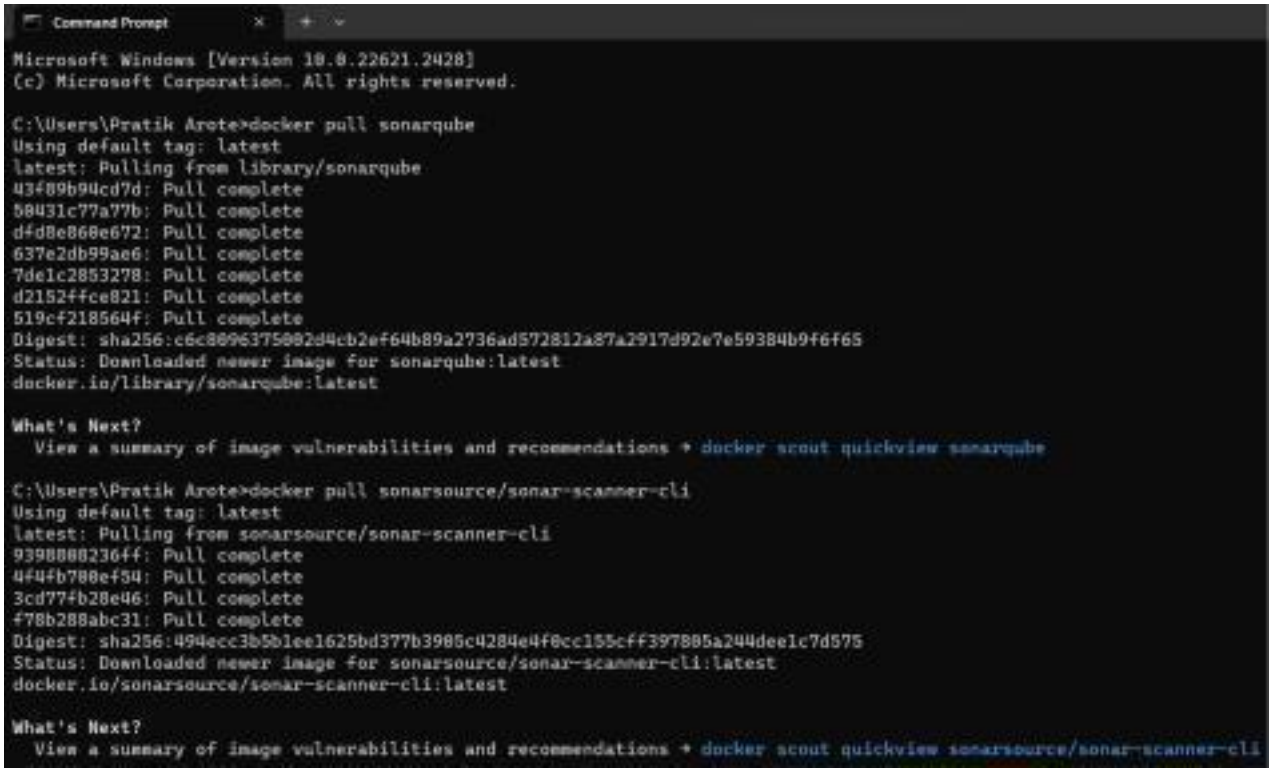
Early Detection: SAST is used early in the development process to find security issues before they can be exploited.

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

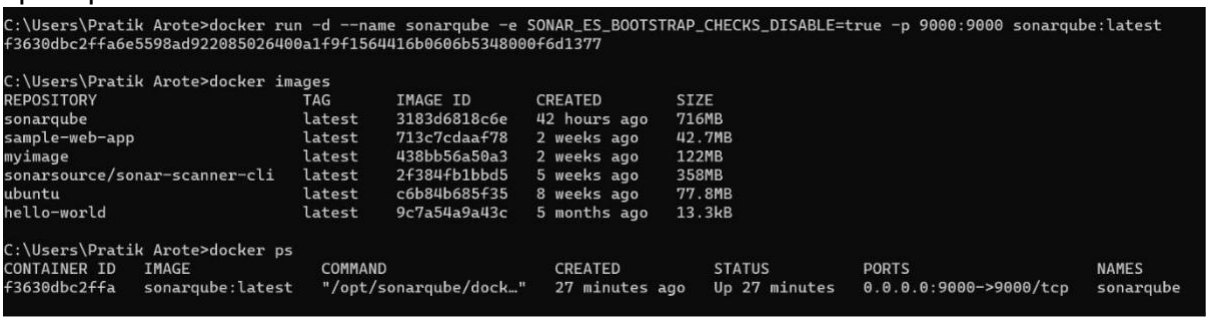
Integration: They can be integrated into CI/CD pipelines to automatically scan code before deployment.

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.

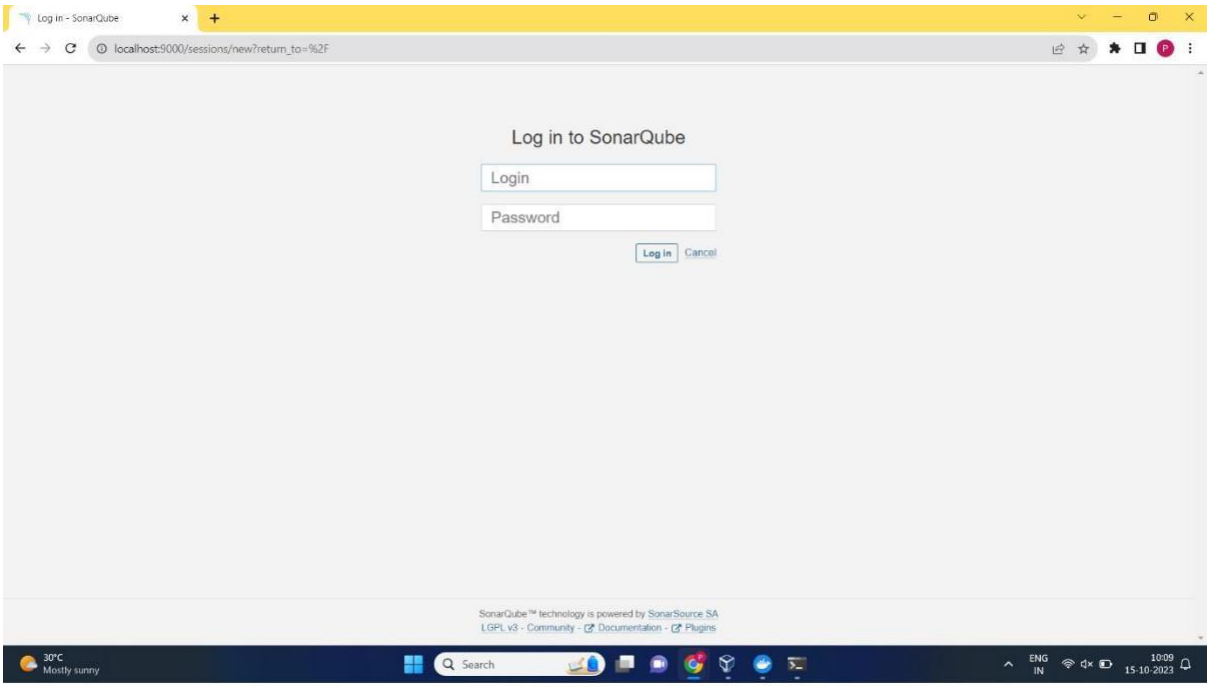
1. INSTALL sonarqube (docker images) and sonarscanner zip file from https://docs.sonarsource.com/sonarqube/latest/analyzing-sourcecode/scanners/sonarscanner/ and set up config file as given in docs.

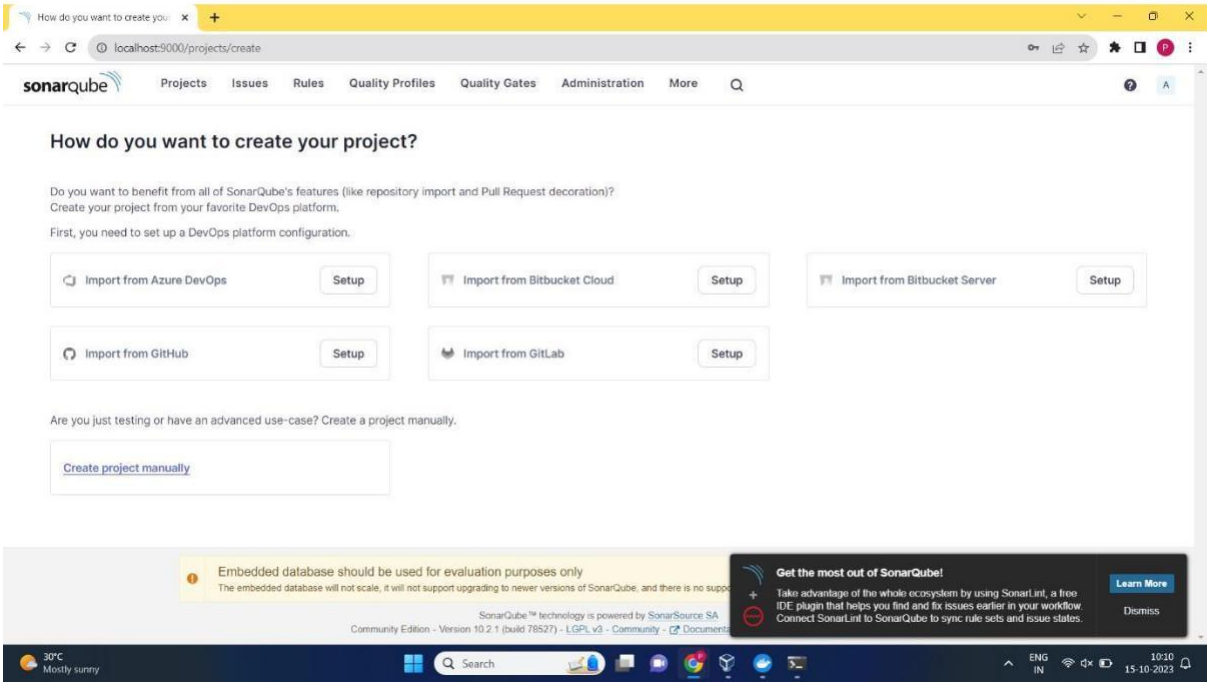


2. Spin up the container

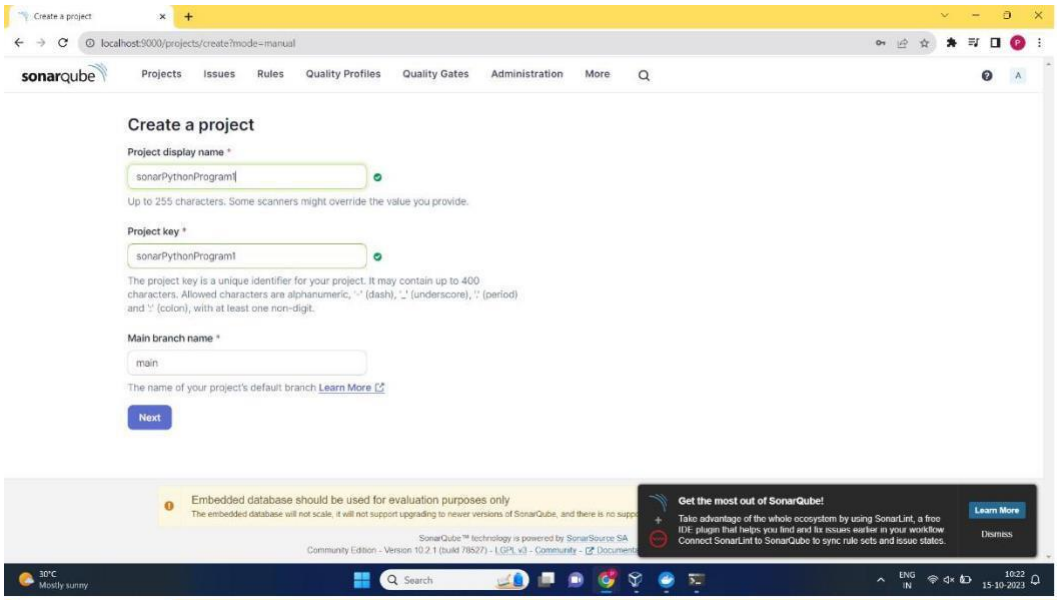


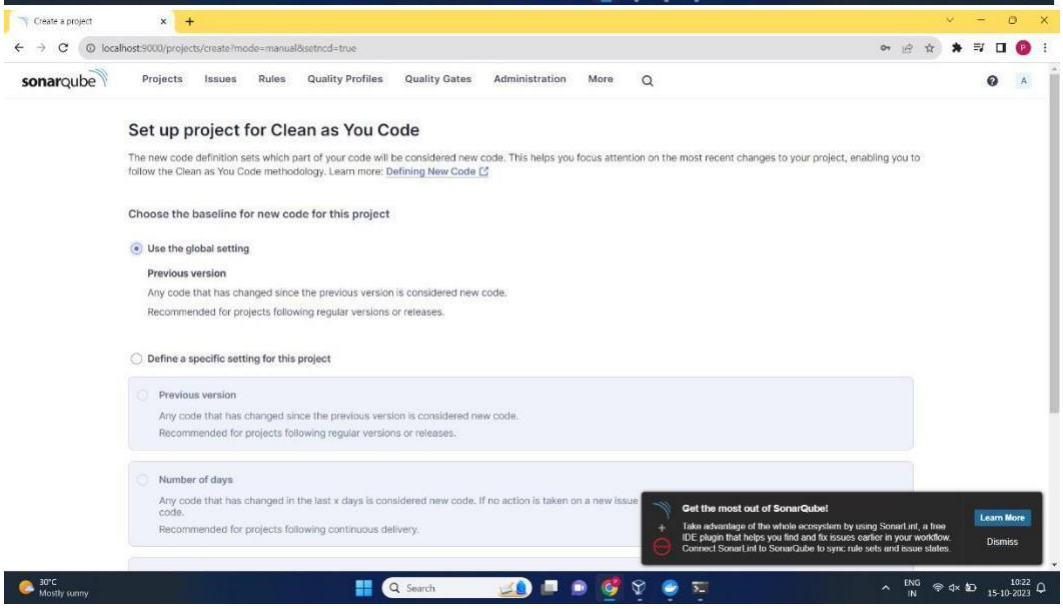
3. Open http://localhost:9000 on the browser. Enter login and password both as “admin” and then set up new password.

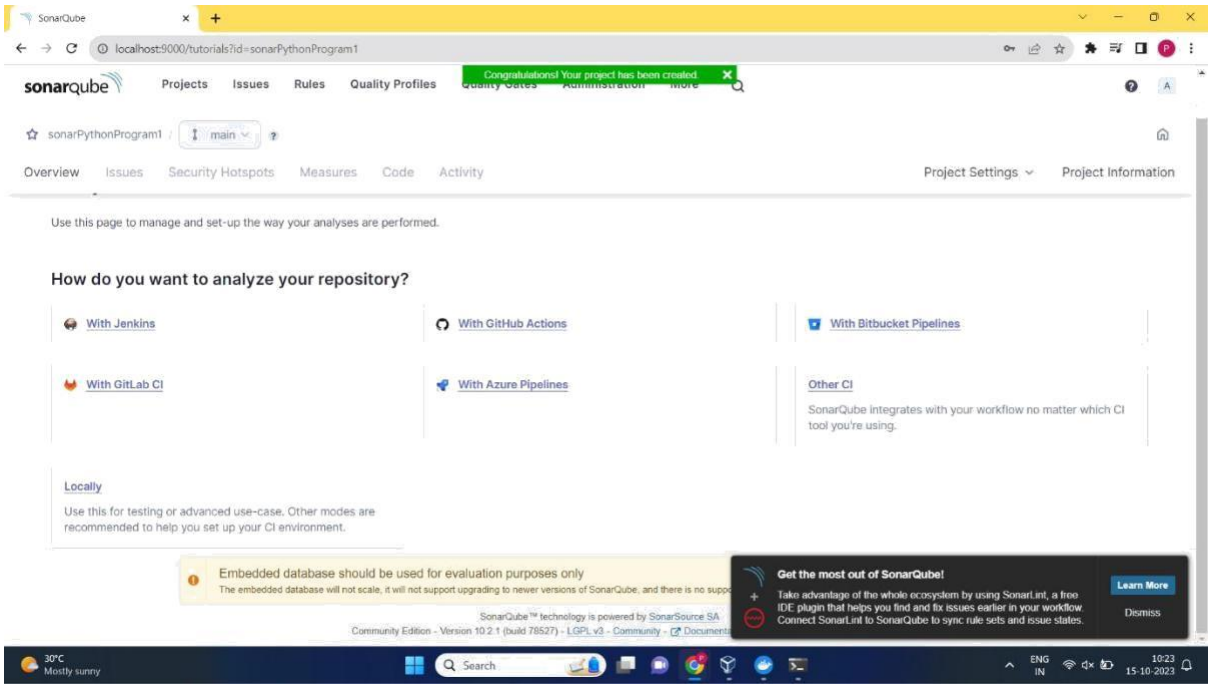




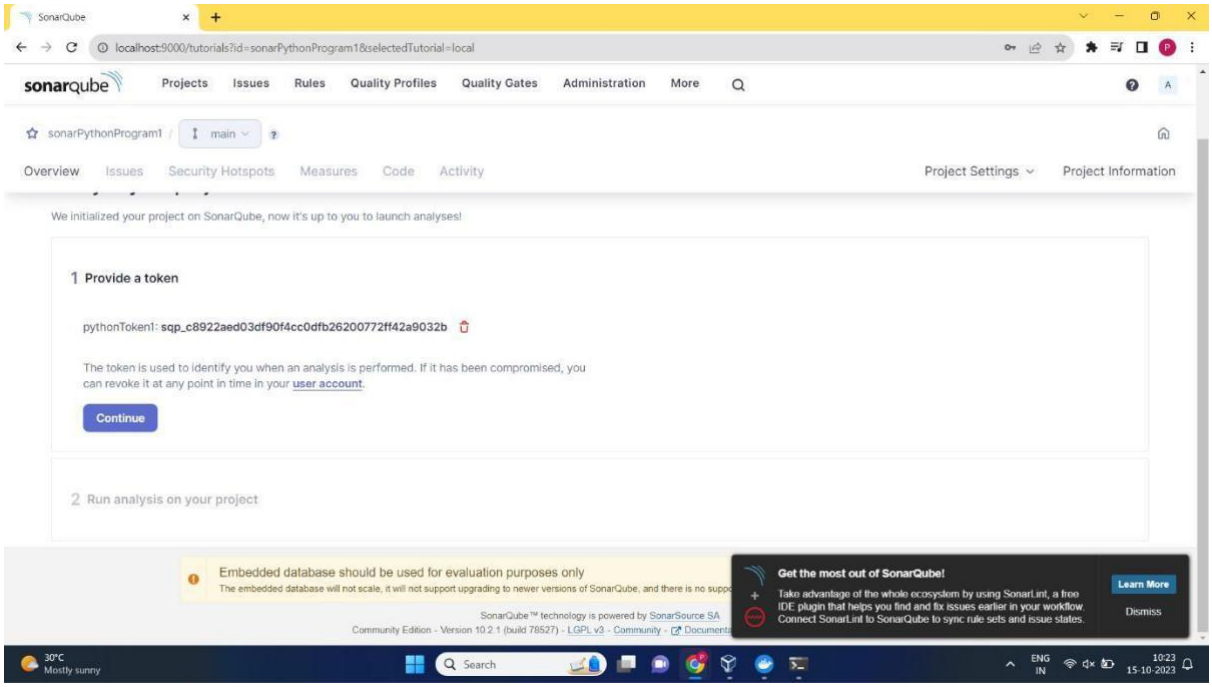
4. Create a project

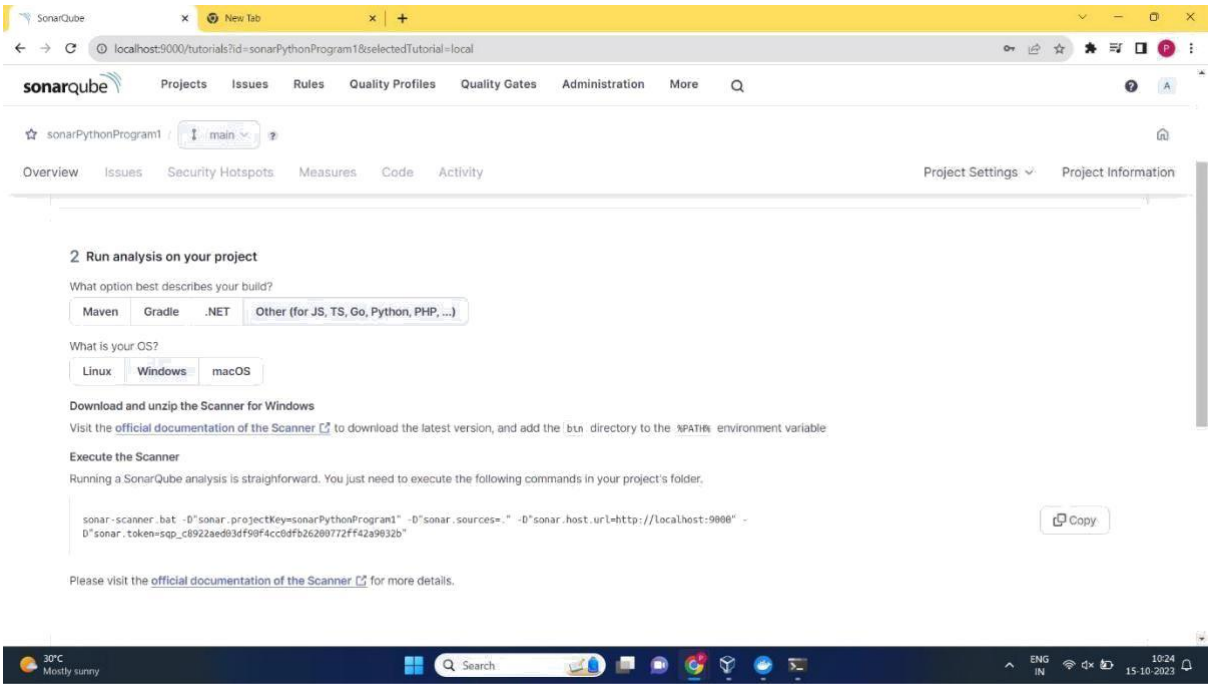

Create a project
localhost:9000/projects/create?mode=manual
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
Create a project
Project display name *
sonarPythonProgram1
Up to 255 characters. Some scanners might override the value you provide.
Project key *
sonarPythonProgram1
The project key is a unique identifier for your project. It may contain up to 400 characters. Allowed characters are alphanumeric, '-' (dash), '_' (underscore), '.' (period) and ':' (colon), with at least one non-digit.
Main branch name *
main
The name of your project's default branch Learn More
Next
Embedded database should be used for evaluation purposes only
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
10:22
15-10-2023


Create a project
localhost:9000/projects/create?mode=manual&setncd=true
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More
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
30°C
Mostly sunny
Search
ENG
IN
10:22
15-10-2023

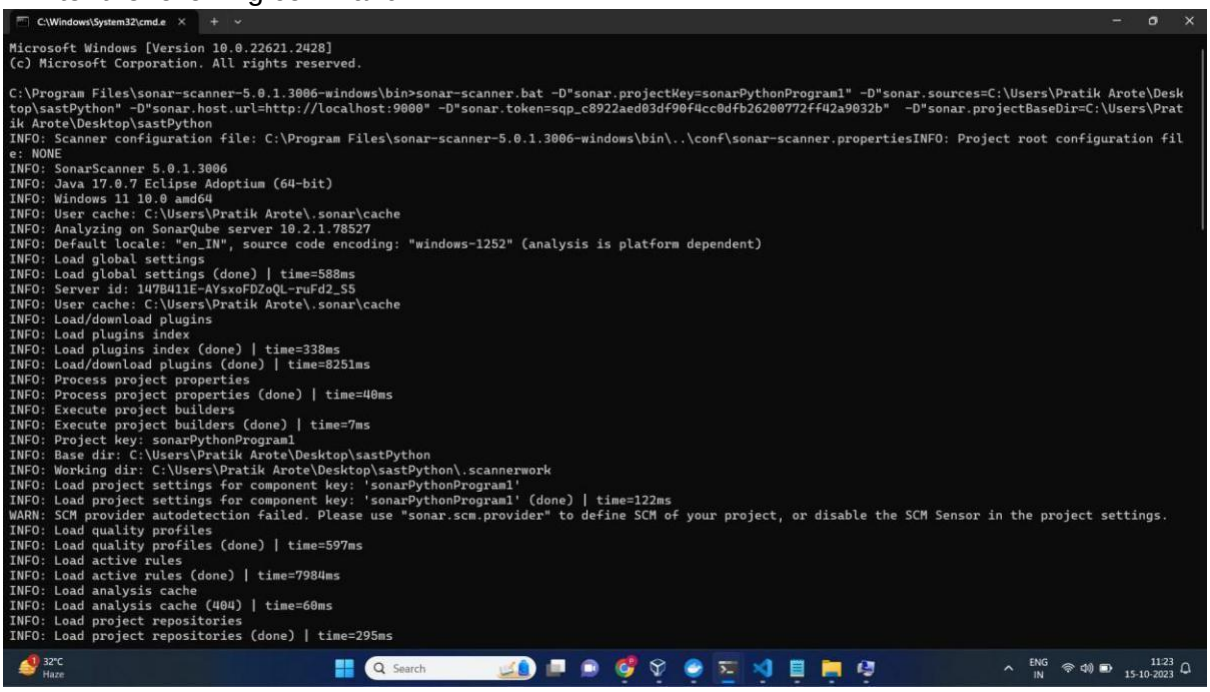


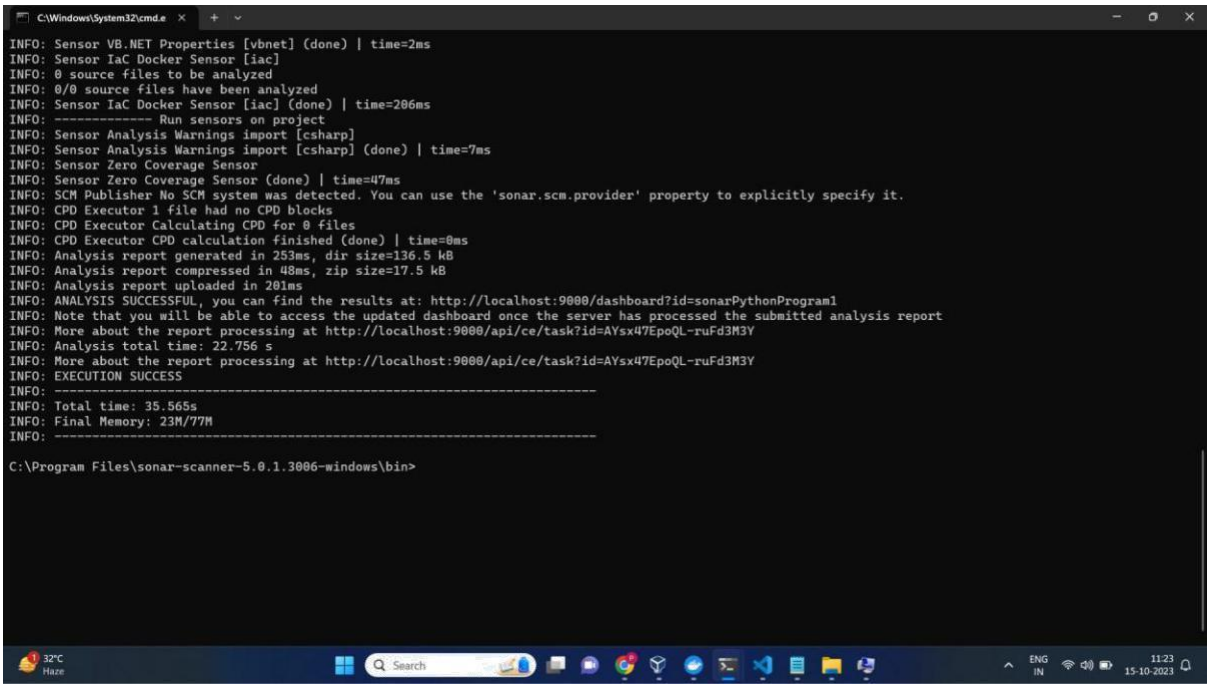
5. Provide token



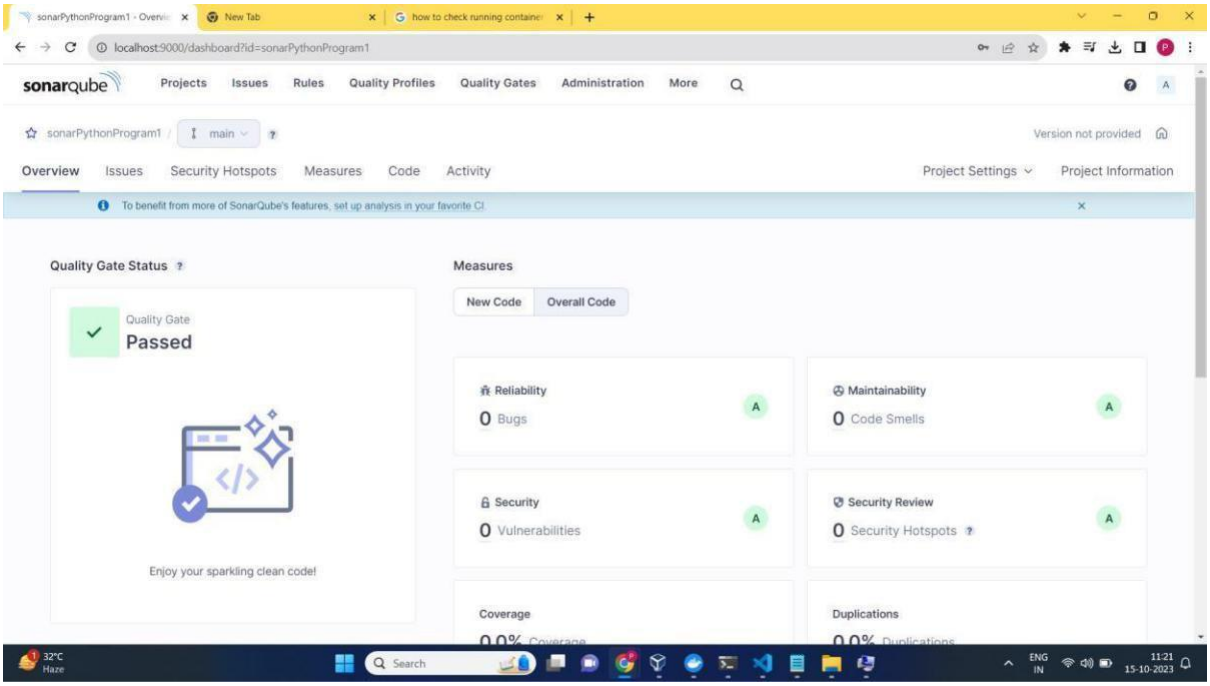


6. Enter the following command





7. See the result of the test



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.

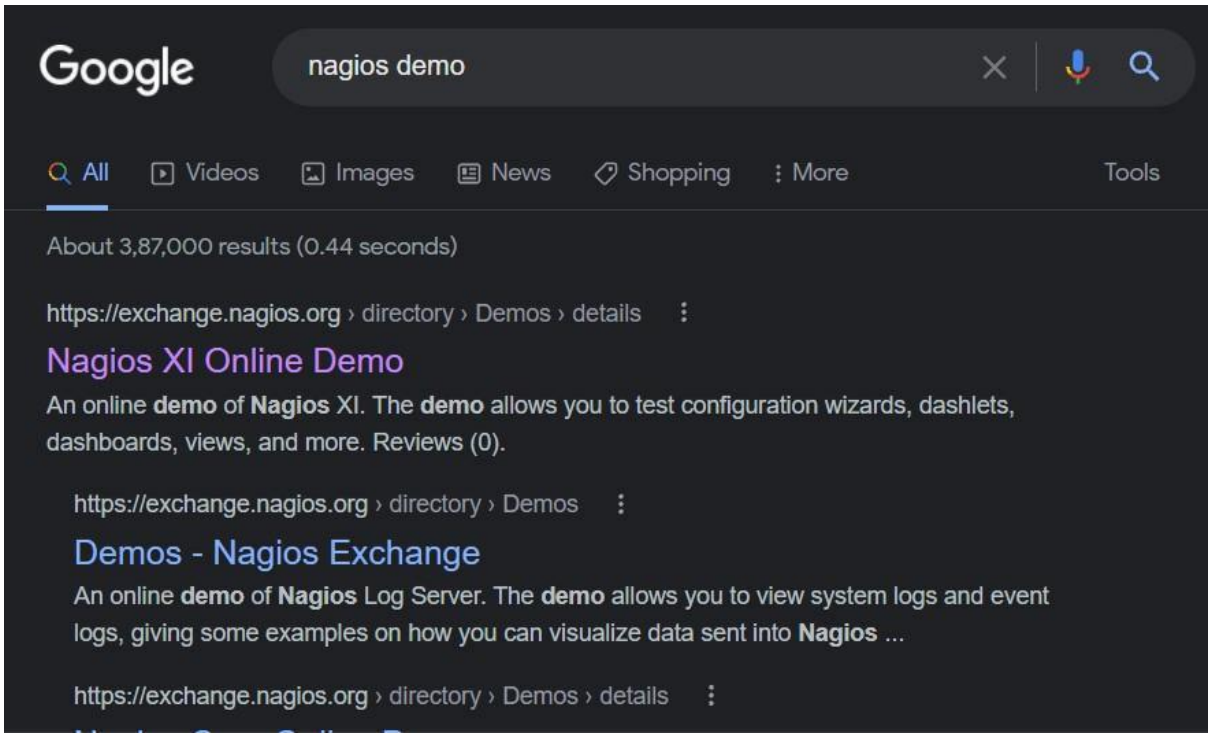
Theory:

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.

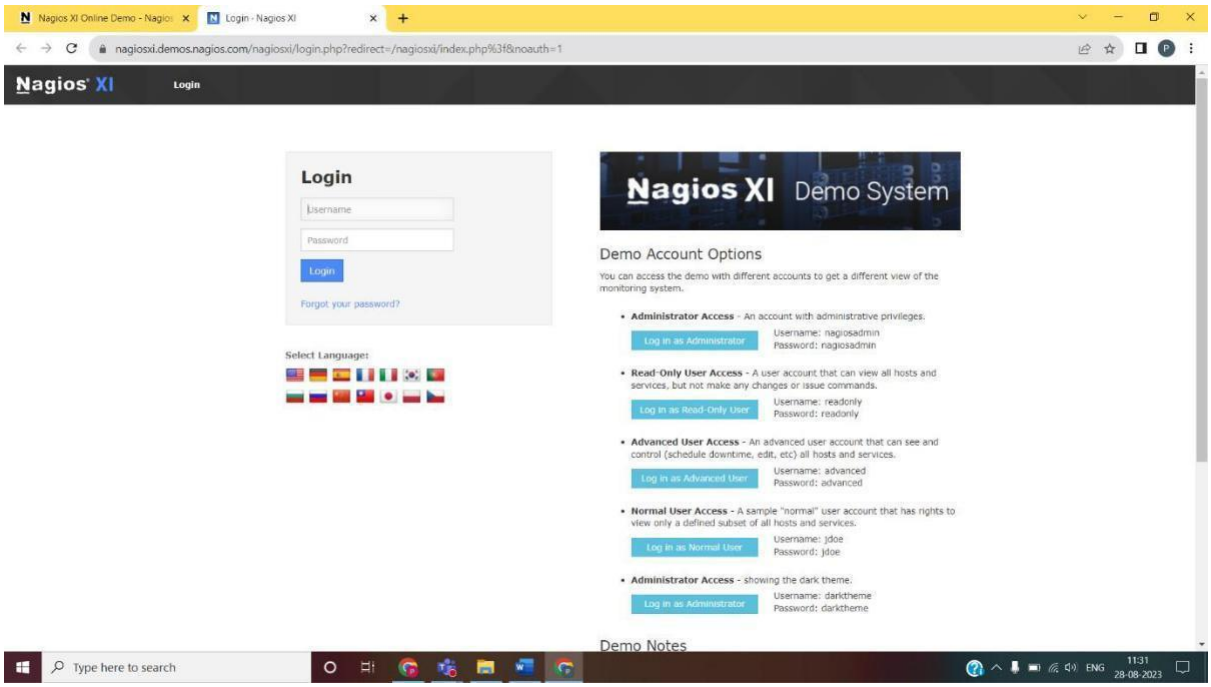
Steps-

Go to google.com, Search Nagios Demo

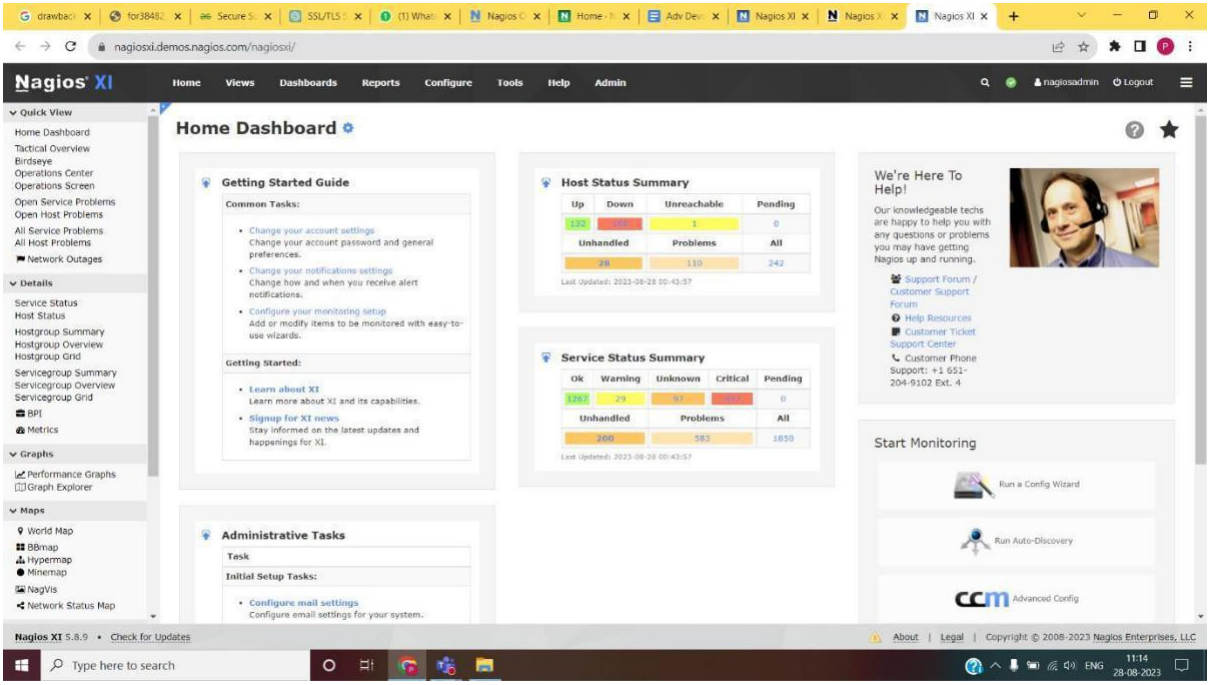


Now click on the website-

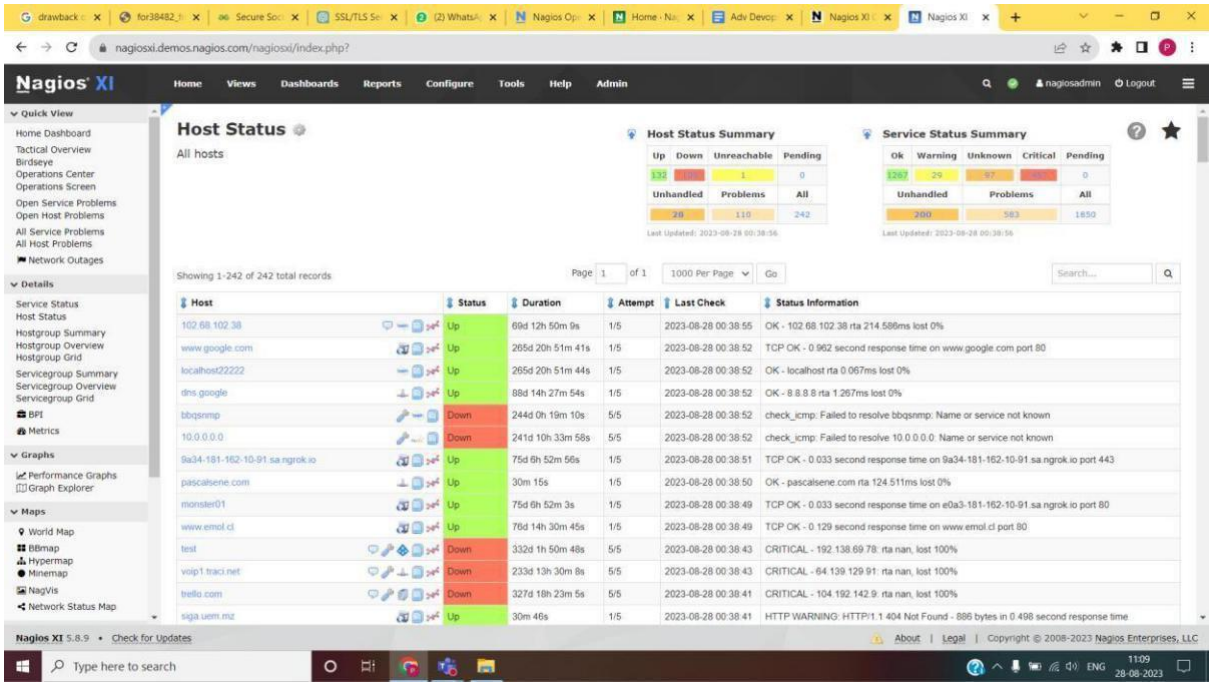
Now click on login as administrator



It will have interface like this

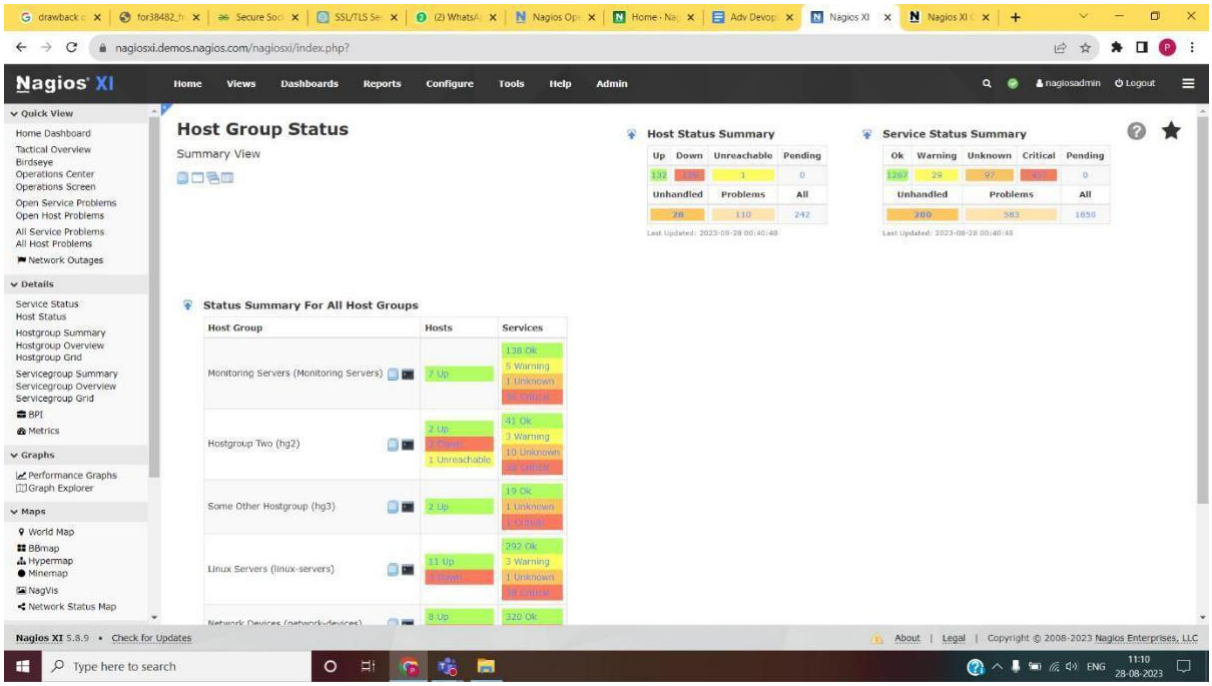


Now click on Host status-



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

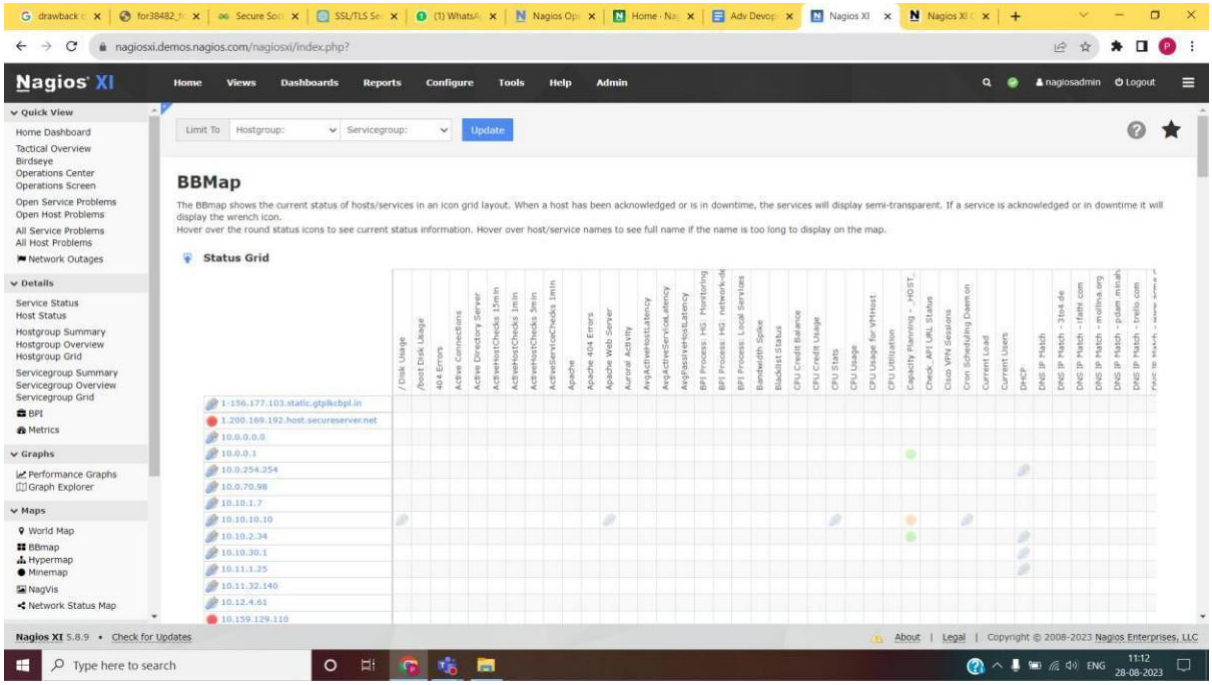
click on Host Group Status.



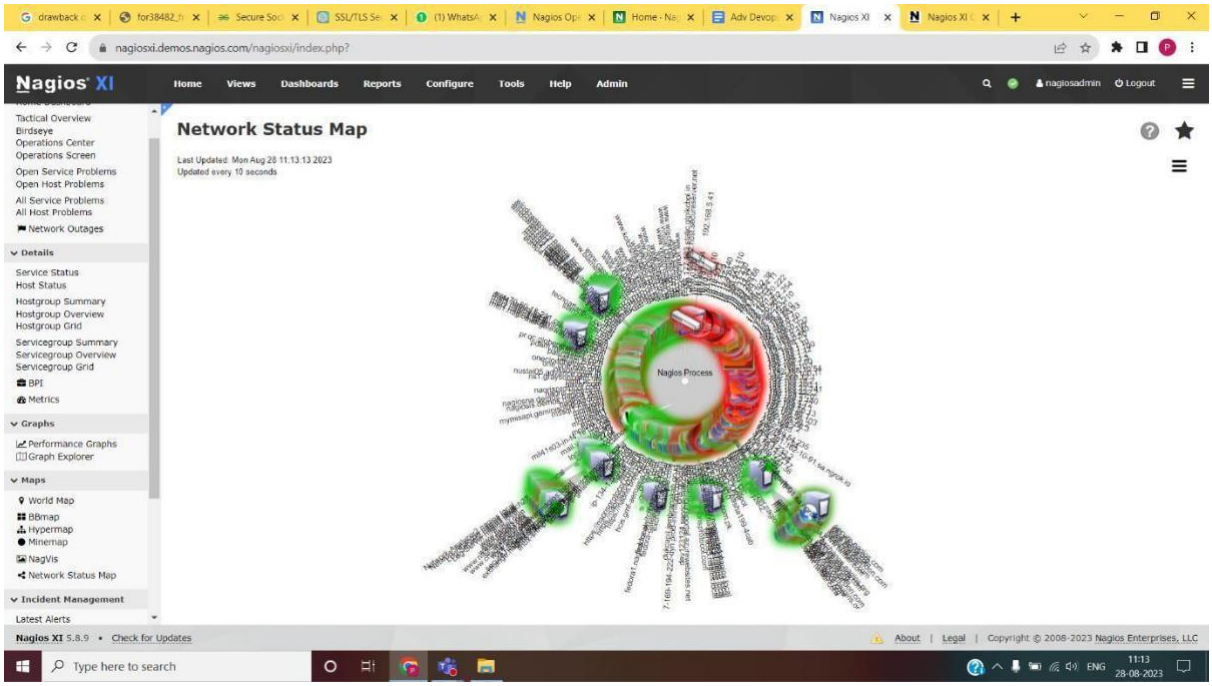
Here we can see Status Summary for All Host Groups

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.

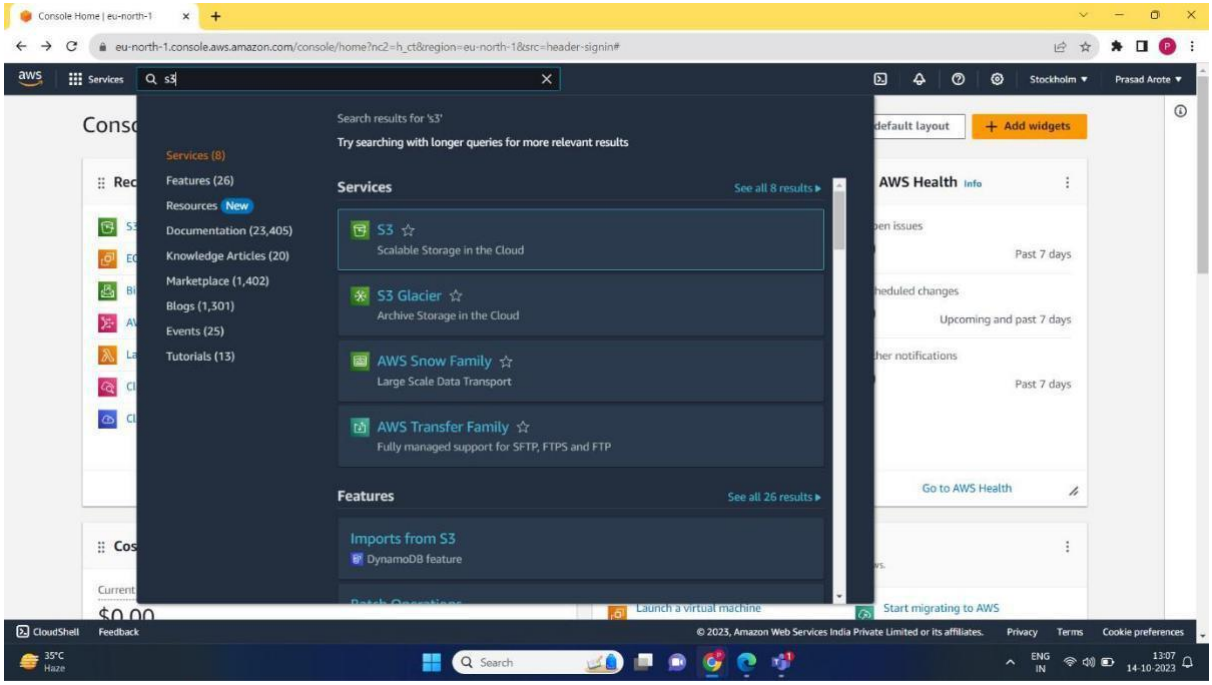
Theory:

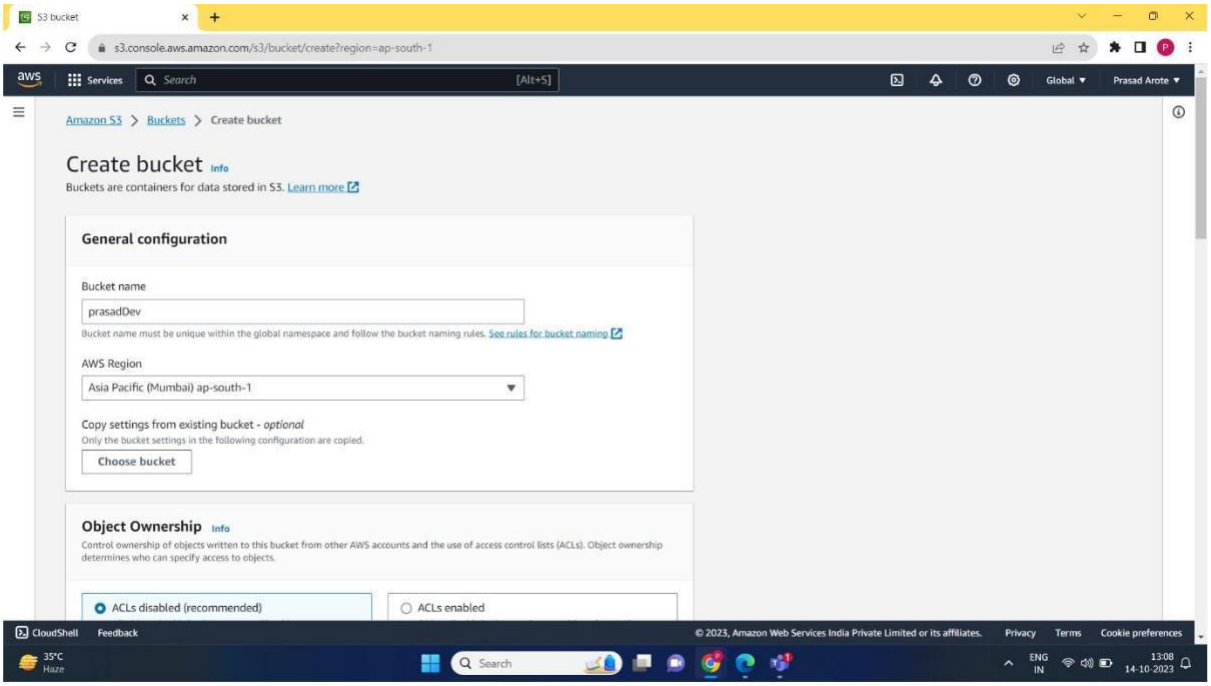
LAMBDA FUNCTION

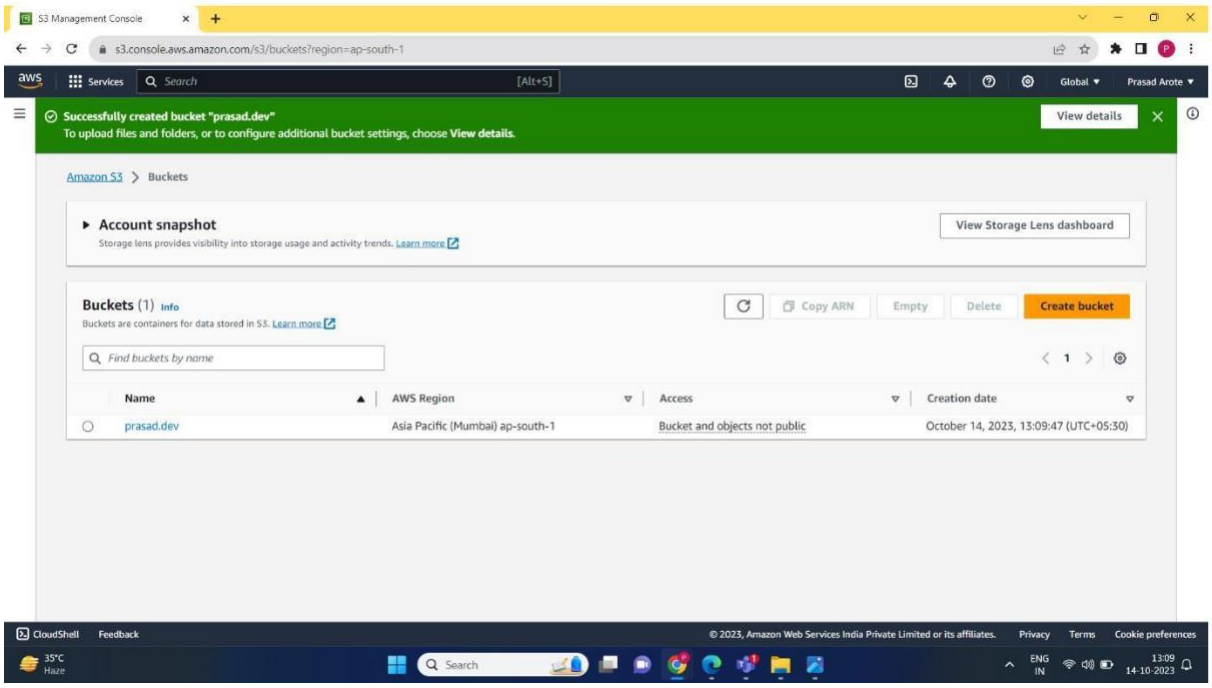
AWS Lambda is a serverless, event-driven compute service that lets you run code for virtually any type of application or backend service without provisioning or managing servers. You can trigger Lambda from over 200 AWS services and software as a service (SaaS) applications, and only pay for what you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

Installation:

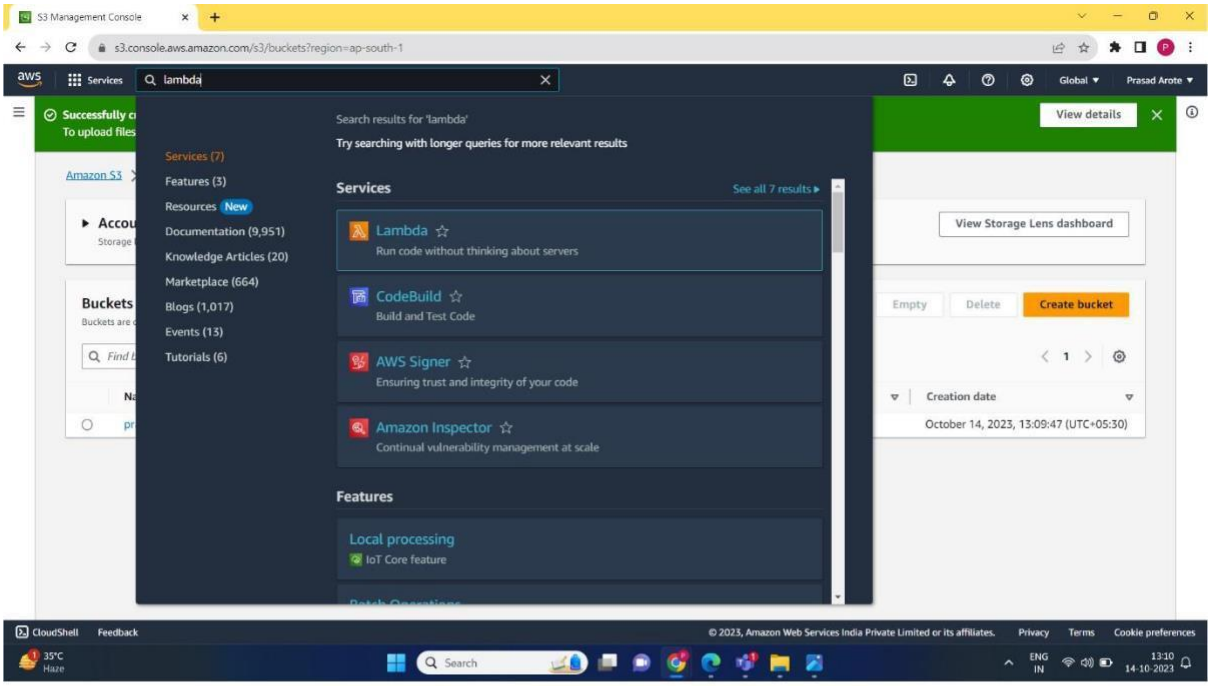
1. Create a S3 bucket

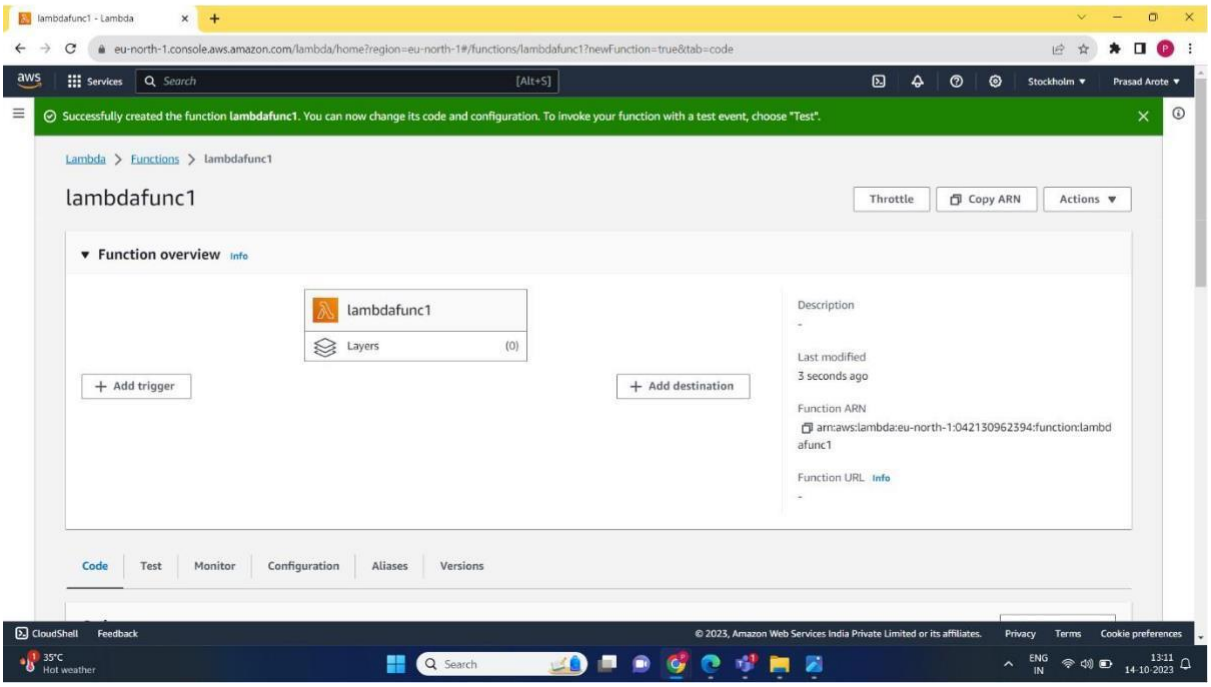


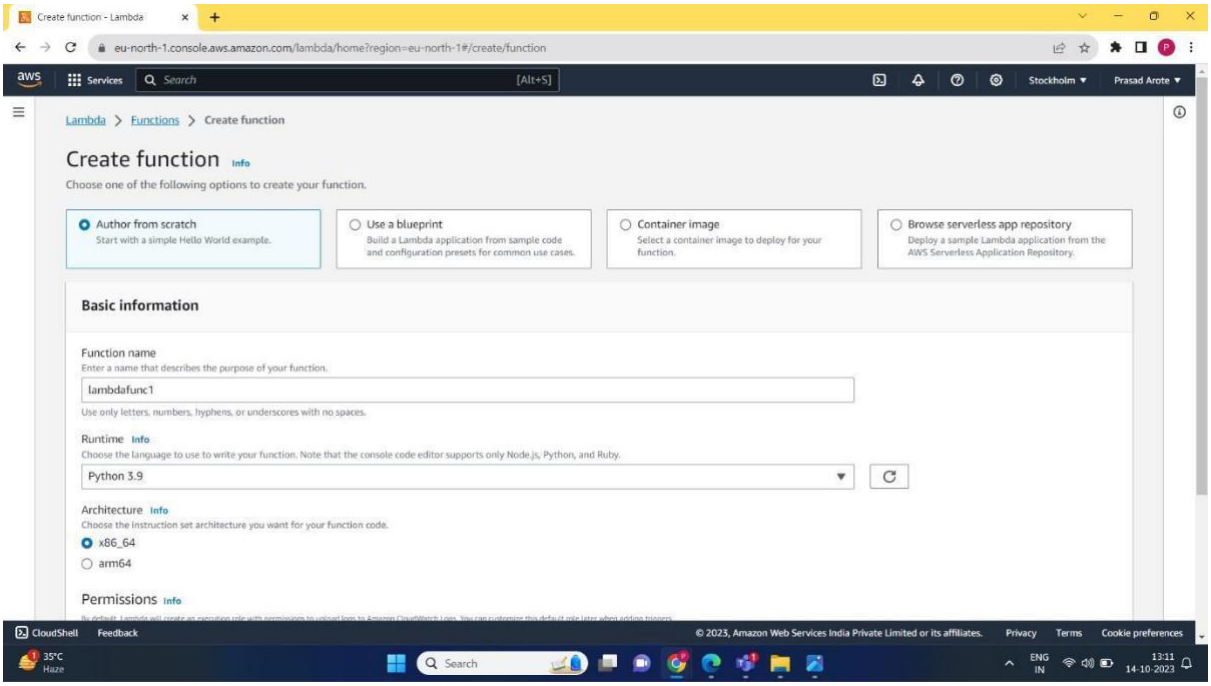




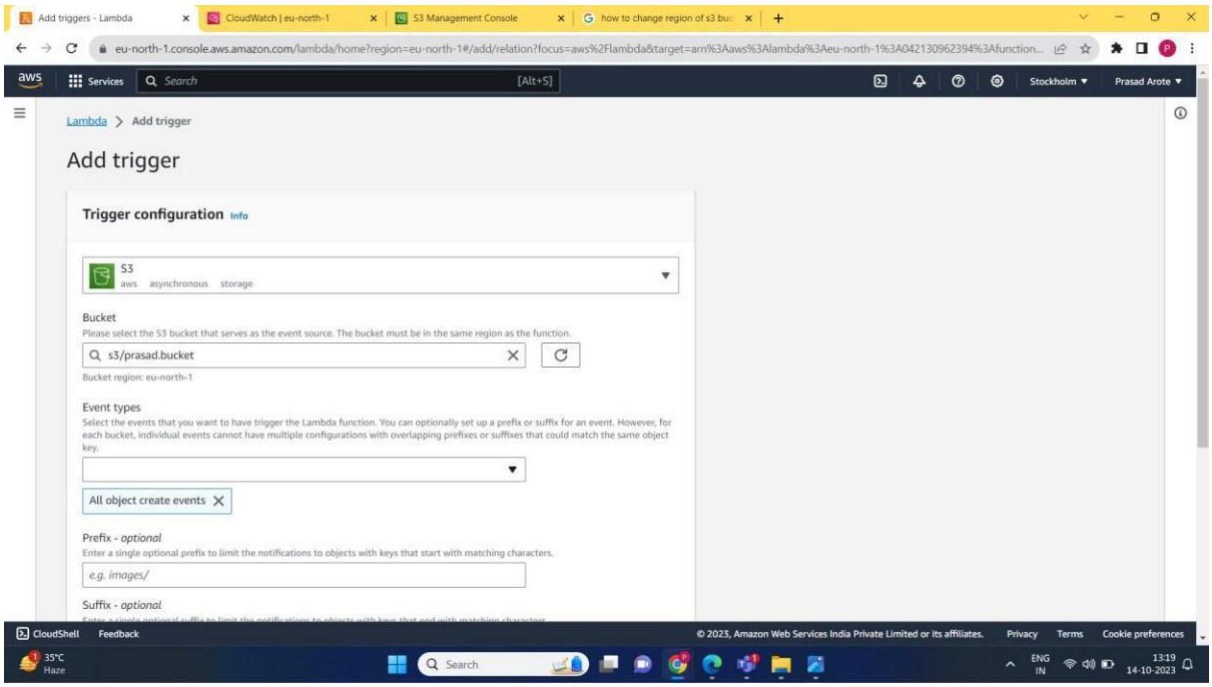
2. Create a Lambda function.

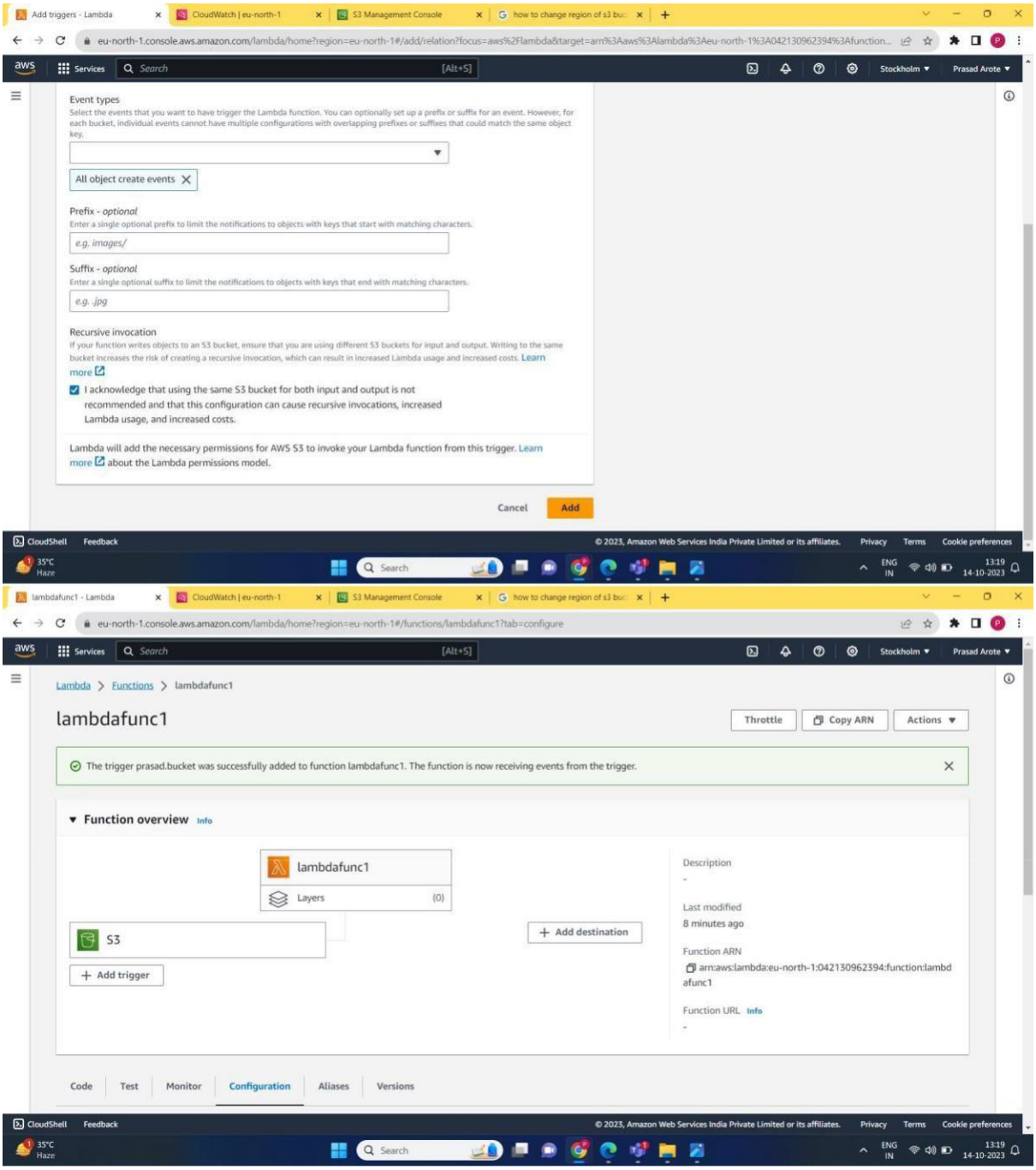

S3 Management Console
s3.console.aws.amazon.com/s3/buckets?region=ap-south-1
lambda
Search results for 'lambda'
Try searching with longer queries for more relevant results
Services (7)
Features (3)
Resources New
Documentation (9,951)
Knowledge Articles (20)
Marketplace (664)
Blogs (1,017)
Events (13)
Tutorials (6)
Services
Lambda
Run code without thinking about servers
CodeBuild
Build and Test Code
AWS Signer
Ensuring trust and integrity of your code
Amazon Inspector
Continual vulnerability management at scale
Features
Local processing
IoT Core feature
View details
View Storage Lens dashboard
Empty
Delete
Create bucket
Creation date
October 14, 2023, 13:09:47 (UTC+05:30)
CloudShell
Feedback
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences


lambdafunc1 - Lambda
eu-north-1.console.aws.amazon.com/lambda/home?region=eu-north-1#/functions/lambdafunc1?newFunction=true&tab=code
Successfully created the function lambdafunc1. You can now change its code and configuration. To invoke your function with a test event, choose "Test".
Lambda > Functions > lambdafunc1
lambdafunc1
Throttle
Copy ARN
Actions
Function overview Info
lambdafunc1
Layers (0)
+ Add trigger
+ Add destination
Description
Last modified
3 seconds ago
Function ARN
arn:aws:lambda:eu-north-1:042130962394:function:lambdafunc1
Function URL Info
Code
Test
Monitor
Configuration
Aliases
Versions
CloudShell
Feedback
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences

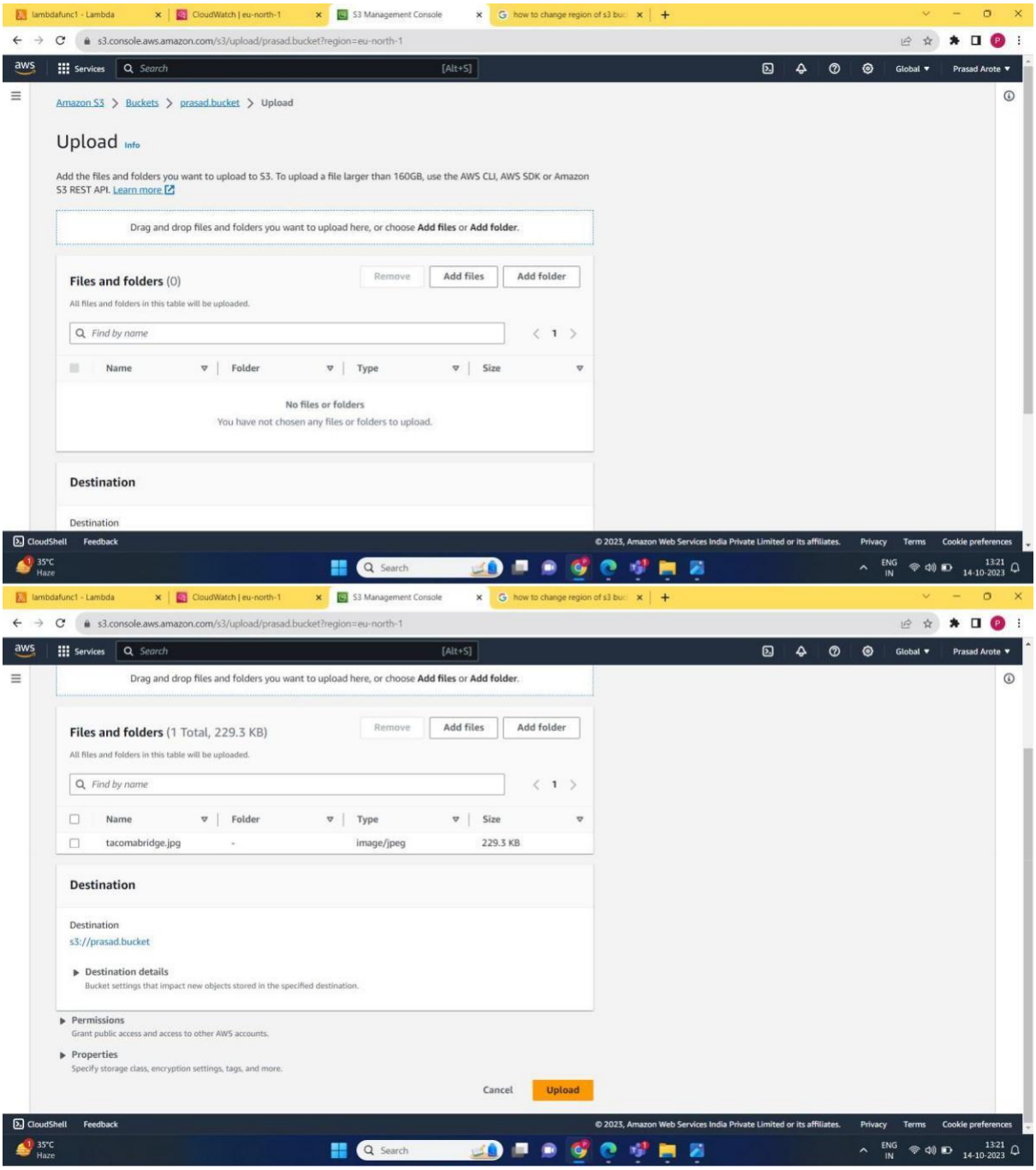


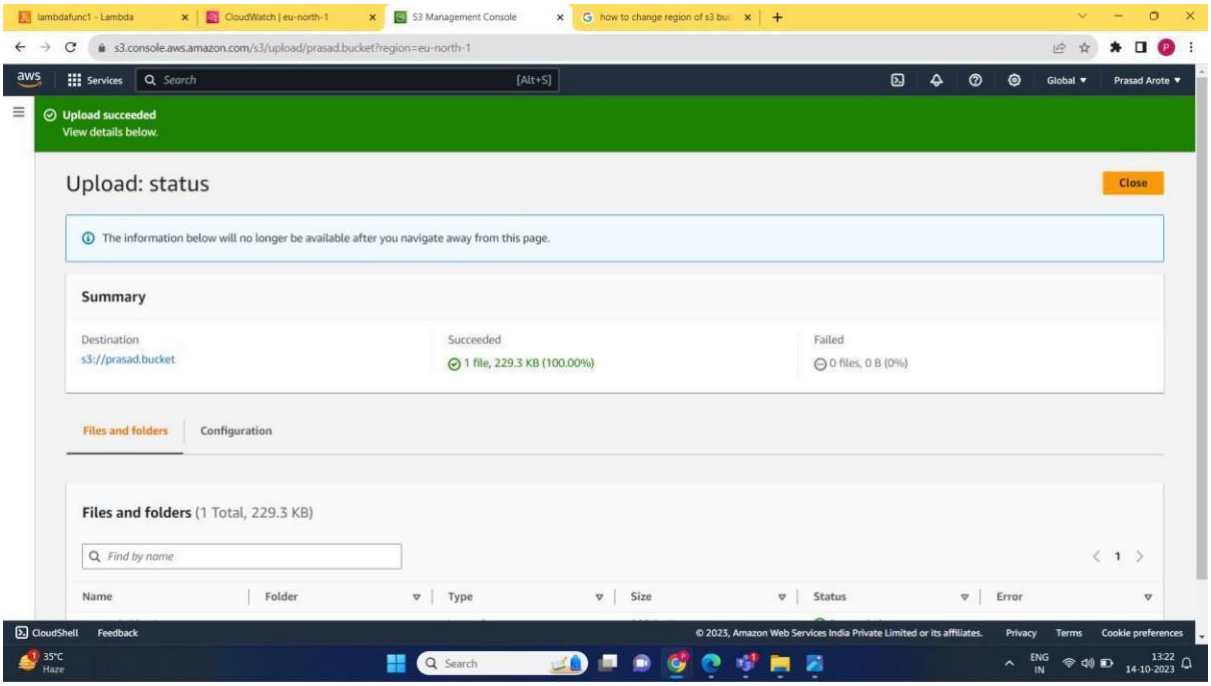
3. Create a trigger



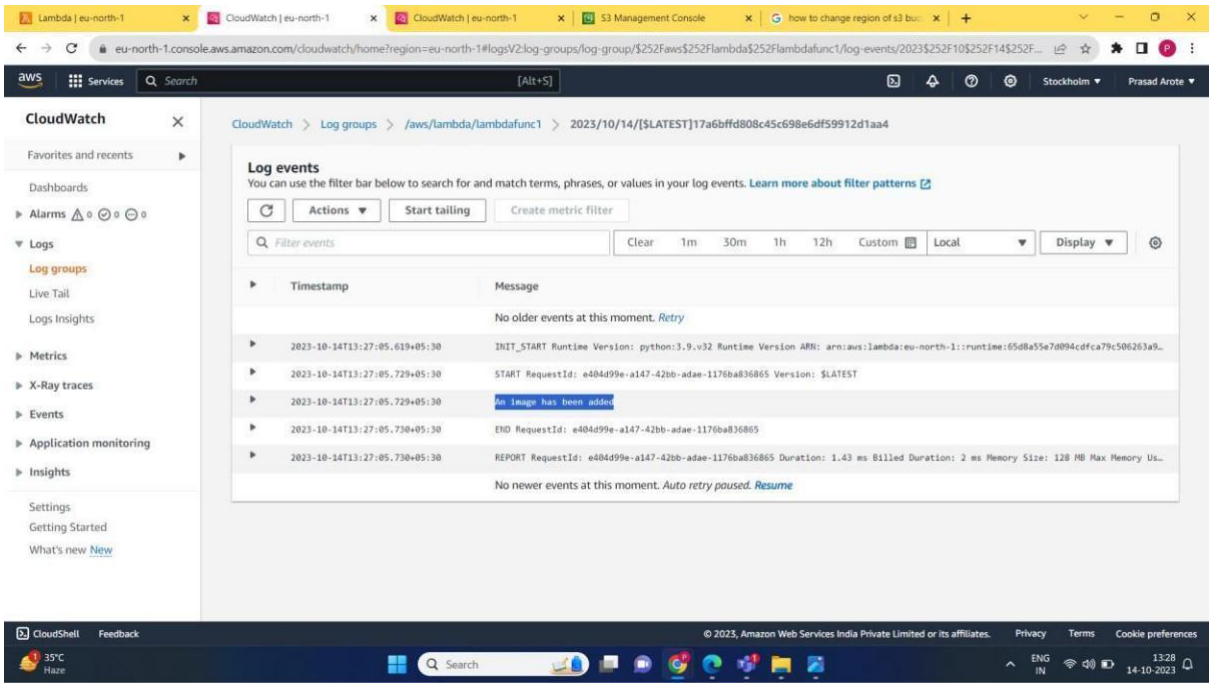

Add triggers - Lambda
CloudWatch | eu-north-1
S3 Management Console
eu-north-1.console.aws.amazon.com/lambda/home?region=eu-north-1#/add/relation?focus=aws%2Flambda&target=arn%3Aaws%3Alambda%3Aeu-north-1%3A042130962394%3Afunction...
Services
Search
[Alt+S]
Stockholm
Prasad Arote
Event types
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.
All object create events
Prefix - optional
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters.
e.g. images/
Suffix - optional
Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters.
e.g. .jpg
Recursive invocation
If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. Learn more
I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.
Lambda will add the necessary permissions for AWS S3 to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.
Cancel
Add
CloudShell
Feedback
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences
35°C
Haze
Search
ENG
IN
13:19
14-10-2023

lambdafunc1 - Lambda
CloudWatch | eu-north-1
S3 Management Console
eu-north-1.console.aws.amazon.com/lambda/home?region=eu-north-1#/functions/lambdafunc1?tab=configure
Services
Search
[Alt+S]
Stockholm
Prasad Arote
Lambda > Functions > lambdafunc1
lambdafunc1
Throttle
Copy ARN
Actions
The trigger prasad.bucket was successfully added to function lambdafunc1. The function is now receiving events from the trigger.
Function overview Info
lambdafunc1
Layers (0)
S3
Add trigger
Add destination
Description
-
Last modified
8 minutes ago
Function ARN
arn:aws:lambda:eu-north-1:042130962394:function:lambdafunc1
Function URL Info
-
Code
Test
Monitor
Configuration
Aliases
Versions
CloudShell
Feedback
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences
35°C
Haze
Search
ENG
IN
13:19
14-10-2023


lambdafunc1 - Lambda
CloudWatch | eu-north-1
S3 Management Console
how to change region of s3 buc
s3.console.aws.amazon.com/s3/upload/prasad.bucket?region=eu-north-1
aws
Services
Search
[Alt+S]
Global
Prasad Arote
Amazon S3 > Buckets > prasad.bucket > Upload
Upload Info
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)
Remove
Add files
Add folder
All files and folders in this table will be uploaded.
Find by name
1
Name
Folder
Type
Size
No files or folders
You have not chosen any files or folders to upload.
Destination
Destination
CloudShell
Feedback
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences
35°C
Haze
Search
ENG
IN
13:21
14-10-2023
lambdafunc1 - Lambda
CloudWatch | eu-north-1
S3 Management Console
how to change region of s3 buc
s3.console.aws.amazon.com/s3/upload/prasad.bucket?region=eu-north-1
aws
Services
Search
[Alt+S]
Global
Prasad Arote
Drag and drop files and folders you want to upload here, or choose Add files or Add folder.
Files and folders (1 Total, 229.3 KB)
Remove
Add files
Add folder
All files and folders in this table will be uploaded.
Find by name
1
Name
Folder
Type
Size
tacomabridge.jpg
-
image/jpeg
229.3 KB
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
CloudShell
Feedback
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences
35°C
Haze
Search
ENG
IN
13:21
14-10-2023



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.

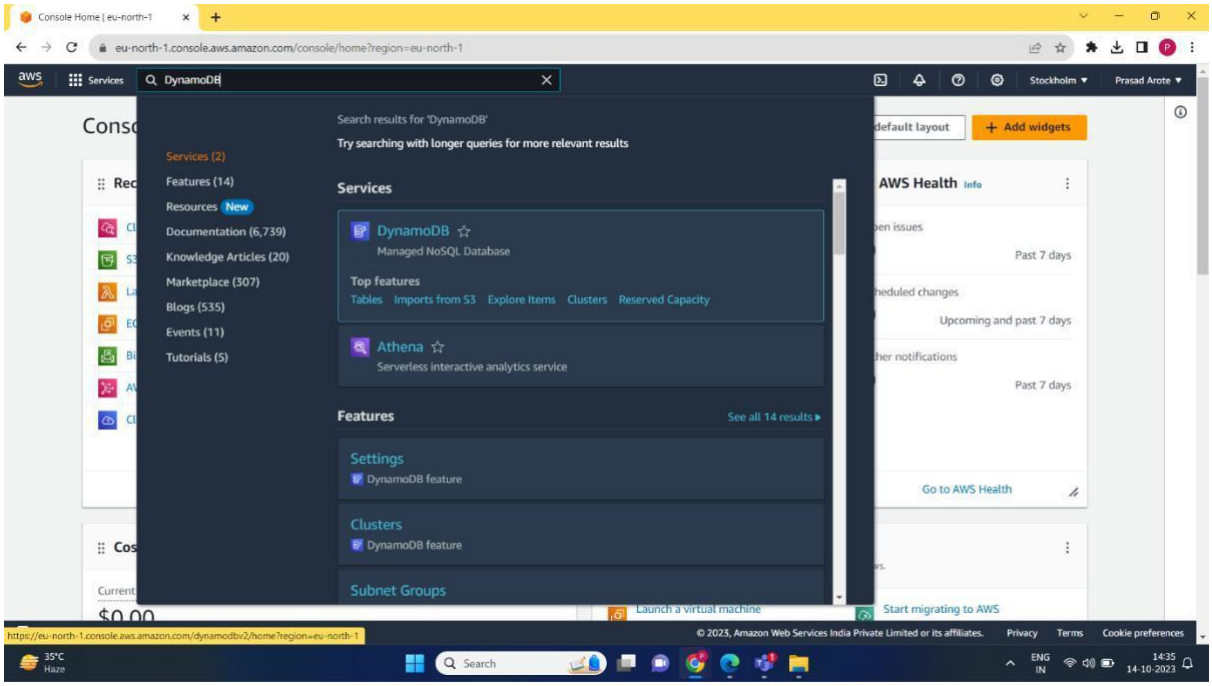
LO6: To engineer a composition of nano services using AWS Lambda and Step Functions with the serverless framework.

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.

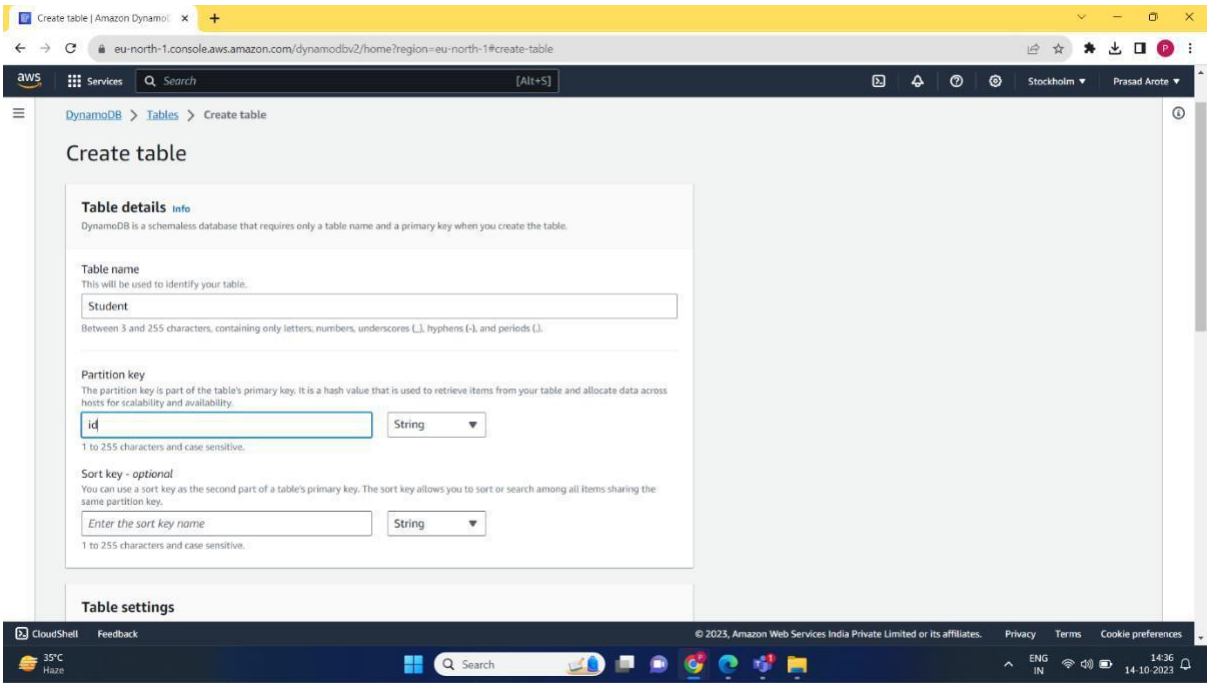
With DynamoDB, you can create database tables that can store and retrieve any amount of data and serve any level of request traffic. You can scale up or scale down your tables' throughput capacity without downtime or performance degradation. You can use the AWS Management Console to monitor resource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



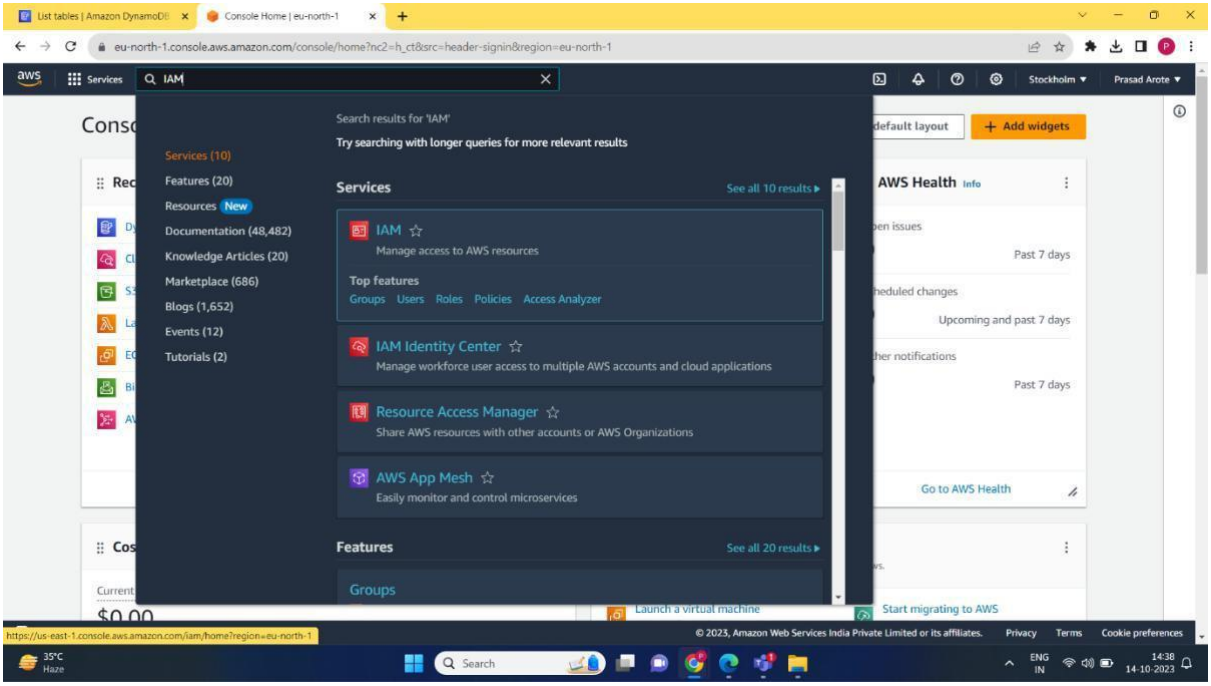
DynamoDB provides on-demand backup capability. It allows you to create full backups of your tables for long-term retention and archival for regulatory compliance needs.

STEPS:

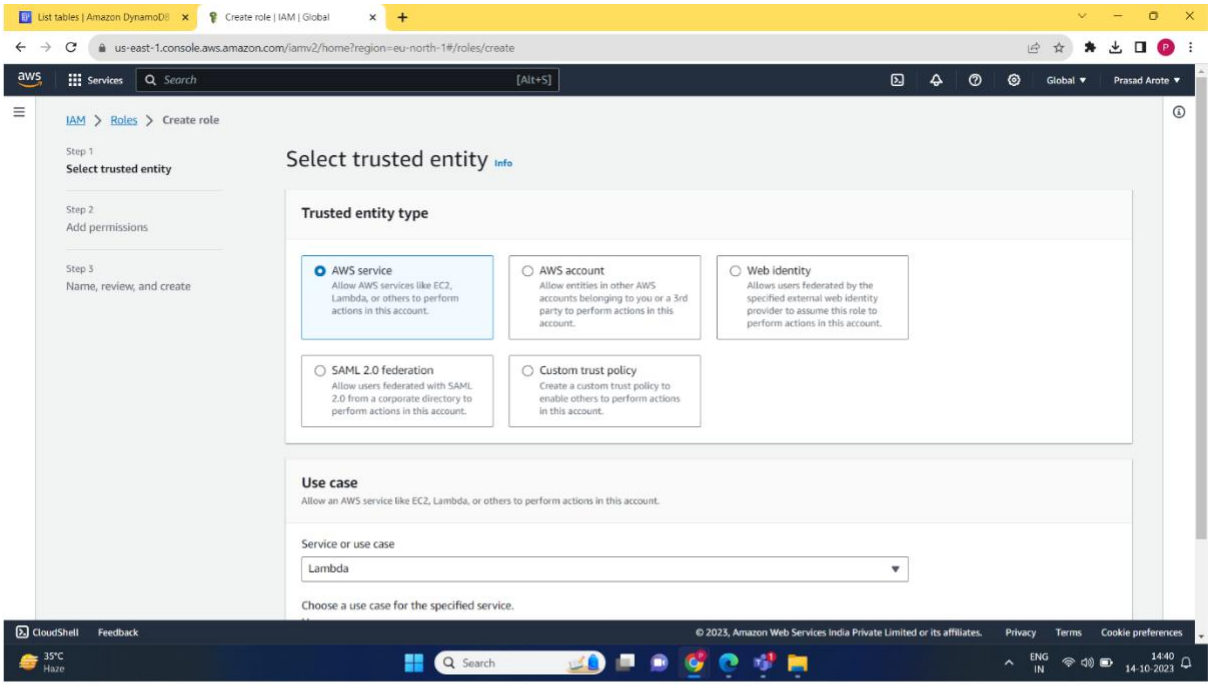
1. Create a table

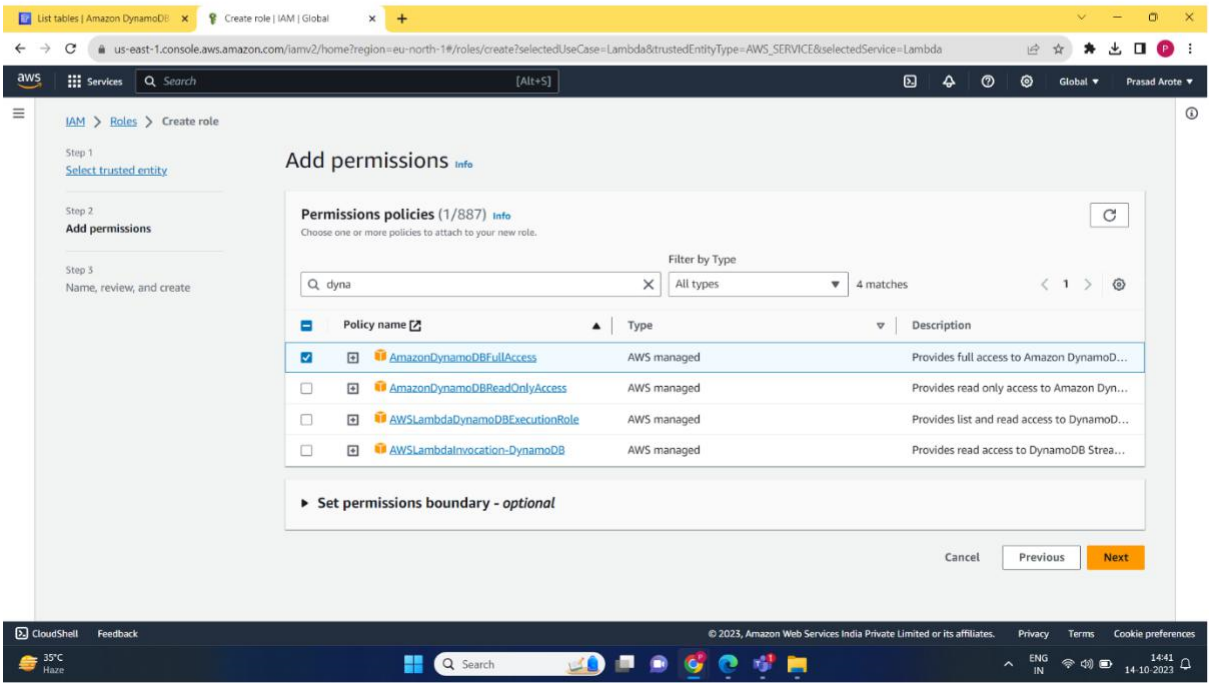


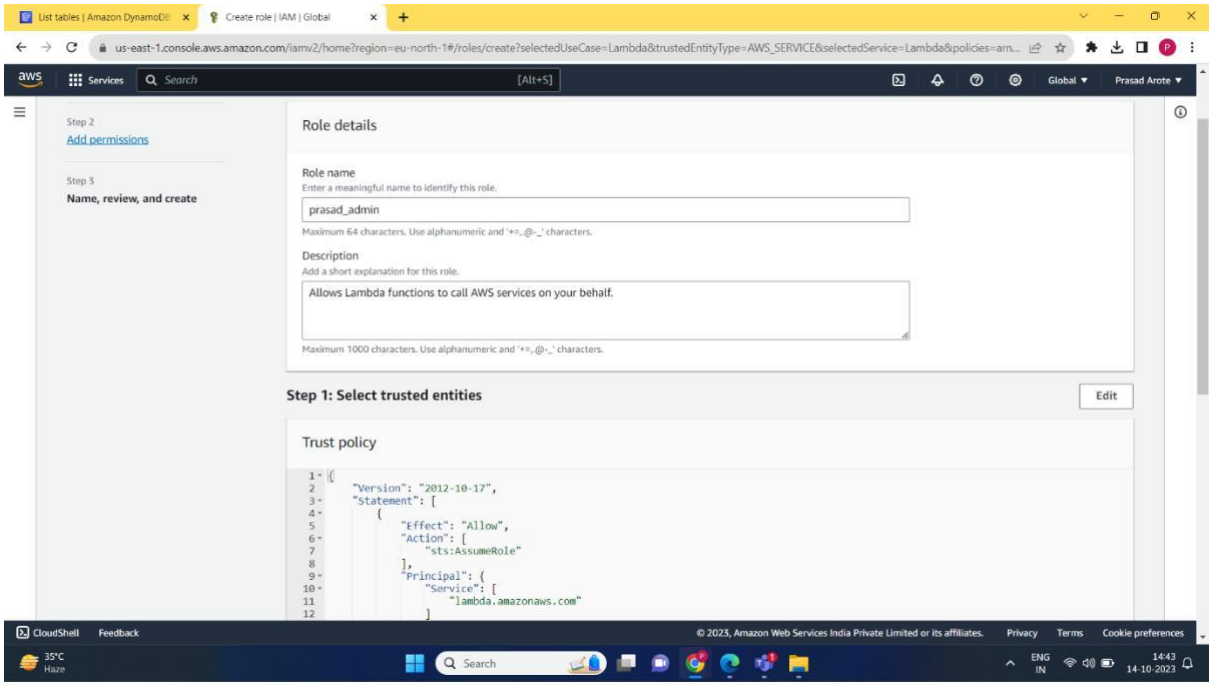
2. Create a role using IAM

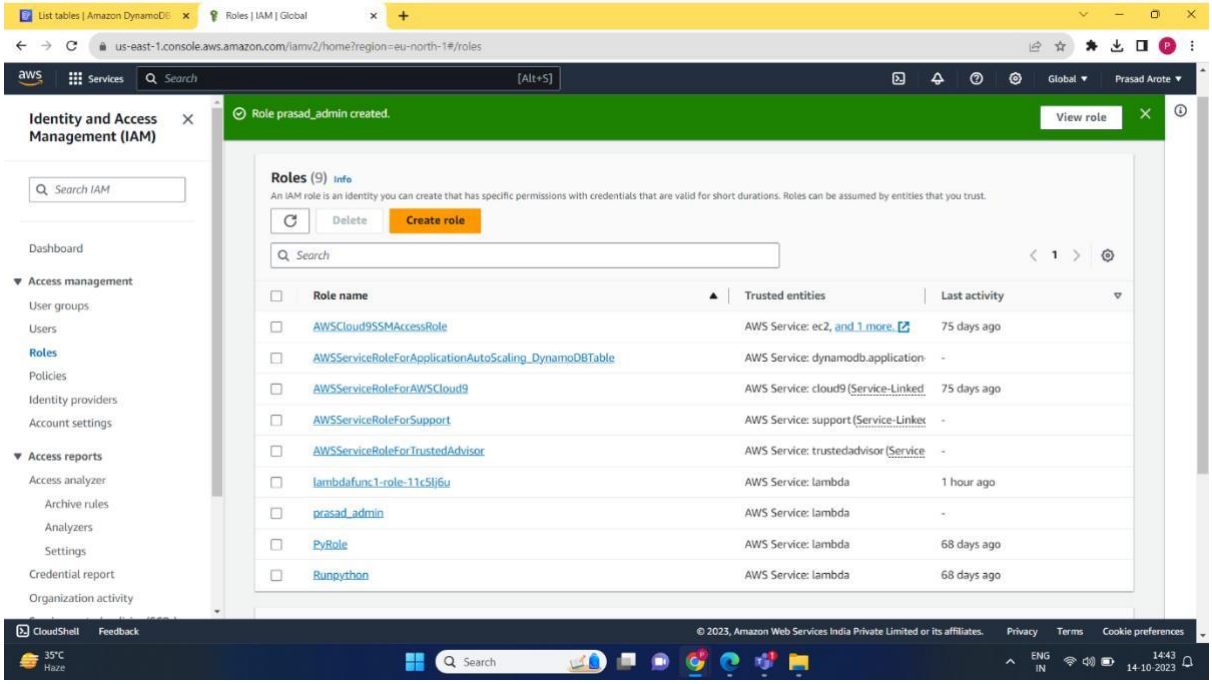


3. Add permissions – AmazonDynamoFullAccess

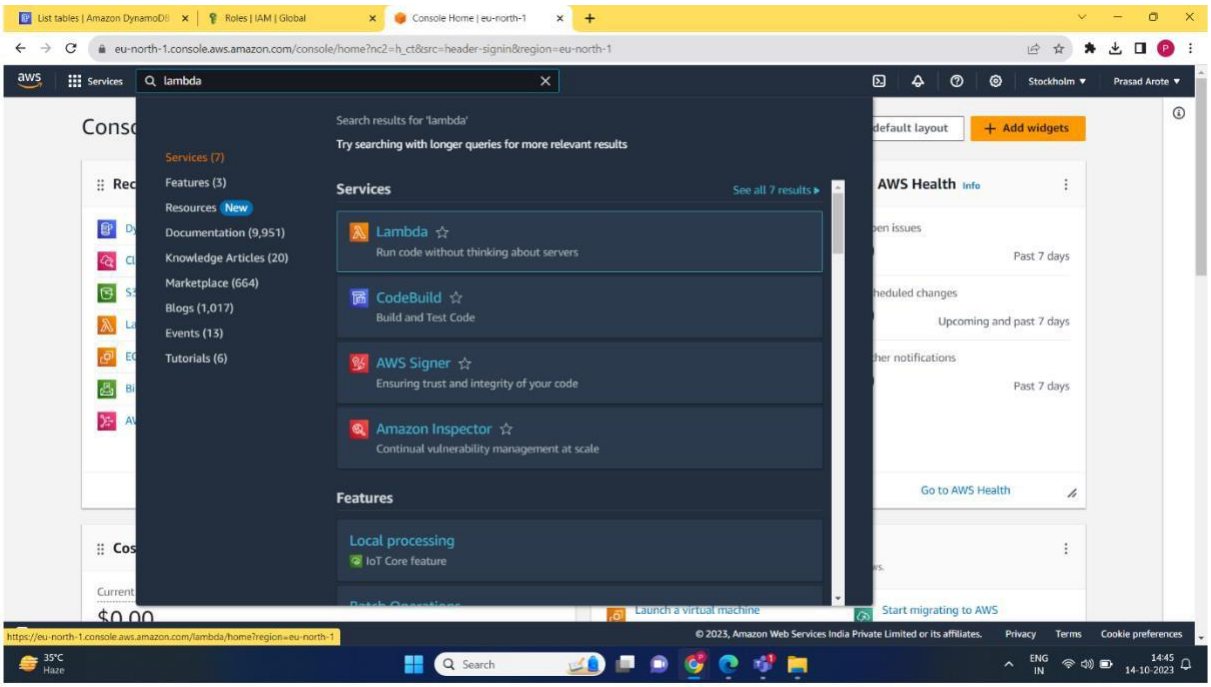


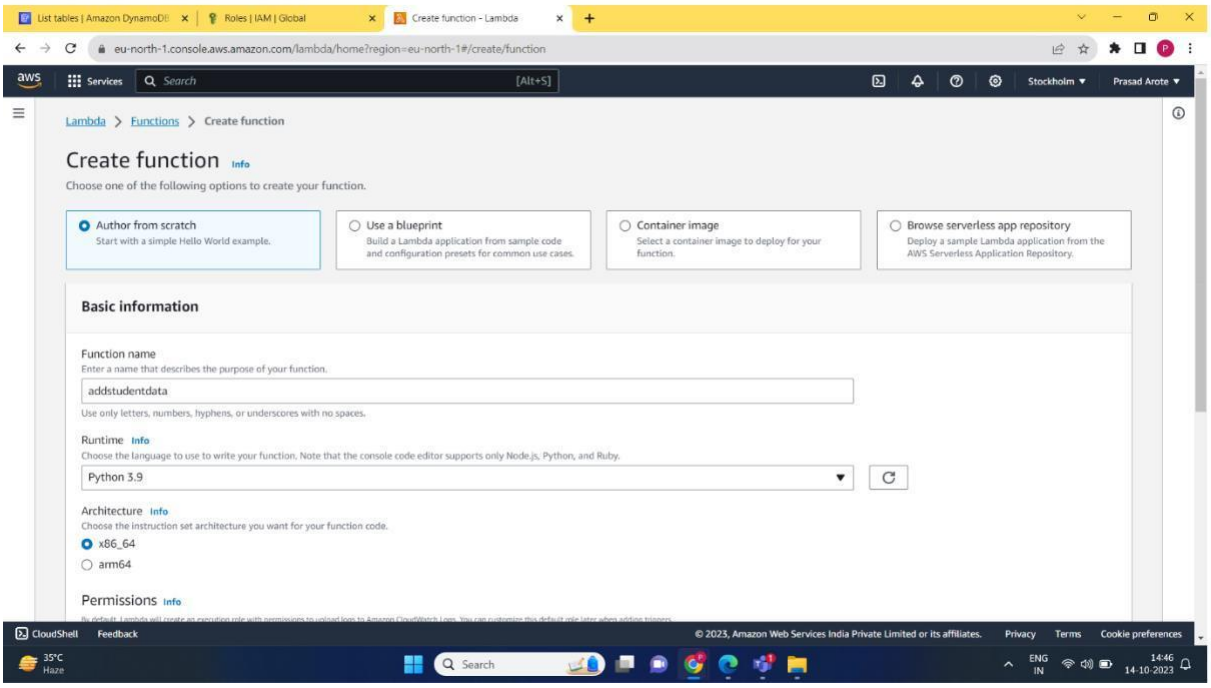


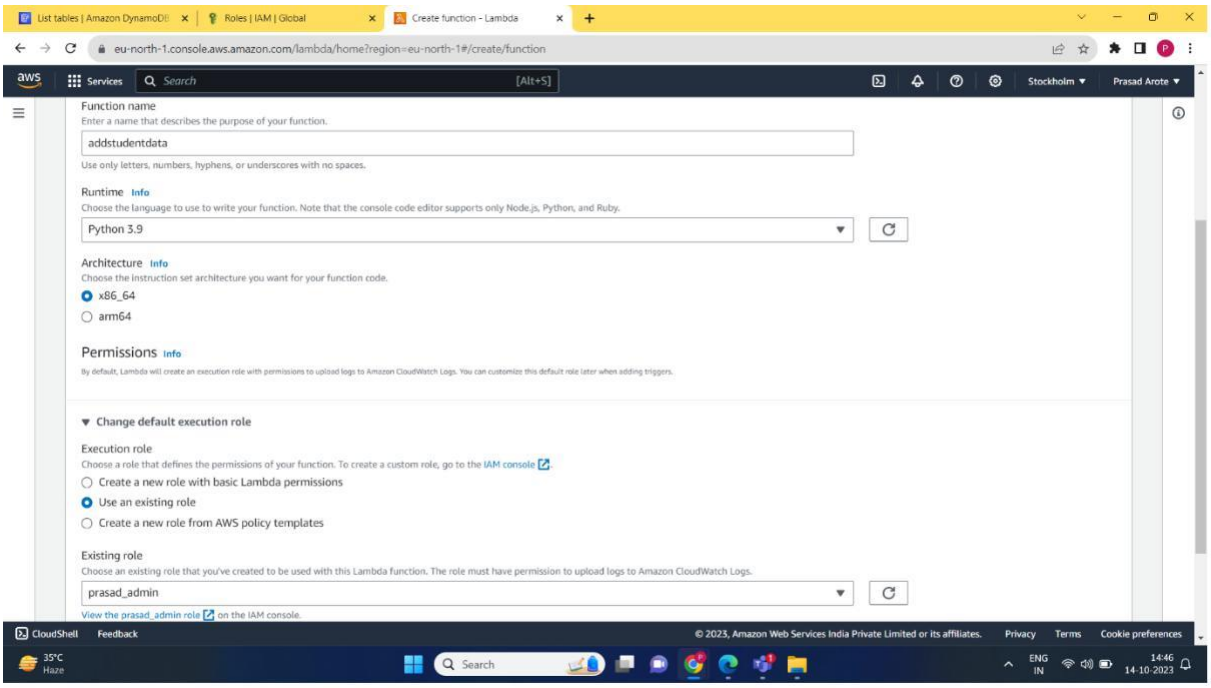


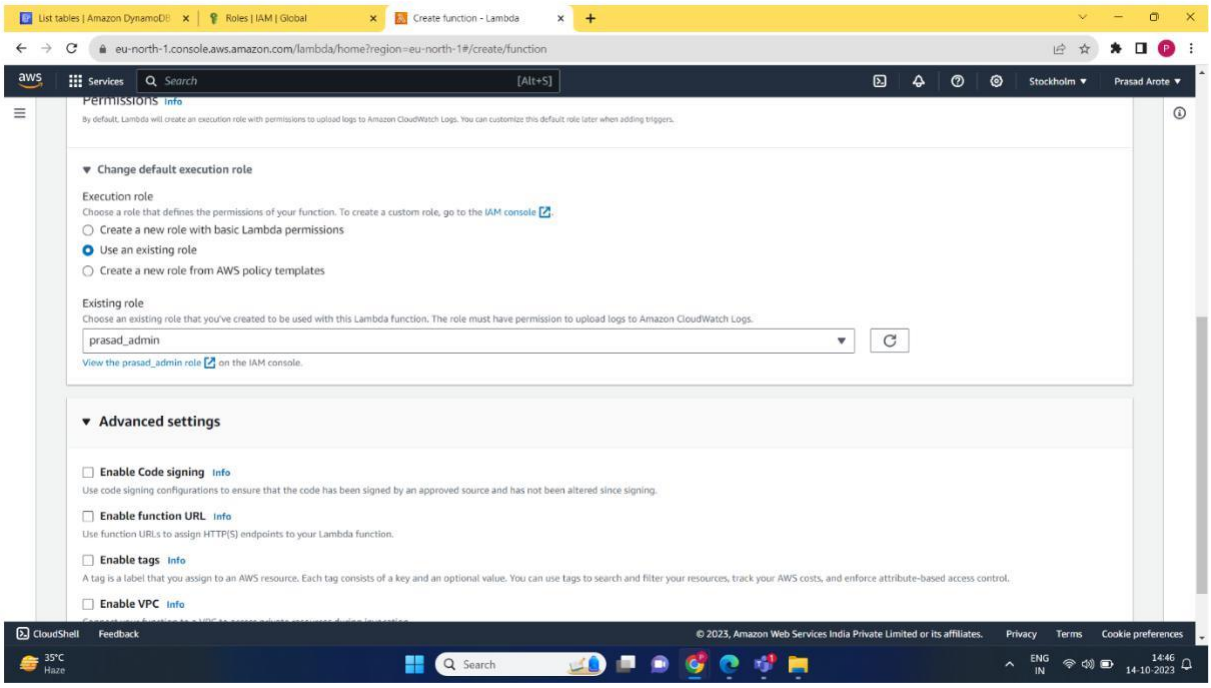


4. Create a Lambda Function

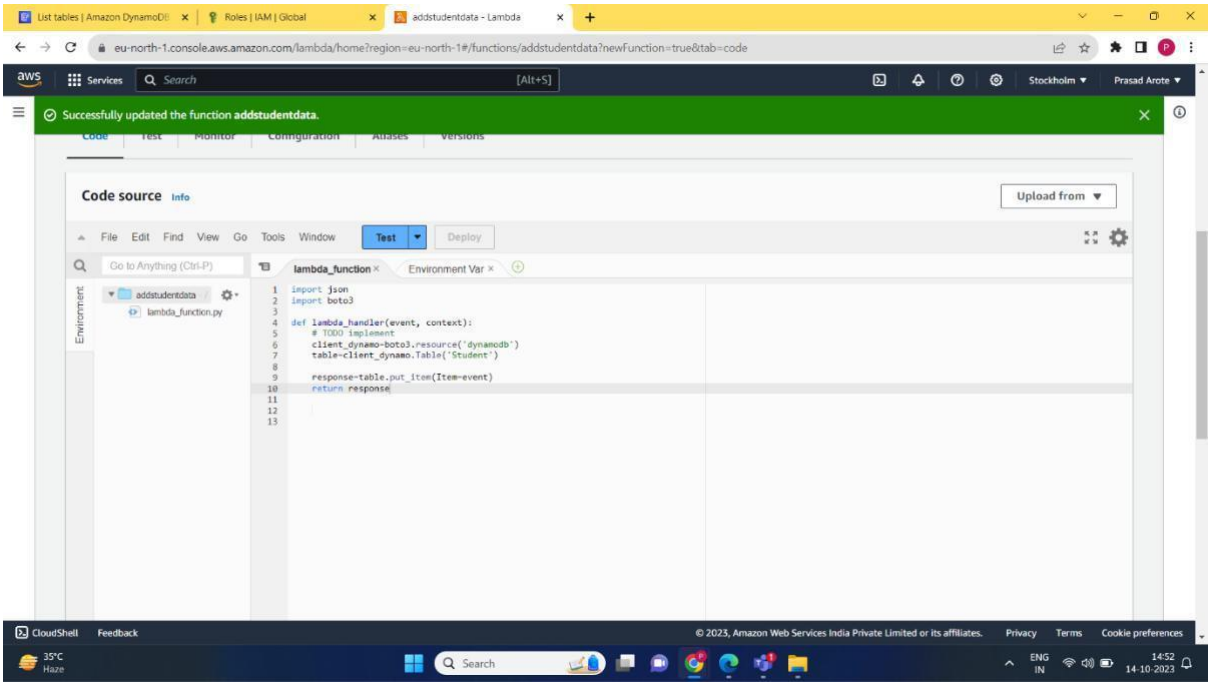

eu-north-1.console.aws.amazon.com/console/home?nc2=h_ct&src=header-signin&region=eu-north-1
lambda
Search results for 'lambda'
Try searching with longer queries for more relevant results
Services (7)
Features (3)
Resources New
Documentation (9,951)
Knowledge Articles (20)
Marketplace (664)
Blogs (1,017)
Events (13)
Tutorials (6)
Services
See all 7 results
Lambda
Run code without thinking about servers
CodeBuild
Build and Test Code
AWS Signer
Ensuring trust and integrity of your code
Amazon Inspector
Continual vulnerability management at scale
Features
Local processing
IoT Core feature
AWS Health Info
Past 7 days
Upcoming and past 7 days
Go to AWS Health
Launch a virtual machine
Start migrating to AWS
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences


eu-north-1.console.aws.amazon.com/lambda/home?region=eu-north-1#/create/function
Lambda > Functions > Create function
Create function Info
Choose one of the following options to create your function.
Author from scratch
Start with a simple Hello World example.
Use a blueprint
Container image
Browse serverless app repository
Basic information
Function name
Enter a name that describes the purpose of your function.
addstudentdata
Use only letters, numbers, hyphens, or underscores with no spaces.
Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.9
Architecture Info
Choose the instruction set architecture you want for your function code.
x86_64
arm64
Permissions Info
CloudShell
Feedback
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences

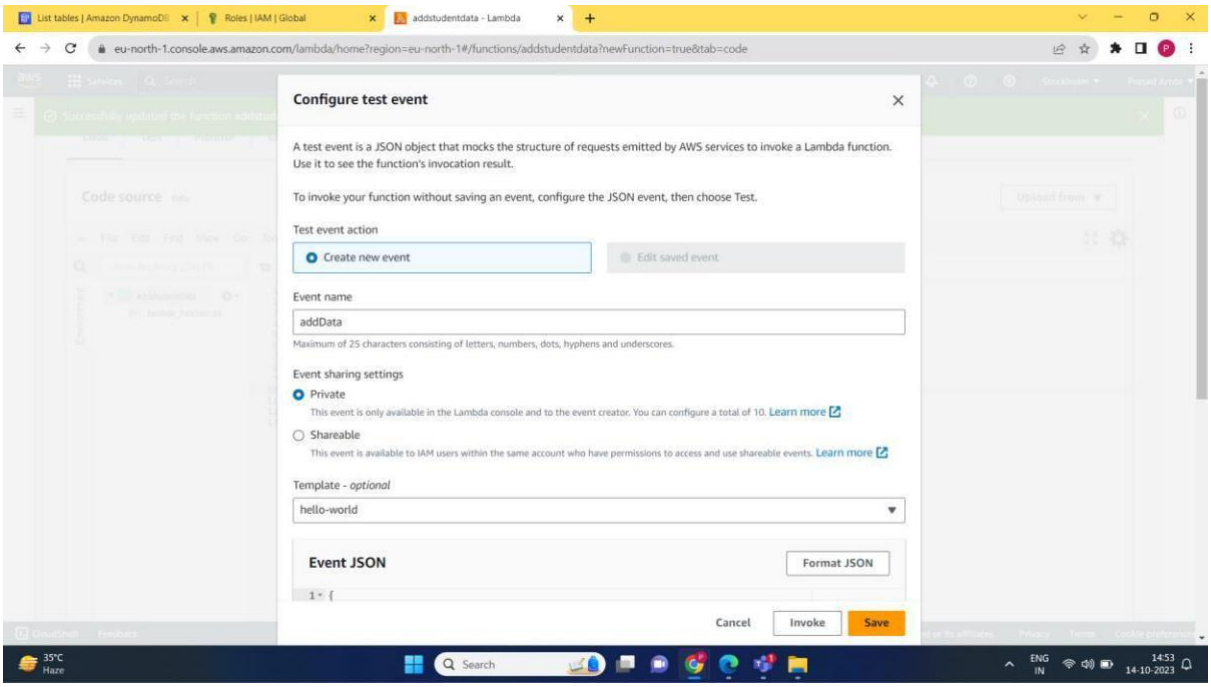


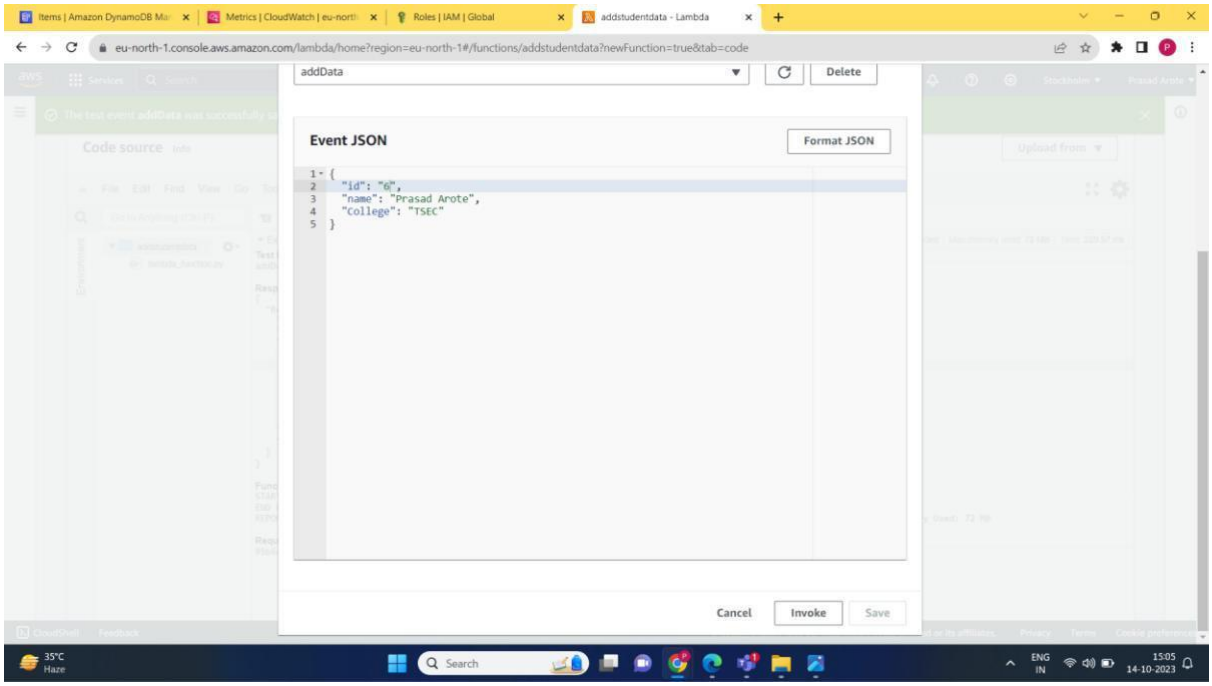


5. Write the following code

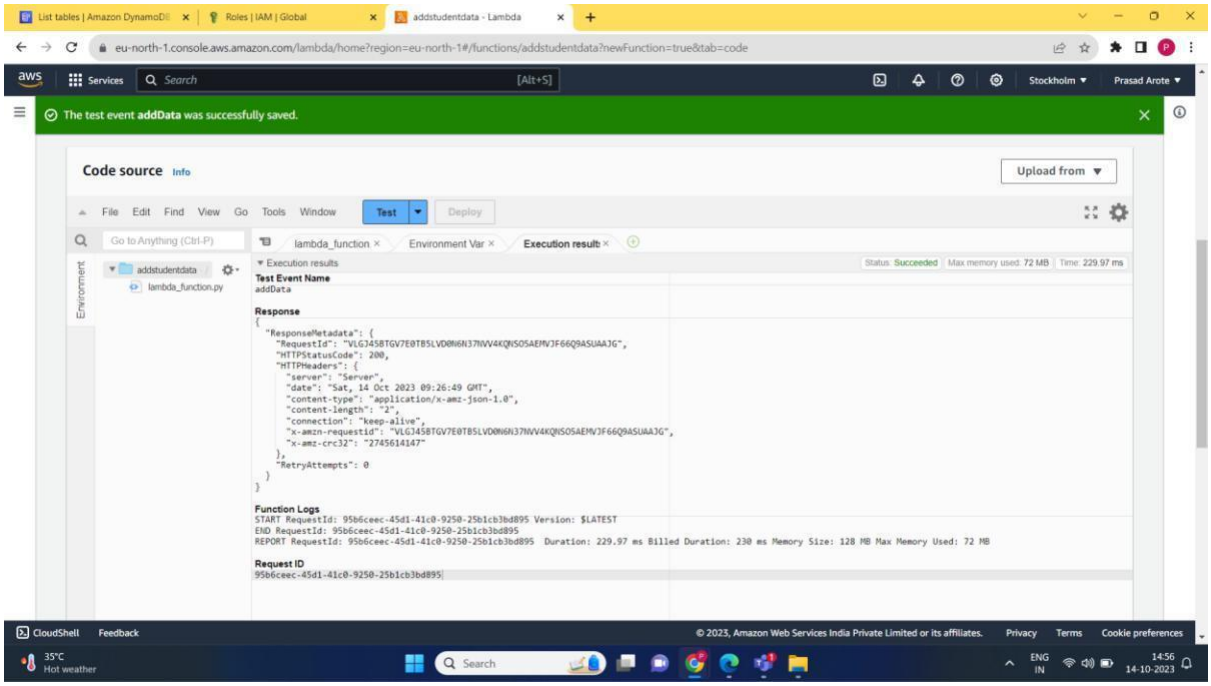


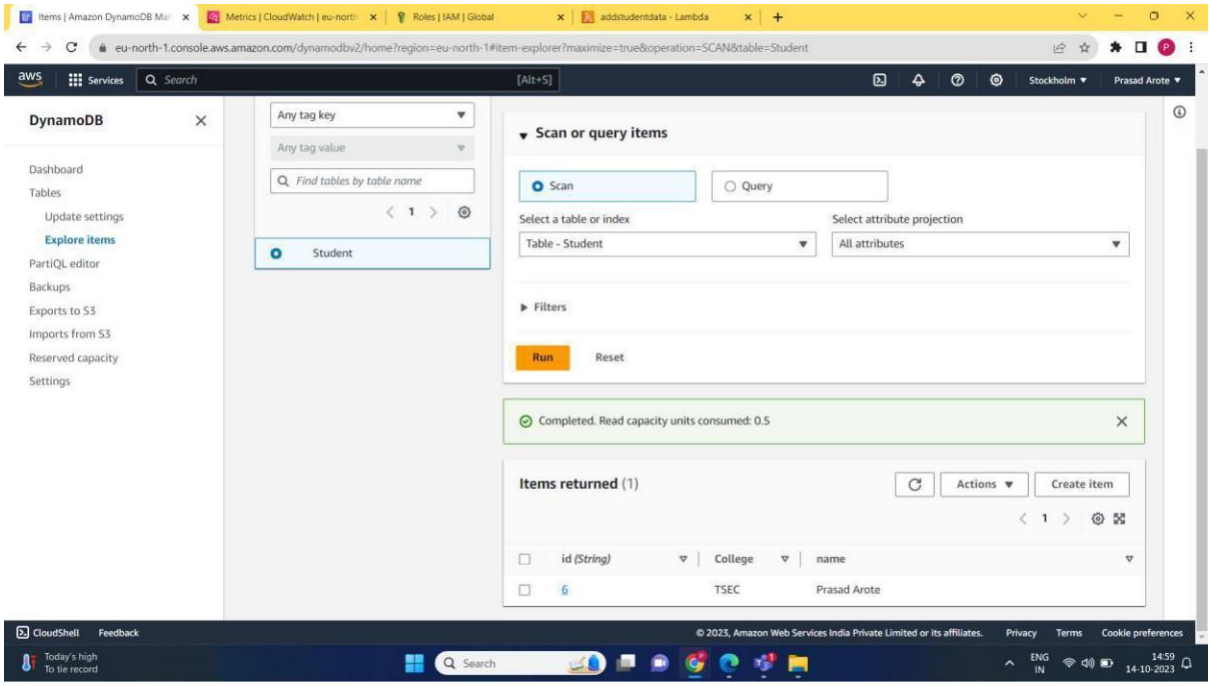
6. Configure test event and Save





7. 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.