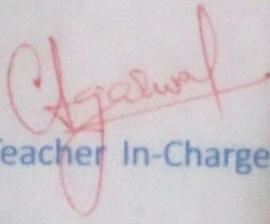


Thadomal Shahani Engineering College

Bandra (W.), Mumbai- 400 050.

CERTIFICATE

Certify that Mr./Miss Bhuvan Pravin Sawant of Information technology Department, Semester V with Roll No. 110 has completed a course of the necessary experiments in the subject Advance devops under my supervision in the **Thadomal Shahani Engineering College** Laboratory in the year 2023 - 2024


Teacher In-Charge

Head of the Department

Date 20/10/23

Principal

CONTENTS

| SR. NO. | EXPERIMENTS | PAGE NO. | DATE | TEACHERS SIGN. |
|------------|--|-------------|---------|--------------------|
| 1. | To study & perform the setup of AWS EC2 service and launch an EC2 instance | | 18/7/23 | |
| 2. | To study & perform the setup of AWS Cloud9 service & launch a python program in Cloud9 | | 25/7/23 | |
| 3. | To study AWS S3 service & create a bucket for hosting static web application | | 1/8/23 | |
| 4. | To study AWS S3 service & deploy a web application using Codepipeline | | 2/8/23 | |
| 5. | To understand the kubernetes cluster Architecture | | 16/8/23 | |
| 6. | To understand the terraform life cycle & to build, change and destroy AWS infrastructure using Terraform | | 22/8/23 | |
| 7. | To perform static analysis on python programs using SonarQube SASR process | | 29/8/23 | Pragya 20/10/23 |
| 8. | To understand continuous monitoring Using Nagios | | 28/8/23 | |
| 9. | To understand AWS lambda function & create a lambda function using python to log "An image has been added" | | 31/8/23 | 8 |
| 10. | To create AWS lambda function using python for adding data to Dynamo DB database | | 5/9/23 | |
| 11. | Written Assignment 1 | | 23/9/23 | |
| 12. | Written Assignment 2 | | 5/10/23 | |

ASSIGNMENT - 1

AIM: To create an EC2 machine-instance using AWS

LO Mapped: LO1

Theory:

Amazon Elastic Compute Cloud is a part of Amazon.com's cloud-computing platform, Amazon Web Services, that allows users to rent virtual computers on which to run their own computer applications. It is designed to make web-scale cloud computing easier for developers. Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction.

The following are the features of AWS:

- Flexibility
- Cost-effective
- Scalable and elastic
- Secure
- Experienced

Steps: Login to AWS account, then search EC2.

Select the cloud computing platform you want to work on

The screenshot shows the AWS EC2 'Launch an instance' wizard. The top navigation bar includes the AWS logo, 'Services' dropdown, a search bar, and user information ('Stockholm' and 'BhuvanSawant'). The main page title is 'Launch an instance' with an 'Info' link. Below it, a sub-section titled 'Name and tags' also has an 'Info' link. A 'Name' input field contains 'MyFirstInstance' and a 'Add additional tags' link. The next section, 'Application and OS Images (Amazon Machine Image)', also has an 'Info' link. It includes a search bar for AMIs and a 'Quick Start' section with icons for Amazon Linux, macOS, Ubuntu, Windows, and Red Hat. A 'Browse more AMIs' link is present. On the right side, a 'Summary' panel shows 'Number of instances' set to 1. It lists the selected 'Software Image (AMI)' as Microsoft Windows Server 2022, 'Virtual server type (instance type)' as t3.micro, and 'Storage (volumes)' as 1 volume(s) - 30 GB. A callout box highlights the 'Free tier' benefit: 'In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.' At the bottom are 'Cancel', 'Launch instance' (in orange), and 'Review commands' buttons.

Now wait till the status check is 2/2 and instance is running.

The screenshot shows the AWS EC2 Instances page. The left sidebar includes sections for EC2 Dashboard, EC2 Global View, Events, Instances (with sub-options like Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations), Images (AMIs, AMI Catalog), Elastic Block Store (Volumes, Snapshots, Lifecycle Manager), and Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs). The main content area displays a table of instances:

| Name | Instance ID | Instance state | Instance type | Status check | Alarm status | Availability zone |
|-----------------|----------------------|----------------|---------------|-------------------|--------------|-------------------|
| MyFirstInstance | i-0703bf821868ccdb89 | Running | t3.micro | 2/2 checks passed | No alarms | + eu-nor |
| INSTANCE2 | i-043f4cd2ba958d4de | Running | t3.micro | Initializing | No alarms | + eu-nor |

Below the table, a modal window titled "Select an instance" is open, showing the same list of instances. At the bottom of the page, there are links for CloudShell, Feedback, Language, and cookie preferences.

Once Check is complete click on launch instances.

The screenshot shows the "Connect to instance" dialog for instance i-043f4cd2ba958d4de (INSTANCE2). The top navigation bar shows the path: EC2 > Instances > i-043f4cd2ba958d4de > Connect to instance. The dialog has tabs for EC2 Instance Connect, Session Manager, SSH client, and EC2 serial console. The EC2 Instance Connect tab is selected. It displays the instance ID (i-043f4cd2ba958d4de (INSTANCE2)), connection type (selected: "Connect using EC2 Instance Connect" - "Connect using the EC2 Instance Connect browser-based client, with a public IPv4 address."), public IP address (13.53.193.236), user name (ec2-user), and a note about the default user name. At the bottom are "Cancel" and "Connect" buttons.

After connecting to instance successfully the portal is ready

The screenshot shows the AWS EC2 Connect interface. At the top, the navigation bar includes 'Services' (selected), 'Search', and 'Stockholm | BhuvanSawant'. Below the navigation, the breadcrumb path is 'EC2 > Instances > i-043f4cd2ba958d4de > Connect to instance'. The main section is titled 'Connect to instance' with a 'Info' link. It displays the instance ID 'i-043f4cd2ba958d4de (INSTANCE2)'. Under 'Connection Type', the 'Connect using EC2 Instance Connect' option is selected. It also lists 'Connect using EC2 Instance Connect Endpoint' and 'Public IP address' (13.53.193.236). The 'User name' field contains 'ec2-user'. A note at the bottom states: 'Note: In most cases, the default user name, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.' At the bottom right are 'Cancel' and 'Connect' buttons. Below this, a terminal window shows an Amazon Linux 2023 session with the URL https://aws.amazon.com/linux/amazon-linux-2023. The terminal output includes the command 'curl -s https://aws.amazon.com/linux/amazon-linux-2023 | /bin/sh' and the message 'Last login: Tue Jul 25 06:25:18 2023 from 13.48.4.203 [ec2-user@ip-172-31-33-162 ~]\$'. A status bar at the bottom provides instance details: 'i-043f4cd2ba958d4de (INSTANCE2)', 'PublicIPs: 13.53.193.236', and 'PrivateIPs: 172.31.33.162'.

Select the instances that you created and terminate them

The screenshot shows the AWS EC2 Instances page. At the top, a green banner displays the message "Successfully stopped i-043f4cd2ba958d4de". Below the banner, the title bar says "Instances (2) Info". The main table lists two instances:

| Name | Instance ID | Instance state | Instance type | Status check | Alarm status | Ava |
|-----------------|---------------------|----------------|---------------|-------------------|--------------|-----|
| MyFirstInstance | i-0703bf821868ccdb9 | Terminated | t3.micro | - | No alarms | eu- |
| INSTANCE2 | i-043f4cd2ba958d4de | Stopping | t3.micro | 2/2 checks passed | No alarms | eu- |

Below the table, a modal window titled "Select an instance" is open, indicating that one or more instances have been selected for termination.

CONCLUSION: Understanding the concept of EC2 and to create an EC2 machine/instance using AWS was performed successfully.

Assignment - 2

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

LO Mapped: LO1

Theory:

An integrated development environment (IDE) for the cloud called AWS Cloud9 enables you to write, run, and debug code using just a web browser. It has a terminal, debugger, and code editor. You don't need to install files or set up your development workstation to start new projects because Cloud9 comes preconfigured with necessary tools for popular programming languages like JavaScript, Python, PHP, and more. Since your Cloud9 IDE is cloud-based, you may use any internet-connected device to work on your projects from your home, office, or anywhere else. Additionally, Cloud9 offers a fluid development environment for serverless apps that makes it simple to declare resources, debug, and switch between local and remote execution. You can easily pair program and keep track of each other's inputs in real time with Cloud9, which enables you to swiftly share your working environment with your colleagues.

Output:

first login to your AWS account. Search for cloud9 in the search bar.

Then click on create environment. Fill the detail of your environment. After filling all details click on create button

The screenshot shows the AWS Cloud9 interface. At the top, there is a green success message: "Successfully created Bhuvan-python. To get the most out of your environment, see Best practices for using AWS Cloud9 [?].". Below this, the navigation bar shows "AWS Cloud9 > Environments > Bhuvan-python". The main content area displays the "Bhuvan-python" environment details. The "Details" section includes fields for Name (Bhuvan-python), Owner ARN (arn:aws:iam::518654585567:root), Description (my first cloud9 enviroment for python program), Number of members (1), Environment type (EC2 instance), Status (Ready), and Lifecycle status (Created). There are "Delete" and "Open in Cloud9" buttons at the top right of the details card. Below the details card, there are tabs for "EC2 instance" (which is selected), "Network settings", and "Tags".

After clicking create button you will redirect to following page. You can open the environment after clicking on Open.

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. The row details are:

| Name | Cloud9 IDE | Environment type | Connection | Permission | Owner ARN |
|---------------|----------------------|------------------|---------------------------|------------|--------------------------------|
| Bhuvan-python | Open | EC2 instance | AWS Systems Manager (SSM) | Owner | arn:aws:iam::518654585567:root |

In file section we created a python file and wrote a python program for calculator. After saving your program you can run program on clicking Run green button at top

The screenshot shows the AWS Cloud9 IDE. The left sidebar shows a project named 'Bhuvan-python - i' with files 'first.py' and 'README.md'. The main area has two tabs: 'first.py' (code editor) and 'first.py - Stopped' (terminal). The code in 'first.py' is:

```

34     |     |     |     |     multiply(number_1, number_2))
35
36 elif select == 4:
37     print(number_1, "/", number_2, "=",
38           |     |     |     divide(number_1, number_2))
39 else:
40     print("Invalid input")
41

```

The terminal window shows the following interaction:

```

Please select operation -
1. Add
2. Subtract
3. Multiply
4. Divide

Select operations form 1, 2, 3, 4 :3
Enter first number: 5
Enter second number: 4
5 * 4 = 20

Process exited with code: 0

```

Conclusion:

In this assignment we learnt and understood the steps create an Cloud9 environment.

Assignment No. - 3

Aim: To build your application using AWS CodeBuild and Deploy on S3/SEBS using AWS CodePipeline.

LO Mapped: LO1, LO2

Theory:

Amazon Simple Storage Service (Amazon S3) is an object storage service offering industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can store and protect any amount of data for virtually any use case, such as data lakes, cloud-native applications, and mobile apps. With cost-effective storage classes and easy-to-use management features, you can optimize costs, organize data, and configure fine-tuned access controls to meet specific business, organizational, and compliance requirements.

Some of the benefits of AWS S3 are:

- Durability: S3 provides 99.999999999 percent durability.
- Low cost: S3 lets you store data in a range of “storage classes.” These classes are based on the frequency and immediacy you require in accessing files.
- Scalability: S3 charges you only for what resources you actually use, and there are no hidden fees or overage charges. You can scale your storage resources to easily meet your organization’s ever-changing demands.
- Availability: S3 offers 99.99 percent availability of objects
- Security: S3 offers an impressive range of access management tools and encryption features that provide top-notch security.
- Flexibility: S3 is ideal for a wide range of uses like data storage, data backup, software delivery, data archiving, disaster recovery, website hosting, mobile applications, IoT devices, and much more.
- Simple data transfer: You don’t have to be an IT genius to execute data transfers on S3. The service revolves around simplicity and ease of use.

First login to your AWS account.

The screenshot shows the AWS S3 console interface. At the top, a green header bar displays the message "Upload succeeded" with a link to "View details below". Below this, a table summarizes the upload results:

| Destination | Succeeded | Failed |
|-----------------------|----------------------------|-------------------|
| s3://mynews3webbucket | 2 files, 13.4 KB (100.00%) | 0 files, 0 B (0%) |

Below the table, there are two tabs: "Files and folders" (selected) and "Configuration". Under "Files and folders", a table lists the uploaded files:

| Name | Folder | Type | Size | Status | Error |
|-------------|--------|------------|---------|-------------|-------|
| welcome.jpg | img/ | image/jpeg | 13.1 KB | ✓ Succeeded | - |
| index.html | - | text/html | 279.0 B | ✓ Succeeded | - |

At the bottom of the page, there are links for CloudShell, Feedback, Language, and a footer with copyright information and links for Privacy, Terms, and Cookie preferences.

Then create a new S3 bucket.

The screenshot shows the AWS S3 console interface. A green header bar displays the message "Successfully edited public access" with a link to "View details below". Below this, a summary table provides an overview of the changes made:

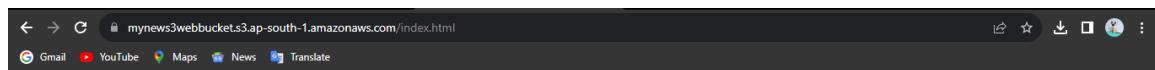
| Source | Successfully edited public access | Failed to edit public access |
|-----------------------|-----------------------------------|------------------------------|
| s3://mynews3webbucket | ✓ 2 objects, 13.4 KB | 0 objects |

Below the summary, there are two tabs: "Failed to edit public access" (selected) and "Configuration". Under "Failed to edit public access", a table shows the count of failed operations:

| Failed to edit public access (0) |
|----------------------------------|
|----------------------------------|

At the bottom of the page, there are links for CloudShell, Feedback, Language, and a footer with copyright information and links for Privacy, Terms, and Cookie preferences.

After that host a static web page using the S3 bucket.



Conclusion: In this assignment we learnt about AWS S3 bucket and hosted a static web page using the S3 bucket.

Assignment Number: 4

Name: bhuvan sawant Branch: IT/V Roll No.:110

Date:10/09/2023

Aim: To study AWS Code Pipeline and deploy web application using Code Pipeline.

LO mapped: LO1, LO2

Theory:

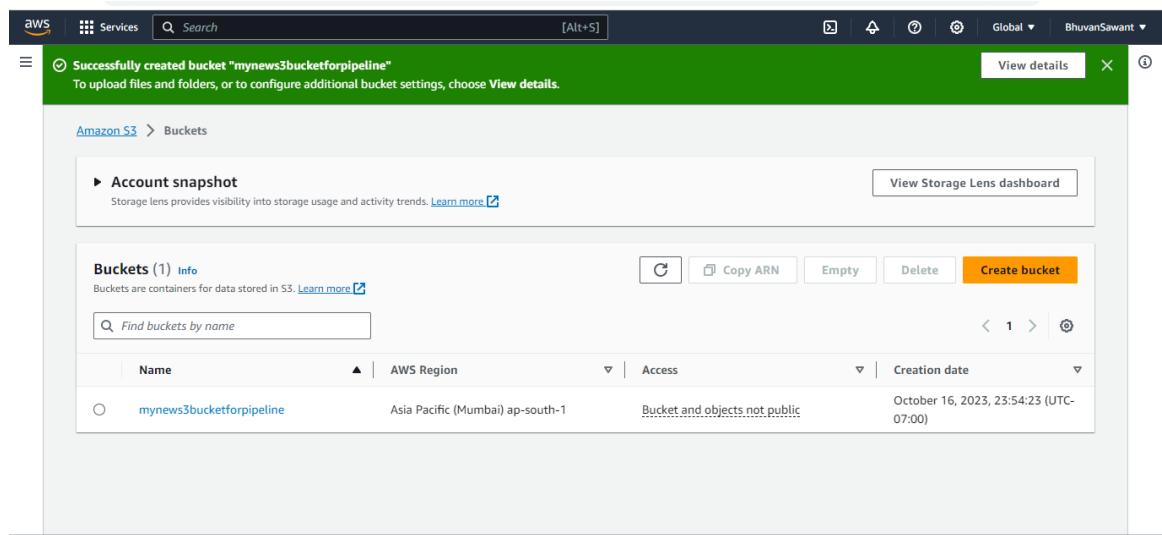
AWS CodePipeline is a continuous integration and continuous delivery (CI/CD) service provided by Amazon Web Services (AWS).

The key aspects of AWS CodePipeline:

Overview:

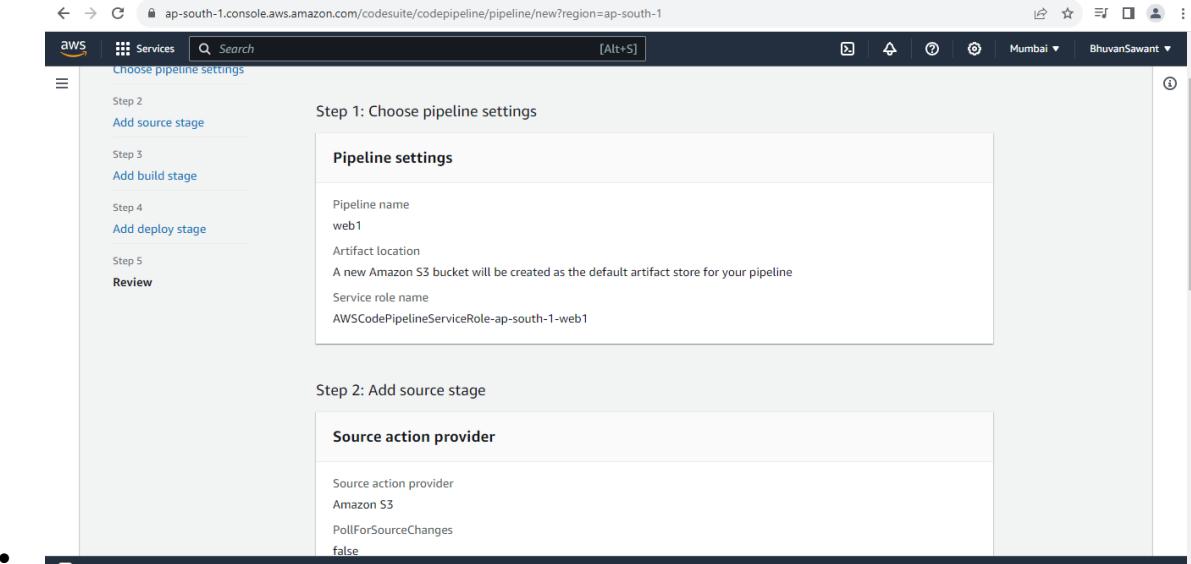
1. CI/CD Workflow:

- AWS CodePipeline facilitates the automation of the build, test, and deployment phases of the release process. It allows you to define a series of stages, each of which can represent a phase in your release pipeline.



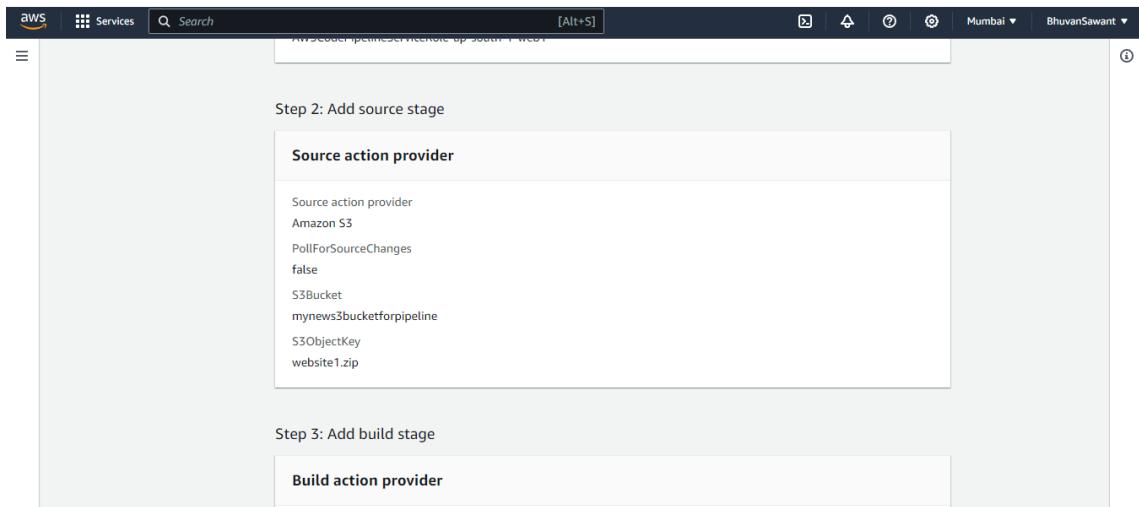
2. Integration with Other AWS Services:

- CodePipeline integrates with various AWS services, such as AWS CodeBuild for building applications, AWS CodeDeploy for automating deployments, and AWS Lambda for running custom actions.



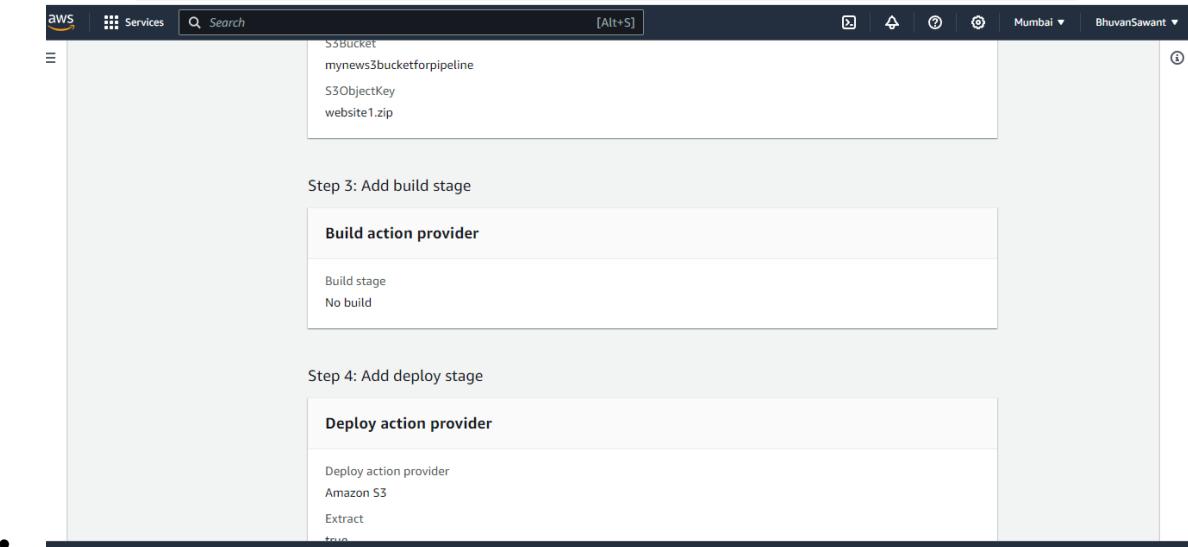
3. Pipeline Execution:

- Pipelines consist of a series of stages, and each stage can have one or more actions. Actions represent a task, such as source code retrieval or deployment to a specific environment.



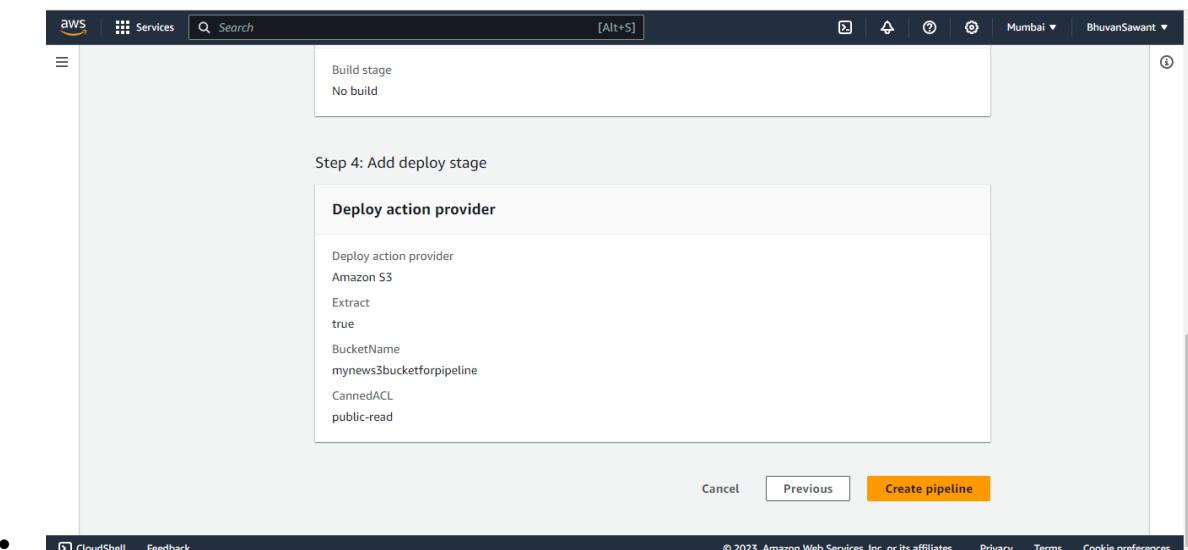
4. Source Providers:

- CodePipeline supports integration with various source code repositories, including AWS CodeCommit, GitHub, and Amazon S3.



5. Artifact Management:

- CodePipeline uses artifacts to store the files and data needed for each action in a pipeline. Artifacts can be passed between stages to ensure consistency in the deployment process.



6. Integration with Third-Party Tools:

- Besides AWS services, CodePipeline supports integration with third-party tools. This is achieved through custom actions, which allow you to use external tools and scripts in your pipeline.

The screenshot shows the AWS S3 console interface. At the top, there's a search bar and a 'Services' dropdown. Below that, a large central area has a placeholder 'Drag and drop files and folders you want to upload here, or choose Add files or Add folder.' A dashed border highlights this area. Below this, a table lists 'Files and folders (1 Total, 285.0 B)'. The table has columns for Name, Folder, Type, and Size. One item, 'website1.zip', is listed with a size of 285.0 B and type application/x-zip-compressed. There are 'Remove', 'Add files', and 'Add folder' buttons at the top of the table. Below the table is a 'Destination' section with a dropdown set to 's3://mynews3bucketforpipeline'. Underneath are sections for 'Destination details' and 'Permissions'. At the bottom of the main window are 'CloudShell', 'Feedback', and copyright information.

7. Pipeline Visualizations:

- CodePipeline provides a visual representation of your release process, making it easy to understand and monitor the status of each stage and action.

The screenshot shows the AWS S3 console after a file upload. The top bar indicates 'Upload succeeded'. Below, a summary table shows the upload status: Destination 's3://mynews3bucketforpipeline', Succeeded '1 file, 285.0 B (100.00%)', and Failed '0 files, 0 B (0%)'. At the bottom, tabs for 'Files and folders' and 'Configuration' are visible, along with a note that the information will no longer be available after navigating away.

Key Concepts:

1. Pipeline:

- A pipeline is a series of stages that represents your release process. Each stage can contain one or more actions.

2. Stage:

- A stage is a logical unit in a pipeline, representing a phase in the release process. Stages are executed sequentially.

3. Action:

- An action represents a task within a stage. Actions can include tasks such as building code, deploying to a test environment, or running tests.

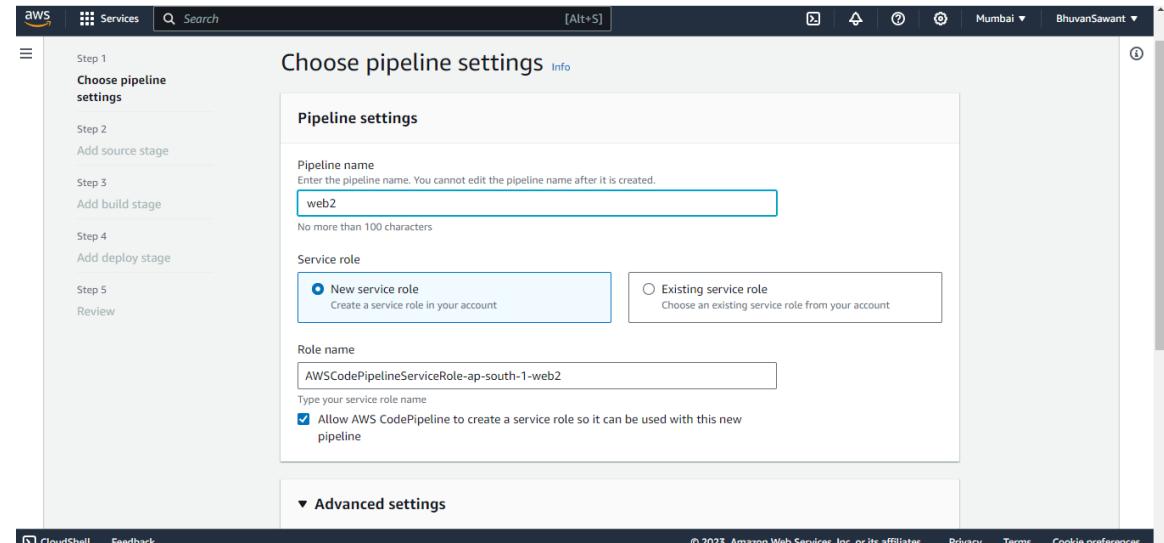
4. Artifact:

- Artifacts are the files and data that are produced as a result of an action. They are used to pass information between stages in a pipeline.

To deploy web application using CodePipeline here are the following steps to be followed:

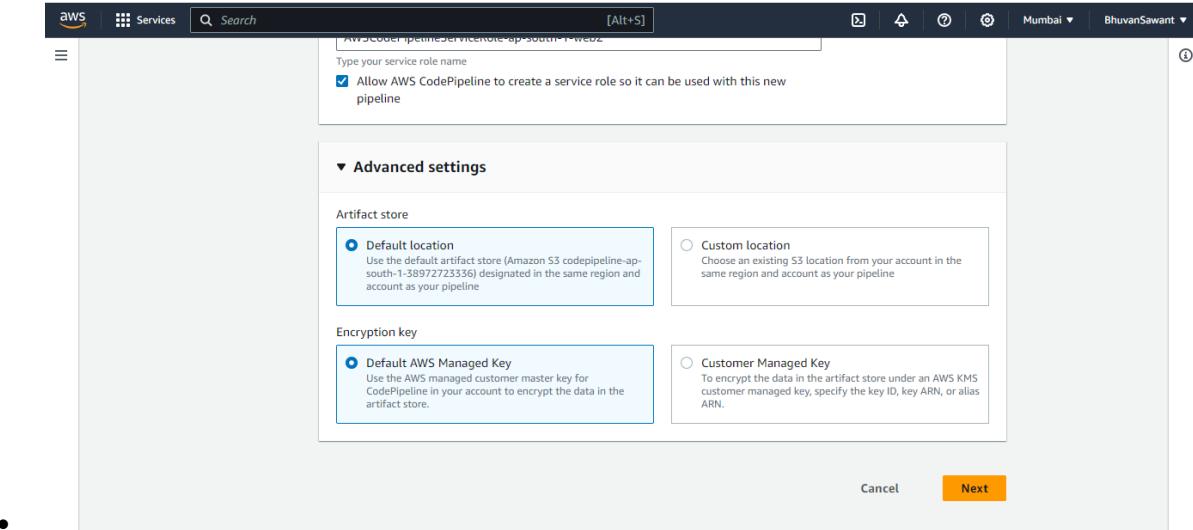
1. Set Up Source Stage:

- Configure a source stage in AWS CodePipeline, linking to your version control system (e.g., CodeCommit, GitHub).



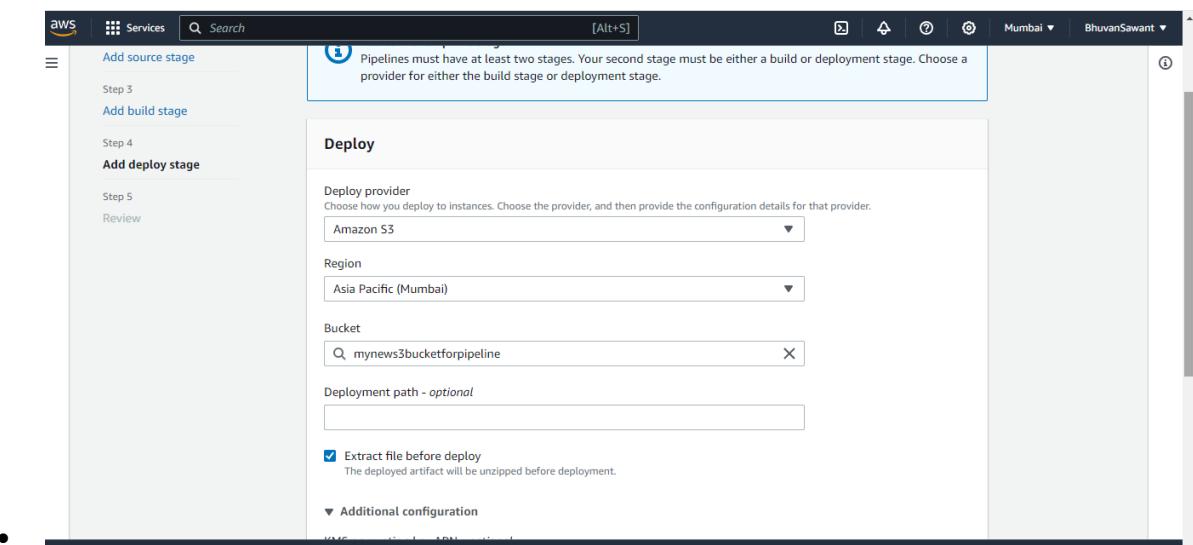
2. Configure Build Stage:

- Set up a build stage using AWS CodeBuild to compile, test, and package your web application.



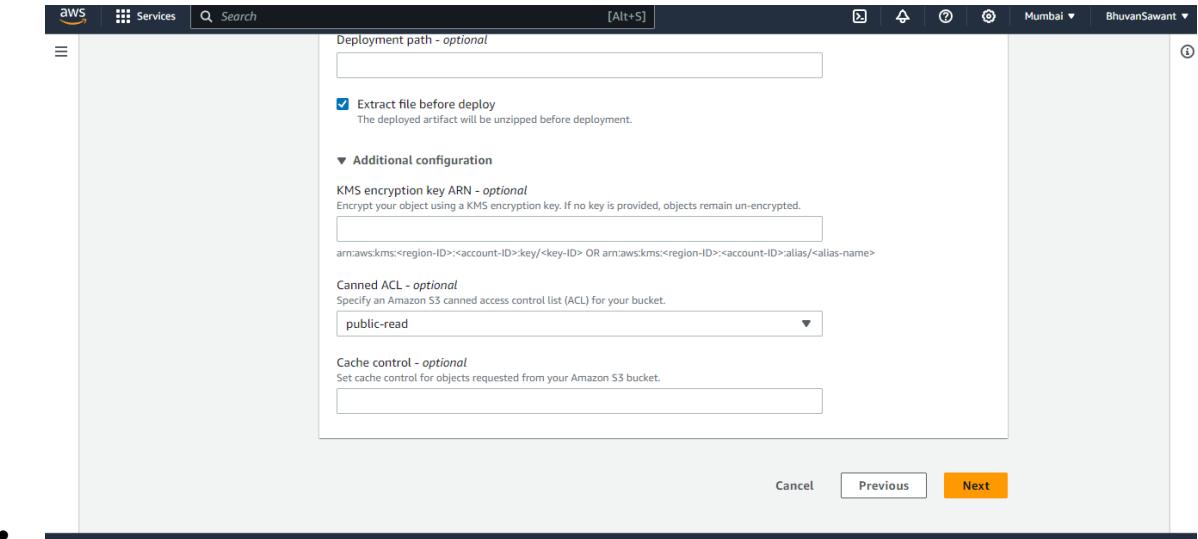
3. Define Deployment Stage:

- Create a deployment stage using AWS CodeDeploy or another deployment provider to deploy your application to target environments.



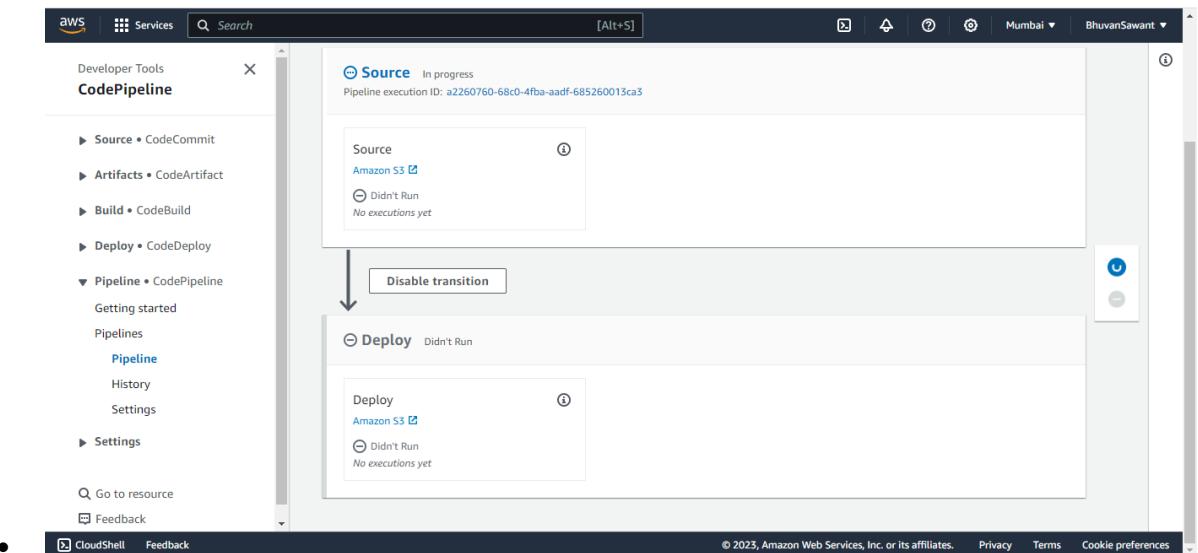
4. Configure Approval (Optional):

- Optionally, add a manual approval stage to review and approve deployments before proceeding to the next stage.



5. Artifact Passing:

- Ensure proper passing of artifacts between stages to maintain consistency in the deployment process.



6. Add Monitoring (Optional):

- Integrate monitoring tools (e.g., AWS CloudWatch) to track the performance and health of your application during and after deployment.

7. Configure Notifications (Optional):

- Set up notifications using AWS SNS or other services to receive alerts about pipeline events and status changes.

8. Test and Validate:

- Test the pipeline by triggering a build, ensuring that each stage executes successfully, and the application deploys as expected.

9. Modify Pipeline as Needed:

- Make adjustments to the pipeline configuration based on the specific requirements of your web application and deployment process.

10. Continuous Improvement:

- Implement continuous improvement practices, such as monitoring feedback, optimizing build and deployment scripts, and iterating on the pipeline structure.

Conclusion: By this assignment we learned how to host static web page through codepipeline.

ASSIGNMENT 5

Aim: To understand Kubernetes Cluster Architecture

Theory:

1. What are the various Kubernetes services running on nodes? Describe the role of each service.

In a Kubernetes cluster, several services and components run on each node to manage various aspects of container orchestration, networking, and communication. Here are some of the key services and their roles:

Kubelet: The Kubelet is an agent that runs on each node and communicates with the Kubernetes Control Plane (Master) to ensure that containers are running in a Pod. It starts and stops containers, monitors their health, and reports the current node's status to the control plane.

Kube Proxy: Kube Proxy is responsible for network connectivity and load balancing. It maintains network rules on the node and forwards traffic to the appropriate container. It helps implement services of type NodePort and LoadBalancer.

Container Runtime: This is not a Kubernetes-specific service, but it's a crucial component. It's the software responsible for running containers. Docker, containerd, and CRI-O are common container runtimes used with Kubernetes.

Node Controller: The Node Controller monitors the health of nodes and manages their lifecycle. If a node becomes unresponsive or fails, it is the Node Controller's responsibility to handle the situation.

Kube DNS: Kube DNS is a cluster DNS service that provides DNS for services within the cluster. It ensures that services and Pods can communicate with each other using their DNS names.

cAdvisor: Container Advisor (cAdvisor) is a tool that collects, aggregates, processes, and exports information about running containers. It is primarily used for monitoring and performance analysis.

Ingress Controller: An Ingress Controller manages external access to services in the cluster. It is responsible for routing external traffic to the appropriate services based on Ingress resources defined in the cluster.

Heapster/Metrics Server: Heapster was used for cluster-wide monitoring and resource usage analysis. In newer versions of Kubernetes, Heapster has been deprecated in favor of the Metrics Server, which provides resource usage metrics to various parts of Kubernetes.

Container Network Interface (CNI): The CNI is not a single service but a framework that allows different networking solutions to be used with Kubernetes. Services like Flannel, Calico, and Weave provide network plugins to implement container networking.

Kubelet Garbage Collection: This service helps clean up old and unused resources, such as unused images and containers, to reclaim disk space.

Node Problem Detector: This service detects and reports hardware and operating system problems on nodes, helping to improve cluster stability.

Kubelet Plug-ins: Kubelet supports dynamic configuration through plug-ins. These can be used for device plugins, resource management, and custom extensions to the Kubelet's functionality.

2. What is Pod Disruption Budget?

A Pod Disruption Budget (PDB) is a Kubernetes resource that allows you to define policies for how disruptions to your Pods can be controlled during events like voluntary evictions (e.g., due to scaling down) or involuntary disruptions (e.g., node failures). PDBs provide a way to ensure the availability of your applications during maintenance or scaling operations while also allowing for safe disruptions.

Here are some key aspects of Pod Disruption Budgets:

Availability Control: PDBs allow you to set constraints on how many Pods of a particular application can be disrupted simultaneously. This helps maintain a minimum level of availability for your application during various operational procedures, such as rolling updates.

Selector-Based Policy: PDBs are associated with one or more Pods through label selectors. You can specify which Pods the PDB applies to by selecting them based on labels.

MaxUnavailable: The most common constraint specified in a PDB is maxUnavailable, which defines the maximum number or percentage of Pods that are allowed to be unavailable during disruptions. For example, you might specify maxUnavailable: 1 to ensure that at least one instance of your application is available at all times.

MinAvailable: Alternatively, you can use minAvailable to specify the minimum number or percentage of Pods that must remain available. This can be useful in scenarios where you want to ensure a certain level of redundancy.

Scope: PDBs can be scoped at the namespace level, allowing you to set different disruption budgets for different applications or components within a cluster.

3. What is the role of Load Balance in Kubernetes?

In Kubernetes, load balancing plays a crucial role in distributing network traffic across multiple Pods or replicas of an application to ensure high availability, optimal resource

utilization, and the efficient use of the cluster's resources. The primary role of load balancing in Kubernetes is to:

Ensure High Availability: Load balancing helps maintain the availability of your applications by distributing incoming traffic across multiple instances of your application (Pods) running on different nodes. If one of the Pods or nodes fails, the load balancer can route traffic to healthy instances.

Optimize Resource Utilization: By evenly distributing traffic, load balancing helps prevent overloading of specific Pods or nodes. This results in better resource utilization, improved performance, and prevents any single point of failure.

Scaling and Horizontal Pod Autoscaling: Load balancing works seamlessly with scaling mechanisms like Horizontal Pod Autoscaling (HPA). When your application experiences increased traffic, HPA can dynamically create more replicas of your Pods, and the load balancer will automatically start routing traffic to the new instances.

Service Discovery: Load balancers often integrate with DNS, making it easier for clients to discover services. Kubernetes provides a built-in DNS service that resolves service names to their associated Pod IP addresses.

Traffic Management: Kubernetes offers different types of services, such as ClusterIP, NodePort, and LoadBalancer services. LoadBalancer services expose applications to external traffic, and the cloud provider's load balancer, in this case, performs the load balancing.

Ingress: Ingress controllers provide a way to manage external access to services within the cluster. They often include load balancing capabilities to distribute incoming traffic to different services based on hostnames or paths.

Session Affinity: Some load balancers can be configured to use session affinity (sticky sessions), which ensures that a client's requests are consistently directed to the same backend Pod. This can be useful for stateful applications.

Security: Load balancers can also enhance security by providing features like SSL termination, which allows them to handle encryption and decryption of traffic.

Global Load Balancing: In multi-region or multi-cloud setups, global load balancing can distribute traffic across different clusters or data centers, ensuring low-latency access for users.

Conclusion: In this assignment we understood the Kubernetes Cluster Architecture.

Assignment 6

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

Lab Outcome:

LO1, LO6 mapped

Theory:

Terraform, developed by HashiCorp, is an open-source infrastructure as code (IaC) software tool that allows users to define and provision data center infrastructure using a declarative configuration language. In simpler terms, it lets you codify your infrastructure, enabling consistent and reproducible deployments. It is used for defining, provisioning, and managing cloud infrastructure using a declarative language.

Core Concepts:

Providers: Terraform uses providers to interact with cloud services. Examples include AWS, Azure, Google Cloud, and many others. Each provider offers resource types that can be managed.

Resources: These are the primary components in Terraform. A resource might be a physical component such as an EC2 instance in AWS or a database in Azure.

State: Terraform maintains a state file that maps real-world resources to your configuration. This state is used to determine what Terraform will do on the next apply.

Modules: These are containers for multiple resources that are used together. They provide a way to group resources, and they can be used to create reusable infrastructure components.

Variables and Outputs: Variables allow for parameterization of the Terraform configuration, making it more dynamic and flexible. Outputs are a way to get information about the infrastructure, like IP addresses or DNS names.

Provisioners: While not always recommended due to their imperative nature, provisioners can be used to model specific actions on the local machine or on a remote machine, like executing scripts.

Steps :

1. Login and Search IAM on AWS Console and select the first option.

The screenshot shows the AWS IAM Dashboard. On the left, there's a sidebar with navigation options like Dashboard, Access management, Access reports, and Quick links. The main area displays security recommendations (Add MFA for root user, Root user has no active access keys), IAM resources (User groups: 1, Users: 1, Roles: 8, Policies: 1, Identity providers: 0), and a 'What's new' section. On the right, there are sections for AWS Account (Account ID: 644557152358, Account Alias: Create, Sign-in URL: https://644557152358.sigin.aws.amazon.co/m/console) and Quick Links (My security credentials). A footer at the bottom includes CloudShell, Feedback, and copyright information.

2. Head to users and Click on Create User

The screenshot shows the 'Specify user details' step of the IAM User creation wizard. It's Step 1 of 3. The user is prompted to enter a 'User name'. A note says the name can have up to 64 characters and lists valid characters. An optional checkbox allows providing console access. A note at the bottom explains generating programmatic access. At the bottom right are 'Cancel' and 'Next' buttons.

3. Enter a new username and click on next
4. Click on create group

Set permissions

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

Permissions options

- Add user to group Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
- Copy permissions Copy all group memberships, attached managed policies, and inline policies from an existing user.
- Attach policies directly Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

| User groups (1) | |
|-----------------|-------|
| Group name | Users |
| group1 | 0 |

Set permissions boundary - optional

Cancel Previous Next

5. Select the first Policy will full access and enter a group name and create the group.

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

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

Permissions policies (1/885)

| Policy name | Type | Use... | Description |
|---|-------------|--------|--|
| <input checked="" type="checkbox"/> AdministratorAccess | AWS managed | None | Provides full access to AWS services |
| <input type="checkbox"/> AdministratorAccess | AWS managed | None | Grants account administrative permission |
| <input type="checkbox"/> AlexaForBusinessD... | AWS managed | None | Provide device setup access to Alexa |
| <input type="checkbox"/> AlexaForBusinessE... | AWS managed | None | Grants full access to AlexaForBusiness |
| <input type="checkbox"/> AlexaForBusinessG... | AWS managed | None | Provide gateway execution access to Alexa |
| <input type="checkbox"/> AlexaForBusinessLi... | AWS managed | None | Provide access to Lifesize AVS devices |
| <input type="checkbox"/> AlexaForBusinessP... | AWS managed | None | Provide access to Poly AVS devices |
| <input type="checkbox"/> AlexaForBusinessR... | AWS managed | None | Provide read only access to AlexaForBusiness |

CloudShell Feedback © 2023, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences

6. Then click on next, select the policy and create user.

Set permissions

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

| Group name | Users | Attached policies | Created |
|-------------|-------|---------------------|--------------------------|
| group1 | 0 | AdministratorAccess | 2023-08-22 (1 month ago) |
| terraform@1 | 0 | AdministratorAccess | 2023-10-16 (Now) |

7. Then create an access key with CLI.

Access key best practices & alternatives [Info](#)

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

| Use case |
|--|
| <input checked="" type="radio"/> Command Line Interface (CLI) You plan to use this access key to enable the AWS CLI to access your AWS account. |
| <input type="radio"/> Local code You plan to use this access key to enable application code in a local development environment to access your AWS account. |
| <input type="radio"/> Application running on an AWS compute service You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account. |
| <input type="radio"/> Third-party service You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources. |
| <input type="radio"/> Application running outside AWS You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources. |
| <input type="radio"/> Other |

8. Leave the description field empty and create a key. Download the access key in the .csv file.

Set description tag - optional [Info](#)

The description for this access key will be attached to this user as a tag and shown alongside the access key.

Description tag value
Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidently later.

Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and _ ; / = + - @

9. Install terraform.

```
Last login: Sat Aug 19 17:44:56 on ttys000
> brew tap hashicorp/tap
Running `brew update --auto-update`...
Installing from the API is now the default behaviour!
You can save space and time by running:
  brew untap homebrew/core
  brew untap homebrew/cask
==> Auto-updated Homebrew!
Updated 3 taps (mongodb/brew, homebrew/core and homebrew/cask).
==> New Formulae
arm-none-eabi-binutils                         medusa
arm-none-eabi-gcc                               mjml
arm-none-eabi-gdb                               mongodb/brew/mongodb-community@6.0
asnmap                                         mongodb/brew/mongodb-enterprise@6.0
cargo-auditable                                mongodb/brew/mongodb-mongocryptd@6.0
cdi                                            mysql-client@8.0
cloudlist                                      mysql@8.0
coder                                           ollama
ctpv                                           proxyf
czkawka                                       python-certifi
dnsrobocert                                    riff
dolphie                                       riscv64-elf-binutils
ebook2cw                                       riscv64-elf-gcc
go@1.20                                        riscv64-elf-gdb
img2pdf                                         rpmspectool
```

10. Create the terraform config file main.tf with required configurations.

```
> nano main.tf
> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 4.16"...
- Installing hashicorp/aws v4.67.0...
- Installed hashicorp/aws v4.67.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

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

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
~/terraform_scripts ➤  ✓ 25s
```

11. Run the command terraform plan and enter yes

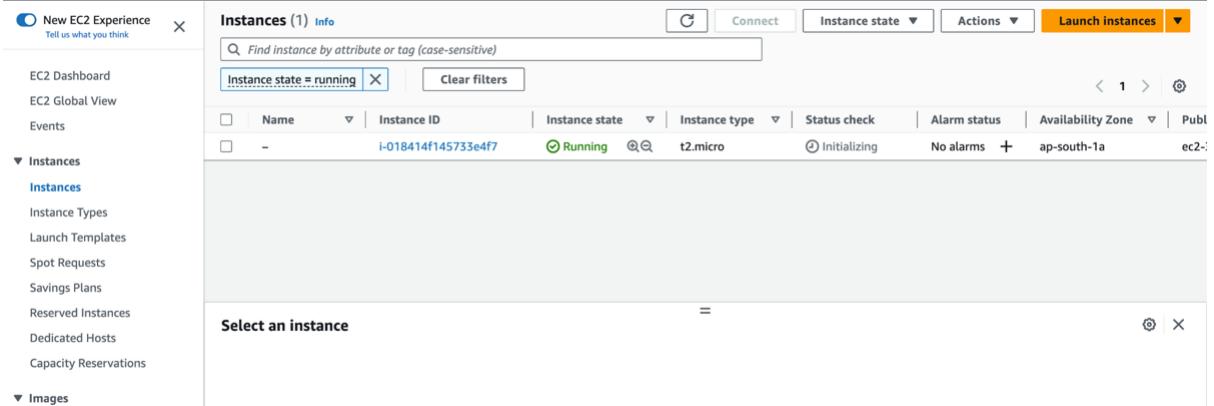
```
If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
> terraform plan

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

Terraform will perform the following actions:

# aws_instance.Ubuntu will be created
+ resource "aws_instance" "Ubuntu" {
    + ami                                     = "ami-0f5ee92e2d63afc18"
    + arn                                     = (known after apply)
    + associate_public_ip_address             = (known after apply)
    + availability_zone                      = (known after apply)
    + cpu_core_count                         = (known after apply)
    + cpu_threads_per_core                  = (known after apply)
    + disable_api_stop                      = (known after apply)
    + disable_api_termination               = (known after apply)
    + ebs_optimized                          = (known after apply)
    + get_password_data                     = false
    + host_id                                = (known after apply)
    + host_resource_group_arn                = (known after apply)
```

12. Check EC2 instances on AWS, you'll find an instance running started by terraform.



The screenshot shows the AWS EC2 Instances page. The left sidebar has a 'New EC2 Experience' header and a 'Tell us what you think' link. Under 'Instances', there are links for 'Instances', 'Instance Types', 'Launch Templates', 'Spot Requests', 'Savings Plans', 'Reserved Instances', 'Dedicated Hosts', and 'Capacity Reservations'. The main content area has a title 'Instances (1) Info' with a 'C' button, 'Connect' button, 'Instance state' dropdown, 'Actions' dropdown, and a 'Launch Instances' button. A search bar says 'Find instance by attribute or tag (case-sensitive)' with a magnifying glass icon. Below it is a filter bar with 'Instance state = running' and a clear filters button. A table lists one instance: 'i-018414f145733e4f7' is running, t2.micro, initializing, no alarms, in ap-south-1a, and ec2-. At the bottom, a modal window titled 'Select an instance' is open.

13. Now run terraform destroy to terminate the instance.

```
- enable_resource_name_dns_a_record    = false -> null
- enable_resource_name_dns_aaaa_record = false -> null
- hostname_type                      = "ip-name" -> null
}

- root_block_device {
  - delete_on_termination = true -> null
  - device_name          = "/dev/sda1" -> null
  - encrypted             = false -> null
  - iops                  = 100 -> null
  - tags                  = {} -> null
  - throughput             = 0 -> null
  - volume_id              = "vol-07a64df7d00e4e7a1" -> null
  - volume_size            = 8 -> null
  - volume_type             = "gp2" -> null
}
}
```

Plan: 0 to add, 0 to change, 1 to destroy.

Do you really want to destroy all resources?

Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

Conclusion:

We understood the workings of Terraform to create and manage EC2 instances in AWS which provides a highly automated and reproducible infrastructure-as-code solution. The ability to effortlessly spin up and tear down instances not only enhances operational agility but also ensures optimal resource utilization.

AdvDevOps Assignment 7

Aim: To perform static analysis on python programs using SonarCube SAST processes

LO Mapped: LO4

Theory:

Download SonarQube and Sonar Scanner

| Community EDITION | Developer EDITION | Enterprise EDITION | Data Center EDITION |
|---|---|---|---|
| Used and loved by 200,000+ companies FREE & OPEN SOURCE | Download | Download | Download |
| All the following features: | Community Edition plus: | Developer Edition plus: | Enterprise Edition plus: |
| <ul style="list-style-type: none"> ✓ Static code analysis for 15 languages Java, JavaScript, C/C++, TypeScript, Kotlin, Ruby, Go, Scala, F#, Python, PHP, Linux, Mac OS X, .NET | <ul style="list-style-type: none"> ✓ C, C++, Obj-C, Swift, ABAP, T-SQL, PL/SQL support ✓ Detection of Injection Flaws | <ul style="list-style-type: none"> ✓ Portfolio Management & PDF Executive Reports ✓ Project PDF reports | <ul style="list-style-type: none"> ✓ Component redundancy ✓ Data resiliency ✓ Horizontal Scalability |

The screenshot shows the SonarScanner documentation page. The main content is titled "Configuring your project" and instructs the user to create a configuration file named "sonar-project.properties". It provides examples of properties such as "sonar.projectKey=MyProject", "sonar.sources=.", and "sonar.projectVersion=1.0". On the left sidebar, there is a navigation menu for SonarQube documentation, including sections like Try Out SonarQube, Requirements, Setup and Upgrade, Analyzing Source Code, Scanners, Languages, Test Coverage & Execution, Importing External Issues, and Background Tasks.

After downloading, set Environment Variables. Add "sonarqube-9.1.0.47736\bin" to Path.

Open command prompt. Run commands:

- cd "sonarqube-9.1.0.47736\bin\windows-x86-64"
- StartSonar.bat

```
Administrator: SonarQube Microsoft Windows [Version 10.0.19043.1237]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\bin\windows-x86-64> StartSonar.bat
[pid: 1] wrapper | --> Wrapper Started as Console
[pid: 1] wrapper | Launching a JVM...
[pid: 1] wrapper |   Wrapper (Version 3.2.3) http://wrapper.tanukisoftware.org
[pid: 1] wrapper |   Copyright 1999-2006 Tanuki Software, Inc. All Rights Reserved.
[pid: 1] wrapper | 
[pid: 1] 2021-09-20 11:56:37 INFO app[[o.s.a.AppFileSystem]] Cleaning or creating temp directory: C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\tmp
[pid: 1] 2021-09-20 11:56:37 INFO app[[o.s.a.ProcessLauncherImpl]] Launch process [key=es_indexer_i, pidIndex=1, logfilename=es_indexer_i.log] from C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\elasticsearch]: C:\Program Files\Java\jdk-11.0.2\bin\java -XX:+UseG1GC -Djava.io.tmpdir=C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\tmp -XX:ErrorFile=-.logs/es_i_err.pid&log=es_indexer_i.log&networkAddress.cache.ttl=60 -Des.networkAddress.cache.size=500 -Des.cache.negativeTimeToLive=60000 -Des.cache.maxSize=1000000 -Des.cache.timeToLive=1000000 -Des.netty.allocator.type=pooled -Des.netty.unsafe.maxDirectMemorySize=256M -XX:+HeapDumpOnOutOfMemoryError -Delaisstach.boot.home=C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\elasticsearch -Ds.path.conf=C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\tmp\conf
[pid: 1] <cp lib/* org.elasticsearch.bootstrap.Bootstrap$SchedulerImpl> Waiting for Elasticsearch to be up and running
[pid: 1] 2021-09-20 11:56:39 ERROR app[[o.s.a.ESManagedProcess]] Failed to check status
[pid: 1] org.elasticsearch.ElasticsearchException: java.util.concurrent.ExecutionException: java.net.ConnectException: Timeout connecting to [/127.0.0.1:9001]
[pid: 1] at org.elasticsearch.client.RestHighLevelClient.innerPerformRequest(RestHighLevelClient.java:200)
[pid: 1] at org.elasticsearch.client.RestHighLevelClient.performRequest(RestHighLevelClient.java:172)
[pid: 1] at org.elasticsearch.client.RestHighLevelClient.listPerformanceEntity(RestHighLevelClient.java:1672)
[pid: 1] at org.elasticsearch.client.es.EsConnectorImpl.getClusterHealthStatus(EsConnectorImpl.java:64)
[pid: 1] at org.sonar.application.es.EsManagedProcess.checkStatus(EsManagedProcess.java:90)
[pid: 1] at org.sonar.application.es.EsManagedProcess.isOperational(EsManagedProcess.java:75)
[pid: 1] at org.sonar.application.es.EsManagedProcess.refreshState(ManageProcessHandler.java:198)
[pid: 1] at org.sonar.application.process.ManageProcessHandler.refreshState(ManageProcessHandler.java:220)
[pid: 1] at org.sonar.application.process.ManageProcessHandler.isOperational(ManageProcessHandler.java:285)
[pid: 1] Caused by: java.util.concurrent.ExecutionException: java.net.ConnectException: Connection refused to [/127.0.0.1:9001]
[pid: 1] at org.elasticsearch.common.util.concurrent.BaseFutureSync.getVal(BaseFuture.java:262)
[pid: 1] at org.elasticsearch.common.util.concurrent.BaseFutureSync.get(BaseFuture.java:249)
[pid: 1] at org.elasticsearch.common.util.concurrent.BaseFuture.get(BaseFuture.java:70)
[pid: 1] at org.elasticsearch.client.RestHighLevelClient.performClientRequest(RestHighLevelClient.java:2075)
[pid: 1] 10 more frames omitted
[pid: 1] Caused by: java.net.ConnectException: Timeout connecting to [/127.0.0.1:9001]
[pid: 1] at org.elasticsearch.client.es.RoutingSpecificPool.timeout(RoutingSpecificPool.java:169)
[pid: 1] at org.apache.http.nio.pool.AbstractNIOConnPool.requestTimeout(AbstractNIOConnPool.java:628)
[pid: 1] at org.apache.http.nio.pool.InternalSessionRequest.timeout(InternalSessionRequest.java:184)
[pid: 1] at org.apache.http.nio.pool.SessionRequestImpl.timeout(SessionRequestImpl.java:184)
[pid: 1] at org.apache.http.impl.nio.reactor.DefaultConnectingIOReactor.connectingIOReactor.connectingIOReactor.java:214)
[pid: 1] at org.apache.http.impl.nio.reactor.DefaultConnectingIOReactor.processEvents(DefaultConnectingIOReactor.java:158)
[pid: 1] at org.apache.http.impl.nio.reactor.AbstractMultiworkerIOReactor.execute(AbstractMultiworkerIOReactor.java:51)
[pid: 1] at org.apache.http.impl.nio.client.PoolingNHttpClientConnectionManager.execute(PoolingNHttpClientConnectionManager.java:221)
[pid: 1] at org.apache.http.impl.nio.client.CloseableHttpAsyncClientBase$1.run(CloseableHttpAsyncClientBase.java:64)
[pid: 1] at java.base/java.lang.Thread.run(Thread.java:84)
```

```
Administrator: SonarQube Microsoft Windows [Version 10.0.19043.1237]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\bin\windows-x86-64> StartSonar.bat
[pid: 1] at org.elasticsearch.client.RestHighLevelClient.innerPerformRequest(RestHighLevelClient.java:172)
[pid: 1] at org.elasticsearch.client.es.RoutingSpecificPool.timeout(RoutingSpecificPool.java:169)
[pid: 1] at org.elasticsearch.client.es.EsConnectorImpl.getClusterHealthStatus(EsConnectorImpl.java:64)
[pid: 1] at org.sonar.application.es.EsManagedProcess.checkStatus(EsManagedProcess.java:90)
[pid: 1] at org.sonar.application.es.EsManagedProcess.isOperational(EsManagedProcess.java:75)
[pid: 1] at org.sonar.application.es.EsManagedProcess.refreshState(ManageProcessHandler.java:198)
[pid: 1] at org.sonar.application.process.ManageProcessHandler.refreshState(ManageProcessHandler.java:220)
[pid: 1] at org.sonar.application.process.ManageProcessHandler.isOperational(ManageProcessHandler.java:285)
[pid: 1] Caused by: java.util.concurrent.ExecutionException: java.net.ConnectException: Timeout connecting to [/127.0.0.1:9001]
[pid: 1] at org.elasticsearch.common.util.concurrent.BaseFutureSync.getVal(BaseFuture.java:262)
[pid: 1] at org.elasticsearch.common.util.concurrent.BaseFutureSync.get(BaseFuture.java:249)
[pid: 1] at org.elasticsearch.common.util.concurrent.BaseFuture.get(BaseFuture.java:70)
[pid: 1] at org.elasticsearch.client.RestHighLevelClient.performClientRequest(RestHighLevelClient.java:2075)
[pid: 1] 10 more frames omitted
[pid: 1] Caused by: java.net.ConnectException: Timeout connecting to [/127.0.0.1:9001]
[pid: 1] at org.apache.http.nio.pool.AbstractNIOConnPool.requestTimeout(AbstractNIOConnPool.java:628)
[pid: 1] at org.apache.http.nio.pool.InternalSessionRequest.timeout(InternalSessionRequest.java:184)
[pid: 1] at org.apache.http.impl.nio.SessionRequestImpl.timeout(SessionRequestImpl.java:184)
[pid: 1] at org.apache.http.impl.nio.reactor.DefaultConnectingIOReactor.connectingIOReactor.connectingIOReactor.java:214)
[pid: 1] at org.apache.http.impl.nio.reactor.DefaultConnectingIOReactor.processEvents(DefaultConnectingIOReactor.java:158)
[pid: 1] at org.apache.http.impl.nio.reactor.AbstractMultiworkerIOReactor.execute(AbstractMultiworkerIOReactor.java:51)
[pid: 1] at org.apache.http.impl.nio.client.PoolingNHttpClientConnectionManager.execute(PoolingNHttpClientConnectionManager.java:221)
[pid: 1] at org.apache.http.impl.nio.client.CloseableHttpAsyncClientBase$1.run(CloseableHttpAsyncClientBase.java:64)
[pid: 1] at java.base/java.lang.Thread.run(Thread.java:84)

[pid: 1] 2021-09-20 11:56:58 INFO app[[o.s.a.SchedulerImpl]] Process[es_i] is up
[pid: 1] 2021-09-20 11:56:58 INFO app[[o.s.a.ProcessLauncherImpl]] Launch process [key=es_indexer_i, logfilename=es_indexer_i.log] from C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\elasticsearch]: C:\Program Files\Java\jdk-11.0.2\bin\java -XX:+UseG1GC -Djava.io.tmpdir=C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\tmp -XX:ErrorFile=-.logs/es_i_err.pid&log=es_indexer_i.log&networkAddress.cache.ttl=60 -Des.networkAddress.cache.size=500 -Des.cache.negativeTimeToLive=60000 -Des.cache.maxSize=1000000 -Des.cache.timeToLive=1000000 -Des.netty.allocator.type=pooled -Des.netty.unsafe.maxDirectMemorySize=256M -XX:+HeapDumpOnOutOfMemoryError -Delaisstach.boot.home=C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\elasticsearch -Ds.path.conf=C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\tmp\conf
[pid: 1] 2021-09-20 11:51:42 INFO app[[o.s.a.SchedulerImpl]] Process[web] is up
[pid: 1] 2021-09-20 11:51:42 INFO app[[o.s.a.ProcessLauncherImpl]] Launch process [key=ce, ipcIndex=3, logfilename=ce.log] from C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\elasticsearch]: C:\Program Files\Java\jdk-11.0.2\bin\java -XX:+UseG1GC -Djava.io.tmpdir=C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\tmp -XX:ErrorFile=-.logs/ce_err.pid&log=ce.log&networkAddress.cache.ttl=60 -Des.networkAddress.cache.size=500 -Des.cache.negativeTimeToLive=60000 -Des.cache.maxSize=1000000 -Des.cache.timeToLive=1000000 -Des.netty.allocator.type=pooled -Des.netty.unsafe.maxDirectMemorySize=256M -XX:+HeapDumpOnOutOfMemoryError -Delaisstach.boot.home=C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\elasticsearch -Ds.path.conf=C:\Users\Priyanshi\Downloads\sonarqube-9.1.0.47736\tmp\conf
[pid: 1] 2021-09-20 11:51:42 INFO app[[o.s.a.SchedulerImpl]] Process[ce] is up
[pid: 1] 2021-09-20 11:51:42 INFO app[[o.s.a.SchedulerImpl]] Default Administrator credentials are still being used. Make sure to change the password or deactivate the account.
[pid: 1] 2021-09-20 11:51:42 INFO app[[o.s.a.SchedulerImpl]] Process[ce] is up
[pid: 1] 2021-09-20 11:51:46 INFO app[[o.s.a.SchedulerImpl]] Process[ce] is up
[pid: 1] 2021-09-20 11:51:46 INFO app[[o.s.a.SchedulerImpl]] SonarQube is up
```

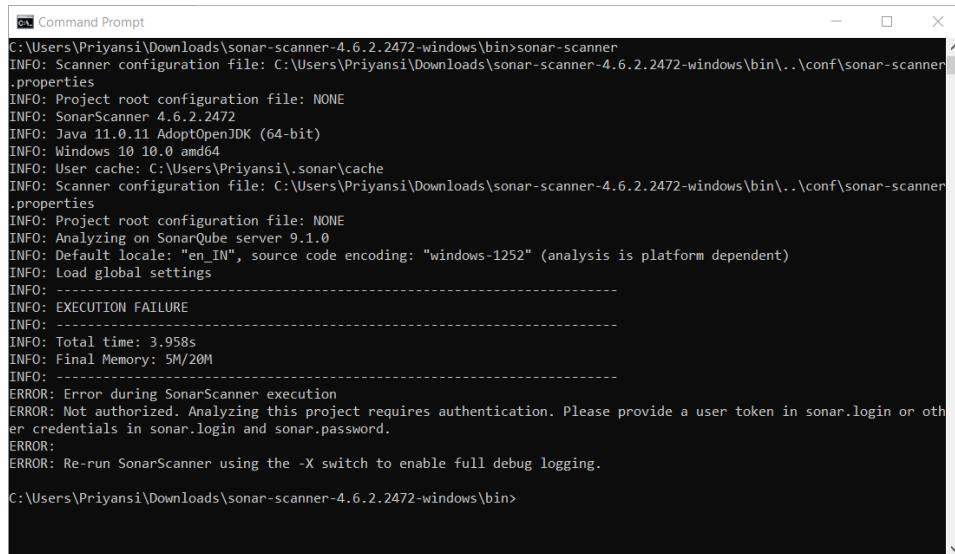
Bhuvan Sawant

T22

Roll No: 110

Open another command prompt. Run command:

- cd "sonar-scanner-4.6.2.2472-windows\bin"
- sonar-scanner

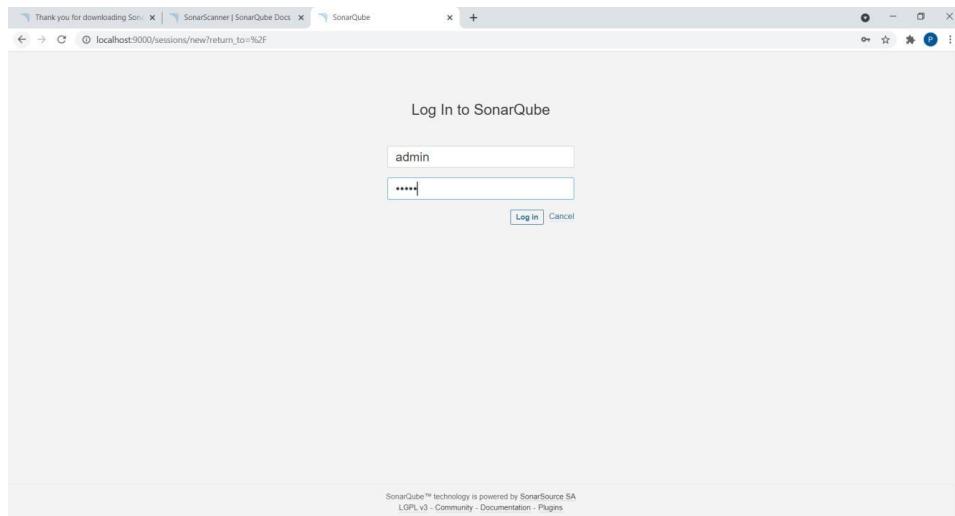


```
c:\ Command Prompt
C:\Users\Priyansi\Downloads\sonar-scanner-4.6.2.2472-windows\bin>sonar-scanner
INFO: Scanner configuration file: C:\Users\Priyansi\Downloads\sonar-scanner-4.6.2.2472-windows\bin..\conf\sonar-scanner.properties
INFO: Project root configuration file: NONE
INFO: SonarScanner 4.6.2.2472
INFO: Java 11.0.11 AdoptOpenJDK (64-bit)
INFO: Windows 10 10.0 amd64
INFO: User cache: C:\Users\Priyansi\.sonar\cache
INFO: Scanner configuration file: C:\Users\Priyansi\Downloads\sonar-scanner-4.6.2.2472-windows\bin..\conf\sonar-scanner.properties
INFO: Project root configuration file: NONE
INFO: Analyzing on SonarQube server 9.1.0
INFO: Default locale: "en_IN", source code encoding: "windows-1252" (analysis is platform dependent)
INFO: Load global settings
INFO: -----
INFO: EXECUTION FAILURE
INFO: -----
INFO: Total time: 3.958s
INFO: Final Memory: 5M/20M
INFO: -----
ERROR: Error during SonarScanner execution
ERROR: Not authorized. Analyzing this project requires authentication. Please provide a user token in sonar.login or other credentials in sonar.login and sonar.password.
ERROR:
ERROR: Re-run SonarScanner using the -X switch to enable full debug logging.

C:\Users\Priyansi\Downloads\sonar-scanner-4.6.2.2472-windows\bin>
```

Server up and running on **localhost:9000**

Login using credentials as User: admin and Password: admin and Set a new password



Bhuvan Sawant

T22

Roll No: 110

The screenshot shows the SonarQube interface for creating a new project. At the top, there are four options for importing from DevOps platforms: 'From Azure DevOps', 'From Bitbucket', 'From GitHub', and 'From GitLab'. Below these, a note says 'Are you just testing or have an advanced use-case? Create a project manually.' A 'Manually' button is highlighted with a yellow arrow. A warning message at the bottom states: 'Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.'

Click on Create a project **Manually**.

The screenshot shows the 'Create a project' page in manual mode. It requires entering a 'Project display name' (sonarPythonProgram) and a 'Project key' (sonarPythonProgram1). Both fields are highlighted with green arrows. A 'Set Up' button is visible. A warning message at the bottom states: 'Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.'

Give any Project display name.

Bhuvan Sawant

T22

Roll No: 110

The screenshot shows the SonarQube dashboard for the 'sonarPythonProgram1' project. At the top, there are links for 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', and 'Administration'. Below this, the project name 'sonarPythonProgram1' is shown with a master branch indicator. A search bar at the top right contains the placeholder 'Search for projects...'. On the left, there's a navigation menu with 'Overview' selected, followed by 'Issues', 'Security Hotspots', 'Measures', 'Code', and 'Activity'. To the right of the menu are 'Project Settings' and 'Project Information' buttons. The main content area has a heading 'How do you want to analyze your repository?'. It asks if you want to integrate with CI and provides links for Jenkins, GitHub Actions, Bitbucket Pipelines, GitLab CI, Azure Pipelines, and Other CI. Below this, a section titled 'Are you just testing or have an advanced use-case? Analyze your project locally' features a 'Locally' button with a local machine icon. A note at the bottom states: 'Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.'

Click on Locally.

The screenshot shows the SonarQube dashboard for the 'sonarPythonProgram1' project, now with the 'Locally' analysis option selected. The interface is similar to the previous screenshot, but the 'Locally' button is now active. In the main content area, there is a step-by-step guide: 'Provide a token' with a 'Generate' button and a text input field containing 'pythonToken!'. A note below explains: 'The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point of time in your user account.' There is also a 'Run analysis on your project' button. A note at the bottom states: 'Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.'

Give any name to token and click on **Generate**.

Bhuvan Sawant

T22

Roll No: 110

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

Provide a token
pythonToken1: 41740dddf269d68dffda1ec55f28cd250be45d48f ✓
The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point of time in your user account.

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 migrating your data out of it into a different database engine.

SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 9.1 (build 47739) - LGPL v3 - Community - Documentation - Plugins - Web API - About

Click on Continue.

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

Provide a token
pythonToken1: 41740dddf269d68dffda1ec55f28cd250be45d48f ✓
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 `sonar` directory to the `PATH` 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.login=41740dddf269d68dffda1ec55f28cd250be45d48f

Save a Python program in a folder.

```
class Solution(object):
    def romanToInt(self, s):
```

```

roman =
{'I':1,'V':5,'X':10,'L':50,'C':100,'D':500,'M':1000,'IV':4,'IX':9,'XL':40,'XC':90,'CD':400,'CM':900}
i = 0
num = " "
while i < len(s):
    if i+1<len(s) and s[i:i+2] in roman:
        num+=roman[s[i:i+2]]
        i+=2
    else:
        #print(i)
        num+=roman[s[i]]
        i+=1
return num
ob1 = Solution()
print(ob1.romanToInt("III"))
print(ob1.romanToInt("CDXLIII"))

```

Open command prompt in this folder and Run program using copied command.

```

sonar-scanner.bat -D"sonar.projectKey=<YourDisplayName>" -
-D"sonar.sources=." -D"sonar.host.url=http://localhost:9000" -
D"sonar.login=<YourTokenGeneratedID>"
```

```

C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19043.1237]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Prayansil\Documents\SonarExp\sonar-scanner.bat -D"sonar.projectKey=sonarPythonProgram" -D"sonar.sources=." -D"sonar.host.url=http://localhost:9000" -D"sonar.login=41740dddf269d68dfdatec55f28cd250be46dd4

INFO: Scanner configuration file: C:\Users\Prayansil\Downloads\sonar-scanner-4.6.2.2472-windows\bin..\conf\sonar-scanner.properties
INFO: Project root configuration file: NONE
INFO: SonarScanner 4.6.2.2472
INFO: Java 11.0.11 (Oracle Corporation 11.0.11+7-b150830030K (64-bit))
INFO: Windows 10 10.0 amd64
INFO: User cache: C:\Users\Prayansil\sonar\cache
INFO: Scanner configuration file: C:\Users\Prayansil\Downloads\sonar-scanner-4.6.2.2472-windows\bin..\conf\sonar-scanner.properties
INFO: Project root configuration file: NONE
INFO: Analyzing on SonarQube server 9.1.0
INFO: Default locale: "en_IN", source code encoding: "windows-1252" (analysis is platform dependent)
INFO: User cache: C:\Users\Prayansil\sonar\cache
INFO: Load global settings (done) | time=20ms
INFO: Server id: 8F4A1A1F2-4Xed09d09d18ce215Cu
INFO: User cache: C:\Users\Prayansil\sonar\cache
INFO: Load/download plugins
INFO: Load plugins index
INFO: Load plugins index (done) | time=10ms
INFO: Load active rules (done) | time=10ms
INFO: Process project properties
INFO: Process project properties (done) | time=20ms
INFO: Execute project builders (done) | time=2ms
INFO: Project key: 'sonarPythonProgram'
INFO: Base working dir: C:\Users\Prayansil\Documents\SonarExp
INFO: Scanning dir: C:\Users\Prayansil\Documents\SonarExp, scannerwork
INFO: Load project settings for component key: 'sonarPythonProgram'
INFO: Load project settings for component key: 'sonarPythonProgram' (done) | time=40ms
INFO: Load quality profiles (done) | time=20ms
INFO: Load active rules
INFO: Load active rules (done) | time=452ms
INFO: SCM provider auto-detection failed. Please use "sonar.scm.provider" to define SCM of your project, or disable the SCM Sensor in the project settings.
INFO: Indexing files...
INFO: Project configuration:
INFO: Quality profile: None
INFO: Quality profile for py: Sonar way
INFO: Quality profile for py: Run sensors on module sonarPythonProgram
INFO: Load metrics repository (done) | time=37ms
INFO: Load metrics repository (done) | time=37ms
INFO: Sensor Python Sensor [python]
INFO: Your code is analyzed and compatible with python 2 and 3 by default. This will prevent the detection of issues specific to python 2 or python 3. You can get a more precise analysis by setting a python version in the sonar.python.version parameter
INFO: Starting global symbols computation
INFO: 1 source file to be analyzed
INFO: Load project repositories

```

Commented [DS1]: sonar-scanner.bat -
 D"sonar.projectKey=<YourDisplayName>" -
 D"sonar.sources=." -
 D"sonar.host.url=http://localhost:9000" -
 D"sonar.login=<YourTokenGeneratedID>" Page 7
 command

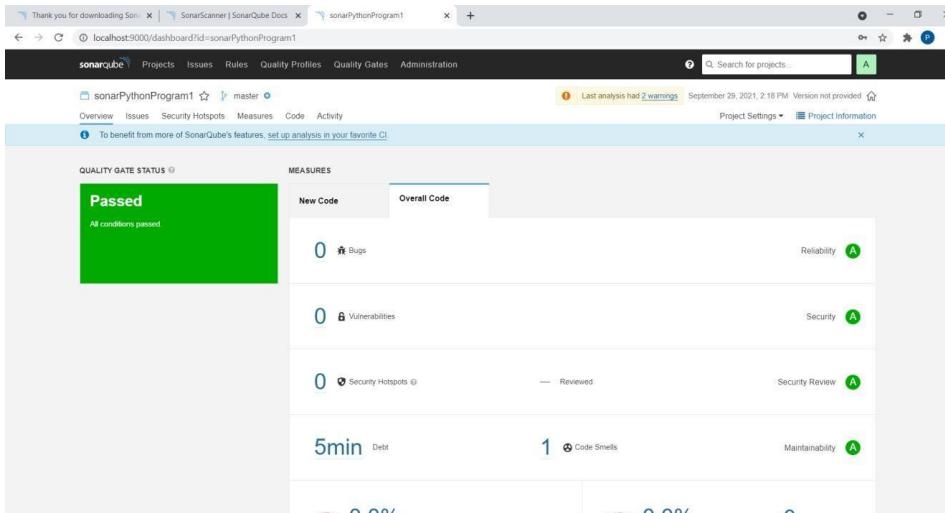
Bhuvan Sawant

T22

Roll No: 110

```
C:\Windows\System32\cmd.exe
INFO: Sensor HTML [web] (done) | time=2ms
INFO: Sensor VB.NET Project Type Information [vbnet]
INFO: Sensor VB.NET Project Type Information [vbnet] (done) | time=1ms
INFO: Sensor VB.NET Analysis Log [vbnet]
INFO: Sensor VB.NET Analysis Log [vbnet] (done) | time=12ms
INFO: Sensor VB.NET Properties [vbnet]
INFO: Sensor VB.NET Properties [vbnet] (done) | time=0ms
INFO: Sensor VB.NET Run Times [vbnet] (done) | time=0ms
INFO: ... Run sensors on project
INFO: Sensor Zero Coverage Sensor
INFO: Sensor Zero Coverage Sensor (done) | time=12ms
INFO: SCM Publisher No SCM system was detected. You can use the 'sonar.scm.provider' property to explicitly specify it.
INFO: CPD Executor Calculating CPD for 1 file
INFO: CPD Executor CPD calculation finished (0pp) | time=10ms
INFO: Analysis report compressed by zip file size=16.9 kB
INFO: Analysis report compressed in 7oms
INFO: Analysis report uploaded in 7oms
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=dAxwvYlhx9tBdxzLXH
INFO: Analysis total time: 7.582 s
INFO: EXECUTION SUCCESS
INFO: -----
INFO: Total time: 10.887s
INFO: Final Memory: 79/30M
INFO: -----
C:\Users\Priyanshi\Documents\SonarExp>
```

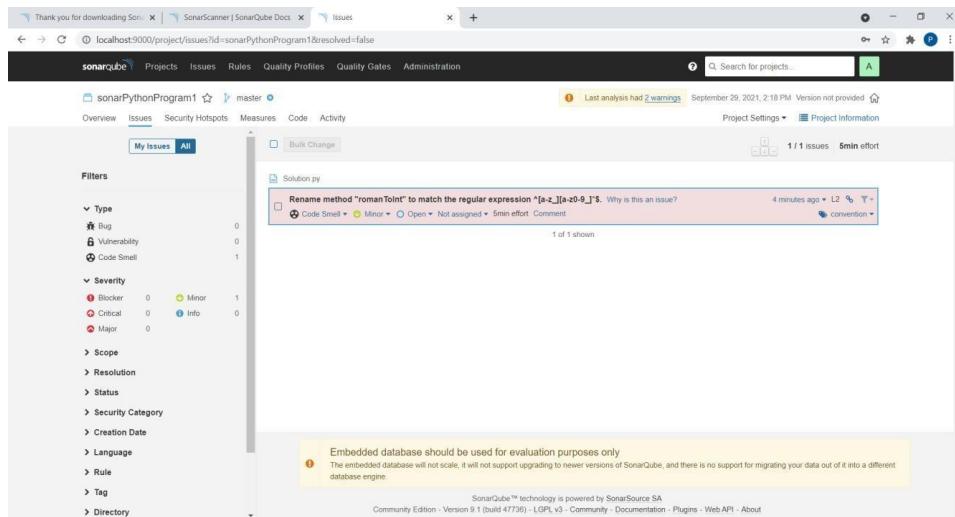
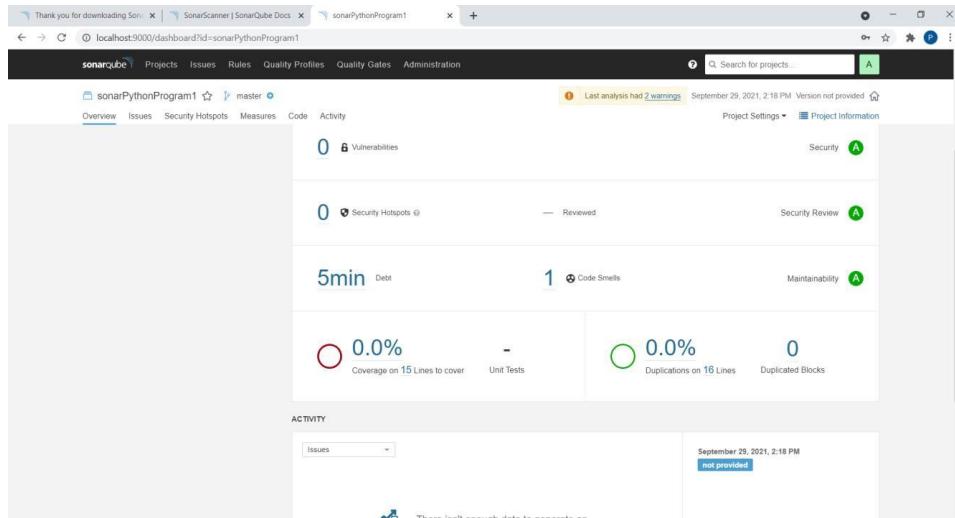
Given below is the inspection of code quality to perform automatic reviews with static analysis of code to detect bugs, code smells, and security vulnerabilities.



Bhuvan Sawant

T22

Roll No: 110



Press "Ctrl + C" to stop the server.

Bhuvan Sawant

T22

Roll No: 110

Conclusion:

In this assignment we implemented static analysis on python programs using SonarCube SAST processes.

Assignment 8

Name: Bhuvan Sawant Roll No.- 110 Branch: IT/T2

Date:17/10/2023

Aim: To understand Continuous Monitoring using Nagios.

LO Mapped: LO5

Theory:

Login to Aws.

Create Ec2 instances on Aws account of any linux os

Then Run the following command in SS

```
ubuntu@ip-172-31-13-219:~$ sudo apt update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [101 kB]
Get:5 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 Packages [8628 kB]
Get:6 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [909 kB]
Get:7 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal/universe Translation-en [5124 kB]
```

```
cd /usr/src/
```

```
sudo wget https://github.com/NagiosEnterprises/nagioscore/archive/nagios-4.4.2.tar.gz
```

```
ubuntu@ip-172-31-13-219:/usr/src$ sudo wget https://github.com/NagiosEnterprises/nagioscore/archive/nagios-4.4.2.tar.gz
--2021-10-05 10:24:05-  https://github.com/NagiosEnterprises/nagioscore/archive/nagios-4.4.2.tar.gz
Resolving github.com (github.com)... 13.234.176.102
Connecting to github.com (github.com)|13.234.176.102|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://code.load.github.com/NagiosEnterprises/nagioscore/tar.gz/nagios-4.4.2 [following]
--2021-10-05 10:24:05-  https://code.load.github.com/NagiosEnterprises/nagioscore/tar.gz/nagios-4.4.2
Resolving code.load.github.com (code.load.github.com)... 13.233.43.20
Connecting to code.load.github.com (code.load.github.com)|13.233.43.20|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/x-gzip]
Saving to: 'nagios-4.4.2.tar.gz'

nagios-4.4.2.tar.gz          [ ==>                               ]  10.78M  16.9MB/s   in 0.6s
2021-10-05 10:24:06 (16.9 MB/s) - 'nagios-4.4.2.tar.gz' saved [11301457]
```

```
sudo tar zxf nagios-*.tar.gz
```

```
cd nagioscore-nagios-*/
```

```
ubuntu@ip-172-31-13-219:/usr/src$ cd nagioscore-nagios-*/
ubuntu@ip-172-31-13-219:/usr/src/nagioscore-nagios-4.4.2$
```

Now finally run the following command

```
sudo ./configure --with-httpd-conf=/etc/apache2/sites-enabled
```

if error comes install c compiler on the linux

by following this link

<https://linuxize.com/post/how-to-install-gcc-compiler-on-ubuntu-18-04/>

Finally-

```
4. Rerun the configure script.

NOTE: If you can't get the configure script to recognize the GD libs
on your system, get over it and move on to other things. The
CGIs that use the GD libs are just a small part of the entire
Nagios package. Get everything else working first and then
revisit the problem. Make sure to check the nagios-users
mailing list archives for possible solutions to GD library
problems when you resume your troubleshooting.

*****
checking ltdl.h usability... no
checking ltdl.h presence... no
checking for ltdl.h... no
checking dlfcn.h usability... yes
checking dlfcn.h presence... yes
checking for dlfcn.h... yes
checking for dlopen in -ldl... yes
checking for extra flags needed to export symbols... -Wl,-export-dynamic
checking for linker flags for loadable modules... -shared
checking for traceroute... no
checking for type va_list... yes
checking for perl... /usr/bin/perl
checking for unzip... no
```

Now let us install plugins by

```
sudo wget -O nagios-plugins.tar.gz https://github.com/nagios-plugins/nagios-plugins/archive/release-2.2.1.tar.g
```

then

```
sudo tar zxf nagios-plugins.tar.gz
```

then

```
cd nagios-plugins-release-2.2.1
```

```
finally start and then check status of nagi os
```

```
sudo systemctl start nagios
```

```
sudo systemctl status nagios
```

```
* nagios.service - Nagios Core 4.4.2
   Loaded: loaded (/lib/systemd/system/nagios.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2018-11-16 14:54:21 PST; 1s ago
     Docs: https://www.nagios.org/documentation
   Process: 18294 ExecStopPost=/bin/rm -f /usr/local/nagios/var/rw/nagios.cmd (code=exited, status=0/SUCCESS)
   Process: 18293 ExecStop=/bin/kill -s TERM ${MAINPID} (code=exited, status=0/SUCCESS)
   Process: 18315 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Process: 18313 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
 Main PID: 18325 (nagios)
    Tasks: 6 (limit: 2319)
   CGroup: /system.slice/nagios.service
```

Steps-

Go to google.com, Search NagiOs Demo

Click on the first link shown below

Google

nagios demo

All Videos Images News Shopping More Tools

About 3,87,000 results (0.44 seconds)

<https://exchange.nagios.org> › directory › Demos › details

Nagios XI Online Demo

An online **demo** of Nagios XI. The **demo** allows you to test configuration wizards, dashlets, dashboards, views, and more. Reviews (0).

<https://exchange.nagios.org> › directory › Demos

Demos - Nagios Exchange

An online **demo** of Nagios Log Server. The **demo** allows you to view system logs and event logs, giving some examples on how you can visualize data sent into Nagios ...

<https://exchange.nagios.org> › directory › Demos › details

Now click on the website-

Nagios®

Network: | Enterprise | Support | Library | Project | Exchan

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...
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

The screenshot shows the Nagios XI login interface on the left, featuring fields for Username and Password, a 'Login' button, and a 'Forgot your password?' link. Below it is a 'Select Language:' dropdown with various flags. To the right is a banner for 'Nagios XI Demo System'. Underneath the banner, the 'Demo Account Options' section lists several user access levels with their respective log-in links and credentials:

- Administrator Access**: Log in as Administrator (Username: nagiosadmin, Password: nagiosadmin)
- Read-Only User Access**: Log in as Read-Only User (Username: readonly, Password: readonly)
- Advanced User Access**: Log in as Advanced User (Username: advanced, Password: advanced)
- Normal User Access**: Log in as Normal User (Username: jdoe, Password: jdoe)
- Administrator Access** (dark theme): Log in as Administrator (Username: darktheme, Password: darktheme)

It will have interface like this

The screenshot displays the Nagios XI Home Dashboard. The top navigation bar includes links for Home, Views, Dashboards, Reports, Configure, Tools, Help, and Admin. The dashboard features several summary cards:

- Getting Started Guide**: Lists common tasks such as changing account settings, notifications, and monitoring setup.
- Host Status Summary**: A grid showing the count of hosts in Up, Down, Unreachable, and Pending states. Last updated: 2021-10-05 05:06:48.

| | Up | Down | Unreachable | Pending |
|-----------|-----|------|-------------|---------|
| Up | 764 | 100 | 0 | 6 |
| Unhandled | 3 | 3 | 59 | All |
- Service Status Summary**: A grid showing the count of services in Ok, Warning, Unknown, Critical, and Pending states. Last updated: 2021-10-05 05:06:48.

| | Ok | Warning | Unknown | Critical | Pending |
|-----------|-----|---------|---------|----------|---------|
| Ok | 764 | 100 | 0 | 6 | 7 |
| Unhandled | 236 | 236 | 0 | 0 | 1007 |
- Administrative Tasks**: Initial Setup Tasks: Configure mail settings.
- We're Here To Help!**: A section with a photo of a support technician and links to Support Forum, Help Resources, Customer Ticket Support Center, and Customer Phone Support.
- Start Monitoring**: Buttons for Run a Config Wizard, Run Auto-Discovery, and Advanced Config.

On the left sidebar, there are sections for Quick View (Home Dashboard, Tactical Overview, Birdseye, Operations Center, Operations Screen, Open Service Problems, Open Host Problems, All Service Problems, All Host Problems, Network Outages), Details (Service Status, Host Status, Hostgroup Summary, Hostgroup Overview, Hostgroup Grid, Servicegroup Summary, Servicegroup Overview, Servicegroup Grid, BPI, Metrics), Graphs (Performance Graphs, Graph Explorer), Maps (World Map, BMap, Hypermap, Minimap, NagVis, Network Status Map), and Incident Management.

Now click on Host status-

The screenshot shows the Nagios XI interface. On the left, a sidebar contains navigation links for Home Dashboard, Quick View, Details, Graphs, Maps, and Incident Management. The main content area displays the 'Host Status' summary. It includes two summary tables: 'Host Status Summary' and 'Service Status Summary'. The 'Host Status Summary' table shows counts for Up (66), Down (3), Unreachable (0), Pending (6), Unhandled (3), Problems (3), and All (59). The 'Service Status Summary' table shows counts for Ok (764), Warning (189), Unknown (5), Critical (131), Pending (7), Unhandled (236), Problems (236), and All (1007). Below these summaries is a detailed table of hosts, each with a status icon, name, status, duration, attempt, last check, and status information. The table lists various hosts like europa.nagios.local, www.acme.com, www.chaoticmoon.com, Firewall, Log-Server, NOAA, Netw, Network-Analyzer, and Router, along with their respective statuses and details.

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

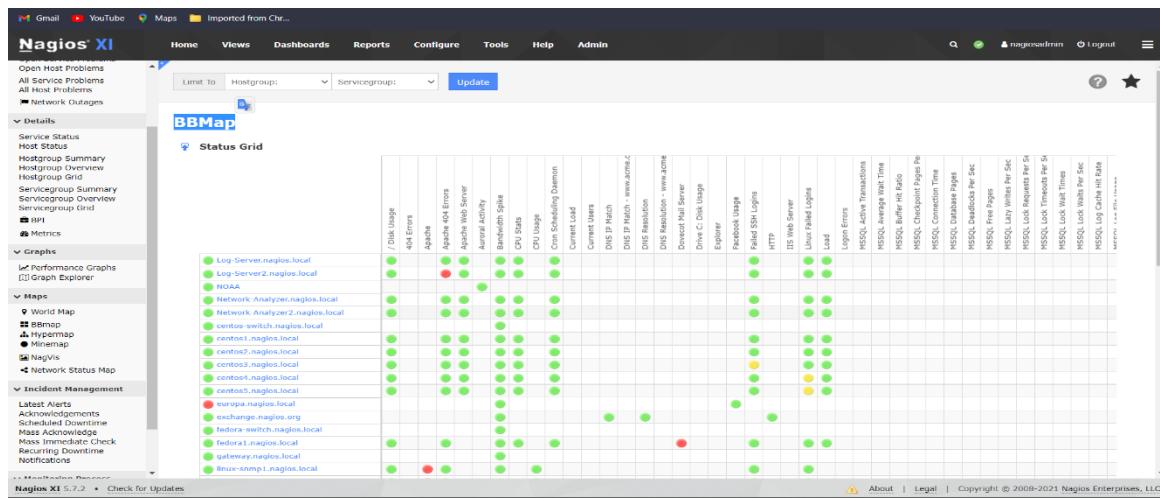
Now click on Host Group Status.

The screenshot shows the Nagios XI interface with the 'Host Group Status' summary selected. The left sidebar remains the same. The main content area displays the 'Host Group Status' summary. It includes two summary tables: 'Host Status Summary' and 'Service Status Summary', which are identical to the ones in the previous screenshot. Below these is a detailed table titled 'Status Summary For All Host Groups'. This table lists various host groups with their status, number of hosts, and services. For example, 'Monitoring Servers (Monitoring Servers)' has 5 hosts, 93 OK, 14 Warning, 2 Unknown, and 2 Critical. 'Hostgroup Two (hg2)' has 11 hosts, 11 OK, 4 Warning, 2 Unknown, and 4 Critical. 'Some Other Hostgroup (hg3)' has 11 hosts, 11 OK, 1 Warning, and 2 Unknown. 'Linux Servers (linux-servers)' has 31 hosts, 218 OK, 27 Warning, 2 Unknown, and 15 Critical. 'Network Devices (network-devices)' has 7 hosts, 215 OK, 37 Warning, and 6 Critical.

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: In this assignment we learned about how to continuously monitor Nagios commands

ASSIGNMENT 9

AIM: To understand Lambda Function and create a Lambda function which will log "An Image has been added" once you add an object to a specific bucket in S3.

LO MAPPED: LO1, LO5

THEORY:

You can use Lambda to process event notifications from Amazon Simple Storage Service. Amazon S3 can send an event to a Lambda function when an object is created or deleted. You configure notification settings on a bucket, and grant Amazon S3 permission to invoke a function on the function's resource-based permissions policy.

STEPS TO FOLLOW:

1. Log in as IAM User

2. Create a S3 bucket and Enable the "Block all public access"

The screenshot shows the AWS S3 console with a green header bar indicating "Successfully created bucket *labdabucket117*". Below the header, there's an "Account snapshot" section with a "View Storage Lens dashboard" button. The main area displays a table titled "Buckets (1) Info" with one entry: "labdabucket117" located in "Europe (Stockholm) eu-north-1" with "Objects can be public" access and created on "August 30, 2023, 22:56:19 (UTC-07:00)". Buttons for "Create bucket" and other actions like "Copy ARN", "Empty", and "Delete" are visible.

3. Search IAM on the console and go to "Roles". Click on Create Roles

The screenshot shows the AWS IAM console with a sidebar for "Identity and Access Management (IAM)". Under "Access management", "Roles" is selected. The main area shows a table of existing roles: "AWSCloud9SSMAccessRole", "AWSServiceRoleForAWSCloud9", "AWSServiceRoleForSupport", and "AWSServiceRoleForTrustedAdvisor". A "Create role" button is at the top right. Below the table is a section titled "Roles Anywhere" with a "Manage" button.

4. Select the options of "AWS service" and "lambda"

The screenshot shows the "Select trusted entity" wizard, Step 2: "Trusted entity type". It shows five options: "AWS service" (selected), "AWS account", "Web identity", "SAML 2.0 federation", and "Custom trust policy". Below this is a "Use case" section with "Common use cases": "EC2" and "Lambda" (selected). A "Choose a service to view use case" dropdown is also present.

5. Enable the "CloudWatchFullAccess" and "AmazonS3FullAccess"

Permissions policies (Selected 2/874) Info
Choose one or more policies to attach to your new role.

| Policy name | Type | Description |
|-----------------------------|-------------|--|
| AWSCloudFormationFullAccess | AWS managed | Provides access to AWS CloudFormation via the AWS Management Console. |
| CloudFrontFullAccess | AWS managed | Provides full access to the CloudFront console plus the ability to list Amazon CloudFront distributions. |
| AWSCloudHSMFullAccess | AWS managed | Provides full access to all CloudHSM resources. |
| AWSCloudHSMReadOnlyAccess | AWS managed | Provides read only access to all CloudHSM resources. |
| CloudFrontReadOnlyAccess | AWS managed | Provides access to CloudFront distribution configuration information and its objects. |
| CloudSearchFullAccess | AWS managed | Provides full access to the Amazon CloudSearch configuration service. |
| CloudSearchReadOnlyAccess | AWS managed | Provides read only access to the Amazon CloudSearch configuration service. |
| CloudWatchFullAccess | AWS managed | Provides full access to CloudWatch. |

Add permissions Info

Permissions policies (Selected 2/874) Info
Choose one or more policies to attach to your new role.

| Policy name | Type | Description |
|--------------------------------|-------------|---|
| AmazonS3FullAccess | AWS managed | Provides full access to all buckets via the AWS Management Console. |
| AmazonS3ReadOnlyAccess | AWS managed | Provides read only access to all buckets via the AWS Management Console. |
| AmazonDMSRedshiftFullAccess | AWS managed | Provides access to manage S3 settings for Redshift endpoints for DMS. |
| QuickSightAccessForAllAccounts | AWS managed | Policy used by QuickSight team to access customer data produced by S3 and Redshift. |
| AmazonS3OutpostsFullAccess | AWS managed | Provides full access to Amazon S3 on Outposts via the AWS Management Console. |
| AmazonS3OutpostsReadOnlyAccess | AWS managed | Provides read only access to Amazon S3 on Outposts via the AWS Management Console. |

6. Click on Create Role and you will be redirected to this dashboard. Give the Role a Name and Click on Done.

Name, review, and create

Role details

Role name
Enter a meaningful name to identify this role.
LambdaUser

Description
Add a short explanation for this role.
Allows Lambda functions to call AWS services on your behalf.

Step 1: Select trusted entities

1 "Version": "2012-10-17",
2

7. Search for Lambda in the console and click on Create Function.

The screenshot shows the AWS Lambda homepage. The main heading is "AWS Lambda" with the tagline "lets you run code without thinking about servers." Below this, a paragraph explains that users pay only for compute time consumed and can run code for various applications. To the right, a "Get started" box contains the text "Author a Lambda function from scratch, or choose from one of many preconfigured examples." A prominent orange "Create a function" button is at the bottom of this box. Below the main heading, there's a "How it works" section with tabs for ".NET", "Go", "Java", "Node.js" (which is selected), "Python", "Ruby", and "Custom runtime". A sample code snippet for Node.js is shown:

```
1 exports.handler = async (event) => {
```

The URL in the browser bar is <https://eu-north-1.console.aws.amazon.com/lambda/home?region=eu-north-1#/create/function?firstrun=true>.

8. Change the settings of the Lambda Function.

The screenshot shows the "Create function" wizard. It starts with a "Choose one of the following options to create your function." section with four options: "Author from scratch" (selected), "Use a blueprint", "Container image", and "Browse serverless app repository". Below this is a "Basic information" section where the function name is set to "LambdaFunctionUser". The "Runtime" is chosen as "Python 3.11". Under "Architecture", "x86_64" is selected. In the "Permissions" section, "Change default execution role" is selected, and "Use an existing role" is chosen. At the bottom, the URL is <https://eu-north-1.console.aws.amazon.com/lambda/CreateFunction?functionName=LambdaFunctionUser®ion=eu-north-1>.

9. Add the python code and click on deploy to save the changes.

The screenshot shows the Lambda function editor for the "LambdaFunctionUser" function. The "Code" tab is selected. At the top, a message says "Successfully created the function LambdaFunctionUser. You can now change its code and configuration. To invoke your function with a test event, choose "Test". The "Test" button is highlighted. Below this is the "Code source" tab with the "Info" sub-tab selected. The code editor shows a single file named "lambda_function.py" with the following content:

```
1 import json
2
3 def lambda_handler(event, context):
4     # TODO implement
5     print("Image successfully uploaded in s3 bucket")
```

The "Upload from" button is visible at the top right of the code editor. The URL in the browser bar is <https://eu-north-1.console.aws.amazon.com/lambda/CodeEditor?functionName=LambdaFunctionUser®ion=eu-north-1>.

10. Scroll above and click on ADD TRIGGER. Select the following options and click on Done.

11. Go to S3 bucket and click on Add files. Select a image and click on Upload.

The screenshot shows the AWS CloudWatch 'Upload: status' page. At the top, a green header bar indicates 'Upload succeeded' with a link to 'View details below.' Below this, a message box states 'The information below will no longer be available after you navigate away from this page.' The main section is titled 'Summary' and shows the following details:

| | | |
|-------------|---------------------------------|-----------------------------|
| Destination | Succeeded s3://labdbucket117 | Failed 0 files, 0 B (0%) |
|-------------|---------------------------------|-----------------------------|

Below the summary, there are two tabs: 'Files and folders' (selected) and 'Configuration'. Under 'Files and folders', it says '(1 Total, 266.2 KB)' and lists one item: '/aws/lambda/LambdaFunctionUser'. The bottom of the page includes standard AWS navigation links: CloudShell, Feedback, Language, and links to 2023, Privacy, Terms, and Cookie preferences.

12. Search CloudWatch and go to Log groups. Select the existing Log Group.

The screenshot shows the AWS CloudWatch 'Log groups' page. The left sidebar has 'Logs' expanded, with 'Log groups' selected. The main area displays a table of log groups:

| Log group | Data pr... | Sensitiv... | Retention | Metric ... |
|--------------------------------|------------|-------------|--------------|------------|
| /aws/lambda/LambdaFunctionUser | - | - | Never expire | - |

At the top right, there are buttons for 'Actions', 'View in Logs Insights', 'Start tailing', and 'Create log group'. A search bar at the top allows filtering by log group prefix. The bottom of the page includes standard AWS navigation links: CloudShell, Feedback, Language, and links to 2023, Privacy, Terms, and Cookie preferences.

13. Click on the link provided and then you will see the message displayed.

The screenshot shows the AWS CloudWatch 'Log streams' page. The left sidebar has 'Logs' expanded, with 'Log streams' selected. The main area displays a table of log streams:

| Log stream | Last event time |
|--|---------------------------------|
| 2023/08/31/[\$LATEST]97bf6ca2f8dd4cf2b383b55caf77... | 2023-08-30 23:08:48 (UTC-07:00) |

At the top right, there are buttons for 'Actions', 'Delete', 'Create log stream', and 'Search all log streams'. A search bar at the top allows filtering by log stream prefix. The bottom of the page includes standard AWS navigation links: CloudShell, Feedback, Language, and links to 2023, Privacy, Terms, and Cookie preferences.

The screenshot shows the AWS CloudWatch Log Events interface. The left sidebar navigation includes 'CloudWatch' (selected), 'Favorites and recents', 'Dashboards', 'Alarms', 'Logs' (selected), 'Log groups' (highlighted in orange), 'Live Tail' (New), 'Logs Insights', 'Metrics', 'X-Ray traces', 'Events', 'Application monitoring', and 'Insights'. The main content area displays 'Log events' for the log group '/aws/lambda/LambdaFunctionUser' at the timestamp '2023/08/31/[LATEST]97bf6ca2f8dd4cf2b383b55caf775fe3'. The log entries are:

| Timestamp | Message |
|-------------------------------|--|
| 2023-08-30T23:08:48.141-07:00 | INIT_START Runtime Version: python:3.11.v10 Runtime Version ARN: arn:aws:lambda:... |
| 2023-08-30T23:08:48.249-07:00 | START RequestId: d87be3bc-640d-4d9a-864d-2d10c71bc4a9 Version: \$LATEST |
| 2023-08-30T23:08:48.249-07:00 | Image successfully uploaded in s3 bucket |
| 2023-08-30T23:08:48.250-07:00 | END RequestId: d87be3bc-640d-4d9a-864d-2d10c71bc4a9 |
| 2023-08-30T23:08:48.250-07:00 | REPORT RequestId: d87be3bc-640d-4d9a-864d-2d10c71bc4a9 Duration: 1.51 ms Billed D... |

At the bottom, it says 'No newer events at this moment. Auto retry paused. Resume'.

CONCLUSION: In this assignment, we learnt how to create a Lambda function which will log "An Image has been added" once you add an object to a specific bucket in S3.

Assignment Number 10

Name: Bhuvan Sawant; Branch: I.T (T.E.); Roll Number: 110

17/10/2023

Aim: To create a Lambda function using Python for adding data to Dynamo DB database.

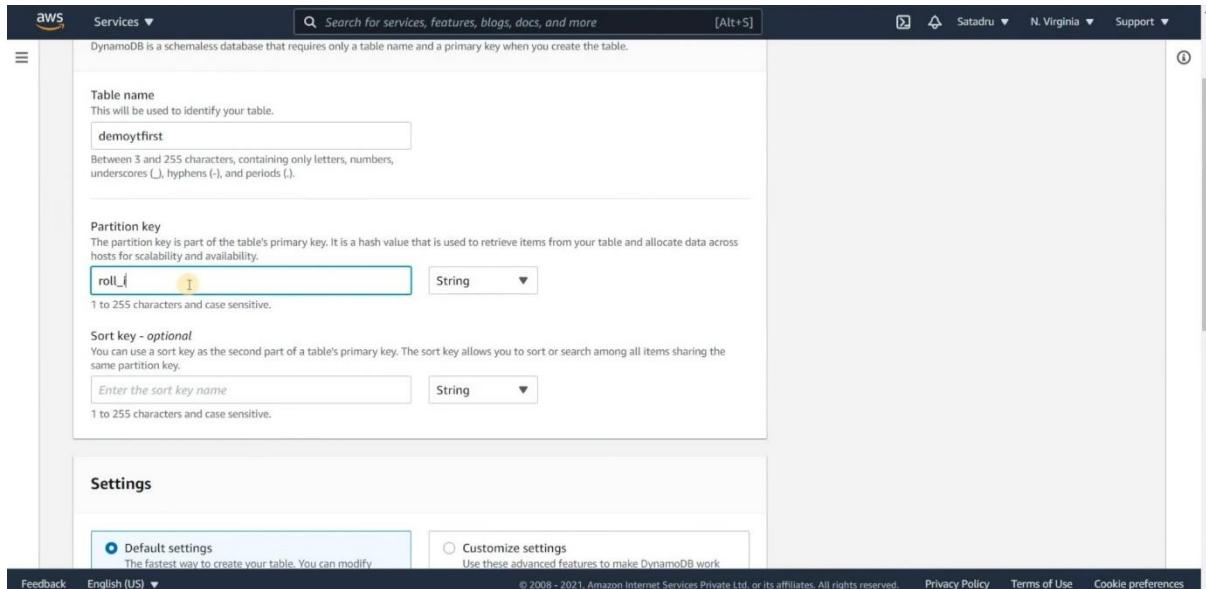
LO mapped: LO6

Theory:

Step 1: Set Up AWS Environment

1. **Create an AWS Account:** If you don't have an AWS account, sign up for one at [AWS Console](#).

2. **Access AWS Lambda Console:**



- Go to the [AWS Lambda Console](#).
- Click on "Create function."

3. Create a New Lambda Function

The screenshot shows two overlapping AWS console pages. The top page is the 'Tables' section of the DynamoDB service, displaying a single table named 'demoytfirst' with a status of 'Active'. The bottom page is the 'Create role' step in the Lambda service, specifically the 'Attach permissions policies' sub-step. It lists several AWS managed policies, with 'AmazonDynamoDBFullAccess' selected. Navigation buttons at the top of the Lambda page indicate this is step 2 of 4.

| Name | Status | Partition key | Sort key | Indexes | Read capacity mode | Write capacity mode |
|-------------|--------|------------------|----------|---------|-----------------------------------|-----------------------------------|
| demoytfirst | Active | roll_no (Number) | - | 0 | Provisioned with auto scaling (5) | Provisioned with auto scaling (5) |

| Policy name | Used as |
|---|------------------------|
| <input checked="" type="checkbox"/> AmazonDynamoDBFullAccess | None |
| <input type="checkbox"/> AmazonDynamoDBReadOnlyAccess | None |
| <input type="checkbox"/> AWSApplicationAutoScalingDynamoDBTablePolicy | Permissions policy (1) |
| <input type="checkbox"/> AWSLambdaDynamoDBExecutionRole | None |
| <input type="checkbox"/> AWSLambdaInvocation-DynamoDB | None |
| <input type="checkbox"/> DynamoDBCloudWatchContributorInsightsServiceRolePolicy | None |
| <input type="checkbox"/> DynamoDBKinesisReplicationServiceRolePolicy | None |
| <input type="checkbox"/> DynamoDBReplicationServiceRolePolicy | None |

- Choose "Author from scratch."
- Enter a name for your function (e.g., **AddToDynamoDBFunction**).
- Choose "Python" as the runtime.

4. Configure Execution Role

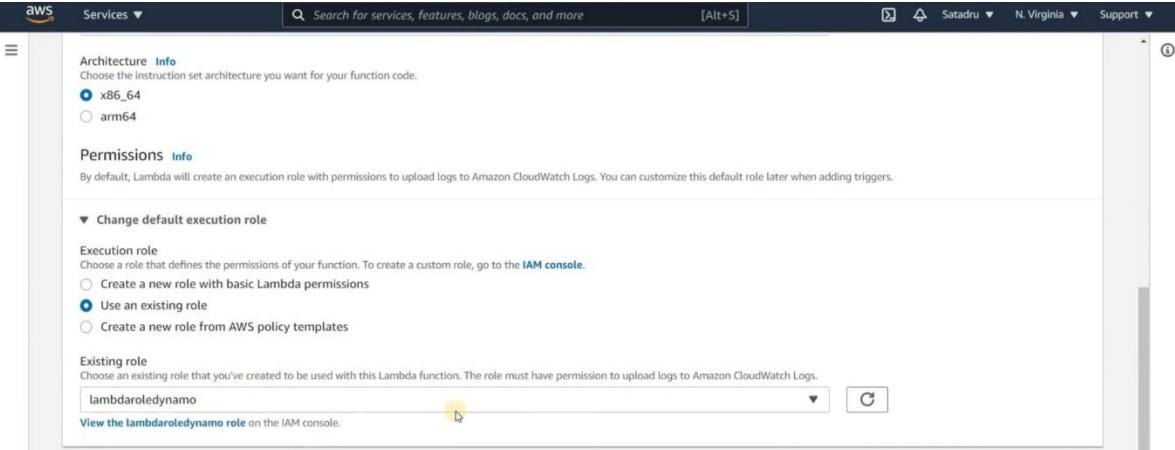
The screenshot shows the 'Create role' wizard on the AWS IAM service. The current step is 'Attach permissions policies'. A search bar at the top right is set to 'Cloudwatch'. Below it, a table lists various AWS CloudWatch policies. One policy, 'CloudWatchFullAccess', has a checked checkbox and is highlighted with a blue background. Other policies listed include CloudWatchEventsInvocationAccess, CloudWatchEventsReadOnlyAccess, CloudWatchEventsServiceRolePolicy, CloudWatchLambdaInsightsExecutionRolePolicy, CloudWatchLogsFullAccess, CloudWatchLogsReadOnlyAccess, and CloudWatchReadOnlyAccess. The table has columns for 'Policy name' and 'Used as'. At the bottom of the table, there are buttons for 'Cancel', 'Previous', 'Next: Tags', and 'Create policy'.

- Create a new role with basic Lambda permissions and additional permissions to interact with DynamoDB.
- Attach the role to your Lambda function.

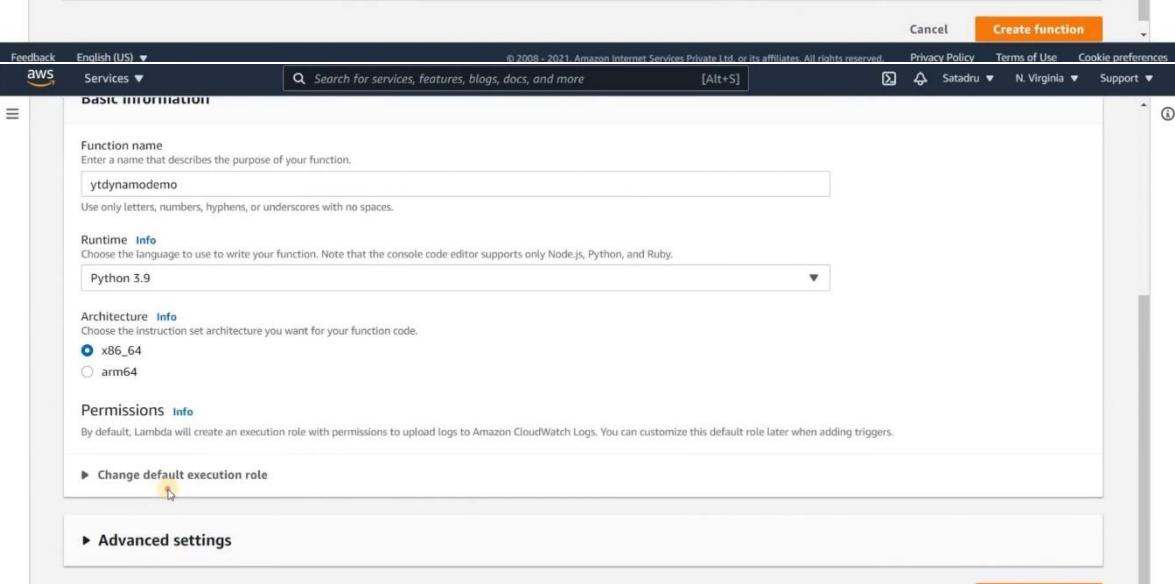
Step 2: Set Up DynamoDB

1. Access DynamoDB Console:

- Go to the [AWS DynamoDB Console](#).



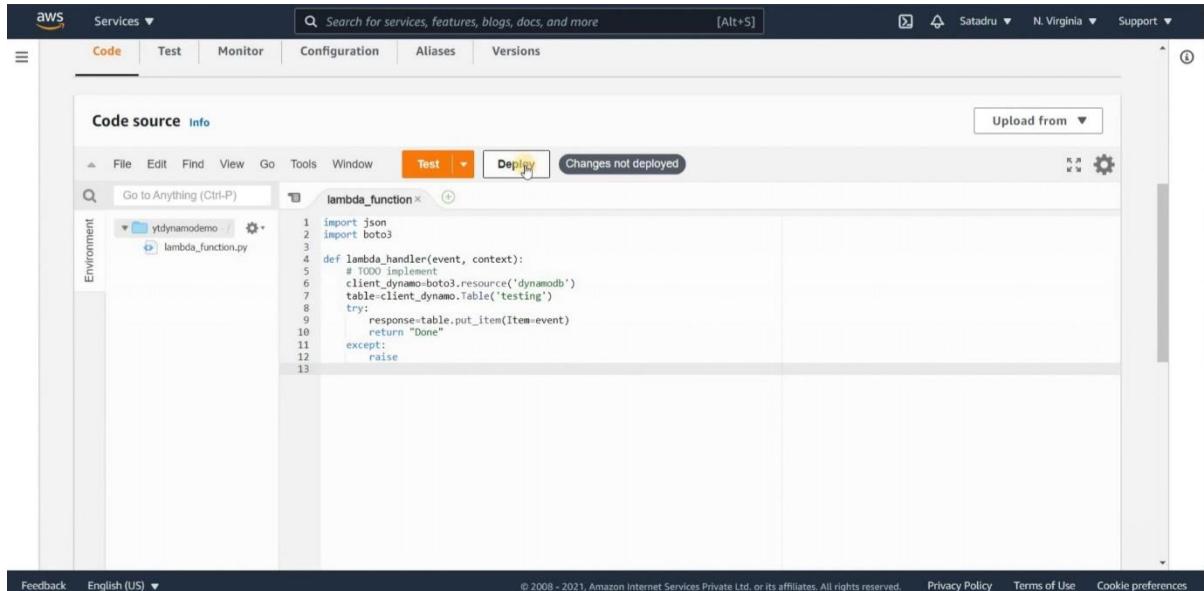
The screenshot shows the AWS Lambda function configuration interface. In the 'Permissions' section, there is a dropdown menu with the value 'lambdafunctionrole'. A yellow box highlights this dropdown, indicating it needs to be changed to 'lambdafunctiondynamodb'.



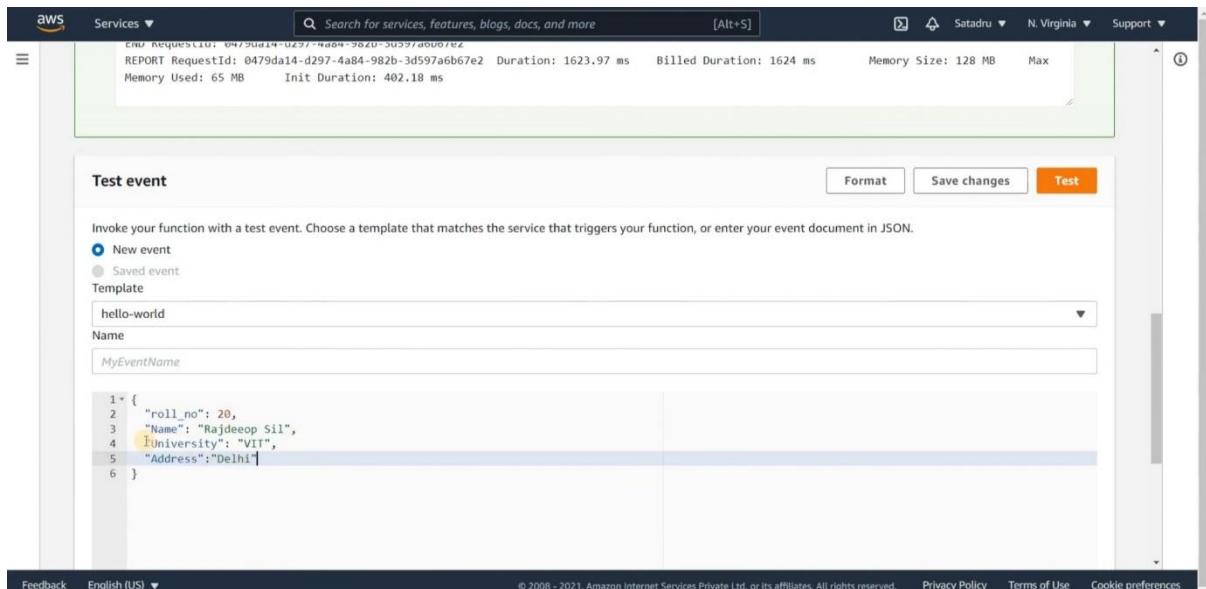
The screenshot shows the AWS Lambda function configuration interface. In the 'Runtime' section, there is a dropdown menu with the value 'Python 3.9'. A yellow box highlights this dropdown, indicating it needs to be changed to 'Python 3.8'.

2. Create a DynamoDB Table:

- Click on "Create table."
- Enter a table name and a primary key (e.g., ID).
- Configure other settings as needed and create the table.



Step 3: Write Lambda Function Code



1. Ensure to replace '**YourDynamoDBTableName**' with the actual name of your DynamoDB table.

2. Configure Lambda Handler:

- In the Lambda function configuration, set the handler to **filename.lambda_handler** (e.g., **yourfilename.lambda_handler**).

Step 4: Configure Environment Variables

1. Add DynamoDB Table Name:

- In the Lambda function configuration, add an environment variable (e.g., **DYNAMODB_TABLE**) with the value set to your DynamoDB table name.

Step 5: Test Locally

1. Test Lambda Function Locally:

- Create a test event with sample data.
- Test the Lambda function using the "Test" button in the Lambda console.

Step 6: Deploy Lambda Function

1. Deploy the Lambda Function:

- Click on the "Deploy" button in the Lambda console.

Step 7: Test in AWS Lambda Console

1. Test in AWS Lambda Console:

- Use the Lambda console to test the function by creating a test event with sample data.
- Verify that the function executes successfully.

Step 8: Monitor and Troubleshoot

1. Check CloudWatch Logs:

- Use CloudWatch Logs to check logs generated by your Lambda function for troubleshooting.

Step 9: Documentation

1. Create Documentation:

- Document the purpose, input parameters, and expected output of your Lambda function.
- Explain any challenges faced during development and how they were resolved.

By following these steps, you can create a Lambda function using Python to add data to a DynamoDB database in the AWS environment.

The screenshot shows the AWS DynamoDB console interface. On the left, there's a sidebar with options like Dashboard, Tables, and DAX. The main area shows a table named 'demoyfirst'. The table has columns: roll_no, Address, Name, and University. There are two items listed: one with roll_no 10, Name Soham, and University KIT; and another with roll_no 20, Name Rajdeep Sil, and University VIT. The 'Completed' status is shown at the bottom left of the table area.

Conclusion: By this assignment we learn how to create a Lambda function using Python for adding data to Dynamo DB database.

Adv. DevOps Written Assignment: 1

1. what security 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.

Combining these measures into a comprehensive security strategy is essential for safeguarding your Kubernetes cluster and the applications running within it. It's important to stay informed about best practices and evolving security threats in the Kubernetes ecosystem.

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:

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

2. 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.

3. 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.

These techniques are often used in combination to create a layered approach to data security, providing multiple levels of protection to safeguard sensitive information from unauthorized access and disclosure.

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

To expose a service using Ingress in Kubernetes, you need to follow these steps:

1. Set up Kubernetes: Ensure you have a Kubernetes cluster up and running, and you have the 'kubectl' command-line tool configured to communicate with the cluster.

2. Deploy Your Application: Deploy your application as a Kubernetes Deployment or a Pod, and create a Kubernetes Service to expose it internally within the cluster. This Service will be the target for the Ingress.

3. Install an Ingress Controller: You need to have an Ingress controller installed in your cluster. Some popular options include Nginx Ingress Controller, Traefik, or HAProxy Ingress. The controller will manage the Ingress resources and configure the load balancer.

For example, to install the Nginx Ingress Controller, you can use:

```
```bash
```

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.0.0/deploy/static/provider/cloud/deploy.yaml
```

```
```
```

4. Create an Ingress Resource: Define an Ingress resource that specifies the rules for routing traffic to your service. Here's an example Ingress resource manifest:

```
```yaml
```

```
apiVersion: networking.k8s.io/v1
```

```
kind: Ingress
```

```
metadata:
```

```
name: my-ingress
```

```
spec:
```

```
rules:
```

```
- host: example.com
```

```
 http:
```

```
 paths:
```

```
 - path: /path
```

```
 pathType: Prefix
```

```
 backend:
```

```
 service:
```

```
 name: your-service
```

```
 port:
```

```
 number: 80
```

```
...
```

In this example, traffic for `example.com/path` will be routed to `your-service`.

**5. Apply the Ingress Resource:** Use `kubectl apply` to create the Ingress resource in your cluster:

```
```bash
```

```
kubectl apply -f your-ingress.yaml
```

```
...
```

6. Configure DNS: Ensure that the DNS records for the specified hostname (e.g., `example.com`) point to the external IP address of your Ingress controller.

7. Access Your Service: After DNS propagation, you should be able to access your service externally via the hostname and path you defined in the Ingress resource.

4. Which service protocols does Kubernetes ingress expose?

Kubernetes Ingress is primarily designed to expose HTTP and HTTPS services, making it suitable for routing and load balancing web traffic. However, with the evolution of Kubernetes and Ingress controllers, it has expanded to support additional protocols and features:

1. HTTP: Ingress is commonly used to expose HTTP services. You can define routing rules based on URL paths, hostnames, and other HTTP attributes.

2. HTTPS: Secure HTTP services can be exposed through Ingress by configuring TLS certificates. This allows you to terminate SSL/TLS encryption at the Ingress controller and route decrypted traffic to your services.

3. TCP: Some Ingress controllers, like Nginx Ingress, support TCP services. This enables you to expose non-HTTP services such as databases or custom protocols. TCP-based routing typically relies on port numbers.

4. UDP: While less common, some Ingress controllers support UDP services. UDP is a connectionless protocol used for various purposes, including DNS and VoIP. Exposing UDP services may require specific controller support.

5. gRPC: If your services use the gRPC protocol, you can configure Ingress resources to handle gRPC traffic. gRPC is a high-performance RPC (Remote Procedure Call) framework often used for communication between microservices.

6. WebSocket: Ingress controllers can be configured to support WebSocket connections. WebSocket is a protocol that enables full-duplex communication over a single TCP connection and is used for real-time applications.

7. Custom Protocols: In some cases, you may need to expose services using custom or proprietary protocols. Depending on your Ingress controller and its capabilities, you might be able to configure it to handle these custom protocols.

Additionally, Ingress controllers often evolve, so it's essential to refer to the documentation and features of the specific controller you plan to use to ensure compatibility with your service protocols.

WRITTEN ASSIGNMENT 2

Q1. How to deploy lambda function on AWS?

Deploying a Lambda function on AWS involves several steps. Lambda is a serverless compute service that lets you run code without provisioning or managing servers. Here's a step-by-step guide on how to deploy a Lambda function:

Step 1: Create a Lambda Function

1. Log in to the AWS Management Console.
2. Navigate to the Lambda service by searching for it in the AWS Services section.
3. Click the "Create function" button.

Step 2: Configure Your Function

4. Choose "Author from scratch."
5. Fill in the basic information for your function, including the name, runtime, and execution role.
6. For "Execution role," you can create a new role from a template or use an existing role if you have one with the necessary permissions.
7. Click the "Create function" button.

Step 3: Upload Your Code

8. In the "Function code" section, you can upload your code either directly (ZIP file or folder) or by specifying a repository from AWS CodeCommit, Amazon S3, or other options.
9. Configure the handler if needed. The handler is the method that AWS Lambda invokes.

10. Set environment variables if your code requires them.
11. Adjust the runtime settings if necessary.
12. Click the "Deploy" button to upload your code.

Step 4: Define the Trigger

13. In the "Add triggers" section, you can specify event sources that will trigger your Lambda function. This can be an API Gateway, an S3 bucket, an SNS topic, etc.
14. Configure the trigger according to your needs and grant permissions as required.

Step 5: Test Your Function

15. You can test your Lambda function by creating a test event or using a sample test event provided by AWS.
16. Click the "Test" button to run your function and see the results.

Step 6: Monitor and Troubleshoot (Optional)

17. AWS Lambda provides various monitoring and logging options. You can configure CloudWatch alarms, inspect logs, and set up notifications to keep an eye on your function's performance.

Step 7: Save and Deploy the Function

18. After you've configured everything to your satisfaction, click the "Save" button to save your changes.
19. Click the "Deploy" button to make your function live and ready to respond to triggers.

Step 8: Invocation and Scaling

Your Lambda function is now deployed and ready to be invoked by the trigger you configured. AWS Lambda handles the scaling of resources based on the incoming workload automatically.

Q2. What are the deployment options for AWS Lambda?

AWS Lambda offers several deployment options:

1. Code Upload: You can directly upload your function code as a ZIP file or a deployment package when creating or updating your Lambda function.

2. AWS Lambda Layers: You can use Lambda Layers to separate your function code from its dependencies. This allows you to manage and version common libraries independently and share them across multiple functions.

3. AWS SAM (Serverless Application Model): AWS SAM is a framework for building serverless applications. You can define your Lambda functions and their associated resources in a SAM template file, then deploy the entire application using AWS SAM CLI.

4. AWS CloudFormation: You can use AWS CloudFormation templates to define and deploy Lambda functions along with other AWS resources. This enables infrastructure as code (IaC) for your serverless applications.

5. AWS Serverless Application Repository: You can publish and share serverless applications and Lambda functions using the AWS Serverless Application Repository. Users can deploy your applications directly from the repository.

6. AWS CodePipeline: You can integrate AWS Lambda deployment into a continuous integration/continuous deployment (CI/CD) pipeline using AWS CodePipeline. This automates the building, testing, and deployment of your Lambda functions.

7. Container Image Support: AWS Lambda now supports deploying functions as container images, allowing you to package your function and dependencies in a Docker container and deploy them as a Lambda function.

These deployment options provide flexibility and enable you to choose the one that best fits your application architecture and development workflow.

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

In the context of AWS Lambda, there are three primary deployment modes:

1. Automatic Deployment: AWS Lambda provides automatic deployment when you directly upload your function code as a ZIP file or specify a deployment package. This is the most common and straightforward deployment method, and it's suitable for most use cases.

2. Infrastructure as Code (IaC) with CloudFormation: AWS CloudFormation is a service that allows you to define and provision AWS infrastructure and resources in a template. You can use CloudFormation to define your Lambda functions, their associated resources, and any other AWS resources your application needs. When you create or update the CloudFormation stack, it deploys the Lambda functions and other resources defined in the template. This approach is more suitable for complex serverless applications and infrastructure orchestration.

3. Serverless Application Model (AWS SAM): AWS SAM is an open-source framework for building serverless applications. It extends AWS CloudFormation to provide a simplified way to define serverless resources, including Lambda functions. You can use AWS SAM to define your functions and related resources in a SAM template, and then deploy the entire application using AWS SAM CLI. This approach is a balance between the simplicity of direct deployment and the power of CloudFormation for more complex applications.

These deployment modes offer different levels of control and abstraction, allowing you to choose the one that best matches your deployment needs and development workflow.

Q4. What are the 3 components of AWS Lambda?

AWS Lambda consists of three primary components:

1. Function Code: This is the core component of AWS Lambda. It's the code that you want to run in response to events. You can write your code in various supported programming languages (e.g., Node.js, Python, Java, C#, etc.). You can package your code along with its dependencies into a deployment package, which you upload to Lambda.

2. Event Sources: Event sources are triggers that invoke your Lambda function. AWS Lambda supports various event sources, such as Amazon S3, Amazon DynamoDB, Amazon SNS, AWS API Gateway, and more. When an event occurs in one of these services, it triggers the execution of your Lambda function.

3. Execution Environment: AWS Lambda manages the execution environment for your function. It automatically scales and provisions the necessary infrastructure to run your code. You don't need to worry about server provisioning or resource management. AWS Lambda also provides runtime support for various programming languages, so your code

runs in an isolated environment with the necessary resources to execute.