NAME:_____ STD.:_____ DIV.:_____

DATE :
PAGE :

Brijesh R Sharma
Roll No. 48
D15C

Adv. Devops Assignment 2

1) Create a REST API with the serverless framework.

Steps to create a REST API with serverless framework

1) Install serverless framework globally using the following command in the terminal

npm install -g serverless

This command installs the serverless framework on your machine globally using npm. It allows you to create, manage and deploy serverless applications across various cloud providers using AWS

2) Create a new service with AWS Node.js template:

serveless create --template aws-nodejs --path rest-api.

This command initialises a new serverless service called REST API. It creates a folder containing basic filter and a template specifically configured for building serverless application using Nodejs on AWS Lambda.

3) Navigate to the project directory:

cd rest-api

This command changes directory to new created project directory.

4) Initialise Node.js project and install dependencies.

npm init -y.

npm install express serverless-http

The express dependency builds the REST API and serverless-http integrates express with AWS Lambda.

5) Edit the serverless.yml file to include ~~service:~~

NAME:_____ STD.:_____ DIV.:_____

DATE :
PAGE :

```
service: rest-api
provider:
  name: aws
  runtime: nodejs 14.x
  stage: dev
  region: us-east-1
functions:
  app:
    handler: handler.app
    events:
      http:
        path: /
        method: any
```

This configuration specifies the service name, AWS provider settings and defines Lambda functions with HTTP event trigger.

6) Edit handler.js to add the Express app

```
const express = require('express');
const serverless = require('serverless-http');
const app = express();
app.get('/helloworld', (req, res) => res.json({message: "Hello World"}));
module.exports.app = serverless(app);
```

This creates a simple express app with a single route /helloworld and export it in a Lambda compatible format.

7) Deploy the service.

```
serverless deploy
```

8) Test the deployed API:

curl https://<api-id>.execute-api.<region>.amazonaws.com/dev/helloworld

Using above command, it returns a JSON message:

{"message": "HelloWorld"}

9) Redeploy after updates

serverless deploy

After modifying the code, redeploy it to update the API with our changes.

10) Remove the service

serverless remove

The above command removes all AWS resources associated with the API ensuring that there are no charges for unused charges.

2) Case study for Sonarqube

a) Create your own profile in sonarqube for testing project quality

b) Use Sonarqube to analyze your Github code

c) Install sonarlint in your Java IntelliJ IDE or Eclipse IDE and analyze your Java code

d) Analyze python project with sonarqube

e) Analyze nodejs project with sonarqube

a) Create your own profile in Sonarqube

1) Download and install SonarQube from the official website Unzip the file and start the server by running:

./bin/windows-x86-64/startsonar.bat

This launches sonarqube locally and can be accessed at http://localhost:9000.

Teacher's Sign.: _____

2] log into SonarQube using the default credentials (user-name: admin, password: admin). After logging in, change the password.

3] Navigate to project tab, click on 'Create New Project' assign on project key and name and generate a project token.

b) Use sonarcloud to analyze your Github code.

1) Signup for sonarcloud from the official website using your Github account.

2) On sonarCloud under Projects > create project, choose your Github repository and grant sonarcloud access to it.

3) Add a sonar project properties file in the root of your
sonar.projectKey=python_project
sonar.language=py
sonar.sources=.

3] Push the analysis of the project by executing the following from the project directory: sonar-scanner
The results will be pushed to your local sonarQube server and the analysis is visible on the dashboard.

c) Analyse Nodejs project with sonarQube

1) Setup a Nodejs project

2) In sonarqube, ensure that all Javascript / Typescript plugins have been installed. Plugins can be installed from the marketplace tab in sonarqube.

3) Create a sonar-project properties file in your project root and include the following in it:

Teacher's Sign.:

sonar.projectkey:node.project
sonar.language.js
sonar.sources:

4) Run the analysis of the project by executing the sonar-scanner command.
SonarQube will analyse the Node.js project and show results on the dashboard, highlighting code quality, bugs and vulnerabilities.

At a large organization, your centralized operations team may get many repetitive infrastructure requests. You can use Terraform to build a "self-serve" infrastructure model that lets product teams manage their own infrastructure independly. You can create and use Terraform models that codify the standards for deployment and managing services in your organization allowing teams to efficiently deploy services in compliance with your organization's practices. Terraform Cloud can also integrate with ticketing systems like ServiceNow to automatically generate new infrastructure requests

Terraform, a leading infrastructure as a code (IAC) enables organisations to outcomes, automate and manage infrastructure using declarative configuration files which reduces manual errors, improves operational efficiency and making it scale central to a self-serve model are reusable, version-controlled Terraform modules, which allows to standardise infrastructure deployments.
Terraform cloud offers secure, centralised state management, preventing from overwriting each other's changes.

Teacher's Sign.:

Sentinel, Terraform cloud's policy-as-code framework enables organisation to ~~ensure~~ enforce government policies and integrate them into CI/CD pipelines. Terraform Cloud's Integration with ticketing system like service now streamlines infrastoastructure provisioning by automating work and approvals, reducing manual intervention while maintaining compliance. It also supports automated workflows like GitLab cost estimation features allow teams to forecast expenses, while multi-cloud and hybrid-cloud support ensures scalability across diverse environment. security is embedded IAM policies state encryption and integrates with secrets management tools, ensuring adherence to frameworks like SOC2 and HIPAA.

Terraforms drift detection ensures that infrastructure remains assigned with desired state, automatically identifying and correcting any manual changes. Implementing a self-serve infrastructure model with Terraform enables large organizations to manage their infrastructure with greater autonomy, efficiency and control. By leveraging Terraform, teams can deploy infrastructure that adheres to organizations security with operational bottlenecks thus making Terraform, an ideal tool for modern infrastructure management in complex integration.

Teacher's Sign.: