Advance DevOps Assignment 2.

Q.1 Create a REST API with the serverless Framework.

→ Following are the steps:

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

install -g serverless.

This command installs the serverless yaa machine globally using upm.

2) Create a new services with AWS Node js serverless

Create template aws-nodejs

This command initializes a new serverless service called Rest-API. Using Node.js on AWS Lambda.

3) Navigate to the project directory:

cd rest-api

This command changes directory into the newly created directory to manage files.

4) Initialise Node js project and install dependencies

npm init -y

npm install express serverless http

The express dependency build the REST API and serverless

5) Edit the serverless your file to include service

rest api provider

name - aws

runtime - node.js 14.x

stage - dev

region - us-east-1

Function :

app:

handler : handler app
events :
http :
path : /
method : any.

This configuration specifies the service name AWS provider setting and defines lambda function with HTTP events.

6) Edit handler.js to add the express app

```
const express = require('express')
const serverless = require('serverless-http')
app.get('/hello-world', (req, res) => {
  res.json({ message: 'Hello world' })
})
```

7) Deploy the service

serverless the service

Deploy the API to AWS setting up resources like lambda and API.

8) Test the deployed API

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

9) Redeploy after update :

serverless deploy

After modifying the code redeploy it to update API with service

10) Remove the service

serverless remove

The above command removes all AWS resources associated with the API ensuring that there are charges for unused services.

Case study for sonarqube.

- Create your own profile in sonarqube for testing project quality.
- Use sonarcloud to analyze your GitHub code.
- Install sonarlint in your Java intellij ide or eclipse ide and analyze your Java code
- Analyze python project with sonarqube
- Analyze node js project with sonarqube

→

* Create your own profile in sonarqube -

1) Download and install sonarqube from the official website Unzip the file and start the server
To start SonarQube server, navigate to bin/ directory and choose the operating system (eg. linux - X86-64) and run the ./sonar.sh start
We can then access the server via http://localhost:9000.

2) Login using the default credentials →
name : admin, password : admin
After logging in change the password

3) Click on create new project, assign a project name and generate a project token.
To generate the token we go to Profile → security → Generate User Token.

* Use SonarCloud to analyze your GitHub code

1) Sign up for sonarcloud from official website using your Github account.

2) On sonarcloud under Project → ~~create~~ Analyze new project, choose your GitHub account and select a repository
Click set up and proceed with the default settings

3) In the github repo, add a workflow YAML file named sonarcloud.yml for continuous integration.
4) Generate a security token in sonarcloud and copy it. In your repo, go to settings → secrets & variable → Actions. Add a new secret called SONAR_CLOUD. Here we will paste the copied token.
5) Push changes to your repository.
Afterwards, you can view the results in SonarCloud by navigating to your project's dashboard. Check for bugs, code smells, vulnerabilities and other code metrics.

* Install sonarlint in your eclipse IDE / Java Intellit
1) Install sonarlint by going to plugin / market place In the IDE, configure sonarLint by linking it to your SonarQube or SonarCloud project
2) Open the Java project you want to analyze in Eclipse Sonarlint will automatically start analyzing code as you edit file. Real time feedback can be seen in the Problems tab.
3) To trigger a full project analysis
Rightclick on project in Project Explorer
Sonarlint → Analyze from menu
4) Sonarlint then displays issues like bugs, code smells and vulnerabilities in the Problems tab of Eclipse.

* Analyze Python project with SonarQube
1) Since we have already installed and started SonarQube we download the sonar scanner from its official website and add the path to your system path.

2) In SonarQube, create a new project and generate a security token.
3) Add SonarQube Properties to the root of your Python project
Create a sonar-project.properties
and add project key, url, login, source & language
Then run : sonar-scanner.

* Analyze nodejs project:
1) We install sonarqube and sonarscanner like stated above
2) In the root of nodejs project, create a file sonar-project.properties
and add project key, url, login, source and language.
The run analysis : sonar-scanner
3) Results can be viewed from SonarQube Project dashboard.

Terraform "self-serve" Infrastructure Model
① Terraform Modules for self-serve infrastructure
• Create Terraform modules that codify the standards for deploying common resources like VPCs, EC2 instances and S3 buckets.

Example module for an EC2 instance

ec2 - module/main.tf

variable "instance-type" {

```
    default = "t2.micro"
  }
resource "aws_instance" "example" {
  ami = "ami-12345678"
instance_type = var.instance_type
  tags = {
          Name = "example-instance"
        }
}
```

ec2-module/outputs.tf

```
output "instance_id" {
   value = aws_instance.example.id
}
```

Teams can now use this module to deploy EC2 instances.

② Terraform Cloud Integration with service Now.

- You can integrate Terraform cloud with serviceNow to automate the infrastructure request process.
- Using Terraform's API-driven approach serviceNow can trigger Terraform runs based on ticket approvals automating resource deployment.

Example workflow

① A product team submits a request in serviceNow for new infrastructure.

② The request triggers a Terraform cloud updates the serviceNow ticket with the status and resource details

③ Creating Terraform Module for Teams

Define reusable modules for commonly requested resources like

1. Networking (VPC, subnets)
2. Compute (EC2) ~~S3 bucket~~
3. Storage (S3)
4. IAM Roles / Policies.

By doing this, teams can manage their own infrastruc-ture while maintaining compliance with organizational standards.