ASSIGNMENT NO: 02

Q.1 Create a Rest API with the serverless framework.

Ans: Step 1: To create Rest API with serverless framework:

1] Install serverless Framework globally using the following command on the terminal:

```
npm install -g serverless
```

This command installs the serverless Framework on your machine globally using npm. It allows you to various cloud providers, including AWS.

2] Create a new service with AWS Node.js template

```
serverless create --template aws-nodejs --path rest-api
```

This command initialises a new serverless service called rest-api It creates a folder containing basis file and a template specifically configures application

- Initialize nodes project & install dependencies.
- npm init -y
- npm install express serverless -http.

Edit the serverless -http integrate building the REST with AWS Lambda.

- Edit the serverless.yml file to include -

```
service: rest-api
provider:
        name: aws
        runtime: nodejs24.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 define theier Lambda functions with http event trigger.

- Edit handler.js to add the express app.

```
- const express = require ('express');
  const app = express ();
  app.get ('/hello', (req, res) => res.json ({msg: 'hello"}));
```

This creates a simple express app with a single route.

- Deploy the app & the API gateway. A virtual lab is generated for testing
- Test the deployed API.

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

Using the above retuins a JSON response.

{ Message: 'hello' }

2. Case study for Sonarqube

i. Setting up your profile for sonarqube: Creating a profile in sonarqube allows devops to analyze the quality of their projects and track improvements over time.
Steps: Install sonarqube & set it up locally.
- Once logged in create a new project in sonarqube by prowing a project name & key.
- Add the sonarqube properties file to the root directory of the project which contains necessary configurations.
- Use sonarqube scanner to analyze your project & uploads the result to the dashboard.

ii) Using sonarqube to analyze github code:
Steps: Signup & connect your github repository.
- Setup github actions to run sonarqube scans whenever code is pushed into the respository. This ensures continuous code quality analysis.
- Create sonar project properties file with the necessary configurations in the root directory of the project.
Sonarcloud scanner is triggered everywhere.

iii) Sonarlist for real time code analysis in IDEA: Sonarlist is a plugin for intelly IDEA & Eclipse that performs code analysis. Steps: Install the plugin for intelly IDEA, goto files 7 settings > plugins find sonarlist & install.
Sonarlist can be linked to sonarqube instance to sync ruler & quality profiles.

- Sonarlist runs automatically as you write code & flag issues directly in the editor.

iv. Analyzing python project with 'sonarqube'

- Steps: Ensure Sonarqube is running configure sonarqube for python (specify the source files).
- Execute sonar scanner from the root of yain project.
- Analysis results will be uploaded to Sonarqube.

v. Analyzing Nodejs project with Sonarqube.

- Steps: Verify javascript plugin is available in your Sonarqube instance.
- Configure project by adding properties in the sonar project file.
- You can also combine Sonarlist with Sonarqube for a more comprehensive javascript analysis.
- Use sonar-scanner to analyze the project.

3. At a large organization your centralized operation team may get infrastructure sequences. You can see Terraform to build a self-service infrastructure model that lets product teams manage infrastructure independently. Create & use terraform modules that codify the standard for deploying & managing services in your organization. Terraform cloud can also integrate with ticketing options like Service Now to automatically generate news infrastructure requests.

→

- Steps: Define infrastructure standard Establish class standards & best practices for infrastructure deployment including resource types, tagging policies & security compliances.

- Step 2: Create terraform module based on the organizations standards. Deploy a common resource using EC2 bucket instances & S3 buckets.

```
eg:- variable "instance- type" {
            default = "t2. micro"
       }
    resource = aws instance "example" {
        ami = "ami - 1245678"
        instance type = " var. instance. type"
        tags = {
             name = "example-instance"
        }
    }
```

e22 modules outputs if :

```
output " instances - id "{
        value = instance. example. id
}
```

Terraform cloud integration to automate infrastructure request process can run on a ticket approval, automating & resource deployment.

Step 3: Creating Terraform modules for teams. Define reusable modules for commonly requested modules Such as-

- Networking CVPC subnets.
- Compute
- Storage
- IAM Roles.

By this teams can manage their own infrastructure while maintaining compliance with internal organization standards.