05/05

Advance DevOps
Assignment No: 2

1. Create a REST API with the Serverless Framework.

Ans: The Serverless Framework is an open source that simplifies the deployment and management of Serverless applications. It allows developers to build and deploy application on cloud platform like AWS without having to manage the underlying infrastructure.

~~Step 1~~:
Prerequisites:
1. Node.js and npm installed on your local machine.
2. Serverless Framework installed globally using npm:
Command: npm install -g serverless.

Step 1: Install the Serverless Framework.
First ensure you have Node.js and npm installed. Then ensure install the serverless Framework globally.

Step 2: create a new serverless Service
serverless create --template aws-nodejs --Path my-service cd my-service.

Step 3: Define function in Serverless.yml
open Serverless.yml file in Project directory.
In this file we define our service Configuratic
including the functions and their triggers/
events.
This file Contains Service, Provider it name
and runtime environment. It also Contains
functions which contains create, read,
update and delete methods.

Step 4: Write Lambda Functions (Handlers)

Open the handler.js file, and write the logic
For the API endPoints.
handler.js Contains the logic For:
- Handling a simple GET request.
- Handling POST request to create a new item

Step 5: Deploy the Service
To deploy the Service and Lambda Functions,
Command: Serverless deploy.
with the 'sls deploy' Command, Serverless
Framework packages your applications,
uploads necessary resources to AW's and
set up the infrastructure.

Step 6: Testing the API:
Once deployed you can test REST API using tools like curl or postman by making POST requests to generated API.

Step 7: Storing data in Dyanmo DB.
To store Submitted Candidate data, you integrate AWS Dynamo DB as a database.

Step 8: AWS IAMpermissions
You need to ensure that Serverless framework is given right permissions to interact with AWS resources like dyanomo DB.

This is the whole process which will create a fully serverless REST API using AWS Lambda, API Gateway the serverless Framework.

Q.2 Case study for Sonarqube:
creating ur own profile in Sonarqube for testing project quality use Sonarqcloud to analyze your Github code.
Install Sonarlint in ur Java intellij ide and analyze Java code Analyze python project with Sonarqube.

Ans: Sonarqube is an open-source platform used for continuous inspection of code quality. It detects bugs, code smells and security vulnerabilities in project across various programming languages.

Soln:
- Create the sonarqube profile for testing project quality.
- Open intellij setting, find Tools > SonarLint entry and select + to open connection wizard
- Enfr a name for this Connection, select sonarcloud or sonarqube.
- Choose the authentication method.

(a) Generate token on sonarqube or Sonarcloud
(b) username + password: This can be used only Sonarqube connection only.

- For Sonarcloud only select organisation that you want to connect.
- Sonarqube and Sonarcloud can push notification to developers.

- Validate the connection creating by selecting Finishing at the end of the wizard.
- Save the connection in global setting by clicking OK.

- BIND PYTHON PROJECT TO SONARQUBE.
- Select SonarLint > Bind Project to Sonarqube/ Sonarcloud.
- Choose the Correct Projects from Sonarqube.
- Analyze the Project. (Python Project)
- Trigger an analysis by going to Code > Analyze Code > SonarLint.

- Analyze Node.js Projects.
- Make sure your Node.js Project is properly Configured with Sonar-Project. Properties file or equivalent for the analysis to run.

Q.3 At large organization, your centralized operations team may get many repetitive infrastructure requests. You can use Terraform modules to build a "self-serve" infrastructure model that lets product teams manage their own infrastructure independently. You can create and use Terraform modules that codify the standards for deploying and managing services in

your organisation, allowing teams to efficiently deploy services in compliance with your organisation's practices. Terraform Cloud can also integrate with ticketing systems like Service Now to automatically generate new infrastructure requests.

Ans: Self-service

Self--serve infrastructure Model with Terraform Modules:

At a large organisation, implementation of a self-serve infrastructure model using Terraform can significantly streamline the process of managing infrastructure across different teams. This approach allows product teams to manage their own infrastructure independently while adhering to organizational standards and best practices.

Key aspects of this self-serve model include:

(a) Standardization through Terraform Modules: By creating and utilizing Terraform modules, organizations can codify their infrastructure deployment and management standards. These modules serve as reusable packages of Terraform configurations that encapsulate common patterns and best practices.

(b) Efficient Deployment: Product teams can leverage these standardized modules to quickly deploy services without needing

to reinvent the wheel or wait for the centralized operation team to handle every request.

(c) Compliance
By using predefined modules, teams, ensure that their deployments comply with the organisation's established practices and security gridelines.

(d) Automation: The use of Terraform modules promotes automation. reducing manual intervention and potential human errors in infrastructure management.

(e) Version Control: With modules stored in version control system like Git, teams can track changes, collaborate on improvements, and maintain a history of infrastructure configurations.

INTEGRATION WITH TICKETING SYSTEMS:
terraform cloud offer integration capabilities that further enhance the self-serve model:
(a) Automatic infrastructure Request: Terraform cloud can integrate with ticketing systems like service Now to automatically generate new infrastructure requests. This automation streamlines the process of submitting and tracking infrastructure changes.

(b) Centralized Management: By Centralizing infrastructure management through terrafor Cloud organisation can maintain better control over who can request and approve infrastructure changes.

(c) Goverance: The integration with ticketing syste allows for better goverance of infrastrecture requests, ensuring that all changes go through proper approval processes before deploymen

Collaborative Infrastructure Management.

a) Team Based Performance Permissions.
b) State and Run History.
c) Sensitive Information Protection.
d) Module Registry.

• By implementing these features and practic large organisations can effectively leverage Terraform to build a robust, scalable and Complaint infrastructure management system that supports both Centralized Control and decentralised team Autonomy.

Date
Page

• In large organizations using Terraform modules to build a self-serve infrastructure model allows product teams to manage their own infrastructure independently, reducing repetitive requests to the centralized operations team. Terraform modules codify standards for deploying & managing services, ensuring consistency, compliance and security. By Integrating with ticketing systems like ServiceNow, infra-structure requests can be automated, further streamlining deployment processes. This model increases efficiency, scalability & flexibility, enabling teams to deploy infrastructure quickly while maintaining governance and control.

Benefits of Self-serve infrastructure Model:

1. Codifying standards with Terraform modules:
Terraform modules, which are reusable configurations that define cloud resources, can serve as blue-prints that adhere to your organization's best practices. teams are guided in deploying consistent & compliant services across various environments.

2. Automation and Autonomy:
Once these modules are created, product tean
Can autonomously use them to deploy resourc
without needing approval from the centrali
ops team for every change.

3. Integration with Ticketing Systems:
Integrating this self-service model with
systems like ServiceNow can automate
the generation of infrastructure reques
This reduces manual overhead and
Standardizes the process for new
infrastructure.

Adopting a self-service modules
streamlines infrastructure management in
large organizations. It empowers prod
teams, ensures consistency, & automate
repetitive tasks while maintaining
governance & control. This allows
organizations to scale infrastructure
management while maintaining a high
level of operational excellence.

Teacher's Sign.: