Page No. 38
Date 04/05/

Swaraj Patil D15A 40

Advance Devops Assignment 2

Create REST API with serverless framework.

Here are steps to be followed to create REST API using serverless frameworks.

1.) Install serverless framework:
Run npm install -g

2.] Initialize project:
Create a new project directory and initialize new service using template

3.] Define Configuration: Edit the serverless.yml file to specify the service name, provider details, runtime.

4.] Create Lambda Functions:
Implement the logic for each API endpoint in a separate file.

5.] Configure Datastore: Choose a database and set permissions in the serverlessyml file if needed.

Teacher's Sign.: _____

6.] Deploy and Test API:

- Use the serverless framework to your API to AW.
- Use tools like postman or curl test the API endpoints.

7.] Monitor and Updates: Monitor perf -mance via AWS cloud watch and update the API as needed.

Q2] Case study for SonarQube:

→ Case study: Quality Analysis of Sona -Qube:-

objectives:

1.] Create a SonarQube profile:
- Log into sonarqube and navigate quality profiles.
- Click create name your profile and customize rules for your desire quality standards.

2] Analyze github code with Sonarcloud Sign in to Sonar Cloud with github and link your repository.
- Set up Sonar-project properties file in your repo, trigger analysis throug CI/CD

) Install SonarLint for Java:
Install SonarLint in intelliJ from
he plugins
Bind your project with SonarQube
nd SonarLint will flag issues with
our Java code and analyse it.

) Analyze Python project:
Create a new project in SonarQube
for Python code
Add a sonar-project.properties file
nd run analysis using sonarscanner
nd check results.

5) Analyze Node.js project:
Create a new project in SonarQube
or Python code.
Add a sonar-project.properties file and
un analysis using sonarscanner and
~~t~~ check results.

Conclusion: This casestudy demonstrates
ow to use SonarQube effectively for
code quality analysis across different
programming languages and platforms.

Q.3] In large organization, the centralized operations team often faces repetitive infrastructure requests, which can be streamlined through automation and self service infrastructure models.

Terraform can play a key role in this by enabling self service

1.] Terraform Modules for standardization
You can create reusable terraform modules that encode your organizational standards and best practices for deploying and managing infrastructure

2.] Self-service Infrastructure:
Teams can manage their own infrastructure without needing to rely on central ops for every request.

- Self-service infrastructure enables product teams to iterate faster and deploy resources more efficiently, leading to higher agility in development

Integration with ticketing systems:
Terraform cloud integrates with systems like servicenow to automatically generate new infrastructure requests based on service needs.

Advantages:
Scalability
Compliance and Governance
Faster Delivery
The self service model improves both efficiency and governance across large organizations.