Roshan B
D15C
4

ASSIGNMENT- 2

04/05

2

Q1 Create a REST API with the Serverless Framework.

① Initialize the Serverless Project.
First, create a new directory and initialize a Serverless Project.

② Modify serverless.yml
Open the serverless.yml file and configure it to define your lambda functions and HTTP endPoint.

③ Implement Lambda functions.
Open handler.js and implement your lambda functions

④ Deploy the API
Deploy your REST API to AWS using the Serverless Framework.

⑤ Testing the API.

You can test your endpoints using tools like Postman or cURL. For example, using URL to test the GET / hello endpoint.

6) Monitor and logs.
To monitor your API, you can use AWS CloudWatch logs. You can view logs for a specific function.

Q3 Case study for Sonarqube.

1) Creating your own quality Profile in Sonarqube.

Sonarqube allows you to define custom quality profiles based on your project's need, rules and preferences

2) Using Sonarcloud to Analyze a GitHub project.

Sonarcloud is a cloud-based version of Sonarqube that integrates directly with GitHub repositories for automatic code analysis.

③ Install SonarLint in IntelliJ IDEA or Eclipse for Java Analysis.

SonarLint is an IDE extension that provides real-time feedback on your code quality while you're coding.

④ Analyzing a Python Project with SonarQube.

SonarQube supports Python code analysis and helps you identify issues like Code Smells, bugs, Vulnerabilites and duplications

⑤ Analyzing a Node.js Project with SonarQube.

Node.js Projects can also be analyzed with SonarQube for identifying Code Quality issues.

Q3 At a large organization, your centralized operations team may get many repetitive infrastructure structure. You can use Terraform to build a "Self-serve" infrastructure model that lets product teams manage their own infrastructure independently. You can create and use Terraform to build a "self-serve" infrastructure independently. You can create and use Terraform modules for deploying and managing services in your organization, allowing teams to efficiently deploy services in compliance with your organization's practice. Terraform cloud can also integrate with ticketing systems like ServiceNow to automatically generate new infrastructure requests

Q.No 2

→ Self-Serve Infrastructure Model.

① Empowers Product teams to manage their own infrastructure independently.
② Reduces workload for centralized operation team.
③ Increases efficiency and agility.

Terraform Modules.
① Codify standards for deploying and managing services
② Ensure compliance with organizational practices.
③ Enhance audit trails and version control.

Additional Benefit.
① Version control and change tracking
② Reusability of infrastructure
③ Consistency across environments
④ Improved collaboration between teams
⑤ Reduced errors and downtime.

By implementing a self-serve infrastructure model with Terraform, organizations can:-

① Increase Productivity
② Improve compilance
③ Enhance scalability
④ Reduces Costs.

This approach enables organizations to adopt a more eligible and efficient infrastructure management Steategy, aligning with Devops best practices.

Features:-
① Version Control:- Track infrastructure Changes and roll back if needed.
② Consistency:- Ensure identical enviroments across dev, Staging and Prod.
③ Efficency- Automate repetative infrastructure tasks.
④ Collaboration:- Simplify infrastructure management across teams
⑤ Compilance:- Enforce organizational standards and Policies.