Sandesh. M. Yadav

DISC 61

Assignment No. 2

1) Create a rest api with serverless framework?

(1) Install serverless node package using npm install -g serverless

(2) Create a serverless.yml configuration file with following code and save it.

```
service: my-rest-api
provider:
name : aws
runtime : nodejs 12.x
region : us-east-1
function:
    ServerInfo:
        handler : handler.ServerInfo
        events:
          -http:
            path : ServerInfo
            method : get
```

Here name of my function is ServerInfo

(3) Create a new file in the same directory as handler.yml handler.js with the Rest API code that the server will return on calling it.

(4) In the same directory where the configuration file was created, initialize the serverless using command serverless Select Login/Register. A new browser window will open asking for us to sign-in / (create account if it does not exist) After setting up, type command serverless again to initialize project.

(5) Select create a new app

(6) Name your app.
(7) Connect your AWS to Serverless framework you will b shown multiple options, you can select any. Here I am using Create AWS IAM Role (Easy and recommended) method Connect. In this a URL will be shown, you have to open i into your browser and sign-in with AWS account. Scroll do and click on create stack.
(8) Now our serverless service is ready to deploy to it use serverless deploy. After succesful deployment we will get a URL of Serverless REST API endpoint.
(9) Visit the provided URL. As we can see, that the Se returns the system information.

Q2.] Create your own profile in Sonarqube for testing project quality.

→ (1) Create a local project in Sonarqube dashboard
(2) Create quality profile select language and name the quality profile.
(3) Activate security checks according to your project
(4) Assign created quality profile to project Go to the project section and select the project we jus created. Then go to project settings situated at right corner and select quality profiles Add language o your project and select the created quality profile.
(5) Set up Jenking. Open Jenkins dashboard and create new item.
(6) Name the project and select item type as pipelin
(7) Go to the pipeline script section and paste the

following code and save it. Here we are analyzing the python project named as requests, a Simple HTTP Library

(7) click on Build now!

As we can see our analysis is completed in Jenkins dashboard

our analysis is passed in Sonarqube also in accordance with the specified quality profiles which we created earlier.

use Sonarcloud to analyze your github code.

(1) Sign up on Sonarcloud
(2) Create a new project in Sonar cloud.
(3) Fork the repository of your choice into your github account
(4) click on "import an organization from github".
(5) Give access permission to repository you want to analyze.
(6) A Summary will be shown, choose free plan and click on "create organization".
(7) Select the repositories under created organization and set up them.
(8) Select analysis method.
(9) Analysis will be started wait for sometime until analysis gets completed. As we can see our analysis is completed. Successfully with list of issues shown in the dashboard. we can see any issue in detail by clicking on it.
(10) In our example, ~~see~~ ~~see~~ ~~see~~ one of the maintainability issues is shown.

Q2. C] Install Sonarlint in your Java Intelli j ide eclipse ide and analyze your Java code.

→ (1) open Eclipse IDE
(2) Go to the help menu and Select eclipse marketplace
(3) In the eclipse marketplace search for Sonarlint
(4) In the search results, find Sonarlint and click the install button, Restart eclipse IDE after Installing Sonarlint.
(5) create a new Java project.
(6) Name your project and finish setting up
(7) Create a new sample Java file (e.g. Calculator.Ja in the newly created project.
(8) Run manual analysis. you can trigger a manual code analysis by right. clicking on the project folder, file in the project explorer and selecting Sonarlint →
(9) See results of analysis in bottom Tab. Here we error from Sonarlint.
(10) Create a new package com.myapp.Calculator
(11) Here we are getting warning to solve this as package.com to the top of our code in Calculator file and move calculator.Java to correct location such as /src/com/calculator.Java.

Analyze python project with Sonarqube.

1. Clone the project repository of your device.
2. Create the Sonar-project properties file in the root of the flask project directory.
3. Install dependencies before running the analysis, install the required dependencies of the flask project.
   pip install -r requirements / dev.txt.
4. Go to my account Go to Security tab. Name your token and select type as Global analysis token and click on generate token. A new token will be generated, copy it in clipboard
5. Now we would use SonarScanner CLI to analyze our project using below command
6. Open the project dashboard using provided url in output or look in projects section of Sonarqube dashboard. As we can see our analysis is completed.

Analyze node.js project with Sonarqube.

1. Clone the Node.js project you want to analyze for this example let us use the Express repository
2. In the root directory of your Node.js, create a sonar project.properties file to configure the analysis.
3. Now we have to create a global analysis token, to do so go to myAccount Go to Security Tab
4. Name your token and select type as Global analysis token and click on generate token.
5. A new token will be generated copy it in clipboard
6. Now we would use Sonar Scanner CLI To analyze our project using Command

(7) open the project dashboard using provided url in output or look in project section of Sonarqube dashboard.
(8) Now we can see our analysis is completed.

Q3.] At a large organization, your centralized operations te may get many repetitive infrastructure request you ca use Terraform to build a "self secure" infrastructu 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 organization, allowing teams to efficiently deploy services in compliance with your organizations practices. Terraform cloud can also integrate with ticketing systems like Service N to automatically generate new infrastructure request

→ Setting up S3 bucket using Terraform Scripts and
(1) Create a parent directory terraform-modules, in it create S3-bucket directory.
(2) Create 4 files named main.tf, variables.tf, o .tf and terraform.tfvars
(3) After creating above files inside S3 bucket direct the module structure should look like this taking directory as terraform-modules After creating all director and files, it should look like this
(4) Now initialize git repository on S3-bucket directory
(5) Create empty repository on Github and push the chang to it.

(6) Create a Terraform cloud account on app.terraform.io
(7) Create organization with organization name.
(8) In your organization, create new workspace that will be linked to your Terraform code repository.
(9) Authority Terraform cloud.
(10) Install Terraform cloud on GitHub app.
(11) Select repository from which terraform config files will be looked from.
(12) Review final settings and click on create.
(13) Give tag name and bucket name then click on start new plan
(14) Give run name and select run type.
(15) Generate access keys for AWS account. Search users select users. Create new user or select existing one if you already have.
(16) Go to settings → Variable sets and click on create Variable set. Name the variable set and set scope to Apply globally. Enter key name as AWS_access-key-id and value as your secret which you have get from AWS.
(17) Go to workspace on Terraform cloud and select S3 bucket Terraform and click on New Run situated at top right corner.
(18) Click on confirm and apply.
(19) Add comment to confirm applying plan. our plan finished applying
(20) Visit AWS S3 bucket dashboard to see newly created bucket our bucket is visible on AWS