

## Steps for task 2

Step 1 : Step 1 : User needs to clone the repository from <https://github.com/prabhat-roy/particle41.git>

Step 2 : Create a user in AWS IAM console and provide AdministratorAccess policy. Also generate a credential to authenticate with terraform

Step 3 : Download and install AWS cli and terraform to create the infrastructure

Step 4 : Run aws configure and provide the credentials generated from step 2.

Step 5 : Navigate to Terraform directory and modify the value in terraform.tfvars file to customize

Step 6 : Run terraform init to initialize

Step 7 : Run terraform validate to check everything

Step 8 : Run terraform plan to generate the plan

Step 9 : Run Terraform apply -auto-approve to create the infrastructure

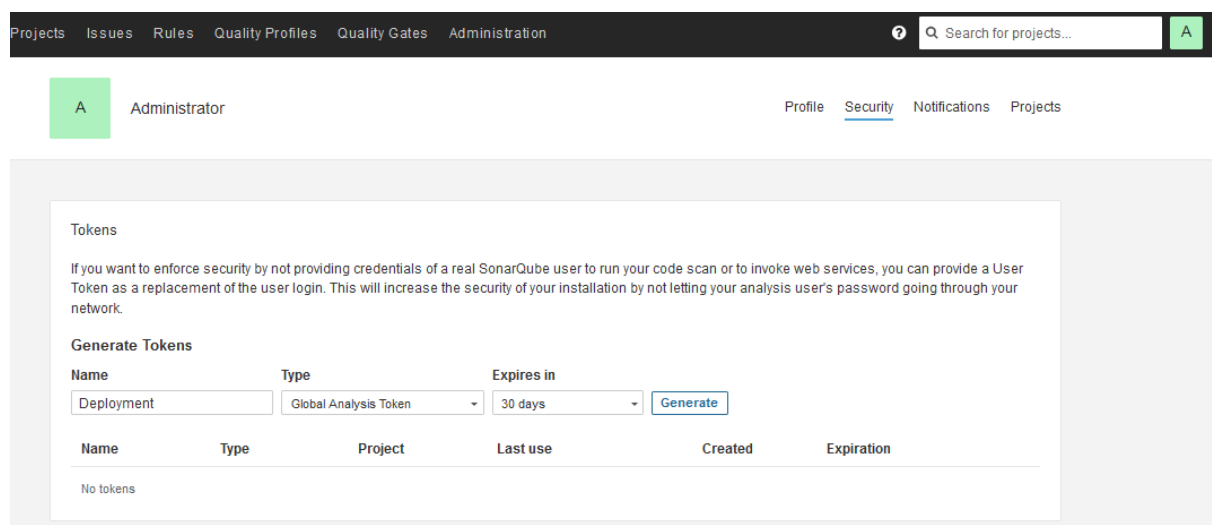
Step 10: When finished, copy the content mentioned in the last line of the output which is the initial admin password of the Jenkins

Step 11 : Visit the URL as part of terraform output to login into Jenkins and sonarqube

Step 12 : With the help of the password (step 10), create a user in Jenkins and then create the pipelines. We will create three pipelines, one is to create and delete Kubernetes cluster, another one is for deployment of the application and last one is to delete resources in Kubernetes.

Step 13 : Login to sonarqube url and login with admin/admin credential. Need to change the password after first login.

Step 14 : Generate a credential in sonarqube to be used in jenkins



The screenshot shows the SonarQube Administration interface. At the top, there's a navigation bar with links: Projects, Issues, Rules, Quality Profiles, Quality Gates, and Administration. A search bar is on the right. Below the navigation bar, the user 'Administrator' is logged in, with links for Profile, Security, Notifications, and Projects. The main content area is titled 'Tokens' and contains a message about using User Tokens for security. Below this is a 'Generate Tokens' section with a form. The form has three fields: 'Name' (set to 'Deployment'), 'Type' (set to 'Global Analysis Token'), and 'Expires in' (set to '30 days'). A 'Generate' button is next to the 'Expires in' field. Below the form is a table with columns: Name, Type, Project, Last use, Created, and Expiration. The table currently shows 'No tokens'.

Name	Type	Project	Last use	Created	Expiration
No tokens					

Step 15 : Go to Jenkins and then go to manage Jenkins -> plugins and add the sonarqube plugin

The screenshot shows the Jenkins plugin search results for the keyword 'sonar'. At the top, there is a search bar with 'sonar' entered and an 'Install' button. Below the search bar, three plugins are listed:

- SonarQube Scanner 2.18**: This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality. It was released 5 days 10 hr ago.
- Sonar Quality Gates 328.vf4369b\_da\_d3c2**: Fails the build whenever the Quality Gates criteria in the Sonar 5.6+ analysis aren't met (the project Quality Gates status is different than "Passed"). It was released 3 mo 17 days ago.
- Quality Gates 2.5**: Fails the build whenever the Quality Gates criteria in the Sonar analysis aren't met (the project Quality Gates status is different than "Passed").

Step 16 : Go to manage Jenkins -> credentials and add a credential of sonarqube of type secret text and put the token from sonarqube.

## New credentials

The screenshot shows the 'New credentials' form in Jenkins. The form has the following fields:

- Kind**: A dropdown menu with 'Secret text' selected.
- Scope**: A dropdown menu with 'Global (Jenkins, nodes, items, all child items, etc)' selected.
- Secret**: A text input field containing a series of dots, indicating a masked secret.
- ID**: A text input field with the value 'sonar'.
- Description**: A text input field that is currently empty.

At the bottom of the form, there is a blue 'Create' button.

Step 17 : Got to manage Jenkins -> System and configure sonarqube details. Since sonarqube is also running in Jenkins server, so the IP will be 127.0.0.1. And also select the credential from the dropdown.

## SonarQube installations

### List of SonarQube installations

Name

SonarQube

Server URL

Default is <http://localhost:9000>

http://127.0.0.1:9000

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

sonar

+ Add

Advanced ▾

Save

Apply

Step 18 : Go to manage Jenkins -> tools and add sonarqube scanner

Add SonarQube Scanner

☰ **SonarQube Scanner**

Name

SonarQube

☒ Install automatically ?

☰ **Install from Maven Central**

Version

SonarQube Scanner 7.0.0.4796

Add Installer ▾

Add SonarQube Scanner

Save

Apply

Step 19 : From the dashboard, go to new item and create a pipeline to create Kubernetes cluster.  
Provide a name and choose pipeline.

## New Item

Enter an item name

Kubernetes Cluster Setup

Select an item type



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps sequentially like archiving artifacts and sending email notifications.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

Suitable for projects that need a large number of different configurations, each with its own set of build steps.

OK

Step 20 : Scroll down and select “This project is parameterized” and click on add parameter button

## Configure

General

Pipeline

Advanced

☐ Do not allow the pipeline to resume if the controller restarts

☐ GitHub project

☐ Pipeline speed/durability override ?

☐ Preserve stashes from completed builds ?

☒ This project is parameterized ?

Add Parameter ▾

☐ Throttle builds ?

### Triggers

☐ Build after other projects are built ?

☐ Build periodically ?

Save

Apply

Step 21 : Select Choice parameter from the dropdown

☐ Boolean Parameter

☐ Choice Parameter

☐ Credentials Parameter

☐ File Parameter

☐ Multi-line String Parameter

☐ Password Parameter

☐ Run Parameter

☒ String Parameter

Add Parameter ^

☐ Throttle builds ?

## Triggers

Step 22 : Provide the name and choices as per below

☒ This project is parameterized ?

**Choice Parameter** ?

Name ?

Action

Choices ?

apply  
destroy

Description ?

Save Apply

Step 23 : scroll down and provide the details of the pipeline. Definition will be Pipeline script from SCM, URL will be <https://github.com/prabhat-roy/particle41.git> , branch will be main and script will be Jenkinsfile\_Kubernetes\_Cluster (as per below) and save it.

Pipeline

Define your Pipeline using Groovy directly or pull it from source control.

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

<https://github.com/prabhat-roy/particle41.git>

Credentials ?

- none -

Save Apply

Branch Specifier (blank for 'any') ?

\*/main

Add Branch

Repository browser ?

(Auto)

Additional Behaviours

Add

Script Path ?

Jenkinsfile\_Kubernetes\_Cluster

Save Apply

Step 24 : from the pipeline page, Click on build with parameters and select apply action from the dropdown and click on build button to run the pipeline. It will take around 15 – 20 minutes to complete.

Status

Changes

Build with Parameters

Configure

Delete Pipeline

Stages

Rename

## Pipeline Kubernetes Cluster Setup

This build requires parameters:

Action

Build

Cancel

Step 25 : From the dashboard, create another pipeline (without parameter) and the script path will be “Jenkinsfile\_Kubernetes\_Deployment”. It is for application deployment in the Kubernetes cluster. After completing above step, run this pipeline to deploy.

```

NAME                                TYPE      CLUSTER-IP      EXTERNAL-IP
AGE
simpletimeservice-service    LoadBalancer    172.20.21.99     a18cc8f4890e648d28342b35c45dbafa-1217682165.ap-south-2.elb.amazonaws.com
80:31250/TCP    65s
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

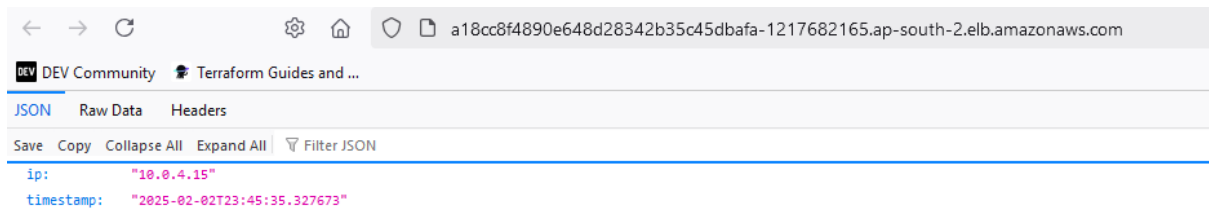
```

Step 26 : From the dashboard, create another pipeline (without parameter) and the script path will be “Jenkinsfile\_Kubernetes\_Resource\_Delete”. Its for deleting of Kubernetes resources created in above steps.

Step 27 : All the pipelines are created.

S	W	Name ↓	Last Success
		<a href="#">Delete Kubernetes Resources</a>	1 min 59 sec <span>#4</span>
		<a href="#">Kubernetes Cluster Setup</a>	1 hr 55 min <span>#1</span>
		<a href="#">Kubernetes Deployment</a>	8 min 11 sec <span>#14</span>

Step 28 : Copy the load balancer url (from step 25) and paste in any browser to get the output.



Step 29 : Go to sonarqube page to check the status.

