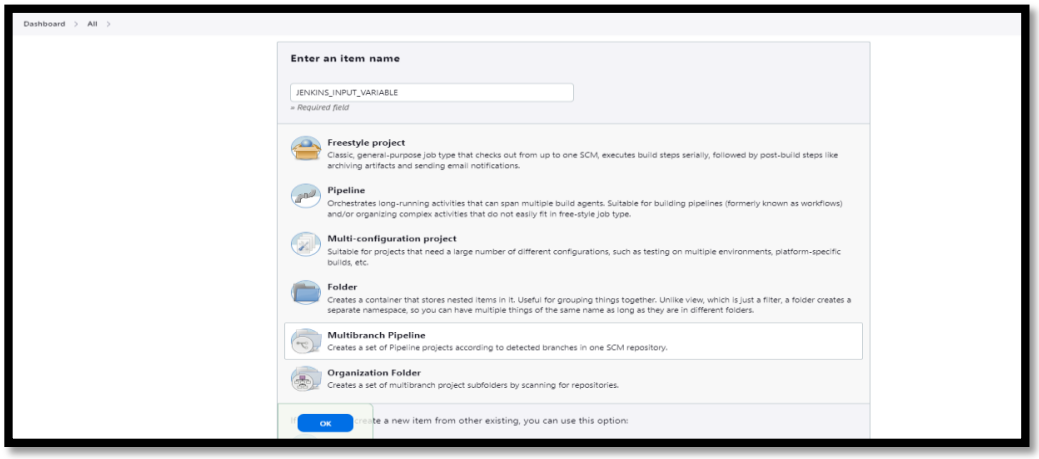
2. Design a Jenkins file to execute any Terraform code, prompting the user for two inputs: Terraform apply and Terraform destroy. Depending on the provided inputs, execute the corresponding Terraform command accordingly.

Create Multi-branch Pipeline



Install Terraform Plugin

INSTALL TERRAFORM

root@DESKTOP-8OOG2HF:JENKINS_INPUT# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

2a4f128a7e7e myjenkins-blueocean:2.440.1-1 "/usr/bin/tini -- /u…" 13 days ago Up About an hour 0.0.0.0:8080->8080/tcp, :::8080->8080/tcp, 0.0.0.0:50000->50000/tcp, :::50000->50000/tcp jenkins-blueocean

```
root@DESKTOP-8OOG2HF:JENKINS_INPUT# docker ps
CONTAINER ID   IMAGE                          COMMAND                   CREATED        STATUS              PORTS
                                                                NAMES
2a4f128a7e7e   myjenkins-blueocean:2.440.1-1   "/usr/bin/tini -- /u…"   13 days ago    Up About an hour   0.0.0.0:8080->8080/tcp, :::8080->8080/tcp, 0.0.0.0:50000->50
000/tcp, :::50000->50000/tcp   jenkins-blueocean
```

root@DESKTOP-8OOG2HF:JENKINS_INPUT# docker container exec -u root -it 2a4f128a7e7e /bin/bash

root@2a4f128a7e7e:/# apt update

root@2a4f128a7e7e:/# apt install sudo

root@2a4f128a7e7e:/# apt install wget

**root@2a4f128a7e7e:/# wget -O- https://apt.releases.hashicorp.com/gpg | **

**gpg --dearmor | **

sudo tee /usr/share/keyrings/hashicorp-archive-keyring.gpg > /dev/null

--2024-03-19 08:47:11-- https://apt.releases.hashicorp.com/gpg

**root@2a4f128a7e7e:/# gpg --no-default-keyring **

**--keyring /usr/share/keyrings/hashicorp-archive-keyring.gpg **

--fingerprint

```
/usr/share/keyrings/hashicorp-archive-keyring.gpg
-------------------------------------------------
pub   rsa4096 2023-01-10 [SC] [expires: 2028-01-09]
      798A EC65 4E5C 1542 8C8E  42EE AA16 FCBC A621 E701
uid           [ unknown] HashiCorp Security (HashiCorp Package Signing) <security+packaging@hashicorp.com>
sub   rsa4096 2023-01-10 [S] [expires: 2028-01-09]
```

**root@2a4f128a7e7e:/# echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] **

**https://apt.releases.hashicorp.com $(lsb_release -cs) main" | **

sudo tee /etc/apt/sources.list.d/hashicorp.list

root@2a4f128a7e7e:/# sudo apt update

root@2a4f128a7e7e:/# sudo apt-get install terraform

root@2a4f128a7e7e:/# terraform –version

```
root@2a4f128a7e7e:/# terraform —version
Terraform v1.7.5
on linux_amd64
```

INSTALL AWS CLI

root@DESKTOP-8OOG2HF:JENKINS_INPUT# docker container exec -u root -it -u root 2a4f128a7e7e /bin/bash

root@2a4f128a7e7e:/# apt update

root@2a4f128a7e7e:/# apt install awscli

root@2a4f128a7e7e:/# aws –version

root@DESKTOP-8OOG2HF:JENKINS_INPUT# docker container exec -it 2a4f128a7e7e /bin/bash

jenkins@2a4f128a7e7e:/$ aws configure

AWS Access Key ID [None]:

AWS Secret Access Key [None]:

Default region name [None:

Default output format [None]:

root@DESKTOP-8OOG2HF:JENKINS_INPUT# cat main.tf

```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.40.0"
    }
  }
}
provider "aws" {
  region = "ap-south-1"
  # Configuration options
}

resource "aws_instance" "instance" {
ami           = "ami-007020fd9c84e18c7"
instance_type = "t2.micro"

  tags = {
    Name = "Hello-ASFIYA"
  }
}
```

root@DESKTOP-8OOG2HF:JENKINS_INPUT# cat Jenkinsfile

```
Jenkinsfile
pipeline {
    agent any

    environment {
        AWS_ACCESS_KEY_ID = ''
        AWS_SECRET_ACCESS_KEY = ''
        AWS_DEFAULT_REGION = 'ap-south-1'
    }

    stages {
        stage('Environment Variables') {
            steps {
                script {
                    withCredentials([[
                        $class: 'AmazonWebServicesCredentialsBinding',
                        credentialsId: 'asfiya_AWS',
                        accessKeyVariable: 'AWS_ACCESS_KEY_ID',
                        secretKeyVariable: 'AWS_SECRET_ACCESS_KEY'
                    ]]) {
                        // Credentials will be automatically injected into environment variables
                    }
                }
            }
        }

        stage('Terraform Execution') {
            steps {
                script {
                    // Terraform initialization
                    echo 'Initializing Terraform...'
                    sh 'terraform init'
                }
            }
        }
```

```
        stage('Prompt for Terraform Action') {
            steps {
                script {
                    // Prompt user for input during runtime
                    def userInput = input(
                        id: 'userInput',
                        message: 'Select Terraform action to execute: apply or destroy',
                        ok: 'Continue',
                        parameters: [choice(
                            name: 'TerraAction',
                            choices: ['apply', 'destroy'],
                            description: 'Select Terraform action to execute'
                        )]
                    )
                    if (userInput == 'apply' && userInput == 'destroy') {
                        error('Both apply and destroy options cannot be selected. Please select only one.')
                    } else {
                        if (userInput == 'apply')  {
                            sh 'terraform apply -auto-approve'
                        } else if (userInput == 'destroy') {
                            sh 'terraform destroy -auto-approve'
                        }
                    }
                }
            }
        }
    }

    post {
        always {
            echo 'Cleaning up...'
            deleteDir()
        }
    }
}
```

root@DESKTOP-8OOG2HF:JENKINS_INPUT# git add .

root@DESKTOP-8OOG2HF:JENKINS_INPUT# git commit -m "Files have been added"

[master dd9ed21] Files have been added

1 file changed, 1 deletion(-)

root@DESKTOP-8OOG2HF:JENKINS_INPUT# git push origin master

Enumerating objects: 5, done.

Counting objects: 100% (5/5), done.

Delta compression using up to 4 threads

Compressing objects: 100% (3/3), done.

Writing objects: 100% (3/3), 350 bytes | 12.00 KiB/s, done.

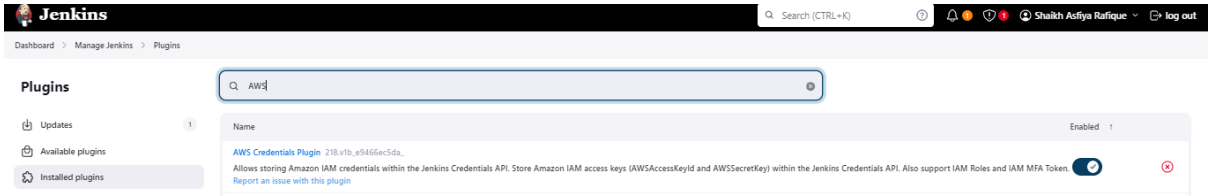
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0

remote: Resolving deltas: 100% (1/1), completed with 1 local object.

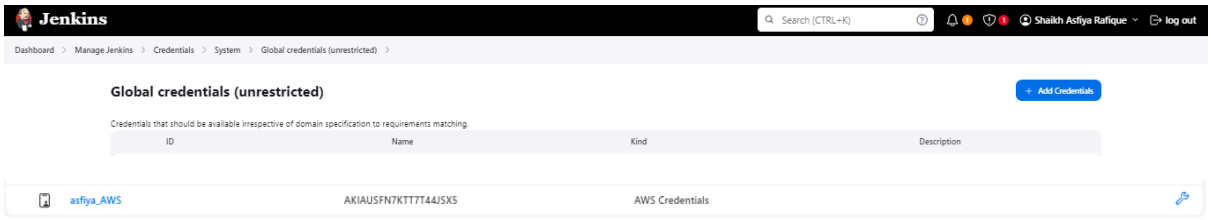
To github.com:ASFIASHAIKH/JENKINS_INPUT.git

146bde1..dd9ed21 master -> master

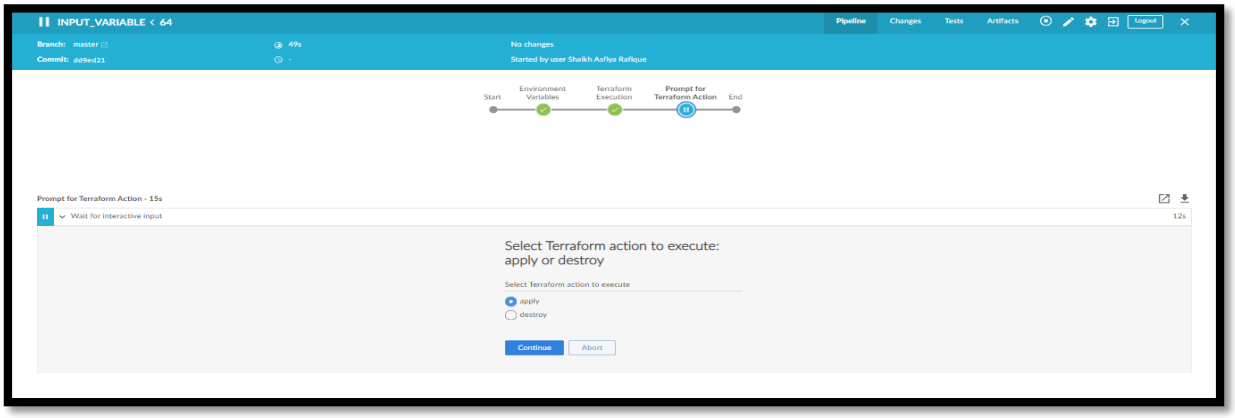
Install AWS Creadentials Plugin

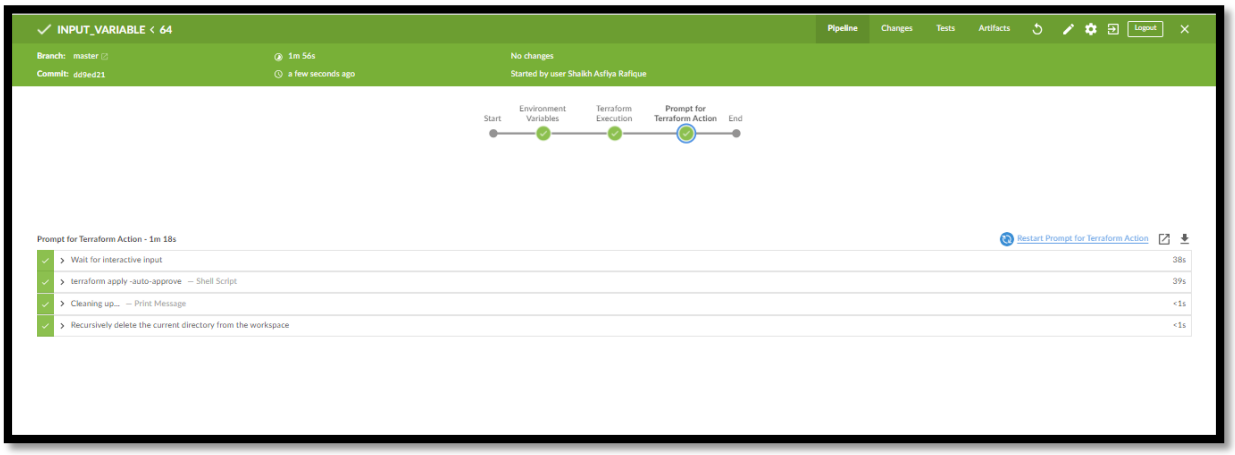


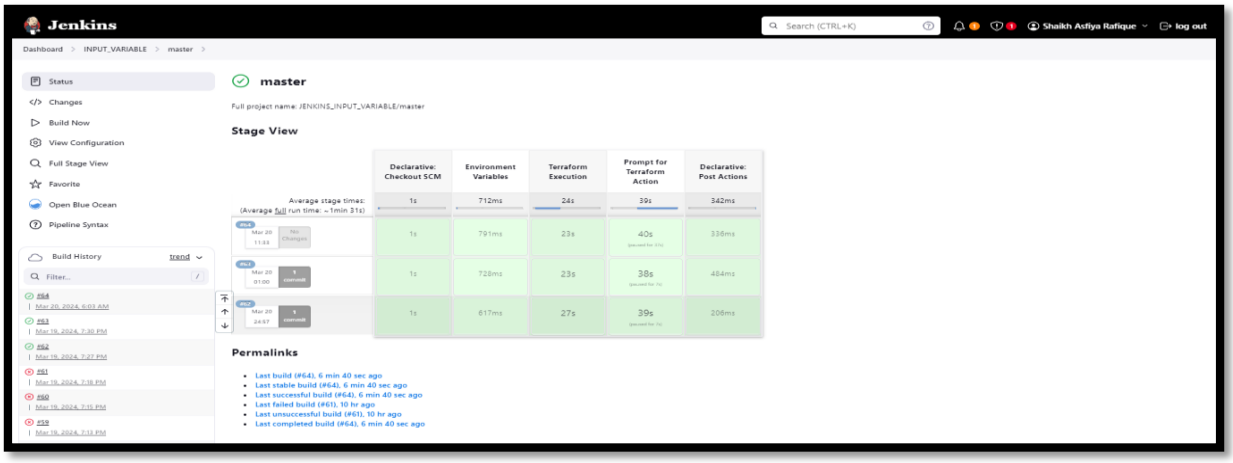
Create New Credential For Access Key And Secret Key



NOW BUILD THE JOB







Console Output

Started by user Shaikh Asfiya Rafique
> git rev-parse --resolve-git-dir /var/jenkins_home/caches/git-aec12c5c5145990adddb7b051578e5e5/.git # timeout=10
Setting origin to https://github.com/ASFIASHAIKH/JENKINS_INPUT.git
> git config remote.origin.url https://github.com/ASFIASHAIKH/JENKINS_INPUT.git # timeout=10
Fetching origin...
Fetching upstream changes from origin
> git --version # timeout=10
> git --version # 'git version 2.39.2'
> git config --get remote.origin.url # timeout=10
using GIT_SSH to set credentials
Verifying host key using known hosts file
You're using 'Known hosts file' strategy to verify ssh host keys, but your known_hosts file does not exist, please go to 'Manage Jenkins' -> 'Security' -> 'Git Host Key Verification Configuration' and configure host key verification.
> git fetch --tags --force --progress -- origin +refs/heads/*:refs/remotes/origin/* # timeout=10
Seen branch in repository origin/main
Seen branch in repository origin/master
Seen 2 remote branches
Obtained Jenkinsfile from dd9ed210905db8b215673d5e70abc437045e92e6
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/jenkins_home/workspace/JENKINS_INPUT_VARIABLE_master
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
using credential 01
Cloning the remote Git repository
Cloning with configured refspecs honoured and without tags
Cloning repository https://github.com/ASFIASHAIKH/JENKINS_INPUT.git
> git init /var/jenkins_home/workspace/JENKINS_INPUT_VARIABLE_master # timeout=10
Fetching upstream changes from https://github.com/ASFIASHAIKH/JENKINS_INPUT.git
> git --version # timeout=10
> git --version # 'git version 2.39.2'
using GIT_SSH to set credentials
Verifying host key using known hosts file
You're using 'Known hosts file' strategy to verify ssh host keys, but your known_hosts file does not exist, please go to 'Manage Jenkins' -> 'Security' -> 'Git Host Key Verification Configuration' and configure host key verification.

> git fetch --no-tags --force --progress -- https://github.com/ASFIASHAIKH/JENKINS_INPUT.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/ASFIASHAIKH/JENKINS_INPUT.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
Checking out Revision dd9ed210905db8b215673d5e70abc437045e92e6 (master)
> git config core.sparsecheckout # timeout=10
> git checkout -f dd9ed210905db8b215673d5e70abc437045e92e6 # timeout=10
Commit message: "Files have been added"
> git rev-list --no-walk dd9ed210905db8b215673d5e70abc437045e92e6 # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Environment Variables)
[Pipeline] script
[Pipeline] {
[Pipeline] withCredentials
Masking supported pattern matches of $AWS_ACCESS_KEY_ID or $AWS_SECRET_ACCESS_KEY
[Pipeline] {
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Terraform Execution)
[Pipeline] script
[Pipeline] {
[Pipeline] echo
Initializing Terraform...
[Pipeline] sh
+ terraform init

[0m [1mInitializing the backend... [0m

[0m [1mInitializing provider plugins... [0m
- Finding hashicorp/aws versions matching "5.40.0"...
- Installing hashicorp/aws v5.40.0...
- Installed hashicorp/aws v5.40.0 (signed by HashiCorp)

Terraform has created a lock file [1m.terraform.lock.hcl [0m to record the provider selections it made above. Include this file in your version control repository so that Terraform can guarantee to make the same selections by default when you run "terraform init" in the future. [0m

[0m [1m [32mTerraform has been successfully initialized! [0m [32m [0m [0m [32m
You may now begin working with Terraform. Try running "terraform plan" to see any changes that are required for your infrastructure. All Terraform commands should now work.

If you ever set or change modules or backend configuration for Terraform, rerun this command to reinitialize your working directory. If you forget, other commands will detect it and remind you to do so if necessary. [0m
[Pipeline] }

[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Prompt for Terraform Action)
[Pipeline] script
[Pipeline] {
[Pipeline] input
Input requested
Approved by Shaikh Asfiya Rafique
[Pipeline] sh
+ terraform apply -auto-approve

Terraform used the selected providers to generate the following execution
plan. Resource actions are indicated with the following symbols:
  [32m+ [0m create [0m

Terraform will perform the following actions:

```
[1m  # aws_instance.instance [0m will be created
[0m   [32m+ [0m [0m resource "aws_instance" "instance" {
    [32m+ [0m [0m ami                                  = "ami-007020fd9c84e18c7"
    [32m+ [0m [0m arn                                  = (known after apply)
    [32m+ [0m [0m associate_public_ip_address          = (known after apply)
    [32m+ [0m [0m availability_zone                    = (known after apply)
    [32m+ [0m [0m cpu_core_count                       = (known after apply)
    [32m+ [0m [0m cpu_threads_per_core                 = (known after apply)
    [32m+ [0m [0m disable_api_stop                     = (known after apply)
    [32m+ [0m [0m disable_api_termination              = (known after apply)
    [32m+ [0m [0m ebs_optimized                        = (known after apply)
    [32m+ [0m [0m get_password_data                    = false
    [32m+ [0m [0m host_id                              = (known after apply)
    [32m+ [0m [0m host_resource_group_arn              = (known after apply)
    [32m+ [0m [0m iam_instance_profile                 = (known after apply)
    [32m+ [0m [0m id                                   = (known after apply)
    [32m+ [0m [0m instance_initiated_shutdown_behavior = (known after apply)
    [32m+ [0m [0m instance_lifecycle                   = (known after apply)
    [32m+ [0m [0m instance_state                       = (known after apply)
    [32m+ [0m [0m instance_type                        = "t2.micro"
    [32m+ [0m [0m ipv6_address_count                   = (known after apply)
    [32m+ [0m [0m ipv6_addresses                       = (known after apply)
    [32m+ [0m [0m key_name                             = (known after apply)
    [32m+ [0m [0m monitoring                           = (known after apply)
    [32m+ [0m [0m outpost_arn                          = (known after apply)
    [32m+ [0m [0m password_data                        = (known after apply)
    [32m+ [0m [0m placement_group                      = (known after apply)
    [32m+ [0m [0m placement_partition_number           = (known after apply)
    [32m+ [0m [0m primary_network_interface_id         = (known after apply)
    [32m+ [0m [0m private_dns                          = (known after apply)
    [32m+ [0m [0m private_ip                           = (known after apply)
    [32m+ [0m [0m public_dns                           = (known after apply)
    [32m+ [0m [0m public_ip                            = (known after apply)
    [32m+ [0m [0m secondary_private_ips                = (known after apply)
    [32m+ [0m [0m security_groups                      = (known after apply)
    [32m+ [0m [0m source_dest_check                    = true
    [32m+ [0m [0m spot_instance_request_id             = (known after apply)
    [32m+ [0m [0m subnet_id                            = (known after apply)
    [32m+ [0m [0m tags                                 = {
      [32m+ [0m [0m "Name" = "Hello-ASFIYA"
    }
```

```
      [32m+   [0m   [0m tags_all                          = {
        [32m+   [0m   [0m "Name" = "Hello-ASFIYA"
      }
      [32m+   [0m   [0m tenancy                         = (known after apply)
      [32m+   [0m   [0m user_data                       = (known after apply)
      [32m+   [0m   [0m user_data_base64                = (known after apply)
      [32m+   [0m   [0m user_data_replace_on_change     = false
      [32m+   [0m   [0m vpc_security_group_ids          = (known after apply)
    }

  [1mPlan:   [0m 1 to add, 0 to change, 0 to destroy.
  [0m   [0m   [1maws_instance.instance: Creating...   [0m   [0m
  [0m   [1maws_instance.instance: Still creating... [10s elapsed]   [0m   [0m
  [0m   [1maws_instance.instance: Still creating... [20s elapsed]   [0m   [0m
  [0m   [1maws_instance.instance: Still creating... [30s elapsed]   [0m   [0m
  [0m   [1maws_instance.instance: Creation complete after 33s [id=i-0c59e79584b6ad387]   [0m
  [0m   [1m   [32m
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
  [0m
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
Cleaning up...
[Pipeline] deleteDir
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Instance Created Successfully