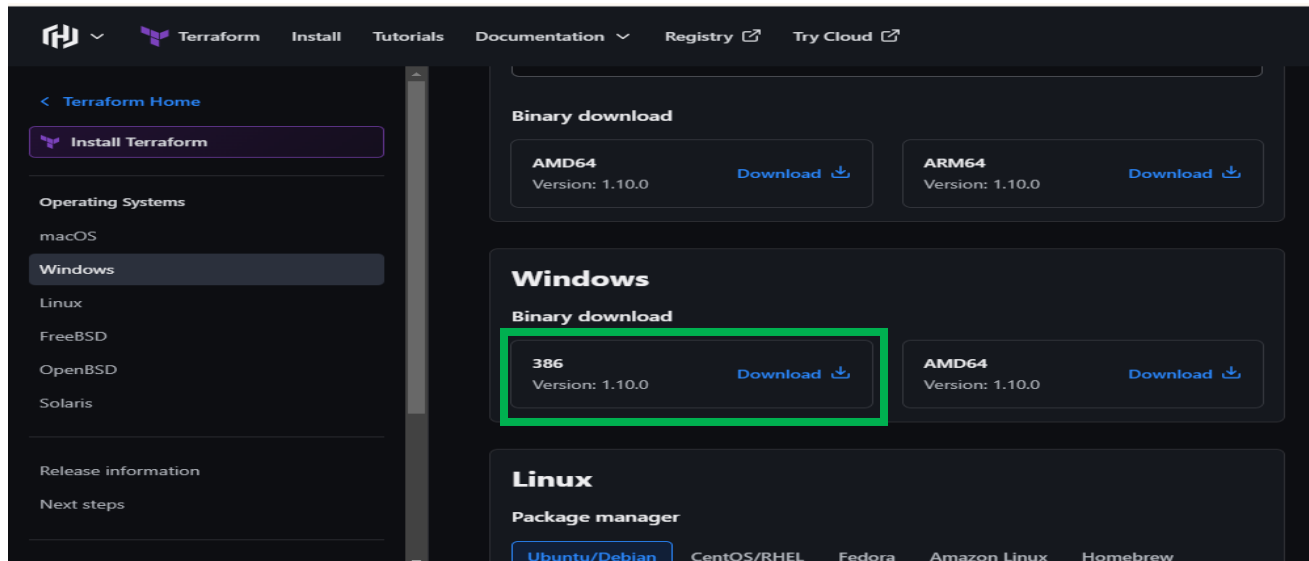


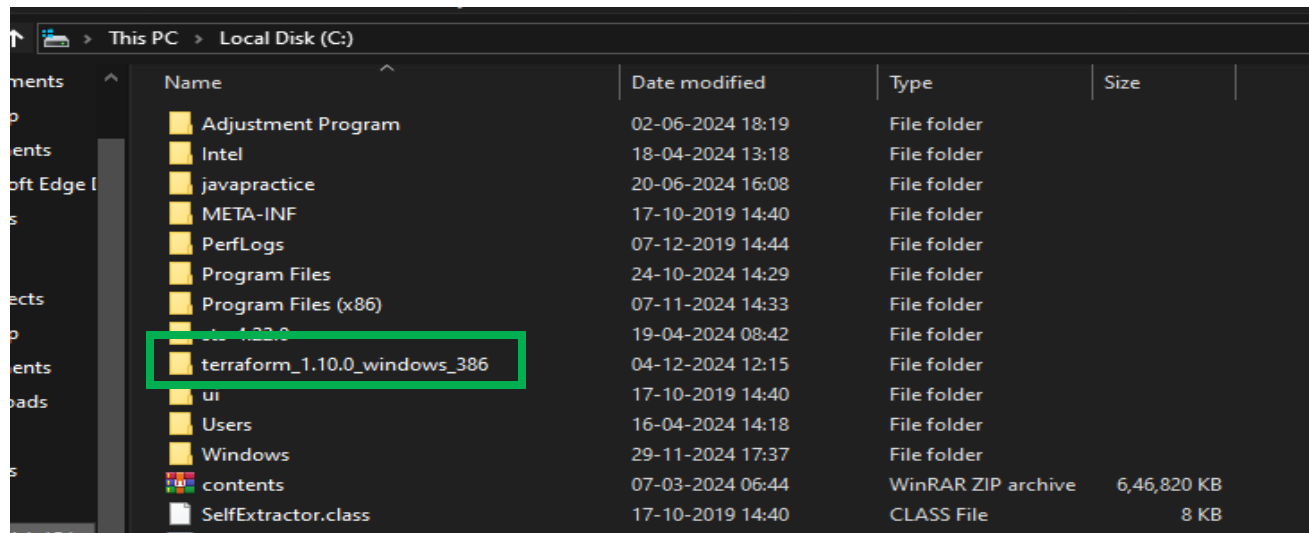
# Terraform

## 1) Install Terraform on your PC

- Steps to Install Terraform on Your PC
  - Download Terraform
    - Visit the <https://developer.hashicorp.com/terraform/install>

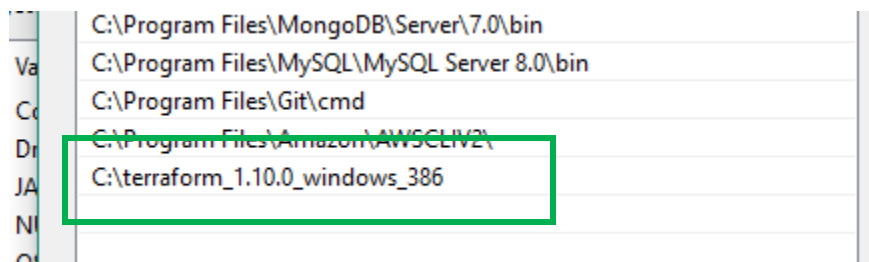


- Extract the Executable



- Add Terraform to Your System PATH

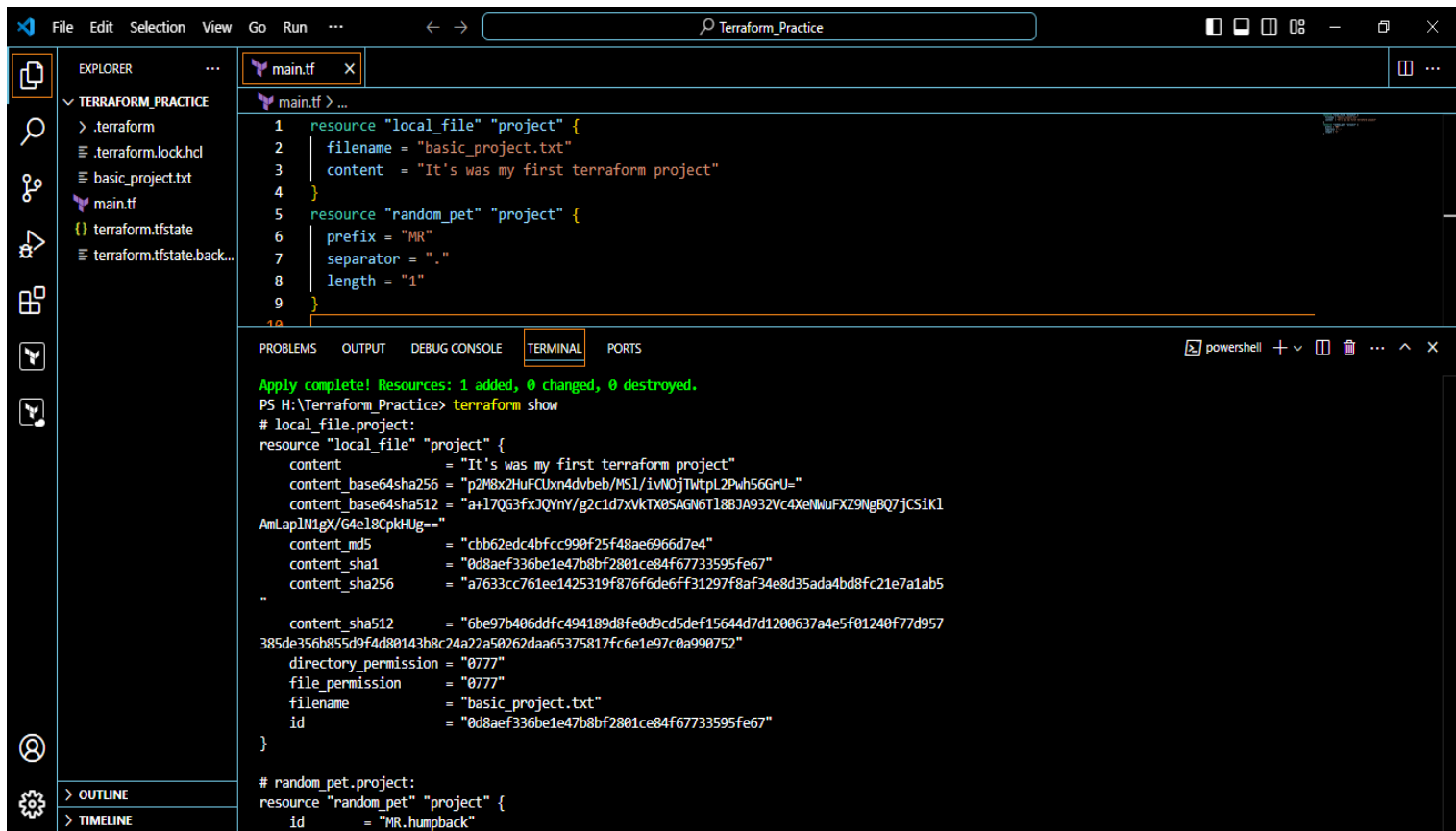
Go to → Checkout **Edit the system environment variables** and set the path



- Verify the Installation

```
PS C:\Users\DELL> terraform -version
Terraform v1.10.0
on windows_386
PS C:\Users\DELL>
```

2) Execute all the templates shown in video.



### 3) Note down below points,

#### Terraform Init

```
main.tf X
main.tf > ...
1 resource "local_file" "project" {
2   filename = "basic_project.txt"
3   content  = "It's was my first terraform project"
4 }
5 resource "random_pet" "project" {
6   prefix = "MR"
7   separator = "."
8   length = "1"
9 }
10

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS H:\Terraform_Practice> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/local...
- Installing hashicorp/local v2.5.2...
- Installed hashicorp/local v2.5.2 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl 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.

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.

rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

#### Terraform Plan

```
main.tf X
main.tf > ...
1 resource "local_file" "project" {
2   filename = "basic_project.txt"
3   content  = "It's was my first terraform project"
4 }
5 resource "random_pet" "project" {
6   prefix = "MR"
7   separator = "."
8   length = "1"
9 }
10

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS H:\Terraform_Practice> terraform plan

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

Terraform will perform the following actions:

# local_file.project will be created
+ resource "local_file" "project" {
+   content          = "It's was my first terraform project"
+   content_base64sha256 = (known after apply)
+   content_base64sha512 = (known after apply)
+   content_md5       = (known after apply)
+   content_sha1       = (known after apply)
+   content_sha256     = (known after apply)
+   content_sha512     = (known after apply)
+   directory_permission = "0777"
+   file_permission    = "0777"
+   filename          = "basic_project.txt"
}

Plan: 1 to add, 0 to change, 0 to destroy.
```

#### Terraform Apply

```
main.tf X
main.tf > ...
1 resource "local_file" "project" {
2   filename = "basic_project.txt"
3   content  = "It's was my first terraform project"
4 }
5 resource "random_pet" "project" {
6   prefix = "MR"
7   separator = "."
8   length = "1"
9 }
10

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS H:\Terraform_Practice> terraform apply
local_file.project: Refreshing state... [id=0d8aef336be1e47b8bf2801ce84f67733595fe67]

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

Terraform will perform the following actions:

# random_pet.project will be created
+ resource "random_pet" "project" {
+   id          = (known after apply)
+   length      = 1
+   prefix      = "MR"
+   separator    = "."
}

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Only 'yes' will be accepted to approve.

Enter a value: yes
```

#### Terraform Provider.

```
main.tf > ...
1 resource "local_file" "project" {
2   filename = "basic_project.txt"
3   content  = "It's was my first terraform project"
4 }
5 resource "random_pet" "project" {
6   prefix = "MR"
7   separator = "."
8   length = "1"
9 }
10

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

}

# random_pet.project:
resource "random_pet" "project" {
  id          = "MR.humpback"
  length      = 1
  prefix      = "MR"
  separator    = "."
}

PS H:\Terraform_Practice> terraform providers

Providers required by configuration:
└─ provider[registry.terraform.io/hashicorp/local]
└─ provider[registry.terraform.io/hashicorp/random]

Providers required by state:

provider[registry.terraform.io/hashicorp/local]

provider[registry.terraform.io/hashicorp/random]
```

#### 4) Integrate a sample Terraform template in Jenkins.

##### CLI

```
36 sudo yum install -y yum-utils shadow-utils
37 sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
38 sudo yum -y install terraform
39 which terraform
310 find / -name terraform
311 terraform -version
312 cd /usr/bin/terraform
313 cd /usr/bin/
314 ls
315 cd /bin/
```

##### GUI

The screenshot shows the Jenkins 'Manage Jenkins' > 'Tools' page. Under the 'Terraform installations' section, a new installation named 'Terraform' is being configured. The 'Install directory' field is set to '/usr/bin/' and is highlighted with a green box. The 'Install automatically' checkbox is unchecked. The 'Save' button is visible at the bottom.

The screenshot shows the Jenkins 'Console Output' for a pipeline build. The output shows the pipeline starting, checking out the code from a GitHub repository, and running the 'Terraform Init' step. The 'Terraform Init' step is highlighted with a green box. The output also shows the 'Terraform Apply' step.

##### Pipeline

###### Definition

Pipeline script

###### Script

```
1 pipeline{
2   agent any
3   tools {
4     terraform 'Terraform'
5   }
6   stages{
7     stage('Git Checkout'){
8       steps{
9         git branch: 'main', credentialsId: 'ghp_cY2KmsvHtMhPpYzZuUfrCUivQt12aD3hHfje', url: 'https://github.com/ChakriSaiCharan-T/terraform-practice'
10      }
11    }
12    stage('Terraform Init'){
13      steps{
14        sh 'terraform init'
15      }
16    }
17    stage('Terraform Apply'){
18    }
```

☒ Use Groovy Sandbox

Pipeline Syntax

Save

Apply

```
[root@ip-172-31-19-218 workspace]# cd terraform
[root@ip-172-31-19-218 terraform]# ll
total 12
-rw-r--r-- 1 jenkins jenkins 124 Dec  5 09:42 main.tf
-rwxr-xr-x 1 jenkins jenkins  45 Dec  5 09:42 project.txt
-rw-r--r-- 1 jenkins jenkins 1620 Dec  5 09:42 terraform.tfstate
[root@ip-172-31-19-218 terraform]#
```

5) Watch the terraform-02 video.

The screenshot shows the VS Code editor with two files open: `main.tf` and `variables.tf`. The `main.tf` file contains a `resource "local_file" "project"` block and a `resource "random_pet" "project"` block. The `variables.tf` file contains a `variable "filename"` and a `variable "content"`. The terminal at the bottom shows the output of a `terraform apply` command, indicating that the `local_file.project` resource was successfully created.

```
main.tf > resource "local_file" "project" {
  filename = var.filename
  content = var.content
}

resource "random_pet" "project" {
  prefix = "MR"
  separator = "."
  length = "1"
}

variables.tf > variable "content"
variable "filename" {
  default = "basic_project.txt"
  type = string
}
variable "content" {
  default = "It's was my first project"
}
```

```
local_file.project: Destroying... [id=0d8aef336be1e47b8bf2801ce84f67733595fe67]
local_file.project: Destruction complete after 0s
local_file.project: Creating...
local_file.project: Creation complete after 0s [id=9a604f5f1cdaa0e4da6949fac9148ea56114dd2f]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
PS H:\Terraform_Practice>
```

6) Execute all the templates shown in video.

The screenshot shows the VS Code editor with the `main.tf` and `variables.tf` files. The `main.tf` file has been updated to include a `resource "random_pet" "project"` block. The `variables.tf` file has been updated to include a `variable "length"` and a `variable "prefix"`. The terminal at the bottom shows the output of a `terraform apply` command, indicating that the `random_pet.project` resource was successfully created.

```
main.tf > resource "random_pet" "project" {
  prefix = var.prefix
  separator = "."
  length = var.length
}

variables.tf > variable "length" > default
variable "filename" {
  default = "basic_project.txt"
  type = string
}
variable "content" {
  default = "It's was my first project"
  type = string
}
variable "prefix" {
  default = "MR"
  type = string
}
variable "length" {
  default = "2"
  type = number
}
```

```
random_pet.project: Destroying... [id=MR.humpback]
random_pet.project: Destruction complete after 0s
random_pet.project: Creating...
random_pet.project: Creation complete after 0s [id=MR.diverse.anemone]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
PS H:\Terraform_Practice>
```

The screenshot shows the VS Code editor with the `main.tf` and `variables.tf` files. The `main.tf` file has been updated to include a `resource "random_pet" "project"` block. The `variables.tf` file has been updated to include a `variable "length"` and a `variable "prefix"`. The terminal at the bottom shows the output of a `terraform apply` command, indicating that the `random_pet.project` resource was successfully created.

```
main.tf > resource "random_pet" "project" {
  prefix = "MR"
  separator = "."
  length = "1"
}

variables.tf > variable "content"
variable "filename" {
  default = "basic_project.txt"
  type = string
}
variable "content" {
  default = "It's was my first project in terraf"
}
```

```
random_pet.project: Destroying... [id=MR.diverse.anemone]
random_pet.project: Destruction complete after 0s
local_file.project: Destruction complete after 1s
random_pet.project: Creating...
local_file.project: Creating...
local_file.project: Creation complete after 0s [id=3085c11ed48369d166edb140030db031807dc85]
random_pet.project: Creation complete after 0s [id=MR.gazelle]

Apply complete! Resources: 2 added, 0 changed, 2 destroyed.
PS H:\Terraform_Practice>
```

ViewGoRun...Terraform\_Practice

main.tf ×variables.tf ×variables.tf ×project007.txt ×

main.tf > resource "random\_pet" "project" > prefix

1 resource "local\_file" "project" {  
2 filename = var.filename  
3 content = var.content  
4 }  
5 resource "random\_pet" "project" {  
6 prefix = "MR"  
7 separator = "."  
8 length = "1"  
9 }  
10

variables.tf > variable "content"

1 variable "filename" {  
2 }  
3 }  
4 variable "content" {  
5 }  
6 }

project007.txt > project007.txt  
1 It's was my second project

PROBLEMSOUTPUTDEBUG CONSOLETERMINALPORTS

random\_pet.project: Creation complete after 0s [id=MR.gazelle]

Apply complete! Resources: 2 added, 0 changed, 2 destroyed.  
PS H:\Terraform\_Practice> terraform apply -var "filename=project007.txt" -var "content=It's was my second project"  
random\_pet.project: Refreshing state... [id=MR.gazelle]  
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:  
-/+ destroy and then create replacement  
Terraform will perform the following actions:  
# local\_file.project must be replaced  
-/+ resource "local\_file" "project" {  
~ content = "It's was my first project in terraform" -> "It's was my second project" # forces replacement  
~ content\_base64sha256 = "cgCNTV/nCC+CjJV8qaj42s+4hiD0TRwUVJrENBwEg8w=" -> (known after apply)  
~ content\_base64sha512 = "NlB268F06x1o3myh3dCLrt1iaD55C1VTgMah0Th/Emxkdw13vom6YOLaPkwWChXvXmE7Hbly5NEiixKtMiwBA=" -> (known after apply)

FileEditSelectionViewGoRun...Terraform\_Practice

EXPLORERTERRAF...main.tf ×variables.tf ×variables.tf ×project.txt ×

main.tf > ...

1 resource "local\_file" "project" {  
2 filename = var.filename  
3 content = var.content  
4 }  
5 resource "random\_pet" "project" {  
6 prefix = var.prefix  
7 separator = "."  
8 length = "1"  
9 }  
10

variables.tf > variable "prefix"

1 variable "filename" {  
2 }  
3 }  
4 variable "content" {  
5 }  
6 }  
7 variable "prefix" {  
8 }  
9 }

project.txt > project.txt  
1 It's my 3rd project in terraform

PROBLEMSOUTPUTDEBUG CONSOLETERMINALPORTS

Do you want to perform these actions?  
Terraform will perform the actions described above.  
Only 'yes' will be accepted to approve.  
Enter a value: yes  
local\_file.project: Destroying... [id=a636dc5c1e1d7a2af5fe449d00449eb09e86adbb]  
random\_pet.project: Destroying... [id=MR.gazelle]  
local\_file.project: Destruction complete after 0s  
random\_pet.project: Destruction complete after 0s  
local\_file.project: Creating...  
random\_pet.project: Creating...  
local\_file.project: Creation complete after 0s [id=7793b5004c103e00f29929c74b71eb239c1b7e8d]  
random\_pet.project: Creation complete after 0s [id=haii.colt]  
Apply complete! Resources: 2 added, 0 changed, 2 destroyed.  
PS H:\Terraform\_Practice> ]

7) Integrate terraform in jenkins using Terraform plugin.

Plugins

Download progress

2

Preparation

- Checking internet connectivity
- Checking update center connectivity
- Success

Oracle Java SE Development Kit Installer

Success

Command Agent Launcher

Success

Terraform

Success

Loading plugin extensions

Success

terraform-practice / main.tf

ChakriSaiCharan-T Update main.tf

Code Blame 4 lines (4 loc) · 124 Bytes Code 55% faster with GitHub Copilot

```
1 resource "local_file" "project" {
2   filename = "project.txt"
3   content  = "I have using integrating terraform in jenkins"
4 }
```

ChakriSaiCharan-T / terraform-practice

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

terraform-practice Public

main 1 Branch 0 Tags

Go to file Add file Code

ChakriSaiCharan-T Update main.tf 476bbc3 · 35 minutes ago 2 Commits

main.tf Update main.tf 35 minutes ago

README

← → ↺ Not secure 54.173.10.84:8080/manage/configureTools/

Dashboard > Manage Jenkins > Tools

Maven installations

Maven installations Edited

Terraform installations

Add Terraform

Terraform

Name

Terraform

Install directory

/usr/bin/

☐ Install automatically

Add Terraform

Save Apply

Pipeline

Definition

Pipeline script

Script

```
1 pipeline{
2   agent any
3   tools {
4     terraform 'Terraform'
5   }
6   stages{
7     stage('Git Checkout'){
8       steps{
9         git branch: 'main', credentialsId: 'ghp_cY2KnsvHtM0PpyZu0frCUlQ12a03MHfje', url: 'https://github.com/ChakriSaiCharan-T/terraform-practice'
10      }
11    }
12    stage('Terraform Init'){
13      steps{
14        sh 'terraform init'
15      }
16    }
17    stage('Terraform Apply'){
```

☒ Use Groovy Sandbox

Pipeline Syntax

Save Apply

Status

Changes

Console Output

Edit Build Information

Delete build #4

Timings

Git Build Data

Pipeline Overview

Pipeline Console

Restart from Stage

Replay

Pipeline Steps

Workspaces

Previous Build

Console Output

Started by user [admin](#)

[Pipeline] Start of Pipeline

[Pipeline] node

Running on [Jenkins](#) in /var/lib/jenkins/workspace/terraform

[Pipeline] {

[Pipeline] stage

[Pipeline] { (Declarative: Tool Install)

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] }

[Pipeline] // stage

[Pipeline] withEnv

[Pipeline] {

[Pipeline] stage

[Pipeline] { (Git Checkout)

[Pipeline] tool

[Pipeline] envVarsForTool

[Pipeline] withEnv

[Pipeline] {

[Pipeline] git

The recommended git tool is: NONE

using credential [ghp\\_cY2XMSvHTMPPYzMuFrcUJvQt12A03HHfje](#)

> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/terraform/.git # timeout=10

Fetching changes from the remote Git repository

> git config remote.origin.url <https://github.com/ChakrisaiCharan-7/terraform-practice> # timeout=10

Fetching upstream changes from <https://github.com/ChakrisaiCharan-7/terraform-practice>

> git --version # timeout=10

> git --version # 'git version 2.40.1'