Lesson 13 Demo 02

Working with Variables and Versions on Terraform Cloud

Objective: To use Terraform Cloud for managing infrastructure by creating workspaces and defining variables to ensure consistent and reliable deployments

Tools required: Terraform Cloud

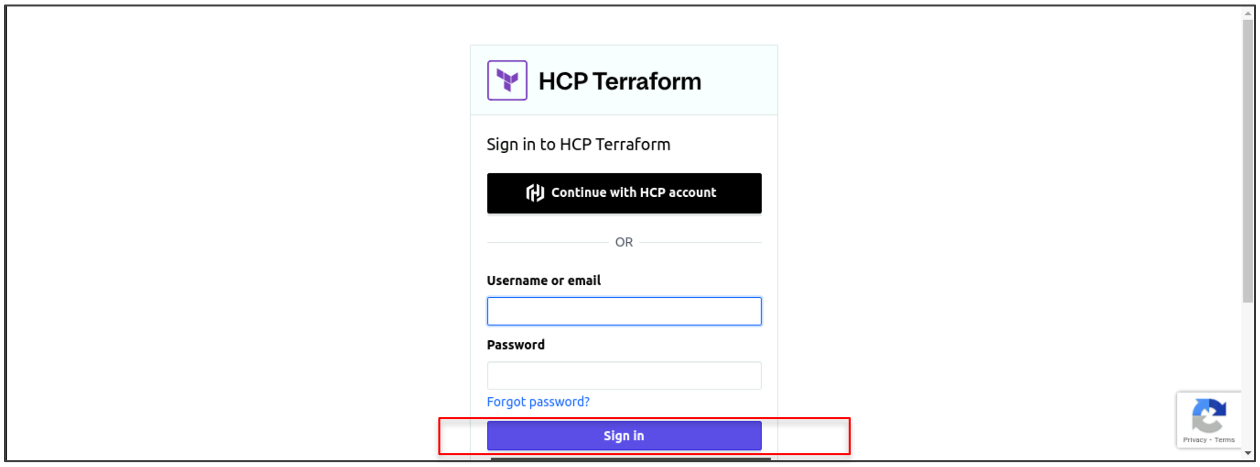
Prerequisites: Ensure you have created and implemented the AWS access key and secret key before starting this demo. Refer to Lesson 08 Demo 02 for detailed steps.

Steps to be followed:

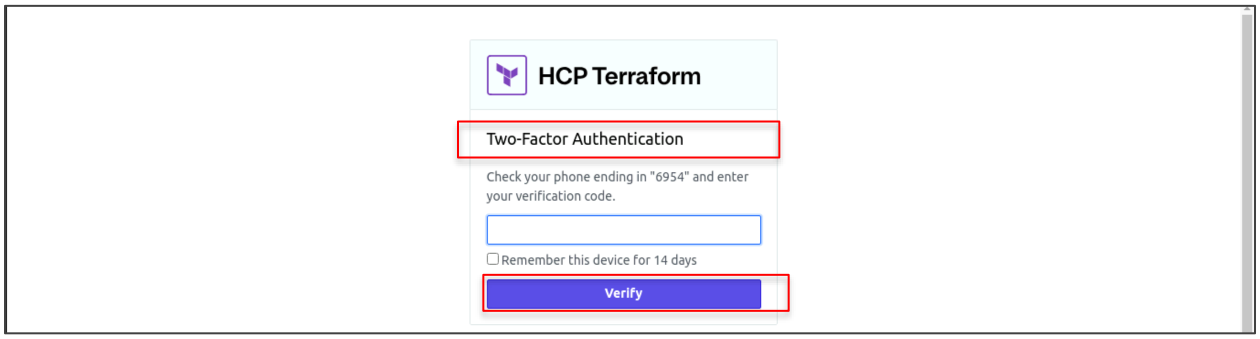
1. Sign in to Terraform Cloud platform
2. Create an organization and workspace
3. Configure Terraform CLI for Terraform Cloud
4. Define and use variables on Terraform Cloud
5. Run the Terraform plan and apply

Step 1: Sign in to Terraform Cloud platform

1.1 Enter the required details and click on **Sign In** by using the following URL: **https://app.terraform.io/session**



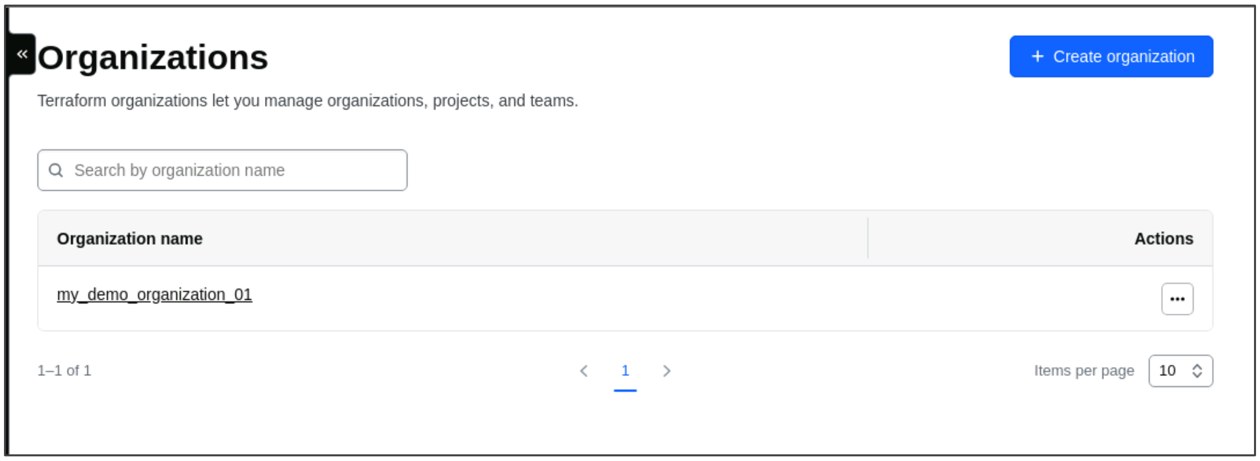
1.2 **Verify** the **Two-Factor Authentication:**



Two-Factor Authentication

Check your phone ending in "6954" and enter your verification code.

☐ Remember this device for 14 days



Organizations

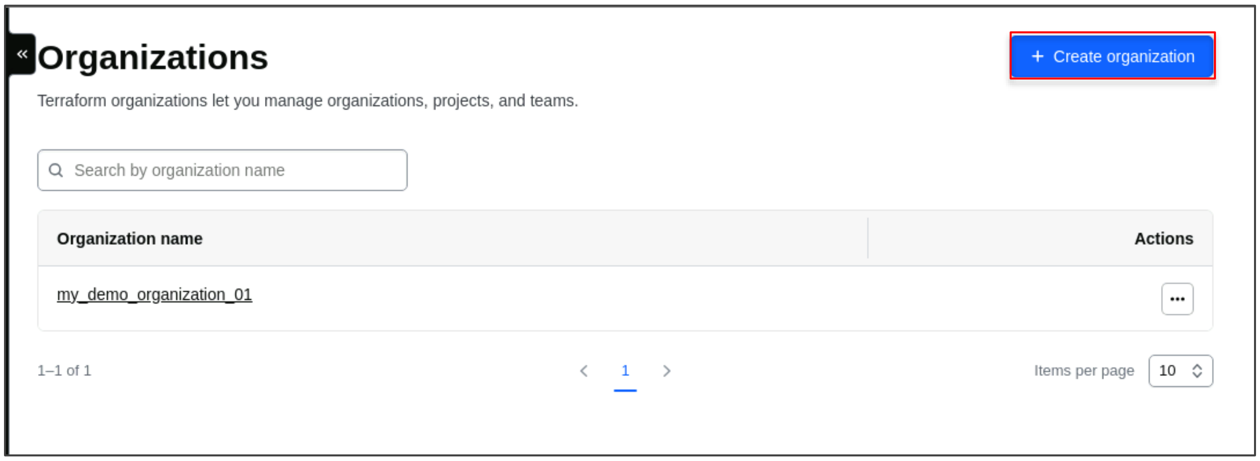
Terraform organizations let you manage organizations, projects, and teams.

Organization name	Actions
my_demo_organization_01	•••

1–1 of 1 — 1 — Items per page 10

Step 2: Create an organization and workspace

2.1 Click on **Create organization:**



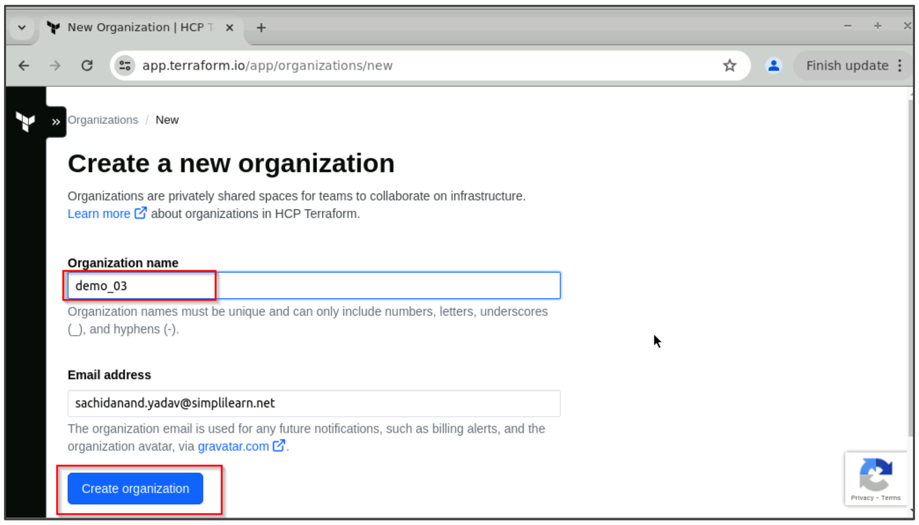
Organizations

Terraform organizations let you manage organizations, projects, and teams.

Organization name	Actions
my_demo_organization_01	•••

1–1 of 1 — 1 — Items per page 10

2.2 Enter the **Organization name** as **demo_03** and click on **Create organization:**



Create a new organization

Organizations are privately shared spaces for teams to collaborate on infrastructure.
Learn more about organizations in HCP Terraform.

Organization name

demo_03

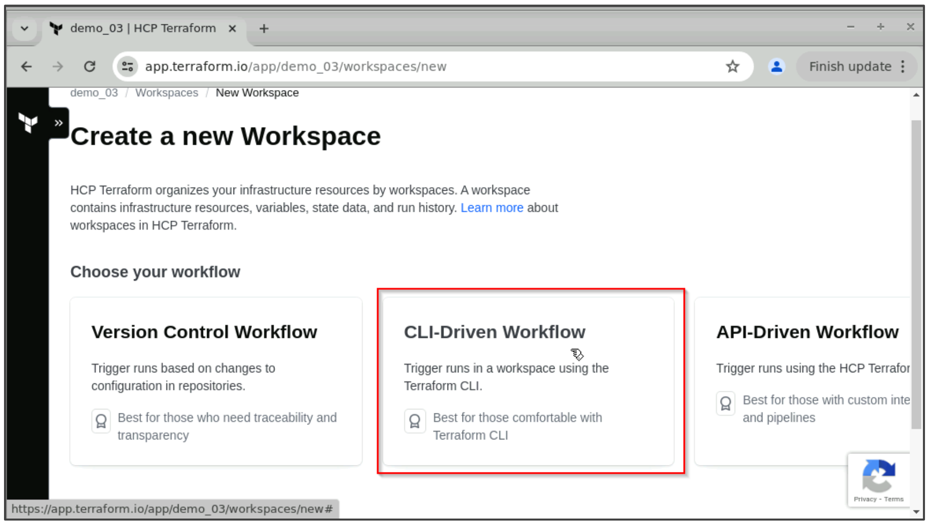
Organization names must be unique and can only include numbers, letters, underscores (_), and hyphens (-).

Email address

sachidanand.yadav@simplilearn.net

The organization email is used for any future notifications, such as billing alerts, and the organization avatar, via gravatar.com.

2.3 In the section of **Choose your workflow**, select **CLI-Driven Workflow:**


demo_03 | HCP Terraform
app.terraform.io/app/demo_03/workspaces/new
Finish update
demo_03 / Workspaces / New Workspace

Create a new Workspace

HCP Terraform organizes your infrastructure resources by workspaces. A workspace contains infrastructure resources, variables, state data, and run history. Learn more about workspaces in HCP Terraform.

Choose your workflow

Version Control Workflow

Trigger runs based on changes to configuration in repositories.

Best for those who need traceability and transparency

CLI-Driven Workflow

Trigger runs in a workspace using the Terraform CLI.

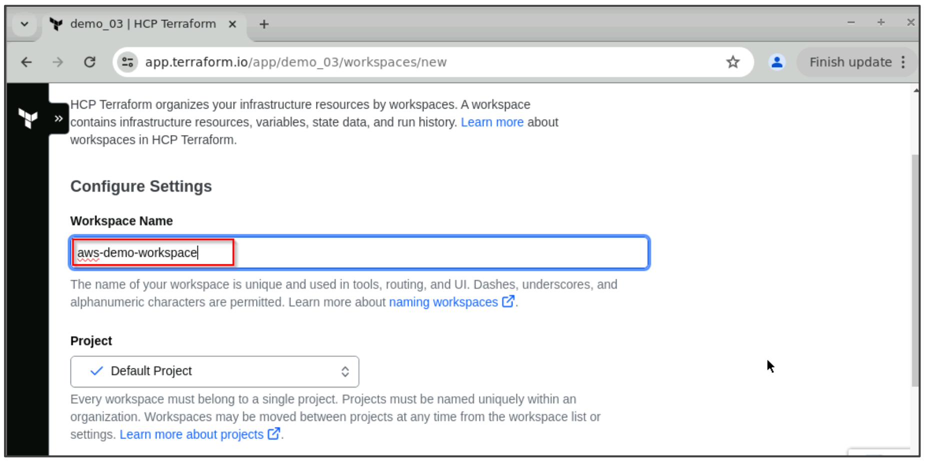
Best for those comfortable with Terraform CLI

API-Driven Workflow

Trigger runs using the HCP Terrafor

Best for those with custom inte and pipelines

2.4 Enter the **Workspace Name** as **aws-demo-workspace:**


demo_03 | HCP Terraform

app.terraform.io/app/demo_03/workspaces/new

HCP Terraform organizes your infrastructure resources by workspaces. A workspace contains infrastructure resources, variables, state data, and run history. Learn more about workspaces in HCP Terraform.

Configure Settings

Workspace Name

aws-demo-workspace

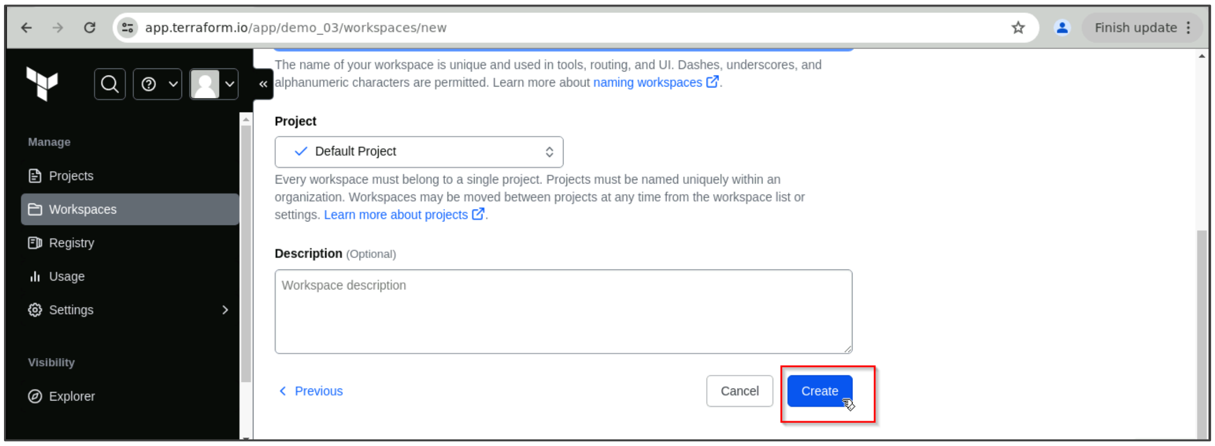
The name of your workspace is unique and used in tools, routing, and UI. Dashes, underscores, and alphanumeric characters are permitted. Learn more about naming workspaces.

Project

Default Project

Every workspace must belong to a single project. Projects must be named uniquely within an organization. Workspaces may be moved between projects at any time from the workspace list or settings. Learn more about projects.

2.5 Scroll down and click on **Create:**


app.terraform.io/app/demo_03/workspaces/new

The name of your workspace is unique and used in tools, routing, and UI. Dashes, underscores, and alphanumeric characters are permitted. Learn more about naming workspaces.

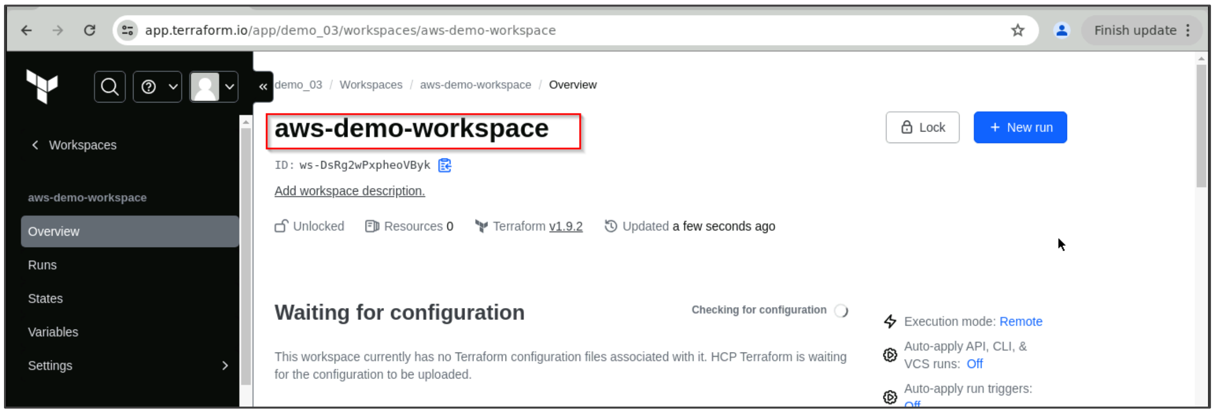
Project

Default Project

Every workspace must belong to a single project. Projects must be named uniquely within an organization. Workspaces may be moved between projects at any time from the workspace list or settings. Learn more about projects.

Description (Optional)

Workspace description


app.terraform.io/app/demo_03/workspaces/aws-demo-workspace

demo_03 / Workspaces / aws-demo-workspace / Overview

aws-demo-workspace

ID: ws-DsRg2wPxpheoVByk

Add workspace description.

Unlocked Resources 0 Terraform v1.9.2 Updated a few seconds ago

Waiting for configuration

Checking for configuration

This workspace currently has no Terraform configuration files associated with it. HCP Terraform is waiting for the configuration to be uploaded.

Execution mode: Remote

Auto-apply API, CLI, & VCS runs: Off

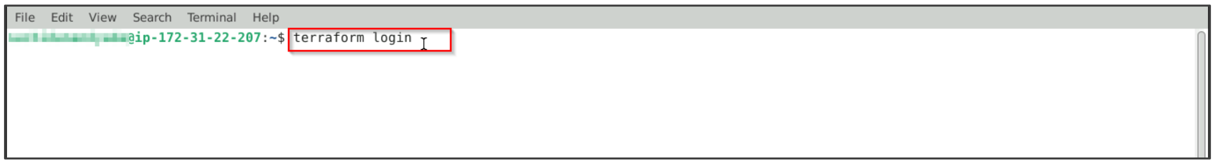
Auto-apply run triggers: Off

The workspace will be created as shown in the above screenshot.

Step 3: Configure Terraform CLI for Terraform Cloud

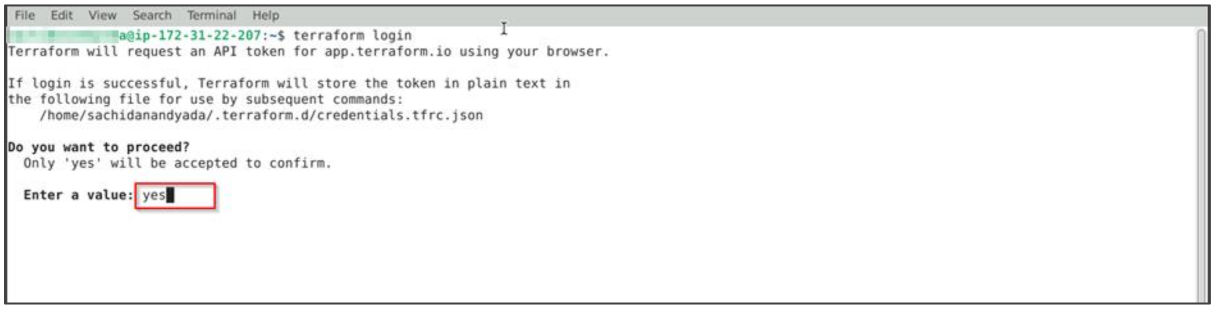
3.1 Go to the terminal and run the following command to log in to Terraform Cloud:

terraform login



```
[redacted]@ip-172-31-22-207:~$ terraform login
```

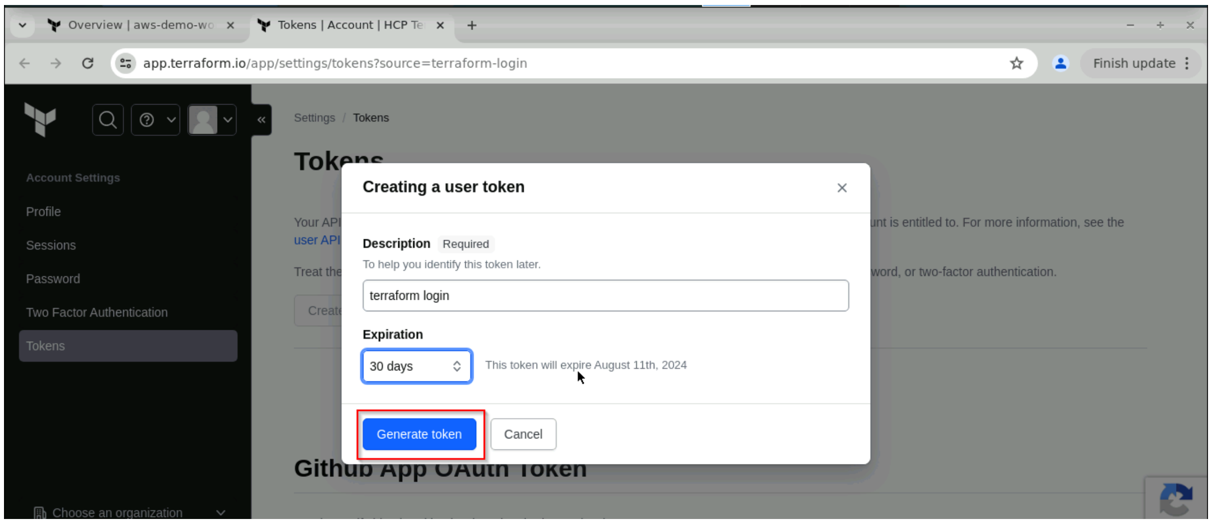
3.2 When prompted, proceed by typing **yes:**



```
[redacted]a@ip-172-31-22-207:~$ terraform login
Terraform will request an API token for app.terraform.io using your browser.

If login is successful, Terraform will store the token in plain text in
the following file for use by subsequent commands:
    /home/sachidanandyada/.terraform.d/credentials.tfrc.json

Do you want to proceed?
  Only 'yes' will be accepted to confirm.

  Enter a value: yes
```

The Terraform Cloud interface will automatically open.

3.3 Create a user token by clicking on **Generate token:**



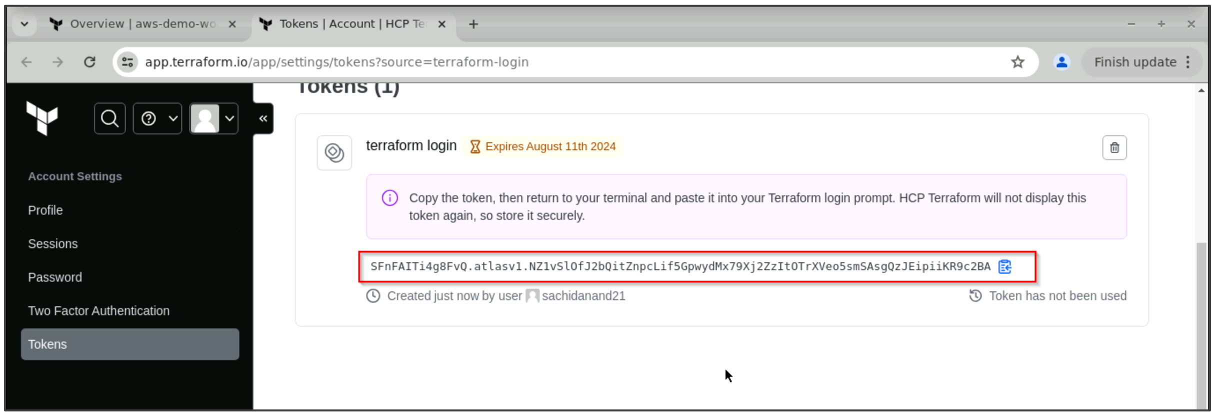
Creating a user token

Description Required
To help you identify this token later.
terraform login

Expiration
30 days
This token will expire August 11th, 2024

Generate token Cancel

3.4 Scroll down and copy the generated token:


Overview | aws-demo-wo
Tokens | Account | HCP Te
app.terraform.io/app/settings/tokens?source=terraform-login
Finish update
Account Settings
Profile
Sessions
Password
Two Factor Authentication
Tokens
Tokens (1)

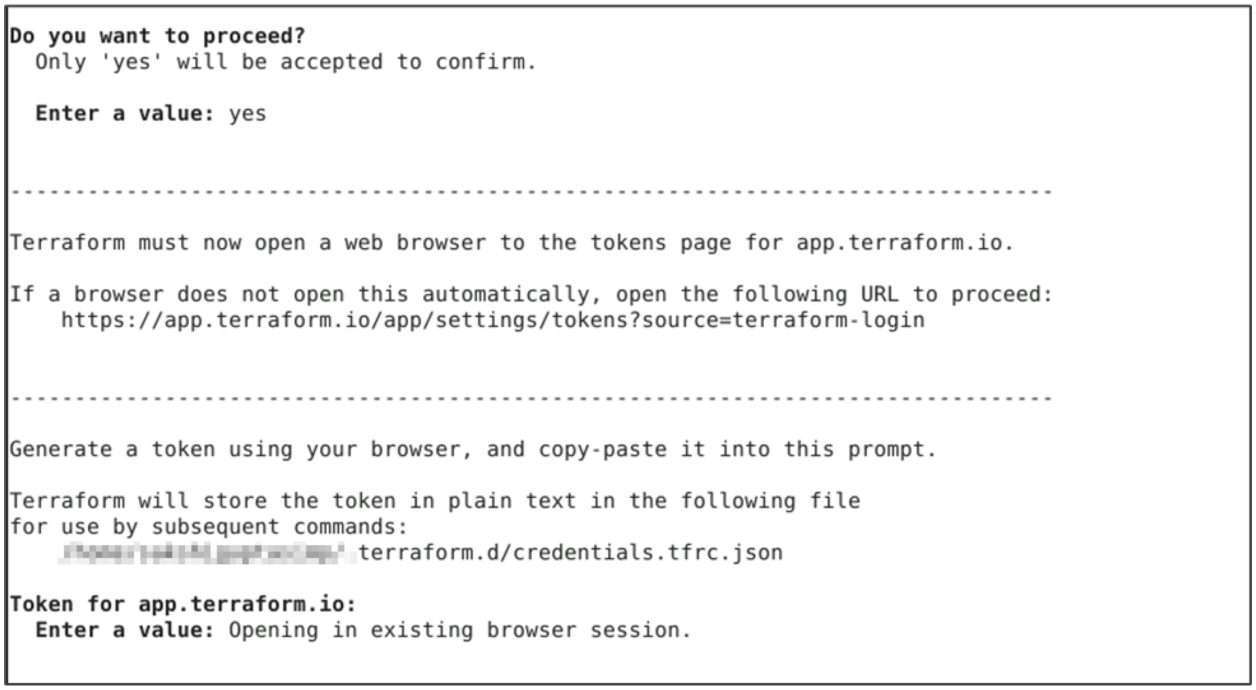
terraform login Expires August 11th 2024

Copy the token, then return to your terminal and paste it into your Terraform login prompt. HCP Terraform will not display this token again, so store it securely.

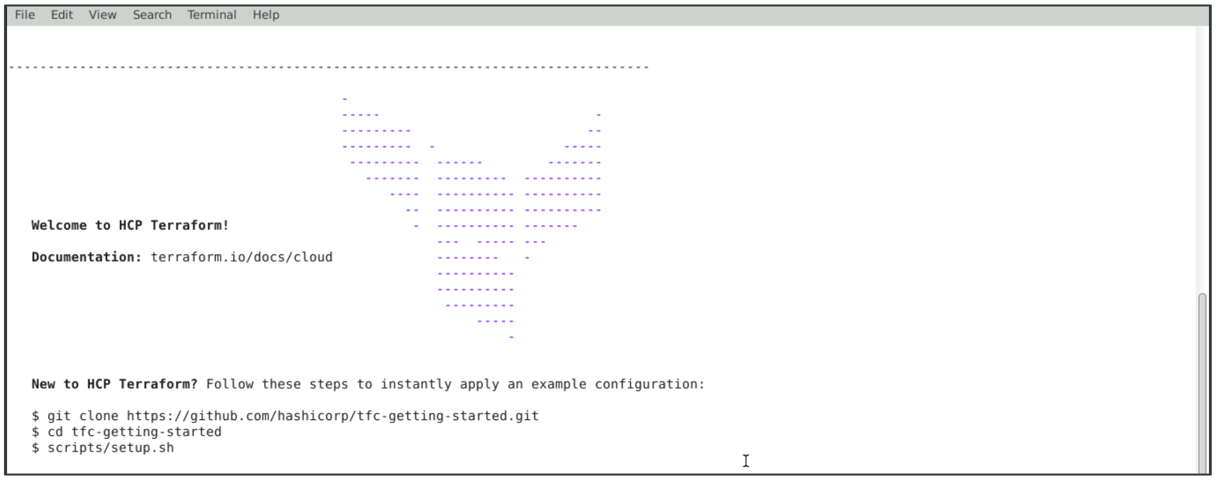
```
SFnFAITi4g8FvQ.atlasv1.NZ1vSlOfJ2bQitZnpcLif5GpwydMx79Xj2ZzItOTrXVeo5smSAsgQzJEipiiKR9c2BA
```

Created just now by user sachidanand21 Token has not been used

3.5 Go to the terminal and paste the copied token:



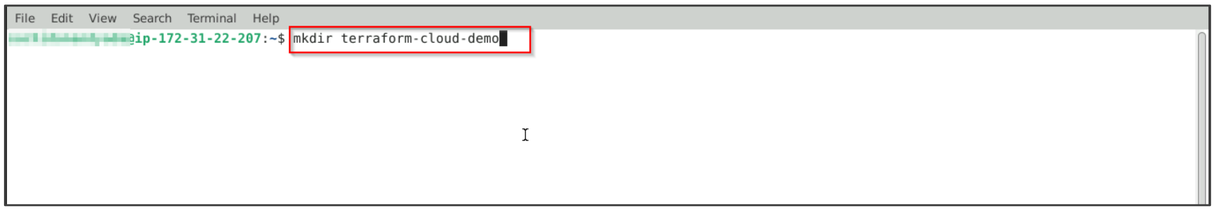
```
Do you want to proceed?
  Only 'yes' will be accepted to confirm.

  Enter a value: yes


--------------------------------------------------------------------------------

Terraform must now open a web browser to the tokens page for app.terraform.io.

If a browser does not open this automatically, open the following URL to proceed:
    https://app.terraform.io/app/settings/tokens?source=terraform-login


--------------------------------------------------------------------------------

Generate a token using your browser, and copy-paste it into this prompt.

Terraform will store the token in plain text in the following file
for use by subsequent commands:
    [redacted]terraform.d/credentials.tfrc.json

Token for app.terraform.io:
  Enter a value: Opening in existing browser session.
```



```
--------------------------------------------------------------------------------

                                          -
                                          -----                         -
                                          ---------                    --
                                          ---------  -               -----
                                           ---------  ------        -------
                                             -------  ---------  ----------
                                                ----  ---------- ----------
                                                  --  ---------- ----------
   Welcome to HCP Terraform!                       -  ---------- -------
                                                      ---  ----- ---
   Documentation: terraform.io/docs/cloud             --------   -
                                                      ----------
                                                      ----------
                                                       ---------
                                                           -----
                                                               -


   New to HCP Terraform? Follow these steps to instantly apply an example configuration:

   $ git clone https://github.com/hashicorp/tfc-getting-started.git
   $ cd tfc-getting-started
   $ scripts/setup.sh
```

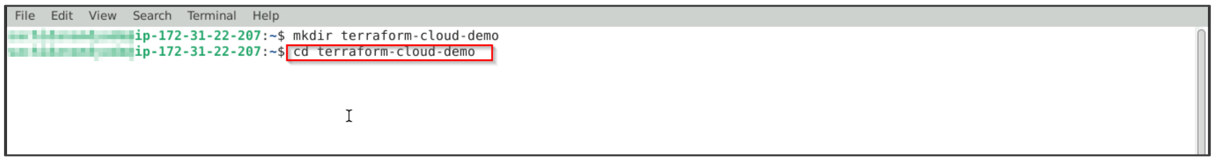
A welcome message from Terraform will appear as shown in the above screenshot.

3.6 Create a folder to proceed with Terraform initialization using the following command:
mkdir terraform-cloud-demo



```
[redacted]@ip-172-31-22-207:~$ mkdir terraform-cloud-demo
```

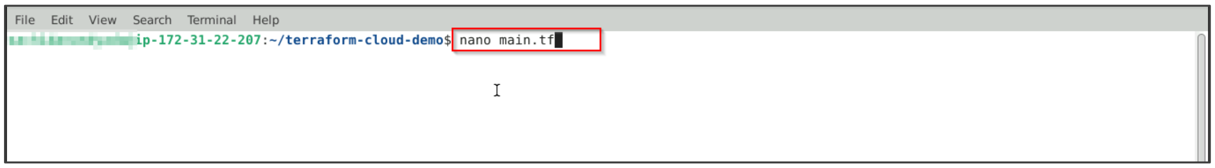
3.7 Navigate to the created folder by using the following command:
cd terraform-cloud-demo



```
[redacted]ip-172-31-22-207:~$ mkdir terraform-cloud-demo
[redacted]ip-172-31-22-207:~$ cd terraform-cloud-demo
```

3.8 Create a new file named **main.tf** in your project directory using the following command:

```
nano main.tf
```



```
[redacted]ip-172-31-22-207:~/terraform-cloud-demo$ nano main.tf
```

3.9 Add the initial Terraform configuration to the **main.tf** file:

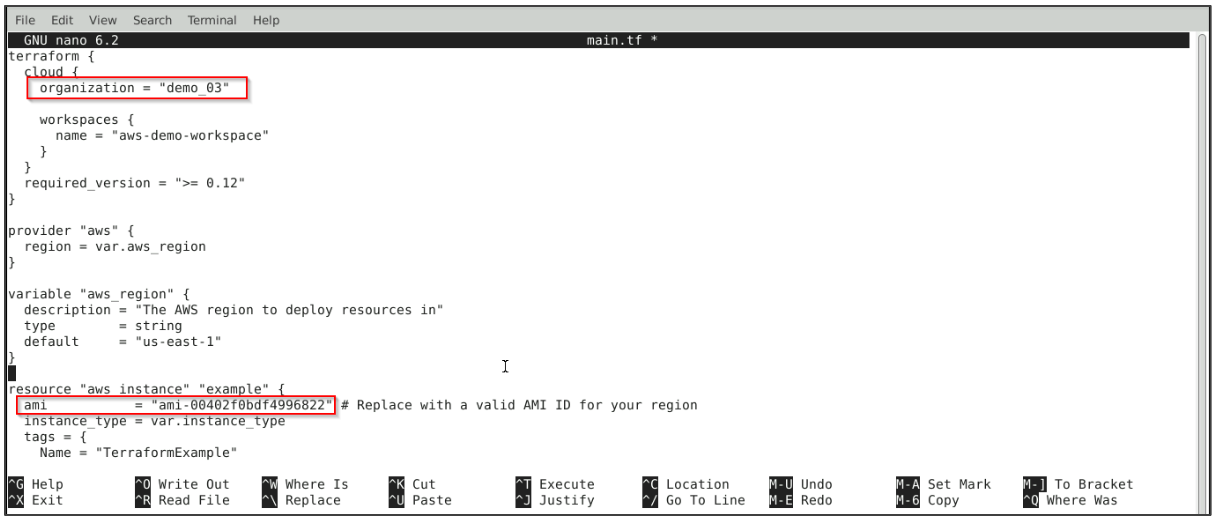
```
terraform {
  cloud {
    organization = "demo_03"

    workspaces {
      name = "aws-demo-workspace"
    }
  }
  required_version = ">= 0.12"
}

provider "aws" {
  region = var.aws_region
}

variable "aws_region" {
  description = "The AWS region to deploy resources in"
  type        = string
  default     = "us-east-1"
}

resource "aws_instance" "example" {
  ami           = "ami-00402f0bdf4996822" # Replace with a valid AMI ID for your region
  instance_type = var.instance_type
  tags = {
    Name = "TerraformExample"
  }
}
```

variable "instance_type" {

description = "The instance type for the EC2 instance"

type = string

}



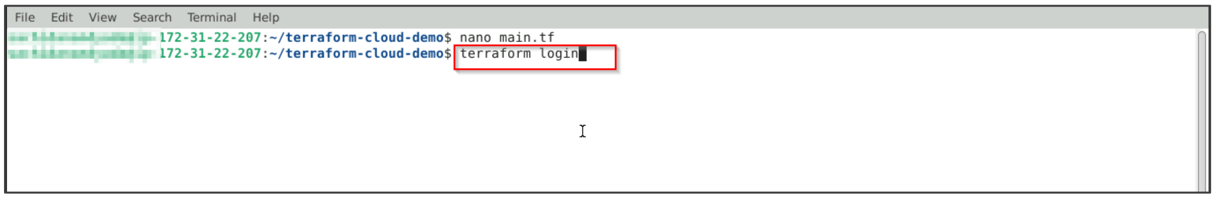
```
  GNU nano 6.2                                        main.tf *
terraform {
  cloud {
    organization = "demo_03"

    workspaces {
      name = "aws-demo-workspace"
    }
  }
  required_version = ">= 0.12"
}

provider "aws" {
  region = var.aws_region
}

variable "aws_region" {
  description = "The AWS region to deploy resources in"
  type        = string
  default     = "us-east-1"
}

resource "aws_instance" "example" {
  ami           = "ami-00402f0bdf4996822" # Replace with a valid AMI ID for your region
  instance_type = var.instance_type
  tags = {
    Name = "TerraformExample"

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark   M-] To Bracket
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy       ^Q Where Was
```

Note: Save the file by pressing Ctrl + X, then Y to confirm changes, and Enter to save and exit

Step 4: Define and use variables on Terraform Cloud

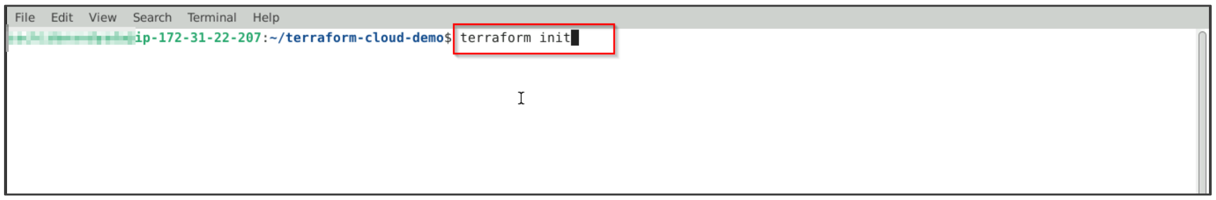
4.1 Log in to Terraform Cloud by using the following command:

terraform login

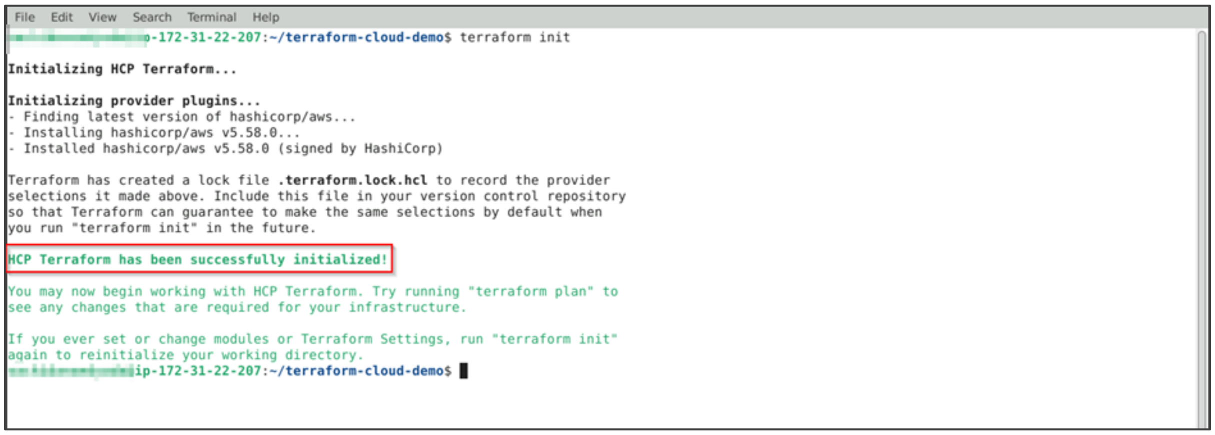


```
[redacted]172-31-22-207:~/terraform-cloud-demo$ nano main.tf
[redacted]172-31-22-207:~/terraform-cloud-demo$ terraform login
```

4.2 Initialize your Terraform configuration with the following command:
terraform init

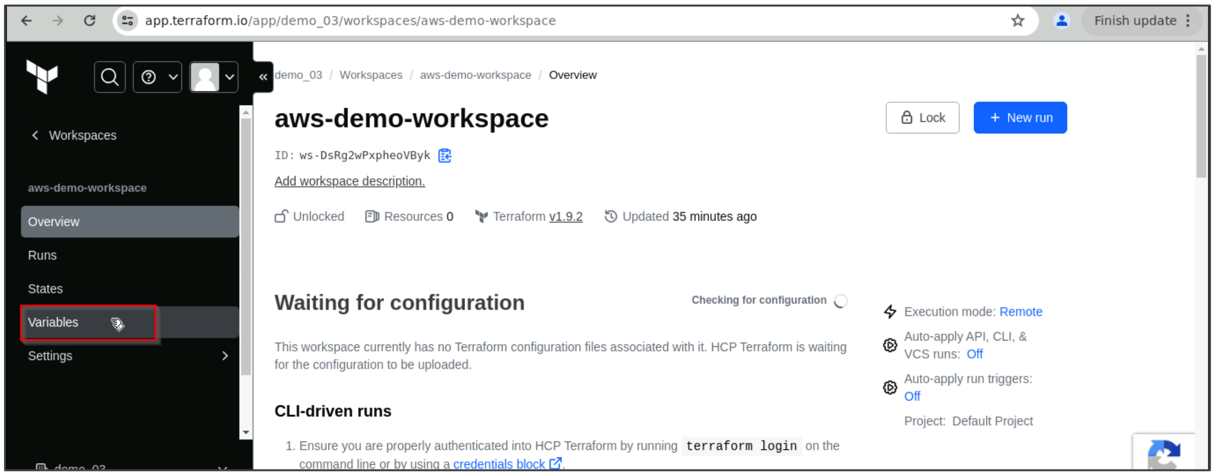


```
[redacted]ip-172-31-22-207:~/terraform-cloud-demo$ terraform init
```



```
[redacted]p-172-31-22-207:~/terraform-cloud-demo$ terraform init

Initializing HCP Terraform...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.58.0...
- Installed hashicorp/aws v5.58.0 (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.

HCP Terraform has been successfully initialized!

You may now begin working with HCP Terraform. Try running "terraform plan" to
see any changes that are required for your infrastructure.

If you ever set or change modules or Terraform Settings, run "terraform init"
again to reinitialize your working directory.
[redacted]ip-172-31-22-207:~/terraform-cloud-demo$
```

4.3 Go to your workspace on Terraform Cloud and navigate to the **Variables** tab:


app.terraform.io/app/demo_03/workspaces/aws-demo-workspace
Finish update
Workspaces
aws-demo-workspace
Overview
Runs
States
Variables
Settings
demo_03 / Workspaces / aws-demo-workspace / Overview
Lock
New run

aws-demo-workspace

ID: ws-DsRg2wPxpheoVByk

Add workspace description.

Unlocked   Resources 0   Terraform v1.9.2   Updated 35 minutes ago

Waiting for configuration

Checking for configuration

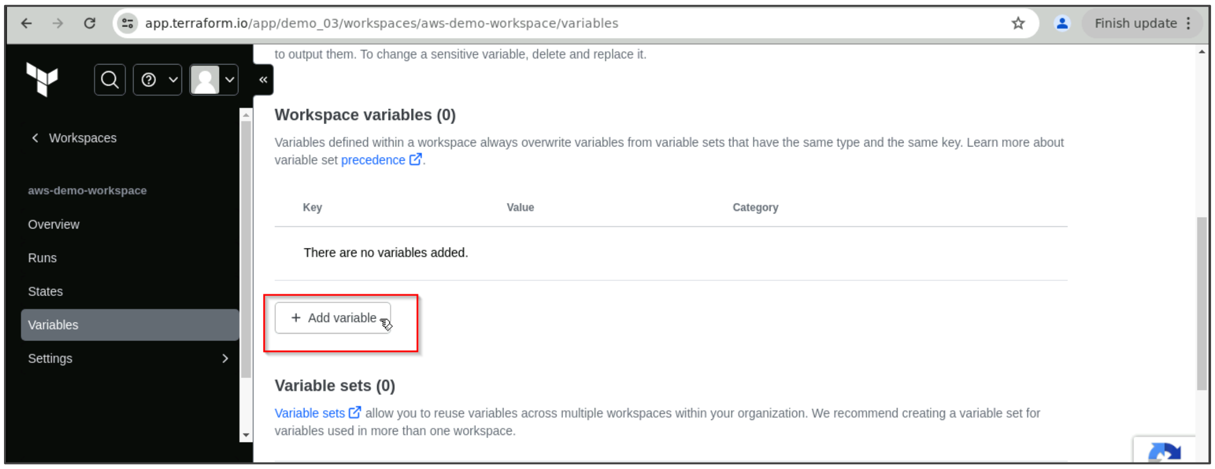
This workspace currently has no Terraform configuration files associated with it. HCP Terraform is waiting for the configuration to be uploaded.

Execution mode: Remote
Auto-apply API, CLI, & VCS runs: Off
Auto-apply run triggers: Off
Project: Default Project

CLI-driven runs

1. Ensure you are properly authenticated into HCP Terraform by running `terraform login` on the command line or by using a credentials block.

4.4 Click on **Add variable:**


app.terraform.io/app/demo_03/workspaces/aws-demo-workspace/variables

to output them. To change a sensitive variable, delete and replace it.

Workspace variables (0)

Variables defined within a workspace always overwrite variables from variable sets that have the same type and the same key. Learn more about variable set precedence.

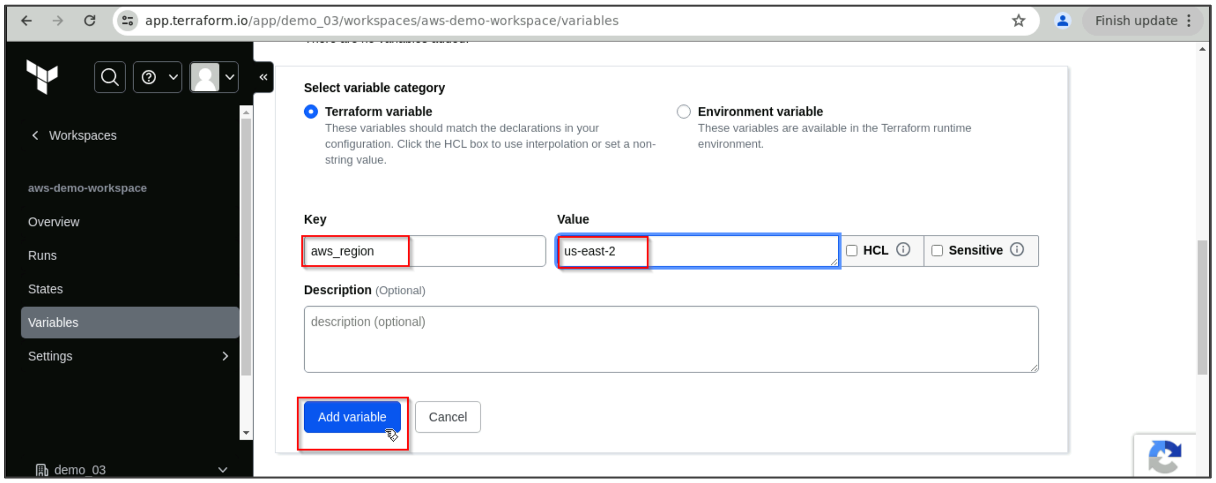
Key	Value	Category
There are no variables added.		

+ Add variable

Variable sets (0)

Variable sets allow you to reuse variables across multiple workspaces within your organization. We recommend creating a variable set for variables used in more than one workspace.

4.5 Enter the **Key** and **Value** as per the requirements:


app.terraform.io/app/demo_03/workspaces/aws-demo-workspace/variables

Select variable category

(●) **Terraform variable**
These variables should match the declarations in your configuration. Click the HCL box to use interpolation or set a non-string value.

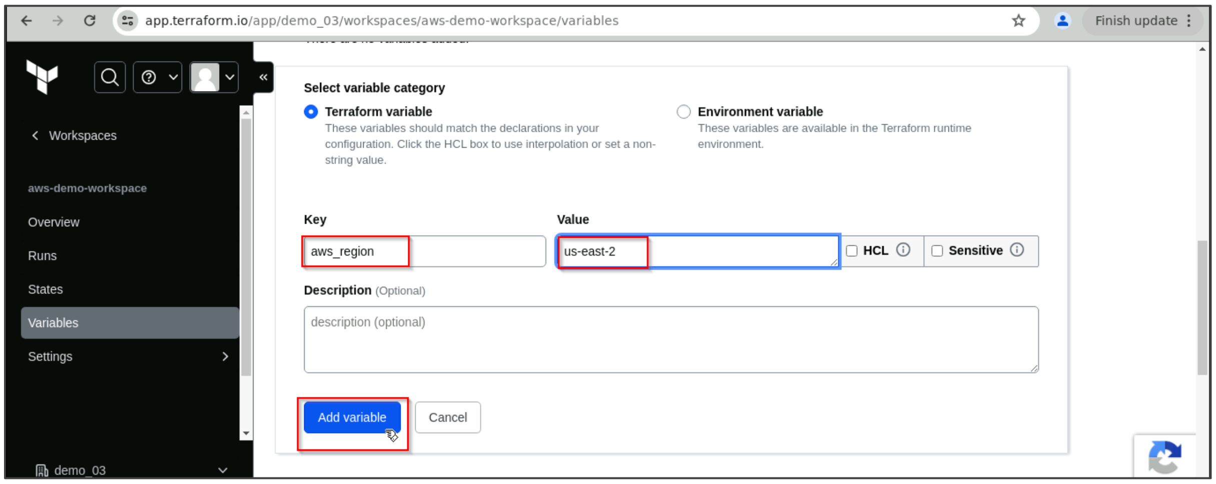
( ) **Environment variable**
These variables are available in the Terraform runtime environment.

Key: aws_region
Value: us-east-2
[ ] **HCL** [ ] **Sensitive**

Description (Optional)
description (optional)

Add variable | Cancel

4.6 Click on **Add variable:**



Select variable category

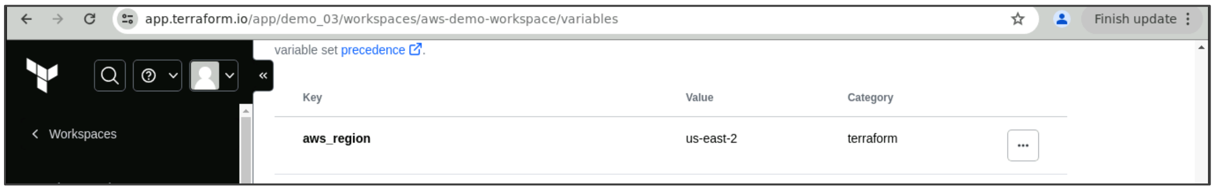
Terraform variable [✓]
These variables should match the declarations in your configuration. Click the HCL box to use interpolation or set a non-string value.

Environment variable
These variables are available in the Terraform runtime environment.

Key	Value	HCL	Sensitive
aws_region	us-east-2	☐	☐

Description (Optional)
description (optional)

Add variable Cancel

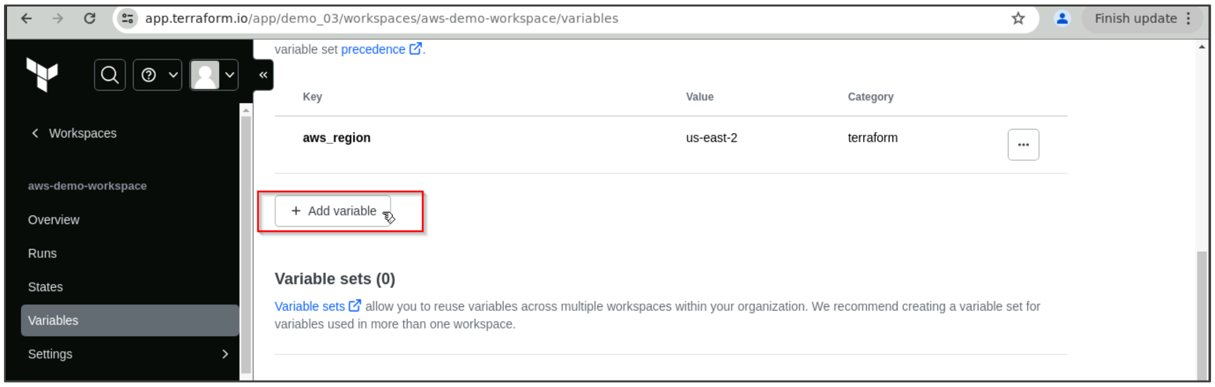

app.terraform.io/app/demo_03/workspaces/aws-demo-workspace/variables
Finish update
Workspaces

variable set precedence.

Key	Value	Category
aws_region	us-east-2	terraform

The required variable is created.

4.7 Click on **Add variable** to add one more variable to the configuration:



variable set precedence.

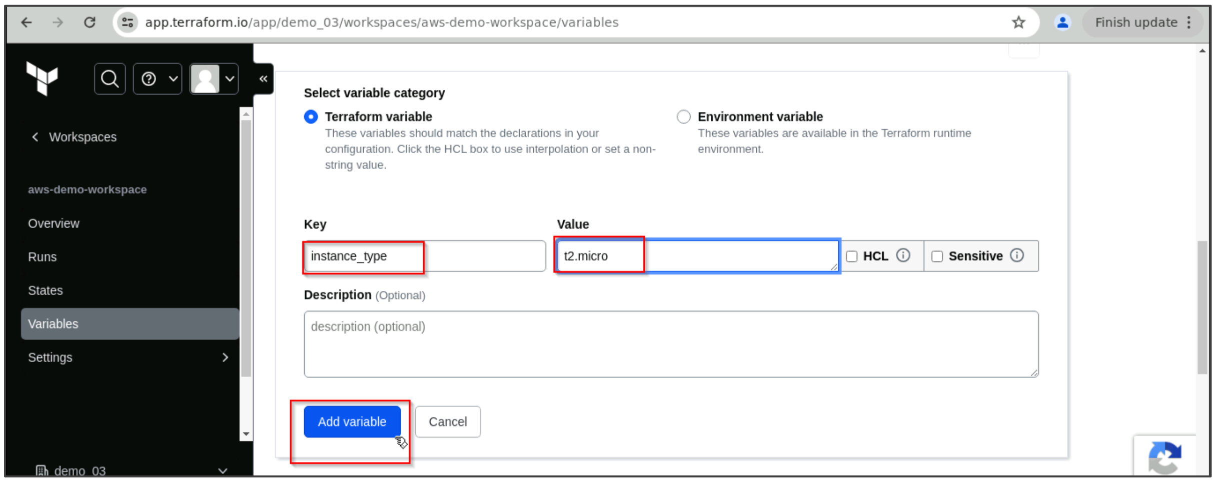
Key	Value	Category
aws_region	us-east-2	terraform

+ Add variable

Variable sets (0)

Variable sets allow you to reuse variables across multiple workspaces within your organization. We recommend creating a variable set for variables used in more than one workspace.

4.8 Enter the second **Key** and **Value** as per the requirements:



Select variable category

● Terraform variable
These variables should match the declarations in your configuration. Click the HCL box to use interpolation or set a non-string value.

○ Environment variable
These variables are available in the Terraform runtime environment.

Key	Value
instance_type	t2.micro

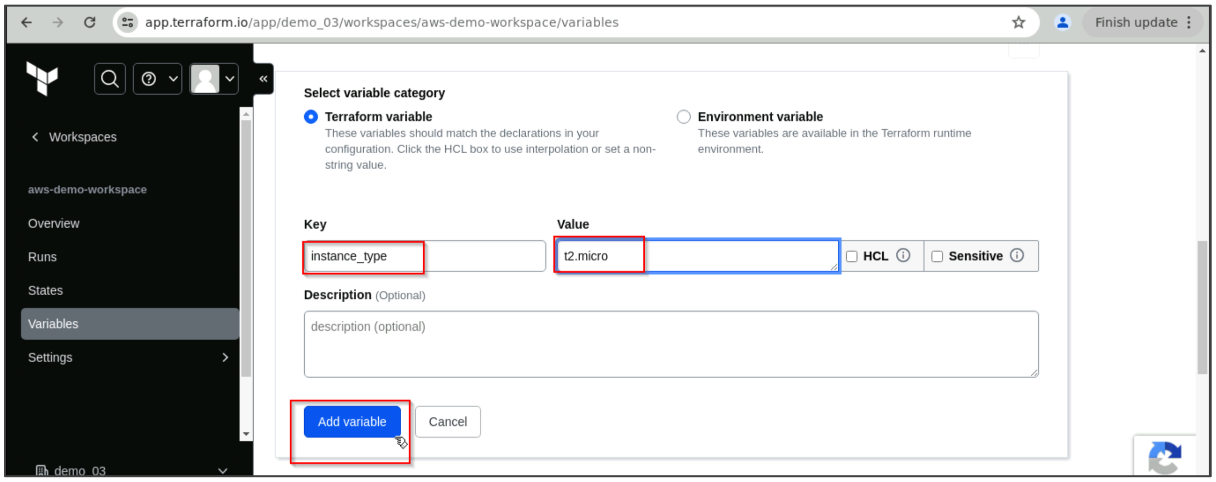
HCL Sensitive

Description (Optional)

description (optional)

Add variable Cancel

4.9 Click on **Add variable:**



Select variable category

● Terraform variable
These variables should match the declarations in your configuration. Click the HCL box to use interpolation or set a non-string value.

○ Environment variable
These variables are available in the Terraform runtime environment.

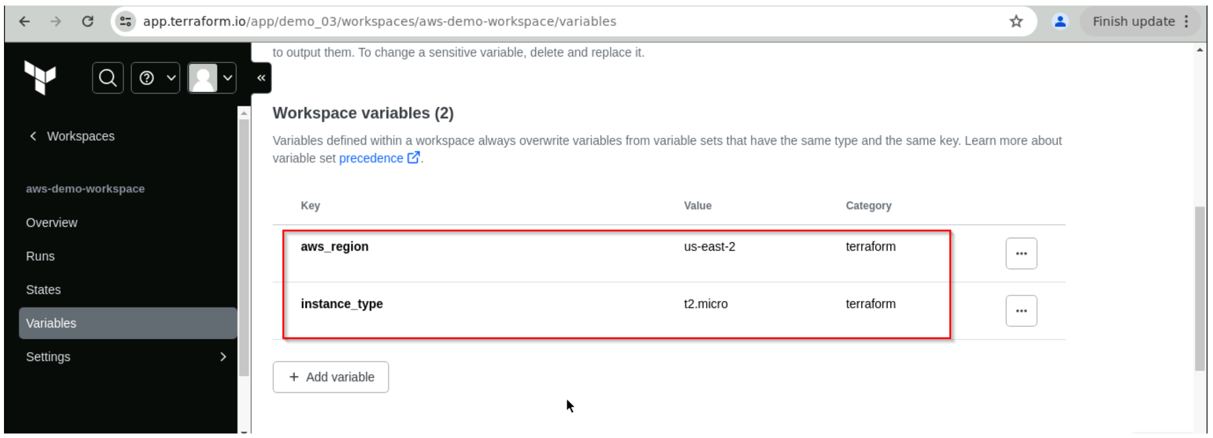
Key	Value
instance_type	t2.micro

HCL Sensitive

Description (Optional)

description (optional)

Add variable Cancel



to output them. To change a sensitive variable, delete and replace it.

Workspace variables (2)

Variables defined within a workspace always overwrite variables from variable sets that have the same type and the same key. Learn more about variable set precedence.

Key	Value	Category
aws_region	us-east-2	terraform
instance_type	t2.micro	terraform

+ Add variable

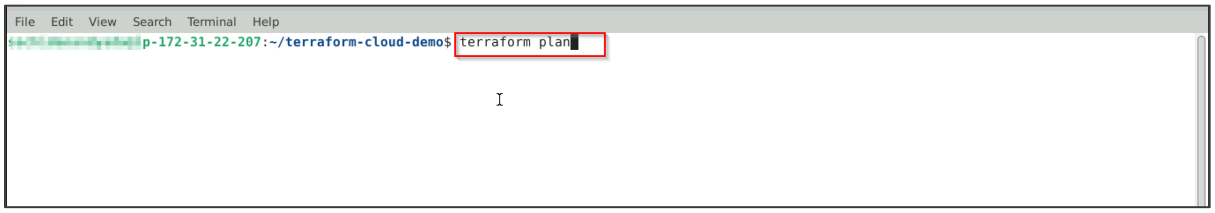
The required variables are created successfully.

By adding these key-value pairs in the **Variables** tab, you can dynamically manage the region and instance type for your Terraform configurations without hardcoding these values into your **main.tf** file.

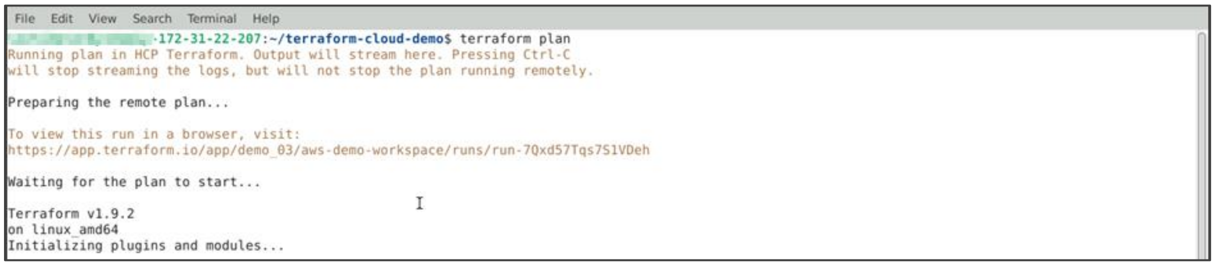
Step 5: Run the Terraform plan and apply

5.1 Navigate to the terminal and run the following command:

terraform plan



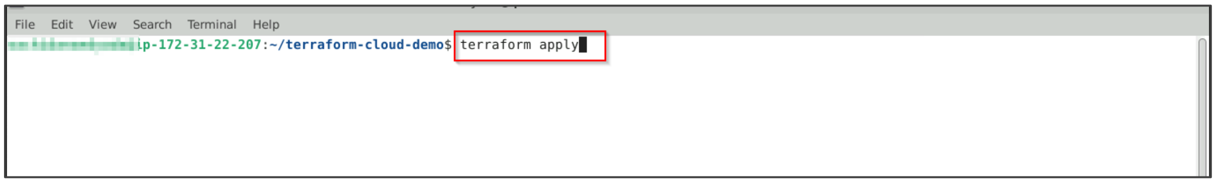
```
[redacted]p-172-31-22-207:~/terraform-cloud-demo$ terraform plan
```



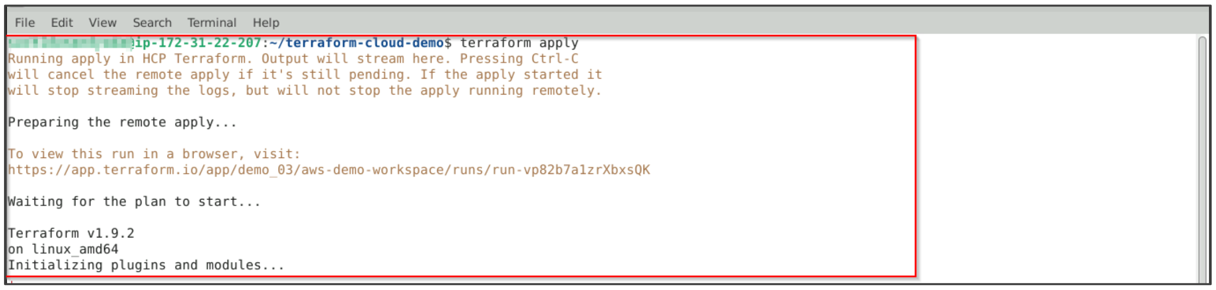
```
[redacted]-172-31-22-207:~/terraform-cloud-demo$ terraform plan
Running plan in HCP Terraform. Output will stream here. Pressing Ctrl-C
will stop streaming the logs, but will not stop the plan running remotely.

Preparing the remote plan...

To view this run in a browser, visit:
https://app.terraform.io/app/demo_03/aws-demo-workspace/runs/run-7Qxd57Tqs7S1VDeh

Waiting for the plan to start...

Terraform v1.9.2
on linux_amd64
Initializing plugins and modules...
```

5.2 Run the following command to apply the plan:

terraform apply

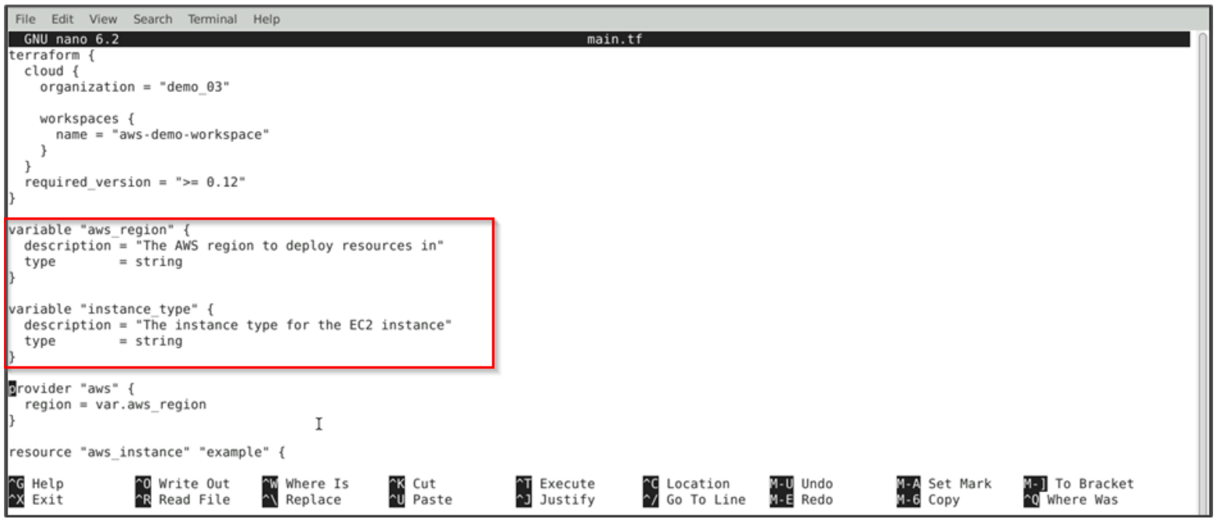


```
[redacted]ip-172-31-22-207:~/terraform-cloud-demo$ terraform apply
```



```
[redacted]ip-172-31-22-207:~/terraform-cloud-demo$ terraform apply
Running apply in HCP Terraform. Output will stream here. Pressing Ctrl-C
will cancel the remote apply if it's still pending. If the apply started it
will stop streaming the logs, but will not stop the apply running remotely.

Preparing the remote apply...

To view this run in a browser, visit:
https://app.terraform.io/app/demo_03/aws-demo-workspace/runs/run-vp82b7a1zrXbxsQK

Waiting for the plan to start...

Terraform v1.9.2
on linux_amd64
Initializing plugins and modules...
```

5.3 Verify the variables for Terraform configurations by using the following command:
nano main.tf


File Edit View Search Terminal Help
GNU nano 6.2
main.tf

```
terraform {
  cloud {
    organization = "demo_03"

    workspaces {
      name = "aws-demo-workspace"
    }
  }
  required_version = ">= 0.12"
}

variable "aws_region" {
  description = "The AWS region to deploy resources in"
  type        = string
}

variable "instance_type" {
  description = "The instance type for the EC2 instance"
  type        = string
}

provider "aws" {
  region = var.aws_region
}

resource "aws_instance" "example" {
```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark M-] To Bracket
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line M-E Redo M-6 Copy ^Q Where Was

By following the above steps, you can effectively work with variables and versions in Terraform Cloud, ensuring a collaborative and managed approach to infrastructure provisioning.