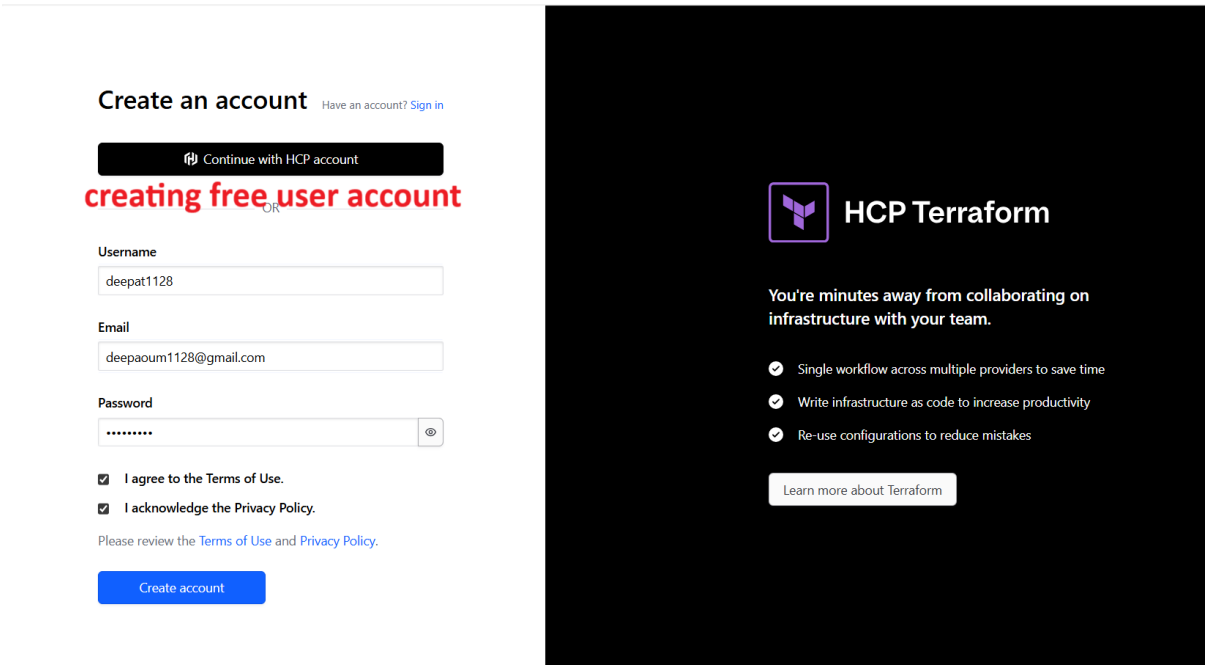
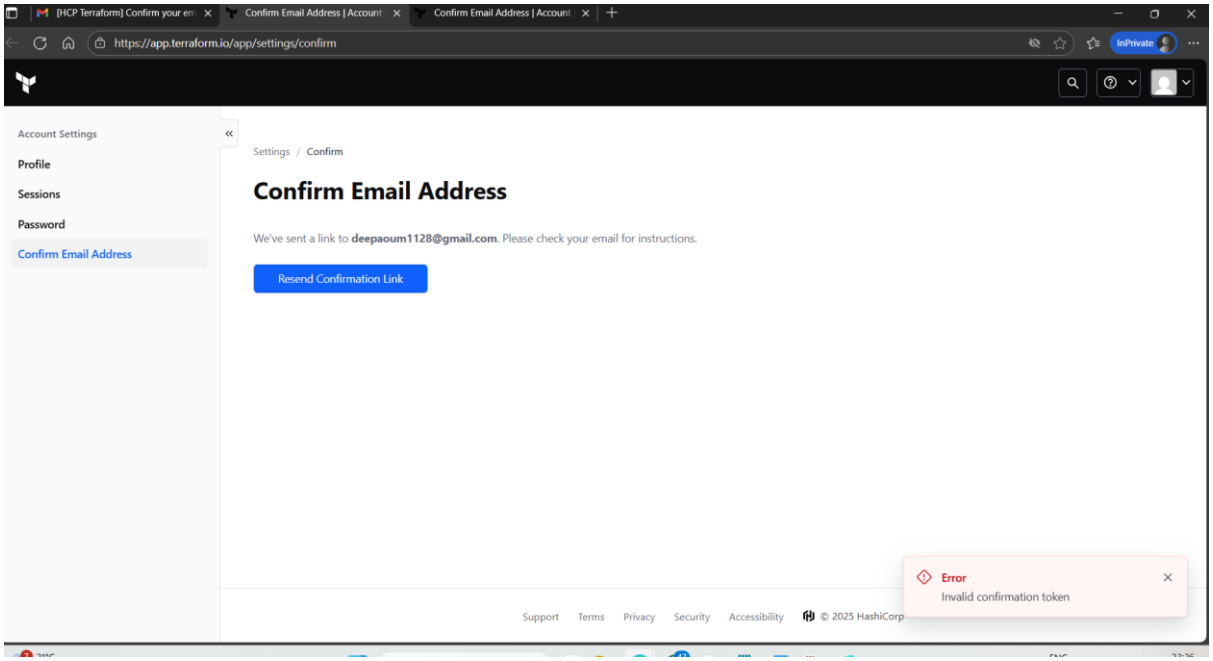
Name: DEEPA T
Date: 14.08.2025
Batch: 11

Terraform cloud:

Log in to app.terraform.io and create free user account.
Username: deepat1128
Email: deepaoum1128@gmail.com

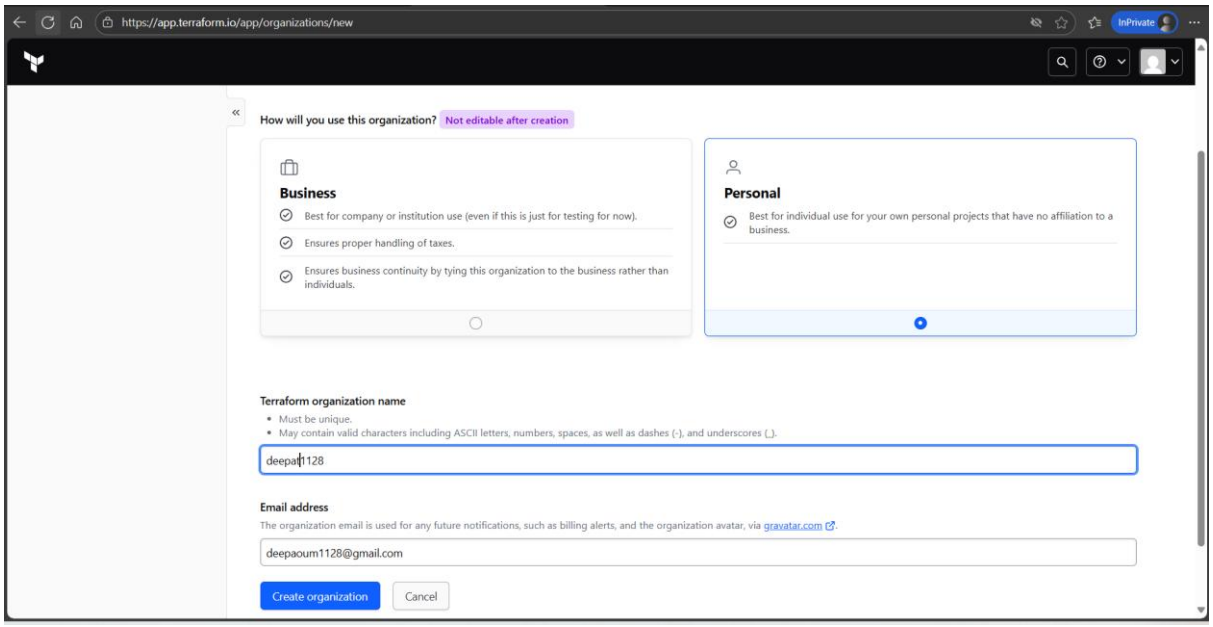


Email confirmation link is sent, link should be clicked in the gmail account.

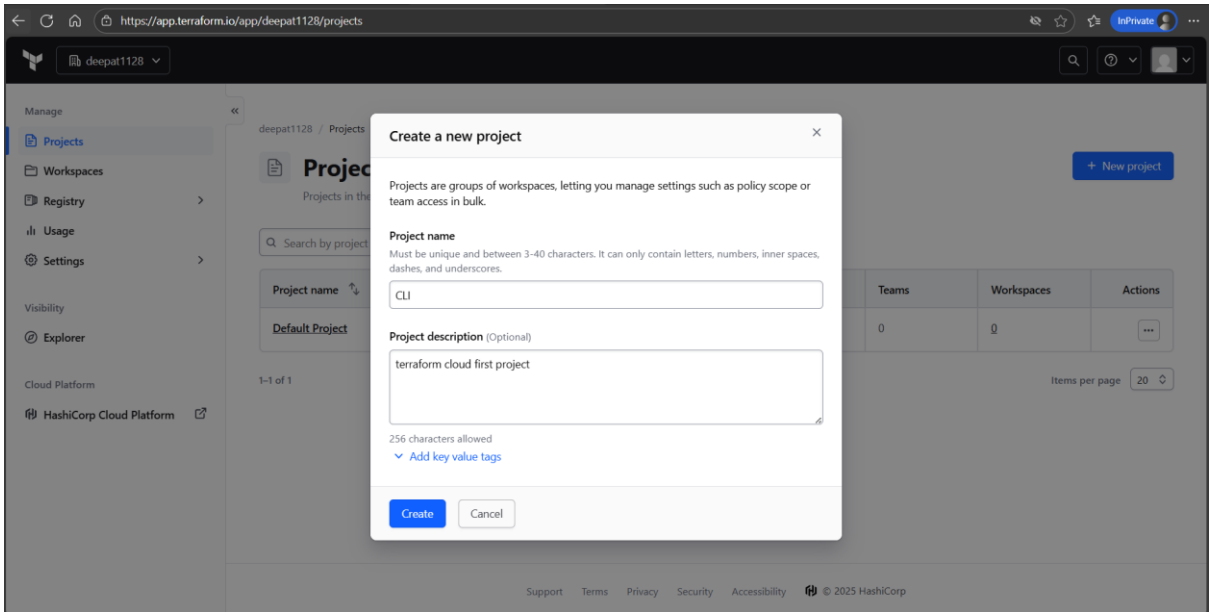


Now , successfully verifying gmail, New organization is created.

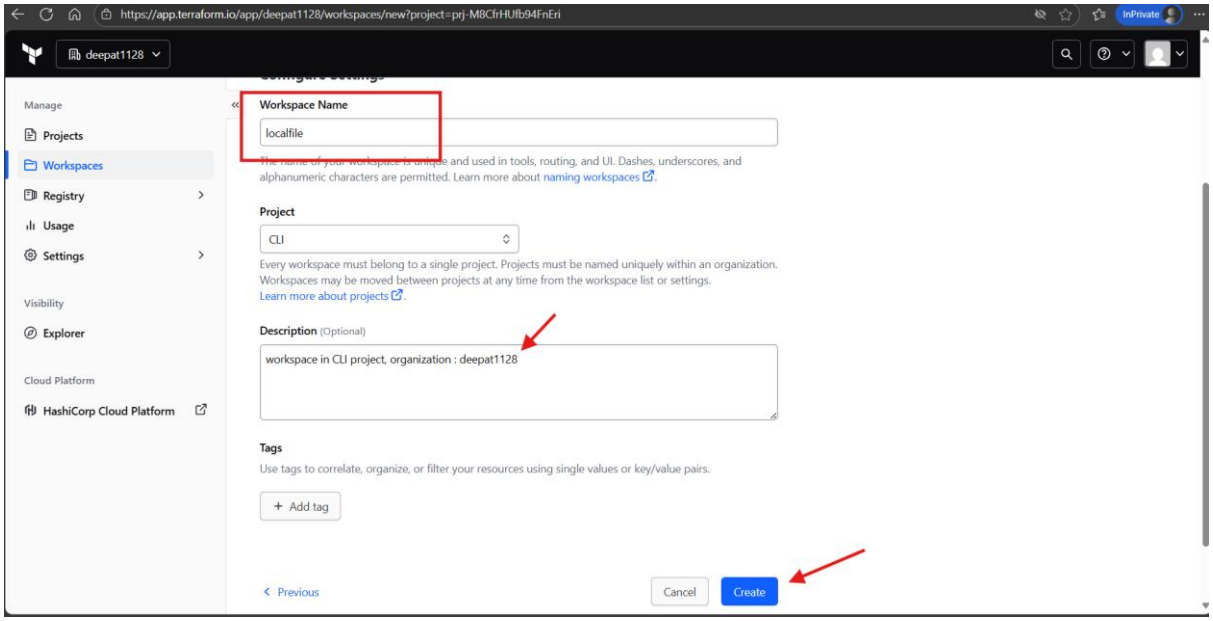
Name: deepat1128



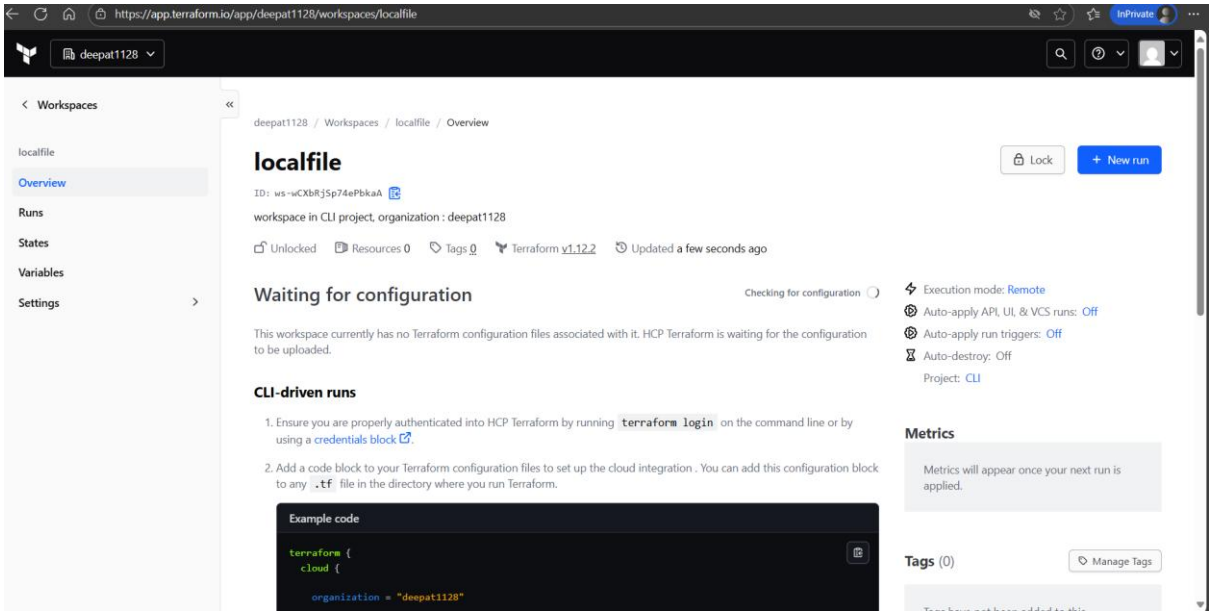
New project: CLI is created:

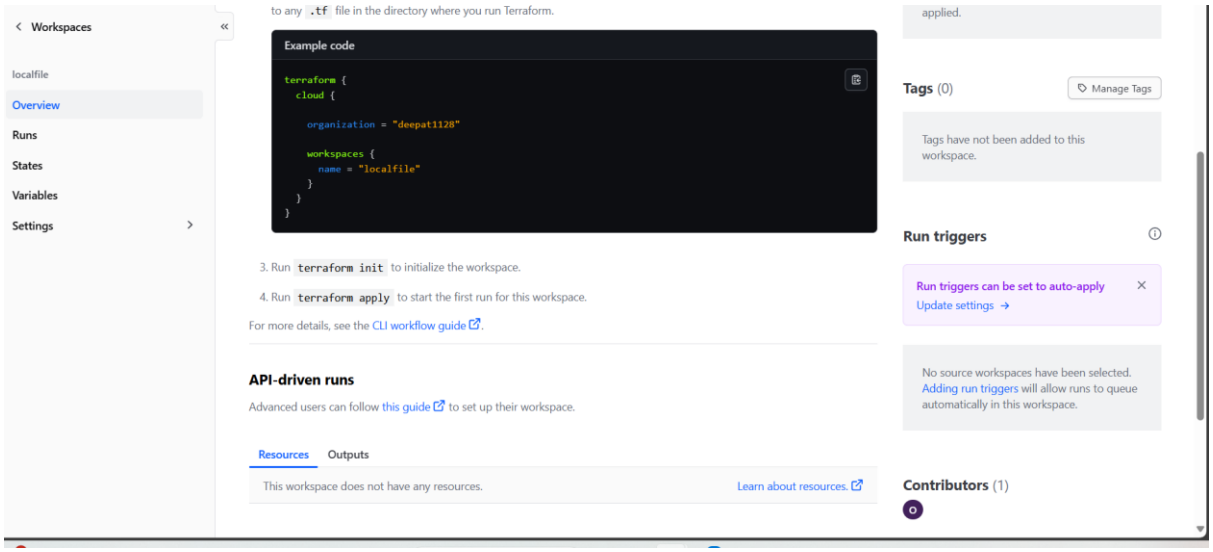


New workspace is created under the project:
Workspace name : localfile



Now to use the workspace, log in to terraform through cli in azure.





Log in to terraform in cli in azure:

Before that in a folder name: terra_cloud , create main.tf, variable.tf, and provider.tf files.

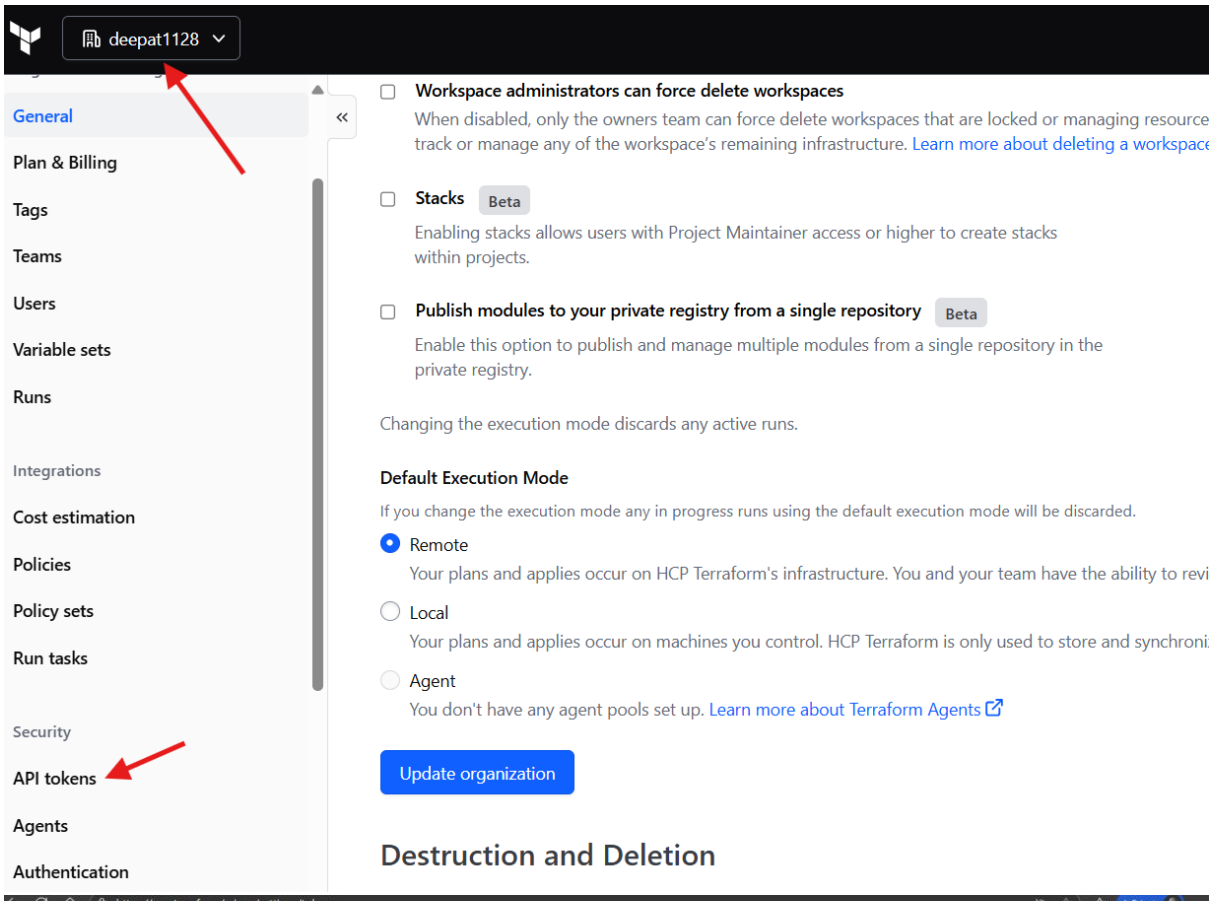
And give , Command: Terraform login

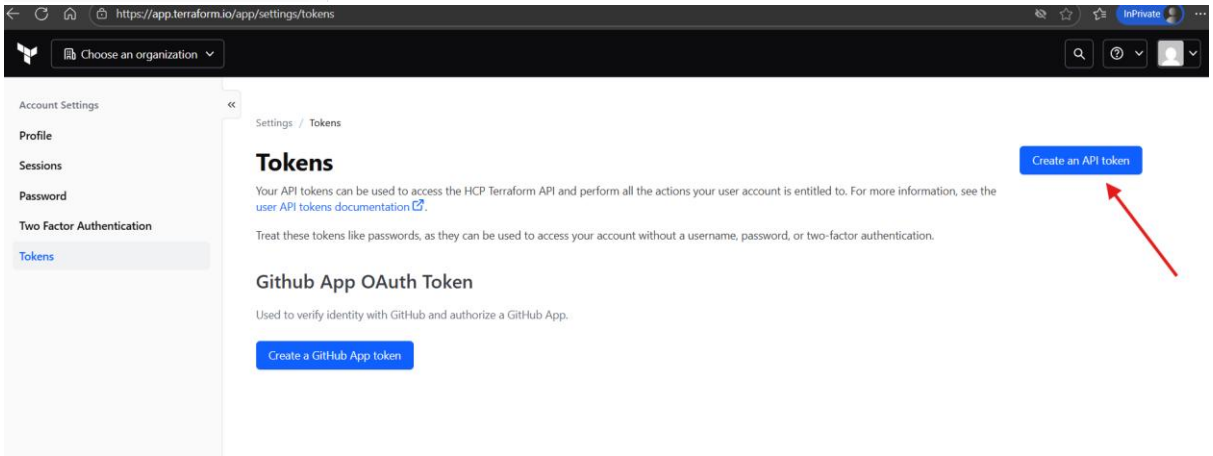
```
deepa [ ~ ]$ cd terra_cloud
deepa [ ~/terra_cloud ]$ mv /home/deepa/main.tf .
deepa [ ~/terra_cloud ]$ ls
main.tf
deepa [ ~/terra_cloud ]$ mv /home/deepa/variable.tf .
deepa [ ~/terra_cloud ]$ mv /home/deepa/provider.tf .
deepa [ ~/terra_cloud ]$ ls
main.tf  provider.tf  variable.tf
deepa [ ~/terra_cloud ]$ terraform login
```

terraform login

After giving yes, it prompts for token for app.terraform.io

So now in app.terraform.io, → settings → API token → user token →Account settings page → create API token →CLI token → generate token → copy the token secure it.





```
Enter a value:

Retrieved token for user deepat1128

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

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

deepa [ ~/terra_cloud ]$
```

Add the token in azure cli

So logged into terraform cloud with token:

Add the example code to cloud.tf file in terraform cloud directory ‘cloud’

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.
2. Add a code block to your Terraform configuration files to set up the cloud integration . You can add this configuration block to any `.tf` file in the directory where you run Terraform.

Example code

```
terraform {
  cloud {

    organization = "deepat1128"

    workspaces {
      name = "localfile"
    }
  }
}
```

3. Run `terraform init` to initialize the workspace.
4. Run `terraform apply` to start the first run for this workspace.

For more details, see the CLI workflow guide.

Project: CLI

Metrics

Metrics will appear once your next run is applied.

Tags (0) Manage Tags

Tags have not been added to this workspace.

Run triggers

Run triggers can be set to auto-apply

Update settings →

Terraform init:

Give service principle details(Tenant ID, Subscription ID, Client secret value, Client ID) now in provider.tf then continue.

```
deepa [ ~/terra_cloud ]$ vi cloud.tf
deepa [ ~/terra_cloud ]$ terraform init
Initializing HCP Terraform...
Initializing provider plugins...
- Finding hashicorp/azurerm versions matching "4.37.0"...
- Installing hashicorp/azurerm v4.37.0...
- Installed hashicorp/azurerm v4.37.0 (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.
```

Terraform plan:

```
deepa [ ~/terra_cloud ]$ 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/deepat1128/localfile/runs/run-7h53qJQfp1nEq3Jv

Waiting for the plan to start...

Terraform v1.12.2
on linux_amd64
Initializing plugins and modules...

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:

  # azurerm_resource_group.example will be created
  + resource "azurerm_resource_group" "example" {
      + id       = (known after apply)
      + location = "westus2"
      + name     = "rg-deepa"
    }

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

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
deepa [ ~/terra_cloud ]$
```

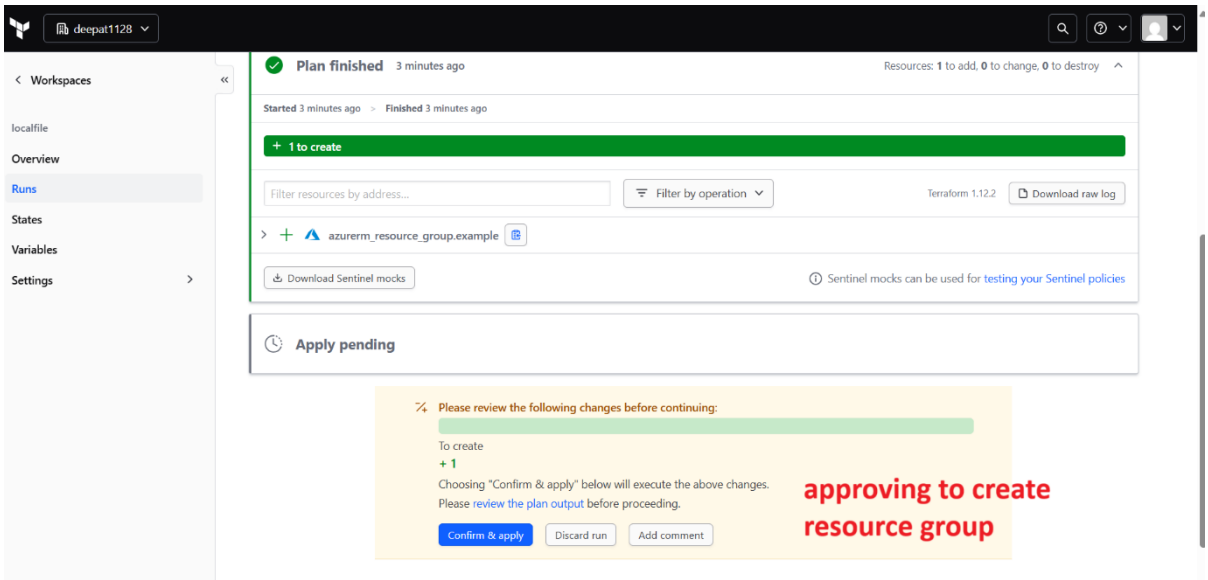
Terraform apply: It runs in the remote

```
Preparing the remote apply...

To view this run in a browser, visit:
https://app.terraform.io/app/deepat1128/localfile/runs/run-Ns963xVhwys4RQs4

Waiting for the plan to start...

Terraform v1.12.2
on linux_amd64
Initializing plugins and modules...

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:

  # azurerm_resource_group.example will be created
  + resource "azurerm_resource_group" "example" {
      + id       = (known after apply)
      + location = "westus2"
      + name     = "rg-deepa"
    }

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

Do you want to perform these actions in workspace "localfile"?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value:
```

approval can be given either in cli or terraform cloud

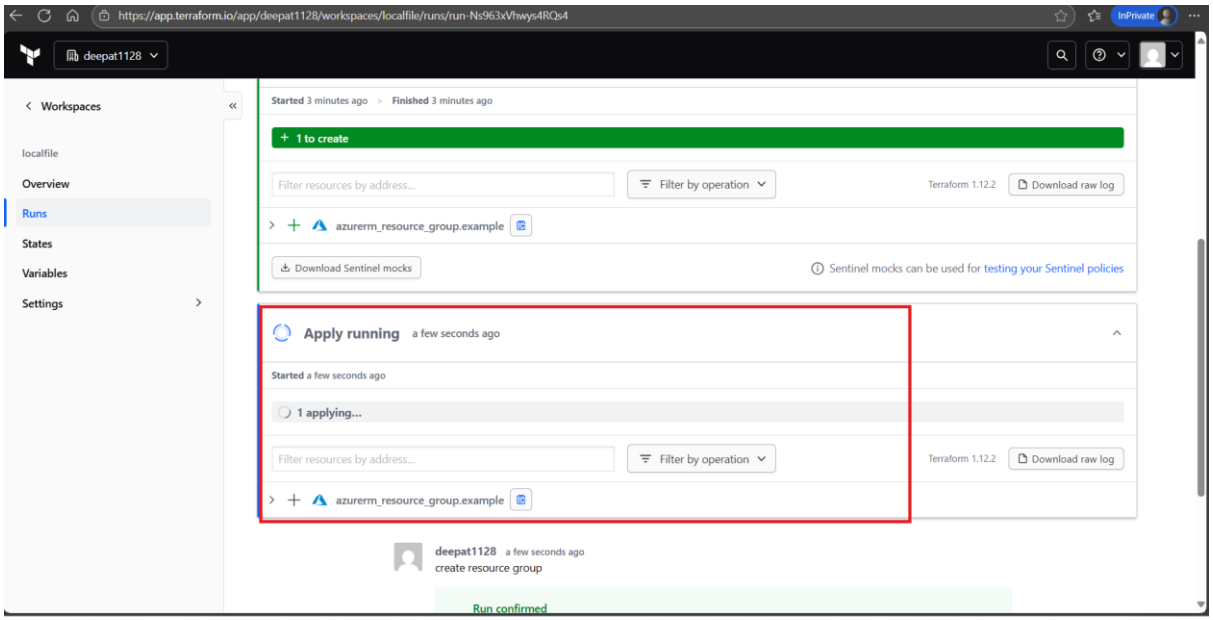
In workspace localfile → view all runs → current run →

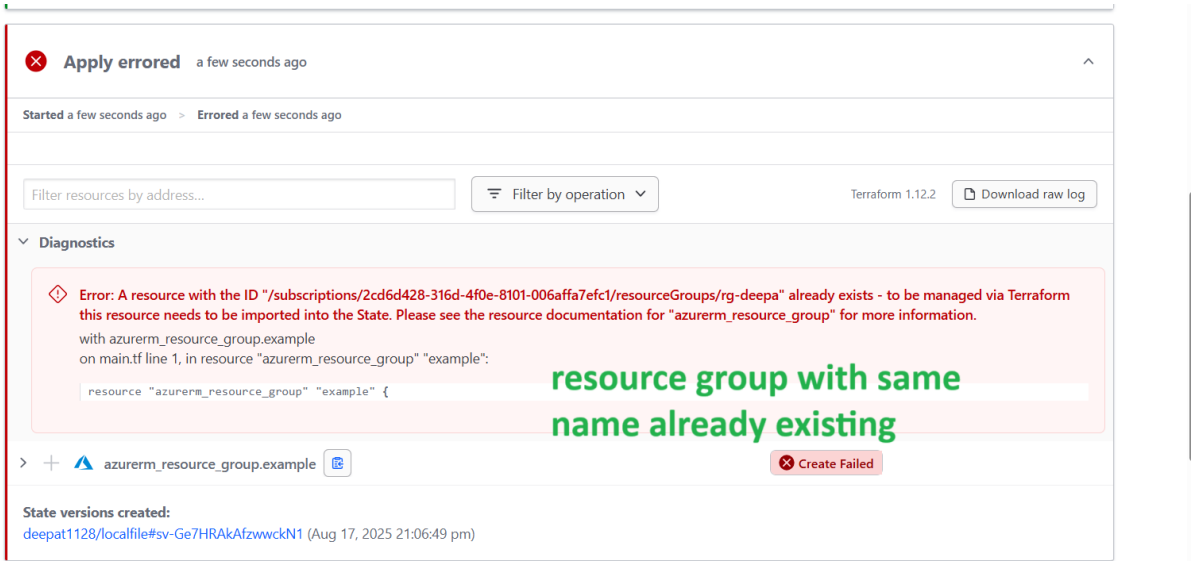
For plan, either we can approve in terraform azure or cloud



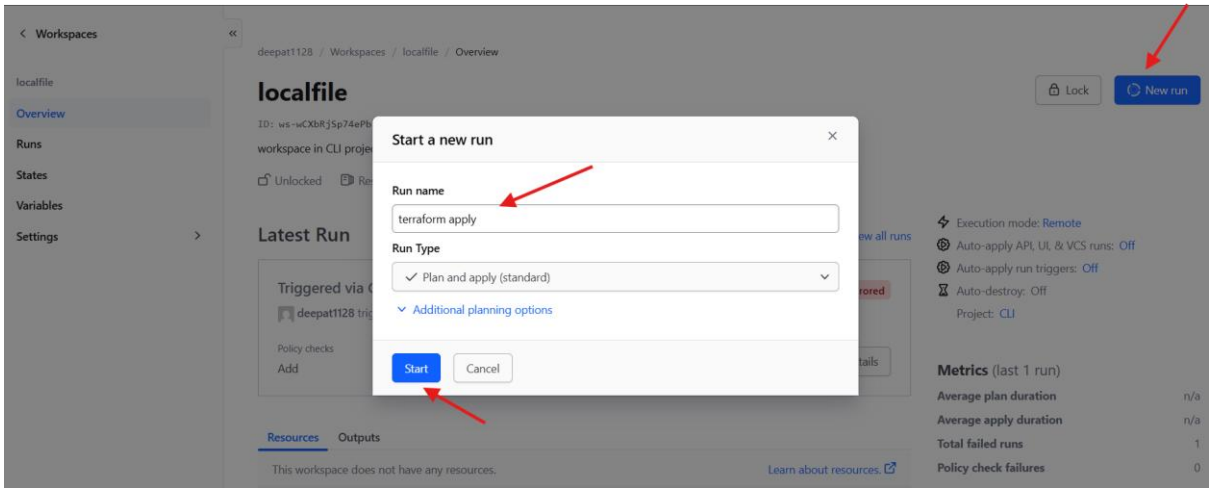
Then it starts applying

Creates the resource as per configurations.

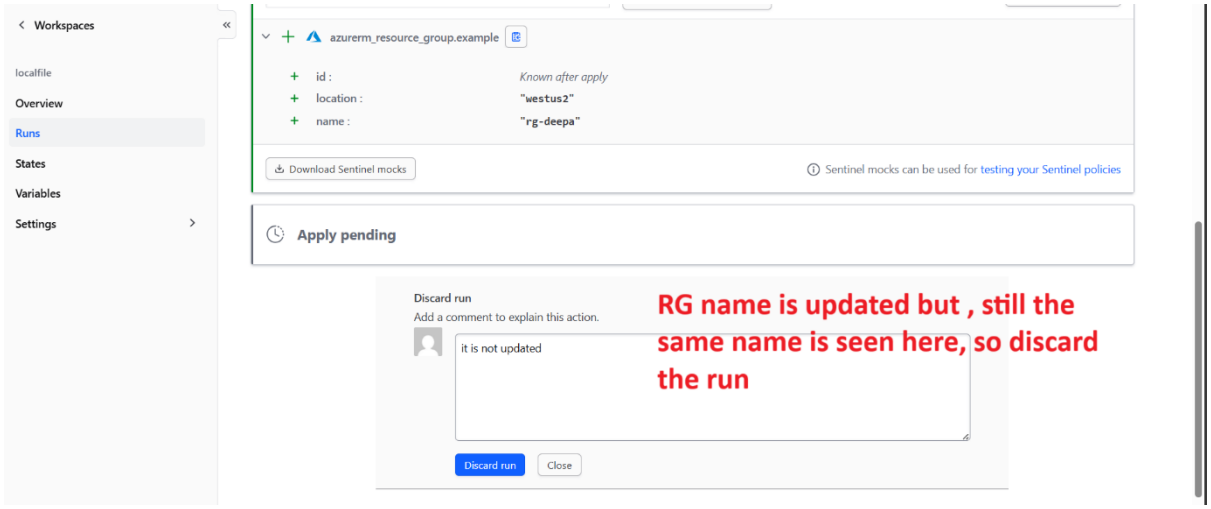




Rename the resource group in variable.tf
Rg-deepa2



Again approval is given in terra cloud only



```
Preparing the remote plan...

To view this run in a browser, visit:
https://app.terraform.io/app/deepat1128/localfile/runs/run-XEg4izMEFg1fzCBd

Waiting for the plan to start...

Terraform v1.12.2
on linux_amd64
Initializing plugins and modules...

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:

  # azurerm_resource_group.example will be created
  + resource "azurerm_resource_group" "example" {
      + id       = (known after apply)
      + location = "westus2"
      + name     = "rg-deepa2"
    }

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


Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
deepa [ ~/terra_cloud ]$
```

plan is run in azure cli, now it is updated with new RG name

Home >

Resource groups

Identify inactive resource groups using ARG. | Filter resource groups by status using ARG. | What are the different statuses available for resource groups?

Default Directory (punyateja5gmail.onmicrosoft.com)

Create | Manage view | Refresh | Export to CSV | Open query | Assign tags

You are viewing a new version of Browse experience. Click here to access the old experience.

Filter for any field... | Subscription equals **all** | Location equals **all** | Add filter

Name ↑	Subscription	Location
ansible-vid	Azure subscription 1	West Europe
NetworkWatcherRG	Azure subscription 1	Japan East
rg-ansible	Azure subscription 1	East US
rg-deepa	Azure subscription 1	West US 2
rg-deepa2	Azure subscription 1	West US 2
rg-jenkins	Azure subscription 1	West US 3
rg-vani	Azure subscription 1	Central US
tej_task1	Azure subscription 1	Japan East

rg-deepa2 created

```
deepa [ ~/terra_cloud ]$ terraform state list
azurerm_resource_group.example
deepa [ ~/terra_cloud ]$ ls
cloud.tf  main.tf  provider.tf  variable.tf
deepa [ ~/terra_cloud ]$ terraform state show azurerm_resource_group.example
# azurerm_resource_group.example:
resource "azurerm_resource_group" "example" {
    id         = "/subscriptions/2cd6d428-316d-4f0e-8101-006affa7efc1/resourceGroups/rg-deepa2"
    location   = "westus2"
    managed_by = null
    name       = "rg-deepa2"
}
deepa [ ~/terra_cloud ]$
```

State file is managed by terraform cloud. It is not seen in the azure cli.

View all runs: