SPCM LAB-1

Objective: Installing terrafrom

- First, install the HashiCorp tap, a repository of all our Homebrew packages.

```
Running `brew update --auto-update`...
==> Auto-updated Homebrew!
Updated 4 taps (hashicorp/tap, homebrew/services, homebrew/core and homebrew/cask).
==> New Formulae
git-grab                  libnsgif                  nowplaying-cli            scnlib                    tfautomv
hashicorp/tap/tfstacks    libspelling               openjph                   senpai                    tomlplusplus
icloudpd                  limesuite                 rsyncy                    terrapin-scanner
==> New Casks
bitbox                    cleanupbuddy              insomnium                 nightshade                ttu-base-suite
bugdom2                   domzilla-caffeine         lyricsfinder              theiaide

You have 25 outdated formulae and 1 outdated cask installed.

gauravbhandari@gauravs-MacBook-Air-2 ~ %
```

- Now, install Terraform with hashicorp/tap/terraform.

```
==> Fetching hashicorp/tap/terraform
==> Downloading https://releases.hashicorp.com/terraform/1.7.1/terraform_1.7.1_darwin_arm64.zip
Already downloaded: /Users/gauravbhandari/Library/Caches/Homebrew/downloads/a9a9a6dfb024c2ab845b9e9eb25cabb393e0a44927b85
e4377c470fdbd46bd39--terraform_1.7.1_darwin_arm64.zip
==> Installing terraform from hashicorp/tap
🍺  /opt/homebrew/Cellar/terraform/1.7.1: 3 files, 88.6MB, built in 4 seconds
==> Running `brew cleanup terraform`...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see `man brew`).
Removing: /Users/gauravbhandari/Library/Caches/Homebrew/terraform--1.6.6.zip... (23.7MB)
gauravbhandari@gauravs-MacBook-Air-2 ~ %
```

- To update to the latest version of Terraform, first update Homebrew.

```
gauravbhandari@gauravs-MacBook-Air-2 ~ % brew update
Already up-to-date.
```

- Then, run the upgrade command to download and use the latest Terraform version.

```
gauravbhandari@gauravs-MacBook-Air-2 ~ % brew upgrade hashicorp/tap/terraform
Warning: hashicorp/tap/terraform 1.7.1 already installed
gauravbhandari@gauravs-MacBook-Air-2 ~ %
```

- Check the installation of terraform

Objective:Terraform AWS Provider and IAM User Setting

- Create a new directory for your Terraform configuration:
- Create a file named main.tf with the following content:

```
Terraform v1.6.6
on darwin_arm64
```

- Run the following command to initialize your Terraform working directory.

SPCM LAB-2

```
Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.31.0"...
- Installing hashicorp/aws v5.31.0...
- Installed hashicorp/aws v5.31.0 (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.

Terraform has been successfully initialized!
```

Objective: Provisioning an EC2 Instance on AWS

- Create Terraform Configuration File for EC2 instance(instance.tf):

instance.tf

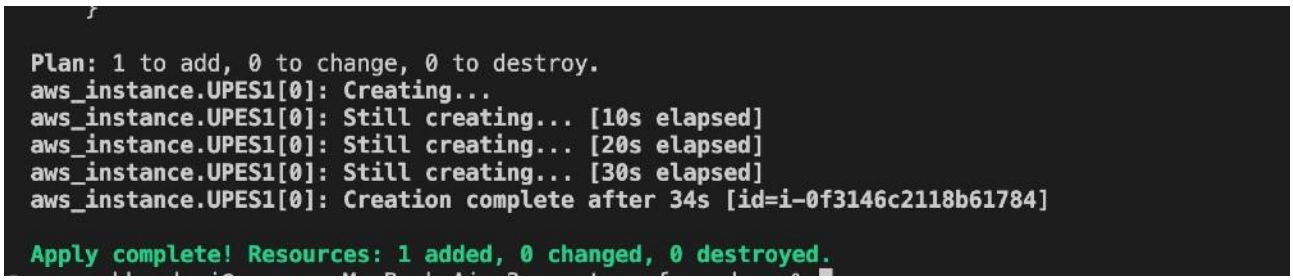
```
resource "aws_instance" "UPES1" {
  ami           = "ami-03f4878755434977f"
  instance_type = "t2.micro"
  count         = 1
  tags = {
    Name = "My-ec2"
  }
}
```

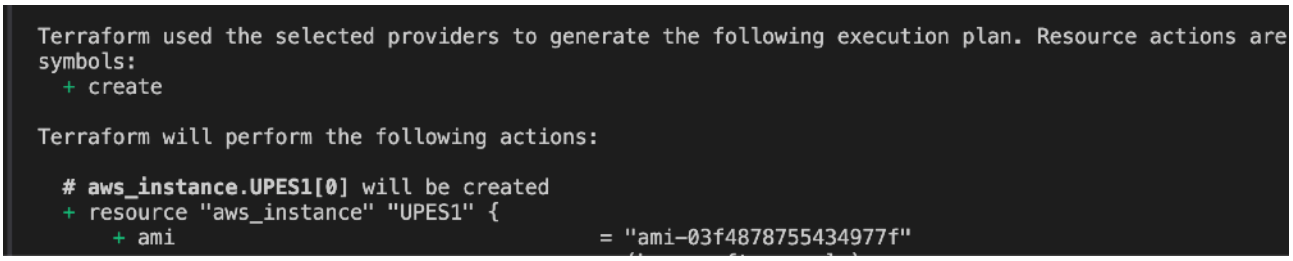
- Run the command “terraform plan” and review the plan to ensure it aligns with your expectations.

```
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:

  # aws_instance.UPES1[0] will be created
  + resource "aws_instance" "UPES1" {
      + ami                                  = "ami-03f4878755434977f"
      + arn                                  = (known after apply)
      + user_data_replace_on_change          = false
      + vpc_security_group_ids               = (known after apply)
    }

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.
```

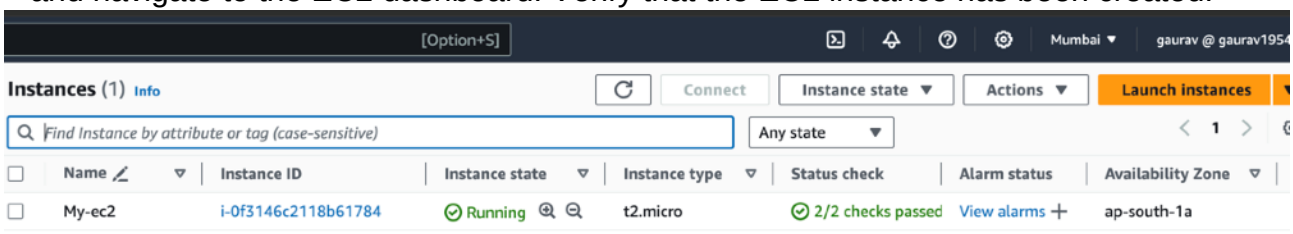
- Apply the changes to create the AWS resources terraform apply.Type yes when prompted.

SPCM LAB-3


Plan: 1 to add, 0 to change, 0 to destroy.
aws_instance.UPES1[0]: Creating...
aws_instance.UPES1[0]: Still creating... [10s elapsed]
aws_instance.UPES1[0]: Still creating... [20s elapsed]
aws_instance.UPES1[0]: Still creating... [30s elapsed]
aws_instance.UPES1[0]: Creation complete after 34s [id=i-0f3146c2118b61784]
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.


Terraform used the selected providers to generate the following execution plan. Resource actions are
symbols:
+ create
Terraform will perform the following actions:
# aws_instance.UPES1[0] will be created
+ resource "aws_instance" "UPES1" {
+ ami = "ami-03f4878755434977f"

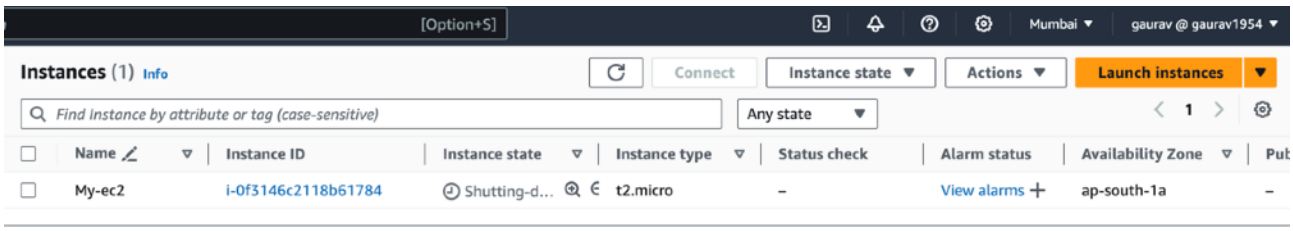
- After the terraform apply command completes, log in to your AWS Management Console and navigate to the EC2 dashboard. Verify that the EC2 instance has been created.



- When you are done experimenting, run the following command to destroy the created resources.

```
aws_instance.UPES1[0]: Refreshing state... [id=i-0f3146c2118b61784]

Terraform used the selected providers to generate the following execution plan. Resource actions are
symbols:
  - destroy

Terraform will perform the following actions:
```

```
Plan: 0 to add, 0 to change, 1 to destroy.
aws_instance.UPES1[0]: Destroying... [id=i-0f3146c2118b61784]
aws_instance.UPES1[0]: Still destroying... [id=i-0f3146c2118b61784, 10s elapsed]
aws_instance.UPES1[0]: Still destroying... [id=i-0f3146c2118b61784, 20s elapsed]
aws_instance.UPES1[0]: Still destroying... [id=i-0f3146c2118b61784, 30s elapsed]
aws_instance.UPES1[0]: Destruction complete after 31s

Destroy complete! Resources: 1 destroyed.
```



SPCM LAB-4

Objective: Learn how to define and use variables in Terraform configuration.

- Create a file variable.tf with the following contents.

instance.tf variable.tf

variable.tf > variable "countNumber"

```
variable "ubuntu_ami" {
  type    = string
  default = "ami-03f4878755434977f"
}
variable "instance_type" {
  type    = string
  default = "t2.micro"
}
variable "countNumber" {
  type    = number
  default = 1
}
```

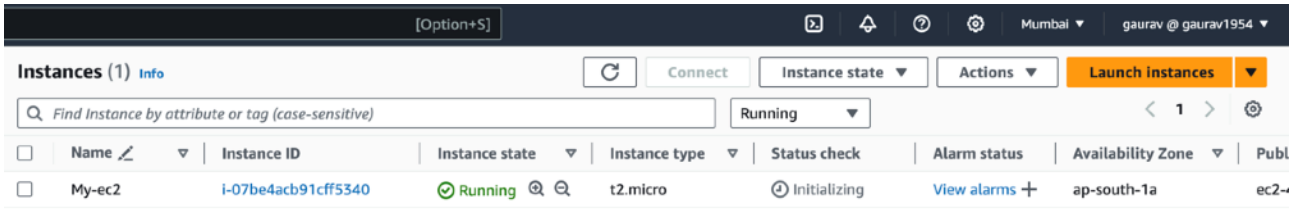
- Use the variable declared and defined in variable.tf in instance.tf

instance.tf variable.tf

instance.tf > ...

```
resource "aws_instance" "UPES1" {
  ami           = var.ubuntu_ami
  instance_type = var.instance_type
  count         = var.countNumber
  tags = {
    Name = "My-ec2"
  }
}
```

- Run the command terraform plan and review the plan to see if it meets your expectations



```
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:

  # aws_instance.UPES1[0] will be created
  + resource "aws_instance" "UPES1" {
      + ami                                  = "ami-03f4878755434977f"
      + arn                                  = (known after apply)
      + user_data_replace_on_change          = false
      + vpc_security_group_ids               = (known after apply)
    }

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.
```

- Run terraform apply and create the resources.

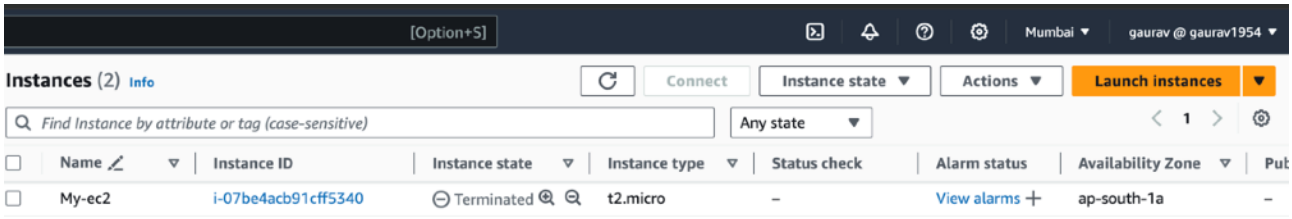
```
Terraform used the selected providers to generate the following execution plan. Resource actions are
symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.UPES1[0] will be created
  + resource "aws_instance" "UPES1" {
      + ami                                  = "ami-03f4878755434977f"
```

```
Plan: 1 to add, 0 to change, 0 to destroy.
aws_instance.UPES1[0]: Creating...
aws_instance.UPES1[0]: Still creating... [10s elapsed]
aws_instance.UPES1[0]: Still creating... [20s elapsed]
aws_instance.UPES1[0]: Still creating... [30s elapsed]
aws_instance.UPES1[0]: Creation complete after 33s [id=i-07be4acb91cff5340]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

- When you are done experimenting, run the following command to destroy the createdre sources.

```
gauravbhandari@gauravs-Air-2 aws-terraform-demo % terraform destroy --auto-approve
aws_instance.UPES1[0]: Refreshing state... [id=i-07be4acb91cff5340]

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

Terraform will perform the following actions:
```

```
Plan: 0 to add, 0 to change, 1 to destroy.
aws_instance.UPES1[0]: Destroying... [id=i-07be4acb91cff5340]
aws_instance.UPES1[0]: Still destroying... [id=i-07be4acb91cff5340, 10s elapsed]
aws_instance.UPES1[0]: Still destroying... [id=i-07be4acb91cff5340, 20s elapsed]
aws_instance.UPES1[0]: Still destroying... [id=i-07be4acb91cff5340, 30s elapsed]
aws_instance.UPES1[0]: Destruction complete after 31s

Destroy complete! Resources: 1 destroyed.
```



SPCM LAB-5

Objective: Learn how to pass values to Terraform variables using command line arguments.

- Create a file variable.tf with the following contents.

```
gauravbhandari@gauravs-MacBook-Air-2 aws-terraform-demo % touch variable.tf
```

instance.tf | variable.tf

variable.tf > variable "countNumber"

```
variable "ubuntu_ami" {
  type    = string
  default = "ami-03f4878755434977f"
}
variable "instance_type" {
  type    = string
  default = "t2.micro"
}
variable "countNumber" {
  type    = number
  default = 1
}
```

- Use the variable declared and defined in variable.tf in instance.tf

instance.tf | variable.tf

instance.tf > ...

```
resource "aws_instance" "UPES1" {
  ami           = var.ubuntu_ami
  instance_type = var.instance_type
  count         = var.countNumber
  tags = {
    Name = "My-ec2"
  }
}
```

- Run the command terraform plan and review the plan to see if it meets your expectations

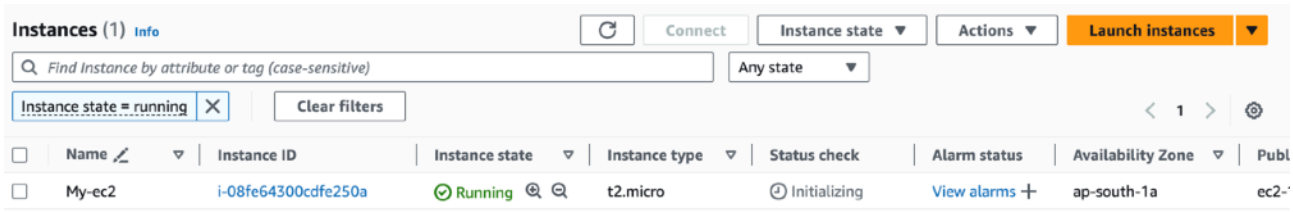
```
977f" -var "instance_type
=t2.micro" -var "countNumber=1"
```

```
      + user_data_replace_on_change          = false
      + vpc_security_group_ids               = (known after apply)
    }

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.
```

- Run terraform apply and create the resources.

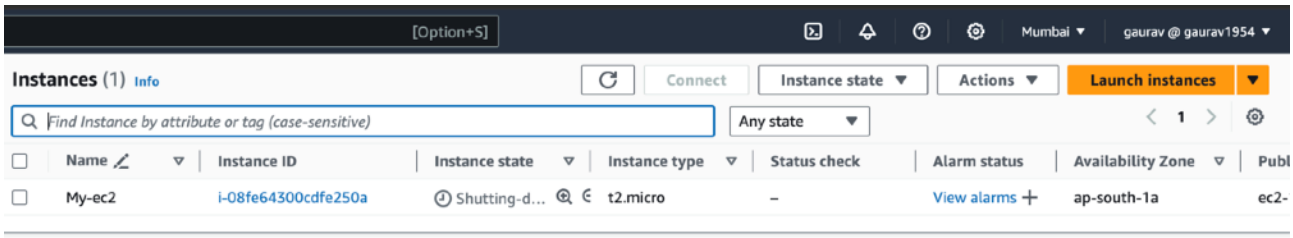
```
gauravbhandari@gauravs-MacBook-Air-2 aws-terraform-demo % terraform apply -var "ubuntu_ami=ami-03f487875543
4977f" -var "instance_type=t2.micro" -var "countNumber=1"
```

```
Plan: 1 to add, 0 to change, 0 to destroy.
aws_instance.UPES1[0]: Creating...
aws_instance.UPES1[0]: Still creating... [10s elapsed]
aws_instance.UPES1[0]: Still creating... [20s elapsed]
aws_instance.UPES1[0]: Still creating... [30s elapsed]
aws_instance.UPES1[0]: Creation complete after 33s [id=i-08fe64300cdfe250a]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```



- When you are done experimenting, run the following command to destroy the created resources.

```
Plan: 0 to add, 0 to change, 1 to destroy.
aws_instance.UPES1[0]: Destroying... [id=i-08fe64300cdfe250a]
aws_instance.UPES1[0]: Still destroying... [id=i-08fe64300cdfe250a, 10s elapsed]
aws_instance.UPES1[0]: Still destroying... [id=i-08fe64300cdfe250a, 20s elapsed]
aws_instance.UPES1[0]: Still destroying... [id=i-08fe64300cdfe250a, 30s elapsed]
aws_instance.UPES1[0]: Destruction complete after 32s

Destroy complete! Resources: 1 destroyed.
```

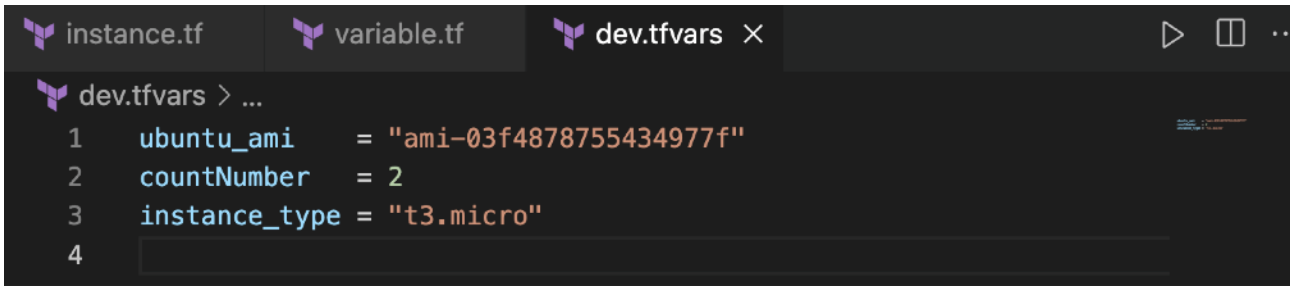


SPCM LAB-6

Objective: Learn how to use multiple tfvars files in Terraform for different environments.

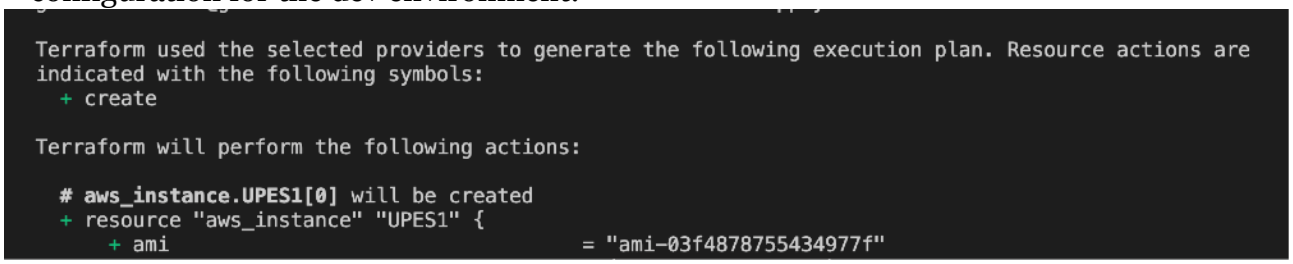
• Create multiple tfvar files

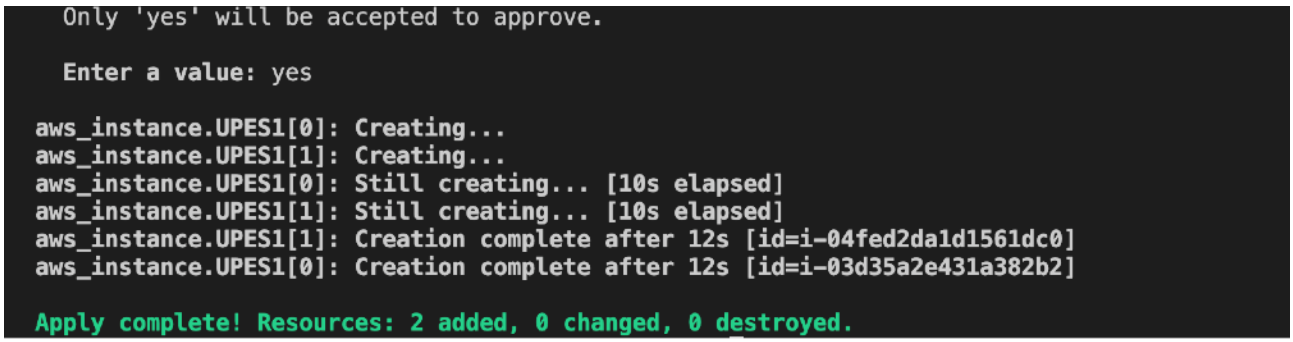
- Create a dev.tfvar file

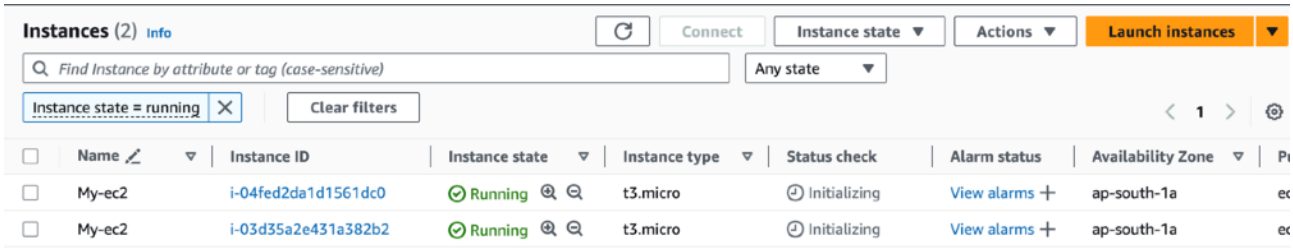


- Create a prod.tfvar file

• Run the terraform apply -var-file=dev.tfvars commands to initialize and apply the configuration for the dev environment:



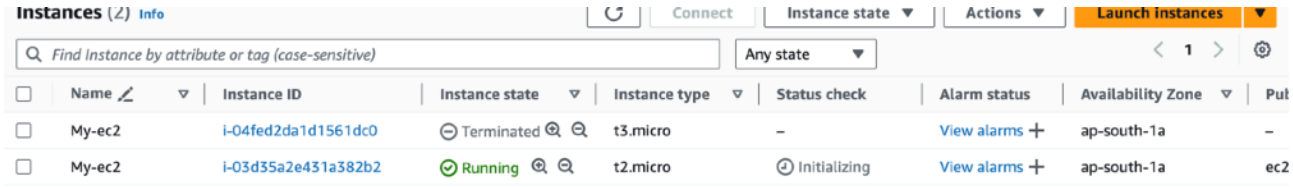




- Run the following Terraform commands to initialize and apply the configurationf or the prod environment:

```
aws_instance.UPES1[0]: Refreshing state... [id=i-03d35a2e431a382b2]
aws_instance.UPES1[1]: Refreshing state... [id=i-04fed2da1d1561dc0]

Terraform used the selected providers to generate the following execution plan. Resource actions are
indicated with the following symbols:
  ~ update in-place
  - destroy

Terraform will perform the following actions:
```

```
aws_instance.UPES1[0]: Still modifying... [id=i-03d35a2e431a382b2, 50s elapsed]
aws_instance.UPES1[0]: Still modifying... [id=i-03d35a2e431a382b2, 1m0s elapsed]
aws_instance.UPES1[0]: Still modifying... [id=i-03d35a2e431a382b2, 1m10s elapsed]
aws_instance.UPES1[0]: Still modifying... [id=i-03d35a2e431a382b2, 1m20s elapsed]
aws_instance.UPES1[0]: Still modifying... [id=i-03d35a2e431a382b2, 1m30s elapsed]
aws_instance.UPES1[0]: Still modifying... [id=i-03d35a2e431a382b2, 1m40s elapsed]
aws_instance.UPES1[0]: Modifications complete after 1m42s [id=i-03d35a2e431a382b2]

Apply complete! Resources: 0 added, 1 changed, 1 destroyed.
```



- After successful experimentation clean up the environment using terraform destroy -varfile=prod.tfvars

```
aws_instance.UPES1[0]: Refreshing state... [id=i-03d35a2e431a382b2]

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

Terraform will perform the following actions:

  # aws_instance.UPES1[0] will be destroyed
  - resource "aws_instance" "UPES1" {
```

```
Plan: 0 to add, 0 to change, 1 to destroy.
aws_instance.UPES1[0]: Destroying... [id=i-03d35a2e431a382b2]
aws_instance.UPES1[0]: Still destroying... [id=i-03d35a2e431a382b2, 10s elapsed]
aws_instance.UPES1[0]: Still destroying... [id=i-03d35a2e431a382b2, 20s elapsed]
aws_instance.UPES1[0]: Still destroying... [id=i-03d35a2e431a382b2, 30s elapsed]
aws_instance.UPES1[0]: Destruction complete after 30s

Destroy complete! Resources: 1 destroyed.
```