Lesson 10 Demo 06

Migrating Terraform State

Objective: To migrate Terraform state between different backends, including the default local backend, AWS S3 backend, and Terraform Cloud remote backend for achieving seamless state management across various environments

Tools required: Visual Studio Code

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

Note: The folder structure created in the previous demos is used here. It is also included in the resources section of LMS. Please refer to Lesson_10_demo_01

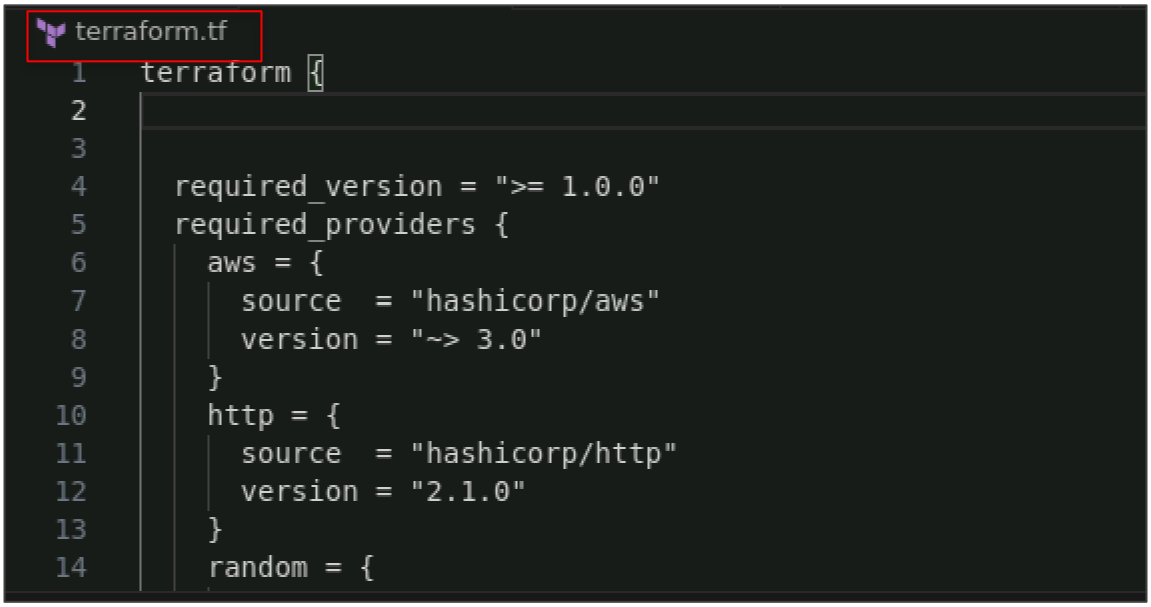
Steps to be followed:

1. Use Terraform default local backend
2. Migrate state to S3 backend
3. Migrate state to remote backend
4. Migrate back to local backend

Step 1: Use Terraform default local backend

1.1 Update the **terraform.tf** configuration block to remove any backend configuration, indicating that Terraform uses its default local backend, using the following code:

```
terraform {
  required_version = ">= 1.0.0"
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 3.0"
    }
    http = {
      source  = "hashicorp/http"
      version = "2.1.0"
    }
    random = {
      source  = "hashicorp/random"
      version = "3.1.0"
    }
    local = {
      source  = "hashicorp/local"
      version = "2.1.0"
    }
    tls = {
      source  = "hashicorp/tls"
      version = "3.1.0"
    }
  }
}
```

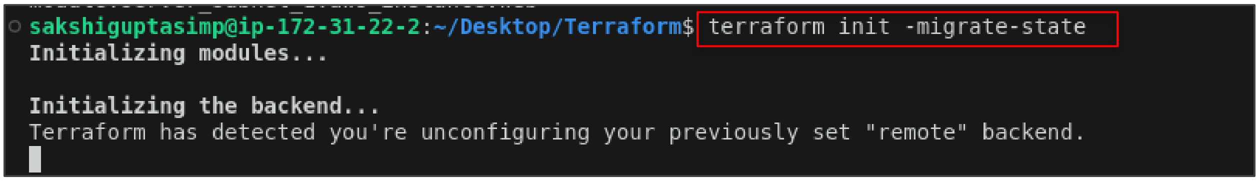


terraform.tf

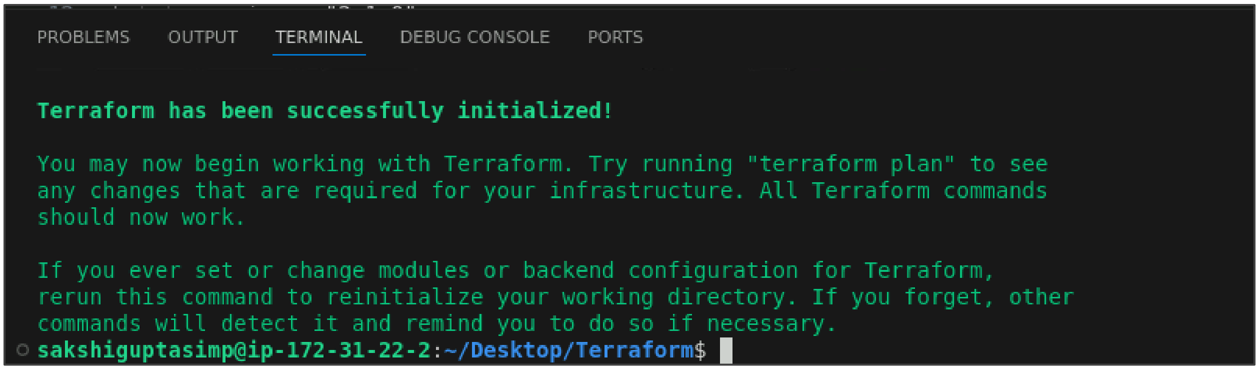
```
terraform {


  required_version = ">= 1.0.0"
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 3.0"
    }
    http = {
      source  = "hashicorp/http"
      version = "2.1.0"
    }
    random = {
```

1.2 Initialize the configuration to use the local backend using the following command:

terraform init -migrate-state



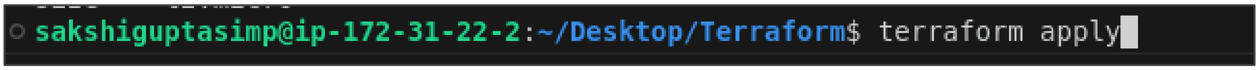
```
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform init -migrate-state
Initializing modules...

Initializing the backend...
Terraform has detected you're unconfiguring your previously set "remote" backend.
```



```
Terraform has been successfully initialized!

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.
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$
```

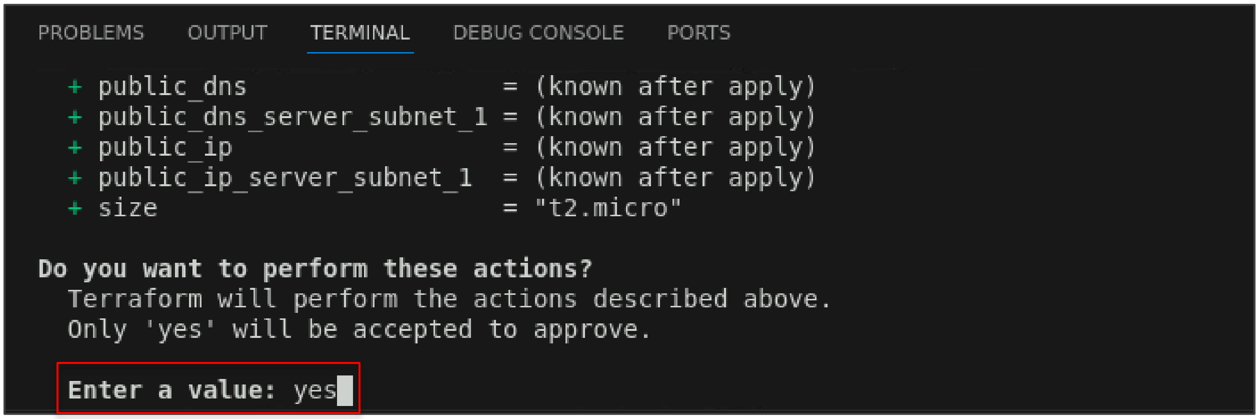
1.3 Apply the Terraform configuration using the following command:

terraform apply

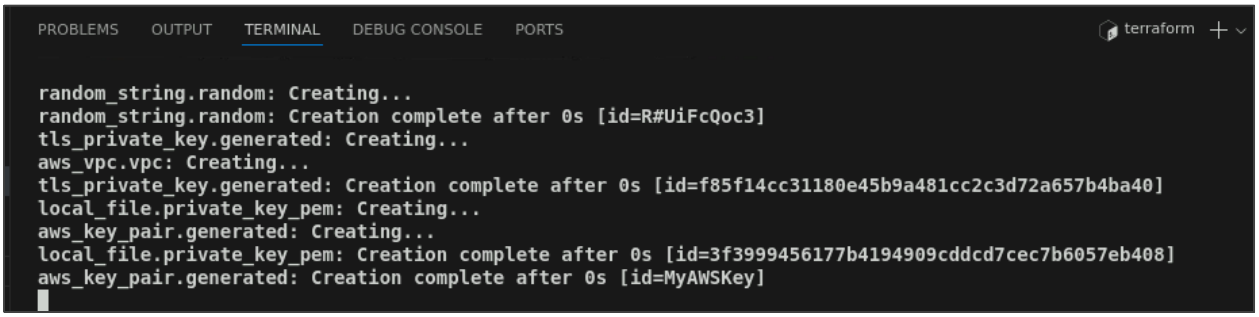


```
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform apply
```

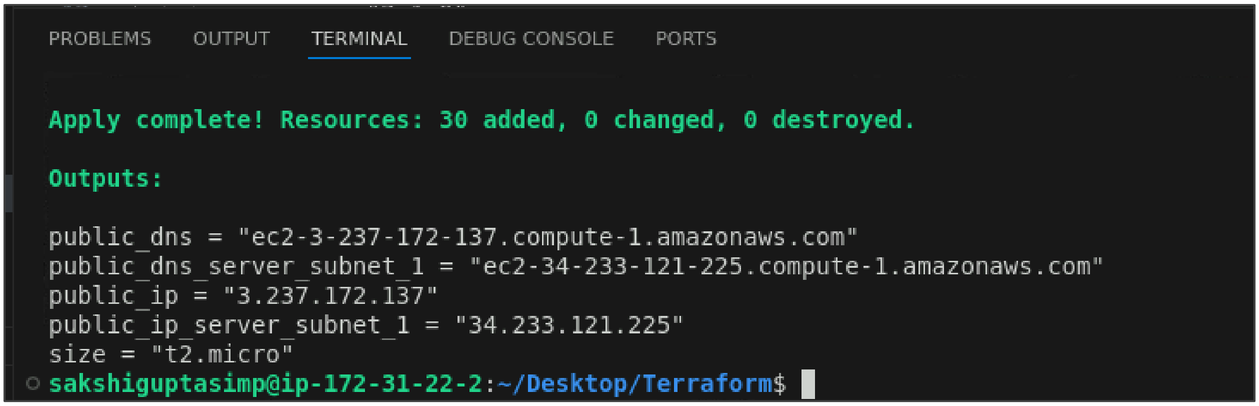
1.4 When prompted, approve the changes by typing **yes**



```
  + public_dns                    = (known after apply)
  + public_dns_server_subnet_1 = (known after apply)
  + public_ip                     = (known after apply)
  + public_ip_server_subnet_1  = (known after apply)
  + size                          = "t2.micro"

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
```



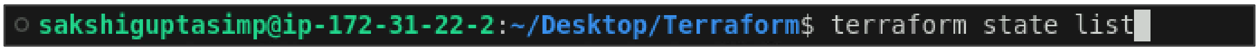
```
random_string.random: Creating...
random_string.random: Creation complete after 0s [id=R#UiFcQoc3]
tls_private_key.generated: Creating...
aws_vpc.vpc: Creating...
tls_private_key.generated: Creation complete after 0s [id=f85f14cc31180e45b9a481cc2c3d72a657b4ba40]
local_file.private_key_pem: Creating...
aws_key_pair.generated: Creating...
local_file.private_key_pem: Creation complete after 0s [id=3f3999456177b4194909cddcd7cec7b6057eb408]
aws_key_pair.generated: Creation complete after 0s [id=MyAWSKey]
```



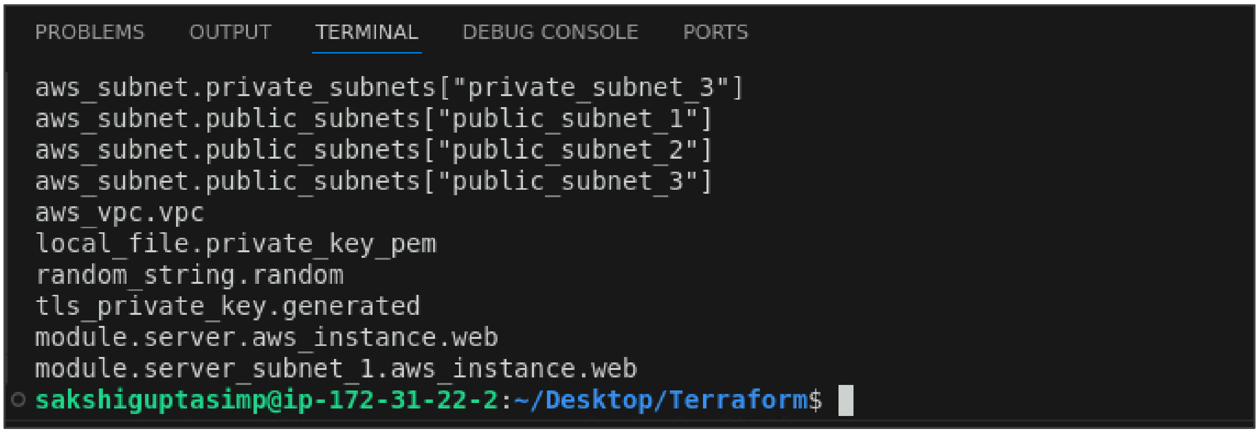
```
Apply complete! Resources: 30 added, 0 changed, 0 destroyed.

Outputs:

public_dns = "ec2-3-237-172-137.compute-1.amazonaws.com"
public_dns_server_subnet_1 = "ec2-34-233-121-225.compute-1.amazonaws.com"
public_ip = "3.237.172.137"
public_ip_server_subnet_1 = "34.233.121.225"
size = "t2.micro"
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$
```

1.5 List the newly created items in the state by using the following command:

terraform state list



```
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform state list
```

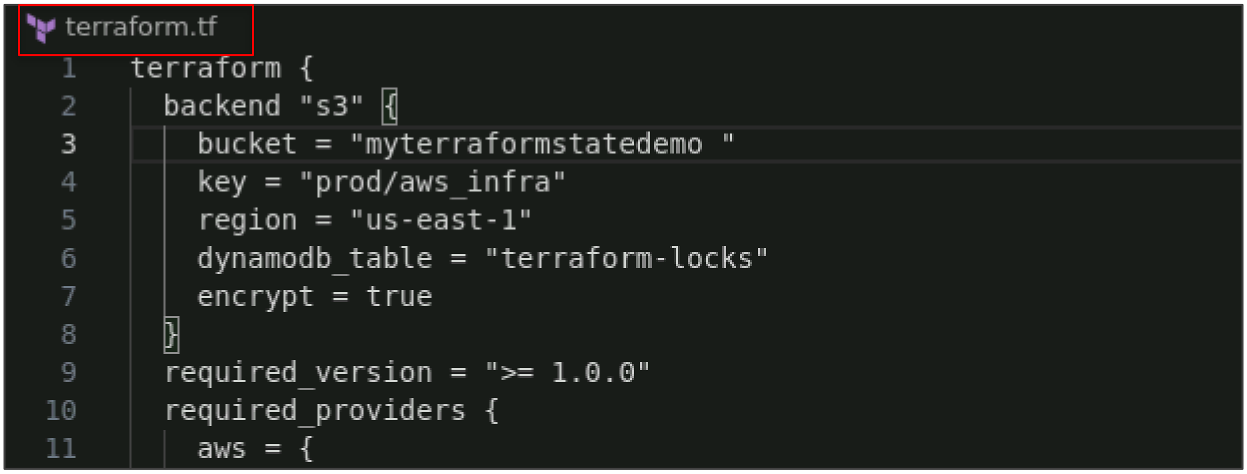


```
aws_subnet.private_subnets["private_subnet_3"]
aws_subnet.public_subnets["public_subnet_1"]
aws_subnet.public_subnets["public_subnet_2"]
aws_subnet.public_subnets["public_subnet_3"]
aws_vpc.vpc
local_file.private_key_pem
random_string.random
tls_private_key.generated
module.server.aws_instance.web
module.server_subnet_1.aws_instance.web
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$
```

Step 2: Migrate state to S3 backend

2.1 Update the **terraform.tf** configuration block to use the S3 backend with the following code:

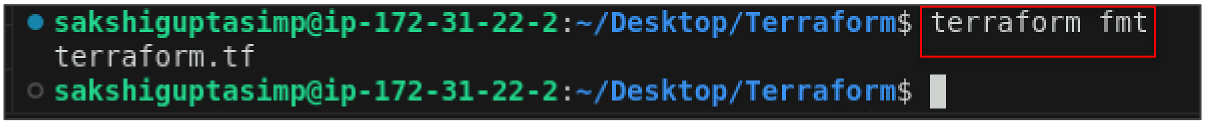
```
terraform {
  backend "s3" {
    bucket         = "myterraformstatedemo"
    key            = "prod/aws_infra"
    region         = "us-east-1"
    dynamodb_table = "terraform-locks"
    encrypt        = true
  }
}
```



```
terraform {
  backend "s3" {
    bucket = "myterraformstatedemo "
    key = "prod/aws_infra"
    region = "us-east-1"
    dynamodb_table = "terraform-locks"
    encrypt = true
  }
  required_version = ">= 1.0.0"
  required_providers {
    aws = {
```

2.2 Format the configuration by using the following command:

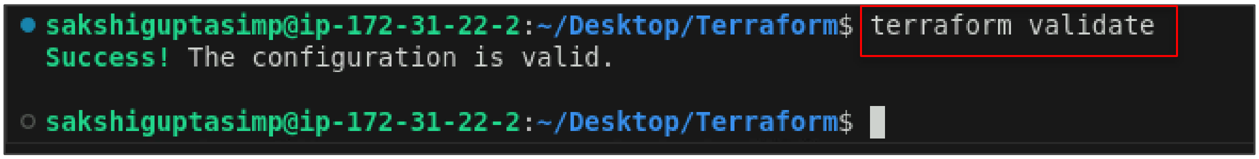
terraform fmt



```
● sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform fmt
  terraform.tf
○ sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ 
```

2.3 Validate the configuration by using the following command:

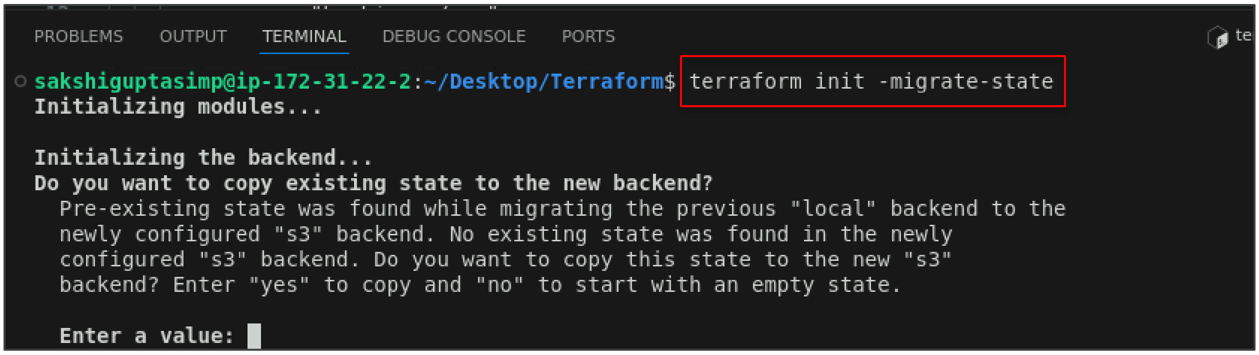
terraform validate



```
● sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform validate
  Success! The configuration is valid.

○ sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ 
```

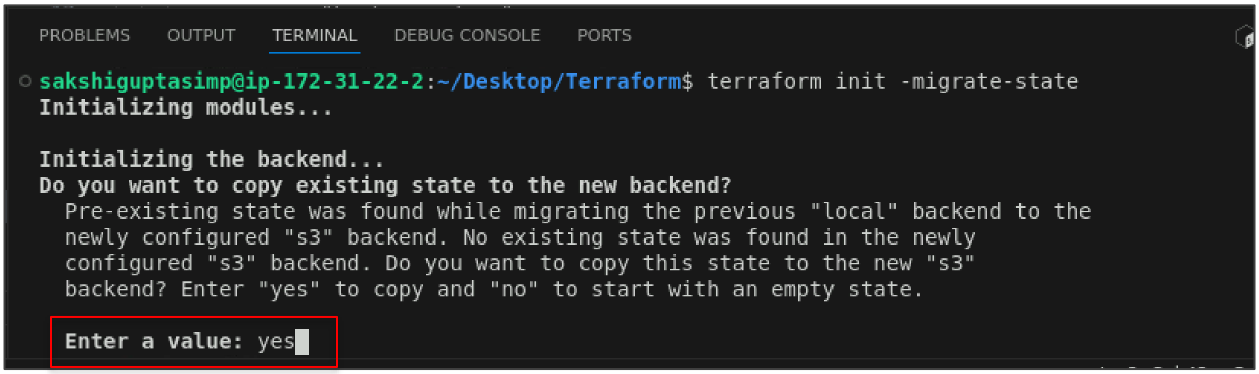
2.4 Initialize the configuration to use the S3 backend by using the following command:

terraform init -migrate-state

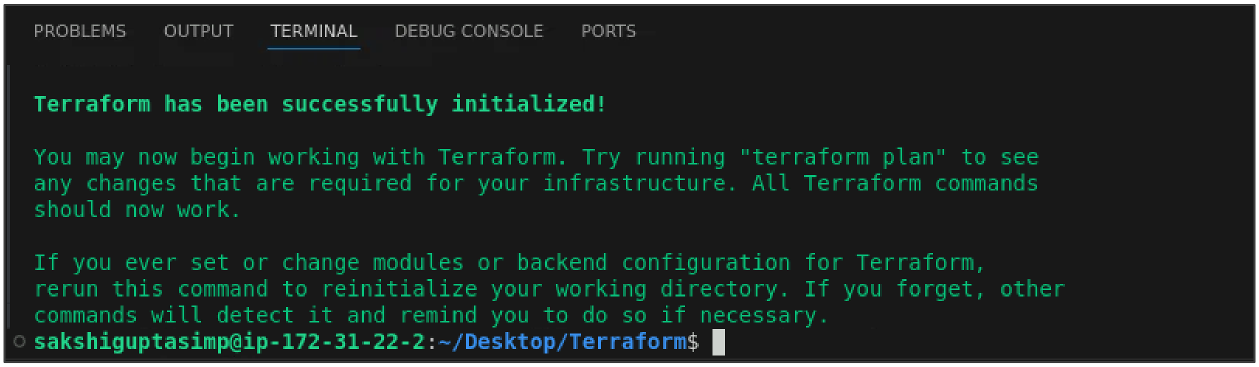


```
○ sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform init -migrate-state
  Initializing modules...

  Initializing the backend...
  Do you want to copy existing state to the new backend?
    Pre-existing state was found while migrating the previous "local" backend to the
    newly configured "s3" backend. No existing state was found in the newly
    configured "s3" backend. Do you want to copy this state to the new "s3"
    backend? Enter "yes" to copy and "no" to start with an empty state.

    Enter a value: 
```

2.5 When prompted, approve the initialization by typing **yes**



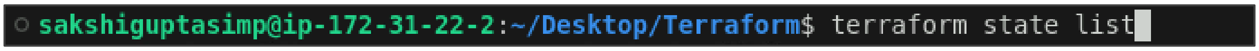
```
○ sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform init -migrate-state
  Initializing modules...

  Initializing the backend...
  Do you want to copy existing state to the new backend?
    Pre-existing state was found while migrating the previous "local" backend to the
    newly configured "s3" backend. No existing state was found in the newly
    configured "s3" backend. Do you want to copy this state to the new "s3"
    backend? Enter "yes" to copy and "no" to start with an empty state.

    Enter a value: yes
```



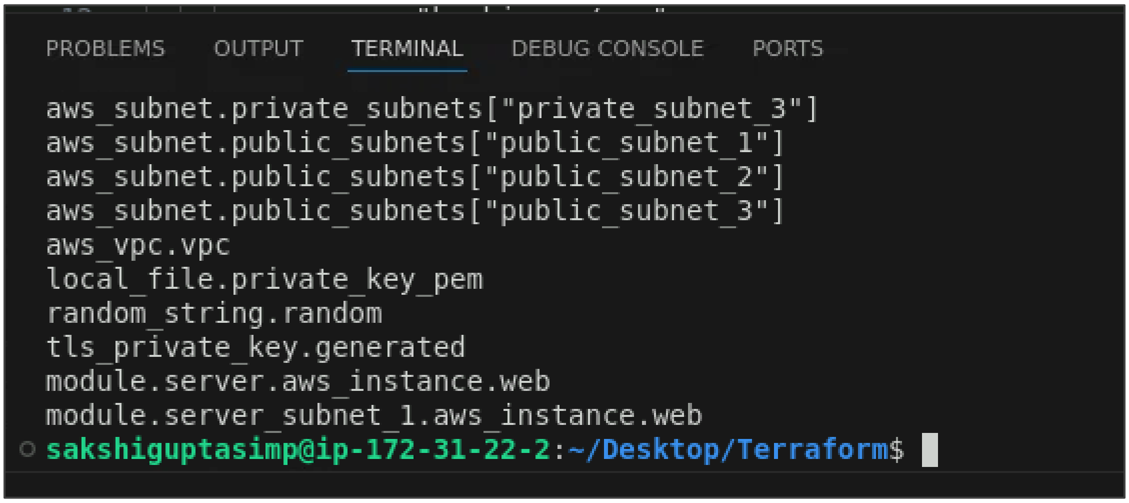
```
Terraform has been successfully initialized!

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.
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$
```

2.6 List the items in the state by using the following command:

terraform state list

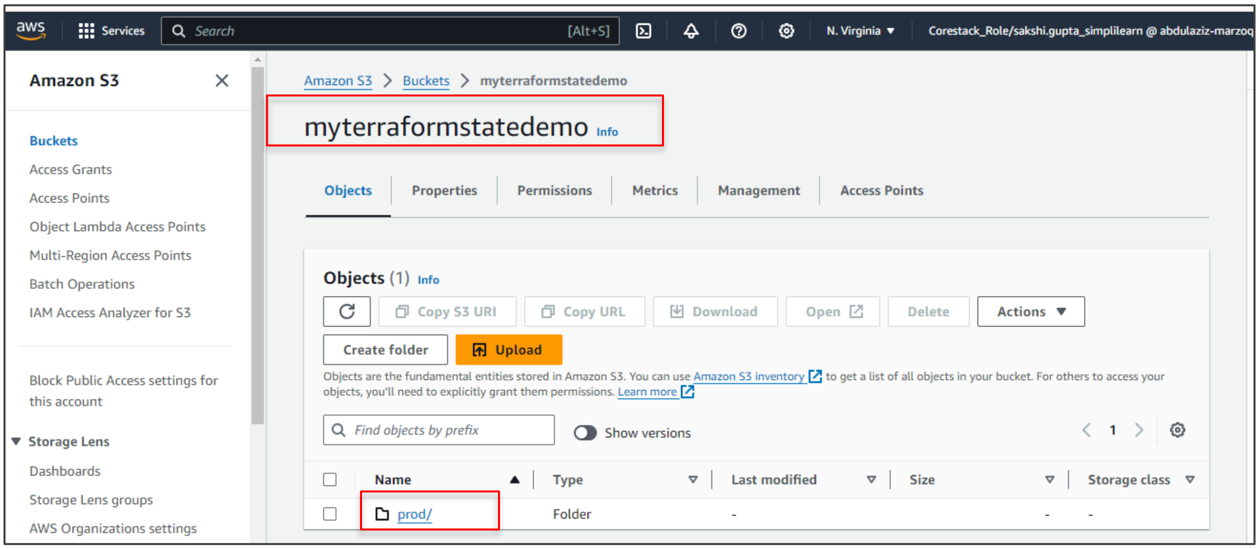


```
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform state list
```



```
aws_subnet.private_subnets["private_subnet_3"]
aws_subnet.public_subnets["public_subnet_1"]
aws_subnet.public_subnets["public_subnet_2"]
aws_subnet.public_subnets["public_subnet_3"]
aws_vpc.vpc
local_file.private_key_pem
random_string.random
tls_private_key.generated
module.server.aws_instance.web
module.server_subnet_1.aws_instance.web
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$
```

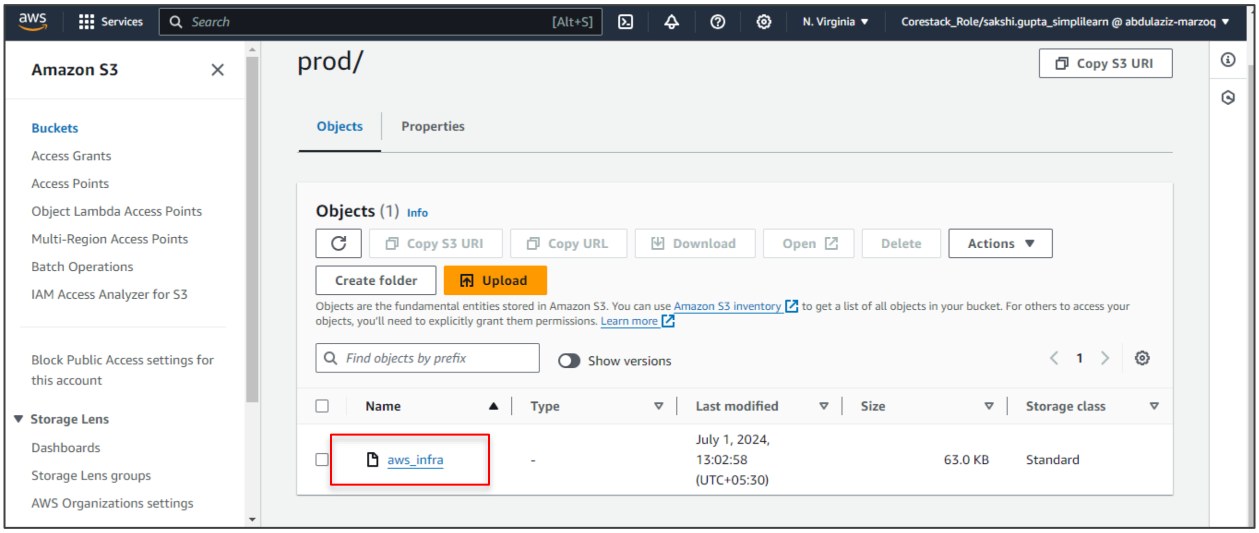
2.7 Log in to your AWS console and go to the bucket named **myterraformstatedemo,** and then click on **prod/**



Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
prod/	Folder	-	-	-

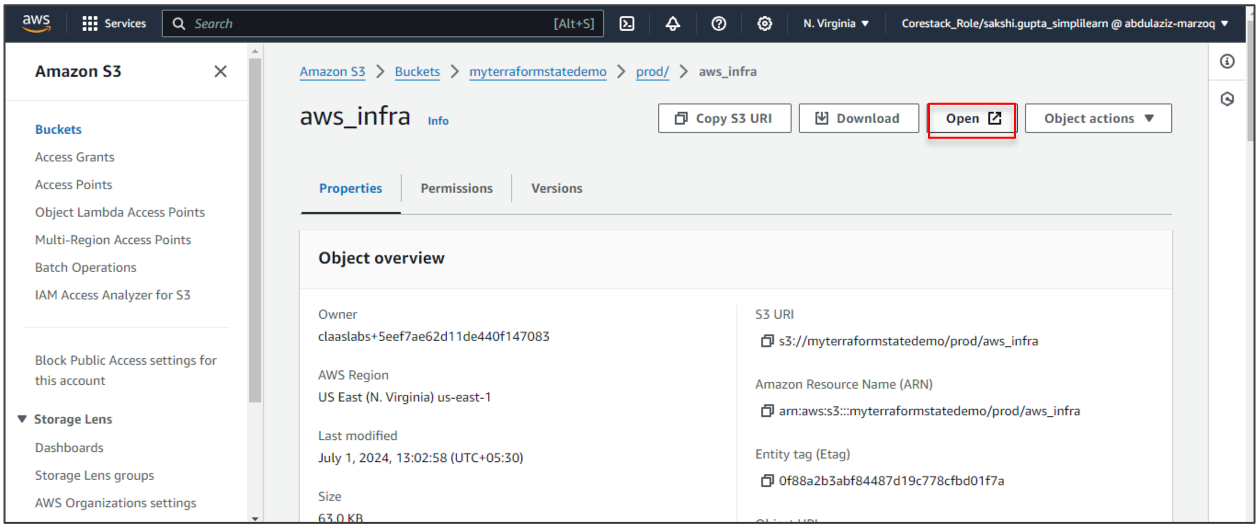
2.8 Click on **aws_infra**



Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
aws_infra	-	July 1, 2024, 13:02:58 (UTC+05:30)	63.0 KB	Standard

2.9 Click on **Open** and you will now be able to see the migrated Terraform state file in your S3 backend


aws
Services
Search
[Alt+S]
N. Virginia ▼
Corestack_Role/sakshi.gupta_simplilearn @ abdulaziz-marzoq ▼
Amazon S3
Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
▼ Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Amazon S3 > Buckets > myterraformstatedemo > prod/ > aws_infra
aws_infra Info
Copy S3 URI
Download
Open
Object actions ▼
Properties
Permissions
Versions

Object overview

Owner
claaslabs+5eef7ae62d11de440f147083

AWS Region
US East (N. Virginia) us-east-1

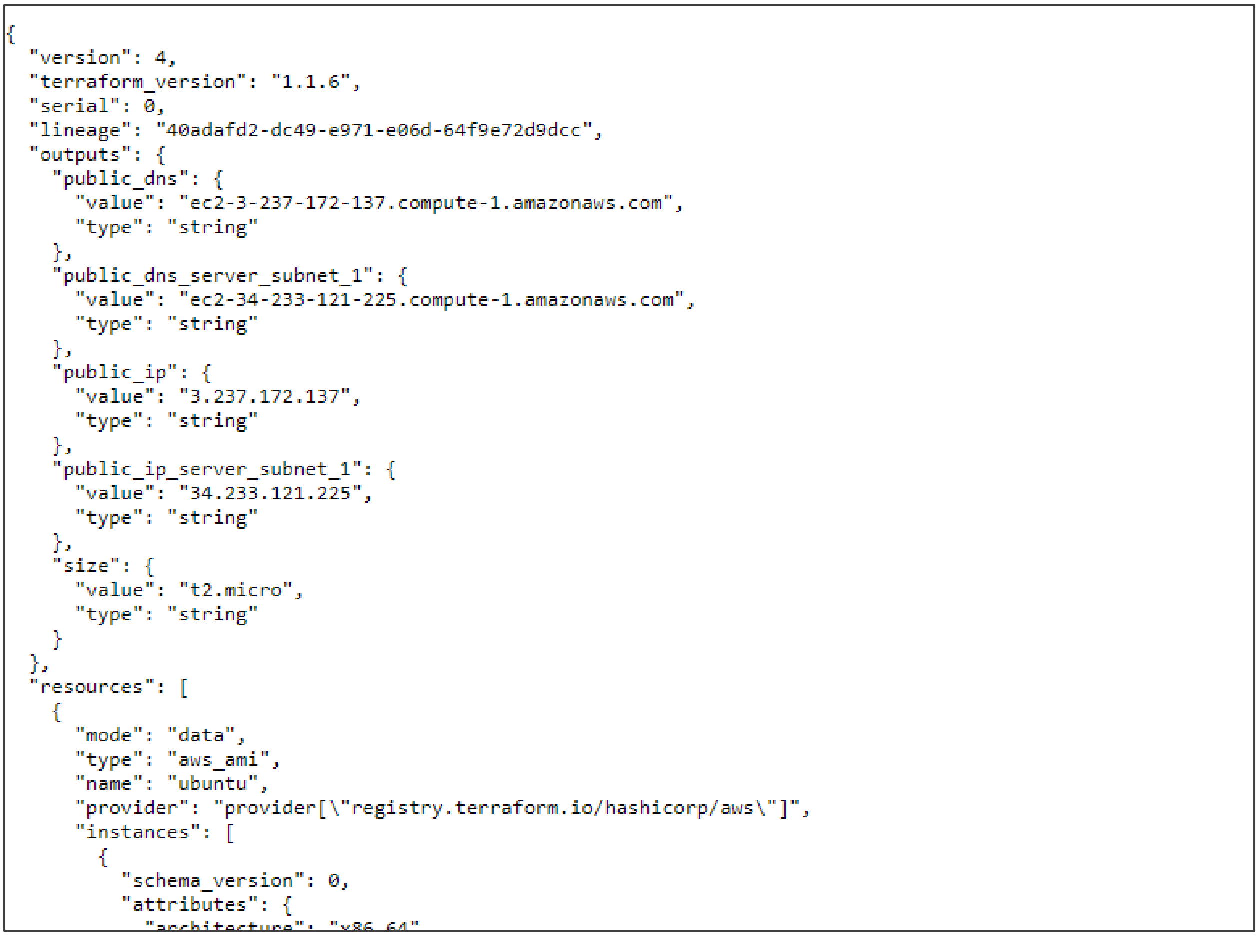
Last modified
July 1, 2024, 13:02:58 (UTC+05:30)

Size
63.0 KB

S3 URI
s3://myterraformstatedemo/prod/aws_infra

Amazon Resource Name (ARN)
arn:aws:s3:::myterraformstatedemo/prod/aws_infra

Entity tag (Etag)
0f88a2b3abf84487d19c778cfbd01f7a

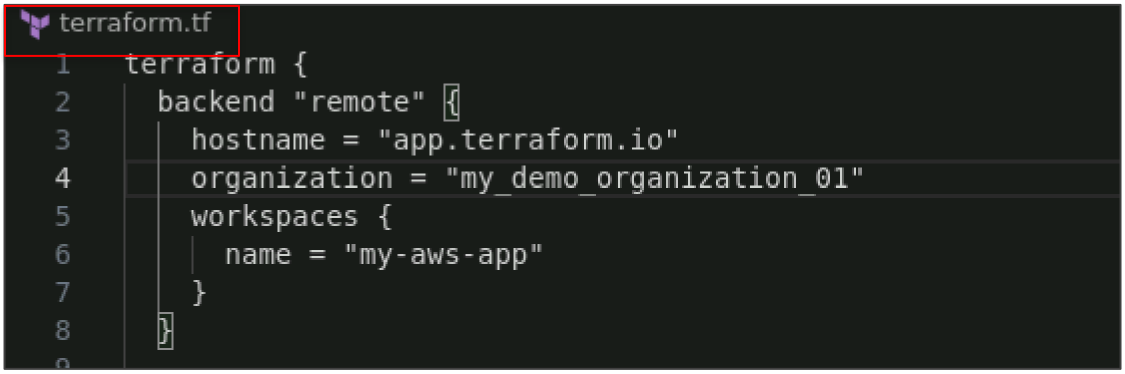


```
{
  "version": 4,
  "terraform_version": "1.1.6",
  "serial": 0,
  "lineage": "40adafd2-dc49-e971-e06d-64f9e72d9dcc",
  "outputs": {
    "public_dns": {
      "value": "ec2-3-237-172-137.compute-1.amazonaws.com",
      "type": "string"
    },
    "public_dns_server_subnet_1": {
      "value": "ec2-34-233-121-225.compute-1.amazonaws.com",
      "type": "string"
    },
    "public_ip": {
      "value": "3.237.172.137",
      "type": "string"
    },
    "public_ip_server_subnet_1": {
      "value": "34.233.121.225",
      "type": "string"
    },
    "size": {
      "value": "t2.micro",
      "type": "string"
    }
  },
  "resources": [
    {
      "mode": "data",
      "type": "aws_ami",
      "name": "ubuntu",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "architecture": "x86_64"
```

Step 3: Migrate state to remote backend

3.1 Update the **terraform.tf** configuration block to use the remote backend with the following code:

```
terraform {
 backend "remote" {
  hostname     = "app.terraform.io"
  organization = "my_demo_organisation_01"
  workspaces {
   name = "my-aws-app"
  }
 }
}
```

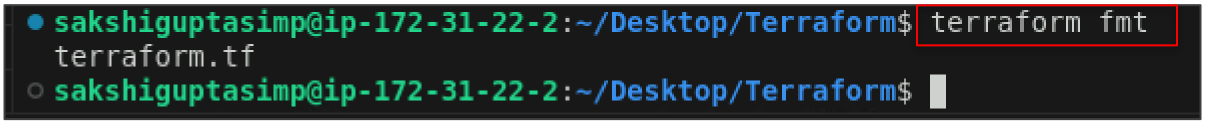


terraform.tf

```
terraform {
  backend "remote" {
    hostname = "app.terraform.io"
    organization = "my_demo_organization_01"
    workspaces {
      name = "my-aws-app"
    }
  }
```

3.2 Format the configuration by using the following command:

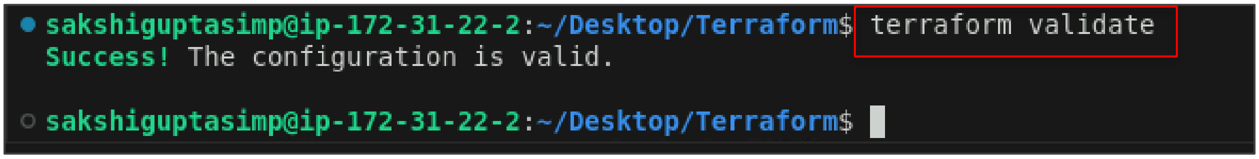
terraform fmt



```
● sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform fmt
  terraform.tf
○ sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ 
```

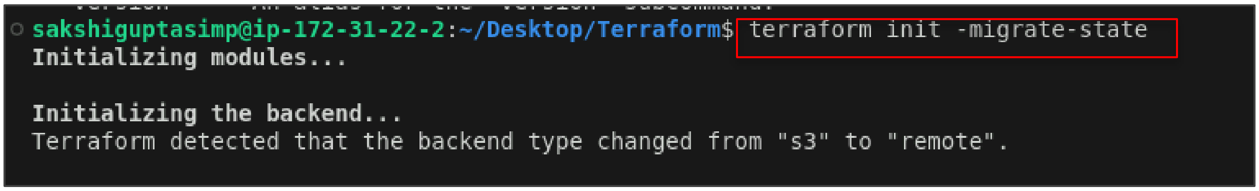
3.3 Validate the configuration by using the following command:

terraform validate



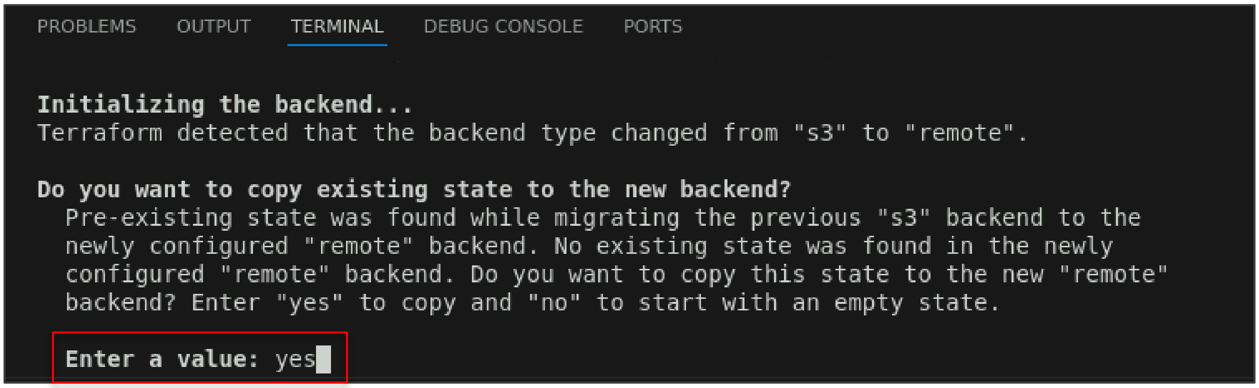
```
● sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform validate
  Success! The configuration is valid.

○ sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ 
```

3.4 Initialize the configuration to use the remote backend by using the following command: **terraform init -migrate-state**

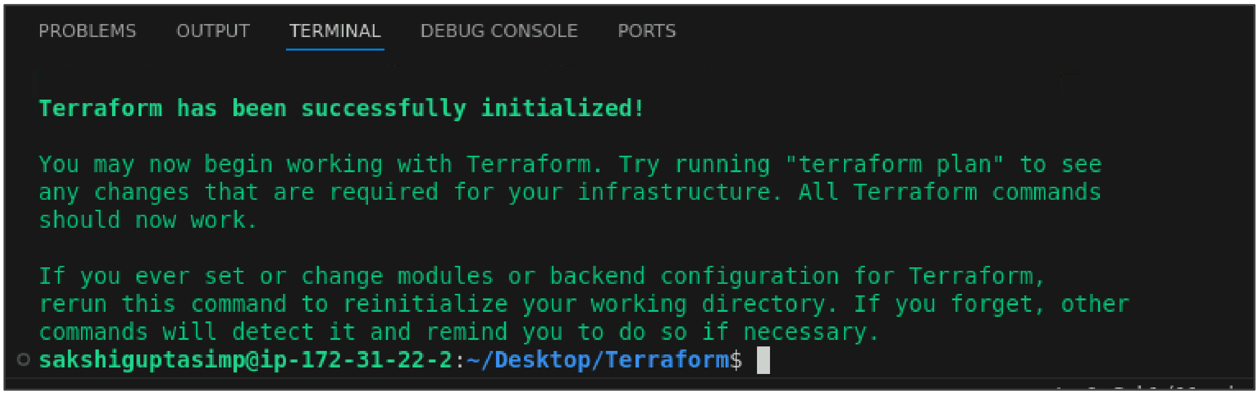


```
  version      An alias for the "version" subcommand.
○ sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform init -migrate-state
  Initializing modules...

  Initializing the backend...
  Terraform detected that the backend type changed from "s3" to "remote".
```

3.5 When prompted, approve the initialization by typing **yes**

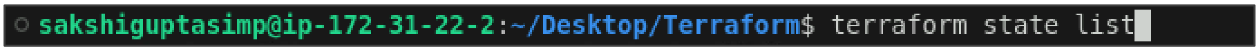


```
Initializing the backend...
Terraform detected that the backend type changed from "s3" to "remote".

Do you want to copy existing state to the new backend?
  Pre-existing state was found while migrating the previous "s3" backend to the
  newly configured "remote" backend. No existing state was found in the newly
  configured "remote" backend. Do you want to copy this state to the new "remote"
  backend? Enter "yes" to copy and "no" to start with an empty state.

  Enter a value: yes
```

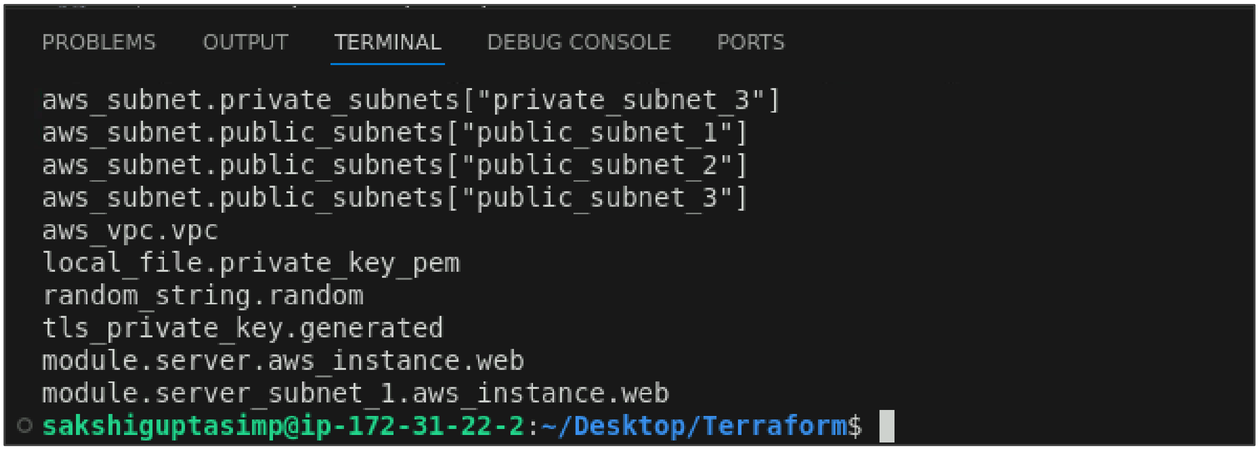


```
Terraform has been successfully initialized!

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.
○ sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$
```

3.6 List the items in the state by using the following command: **terraform state list**

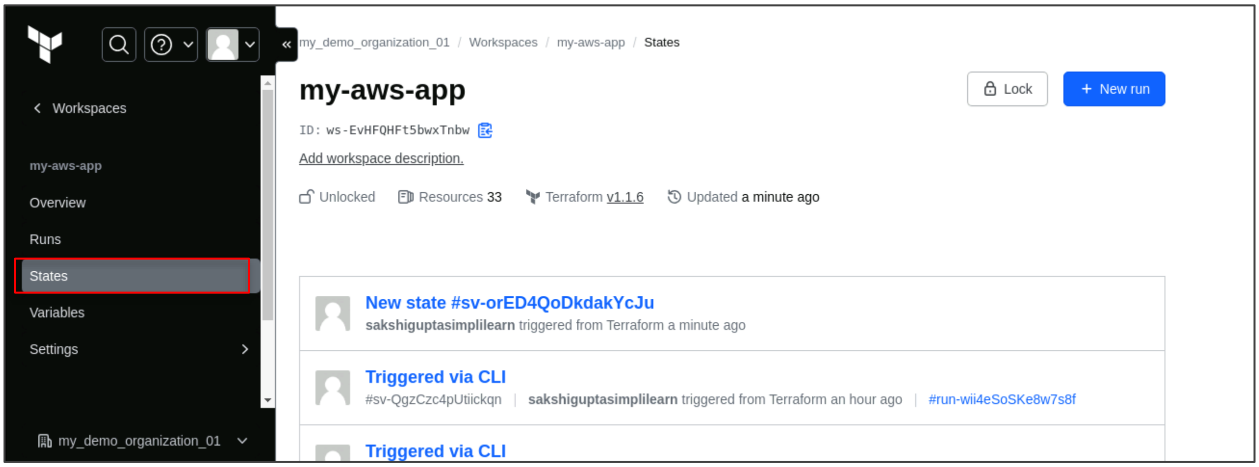


```
○ sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform state list
```



```
aws_subnet.private_subnets["private_subnet_3"]
aws_subnet.public_subnets["public_subnet_1"]
aws_subnet.public_subnets["public_subnet_2"]
aws_subnet.public_subnets["public_subnet_3"]
aws_vpc.vpc
local_file.private_key_pem
random_string.random
tls_private_key.generated
module.server.aws_instance.web
module.server_subnet_1.aws_instance.web
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$
```

3.7 Go to your Terraform Cloud workspace dashboard and click on **States**



my_demo_organization_01 / Workspaces / my-aws-app / States

my-aws-app

ID: ws-EvHFQHFt5bwxTnbw

Add workspace description.

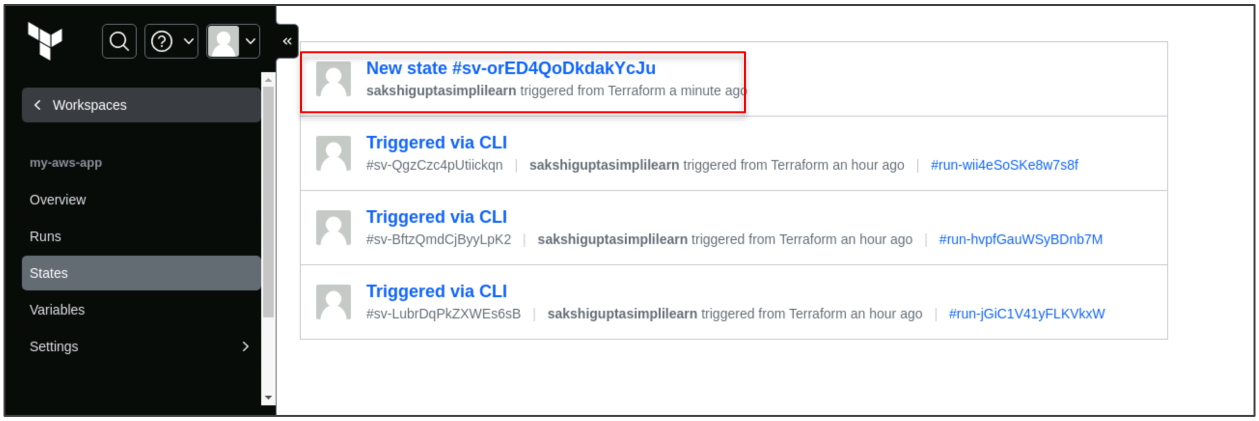
Unlocked   Resources 33   Terraform v1.1.6   Updated a minute ago

New state #sv-orED4QoDkdakYcJu
sakshiguptasimplilearn triggered from Terraform a minute ago

Triggered via CLI
#sv-QgzCzc4pUtiickqn | sakshiguptasimplilearn triggered from Terraform an hour ago | #run-wii4eSoSKe8w7s8f

Triggered via CLI

3.8 Click on the newly created state entry and you will now be able to see the migrated Terraform state file in your remote backend

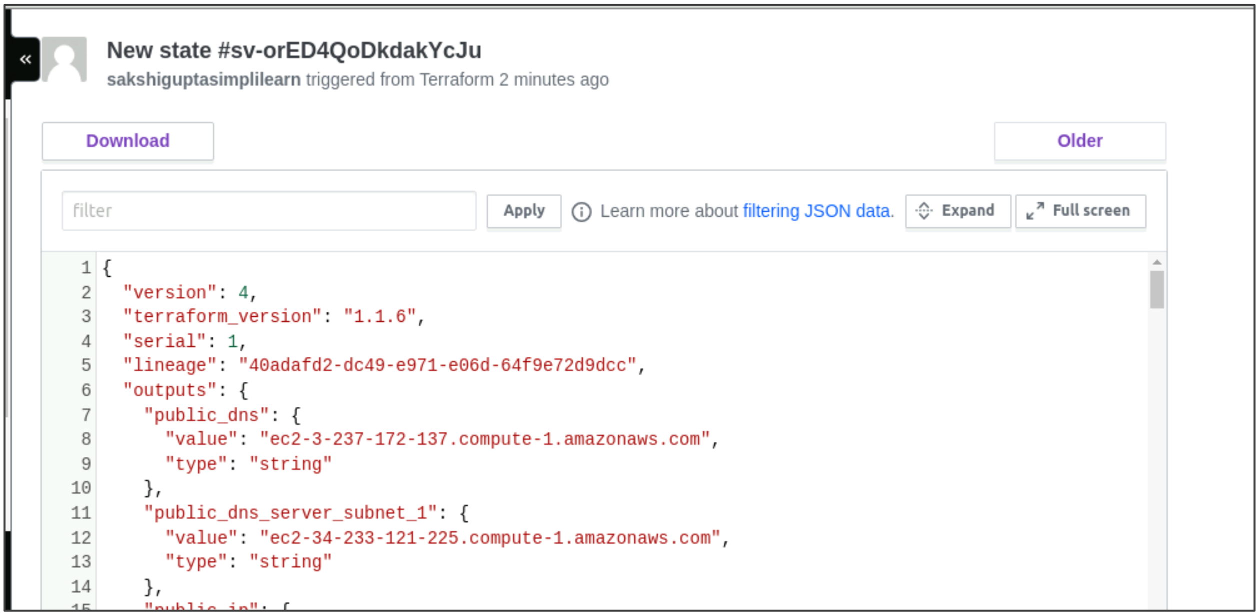


New state #sv-orED4QoDkdakYcJu
sakshiguptasimplilearn triggered from Terraform a minute ago

Triggered via CLI
#sv-QgzCzc4pUtiickqn | sakshiguptasimplilearn triggered from Terraform an hour ago | #run-wii4eSoSKe8w7s8f

Triggered via CLI
#sv-BftzQmdCjByyLpK2 | sakshiguptasimplilearn triggered from Terraform an hour ago | #run-hvpfGauWSyBDnb7M

Triggered via CLI
#sv-LubrDqPkZXWEs6sB | sakshiguptasimplilearn triggered from Terraform an hour ago | #run-jGiC1V41yFLKVkxW



New state #sv-orED4QoDkdakYcJu

sakshiguptasimplilearn triggered from Terraform 2 minutes ago

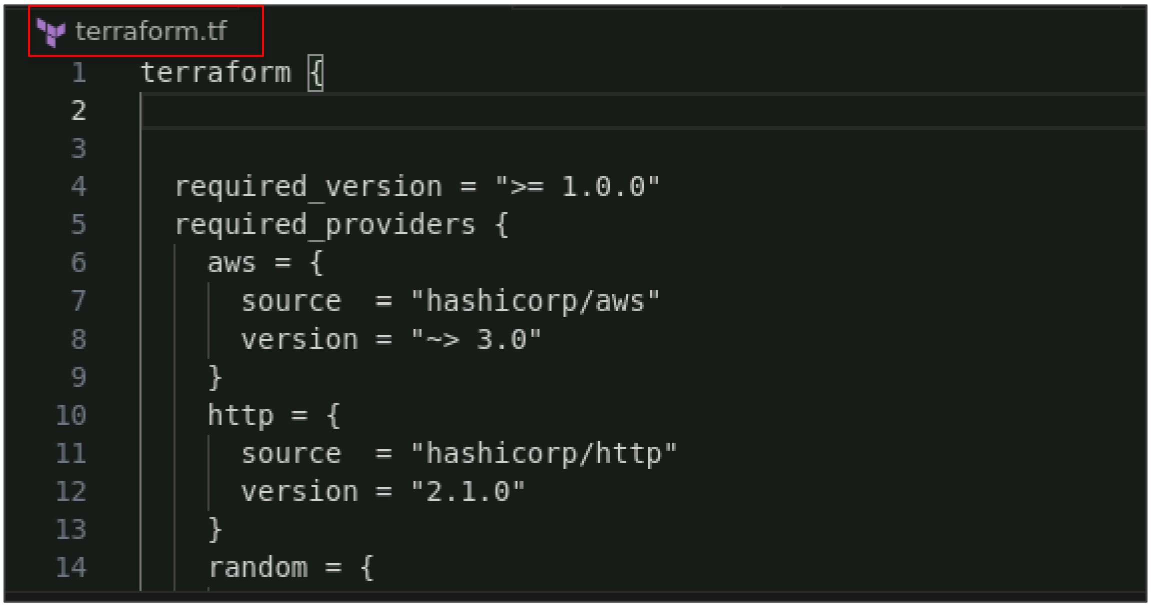
```
{
  "version": 4,
  "terraform_version": "1.1.6",
  "serial": 1,
  "lineage": "40adafd2-dc49-e971-e06d-64f9e72d9dcc",
  "outputs": {
    "public_dns": {
      "value": "ec2-3-237-172-137.compute-1.amazonaws.com",
      "type": "string"
    },
    "public_dns_server_subnet_1": {
      "value": "ec2-34-233-121-225.compute-1.amazonaws.com",
      "type": "string"
    },
    "public_ip": {
```

Step 4: Migrate back to local backend

4.1 Update the **terraform.tf** configuration block to remove any backend configuration, indicating that Terraform should use its default local backend, using the following code:

```
terraform {
 required_version = ">= 1.0.0"
 required_providers {
  aws = {
   source  = "hashicorp/aws"
   version = "~> 3.0"
  }
  http = {
   source  = "hashicorp/http"
   version = "2.1.0"
  }
  random = {
   source  = "hashicorp/random"
   version = "3.1.0"
  }
  local = {
   source  = "hashicorp/local"
```

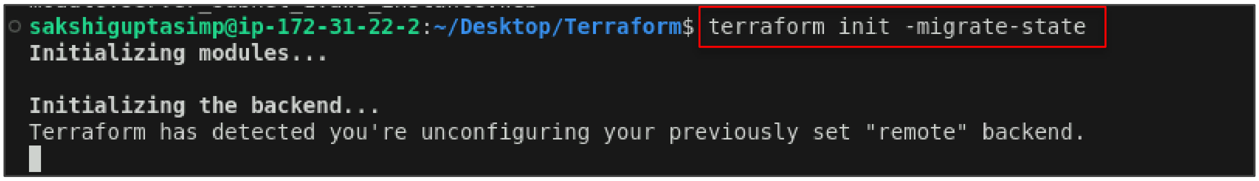
```
    version = "2.1.0"
   }
   tls = {
    source  = "hashicorp/tls"
    version = "3.1.0"
   }
  }
}
```



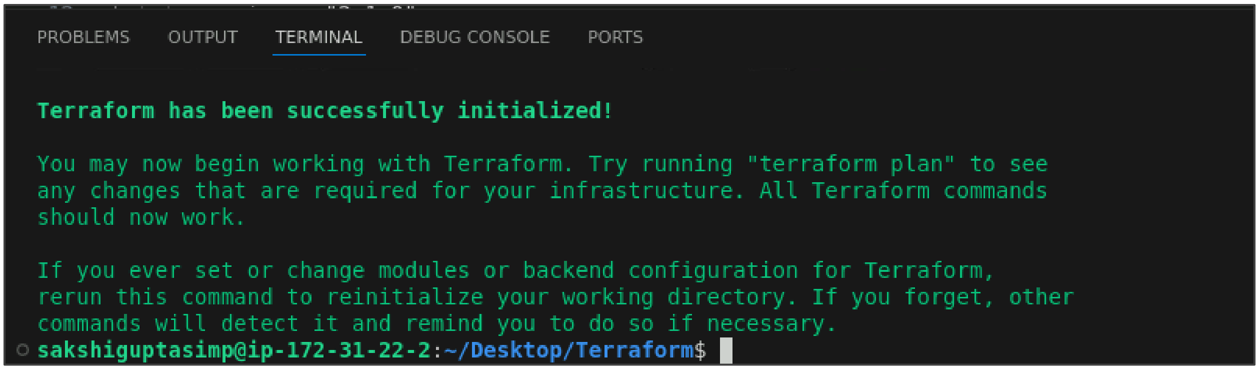
```
terraform {


  required_version = ">= 1.0.0"
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 3.0"
    }
    http = {
      source  = "hashicorp/http"
      version = "2.1.0"
    }
    random = {
```

4.2 Initialize the configuration to use the local backend using the following command:

terraform init -migrate-state

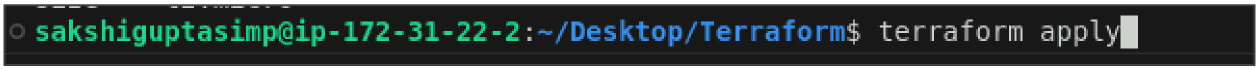


```
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform init -migrate-state
Initializing modules...

Initializing the backend...
Terraform has detected you're unconfiguring your previously set "remote" backend.
```



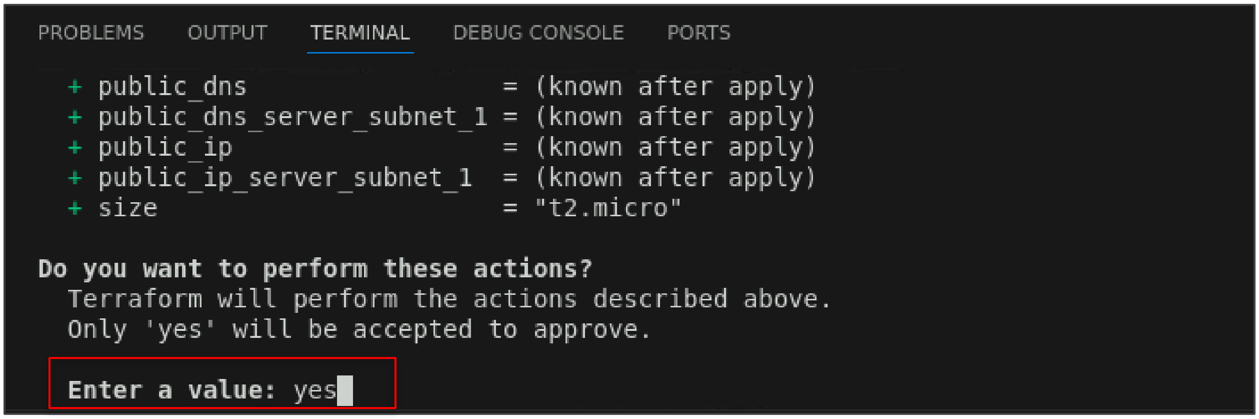
```
Terraform has been successfully initialized!

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.
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$
```

4.3 Apply the Terraform configuration using the following command:
terraform apply

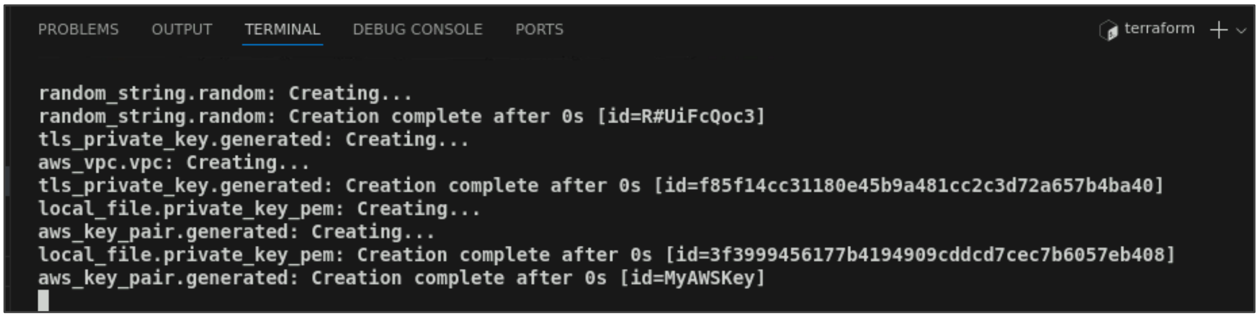


```
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform apply
```

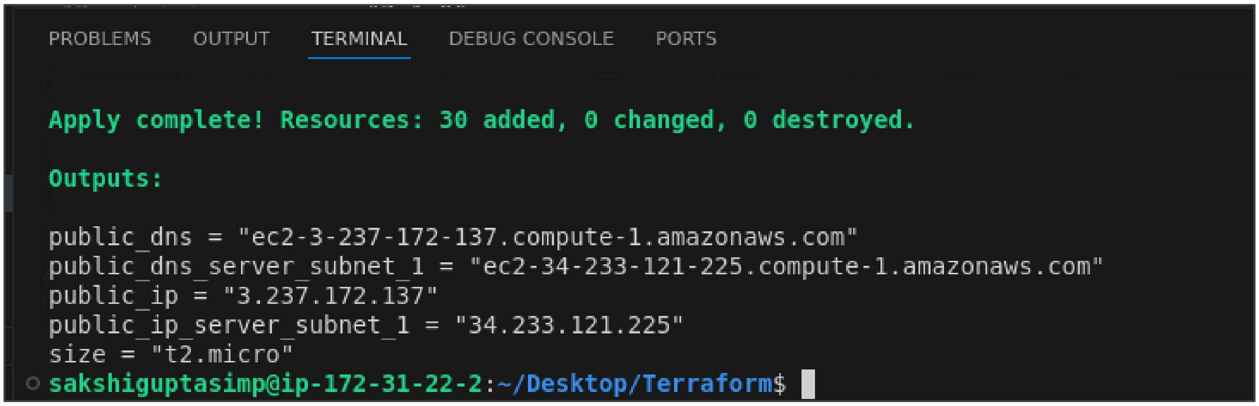
4.4 When prompted, approve the changes by typing **yes**



```
  + public_dns                  = (known after apply)
  + public_dns_server_subnet_1 = (known after apply)
  + public_ip                   = (known after apply)
  + public_ip_server_subnet_1  = (known after apply)
  + size                        = "t2.micro"

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
```

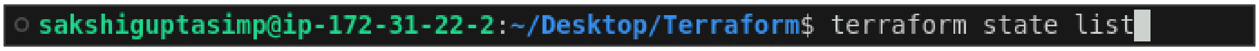


```
random_string.random: Creating...
random_string.random: Creation complete after 0s [id=R#UiFcQoc3]
tls_private_key.generated: Creating...
aws_vpc.vpc: Creating...
tls_private_key.generated: Creation complete after 0s [id=f85f14cc31180e45b9a481cc2c3d72a657b4ba40]
local_file.private_key_pem: Creating...
aws_key_pair.generated: Creating...
local_file.private_key_pem: Creation complete after 0s [id=3f3999456177b4194909cddcd7cec7b6057eb408]
aws_key_pair.generated: Creation complete after 0s [id=MyAWSKey]
```

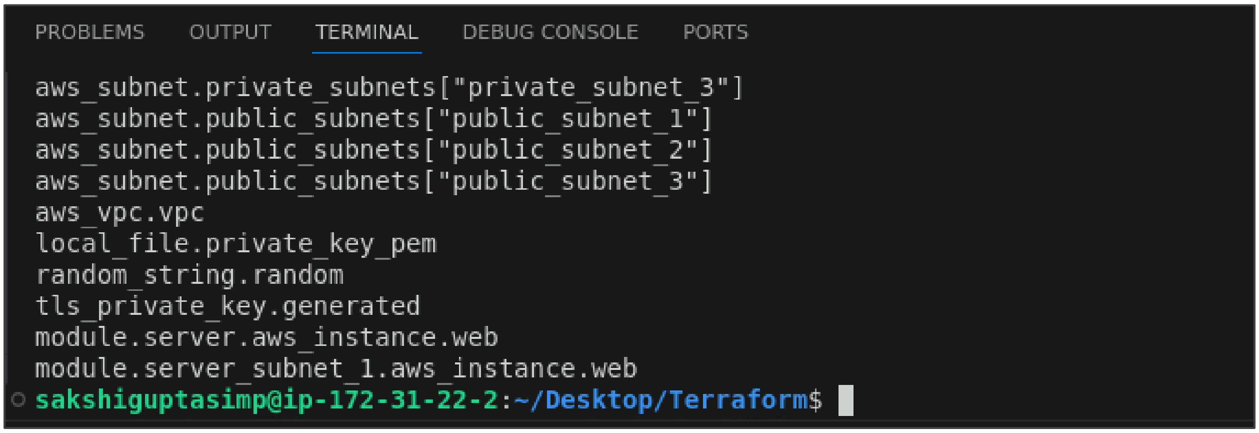


```
Apply complete! Resources: 30 added, 0 changed, 0 destroyed.

Outputs:

public_dns = "ec2-3-237-172-137.compute-1.amazonaws.com"
public_dns_server_subnet_1 = "ec2-34-233-121-225.compute-1.amazonaws.com"
public_ip = "3.237.172.137"
public_ip_server_subnet_1 = "34.233.121.225"
size = "t2.micro"
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$
```

4.5 List the newly created items in the state by using the following command:
terraform state list



```
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$ terraform state list
```



```
aws_subnet.private_subnets["private_subnet_3"]
aws_subnet.public_subnets["public_subnet_1"]
aws_subnet.public_subnets["public_subnet_2"]
aws_subnet.public_subnets["public_subnet_3"]
aws_vpc.vpc
local_file.private_key_pem
random_string.random
tls_private_key.generated
module.server.aws_instance.web
module.server_subnet_1.aws_instance.web
sakshiguptasimp@ip-172-31-22-2:~/Desktop/Terraform$
```

By following these steps, you have successfully migrated Terraform state between different backends, including the default local backend, AWS S3 backend, and Terraform Cloud remote backend.