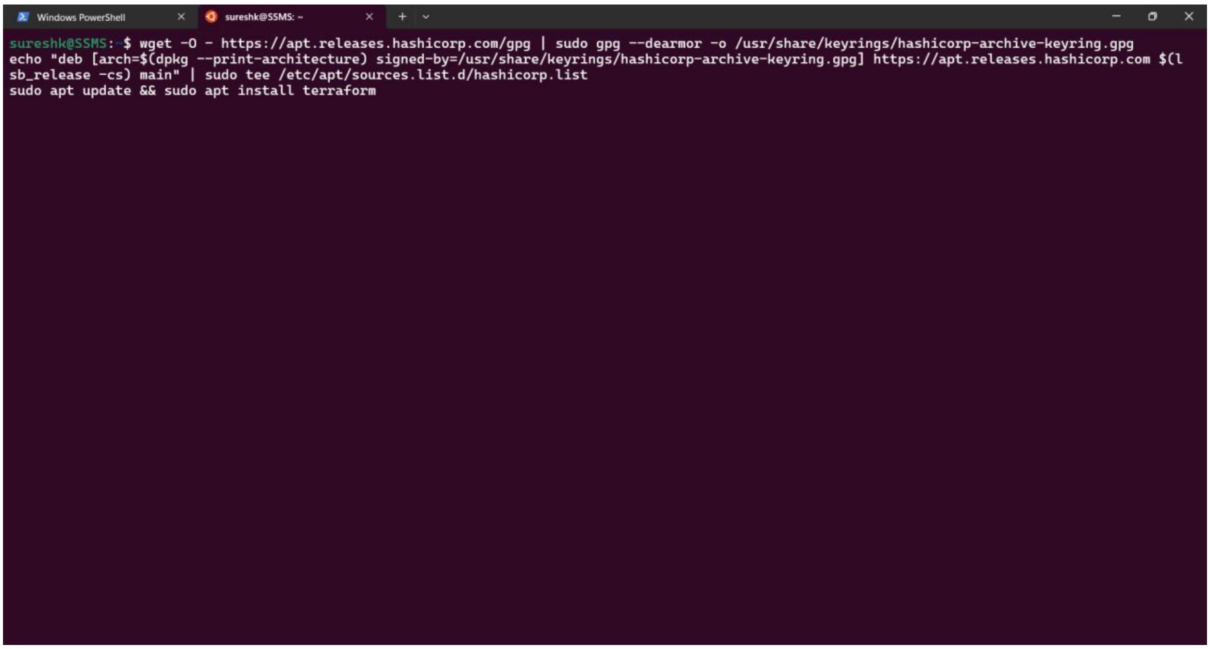
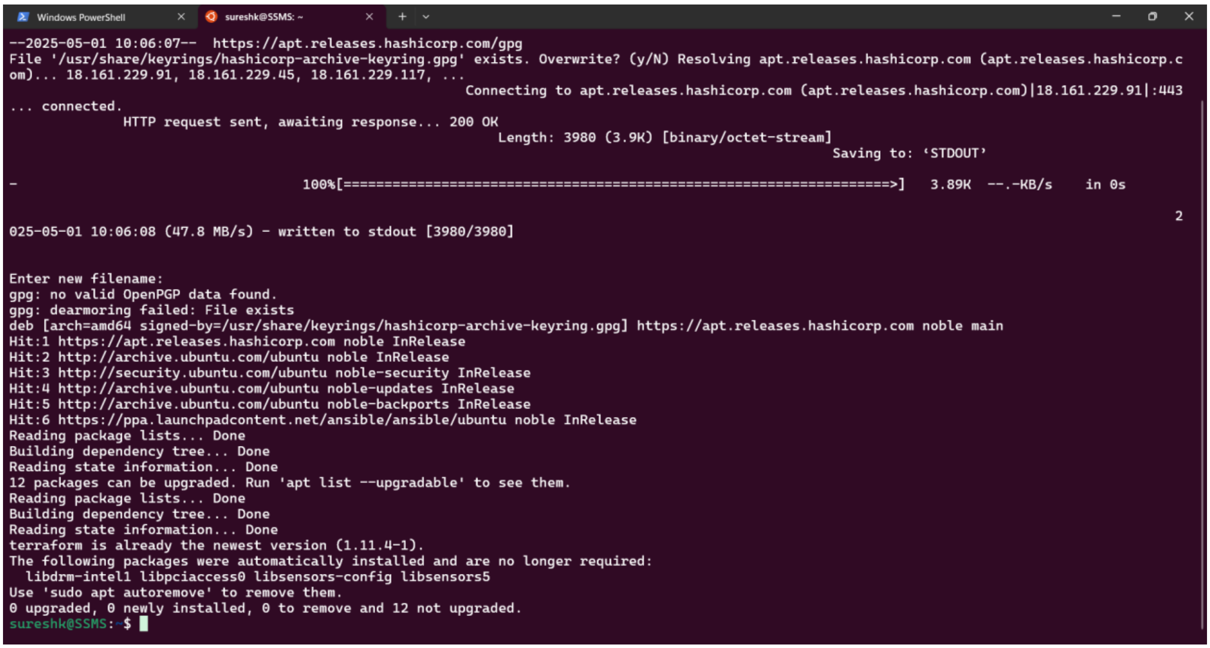
Terraform Task

Installing Terraform:

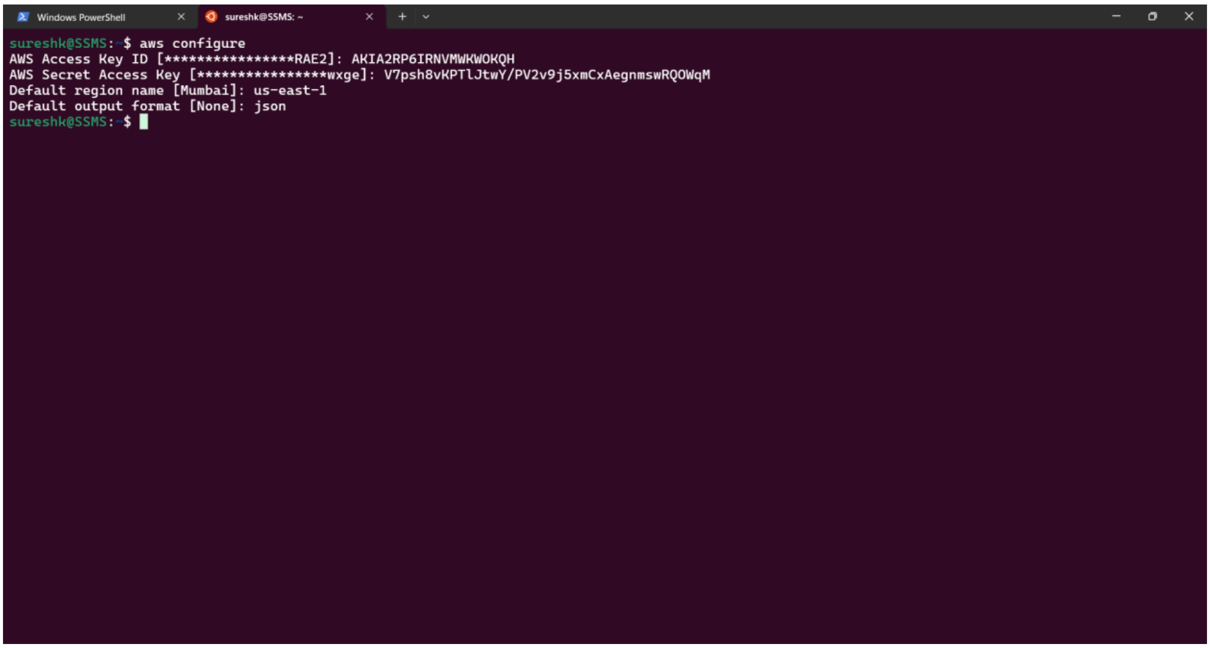


```
sureshk@SSMS:~$ wget -O - https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(l
sb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list
sudo apt update && sudo apt install terraform
```



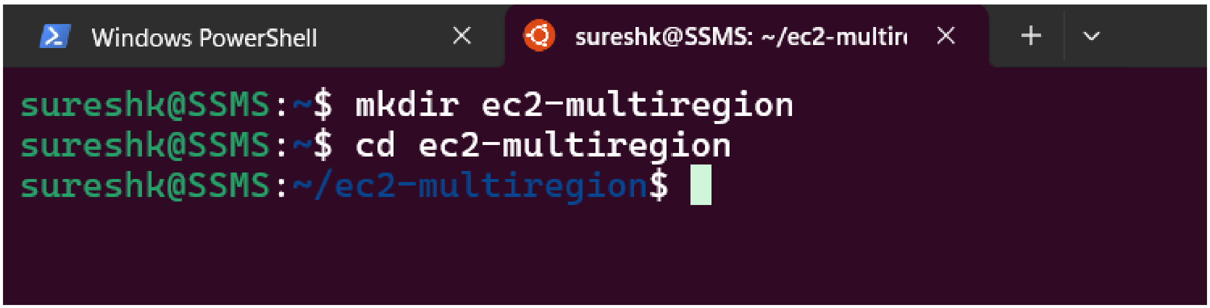
```
--2025-05-01 10:06:07--  https://apt.releases.hashicorp.com/gpg
File '/usr/share/keyrings/hashicorp-archive-keyring.gpg' exists. Overwrite? (y/N) Resolving apt.releases.hashicorp.com (apt.releases.hashicorp.c
om)... 18.161.229.91, 18.161.229.45, 18.161.229.117, ...
                                                        Connecting to apt.releases.hashicorp.com (apt.releases.hashicorp.com)|18.161.229.91|:443
... connected.
              HTTP request sent, awaiting response... 200 OK
                                                            Length: 3980 (3.9K) [binary/octet-stream]
                                                                                                     Saving to: ‘STDOUT’

-                                    100%[===================================================================>]   3.89K  --.-KB/s    in 0s

                                                                                                                                             2
025-05-01 10:06:08 (47.8 MB/s) - written to stdout [3980/3980]


Enter new filename:
gpg: no valid OpenPGP data found.
gpg: dearmoring failed: File exists
deb [arch=amd64 signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com noble main
Hit:1 https://apt.releases.hashicorp.com noble InRelease
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:3 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:4 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:6 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu noble InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
12 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
terraform is already the newest version (1.11.4-1).
The following packages were automatically installed and are no longer required:
  libdrm-intel1 libpciaccess0 libsensors-config libsensors5
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 12 not upgraded.
sureshk@SSMS:~$
```

Installing AWS CLI:



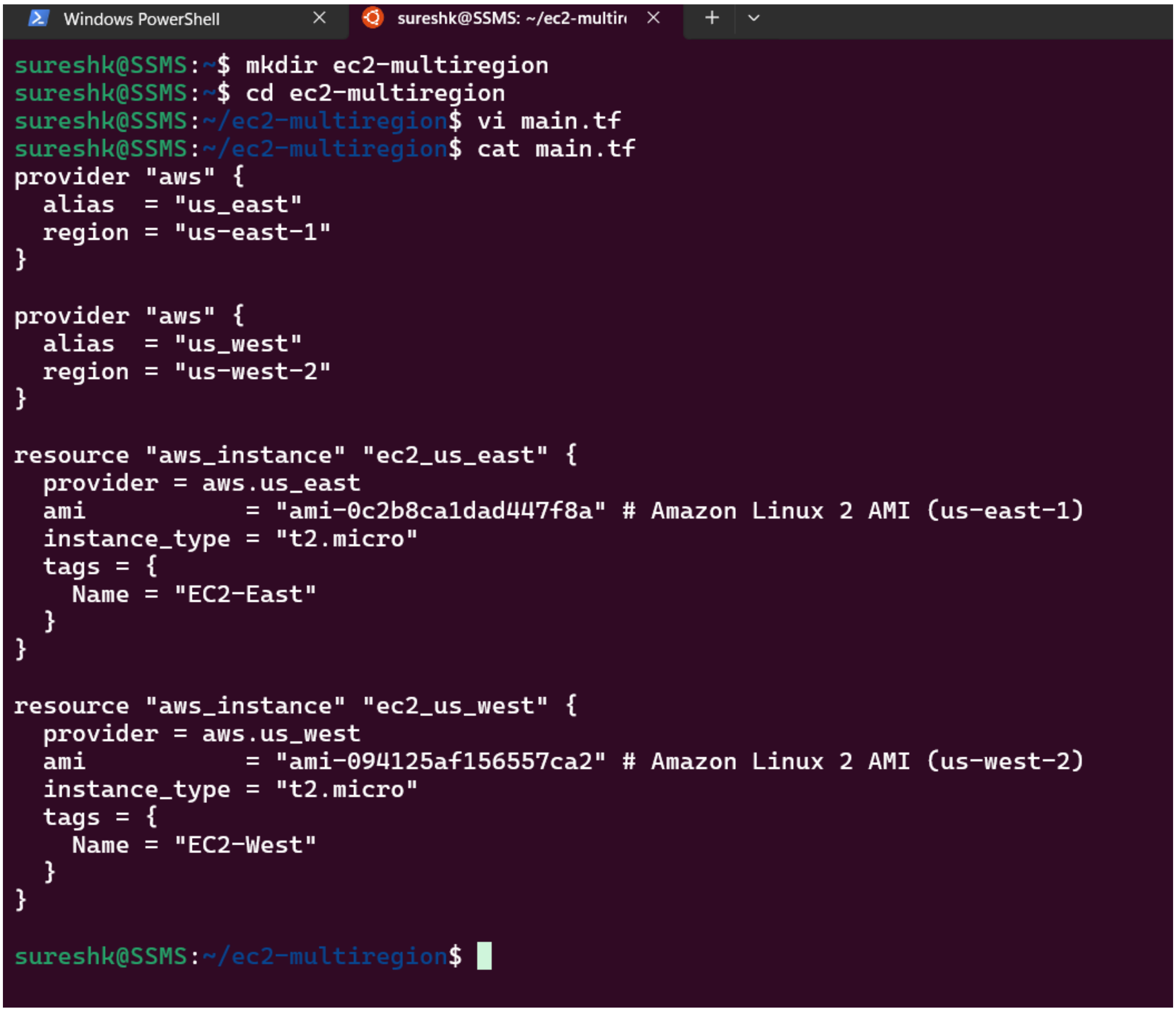
```
sureshk@SSMS:~$ aws configure
AWS Access Key ID [****************RAE2]: AKIA2RP6IRNVMWKWOKQH
AWS Secret Access Key [****************wxge]: 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
Default region name [Mumbai]: us-east-1
Default output format [None]: json
sureshk@SSMS:~$
```

Creating Directory for Terraform:



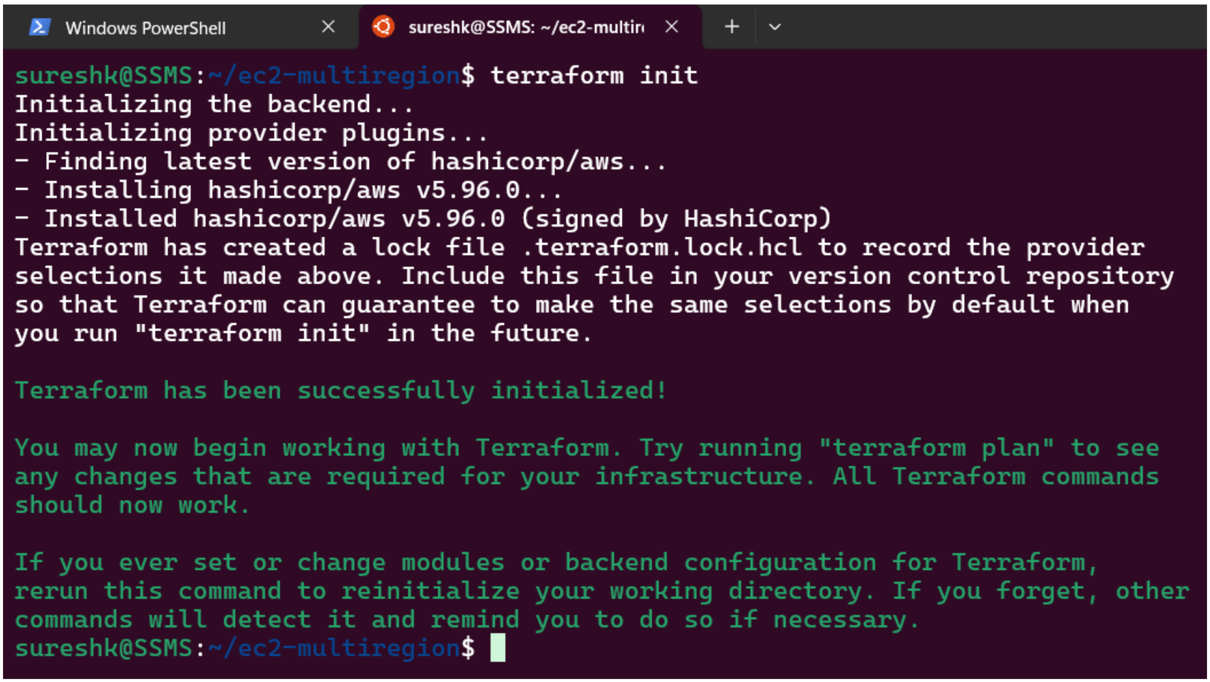
```
sureshk@SSMS:~$ mkdir ec2-multiregion
sureshk@SSMS:~$ cd ec2-multiregion
sureshk@SSMS:~/ec2-multiregion$
```

Creating Terraform file:



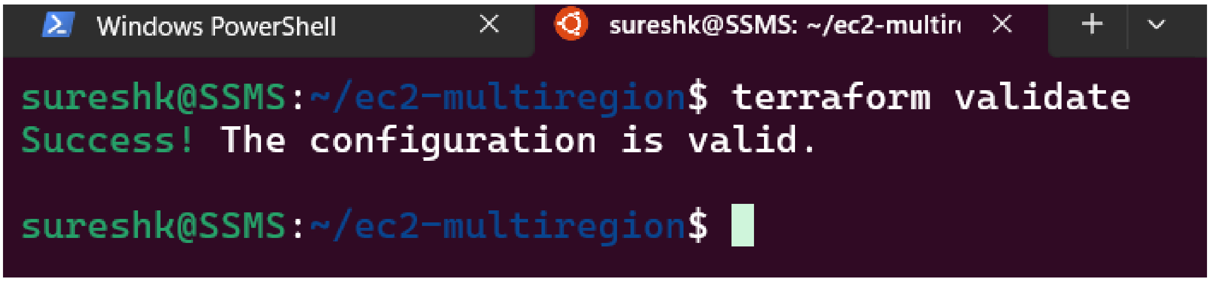
```
sureshk@SSMS:~$ mkdir ec2-multiregion
sureshk@SSMS:~$ cd ec2-multiregion
sureshk@SSMS:~/ec2-multiregion$ vi main.tf
sureshk@SSMS:~/ec2-multiregion$ cat main.tf
provider "aws" {
  alias  = "us_east"
  region = "us-east-1"
}

provider "aws" {
  alias  = "us_west"
  region = "us-west-2"
}

resource "aws_instance" "ec2_us_east" {
  provider = aws.us_east
  ami           = "ami-0c2b8ca1dad447f8a" # Amazon Linux 2 AMI (us-east-1)
  instance_type = "t2.micro"
  tags = {
    Name = "EC2-East"
  }
}

resource "aws_instance" "ec2_us_west" {
  provider = aws.us_west
  ami           = "ami-094125af156557ca2" # Amazon Linux 2 AMI (us-west-2)
  instance_type = "t2.micro"
  tags = {
    Name = "EC2-West"
  }
}

sureshk@SSMS:~/ec2-multiregion$
```

Initializing Terraform:



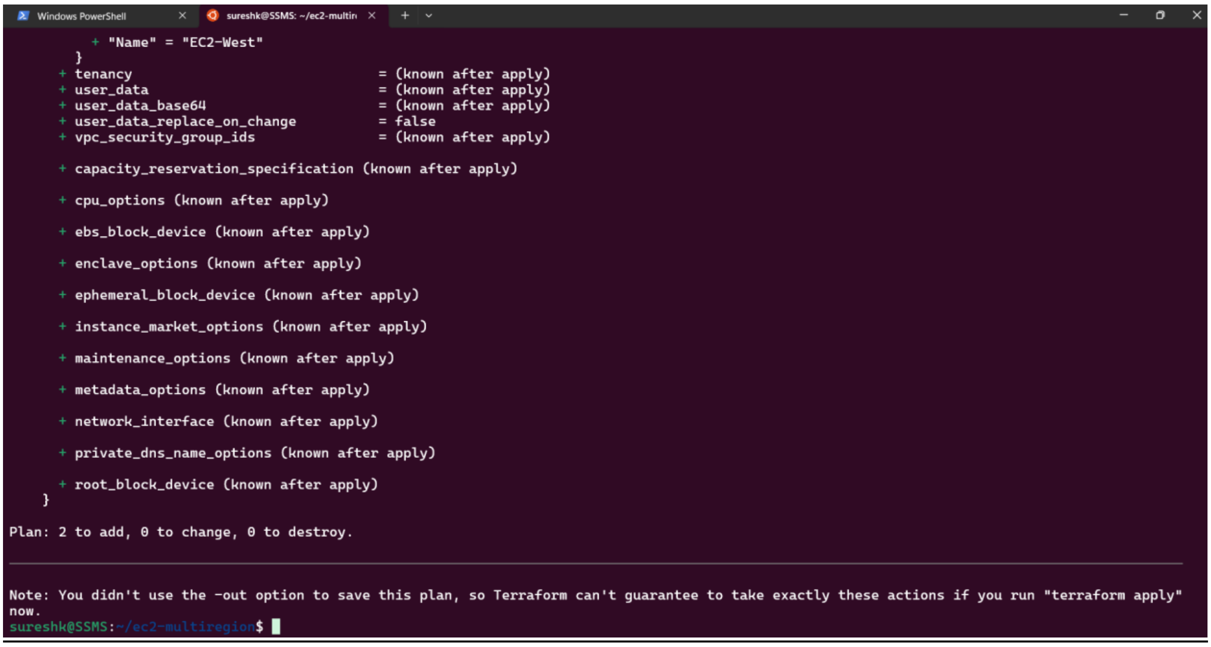
```
sureshk@SSMS:~/ec2-multiregion$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.96.0...
- Installed hashicorp/aws v5.96.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl 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!

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.
sureshk@SSMS:~/ec2-multiregion$
```

Validating the Configuration:



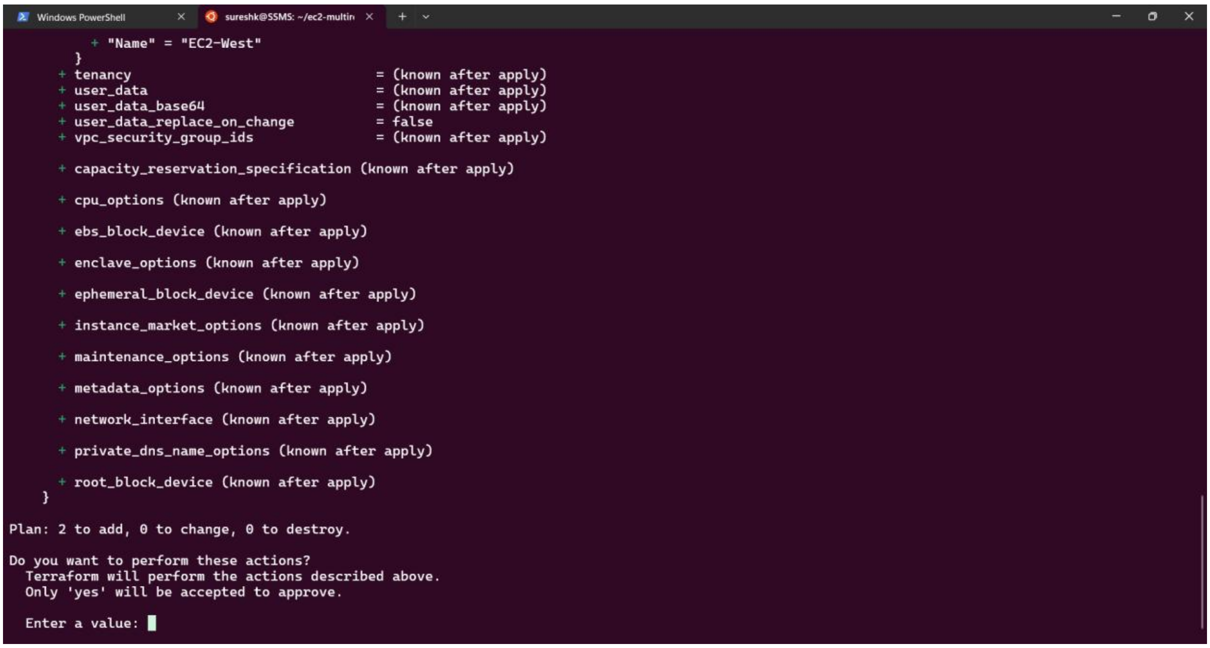
```
sureshk@SSMS:~/ec2-multiregion$ terraform validate
Success! The configuration is valid.

sureshk@SSMS:~/ec2-multiregion$
```

Terraform Plan:

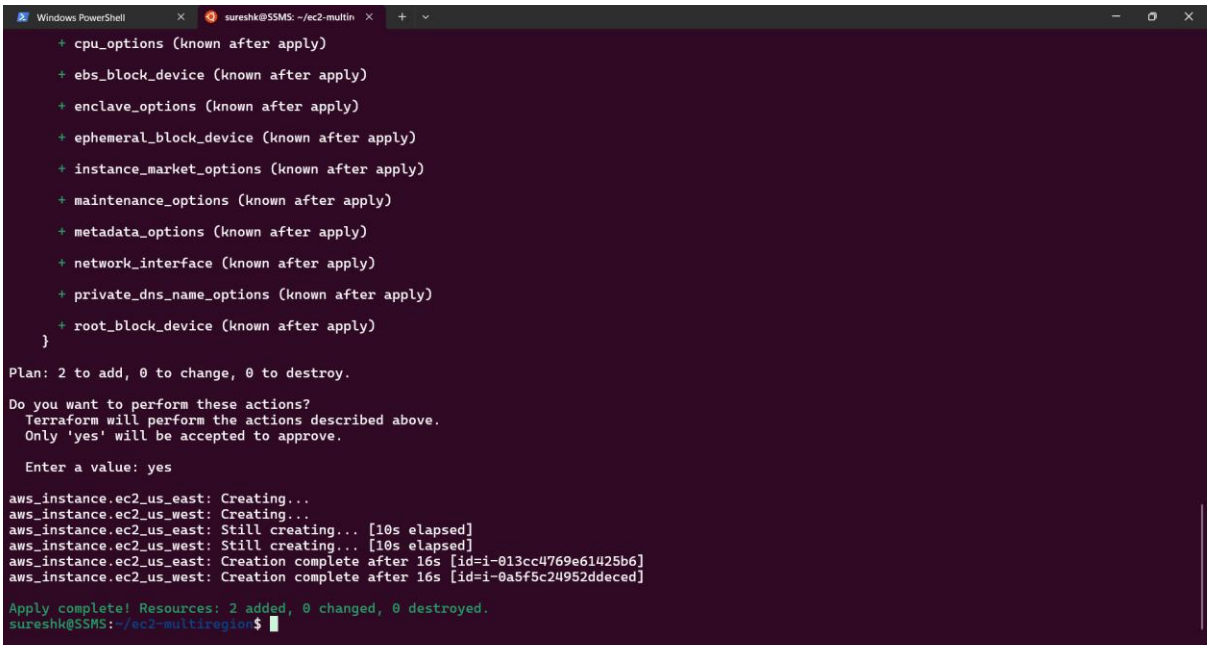


```
          + "Name" = "EC2-West"
        }
      + tenancy                              = (known after apply)
      + user_data                            = (known after apply)
      + user_data_base64                     = (known after apply)
      + user_data_replace_on_change          = false
      + vpc_security_group_ids               = (known after apply)

      + capacity_reservation_specification (known after apply)

      + cpu_options (known after apply)

      + ebs_block_device (known after apply)

      + enclave_options (known after apply)

      + ephemeral_block_device (known after apply)

      + instance_market_options (known after apply)

      + maintenance_options (known after apply)

      + metadata_options (known after apply)

      + network_interface (known after apply)

      + private_dns_name_options (known after apply)

      + root_block_device (known after apply)
    }

Plan: 2 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.
sureshk@SSMS:~/ec2-multiregion$
```

Terraform Apply:

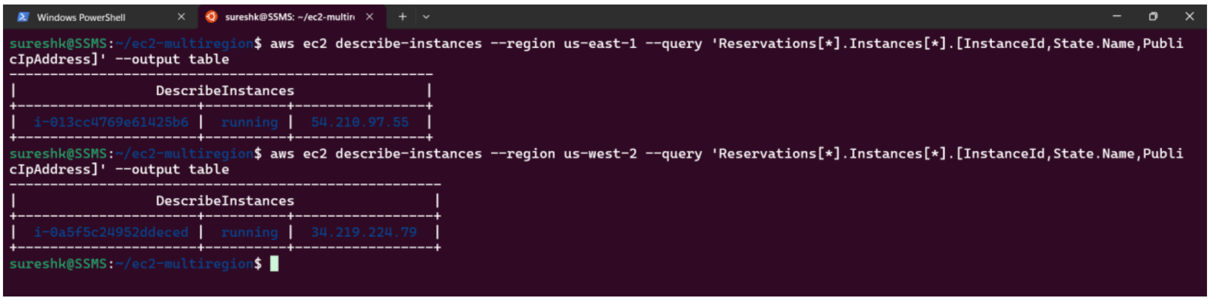


```
          + "Name" = "EC2-West"
        }
      + tenancy                              = (known after apply)
      + user_data                            = (known after apply)
      + user_data_base64                     = (known after apply)
      + user_data_replace_on_change          = false
      + vpc_security_group_ids               = (known after apply)

      + capacity_reservation_specification (known after apply)

      + cpu_options (known after apply)

      + ebs_block_device (known after apply)

      + enclave_options (known after apply)

      + ephemeral_block_device (known after apply)

      + instance_market_options (known after apply)

      + maintenance_options (known after apply)

      + metadata_options (known after apply)

      + network_interface (known after apply)

      + private_dns_name_options (known after apply)

      + root_block_device (known after apply)
    }

Plan: 2 to add, 0 to change, 0 to destroy.

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

  Enter a value: 
```



```
      + cpu_options (known after apply)

      + ebs_block_device (known after apply)

      + enclave_options (known after apply)

      + ephemeral_block_device (known after apply)

      + instance_market_options (known after apply)

      + maintenance_options (known after apply)

      + metadata_options (known after apply)

      + network_interface (known after apply)

      + private_dns_name_options (known after apply)

      + root_block_device (known after apply)
    }

Plan: 2 to add, 0 to change, 0 to destroy.

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

aws_instance.ec2_us_east: Creating...
aws_instance.ec2_us_west: Creating...
aws_instance.ec2_us_east: Still creating... [10s elapsed]
aws_instance.ec2_us_west: Still creating... [10s elapsed]
aws_instance.ec2_us_east: Creation complete after 16s [id=i-013cc4769e61425b6]
aws_instance.ec2_us_west: Creation complete after 16s [id=i-0a5f5c24952ddeced]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
sureshk@SSMS:~/ec2-multiregion$ 
```

EC2 Instance Output:



```
sureshk@SSMS:~/ec2-multiregion$ aws ec2 describe-instances --region us-east-1 --query 'Reservations[*].Instances[*].[InstanceId,State.Name,Publi
cIpAddress]' --output table
-----------------------------------------------------
|                 DescribeInstances                 |
+----------------------+----------+-----------------+
|  i-013cc4769e61425b6 |  running |  54.210.97.55   |
+----------------------+----------+-----------------+
sureshk@SSMS:~/ec2-multiregion$ aws ec2 describe-instances --region us-west-2 --query 'Reservations[*].Instances[*].[InstanceId,State.Name,Publi
cIpAddress]' --output table
------------------------------------------------------
|                 DescribeInstances                  |
+----------------------+----------+------------------+
|  i-0a5f5c24952ddeced |  running |  34.219.224.79   |
+----------------------+----------+------------------+
sureshk@SSMS:~/ec2-multiregion$ 
```