Lesson 09 Demo 02

Running Multiple Cloud Provider Configurations

Objective: To demonstrate the use of multiple AWS provider configurations in Terraform for managing resources across different regions, showcasing the ability to define, configure, and deploy resources in various regions within a single Terraform configuration

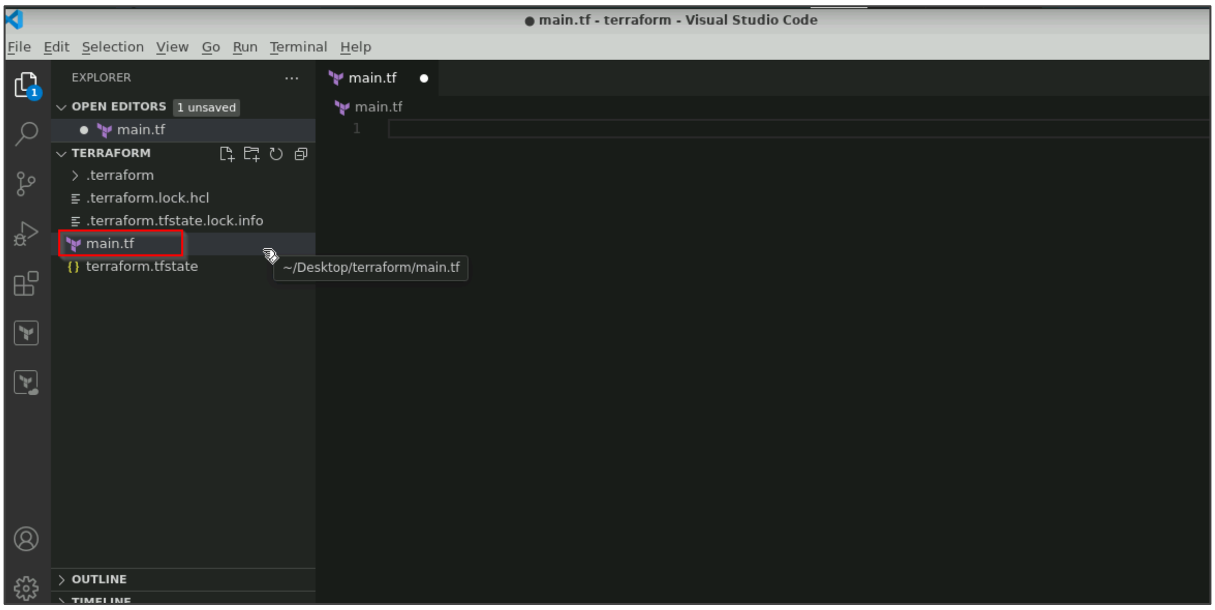
Tools required: VS Code and Linux terminal

Prerequisites: None

Steps to be followed:
1. Define providers with aliases in the **main.tf** file
2. Specify provider configurations for each resource in the **main.tf** file
3. Initialize and apply the configuration

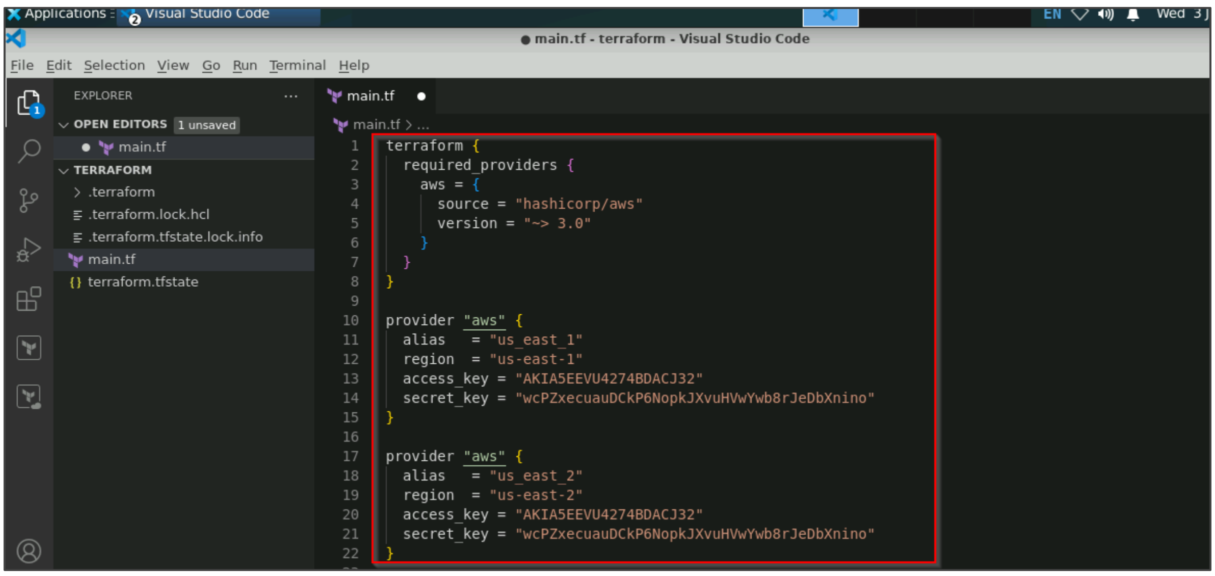
Step 1: Define providers with aliases in the main.tf file

1.1 Open the **main.tf** file in the **Terraform** folder


● main.tf - terraform - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS 1 unsaved
main.tf
TERRAFORM
.terraform
.terraform.lock.hcl
.terraform.tfstate.lock.info
main.tf
terraform.tfstate
~/Desktop/terraform/main.tf
OUTLINE
TIMELINE

1.2 Enter the code given below in the **main.tf** file to define multiple AWS providers with different aliases and save the file:

```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "~> 3.0"
    }
  }
}

provider "aws" {
  alias    = "us_east_1"
  region   = "us-east-1"
  access_key = "AKIA5EEVU4274BDACJ32"
  secret_key = "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"
}

provider "aws" {
  alias    = "us_east_2"
  region   = "us-east-2"
  access_key = "AKIA5EEVU4274BDACJ32"
  secret_key = "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"
}

provider "aws" {
  alias    = "eu_central_1"
  region   = "eu-central-1"
  access_key = "AKIA5EEVU4274BDACJ32"
  secret_key = "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"
}
```

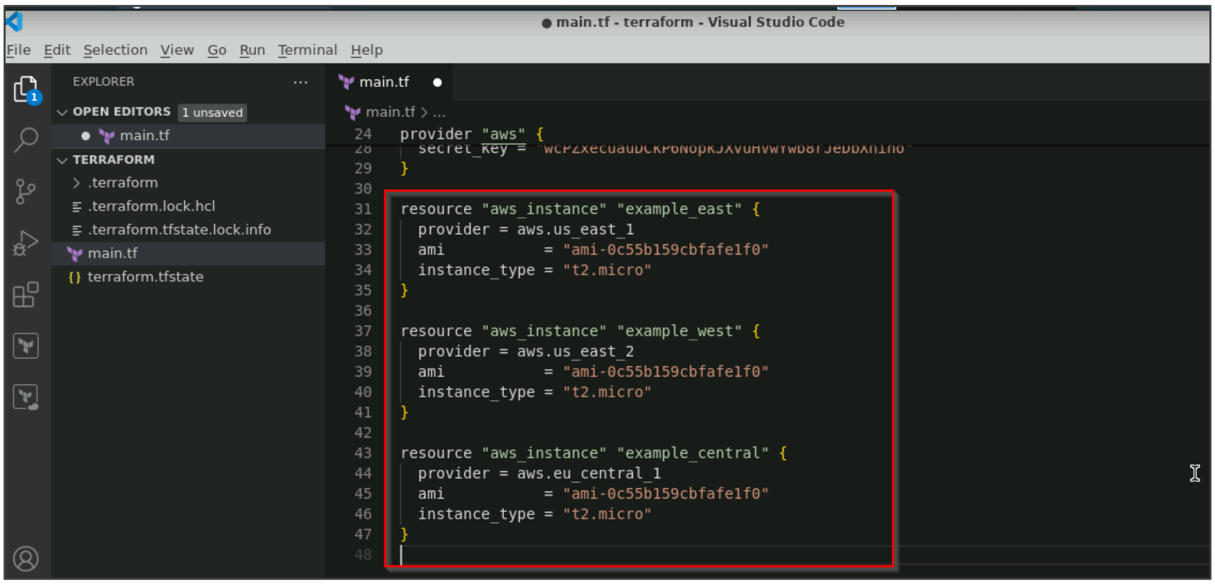


```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "~> 3.0"
    }
  }
}

provider "aws" {
  alias    = "us_east_1"
  region   = "us-east-1"
  access_key = "AKIA5EEVU4274BDACJ32"
  secret_key = "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"
}

provider "aws" {
  alias    = "us_east_2"
  region   = "us-east-2"
  access_key = "AKIA5EEVU4274BDACJ32"
  secret_key = "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"
}
```

Step 2: Specify provider configurations for each resource in the main.tf file

2.1 Update the **main.tf** file with the provider configurations for each resource as given below and save the file:

```
resource "aws_instance" "example_east" {
  provider = aws.us_east_1
  ami           = "ami-0c55b159cbfafe1f0"
  instance_type = "t2.micro"
}

resource "aws_instance" "example_west" {
  provider = aws.us_east_2
  ami           = "ami-0c55b159cbfafe1f0"
  instance_type = "t2.micro"
}

resource "aws_instance" "example_central" {
  provider = aws.eu_central_1
  ami           = "ami-0c55b159cbfafe1f0"
  instance_type = "t2.micro"
}
```

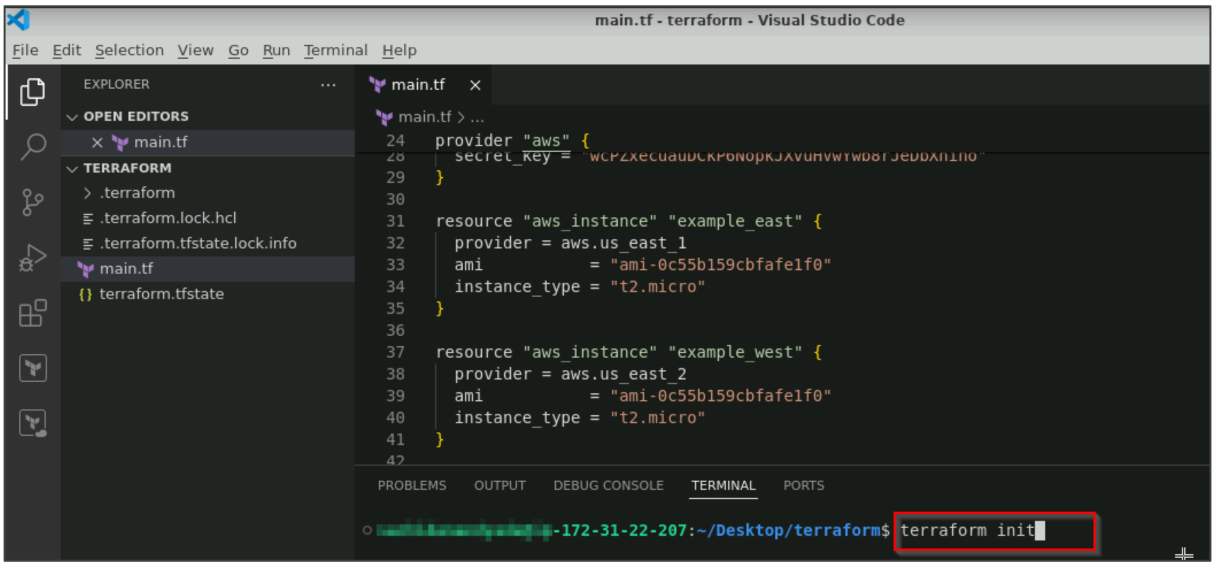


```
provider "aws" {
  secret_key = "wcPZxecuaUDCKP6NopKJXVuHVWYWb8rJeDbXn1no"
}

resource "aws_instance" "example_east" {
  provider = aws.us_east_1
  ami           = "ami-0c55b159cbfafe1f0"
  instance_type = "t2.micro"
}

resource "aws_instance" "example_west" {
  provider = aws.us_east_2
  ami           = "ami-0c55b159cbfafe1f0"
  instance_type = "t2.micro"
}

resource "aws_instance" "example_central" {
  provider = aws.eu_central_1
  ami           = "ami-0c55b159cbfafe1f0"
  instance_type = "t2.micro"
}
```

Step 3: Initialize and apply the configuration

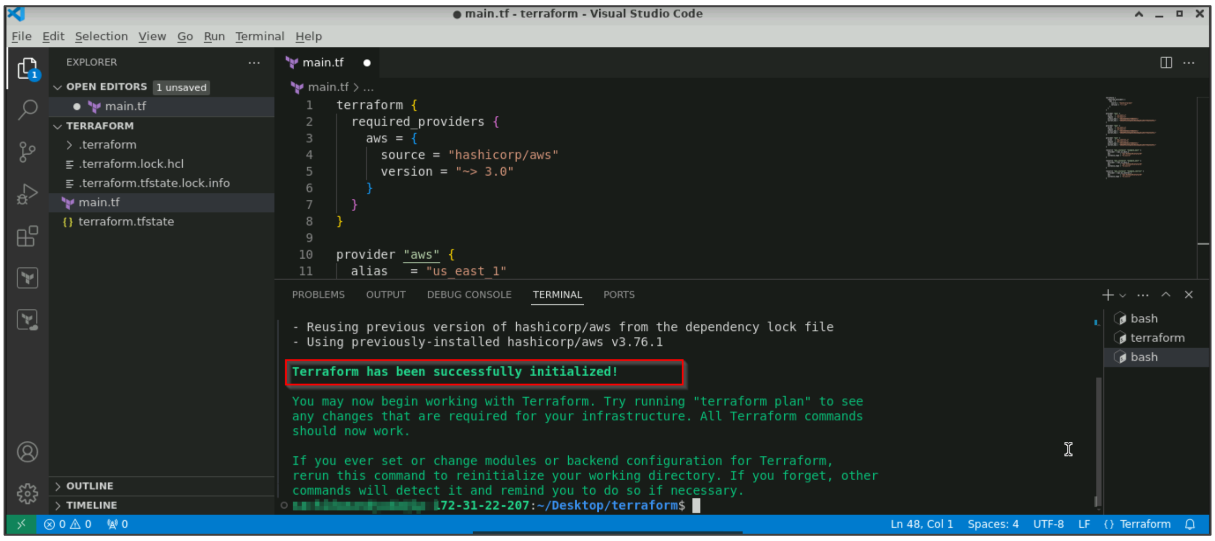
3.1 Open the terminal and run the command given below to initialize the configurations:

terraform init


main.tf - terraform - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS
main.tf
TERRAFORM
.terraform
.terraform.lock.hcl
.terraform.tfstate.lock.info
main.tf
terraform.tfstate
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
provider "aws" {
  secret_key = "wCPZxecuaUDCKP6NopKJXvuHVWYWb8rJeDbXh1no"
}

resource "aws_instance" "example_east" {
  provider = aws.us_east_1
  ami           = "ami-0c55b159cbfafe1f0"
  instance_type = "t2.micro"
}

resource "aws_instance" "example_west" {
  provider = aws.us_east_2
  ami           = "ami-0c55b159cbfafe1f0"
  instance_type = "t2.micro"
}
```

```
[redacted]-172-31-22-207:~/Desktop/terraform$ terraform init
```


main.tf - terraform - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS 1 unsaved
main.tf
TERRAFORM
.terraform
.terraform.lock.hcl
.terraform.tfstate.lock.info
main.tf
terraform.tfstate
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash
terraform
bash
Ln 48, Col 1 Spaces: 4 UTF-8 LF {} Terraform

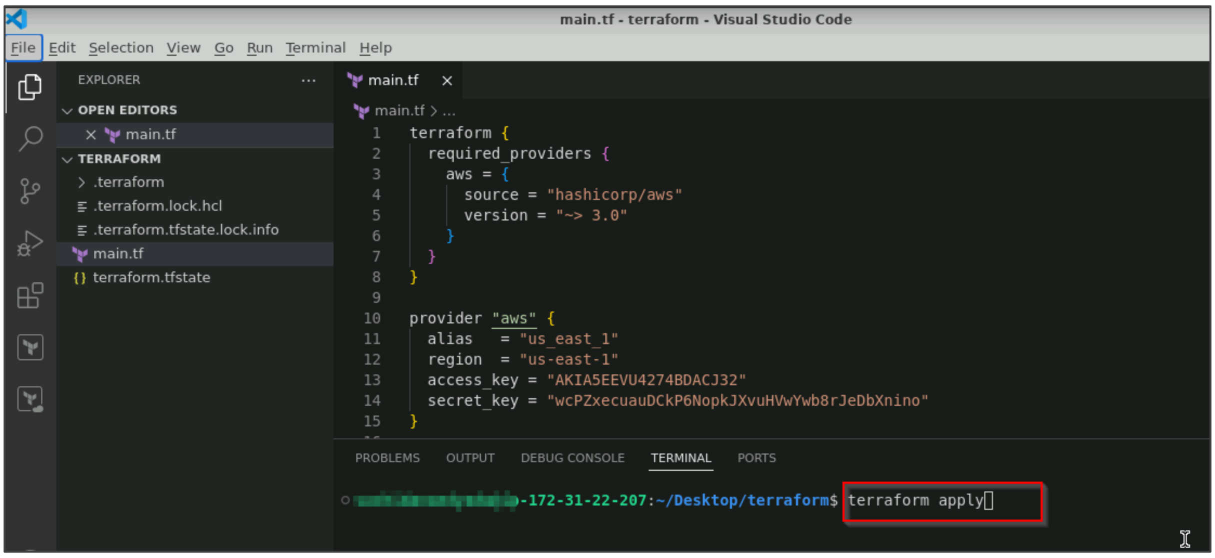
```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "~> 3.0"
    }
  }
}

provider "aws" {
  alias   = "us_east_1"
```

```
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v3.76.1

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.
[redacted]172-31-22-207:~/Desktop/terraform$
```

The Terraform configuration is successfully initialized.

3.2 Open the terminal and run the command given below to apply the configurations:

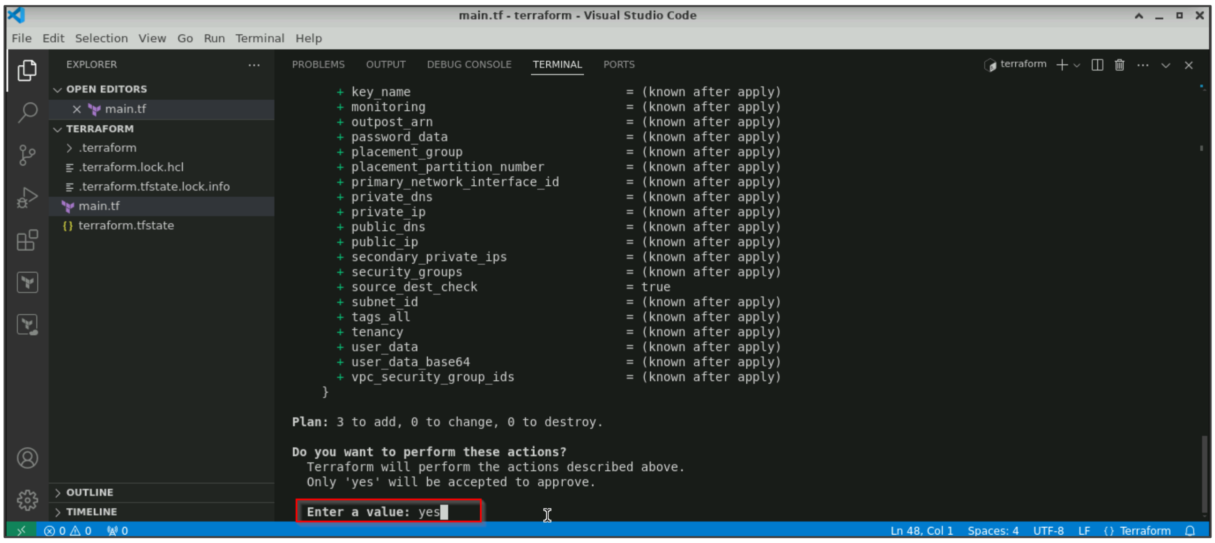
terraform apply


main.tf - terraform - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS
main.tf
TERRAFORM
.terraform
.terraform.lock.hcl
.terraform.tfstate.lock.info
main.tf
terraform.tfstate
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

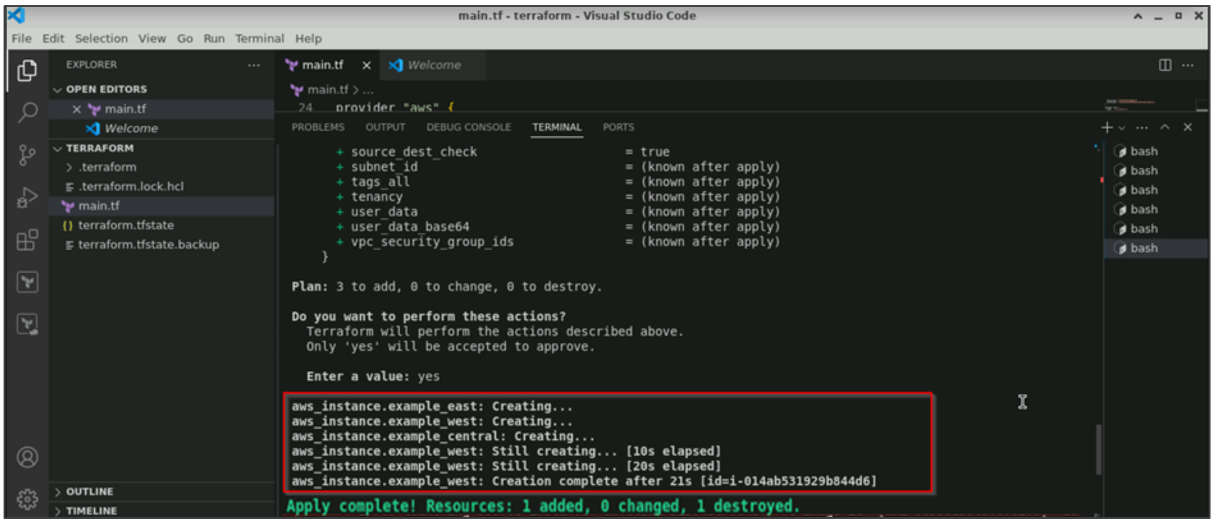
```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "~> 3.0"
    }
  }
}

provider "aws" {
  alias    = "us_east_1"
  region   = "us-east-1"
  access_key = "AKIA5EEVU4274BDACJ32"
  secret_key = "wcPZxecuauDCkP6NopkJXvuHVwYwb8rJeDbXnino"
}
```

```
[redacted]-172-31-22-207:~/Desktop/terraform$ terraform apply
```

3.3 Enter **yes** to apply the configuration


main.tf - terraform - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS
main.tf
TERRAFORM
.terraform
.terraform.lock.hcl
.terraform.tfstate.lock.info
main.tf
terraform.tfstate
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
terraform
Ln 48, Col 1 Spaces: 4 UTF-8 LF Terraform

```
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
      + placement_partition_number           = (known after apply)
      + primary_network_interface_id         = (known after apply)
      + private_dns                          = (known after apply)
      + private_ip                           = (known after apply)
      + public_dns                           = (known after apply)
      + public_ip                            = (known after apply)
      + secondary_private_ips                = (known after apply)
      + security_groups                      = (known after apply)
      + source_dest_check                    = true
      + subnet_id                            = (known after apply)
      + tags_all                             = (known after apply)
      + tenancy                              = (known after apply)
      + user_data                            = (known after apply)
      + user_data_base64                     = (known after apply)
      + vpc_security_group_ids               = (known after apply)
    }

Plan: 3 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
```



```
main.tf > ...
 24   provider "aws" {
```

```
      + source_dest_check                      = true
      + subnet_id                              = (known after apply)
      + tags_all                               = (known after apply)
      + tenancy                                = (known after apply)
      + user_data                              = (known after apply)
      + user_data_base64                       = (known after apply)
      + vpc_security_group_ids                 = (known after apply)
    }

Plan: 3 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.example_east: Creating...
aws_instance.example_west: Creating...
aws_instance.example_central: Creating...
aws_instance.example_west: Still creating... [10s elapsed]
aws_instance.example_west: Still creating... [20s elapsed]
aws_instance.example_west: Creation complete after 21s [id=i-014ab531929b844d6]
Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```

By following these steps, you have successfully demonstrated the use of multiple AWS provider configurations in Terraform for managing resources across different regions, showcasing the ability to define, configure, and deploy resources in various regions within a single Terraform configuration.