Lab Exercise 9– Creating Multiple EC2 Instances with for_each in Terraform Objective:

Learn how to use for each in Terraform to create multiple AWS EC2 instances with specific settings for each instance.

Prerequisites:

- Terraform installed on your machine.
- AWS CLI configured with the necessary credentials.

Steps:

1. Create a Terraform Directory:

 # main.tf

```
main.tf > terraform
1   terraform {
2     required_providers {
3       aws = {
4         source = "hashicorp/aws"
5         version = "5.31.0"
6       }
7     }
8   }
9
10  provider "aws" {
11      region = "ap-south-1"
12      access_key = "AKIAZI2LIAJGSHGMMMHP"
13      secret_key = "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"
14  }
```

instance.tf > resource "aws_instance" "ec2_instances" > for_each

```
variable "instances" {
  default = {
    "instance1" = {
    },
    "instance2" = {
      ami           = "ami-0123456789abcdef0"
      instance_type = "t2. micro "
    },
    "instance3" = {
      ami           = "ami-9876543210fedcba0"
      instance_type = "t2. micro "
    }
  }
}

resource "aws_instance" "ec2_instances" {
  for_each = var.instances
  ami = var.instances[each.key].ami
  instance_type = var.instances[each.key].instance_type
  tags = {
    Name = "EC2-Instance-${each.key}"
  }
}
```

2. Initialize and Plan:

```
PS E:\Desktop\DevOps\SPCM9> terraform init

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!

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.
```

```
PS E:\Desktop\DevOps\SPCM9> terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following s
  + create

Terraform will perform the following actions:

  # aws_instance.ec2_instances["instance1"] will be created
  + resource "aws_instance" "ec2_instances" {
      + ami                                  = "ami-0c55b159cbfafe1f0"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t2.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = (known after apply)
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
```

```
      + tags                                 = {
          + "Name" = "EC2-Instance-instance1"
        }
      + tags_all                             = {
          + "Name" = "EC2-Instance-instance1"
        }
      + 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)
    }

  # aws_instance.ec2_instances["instance2"] will be created
  + resource "aws_instance" "ec2_instances" {
      + ami                                  = "ami-0123456789abcdef0"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
      + instance_state                       = (known after apply)
```