System Provisioning and Configuration Management LAB

SUBMITTED TO

Dr. Hitesh Kumar Sharma

SUBMITTED BY

Siddharth Agarwal

500107594

R2142220663

Btech CSE DevOps B1

Lab Exercise 11– Creating a VPC in Terraform Objective:

Objective:

Learn how to use Terraform to create a basic Virtual Private Cloud (VPC) in AWS.

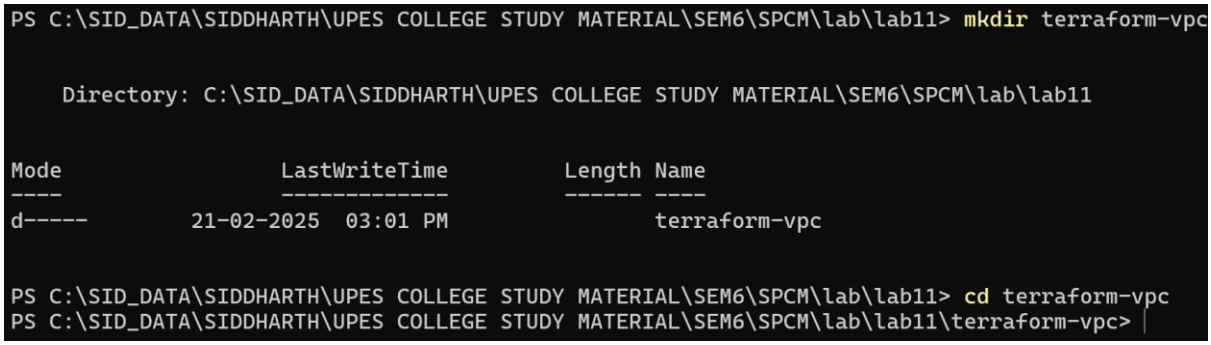
Prerequisites:

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

Steps:

1. Create a Terraform Directory:

```
mkdir terraform-vpc
cd terraform-vpc
```

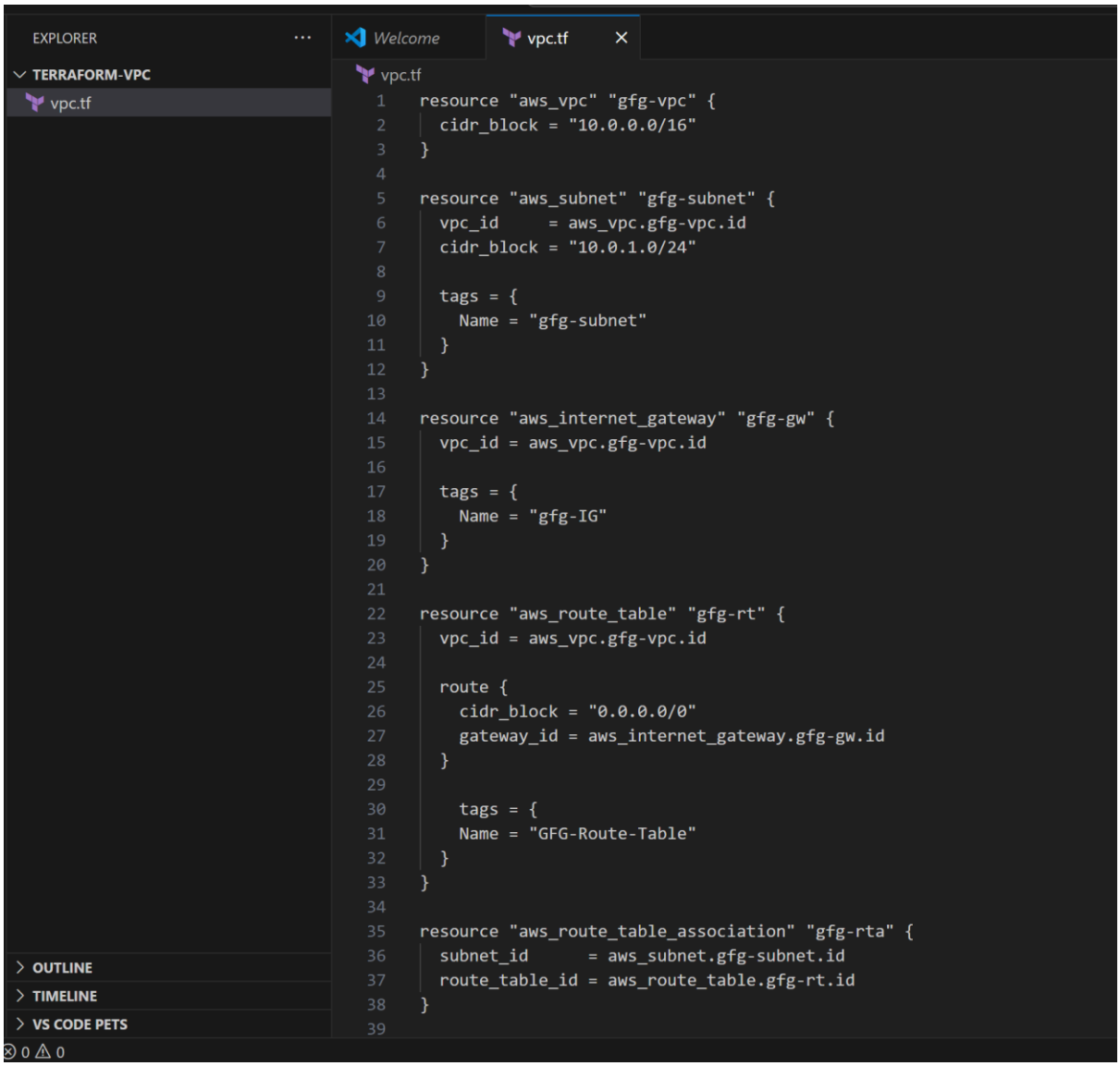


- Create Terraform Configuration Files:
- Create a file named main.tf:

 # vpc.tf

```
resource "aws_vpc" "gfg-vpc" {
  cidr_block = "10.0.0.0/16"
}

resource "aws_subnet" "gfg-subnet" {
  vpc_id     = aws_vpc.gfg-vpc.id
  cidr_block = "10.0.1.0/24"

  tags = {
    Name = "gfg-subnet"
  }
}

resource "aws_internet_gateway" "gfg-gw" {
  vpc_id = aws_vpc.gfg-vpc.id

  tags = {
    Name = "gfg-IG"
  }
}

resource "aws_route_table" "gfg-rt" {
  vpc_id = aws_vpc.gfg-vpc.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.gfg-gw.id
  }

    tags = {
    Name = "GFG-Route-Table"
  }
```

```
}

resource "aws_route_table_association" "gfg-rta" {
  subnet_id      = aws_subnet.gfg-subnet.id
  route_table_id = aws_route_table.gfg-rt.id
}

resource "aws_security_group" "gfg-sg" {
  name        = "my-gfg-sg"
  vpc_id      = aws_vpc.gfg-vpc.id

  ingress {
    description      = "TLS from VPC"
    from_port        = 20
    to_port          = 20
    protocol         = "tcp"
    cidr_blocks      = ["0.0.0.0/0"]
    ipv6_cidr_blocks = ["::/0"]
  }

  egress {
    from_port        = 0
    to_port          = 0
    protocol         = "-1"
    cidr_blocks      = ["0.0.0.0/0"]
    ipv6_cidr_blocks = ["::/0"]
  }

  tags = {
    Name = "my-gfg-sg"
  }
}
```



In this configuration, we define an AWS provider, a VPC with a specified CIDR block, and two subnets within the VPC.

2. Initialize and Apply:

- Run the following Terraform commands to initialize and apply the configuration:

terraform init

terraform apply

```
PS C:\SID_DATA\SIDDHARTH\UPES COLLEGE STUDY MATERIAL\SEM6\SPCM\lab\lab11\terraform-vpc> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.88.0...
- Installed hashicorp/aws v5.88.0 (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.
```

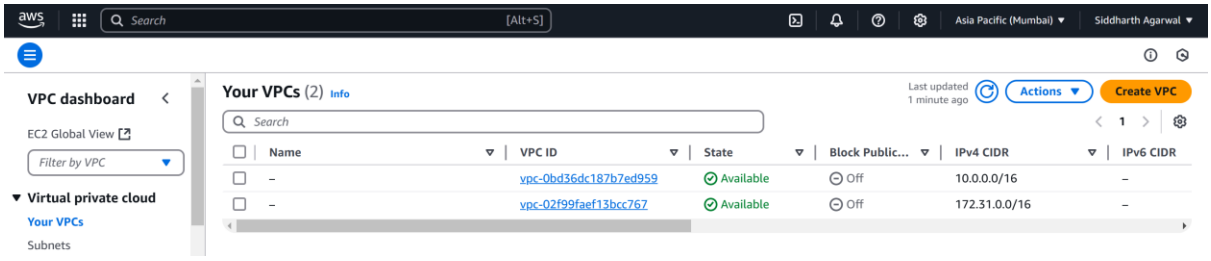
```
PS C:\SID_DATA\SIDDHARTH\UPES COLLEGE STUDY MATERIAL\SEM6\SPCM\lab\lab11\terraform-vpc> terraform apply -auto-approve

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_internet_gateway.gfg-gw will be created
  + resource "aws_internet_gateway" "gfg-gw" {
      + arn      = (known after apply)
      + id       = (known after apply)
      + owner_id = (known after apply)
      + tags     = {
          + "Name" = "gfg-IG"
        }
      + tags_all = {
          + "Name" = "gfg-IG"
        }
      + vpc_id   = (known after apply)
    }

  # aws_route_table.gfg-rt will be created
  + resource "aws_route_table" "gfg-rt" {
      + arn              = (known after apply)
      + id               = (known after apply)
```

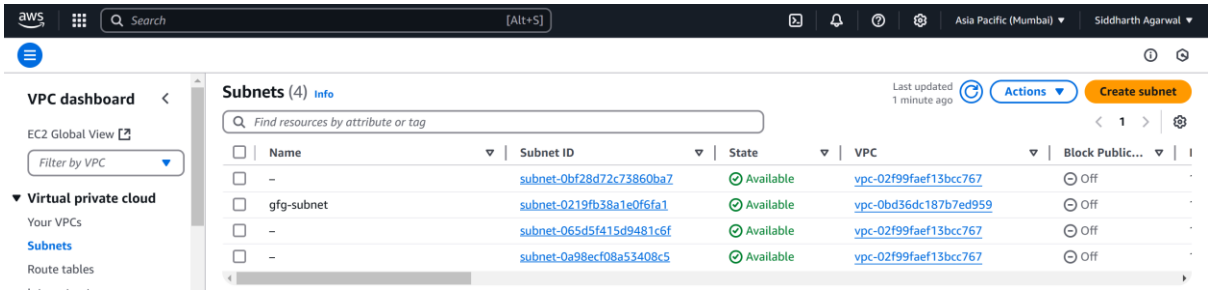
```
Plan: 6 to add, 0 to change, 0 to destroy.
aws_vpc.gfg-vpc: Creating...
aws_vpc.gfg-vpc: Creation complete after 1s [id=vpc-0bd36dc187b7ed959]
aws_internet_gateway.gfg-gw: Creating...
aws_subnet.gfg-subnet: Creating...
aws_security_group.gfg-sg: Creating...
aws_internet_gateway.gfg-gw: Creation complete after 1s [id=igw-0cb93272440d23057]
aws_route_table.gfg-rt: Creating...
aws_subnet.gfg-subnet: Creation complete after 1s [id=subnet-0219fb38a1e0f6fa1]
aws_route_table.gfg-rt: Creation complete after 1s [id=rtb-0bf7103c8676ac6d4]
aws_route_table_association.gfg-rta: Creating...
aws_route_table_association.gfg-rta: Creation complete after 0s [id=rtbassoc-001eb0520a40dc441]
aws_security_group.gfg-sg: Creation complete after 3s [id=sg-01264c66904c2cf38]

Apply complete! Resources: 6 added, 0 changed, 0 destroyed.
```

- Terraform will prompt you to confirm the creation of the VPC and subnets. Type yes and press Enter.

3. Verify Resources in AWS Console:

- Log in to the AWS Management Console and navigate to the VPC service.
- Verify that the VPC and subnets with the specified names and settings have been created.





4. Update VPC Configuration:

- If you want to modify the VPC configuration, update the main.tf file with the desired changes.
- Rerun the terraform apply command to apply the changes:

terraform apply

```
PS C:\SID_DATA\SIDDHARTH\UPES COLLEGE STUDY MATERIAL\SEM6\SPCM\lab\lab11\terraform-vpc> terraform apply -auto-approve

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_internet_gateway.gfg-gw will be created
  + resource "aws_internet_gateway" "gfg-gw" {
      + arn      = (known after apply)
      + id       = (known after apply)
      + owner_id = (known after apply)
      + tags     = {
```

5. Clean Up:

After testing, you can clean up the VPC and subnets:

terraform destroy

```
PS C:\SID_DATA\SIDDHARTH\UPES COLLEGE STUDY MATERIAL\SEM6\SPCM\lab\lab11\terraform-vpc> terraform destroy -auto-approve
aws_vpc.gfg-vpc: Refreshing state... [id=vpc-0bd36dc187b7ed959]
aws_subnet.gfg-subnet: Refreshing state... [id=subnet-0219fb38a1e0f6fa1]
aws_internet_gateway.gfg-gw: Refreshing state... [id=igw-0cb93272440d23057]
aws_security_group.gfg-sg: Refreshing state... [id=sg-01264c66904c2cf38]
aws_route_table.gfg-rt: Refreshing state... [id=rtb-0bf7103c8676ac6d4]
aws_route_table_association.gfg-rta: Refreshing state... [id=rtbassoc-001eb0520a40dc441]

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_internet_gateway.gfg-gw will be destroyed
  - resource "aws_internet_gateway" "gfg-gw" {
      - arn      = "arn:aws:ec2:ap-south-1:690511669638:internet-gateway/igw-0cb93272440d23057" -> null
      - id       = "igw-0cb93272440d23057" -> null
      - owner_id = "690511669638" -> null
      - tags     = {
          - "Name" = "gfg-IG"
        } -> null
      - tags_all = {
          - "Name" = "gfg-IG"
        } -> null
      - vpc_id   = "vpc-0bd36dc187b7ed959" -> null
    }

  # aws_route_table.gfg-rt will be destroyed
  - resource "aws_route_table" "gfg-rt" {
      - arn              = "arn:aws:ec2:ap-south-1:690511669638:route-table/rtb-0bf7103c8676ac6d4" -> null
      - id               = "rtb-0bf7103c8676ac6d4" -> null
      - owner_id         = "690511669638" -> null
      - propagating_vgws = [] -> null
      - route            = [
          - {
              - cidr_block                = "0.0.0.0/0"
```

```
  # aws_vpc.gfg-vpc will be destroyed
  - resource "aws_vpc" "gfg-vpc" {
      - arn                                  = "arn:aws:ec2:ap-south-1:690511669638:vpc/vpc-0bd36dc187b7ed959" -> null
      - assign_generated_ipv6_cidr_block     = false -> null
      - cidr_block                           = "10.0.0.0/16" -> null
      - default_network_acl_id               = "acl-08b661e482265bdee" -> null
      - default_route_table_id               = "rtb-04089017ba541c51b" -> null
      - default_security_group_id            = "sg-0f84154173c598083" -> null
      - dhcp_options_id                      = "dopt-00baeef9542056942" -> null
      - enable_dns_hostnames                 = false -> null
      - enable_dns_support                   = true -> null
      - enable_network_address_usage_metrics = false -> null
      - id                                   = "vpc-0bd36dc187b7ed959" -> null
      - instance_tenancy                     = "default" -> null
      - ipv6_netmask_length                  = 0 -> null
      - main_route_table_id                  = "rtb-04089017ba541c51b" -> null
      - owner_id                             = "690511669638" -> null
      - tags                                 = {} -> null
      - tags_all                             = {} -> null
        # (4 unchanged attributes hidden)
    }

Plan: 0 to add, 0 to change, 6 to destroy.
aws_route_table_association.gfg-rta: Destroying... [id=rtbassoc-001eb0520a40dc441]
aws_security_group.gfg-sg: Destroying... [id=sg-01264c66904c2cf38]
aws_route_table_association.gfg-rta: Destruction complete after 0s
aws_route_table.gfg-rt: Destroying... [id=rtb-0bf7103c8676ac6d4]
aws_subnet.gfg-subnet: Destroying... [id=subnet-0219fb38a1e0f6fa1]
aws_security_group.gfg-sg: Destruction complete after 1s
aws_subnet.gfg-subnet: Destruction complete after 1s
aws_route_table.gfg-rt: Destruction complete after 1s
aws_internet_gateway.gfg-gw: Destroying... [id=igw-0cb93272440d23057]
aws_internet_gateway.gfg-gw: Destruction complete after 0s
aws_vpc.gfg-vpc: Destroying... [id=vpc-0bd36dc187b7ed959]
aws_vpc.gfg-vpc: Destruction complete after 1s

Destroy complete! Resources: 6 destroyed.
```

Confirm the destruction by typing yes.

6. Conclusion:

This lab exercise demonstrates how to create a basic Virtual Private Cloud (VPC) with subnets in AWS using Terraform. The example includes a simple VPC configuration with two subnets. Experiment with different CIDR blocks, settings, and additional AWS resources to customize your VPC.