**School of Computer Science**

**UNIVERSITY OF PETROLEUM AND ENERGY STUDIES**

**DEHRADUN, UTTARAKHAND**



**System Provisioning and Configuration Management**

**Lab File (2022-2026)**

# **6th Semester**

|  |  |
| --- | --- |
| *Submitted To:*  ***Dr. Hitesh Kumar Sharma*** | *Submitted By:*  *Madhav Madan*  *(500105699)*  *B Tech CSE*  *DevOps[6th Semester]*  *R2142220265*  *Batch - 1* |

**EXPERIMENT 11**

**Lab Exercise: Creating a VPC in Terraform.**

**Objective:**

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

**Prerequisites:**

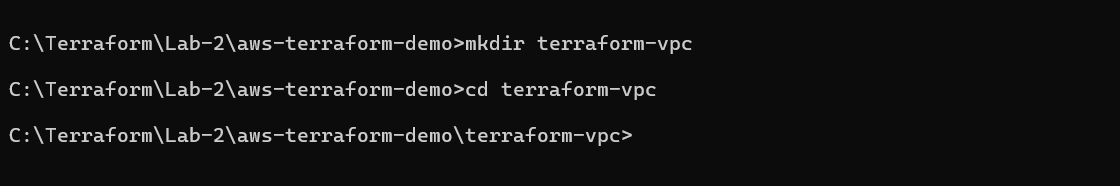
* Terraform installed on your machine.
* AWS CLI configured with the necessary credentials.

**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 = {

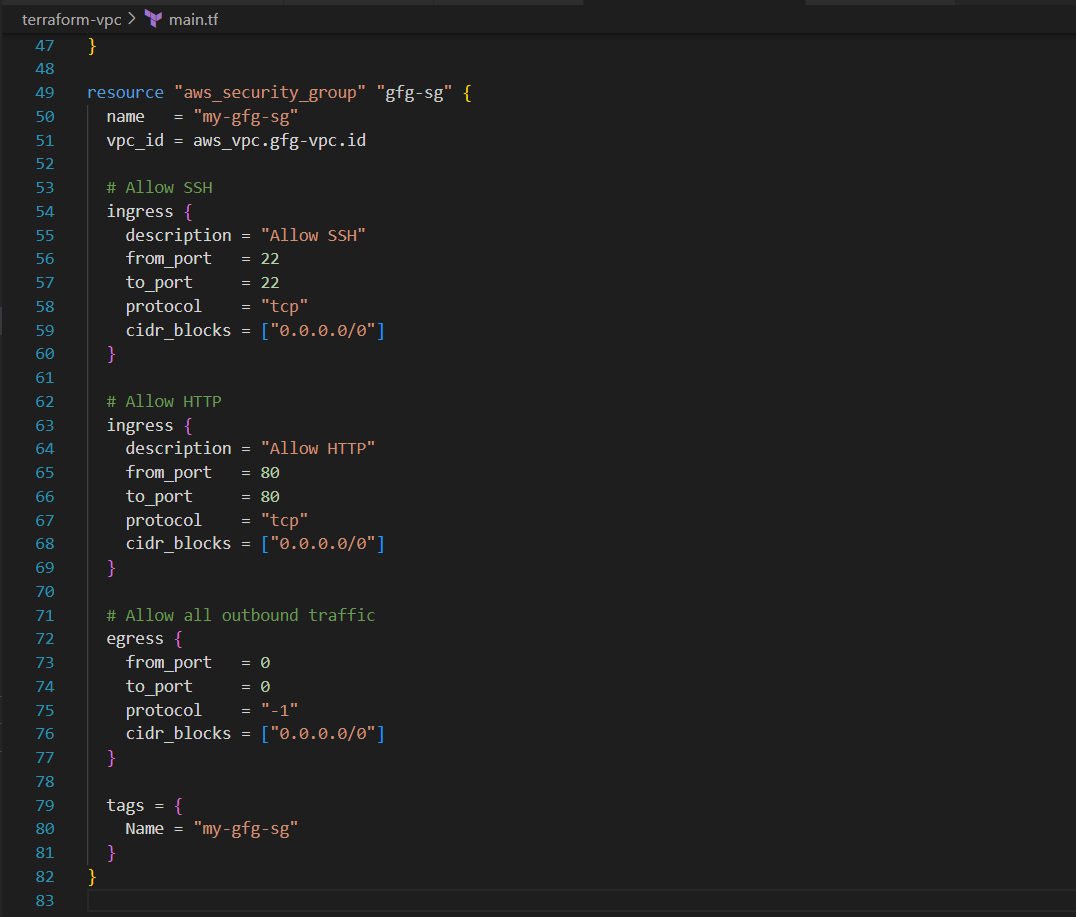
Name = "my-gfg-sg"

}

}

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





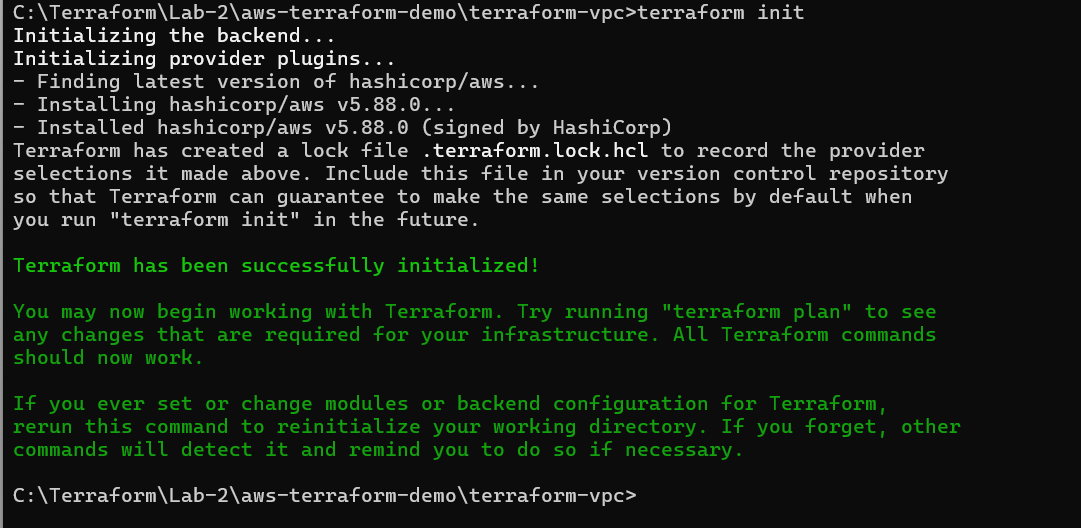
1. **Initialize and Apply:**

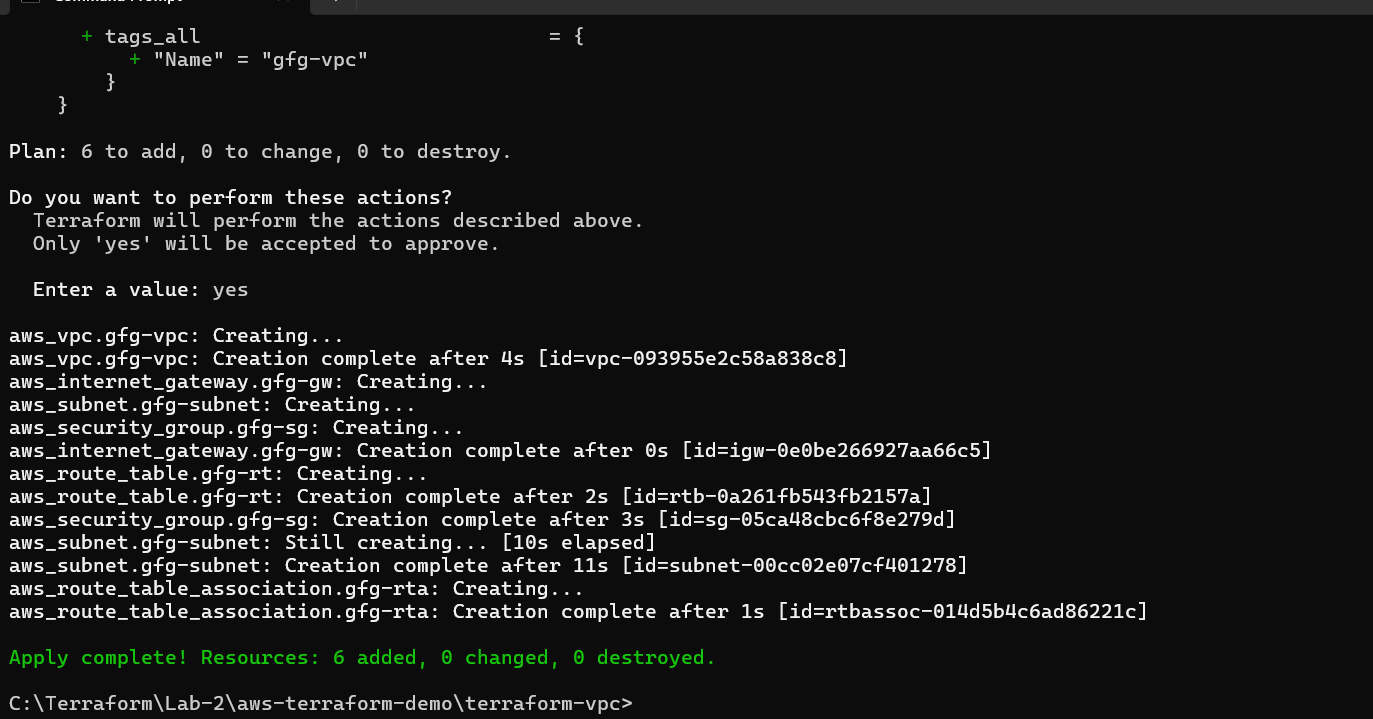
* Run the following Terraform commands to initialize and apply the configuration:

**terraform init**

**terraform apply**

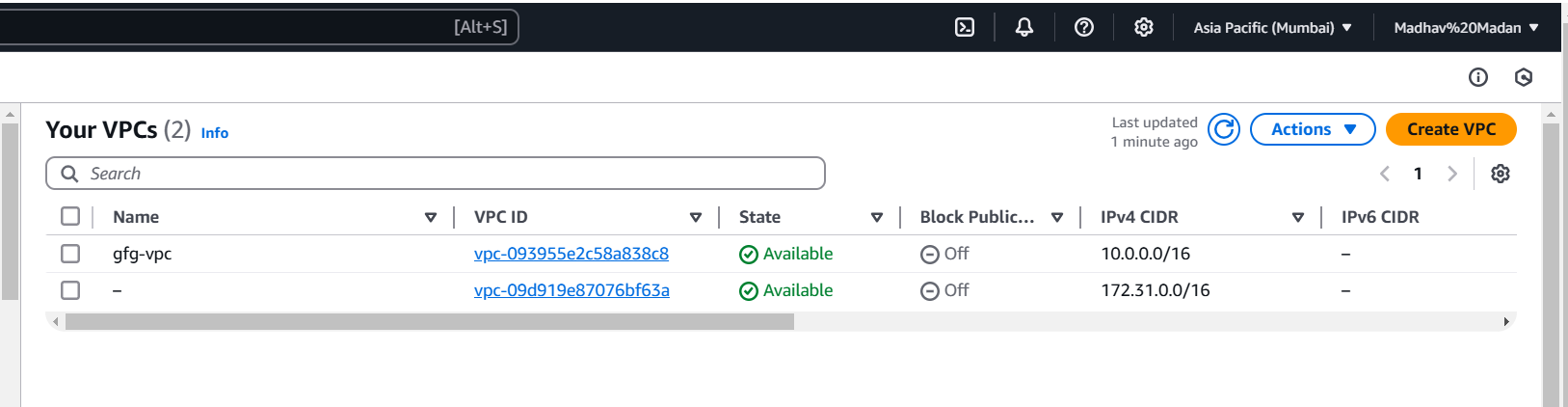
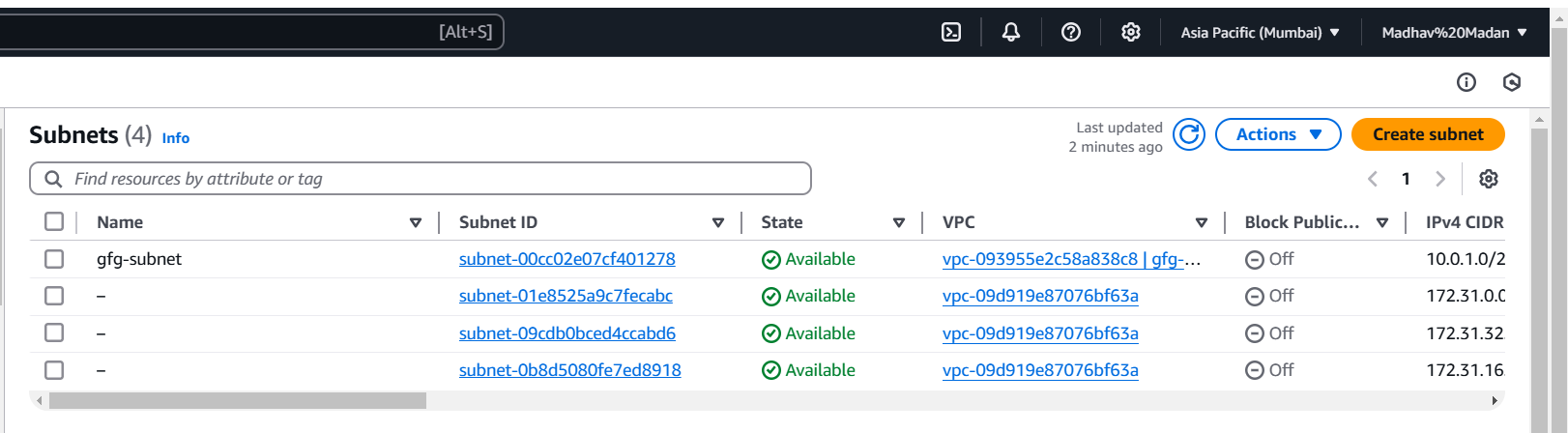
* Terraform will prompt you to confirm the creation of the VPC and subnets. Type yes and press Enter.





1. **Verify Resources in AWS Console:**

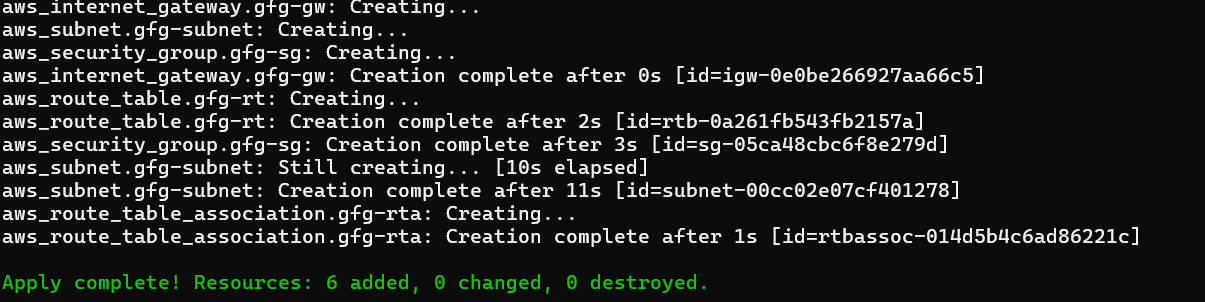
* Log in to the AWS Management Console and navigate to the VPC service.
* Verify that the VPC and subnets with the specified names and settings have been created.

1. **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**

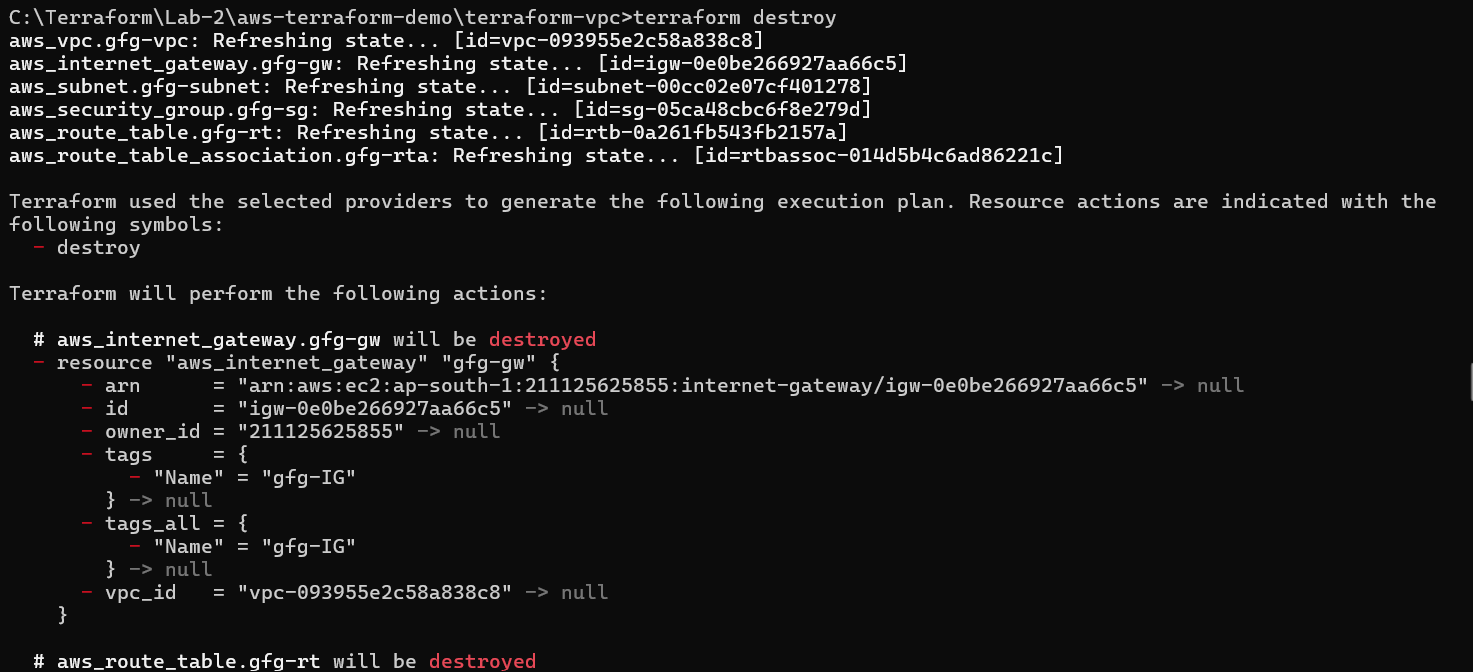


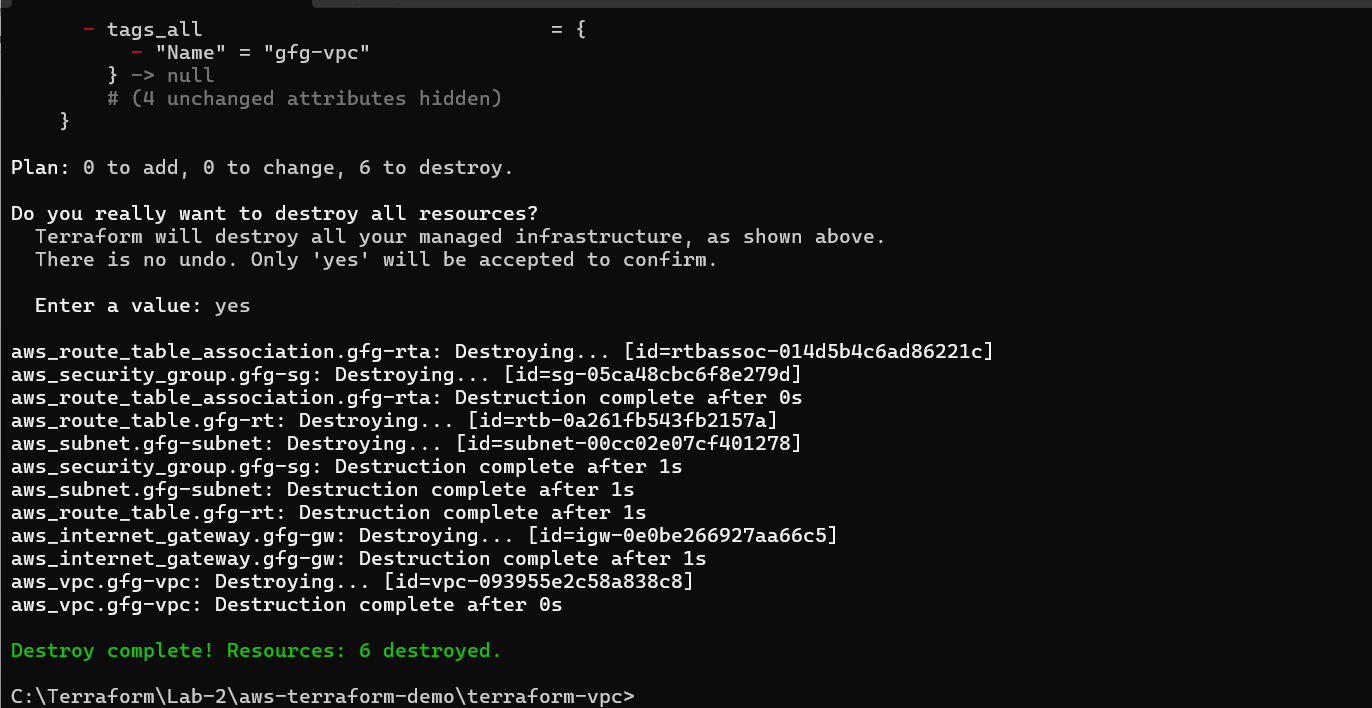
1. **Clean Up:**

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

**terraform destroy**

Confirm the destruction by typing yes.





1. **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.