Lesson 11 Demo 07

Working with Graphs

Objective: To create and visualize Terraform resource dependency graphs for better infrastructure management

Tools required: Terraform**,** AWS, and Visual Studio Code

Prerequisites: Refer to the **Demo 01** of **Lesson 11** for creating access and secret key

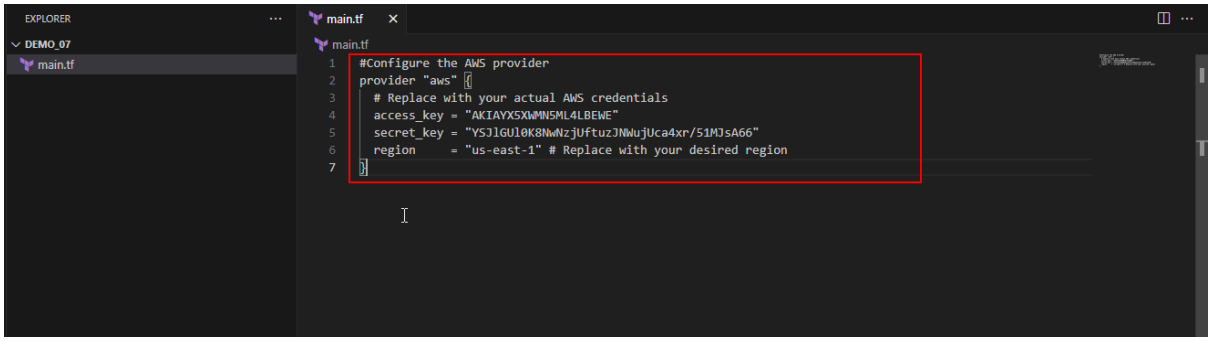
Steps to be followed:

1. Set up infrastructure with dependencies
2. Generate and visualize the resource graph

Step 1: Set up infrastructure with dependencies

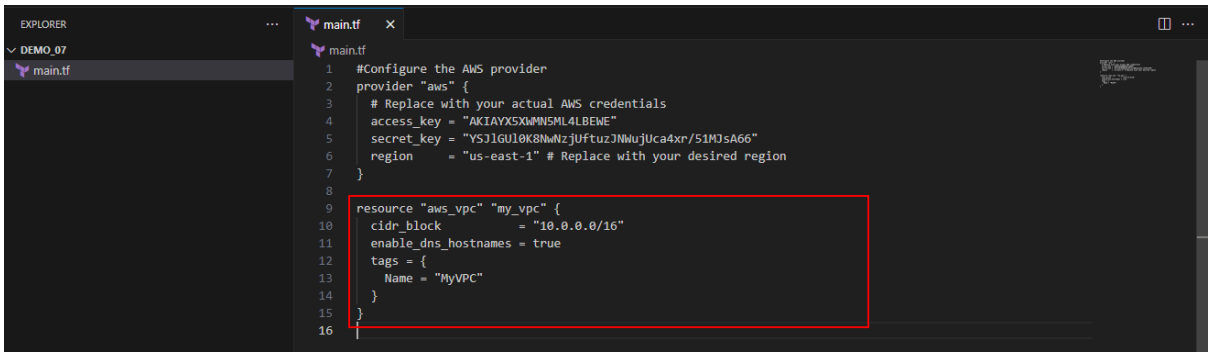
1.1 Open your Terraform configuration environment and create a file named **main.tf**, and add the following configuration block as shown in the screenshot below:

```
#Configure the AWS provider
provider "aws" {
  # Replace with your actual AWS credentials
  access_key = "YOUR_ACCESS_KEY"
  secret_key = "YOUR_SECRET_KEY"
  region     = "us-east-1" # Replace with your desired region
}
```



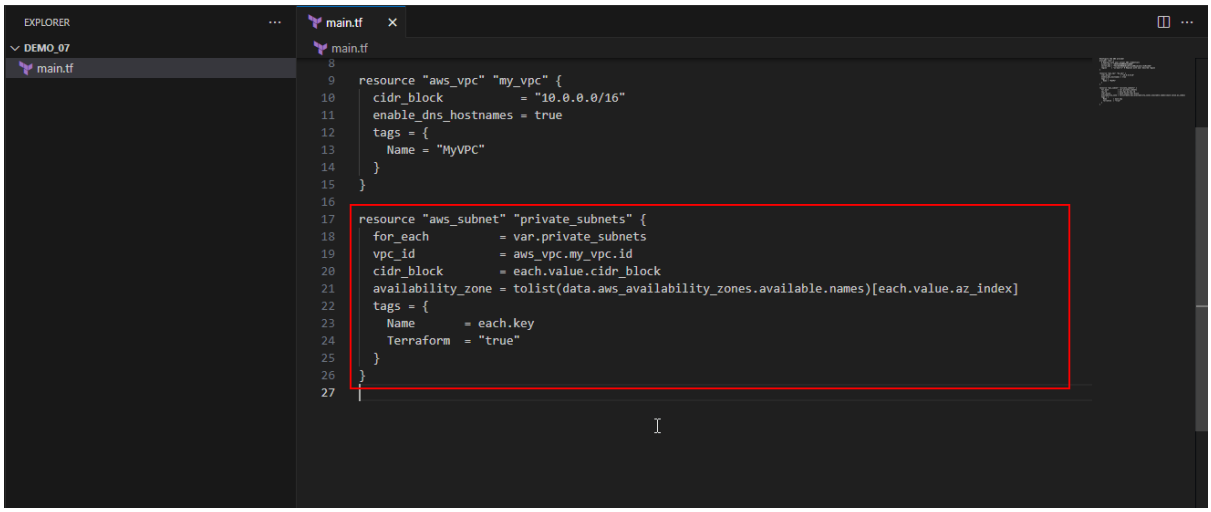
1.2 Add the following block to define a VPC with a CIDR block and enable DNS hostnames as shown in the screenshot below:

```
resource "aws_vpc" "my_vpc" {
  cidr_block          = "10.0.0.0/16"
  enable_dns_hostnames = true
  tags = {
    Name = "MyVPC"
  }
}
```



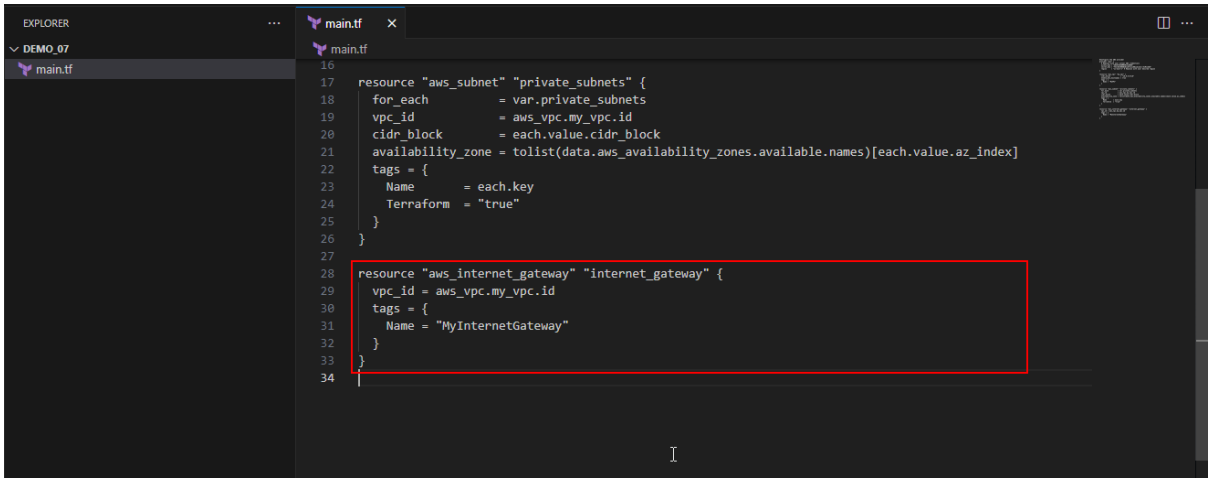
1.3 Add the following block to declare subnets within the VPC:

```
resource "aws_subnet" "private_subnets" {
  for_each        = var.private_subnets
  vpc_id          = aws_vpc.my_vpc.id
  cidr_block      = each.value.cidr_block
  availability_zone =
tolist(data.aws_availability_zones.available.names)[each.value.az_index]
  tags = {
    Name      = each.key
    Terraform  = "true"
  }
}
```

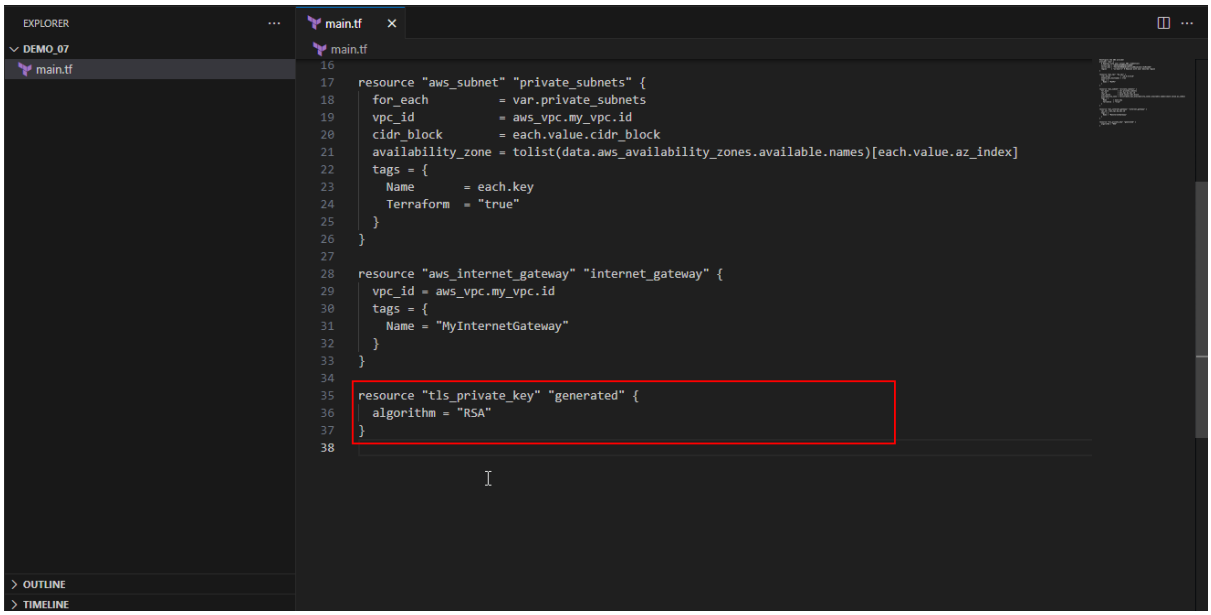


1.4 Add the following block and set up an internet gateway as shown in the screenshot below:

```
resource "aws_internet_gateway" "internet_gateway" {
  vpc_id = aws_vpc.my_vpc.id
  tags = {
    Name = "MyInternetGateway"
  }
}
```

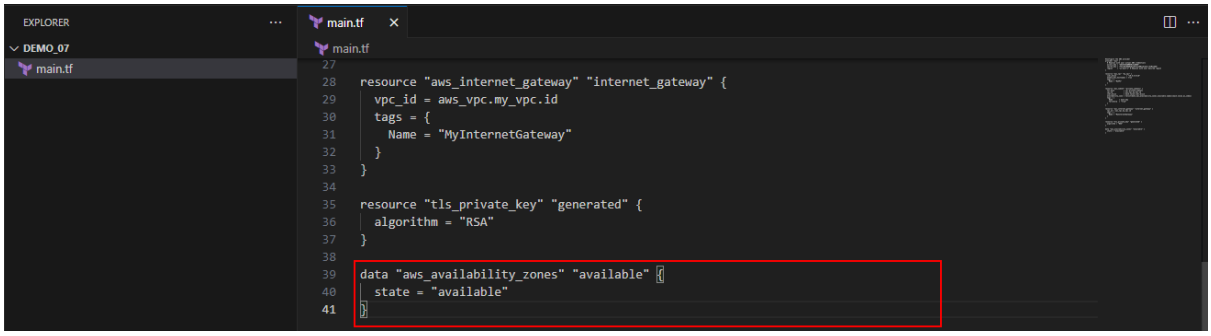


1.5 Add the following block to generate a TLS private key as shown in the screenshot below:



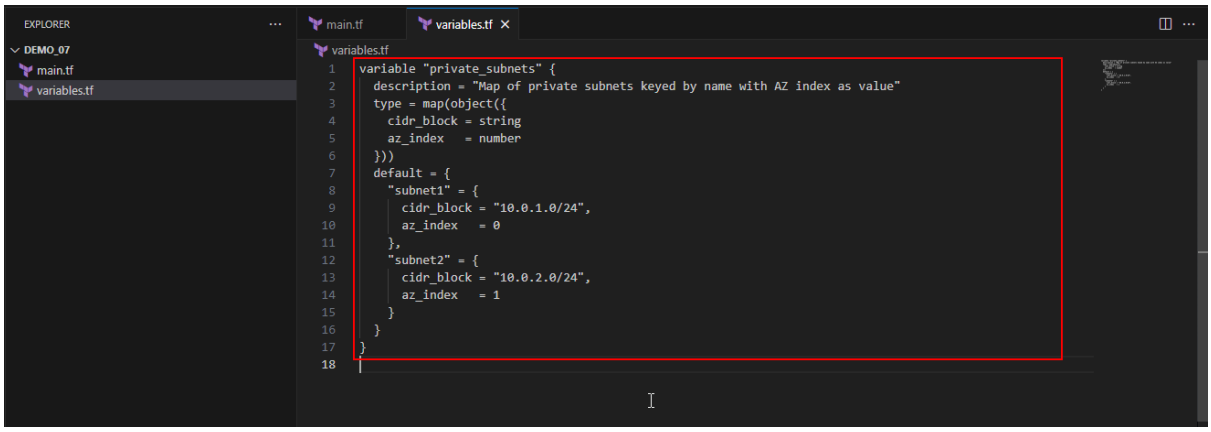
1.6 Add the following data block to retrieve available AWS availability zones:

```
data "aws_availability_zones" "available" {
 state = "available"
}
```

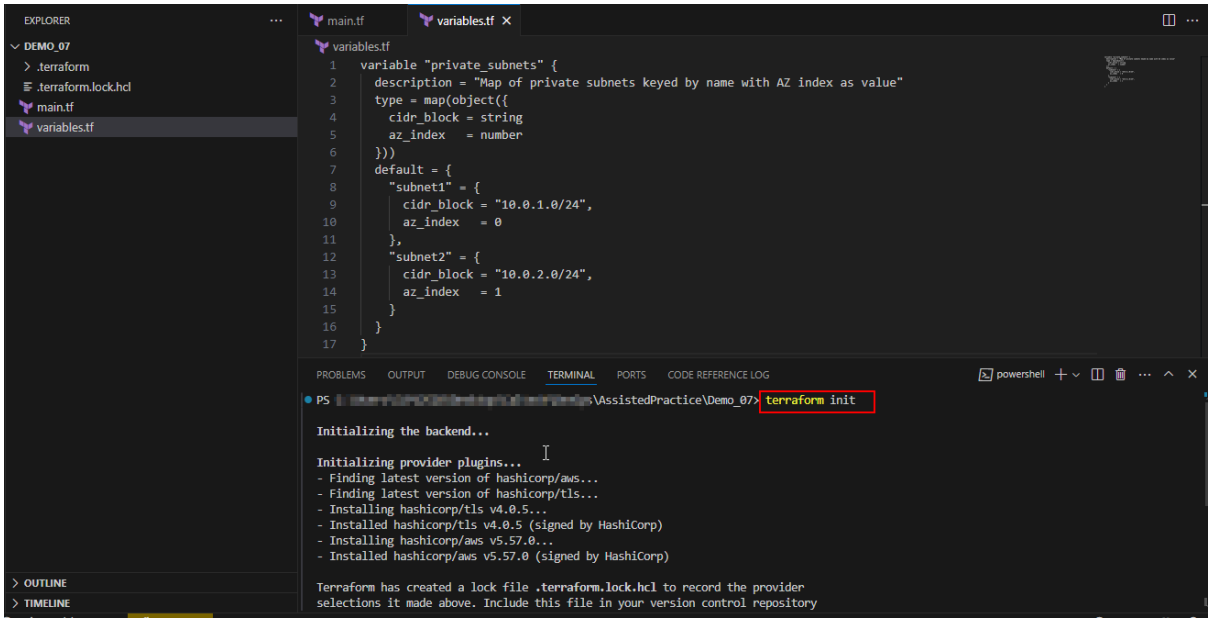


1.7 Create a file named **variables.tf** and add the following block to declare the **private_subnets** variable in your Terraform configuration:

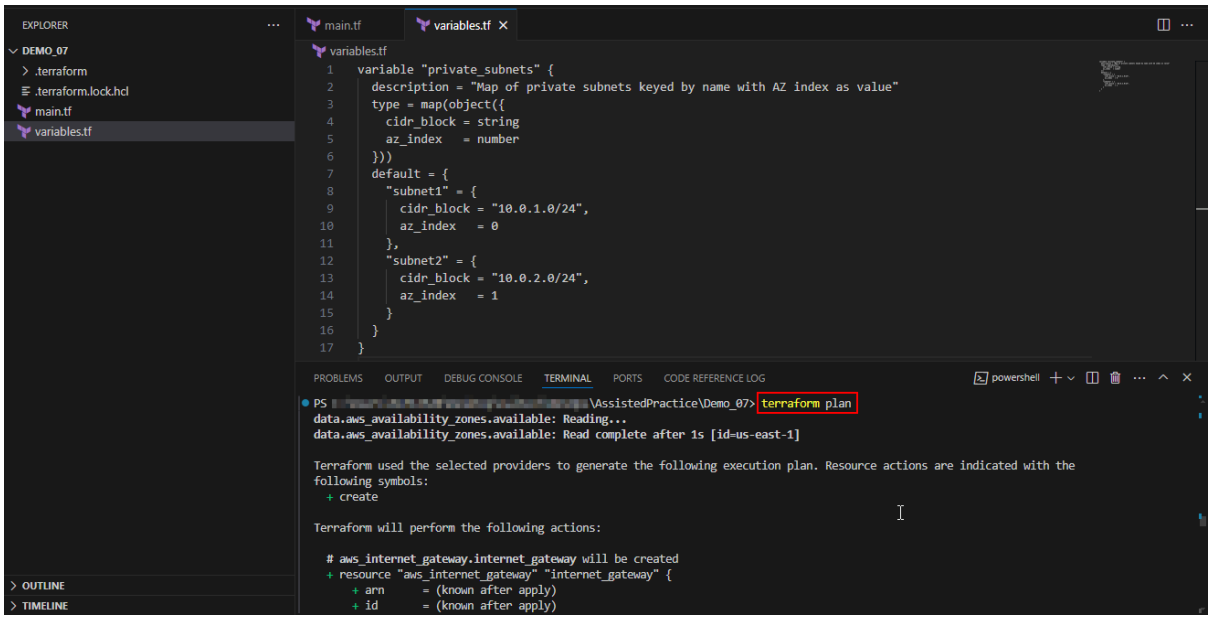
```
variable "private_subnets" {
 description = "Map of private subnets keyed by name with AZ index as value"
 type = map(object({
  cidr_block = string
  az_index   = number
 }))
 default = {
  "subnet1" = {
   cidr_block = "10.0.1.0/24",
   az_index   = 0
  },
  "subnet2" = {
   cidr_block = "10.0.2.0/24",
   az_index   = 1
  }
 }
}
```



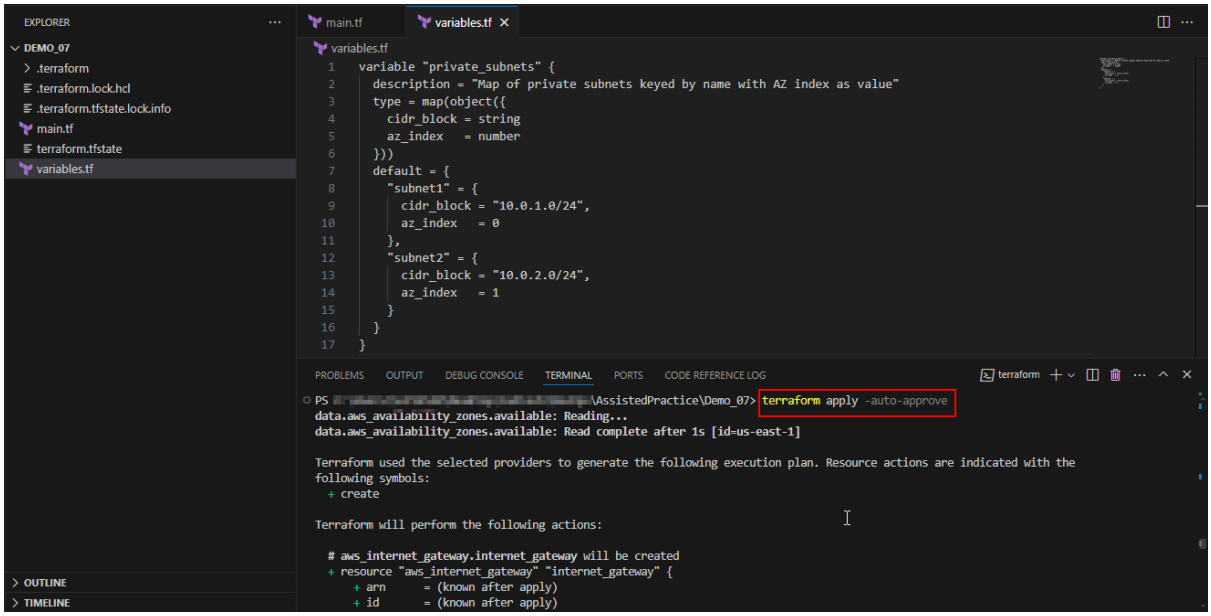
1.8 Run **terraform init** to initialize the Terraform configuration

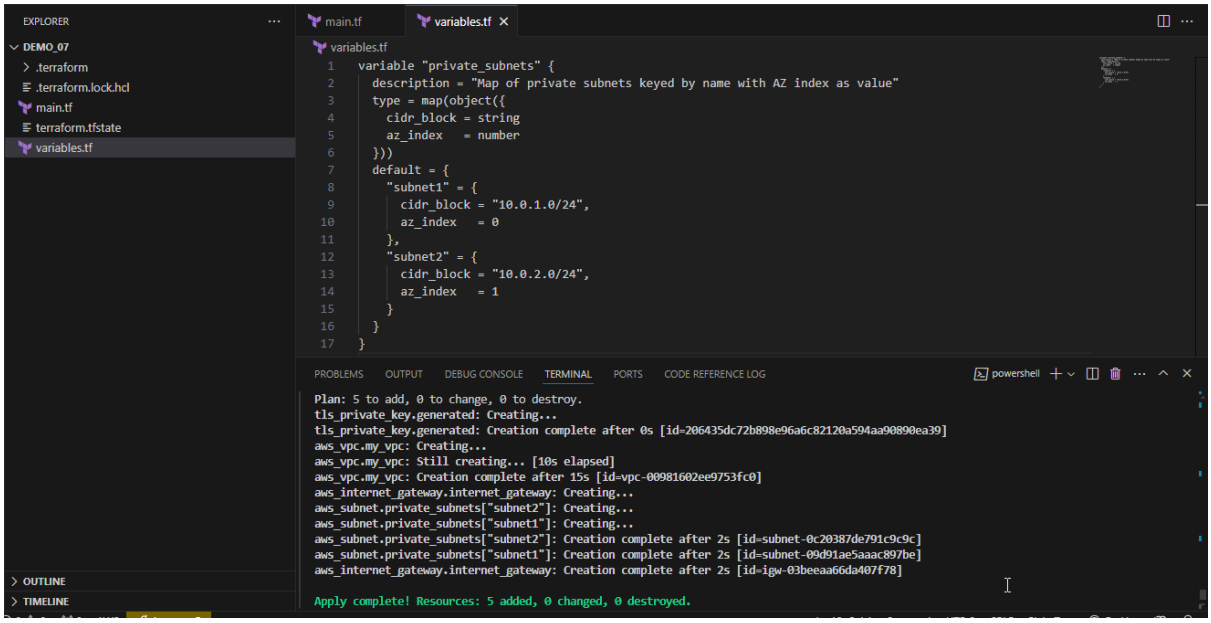


1.9 Plan the deployment using the following command to see the proposed changes: **terraform plan**



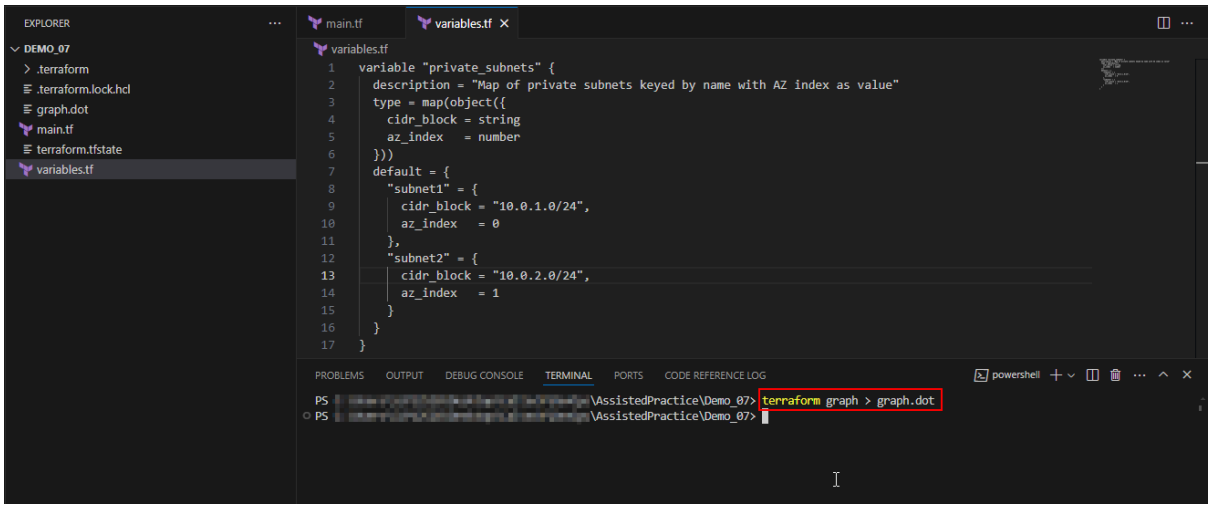
1.10 Execute **terraform apply -auto-approve** to create the infrastructure as defined



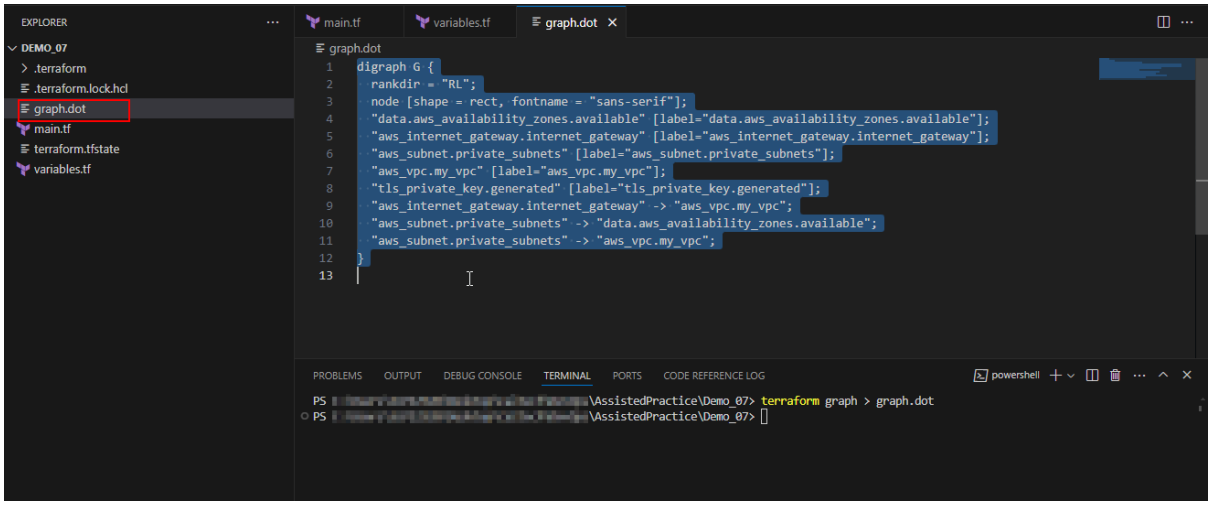


Step 2: Generate and visualize the resource graph

2.1 Use the following command to generate a dependency graph of the Terraform configuration:
terraform graph > graph.dot



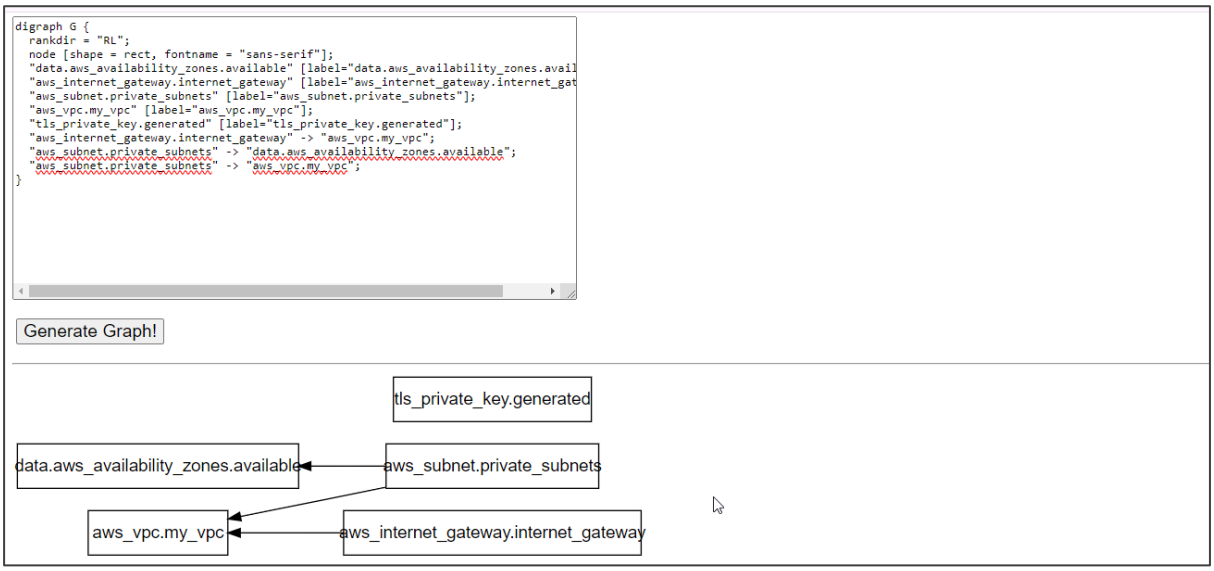
2.2 Click on the **graph.dot** file and copy the diagraph as shown in the screenshot below:



2.3 Navigate to **http://www.webgraphviz.com/?tab=map** as shown in the screenshot below:

2.4 Paste the copied data in the input box and click on the **Generate Graph!** button as shown in the screenshot below:

You will see the generated graph as shown in the screenshot below:



By following these steps, you have successfully created and visualized Terraform resource dependency graphs for better infrastructure management.