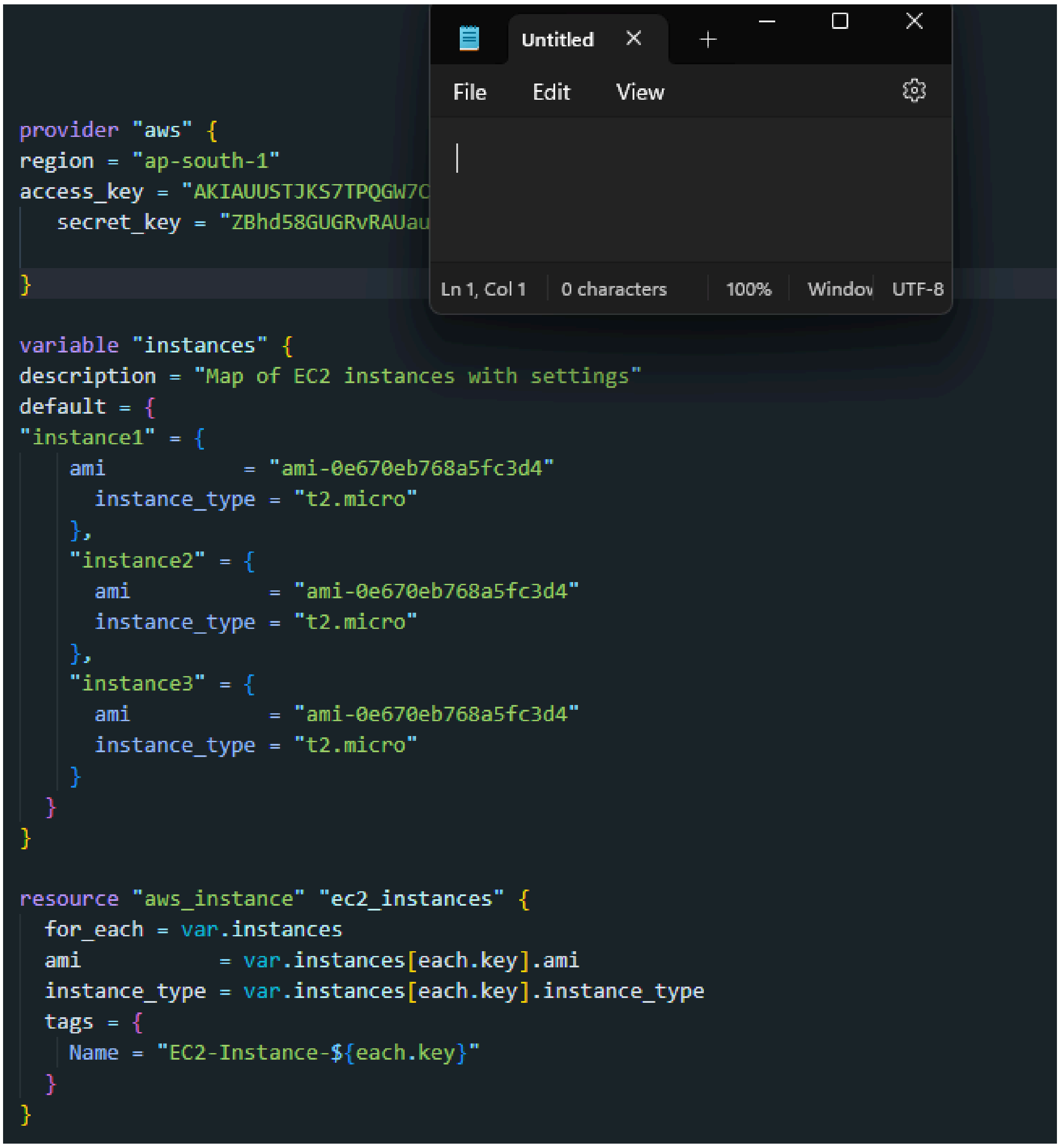
SUBMITTED BY: Pranay Mayal, B2 (NON HONS.)

Lab Exercise 8– 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.

Step 1 Create main.tf



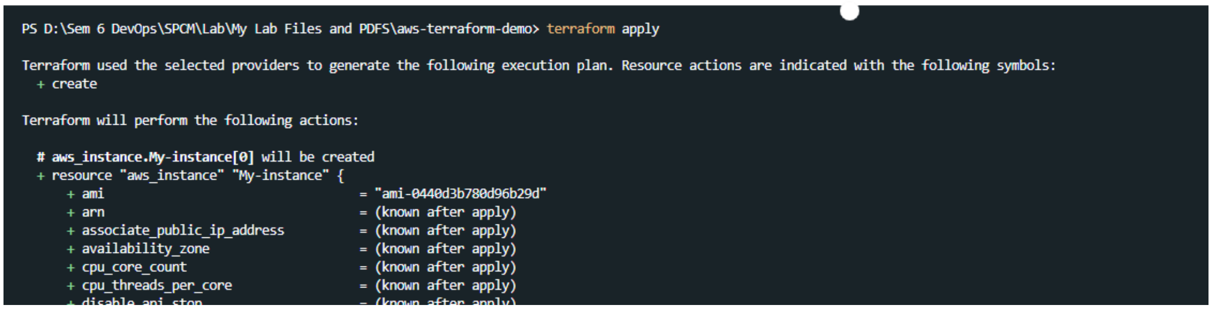
```
provider "aws" {
region = "ap-south-1"
access_key = "AKIAUUSTJKS7TPQGW7C
   secret_key = "ZBhd58GUGRvRAUau

}

variable "instances" {
description = "Map of EC2 instances with settings"
default = {
"instance1" = {
    ami           = "ami-0e670eb768a5fc3d4"
      instance_type = "t2.micro"
    },
    "instance2" = {
      ami           = "ami-0e670eb768a5fc3d4"
      instance_type = "t2.micro"
    },
    "instance3" = {
      ami           = "ami-0e670eb768a5fc3d4"
      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}"
  }
}
```

Step 2



```
PS D:\Sem 6 DevOps\SPCM\Lab\My Lab Files and PDFS\aws-terraform-demo> terraform apply

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_instance.My-instance[0] will be created
  + resource "aws_instance" "My-instance" {
      + ami                                  = "ami-0440d3b780d96b29d"
      + 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)
```