# **Lab Exercise 7 – Creating Multiple IAM Users in Terraform**

**Objective:**

Learn how to use Terraform to create multiple IAM users with unique settings.

**Prerequisites:**

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

**Steps:**

1. **Create a Terraform Directory:**

**mkdir terraform-iam-users cd terraform-iam-users**

* + Create Terraform Configuration Files:
  + Create a file named main.tf:

## # main.tf

**provider "aws" { region = "us-east-1"**

**}**

**variable "iam\_users" { type = list(string)**

**default = ["user1", "user2", "user3"]**

**}**

**resource "aws\_iam\_user" "iam\_users" { count = length(var.iam\_users)**

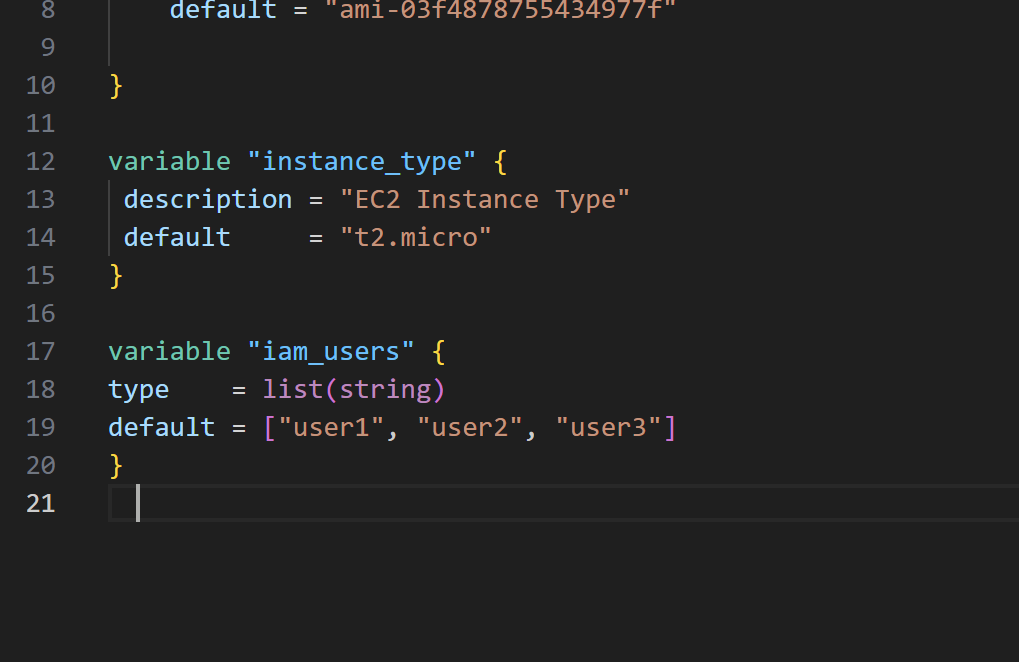
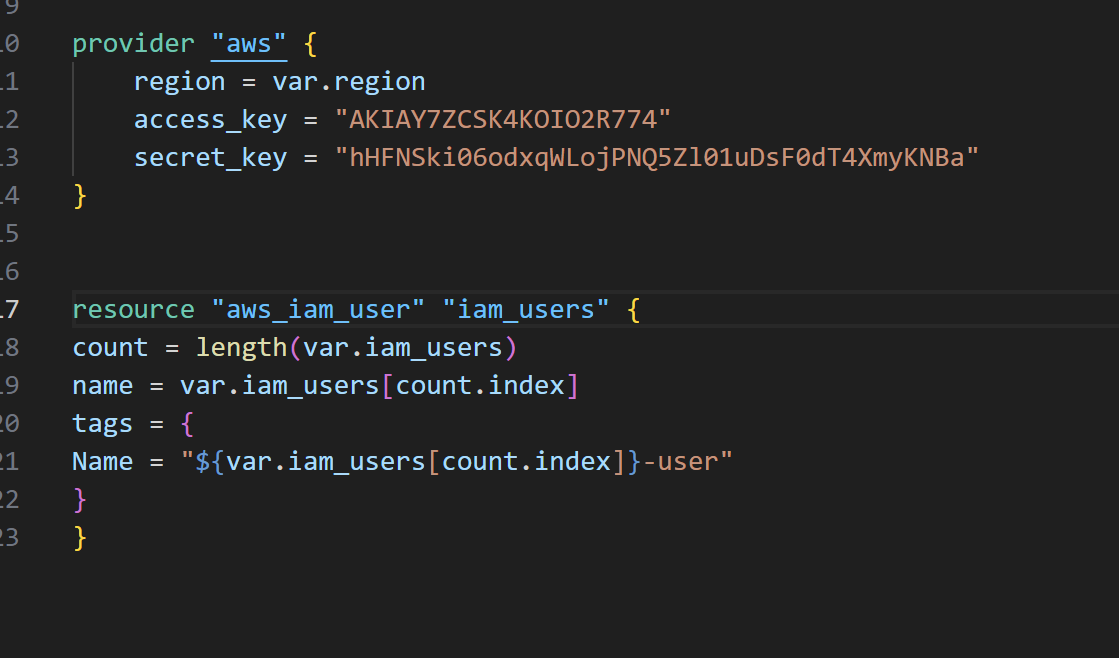
**name = var.iam\_users[count.index]**

**tags = {**

**Name = "${var.iam\_users[count.index]}-user"**

**}**

**}**



In this configuration, we define a list variable iam\_users containing the names of the IAM users we want to create. The aws\_iam\_user resource is then used in a loop to create users based on the values in the list.

# **Initialize and Apply:**

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

**Terraform init provider "aws" { region = "us-east-1"**

**}**

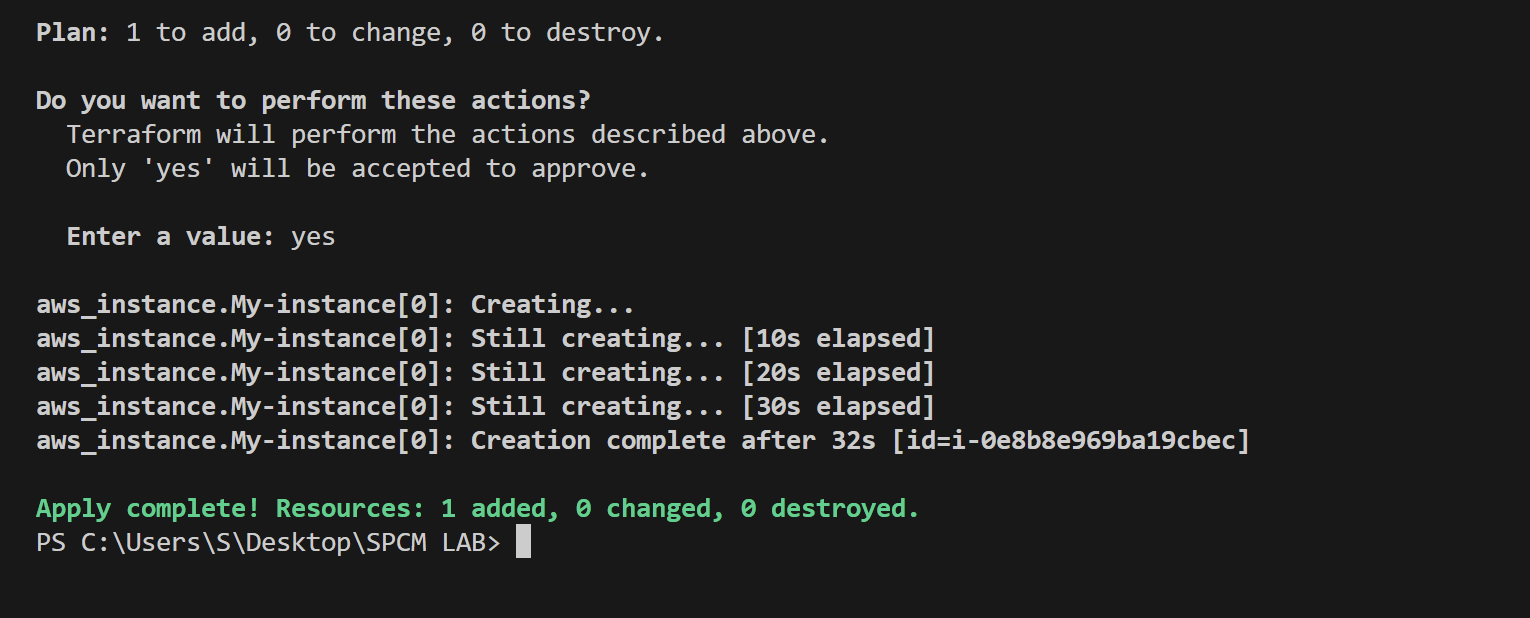
**resource "aws\_vpc" "my\_vpc" { cidr\_block = "10.0.0.0/16" enable\_dns\_support = true enable\_dns\_hostnames = true**

**tags = {**

**Name = "MyVPC"**

**terraform apply**

Terraform will prompt you to confirm the creation of IAM users. Type yes and press Enter.



# **Verify Users in AWS Console:**

* + Log in to the AWS Management Console and navigate to the IAM service.
  + Verify that the IAM users with the specified names and tags have been created.

# **Update IAM Users:**

* + If you want to add or remove IAM users, modify the iam\_users list in the main.tf file.
  + Rerun the terraform apply command to apply the changes:

**terraform apply**

# **Clean Up:**

* + After testing, you can clean up the IAM users:

**terraform destroy**

* + Confirm the destruction by typing yes.

# **Conclusion:**

This lab exercise demonstrates creating multiple IAM users in AWS using Terraform. The use of variables and loops allows you to easily manage and scale the creation of IAM users. Experiment with different user names and settings in the main.tf file to understand how Terraform provisions resources based on your configuration.

# **Lab Exercise 8– 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: #main.tf:

**provider "aws" { region = "us-east-1"**

**}**

**resource "aws\_vpc" "my\_vpc" { cidr\_block = "10.0.0.0/16" enable\_dns\_support = true enable\_dns\_hostnames = true**

**tags = {**

**Name = "MyVPC"**

**}**

**}**

**resource "aws\_subnet" "my\_subnet" {**

**count = 2**

**vpc\_id = aws\_vpc.my\_vpc.id**

**cidr\_block = "10.0.${count.index + 1}.0/24"**

**availability\_zone = "us-east-1a"**

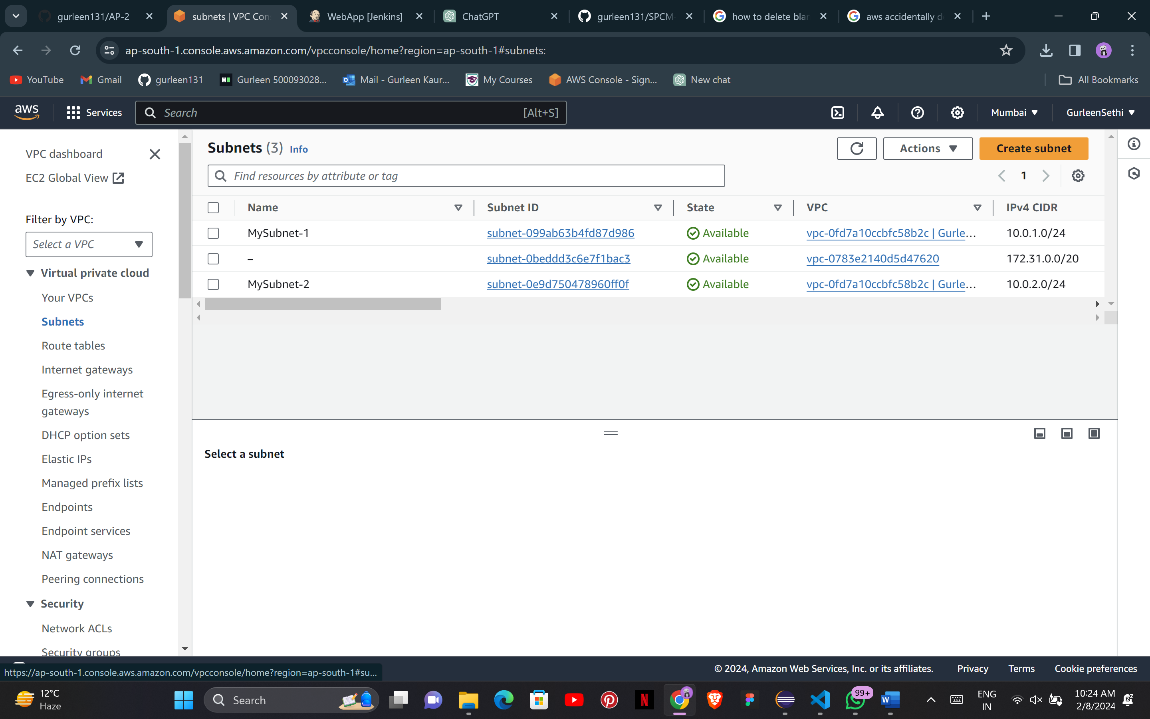
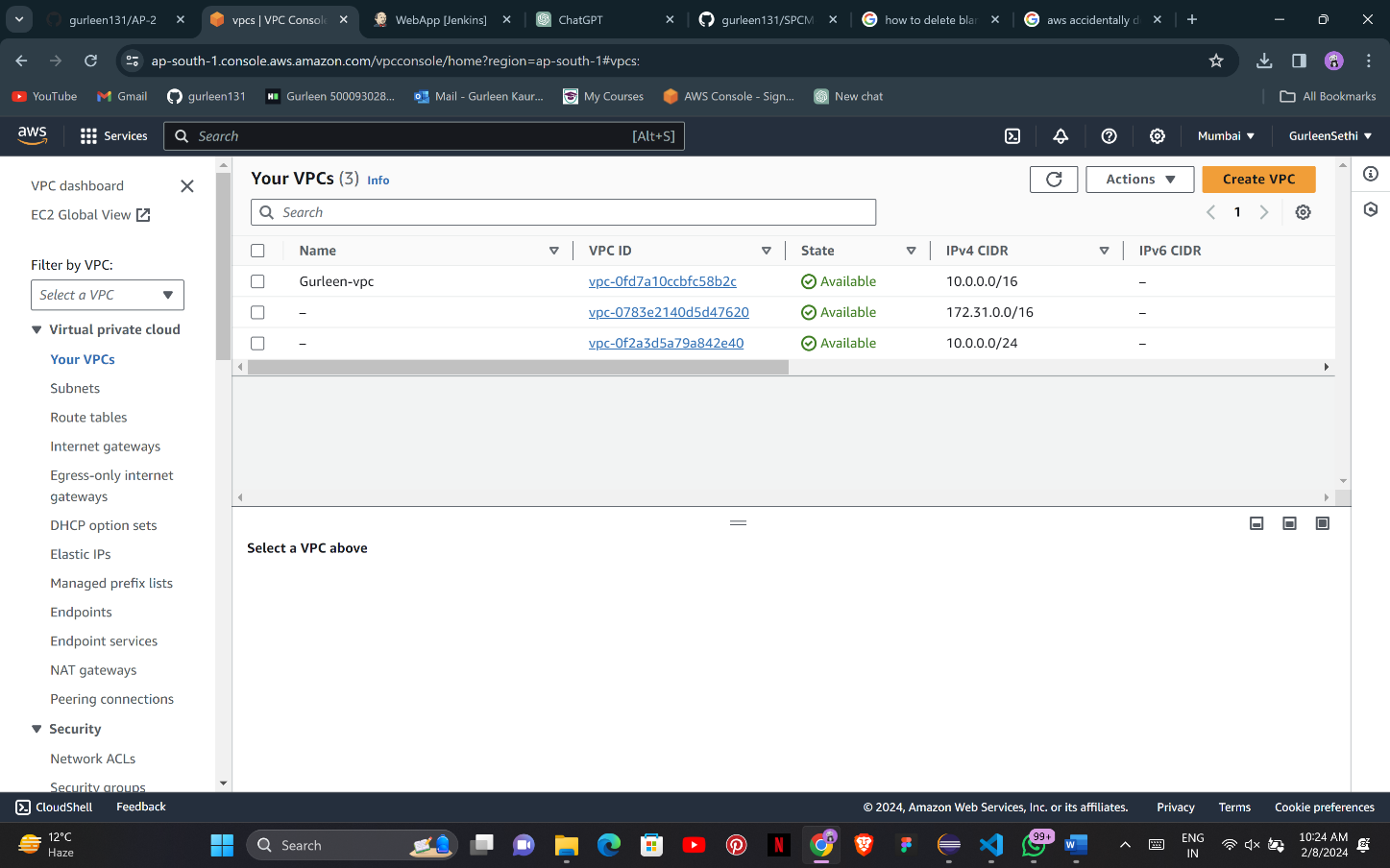
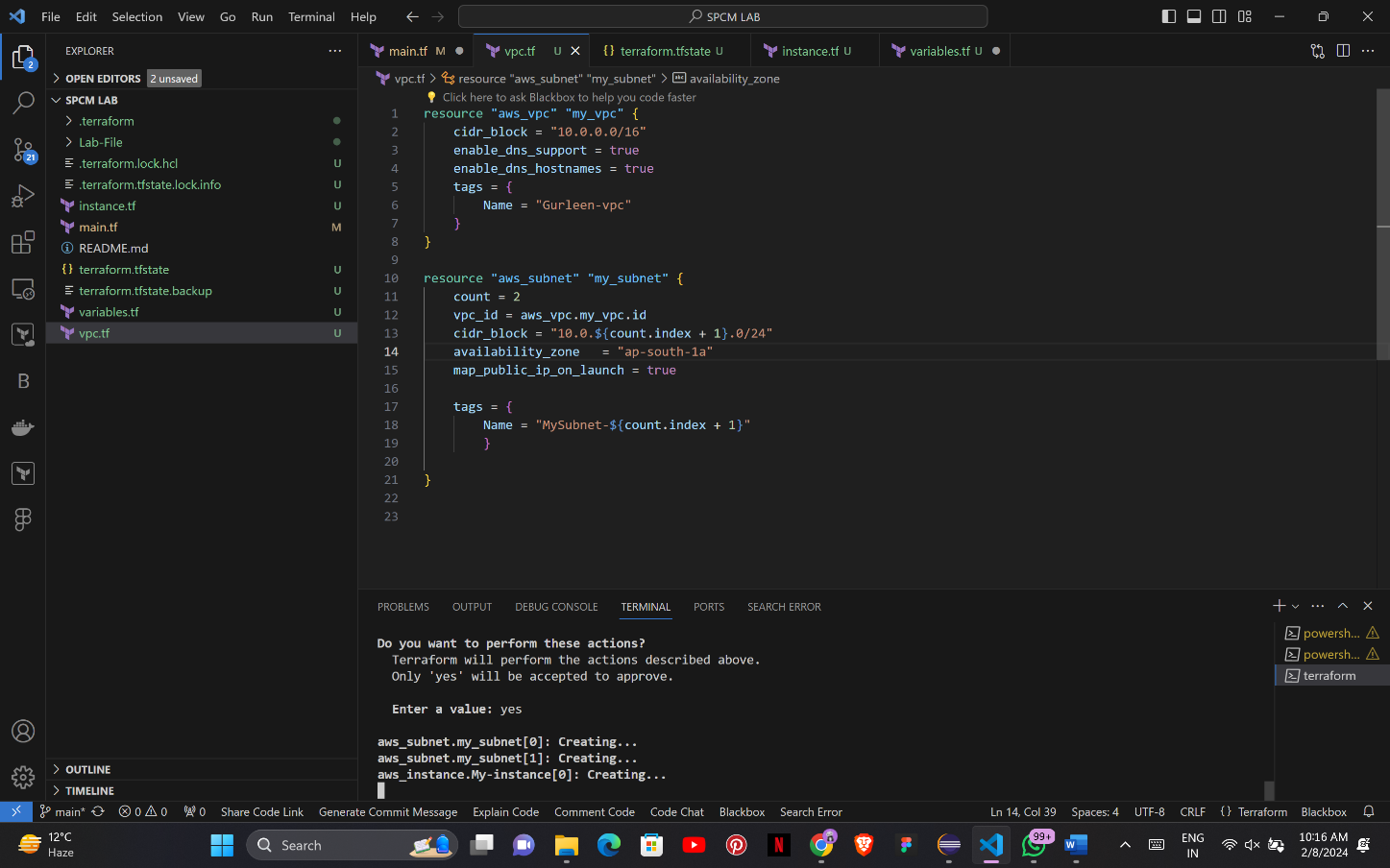
**map\_public\_ip\_on\_launch = true**

**tags = {**

**Name = "MySubnet-${count.index + 1}"**

**}**

**}**



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

# **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.

# **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.

# **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:

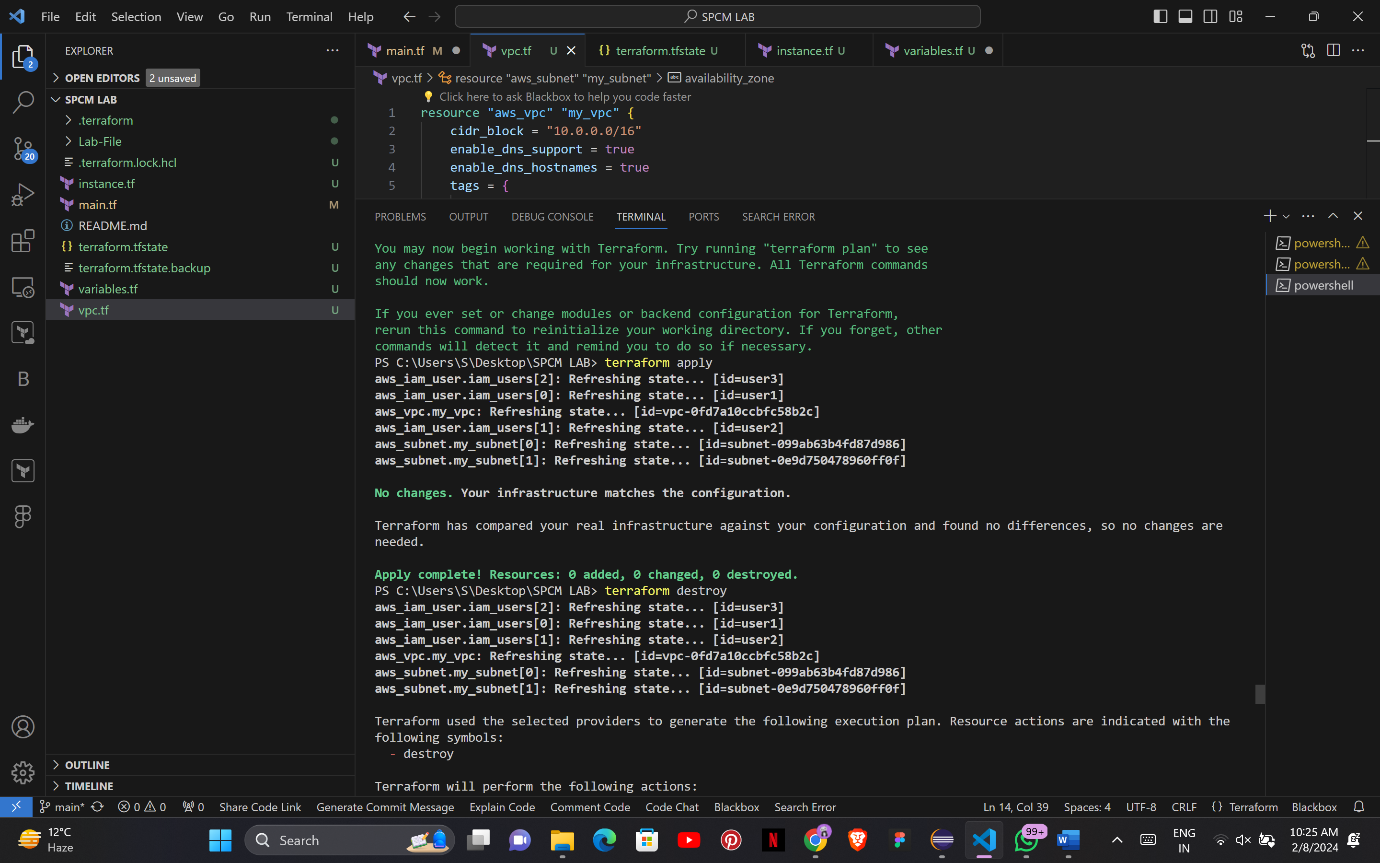
**terraform apply**

**Clean Up:**

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

**terraform destroy**

Confirm the destruction by typing yes.



# **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.