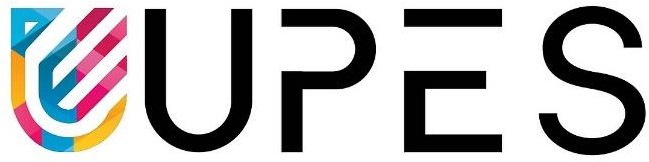
****

**SYSTEM PROVISIONING AND CONFIGURATION MANAGEMENT**

**LAB**

YEAR: 2024

NAME: Aastha Mishra

SAP ID: 500094912

ROLL NO: R2142210013

COURSE: BTech CSE (DevOps)

SEMESTER: Sixth (B3)

FACULTY NAME: Dr.Hitesh Kumar Sharma

# Lab Exercise 5– Terraform Variables with Command Line Arguments

**Objective:**

Learn how to pass values to Terraform variables using command line arguments.

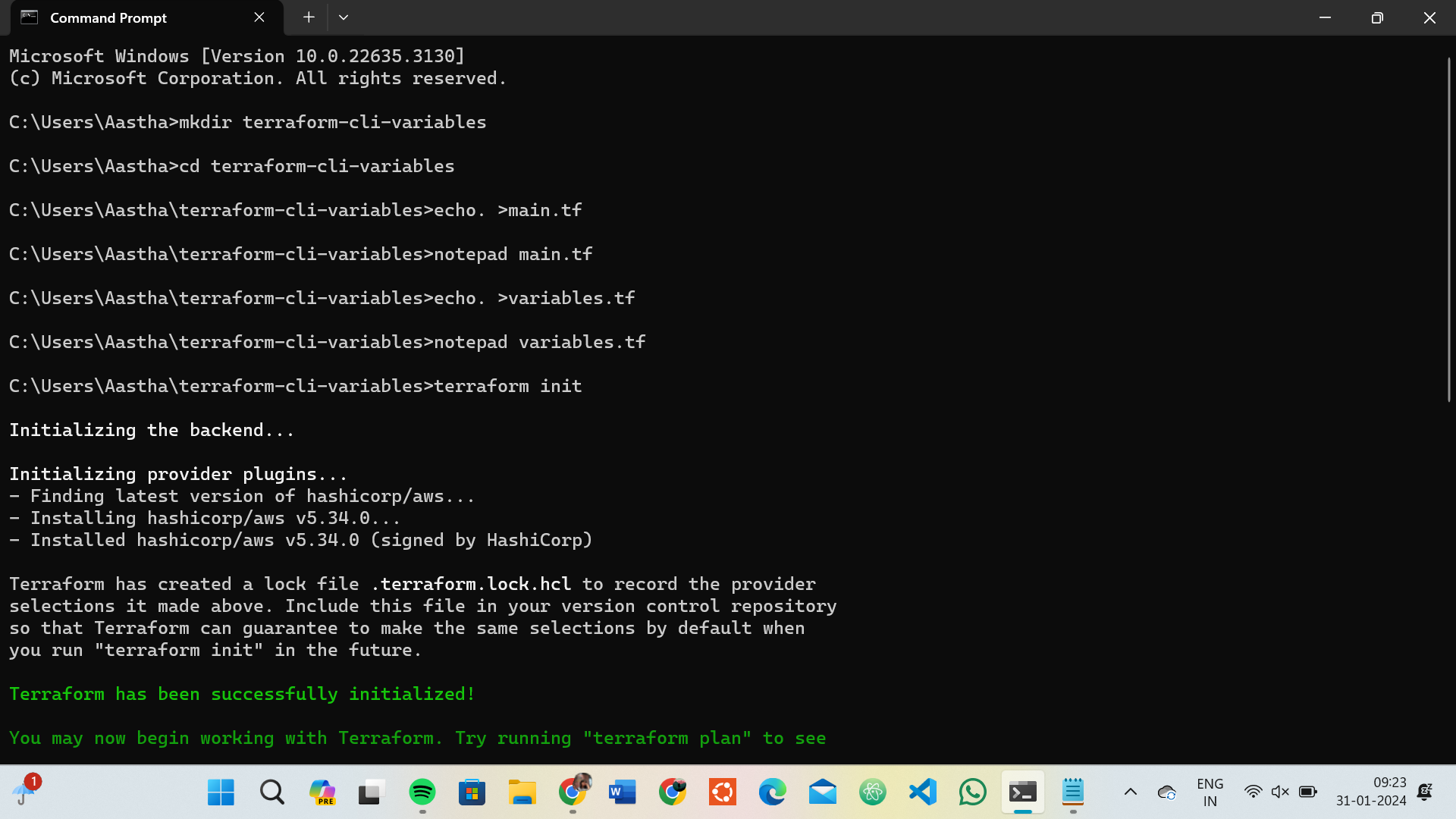
# Prerequisites:

* Terraform installed on your machine.
* Basic knowledge of Terraform variables.

# Steps:

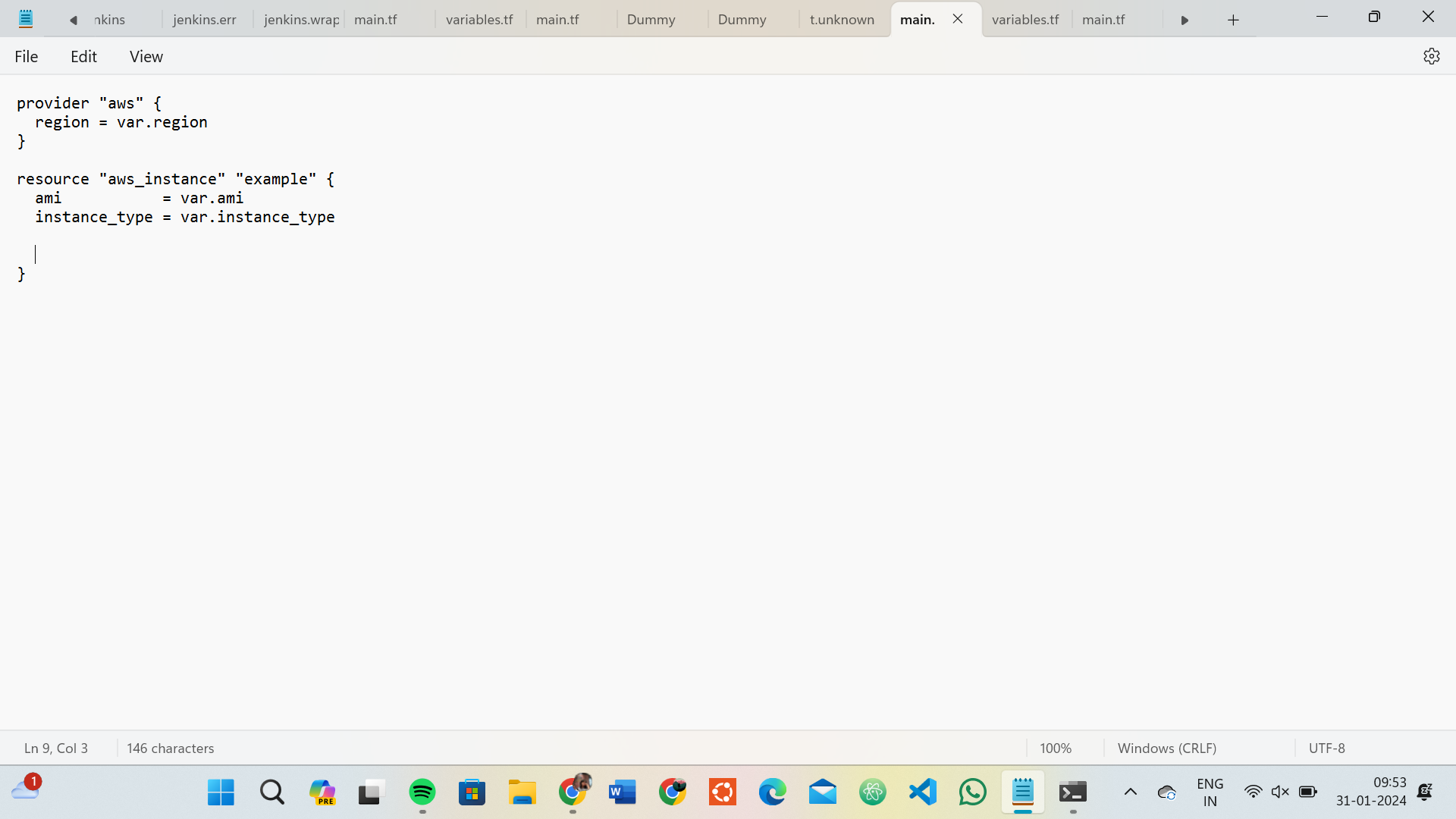
1. **Create a Terraform Directory:**

**mkdir terraform-cli-variables cd terraform-cli-variables**

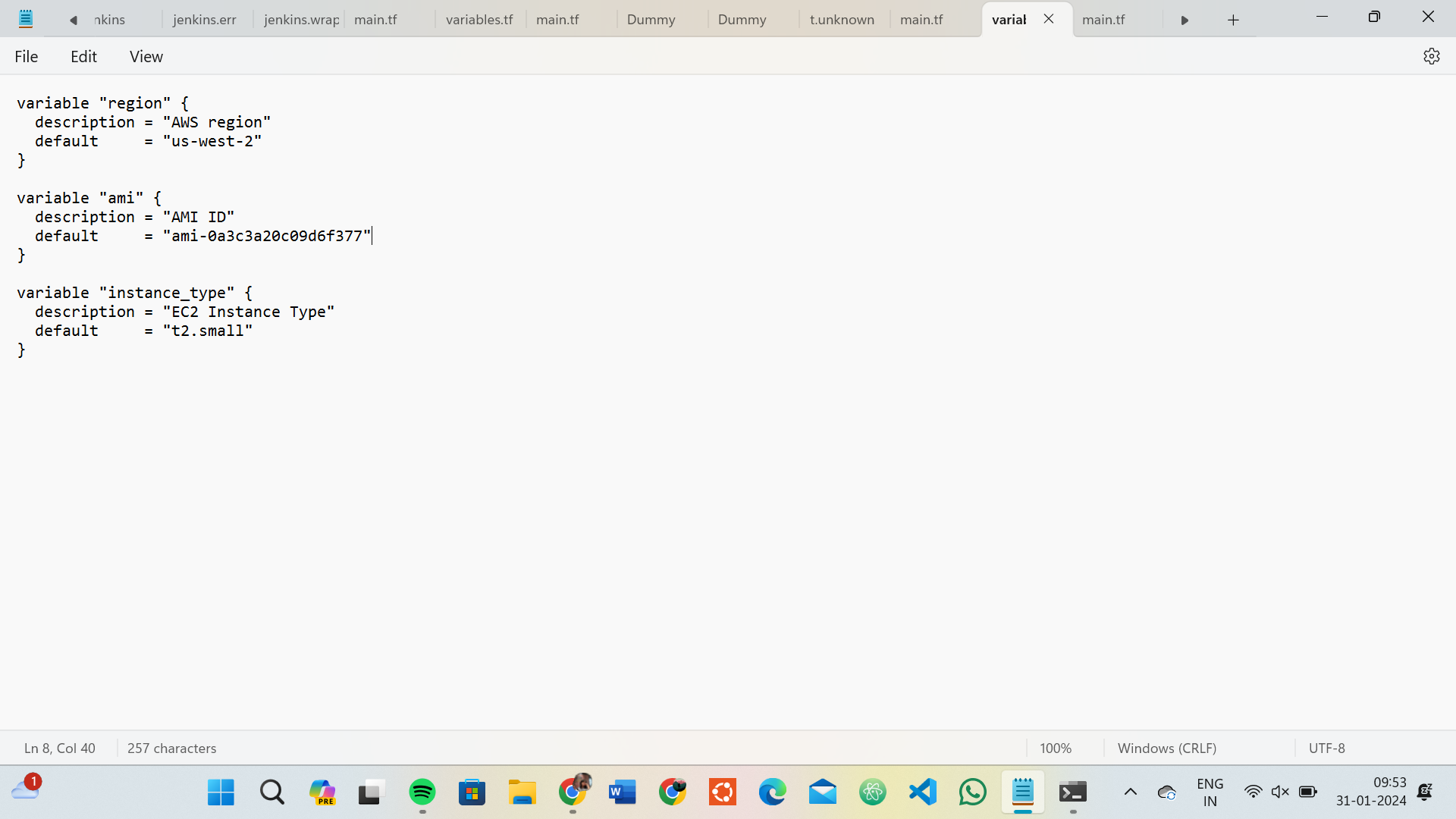
****

# Create Terraform Configuration Files:

* + Create a file named main.tf:



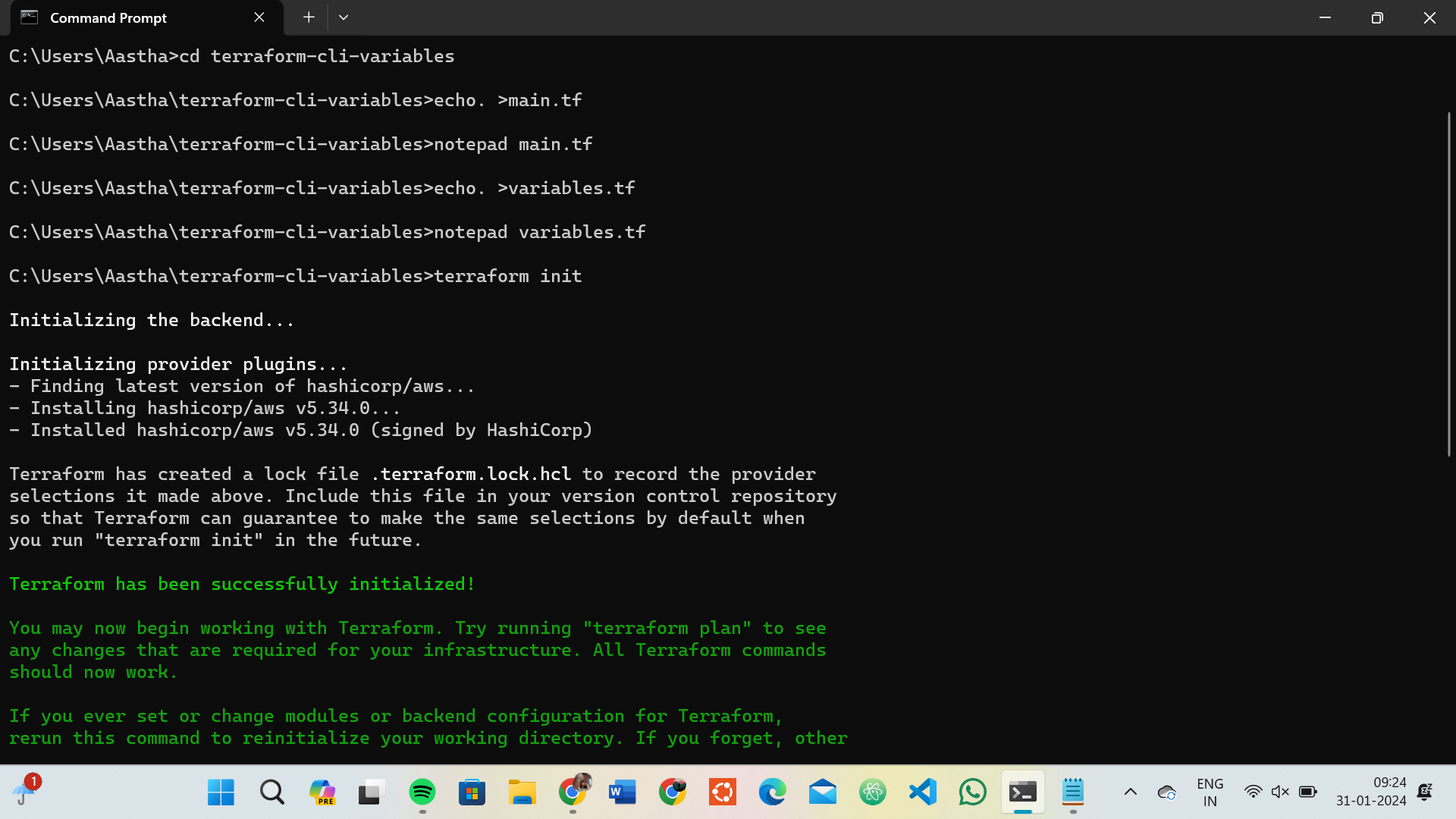
* + Create a file named variables.tf:



# Use Command Line Arguments:

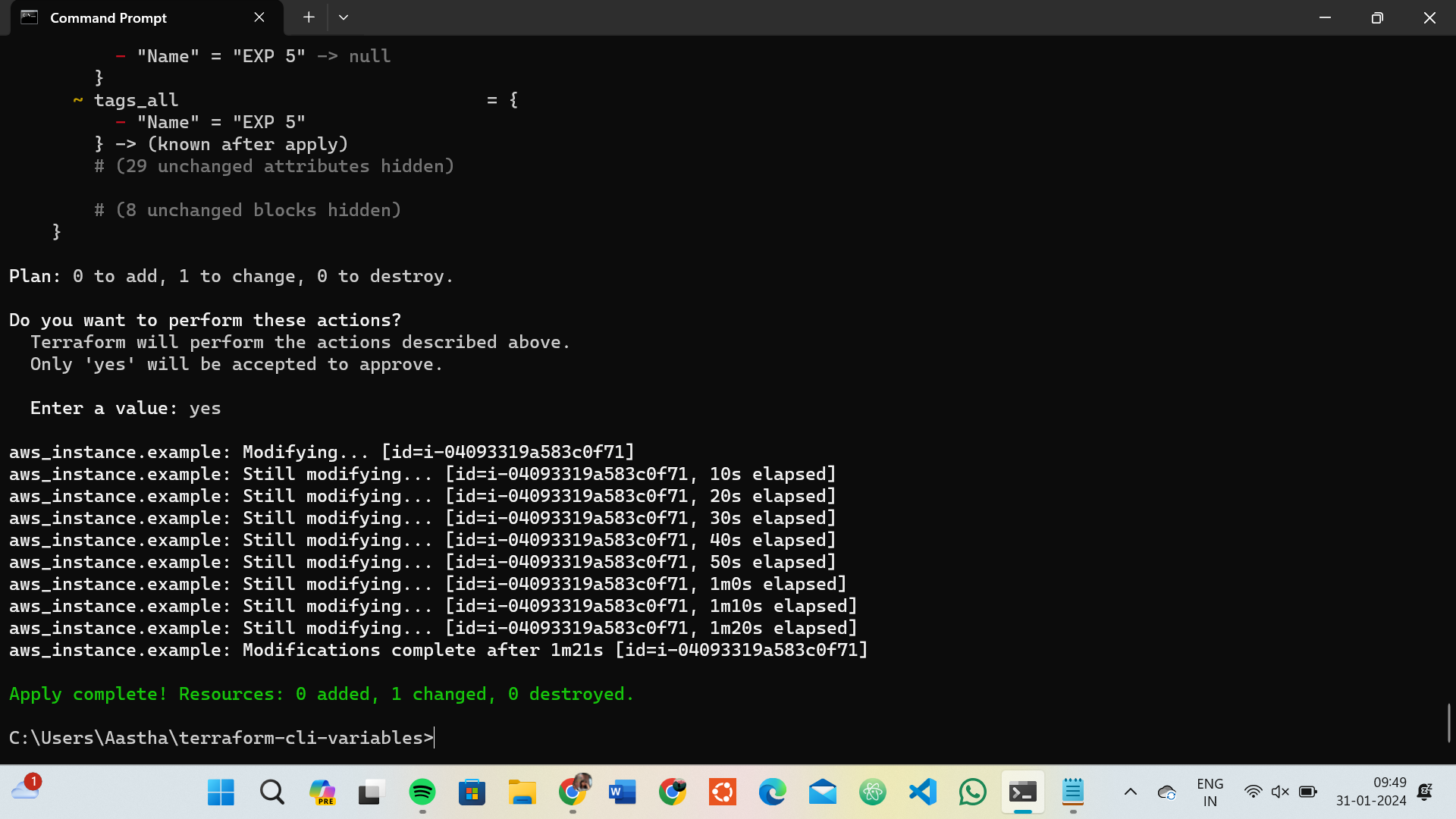
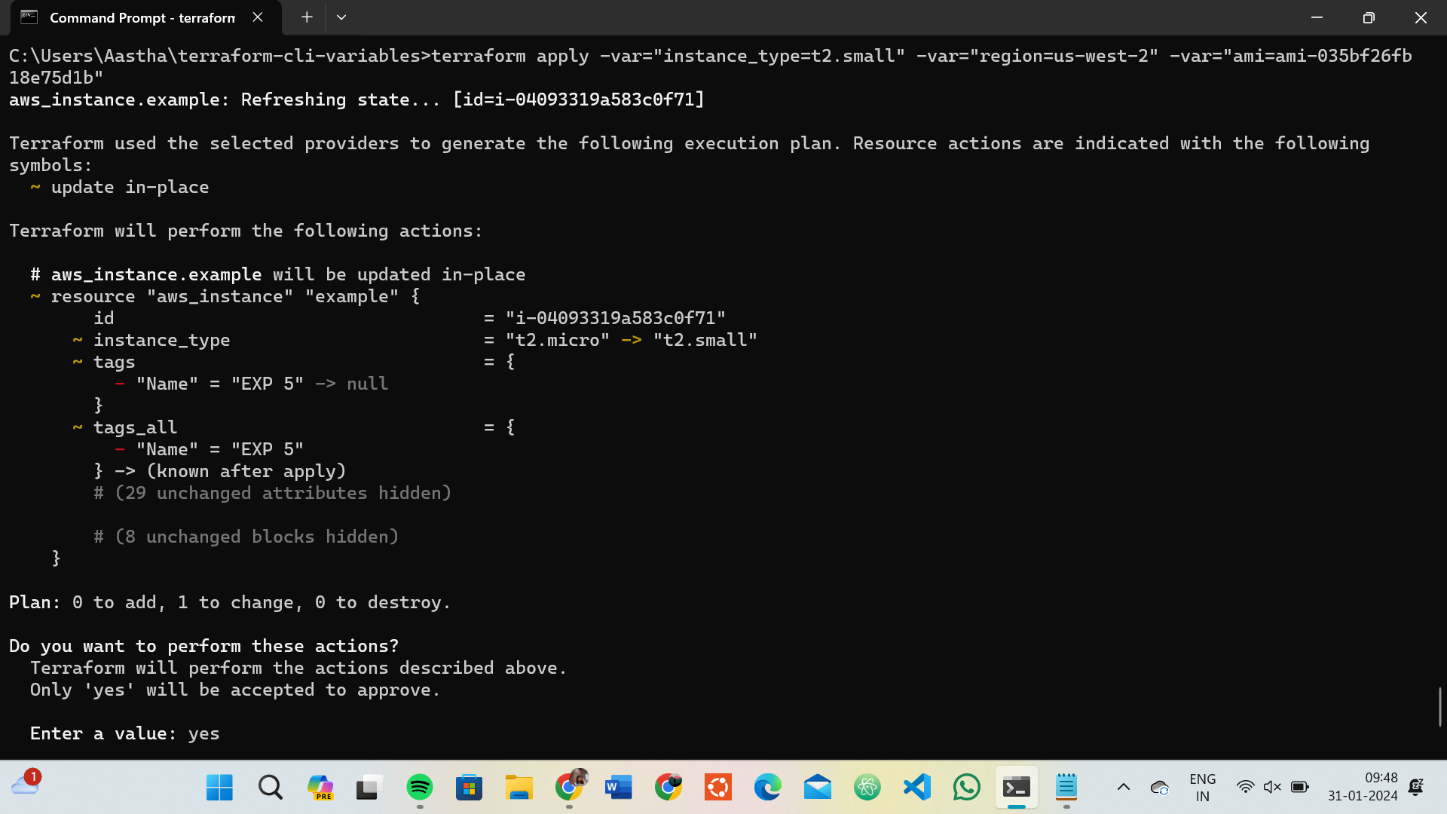
* + Open a terminal and navigate to your Terraform project directory.
  + Run the terraform init command:

**terraform init**



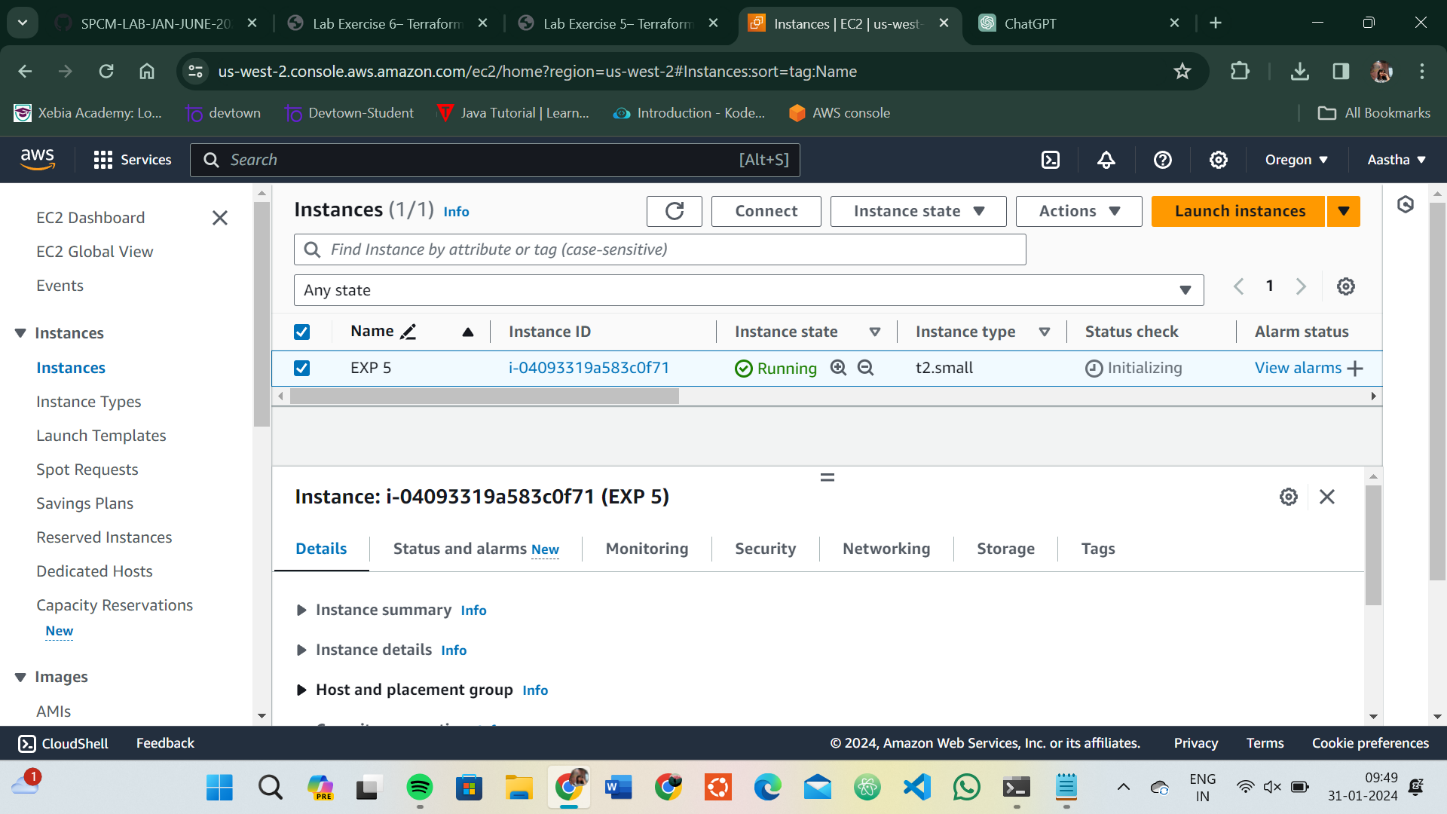
* + Run the terraform apply command with command line arguments to set variable values:

**terraform apply -var 'region=us-east-1' -var 'ami=ami-12345678' -var 'instance\_type=t2.micro**



# Test and Verify:

* + Observe how the command line arguments dynamically set the variable values during the apply process.
  + Access the AWS Management Console or use the AWS CLI to verify the creation of resources in the specified region.

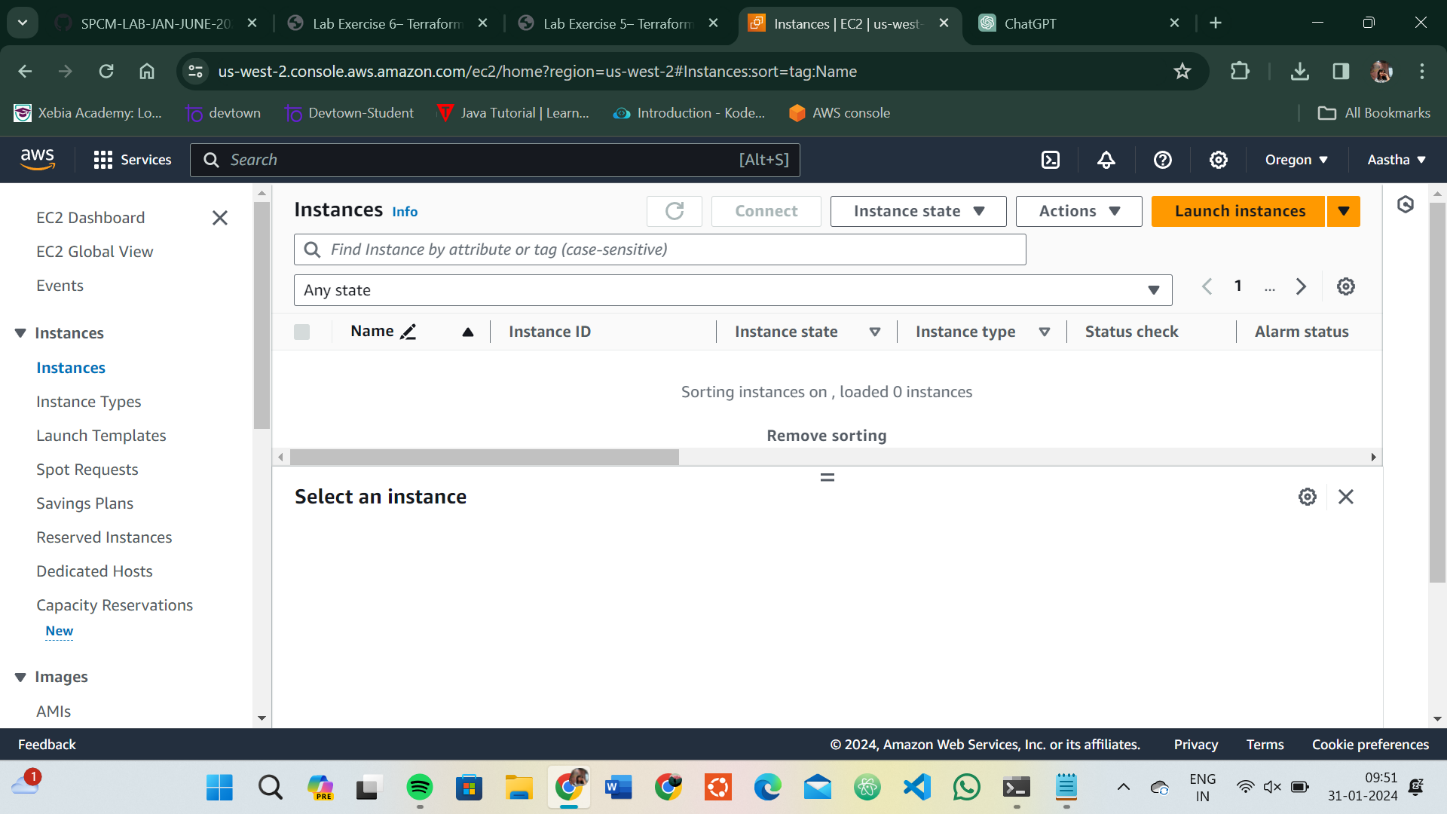
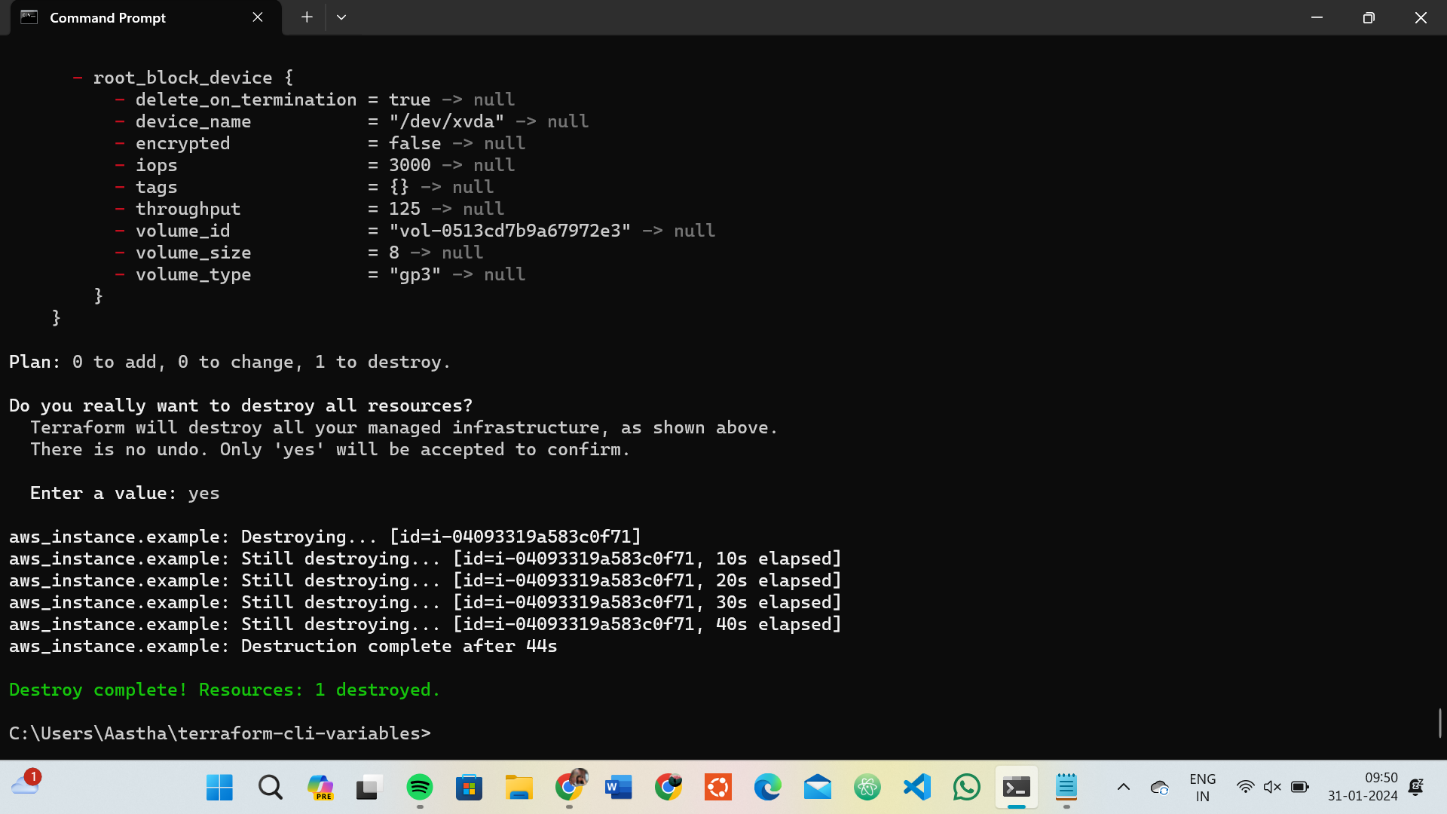


# Clean Up:

After testing, you can clean up resources:

**terraform destroy**

Confirm the destruction by typing yes.



# Conclusion:

This lab exercise demonstrates how to use command line arguments to set variable values dynamically during the terraform apply process. It allows you to customize your Terraform deployments without modifying the configuration files directly. Experiment with different variable values and observe how command line arguments impact the infrastructure provisioning process.

# Lab Exercise 6– Terraform Multiple tfvars Files Objective:

Learn how to use multiple tfvars files in Terraform for different environments.

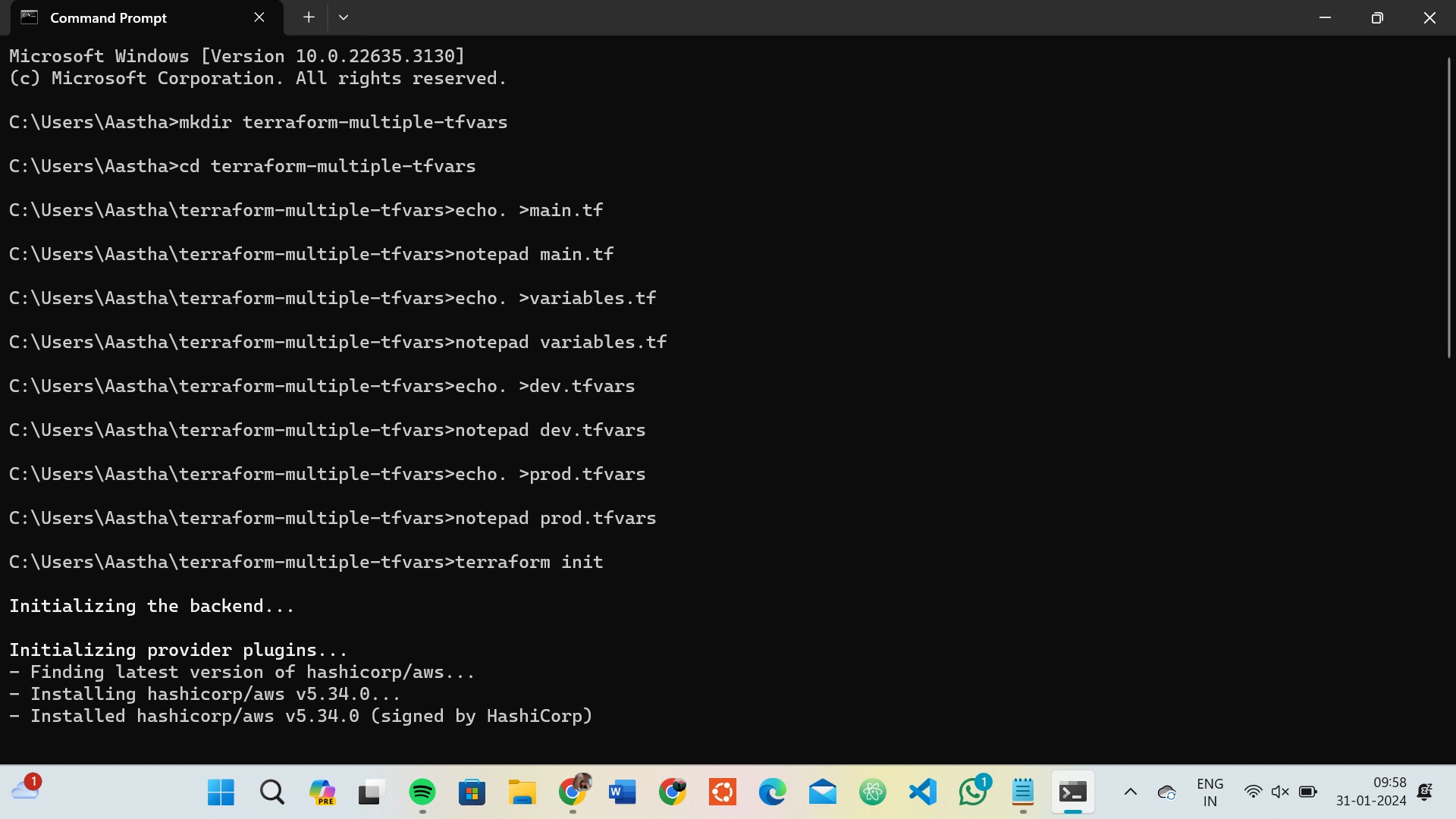
# Prerequisites:

* Terraform installed on your machine.
* Basic knowledge of Terraform configuration and variables.

# Steps:

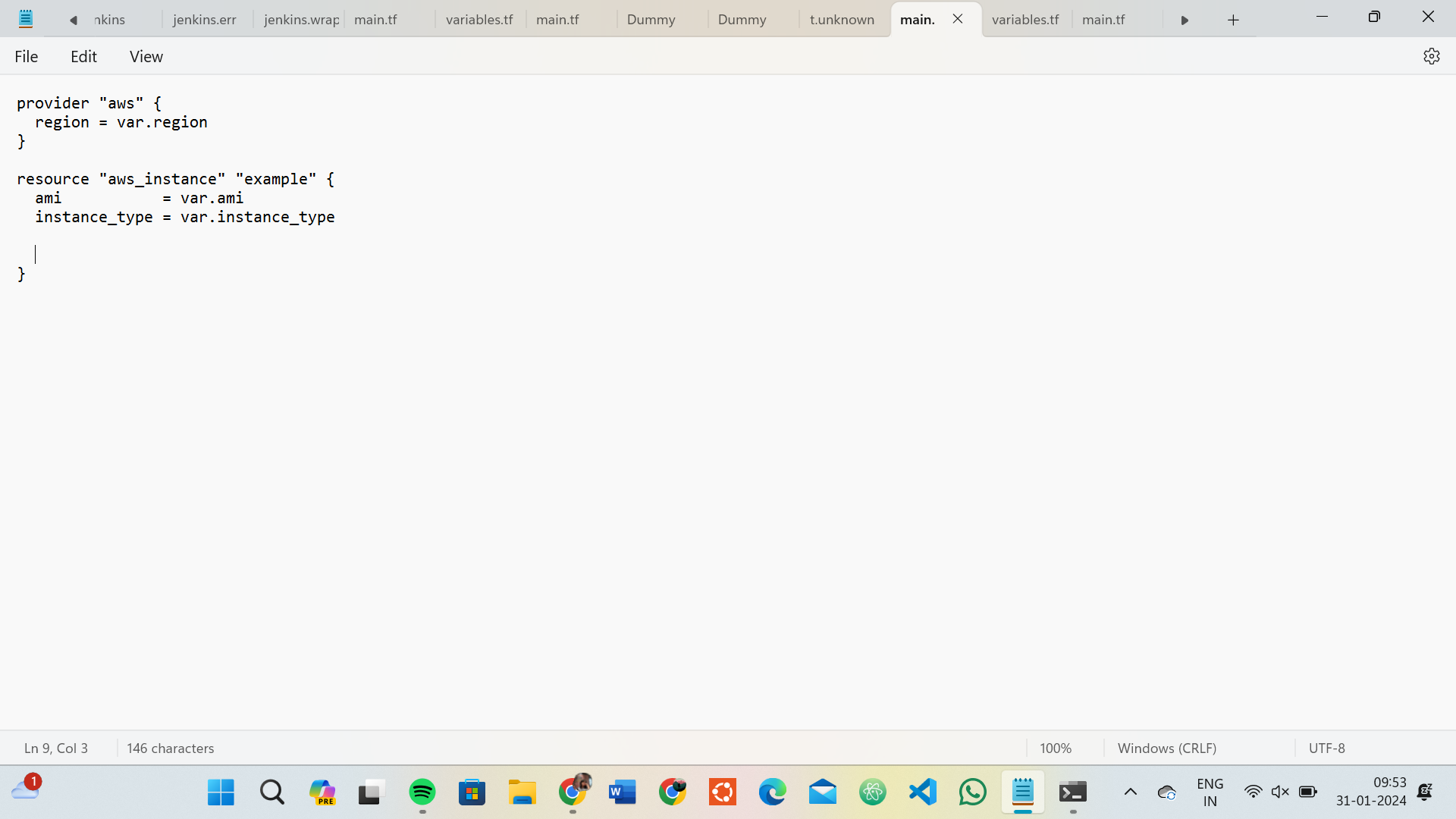
1. **Create a Terraform Directory:**

**mkdir terraform-multiple-tfvars cd terraform-multiple-tfvars**



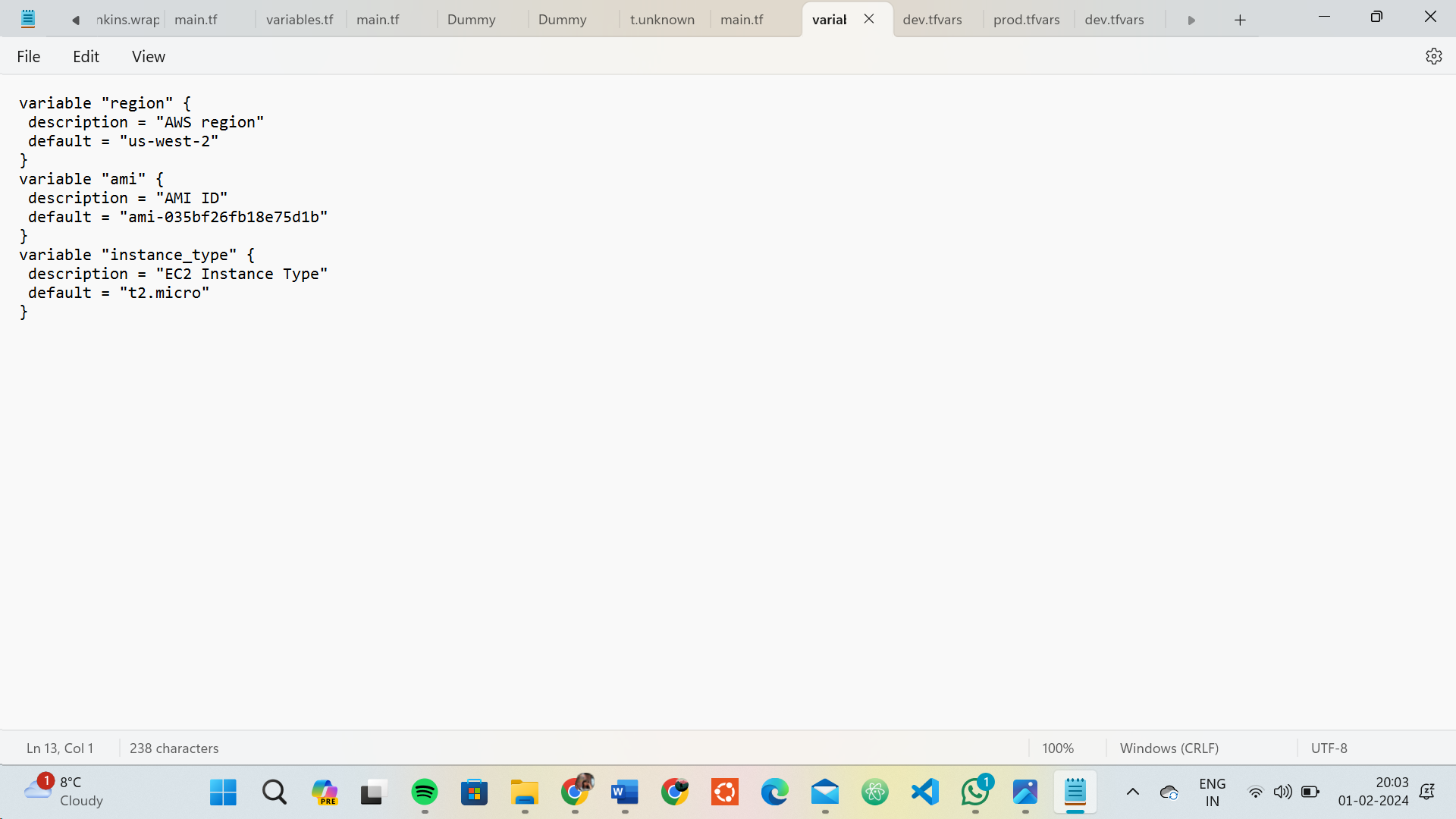
* + Create Terraform Configuration Files:
  + Create a file named main.tf:

## # main.tf



* + Create a file named variables.tf:

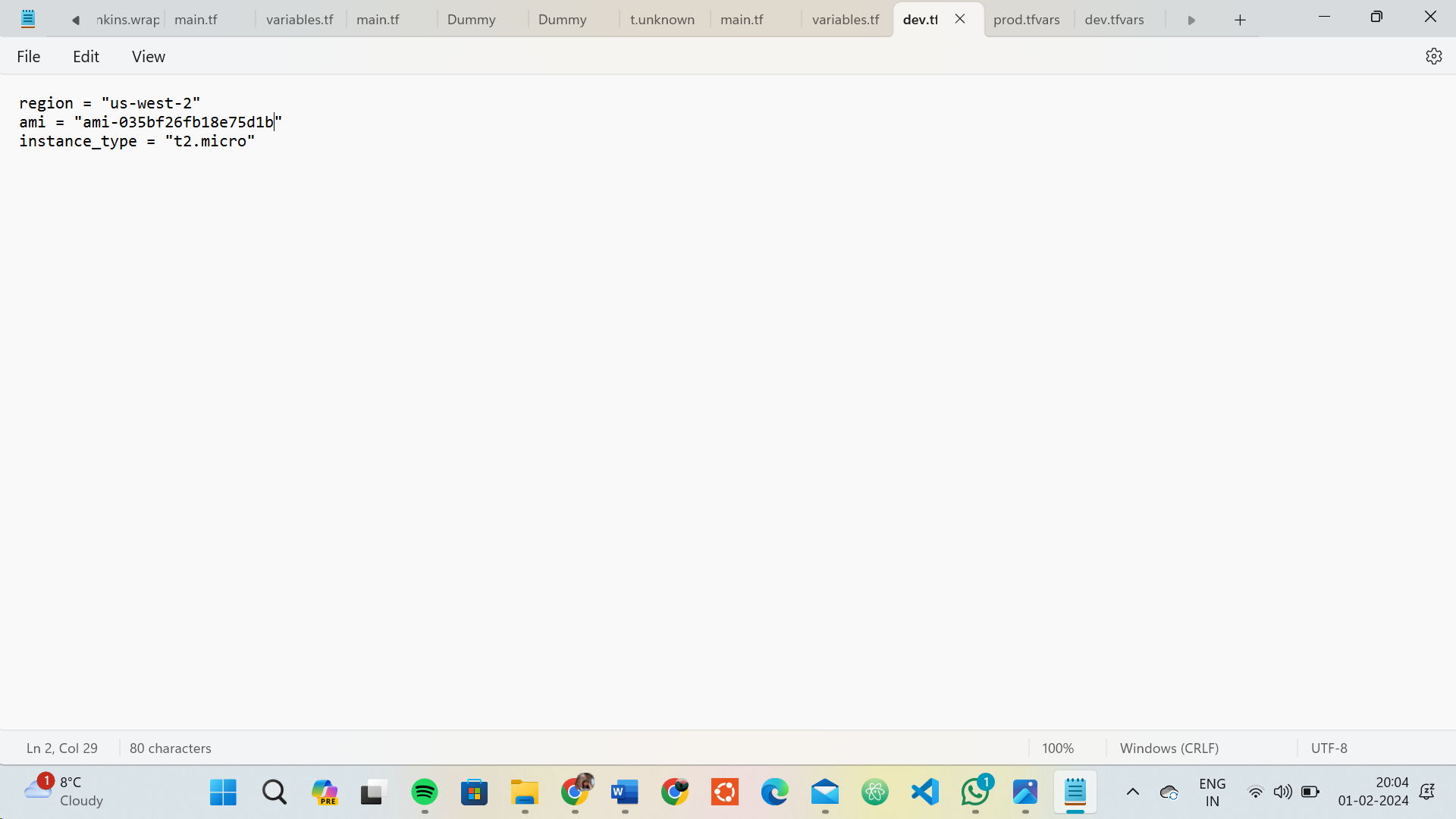
**# variables.tf**



# Create Multiple tfvars Files:

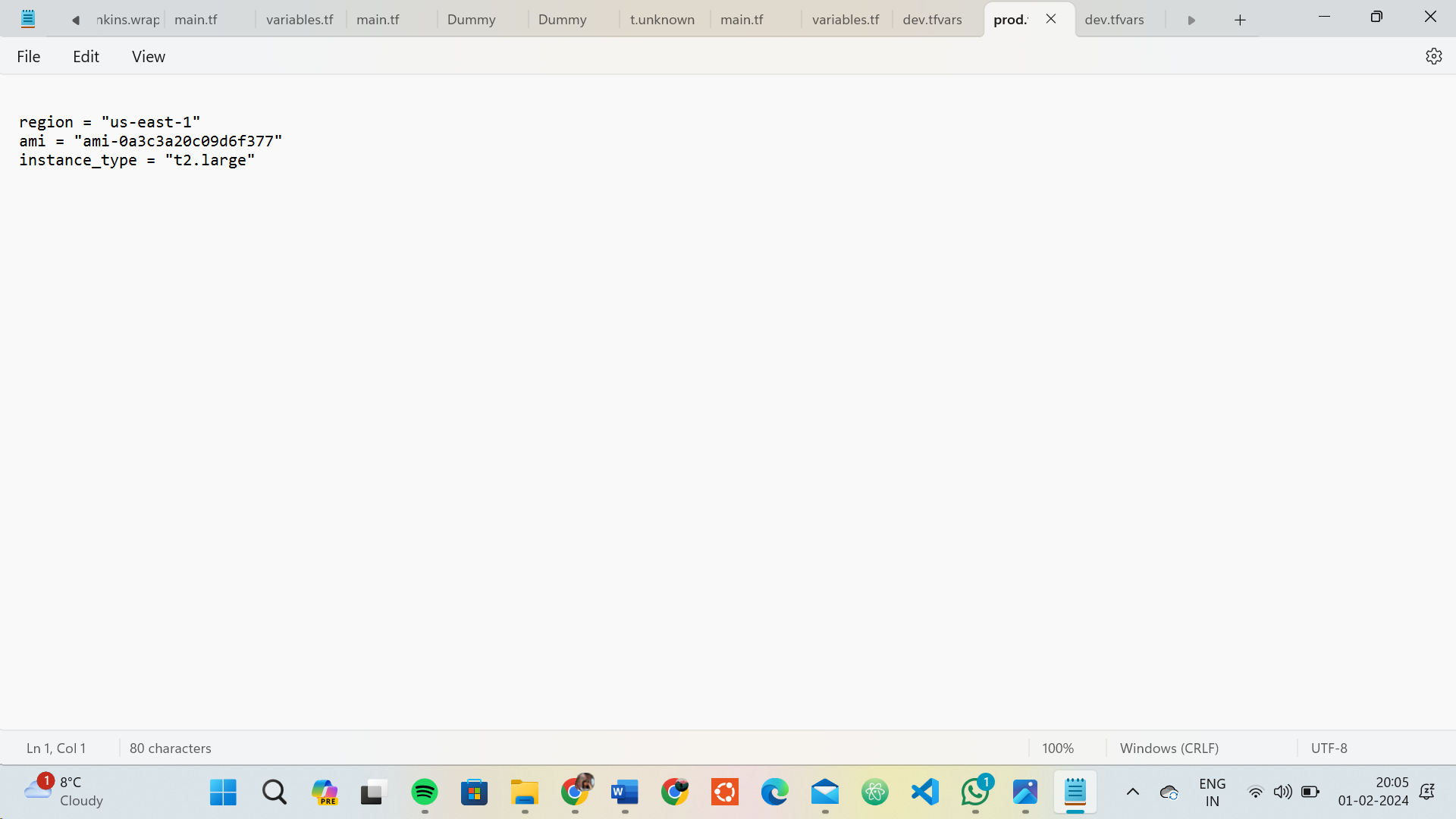
* Create a file named dev.tfvars:

## # dev.tfvars



* Create a file named prod.tfvars:

## # prod.tfvars

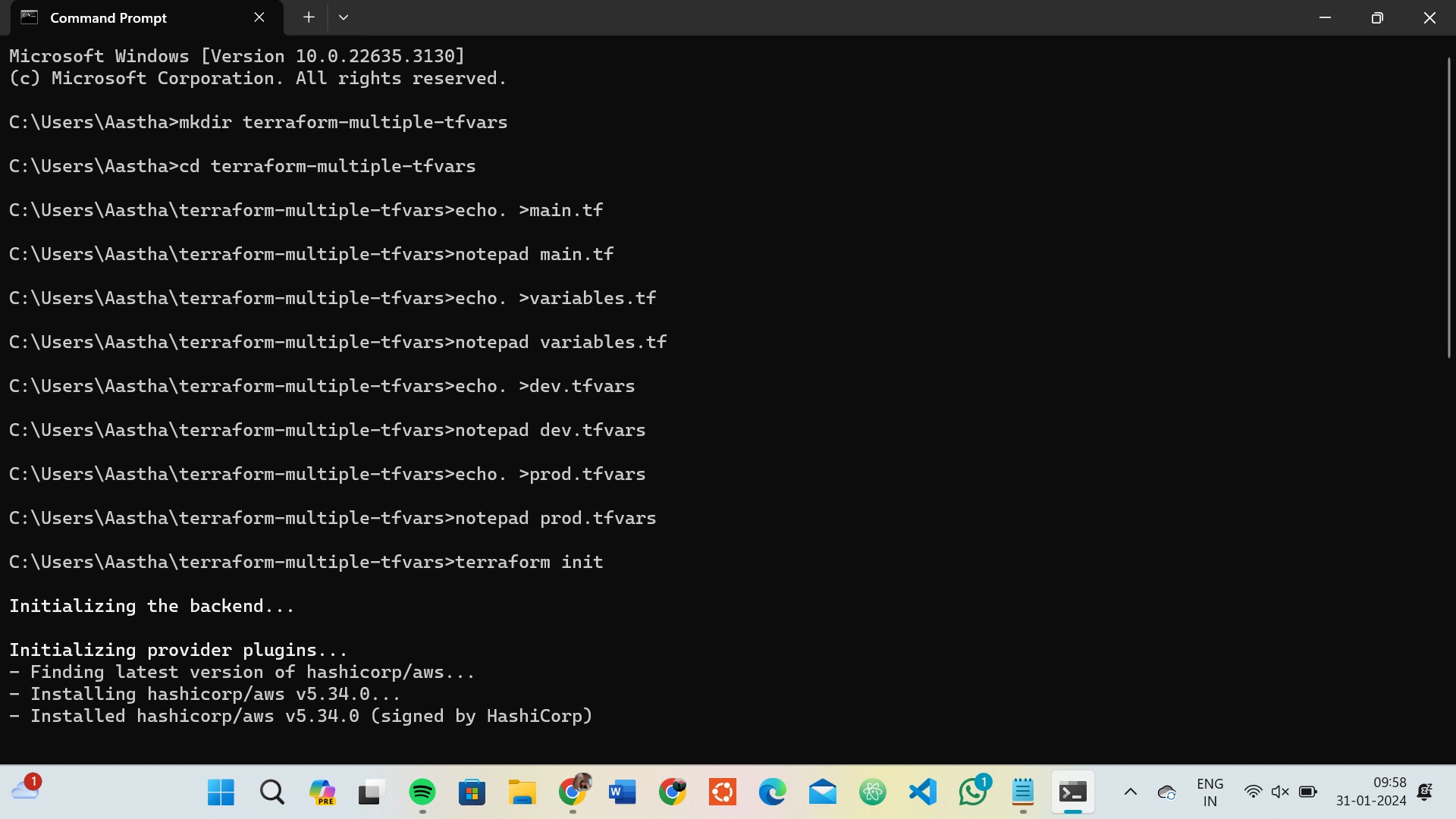


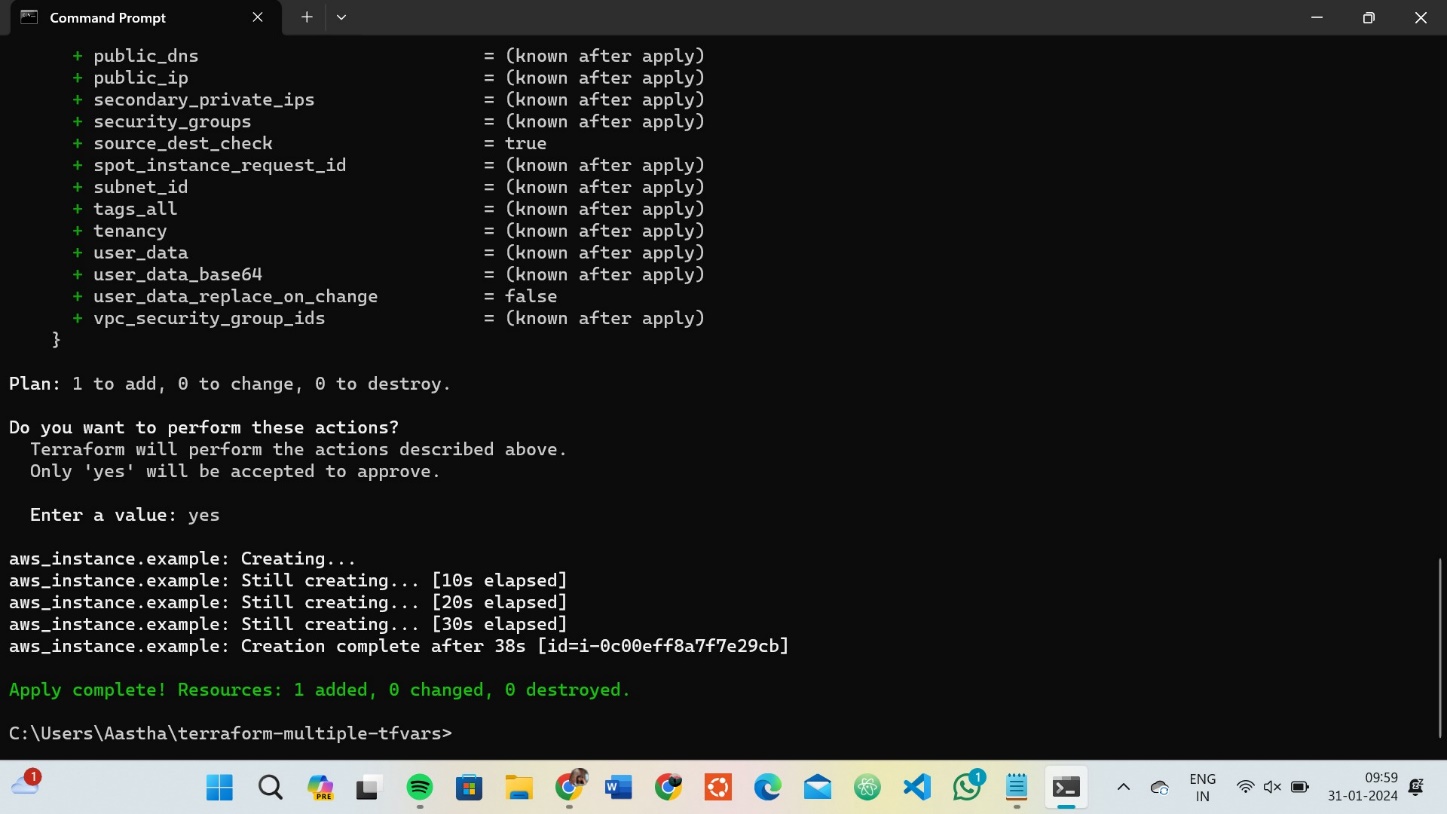
# Initialize and Apply for Prod Environment:

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

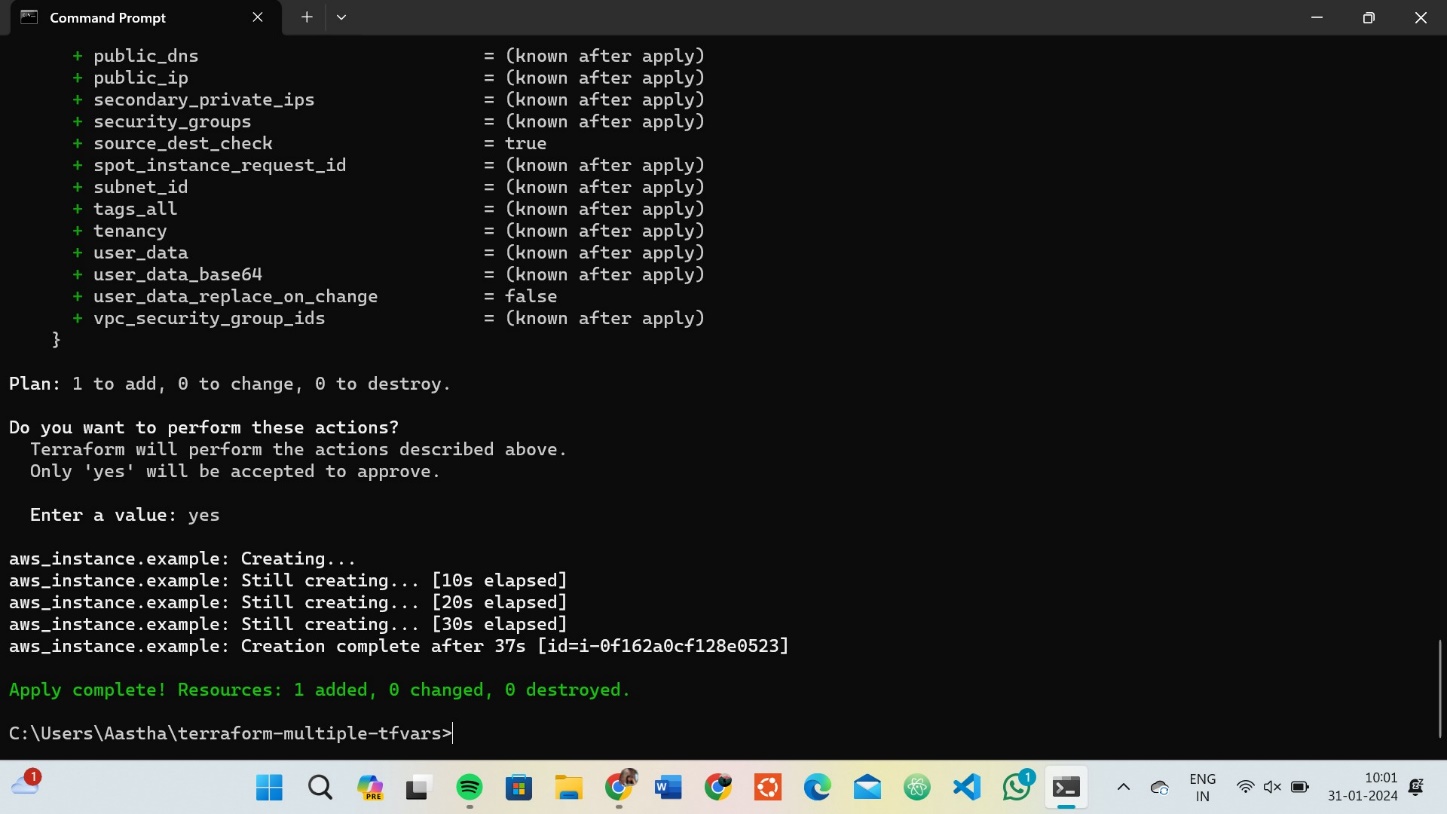
**terraform init**

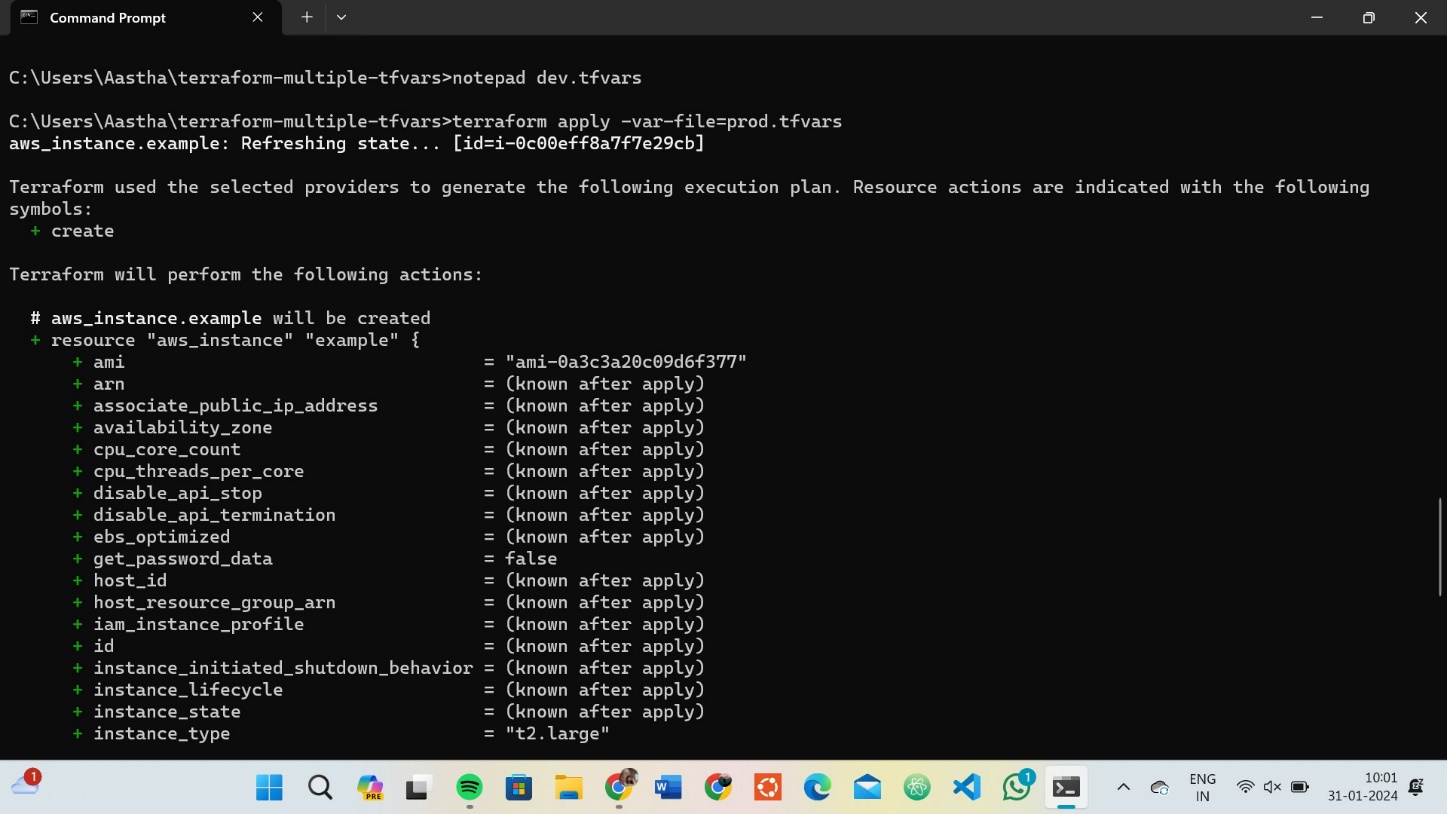
**terraform apply -var-file=prod.tfvars**





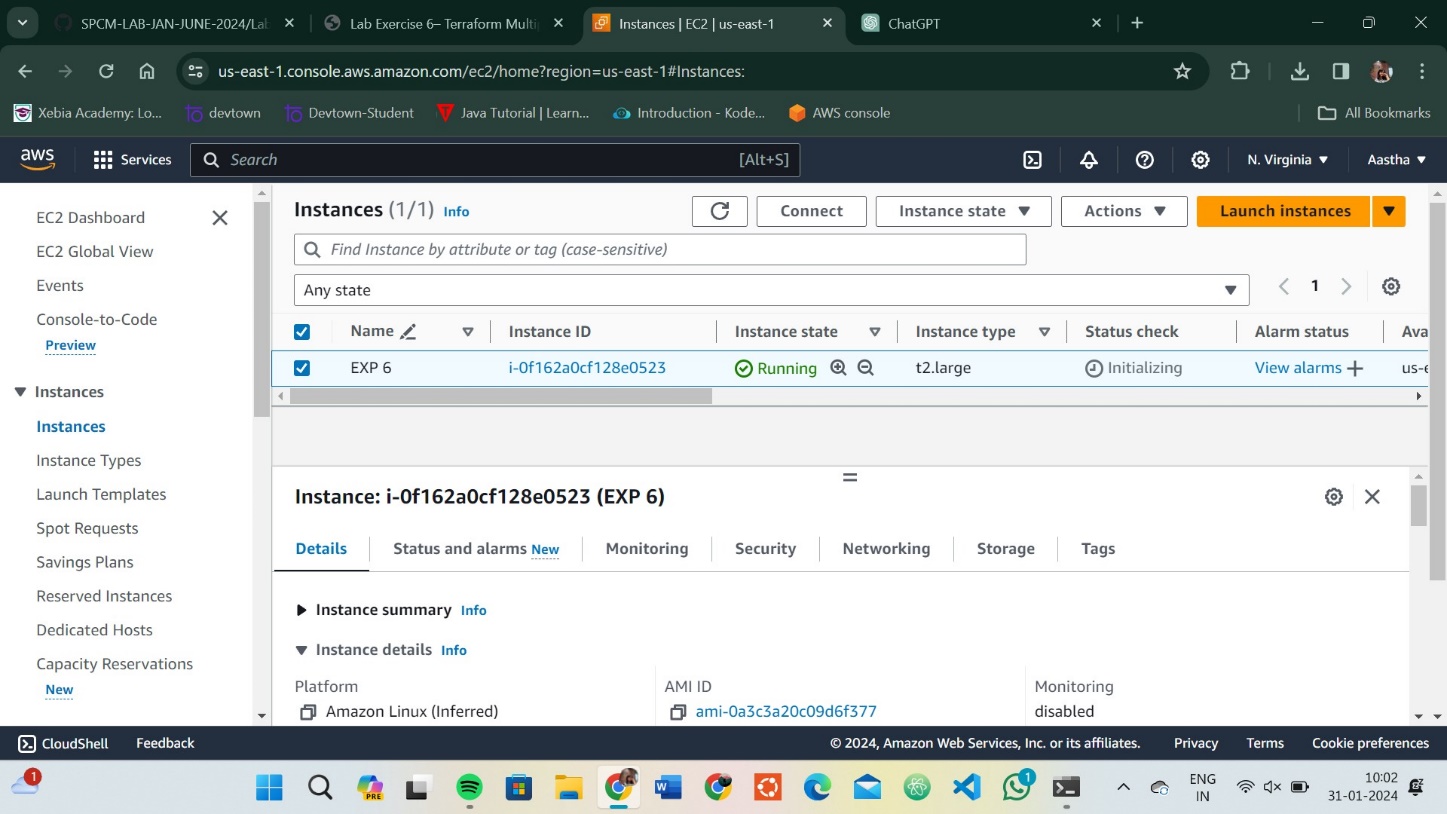






# Test and Verify:

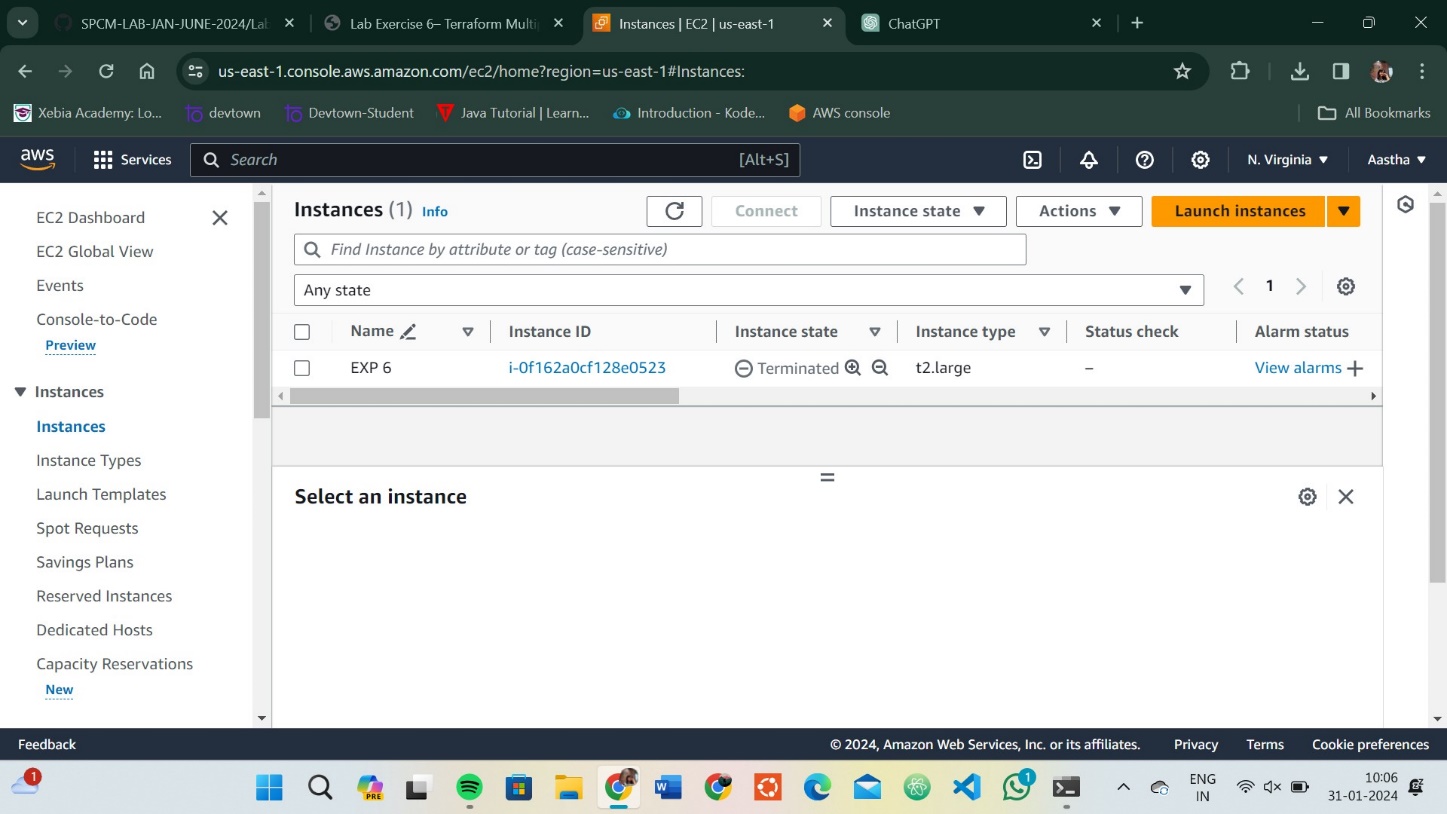
* Observe how different tfvars files are used to set variable values for different environments during the apply process.
* Access the AWS Management Console or use the AWS CLI to verify the creation of resources in the specified regions and instance types.



# Clean Up:

* + After testing, you can clean up resources:

**terraform destroy -var-file=dev.tfvars terraform destroy -var-file=prod.tfvars**



* + Confirm the destruction by typing yes.