Animesh Parab T2-T21 88

ASSIGNMENT - 06

AIM- To understand terraform lifecycle and to Build, change, and destroy AWS infrastructure Using Terraform.

LO MAPPED – LO1,LO3

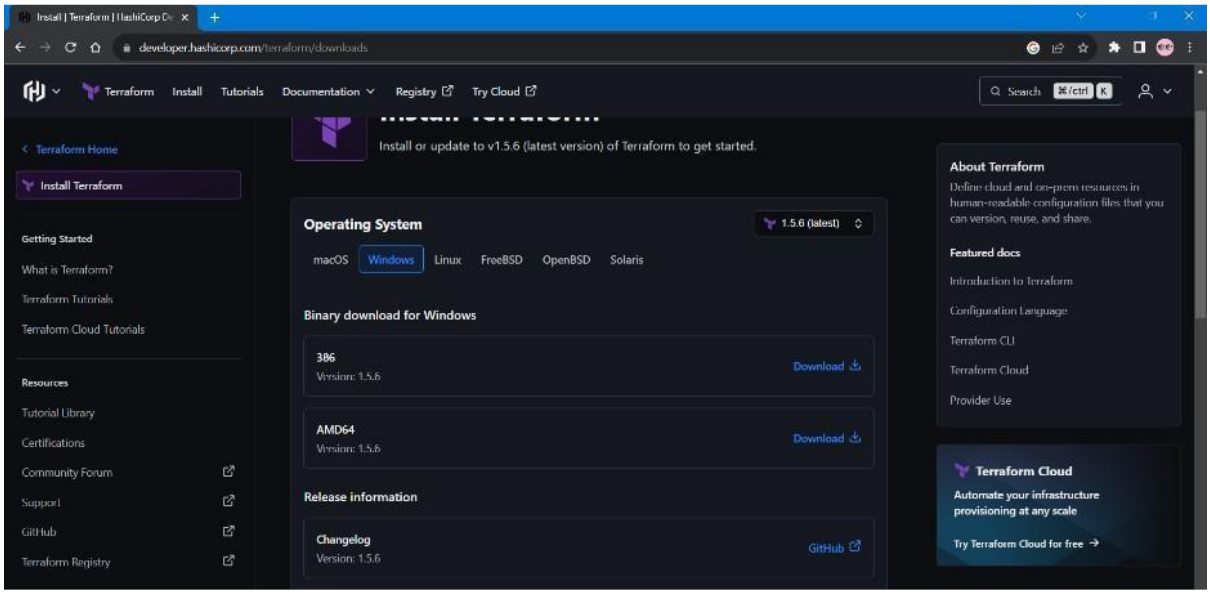
THEORY-
Terraform is a tool for building, changing, and versioning infrastructure safely and efficiently. Terraform can manage existing and popular service providers as well as custom in-house solutions.
The key features of Terraform are:
• **Infrastructure as Code**: Infrastructure is described using a high-level configuration syntax. This allows a blueprint of your datacenter to be versioned and treated as you would any other code. Additionally, infrastructure can be shared and re-used.

• **Execution Plans**: Terraform has a "planning" step where it generates an execution plan. The execution plan shows what Terraform will do when you call apply. This lets you avoid any surprises when Terraform manipulates infrastructure.

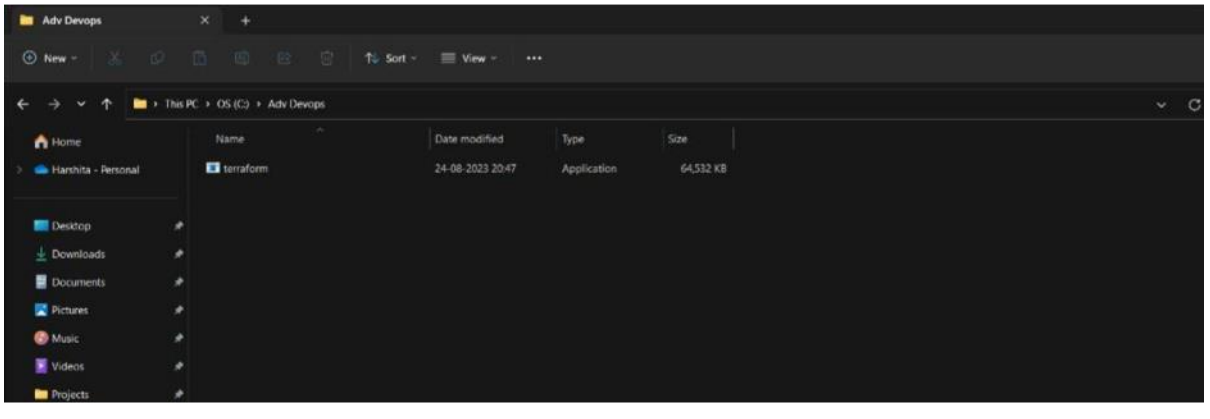
• **Resource Graph**: Terraform builds a graph of all your resources, and parallelizes the creation and modification of any non-dependent resources. Because of this, Terraform

builds infrastructure as efficiently as possible, and operators get insight into dependencies in their infrastructure.

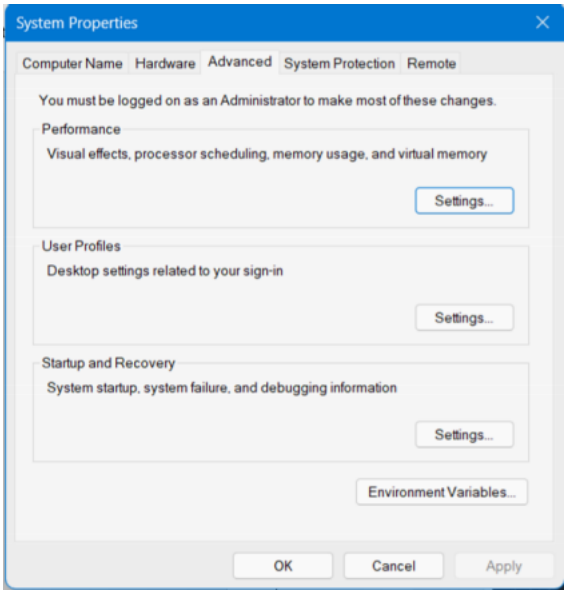
• **Change Automation**: Complex changesets can be applied to your infrastructure with minimal human interaction. With the previously mentioned execution plan and resource graph, you know exactly what Terraform will change and in what order, avoiding many possible human errors.
STEPS-
Google Search – Terraform DOWNLOAD



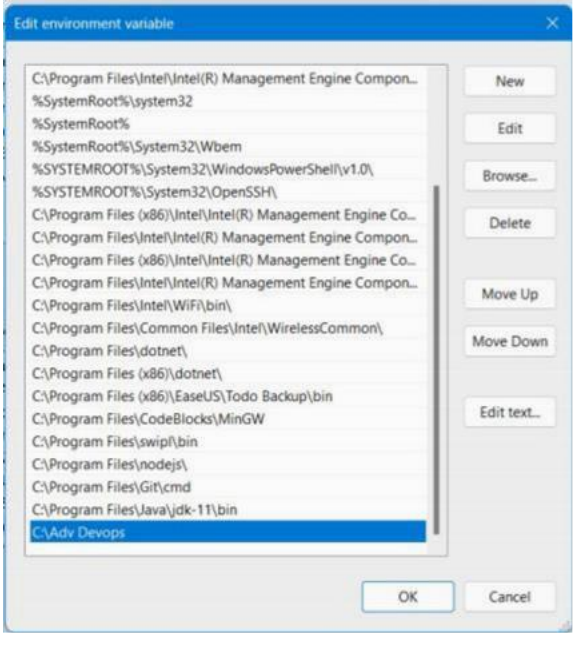
Create a Folder in Your C drive Named AdvDevops
Then Extract The downloaded file in The C drive Folder Named AdvDevops Shown Below



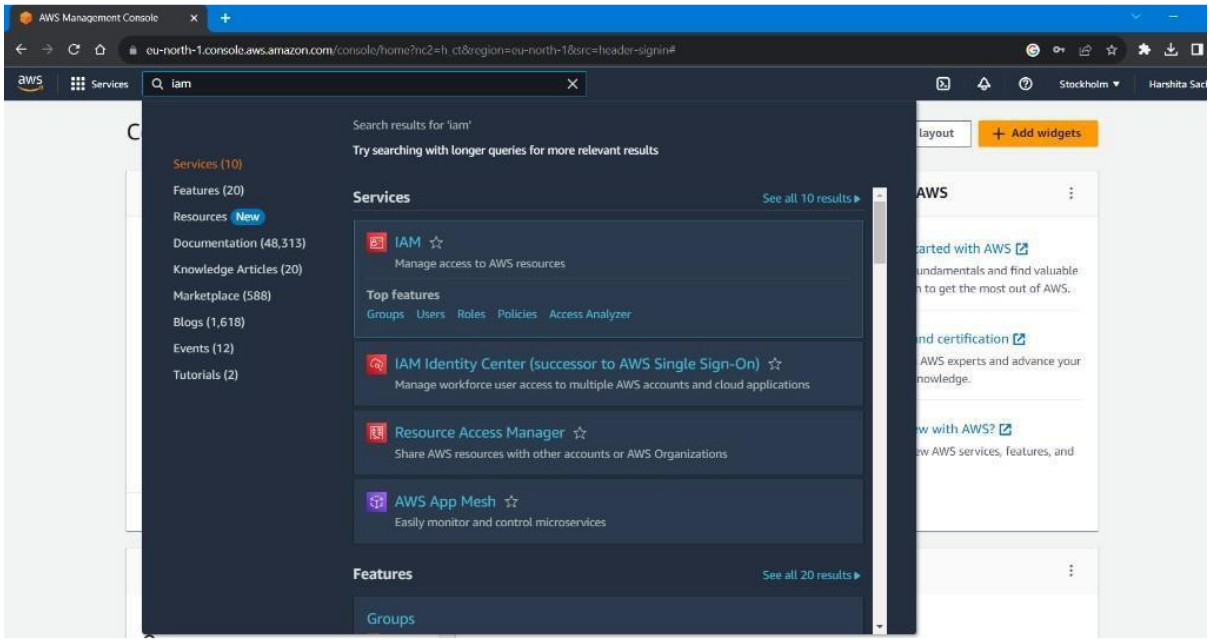
Now search EDIT THE SYSTEM ENVIRONMENT VARIABLES in your windows se



Now click on PATH OF USER VARIBLES, then click on Edit option
Now go to edit and then add new path C:\AdvDevOps



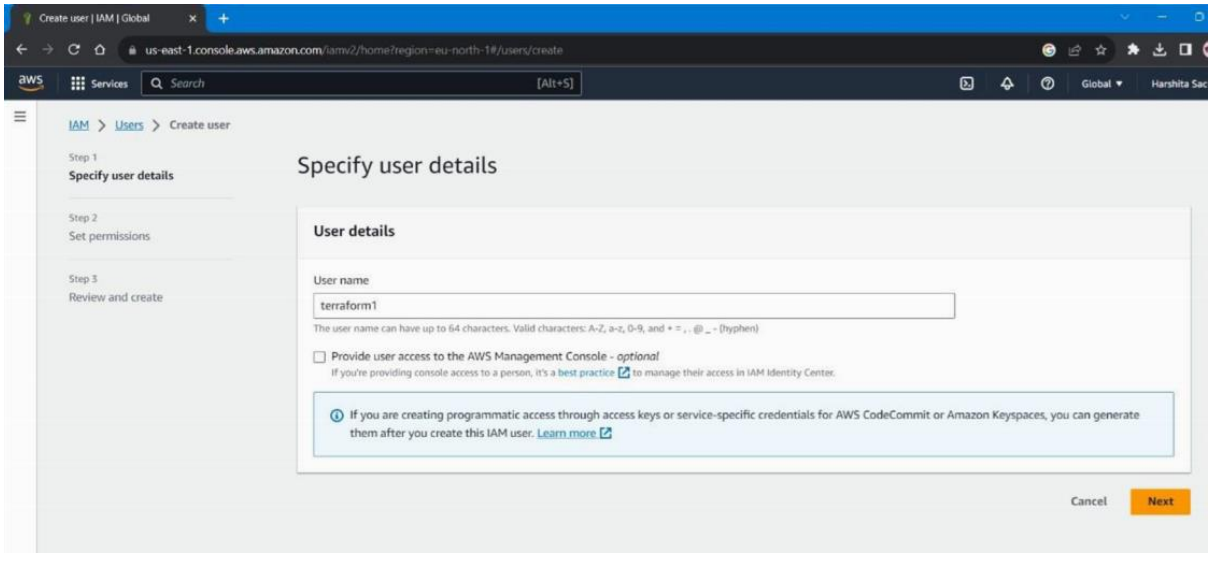
Repeat same procedure for system variables.

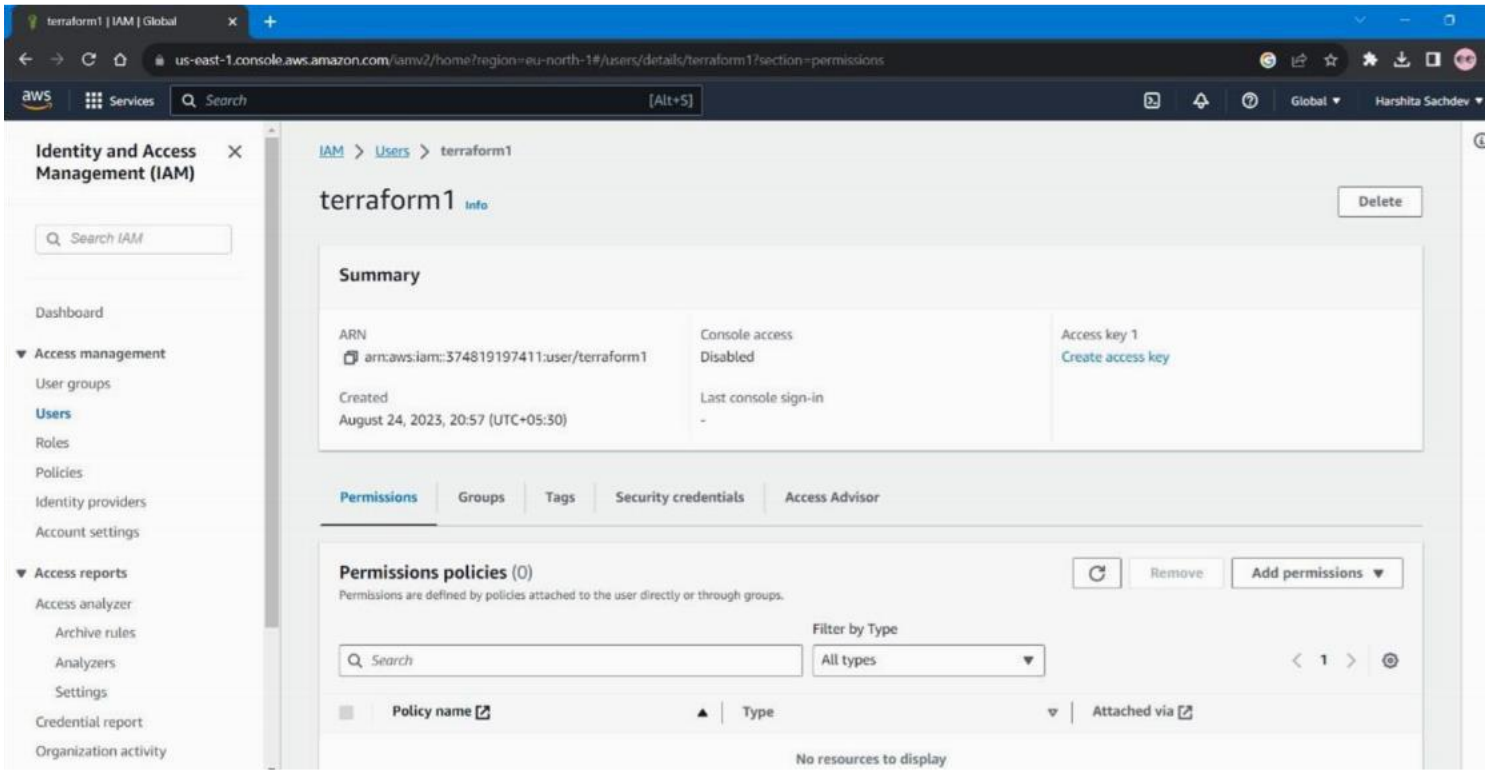


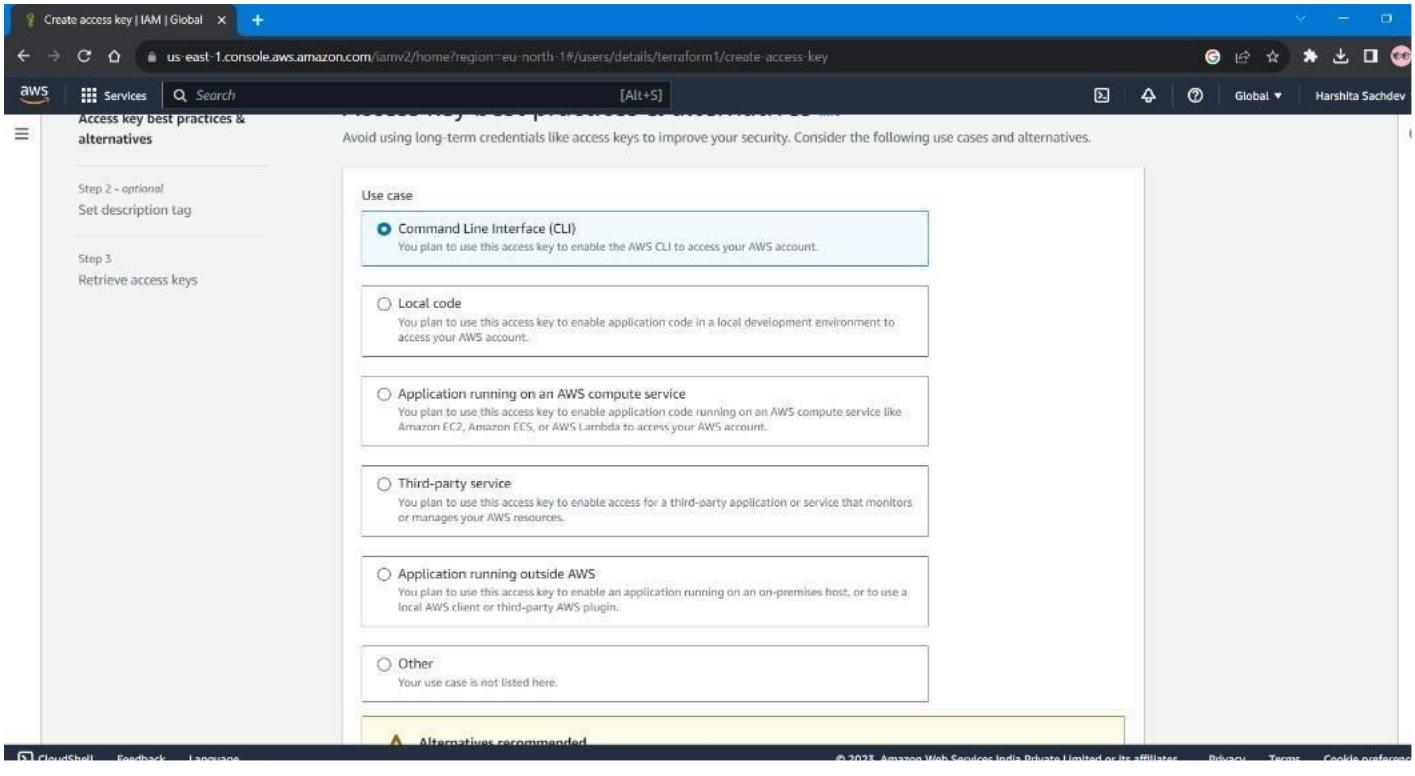
Open And Login to your AWS console- And search IAM and click on it

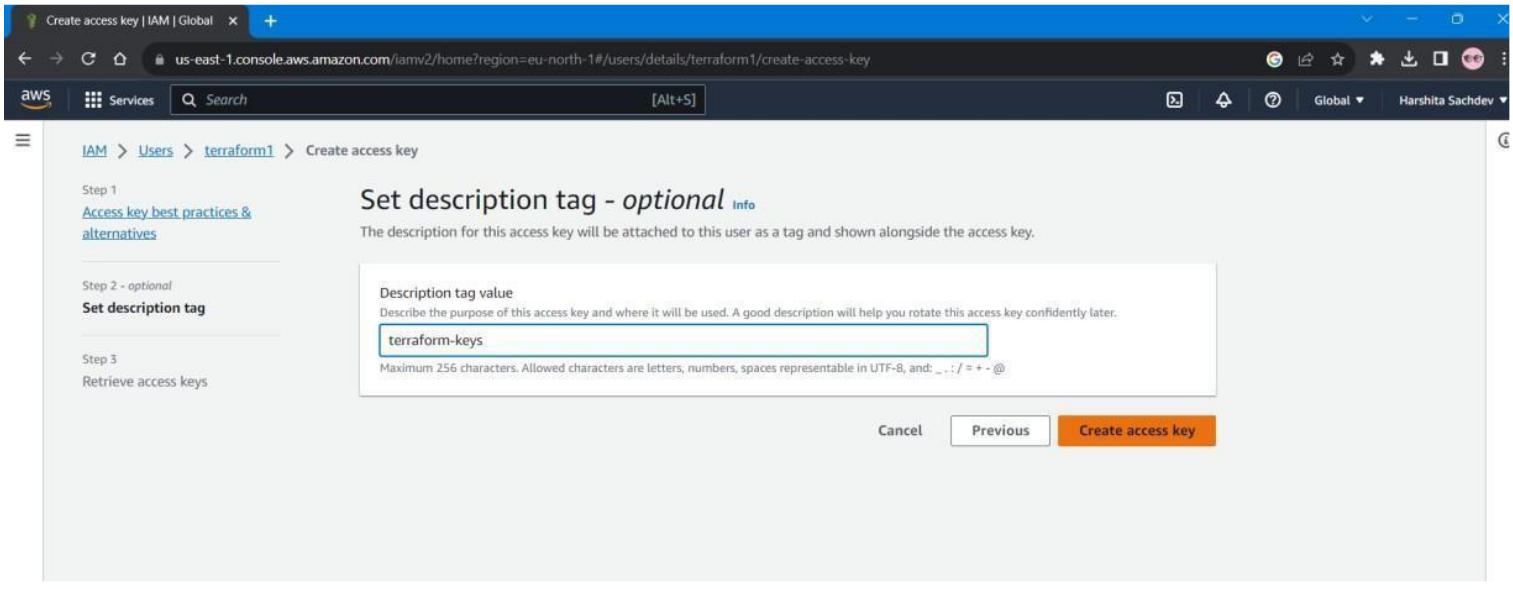
Now click on Add Users in The User Section as shown in the image and add the user

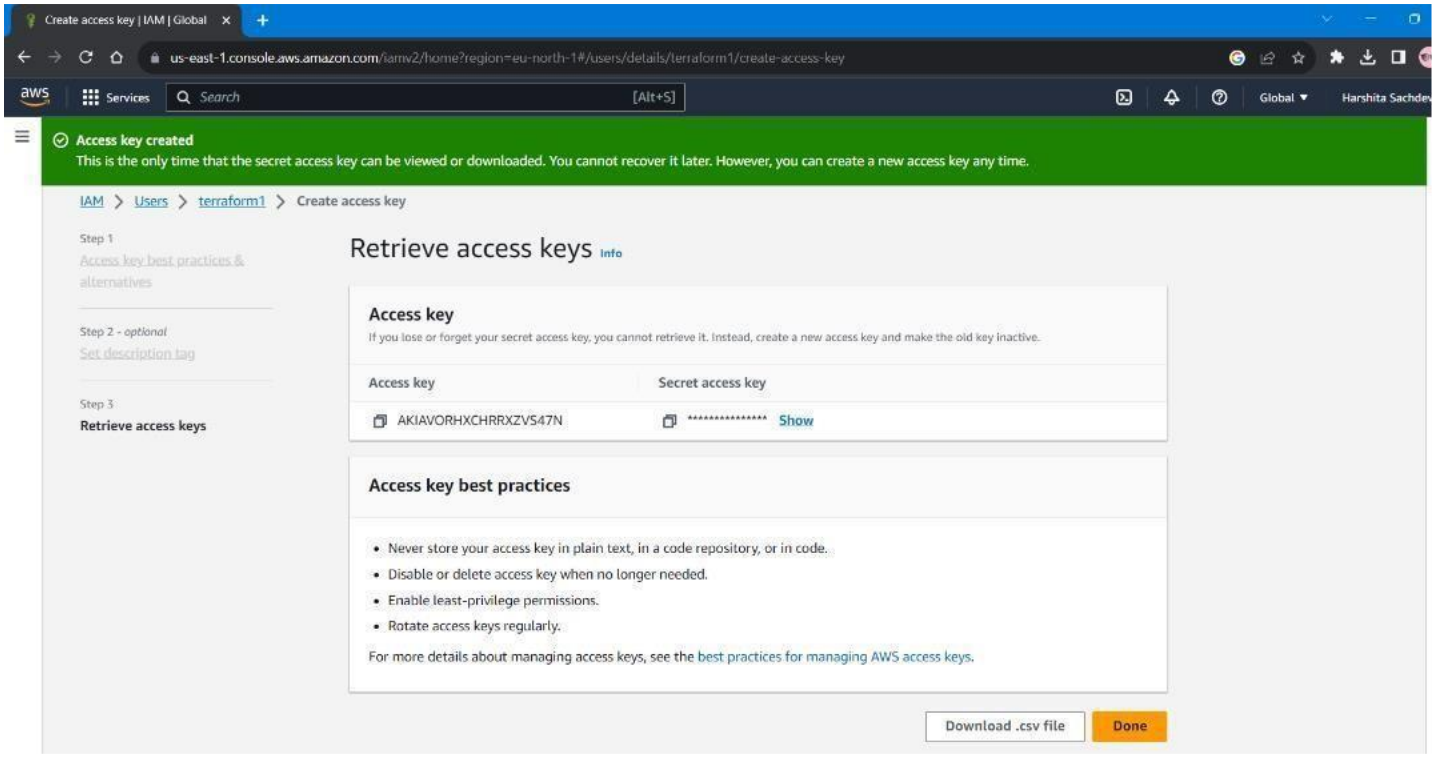
name

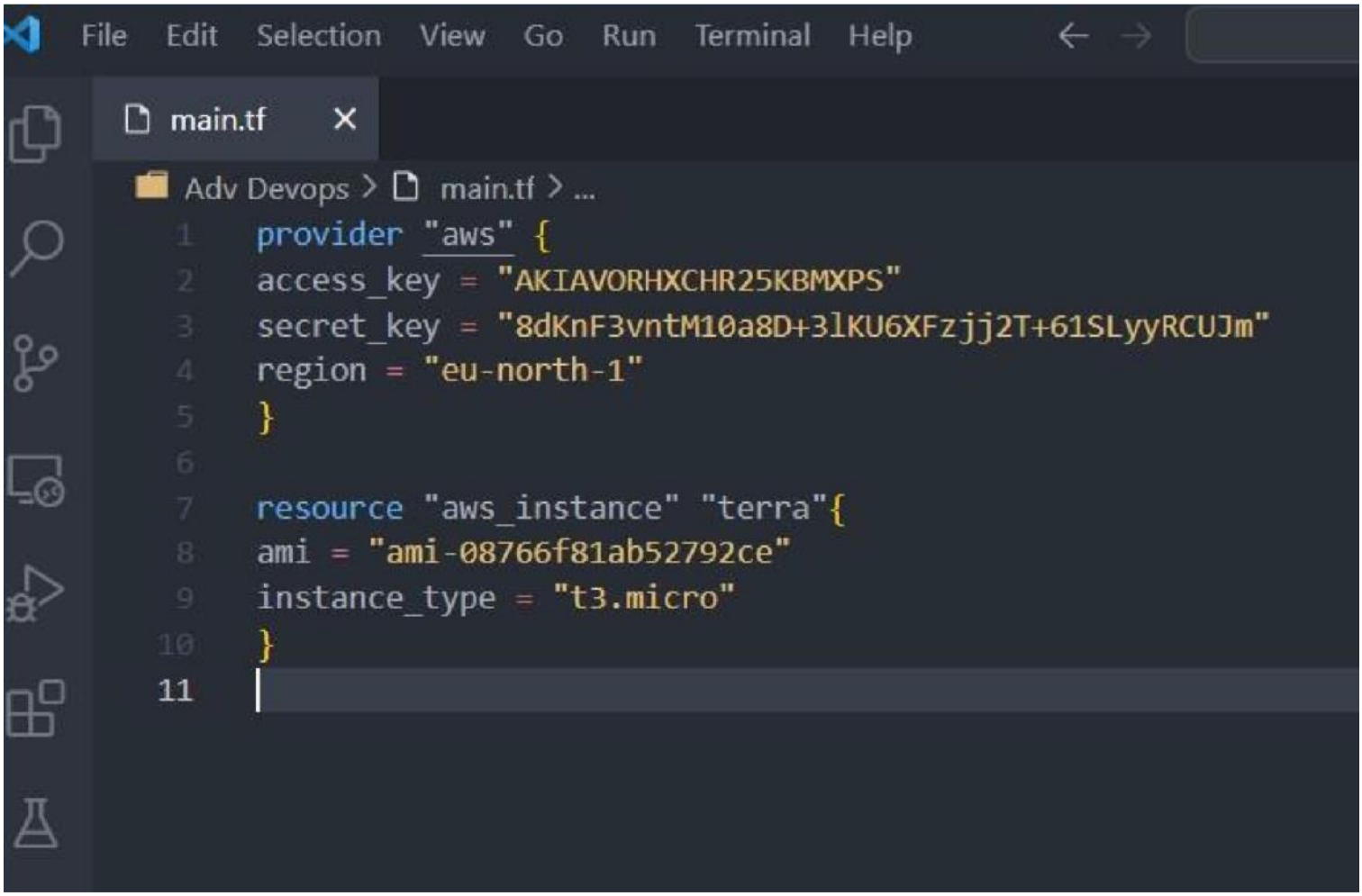


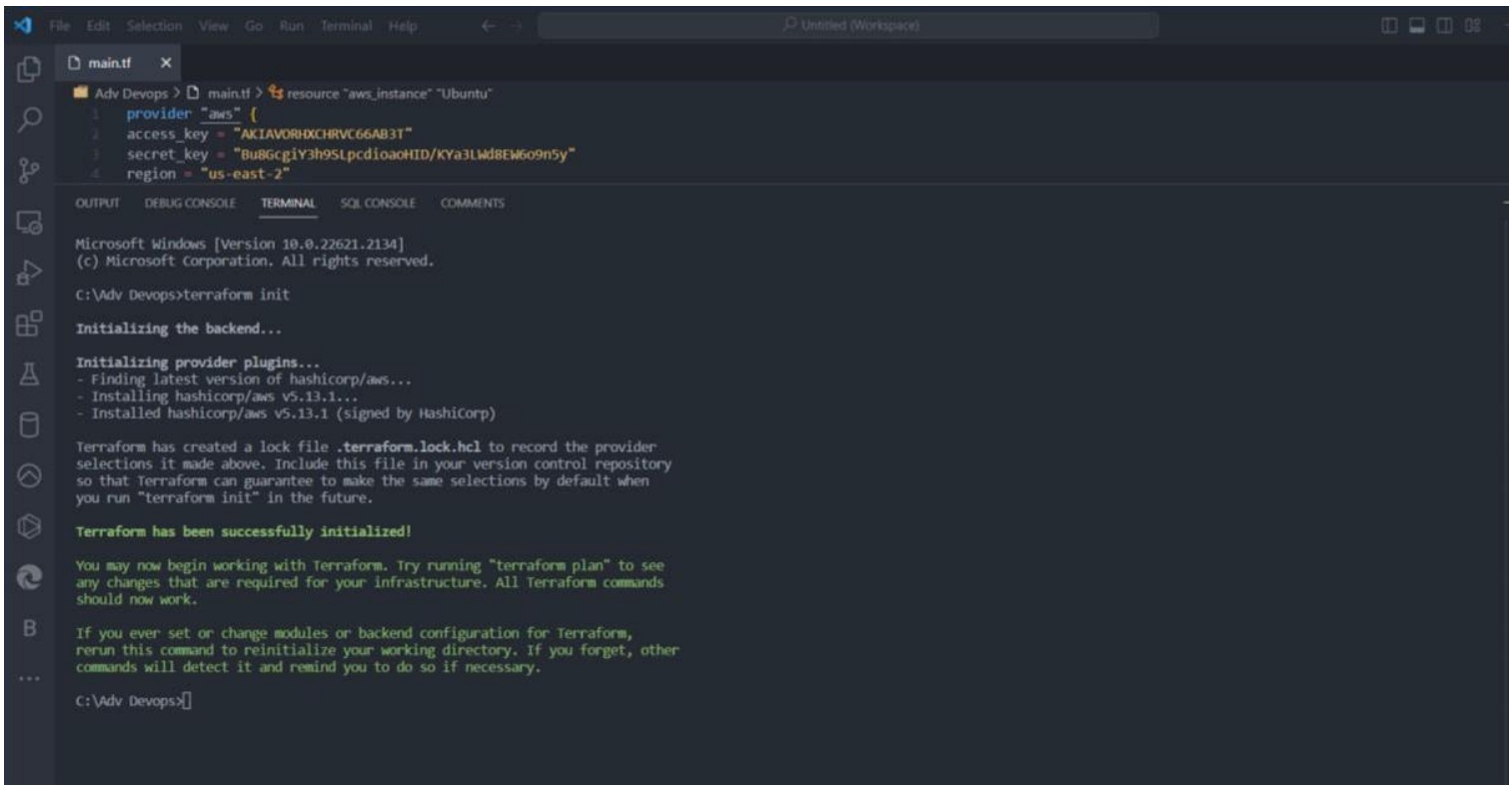

terraform1 | IAM | Global
us-east-1.console.aws.amazon.com/iamv2/home?region=eu-north-1#/users/details/terraform1?section=permissions
aws
Services
Search
[Alt+S]
Global
Harshita Sachdev
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access analyzer
Archive rules
Analyzers
Settings
Credential report
Organization activity
IAM > Users > terraform1
terraform1 Info
Delete
Summary
ARN
arn:aws:iam::374819197411:user/terraform1
Console access
Disabled
Access key 1
Create access key
Created
August 24, 2023, 20:57 (UTC+05:30)
Last console sign-in
-
Permissions
Groups
Tags
Security credentials
Access Advisor
Permissions policies (0)
Permissions are defined by policies attached to the user directly or through groups.
Remove
Add permissions
Filter by Type
Search
All types
1
Policy name
Type
Attached via
No resources to display

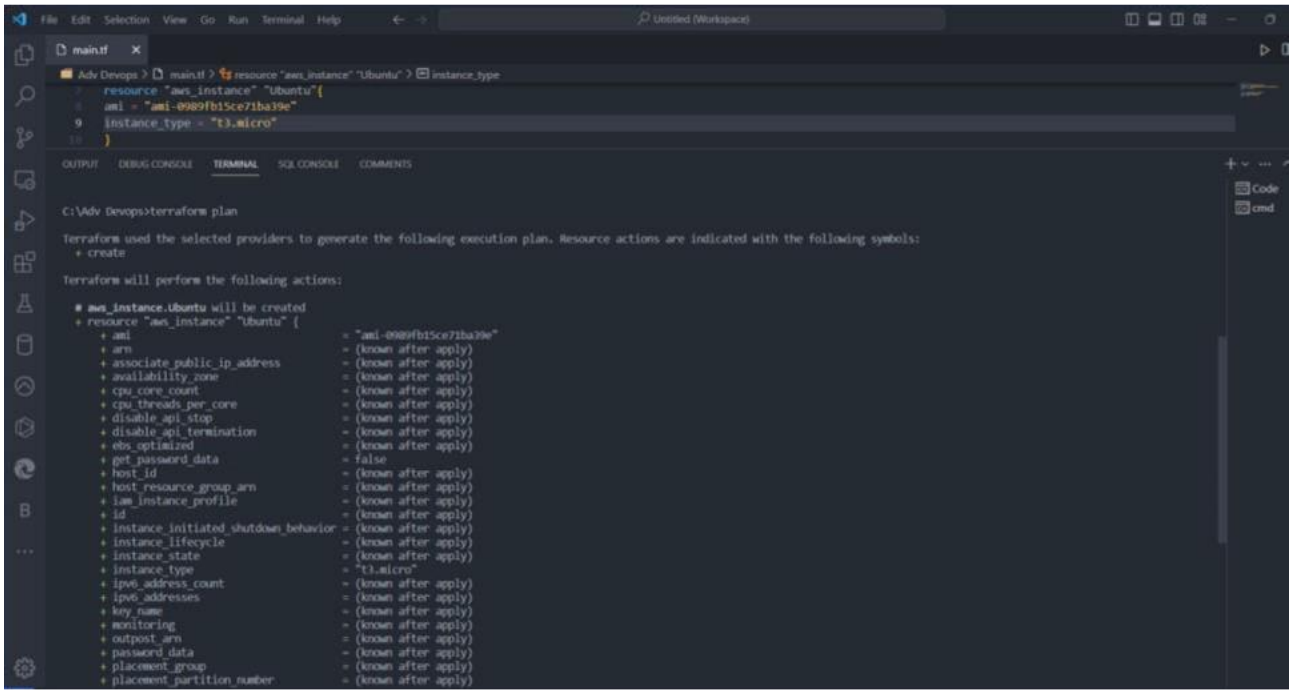

Create access key | IAM | Global
us-east-1.console.aws.amazon.com/iamv2/home?region=eu-north-1#/users/details/terraform1/create-access-key
aws
Services
Search
[Alt+S]
Global
Harshita Sachdev
Access key best practices & alternatives
Step 2 - optional
Set description tag
Step 3
Retrieve access keys
Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.
Use case
Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.
Local code
You plan to use this access key to enable application code in a local development environment to access your AWS account.
Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.
Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.
Application running outside AWS
You plan to use this access key to enable an application running on an on-premises host, or to use a local AWS client or third-party AWS plugin.
Other
Your use case is not listed here.
Alternatives recommended
CloudShell
Feedback
Language

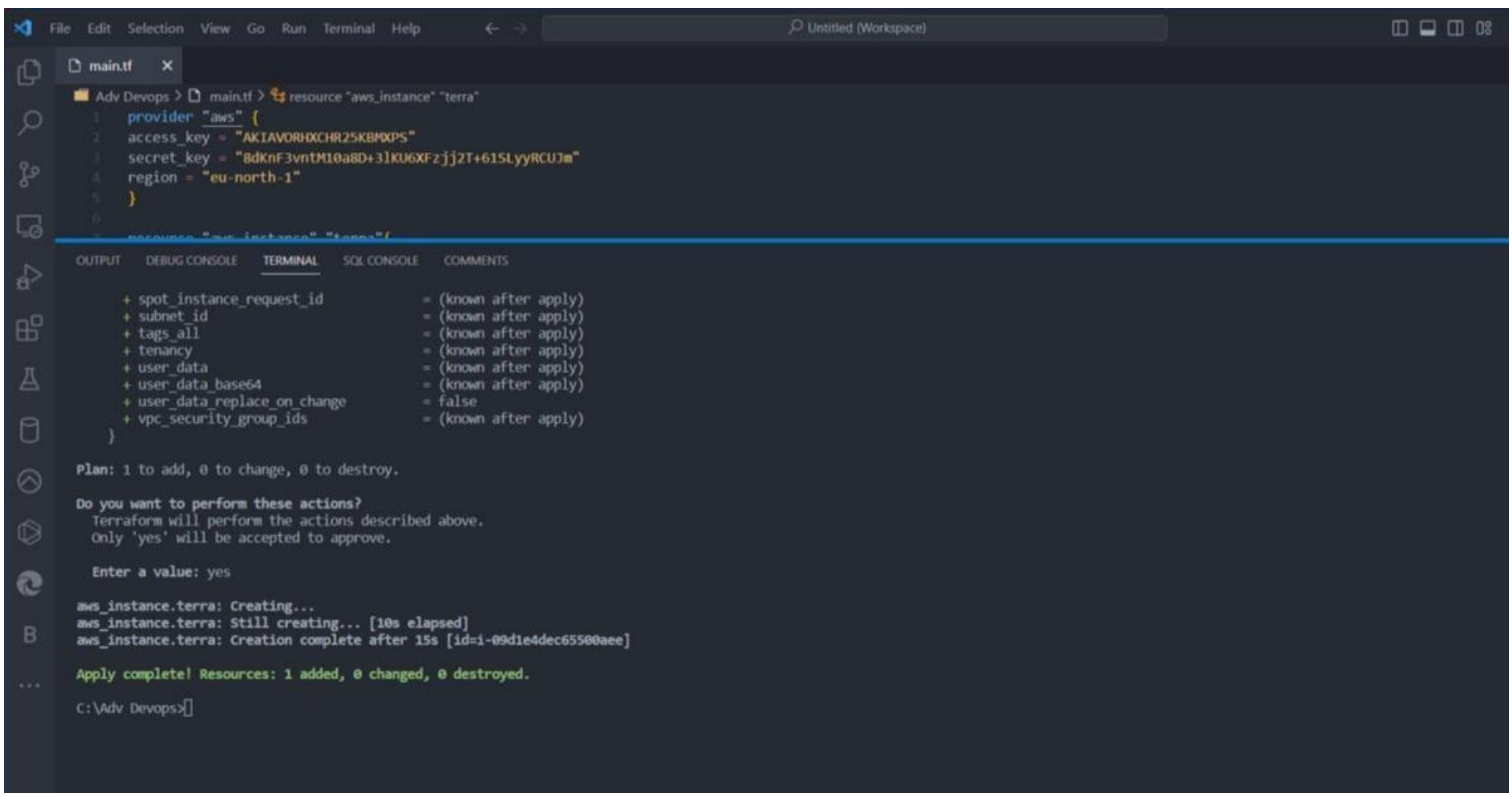

Create access key | IAM | Global
us-east-1.console.aws.amazon.com/iamv2/home?region=eu-north-1#/users/details/terraform1/create-access-key
Services
Search
[Alt+S]
Global
Harshita Sachdev
IAM > Users > terraform1 > Create access key
Step 1
Access key best practices & alternatives
Step 2 - optional
Set description tag
Step 3
Retrieve access keys
Set description tag - optional Info
The description for this access key will be attached to this user as a tag and shown alongside the access key.
Description tag value
Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidently later.
terraform-keys
Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: _ . : / = + - @
Cancel
Previous
Create access key

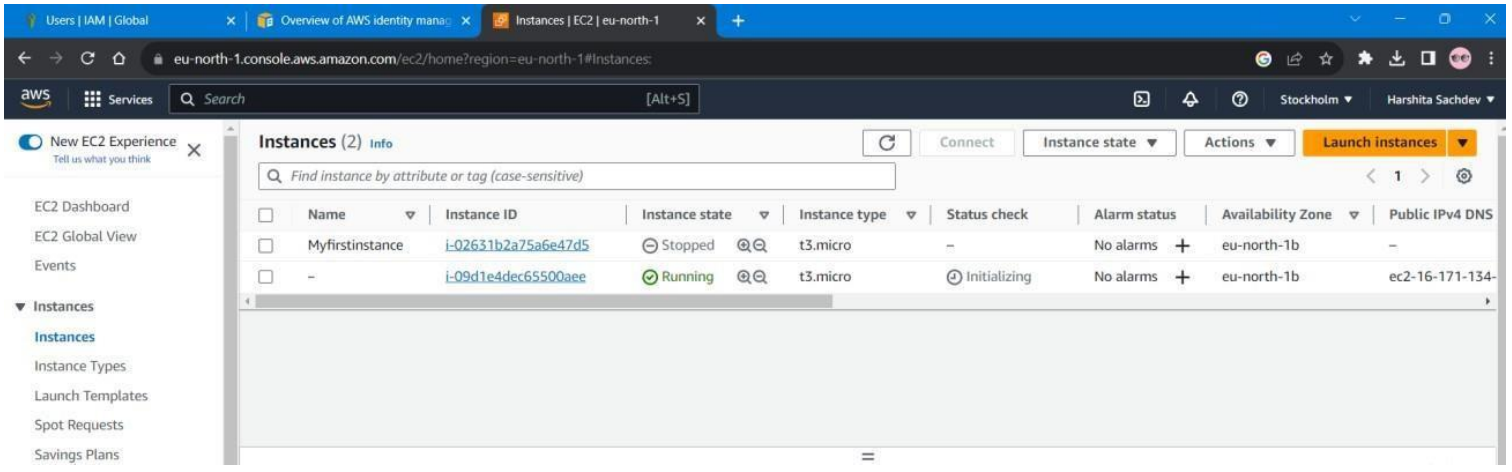

Create access key | IAM | Global
us-east-1.console.aws.amazon.com/iamv2/home?region=eu-north-1#/users/details/terraform1/create-access-key
Services
Search
[Alt+S]
Global
Harshita Sachdev
Access key created
This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.
IAM > Users > terraform1 > Create access key
Step 1
Access key best practices & alternatives
Step 2 - optional
Set description tag
Step 3
Retrieve access keys
Retrieve access keys Info
Access key
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.
Access key
Secret access key
AKIAVORHXCHRRXZVS47N
********** Show
Access key best practices
Never store your access key in plain text, in a code repository, or in code.
Disable or delete access key when no longer needed.
Enable least-privilege permissions.
Rotate access keys regularly.
For more details about managing access keys, see the best practices for managing AWS access keys.
Download .csv file
Done

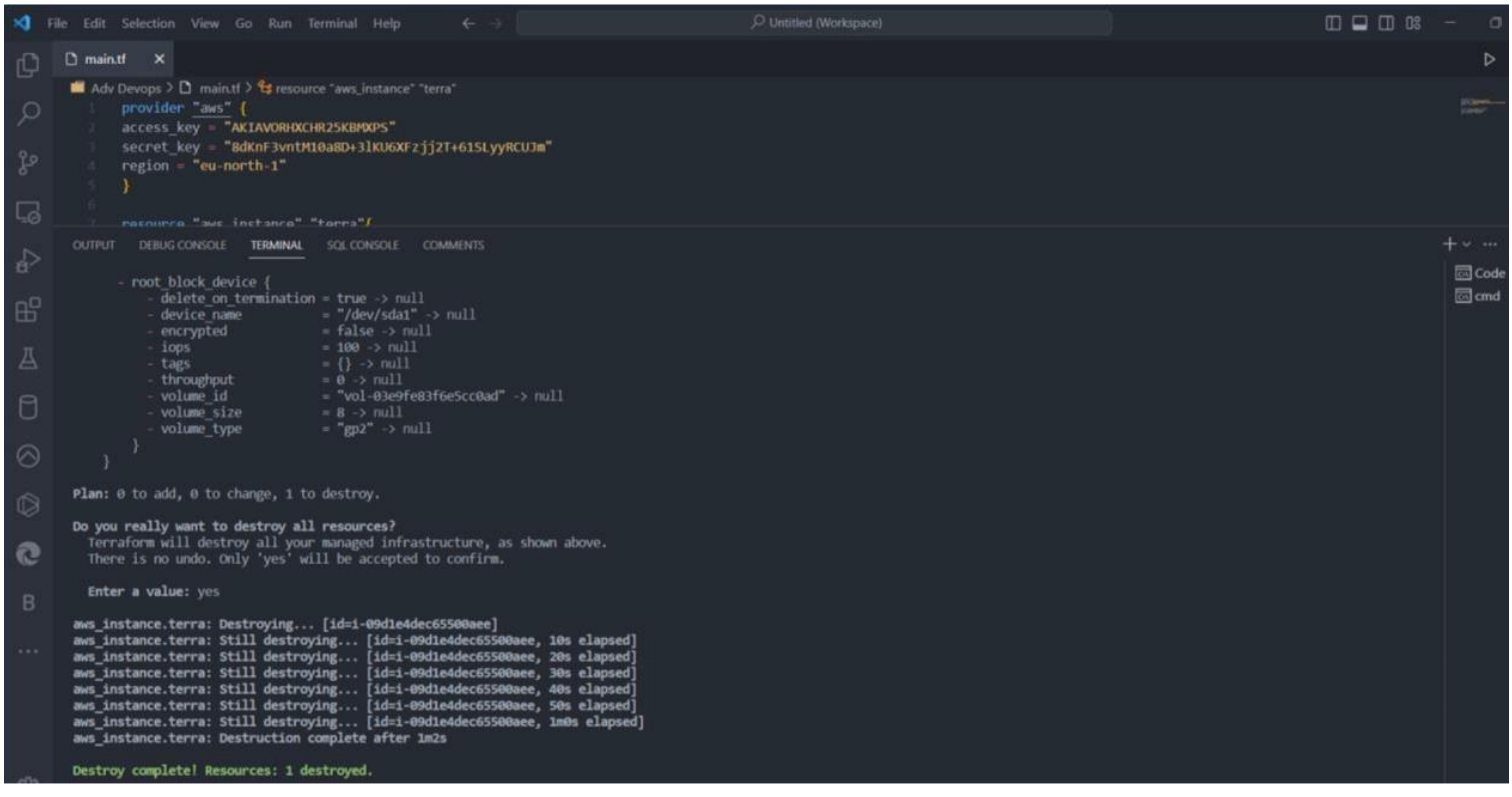

File Edit Selection View Go Run Terminal Help
main.tf
Adv Devops > main.tf > ...
provider "aws" {
access_key = "AKIAVORHXCHR25KBMXPS"
secret_key = "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"
region = "eu-north-1"
}
resource "aws_instance" "terra"{
ami = "ami-08766f81ab52792ce"
instance_type = "t3.micro"
}

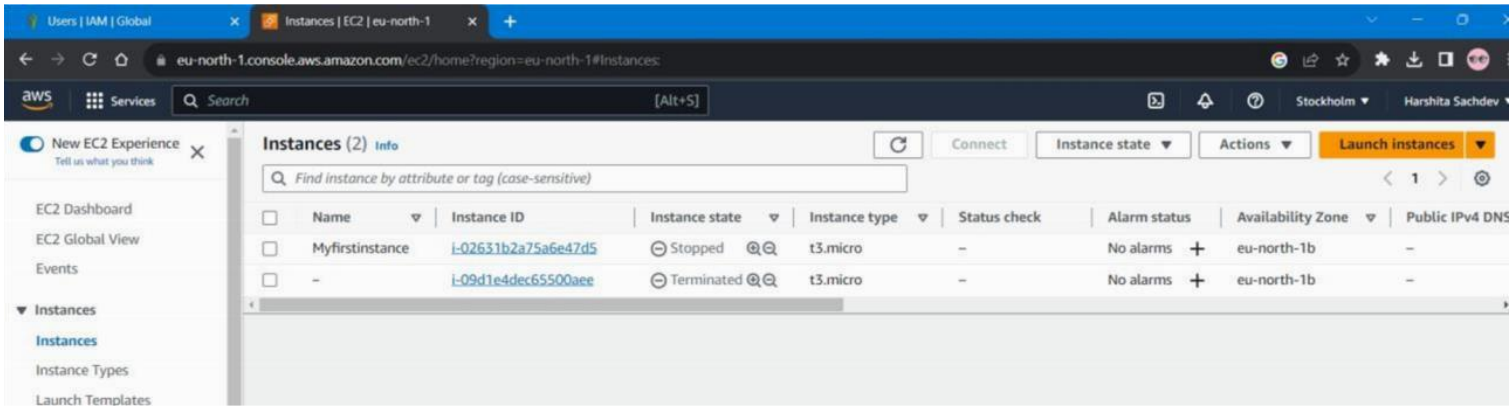

main.tf
Adv Devops > main.tf > resource "aws_instance" "Ubuntu"
provider "aws" {
access_key = "AKIAVORHXCHRVC66AB3T"
secret_key = "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"
region = "us-east-2"
OUTPUT DEBUG CONSOLE TERMINAL SQL CONSOLE COMMENTS
Microsoft Windows [Version 10.0.22621.2134]
(c) Microsoft Corporation. All rights reserved.
C:\Adv Devops>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.13.1...
- Installed hashicorp/aws v5.13.1 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider selections it made above. Include this file in your version control repository so that Terraform can guarantee to make the same selections by default when you run "terraform init" in the future.
Terraform has been successfully initialized!
You may now begin working with Terraform. Try running "terraform plan" to see any changes that are required for your infrastructure. All Terraform commands should now work.
If you ever set or change modules or backend configuration for Terraform, rerun this command to reinitialize your working directory. If you forget, other commands will detect it and remind you to do so if necessary.
C:\Adv Devops>


File Edit Selection View Go Run Terminal Help
Untitled (Workspace)
main.tf
Adv Devops > main.tf > resource "aws_instance" "Ubuntu" > instance_type
resource "aws_instance" "Ubuntu"{
ami = "ami-0989fb15ce71ba39e"
instance_type = "t3.micro"
}
OUTPUT DEBUG CONSOLE TERMINAL SQL CONSOLE COMMENTS
Code
cmd
C:\Adv Devops>terraform plan
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.Ubuntu will be created
+ resource "aws_instance" "Ubuntu" {
+ ami = "ami-0989fb15ce71ba39e"
+ 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)
+ disable_api_termination = (known after apply)
+ ebs_optimized = (known after apply)
+ get_password_data = false
+ host_id = (known after apply)
+ host_resource_group_arn = (known after apply)
+ iam_instance_profile = (known after apply)
+ id = (known after apply)
+ instance_initiated_shutdown_behavior = (known after apply)
+ instance_lifecycle = (known after apply)
+ instance_state = (known after apply)
+ instance_type = "t3.micro"
+ ipv6_address_count = (known after apply)
+ ipv6_addresses = (known after apply)
+ key_name = (known after apply)
+ monitoring = (known after apply)
+ outpost_arn = (known after apply)
+ password_data = (known after apply)
+ placement_group = (known after apply)
+ placement_partition_number = (known after apply)


File Edit Selection View Go Run Terminal Help
Untitled (Workspace)
main.tf
Adv Devops > main.tf > resource "aws_instance" "terra"
provider "aws" {
access_key = "AKIAVORHXCHR25KBMXPS"
secret_key = "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"
region = "eu-north-1"
}
OUTPUT DEBUG CONSOLE TERMINAL SQL CONSOLE COMMENTS
+ spot_instance_request_id = (known after apply)
+ subnet_id = (known after apply)
+ tags_all = (known after apply)
+ tenancy = (known after apply)
+ user_data = (known after apply)
+ user_data_base64 = (known after apply)
+ user_data_replace_on_change = false
+ vpc_security_group_ids = (known after apply)
}
Plan: 1 to add, 0 to change, 0 to destroy.
Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.
Enter a value: yes
aws_instance.terra: Creating...
aws_instance.terra: Still creating... [10s elapsed]
aws_instance.terra: Creation complete after 15s [id=i-09d1e4dec65500aee]
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
C:\Adv Devops>







Delete all security keys and csv files.
Delete users and user groups

CONCLUSION –

In this assignment we learnt about the terraform lifecycle, core concepts/terminologies and installation of terraform in windows and creating/destroying an EC2 instance.