

Name - Mikil Lakwani

D 15B/37

## Advance DevOps Lab

### Experiment 5:

Aim -

To understand terraform lifecycle, core concepts/terminologies and install it.

#### Theory

Terraform is an Infrastructure as code (IaC) tool that allows you to build, change and version infrastructure safely and efficiently. This includes low-level components such as compute instances, storage and networking as well as high-level components such as DNS entries, SaaS features etc.

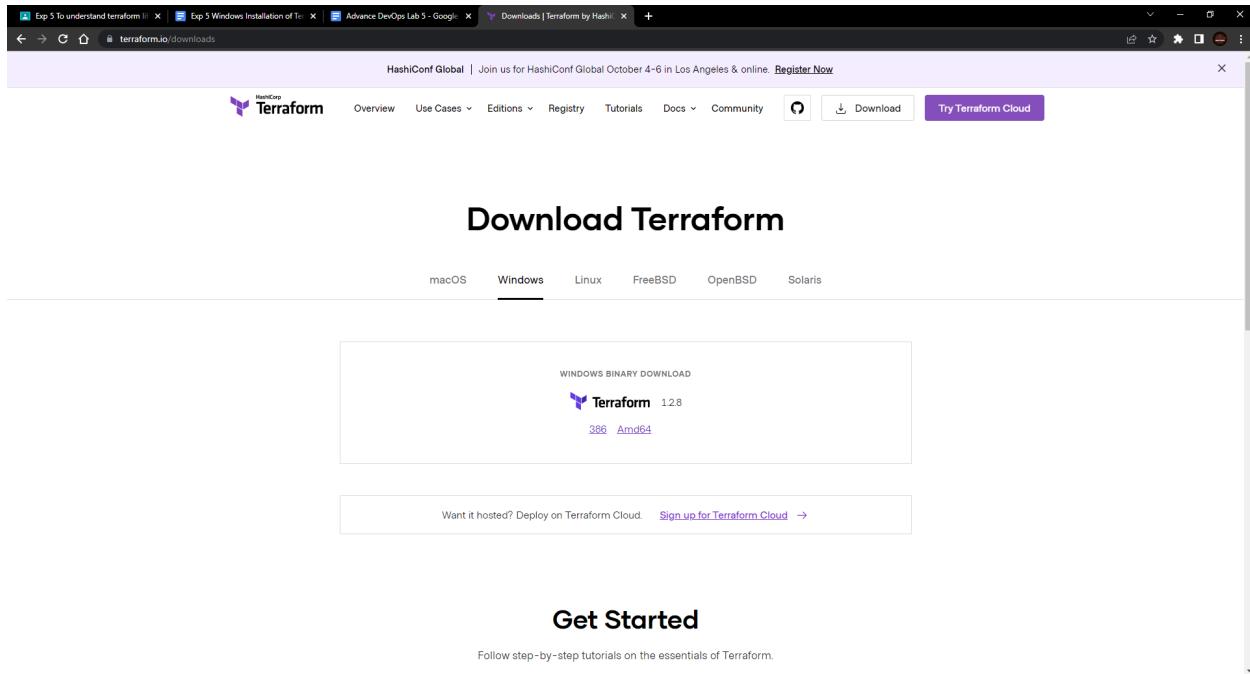
Terraform can manage infrastructure on multiple cloud platforms. Terraform's state allows you to track resource changes throughout your deployments. You can commit your configurations to version control to safely collaborate on infrastructure. Terraform plugins called providers let Terraform interact with cloud platforms and other services via their APIs.

## Steps:

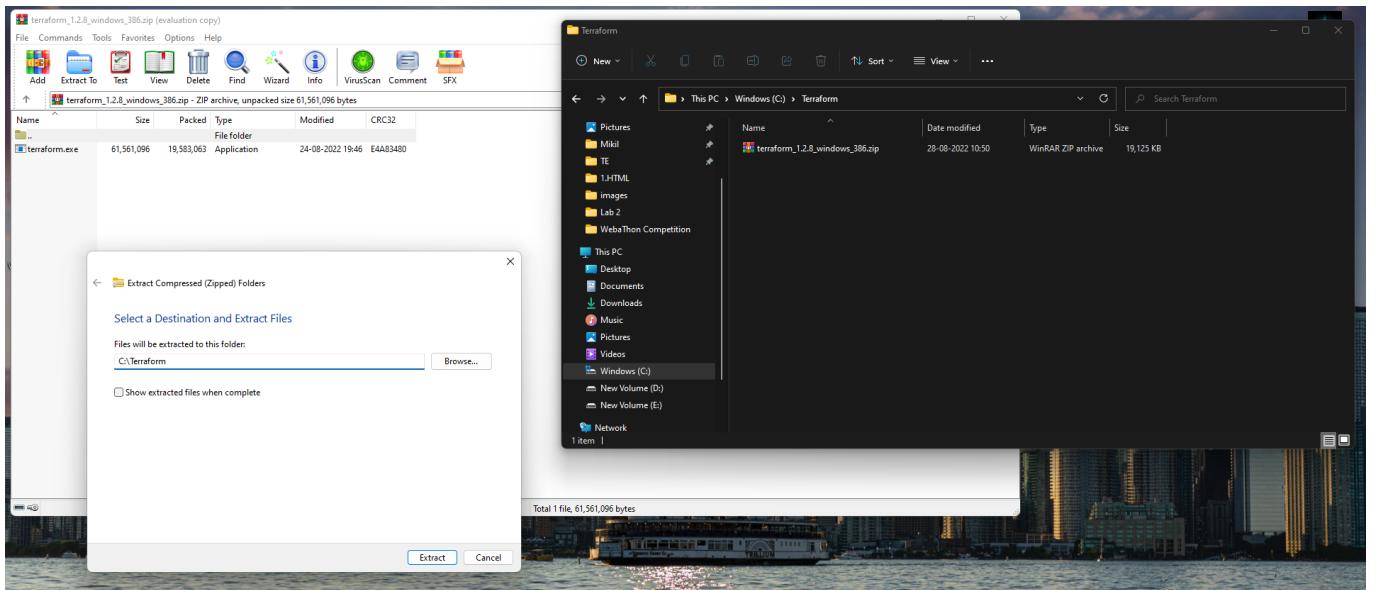
### 1. Download terraform

To install Terraform, First Download the Terraform Cli Utility for windows from terraforms official website

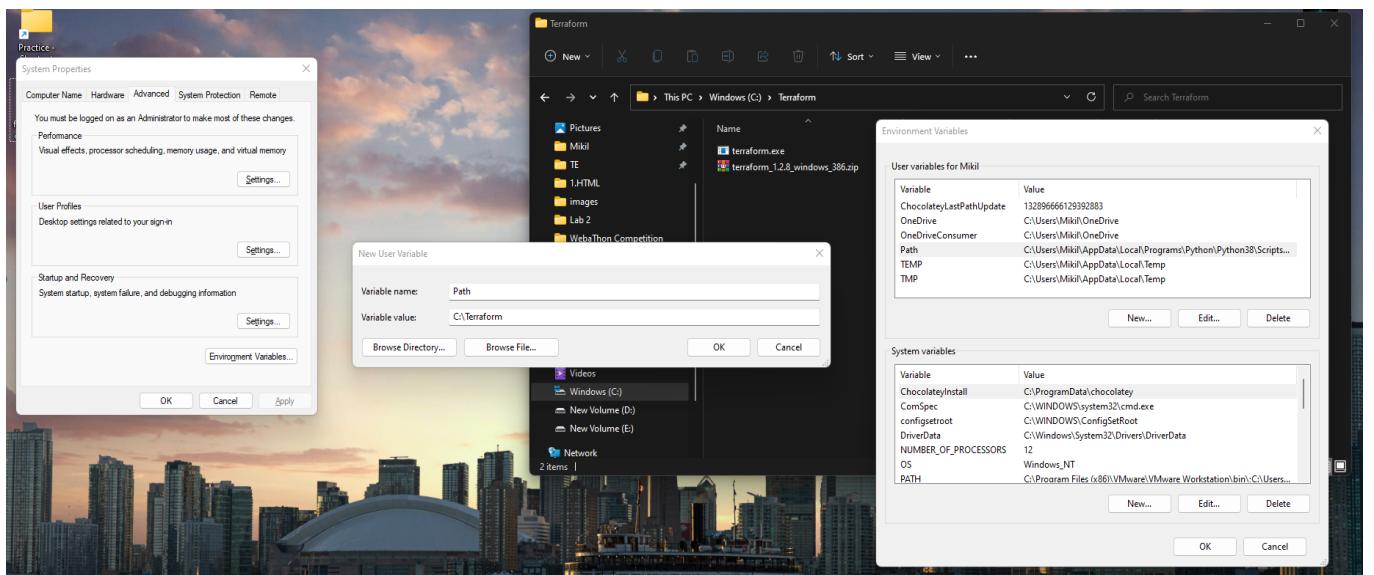
Select the Operating System Windows followed by either 32-bit or 64-bit based on your OS type.



### 2. Extract the downloaded setup file Terraform.exe in the C:\Terraform directory.

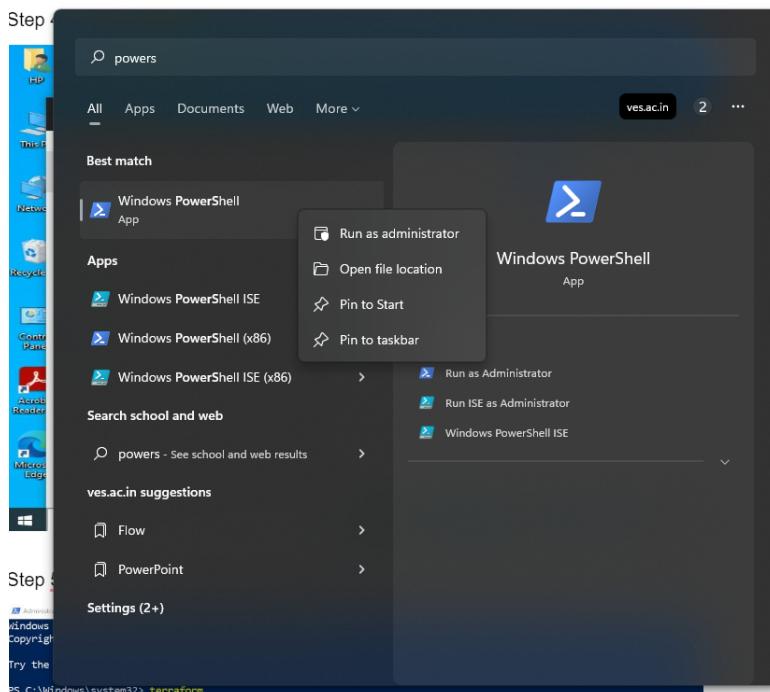


### 3. Set the System path for Terraform in Environment Variables



#### 4. Open PowerShell with Admin Access.

Step 4



#### 5. Open Terraform in PowerShell and check its functionality.

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\WINDOWS\system32> terraform
Usage: terraform [global options] <subcommand> [args]

The available commands for execution are listed below.
The primary workflow commands are given first, followed by
less common or more advanced commands.

Main commands:
  init      Prepare your working directory for other commands
  validate  Check whether the configuration is valid
  plan     Show changes required by the current configuration
  apply    Create or update infrastructure
  destroy   Destroy previously-created infrastructure

All other commands:
  console   Try Terraform expressions at an interactive command prompt
  fmt       Reformat your configuration in the standard style
  force-unlock Release a stuck lock on the current workspace
  get       Install or upgrade remote Terraform modules
  graph    Generate a Graphviz graph of the steps in an operation
  import   Associate existing infrastructure with a Terraform resource
  login    Obtain and save credentials for a remote host
  logout   Remove locally-stored credentials for a remote host
  output   Show output values from your root module
  providers Show the providers required for this configuration
  refresh  Update the state to match remote systems
  show     Show the current state or a saved plan
  state    Advanced state management
  taint    Mark a resource instance as not fully functional
  test     Experimental support for module integration testing
  untaint Remove the 'tainted' state from a resource instance
  version  Show the current Terraform version
  workspace Management
```

**Note: If any error comes, then please recheck or set the path of Terraform in the Environment variable again.**

## Conclusion -

Thus, we have successfully installed Terraform.

Now we can use Terraform to manage our infrastructure. We can define resources in Terraform files and then apply them to provision infrastructure. Terraform will then automatically detect changes and apply them to the infrastructure. This makes it easier to manage infrastructure as code. Terraform also provides a visual interface for managing infrastructure, making it easier for non-technical members of the team to understand and contribute to the infrastructure management process. Overall, Terraform is a powerful tool for managing infrastructure in a more efficient and reliable way.