EXPERIMENT NO. 5

NAME : PRANAV POL CLASS : D15A ROLL NO. : 42

Aim : To understand terraform lifecycle, core concepts/terminologies and install it on a Linux Machine and Windows.

Theory :

Terraform is an open-source Infrastructure as Code (IaC) tool developed by HashiCorp. It allows users to define and provision infrastructure using a high-level configuration language known as HashiCorp Configuration Language (HCL) or JSON. Terraform supports a wide range of cloud providers, such as AWS, Azure, Google Cloud, and on-premises solutions, enabling users to manage infrastructure across multiple environments consistently.

Core Concepts and Terminologies

1. **Providers:**
   Providers are plugins that allow Terraform to interact with various APIs of cloud providers, SaaS providers, and other services. Each provider requires configuration and manages resources for that specific service.
2. **Resources:**
   Resources are the most fundamental elements in Terraform. They represent components of your infrastructure, such as virtual machines, databases, networks, and more.
3. **Modules:**
   Modules are containers for multiple resources that are used together. A module can call other modules, creating a hierarchical structure. This makes it easier to organize and reuse code.
4. **State:**
   Terraform maintains a state file that keeps track of the infrastructure managed by Terraform. The state file is crucial as it provides a mapping between the real-world resources and the configuration defined in Terraform.
5. **Variables:**
   Variables in Terraform are used to make configurations dynamic and reusable. They can be defined in the configuration files and assigned values at runtime.

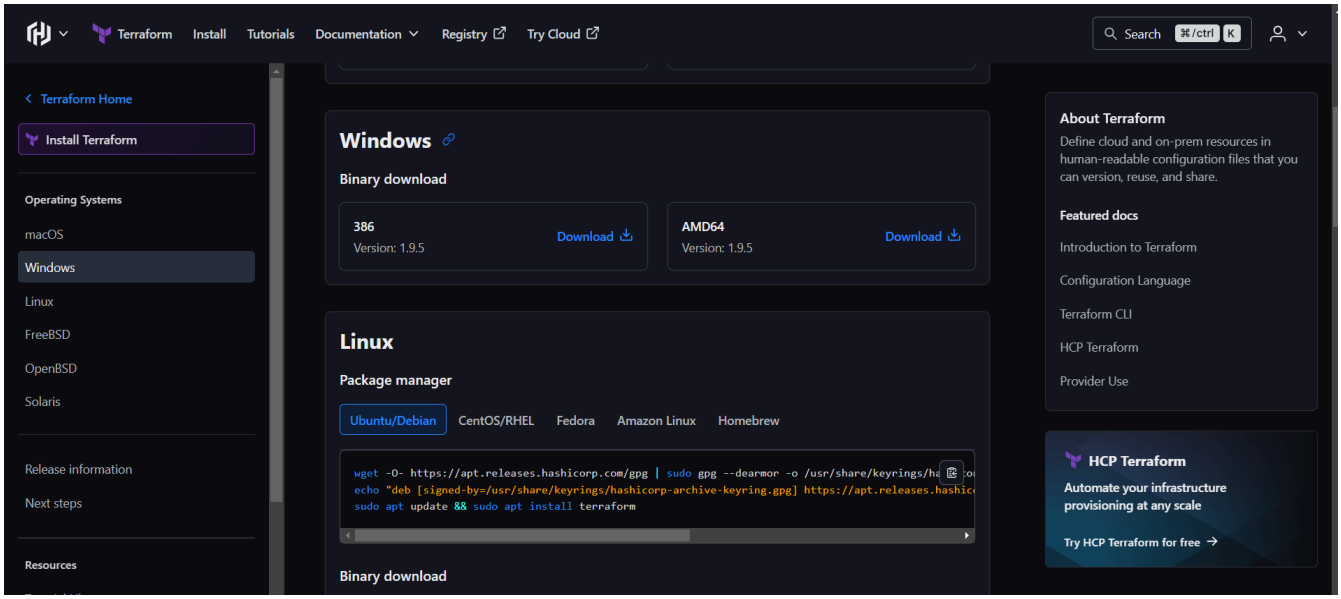
6. **Outputs:**
   Outputs are used to extract information from the Terraform-managed infrastructure and display it after the execution of a Terraform plan or apply.

Terraform Lifecycle

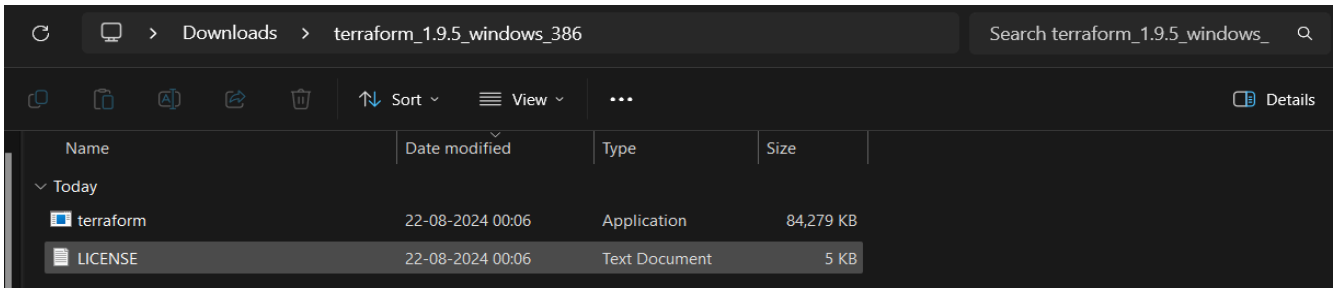
1. **Write:**
   Write the configuration file (typically with .tf extension) using HCL to describe the desired infrastructure.
2. **Initialize (terraform init):**
   Initialize the working directory containing the configuration files. This command downloads the necessary provider plugins and sets up the environment.
3. **Plan (terraform plan):**
   Terraform creates an execution plan based on the configuration files. It compares the current state with the desired state and shows the changes that will be made.
4. **Apply (terraform apply):**
   Apply the changes required to reach the desired state of the configuration. Terraform will prompt for confirmation before making any changes.
5. **Destroy (terraform destroy):**
   Destroy the infrastructure managed by Terraform. This command is used to remove all resources defined in the configuration files.

Implementation :

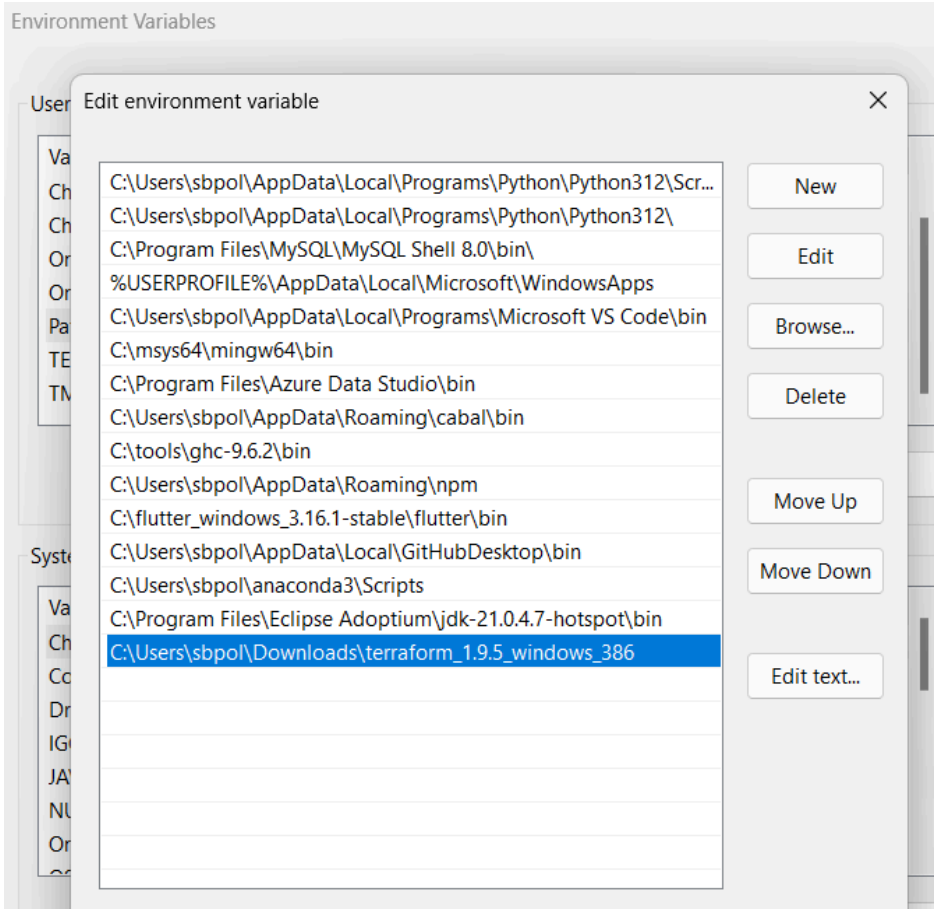
Step 1: Go to Terraform website and download 386 .



Step 2 : Extract the zip file and copy the path of file



Step 3: Edit environment variables and paste copied path



Step 4 : check if the Terraform is installed correctly.

```
(base) PS C:\Users\sbpol> 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
  metadata      Metadata related commands
  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
```

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

Loading personal and system profiles took 1476ms.
(base) PS C:\Users\sbpol> terraform --version
Terraform v1.9.5
on windows_386
(base) PS C:\Users\sbpol>
```