Name: Harsh Dalvi Subject: Advance DevOps , Sem: SEM V
Roll No: 13 Class / Batch: TE-IT / Batch B

Experiment No. 5

Aim: To explain terraform lifecycle, core concepts/terminologies and install it on a Linux Machine. **(LO1, LO3)**

Theory:

Terraform:
Terraform is an open-source, infrastructure as code, software tool created by HashiCorp. Users define and provide data center infrastructure using a declarative configuration language known as HashiCorp Configuration Language (HCL), or optionally JSON. It is used to define and provision the complete infrastructure using an easy-to-learn declarative language.

It is an infrastructure provisioning tool where you can store your cloud infrastructure setup as codes. It's very similar to tools such as CloudFormation, which you would use to automate your AWS infrastructure, but you can only use that on AWS. With Terraform, you can use it on other cloud platforms as well.

Each cloud platform has its own set of rules, syntax, and commands to work with, Terraform makes it easy for us to work with all such clouds at the same time. Terraform uses different plugins for different cloud platforms. So, if you use terraform you do not need to learn the different syntax, commands, or rules required by the platforms. Terraform will automatically connect to platforms without you being worry.

Some of the Benefits of using Terraform:

- Does orchestration, not just configuration management.
- Supports multiple providers such as AWS, Azure, GCP, DigitalOcean and many more.
- Provide immutable infrastructure where configuration changes smoothly.
- Uses easy to understand language, HCL (HashiCorp configuration language).
- Easily portable to any other provider.
- Supports Client only architecture, so no need for additional configuration management on server.

Terraform Core concepts:
Below are the core concepts/terminologies used in Terraform:

1. **Variables:** Also used as input-variables, it is key-value pair used by Terraform modules to allow customization.

2. **Provider:** It is a plugin to interact with APIs of service and access its related resources.

3. **Module:** It is a folder with Terraform templates where all the configurations are defined.

4. **State:** It consists of cached information about the infrastructure managed by Terraform and the related configurations.

5. **Resources:** It refers to a block of one or more infrastructure objects (compute instances, virtual networks, etc.), which are used in configuring and managing the infrastructure.

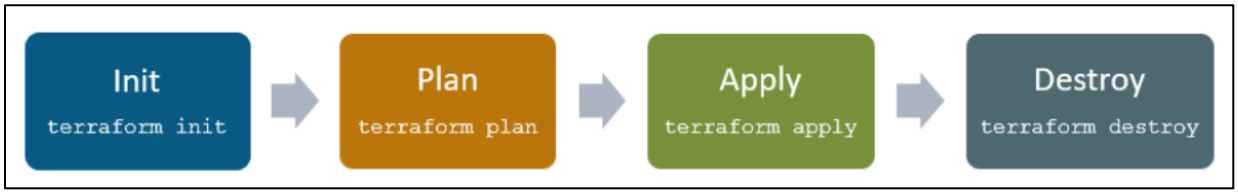
6. **Data Source:** It is implemented by providers to return information on external objects to terraform.

7. **Output Values:** These are return values of a terraform module that can be used by other configurations.

8. **Plan:** It is one of the stages where it determines what needs to be created, updated, or destroyed to move from real/current state of the infrastructure to the desired state.

9. **Apply:** It is one of the stages where it applies the changes real/current state of the infrastructure in order to move to the desired state.

Terraform Lifecycle:
Terraform lifecycle consists of – init, plan, apply, and destroy.



- Terraform init initializes the working directory which consists of all the configuration files

- Terraform plan is used to create an execution plan to reach a desired state of the infrastructure. Changes in the configuration files are done in order to achieve the desired state.

- Terraform apply then makes the changes in the infrastructure as defined in the plan, and the infrastructure comes to the desired state.

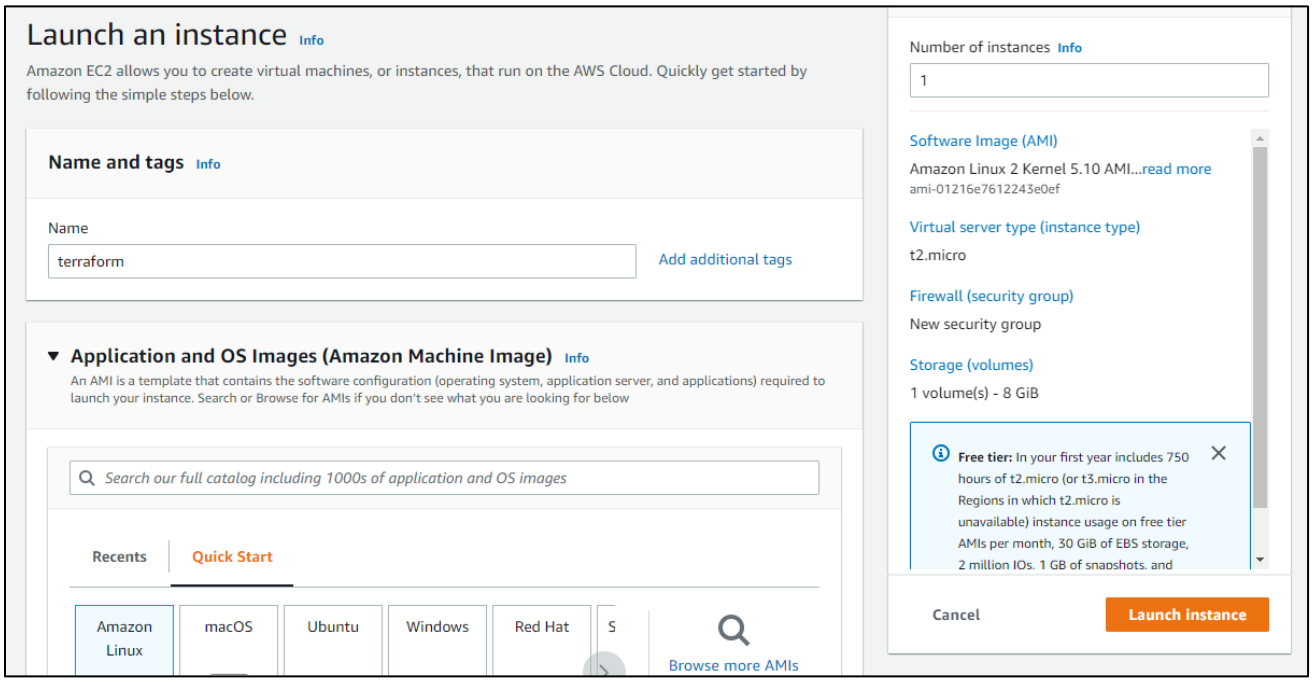
- Terraform destroy is used to delete all the old infrastructure resources, which are marked tainted after the apply phase.

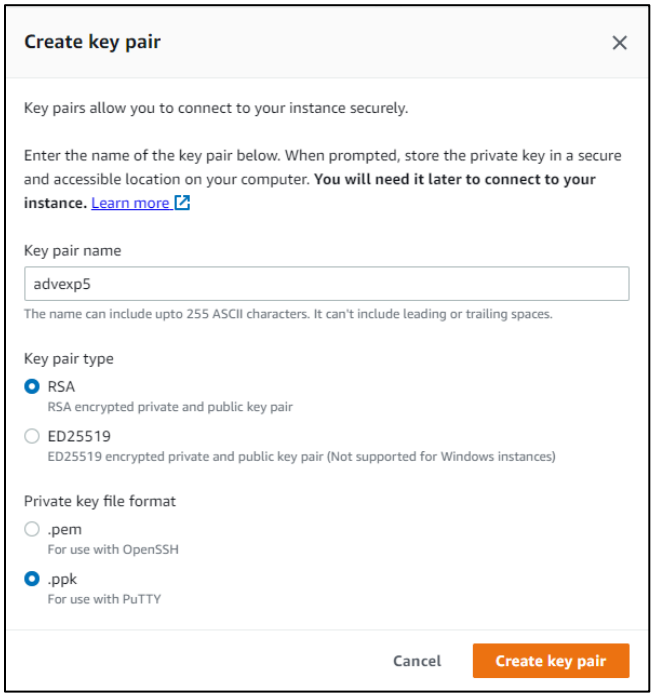
Name: Harsh Dalvi
Roll No: 13

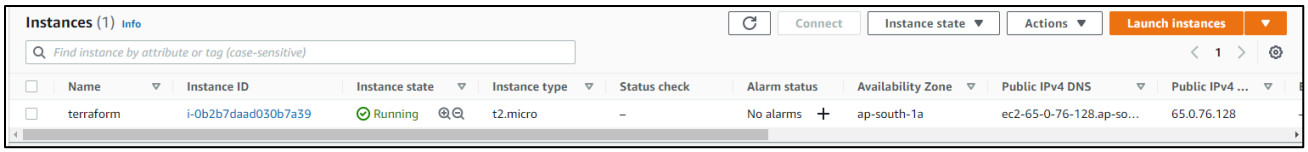
Subject: Advance DevOps , Sem: SEM V
Class / Batch: TE-IT / Batch B

Steps to install terraform on a Linux Machine:

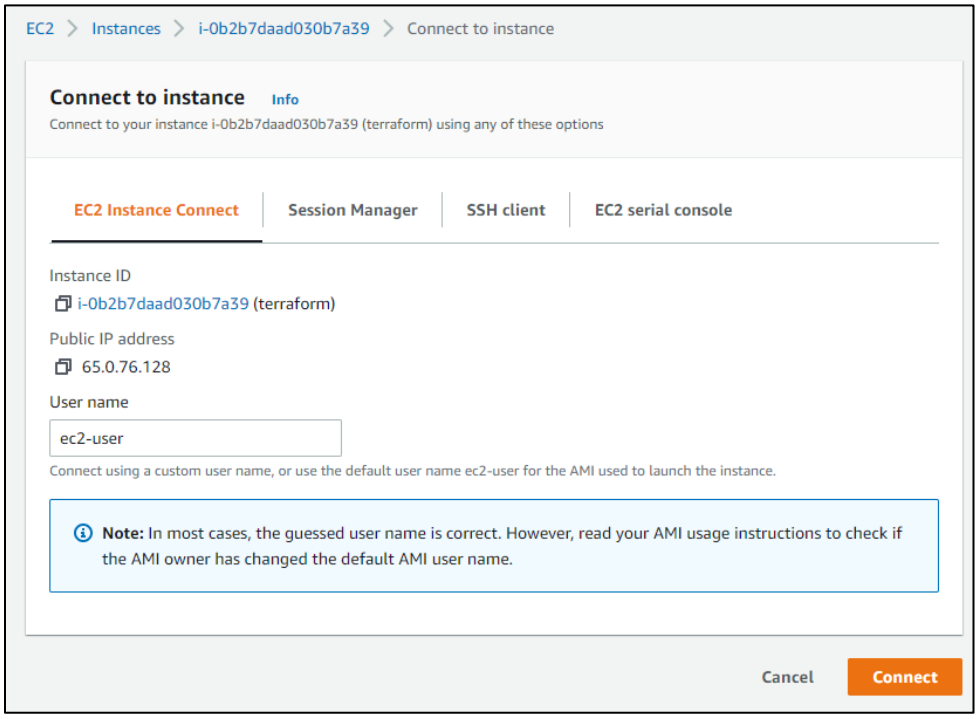
1. Create a Linux Machine instance:



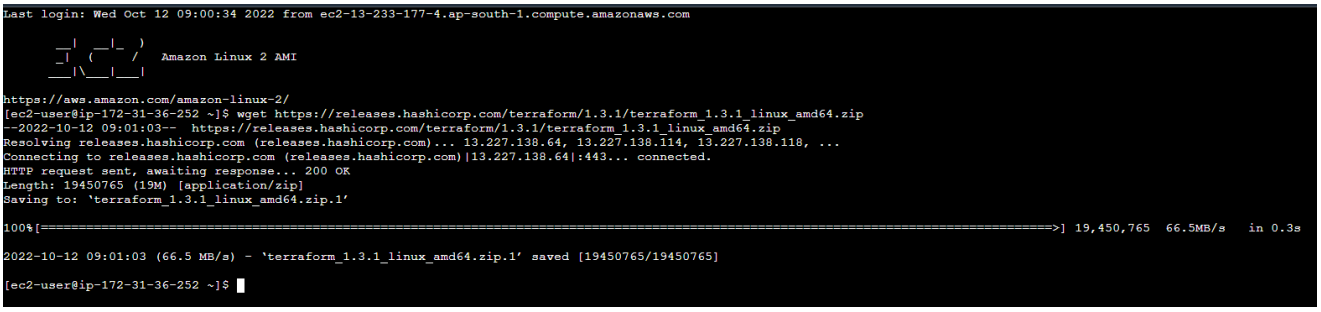




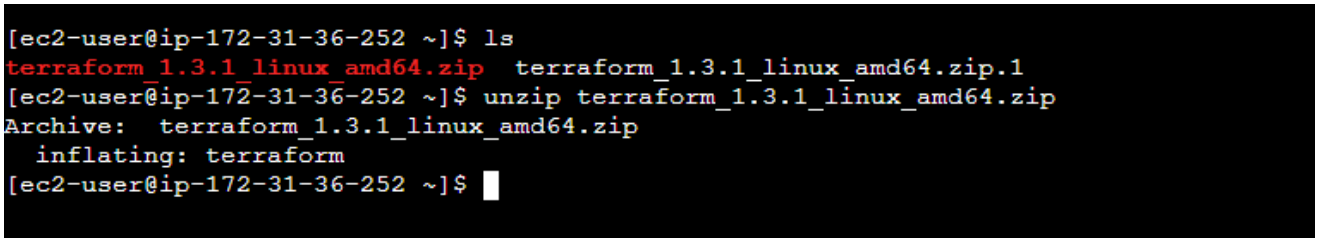
2. Connect to the instance:



3. Install Terraform
   Command:
   wget https://releases.hashicorp.com/terraform/1.3.1/terraform_1.3.1_linux_amd64.zip



4. Check for the zip folder
   Command: ls



5. Unzip the zip folder:
   Command: unzip terraform_1.3.1_linux_amd64.zip

```
[ec2-user@ip-172-31-36-252 ~]$ unzip terraform_1.3.1_linux_amd64.zip
Archive:  terraform_1.3.1_linux_amd64.zip
  inflating: terraform
[ec2-user@ip-172-31-36-252 ~]$
```

6. Check for installation and version
   Command:
   a. sudo mv terraform /usr/local/bin/
   b. terraform –version

```
[ec2-user@ip-172-31-36-252 ~]$ sudo mv terraform /usr/local/bin/
[ec2-user@ip-172-31-36-252 ~]$ terraform --version
Terraform v1.3.1
on linux_amd64

Your version of Terraform is out of date! The latest version
is 1.3.2. You can update by downloading from https://www.terraform.io/downloads.html
[ec2-user@ip-172-31-36-252 ~]$
```

Conclusion: From this experiment, it is concluded that we have learnt the concept of Terraform. In this experiment, we successfully installed Terraform on Linux Machine. We understood and learnt terraform core concepts and terraform lifecycle. Hence, we have successfully achieved the Lab Outcome 1 and 3 (LO1 and LO3). Also, we have achieved PO1, PO2, PO3, PO4, PO5, PO9, PO10 and PO12 from this experiment.