

Terraform Task1

1. Notes on Terraform

a. What is Terraform?

Terraform is an open-source infrastructure as code (IaC) tool developed by HashiCorp. It allows users to define and provision data center infrastructure using a high-level configuration language called HashiCorp Configuration Language (HCL). Terraform manages external resources (such as public cloud infrastructure, private cloud infrastructure, network appliances, software as a service, and platform as a service) with a modular and efficient approach.

b. Why Terraform?

Terraform is widely used for several reasons:

1. **Platform Agnostic:** Terraform supports multiple service providers, including AWS, Azure, Google Cloud, and others, making it versatile for various cloud environments.
2. **Infrastructure as Code:** Enables writing and managing infrastructure through code, which promotes version control, collaboration, and automation.
3. **Declarative Language:** Users describe the desired state of their infrastructure, and Terraform takes care of achieving that state.
4. **Modular and Reusable:** Supports modular configurations, making it easy to reuse code for different projects and teams.
5. **State Management:** Keeps track of infrastructure state, ensuring updates and changes are applied correctly and safely.

c. Benefits of Terraform

1. **Scalability:** Simplifies scaling infrastructure up or down by modifying configuration files.
2. **Consistency:** Ensures consistent environments across different deployments by using the same configuration files.
3. **Collaboration:** Facilitates teamwork through shared configuration files and version control.
4. **Automation:** Automates the provisioning and management of infrastructure, reducing manual intervention and errors.
5. **Cost-Effective:** Optimizes resource management and minimizes unnecessary expenditures.

2. Launching Two EC2 Instances

To launch two EC2 instances named "myapp-1" and "myapp-2" using Amazon Linux OS in the `ap-south-1` region, follow these Terraform steps:

1. **Create a Terraform Configuration File:** create a new file named `main.tf`.

```
provider "aws" {

    region    = "ap-south-1"

    access_key = "AKIA4MTTRYQCL6HZ"

    secret_key = "Ad9jTGv4evN511/JCb3qfYdqn/bhRI"

}


resource "aws_instance" "myapp_1" {

    ami        = "ami-0e1d0622679bc1c5" # Replace with the latest Amazon Linux 2
    AMI ID

    instance_type = "t2.micro"

    tags = {

        Name = "myapp-1"

    }

}


resource "aws_instance" "myapp_2" {

    ami        = "ami-0e1d06225679c1c5" # Replace with the latest Amazon Linux 2
    AMI ID

    instance_type = "t2.micro"

    tags = {

        Name = "myapp-2"

    }

}
```

After creating this file use terraform init command for initialization

Then use terraform plan and after terraform apply

nano main.tf

```
ubuntu@ip-172-31-40-189:~/assignment$ ls
ubuntu@ip-172-31-40-189:~/assignment$ main.tf
main.tf: command not found
ubuntu@ip-172-31-40-189:~/assignment$ nano main.tf
ubuntu@ip-172-31-40-189:~/assignment$ ls
main.tf
ubuntu@ip-172-31-40-189:~/assignment$
```

i-05a3932caa1b7821c (Terraform)

PublicIPs: 65.2.166.215 PrivateIPs: 172.31.40.189

terraform plan

```
ubuntu@ip-172-31-40-189:~/assignment$ nano main.tf
ubuntu@ip-172-31-40-189:~/assignment$ terraform plan
```

i-05a3932caa1b7821c (Terraform)

terraform apply

```
ubuntu@ip-172-31-40-189:~/assignment$ nano main.tf
ubuntu@ip-172-31-40-189:~/assignment$ terraform apply
```

i-05a3932caa1b7821c (Terraform)

This is output

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Put
<input type="checkbox"/>	jenkins	i-0d7d0761e3bb7df38	⏸ Stopped	t2.small	-	View alarms +	ap-south-1a	-
<input checked="" type="checkbox"/>	Terraform	i-05a3932caa1b7821c	🟢 Running	t2.micro	🟢 2/2 checks passed	View alarms +	ap-south-1a	ec2
<input type="checkbox"/>	myapp-2	i-0022e59efda89753b	🟢 Running	t2.micro	🟡 Initializing	View alarms +	ap-south-1a	ec2
<input type="checkbox"/>	myapp-1	i-01fd5d65780fbab17	🟢 Running	t2.micro	🟡 Initializing	View alarms +	ap-south-1a	ec2

3. Installing Terraform, Integrating AWS, and Launching an EC2 Instance

Installing Terraform

1. **Download Terraform:** Download the Terraform binary from the Terraform website.
2. **Install Terraform:** Unzip the binary and move it to a directory included in your system's PATH.

```
$ unzip terraform_<VERSION>_linux_amd64.zip
$ sudo mv terraform /usr/local/bin/
$ terraform --version
```

Integrating AWS with Terraform

1. **Configure AWS CLI:** Install and configure the AWS CLI with your credentials.

```
$ aws configure
```

2. **Set Up VS Code:** Install the Terraform extension for Visual Studio Code.

Launching an EC2 Instance using VS Code

2. **Create a Terraform Configuration File:** In VS Code, create a new file named `main.tf`.

```
resource "aws_instance" "myapp_2" {
  ami          = "ami-0e1d06225679bc1c5"

  instance_type = "t2.micro"
  tags = {
    Name = "myserver"
  }
}

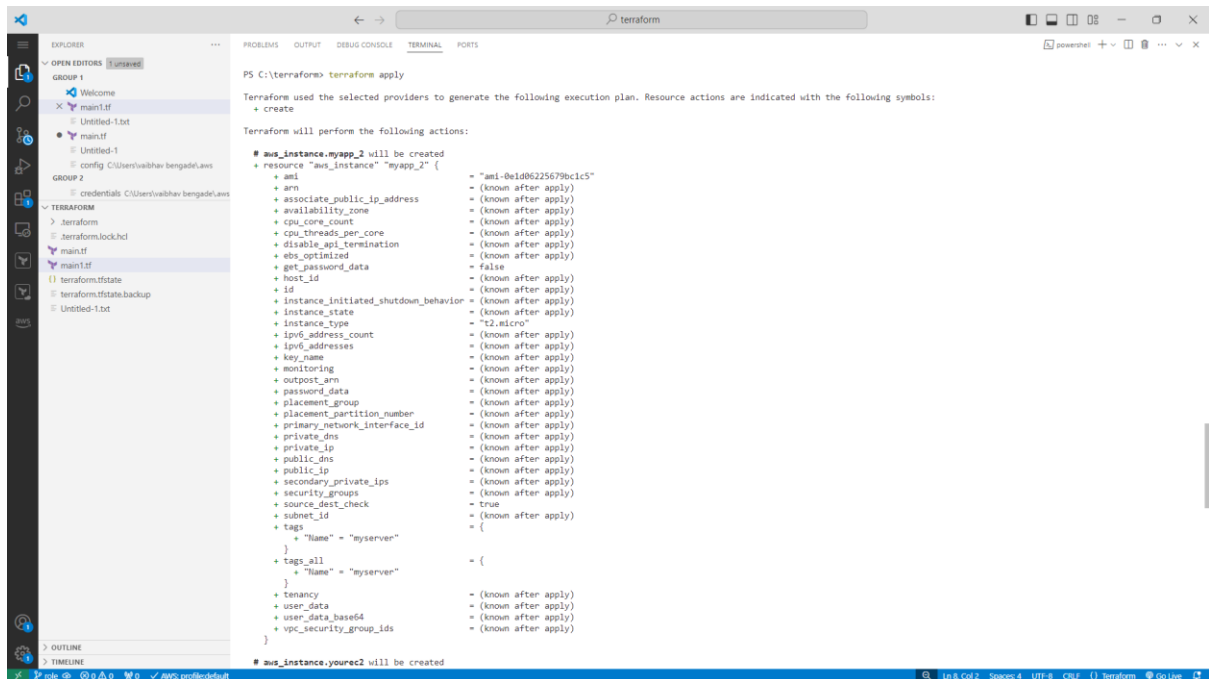
provider "aws" {
  region      = "ap-south-1"
  access_key  = "AKIA4MTWNJ2TRYQCLZ"
  secret_key  = "Ad9jTGv4evN511Nog/JCb3qfYdqn/bhRIt"
}
```

2. **Initialize Terraform:** Initialize Terraform in your working directory.

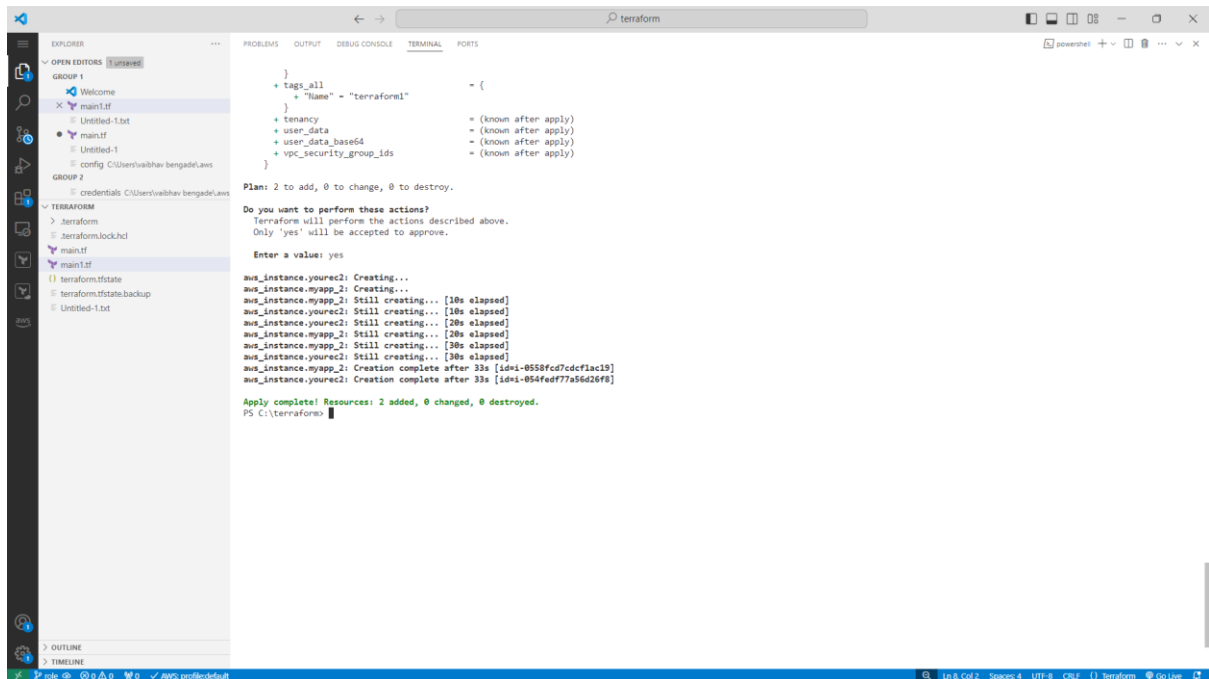
```
$ terraform init
```

3. **Apply the Configuration:** Apply the configuration to create the instance.

```
$ terraform apply
```

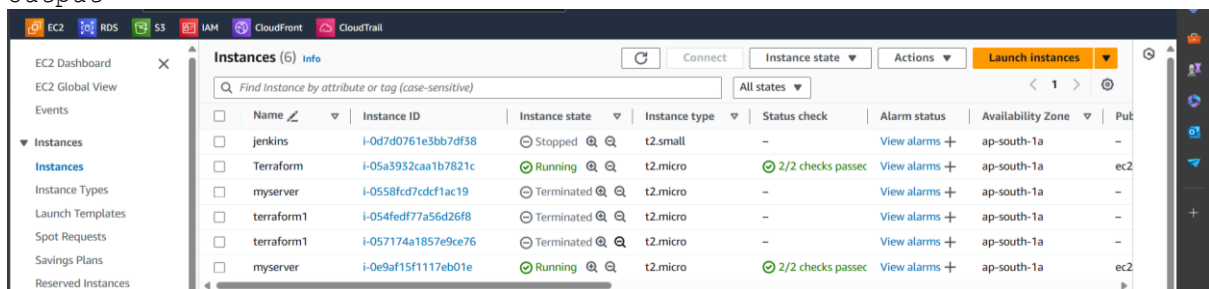


The screenshot shows the VS Code interface with the terminal window open. The terminal displays the output of the 'terraform apply' command. It starts with 'PS C:\terraform> terraform apply', followed by a confirmation message: 'Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols: + create'. The plan shows that 'aws_instance.myapp_2' will be created. The configuration for 'myapp_2' is detailed, including attributes like ami, availability_zone, instance_type ('t2.micro'), and various security and network settings. The plan also indicates that 'aws_instance.yourec2' will be created.



The screenshot shows the continuation of the 'terraform apply' command output in the VS Code terminal. It displays the 'Plan: 2 to add, 0 to change, 0 to destroy.' message, followed by a confirmation prompt: 'Do you want to perform these actions? Terraform will perform the actions described above. Only 'yes' will be accepted to approve.' The user enters 'yes', and the resources are created. The output shows the creation progress for 'aws_instance.yourec2' and 'aws_instance.myapp_2', with completion times of 33s for each. The final summary is 'Apply complete! Resources: 2 added, 0 changed, 0 destroyed.'.

Output



The screenshot shows the AWS Management Console 'Instances' page. The table lists the instances created by the Terraform configuration, including 'jenkins', 'Terraform', 'myserver', 'terraform1', and 'myserver'. The table columns include Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Put.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Put
jenkins	i-0d7d0761e3bb7df38	Stopped	t2.small	-	View alarms +	ap-south-1a	-
Terraform	i-05a3932caa1b7821c	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a	ec2
myserver	i-0558fcd7cdf1ac19	Terminated	t2.micro	-	View alarms +	ap-south-1a	-
terraform1	i-054fed77a56d26f8	Terminated	t2.micro	-	View alarms +	ap-south-1a	-
terraform1	i-057174a1857e9ce76	Terminated	t2.micro	-	View alarms +	ap-south-1a	-
myserver	i-0e9af15f1117eb01e	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a	ec2

4. Preparing Documentation and Pushing to GitHub

Pushing to GitHub

1. **Initialize Git Repository:** Navigate to your project directory and initialize a git repository.

```
$ git init
$ git add .
$ git commit -m "Initial commit with Terraform configuration and documentation"
```

2. **Push to GitHub:** Push the local repository to GitHub.

```
$ git remote add origin <your-github-repository-url>
$ git push -u origin master
```

Example Repository Link

Here's an example repository link for reference: <https://github.com/yourusername/terraform-ec2-setup>