Terraform Import Automation using Python

TASK:

- Terraform Import Automation by using Python for VPCs and their resources.

STEP by STEP DEPLOYMENT:

- To automate the Terraform import of VPCs across regions using Python, we can approach it by using AWS SDK (boto3) for fetching the VPCs and Python's subprocess module to invoke terraform import commands. Here's an example of both the Python and Terraform code you would need.

STEP 1: To create the Terraform configuration for managing VPCs.

STEP 2: To create the main.tf for providing the necessary and dependency for VPCs configuration.

Main.tf

```
provider "aws" {
  region = var.aws_region
}

resource "aws_vpc" "imported_vpc" {
  # The block remains empty for now since we're importing
}

variable "aws_region" {
  description = "AWS region"
  type        = string
}

variable "vpc_id" {
  description = "VPC ID to import"
  type        = string
}
```

STEP 3: To provide the variable.tf for the main.tf to configure the terraform infrastructure.

Variables.tf

```
variable "aws_region" {
  description = "AWS region to use"
  type        = string
}

variable "vpc_id" {
  description = "VPC ID to import"
  type        = string
}
```

STEP 4:

- This Python script uses boto3 to list the VPCs and then automatically generates terraform import commands for each VPC. The script can handle VPCs across multiple regions.

terraform_import_vpc.py

```
import boto3
import subprocess

# List of AWS regions where you want to import VPCs
regions = ["us-east-1", "us-west-2"]  # Add other regions if needed

def get_vpcs(region):
    """
    Get list of VPCs in a specific AWS region
    """
    ec2_client = boto3.client("ec2", region_name=region)
    response = ec2_client.describe_vpcs()
    vpcs = response.get("Vpcs", [])
    return vpcs

def terraform_import(vpc_id, region):
    """
    Run the terraform import command to import the VPC
    """
    # Corrected the -var flag quoting
    terraform_cmd = f"terraform import -var=\"aws_region={region}\" -
var=\"vpc_id={vpc_id}\" aws_vpc.imported_vpc {vpc_id}"
    print(f"Running: {terraform_cmd} in region: {region}")
```

```
    # Run the Terraform import command
    subprocess.run(terraform_cmd, shell=True, check=True)

def main():
    for region in regions:
        print(f"Fetching VPCs from region: {region}")
        vpcs = get_vpcs(region)

        for vpc in vpcs:
            vpc_id = vpc.get("VpcId")
            print(f"Importing VPC: {vpc_id} from region: {region}")

            # Run terraform import
            terraform_import(vpc_id, region)

if __name__ == "__main__":
    main()
```

Workflow of the Script :

- Fetch VPCs: It uses boto3 to fetch the VPCs for each specified region.
- Terraform Plan and Import: It sets the necessary Terraform variables (region and VPC ID) and invokes terraform import using subprocess.run.
- Multiple Regions: You can modify the regions list to cover more AWS regions as needed.

STEP 6:

- Initialize Terraform in your working directory using **Terraform.init** command.

STEP 7:

- Run the python script by using **Python terraform_import_vpc.py** This will automatically fetch VPCs from the specified regions and run terraform import for each one**.**

Terraform Import Automation using Python

VERIFICATION:

```
C:\Users\Arun Muthukumarsamy\Desktop\Terra-Auto>terraform init -upgrade
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.69.0...
- Installed hashicorp/aws v5.69.0 (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:\Users\Arun Muthukumarsamy\Desktop\Terra-Auto>python terraform_import_vpc.py
Fetching VPCs from region: us-east-1
Importing VPC: vpc-0001c3c50a3435739 from region: us-east-1
Running: terraform import -var="aws_region=us-east-1" -var="vpc_id=vpc-0001c3c50a3435739" aws_vpc.imported_vpc vpc-0001c3c50a3435739 in region: us-east-1
aws_vpc.imported_vpc: Importing from ID "vpc-0001c3c50a3435739"...
aws_vpc.imported_vpc: Import prepared!
  Prepared aws_vpc for import
aws_vpc.imported_vpc: Refreshing state... [id=vpc-0001c3c50a3435739]

Import successful!

The resources that were imported are shown above. These resources are now in
your Terraform state and will henceforth be managed by Terraform.
```

```
Fetching VPCs from region: us-west-2
Importing VPC: vpc-0c089d44c05cefa6f from region: us-west-2
Running: terraform import -var="aws_region=us-west-2" -var="vpc_id=vpc-0c089d44c05cefa6f" aws_vpc.imported_vpc vpc-0c089d44c05cefa6f in region: us-west-2
aws_vpc.imported_vpc: Importing from ID "vpc-0c089d44c05cefa6f"...
aws_vpc.imported_vpc: Import prepared!
  Prepared aws_vpc for import

Error: Resource already managed by Terraform

Terraform is already managing a remote object for aws_vpc.imported_vpc. To import to this address you must first remove the existing object from the
state.


Traceback (most recent call last):
  File "C:\Users\Arun Muthukumarsamy\Desktop\Terra-Auto\terraform_import_vpc.py", line 40, in <module>
    main()
  File "C:\Users\Arun Muthukumarsamy\Desktop\Terra-Auto\terraform_import_vpc.py", line 37, in main
    terraform_import(vpc_id, region)
  File "C:\Users\Arun Muthukumarsamy\Desktop\Terra-Auto\terraform_import_vpc.py", line 25, in terraform_import
    subprocess.run(terraform_cmd, shell=True, check=True)
  File "C:\Users\Arun Muthukumarsamy\Lib\subprocess.py", line 571, in run
    raise CalledProcessError(retcode, process.args,
subprocess.CalledProcessError: Command 'terraform import -var="aws_region=us-west-2" -var="vpc_id=vpc-0c089d44c05cefa6f" aws_vpc.imported_vpc vpc-0c089d44c0
5cefa6f' returned non-zero exit status 1.
```