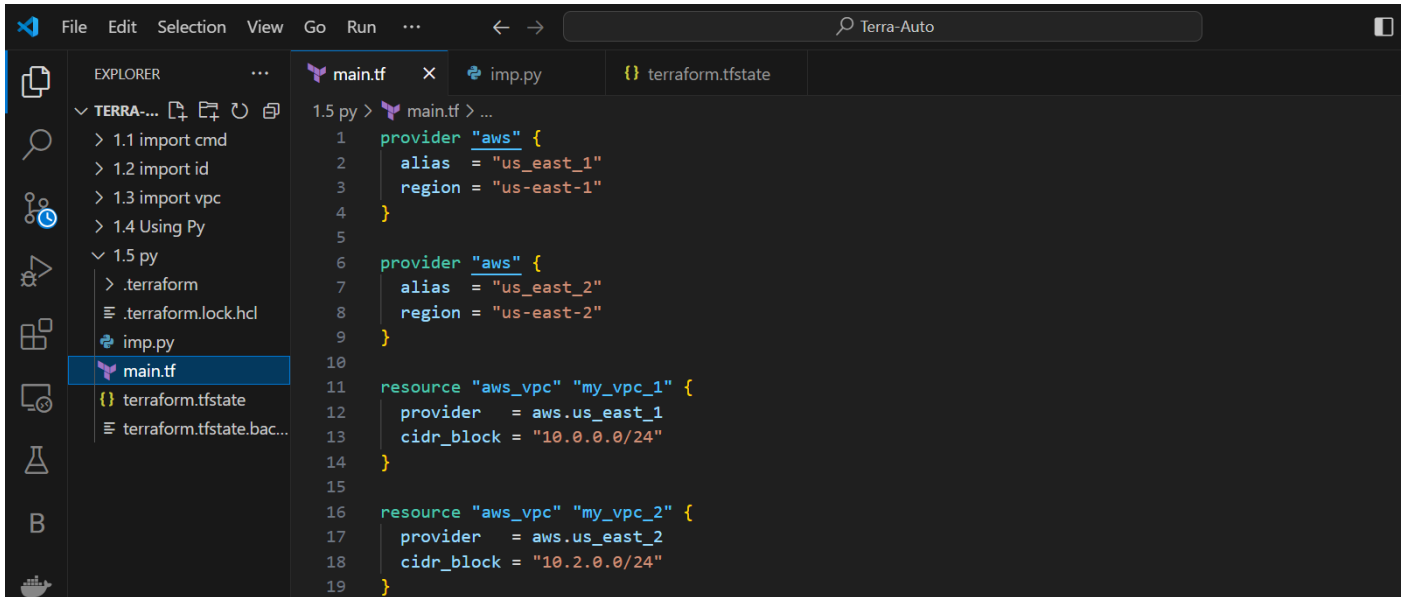


Import an existing VPC through Python

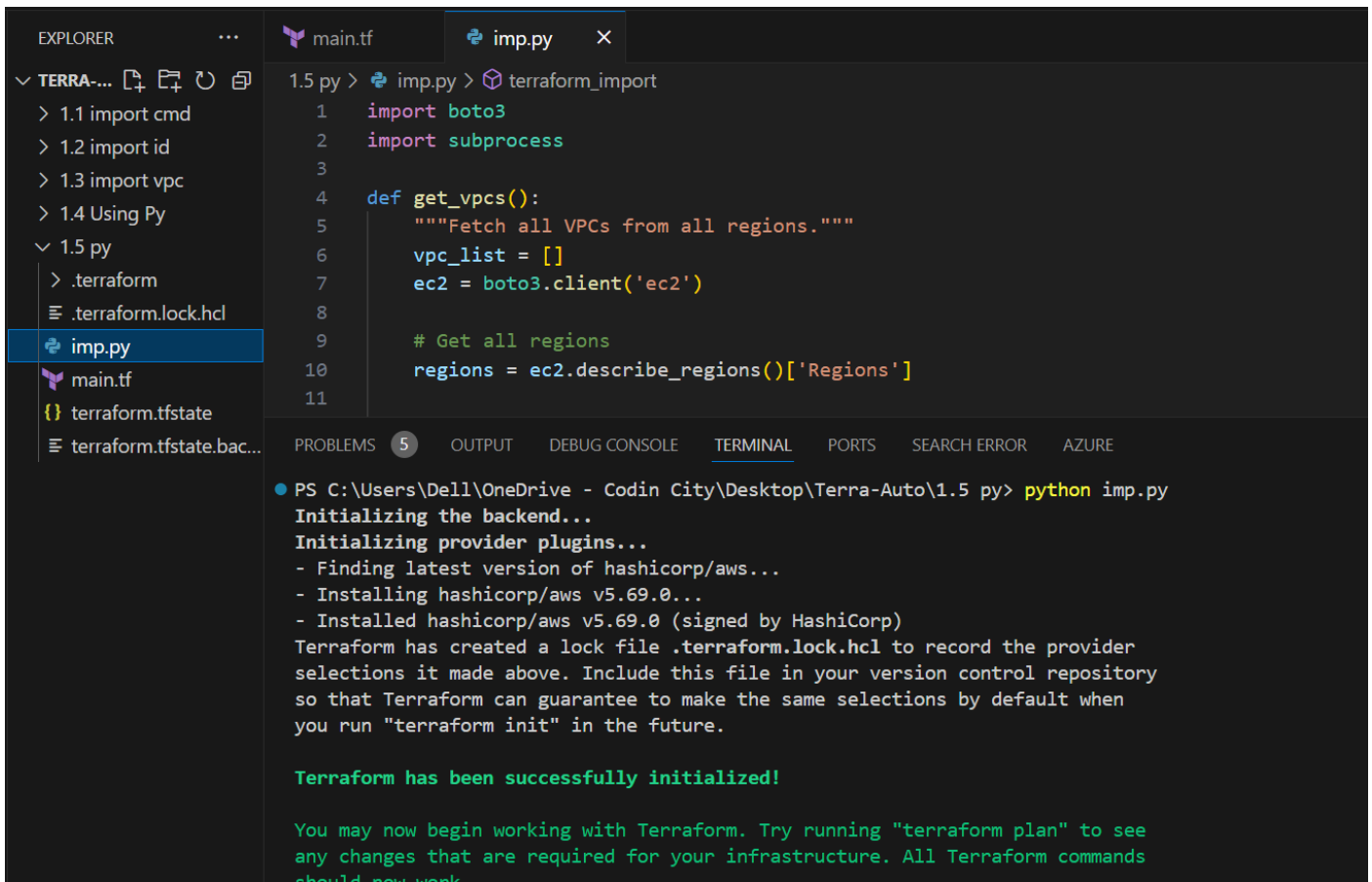
Task 1.4 → Using terraform import

Step:1 I mentioned the **main.tf** file for terraform configurations



```
1 provider "aws" {
2   alias   = "us_east_1"
3   region = "us-east-1"
4 }
5
6 provider "aws" {
7   alias   = "us_east_2"
8   region = "us-east-2"
9 }
10
11 resource "aws_vpc" "my_vpc_1" {
12   provider = aws.us_east_1
13   cidr_block = "10.0.0.0/24"
14 }
15
16 resource "aws_vpc" "my_vpc_2" {
17   provider = aws.us_east_2
18   cidr_block = "10.2.0.0/24"
19 }
```

Step:2 After running the **imp.py** file, it will automatically be initializing the terraform



```
1 import boto3
2 import subprocess
3
4 def get_vpcs():
5     """Fetch all VPCs from all regions."""
6     vpc_list = []
7     ec2 = boto3.client('ec2')
8
9     # Get all regions
10    regions = ec2.describe_regions()['Regions']
11
```

PS C:\Users\Dell\OneDrive - Codin City\Desktop\Terra-Auto\1.5 py> python imp.py

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.

Step:3 It will list out the VPC at specified regions

```
Checking region: us-east-1
Found VPC: vpc-0408de3b6d3e92705 in region: us-east-1
Found VPC: vpc-0ac3883de5bde45b6 in region: us-east-1
Checking region: us-east-2
Found VPC: vpc-001cedaa66b3d63f6 in region: us-east-2
```

Step:4 After it all VPC has been importing to the statefile

```
aws_vpc.my_vpc_1: Importing from ID "vpc-0408de3b6d3e92705"...
aws_vpc.my_vpc_1: Import prepared!
  Prepared aws_vpc for import
aws_vpc.my_vpc_1: Refreshing state... [id=vpc-0408de3b6d3e92705]

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.

Successfully imported aws_vpc.my_vpc_1 with ID vpc-0408de3b6d3e92705.
Resource aws_vpc.my_vpc_1 is already managed by Terraform. Skipping import.
aws_vpc.my_vpc_2: Importing from ID "vpc-001cedaa66b3d63f6"...
aws_vpc.my_vpc_2: Import prepared!
  Prepared aws_vpc for import
aws_vpc.my_vpc_2: Refreshing state... [id=vpc-001cedaa66b3d63f6]

Import successful!

The resources that were imported are shown above. These resources are now in
Import successful!
```

Step:5 We can see the statefile all resource has been imported

```
1.5 py > {} terraform.tfstate > [ ] resources > [ ] 0 > [ ] instances > [ ] 0 > [ ] attributes
1 {
2   "version": 4,
3   "terraform_version": "1.9.5",
4   "serial": 2,
5   "lineage": "4248d65a-d1b7-771b-067d-911c204531f4",
6   "outputs": {},
7   "resources": [
8     {
9       "mode": "managed",
10      "type": "aws_vpc",
11      "name": "my_vpc_1",
12      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"].us_east_1",
13      "instances": [
14        {
15          "schema_version": 1,
16          "attributes": {
17            "arn": "arn:aws:ec2:us-east-1:339713187727:vpc/vpc-0408de3b6d3e92705",
18            "assign_generated_ipv6_cidr_block": false,
19            "cidr_block": "10.0.0.0/24",
20            "default_network_acl_id": "acl-0be74e7ccccbce2c2",
21            "default_route_table_id": "rtb-0aa5d04ebd1493773",
22            "default_security_group_id": "sg-0a7d8d2c8be0a583b",
23            "dhcp_options_id": "dopt-09cd5f3382f696b6f",
24            "enable_dns_hostnames": false,
25            "enable_dns_support": true,
26            "enable_network_address_usage_metrics": false,
27            "id": "vpc-0408de3b6d3e92705",
28            "instance_tenancy": "default",
29            "ipv4_ipam_pool_id": null,
30            "ipv4_netmask_length": null,
```

Step:6 Also we can list out the state

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR AZURE

```
● PS C:\Users\Dell\OneDrive - Codin City\Desktop\Terra-Auto\1.5 py> terraform state list  
aws_vpc.my_vpc_1  
aws_vpc.my_vpc_2  
○ PS C:\Users\Dell\OneDrive - Codin City\Desktop\Terra-Auto\1.5 py> █
```