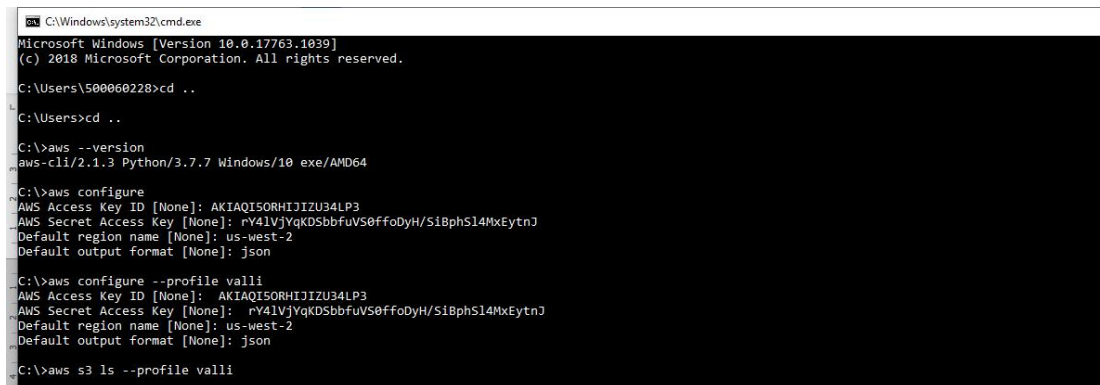


ASSIGNMENT 1

Terraform does all that for you by preparing the whole infrastructure using terraform scripts. Thus, it is a software tool that provides Infrastructure as code.

Download AWS CLI in the windows system. And install it.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.17763.1039]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\500060228>cd ..
C:\Users>cd ..
C:\Users>cd ..
C:\>aws --version
aws-cli/2.1.3 Python/3.7.7 Windows/10 exe/AMD64

C:\>aws configure
AWS Access Key ID [None]: AKIAQISORHIJIZU34LP3
AWS Secret Access Key [None]: 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
Default region name [None]: us-west-2
Default output format [None]: json

C:\>aws configure --profile valli
AWS Access Key ID [None]: AKIAQISORHIJIZU34LP3
AWS Secret Access Key [None]: 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
Default region name [None]: us-west-2
Default output format [None]: json

C:\>aws s3 ls --profile valli
```

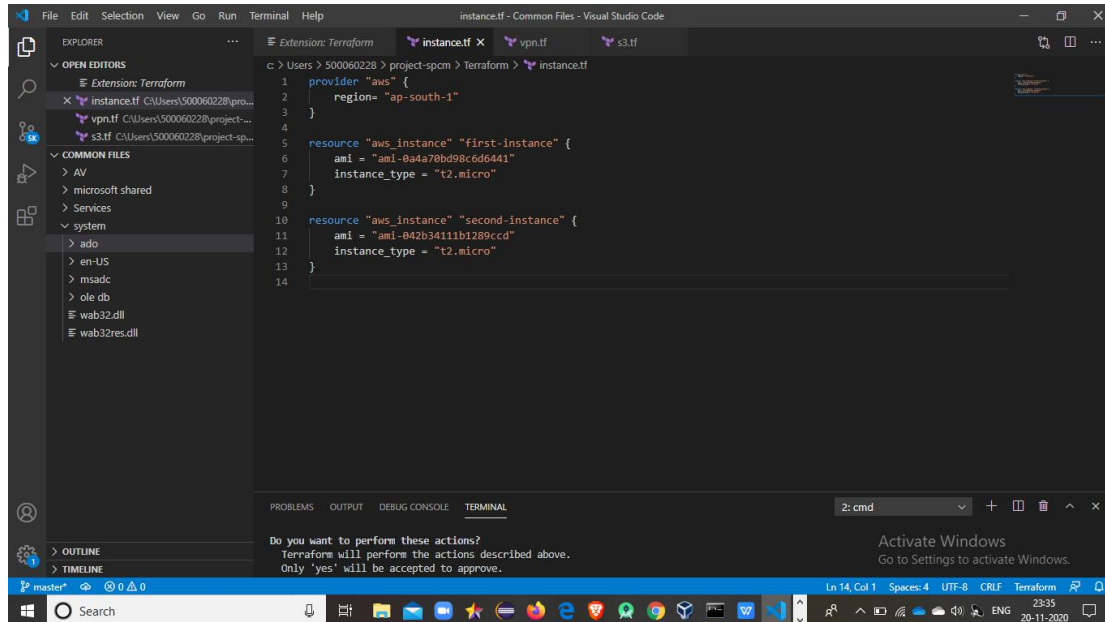
1. Download the terraform zip file, extract it in a folder. Create a path in the env variables.

Here I have extracted in -

C:\Users\500060228\project-spcm\Terraform

2. Open the visual studio code and install the “Terraform” extension.

3. Create a file instance.tf and write code for creating 2 instances in a particular region. Create two T2 Micro EC2 Instances.

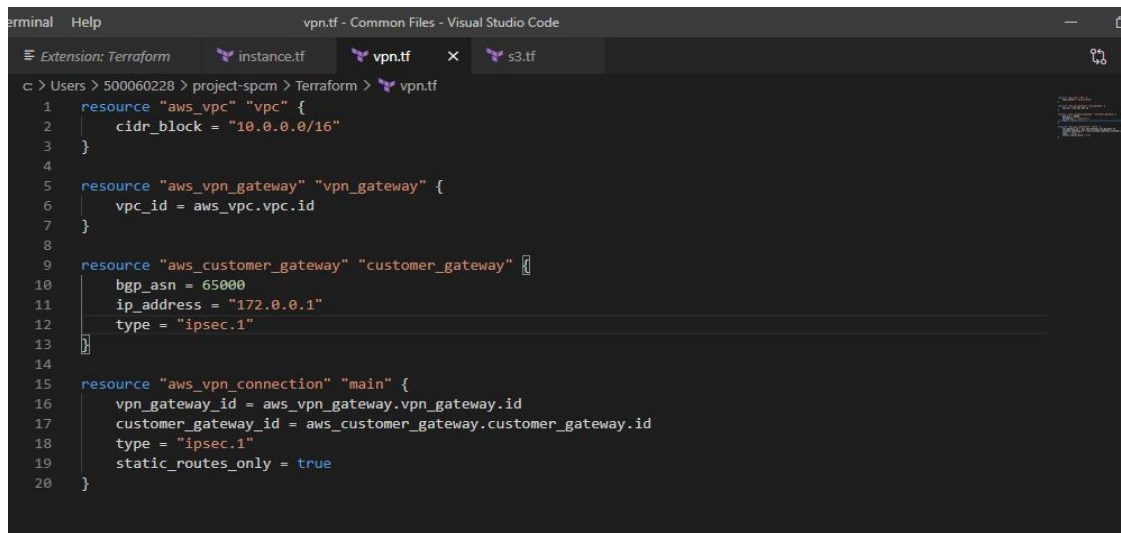


The screenshot shows the Visual Studio Code interface with the 'instance.tf' file open. The file contains Terraform code for creating two AWS EC2 instances. The code is as follows:

```
1 provider "aws" {
2   region = "ap-south-1"
3 }
4
5 resource "aws_instance" "first-instance" {
6   ami = "ami-0a4a78bd98c6d6441"
7   instance_type = "t2.micro"
8 }
9
10 resource "aws_instance" "second-instance" {
11   ami = "ami-042b3411b1289ccd"
12   instance_type = "t2.micro"
13 }
14
```

The Explorer sidebar on the left shows the file structure, including 'instance.tf', 'vpn.tf', and 's3.tf'. The Terminal at the bottom shows a message from Windows: 'Do you want to perform these actions? Terraform will perform the actions described above. Only 'yes' will be accepted to approve.'

4. Now create another file “vpn.tf” to create the vpc in the aws.

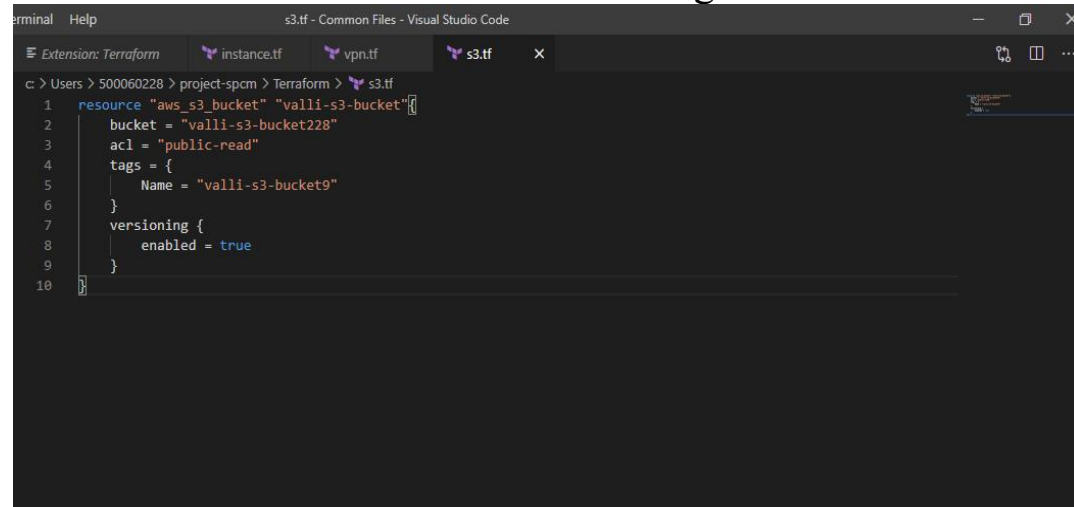


The screenshot shows the Visual Studio Code interface with the 'vpn.tf' file open. The file contains Terraform code for creating AWS VPC and VPN resources. The code is as follows:

```
1 resource "aws_vpc" "vpc" {
2   cidr_block = "10.0.0.0/16"
3 }
4
5 resource "aws_vpn_gateway" "vpn_gateway" {
6   vpc_id = aws_vpc.vpc.id
7 }
8
9 resource "aws_customer_gateway" "customer_gateway" {
10  bgp_asn = 65000
11  ip_address = "172.0.0.1"
12  type = "ipsec.1"
13 }
14
15 resource "aws_vpn_connection" "main" {
16  vpn_gateway_id = aws_vpn_gateway.vpn_gateway.id
17  customer_gateway_id = aws_customer_gateway.customer_gateway.id
18  type = "ipsec.1"
19  static_routes_only = true
20 }
```

The Explorer sidebar on the left shows the file structure, including 'instance.tf', 'vpn.tf', and 's3.tf'. The Terminal at the bottom shows a message from Windows: 'Do you want to perform these actions? Terraform will perform the actions described above. Only 'yes' will be accepted to approve.'

5. Create another file “s3.tf” for creating a s3 bucket in the aws.

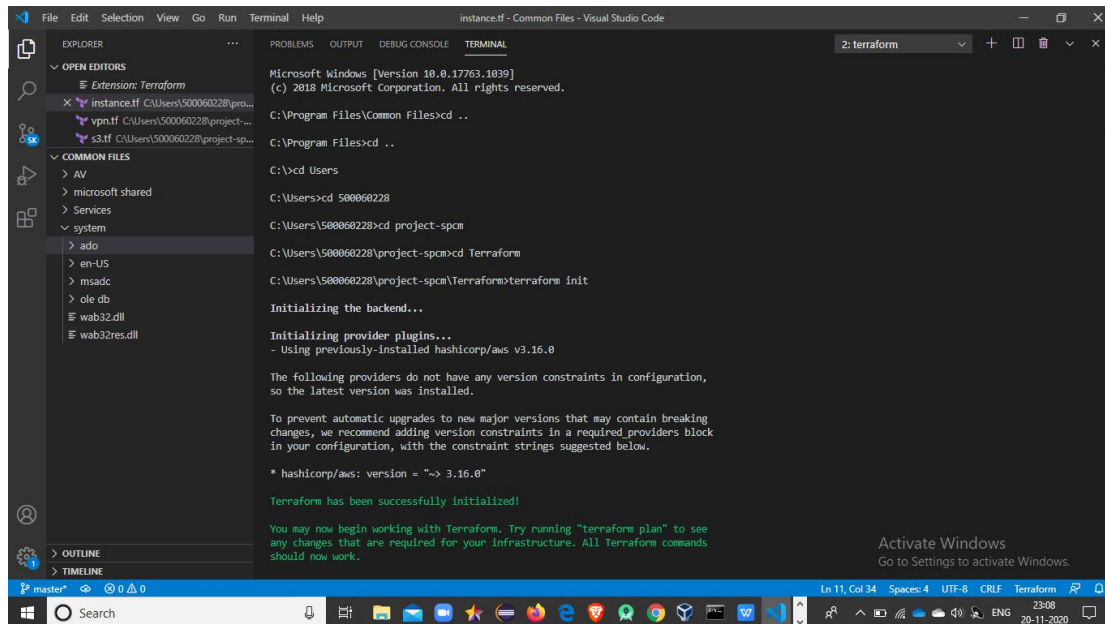
A screenshot of the Visual Studio Code editor interface. The top bar shows the file path 's3.tf - Common Files - Visual Studio Code'. The left sidebar has tabs for 'Extension: Terraform', 'instance.tf', 'vpn.tf', and 's3.tf'. The main editor area displays the content of 's3.tf' with line numbers 1 through 10. The code is a Terraform resource definition for an AWS S3 bucket.

```
1 resource "aws_s3_bucket" "valli-s3-bucket" {  
2     bucket = "valli-s3-bucket228"  
3     acl = "public-read"  
4     tags = {  
5         Name = "valli-s3-bucket9"  
6     }  
7     versioning {  
8         enabled = true  
9     }  
10 }
```

6. Now to run the “.tf” files , run the following commands-

- ✓ Terraform init: initiates the terraform files in the particular location/path
- ✓ Terraform validate: checks the files whether it is having any errors or not.
- ✓ Terraform plan: It creates an execution plan that helps you check whether the execution plan matches your expectations.
- ✓ Terraform apply: applies the changes to the desired state.

Init



The screenshot shows the Visual Studio Code interface with the 'instance.tf' file open. The Explorer pane on the left shows the file structure. The Terminal pane on the right shows the output of the 'terraform init' command. The output indicates that the backend is being initialized, provider plugins are being installed, and the configuration is valid. The status bar at the bottom shows the file is at line 11, column 34, with 4 spaces, UTF-8 encoding, and CRLF line endings.

```
Microsoft Windows [Version 10.0.17763.1039]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Program Files\Common Files>cd ..
C:\Program Files>cd ..
C:\>cd Users
C:\Users>cd 500060228
C:\Users\500060228>cd project-spcm
C:\Users\500060228\project-spcm>cd Terraform
C:\Users\500060228\project-spcm\Terraform>terraform init

Initializing the backend...

Initializing provider plugins...
- Using previously-installed hashicorp/aws v3.16.0

The following providers do not have any version constraints in configuration,
so the latest version was installed.

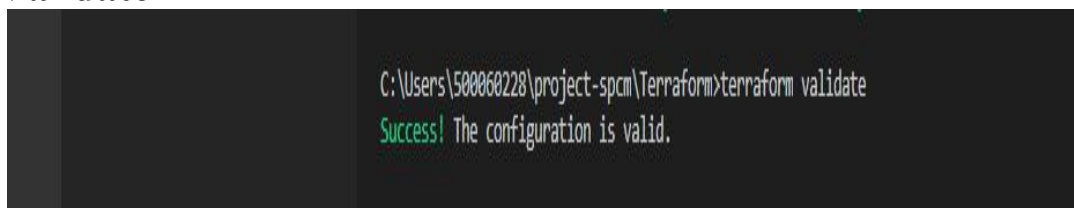
To prevent automatic upgrades to new major versions that may contain breaking
changes, we recommend adding version constraints in a required providers block
in your configuration, with the constraint strings suggested below.

* hashicorp/aws: version = "~> 3.16.0"

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.
```

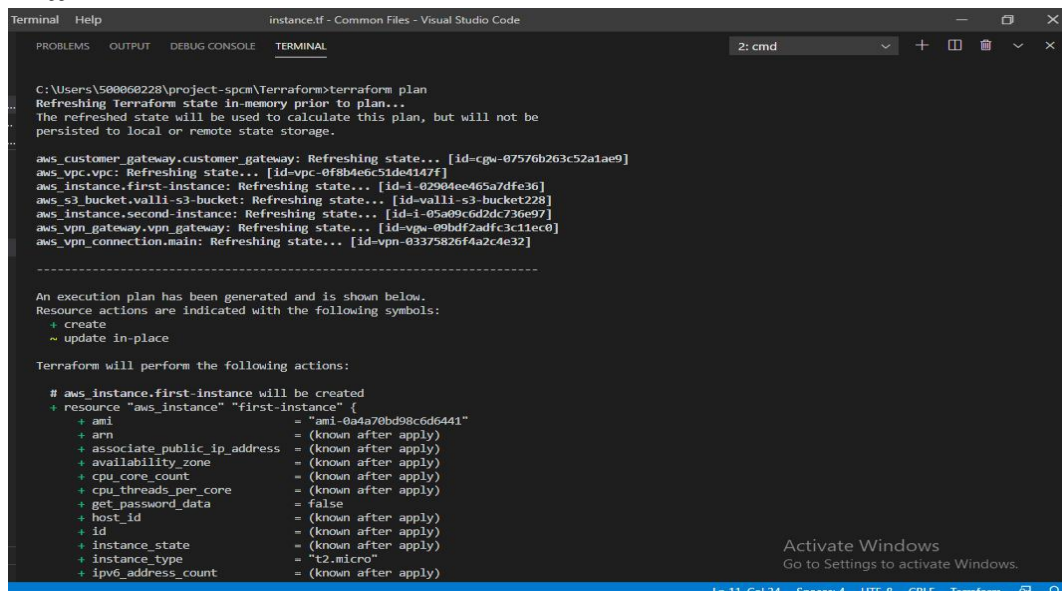
Validate



The screenshot shows a terminal window with the command 'C:\Users\500060228\project-spcm\Terraform>terraform validate' and the output 'Success! The configuration is valid.'

```
C:\Users\500060228\project-spcm\Terraform>terraform validate
Success! The configuration is valid.
```

Plan



The screenshot shows the Visual Studio Code interface with the 'instance.tf' file open. The Terminal pane on the right shows the output of the 'terraform plan' command. The output indicates that the state is being refreshed and the plan is being generated. The plan shows that the 'aws_instance.first-instance' resource will be created. The status bar at the bottom shows the file is at line 11, column 34, with 4 spaces, UTF-8 encoding, and CRLF line endings.

```
C:\Users\500060228\project-spcm\Terraform>terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.

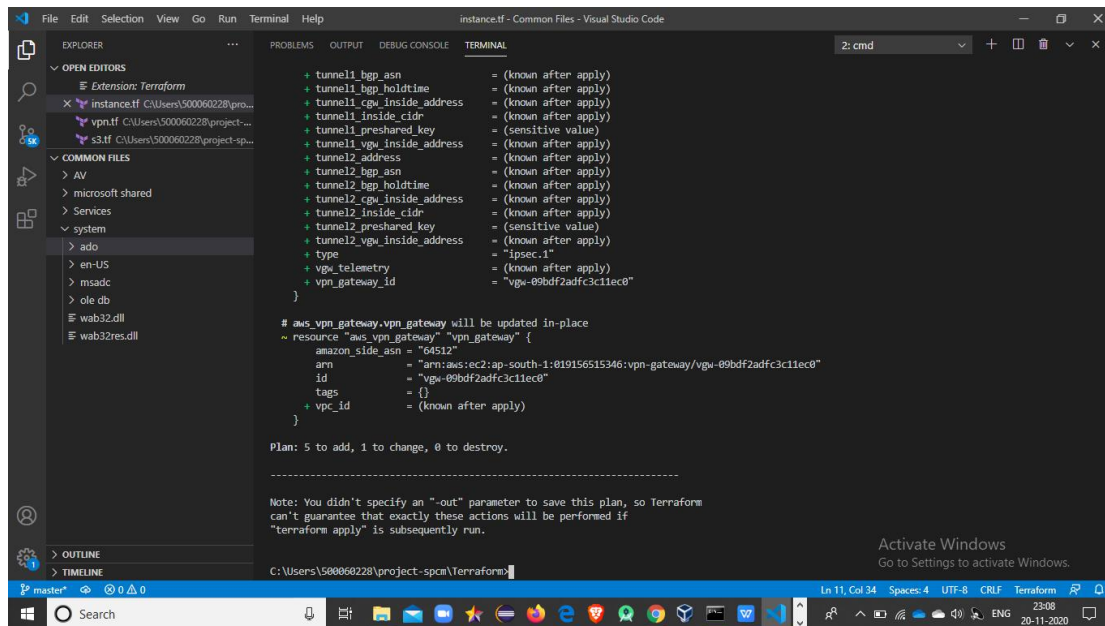
aws_customer_gateway.customer_gateway: Refreshing state... [id=cgw-07576b263c52a1ae9]
aws_vpc.vpc: Refreshing state... [id=vpc-0f8b4e6c51de4147f]
aws_instance.first-instance: Refreshing state... [id=i-02904ee465a7dfe36]
aws_s3_bucket.valli-s3-bucket: Refreshing state... [id=valli-s3-bucket228]
aws_instance.second-instance: Refreshing state... [id=i-05a09c6d2dc736e97]
aws_vpn_gateway.vpn_gateway: Refreshing state... [id=vgw-09bdf2adfc3c11ec0]
aws_vpn_connection.main: Refreshing state... [id=vpn-03375826f4a2c4e32]

-----

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
+ create
~ update in-place

Terraform will perform the following actions:

# aws_instance.first-instance will be created
+ resource "aws_instance" "first-instance" {
+   ami               = "ami-0a4a70bd98c6d6441"
+   arn               = (known after apply)
+   associate_public_ip_address = (known after apply)
+   availability_zone  = (known after apply)
+   cpu_core_count     = (known after apply)
+   cpu_threads_per_core = (known after apply)
+   get_password_data  = false
+   host_id            = (known after apply)
+   id                 = (known after apply)
+   instance_state     = (known after apply)
+   instance_type      = "t2.micro"
+   ipv6_address_count = (known after apply)
}
```



```
File Edit Selection View Go Run Terminal Help
instance.tf - Common Files - Visual Studio Code

EXPLORER
  OPEN EDITORS
    Extension: Terraform
    instance.tf C:\Users\S00060228\proj...
    vpn.tf C:\Users\S00060228\project-sp...
    s3.tf C:\Users\S00060228\project-sp...
  COMMON FILES
    > AV
    > microsoft shared
    > Services
    > system
    > ado
    > en-US
    > msadc
    > ole db
    wab32res.dll
    wab32res.dll
  OUTLINE
  TIMELINE

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
2: cmd

+ tunnel1_bgp_asn = (known after apply)
+ tunnel1_bgp_holdtime = (known after apply)
+ tunnel1_cgw_inside_address = (known after apply)
+ tunnel1_inside_cidr = (known after apply)
+ tunnel1_preshared_key = (sensitive value)
+ tunnel1_vgw_inside_address = (known after apply)
+ tunnel2_address = (known after apply)
+ tunnel2_bgp_asn = (known after apply)
+ tunnel2_bgp_holdtime = (known after apply)
+ tunnel2_cgw_inside_address = (known after apply)
+ tunnel2_inside_cidr = (known after apply)
+ tunnel2_preshared_key = (sensitive value)
+ tunnel2_vgw_inside_address = (known after apply)
+ type = "ipsec.1"
+ vgw_telemetry = (known after apply)
+ vpn_gateway_id = "vgw-09bdf2adfc3c11ec0"
}

# aws_vpn_gateway.vpn_gateway will be updated in-place
~ resource "aws_vpn_gateway" "vpn_gateway" {
  amazon_side_asn = "64512"
  arn = "arn:aws:ec2:ap-south-1:019156515346:vpn-gateway/vgw-09bdf2adfc3c11ec0"
  id = "vgw-09bdf2adfc3c11ec0"
  tags = {}
  vpc_id = (known after apply)
}

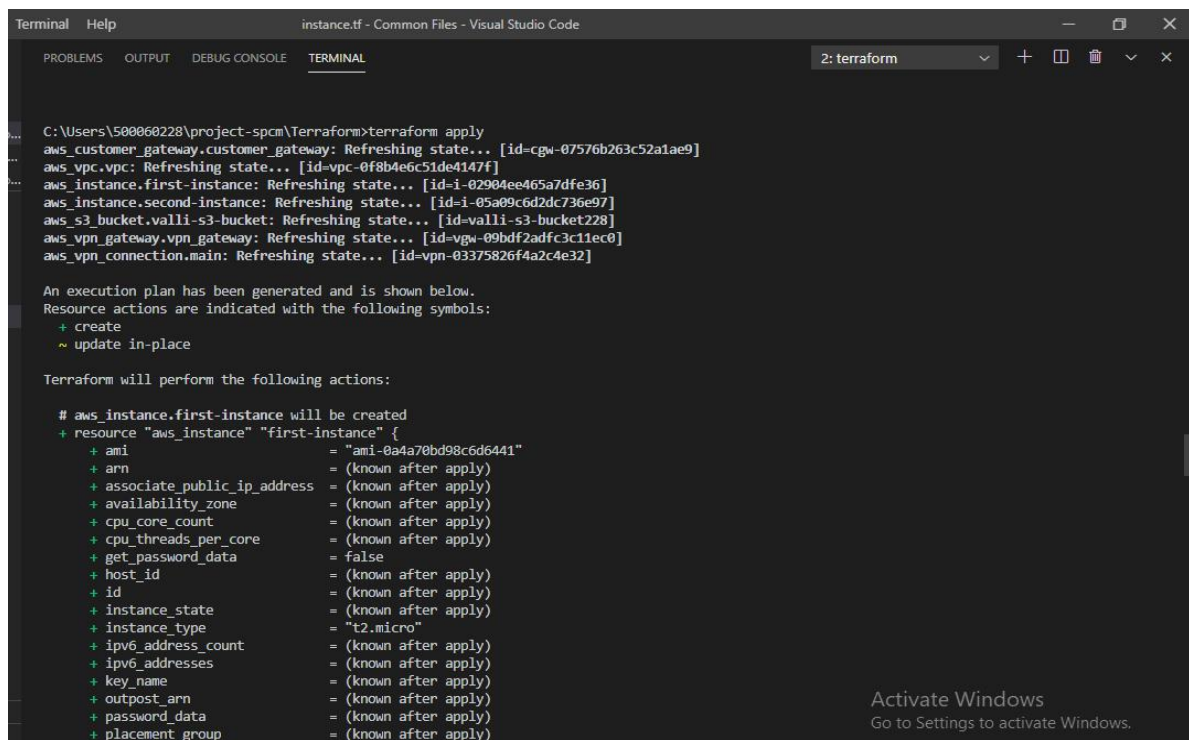
Plan: 5 to add, 1 to change, 0 to destroy.

Note: You didn't specify an "-out" parameter to save this plan, so Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.

Activate Windows
Go to Settings to activate Windows.

C:\Users\S00060228\project-sp\Terraform
```

Apply (There was a small error while running the cmd at first time for creating instances, it is rectified and the “terraform apply” cmd is passed again)



```
Terminal Help
instance.tf - Common Files - Visual Studio Code

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
2: terraform

C:\Users\S00060228\project-sp\Terraform>terraform apply
aws_customer_gateway.customer_gateway: Refreshing state... [id-cgw-07576b263c52a1ae9]
aws_vpc.vpc: Refreshing state... [id-vpc-0f8b4e6c51de4147f]
aws_instance.first-instance: Refreshing state... [id-i-02904ee465a7dfe36]
aws_instance.second-instance: Refreshing state... [id-i-05a09c6d2dc736e97]
aws_s3_bucket.vall-i-s3-bucket: Refreshing state... [id-vall-i-s3-bucket228]
aws_vpn_gateway.vpn_gateway: Refreshing state... [id-vgw-09bdf2adfc3c11ec0]
aws_vpn_connection.main: Refreshing state... [id-vpn-03375826f4a2c4e32]

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
+ create
~ update in-place

Terraform will perform the following actions:

# aws_instance.first-instance will be created
+ resource "aws_instance" "first-instance" {
+ ami = "ami-0a4a70bd98c6d6441"
+ arn = (known after apply)
+ associate_public_ip_address = (known after apply)
+ availability_zone = (known after apply)
+ cpu_core_count = (known after apply)
+ cpu_threads_per_core = (known after apply)
+ get_password_data = false
+ host_id = (known after apply)
+ id = (known after apply)
+ instance_state = (known after apply)
+ instance_type = "t2.micro"
+ ipv6_address_count = (known after apply)
+ ipv6_addresses = (known after apply)
+ key_name = (known after apply)
+ outpost_arn = (known after apply)
+ password_data = (known after apply)
+ placement_group = (known after apply)
}
```

The screenshot shows the Visual Studio Code interface with the Terraform extension. The Explorer pane on the left shows the project structure with files like `instance.tf`, `vpn.tf`, and `s3.tf`. The main editor displays the `instance.tf` file with the following content:

```
1 provider "aws" {
```

The terminal window shows the output of the `terraform apply` command. It starts with "Acquiring state lock. This may take a few moments..." followed by a list of resources being refreshed: `aws_customer_gateway.customer_gateway`, `aws_s3_bucket.valli-s3-bucket`, `aws_vpc.vpc`, `aws_vpn_gateway.vpn_gateway`, and `aws_vpn_connection.main`. The output then shows the execution plan, indicating that a new `aws_instance` resource will be created. The plan details the configuration for the first instance, including attributes like `ami`, `arn`, `associate_public_ip_address`, `availability_zone`, `cpu_core_count`, `cpu_threads_per_core`, `get_password_data`, `host_id`, `id`, `instance_state`, `instance_type`, `ipv6_address_count`, `ipv6_addresses`, `key_name`, and `outpost_arn`.

The screenshot shows the Visual Studio Code interface with the Terraform extension. The Explorer pane on the left shows the project structure with files like `instance.tf`, `vpn.tf`, and `s3.tf`. The main editor displays the `instance.tf` file with the following content:

```
1 provider "aws" {
```

The terminal window shows the output of the `terraform apply` command. It starts with "Acquiring state lock. This may take a few moments..." followed by a list of resources being refreshed: `aws_customer_gateway.customer_gateway`, `aws_s3_bucket.valli-s3-bucket`, `aws_vpc.vpc`, `aws_vpn_gateway.vpn_gateway`, and `aws_vpn_connection.main`. The output then shows the execution plan, indicating that a new `aws_instance` resource will be created. The plan details the configuration for the first instance, including attributes like `ami`, `arn`, `associate_public_ip_address`, `availability_zone`, `cpu_core_count`, `cpu_threads_per_core`, `get_password_data`, `host_id`, `id`, `instance_state`, `instance_type`, `ipv6_address_count`, `ipv6_addresses`, `key_name`, and `outpost_arn`.

OUTPUTS ON AWS:

The image displays two screenshots of the AWS Management Console. The top screenshot shows the EC2 Instances page, and the bottom screenshot shows the Amazon S3 Buckets page.

Top Screenshot: EC2 Instances

The top screenshot shows the AWS Management Console interface for the EC2 Instances page. The left sidebar contains navigation links for EC2 Dashboard, Events, Tags, Limits, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, and Elastic Block Store. The main content area displays a table of instances with the filter 'Instance state: running' applied. The table shows two instances, both in the 'Running' state, with instance IDs i-0b34c9dbfb18b0021 and i-047d251589e1d5194, both using t2.micro instance types. The status check for both instances is 'Initializing', and there are no alarms. The availability zone for both instances is ap-south-1b.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
-	i-0b34c9dbfb18b0021	Running	t2.micro	Initializing	No alarms	ap-south-1b
-	i-047d251589e1d5194	Running	t2.micro	Initializing	No alarms	ap-south-1b

Bottom Screenshot: Amazon S3 Buckets

The bottom screenshot shows the AWS Management Console interface for the Amazon S3 Buckets page. The left sidebar contains navigation links for Buckets, Access points, Batch Operations, Access analyzer for S3, Account settings for Block Public Access, Storage Lens, Dashboards, AWS Organizations settings, and Feature spotlight. The main content area displays a table of buckets with the filter 'Find buckets by name' applied. The table shows one bucket, 'valli-s3-bucket228', in the 'Asia Pacific (Mumbai) ap-south-1' region, with 'Public' access and a creation date of November 20, 2020, 23:09 (UTC+05:30).

Name	Region	Access	Creation date
valli-s3-bucket228	Asia Pacific (Mumbai) ap-south-1	Public	November 20, 2020, 23:09 (UTC+05:30)

ter (2) Sy R1 Dc SP Dc Df Int AV Ge Int Co An Ut La Fv De x + -

ap-south-1.console.aws.amazon.com/vpc/home?region=ap-south-1#vpcs:

aws Services

srisaivalli Mumbai Support

New VPC Experience

Tell us what you think

VPC Dashboard

New

Filter by VPC:

Select a VPC

VIRTUAL PRIVATE CLOUD

Your VPCs

New

Subnets

New

Route Tables

Internet Gateways

New

Egress Only Internet Gateways

New

DHCP Options Sets

New

Elastic IPs

New

Managed Prefix Lists

New

Endpoints

Endpoint Services

NAT Gateways

New

Your VPCs (2) Info

Actions

Create VPC

Filter VPCs

< 1 >

	Name	VPC ID	State	IPv4 CIDR	IPv6 C
<input type="checkbox"/>	-	vpc-0a241bbb797ae59cf	Available	10.0.0.0/16	-
<input type="checkbox"/>	-	vpc-0d11254b54b404590	Available	172.31.0.0/16	-

Select a VPC above

Activate Windows
Go to Settings to activate Windows.

Feedback English (US)

© 2008 - 2020 Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Search

23:37

20-11-2020