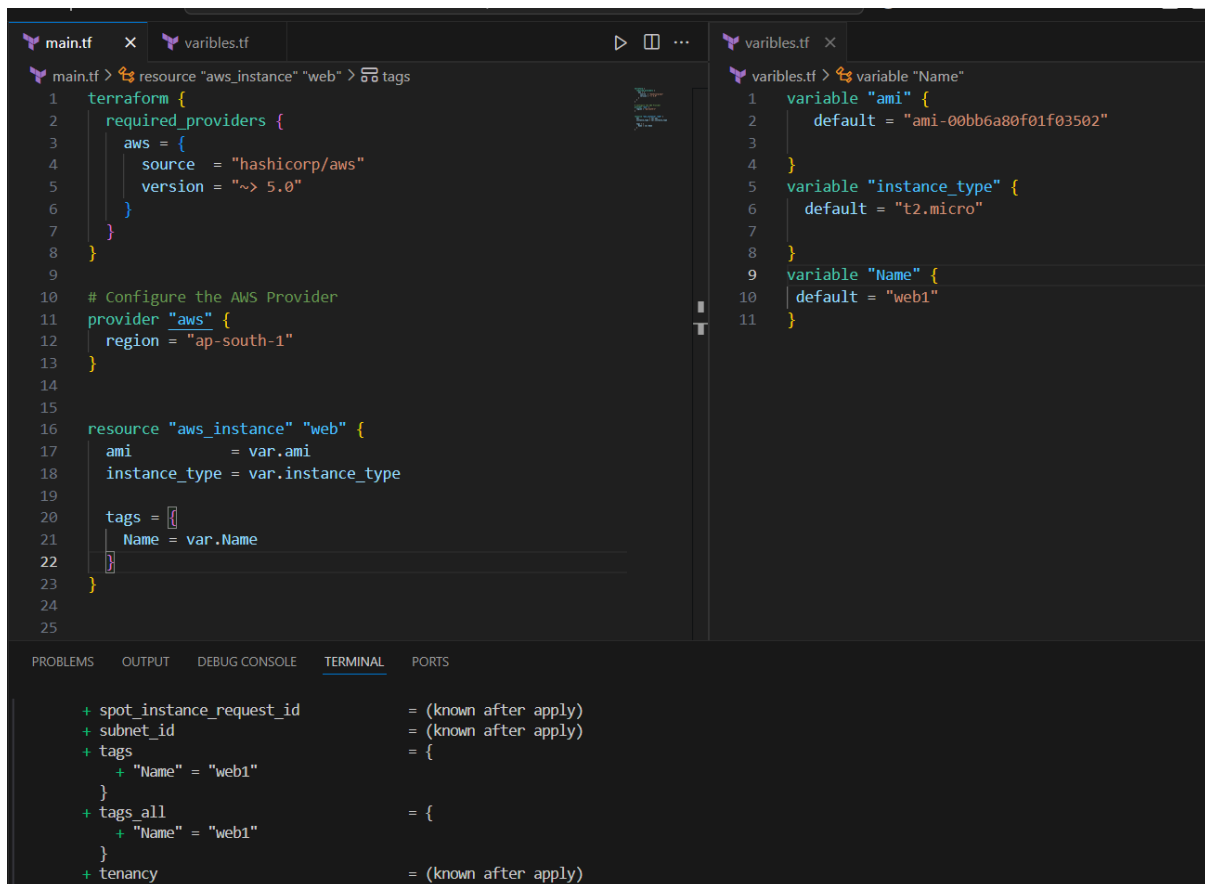


Provision EC2 with main file with default variables from variables.tf



The screenshot shows an IDE with two Terraform files and a terminal output.

main.tf

```
1 terraform {
2   required_providers {
3     aws = {
4       source = "hashicorp/aws"
5       version = "~> 5.0"
6     }
7   }
8 }
9
10 # Configure the AWS Provider
11 provider "aws" {
12   region = "ap-south-1"
13 }
14
15
16 resource "aws_instance" "web" {
17   ami           = var.ami
18   instance_type = var.instance_type
19
20   tags = {
21     Name = var.Name
22   }
23 }
24
25
```

variables.tf

```
1 variable "ami" {
2   default = "ami-00bb6a80f01f03502"
3 }
4
5 variable "instance_type" {
6   default = "t2.micro"
7 }
8
9 variable "Name" {
10  default = "web1"
11 }
```

Terminal Output

```
+ spot_instance_request_id      = (known after apply)
+ subnet_id                    = (known after apply)
+ tags                          = {
+   + "Name" = "web1"
+ }
+ tags_all                     = {
+   + "Name" = "web1"
+ }
+ tenancy                       = (known after apply)
```

Provision EC2 with main file without default variables.tf it asks inputs from user for required values.

```
main.tf  variables.tf X
variables.tf > variable "Name"
1  variable "ami" {
2      # default = "ami-00bb6a80f01f03502"
3
4  }
5  variable "instance_type" {
6      # default = "t2.micro"
7
8  }
9  variable "Name" {
10     # default = "web1"
11 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

rerun this command to reinitialize your working directory. If you forget, other commands will detect it and remind you to do so if necessary.

DELL@TECH-SIRT MINGW64 ~/OneDrive/Desktop/terraform (master)

\$ terraform plan

var.Name
Enter a value: web1

var.ami
Enter a value: ami-00bb6a80f01f03502

var.instance_type
Enter a value: t2.micro

Terraform used the selected providers to generate the following execution plan. Resource ac
+ create

Create terraform.tfvars file and define variables on it and run terraform plan, variables picked from terraform.tfvars files

The screenshot shows a VS Code editor with three files open: `main.tf`, `variables.tf`, and `terraform.tfvars`. The `variables.tf` file contains the following code:

```
1 variable "ami" {
2   default = "ami-00bb6a80f01f03502"
3 }
4
5 variable "instance_type" {
6   default = "t2.micro"
7 }
8
9 variable "Name" {
10  default = "web1"
11 }
```

The `terraform.tfvars` file contains the following code:

```
1 ami="ami-00bb6a80f01f03502"
2 instance_type = "t3.micro"
3 Name="web22"
```

The terminal window shows the output of a Terraform command, listing various attributes and their values. The output is as follows:

```
+ instance_type           = "t3.micro"
+ ipv6_address_count      = (known after apply)
+ ipv6_addresses          = (known after apply)
+ key_name                = (known after apply)
+ monitoring              = (known after apply)
+ outpost_arn             = (known after apply)
+ password_data           = (known after apply)
+ placement_group         = (known after apply)
+ placement_partition_number = (known after apply)
+ primary_network_interface_id = (known after apply)
+ private_dns             = (known after apply)
+ private_ip              = (known after apply)
+ public_dns              = (known after apply)
+ public_ip               = (known after apply)
+ secondary_private_ips   = (known after apply)
+ security_groups         = (known after apply)
+ source_dest_check       = true
+ spot_instance_request_id = (known after apply)
+ subnet_id               = (known after apply)
+ tags                    = {
+   + "Name" = "web22"
+ }
+ tags_all                = {
+   + "Name" = "web22"
```

Create `testing.tfvars` & `prod.tfvars` file and define environmental specific variables on them and pass *.tfvars file during run time and notice what values are being picked during run time.

The screenshot shows three Terraform variable files in VS Code: `testing.tfvars`, `variables.tf`, and `prod.tfvars`. `testing.tfvars` contains `ami="ami-00bb6a80f01f03502"`, `instance_type = "t2.micro"`, and `Name="web3"`. `variables.tf` defines `ami` with a default of `"ami-00bb6a80f01f03502"`, `instance_type` with a default of `"t2.micro"`, and `Name` with a default of `"web1"`. `prod.tfvars` contains `ami="ami-00bb6a80f01f03502"`, `instance_type = "t2.micro"`, and `Name="web2"`. The terminal shows the command `$ terraform plan -var-file="testing.tfvars"` and the output indicating that `aws_instance.web` will be created with the specified AMI and instance type, and the tag `Name` will be `web3`.

```
testing.tfvars > Name
1 ami="ami-00bb6a80f01f03502"
2 instance_type = "t2.micro"
3 Name="web3"

variables.tf > variable "Name"
1 variable "ami" {
2   default = "ami-00bb6a80f01f03502"
3 }
4
5 variable "instance_type" {
6   default = "t2.micro"
7 }
8
9 variable "Name" {
10  default = "web1"
11 }

prod.tfvars > Name
1 ami="ami-00bb6a80f01f03502"
2 instance_type = "t2.micro"
3 Name="web2"

DELL@TECH-SIRI MINGW64 ~/OneDrive/Desktop/terraform (master)
$ terraform plan -var-file="testing.tfvars"

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# aws_instance.web will be created
+ resource "aws_instance" "web" {
+   ami           = "ami-00bb6a80f01f03502"
+   arn           = (known after apply)
```

The screenshot shows the same three Terraform variable files as the first image. The terminal shows the command `$ terraform plan -var-file="testing.tfvars"` and the output indicating that `aws_instance.web` will be created with the specified AMI and instance type, and the tag `Name` will be `web3`. The output also shows the `tags` and `tags_all` blocks, both with `Name` set to `web3`.

```
testing.tfvars > Name
1 ami="ami-00bb6a80f01f03502"
2 instance_type = "t2.micro"
3 Name="web3"

variables.tf > variable "Name"
1 variable "ami" {
2   default = "ami-00bb6a80f01f03502"
3 }
4
5 variable "instance_type" {
6   default = "t2.micro"
7 }
8
9 variable "Name" {
10  default = "web1"
11 }

prod.tfvars > Name
1 ami="ami-00bb6a80f01f03502"
2 instance_type = "t2.micro"
3 Name="web2"

+ security_groups           = (known after apply)
+ source_dest_check         = true
+ spot_instance_request_id  = (known after apply)
+ subnet_id                 = (known after apply)
+ tags                      = {
+   "Name" = "web3"
+ }
+ tags_all                  = {
+   "Name" = "web3"
+ }
+ tenancy                   = (known after apply)
+ user_data                 = (known after apply)
```