Terraform
Workspace in Terraform
Terraform vault --> Scenario based on Terraform

Environments of the project:
Dev, QA, UAT, Pilot, Production

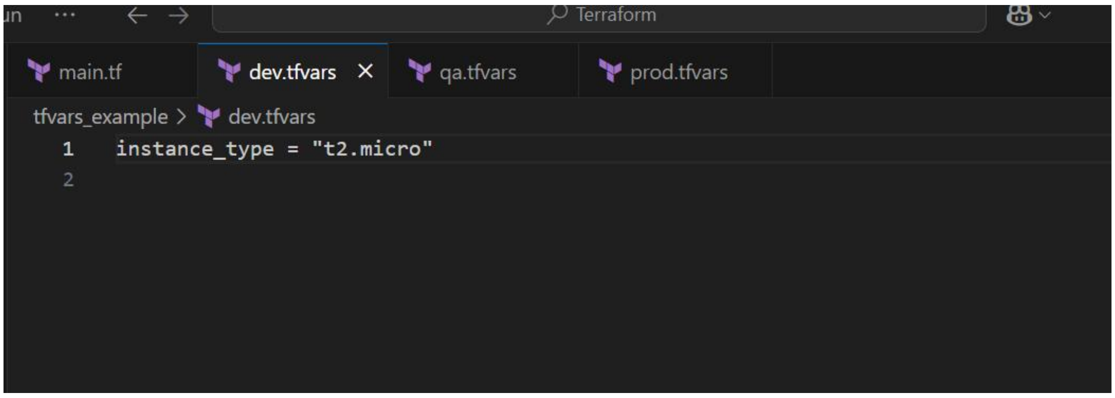
Environment refers to a platform or setup that's required to run our application (Servers, Database, Storage, Networking,…)

Generally we have multiple environments to run our projects

Say I want to use instance_type = "t2.micro" for Dev, "t2.medium" for Production , like different instance_types for different environments. If you run the Terraform script, a new instance is created. If "terraform.tfstate" file is already there, then when we re-apply or re-run the same scripts, will a new instance be created? No

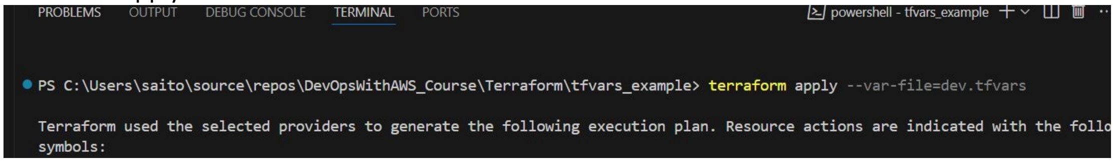
I want to use the same script but want to create different resources for different environments. Then we have the concept .tfvars

dev.tfvars, qa.tfvars, prod.tfvars --> different instance_type for different environments

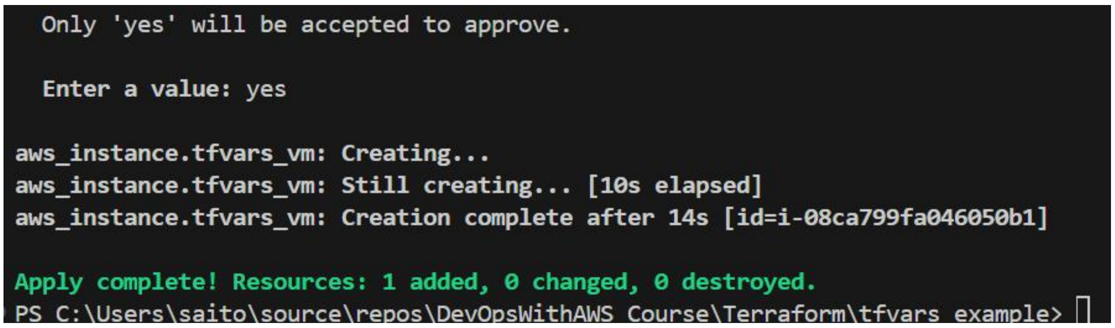


```
instance_type = "t2.micro"
```

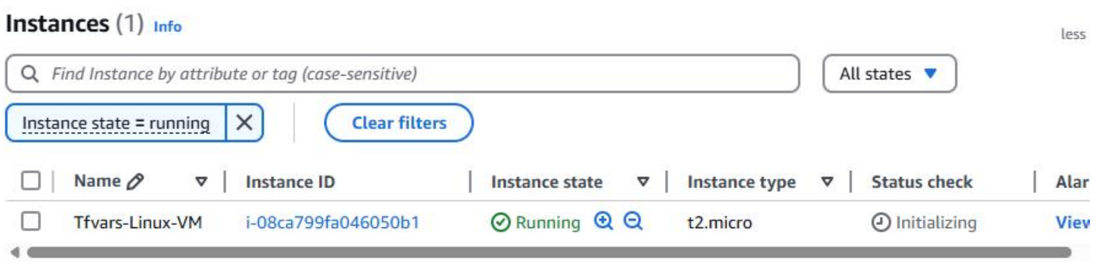
Dynamically pass the variable values
 terraform apply --var-file=dev.tfvars



```
PS C:\Users\saito\source\repos\DevOpsWithAWS_Course\Terraform\tfvars_example> terraform apply --var-file=dev.tfvars

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



```
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.tfvars_vm: Creating...
aws_instance.tfvars_vm: Still creating... [10s elapsed]
aws_instance.tfvars_vm: Creation complete after 14s [id=i-08ca799fa046050b1]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\saito\source\repos\DevOpsWithAWS_Course\Terraform\tfvars_example>
```


Instances (1) Info

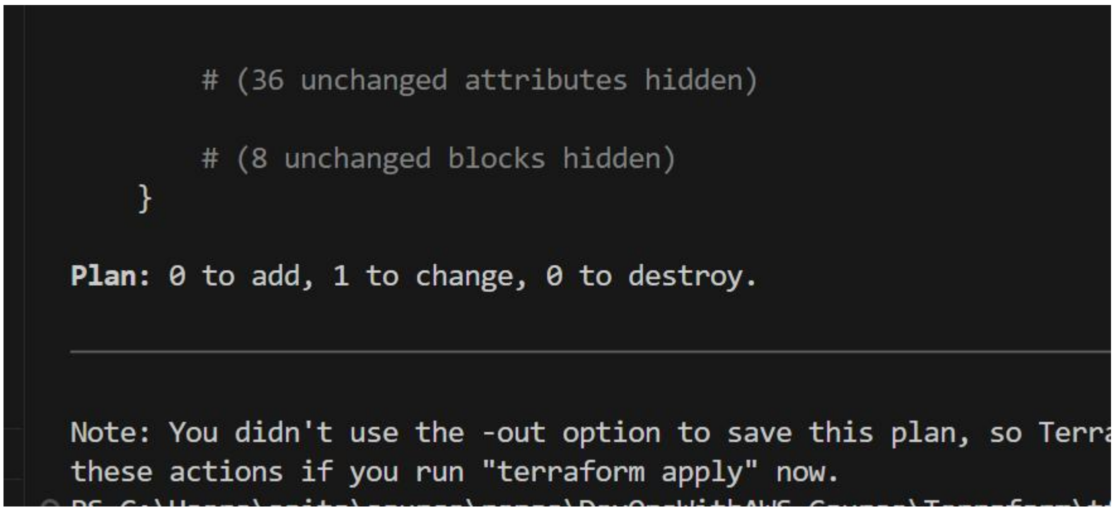
	Name	Instance ID	Instance state	Instance type	Status check	Alar
☐	Tfvars-Linux-VM	i-08ca799fa046050b1	Running	t2.micro	Initializing	View

See instance_type=”t2.micro”

We have created different ‘tfvars’ files for different environments but the state file is shared.

terraform plan --var-file=qa.tfvars

Then when I do a PLAN it is only trying to change the existing resource not ADD a new resource instead because of the common state file

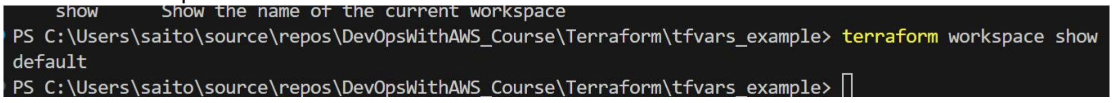

```
        # (36 unchanged attributes hidden)

        # (8 unchanged blocks hidden)
    }

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

─────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terra
these actions if you run "terraform apply" now.
```

What’s the solution? Workspace in Terraform
Dev environment --> t2.micro
QA environment --> t2.medium
Prod environment --> t2.xlarge

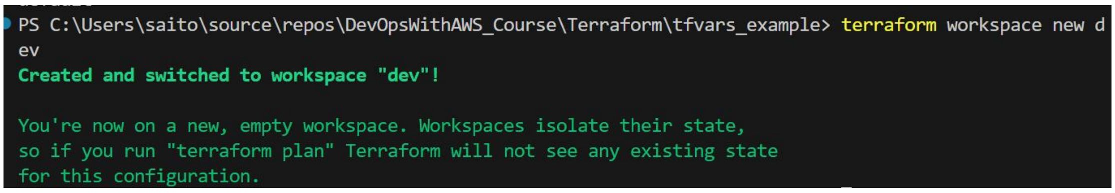
Now I want to create different state files for different environment, how’s this possible? **Workspace**

To manage infrastructure for Multiple environments, we will go with concept of Terraform workspace
If we go with Workspace concept then it wil maintain separate state files for every workspace

==> We can execute same script for multiple environments
Currently only one workspace is there: default
terraform workspace show

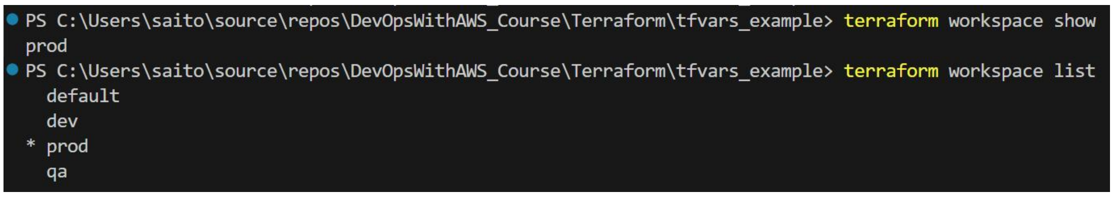

```
    show        Show the name of the current workspace
PS C:\Users\saito\source\repos\DevOpsWithAWS_Course\Terraform\tfvars_example> terraform workspace show
default
PS C:\Users\saito\source\repos\DevOpsWithAWS_Course\Terraform\tfvars_example>
```

terraform workspace new dev --> create a new workspace for dev



```
PS C:\Users\saito\source\repos\DevOpsWithAWS_Course\Terraform\tfvars_example> terraform workspace new d
ev
Created and switched to workspace "dev"!

You're now on a new, empty workspace. Workspaces isolate their state,
so if you run "terraform plan" Terraform will not see any existing state
for this configuration.
```

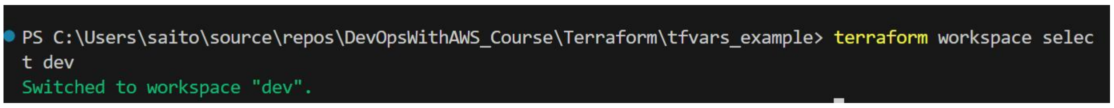
terraform workspace new qa
terraform workspace new prod



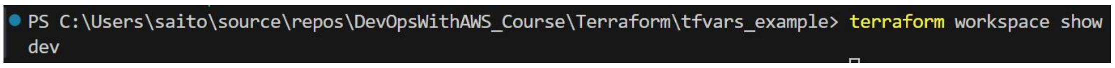
```
PS C:\Users\saito\source\repos\DevOpsWithAWS_Course\Terraform\tfvars_example> terraform workspace show
prod
PS C:\Users\saito\source\repos\DevOpsWithAWS_Course\Terraform\tfvars_example> terraform workspace list
  default
  dev
* prod
  qa
```

terraform workspace list

terraform workspace select dev
Switched to workspace "dev".



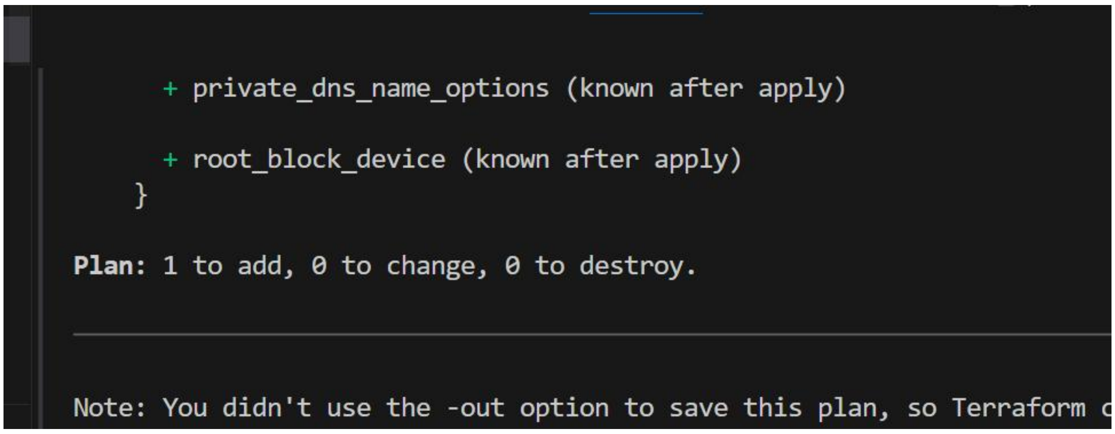
```
PS C:\Users\saito\source\repos\DevOpsWithAWS_Course\Terraform\tfvars_example> terraform workspace selec
t dev
Switched to workspace "dev".
```



```
PS C:\Users\saito\source\repos\DevOpsWithAWS_Course\Terraform\tfvars_example> terraform workspace show
dev
```

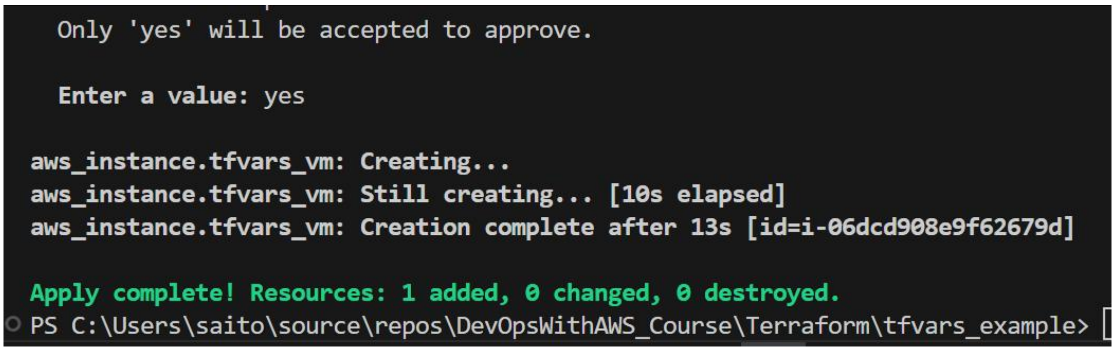
terraform plan --var-file=dev.tfvars

Now see Plan is 1 to add

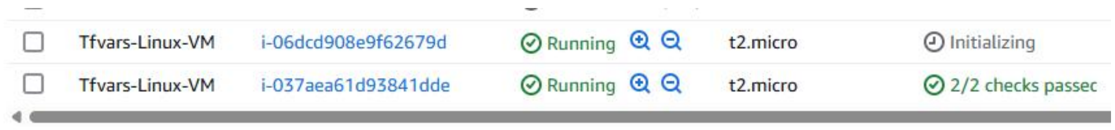


```
      + private_dns_name_options (known after apply)

      + root_block_device (known after apply)
    }

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

─────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform c
```

terraform apply --var-file=dev.tfvars



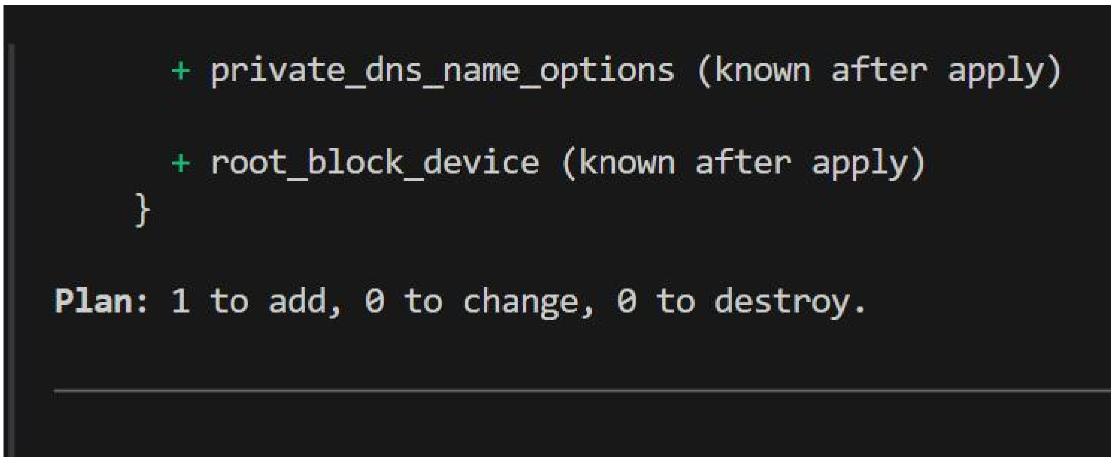
```
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.tfvars_vm: Creating...
aws_instance.tfvars_vm: Still creating... [10s elapsed]
aws_instance.tfvars_vm: Creation complete after 13s [id=i-06dcd908e9f62679d]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\saito\source\repos\DevOpsWithAWS_Course\Terraform\tfvars_example>
```



☐	Tfvars-Linux-VM	i-06dcd908e9f62679d	✓ Running	t2.micro	Initializing
☐	Tfvars-Linux-VM	i-037aea61d93841dde	✓ Running	t2.micro	✓ 2/2 checks passec

terraform workspace select qa
Switched to workspace "qa".

terraform plan --var-file=qa.tfvars



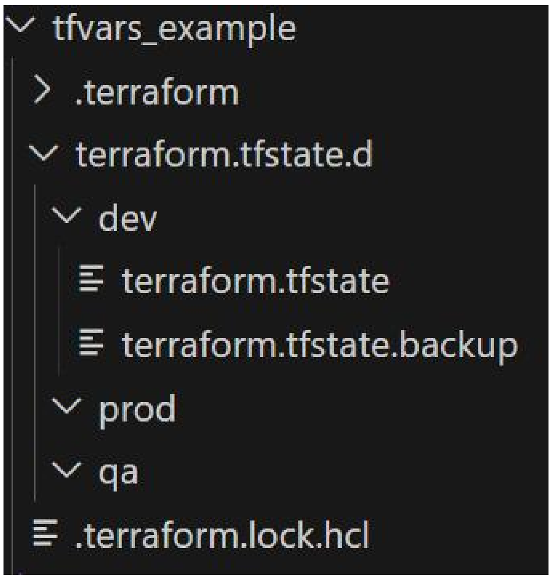
```
      + private_dns_name_options (known after apply)

      + root_block_device (known after apply)
    }

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

terraform workspace select dev
Switched to workspace "dev".

terraform destroy --auto-approve

Commands:
terraform workspace show --> shows current workspace
terraform workspace list --> show list of workspace
terraform workspace new dev --> create new 'dev' workspace
terraform workspace new qa --> creates new 'qa' workspace
terraform workspace select dev --> it will go to 'dev' workspace
terraform apply --var-file=dev.tfvars

We can see here it creates multiple State files



```
tfvars_example
  .terraform
  terraform.tfstate.d
    dev
      terraform.tfstate
      terraform.tfstate.backup
    prod
    qa
  .terraform.lock.hcl
```

Infrastructure as a Code (IaC)

Terraform setup (Linux and Windows)
Terraform architecture
Terraform scripts (HCL)
Variables (Input variables, Output variables)
EC2 VM
S3 Buckets
IAM, VPC, RDS

Terraform modules:
State file, lockfile
Resource taint and untaint
Terraform workspace
Terraform vault