Lab Exercise 7– Terraform Varia
Line Arguments

Objective:

Learn how to pass values to Terraform variables usin

Prerequisites:

- Terraform installed on your machine.
- Basic knowledge of Terraform variables.

Steps:

1. Create a Terraform Directory:

1. Create a Terraform Directory:

```
mkdir terraform-cli-variables

cd terraform-cli-variables
```

```
(base) → MyLab mkdir terrafo
cd terraform-cli-variables

(base) → terraform-cli-varia
```

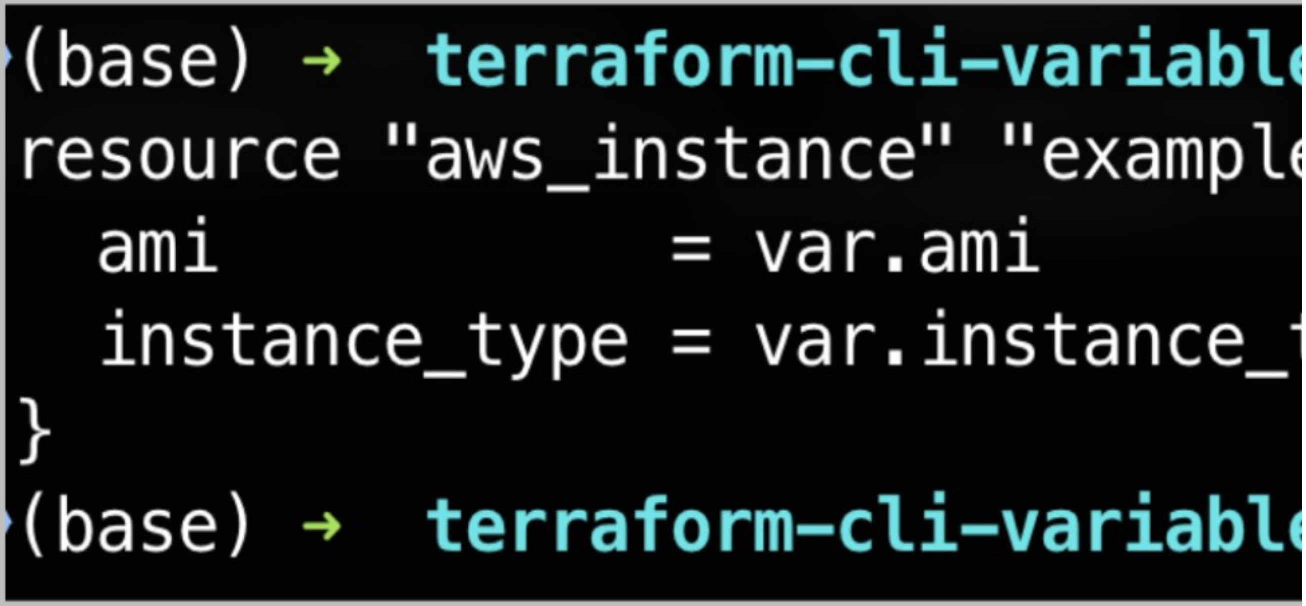
2. Create Terraform Configuratio

- Create a file named main.tf:

instance.tf

```
resource "aws_instance" "example" {
  ami           = var.ami
```

```
 instance_type = var.instance_type
}
```



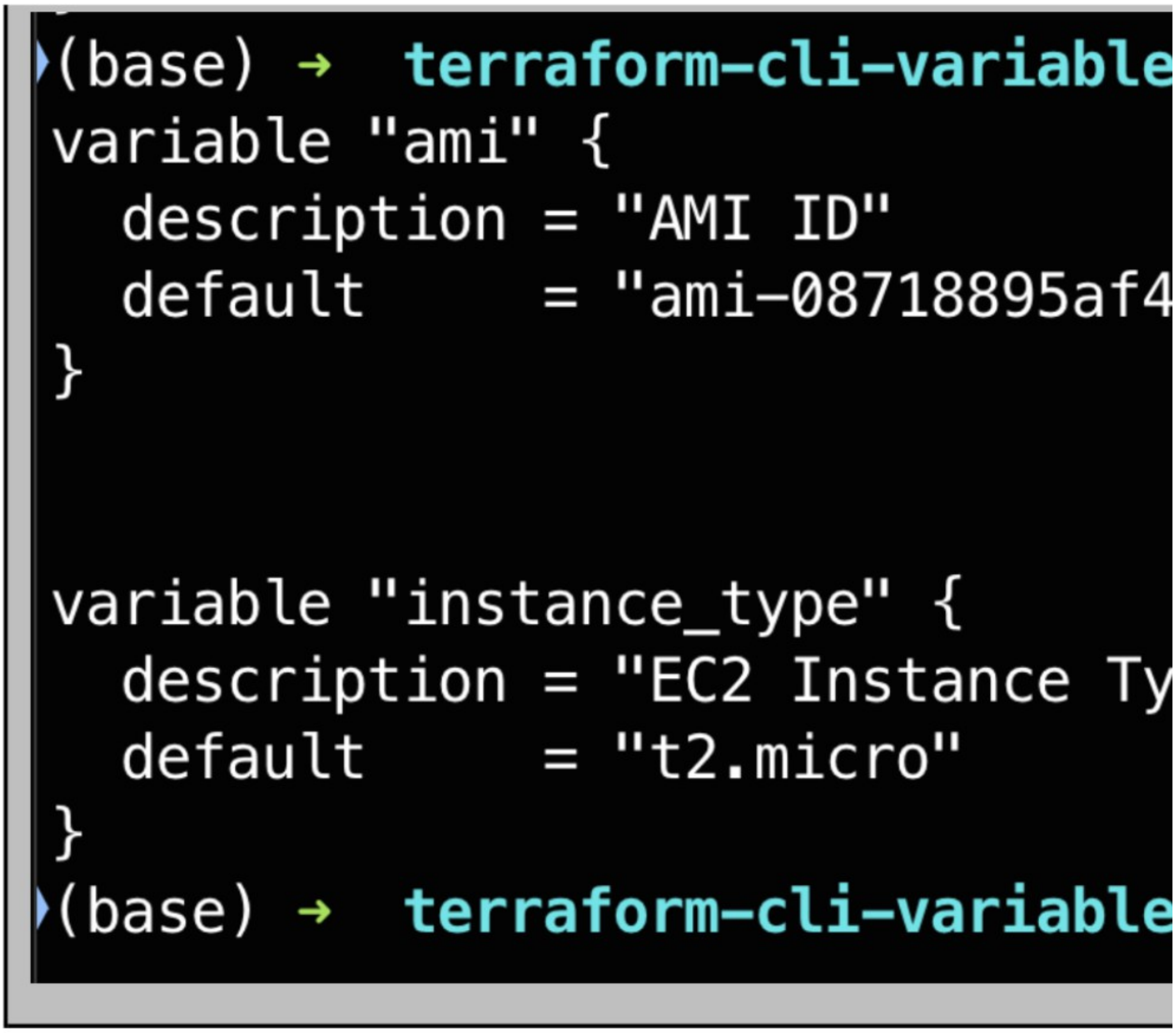
```
(base) → terraform-cli-variable
resource "aws_instance" "exampl
  ami           = var.ami
  instance_type = var.instance_
}
(base) → terraform-cli-variable
```

- Create a file named variables.tf:

```
# variables.tf
```

```
variable "ami" {
  description = "AMI ID"
  default     = " ami-08718895af4dfa033"
}

variable "instance_type" {
  description = "EC2 Instance Type"
  default     = "t2.micro"
}
```



```
(base) ➜  terraform-cli-variable
variable "ami" {
  description = "AMI ID"
  default     = "ami-08718895af4
}


variable "instance_type" {
  description = "EC2 Instance Ty
  default     = "t2.micro"
}
(base) ➜  terraform-cli-variable
```

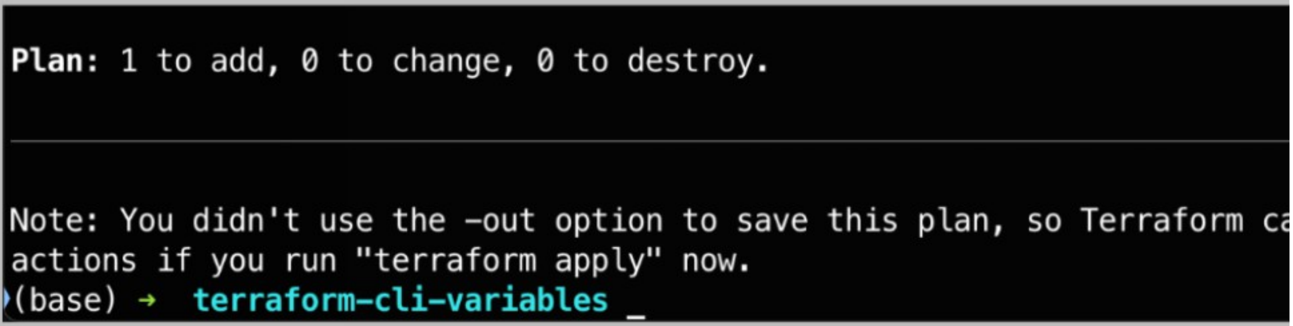
3. Use Command Line Arguments:

- Open a terminal and navigate to your Terrafor
- Run the terraform init command:

```
terraform init
```

- Run the terraform apply command with co
  variable values:

```
terraform plan -var=”ami=ami-0522ab6e1ddcc7055” -v
```

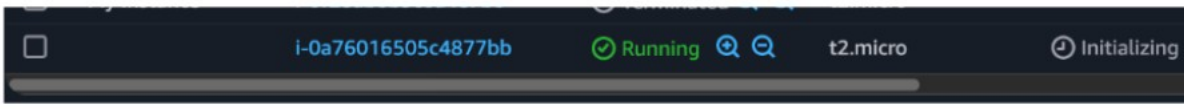


```
Plan: 1 to add, 0 to change, 0 to destroy.

Note: You didn't use the -out option to save this plan, so Terraform ca
actions if you run "terraform apply" now.
(base) ➜  terraform-cli-variables _
```

- Adjust the values based on your preferences.

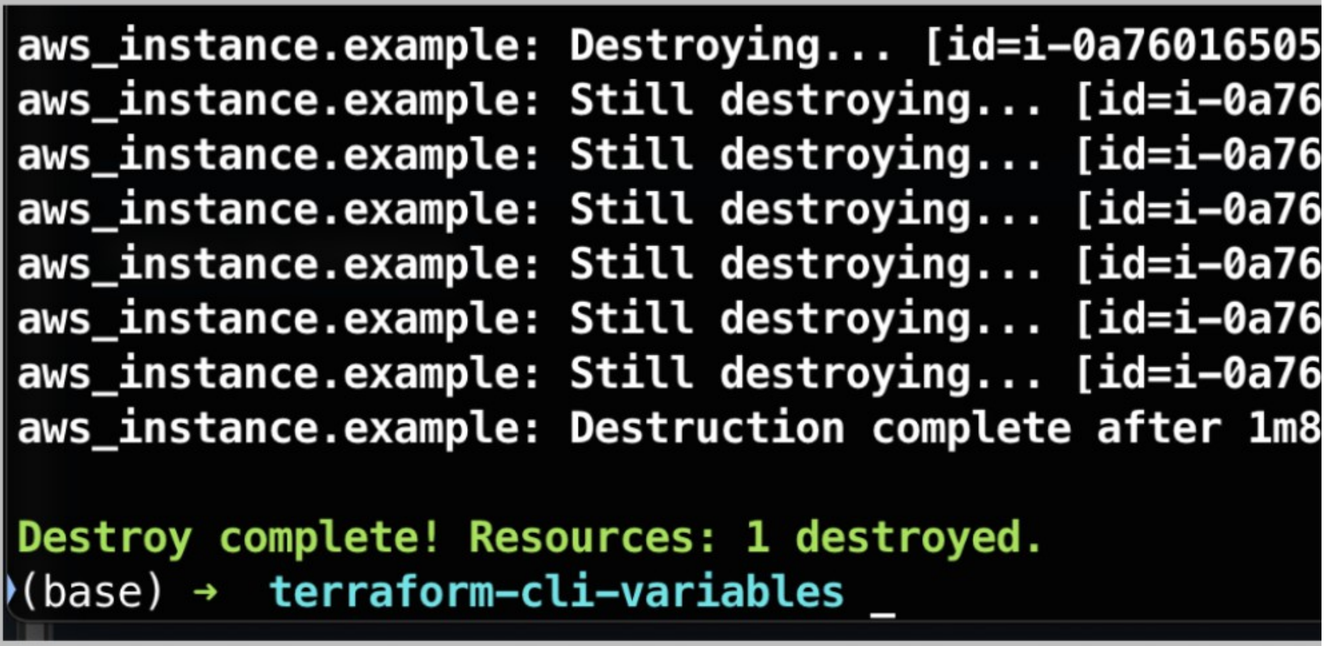
4. Test and Verify:

- Observe how the command line arguments dyna
  during the apply process.
-
- Access the AWS Management Console or use
  creation of resources in the specified region.
- 
  i-0a76016505c4877bb Running t2.micro Initializing

5. Clean Up:

After testing, you can clean up resources:

```
terraform destroy
```



```
aws_instance.example: Destroying... [id=i-0a76016505
aws_instance.example: Still destroying... [id=i-0a76
aws_instance.example: Still destroying... [id=i-0a76
aws_instance.example: Still destroying... [id=i-0a76
aws_instance.example: Still destroying... [id=i-0a76
aws_instance.example: Still destroying... [id=i-0a76
aws_instance.example: Still destroying... [id=i-0a76
aws_instance.example: Destruction complete after 1m8

Destroy complete! Resources: 1 destroyed.
(base) → terraform-cli-variables _
```

Confirm the destruction by typing yes.

6. Conclusion:

This lab exercise demonstrates how to use command
values dynamically during the terraform apply proce
your Terraform deployments without modifying th
Experiment with different variable values and observe
impact the infrastructure provisioning process.