KODEKLOUD

Certification Details

60 Minutes

70.50 USD

Valid for 2 years

Multiple Choice Questions

Multiple Options

True/False

~57 questions

Online proctored

No VM’s

PSI Secure Browser

No Additional Monitors / Headphones

Webcam, Speakers and Microphone ON

Quiet, well lit and clean room

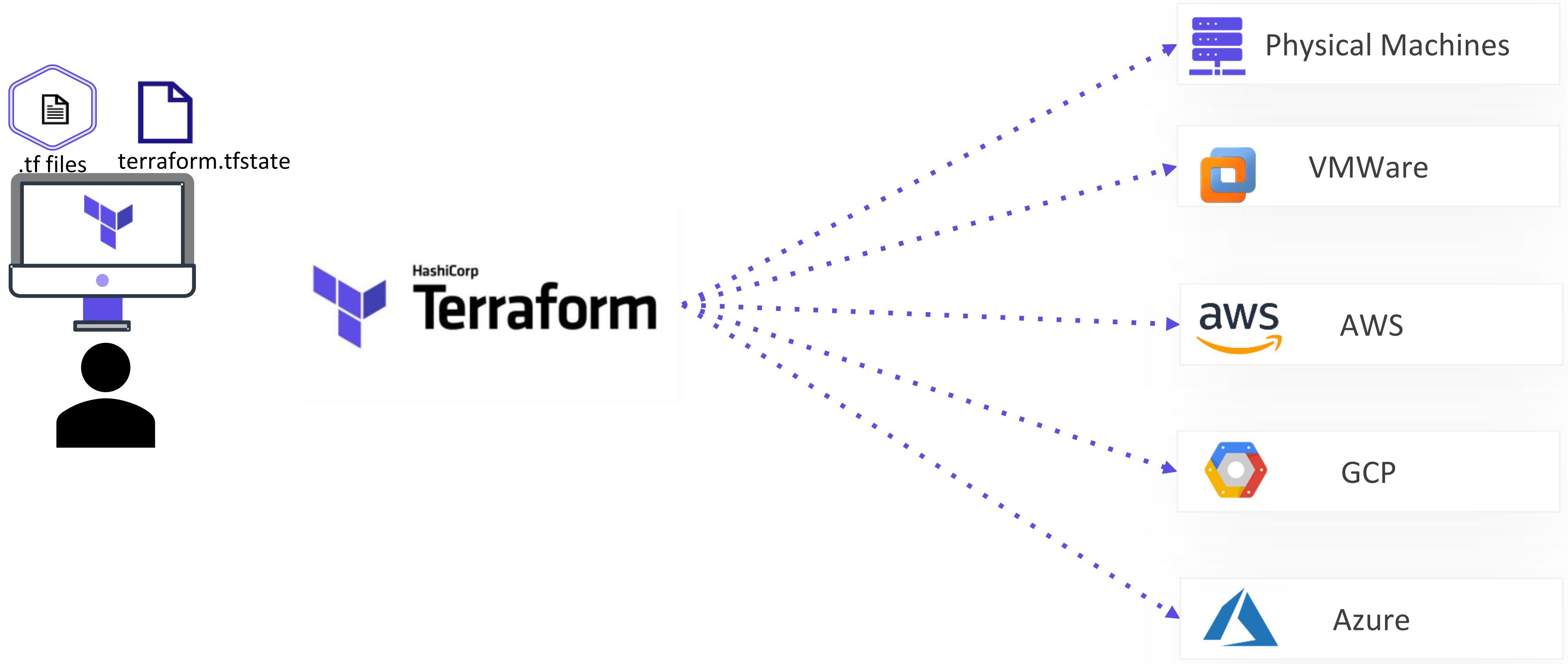
Register

https://www.hashicorp.com/certification/terraform-associate

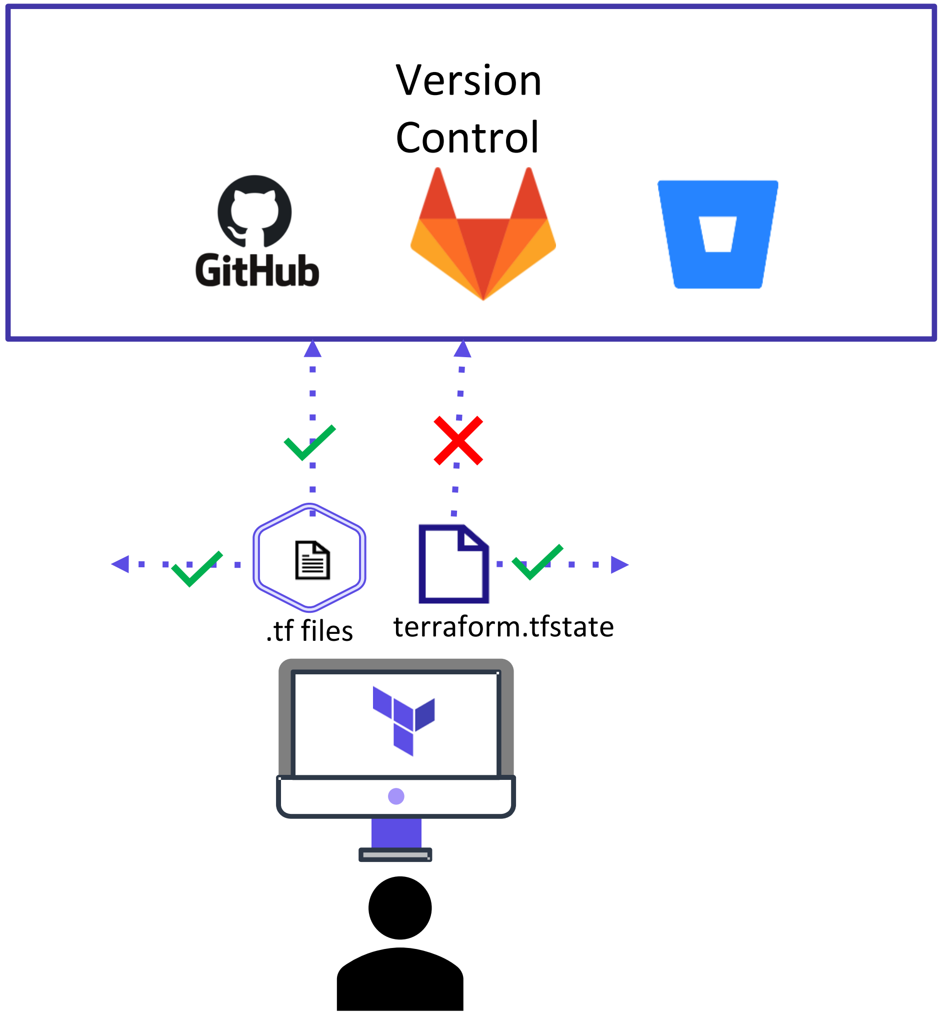
KODEKLOUD

Introduction to Terraform Cloud

Terraform Cloud

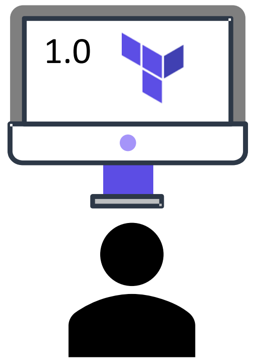


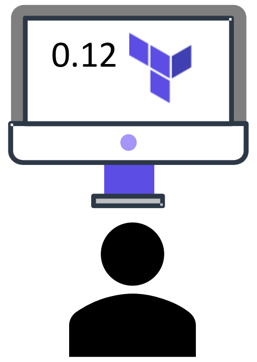
Terraform Cloud



Terraform Cloud

HashiCorp Terraform Cloud





Terraform Cloud

HashiCorp Terraform Cloud

Shared State

Consistent and Reliable Environment

UI Interface

Secret Management

Access Controls

Private Registry

Policy Controls

KODEKLOUD

Terraform Cloud Plans

https://www.hashicorp.com/products/terraform/pricing

Terraform Cloud

HashiCorp Terraform Cloud

FREE PLAN	TEAM PLAN	TEAM and GOVERNANCE
Remote State	Team Management	Team Management
Remote Operations		Policy as Code (Sentinel)
Private Module Registry		Policy Enforcement
Community Support		Cloud SLA and Support

Terraform Cloud

HashiCorp Terraform Cloud

BUSINESS

SSO

Custom Concurrency

Self-hosted options

Premium Support

okta

KODEKLOUD

Recap - Infrastructure as Code

Types of IAC Tools

docker

ANSIBLE

HashiCorp
Terraform

CloudFormation

HashiCorp
Vagrant

HashiCorp
Packer

SALTSTACK

puppet

Types of IAC Tools

Configuration Management

Server Templating

Provisioning Tools

Types of IAC Tools

Configuration Management

ANSIBLE

puppet

SALTSTACK

Designed to Install and Manage Software

Maintains Standard Structure

Version Control

Idempotent

Server Templating Tools

Pre Installed Software and Dependencies

Virtual Machine or Docker Images

Immutable Infrastructure

docker

HashiCorp Packer

HashiCorp Vagrant

Provisioning Tools

Deploy Immutable Infrastructure resources

Servers, Databases, Network Components etc.

Multiple Providers

HashiCorp Terraform

CloudFormation

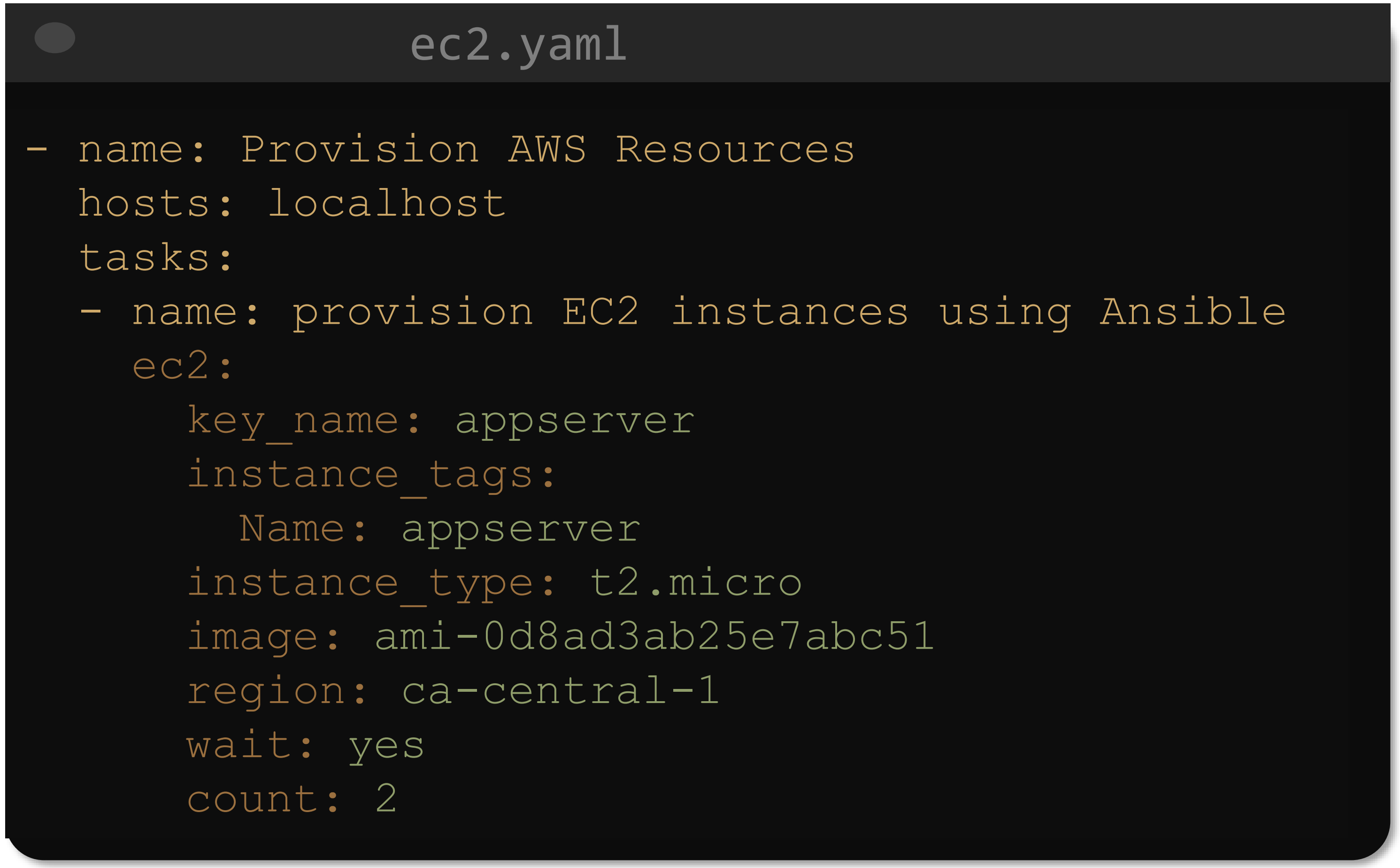
Which IaC Tools Should I Use?

Configuration Management

ANSIBLE

puppet

SALTSTACK



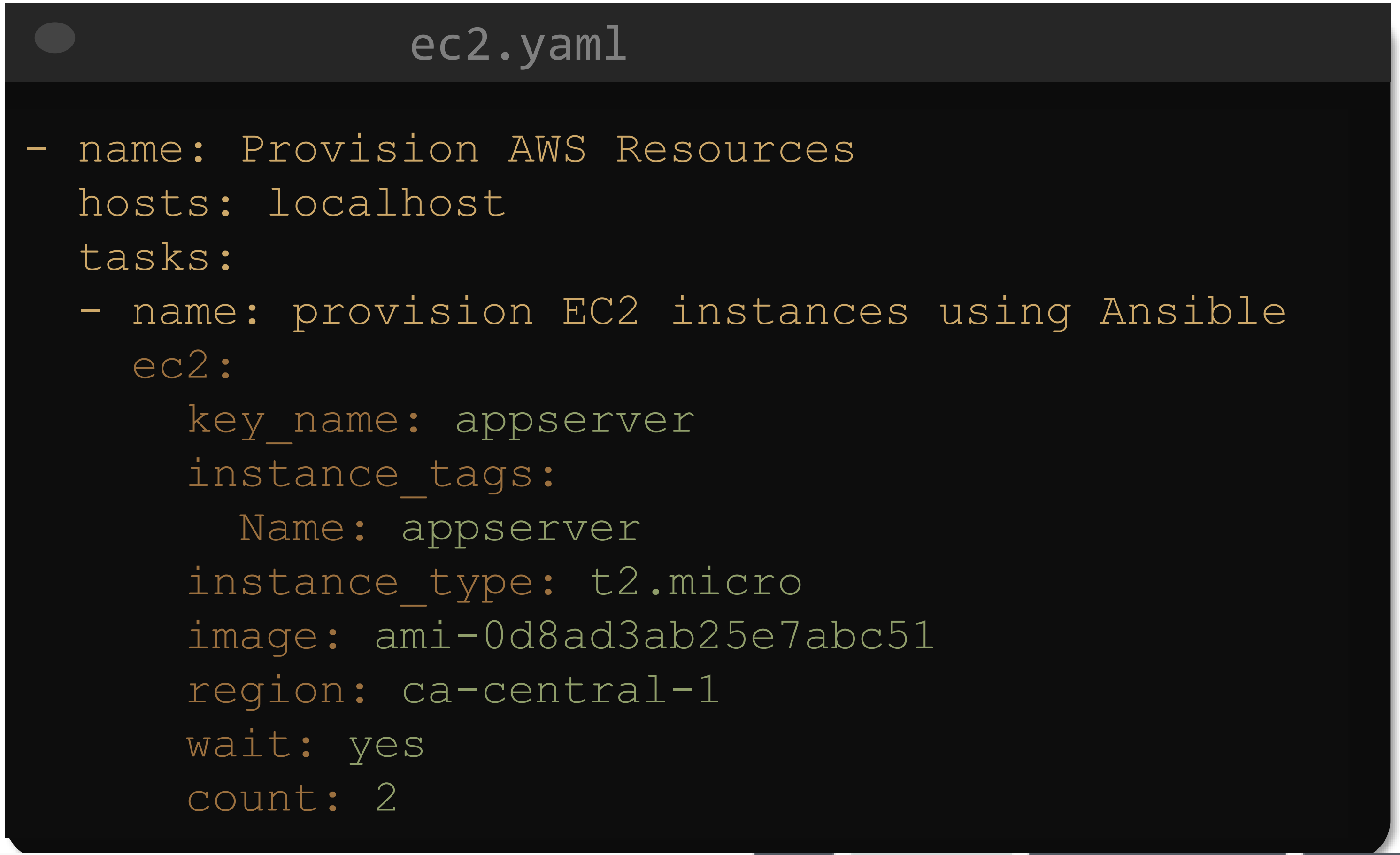
ec2.yaml

```yaml
- name: Provision AWS Resources
  hosts: localhost
  tasks:
  - name: provision EC2 instances using Ansible
    ec2:
      key_name: appserver
      instance_tags:
        Name: appserver
      instance_type: t2.micro
      image: ami-0d8ad3ab25e7abc51
      region: ca-central-1
      wait: yes
      count: 2
```

Which IaC Tools Should I Use?

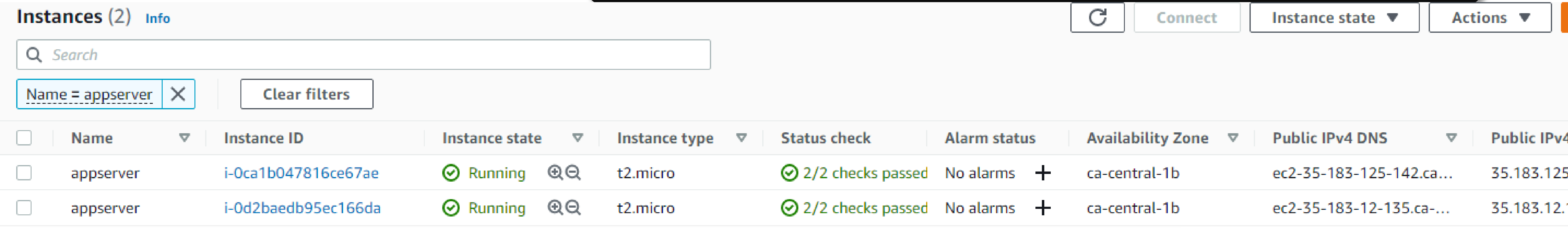
Configuration Management

ANSIBLE



ec2.yaml

```yaml
- name: Provision AWS Resources
  hosts: localhost
  tasks:
  - name: provision EC2 instances using Ansible
    ec2:
      key_name: appserver
      instance_tags:
        Name: appserver
      instance_type: t2.micro
      image: ami-0d8ad3ab25e7abc51
      region: ca-central-1
      wait: yes
      count: 2
```



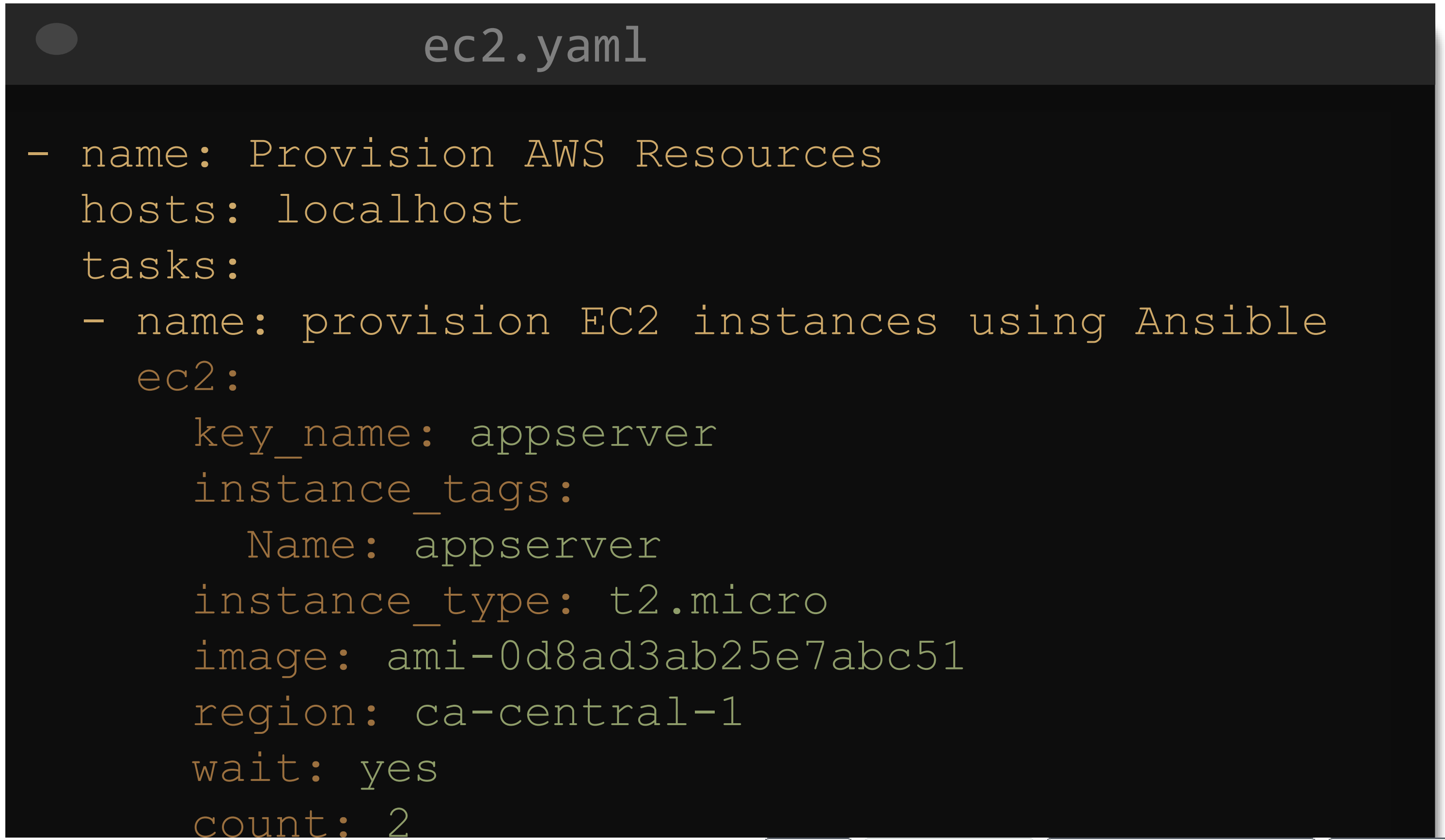
Instances (2) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4
appserver	i-0ca1b047816ce67ae	Running	t2.micro	2/2 checks passed	No alarms	ca-central-1b	ec2-35-183-125-142.ca...	35.183.125
appserver	i-0d2baedb95ec166da	Running	t2.micro	2/2 checks passed	No alarms	ca-central-1b	ec2-35-183-12-135.ca-...	35.183.12.

Which IaC Tools Should I Use?

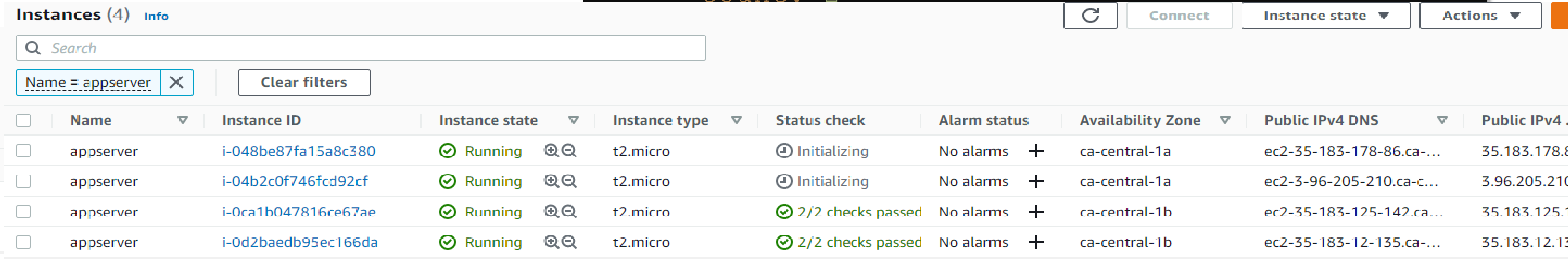
Configuration Management

ANSIBLE



ec2.yaml

```yaml
- name: Provision AWS Resources
  hosts: localhost
  tasks:
  - name: provision EC2 instances using Ansible
    ec2:
      key_name: appserver
      instance_tags:
        Name: appserver
      instance_type: t2.micro
      image: ami-0d8ad3ab25e7abc51
      region: ca-central-1
      wait: yes
      count: 2
```



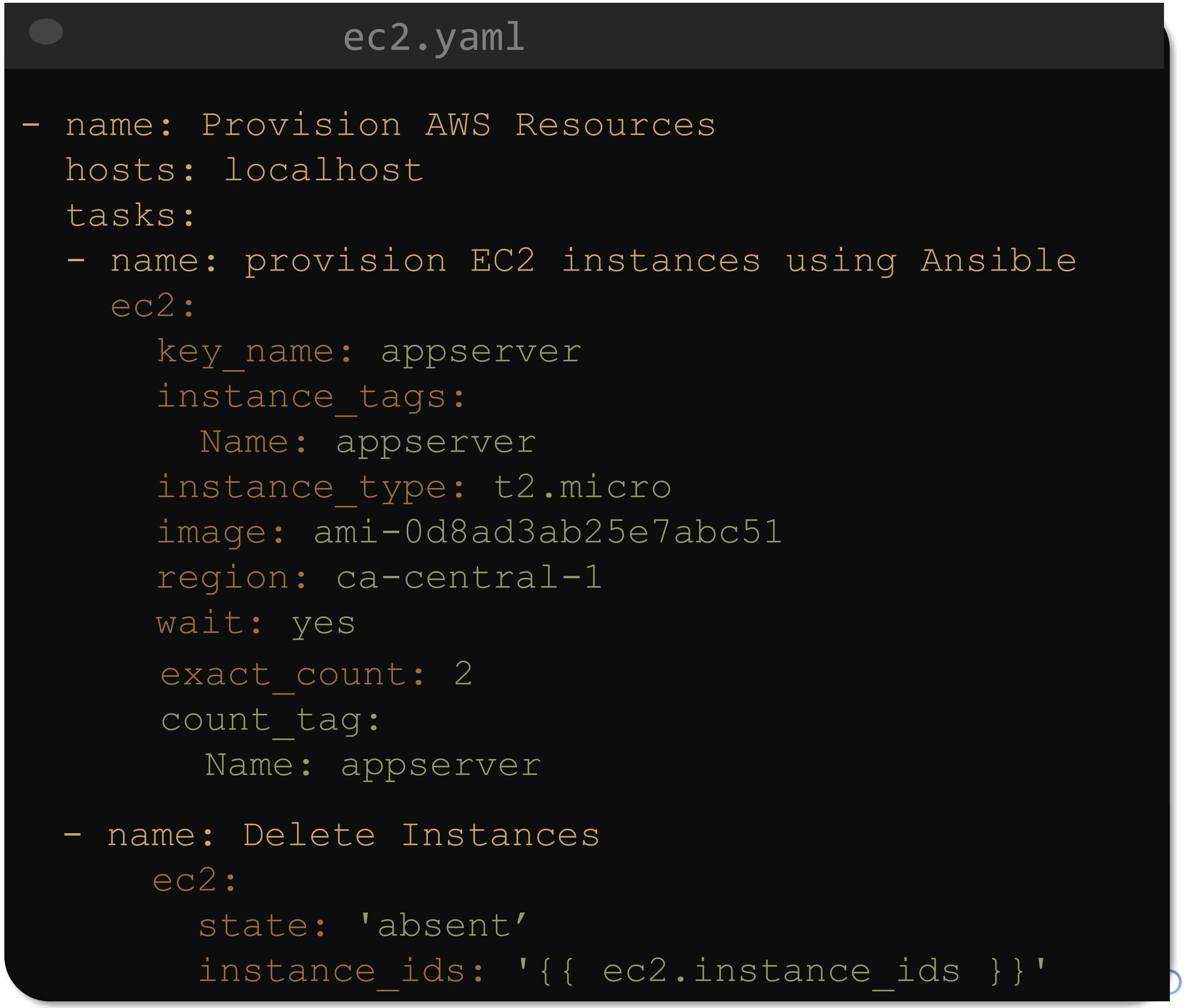
Instances (4) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4
appserver	i-048be87fa15a8c380	Running	t2.micro	Initializing	No alarms	ca-central-1a	ec2-35-183-178-86.ca-...	35.183.178.8
appserver	i-04b2c0f746fcd92cf	Running	t2.micro	Initializing	No alarms	ca-central-1a	ec2-3-96-205-210.ca-c...	3.96.205.210
appserver	i-0ca1b047816ce67ae	Running	t2.micro	2/2 checks passed	No alarms	ca-central-1b	ec2-35-183-125-142.ca...	35.183.125.
appserver	i-0d2baedb95ec166da	Running	t2.micro	2/2 checks passed	No alarms	ca-central-1b	ec2-35-183-12-135.ca-...	35.183.12.13

Which IaC Tools Should I Use?

Configuration Management

ANSIBLE



ec2.yaml

```yaml
- name: Provision AWS Resources
  hosts: localhost
  tasks:
  - name: provision EC2 instances using Ansible
    ec2:
      key_name: appserver
      instance_tags:
        Name: appserver
      instance_type: t2.micro
      image: ami-0d8ad3ab25e7abc51
      region: ca-central-1
      wait: yes
      exact_count: 2
      count_tag:
        Name: appserver

  - name: Delete Instances
    ec2:
      state: 'absent'
      instance_ids: '{{ ec2.instance_ids }}'
```

Which IaC Tools Should I Use?

Provisioning Tools

HashiCorp Terraform

ec2.tf

```
resource "aws_instance" "app" {
   ami           = "ami-0d8ad3ab25e7abc51"
   instance_type = "t2.micro"
   count         = 2
   key_name      = "appserver"
   tags = {
      Name = "appserver"
   }
}
```

```
> terraform apply
.
.
No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and
found no differences, so no changes are needed.

Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
```

terraform.tfstate

Which IaC Tools Should I Use?

Provisioning Tools

HashiCorp Terraform

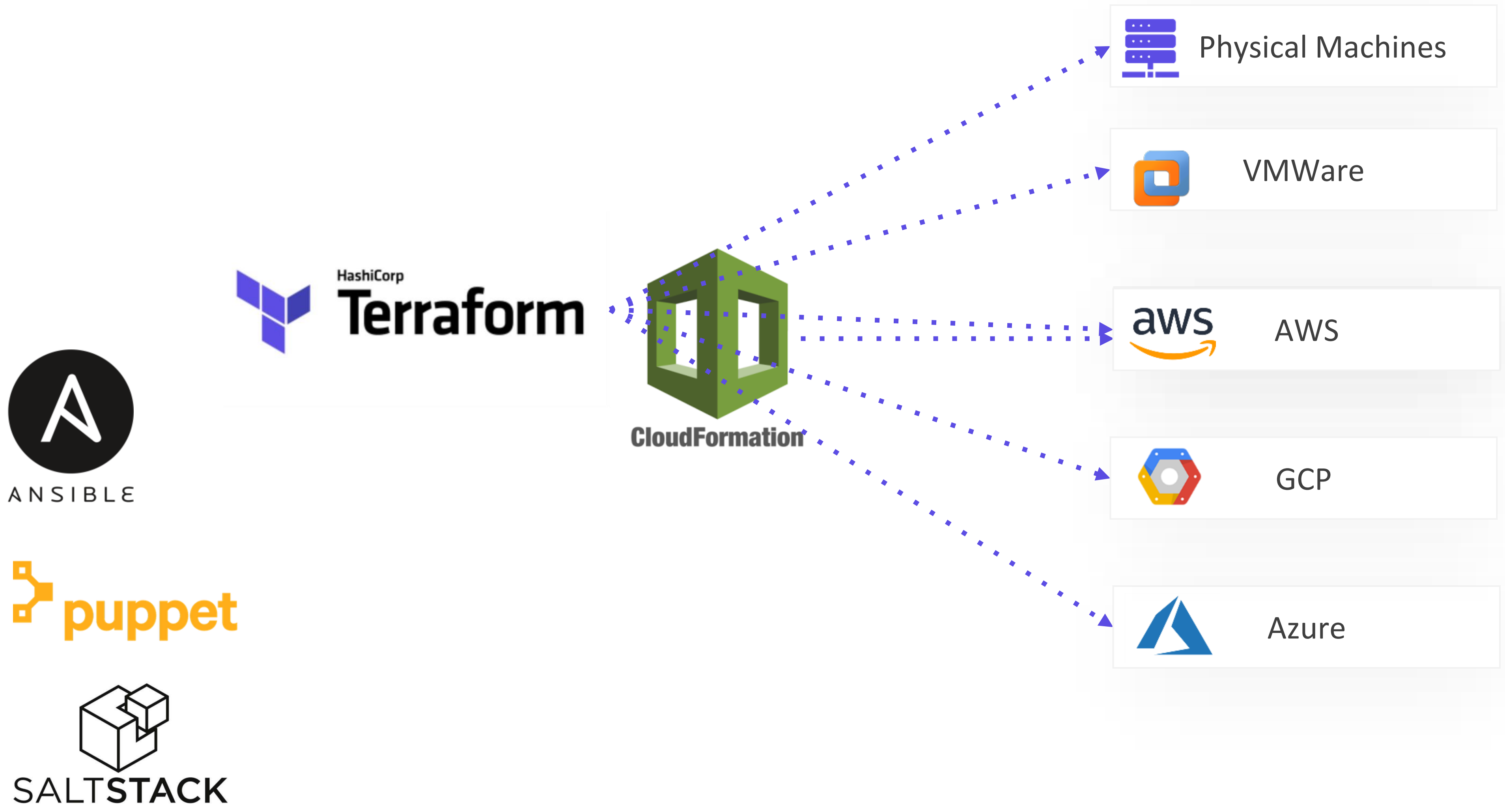
ec2.tf

```
resource "aws_instance" "app" {
  ami           = "ami-0d8ad3ab25e7abc51"
  instance_type = "t2.micro"
  count         = 2
  key_name      = "appserver"
  tags = {
    Name = "appserver"
  }
}
```

```
> terraform destroy
.
.
aws_instance.app[1]: Destroying... [id=i-0fc7d85da32d24c63]
aws_instance.app[0]: Destroying... [id=i-014c93c14e12a6442]
.

aws_instance.app[1]: Destruction complete after 50s

Destroy complete! Resources: 2 destroyed.
```

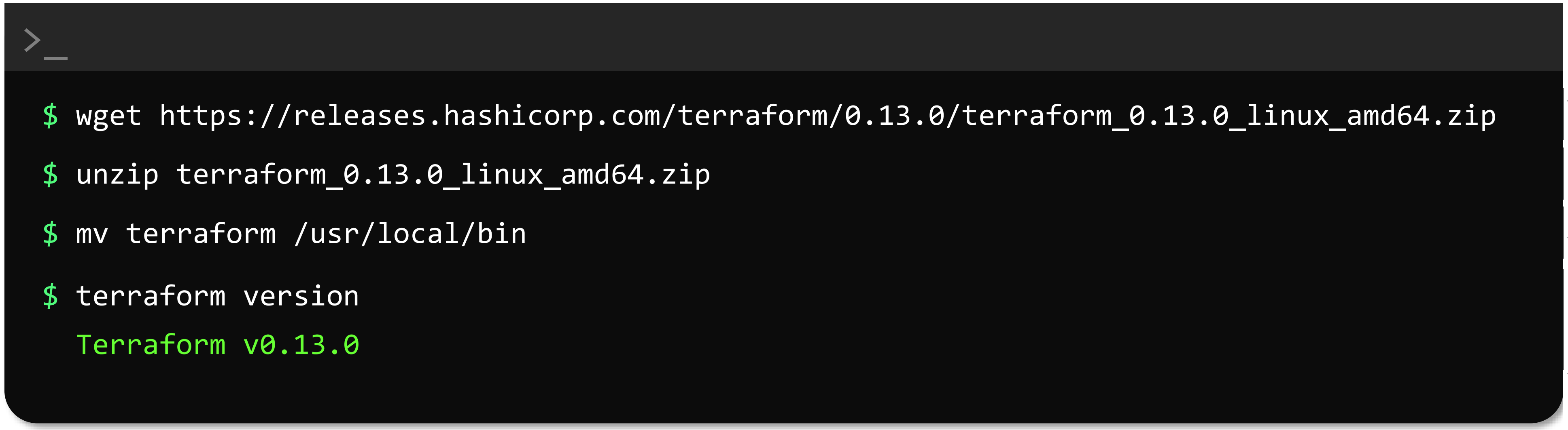
terraform.tfstate

Which IaC Tools Should I Use?



KODEKLOUD

Installing Terraform & HCL Basics



```
$ wget https://releases.hashicorp.com/terraform/0.13.0/terraform_0.13.0_linux_amd64.zip
$ unzip terraform_0.13.0_linux_amd64.zip
$ mv terraform /usr/local/bin
$ terraform version
  Terraform v0.13.0
```

macOS
64-bit

FreeBSD
32-bit | 64-bit | Arm

Linux
32-bit | 64-bit | Arm

OpenBSD
32-bit | 64-bit

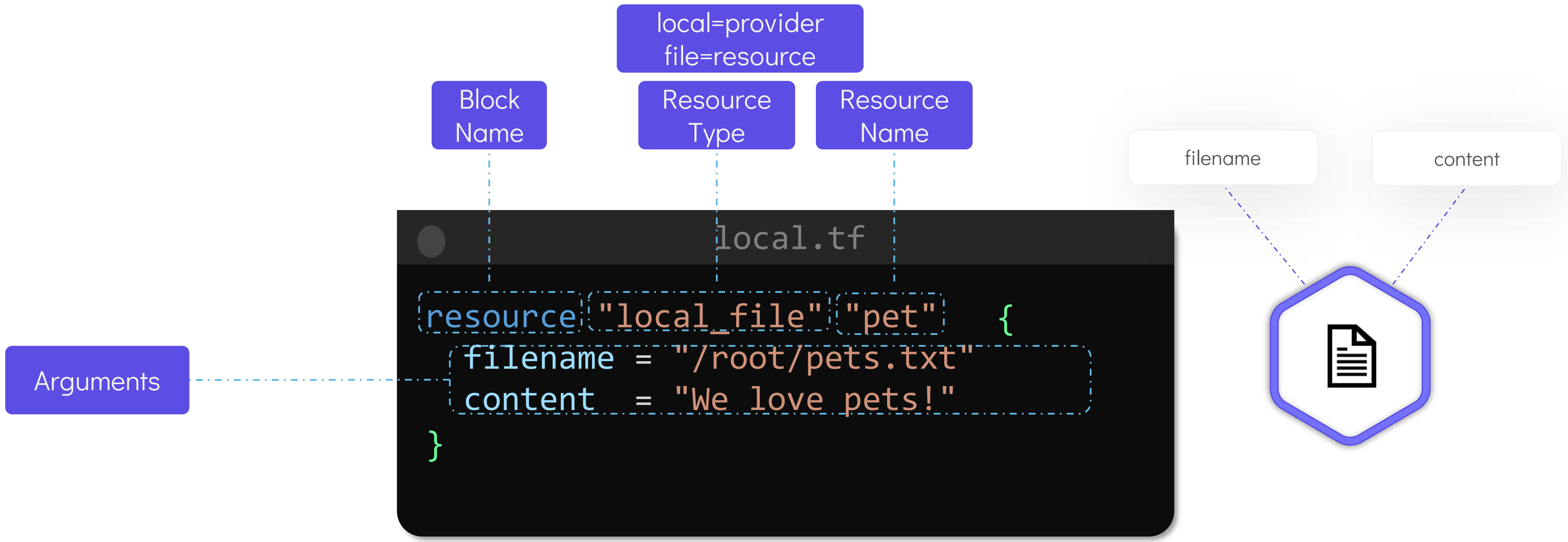
Solaris
64-bit

Windows
32-bit | 64-bit

HCL – Declarative Language

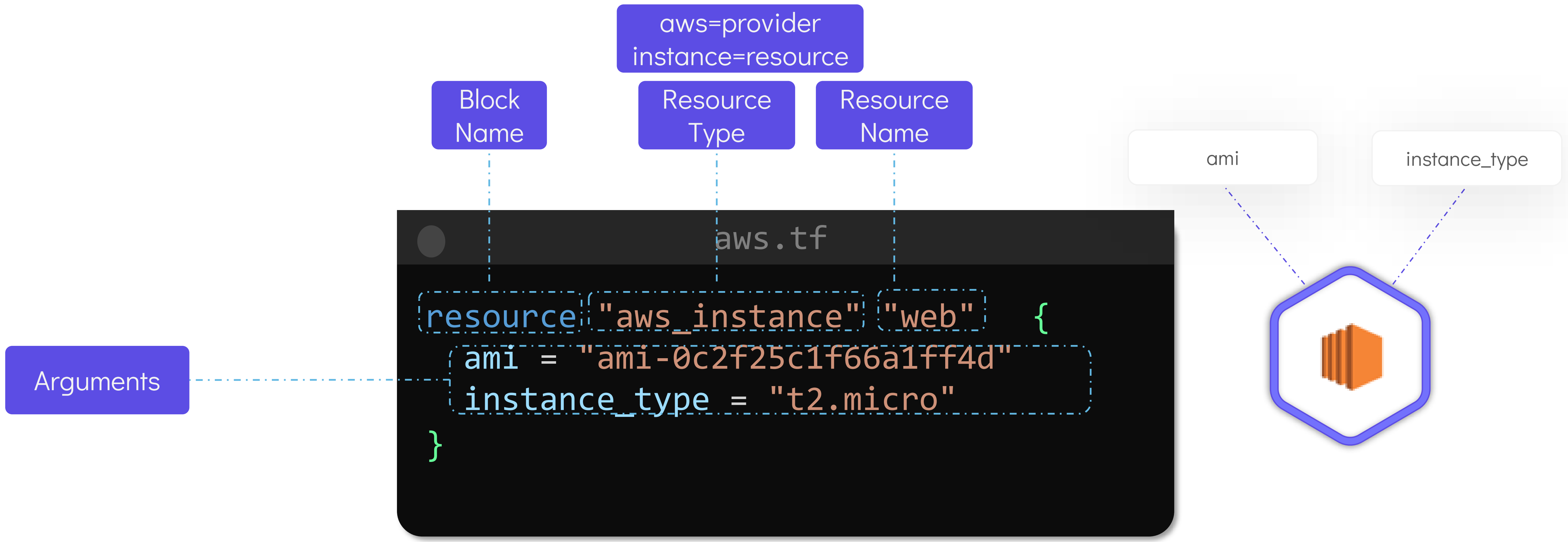
local.tf

```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
}
```



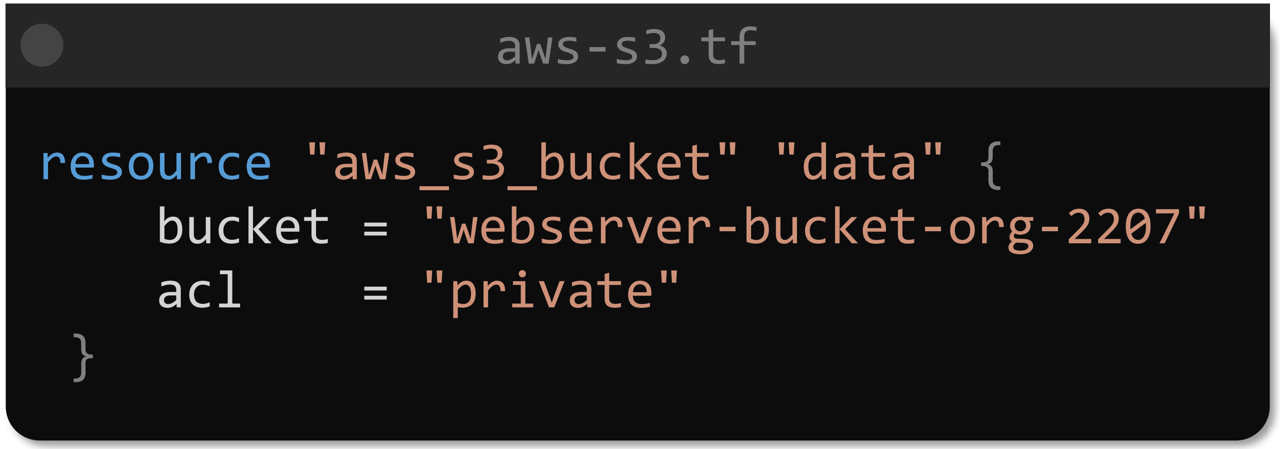
```
resource "local_file" "pet"   {
  filename = "/root/pets.txt"
  content  = "We love pets!"
}
```

KODEKLOUD



aws.tf

```
resource "aws_instance" "web"   {
  ami = "ami-0c2f25c1f66a1ff4d"
  instance_type = "t2.micro"
}
```



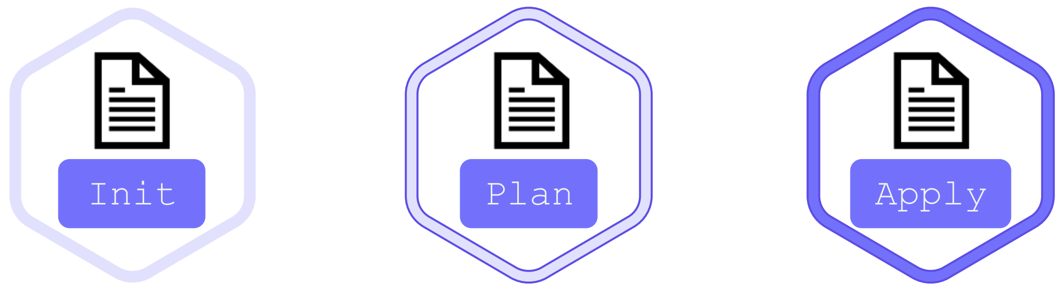
aws-s3.tf

```
resource "aws_s3_bucket" "data" {
    bucket = "webserver-bucket-org-2207"
    acl    = "private"
 }
```

Resource

local.tf

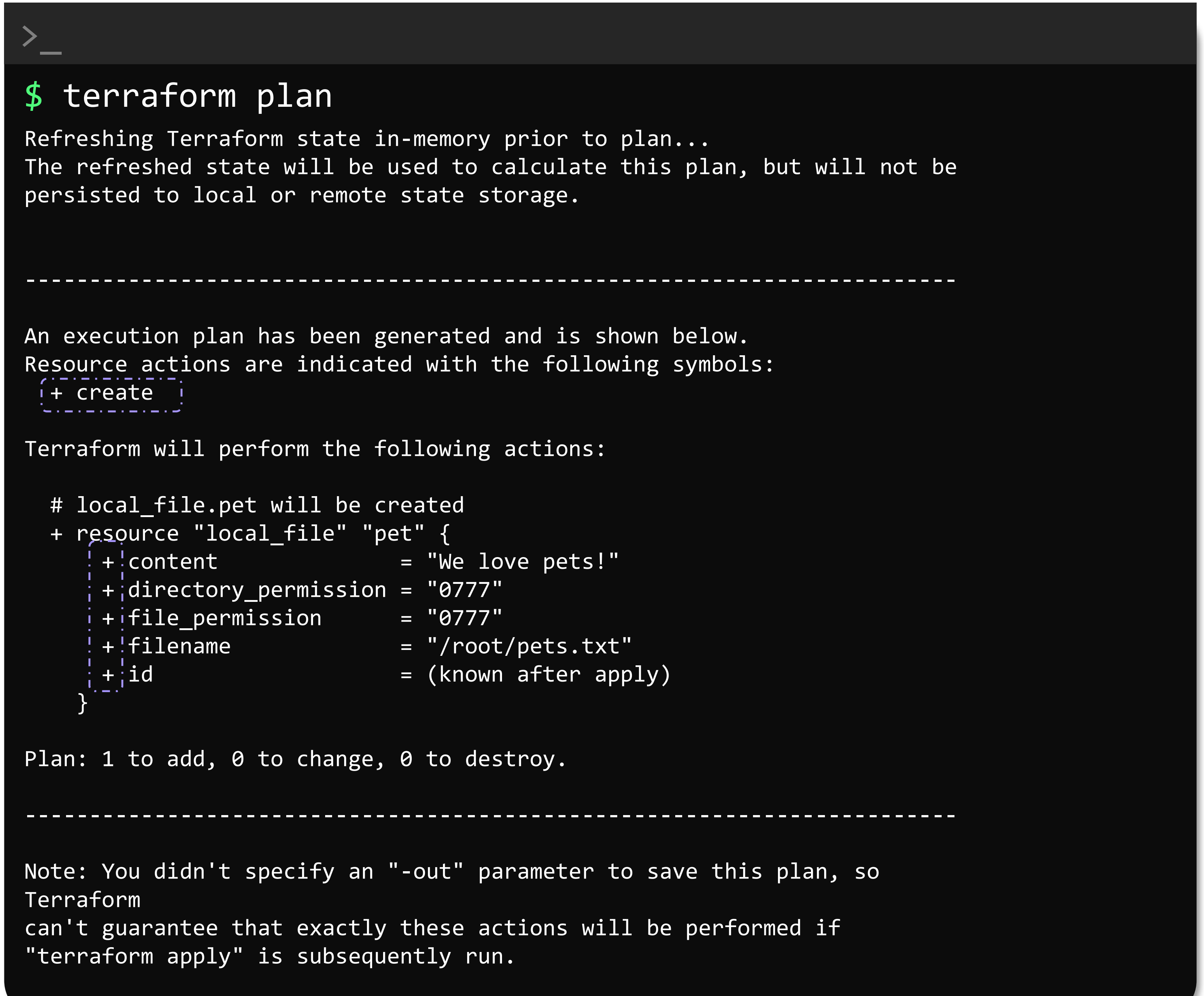
```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
}
```



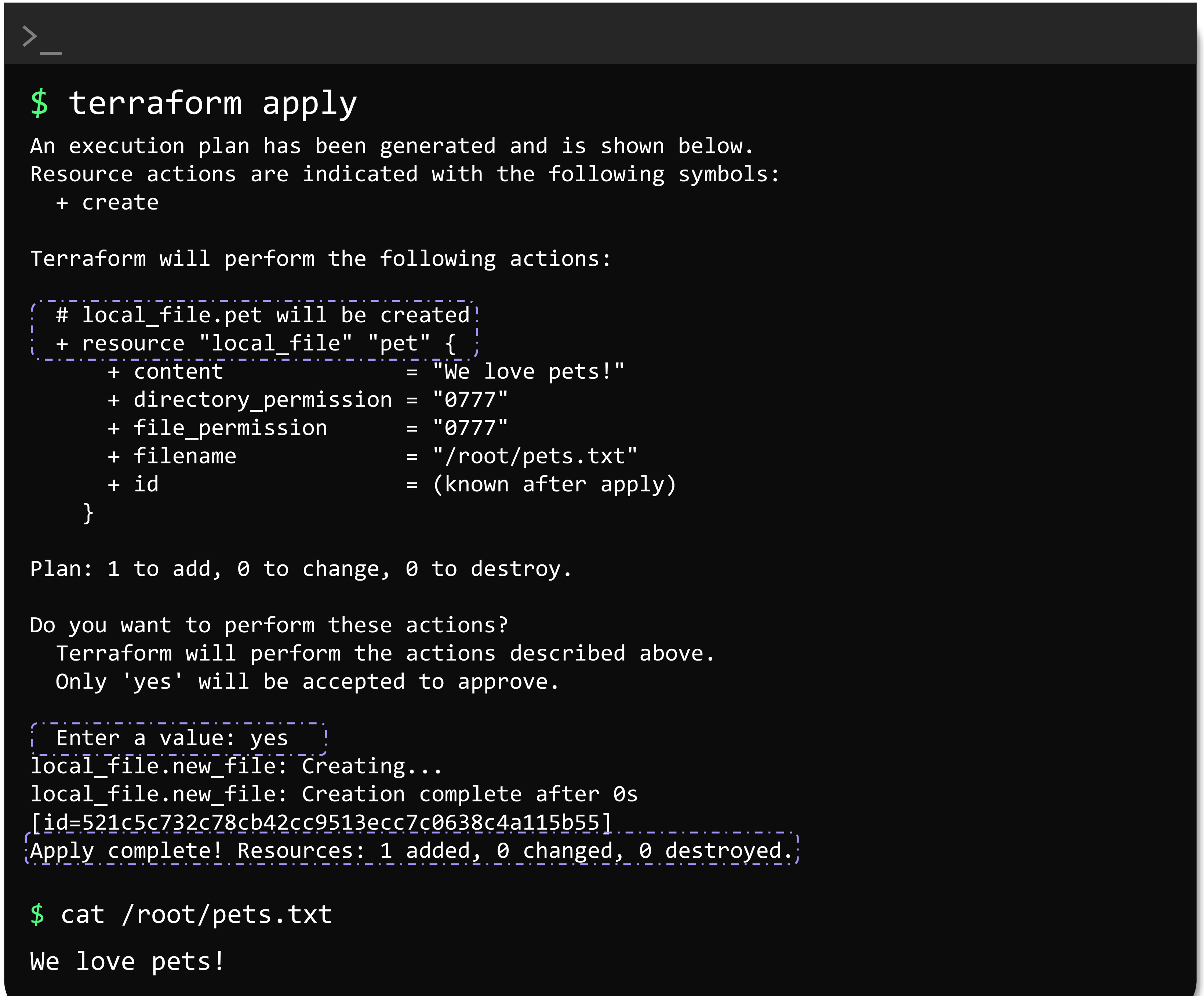
local.tf

```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
}
```

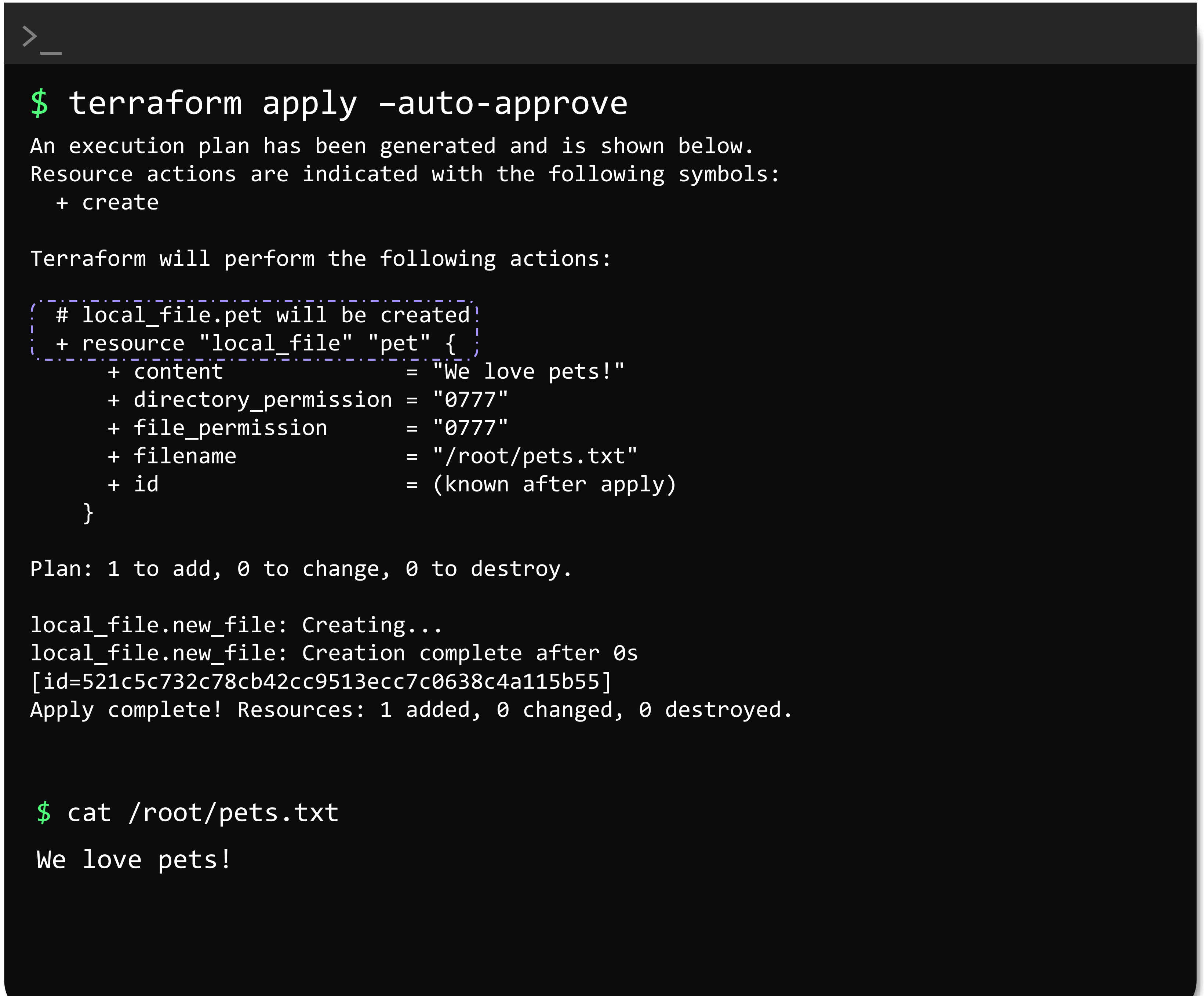
```
$ terraform init
Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/local...
- Installing hashicorp/local v1.4.0...
- Installed hashicorp/local v1.4.0 (signed by HashiCorp)

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/local: version = "~> 1.4.0"

Terraform has been successfully initialized!
```



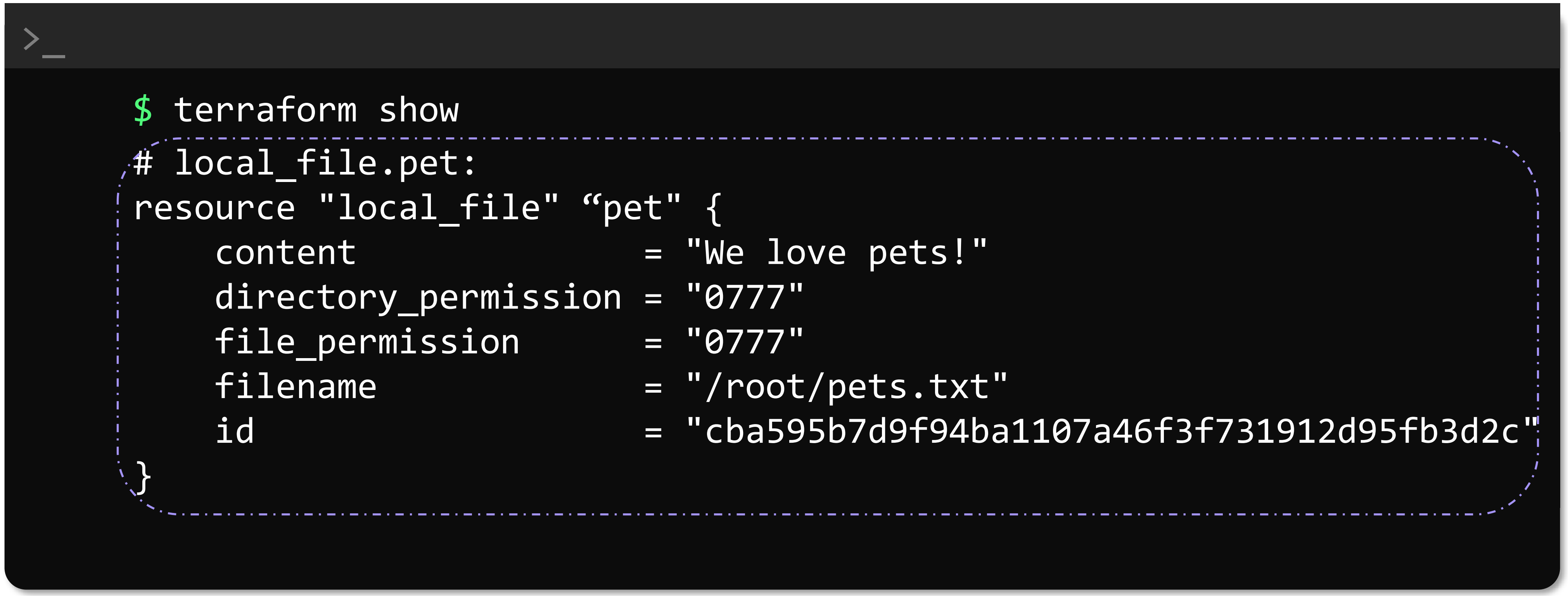
```
$ terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.


------------------------------------------------------------------------

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

Terraform will perform the following actions:

  # local_file.pet will be created
  + resource "local_file" "pet" {
      + content              = "We love pets!"
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/pets.txt"
      + id                   = (known after apply)
    }

Plan: 1 to add, 0 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.
```



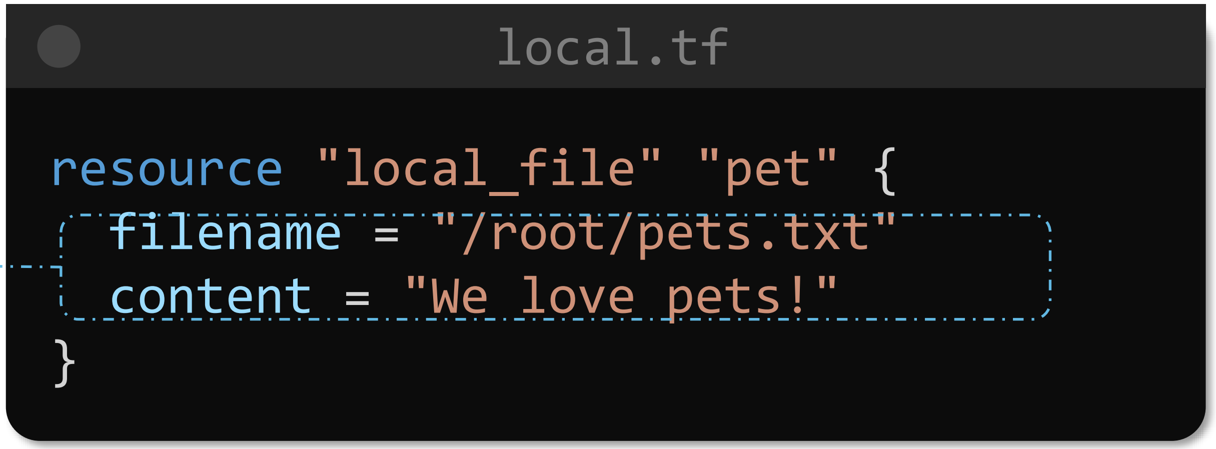
```
$ terraform apply
An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # local_file.pet will be created
  + resource "local_file" "pet" {
      + content              = "We love pets!"
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/pets.txt"
      + id                   = (known after apply)
    }

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

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes
local_file.new_file: Creating...
local_file.new_file: Creation complete after 0s
[id=521c5c732c78cb42cc9513ecc7c0638c4a115b55]
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

$ cat /root/pets.txt
We love pets!
```



```
$ terraform apply -auto-approve
An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # local_file.pet will be created
  + resource "local_file" "pet" {
      + content              = "We love pets!"
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/pets.txt"
      + id                   = (known after apply)
    }

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

local_file.new_file: Creating...
local_file.new_file: Creation complete after 0s
[id=521c5c732c78cb42cc9513ecc7c0638c4a115b55]
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.


$ cat /root/pets.txt
We love pets!
```

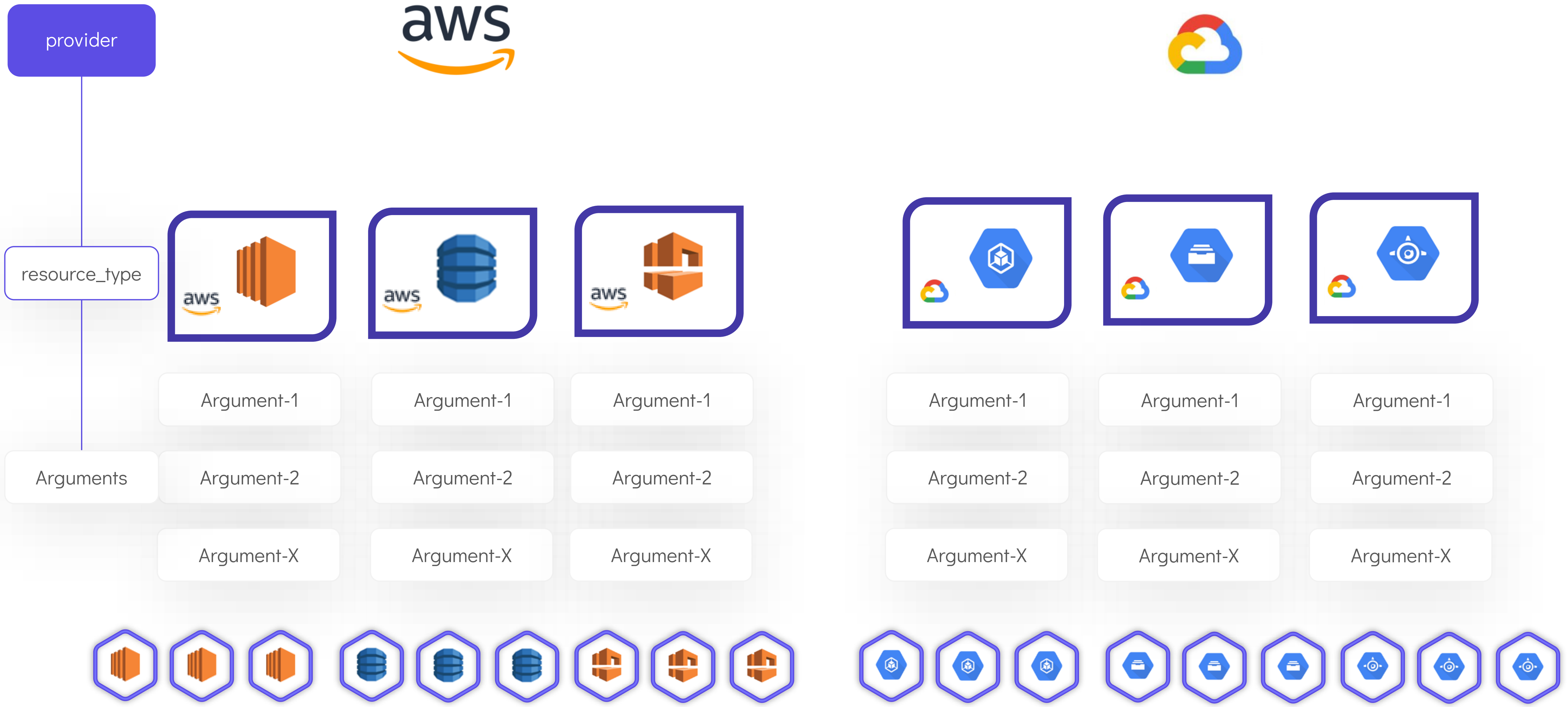


```
$ terraform show
# local_file.pet:
resource "local_file" “pet" {
    content              = "We love pets!"
    directory_permission = "0777"
    file_permission      = "0777"
    filename             = "/root/pets.txt"
    id                   = "cba595b7d9f94ba1107a46f3f731912d95fb3d2c"
}
```

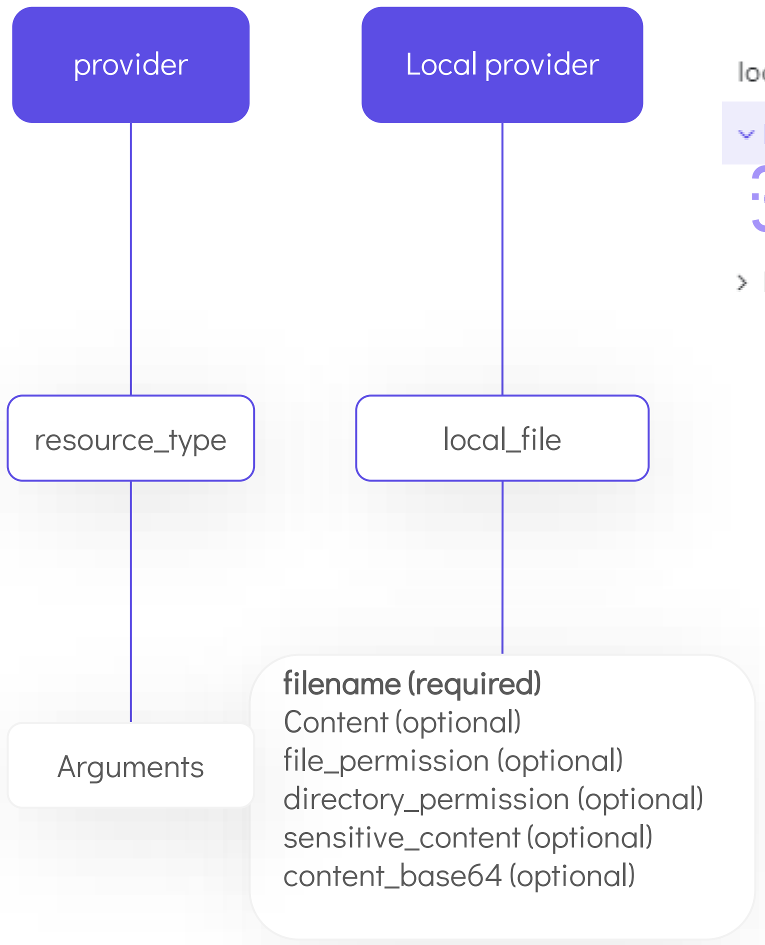


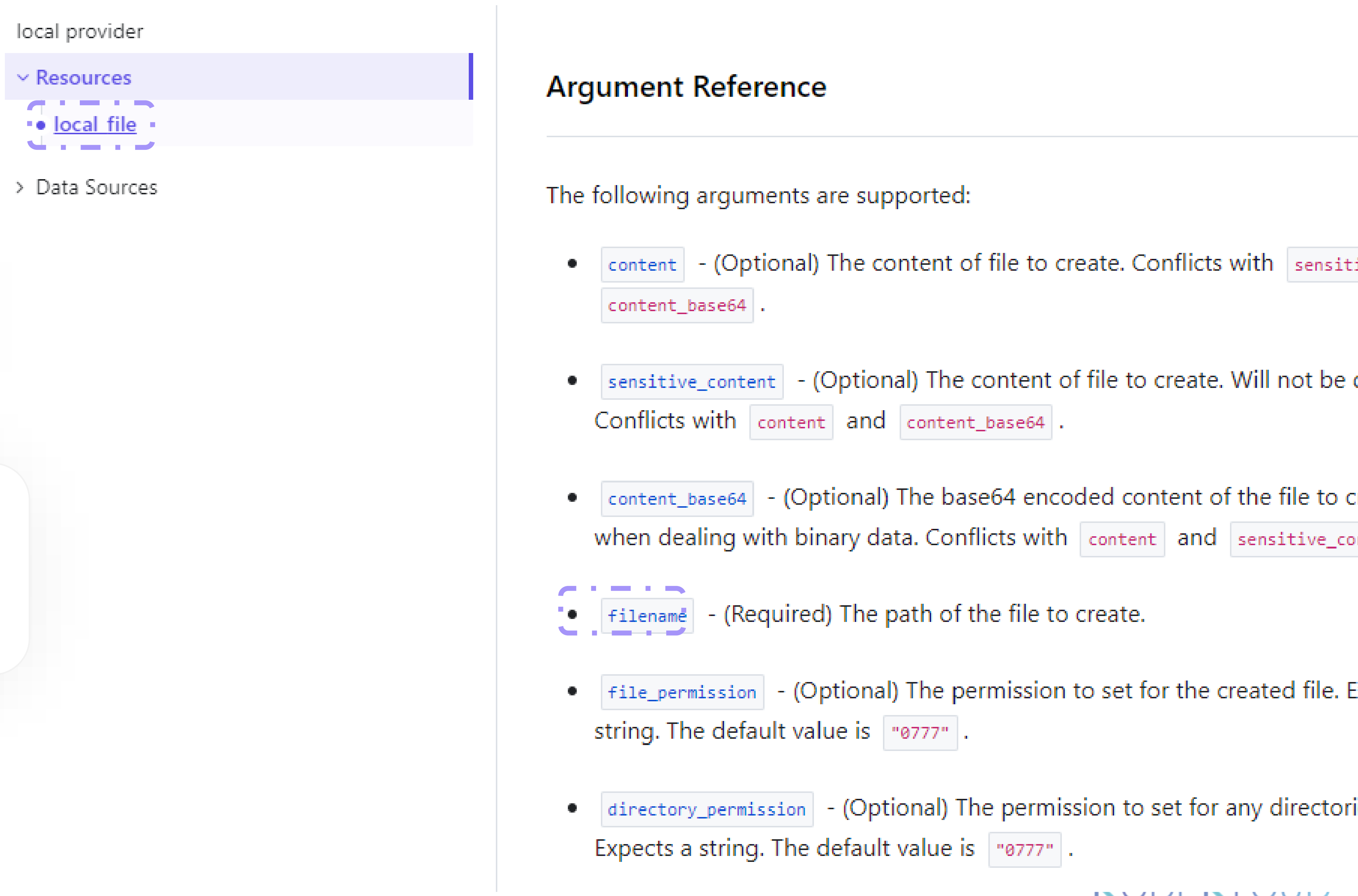
local.tf

```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
}
```



https://registry.terraform.io/providers/hashicorp/local/latest/docs


provider
Local provider
resource_type
local_file
Arguments
filename (required)
Content (optional)
file_permission (optional)
directory_permission (optional)
sensitive_content (optional)
content_base64 (optional)



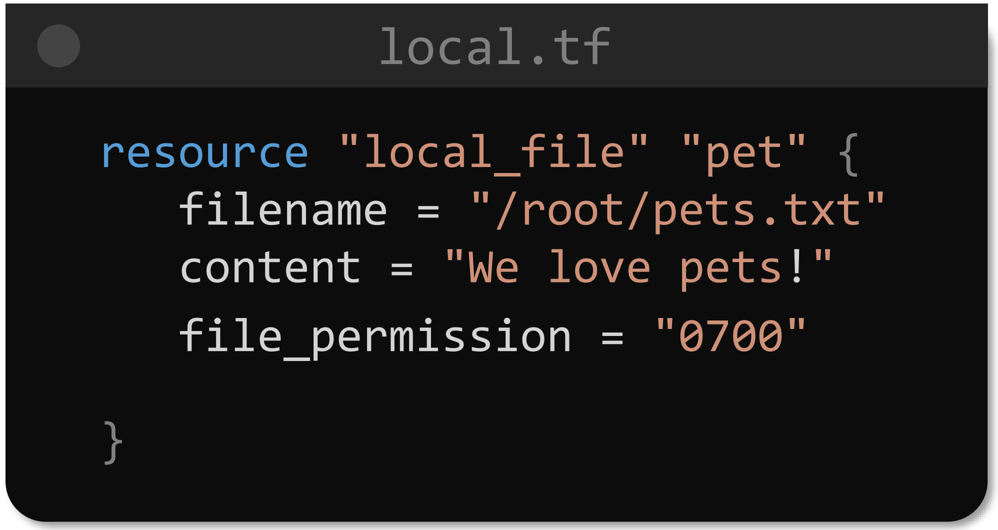
Argument Reference

The following arguments are supported:

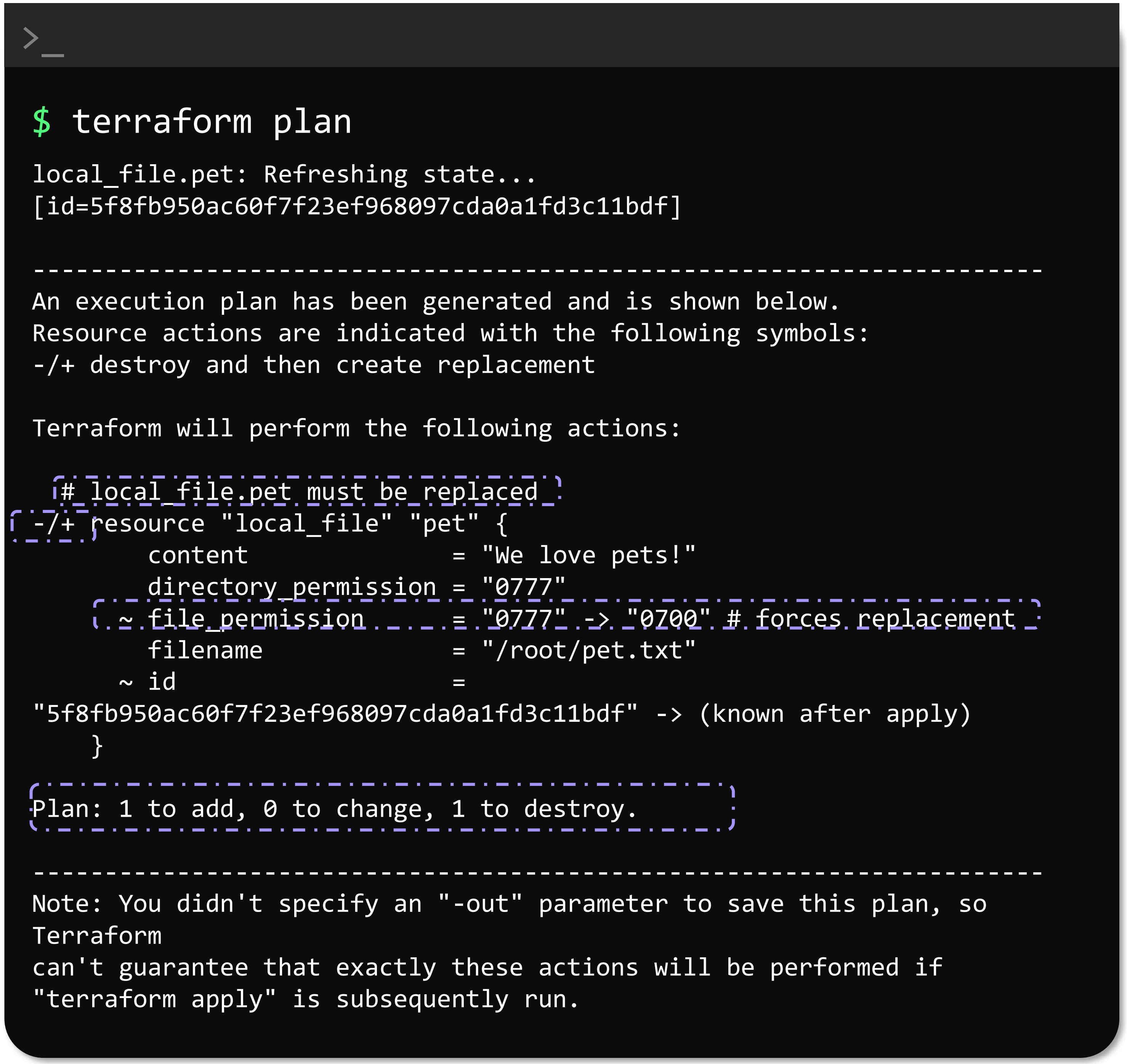
- `content` - (Optional) The content of file to create. Conflicts with `sensit` `content_base64` .
- `sensitive_content` - (Optional) The content of file to create. Will not be c Conflicts with `content` and `content_base64` .
- `content_base64` - (Optional) The base64 encoded content of the file to c when dealing with binary data. Conflicts with `content` and `sensitive_co`
- `filename` - (Required) The path of the file to create.
- `file_permission` - (Optional) The permission to set for the created file. E string. The default value is `"0777"` .
- `directory_permission` - (Optional) The permission to set for any directori Expects a string. The default value is `"0777"` .

KODEKLOUD

Update and Destroy Infrastructure


local.tf

```
resource "local_file" "pet" {
   filename = "/root/pets.txt"
   content = "We love pets!"
   file_permission = "0700"

}
```

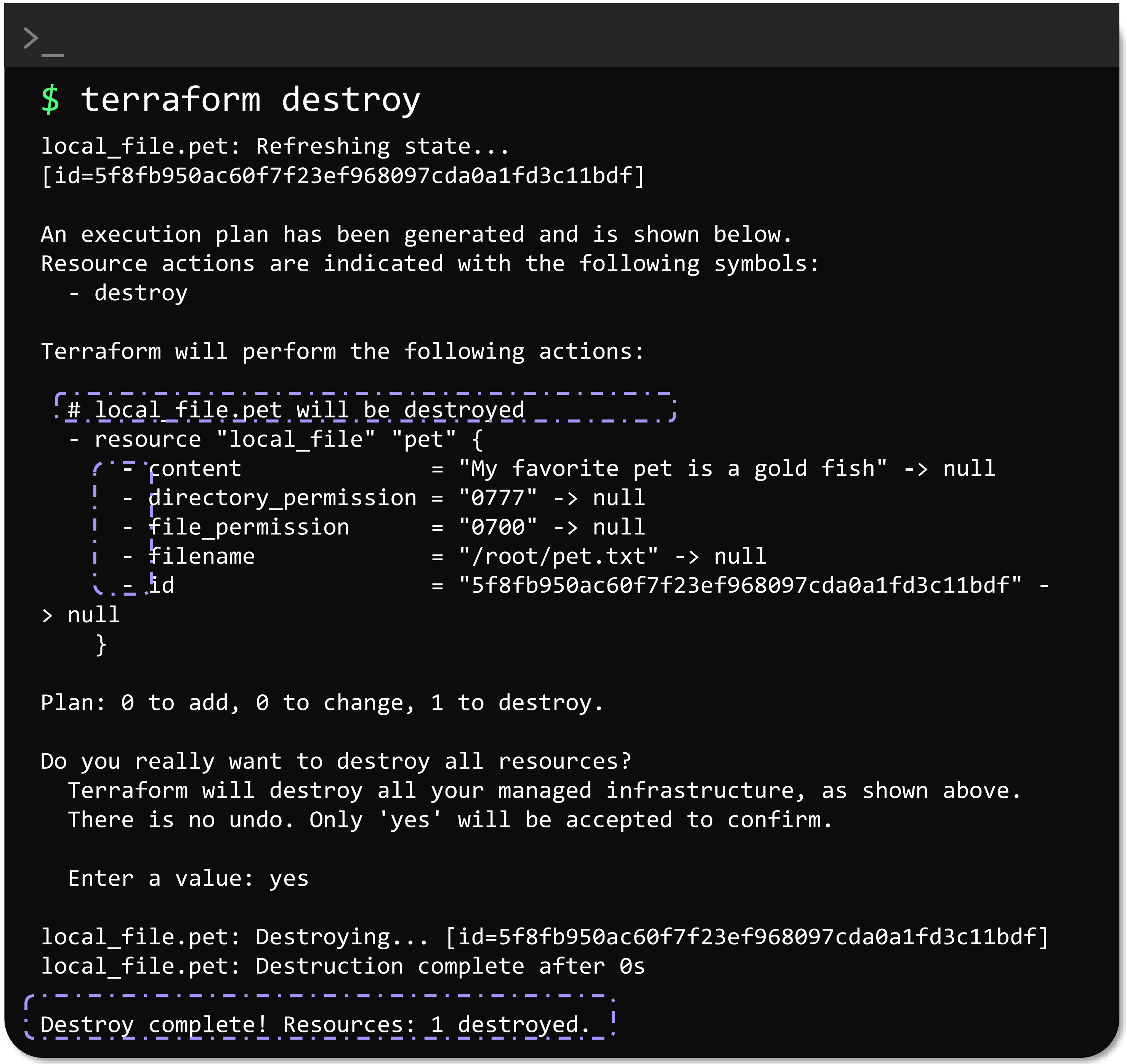


```
$ terraform plan
local_file.pet: Refreshing state...
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]

----------------------------------------------------------------------
An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # local_file.pet must be replaced
-/+ resource "local_file" "pet" {
        content              = "We love pets!"
        directory_permission = "0777"
      ~ file_permission      = "0777" -> "0700" # forces replacement
        filename             = "/root/pet.txt"
      ~ id                   =
"5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -> (known after apply)
    }

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

local.tf

```
resource "local_file" "pet" {
   filename = "/root/pets.txt"
   content = "We love pets!"
   file_permission = "0700"

}
```

```
$ terraform apply

 local_file.pet: Refreshing state...
 [id=fefacccdae259f25533749abfb90e27558256459]

 -/+ destroy and then create replacement

 .
 .

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

 Do you want to perform these actions?
   Terraform will perform the actions described above.
   Only 'yes' will be accepted to approve.

   Enter a value: yes

 local_file.pet: Destroying...
 [id=fefacccdae259f25533749abfb90e27558256459]
 local_file.pet: Destruction complete after 0s
 local_file.pet: Creating...
 local_file.pet: Creation complete after 0s
 [id=fefacccdae259f25533749abfb90e27558256459]

 Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```



```
$ terraform destroy
local_file.pet: Refreshing state...
[id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # local_file.pet will be destroyed
  - resource "local_file" "pet" {
      - content              = "My favorite pet is a gold fish" -> null
      - directory_permission = "0777" -> null
      - file_permission      = "0700" -> null
      - filename             = "/root/pet.txt" -> null
      - id                   = "5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf" -
> null
    }

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

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

local_file.pet: Destroying... [id=5f8fb950ac60f7f23ef968097cda0a1fd3c11bdf]
local_file.pet: Destruction complete after 0s

Destroy complete! Resources: 1 destroyed.
```

```
[terraform-local-file]$ ls /root/terraform-local-file
local.tf
```

local.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
}
```

cat.tf

```
resource "local_file"  "cat" {
    filename = "/root/cat.txt"
    content = "My favorite pet is Mr. Whiskers"

}
```

local.tf

```
resource "local_file" "pet" {
    filename = "/root/pets.txt"
    content = "We love pets!"
}
```

cat.tf

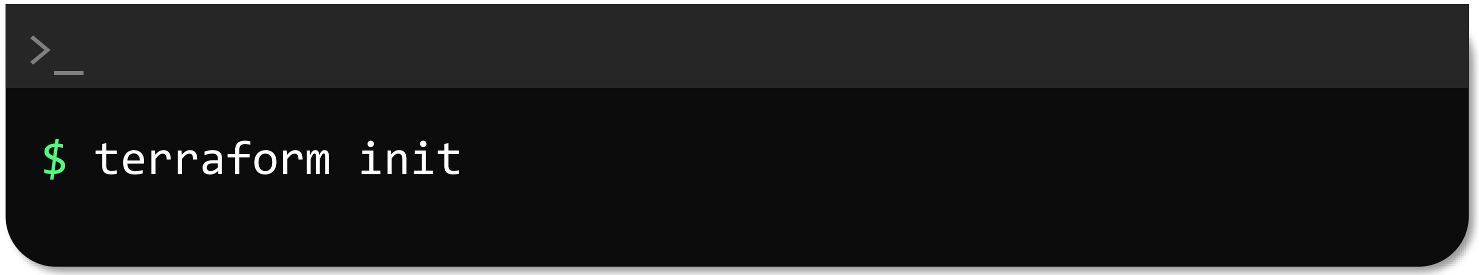
```
resource "local_file"  "cat" {
    filename = "/root/cat.txt"
    content = "My favorite pet is Mr. Whiskers"

}
```

main.tf

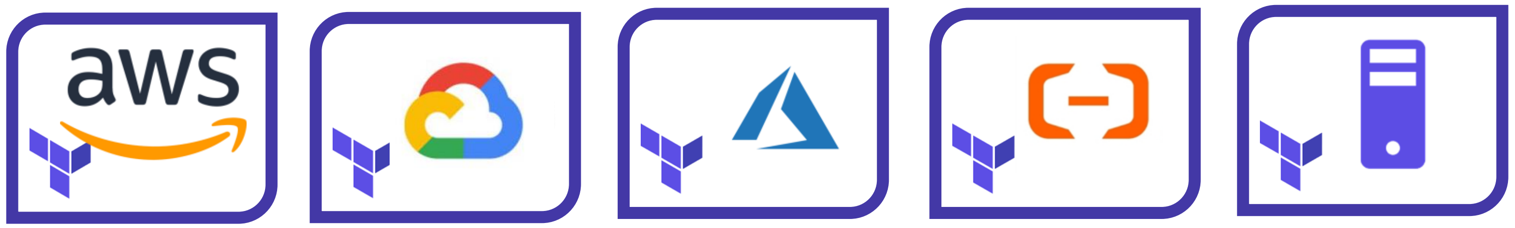
File Name	Purpose
main.tf	Main configuration file containing resource definition
variables.tf	Contains variable declarations
outputs.tf	Contains outputs from resources
provider.tf	Contains Provider definition
terraform.tf	Configure Terraform behaviour

KODEKLOUD

Recap Using Terraform Providers



```
$ terraform init
```



Official

Verified

bigip

by: F5Networks

heroku

by: heroku

digitalocean

by: digitalocean

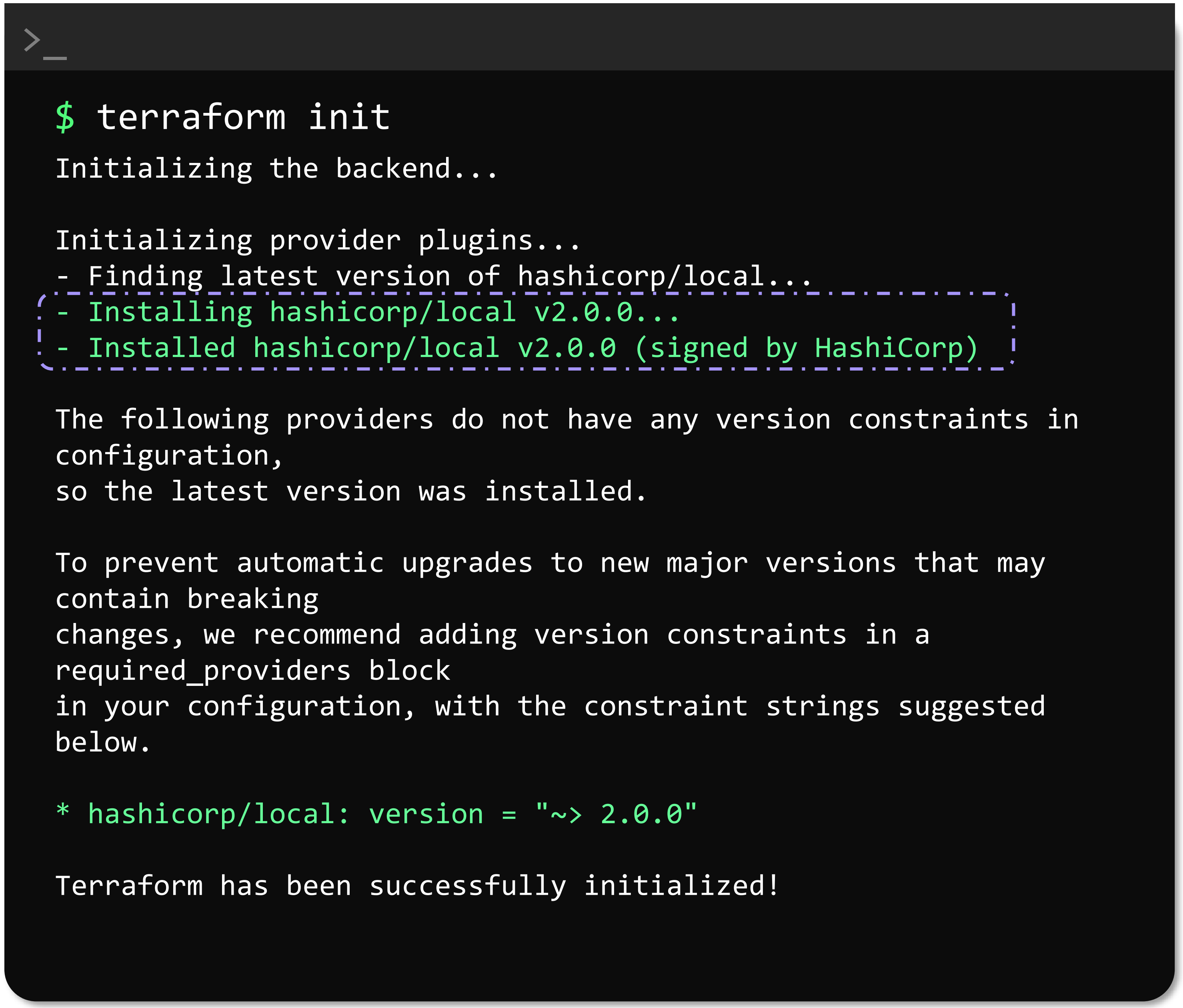
Community

activedirectory

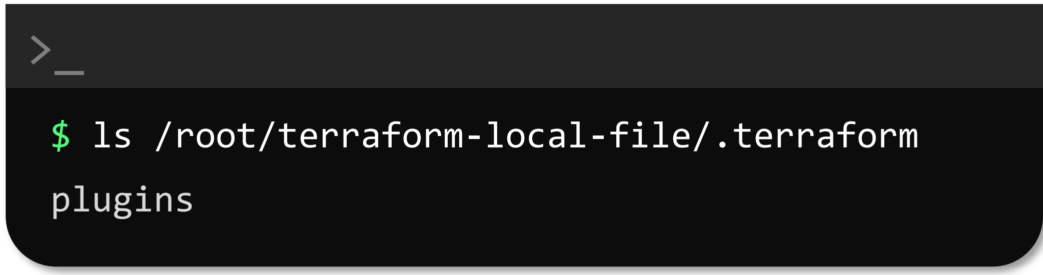
ucloud

netapp-gcp

registry.terraform.io



```
$ terraform init
Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/local...
- Installing hashicorp/local v2.0.0...
- Installed hashicorp/local v2.0.0 (signed by HashiCorp)

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/local: version = "~> 2.0.0"

Terraform has been successfully initialized!
```



```
$ ls /root/terraform-local-file/.terraform
plugins
```

```
To prevent automatic upgrades to new maj
contain breaking
changes, we recommend adding version con
required_providers block
in your configuration, with the constrai
below.

* hashicorp/local: version = "~> 2.0.0"
```

Organizational Namespace

Type

```
Terraform has been successfully initiali
```

```
To prevent automatic upgrades to new maj
contain breaking
changes, we recommend adding version con
required_providers block
in your configuration, with the constrai
below.

*   registry.terraform.io/hashicorp/loc
```

Hostname

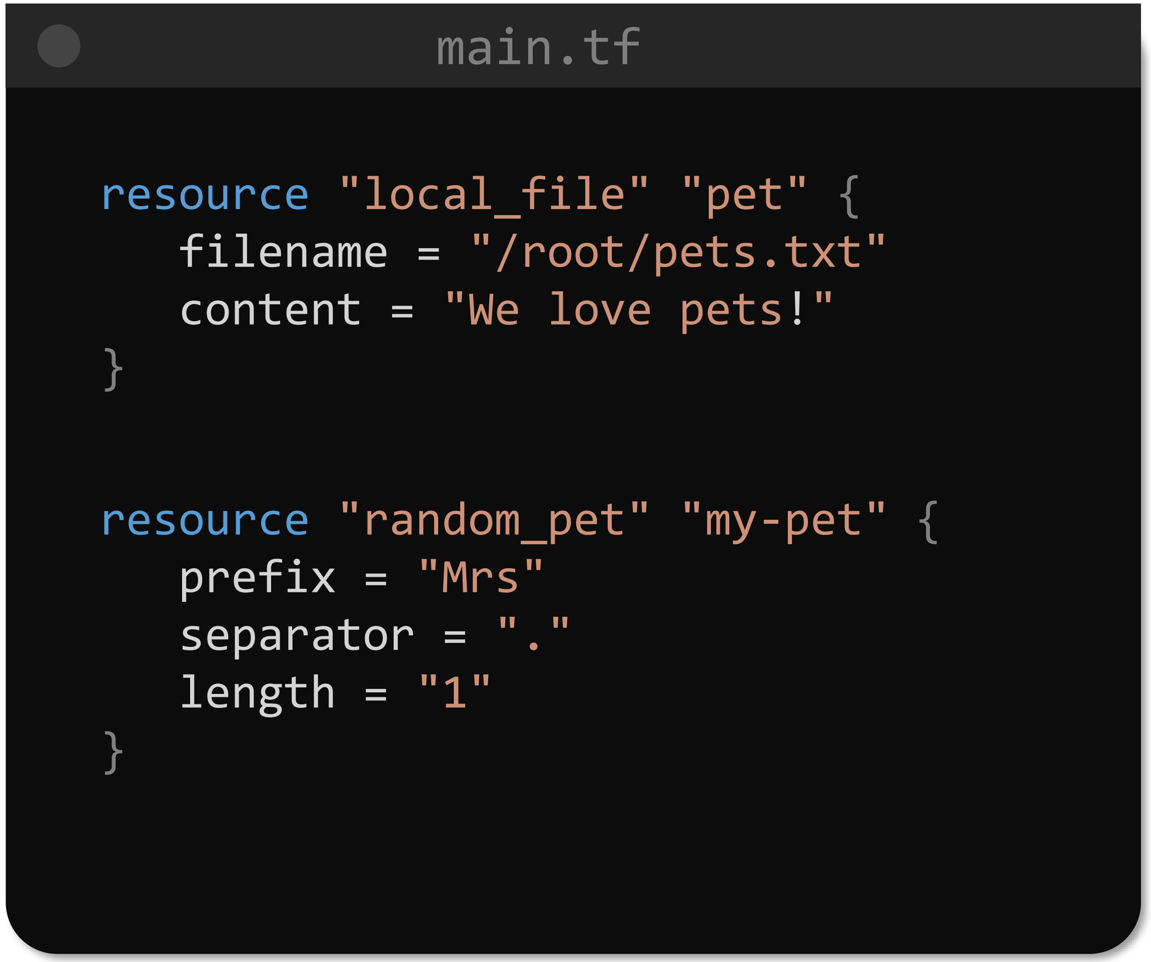
Organizational Namespace

Ty

```
Terraform has been successfully initiali
```

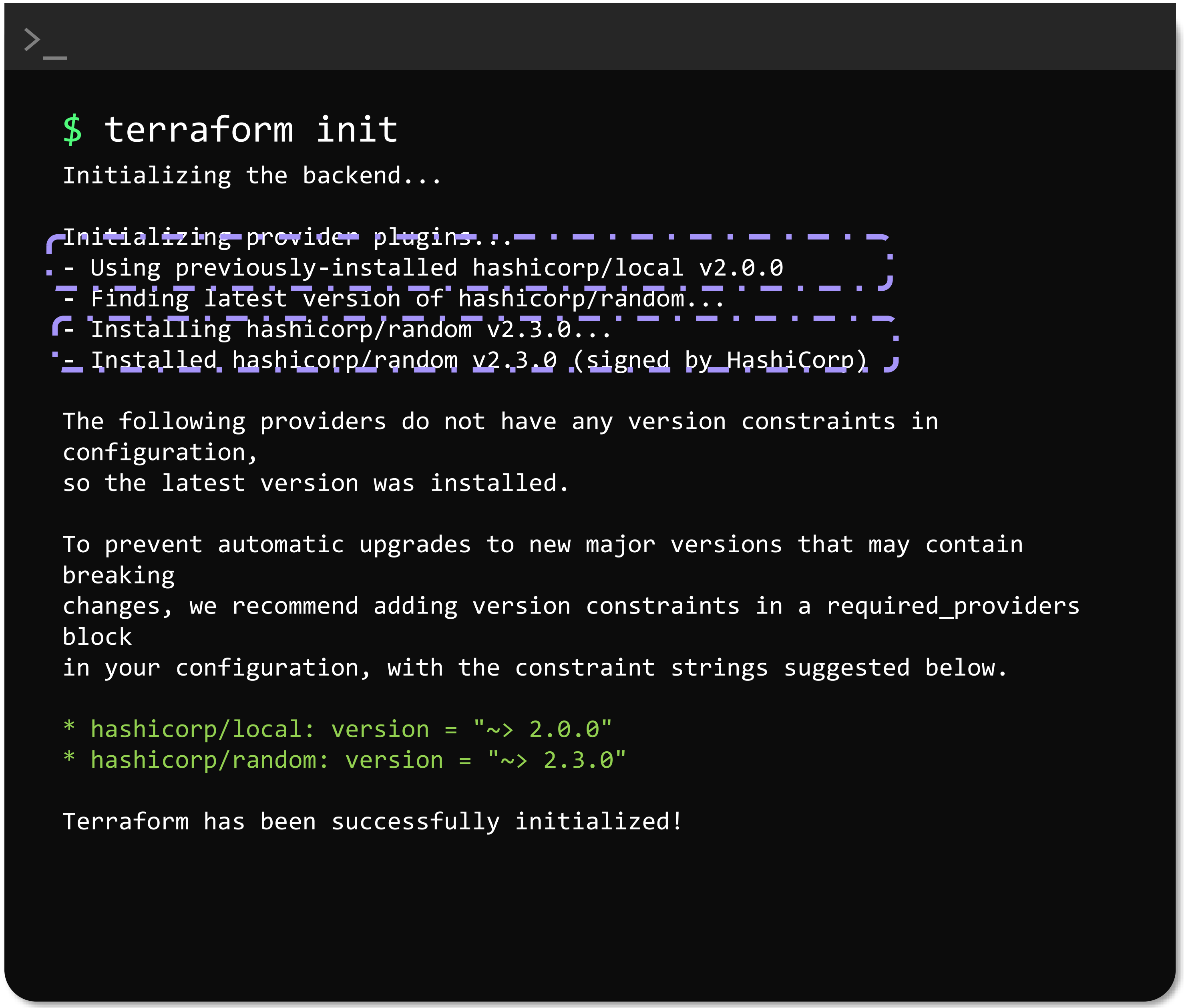
KODEKLOUD

Multiple Providers

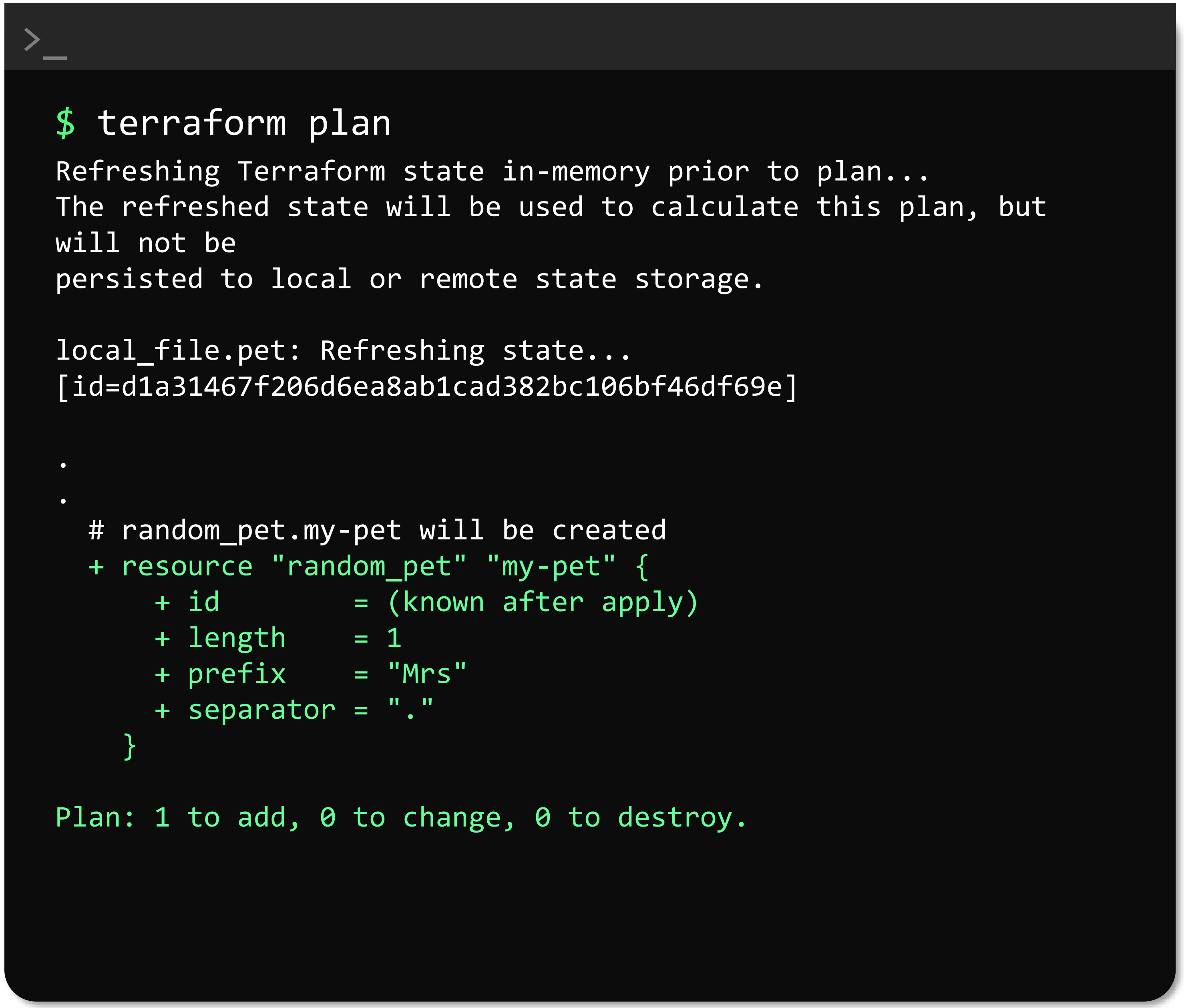


main.tf

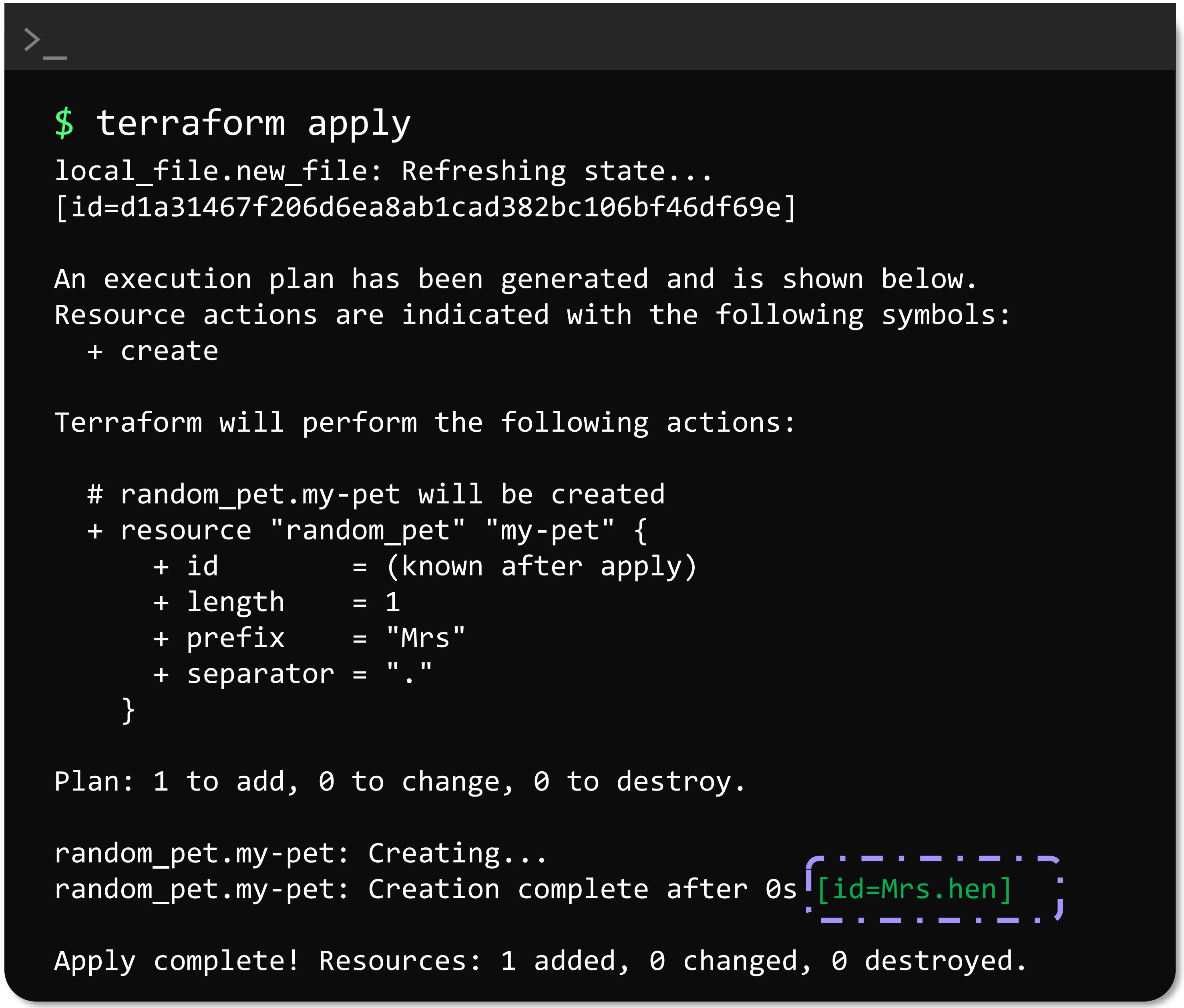
```
resource "local_file" "pet" {
   filename = "/root/pets.txt"
   content = "We love pets!"
}


resource "random_pet" "my-pet" {
   prefix = "Mrs"
   separator = "."
   length = "1"
}
```



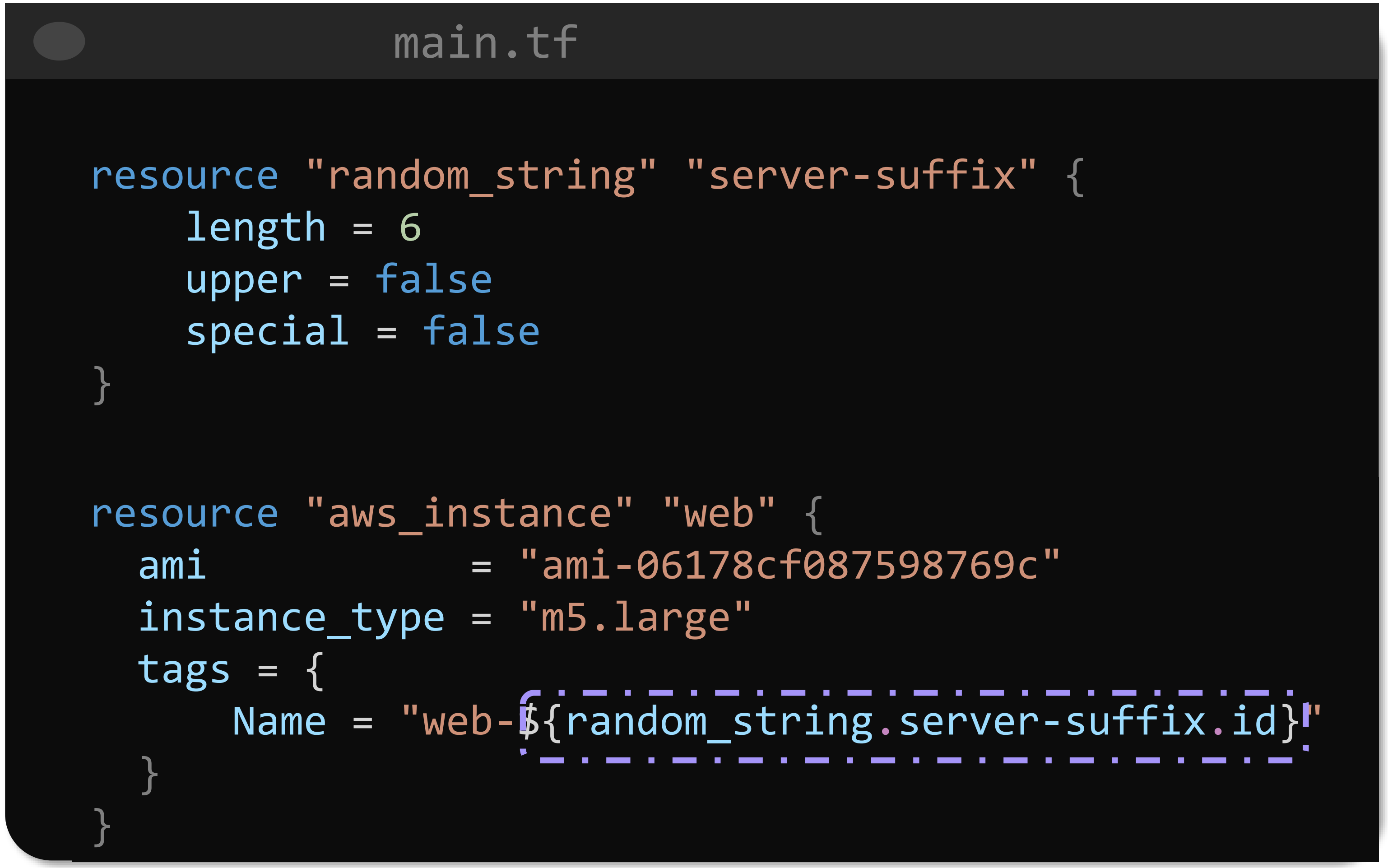
```
$ terraform init
Initializing the backend...

Initializing provider plugins...
- Using previously-installed hashicorp/local v2.0.0
- Finding latest version of hashicorp/random...
- Installing hashicorp/random v2.3.0...
- Installed hashicorp/random v2.3.0 (signed by HashiCorp)

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/local: version = "~> 2.0.0"
* hashicorp/random: version = "~> 2.3.0"

Terraform has been successfully initialized!
```



```
$ 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.

local_file.pet: Refreshing state...
[id=d1a31467f206d6ea8ab1cad382bc106bf46df69e]

.
.
  # random_pet.my-pet will be created
  + resource "random_pet" "my-pet" {
      + id        = (known after apply)
      + length    = 1
      + prefix    = "Mrs"
      + separator = "."
    }

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

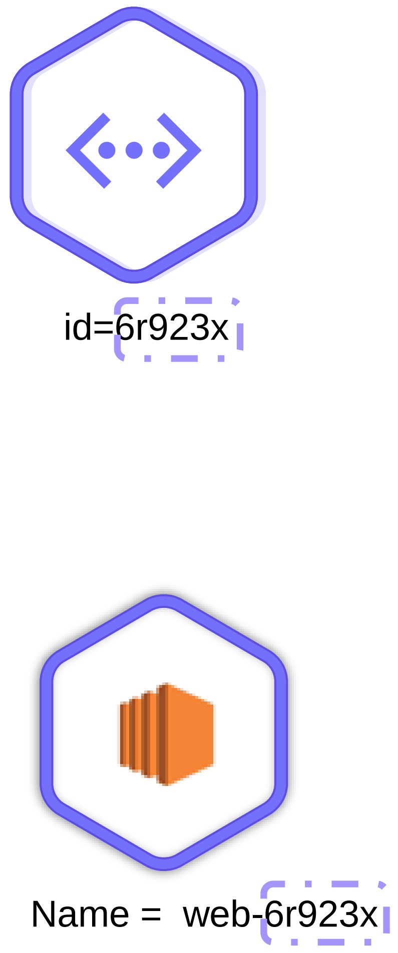


```
$ terraform apply
local_file.new_file: Refreshing state...
[id=d1a31467f206d6ea8ab1cad382bc106bf46df69e]

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

Terraform will perform the following actions:

  # random_pet.my-pet will be created
  + resource "random_pet" "my-pet" {
      + id        = (known after apply)
      + length    = 1
      + prefix    = "Mrs"
      + separator = "."
    }

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

random_pet.my-pet: Creating...
random_pet.my-pet: Creation complete after 0s [id=Mrs.hen]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```



main.tf

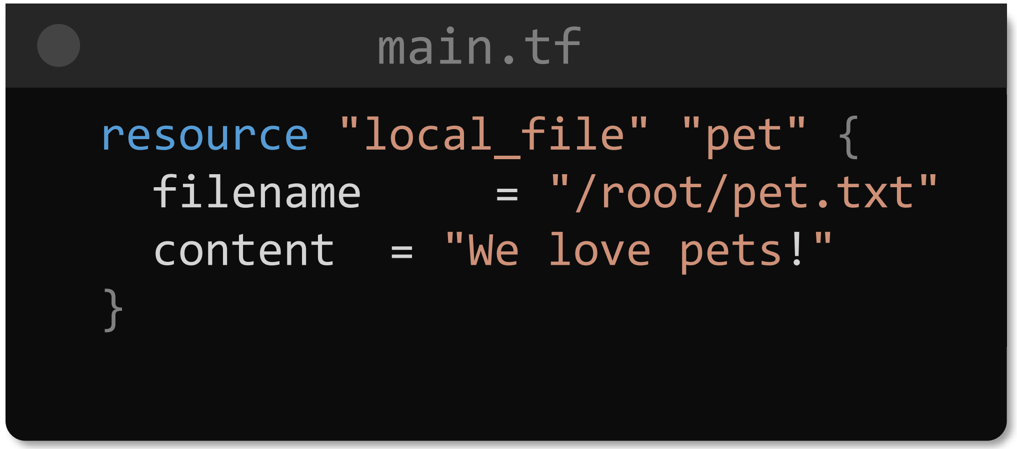
```
resource "random_string" "server-suffix" {
    length = 6
    upper = false
    special = false
}

resource "aws_instance" "web" {
  ami           = "ami-06178cf087598769c"
  instance_type = "m5.large"
  tags = {
      Name = "web-${random_string.server-suffix.id}"
  }
}
```



KODEKLOUD

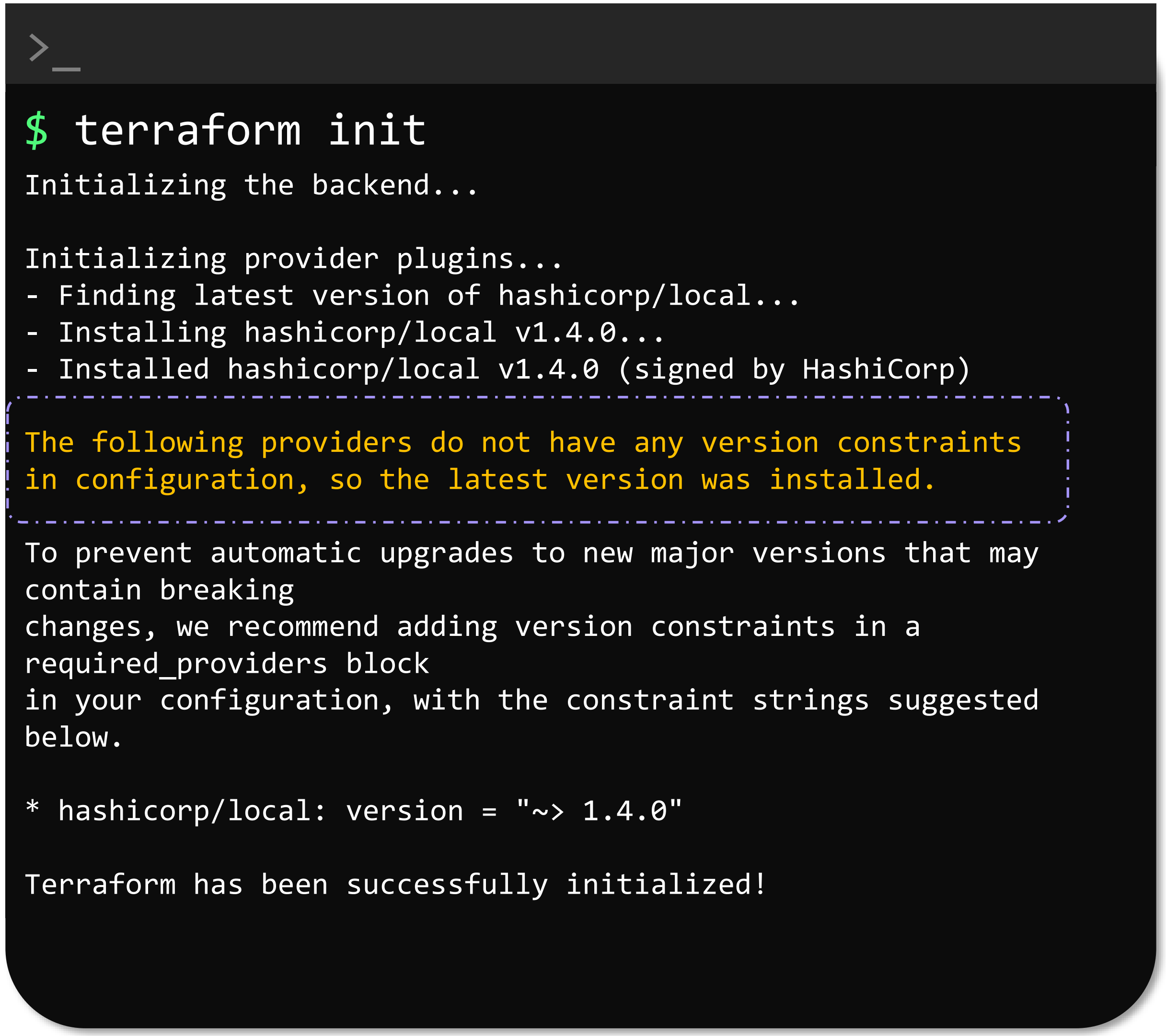
KODEKLOUD

Version Constraints



main.tf

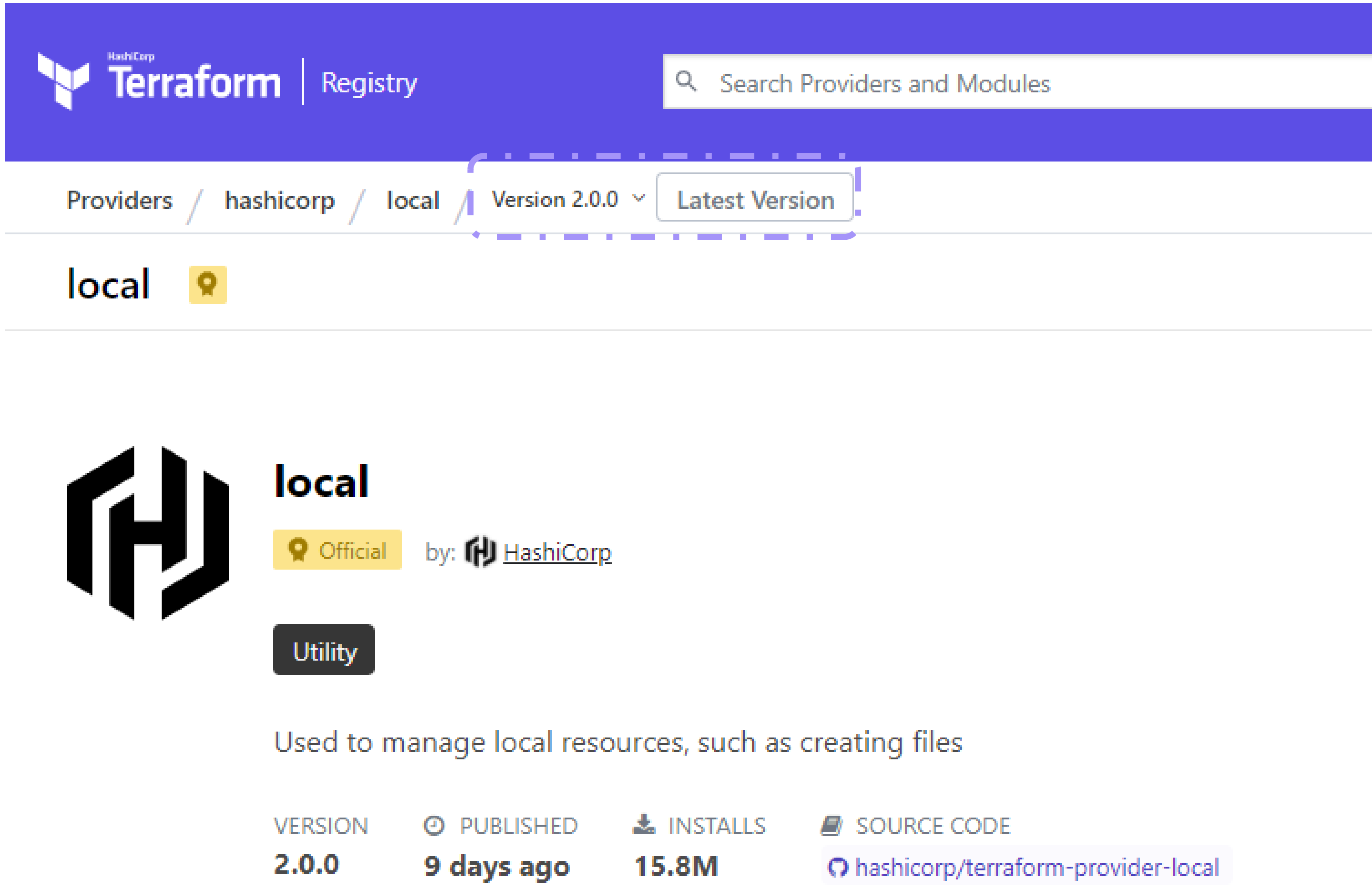
```
resource "local_file" "pet" {
  filename      = "/root/pet.txt"
  content   = "We love pets!"
}
```



```
$ terraform init
Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/local...
- Installing hashicorp/local v1.4.0...
- Installed hashicorp/local v1.4.0 (signed by HashiCorp)

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/local: version = "~> 1.4.0"

Terraform has been successfully initialized!
```

main.tf

```
resource "local_file" "pet" {
  filename     = "/root/pet.txt"
  content  = "We love pets!"
}
```



Terraform | Registry

Search Providers and Modules

Providers / hashicorp / local / Version 2.0.0 Latest Version

local

local

Official by: HashiCorp

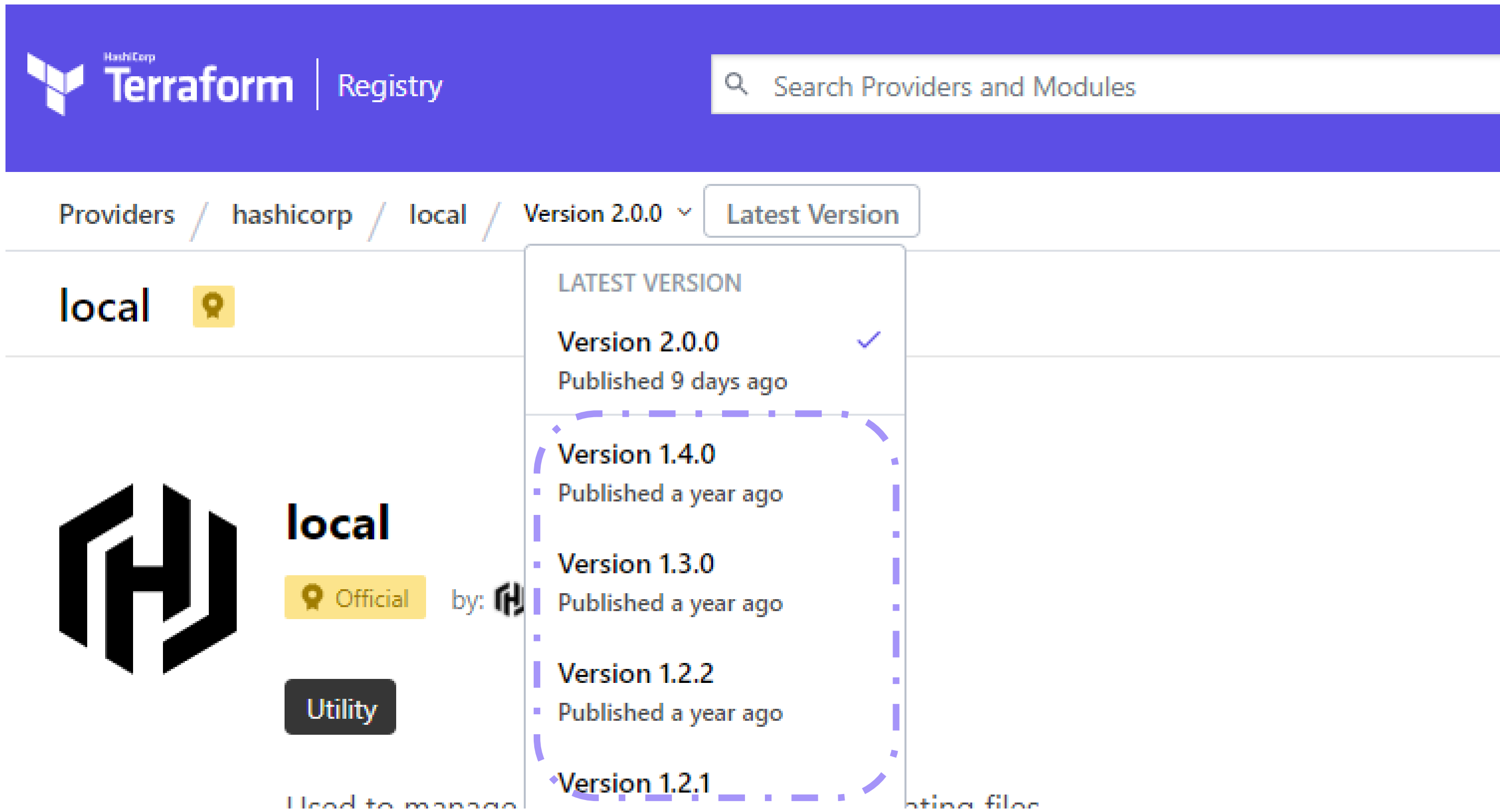
Utility

Used to manage local resources, such as creating files

VERSION	PUBLISHED	INSTALLS	SOURCE CODE
2.0.0	9 days ago	15.8M	hashicorp/terraform-provider-local

main.tf

```
resource "local_file" "pet" {
  filename     = "/root/pet.txt"
  content  = "We love pets!"
}
```



Terraform | Registry

Search Providers and Modules

Providers / hashicorp / local / Version 2.0.0 ⌄ Latest Version

local

LATEST VERSION

Version 2.0.0 ✓
Published 9 days ago

Version 1.4.0
Published a year ago

Version 1.3.0
Published a year ago

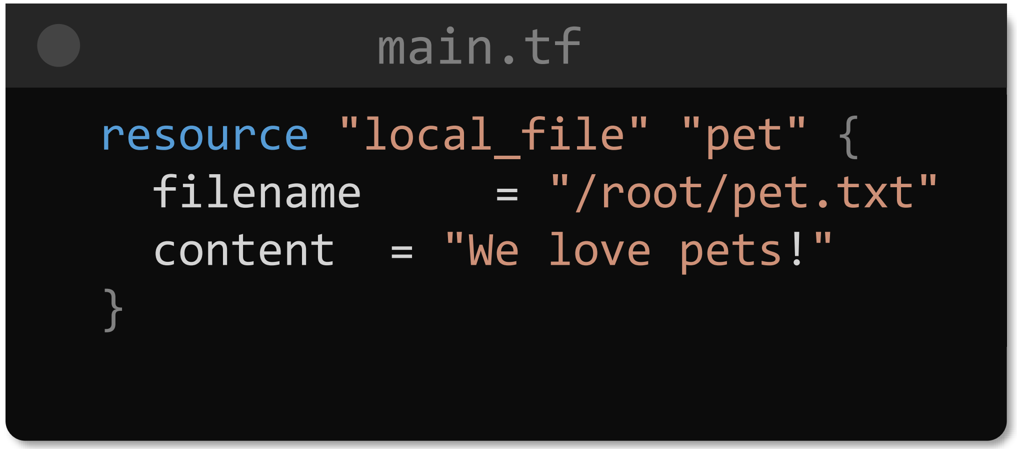
Version 1.2.2
Published a year ago

Version 1.2.1

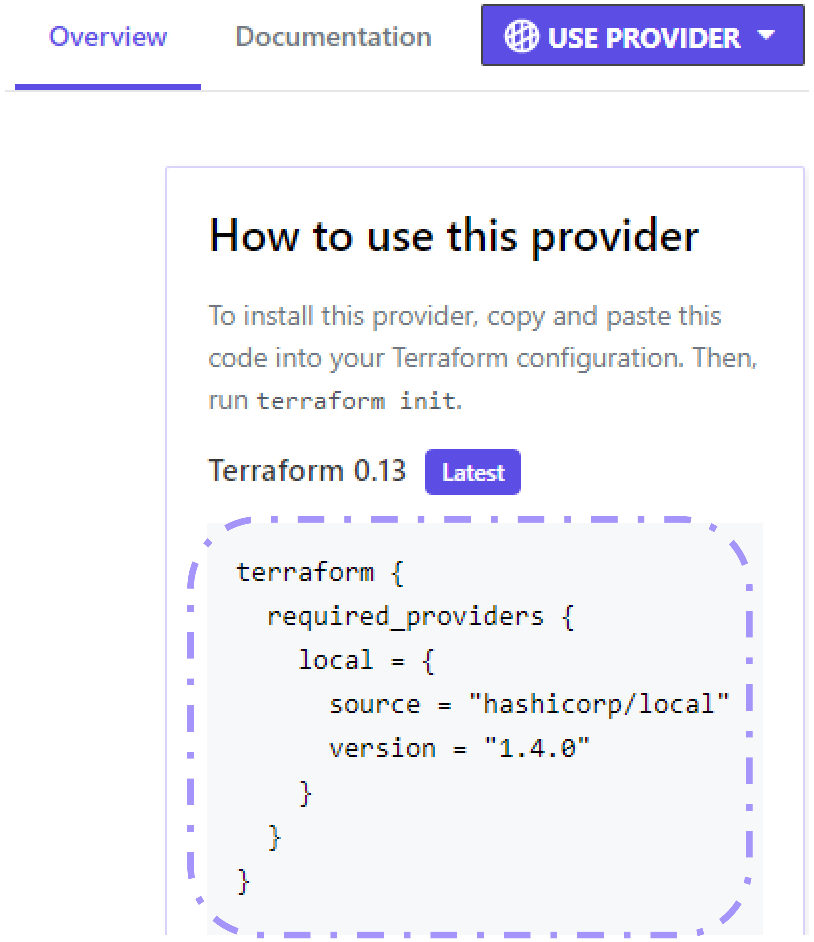
local

Official by:

Utility



```
main.tf

resource "local_file" "pet" {
  filename     = "/root/pet.txt"
  content   = "We love pets!"
}
```

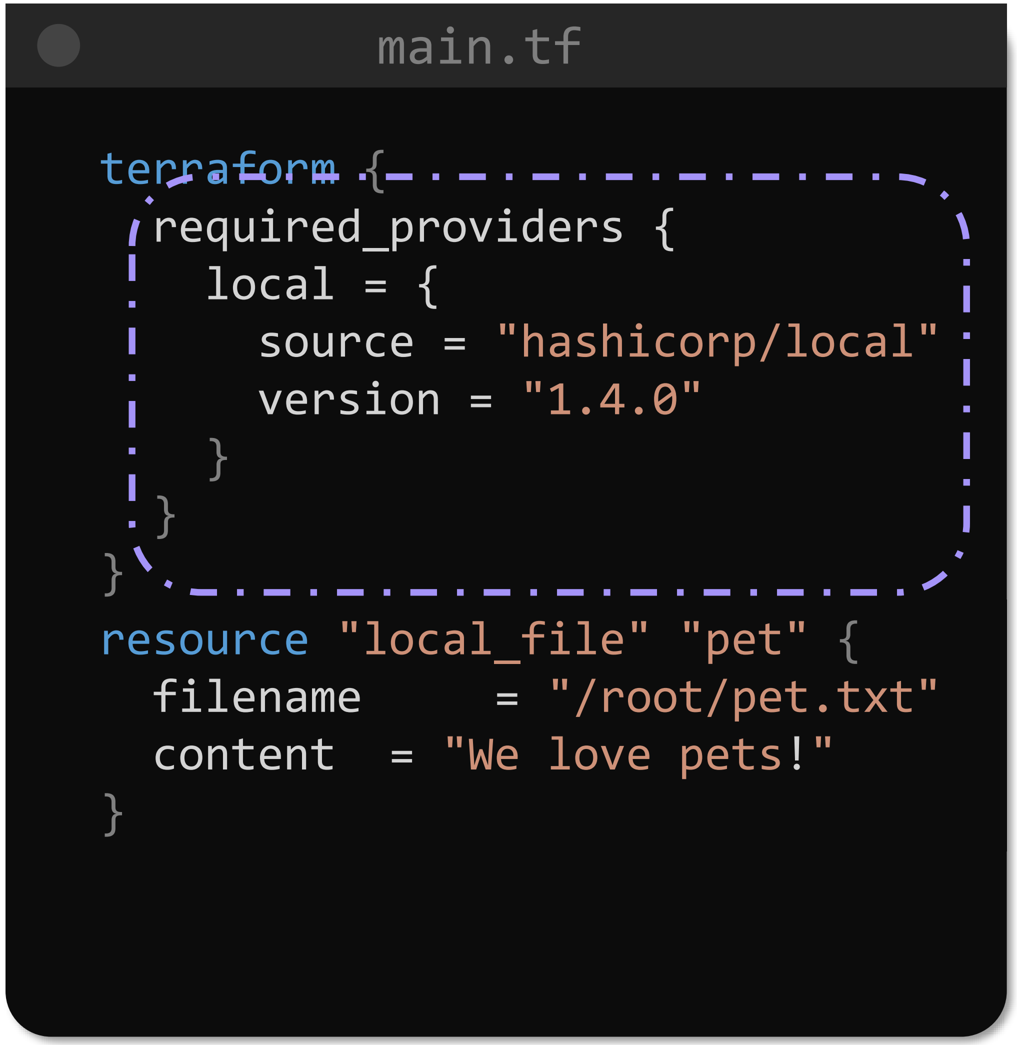


How to use this provider

To install this provider, copy and paste this code into your Terraform configuration. Then, run `terraform init`.

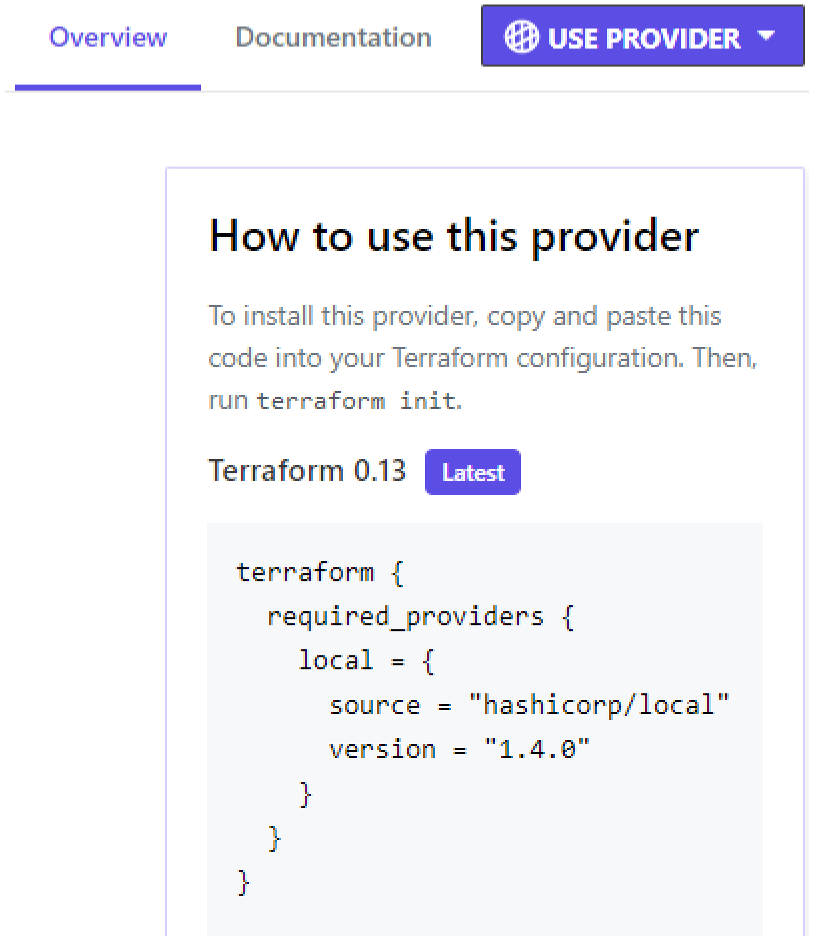
Terraform 0.13 Latest

```
terraform {
  required_providers {
    local = {
      source = "hashicorp/local"
      version = "1.4.0"
    }
  }
}
```



main.tf

```
terraform {
  required_providers {
    local = {
      source = "hashicorp/local"
      version = "1.4.0"
    }
  }
}
resource "local_file" "pet" {
  filename      = "/root/pet.txt"
  content   = "We love pets!"
}
```

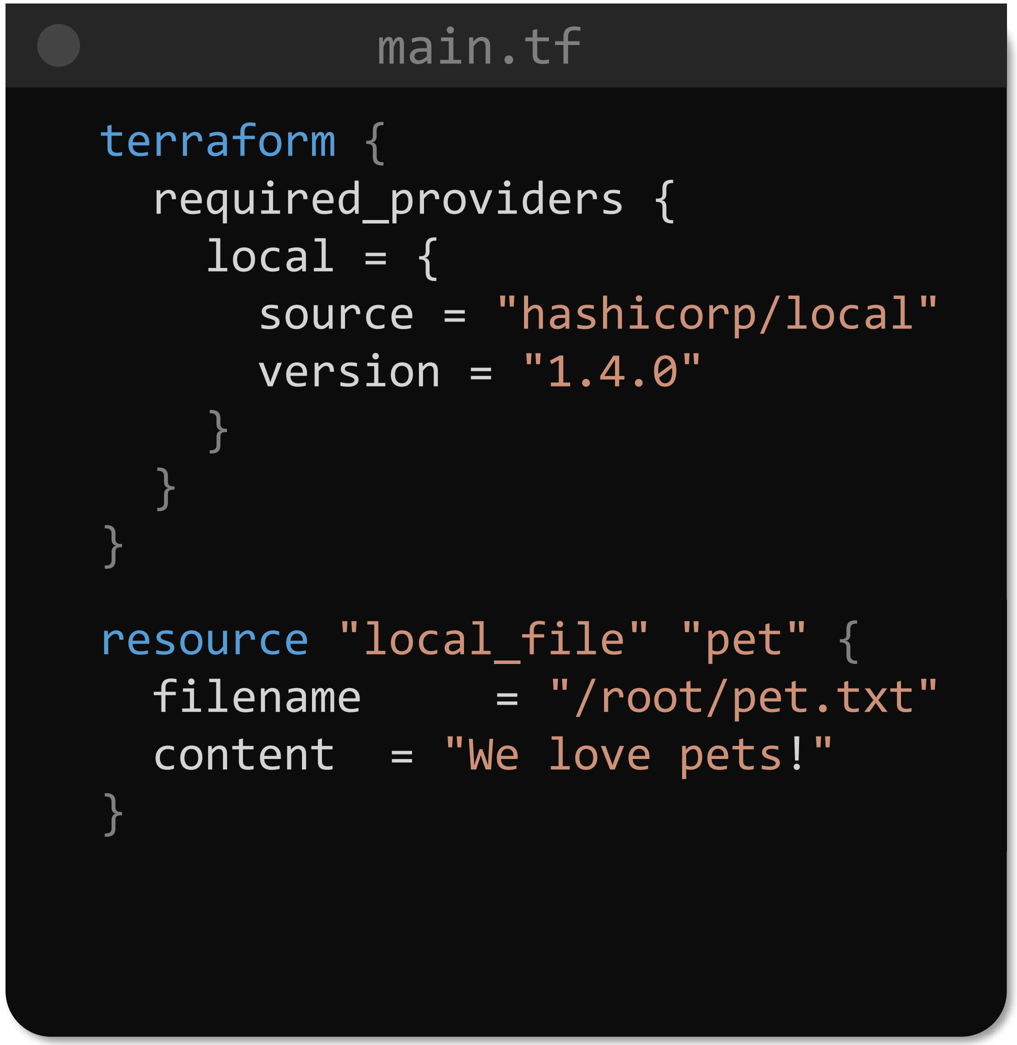


How to use this provider

To install this provider, copy and paste this code into your Terraform configuration. Then, run `terraform init`.

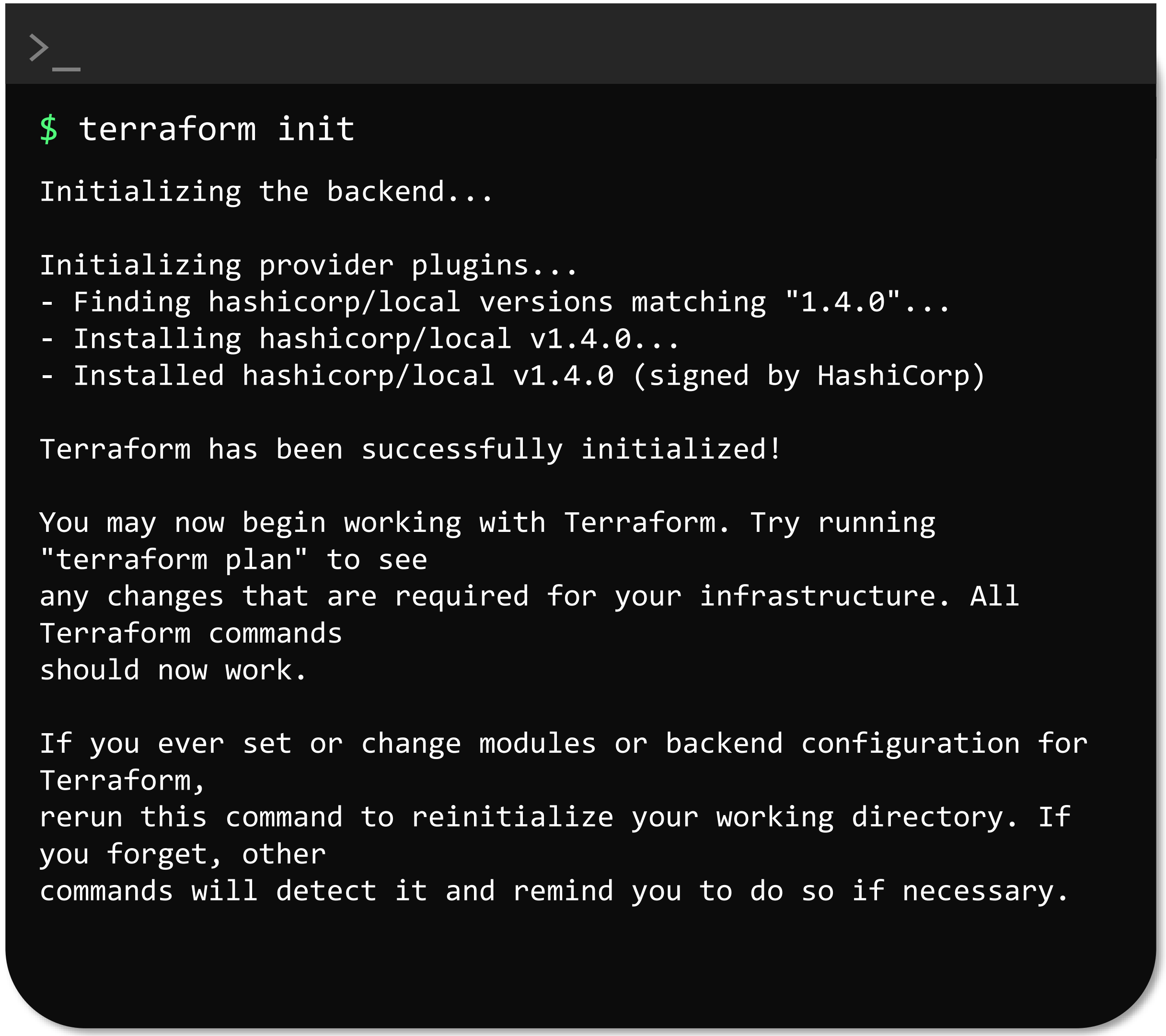
Terraform 0.13 Latest

```
terraform {
  required_providers {
    local = {
      source = "hashicorp/local"
      version = "1.4.0"
    }
  }
}
```

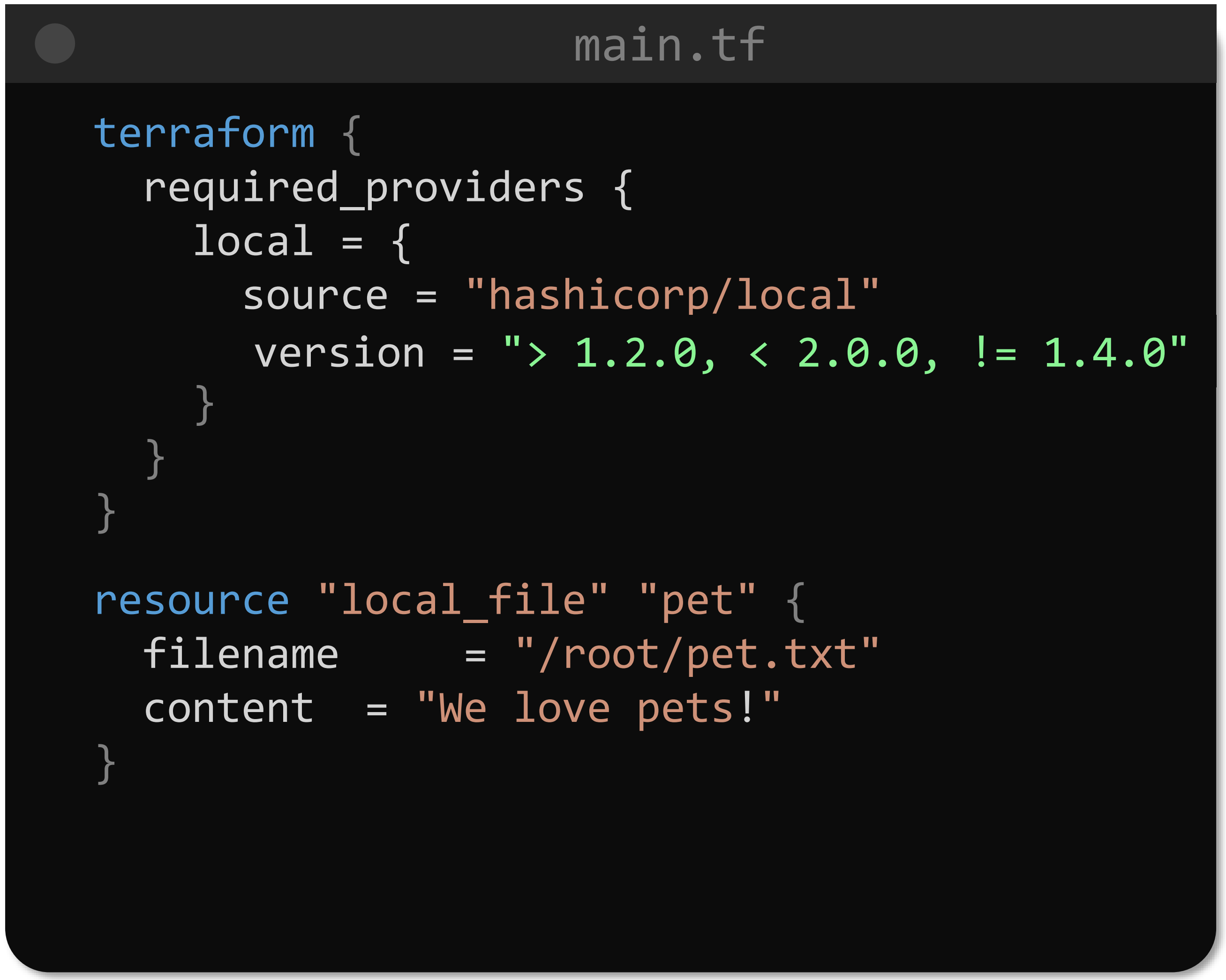


main.tf

```
terraform {
  required_providers {
    local = {
      source = "hashicorp/local"
      version = "1.4.0"
    }
  }
}

resource "local_file" "pet" {
  filename     = "/root/pet.txt"
  content   = "We love pets!"
}
```

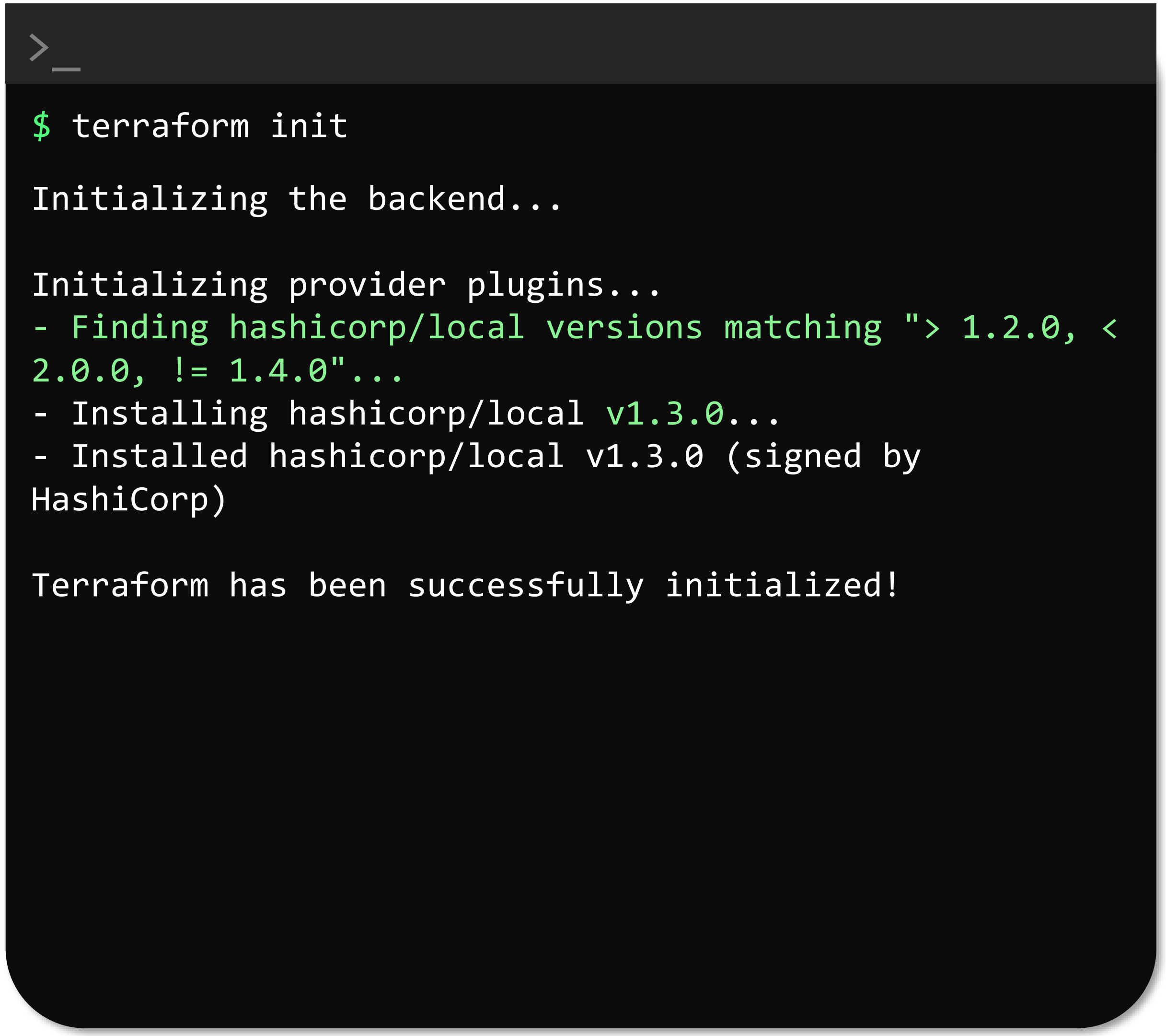


```
$ terraform init
Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/local versions matching "1.4.0"...
- Installing hashicorp/local v1.4.0...
- Installed hashicorp/local v1.4.0 (signed by HashiCorp)

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.

If you ever set or change modules or backend configuration for
Terraform,
rerun this command to reinitialize your working directory. If
you forget, other
commands will detect it and remind you to do so if necessary.
```

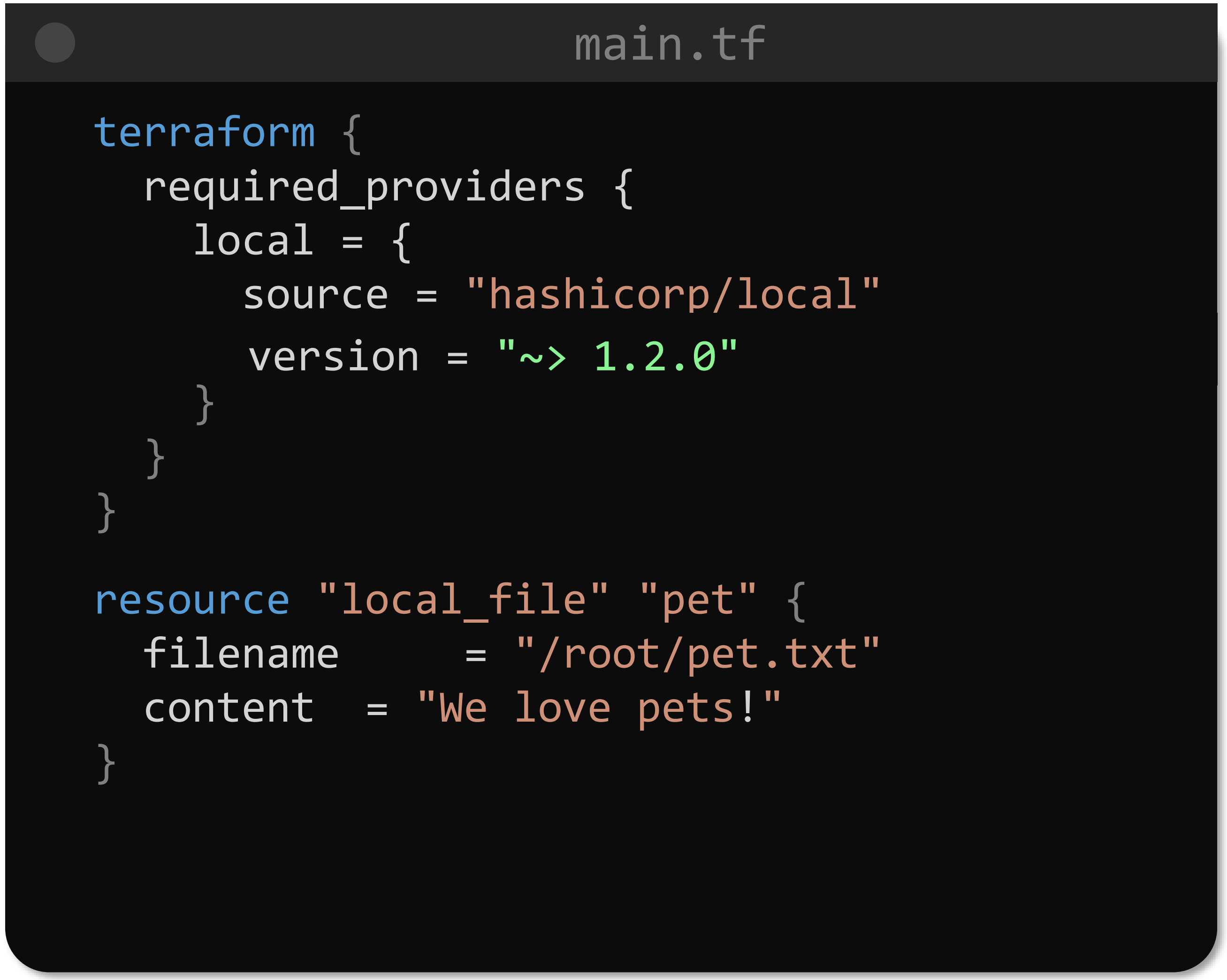


main.tf

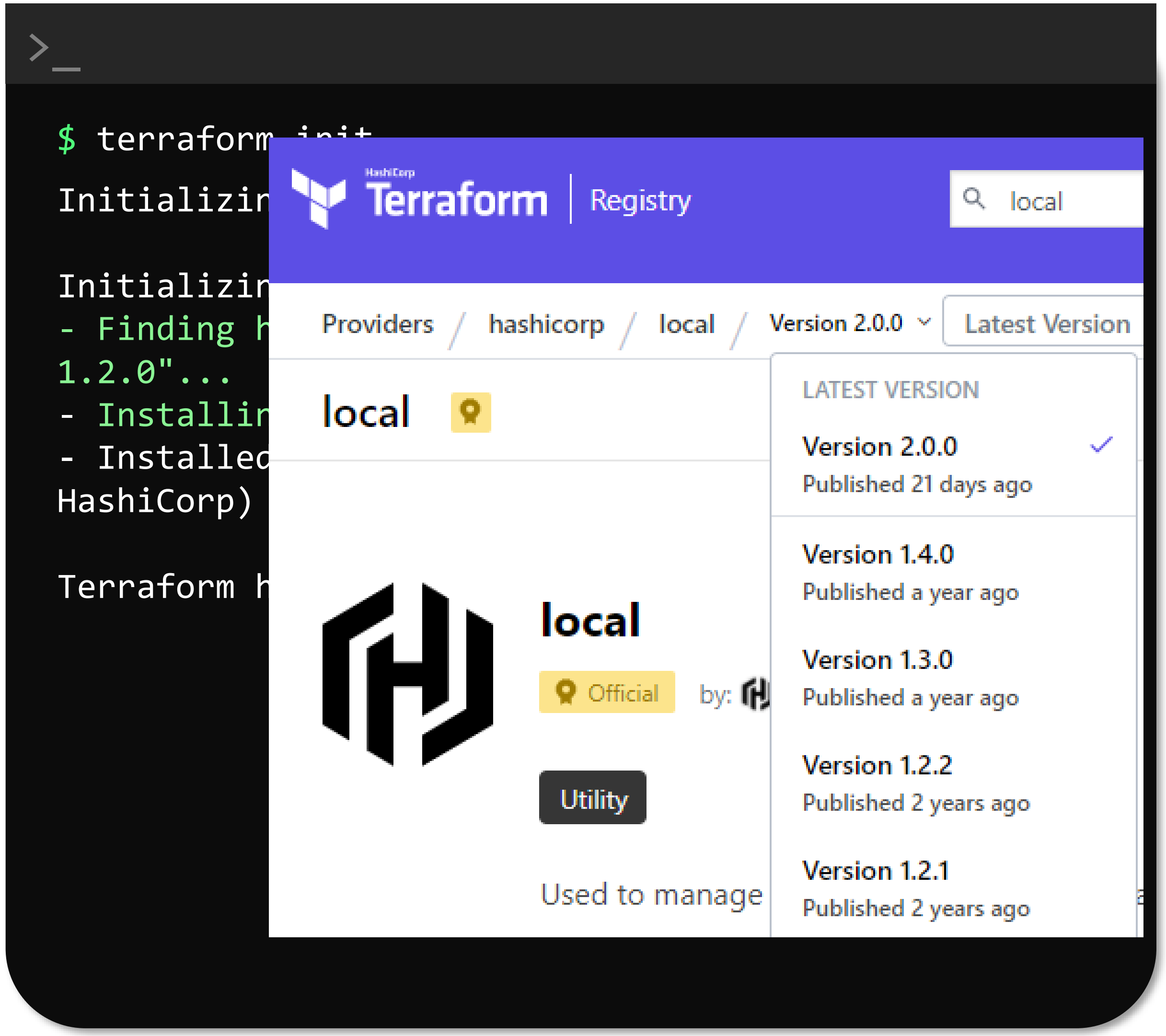
```
terraform {
  required_providers {
    local = {
      source = "hashicorp/local"
       version = "> 1.2.0, < 2.0.0, != 1.4.0"
    }
  }
}

resource "local_file" "pet" {
  filename      = "/root/pet.txt"
  content   = "We love pets!"
}
```



```
$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/local versions matching "> 1.2.0, <
2.0.0, != 1.4.0"...
- Installing hashicorp/local v1.3.0...
- Installed hashicorp/local v1.3.0 (signed by
HashiCorp)

Terraform has been successfully initialized!
```


main.tf

```
terraform {
  required_providers {
    local = {
      source = "hashicorp/local"
      version = "~> 1.2.0"
    }
  }
}

resource "local_file" "pet" {
  filename     = "/root/pet.txt"
  content  = "We love pets!"
}
```


```
$ terraform init
Initializin

Initializin
- Finding h
1.2.0"...
- Installin
- Installed
HashiCorp)

Terraform h
```

LATEST VERSION

Version 2.0.0 ✓
Published 21 days ago

Version 1.4.0
Published a year ago

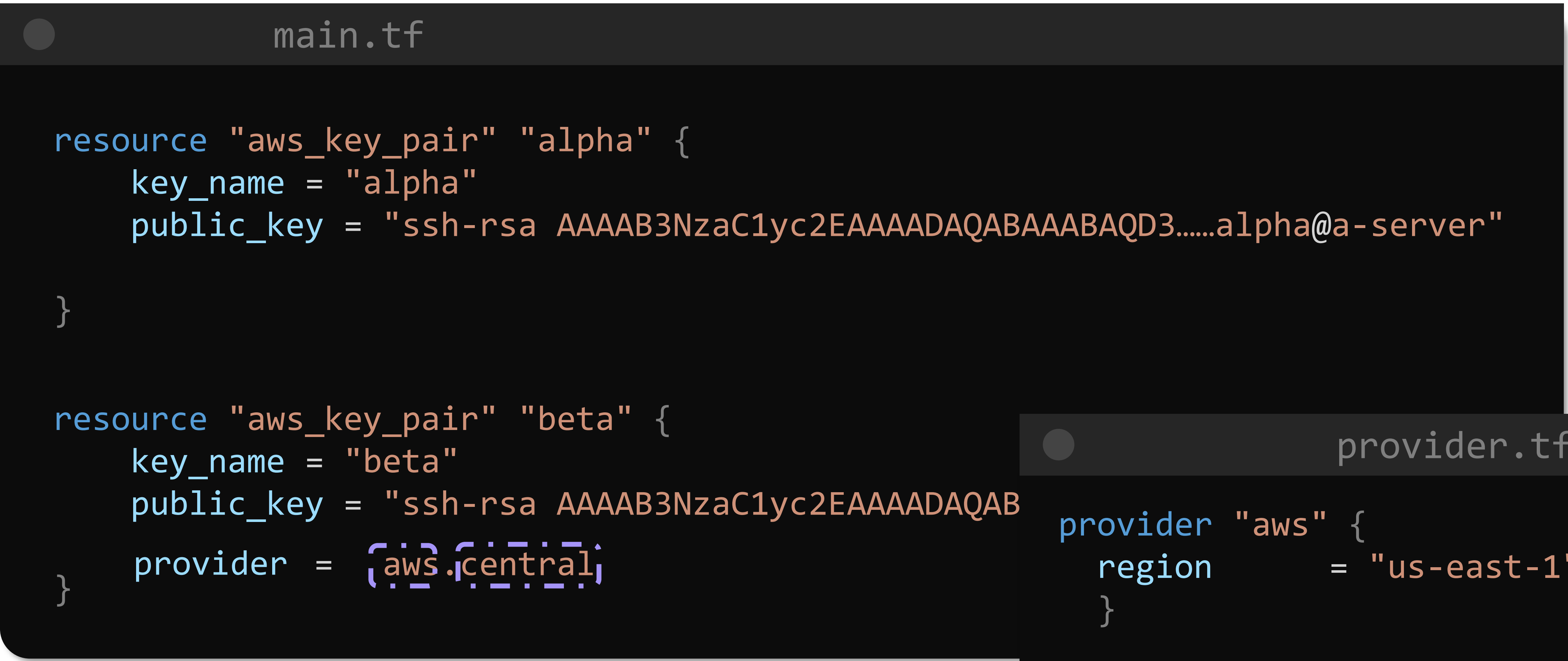
Version 1.3.0
Published a year ago

Version 1.2.2
Published 2 years ago

Version 1.2.1
Published 2 years ago

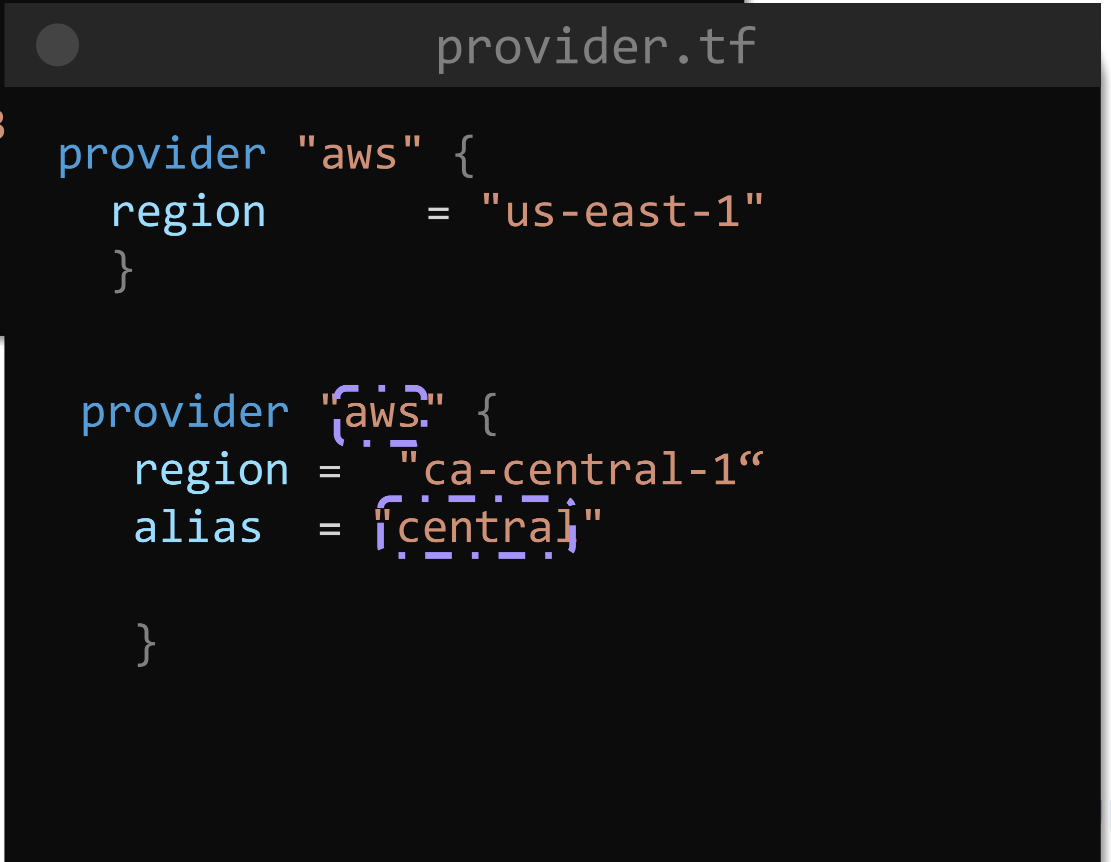
KODEKLOUD

Aliases



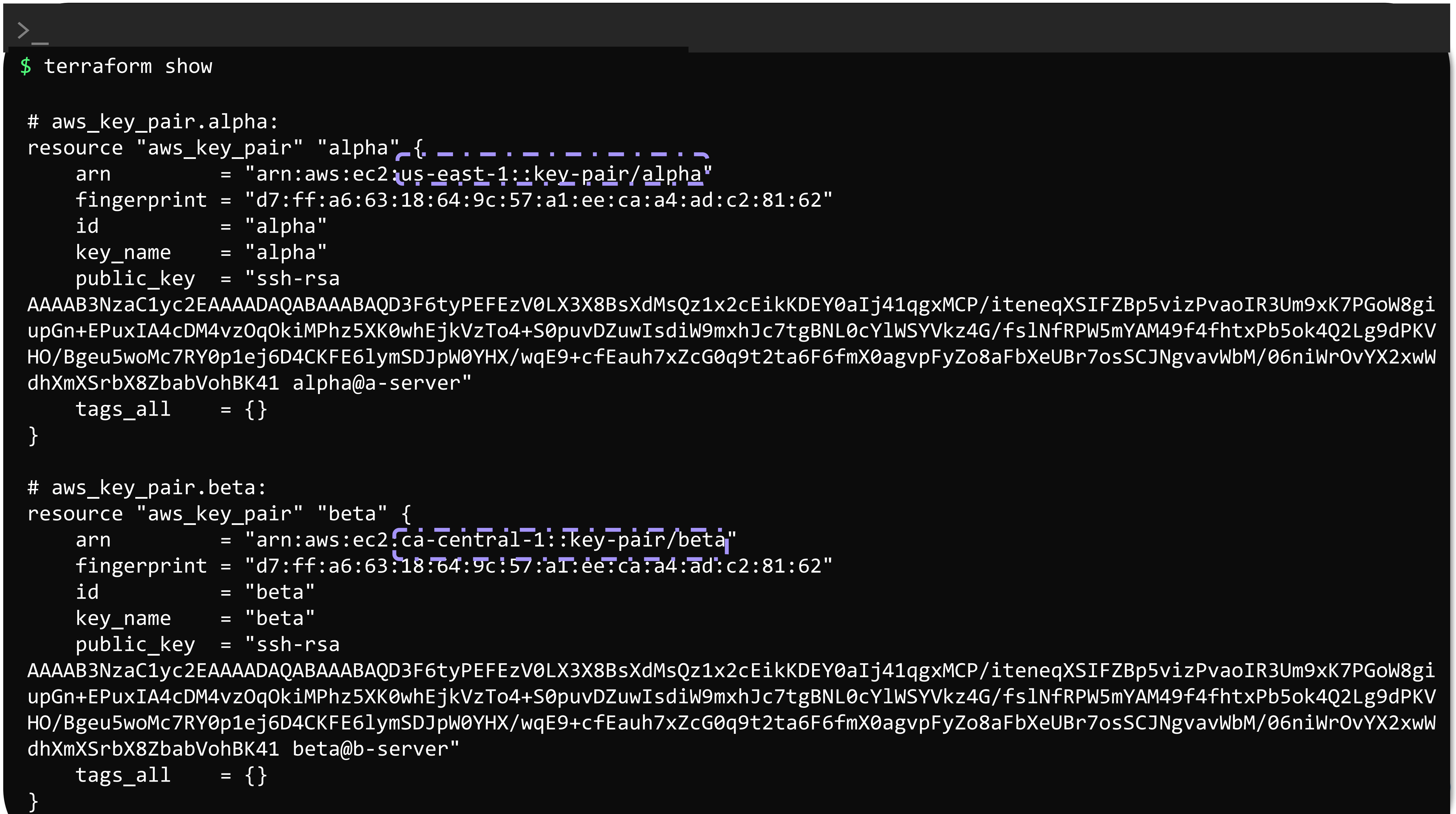
main.tf

```
resource "aws_key_pair" "alpha" {
    key_name = "alpha"
    public_key = "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQD3……alpha@a-server"

}


resource "aws_key_pair" "beta" {
    key_name = "beta"
    public_key = "ssh-rsa AAAAB3NzaC1yc2EAAAADAQAB
    provider  =  aws.central
}
```



provider.tf

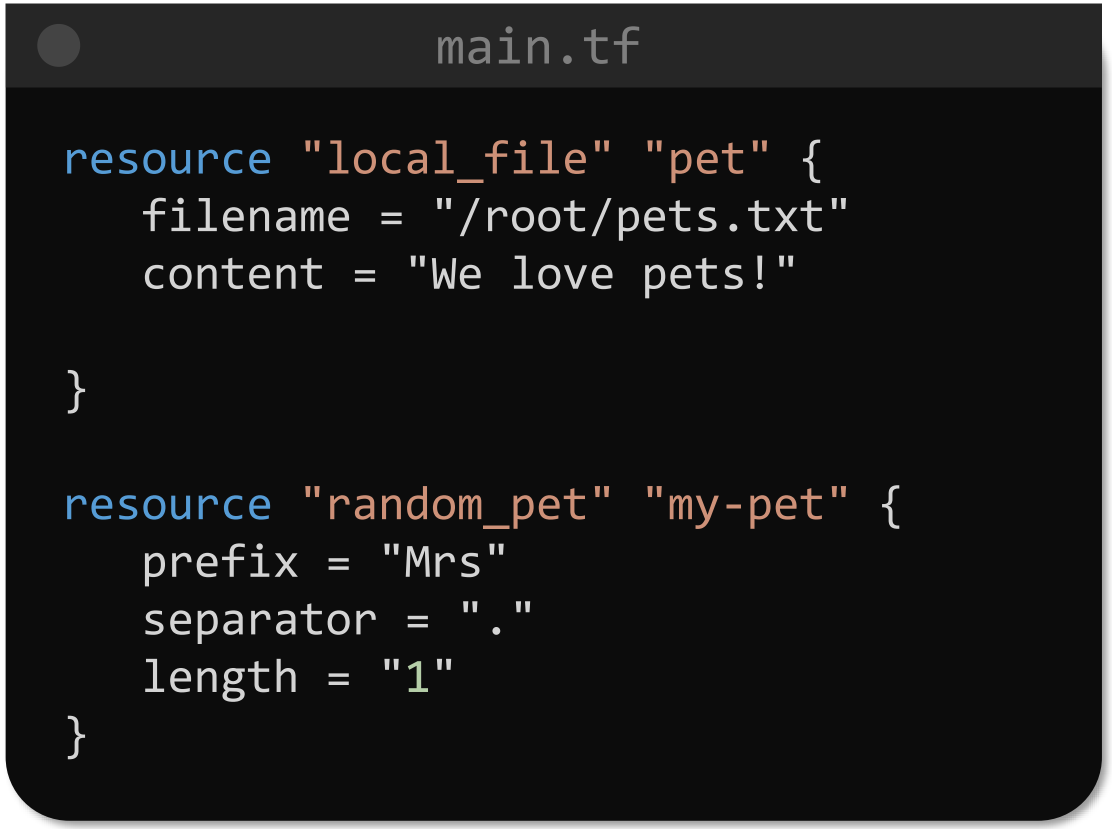
```
provider "aws" {
  region       = "us-east-1"
  }


 provider "aws" {
   region =  "ca-central-1“
   alias  = "central"

   }
```



```
$ terraform show

# aws_key_pair.alpha:
resource "aws_key_pair" "alpha" {
    arn         = "arn:aws:ec2:us-east-1::key-pair/alpha"
    fingerprint = "d7:ff:a6:63:18:64:9c:57:a1:ee:ca:a4:ad:c2:81:62"
    id          = "alpha"
    key_name    = "alpha"
    public_key  = "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQD3F6tyPEFEzV0LX3X8BsXdMsQz1x2cEikKDEY0aIj41qgxMCP/iteneqXSIFZBp5vizPvaoIR3Um9xK7PGoW8gi
upGn+EPuxIA4cDM4vzOqOkiMPhz5XK0whEjkVzTo4+S0puvDZuwIsdiW9mxhJc7tgBNL0cYlWSYVkz4G/fslNfRPW5mYAM49f4fhtxPb5ok4Q2Lg9dPKV
HO/Bgeu5woMc7RY0p1ej6D4CKFE6lymSDJpW0YHX/wqE9+cfEauh7xZcG0q9t2ta6F6fmX0agvpFyZo8aFbXeUBr7osSCJNgvavWbM/06niWrOvYX2xwW
dhXmXSrbX8ZbabVohBK41 alpha@a-server"
    tags_all    = {}
}

# aws_key_pair.beta:
resource "aws_key_pair" "beta" {
    arn         = "arn:aws:ec2:ca-central-1::key-pair/beta"
    fingerprint = "d7:ff:a6:63:18:64:9c:57:a1:ee:ca:a4:ad:c2:81:62"
    id          = "beta"
    key_name    = "beta"
    public_key  = "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQD3F6tyPEFEzV0LX3X8BsXdMsQz1x2cEikKDEY0aIj41qgxMCP/iteneqXSIFZBp5vizPvaoIR3Um9xK7PGoW8gi
upGn+EPuxIA4cDM4vzOqOkiMPhz5XK0whEjkVzTo4+S0puvDZuwIsdiW9mxhJc7tgBNL0cYlWSYVkz4G/fslNfRPW5mYAM49f4fhtxPb5ok4Q2Lg9dPKV
HO/Bgeu5woMc7RY0p1ej6D4CKFE6lymSDJpW0YHX/wqE9+cfEauh7xZcG0q9t2ta6F6fmX0agvpFyZo8aFbXeUBr7osSCJNgvavWbM/06niWrOvYX2xwW
dhXmXSrbX8ZbabVohBK41 beta@b-server"
    tags_all    = {}
}
```

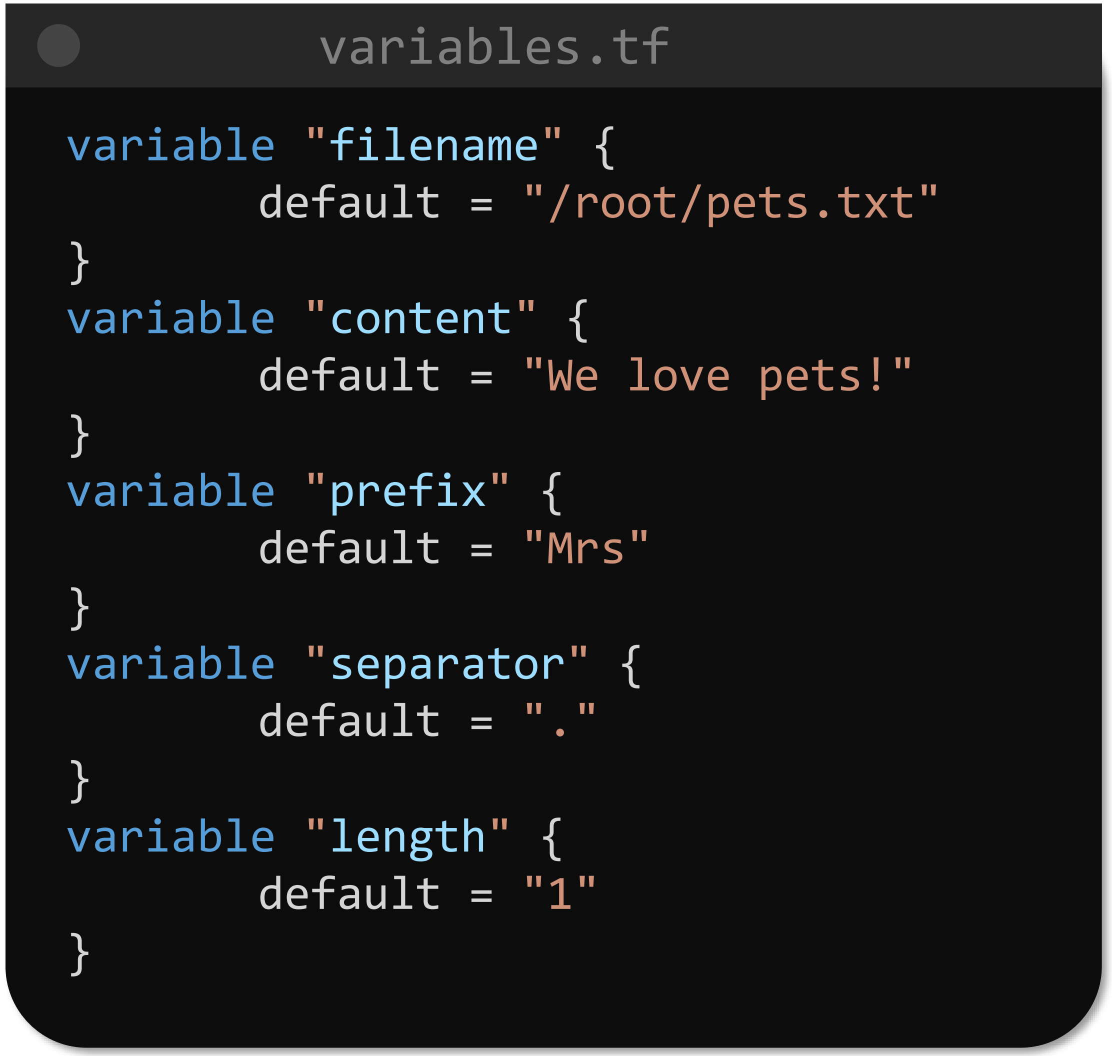
KODEKLOUD

Define Input Variables



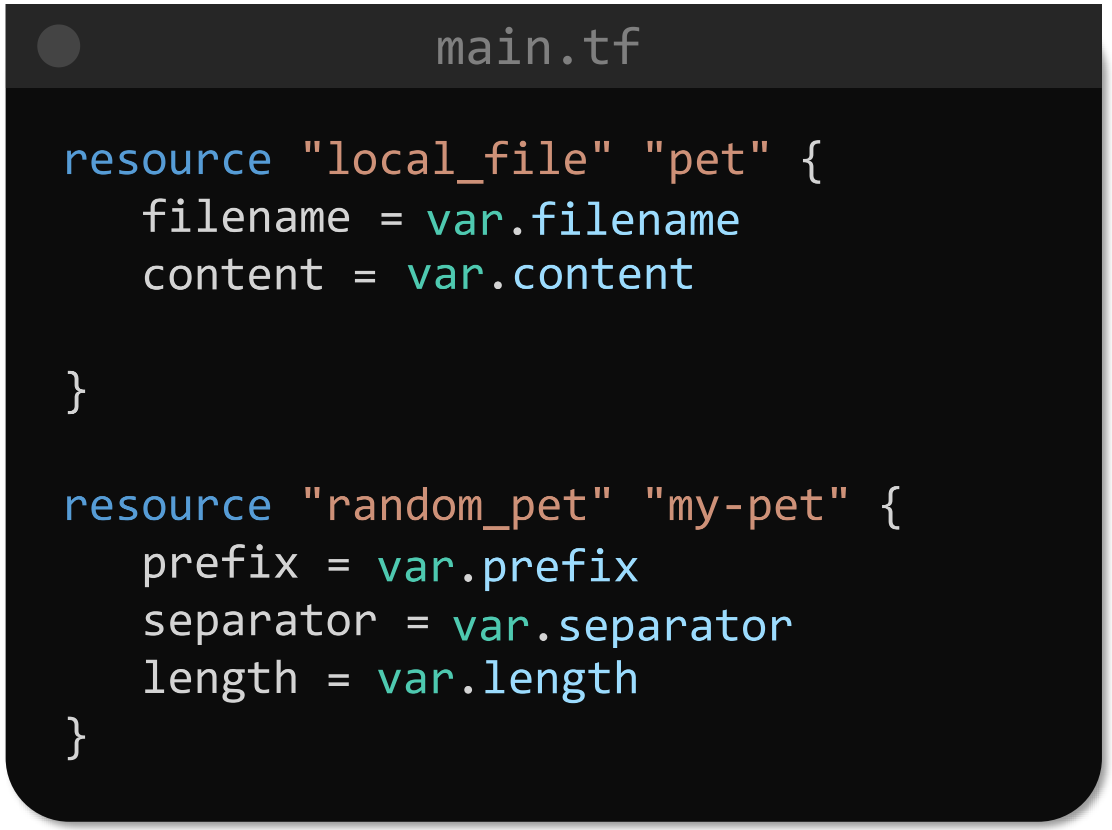
main.tf

```
resource "local_file" "pet" {
   filename = "/root/pets.txt"
   content = "We love pets!"

}

resource "random_pet" "my-pet" {
   prefix = "Mrs"
   separator = "."
   length = "1"
}
```

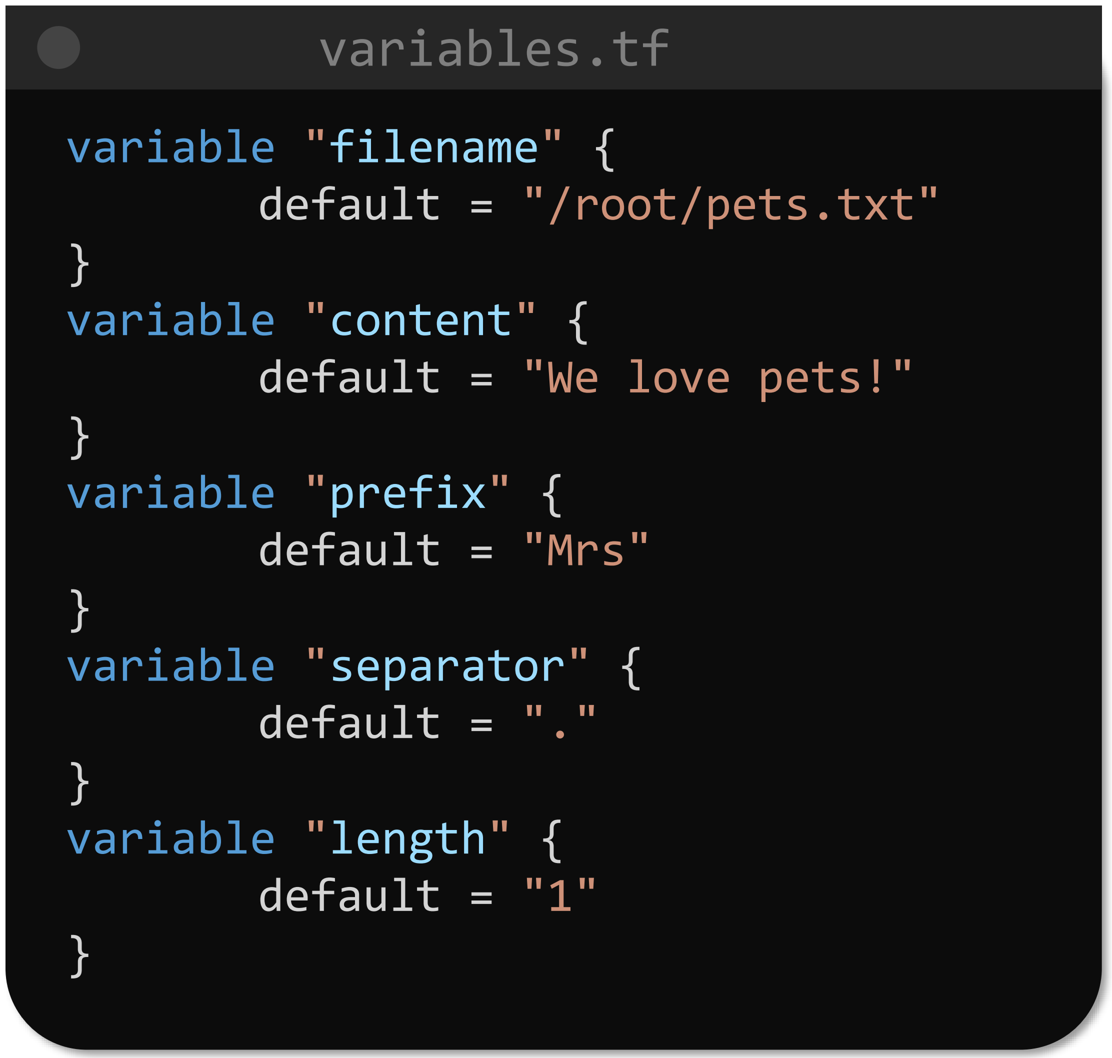


variables.tf

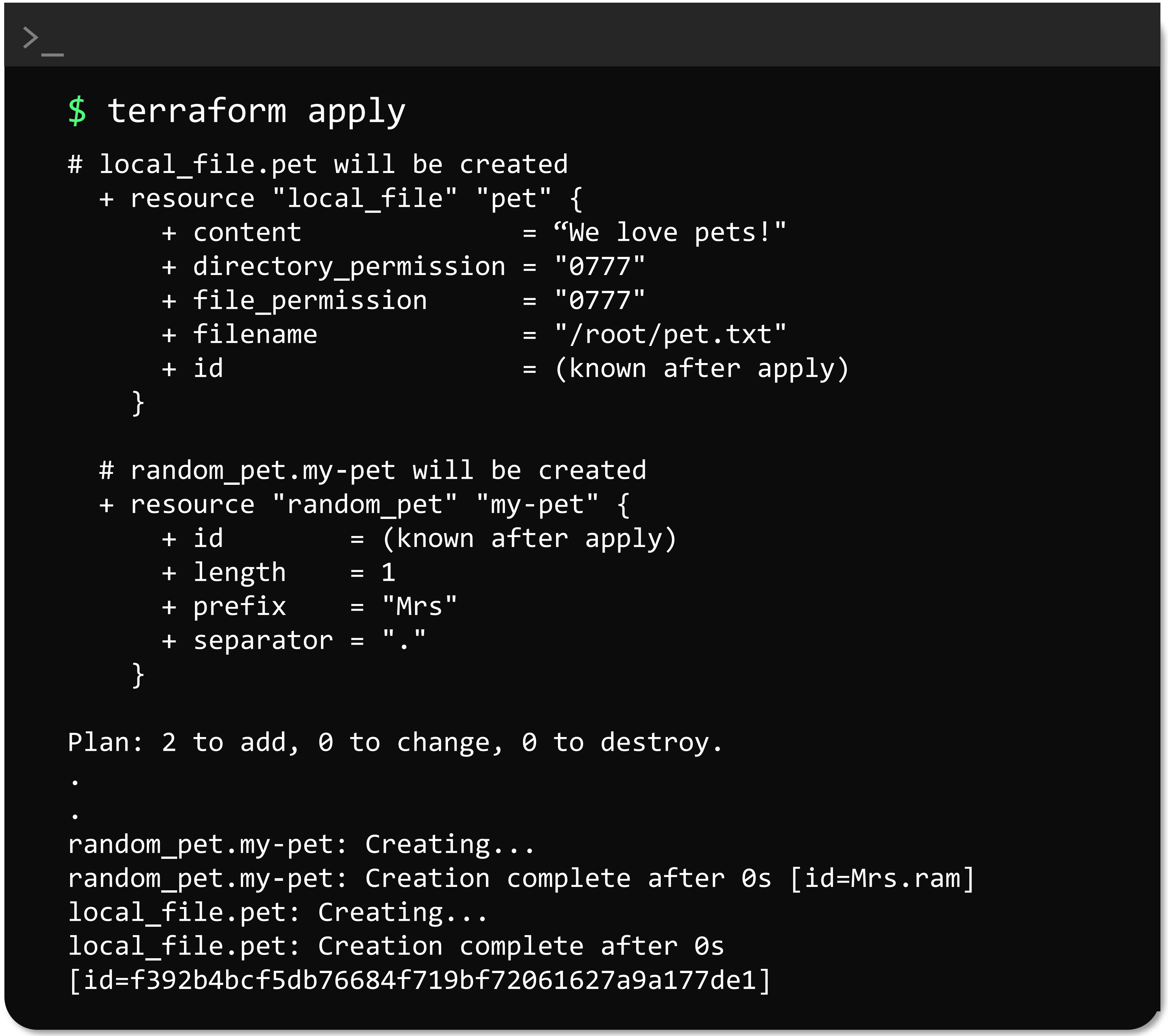
```
variable "filename" {
       default = "/root/pets.txt"
}
variable "content" {
       default = "We love pets!"
}
variable "prefix" {
       default = "Mrs"
}
variable "separator" {
       default = "."
}
variable "length" {
       default = "1"
}
```


main.tf

```
resource "local_file" "pet" {
   filename = var.filename
   content = var.content

}

resource "random_pet" "my-pet" {
   prefix = var.prefix
   separator = var.separator
   length = var.length
}
```


variables.tf

```
variable "filename" {
       default = "/root/pets.txt"
}
variable "content" {
       default = "We love pets!"
}
variable "prefix" {
       default = "Mrs"
}
variable "separator" {
       default = "."
}
variable "length" {
       default = "1"
}
```



```
$ terraform apply

# local_file.pet will be created
  + resource "local_file" "pet" {
      + content              = “We love pets!"
      + directory_permission = "0777"
      + file_permission      = "0777"
      + filename             = "/root/pet.txt"
      + id                   = (known after apply)
    }

  # random_pet.my-pet will be created
  + resource "random_pet" "my-pet" {
      + id        = (known after apply)
      + length    = 1
      + prefix    = "Mrs"
      + separator = "."
    }

Plan: 2 to add, 0 to change, 0 to destroy.
.
.
random_pet.my-pet: Creating...
random_pet.my-pet: Creation complete after 0s [id=Mrs.ram]
local_file.pet: Creating...
local_file.pet: Creation complete after 0s
[id=f392b4bcf5db76684f719bf72061627a9a177de1]
```

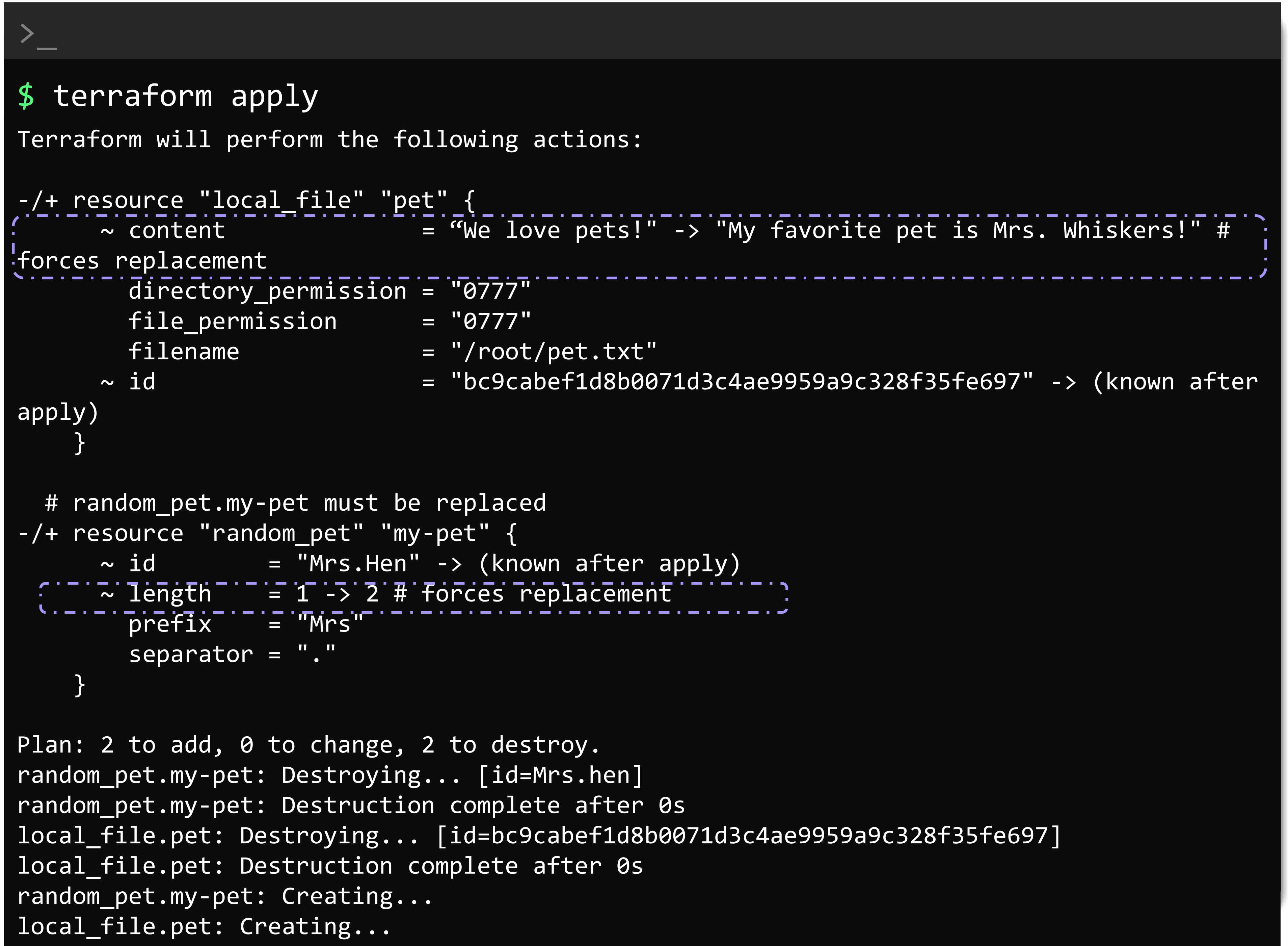
main.tf

```
resource "local_file" "pet" {
   filename = var.filename
   content = var.content

}

resource "random_pet" "my-pet" {
   prefix = var.prefix
   separator = var.separator
   length = var.length
}
```

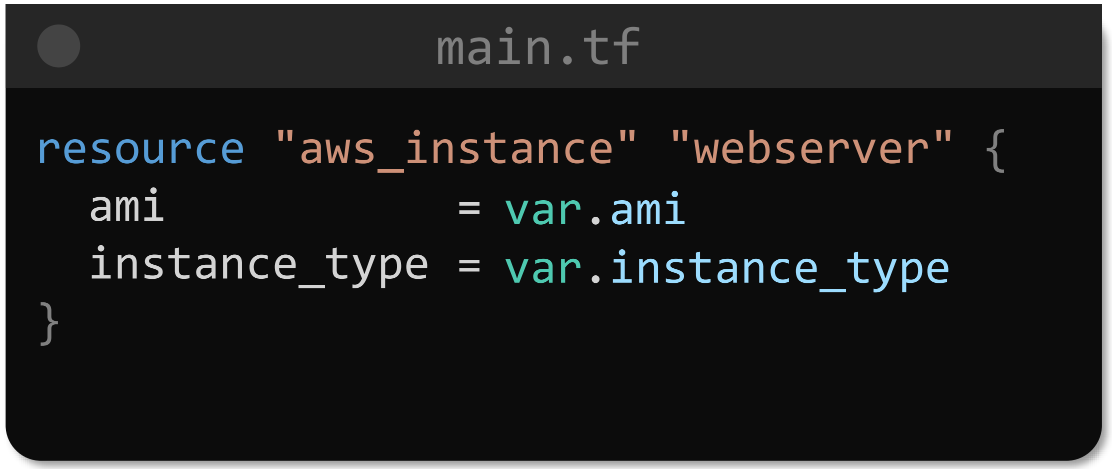
variables.tf

```
variable "filename" {
   default = "/root/pets.txt"
}
variable "content" {
    default = "My favorite pet is Mrs. Whiskers"
}
variable "prefix" {
    default = "Mrs"
}
variable "separator" {
    default = "."
}
variable "length" {
    default = "2"
}
```

KODEKLOUD

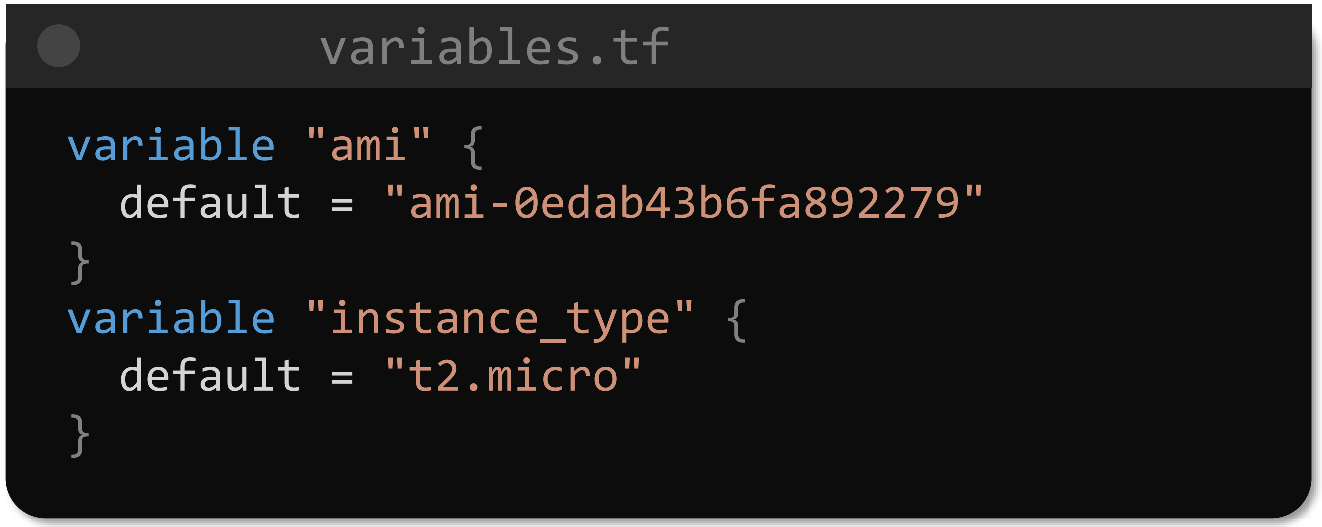


```
$ terraform apply
Terraform will perform the following actions:

-/+ resource "local_file" "pet" {
      ~ content              = “We love pets!" -> "My favorite pet is Mrs. Whiskers!" #
forces replacement
        directory_permission = "0777"
        file_permission      = "0777"
        filename             = "/root/pet.txt"
      ~ id                   = "bc9cabef1d8b0071d3c4ae9959a9c328f35fe697" -> (known after
apply)
    }

  # random_pet.my-pet must be replaced
-/+ resource "random_pet" "my-pet" {
      ~ id        = "Mrs.Hen" -> (known after apply)
      ~ length    = 1 -> 2 # forces replacement
        prefix    = "Mrs"
        separator = "."
    }

Plan: 2 to add, 0 to change, 2 to destroy.
random_pet.my-pet: Destroying... [id=Mrs.hen]
random_pet.my-pet: Destruction complete after 0s
local_file.pet: Destroying... [id=bc9cabef1d8b0071d3c4ae9959a9c328f35fe697]
local_file.pet: Destruction complete after 0s
random_pet.my-pet: Creating...
local_file.pet: Creating...
```



main.tf

```
resource "aws_instance" "webserver" {
  ami           = var.ami
  instance_type = var.instance_type
}
```



variables.tf

```
variable "ami" {
  default = "ami-0edab43b6fa892279"
}
variable "instance_type" {
  default = "t2.micro"
}
```

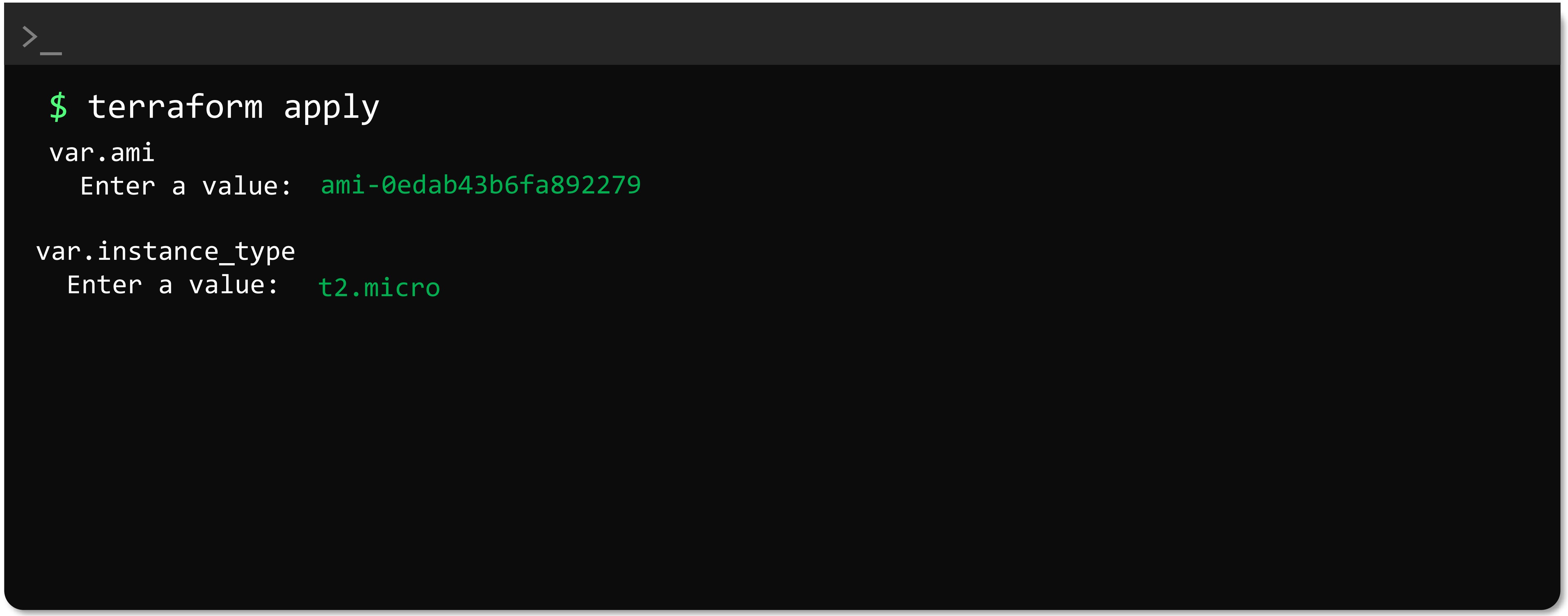
main.tf

```
resource "aws_instance" "webserver" {
  ami           = var.ami
  instance_type = var.instance_type
}
```

variables.tf

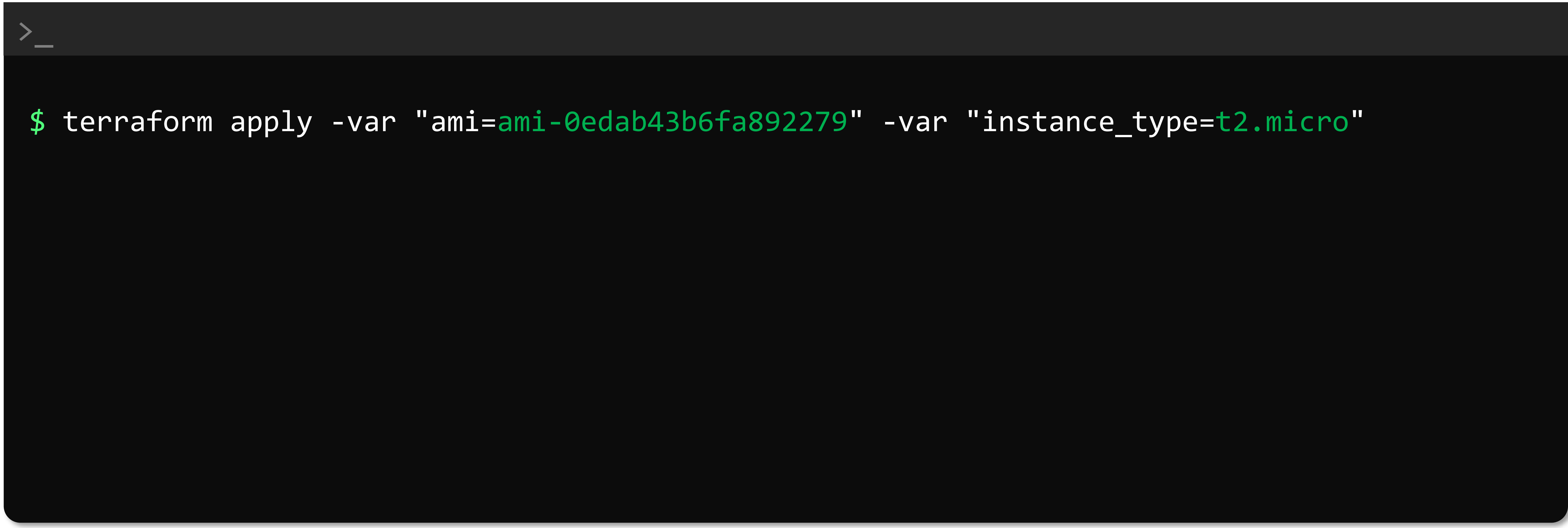
```
variable "ami" {

}
variable "instance_type" {

}
```

Interactive Mode



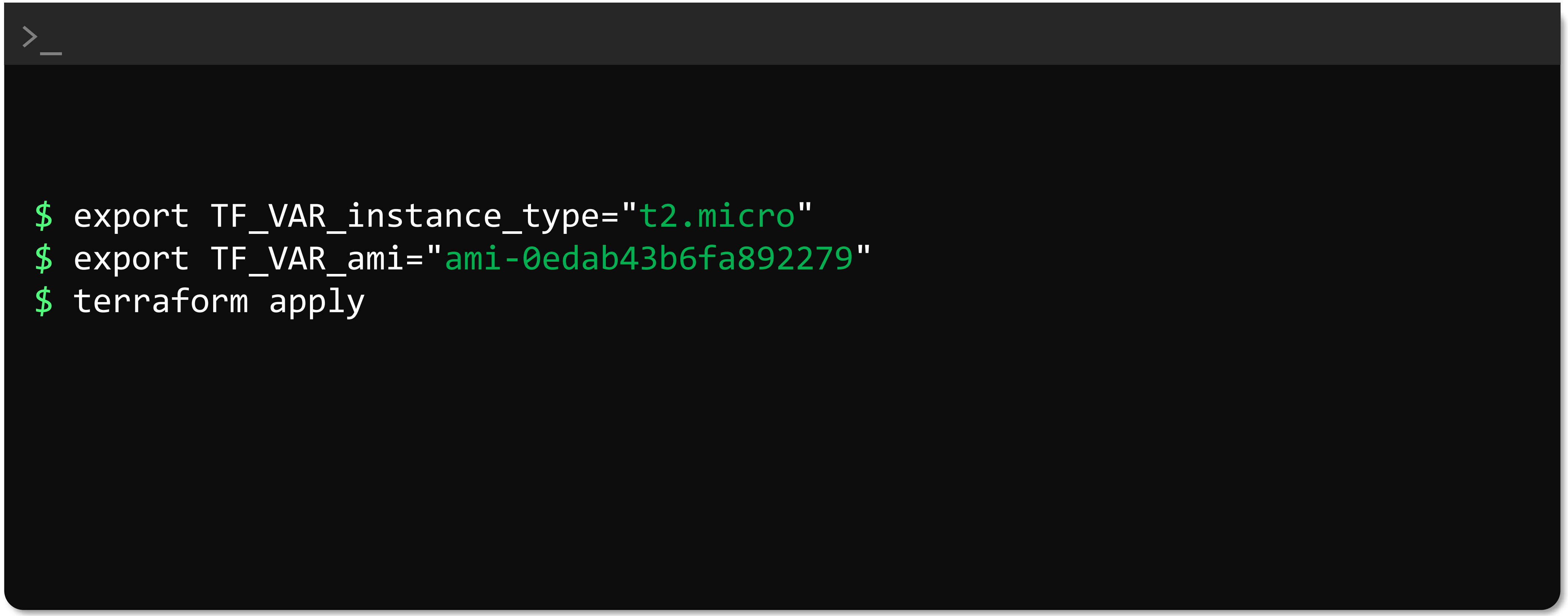
```
$ terraform apply
var.ami
  Enter a value:  ami-0edab43b6fa892279

var.instance_type
  Enter a value:   t2.micro
```

Command Line Flags



```
$ terraform apply -var "ami=ami-0edab43b6fa892279" -var "instance_type=t2.micro"
```

Environment Variables



```
$ export TF_VAR_instance_type="t2.micro"
$ export TF_VAR_ami="ami-0edab43b6fa892279"
$ terraform apply
```

Variable Definition Files

variable.tfvars

```
ami="ami-0edab43b6fa892279"
instance_type="t2.micro"
```

```
$ terraform apply -var-file variable.tfvars
```

Automatically Loaded

terraform.tfvars	terraform.tfvars.json
*.auto.tfvars	*.auto.tfvars.json

Variable Definition Precedence

Order	Option
1	Environment Variables
2	terraform.tfvars
3	*.auto.tfvars (alphabetical order)
4	-var or –var-file (command-line flags)

```
>_
$ export TF_VAR_type= "t2.nano"
```
1

```
terraform.tfvars
type =  "t3.micro"
```
2

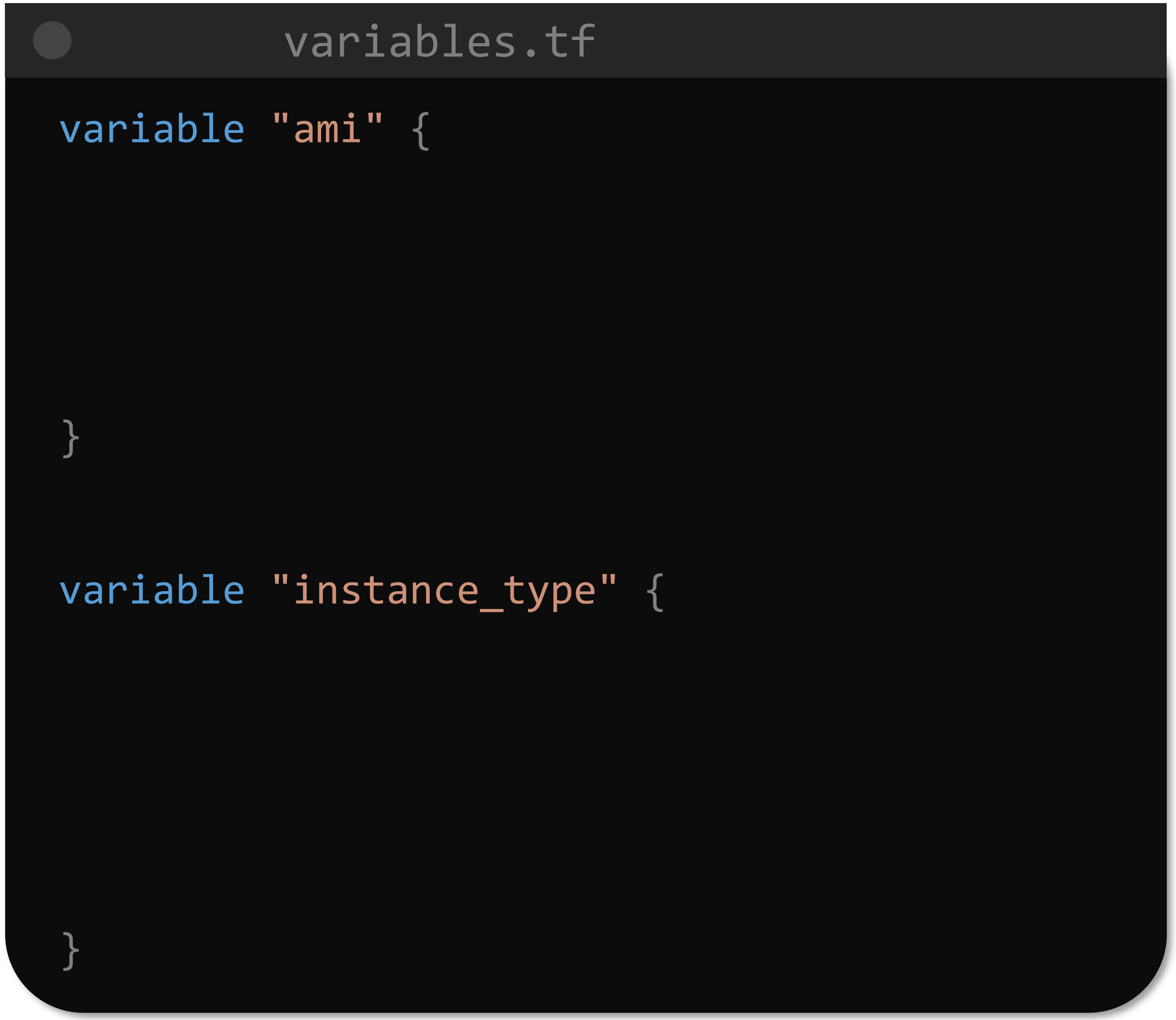
```
variable.auto.tfvars
type =  "t3.small"
```
3

```
>_
$ terraform apply -var "type=t2.medium"
```
4

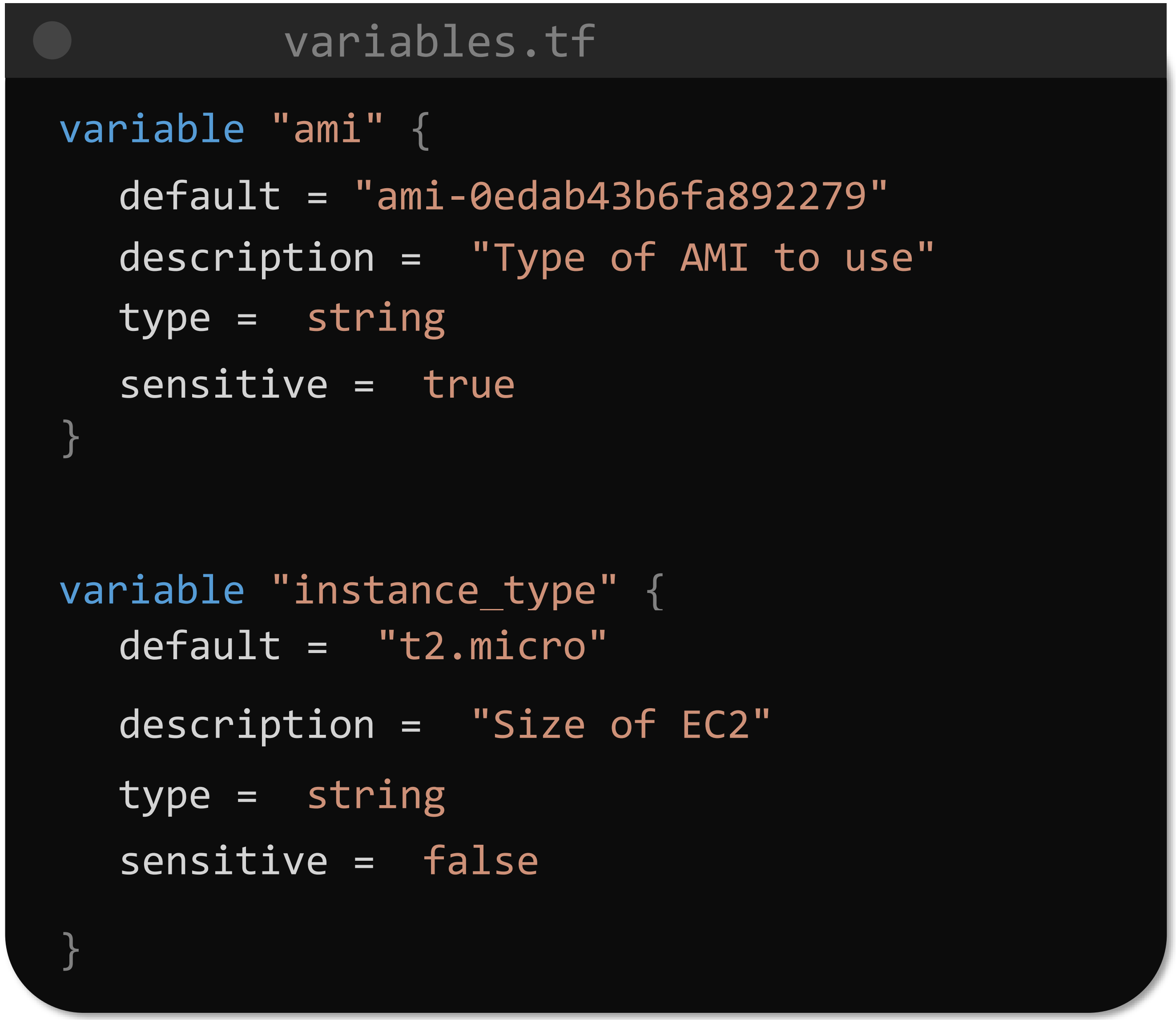
KODEKLOUD

KODEKLOUD

Understanding the Variable Block

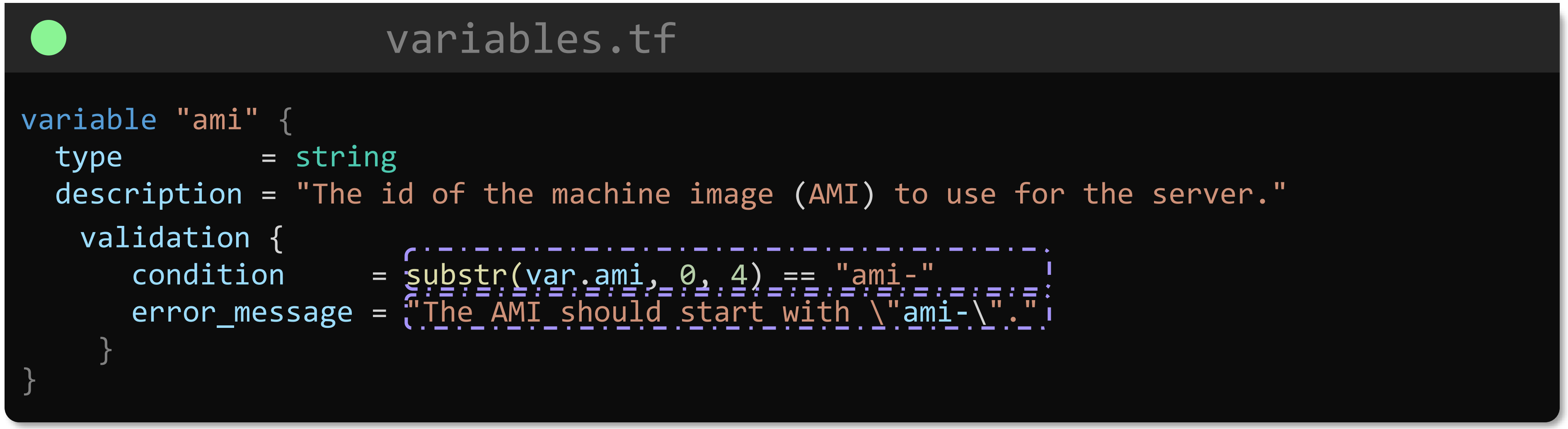

variables.tf

```
variable "ami" {





}


variable "instance_type" {






}
```

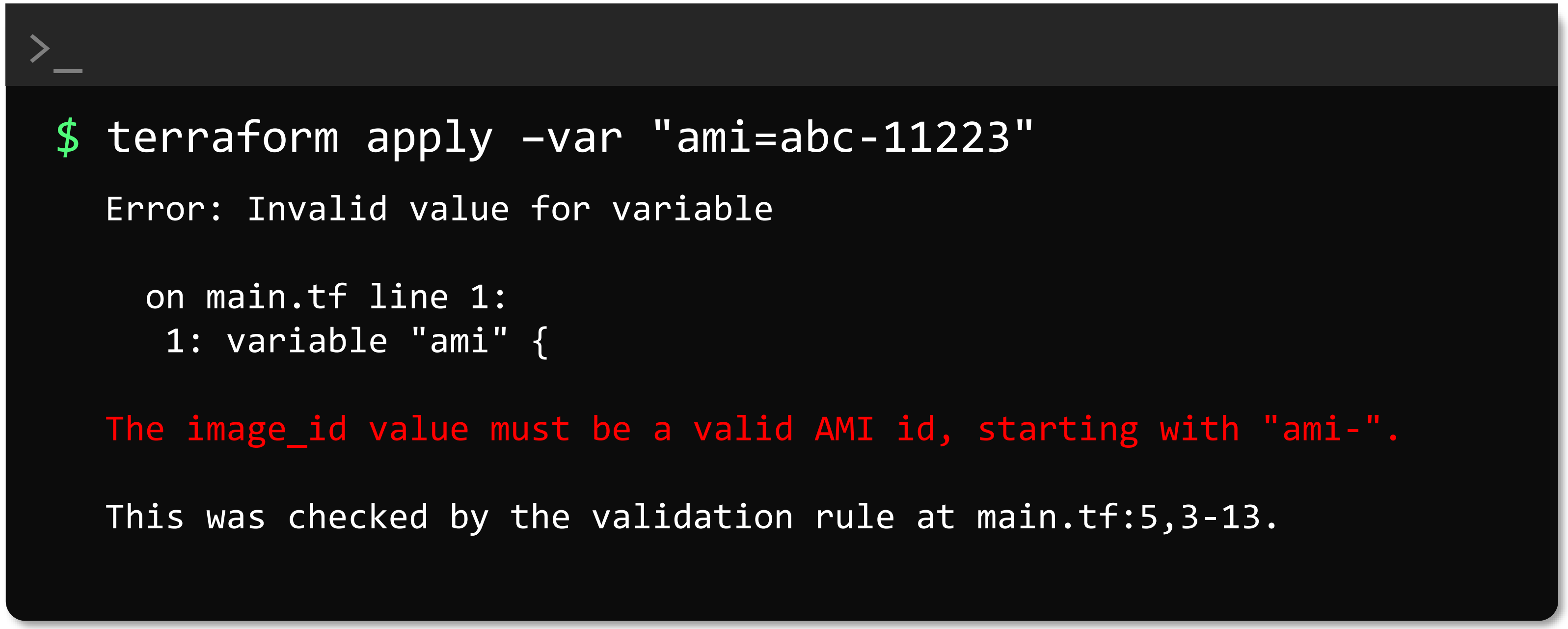


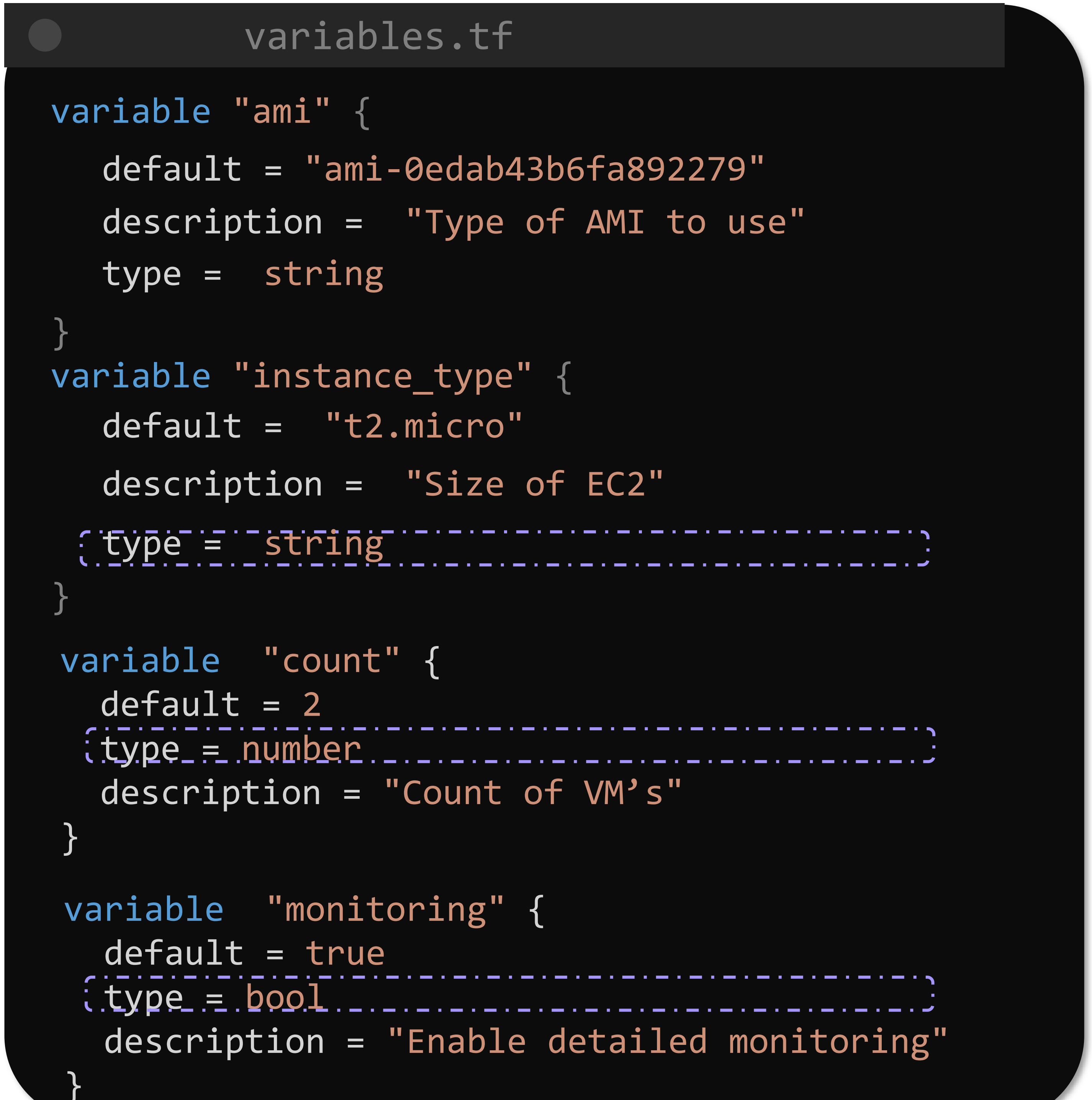
variables.tf

```
variable "ami" {
   default = "ami-0edab43b6fa892279"
   description =  "Type of AMI to use"
   type =  string
   sensitive =  true
}


variable "instance_type" {
   default =  "t2.micro"

   description =  "Size of EC2"

   type =  string
   sensitive =  false

}
```


variables.tf

```
variable "ami" {
  type        = string
  description = "The id of the machine image (AMI) to use for the server."
   validation {
      condition     = substr(var.ami, 0, 4) == "ami-"
      error_message = "The AMI should start with \"ami-\"."
    }
}
```


```
$ terraform apply -var "ami=abc-11223"
  Error: Invalid value for variable

    on main.tf line 1:
     1: variable "ami" {

  The image_id value must be a valid AMI id, starting with "ami-".

  This was checked by the validation rule at main.tf:5,3-13.
```



variables.tf

```
variable "ami" {
   default = "ami-0edab43b6fa892279"
   description =  "Type of AMI to use"
   type =  string
}
variable "instance_type" {
   default =  "t2.micro"
   description =  "Size of EC2"
   type =  string
}

 variable  "count" {
   default = 2
   type = number
   description = "Count of VM’s"
 }

 variable  "monitoring" {
   default = true
   type = bool
   description = "Enable detailed monitoring"
 }
```

Type	Example
string	"/root/pets.txt"
number	1
bool	true/false
any	Default Value

Type	Example
string	"t2.micro"
number	2
bool	true/false
any	Default Value
list	["web1", “web2"]

map	region1 = us-east-1
	region2 = us-west-2

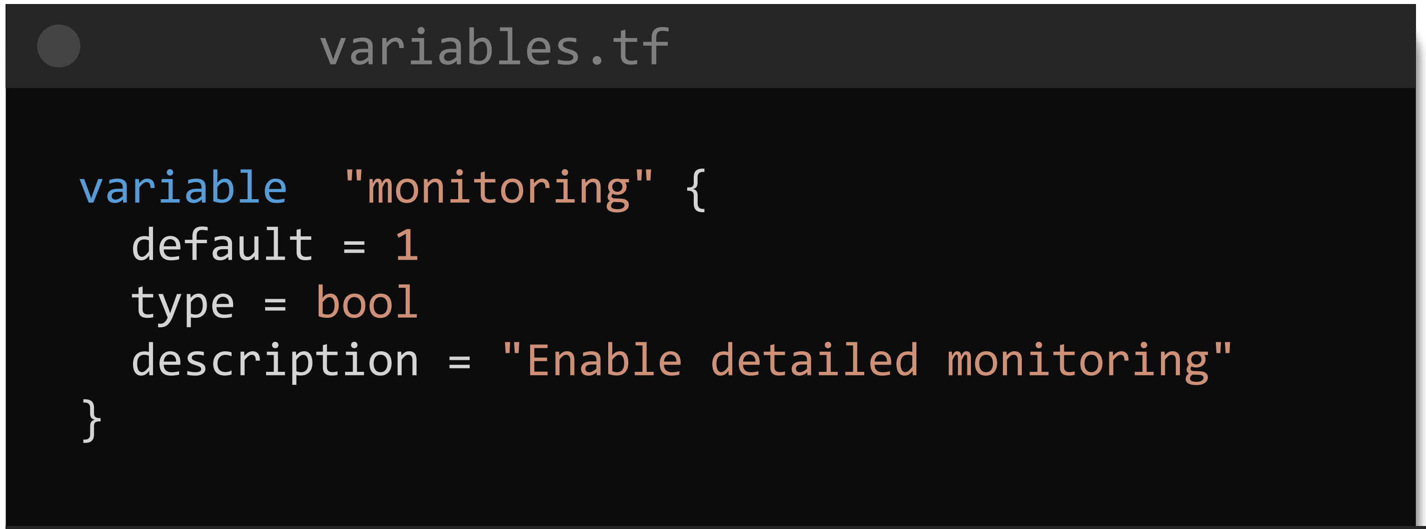
object	Complex Data Structure
tuple	Complex Data Structure

variables.tf

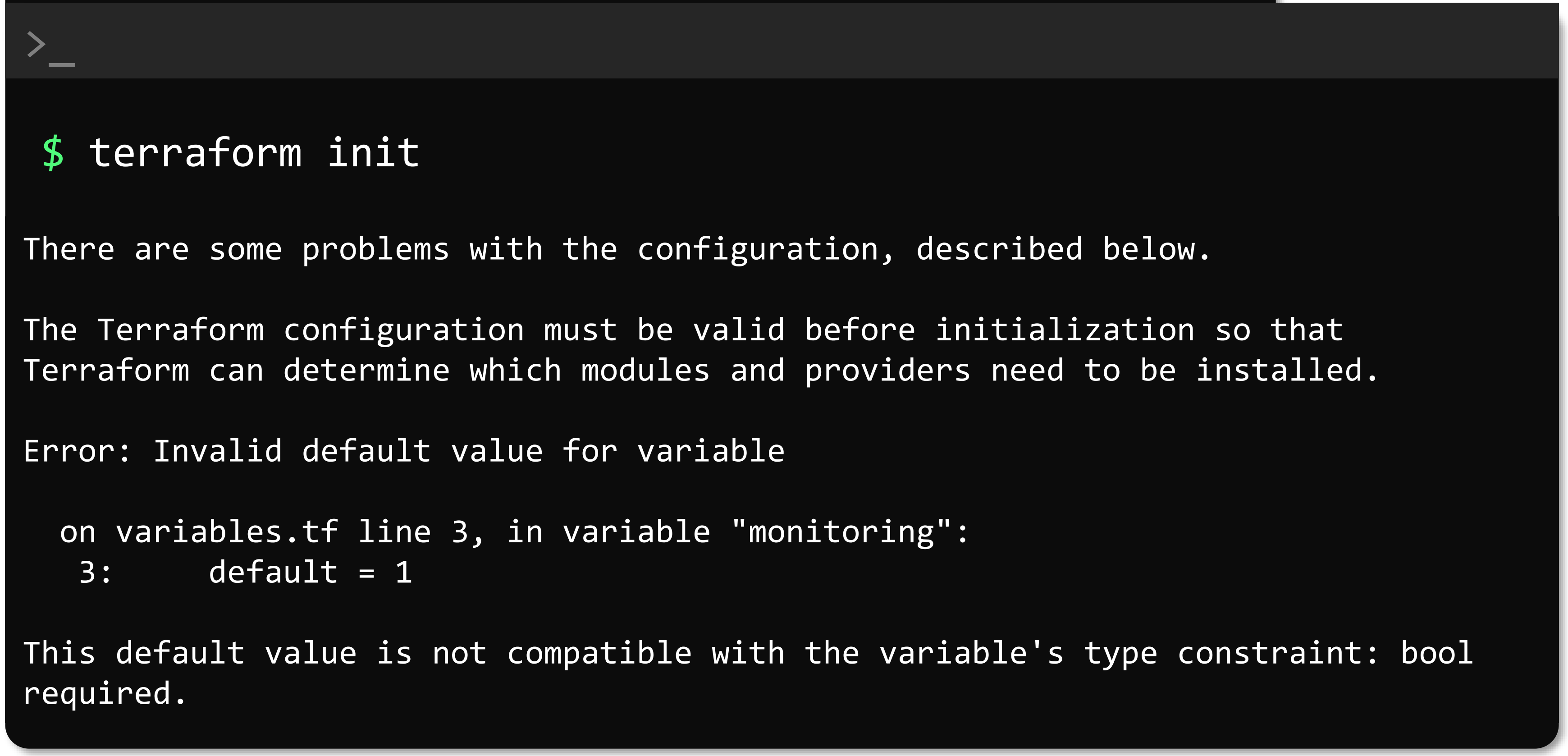
```
variable  "count" {
  default = 2

  type = number
  description = "Count of VM’s"
}

 variable  "monitoring" {
   default = true

   type = bool
   description = "Enable detailed monitoring"
 }
```

variables.tf

```
variable  "count" {
  default =  "2"

  type = number
  description = "Count of VM’s"
}

 variable  "monitoring" {
   default = "true"

   type = bool
   description = "Enable detailed monitoring"
 }
```


variables.tf

```
variable  "monitoring" {
  default = 1
  type = bool
  description = "Enable detailed monitoring"
}
```



```
$ terraform init

There are some problems with the configuration, described below.

The Terraform configuration must be valid before initialization so that
Terraform can determine which modules and providers need to be installed.

Error: Invalid default value for variable

  on variables.tf line 3, in variable "monitoring":
   3:     default = 1

This default value is not compatible with the variable's type constraint: bool
required.
```

List

variables.tf

```
variable  "servers" {
  default = ["web1", "web2", "web3"]
  type = list  0         1         2
}
```

maint.tf

```
resource "aws_instance" "web" {
    ami = var.ami
    instance_type =  var.instance_type
    tags = {
        name =  var.servers[0]
    }
}
```

Index	Value
0	web1
1	web2
2	web3

Map

variables.tf

```
variable instance_type {
  type          = map
  default       = {
       "production" =  "m5.large"
       "development"  =  "t2.micro"
  }
 }
```

maint.tf

```
resource "aws_instance"  "prodcution" {
    ami = var.ami
    instance_type  var.instance_type["development"]
    tags = {
        name =  var.servers[0]
    }
}
```

Key	Value
production	m5.large
development	t2.micro

KODEKLOUD

List of a Type

variables.tf

```
variable  "servers" {
  default = ["web1", "web2", "web3"]
  type = list(string)
}
```

variables.tf

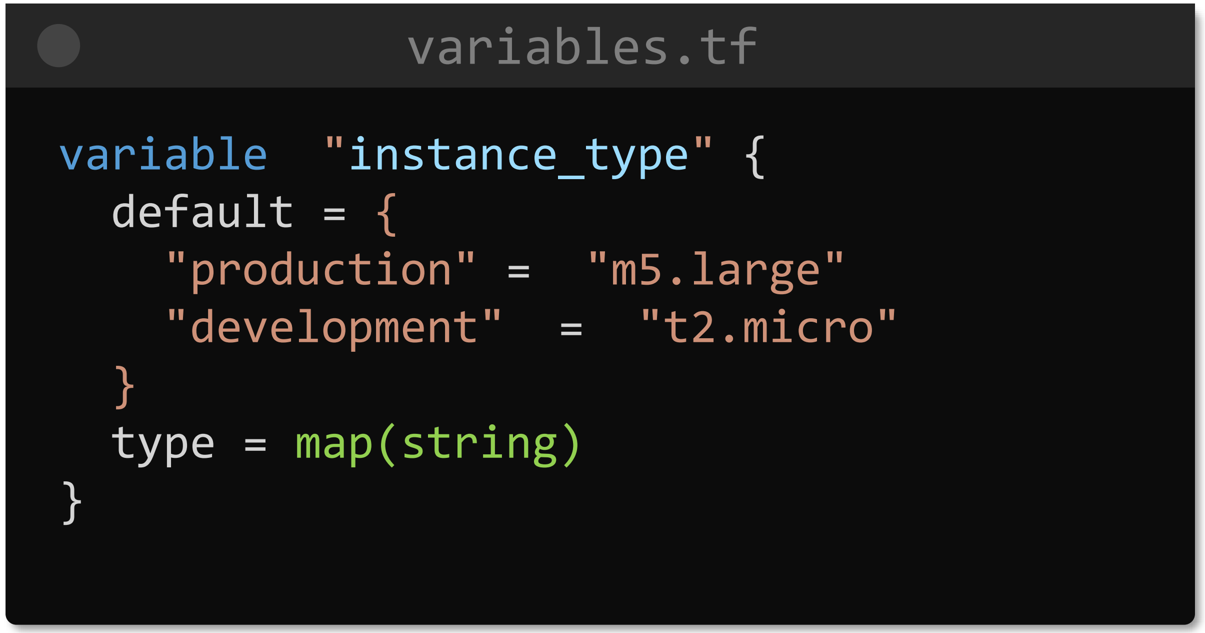
```
variable "servers" {
  default = ["web1", "web2", "web3"]
  type = list(number)
}
```

variables.tf

```
variable "prefix" {
  default = [1, 2, 3]
  type = list(number)
}
```

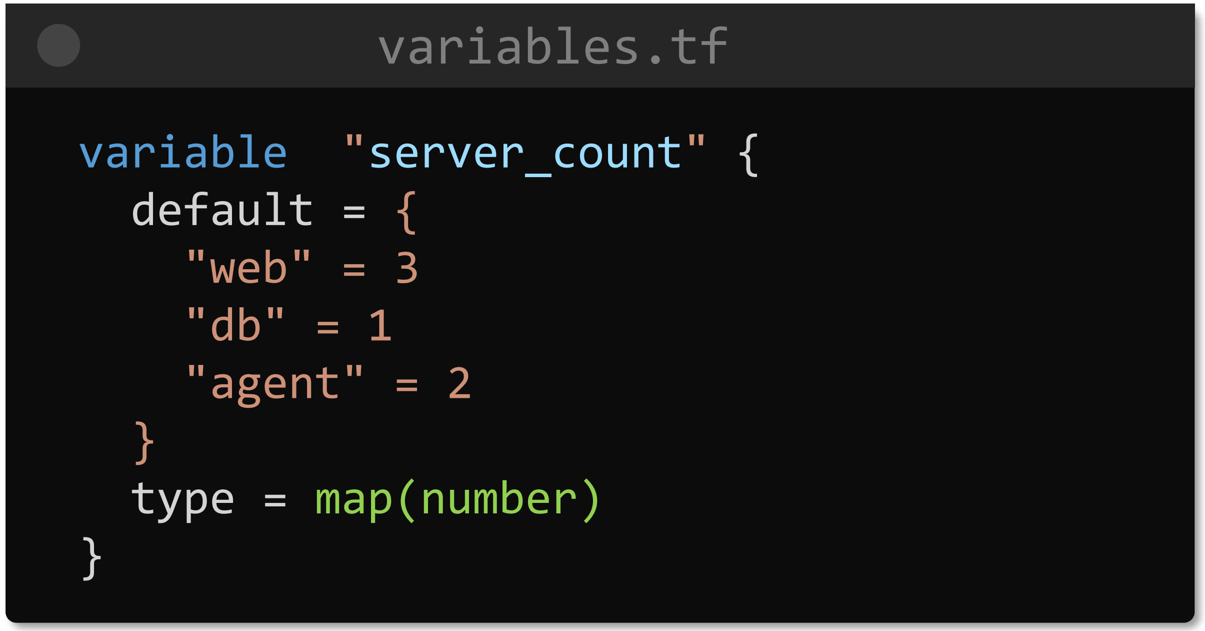
```
>_
$ terraform plan
Error: Invalid default value for variable

  on variables.tf line 3, in variable "prefix":
   3:    default     = ["Mr", "Mrs", "Sir"]

This default value is not compatible with the
variable's type constraint: a number is required.
```

Map of a Type



variables.tf

```
variable  "instance_type" {
  default = {
    "production" =  "m5.large"
    "development"  =  "t2.micro"
  }
  type = map(string)
}
```



variables.tf

```
variable  "server_count" {
  default = {
    "web" = 3
    "db" = 1
    "agent" = 2
  }
  type = map(number)
}
```

Set

variables.tf
```
variable  "servers" {
  default = ["web1", "web2", "web3"]
  type = set(string)
}
```

variables.tf
```
variable "prefix" {
  default =  ["web1", "web2", "web2"]
  type = set(string)
}
```

variables.tf
```
variable "db" {
  default = ["db1", "db2"]
  type = set(string)
}
```

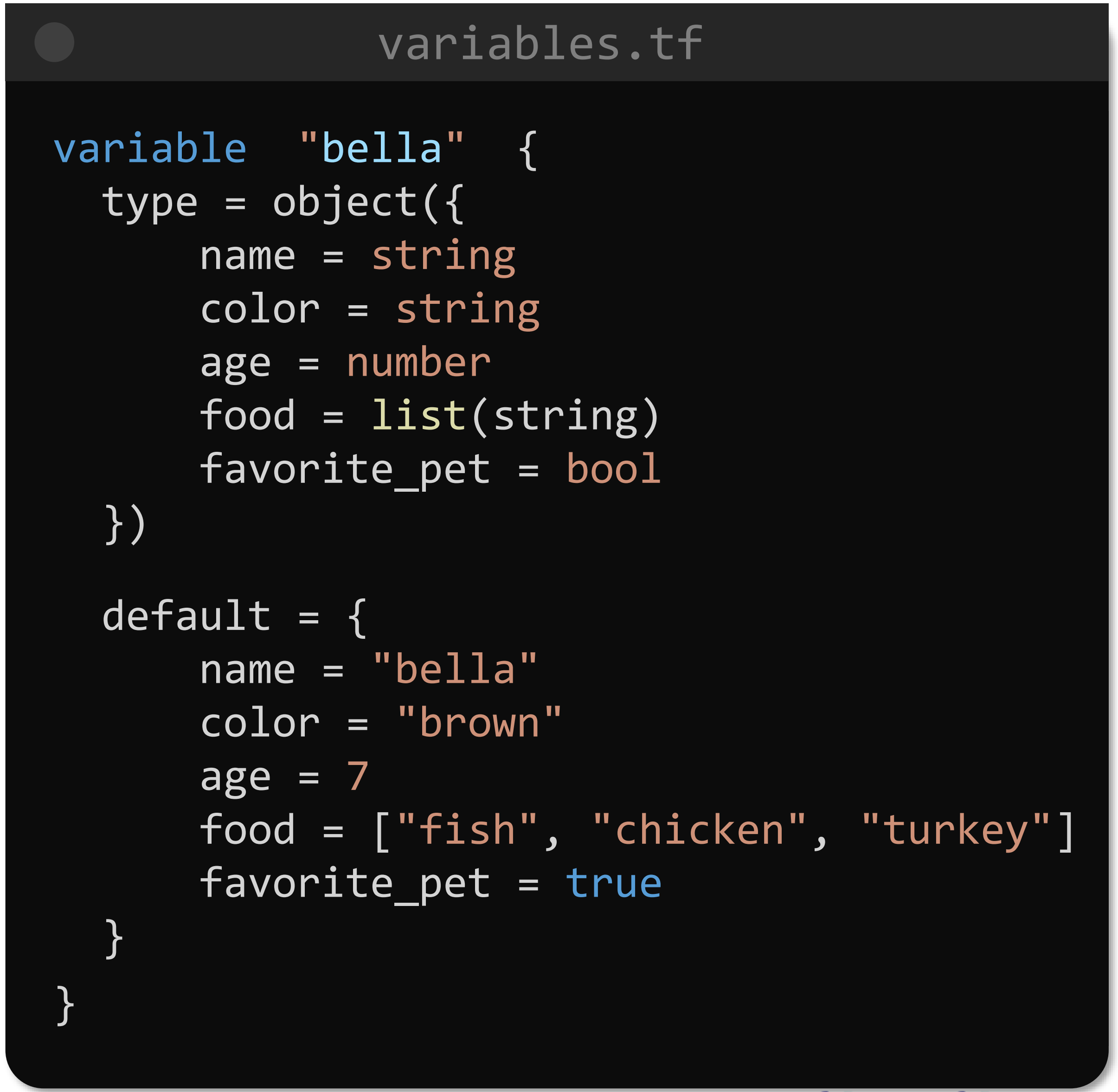
variables.tf
```
variable "db"  {
  default =  ["db1", "db2", "db1" ]
  type = set(string)
}
```

variables.tf
```
variable  "count" {
  default = [10, 12, 15]
  type = set(number)
}
```

variables.tf
```
variable "count"  {
  default = [10, 12, 15, 10]
  type = set(number)
}
```

Objects

Key	Example	Type
name	bella	string
color	brown	string
age	7	number
food	["fish", "chicken", "turkey"]	list
favorite_pet	true	bool



variables.tf

```
variable  "bella"  {
  type = object({
      name = string
      color = string
      age = number
      food = list(string)
      favorite_pet = bool
  })

  default = {
      name = "bella"
      color = "brown"
      age = 7
      food = ["fish", "chicken", "turkey"]
      favorite_pet = true
  }
}
```

Tuples

variables.tf

```
variable web {
  type         = tuple([string, number, bool])
  default      = ["web1", 3, true]

}
```

variables.tf

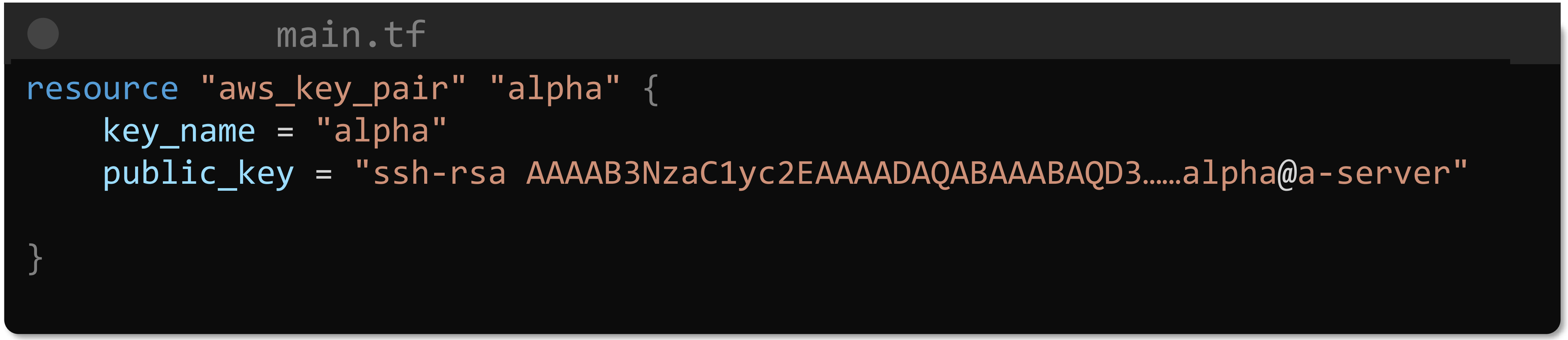
```
variable db {
  type         = tuple([string, number, bool])
  default      = ["db1", 1, true, "db2"]

}
```

```
>_
$ terraform plan
  Error: Invalid default value for variable

    on variables.tf line 3, in variable "db":
     3:   default      = ["db1", 1, true, "db2"]

  This default value is not compatible with the
  variable's type constraint:
  tuple required.
```

KODEKLOUD

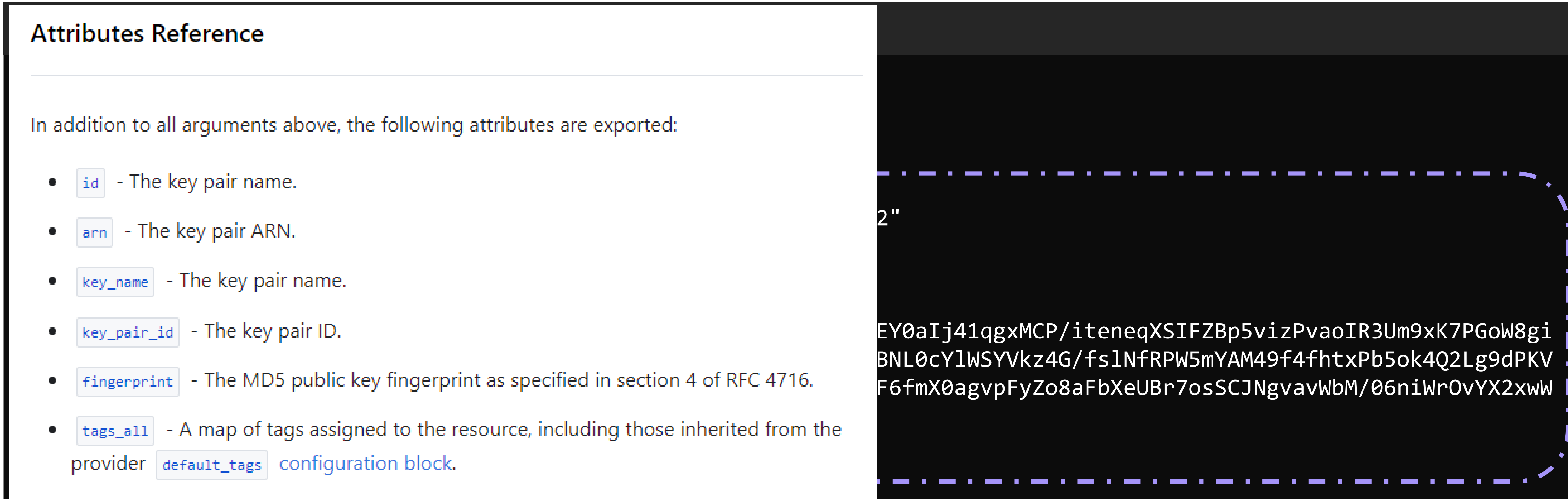
Resource Attributes and Dependencies

https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/key_pair#attributes-reference



main.tf

```
resource "aws_key_pair" "alpha" {
    key_name = "alpha"
    public_key = "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQD3……alpha@a-server"

}
```

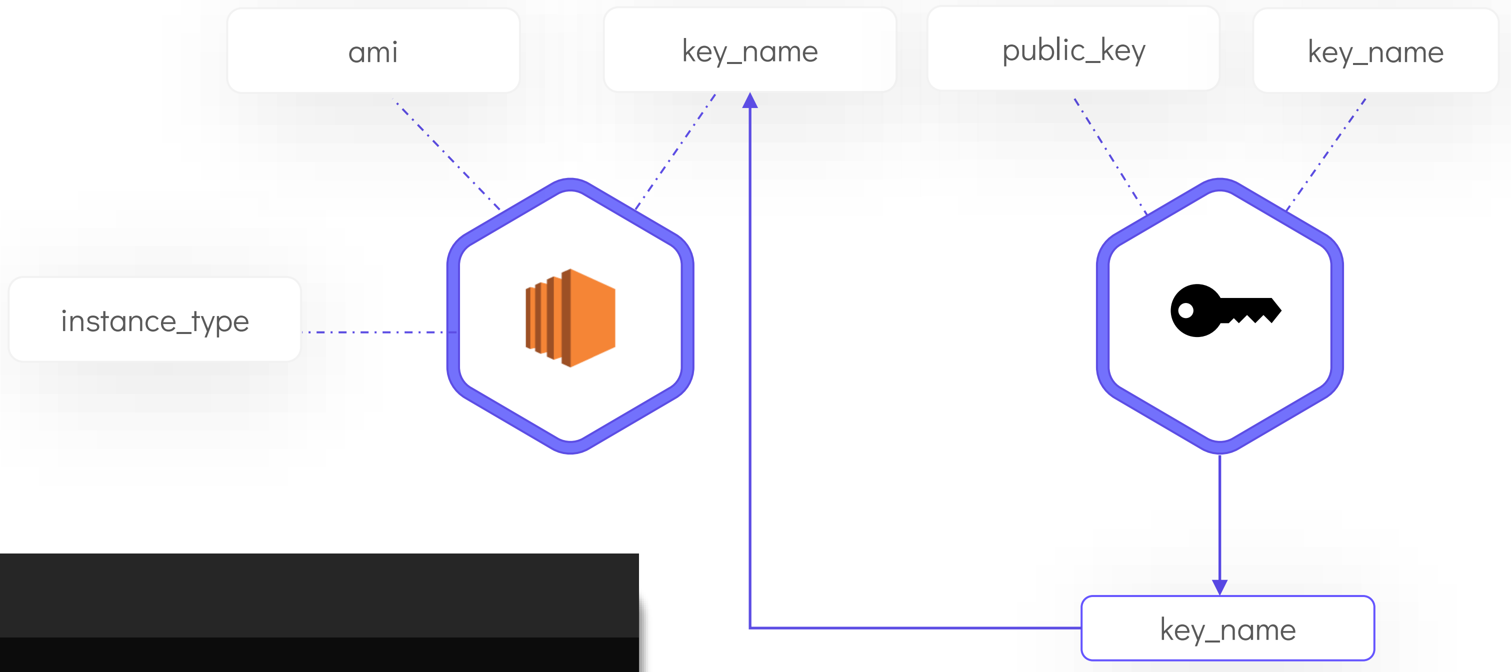


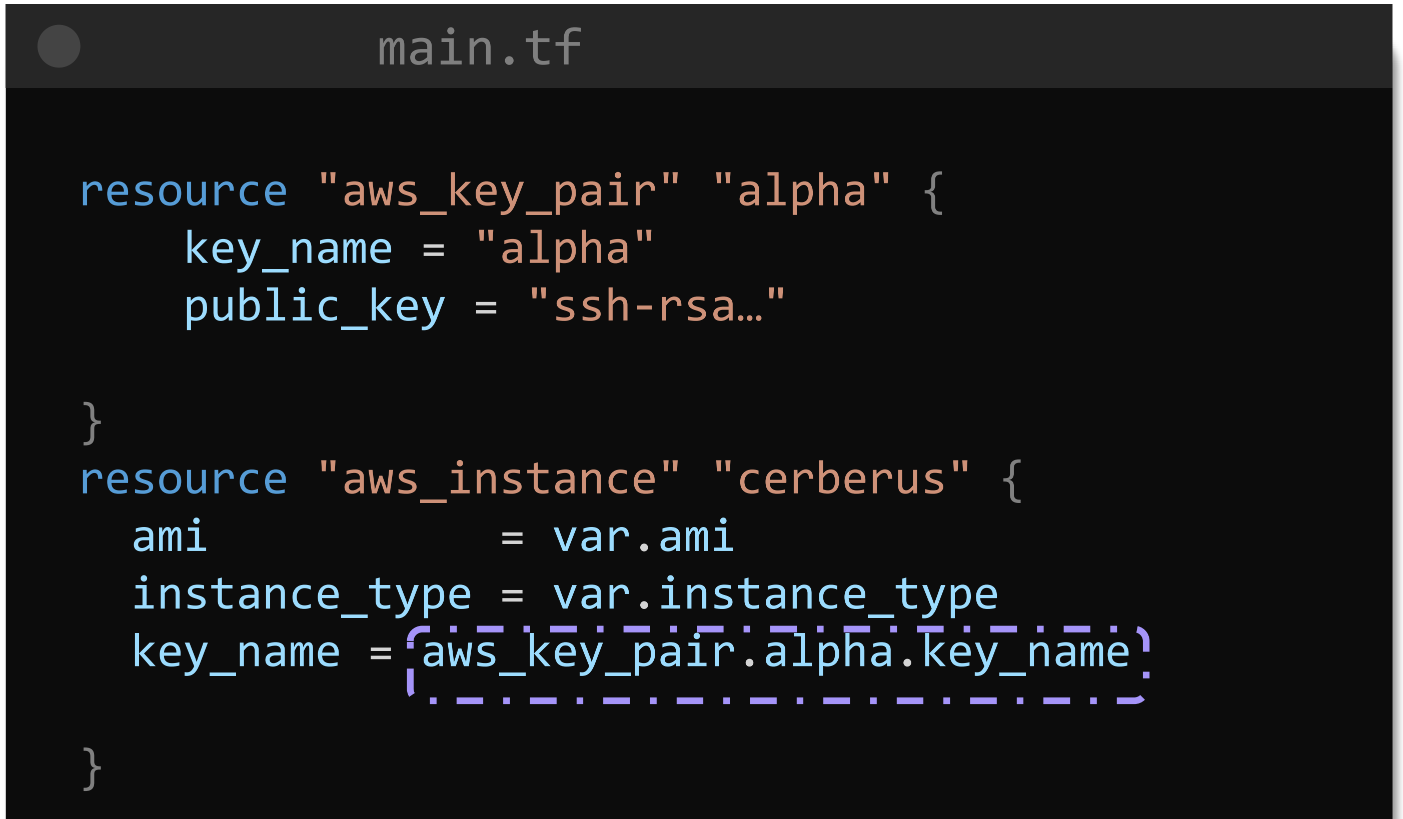
Attributes Reference

In addition to all arguments above, the following attributes are exported:

- `id` - The key pair name.
- `arn` - The key pair ARN.
- `key_name` - The key pair name.
- `key_pair_id` - The key pair ID.
- `fingerprint` - The MD5 public key fingerprint as specified in section 4 of RFC 4716.
- `tags_all` - A map of tags assigned to the resource, including those inherited from the provider `default_tags` configuration block.

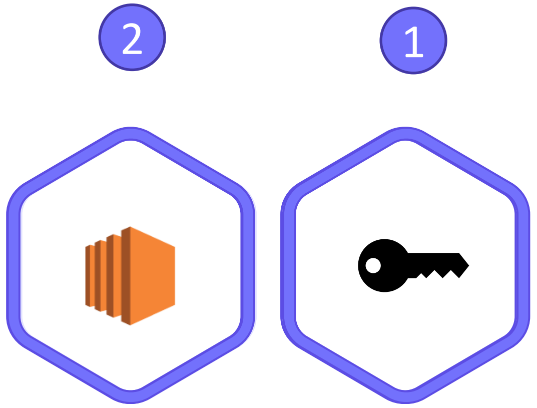
```
2"

EY0aIj41qgxMCP/iteneqXSIFZBp5vizPvaoIR3Um9xK7PGoW8gi
BNL0cYlWSYVkz4G/fslNfRPW5mYAM49f4fhtxPb5ok4Q2Lg9dPKV
F6fmX0agvpFyZo8aFbXeUBr7osSCJNgvavWbM/06niWrOvYX2xwW
```

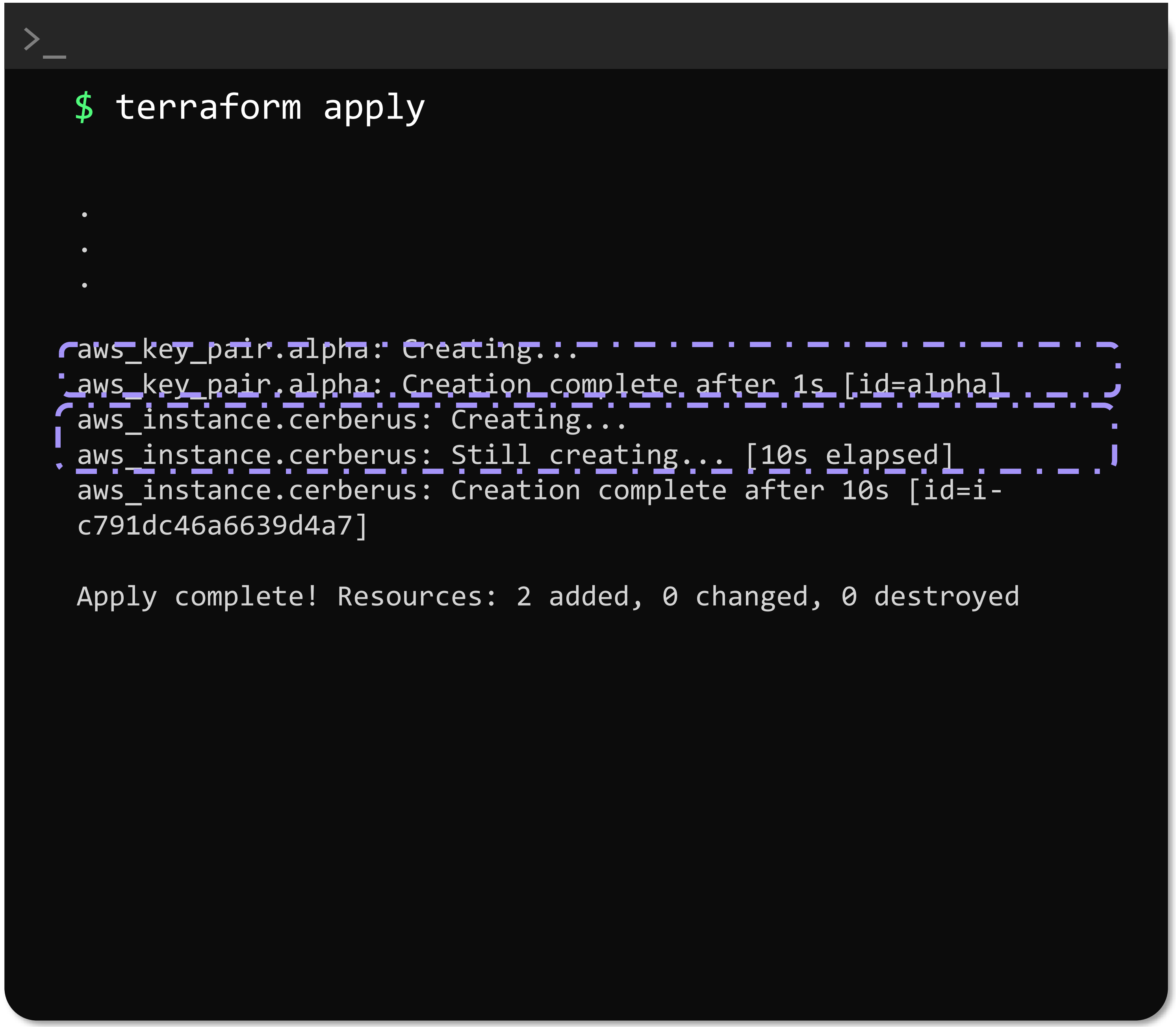



main.tf

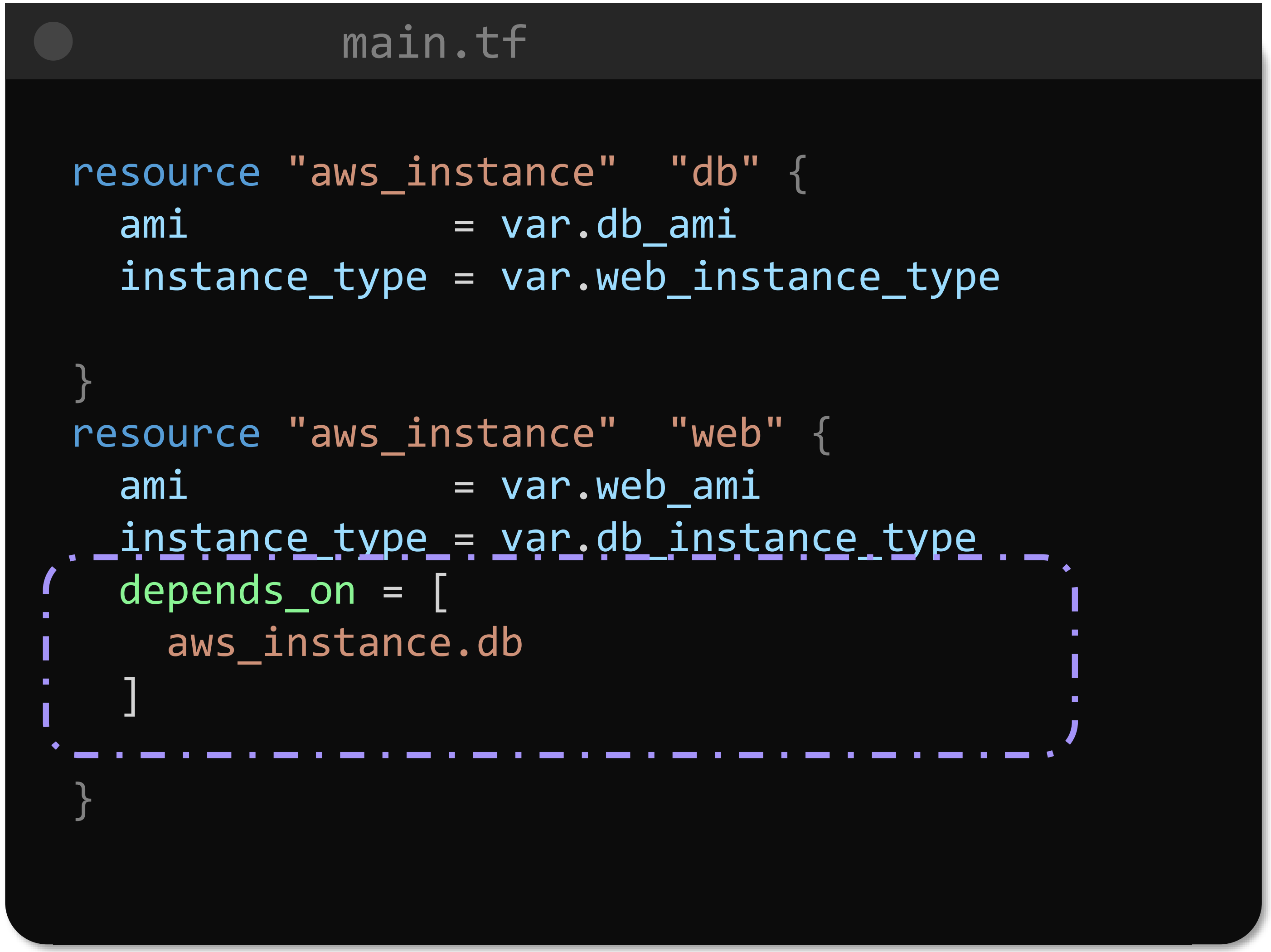
```
resource "aws_key_pair" "alpha" {
    key_name = "alpha"
    public_key = "ssh-rsa…"

}
resource "aws_instance" "cerberus" {
  ami           = var.ami
  instance_type = var.instance_type
  key_name  = aws_key_pair.alpha.key_name

}
```

<RESOURCE TYPE>.<RESOURCE_NAME>.<ATTRIBUTE>



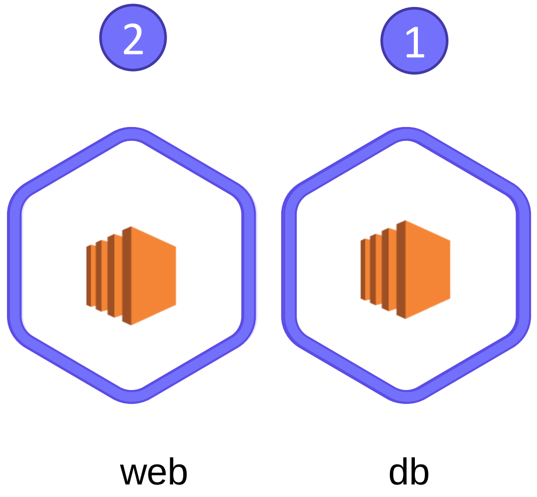


```
$ terraform apply

 .
 .
 .

aws_key_pair.alpha: Creating...
aws_key_pair.alpha: Creation complete after 1s [id=alpha]
aws_instance.cerberus: Creating...
aws_instance.cerberus: Still creating... [10s elapsed]
aws_instance.cerberus: Creation complete after 10s [id=i-
c791dc46a6639d4a7]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed
```



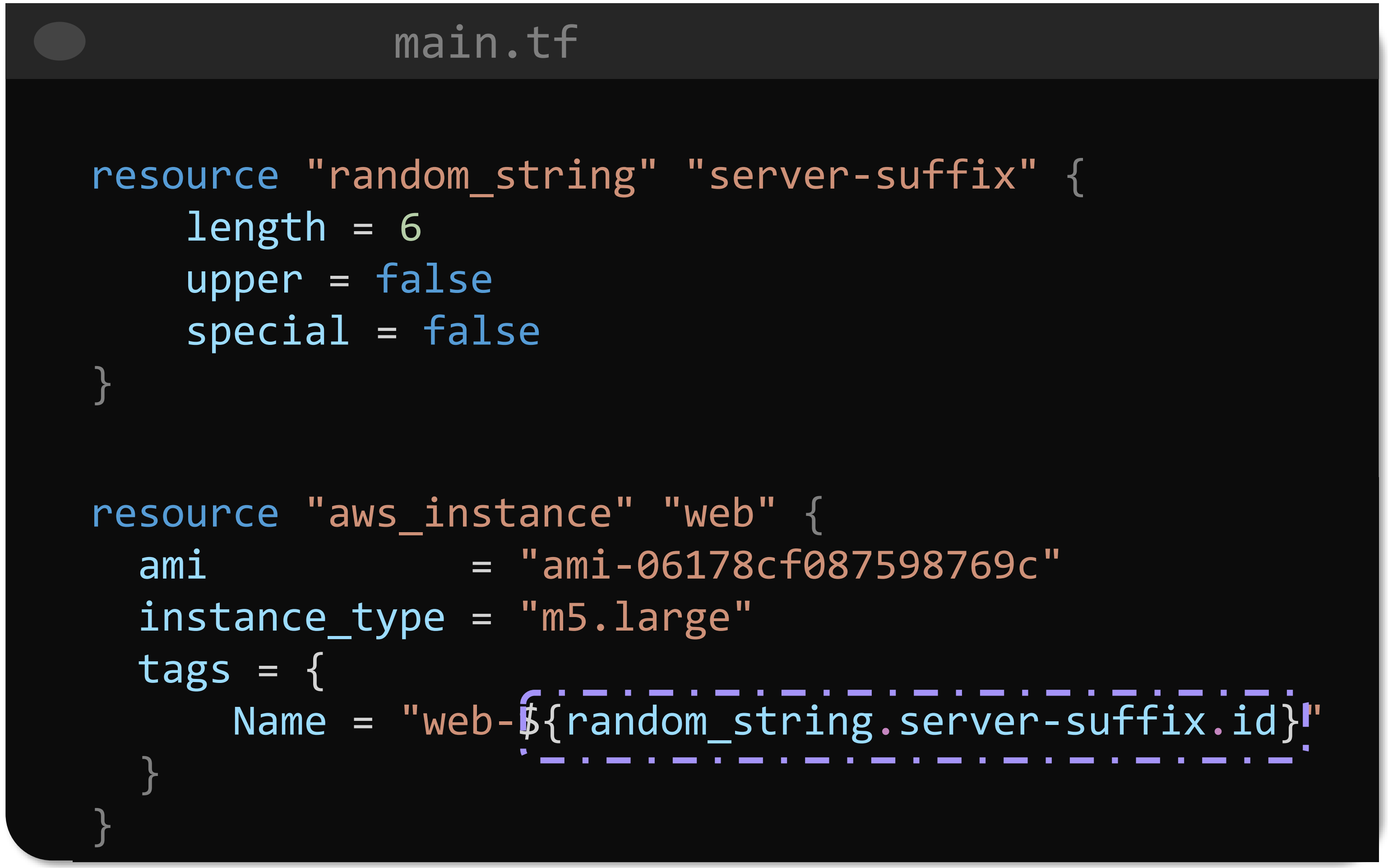
main.tf

```
resource "aws_instance"  "db" {
  ami           = var.db_ami
  instance_type = var.web_instance_type

}
resource "aws_instance"  "web" {
  ami           = var.web_ami
  instance_type = var.db_instance_type
  depends_on = [
    aws_instance.db
  ]
}
```



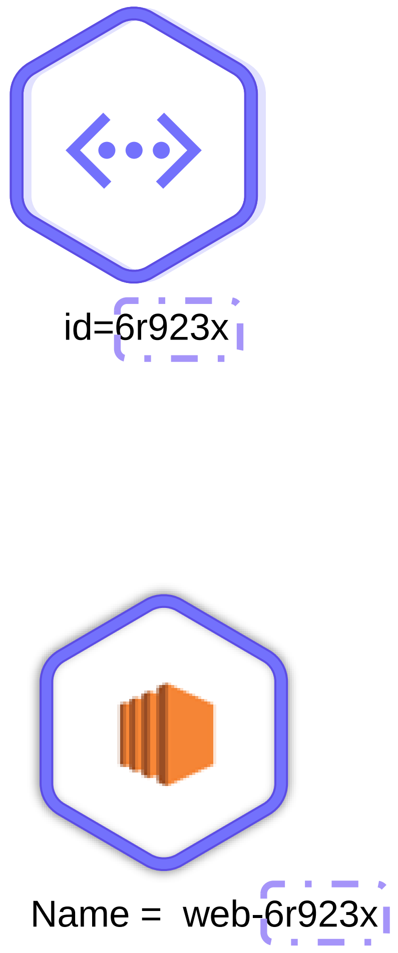
KODEKLOUD

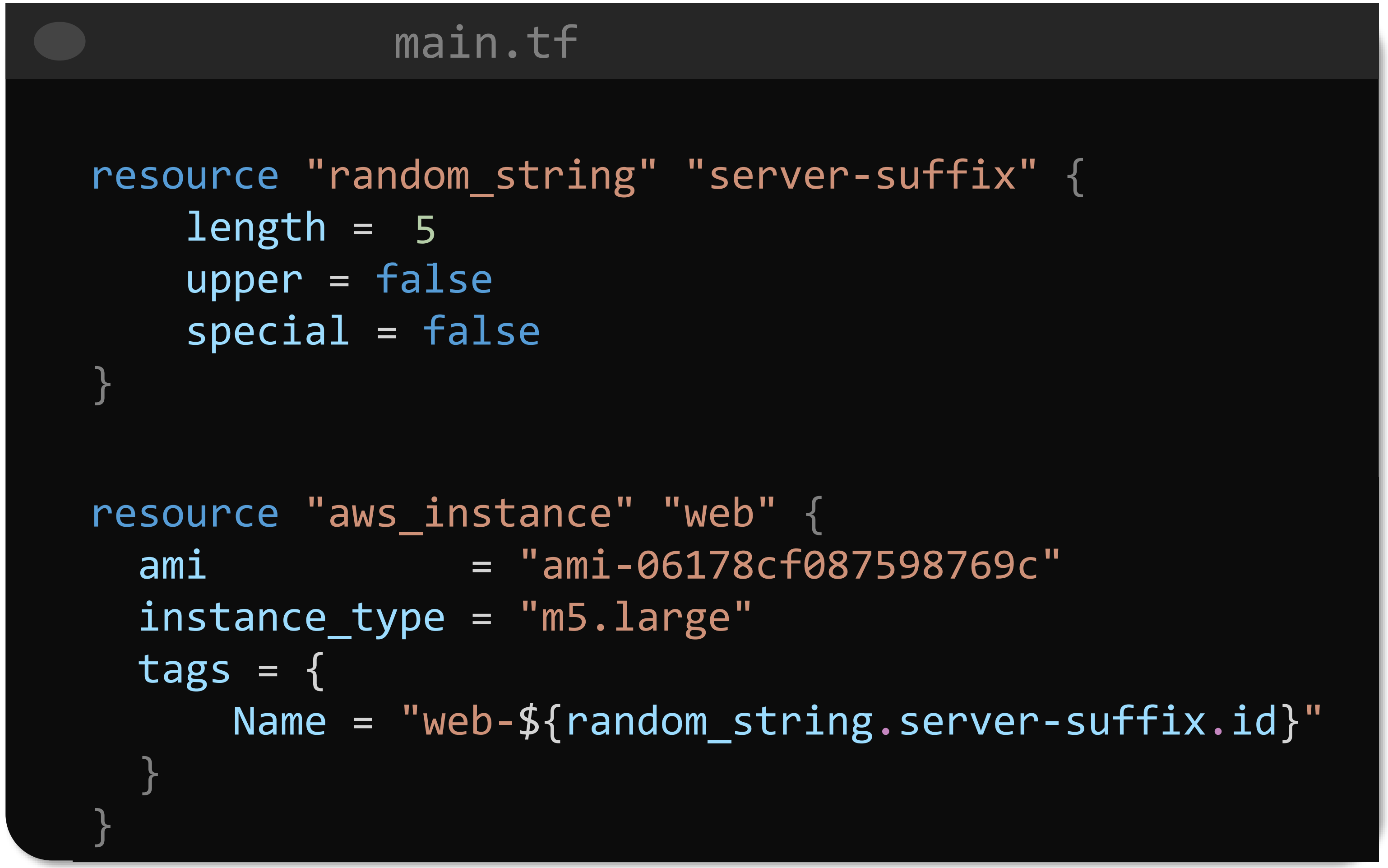
Resource Targetting



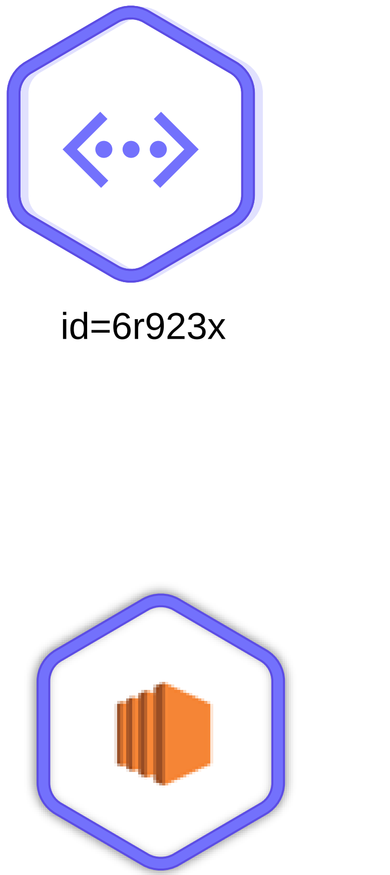
main.tf

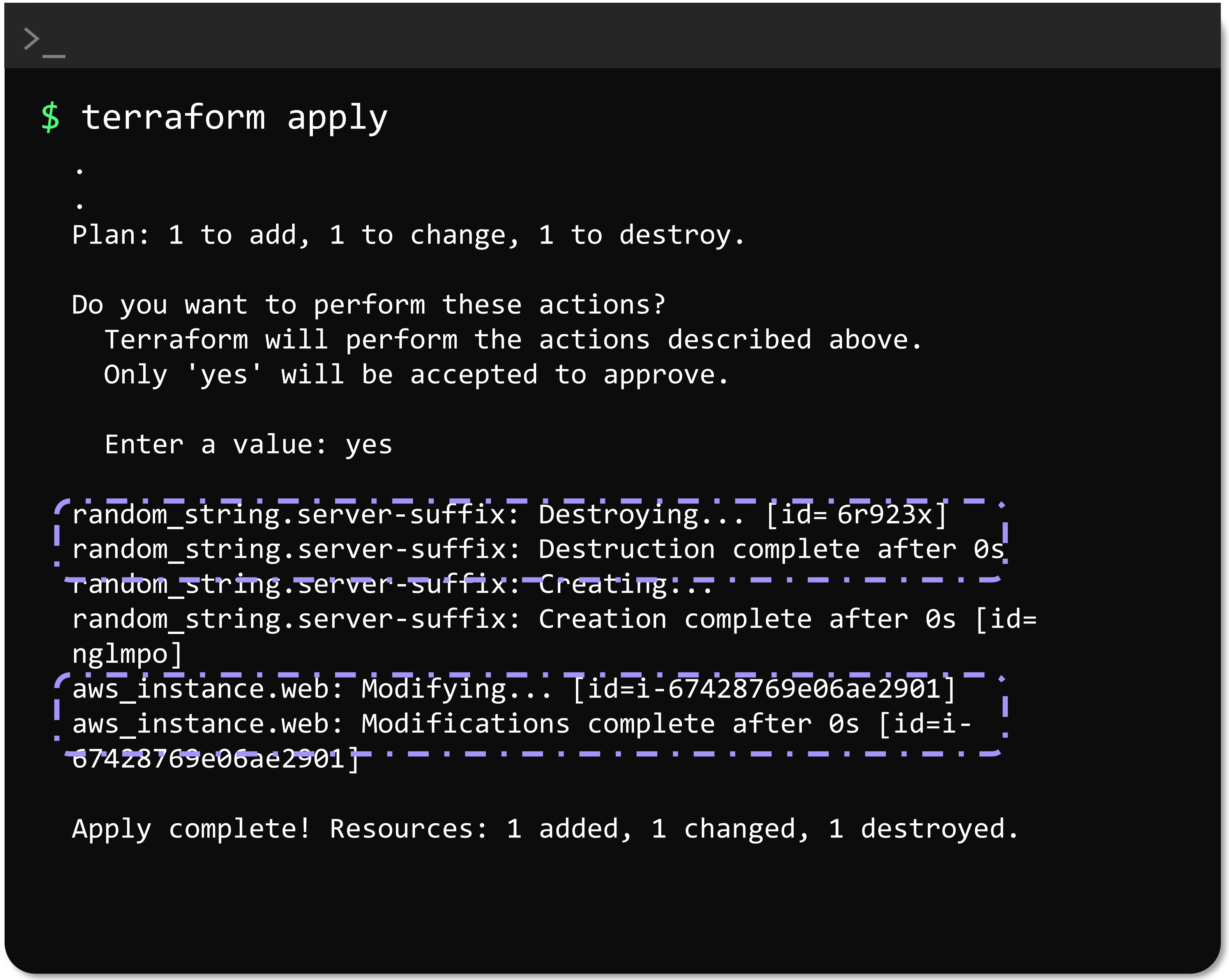
```
resource "random_string" "server-suffix" {
    length = 6
    upper = false
    special = false
}

resource "aws_instance" "web" {
  ami           = "ami-06178cf087598769c"
  instance_type = "m5.large"
  tags = {
      Name = "web-${random_string.server-suffix.id}"
  }
}
```



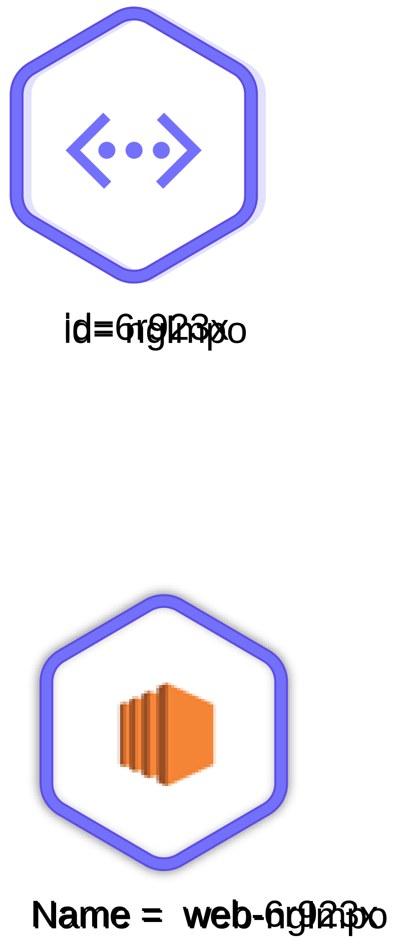

main.tf

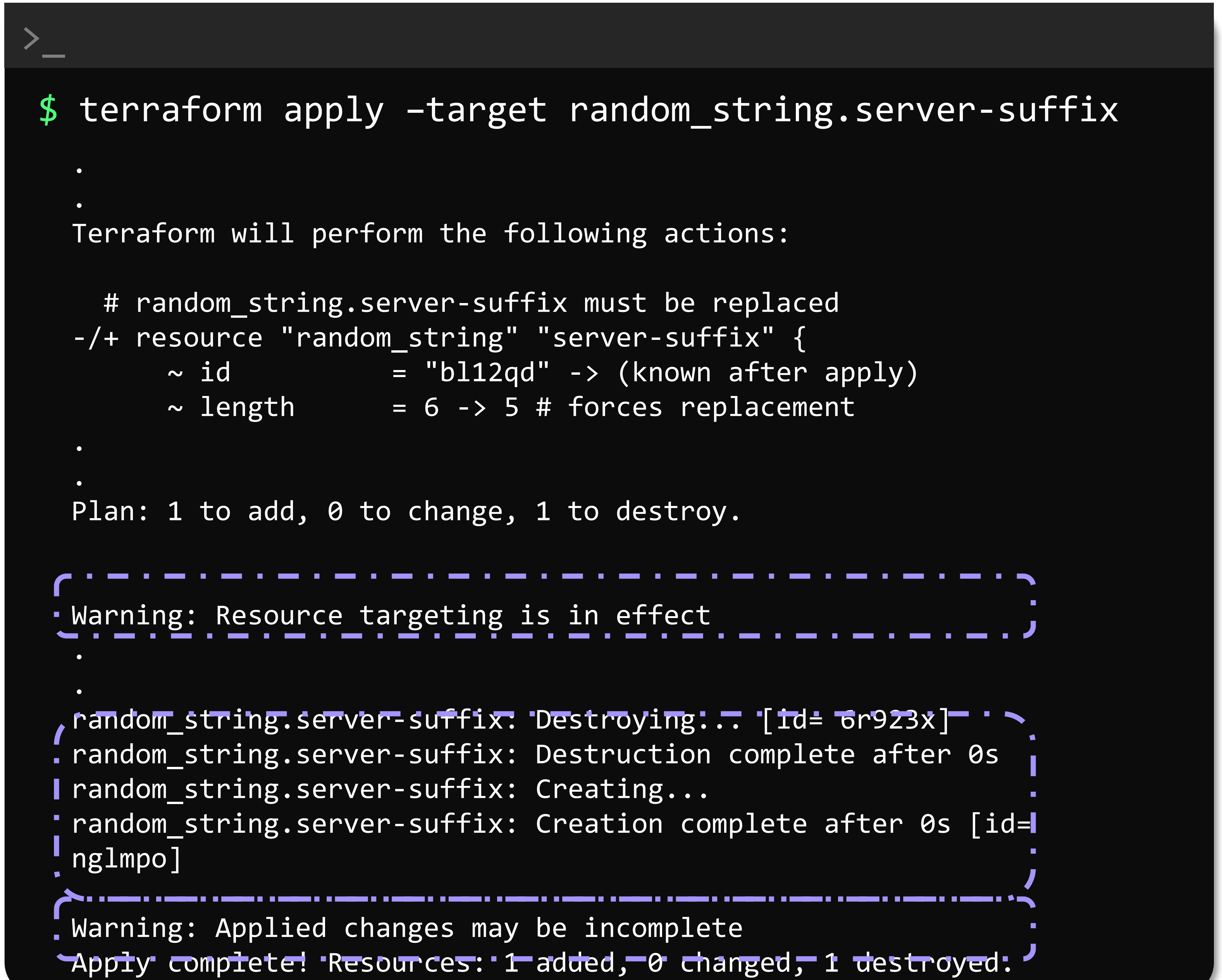
```
resource "random_string" "server-suffix" {
    length =  5
    upper = false
    special = false
}


resource "aws_instance" "web" {
  ami             = "ami-06178cf087598769c"
  instance_type = "m5.large"
  tags = {
      Name = "web-${random_string.server-suffix.id}"
  }
}
```



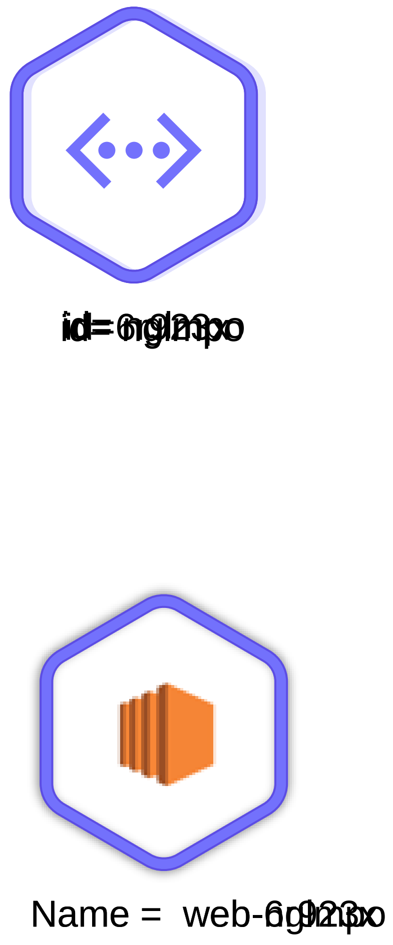


```
$ terraform apply
  .
  .
  Plan: 1 to add, 1 to change, 1 to destroy.

  Do you want to perform these actions?
    Terraform will perform the actions described above.
    Only 'yes' will be accepted to approve.

    Enter a value: yes

  random_string.server-suffix: Destroying... [id= 6r923x]
  random_string.server-suffix: Destruction complete after 0s
  random_string.server-suffix: Creating...
  random_string.server-suffix: Creation complete after 0s [id=
  nglmpo]
  aws_instance.web: Modifying... [id=i-67428769e06ae2901]
  aws_instance.web: Modifications complete after 0s [id=i-
  67428769e06ae2901]

  Apply complete! Resources: 1 added, 1 changed, 1 destroyed.
```





```
$ terraform apply -target random_string.server-suffix
  .
  .
  Terraform will perform the following actions:

    # random_string.server-suffix must be replaced
  -/+ resource "random_string" "server-suffix" {
        ~ id            = "bl12qd" -> (known after apply)
        ~ length        = 6 -> 5 # forces replacement
  .
  .
  Plan: 1 to add, 0 to change, 1 to destroy.

  Warning: Resource targeting is in effect
  .
  .
  random_string.server-suffix: Destroying... [id= 6r923x]
  random_string.server-suffix: Destruction complete after 0s
  random_string.server-suffix: Creating...
  random_string.server-suffix: Creation complete after 0s [id=
  nglmpo]

  Warning: Applied changes may be incomplete
  Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```



KODEKLOUD

KODEKLOUD

Output Variables

main.tf

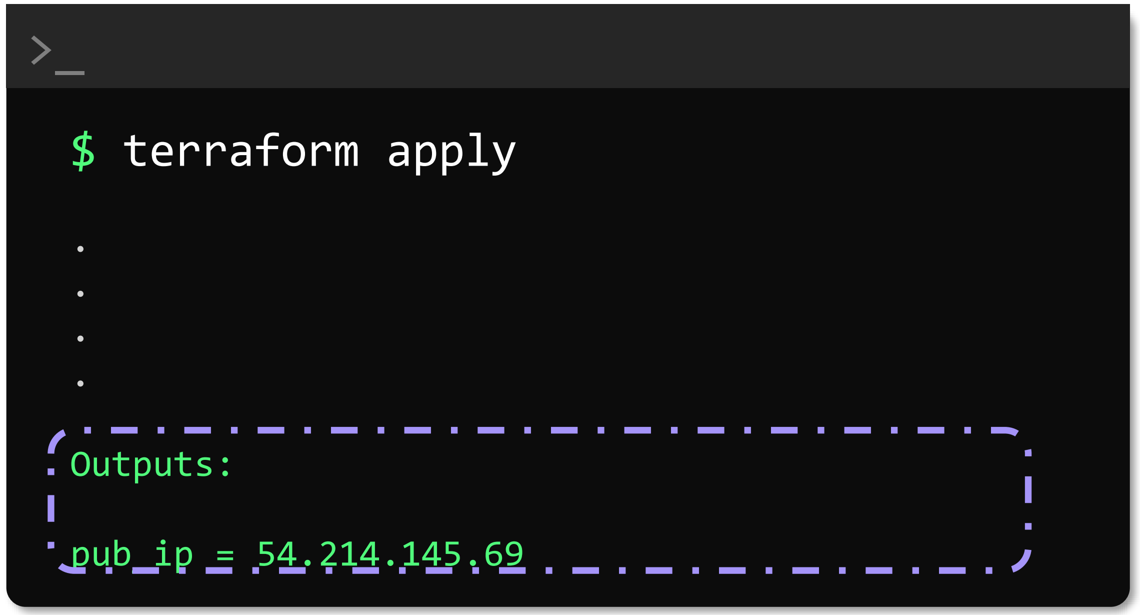
```
resource "aws_instance" "cerberus" {
  ami           = var.ami
  instance_type = var.instance_type
}


output "pub_ip" {
  value = aws_instance.cerberus.public_ip
  description = "print the public IPv4 address"
}
```

variables.tf

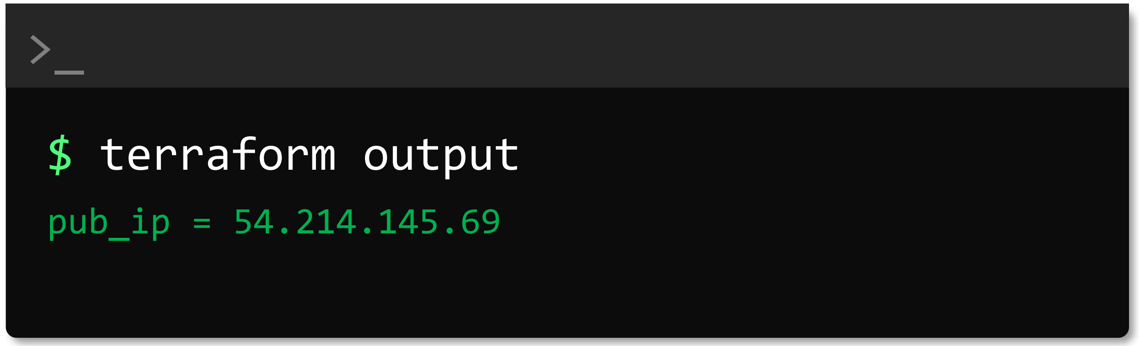
```
variable "ami" {
  default = "ami-06178cf087598769c"
}

variable "instance_type" {
  default = "m5.large"
}

variable "region" {
  default = "eu-west-2"
}
```

```
output "<variable_name>" {
  value = "<variable_value>"
  <arguments>
}
```

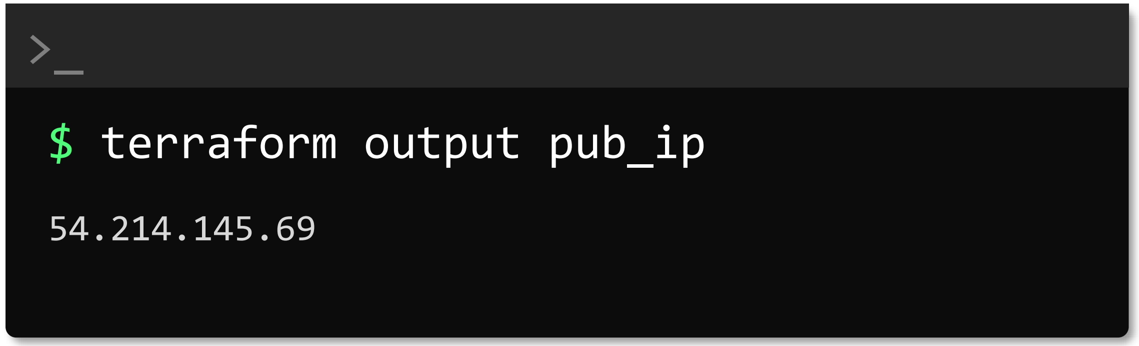
KODEKLOUD



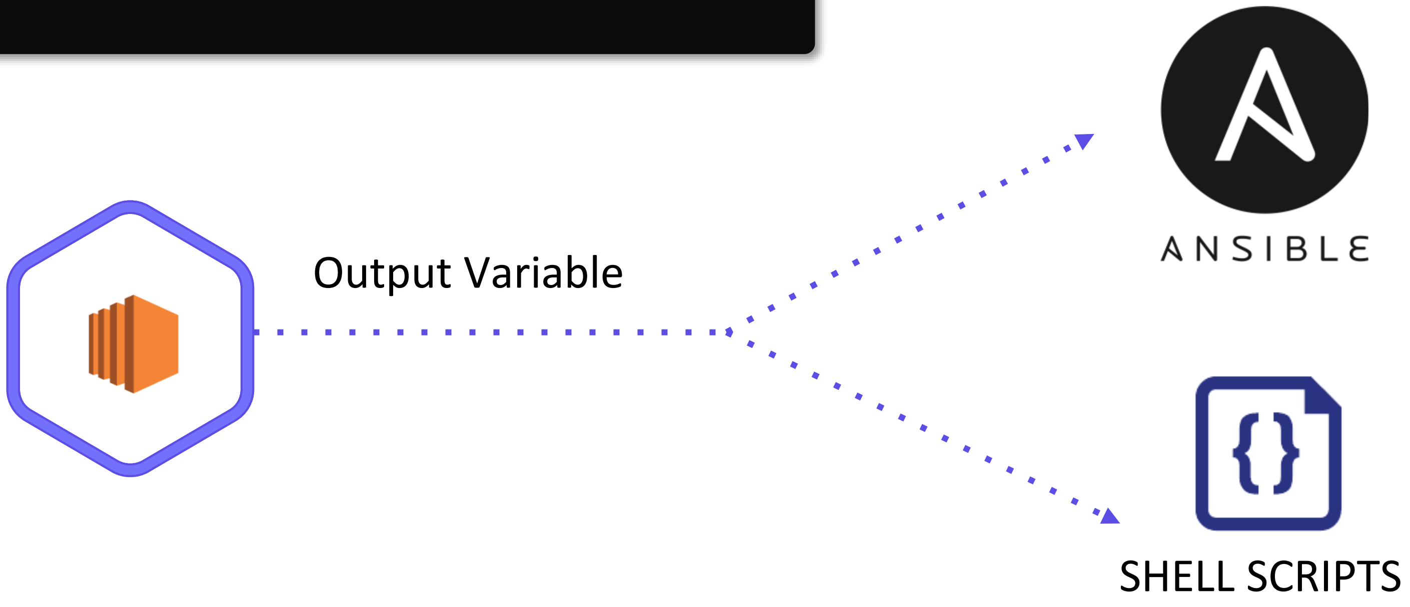
```
$ terraform apply

.
.
.
.
Outputs:

pub_ip = 54.214.145.69
```



```
$ terraform output
pub_ip = 54.214.145.69
```



```
$ terraform output pub_ip
54.214.145.69
```



KODEKLOUD

Recap Terraform State

main.tf

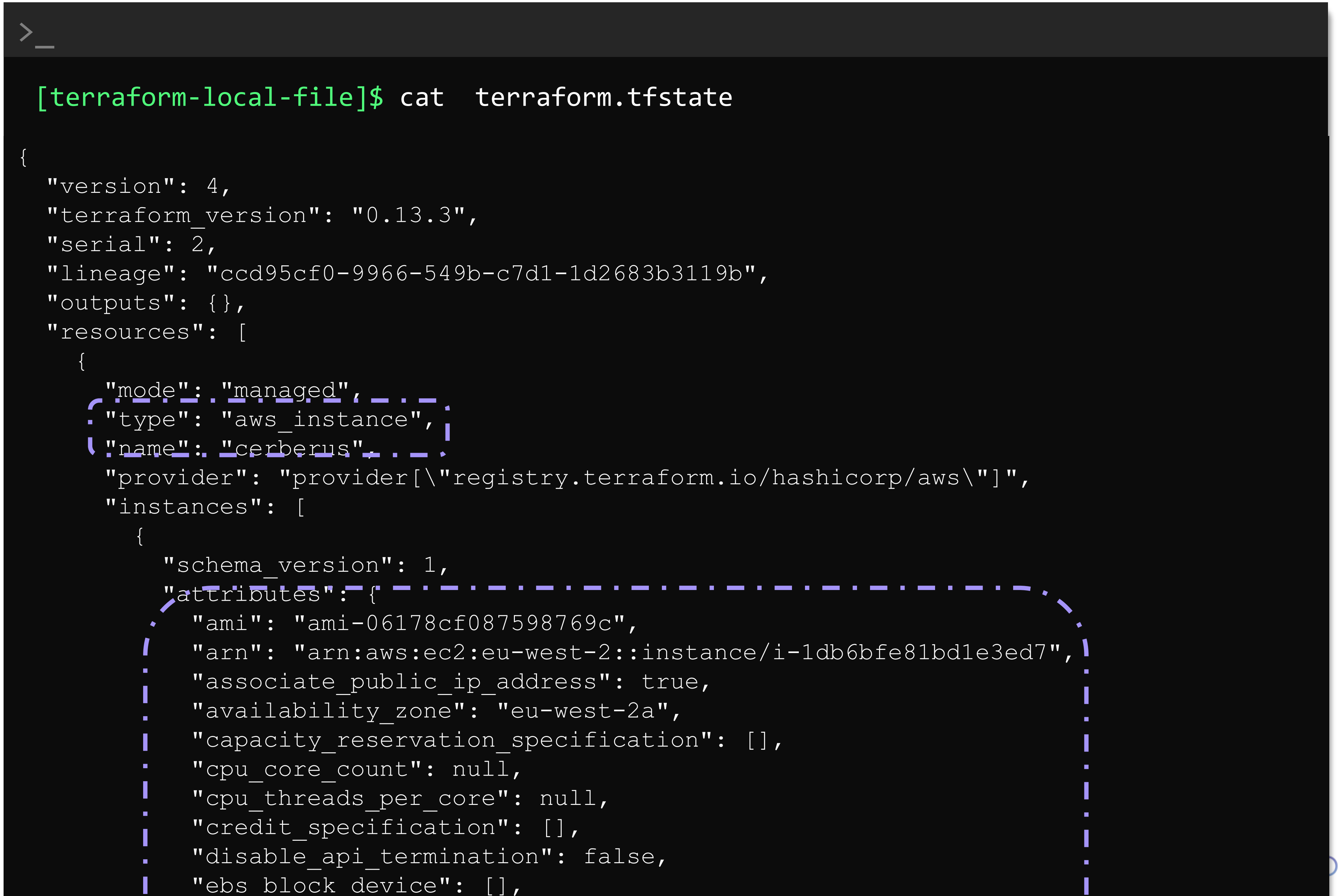
```
resource "aws_instance" "cerberus" {
  ami           = var.ami
  instance_type = var.instance_type
}
```

variables.tf

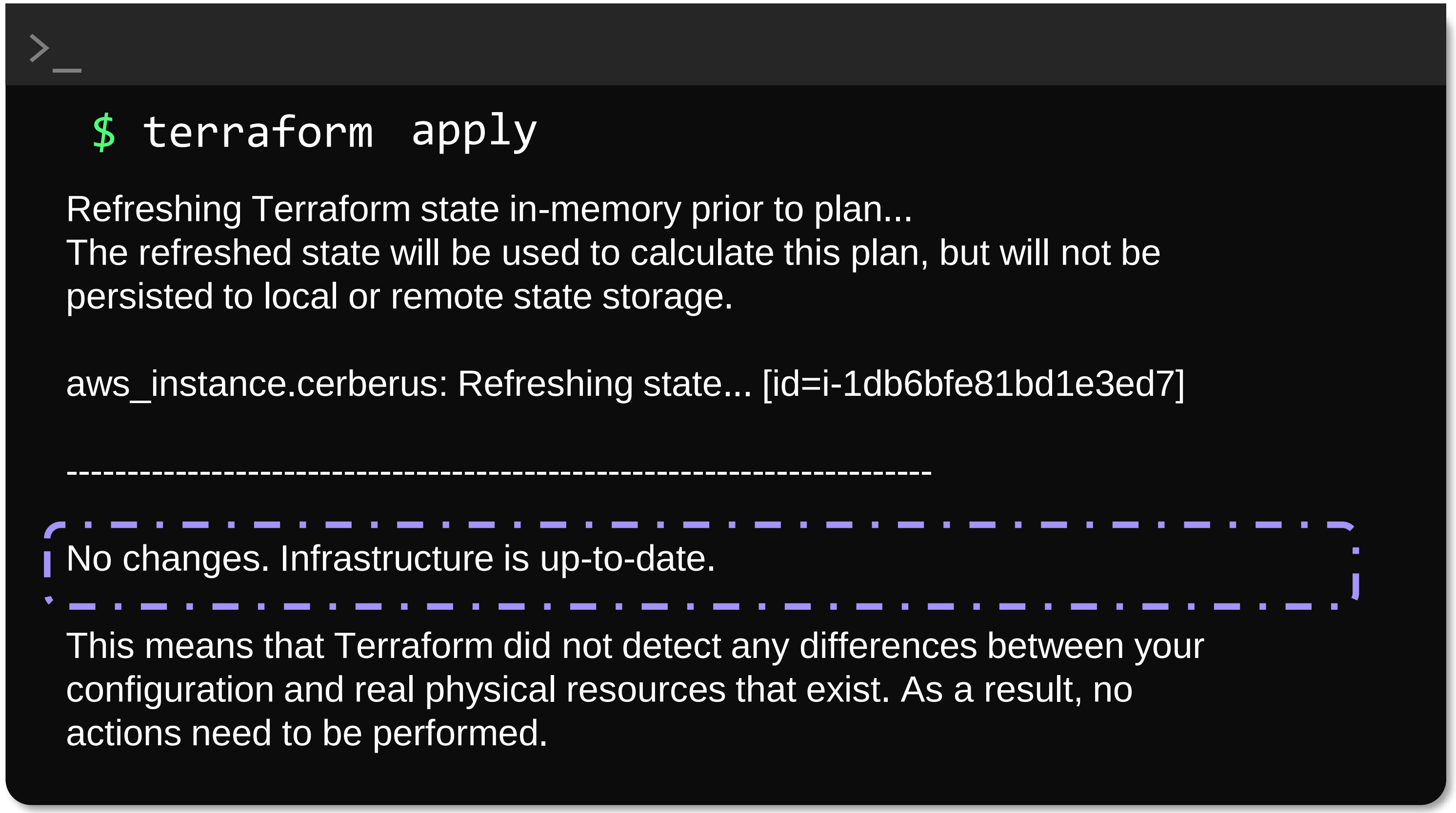
```
variable "ami" {
  default = "ami-06178cf087598769c"
}
variable "instance_type" {
  default = "m5.large"
}
```

```
$ terraform apply

.
.
.

aws_instance.cerberus: Creating...
aws_instance.cerberus: Still creating... [10s elapsed]
aws_instance.cerberus: Creation complete after 10s [id=i-
c791dc46a6639d4a7]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed
```

```
$ ls

main.tf  variables.tf  terraform.tfstate  terraform.tfstate.backup
```

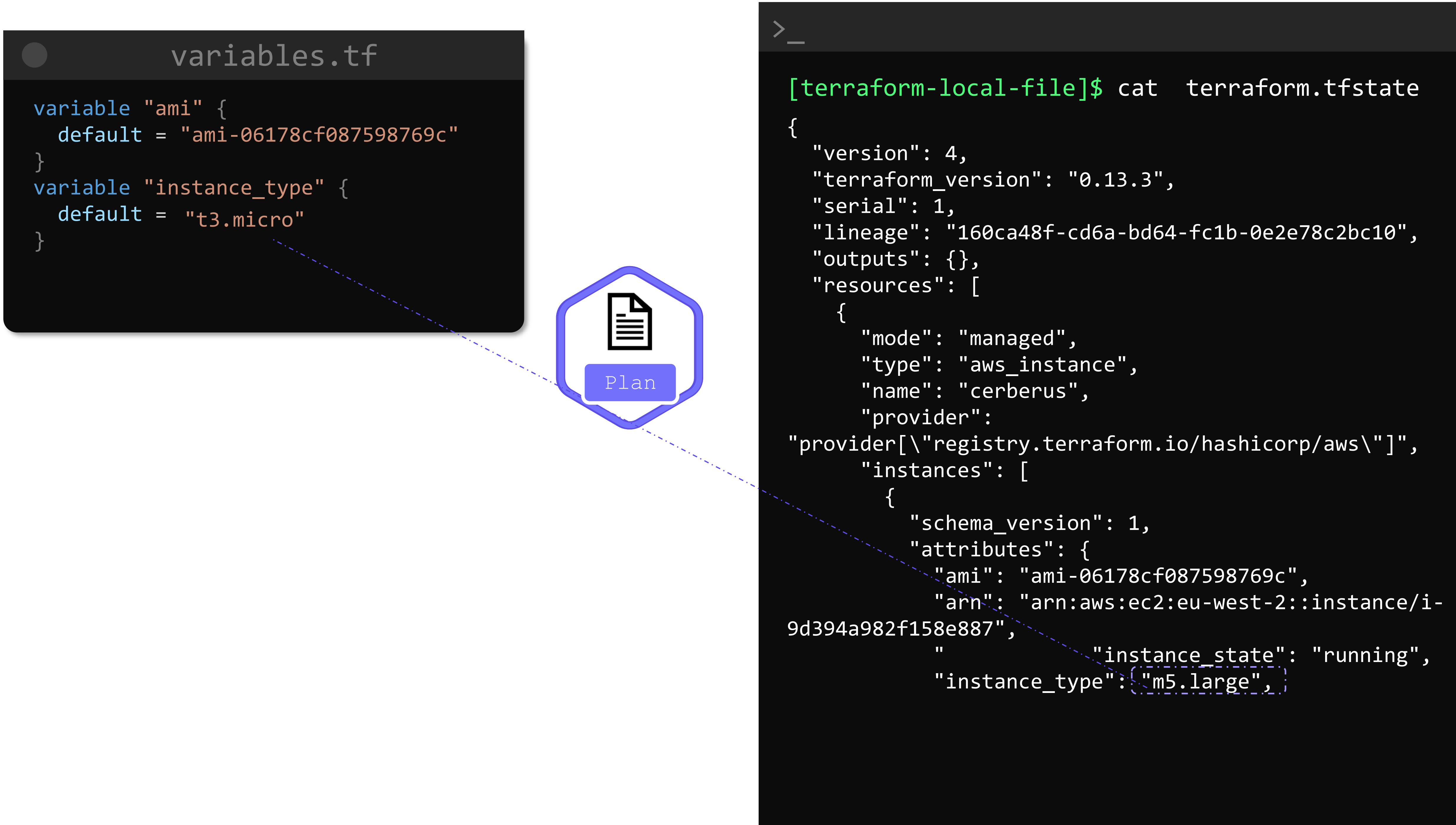


```
[terraform-local-file]$ cat  terraform.tfstate

{
  "version": 4,
  "terraform_version": "0.13.3",
  "serial": 2,
  "lineage": "ccd95cf0-9966-549b-c7d1-1d2683b3119b",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "aws_instance",
      "name": "cerberus",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-06178cf087598769c",
            "arn": "arn:aws:ec2:eu-west-2::instance/i-1db6bfe81bd1e3ed7",
            "associate_public_ip_address": true,
            "availability_zone": "eu-west-2a",
            "capacity_reservation_specification": [],
            "cpu_core_count": null,
            "cpu_threads_per_core": null,
            "credit_specification": [],
            "disable_api_termination": false,
            "ebs_block_device": [],
```



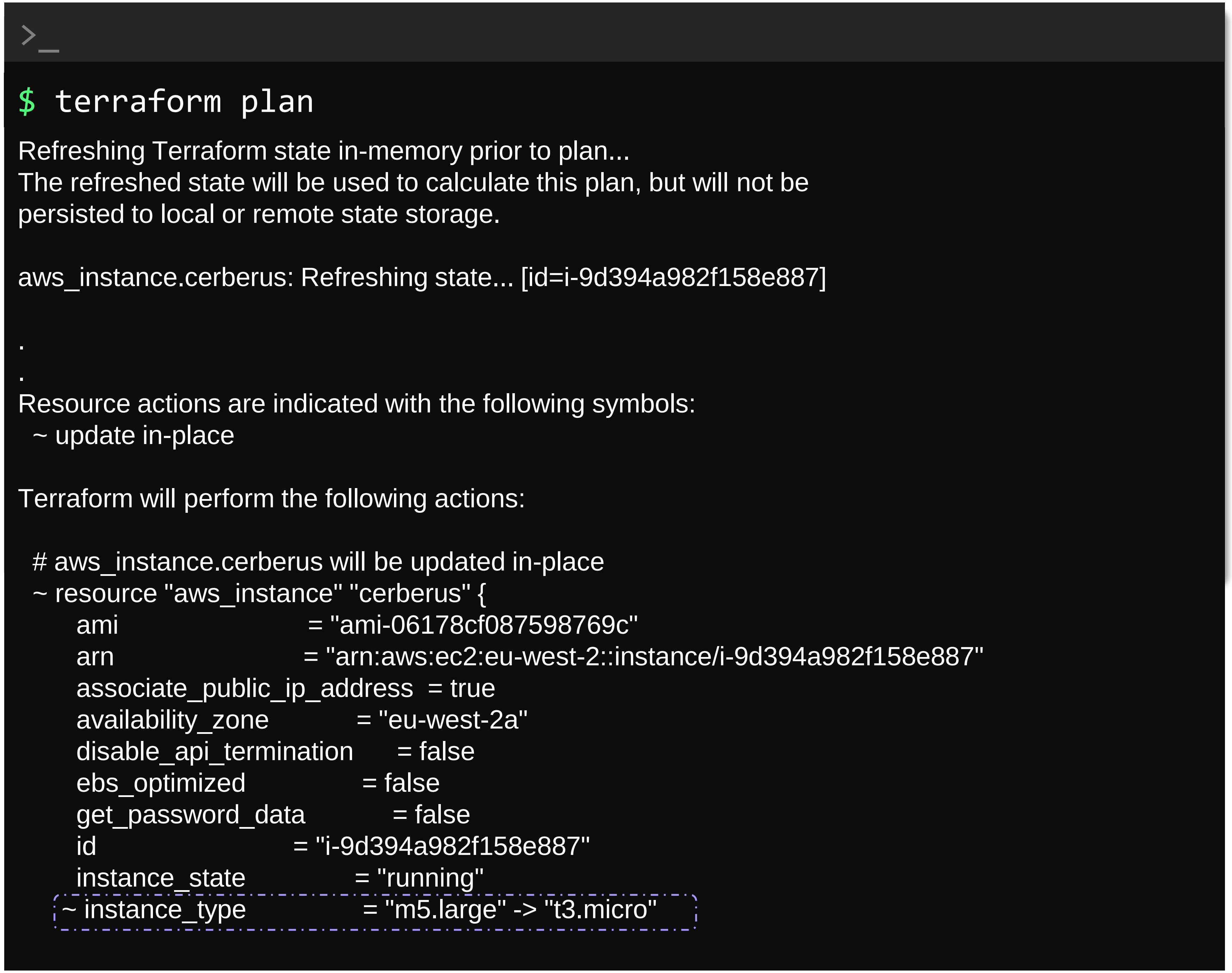
```
$ terraform apply

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_instance.cerberus: Refreshing state... [id=i-1db6bfe81bd1e3ed7]

------------------------------------------------------------------------

No changes. Infrastructure is up-to-date.

This means that Terraform did not detect any differences between your
configuration and real physical resources that exist. As a result, no
actions need to be performed.
```

```
$ terraform apply -refresh=false

  Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
```

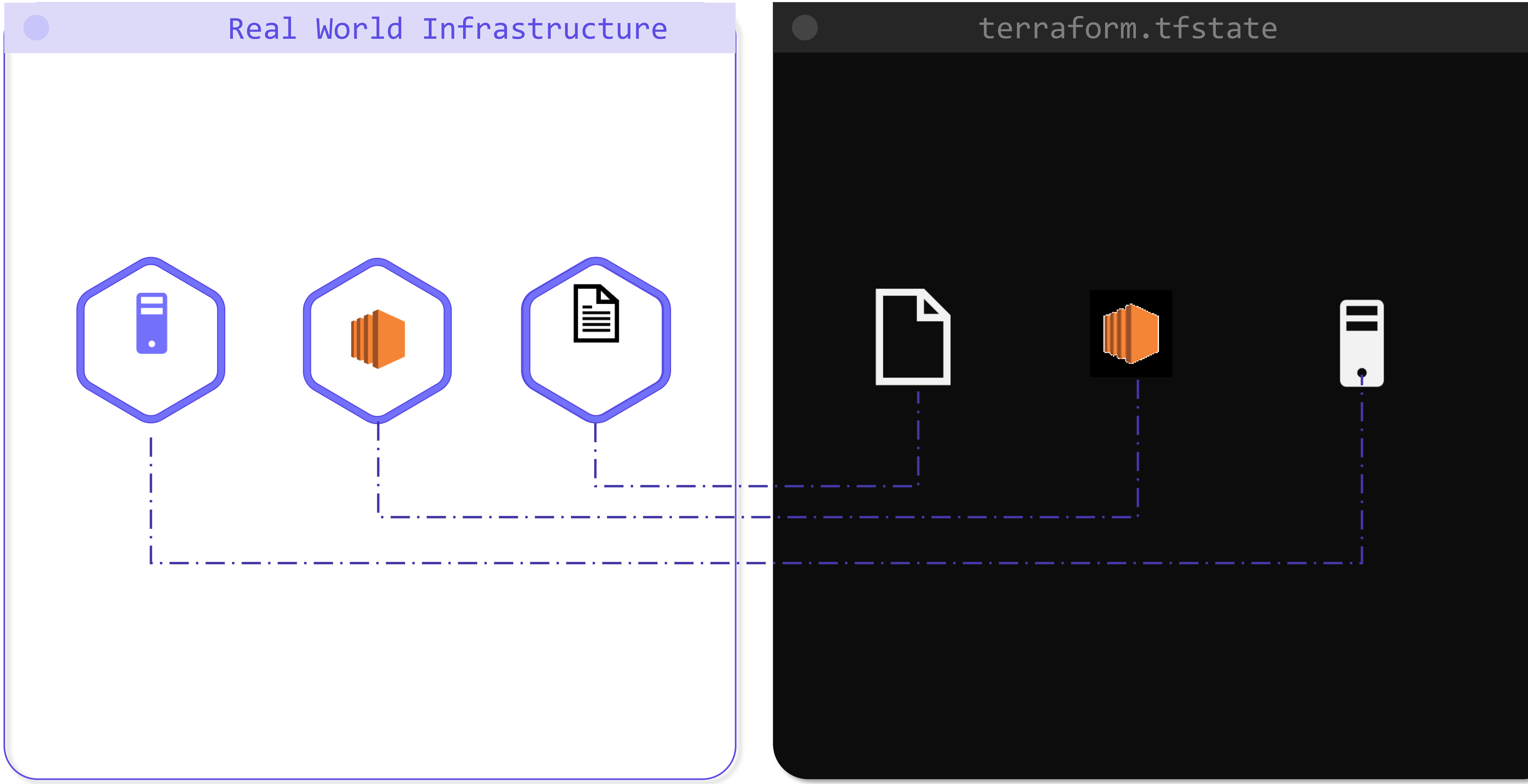

variables.tf

```
variable "ami" {
  default = "ami-06178cf087598769c"
}
variable "instance_type" {
  default = "t3.micro"
}
```

```
>_
[terraform-local-file]$ cat  terraform.tfstate
{
  "version": 4,
  "terraform_version": "0.13.3",
  "serial": 1,
  "lineage": "160ca48f-cd6a-bd64-fc1b-0e2e78c2bc10",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "aws_instance",
      "name": "cerberus",
      "provider":
"provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-06178cf087598769c",
            "arn": "arn:aws:ec2:eu-west-2::instance/i-
9d394a982f158e887",
            "             "instance_state": "running",
            "instance_type": "m5.large",
```

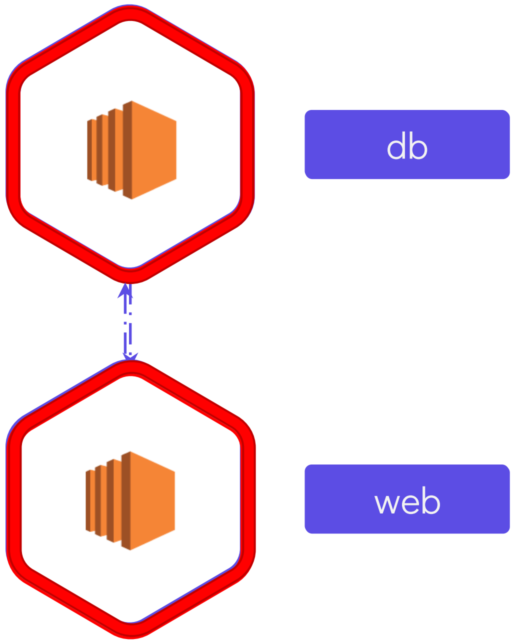


```
$ terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.

aws_instance.cerberus: Refreshing state... [id=i-9d394a982f158e887]

.
.
Resource actions are indicated with the following symbols:
  ~ update in-place

Terraform will perform the following actions:

  # aws_instance.cerberus will be updated in-place
  ~ resource "aws_instance" "cerberus" {
        ami                          = "ami-06178cf087598769c"
        arn                          = "arn:aws:ec2:eu-west-2::instance/i-9d394a982f158e887"
        associate_public_ip_address  = true
        availability_zone            = "eu-west-2a"
        disable_api_termination      = false
        ebs_optimized                = false
        get_password_data            = false
        id                           = "i-9d394a982f158e887"
        instance_state               = "running"
      ~ instance_type                = "m5.large" -> "t3.micro"
```


Real World Infrastructure
terraform.tfstate

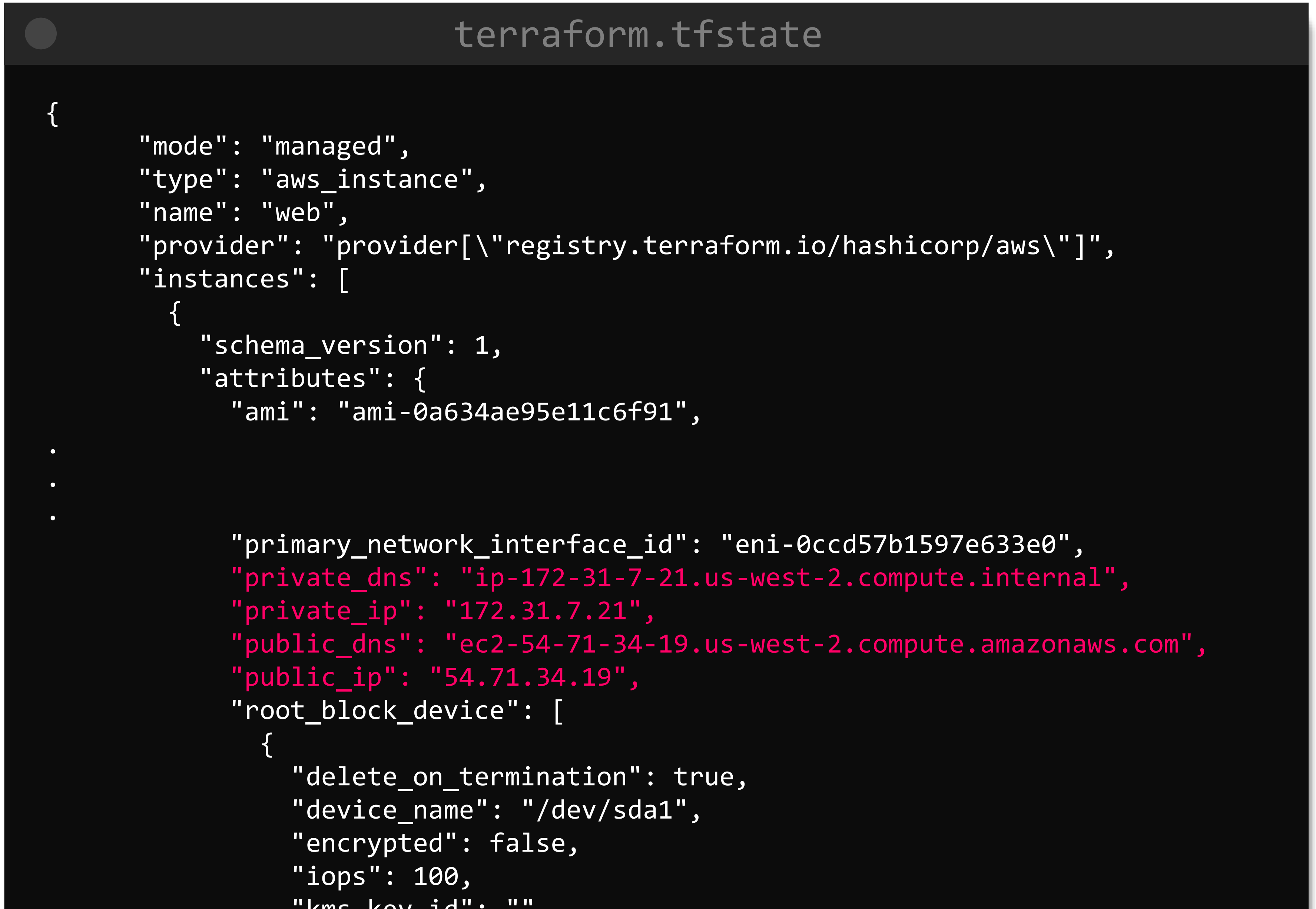
main.tf

```
resource "aws_instance"  "db" {
  ami           = var.ami
  instance_type = var.instance_type
}

resource "aws_instance"  "web" {
  ami           = var.ami
  instance_type = var.instance_type
  depends_on = [ aws_instance.db ]

}
```



```
>_
[terraform-local-file]$ cat  terraform.tfstate
 {
      "mode": "managed",
      "type": "aws_instance",
      "name": "web",
      "provider":
"provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-06178cf087598769c",
            "arn": "arn:aws:ec2:eu-west-2::instance/i-
33b55018bd1a8d8ca",
.
.
.

          "dependencies": [
            "aws_instance.db"
          ]
```

Sensitive Data



terraform.tfstate

```
{
      "mode": "managed",
      "type": "aws_instance",
      "name": "web",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-0a634ae95e11c6f91",
.
.
.
            "primary_network_interface_id": "eni-0ccd57b1597e633e0",
            "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
            "private_ip": "172.31.7.21",
            "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
            "public_ip": "54.71.34.19",
            "root_block_device": [
              {
                "delete_on_termination": true,
                "device_name": "/dev/sda1",
                "encrypted": false,
                "iops": 100,
                "kms_key_id": "",
```

Terraform State Considerations

Remote State Backends

amazon S3   HashiCorp Terraform Cloud

Version Control

GitHub

terraform.tfstate

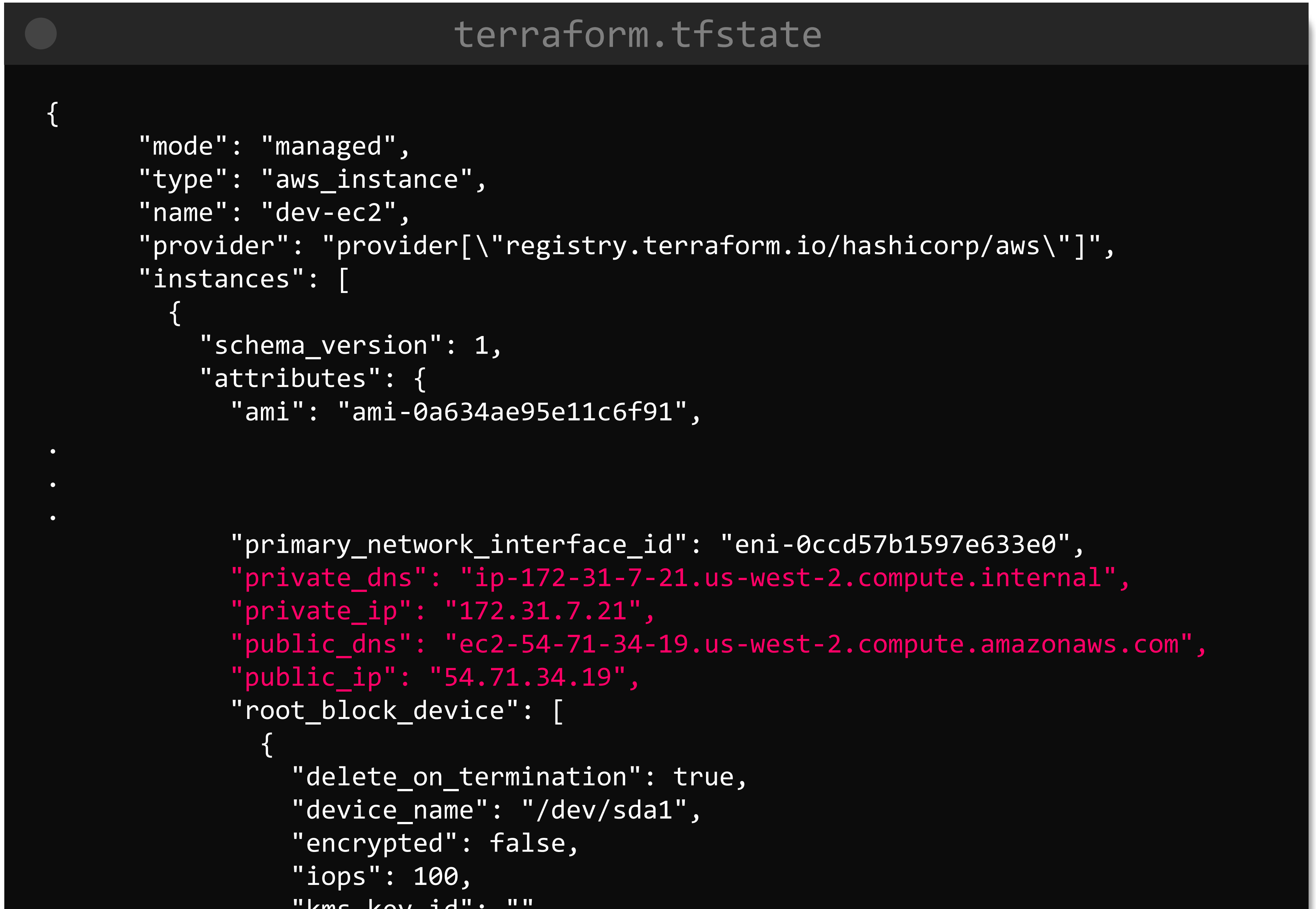
```
{
      "mode": "managed",
      "type": "aws_instance",
      "name": "web",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-0a634ae95e11c6f91",
.
.
.
            "primary_network_interface_id": "eni-0ccd57b1597e633e0",
            "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
            "private_ip": "172.31.7.21",
            "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
            "public_ip": "54.71.34.19",
            "root_block_device": [
              {
                "delete_on_termination": true,
                "device_name": "/dev/sda1",
                "encrypted": false,
                "iops": 100,
                "kms_key_id": "",
                "volume_id": "vol-070720a3636979c22",
                "volume_size": 8,
```

main.tf

```
resource "aws_instance"  "db" {
  ami            = var.ami
  instance_type = var.instance_type
}

resource "aws_instance"  "web" {
  ami            = var.ami
  instance_type = var.instance_type
  depends_on = [ aws_instance.db ]

}
```

KODEKLOUD

No Manual Edits



terraform.tfstate

```
{
      "mode": "managed",
      "type": "aws_instance",
      "name": "dev-ec2",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-0a634ae95e11c6f91",
.
.
.
            "primary_network_interface_id": "eni-0ccd57b1597e633e0",
            "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
            "private_ip": "172.31.7.21",
            "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
            "public_ip": "54.71.34.19",
            "root_block_device": [
              {
                "delete_on_termination": true,
                "device_name": "/dev/sda1",
                "encrypted": false,
                "iops": 100,
                "kms_key_id": "",
```

KODEKLOUD

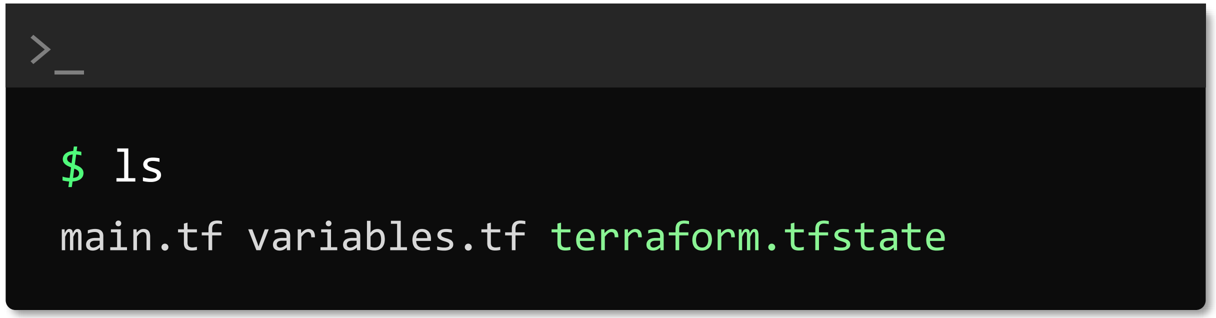
Remote State

Mapping Configuration to Real World

Tracking Metadata

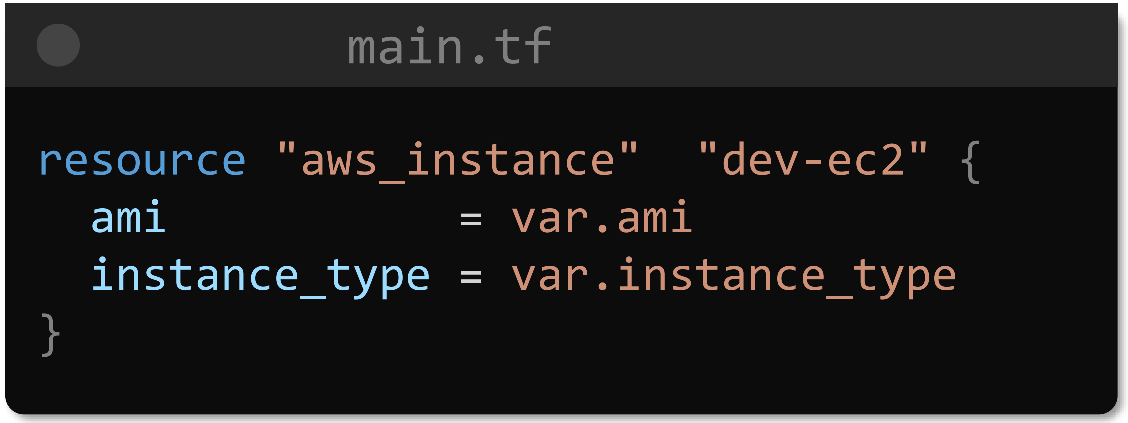
Performance

Collaboration



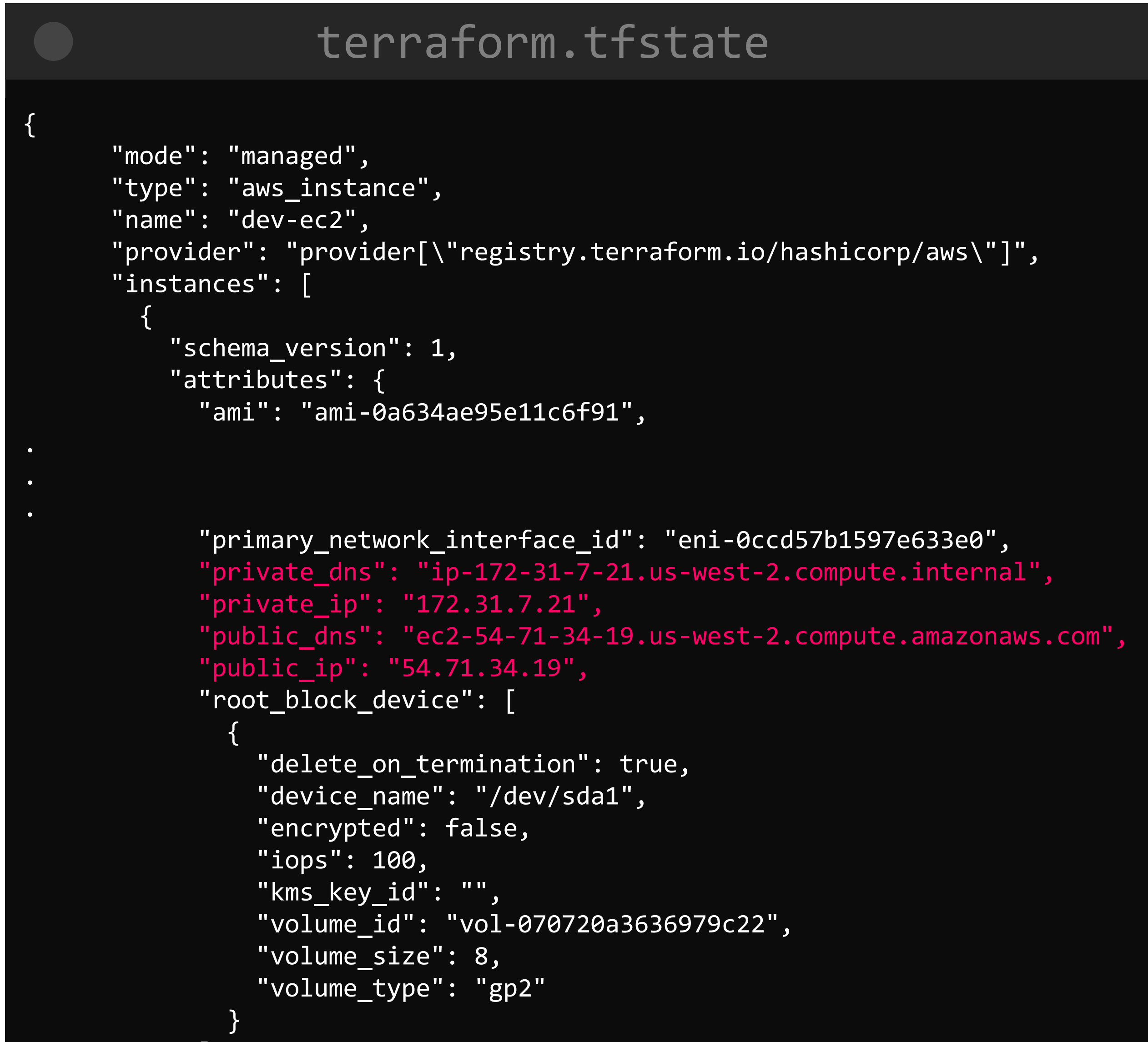
```
$ ls
main.tf variables.tf terraform.tfstate
```

Version
Control



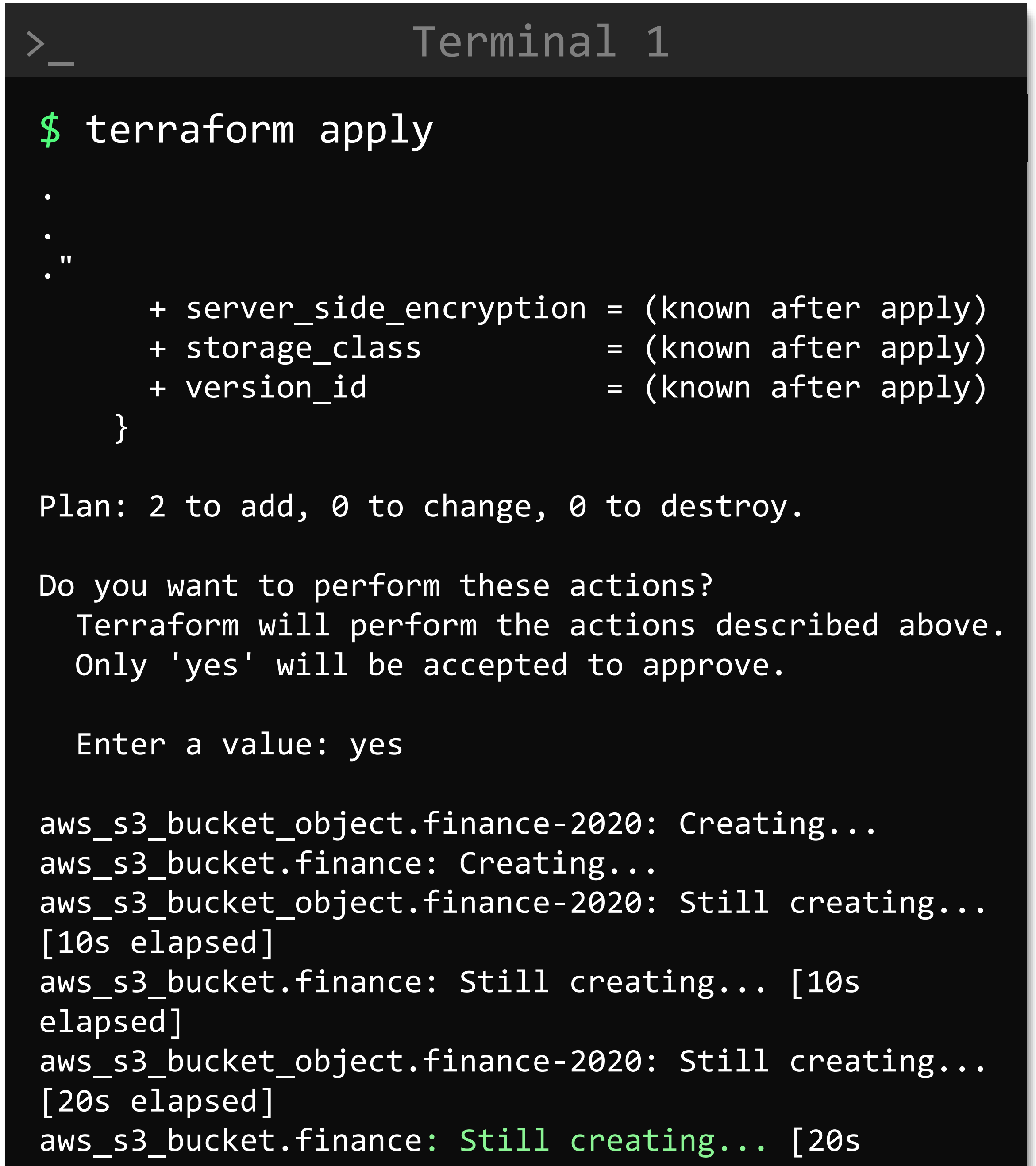
main.tf

```
resource "aws_instance"  "dev-ec2" {
  ami           = var.ami
  instance_type = var.instance_type
}
```

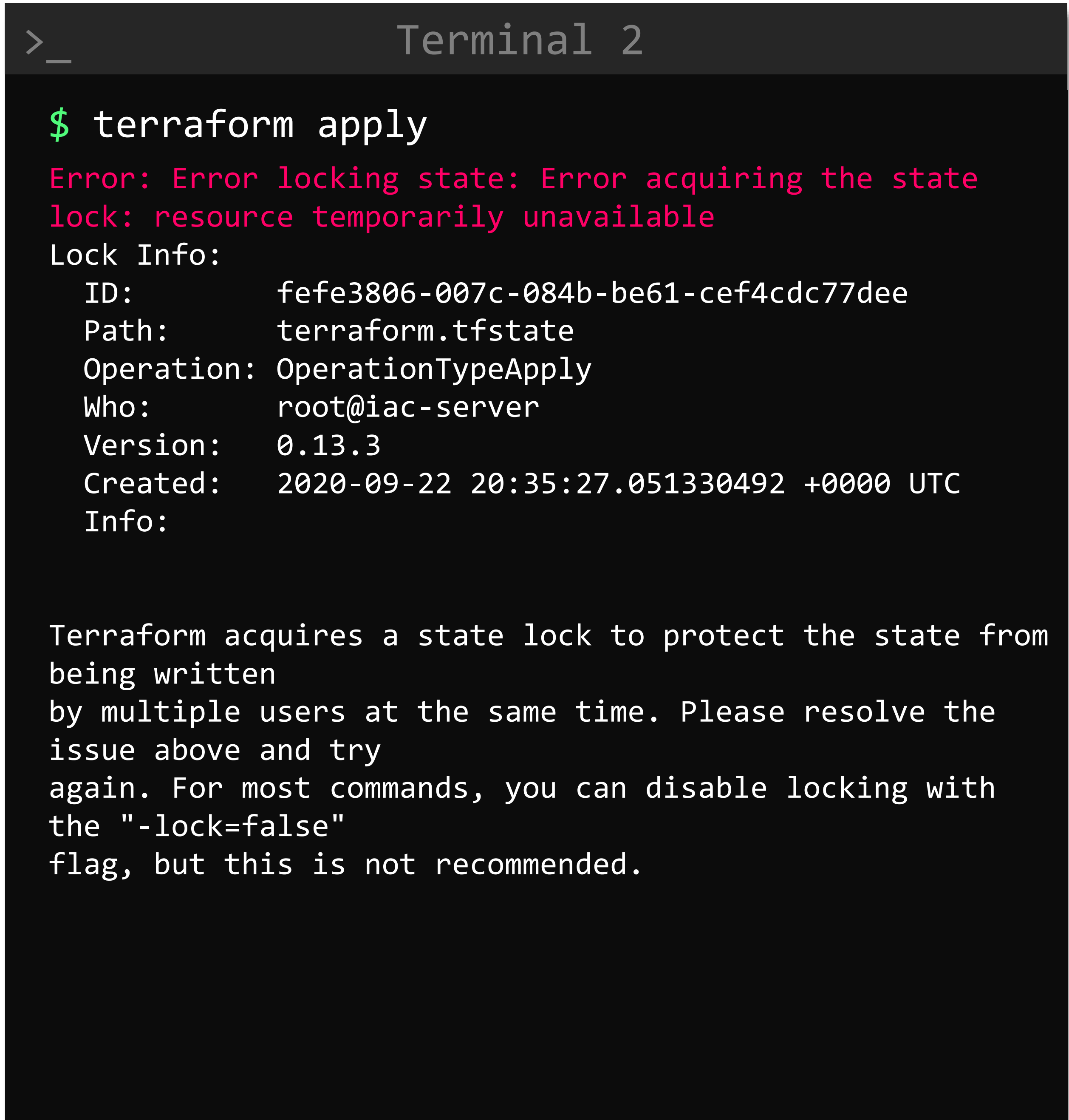


terraform.tfstate

```
{
      "mode": "managed",
      "type": "aws_instance",
      "name": "dev-ec2",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-0a634ae95e11c6f91",
.
.
.
            "primary_network_interface_id": "eni-0ccd57b1597e633e0",
            "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
            "private_ip": "172.31.7.21",
            "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
            "public_ip": "54.71.34.19",
            "root_block_device": [
              {
                "delete_on_termination": true,
                "device_name": "/dev/sda1",
                "encrypted": false,
                "iops": 100,
                "kms_key_id": "",
                "volume_id": "vol-070720a3636979c22",
                "volume_size": 8,
                "volume_type": "gp2"
              }
```

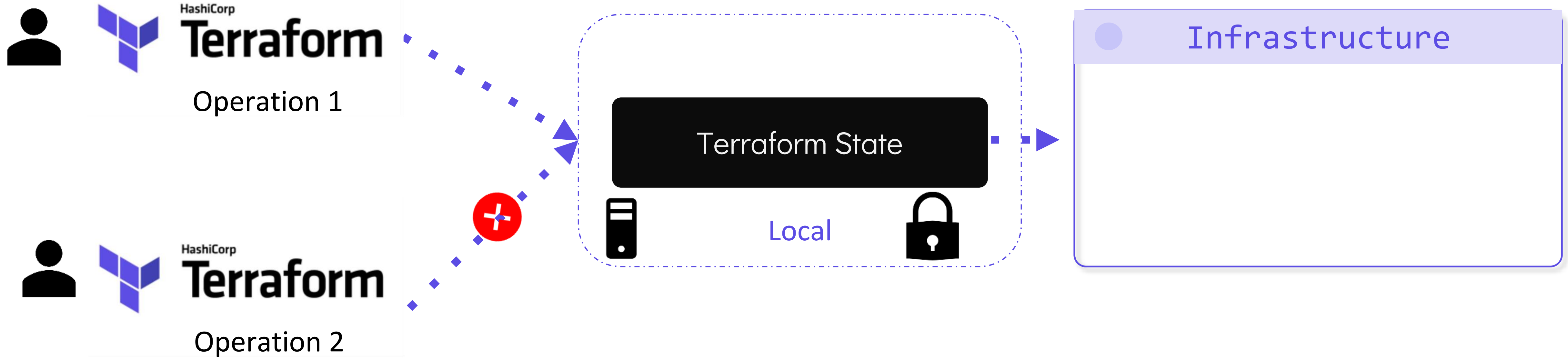


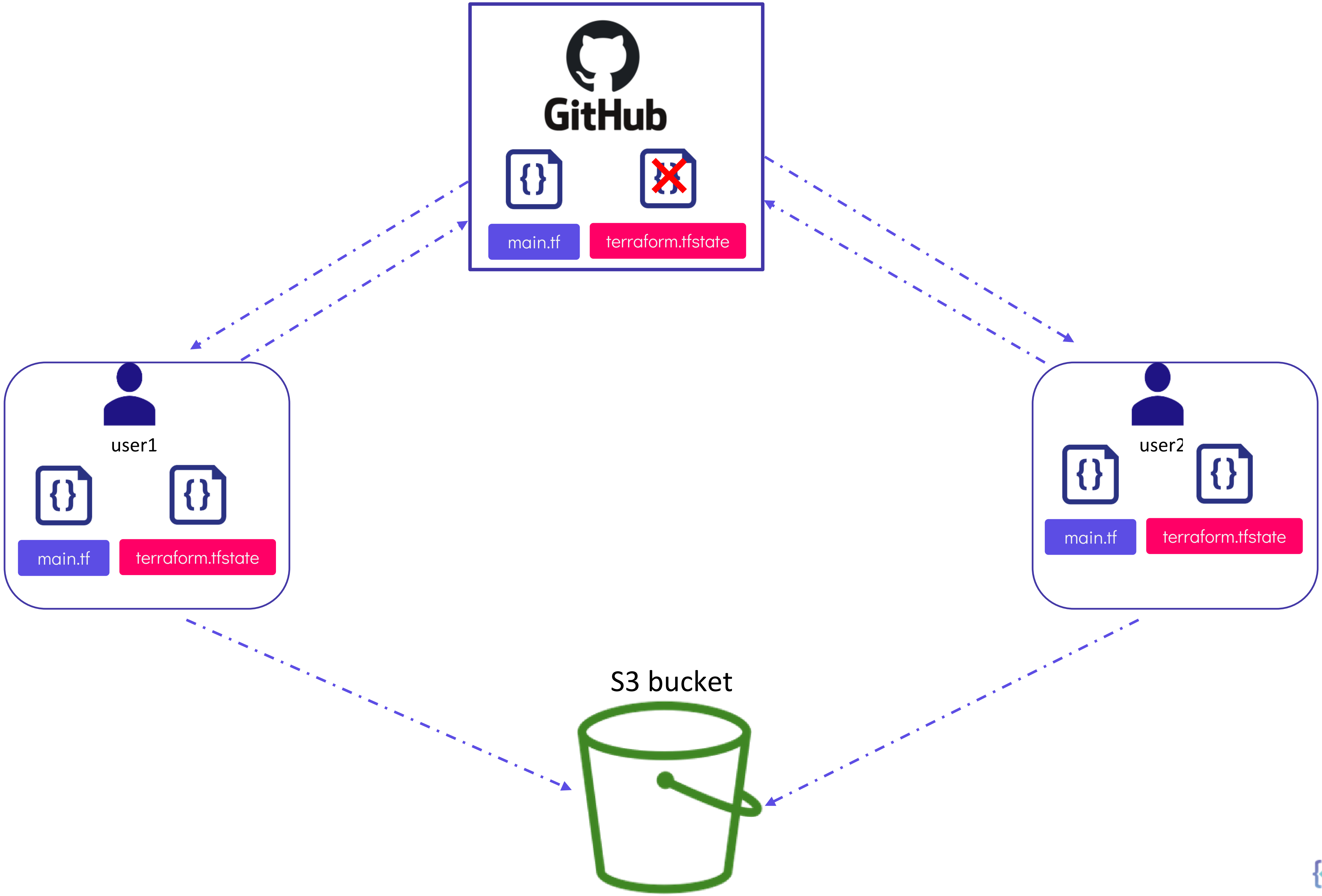
```
$ terraform apply
.
.
."
        + server_side_encryption = (known after apply)
        + storage_class          = (known after apply)
        + version_id             = (known after apply)
    }

Plan: 2 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_s3_bucket_object.finance-2020: Creating...
aws_s3_bucket.finance: Creating...
aws_s3_bucket_object.finance-2020: Still creating...
[10s elapsed]
aws_s3_bucket.finance: Still creating... [10s
elapsed]
aws_s3_bucket_object.finance-2020: Still creating...
[20s elapsed]
aws_s3_bucket.finance: Still creating... [20s
```



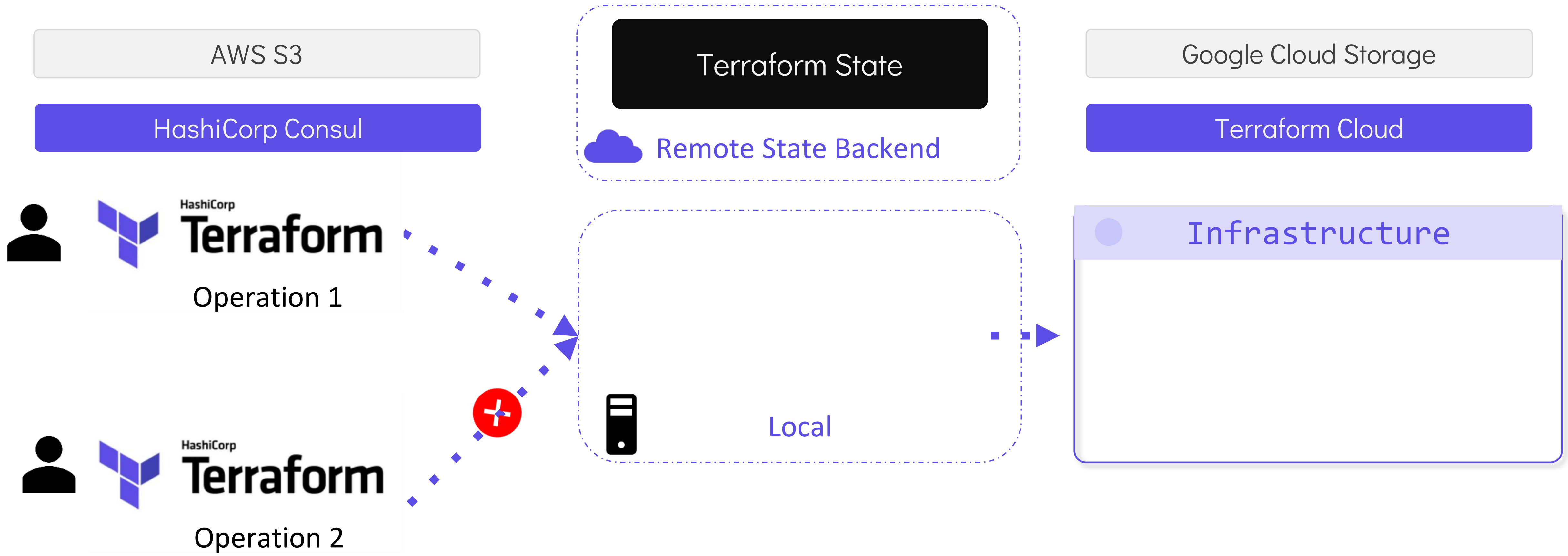
```
$ terraform apply
Error: Error locking state: Error acquiring the state
lock: resource temporarily unavailable
Lock Info:
  ID:        fefe3806-007c-084b-be61-cef4cdc77dee
  Path:      terraform.tfstate
  Operation: OperationTypeApply
  Who:       root@iac-server
  Version:   0.13.3
  Created:   2020-09-22 20:35:27.051330492 +0000 UTC
  Info:


Terraform acquires a state lock to protect the state from
being written
by multiple users at the same time. Please resolve the
issue above and try
again. For most commands, you can disable locking with
the "-lock=false"
flag, but this is not recommended.
```

State Locking




GitHub
main.tf
terraform.tfstate
user1
main.tf
terraform.tfstate
user2
main.tf
terraform.tfstate
S3 bucket

State Locking



State Locking

AWS S3

HashiCorp Consul

Terraform State

Remote State Backend

Google Cloud Storage

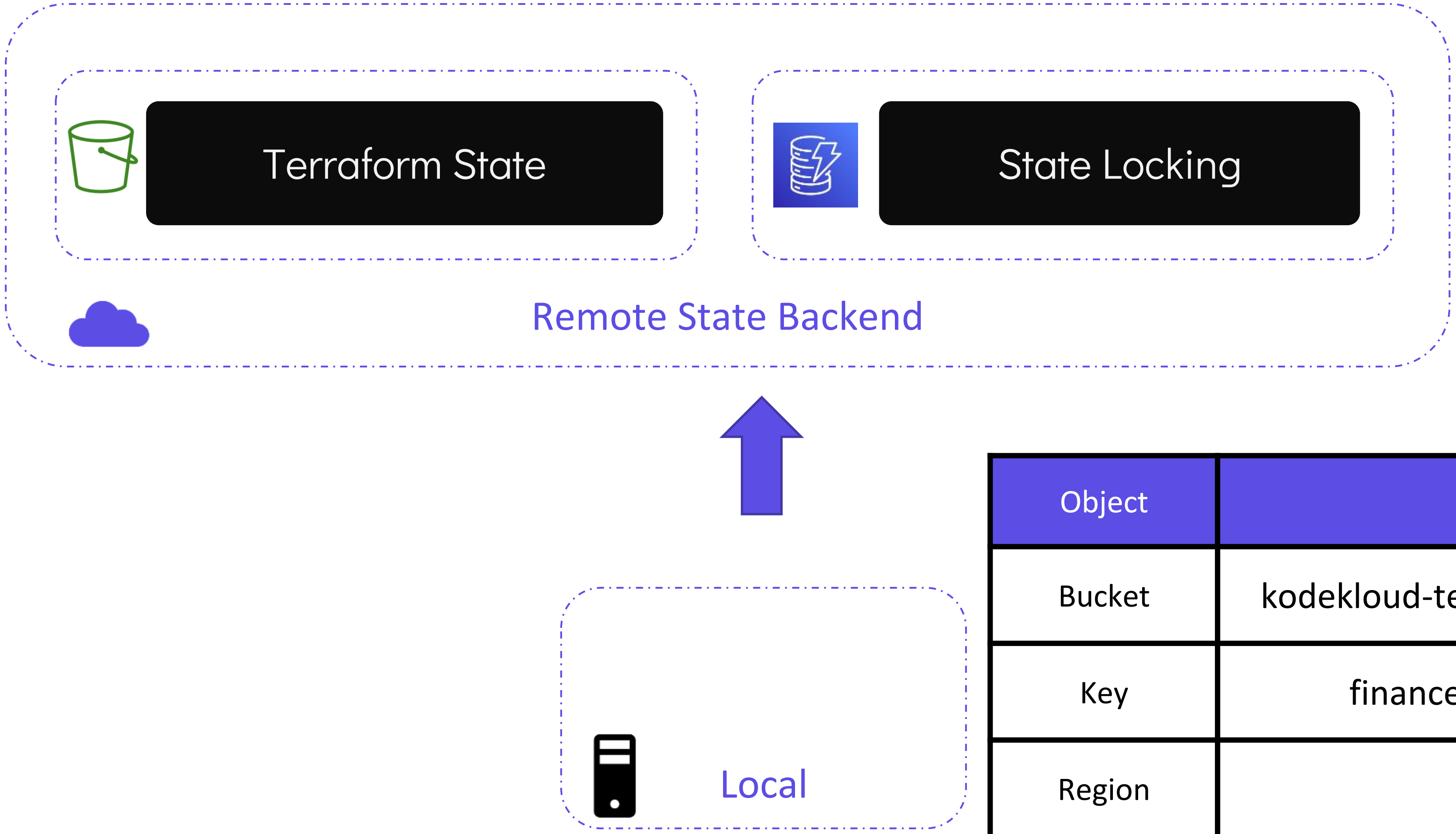
Terraform Cloud

Automatically Load and Upload State File

Many Backends Support State Locking

Security

Remote Backend



Object	Value
Bucket	kodekloud-terraform-state-bucket01
Key	finance/terraform.tfstate
Region	us-west-1
DynamoDB Table	state-locking

main.tf

```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
}
```

terraform.tf

```
terraform {
  backend "s3" {
    bucket         = "kodekloud-terraform-state-bucket01"
    key            = "finance/terraform.tfstate"
    region         = "us-west-1"
    dynamodb_table = "state-locking"
  }
}
```

Object	Value
Bucket	kodekloud-terraform-state-bucket01
Key	finance/terraform.tfstate
Region	us-west-1
DynamoDB Table	state-locking

main.tf

```
resource "local_file" "pet" {
  filename = "/root/pets.txt"
  content = "We love pets!"
}
```

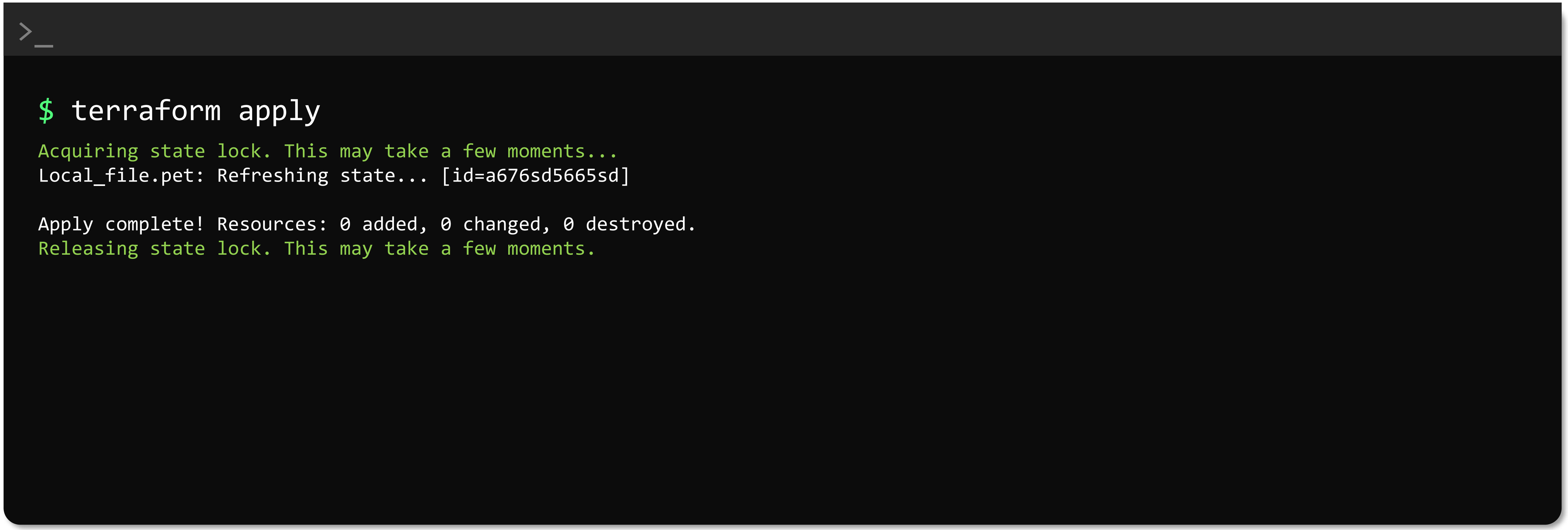
terraform.tf

```
terraform {
  backend "s3" {
    bucket         = "kodekloud-terraform-state-bucket01"
    key            = "finance/terraform.tfstate"
    region         = "us-west-1"
    dynamodb_table = "state-locking"
  }
}
```

```
>_
$ terraform apply
Backend reinitialization required. Please run "terraform init". Reason: Initial configuration of the requested backend "s3"

The "backend" is the interface that Terraform uses to store state, perform operations, etc. If this message is showing up, it means
that the Terraform configuration you're using is using a custom configuration for the Terraform backend.

Changes to backend configurations require reinitialization. This allows Terraform to setup the new configuration, copy existing
state, etc. This is only done during "terraform init". Please run that command now then try again.

Error: Initialization required. Please see the error message above.
```

```
$ terraform init
Initializing the backend...
Do you want to copy existing state to the new backend?
  Pre-existing state was found while migrating the previous "local" backend to the newly configured "s3" backend. No existing state
was found in the newly configured "s3" backend. Do you want to copy this state to the new "s3"
  backend? Enter "yes" to copy and "no" to start with an empty state.

  Enter a value: yes

Successfully configured the backend "s3"! Terraform will automatically use this backend unless the backend configuration changes.

Initializing provider plugins...
- Using previously-installed hashicorp/aws v3.7.0
.
.[Output Truncated]
```

```
$ rm -rf terraform.tfstate
```

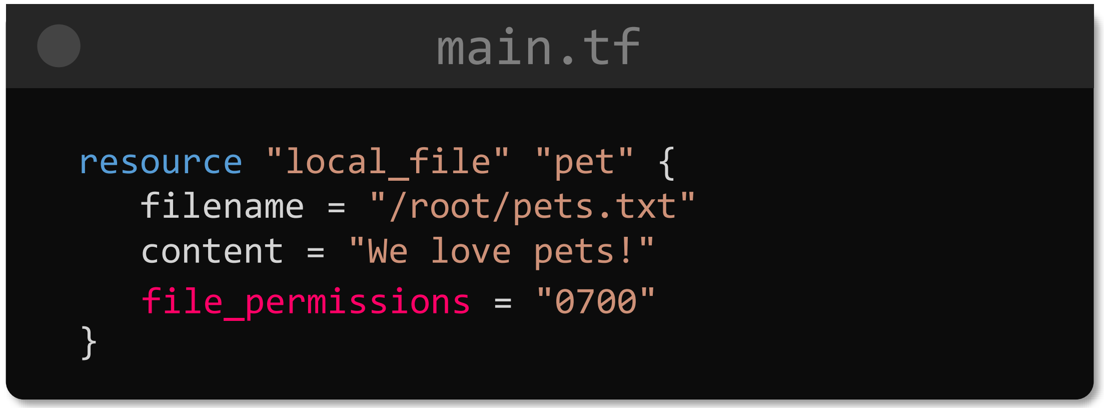


```
$ terraform apply
Acquiring state lock. This may take a few moments...
Local_file.pet: Refreshing state... [id=a676sd5665sd]

Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
Releasing state lock. This may take a few moments.
```

KODEKLOUD

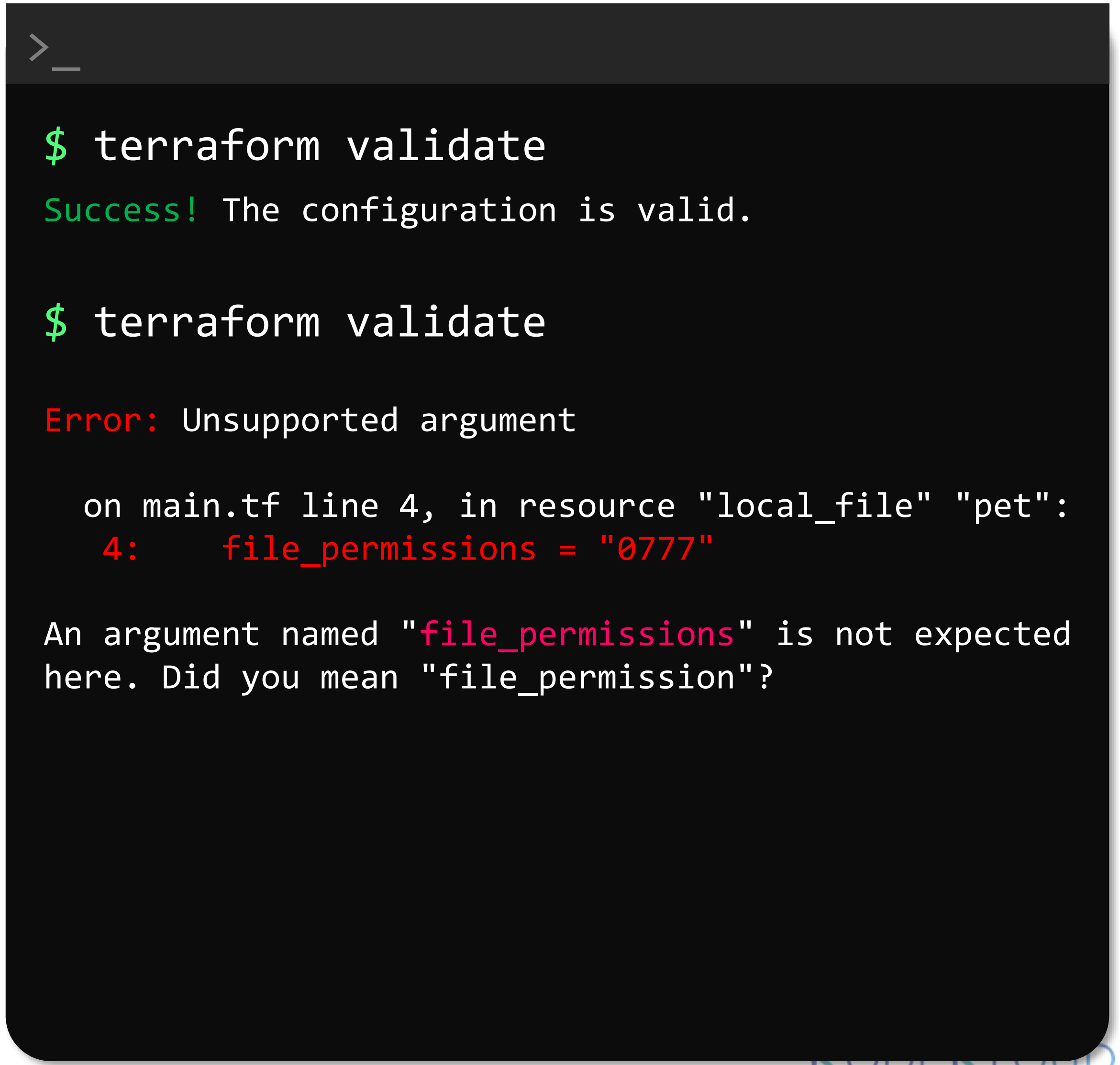
Terraform Commands

terraform validate



main.tf

```
resource "local_file" "pet" {
   filename = "/root/pets.txt"
   content = "We love pets!"
   file_permissions = "0700"
}
```



```
$ terraform validate
Success! The configuration is valid.

$ terraform validate

Error: Unsupported argument

  on main.tf line 4, in resource "local_file" "pet":
   4:     file_permissions = "0777"

An argument named "file_permissions" is not expected
here. Did you mean "file_permission"?
```

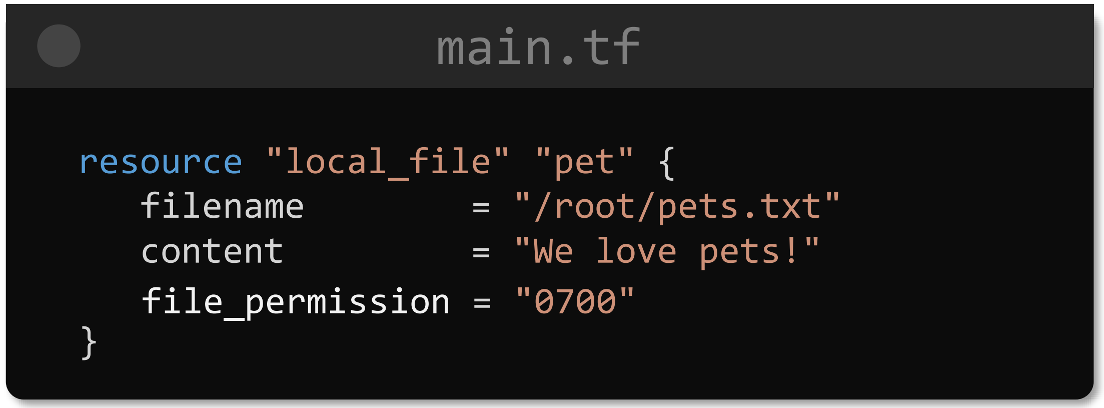
terraform fmt

main.tf

```
resource "local_file" "pet" {
   filename = "/root/pets.txt"
   content = "We love pets!"
   file_permission = "0700"
}
```

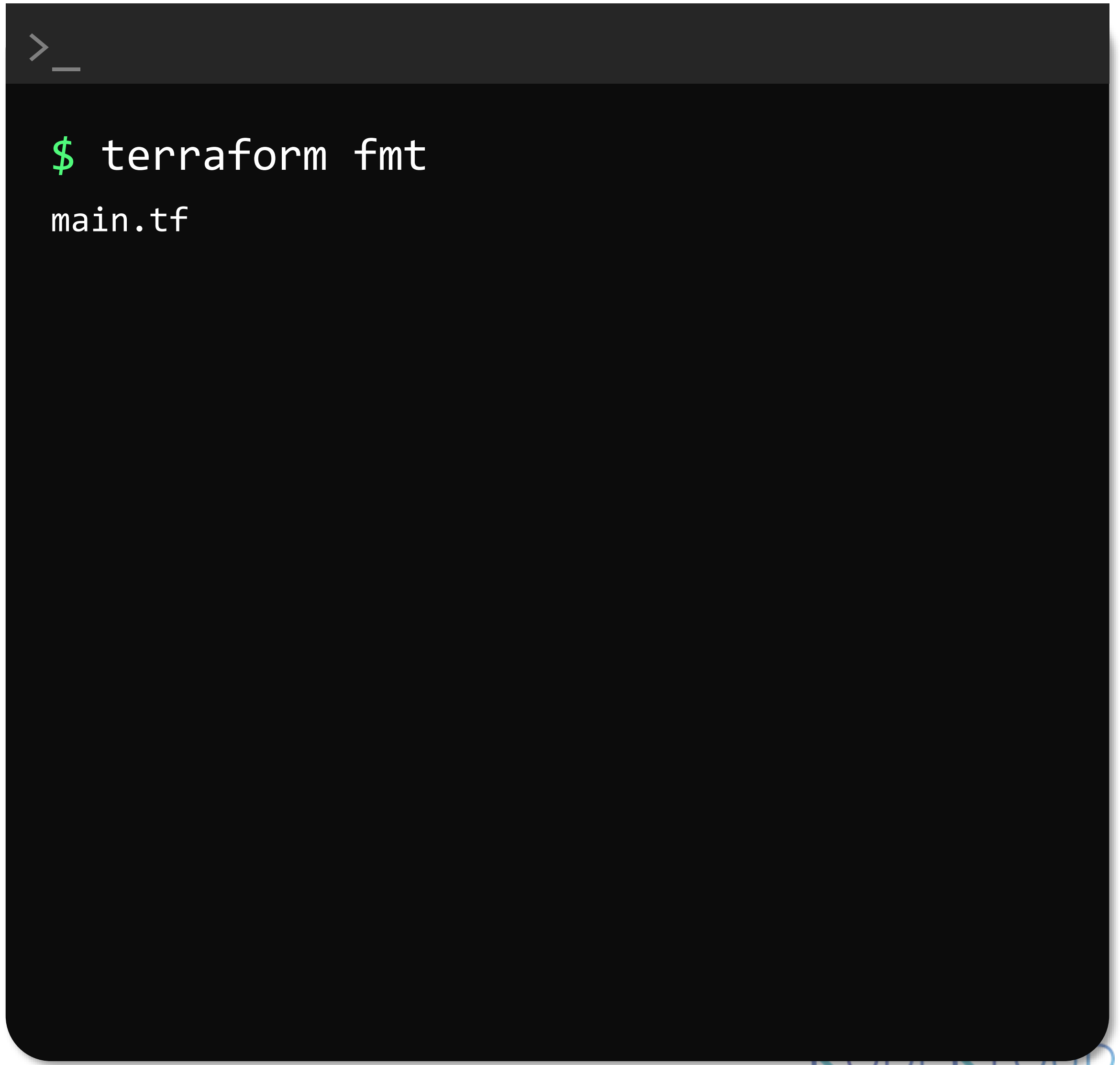
```
$ terraform fmt
```

terraform fmt



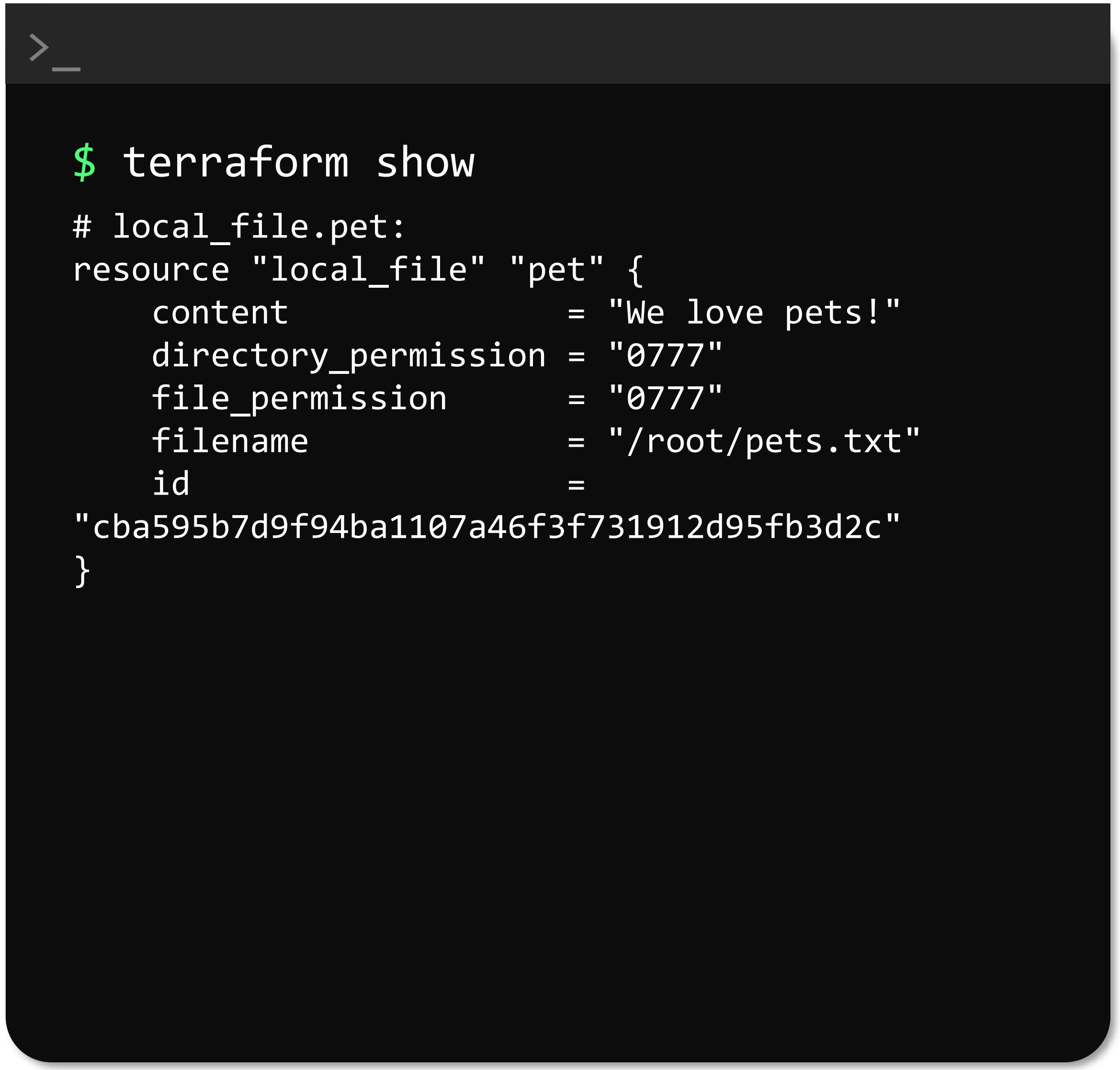
main.tf

```
resource "local_file" "pet" {
   filename        = "/root/pets.txt"
   content         = "We love pets!"
   file_permission = "0700"
}
```

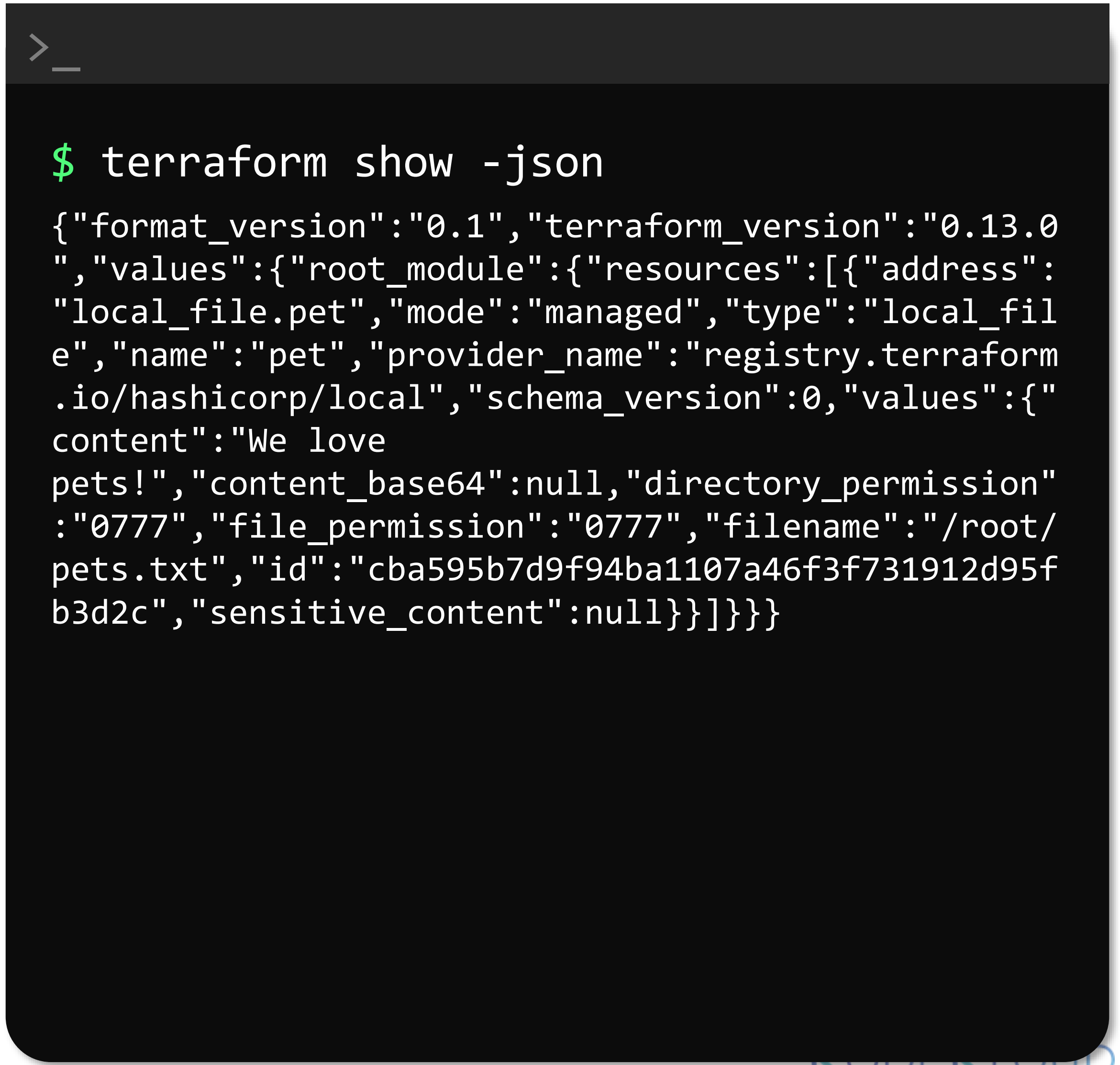


```
$ terraform fmt
main.tf
```

terraform show

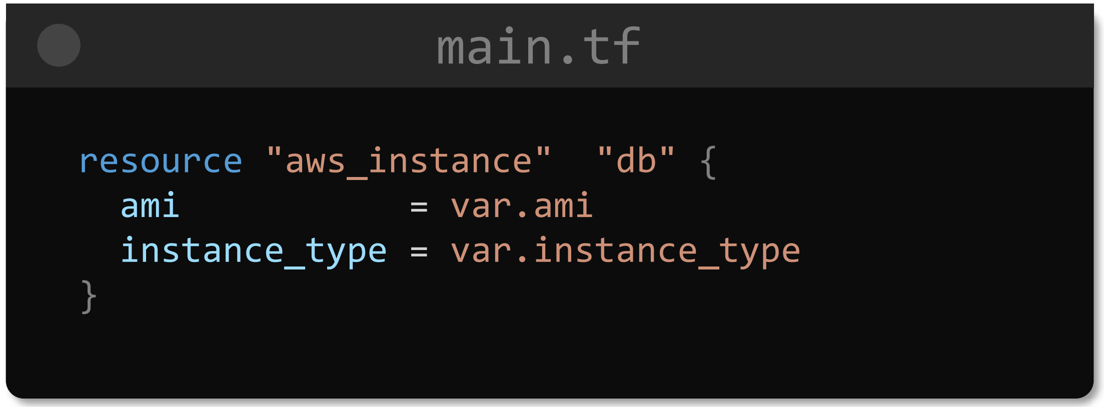


```
$ terraform show
# local_file.pet:
resource "local_file" "pet" {
    content              = "We love pets!"
    directory_permission = "0777"
    file_permission      = "0777"
    filename             = "/root/pets.txt"
    id                   =
"cba595b7d9f94ba1107a46f3f731912d95fb3d2c"
}
```



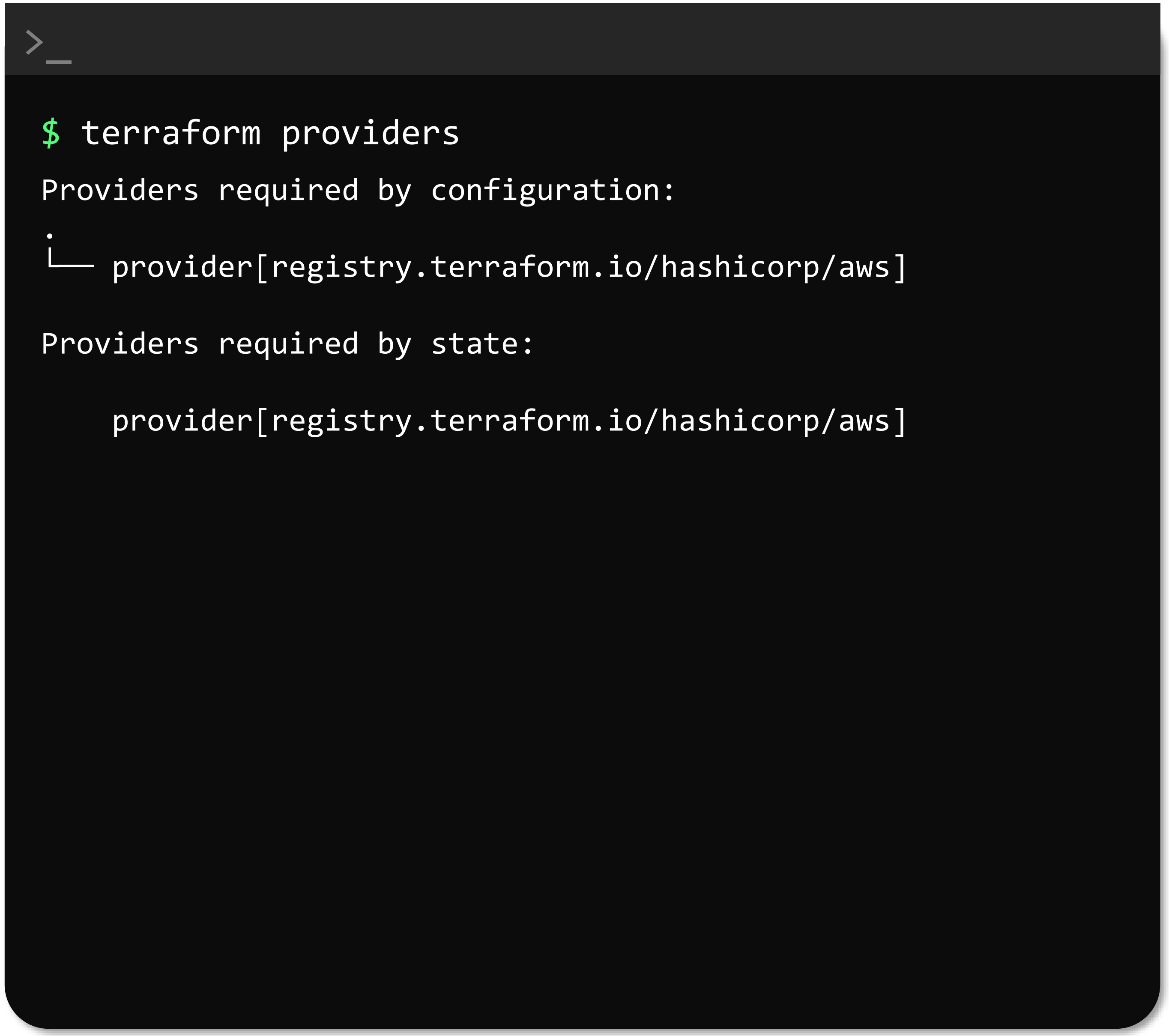
```
$ terraform show -json
{"format_version":"0.1","terraform_version":"0.13.0
","values":{"root_module":{"resources":[{"address":
"local_file.pet","mode":"managed","type":"local_fil
e","name":"pet","provider_name":"registry.terraform
.io/hashicorp/local","schema_version":0,"values":{"
content":"We love
pets!","content_base64":null,"directory_permission"
:"0777","file_permission":"0777","filename":"/root/
pets.txt","id":"cba595b7d9f94ba1107a46f3f731912d95f
b3d2c","sensitive_content":null}}]}}}
```

terraform providers



main.tf

```
resource "aws_instance"  "db" {
  ami           = var.ami
  instance_type = var.instance_type
}
```



```
$ terraform providers
Providers required by configuration:
.
└── provider[registry.terraform.io/hashicorp/aws]

Providers required by state:

    provider[registry.terraform.io/hashicorp/aws]
```

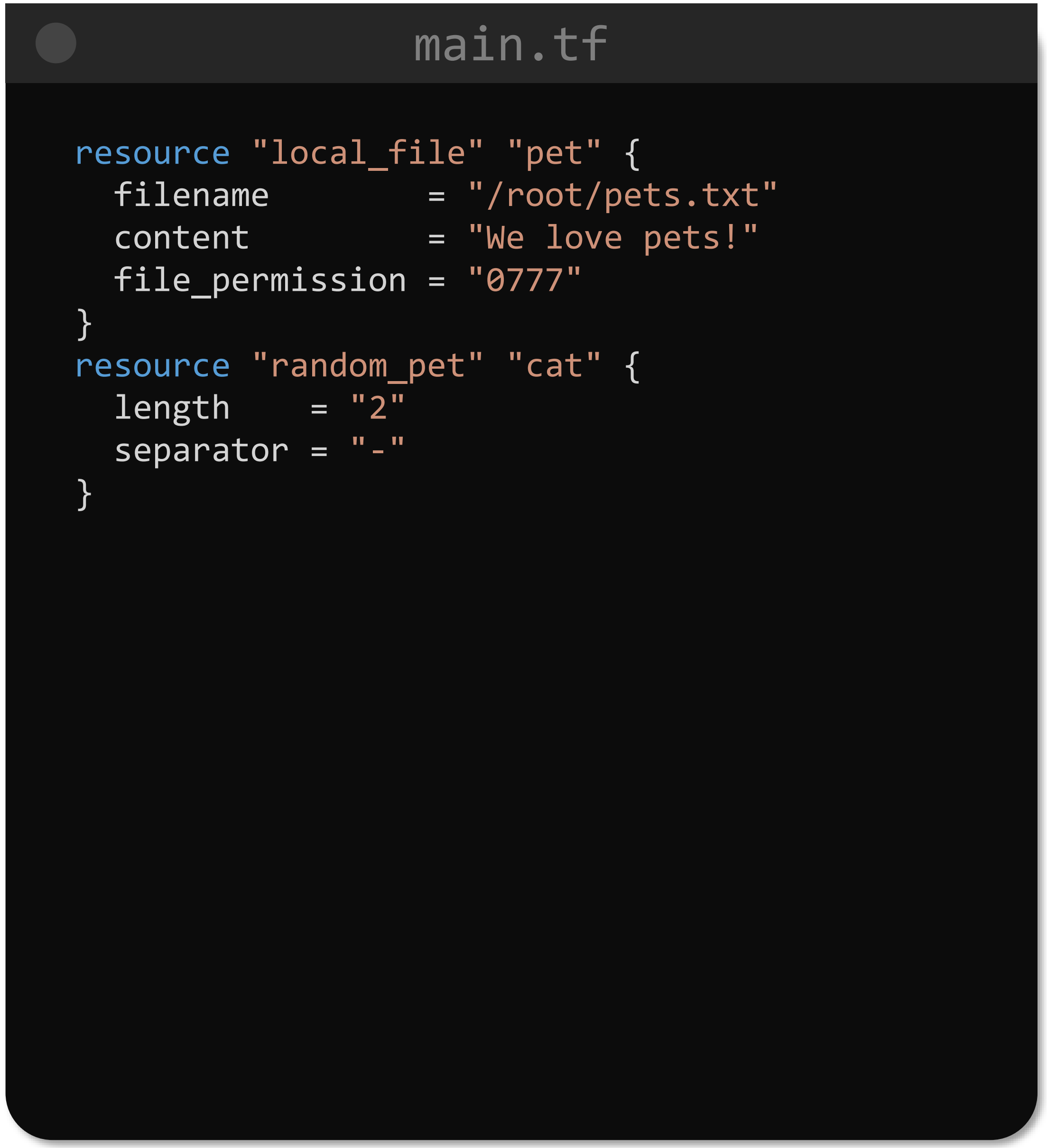
terraform output

main.tf

```hcl
resource "local_file" "pet" {
  filename        = "/root/pets.txt"
  content         = "We love pets!"
  file_permission = "0777"
}
resource "random_pet" "cat" {
  length    = "2"
  separator = "-"
}
output content {
  value       = local_file.pet.content
  sensitive   = false
  description = "Print the content of the file"

}
output pet-name {
  value       = random_pet.cat.id
  sensitive   = false
  description = "Print the name of the pet"

}
```

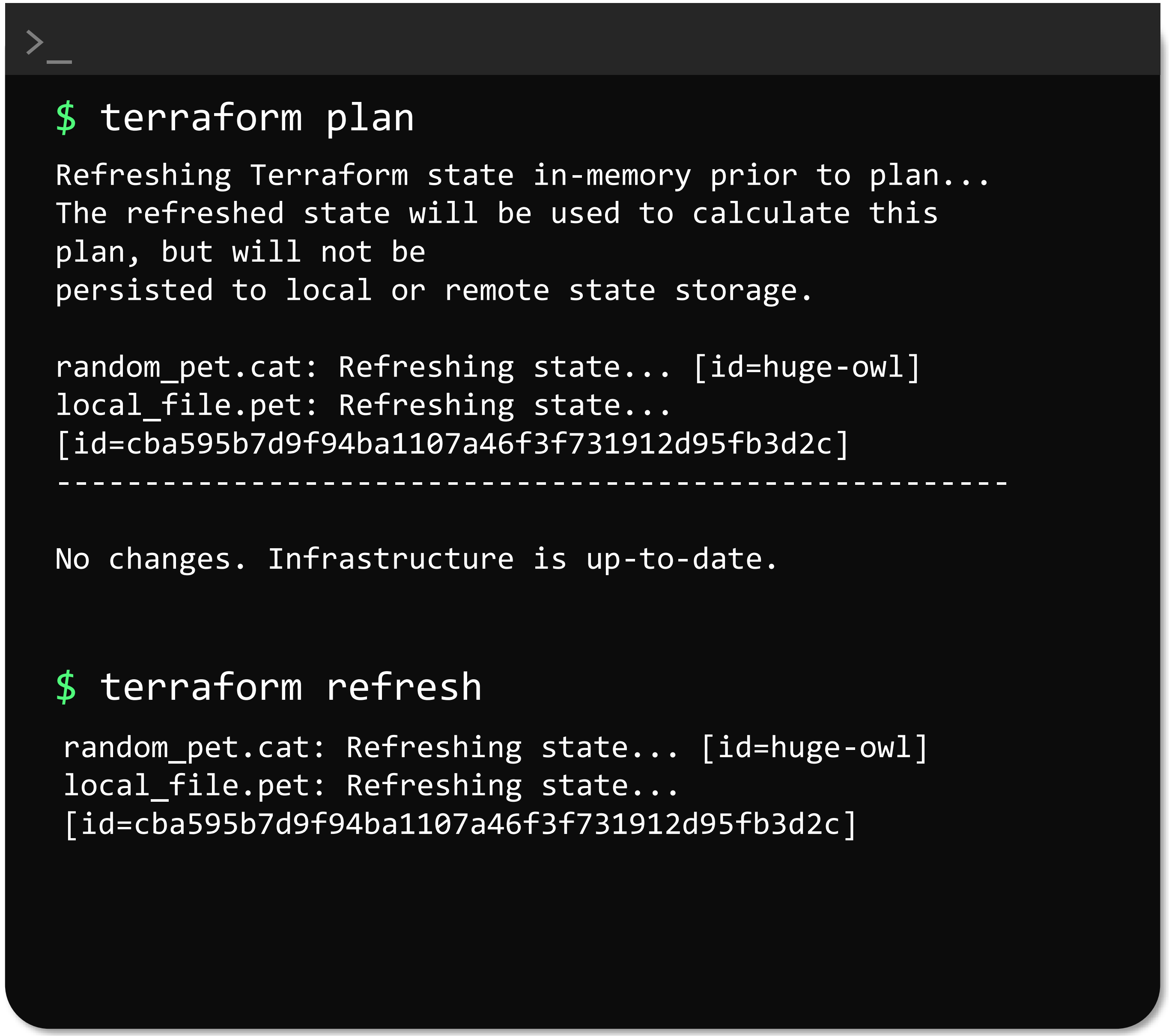
```
>_

$ terraform output
content = We love pets!
pet-name = huge-owl



$ terraform output pet-name
pet-name = huge-owl
```

terraform refresh



main.tf

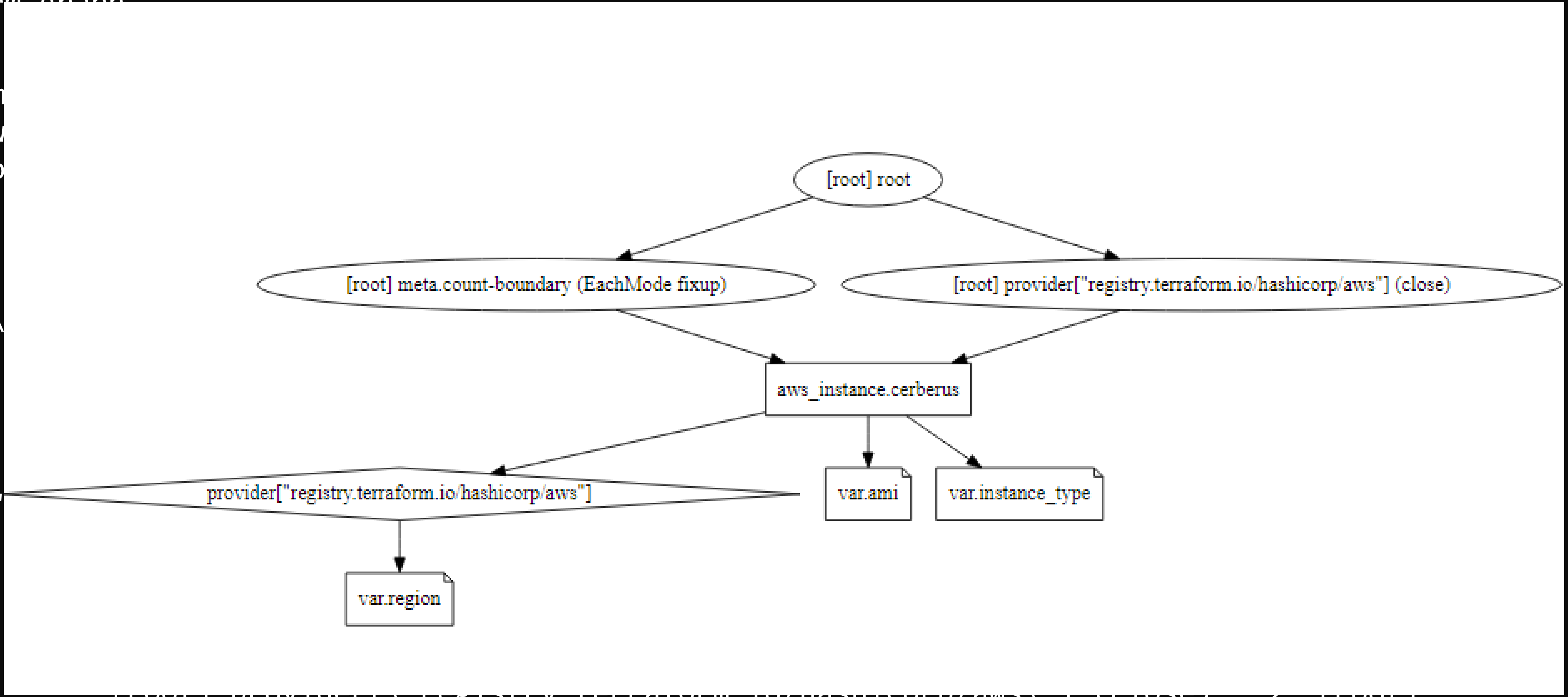
```
resource "local_file" "pet" {
  filename        = "/root/pets.txt"
  content         = "We love pets!"
  file_permission = "0777"
}
resource "random_pet" "cat" {
  length    = "2"
  separator = "-"
}
```



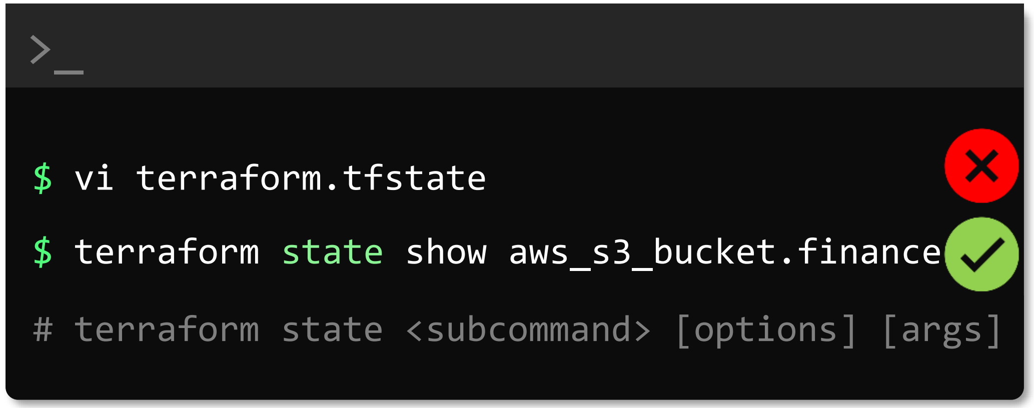
```
>_
$ terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this
plan, but will not be
persisted to local or remote state storage.

random_pet.cat: Refreshing state... [id=huge-owl]
local_file.pet: Refreshing state...
[id=cba595b7d9f94ba1107a46f3f731912d95fb3d2c]
------------------------------------------------------

No changes. Infrastructure is up-to-date.


$ terraform refresh
 random_pet.cat: Refreshing state... [id=huge-owl]
 local_file.pet: Refreshing state...
 [id=cba595b7d9f94ba1107a46f3f731912d95fb3d2c]
```

terraform graph

```
$ terraform graph
digraph {
        com
        new
        sub

"box"]

"provider[\

provider[\"

(expand)"
                 [root] provider[\"registry.terraform.io/hashicorp/aws\"] (close)" -> "[root]
aws_instance.cerberus (expand)"
                "[root] provider[\"registry.terraform.io/hashicorp/aws\"]" -> "[root] var.region"
                "[root] root" -> "[root] meta.count-boundary (EachMode fixup)"
                "[root] root" -> "[root] provider[\"registry.terraform.io/hashicorp/aws\"] (close)"
        }
}
```

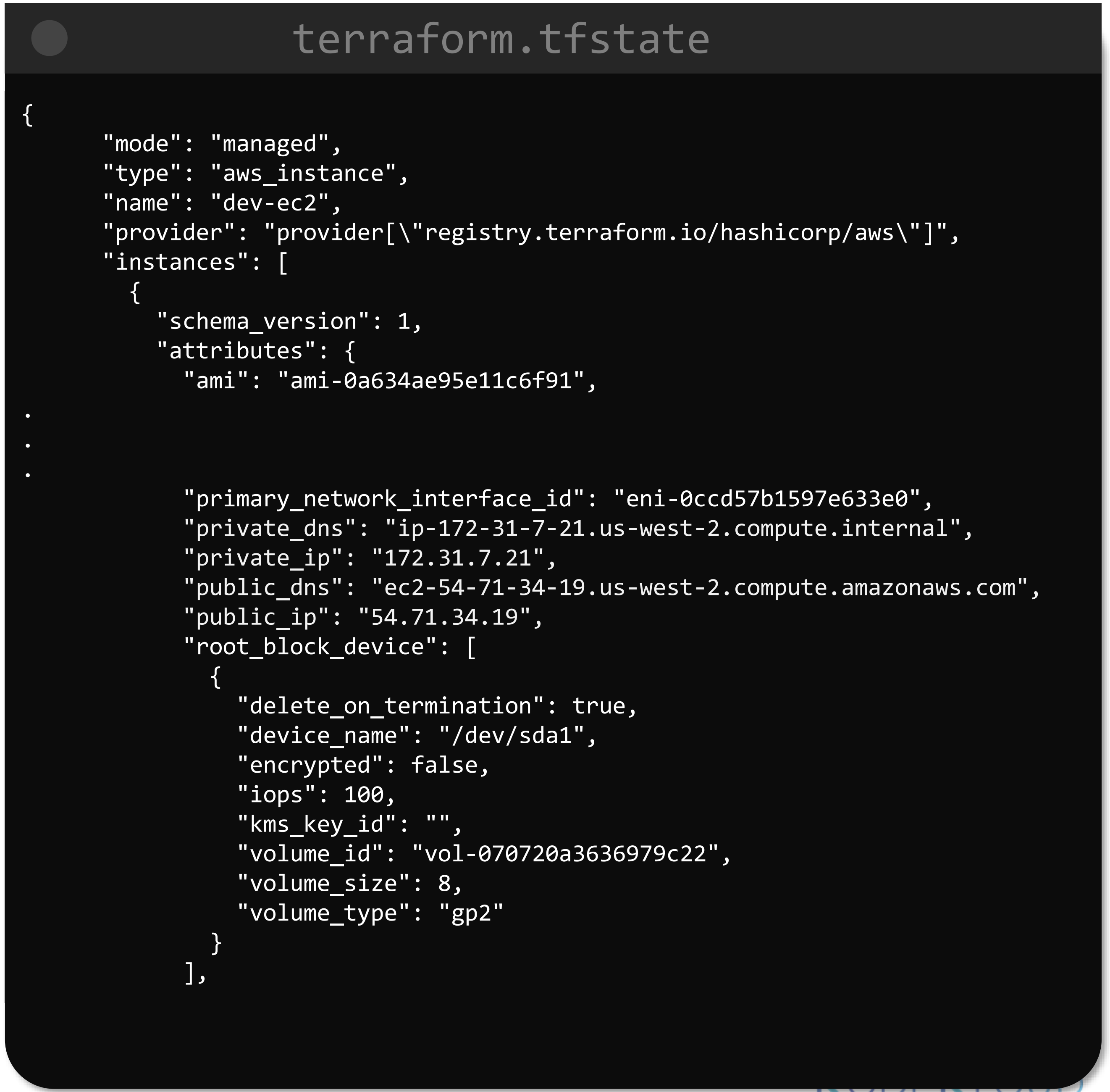


[root] root
[root] meta.count-boundary (EachMode fixup)
[root] provider["registry.terraform.io/hashicorp/aws"] (close)
aws_instance.cerberus
provider["registry.terraform.io/hashicorp/aws"]
var.ami
var.instance_type
var.region



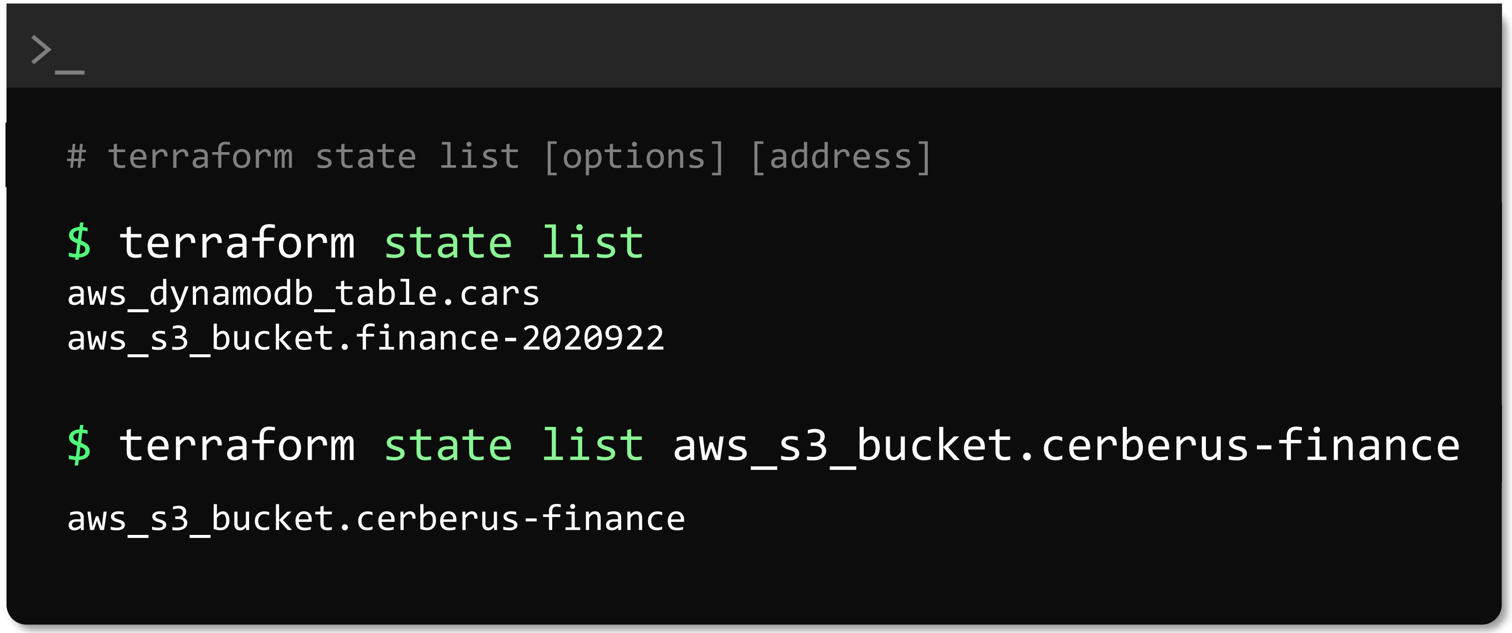
```
>_

$ vi terraform.tfstate                              ✗

$ terraform state show aws_s3_bucket.finance       ✓

# terraform state <subcommand> [options] [args]
```

Sub-command
list
mv
pull
rm
show
push



terraform.tfstate

```
{
      "mode": "managed",
      "type": "aws_instance",
      "name": "dev-ec2",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-0a634ae95e11c6f91",
.
.
.
            "primary_network_interface_id": "eni-0ccd57b1597e633e0",
            "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
            "private_ip": "172.31.7.21",
            "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
            "public_ip": "54.71.34.19",
            "root_block_device": [
              {
                "delete_on_termination": true,
                "device_name": "/dev/sda1",
                "encrypted": false,
                "iops": 100,
                "kms_key_id": "",
                "volume_id": "vol-070720a3636979c22",
                "volume_size": 8,
                "volume_type": "gp2"
              }
            ],
```



```
>_

# terraform state list [options] [address]

$ terraform state list
aws_dynamodb_table.cars
aws_s3_bucket.finance-2020922

$ terraform state list aws_s3_bucket.cerberus-finance

aws_s3_bucket.cerberus-finance
```

```
# terraform state show [options] [address]

$ terraform state show aws_s3_bucket.cerberus-finance

resource "aws_s3_bucket" "terraform-state" {
    acl                         = "private"
    arn                         = "arn:aws:s3:::cerberus-finance"
    bucket                      = "cerberus-finance"
    bucket_domain_name          = "cerberus-finance.s3.amazonaws.com"
    bucket_regional_domain_name = "cerberus-finance.s3.us-west-1.amazonaws.com"
    force_destroy               = false
    hosted_zone_id              = "Z2F5ABCDE1ACD"
    id                          = "cerberus-finance"
    region                      = "us-west-1"
    request_payer               = "BucketOwner"
    tags                        = {
        "Description" = "Bucket to store Finance and Payroll Information"
    }

    versioning {
        enabled    = false
        mfa_delete = false
    }
}
```

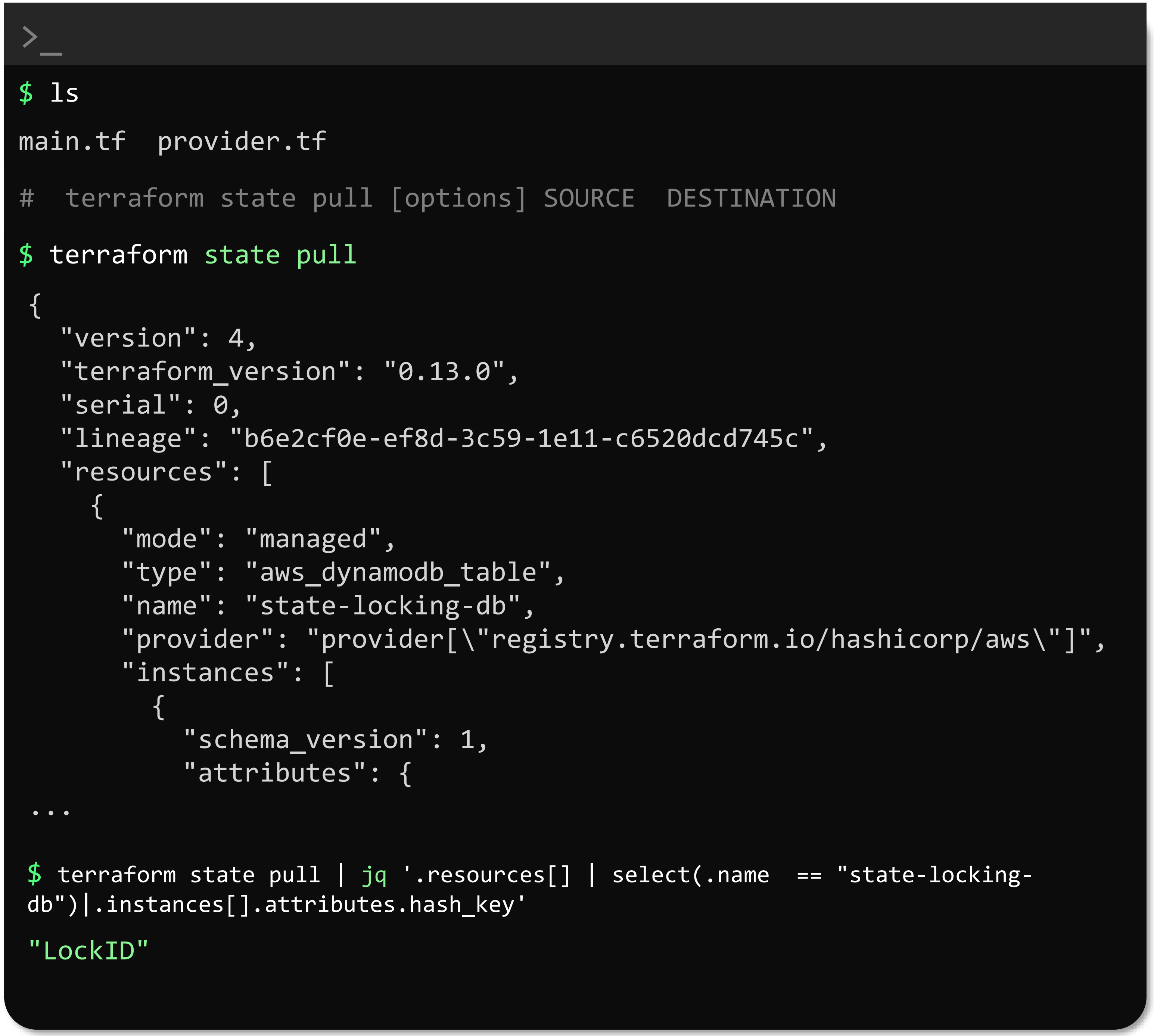
main.tf

```
resource "aws_dynamodb_table" "state-locking-db"
    name = "state-locking"
    billing_mode = "PAY_PER_REQUEST"
    hash_key = "LockID"
    attribute {
        name =  "LockID"
        type = "S"
    }
}
```

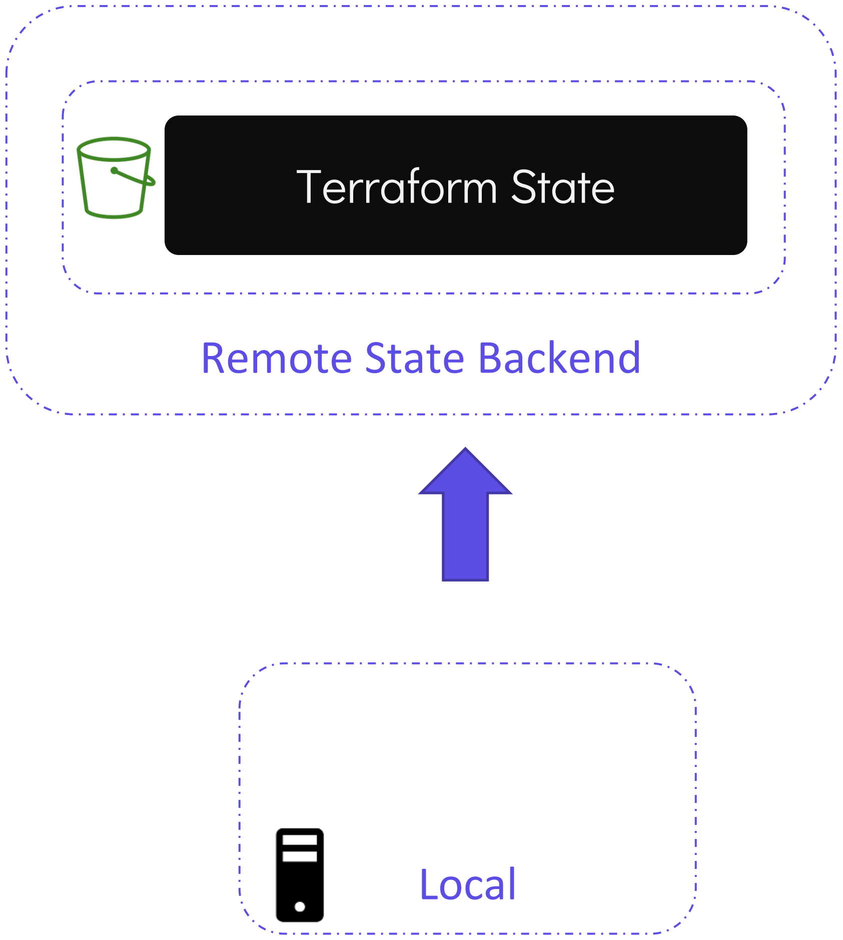
terraform.tfstate

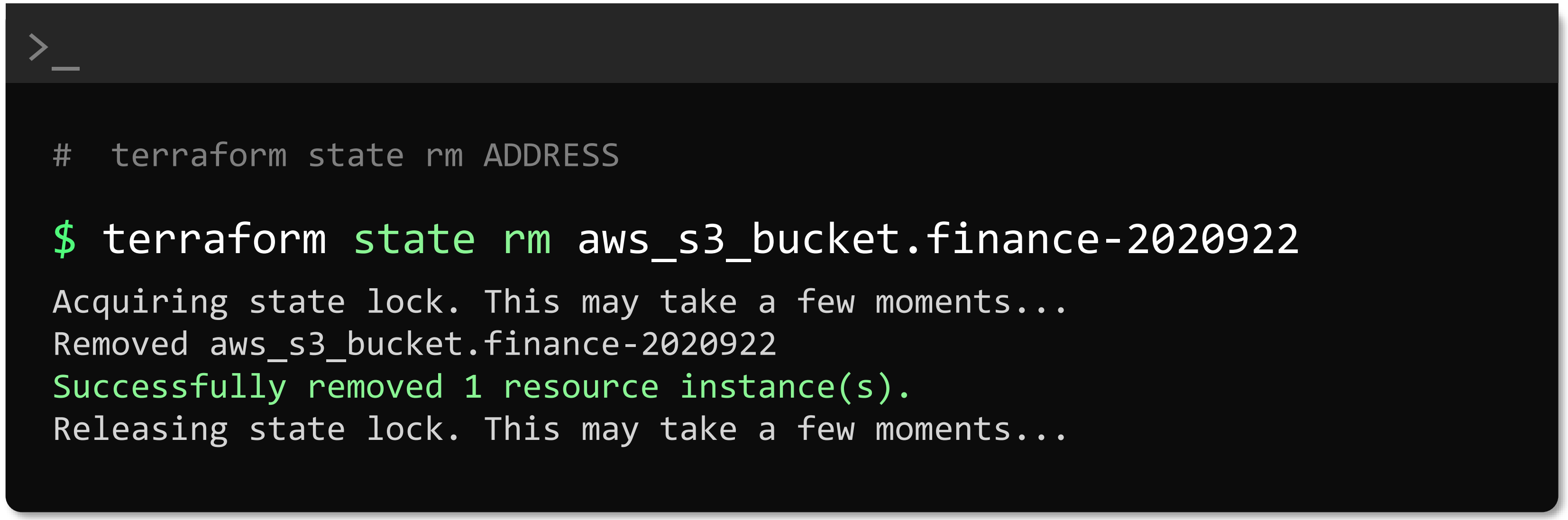
```
"resources": [
    {
      "mode": "managed",
      "type": "aws_dynamodb_table",
      "name": "state-locking-db"
      "provider":
"provider[\"registry.terraform.io/hashicorp/aws\"
]",
.
.
```

```
>_

# terraform state mv [options] SOURCE  DESTINATION

$ terraform state mv aws_dynamodb_table.state-locking aws_dynamodb_table.state-locking-db
Move "aws_dynamodb_table.state-locking" to "aws_dynamodb_table.state-locking-db"
Successfully moved 1 object(s).

$ terraform apply
                     Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
```

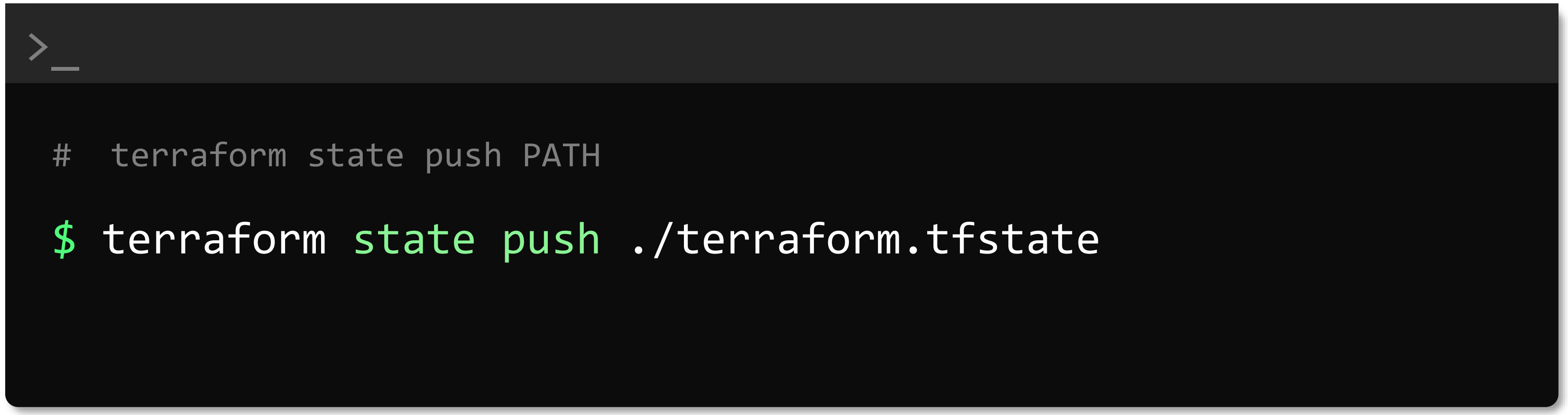


```
$ ls
main.tf  provider.tf

#  terraform state pull [options] SOURCE  DESTINATION

$ terraform state pull
 {
   "version": 4,
   "terraform_version": "0.13.0",
   "serial": 0,
   "lineage": "b6e2cf0e-ef8d-3c59-1e11-c6520dcd745c",
   "resources": [
     {
       "mode": "managed",
       "type": "aws_dynamodb_table",
       "name": "state-locking-db",
       "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
       "instances": [
         {
           "schema_version": 1,
           "attributes": {
 ...

$ terraform state pull | jq '.resources[] | select(.name  == "state-locking-
db")|.instances[].attributes.hash_key'
"LockID"
```

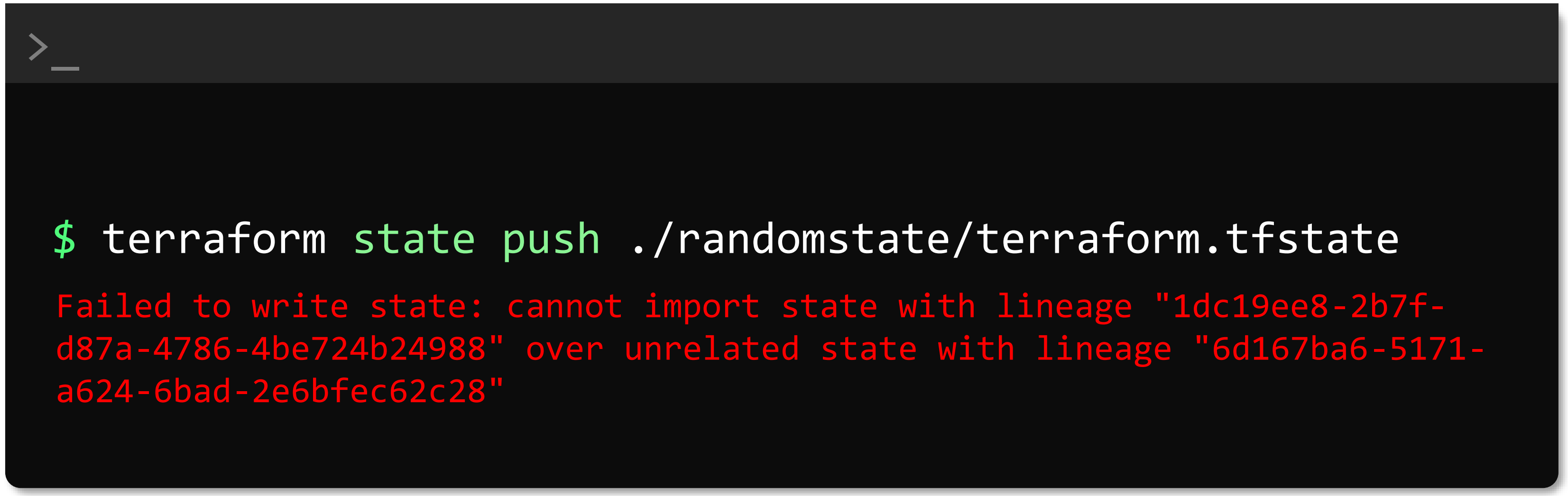




```
#  terraform state rm ADDRESS

$ terraform state rm aws_s3_bucket.finance-2020922
Acquiring state lock. This may take a few moments...
Removed aws_s3_bucket.finance-2020922
Successfully removed 1 resource instance(s).
Releasing state lock. This may take a few moments...
```



```
#  terraform state push PATH

$ terraform state push ./terraform.tfstate
```



```
$ terraform state push ./randomstate/terraform.tfstate
Failed to write state: cannot import state with lineage "1dc19ee8-2b7f-
d87a-4786-4be724b24988" over unrelated state with lineage "6d167ba6-5171-
a624-6bad-2e6bfec62c28"
```

KODEKLOUD

Lifecycle Rules

create_before_destroy

main.tf

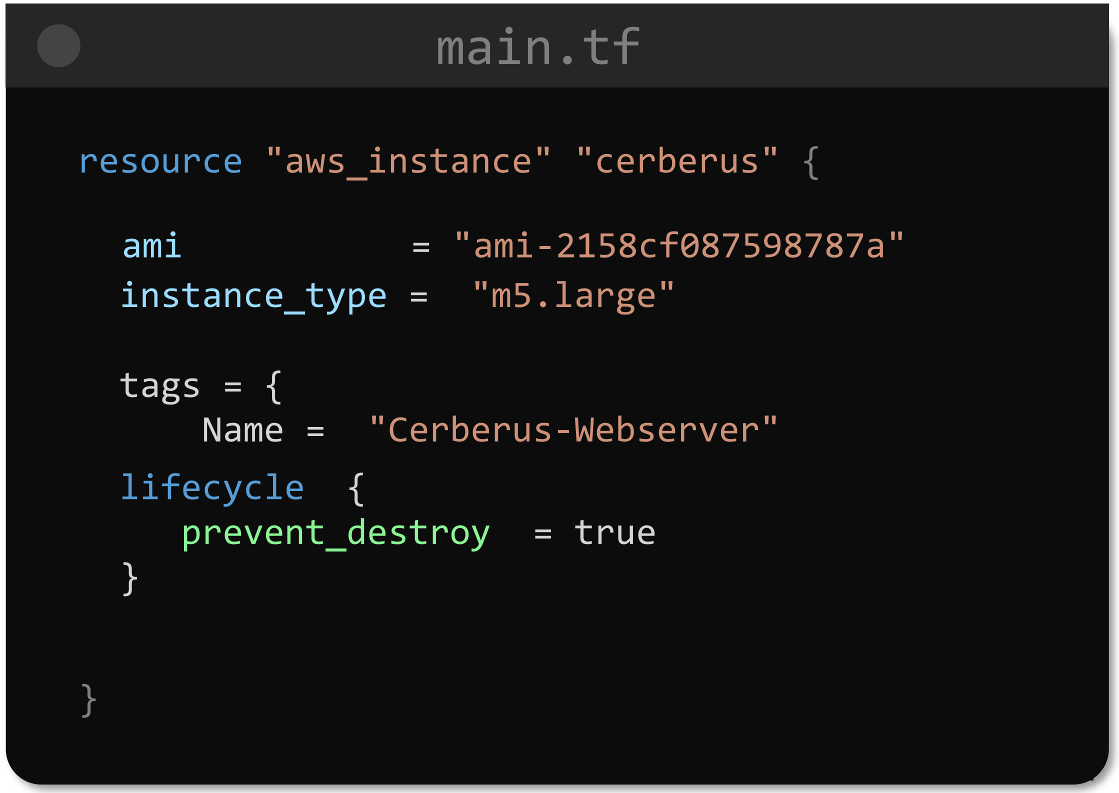
```
resource "aws_instance" "cerberus" {

  ami           = "ami-2158cf087598787a"
  instance_type =  "m5.large“

  tags = {
      Name =  "Cerberus-Webserver"

  lifecycle  {
     create_before_destroy  = true
  }

}
```

>_

```
$ terraform apply

aws_instance.cerberus: Refreshing state... [id=i-a6e22ec530319025

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
+/- create replacement and then destroy

Terraform will perform the following actions:

  # aws_instance.cerberus must be replaced
+/- resource "aws_instance" "cerberus" {
      ~ ami                                       = "ami-
06178cf087598769c" -> "ami-2158cf087598787a" # forces replacement
Plan: 1 to add, 0 to change, 1 to destroy.
...
aws_instance.cerberus: Creating...
aws_instance.cerberus: Still creating... [10s elapsed]
aws_instance.cerberus: Creation complete after 10s [id=i-
477150603640c96f4]
aws_instance.cerberus: Destroying... [id=i-a6e22ec5303190252]
aws_instance.cerberus: Still destroying... [id=i-a6e22ec530319025
10s elapsed]
aws_instance.cerberus: Destruction complete after 10s

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```

ami-06178cf087598769c
ami-2158cf087598787a

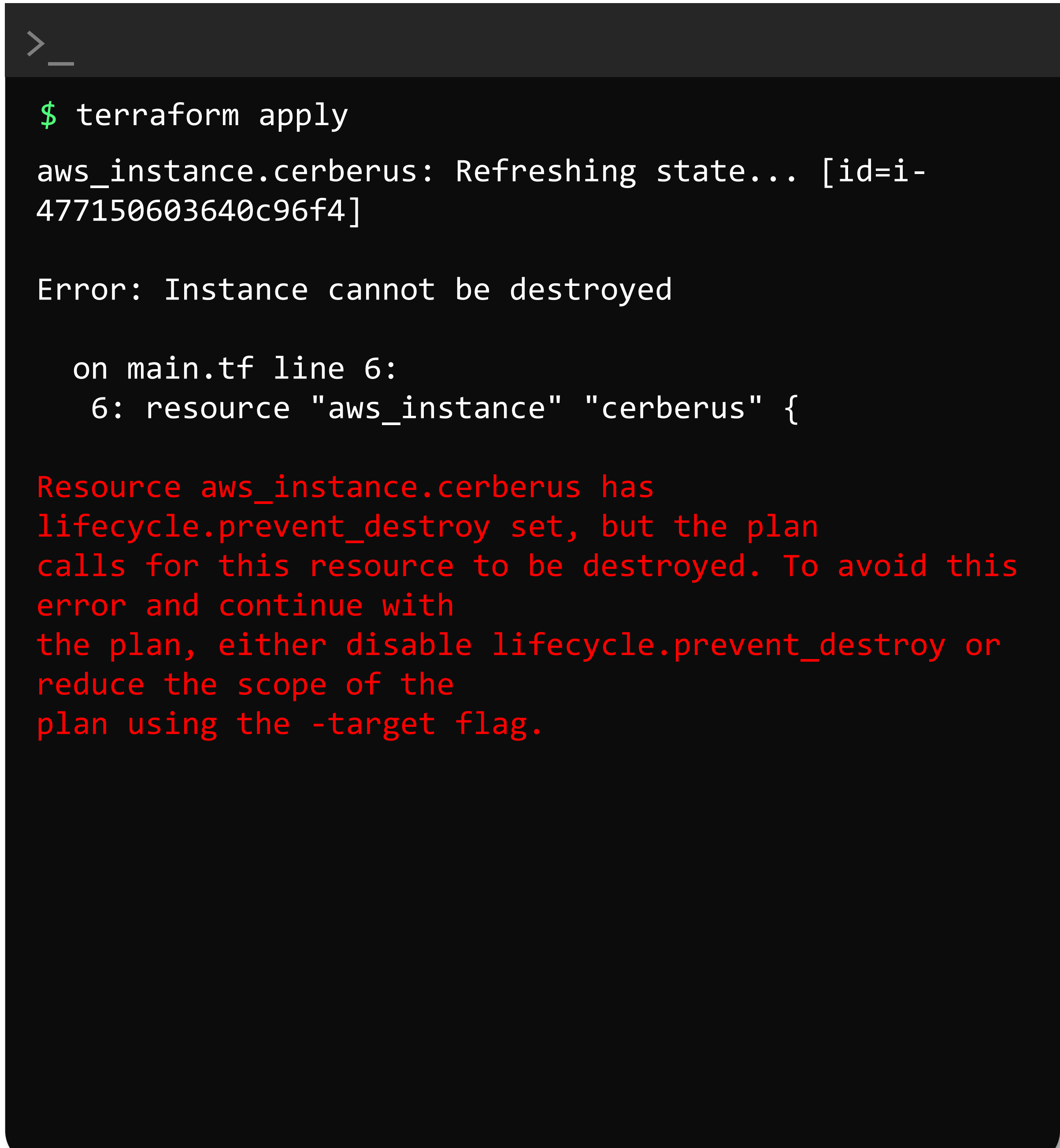
prevent_destroy



main.tf

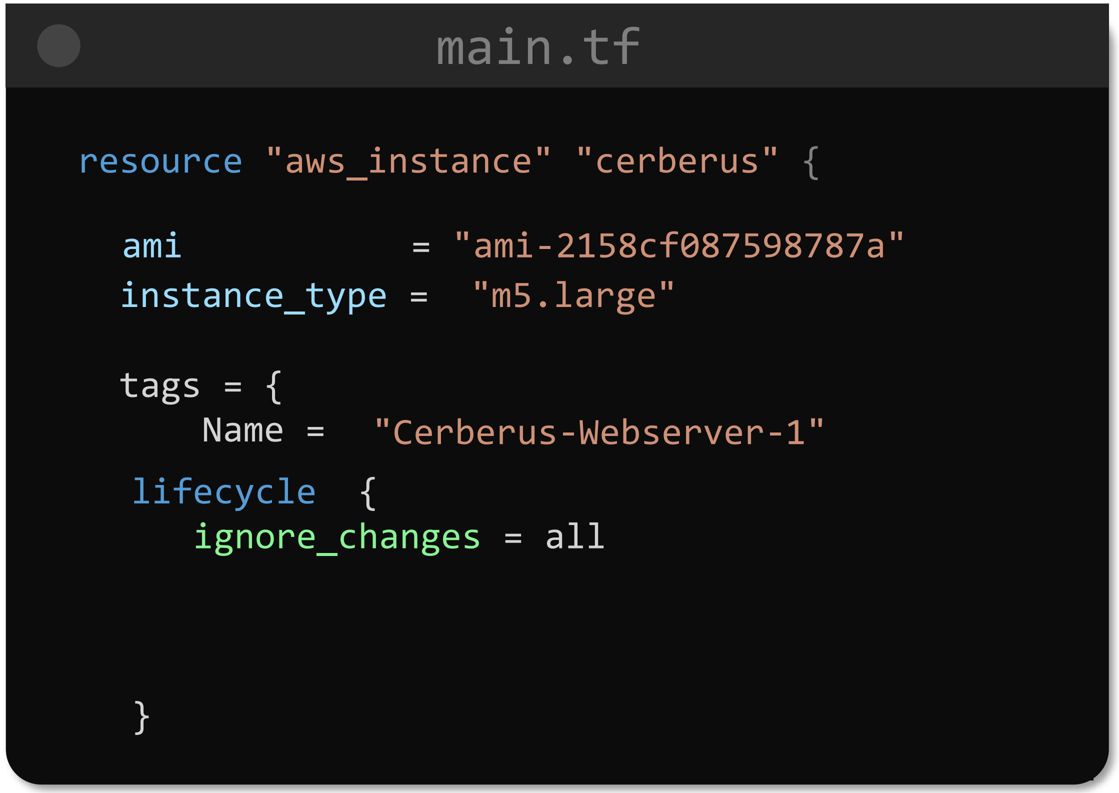
```
resource "aws_instance" "cerberus" {

  ami           = "ami-2158cf087598787a"
  instance_type =  "m5.large"

  tags = {
      Name =  "Cerberus-Webserver"
  lifecycle  {
     prevent_destroy  = true
  }

}
```

ami-2158cf087598787a



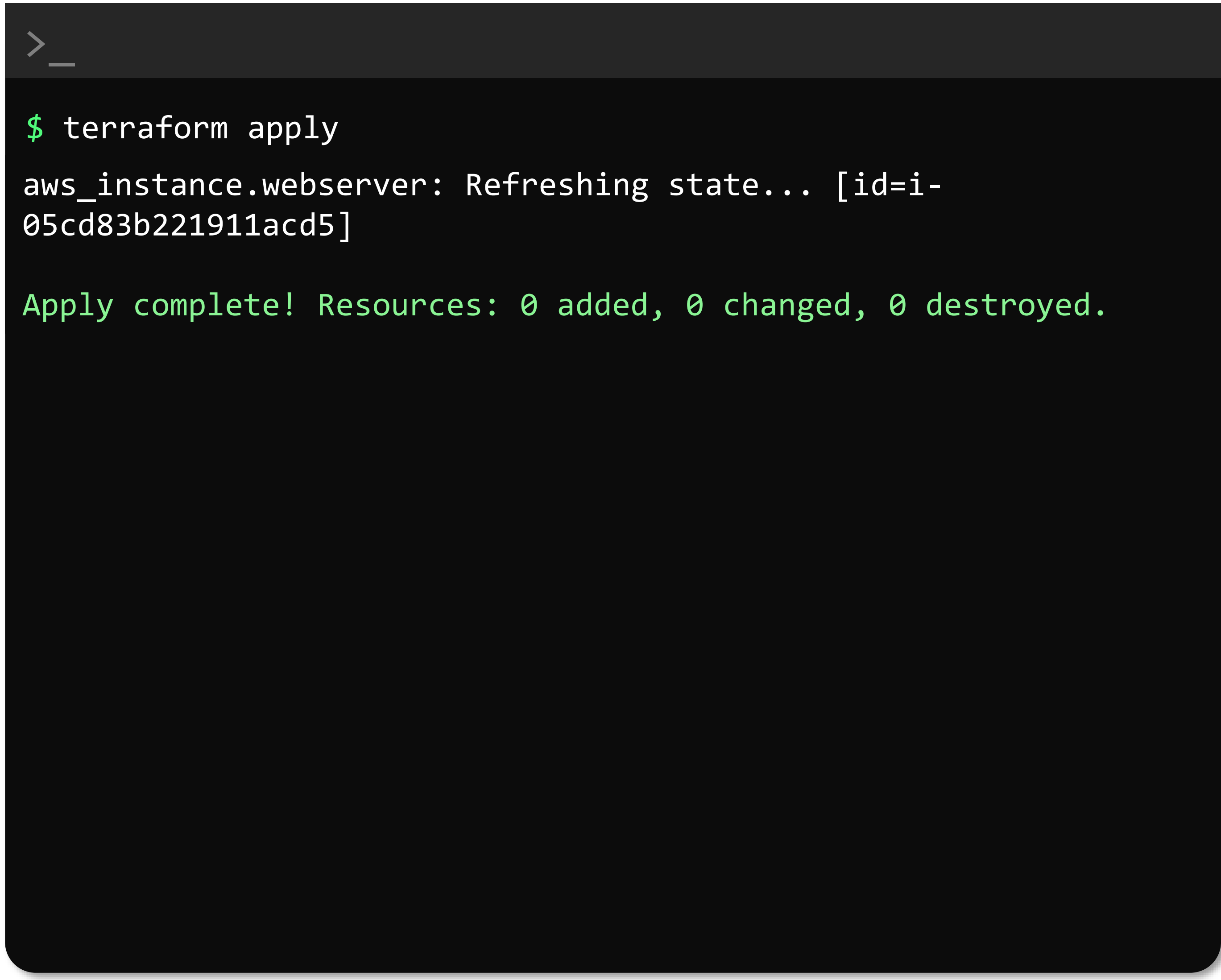
```
$ terraform apply
aws_instance.cerberus: Refreshing state... [id=i-
477150603640c96f4]

Error: Instance cannot be destroyed

  on main.tf line 6:
   6: resource "aws_instance" "cerberus" {

Resource aws_instance.cerberus has
lifecycle.prevent_destroy set, but the plan
calls for this resource to be destroyed. To avoid this
error and continue with
the plan, either disable lifecycle.prevent_destroy or
reduce the scope of the
plan using the -target flag.
```

ignore_changes



main.tf

```
resource "aws_instance" "cerberus" {

  ami           = "ami-2158cf087598787a"
  instance_type =  "m5.large"

  tags = {
      Name =   "Cerberus-Webserver-1"
   lifecycle  {
      ignore_changes = all



   }
```

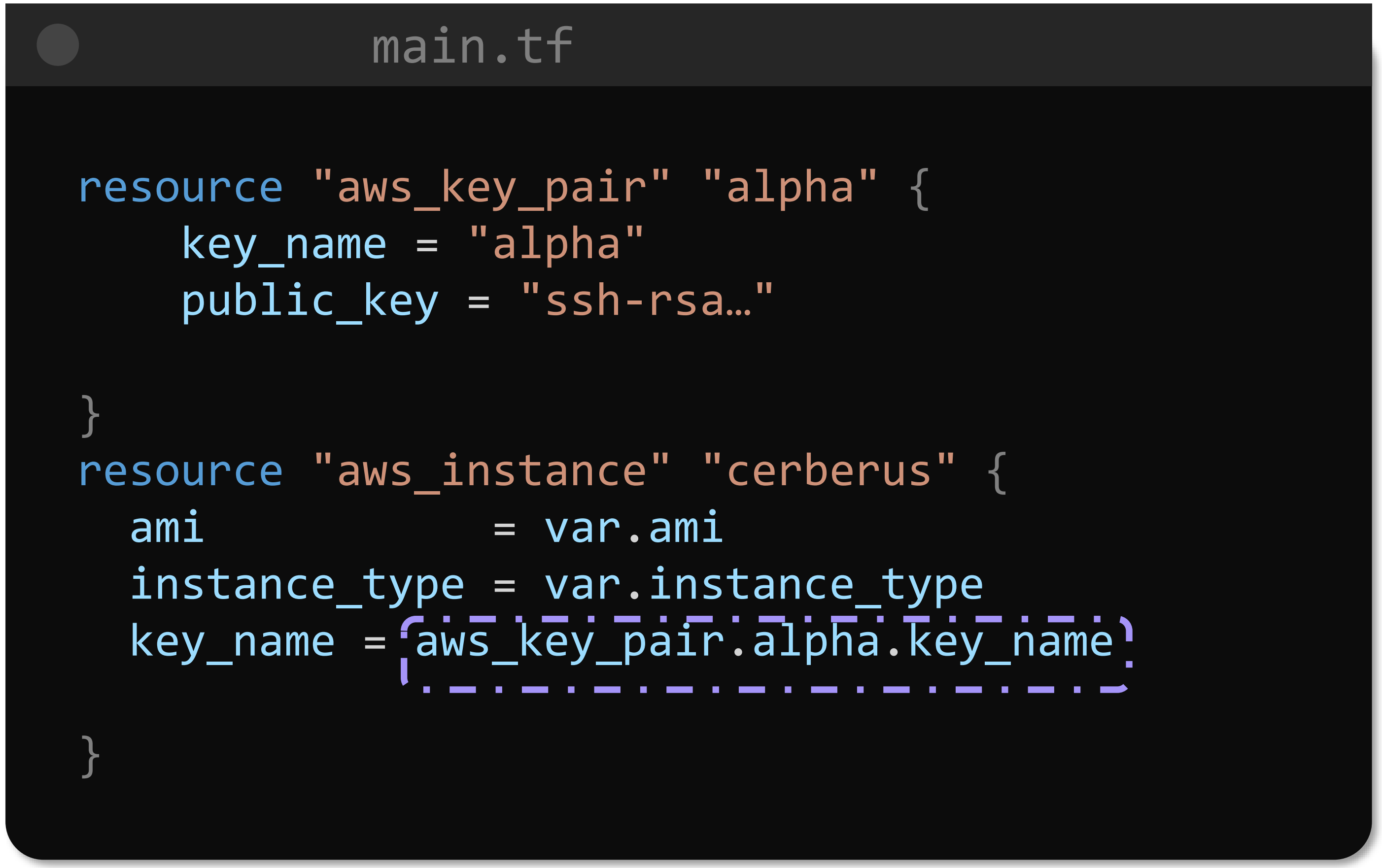


```
$ terraform apply
aws_instance.webserver: Refreshing state... [id=i-
05cd83b221911acd5]

Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
```

Name = Cerberus-Webserver

KODEKLOUD

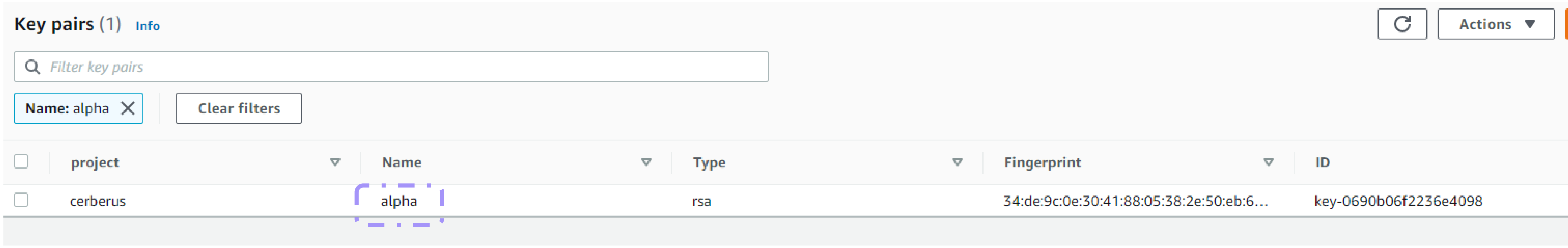
Data Sources



main.tf

```
resource "aws_key_pair" "alpha" {
    key_name = "alpha"
    public_key = "ssh-rsa…"

}
resource "aws_instance" "cerberus" {
  ami           = var.ami
  instance_type = var.instance_type
  key_name = aws_key_pair.alpha.key_name

}
```

https://registry.terraform.io/providers/hashicorp/aws/latest/docs/data-sources/key_pair

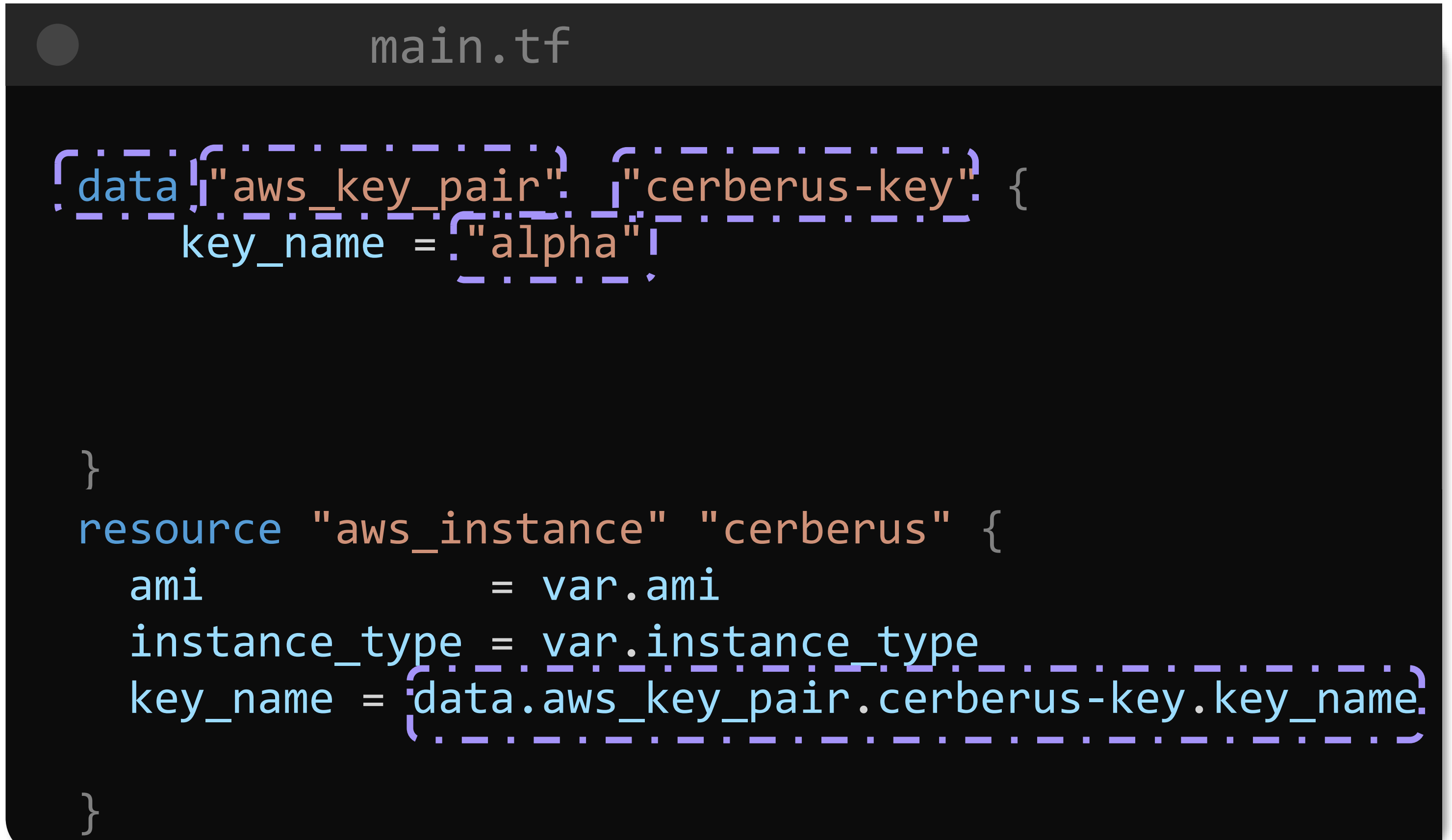


Key pairs (1) Info

Filter key pairs

Name: alpha ✕ | Clear filters

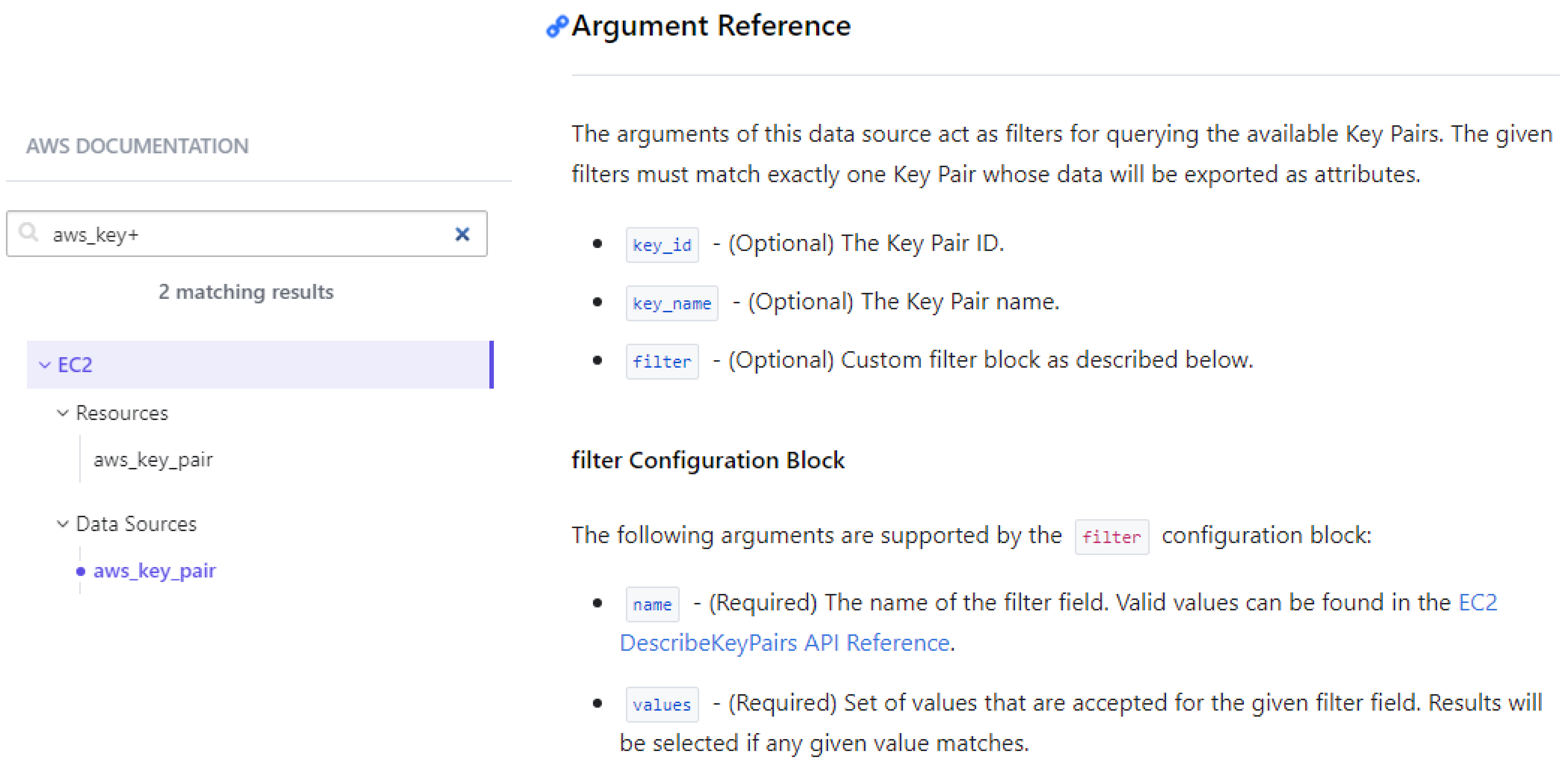
	project	Name	Type	Fingerprint	ID
☐	cerberus	alpha	rsa	34:de:9c:0e:30:41:88:05:38:2e:50:eb:6...	key-0690b06f2236e4098



main.tf

```
data "aws_key_pair" "cerberus-key" {
    key_name = "alpha"



}
resource "aws_instance" "cerberus" {
  ami           = var.ami
  instance_type = var.instance_type
  key_name = data.aws_key_pair.cerberus-key.key_name

}
```

https://registry.terraform.io/providers/hashicorp/aws/latest/docs/data-sources/key_pair



AWS DOCUMENTATION

aws_key+

2 matching results

EC2

Resources

aws_key_pair

Data Sources

aws_key_pair

Argument Reference

The arguments of this data source act as filters for querying the available Key Pairs. The given filters must match exactly one Key Pair whose data will be exported as attributes.

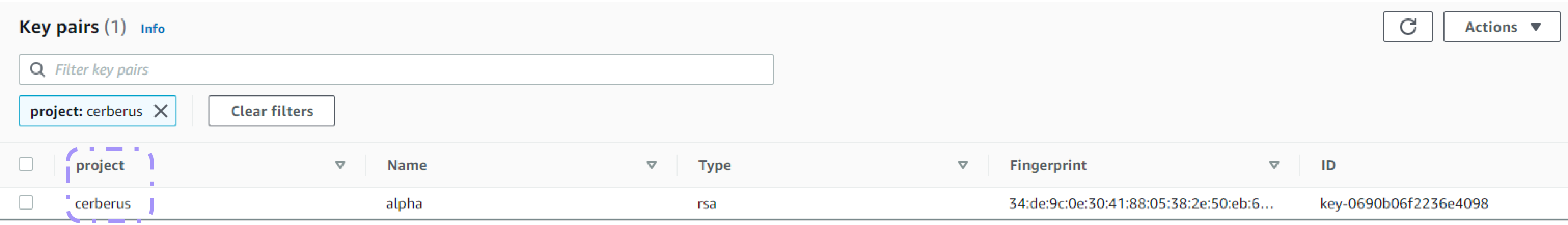
- `key_id` - (Optional) The Key Pair ID.
- `key_name` - (Optional) The Key Pair name.
- `filter` - (Optional) Custom filter block as described below.

filter Configuration Block

The following arguments are supported by the `filter` configuration block:

- `name` - (Required) The name of the filter field. Valid values can be found in the EC2 DescribeKeyPairs API Reference.
- `values` - (Required) Set of values that are accepted for the given filter field. Results will be selected if any given value matches.

https://registry.terraform.io/providers/hashicorp/aws/latest/docs/data-sources/key_pair

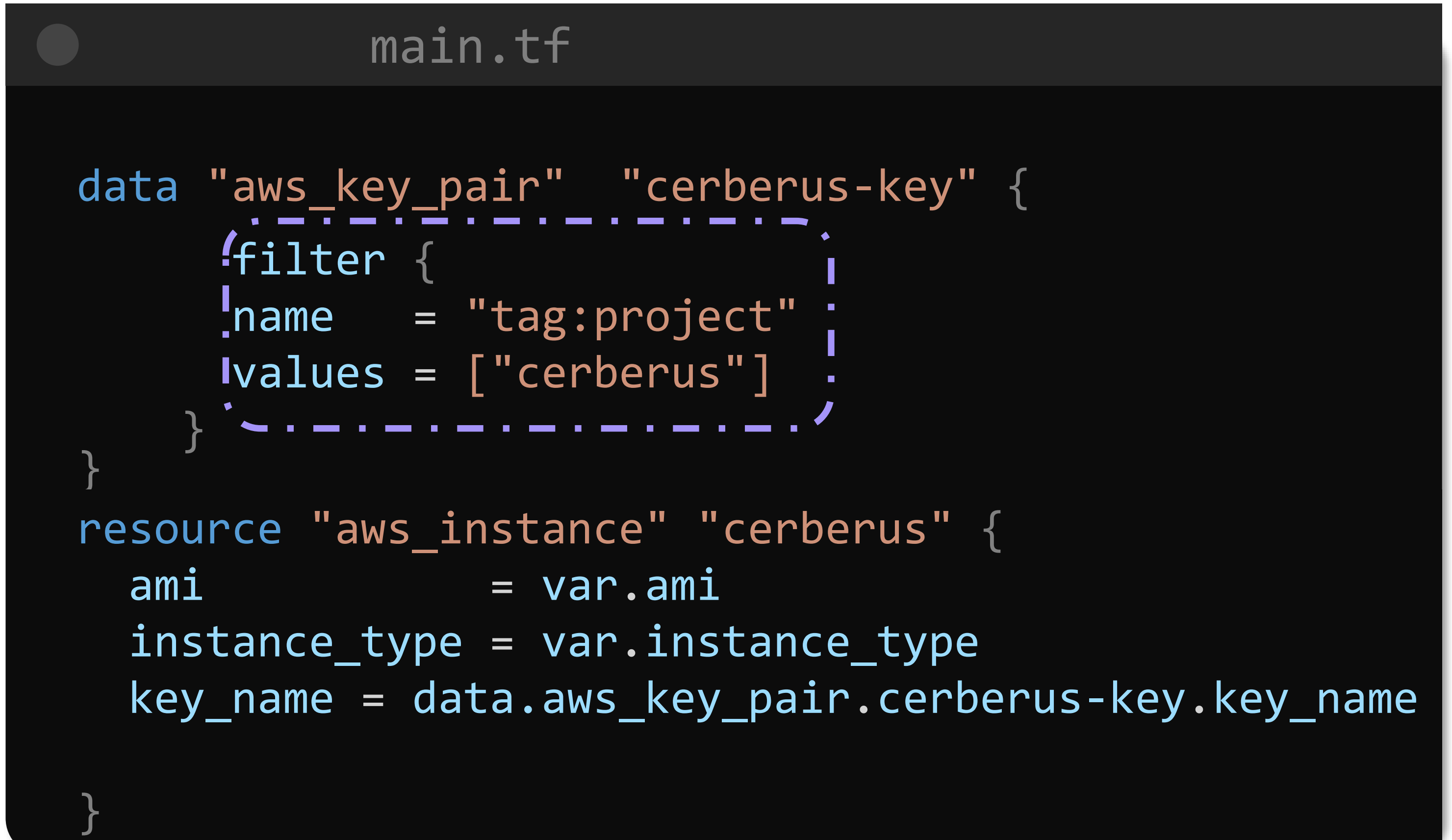


Key pairs (1) Info

Filter key pairs

project: cerberus | Clear filters

project	Name	Type	Fingerprint	ID
cerberus	alpha	rsa	34:de:9c:0e:30:41:88:05:38:2e:50:eb:6...	key-0690b06f2236e4098



main.tf

```
data "aws_key_pair"  "cerberus-key" {
      filter {
      name    = "tag:project"
      values = ["cerberus"]
    }
}
resource "aws_instance" "cerberus" {
  ami               = var.ami
  instance_type = var.instance_type
  key_name = data.aws_key_pair.cerberus-key.key_name

}
```

terraform.tfstate

Resource	Data Source
Keyword: **resource**	Keyword: **data**
Creates, Updates, Destroys Infrastructure	Only **Reads** Infrastructure
Also called **Managed Resources**	Also called **Data Resources**

KODEKLOUD

count and for-each

count

main.tf

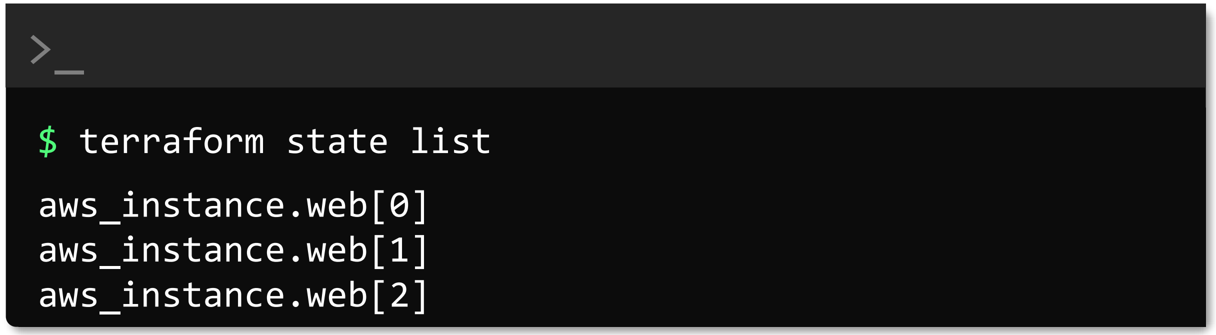
```
resource "aws_instance"  "web" {
  ami           = var.ami
  instance_type = var.instance_type
   count    = 3

}
```

variables.tf

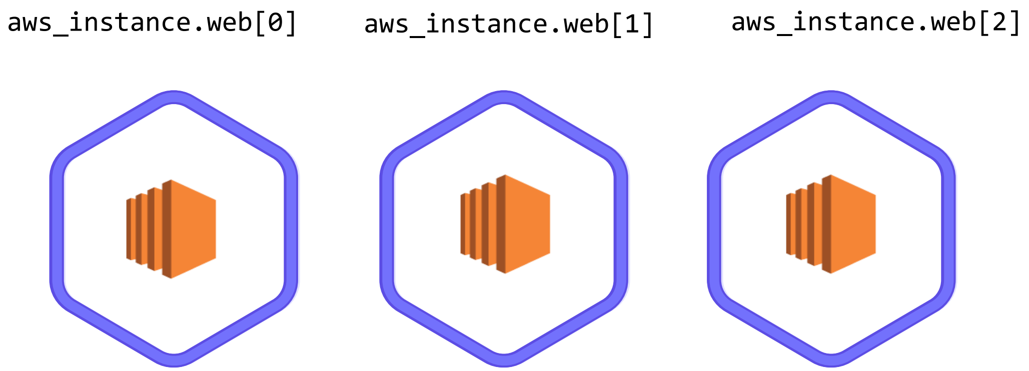
```
variable "ami" {
  default = "ami-06178cf087598769c"
}
variable "instance_type" {
  default = "m5.large"
}
```

```
>_

$ terraform apply
[Output Truncated]
Terraform will perform the following actions:
...
# # aws_instance.web[2] will be created
.
 + volume_size             = (known after apply)
          + volume_type             = (known after apply)
        }
    }

Plan: 3 to add, 0 to change, 0 to destroy.
```

count



```
$ terraform state list
aws_instance.web[0]
aws_instance.web[1]
aws_instance.web[2]
```



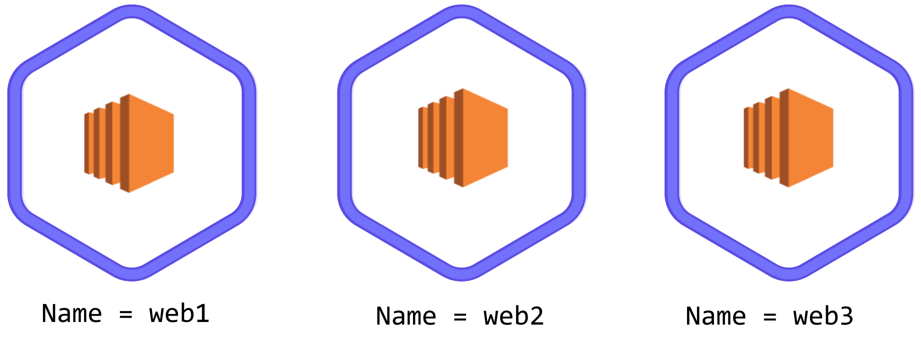
count

main.tf

```
resource "aws_instance"  "web" {
  ami           = var.ami
  instance_type = var.instance_type

   count    = length(var.webservers)

  tags = {
      Name = var.webservers[count.index]
  }             var.webservers[0] = web1
}
```

variables.tf

```
variable "ami" {
  default = "ami-06178cf087598769c"
}
variable "instance_type" {
  default = "m5.large"
}

variable "webservers" {
    type = list
    default = ["web1", "web2", "web3"]

}
```



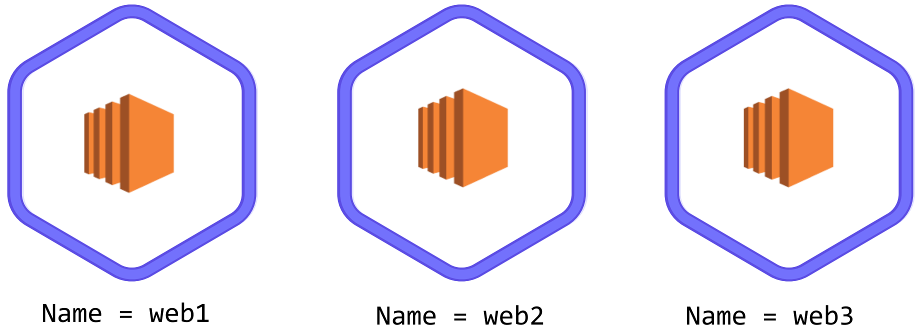
count

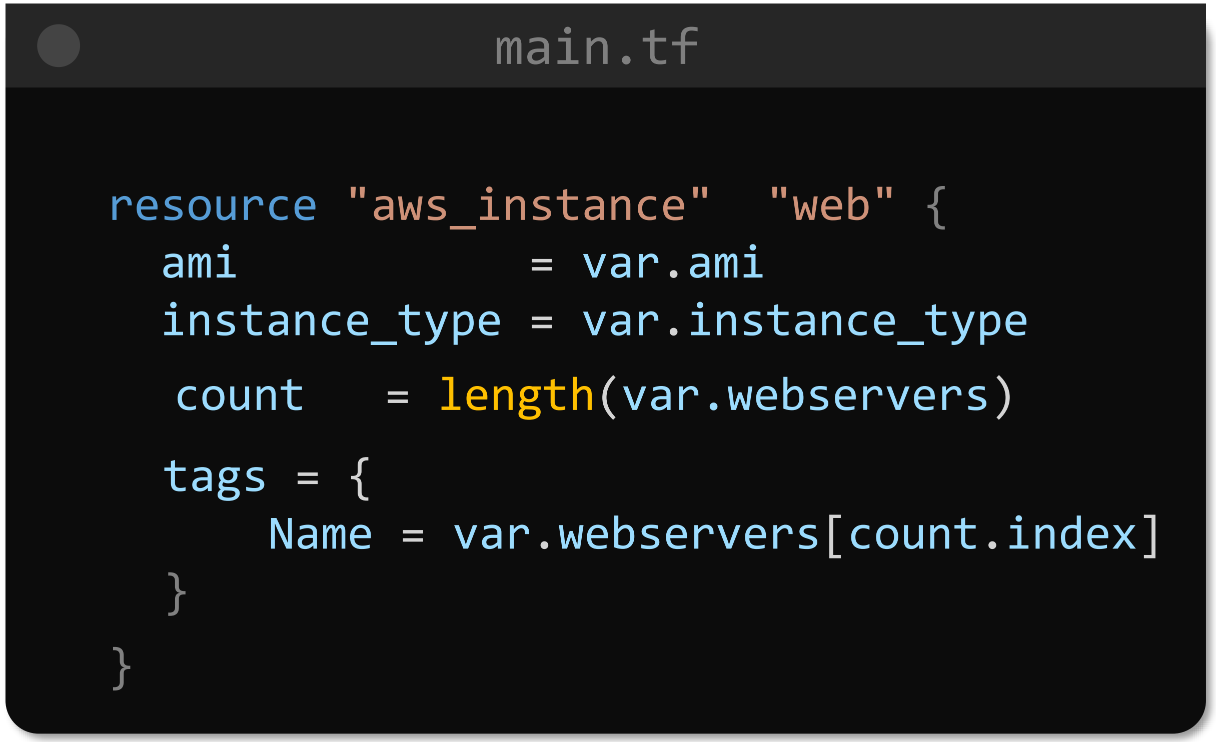
main.tf

```
resource "aws_instance"  "web" {
  ami           = var.ami
  instance_type = var.instance_type
   count    = length(var.webservers)
  tags = {
      Name = var.webservers[count.index]
  }
}
```

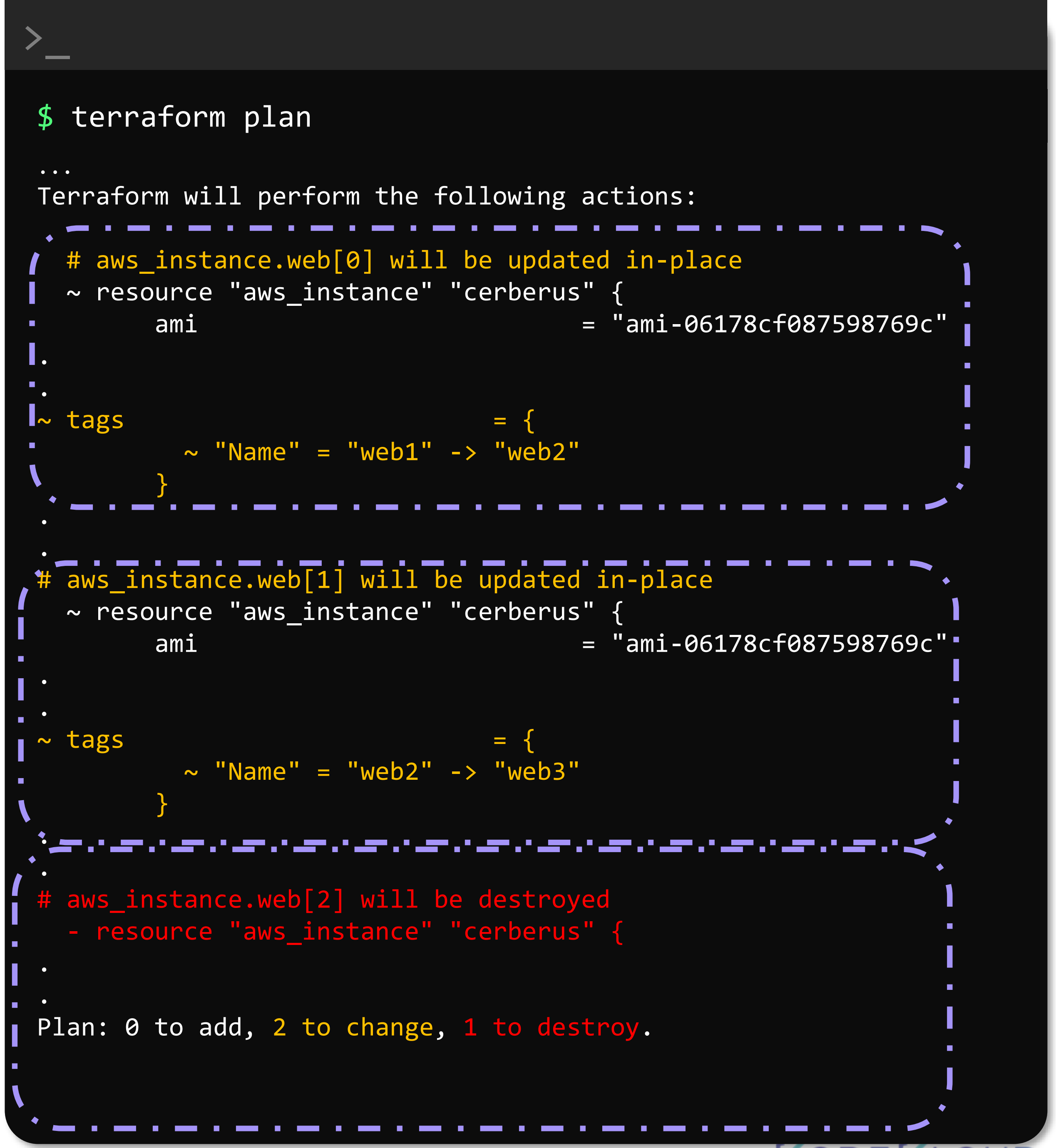
variables.tf

```
variable "ami" {
  default = "ami-06178cf087598769c"
}
variable "instance_type" {
  default = "m5.large"
}

variable "webservers" {
    type = list
    default =  ["web2", "web3"]

}
```

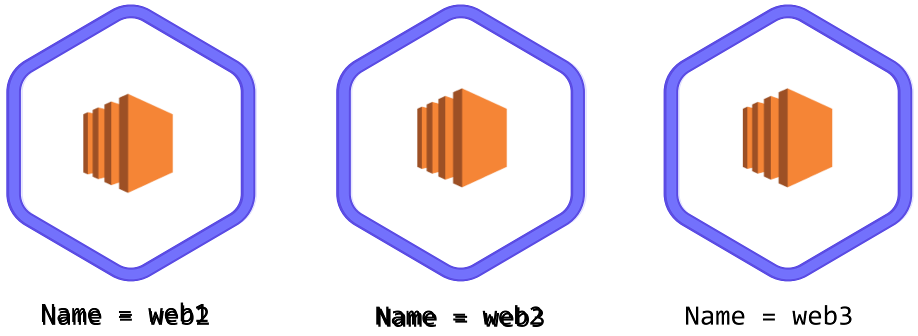



main.tf

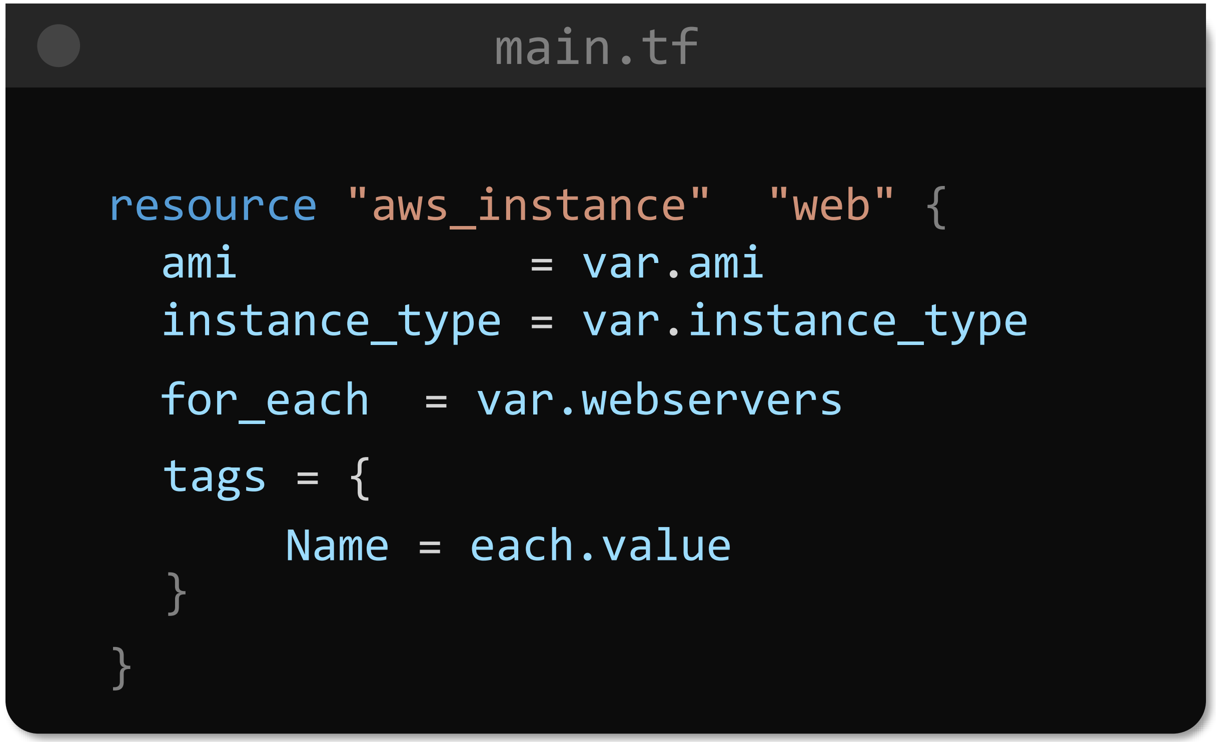
```
resource "aws_instance"  "web" {
  ami           = var.ami
  instance_type = var.instance_type
  count    = length(var.webservers)
  tags = {
      Name = var.webservers[count.index]
  }
}
```



```
$ terraform plan
...
Terraform will perform the following actions:
 # aws_instance.web[0] will be updated in-place
 ~ resource "aws_instance" "cerberus" {
       ami                          = "ami-06178cf087598769c"
.
.
~ tags                         = {
          ~ "Name" = "web1" -> "web2"
        }
.
.
# aws_instance.web[1] will be updated in-place
 ~ resource "aws_instance" "cerberus" {
       ami                          = "ami-06178cf087598769c"
.
.
~ tags                         = {
          ~ "Name" = "web2" -> "web3"
        }
.
.
# aws_instance.web[2] will be destroyed
 - resource "aws_instance" "cerberus" {
.
.
Plan: 0 to add, 2 to change, 1 to destroy.
```

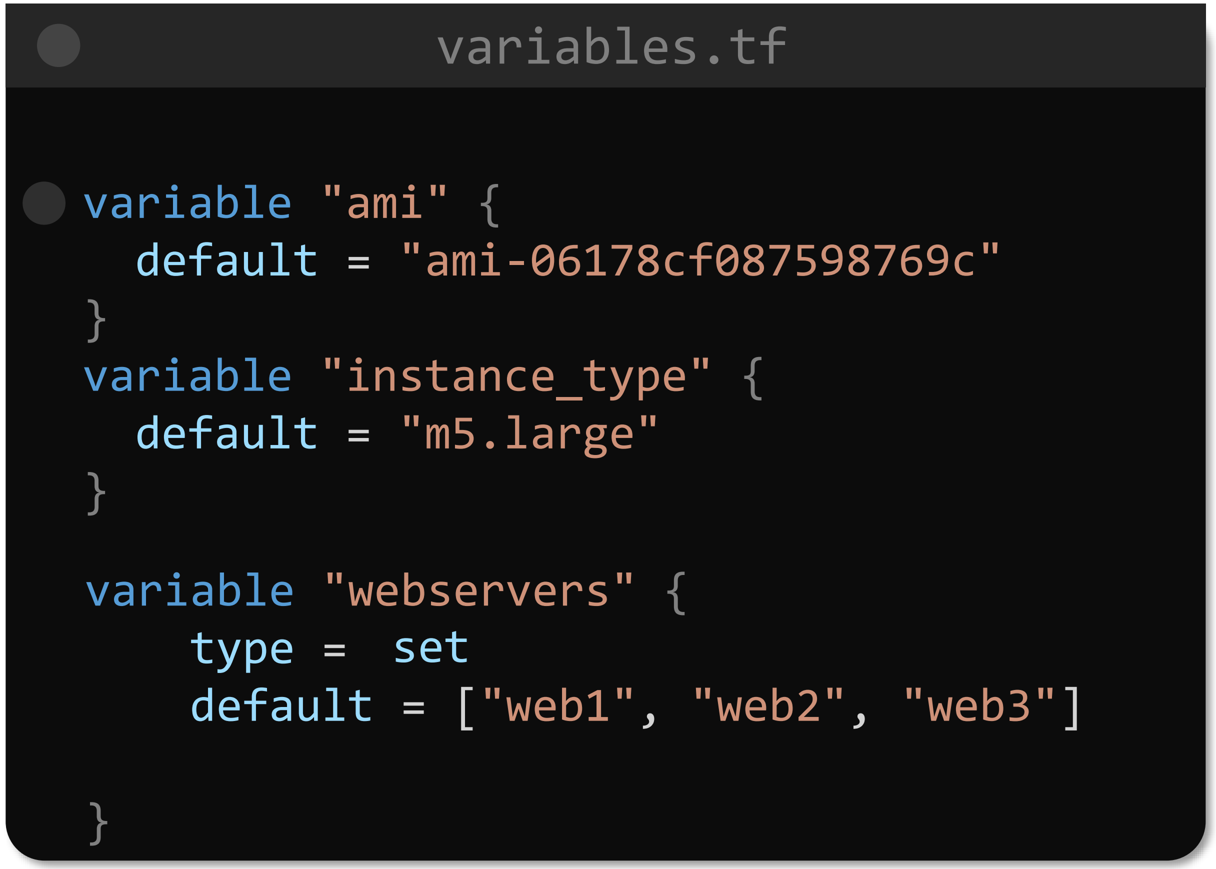


for_each



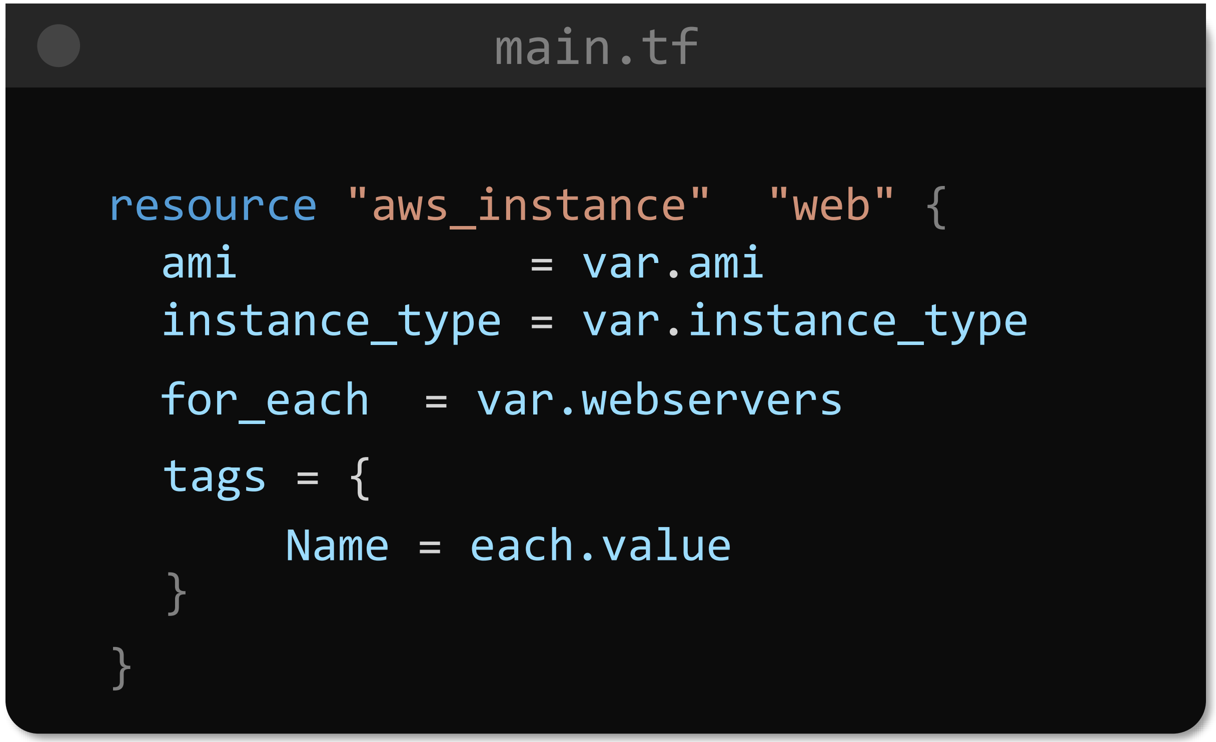
main.tf

```
resource "aws_instance"  "web" {
  ami           = var.ami
  instance_type = var.instance_type

  for_each   = var.webservers

  tags = {
       Name = each.value
  }
}
```



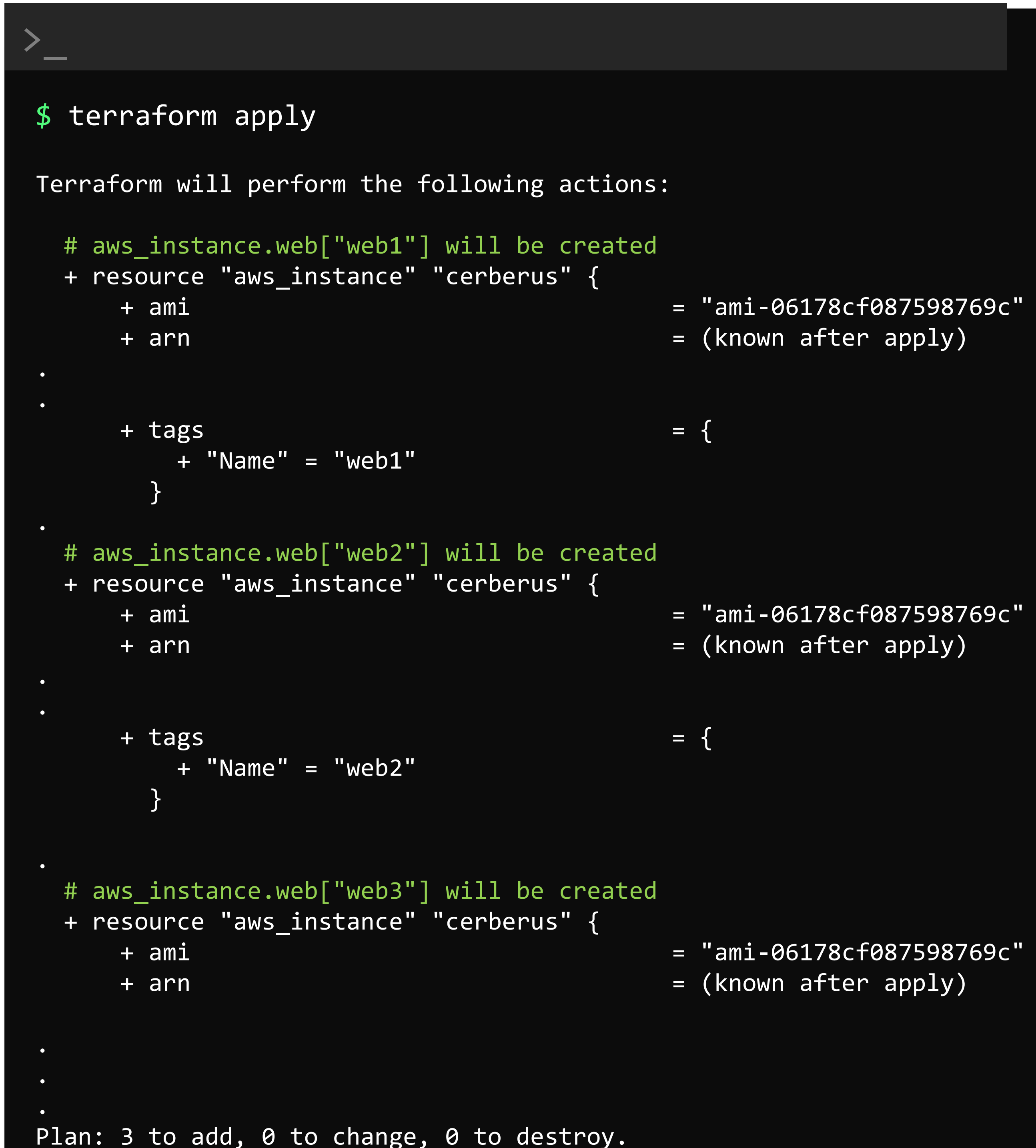
variables.tf

```
variable "ami" {
  default = "ami-06178cf087598769c"
}
variable "instance_type" {
  default = "m5.large"
}

variable "webservers" {
    type =  set
    default = ["web1", "web2", "web3"]

}
```

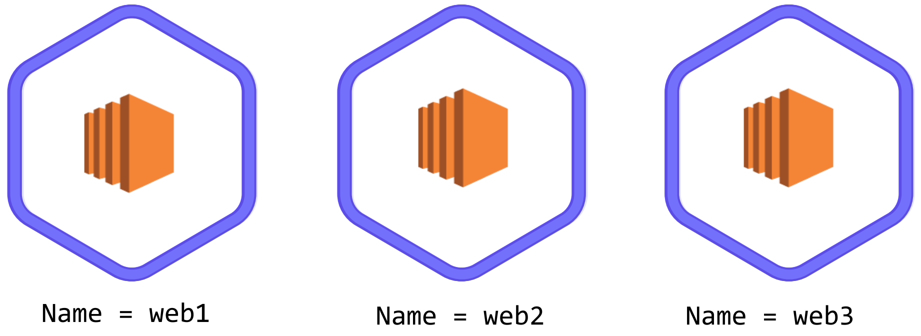


main.tf

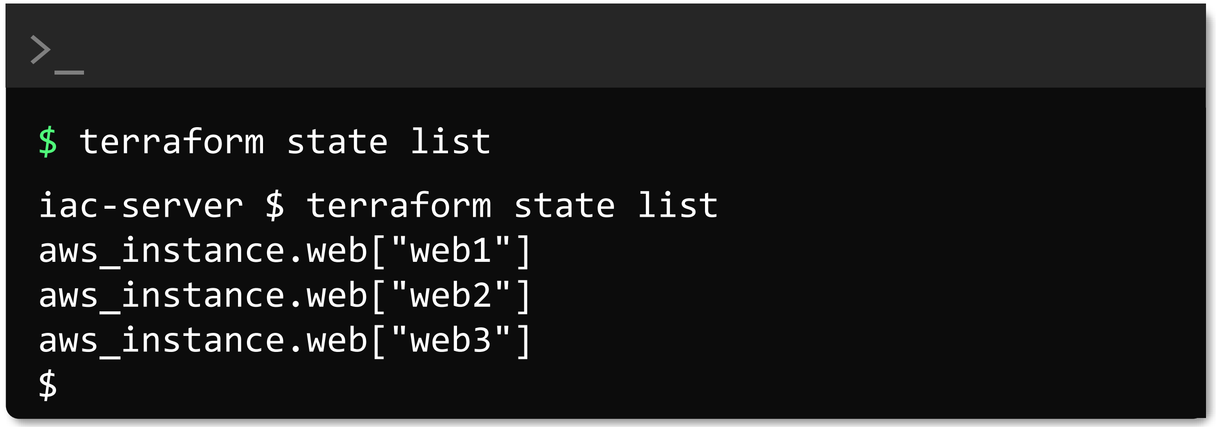
```
resource "aws_instance"  "web" {
  ami           = var.ami
  instance_type = var.instance_type

  for_each  = var.webservers

  tags = {
        Name = each.value
  }
}
```



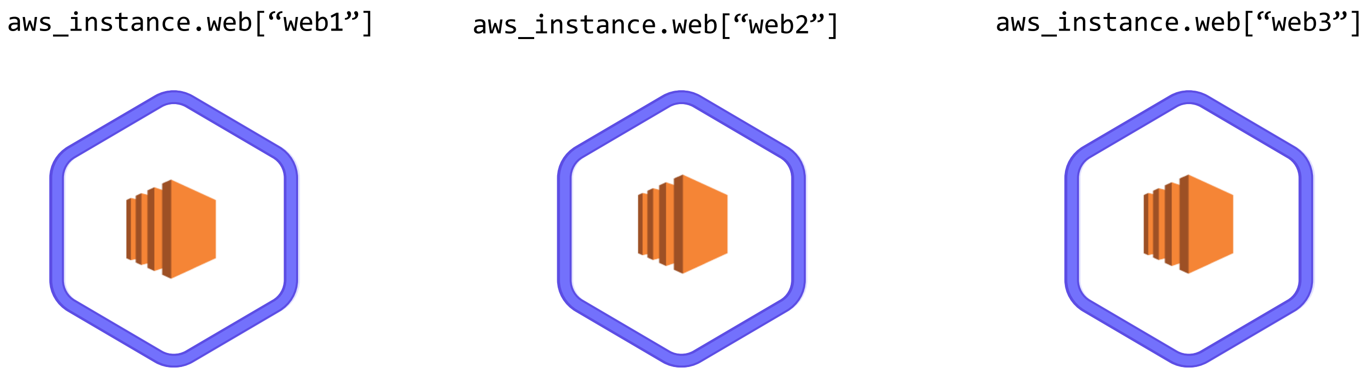
```
>_

$ terraform apply

Terraform will perform the following actions:

  # aws_instance.web["web1"] will be created
  + resource "aws_instance" "cerberus" {
      + ami                                = "ami-06178cf087598769c"
      + arn                                = (known after apply)
.
.
      + tags                               = {
          + "Name" = "web1"
        }
.
  # aws_instance.web["web2"] will be created
  + resource "aws_instance" "cerberus" {
      + ami                                = "ami-06178cf087598769c"
      + arn                                = (known after apply)
.
.
      + tags                               = {
          + "Name" = "web2"
        }

.
  # aws_instance.web["web3"] will be created
  + resource "aws_instance" "cerberus" {
      + ami                                = "ami-06178cf087598769c"
      + arn                                = (known after apply)

.
.
.
Plan: 3 to add, 0 to change, 0 to destroy.
```

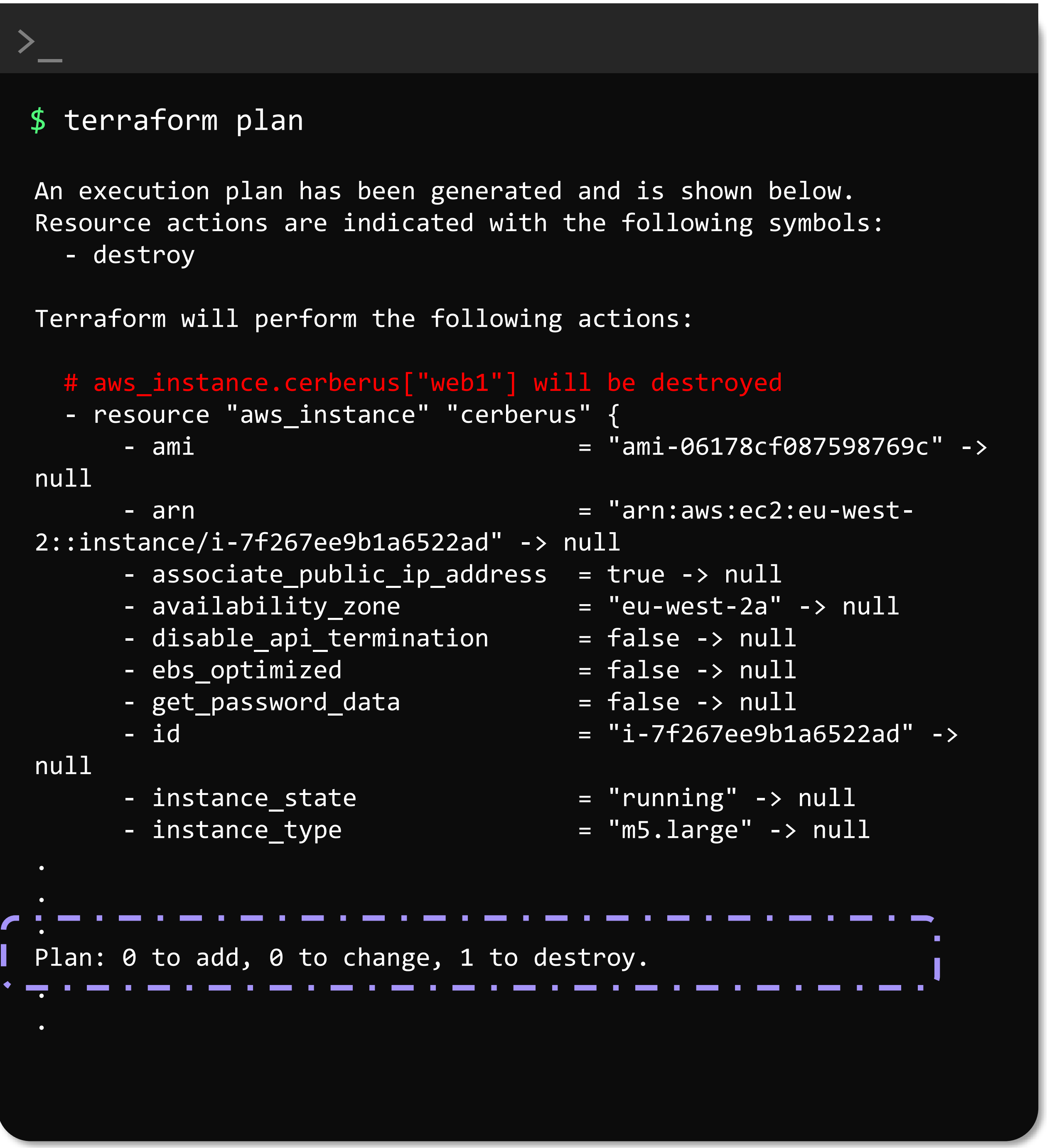


for_each

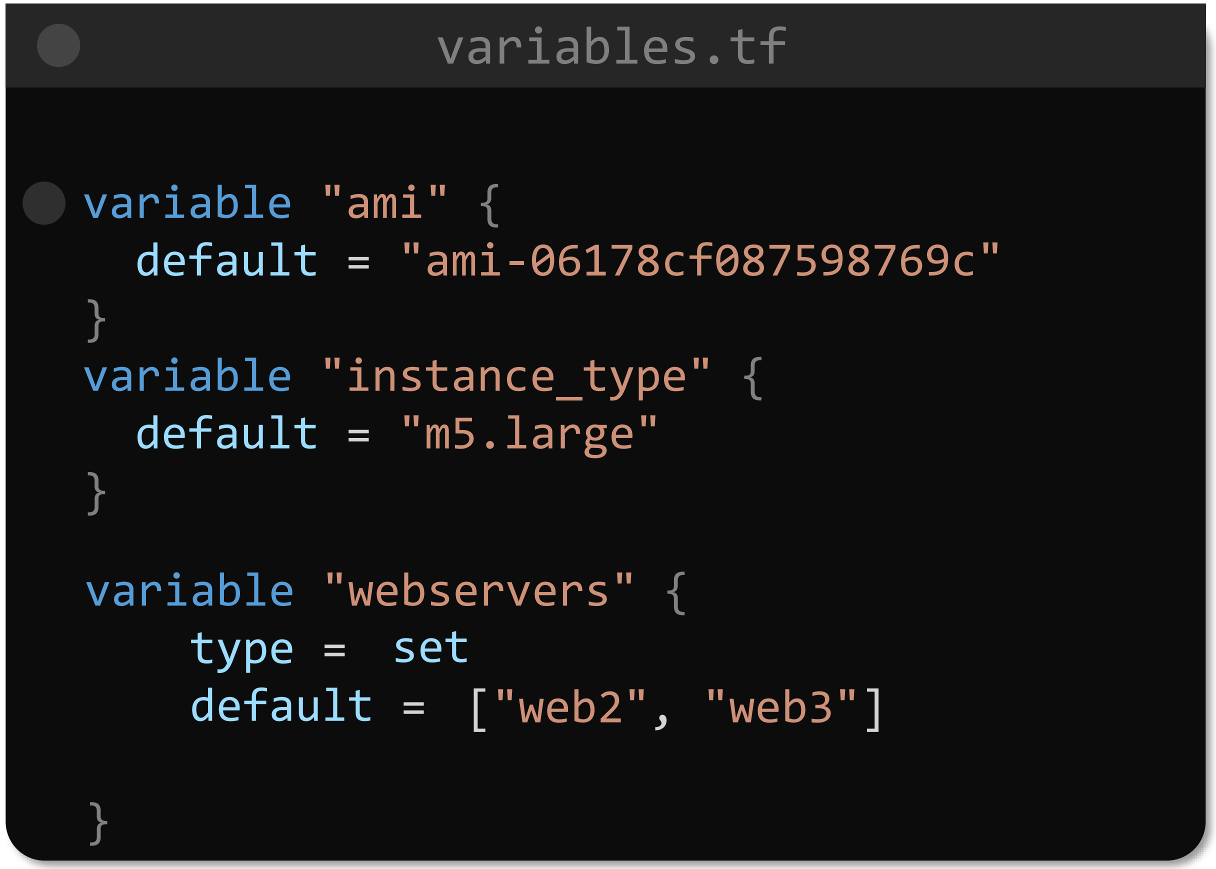


```
$ terraform state list
iac-server $ terraform state list
aws_instance.web["web1"]
aws_instance.web["web2"]
aws_instance.web["web3"]
$
```





```
>_
$ terraform plan

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_instance.cerberus["web1"] will be destroyed
  - resource "aws_instance" "cerberus" {
      - ami                          = "ami-06178cf087598769c" ->
null
      - arn                          = "arn:aws:ec2:eu-west-
2::instance/i-7f267ee9b1a6522ad" -> null
      - associate_public_ip_address  = true -> null
      - availability_zone            = "eu-west-2a" -> null
      - disable_api_termination      = false -> null
      - ebs_optimized                = false -> null
      - get_password_data            = false -> null
      - id                           = "i-7f267ee9b1a6522ad" ->
null
      - instance_state               = "running" -> null
      - instance_type                = "m5.large" -> null
.
.
.
Plan: 0 to add, 0 to change, 1 to destroy.
.
.
```



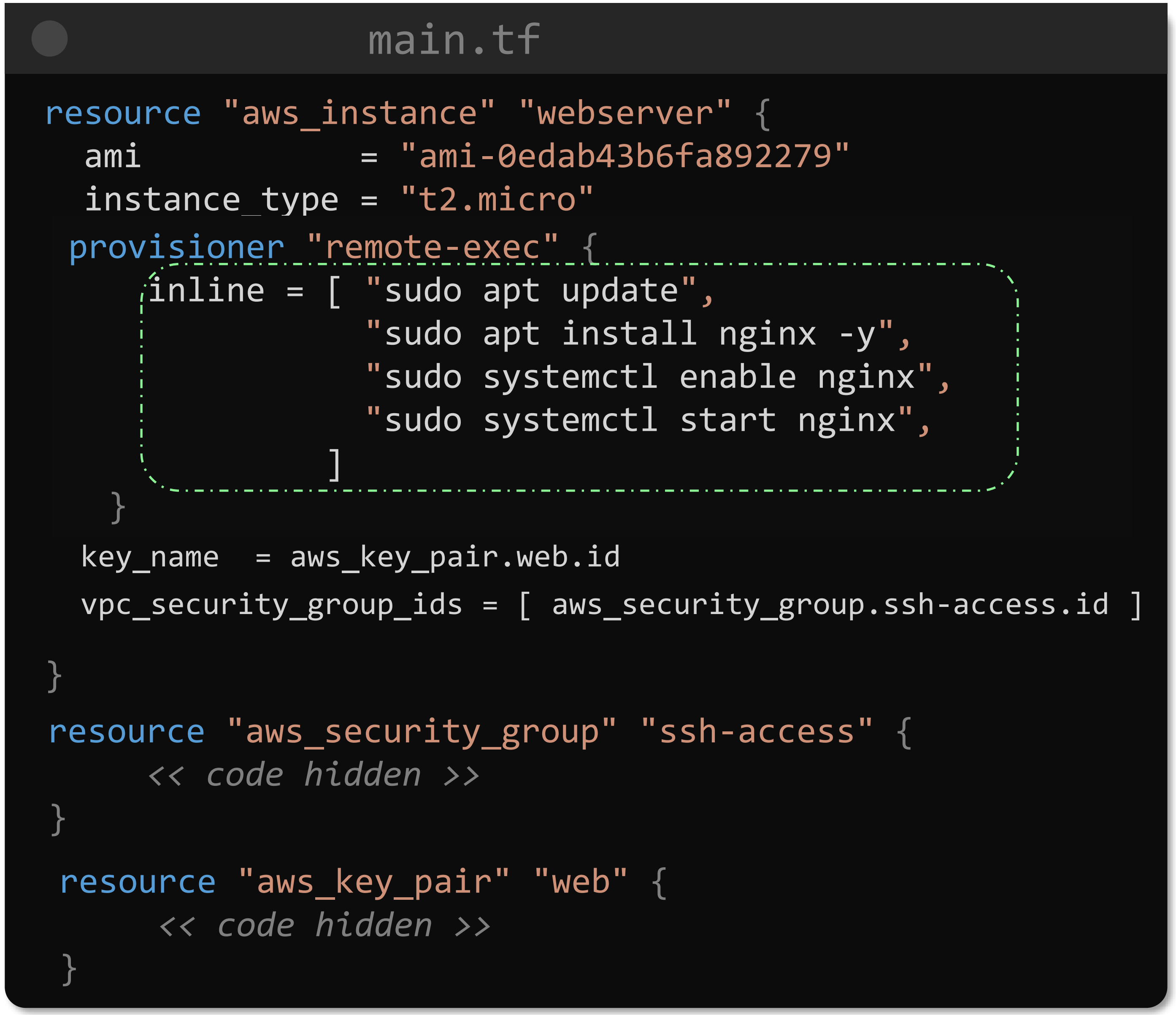
variables.tf

```
variable "ami" {
  default = "ami-06178cf087598769c"
}
variable "instance_type" {
  default = "m5.large"
}

variable "webservers" {
    type =  set
    default =  ["web2", "web3"]

}
```

KODEKLOUD

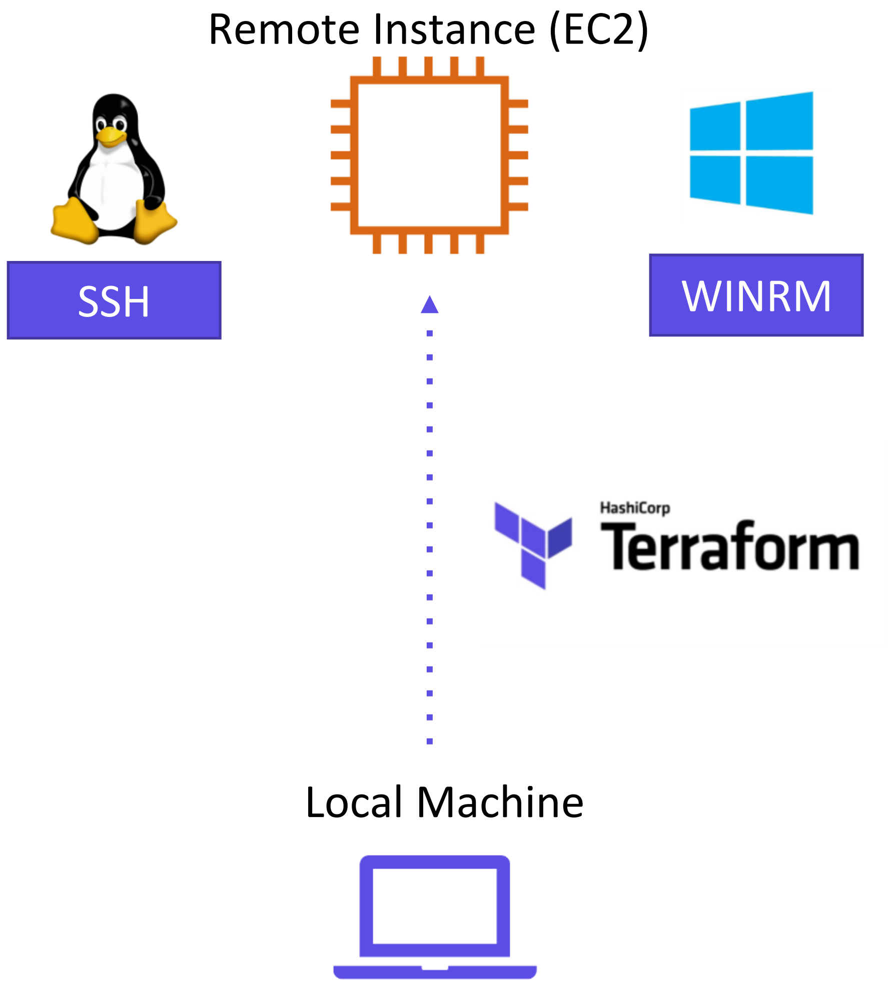
Terraform Provisioners

Remote Exec



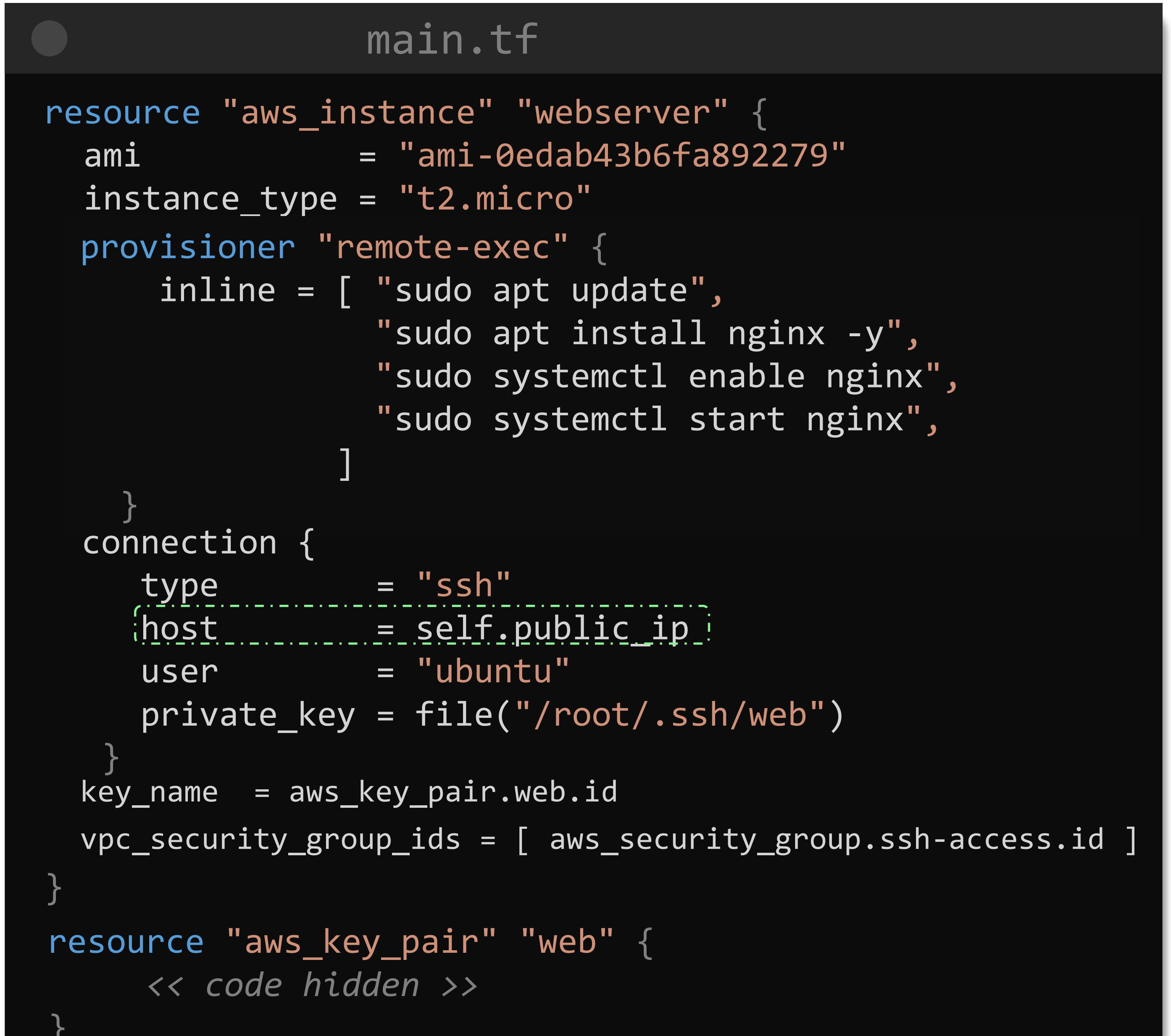
main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
 provisioner "remote-exec" {
     inline = [ "sudo apt update",
                "sudo apt install nginx -y",
                "sudo systemctl enable nginx",
                "sudo systemctl start nginx",
              ]
   }
  key_name  = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id ]

}
resource "aws_security_group" "ssh-access" {
     << code hidden >>
}
 resource "aws_key_pair" "web" {
      << code hidden >>
 }
```



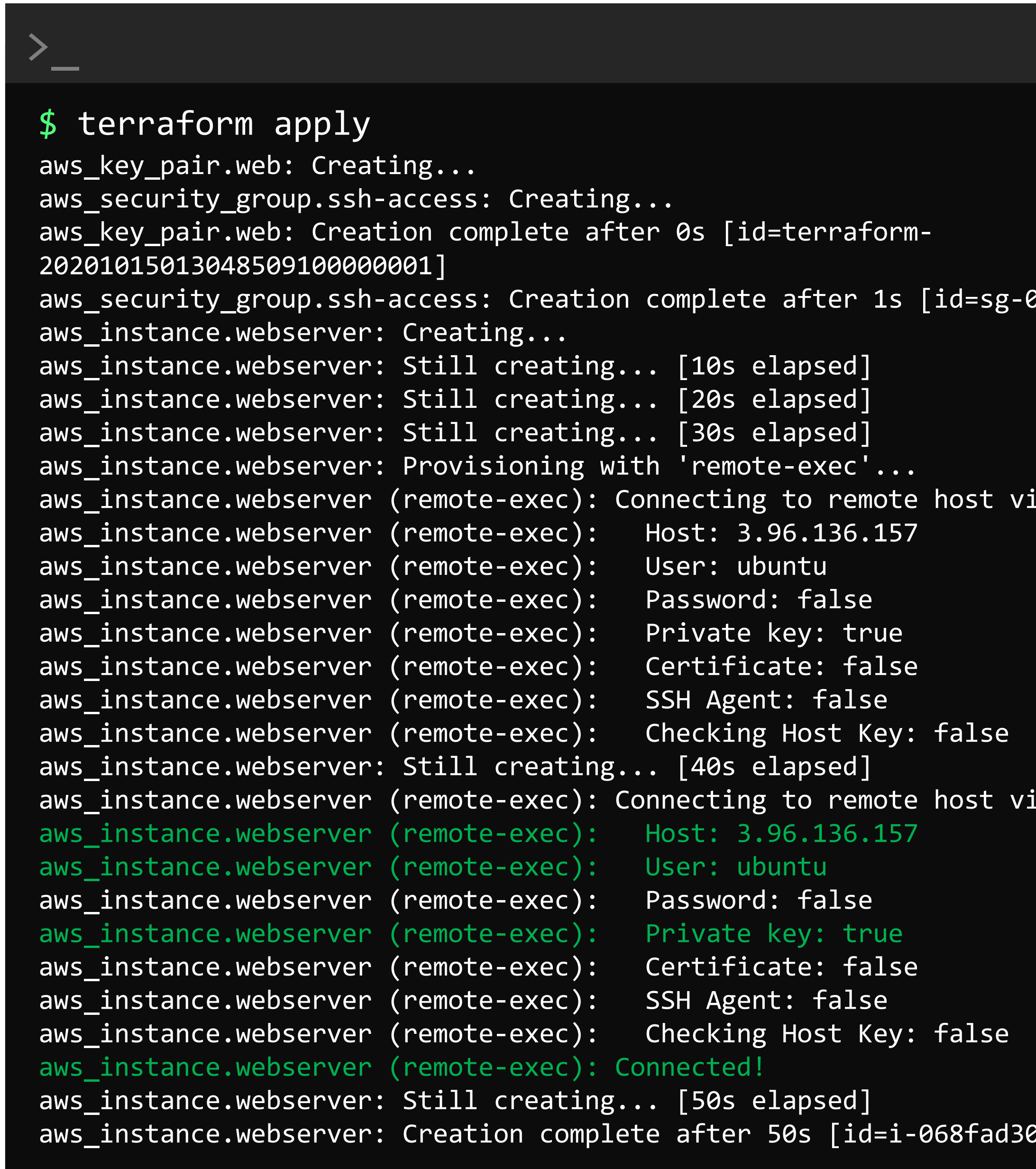
✔ Network Connectivity (Security Group)
✔ Authentication (SSH Key Pair)

Remote Exec



main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  provisioner "remote-exec" {
      inline = [ "sudo apt update",
                 "sudo apt install nginx -y",
                 "sudo systemctl enable nginx",
                 "sudo systemctl start nginx",
               ]
    }
  connection {
     type        = "ssh"
     host        = self.public_ip
     user        = "ubuntu"
     private_key = file("/root/.ssh/web")
   }
  key_name  = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id ]
}
resource "aws_key_pair" "web" {
     << code hidden >>
}
```



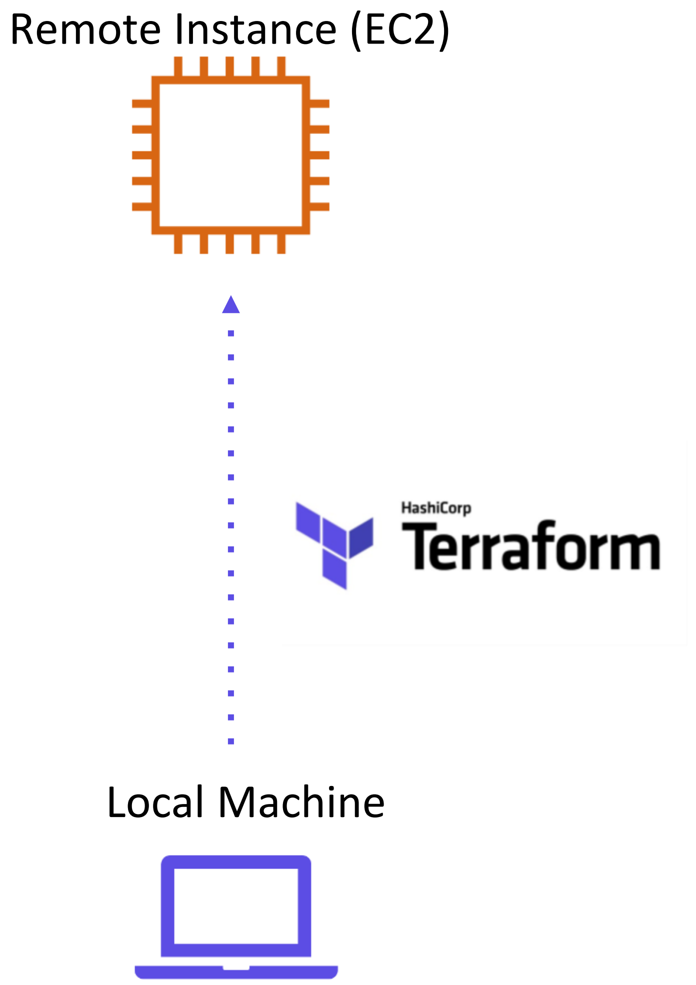
```
$ terraform apply
aws_key_pair.web: Creating...
aws_security_group.ssh-access: Creating...
aws_key_pair.web: Creation complete after 0s [id=terraform-
20201015013048509100000001]
aws_security_group.ssh-access: Creation complete after 1s [id=sg-0
aws_instance.webserver: Creating...
aws_instance.webserver: Still creating... [10s elapsed]
aws_instance.webserver: Still creating... [20s elapsed]
aws_instance.webserver: Still creating... [30s elapsed]
aws_instance.webserver: Provisioning with 'remote-exec'...
aws_instance.webserver (remote-exec): Connecting to remote host vi
aws_instance.webserver (remote-exec):   Host: 3.96.136.157
aws_instance.webserver (remote-exec):   User: ubuntu
aws_instance.webserver (remote-exec):   Password: false
aws_instance.webserver (remote-exec):   Private key: true
aws_instance.webserver (remote-exec):   Certificate: false
aws_instance.webserver (remote-exec):   SSH Agent: false
aws_instance.webserver (remote-exec):   Checking Host Key: false
aws_instance.webserver: Still creating... [40s elapsed]
aws_instance.webserver (remote-exec): Connecting to remote host vi
aws_instance.webserver (remote-exec):   Host: 3.96.136.157
aws_instance.webserver (remote-exec):   User: ubuntu
aws_instance.webserver (remote-exec):   Password: false
aws_instance.webserver (remote-exec):   Private key: true
aws_instance.webserver (remote-exec):   Certificate: false
aws_instance.webserver (remote-exec):   SSH Agent: false
aws_instance.webserver (remote-exec):   Checking Host Key: false
aws_instance.webserver (remote-exec): Connected!
aws_instance.webserver: Still creating... [50s elapsed]
aws_instance.webserver: Creation complete after 50s [id=i-068fad30
```

Local Exec

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"

  provisioner "local-exec" {
   command = "echo ${aws_instance.webserver2.public_ip} >> /tmp/ips.txt"
  }



}
```

```
$ cat /tmp/ips.txt
54.214.68.27
```



KODEKLOUD

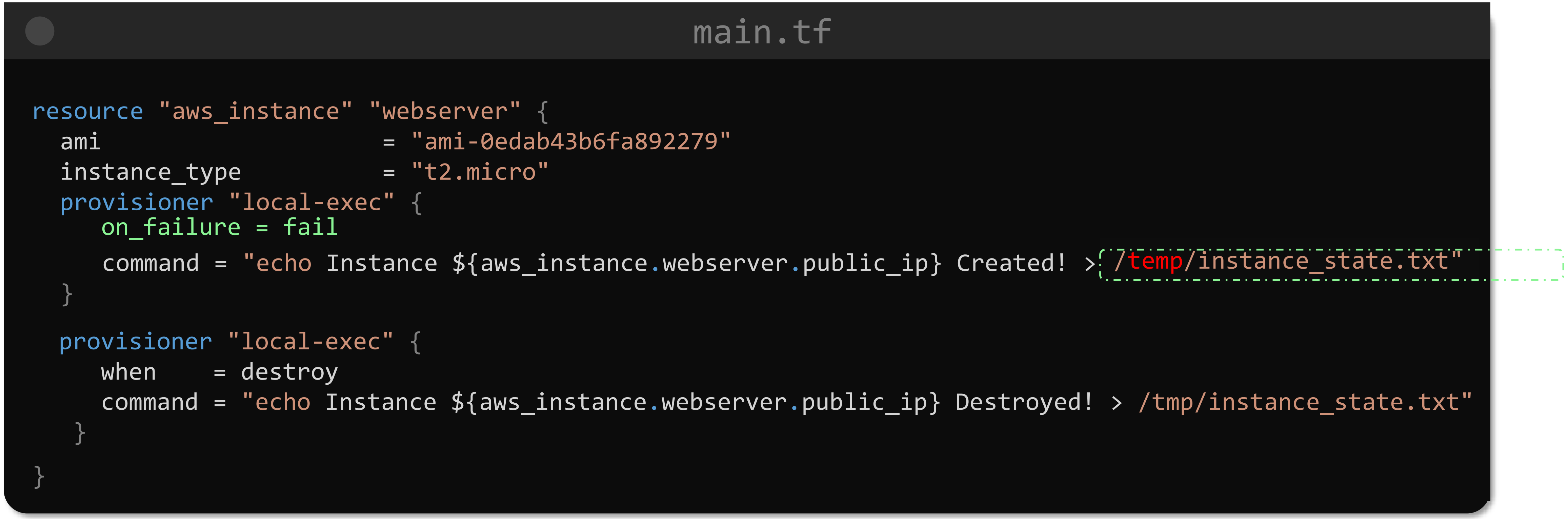
Destroy Time Provisioner

main.tf

```
resource "aws_instance" "webserver" {
  ami                     = "ami-0edab43b6fa892279"
  instance_type           = "t2.micro"
  provisioner "local-exec" {
     command = "echo Instance ${aws_instance.webserver.public_ip} Created! > /tmp/instance_state.txt"
  }

  provisioner "local-exec" {
     when    = destroy
     command = "echo Instance ${aws_instance.webserver.public_ip} Destroyed! > /tmp/instance_state.txt"
   }

}
```

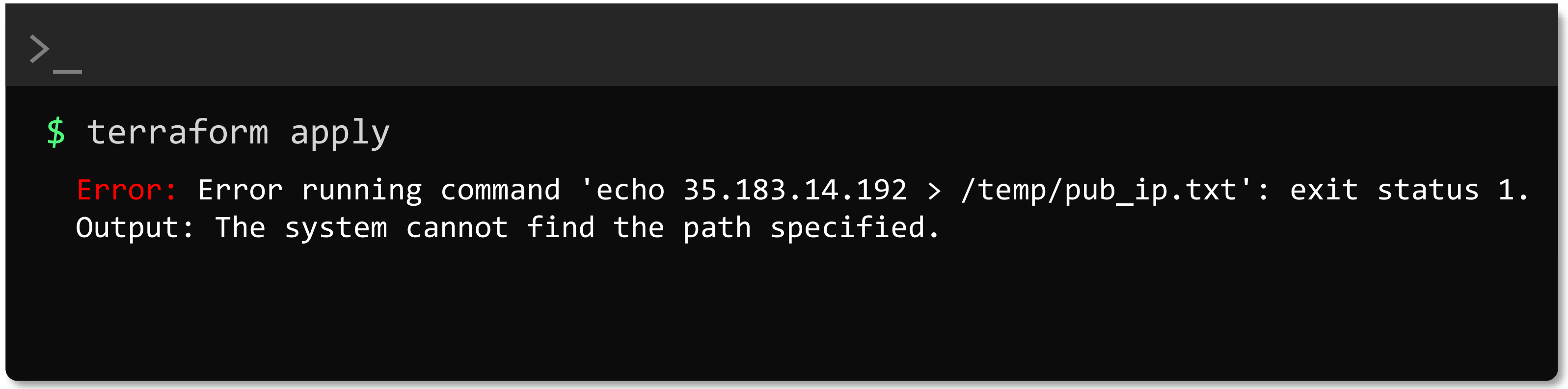
```
>_

$ cat /tmp/instance_state.txt
  Instance 3.96.136.157 Deleted!
```

Failure Behavior



main.tf

```
resource "aws_instance" "webserver" {
  ami                       = "ami-0edab43b6fa892279"
  instance_type             = "t2.micro"
  provisioner "local-exec" {
     on_failure = fail
     command = "echo Instance ${aws_instance.webserver.public_ip} Created! > /temp/instance_state.txt"
  }

  provisioner "local-exec" {
     when    = destroy
     command = "echo Instance ${aws_instance.webserver.public_ip} Destroyed! > /tmp/instance_state.txt"
   }

}
```



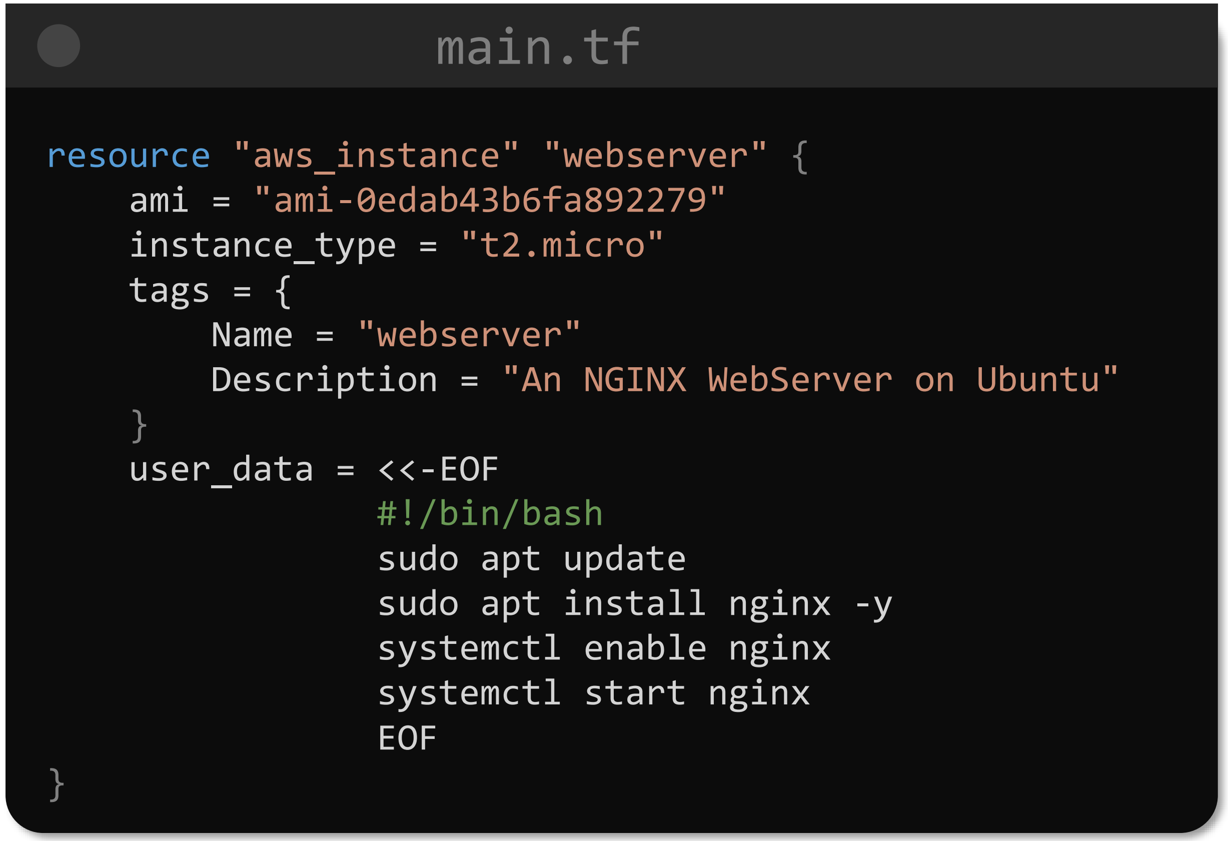
```
$ terraform apply
  Error: Error running command 'echo 35.183.14.192 > /temp/pub_ip.txt': exit status 1.
  Output: The system cannot find the path specified.
```

Failure Behavior

main.tf

```
resource "aws_instance" "webserver" {
  ami                     = "ami-0edab43b6fa892279"
  instance_type           = "t2.micro"
  provisioner "local-exec" {
     on_failure = continue
     command = "echo Instance ${aws_instance.webserver.public_ip} Created! > /temp/instance_state.txt"
  }

  provisioner "local-exec" {
     when    = destroy
     command = "echo Instance ${aws_instance.webserver.public_ip} Destroyed! > /tmp/instance_state.txt"
   }
}
```

```
>_

$ terraform apply
 aws_instance.webserver (local-exec) The system cannot find the path specified.
 aws_instance.project: Creation complete after 22s [id=i-01585c2b9dbc445db]

 Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
```



main.tf

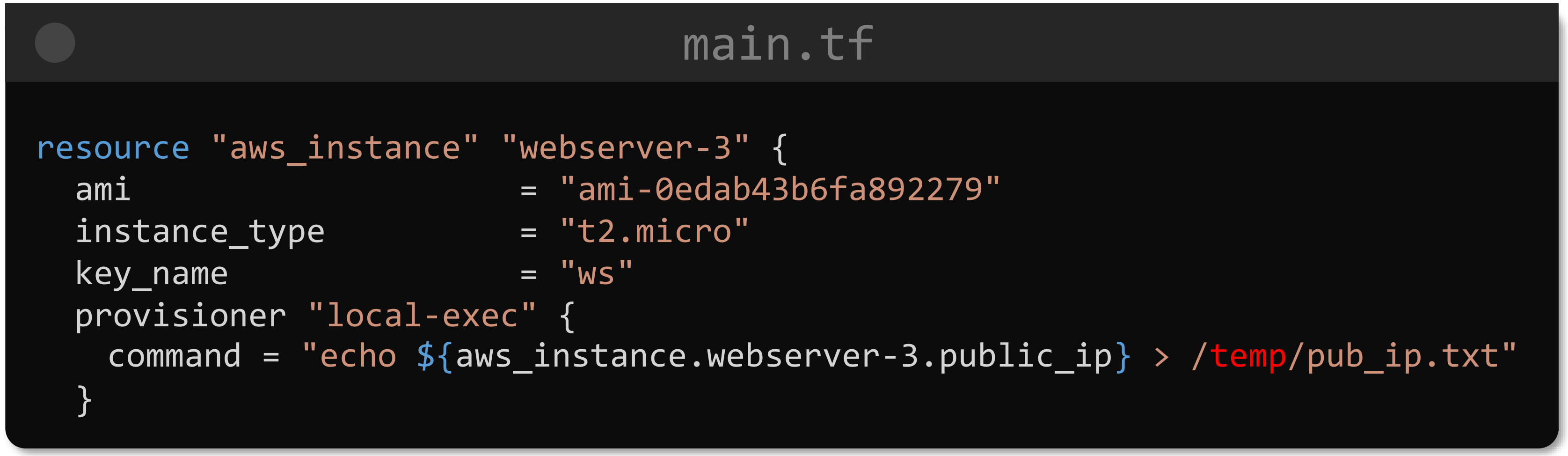
```
resource "aws_instance" "webserver" {
    ami = "ami-0edab43b6fa892279"
    instance_type = "t2.micro"
    tags = {
        Name = "webserver"
        Description = "An NGINX WebServer on Ubuntu"
    }
    user_data = <<-EOF
                #!/bin/bash
                sudo apt update
                sudo apt install nginx -y
                systemctl enable nginx
                systemctl start nginx
                EOF
}
```

Provider	Resource	Option
AWS	aws_instance	user_data
Azure	azurrerm_virtual_machine	custom_data
GCP	google_compute_instance	meta_data
Vmware vSphere	vsphere_virtual_machine	user_data.txt

KODEKLOUD

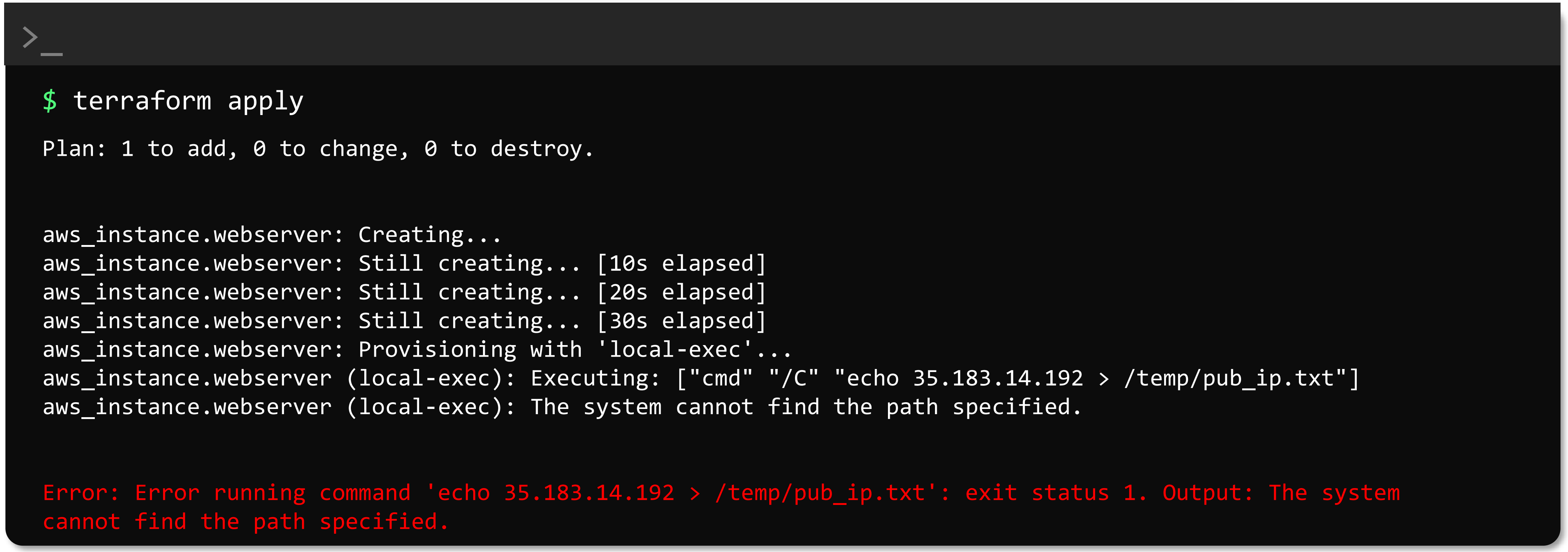
Terraform Taint

Taint



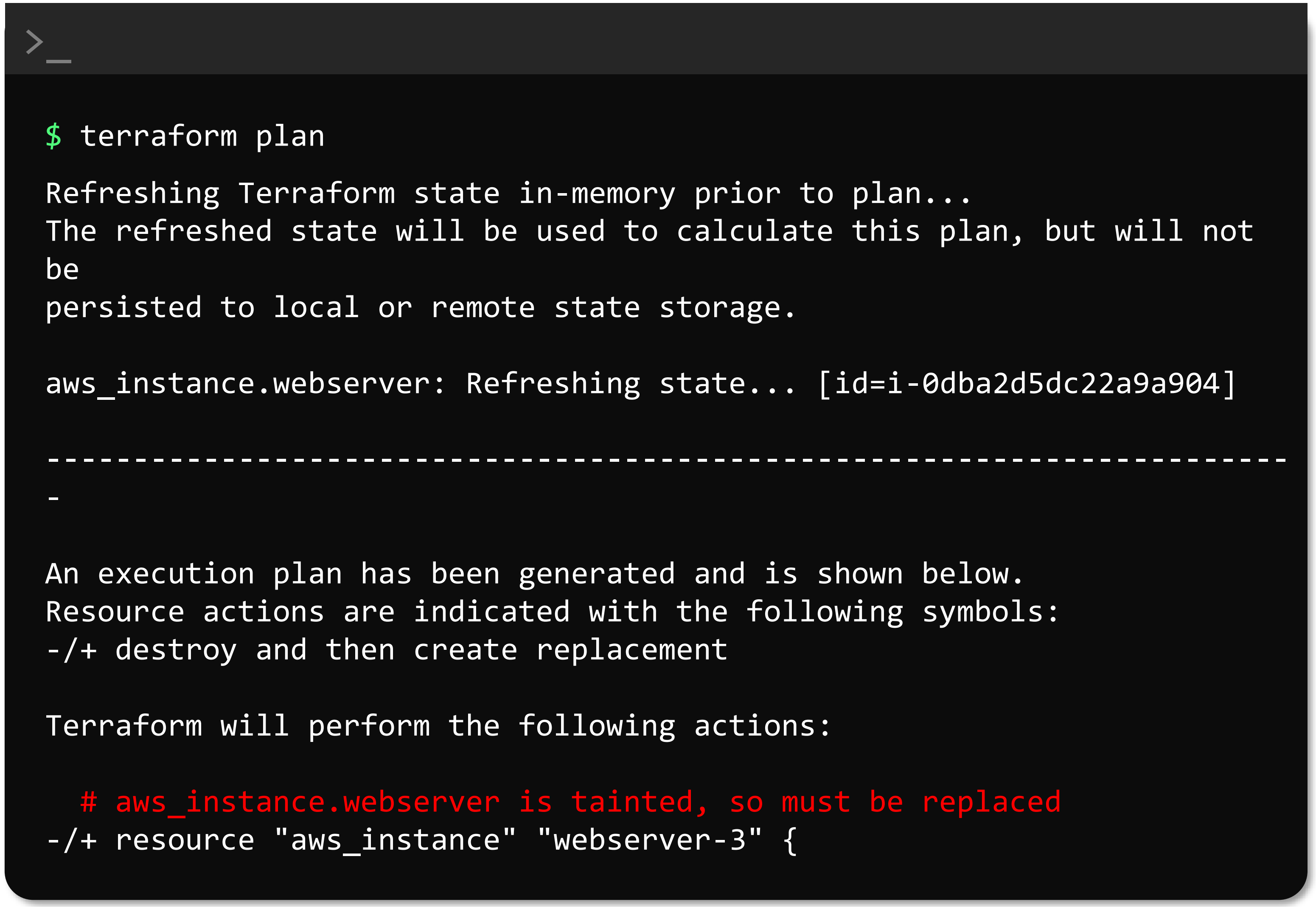
main.tf

```
resource "aws_instance" "webserver-3" {
  ami                    = "ami-0edab43b6fa892279"
  instance_type          = "t2.micro"
  key_name               = "ws"
  provisioner "local-exec" {
    command = "echo ${aws_instance.webserver-3.public_ip} > /temp/pub_ip.txt"
  }
```



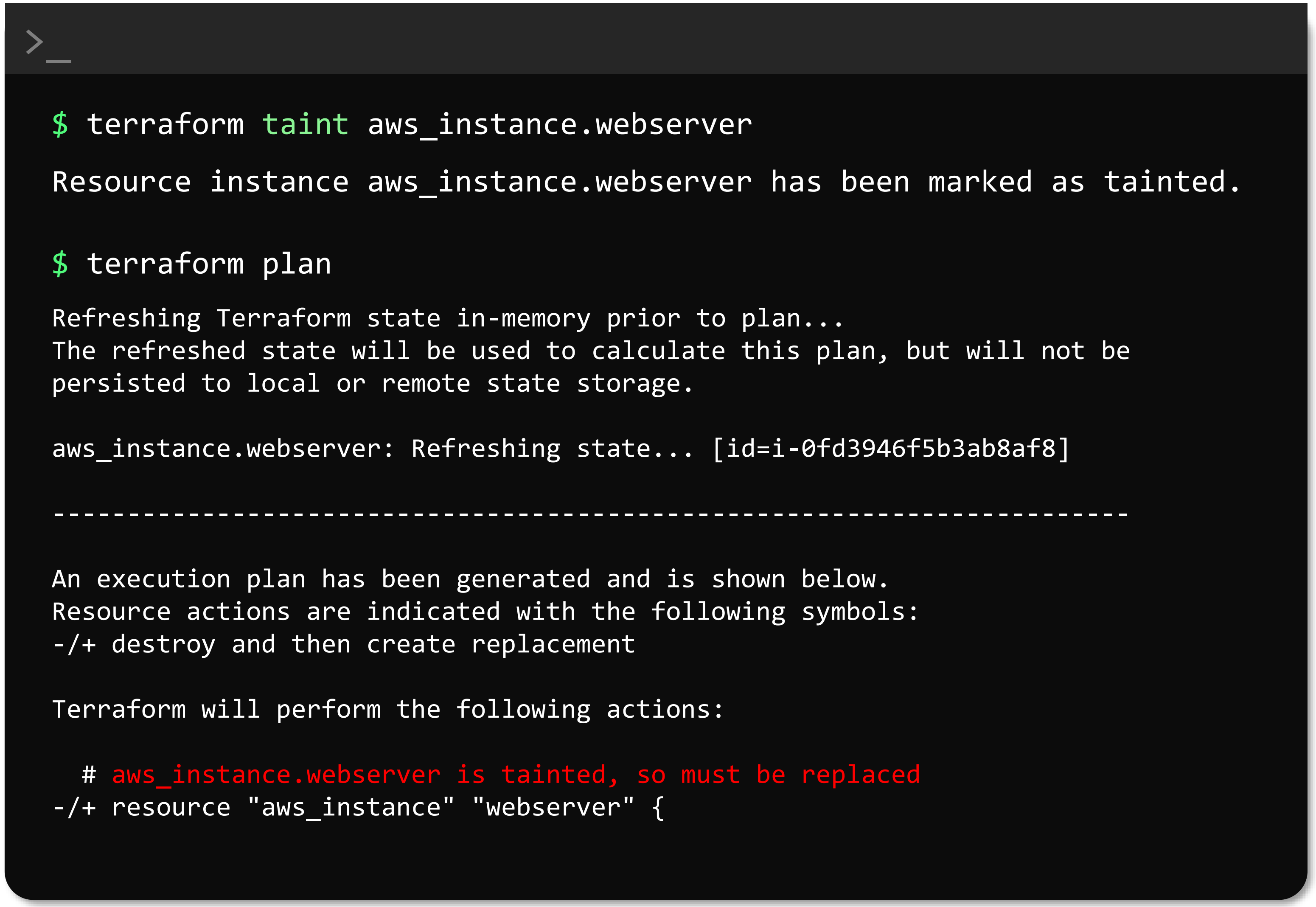
```
$ terraform apply
Plan: 1 to add, 0 to change, 0 to destroy.


aws_instance.webserver: Creating...
aws_instance.webserver: Still creating... [10s elapsed]
aws_instance.webserver: Still creating... [20s elapsed]
aws_instance.webserver: Still creating... [30s elapsed]
aws_instance.webserver: Provisioning with 'local-exec'...
aws_instance.webserver (local-exec): Executing: ["cmd" "/C" "echo 35.183.14.192 > /temp/pub_ip.txt"]
aws_instance.webserver (local-exec): The system cannot find the path specified.


Error: Error running command 'echo 35.183.14.192 > /temp/pub_ip.txt': exit status 1. Output: The system
cannot find the path specified.
```

Taint

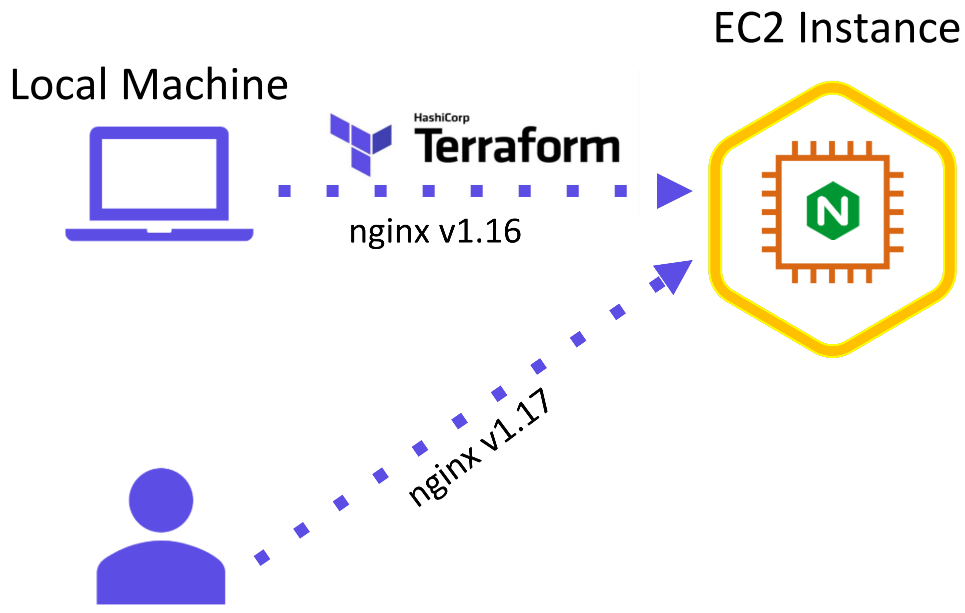


```
$ terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not
be
persisted to local or remote state storage.

aws_instance.webserver: Refreshing state... [id=i-0dba2d5dc22a9a904]

----------------------------------------------------------------------
-

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # aws_instance.webserver is tainted, so must be replaced
-/+ resource "aws_instance" "webserver-3" {
```

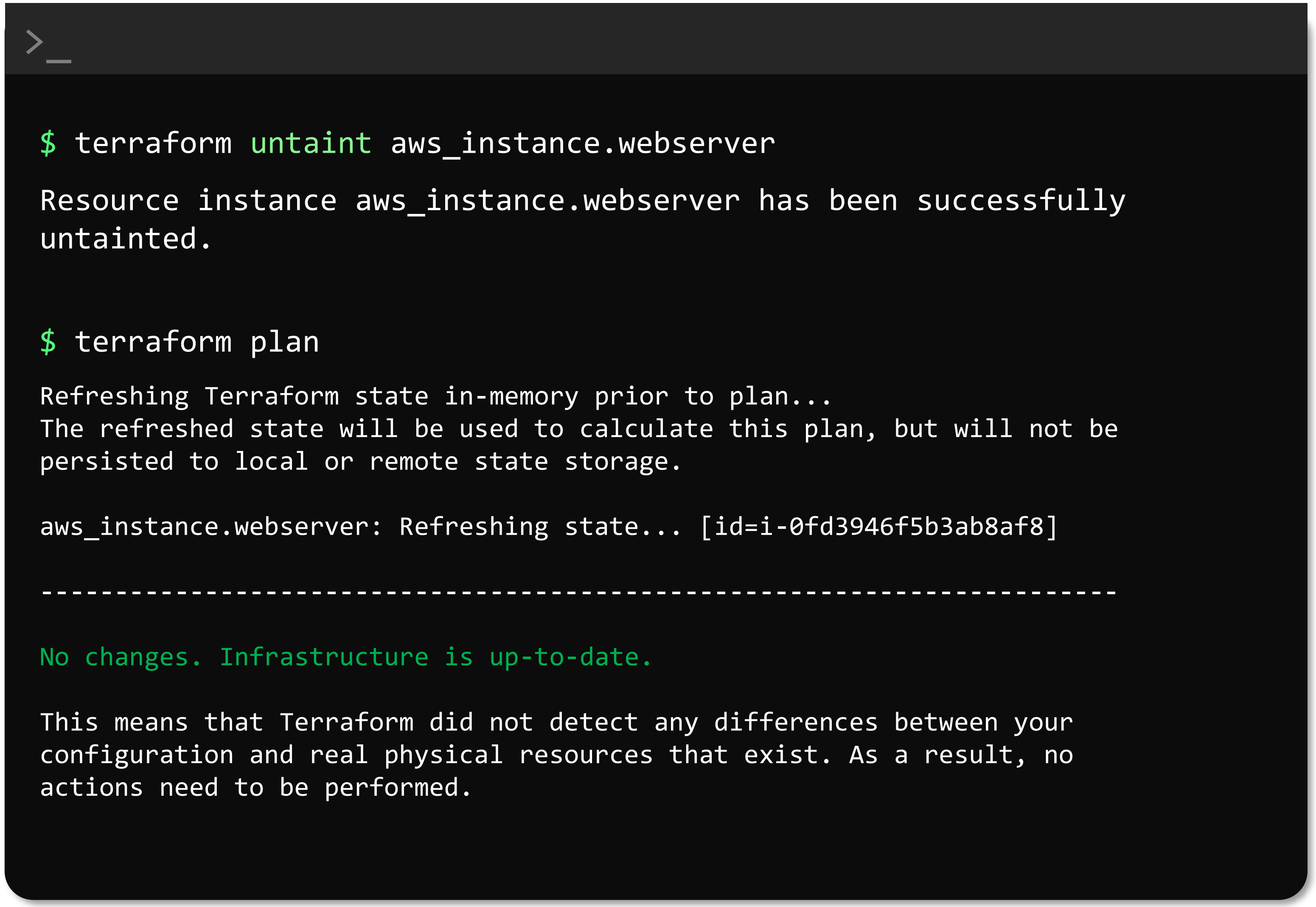
Taint



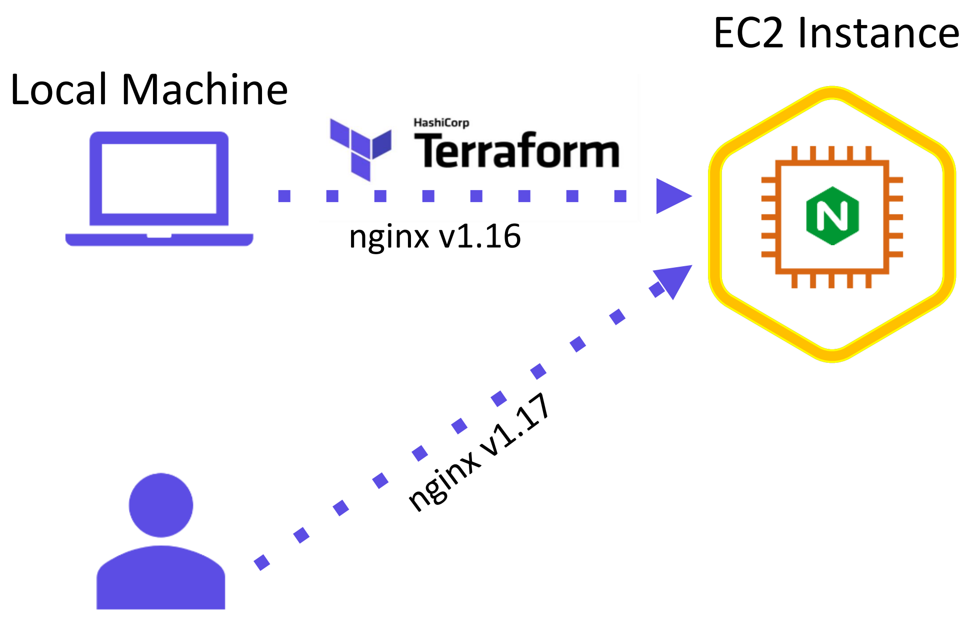
```
$ terraform taint aws_instance.webserver
Resource instance aws_instance.webserver has been marked as tainted.

$ 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_instance.webserver: Refreshing state... [id=i-0fd3946f5b3ab8af8]

------------------------------------------------------------------------

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # aws_instance.webserver is tainted, so must be replaced
-/+ resource "aws_instance" "webserver" {
```



Taint



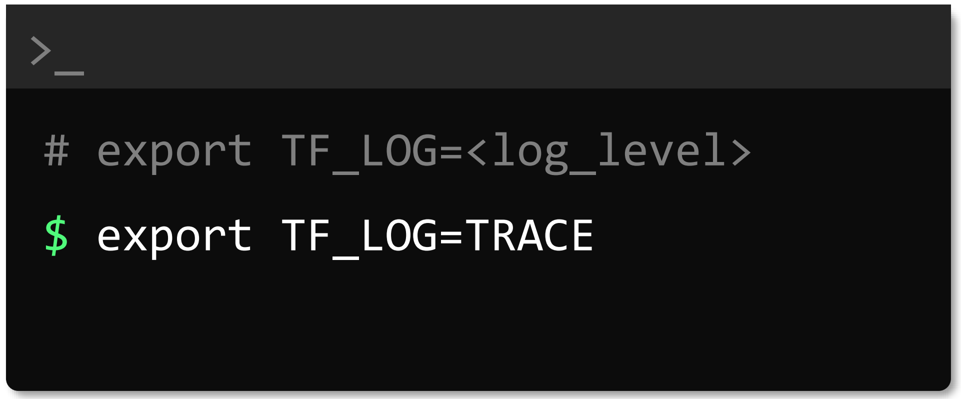
```
$ terraform untaint aws_instance.webserver
Resource instance aws_instance.webserver has been successfully
untainted.


$ 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_instance.webserver: Refreshing state... [id=i-0fd3946f5b3ab8af8]

------------------------------------------------------------------------

No changes. Infrastructure is up-to-date.

This means that Terraform did not detect any differences between your
configuration and real physical resources that exist. As a result, no
actions need to be performed.
```



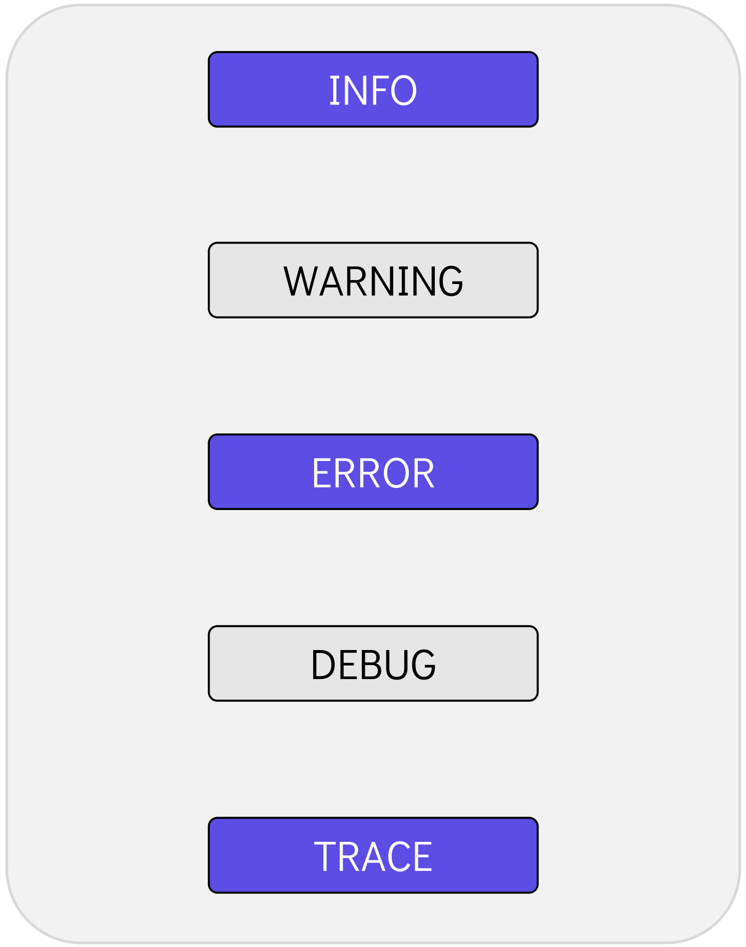
KODEKLOUD

Debugging

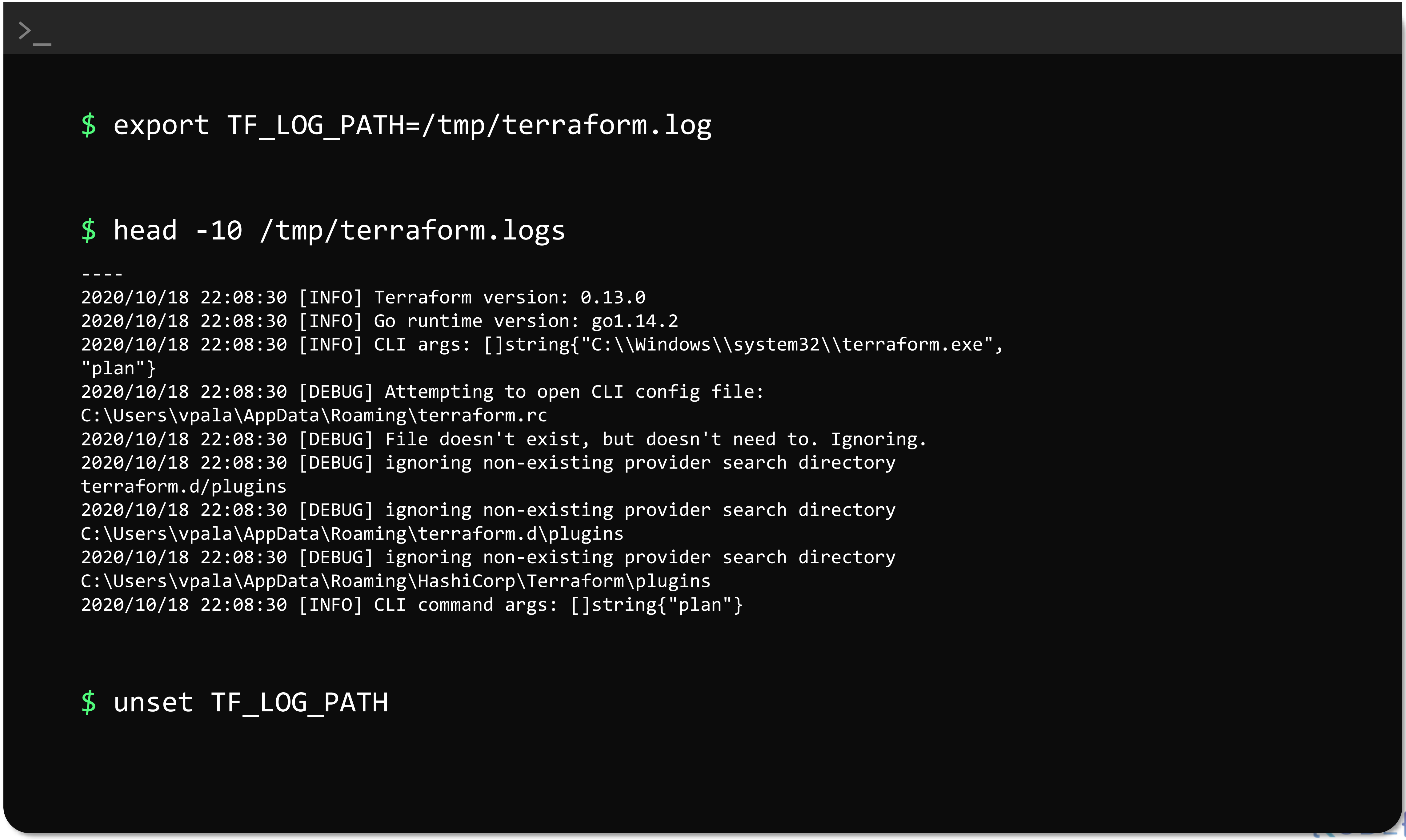
Log Levels



```
>_
# export TF_LOG=<log_level>
$ export TF_LOG=TRACE
```



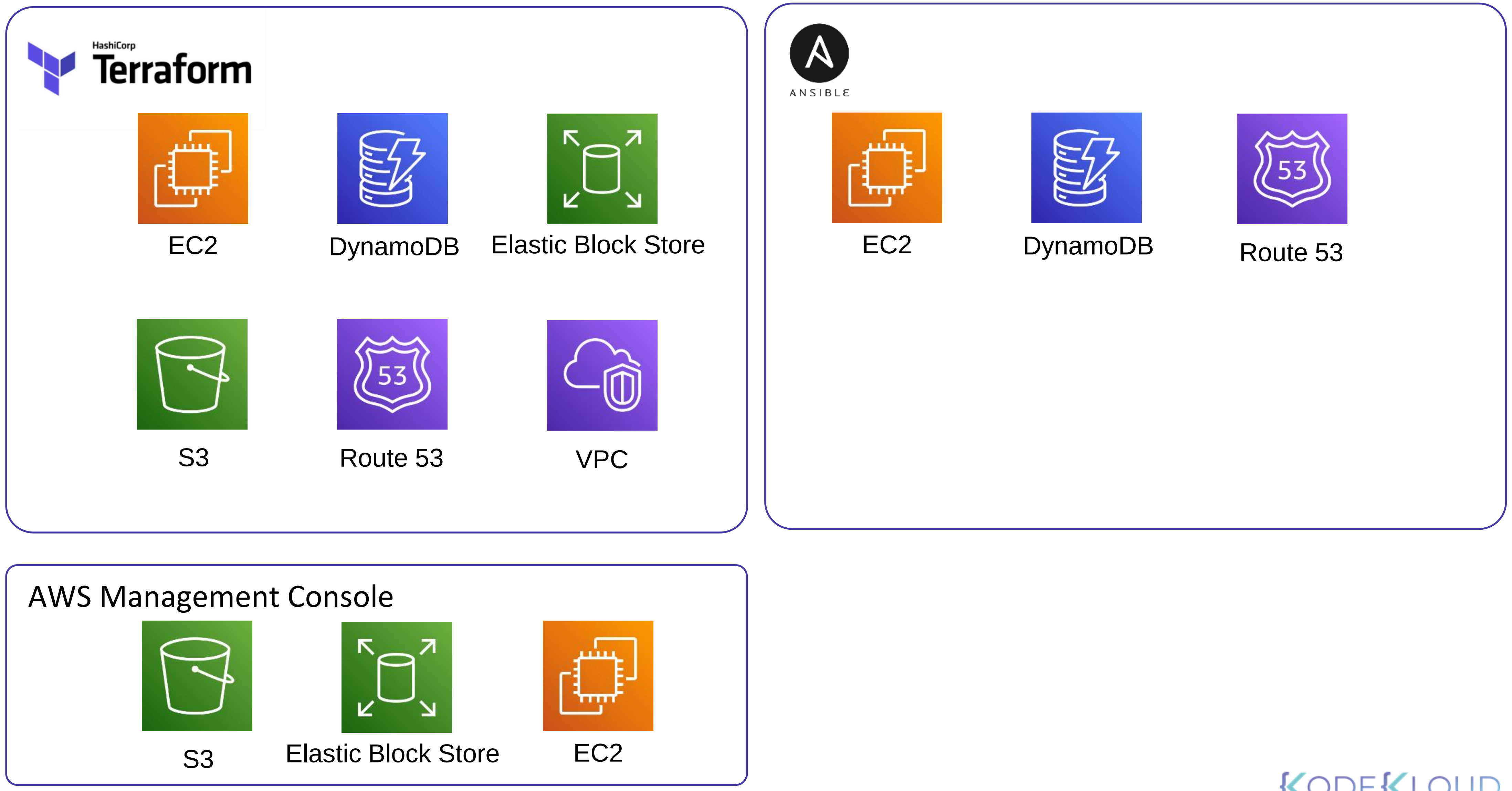
```
>_

 $ terraform plan
 ----
2020/10/18 22:08:30 [INFO] Terraform version: 0.13.0
2020/10/18 22:08:30 [INFO] Go runtime version: go1.14.2
2020/10/18 22:08:30 [INFO] CLI args: []string{"C:\\Windows\\system32\\terraform.exe", "plan"}
2020/10/18 22:08:30 [DEBUG] Attempting to open CLI config file: C:\Users\vpala\AppData\Roaming\terraform.rc
2020/10/18 22:08:30 [DEBUG] File doesn't exist, but doesn't need to. Ignoring.
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory terraform.d/plugins
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory C:\Users\vpala\AppData\Roaming\terraform.d\plugins
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory
C:\Users\vpala\AppData\Roaming\HashiCorp\Terraform\plugins
2020/10/18 22:08:30 [INFO] CLI command args: []string{"plan"}
2020/10/18 22:08:30 [WARN] Log levels other than TRACE are currently unreliable, and are supported only for backward
compatibility.
  Use TF_LOG=TRACE to see Terraform's internal logs.
  ----
2020/10/18 22:08:30 [DEBUG] New state was assigned lineage "f413959c-538a-f9ce-524e-1615073518d4"
2020/10/18 22:08:30 [DEBUG] checking for provisioner in "."
2020/10/18 22:08:30 [DEBUG] checking for provisioner in "C:\\Windows\\system32"
2020/10/18 22:08:30 [INFO] Failed to read plugin lock file .terraform\plugins\windows_amd64\lock.json: open
.terraform\plugins\windows_amd64\lock.json: The system cannot find the path specified.
2020/10/18 22:08:30 [INFO] backend/local: starting Plan operation
2020-10-18T22:08:30.625-0400 [INFO]  plugin: configuring client automatic mTLS
2020-10-18T22:08:30.646-0400 [DEBUG] plugin: starting plugin:
path=.terraform/plugins/registry.terraform.io/hashicorp/aws/3.11.0/windows_amd64/terraform-provider-aws_v3.11.0_x5.exe
args=[.terraform/plugins/registry.terraform.io/hashicorp/aws/3.11.0/windows_amd64/terraform-provider-aws_v3.11.0_x5.exe]
2020-10-18T22:08:30.935-0400 [DEBUG] plugin: plugin started:
path=.terraform/plugins/registry.terraform.io/hashicorp/aws/3.11.0/windows_amd64/terraform-provider-aws_v3.11.0_x5.exe
pid=34016
2020-10-18T22:08:30.935-0400 [DEBUG] plugin: waiting for RPC address:
path=.terraform/plugins/registry.terraform.io/hashicorp/aws/3.11.0/windows_amd64/terraform-provider-aws_v3.11.0_x5.exe
2020-10-18T22:08:30.974-0400 [INFO]  plugin.terraform-provider-aws_v3.11.0_x5.exe: configuring server automatic mTLS:
```

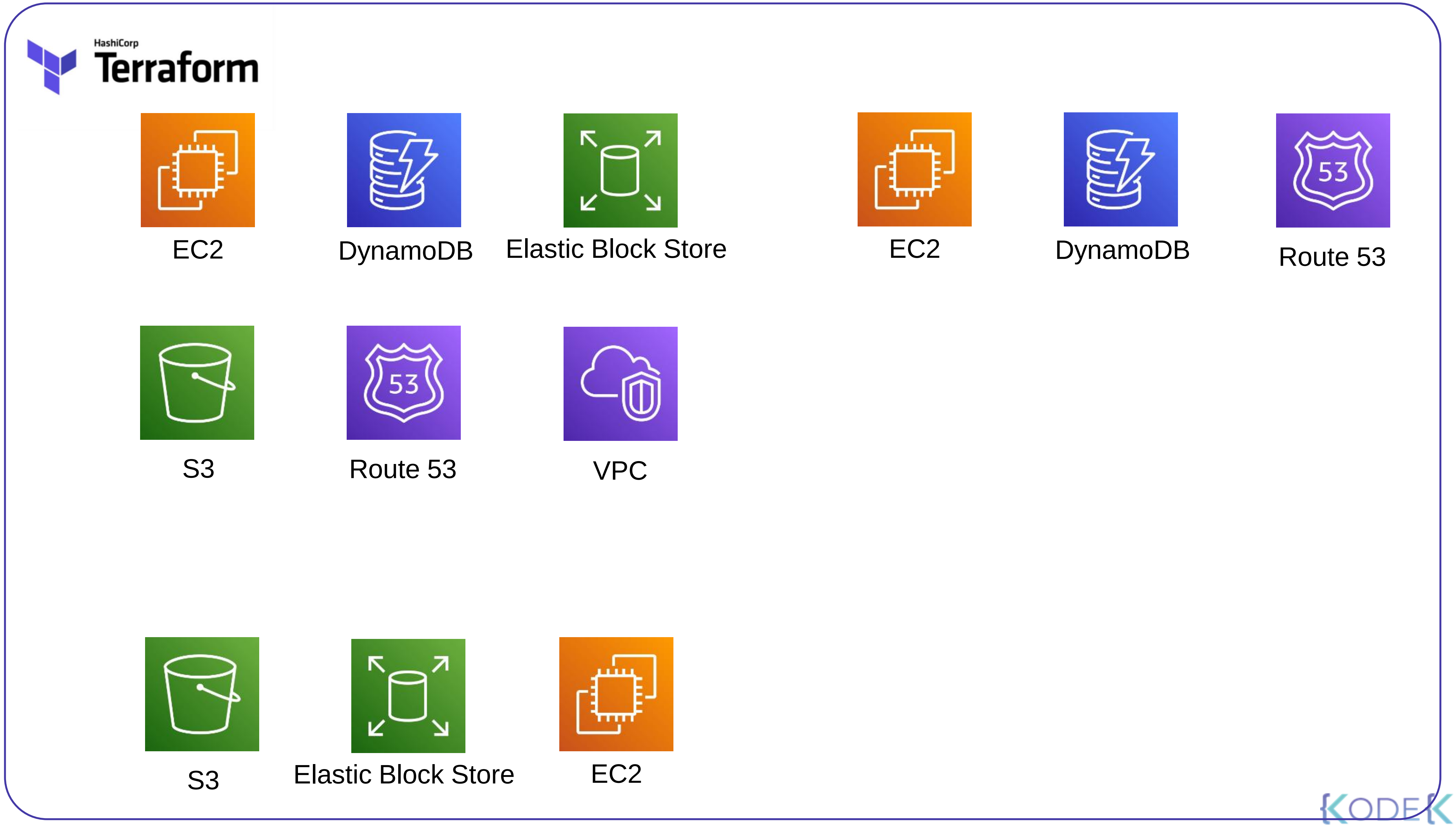


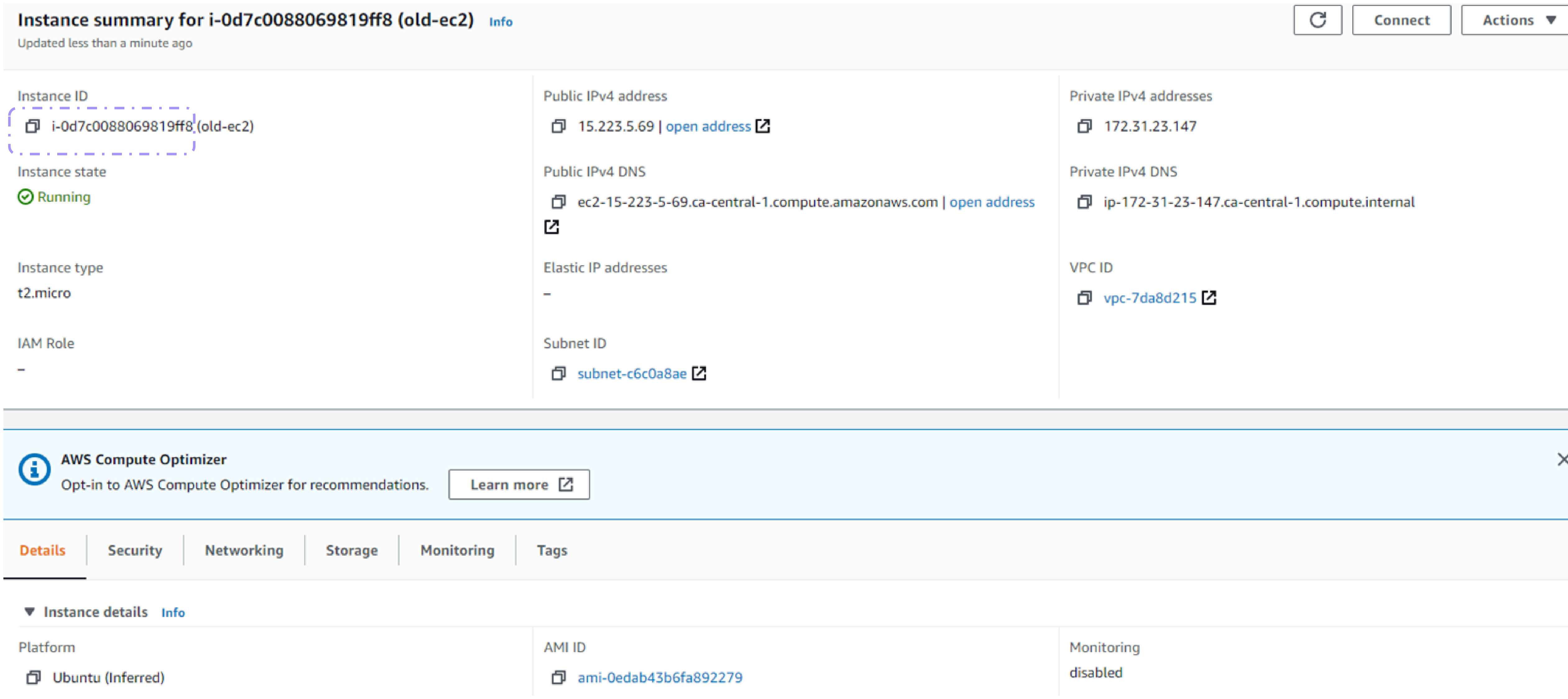
```
$ export TF_LOG_PATH=/tmp/terraform.log


$ head -10 /tmp/terraform.logs
----
2020/10/18 22:08:30 [INFO] Terraform version: 0.13.0
2020/10/18 22:08:30 [INFO] Go runtime version: go1.14.2
2020/10/18 22:08:30 [INFO] CLI args: []string{"C:\\Windows\\system32\\terraform.exe",
"plan"}
2020/10/18 22:08:30 [DEBUG] Attempting to open CLI config file:
C:\Users\vpala\AppData\Roaming\terraform.rc
2020/10/18 22:08:30 [DEBUG] File doesn't exist, but doesn't need to. Ignoring.
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory
terraform.d/plugins
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory
C:\Users\vpala\AppData\Roaming\terraform.d\plugins
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory
C:\Users\vpala\AppData\Roaming\HashiCorp\Terraform\plugins
2020/10/18 22:08:30 [INFO] CLI command args: []string{"plan"}


$ unset TF_LOG_PATH
```

KODEKLOUD

Terraform Import


HashiCorp
Terraform
EC2
DynamoDB
Elastic Block Store
S3
Route 53
VPC
ANSIBLE
EC2
DynamoDB
Route 53
AWS Management Console
S3
Elastic Block Store
EC2


HashiCorp Terraform
EC2
DynamoDB
Elastic Block Store
EC2
DynamoDB
Route 53
S3
Route 53
VPC
S3
Elastic Block Store
EC2



Instance summary for i-0d7c0088069819ff8 (old-ec2) Info

Updated less than a minute ago

Instance ID	Public IPv4 address	Private IPv4 addresses
i-0d7c0088069819ff8 (old-ec2)	15.223.5.69 | open address	172.31.23.147
Instance state	Public IPv4 DNS	Private IPv4 DNS
Running	ec2-15-223-5-69.ca-central-1.compute.amazonaws.com | open address	ip-172-31-23-147.ca-central-1.compute.internal
Instance type	Elastic IP addresses	VPC ID
t2.micro	–	vpc-7da8d215
IAM Role	Subnet ID	
–	subnet-c6c0a8ae	

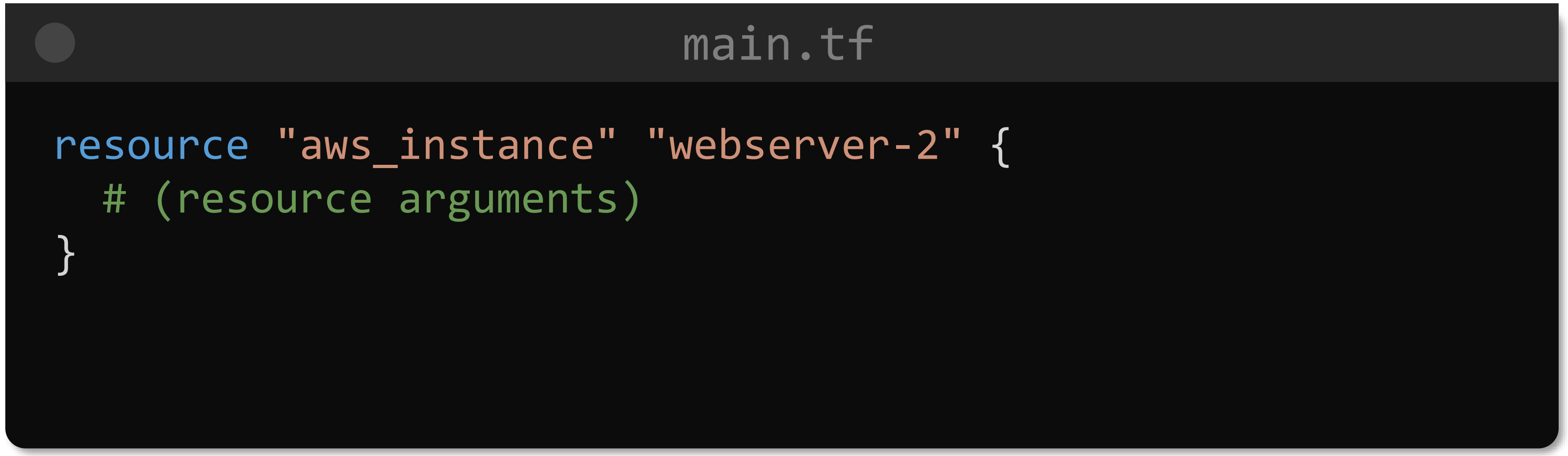
AWS Compute Optimizer

Opt-in to AWS Compute Optimizer for recommendations. Learn more

▼ Instance details Info

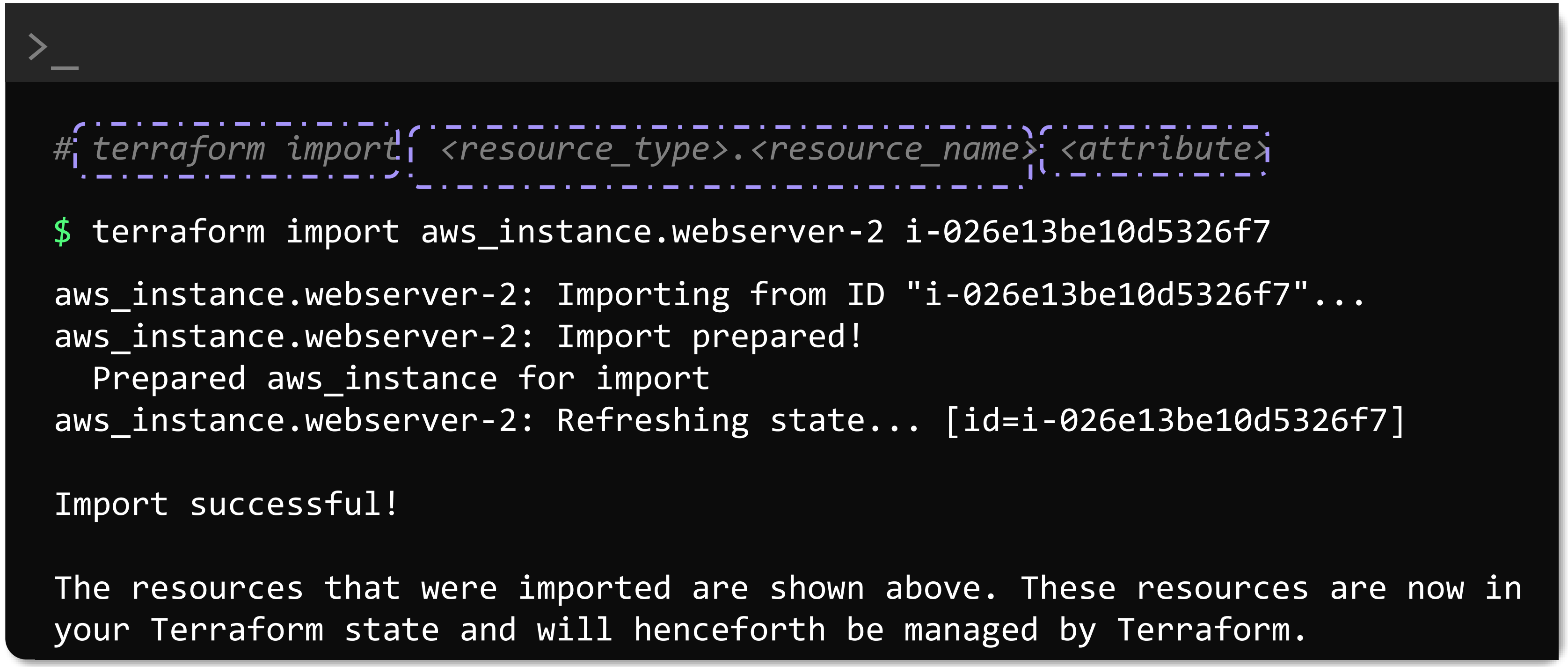
Platform	AMI ID	Monitoring
Ubuntu (Inferred)	ami-0edab43b6fa892279	disabled

Terraform Import

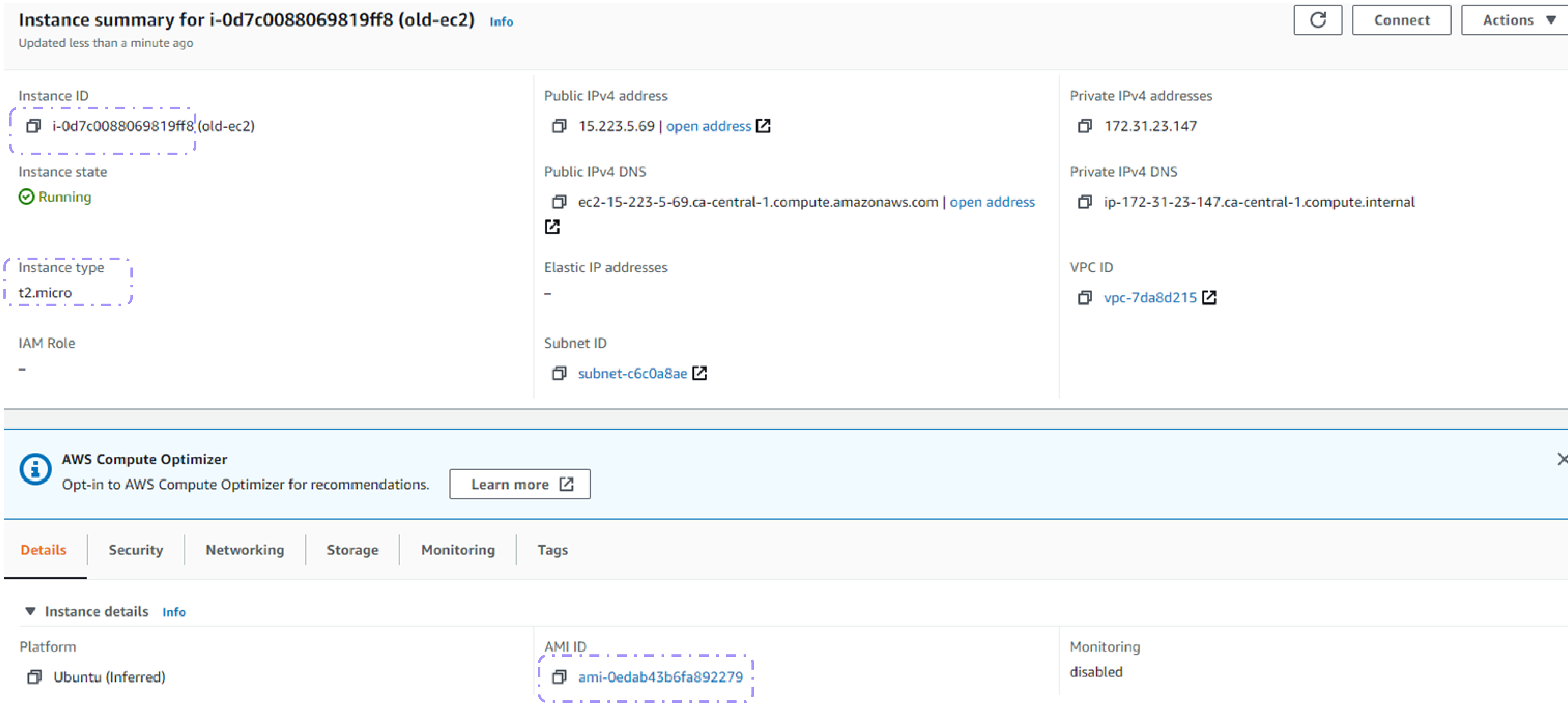


main.tf

```
resource "aws_instance" "webserver-2" {
  # (resource arguments)
}
```



```
# terraform import <resource_type>.<resource_name> <attribute>

$ terraform import aws_instance.webserver-2 i-026e13be10d5326f7

aws_instance.webserver-2: Importing from ID "i-026e13be10d5326f7"...
aws_instance.webserver-2: Import prepared!
  Prepared aws_instance for import
aws_instance.webserver-2: Refreshing state... [id=i-026e13be10d5326f7]

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.
```



Instance summary for i-0d7c0088069819ff8 (old-ec2) Info

Updated less than a minute ago

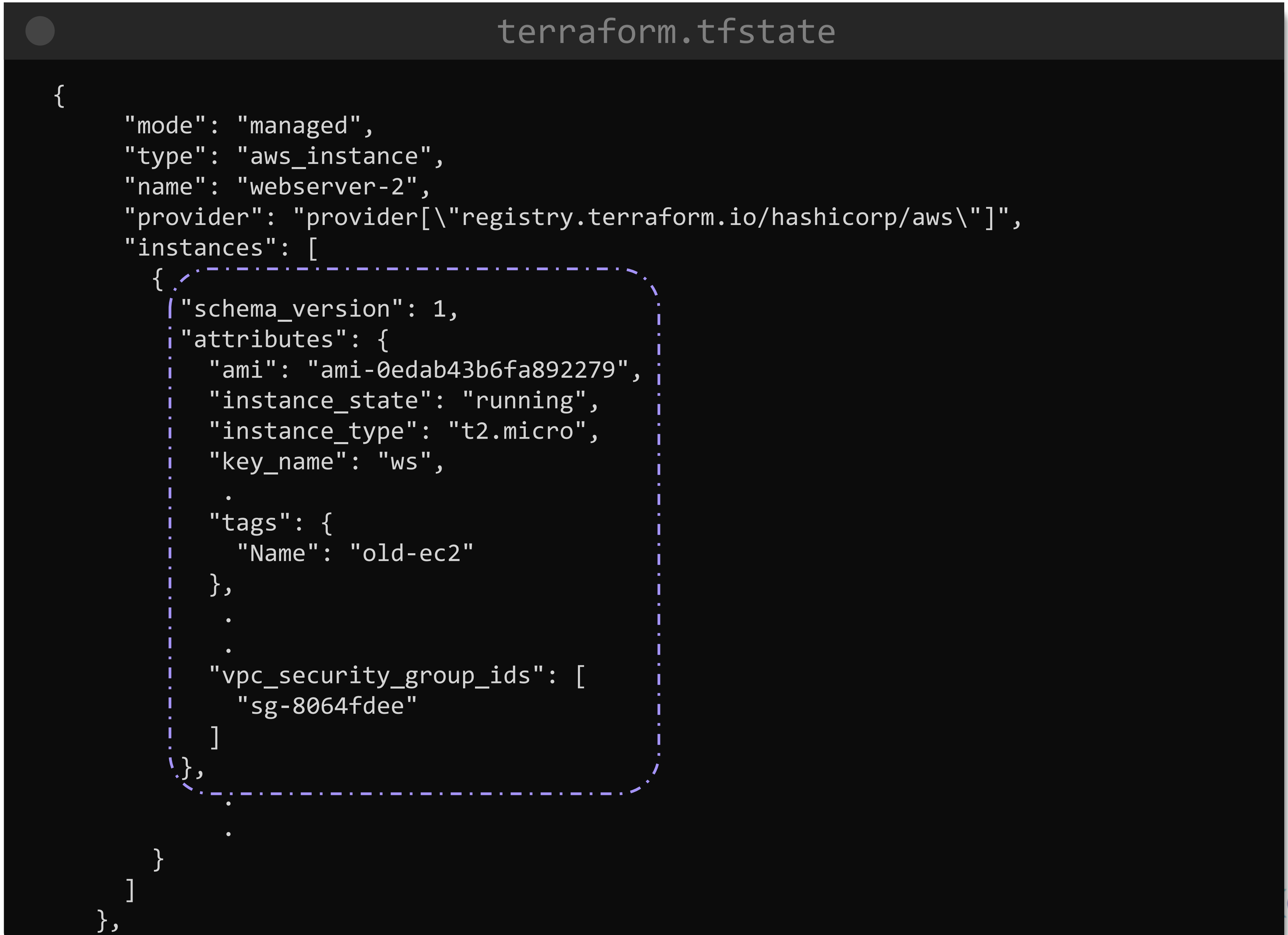
Instance ID	Public IPv4 address	Private IPv4 addresses
i-0d7c0088069819ff8 (old-ec2)	15.223.5.69 | open address	172.31.23.147
Instance state	Public IPv4 DNS	Private IPv4 DNS
Running	ec2-15-223-5-69.ca-central-1.compute.amazonaws.com | open address	ip-172-31-23-147.ca-central-1.compute.internal
Instance type	Elastic IP addresses	VPC ID
t2.micro	–	vpc-7da8d215
IAM Role	Subnet ID	
–	subnet-c6c0a8ae	

AWS Compute Optimizer

Opt-in to AWS Compute Optimizer for recommendations.

Instance details Info

Platform	AMI ID	Monitoring
Ubuntu (Inferred)	ami-0edab43b6fa892279	disabled


terraform.tfstate

```
{
        "mode": "managed",
        "type": "aws_instance",
        "name": "webserver-2",
        "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
        "instances": [
          {
           "schema_version": 1,
           "attributes": {
             "ami": "ami-0edab43b6fa892279",
             "instance_state": "running",
             "instance_type": "t2.micro",
             "key_name": "ws",
              .
             "tags": {
               "Name": "old-ec2"
             },
              .
              .
             "vpc_security_group_ids": [
               "sg-8064fdee"
             ]
           },
              .
              .
          }
        ]
    },
```

main.tf

```
resource "aws_instance" "webserver-2" {

}
```

```
>_
```

main.tf

```
resource "aws_instance" "webserver-2" {
    ami                    = "ami-0edab43b6fa892279"
    instance_type          = "t2.micro"
    key_name               = "ws"
    vpc_security_group_ids = ["sg-8064fdee"]

}
```

```
$ 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_instance.webserver-2: Refreshing state... [id=i-0d7c0088069819ff8]

------------------------------------------------------------------------

No changes. Infrastructure is up-to-date.

This means that Terraform did not detect any differences between your
configuration and real physical resources that exist. As a result, no
actions need to be performed
```

KODEKLOUD

Terraform Modules

Root Module

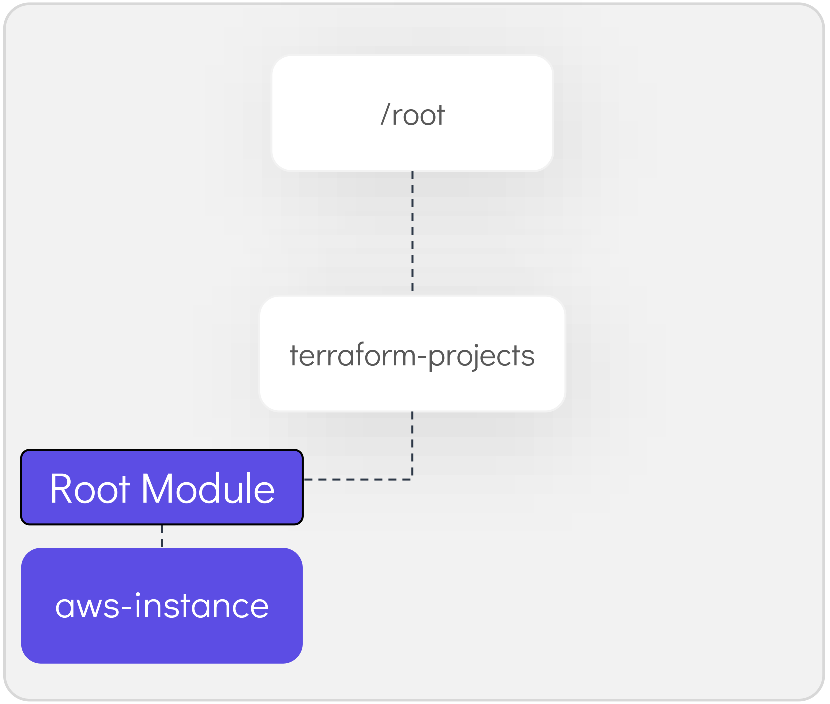
```
$ ls /root/terraform-projects/aws-instance
main.tf      variables.tf
```

main.tf

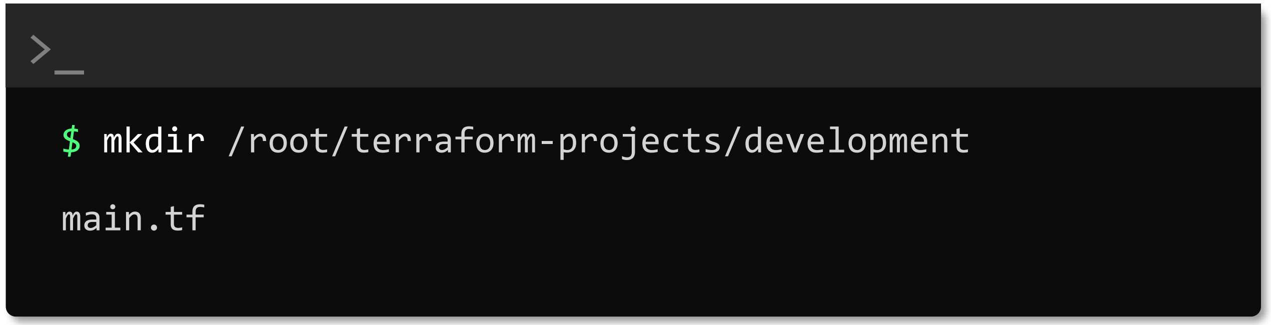
```
resource "aws_instance" "webserver" {
    ami = var.ami
    instance_type = var.instance_type
    key_name = var.key
}
```

variables.tf

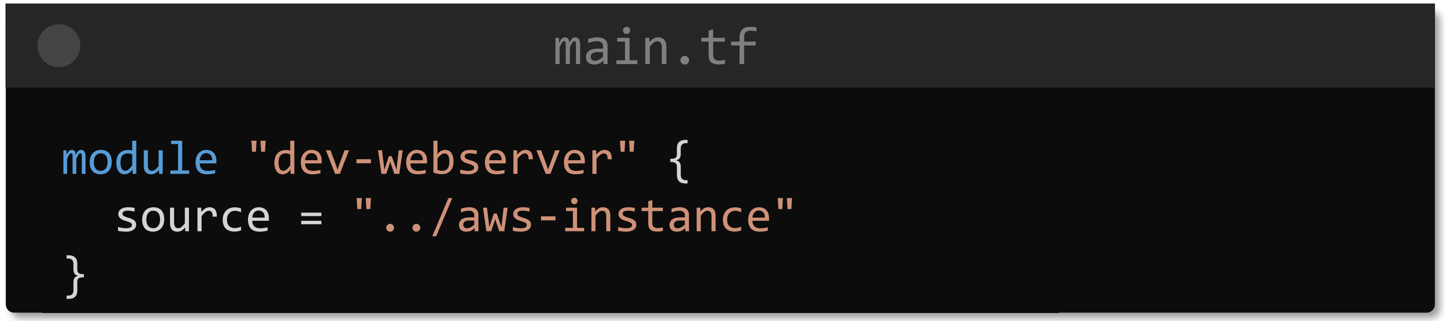
```
variable ami {
  type        = string
  default     = "ami-0edab43b6fa892279"
  description = "Ubuntu AMI ID in the ca-
central-1 region"
}
```



Root Module

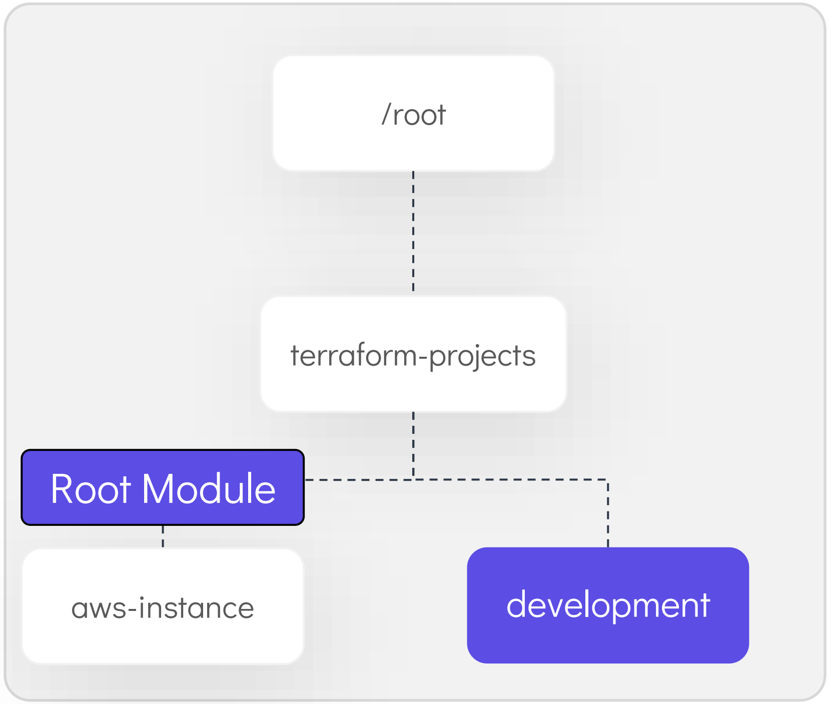


```
$ mkdir /root/terraform-projects/development

main.tf
```

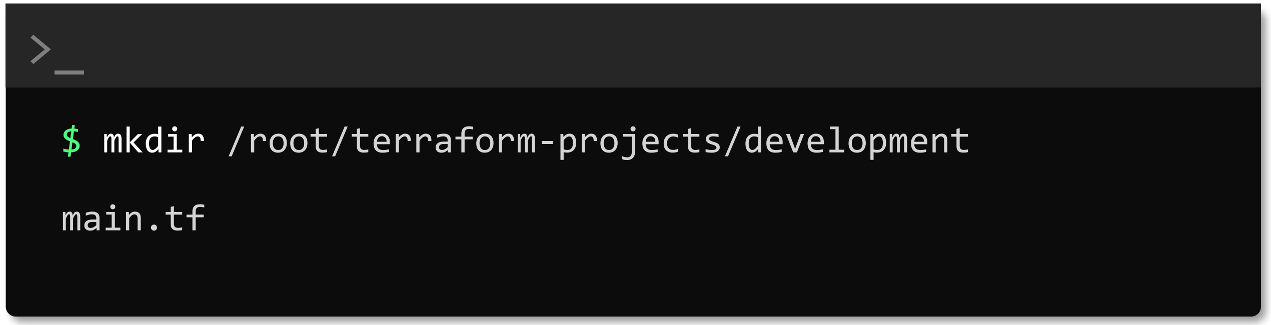


main.tf

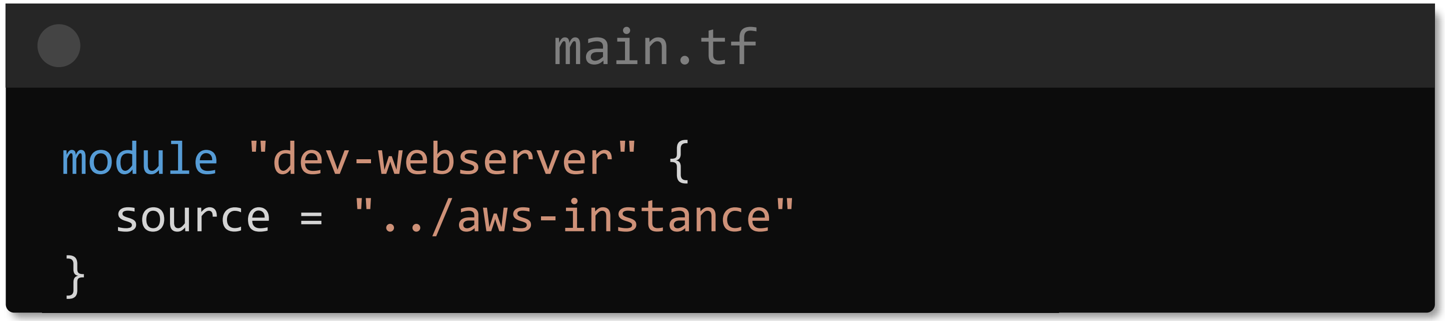
```
module "dev-webserver" {
  source = "../aws-instance"
}
```



Root Module

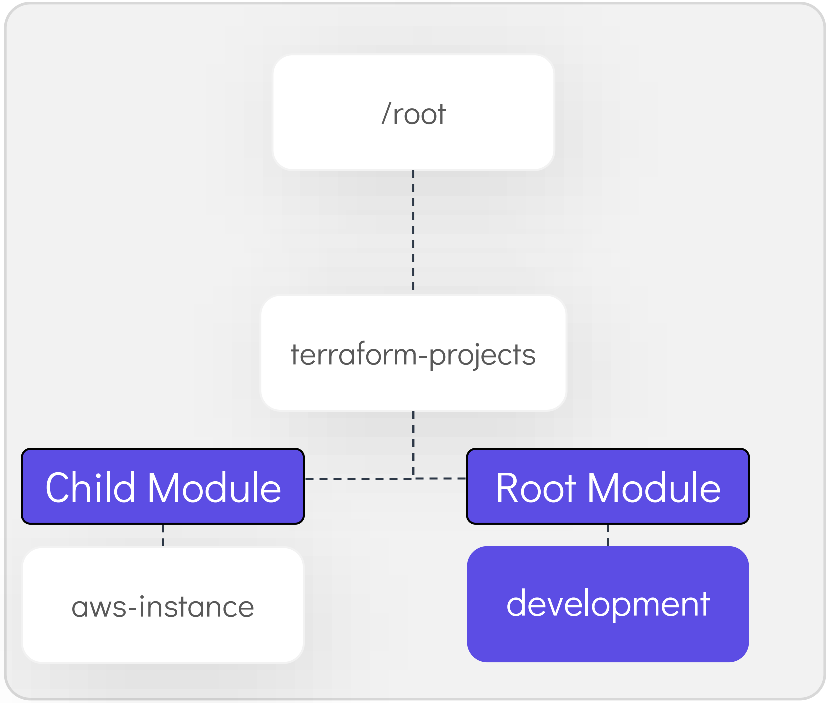


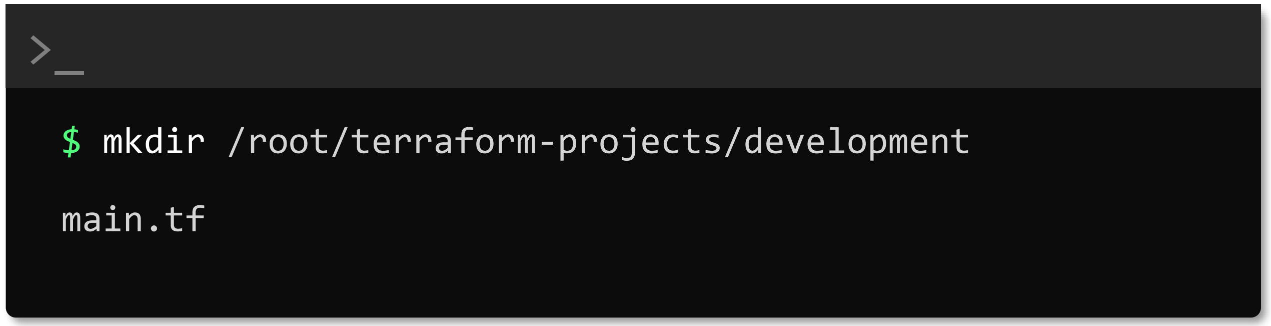
```
$ mkdir /root/terraform-projects/development

main.tf
```



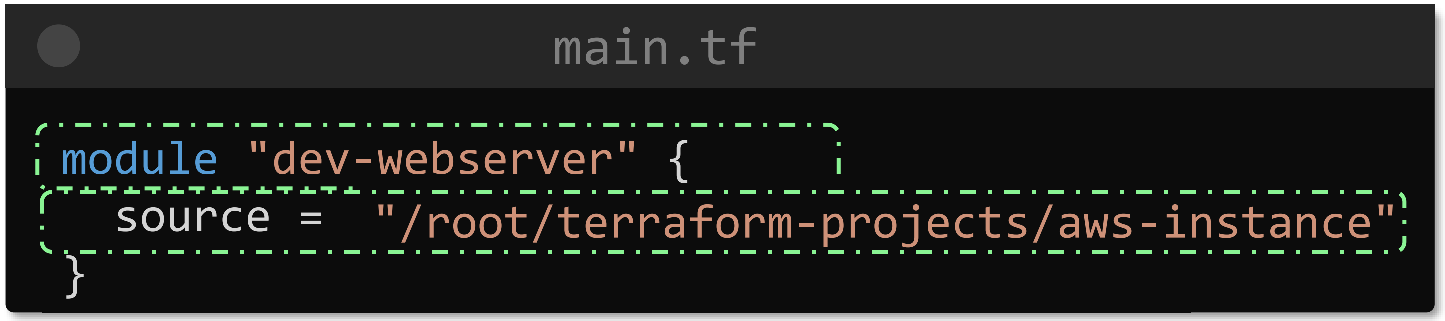
main.tf

```
module "dev-webserver" {
  source = "../aws-instance"
}
```



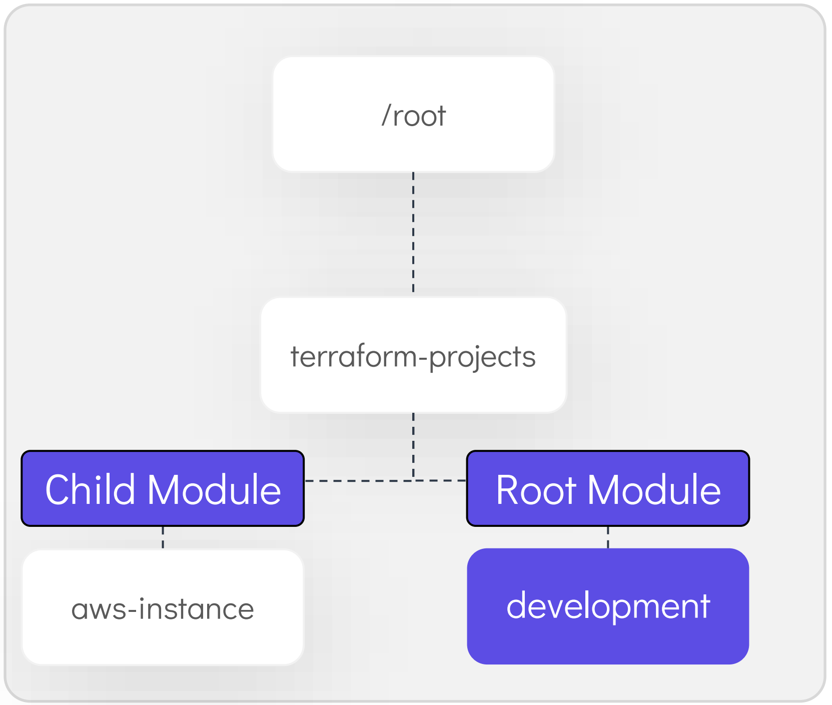


```
$ mkdir /root/terraform-projects/development

main.tf
```



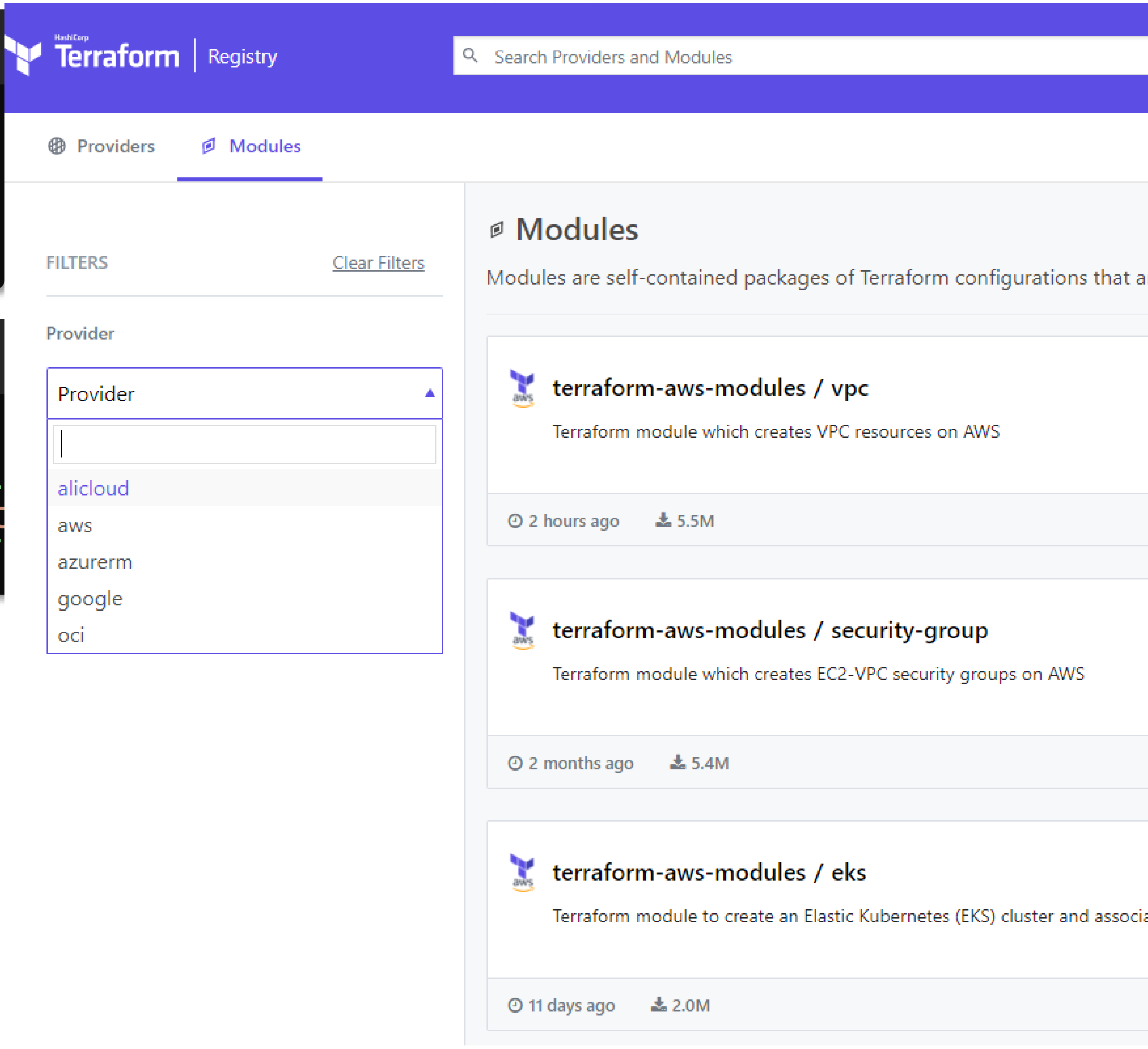
main.tf

```
module "dev-webserver" {
   source =  "/root/terraform-projects/aws-instance"
}
```



```
>_
$ mkdir /root/terraform-projects/development

main.tf
```

```
main.tf

module "dev-webserver" {
   source =  "/root/terraform-projects/aws-inst
}
```



Terraform | Registry

Search Providers and Modules

Providers   Modules

FILTERS   Clear Filters

Provider

Provider
- alicloud
- aws
- azurerm
- google
- oci

Modules

Modules are self-contained packages of Terraform configurations that a

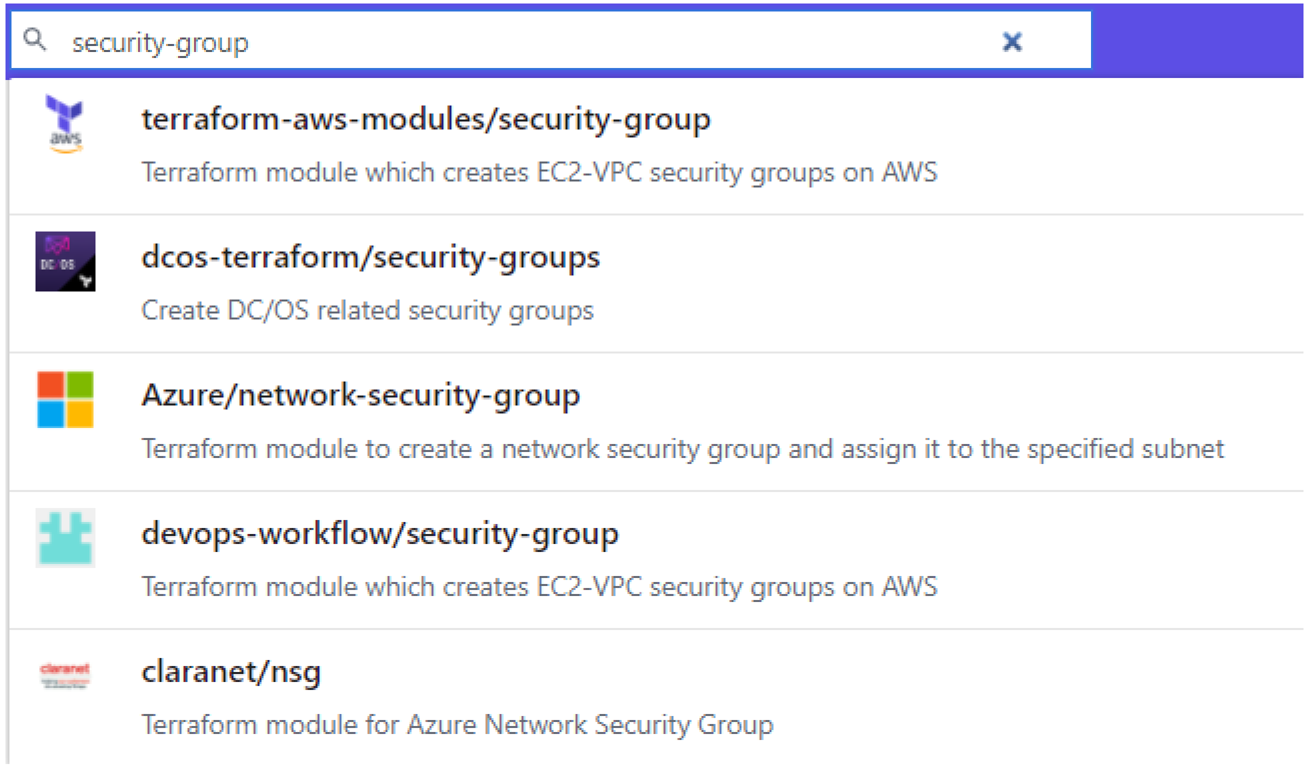
terraform-aws-modules / vpc
Terraform module which creates VPC resources on AWS
2 hours ago   5.5M

terraform-aws-modules / security-group
Terraform module which creates EC2-VPC security groups on AWS
2 months ago   5.4M

terraform-aws-modules / eks
Terraform module to create an Elastic Kubernetes (EKS) cluster and associa
11 days ago   2.0M

https://registry.terraform.io/browse/modules

Terraform Registry



security-group

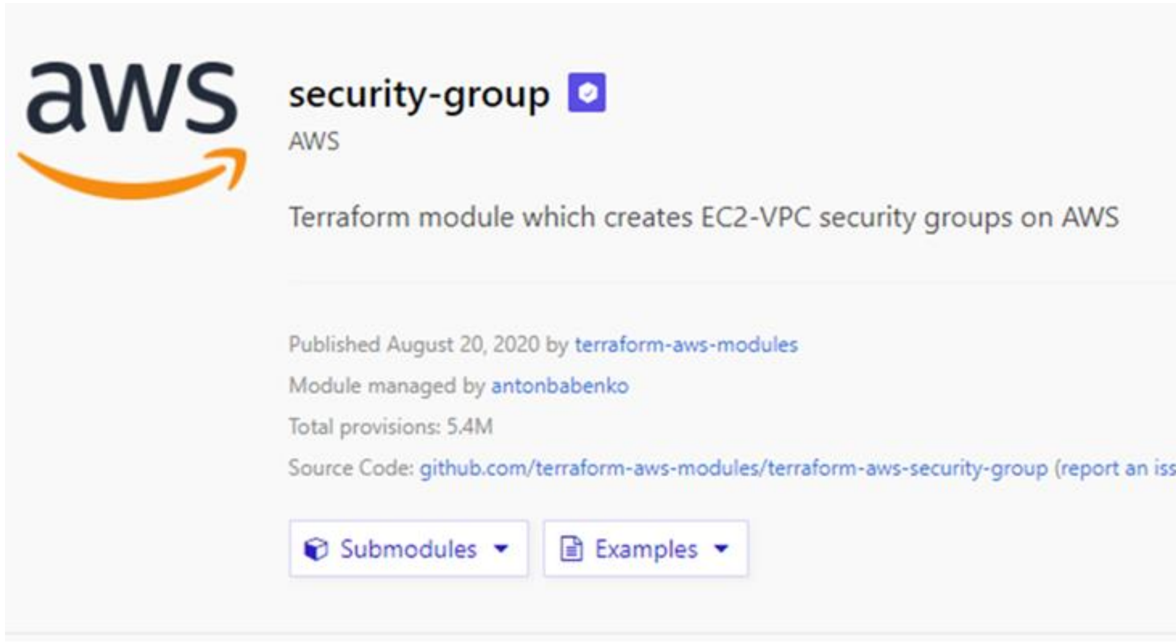
terraform-aws-modules/security-group
Terraform module which creates EC2-VPC security groups on AWS

dcos-terraform/security-groups
Create DC/OS related security groups

Azure/network-security-group
Terraform module to create a network security group and assign it to the specified subnet

devops-workflow/security-group
Terraform module which creates EC2-VPC security groups on AWS

claranet/nsg
Terraform module for Azure Network Security Group

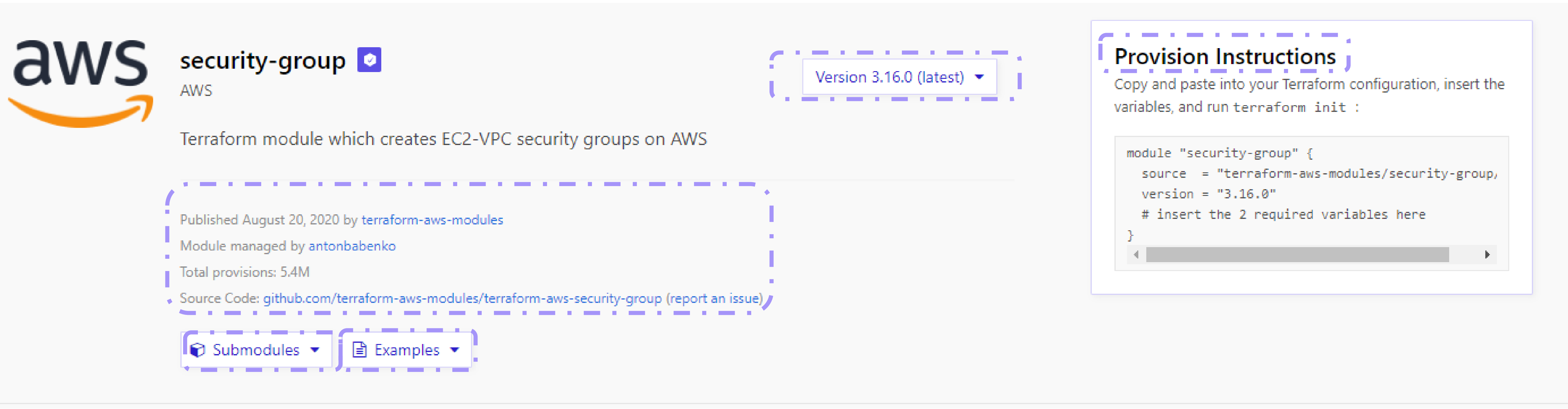


security-group
AWS

Terraform module which creates EC2-VPC security groups on AWS

Published August 20, 2020 by terraform-aws-modules
Module managed by antonbabenko
Total provisions: 5.4M
Source Code: github.com/terraform-aws-modules/terraform-aws-security-group (report an iss

Terraform Module



security-group

AWS

Version 3.16.0 (latest)

Terraform module which creates EC2-VPC security groups on AWS

Published August 20, 2020 by terraform-aws-modules

Module managed by antonbabenko

Total provisions: 5.4M

Source Code: github.com/terraform-aws-modules/terraform-aws-security-group (report an issue)

Submodules

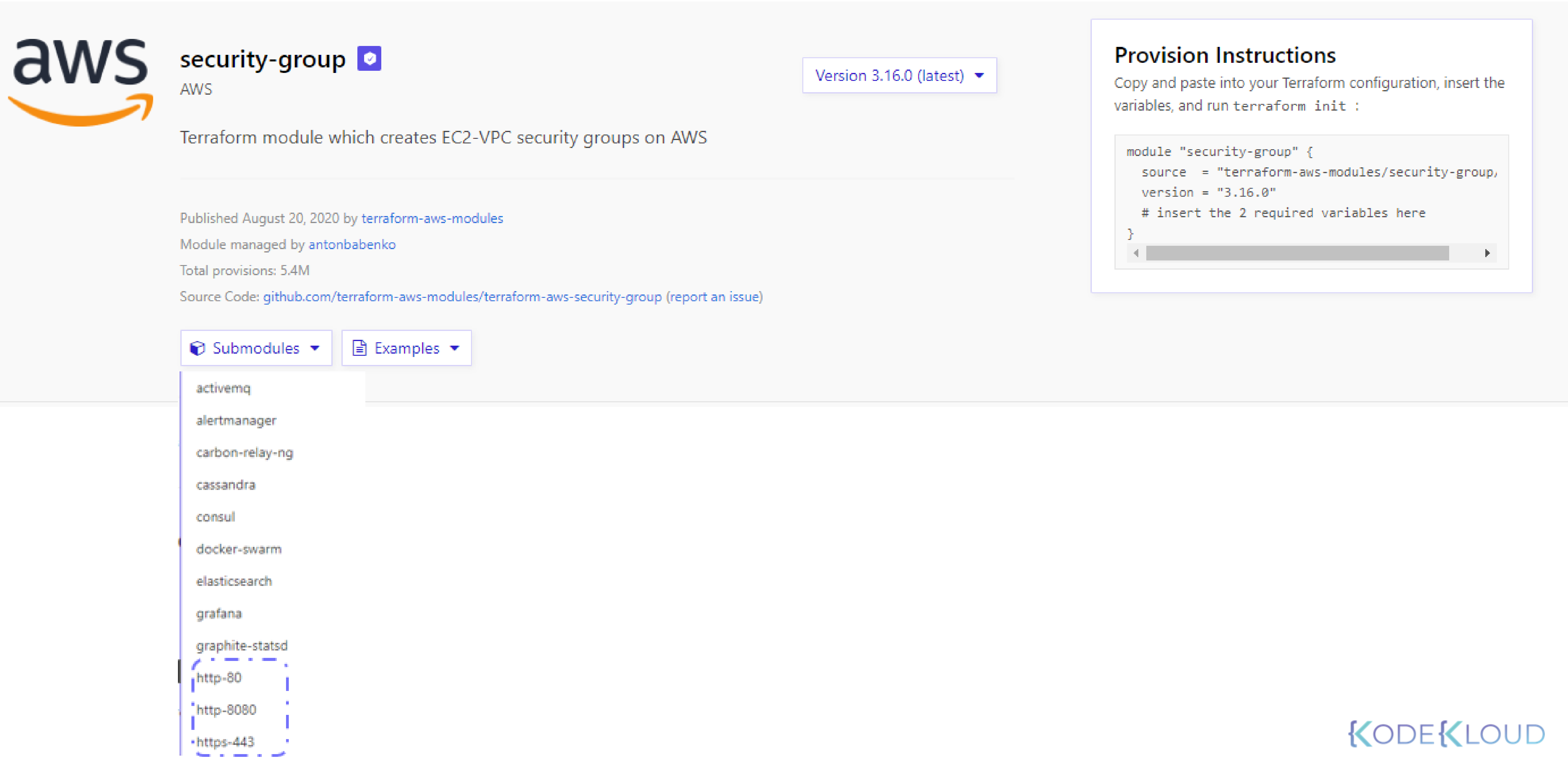
Examples

Provision Instructions

Copy and paste into your Terraform configuration, insert the variables, and run `terraform init` :

```
module "security-group" {
  source  = "terraform-aws-modules/security-group/
  version = "3.16.0"
  # insert the 2 required variables here
}
```

Terraform Module



security-group

AWS

Version 3.16.0 (latest)

Terraform module which creates EC2-VPC security groups on AWS

Published August 20, 2020 by terraform-aws-modules

Module managed by antonbabenko

Total provisions: 5.4M

Source Code: github.com/terraform-aws-modules/terraform-aws-security-group (report an issue)

Submodules

Examples

- activemq
- alertmanager
- carbon-relay-ng
- cassandra
- consul
- docker-swarm
- elasticsearch
- grafana
- graphite-statsd
- http-80
- http-8080
- https-443

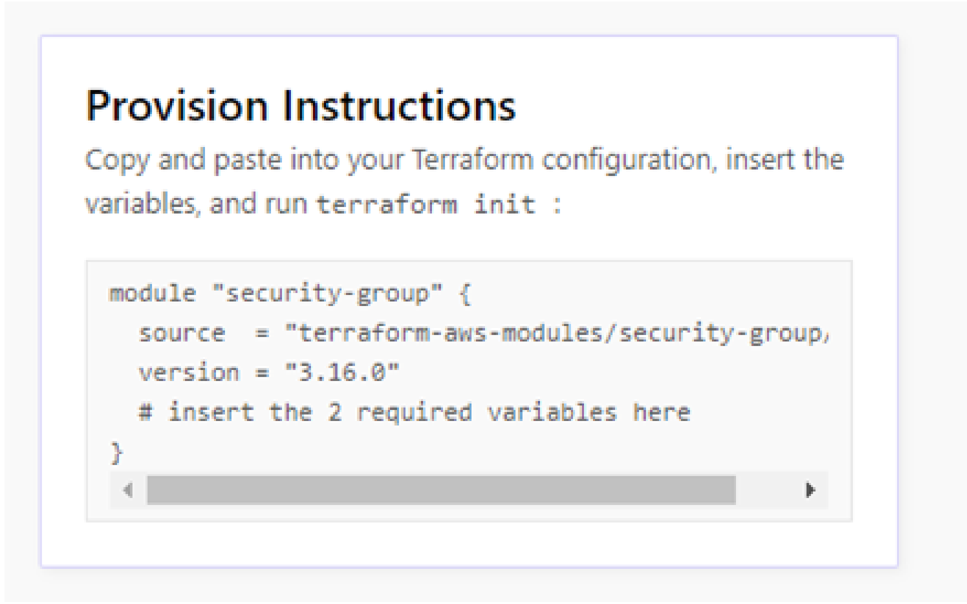
Provision Instructions

Copy and paste into your Terraform configuration, insert the variables, and run `terraform init` :

```
module "security-group" {
  source  = "terraform-aws-modules/security-group/
  version = "3.16.0"
  # insert the 2 required variables here
}
```

main.tf

```
module "security-group_ssh" {
  source  = "terraform-aws-modules/security-group/aws/modules/ssh"
  version = "3.16.0"
  # insert the 2 required variables here
  vpc_id = "vpc-7d8d215"
  ingress_cidr_blocks = [ "10.10.0.0/16"]
  name = "ssh-access"

}
```



Provision Instructions

Copy and paste into your Terraform configuration, insert the variables, and run `terraform init` :

```
module "security-group" {
  source  = "terraform-aws-modules/security-group/
  version = "3.16.0"
  # insert the 2 required variables here
}
```

```
$ terraform apply
Downloading terraform-aws-modules/security-group/aws 3.16.0 for security-group_ssh...
- security-group_ssh in .terraform\modules\security-group_ssh\modules\ssh
```

https://registry.terraform.io/modules/terraform-aws-modules/security-group/aws/latest/submodules/ssh

Simpler Configuration Files

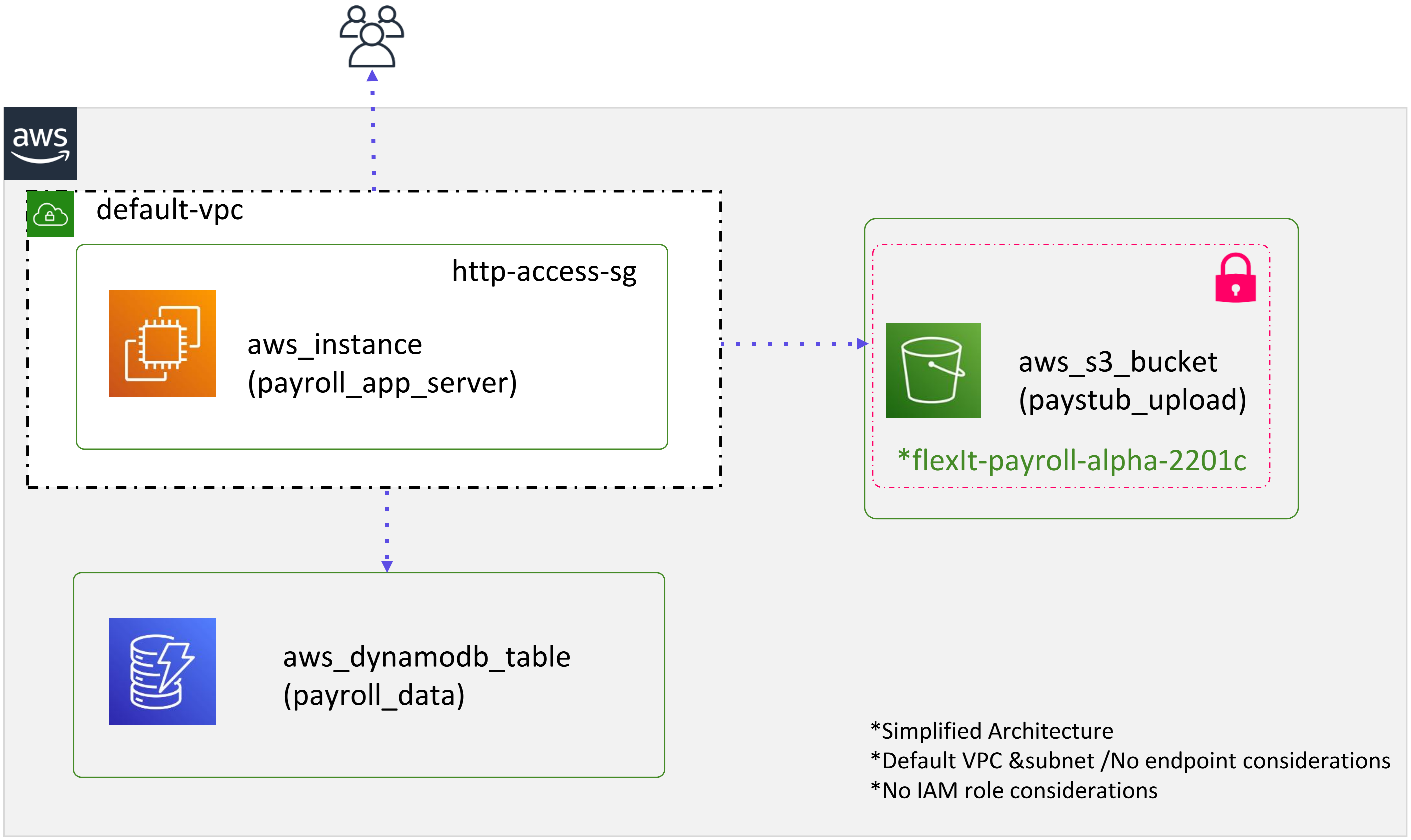
Lower Risk

Re-Usability

Standardized Configuration

KODEKLOUD

Creating and Using a Module

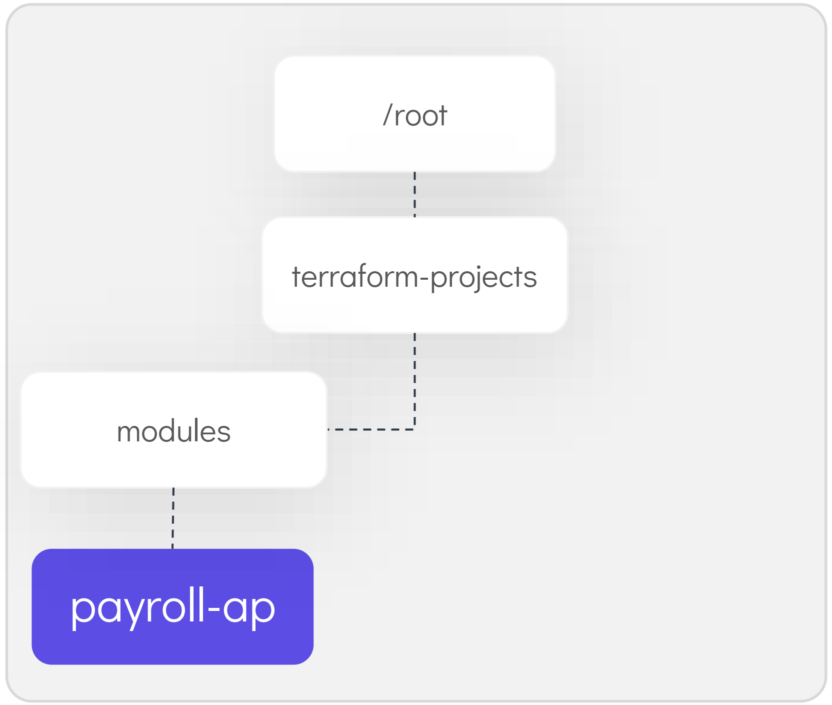


*Simplified Architecture
*Default VPC &subnet /No endpoint considerations
*No IAM role considerations

```
>_
$ mkdir /root/terraform-projects/modules/payroll-app

app_server.tf dynamodb_table.tf s3_bucket.tf variables.tf
```

app_server.tf

```
resource "aws_instance" "app_server" {
  ami             = var.ami
  instance_type =   "t2.medium"
  tags = {
    Name = "${var.app_region}-app-server"
  }
  depends_on = [ aws_dynamodb_table.payroll_db,
                 aws_s3_bucket.payroll_data
               ]
}
```

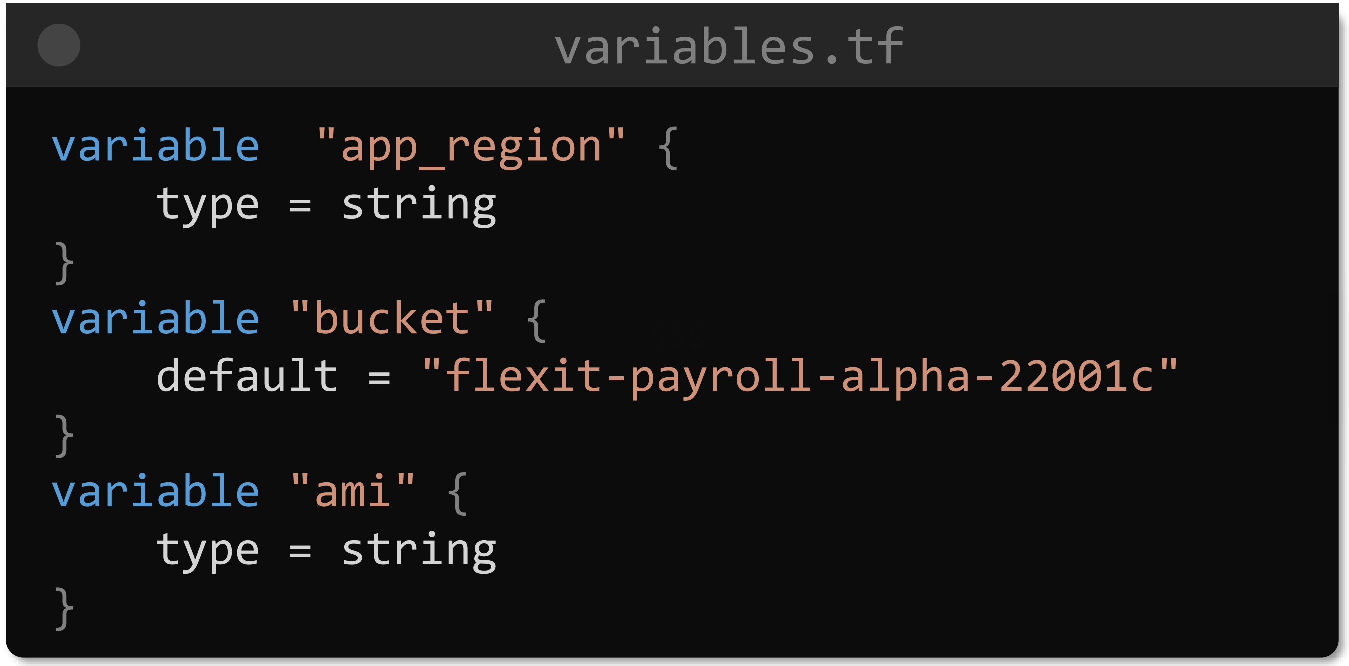


s3_bucket.tf

```
 resource  "aws_s3_bucket"  "payroll_data" {
  bucket = "${var.app_region}-${var.bucket}"
 }
```

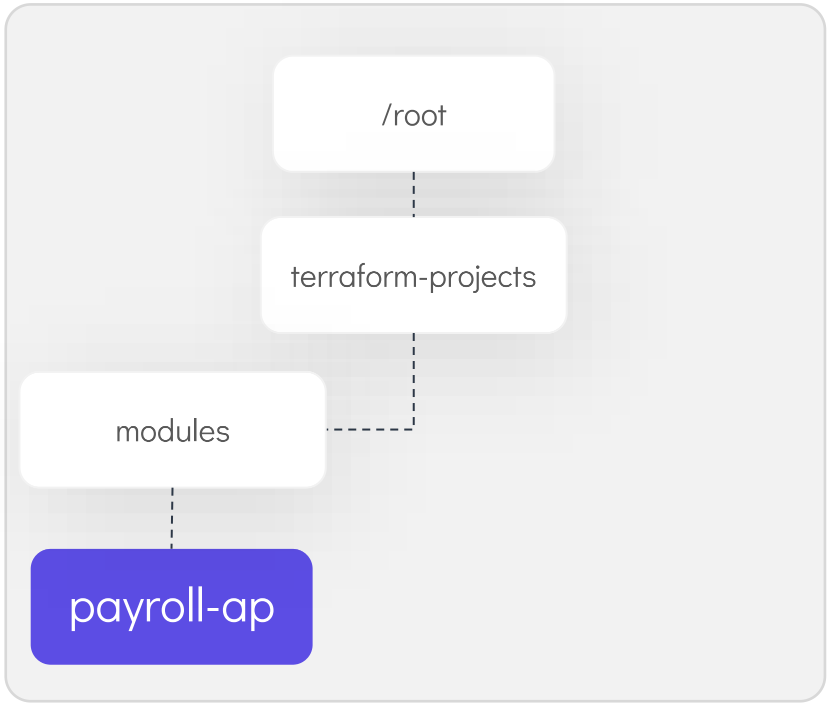
dynamodb_table.tf

```
resource "aws_dynamodb_table" "payroll_db" {
  name                = "user_data"
  billing_mode        = "PAY_PER_REQUEST"
  hash_key            = "EmployeeID"

  attribute {
    name = "EmployeeID"
    type = "N"
  }
}
```



variables.tf

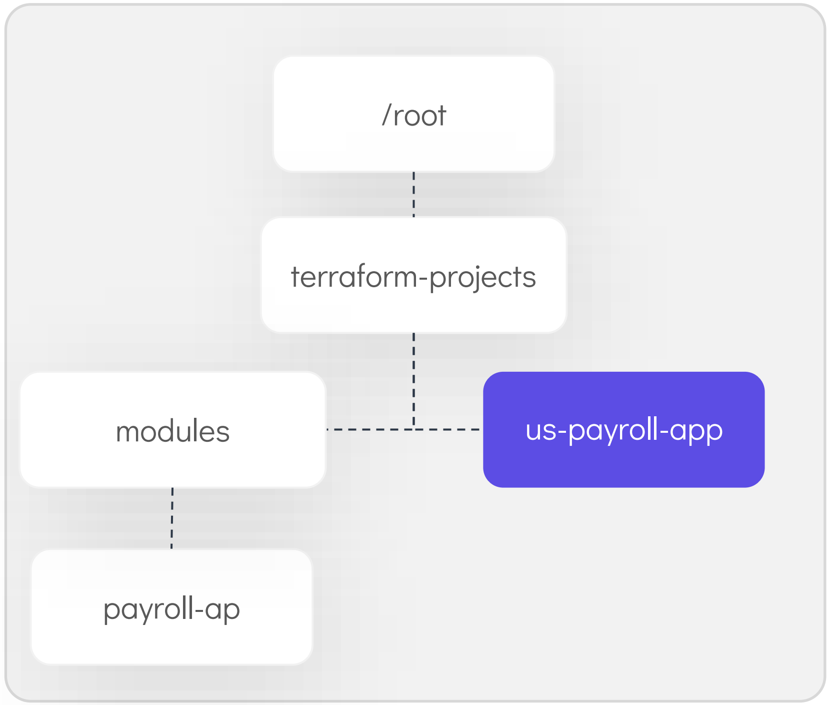
```
variable  "app_region" {
    type = string
}
variable "bucket" {
    default = "flexit-payroll-alpha-22001c"
}
variable "ami" {
    type = string
}
```

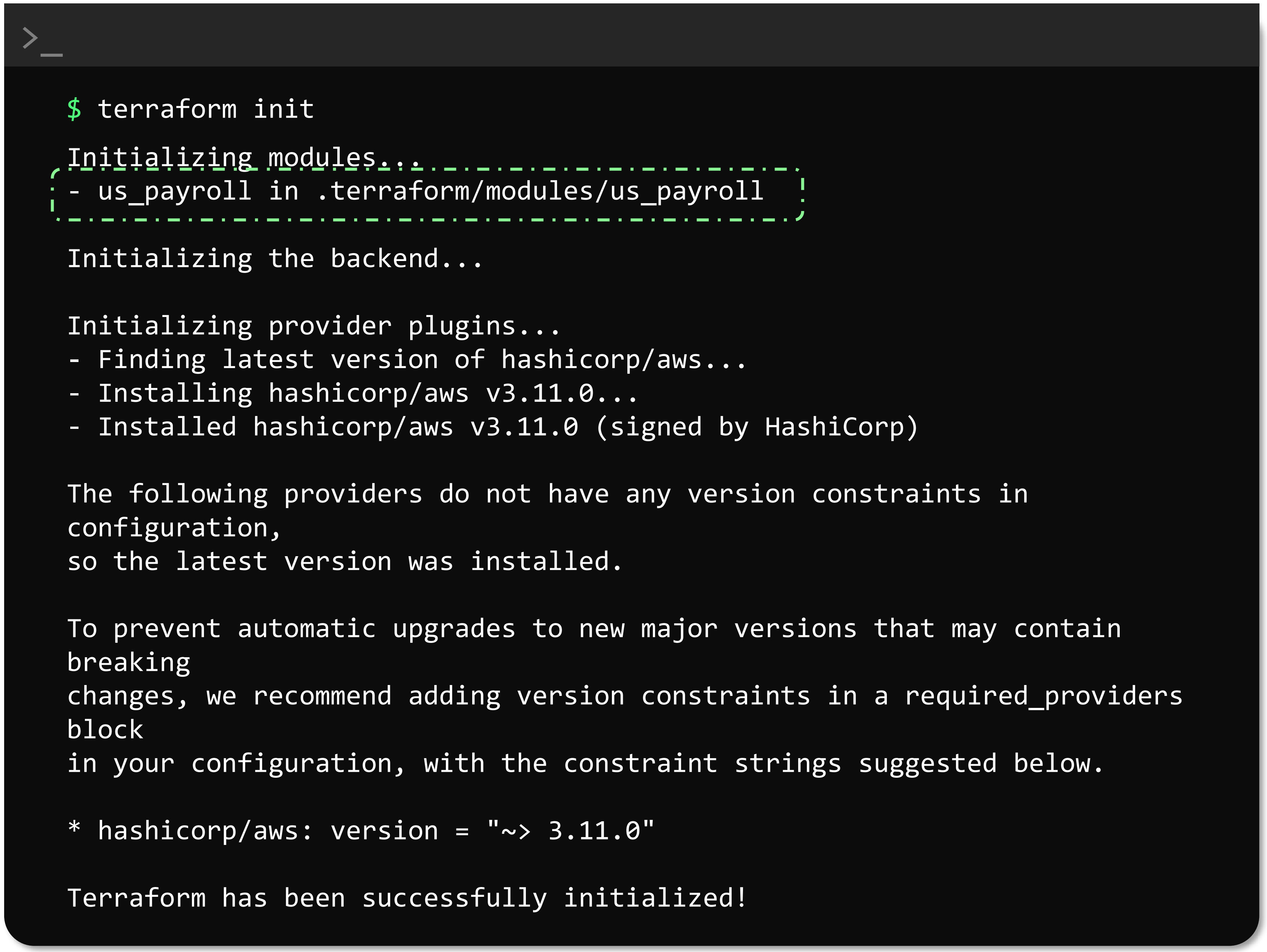


```
$ mkdir /root/terraform-projects/us-payroll-app

main.tf provider.tf
```

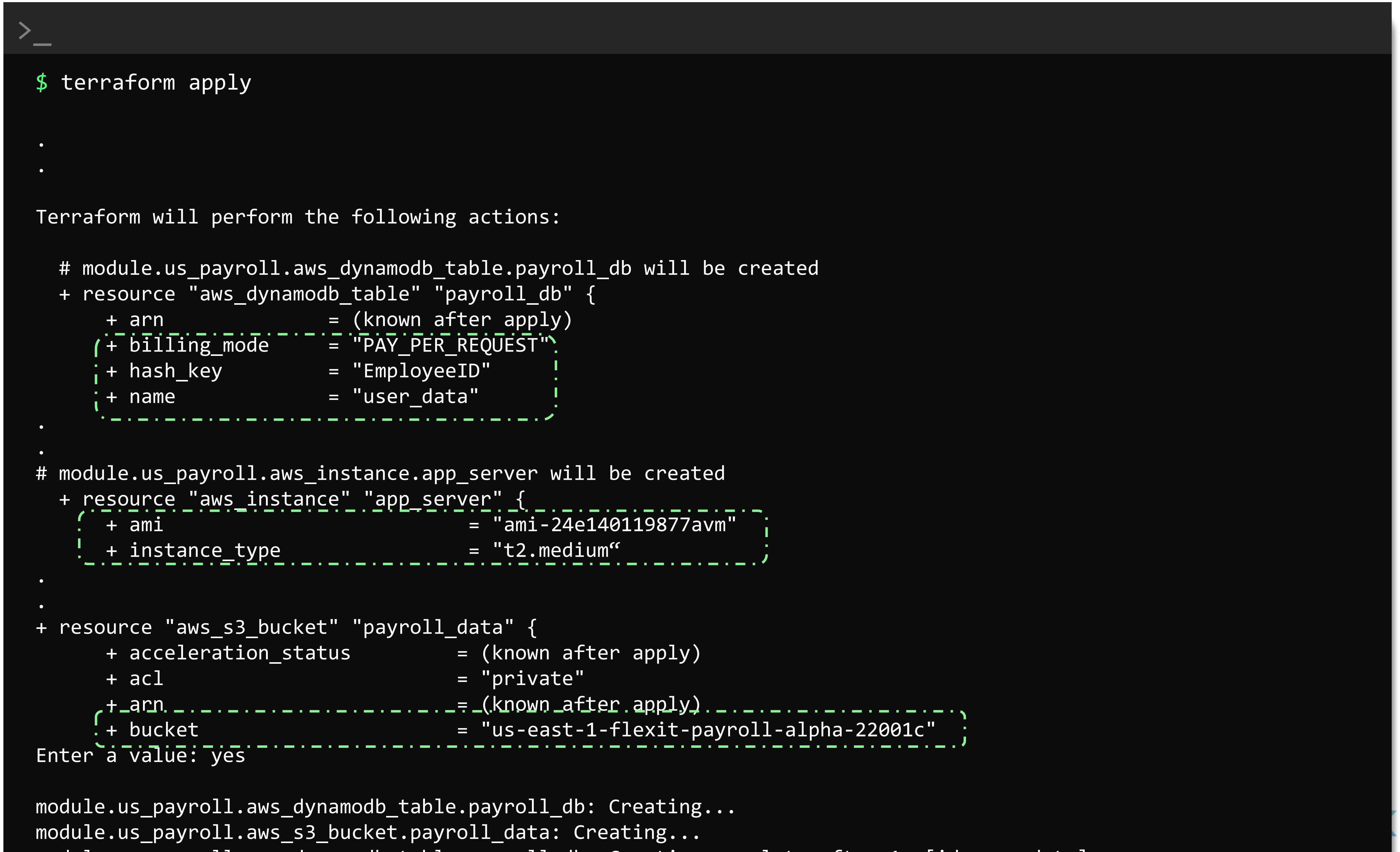
main.tf

```
module  "us_payroll" {
  source = "../modules/payroll-app"
  app_region = "us-east-1"
  ami        = "ami-24e140119877avm"
}
```





```
$ terraform init

Initializing modules...
- us_payroll in .terraform/modules/us_payroll

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v3.11.0...
- Installed hashicorp/aws v3.11.0 (signed by HashiCorp)

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 = "~> 3.11.0"

Terraform has been successfully initialized!
```



```
$ terraform apply

.
.

Terraform will perform the following actions:

  # module.us_payroll.aws_dynamodb_table.payroll_db will be created
  + resource "aws_dynamodb_table" "payroll_db" {
      + arn              = (known after apply)
      + billing_mode     = "PAY_PER_REQUEST"
      + hash_key         = "EmployeeID"
      + name             = "user_data"
.
.
# module.us_payroll.aws_instance.app_server will be created
  + resource "aws_instance" "app_server" {
      + ami                          = "ami-24e140119877avm"
      + instance_type                = "t2.medium“
.
.
+ resource "aws_s3_bucket" "payroll_data" {
      + acceleration_status         = (known after apply)
      + acl                         = "private"
      + arn                         = (known after apply)
      + bucket                      = "us-east-1-flexit-payroll-alpha-22001c"
Enter a value: yes

module.us_payroll.aws_dynamodb_table.payroll_db: Creating...
module.us_payroll.aws_s3_bucket.payroll_data: Creating...
```

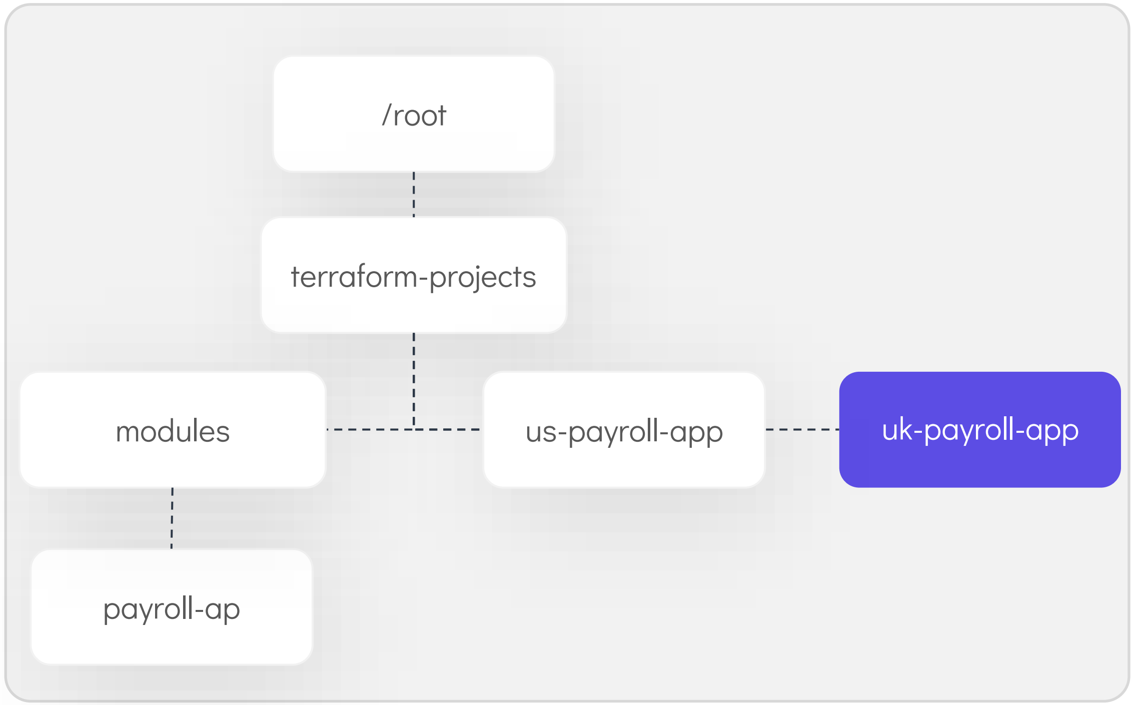
```
$ mkdir /root/terraform-projects/uk-payroll-app

main.tf provider.tf
```

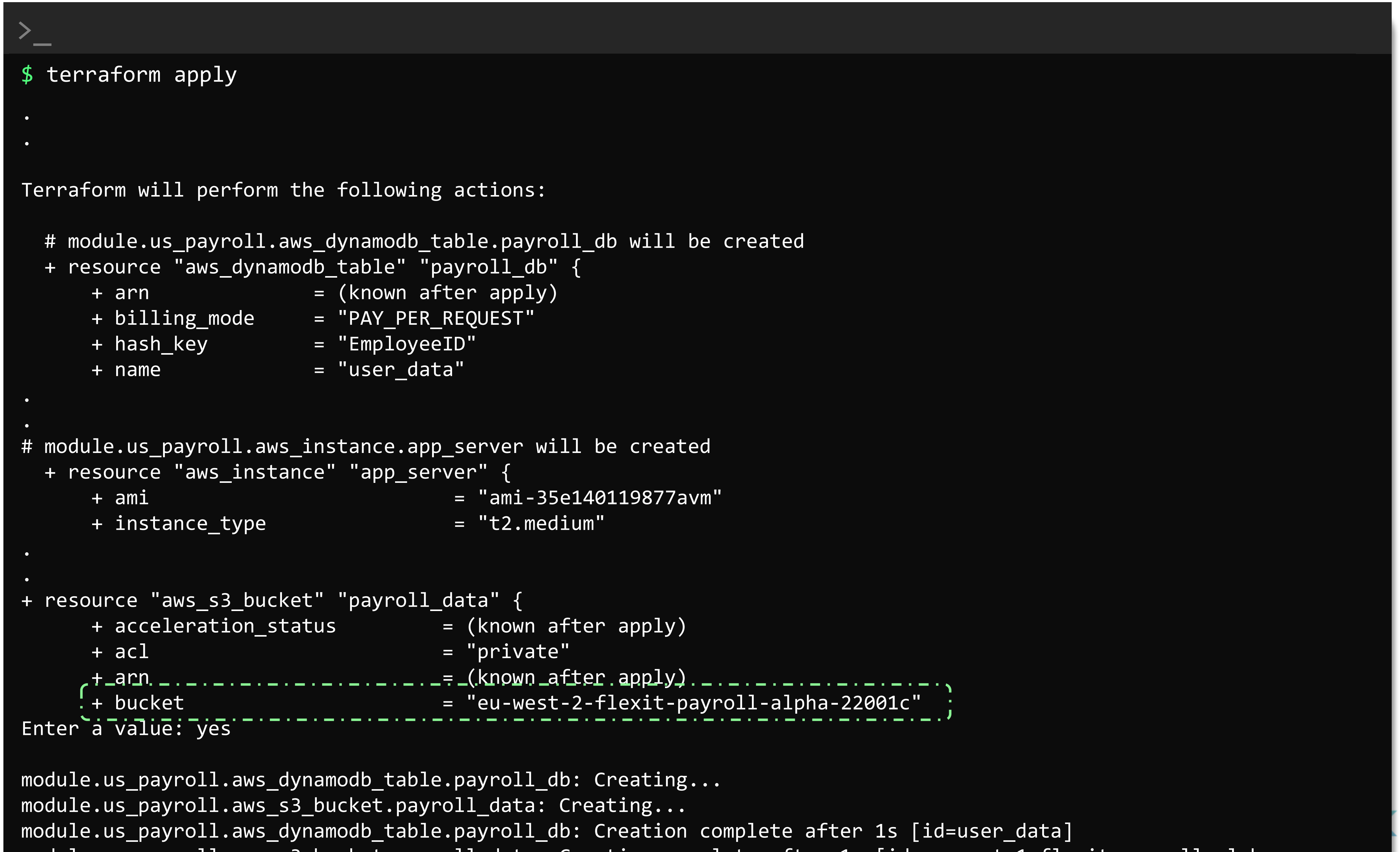
main.tf

```
module  "uk_payroll" {
   source = "../modules/payroll-app"
   app_region = "eu-west-2"
   ami         = "ami-35e140119877avm"
 }
```

provider.tf

```
provider "aws" {
  region    = "eu-west-2"
}
```





```
$ terraform apply

.
.

Terraform will perform the following actions:

  # module.us_payroll.aws_dynamodb_table.payroll_db will be created
  + resource "aws_dynamodb_table" "payroll_db" {
      + arn                = (known after apply)
      + billing_mode       = "PAY_PER_REQUEST"
      + hash_key           = "EmployeeID"
      + name               = "user_data"
.
.
# module.us_payroll.aws_instance.app_server will be created
  + resource "aws_instance" "app_server" {
      + ami                            = "ami-35e140119877avm"
      + instance_type                  = "t2.medium"
.
.
+ resource "aws_s3_bucket" "payroll_data" {
      + acceleration_status           = (known after apply)
      + acl                           = "private"
      + arn                           = (known after apply)
      + bucket                        = "eu-west-2-flexit-payroll-alpha-22001c"
Enter a value: yes

module.us_payroll.aws_dynamodb_table.payroll_db: Creating...
module.us_payroll.aws_s3_bucket.payroll_data: Creating...
module.us_payroll.aws_dynamodb_table.payroll_db: Creation complete after 1s [id=user_data]
```

```
.
.

Terraform will perform the following actions:

  # module.us_payroll.aws_dynamodb_table.payroll_db will be c
  + resource "aws_dynamodb_table" "payroll_db" {
      + arn                  = (known after apply)
      + billing_mode         = "PAY_PER_REQUEST"
      + hash_key             = "EmployeeID"
      + name                 = "user_data"
.
.
# module.us_payroll.aws_instance.app_server will be created
  + resource "aws_instance" "app_server" {
      + ami                              = "ami-35e140119877avm"
```

KODEKLOUD

Functions, Operators & Conditional Expressions

,

Functions

main.tf

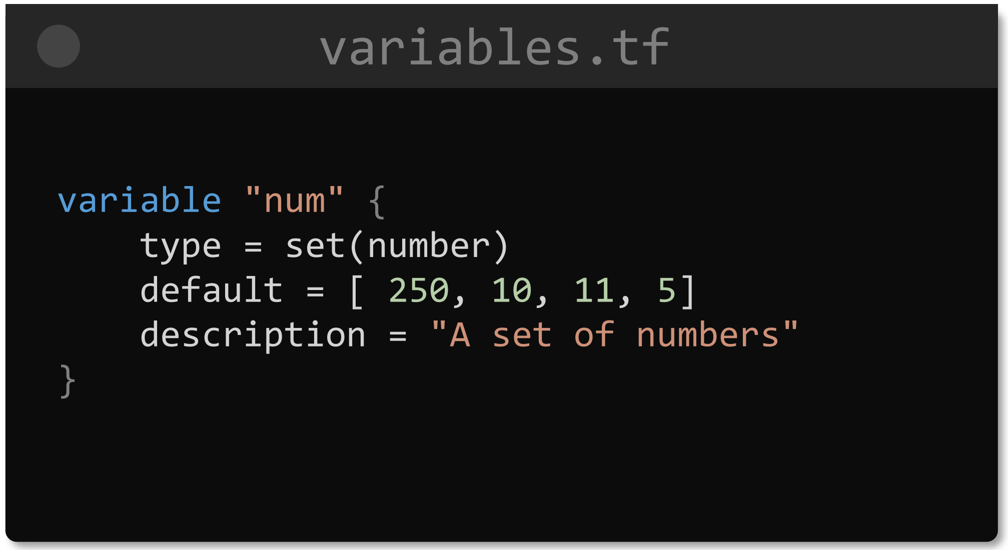
```
resource "aws_iam_policy" "adminUser" {
  name   = "AdminUsers"
  policy = file("admin-policy.json")
}

resource "local_file" "pet" {
   filename = var.filename
   count = length(var.filename)
}
```

main.tf

```
resource "local_file" "pet" {
   filename = var.filename

   for_each = toset(var.region)

}

variable region {
  type         = list
  default      = ["us-east-1",
                  "us-east-1",
                  "ca-central-1"]
  description = "A list of AWS Regions"
}
```

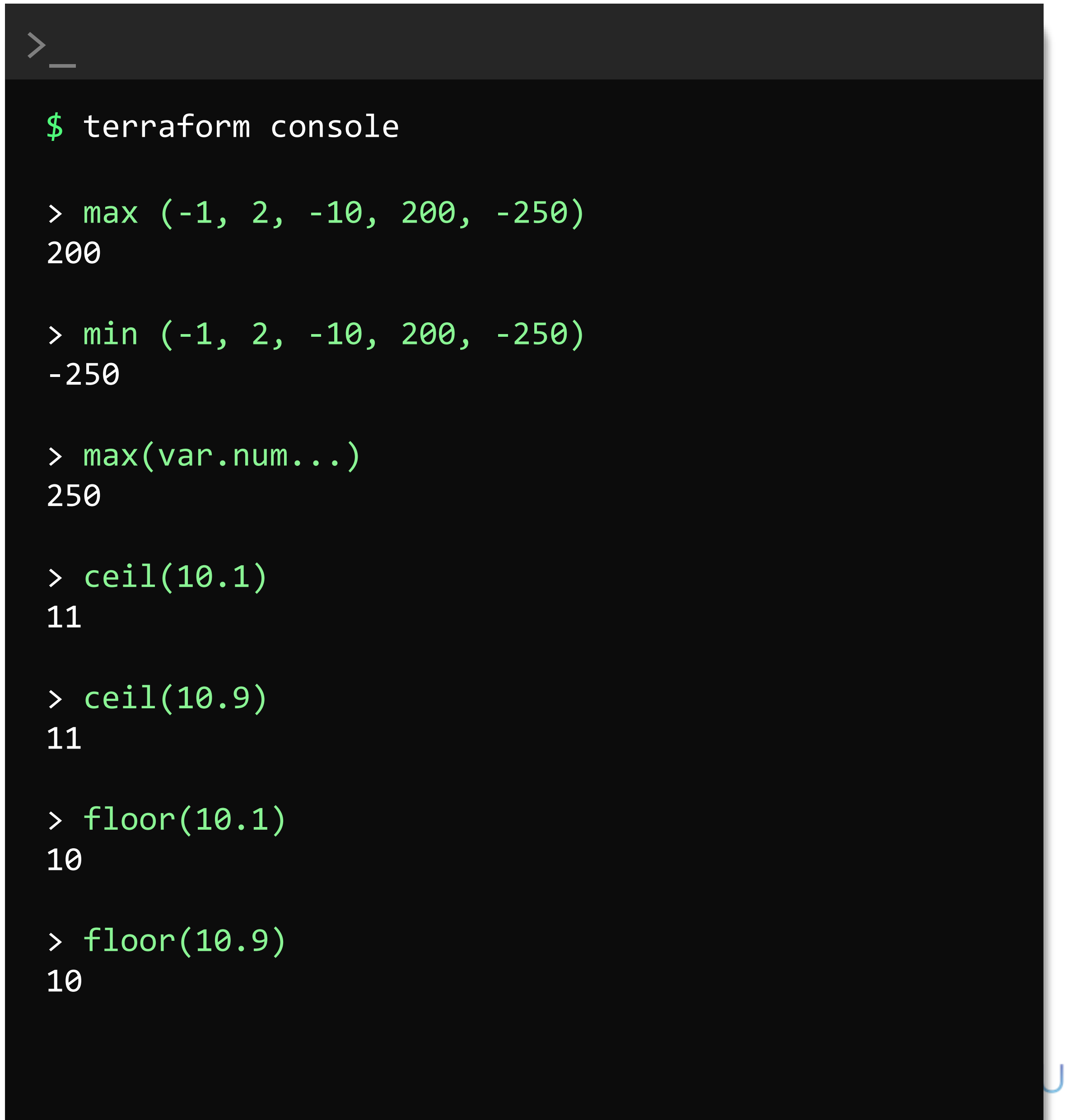
```
>_
$ terraform console
>file("/root/terraform-projects/main.tf)
 resource "aws_instance" "development" {
   ami           = "ami-0edab43b6fa892279"
   instance_type = "t2.micro"
 }
> length(var.region)
3
> toset(var.region)
[
  "ca-central-1",
  "us-east-1",

]
>
```

Numeric Functions



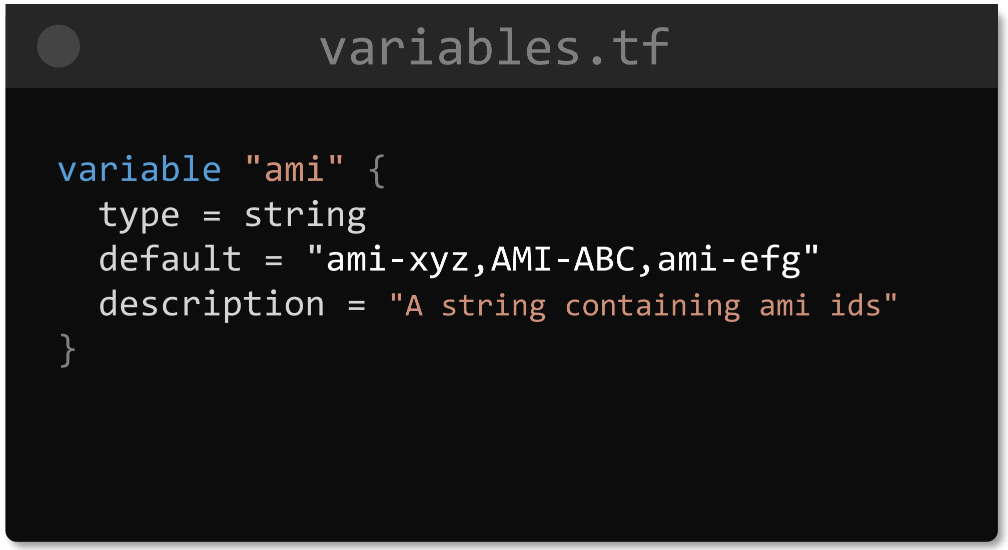
variables.tf

```
variable "num" {
    type = set(number)
    default = [ 250, 10, 11, 5]
    description = "A set of numbers"
}
```



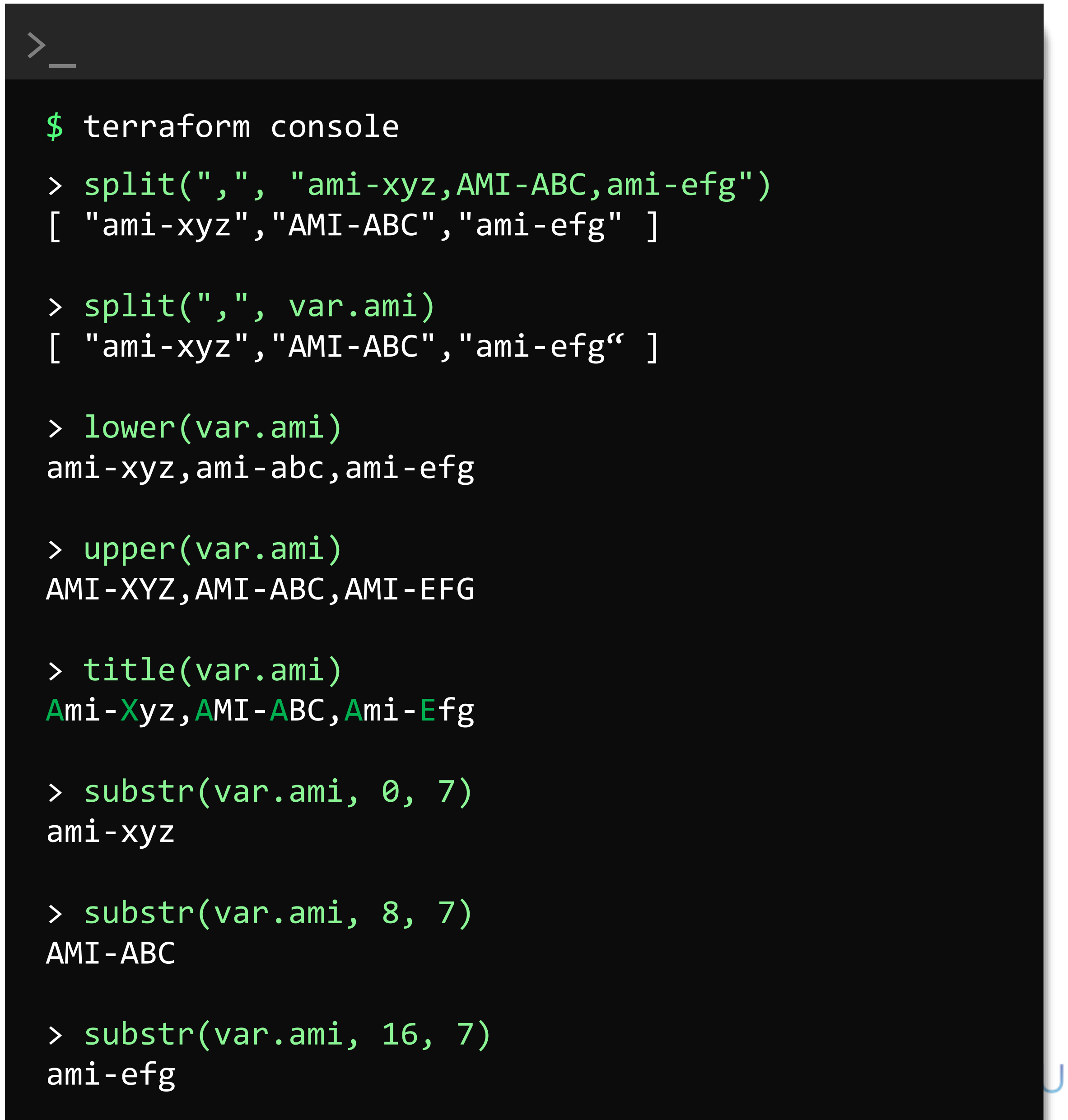
```
$ terraform console

> max (-1, 2, -10, 200, -250)
200

> min (-1, 2, -10, 200, -250)
-250

> max(var.num...)
250

> ceil(10.1)
11

> ceil(10.9)
11

> floor(10.1)
10

> floor(10.9)
10
```

String Functions



variables.tf

```
variable "ami" {
  type = string
  default = "ami-xyz,AMI-ABC,ami-efg"
  description = "A string containing ami ids"
}
```



```
$ terraform console
> split(",", "ami-xyz,AMI-ABC,ami-efg")
[ "ami-xyz","AMI-ABC","ami-efg" ]

> split(",", var.ami)
[ "ami-xyz","AMI-ABC","ami-efg“ ]

> lower(var.ami)
ami-xyz,ami-abc,ami-efg

> upper(var.ami)
AMI-XYZ,AMI-ABC,AMI-EFG

> title(var.ami)
Ami-Xyz,AMI-ABC,Ami-Efg

> substr(var.ami, 0, 7)
ami-xyz

> substr(var.ami, 8, 7)
AMI-ABC

> substr(var.ami, 16, 7)
ami-efg
```

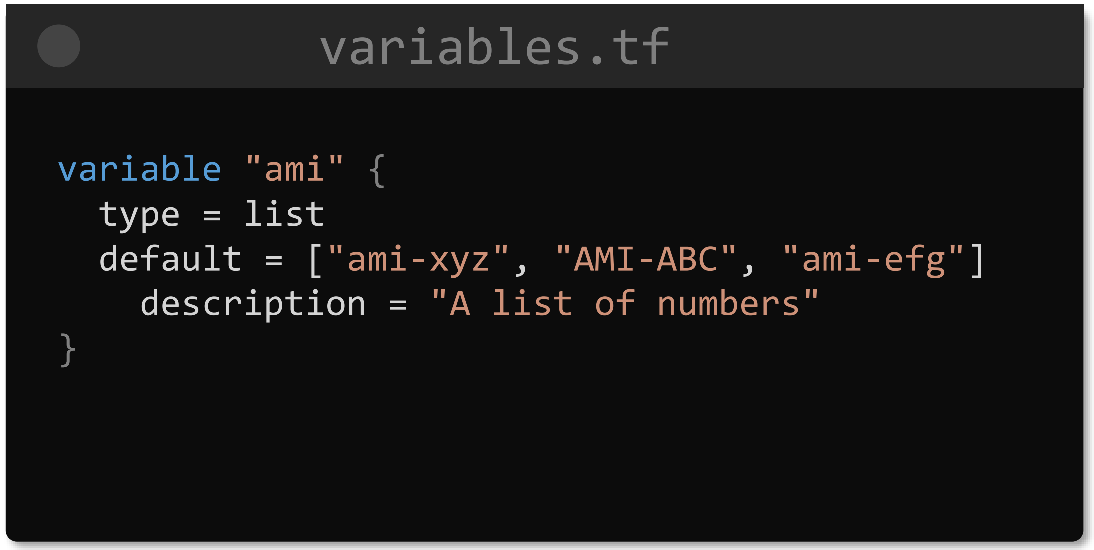
String Functions

variables.tf

```
variable "ami" {
  type = list
  default = ["ami-xyz", "AMI-ABC", "ami-efg"]
    description = "A list of numbers"
}
```

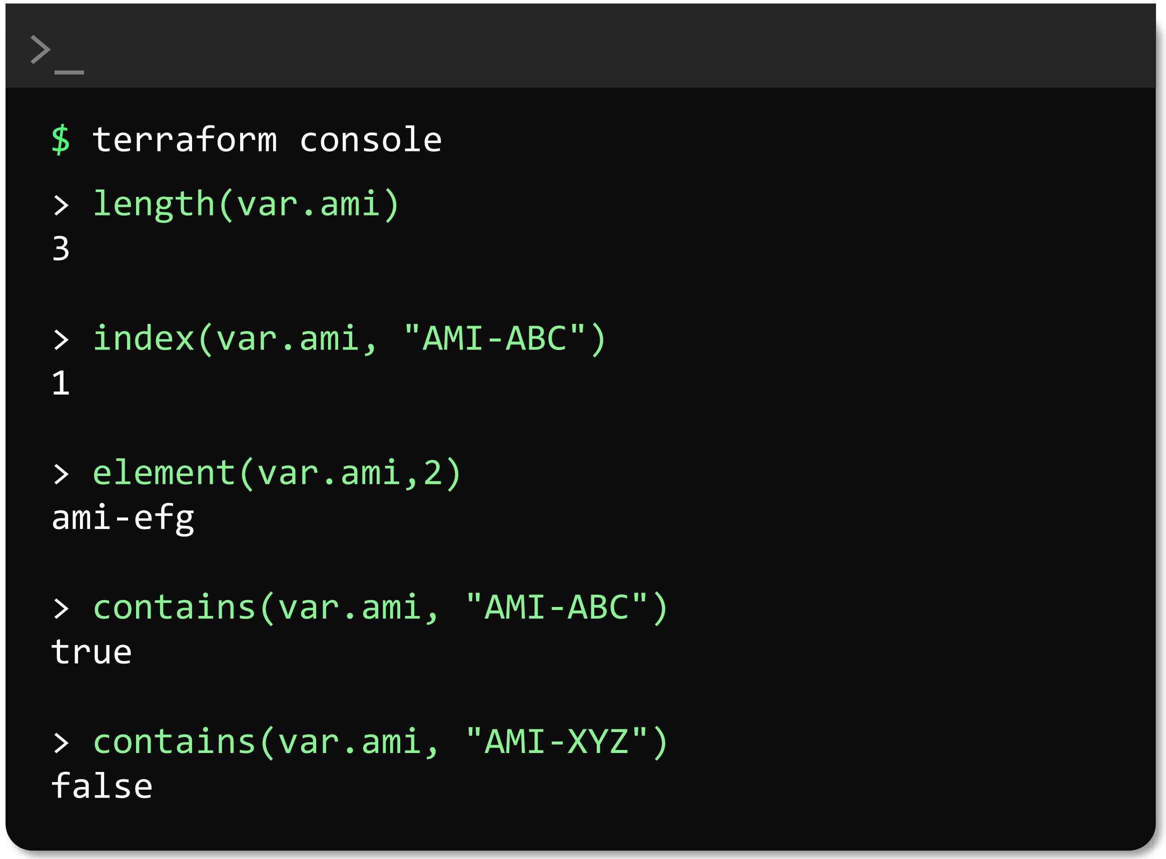
```
>_
$ terraform console
> join(",", ["ami-xyz", "AMI-ABC", "ami-efg"])
ami-xyz,AMI-ABC,ami-efg

> join(",", var.ami)
ami-xyz,AMI-ABC,ami-efg
```

Collection Functions



variables.tf

```
variable "ami" {
  type = list
  default = ["ami-xyz", "AMI-ABC", "ami-efg"]
    description = "A list of numbers"
}
```



```
>_
$ terraform console
> length(var.ami)
3

> index(var.ami, "AMI-ABC")
1

> element(var.ami,2)
ami-efg

> contains(var.ami, "AMI-ABC")
true

> contains(var.ami, "AMI-XYZ")
false
```

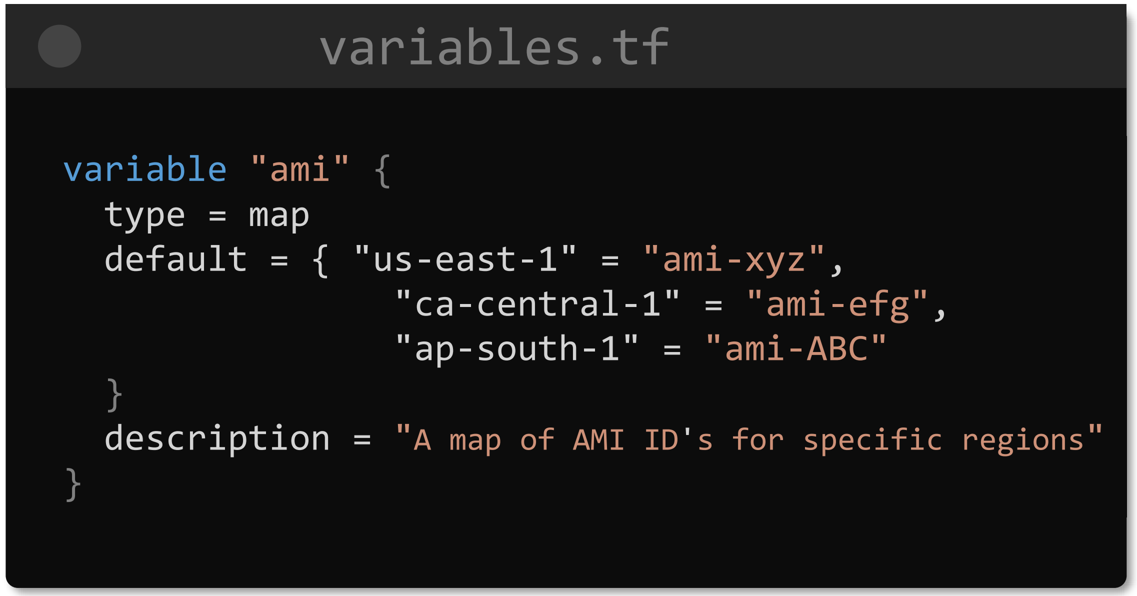
Map Functions

variables.tf

```
variable "ami" {
  type = map
  default = { "us-east-1" = "ami-xyz",
              "ca-central-1" = "ami-efg",
              "ap-south-1" = "ami-ABC"
  }
  description = "A map of AMI ID's for specific regions"
}
```

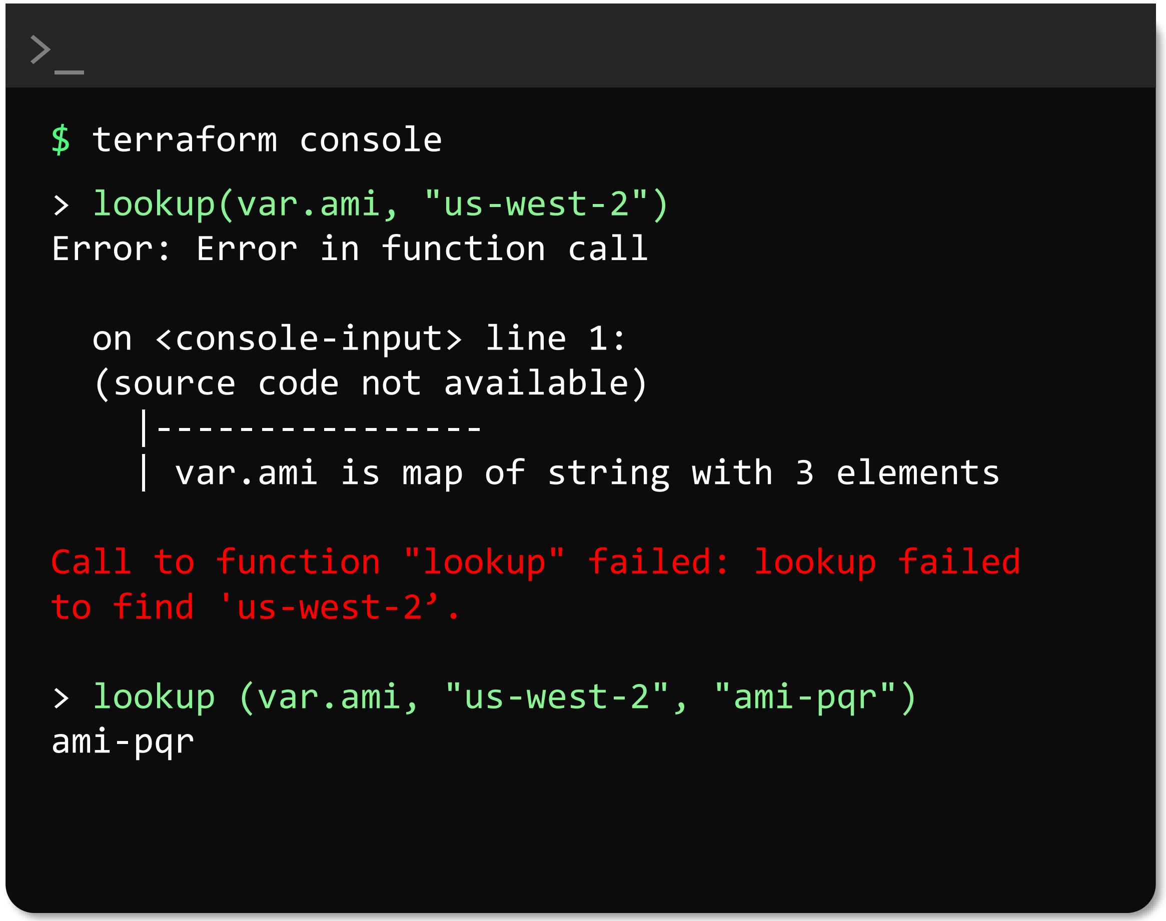
```
>_

$ terraform console
> keys(var.ami)
[
  "ap-south-1",
  "ca-central-1",
  "us-east-1",
]

> values(var.ami)
[
  "ami-ABC",
  "ami-efg",
  "ami-xyz",
]

> lookup(var.ami, "ca-central-1")
ami-efg
```

Map Functions



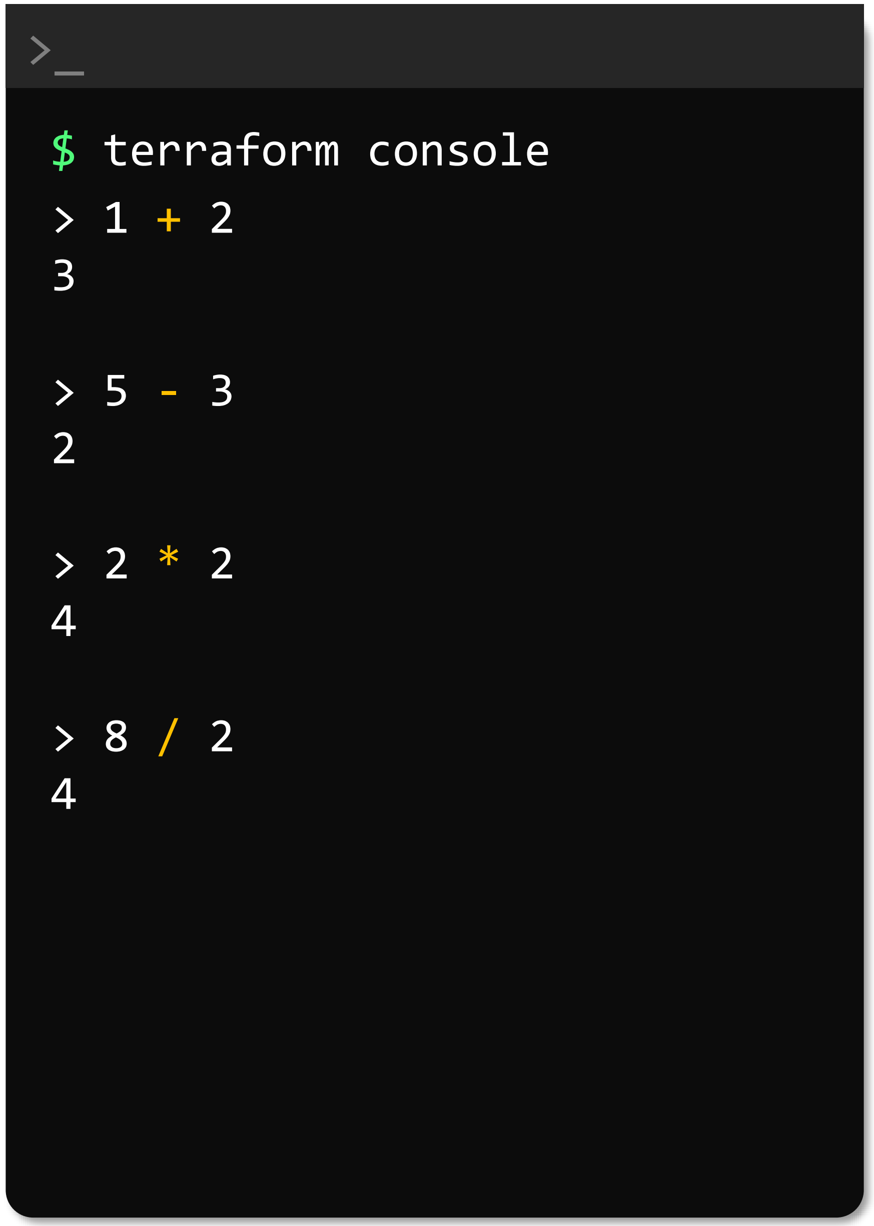
variables.tf

```
variable "ami" {
  type = map
  default = { "us-east-1" = "ami-xyz",
              "ca-central-1" = "ami-efg",
              "ap-south-1" = "ami-ABC"
  }
  description = "A map of AMI ID's for specific regions"
}
```



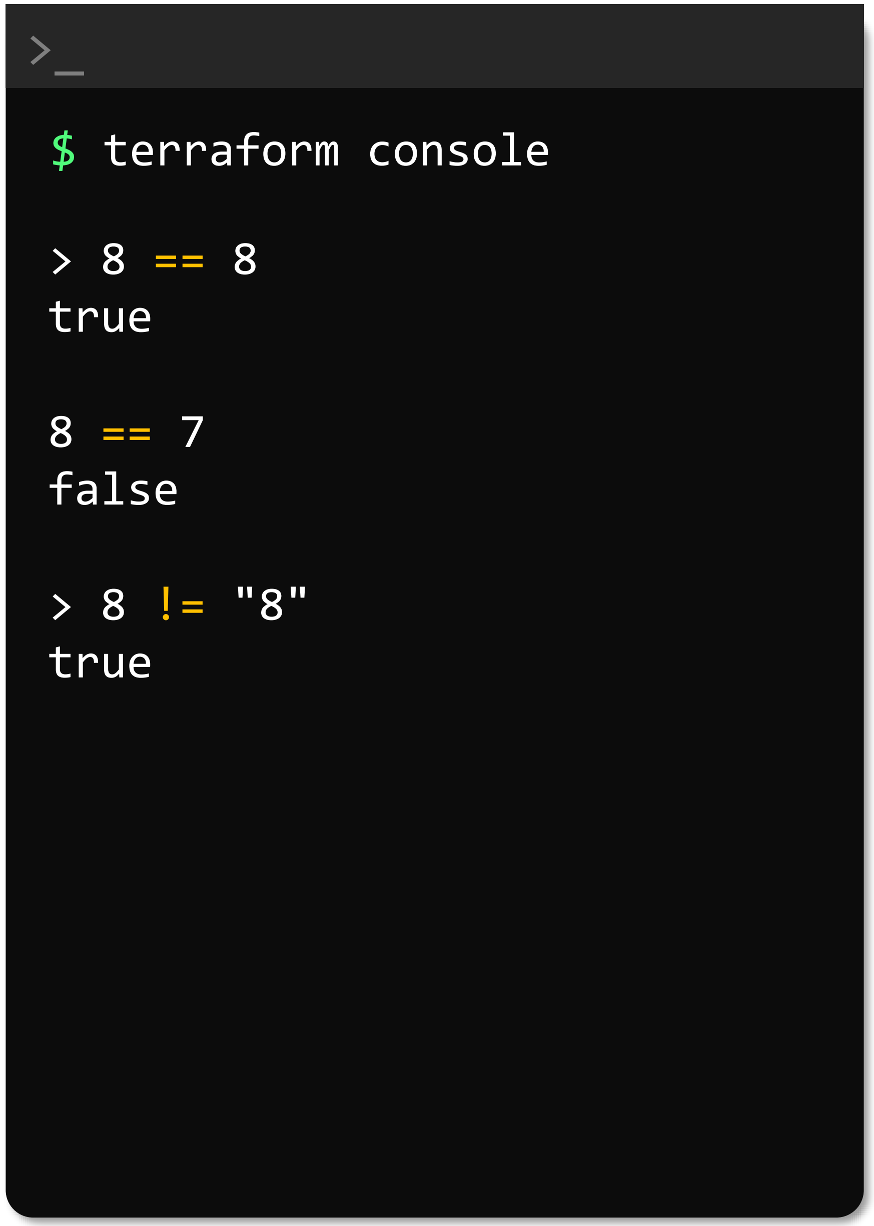
```
$ terraform console
> lookup(var.ami, "us-west-2")
Error: Error in function call

  on <console-input> line 1:
  (source code not available)
    |----------------
    | var.ami is map of string with 3 elements

Call to function "lookup" failed: lookup failed
to find 'us-west-2’.

> lookup (var.ami, "us-west-2", "ami-pqr")
ami-pqr
```

Numeric Operators



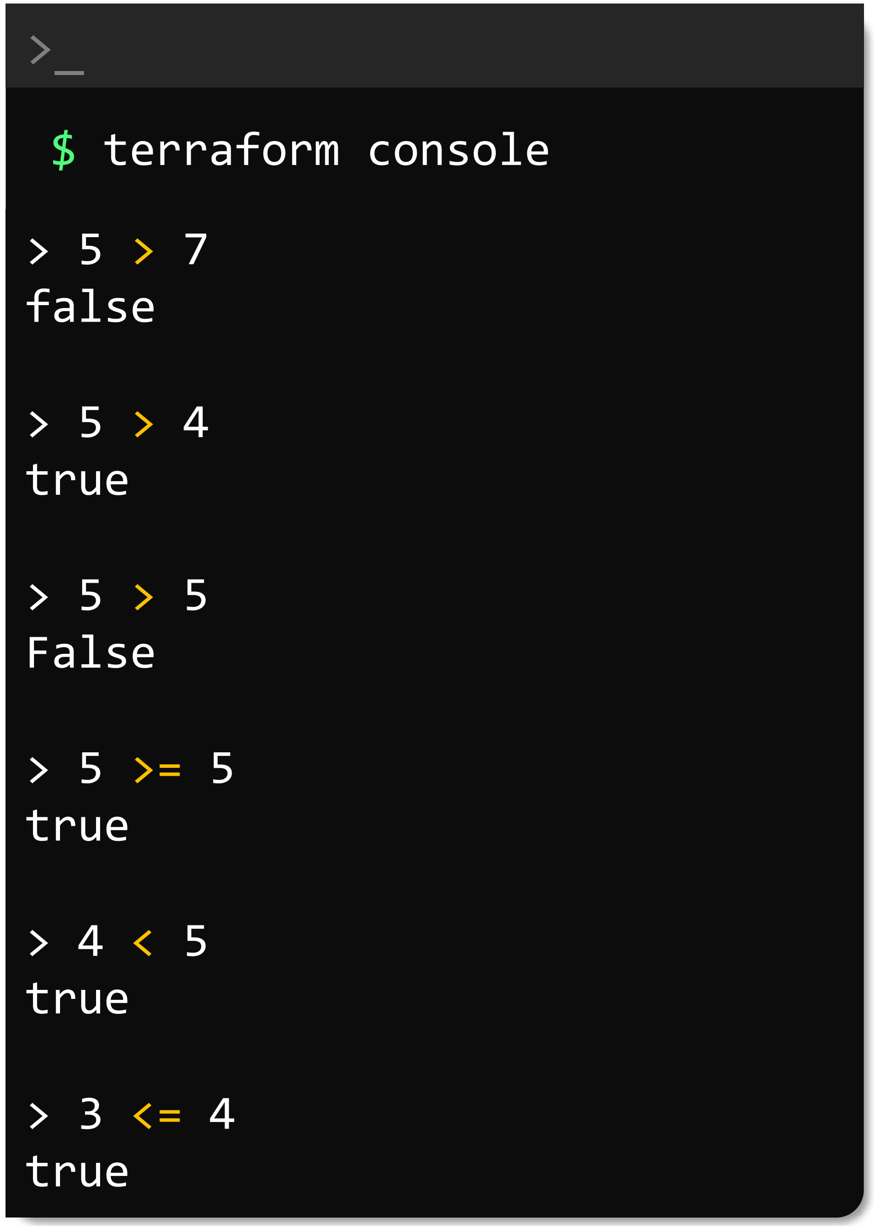
```
$ terraform console
> 1 + 2
3

> 5 - 3
2

> 2 * 2
4

> 8 / 2
4
```

Equality Operators



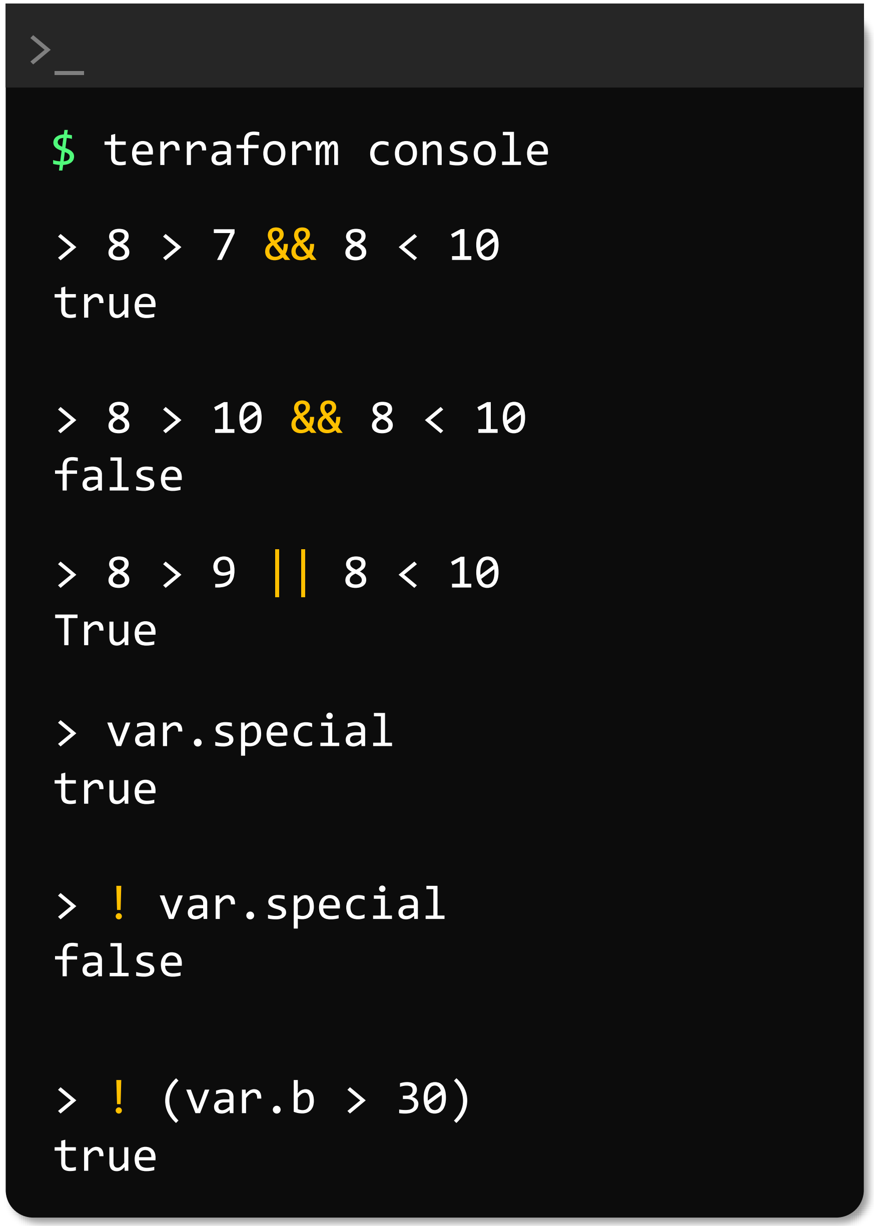
```
$ terraform console

> 8 == 8
true

8 == 7
false

> 8 != "8"
true
```

Comparison Operators

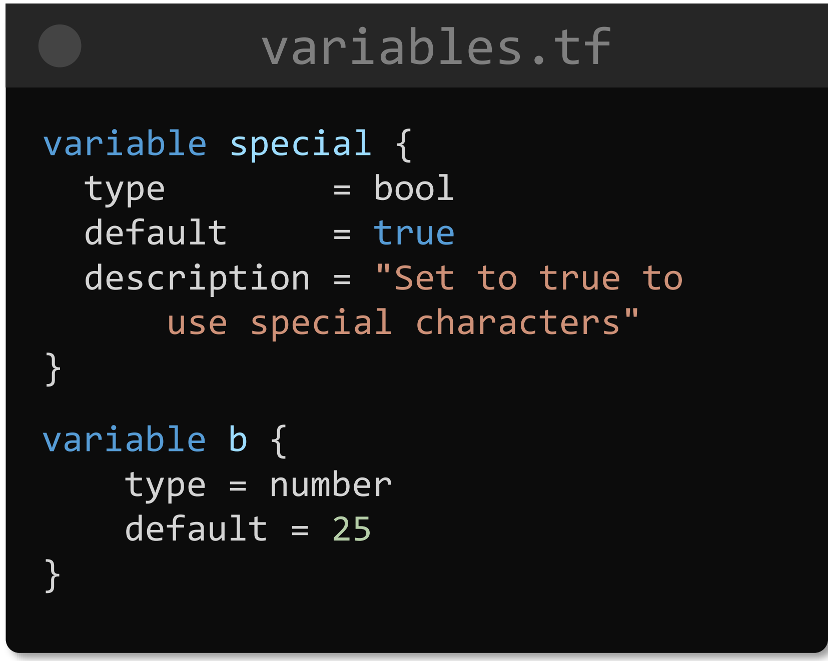


```
$ terraform console

> 5 > 7
false

> 5 > 4
true

> 5 > 5
False

> 5 >= 5
true

> 4 < 5
true

> 3 <= 4
true
```

Logical Operators



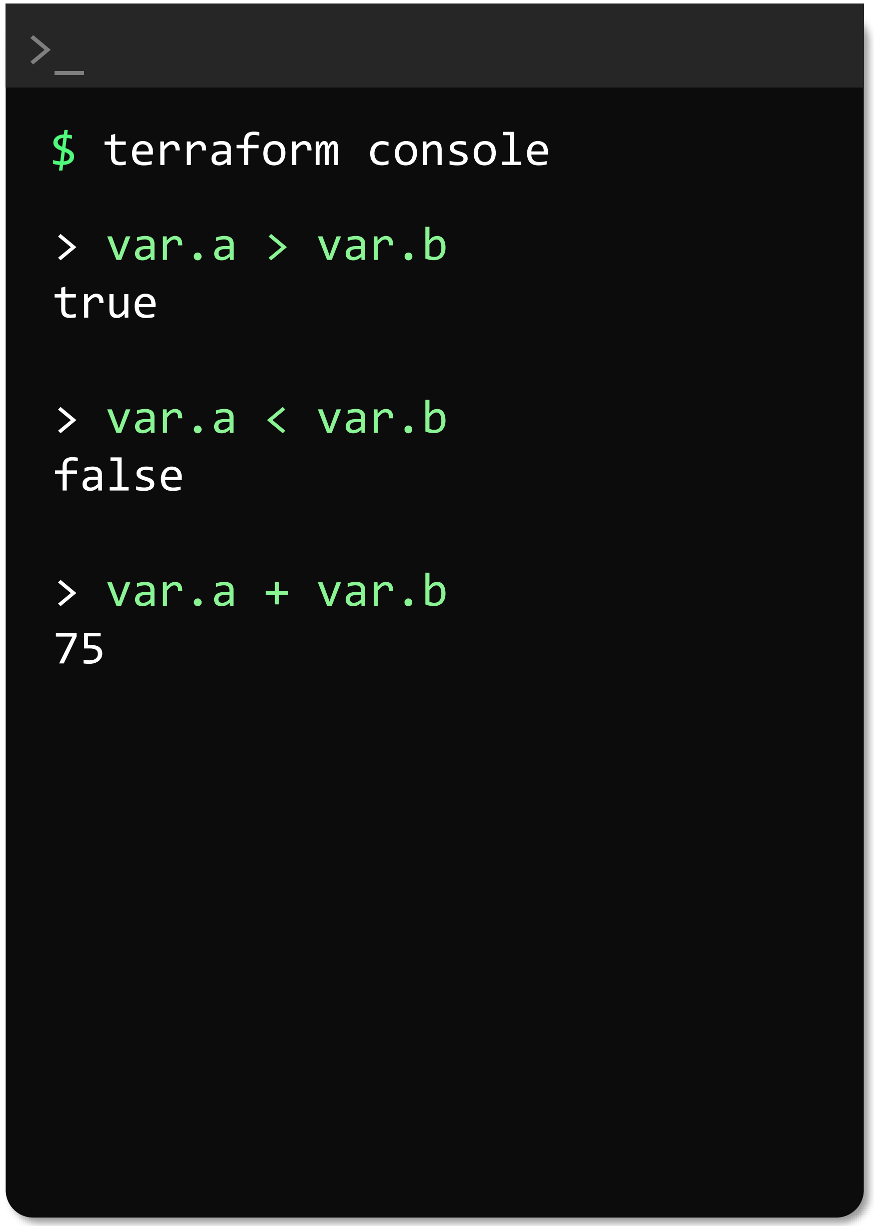
```
>_
$ terraform console

> 8 > 7 && 8 < 10
true

> 8 > 10 && 8 < 10
false

> 8 > 9 || 8 < 10
True

> var.special
true

> ! var.special
false


> ! (var.b > 30)
true
```



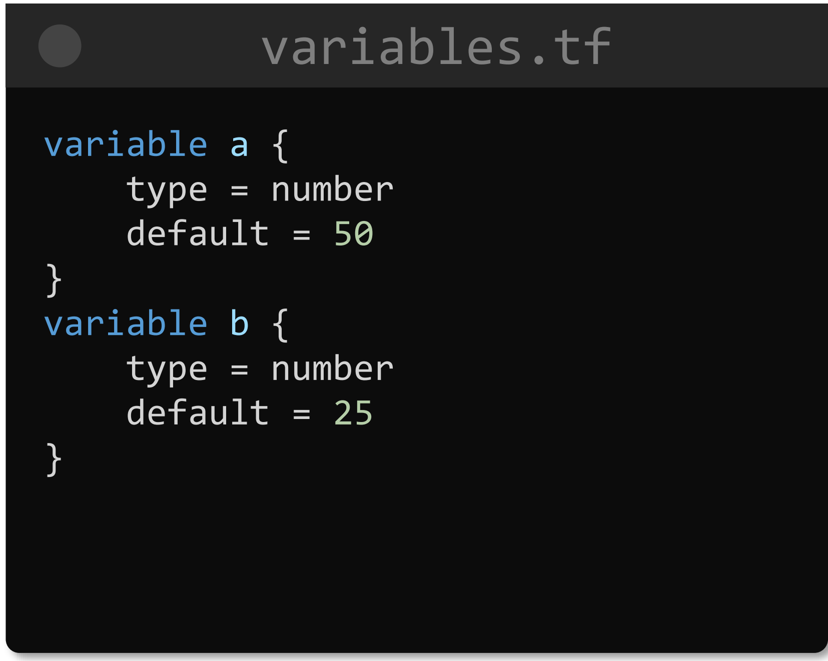
variables.tf

```
variable special {
  type        = bool
  default     = true
  description = "Set to true to
      use special characters"
}

variable b {
    type = number
    default = 25
}
```

Logical Operators



```
>_
$ terraform console

> var.a > var.b
true

> var.a < var.b
false

> var.a + var.b
75
```



```
variables.tf

variable a {
    type = number
    default = 50
}
variable b {
    type = number
    default = 25
}
```

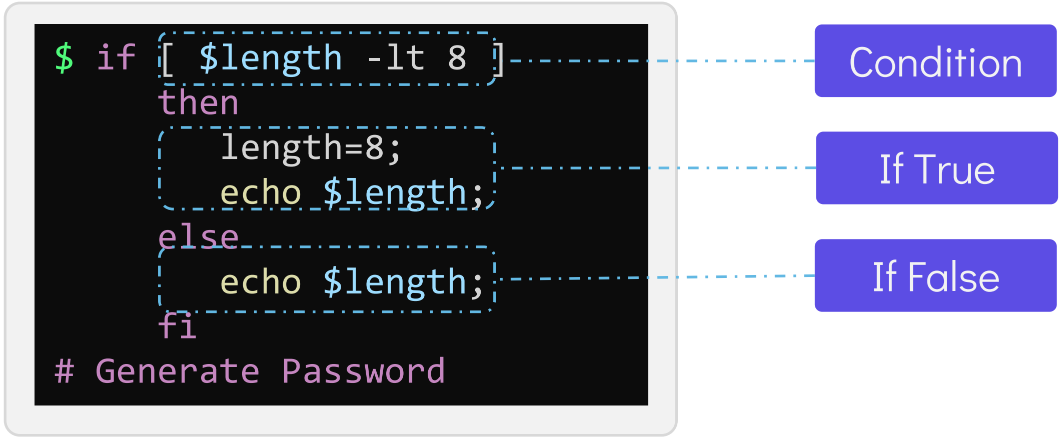
main.tf

```
resource "random_password" "password-generator" {
    length  = var.length
}

output password {
  value  = random_password.password-generator.result
}
```

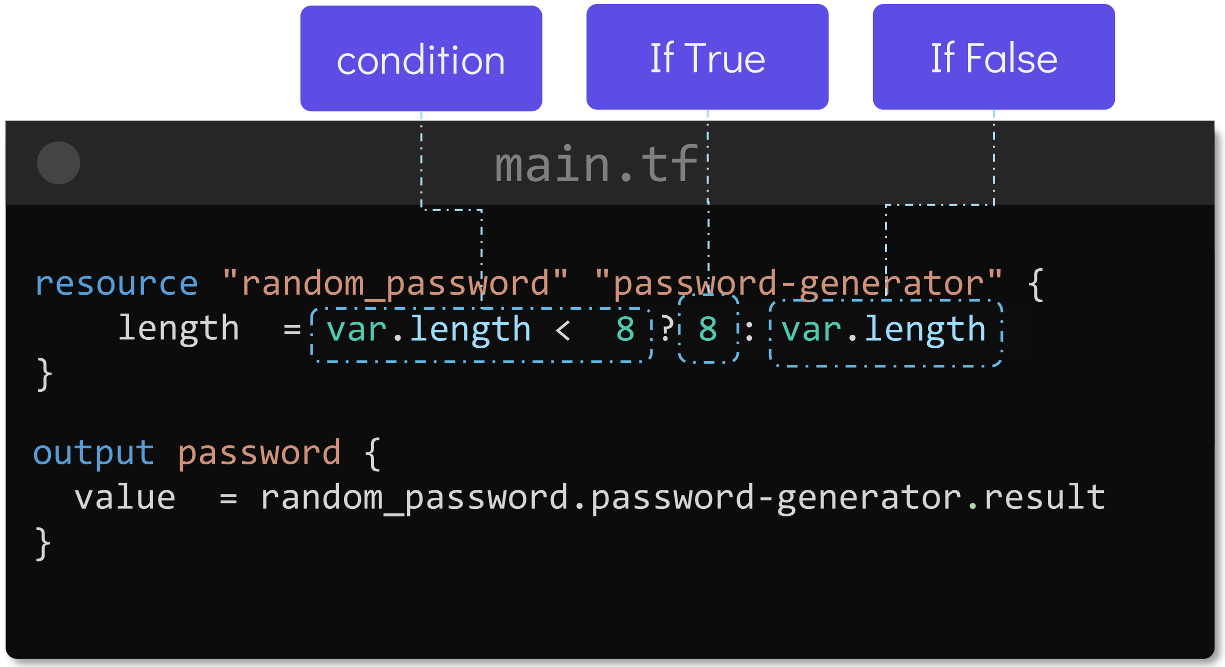
variables.tf

```
variable length {
  type        = number
  description = "The length of the password"
}
```

```
>_
$ terraform apply -var=length=5 -auto-approve
random_password.password-generator: Creating...
random_password.password-generator: Creation
complete after 0s [id=none]

Apply complete! Resources: 1 added, 0 changed, 0
destroyed.

Outputs:

password = sjsrW]
```



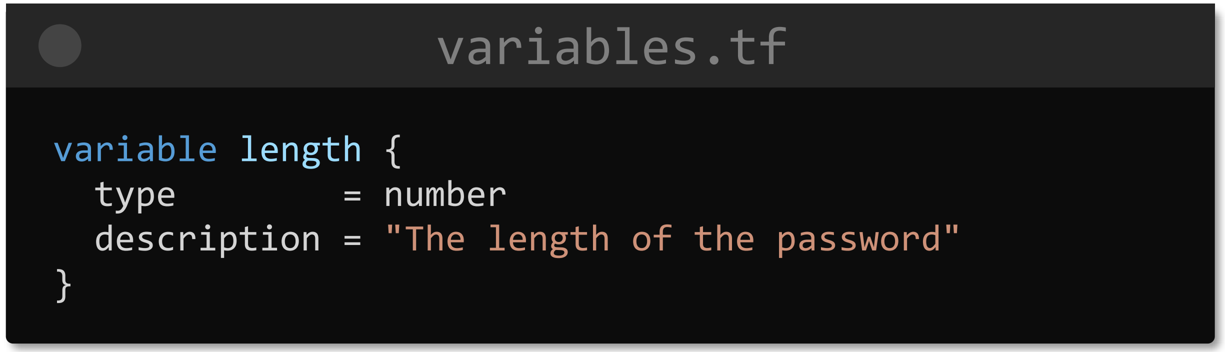
```
$ if [ $length -lt 8 ]
     then
        length=8;
        echo $length;
     else
        echo $length;
     fi
# Generate Password
```



main.tf

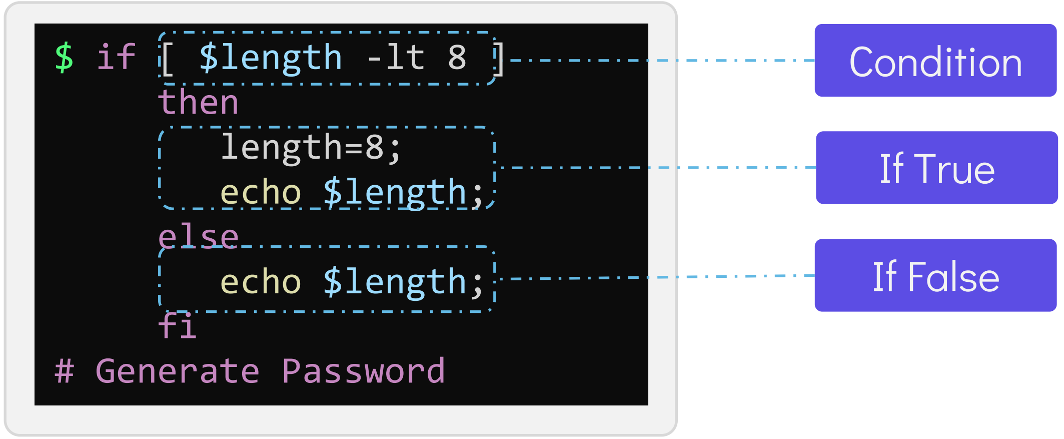
```
resource "random_password" "password-generator" {
    length   = var.length <  8 ? 8 : var.length
}

output password {
  value  = random_password.password-generator.result
}
```

```
condition ? true_val : false_val
```

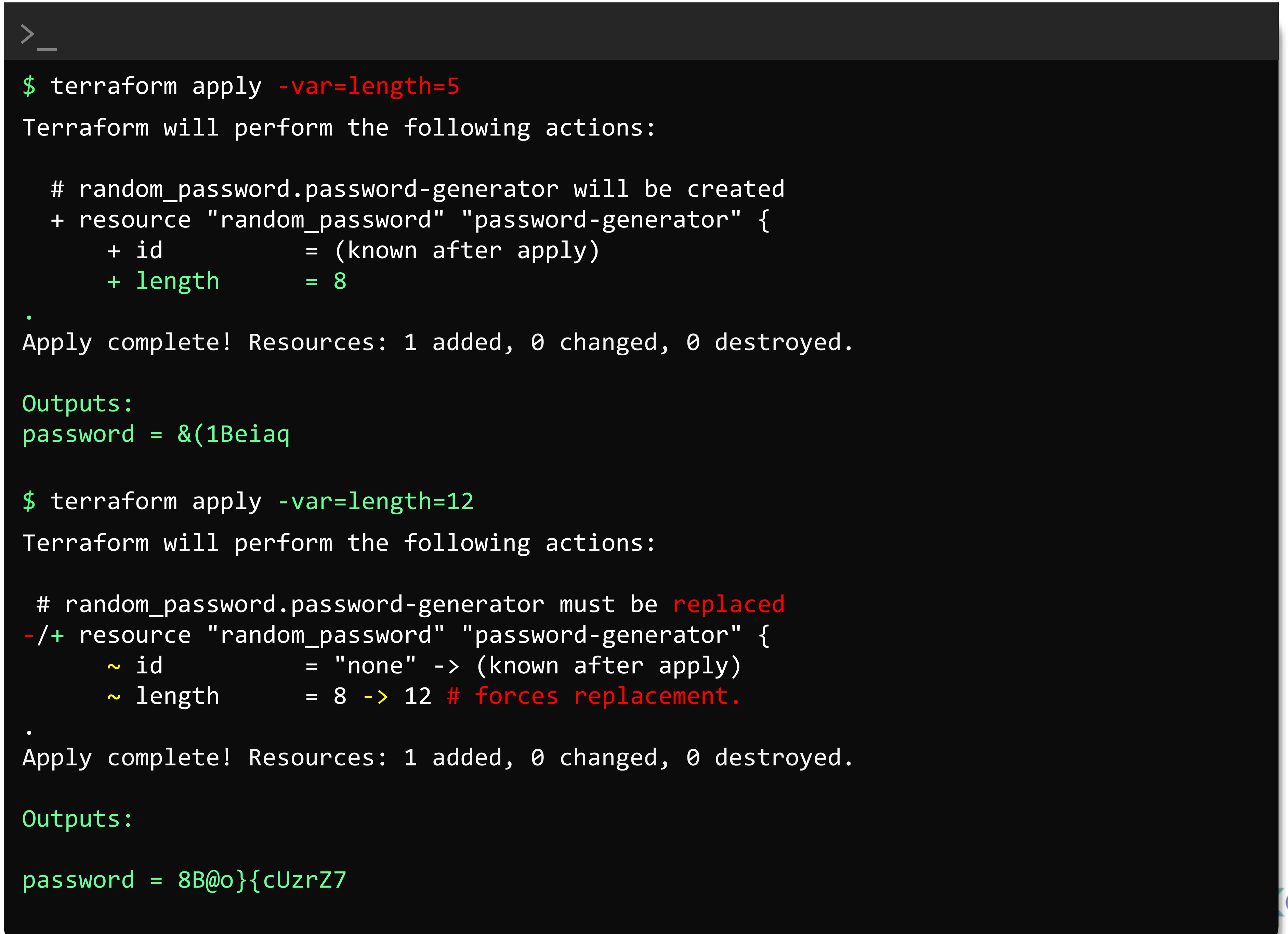


variables.tf

```
variable length {
  type        = number
  description = "The length of the password"
}
```



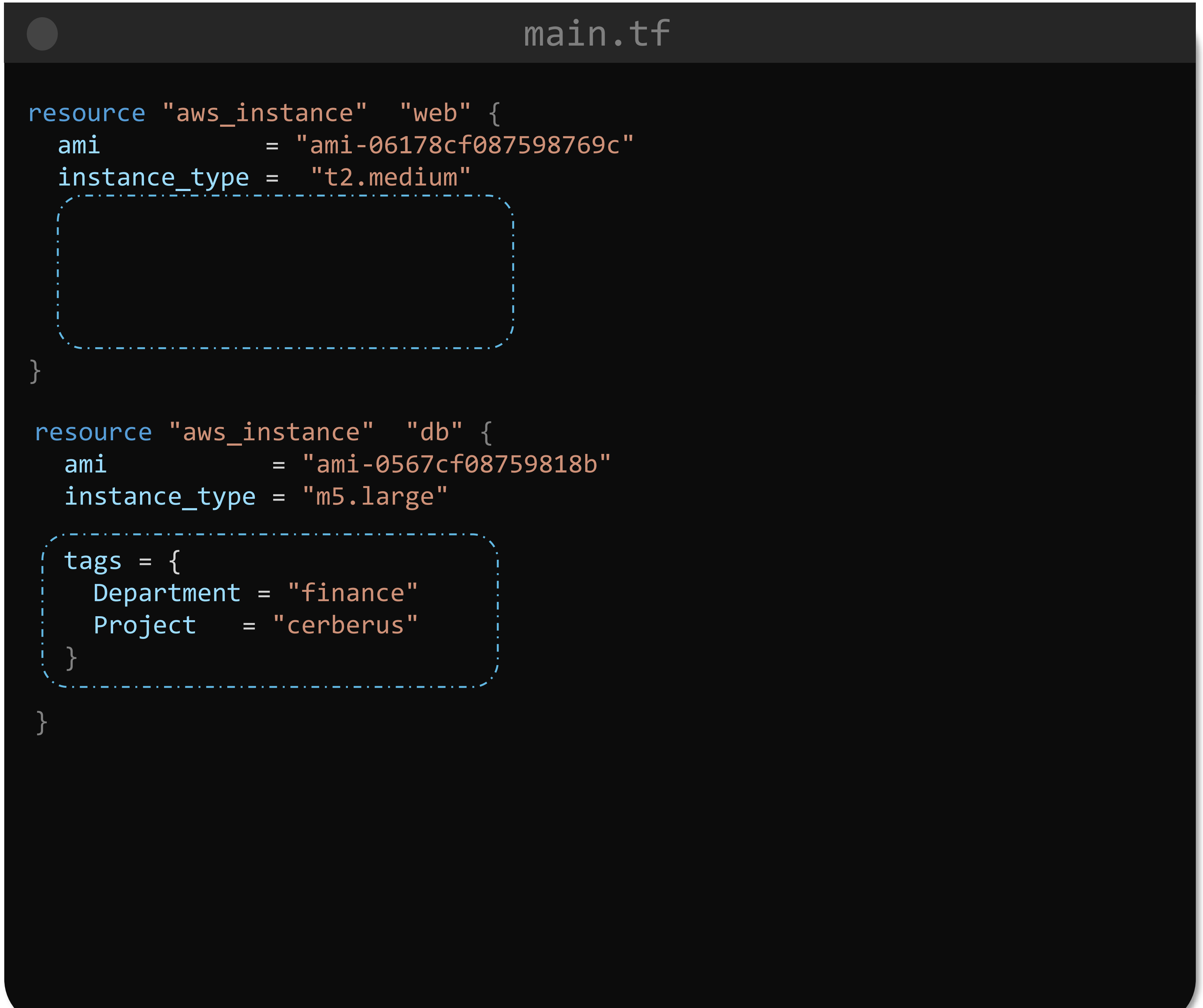
```
$ if [ $length -lt 8 ]
     then
        length=8;
        echo $length;
     else
        echo $length;
     fi
# Generate Password
```



```
$ terraform apply -var=length=5
Terraform will perform the following actions:

  # random_password.password-generator will be created
  + resource "random_password" "password-generator" {
      + id            = (known after apply)
      + length        = 8
.
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:
password = &(1Beiaq

$ terraform apply -var=length=12
Terraform will perform the following actions:

 # random_password.password-generator must be replaced
-/+ resource "random_password" "password-generator" {
      ~ id            = "none" -> (known after apply)
      ~ length        = 8 -> 12 # forces replacement.
.
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

password = 8B@o}{cUzrZ7
```

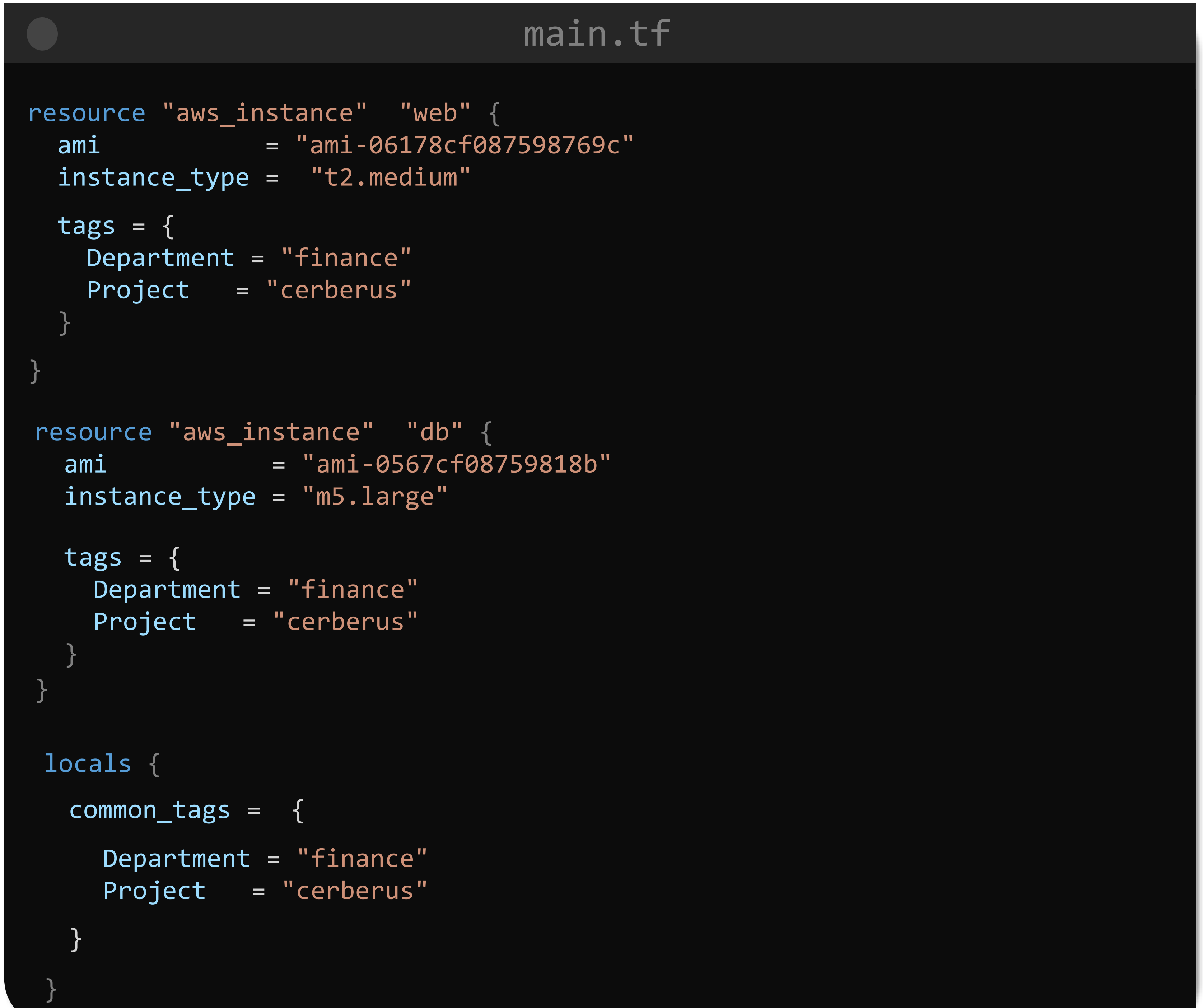
KODEKLOUD

Local Values



main.tf

```
resource "aws_instance"  "web" {
  ami             = "ami-06178cf087598769c"
  instance_type =  "t2.medium"




}

resource "aws_instance"  "db" {
  ami             = "ami-0567cf08759818b"
  instance_type = "m5.large"

  tags = {
    Department = "finance"
    Project    = "cerberus"
  }

}
```


main.tf

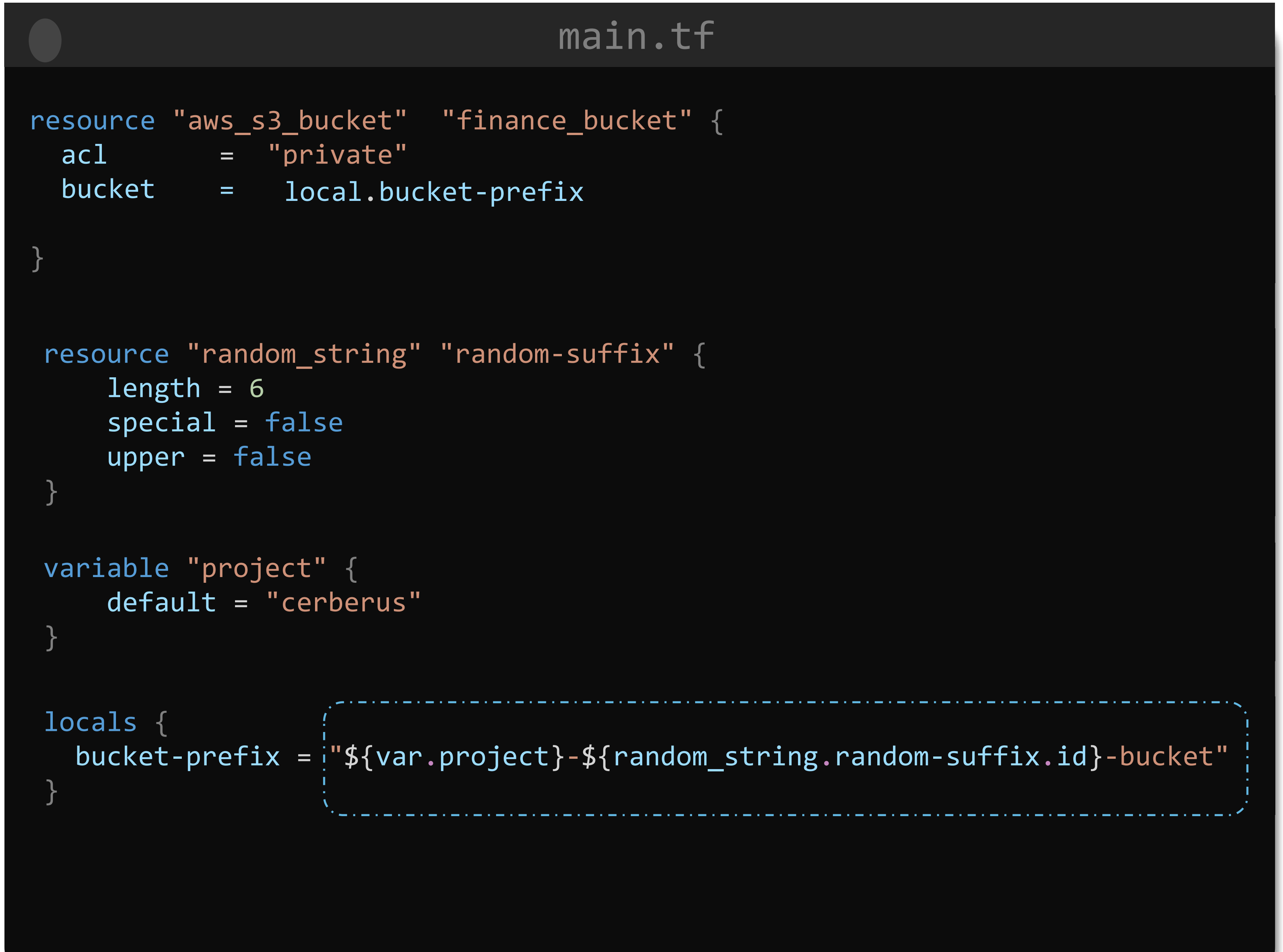
```
resource "aws_instance"  "web" {
  ami           = "ami-06178cf087598769c"
  instance_type =  "t2.medium"

  tags = {
    Department = "finance"
    Project    = "cerberus"
  }

}

resource "aws_instance"  "db" {
  ami           = "ami-0567cf08759818b"
  instance_type = "m5.large"

  tags = {
    Department = "finance"
    Project    = "cerberus"
  }
}


locals {

  common_tags =  {

    Department = "finance"
    Project    = "cerberus"

  }
}
```

main.tf

```
resource "aws_instance"  "web" {
  ami           = "ami-06178cf087598769c"
  instance_type =  "t2.medium"
  tags = local.common_tags




}

resource "aws_instance"  "db" {
  ami           = "ami-0567cf08759818b"
  instance_type = "m5.large"
  tags = local.common_tags




}

locals {
  common_tags =  {
    Department = "finance"
    Project    = "cerberus"
  }
}
```

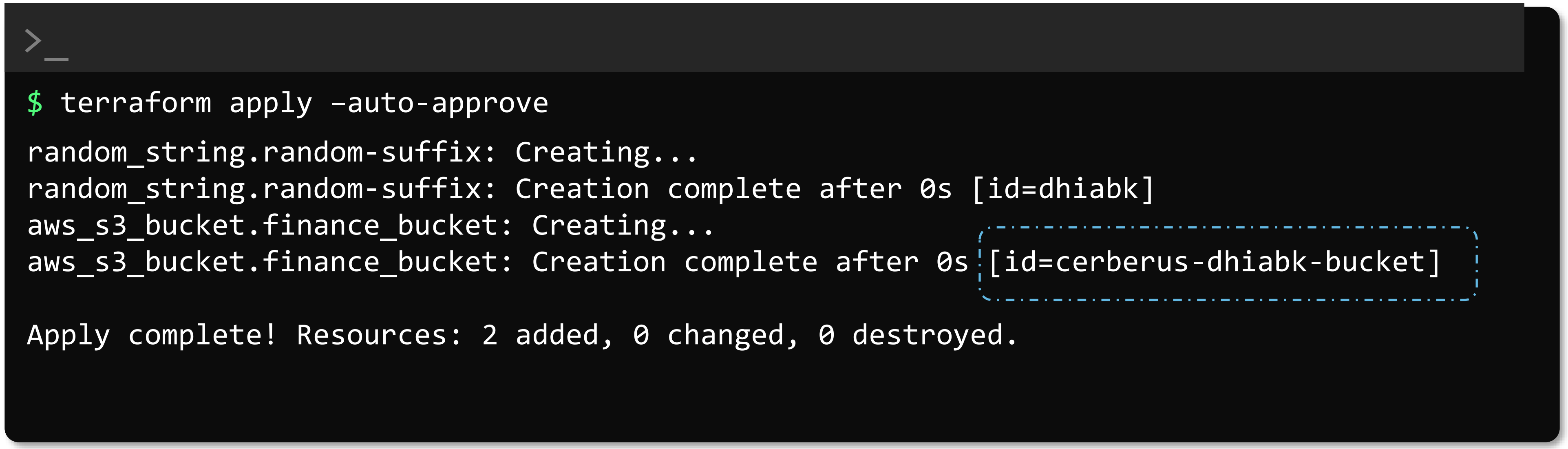
```
>_
$ terraform apply
iac-server $ terraform apply

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

Terraform will perform the following actions:

  # aws_instance.db will be created
  + resource "aws_instance" "db" {
      .
      .
      + tags                                    = {
          + "Department" = "finance"
          + "Project"    = "cerberus"
        }
.
.

  # aws_instance.web will be created
  + resource "aws_instance" "web" {
.
.
      + tags                                    = {
          + "Department" = "finance"
          + "Project"    = "cerberus"
        }
```

bucket = <random_string>-<project_name>-bucket



main.tf

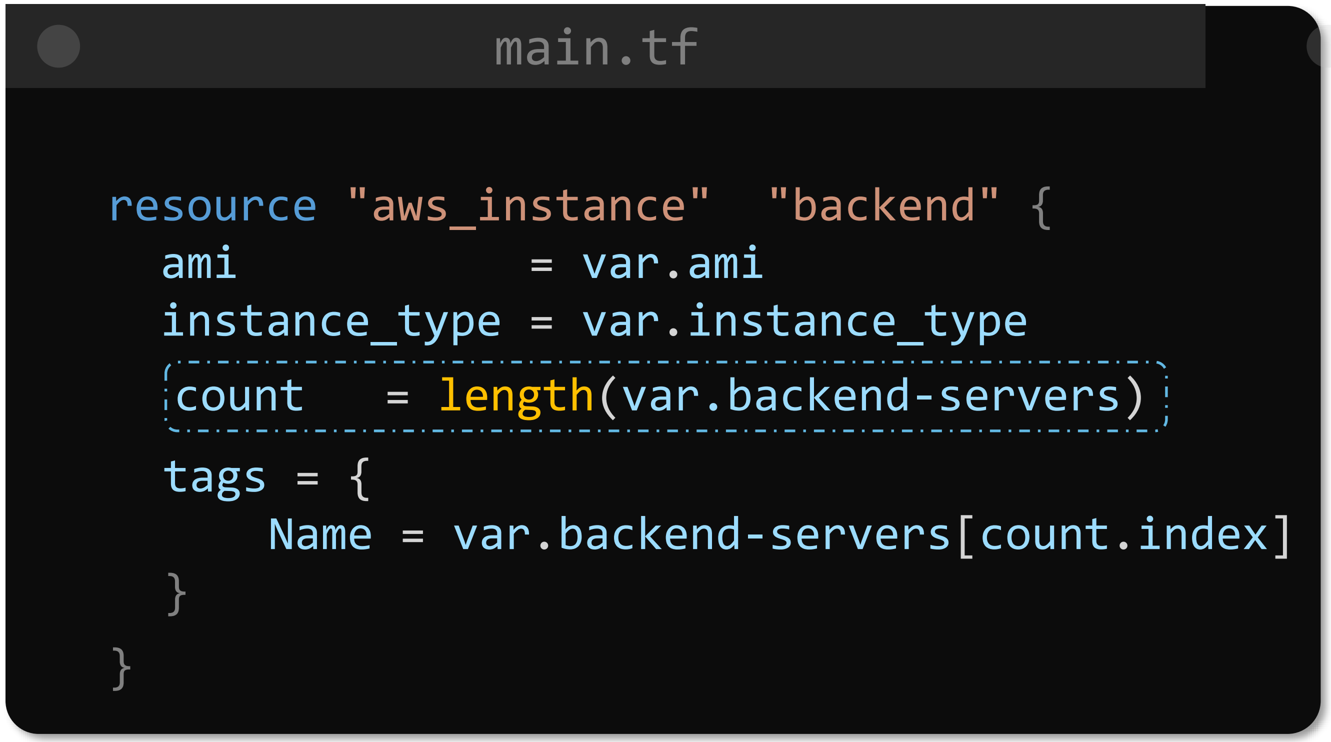
```
resource "aws_s3_bucket"  "finance_bucket" {
  acl        =  "private"
  bucket     =   local.bucket-prefix

}


 resource "random_string" "random-suffix" {
     length = 6
     special = false
     upper = false
 }

 variable "project" {
     default = "cerberus"
 }

 locals {
   bucket-prefix = "${var.project}-${random_string.random-suffix.id}-bucket"
 }
```



```
$ terraform apply -auto-approve
random_string.random-suffix: Creating...
random_string.random-suffix: Creation complete after 0s [id=dhiabk]
aws_s3_bucket.finance_bucket: Creating...
aws_s3_bucket.finance_bucket: Creation complete after 0s [id=cerberus-dhiabk-bucket]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```

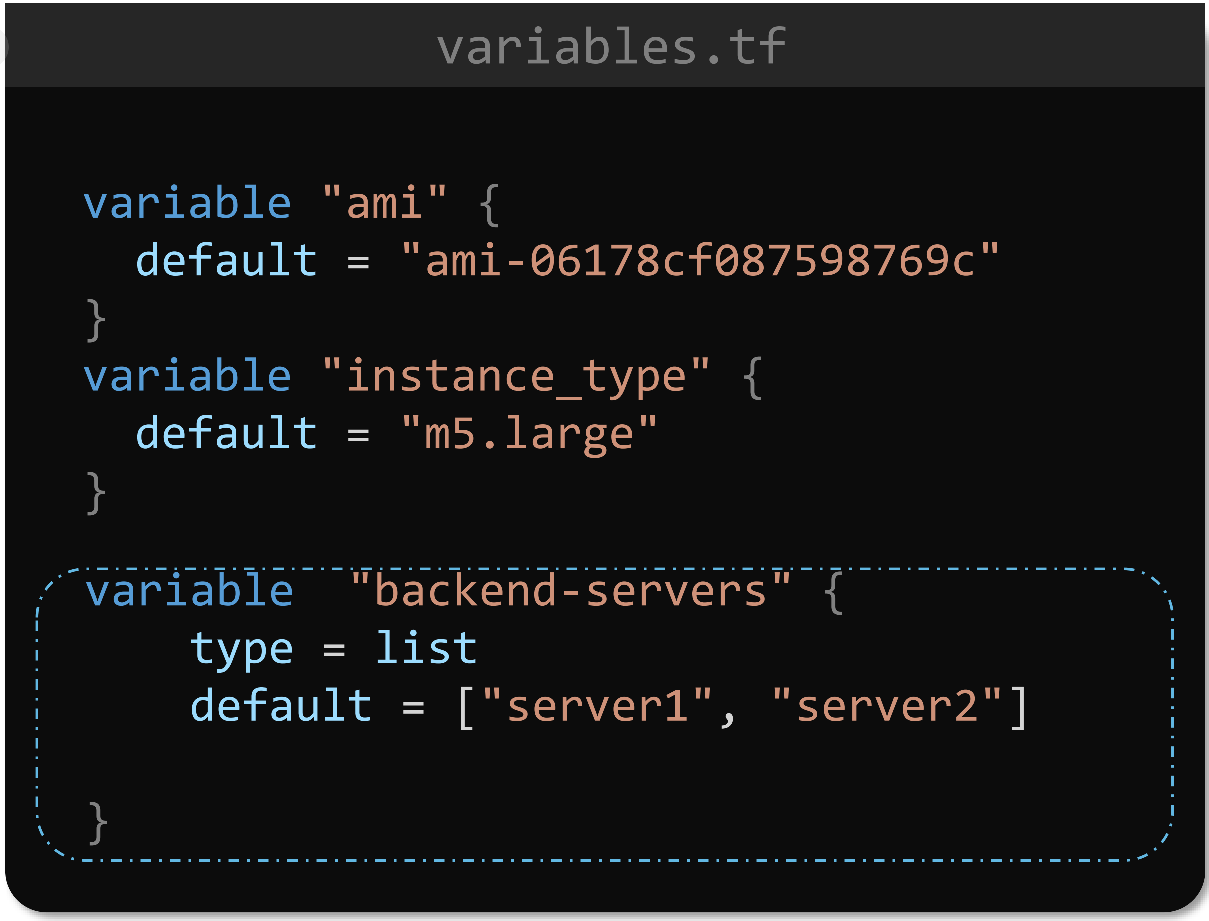
KODEKLOUD

Dynamic Blocks and Splat Expressions



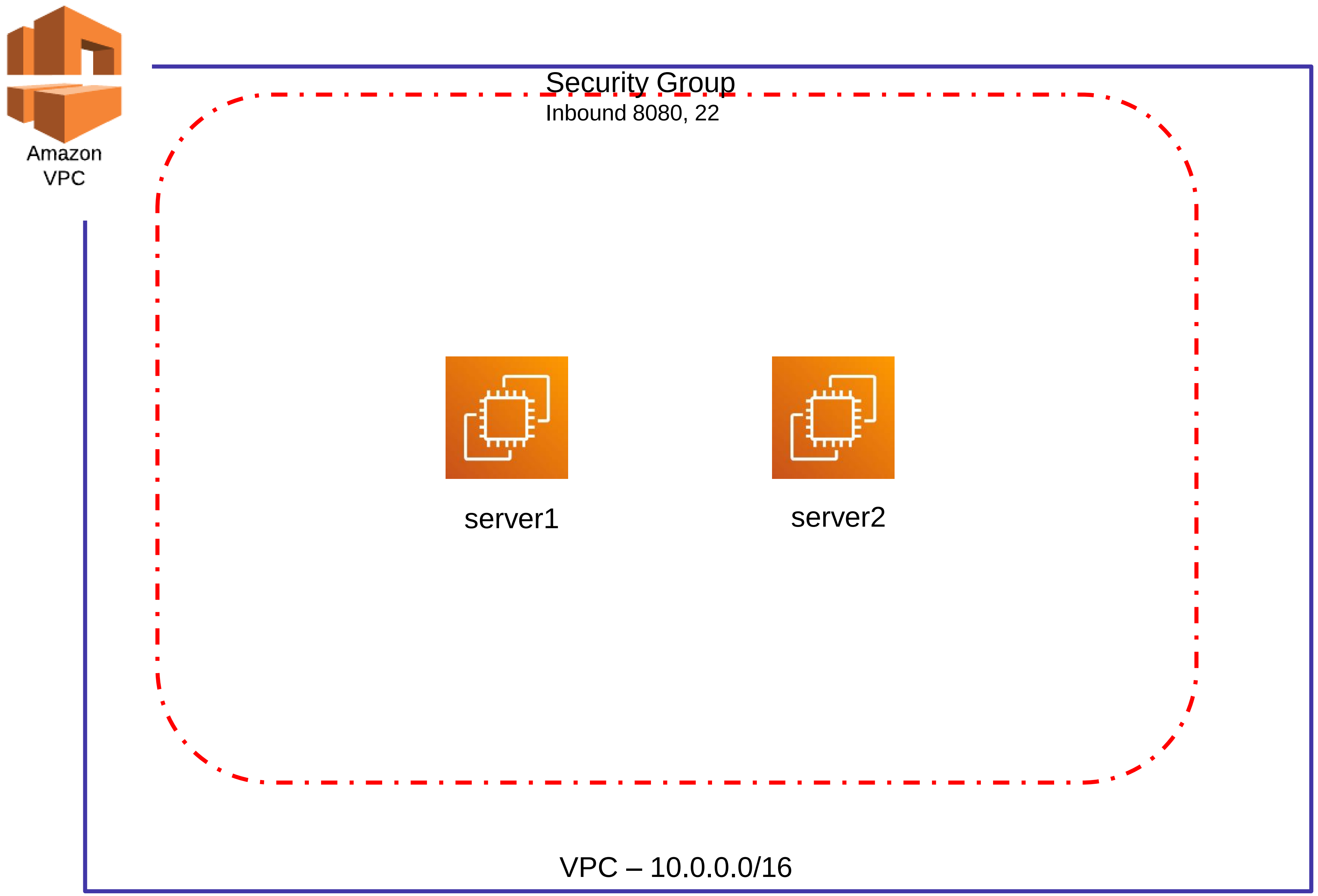
main.tf

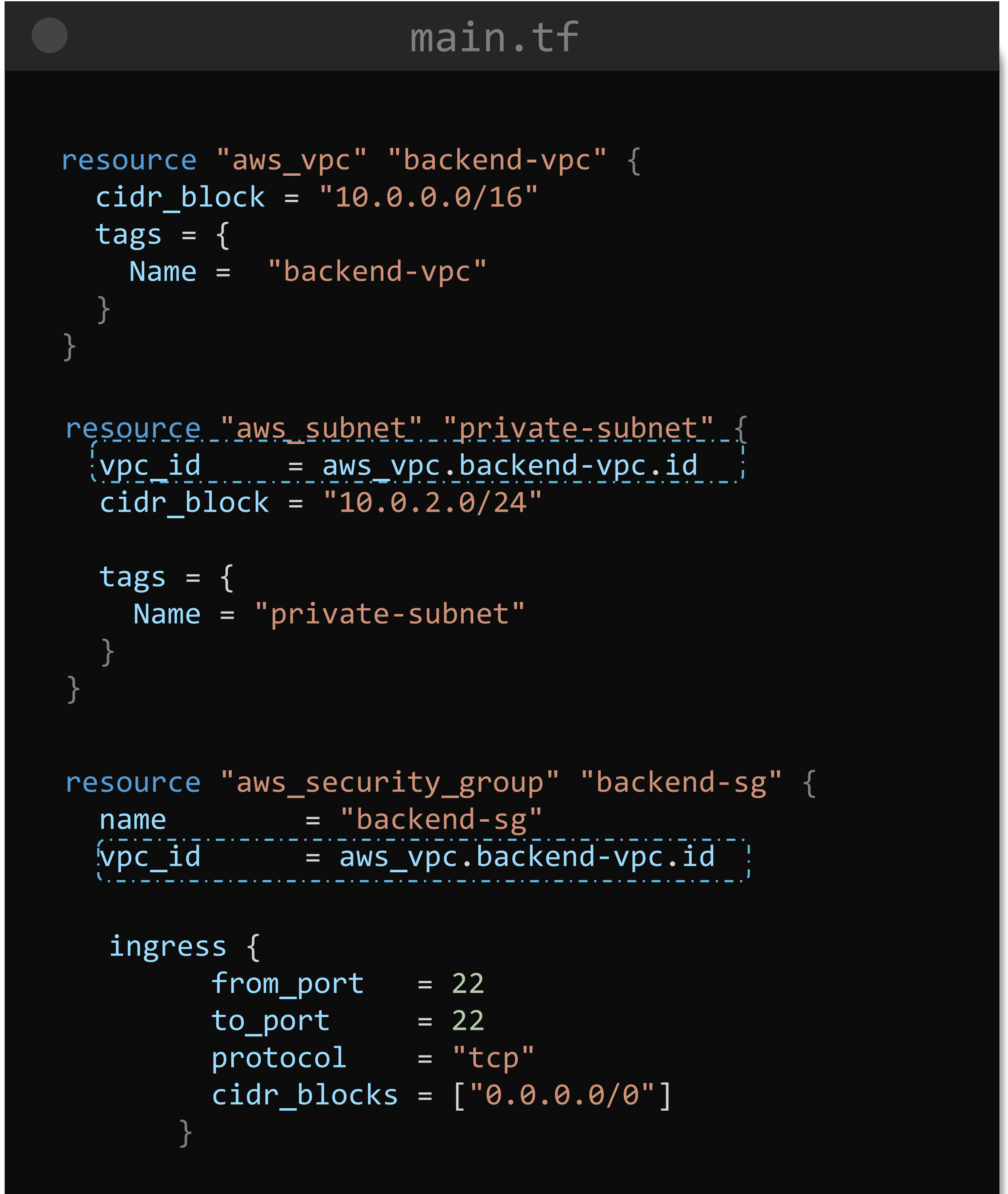
```
resource "aws_instance"  "backend" {
  ami           = var.ami
  instance_type = var.instance_type
  count    = length(var.backend-servers)
  tags = {
       Name = var.backend-servers[count.index]
  }
}
```



variables.tf

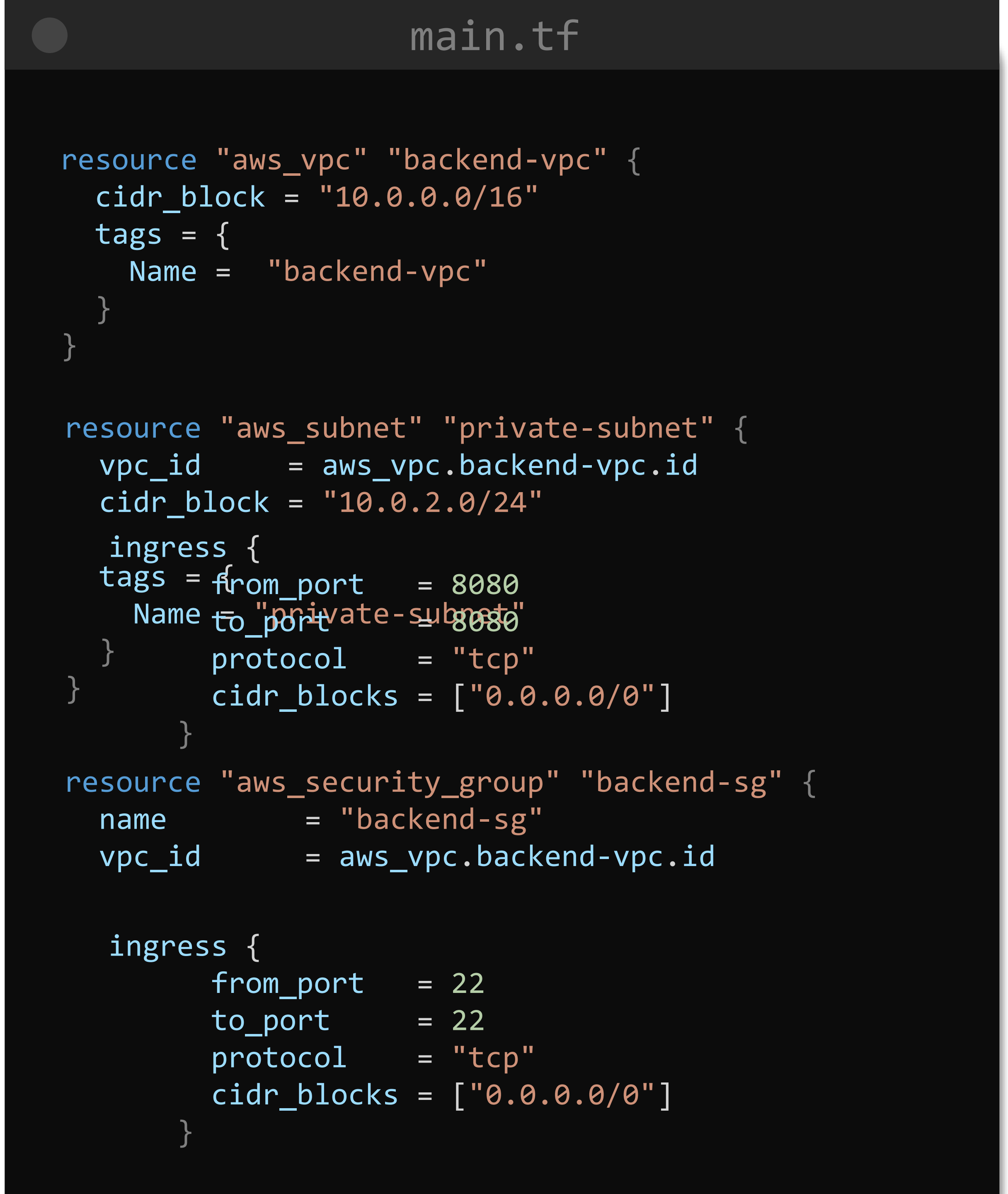
```
variable "ami" {
  default = "ami-06178cf087598769c"
}
variable "instance_type" {
  default = "m5.large"
}

variable  "backend-servers" {
    type = list
    default = ["server1", "server2"]

}
```


Amazon VPC
Security Group
Inbound 8080, 22
server1
server2
VPC – 10.0.0.0/16



main.tf

```
resource "aws_vpc" "backend-vpc" {
  cidr_block = "10.0.0.0/16"
  tags = {
    Name =  "backend-vpc"
  }
}

resource "aws_subnet" "private-subnet" {
  vpc_id     = aws_vpc.backend-vpc.id
  cidr_block = "10.0.2.0/24"

  tags = {
    Name = "private-subnet"
  }
}

resource "aws_security_group" "backend-sg" {
  name         = "backend-sg"
  vpc_id       = aws_vpc.backend-vpc.id

   ingress {
         from_port   = 22
         to_port     = 22
         protocol    = "tcp"
         cidr_blocks = ["0.0.0.0/0"]
      }
```



main.tf

```
resource "aws_vpc" "backend-vpc" {
  cidr_block = "10.0.0.0/16"
  tags = {
    Name =  "backend-vpc"
  }
}

resource "aws_subnet" "private-subnet" {
  vpc_id     = aws_vpc.backend-vpc.id
  cidr_block = "10.0.2.0/24"
  tags = {
    Name = "private-subnet"
  }
}
```

```
   ingress {
         from_port   = 8080
         to_port     = 8080
         protocol    = "tcp"
         cidr_blocks = ["0.0.0.0/0"]
      }
```

```
resource "aws_security_group" "backend-sg" {
  name        = "backend-sg"
  vpc_id      = aws_vpc.backend-vpc.id


   ingress {
         from_port   = 22
         to_port     = 22
         protocol    = "tcp"
         cidr_blocks = ["0.0.0.0/0"]
      }
```

main.tf

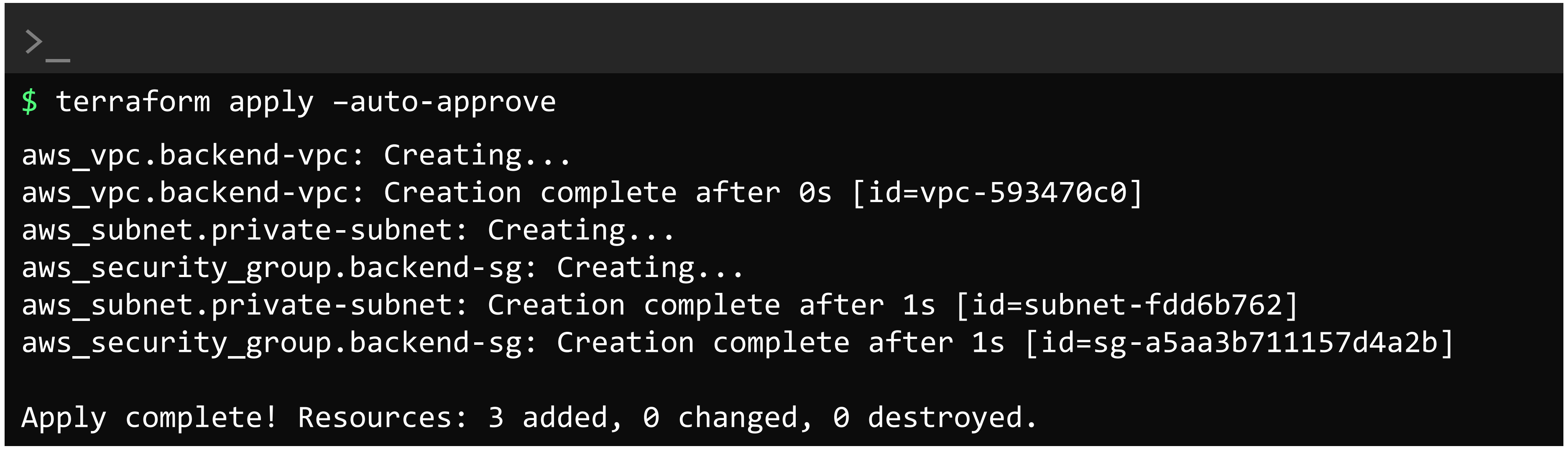
```
resource "aws_security_group" "backend-sg" {
  name         = "backend-sg"
  vpc_id       = aws_vpc.backend-vpc.id

   ingress {
         from_port   = 22
         to_port     = 22
         protocol    = "tcp"
         cidr_blocks = ["0.0.0.0/0"]
       }
   ingress {
         from_port   = 8080
         to_port     = 8080
         protocol    = "tcp"
         cidr_blocks = ["0.0.0.0/0"]
       }

 }
```

main.tf

```
resource "aws_security_group" "backend-sg" {
  name          = "backend-sg"
  vpc_id        = aws_vpc.backend-vpc.id
   dynamic "ingress" {

   for_each = var.ingress_ports

   content {
         from_port    = ingress.value
         to_port      = ingress.value
         protocol     = "tcp"
         cidr_blocks = ["0.0.0.0/0"]
      }
   }


 }
```

variables.tf

```
variable  "ingress_ports" {
    type = list
    default = [22, 8080]

}
```



```
>_
$ terraform apply -auto-approve
aws_vpc.backend-vpc: Creating...
aws_vpc.backend-vpc: Creation complete after 0s [id=vpc-593470c0]
aws_subnet.private-subnet: Creating...
aws_security_group.backend-sg: Creating...
aws_subnet.private-subnet: Creation complete after 1s [id=subnet-fdd6b762]
aws_security_group.backend-sg: Creation complete after 1s [id=sg-a5aa3b711157d4a2b]

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.
```

main.tf

```
resource "aws_security_group" "backend-sg" {
  name          = "backend-sg"
  vpc_id        = aws_vpc.backend-vpc.id

  dynamic "ingress" {
  iterator = port
  for_each = var.ingress_ports

  content {
        from_port    = port.value
        to_port      = port.value
        protocol     = "tcp"
        cidr_blocks = ["0.0.0.0/0"]
      }

  }


 }
```

variables.tf

```
variable  "ingress_ports" {
    type = list
    default = [22, 8080]

}
```

KODEKLOUD

main.tf

```
resource "aws_security_group" "backend-sg" {
  name         = "backend-sg"
  vpc_id       = aws_vpc.backend-vpc.id

   dynamic "ingress" {
   iterator = port
   for_each = var.ingress_ports
   content {
         from_port   = port.value
         to_port     = port.value
         protocol    = "tcp"
         cidr_blocks = ["0.0.0.0/0"]
       }
   }


 }
```

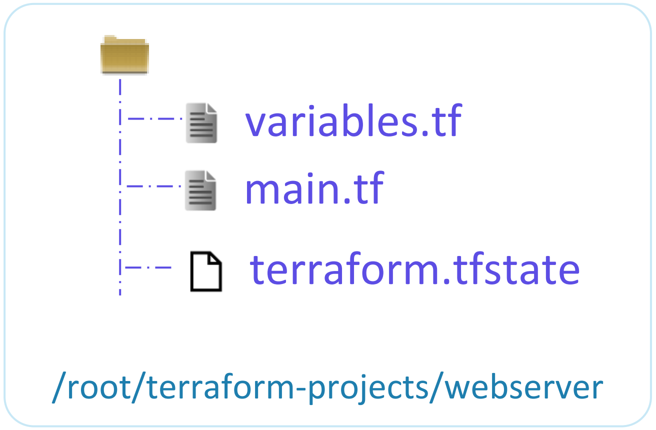
variables.tf

```
 variable  "ingress_ports" {
     type = list
     default = [22, 8080]

 }


output "to_ports" {
  value = aws_security_group.backend-sg.ingress[*].to_port

}
```

```
>_
$ terraform output
to_ports = [
  22,
  8080,
]
```

KODEKLOUD

Terraform Workspaces



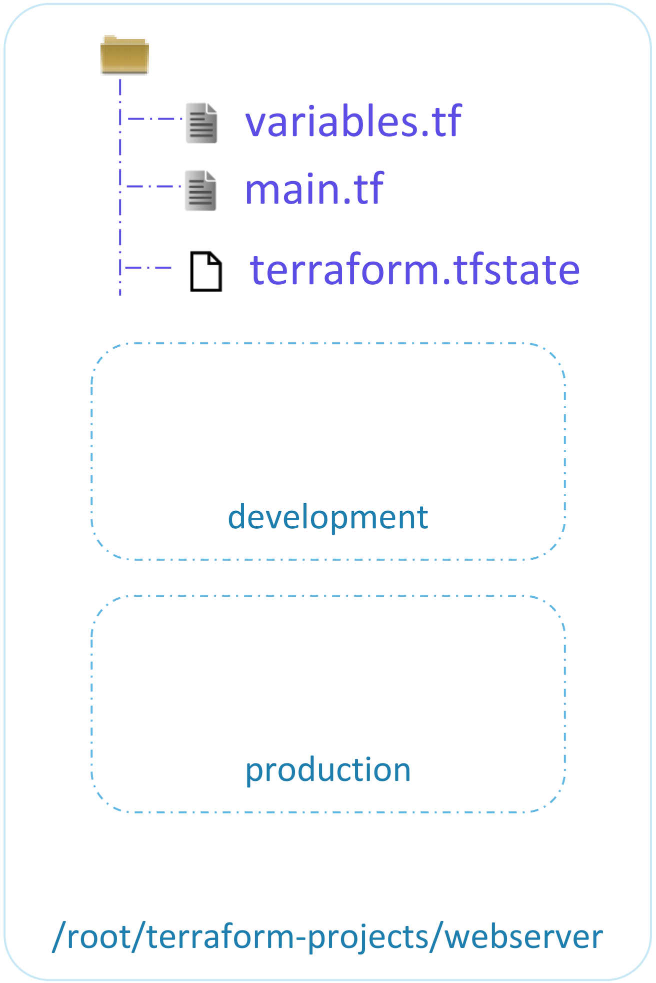
main.tf

```
resource "aws_instance" "webserver" {
    ami = var.ami
    instance_type = var.instance_type
    tags = {
        Environment = "Development"
    }
}
```

variable.tf

```
variable "ami" {
    default = "ami-24e140119877avm"
}
variable "instance_type" {
    default = "t2.micro"
}
variable  "region" {
    default =  "ca-central-1"
}
```

KODEKLOUD

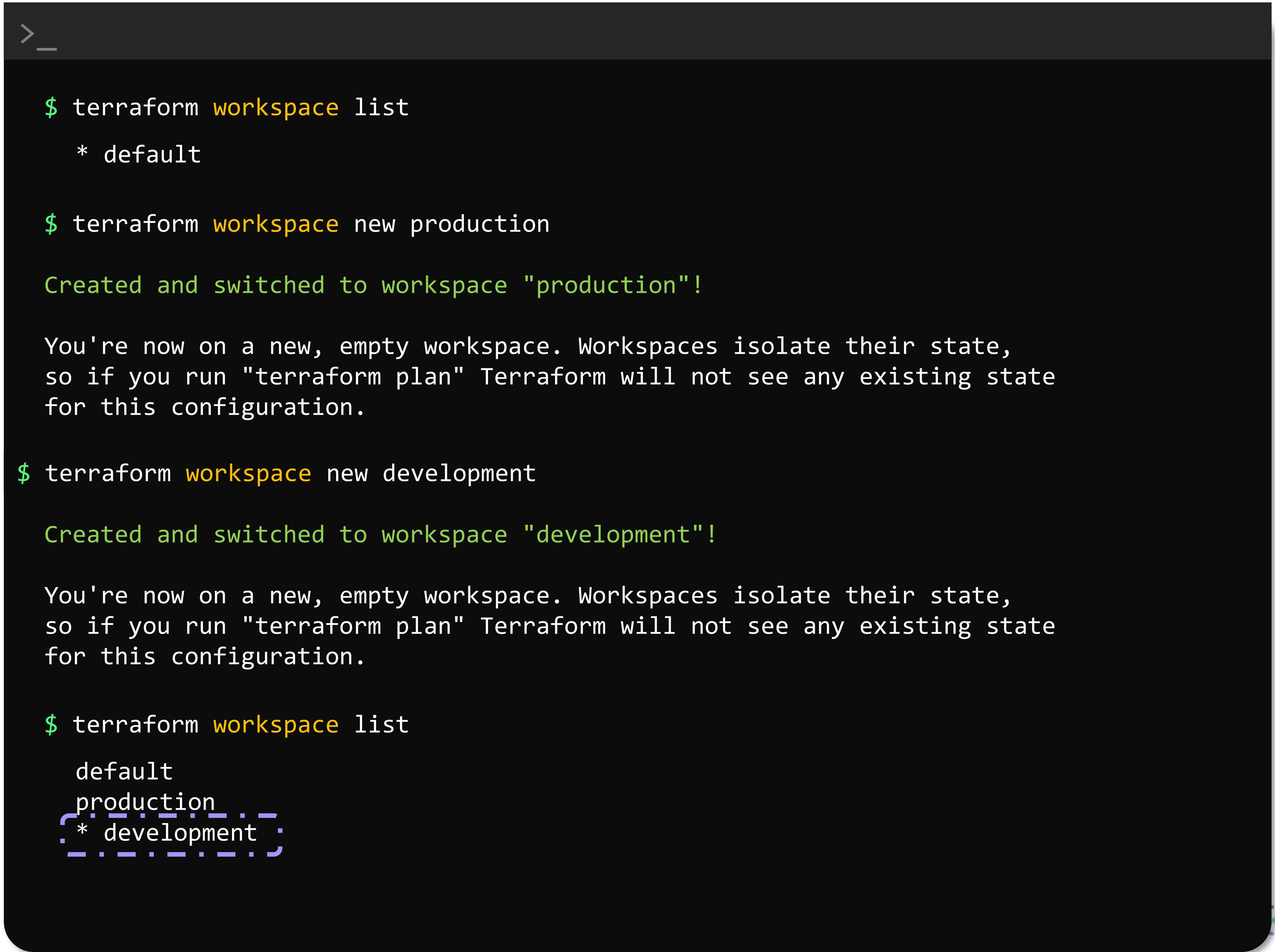


main.tf

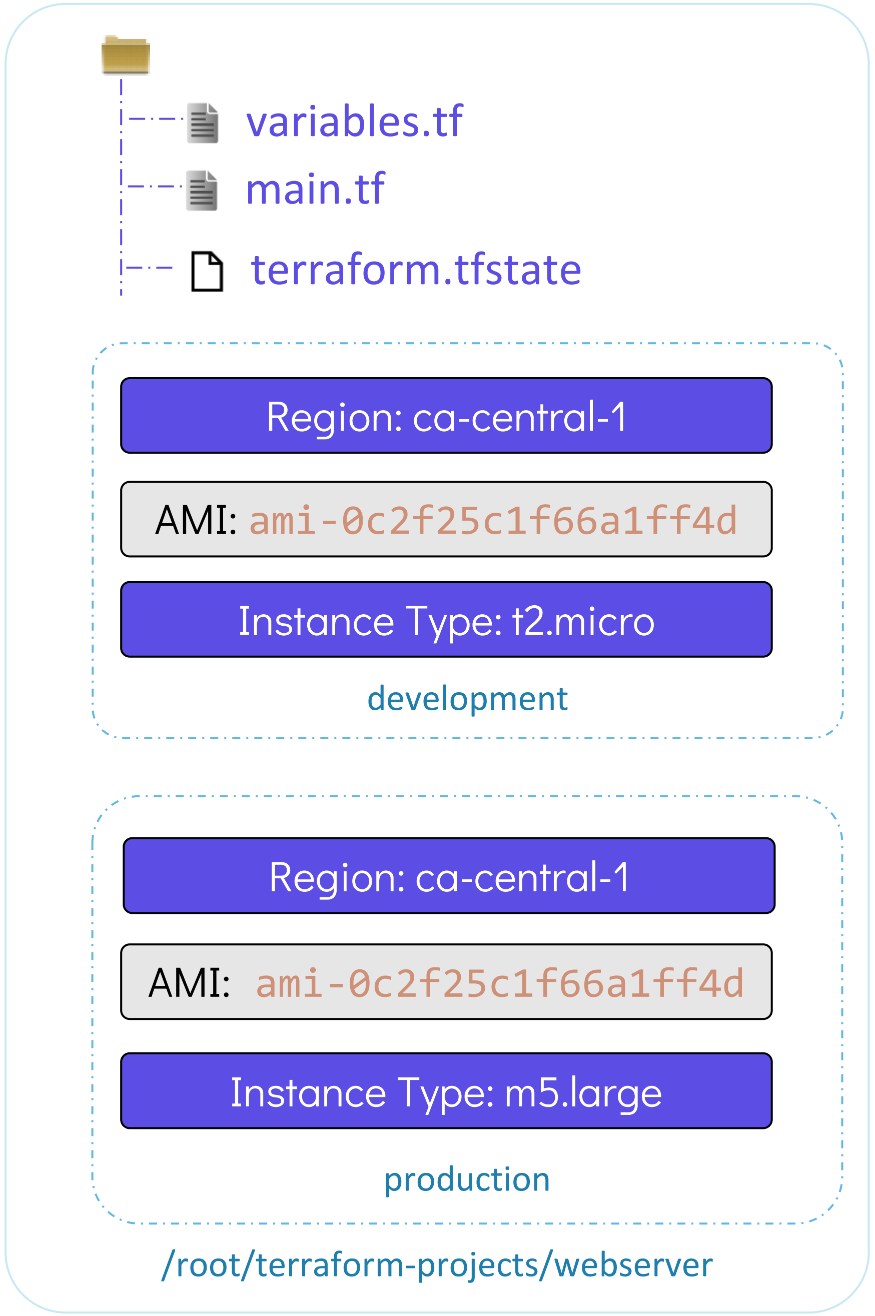
```
resource "aws_instance" "webserver" {
    ami = var.ami
    instance_type = var.instance_type
    tags = {
        Environment = "Development"
    }
}
```

variable.tf

```
variable "ami" {
    default = "ami-24e140119877avm"
}
variable "instance_type" {
    default = "t2.micro"
}
variable  "region" {
    default =  "ca-central-1"
}
```



```
$ terraform workspace list
    * default

$ terraform workspace new production

Created and switched to workspace "production"!

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 development

Created and switched to workspace "development"!

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 list
    default
    production
  * development
```



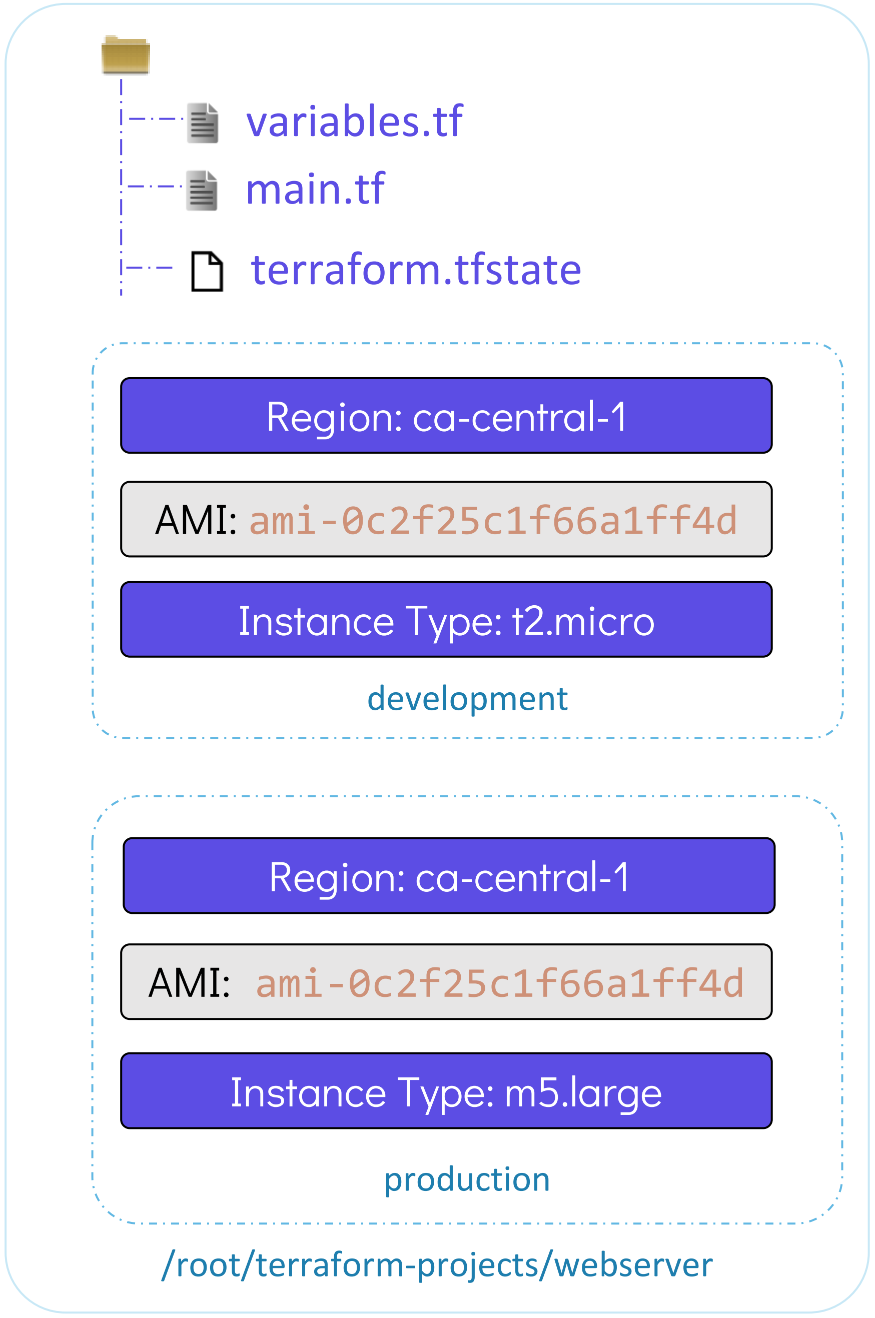
main.tf

```
resource "aws_instance" "webserver" {
    ami = var.ami
    instance_type = var.instance_type
    tags = {
        Environment = "Development"
    }
}
```

variable.tf

```
variable "ami" {
    default = "ami-24e140119877avm"
}
variable  "region" {
    default =  "ca-central-1"
}
variable "instance_type" {
    default = "t2.micro"
}
```

KODEKLOUD



main.tf

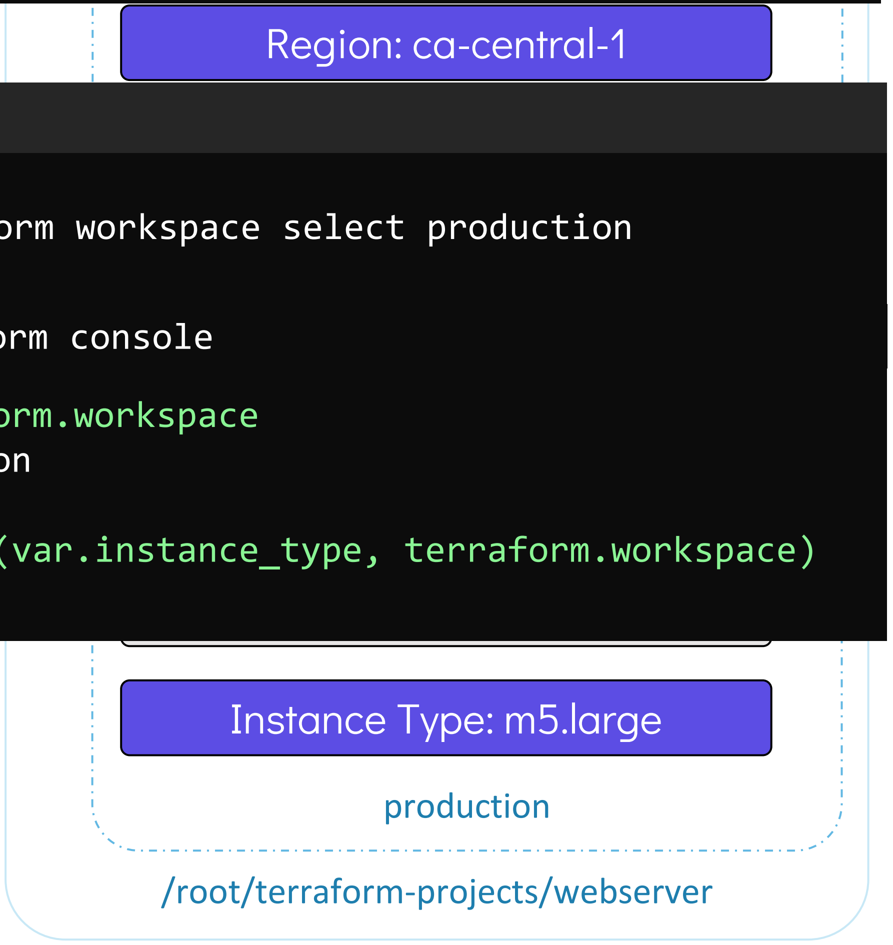
```
resource "aws_instance" "webserver" {
    ami = var.ami
    instance_type = lookup(var.instance_type, terraform.workspace)
    tags = {
        Environment = "Development"
    }
}
```

variable.tf

```
variable "ami" {
    default = "ami-24e140119877avm"
}
variable  "region" {
    default =  "ca-central-1"
}

variable "instance_type" {
    type = map
    default = {
        "development" = "t2.micro"
        "production" = "m5.large"
    }
}
```

KODEKLOUD

```
$ terraform console

> terraform.workspace
development

> lookup(var.instance_type, terraform.workspace)
t2.micro
```



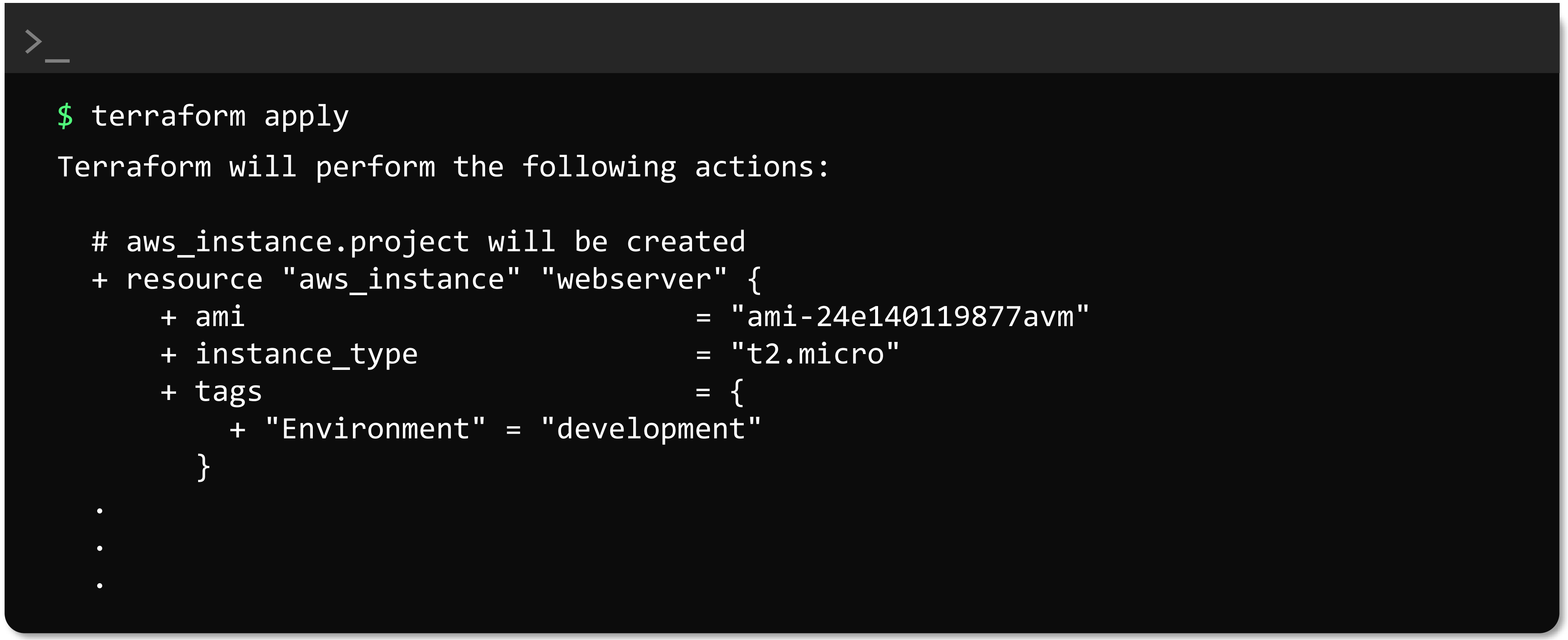
```
$ terraform workspace select production

$ terraform console

> terraform.workspace
production

> lookup(var.instance_type, terraform.workspace)
m5.large
```

main.tf

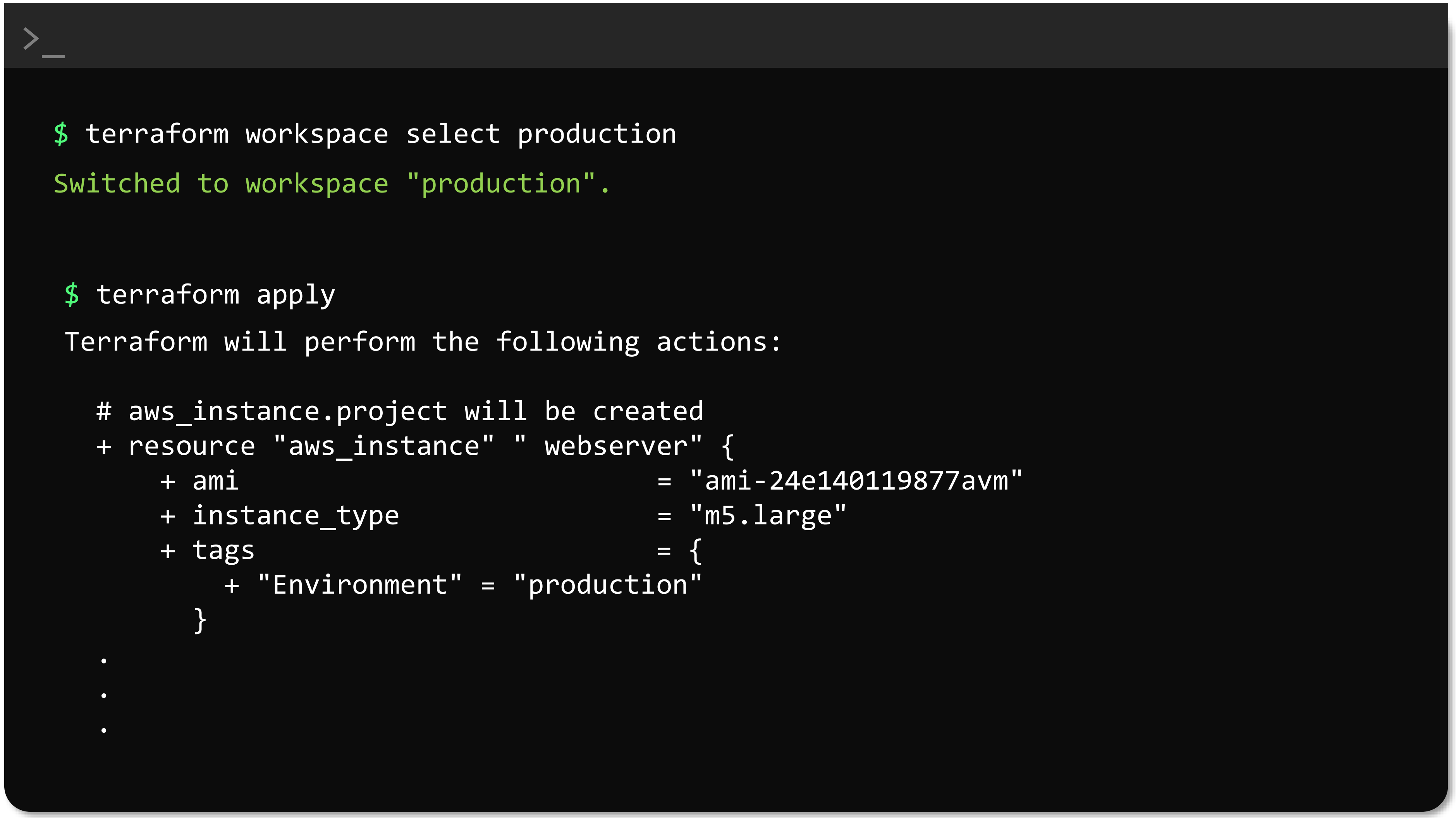
```
resource "aws_instance" "webserver" {
    ami = var.ami
    instance_type =  lookup(var.instance_type, terraform.workspace)
    tags = {
        Environment = "Development"
    }
}
```

variable.tf

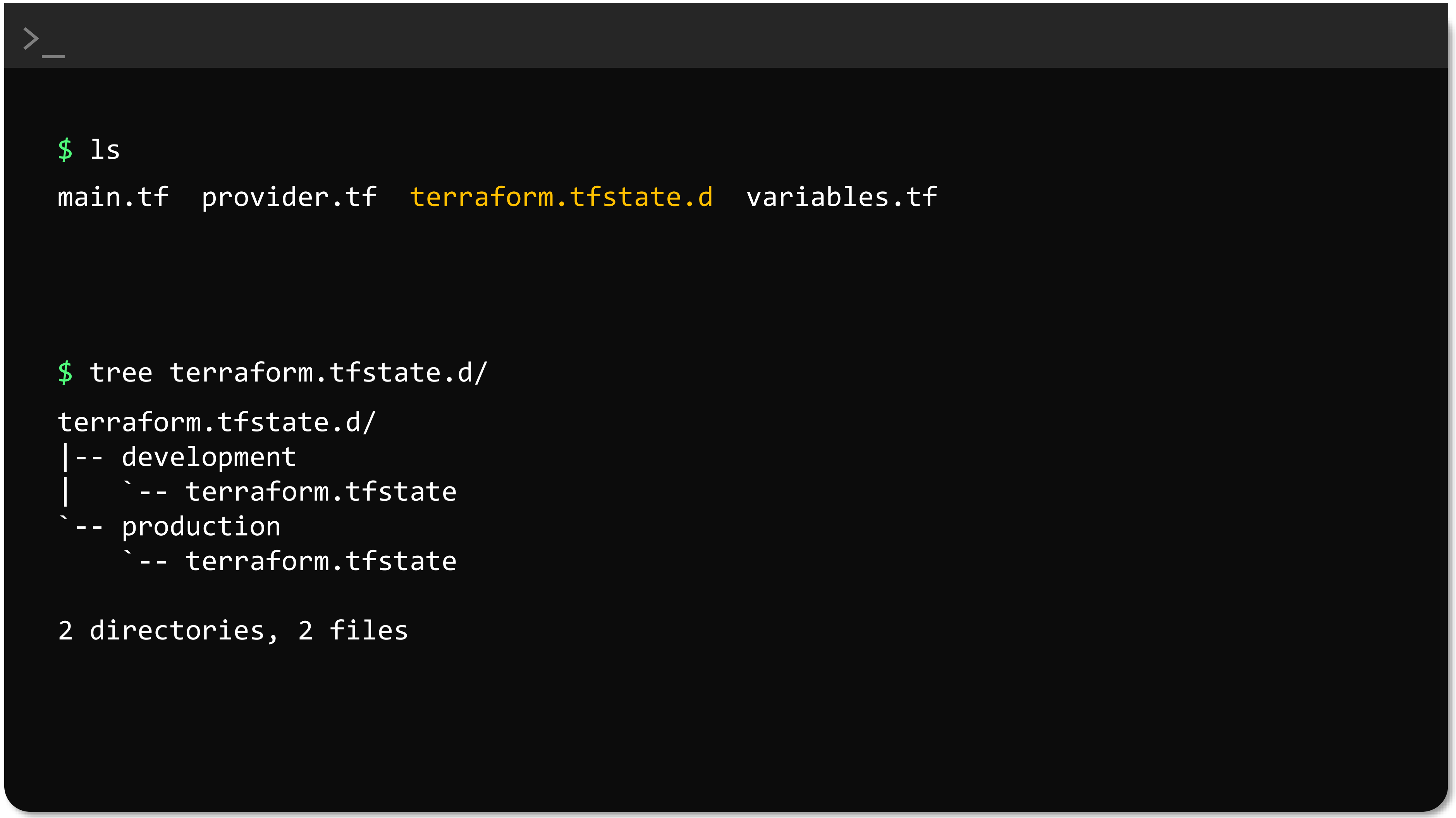
```
variable "ami" {
    default = "ami-24e140119877avm"
}
variable  "region" {
    default =  "ca-central-1"
}
variable "instance_type" {
    type = map
    default = {
        "development" = "t2.micro"
        "production" = "m5.large"
    }
}
```



```
$ terraform apply
Terraform will perform the following actions:

  # aws_instance.project will be created
  + resource "aws_instance" "webserver" {
      + ami                          = "ami-24e140119877avm"
      + instance_type                = "t2.micro"
      + tags                         = {
          + "Environment" = "development"
        }
  .
  .
  .
```



```
$ terraform workspace select production
Switched to workspace "production".


 $ terraform apply
 Terraform will perform the following actions:

   # aws_instance.project will be created
   + resource "aws_instance" " webserver" {
       + ami                              = "ami-24e140119877avm"
       + instance_type                    = "m5.large"
       + tags                             = {
           + "Environment" = "production"
         }
   .
   .
   .
```



```
$ ls
main.tf  provider.tf  terraform.tfstate.d  variables.tf
```

```
$ tree terraform.tfstate.d/
terraform.tfstate.d/
|-- development
|   `-- terraform.tfstate
`-- production
    `-- terraform.tfstate

2 directories, 2 files
```

local_file

random_pet

aws_instance

aws_s3_bucket

aws_dynamodb_table

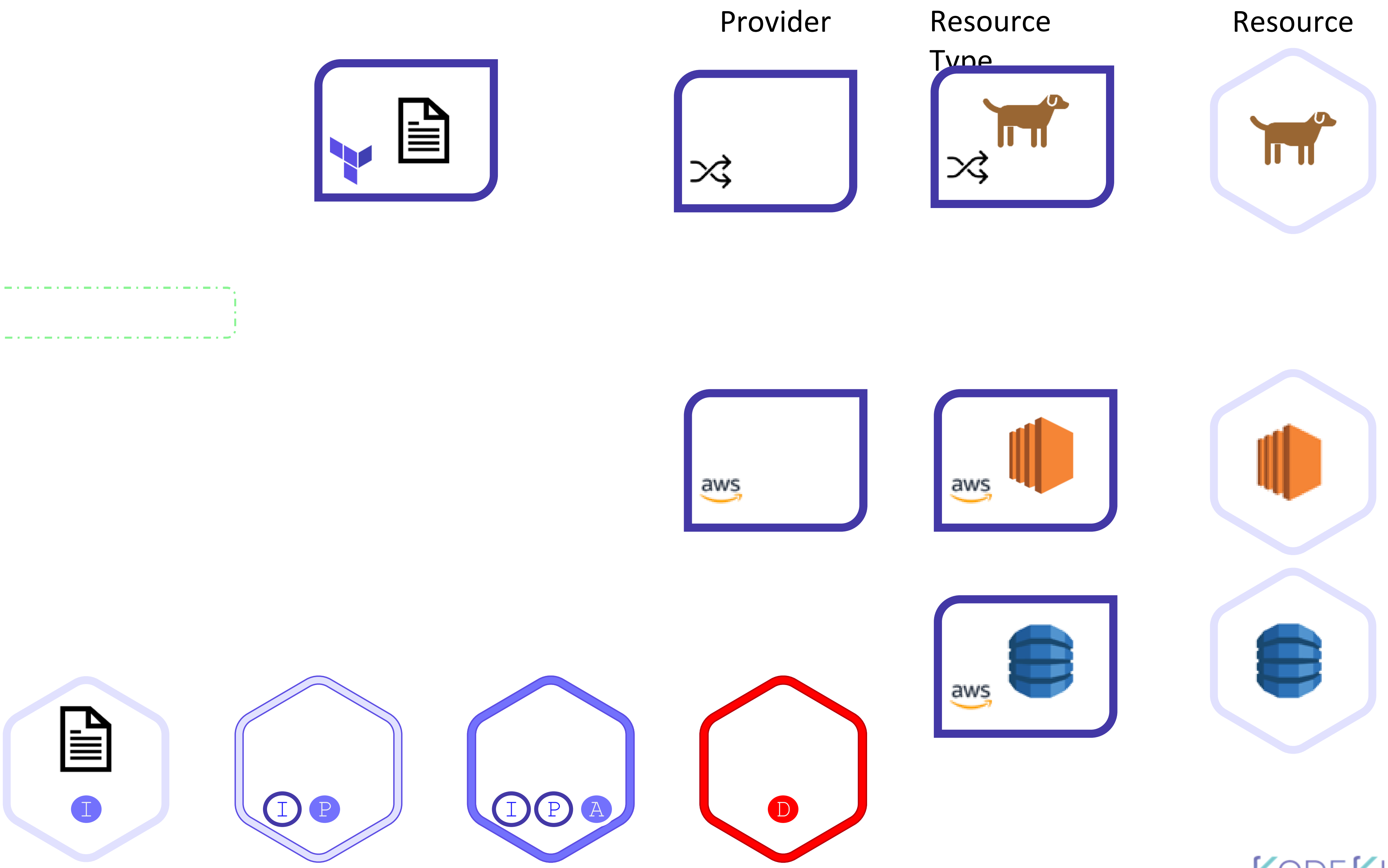
aws_instance + S3 Bucket

aws_iam_policy

aws_iam_user

Provider

Resource



KODEKLOUD