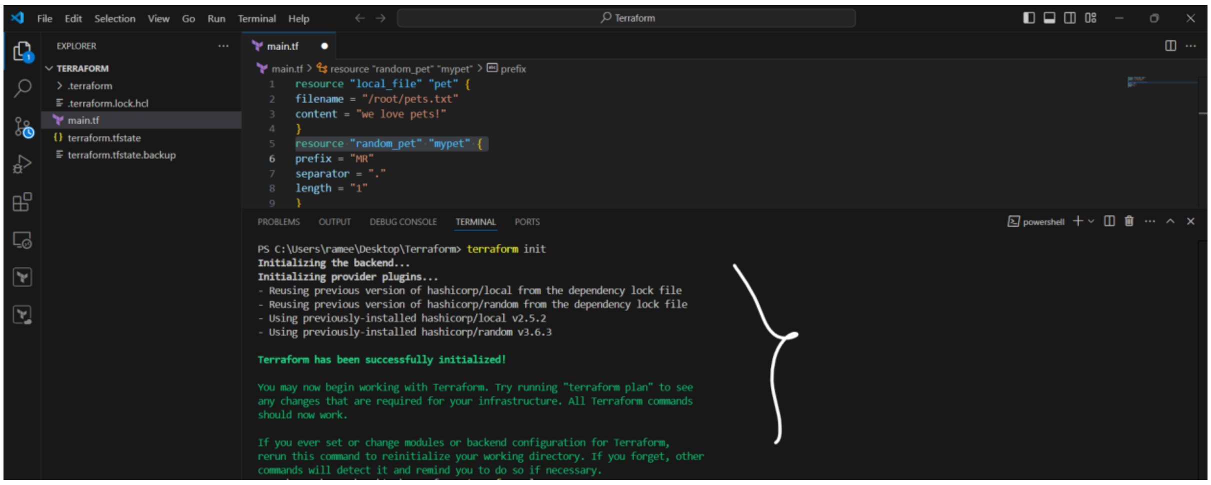
1) Execute all the templates shown in video.

To execute the all

I am using the same file to execute the task's.

This is my actual file.

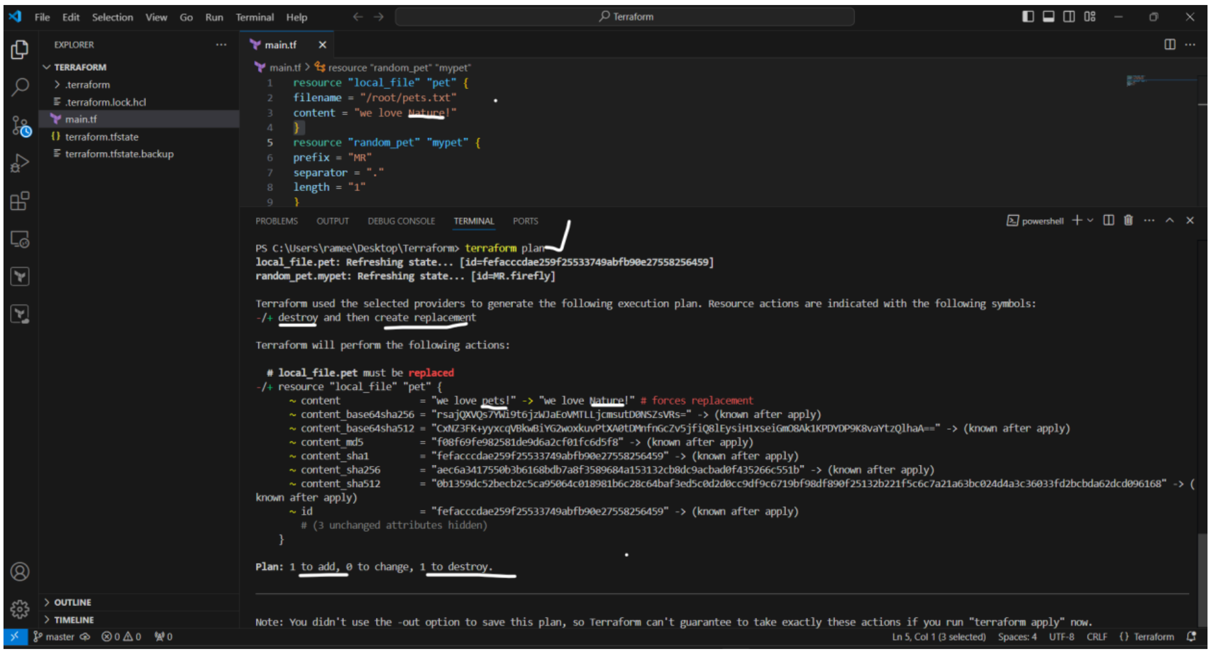


```
resource "local_file" "pet" {
filename = "/root/pets.txt"
content = "we love pets!"
}
resource "random_pet" "mypet" {
prefix = "MR"
separator = "."
length = "1"
}
```

```
PS C:\Users\ramee\Desktop\Terraform> terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/local from the dependency lock file
- Reusing previous version of hashicorp/random from the dependency lock file
- Using previously-installed hashicorp/local v2.5.2
- Using previously-installed hashicorp/random v3.6.3

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.
```

In this file I need to edit the content in the pets.txt file

- Here below I am checking the file plan.
- In the plan it will show the what is the actual and what we change the content.
- Below it's shows --- 1 to add and 1 to destroy.



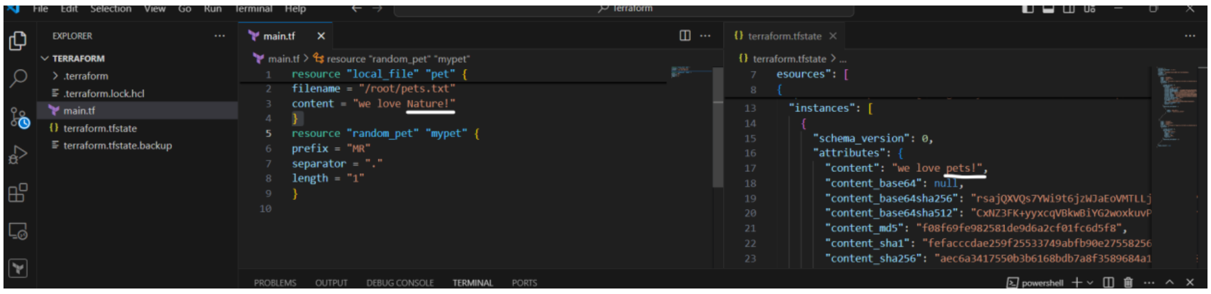
```
resource "local_file" "pet" {
filename = "/root/pets.txt"
content = "we love Nature!"
}
resource "random_pet" "mypet" {
prefix = "MR"
separator = "."
length = "1"
}
```

```
PS C:\Users\ramee\Desktop\Terraform> terraform plan
local_file.pet: Refreshing state... [id=fefacccdae259f25533749abfb90e27558256459]
random_pet.mypet: Refreshing state... [id=MR.firefly]

Terraform used the selected providers to generate the following execution plan. 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!" -> "we love Nature!" # forces replacement
      ~ content_base64sha256 = "rsajQXVQs7YWi9t6jzWJaEoVMTLLjcmsutD0NSZsVRs=" -> (known after apply)
      ~ content_base64sha512 = "CxNZ3FK+yyxcqVBkwBiYG2woxkuvPtXA0tDMnfnGcZv5jfiQ8lEysiH1xseiGmO8Ak1KPDYDP9K8vaYtzQlhaA==" -> (known after apply)
      ~ content_md5          = "f08f69fe982581de9d6a2cf01fc6d5f8" -> (known after apply)
      ~ content_sha1         = "fefacccdae259f25533749abfb90e27558256459" -> (known after apply)
      ~ content_sha256       = "aec6a3417550b3b6168bdb7a8f3589684a153132cb8dc9acbad0f435266c551b" -> (known after apply)
      ~ content_sha512       = "0b1359dc52becb2c5ca95064c018981b6c28c64baf3ed5c0d2d0cc9df9c6719bf98df890f25132b221f5c6c7a21a63bc024d4a3c36033fd2bcbda62dcd096168" -> (
known after apply)
      ~ id                   = "fefacccdae259f25533749abfb90e27558256459" -> (known after apply)
        # (3 unchanged attributes hidden)
    }

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

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

Before entering the terraform apply we can check the main.tf file and terraform.tfstate.

Here observer two file in main.tf content we change.

terraform.tfstate file have pervious content as we love pets!

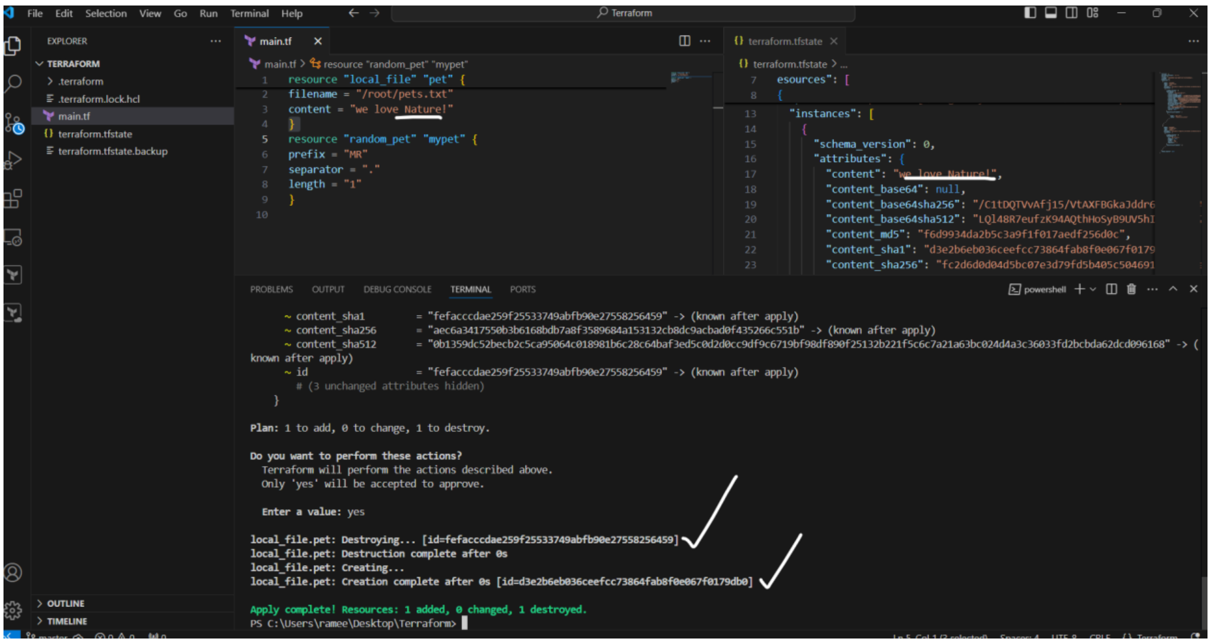


```
resource "local_file" "pet" {
filename = "/root/pets.txt"
content = "we love Nature!"
}
resource "random_pet" "mypet" {
prefix = "MR"
separator = "."
length = "1"
}
```

```
esources": [
{
  "instances": [
    {
      "schema_version": 0,
      "attributes": {
        "content": "we love pets!",
        "content_base64": null,
        "content_base64sha256": "rsajQXVQs7YWi9t6jzWJaEoVMTLLj
        "content_base64sha512": "CxNZ3FK+yyxcqVBkwBiYG2woxkuvP
        "content_md5": "f08f69fe982581de9d6a2cf01fc6d5f8",
        "content_sha1": "fefacccdae259f25533749abfb90e27558256
        "content_sha256": "aec6a3417550b3b6168bdb7a8f3589684a1
```

After entering CMD: terraform apply

First it will the previous content and create new content.



```
resource "local_file" "pet" {
filename = "/root/pets.txt"
content = "we love Nature!"
}
resource "random_pet" "mypet" {
prefix = "MR"
separator = "."
length = "1"
}
```

```
esources": [
{
  "instances": [
    {
      "schema_version": 0,
      "attributes": {
        "content": "we love Nature!",
        "content_base64": null,
        "content_base64sha256": "/C1tDQTVvAfj15/VtAXFBGkaJddr6
        "content_base64sha512": "LQl48R7eufzK94AQthHoSyB9UV5hI
        "content_md5": "f6d9934da2b5c3a9f1f017aedf256d0c",
        "content_sha1": "d3e2b6eb036ceefcc73864fab8f0e067f0179
        "content_sha256": "fc2d6d0d04d5bc07e3d79fd5b405c504691
```

```
      ~ content_sha1         = "fefacccdae259f25533749abfb90e27558256459" -> (known after apply)
      ~ content_sha256       = "aec6a3417550b3b6168bdb7a8f3589684a153132cb8dc9acbad0f435266c551b" -> (known after apply)
      ~ content_sha512       = "0b1359dc52becb2c5ca95064c018981b6c28c64baf3ed5c0d2d0cc9df9c6719bf98df890f25132b221f5c6c7a21a63bc024d4a3c36033fd2bcbda62dcd096168" -> (
known after apply)
      ~ id                   = "fefacccdae259f25533749abfb90e27558256459" -> (known after apply)
        # (3 unchanged attributes hidden)
    }

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=d3e2b6eb036ceefcc73864fab8f0e067f0179db0]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
PS C:\Users\ramee\Desktop\Terraform>
```

Now I want to change the order first I want create then delete.

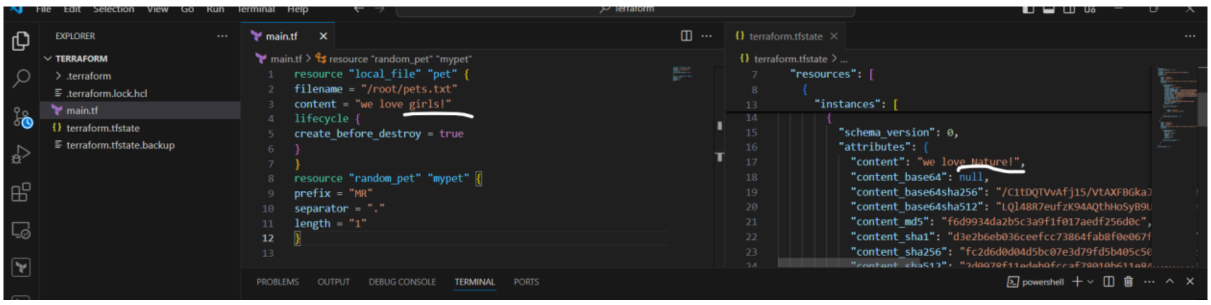
Terraform Lifecycle Rule Explanation: In Terraform, **lifecycle rules** allow us to control how resources are created, updated, or destroyed. These rules help manage situations where certain actions on resources need to be restricted or controlled.

Key Lifecycle Rules:

create_before_destroy:

- Ensures that Terraform creates a new resource before destroying the old one.
- Useful when downtime must be avoided.
- Example: Updating a load balancer without taking it offline.

Here I changing the content in main.tf .

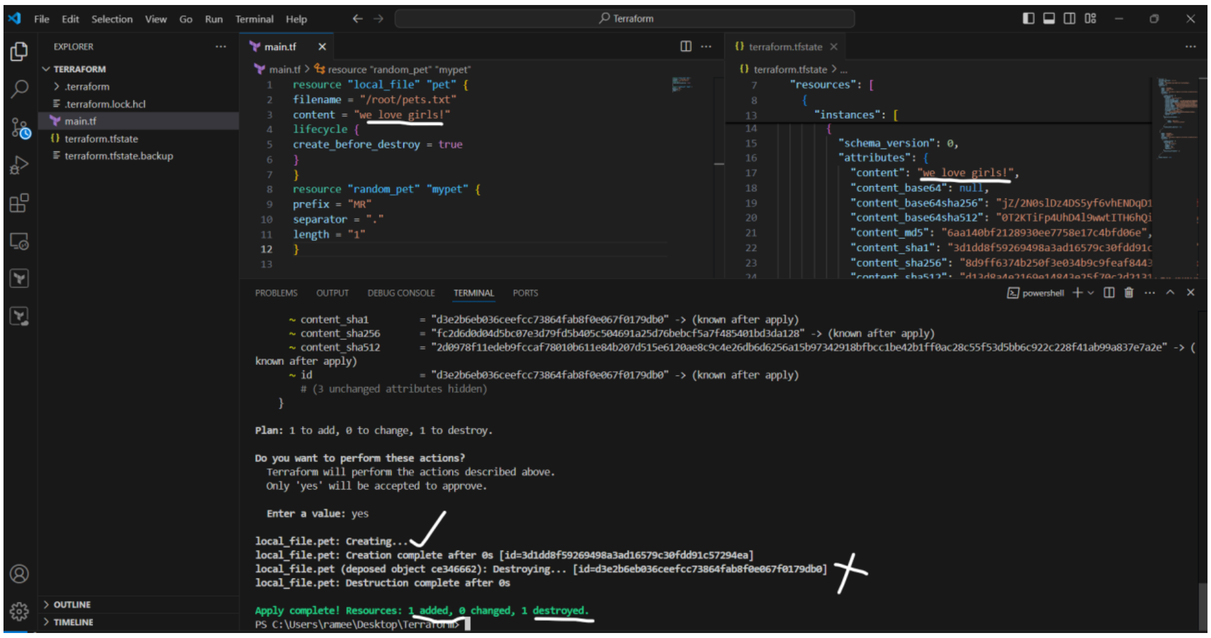


```
resource "local_file" "pet" {
filename = "/root/pets.txt"
content = "we love girls!"
lifecycle {
create_before_destroy = true
}
}
resource "random_pet" "mypet" {
prefix = "MR"
separator = "."
length = "1"
}
```

```
  "resources": [
    {
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "we love Nature!",
            "content_base64": null,
            "content_base64sha256": "/C1tDQTVvAfj15/VtAXFBGkaJ
            "content_base64sha512": "LQl48R7eufzK94AQthHoSyB9U
            "content_md5": "f6d9934da2b5c3a9f1f017aedf256d0c",
            "content_sha1": "d3e2b6eb036ceefcc73864fab8f0e067f
            "content_sha256": "fc2d6d0d04d5bc07e3d79fd5b405c5e
```

After entering the terraform apply.

Here first created and deleted the content.



```
resource "local_file" "pet" {
filename = "/root/pets.txt"
content = "we love girls!"
lifecycle {
create_before_destroy = true
}
}
resource "random_pet" "mypet" {
prefix = "MR"
separator = "."
length = "1"
}
```

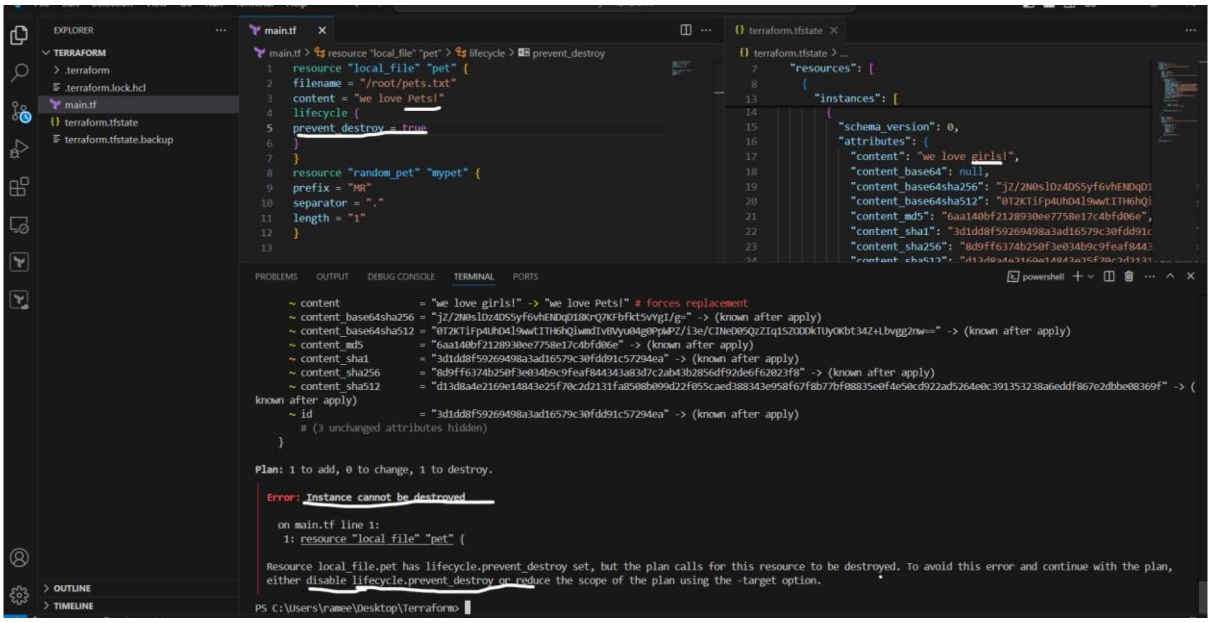
```
  "resources": [
    {
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "content": "we love girls!",
            "content_base64": null,
            "content_base64sha256": "jZ/2N0slDz4DS5yf6vhENDqD1
            "content_base64sha512": "0T2KTiFp4UhD4l9wwtITH6hQi
            "content_md5": "6aa140bf2128930ee7758e17c4bfd06e",
            "content_sha1": "3d1dd8f59269498a3ad16579c30fdd91c
            "content_sha256": "8d9ff6374b250f3e034b9c9feaf8443
```

```
      ~ content_sha1         = "d3e2b6eb036ceefcc73864fab8f0e067f0179db0" -> (known after apply)
      ~ content_sha256       = "fc2d6d0d04d5bc07e3d79fd5b405c504691a25d76bebcf5a7f485401bd3da128" -> (known after apply)
      ~ content_sha512       = "2d0978f11edeb9fccaf78010b611e84b207d515e6120ae8c9c4e26db6d6256a15b97342918bfbcc1be42b1ff0ac28c55f53d5bb6c922c228f41ab99a837e7a2e" -> (
known after apply)
      ~ id                   = "d3e2b6eb036ceefcc73864fab8f0e067f0179db0" -> (known after apply)
        # (3 unchanged attributes hidden)
    }

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: Creating...
local_file.pet: Creation complete after 0s [id=3d1dd8f59269498a3ad16579c30fdd91c57294ea]
local_file.pet (deposed object ce346662): Destroying... [id=d3e2b6eb036ceefcc73864fab8f0e067f0179db0]
local_file.pet: Destruction complete after 0s

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
PS C:\Users\ramee\Desktop\Terraform>
```

prevent_destroy:

- Prevents Terraform from accidentally destroying a resource.
- Useful for critical resources like databases or production servers.

Here I am using rule is **prevent_destroy** so it won't be delete the previous infrastructure.



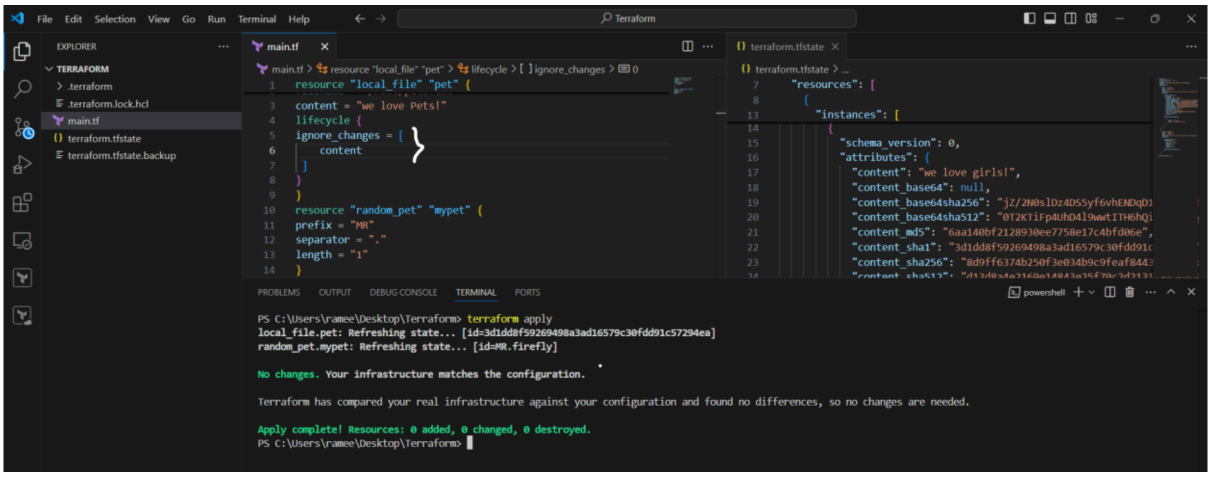
```
resource "local_file" "pet" {
filename = "/root/pets.txt"
content = "we love Pets!"
lifecycle {
prevent_destroy = true
}
}
resource "random_pet" "mypet" {
prefix = "MR"
separator = "."
length = "1"
}
```

```
    "resources": [
      {
        "instances": [
          {
            "schema_version": 0,
            "attributes": {
              "content": "we love girls!",
              "content_base64": null,
              "content_base64sha256": "jZ/2N0slDz4DS5yf6vhENDqD1
              "content_base64sha512": "0T2KTiFp4UhD4l9wwtITH6hQi
              "content_md5": "6aa140bf2128930ee7758e17c4bfd06e",
              "content_sha1": "3d1dd8f59269498a3ad16579c30fdd91c
              "content_sha256": "8d9ff6374b250f3e034b9c9feaf8443
```

```
      ~ content              = "we love girls!" -> "we love Pets!" # forces replacement
      ~ content_base64sha256 = "jZ/2N0slDz4DS5yf6vhENDqD18KrQ7KFbfkt5vYgI/g=" -> (known after apply)
      ~ content_base64sha512 = "0T2KTiFp4UhD4l9wwtITH6hQiwmdIvBVyu04g0PpWPZ/i3e/CINeD05QzZIq1SZODDkTUyOKbt34Z+Lbvgg2nw==" -> (known after apply)
      ~ content_md5          = "6aa140bf2128930ee7758e17c4bfd06e" -> (known after apply)
      ~ content_sha1         = "3d1dd8f59269498a3ad16579c30fdd91c57294ea" -> (known after apply)
      ~ content_sha256       = "8d9ff6374b250f3e034b9c9feaf844343a83d7c2ab43b2856df92de6f62023f8" -> (known after apply)
      ~ content_sha512       = "d13d8a4e2169e14843e25f70c2d2131fa8508b099d22f055caed388343e958f67f8b77bf08835e0f4e50cd922ad5264e0c391353238a6eddf867e2dbbe08369f" -> (
known after apply)
      ~ id                   = "3d1dd8f59269498a3ad16579c30fdd91c57294ea" -> (known after apply)
        # (3 unchanged attributes hidden)
    }

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

  Error: Instance cannot be destroyed

    on main.tf line 1:
     1: resource "local_file" "pet" {

  Resource local_file.pet 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 option.

PS C:\Users\ramee\Desktop\Terraform>
```

ignore_changes:

- Tells Terraform to ignore changes to specific attributes of a resource.
- Useful when some attributes are managed outside Terraform (like auto-scaling or manually updated tags).



```
resource "local_file" "pet" {
content = "we love Pets!"
lifecycle {
ignore_changes = [
    content
]
}
}
resource "random_pet" "mypet" {
prefix = "MR"
separator = "."
length = "1"
}
```

```
PS C:\Users\ramee\Desktop\Terraform> terraform apply
local_file.pet: Refreshing state... [id=3d1dd8f59269498a3ad16579c30fdd91c57294ea]
random_pet.mypet: Refreshing state... [id=MR.firefly]

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.
PS C:\Users\ramee\Desktop\Terraform>
```