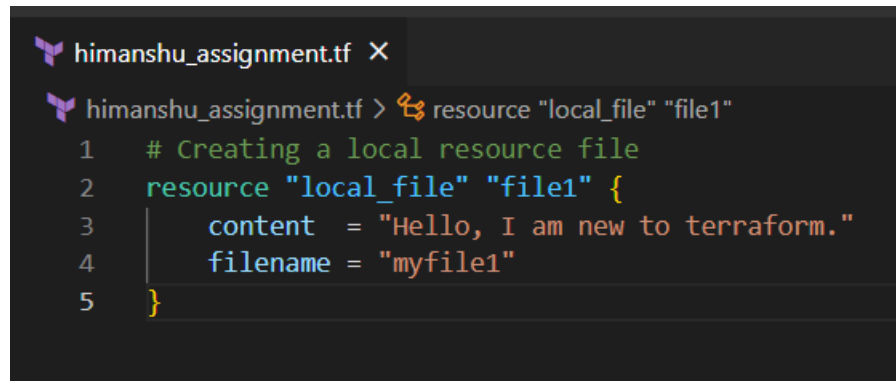


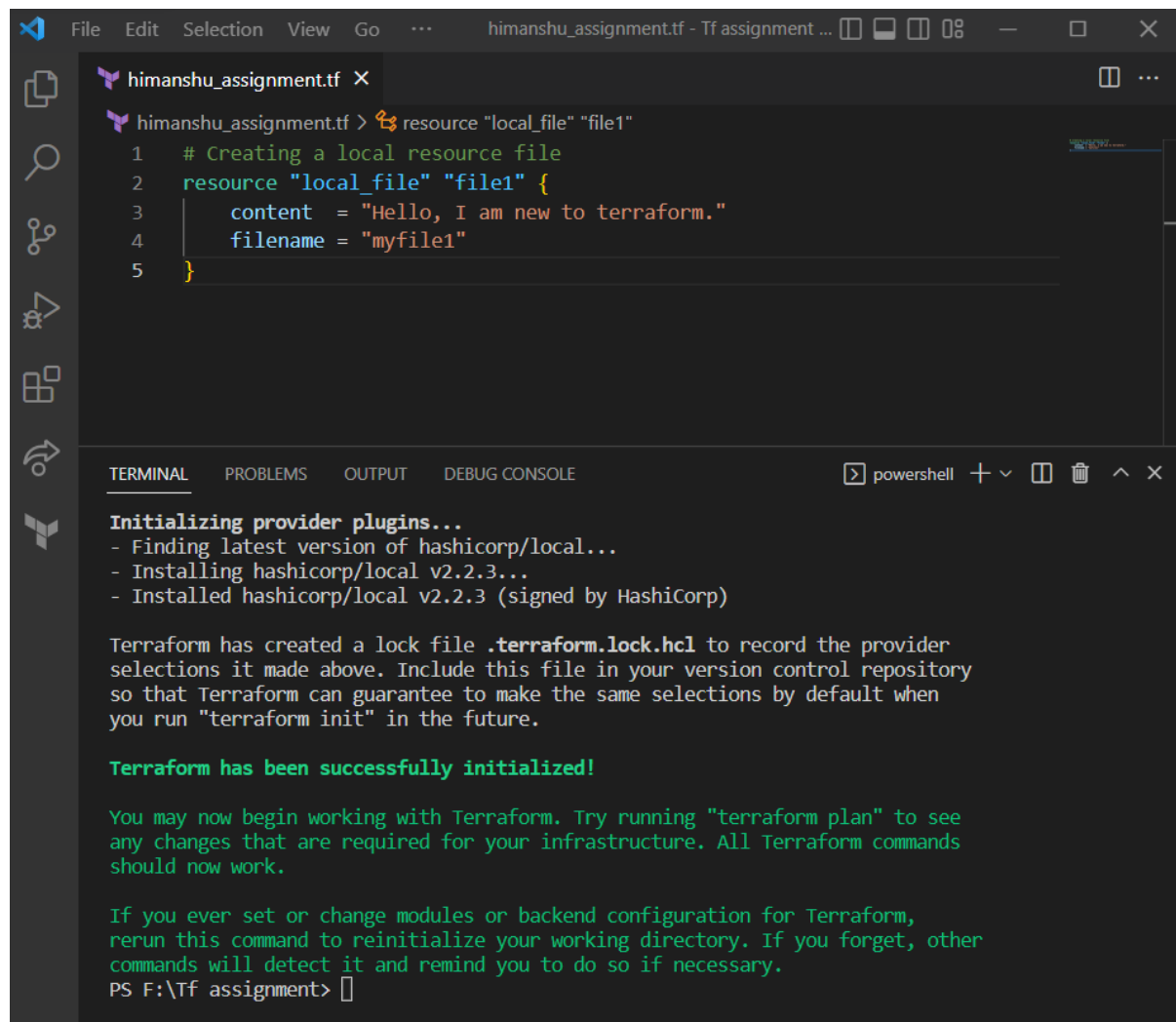
Creating a resource

Creating a local resource file with name file1



```
himanshu_assignment.tf X
himanshu_assignment.tf > resource "local_file" "file1"
1  # Creating a local resource file
2  resource "local_file" "file1" {
3      content = "Hello, I am new to terraform."
4      filename = "myfile1"
5  }
```

Terraform init



```
File Edit Selection View Go ... himanshu_assignment.tf - Tf assignment ...
himanshu_assignment.tf X
himanshu_assignment.tf > resource "local_file" "file1"
1  # Creating a local resource file
2  resource "local_file" "file1" {
3      content = "Hello, I am new to terraform."
4      filename = "myfile1"
5  }
```

INITIALIZING PROVIDER PLUGINS...

- Finding latest version of hashicorp/local...
- Installing hashicorp/local v2.2.3...
- Installed hashicorp/local v2.2.3 (signed by HashiCorp)

Terraform has created a lock file **.terraform.lock.hcl** to record the provider selections it made above. Include this file in your version control repository so that Terraform can guarantee to make the same selections by default when you run "terraform init" in the future.

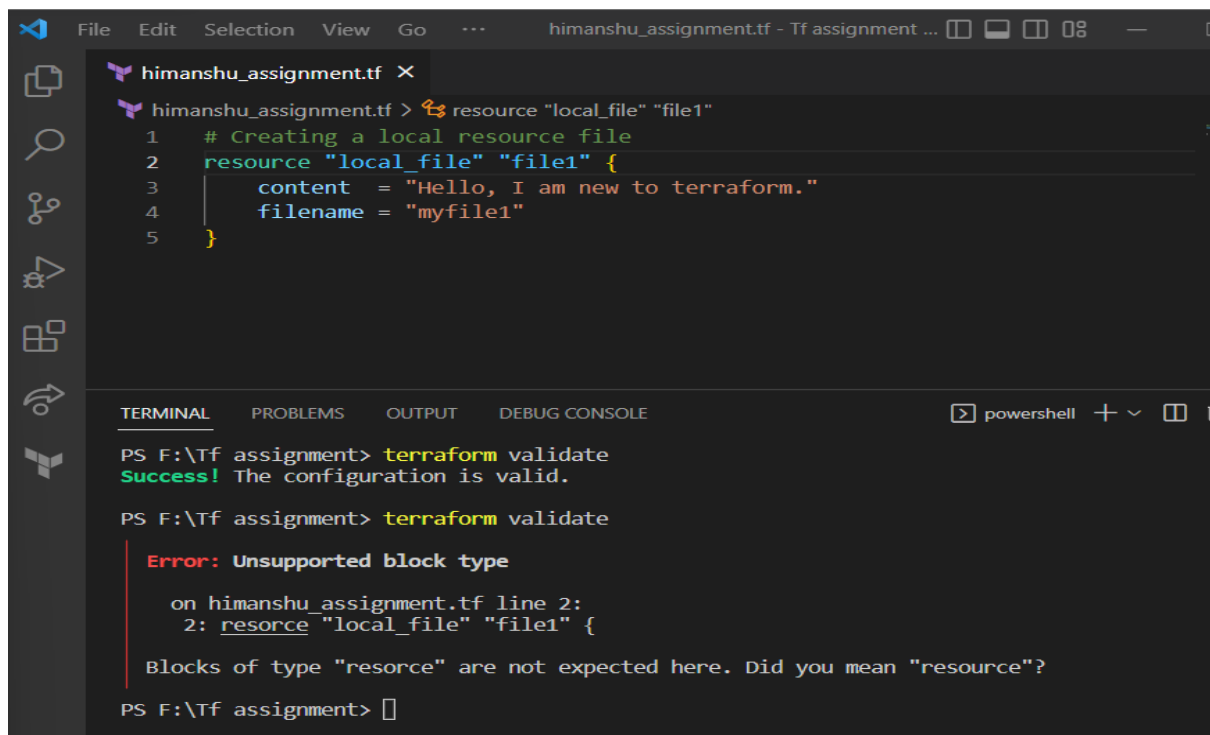
Terraform has been successfully initialized!

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.

PS F:\Tf assignment>

Terraform validate



```
himanshu_assignment.tf X
himanshu_assignment.tf > resource "local_file" "file1"
1 # Creating a local resource file
2 resource "local_file" "file1" {
3     content = "Hello, I am new to terraform."
4     filename = "myfile1"
5 }

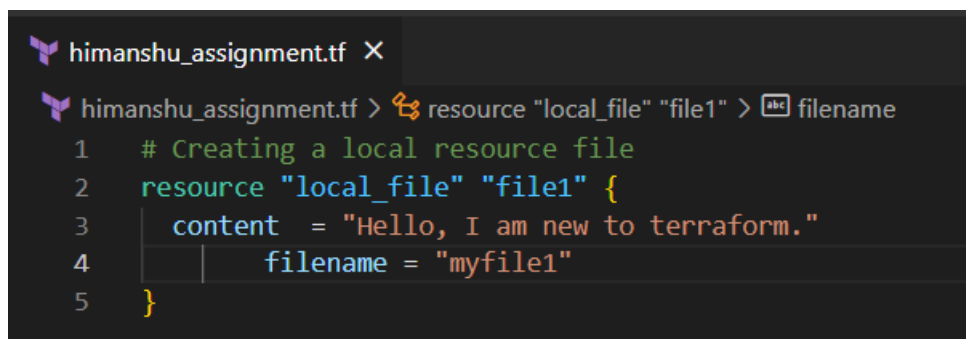
TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE powershell + v []
PS F:\Tf assignment> terraform validate
Success! The configuration is valid.

PS F:\Tf assignment> terraform validate

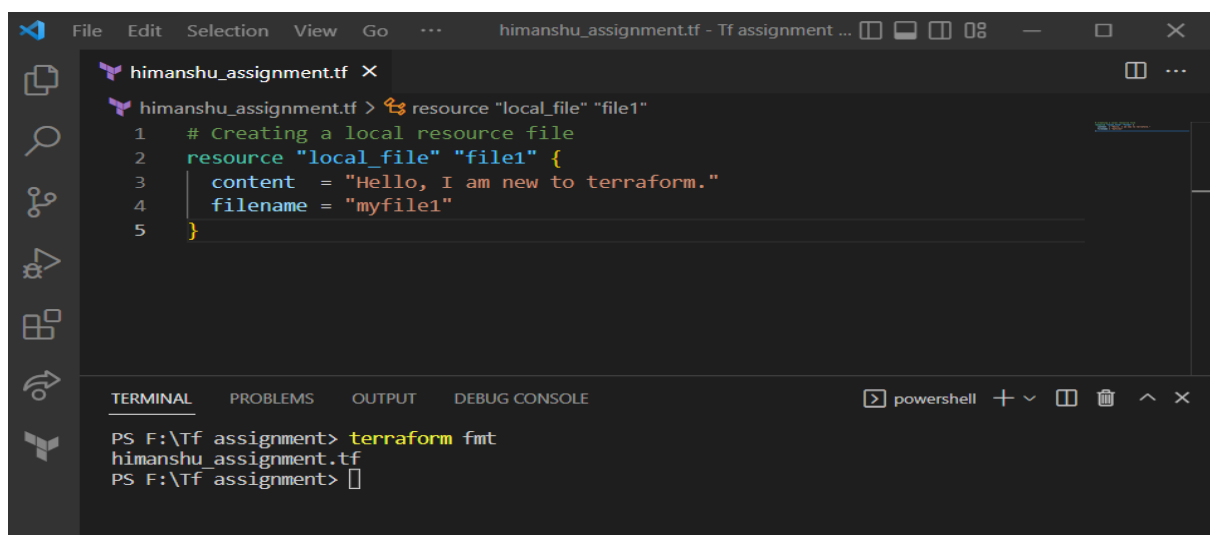
Error: Unsupported block type
   on himanshu_assignment.tf line 2:
   2: resorce "local_file" "file1" {
     ~~~~~
Blocks of type "resorce" are not expected here. Did you mean "resource"?

PS F:\Tf assignment>
```

Terraform fmt



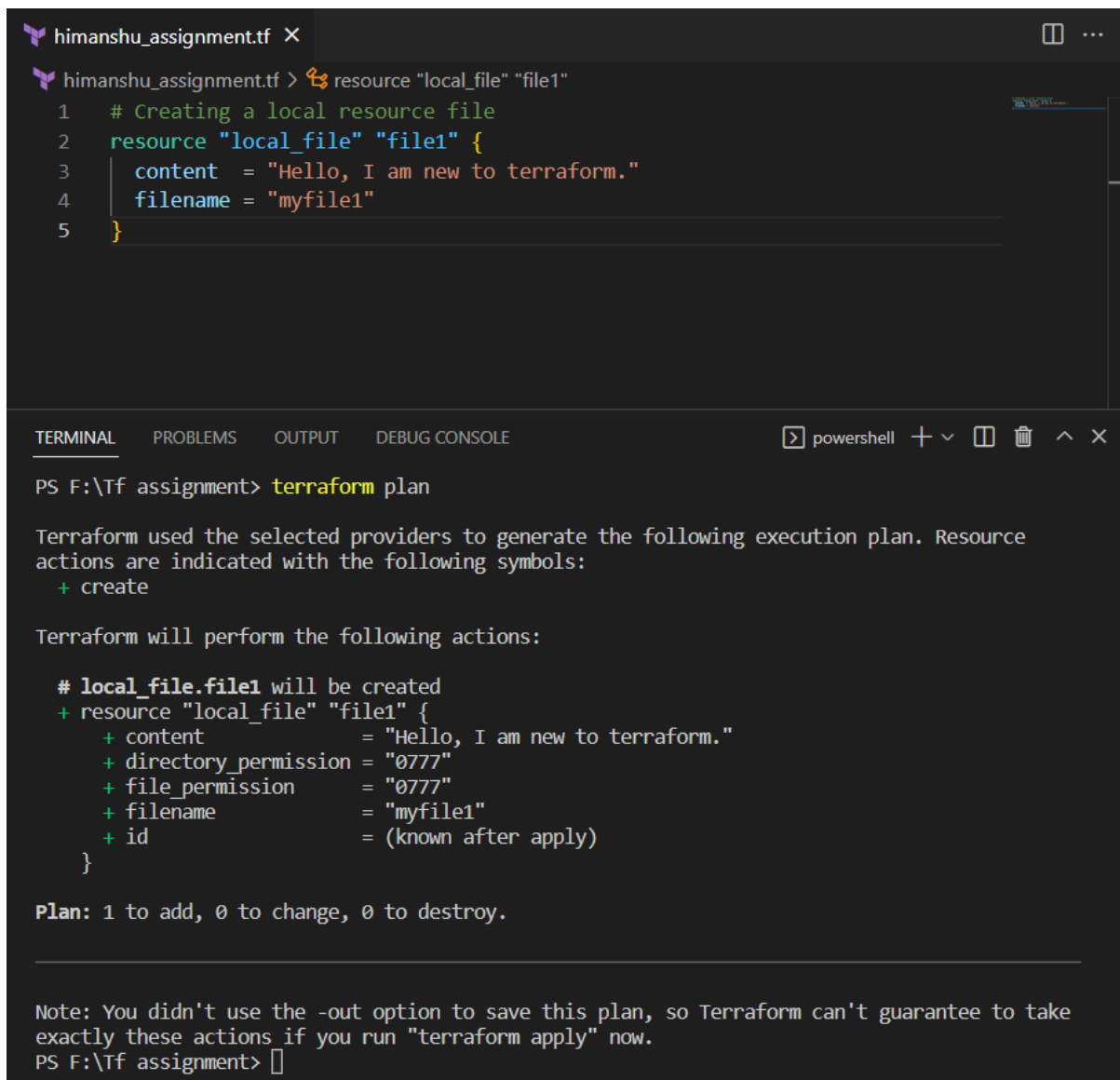
```
himanshu_assignment.tf X
himanshu_assignment.tf > resource "local_file" "file1" > filename
1 # Creating a local resource file
2 resource "local_file" "file1" {
3     content = "Hello, I am new to terraform."
4     filename = "myfile1"
5 }
```



```
himanshu_assignment.tf X
himanshu_assignment.tf > resource "local_file" "file1"
1 # Creating a local resource file
2 resource "local_file" "file1" {
3     content = "Hello, I am new to terraform."
4     filename = "myfile1"
5 }

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE powershell + v []
PS F:\Tf assignment> terraform fmt
himanshu_assignment.tf
PS F:\Tf assignment>
```

Terraform plan



The image shows a VS Code editor window with a file named `himanshu_assignment.tf`. The file contains a Terraform configuration for creating a local file resource. Below the editor, a terminal window shows the output of the `terraform plan` command.

```
himanshu_assignment.tf X
himanshu_assignment.tf > resource "local_file" "file1"
1  # Creating a local resource file
2  resource "local_file" "file1" {
3      content = "Hello, I am new to terraform."
4      filename = "myfile1"
5  }
```

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE powershell + -

```
PS F:\Tf assignment> terraform plan

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

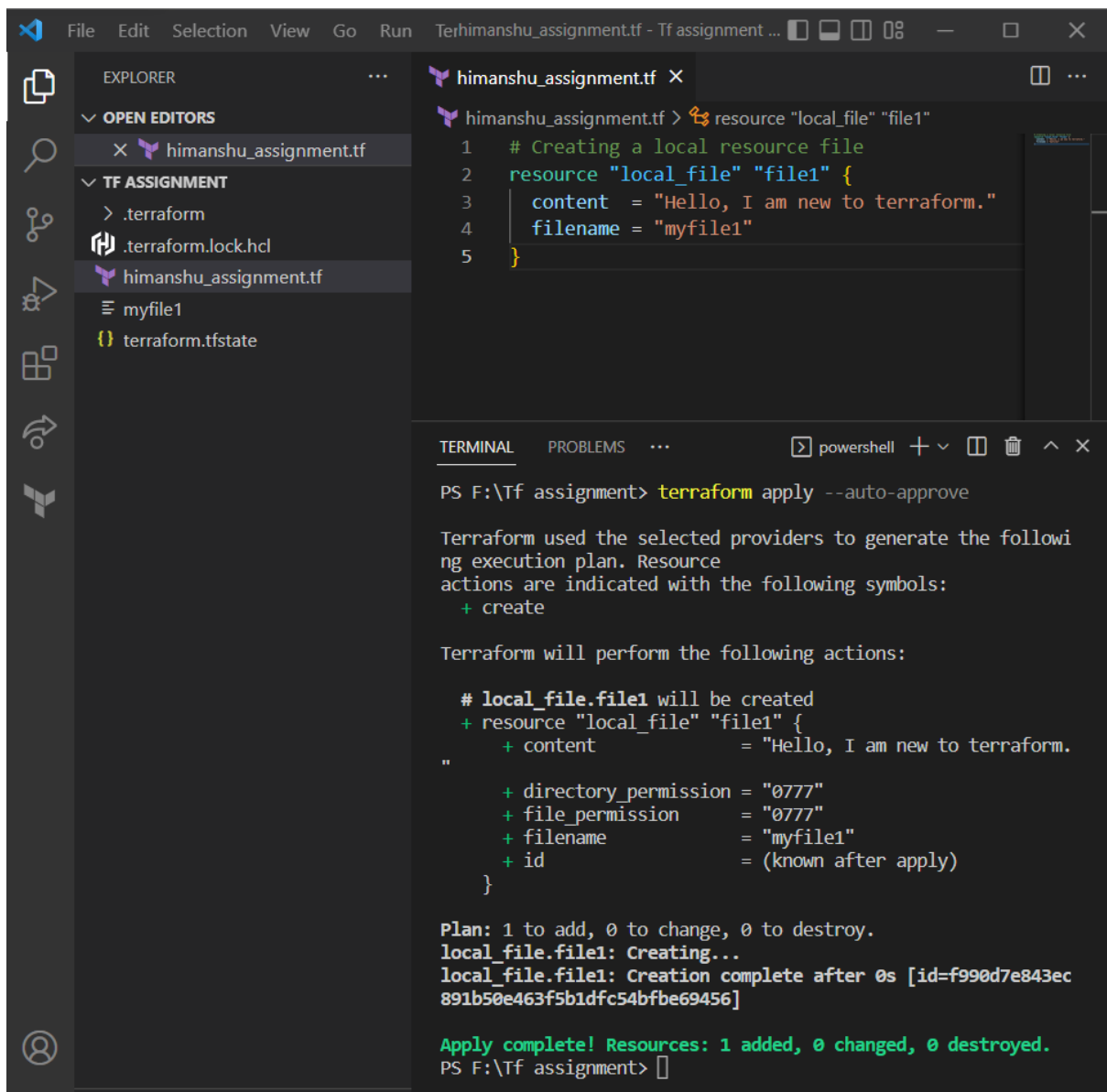
Terraform will perform the following actions:

# local_file.file1 will be created
+ resource "local_file" "file1" {
  + content             = "Hello, I am new to terraform."
  + directory_permission = "0777"
  + file_permission     = "0777"
  + filename            = "myfile1"
  + id                  = (known after apply)
}

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

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

Terraform apply and auto-approve



The screenshot shows the Visual Studio Code interface with a Terraform configuration file named `himanshu_assignment.tf` open in the editor. The file contains a `resource "local_file" "file1"` block with the following configuration:

```
1 # Creating a local resource file
2 resource "local_file" "file1" {
3     content = "Hello, I am new to terraform."
4     filename = "myfile1"
5 }
```

The Explorer sidebar on the left shows the project structure with the following files:

- `himanshu_assignment.tf` (selected)
- `.terraform`
- `.terraform.lock.hcl`
- `myfile1`
- `terraform.tfstate`

The Terminal window at the bottom shows the output of the `terraform apply --auto-approve` command. The output indicates that the resource `local_file.file1` will be created and provides the following details:

```
PS F:\Tf assignment> terraform apply --auto-approve

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

Terraform will perform the following actions:

# local_file.file1 will be created
+ resource "local_file" "file1" {
+   content = "Hello, I am new to terraform."
+   directory_permission = "0777"
+   file_permission = "0777"
+   filename = "myfile1"
+   id = (known after apply)
}

Plan: 1 to add, 0 to change, 0 to destroy.
local_file.file1: Creating...
local_file.file1: Creation complete after 0s [id=f990d7e843ec891b50e463f5b1dfc54bfb69456]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS F:\Tf assignment>
```

Creating different file type

The screenshot shows the Visual Studio Code interface with a Terraform configuration file open in the editor and its execution plan output in the terminal.

EXPLORER

- OPEN EDITORS
 - himanshu_assignment.tf
- TF ASSIGNMENT
 - .terraform
 - .terraform.lock.hcl
 - himanshu_assignment.tf
 - myfile1
 - terraform.tfstate
 - terraform.tfstate.backup

himanshu_assignment.tf

```
resource "local_file" "file1" {
  filename = "himanshu.html"
  content  = "Hello, I am new to terraform."
}
```

TERMINAL

```
PS F:\Tf assignment> terraform plan
local_file.file1: Refreshing state... [id=f990d7e843ec891b50e463f5b1dfc54bfbe69456]

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.file1 must be replaced
-/+ resource "local_file" "file1" {
  ~ filename           = "myfile1" -> "himanshu.html" #
  forces replacement
  ~ id                 = "f990d7e843ec891b50e463f5b1dfc54bfbe69456" -> (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.

```
PS F:\Tf assignment>
```

EXPLORER

...

OPEN EDITORS

himanshu_assignment.tf

TF ASSIGNMENT

.terraform

.terraform.lock.hcl

himanshu_assignment.tf

himanshu.html

terraform.tfstate

terraform.tfstate.backup

himanshu_assignment.tf

resource "local_file" "file1"

Creating a local resource file

resource "local_file" "file1" {

content = "Hello, I am new to terraform."

filename = "himanshu.html"

}

TERMINAL

PROBLEMS

...

powershell

+

^

✕

local_file.file1: Refreshing state... [id=f990d7e843ec891b50e463f5b1dfc54bfbe69456]

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.file1 must be replaced

-/+ resource "local_file" "file1" {

~ filename = "myfile1" -> "himanshu.html" # forces replacement

~ id = "f990d7e843ec891b50e463f5b1dfc54bfbe69456" -> (known after apply)

(3 unchanged attributes hidden)

}

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

local_file.file1: Destroying... [id=f990d7e843ec891b50e463f5b1dfc54bfbe69456]

local_file.file1: Destruction complete after 0s

local_file.file1: Creating...

local_file.file1: Creation complete after 0s [id=f990d7e843ec891b50e463f5b1dfc54bfbe69456]

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

PS F:\Tf assignment>

Changing the content

Old content

```
himanshu_assignment.tf ×
himanshu_assignment.tf > resource "local_file" "file1"
1  # Creating a local resource file
2  resource "local_file" "file1" {
3      content = "Hello, I am new to terraform."
4      filename = "himanshu.html"
5  }
```

New content

```

himanshu_assignment.tf
himanshu_assignment.tf > resource "local_file" "file1" > content
1  # Creating a local resource file
2  resource "local_file" "file1" {
3      content = "Hello everyone, I am new to terraform."
4      filename = "himanshu.html"
5  }

```

Doing terraform plan

```
himanshu_assignment.tf X
himanshu_assignment.tf > resource "local_file" "file1"
1 # Creating a local resource file
2 resource "local_file" "file1" {
3     content = "Hello everyone, I am new to terraform."
4     filename = "himanshu.html"
5 }

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE powershell + - X
PS F:\Tf assignment> terraform plan
local_file.file1: Refreshing state... [id=f990d7e843ec891b50e463f5b1dfc54bfbe69456]

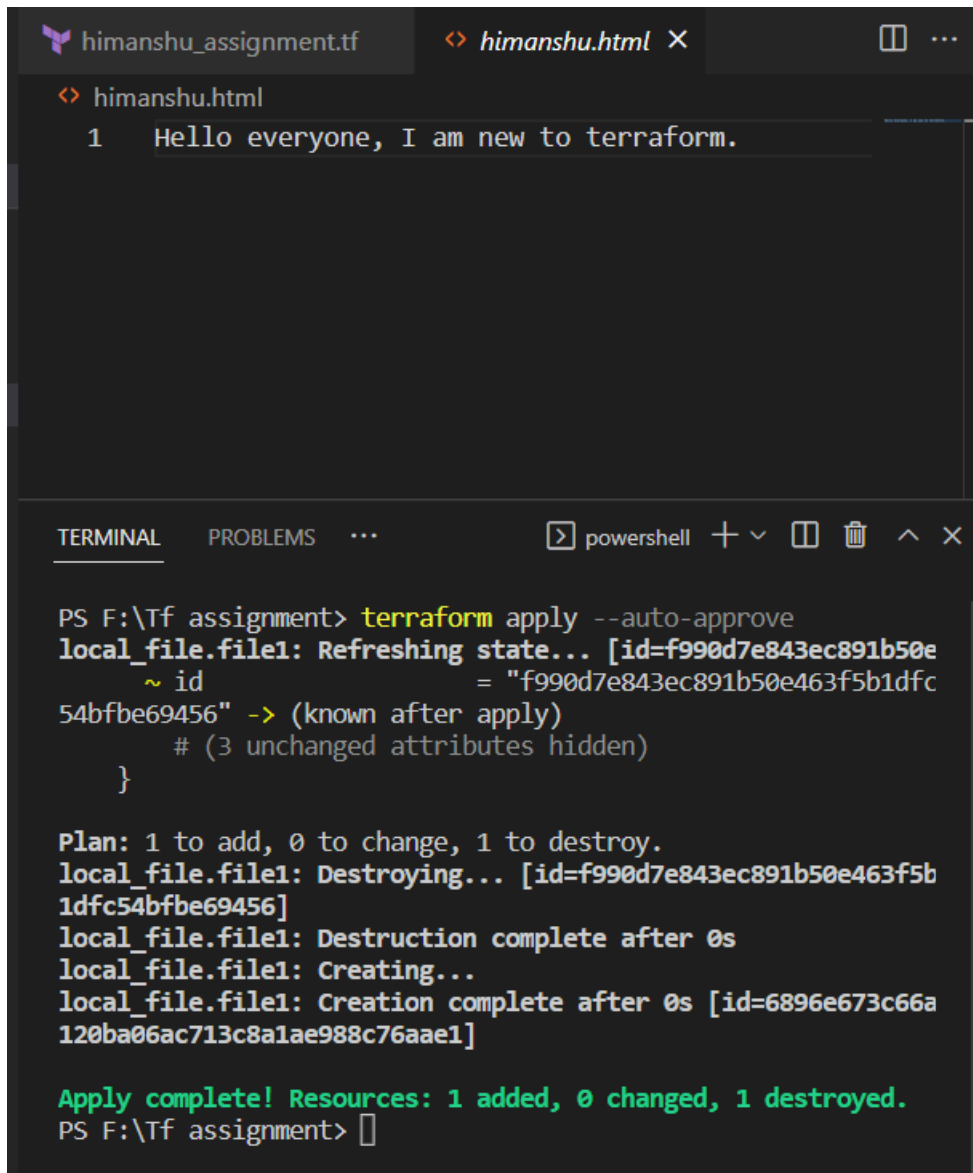
Terraform used the selected providers to generate the following execution plan. Resource
actions are indicated with the following symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

# local_file.file1 must be replaced
-/+ resource "local_file" "file1" {
    ~ content = "Hello, I am new to terraform." -> "Hello everyone, I am new to
terraform." # forces replacement
    ~ id = "f990d7e843ec891b50e463f5b1dfc54bfbe69456" -> (known after appl
y)
    # (3 unchanged attributes hidden)
}

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

Terraform apply



The image shows a Visual Studio Code editor window with two tabs: `himanshu_assignment.tf` and `himanshu.html`. The `himanshu.html` tab is active, displaying a single line of text: `1 Hello everyone, I am new to terraform.`

Below the editor, the **TERMINAL** panel is open, showing the output of a Terraform `apply` command. The terminal text is as follows:

```
PS F:\Tf assignment> terraform apply --auto-approve
local_file.file1: Refreshing state... [id=f990d7e843ec891b50e
~ id = "f990d7e843ec891b50e463f5b1dfc
54bfbe69456" -> (known after apply)
# (3 unchanged attributes hidden)
}

Plan: 1 to add, 0 to change, 1 to destroy.
local_file.file1: Destroying... [id=f990d7e843ec891b50e463f5b
1dfc54bfbe69456]
local_file.file1: Destruction complete after 0s
local_file.file1: Creating...
local_file.file1: Creation complete after 0s [id=6896e673c66a
120ba06ac713c8a1ae988c76aae1]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
PS F:\Tf assignment>
```

Local values

Declaring a local value

```
himanshu_assignment.tf > ...
6
7 # Creating a local resource file
8 resource "local_file" "file2" {
9     content = local.content
10    filename = local.filename
11 }
12
13
14 # Usage of local values
15 # Declaring a local value
16 locals {
17     filename = "newfile.py"
18     content = "python file"
19 }
```

After terraform plan and terraform apply -auto-approve, himanshu2.html file will be replaced with the new file that is newfile.py.

The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows the file structure of a Terraform project, including `himanshu_assignment.tf` and `himanshu.html`. The main editor displays the content of `himanshu_assignment.tf`, which defines a `local_file` resource named `file2` using `local` values for `content` and `filename`. The `locals` block defines `filename` as `"newfile.py"` and `content` as `"python file"`.

The Terminal pane at the bottom shows the output of a Terraform plan and apply command. It indicates that the `local_file.file2` resource must be replaced because its `content` attribute has changed from `"Hello, I am new to terraform."` to `"python file"`. The plan shows that the resource will be destroyed and then recreated. The apply command successfully replaces the resource, creating a new file named `newfile.py` instead of `himanshu2.html`.

```
# local_file.file2 must be replaced
-/+ resource "local_file" "file2" {
  ~ content      = "Hello, I am new to terraform."
  ~ filename     = "himanshu2.html" -> "newfile.py" # forces replacement
  ~ id          = "f990d7e843ec891b50e463f5b1dfc54bfb69456" -> (known after apply)
  # (2 unchanged attributes hidden)
}

Plan: 1 to add, 0 to change, 1 to destroy.
local_file.file2: Destroying... [id=f990d7e843ec891b50e463f5b1dfc54bfb69456]
local_file.file2: Destruction complete after 0s
local_file.file2: Creating...
local_file.file2: Creation complete after 0s [id=fd1b5825bcc12943b432217d6851801431a7cf0]

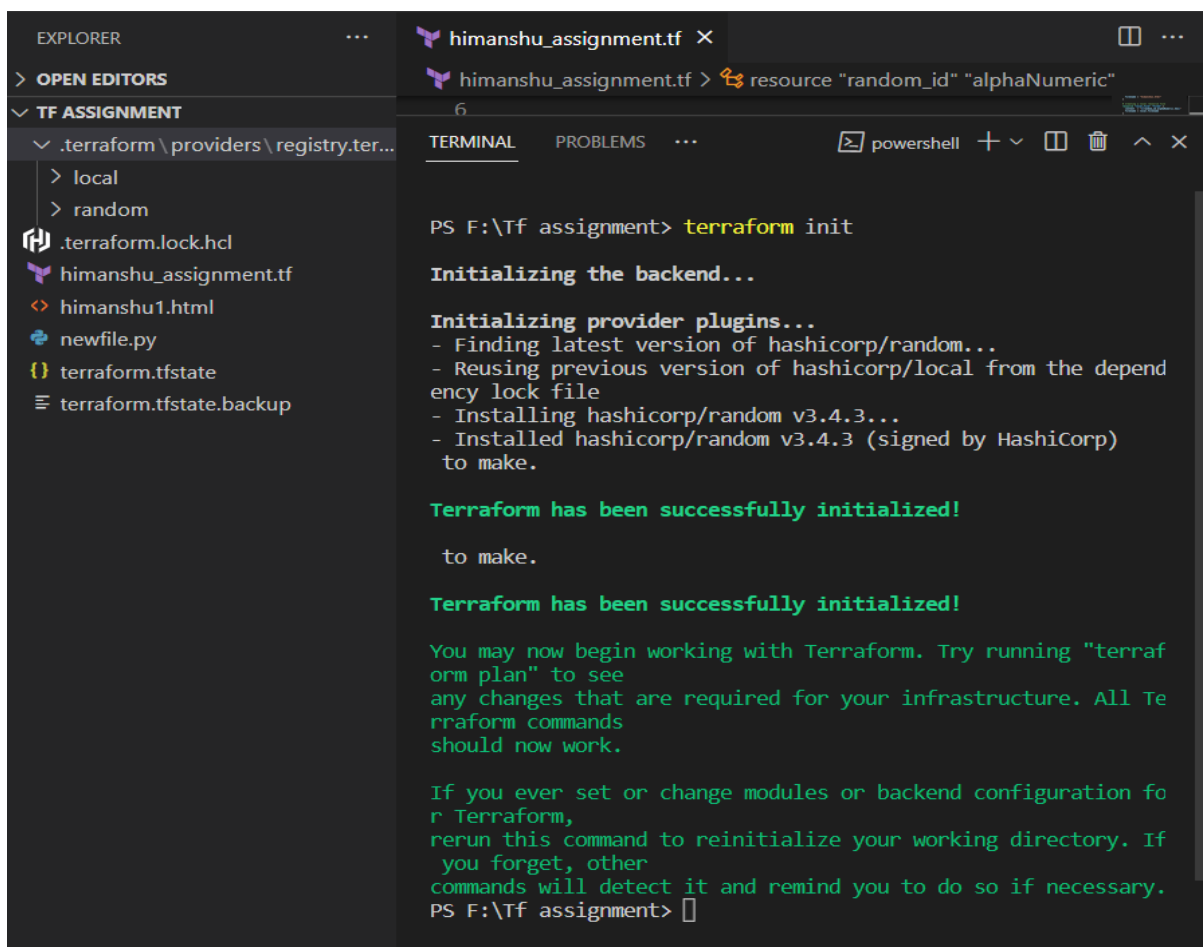
Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
PS F:\Tf assignment>
```

Terraform random to generate random string

```
himanshu_assignment.tf X
himanshu_assignment.tf > resource "local_file" "file2"

6
7 # Creating a local resource file
8 resource "local_file" "file2" {
9     content = "${random_id.alphaNumeric.hex}"
10    filename = local.filename
11 }
12
13 # Usage of local values
14 # Declaring a local value
15 locals {
16     filename = "newfile.py"
17     content = "python file"
18 }
19
20 # terraform random
21 resource "random_id" "alphaNumeric" {
22     byte_length = 8
23 }
24 }
```

Now terraform init as we are creating a new resource.



The screenshot shows the Visual Studio Code interface with the Explorer, Open Editors, and Terminal panels. The Explorer panel shows the file structure of the 'TF ASSIGNMENT' directory, including '.terraform\providers\registry.ter...', 'local', 'random', '.terraform.lock.hcl', 'himanshu_assignment.tf', 'himanshu1.html', 'newfile.py', 'terraform.tfstate', and 'terraform.tfstate.backup'. The Open Editors panel shows the 'himanshu_assignment.tf' file with the 'resource "random_id" "alphaNumeric"' block selected. The Terminal panel shows the execution of the 'terraform init' command in a PowerShell window, displaying the following output:

```
PS F:\Tf assignment> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/random...
- Reusing previous version of hashicorp/local from the dependency lock file
- Installing hashicorp/random v3.4.3...
- Installed hashicorp/random v3.4.3 (signed by HashiCorp) to make.

Terraform has been successfully initialized!

to make.

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.
PS F:\Tf assignment>
```

Terraform plan

```
TERMINAL  PROBLEMS  ...  powershell  + v  [ ]  [ ]  v  x

PS F:\Tf assignment> terraform plan
local_file.file1: Refreshing state... [id=6896e673c66a120ba06ac713c8a1ae988c76aae1]
local_file.file2: Refreshing state... [id=fd1b5825bcca12943b432217d6851801431a7cf0]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create
-/+ destroy and then create replacement

Terraform will perform the following actions:

# local_file.file2 must be replaced
-/+ resource "local_file" "file2" {
  ~ content          = "python file" -> (known after apply) # forces replacement
  ~ id               = "fd1b5825bcca12943b432217d6851801431a7cf0" -> (known after apply)
    # (3 unchanged attributes hidden)
}

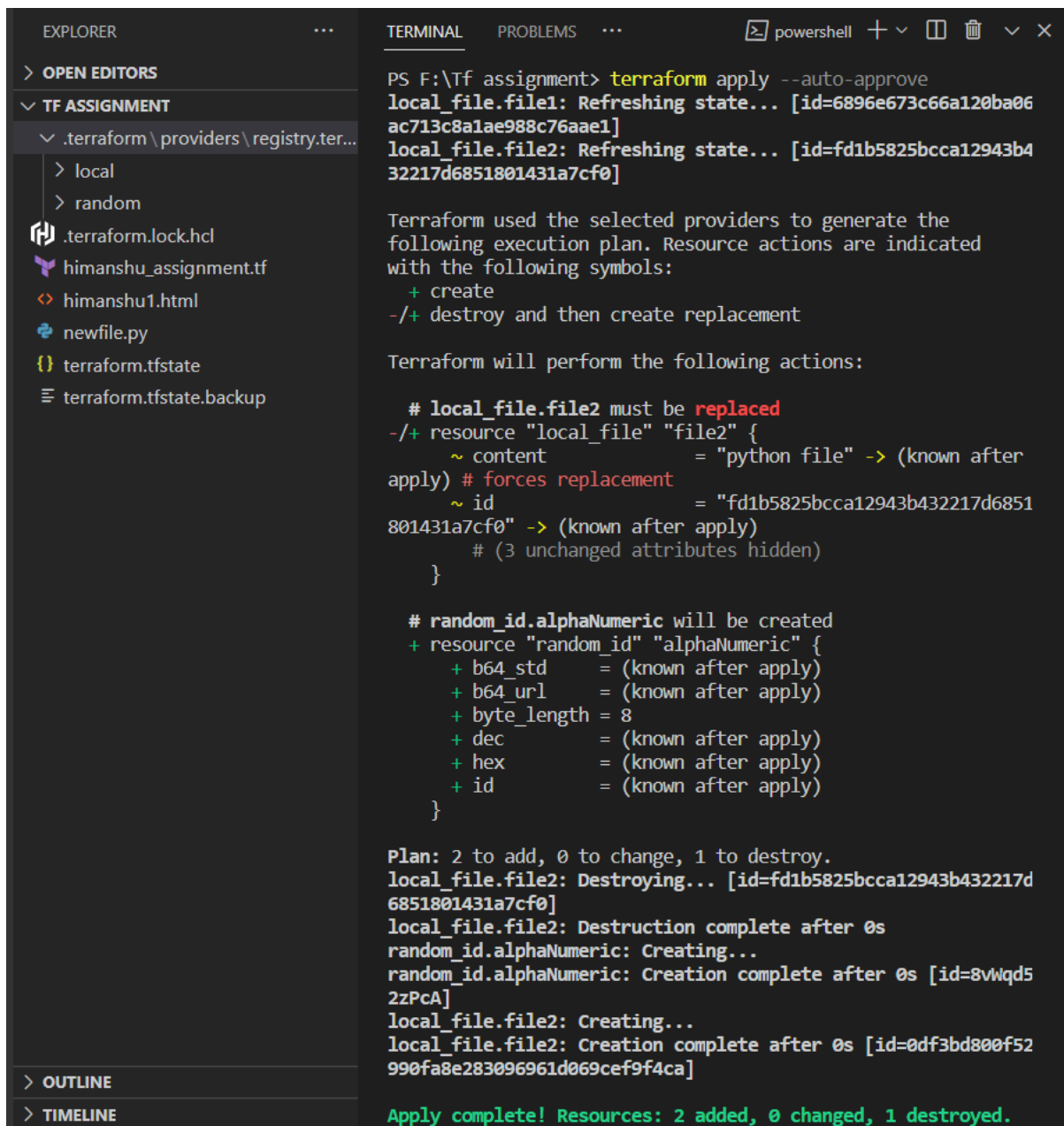
# random_id.alphaNumeric will be created
+ resource "random_id" "alphaNumeric" {
  + b64_std          = (known after apply)
  + b64_url          = (known after apply)
  + byte_length      = 8
  + dec              = (known after apply)
  + hex              = (known after apply)
  + id               = (known after apply)
}

Plan: 2 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.
PS F:\Tf assignment> [ ]
```

Terraform apply --auto-approve



The screenshot shows the Visual Studio Code interface with the Explorer, Explorer, and Terminal panels. The Explorer panel on the left shows the file structure of a project named 'TF ASSIGNMENT', including files like '.terraform.lock.hcl', 'himanshu_assignment.tf', 'himanshu1.html', 'newfile.py', 'terraform.tfstate', and 'terraform.tfstate.backup'. The Terminal panel on the right shows the output of the command 'terraform apply --auto-approve' executed in a PowerShell window. The output indicates that the state is being refreshed for 'local_file.file1' and 'local_file.file2'. It then shows the execution plan, which includes creating a new 'random_id.alphaNumeric' resource and replacing 'local_file.file2'. The plan is applied successfully, resulting in 2 resources added, 0 changed, and 1 destroyed.

```
PS F:\Tf assignment> terraform apply --auto-approve
local_file.file1: Refreshing state... [id=6896e673c66a120ba06ac713c8a1ae988c76aae1]
local_file.file2: Refreshing state... [id=fd1b5825bcca12943b432217d6851801431a7cf0]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create
-/+ destroy and then create replacement

Terraform will perform the following actions:

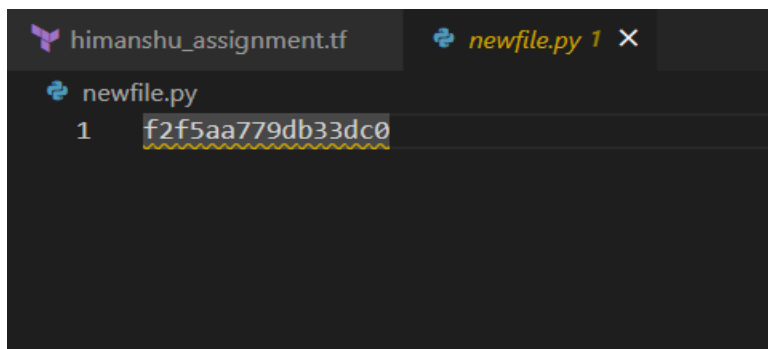
# local_file.file2 must be replaced
-/+ resource "local_file" "file2" {
  ~ content      = "python file" -> (known after apply) # forces replacement
  ~ id           = "fd1b5825bcca12943b432217d6851801431a7cf0" -> (known after apply)
    # (3 unchanged attributes hidden)
}

# random_id.alphaNumeric will be created
+ resource "random_id" "alphaNumeric" {
  + b64_std      = (known after apply)
  + b64_url      = (known after apply)
  + byte_length  = 8
  + dec          = (known after apply)
  + hex          = (known after apply)
  + id           = (known after apply)
}

Plan: 2 to add, 0 to change, 1 to destroy.
local_file.file2: Destroying... [id=fd1b5825bcca12943b432217d6851801431a7cf0]
local_file.file2: Destruction complete after 0s
random_id.alphaNumeric: Creating...
random_id.alphaNumeric: Creation complete after 0s [id=8vWqd52zPcA]
local_file.file2: Creating...
local_file.file2: Creation complete after 0s [id=0df3bd800f52990fa8e283096961d069cef9f4ca]

Apply complete! Resources: 2 added, 0 changed, 1 destroyed.
```

A random string is generated in the file.



The screenshot shows the Visual Studio Code interface with the Explorer panel on the left and the Editor panel on the right. The Explorer panel shows the file structure of a project named 'TF ASSIGNMENT', including files like '.terraform.lock.hcl', 'himanshu_assignment.tf', 'himanshu1.html', 'newfile.py', 'terraform.tfstate', and 'terraform.tfstate.backup'. The Editor panel shows the content of the file 'newfile.py', which contains a single line of text: 'f2f5aa779db33dc0'.

```
himanshu_assignment.tf
newfile.py 1 X
1 f2f5aa779db33dc0
```

Terraform variables

```
26
27 # Creating variables
28
29 variable "var-filename" {
30     type = string
31     description = "Enter data"
32     default = "Content for the file"
33 }
34
```

Using a variable

```
himanshu_assignment.tf > ...
1  # Creating a local resource file
2  # dec is used to obtain decimal values
3  resource "local_file" "file1" {
4      content = "${random_id.alphaNumeric.dec}"
5      filename = var.var-filename
6  }
```

Terraform plan

```

TERMINAL      PROBLEMS      OUTPUT      DEBUG CONSOLE
powershell + - [ ] [X] ^ x

PS F:\Tf assignment> terraform plan
random_id.alphaNumeric: Refreshing state... [id=8vWqd52zPcA]
local_file.file1: Refreshing state... [id=c13147f1065f49f55ad3e8b4ef1c2868a041edaa]
local_file.file2: Refreshing state... [id=0df3bd800f52990fa8e283096961d069cef9f4ca]

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.file1 must be replaced
-/+ resource "local_file" "file1" {
    ~ filename           = "himanshu1.html" -> "def-file1.txt" # forces replacement
    ~ id                 = "c13147f1065f49f55ad3e8b4ef1c2868a041edaa" -> (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.

```
PS F:\Tf assignment>
```

Terraform apply

The screenshot shows the Visual Studio Code interface with the Explorer, Explorer, and Terminal panels. The Explorer panel on the left shows the file structure of a Terraform project, including a `himanshu_assignment.tf` file. The Explorer panel on the right shows the content of `himanshu_assignment.tf`, which defines a variable `var-filename` with a type of `string`, a description of `"Enter data"`, and a default value of `"def-file1.txt"`. The Terminal panel at the bottom shows the output of the `terraform apply --auto-approve` command. The output indicates that the state is being refreshed for `random_id.alphaNumeric` and `local_file.file2`. It then shows the execution plan, which indicates that the `local_file.file1` resource must be replaced. The plan shows that the `local_file.file1` resource is being destroyed and then recreated with the new filename `def-file1.txt`. The apply command completes successfully, showing that the resources are added, changed, or destroyed as planned.

```
EXPLORER
> OPEN EDITORS
TF ASSIGNMENT
  .terraform\providers\registry.ter...
    > local
    > random
  .terraform.lock.hcl
  def-file1.txt
  himanshu_assignment.tf
  newfile.py
  terraform.tfstate
  terraform.tfstate.backup

himanshu_assignment.tf
27 # Creating variables
28
29 variable "var-filename" {
30   type = string
31   description = "Enter data"
32   default = "def-file1.txt"
33 }
34

TERMINAL
powershell
PS F:\Tf assignment> terraform apply --auto-approve
random_id.alphaNumeric: Refreshing state... [id=8VWqd52zPcA]
local_file.file2: Refreshing state... [id=0df3bd800f52990fa8e

Terraform used the selected providers to generate the followi
ng execution plan. Resource
actions are indicated with the following symbols:
-/+ destroy and then create replacement

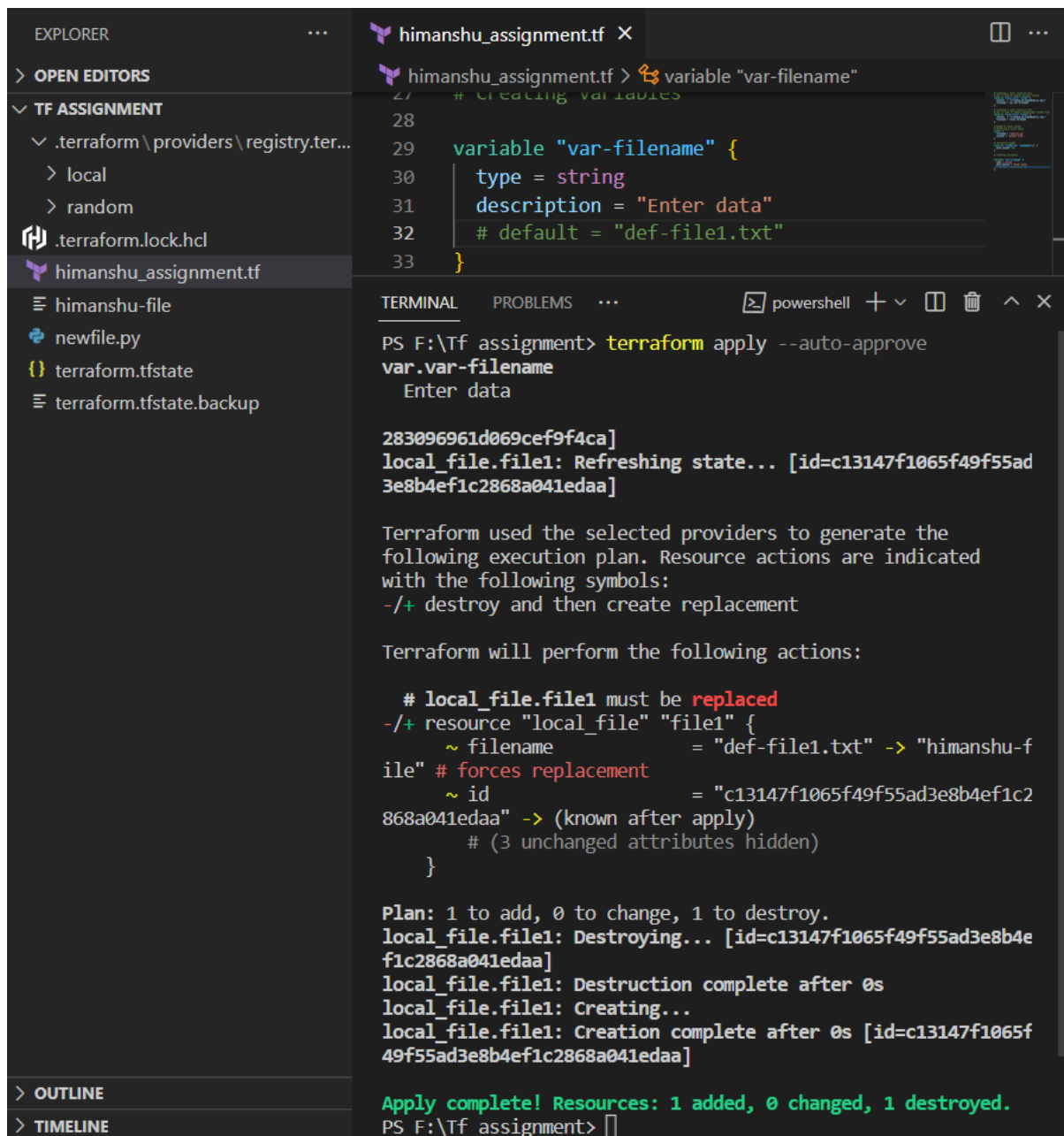
Terraform will perform the following actions:

  # local_file.file1 must be replaced
-/+ resource "local_file" "file1" {
  ~ filename           = "himanshu1.html" -> "def-file1
.txt" # forces replacement
  ~ id                 = "c13147f1065f49f55ad3e8b4ef1c2
868a041edaa" -> (known after apply)
    # (3 unchanged attributes hidden)
}

Plan: 1 to add, 0 to change, 1 to destroy.
local_file.file1: Destroying... [id=c13147f1065f49f55ad3e8b4e
f1c2868a041edaa]
local_file.file1: Destruction complete after 0s
local_file.file1: Creating...
local_file.file1: Creation complete after 0s [id=c13147f1065f
49f55ad3e8b4ef1c2868a041edaa]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
PS F:\Tf assignment>
```

If we remove the default from variable and then run terraform apply, then it will ask the user to enter a filename.



The screenshot shows the Visual Studio Code interface with a Terraform configuration file named `himanshu_assignment.tf` open in the editor. The file defines a variable `var-filename` of type `string` with a description "Enter data" and a default value `"def-file1.txt"`. The Explorer sidebar on the left shows the project structure, including `.terraform\providers\registry.ter...`, `local`, `random`, `.terraform.lock.hcl`, and `himanshu_assignment.tf`. The Terminal panel at the bottom shows the output of the `terraform apply --auto-approve` command. The output indicates that the `local_file.file1` resource must be replaced because the `filename` attribute has changed from `"def-file1.txt"` to `"himanshu-file"`. The plan shows that the existing resource will be destroyed and a new one will be created with the same ID.

```
27 # Creating variables
28
29 variable "var-filename" {
30     type = string
31     description = "Enter data"
32     # default = "def-file1.txt"
33 }
```

```
PS F:\Tf assignment> terraform apply --auto-approve
var.var-filename
Enter data

283096961d069cef9f4ca]
local_file.file1: Refreshing state... [id=c13147f1065f49f55ad3e8b4ef1c2868a041edaa]

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.file1 must be replaced
-/+ resource "local_file" "file1" {
    ~ filename           = "def-file1.txt" -> "himanshu-f
ile" # forces replacement
    ~ id                 = "c13147f1065f49f55ad3e8b4ef1c2
868a041edaa" -> (known after apply)
    # (3 unchanged attributes hidden)
}

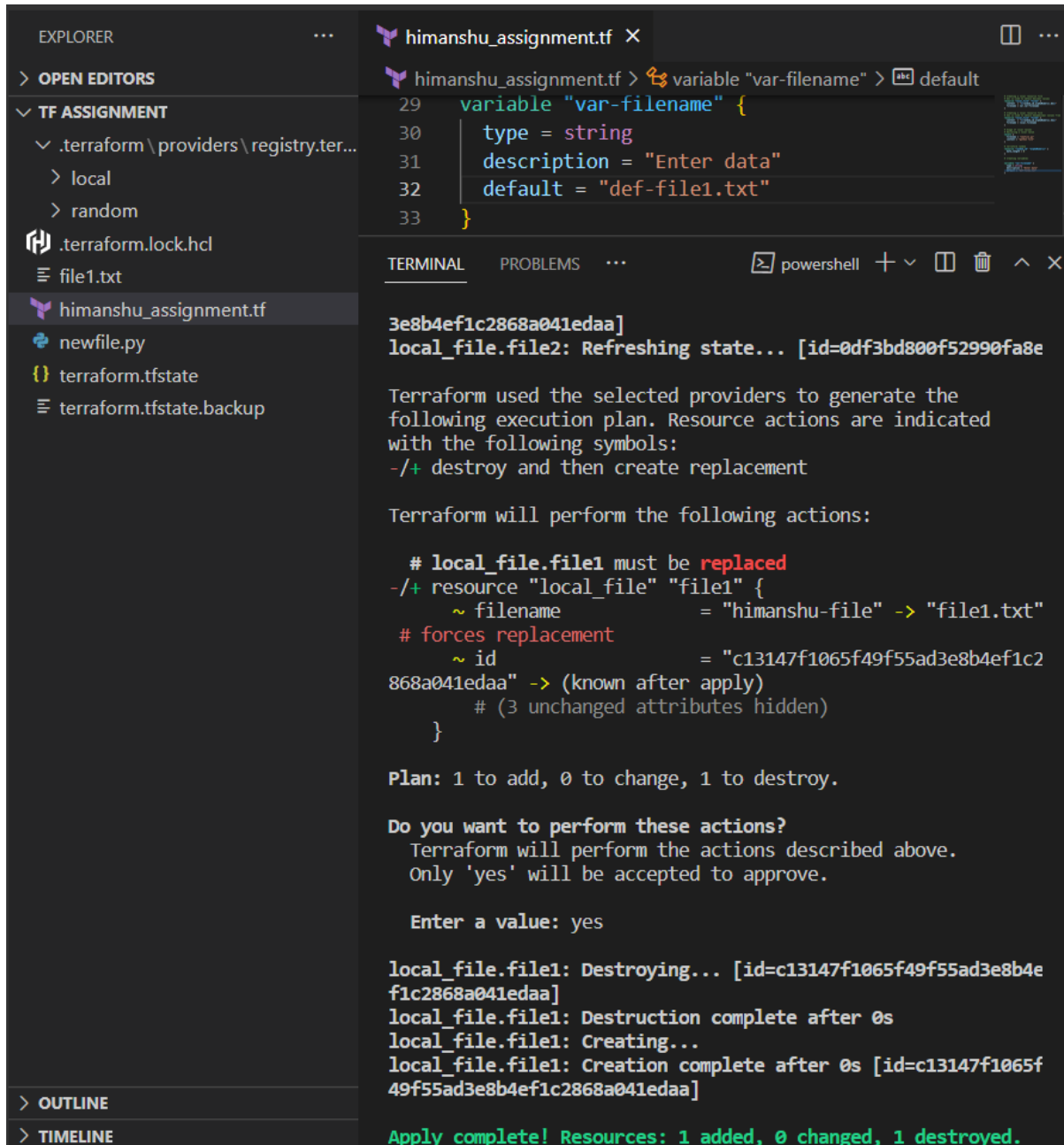
Plan: 1 to add, 0 to change, 1 to destroy.
local_file.file1: Destroying... [id=c13147f1065f49f55ad3e8b4ef1c2868a041edaa]
local_file.file1: Destruction complete after 0s
local_file.file1: Creating...
local_file.file1: Creation complete after 0s [id=c13147f1065f49f55ad3e8b4ef1c2868a041edaa]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
PS F:\Tf assignment>
```

If user wish to supply the filename even the default is present.

Syntax: Terraform apply -var variableName=filename.extension

Command: Terraform apply -var var-filename=file1.txt



The screenshot shows the Visual Studio Code interface. On the left, the Explorer pane shows a project structure with a folder named 'TF ASSIGNMENT' containing files like '.terraform.lock.hcl', 'file1.txt', and 'himanshu_assignment.tf'. The main editor displays the content of 'himanshu_assignment.tf', which defines a variable 'var-filename' of type 'string' with a default value of 'def-file1.txt'. The bottom panel shows the 'TERMINAL' output, which details the execution of 'terraform apply'. It shows that the resource 'local_file.file1' is being replaced because the 'filename' attribute has changed from 'himanshu-file' to 'file1.txt'. The terminal output includes the execution plan, a confirmation prompt 'Do you want to perform these actions?', and the final result: 'Apply complete! Resources: 1 added, 0 changed, 1 destroyed.'

```
EXPLORER
> OPEN EDITORS
TF ASSIGNMENT
  .terraform\providers\registry.ter...
    > local
    > random
  .terraform.lock.hcl
  file1.txt
  himanshu_assignment.tf
  newfile.py
  terraform.tfstate
  terraform.tfstate.backup

himanshu_assignment.tf
29 variable "var-filename" {
30     type = string
31     description = "Enter data"
32     default = "def-file1.txt"
33 }
```

```
3e8b4ef1c2868a041edaa]
local_file.file2: Refreshing state... [id=0df3bd800f52990fa8e

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.file1 must be replaced
-/+ resource "local_file" "file1" {
  ~ filename           = "himanshu-file" -> "file1.txt"
  # forces replacement
  ~ id                 = "c13147f1065f49f55ad3e8b4ef1c2
868a041edaa" -> (known after apply)
  # (3 unchanged attributes hidden)
}

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

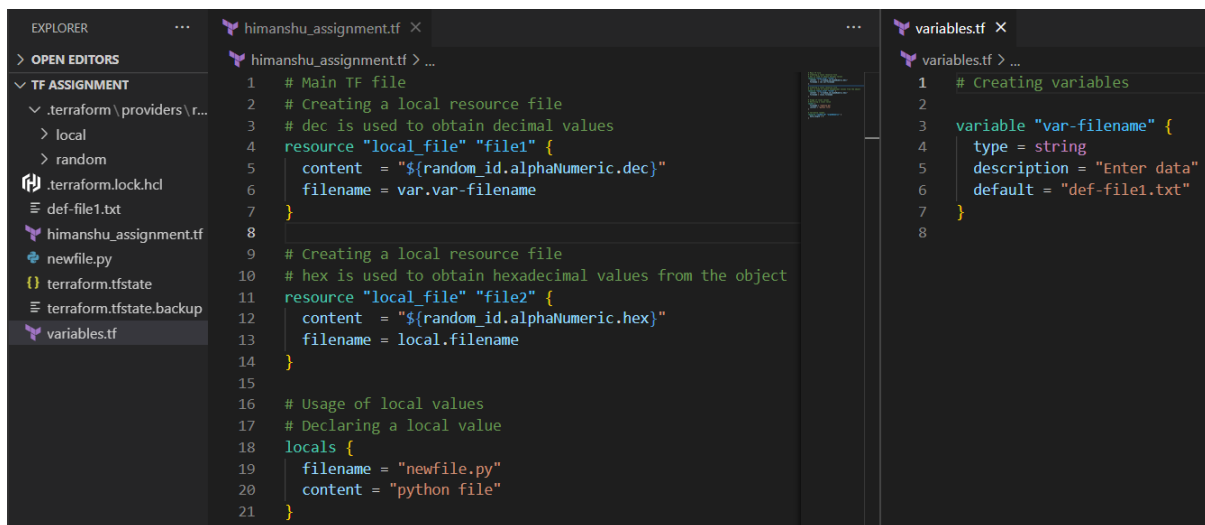
Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

local_file.file1: Destroying... [id=c13147f1065f49f55ad3e8b4e
f1c2868a041edaa]
local_file.file1: Destruction complete after 0s
local_file.file1: Creating...
local_file.file1: Creation complete after 0s [id=c13147f1065f
49f55ad3e8b4ef1c2868a041edaa]

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


Creating the variables externally in different terraform file

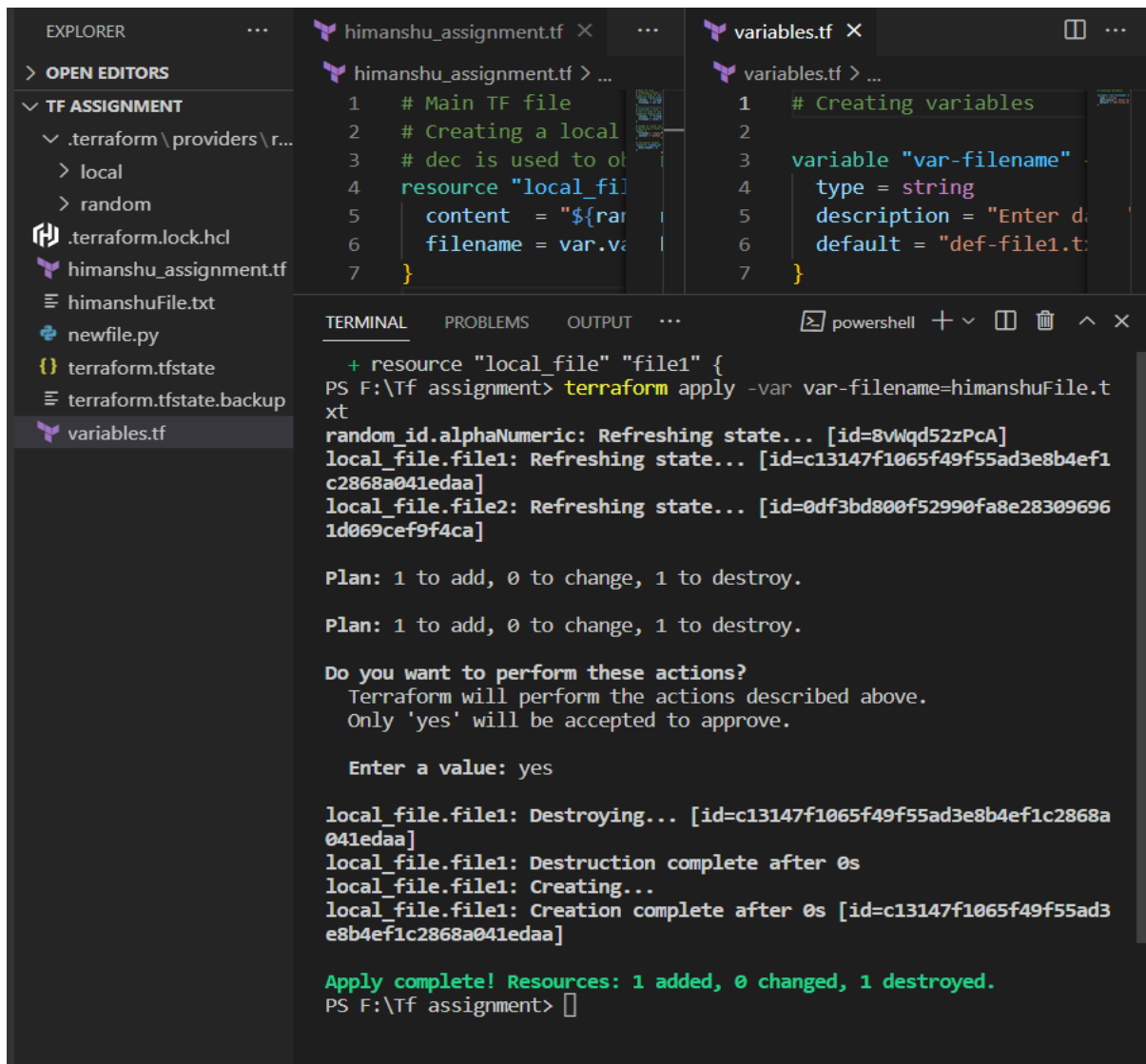


The screenshot shows the VS Code interface with two files open. The left file, `himanshu_assignment.tf`, contains Terraform code for creating two local files. The right file, `variables.tf`, defines a variable `var-filename`.

```
1 # Main TF file
2 # Creating a local resource file
3 # dec is used to obtain decimal values
4 resource "local_file" "file1" {
5   content = "${random_id.alphaNumeric.dec}"
6   filename = var.var-filename
7 }
8
9 # Creating a local resource file
10 # hex is used to obtain hexadecimal values from the object
11 resource "local_file" "file2" {
12   content = "${random_id.alphaNumeric.hex}"
13   filename = local.filename
14 }
15
16 # Usage of local values
17 # Declaring a local value
18 locals {
19   filename = "newfile.py"
20   content = "python file"
21 }
```

```
1 # Creating variables
2
3 variable "var-filename" {
4   type = string
5   description = "Enter data"
6   default = "def-file1.txt"
7 }
8
```

Now doing Terraform apply -var var-filename=himanshuFile.txt



The screenshot shows the VS Code interface with the same two files open. The terminal window at the bottom shows the execution of the `terraform apply` command with the variable `var-filename` set to `himanshuFile.txt`. The output shows the state of the resources and the plan.

```
PS F:\Tf assignment> terraform apply -var var-filename=himanshuFile.txt
random_id.alphaNumeric: Refreshing state... [id=8vWqd52zPCA]
local_file.file1: Refreshing state... [id=c13147f1065f49f55ad3e8b4ef1c2868a041edaa]
local_file.file2: Refreshing state... [id=0df3bd800f52990fa8e283096961d069cef9f4ca]

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

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.file1: Destroying... [id=c13147f1065f49f55ad3e8b4ef1c2868a041edaa]
local_file.file1: Destruction complete after 0s
local_file.file1: Creating...
local_file.file1: Creation complete after 0s [id=c13147f1065f49f55ad3e8b4ef1c2868a041edaa]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
PS F:\Tf assignment>
```

The filename changes from `def-file1.txt` to `himanshuFile.txt`.

Types of variables in terraform.

1. String variable

Creating a string variable

```
3  # String variable
4  variable "var-filename" {
5      type = string
6      description = "Enter data"
7      default = "def-file1.txt"
8  }
```

Accessing the string variable

```
17  # Declaring a local value
18  locals {
19      a = "string.py"
20      content = "python file"
21  }
22
23  # Accessing variables
24  # Accessing string variable
25  resource "local_file" "file4" {
26      content = var.var-filename
27      filename = local.a
28  }
```

After that executing terraform plan will show:

```
PS F:\Tf assignment> terraform plan
local_file.file3: Refreshing state... [id=c016f85c06c121220804903db934442950ee1c74]
```

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

```
+ create
- destroy
  + filename      = "string.py"
  + id            = (known after apply)
}

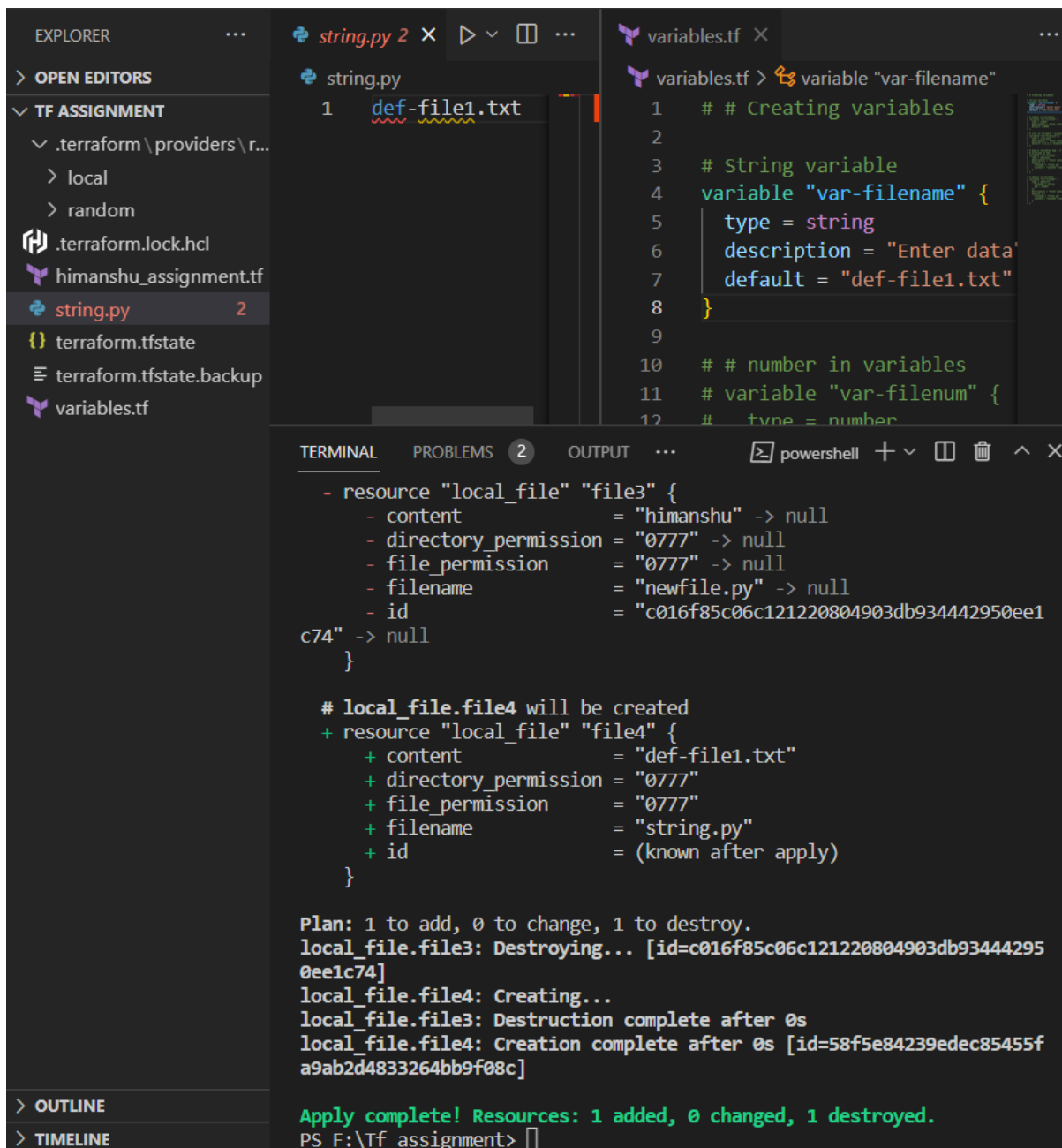
  + filename      = "string.py"
  + id            = (known after apply)
}
```

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.

```
PS F:\Tf assignment> 
```

Then terraform apply -auto-approve:



The screenshot shows a Visual Studio Code editor with two files open: `string.py` and `variables.tf`. The Explorer sidebar on the left shows the project structure under 'TF ASSIGNMENT', including `.terraform\providers\r...`, `local`, `random`, `.terraform.lock.hcl`, `himanshu_assignment.tf`, `string.py` (with a count of 2), `terraform.tfstate`, `terraform.tfstate.backup`, and `variables.tf`.

The `string.py` file contains the following code:

```
1 def-file1.txt
```

The `variables.tf` file contains the following code:

```
1 # # Creating variables
2
3 # String variable
4 variable "var-filename" {
5     type = string
6     description = "Enter data"
7     default = "def-file1.txt"
8 }
9
10 # # number in variables
11 # variable "var-filenameum" {
12 #     type = number
```

The Terminal at the bottom shows the output of a Terraform apply command, indicating that a resource is being destroyed and another is being created.

```
- resource "local_file" "file3" {
-   content          = "himanshu" -> null
-   directory_permission = "0777" -> null
-   file_permission    = "0777" -> null
-   filename          = "newfile.py" -> null
-   id                = "c016f85c06c121220804903db934442950ee1c74" -> null
- }

# local_file.file4 will be created
+ resource "local_file" "file4" {
+   content          = "def-file1.txt"
+   directory_permission = "0777"
+   file_permission    = "0777"
+   filename          = "string.py"
+   id                = (known after apply)
+ }

Plan: 1 to add, 0 to change, 1 to destroy.
local_file.file3: Destroying... [id=c016f85c06c121220804903db934442950ee1c74]
local_file.file4: Creating...
local_file.file3: Destruction complete after 0s
local_file.file4: Creation complete after 0s [id=58f5e84239edec85455fa9ab2d4833264bb9f08c]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
PS F:\Tf assignment>
```

2. Integer/ number variable.

Creating a number variable

```
# # number in variables
variable "var-filename" {
  type = number
  description = "Enter data"
  default = 0898
}
```

Accessing the number variable

```
22
23     locals {
24         b = "number.py"
25         content = "python file"
26     }
27
28     # Accessing number variable
29     resource "local_file" "file5" {
30         content = var.var-filename
31         filename = local.b
32     }
33
```

After that executing terraform plan will show:

```
PS F:\Tf assignment> terraform plan
local_file.file4: Refreshing state... [id=58f5e84239edec85455fa9ab2d4
833264bb9f08c]

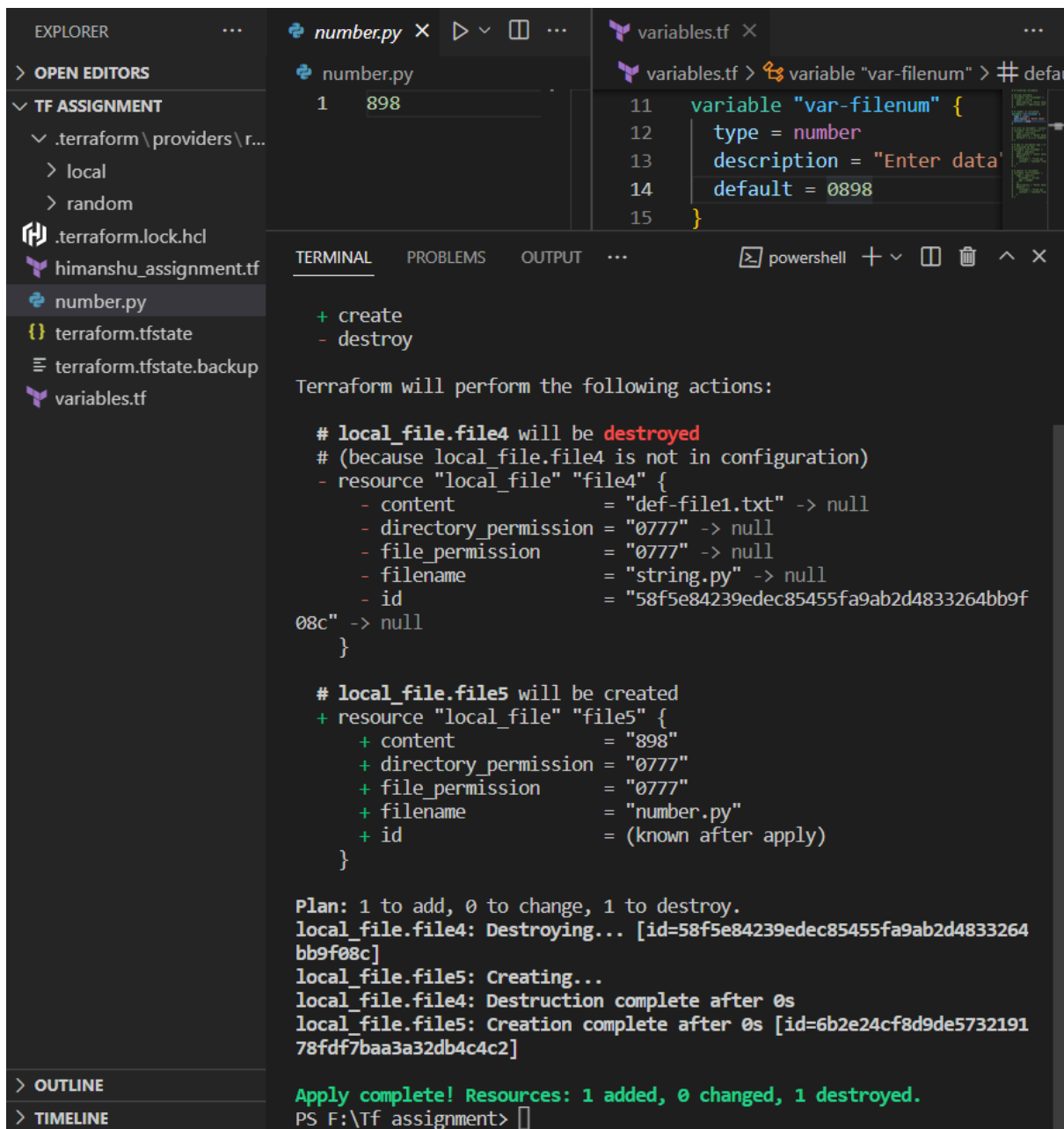
# local_file.file4 will be destroyed
# (because local_file.file4 is not in configuration)
- resource "local_file" "file4" {
  - content          = "def-file1.txt" -> null
  - directory_permission = "0777" -> null
  - file_permission    = "0777" -> null
  - filename          = "string.py" -> null
  - id                = "58f5e84239edec85455fa9ab2d4833264bb9f
08c" -> null
}

# local_file.file5 will be created
+ resource "local_file" "file5" {
  + content          = "898"
  + directory_permission = "0777"
  + file_permission    = "0777"
  + filename          = "number.py"
  + id                = (known after apply)
}

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 a
pply" now.
```

Then terraform apply -auto-approve:



The screenshot shows the Visual Studio Code interface with the following components:

- EXPLORER:** Shows the file structure with folders like `providers`, `local`, `random`, and files like `.terraform.lock.hcl`, `himanshu_assignment.tf`, `number.py`, `terraform.tfstate`, `terraform.tfstate.backup`, and `variables.tf`.
- number.py:** Contains the number `898`.
- variables.tf:** Contains a variable definition:

```
variable "var-filename" {  
  type = number  
  description = "Enter data"  
  default = 0898  
}
```
- TERMINAL:** Shows the output of the Terraform apply command.

```
+ create  
- destroy  
  
Terraform will perform the following actions:  
  
# local_file.file4 will be destroyed  
# (because local_file.file4 is not in configuration)  
- resource "local_file" "file4" {  
  - content          = "def-file1.txt" -> null  
  - directory_permission = "0777" -> null  
  - file_permission    = "0777" -> null  
  - filename          = "string.py" -> null  
  - id                = "58f5e84239edec85455fa9ab2d4833264bb9f08c" -> null  
}  
  
# local_file.file5 will be created  
+ resource "local_file" "file5" {  
  + content          = "898"  
  + directory_permission = "0777"  
  + file_permission    = "0777"  
  + filename          = "number.py"  
  + id                = (known after apply)  
}  
  
Plan: 1 to add, 0 to change, 1 to destroy.  
local_file.file4: Destroying... [id=58f5e84239edec85455fa9ab2d4833264bb9f08c]  
local_file.file5: Creating...  
local_file.file4: Destruction complete after 0s  
local_file.file5: Creation complete after 0s [id=6b2e24cf8d9de573219178fdf7baa3a32db4c4c2]  
  
Apply complete! Resources: 1 added, 0 changed, 1 destroyed.  
PS F:\Tf assignment>
```

3. List variable

Creating a list variable

```
# list in variable - accessed by indexing
variable "var-filelist" {
  type = list(any)
  description = "Enter data"
  default = [1,2,3,"himanshu","gupta"]
}
```

Accessing the list variable

```
# Usage of local values
# Declaring a local value
locals {
  filename = "newfile.py"
  content = "python file"
}

# Accessing list variable
resource "local_file" "file3" {
  content = var.var-filelist[3]
  filename = local.filename
}
```

After that executing terraform plan will show:

```
TERMINAL  PROBLEMS  1  OUTPUT  DEBUG CONSOLE

PS F:\Tf assignment> terraform plan
local_file.file2: Refreshing state... [id=0df3bd800f52990fa8e283096961d069cef9f4ca]
local_file.file1: Refreshing state... [id=c13147f1065f49f55ad3e8b4ef1c2868a041edaa]
random_id.alphaNumeric: Refreshing state... [id=8vWqd52zPcA]

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

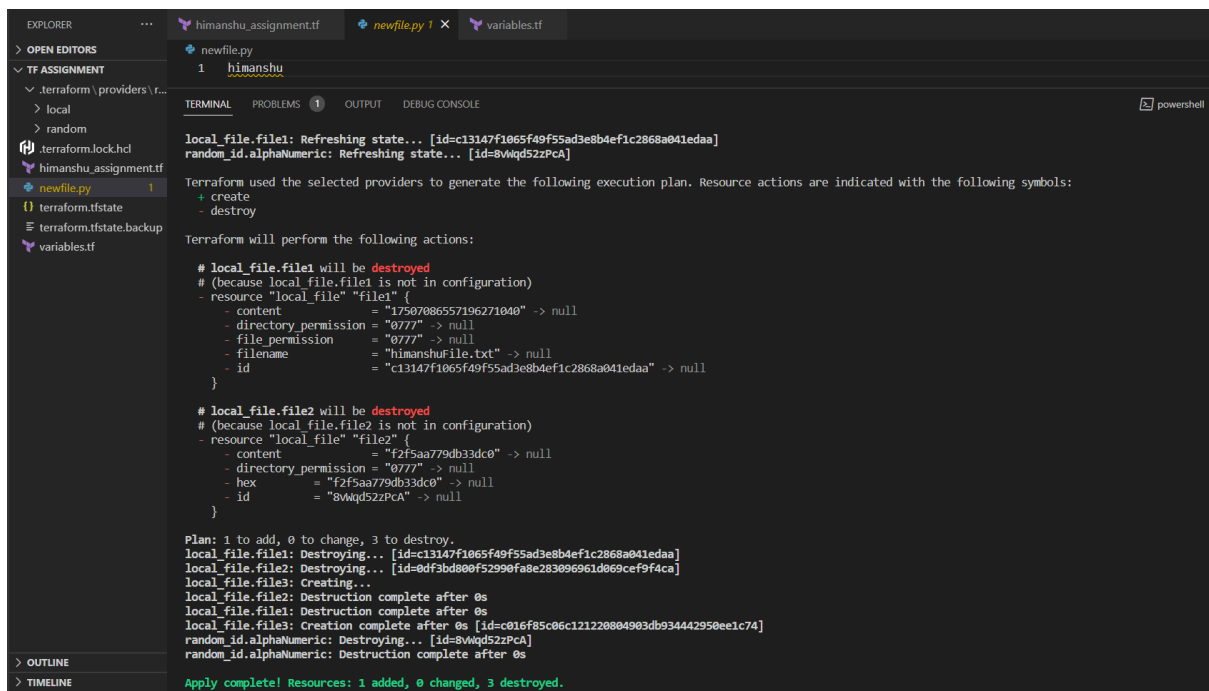
Terraform will perform the following actions:

# local_file.file1 will be destroyed
# (because local_file.file1 is not in configuration)
- resource "local_file" "file1" {
  - b64_std      = "8vWqd52zPcA=" -> null
  - b64_url      = "8vWqd52zPcA" -> null
  - byte_length = 8 -> null
  - dec          = "17507086557196271040" -> null
  - hex          = "f2f5aa779db33dc0" -> null
  - id           = "8vWqd52zPcA" -> null
}

Plan: 1 to add, 0 to change, 3 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.
PS F:\Tf assignment> terraform apply --auto-approve
```

Then terraform apply –auto-approve:



The screenshot shows a VS Code editor with a Terraform configuration file named `himanshu_assignment.tf` open. The configuration defines a `local_file` resource named `file1` and a `random_id` resource named `alphaNumeric`. The `local_file` resource is configured with a `content` of `"17507086557196271040"`, a `directory_permission` of `"0777"`, a `file_permission` of `"0777"`, a `filename` of `"himanshuFile.txt"`, and an `id` of `"c13147f1065f49f55ad3e8b4ef1c2868a041edaa"`. The `random_id` resource is configured with a `hex` of `"f2f5aa779db33dc0"` and an `id` of `"8VwQd52zPCA"`. The `local_file` resource is also configured with a `content` of `"f2f5aa779db33dc0"`, a `directory_permission` of `"0777"`, a `hex` of `"f2f5aa779db33dc0"`, and an `id` of `"8VwQd52zPCA"`.

The terminal output shows the execution plan for the configuration. The plan indicates that the `local_file` resource will be destroyed, the `random_id` resource will be destroyed, and the `local_file` resource will be created. The plan also shows the state of the resources before and after the apply operation.

```
local_file.file1: Refreshing state... [id=c13147f1065f49f55ad3e8b4ef1c2868a041edaa]
random_id.alphaNumeric: Refreshing state... [id=8VwQd52zPCA]

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

Terraform will perform the following actions:

# local_file.file1 will be destroyed
# (because local_file.file1 is not in configuration)
- resource "local_file" "file1" {
  - content           = "17507086557196271040" -> null
  - directory_permission = "0777" -> null
  - file_permission    = "0777" -> null
  - filename           = "himanshuFile.txt" -> null
  - id                 = "c13147f1065f49f55ad3e8b4ef1c2868a041edaa" -> null
}

# local_file.file2 will be destroyed
# (because local_file.file2 is not in configuration)
- resource "local_file" "file2" {
  - content           = "f2f5aa779db33dc0" -> null
  - directory_permission = "0777" -> null
  - hex               = "f2f5aa779db33dc0" -> null
  - id                = "8VwQd52zPCA" -> null
}

Plan: 1 to add, 0 to change, 3 to destroy.
local_file.file1: Destroying... [id=c13147f1065f49f55ad3e8b4ef1c2868a041edaa]
local_file.file2: Destroying... [id=0df3bd800f52990fa8e283096961d069cef9f4ca]
local_file.file3: Creating...
local_file.file2: Destruction complete after 0s
local_file.file1: Destruction complete after 0s
local_file.file3: Creation complete after 0s [id=c016f85c06c121228804983db934442950ee1c74]
random_id.alphaNumeric: Destroying... [id=8VwQd52zPCA]
random_id.alphaNumeric: Destruction complete after 0s

Apply complete! Resources: 1 added, 0 changed, 3 destroyed.
```

Creating a map variable

Creating a map variable

```
24 # map in variables map = {"key":"value"}
25 # accessed by key
26 variable "var-filemap" {
27     type = map(any)
28     description = "Enter data"
29     default = {
30         filename = "file1.py"
31         content = "Content from map variable"
32     }
33 }
34
```

Accessing the map variable

```

33  locals {
34      d = "map.py"
35      content = "python file"
36  }
37
38  # Accessing map variable
39  resource "local_file" "file6" {
40      content = var.var-filemap
41      filename = local.d
42  }
43

```

After that executing terraform plan will show:

```
himanshu_assignment.tf × ... variables.tf × ...
59 # content = var.var-filename
21 # default = ["1 2 3 "himanshu"]

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE powershell + - [ ] [X] [^] [v]

PS F:\Tf assignment> terraform plan
local_file.file5: Refreshing state... [id=6b2e24cf8d9de573219178fdf7baa3a32db4c4c2]

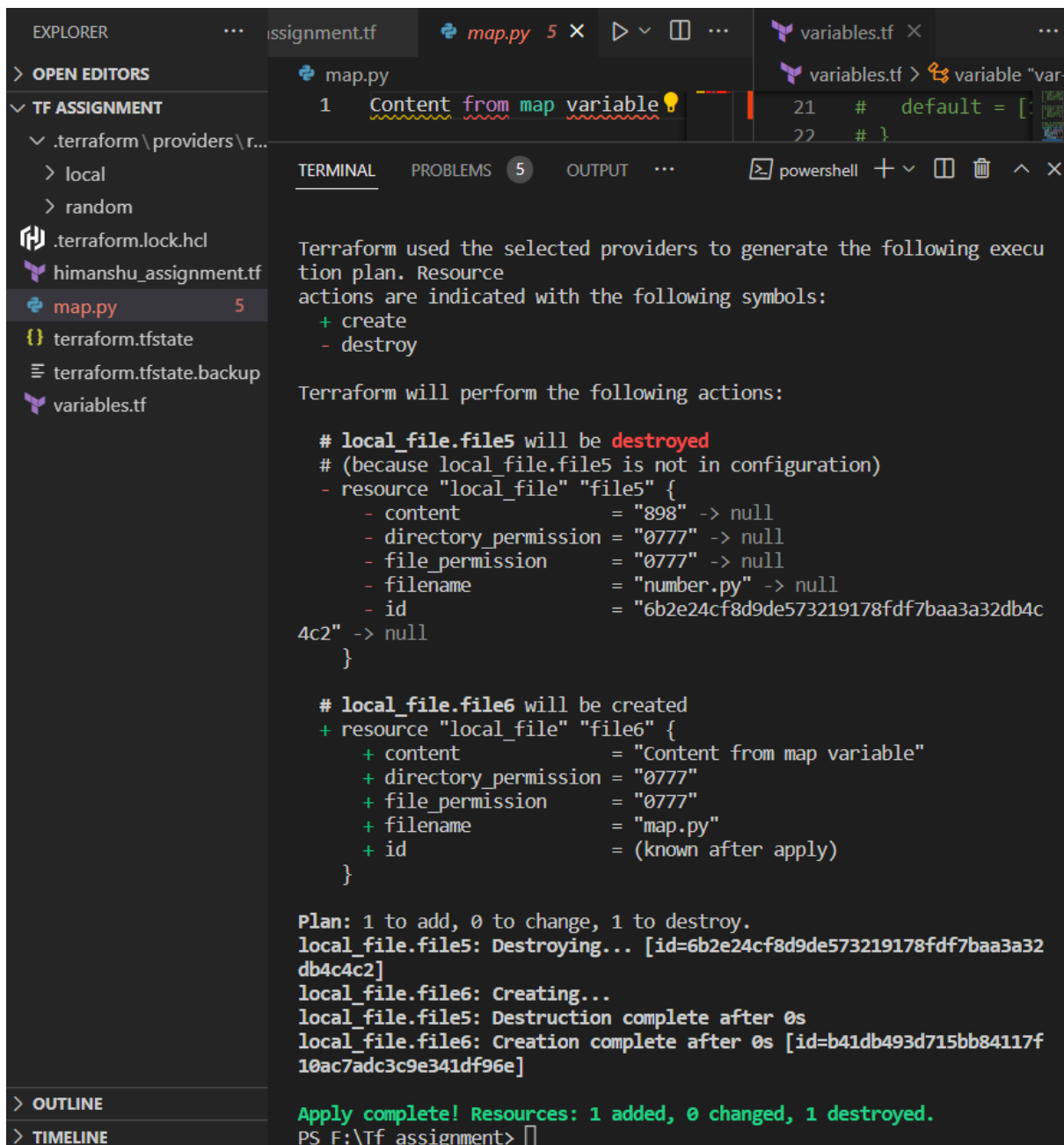
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create
- destroy

Terraform will perform the following actions:

# local_file.file5 will be destroyed
# (because local_file.file5 is not in configuration)
- resource "local_file" "file5" {
  - content           = "898" -> null
  - directory_permission = "0777" -> null
  - file_permission    = "0777" -> null
  - filename           = "number.py" -> null
  - id                 = "6b2e24cf8d9de573219178fdf7baa3a32db4c4c2" -> null
}

# local_file.file6 will be created
+ directory_permission = "0777"
+ file_permission      = "0777"
+ filename              = "map.py"
+ id                    = (known after apply)
}
```


Then terraform apply -auto-approve:



The screenshot shows the Visual Studio Code interface with a Terraform plan output in the terminal. The Explorer sidebar on the left shows the project structure, including files like `assignment.tf`, `map.py`, `variables.tf`, `terraform.lock.hcl`, `himanshu_assignment.tf`, `terraform.tfstate`, `terraform.tfstate.backup`, and `variables.tf`. The map.py file is selected and shows a single line of code: `1 Content from map variable`. The terminal window displays the Terraform plan output, which includes details about resource actions (destroying and creating) and the final state of the resources.

```
map.py
1 Content from map variable

TERMINAL
PROBLEMS 5 OUTPUT ... powershell + - + ^ x

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

Terraform will perform the following actions:

# local_file.file5 will be destroyed
# (because local_file.file5 is not in configuration)
- resource "local_file" "file5" {
  - content          = "898" -> null
  - directory_permission = "0777" -> null
  - file_permission   = "0777" -> null
  - filename          = "number.py" -> null
  - id                = "6b2e24cf8d9de573219178fdf7baa3a32db4c4c2" -> null
}

# local_file.file6 will be created
+ resource "local_file" "file6" {
  + content          = "Content from map variable"
  + directory_permission = "0777"
  + file_permission   = "0777"
  + filename          = "map.py"
  + id                = (known after apply)
}

Plan: 1 to add, 0 to change, 1 to destroy.
local_file.file5: Destroying... [id=6b2e24cf8d9de573219178fdf7baa3a32db4c4c2]
local_file.file6: Creating...
local_file.file5: Destruction complete after 0s
local_file.file6: Creation complete after 0s [id=b41db493d715bb84117f10ac7adc3c9e341df96e]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
PS F:\Tf assignment>
```

5. Object variable

Creating an object variable

```
35 # # object in variable
36 variable "var-fileobj" {
37     type = object({
38         filename = string
39         age = number
40     })
41     description = "Enter data"
42     default = {
43         age = 24
44         filename = "value"
45     }
46 }
```

Accessing the object variable

```
38 locals {
39     e = "object.py"
40     content = "python file"
41 }
42
43 # # Accessing object variable
44 resource "local_file" "file7" {
45     content = var.var-fileobj
46     filename = local.e
47 }
```

After that executing terraform plan will show:

```
TERMINAL  PROBLEMS  OUTPUT  DEBUG CONSOLE  powershell + v [ ] [ ] ^ x

| on <value for var.var-fileobj> line 1:
PS F:\Tf assignment> terraform plan
local_file.file6: Refreshing state... [id=b41db493d715bb84117f10ac7adc3c9e341df96e]

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

Terraform will perform the following actions:

# local_file.file6 will be destroyed
# (because local_file.file6 is not in configuration)
- resource "local_file" "file6" {
  - content          = "Content from map variable" -> null
  - directory_permission = "0777" -> null
  - file_permission   = "0777" -> null
  - filename         = "map.py" -> null
  - id               = "b41db493d715bb84117f10ac7adc3c9e341df96e" -> null
}

# local_file.file7 will be created
+ directory_permission = "0777"
+ file_permission     = "0777"
+ filename            = "object.py"
+ id                  = (known after apply)
}

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

Then terraform apply -auto-approve:

The screenshot shows a Visual Studio Code window with a Terraform assignment. The left sidebar has a 'TF ASSIGNMENT' section with a file explorer showing files like .terraform\providers\registry.ter..., local, random, .terraform.lock.hcl, himanshu_assignment.tf, object.py, terraform.tfstate, terraform.tfstate.backup, and variables.tf. The main editor area shows the 'object.py' file with Terraform configuration for a local_file resource. The configuration includes a 'description' variable and a 'content' variable. The 'actions' section indicates that 'local_file.file6' will be destroyed and 'local_file.file7' will be created. The 'Plan' section shows the results of the terraform plan command, indicating that 'local_file.file6' will be destroyed and 'local_file.file7' will be created. The 'Apply' section shows the results of the terraform apply command, indicating that the resources were successfully applied.

Output block

Initializing a output block

```
himanshu_assignment.tf ●
himanshu_assignment.tf > ...
1  ## Main TF file
2  ## Creating a local resource file1
3  ## dec is used to obtain decimal values
4  resource "local_file" "file1" {
5      content = "${random_id.alphaNumeric.dec}"
6      filename = "out.py"
7  }
8
9  # terraform random
10 resource "random_id" "alphaNumeric" {
11     byte_length = 8
12 }
13
14 # Output block
15 output "outputfile" {
16     value = local_file.file1.content
17 }
```

Terraform plan

```
himanshu_assignment.tf ●
himanshu_assignment.tf > ...
1  ## Main TF file
2  ## Creating a local resource file1
3  ## dec is used to obtain decimal values
4  resource "local_file" "file1" {
5      content = "${random_id.alphaNumeric.dec}"
6  }
7  }

TERMINAL  PROBLEMS  OUTPUT  DEBUG CONSOLE
powershell + v [] [X]

PS F:\Tf assignment> terraform plan
local_file.file7: Refreshing state... [id=4d134bc072212ace2df385dae143139da74ec0ef]

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

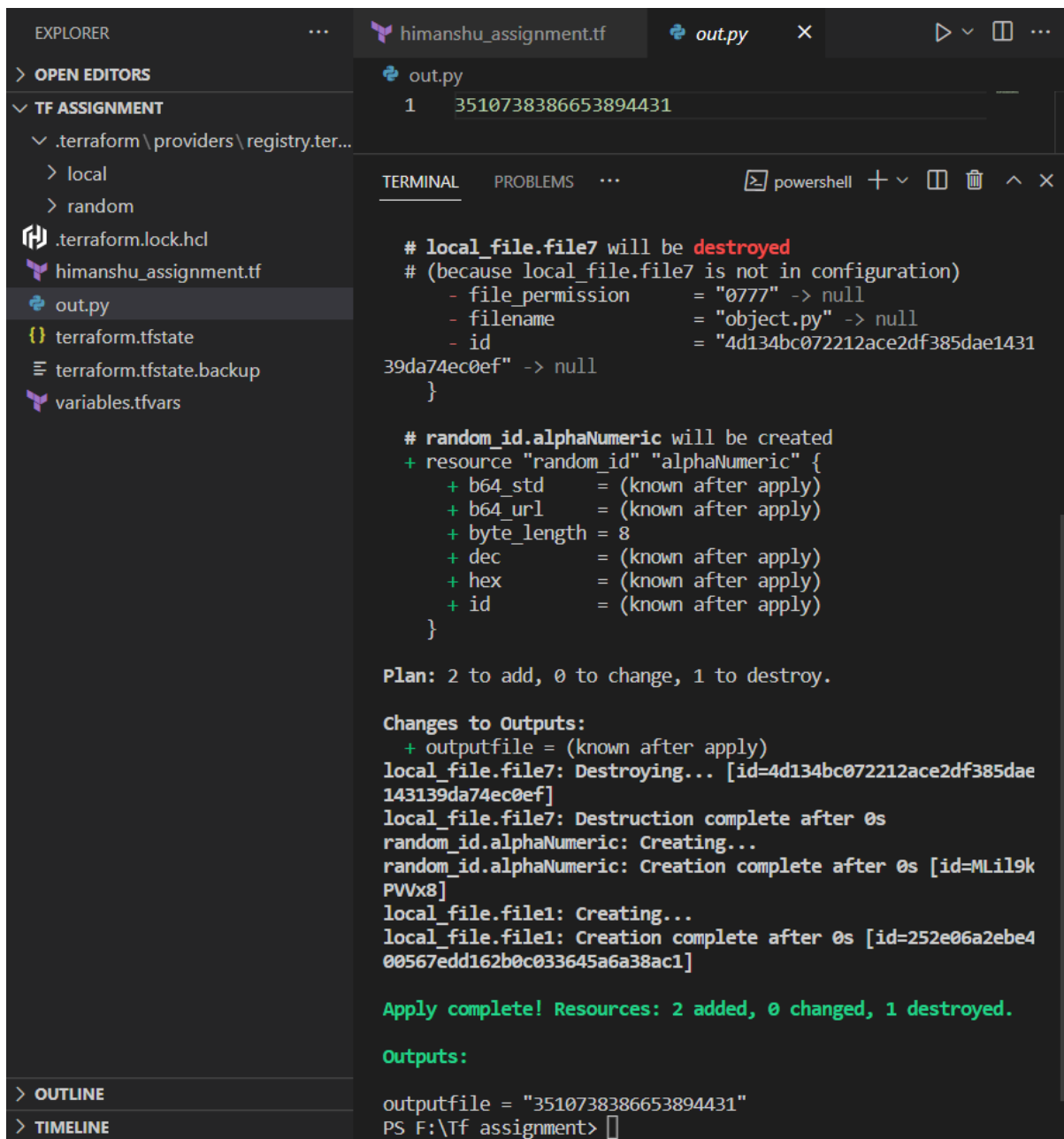
Terraform will perform the following actions:

# local_file.file1 will be created
+ resource "local_file" "file1" {
+   b64_url      = (known after apply)
+   byte_length = 8
+   dec         = (known after apply)
+   b64_url      = (known after apply)
+   byte_length = 8
+   dec         = (known after apply)
+   hex         = (known after apply)
+   id          = (known after apply)
}

Plan: 2 to add, 0 to change, 1 to destroy.

Changes to Outputs:
+ outputfile = (known after apply)
```

Terraform apply –auto-approve



```
EXPLORER
...
himanshu_assignment.tf
out.py
...
> OPEN EDITORS
TF ASSIGNMENT
  .terraform\providers\registry.ter...
    > local
    > random
.terraform.lock.hcl
himanshu_assignment.tf
out.py
terraform.tfstate
terraform.tfstate.backup
variables.tfvars

# local_file.file7 will be destroyed
# (because local_file.file7 is not in configuration)
- file_permission = "0777" -> null
- filename         = "object.py" -> null
- id               = "4d134bc072212ace2df385dae143139da74ec0ef" -> null
}

# random_id.alphaNumeric will be created
+ resource "random_id" "alphaNumeric" {
+   b64_std      = (known after apply)
+   b64_url      = (known after apply)
+   byte_length = 8
+   dec         = (known after apply)
+   hex         = (known after apply)
+   id          = (known after apply)
}

Plan: 2 to add, 0 to change, 1 to destroy.

Changes to Outputs:
+ outputfile = (known after apply)
local_file.file7: Destroying... [id=4d134bc072212ace2df385dae143139da74ec0ef]
local_file.file7: Destruction complete after 0s
random_id.alphaNumeric: Creating...
random_id.alphaNumeric: Creation complete after 0s [id=MLil9kPVVx8]
local_file.file1: Creating...
local_file.file1: Creation complete after 0s [id=252e06a2ebe400567edd162b0c033645a6a38ac1]

Apply complete! Resources: 2 added, 0 changed, 1 destroyed.

Outputs:
outputfile = "3510738386653894431"
PS F:\Tf assignment>
```