TASKS ON TERRAFORM

- **NAME: S.MUZZAMMIL HUSSAIN**
- **DATE: 21/07/2025**
- **BATCH: 11**
- **NO.OF TASKS: 1**

Task 1. Inject values from different ways of precedence in terraform to different resources of random

Use different sources of input values (CLI, env vars, tfvars, auto.tfvars, default, etc.)

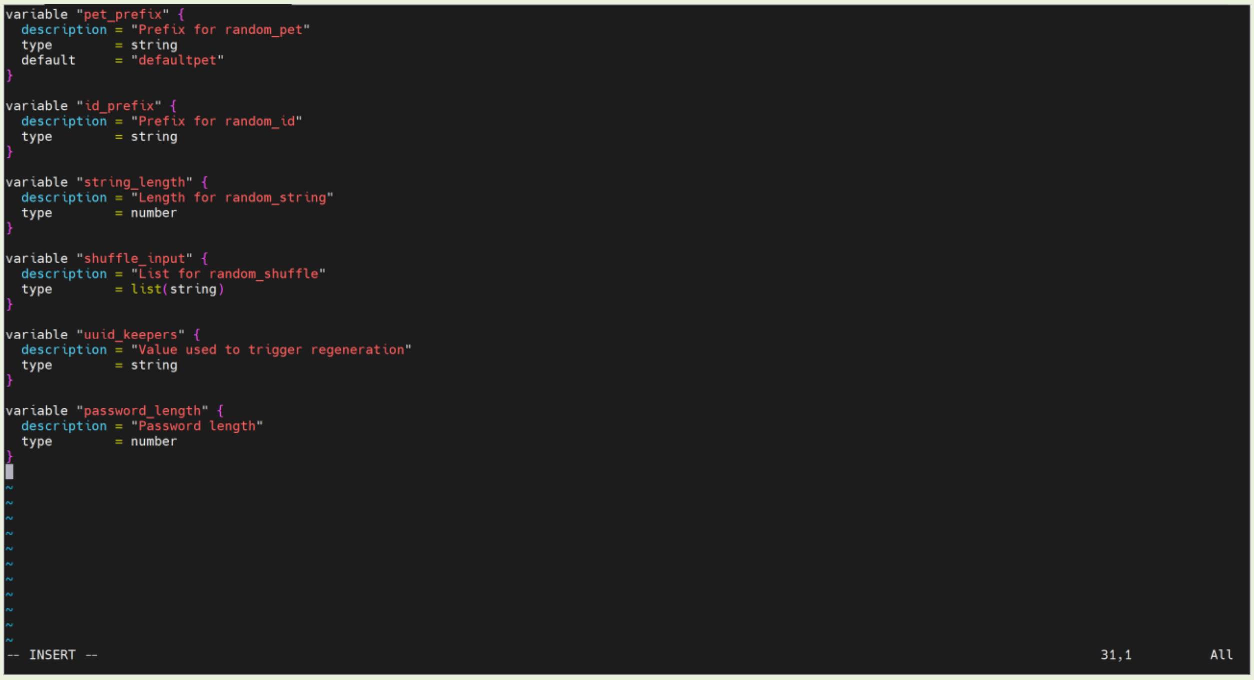
Inject those values into **different random resources** like random_pet, random_string, and random_id etc

Inject values from the **different input modes**, each targeting to a **different resource**:

Declaration of Variables:

```
variable "pet_prefix" {
  description = "Prefix for random_pet"
  type        = string
  default     = "defaultpet"
}

variable "id_prefix" {
  description = "Prefix for random_id"
  type        = string
}

variable "string_length" {
  description = "Length for random_string"
  type        = number
}

variable "shuffle_input" {
  description = "List for random_shuffle"
  type        = list(string)
}

variable "uuid_keepers" {
  description = "Value used to trigger regeneration"
  type        = string
}

variable "password_length" {
  description = "Password length"
  type        = number
}
```

Create a variables.tf file using **vi variables.tf** and enter the contents as follows and save using **esc:wq**

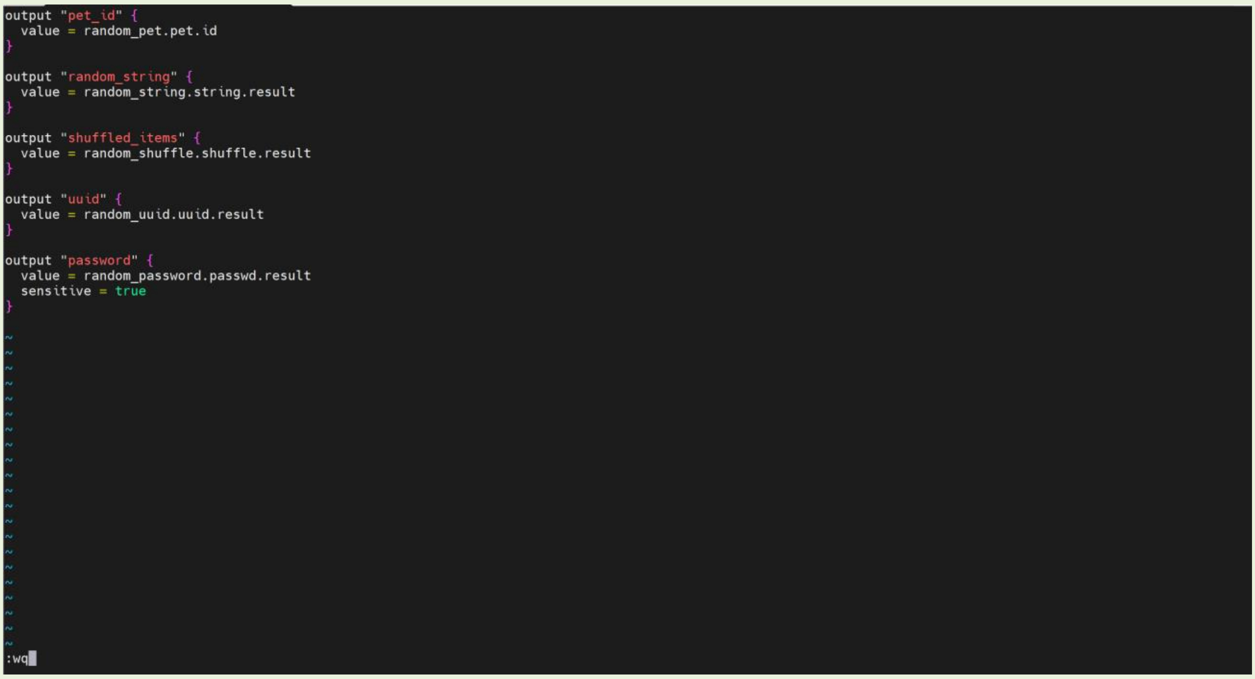


```
variable "pet_prefix" {
  description = "Prefix for random_pet"
  type        = string
  default     = "defaultpet"
}

variable "id_prefix" {
  description = "Prefix for random_id"
  type        = string
}

variable "string_length" {
  description = "Length for random_string"
  type        = number
}

variable "shuffle_input" {
  description = "List for random_shuffle"
  type        = list(string)
}

variable "uuid_keepers" {
  description = "Value used to trigger regeneration"
  type        = string
}

variable "password_length" {
  description = "Password length"
  type        = number
}

~
~
~
~
~
~
~
~
~
~
~
-- INSERT --                                                          31,1          All
```

We need to add output variables as follows

```
output "pet_id" {
  value = random_pet.pet.id
}
output "random_string" {
  value = random_string.string.result
}
output "shuffled_items" {
  value = random_shuffle.shuffle.result
}
output "uuid" {
  value = random_uuid.uuid.result
}
output "password" {
  value = random_password.passwd.result
  sensitive = true
}
```

Create a outputs.tf file using **vi outputs.tf** and enter the contents as follows and save using **esc:wq**

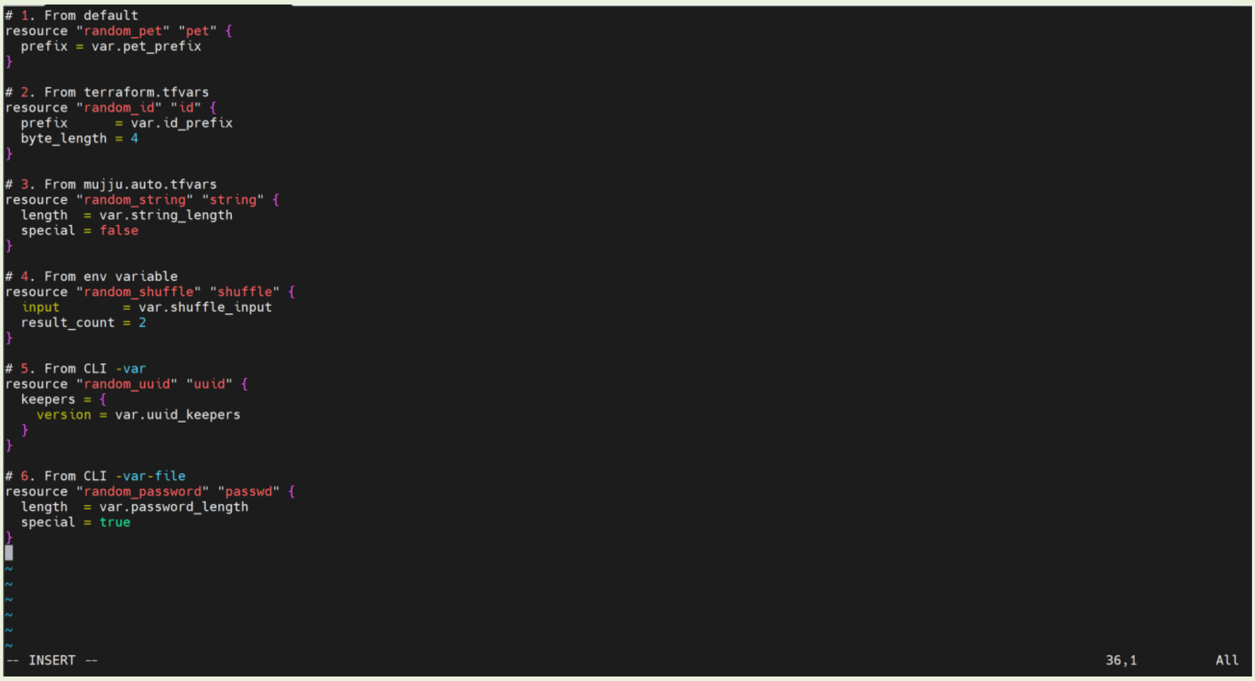


```
output "pet_id" {
  value = random_pet.pet.id
}

output "random_string" {
  value = random_string.string.result
}

output "shuffled_items" {
  value = random_shuffle.shuffle.result
}

output "uuid" {
  value = random_uuid.uuid.result
}

output "password" {
  value = random_password.passwd.result
  sensitive = true
}

~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq
```

Now create Resource Definitions

```
# 1. From default

resource "random_pet" "pet" {

  prefix = var.pet_prefix

}

# 2. From terraform.tfvars

resource "random_id" "id" {

  prefix      = var.id_prefix

  byte_length = 4

}

# 3. From mujju.auto.tfvars

resource "random_string" "string" {

  length  = var.string_length

  special = false

}

# 4. From env variable

resource "random_shuffle" "shuffle" {

  input       = var.shuffle_input

  result_count = 2
}

# 5. From CLI -var

resource "random_uuid" "uuid" {

  keepers = {

    version = var.uuid_keepers

  }

}

# 6. From CLI -var-file

resource "random_password" "passwd" {

  length  = var.password_length

  special = true

}
```

Create a file named main.tf using **vi main.tf** and enter the contents as follows and save using esc:wq

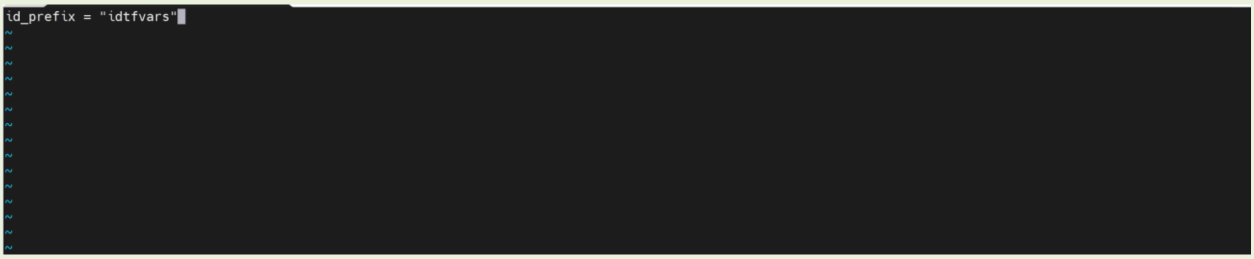


```
# 1. From default
resource "random_pet" "pet" {
  prefix = var.pet_prefix
}

# 2. From terraform.tfvars
resource "random_id" "id" {
  prefix      = var.id_prefix
  byte_length = 4
}

# 3. From mujju.auto.tfvars
resource "random_string" "string" {
  length  = var.string_length
  special = false
}

# 4. From env variable
resource "random_shuffle" "shuffle" {
  input        = var.shuffle_input
  result_count = 2
}

# 5. From CLI -var
resource "random_uuid" "uuid" {
  keepers = {
    version = var.uuid_keepers
  }
}

# 6. From CLI -var-file
resource "random_password" "passwd" {
  length  = var.password_length
  special = true
}

~
~
~
~
~
~
-- INSERT --                                                    36,1          All
```

Create a file named **terraform.tfvars** using **vi terraform.tfvars** and enter the contents as follows and save using esc:wq

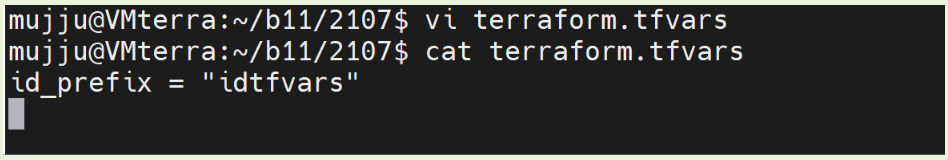
id_prefix = "idtfvars"

terraform.tfvars → Value for random_id



```
id_prefix = "idtfvars"
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

Check the contents inside **terraform.tfvars** file using **cat terraform.tfvars**

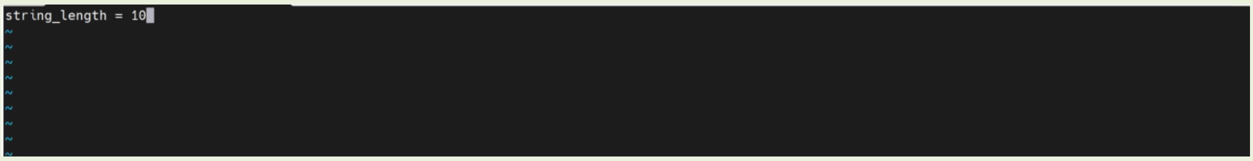


```
mujju@VMterra:~/b11/2107$ vi terraform.tfvars
mujju@VMterra:~/b11/2107$ cat terraform.tfvars
id_prefix = "idtfvars"
```

Create a file named **mujju.auto.tfvars** using **vi mujju.auto.tfvars** and enter the contents as follows and save using esc:wq

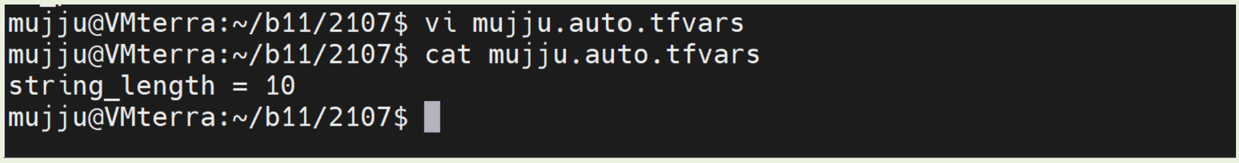
string_length = 10

mujju.auto.tfvars → Value for random_string



```
string_length = 10
~
~
~
~
~
~
~
~
~
```

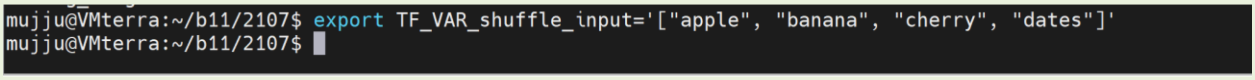
Check the contents inside **mujju.auto.tfvars** file using **cat mujju.auto.tfvars**



```
mujju@VMterra:~/b11/2107$ vi mujju.auto.tfvars
mujju@VMterra:~/b11/2107$ cat mujju.auto.tfvars
string_length = 10
mujju@VMterra:~/b11/2107$
```

Export Env Variable for random_shuffle

export TF_VAR_shuffle_input='["apple", "banana", "cherry", "dates"]'



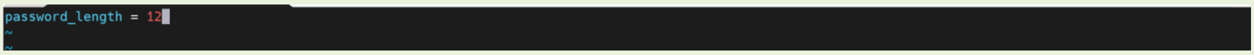
```
mujju@VMterra:~/b11/2107$ export TF_VAR_shuffle_input='["apple", "banana", "cherry", "dates"]'
mujju@VMterra:~/b11/2107$
```

Create CLI -var-file for random_password using

vi prod.properties

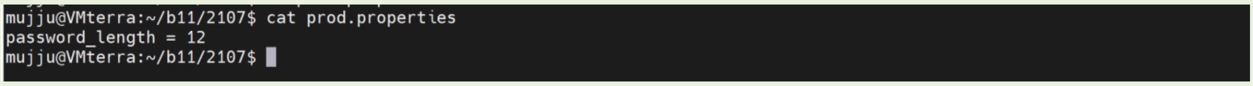
Enter the contents as follows and save using **esc:wq**

password_length = 12



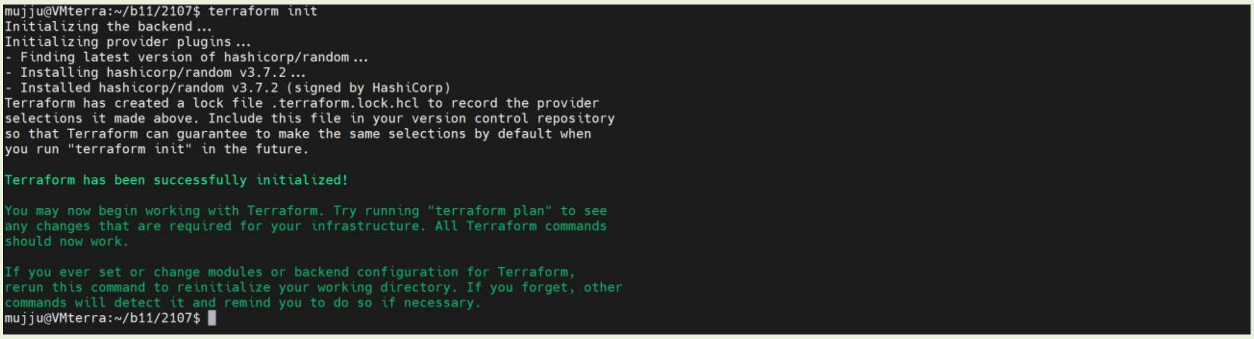
```
password_length = 12
~
```

Check the contents inside **prod.properties** file using **cat prod.properties**



```
mujju@VMterra:~/b11/2107$ cat prod.properties
password_length = 12
mujju@VMterra:~/b11/2107$ 
```

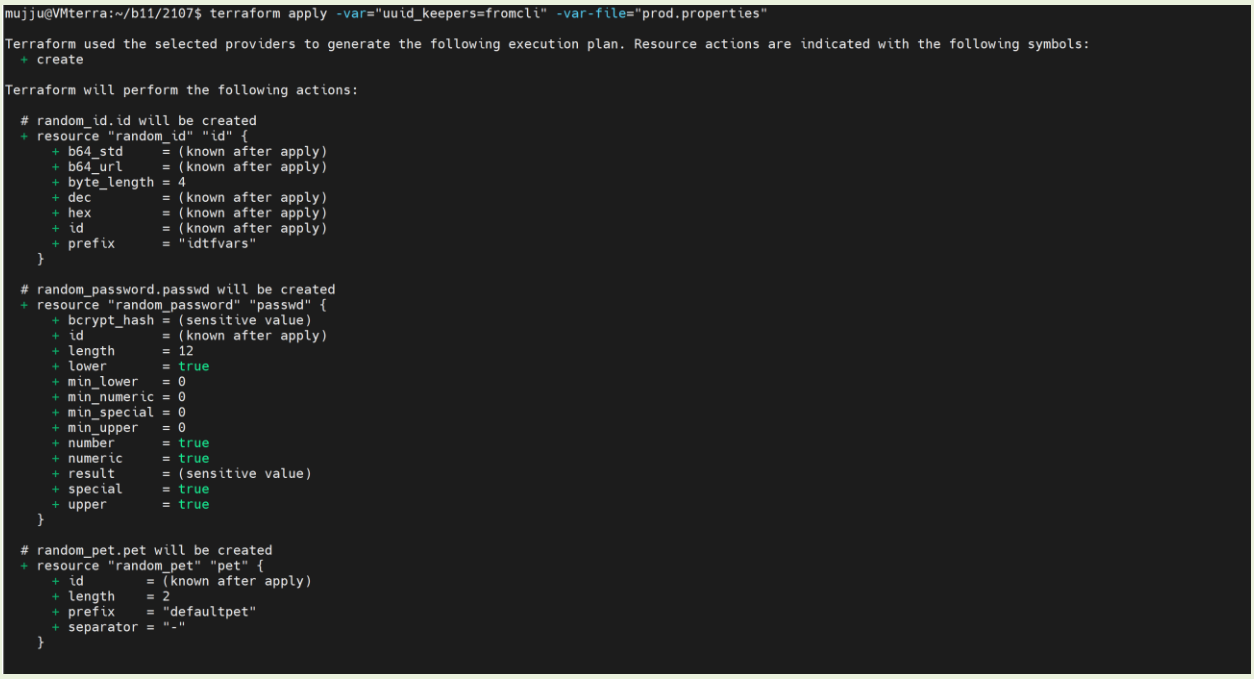
Run the command **terraform init**



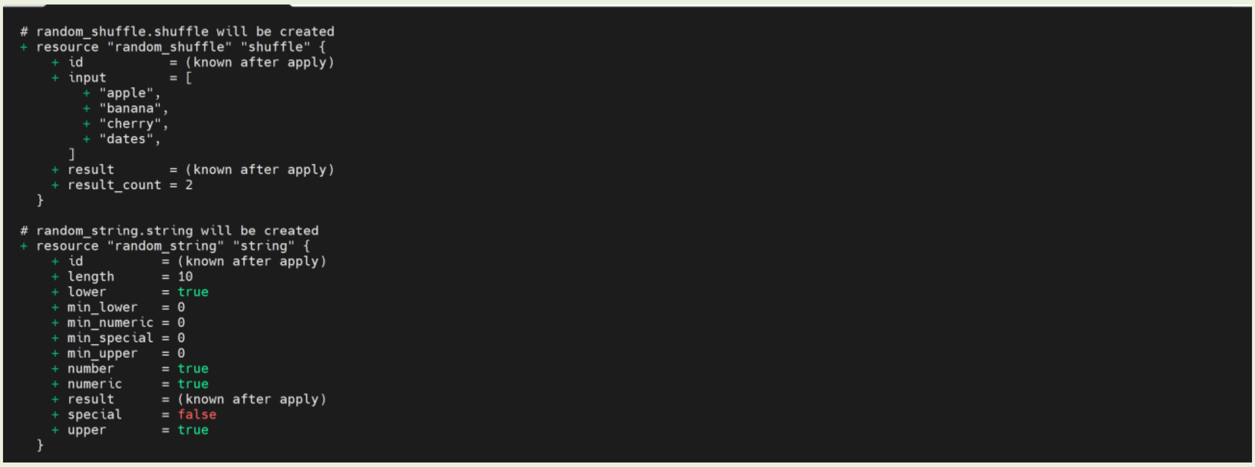
```
mujju@VMterra:~/b11/2107$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/random...
- Installing hashicorp/random v3.7.2...
- Installed hashicorp/random v3.7.2 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl 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.
mujju@VMterra:~/b11/2107$ 
```

Run the command

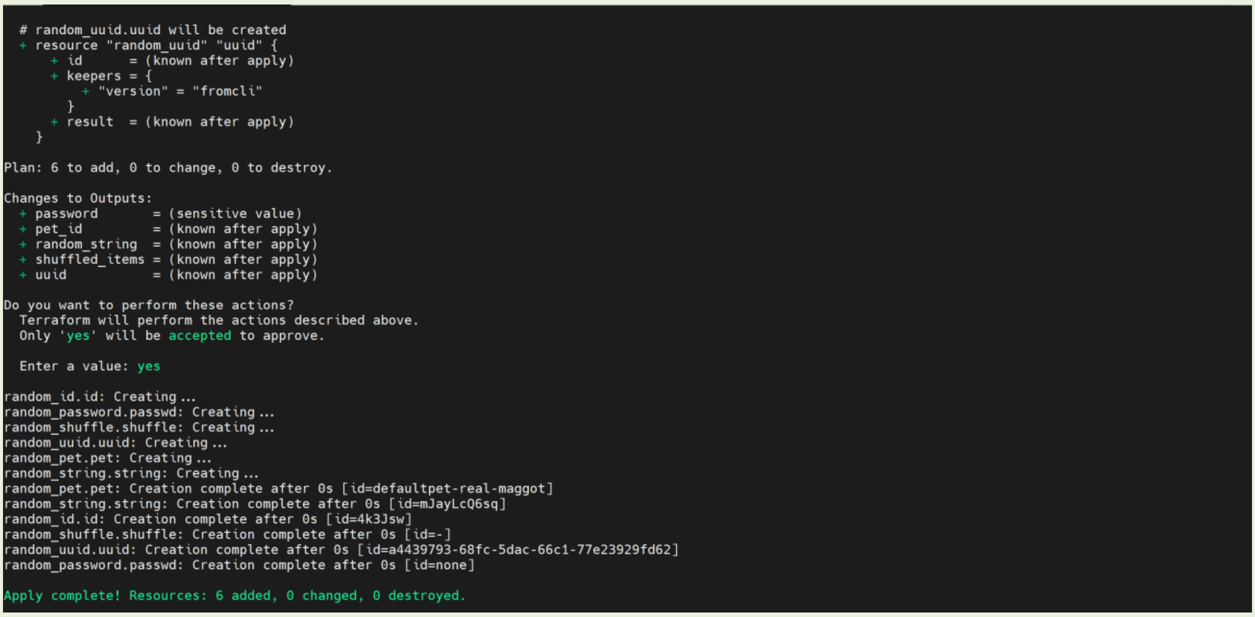
terraform apply -var="uuid_keepers=fromcli" -var-file="prod.properties"



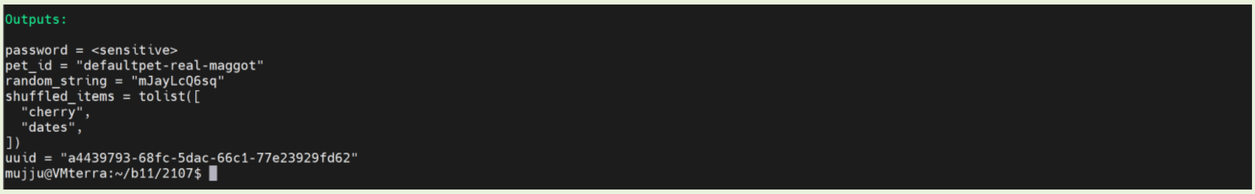
```
mujju@VMterra:~/b11/2107$ terraform apply -var="uuid_keepers=fromcli" -var-file="prod.properties"

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:

  # random_id.id will be created
  + resource "random_id" "id" {
      + b64_std     = (known after apply)
      + b64_url     = (known after apply)
      + byte_length = 4
      + dec         = (known after apply)
      + hex         = (known after apply)
      + id          = (known after apply)
      + prefix      = "idtfvars"
    }

  # random_password.passwd will be created
  + resource "random_password" "passwd" {
      + bcrypt_hash = (sensitive value)
      + id          = (known after apply)
      + length      = 12
      + lower       = true
      + min_lower   = 0
      + min_numeric = 0
      + min_special = 0
      + min_upper   = 0
      + number      = true
      + numeric     = true
      + result      = (sensitive value)
      + special     = true
      + upper       = true
    }

  # random_pet.pet will be created
  + resource "random_pet" "pet" {
      + id        = (known after apply)
      + length    = 2
      + prefix    = "defaultpet"
      + separator = "-"
    }
```



```
  # random_shuffle.shuffle will be created
  + resource "random_shuffle" "shuffle" {
      + id           = (known after apply)
      + input        = [
          + "apple",
          + "banana",
          + "cherry",
          + "dates",
        ]
      + result       = (known after apply)
      + result_count = 2
    }

  # random_string.string will be created
  + resource "random_string" "string" {
      + id          = (known after apply)
      + length      = 10
      + lower       = true
      + min_lower   = 0
      + min_numeric = 0
      + min_special = 0
      + min_upper   = 0
      + number      = true
      + numeric     = true
      + result      = (known after apply)
      + special     = false
      + upper       = true
    }
```

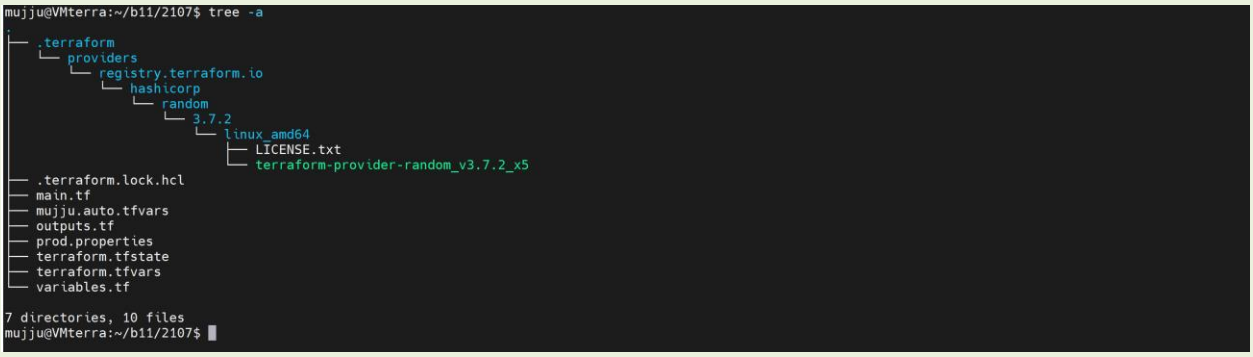


```
  # random_uuid.uuid will be created
  + resource "random_uuid" "uuid" {
      + id      = (known after apply)
      + keepers = {
          + "version" = "fromcli"
        }
      + result  = (known after apply)
    }

Plan: 6 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + password       = (sensitive value)
  + pet_id         = (known after apply)
  + random_string  = (known after apply)
  + shuffled_items = (known after apply)
  + uuid           = (known after apply)

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_id.id: Creating...
random_password.passwd: Creating...
random_shuffle.shuffle: Creating...
random_uuid.uuid: Creating...
random_pet.pet: Creating...
random_string.string: Creating...
random_pet.pet: Creation complete after 0s [id=defaultpet-real-maggot]
random_string.string: Creation complete after 0s [id=mJayLcQ6sq]
random_id.id: Creation complete after 0s [id=4k3Jsw]
random_shuffle.shuffle: Creation complete after 0s [id=-]
random_uuid.uuid: Creation complete after 0s [id=a4439793-68fc-5dac-66c1-77e23929fd62]
random_password.passwd: Creation complete after 0s [id=none]

Apply complete! Resources: 6 added, 0 changed, 0 destroyed.
```



```
Outputs:

password = <sensitive>
pet_id = "defaultpet-real-maggot"
random_string = "mJayLcQ6sq"
shuffled_items = tolist([
  "cherry",
  "dates",
])
uuid = "a4439793-68fc-5dac-66c1-77e23929fd62"
mujju@VMterra:~/b11/2107$ 
```

Run the command tree –a



```
mujju@VMterra:~/b11/2107$ tree -a
.
├── .terraform
│   └── providers
│       └── registry.terraform.io
│           └── hashicorp
│               └── random
│                   └── 3.7.2
│                       └── linux_amd64
│                           ├── LICENSE.txt
│                           └── terraform-provider-random_v3.7.2_x5
├── .terraform.lock.hcl
├── main.tf
├── mujju.auto.tfvars
├── outputs.tf
├── prod.properties
├── terraform.tfstate
├── terraform.tfvars
└── variables.tf

7 directories, 10 files
mujju@VMterra:~/b11/2107$ 
```

Final Folder Structure

terraform

```
├── main.tf
├── mujju.auto.tfvars
├── outputs.tf
├── prod.properties
├── terraform.tfvars
├── variables.tf
```

Summary

Input Mode	Resource Used	Variable	File/Command
CLI / Interactive	random_pet.pet	pet_prefix	Prompted on terraform apply
Default	random_pet.pet	pet_prefix	Default in variables.tf
Environment Variable	random_shuffle	shuffle_input	TF_VAR_shuffle_input=...
terraform.tfvars	random_id.id	id_prefix	terraform.tfvars file
.auto.tfvars	random_string	string_length	mujju.auto.tfvars
CLI -var	random_uuid.uuid	uuid_keepers	-var="uuid_keepers=..."
CLI -var-file	random_password	password_length	-var-file="prod.properties"

Terraform Resources Used

Resource Type	Description
random_pet	Friendly, human-readable name
random_id	Random ID with a prefix
random_string	Random string of given length
random_shuffle	Shuffled input list
random_uuid	UUID generated with keepers
random_password	Secure random password