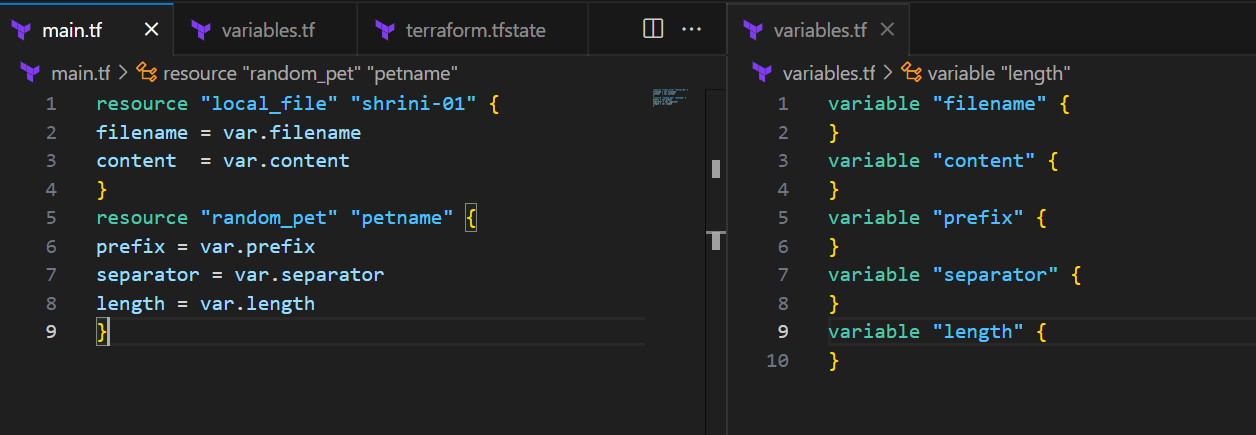
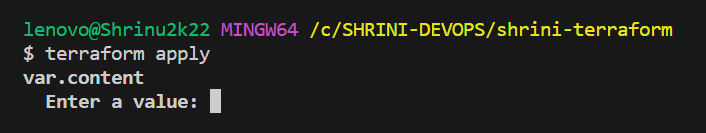
**Terraform-03**

**\*\*Using of variables.tf:**

**1. Interactive mode:**

****

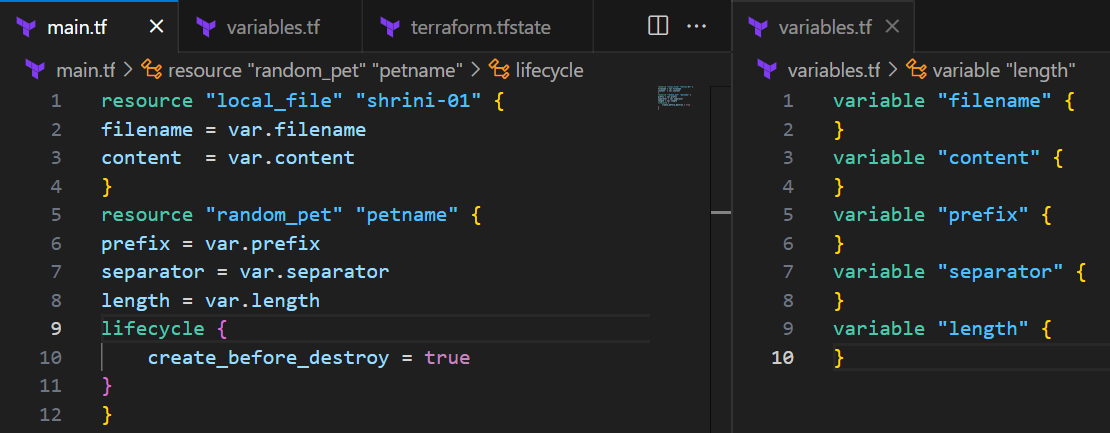
**“This will get activated if we don’t pass default value in variable.tf file”**

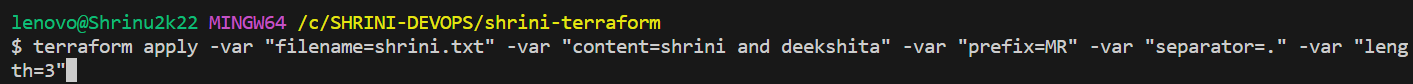
**2. Using Variable.tf file:**

****

**“In the variables.tf we are declaring our variables value in the variables.tf file”**

**3. Command line flags:**

****

****

**If we didn’t declared any variables in the variables.tf and want declared through the command we can use:**

**“Command line flag = -var”**

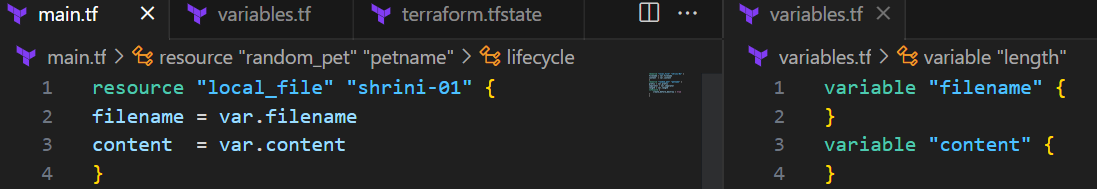
**4. Environment variables:**

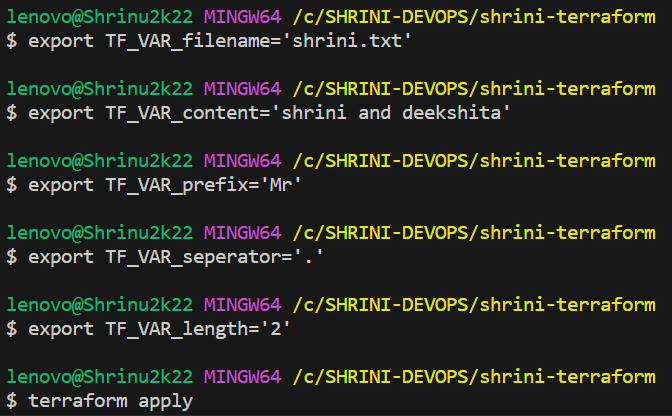
* **If we don’t want to save our password in variables.tf we can declared env variables.**
* **By commands :  
  export TF\_VAR\_filename="/root.pets.txt"  
  export command will work only on Linux**

**Set-Item –Path env:TF\_VAR\_filename -Value 'wild.txt'**

**Set-item –Path will work on windows**

* **These are temporary, if we reboot/close our server they get collapse**

****

****

**5. Variable definition file:**

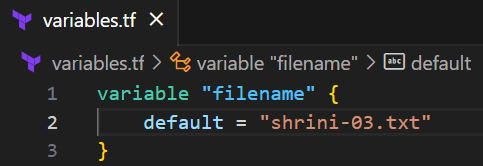
**\*\* Precedence order for variables:**

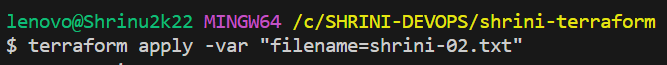
**“ if we declaring variables by using Variables.tf, environment variable, variable definition file and command line flags at a same time then it will take the variables from as precedence order”**

**1. Environment variables:**

****

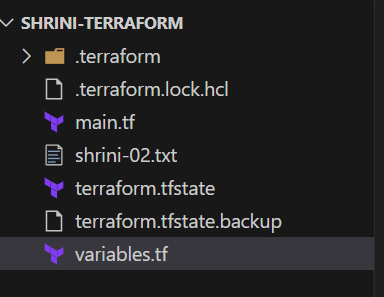
**2. Terraform.tf vars:**

****

**4. -var or -var-file (Command line flags):**

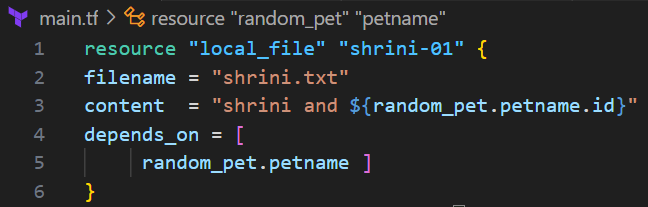
|  |
| --- |
| Order Option |
| 1. Environment variables |
| 2. Terraform.tf vars |
| 3. \*.auto.tfvars (alphabetical order) |
| 4. -var or -var-file (Command line flags) |

**{{ It will take the variables according to the order as below pic we are seeing that “shrini-02.txt” is created and it is over-riding the env.variables, variables.tf, and in last we mentioned command line flags “shrini-02.txt”}}**

****

**\*\*Resource Attribute reference: (with the help of “$”)**

**If we want to link any two resources together by using resources attributes we can mention our dependencies on our resources file like:**

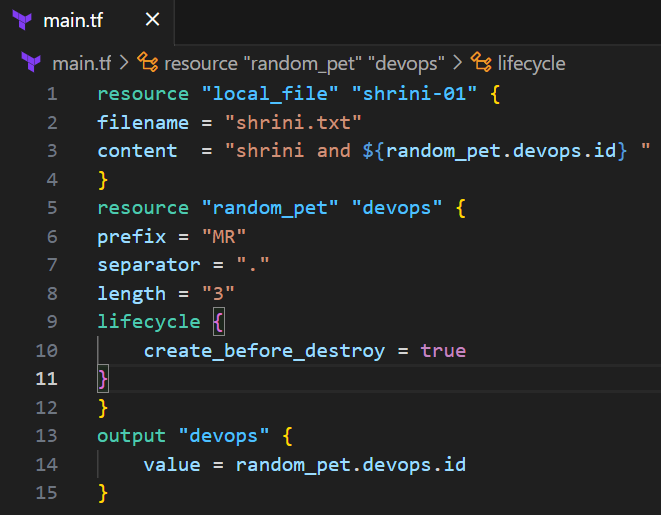
****

**Implicitly-behavior:   
terraform smart enough to pick the dependences on the resources file;**

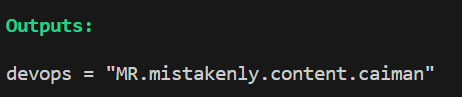
****

**\*\*Terraform output:**

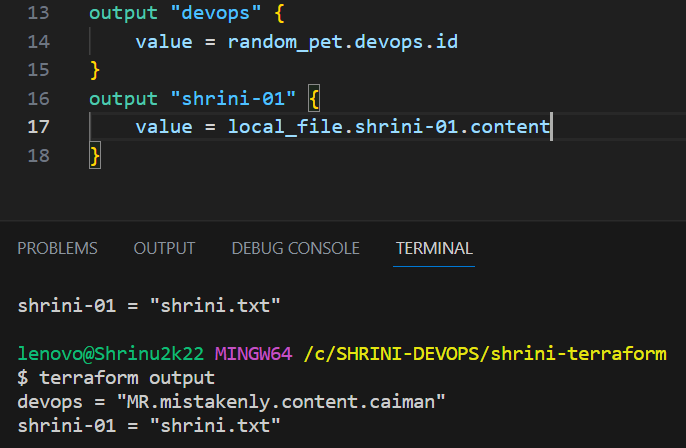
**To print the output present in the resources file**

****

**In the output we have to give “logical-name” and in value we have to mention the “provider.logicalname.Attributes” then we have to run “terraform output” and make sure the template should already applied “terraform apply”**

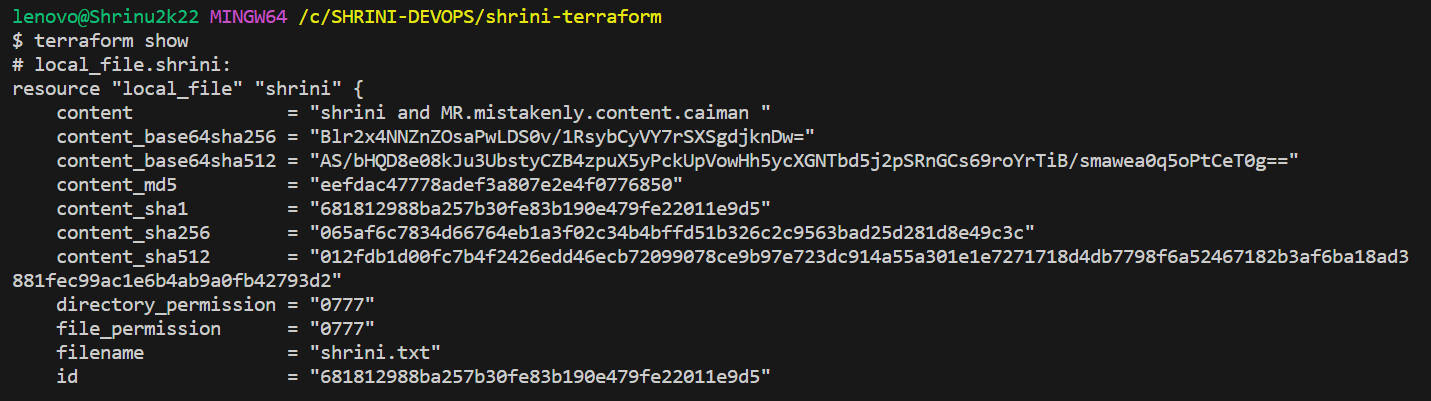
****

**Some more examples of terraform output;**

****

**\*\*Terraform show:**

**Display the current state. file (or) it will show the complete info of our resources**

****

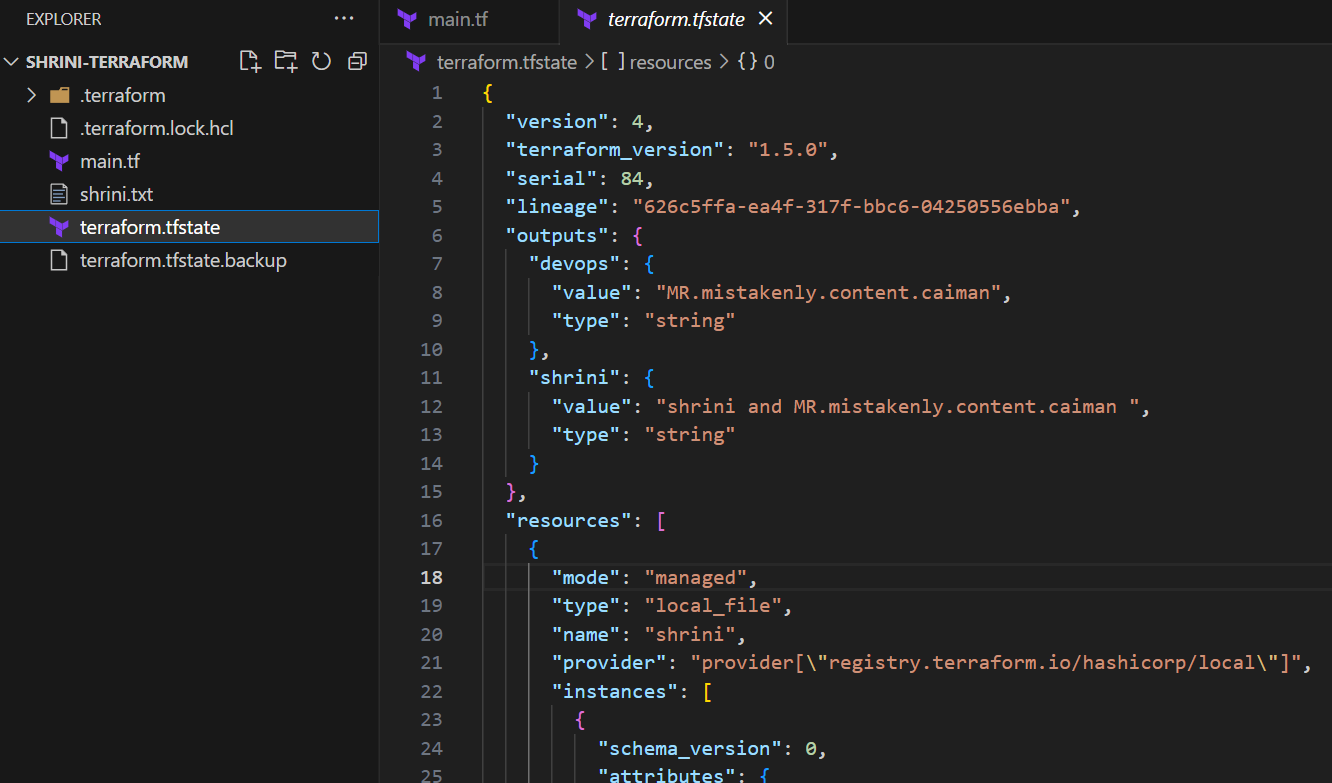
**\*\*state. File:**

**It is in “JSON” format**

**Terraform state file will have the complete record of the infra created by terraform.**

**State file is considered as a blue print of all the resources terraform manages.**

**terraform.tf.state will be the name of the file and this will created only after using terraform apply command.**

****