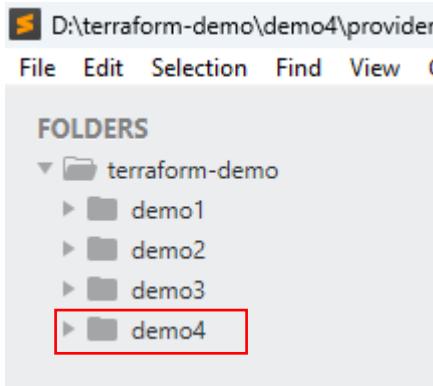
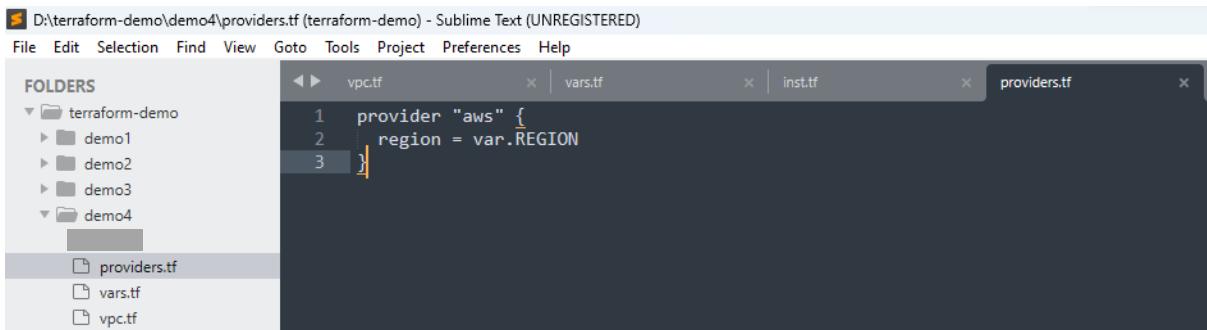


Multi-Resources

1. So, in this lab you are going to launch multiple resource through terraform.
2. First you are going to create a new sub folder out of your main folder.



3. After that you are going to create multiple files which will contains the variables and resources.
4. Start with creating providers file then you are going to create variables file after that VPC file because we are going to launch the VPC.



5. Now you are going to write the code in these files. In some files the code is long to get in the snapshot.

The screenshot shows a Sublime Text window with the title bar "D:\terraform-demo\demo4\vars.tf • (terraform-demo) - Sublime Text (UNREGISTERED)". The menu bar includes File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, and Help. On the left, a sidebar titled "FOLDERS" shows the directory structure:

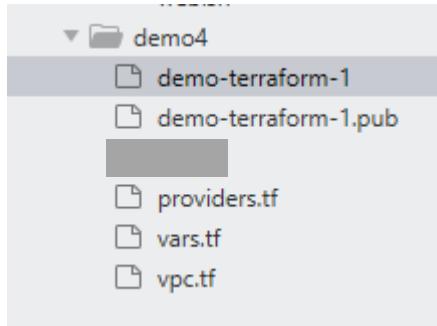
- terraform-demo
 - demo1
 - demo2
 - demo3
 - .terraform
 - .terraform.lock.hcl
 - demo-terraform-1
 - demo-terraform-1.pub
 - inst.tf
 - providers.tf
 - terraform.tfstate
 - terraform.tfstate.backup
 - vars.tf
 - /* web.sh
 - demo4
 - demo-terraform-1
 - demo-terraform-1.pub
 - providers.tf
 - vars.tf
 - vpc.tf

6. Again, you can get this code from GitHub.
7. Now you have to copy the public and private key to this folder which you had created in the previous lab.

```
cp demo-terraform-1 .../demo4/  
cp demo-terraform-.pub .../demo4/
```

```
LAPTOP-G2CAKBK8 MINGW64 /d/terraform-demo/demo3
$ cp demo-terraform-1 .../demo4/

LAPTOP-G2CAKBK8 MINGW64 /d/terraform-demo/demo3
$ ls
demo-terraform-1      inst.tf      terraform.tfstate      vars.tf
demo-terraform-1.pub  providers.tf  terraform.tfstate.backup  web.sh*
LAPTOP-G2CAKBK8 MINGW64 /d/terraform-demo/demo3
$ cp demo-terraform-1.pub .../demo4/
```



8. In the VPC file in the code we have created a VPC, subnets, route table and internet gateway.
9. Once you have copied your key and created your providers file, variable file and your VPC file.
10. Then open your gitbash and execute it, this will create your VPC.
11. Now in gitbash go into your current folder and do the listing of files.

```
$ ls
demo-terraform-1  demo-terraform-1.pub  providers.tf  vars.tf  vpc.tf
```

12. Now run the initialization command and initialize your code.

terraform init

```
$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.38.0...
- Installed hashicorp/aws v5.38.0 (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.
```

13. Then run the validate and format command.

```
terraform validate
```

```
terraform fmt
```

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

14. Now run the plan command to see what is going to add. Here you can see that the VPC
will be created and 12 things are going to add in it.

15. After that run the apply command to execute your resources.

```
terraform plan
```

```
terraform apply
```

```
# aws_vpc.demo will be created
+ resource "aws_vpc" "demo" {
    + arn                                = (known after apply)
    + cidr_block                          = "10.0.0.0/16"
    + default_network_acl_id             = (known after apply)
    + default_route_table_id              = (known after apply)
    + default_security_group_id          = (known after apply)
    + dhcp_options_id                   = (known after apply)
    + enable_dns_hostnames               = true
    + enable_dns_support                 = true
    + enable_network_address_usage_metrics = (known after apply)
    + id                                 = (known after apply)
    + instance_tenancy                  = "default"
    + ipv6_association_id                = (known after apply)
    + ipv6_cidr_block                   = (known after apply)
    + ipv6_cidr_block_network_border_group = (known after apply)
    + main_route_table_id                = (known after apply)
    + owner_id                           = (known after apply)
    + tags                               = {
        + "Name" = "demo-vpc"
    }
    + tags_all                           = {
        + "Name" = "demo-vpc"
    }
}
```

```
Plan: 12 to add, 0 to change, 0 to destroy.
```

16. Here you can see that everything has been created. Now you can navigate to your console in the VPC you will see all the things that you have created so far.

```

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_vpc.demo: Creating...
aws_vpc.demo: Still creating... [10s elapsed]
aws_vpc.demo: Creation complete after 11s [id=vpc-0fcc43b620eaec17a]
aws_internet_gateway.demo-IGW: Creating...
aws_subnet.demo-public-3: Creating...
aws_subnet.demo-private-2: Creating...
aws_subnet.demo-public-2: Creating...
aws_subnet.demo-public-1: Creating...
aws_subnet.demo-private-3: Creating...
aws_subnet.demo-private-1: Creating...
aws_internet_gateway.demo-IGW: Creation complete after 1s [id=igw-08db3838671877a39]
aws_route_table.demo-public-RT: Creating...
aws_route_table.demo-public-RT: Creation complete after 1s [id=rtb-0e6adabbd8f1a6963]
aws_subnet.demo-public-3: Still creating... [10s elapsed]
aws_subnet.demo-public-1: Still creating... [10s elapsed]
aws_subnet.demo-private-2: Still creating... [10s elapsed]
aws_subnet.demo-public-2: Still creating... [10s elapsed]
aws_subnet.demo-private-1: Still creating... [10s elapsed]
aws_subnet.demo-private-3: Still creating... [10s elapsed]
aws_subnet.demo-private-1: Creation complete after 11s [id=subnet-0c94939d4f4101bd8]
aws_subnet.demo-public-2: Creation complete after 11s [id=subnet-0a2d979e6f57d921d]
aws_route_table_association.demo-public-2-a: Creating...
aws_subnet.demo-private-2: Creation complete after 11s [id=subnet-0f6112a4aeb94f2b2]
aws_subnet.demo-public-3: Creation complete after 11s [id=subnet-0bf38c1d06b6bdd15]
aws_route_table_association.demo-public-3-a: Creating...
aws_subnet.demo-public-1: Creation complete after 11s [id=subnet-01d2349d0ecdc14b7]
aws_route_table_association.demo-public-1-a: Creating...
aws_subnet.demo-private-1: Creation complete after 11s [id=subnet-0c2826f68b90111af]
aws_route_table_association.demo-public-2-a: Creation complete after 1s [id=rtbassoc-0ecc0c4aea34aab1e]
aws_route_table_association.demo-public-3-a: Creation complete after 1s [id=rtbassoc-09dda50f7b3fe115f]
aws_route_table_association.demo-public-1-a: Creation complete after 1s [id=rtbassoc-0da500b51540846a2]

Apply complete! Resources: 12 added, 0 changed, 0 destroyed.

```

17. Below you can see the resources that are created.

Your VPCs (1/2) Info					
<input type="button" value="C"/> <input type="button" value="Actions ▾"/> <input type="button" value="Create VPC"/> < 1 > 					
Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP
<input type="checkbox"/>	vpc-00e852ef26c39581b	Available	172.31.0.0/16	-	dopt-C
<input checked="" type="checkbox"/>	vpc-0fcc43b620eaec17a	Available	10.0.0.0/16	-	dopt-C

vpc-0fcc43b620eaec17a / demo-vpc			
Details	Resource map	CIDRs	Flow logs
Details CIDRs Flow logs Tags Integrations			
Details			
VPC ID vpc-0fcc43b620eaec17a	State Available	DNS hostnames Enabled	DNS resolution Enabled
Tenancy Default	DHCP option set dopt-0689daf4ca5f7be79	Main route table rtb-05ae8aa93c8d010a4	Main network ACL acl-060ef2959226ce0f4
Default VPC No	IPv4 CIDR 10.0.0.0/16	IPv6 pool -	IPv6 CIDR (Network border group) -
Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups -	Owner ID 878893308172	

Subnets (9) Info

<input type="checkbox"/>	Name	Subnet ID	State	VPC	IPv4 CIDR
<input type="checkbox"/>	-	subnet-090908b5b996470fc	Available	vpc-00e852ef26c39581b	172.31.16.0/20
<input type="checkbox"/>	-	subnet-01c162e279b989d09	Available	vpc-00e852ef26c39581b	172.31.32.0/20
<input type="checkbox"/>	-	subnet-05b1afa053579e078	Available	vpc-00e852ef26c39581b	172.31.0.0/20
<input type="checkbox"/>	demo-private-1	subnet-0c2826f68b90111af	Available	vpc-0fcc43b620eaec17a demo...	10.0.4.0/24
<input type="checkbox"/>	demo-private-2	subnet-0f6112a4aeb94f2b2	Available	vpc-0fcc43b620eaec17a demo...	10.0.5.0/24
<input type="checkbox"/>	demo-private-3	subnet-0c94939d4f4101bd8	Available	vpc-0fcc43b620eaec17a demo...	10.0.6.0/24
<input type="checkbox"/>	demo-public-1	subnet-01d2349d0ecd14b7	Available	vpc-0fcc43b620eaec17a demo...	10.0.1.0/24
<input type="checkbox"/>	demo-public-2	subnet-0a2d979e6f57d921d	Available	vpc-0fcc43b620eaec17a demo...	10.0.2.0/24
<input type="checkbox"/>	demo-public-3	subnet-0bf38c1d06b6bdd15	Available	vpc-0fcc43b620eaec17a demo...	10.0.3.0/24

Route tables (3) Info

<input type="checkbox"/>	Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC
<input type="checkbox"/>	demo-public-RT	rtb-0e6adabbd8f1a6963	3 subnets	-	No	vpc-0fcc43b620eaec17a
<input type="checkbox"/>	-	rtb-05ae8aa93c8d010a4	-	-	Yes	vpc-0fcc43b620eaec17a
<input type="checkbox"/>	-	rtb-0dd95fab2ddaaee74	-	-	Yes	vpc-00e852ef26c39581b

Internet gateways (1/2) Info

<input type="checkbox"/>	Name	Internet gateway ID	State	VPC ID	Owner
<input checked="" type="checkbox"/>	demo-IGW	igw-08db3838671877a39	Attached	vpc-0fcc43b620eaec17a demo-vpc	878893308172
<input type="checkbox"/>	-	igw-0dbb1d264233ccbaf	Attached	vpc-00e852ef26c39581b	878893308172

==

igw-08db3838671877a39 / demo-IGW

[Details](#) [Tags](#)

Details

Internet gateway ID igw-08db3838671877a39	State Available	VPC ID vpc-0fcc43b620eaec17a demo-vpc	Owner 878893308172
--	--------------------	--	-----------------------

18. Now we are going to create the security group.

The screenshot shows a Sublime Text window with the following file structure:

- FOLDERS:**
 - terraform-demo
 - demo1
 - demo2
 - demo3
 - .terraform
 - .terraform.lock.hcl
 - demo-terraform-1
 - demo-terraform-1.pub
 - inst.tf
 - providers.tf
 - terraform.tfstate
 - terraform.tfstate.backup
 - vars.tf
 - /* web.sh
- demo4
- .terraform
- .terraform.lock.hcl
- demo-terraform-1
- demo-terraform-1.pub
- providers.tf
- securitygrp.tf
- terraform.tfstate
- vars.tf
- vpc.tf

The **securitygrp.tf** file contains the following Terraform code:

```

1 resource "aws_security_group" "demo_terraform_sg" {
2   vpc_id      = aws_vpc.demo.id
3   name        = "demo-terraform-sg"
4   description = "Security Group for demo ssh"
5   egress {
6     from_port  = 0
7     to_port    = 0
8     protocol   = "-1"
9     cidr_blocks = ["0.0.0.0/0"]
10 }
11
12 ingress {
13   from_port  = 22
14   to_port    = 22
15   protocol   = "tcp"
16   cidr_blocks = [var.Anywhere-IPv4]
17 }
18 tags = {
19   Name = "allow-ssh"
20 }
21 }
22

```

19. Again, go back to gitbash and first run plan command because this will check for the new resource afterwards run the apply command to execute our resource.
20. Below you can see that it has added one resource and then it created it and applied it.
21. You can also check your security group on the console as well.

```

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

aws_security_group.demo_terraform_sg: Creating...
aws_security_group.demo_terraform_sg: Creation complete after 3s [id=sg-03659cec435c66432]

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

```

The screenshot shows the AWS VPC Security Groups console. A single security group, 'allow-ssh', is listed with the following details:

- Name:** allow-ssh
- Security group ID:** sg-03659cec435c66432
- Security group name:** demo-terraform-sg
- VPC ID:** vpc-0fcc43b620eaec17a
- Description:** Sec

The 'Inbound rules' tab is selected, showing one rule:

Name	Security group rule...	IP version	Type	Protocol	Port range
-	sgr-09189b5ed84793...	IPv4	SSH	TCP	22

22. Now you can also copy your web.sh file from previous lab to this lab.

23. After that create your instance file.

```

D:\terraform-demo\demo4\inst.tf (terraform-demo) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
  - terraform-demo
    - demo1
    - demo2
    - demo3
      - .terraform
        - .terraform.lock.hcl
        - demo-terraform-1
        - demo-terraform-1.pub
        - inst.tf
        - providers.tf
        - terraform.tfstate
        - terraform.tfstate.backup
        - vars.tf
        /* web.sh
    - demo4
      - .terraform
        - .terraform.lock.hcl
        - demo-terraform-1
        - demo-terraform-1.pub
        - inst.tf
        - providers.tf
        - securitygrp.tf
        - terraform.tfstate
        - terraform.tfstate.backup
        - vars.tf
        - vpc.tf
        /* web.sh
vars.tf
vpc.tf
web.sh

1 resource "aws_key_pair" "demo-terraform-1" {
2   key_name   = "demo-terraform-1"
3   public_key = file(var.PUB_KEY)
4 }
5
6 resource "aws_instance" "demo-terra" {
7   ami           = var.AMIS[var.REGION]
8   instance_type = "t2.micro"
9   subnet_id     = aws_subnet.demo-public-1.id
10  key_name      = aws_key_pair.demo-key.key_name
11  vpc_security_group_ids = [aws_security_group.demo_terraform_sg.id]
12  tags = [
13    Name = "my-demo-instance"
14  ]
15 }
16
17 resource "aws_ebs_volume" "vol_4_demo" {
18   availability_zone = var.ZONE1
19   size              = 3
20   tags = [
21     Name = "extr-vol-4-demo"
22   ]
23 }
24
25 resource "aws_volume_attachment" "atchk_vol_demo" {
26   device_name = "/dev/xvdh"
27   volume_id   = aws_ebs_volume.vol_4_demo.id
28   instance_id = aws_instance.demo-terra.id
29 }
30
31 output "PublicIP" {
32   value = aws_instance.demo-terra.public_ip
33 }
34

```

24. Once you are done setting up your instance. Then run the validate command to validate your code.

25. After that run the plan and apply command to execute your resources.

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

```

# aws_key_pair.demo-terraform-1 will be created
+ resource "aws_key_pair" "demo-terraform-1" {
  + arn      = (known after apply)
  + fingerprint = (known after apply)
  + id       = (known after apply)
  + key_name = "demo-terraform-1"
  + key_name_prefix = (known after apply)
  + key_pair_id = (known after apply)
  + key_type   = (known after apply)
  + public_key = "ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAI03+sM29I9+KJ8fPldt1X3rxQlbfwUMvvRiyCNe60kEa PULKIT@LAPTOP-G2CAKBK8"
  + tags_all  = (known after apply)
}

# aws_volume_attachment.atch_vol_demo will be created
+ resource "aws_volume_attachment" "atch_vol_demo" {
  + device_name = "/dev/xvdh"
  + id          = (known after apply)
  + instance_id = (known after apply)
  + volume_id   = (known after apply)
}

Plan: 4 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ PublicIP = (known after apply)

```

```

aws_key_pair.demo-terraform-1: Creating...
aws_ebs_volume.vol_4_demo: Creating...
aws_key_pair.demo-terraform-1: Creation complete after 1s [id=demo-terraform-1]
aws_instance.demo-terra: Creating...
aws_ebs_volume.vol_4_demo: Still creating... [10s elapsed]
aws_ebs_volume.vol_4_demo: Creation complete after 11s [id=vol-06a344806cbf3e19c]
aws_instance.demo-terra: Still creating... [10s elapsed]
aws_instance.demo-terra: Still creating... [20s elapsed]
aws_instance.demo-terra: Still creating... [30s elapsed]
aws_instance.demo-terra: Creation complete after 31s [id=i-09b39167047e10f2d]
aws_volume_attachment.atch_vol_demo: Creating...
aws_volume_attachment.atch_vol_demo: Still creating... [10s elapsed]
aws_volume_attachment.atch_vol_demo: Still creating... [20s elapsed]
aws_volume_attachment.atch_vol_demo: Creation complete after 21s [id=vai-149348497]

Apply complete! Resources: 4 added, 0 changed, 0 destroyed.

Outputs:

PublicIP = "13.233.79.85"

```

26. Below you can see the instance.

The screenshot shows the AWS CloudWatch Logs Insights interface. At the top, there's a search bar with the query 'HelloWorld'. Below it, a table lists log events for the 'HelloWorld' function:

Time	Log Stream	Event ID	Type	Message
2023-08-15T10:00:00+00:00	lambda.xxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx	INFO	Function started	Function started
2023-08-15T10:00:00+00:00	lambda.xxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx	INFO	Function completed	Function completed

At the bottom, there's a detailed view for the first event:

Event Data:

```

{
    "log": "Function started\n-----\n\n-----\nFunction completed\n-----\n\n-----\n"
}

```

27. Here you can see the storage too, the extra 3 gb that we add and attached to it.

Instance: i-09b39167047e10f2d (my-demo-instance)		=	⋮	X
Root device details				
Root device name /dev/xvda	Root device type EBS		EBS optimization disabled	
Block devices				
<input type="text"/> Filter block devices				
Volume ID	Device name	Volume size (GiB)	Attachment status	Attachment time
vol-0aa7fb8d7c16ff4f2	/dev/xvda	8	Attached	2024/02/25 00:17 GMT+5:30
vol-06a344806cbf3e19c	/dev/xvdh	3	Attached	2024/02/25 00:18 GMT+5:30

28. Once you are done just run the destroy command and destroy all of your resources.