

LAB ASSIGNMENTS 1: Terraform

What is IaC?

As Code (IaC) is a practice where infrastructure is managed and provisioned using code rather than manual processes.

Similar to application code, the infrastructure code is stored in a version control systems (VCS), that ensure infrastructure changes are trackable and scalable.

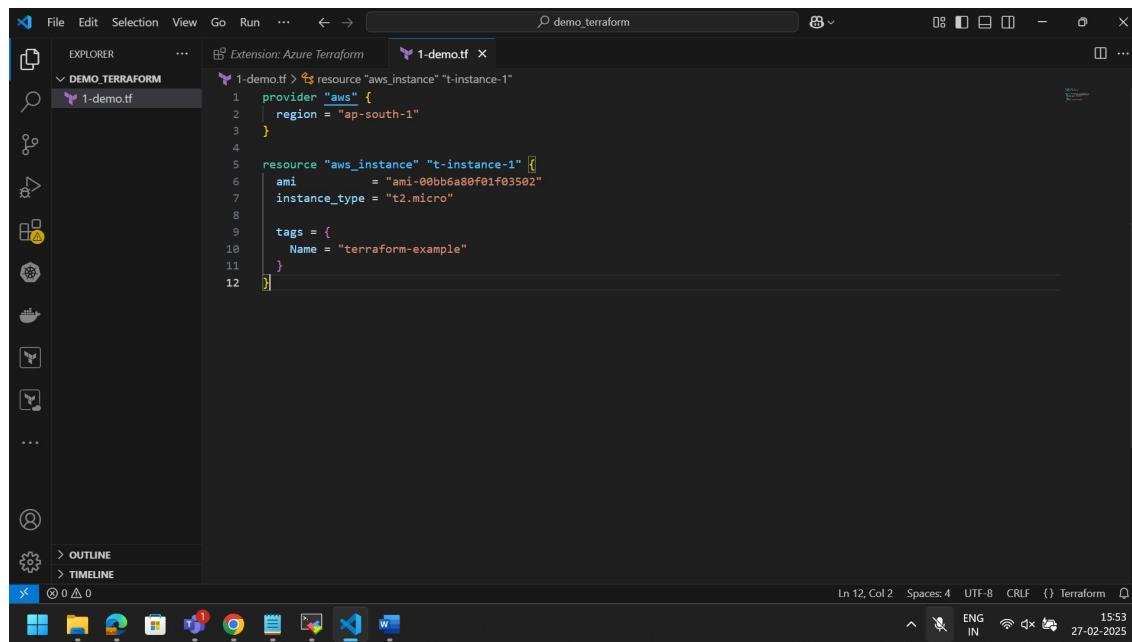
By defining infrastructure in descriptive, machine-readable files,IaC enables automation, consistency, and repeatability while reducing human error.

Why use Terraform?

Declarative approach allows users to define the desired state of their infrastructure, which Terraform then enforces. This capability extends across various cloud providers and on-premises environments, offering flexibility and reducing the complexity of managing multi-cloud or hybrid environments.

Lab assignment

1: creating an ec2 instance of aws



The screenshot shows the Visual Studio Code editor with a dark theme. The left sidebar has a 'DEMO_TERRAFORM' folder containing a file named '1-demo.tf'. The main editor area displays the following Terraform code:

```
provider "aws" {
  region = "ap-south-1"
}

resource "aws_instance" "t-instance-1" [
  ami           = "ami-0ebb6a80f01f03502"
  instance_type = "t2.micro"
  tags = {
    Name = "terraform-example"
  }
]
```

The status bar at the bottom shows the file path as '1-demo.tf', line 12, column 2, and other details like 'Spaces: 4', 'UTF-8', 'CRLF', and the date '27-02-2025'.

```
show      Show the current state or a saved plan
state     Advanced state management
taint     Mark a resource instance as not fully functional
test      Execute integration tests for Terraform modules
untaint   Remove the 'tainted' state from a resource instance
version   Show the current Terraform version
workspace Workspace management

Global options (use these before the subcommand, if any):
  -chdir=DIR  Switch to a different working directory before executing the given subcommand.
  -help       Show this help output, or the help for a specified subcommand.
  -version    An alias for the "version" subcommand.

root@ip-172-31-8-251:/home/ubuntu# terraform version
Terraform v1.10.5
on linux_amd64
root@ip-172-31-8-251:/home/ubuntu# mkdir demo1
root@ip-172-31-8-251:/home/ubuntu# cd demo1/
root@ip-172-31-8-251:/home/ubuntu/demo1# vim main.tf
root@ip-172-31-8-251:/home/ubuntu/demo1# cat main.tf
provider "aws" {
  region = "ap-south-1"
}

resource "aws_instance" "t-instance-1" {
  ami           = "ami-00bb6a80f01f03502"
  instance_type = "t2.micro"

  tags = {
    Name = "terraform-example"
  }
}
root@ip-172-31-8-251:/home/ubuntu/demo1#
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

```
798A EC65 4E5C 1542 8C8E 42EE AA16 FCBC A621 E701
uid [ unknown] HashiCorp Security (HashiCorp Package Signing) <security+packaging@hashicorp.com>
sub rsa4096 2023-01-10 [S] [expires: 2028-01-09]

root@ip-172-31-8-251:/home/ubuntu# echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb_release -cs) main" | \
sudo tee /etc/apt/sources.list.d/hashicorp.list
deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com noble main
root@ip-172-31-8-251:/home/ubuntu# sudo apt update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 https://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:4 https://apt.releases.hashicorp.com noble InRelease [12.9 kB]
Get:5 https://apt.releases.hashicorp.com/noble/main amd64 Packages [169 kB]
Get:6 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [9 012 B]
Get:8 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.0 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:10 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Fetched 370 kB in 2s (238 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
120 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ip-172-31-8-251:/home/ubuntu# sudo apt-get install terraform
Reading package lists... Done
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

15.207.19.95 (ubuntu)

Terminal Sessions View X server Tools Games Settings Macros Help

Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help

Quick connect... /home/ubuntu/ [2.15.207.19.95 (ubuntu)] +

```
provider "aws" {
  region = "ap-south-1"
}

resource "aws_instance" "t-instance-1" {
  ami           = "ami-00bb6a80f01f03502"
  instance_type = "t2.micro"

  tags = {
    Name = "terraform-example"
  }
}
root@ip-172-31-8-251:/home/ubuntu/demo1# terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.88.0...
- Installed hashicorp/aws v5.88.0 (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.
root@ip-172-31-8-251:/home/ubuntu/demo1#
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Remote monitoring allow terminal fold

15:52 27-02-2025

This screenshot shows the MobaXterm interface with a terminal window titled '[2.15.207.19.95 (ubuntu)]'. The user is running a Terraform init command on an Ubuntu system. The output indicates that the provider 'aws' is being initialized, and a lock file '.terraform.lock.hcl' is created to record the provider selections. A message at the bottom encourages users to subscribe to the professional edition.

15.207.19.95 (ubuntu)

Terminal Sessions View X server Tools Games Settings Macros Help

Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help

Quick connect... /home/ubuntu/ [2.15.207.19.95 (ubuntu)] +

```
root@ip-172-31-8-251:/home/ubuntu/demo1#
root@ip-172-31-8-251:/home/ubuntu/demo1# 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:

# aws_instance.t-instance-1 will be created
+ resource "aws_instance" "t-instance-1" {
  + ami           = "ami-00bb6a80f01f03502"
  + arn           = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone      = (known after apply)
  + cpu_core_count         = (known after apply)
  + cpu_threads_per_core   = (known after apply)
  + disable_api_stop       = (known after apply)
  + disable_api_termination = (known after apply)
  + ebs_optimized          = (known after apply)
  + enable_primary_ipv6    = (known after apply)
  + get_password_data      = false
  + host_id              = (known after apply)
  + host_resource_group_arn = (known after apply)
  + iam_instance_profile   = (known after apply)
  + id                   = (known after apply)
  + instance_initiated_shutdown_behavior = (known after apply)
  + instance_lifecycle     = (known after apply)
  + instance_state         = (known after apply)
  + instance_type          = "t2.micro"
  + ipv6_address_count     = (known after apply)
  + ipv6_addresses          = (known after apply)
  + key_name               = (known after apply)
}
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Remote monitoring allow terminal fold

15:53 27-02-2025

This screenshot shows the MobaXterm interface with a terminal window titled '[2.15.207.19.95 (ubuntu)]'. The user is running a Terraform plan command. The output shows that an AWS instance named 't-instance-1' will be created. The plan details the configuration settings for the instance, such as AMI, instance type, and network interfaces.

15.207.19.95 (ubuntu)

Terminal Sessions View X server Tools Games Settings Macros Help

Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help

Quick connect... 2.15.207.19.95 (ubuntu) +

Name /home/ubuntu/

+ ebs_block_device (known after apply)
+ enclave_options (known after apply)
+ ephemeral_block_device (known after apply)
+ instance_market_options (known after apply)
+ maintenance_options (known after apply)
+ metadata_options (known after apply)
+ network_interface (known after apply)
+ private_dns_name_options (known after apply)
+ root_block_device (known after apply)

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_instance.t-t-instance-1: Creating...
aws_instance.t-t-instance-1: Still creating... [10s elapsed]
aws_instance.t-t-instance-1: Creation complete after 12s [id=i-02bb6a5a7f6418402]

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

root@ip-172-31-8-251:/home/ubuntu/demo1#

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Remote monitoring

allow terminal fold

15:54 27-02-2025

This screenshot shows the MobaXterm interface running on an Ubuntu session. It displays the output of a Terraform apply command, which creates a new AWS Lambda instance. The terminal also shows the AWS CloudWatch logs for the newly created instance. The status bar at the bottom indicates an unregistered version of MobaXterm.

Instances | EC2 | ap-south-1

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#instances:v=3;\$case=tags:true%5C,clientfalse;\$regex=tags:false%5C,clientfalse

aws EC2 S3 Elastic Kubernetes Service

EC2 Instances

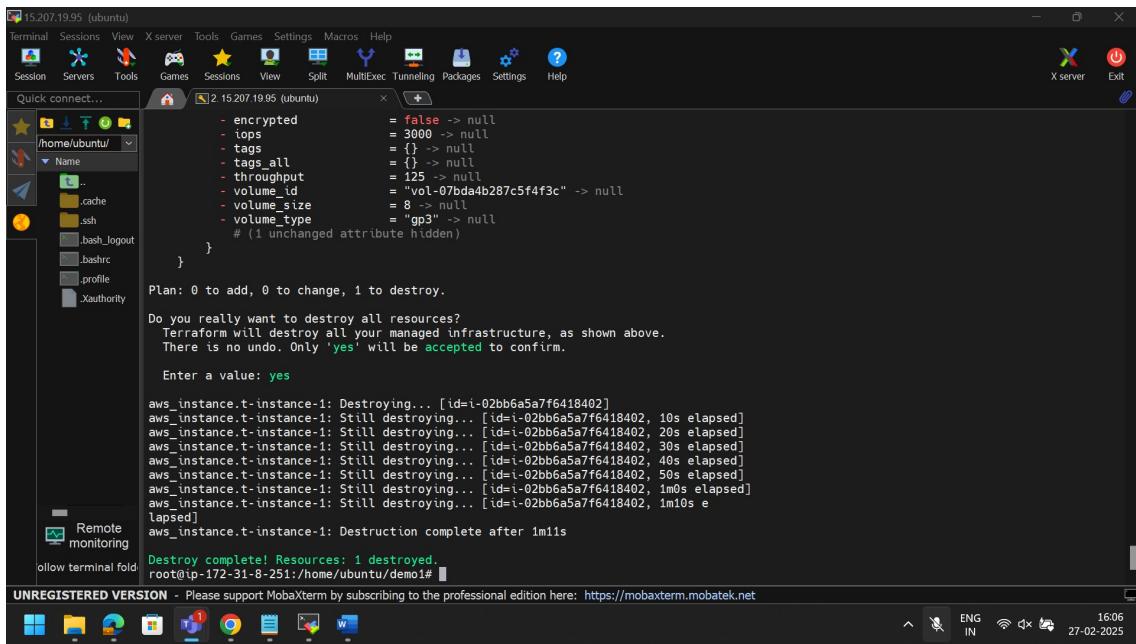
Instances (2) Info Last updated less than a minute ago Connect Instance state Actions Launch instances

Name	Instance ID	Instance state	Instance type	Status check
terraform-example	i-02bb6a5a7f6418402	Running	t2.micro	2/2 checks passed
terraform-client	i-054bc95ab345c21a4	Running	t2.micro	2/2 checks passed

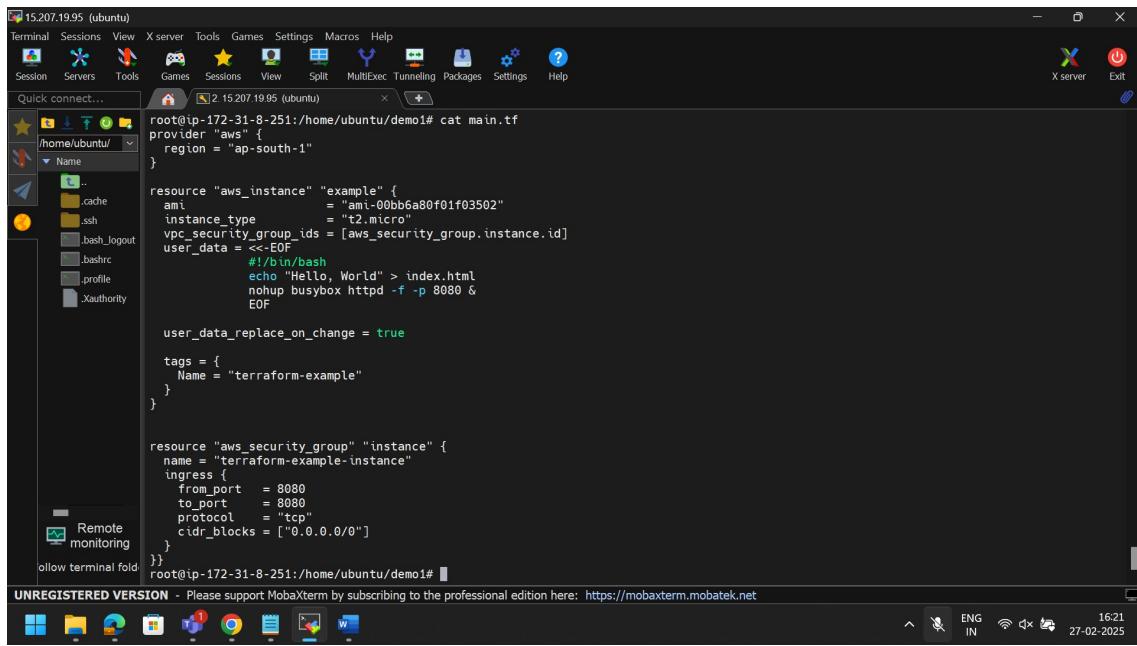
Select an instance

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 15:57 27-02-2025

This screenshot shows the AWS EC2 Instances page. It lists two instances: 'terraform-example' and 'terraform-client', both of which are currently running. The instances are of type t2.micro. The status bar at the bottom indicates the current time as 15:57 on 27-02-2025.



2. creating an web server using terraform



root@ip-172-31-8-251:/home/ubuntu/demo1# cat main.tf

```
provider "aws" {
  region = "ap-south-1"
}

resource "aws_instance" "example" {
  ami           = "ami-00bb6a80f01f03502"
  instance_type = "t2.micro"
  vpc_security_group_ids = [aws_security_group.instance.id]
  user_data = <<-EOF
    #!/bin/bash
    echo "Hello, World" > index.html
    nohup busybox httpd -f -p 8080 &
  EOF

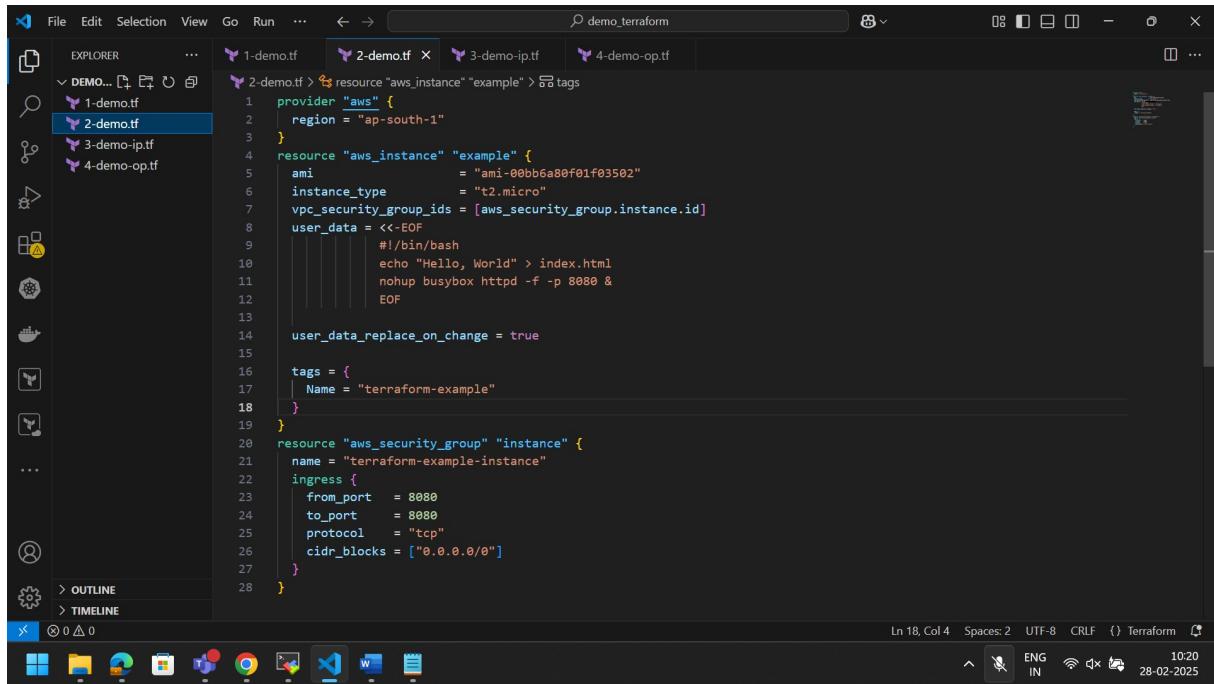
  user_data_replace_on_change = true

  tags = {
    Name = "terraform-example"
  }
}

resource "aws_security_group" "instance" {
  name = "terraform-example-instance"
  ingress {
    from_port   = 8080
    to_port     = 8080
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}
root@ip-172-31-8-251:/home/ubuntu/demo1#
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

16:21 27-02-2025



File Edit Selection View Go Run ... demo_terraform

EXPLORER

- DEMO...
- 1-demo.tf
- 2-demo.tf
- 3-demo-ip.tf
- 4-demo-op.tf

2-demo.tf > resource "aws_instance" "example" > tags

```
provider "aws" {
  region = "ap-south-1"
}

resource "aws_instance" "example" {
  ami           = "ami-00bb6a80f01f03502"
  instance_type = "t2.micro"
  vpc_security_group_ids = [aws_security_group.instance.id]
  user_data = <<-EOF
    #!/bin/bash
    echo "Hello, World" > index.html
    nohup busybox httpd -f -p 8080 &
  EOF

  user_data_replace_on_change = true

  tags = {
    Name = "terraform-example"
  }
}

resource "aws_security_group" "instance" {
  name = "terraform-example-instance"
  ingress {
    from_port   = 8080
    to_port     = 8080
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}
```

Ln 18, Col 4 Spaces: 2 UTF-8 CRLF {} Terraform

10:20 28-02-2025

```

15.207.19.95 (ubuntu)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
X server Exit
Quick connect...
/home/ubuntu/
+ ipv6_cidr_blocks = []
+ prefix_list_ids = []
+ protocol = "tcp"
+ security_groups = []
+ self = false
+ to_port = 8080
    # (1 unchanged attribute hidden)
},
]
+ name = "terraform-example-instance"
+ name_prefix = (known after apply)
+ owner_id = (known after apply)
+ revoke_rules_on_delete = false
+ tags_all = (known after apply)
+ vpc_id = (known after apply)
}

Plan: 2 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.instance: Creating...
aws_security_group.instance: Creation complete after 2s [id=sg-04a452e704ac12496]
aws_instance.example: Creating...
aws_instance.example: Still creating... [10s elapsed]
aws_instance.example: Creation complete after 12s [id=i-09e10fa955cac9b35]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
root@ip-172-31-8-251:/home/ubuntu/demo1# 

```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Instances | EC2 | ap-south-1

aws EC2 S3 Elastic Kubernetes Service

EC2 > Instances

Name	Instance ID	Instance state	Instance type	Status check
terraform-example	i-02bb6a5a7f6418402	Terminated	t2.micro	-
terraform-client	i-054bc95ab345c21a4	Running	t2.micro	2/2 checks passed
terraform-example	i-09e10fa955cac9b35	Running	t2.micro	Initializing

Select an instance

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 16:26 27-02-2025

The screenshot shows the AWS Cloud9 IDE interface. The code editor contains the following Python script:

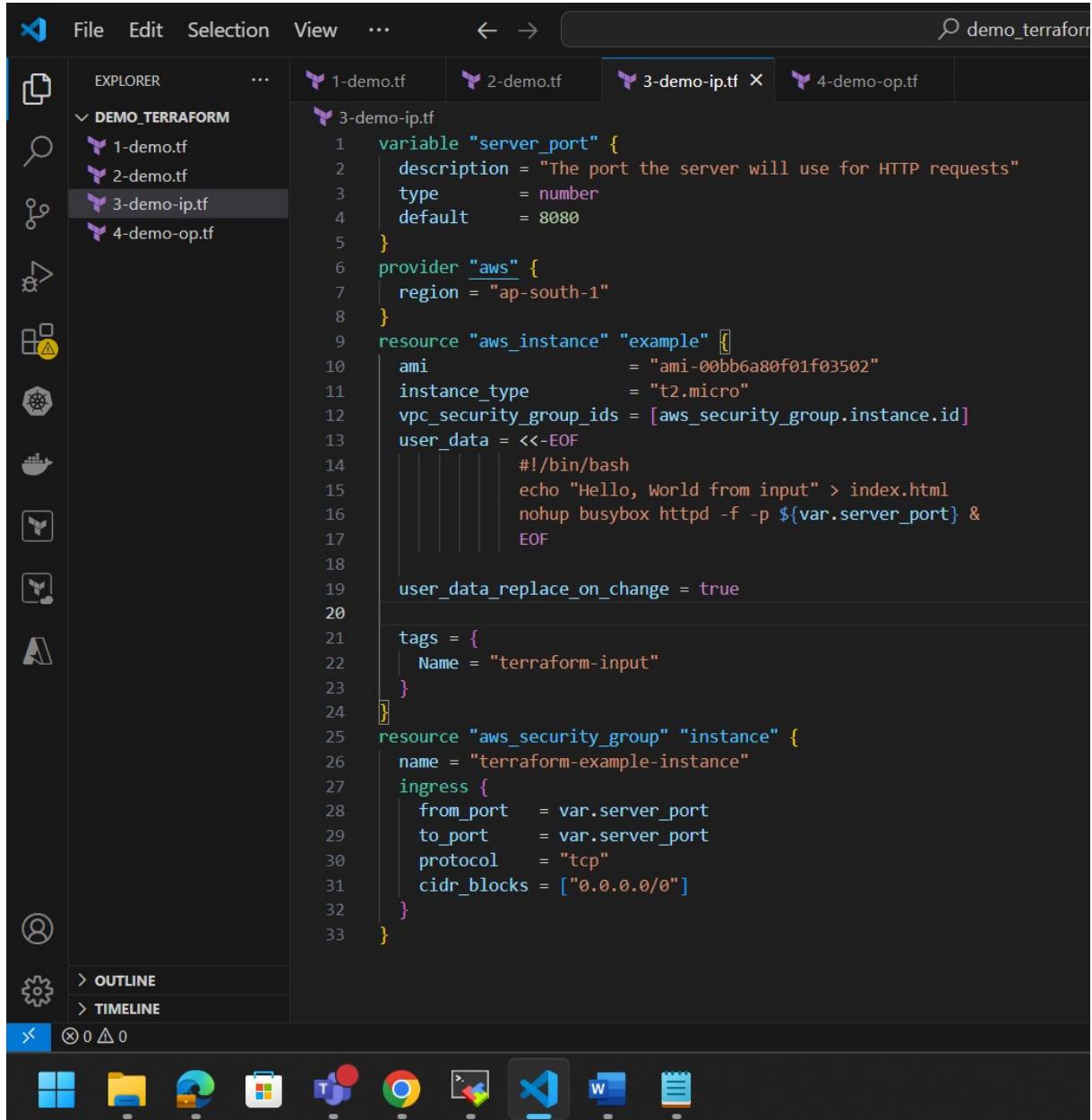
```
print("Hello, World")
```

Hello, World



3. Apply terraform input/output variables to assignment 2 so that we dont have hard coding of port numbers

Input



```
variable "server_port" {
  description = "The port the server will use for HTTP requests"
  type        = number
  default     = 8080
}

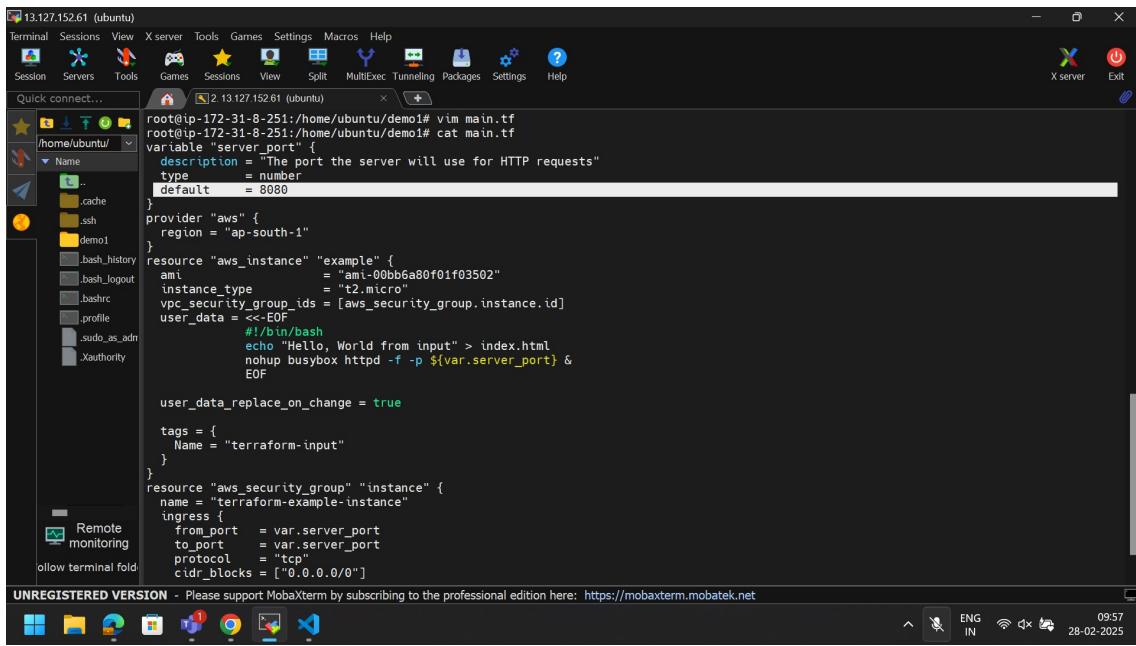
provider "aws" {
  region = "ap-south-1"
}

resource "aws_instance" "example" {
  ami           = "ami-00bb6a80f01f03502"
  instance_type = "t2.micro"
  vpc_security_group_ids = [aws_security_group.instance.id]
  user_data = <<-EOF
    #!/bin/bash
    echo "Hello, World from input" > index.html
    nohup busybox httpd -f -p ${var.server_port} &
  EOF

  user_data_replace_on_change = true

  tags = {
    Name = "terraform-input"
  }
}

resource "aws_security_group" "instance" {
  name = "terraform-example-instance"
  ingress {
    from_port   = var.server_port
    to_port     = var.server_port
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}
```

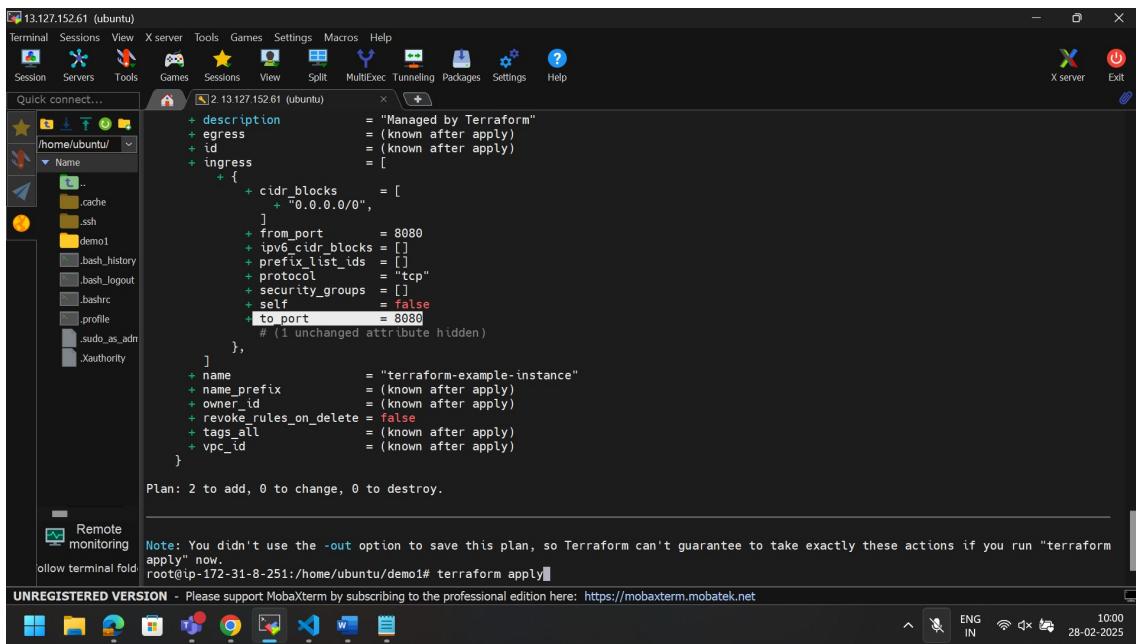


```
root@ip-172-31-8-251:/home/ubuntu/demo1# vim main.tf
root@ip-172-31-8-251:/home/ubuntu/demo1# cat main.tf
variable "server_port" {
  description = "The port the server will use for HTTP requests"
  type        = number
  default     = 8080
}
provider "aws" {
  region = "ap-south-1"
}
resource "aws_instance" "example" {
  ami           = "ami-00bb6a80f01f03502"
  instance_type = "t2.micro"
  vpc_security_group_ids = [aws_security_group.instance.id]
  user_data = <><EOF>
    #!/bin/bash
    echo "Hello, World from input" > index.html
    nohup busybox httpd -f -p ${var.server_port} &
    EOF
  user_data_replace_on_change = true

  tags = {
    Name = "terraform-input"
  }
}
resource "aws_security_group" "instance" {
  name = "terraform-example-instance"
  ingress {
    from_port   = var.server_port
    to_port     = var.server_port
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
```

Terraform plan command gives 8080



```
+ description      = "Managed by Terraform"
+ egress          = (known after apply)
+ id              = (known after apply)
+ ingress         = [
+   {
+     + cidr_blocks  = [
+       + "0.0.0.0/0",
+     ]
+     + from_port    = 8080
+     + ipv6_cidr_blocks = []
+     + prefix_list_ids = []
+     + protocol     = "tcp"
+     + security_groups = []
+     + self         = false
+     + to_port      = 8080
+   },
+ ],
+ name            = "terraform-example-instance"
+ name_prefix     = (known after apply)
+ owner_id        = (known after apply)
+ revoke_rules_on_delete = false
+ tags_all        = (known after apply)
+ vpc_id          = (known after apply)

Plan: 2 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.
root@ip-172-31-8-251:/home/ubuntu/demo1# terraform apply
```

The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main area displays a table of instances:

Name	Instance ID	Instance state	Instance type	Status check
terraform-client	i-054bc95ab345c21a4	Running	t2.micro	2/2 checks passed
terraform-input	i-070ddc00f13eb4260	Running	t2.micro	Initializing

Below the table, a message says "Select an instance". The status bar at the bottom right shows "10:01 28-02-2025".

Running on port 8080

The screenshot shows a browser window with the URL "35.154.213.51:8080". The page content is "Hello, World from input". The status bar at the bottom right shows "10:02 28-02-2025".

Ran `terraform plan -var="server_port=8090"`

So port changed to 8090

```

2.13.127.152.61 (ubuntu)
/home/ubuntu/terraform-example-instance.tf
[...]
resource "aws_instance" "example" {
  ami           = "ami-070ddc00f13eb4260"
  instance_type = "t2.micro"
  key_name      = "tf-key-pair"
  security_groups = []
  self         = false
  to_port     = 8090
  # (1 unchanged attribute hidden)

  + cidr_blocks  = [
    + "0.0.0.0/0",
  ]
  + from_port    = 8090
  + ipv6_cidr_blocks = []
  + prefix_list_ids = []
  + protocol     = "tcp"
  + security_groups = []
  + self         = false
  + to_port     = 8090
  # (1 unchanged attribute hidden)

  ],
  name          = "terraform-example-instance"
  tags          = {}
  # (8 unchanged attributes hidden)
}

Plan: 1 to add, 1 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.
root@ip-172-31-8-251:/home/ubuntu/demo1# 

```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Then `terraform apply -var="server_port=8090"`

```

2.13.127.152.61 (ubuntu)
/home/ubuntu/terraform-example-instance.tf
[...]
resource "aws_instance" "example" {
  ami           = "ami-070ddc00f13eb4260"
  instance_type = "t2.micro"
  key_name      = "tf-key-pair"
  security_groups = []
  self         = false
  to_port     = 8090
  # (1 unchanged attribute hidden)

  ],
  name          = "terraform-example-instance"
  tags          = {}
  # (8 unchanged attributes hidden)
}

Plan: 1 to add, 1 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

aws_instance.example: Destroying... [id=i-070ddc00f13eb4260]
aws_instance.example: Still destroying... [id=i-070ddc00f13eb4260, 10s elapsed]
aws_instance.example: Still destroying... [id=i-070ddc00f13eb4260, 20s elapsed]
aws_instance.example: Still destroying... [id=i-070ddc00f13eb4260, 30s elapsed]
aws_instance.example: Destruction complete after 40s
aws_security_group.instance: Modifying... [ids=sg-001db00e0133745c4]
aws_security_group.instance: Modifications complete after 1s [id=sg-001db00e0133745c4]
aws_instance.example: Creating...
aws_instance.example: Still creating... [10s elapsed]
aws_instance.example: Creation complete after 12s [id=i-06a63cf7c94bf49f]

Apply complete! Resources: 1 added, 1 changed, 1 destroyed.
root@ip-172-31-8-251:/home/ubuntu/demo1# 

```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Added: new instance

Changes: port number variable

Destroyed: new instance

The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main area displays a table of instances:

Name	Instance ID	Instance state	Instance type	Status check
terraform-client	i-054bc95ab345c21a4	Running	t2.micro	2/2 checks passed
terraform-input	i-06a63cf7c94bf49f	Pending	t2.micro	-
terraform-input	i-070ddc00f13eb4260	Terminated	t2.micro	-

The instance **i-070ddc00f13eb4260 (terraform-input)** is selected. Below it, the **Instance summary** tab is active. The bottom right corner shows the date and time: 28-02-2025.

The screenshot shows a browser window with the URL `13.201.100.150:8090`. The page content is:

Hello, World from input

The bottom right corner shows the date and time: 28-02-2025.

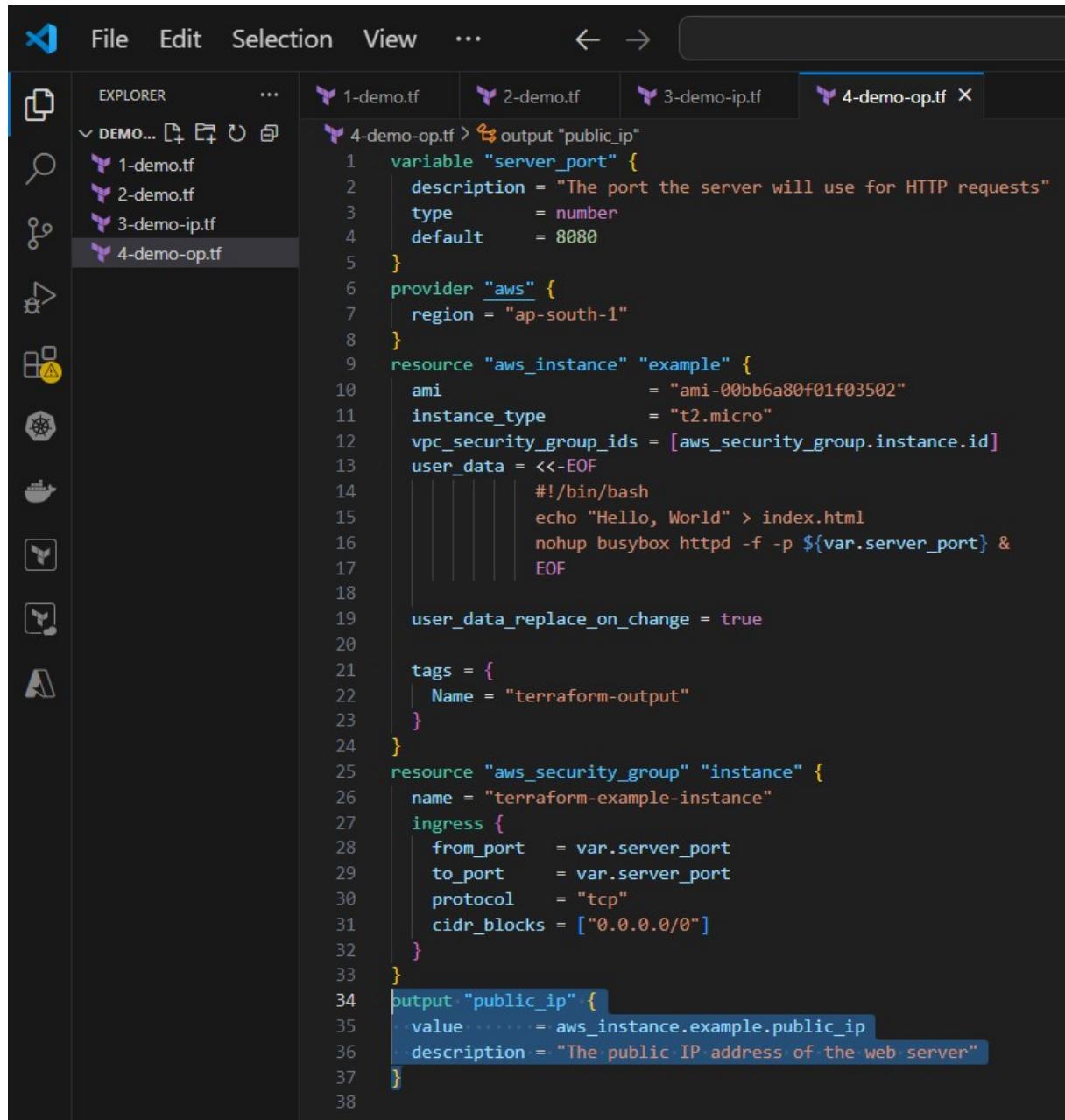
So now it runs on port 8090

We can also create a separate variables.tf file for variable “server_port”
vim variables.tf

```
-----  
variable "server_port" {  
  description = "The port the server will use for HTTP requests"  
  type       = number  
  default    = 8080  
}
```

terraform apply -var-file

Output variables



```
File Edit Selection View ... ← →   
EXPLORER 1-demo.tf 2-demo.tf 3-demo-ip.tf 4-demo-op.tf  
DEMO... 1-demo.tf 2-demo.tf 3-demo-ip.tf 4-demo-op.tf  
1 variable "server_port" {  
  description = "The port the server will use for HTTP requests"  
  type       = number  
  default    = 8080  
}  
2 provider "aws" {  
  region = "ap-south-1"  
}  
3 resource "aws_instance" "example" {  
  ami           = "ami-00bb6a80f01f03502"  
  instance_type = "t2.micro"  
  vpc_security_group_ids = [aws_security_group.instance.id]  
  user_data = <<-EOF  
14          #!/bin/bash  
15          echo "Hello, World" > index.html  
16          nohup busybox httpd -f -p ${var.server_port} &  
17          EOF  
18  
19  user_data_replace_on_change = true  
20  
21  tags = {  
22    Name = "terraform-output"  
23  }  
24}  
25 resource "aws_security_group" "instance" {  
26  name = "terraform-example-instance"  
27  ingress {  
28    from_port   = var.server_port  
29    to_port     = var.server_port  
30    protocol    = "tcp"  
31    cidr_blocks = ["0.0.0.0/0"]  
32  }  
33}  
34 output "public_ip" {  
35  value      = aws_instance.example.public_ip  
36  description = "The public IP address of the web server"  
37}  
38
```

The screenshot shows the MobaXterm interface with a terminal window titled '2. 15.207.19.95 (ubuntu)'. The terminal displays the contents of a Terraform configuration file ('main.tf'):

```
root@ip-172-31-8-251:/home/ubuntu/demo1# rm main.tf
root@ip-172-31-8-251:/home/ubuntu/demo1# vim main.tf
root@ip-172-31-8-251:/home/ubuntu/demo1# cat main.tf
variable "server_port" {
  description = "The port the server will use for HTTP requests"
  type        = number
  default     = 8080
}
provider "aws" {
  region = "ap-south-1"
}
resource "aws_instance" "example" {
  ami           = "ami-00bb6a80f01f03502"
  instance_type = "t2.micro"
  vpc_security_group_ids = [aws_security_group.instance.id]
  user_data = <>-EOF
  #!/bin/bash
  echo "Hello, World" > index.html
  nohup busybox httpd -f -p ${var.server_port} &
  EOF

  user_data_replace_on_change = true

  tags = {
    Name = "terraform-output"
  }
}
resource "aws_security_group" "instance" {
  name = "terraform-example-instance"
  ingress {
    from_port  = var.server_port
    to_port    = var.server_port
    protocol   = "tcp"
  }
}
```

Below the terminal, a message reads: 'UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>'.

The screenshot shows the MobaXterm interface with a terminal window titled '2. 15.207.19.95 (ubuntu)'. The terminal displays the output of the 'terraform plan' command:

```
}, ],
+ name          = "terraform-example-instance"
+ name_prefix   = (known after apply)
+ owner_id      = (known after apply)
+ revoke_rules_on_delete = false
+ tags_all      = (known after apply)
+ vpc_id         = (known after apply)
}

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

Changes to Outputs:
+ public_ip = (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

aws_security_group.instance: Creating...
aws_group.instance: Creation complete after 1s [id=sg-0edbe93ac3f40f106]
aws_instance.example: Creating...
aws_instance.example: Still creating... [10s elapsed]
aws_instance.example: Creation complete after 13s [id=i-01acc0f5ab277f2f]

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

Outputs:
public_ip = "3.108.191.59"
root@ip-172-31-8-251:/home/ubuntu/demo1#
```

Below the terminal, a message reads: 'UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>'.

We can see the public_ip on the output in the shell.



A screenshot of the AWS EC2 Instances page. The left sidebar shows navigation options like Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, and Images. The main content area shows a table of instances with columns: Name, Instance ID, Instance state, Instance type, Status check, and Alarm. Four instances are listed: 'terraform-example' (terminated), 'terraform-client' (running), 'terraform-example' (terminated), and 'terraform-output' (running). The 'terraform-output' row is selected. Below the table, a detailed view for 'i-01acc0f5ab277f2f (terraform-output)' is shown with tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. The 'Details' tab is selected. At the bottom, there are links for CloudShell and Feedback, and standard footer links for Privacy, Terms, and Cookie preferences.

4. Create AWS IAM User and Policy using terraform as discussed in the session

The screenshot shows a MobaXterm window with two terminal sessions. The left session (root@ip-172-31-8-251:/home/ubuntu/demo1#) displays the creation of three IAM users: raj_bca_user, ekta_bca_user, and gun_bca_user. The right session (root@ip-172-31-8-251:/home/ubuntu/demo1#) shows the output of these users and the ARNs for the ec2 and s3 policies.

```
root@ip-172-31-8-251:/home/ubuntu/demo1# vim outputs.tf
root@ip-172-31-8-251:/home/ubuntu/demo1# vim main.tf
root@ip-172-31-8-251:/home/ubuntu/demo1# cat outputs.tf
output "raj_bca_user" {
    value = aws_iam_user.raj_bca.name
}

output "ekta_bca_user" {
    value = aws_iam_user.ekta_bca.name
}

output "gun_bca_user" {
    value = aws_iam_user.gun_bca.name
}

output "ec2_list_policy_arn" {
    value = aws_iam_policy.ec2_list.arn
}

output "s3_list_policy_arn" {
    value = aws_iam_policy.s3_list.arn
}

root@ip-172-31-8-251:/home/ubuntu/demo1#
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Windows taskbar icons are visible at the bottom of the screen.

```
root@ip-172-31-8-251:/home/ubuntu/demo1# cat main.tf
provider "aws" {
  region = "ap-south-1"
}

# creating users iam
resource "aws_iam_user" "raj_bca" {
  name = "raj-bca"
}

resource "aws_iam_user" "ekta_bca" {
  name = "ekta-bca"
}

resource "aws_iam_user" "gun_bca" {
  name = "gun-bca"
}

# policy for raj bca ec2
resource "aws_iam_policy" "ec2_list" {
  name      = "ec2-list-policy"
  description = "Allows EC2 list actions"
  policy    = jsonencode({
    Version = "2012-10-17"
    Statement = [
      {
        Action   = "ec2:DescribeInstances"
        Effect   = "Allow"
        Resource = "*"
      }
    ]
  })
}

# s3 policy for ekta
resource "aws_iam_policy" "s3_list" {
  name      = "s3-list-policy"
  description = "Allows S3 list actions"
  policy    = jsonencode({
    Version = "2012-10-17"
    Statement = [
      {
        Action   = "s3>ListBucket"
        Effect   = "Allow"
        Resource = "*"
      }
    ]
  })
}

# attaching the policies here
resource "aws_iam_policy_attachment" "raj_bca_policy_attachment" {
  name      = "raj-bca-policy-attachment"
  users     = [aws_iam_user.raj_bca.name]
  policy_arn = aws_iam_policy.ec2_list.arn
}

resource "aws_iam_policy_attachment" "ekta_bca_policy_attachment" {
  name      = "ekta-bca-policy-attachment"
  users     = [aws_iam_user.ekta_bca.name]
  policy_arn = aws_iam_policy.s3_list.arn
}

# default policy for gun
resource "aws_iam_policy_attachment" "gun_bca_policy_attachment" {
  name      = "gun-bca-policy-attachment"
}

SION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
```

The terraform files are above

Doing terraform init,plan,apply

```
65.1.134.151 (ubuntu)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect... 2.65.1.134.151 (ubuntu) 3.65.1.134.151 (ubuntu) +
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_iam_user.raj_bca: Creating...
aws_iam_policy.s3_list: Creating...
aws_iam_user.ekta_bca: Creating...
aws_iam_user.gun_bca: Creating...
aws_iam_policy.ec2_list: Creating...
aws_iam_user.ekta_bca: Creation complete after 1s [id=ekta-bca]
aws_iam_user.gun_bca: Creation complete after 1s [id=gun-bca]
aws_iam_policy_attachment.gun_bca_policy_attachment: Creating...
aws_iam_user.raj_bca: Creation complete after 1s [id=raj-bca]
aws_iam_policy.ec2_list: Creation complete after 1s [id=arn:aws:iam::396608804235:policy/ec2-list-policy]
aws_iam_policy_attachment.raj_bca_policy_attachment: Creating...
aws_iam_policy.s3_list: Creation complete after 1s [id=arn:aws:iam::396608804235:policy/s3-list-policy]
aws_iam_policy_attachment.ekta_bca_policy_attachment: Creating...
aws_iam_policy_attachment.gun_bca_policy_attachment: Creating...
aws_iam_policy_attachment.raj_bca_policy_attachment: Creation complete after 0s [id=raj-bca-policy-attachment]
aws_iam_policy_attachment.ekta_bca_policy_attachment: Creation complete after 0s [id=ekta-bca-policy-attachment]

Apply complete! Resources: 8 added, 0 changed, 0 destroyed.

Outputs:

ec2_list_policy_arn = "arn:aws:iam::396608804235:policy/ec2-list-policy"
ekta_bca_user = "ekta-bca"
gun_bca_user = "gun-bca"
raj_bca_user = "raj-bca"
s3_list_policy_arn = "arn:aws:iam::396608804235:policy/s3-list-policy"
root@ip-172-31-8-251:/home/ubuntu/demo1#
```

IAM users created in AWS

The screenshot shows the AWS IAM Users page. The left sidebar has 'IAM' selected under 'Identity and Access Management (IAM)'. The main area displays a table of users:

User name	Path	Group	Last activity	MFA	Password age
bca-admin-rahu	/	0	34 minutes ago	-	4 days
ekta-bca	/	0	-	-	-
gun-bca	/	0	-	-	-
raj-bca	/	0	-	-	-

At the bottom, there are links for 'CloudShell' and 'Feedback'.

Policy applied to raj-bca : ec2-list-policy

The screenshot shows the AWS IAM console for the user 'raj-bca'. The 'Permissions' tab is selected. A single policy, 'ec2-list-policy', is listed under 'Permissions policies (1)'. This policy is of type 'Customer managed' and is attached 'Directly'. The ARN of the user is listed as 'arn:aws:iam::396608804235:user/raj-bca'. The user was created on March 01, 2025, at 00:31 (UTC+05:30). The 'Console access' status is 'Disabled'.

S3 for ekta-bca

The screenshot shows the AWS IAM console for the user 'ekta-bca'. The 'Permissions' tab is selected. A single policy, 's3-list-policy', is listed under 'Permissions policies (1)'. This policy is of type 'Customer managed' and is attached 'Directly'. The ARN of the user is listed as 'arn:aws:iam::396608804235:user/ekta-bca'. The user was created on March 01, 2025, at 00:31 (UTC+05:30). The 'Console access' status is 'Disabled'.

Read only for gun-bca

The screenshot shows the AWS IAM User details page for the user 'gun-bca'. The user was created on March 01, 2025, at 00:31 UTC+05:30. There is no last console sign-in information. The 'Permissions' tab is selected, showing one attached policy named 'ReadOnlyAccess'. This policy is an AWS managed job function attached directly to the user. The 'Permissions boundary' is listed as '(not set)'. The left sidebar shows navigation options for IAM, Access management, and Access reports.

Identity and Access Management (IAM)

Created
March 01, 2025, 00:31 (UTC+05:30)

Last console sign-in
-

Permissions Groups Tags Security credentials Last Accessed

Permissions policies (1)

Permissions are defined by policies attached to the user directly or through groups.

Filter by Type

Policy name ▾ Type Attached via ▾

ReadOnlyAccess AWS managed - job function Directly

Permissions boundary (not set)

CloudShell Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 00:35 01-03-2025