

Lab Assignment 4

Rishi (8972657)

Scripting and Automation

PROG8830 - Winter 2025 - Section 1 Prof. Mike Dabydeen

Jan 18, 2025

1. Initialize Terraform: Open a terminal window and navigate to the directory where you have your Terraform configuration file (e.g., main.tf). Run the following command to initialize Terraform:

```
terraform init
```

The screenshot shows a code editor with a Terraform configuration file named rp2657.tf. The file contains code for setting up an AWS VPC, including provider configurations, data sources, and resources like an SSM parameter, VPC, Internet Gateway, and subnet. To the right of the code editor is a terminal window showing the command terraform init being run in a PowerShell session. The terminal output indicates that Terraform is initializing the backend, reusing previous versions of provider plugins, and successfully initializing the configuration.

```
lab4 > cd rp2657_tf & terraform init
provider "aws" {
  region = "us-east-1"
  access_key = "AKIAINOMQNLJGAZ2BY3B"
  secret_key = "qd+kN3ayAtz6iaFOGhcwv4vsS3Ov0IZg/trKHR"
}

data "aws_ssm_parameter" "amzn2_linux" {
  name = "/aws/service/amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2"
}

resource "aws_vpc" "main_vpc" {
  cidr_block = "10.0.0.0/16"
  tags = [
    { Name = "MainVPC" }
  ]
}

resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.main_vpc.id
  tags = [
    { Name = "MainIGN" }
  ]
}

resource "aws_subnet" "public_subnet" {
  vpc_id = aws_vpc.main_vpc.id
  cidr_block = "10.0.0.0/24"
  map_public_ip_on_launch = true
  availability_zone = "us-east-1a"
  tags = [
    { Name = "public_subnet" }
  ]
}

resource "aws_route_table" "public_rt" {
  vpc_id = aws_vpc.main_vpc.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.gw.id
  }
  tags = [
    { Name = "PublicRouteTable" }
  ]
}

TERRMINAL
PS C:\Users\patel\OneDrive\Documents\GitHub\automation-class\lab4> terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.86.0

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 C:\Users\patel\OneDrive\Documents\GitHub\automation-class\lab4>
```

2. Provision Infrastructure: Once you have written the Terraform configuration, run the following command to preview the changes that Terraform will make:

terraform plan

```

File Edit Selection View Go Run Terminal Help ⏎ → automation-class
lab4 > "rp2657.tf 2.U"
provider "aws" {
  region      = "us-east-1"
  access_key  = "AKIAVMONQNLJGA2YBY3B"
  secret_key  = "gd+kN3eYat6JafOPGcvw4vsSJOv0IZg/trKIR"
}

data "aws_ssm_parameter" "amzn2_linux" {
  name = "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2"
}

resource "aws_vpc" "main_vpc" {
  cidr_block = "10.0.0.0/16"
  tags = [
    { Name = "MainVPC" }
  ]
}

resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.main_vpc.id
  tags = [
    { Name = "MainIGN" }
  ]
}

resource "aws_subnet" "public_subnet" {
  vpc_id      = aws_vpc.main_vpc.id
  cidr_block  = "10.0.0.0/24"
  map_public_ip_on_launch = true
  availability_zone   = "us-east-1a"
  tags = [
    { Name = "public_subnet" }
  ]
}

resource "aws_route_table" "public_rt" {
  vpc_id = aws_vpc.main_vpc.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.gw.id
  }
  tags = [
    { Name = "PublicRouteTable" }
  ]
}

data "aws_ssm_parameter" "amzn2_linux" {
  name = "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2"
}

resource "aws_vpc" "main_vpc" {
  cidr_block = "10.0.0.0/16"
  tags = [
    { Name = "MainVPC" }
  ]
}

resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.main_vpc.id
  tags = [
    { Name = "MainIGN" }
  ]
}

resource "aws_subnet" "public_subnet" {
  vpc_id      = aws_vpc.main_vpc.id
  cidr_block  = "10.0.0.0/24"
  map_public_ip_on_launch = true
  availability_zone   = "us-east-1a"
  tags = [
    { Name = "public_subnet" }
  ]
}

resource "aws_route_table" "public_rt" {
  vpc_id = aws_vpc.main_vpc.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.gw.id
  }
  tags = [
    { Name = "PublicRouteTable" }
  ]
}

```

TERMINAL

```

PS C:\Users\patel\OneDrive\Documents\GitHub\automation-class\lab4> terraform fmt
PS C:\Users\patel\OneDrive\Documents\GitHub\automation-class\lab4> terraform plan
data.aws_ssm_parameter.amzn2_linux: Reading...
data.aws_ssm_parameter.amzn2_linux: Read complete after 1s [id=/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2]

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

Terraform will perform the following actions:

# aws_instance.nginx1 will be created
+ resource "aws_instance" "nginx1" {
  + ami                                = (sensitive value)
  + arm                                = (known after apply)
  + associate_public_ip_address         = true
  + 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)
  + ec2_eni_id                         = (known after apply)
  + enable_ip_forwarding_ipv6          = false
  + get_password_data                 = (known after apply)
  + 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                     = "t3.micro"
  + ipv6_address_count                = (known after apply)
  + ipv6_addresses                     = (known after apply)
  + key_name                           = (known after apply)
  + monitoring                        = (known after apply)
  + outpost_arn                       = (known after apply)
  + password_data                     = (known after apply)
  + placement_group                   = (known after apply)
  + placement_partition_number        = (known after apply)
  + primary_network_interface_id     = (known after apply)
  + private_dns                        = (known after apply)
  + private_ip                         = (known after apply)
  + public_dns                         = (known after apply)
  + public_ip                          = (known after apply)
  + secondary_private_ips             = (known after apply)
  + security_groups                   = (known after apply)
  + source_dest_check                 = true
  + spot_instance_request_id          = (known after apply)
}
```

CHAT

ln 25, Col 1 Spaces: 4 LF {} Terraform Go Live Prettier 411PM 2/8/2025

```

File Edit Selection View Go Run Terminal Help ⏎ → automation-class
lab4 > "rp2657.tf 2.U"
provider "aws" {
  region      = "us-east-1"
  access_key  = "AKIAVMONQNLJGA2YBY3B"
  secret_key  = "gd+kN3eYat6JafOPGcvw4vsSJOv0IZg/trKIR"
}

data "aws_ssm_parameter" "amzn2_linux" {
  name = "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2"
}

resource "aws_vpc" "main_vpc" {
  cidr_block = "10.0.0.0/16"
  tags = [
    { Name = "MainVPC" }
  ]
}

resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.main_vpc.id
  tags = [
    { Name = "MainIGN" }
  ]
}

resource "aws_subnet" "public_subnet" {
  vpc_id      = aws_vpc.main_vpc.id
  cidr_block  = "10.0.0.0/24"
  map_public_ip_on_launch = true
  availability_zone   = "us-east-1a"
  tags = [
    { Name = "public_subnet" }
  ]
}

resource "aws_route_table" "public_rt" {
  vpc_id = aws_vpc.main_vpc.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.gw.id
  }
  tags = [
    { Name = "PublicRouteTable" }
  ]
}

data "aws_ssm_parameter" "amzn2_linux" {
  name = "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2"
}

resource "aws_vpc" "main_vpc" {
  cidr_block = "10.0.0.0/16"
  tags = [
    { Name = "MainVPC" }
  ]
}

resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.main_vpc.id
  tags = [
    { Name = "MainIGN" }
  ]
}

resource "aws_subnet" "public_subnet" {
  vpc_id      = aws_vpc.main_vpc.id
  cidr_block  = "10.0.0.0/24"
  map_public_ip_on_launch = true
  availability_zone   = "us-east-1a"
  tags = [
    { Name = "public_subnet" }
  ]
}

resource "aws_route_table" "public_rt" {
  vpc_id = aws_vpc.main_vpc.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.gw.id
  }
  tags = [
    { Name = "PublicRouteTable" }
  ]
}

```

TERMINAL

```

# aws_vpc.main_vpc will be created
resource "aws_vpc" "main_vpc" {
  + enable_resource_name_dns_a_record_on_launch = false
  + enable_resource_name_dns_aaaa_record_on_launch = false
  + id                                         = (known after apply)
  + ipv6_cidr_block_association_id            = (known after apply)
  + ipv6_native                               = true
  + map_public_ip_on_launch                  = (known after apply)
  + owner_id                                  = (known after apply)
  + private_dns_hostname_type_on_launch       = (known after apply)
  + tags {
      + "Name" = "public_subnet"
    }
  + tags_all {
      + "Name" = "public_subnet"
    }
  + vpc_id                                     = (known after apply)
}

# Plan: 7 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 C:\Users\patel\OneDrive\Documents\GitHub\automation-class\lab4>

```

CHAT

ln 25, Col 1 Spaces: 4 LF {} Terraform Go Live Prettier 411PM 2/8/2025

3. Apply Changes: If the plan looks correct, apply the changes to provision the server in AWS:

`terraform apply`

The screenshot shows a code editor with a left pane containing Terraform code and a right pane showing AWS CloudWatch logs.

Terraform Code (Left Pane):

```
File Edit Selection View Go Run Terminal Help ← → automation-class

lab4 > rp2657142.0 ✘
lab4 > rp2657142.0 ✘ resource "aws_subnet" "public_subnet"
1 provider "aws" {
2   region     = "us-east-1"
3   access_key = "AKIAJWVQWVQLQGA2YBY38"
4   secret_key = "gdd+kCDeAzt6zAfCPGhcuvw4vsSJ0v0IZg/trKRa"
5 }
6
7 data "aws_ssm_parameter" "amzn2_linux" {
8   name = "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2"
9 }
10
11 resource "aws_vpc" "main_vpc" {
12   cidr_block = "10.0.0.0/16"
13   tags = [
14     { Name = "MainVPC" }
15   ]
16 }
17
18 resource "aws_internet_gateway" "gw" {
19   vpc_id = aws_vpc.main_vpc.id
20
21   tags = [
22     { Name = "MainIGW" }
23   ]
24 }
25
26 resource "aws_subnet" "public_subnet" {
27   vpc_id          = aws_vpc.main_vpc.id
28   cidr_block      = "10.0.0.0/24"
29   map_public_ip_on_launch = true
30   availability_zone = "us-east-1a"
31
32   tags = [
33     { Name = "public_subnet" }
34   ]
35 }
36
37 resource "aws_route_table" "public_rt" {
38   vpc_id = aws_vpc.main_vpc.id
39
40   route {
41     cidr_block = "0.0.0.0/0"
42     gateway_id = aws_internet_gateway.gw.id
43   }
44
45   tags = [
46     { Name = "PublicRouteTable" }
47   ]
48 }
```

CloudWatch Logs (Right Pane):

```
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            = (known after apply)
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" = "MainVPC"
}
tags_all                         = {
  + "Name" = "MainVPC"
}

Plan: 7 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_vpc.main_vpc: Creating...
aws_vpc.main_vpc: Creation complete after 2s [id=vpc-06c4934b2bd2a436d]
aws_internet_gateway.gw: Creating...
aws_subnet.public_subnet: Creating...
aws_security_group.web_sg: Creating...
aws_internet_gateway.gateway: Creation complete after 0s [id=igw-0963266ea6ca0342]
aws_route_table.public_rt: Creating...
aws_route_table.public_rt: Creation complete after 1s [id=rtb-0d39e701693e10f8d]
aws_security_group.web_sg: Creation complete after 0s [id=sgr-8d436cd9596fa577]
aws_subnet.public_subnet: Still creating... [10s elapsed]
aws_subnet.public_subnet: Creation complete after 1s [id=subnet-0f706ef28e79a967e]
aws_route_table_association.public_assoc: Creating...
aws_instance.nginx1: Creating...
aws_route_table_association.public_assoc: Creation complete after 0s [id=rtbassoc-093d0b78a990c679]
aws_instance.nginx1: Still creating... [10s elapsed]
aws_instance.nginx1: Creation complete after 1s [id=eni-0cea747af0d57198]

Apply complete! Resources: 7 added, 0 changed, 0 destroyed.
PS C:\Users\patel\OneDrive\Documents\GitHub\automation-class\lab4 ]
```

EC2 instance :

The screenshot shows the AWS EC2 Instances page. The left sidebar includes sections for Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces, Load Balancing, and Load Balancers. The main content area displays a table of instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
<input checked="" type="checkbox"/> 8972657_RishikumarPatel_Lab4	i-09cea7474f0d57198	Running	t3.micro	2/2 checks passed	View alarms +	us-east-1a	-
<input type="checkbox"/> Bastion Host	i-0b0bf1c3eb34a4c6	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-18-212-1-83.c
<input type="checkbox"/> 8972657_RishikumarPatel_Lab4	i-00337325f848b3a49	Terminated	t3.micro	-	View alarms +	us-east-1a	-

The details for instance i-09cea7474f0d57198 are shown in the modal, including its summary, networking, and storage configurations.

Public IP and DNS :

This screenshot is identical to the one above, showing the AWS EC2 Instances page with the same list of three instances and their detailed configurations. The interface and layout are consistent with the first screenshot.

Security Groups :

The screenshot shows the AWS Cloud Console interface. The left sidebar is collapsed, showing the following navigation structure:

- Spot Requests
- Savings Plans
- Reserved Instances
- Dedicated Hosts
- Capacity Reservations
- Images**
 - AMIs
 - AMI Catalog
- Elastic Block Store**
 - Volumes
 - Snapshots
 - Lifecycle Manager
- Network & Security**
 - Security Groups
 - Elastic IPs
 - Placement Groups
 - Key Pairs
 - Network Interfaces
- Load Balancing**
 - Load Balancers
 - Target Groups
 - Trust Stores New
- Auto Scaling**
 - Auto Scaling Groups

Below the sidebar, there is a "Settings" link.

The main content area displays the details for a security group named "sg-0da36cd9036dfa577 - terraform-20250208211304757400000001". The "Details" section includes:

- Security group name: terraform-20250208211304757400000001
- Security group ID: sg-0da36cd9036dfa577
- Description: Managed by Terraform
- VPC ID: vpc-06c4934b2bd2a436d
- Owner: 576510067410
- Inbound rules count: 2 Permission entries
- Outbound rules count: 1 Permission entry

The "Inbound rules" tab is selected, showing a table with the following data:

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
-	sgr-019e85ed4dd4ca94	IPv4	HTTP	TCP	80	0.0.0.0/0
-	sgr-01d12d22d51848f55	IPv4	SSH	TCP	22	0.0.0.0/0

4. Access the Server: Once Terraform has finished provisioning the server, you should see the output with the server's public IP address

The screenshot shows a Windows desktop environment. A browser window is open, displaying the URL "18.212.196.16". The page content is a simple message: "You did it! Have a 🎉". The browser toolbar includes "CloudShell" and "Feedback" buttons. The taskbar at the bottom shows several pinned icons, including File Explorer, Edge, Google Chrome, Spotify, and others. The system tray indicates the date as "2/8/2025" and the time as "4:20 PM".

5. Cleanup: After completing the lab, remember to destroy the resources to avoid incurring unnecessary charges:

terraform destroy

The screenshot shows a terminal window with the following output:

```
PS C:\Users\spatel\OneDrive\Documents\GitHub\automation-class\lab4> terraform destroy
data.aws_ssm_parameter.mzn2_linux: Reading...
aws_vpc.main_vpc: Refreshing state... [id=vpc-06c4934b2d2a436d]
data.aws_ssm_parameter.mzn2_amazon_linux: Read complete after 0s
aws_vpc.main_vpc: Refreshing state... [id=vpc-06c4934b2d2a436d]
aws_internet_gateway.gw: Refreshing state... [id=sgp-0f796f28e79a967e]
aws_security_group.web_sg: Refreshing state... [id=sgp-0a3c5cd0035df5a77]
aws_route_table.public_rt: Refreshing state... [id=rtr-d39e701093e10f8d]
aws_route_table_association.public_assoc: Refreshing state... [id=rtrbassoc-093dd78a9496c679]
aws_instance.nginx1: Refreshing state... [id=i-09cea7474f0d57198]

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

Terraform will perform the following actions:

# aws_instance.nginx1 will be destroyed
- resource "aws_instance" "nginx1" {
    ami           = "ami-093dd78a9496c679"
    ami_id        = "ami-093dd78a9496c679"
    arn          = "arn:aws:ec2:us-east-1:576510067410:instance/i-09cea7474f0d57198"
    associate_public_ip_address = true
    availability_zone      = "us-east-1a"
    cpu_core_count        = 2
    disable_api_termination = false
    disable_eni_termination = false
    ebs_optimized         = false
    get_password_data     = false
    hibernation           = false
    id                   = "i-09cea7474f0d57198"
    instance_initiated_shutdown_behavior = "stop"
    instance_state        = "running"
    instance_type         = "t2.micro"
    ipv4_address_count   = 0
    ipv4_addresses        = []
    monitoring            = false
    placement_partition_number = 0
    primary_network_interface_id = "eni-093dd78a9496c679"
    private_dns           = "ip-10-0-0-234.ec2.internal"
    private_ip             = "10.0.0.234"
    public_ip              = "10.0.0.236.16"
    secondary_private_ips = []
    security_groups        = []
    source_dest_check     = true
    subnet_id             = "subnet-0f076ef28e79a967e"
    tags
}
```

The screenshot shows a terminal window with the following content:

```
File Edit Selection View Go Run Terminal Help ← → automation-class

lab4 > rp2657.tl 2.0
lab4 > rp2657.hf > resource "aws_vpc" "main_vpc"
1 provider "aws" {
2   region     = "us-east-1"
3   access_key = "AKIAJYMOVNQLJG2AYB38"
4   secret_key = "qd+Lch3eAyAzt6zAf0PGcvwVsSJOv0IZg/trKHRR"
5 }
6
7 data "aws_ssm_parameter" "amzn2_linux" {
8   name = "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2"
9 }
10
11 resource "aws_vpc" "main_vpc" {
12   cidr_block = "10.0.0.0/16"
13   tags = [
14     { Name = "MainVPC" }
15   ]
16 }
17
18 resource "aws_internet_gateway" "gw" {
19   vpc_id = aws_vpc.main_vpc.id
20
21   tags = [
22     { Name = "MainIGW" }
23   ]
24 }
25
26 resource "aws_subnet" "public_subnet" {
27   vpc_id           = aws_vpc.main_vpc.id
28   cidr_block       = "10.0.0.0/24"
29   map_public_ip_on_launch = true
30   availability_zone = "us-east-1a"
31
32   tags = [
33     { Name = "public_subnet" }
34   ]
35 }
36
37 resource "aws_route_table" "public_rt" {
38   vpc_id = aws_vpc.main_vpc.id
39
40   route {
41     cidr_block = "0.0.0.0/0"
42     gateway_id = aws_internet_gateway.gw.id
43   }
44
45   tags = [
46     { Name = "PublicRouteTable" }
47   ]
48 }

Plan: 0 to add, 0 to change, 7 to destroy.

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

aws_route_table_association.public_assoc: Destroying... [id=rtbassoc-093ddb8a9490c679]
]+ aws_instance.nginx1: Destroying... [id=i-0ca7a47fa0f57198]
aws_route_table_association.public_assoc: Destruction complete after 1s
aws_route_table.public_rt: Destroying... [id=rth-0d9a7e0163e10f8d]
aws_route_table.public_rt: Destruction complete after 0s
aws_internet_gateway.gw: Destroying... [id=igw-0963266eac6a0432]
aws_instance.nginx1: Still destroying... [id=i-0ca7a47fa0f57198, 10s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0963266eac6a0432, 10s elapsed]
aws_instance.nginx1: Still destroying... [id=i-0ca7a47fa0f57198, 20s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0963266eac6a0432, 20s elapsed]
aws_instance.nginx1: Still destroying... [id=i-0ca7a47fa0f57198, 30s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0963266eac6a0432, 30s elapsed]
aws_instance.nginx1: Still destroying... [id=i-0ca7a47fa0f57198, 40s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0963266eac6a0432, 40s elapsed]
aws_instance.nginx1: Still destroying... [id=i-0ca7a47fa0f57198, 50s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0963266eac6a0432, 50s elapsed]
aws_instance.nginx1: Still destroying... [id=i-0ca7a47fa0f57198, 1m0s elapsed]
aws_internet_gateway.gw: Still destroying... [id=igw-0963266eac6a0432, 1m0s elapsed]
aws_internet_gateway.gw: Destruction complete after 1m0s
aws_instance.nginx1: Still destroying... [id=i-0ca7a47fa0f57198, 1m1s elapsed]
aws_instance.nginx1: Destruction complete after 1m1s
aws_subnet.public_subnet: Destroying... [id=sesubnet-0f076ef28e79967e]
aws_security_group.web_sg: Destroying... [id=sg-0da36cd9036dfa577]
aws_subnet.public_subnet: Destruction complete after 0s
aws_security_group.web_sg: Destruction complete after 0s
aws_vpc.main_vpc: Destroying... [id=vpc-06c4954bd2b2d365d]
aws_vpc.main_vpc: Destruction complete after 0s

Destroy complete! Resources: 7 destroyed.
```

Screenshot of the AWS Management Console showing the EC2 Instances page.

The left sidebar navigation includes:

- Dashboard
- EC2 Global View
- Events
- Instances
 - Instances
 - Instance Types
 - Launch Templates
 - Spot Requests
 - Savings Plans
 - Reserved Instances
 - Dedicated Hosts
 - Capacity Reservations
- Images
 - AMIs
 - AMI Catalog
- Elastic Block Store
 - Volumes
 - Snapshots
 - Lifecycle Manager
- Network & Security
 - Security Groups
 - Elastic IPs
 - Placement Groups
 - Key Pairs
 - Network Interfaces
- Load Balancing
 - Load Balancers

The main content area displays the "Instances (1/3) Info" table:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
8972657_RishikumarPatel_Lab4	i-09cea7474f0d57198	Terminated	t3.micro	-	View alarms +	us-east-1a	-
Bastion Host	i-0b0bf11c3eb3494c6	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-18-212-1-83.c
8972657_RishikumarPatel_Lab4	i-00357325f848b5a49	Terminated	t3.micro	-	View alarms +	us-east-1a	-

Details for the selected instance (i-09cea7474f0d57198):

- Subnet ID: -
- IPV4 addresses: -
- Carrier IP addresses (ephemeral): -
- Secondary private IPV4 addresses: -
- Outpost ID: -
- Availability zone: us-east-1a
- Use RBN as guest OS hostname: Disabled
- Network Interfaces (0): No network interfaces attached to this instance.
- Elastic IP addresses (0): No Elastic IP addresses are associated with this instance.

Footer:

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 4:32 PM 2/8/2025