

# **Lab Assignment 5**

Rishi (8972657)

Scripting and Automation

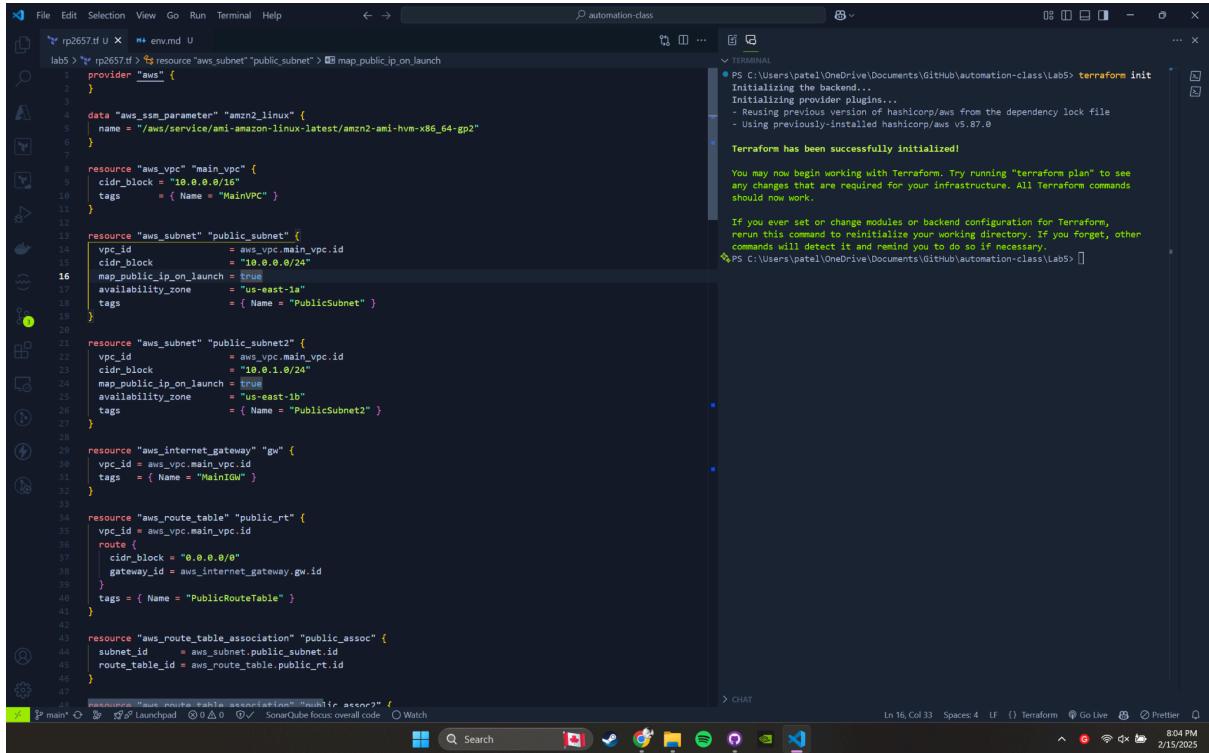
PROG8830 - Winter 2025 - Section 1

Prof. Mike Dabydeen

Feb 15, 2025

**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 Windows desktop environment. In the center is a code editor window titled "rp2657.tf U" containing Terraform configuration code. The code defines resources like VPCs, subnets, and route tables. To the right of the code editor is a terminal window titled "TERMINAL". The terminal output shows the execution of the "terraform init" command, which initializes the provider plugins and successfully initializes the Terraform state. Below the terminal is a taskbar with various icons, including a search bar, a Canadian flag icon, and system status indicators.

```
rp2657.tf U + envmd U
lab5 > terraform init
  provider "aws" {
    }
  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_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 = "PublicSubnet" }
  }
  resource "aws_subnet" "public_subnet2" {
    vpc_id = aws_vpc.main_vpc.id
    cidr_block = "10.0.1.0/24"
    map_public_ip_on_launch = true
    availability_zone = "us-east-1b"
    tags = { Name = "PublicSubnet2" }
  }
  resource "aws_internet_gateway" "gw" {
    vpc_id = aws_vpc.main_vpc.id
    tags = { Name = "MainIGW" }
  }
  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" }
  }
  resource "aws_route_table_association" "public_assoc" {
    subnet_id = aws_subnet.public_subnet.id
    route_table_id = aws_route_table.public_rt.id
  }
  resource "aws_route_table_association" "public_assoc2" {
    subnet_id = aws_subnet.public_subnet2.id
    route_table_id = aws_route_table.public_rt.id
  }
```

```
PS C:\Users\patel\OneDrive\Documents\GitHub\automation-class\Lab5> terraform init
  provider "aws" {
    }
  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_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 = "PublicSubnet" }
  }
  resource "aws_subnet" "public_subnet2" {
    vpc_id = aws_vpc.main_vpc.id
    cidr_block = "10.0.1.0/24"
    map_public_ip_on_launch = true
    availability_zone = "us-east-1b"
    tags = { Name = "PublicSubnet2" }
  }
  resource "aws_internet_gateway" "gw" {
    vpc_id = aws_vpc.main_vpc.id
    tags = { Name = "MainIGW" }
  }
  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" }
  }
  resource "aws_route_table_association" "public_assoc" {
    subnet_id = aws_subnet.public_subnet.id
    route_table_id = aws_route_table.public_rt.id
  }
  resource "aws_route_table_association" "public_assoc2" {
    subnet_id = aws_subnet.public_subnet2.id
    route_table_id = aws_route_table.public_rt.id
  }

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, run 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\Lab5>
```

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

terraform plan

The screenshot shows a terminal window titled "automation-class" running on Windows. The command "terraform plan" has been executed, and the output is displayed in the terminal pane. The output shows the execution plan for the Terraform configuration, detailing the creation of two AWS instances (nginx1 and nginx2), their associated resources like security groups and subnets, and the deployment of an Nginx configuration file. The terminal also shows the status bar with file paths, line numbers, and other system information.

```
PS C:\Users\patel\OneDrive\Documents\GitHub\automation-class\Lab5> terraform plan
data.aws_ssm_parameter.amzn2_linux: Reading...
data.aws_ssm_parameter.amzn2_linux: Read complete after 0s [id=/aws/service/ami-amazonlinux-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
Terraform will perform the following actions:
# aws_instance.nginx1 will be created
+ resource "aws_instance" "nginx1" {
  + ami = (sensitive value)
  + arn = (known after apply)
  + associate_public_ip_address = true
  + availability_zone = (known after apply)
  + cpu_core_count = (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 = "t3.micro"
  + ipv6_addresses_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)
  + 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)
  + subnet_id = (known after apply)
}
+ aws_lb_target_group_attachment.nginx1 {
  + target_group_arn = aws_lb_target_group.nginx_target_group.arn
  + target_id = aws_instance.nginx1.id
  + port = 80
}
+ aws_lb_target_group_attachment.nginx2 {
  + target_group_arn = aws_lb_target_group.nginx_target_group.arn
}
data.aws_ssm_parameter.amzn2_linux: Reading...
data.aws_ssm_parameter.amzn2_linux: Read complete after 0s [id=/aws/service/ami-amazonlinux-latest/amzn2-ami-hvm-x86_64-gp2]
```

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

terraform apply

```

resource "aws_instance" "nginx1" {
  instance_type      = "t3.micro"
  subnet_id          = aws_subnet.public_subnet.id
  vpc_security_group_ids = [aws_security_group.load_balancer_security_group.id]
  associate_public_ip_address = true
  tags = {
    Name = "8972657_RishiPatel_Nginx-Server-1"
  }
  user_data = <>EOF
#!/bin/bash
sudo amzn-linux-extras enable nginx1
sudo yum install -y nginx
sudo systemctl start nginx
sudo systemctl enable nginx
sudo rm /usr/share/nginx/html/index.html
echo "<html><head><title>Rishi Server 1</title></head><body style="background-color:#1F77B0"><p style="text-align: center;">Rishi Server 1</p></body></html>" | sudo tee /usr/share/nginx/html/index.html
</EOF>
}
resource "aws_instance" "nginx2" {
  ami                = data.aws_ssm_parameter.amzn2_linux.value
  instance_type      = "t3.micro"
  subnet_id          = aws_subnet.public_subnet2.id
  vpc_security_group_ids = [aws_security_group.load_balancer_security_group.id]
  associate_public_ip_address = true
  tags = {
    Name = "8972657_RishiPatel_Nginx-Server-2"
  }
  user_data = <>EOF
#!/bin/bash
sudo amzn-linux-extras enable nginx1
sudo yum install -y nginx
sudo systemctl start nginx
sudo systemctl enable nginx
sudo rm /usr/share/nginx/html/index.html
echo "<html><head><title>Rishi Server 2</title></head><body style="background-color:#000000"><p style="text-align: center;">Rishi Server 2</p></body></html>" | sudo tee /usr/share/nginx/html/index.html
</EOF>
}
resource "aws_lb_target_group_attachment" "nginx1" {
  target_group_arn = aws_lb_target_group.nginx_target_group.arn
  target_id        = aws_instance.nginx1.id
  port             = 80
}
resource "aws_lb_target_group_attachment" "nginx2" {
  target_group_arn = aws_lb_target_group.nginx_target_group.arn
}

```

```

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_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 = "PublicSubnet" }
}

resource "aws_subnet" "public_subnet2" {
  vpc_id          = aws_vpc.main_vpc.id
  cidr_block     = "10.0.1.0/24"
  map_public_ip_on_launch = true
  availability_zone = "us-east-1b"
  tags           = { Name = "PublicSubnet2" }
}

resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.main_vpc.id
  tags   = { Name = "MainIGW" }
}

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" }
}

resource "aws_route_table_association" "public_assoc" {
  subnet_id  = aws_subnet.public_subnet.id
  route_table_id = aws_route_table.public_rt.id
}

resource "aws_route_table_association" "public_assoc2" {
  subnet_id  = aws_subnet.public_subnet2.id
  route_table_id = aws_route_table.public_rt.id
}

```

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

EC2 :

**Instances (1/3) Info**

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
8972657_RishiPatel_Nginx-Server-2	i-02ac36bbcfe75ab53	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1b	-
Bastion Host	i-07e2ff9e6dec2700	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-52-23-
<b>8972657_RishiPatel_Nginx-Server-1</b>	<b>i-05f38cba0df47eb6c</b>	<b>Running</b>	<b>t3.micro</b>	<b>3/3 checks passed</b>	<b>View alarms +</b>	<b>us-east-1a</b>	<b>-</b>

**i-05f38cba0df47eb6c (8972657\_RishiPatel\_Nginx-Server-1)**

**Networking**

Public IPv4 address	3.92.176.67   open address	Private IPv4 addresses	10.0.0.26	VPC ID	vpc-08cc3e73550bad003 (MainVPC)
Public IPv4 DNS	-	Private IP DNS name (IPv4 only)	ip-10-0-0-26.ec2.internal	Secondary private IPv4 addresses	-
Subnet ID	subnet-034542ebded103073 (PublicSubnet)	IPV6 addresses	-	Outpost ID	-
Availability zone	us-east-1a	Carrier IP addresses (ephemeral)	-		
Use RBN as guest OS hostname	Disabled	Answer RBN DNS hostname IPv4	Disabled		

**Instances (1/3) Info**

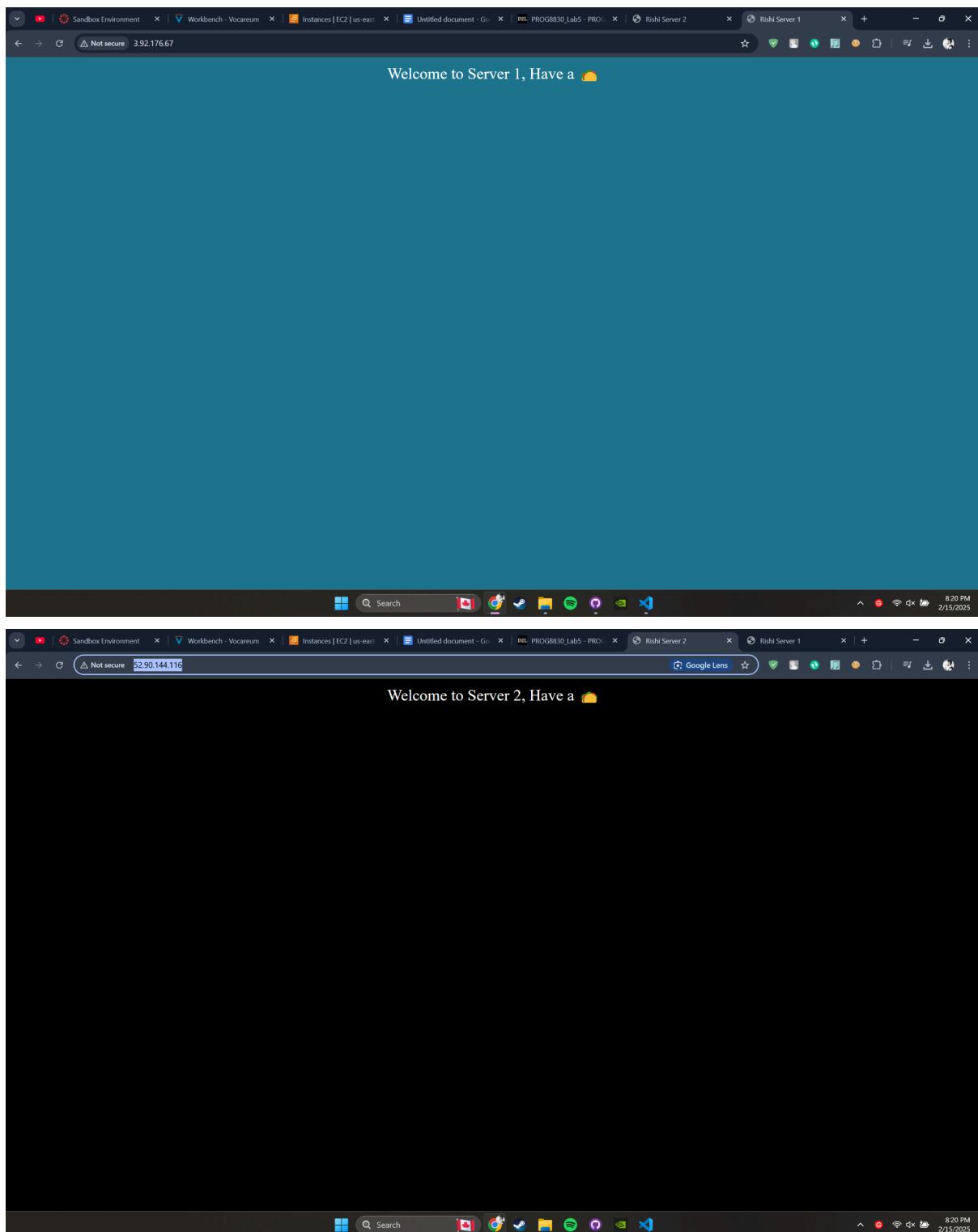
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
8972657_RishiPatel_Nginx-Server-2	i-02ac36bbcfe75ab53	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1b	-
Bastion Host	i-07e2ff9e6dec2700	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-52-23-
8972657_RishiPatel_Nginx-Server-1	i-05f38cba0df47eb6c	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	-

**i-02ac36bbcfe75ab53 (8972657\_RishiPatel\_Nginx-Server-2)**

**Networking**

Public IPv4 address	52.90.144.116   open address	Private IPv4 addresses	10.0.1.193	VPC ID	vpc-08cc3e73550bad003 (MainVPC)
Public IPv4 DNS	-	Private IP DNS name (IPv4 only)	ip-10-0-1-193.ec2.internal	Secondary private IPv4 addresses	-
Subnet ID	subnet-070d8738363a29b13 (PublicSubnet2)	IPV6 addresses	-	Outpost ID	-
Availability zone	us-east-1b	Carrier IP addresses (ephemeral)	-		
Use RBN as guest OS hostname	Disabled	Answer RBN DNS hostname IPv4	Disabled		

## Public Servers :



## Target Groups:

The screenshot shows the AWS EC2 Target Groups console. The main page title is "nginxtargetgroup". The left sidebar includes sections for Instances, Images, Elastic Block Store, Network & Security, and Load Balancing. The main content area displays target group details: Target type (Instance), Protocol: Port (HTTP: 80), Protocol version (HTTP1), and VPC (vpc-08cc3e73550bad003). It shows 2 total targets, with 2 healthy and 0 unhealthy. Below this is a table titled "Distribution of targets by Availability Zone (AZ)" with columns for Total targets, AZ, Status, and Count. A link to "Registered targets (2)" is present, leading to a detailed list of two targets: "i-05f38cba0df47eb6c" and "i-02ac36bbcfe75ab53", both of which are healthy.

## Security Groups:

The screenshot shows the AWS EC2 Security Groups console. The main page title is "sg-0a97ea4127f92b183 - terraform-20250216011516017200000001". The left sidebar includes sections for Instances, Images, Elastic Block Store, Network & Security, and Load Balancing. The main content area displays security group details: Security group name (terraform-20250216011516017200000001), Security group ID (sg-0a97ea4127f92b183), Owner (576510067410), Description (Managed by Terraform), and VPC ID (vpc-08cc3e73550bad003). Below this is a table titled "Inbound rules (1)" with columns for Name, Security group rule ID, IP version, Type, Protocol, Port range, and Source. One rule is listed: sgr-03d080243b4d49f0d, IPv4, HTTP, TCP, port 80, source 0.0.0.0/0.

## Load balancer:

The screenshot shows the AWS CloudWatch Metrics interface. On the left, there's a navigation pane with links like 'Metrics Home', 'Metrics Overview', 'Metrics Insights', 'Metrics Metrics', and 'Metrics Metrics Insights'. The main area displays a table of metrics for a single function. The columns include Metric Name, Unit, Value, and Last Value. The last row shows the metric 'LambdaFunction:Invocations' with a value of 1000 and a last value of 1000.

Metric Name	Unit	Value	Last Value
LambdaFunction:Invocations	Count	1000	1000

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

## terraform destroy

The screenshot shows a Windows desktop environment with several open windows. The most prominent is a PowerShell window titled 'TERMINAL' where the command `terraform destroy` is being run against a local repository. Below the terminal, the output of the Terraform plan is visible, detailing the resources that will be destroyed. Another window shows a GitHub pull request for 'automation-class' with the commit message 'Terraform destroy - Lab5'. The desktop also features a taskbar with icons for File Explorer, Task View, Start, and various pinned applications like Edge, File Explorer, and a file manager.

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

```
lab5 > $pr2657tf > env.md
```

```
resource "aws_instance" "nginx1" > user_data
```

```
4 data "aws_ssm_parameter" "amzn2_linux" {
```

```
5   name = "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2"
```

```
6 }
```

```
7
```

```
resource "aws_vpc" "main_vpc" {
```

```
8   cidr_block = "10.0.0.0/16"
```

```
9   tags = { Name = "MainVPC" }
```

```
10 }
```

```
11
```

```
resource "aws_subnet" "public_subnet" {
```

```
12   vpc_id           = aws_vpc.main_vpc.id
```

```
13   cidr_block       = "10.0.0.0/24"
```

```
14   map_public_ip_on_launch = true
```

```
15   availability_zone = "us-east-1a"
```

```
16   tags             = { Name = "PublicSubnet" }
```

```
17 }
```

```
18
```

```
resource "aws_subnet" "public_subnet2" {
```

```
19   vpc_id           = aws_vpc.main_vpc.id
```

```
20   cidr_block       = "10.0.1.0/24"
```

```
21   map_public_ip_on_launch = true
```

```
22   availability_zone = "us-east-1b"
```

```
23   tags             = { Name = "PublicSubnet2" }
```

```
24 }
```

```
25
```

```
resource "aws_internet_gateway" "gw" {
```

```
26   vpc_id = aws_vpc.main_vpc.id
```

```
27   tags   = { Name = "MainIGW" }
```

```
28 }
```

```
29
```

```
resource "aws_route_table" "public_rt" {
```

```
30   vpc_id = aws_vpc.main_vpc.id
```

```
31   route {
```

```
32     cidr_block = "0.0.0.0/8"
```

```
33     gateway_id = aws_internet_gateway.gw.id
```

```
34   }
```

```
35   tags = { Name = "PublicRouteTable" }
```

```
36 }
```

```
37
```

```
resource "aws_route_table_association" "public_assoc" {
```

```
38   subnet_id      = aws_subnet.public_subnet.id
```

```
39   route_table_id = aws_route_table.public_rt.id
```

```
40 }
```

```
41
```

```
resource "aws_route_table_association" "public_assoc2" {
```

```
42   subnet_id      = aws_subnet.public_subnet2.id
```

```
43   route_table_id = aws_route_table.public_rt.id
```

```
44 }
```

```
45
```

```
data.aws_ssm_parameter.amzn2_linux: Reading...
```

```
data.aws_ssm_parameter.amzn2_linux: Read complete after 0s [id:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2]
```

```
aws_internet_gateway: Refreshing state... [id:igw-0b7d15f63357461a]
```

```
aws_subnet_public_subnet: Refreshing state... [id:subnet-034542bde0d103073]
```

```
aws_subnet_public_subnet2: Refreshing state... [id:subnet-070d78363a29b13]
```

```
aws_security_group_load_balancer_security_group: Refreshing state... [id:sg-0a97e4a127f92b1b3]
```

```
aws_lb_target_group.nginx1: Refreshing state... [id:arn:aws:elasticloadbalancing:us-east-1:576510067410:targetgroup/nginxtargetgroup:f0e683084000000cf]
```

```
aws_route_table_public_rt: Refreshing state... [id:rtrt-0eccd02b58f536c5]
```

```
aws_route_table_association_public_assoc2: Refreshing state... [id:rtrbassoc-07c186a8a9e2ca188]
```

```
aws_instance.nginx1: Refreshing state... [id:eni-02ac36b0c75ab55]
```

```
aws_route_table_association_public_assoc: Refreshing state... [id:rtrbassoc-04def64ea2e92fd3c]
```

```
aws_lb_listener.nginx1: Refreshing state... [id:arn:aws:elasticloadbalancing:us-east-1:576510067410:listener/app/lb-8972657/dfa936062d2459]
```

```
aws_instance.nginx1: Refreshing state... [id:eni-05f3c8ba0df7eb6c]
```

```
aws_lb_listener.front_end: Refreshing state... [id:arn:aws:elasticloadbalancing:us-east-1:576510067410:listener/app/lb-8972657/dfa936062d2459/aecc0ff1c899347]
```

```
aws_lb_target_group_attachment.nginx1: Refreshing state... [id:arn:aws:elasticloadbalancing:us-east-1:576510067410:targetgroup/nginxtargetgroup:f9e68308400000005]
```

```
aws_lb_target_group_attachment.nginx1: Refreshing state... [id:arn:aws:elasticloadbalancing:us-east-1:576510067410:targetgroup/nginxtargetgroup:f9e68308400000005/2025021601154053460000000005]
```

```
aws_lb_target_group_attachment.nginx1: Refreshing state... [id:arn:aws:elasticloadbalancing:us-east-1:576510067410:targetgroup/nginxtargetgroup:f9e68308400000005/2025021601154053460000000005]
```

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                                     = (sensitive value) => null
    - ami_name                                = (sensitive value) => null
    - associate_public_ip_address               = true => null
    - availability_zone                        = "us-east-1a" => null
    - cpu_core_count                           = 1 => null
    - cpu_threads_per_core                    = 2 => null
    - disable_api_stop                         = false => null
    - disable_api_termination                 = false => null
    - ebs_optimized                            = false => null
    - get_password_data                       = false => null
}
```

The screenshot shows a terminal window with several tabs open. The active tab displays a CloudFormation template named 'rp2657.tf' containing AWS resources like VPCs, subnets, and route tables. To the right of the code, a log stream titled 'TERMINAL' shows the destruction of these resources. The log entries are timestamped and include details such as the resource ID, type, and the command used to destroy it (e.g., 'aws\_lb\_listener.front\_end: Destroying...'). The log output is color-coded, with red text indicating errors or warnings.

```
File Edit Selection View Go Run Terminal Help ↺ ↻ automation-class

rp2657.tf x env.md

lab5 > rp2657.tf > resource "aws_route_table" "public_rt" > route
4   data "aws_ssm_parameter" "amzn2_linux" {
5     name = "/aws/service/amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2"
6   }
7
8   resource "aws_vpc" "main_vpc" {
9     cidr_block = "10.0.0.0/16"
10    tags = { Name = "MainVPC" }
11  }
12
13  resource "aws_subnet" "public_subnet" {
14    vpc_id      = aws_vpc.main_vpc.id
15    cidr_block  = "10.0.0.0/24"
16    map_public_ip_on_launch = true
17    availability_zone = "us-east-1a"
18    tags        = { Name = "PublicSubnet" }
19  }
20
21  resource "aws_subnet" "public_subnet2" {
22    vpc_id      = aws_vpc.main_vpc.id
23    cidr_block  = "10.0.1.0/24"
24    map_public_ip_on_launch = true
25    availability_zone = "us-east-1b"
26    tags        = { Name = "PublicSubnet2" }
27  }
28
29  resource "aws_internet_gateway" "gw" {
30    vpc_id = aws_vpc.main_vpc.id
31    tags  = { Name = "MainIGW" }
32  }
33
34  resource "aws_route_table" "public_rt" {
35    vpc_id = aws_vpc.main_vpc.id
36    route {
37      cidr_block = "0.0.0.0/0"
38      gateway_id = aws_internet_gateway.gw.id
39    }
40    tags = { Name = "PublicRouteTable" }
41  }
42
43  resource "aws_route_table_association" "public_assoc" {
44    subnet_id  = aws_subnet.public_subnet.id
45    route_table_id = aws_route_table.public_rt.id
46  }
47
48  resource "aws_route_table_association" "public_assoc2" {
49    subnet_id  = aws_subnet.public_subnet2.id
50    route_table_id = aws_route_table.public_rt.id
51  }

gus-east-1:576510067410>:targetgroup/inginxtargetgroup/9e6e8308400c8ff-20250216011540397500000005
aws_lb_listener.front_end: Destroying... [id=arn:aws:elasticloadbalancing:us-east-1:576510067410:listener/app/bc893f06e62d2459:f4d893f06e62d2459:c8893347]
aws_lb_target_group_attachment.nginx1: Destruction complete after 1s
aws_instance.nginx1: Destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:f74eb6c]
aws_lb_target_group_attachment.nginx2: Destruction complete after 1s
aws_instance.nginx2: Destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:f74eb6c]
aws_lb_listener.front_end: Destroying... [id=arn:aws:elasticloadbalancing:us-east-1:576510067410:listener/group/inginxtargetgroup/9e6e8308400c8ff]
aws_lb.nginx: Destroying... [id=arn:aws:elasticloadbalancing:us-east-1:576510067410:loadbalancer/app/bc893f06e62d2459]
aws_route_table_association.nginx1: Destruction complete after 1s
aws_route_table_association.nginx2: Destruction complete after 1s
aws_route_table_public_rt: Destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:807ec02b6586f38c]
aws_lb_target_group.nginx_target_group: Destruction complete after 0s
aws_table.public_table: Destruction complete after 0s
aws_internet_gateway.gw: Destroying... [id=dgw-071d15f6357461a]
aws_lb.nginx: Still destroying... [id=arn:aws:elasticloadbalancing:us-east-1:576510067410:loadbalancer/app/bc893f06e62d2459]
aws_instance.nginx1: Still destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:f74eb6c, 10s elapsed]
aws_instance.nginx2: Still destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:f74eb6c, 10s elapsed]
aws_lb_target_group_attachment.nginx1: Still destroying... [id=arn:aws:elasticloadbalancing:us-east-1:576510067410:targetgroup/inginxtargetgroup/9e6e8308400c8ff, 20s elapsed]
aws_instance.nginx1: Still destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:f74eb6c, 20s elapsed]
aws_instance.nginx2: Still destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:f74eb6c, 20s elapsed]
aws_internet_gateway.gw: Still destroying... [id=dgw-071d15f6357461a, 20s elapsed]
aws_instance.nginx1: Still destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:f74eb6c, 30s elapsed]
aws_instance.nginx2: Still destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:f74eb6c, 30s elapsed]
aws_instance.nginx1: Destruction complete after 30s
aws_subnet.public_subnet: Destroying... [id=subnet-034524bedded103073]
aws_subnet.public_subnet: Destruction complete after 30s
aws_subnet.public_subnet: Still destroying... [id=subnet-034524bedded103073, 40s elapsed]
aws_subnet.public_subnet: Still destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:9e6e8308400c491a, 40s elapsed]
aws_instance.nginx2: Still destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:9e6e8308400c491a, 40s elapsed]
aws_internet_gateway.gw: Still destroying... [id=dgw-071d15f6357461a, 40s elapsed]
aws_instance.nginx2: Still destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:9e6e8308400c491a, 50s elapsed]
aws_internet_gateway.gw: Still destroying... [id=dgw-071d15f6357461a, 50s elapsed]
aws_internet_gateway.gw: Destruction complete after 58s
aws_instance.nginx2: Still destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:9e6e8308400c491a, 1m0s elapsed]
aws_instance.nginx2: Still destroying... [id=arn:aws:lambda:us-east-1:576510067410:function:9e6e8308400c491a, 1m0s elapsed]
aws_subnet.public_subnet2: Destroying... [id=subnet-07d0738363a29b13]
aws_security_group.load_balancer_security_group: Destroying... [id=sg-a97ea41279f2b183]
aws_subnet.public_subnet2: Destruction complete after 0s
aws_security_group.load_balancer_security_group: Destruction complete after 0s
aws_vpc.main_vpc: Destroying... [id=vpc-08c3e73550bad003]
aws_vpc.main_vpc: Destruction complete after 0s

Destroy completed: Resources: 15 destroyed.
PS C:\Users\psatel\OneDrive\Documents\GitHub\automation-class\Lab5>
```