



## **Assignment 2**

**Lab title: Advanced Terraform & Nginx Multi-Tier Architecture**

**Submitted to: Engr. Mohammad Shoaib**

**Submitted by: Anara Hayat**

**Reg#No: 2023-BSE-008**

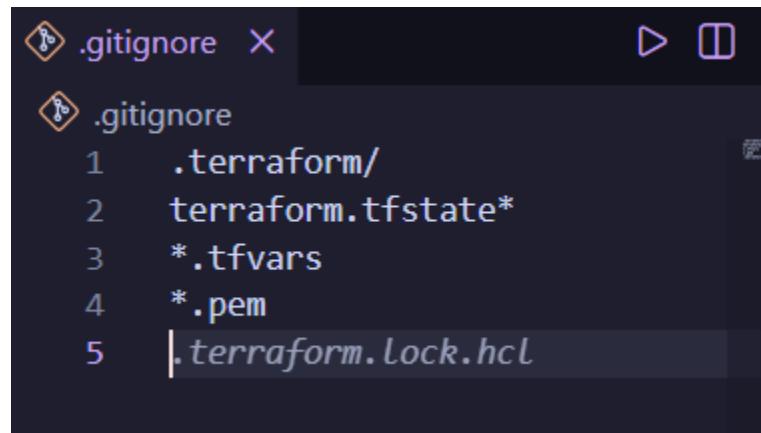
## Requirements

### Part 1: Infrastructure Setup

#### 1.1 Project Structure

Create a well-organized Terraform project with the following structure:

```
~\OneDrive\Desktop\CC Labs (0.06s)
tree Assignment2
Folder PATH listing
Volume serial number is 00000089 3A0B:38A3
C:\USERS\ANARA HAYAT\ONEDRIVE\DESKTOP\CC LABS\ASSIGNMENT2
├── modules
│   ├── networking
│   ├── security
│   └── webserver
└── scripts
```



```
.gitignore
.terraform/
terraform.tfstate*
*.tfvars
*.pem
|.terraform.lock.hcl
```

#### 1.2 Variable Configuration (5 marks)

Define all required variables in variables.tf:

```
variables.tf
1  variable "aws_region" {
2    description = "AWS region"
3    type        = string
4    default     = "me-central-1"
5  }
6
7  variable "vpc_cidr_block" {
8    description = "CIDR block for VPC"
9    type        = string
10   validation {
11     condition    = can(cidrnetmask(var.vpc_cidr_block))
12     error_message = "Invalid VPC CIDR block"
13   }
14 }
15
16 variable "subnet_cidr_block" {
17   description = "CIDR block for subnet"
18   type        = string
19   validation {
20     condition    = can(cidrnetmask(var.subnet_cidr_block))
21     error_message = "Invalid subnet CIDR block"
22   }
23 }
24
25 variable "availability_zone" {
26   description = "Availability Zone"
27   type        = string
28 }
29
30 variable "env_prefix" {
31   description = "Environment prefix"
32   type        = string
33 }
34
```

```
terraform.tfvars X
terraform.tfvars
1  vpc_cidr_block      = "10.0.0.0/16"
2  subnet_cidr_block   = "10.0.10.0/24"
3  availability_zone   = "me-central-1a"
4  env_prefix           = "prod"
5  instance_type         = "t3.micro"
6  public_key            = "~/.ssh/id_ed25519.pub"
7  private_key           = "~/.ssh/id_ed25519"
```

### 1.3 Networking Module

Create a networking module that provisions:

- VPC with specified CIDR block
- Subnet with public IP assignment enabled
- Internet Gateway
- Route table with default route to IGW
- Associate route table with subnet

#### Tasks:

- Create VPC resource
- Create subnet with map\_public\_ip\_on\_launch = true
- Create and attach Internet Gateway
- Configure routing table
- Add proper tags to all resources using env\_prefix

```
modules > networking > main.tf
1   resource "aws_vpc" "this" {
2     cidr_block = var.vpc_cidr_block
3     tags        = merge(var.common_tags, { Name = "${var.env_prefix}VPC" })
4   }
5
6   resource "aws_subnet" "this" {
7     vpc_id          = aws_vpc.this.id
8     cidr_block      = var.subnet_cidr_block
9     availability_zone = var.availability_zone
10    map_public_ip_on_launch = true
11    tags           = merge(var.common_tags, { Name = "${var.env_prefix}Subnet" })
12  }
13
14  resource "aws_internet_gateway" "this" {
15    vpc_id = aws_vpc.this.id
16    tags   = merge(var.common_tags, { Name = "${var.env_prefix}IGW" })
17  }
18
19  resource "aws_route_table" "this" {
20    vpc_id = aws_vpc.this.id
21    route {
22      cidr_block = "0.0.0.0/0"
23      gateway_id = aws_internet_gateway.this.id
24    }
25  }
26
27  resource "aws_route_table_association" "this" {
28    subnet_id      = aws_subnet.this.id
29    route_table_id = aws_route_table.this.id
30  }
```

```
modules > networking > outputs.tf
1   output "vpc_id" { value = aws_vpc.this.id }
2   output "subnet_id" { value = aws_subnet.this.id }
3
```

## 1.4 Security Module

Create a security module that provisions:

### Tasks:

- Create Nginx security group with appropriate rules

- Create backend security group with appropriate rules
- Use security group IDs for backend ingress (not CIDR blocks)
- Add descriptive names and tags

```
modules > security > main.tf
  1   resource "aws_security_group" "nginx" {
  2     vpc_id = var.vpc_id
  3     name   = "${var.env_prefix}-nginx-sg"
  4
  5     ingress {
  6       from_port   = 22
  7       to_port    = 22
  8       protocol   = "tcp"
  9       cidr_blocks = [var.my_ip]
 10    }
 11
 12    ingress {
 13      from_port   = 80
 14      to_port    = 80
 15      protocol   = "tcp"
 16      cidr_blocks = ["0.0.0.0/0"]
 17    }
 18
 19    ingress {
 20      from_port   = 443
 21      to_port    = 443
 22      protocol   = "tcp"
 23      cidr_blocks = ["0.0.0.0/0"]
 24    }
 25
 26    egress {
 27      from_port   = 0
 28      to_port    = 0
 29      protocol   = "-1"
 30      cidr_blocks = ["0.0.0.0/0"]
 31    }
 32
 33    tags = var.common_tags
 34  }
```

## sg-0c22531f5e2e5be5f - prod-nginx-sg

Actions ▾

### Details

**Security group name**

prod-nginx-sg

**Security group ID**

sg-0c22531f5e2e5be5f

**Description**

Security group for Nginx reverse proxy

**VPC ID**

[vpc-062a6835cf68bb46f ↗](#)

**Owner**

249471344514

**Inbound rules count**

3 Permission entries

**Outbound rules count**

1 Permission entry

**Inbound rules**

**Outbound rules**

**Sharing**

**VPC associations**

**Tags**

## sg-0d52409f0959ff8e3 - prod-backend-sg

Actions ▾

### Details

**Security group name**

prod-backend-sg

**Security group ID**

sg-0d52409f0959ff8e3

**Description**

Security group for backend servers

**VPC ID**

[vpc-062a6835cf68bb46f ↗](#)

**Owner**

249471344514

**Inbound rules count**

2 Permission entries

**Outbound rules count**

1 Permission entry

### 1.5 Locals Configuration (5 marks)

Create locals.tf with:

- Dynamic IP detection for my\_ip
- Resource naming conventions
- Common tags

- Backend server configurations

**Tasks:**

- Implement dynamic IP detection
- Define common tags
- Define backend server list
- Add any other reusable local values

```
# Local values
locals {
    # Get current public IP with /32 CIDR notation
    my_ip = "${chomp(data.http.my_ip.response_body)}/32"

    # Common tags for all resources
    common_tags = {
        Environment = var.env_prefix
        Project      = "Assignment-2"
        ManagedBy    = "Terraform"
        CreatedDate  = timestamp()
    }

    # Backend server configurations
    backend_servers = [
        {
            name      = "web-1"
            suffix    = "1"
            script_path = "./scripts/apache-setup.sh"
        },
        {
            name      = "web-2"
            suffix    = "2"
            script_path = "./scripts/apache-setup.sh"
        },
        {
            name      = "web-3"
            suffix    = "3"
            script_path = "./scripts/apache-setup.sh"
        }
    ]
}
```

## Part 2: Webserver Module

### 2.1 Module Design (10 marks)

Create a reusable webserver module in modules/webserver/

**Tasks:**

- Create variables.tf with all required variables
- Create main.tf with key pair and instance resources
- Create outputs.tf with all required outputs
- Ensure module is reusable for both Nginx and backend servers

```
modules > webserver > 📄 variables.tf
  variable "env_prefix" { type = string }
  variable "instance_name" { type = string }
  variable "instance_suffix" { type = string }
  variable "instance_type" { type = string }
  variable "availability_zone" { type = string }
  variable "vpc_id" { type = string }
  variable "subnet_id" { type = string }
  variable "security_group_id" { type = string }
  variable "public_key" { type = string }
  variable "script_path" { type = string }
  variable "common_tags" { type = map(string) }
|
```

```
main.tf
modules > webserver > main.tf
  1  data "aws_ami" "amazon_linux" {
  2    most_recent = true
  3    owners      = ["amazon"]
  4
  5    filter {
  6      name    = "name"
  7      values  = ["al2023-ami-*-x86_64"]
  8    }
  9  }
 10
 11 resource "aws_key_pair" "this" {
 12   key_name    = "${var.env_prefix}-${var.instance_suffix}-key"
 13   public_key  = file(var.public_key)
 14 }
 15
 16 resource "aws_instance" "this" {
 17   ami           = data.aws_ami.amazon_linux.id
 18   instance_type = var.instance_type
 19   availability_zone = var.availability_zone
 20   subnet_id     = var.subnet_id
 21   vpc_security_group_ids = [var.security_group_id]
 22   key_name       = aws_key_pair.this.key_name
 23   user_data      = file(var.script_path)
 24
 25   tags = merge(var.common_tags, {
 26     Name = "${var.env_prefix}-${var.instance_name}"
 27   })
 28 }
```

```
outputs.tf
modules > webserver > outputs.tf
  1  output "instance_id" { value = aws_instance.this.id }
  2  output "public_ip" { value = aws_instance.this.public_ip }
  3  output "private_ip" { value = aws_instance.this.private_ip }
  4
```

## 2.2 Module Usage (5 marks)

In root main.tf, instantiate the webserver module for:

1. One Nginx server (using nginx-setup.sh)

2. Three backend servers (web-1, web-2, web-3 using apache-setup.sh)

Use dynamic blocks or for\_each for backend servers:

Tasks:

- Create Nginx server module instance
- Create backend server module instances using for\_each
- Ensure proper security group assignment
- Pass all required variables

```
└─ main.tf
  36 # NGINX SERVER
  37 module "nginx_server" {
  38   source          = "./modules/webserver"
  39   env_prefix      = var.env_prefix
  40   instance_name   = "nginx-proxy"
  41   instance_suffix = "nginx"
  42   instance_type   = var.instance_type
  43   availability_zone = var.availability_zone
  44   vpc_id          = module.networking.vpc_id
  45   subnet_id       = module.networking.subnet_id
  46   security_group_id = module.security.nginx_sg_id
  47   public_key       = var.public_key
  48   script_path      = "./scripts/nginx-setup.sh"
  49   common_tags      = local.common_tags
  50 }
  51
  52 # BACKEND SERVERS
  53 module "backend_servers" {
  54   for_each = { for s in local.backend_servers : s.name => s }
  55
  56   source          = "./modules/webserver"
  57   env_prefix      = var.env_prefix
  58   instance_name   = each.value.name
  59   instance_suffix = each.value.suffix
  60   instance_type   = var.instance_type
  61   availability_zone = var.availability_zone
  62   vpc_id          = module.networking.vpc_id
  63   subnet_id       = module.networking.subnet_id
  64   security_group_id = module.security.backend_sg_id
  65   public_key       = var.public_key
  66   script_path      = each.value.script_path
  67   common_tags      = local.common_tags
  68 }
```

## **Part 3: Server Configuration Scripts (20 marks)**

### **3.1 Apache Backend Server Script (10 marks)**

Create scripts/apache-setup.sh that:

- Updates system packages
- Installs Apache HTTP Server
- Starts and enables Apache service
- Creates custom HTML page displaying:
  - Server name/hostname
  - Private IP address
  - Public IP address
  - Public DNS hostname
  - Custom message indicating which backend server (web-1, web-2, or web-3)
  - Timestamp of deployment
  - Server status (Primary/Backup)

#### **Tasks:**

- Create the script with all required features
- Ensure it uses IMDSv2 for metadata
- Create visually appealing HTML output
- Make script executable
- Test script independently

```
scripts > $ apache-setup.sh ● ▶ ⓘ ...  
1  #!/bin/bash  
2  set -e  
3  
4  # Update system  
5  yum update -y  
6  
7  # Install Apache  
8  yum install httpd -y  
9  
10 # Start and enable Apache  
11 systemctl start httpd  
12 systemctl enable httpd  
13  
14 # Get metadata token (IMDSv2)  
15 TOKEN=$(curl -s -X PUT "http://169.254.169.254/latest/api/token"  
16   -H "X-aws-ec2-metadata-token-ttl-seconds: 21600")  
17  
18 # Get instance metadata  
19 PRIVATE_IP=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \  
20   http://169.254.169.254/latest/meta-data/local-ipv4)  
21 PUBLIC_IP=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \  
22   http://169.254.169.254/latest/meta-data/public-ipv4)  
23 PUBLIC_DNS=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \  
24   http://169.254.169.254/latest/meta-data/public-hostname)  
25 INSTANCE_ID=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \  
26   http://169.254.169.254/latest/meta-data/instance-id)  
27  
28 # Set hostname  
29 hostnamectl set-hostname myapp-webserver  
30  
31 # Create custom HTML page  
32 cat > /var/www/html/index.html <<EOF  
33 <!DOCTYPE html>  
34 <html>  
35 <head>  
36   <title>Backend Web Server</title>
```

**Resources Created:**

- 1 Nginx reverse proxy server (Public IP: 3.29.231.86)
- 3 Backend web servers:
  - web-1: 10.0.10.203 (Public: 3.28.204.70)
  - web-2: 10.0.10.146 (Public: 158.252.81.36)
  - web-3: 10.0.10.83 (Public: 3.29.64.238)
- VPC with networking infrastructure (subnet, internet gateway, route table)
- Security groups for SSH and HTTP/HTTPS access

The error was fixed by changing the availability zone from `me-central-1` to `me-central-1a` in the `terraform.tfvars` file.



## Backend Web Server - Assignment

2

**Hostname:** myapp-webserver

**Instance ID:** i-060422b95ce8e430e

**Private IP:** 10.0.10.203

**Public IP:** 3.28.204.70

**Public DNS:**

**Deployed:** Mon Dec 29 19:47:55 UTC 2025

**Status:**  Active and Running

**Managed By:** Terraform

### 3.2 Nginx Server Setup Script (10 marks)

**Create scripts/nginx-setup.sh that:**

- Updates system packages
- Installs Nginx
- Generates self-signed SSL certificate
- Configures Nginx with:
  - HTTPS on port 443
  - HTTP to HTTPS redirect
  - Upstream backend servers (web-1, web-2, web-3)
  - Load balancing configuration
  - Caching configuration
  - Proper logging
  - Security headers

**Tasks:**

- **Create script with SSL certificate generation**
- **Configure upstream with placeholder IPs**
- **Implement caching**
- **Add security headers**
- **Configure HTTP to HTTPS redirect**
- **Add health check endpoint**

```
scripts > $ nginx-setup.sh
1  #!/bin/bash
2  set -e
3
4  # Update and install Nginx
5  yum update -y
6  yum install -y nginx openssl
7  systemctl start nginx
8  systemctl enable nginx
9
10 # Create SSL directories
11 mkdir -p /etc/ssl/private
12 mkdir -p /etc/ssl/certs
13
14 # Get metadata token
15 TOKEN=$(curl -s -X PUT "http://169.254.169.254/latest/api/to
16   -H "X-aws-ec2-metadata-token-ttl-seconds: 21600")
17
18 # Get public IP
19 PUBLIC_IP=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \
20   http://169.254.169.254/latest/meta-data/public-ipv4)
21
22 # Generate self-signed certificate
23 openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
24   -keyout /etc/ssl/private/selfsigned.key \
25   -out /etc/ssl/certs/selfsigned.crt \
26   -subj "/CN=$PUBLIC_IP" \
27   -addext "subjectAltName=IP:$PUBLIC_IP" \
28   -addext "basicConstraints=CA:FALSE" \
29   -addext "keyUsage=digitalSignature,keyEncipherment" \
30   -addext "extendedKeyUsage=serverAuth"
31
32 echo "Self-signed certificate created for IP: $PUBLIC_IP"
33
34 # Backup original config
35 cp /etc/nginx/nginx.conf /etc/nginx/nginx.conf.bak
36
37 # Create Nginx configuration
```

# Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](http://nginx.org).  
Commercial support is available at [nginx.com](http://nginx.com).

*Thank you for using nginx.*

## Part 4: Infrastructure Deployment (15 marks)

### 4.1 Initial Deployment (5 marks)

Deploy the infrastructure using Terraform.

#### Tasks:

- Generate SSH key pair if not exists
- Initialize Terraform (terraform init)
- Validate configuration (terraform validate)
- Plan deployment (terraform plan)
- Apply configuration (terraform apply -auto-approve)



```
~\OneDrive\Desktop\CC Labs\Assignment2 (9.87s)
terraform init
Initializing the backend...
Initializing modules...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Reusing previous version of hashicorp/http from the dependency lock file
- Using previously-installed hashicorp/aws v5.100.0
- Using previously-installed hashicorp/http v3.5.0

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.
```

```
~\OneDrive\Desktop\CC Labs\Assignment2 (4.403s)
terraform validate
Success! The configuration is valid.
```

```
~\OneDrive\Desktop\CC Labs\Assignment2 (8.709s)
terraform plan
data.http.my_ip: Read complete after 0s [id=https://ipv4.icanhazip.com]
module.nginx_server.data.aws_ami.amazon_linux: Reading...
module.backend_servers["web-2"].data.aws_ami.amazon_linux: Reading...
module.backend_servers["web-3"].aws_key_pair.this: Refreshing state... [id=prod-3-key]
module.backend_servers["web-2"].aws_key_pair.this: Refreshing state... [id=prod-2-key]
module.backend_servers["web-1"].aws_key_pair.this: Refreshing state... [id=prod-1-key]
module.backend_servers["web-1"].data.aws_ami.amazon_linux: Reading...
module.networking.aws_vpc.this: Refreshing state... [id=vpc-062a6835cf68bb46f]
module.backend_servers["web-3"].data.aws_ami.amazon_linux: Reading...
module.nginx_server.aws_key_pair.this: Refreshing state... [id=prod-nginx-key]
module.backend_servers["web-3"].data.aws_ami.amazon_linux: Read complete after 1s [id=ami-009ce8169fc88edf5]
module.backend_servers["web-2"].data.aws_ami.amazon_linux: Read complete after 1s [id=ami-009ce8169fc88edf5]
module.backend_servers["web-1"].data.aws_ami.amazon_linux: Read complete after 1s [id=ami-009ce8169fc88edf5]
module.nginx_server.data.aws_ami.amazon_linux: Read complete after 1s [id=ami-009ce8169fc88edf5]
module.networking.aws_internet_gateway.this: Refreshing state... [id=igw-01ca0e583213bc718]
module.networking.aws_subnet.this: Refreshing state... [id=subnet-0f4ce09fd93138b7f]
module.security.aws_security_group.nginx: Refreshing state... [id=sg-0c22531f5e2e5be5f]
```

```
terraform apply -auto-approve

data.http.my_ip: Reading...
data.http.my_ip: Read complete after 0s [id=https://ipv4.icanhazip.com]
module.backend_servers["web-3"].data.aws_ami.amazon_linux: Reading...
module.nginx_server.aws_key_pair.this: Refreshing state... [id=prod-nginx-key]
module.backend_servers["web-2"].data.aws_ami.amazon_linux: Reading...
module.backend_servers["web-1"].data.aws_ami.amazon_linux: Reading...
module.backend_servers["web-2"].aws_key_pair.this: Refreshing state... [id=prod-2-key]
module.nginx_server.data.aws_ami.amazon_linux: Reading...
module.backend_servers["web-1"].aws_key_pair.this: Refreshing state... [id=prod-1-key]
module.backend_servers["web-3"].aws_key_pair.this: Refreshing state... [id=prod-3-key]
module.networking.aws_vpc.this: Refreshing state... [id=vpc-062a6835cf68bb46f]
module.backend_servers["web-1"].data.aws_ami.amazon_linux: Read complete after 1s [id=ami-009ce8169fc88edf5]
module.nginx_server.data.aws_ami.amazon_linux: Read complete after 1s [id=ami-009ce8169fc88edf5]
module.backend_servers["web-3"].data.aws_ami.amazon_linux: Read complete after 1s [id=ami-009ce8169fc88edf5]
module.backend_servers["web-2"].data.aws_ami.amazon_linux: Read complete after 1s [id=ami-009ce8169fc88edf5]
module.networking.aws_internet_gateway.this: Refreshing state... [id=igw-01ca0e583213bc718]
module.networking.aws_subnet.this: Refreshing state... [id=subnet-0f4ce09fd93138b7f]
module.security.aws_security_group.nginx: Refreshing state... [id=sg-0c22531f5e2e5be5f]
module.networking.aws_route_table.this: Refreshing state... [id=rtb-01b3dc985c94023]
```

## 4.2 Output Configuration (5 marks)

Create comprehensive outputs in outputs.tf

```
~\OneDrive\Desktop\CC Labs\Assignment2 (2.334s)
  terraform output

  backend_servers_info = {
    "web-1" = {
      "private_ip" = "10.0.10.203"
      "public_ip" = "3.28.204.70"
    }
    "web-2" = {
      "private_ip" = "10.0.10.146"
      "public_ip" = "158.252.81.36"
    }
    "web-3" = {
      "private_ip" = "10.0.10.83"
      "public_ip" = "3.29.64.238"
    }
  }
  nginx_public_ip = "3.28.207.249"
```

```
{...} outputs.json > ...
```

```
1  {
2    "backend_servers_info": {
3      "sensitive": false,
4      "type": [
5        "object",
6        {
7          "web-1": [
8            "object",
9            {
10              "private_ip": "string",
11              "public_ip": "string"
12            }
13          ],
14          "web-2": [
15            "object",
16            {
17              "private_ip": "string",
18              "public_ip": "string"
19            }
20          ],
21          "web-3": [
22            "object",
23            {
24              "private_ip": "string",
25              "public_ip": "string"
26            }
27          ]
28        }
29      ],
30      "value": {
31        "web-1": {
32          "private_ip": "10.0.10.203",
33          "public_ip": "3.28.204.70"
34        },
35        "web-2": {
36          "private_ip": "10.0.10.146",
37          "public_ip": "159.252.91.26"
38      }
39    }
40  }
```

### 4.3 AWS Console Verification (5 marks)

Verify all resources in AWS Console.

Tasks:

- Verify VPC created
- Verify Subnet created
- Verify Internet Gateway attached
- Verify Route Table configured
- Verify Security Groups created with correct rules
- Verify all 4 EC2 instances running
- Verify Key Pairs created

vpc-062a6835cf68bb46f / prod-vpc		Actions ▾
<b>Details</b> <small>Info</small>		
VPC ID	<input type="checkbox"/> vpc-062a6835cf68bb46f	<b>State</b> <span>✓ Available</span>
DNS resolution	Enabled	<b>Tenancy</b> default
Main network ACL	<a href="#">acl-0592385c7119d7bae</a>	<b>Default VPC</b> No
IPv6 CIDR	-	<b>Network Address Usage metrics</b> Disabled
Encryption control ID	-	<b>Encryption control mode</b> -
Block Public Access	<span>⊖ Off</span>	<b>DNS hostnames</b> Disabled
DHCP option set	<a href="#">dopt-0df9d1f801b853f87</a>	<b>Main route table</b> <a href="#">rtb-02a0e12a59815e909</a>
IPv4 CIDR	10.0.0.0/16	<b>IPv6 pool</b> -
Route 53 Resolver DNS Firewall rule groups	-	<b>Owner ID</b> <input type="checkbox"/> 249471344514

# subnet-0f4ce09fd93138b7f / prod-subnet

Actions ▾

## Details

Subnet ID	<input type="checkbox"/> subnet-0f4ce09fd93138b7f	Subnet ARN	<input type="checkbox"/> arn:aws:ec2:me-central-1:249471344514:subnet/subnet-0f4ce09fd93138b7f
IPv4 CIDR	<input type="checkbox"/> 10.0.10.0/24	Available IPv4 addresses	<input type="checkbox"/> 247
Availability Zone	<input type="checkbox"/> mec1-az1 (me-central-1a)	VPC	vpc-062a6835cf68bb46f   prod-vpc
Default subnet	No	Auto-assign public IPv4 address	Yes
IPv4 CIDR reservations	—	IPv6 CIDR reservations	—
Resource name DNS A record	Disabled	Resource name DNS AAAA record	Disabled
State	<input checked="" type="checkbox"/> Available	Block Public Access	<input type="radio"/> Off
IPv6 CIDR	—	IPv6 CIDR association ID	—
Route table	<a href="#">rtb-01b3dc985c94023e1</a>	Network ACL	<a href="#">acl-0592385c7119d7bae</a>
Auto-assign IPv6 address	No	Outpost ID	—
IPv6-only	No	Hostname type	IP name
DNS64	—	Owner	—

Route tables (4) <a href="#">Info</a>		
Last updated 1 minute ago		
<a href="#">Actions</a> ▾ <a href="#">Create route table</a>		
<input type="text"/> Find route tables by attribute or tag		
Name	Route table ID	Explicit subnet
<input type="checkbox"/> -	<a href="#">rtb-0aa33b96fa897436c</a>	-
<input type="checkbox"/> -	<a href="#">rtb-01b3dc985c94023e1</a>	<a href="#">subnet-0f4ce09</a>
<input type="checkbox"/> -	<a href="#">rtb-0155e1cff4ee9fb20</a>	-
<input type="checkbox"/> -	<a href="#">rtb-02a0e12a59815e909</a>	-

igw-01ca0e583213bc718 / prod-igw		<a href="#">Actions</a> ▾
Details <a href="#">Info</a>		
Internet gateway ID	<input checked="" type="checkbox"/> igw-01ca0e583213bc718	State <span style="color: green;">✓ Attached</span>
VPC ID	<a href="#">vpc-062a6835cf68bb46f</a>   <a href="#">prod-vpc</a>	Owner <input type="checkbox"/> 249471344514
Tags (4)		
<input type="text"/> Search tags		
Key	Value	
ManagedBy	Terraform	
Project	Assignment-2	
Name	prod-igw	
Environment	prod	

## sg-0c22531f5e2e5be5f - prod-nginx-sg

Actions ▾

### Details

**Security group name**

prod-nginx-sg

**Security group ID**

sg-0c22531f5e2e5be5f

**Description**

Security group for Nginx reverse proxy

**VPC ID**

[vpc-062a6835cf68bb46f ↗](#)

**Owner**

249471344514

**Inbound rules count**

3 Permission entries

**Outbound rules count**

1 Permission entry

**Inbound rules**

**Outbound rules**

**Sharing**

**VPC associations**

**Tags**

## sg-0d52409f0959ff8e3 - prod-backend-sg

Actions ▾

### Details

**Security group name**

prod-backend-sg

**Security group ID**

sg-0d52409f0959ff8e3

**Description**

Security group for backend servers

**VPC ID**

[vpc-062a6835cf68bb46f ↗](#)

**Owner**

249471344514

**Inbound rules count**

2 Permission entries

**Outbound rules count**

1 Permission entry

Instances (4) <a href="#">Info</a>						
<a href="#">Connect</a>	Instance state ▾	Actions ▾	Launch instances	▼		
<input type="text"/> Find Instance by attribute or tag (case-sensitive)				All states ▾		
<input type="checkbox"/>	Name ↴	Instance ID	Instance state	Instance type		
<input type="checkbox"/>	prod-web-2	i-09df089f29185b455	<span>✓ Running</span> <a href="#">+ </a> <a href="#">-</a>	t3.micro		
<input type="checkbox"/>	prod-web-3	i-0e55f5d49a75862d3	<span>✓ Running</span> <a href="#">+ </a> <a href="#">-</a>	t3.micro		
<input type="checkbox"/>	prod-web-1	i-060422b95ce8e430e	<span>✓ Running</span> <a href="#">+ </a> <a href="#">-</a>	t3.micro		
<input type="checkbox"/>	prod-nginx-pr...	i-0974031199bc82f41	<span>✓ Running</span> <a href="#">+ </a> <a href="#">-</a>	t3.micro		

## Part 5: Nginx Configuration & Testing (25 marks)

### 5.1 Update Nginx Backend Configuration (5 marks)

SSH into the Nginx server and update the configuration with actual backend IPs.

#### Tasks:

- SSH into Nginx server
- Edit /etc/nginx/nginx.conf
- Replace placeholder IPs with actual private IPs of backend servers
- Test Nginx configuration
- Restart Nginx service

```
~\OneDrive\Desktop\CC Labs\Assignment2
ssh ec2-user@3.28.207.249

The authenticity of host '3.28.207.249 (3.28.207.249)' can't be established.
ED25519 key fingerprint is SHA256:fbVKLetSx5LWfra6e7ejUnzCXQqZ2Y/gP+XAv91R4zU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.28.207.249' (ED25519) to the list of known hosts.

,
  #
  ~\_ ####_
  ~~ \| ##### \
  ~~ \###|
  ~~   \#/   Amazon Linux 2023 (ECS Optimized)
  ~~     V~' '-->
  ~~
  ~~\_/
  ~~\_/
  _/m/' 

For documentation, visit http://aws.amazon.com/documentation/ecs
[ec2-user@ip-10-0-10-249 ~]$
```

```
GNU nano 8.3          /etc/nginx/nginx.conf      Modified
events {}

http {
    upstream backend_pool {
        server 10.0.10.203:80;
        server 10.0.10.146:80;
        server 10.0.10.83:80 backup;
    }

    proxy_cache_path /var/cache/nginx levels=1:2 keys_zone=mycache:10m inactive=60m;
    server {
        listen 80;

        location / {
            proxy_pass http://backend_pool;
            proxy_set_header Host $host;
            proxy_cache mycache;
            add_header X-Cache-Status $upstream_cache_status;
        }
    }
}
```

```
[ec2-user@ip-10-0-10-249 ~]$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
[ec2-user@ip-10-0-10-249 ~]$
```

```
[ec2-user@ip-10-0-10-249 ~]$ sudo systemctl restart nginx
[ec2-user@ip-10-0-10-249 ~]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
   Active: active (running) since Mon 2025-12-29 21:15:03 UTC; 21s ago
     Process: 48557 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
     Process: 48558 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
     Process: 48559 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 48560 (nginx)
      Tasks: 4 (limit: 1065)
        Memory: 3.0M
          CPU: 31ms
        CGroup: /system.slice/nginx.service
                  ├─48560 "nginx: master process /usr/sbin/nginx"
                  ├─48561 "nginx: worker process"
                  ├─48562 "nginx: cache manager process"
                  └─48563 "nginx: cache loader process"

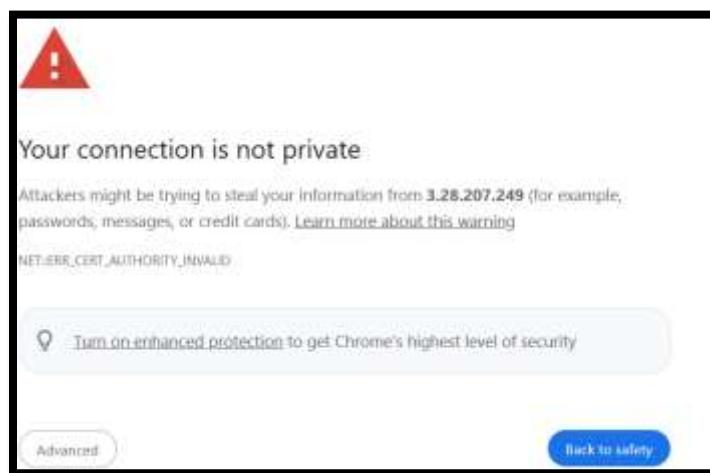
Dec 29 21:15:03 ip-10-0-10-249.me-central-1.compute.internal systemd[1]: Starting >
Dec 29 21:15:03 ip-10-0-10-249.me-central-1.compute.internal nginx[48558]: nginx: >
Dec 29 21:15:03 ip-10-0-10-249.me-central-1.compute.internal nginx[48558]: nginx: >
Dec 29 21:15:03 ip-10-0-10-249.me-central-1.compute.internal systemd[1]: Started n>
lines 1-20/20 (END)
```

## 5.2 Test Load Balancing (5 marks)

Test that Nginx is properly load balancing between web-1 and web-2.

### Tasks:

- Open browser to `https://<nginx-public-ip>`
- Accept the security warning for self-signed certificate
- Reload page multiple times (at least 10 times)
- Verify traffic alternates between web-1 and web-2
- Verify web-3 is NOT serving traffic (it's backup only)







## Backend Web Server - Assignment 2

**Hostname:** myapp-webserver

**Instance ID:** i-09df089f29185b455

**Private IP:** 10.0.10.146

**Public IP:** 158.252.81.36

**Public DNS:**

**Deployed:** Mon Dec 29 19:47:55 UTC 2025

**Status:**  Active and Running

**Managed By:** Terraform

### 5.3 Test Cache Functionality (5 marks)

Verify that Nginx caching is working correctly.

#### Tasks:

- Open browser developer tools (F12)
- Navigate to Network tab
- Clear browser cache
- Load <https://<nginx-public-ip>>
- Check response headers for X-Cache-Status: MISS (first request)

- Reload page
- Check response headers for X-Cache-Status: HIT (cached request)
- Verify cache directory on Nginx server

```
[ec2-user@ip-10-0-10-249 ~]$ sudo ls -la /var/cache/nginx/
total 0
drwx----- 2 nginx root 6 Dec 29 19:47 .
drwxr-xr-x 9 root root 94 Dec 29 19:47 ..
ls: cannot access 'nginx': No such file or directory
```

X Headers Preview Response Initiator >

▼ General

Request URL	http://3.28.207.249/
Request Method	GET
Status Code	200 OK
Remote Address	3.28.207.249:80
Referrer Policy	strict-origin-when-cross-origin

▼ Response Headers  Raw

Accept-Ranges	bytes
Connection	keep-alive
Content-Length	1534
Content-Type	text/html; charset=UTF-8
Date	Tue, 30 Dec 2025 09:01:47 GMT
Etag	"5fe-6471c84f87270"
Last-Modified	Mon, 29 Dec 2025 19:47:55 GMT
Server	nginx/1.28.0
X-Cache-Status	MISS

▼ Request Headers  Raw

Headers	
Request URL	http://3.28.207.249/
Request Method	GET
Status Code	304 Not Modified
Remote Address	3.28.207.249:80
Referrer Policy	strict-origin-when-cross-origi
▼ Response Headers	<input type="checkbox"/> Raw
Connection	keep-alive
Date	Tue, 30 Dec 2025 10:00:49 GMT
Etag	"5fc-6471c84fb0feb"
Last-Modified	Mon, 29 Dec 2025 19:47:55 GMT
Server	nginx/1.28.0
X-Cache-Status	HIT
▼ Request Headers	<input type="checkbox"/> Raw
Accept	text/html,application/xhtml+xe ml,application/xml;q=0.9,image/ avif,image/webp,image/apn ,*/*;q=0.8,application/signed- exchange;v=b3;q=0.7

```
[ec2-user@ip-10-0-10-249 ~]$ sudo ls -la /var/cache/nginx  
total 0  
drwxr-xr-x. 3 nginx nginx 15 Dec 30 10:00 .  
drwxr-xr-x. 9 root  root  94 Dec 29 19:47 ..  
drwx----- 3 nginx nginx 16 Dec 30 10:00 6
```

```
[ec2-user@ip-10-0-10-249 ~]$ sudo tail -f /var/log/nginx/access.log
103.252.91.84 - - [30/Dec/2025:09:46:54 +0000] "CONNECT google.com:443 HTTP/1.1" 400 157 "-" "-"
103.53.162.0 - - [30/Dec/2025:09:54:25 +0000] "GET / HTTP/1.1" 200 1532 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:09:54:42 +0000] "GET / HTTP/1.1" 200 1534 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:09:54:47 +0000] "GET / HTTP/1.1" 200 1532 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:09:56:22 +0000] "GET / HTTP/1.1" 200 1534 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:09:56:28 +0000] "GET / HTTP/1.1" 200 1532 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
185.242.226.110 - - [30/Dec/2025:09:57:43 +0000] "\x16\x03\x01\x00\x8A\x01\x00\x00\x86\x03\x03\x9D6\x09U\xBA\xC9\xF8Lk\x0F\xF8\xB9\x06\xD4\x9A\xC2\xC90\x94\xB4\xE6\xF8\xD8f\xAF\x850\x1Fi\xDD\xE8\xCD\x00\x00\x1A\xC0/\xC0+\xC0\x11\xC0\x07\xC0\x13\xC0\x09\xC0\x14\xC0" 400 157 "-" "-"
103.53.162.0 - - [30/Dec/2025:10:00:44 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:10:00:49 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:10:01:22 +0000] "GET / HTTP/1.1" 200 1532 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
```

#### 5.4 Test High Availability (Backup Server) (5 marks)

Test the backup server functionality by simulating primary server failure.

##### Tasks:

- SSH into web-1 and stop Apache
- Reload Nginx page - should show web-2 only
- SSH into web-2 and stop Apache
- Reload Nginx page - should now show web-3 (backup activated)
- Restart web-1 and web-2
- Verify traffic returns to web-1 and web-2

```
ssh ec2-user@3.28.204.70
```

```
The authenticity of host '3.28.204.70 (3.28.204.70)' can't be established.  
ED25519 key fingerprint is SHA256:UqqCw5aL7R9NUuF/bqDD6xMpYj/zpFuN+kBaFI51BqQ.  
This key is not known by any other names.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added '3.28.204.70' (ED25519) to the list of known hosts.
```

```
,      #  
~\_\_ #####  
~~ \#####\  
~~ \###|  
~~  \#/ _-- Amazon Linux 2023 (ECS Optimized)  
~~   V~' '-'>  
~~~  
~~. . /  
~/'
```

```
For documentation, visit http://aws.amazon.com/documentation/ecs
```

```
[ec2-user@myapp-webserver ~]$ sudo systemctl stop httpd
```

```
[ec2-user@myapp-webserver ~]$ sudo systemctl status httpd
```

```
○ httpd.service - The Apache HTTP Server
```

```
    Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)  
      Active: inactive (dead) since Tue 2025-12-30 10:07:52 UTC; 16s ago  
        Duration: 14h 19min 55.591s  
          Docs: man:httpd.service(8)  
        Process: 2159 ExecStart=/usr/sbin/httpd $OPTIONS -DFOREGROUND (code=exited, st  
      Main PID: 2159 (code=exited, status=0/SUCCESS)  
        Status: "Total requests: 225; Idle/Busy workers 100/0;Requests/sec: 0.00436; B  
        CPU: 57.422s
```

```
Dec 29 19:47:55 ip-10-0-10-203.me-central-1.compute.internal systemd[1]: Starting The Apache H  
Dec 29 19:47:55 ip-10-0-10-203.me-central-1.compute.internal httpd[2159]: Server c  
Dec 29 19:47:55 ip-10-0-10-203.me-central-1.compute.internal systemd[1]: Started The Apache H  
Dec 30 10:07:51 myapp-webserver systemd[1]: Stopping httpd.service - The Apache HT  
Dec 30 10:07:52 myapp-webserver systemd[1]: httpd.service: Deactivated successfully  
Dec 30 10:07:52 myapp-webserver systemd[1]: Stopped httpd.service - The Apache HT  
Dec 30 10:07:52 myapp-webserver systemd[1]: httpd.service: Consumed 57.422s CPU time
```

```
ssh ec2-user@158.252.81.36
The authenticity of host '158.252.81.36 (158.252.81.36)' can't be established.
ED25519 key fingerprint is SHA256:Yx8+W230ruCYeKN57IMj+xpPhHnSl8mddk8vTE948+g.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '158.252.81.36' (ED25519) to the list of known hosts.

      #_
~\_ #####_
~~ \#####\
~~ \###|
~~   \#/ ___ Amazon Linux 2023 (ECS Optimized)
~~     V~, '->
~~~   /
~~.._ /_
~/ _/
~/m/'

For documentation, visit http://aws.amazon.com/documentation/ecs
[ec2-user@myapp-webserver ~]$ sudo systemctl stop httpd
[ec2-user@myapp-webserver ~]$ sudo systemctl status httpd
○ httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
   Active: inactive (dead) since Tue 2025-12-30 10:10:19 UTC; 20s ago
     Duration: 14h 22min 22.599s
       Docs: man:httpd.service(8)
    Process: 2155 ExecStart=/usr/sbin/httpd $OPTIONS -DFOREGROUND (code=exited, st
 Main PID: 2155 (code=exited, status=0/SUCCESS)
    Status: "Total requests: 219; Idle/Busy workers 100/0;Requests/sec: 0.00423;
      CPU: 46.999s

Dec 29 19:47:55 ip-10-0-10-146.me-central-1.compute.internal systemd[1]: Starting
Dec 29 19:47:55 ip-10-0-10-146.me-central-1.compute.internal systemd[1]: Started h
Dec 29 19:47:55 ip-10-0-10-146.me-central-1.compute.internal httpd[2155]: Server c
Dec 30 10:10:18 myapp-webserver systemd[1]: Stopping httpd.service - The Apache HT
Dec 30 10:10:19 myapp-webserver systemd[1]: httpd.service: Deactivated successfully
Dec 30 10:10:19 myapp-webserver systemd[1]: Stopped httpd.service - The Apache HTT
Dec 30 10:10:19 myapp-webserver systemd[1]: httpd.service: Consumed 46.999s CPU ti
```

```
ssh ec2-user@3.29.64.238
The authenticity of host '3.29.64.238 (3.29.64.238)' can't be established.
ED25519 key fingerprint is SHA256:ec7ElnOM8f+0KtDDCHb9dRgA08LvP8LFK4KF1TTeX9o.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.29.64.238' (ED25519) to the list of known hosts.

      _#
     ~\_ #####
     ~~ \#####\
     ~~  \###|
     ~~   \#/  -- Amazon Linux 2023 (ECS Optimized)
     ~~   \~'  '->
     ~~~   / \
     ~~.~. / \
     /m/ \
For documentation, visit http://aws.amazon.com/documentation/ecs
[ec2-user@myapp-webserver ~]$ sudo systemctl start httpd
[ec2-user@myapp-webserver ~]$ sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
   Active: active (running) since Mon 2025-12-29 19:47:56 UTC; 14h ago
     Docs: man:httpd.service(8)
 Main PID: 2158 (httpd)
   Status: "Total requests: 52; Idle/Busy workers 100/0;Requests/sec: 0.000999; >
    Tasks: 230 (limit: 1065)
   Memory: 17.8M
      CPU: 1min 4.606s
    CGroup: /system.slice/httpd.service
            └─ 2158 /usr/sbin/httpd -DFOREGROUND
              ├─ 2159 /usr/sbin/httpd -DFOREGROUND
              ├─ 2160 /usr/sbin/httpd -DFOREGROUND
              ├─ 2161 /usr/sbin/httpd -DFOREGROUND
              ├─ 2163 /usr/sbin/httpd -DFOREGROUND
              └─ 38294 /usr/sbin/httpd -DFOREGROUND

Dec 29 19:47:56 ip-10-0-10-83.me-central-1.compute.internal systemd[1]: Starting httpd...
Dec 29 19:47:56 ip-10-0-10-83.me-central-1.compute.internal systemd[1]: Started httpd.
Dec 29 19:47:56 ip-10-0-10-83.me-central-1.compute.internal httpd[2158]: Server co...
```

```
ssh ec2-user@3.28.207.249
```

```
,      #
~\_ ####_
~~ \#####\
~~  \###|
~~   \#/ ___ Amazon Linux 2023 (ECS Optimized)
~~    V~' '-->
~~     /
~~_. /_/
/_m/'
```

```
For documentation, visit http://aws.amazon.com/documentation/ecs
```

```
Last login: Tue Dec 30 10:11:43 2025 from 103.53.162.0
```

```
[ec2-user@ip-10-0-10-249 ~]$ sudo tail -f /var/log/nginx/error.log
```

```
2025/12/29 20:42:18 [error] 1652#1652: *27 upstream timed out (110: Connection time  
d out) while connecting to upstream, client: 154.192.136.50, server: , request: "GE
```

```
T / HTTP/1.1", upstream: "http://10.0.1.12:80/", host: "3.28.207.249"
```

```
2025/12/29 20:43:18 [error] 1652#1652: *27 upstream timed out (110: Connection time  
d out) while connecting to upstream, client: 154.192.136.50, server: , request: "GE
```

```
T /favicon.ico HTTP/1.1", upstream: "http://10.0.1.10:80/favicon.ico", host: "3.28.  
207.249", referrer: "http://3.28.207.249/"
```

```
2025/12/29 20:44:19 [error] 1652#1652: *27 upstream timed out (110: Connection time  
d out) while connecting to upstream, client: 154.192.136.50, server: , request: "GE
```

```
T /favicon.ico HTTP/1.1", upstream: "http://10.0.1.11:80/favicon.ico", host: "3.28.  
207.249", referrer: "http://3.28.207.249/"
```

```
2025/12/29 20:45:19 [error] 1652#1652: *27 upstream timed out (110: Connection time  
d out) while connecting to upstream, client: 154.192.136.50, server: , request: "GE
```

```
T /favicon.ico HTTP/1.1", upstream: "http://10.0.1.12:80/favicon.ico", host: "3.28.  
207.249", referrer: "http://3.28.207.249/"
```

```
2025/12/30 09:50:32 [notice] 648394#648394: signal process started
```

```
2025/12/30 09:51:11 [notice] 649171#649171: signal process started
```

```
2025/12/30 09:53:48 [notice] 651439#651439: signal process started
```

```
2025/12/30 09:59:44 [notice] 656978#656978: signal process started
```

```
2025/12/30 10:17:27 [error] 656979#656979: *745 connect() failed (111: Connection r  
efused) while connecting to upstream, client: 103.53.162.0, server: , request: "GET
```

```
 / HTTP/1.1", upstream: "http://10.0.10.146:80/", host: "3.28.207.249"
```

```
2025/12/30 10:17:27 [error] 656979#656979: *745 connect() failed (111: Connection r  
efused) while connecting to upstream, client: 103.53.162.0, server: , request: "GET
```

```
 / HTTP/1.1", upstream: "http://10.0.10.203:80/", host: "3.28.207.249"
```

```
~\OneDrive\Desktop\CC Labs\Assignment2 (0.091s)
ssh ec2-user@3.28.204.70
  ~~~ \###|
  ~~~ \#/ ___ Amazon Linux 2023 (ECS Optimized)
  ~~~ V~' .->
  ~~~ /
  ~~.~. /_
  _/_/
  _/m/.'

For documentation, visit http://aws.amazon.com/documentation/ecs
Last login: Tue Dec 30 10:07:35 2025 from 103.53.162.0
[ec2-user@myapp-webserver ~]$ sudo systemctl start httpd
[ec2-user@myapp-webserver ~]$ exit
logout
Connection to 3.28.204.70 closed.
```

```
~\OneDrive\Desktop\CC Labs\Assignment2 (0.093s)
ssh ec2-user@*****
ssh: Could not resolve hostname *****: No such host is known.
```

```
~\OneDrive\Desktop\CC Labs\Assignment2 (0.045s)
```

```
~\OneDrive\Desktop\CC Labs\Assignment2 (38.947s)
ssh ec2-user@158.252.81.36
  , #
  ~\_ ####_
  ~\_\####\_
  ~~ \###|
  ~~ \#/ ___ Amazon Linux 2023 (ECS Optimized)
  ~~ V~' .->
  ~~~ /
  ~~.~. /_
  _/_/
  _/m/.'

For documentation, visit http://aws.amazon.com/documentation/ecs
Last login: Tue Dec 30 10:09:57 2025 from 103.53.162.0
[ec2-user@myapp-webserver ~]$ sudo systemctl start httpd
sudo: systemctl: command not found
[ec2-user@myapp-webserver ~]$ sudo systemctl start httpd
[ec2-user@myapp-webserver ~]$ exit
logout
Connection to 158.252.81.36 closed.
```

## 5.5 Security & Performance Analysis (5 marks)

Analyze the security headers and performance of your Nginx setup.

**Tasks:**

- Check SSL/TLS certificate details
- Verify security headers in response
- Test HTTP to HTTPS redirect
- Check response times
- Analyze Nginx logs

Ssl certificate

```
Anara Hayat@DESKTOP-BLT0JUH MINGW64 ~/OneDrive/Desktop/CC Labs/Assignment2
$ openssl s_client -connect 3.28.207.249:443
Connecting to 3.28.207.249
CONNECTED(0000015C)
Can't use SSL_get_servername
depth=0 C=US, ST=State, L=City, O=Organization, OU=IT, CN=ip-10-0-10-249.me-central-1.compute.internal
verify error:num=18:self-signed certificate
verify return:1
depth=0 C=US, ST=State, L=City, O=Organization, OU=IT, CN=ip-10-0-10-249.me-central-1.compute.internal
verify return:1
---
Certificate chain
 0 s:C=US, ST=State, L=City, O=Organization, OU=IT, CN=ip-10-0-10-249.me-central-1.compute.internal
    i:C=US, ST=State, L=City, O=Organization, OU=IT, CN=ip-10-0-10-249.me-central-1.compute.internal
      a:PKEY: rsaEncryption, 2048 (bit); sigalg: RSA-SHA256
      v:NotBefore: Dec 30 11:00:57 2025 GMT; NotAfter: Dec 30 11:00:57 2026 GMT
---
Server certificate
-----BEGIN CERTIFICATE-----
MIID8TCCAtmgAwIBAgIUZyxwVwh9orYV9KjIib83P0Lq1R8wDQYJKoZIhvcNAQEL
BQAwgYcxCzAJBgNVBAYTA1VTMQ4wDAYDVQQIDAVTdGF0ZTENMAsGA1UEBwwEQ210
eTEVMBMGA1UECgwMT3JnYW5pemF0aw9uMQswCQYDVQQLDAJJVDE1MDMGA1UEAwws
aXAtMTAtMC0xMC0yNDkubWUtY2VudHJhbC0xLmNvbXB1dGUuaw50ZXJuYWhwHhcN
MjUxMjMwMTEwMDU3WhcNMjYxMjMwMTEwMDU3WjCBhzELMAkGA1UEBhMCVVMxDjAM
BgNVBAgMBVNOYXR1MQ0wCwYDVQQHDARDaXR5MRUwEwYDVQQKDAxPcmdhbm16YXRp
b24xCzAJBgNVBAAsMAk1UMTuMwYDVQQDDCxpcc0xMC0wLTEwLTI0OS5tZS1jZW50
cmFsLTEuY29tcHV0ZS5pbnR1cm5hbDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
```

```
[ec2-user@ip-10-0-10-249 ~]$ sudo openssl x509 -in /etc/ssl/certs/selfsigned.crt -text -noout
Certificate:
Data:
    Version: 3 (0x2)
    Serial Number:
        67:2c:70:57:08:7d:a2:b6:15:f4:a8:e5:21:bf:37:3f:42:ea:95:1f
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: C=US, ST=State, L=City, O=Organization, OU=IT, CN=ip-10-0-10-249.m
-e-central-1.compute.internal
    Validity
        Not Before: Dec 30 11:00:57 2025 GMT
        Not After : Dec 30 11:00:57 2026 GMT
    Subject: C=US, ST=State, L=City, O=Organization, OU=IT, CN=ip-10-0-10-249.m
e-central-1.compute.internal
    Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
        Public-Key: (2048 bit)
            Modulus:
                00:b3:42:09:d0:8c:18:30:3d:5c:18:d6:7f:c8:4f:
                63:30:a7:37:ea:38:a7:73:dc:3f:f0:cb:c4:15:3c:
                35:74:25:83:6b:2e:79:9f:b7:79:c8:93:6b:b7:63:
                1c:68:b2:ae:20:6e:b2:a8:7b:5e:1d:60:1c:1f:7b:
                05:41:3f:67:cc:3d:77:ea:e5:3b:a9:f4:a1:06:fd:
                7f:60:5c:5c:d5:e8:5c:97:de:4b:43:f9:a1:c9:65:
                ea:8d:e4:39:e0:6f:e8:fb:bd:1a:cb:2c:88:0c:e8:
                cc:23:35:24:6e:07:74:b4:4a:92:16:4a:73:95:95:
                86:ab:b1:26:4f:1c:18:1a:4a:41:d8:72:a9:86:7e:
                0e:32:df:7d:2a:8e:b3:97:71:52:13:71:66:fe:c4:
                40:e7:0c:aa:10:fe:92:d3:ff:34:f8:64:70:90:3d:
                f5:62:55:14:fa:5a:cb:b8:e3:89:18:fb:3e:ab:b1:
                12:26:b8:c4:5a:79:9a:e4:33:18:88:16:4e:7c:fb:
                84:5c:53:ae:c2:4f:5b:56:20:dd:6b:46:be:42:7a:
                56:b8:ef:4f:15:39:a5:31:4e:65:5c:27:2c:c2:07:
                41:42:46:84:89:e0:30:51:26:10:1f:7d:25:59:c5:
                41:42:46:84:89:e0:30:51:26:10:1f:7d:25:59:c5:
```

## Security header

```
Anara Hayat@DESKTOP-BLT0JUH MINGW64 ~/OneDrive/Desktop/CC Labs/
$ curl -I -k https://3.28.207.249
HTTP/1.1 200 OK
Server: nginx/1.28.0
Date: Tue, 30 Dec 2025 11:24:10 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 1532
Connection: keep-alive
Last-Modified: Mon, 29 Dec 2025 19:47:55 GMT
ETag: "5fc-6471c84fb0feb"
X-Cache-Status: HIT
Accept-Ranges: bytes
```

## Test http redirect

```
Anara Hayat@DESKTOP-BLT0JUH MINGW64 ~/OneDrive/Desktop/CC Labs/Assignment2
$ curl -Ik https://3.28.207.249
HTTP/1.1 200 OK
Server: nginx/1.28.0
Date: Tue, 30 Dec 2025 11:22:01 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 1532
Connection: keep-alive
Last-Modified: Mon, 29 Dec 2025 19:47:55 GMT
ETag: "5fc-6471c84fb0feb"
X-Cache-Status: MISS
Accept-Ranges: bytes
```

## Monitor error logs

```
ssh ec2-user@3.28.207.249
[ec2-user@ip-10-0-10-249 ~]$ sudo tail -50 /var/log/nginx/error.log
2025/12/29 20:04:09 [error] 2029#2029: *23 upstream timed out (110: Connection time
d out) while connecting to upstream, client: 154.192.136.50, server: , request: "GE
T / HTTP/1.1", upstream: "http://10.0.1.10:80/", host: "3.29.231.86"
2025/12/29 20:04:22 [error] 2029#2029: *17 upstream timed out (110: Connection time
d out) while connecting to upstream, client: 154.192.136.50, server: , request: "GE
T /favicon.ico HTTP/1.1", upstream: "http://10.0.1.11:80/favicon.ico", host: "3.29.
231.86", referrer: "http://3.29.231.86/"
2025/12/29 20:05:22 [error] 2029#2029: *17 upstream timed out (110: Connection time
d out) while connecting to upstream, client: 154.192.136.50, server: , request: "GE
T /favicon.ico HTTP/1.1", upstream: "http://10.0.1.12:80/favicon.ico", host: "3.29.
231.86", referrer: "http://3.29.231.86/"
2025/12/29 20:10:34 [error] 2029#2029: *29 upstream timed out (110: Connection time
d out) while connecting to upstream, client: 154.192.136.50, server: , request: "GE
T / HTTP/1.1", upstream: "http://10.0.1.10:80/", host: "3.29.231.86"
2025/12/29 20:11:24 [error] 2029#2029: *32 upstream timed out (110: Connection time
d out) while connecting to upstream, client: 185.16.39.146, server: , request: "GET
/ HTTP/1.1", upstream: "http://10.0.1.10:80/", host: "3.29.231.86"
2025/12/29 20:11:34 [error] 2029#2029: *29 upstream timed out (110: Connection time
d out) while connecting to upstream, client: 154.192.136.50, server: , request: "GE
T / HTTP/1.1", upstream: "http://10.0.1.11:80/", host: "3.29.231.86"
2025/12/29 20:12:34 [error] 2029#2029: *29 upstream timed out (110: Connection time
d out) while connecting to upstream, client: 154.192.136.50, server: , request: "GE
T / HTTP/1.1", upstream: "http://10.0.1.12:80/", host: "3.29.231.86"
2025/12/29 20:13:34 [error] 2029#2029: *29 upstream timed out (110: Connection time
d out) while connecting to upstream, client: 154.192.136.50, server: , request: "GE
T /favicon.ico HTTP/1.1", upstream: "http://10.0.1.11:80/favicon.ico", host: "3.29.
231.86", referrer: "http://3.29.231.86/"
2025/12/29 20:14:34 [error] 2029#2029: *29 upstream timed out (110: Connection time
d out) while connecting to upstream, client: 154.192.136.50, server: , request: "GE
T /favicon.ico HTTP/1.1", upstream: "http://10.0.1.10:80/favicon.ico", host: "3.29.
231.86", referrer: "http://3.29.231.86/"
2025/12/29 20:15:34 [error] 2029#2029: *29 upstream timed out (110: Connection time
d out) while connecting to upstream, client: 154.192.136.50, server: , request: "GE
T /favicon.ico HTTP/1.1", upstream: "http://10.0.1.12:80/favicon.ico", host: "3.29.
231.86", referrer: "http://3.29.231.86/"
```

## Monitor access logs

```

ssh ec2-user@3.28.207.249
[ec2-user@ip-10-0-10-249 ~]$ sudo tail -50 /var/log/nginx/access.log
185.242.226.110 - - [30/Dec/2025:09:57:43 +0000] "\x16\x03\x01\x00\x8A\x01\x00\x00\x86\x03\x03\x9D6\x09U\xBA\xC9\xF8Lk\x0F\xF8\xB9\x06\xD4\x9A\xC2\xC90\x94\xB4\xE6\xF8\xD8f\xAF\x850\x1Fi\xDD\xE8\xCD\x00\x00\x1A\xC0/\xC0+\xC0\x11\xC0\x07\xC0\x13\xC0\x09\xC0\x14\xC0" 400 157 "-" "-"
103.53.162.0 - - [30/Dec/2025:10:00:44 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:10:00:49 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:10:01:22 +0000] "GET / HTTP/1.1" 200 1532 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:10:17:27 +0000] "GET / HTTP/1.1" 200 1531 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:10:17:29 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:10:17:32 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:10:17:33 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:10:17:35 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:10:17:36 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
45.156.128.49 - - [30/Dec/2025:10:27:54 +0000] "GET / HTTP/1.1" 200 1532 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.113 Safari/537.36"
103.53.162.0 - - [30/Dec/2025:10:41:12 +0000] "OPTIONS /* HTTP/1.1" 200 0 "-" "Microsoft-WebDAV-MiniRedir/10.0.19045"

```

Check Nginx worker processes

```

[ec2-user@ip-10-0-10-249 ~]$ ps aux | grep nginx
root      1649  0.0  0.6 26880  6524 ?        Ss   Dec29   0:00 nginx: master pr
ocess /usr/sbin/nginx
nginx    717635  0.0  0.7 27060  6920 ?        S    11:07   0:00 nginx: worker pr
ocess
nginx    717636  0.0  0.3 27060  3452 ?        S    11:07   0:00 nginx: cache man
ager process
ec2-user  738931  0.0  0.2 222328  2116 pts/0     S+   11:30   0:00 grep --color=aut
o nginx
[ec2-user@ip-10-0-10-249 ~]$ 

```

## Bonus Tasks

### Bonus 1: Custom Error Pages

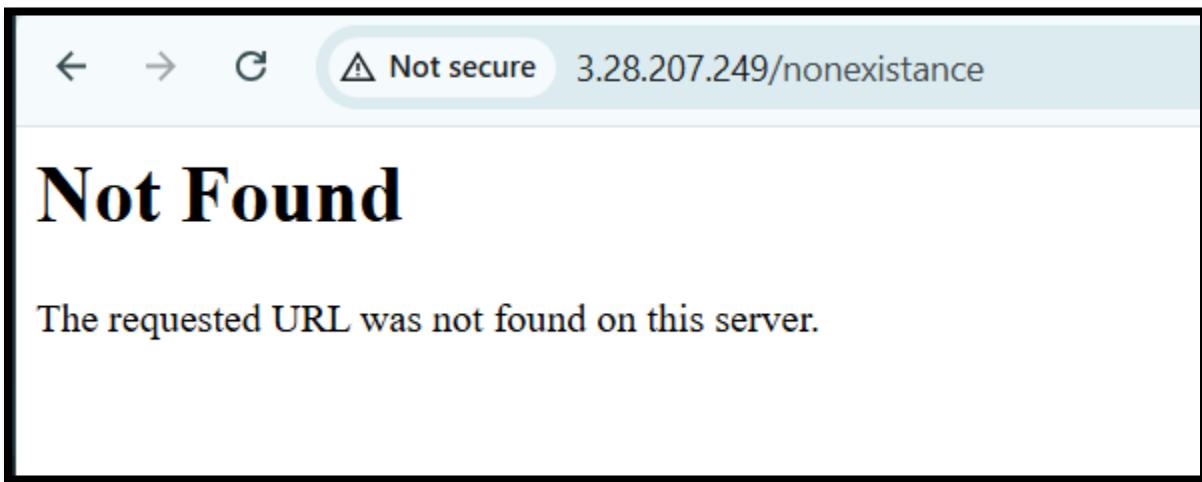
Create custom error pages for Nginx (404, 502, 503).

**Tasks:**

- Create custom HTML error pages
- Configure Nginx to use custom error pages
- Test by accessing non-existent URL and stopping backend servers

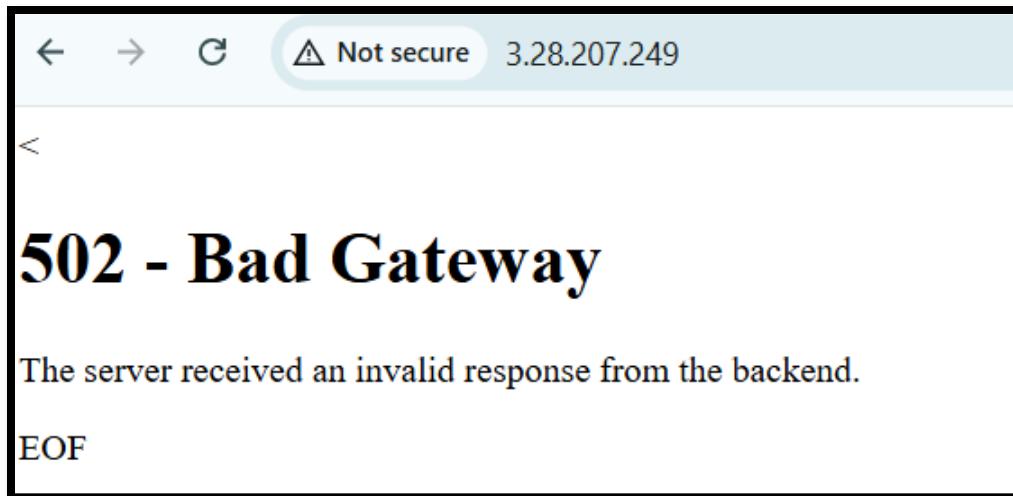
404

```
Anara Hayat@DESKTOP-BLTOJUH MINGW64 ~/OneDrive/Desktop/CC Labs/Assignment2
$ curl -I http://3.28.207.249/nonexistence
HTTP/1.1 404 Not Found
Server: nginx/1.28.0
Date: Tue, 30 Dec 2025 11:54:58 GMT
Content-Type: text/html; charset=iso-8859-1
Content-Length: 196
Connection: keep-alive
X-Cache-Status: MISS
```



502

```
Anara Hayat@DESKTOP-BLTOJUH MINGW64 ~/OneDrive/Desktop/CC Labs
$ curl -I -H "Cache-Control: no-cache" http://3.28.207.249/
HTTP/1.1 502 Bad Gateway
Server: nginx/1.28.0
Date: Tue, 30 Dec 2025 12:11:15 GMT
Content-Type: text/html
Content-Length: 192
Connection: keep-alive
ETag: "6953b934-c0"
```



## Bonus 2: Implement Rate Limiting (3 marks)

Add rate limiting to prevent abuse.

### Tasks:

- Implement rate limiting
- Test with rapid requests
- Show 429 (Too Many Requests) response

```
events{}  
http {  
    proxy_cache_path /var/cache/nginx levels=1:2 keys_zone=mycache:10m inactive=60m  
    # Define a shared memory zone for rate limiting  
    limit_req_zone $binary_remote_addr zone=mylimit:10m rate=10r/s;  
  
    upstream backend_pool {  
        server 10.0.10.203;  
        server 10.0.10.146;  
        server 10.0.10.83 backup;  
    }  
  
    server {  
        listen 80;  
        server_name _;  
  
        location / {  
            # Apply rate limiting  
            limit_req zone=mylimit burst=20 nodelay;  
  
            proxy_pass http://backend_pool;  
            proxy_set_header Host $host;  
  
            # Cache settings (optional)  
            proxy_cache mycache;  
            proxy_cache_key "$scheme$request_method$host$request_uri";  
            proxy_cache_valid 200 301 302 10m;  
            proxy_cache_valid 404 1m;  
            proxy_ignore_headers Cache-Control Expires Set-Cookie;  
  
            add_header X-Cache-Status $upstream_cache_status always;  
        }  
    }  
}
```

```
Anara Hayat@DESKTOP-BLTOJUH MINGW64 ~/OneDrive/Desktop/CC Labs/Assignment2
$ for i in {1..5}; do curl -I http://3.28.207.249/; done
HTTP/1.1 502 Bad Gateway
Server: nginx/1.28.0
Date: Tue, 30 Dec 2025 12:25:54 GMT
Content-Type: text/html
Content-Length: 157
Connection: keep-alive
X-Cache-Status: EXPIRED

HTTP/1.1 502 Bad Gateway
Server: nginx/1.28.0
Date: Tue, 30 Dec 2025 12:25:54 GMT
Content-Type: text/html
Content-Length: 157
Connection: keep-alive
X-Cache-Status: EXPIRED

HTTP/1.1 502 Bad Gateway
Server: nginx/1.28.0
Date: Tue, 30 Dec 2025 12:25:55 GMT
Content-Type: text/html
Content-Length: 157
Connection: keep-alive
X-Cache-Status: EXPIRED

HTTP/1.1 502 Bad Gateway
Server: nginx/1.28.0
Date: Tue, 30 Dec 2025 12:25:55 GMT
Content-Type: text/html
Content-Length: 157
Connection: keep-alive
X-Cache-Status: EXPIRED

HTTP/1.1 502 Bad Gateway
Server: nginx/1.28.0
Date: Tue, 30 Dec 2025 12:25:55 GMT
Content-Type: text/html
Content-Length: 157
Connection: keep-alive
X-Cache-Status: EXPIRED
```

### Bonus 3: Health Check Automation

Create a shell script that monitors backend server health.

#### Script Requirements:

- Check each backend server every 30 seconds
- Log health status

- Alert if server is down
- Restart Apache if needed

```
GNU nano 8.3          /home/ec2-user/backend_health_check.sh
#!/bin/bash

# List of backend servers (private IPs)
BACKENDS=("10.0.10.203" "10.0.10.146" "10.0.10.83")

# Log file
LOG_FILE="/home/ec2-user/backend_health.log"

# Interval between checks (seconds)
INTERVAL=30

while true; do
    TIMESTAMP=$(date '+%Y-%m-%d %H:%M:%S')

    for SERVER in "${BACKENDS[@]}"; do
        # Check HTTP response
        RESPONSE=$(curl -s -o /dev/null -w "%{http_code}" http://$SERVER)

        if [ "$RESPONSE" == "200" ]; then
            echo "$TIMESTAMP - $SERVER is UP" >> $LOG_FILE
        else
            echo "$TIMESTAMP - $SERVER is DOWN (HTTP $RESPONSE)" >> $LOG_FILE

            # Restart Apache (requires SSH access to backend server)
            echo "$TIMESTAMP - Restarting Apache on $SERVER" >> $LOG_FILE
            ssh -o StrictHostKeyChecking=no ec2-user@$SERVER 'sudo systemctl restart httpd'
        fi
    done
    sleep $INTERVAL
done
```

^C  
[ec2-user@ip-10-0-10-249 ~]\$ tail -f /home/ec2-user/backend\_health.log  
2025-12-30 12:50:55 - 10.0.10.203 is DOWN (HTTP 000)  
2025-12-30 12:50:55 - Restarting Apache on 10.0.10.203

## Part 6: Documentation & Cleanup

### 6.1 README Documentation

Create a comprehensive README.md

```
① README.md X ▶ ⌂
```

nd > abc # Assignment 2 - Multi-Tier Web Infrastructure > abc ## 3. Deployment In

1 # Assignment 2 - Multi-Tier Web Infrastructure

2

3 ## 1. Project Overview

4 This project implements a multi-tier web

infrastructure using AWS, Terraform, and Nginx.

5 The architecture includes:

6

7 - Nginx as a reverse proxy and load balancer with

caching.

8 - Multiple backend web servers (primary and backup).

9 - Security groups for network access control.

10 - Automated health checks and rate limiting.

11

12 ## 1.1. Architecture Diagram

13

14 [Internet]

15

16

17 | HTTPS (443)

18 | HTTP (80)

19 | ▼

20

21 [Nginx Server (Load Balancer)]

22 - SSL/TLS

23 - Caching

24 - Reverse Proxy

25

26

27

28

29 | ▼

30 | ▼

31 | ▼

32 [Web-1]

33 [Web-2]

34 [Web-3 (BKP)]

## ## 1.2. Components Description

Component	Description
Nginx Server	Reverse proxy, load balancer, caching, SSL/TLS termination
Web Servers	Backend application servers (HTTPD)
Security Groups	Control inbound/outbound access
Terraform	Infrastructure as code (provisioning EC2, SG, etc.)
Health Check	Shell script monitoring backend server health
Rate Limiting	Prevents abuse of web servers (429 responses)

---

## ## 2. Prerequisites

### - **\*\*Required Tools\*\*:**

- Terraform >= 1.5
- AWS CLI
- SSH client
- cURL (for testing)

### - **\*\*AWS Setup\*\*:**

- Configure credentials: `aws configure`
- Ensure IAM user has EC2, VPC, SecurityGroup permissions

### - **\*\*SSH Key Setup\*\*:**

- Generate key pair: `ssh-keygen -t ed25519`
- Store private key securely
- Public key used in Terraform EC2 provisioning

## 6.2 Infrastructure Cleanup (5 marks)

Properly destroy all resources and verify cleanup.

Tasks:

- Run terraform destroy
- Confirm resource deletion in AWS Console
- Verify no remaining resources
- Document the destruction process

```
~\OneDrive\Desktop\CC Labs\Assignment2
terraform destroy
- nginx_instance_id      = "i-0974031199bc82f41" -> null
- nginx_public_ip        = "3.28.207.249" -> null
- subnet_id               = "subnet-0f4ce09fd93138b7f" -> null
- vpc_id                  = "vpc-062a6835cf68bb46f" -> null

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: [red box]
```

```
~\OneDrive\Desktop\CC Labs\Assignment2 (54.09.15)
terraform destroy -auto-approve
module.backend_servers["web-2"].aws_instance.this: Still destroying... [id=i-09df9f29185b455, 00m40s elapsed]
module.backend_servers["web-1"].aws_instance.this: Still destroying... [id=i-0604b95ce8e430e, 00m40s elapsed]
module.backend_servers["web-1"].aws_instance.this: Destruction complete after 41s
module.backend_servers["web-2"].aws_instance.this: Destruction complete after 41s
module.backend_servers["web-2"].aws_key_pair.this: Destroying... [id=prod-2-key]
module.networking.aws_subnet.this: Destroying... [id=subnet-0f4ce09fd93138b7f]
module.security.aws_security_group.backend: Destroying... [id=sg-0d52409f0959ff8e]
module.backend_servers["web-1"].aws_key_pair.this: Destruction complete after 0s
module.backend_servers["web-2"].aws_key_pair.this: Destruction complete after 1s
module.networking.aws_subnet.this: Destruction complete after 1s
module.security.aws_security_group.backend: Destruction complete after 1s
module.security.aws_security_group.nginx: Destroying... [id=sg-0c22531f5e2e5be5f]
module.security.aws_security_group.nginx: Destruction complete after 1s
module.networking.aws_vpc.this: Destroying... [id=vpc-062a6835cf68bb46f]
module.networking.aws_vpc.this: Destruction complete after 1s

Destroy complete! Resources: 15 destroyed.
```

Instances (4) <a href="#">Info</a>				
<input type="text"/> Find Instance by attribute or tag (case-sensitive)			All states	▼
<input type="checkbox"/>	Name <a href="#">🔗</a>	Instance ID	Instance state	Instance type
<input type="checkbox"/>	prod-web-2	i-09df089f29185b455	<a href="#">Terminated</a> <a href="#">🔗</a> <a href="#">🔗</a>	t3.micro
<input type="checkbox"/>	prod-web-3	i-0e55f5d49a75862d3	<a href="#">Terminated</a> <a href="#">🔗</a> <a href="#">🔗</a>	t3.micro
<input type="checkbox"/>	prod-web-1	i-060422b95ce8e430e	<a href="#">Terminated</a> <a href="#">🔗</a> <a href="#">🔗</a>	t3.micro
<input type="checkbox"/>	prod-nginx-proxy	i-0974031199bc82f41	<a href="#">Terminated</a> <a href="#">🔗</a> <a href="#">🔗</a>	t3.micro

```
~\OneDrive\Desktop\CC Labs\Assignment2 (0.035s)
cat terraform.tfstate
```

\*\*\*\*\*