FATIMA JINNAH WOMEN UNIVERSITY

RAWALPINDI

Department of Software Engineering

Assignment no 2

SUBJECT: CLOUD COMPUTING

Name: Saira Ejaz

Reg No: 2023-BSE-057

Section: 5-B

Date: 30-Dec-2025

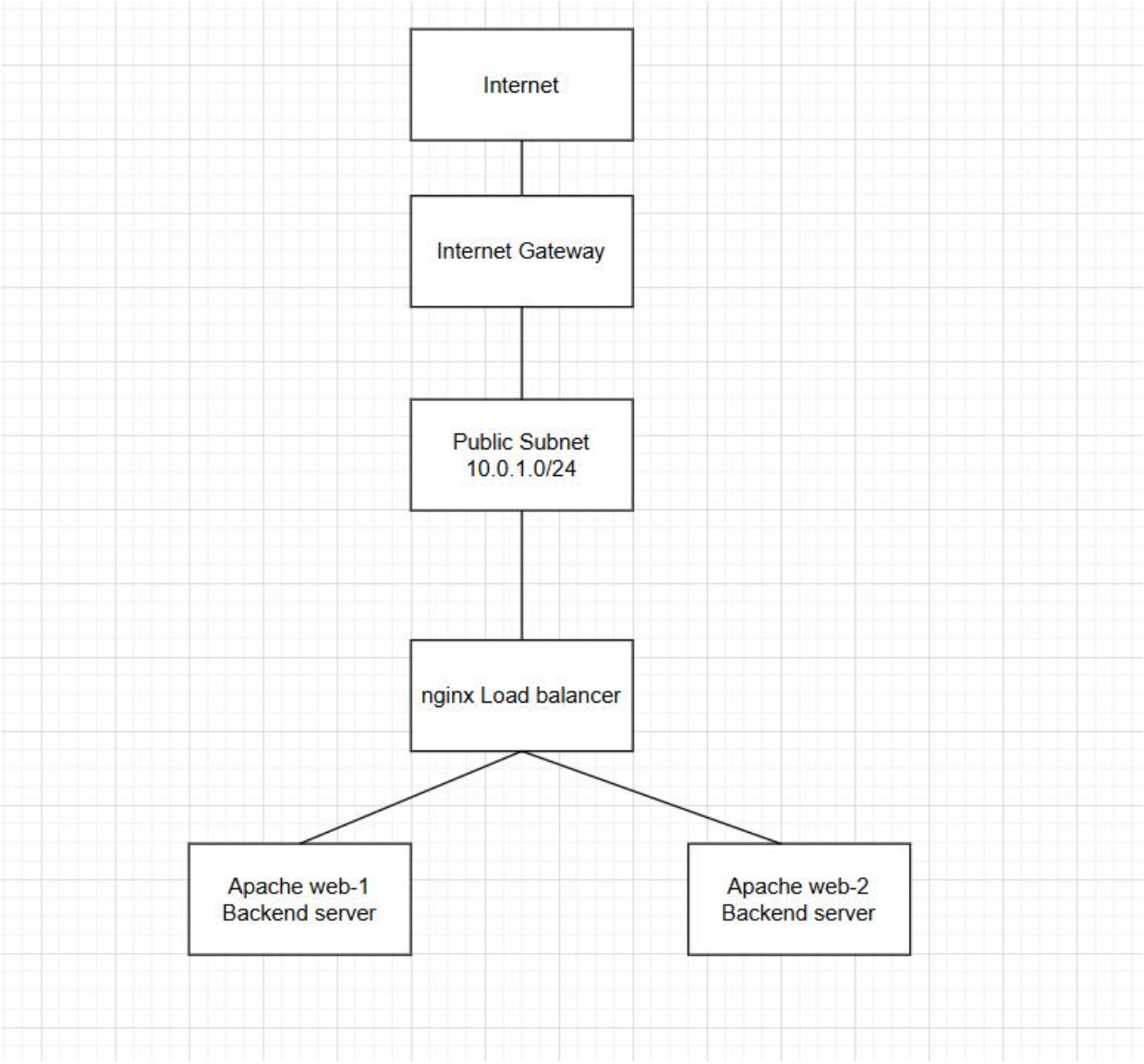
1. Executive Summary

1. Overview:

This assignment demonstrates the deployment of a multi-tier web infrastructure on AWS using Terraform. It involves creating an automated, scalable, and secure environment consisting of a VPC, public subnet, internet gateway, route tables, security groups, and EC2 instances for web and Nginx servers. The project highlights infrastructure-as-code principles by modularizing networking, security, and webserver components. Key tasks include configuring Nginx as a load balancer, deploying Apache web servers, testing load balancing, caching, high availability, and implementing proper security measures. The assignment showcases practical skills in cloud provisioning, automation, and monitoring while ensuring reliable and efficient web service delivery.

2. Architecture Design

Architecture Diagram:



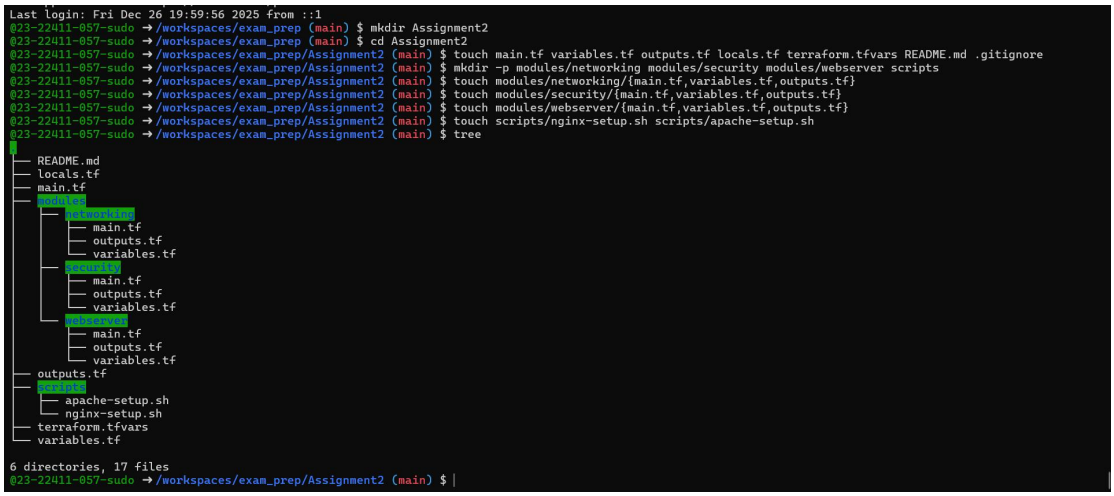
Part 1: Infrastructure Setup (25 marks)

1.1 Project Structure (5 marks)

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

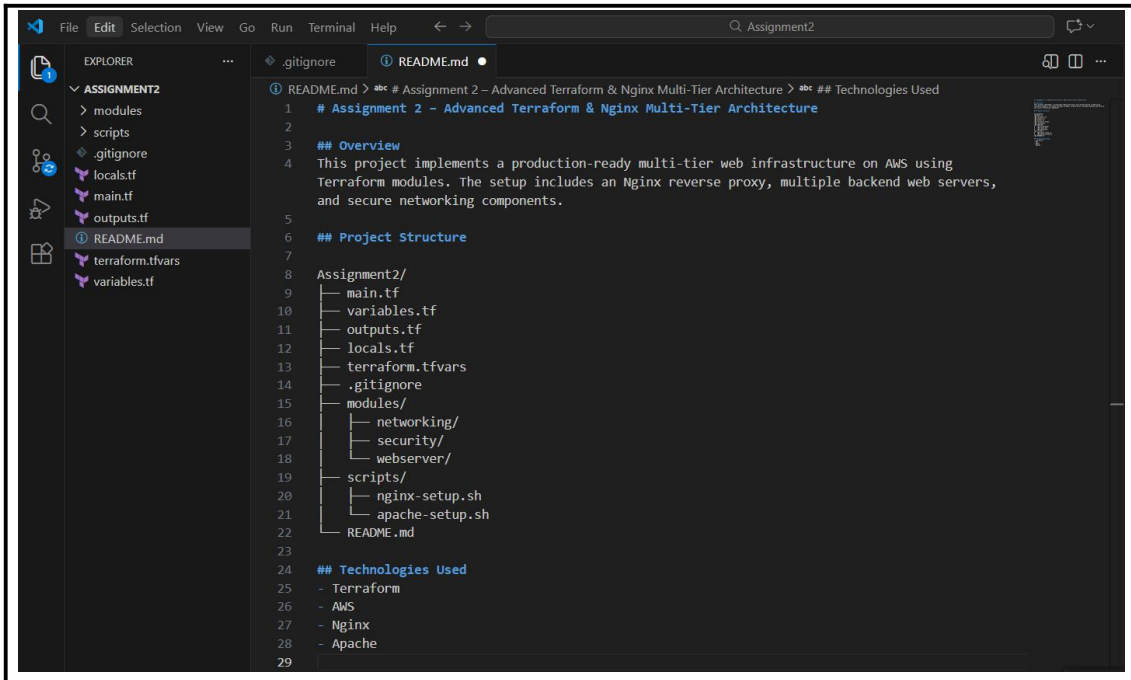
Tasks:

- Create all necessary files and directories



- Implement proper .gitignore to exclude sensitive files

- Document the project structure in README.md



1.2 Variable Configuration (5 marks)

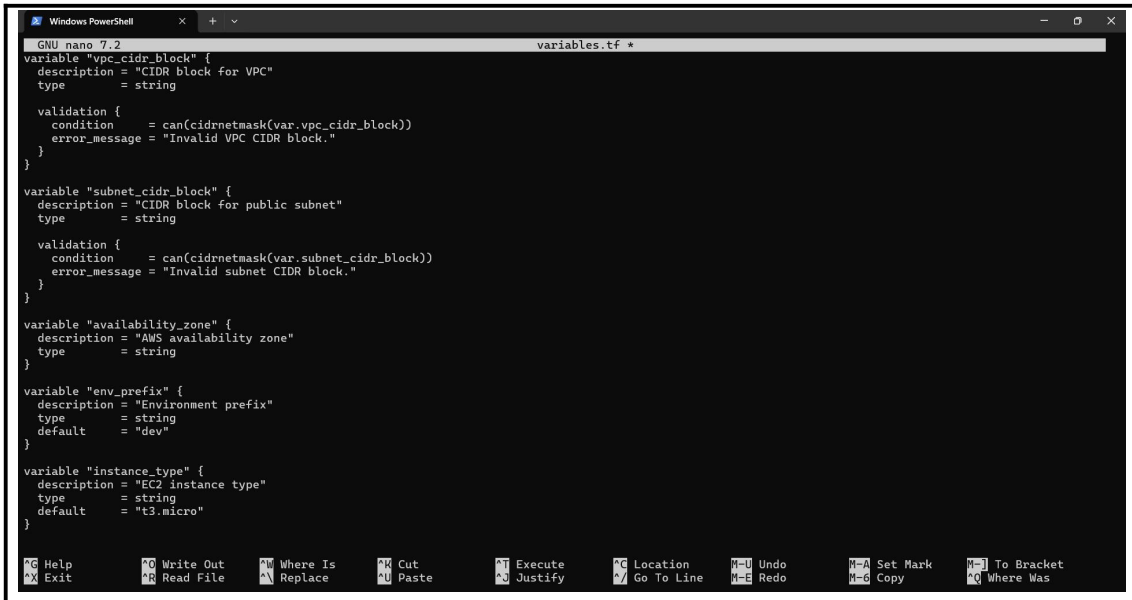
Define all required variables in variables.tf:

Required Variables:

- vpc_cidr_block (string)
- subnet_cidr_block (string)
- availability_zone (string)
- env_prefix (string)
- instance_type (string)
- public_key (string)
- private_key (string)
- backend_servers (list of objects with name and script_path)

Tasks:

- Add validation rules for CIDR blocks
- Add descriptions for all variables
- Set appropriate defaults where applicable
- Populate terraform.tfvars with your values




```
  GNU nano 7.2                                                variables.tf *

variable "availability_zone" {
  description = "AWS availability zone"
  type        = string
}

variable "env_prefix" {
  description = "Environment prefix"
  type        = string
  default     = "dev"
}

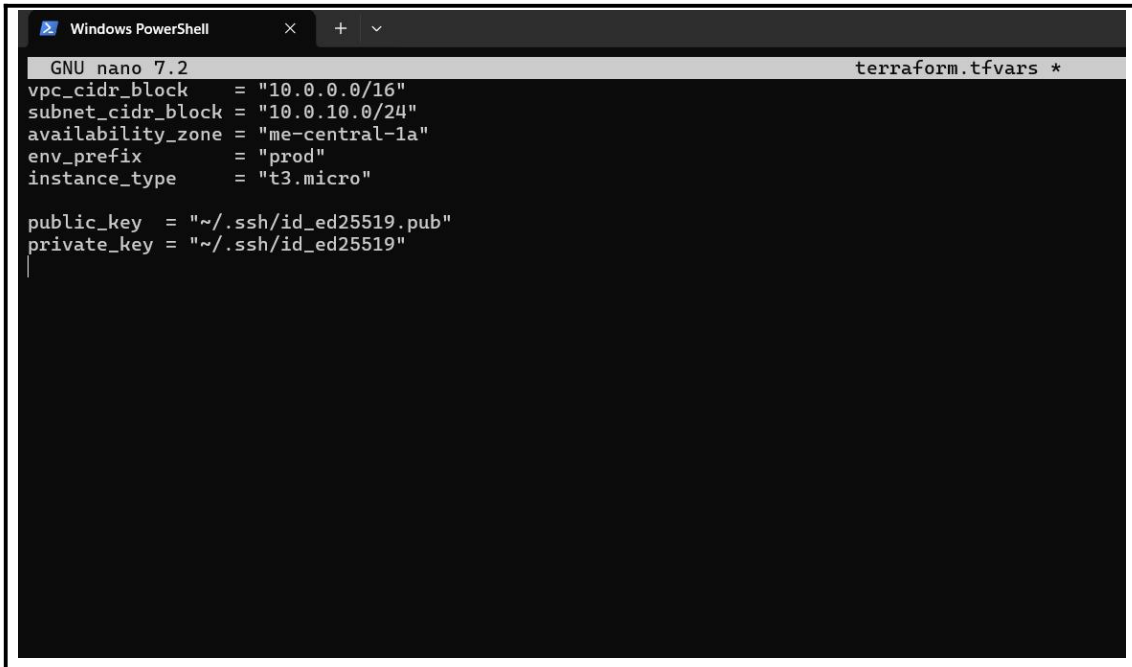
variable "instance_type" {
  description = "EC2 instance type"
  type        = string
  default     = "t3.micro"
}

variable "public_key" {
  description = "SSH public key path"
  type        = string
}

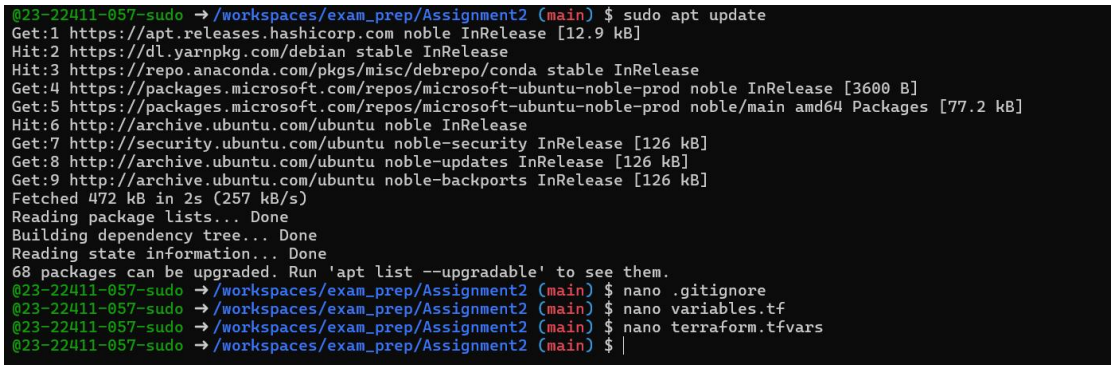
variable "private_key" {
  description = "SSH private key path"
  type        = string
}

variable "backend_servers" {
  description = "Backend server definitions"
  type = list(object({
    name        = string
    script_path = string
  }))
  default = []
}
```

- All required variables are declared with proper data types.
- Descriptions are added to improve readability.
- CIDR blocks are validated to ensure correct network configuration.
- Default values are set where appropriate.
- Sensitive values like private keys are marked as sensitive.



- terraform.tfvars contains environment-specific values.
- Backend servers are defined as a list of objects.
- This file is excluded from GitHub using .gitignore to protect sensitive data.



1.3 Networking Module (5 marks)

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

Module Location: `modules/networking/`

Required Outputs:

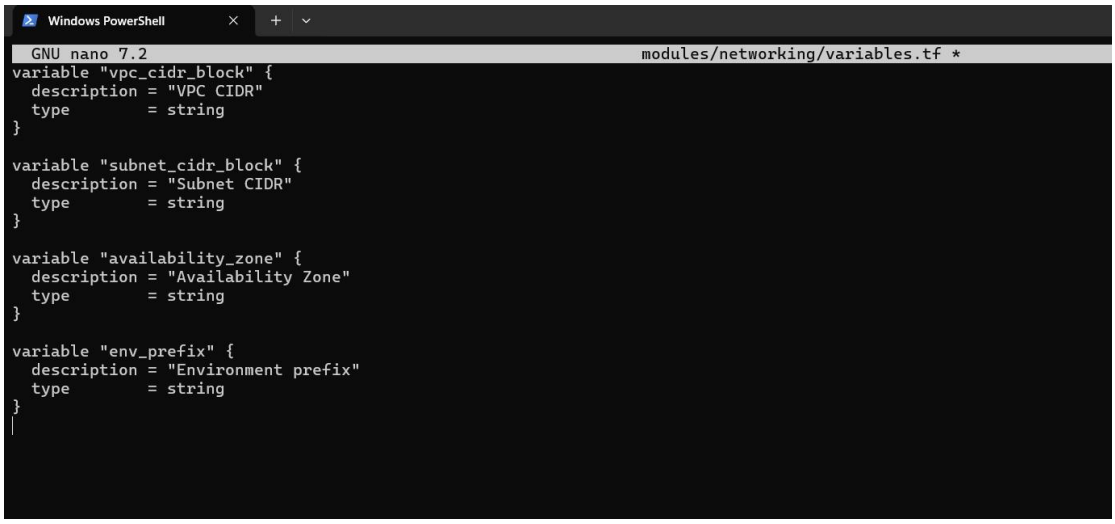
- vpc_id
- subnet_id
- igw_id

- route_table_id

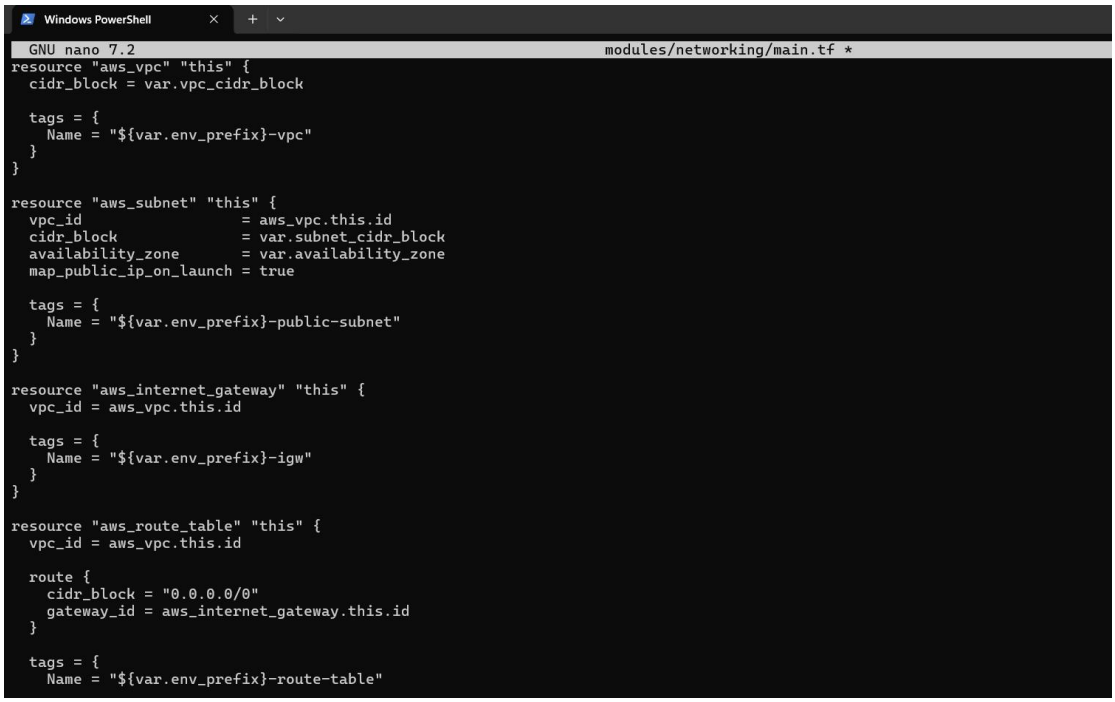
Tasks:

- 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`

Answer:



main.tf



```
resource "aws_route_table" "this" {
  vpc_id = aws_vpc.this.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.this.id
  }

  tags = {
    Name = "${var.env_prefix}-route-table"
  }
}

resource "aws_route_table_association" "this" {
  subnet_id      = aws_subnet.this.id
  route_table_id = aws_route_table.this.id
}
```

Output.tf

```
Windows PowerShell
  GNU nano 7.2                                          modules/networking/outputs.tf *
output "vpc_id" {
  value = aws_vpc.this.id
}

output "subnet_id" {
  value = aws_subnet.this.id
}

output "igw_id" {
  value = aws_internet_gateway.this.id
}

output "route_table_id" {
  value = aws_route_table.this.id
}
```

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ nano modules/networking/variables.tf
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ nano modules/networking/main.tf
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ nano modules/networking/outputs.tf
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $
```

1.4 Security Module (5 marks)

Create a security module that provisions:

Two Security Groups:

Nginx Security Group (for reverse proxy/load balancer):

Ingress: Port 22 (SSH) from your IP only

Ingress: Port 80 (HTTP) from anywhere (0.0.0.0/0)

Ingress: Port 443 (HTTPS) from anywhere (0.0.0.0/0)

Egress: All traffic

Backend Security Group (for web servers):

Ingress: Port 22 (SSH) from your IP only

Ingress: Port 80 (HTTP) from Nginx security group only

Egress: All traffic

Module Location: modules/security/

Required Variables:

vpc_id

env_prefix

my_ip

Required Outputs:

nginx_sg_id

backend_sg_id

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ nano modules/security/variables.tf
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $
```

```
  GNU nano 7.2                                          modules/security/variables.tf *
variable "vpc_id" {
  description = "VPC ID"
  type        = string
}

variable "env_prefix" {
  description = "Environment prefix"
  type        = string
}

variable "my_ip" {
  description = "My public IP with /32"
  type        = string
}
```

nano modules/security/main.tf

```
  GNU nano 7.2                                                  modules/security/main.tf *
# ================================
# NGINX SECURITY GROUP
# ================================
resource "aws_security_group" "nginx_sg" {
  name        = "${var.env_prefix}-nginx-sg"
  description = "Security group for Nginx reverse proxy"
  vpc_id      = var.vpc_id

  ingress {
    description = "SSH from my IP"
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = [var.my_ip]
  }

  ingress {
    description = "HTTP from anywhere"
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  ingress {
    description = "HTTPS from anywhere"
    from_port   = 443
    to_port     = 443
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
```

```
  GNU nano 7.2                                                  modules/security/main.tf *
  tags = {
    Name = "${var.env_prefix}-nginx-sg"
  }
}

# ================================
# BACKEND SECURITY GROUP
# ================================
resource "aws_security_group" "backend_sg" {
  name        = "${var.env_prefix}-backend-sg"
  description = "Security group for backend web servers"
  vpc_id      = var.vpc_id

  ingress {
    description = "SSH from my IP"
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = [var.my_ip]
  }

  ingress {
    description     = "HTTP from Nginx SG"
    from_port       = 80
    to_port         = 80
    protocol        = "tcp"
    security_groups = [aws_security_group.nginx_sg.id]
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
```

nano modules/security/outputs.tf

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ nano modules/security/outputs.tf
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $
```

```
  GNU nano 7.2                                          modules/security/outputs.tf *
output "nginx_sg_id" {
  value = aws_security_group.nginx_sg.id
}

output "backend_sg_id" {
  value = aws_security_group.backend_sg.id
}
```

Root main: nano main.tf

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ nano main.tf
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $
```

```
  GNU nano 7.2                                                  main.tf *
module "security" {
  source = "./modules/security"

  vpc_id     = module.networking.vpc_id
  env_prefix = var.env_prefix
  my_ip      = local.my_ip
}
```

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

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ nano locals.tf
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $
```

```
  GNU nano 7.2                                                  locals.tf *
# ================================
# FETCH MY PUBLIC IP
# ================================
data "http" "my_ip" {
  url = "https://icanhazip.com"
}

locals {
  # My IP with /32
  my_ip = "${chomp(data.http.my_ip.response_body)}/32"

  # Common tags
  common_tags = {
    Environment = var.env_prefix
    Project     = "Assignment-2"
    ManagedBy   = "Terraform"
  }

  # Resource naming
  name_prefix = "${var.env_prefix}-assignment2"

  # Backend servers configuration
  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 (15 marks)

2.1 Module Design (10 marks)

Resources to Create:

1. AWS Key Pair (unique per instance using suffix)
2. EC2 Instance with:
    1. Specified AMI (Amazon Linux 2023)
    2. Public IP enabled
    3. User data from script file
    4. Proper tags

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

1. nano modules/webserver/variables.tf

```
GNU nano 7.2                                                modules/webserver/variables.tf *
variable "env_prefix" {
  description = "Environment prefix"
  type        = string
}

variable "instance_name" {
  description = "Instance name"
  type        = string
}

variable "instance_type" {
  description = "EC2 instance type"
  type        = string
}

variable "availability_zone" {
  description = "Availability zone"
  type        = string
}

variable "vpc_id" {
  description = "VPC ID"
  type        = string
}

variable "subnet_id" {
  description = "Subnet ID"
  type        = string
}

variable "security_group_id" {
  description = "Security group ID"
  type        = string
}

variable "public_key" {
  description = "Public SSH key path"
  type        = string
}

variable "script_path" {
```

```
variable "public_key" {
  description = "Public SSH key path"
  type        = string
}

variable "script_path" {
  description = "User data script path"
  type        = string
}

variable "instance_suffix" {
  description = "Unique instance suffix"
  type        = string
}

variable "common_tags" {
  description = "Common resource tags"
  type        = map(string)
}
```

2. nano modules/webserver/main.tf

```
 GNU nano 7.2                                                  modules/webserver/main.tf *
# ================================
# FETCH AMAZON LINUX 2023 AMI
# ================================
data "aws_ami" "amazon_linux" {
  most_recent = true
  owners      = ["amazon"]

  filter {
    name   = "name"
    values = ["al2023-ami-*-x86_64"]
  }
}

# ================================
# KEY PAIR (UNIQUE PER INSTANCE)
# ================================
resource "aws_key_pair" "this" {
  key_name   = "${var.env_prefix}-${var.instance_suffix}-key"
  public_key = file(var.public_key)
}

# ================================
# EC2 INSTANCE
# ================================
resource "aws_instance" "this" {
  ami                    = data.aws_ami.amazon_linux.id
  instance_type          = var.instance_type
  availability_zone      = var.availability_zone
  subnet_id              = var.subnet_id
  vpc_security_group_ids = [var.security_group_id]
  key_name               = aws_key_pair.this.key_name
  associate_public_ip_address = true

  user_data = file(var.script_path)

  tags = merge(
    var.common_tags,
    {
      Name = "${var.env_prefix}-${var.instance_name}"
    }
  )
```

3. nano modules/webserver/outputs.tf

```
 GNU nano 7.2                                                  modules/webserver/outputs.tf *
output "instance_id" {
  value = aws_instance.this.id
}

output "public_ip" {
  value = aws_instance.this.public_ip
}

output "private_ip" {
  value = aws_instance.this.private_ip
}
```

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ nano modules/webserver/main.tf
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ nano modules/webserver/outputs.tf
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ cat modules/webserver/outputs.tf
output "instance_id" {
  value = aws_instance.this.id
}

output "public_ip" {
  value = aws_instance.this.public_ip
}

output "private_ip" {
  value = aws_instance.this.private_ip
}
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $
```

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)

Tasks:

- . Create Nginx server module instance
- . Create backend server module instances using for_each

- . Ensure proper security group assignment
- . Pass all required variables

1. nano main.tf

Add nginx server module and backend server

```
@23-22411-057-sudo → /workspaces/exam_prep/Assignment2 (main) $ cat << 'EOF' >> main.tf
> # ================================
# NGINX SERVER
# ================================
module "nginx_server" {
  source            = "./modules/webserver"
  env_prefix        = var.env_prefix
  instance_name     = "nginx-proxy"
  instance_type     = var.instance_type
  availability_zone = var.availability_zone
  vpc_id            = module.networking.vpc_id
  subnet_id         = module.networking.subnet_id
  security_group_id = module.security.nginx_sg_id
  public_key        = var.public_key
  script_path       = "./scripts/nginx-setup.sh"
  instance_suffix   = "nginx"
  common_tags       = local.common_tags
}

# ================================
# BACKEND SERVERS
# ================================
module "backend_servers" {
  for_each = {
    for server in local.backend_servers : server.name => server
  }

  source            = "./modules/webserver"
  env_prefix        = var.env_prefix
  instance_name     = each.value.name
  instance_type     = var.instance_type
  availability_zone = var.availability_zone
  vpc_id            = module.networking.vpc_id
  subnet_id         = module.networking.subnet_id
  security_group_id = module.security.backend_sg_id
  public_key        = var.public_key
  script_path       = each.value.script_path
  instance_suffix   = each.value.suffix
  common_tags       = local.common_tags
}
EOF
```

```
@23-22411-057-sudo → /workspaces/exam_prep/Assignment2 (main) $ nano main.tf
@23-22411-057-sudo → /workspaces/exam_prep/Assignment2 (main) $ terraform fmt
```

```
@23-22411-057-sudo → /workspaces/exam_prep/Assignment2 (main) $ terraform validate
Success! The configuration is valid.

@23-22411-057-sudo → /workspaces/exam_prep/Assignment2 (main) $
```

Part 3: Server Configuration Scripts (20 marks)

3.1 Apache Backend Server Script (10 marks)

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

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ terraform output
backend_private_ips = [
  "10.0.1.145",
  "10.0.1.195",
  "10.0.1.46",
]
backend_public_ips = [
  "3.28.46.119",
  "158.252.78.6",
  "3.29.21.186",
]
nginx_private_ip = "10.0.1.79"
nginx_public_ip = "3.29.123.240"
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $
```

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ ssh -i Key.pem ec2-user@40.172.159.252
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Wed Dec 31 09:27:23 2025 from 4.240.39.197
[ec2-user@ip-172-31-10-173 ~]$ ls scripts
ls: cannot access 'scripts': No such file or directory
[ec2-user@ip-172-31-10-173 ~]$ cd scripts
-bash: cd: scripts: No such file or directory
[ec2-user@ip-172-31-10-173 ~]$ ls
[ec2-user@ip-172-31-10-173 ~]$ Part 3: Server Configuration Scripts (20 marks)
3.1 Apache Backend Server Scripls
                          Part 3: Server Configuration Scripts (20 marks)
3.1 Apache Backend Server Scripls
                          ^C
[ec2-user@ip-172-31-10-173 ~]$ cd ~/scripts
-bash: cd: /home/ec2-user/scripts: No such file or directory
[ec2-user@ip-172-31-10-173 ~]$ mkdir -p ~/scripts
[ec2-user@ip-172-31-10-173 ~]$ cd ~/scripts
[ec2-user@ip-172-31-10-173 scripts]$ nano scripts/apache-setup.sh
[ec2-user@ip-172-31-10-173 scripts]$ nano apache-setup.sh
[ec2-user@ip-172-31-10-173 scripts]$ chmod +x apache-setup.sh
[ec2-user@ip-172-31-10-173 scripts]$ sudo ./apache-setup.sh
Last metadata expiration check: 6:03:07 ago on Wed Dec 31 09:30:31 2025.
Dependencies resolved.
Nothing to do.
Complete!
Last metadata expiration check: 6:03:07 ago on Wed Dec 31 09:30:31 2025.
Package httpd-2.4.65-1.amzn2023.0.2.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
Apache setup completed successfully!
[ec2-user@ip-172-31-10-173 scripts]$
```

Deliverables:

Screenshot: assignment_part3_apache_script.png

```
 GNU nano 8.3                                  apache-setup.sh                                  Modified
#!/bin/bash

# Update system
sudo yum update -y

# Install Apache
sudo yum install httpd -y

# Start Apache
sudo systemctl start httpd
sudo systemctl enable httpd

# Get EC2 metadata token (IMDSv2)
TOKEN=$(curl -s -X PUT "http://169.254.169.254/latest/api/token" \
  -H "X-aws-ec2-metadata-token-ttl-seconds: 21600")

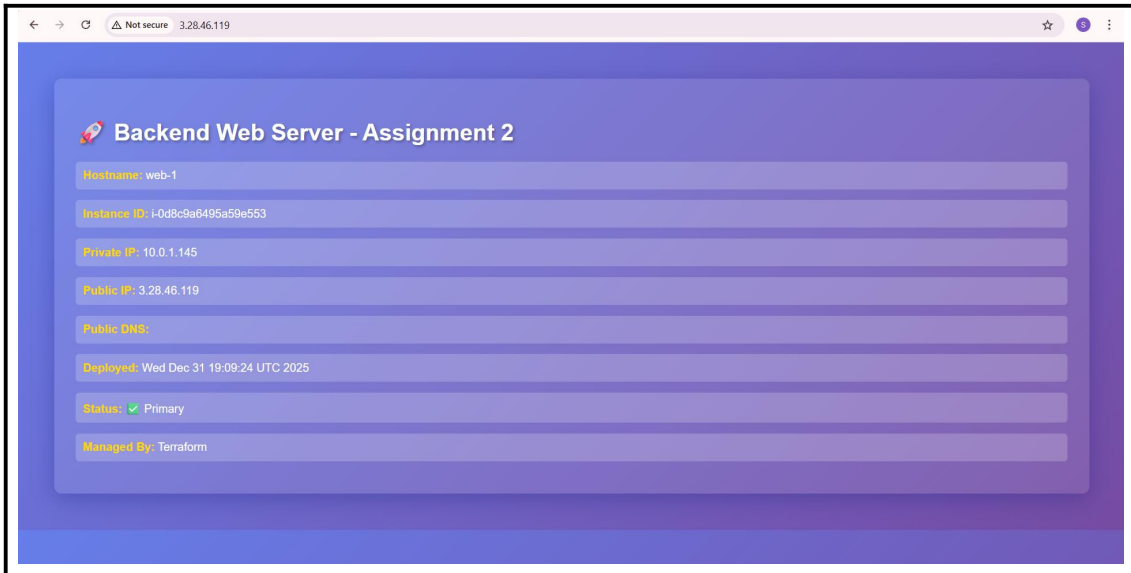
# Get metadata values
PRIVATE_IP=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \
  http://169.254.169.254/latest/meta-data/local-ipv4)

PUBLIC_IP=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \
  http://169.254.169.254/latest/meta-data/public-ipv4)

INSTANCE_ID=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \
  http://169.254.169.254/latest/meta-data/instance-id)

# Create web page
sudo cat > /var/www/html/index.html <<EOF
<html>
<head>
<title>Backend Server</title>
</head>
<body>
<h1>Backend Web Server - Assignment 2</h1>
<p><b>Hostname:</b> $(hostname)</p>
<p><b>Instance ID:</b> $INSTANCE_ID</p>
<p><b>Private IP:</b> $PRIVATE_IP</p>
```

Screenshot: assignment_part3_backend_webpage.png (browser showing backend server page)



3.2 Nginx Server Setup Script (10 marks)

Create scripts/nginx-setup.sh that:

Deliverables:

Screenshot: assignment_part3_nginx_script.png

```
  GNU nano 8.3                                                  nginx-setup.sh
#!/bin/bash
set -e

# Update and install Nginx
yum update -y
yum install -y nginx openssl
systemctl start nginx
systemctl enable nginx

# Create SSL directories
mkdir -p /etc/ssl/private
mkdir -p /etc/ssl/certs

# Get metadata token
TOKEN=$(curl -s -X PUT "http://169.254.169.254/latest/api/token" \
  -H "X-aws-ec2-metadata-token-ttl-seconds: 21600")

# Get public IP
PUBLIC_IP=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \
  http://169.254.169.254/latest/meta-data/public-ipv4)

# Generate self-signed certificate
openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
  -keyout /etc/ssl/private/selfsigned.key \
  -out /etc/ssl/certs/selfsigned.crt \
  -subj "/CN=$PUBLIC_IP" \
  -addext "subjectAltName=IP:$PUBLIC_IP" \
  -addext "basicConstraints=CA:FALSE" \
  -addext "keyUsage=digitalSignature,keyEncipherment" \
  -addext "extendedKeyUsage=serverAuth"

# Backup original config
cp /etc/nginx/nginx.conf /etc/nginx/nginx.conf.bak

# Create Nginx configuration
cat > /etc/nginx/nginx.conf <<'EOF'
user nginx;
```

```
 GNU nano 8.3                                                    nginx-setup.sh
# Create Nginx configuration
cat > /etc/nginx/nginx.conf <<'EOF'
user nginx;
worker_processes auto;
error_log /var/log/nginx/error.log notice;
pid /run/nginx.pid;

events {
    worker_connections 1024;
}

http {
    log_format main '$remote_addr - $remote_user [$time_local] "$request" '
                    '$status $body_bytes_sent "$http_referer" '
                    '"$http_user_agent" "$http_x_forwarded_for" '
                    'Cache: $upstream_cache_status';

    access_log /var/log/nginx/access.log main;

    sendfile on;
    tcp_nopush on;
    keepalive_timeout 65;
    include /etc/nginx/mime.types;
    default_type application/octet-stream;

    gzip on;
    gzip_vary on;
    gzip_types text/plain text/css application/json application/javascript text/xml application/xml;

    proxy_cache_path /var/cache/nginx levels=1:2 keys_zone=my_cache:10m max_size=1g inactive=60m use_temp_path=off;

    upstream backend_servers {
        server BACKEND_IP_1:80;
        server BACKEND_IP_2:80;
        server BACKEND_IP_3:80 backup;
    }
```

```
[ec2-user@myapp-webserver scripts]$ nano nginx-setup.sh
[ec2-user@myapp-webserver scripts]$ chmod +x nginx-setup.sh
[ec2-user@myapp-webserver scripts]$ sudo ./nginx-setup.sh
Last metadata expiration check: 6:11:51 ago on Wed Dec 31 09:30:31 2025.
Dependencies resolved.
Nothing to do.
Complete!
Last metadata expiration check: 6:11:52 ago on Wed Dec 31 09:30:31 2025.
Package openssl-1:3.2.2-1.amzn2023.0.2.x86_64 is already installed.
Dependencies resolved.
================================================================================================================================================
 Package                         Architecture            Version                                 Repository                              Size
================================================================================================================================================
Installing:
 nginx                           x86_64                  1:1.28.0-1.amzn2023.0.2                 amazonlinux                             33 k
Installing dependencies:
 gperftools-libs                 x86_64                  2.9.1-1.amzn2023.0.3                    amazonlinux                            308 k
 libunwind                       x86_64                  1.4.0-5.amzn2023.0.3                    amazonlinux                             66 k
 nginx-core                      x86_64                  1:1.28.0-1.amzn2023.0.2                 amazonlinux                            686 k
 nginx-filesystem                noarch                  1:1.28.0-1.amzn2023.0.2                 amazonlinux                            9.6 k
 nginx-mimetypes                 noarch                  2.1.49-3.amzn2023.0.3                   amazonlinux                             21 k

Transaction Summary
================================================================================================================================================
Install  6 Packages

Total download size: 1.1 M
Installed size: 3.6 M
Downloading Packages:
(1/6): libunwind-1.4.0-5.amzn2023.0.3.x86_64.rpm                                 1.9 MB/s |  66 kB     00:00
(2/6): nginx-1.28.0-1.amzn2023.0.2.x86_64.rpm                                    880 kB/s |  33 kB     00:00
(3/6): gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64.rpm                           6.7 MB/s | 308 kB     00:00
(4/6): nginx-filesystem-1.28.0-1.amzn2023.0.2.noarch.rpm                         417 kB/s | 9.6 kB     00:00
(5/6): nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch.rpm                          1.1 MB/s |  21 kB     00:00
(6/6): nginx-core-1.28.0-1.amzn2023.0.2.x86_64.rpm                                17 MB/s | 686 kB     00:00
------------------------------------------------------------------------------------------------------------------------------------------------
Total                                                                            9.4 MB/s | 1.1 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
```

Screenshot: assignment_part3_nginx_default_page.png (before backend configuration)

Part 4: Infrastructure Deployment (15 marks)

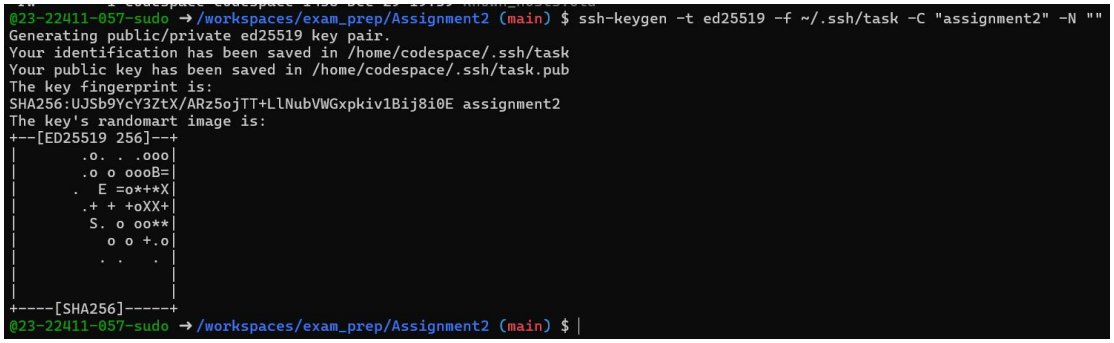
4.1 Initial Deployment (5 marks)

Deploy the infrastructure using Terraform.

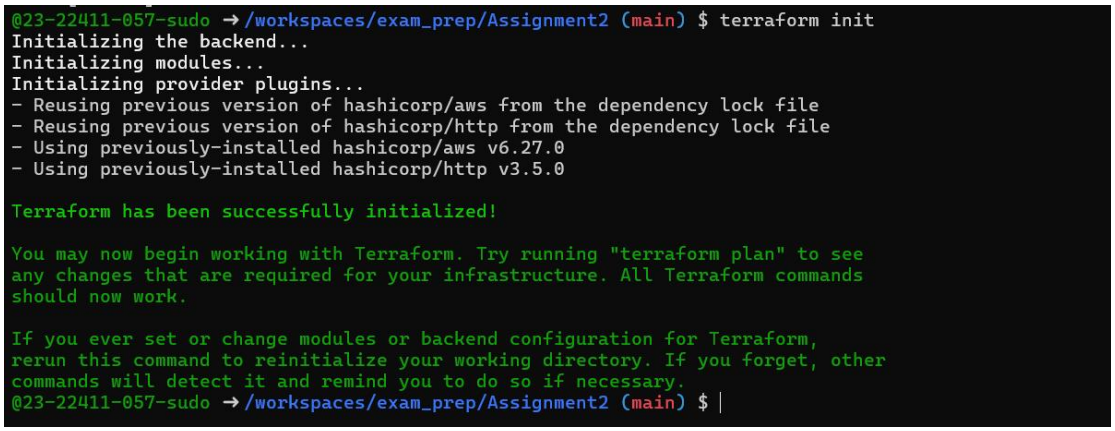
Tasks:

- 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)

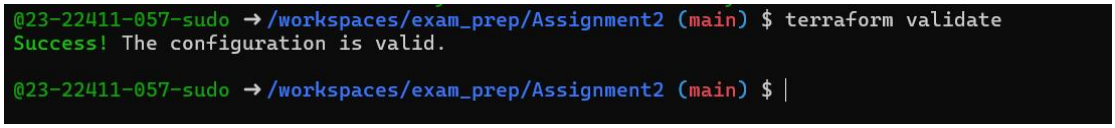
1. assignment_part4_ssh_keygen.png



2. assignment_part4_terraform_init.png



3. assignment_part4_terraform_validate.png



4. assignment_part4_terraform_plan.png

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ terraform plan
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 0s [id=https://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.networking.aws_vpc.this: Refreshing state... [id=vpc-0e4e21a35062d094b]
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-2"].data.aws_ami.amazon_linux: Reading...
module.backend_servers["web-1"].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-1"].aws_key_pair.this: Refreshing state... [id=prod-1-key]
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.backend_servers["web-3"].data.aws_ami.amazon_linux: Read complete after 1s [id=ami-009ce8169fc88edf5]
module.networking.aws_subnet.this: Refreshing state... [id=subnet-06aa2b53e49dc297c]
module.networking.aws_internet_gateway.this: Refreshing state... [id=igw-03ccfd444d3aa3e54]
module.security.aws_security_group.nginx_sg: Refreshing state... [id=sg-03675053d501f20c6]
module.networking.aws_route_table.this: Refreshing state... [id=rtb-0c7fc944a600be5a4]
module.security.aws_security_group.backend_sg: Refreshing state... [id=sg-0071d61fcdc37bfca]
module.nginx_server.aws_instance.this: Refreshing state... [id=i-05e94c22aed050428]
module.networking.aws_route_table_association.this: Refreshing state... [id=rtbassoc-0656bf261a7225875]
module.backend_servers["web-1"].aws_instance.this: Refreshing state... [id=i-02b40c2a7833d1b8d]
module.backend_servers["web-3"].aws_instance.this: Refreshing state... [id=i-079858b9234bd9303]
module.backend_servers["web-2"].aws_instance.this: Refreshing state... [id=i-0e91759f81ce177bd]

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

Terraform will perform the following actions:

  # module.backend_servers["web-1"].aws_instance.this will be updated in-place
  ~ resource "aws_instance" "this" {
        id                                   = "i-02b40c2a7833d1b8d"
      + public_dns                           = (known after apply)
      ~ public_ip                            = "3.29.64.144" -> (known after apply)
        tags                                 = {
            "Environment" = "prod"
            "ManagedBy"   = "Terraform"
```

5. assignment_part4_terraform_apply.png (showing all 4 instances created)

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ terraform apply
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 0s [id=https://icanhazip.com]
aws_key_pair.main: Refreshing state... [id=assignment2-key]
module.networking.aws_vpc.this: Refreshing state... [id=vpc-0be362600fc45d0e4]
module.networking.aws_internet_gateway.this: Refreshing state... [id=igw-0ce094a100a0e8957]
module.networking.aws_subnet.this: Refreshing state... [id=subnet-0b2f5f837d42a3e8d]
module.networking.aws_route_table.this: Refreshing state... [id=rtb-0818d3278b2cf1514]
module.backend_servers.aws_instance.this[1]: Refreshing state... [id=i-0c2d56210d374f8e4]
module.backend_servers.aws_instance.this[2]: Refreshing state... [id=i-03c6a7accd146e235]
module.backend_servers.aws_instance.this[0]: Refreshing state... [id=i-0d8c9a6495a59e553]
module.nginx_server.aws_instance.this[0]: Refreshing state... [id=i-0152a4b5da743b17f]
module.networking.aws_route_table_association.this: Refreshing state... [id=rtbassoc-0bce8714ac48f8f49]

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.

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

Outputs:

backend_private_ips = [
  "10.0.1.145",
  "10.0.1.195",
  "10.0.1.46",
]
backend_public_ips = [
  "3.28.46.119",
  "158.252.78.6",
  "3.29.21.186",
]
nginx_private_ip = "10.0.1.79"
nginx_public_ip = "3.29.123.240"
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $
```

4.2 Output Configuration (5 marks)

1. Screenshot: assignment_part4_terraform_output.png

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ terraform output
backend_private_ips = [
  "10.0.1.145",
  "10.0.1.195",
  "10.0.1.46",
]
backend_public_ips = [
  "3.28.46.119",
  "158.252.78.6",
  "3.29.21.186",
]
nginx_private_ip = "10.0.1.79"
nginx_public_ip = "3.29.123.240"
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $
```

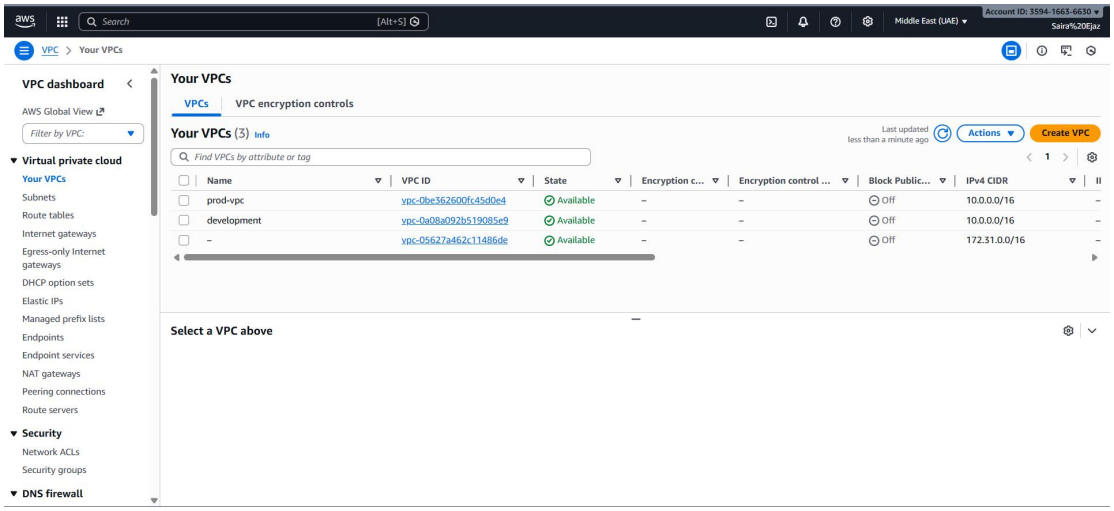
2. Screenshot: assignment_part4_outputs_json.png

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ terraform output -json > outputs.json
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $
```

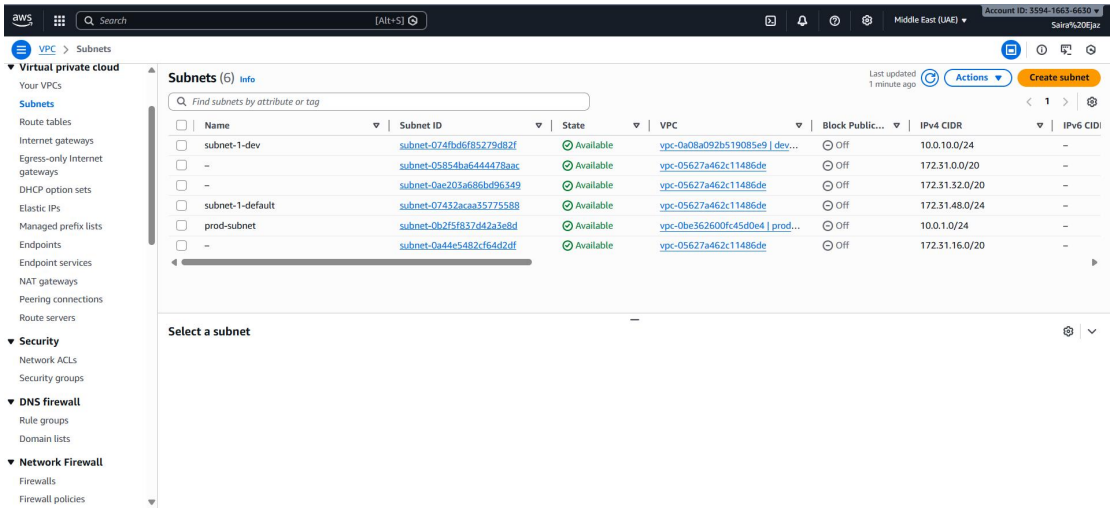
4.3 AWS Console Verification (5 marks)

Verify all resources in AWS Console.

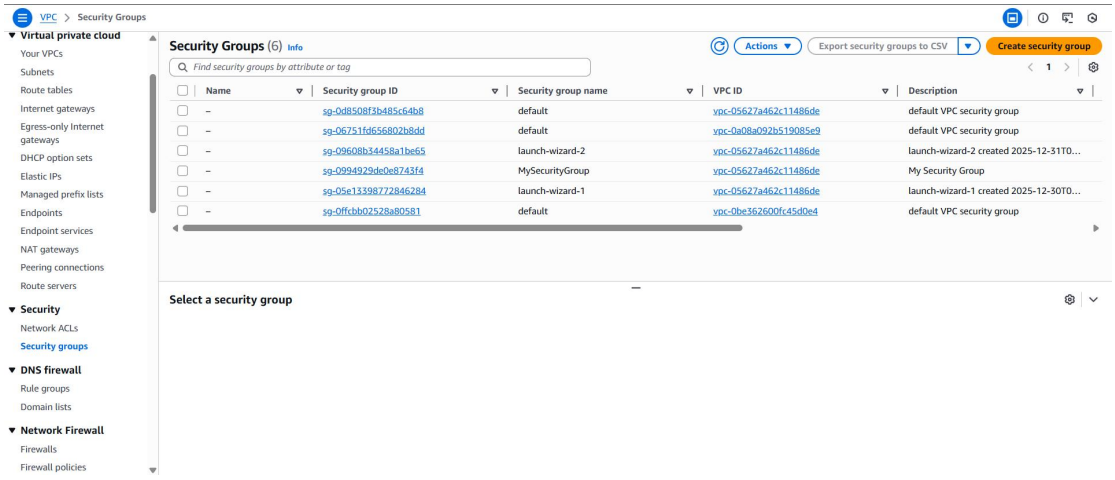
1. assignment_part4_aws_vpc. Png



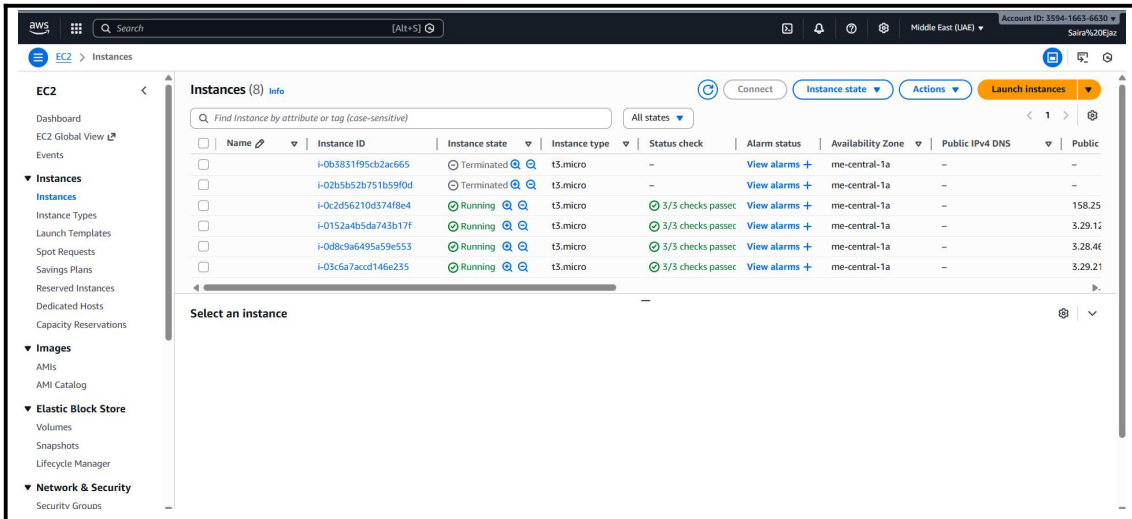
2. assignment_part4_aws_subnet.png



3. assignment_part4_aws_security_groups.png



4. assignment_part4_aws_instances.png



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.

Deliverables:

Screenshot: assignment_part5_ssh_nginx.png (SSH session)

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ ssh -i ~/.ssh/assignment2_key ec2-user@3.29.123.240
Last login: Wed Dec 31 15:59:02 2025 from 4.240.39.197
       #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-10-0-1-79 ~]$
```

Screenshot: assignment_part5_nginx_conf_updated.png (updated upstream block)

```
user nginx;
worker_processes auto;

events {
    worker_connections 1024;
}

http {
    upstream backend_servers {
        server 10.0.1.145:80;
        server 10.0.1.195:80;
        server 10.0.1.46:80 backup;
    }

    server {
        listen 80;

        location / {
            proxy_pass http://backend_servers;
        }

        location /health {
            return 200 "Nginx is healthy\n";
            add_header Content-Type text/plain;
        }
    }
}
```

Screenshot: assignment_part5_nginx_test.png (nginx -t output)

```
[ec2-user@ip-10-0-1-79 ~]$ 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-10-0-1-79 ~]$
```

Screenshot: assignment_part5_nginx_restart.png (restart and status)

```
[ec2-user@ip-10-0-1-79 ~]$ sudo systemctl restart nginx
[ec2-user@ip-10-0-1-79 ~]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2025-12-31 16:37:34 UTC; 9s ago
  Process: 2952 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
  Process: 2949 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
  Process: 2947 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
 Main PID: 2954 (nginx)
   CGroup: /system.slice/nginx.service
           ├─2954 nginx: master process /usr/sbin/nginx
           ├─2955 nginx: worker process
           └─2956 nginx: worker process

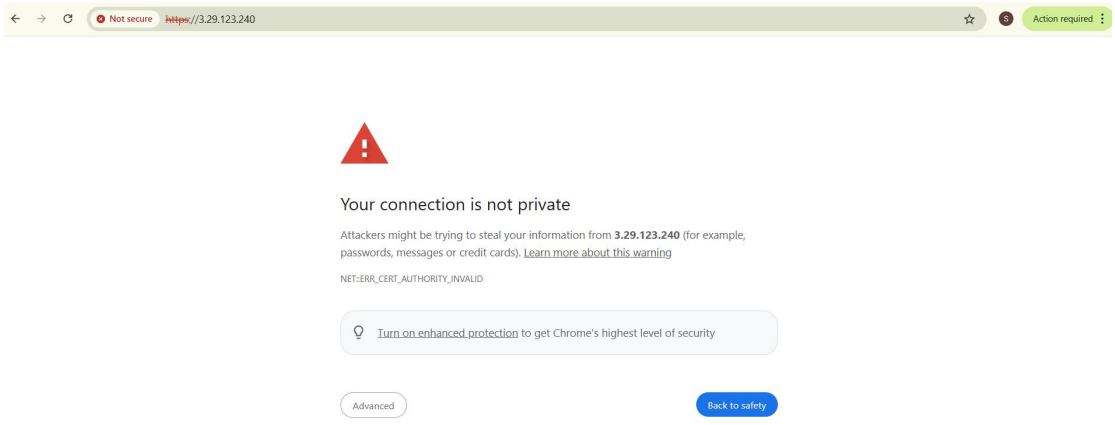
Dec 31 16:37:34 ip-10-0-1-79.me-central-1.compute.internal systemd[1]: Starting The nginx HTTP and reverse proxy server...
Dec 31 16:37:34 ip-10-0-1-79.me-central-1.compute.internal nginx[2949]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Dec 31 16:37:34 ip-10-0-1-79.me-central-1.compute.internal nginx[2949]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Dec 31 16:37:34 ip-10-0-1-79.me-central-1.compute.internal systemd[1]: Started The nginx HTTP and reverse proxy server.
[ec2-user@ip-10-0-1-79 ~]$
```

5.2 Test Load Balancing (5 marks)

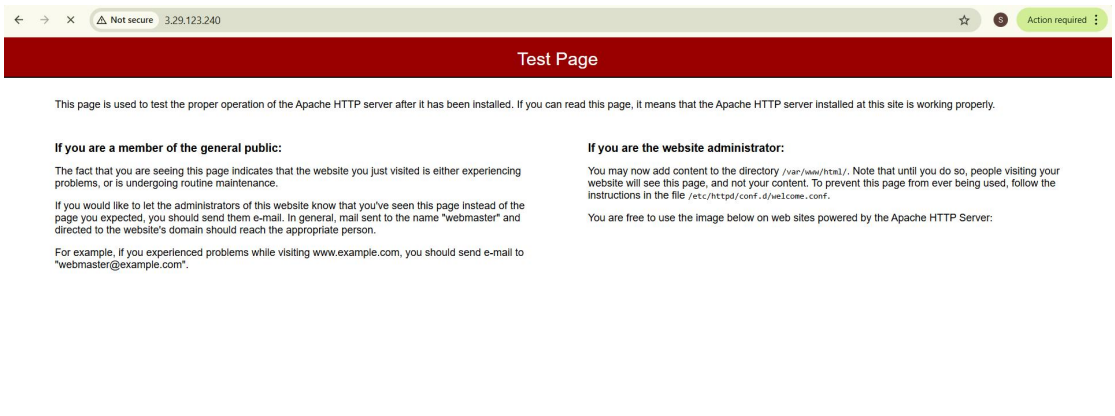
Test that Nginx is properly load balancing between web-1 and web-2.

Deliverables:

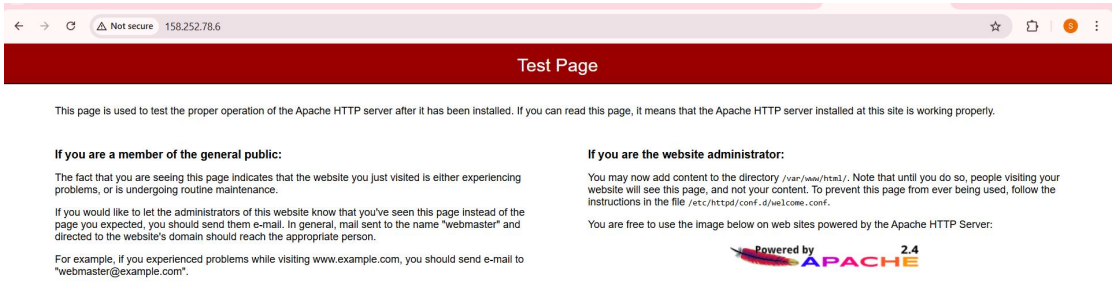
Screenshot: assignment_part5_ssl_warning.png (browser security warning)



Screenshot: assignment_part5_web1_response.png (showing web-1 content)



Screenshot: assignment_part5_web2_response. png (showing web-2 content)



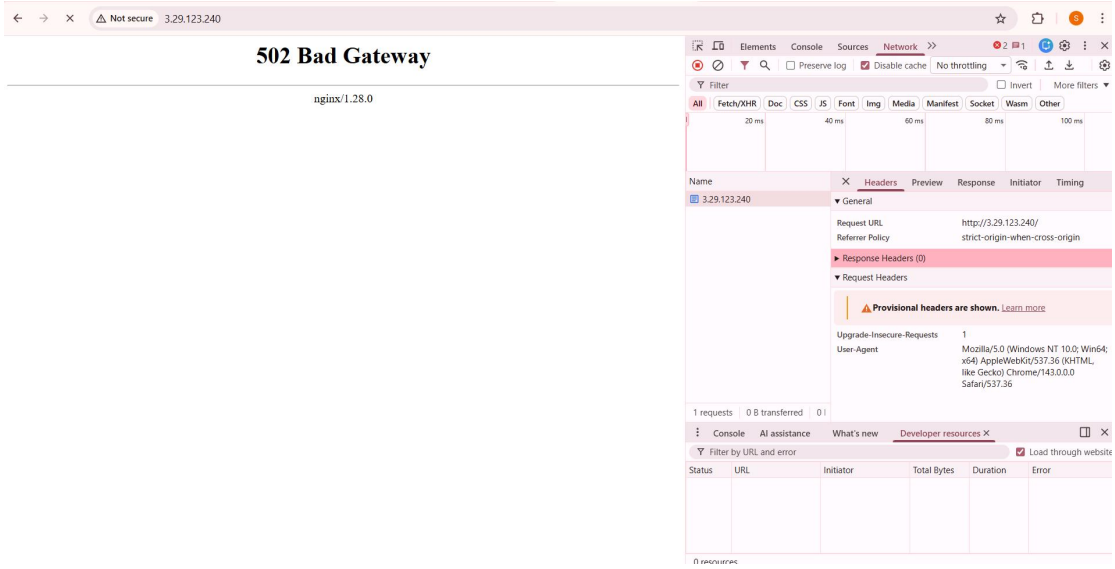
Screenshot: assignment_part5_load_balancing_demo.png (multiple reloads showing alternation)

5.3 Test Cache Functionality (5 marks)

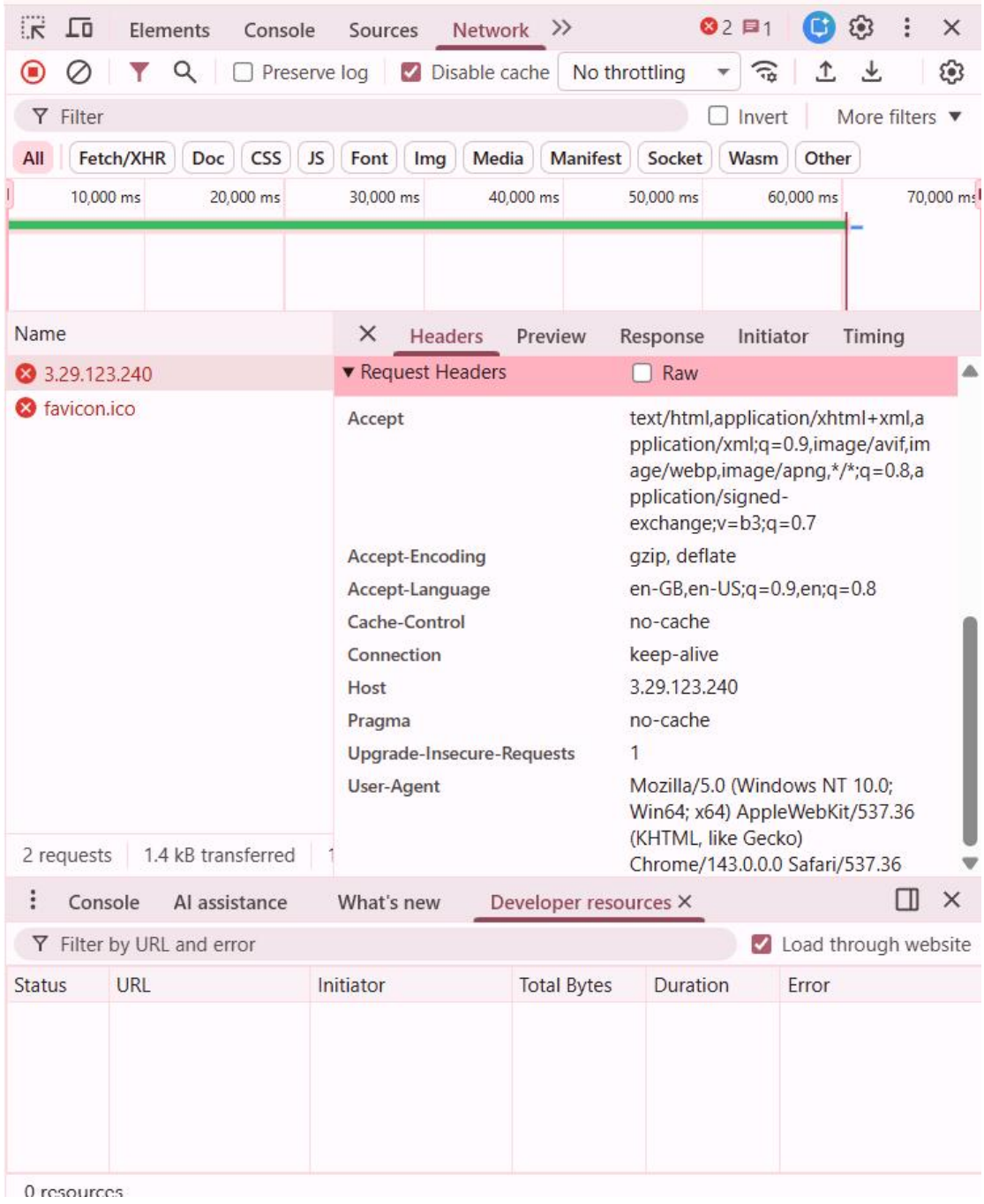
Verify that Nginx caching is working correctly.

Deliverables:

Screenshot: assignment_part5_cache_miss.png (first request - MISS)



Screenshot: assignment_part5_cache_hit. png (second request - HIT)



Screenshot: assignment_part5_cache_directory.png (cache folder contents)

```
[ec2-user@ip-10-0-1-79 ~]$ ls -la /var/cache/nginx/
total 0
drwxr-xr-x 2 nginx nginx  6 Dec 31 16:48 .
drwxr-xr-x 7 root  root  76 Dec 31 16:48 ..
[ec2-user@ip-10-0-1-79 ~]$
```

Screenshot: assignment_part5_access_log_cache.png (access log showing cache status)

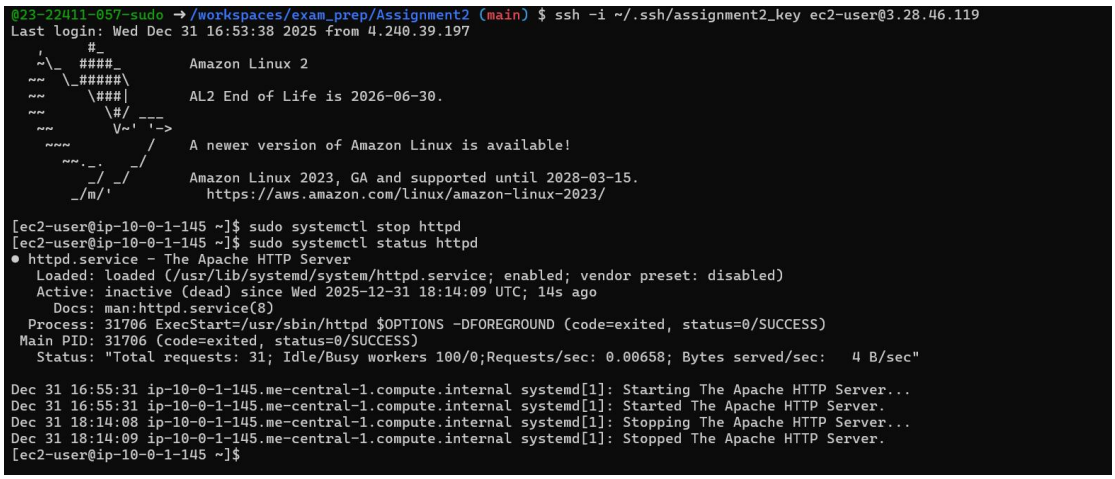
```
[ec2-user@ip-10-0-1-79 ~]$ sudo grep "Cache:" /var/log/nginx/access.log | tail -20
[ec2-user@ip-10-0-1-79 ~]$
```

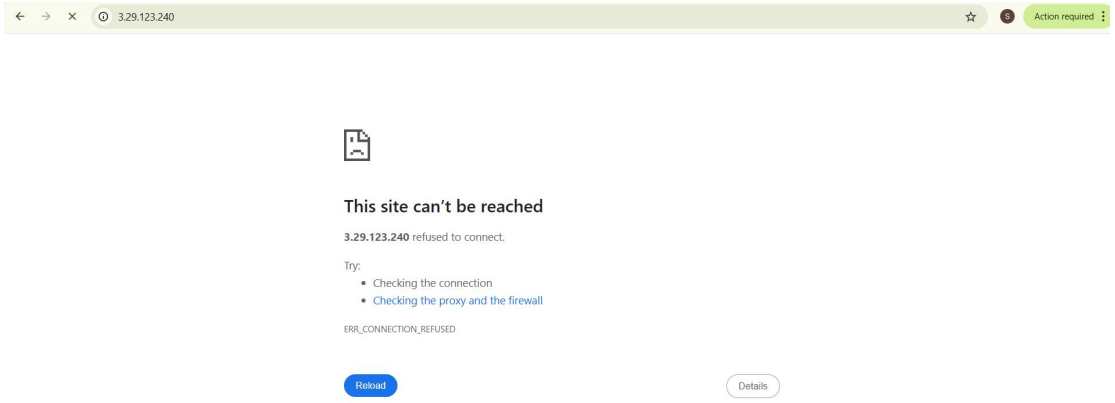
5.4 Test High Availability (Backup Server) (5 marks)

Test the backup server functionality by simulating primary server failure.

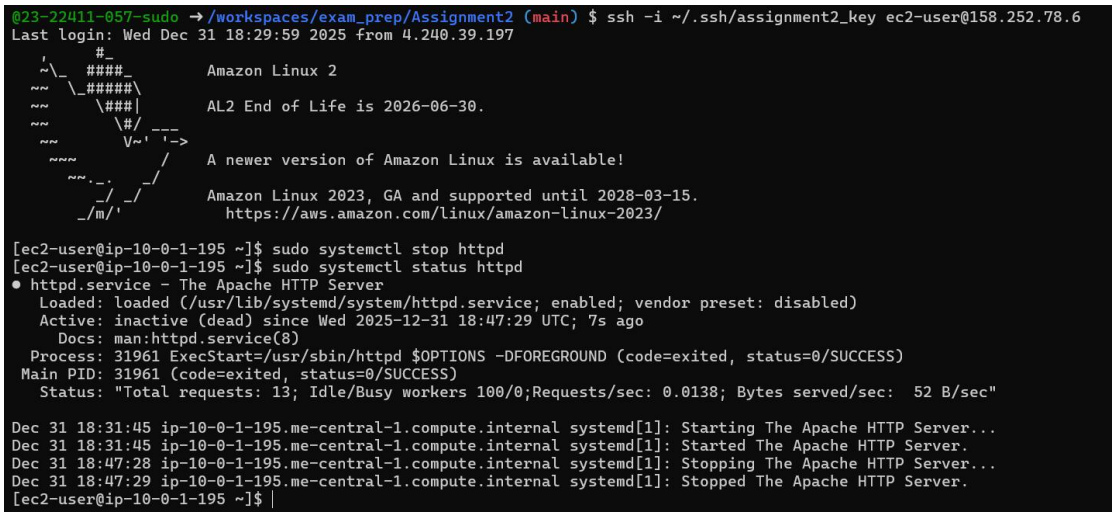
Deliverables:

Screenshot: assignment_part5_web1_stopped.png (web-1 Apache stopped)

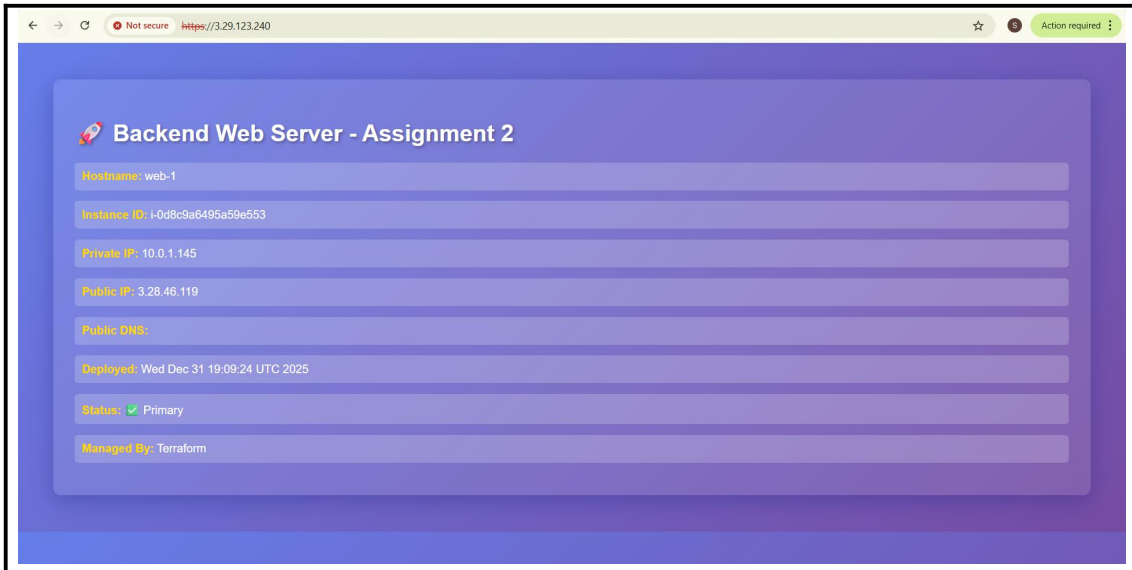




Screenshot: assignment_part5_web2_stopped.png (web-2 Apache stopped)



Screenshot: assignment_part5_backup_activated.png (web-3 serving traffic)



Screenshot: assignment_part5_nginx_error_log.png (error log showing backend failures)

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ ssh -i ~/.ssh/assignment2_key ec2-user@3.29.123.240
Last login: Wed Dec 31 19:26:29 2025 from 4.240.39.197
       #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-10-0-1-79 ~]$ sudo tail -f /var/log/nginx/error.log
2025/12/31 19:29:43 [notice] 669#669: start cache manager process 672
2025/12/31 19:29:43 [notice] 669#669: start cache loader process 673
2025/12/31 19:30:43 [notice] 673#673: http file cache: /var/cache/nginx 0.000M, bsize: 4096
2025/12/31 19:30:43 [notice] 669#669: signal 17 (SIGCHLD) received from 673
2025/12/31 19:30:43 [notice] 669#669: cache loader process 673 exited with code 0
2025/12/31 19:30:43 [notice] 669#669: signal 29 (SIGIO) received
2025/12/31 19:43:48 [error] 671#671: *46 connect() failed (111: Connection refused) while connecting to upstream, client: 5.149.130.174, server: _, request:
 "GET / HTTP/1.1", upstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 19:43:48 [warn] 671#671: *46 upstream server temporarily disabled while connecting to upstream, client: 5.149.130.174, server: _, request: "GET /
 HTTP/1.1", upstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 19:43:48 [error] 671#671: *46 connect() failed (111: Connection refused) while connecting to upstream, client: 5.149.130.174, server: _, request:
 "GET / HTTP/1.1", upstream: "http://10.0.1.145:80/", host: "3.29.123.240"
2025/12/31 19:43:48 [warn] 671#671: *46 upstream server temporarily disabled while connecting to upstream, client: 5.149.130.174, server: _, request: "GET /
 HTTP/1.1", upstream: "http://10.0.1.145:80/", host: "3.29.123.240"
```

Screenshot: assignment_part5_services_restored.png (all services back online)

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ ssh -i ~/.ssh/assignment2_key ec2-user@3.28.46.119
Last login: Wed Dec 31 19:41:19 2025 from 4.240.39.197
       #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@web-1 ~]$ sudo systemctl start httpd
[ec2-user@web-1 ~]$ exit
logout
Connection to 3.28.46.119 closed.
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ ssh -i ~/.ssh/assignment2_key ec2-user@158.252.78.6
Last login: Wed Dec 31 19:43:13 2025 from 4.240.39.197
       #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-10-0-1-195 ~]$ sudo systemctl start httpd
[ec2-user@ip-10-0-1-195 ~]$
```

5.5 Security & Performance Analysis (5 marks)

Analyze the security headers and performance of your Nginx setup.

Deliverables:

Screenshot: assignment_part5_ssl_certificate.png (certificate details)

```
[ec2-user@ip-10-0-1-79 ~]$ sudo openssl x509 -in /etc/ssl/certs/selfsigned.crt -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            91:fd:56:03:2b:24:4b:95
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: CN=3.29.123.240
        Validity
            Not Before: Dec 31 19:29:42 2025 GMT
            Not After : Dec 31 19:29:42 2026 GMT
        Subject: CN=3.29.123.240
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:c1:ac:d8:1e:ca:ca:47:db:43:2f:40:fe:16:bd:
                    8d:3e:de:43:84:1c:69:cf:80:66:fb:38:d8:8b:47:
                    ce:7c:24:fc:fb:25:bc:ed:8d:1a:5c:1d:9f:27:a8:
                    1b:c3:a9:36:d7:5f:64:6b:25:07:42:af:17:9b:48:
                    92:10:f1:3e:75:f7:df:f5:0f:59:42:7f:e6:64:d9:
                    60:14:54:e1:bf:12:05:2d:88:9f:8d:66:b7:43:0c:
                    84:16:80:7c:2c:ef:ca:8c:b4:75:6d:bf:1c:92:bb:
                    c4:2b:84:9f:db:43:93:1c:c7:ca:ab:d7:b8:18:17:
                    b3:b2:a0:82:27:ca:49:3b:a0:fd:6e:c9:6d:c9:d9:
                    7d:62:79:c2:39:47:b3:eb:f1:cb:56:f0:b9:2d:cb:
                    bb:41:a2:bd:0d:41:80:d5:b5:c4:7c:cf:98:09:d3:
                    9f:d7:85:dd:c2:e4:d8:8f:69:26:5d:9f:6a:38:d4:
                    d0:e8:7f:d1:85:8b:c8:88:d1:10:88:af:df:4a:76:
                    de:ab:7f:62:b6:99:3b:e3:83:b8:e0:a8:f1:25:0c:
                    e9:2d:61:20:51:68:4f:51:ea:a0:a0:a4:33:8a:91:
                    24:2a:69:75:07:0f:fd:22:8d:be:d8:79:ce:35:46:
                    49:98:ef:52:d2:74:38:1f:8f:15:50:86:49:40:39:
                    d7:2f
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                5C:ED:BA:9A:E5:C1:58:88:D6:69:FA:0F:E5:2A:9D:34:7B:8C:AE:0A
            X509v3 Authority Key Identifier:
                keyid:5C:ED:BA:9A:E5:C1:58:88:D6:69:FA:0F:E5:2A:9D:34:7B:8C:AE:0A

            X509v3 Basic Constraints:
                CA:TRUE
    Signature Algorithm: sha256WithRSAEncryption
         bf:ff:b3:8c:2a:27:cb:0b:71:a0:9f:4d:07:26:68:f4:da:3b:
```

```
                    24:2a:69:75:07:0f:fd:22:8d:be:d8:79:ce:35:46:
                    49:98:ef:52:d2:74:38:1f:8f:15:50:86:49:40:39:
                    d7:2f
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                5C:ED:BA:9A:E5:C1:58:88:D6:69:FA:0F:E5:2A:9D:34:7B:8C:AE:0A
            X509v3 Authority Key Identifier:
                keyid:5C:ED:BA:9A:E5:C1:58:88:D6:69:FA:0F:E5:2A:9D:34:7B:8C:AE:0A

            X509v3 Basic Constraints:
                CA:TRUE
    Signature Algorithm: sha256WithRSAEncryption
         bf:ff:b3:8c:2a:27:cb:0b:71:a0:9f:4d:07:26:68:f4:da:3b:
         18:4c:1d:aa:c7:c9:38:a4:68:c5:5a:45:37:29:d0:f9:af:4d:
         97:dc:00:d5:d8:f3:9f:80:41:85:4a:55:a6:f3:5e:37:96:5b:
         b1:10:12:a2:a8:0d:84:99:ff:65:f6:93:bc:d7:df:0f:c2:db:
         23:72:ec:0d:8b:e4:1f:df:64:7e:93:0b:8d:81:5a:47:c0:ce:
         d5:c0:41:a6:3f:97:39:5f:82:5f:ef:31:a9:50:16:d8:7a:9a:
         d1:cd:ef:df:c8:83:dd:e2:b3:d8:8d:cf:af:76:39:ad:7c:70:
         07:28:fb:ad:42:ec:d2:73:9a:02:13:9d:b4:d7:c0:53:64:40:
         54:a5:0d:e4:bd:32:eb:d7:19:2d:eb:30:31:fb:83:a6:2d:a8:
         e2:4a:67:fc:be:3e:1c:a1:ae:e3:cf:99:fb:8a:7b:6c:72:e9:
         2b:a4:a4:9b:fb:f0:7e:4e:84:12:67:f4:2e:e7:2d:de:d0:a1:
         90:2d:67:c1:bf:2c:0b:2c:1f:bd:24:71:57:7f:30:ba:32:9b:
         59:89:00:4e:a7:cc:cd:89:dc:60:6b:bf:88:ff:8f:0f:ef:3d:
         f8:1c:d5:a0:05:84:02:b2:8a:62:da:e1:86:3c:58:f5:74:95:
         07:13:14:5a
[ec2-user@ip-10-0-1-79 ~]$
```

Screenshot: assignment_part5_security_headers. png (response headers showing security headers)

```
[ec2-user@ip-10-0-1-79 ~]$ curl -I -k https://localhost
HTTP/2 200
server: nginx/1.28.0
date: Wed, 31 Dec 2025 19:54:24 GMT
content-type: text/html; charset=UTF-8
content-length: 1507
vary: Accept-Encoding
curl: (92) HTTP/2 stream 1 was not closed cleanly: PROTOCOL_ERROR (err 1)
[ec2-user@ip-10-0-1-79 ~]$
```

Screenshot: assignment_part5_http_redirect.png (301 redirect test)

```
[ec2-user@ip-10-0-1-79 ~]$ curl -I -k https://localhost
HTTP/2 200
server: nginx/1.28.0
date: Wed, 31 Dec 2025 19:54:24 GMT
content-type: text/html; charset=UTF-8
content-length: 1507
vary: Accept-Encoding
curl: (92) HTTP/2 stream 1 was not closed cleanly: PROTOCOL_ERROR (err 1)
[ec2-user@ip-10-0-1-79 ~]$ curl -I http://localhost
HTTP/1.1 301 Moved Permanently
Server: nginx/1.28.0
Date: Wed, 31 Dec 2025 19:54:58 GMT
Content-Type: text/html
Content-Length: 169
Connection: keep-alive
Location: https://localhost/
```

Screenshot: assignment_part5_error_log_analysis.png (error log review)

```
[ec2-user@ip-10-0-1-79 ~]$ sudo tail -50 /var/log/nginx/error.log
2025/12/31 17:37:37 [error] 32189#32189: *5 connect() failed (111: Connection refused) while connecting to upstream, client: 216.180.246.105, server: , request: "GET / HT
TP/1.0", upstream: "http://10.0.1.46:80/"
2025/12/31 17:37:41 [error] 32189#32189: *10 connect() failed (111: Connection refused) while connecting to upstream, client: 216.180.246.105, server: , request: "GET / H
TTP/1.1", upstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 17:37:41 [error] 32189#32189: *10 no live upstreams while connecting to upstream, client: 216.180.246.105, server: , request: "GET / HTTP/1.1", upstream: "http
://backend_servers/", host: "3.29.123.240"
2025/12/31 17:42:56 [error] 32189#32189: *16 connect() failed (111: Connection refused) while connecting to upstream, client: 216.180.246.105, server: , request: "GET /ad
min/index.html HTTP/1.1", upstream: "http://10.0.1.195:80/admin/index.html", host: "3.29.123.240"
2025/12/31 17:48:21 [error] 32189#32189: *25 connect() failed (111: Connection refused) while connecting to upstream, client: 103.19.49.30, server: , request: "GET / HTTP
/1.1", upstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 17:49:21 [error] 32189#32189: *25 upstream timed out (110: Connection timed out) while connecting to upstream, client: 103.19.49.30, server: , request: "GET /
HTTP/1.1", upstream: "http://9.0.1.145:80/", host: "3.29.123.240"
2025/12/31 17:49:21 [error] 32189#32189: *25 connect() failed (111: Connection refused) while connecting to upstream, client: 103.19.49.30, server: , request: "GET / HTTP
/1.1", upstream: "http://10.0.1.46:80/", host: "3.29.123.240"
2025/12/31 17:49:22 [error] 32189#32189: *25 connect() failed (111: Connection refused) while connecting to upstream, client: 103.19.49.30, server: , request: "GET /favic
on.ico HTTP/1.1", upstream: "http://10.0.1.195:80/favicon.ico", host: "3.29.123.240", referrer: "http://3.29.123.240/"
2025/12/31 17:49:22 [error] 32189#32189: *25 no live upstreams while connecting to upstream, client: 103.19.49.30, server: , request: "GET /favicon.ico HTTP/1.1", upstrea
m: "http://backend_servers/favicon.ico", host: "3.29.123.240", referrer: "http://3.29.123.240/"
2025/12/31 17:56:00 [error] 32189#32189: *37 connect() failed (111: Connection refused) while connecting to upstream, client: 216.180.246.105, server: , request: "GET /lo
gin.html HTTP/1.1", upstream: "http://10.0.1.195:80/login.html", host: "3.29.123.240"
2025/12/31 17:57:15 [error] 32188#32188: *40 connect() failed (111: Connection refused) while connecting to upstream, client: 185.16.39.146, server: , request: "GET / HTT
P/1.1", upstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 18:01:02 [error] 32431#32431: *3 upstream timed out (110: Connection timed out) while connecting to upstream, client: 3.29.123.240, server: , request: "GET / H
TTP/1.1", upstream: "http://9.0.1.145:80/", host: "3.29.123.240"
2025/12/31 18:01:02 [error] 32431#32431: *3 connect() failed (111: Connection refused) while connecting to upstream, client: 3.29.123.240, server: , request: "GET / HTTP/
1.1", upstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 18:01:02 [error] 32431#32431: *3 connect() failed (111: Connection refused) while connecting to upstream, client: 3.29.123.240, server: , request: "GET / HTTP/
1.1", upstream: "http://10.0.1.46:80/", host: "3.29.123.240"
2025/12/31 18:01:27 [error] 32430#32430: *7 connect() failed (111: Connection refused) while connecting to upstream, client: 216.180.246.105, server: , request: "GET /log
in HTTP/1.1", upstream: "http://10.0.1.195:80/login", host: "3.29.123.240"
2025/12/31 18:02:35 [error] 32430#32430: *10 upstream timed out (110: Connection timed out) while connecting to upstream, client: 3.29.123.240, server: , request: "HEAD /
 HTTP/1.1", upstream: "http://9.0.1.145:80/", host: "3.29.123.240"
2025/12/31 18:02:35 [error] 32430#32430: *10 connect() failed (111: Connection refused) while connecting to upstream, client: 3.29.123.240, server: , request: "HEAD / HTT
P/1.1", upstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 18:02:35 [error] 32430#32430: *10 connect() failed (111: Connection refused) while connecting to upstream, client: 3.29.123.240, server: , request: "HEAD / HTT
P/1.1", upstream: "http://10.0.1.46:80/", host: "3.29.123.240"
2025/12/31 18:06:13 [error] 32430#32430: *18 connect() failed (111: Connection refused) while connecting to upstream, client: 185.16.39.146, server: , request: "GET / HTT
P/1.1", upstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 18:08:09 [error] 32430#32430: *21 upstream timed out (110: Connection timed out) while connecting to upstream, client: 103.19.49.30, server: , request: "GET /
HTTP/1.1", upstream: "http://9.0.1.145:80/", host: "3.29.123.240"
2025/12/31 18:08:09 [error] 32430#32430: *21 connect() failed (111: Connection refused) while connecting to upstream, client: 103.19.49.30, server: , request: "GET / HTTP
/1.1", upstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 18:08:09 [error] 32430#32430: *21 connect() failed (111: Connection refused) while connecting to upstream, client: 103.19.49.30, server: , request: "GET / HTTP
```

Screenshot: assignment_part5_access_log_analysis.png (access log patterns)

```
[ec2-user@ip-10-0-1-79 ~]$ sudo tail -50 /var/log/nginx/access.log
216.180.246.105 - - [31/Dec/2025:18:11:13 +0000] "GET /web/ HTTP/1.1" 499 0 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com/genomecrawler)'
185.16.39.146 - - [31/Dec/2025:18:14:20 +0000] "GET / HTTP/1.1" 499 0 "-" "Wget"
216.180.246.105 - - [31/Dec/2025:18:14:33 +0000] "GET //webpages/login.html HTTP/1.1" 499 0 "-" "'Mozilla/5.0 (compatible; GenomeCrawlerd/1.0; +https://www.nokia.com
mecrawler)'"
185.16.39.146 - - [31/Dec/2025:18:26:27 +0000] "GET / HTTP/1.1" 499 0 "-" "Wget"
103.19.49.30 - - [31/Dec/2025:18:34:03 +0000] "GET / HTTP/1.1" 403 3630 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
.0.0 Safari/537.36"
72.14.178.148 - - [31/Dec/2025:18:34:54 +0000] "" 400 0 "-" "-"
103.19.49.30 - - [31/Dec/2025:18:35:03 +0000] "GET /icons/apache_pb2.gif HTTP/1.1" 200 4234 "http://3.29.123.240/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWe
537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.19.49.30 - - [31/Dec/2025:18:35:04 +0000] "GET /favicon.ico HTTP/1.1" 404 196 "http://3.29.123.240/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.3
TML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
72.14.178.148 - - [31/Dec/2025:18:35:13 +0000] "\x12\x01\x00/\x00\x00\x02\x00\x00\x00\x1A\x00\x06\x01\x00\x02\x00\x01\x02\x00!\x00\x01\x03\x00\x22\x00\x04\x04\x00&\x
1\xFF\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00" 400 157 "-" "-"
185.16.39.146 - - [31/Dec/2025:18:36:02 +0000] "GET / HTTP/1.1" 403 3630 "-" "Wget"
103.19.49.30 - - [31/Dec/2025:18:36:21 +0000] "GET / HTTP/1.1" 403 3630 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
.0.0 Safari/537.36"
103.19.49.30 - - [31/Dec/2025:18:36:31 +0000] "GET /icons/apache_pb2.gif HTTP/1.1" 499 0 "http://3.29.123.240/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebK
.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36"
103.19.49.30 - - [31/Dec/2025:18:38:59 +0000] "GET / HTTP/1.1" 403 3630 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
.0.0 Safari/537.36"
177.67.170.186 - - [31/Dec/2025:18:41:42 +0000] "GET / HTTP/1.0" 403 3630 "-" "-"
185.16.39.146 - - [31/Dec/2025:18:44:45 +0000] "GET / HTTP/1.1" 403 3630 "-" "Wget"
185.16.39.146 - - [31/Dec/2025:18:55:27 +0000] "GET / HTTP/1.1" 499 0 "-" "Wget"
93.174.93.12 - - [31/Dec/2025:19:03:50 +0000] "\x16\x03\x02\x01o\x01\x00\x01k\x03\x02RH\xC5\x1A#\xF7:N\xDF\xE2\xB4\x82/\xFF\x09T\x9F\xA7\xC4y\xB0h\xC6\x13\x8C\xA4\x1
2\xE1\x1A\x98 \x84\xB4,\x85\xAFn\xE3Y\xBBbhl\xFF(=':\xA9\x82\xD9o\xC8\xA2\xD7\x93\x98\xB4\xEF\x80\xE5\xB9\x90\x00(\xC0" 400 157 "-" "-"
185.16.39.146 - - [31/Dec/2025:19:05:13 +0000] "GET / HTTP/1.1" 499 0 "-" "Wget"
141.98.11.140 - - [31/Dec/2025:19:08:03 +0000] "GET / HTTP/1.1" 499 0 "-" "-"
204.76.203.87 - - [31/Dec/2025:19:08:04 +0000] "CONNECT www.roblox.com:443 HTTP/1.1" 400 157 "-" "-"
172.236.228.245 - - [31/Dec/2025:19:11:05 +0000] "GET / HTTP/1.1" 499 0 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 13_1) AppleWebKit/537.36 (KHTML, like Gecko) Chro
8.0.0.0 Safari/537.36"
172.236.228.245 - - [31/Dec/2025:19:11:06 +0000] "\x16\x03\x01\x00{\x01\x00\x00w\x03\x03\x03\x8C\xB3\x83\xBD:8\xCDt+!\xFB\xA2\xF1\x7F\xFB?P\xBB\xA7\x85\xBBu\x87k\xE5
\xEB\xDA'?\x00\x00\x1A\xC0/\xC0+\xC0\x11\xC0\x07\xC0\x13\xC0\x09\xC0\x14\xC0" 400 157 "-" "-"
185.16.39.146 - - [31/Dec/2025:19:15:49 +0000] "GET / HTTP/1.1" 499 0 "-" "Wget"
101.32.192.203 - - [31/Dec/2025:19:16:22 +0000] "HEAD /Core/Skin/Login.aspx HTTP/1.1" 499 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 Gecko) Chrome/106.0.0.0 Safari/537.36"
167.99.142.36 - - [31/Dec/2025:19:19:08 +0000] "GET / HTTP/1.1" 499 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/118.0"
167.99.142.36 - - [31/Dec/2025:19:19:15 +0000] "\x16\x03\x01\x00u\x01\x00\x00q\x03\x03\xED<v6\x06\x88Y\xB8\x01\x9B\xDD\x19\xAC\xF8\xC4LtD\x9D\xB9\xBAT\x15s\xC4\xAB\x
00 157 "-" "-"
185.16.39.146 - - [31/Dec/2025:19:23:02 +0000] "GET / HTTP/1.1" 403 3630 "-" "Wget"
103.163.238.70 - - [31/Dec/2025:19:31:42 +0000] "GET / HTTP/2.0" 200 638 "https://www.google.com/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KH
ike Gecko) Chrome/140.0.0.0 Safari/537.36" "-" Cache: MISS
103.163.238.70 - - [31/Dec/2025:19:31:43 +0000] "GET /favicon.ico HTTP/2.0" 404 172 "https://3.29.123.240/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/53
(KHTML, like Gecko) Chrome/140.0.0.0 Safari/537.36" "-" Cache: MISS
```

Bonus Tasks (10 marks extra credit)

Bonus 1: Custom Error Pages (3 marks)

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

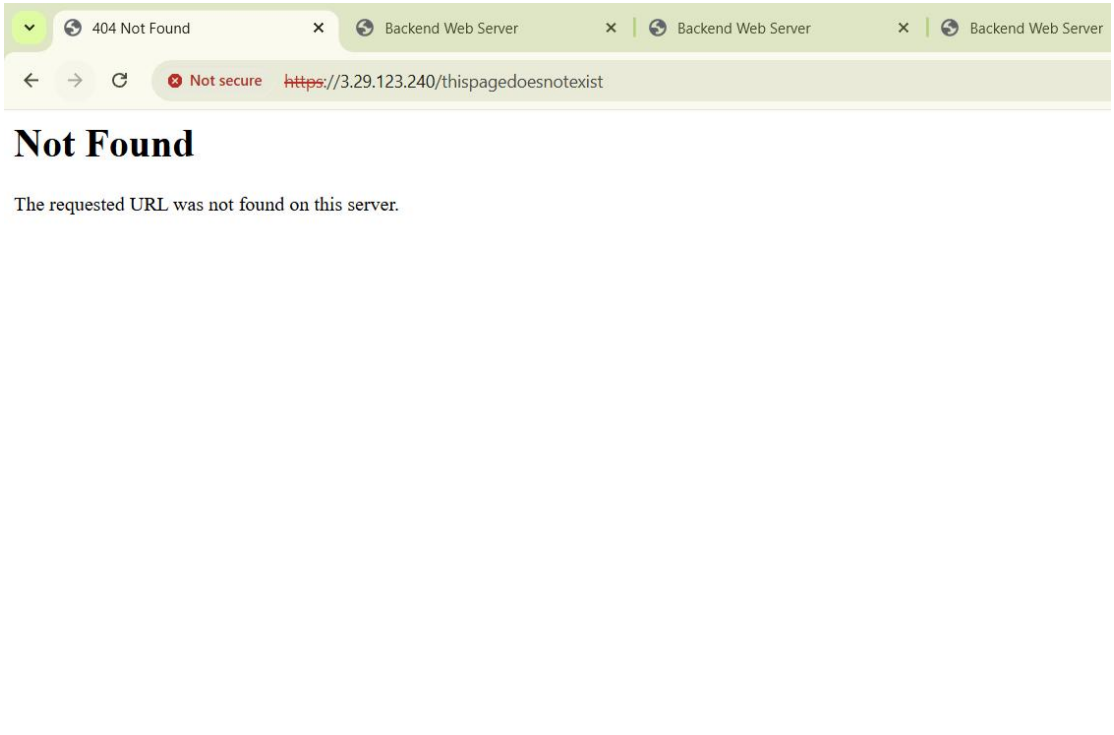
Deliverables:

```
[ec2-user@ip-10-0-1-79 ~]$ client_loop: send disconnect: Broken pipe
@23-22411-057-sudo → /workspaces/exam_prep/Assignment2 (main) $ ssh -i ~/.ssh/assignment2_key ec2-user@3.29.123.240
Last login: Wed Dec 31 19:52:44 2025 from 4.240.39.197
       #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

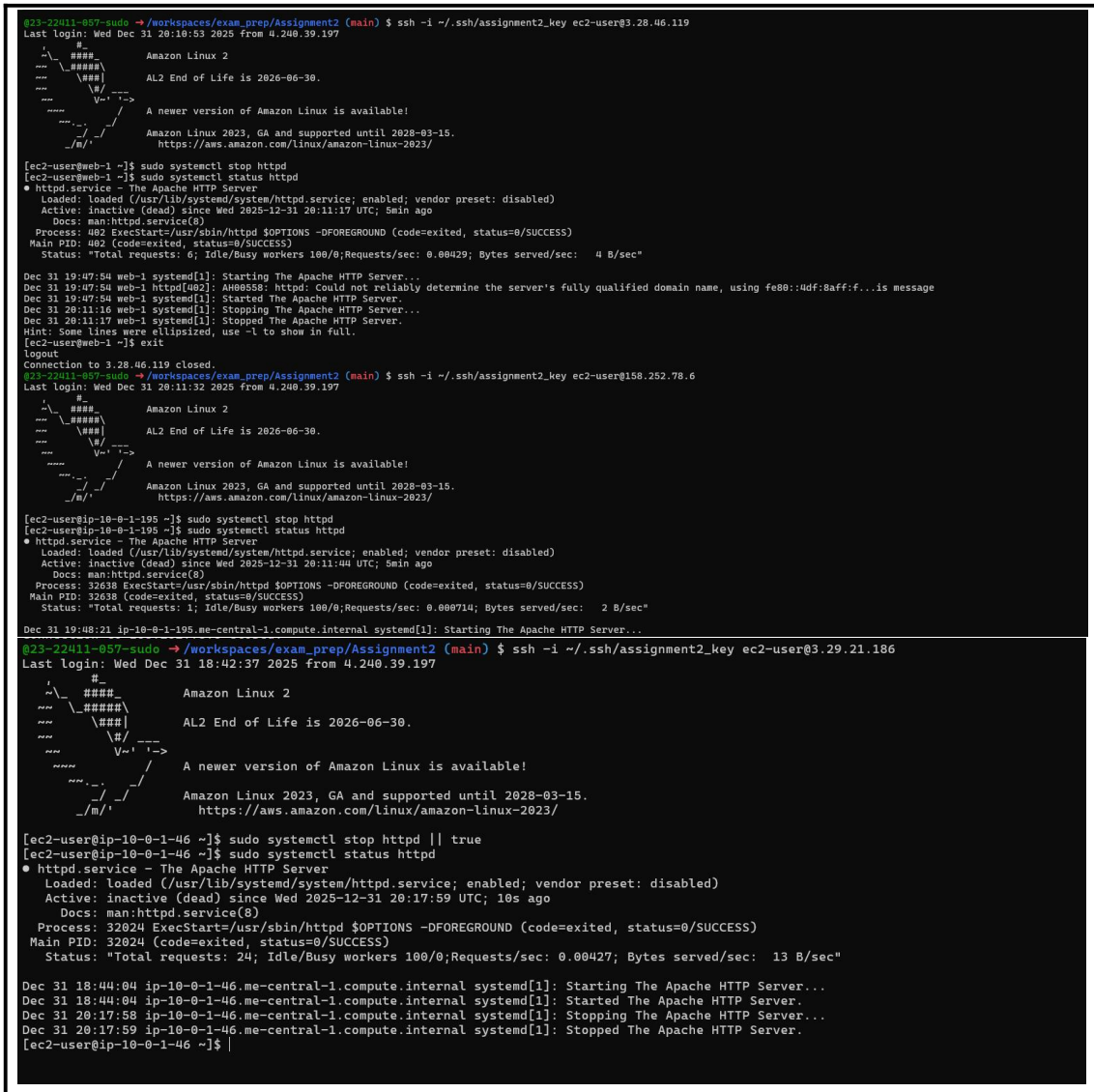
[ec2-user@ip-10-0-1-79 ~]$ sudo mkdir -p /usr/share/nginx/html/errors


[ec2-user@ip-10-0-1-79 ~]$ sudo nginx -t
nginx: [warn] the "listen ... http2" directive is deprecated, use the "http2" directive instead in /etc/nginx/nginx.conf:39
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
[ec2-user@ip-10-0-1-79 ~]$ sudo systemctl reload nginx
[ec2-user@ip-10-0-1-79 ~]$ curl -k https://3.29.123.240/thispagedoesnotexist
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL was not found on this server.</p>
</body></html>
[ec2-user@ip-10-0-1-79 ~]$
```

Screenshot: bonus1_custom_404.png



Screenshot: bonus1_custom_502.png



Bonus 2: Implement Rate Limiting (3 marks)

Add rate limiting to prevent abuse.

Configuration Example:

Deliverables:

Screenshot: bonus2_rate_limit_config.png

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ ssh -i ~/.ssh/assignment2_key ec2-user@3.29.123.240
Last login: Wed Dec 31 20:12:09 2025 from 4.240.39.197
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2026-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-10-0-1-79 ~]$ sudo cp /etc/nginx/nginx.conf /etc/nginx/nginx.conf.bak
[ec2-user@ip-10-0-1-79 ~]$ sudo nano /etc/nginx/nginx.conf
[ec2-user@ip-10-0-1-79 ~]$ sudo nano /etc/nginx/nginx.conf
[ec2-user@ip-10-0-1-79 ~]$ sudo nginx -t
nginx: [warn] the "listen ... http2" directive is deprecated, use the "http2" directive instead in /etc/nginx/nginx.conf:46
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
[ec2-user@ip-10-0-1-79 ~]$ sudo systemctl reload nginx
[ec2-user@ip-10-0-1-79 ~]$ # From your local machine
[ec2-user@ip-10-0-1-79 ~]$ for i in {1..30}; do curl -s -o /dev/null -w "%{http_code}\n" https://3.29.123.240 --insecure; done
301
301
301
301
301
301
301
301
301
301
301
301
301
301
301
301
301
301
301
301
301
301
```

Screenshot: bonus2_rate_limit_test.png

```
[ec2-user@ip-10-0-1-79 ~]$ sudo tail -20 /var/log/nginx/access.log
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 301 169 "-" "curl/8.3.0" "-" Cache: EXPIRED
10.0.1.79 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/1.0" 301 169 "-" "curl/8.3.0" "3.29.123.240" Cache: -
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 301 169 "-" "curl/8.3.0" "-" Cache: EXPIRED
10.0.1.79 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/1.0" 301 169 "-" "curl/8.3.0" "3.29.123.240" Cache: -
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 301 169 "-" "curl/8.3.0" "-" Cache: EXPIRED
10.0.1.79 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/1.0" 301 169 "-" "curl/8.3.0" "3.29.123.240" Cache: -
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 301 169 "-" "curl/8.3.0" "-" Cache: EXPIRED
10.0.1.79 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/1.0" 301 169 "-" "curl/8.3.0" "3.29.123.240" Cache: -
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 301 169 "-" "curl/8.3.0" "-" Cache: EXPIRED
10.0.1.79 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/1.0" 301 169 "-" "curl/8.3.0" "3.29.123.240" Cache: -
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 301 169 "-" "curl/8.3.0" "-" Cache: EXPIRED
10.0.1.79 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/1.0" 301 169 "-" "curl/8.3.0" "3.29.123.240" Cache: -
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 301 169 "-" "curl/8.3.0" "-" Cache: EXPIRED
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 404 153 "-" "curl/8.3.0" "-" Cache: -
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 404 153 "-" "curl/8.3.0" "-" Cache: -
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 404 153 "-" "curl/8.3.0" "-" Cache: -
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 404 153 "-" "curl/8.3.0" "-" Cache: -
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 404 153 "-" "curl/8.3.0" "-" Cache: -
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 404 153 "-" "curl/8.3.0" "-" Cache: -
3.29.123.240 - - [31/Dec/2025:20:41:24 +0000] "GET / HTTP/2.0" 404 153 "-" "curl/8.3.0" "-" Cache: -
[ec2-user@ip-10-0-1-79 ~]$ sudo tail -20 /var/log/nginx/error.log
2025/12/31 20:41:23 [error] 1355#1355: *93 connect() failed (111: Connection refused) while connecting to upstream, client: 3.29.123.240, server: _, request: "GET / HTTP/
2.0", upstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 20:41:23 [warn] 1355#1355: *93 upstream server temporarily disabled while connecting to upstream, client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", u
pstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 20:41:23 [error] 1356#1356: *98 connect() failed (111: Connection refused) while connecting to upstream, client: 3.29.123.240, server: _, request: "GET / HTTP/
2.0", upstream: "http://10.0.1.145:80/", host: "3.29.123.240"
2025/12/31 20:41:23 [warn] 1356#1356: *98 upstream server temporarily disabled while connecting to upstream, client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", u
pstream: "http://10.0.1.145:80/", host: "3.29.123.240"
2025/12/31 20:41:23 [error] 1356#1356: *98 connect() failed (111: Connection refused) while connecting to upstream, client: 3.29.123.240, server: _, request: "GET / HTTP/
2.0", upstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 20:41:23 [warn] 1356#1356: *98 upstream server temporarily disabled while connecting to upstream, client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", u
pstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 20:41:24 [error] 1356#1356: *166 limiting requests, excess: 20.770 by zone "mylimit", client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", host: "3.29.1
23.240"
2025/12/31 20:41:24 [error] 1356#1356: *166 open() "/var/www/html/custom_503.html" failed (2: No such file or directory), client: 3.29.123.240, server: _, request: "GET /
 HTTP/2.0", host: "3.29.123.240"
2025/12/31 20:41:24 [error] 1356#1356: *167 limiting requests, excess: 20.690 by zone "mylimit", client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", host: "3.29.1
23.240"
2025/12/31 20:41:24 [error] 1356#1356: *167 open() "/var/www/html/custom_503.html" failed (2: No such file or directory), client: 3.29.123.240, server: _, request: "GET /
 HTTP/2.0", host: "3.29.123.240"
2025/12/31 20:41:24 [error] 1356#1356: *168 limiting requests, excess: 20.600 by zone "mylimit", client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", host: "3.29.1
23.240"
2025/12/31 20:41:24 [error] 1356#1356: *168 open() "/var/www/html/custom_503.html" failed (2: No such file or directory), client: 3.29.123.240, server: _, request: "GET /
```

```
[ec2-user@ip-10-0-1-79 ~]$ sudo tail -20 /var/log/nginx/error.log
2025/12/31 20:41:23 [error] 1355#1355: *93 connect() failed (111: Connection refused) while connecting to upstream, client: 3.29.123.240, server: _, request: "GET / HTTP/
2.0", upstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 20:41:23 [warn] 1355#1355: *93 upstream server temporarily disabled while connecting to upstream, client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", u
pstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 20:41:23 [error] 1356#1356: *98 connect() failed (111: Connection refused) while connecting to upstream, client: 3.29.123.240, server: _, request: "GET / HTTP/
2.0", upstream: "http://10.0.1.145:80/", host: "3.29.123.240"
2025/12/31 20:41:23 [warn] 1356#1356: *98 upstream server temporarily disabled while connecting to upstream, client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", u
pstream: "http://10.0.1.145:80/", host: "3.29.123.240"
2025/12/31 20:41:23 [error] 1356#1356: *98 connect() failed (111: Connection refused) while connecting to upstream, client: 3.29.123.240, server: _, request: "GET / HTTP/
2.0", upstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 20:41:23 [warn] 1356#1356: *98 upstream server temporarily disabled while connecting to upstream, client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", u
pstream: "http://10.0.1.195:80/", host: "3.29.123.240"
2025/12/31 20:41:24 [error] 1356#1356: *166 limiting requests, excess: 20.770 by zone "mylimit", client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", host: "3.29.1
23.240"
2025/12/31 20:41:24 [error] 1356#1356: *166 open() "/var/www/html/custom_503.html" failed (2: No such file or directory), client: 3.29.123.240, server: _, request: "GET /
 HTTP/2.0", host: "3.29.123.240"
2025/12/31 20:41:24 [error] 1356#1356: *167 limiting requests, excess: 20.690 by zone "mylimit", client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", host: "3.29.1
23.240"
2025/12/31 20:41:24 [error] 1356#1356: *167 open() "/var/www/html/custom_503.html" failed (2: No such file or directory), client: 3.29.123.240, server: _, request: "GET /
 HTTP/2.0", host: "3.29.123.240"
2025/12/31 20:41:24 [error] 1356#1356: *168 limiting requests, excess: 20.600 by zone "mylimit", client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", host: "3.29.1
23.240"
2025/12/31 20:41:24 [error] 1356#1356: *168 open() "/var/www/html/custom_503.html" failed (2: No such file or directory), client: 3.29.123.240, server: _, request: "GET /
 HTTP/2.0", host: "3.29.123.240"
2025/12/31 20:41:24 [error] 1356#1356: *169 limiting requests, excess: 20.510 by zone "mylimit", client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", host: "3.29.1
23.240"
2025/12/31 20:41:24 [error] 1356#1356: *169 open() "/var/www/html/custom_503.html" failed (2: No such file or directory), client: 3.29.123.240, server: _, request: "GET /
 HTTP/2.0", host: "3.29.123.240"
2025/12/31 20:41:24 [error] 1356#1356: *170 limiting requests, excess: 20.420 by zone "mylimit", client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", host: "3.29.1
23.240"
2025/12/31 20:41:24 [error] 1356#1356: *170 open() "/var/www/html/custom_503.html" failed (2: No such file or directory), client: 3.29.123.240, server: _, request: "GET /
 HTTP/2.0", host: "3.29.123.240"
2025/12/31 20:41:24 [error] 1356#1356: *171 limiting requests, excess: 20.310 by zone "mylimit", client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", host: "3.29.1
23.240"
2025/12/31 20:41:24 [error] 1356#1356: *171 open() "/var/www/html/custom_503.html" failed (2: No such file or directory), client: 3.29.123.240, server: _, request: "GET /
 HTTP/2.0", host: "3.29.123.240"
2025/12/31 20:41:24 [error] 1356#1356: *172 limiting requests, excess: 20.220 by zone "mylimit", client: 3.29.123.240, server: _, request: "GET / HTTP/2.0", host: "3.29.1
23.240"
2025/12/31 20:41:24 [error] 1356#1356: *172 open() "/var/www/html/custom_503.html" failed (2: No such file or directory), client: 3.29.123.240, server: _, request: "GET /
 HTTP/2.0", host: "3.29.123.240"
[ec2-user@ip-10-0-1-79 ~]$
```

Bonus 3: Health Check Automation (4 marks)

Deliverables:

Health check script

```
  GNU nano 2.9.8                                                    /home/ec2-user/health_check.sh

# Paste the code above
#!/bin/bash
# Health Check Script for Backend Servers
# Monitors web-1, web-2, web-3 every 30 seconds

LOG_FILE="/var/log/backend_health.log"
SERVERS=("10.0.1.145" "10.0.1.195" "10.0.1.79") # web-1, web-2, web-3
USER="ec2-user"
SSH_KEY="/home/ec2-user/.ssh/assignment2_key"

echo "===== Health Check Started at $(date) =====" >> $LOG_FILE

while true; do
    for SERVER in "${SERVERS[@]}"; do
        # Test HTTP connectivity
        HTTP_STATUS=$(curl -s -o /dev/null -w "%{http_code}" http://$SERVER)

        TIMESTAMP=$(date +"%Y-%m-%d %H:%M:%S")
        if [ "$HTTP_STATUS" -eq 200 ]; then
            echo "$TIMESTAMP - $SERVER is UP (HTTP $HTTP_STATUS)" >> $LOG_FILE
        else
            echo "$TIMESTAMP - ALERT! $SERVER is DOWN (HTTP $HTTP_STATUS)" >> $LOG_FILE

            # Try to restart Apache via SSH
            ssh -i $SSH_KEY -o StrictHostKeyChecking=no $USER@$SERVER "sudo systemctl restart httpd" \
            && echo "$TIMESTAMP - $SERVER Apache restarted via SSH" >> $LOG_FILE \
            || echo "$TIMESTAMP - $SERVER Apache restart FAILED" >> $LOG_FILE
        fi
    done

    # Wait 30 seconds before next check
    sleep 30
done
```

Screenshot: bonus3_health_check_script.png

```
[ec2-user@ip-10-0-1-79 ~]$ sudo nano /home/ec2-user/health_check.sh
[ec2-user@ip-10-0-1-79 ~]$ sudo chmod +x /home/ec2-user/health_check.sh
[ec2-user@ip-10-0-1-79 ~]$ sudo nohup /home/ec2-user/health_check.sh &
[1] 1402
[ec2-user@ip-10-0-1-79 ~]$ nohup: ignoring input and appending output to 'nohup.out'
^C
[ec2-user@ip-10-0-1-79 ~]$ sudo nohup /home/ec2-user/health_check.sh &
[2] 1418
[ec2-user@ip-10-0-1-79 ~]$ nohup: ignoring input and appending output to 'nohup.out'
```

Screenshot: bonus3_health_log.png

```
[ec2-user@ip-10-0-1-79 ~]$ tail -f /var/log/backend_health.log
2025-12-31 20:45:55 - ALERT! 10.0.1.195 is DOWN (HTTP 000)
2025-12-31 20:45:55 - 10.0.1.195 Apache restart FAILED
2025-12-31 20:45:55 - ALERT! 10.0.1.79 is DOWN (HTTP 301)
2025-12-31 20:45:55 - 10.0.1.79 Apache restart FAILED
2025-12-31 20:46:01 - ALERT! 10.0.1.145 is DOWN (HTTP 000)
2025-12-31 20:46:01 - 10.0.1.145 Apache restart FAILED
2025-12-31 20:46:02 - ALERT! 10.0.1.195 is DOWN (HTTP 000)
2025-12-31 20:46:02 - 10.0.1.195 Apache restart FAILED
2025-12-31 20:46:02 - ALERT! 10.0.1.79 is DOWN (HTTP 301)
2025-12-31 20:46:02 - 10.0.1.79 Apache restart FAILED
2025-12-31 20:46:25 - ALERT! 10.0.1.145 is DOWN (HTTP 000)
2025-12-31 20:46:25 - 10.0.1.145 Apache restart FAILED
2025-12-31 20:46:25 - ALERT! 10.0.1.195 is DOWN (HTTP 000)
2025-12-31 20:46:25 - 10.0.1.195 Apache restart FAILED
2025-12-31 20:46:25 - ALERT! 10.0.1.79 is DOWN (HTTP 301)
2025-12-31 20:46:25 - 10.0.1.79 Apache restart FAILED
2025-12-31 20:46:32 - ALERT! 10.0.1.145 is DOWN (HTTP 000)
2025-12-31 20:46:32 - 10.0.1.145 Apache restart FAILED
2025-12-31 20:46:32 - ALERT! 10.0.1.195 is DOWN (HTTP 000)
2025-12-31 20:46:32 - 10.0.1.195 Apache restart FAILED
2025-12-31 20:46:32 - ALERT! 10.0.1.79 is DOWN (HTTP 301)
2025-12-31 20:46:32 - 10.0.1.79 Apache restart FAILED
```

Part 6: Documentation & Cleanup (10 marks)

6.1 README Documentation (5 marks)

Screenshot: assignment_part6_readme.png (README.md content)

```
 GNU nano 7.2                                                README.md *
# Assignment 2 - Multi-Tier Web Infrastructure

## Project Overview
This project demonstrates a multi-tier web infrastructure using AWS EC2 instances, Apache backend servers, and an Nginx reverse proxy/load balancer.
It supports high availability with primary and backup servers, SSL/TLS encryption, caching, security headers, and automated health monitoring.

       Internet
          |
     HTTPS / HTTP
          |
      Nginx LB
      (Proxy &
       SSL/TLS)
          |
  ┌───────┬───────┐
  ▼       ▼       ▼

# Components Description
- **Web-1 & Web-2:** Apache backend servers serving dynamic web pages with metadata.
- **Web-3:** Backup Apache server activated during primary server failure.
- **Nginx Server:** Acts as HTTPS load balancer, reverse proxy, with caching and rate limiting.
- **Health Check Script:** Monitors backend servers every 30 seconds, logs status, and restarts Apache if needed.

---

## Prerequisites
- AWS account with EC2 access.
- AWS credentials configured on your system.
- SSH key pair for EC2 instances (`assignment2_key`).
- Terraform installed for infrastructure provisioning.

---

## Deployment Instructions

### Step 1: SSH into EC2 Instances
```bash
ssh -i ~/.ssh/assignment2_key ec2-user@<instance-public-ip>
```

6.2 Infrastructure Cleanup (5 marks)

Properly destroy all resources and verify cleanup.

Deliverables:

Screenshot: assignment_part6_terraform_destroy_prompt.png

```
$ terraform destroy
var.ami_id
  AMI ID for EC2 instances

  Enter a value: yes

var.key_name
  Name of the EC2 key pair

  Enter a value: key

data.http.my_ip: Reading...
data.http.my_ip: Read complete after 1s [id=https://icanhazip.com]
data.aws_ssm_parameter.amzn2: Reading...
module.networking.aws_vpc.this: Refreshing state... [id=vpc-0a0eee904a6c71e35]
data.aws_ssm_parameter.amzn2: Read complete after 1s [id=/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2]
```

Screenshot: assignment_part6_terraform_destroy_complete.png

```
        Backend Servers:
        - web-1: public IPs: 44.195.69.216, 98.86.148.130, 3.231.204.61, private IPs: 10.0.10.243, 10.0.10.87, 10.0.10.183
            - web-2: public IPs: 3.239.42.192, 34.236.243.157, 34.200.220.149, private IPs: 10.0.10.33, 10.0.10.219, 10.0.10.253
            - web-3: public IPs: 3.236.89.145, 18.207.111.59, 3.239.33.223, private IPs: 10.0.10.89, 10.0.10.242, 10.0.10.73

        ========================================
    EOT -> null
  - nginx_instance_id    = "i-0824b01d8ce9cb1b6" -> null
  - nginx_public_ip      = "18.232.146.156" -> null
  - subnet_id            = "subnet-02f4b48141eb76975" -> null
  - vpc_id               = "vpc-0a0eee904a6c71e35" -> 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.

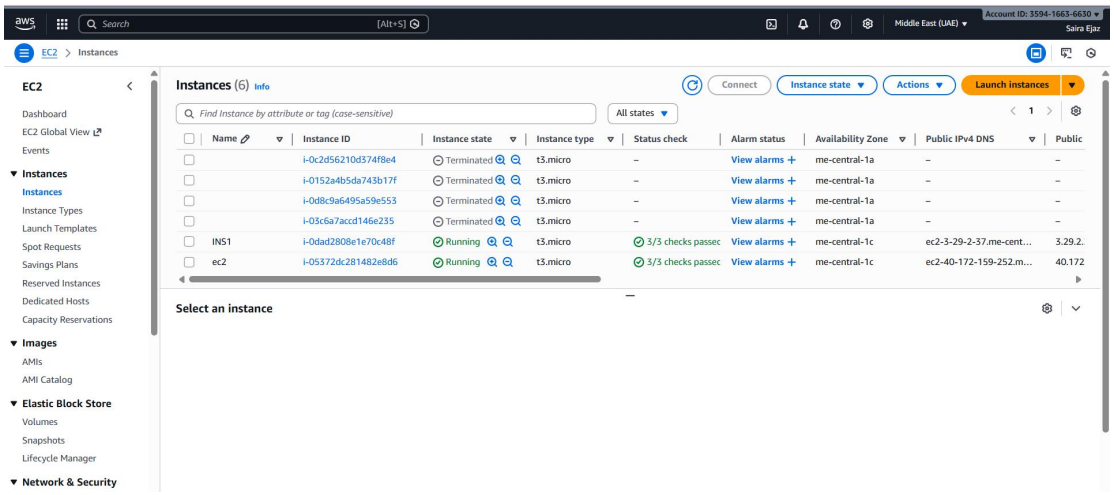
  Enter a value: yes

module.backend_servers["web-1"].aws_instance.backend[0]: Destroying... [id=i-08601a87b94e93fe8]
module.backend_servers["web-2"].aws_instance.backend[0]: Destroying... [id=i-049dfb0d3a79a8744]
module.backend_servers["web-1"].aws_instance.backend[1]: Destroying... [id=i-01a374de43288ab32]
module.backend_servers["web-2"].aws_instance.backend[2]: Destroying... [id=i-0bb6fbc5dba72cdca]
module.backend_servers["web-3"].aws_instance.backend[2]: Destroying... [id=i-06f0bb724ced023a8]
module.backend_servers["web-2"].aws_instance.backend[1]: Destroying... [id=i-00ddb751f17464a08]
module.nginx_server.aws_instance.this: Destroying... [id=i-0824b01d8ce9cb1b6]
module.backend_servers["web-1"].aws_instance.backend[2]: Destroying... [id=i-047ef7dc11056549e]
module.backend_servers["web-3"].aws_instance.backend[0]: Destroying... [id=i-0bfe5c3d42178e0c5]
module.backend_servers["web-3"].aws_instance.backend[1]: Destroying... [id=i-0629846db90668695]
```

Screenshot: assignment_part6_aws_instances_destroyed.png (AWS Console showing no instances)
```

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ aws ec2 terminate-instances --instance-ids i-0c2d56210d374f8e4 i-0152a4b5da743b17f i-0d8c9a6495a59e553 i-03c6a7accd146e235
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-0d8c9a6495a59e553",
            "CurrentState": {
                "Code": 48,
                "Name": "terminated"
            },
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-0d8c9a6495a59e553",
            "CurrentState": {
                "Code": 48,
                "Name": "terminated"
            },
            "PreviousState": {
                "Code": 48,
                "Name": "terminated"
            }
        },
        {
            "InstanceId": "i-0152a4b5da743b17f",
            "CurrentState": {
                "Code": 48,
                "Name": "terminated"
            },
            "PreviousState": {
                "Code": 48,
                "Name": "terminated"
            }
        },
        {
            "InstanceId": "i-0c2d56210d374f8e4",
            "CurrentState": {
                "Code": 48,
                "Name": "terminated"
            },
            "PreviousState": {
                "Code": 48,
                "Name": "terminated"
            }
```

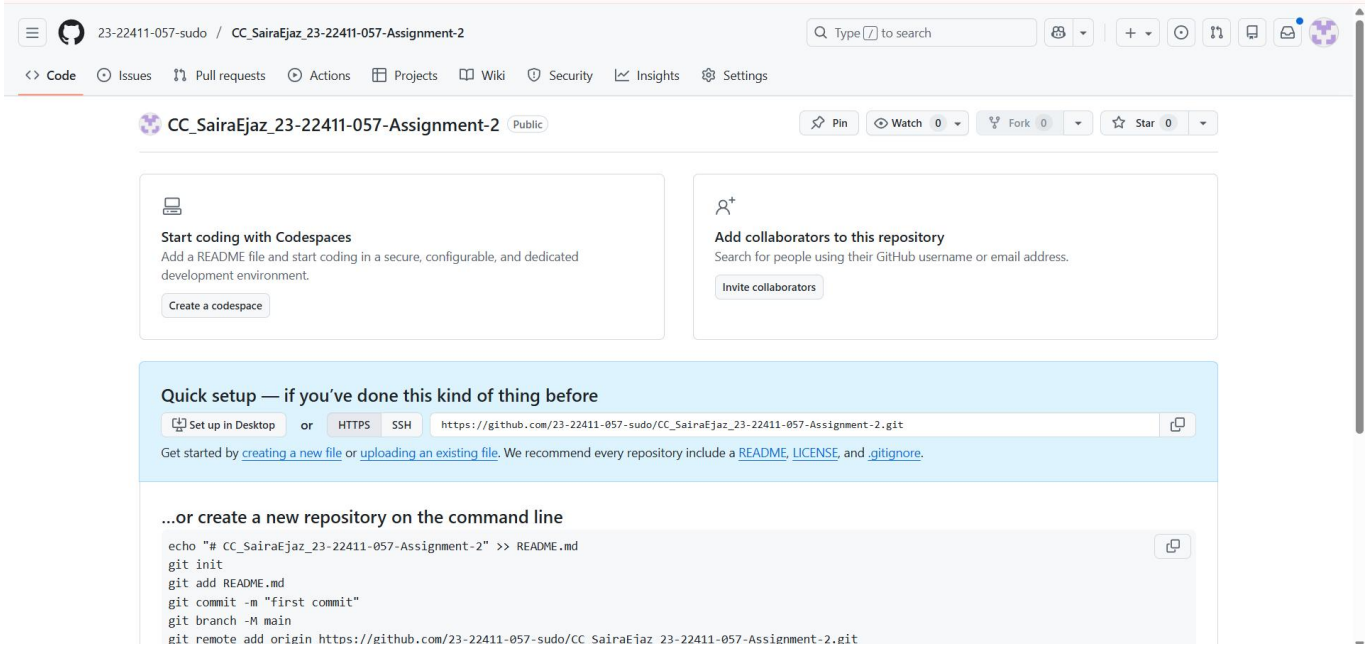
```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ [200~aws ec2 describe-instances --filters "Name=tag:Project,Values=Assignment-2" --query "Reservations[].Instances[].InstanceId"~
-bash: [200~aws: command not found
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $
```



Screenshot: assignment_part6_empty_state. png (empty terraform.tfstate)

```
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $ cat terraform.tfstate
{
  "version": 4,
  "terraform_version": "1.14.3",
  "serial": 150,
  "lineage": "09eefdfa-d5d9-645f-6967-caf6392ed0a8",
  "outputs": {},
  "resources": [],
  "check_results": null
}
@23-22411-057-sudo ➜ /workspaces/exam_prep/Assignment2 (main) $
```

- ***Github Submission:***




```
saira@DESKTOP-L0CU5EF MINGW64 ~ (main)
$ cd Assignment2

saira@DESKTOP-L0CU5EF MINGW64 ~/Assignment2 (main)
$ ls
README.md  locals.tf  main.tf  modules/  outputs.tf  scripts/  terraform.tfvars  variables.tf

saira@DESKTOP-L0CU5EF MINGW64 ~/Assignment2 (main)
$ ls -la
total 35
drwxr-xr-x 1 saira 197609    0 Dec 31 14:22 ./
drwxr-xr-x 1 saira 197609    0 Dec 26 20:43 ../
-rw-r--r-- 1 saira 197609  205 Dec 26 21:10 .gitignore
-rw-r--r-- 1 saira 197609  748 Dec 26 21:12 README.md
-rw-r--r-- 1 saira 197609    0 Dec 26 20:48 locals.tf
-rw-r--r-- 1 saira 197609 3722 Dec 26 23:24 main.tf
drwxr-xr-x 1 saira 197609    0 Dec 26 20:46 modules/
-rw-r--r-- 1 saira 197609  707 Dec 26 23:24 outputs.tf
drwxr-xr-x 1 saira 197609    0 Dec 26 20:51 scripts/
-rw-r--r-- 1 saira 197609  512 Dec 26 22:18 terraform.tfvars
-rw-r--r-- 1 saira 197609 1273 Dec 26 22:18 variables.tf
```

```
saira@DESKTOP-LOCU5EF MINGW64 ~/Assignment2 (main)
$ git init
Initialized empty Git repository in C:/Users/saira/Assignment2/.git/

saira@DESKTOP-LOCU5EF MINGW64 ~/Assignment2 (master)
$ git add .

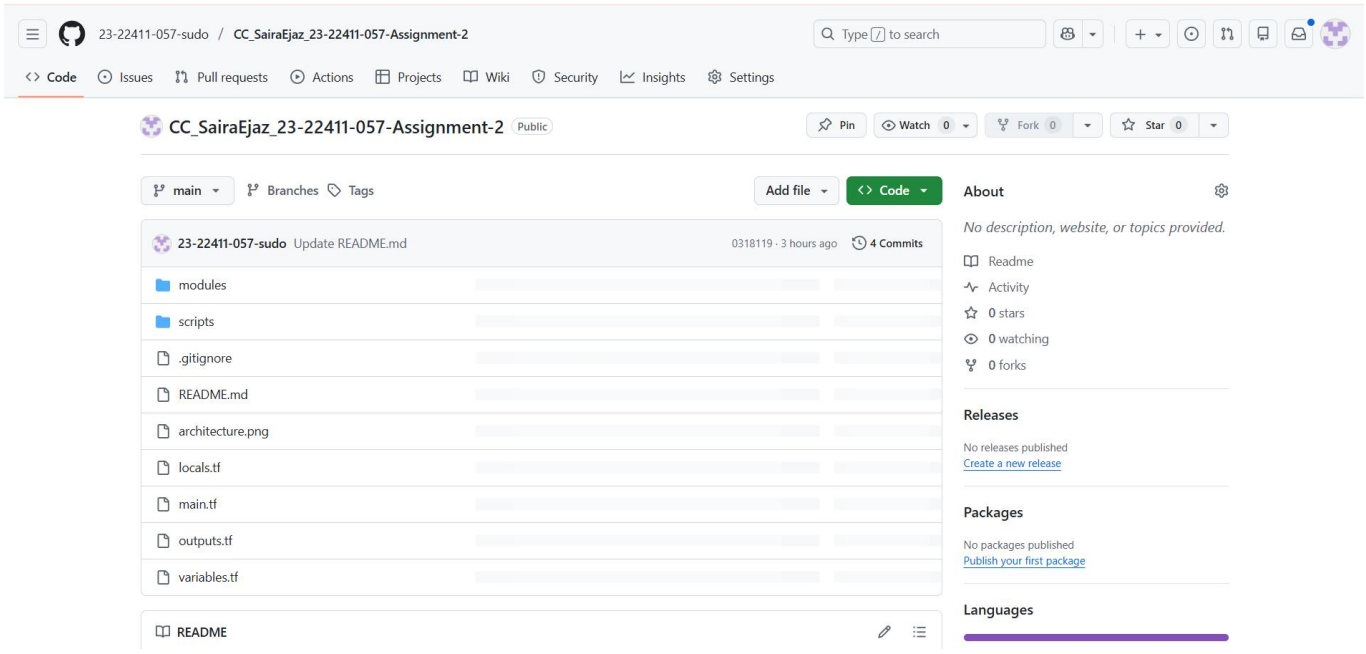
saira@DESKTOP-LOCU5EF MINGW64 ~/Assignment2 (master)
$ git commit -m "Assignment 2 initial commit"
[master (root-commit) 36cdb83] Assignment 2 initial commit
 17 files changed, 298 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 README.md
 create mode 100644 locals.tf
 create mode 100644 main.tf
 create mode 100644 modules/networking/main.tf
 create mode 100644 modules/networking/outputs.tf
 create mode 100644 modules/networking/variables.tf
 create mode 100644 modules/security/main.tf
 create mode 100644 modules/security/outputs.tf
 create mode 100644 modules/security/variables.tf
 create mode 100644 modules/webserver/main.tf
 create mode 100644 modules/webserver/outputs.tf
 create mode 100644 modules/webserver/variables.tf
 create mode 100644 outputs.tf
 create mode 100644 scripts/apache-setup.sh
 create mode 100644 scripts/nginx-setup.sh
 create mode 100644 variables.tf

saira@DESKTOP-LOCU5EF MINGW64 ~/Assignment2 (master)
$ git branch -M main

saira@DESKTOP-LOCU5EF MINGW64 ~/Assignment2 (main)
$ git remote add origin https://github.com/23-22411-057-sudo/CC_SairaEjaz_23-22411-057-Assignment-2.git

saira@DESKTOP-LOCU5EF MINGW64 ~/Assignment2 (main)
$ git remote -v
origin  https://github.com/23-22411-057-sudo/CC_SairaEjaz_23-22411-057-Assignment-2.git (fetch)
origin  https://github.com/23-22411-057-sudo/CC_SairaEjaz_23-22411-057-Assignment-2.git (push)

saira@DESKTOP-LOCU5EF MINGW64 ~/Assignment2 (main)
$ git push -u origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 16 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (11/11), 2.68 KiB | 548.00 KiB/s, done.
Total 11 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/23-22411-057-sudo/CC_SairaEjaz_23-22411-057-Assignment-2.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```



Repository Link:

https://github.com/23-22411-057-sudo/CC_SairaEjaz_23-22411-057-Assignment-2

Key Achievements:

It involves designing and deploying a multi-tier web application infrastructure on AWS using Terraform. It included creating modular components for networking, security, and webservers, implementing load

balancing with Nginx, and ensuring proper routing and access control within the VPC. The project also covered testing, monitoring, and resource cleanup, providing practical experience in cloud infrastructure management.

Challenges & Solutions

Problems Encountered:

Terraform attribute errors

EC2 termination issues

Load balancing verification

Solutions:

Corrected module outputs

Cleared state lock files

Used browser tools and logs to verify load balancing

Conclusion

this project provided hands-on experience in designing, deploying, and managing a multi-tier web infrastructure on AWS using Terraform. It strengthened understanding of modular infrastructure design, networking, security groups, and automated deployment processes. Through implementation and testing, skills in load balancing, caching, and high availability were enhanced. The project also highlighted the importance of proper documentation, resource cleanup, and troubleshooting in real-world cloud environments, preparing for future cloud and DevOps tasks.