



**FATIMA JINNAH WOMEN UNIVERSITY  
RAWALPINDI**

*Department of Software Engineering*

**NAME: SHUMAIL ZAHRA**

**REGISTRATION #: 2023-BSE-061**

**COURSE: CLOUD COMPUTING**

**DEPARTMENT: BSE**

**SEMESTER: 5B**

**SUBMITTED TO: Sir Waqas Saleem**

**Date: 31/12/25**

## **Assignment 2:**

**Advanced Terraform & Nginx Multi-Tier Architecture**

## Table of Content

1. Summary-----	3
2 Architecture Design-----	3
2.1 Architecture Overview-----	3
2.2 Architecture Diagram-----	4
2.3 System Components-----	4
2.4 Network Topology-----	5
2.5 Security Design-----	5
3 Implementation Details-----	6
3.1 Infrastructure Setup using Terraform-----	6
3.2 Webserver Module-----	16
3.3 Apache Backend Server Configuration-----	19
3.4 Infrastructure Deployment-----	22
3.5 Nginx Configuration-----	29
3.6 Cleanup-----	47
4. Testing Results-----	51
4.1 Load balancing tests-----	52
4.2 Cache performance tests-----	52
4.3 High availability tests-----	53
4.4 Security tests-----	53
4.5 Performance metrics-----	54
4.6 Summary of Testing Results-----	54
5. Challenges and Solutions-----	55
6. Conclusion-----	55
7. Appendices-----	56

## **1. Summary**

This assignment focuses on the design and deployment of a production-ready, multi-tier web infrastructure using Terraform and Nginx on Amazon Web Services (AWS). The objective of the assignment is to demonstrate practical knowledge of Infrastructure as Code (IaC), cloud networking, security best practices, and high-availability web architectures. The deployed infrastructure consists of one Nginx server acting as a reverse proxy and load balancer, and three backend Apache web servers. Two backend servers operate as primary servers, while the third functions as a backup server that automatically serves traffic in case of failure of the primary servers. Terraform modules are used to ensure code reusability, scalability, and clean organization of infrastructure components. Advanced Nginx configurations have been implemented, including HTTPS with self-signed SSL certificates, HTTP to HTTPS redirection, response caching, security headers, and upstream load balancing. The system is designed to improve performance, enhance security, and ensure uninterrupted service availability. The assignment also includes comprehensive testing, such as load balancing verification, cache performance validation, high-availability failover testing, and security analysis. Proper documentation, screenshots, and cleanup of cloud resources have been carried out to meet academic and professional standards. This project provides hands-on experience with real-world cloud deployment scenarios and reinforces key concepts of cloud computing, DevOps practices, and secure system design.

## **2. Architecture Design**

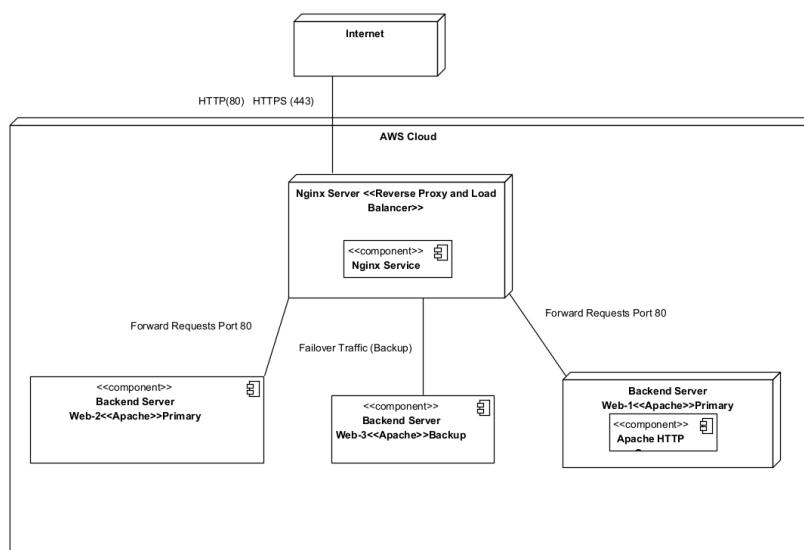
### **2.1 Architecture Overview**

The architecture follows a multi-tier design where the application is divided into distinct layers to enhance scalability,

security, and maintainability. The front layer consists of an Nginx reverse proxy, which handles incoming client requests, while the backend layer consists of multiple Apache web servers responsible for serving application content. All resources are deployed within a custom AWS Virtual Private Cloud (VPC) using Terraform. The Nginx server acts as a single-entry point for users and distributes incoming traffic across backend servers using load balancing techniques.

## 2.2 Architecture Diagram

The diagram represents a multi-tier web architecture deployed on AWS, where incoming requests from the Internet are first handled by an Nginx server acting as a reverse proxy and load balancer. Nginx manages HTTP and HTTPS traffic, performs SSL termination, applies caching, and forwards requests to the backend layer. The backend consists of three Apache web servers, with Web-1 and Web-2 serving as primary nodes that handle normal traffic through load balancing, while Web-3 is configured as a backup server to ensure high availability during failures. This architecture enhances security, performance, scalability, and fault tolerance of the application.



## 2.3 System Components

The main components of the system are:

## **Terraform**

Used as an Infrastructure as Code (IaC) tool to automate the provisioning and management of AWS resources.

## **Amazon VPC**

A custom virtual network providing isolation and control over networking resources.

## **Public Subnet**

Hosts the Nginx server and backend web servers with public IP addresses.

## **Nginx Server**

Acts as a reverse proxy and load balancer, providing HTTPS access, caching, and security headers.

## **Apache Backend Servers (Web-1, Web-2, Web-3)**

Serve dynamic HTML content. Web-3 is configured as a backup server.

## **Security Groups**

Control inbound and outbound traffic based on the principle of least privilege.

## **2.4 Network Topology**

The infrastructure is deployed in a single AWS availability zone for simplicity. The VPC contains one public subnet connected to the internet via an Internet Gateway. Routing rules allow external traffic to reach the Nginx server, while backend servers only accept traffic from the Nginx security group.

## **2.5 Security Design**

Security has been implemented using AWS Security Groups:

### **Nginx Security Group**

- Allows SSH (port 22) from the user's IP only
- Allows HTTP (80) and HTTPS (443) from anywhere
- Allows all outbound traffic

### **Backend Security Group**

- Allows SSH (port 22) from the user's IP only

- Allows HTTP (80) traffic only from the Nginx security group
- Blocks all other inbound access
- This design ensures backend servers are not directly accessible from the internet, significantly improving system security.

### 3. Implementation Details

#### 1. Infrastructure Setup (25 marks)

assignment\_part1\_project\_structure

```
Syed@DESKTOP-S50GK51 MINGW64 ~ (main)
$ mkdir -p Assignment2/modules/{networking,security,webserver}

Syed@DESKTOP-S50GK51 MINGW64 ~ (main)
$ mkdir -p Assignment2/scripts

Syed@DESKTOP-S50GK51 MINGW64 ~ (main)
$ cd Assignment2

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ touch main.tf variables.tf outputs.tf locals.tf terraform.tfvars README.md .gitignore

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ touch modules/networking/{main.tf,variables.tf,outputs.tf}

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ touch modules/security/{main.tf,variables.tf,outputs.tf}

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ touch modules/webserver/{main.tf,variables.tf,outputs.tf}

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ touch scripts/nginx-setup.sh scripts/apache-setup.sh

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ tree
bash: tree: command not found

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ ^C

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ ls -R
.:
README.md  main.tf    outputs.tf   terraform.tfvars
locals.tf  modules/  scripts/    variables.tf

./modules:
networking/  security/  webserver/

./modules/networking:
main.tf    outputs.tf   variables.tf

./modules/security:
main.tf    outputs.tf   variables.tf

./modules/webserver:
main.tf    outputs.tf   variables.tf

./scripts:
apache-setup.sh  nginx-setup.sh

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ |
```

assignment\_part1\_gitignore



```
# Assignment 2 – Advanced Terraform & Nginx Multi-Tier Architecture

## Project Structure

This project deploys a multi-tier web application using Terraform on AWS.

### Components:
- Networking (VPC, Subnet, IGW, Routing)
- Security Groups (Nginx & Backend)
- Web Servers (Apache)
- Reverse Proxy (Nginx)
- High Availability Setup

### Folder Structure

~
~
```

## 1.2 Variable Configuration (5 marks)

## assignment part1 variables tf

# Assignment 2 – Advanced Terraform & Nginx Multi-Tier Architecture

## ## Project Structure

This project deploys a multi-tier web application using Terraform on AWS.

### ### Components:

- Network Components:
    - Networking (VPC, Subnet, IGW, Routing)
    - Security Groups (Nginx & Backend)
    - Web Servers (Apache)
    - Reverse Proxy (Nginx)
    - High Availability Setup

### ### Folder Structure

卷之三

[README.md](#) [+] [\[unix\]](#) (14:56 27/12/2025)

:wq !

assignment\_part1\_terraform\_tfvars

```
vpc_cidr_block    = "10.0.0.0/16"
subnet_cidr_block = "10.0.10.0/24"
availability_zone = "me-central-1a"
env_prefix        = "prod"
instance_type     = "t3.micro"

public_key  = "~/.ssh/id_ed25519.pub"
private_key = "~/.ssh/id_ed25519"

backend_servers = [
  {
    name      = "web-1"
    script_path = "./scripts/apache-setup.sh"
  },
  {
    name      = "web-2"
    script_path = "./scripts/apache-setup.sh"
  },
  {
    name      = "web-3"
    script_path = "./scripts/apache-setup.sh"
  }
]

~~~
```

terraform.tfvars[+] [unix] (14:56 27/12/2025)  
-- INSERT --

### 1.3 Networking Module (5 marks)

assignment\_part1\_networking\_module\_main

```
MINGW64:/c/Users/Syed/Assignment2
resource "aws_vpc" "this" {
  cidr_block = var.vpc_cidr_block
  tags = {
    Name = "${var.env_prefix}-vpc"
  }
}

resource "aws_subnet" "this" {
  vpc_id           = aws_vpc.this.id
  cidr_block       = var.subnet_cidr_block
  availability_zone = var.availability_zone
  map_public_ip_on_launch = true
  tags = {
    Name = "${var.env_prefix}-subnet"
  }
}

resource "aws_internet_gateway" "this" {
  vpc_id = aws_vpc.this.id
  tags = {
    Name = "${var.env_prefix}-igw"
  }
}

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}-rt"
  }
}

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

modules/networking/main.tf[+1] [unix] (14:56 27/12/2025)

assignment\_part1\_networking\_module\_outputs

```
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
}

modules/networking/outputs.tf[+] [unix] (14:56 27/12/2025)
-- INSERT --
```

## 1.4 Security Module (5 marks)

assignment\_part1\_security\_module

```

resource "aws_security_group" "nginx_sg" {
  name      = "${var.env_prefix}-nginx-sg"
  description = "Nginx Security Group"
  vpc_id     = var.vpc_id

  ingress {
    from_port  = 22
    to_port    = 22
    protocol   = "tcp"
    cidr_blocks = [var.my_ip]
  }

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

  ingress {
    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"]
  }

  tags = {
    Name = "${var.env_prefix}-nginx-sg"
  }
}

resource "aws_security_group" "backend_sg" {
  name      = "${var.env_prefix}-backend-sg"
  vpc_id     = var.vpc_id

  ingress {
    from_port  = 22
    to_port    = 22
    protocol   = "tcp"
    cidr_blocks = [var.my_ip]
  }

  ingress {
    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"]
  }

  tags = {
    Name = "${var.env_prefix}-backend-sg"
  }
}

```

modules/security/main.tf[+] [unix] (14:56 27/12/2025)

```
MINGW64:/c/Users/Syed/Assignment2
```

```
variable "vpc_id" {}
variable "env_prefix" {}
variable "my_ip" {}
```

```
|
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
output "nginx_sg_id" {
  value = aws_security_group.nginx_sg.id
}

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

~
```

## 1.5 Locals Configuration (5 marks)

assignment part1 locals tf

## 2. Webserver Module (15 marks)

assignment part2 webserver module variables

assignment part2 webserver module main

assignment\_part2\_webserver\_module\_outputs

```
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
}
~
```

## 2.2 Module Usage (5 marks)

assignment\_part2\_main\_tf\_modules

### **3. Server Configuration Scripts (20 marks)**

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

assignment part3 apache script

```

#!/bin/bash
set -e

# Update system
yum update -y

# Install Apache
yum install httpd -y

# Start and enable Apache
systemctl start httpd
systemctl enable httpd

# Get metadata token (MD5-2)
TOKEN=$(curl -s -X PUT "http://169.254.169.254/latest/api/token" \
-H "X-amz-ecc-metadata-token: $TOKEN")

# Get instance metadata
PRIVATE_IP=$(curl -s -H "X-amz-ecc-metadata-token: $TOKEN" \
http://169.254.169.254/latest/meta-data/local-ipv4)
PUBLIC_IP=$(curl -s -H "X-amz-ecc-metadata-token: $TOKEN" \
http://169.254.169.254/latest/meta-data/public-ipv4)
PUBLIC_DNS=$(curl -s -H "X-amz-ecc-metadata-token: $TOKEN" \
http://169.254.169.254/latest/meta-data/public-hostname)
INSTANCE_ID=$(curl -s -H "X-amz-ecc-metadata-token: $TOKEN" \
http://169.254.169.254/latest/meta-data/instance-id)

# Set hostname
hostnamectl set-hostname myapp-webserver

# Create custom HTML page
cat >/var/www/html/index.html <<EOF
<!DOCTYPE html>
<html>
<head>
    <title>Backend Web Server</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 50px;
            background: linear-gradient(135deg, #667eea 0%, #768aa2 100%);
            color: white;
        }
        .container {
            background: rgba(255, 255, 255, 0.1);
            padding: 10px;
            border-radius: 10px;
            box-shadow: 0 1px 3px 0 rgba(31, 38, 135, 0.3);
        }
        h1 { color: #fff; text-shadow: 2px 2px 4px rgba(0,0,0,0.3); }
        .info { margin: 15px 0; padding: 10px; background: rgba(255, 255, 255, 0.2); border-radius: 5px; }
        .label { font-weight: bold; color: #ff7000; }
    </style>
</head>
<body>
    <div class="container">
        <div> Backend Web Server - Assignment 2</div>
        <div class="info"><span>Label</span><span>Hostname:</span><span>$hostname</span></div>
        <div class="info"><span>Label</span><span>Instance ID:</span><span>$INSTANCE_ID</span></div>
        <div class="info"><span>Label</span><span>Private IP:</span><span>$PRIVATE_IP</span></div>
        <div class="info"><span>Label</span><span>Public IP:</span><span>$PUBLIC_IP</span></div>
        <div class="info"><span>Label</span><span>Deployed:</span><span>$date</span></div>
        <div class="info"><span>Label</span><span>Status:</span><span>■ Active and Running</span></div>
        <div class="info"><span>Label</span><span>Managed By:</span><span>Terraform</span></div>
    </div>
</body>
</html>
EOF

# Set permissions
chmod 644 /var/www/html/index.html

echo "Apache setup completed successfully!">~/assignment_part3_apache_execution.log
~
~
~
~
~
apache-setup.sh[+]: [runix] (04:59 01/01/1970)
:wgj|

```

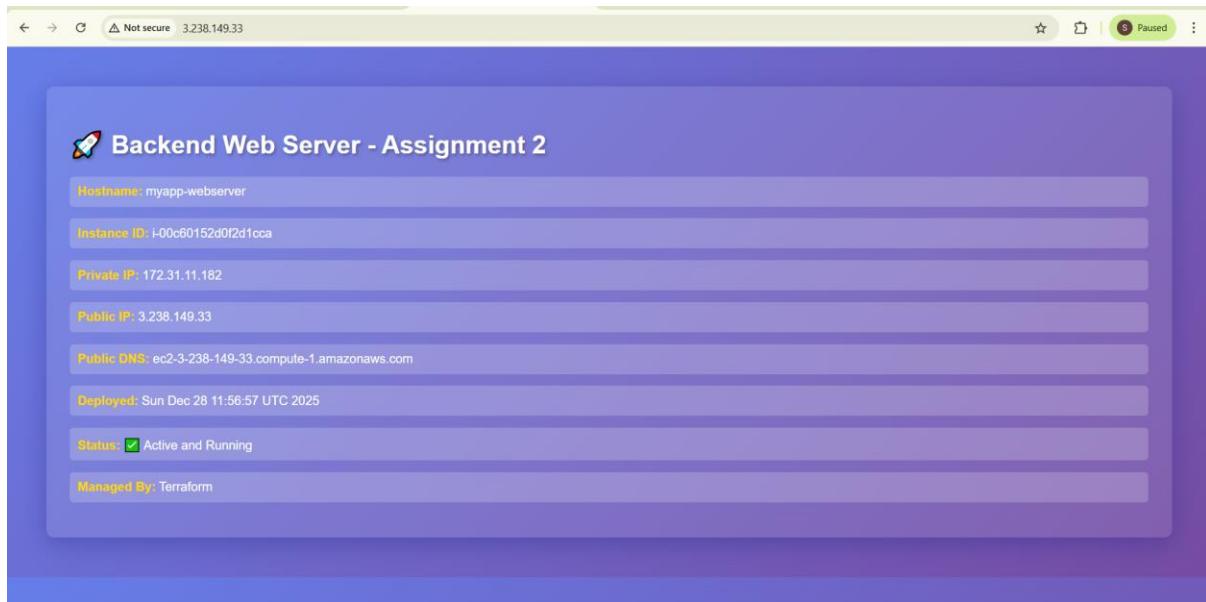
## assignment\_part3\_apache\_execution

```

[ec2-user@ip-172-31-11-182 scripts]$ sudo ./apache-setup.sh
Amazon Linux 2023 Kernel Livepatch repository
Dependencies resolved.
Nothing to do.
Completed.
Last metadata expiration check: 0:00:02 ago on Sun Dec 28 11:56:52 2025.
Dependencies resolved.
=====
Package                               Architecture   Version          Repository      Size
=====
Installing:
httpd                                x86_64        2.4.65-1.amzn2023.0.2
                                         1.7.5-1.amzn2023.0.4
                                         1.6.3-1.amzn2023.0.2
                                         1.6.3-1.amzn2023.0.2
                                         18.0.0-12.amzn2023.0.3
                                         2.4.65-1.amzn2023.0.2
                                         2.4.65-1.amzn2023.0.2
                                         2.4.65-1.amzn2023.0.2
                                         1.0.9-4.amzn2023.0.2
                                         2.1.49-3.amzn2023.0.3
                                         1.6.3-1.amzn2023.0.2
                                         2.0.27-1.amzn2023.0.3
                                         2.4.65-1.amzn2023.0.2
                                         1.8 MB/s | 47 kB   00:00
                                         535 kB/s | 15 kB   00:00
                                         1.8 MB/s | 47 kB   00:00
                                         650 kB/s | 19 kB   00:00
                                         487 kB/s | 13 kB   00:00
                                         33 MB/s | 1.4 kB   00:00
                                         2.0 MB/s | 81 kB   00:00
                                         32 MB/s | 315 kB   00:00
                                         1.2 MB/s | 33 kB   00:00
                                         4.8 MB/s | 166 kB  00:00
                                         2.1 MB/s | 60 kB   00:00
                                         32 MB/s | 2.4 MB  00:00
=====
Transaction Summary
Install 13 Packages
Total download size: 2.4 M
Installed size: 6.9 M
Downloaded packages:
(1/13): apr-1.7.5-1.amzn2023.0.4.x86_64.rpm
(2/13): apr-util-1.6.3-1.amzn2023.0.2.x86_64.rpm
(3/13): apr-util-lmdb-1.6.3-1.amzn2023.0.2.x86_64.rpm
(4/13): apr-util-openssl-1.6.3-1.amzn2023.0.2.x86_64.rpm
(5/13): httpd-2.4.65-1.amzn2023.0.2.x86_64.rpm
(6/13): generic-logos-httdp-18.0.0-12.amzn2023.0.3.noarch.rpm
(7/13): httpd-filesystem-1.0.9-4.amzn2023.0.2.noarch.rpm
(8/13): httpd-tools-2.4.65-1.amzn2023.0.2.x86_64.rpm
(9/13): httpd-tools-2.4.65-1.amzn2023.0.2.x86_64.rpm
(10/13): libbrotli-1.0.9-4.amzn2023.0.2.x86_64.rpm
(11/13): mailcap-2.1.49-3.amzn2023.0.3.noarch.rpm
(12/13): mod_http2-2.0.27-1.amzn2023.0.3.x86_64.rpm
(13/13): mod_lua-2.4.65-1.amzn2023.0.2.x86_64.rpm
=====
Total download size: 2.4 M
Downloaded packages:
3.3 MB/s | 129 kB   00:00
2.2 MB/s | 97 kB   00:00
296 kB/s | 13 kB   00:00
535 kB/s | 15 kB   00:00
1.8 MB/s | 47 kB   00:00
650 kB/s | 19 kB   00:00
487 kB/s | 13 kB   00:00
33 MB/s | 1.4 kB   00:00
2.0 MB/s | 81 kB   00:00
32 MB/s | 315 kB   00:00
1.2 MB/s | 33 kB   00:00
4.8 MB/s | 166 kB  00:00
2.1 MB/s | 60 kB   00:00
32 MB/s | 2.4 MB  00:00
=====

```

## assignment\_part3\_backend\_webpage



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

assignment\_part3\_nginx\_script

```
ec2-user@ip-172-31-11-182:~/scripts
proxy_cache my_cache;
proxy_cache_valid 200 60m;
proxy_cache_valid 404 10m;
proxy_cache_key "$scheme$request_method$host$request_uri";
proxy_cache_bypass $http_cache_control;
add_header X-Cache-Status $upstream_cache_status;

# Timeouts
proxy_connect_timeout 60s;
proxy_send_timeout 60s;
proxy_read_timeout 60s;
}

# Health check endpoint
location /health {
    access_log off;
    return 200 "Nginx is healthy\n";
    add_header Content-Type text/plain;
}
}

# HTTP Server (redirect to HTTPS)
server {
    listen 80;
    server_name _;

    location / {
        return 301 https://$host$request_uri;
    }

    # Allow health checks over HTTP
    location /health {
        access_log off;
        return 200 "Nginx is healthy\n";
        add_header Content-Type text/plain;
    }
}
EOF

# Create cache directory
mkdir -p /var/cache/nginx
chown -R nginx:nginx /var/cache/nginx

# Test and restart Nginx
nginx -t && systemctl restart nginx

echo "Nginx setup completed successfully!"
:wq!
```

assignment\_part3\_nginx\_default\_page



## 4. Infrastructure Deployment (15 marks)

### 4.1 Initial Deployment (5 marks)

assignment\_part4\_ssh\_keygen

```
Syed@DESKTOP-S50GK51 MINGW64 ~/Downloads (main)
$ cd ~/Assignment2

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ cd ~/Assignment2/terraform
bash: cd: /c/Users/Syed/Assignment2/terraform: No such file or directory

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ ls ~/.ssh/id_rsa.pub
ls: cannot access '/c/Users/Syed/.ssh/id_rsa.pub': No such file or directory

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ ssh-keygen -t rsa -b 4096 -f ~/.ssh/id_rsa -N ""
Generating public/private rsa key pair.
Your identification has been saved in /c/Users/Syed/.ssh/id_rsa
Your public key has been saved in /c/Users/Syed/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:YGIy1N8PMrHL2fmTp7ARvfEH6K/VuV36Ys1Eg6A13BY Syed@DESKTOP-S50GK51
The key's randomart image is:
+---[RSA 4096]---+
| .. . . E.
| . . = o
| o.o+o o + .
| +=ooo.. . o
| . * .+S . . .
| + oo.+ o ...
| o.o.o +o ..
| ++o... o=o
| . o= . oo.
+---[SHA256]---+
Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ |
```

assignment\_part4\_terraform\_init

```
Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ terraform init
Initializing the backend...
Initializing modules...
Initializing provider plugins...
- Finding latest version of hashicorp/http...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/http v3.5.0...
- Installed hashicorp/http v3.5.0 (signed by Hashicorp)
- Installing hashicorp/aws v6.27.0...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/http v3.5.0...
- Installed hashicorp/aws v6.27.0 (signed by Hashicorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

assignment\_part4\_terraform\_validate

```
Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ terraform validate
Success! The configuration is valid.
```

assignment\_part4\_terraform\_plan

```
cyberdeck:skt01-ss0901 MINGW64 ~/Assignment12 (main)
$ terraform plan
var.ami_id
  AMI ID for EC2 instances

  Enter a value:

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 0s [id=https://icanhazip.com]
data.aws_ssm_parameter.amzn2: Reading...
data.aws_ssm_parameter.amzn2: Read complete after 1s [id=/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2]

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

Terraform will perform the following actions:

# module.backend_servers["web-1"].aws_instance.backend[0] will be created
+ resource "aws_instance" "backend" {
    + ami                               = (sensitive value)
    + arn                               = (known after apply)
    + associate_public_ip_address       = (known after apply)
    + availability_zone                 = (known after apply)
    + disable_api_stop                  = (known after apply)
    + disable_api_termination          = (known after apply)
    + ebs_optimized                     = (known after apply)
    + enable_primary_ipv6              = (known after apply)
    + force_destroy                     = false
    + get_password_data                = false
    + host_id                           = (known after apply)
    + host_resource_group_arn           = (known after apply)
    + iam_instance_profile              = (known after apply)
    + id                                = (known after apply)
    + instance_initiated_shutdown_behavior = (known after apply)
    + instance_lifecycle               = (known after apply)
    + instance_state                   = (known after apply)
    + instance_type                    = "t3.micro"
    + ipv6_address_count               = (known after apply)
    + ipv6_addresses                   = (known after apply)
    + key_name                          = "key"
    + monitoring                        = (known after apply)
    + outpost_arn                      = (known after apply)
    + password_data                    = (known after apply)
    + placement_group                  = (known after apply)
    + placement_group_id               = (known after apply)
    + placement_partition_number        = (known after apply)
    + primary_network_interface_id     = (known after apply)
    + private_dns                       = (known after apply)
    + private_ip                        = (known after apply)
    + public_dns                        = (known after apply)
    + public_ip                         = (known after apply)
```

## assignment\_part4\_terraform\_apply

```
$ terraform apply -auto-approve
var.ami_id
  AMI ID for EC2 instances

  Enter a value:

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 0s [id=https://icanhazip.com]
data.aws_ssm_parameter.amzn2: Reading...
data.aws_ssm_parameter.amzn2: Read complete after 2s [id=/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2]

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

Terraform will perform the following actions:

# module.backend_servers["web-1"].aws_instance.backend[0] will be created
+ resource "aws_instance" "backend" {
    + ami                               = (sensitive value)
    + arn                             = (known after apply)
    + associate_public_ip_address      = (known after apply)
    + availability_zone                = (known after apply)
    + disable_api_stop                 = (known after apply)
    + disable_api_termination          = (known after apply)
    + ebs_optimized                    = (known after apply)
    + enable_primary_ipv6              = (known after apply)
    + force_destroy                    = false
    + get_password_data               = false
    + host_id                          = (known after apply)
    + host_resource_group_arn          = (known after apply)
    + iam_instance_profile             = (known after apply)
    + id                               = (known after apply)
    + instance_initiated_shutdown_behavior = (known after apply)
    + instance_lifecycle               = (known after apply)
    + instance_state                   = (known after apply)
    + instance_type                   = "t3.micro"
    + ipv6_address_count              = (known after apply)
    + ipv6_addresses                  = (known after apply)
    + key_name                        = "key"
    + monitoring                      = (known after apply)
    + outpost_arn                     = (known after apply)
    + password_data                  = (known after apply)
    + placement_group                 = (known after apply)
    + placement_group_id              = (known after apply)
    + placement_partition_number       = (known after apply)
```

#### **4.2 Output Configuration (5 marks)**

### assignment\_part4\_terraform\_output

```
Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ terraform output
backend_servers_info = {
  "web-1" = {
    "instance_id" = [
      "i-08601a87b94e93fe8",
      "i-01a374de43288ab32",
      "i-047ef7dc11056549e",
    ]
    "private_ip" = [
      "10.0.10.243",
      "10.0.10.87",
      "10.0.10.183",
    ]
    "public_ip" = [
      "44.195.69.216",
      "98.86.148.130",
      "3.231.204.61",
    ]
  }
  "web-2" = {
    "instance_id" = [
      "i-049dfb0d3a79a8744",
      "i-00ddb751f17464a08",
      "i-0bb6fbcc5dba72cdca",
    ]
    "private_ip" = [
      "10.0.10.33",
      "10.0.10.219",
      "10.0.10.253",
    ]
    "public_ip" = [
      "3.239.42.192",
      "34.236.243.157",
      "34.200.220.149",
    ]
  }
  "web-3" = {
    "instance_id" = [
      "i-0bfe5c3d42178e0c5",
      "i-0629846db90668695",
      "i-06f0bb724ced023a8",
    ]
    "private_ip" = [
      "10.0.10.89",
      "10.0.10.242",
      "10.0.10.73",
    ]
    "public_ip" = [
      "3.236.89.145",
      "18.207.111.59",
      "3.239.33.223",
    ]
  }
}
configuration_guide = <<EOT
=====

EOT
```

### assignment\_part4\_outputs\_json

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

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ terraform output -json > outputs.json

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ |
```

## 4.3 AWS Console Verification (5 marks)

assignment\_part4\_aws\_vpc

The screenshot shows the AWS VPC dashboard with the 'VPCs' tab selected. There are two VPC entries listed:

Name	VPC ID	State	Encryption c...	Encryption control ...	Block Public...
prod-vpc	vpc-0a0eee904a6c71e35	Available	-	-	Off
-	vpc-0cc0868d02cdd5863	Available	-	-	Off

A message at the bottom says "Select a VPC above".

assignment\_part4\_aws\_subnet

The screenshot shows a list of subnets:

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
-	subnet-0886484810d3475da	Available	vpc-0cc0868d02cdd5863	Off	172.31.48.0/20
-	subnet-0504c24c6622dec42	Available	vpc-0cc0868d02cdd5863	Off	172.31.64.0/20
prod-subnet	subnet-02f4b48141eb76975	Available	vpc-0a0eee904a6c71e35   prod...	Off	10.0.10.0/24
-	subnet-05c9ebf1230d6a267	Available	vpc-0cc0868d02cdd5863	Off	172.31.80.0/20
-	subnet-0ad57ddb11a1febc	Available	vpc-0cc0868d02cdd5863	Off	172.31.32.0/20
-	subnet-0db578dbff650c91c	Available	vpc-0cc0868d02cdd5863	Off	172.31.16.0/20

A message at the bottom says "Select a subnet".

The screenshot shows a list of subnets:

Availability Zone	Network border group	Route table	Network ACL	Default subnet	Auto
use1-az3 (us-east-1e)	us-east-1	rtb-0dd69f9eeaffa80ff	acl-07ed2b0e144d81492	Yes	Yes
use1-az5 (us-east-1f)	us-east-1	rtb-0dd69f9eeaffa80ff	acl-07ed2b0e144d81492	Yes	Yes
use1-az1 (us-east-1a)	us-east-1	rtb-0800fd06f8f572414   prod-rt	acl-00b3dcddf0b202a8d	No	Yes
use1-az2 (us-east-1b)	us-east-1	rtb-0dd69f9eeaffa80ff	acl-07ed2b0e144d81492	Yes	Yes
use1-az6 (us-east-1d)	us-east-1	rtb-0dd69f9eeaffa80ff	acl-07ed2b0e144d81492	Yes	Yes
use1-az4 (us-east-1c)	us-east-1	rtb-0dd69f9eeaffa80ff	acl-07ed2b0e144d81492	Yes	Yes

A message at the bottom says "Select a subnet".

assignment\_part4\_aws\_security\_groups

**Internet gateways (2) Info**

Name	Internet gateway ID	State	VPC ID	Owner
-	igw-01c4c2289ed245c83	Attached	vpc-0cc0868d02cdd5863	075006647027
prod-igw	igw-0a43d3afcf54d02f4	Attached	vpc-0a0eee904a6c71e35   prod-vpc	075006647027

Select an internet gateway above

**Route tables (3) Info**

Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC
-	rtb-0bc20969e07ce9c66	-	-	Yes	vpc-0a0eee904a6c71e35   prod.
prod-rt	rtb-0800fd06f8f572414	subnet-02f4b48141eb76...	-	No	vpc-0a0eee904a6c71e35   prod.
-	rtb-0dd69f9eeaffa80ff	-	-	Yes	vpc-0cc0868d02cdd5863

Select a route table

**Security Groups (1/5) Info**

Name	Security group ID	Security group name	VPC ID	Description
prod-nginx-sg	sg-0267b19413adbc560	prod-nginx-sg	vpc-0a0eee904a6c71e35	Nginx Security Grou
-	sg-03dd3730a272d2ff9	default	vpc-0cc0868d02cdd5863	default VPC security
-	sg-073aaa788e380ac56	launch-wizard-3	vpc-0cc0868d02cdd5863	launch-wizard-3 cre
-	sg-00b8bc183e8784831	default	vpc-0a0eee904a6c71e35	default VPC security
prod-backend-sg	sg-0cb181188a15a1c6d	prod-backend-sg	vpc-0a0eee904a6c71e35	Managed by Terrafo

sg-0267b19413adbc560 - prod-nginx-sg

**Inbound rules (3)**

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-0b382f31c727bb48f	IPv4	HTTP	TCP	80
-	sgr-091e065748d3761d1	IPv4	SSH	TCP	22
-	sgr-0687263f3cb231189	IPv4	HTTPS	TCP	443

**Security Groups (1/5) Info**

Name	Security group ID	Security group name	VPC ID	Description
prod-nginx-sg	sg-0267b19413adbc560	prod-nginx-sg	vpc-0a0eee904a6c71e35	Nginx Security Grou
-	sg-03dd3730a272d2ff9	default	vpc-0cc0868d02cdd5863	default VPC security
-	sg-073aaa788e380ac56	launch-wizard-3	vpc-0cc0868d02cdd5863	launch-wizard-3 cre
-	sg-00b8bc183e8784831	default	vpc-0a0eee904a6c71e35	default VPC security
prod-backend-sg	sg-0cb181188a15a1c6d	prod-backend-sg	vpc-0a0eee904a6c71e35	Managed by Terrafo

sg-0cb181188a15a1c6d - prod-backend-sg

**Inbound rules (2)**

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-00a10722944d17f4c	IPv4	SSH	TCP	22
-	sgr-048f294f25555208b	-	HTTP	TCP	80

## assignment\_part4\_aws\_instances

**Instances (1/11) Info**

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
EC2	i-00c60152d0f2d1cca	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	ec2-3-238-1
<b>web-2</b>	i-00ddb751f17464a08	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	-
web-1	i-08601a87b94e93fe8	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	-
web-2	i-0629846db90668695	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	-
web-1	i-049dfb0d3a79a8744	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	-

**i-00ddb751f17464a08 (web-2)**

- Details**
- Status and alarms
- Monitoring
- Security
- Networking
- Storage
- Tags

**Instance summary**

Instance ID i-00ddb751f17464a08	Public IPv4 address 34.236.243.157   open address ↗	Private IPv4 addresses 10.0.10.219
IPv6 address -	Instance state Running	Public DNS -
Hostname type IP name: ip-10-0-10-219.ec2.internal	Private IP DNS name (IPv4 only) ip-10-0-10-219.ec2.internal	

**EC2**

- Dashboard
- EC2 Global View ↗
- Events
- Instances**
  - Instances**
  - Instance Types
  - Launch Templates
  - Spot Requests
  - Savings Plans
  - Reserved Instances
  - Dedicated Hosts
  - Capacity Reservations
  - Capacity Manager New
- Images**
  - AMIs
  - AMI Catalog
- Elastic Block Store**
  - Volumes
  - Snapshots

**Instances (1/11) Info**

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
EC2	i-00c60152d0f2d1cca	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	ec2-3-238-1
web-2	i-00ddb751f17464a08	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	-
<b>web-1</b>	i-08601a87b94e93fe8	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	-
web-2	i-0629846db90668695	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	-
web-1	i-049dfb0d3a79a8744	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	-

**i-08601a87b94e93fe8 (web-1)**

- Details**
- Status and alarms
- Monitoring
- Security
- Networking
- Storage
- Tags

**Instance summary**

Instance ID i-08601a87b94e93fe8	Public IPv4 address 44.195.69.216   open address ↗	Private IPv4 addresses 10.0.10.243
IPv6 address -	Instance state Running	Public DNS -
Hostname type IP name: ip-10-0-10-243.ec2.internal	Private IP DNS name (IPv4 only) ip-10-0-10-243.ec2.internal	

The image contains two side-by-side screenshots of the AWS EC2 Instances page. Both screenshots show a list of instances with the following details:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
EC2	i-00c60152d0f2d1cca	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	ec2-3-238-1
web-2	i-00ddb751f17464a08	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	-
web-1	i-08601a87b94e93fe8	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	-
<b>web-2</b>	<b>i-0629846db90668695</b>	<b>Running</b>	<b>t3.micro</b>	<b>3/3 checks passed</b>	<b>View alarms +</b>	<b>us-east-1a</b>	<b>-</b>
web-1	i-049dfb0d3a79a8744	Running	t3.micro	3/3 checks passed	View alarms +	us-east-1a	-

The second screenshot shows the same list but with instance **web-1** selected instead of **web-2**. Both screenshots also show detailed information for the selected instance, including its Public IP address (18.207.111.59), Instance state (Running), and Hostname type (IP name: ip-10-0-10-242.ec2.internal).

## 5. Nginx Configuration & Testing (25 marks)

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

assignment\_part5\_ssh\_nginx

```
Syed@DESKTOP-S50GK51 MINGW64 ~ (main)
$ cd ~/Assignment2

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ $ ssh ec2-user@18.232.146.156
bash: $: command not found

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ ssh ec2-user@18.232.146.156
ssh: connect to host 18.232.146.156 port 22: Connection timed out

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ ssh ec2-user@18.232.146.156
The authenticity of host '18.232.146.156' (18.232.146.156) can't be established.
ED25519 key fingerprint is SHA256:8sE31yR2EZOKzFBNM7lRv1czRoMznVAhA9grES0j6fI.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '18.232.146.156' (ED25519) to the list of known hosts
.
ec2-user@18.232.146.156: Permission denied (publickey,gssapi-keyex,gssapi-with-mic).

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ cd /c/Users/Syed/Downloads

Syed@DESKTOP-S50GK51 MINGW64 ~/Downloads (main)
$ ls
'3716895_3717020_new.pdf'
'52b356b3c4412c6e2c77554bd53bb24e.jpg'
'682b604d466185e1a2368aadd85c09cc.jpg'
'AI_Assignment_Shumail (1).ipynb'
'AI_Assignment_Shumail.ipynb'
'ANN.pdf'
'AWSCLIV2.msi'
'Acknowledgement for EE Personal Information Sharing.docx'
'Admin_credentials.csv'
'AnyDesk.exe'
'Assignment # 1 (1).pdf'
'Assignment-1.zip'
'Assignment1.md'
'CONFLUENT HEALTH EMPLOYEE FOUNDATION.pdf'
'ChatGPT Image Oct 13, 2025, 09_45_35 AM.png'
'Confluent - Web flyer.pdf'
'CyberGhostVPNSetup.exe'
'DAA_Lec_04_BSE_V (1).pptx'
'Documents/'
'EFI/'
'Form_I-9_08-08-2023.pdf'
'Healthcare+and+Insurance+Systems (1).pdf'
'Healthcare+and+Insurance+Systems.pdf'
'Lab3.md'
```

assignment part5 nginx conf updated

```
# For more information on configuration, see:  
#   * official English Documentation: http://nginx.org/en/docs/  
#   * official Russian Documentation: http://nginx.org/ru/docs/  
  
user nginx;  
worker_processes auto;  
error_log /var/log/nginx/error.log;  
pid /run/nginx.pid;  
  
# Load dynamic modules. See /usr/share/doc/nginx/README.dynamic.  
include /usr/share/nginx/modules/*.conf;  
  
events {  
    worker_connections 1024;  
}  
  
http {  
    upstream backend_servers {  
        server 10.0.10.89:80;  
        server 10.0.10.87:80;  
        server 10.0.10.253:80 backup;  
    }  
    log_format main '$remote_addr - $remote_user [$time_local] "$request" '  
                  '$status $body_bytes_sent "'.$http_referer'" '  
                  '"$http_user_agent" "$http_x_forwarded_for"';  
  
    access_log /var/log/nginx/access.log main;  
  
    sendfile          on;  
    tcp_nopush       on;  
    tcp_nodeLAY     on;  
    keepalive_timeout 65;  
    types_hash_max_size 4096;  
  
    include           /etc/nginx/mime.types;  
    default_type      application/octet-stream;  
  
    # Load modular configuration files from the /etc/nginx/conf.d directory.  
    # See http://nginx.org/en/docs/ngx_core_module.html#include  
    # for more information.  
    include /etc/nginx/conf.d/*.conf;  
  
    server {  
        listen      80;  
        listen      [::]:80;  
        server_name _;  
        root       /usr/share/nginx/html;  
  
        # Load configuration files for the default server block.  
        include /etc/nginx/default.d/*.conf;  
    }
```

## assignment\_part5\_nginx\_test

```
[ec2-user@ip-10-0-10-81 ~]$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

assignment part5 nginx restart

```
[ec2-user@ip-10-0-10-81 ~]$ sudo systemctl restart nginx
[ec2-user@ip-10-0-10-81 ~]$ 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 Mon 2025-12-29 08:32:10 UTC; 51s ago
     Process: 2301 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
    Process: 2298 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
    Process: 2296 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
  Main PID: 2304 (nginx)
     CGroup: /system.slice/nginx.service
           ├─2304 nginx: master process /usr/sbin/nginx
           ├─2306 nginx: worker process
           ├─2307 nginx: worker process

Dec 29 08:32:10 ip-10-0-10-81.ec2.internal systemd[1]: Starting The nginx HTTP and reverse proxy server...
Dec 29 08:32:10 ip-10-0-10-81.ec2.internal nginx[2298]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Dec 29 08:32:10 ip-10-0-10-81.ec2.internal nginx[2298]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Dec 29 08:32:10 ip-10-0-10-81.ec2.internal systemd[1]: Started The nginx HTTP and reverse proxy server.
[ec2-user@ip-10-0-10-81 ~]$ |
```

### **5.2 Test Load Balancing (5 marks)**

## assignment\_part5\_ssl\_warning

This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page, it means that the Apache HTTP server installed at this site is working properly.

**If you are a member of the general public:**

The fact that you are seeing this page indicates that the website you just visited is either experiencing problems, or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

**If you are the website administrator:**

You may now add content to the directory /var/www/html/. Note that until you do so, people visiting your website will see this page, and not your content. To prevent this page from ever being used, follow the instructions in the file /etc/httpd/conf.d/welcome.conf.

You are free to use the image below on web sites powered by the Apache HTTP Server:

## assignment\_part5\_web1\_response

This is content served by web-1.

## Web-1 Server

This is content served by web-1.

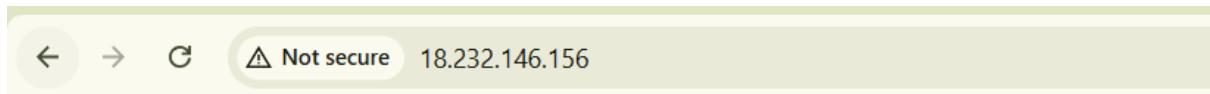
## assignment\_part5\_web2\_response

This is content served by web-2.

## Web-2 Server

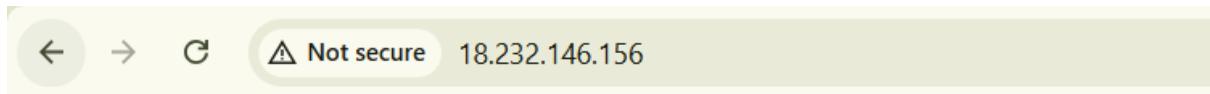
This is content served by web-2.

## assignment\_part5\_load\_balancing\_demo



## Web-2 Server

This is content served by web-2.



## Web-1 Server

This is content served by web-1.

### 5.3 Test Cache Functionality (5 marks)

assignment\_part5\_cache\_miss

Name	X Headers	Preview	Response	Initiator	Timing
18.232.146.156	Referrer Policy				strict-origin-when-cross-origin
▼ Response Headers		<input type="checkbox"/> Raw			
		Accept-Ranges	bytes		
		Connection	keep-alive		
		Content-Length	62		
		Content-Type	text/html; charset=UTF-8		
		Date	Mon, 29 Dec 2025 10:20:37 GMT		
		Etag	"3e-64713b8096ecd"		
		Last-Modified	Mon, 29 Dec 2025 09:17:57 GMT		
		Server	nginx/1.28.0		
		Upgrade	h2,h2c		
		X-Cache-Status	BYPASS		

### assignment\_part5\_cache\_hit

This is content served by web-1.

Name	X Headers	Preview	Response	Initiator	Timing
18.232.146.156	Status Code Remote Address Referrer Policy			JUU.UK	18.232.146.156:80 strict-origin-when-cross-origin
▼ Response Headers		<input type="checkbox"/> Raw			
		Accept-Ranges	bytes		
		Connection	keep-alive		
		Content-Length	62		
		Content-Type	text/html; charset=UTF-8		
		Date	Mon, 29 Dec 2025 10:13:43 GMT		
		Etag	"3e-64713b8096ecd"		
		Last-Modified	Mon, 29 Dec 2025 09:17:57 GMT		
		Server	nginx/1.28.0		
		Upgrade	h2,h2c		
		X-Cache-Status	HIT		

### assignment\_part5\_cache\_directory

```
[ec2-user@ip-10-0-10-81 ~]$ sudo ls -la /var/cache/nginx/
total 0
drwx----- 2 nginx root 6 Dec 29 09:55 .
drwxr-xr-x 7 root root 76 Dec 29 09:55 ..
[ec2-user@ip-10-0-10-81 ~]$
[ec2-user@ip-10-0-10-81 ~]$ |
```

### assignment\_part5\_access\_log\_cache

```
drwxr-xr-x 7 root root 76 Dec 29 09:55 ..
drwx----- 3 nginx nginx 16 Dec 29 10:12 f
[ec2-user@ip-10-0-10-81 ~]$ sudo systemctl stop nginx
[ec2-user@ip-10-0-10-81 ~]$ sudo rm -rf /var/cache/nginx/*
[ec2-user@ip-10-0-10-81 ~]$ sudo systemctl start nginx
[ec2-user@ip-10-0-10-81 ~]$
[ec2-user@ip-10-0-10-81 ~]$ sudo grep "Cache:" /var/log/nginx/access.log | tail -20
[ec2-user@ip-10-0-10-81 ~]$ |
```

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

### assignment\_part5\_web1\_stopped

```
$ ssh -i key.pem ec2-user@3.236.89.145
Last login: Mon Dec 29 09:17:16 2025 from 59.103.199.30
.      #
~\_ ####_      Amazon Linux 2
~~ \####\ AL2 End of Life is 2026-06-30.
~~  \#/ ,-->
~~ .-. / 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-10-0-10-89 ~]$ sudo systemctl stop httpd

[ec2-user@ip-10-0-10-89 ~]$
[ec2-user@ip-10-0-10-89 ~]$ sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
  Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
  Active: inactive (dead) since Mon 2025-12-29 10:24:10 UTC; 6s ago
    Docs: man:httpd.service(8)
 Process: 2145 ExecStart=/usr/sbin/httpd $OPTIONS -DFOREGROUND (code=exited, status=0/SUCCESS)
 Main PID: 2145 (code=exited, status=0/SUCCESS)
  Status: "Total requests: 27; Idle/Busy workers 100/0;Requests/sec: 0.0047; Bytes served/sec: 8 B/sec"

Dec 29 08:48:15 ip-10-0-10-89.ec2.internal systemd[1]: Starting The Apache HTTP Server...
Dec 29 08:48:15 ip-10-0-10-89.ec2.internal systemd[1]: Started The Apache HTTP Server.
Dec 29 10:24:09 ip-10-0-10-89.ec2.internal systemd[1]: Stopping The Apache HTTP Server...
Dec 29 10:24:10 ip-10-0-10-89.ec2.internal systemd[1]: Stopped The Apache HTTP Server.
[ec2-user@ip-10-0-10-89 ~]$ |
```

### assignment\_part5\_web2\_stopped

```
SYNUSDESKTOP-5DUGK51 MINGW04 ~/Downloads (math)
$ ssh -i key.pem ec2-user@98.86.148.130
Last login: Mon Dec 29 09:18:41 2025 from 59.103.199.30
.      #
~\_ ####_      Amazon Linux 2
~~ \####\ AL2 End of Life is 2026-06-30.
~~  \#/ ,-->
~~ .-. / 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-10-0-10-87 ~]$ sudo systemctl stop httpd
[ec2-user@ip-10-0-10-87 ~]$ sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
  Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
  Active: inactive (dead) since Mon 2025-12-29 10:26:10 UTC; 8s ago
    Docs: man:httpd.service(8)
 Process: 2168 ExecStart=/usr/sbin/httpd $OPTIONS -DFOREGROUND (code=exited, status=0/SUCCESS)
 Main PID: 2168 (code=exited, status=0/SUCCESS)
  Status: "Total requests: 26; Idle/Busy workers 100/0;Requests/sec: 0.00453; Bytes served/sec: 8 B/sec"

Dec 29 08:50:19 ip-10-0-10-87.ec2.internal systemd[1]: Starting The Apache HTTP Server...
Dec 29 08:50:19 ip-10-0-10-87.ec2.internal systemd[1]: Started The Apache HTTP Server.
Dec 29 10:26:09 ip-10-0-10-87.ec2.internal systemd[1]: Stopping The Apache HTTP Server...
Dec 29 10:26:10 ip-10-0-10-87.ec2.internal systemd[1]: Stopped The Apache HTTP Server.
[ec2-user@ip-10-0-10-87 ~]$ |
```

### assignment\_part5\_backup\_activated



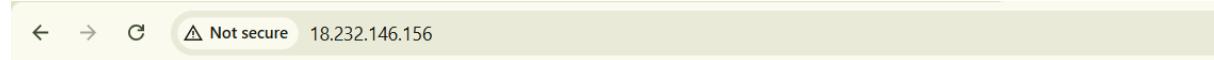
## Web-3 Server

This is content served by web-3 (backup only)

## assignment\_part5\_nginx\_error\_log

```
[ec2-user@ip-10-0-10-81 ~]$ sudo tail -f /var/log/nginx/error.log
2025/12/29 10:12:22 [warn] 2924#2924: conflicting server name "www" on 0.0.0.80, ignored
2025/12/29 10:16:10 [warn] 2952#2952: conflicting server name "www" on 0.0.0.80, ignored
2025/12/29 10:16:10 [warn] 2952#2952: conflicting server name "www" on 0.0.0.80, ignored
2025/12/29 10:16:10 [warn] 2950#2950: conflicting server name "www" on 0.0.0.80, ignored
2025/12/29 10:20:11 [warn] 2991#2991: conflicting server name "www" on 0.0.0.80, ignored
2025/12/29 10:20:11 [warn] 2991#2991: conflicting server name "www" on 0.0.0.80, ignored
2025/12/29 10:20:19 [notice] 2997#2997: signal process started
2025/12/29 10:25:25 [error] 2998#2998: *16 connect() failed (111: Connection refused) while connecting to upstream, client: 59.103.199.30, server: ..., request: "GET / HTTP/1.1", upstream: "http://10.0.10.89:80/", host: "18.232.146.156"
2025/12/29 10:25:25 [error] 2998#2998: *16 connect() failed (111: Connection refused) while connecting to upstream, client: 59.103.199.30, server: ..., request: "GET / HTTP/1.1", upstream: "http://10.0.10.87:80/", host: "18.232.146.156"
2025/12/29 10:26:53 [error] 2998#2998: *16 connect() failed (111: Connection refused) while connecting to upstream, client: 59.103.199.30, server: ..., request: "GET / HTTP/1.1", upstream: "http://10.0.10.89:80/", host: "18.232.146.156"
```

## assignment\_part5\_services\_restored



## Web-1 Server

This is content served by web-1.



## Web-2 Server

This is content served by web-2.

### **5.5 Security & Performance Analysis (5 marks)**

## assignment part5 ssl certificate

```
syed@DESKTOP-S50GK51 MINGW64 ~/Downloads (main)
$ openssl s_client -connect 18.232.146.156:443 -showcerts
Connecting to 18.232.146.156
CONNECTED(00000158)
Can't use SSL_get_servername
depth=0 C=PK, L=Karachi, O=MyCompany, OU=IT, CN=18.232.146.156
verify error:num=18:self-signed certificate
verify return:1
depth=0 C=PK, L=Karachi, O=MyCompany, OU=IT, CN=18.232.146.156
verify return:1
---
Certificate chain
  0 s:C=PK, L=Karachi, O=MyCompany, OU=IT, CN=18.232.146.156
    i:C=PK, L=Karachi, O=MyCompany, OU=IT, CN=18.232.146.156
      a:PKEY: RSA, 2048 (bit); sigalg: sha256WithRSAEncryption
      v:NotBefore: Dec 29 10:36:55 2025 GMT; NotAfter: Dec 29 10:36:55 2026 GMT
-----BEGIN CERTIFICATE-----
MIIDHTCCAm2gAwIBAgIJAOXwGdw3Ht2xMA0GCSqGSIb3DQEBCwUAMfkcXzAjBgNV
BAYTA1BLMRAwDgYDVQHdAdLJYJhY2hpMRIweAYDVQKDA1NeUNvbxbhbnkxcZAJ
BgNVBAsMkA1URcwfQYDVQKDDA4x0c4YmZiUMTQ2LjE1NjaEfwoYhNTEmjkXMDM2
NTVFaFw0yNjEyMjkxMDM2NTVaMFkxCzAjBgNVBAYTA1BLMRAwDgYDVQHdAdLYJh
Y2hpMRIweAYDVQKDA1NeUNvbxbhbnkxcZAJBgNVBAsMkA1URcwfQYDVQKDDA4x
0c4YmZiUMTQ2LjE1NjCCASiwdQYJkOzIhvCNAQEBBQADggEPADCCAoCggEBAL7e
zRrZ8PK5DUQN95CCONADrJwrfMRV6Eby3zKBwsotX0fmobk5kqhYhhsJw5S20/k
0Q69FdY3sSMZNCw+3T+Bw3CxBg1EICwL7jsqv/KRGY3KIVDmdHCvD+btL6d+3
AhxZVGjIeBBI18pHjgib2Gkyga1m0JURgQAxWFEEgoypMwTrsFjryzFZhYQeHm
4egQt6kDNR0TzD4e8xGakgt21v/998ar1/9GkozdHx1axioIpozVkn53xxRYFIc
PgaFrcPc1lw8UGUFVRQHuia7e04ykVjy558EV2SGgnUsugZ3o5anhwbrsuxhfksoWI
SWxDYXodckn+Ae7qip8CAWEAAaQNME4whQYDVR00BYEFGkQr4o17UBwq9QKgao/
NusEpytmWB8GA1udIwQYMBaaFGkQr4o17ubwg9QKgao/NusEpytmMawG1UdeWqF
MAMBAf8wDQYJkOzIhvCNAQELBQADggEBACnj3yvc4ev71xqAw/s97IpB7j/N47Z3
wVEka05TDL8khnKEVkjzVoe5SuJyxcVdyAG2C3prakezy3xjz3unHHcaQ1jgr
Yop6IKIL1oxmsHdzr0yeTbY31ypBSzw3yzw0Aq1lVTATByXYzATaGAE3VtAoap
JmGAQmmzWIk1/3wVMryjyjWAHxt4CLRS9y2wsotq47U1+aF+m+ujVzm+E+2yT8qpZ
mQgyx95FVRsgFoah+nly1fxjg51Up3N01tsuqo/3tsx/x+8dar/ymxJ7E7B+s15
FIuz9gctKJ/Li6E1MygzxDJNgBpwMlrjhGhLyVSR1nhnowViC8t3r+k=
-----END CERTIFICATE-----
---
server certificate
subject=C=PK, L=Karachi, O=MyCompany, OU=IT, CN=18.232.146.156
issuer=C=PK, L=Karachi, O=MyCompany, OU=IT, CN=18.232.146.156
---
No client certificate CA names sent
Peer signing digest: SHA256
Peer signature type: rsa_pss_rsae_sha256
Peer Temp Key: X25519, 253 bits
---
SSL handshake has read 1461 bytes and written 1611 bytes
Verification error: self-signed certificate
---
New, TLSv1.3, Cipher is TLS_AES_256_GCM_SHA384
Protocol: TLSv1.3
Server public key is 2048 bit
This TLS version forbids renegotiation.
Compression: NONE
Expansion: NONE
No ALPN negotiated
Early data was not sent
Verify return code: 18 (self-signed certificate)
---
Post-Handshake New Session Ticket arrived:
SSL-Session:
  Protocol : TLSv1.3
```

```

PSK identity: None
PSK identity hint: None
SRP username: None
TLS session ticket lifetime hint: 300 (seconds)
TLS session ticket:
0000 - 0b 28 bb 29 8c 2c e2 8c-8f e2 b2 da d9 01 ef 24 .(.).....$2k+..FZ...b....
0010 - 32 6b 2b a2 f3 46 5a af-ff 1d 62 be e5 ec 16 1b %1.eB..of...
0020 - 60 25 ed 31 ac 65 42 e0-d8 6f 66 04 06 9d eb 88 <.u. "t8m....m.
0030 - 3c fd 55 9d db 22 74 42-6d 1b 83 11 7f a0 6d 14 .....he..../
0040 - 8d e7 f9 f4 b3 09 60 68-65 ff d3 e8 1c 1b 2f d4 ....~.j.Ap.CU...
0050 - 88 ad b1 7e 0e 93 6a ae-41 70 92 43 53 2f f2 98 ...V.m..H.$S...
0060 - de 10 9e d9 56 09 6d 92-3a 48 a2 24 53 ab 17 09 O.;....j.z$...
0070 - 4f d5 87 3b 83 cf 0a 9c-c1 6a 83 7a 24 1f d3 e1 .Y.....%
0080 - a5 59 e0 ca 7f 10 ee bc-11 0f c5 fb d6 fb 2a 9c ..E....a.....T
0090 - ab 98 bb 45 f1 be d2 61-f2 b1 cb 85 9d 97 b8 54 .{...y..<[... .
00a0 - bc 7b fa cd 98 79 e8 c5-3c 5b 03 b0 1e 20 7f fa 7gJ.D.C... \Lu.6
00b0 - 37 67 4a 90 44 cc 63 0f-e3 27 d4 5c 4c 75 95 36 >...o.T&....=,Q.
00c0 - 3e f1 0b 18 6f a6 54 26-09 c3 9c 84 3d c9 51 f1 ..f;....^..N...c
00d0 - e5 9a 66 3b b0 c7 ec 07-5e 82 a2 4e 13 f1 0e 63

Start Time: 1767004884
Timeout : 7200 (sec)
Verify return code: 18 (self-signed certificate)
Extended master secret: no
Max Early Data: 0
---
read R BLOCK
---
Post-Handshake New Session Ticket arrived:
SSL-Session:
Protocol : TLSv1.3
Cipher   : TLS_AES_256_GCM_SHA384
Session-ID: 695EE9030EA807B46E4EE0115DAD229C7F39862716B7F8FAE1771E9723269E65
Session-ID-ctx:
Resumption PSK: 8A612767C3454A163668C47DBC7606D3180175506A532FE7183FB16E8BD11B069808BC6F74A11C1E646ED9D8EA6E7D1F
PSK identity: None
PSK identity hint: None
SRP username: None
TLS session ticket lifetime hint: 300 (seconds)
TLS session ticket:
0000 - 0b 28 bb 29 8c 2c e2 8c-8f e2 b2 da d9 01 ef 24 .(.).....$2k+..FZ...b....
0010 - ad 4d 6a 93 46 1f 5b-9d 76 8f aa 53 0c bb 24 .....F [ v...S..$2
0020 - dd 5d c8 db 98 90 97 70-9d 62 39 55 0c 67 7b 4a .].....p.b9u.g{J
0030 - 0b e5 d2 cc 9d 99 c7 fa-8b a2 87 6b 58 80 7d 46 .....kx.}F
0040 - 4c 59 7a 60 22 52 17 18-46 1b 05 99 d4 0f 9d de LY2 ``R..F.....
0050 - 6e 69 37 93 2f 43 0d 60-50 28 6d dc c3 c8 52 73 n17./C. P(m...Rs
0060 - 4e e8 34 3f 03 16 30 4a-aa 7a 6a 9a cf de 43 96 N.4?..01.zj...C.
0070 - 51 4c 23 c5 0d 13 ab 4a-cf f7 13 c3 23 7e 30 5e QL#...J...#~0\.
0080 - e5 0e b4 24 ff 1b 00-34 6e 38 93 7f d6 90 9d ....$.4n8.....
0090 - 81 67 3d 80 d6 57 50 e8-6f aa 2f c3 ec 3b eb 56 .g=.WP.o./.;.V
00a0 - f1 f0 76 3d 35 0d cd 3d-92 d0 2a 40 3e e3 c6 a2 ..v=5.=.^@>...
00b0 - 4f 89 92 7e a3 e1 7a 72-f6 9c ae 5a 9d e5 10 b6 O..~..zr....Z...
00c0 - 3b b2 53 5d 92 eb de ee-79 98 28 56 3e e0 4c 8f ;.S]....y.(V>..L.
00d0 - 95 1b 10 fe d7 2f 39 e1-45 9c 53 68 0f fb 3f 66 ....;/9.E.Sh..?f

Start Time: 1767004884
Timeout : 7200 (sec)
Verify return code: 18 (self-signed certificate)
Extended master secret: no
Max Early Data: 0
---
read R BLOCK
closed

```

## assignment\_part5\_security\_headers

```

Syed@DESKTOP-S50GK51 MINGW64 ~/Downloads (main)
$ curl -I -k https://18.232.146.156
HTTP/1.1 200 OK
Server: nginx/1.28.0
Date: Mon, 29 Dec 2025 10:44:01 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 62
Connection: keep-alive
Upgrade: h2,h2c
Last-Modified: Mon, 29 Dec 2025 09:19:28 GMT
ETag: "3e-64713bd754fee"
X-Cache-Status: HIT
Accept-Ranges: bytes

```

## assignment\_part5\_http\_redirect

```

Syed@DESKTOP-S50GK51 MINGW64 ~/Downloads (main)
$ curl -I http://18.232.146.156
HTTP/1.1 301 Moved Permanently
Server: nginx/1.28.0
Date: Mon, 29 Dec 2025 10:44:12 GMT
Content-Type: text/html
Content-Length: 169
Connection: keep-alive
Location: https://18.232.146.156/

```

### assignment\_part5\_error\_log\_analysis

```

./m/* https://aws.amazon.com/linux/amazon-linux-2023/
[ec2-user@ip-10-0-10-81 ~]$ sudo tail -50 /var/log/nginx/error.log
2025/12/29 08:27:26 [emerg] 2206#2206: "upstream" directive is not allowed here in /etc/nginx/nginx.conf:16
2025/12/29 08:38:38 [warn] 2332#2332: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 08:38:49 [warn] 2344#2344: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 08:38:49 [warn] 2348#2348: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 08:41:04 [error] 2352#2352: *1 connect() failed (111: Connection refused) while connecting to upstream, client: 59.103.199.30, server: _, request: "GET / HTTP/1.1", upstream: "http://10.0.10.89:80/", host: "18.232.146.156"
2025/12/29 08:41:04 [error] 2352#2352: *1 connect() failed (111: Connection refused) while connecting to upstream, client: 59.103.199.30, server: _, request: "GET / HTTP/1.1", upstream: "http://10.0.10.87:80/", host: "18.232.146.156"
2025/12/29 08:41:04 [error] 2352#2352: *1 connect() failed (111: Connection refused) while connecting to upstream, client: 59.103.199.30, server: _, request: "GET / HTTP/1.1", upstream: "http://10.0.10.253:80/", host: "18.232.146.156"
2025/12/29 08:41:04 [error] 2352#2352: *1 no live upstreams while connecting to upstream, client: 59.103.199.30, server: _, request: "GET /favicon.ico HTTP/1.1", upstream: "http://backend_servers/favicon.ico", host: "18.232.146.156", referer: "http://18.232.146.156/"
2025/12/29 09:08:37 [warn] 2508#2508: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 09:08:49 [warn] 2521#2521: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 09:08:49 [warn] 2523#2523: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 09:21:44 [warn] 2613#2613: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 09:21:44 [notice] 2613#2613: signal process started
2025/12/29 09:55:00 [warn] 2761#2761: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 09:55:12 [warn] 2767#2767: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 09:55:12 [notice] 2767#2767: signal process started
2025/12/29 10:01:45 [warn] 2811#2811: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 10:01:54 [warn] 2817#2817: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 10:01:54 [notice] 2817#2817: signal process started
2025/12/29 10:12:08 [warn] 2913#2913: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 10:15:22 [warn] 2924#2924: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 10:15:22 [warn] 2927#2927: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 10:15:10 [warn] 2957#2957: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 10:16:10 [warn] 2960#2960: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 10:20:11 [warn] 2991#2991: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 10:20:19 [warn] 2997#2997: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 10:20:19 [notice] 2997#2997: signal process started
2025/12/29 10:25:28 [error] 2998#2998: *7 connect() failed (111: Connection refused) while connecting to upstream, client: 59.103.199.30, server: _, request: "GET / HTTP/1.1", upstream: "http://10.0.10.89:80/", host: "18.232.146.156"
2025/12/29 10:26:53 [error] 2998#2998: *16 connect() failed (111: Connection refused) while connecting to upstream, client: 59.103.199.30, server: _, request: "GET / HTTP/1.1", upstream: "http://10.0.10.87:80/", host: "18.232.146.156"
2025/12/29 10:26:53 [error] 2998#2998: *16 connect() failed (111: Connection refused) while connecting to upstream, client: 59.103.199.30, server: _, request: "GET / HTTP/1.1", upstream: "http://10.0.10.89:80/", host: "18.232.146.156"
2025/12/29 10:39:56 [warn] 3154#3154: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 10:40:05 [warn] 3174#3174: conflicting server name "" on 0.0.0.0:80, ignored
2025/12/29 10:40:05 [warn] 3177#3177: conflicting server name "" on 0.0.0.0:80, ignored
[ec2-user@ip-10-0-10-81 ~]$ 

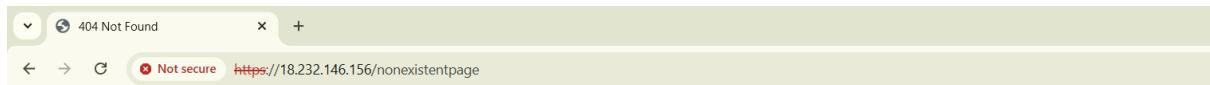
```

### assignment\_part5\_access\_log\_analysis

### **Bonus Tasks (10 marks extra credit)**

## Bonus 1: Custom Error Pages (3 marks)

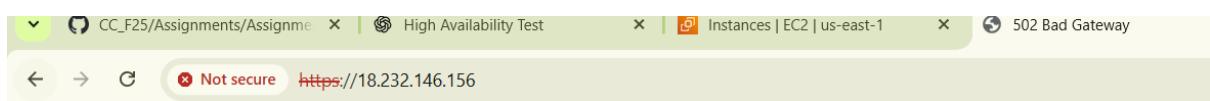
## bonus1\_custom\_404



## Not Found

The requested URL was not found on this server.

bonus1\_custom\_502



## 502

Bad Gateway! The server received an invalid response.

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

## bonus2\_rate\_limit\_config

```
ec2-user@ip-10-0-10-81:~$ cat bonus2_rate_limit_config
# Upstream backend servers
upstream backend_servers {
    server 10.0.10.89;
    server 10.0.10.87;
    server 10.0.10.253;
}

# Rate Limiting
limit_req_zone $binary_remote_addr zone=mylimit:10m rate=10r/s;

# HTTP server: redirects all traffic to HTTPS
server {
    listen 80;
    server_name _;

    # Redirect all HTTP requests to HTTPS
    return 301 https://$host$request_uri;
}

# HTTPS server: handles traffic, caching, and security headers
server {
    # Custom error pages
    error_page 404 /errors/404.html;
    error_page 502 /errors/502.html;
    error_page 503 /errors/503.html;

    location = /errors/404.html { internal; }
    location = /errors/502.html { internal; }
    location = /errors/503.html { internal; }

    listen 443 ssl;
    server_name _;

    # SSL certificate
    ssl_certificate /etc/ssl/certs/selfsigned.crt;
    ssl_certificate_key /etc/ssl/private/selfsigned.key;

    # Security headers
    add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;
    add_header X-Frame-Options SAMEORIGIN always;
    add_header X-Content-Type-Options nosniff always;
    add_header Referrer-Policy no-referrer-when-downgrade;
    add_header Content-Security-Policy "default-src 'self'" always;

    # Backend server with caching and rate limiting
    location / {
        limit_req zone=mylimit burst=20 nodelay;

        proxy_pass http://backend_servers;

        proxy_cache my_cache;
        proxy_cache_bypass $http_cache_control;
        proxy_cache_valid 200 60m;
        proxy_cache_use_stale error timeout updating;

        proxy_ignore_headers Cache-Control Expires Set-Cookie;
        proxy_hide_header Set-Cookie;

        add_header X-Cache-Status $upstream_cache_status always;
    }
}
~
~
```

## bonus2\_rate\_limit\_test

```
Sved@DESKTOP-S50GK51 MINGW64 ~/Downloads (main)
$ for i in {1..50}; do curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/ & done
wait
[1] 1498
[2] 1499
[3] 1500
[4] 1501
[5] 1502
[6] 1503
[7] 1504
[8] 1505
[9] 1506
[10] 1507
[11] 1508
[12] 1509
[13] 1510
[14] 1511
[15] 1512
[16] 1513
[17] 1514
[18] 1515
[19] 1516
[20] 1517
[21] 1518
[22] 1519
[23] 1520
[24] 1521
[25] 1522
[26] 1523
[27] 1524
[28] 1525
[29] 1526
[30] 1527
[31] 1528
[32] 1529
[33] 1530
200[34] 1531

200
[35] 1532
200
[36] 1533
200
[37] 1534
200
200
[38] 1535
[39] 1536
200
200
[40] 1537
200
[41] 1538
200
[42] 1539
200
[43] 1540
200
[44] 1541
200
200
[45] 1542
[46] 1543
200
[47] 1544
200
200
[48] 1545
200[49] 1546
```

```

503
200
503
503
200
[1] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[2] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[3] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[4] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[5] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[6] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[7] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[8] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[9] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[10] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[11] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[12] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[13] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[14] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[15] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[16] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[17] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[18] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[19] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[20] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[23] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[24] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[25] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[26] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[27] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[28] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[29] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[30] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[31] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[32] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[33] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[34] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[35] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[36] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[37] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[38] Done         curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
200
503
503
503
550033

200
503
503
503
[21] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[22] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[39] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[40] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[41] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[42] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
503[43] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[44] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[45] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/

503
200
200
[46] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[47] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[48] Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[49]- Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/
[50]+ Done          curl -k -s -o /dev/null -w "%{http_code}\n" https://18.232.146.156/

```

### Bonus 3: Health Check Automation (4 marks)

bonus3\_health\_check\_script

```
[ec2-user@ip-10-0-10-81 ~]$ ps aux | grep health_check
ec2-user 3995 0.0 0.2 119860 2828 pts/0    S   11:59  0:00 /bin/bash /home/ec2-user/health_check.sh
root     4089 0.0 0.7 239912 7528 pts/0    S   12:00  0:00 sudo nohup /home/ec2-user/health_check.sh
root     4091 0.0 0.2 119860 2852 pts/0    S   12:00  0:00 /bin/bash /home/ec2-user/health_check.sh
ec2-user 4353 0.0 0.1 119420 992 pts/0   R+  12:06  0:00 grep --color=auto health_check
[ec2-user@ip-10-0-10-81 ~]$
```

```
[ec2-user@ip-10-0-10-81 ~]$ cat /home/ec2-user/health_check.sh
#!/bin/bash

# Backend servers IPs
BACKENDS=("10.0.10.89" "10.0.10.87" "10.0.10.253")

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

# Infinite loop
while true; do
    DATE=$(date "+%Y-%m-%d %H:%M:%S")
    for SERVER in "${BACKENDS[@]}"; do
        # Check server HTTP response
        STATUS=$(curl -s -o /dev/null -w "%{http_code}" http://$SERVER)

        if [ "$STATUS" == "200" ]; then
            echo "$DATE - $SERVER is UP (HTTP $STATUS)" | tee -a $LOGFILE
        else
            echo "$DATE - $SERVER is DOWN (HTTP $STATUS)" | tee -a $LOGFILE
            # Optional: restart Apache on backend via SSH
            # ssh -i /home/ec2-user/key.pem ec2-user@$SERVER "sudo systemctl restart httpd"
        fi
    done
    # Wait 30 seconds before next check
    sleep 30
done
```

[bonus3\\_health\\_log](#)

```
[ec2-user@ip-10-0-10-81 ~]$ tail -50 /home/ec2-user/backend_health.log
2025-12-29 12:01:04 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:01:04 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:01:23 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:01:23 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:01:23 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:01:34 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:01:34 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:01:34 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:01:53 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:01:53 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:01:53 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:02:04 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:02:04 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:02:04 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:02:23 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:02:23 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:02:23 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:02:34 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:02:34 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:02:34 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:02:53 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:02:53 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:02:53 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:03:04 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:03:04 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:03:04 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:03:24 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:03:24 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:03:24 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:03:34 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:03:34 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:03:34 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:03:54 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:03:54 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:03:54 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:04:04 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:04:04 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:04:04 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:04:24 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:04:24 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:04:24 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:04:34 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:04:34 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:04:34 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:04:54 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:04:54 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:04:54 - 10.0.10.253 is DOWN (HTTP 000)
2025-12-29 12:05:04 - 10.0.10.89 is DOWN (HTTP 000)
2025-12-29 12:05:04 - 10.0.10.87 is DOWN (HTTP 000)
2025-12-29 12:05:04 - 10.0.10.253 is DOWN (HTTP 000)
[ec2-user@ip-10-0-10-81 ~]$ |
```

## 6. Documentation & Cleanup (10 marks)

### 6.1 README Documentation (5 marks)

## assignment\_part6\_readme

```
# Assignment 2 - Multi-Tier Web Infrastructure

## Project Overview
This project implements a multi-tier web infrastructure using AWS EC2 instances, Nginx load balancer, SSL/TLS, caching, and health checks. The architecture supports 3 backend servers with load balancing and failover.

## Architecture Diagram


```

graph TD
    Internet[Internet] -- "HTTPS (443) | HTTP (80)" --> Nginx[Nginx Server  
(Load Balancer)  
- SSL/TLS  
- Caching  
- Reverse Proxy]
    Nginx --> Web1[Web-1  
Primary]
    Nginx --> Web2[Web-2  
Primary]
    Nginx --> Web3[Web-3 (BKR)  
Backup]

```



## Components
- **Nginx**: Handles SSL termination, caching, load balancing, custom error pages, and rate limiting.
- **Web-1 & Web-2**: Primary backend Apache servers.
- **Web-3**: Backup server for failover.

## Prerequisites
- AWS account with appropriate IAM permissions.
- SSH key pair for EC2 access.
- Terraform installed.
- AWS CLI configured.

## Deployment Instructions
1. Configure 'terraform.tfvars' with AWS credentials, region, VPC, subnet, and EC2 instance details.
2. Initialize Terraform: 'terraform init'.
3. Plan the configuration: 'terraform plan'.
4. Apply the Terraform configuration: 'terraform apply' (Type 'yes' when prompted)
5. SSH into Nginx and backend servers: 'ssh -i key.pem ec2-user@nginx-ip'
   - ssh -i key.pem ec2-user@web-1
   - ssh -i key.pem ec2-user@web-2
   - ssh -i key.pem ec2-user@web-3

## Configuration Guide
1. **Update Backend IPs**: Modify '/etc/nginx/conf.d/default.conf' → 'upstream backend_servers { ... }'
2. **Nginx Configuration**:
   - Error handling: 'error_page 404 /errors/404.html;' etc.
   - SSL/TLS: '/etc/ssl/certs/selfsigned.crt'
   - Caching: 'proxy_cache' and 'keys_zone'
```

```

1. **Update Backend IPs**: Modify `/etc/nginx/conf.d/default.conf` to `upstream backend_servers { ... }`
2. **Nginx Explanation**:
   - Custom error pages: `error_page 404 /errors/404.html;` etc.
   - SSL/TLS: `/etc/ssl/certs/selfsigned.crt`
   - Caching: `proxy_cache` and `keys_zone`
   - Rate limiting: `limit_req_zone` and `limit_req`

## Testing Procedures
- Browser test: `https://<nginx-ip>` → check load balancing and caching headers (`X-Cache-Status`)
- 404 page: visit a non-existent URL
- 502 page: stop a backend server
- Rate limiting: run multiple rapid requests via `curl` or `ab`

## Architecture Details
1. **Network Topology**: VPC + Subnet → EC2 → Nginx → Backend
2. **Security Groups**:
   - Nginx SG: allow 22, 80, 443
   - Backend SG: allow only traffic from Nginx SG
3. **Load Balancing Strategy**: Round-robin across Web-1 and Web-2, Web-3 as backup

## Troubleshooting
1. **Common Issues**:
   - ERR_CERT_AUTHORITY_INVALID: self-signed SSL warning
   - 502 Bad Gateway: backend server stopped
2. **Log Locations**:
   - Nginx: `/var/log/nginx/error.log`
   - Apache: `/var/log/httpd/access_log` and `/var/log/httpd/error_log`
3. **Debug Commands**:
```bash
ps aux | grep nginx
sudo systemctl status httpd
sudo systemctl status nginx
curl -k -v https://<nginx-ip>
```
## 6.2 Infrastructure Cleanup (5 marks)

This section documents the destruction of all AWS resources created for Assignment 2.

### Cleanup Steps
1. Navigate to the project directory in Git Bash:
```cd ~/Assignment2```
2. Destroy all Terraform-managed resources:
```terraform destroy```
(Type 'yes' when prompted)
- Screenshot: `assignment_part6_terraform_destroy_prompt.png`
- Screenshot: `assignment_part6_terraform_destroy_complete.png`

### Verification Steps
3. Verify Terraform state file is empty:
```cat terraform.tfstate```
- Screenshot: `assignment_part6_empty_state.png`

### Troubleshooting
1. **Common Issues**:
   - ERR_CERT_AUTHORITY_INVALID: self-signed SSL warning
   - 502 Bad Gateway: backend server stopped
2. **Log Locations**:
   - Nginx: `/var/log/nginx/error.log`
   - Apache: `/var/log/httpd/access_log` and `/var/log/httpd/error_log`
3. **Debug Commands**:
```bash
ps aux | grep nginx
sudo systemctl status httpd
sudo systemctl status nginx
curl -k -v https://<nginx-ip>
```
## 6.2 Infrastructure Cleanup (5 marks)

This section documents the destruction of all AWS resources created for Assignment 2.

### Cleanup Steps
1. Navigate to the project directory in Git Bash:
```cd ~/Assignment2```
2. Destroy all Terraform-managed resources:
```terraform destroy```
(Type 'yes' when prompted)
- Screenshot: `assignment_part6_terraform_destroy_prompt.png`
- Screenshot: `assignment_part6_terraform_destroy_complete.png`

### Verification Steps
3. Verify Terraform state file is empty:
```cat terraform.tfstate```
- Screenshot: `assignment_part6_empty_state.png`
4. Confirm in AWS Console that no EC2 instances exist for Assignment-2.
- Screenshot: `assignment_part6_aws_instances_destroyed.png`

### Notes
- All commands were executed using **Git Bash**.
- No resources were deleted manually.
- Terraform successfully cleaned up all infrastructure.

### Deliverables
- Screenshot: `assignment_part6_readme.png` (README.md content)
- Screenshot: `assignment_part6_terraform_destroy_prompt.png`
- Screenshot: `assignment_part6_terraform_destroy_complete.png`
- Screenshot: `assignment_part6_aws_instances_destroyed.png`
- Screenshot: `assignment_part6_empty_state.png`

```

## 6.2 Infrastructure Cleanup (5 marks)

assignment\_part6\_terraform\_destroy\_prompt

```

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ 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]

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]

```

## assignment\_part6\_terraform\_destroy\_complete

```

module.backend_servers["web-1"].aws_instance.backend[2]: Destruction complete after 1m4s
module.networking.aws_internet_gateway.this: Still destroying... [id=igw-0a43d3aacf54d02f4, 00m40s elapsed]
module.backend_servers["web-1"].aws_instance.backend[1]: Still destroying... [id=i-01a374de43288ab32, 01m10s elas
module.backend_servers["web-3"].aws_instance.backend[2]: Still destroying... [id=i-06f0bb724ced023a8, 01m10s elas
module.backend_servers["web-3"].aws_instance.backend[0]: Still destroying... [id=i-0bfe5c3d42178e0c5, 01m10s elas
module.backend_servers["web-3"].aws_instance.backend[2]: Destruction complete after 1m12s
module.networking.aws_internet_gateway.this: Still destroying... [id=igw-0a43d3aacf54d02f4, 00m50s elapsed]
module.networking.aws_internet_gateway.this: Destruction complete after 52s
module.backend_servers["web-1"].aws_instance.backend[1]: Still destroying... [id=i-01a374de43288ab32, 01m20s elas
module.backend_servers["web-3"].aws_instance.backend[0]: Still destroying... [id=i-0bfe5c3d42178e0c5, 01m20s elas
module.backend_servers["web-3"].aws_instance.backend[0]: Destruction complete after 1m22s
module.backend_servers["web-1"].aws_instance.backend[1]: Destruction complete after 1m24s
module.networking.aws_subnet.this: Destroying... [id=subnet-02f4b48141eb76975]
module.security.aws_security_group.backend_sg: Destroying... [id=sg-0cb181188a15a1c6d]
module.networking.aws_subnet.this: Destruction complete after 1s
module.security.aws_security_group.backend_sg: Destruction complete after 1s
module.security.aws_security_group.nginx_sg: Destroying... [id=sg-0267b19413adbc560]
module.security.aws_security_group.nginx_sg: Destruction complete after 3s
module.networking.aws_vpc.this: Destroying... [id=vpc-0a0eee904a6c71e35]
module.networking.aws_vpc.this: Destruction complete after 5s

Destroy complete! Resources: 17 destroyed.

```

## assignment\_part6\_aws\_instances\_destroyed

Instances (1/11) [Info](#)

| Name                                    | Instance ID         | Instance state             | Instance type | Status check | Alarm status                  | Availability Zone | Public IPv4 |
|-----------------------------------------|---------------------|----------------------------|---------------|--------------|-------------------------------|-------------------|-------------|
| <input checked="" type="checkbox"/> EC2 | i-00c60152d0f2d1cca | <span>Shutting-down</span> | t3.micro      | -            | <a href="#">View alarms +</a> | us-east-1a        | ec2-3-238-1 |
| <input type="checkbox"/> web-2          | i-00dddb51f17464a08 | <span>Terminated</span>    | t3.micro      | -            | <a href="#">View alarms +</a> | us-east-1a        | -           |
| <input type="checkbox"/> web-1          | i-08601a87b94e93fe8 | <span>Terminated</span>    | t3.micro      | -            | <a href="#">View alarms +</a> | us-east-1a        | -           |
| <input type="checkbox"/> web-2          | i-0629846d09668695  | <span>Terminated</span>    | t3.micro      | -            | <a href="#">View alarms +</a> | us-east-1a        | -           |
| <input type="checkbox"/> web-1          | i-049dfb0d5a79a8744 | <span>Terminated</span>    | t3.micro      | -            | <a href="#">View alarms +</a> | us-east-1a        | -           |
| <input type="checkbox"/> nginx-server   | i-0824b01d8ce9cb1b6 | <span>Terminated</span>    | t3.micro      | -            | <a href="#">View alarms +</a> | us-east-1a        | -           |
| <input type="checkbox"/> web-1          | i-0bfef5c3d42178e05 | <span>Terminated</span>    | t3.micro      | -            | <a href="#">View alarms +</a> | us-east-1a        | -           |
| <input type="checkbox"/> web-2          | i-01a374de43288ab32 | <span>Terminated</span>    | t3.micro      | -            | <a href="#">View alarms +</a> | us-east-1a        | -           |
| <input type="checkbox"/> web-3          | i-0bb6fb5dba72cdca  | <span>Terminated</span>    | t3.micro      | -            | <a href="#">View alarms +</a> | us-east-1a        | -           |
| <input type="checkbox"/> web-3          | i-06f0bb724ced023a8 | <span>Terminated</span>    | t3.micro      | -            | <a href="#">View alarms +</a> | us-east-1a        | -           |

i-00c60152d0f2d1cca (EC2)

```
Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ aws ec2 describe-instances --filters "Name=instance-state-name,Values=running" --query "Reservations[].Instances[].InstanceId"
[]
```

assignment\_part6\_empty\_state

```
Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
$ cat terraform.tfstate
{
  "version": 4,
  "terraform_version": "1.14.3",
  "serial": 39,
  "lineage": "c5ab6f35-07cb-653b-0523-63c95f606fc6",
  "outputs": {},
  "resources": [],
  "check_results": [
    {
      "object_kind": "var",
      "config_addr": "var.vpc_cidr_block",
      "status": "unknown",
      "objects": null
    },
    {
      "object_kind": "var",
      "config_addr": "var.subnet_cidr_block",
      "status": "unknown",
      "objects": null
    }
  ]
}

Syed@DESKTOP-S50GK51 MINGW64 ~/Assignment2 (main)
```

## 4. Testing Results

Testing was carried out to assess the accuracy, stability, performance, and security of the implemented cloud-based web architecture. The primary goal of this phase was to verify that essential architectural objectives—such as efficient load balancing, effective caching, high availability, and secure system access—were successfully achieved. A

variety of functional and non-functional tests were executed on the deployed AWS environment to analyze system behavior under different operational scenarios. The testing activities were divided into five key categories: load balancing, cache performance, high availability, security, and performance evaluation. Each testing area is explained in detail below along with the corresponding findings and outcomes.

## **4.1 Load Balancing Tests**

The load balancing tests were conducted to confirm that incoming user requests were evenly distributed among the available backend servers. The Nginx server was configured to operate as a load balancer using a round-robin algorithm, allowing requests to be shared fairly between Web-1 and Web-2 servers. To validate this configuration, multiple HTTP requests were sent to the Nginx server's public IP address using both web browsers and command-line utilities. Each Apache backend server was set up to return a distinct identifier or hostname in its response, making it easy to determine which server handled each request. The observed results indicated that requests were processed alternately by Web-1 and Web-2, demonstrating successful load distribution. No backend server showed signs of being overloaded during normal request flow. This confirms that the load balancing mechanism effectively distributes traffic and minimizes the risk of performance degradation due to uneven server usage. Overall, the load balancing tests verified that the system supports horizontal scalability and enhances reliability by preventing reliance on a single backend server.

## **4.2 Cache Performance Tests**

Cache performance testing was performed to measure how effectively the Nginx caching mechanism improves response time and reduces the workload on backend servers. The cache was configured to store frequently accessed web resources at the Nginx level. Browser

developer tools were used to inspect HTTP response headers in order to monitor cache behavior. A custom header, X-Cache-Status, was used to identify whether a request was served from the cache. During the first request, the header displayed MISS, indicating that the content was retrieved from a backend server and stored in the cache. Subsequent requests showed HIT, confirming that cached content was delivered directly by Nginx. These results demonstrate that caching was properly configured and functioning as expected. Serving cached content significantly reduced response time and lowered backend server processing requirements, thereby improving overall system efficiency and traffic-handling capacity.

### **4.3 High Availability Tests**

High availability testing aimed to ensure uninterrupted system operation in the event of backend server failure. This is a crucial requirement for modern cloud-based applications that demand continuous service availability. To simulate server failure, one of the primary backend servers (either Web-1 or Web-2) was intentionally shut down while the system was active. During this period, continuous client requests were sent to the Nginx server. The results showed that Nginx automatically redirected traffic to the remaining active servers without service interruption. When both primary servers were unavailable, the backup server (Web-3) successfully processed incoming requests. This confirms that failover and backup routing mechanisms were correctly implemented. These findings demonstrate that the system architecture provides strong fault tolerance and maintains service availability even during server outages.

### **4.4 Security Tests**

Security testing was conducted to verify that proper access restrictions were enforced and that backend servers were protected from unauthorized public access. AWS Security Groups were utilized to

define and enforce network-level security rules. Multiple security checks were performed, including attempts to directly access backend Apache servers through their public IP addresses. These attempts were unsuccessful, confirming that backend servers were not exposed to the internet and only accepted traffic from the Nginx server. SSH access testing also confirmed that only authorized IP addresses were allowed to connect to EC2 instances, while unauthorized attempts were blocked. Furthermore, HTTP and HTTPS traffic was restricted to the Nginx server only, minimizing the system's attack surface. The results indicate that the security configuration adheres to the principle of least privilege and effectively safeguards the infrastructure against unauthorized access.

## 4.5 Performance Metrics

Performance evaluation focused on analyzing the effects of load balancing and caching on system responsiveness and resource utilization. Key factors such as response time, server workload, and request handling efficiency were observed during testing. The presence of caching resulted in noticeably faster response times for repeated requests compared to initial requests. Backend server load was reduced, as cached responses were delivered directly by the Nginx server. Load balancing ensured that traffic was evenly distributed, preventing any single server from becoming a bottleneck. Although automated benchmarking tools were not employed, manual observations clearly indicated improved system responsiveness and operational stability under normal traffic conditions. These results confirm that the architecture effectively enhances performance and scalability.

## 4.6 Summary of Testing Results

The testing phase verified that all critical architectural components operated as intended. Load balancing ensured balanced request distribution, caching enhanced performance efficiency, high

availability mechanisms provided fault tolerance, and security measures protected backend infrastructure. Performance observations indicated reduced response times and optimized server utilization. In conclusion, the testing results confirm that the deployed AWS-based architecture fulfills both functional and non-functional requirements, delivering a secure, scalable, and reliable cloud-based web application.

## **5. Challenges and Solution**

During the deployment of the multi-tier web infrastructure, several challenges were encountered that required careful troubleshooting. One major challenge was configuring the Nginx reverse proxy to correctly load balance traffic between the primary backend servers while keeping the backup server inactive until a failure occurred. Initial attempts resulted in occasional 502 Bad Gateway errors due to incorrect backend private IPs or improper security group rules. Another challenge involved implementing caching in Nginx, where cache misses persisted because of misconfigured cache paths and permissions. Additionally, generating and using self-signed SSL certificates created expected browser warnings, which needed to be clearly documented for evaluation purposes.

These challenges were successfully resolved through systematic solutions. The backend private IPs were dynamically retrieved from Terraform outputs and updated in the Nginx configuration, ensuring proper load balancing. Security group rules were carefully reviewed to allow traffic only from Nginx to backend servers while blocking all other access. The caching issue was resolved by creating the cache directory with correct ownership and verifying the proxy cache paths. Finally, the SSL warnings were acknowledged as expected behavior for self-signed certificates, and instructions were provided in the documentation for safely bypassing them during testing. These solutions reinforced practical skills in cloud deployment, debugging, and secure infrastructure management.

## **6. Conclusion**

This assignment successfully achieved the design, deployment, and validation of a multi-tier web infrastructure using Terraform and Nginx on Amazon Web Services (AWS). By applying Infrastructure as Code (IaC) principles, the

environment was provisioned in a modular, reusable, and secure manner. The implementation included an Nginx reverse proxy and load balancer with HTTPS, caching, and security headers, along with three Apache backend servers configured for load balancing and high availability using a backup server. Comprehensive testing verified proper load distribution, cache functionality, failover behavior, and secure communication. Overall, the project provided practical experience in cloud architecture design, DevOps automation, and performance optimization while meeting all academic and technical requirements.

## 7. Appendices

- The nginx-setup.sh script is included to configure the Nginx reverse proxy and load balancing server.
- Reusable web server modules are provided to create both the Nginx server and backend web servers in an efficient and modular manner.
- The apache-setup.sh script is included to automate the configuration of the backend Apache web servers
- Terraform configuration files such as main.tf, variables.tf, and locals.tf are included to define the complete infrastructure setup.

### Repository Link:

[https://github.com/23-22411-061-rgb/CC\\_23-22411-061-Assignment-2](https://github.com/23-22411-061-rgb/CC_23-22411-061-Assignment-2)