

**CLOUD COMPUTING**

**BSE ( V-B )**

Submitted By:

Musfira Farooq

Roll No:

2023-BSE-045

**ASSIGNMENT 02**

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

**Table of Contents**

[1. Executive Summary 3](#_Toc217940600)

[2. Architecture Design 3](#_Toc217940601)

[2.1 Architecture Overview 3](#_Toc217940602)

[2.2 Architecture Diagram 3](#_Toc217940603)

[2.3 Component Description 4](#_Toc217940604)

[2.4 Network Topology 4](#_Toc217940605)

[2.5 Security Design 4](#_Toc217940606)

[3. Implementation Details 4](#_Toc217940607)

[Part 1 : Infrastructure Setup 4](#_Toc217940608)

[Part 2 : Webserver Module 9](#_Toc217940609)

[Part 3 : Server Configuration Scripts 11](#_Toc217940610)

[Part 4 : Infrastructure Deployment 15](#_Toc217940611)

[Part 5 : Nginx Configuration & Testing 18](#_Toc217940612)

[Bonus Tasks 25](#_Toc217940613)

[Part 6 : Documentation & Cleanup 28](#_Toc217940614)

[4. Testing Results 31](#_Toc217940615)

[4.1 Load Balancing Validation 32](#_Toc217940616)

[4.2 Caching Efficiency 32](#_Toc217940617)

[4.3 High Availability Verification 32](#_Toc217940618)

[4.4 Security Verification 32](#_Toc217940619)

[5. Challenges & Solutions 32](#_Toc217940620)

[6. Conclusion 33](#_Toc217940621)

[7. Appendices 33](#_Toc217940622)

[Appendix A: Code Listings 33](#_Toc217940623)

[Appendix B: Configuration Files 33](#_Toc217940624)

[Appendix C: Additional Screenshots 33](#_Toc217940625)

# ****1. Executive Summary****

This assignment focuses on the design, deployment, testing, and cleanup of a **production-ready multi-tier web infrastructure** using **Terraform** and **Nginx** on **Amazon Web Services (AWS).** The main objective was to implement **Infrastructure as Code (IaC)** .

The infrastructure consists of a **custom VPC**, public subnets, security groups, EC2 instances running **Apache web servers**, and **Nginx proxy** acting as a load balancer. Terraform modules were used to ensure modularity, reusability, and clean project organization.

**Key features include:**

* Fully automated infrastructure provisioning using Terraform
* Modular Terraform design (networking, security, webserver)
* Nginx load balancing with SSL, caching, and security headers
* High availability testing with backend failover
* Proper infrastructure cleanup to avoid AWS charges

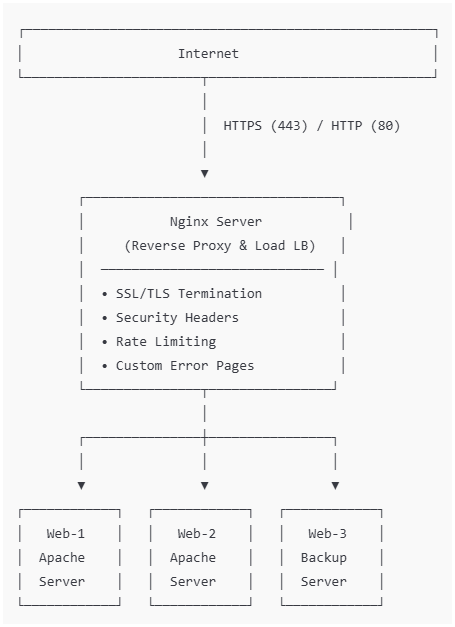
# ****2. Architecture Design****

## **2.1 Architecture Overview**

The architecture follows a **three-tier model:**

* **Client Layer:** End users accessing the application via browser
* **Load Balancer Layer:** Nginx reverse proxy handling traffic
* **Application Layer:** Multiple Apache web servers

### **2.2 Architecture Diagram**



### **2.3 Component Description**

**Nginx Server (Reverse Proxy & Load Balancer)**

The Nginx server handles all incoming user requests. It manages HTTP/HTTPS traffic, performs SSL/TLS termination, balances load across backend servers, applies caching and rate limiting, and displays custom error pages when needed.

**Web Servers (Web-1, Web-2, Web-3)**

The web servers run Apache and serve application content. Web-1 and Web-2 handle normal traffic, while Web-3 works as a backup server. All servers are monitored using health checks.

## **2.4 Network Topology**

The infrastructure uses a custom VPC with public subnets. An Internet Gateway enables internet access, route tables manage traffic flow, and security groups control network access.

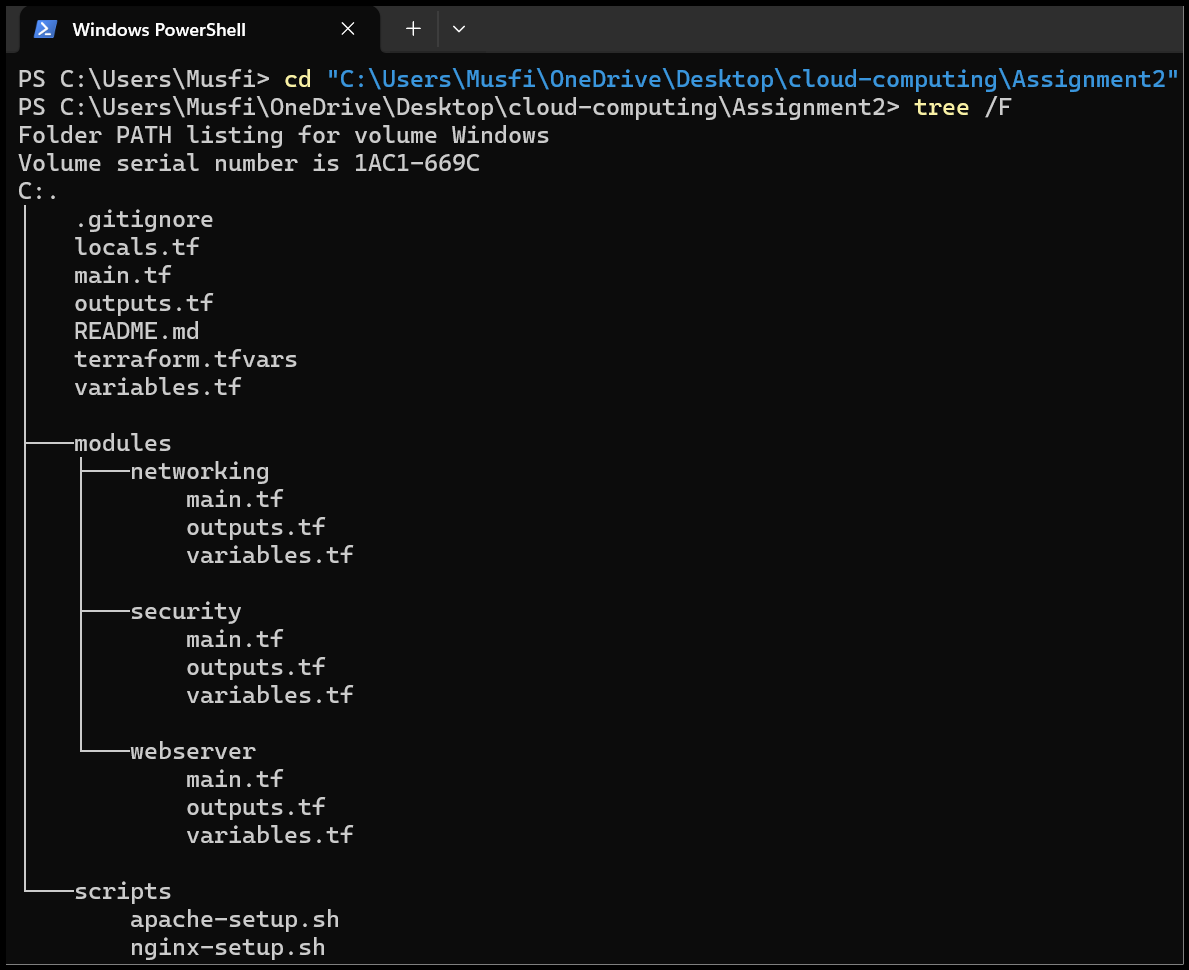
## **2.5 Security Design**

Security groups restrict access to required ports only. SSH access is limited, backend servers are accessible only through Nginx, and sensitive files are protected using .gitignore.

# 3. Implementation Details

## **Part 1 : Infrastructure Setup**

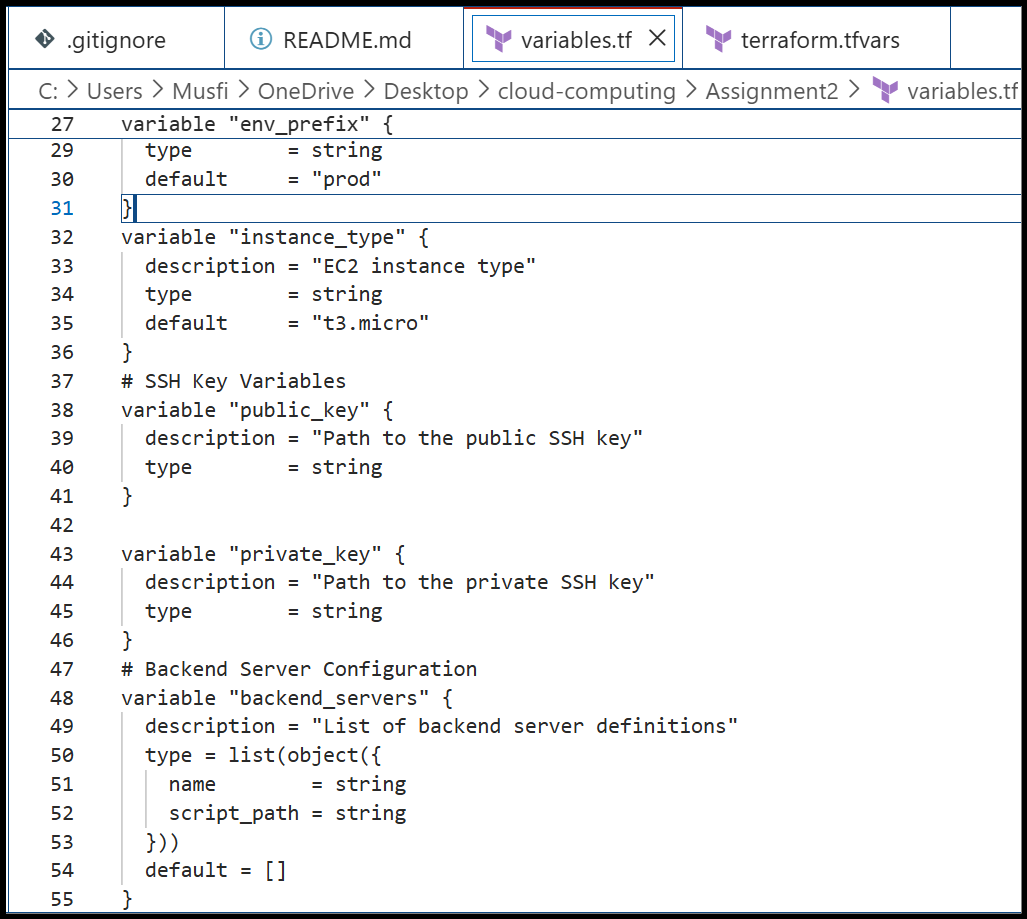
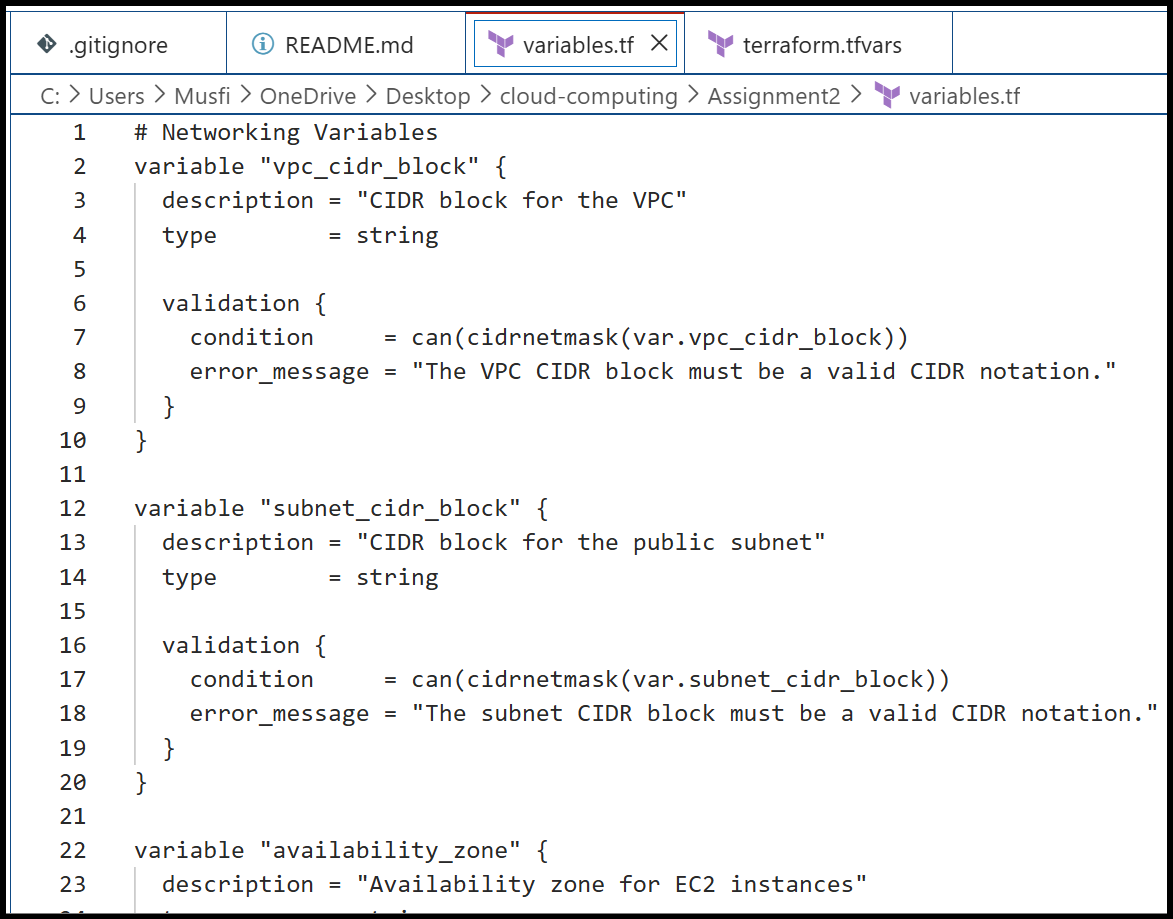
**1.1-Project Structure:**

****

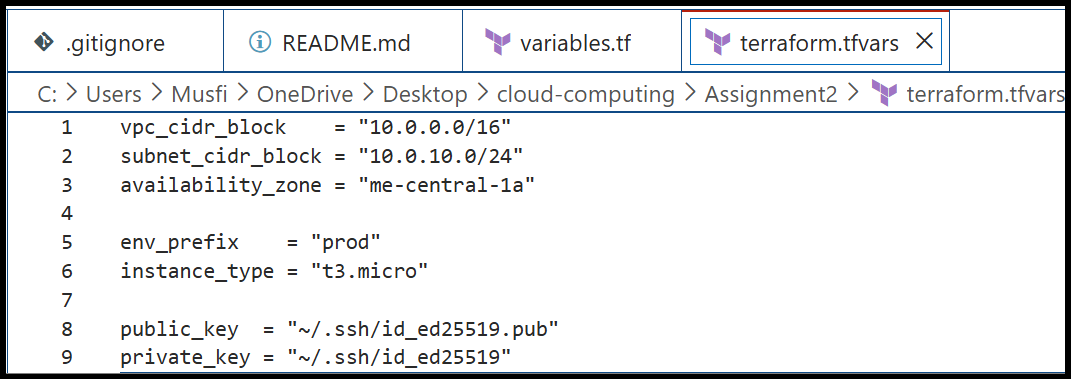
* **content of .gitignore:**

****

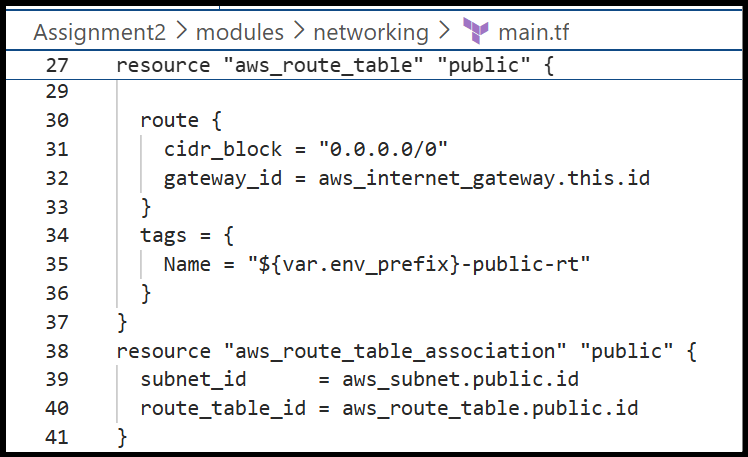
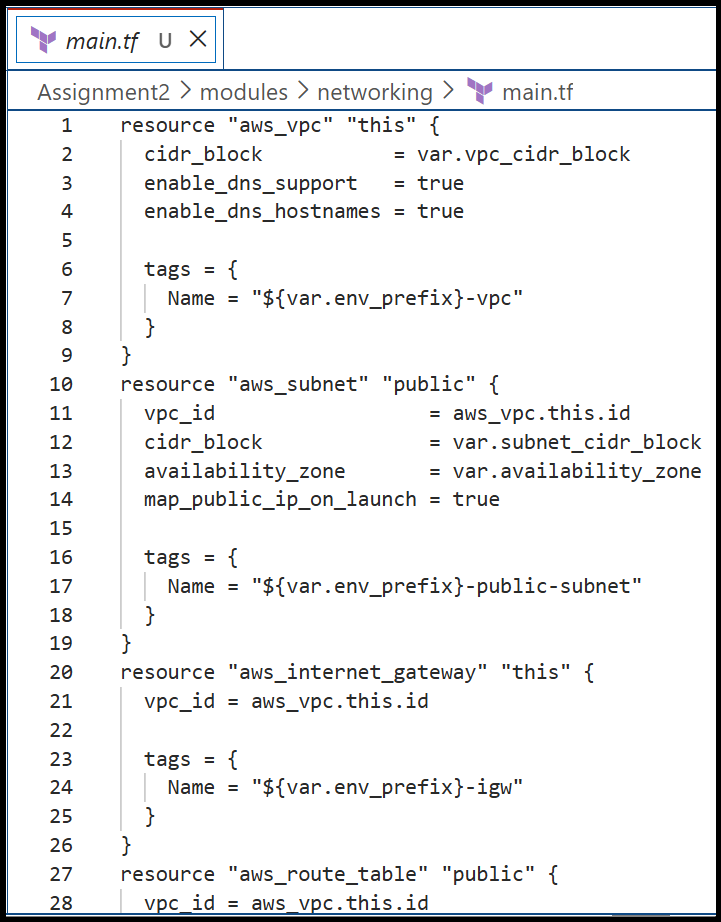
**1.2-Variable Configuration:**

****

* **terraform\_tfvars:**

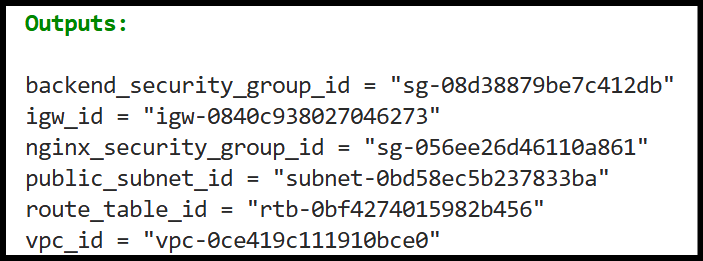
****

**1.3 Networking Module:**

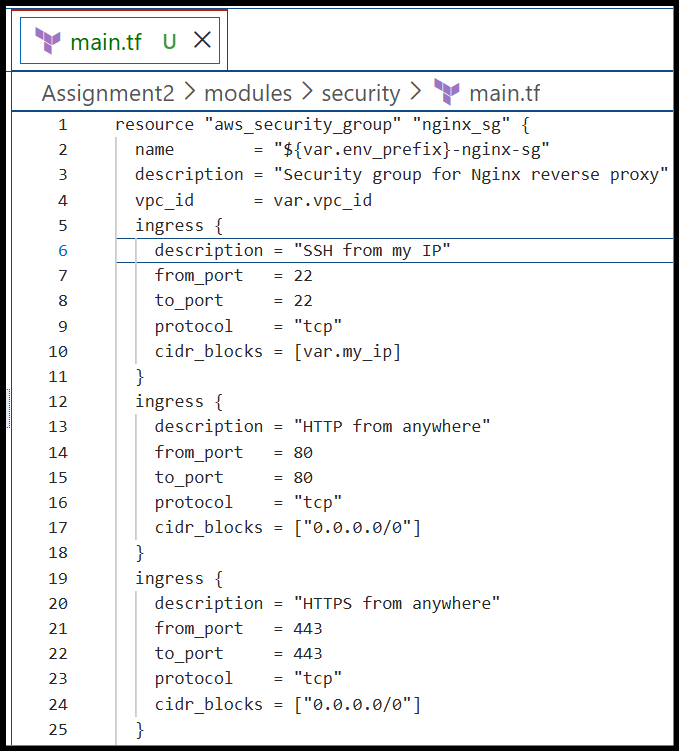
****

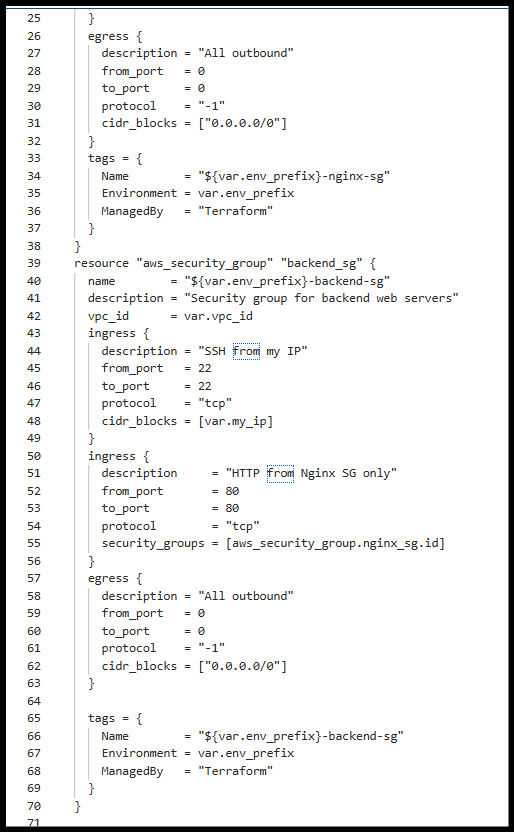
* **networking\_module\_outputs:**

****

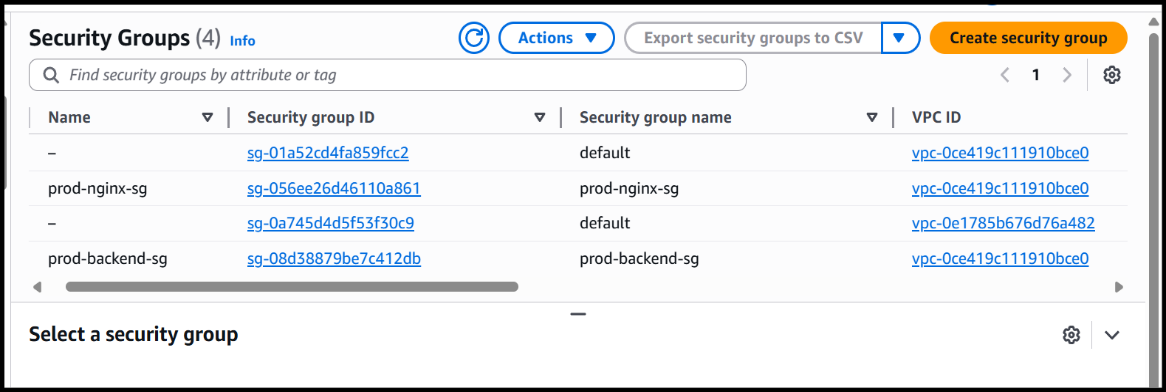
****

**1.4 Security Module:**

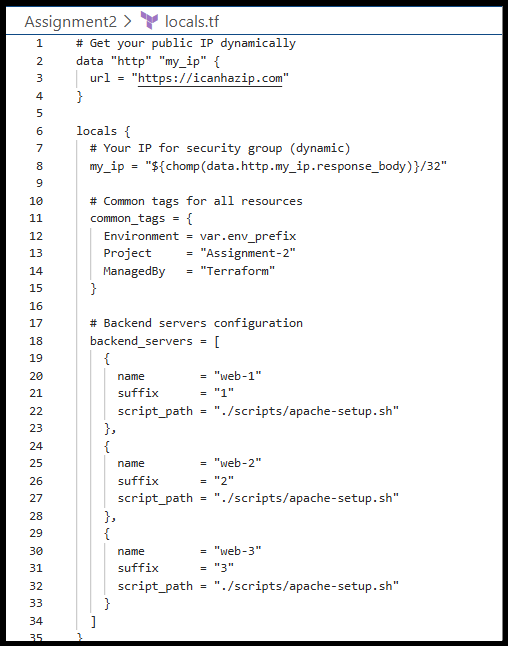
****

****

* **AWS Console:**

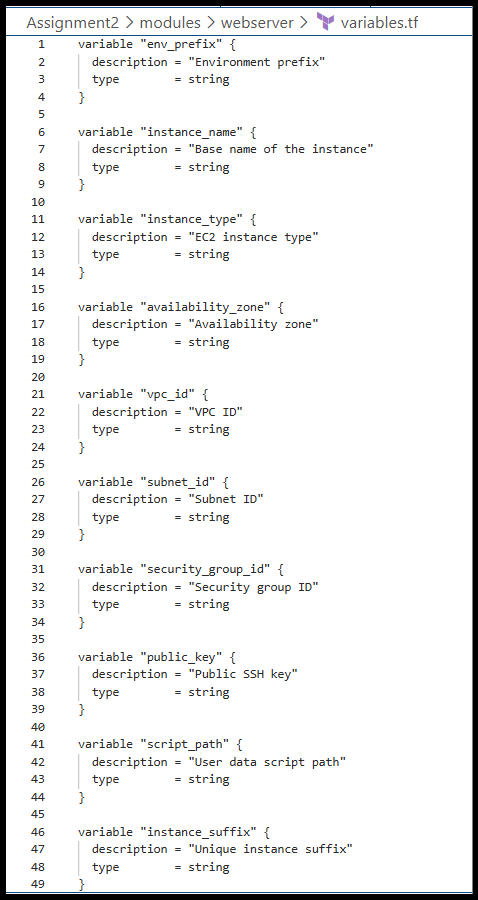
****

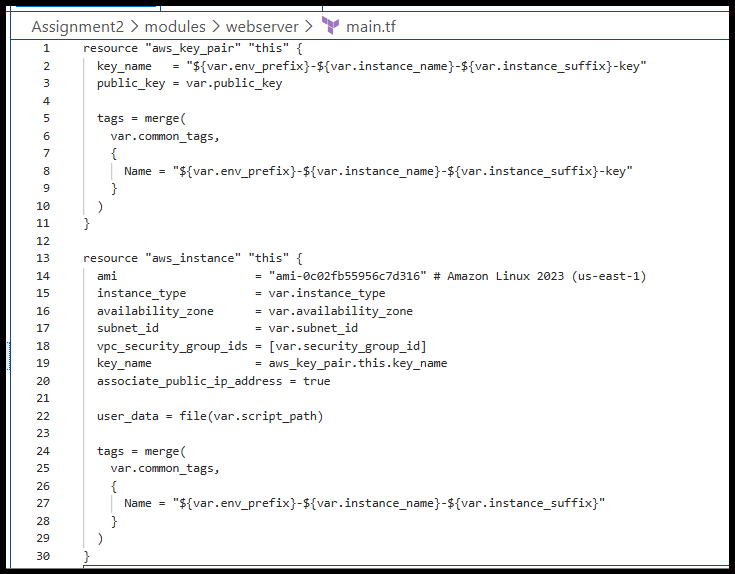
**1.5 Locals Configuration:**

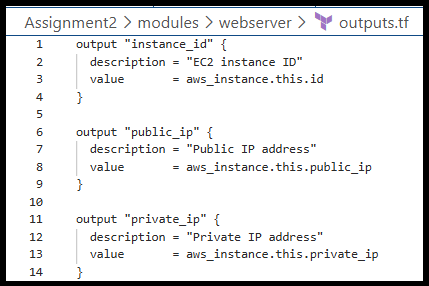
****

## **Part 2 : Webserver Module**

**2.1 Module Design:**

****

****

****

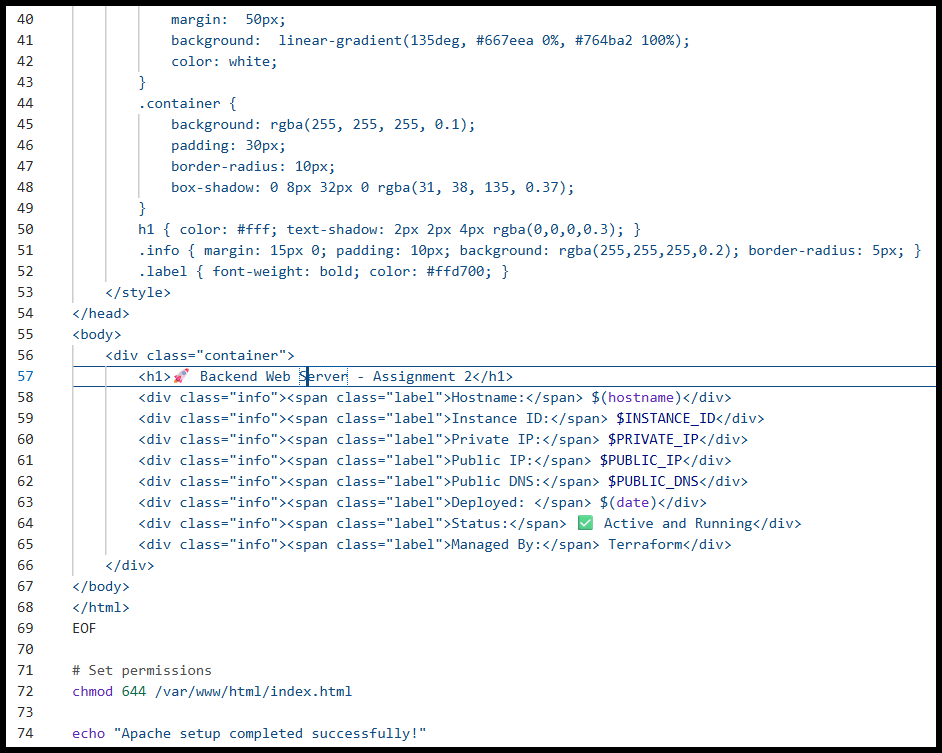
**2.2 Module Usage:**

****

## **Part 3 : Server Configuration Scripts**

**3.1 Apache Backend Server Script:**

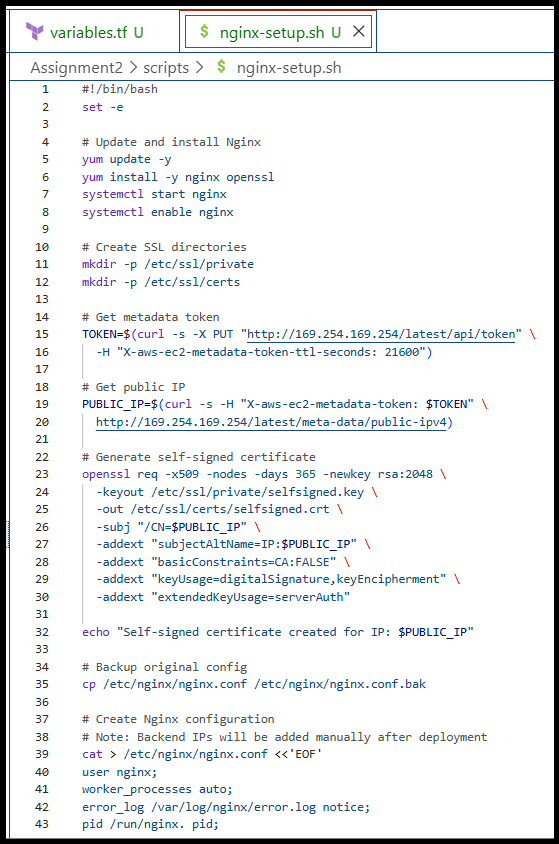
****

****

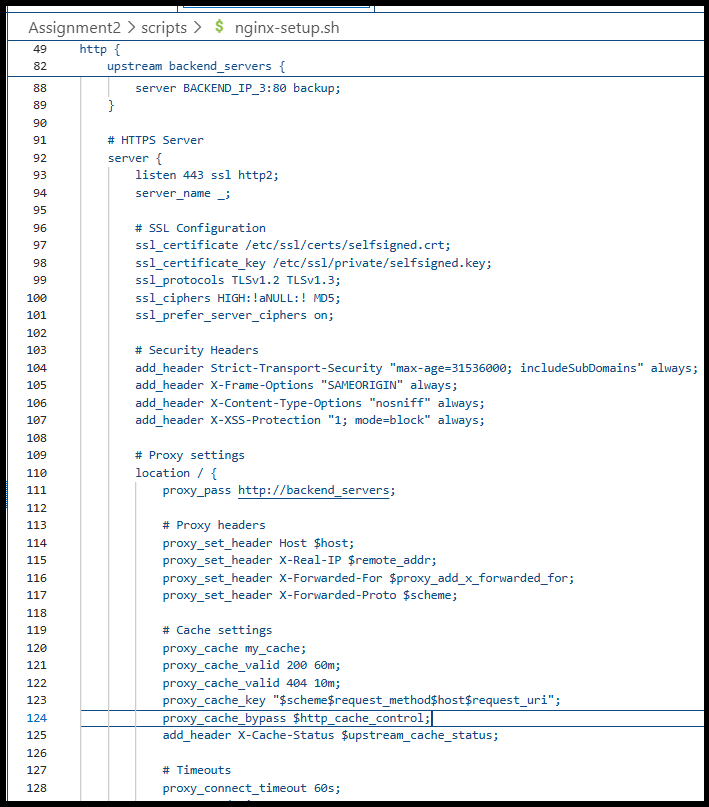
* **browser showing backend server page:**

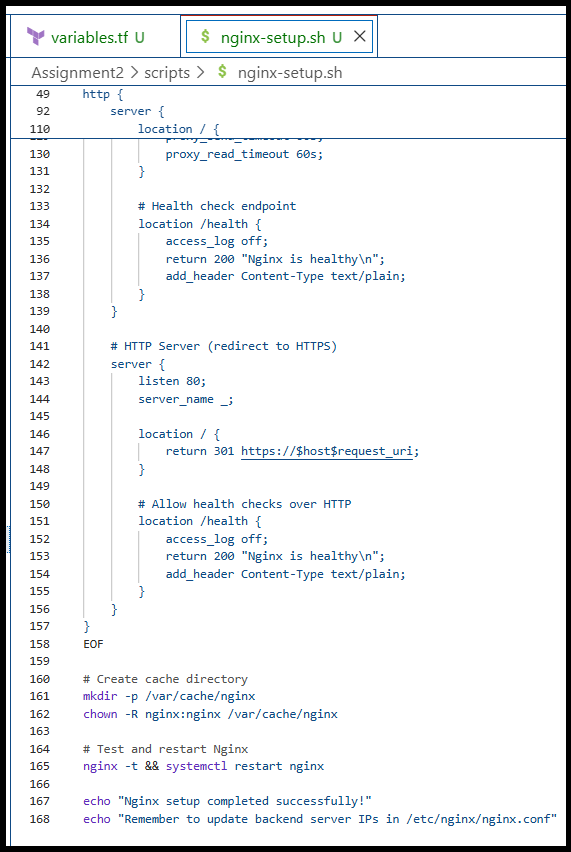
****

**3.2 Nginx Server Setup Script:**

****

****

****

****

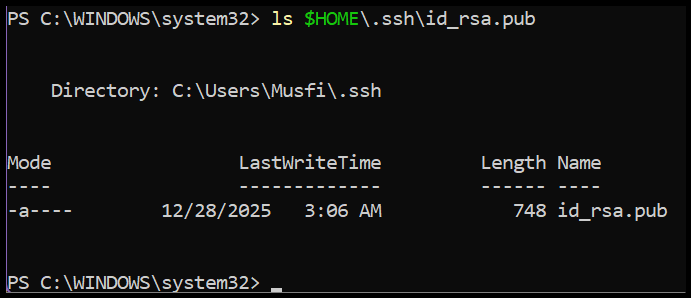
* **before backend configuration:**

****

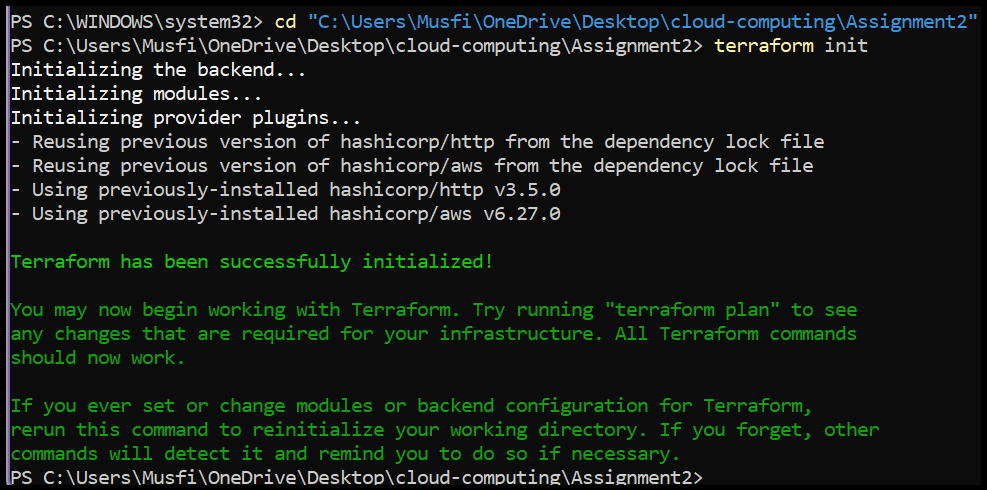
## **Part 4 : Infrastructure Deployment**

**4.1 Initial Deployment:**

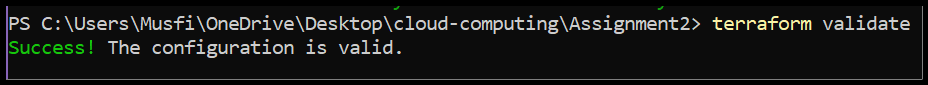
* **ssh keygen:**

****

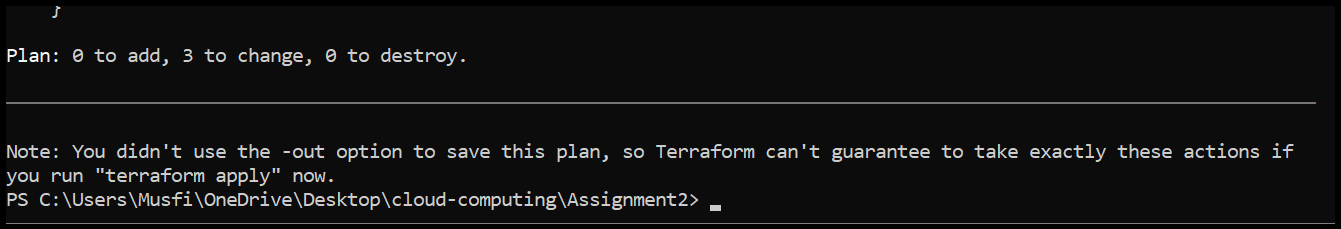
* **terraform init:**

****

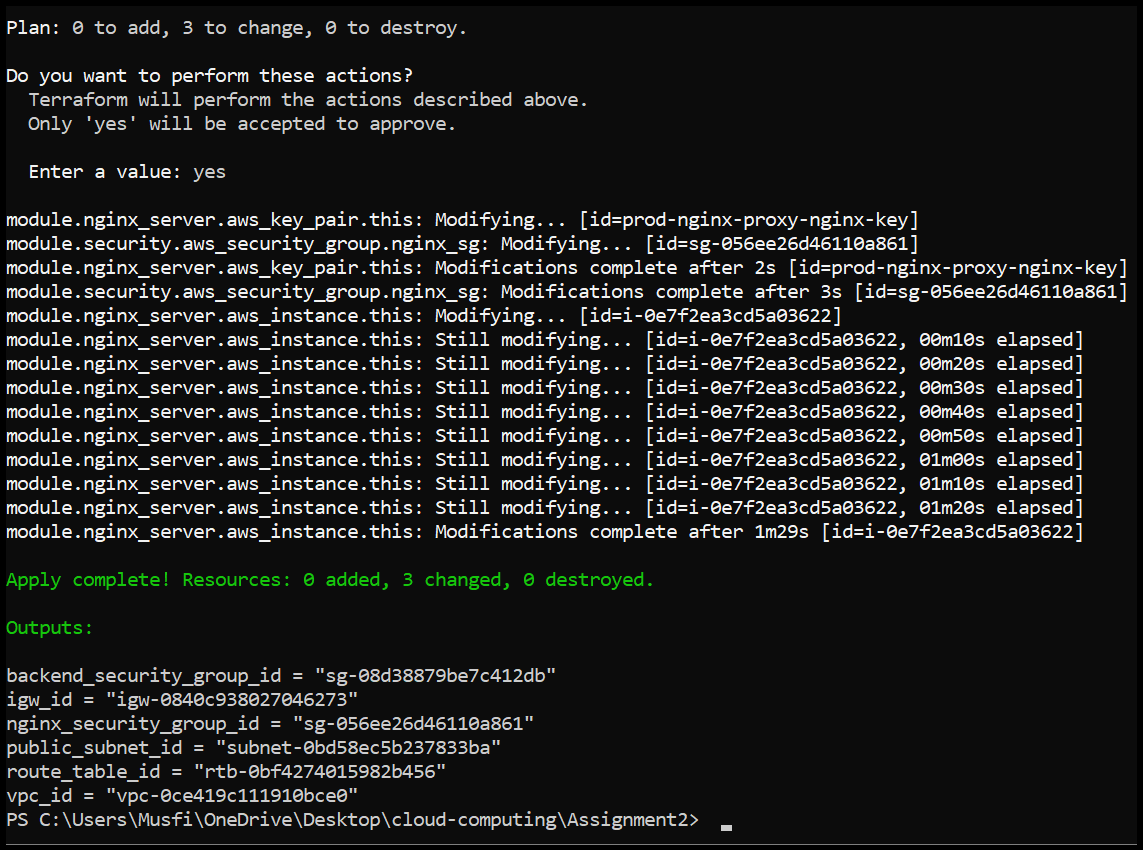
* **terraform validate:**

****

* **terraform plan:**

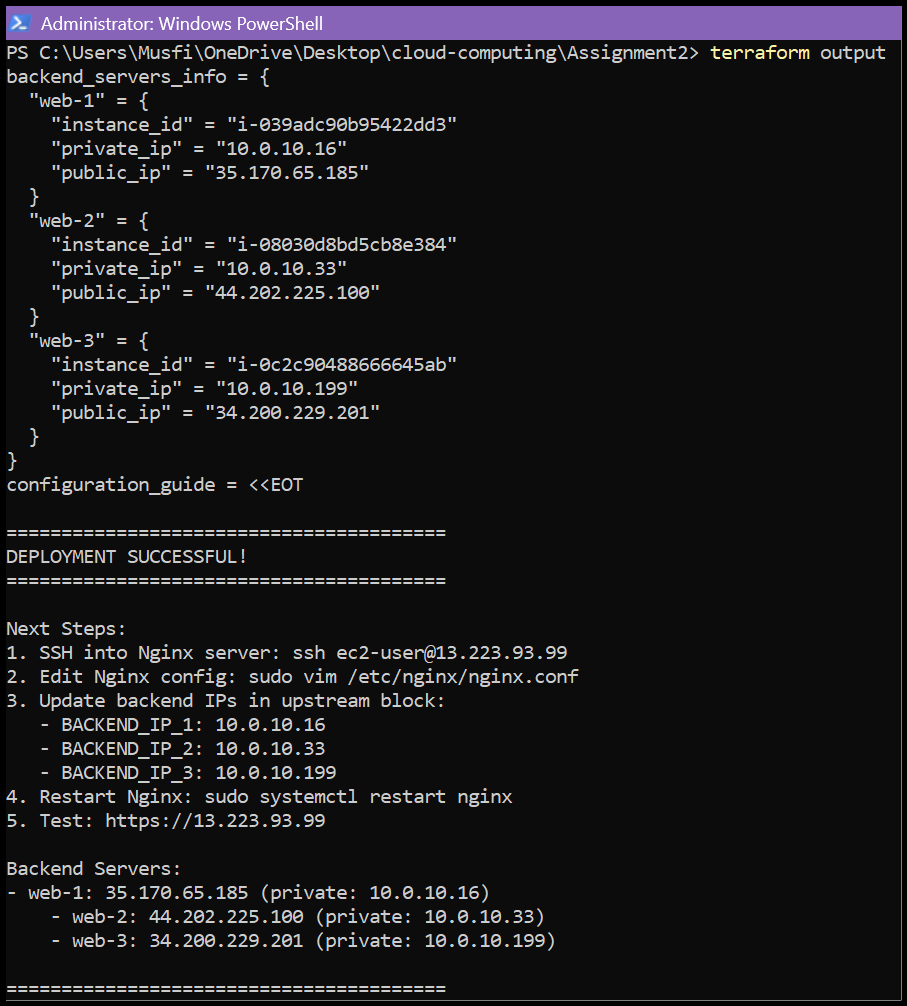
****

* **terraform apply:**

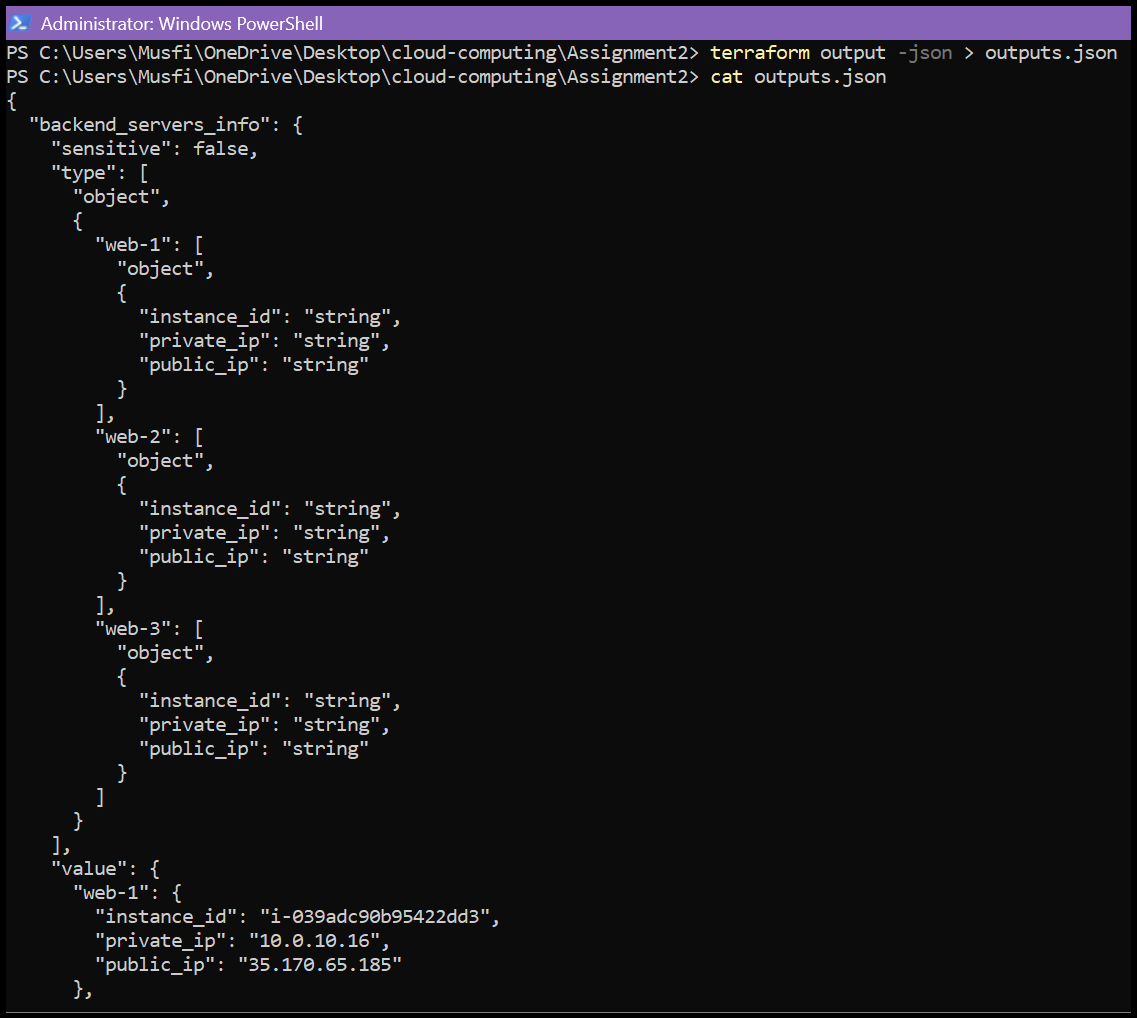
****

**4.2 Output Configuration:**

* **Terraform output:**

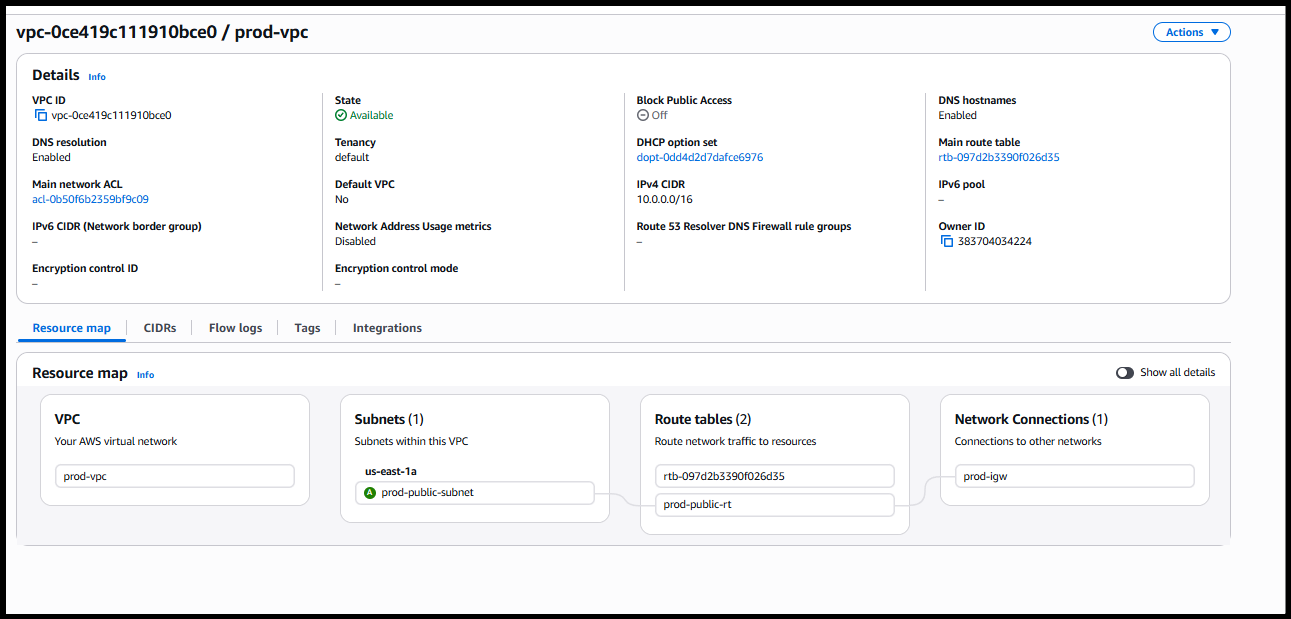
****

* **Terraform outputs json:**

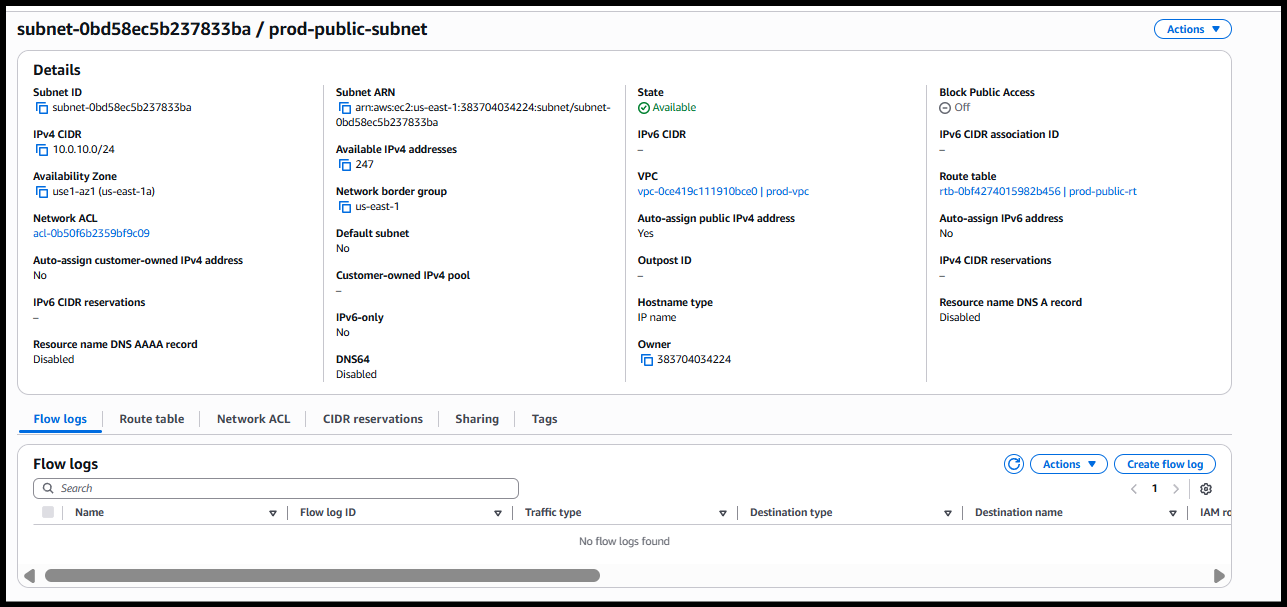
****

**4.3 AWS Console Verification:**

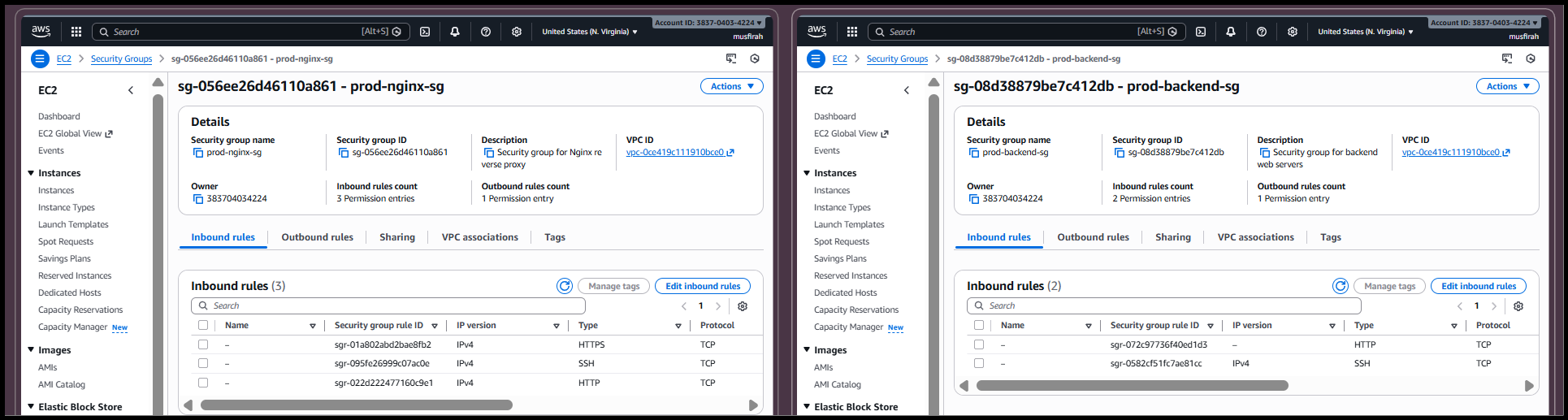
* **aws vpc:**

****

* **aws subnet:**

****

* **aws security groups:**

****

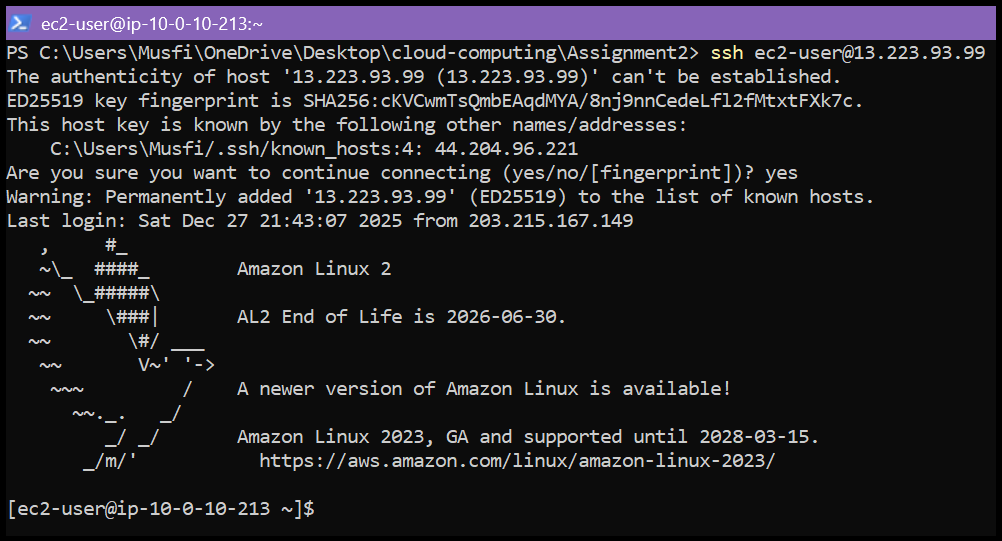
* **aws instances:**

****

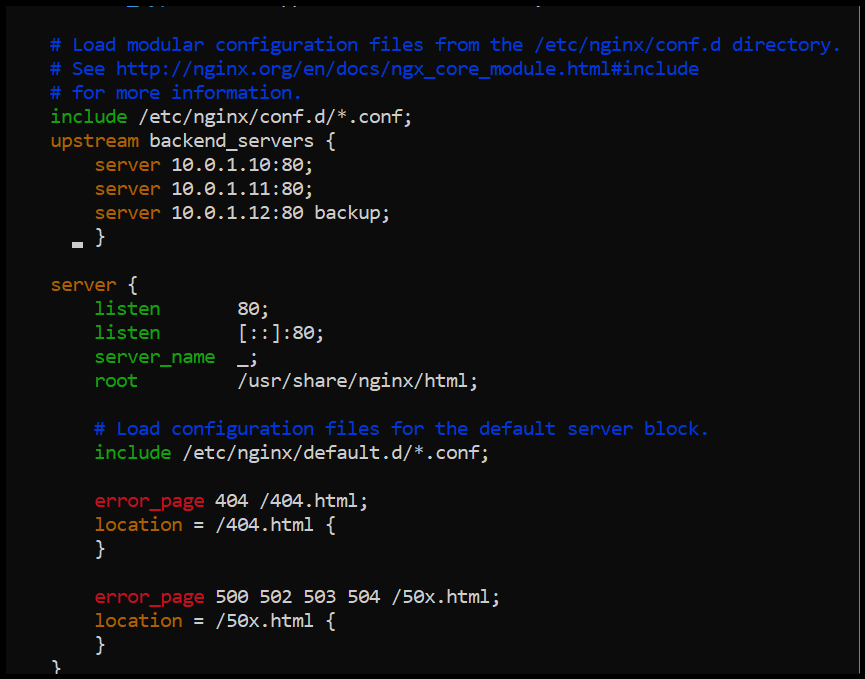
## **Part 5 : Nginx Configuration & Testing**

**5.1 Update Nginx Backend Configuration:**

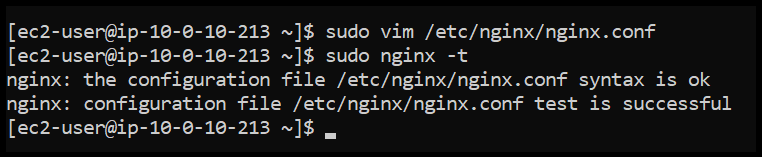
* **ssh nginx:**

****

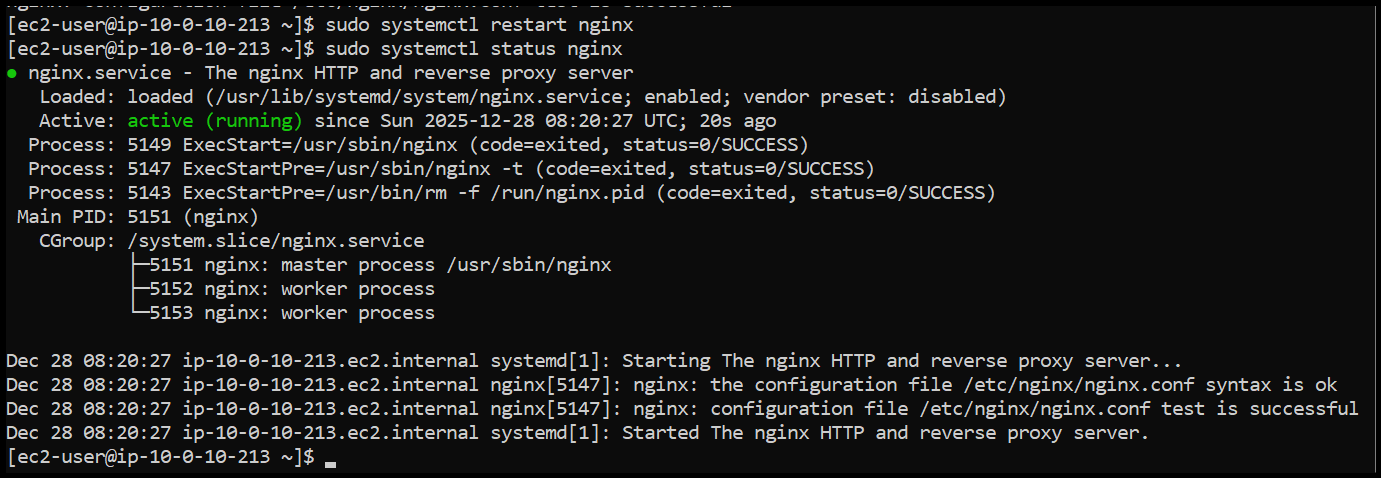
* **nginx conf updated:**

****

* **nginx test:**

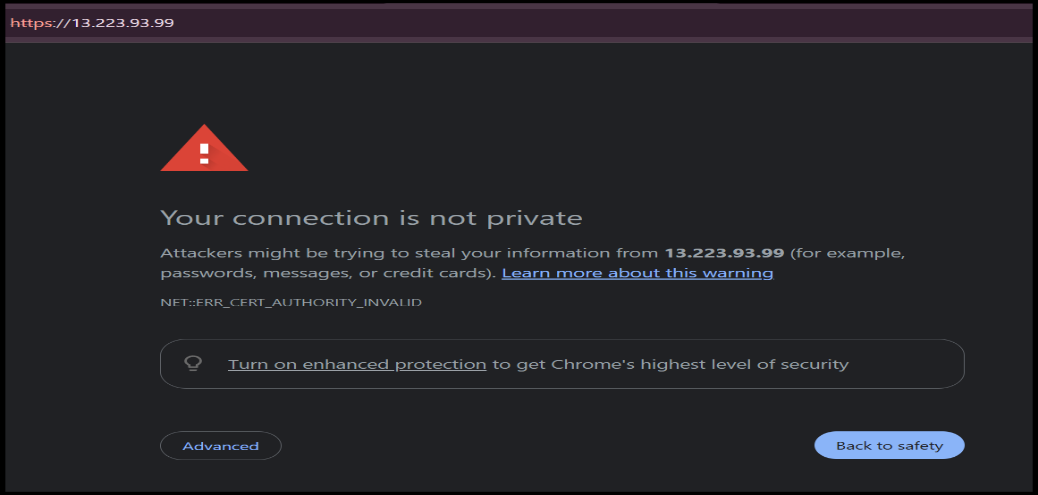
****

* **nginx restart and status:**

****

**5.2 Test Load Balancing:**

* **ssl warning:**

****

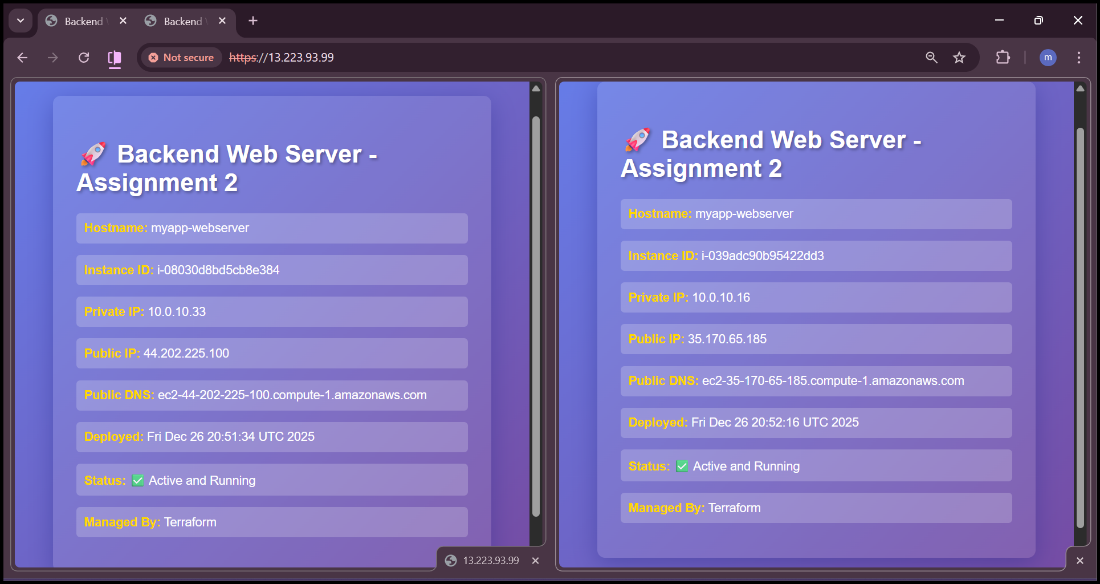
* **web1 response:**

****

* **web2 response:**

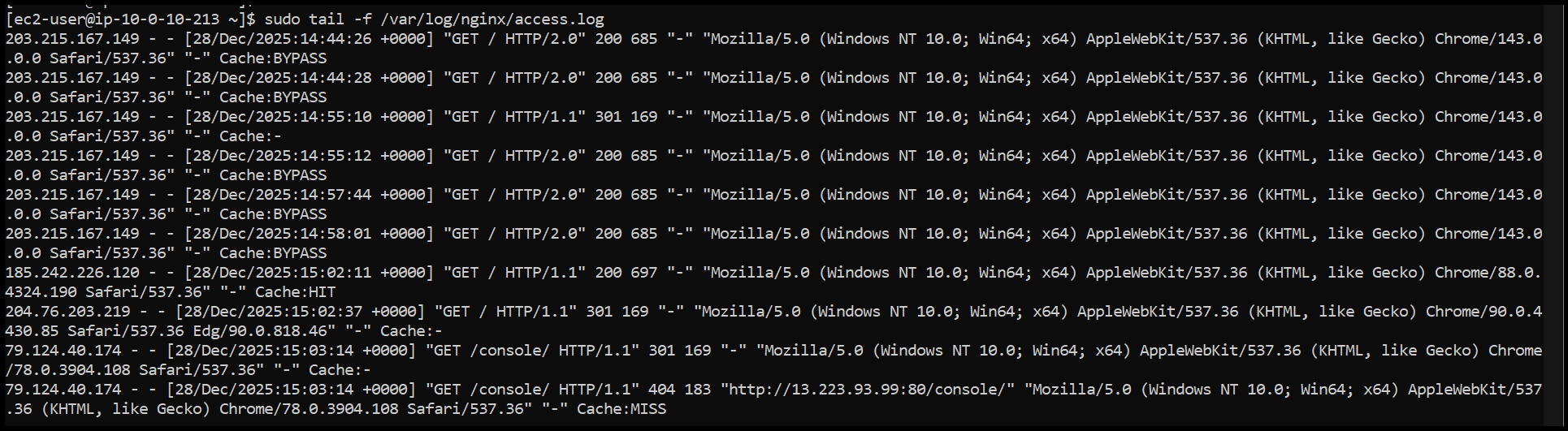
****

* **load balancing demo:**

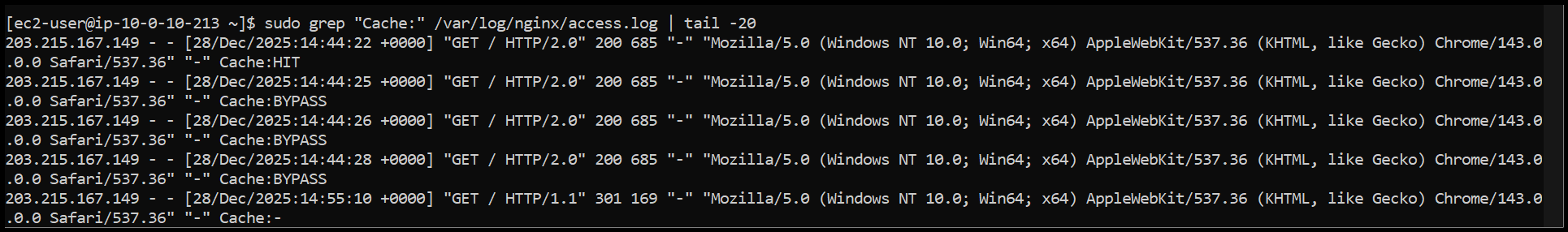
****

**5.3 Test Cache Functionality:**

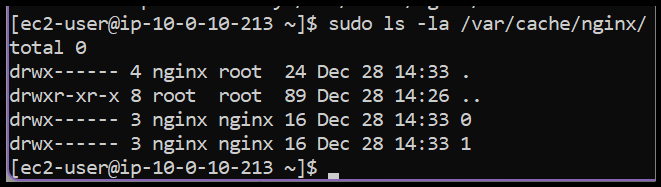
* **cache miss:**

****

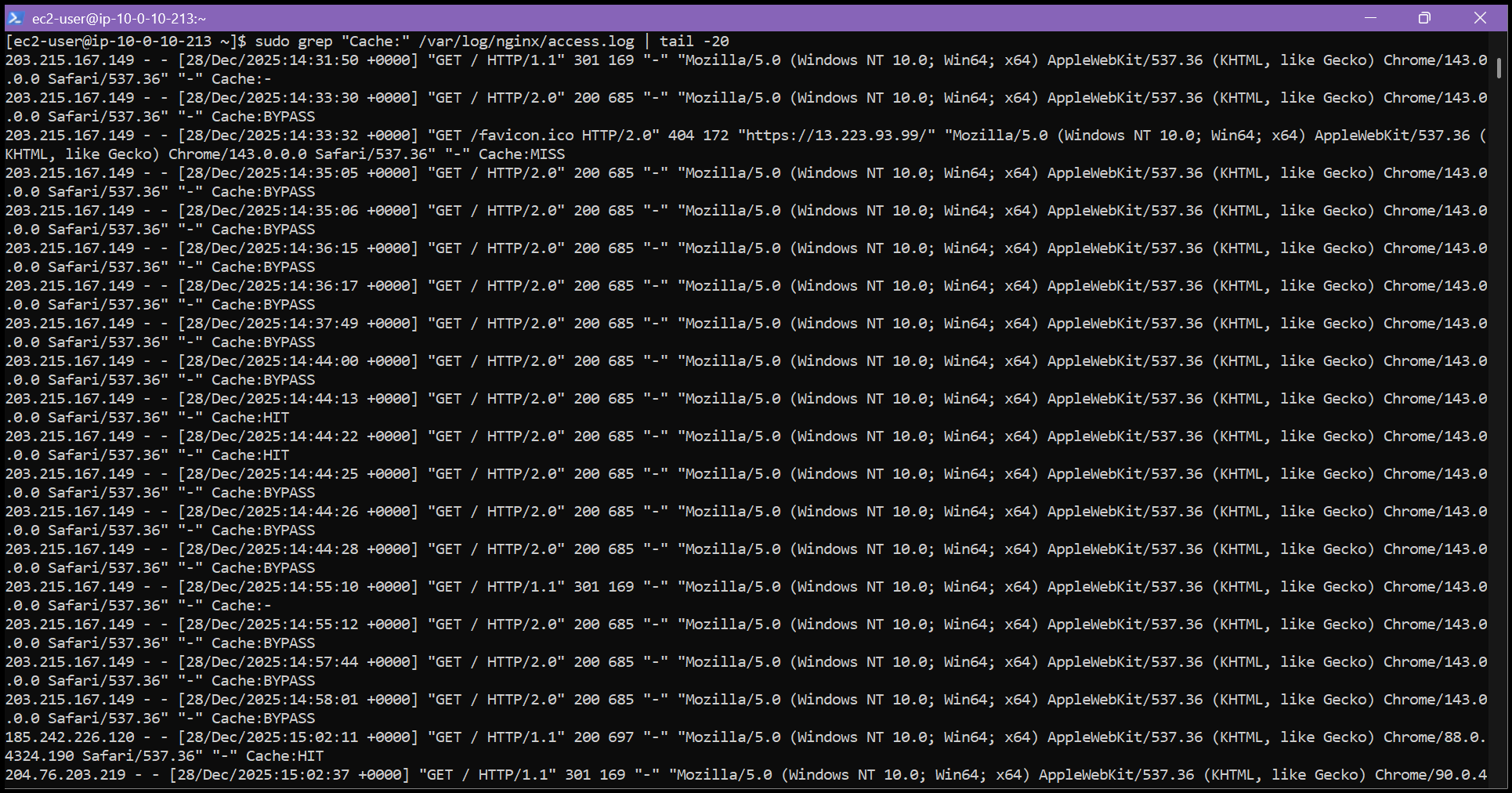
* **cache hit:**

****

* **cache directory:**

****

* **access log cache:**

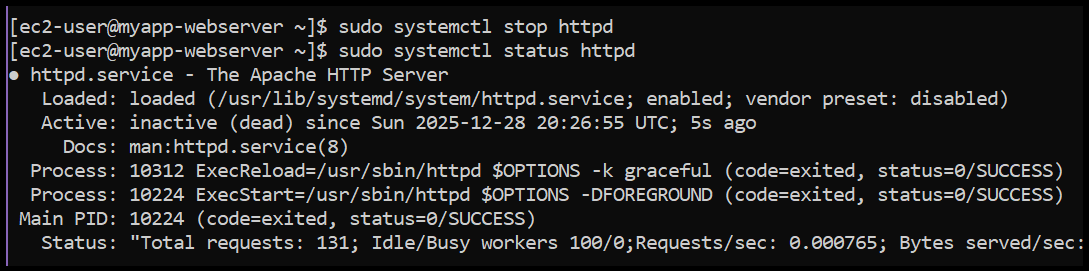
****

**5.4 Test High Availability (Backup Server):**

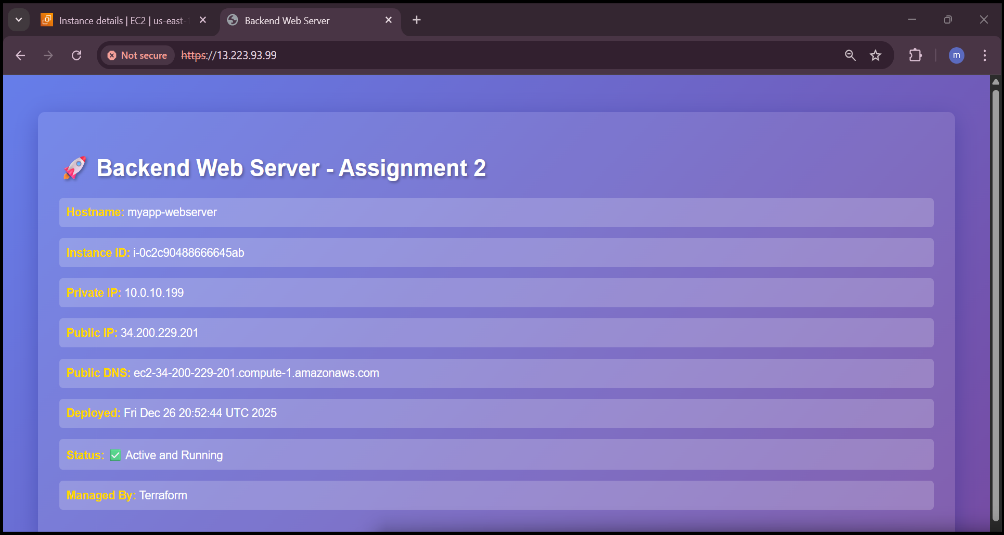
* **web1 stopped:**

****

* **web2 stopped:**

****

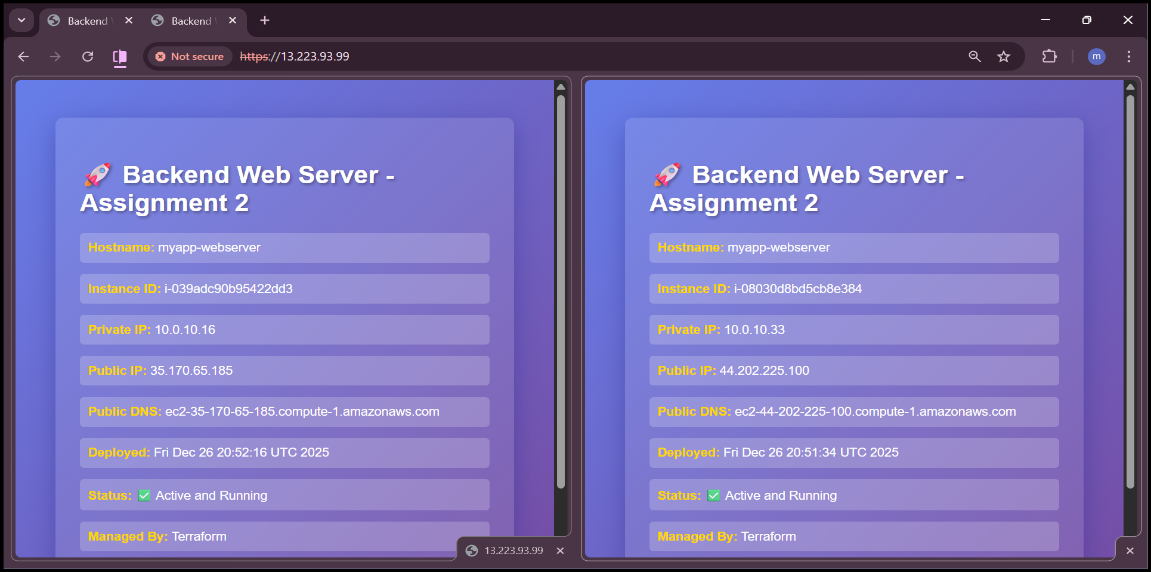
* **backup activated:**

****

* **nginx error log:**

****

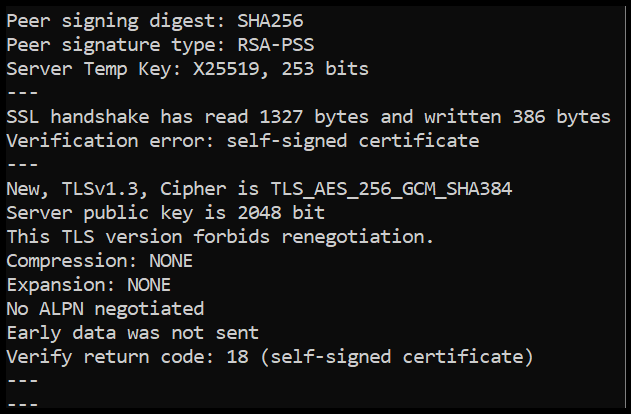
* **services restored:**

****

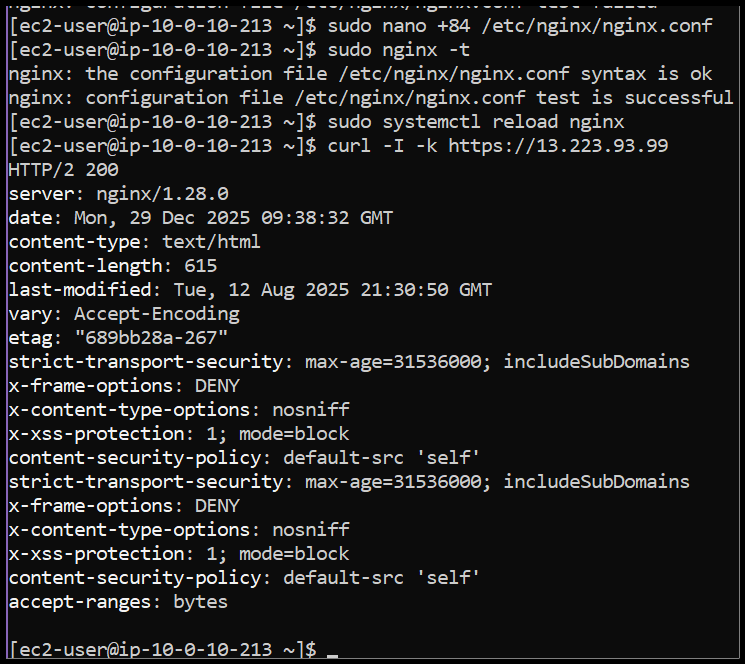
**5.5 Security & Performance Analysis:**

* **ssl certificate:**

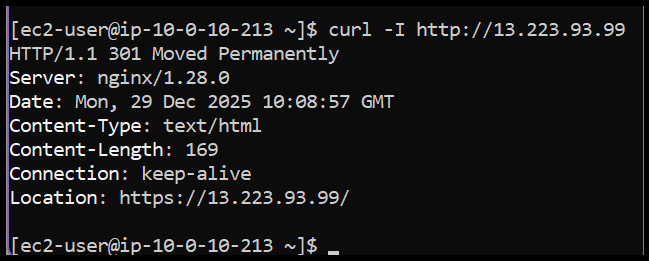
****

****

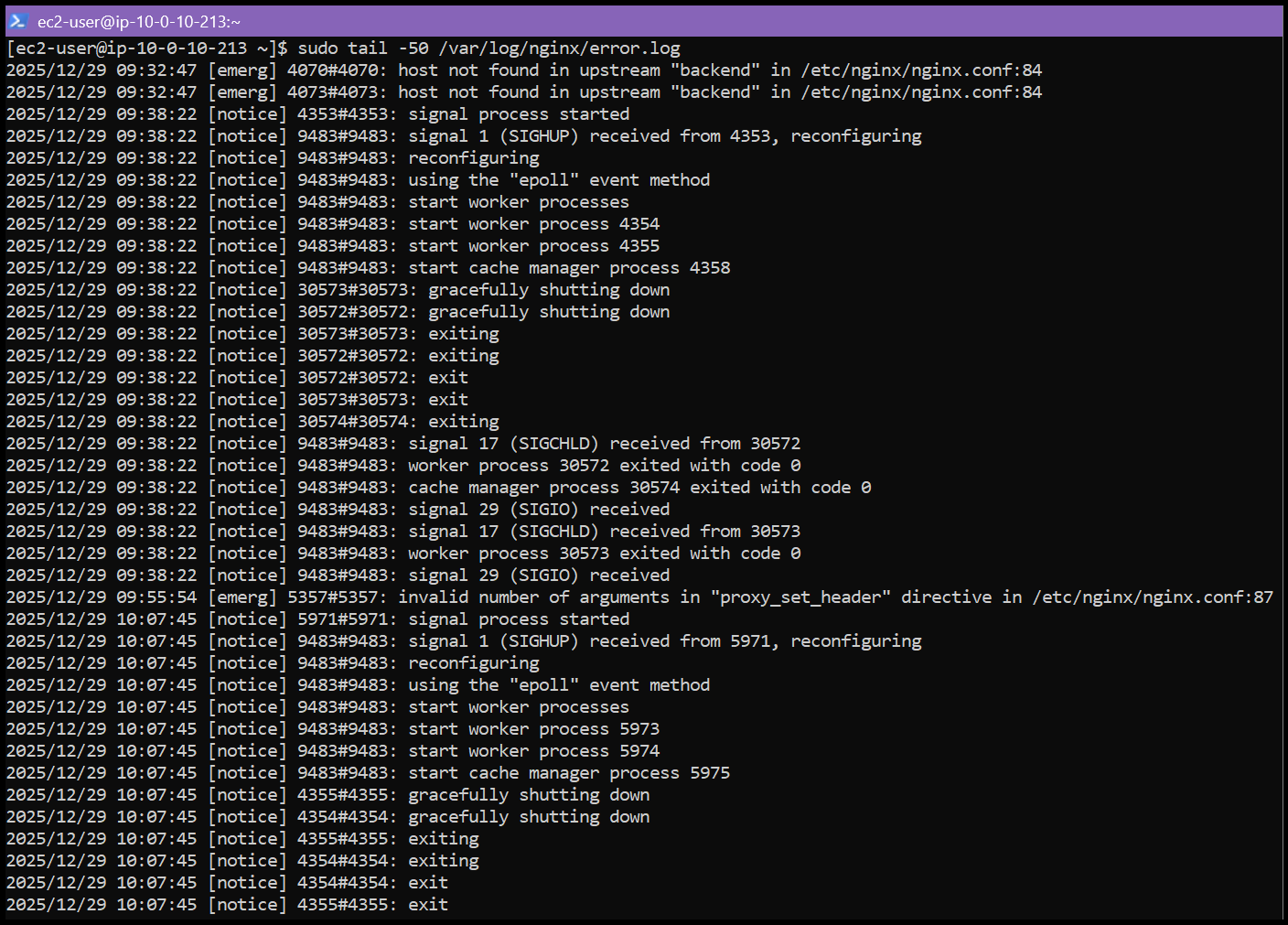
* **security headers:**

****

* **http redirect:**

****

* **error log analysis:**

****

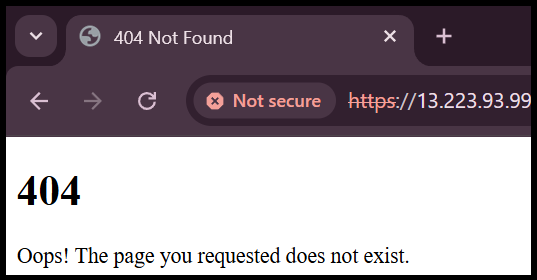
* **access log analysis:**

****

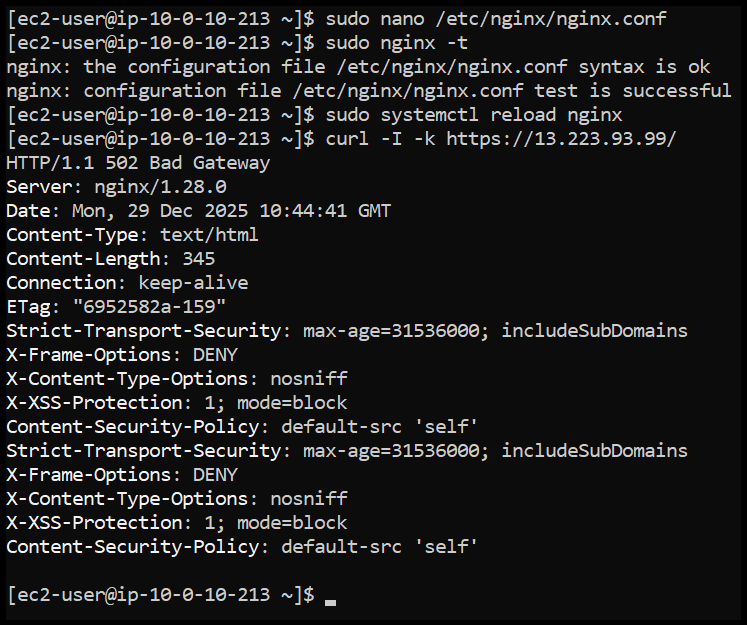
## **Bonus Tasks**

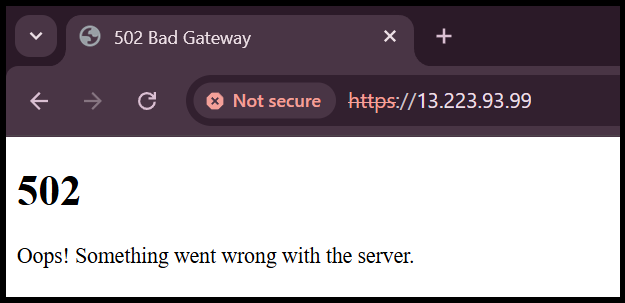
**Bonus 1: Custom Error Pages**

* **custom\_404:**

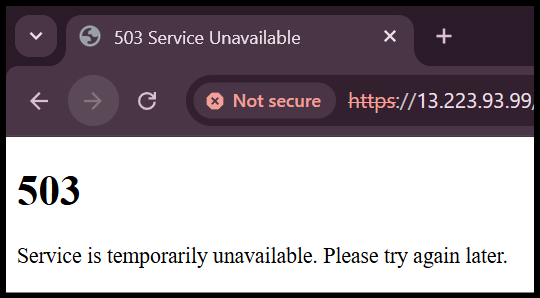
****

* **custom\_502:**

****

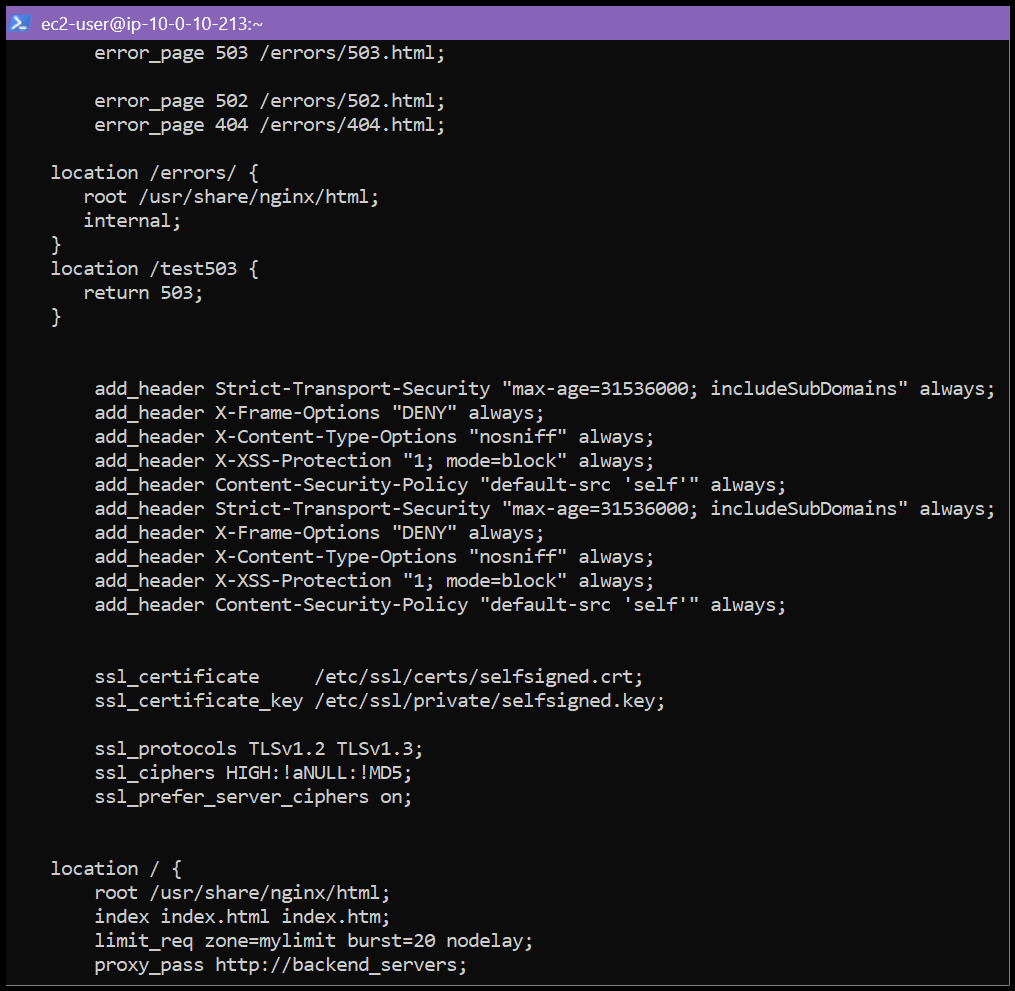
****

* **custom\_503:**

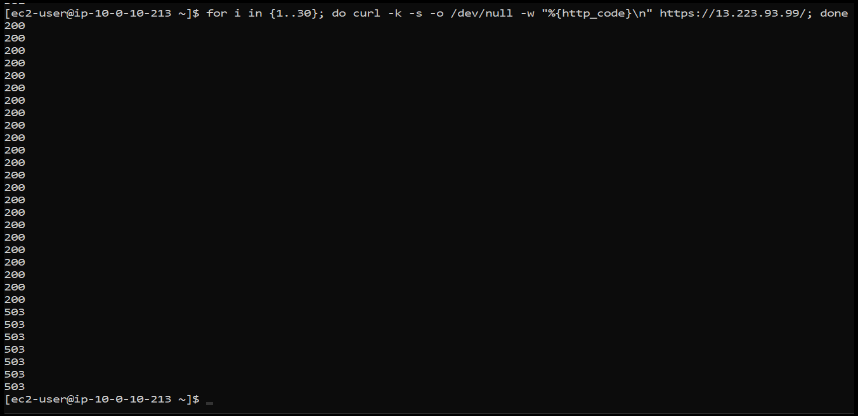
****

**Bonus 2: Implement Rate Limiting**

* **rate\_limit\_config:**

****

* **rate\_limit\_test**

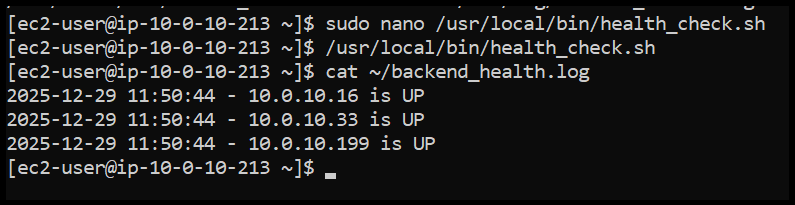
****

**Bonus 3: Health Check Automation**

* **health\_check\_script:**

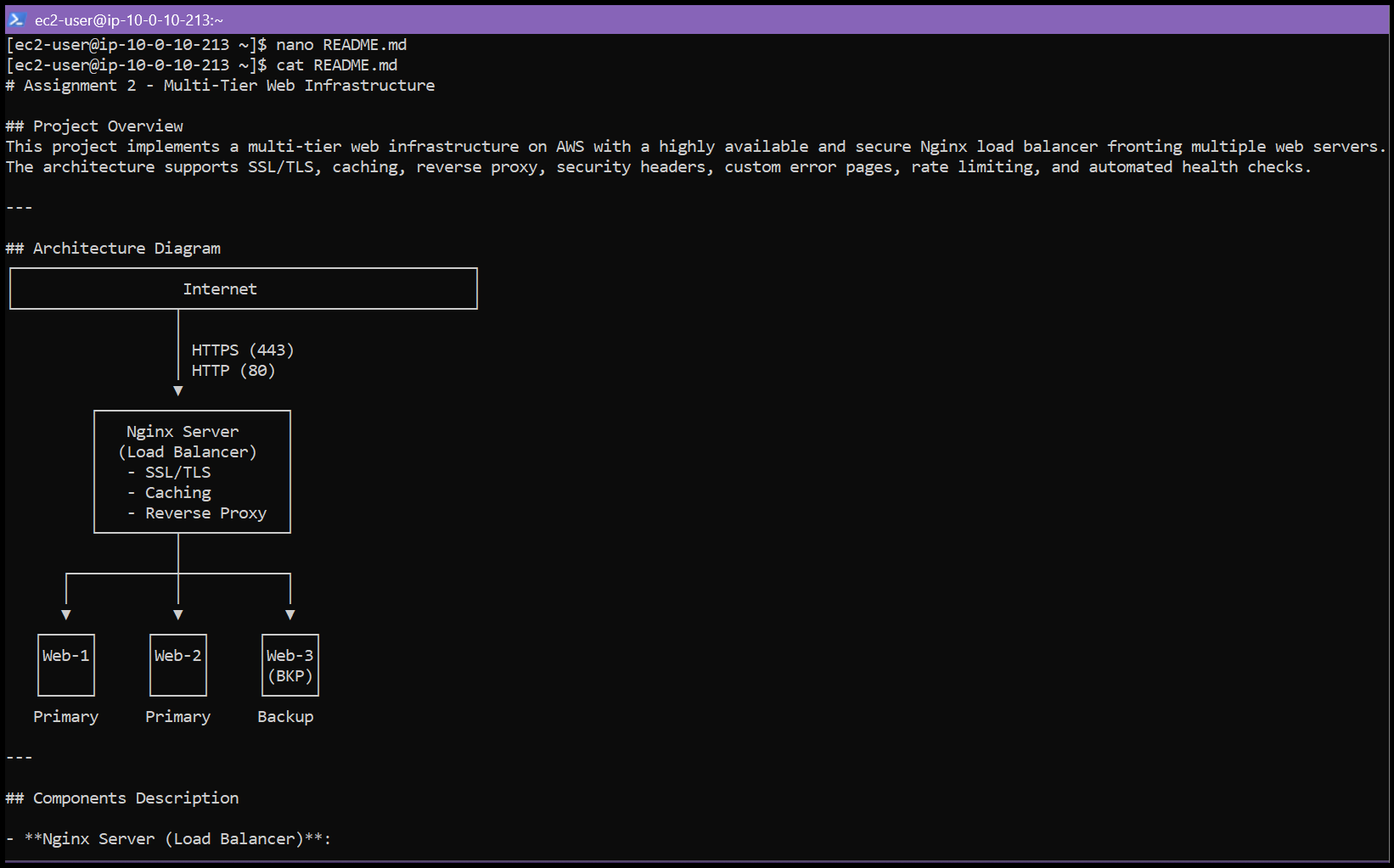
****

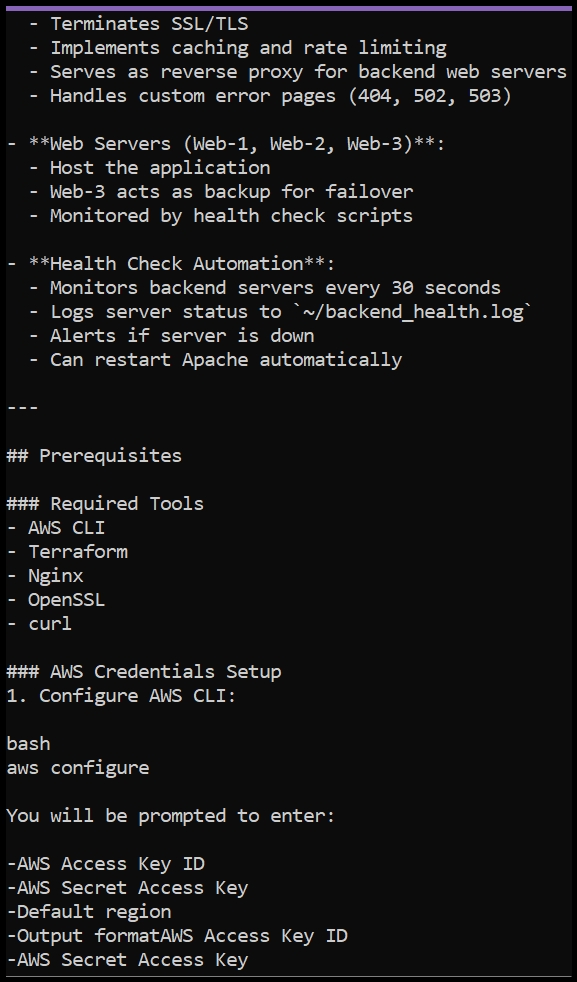
* **health\_log:**

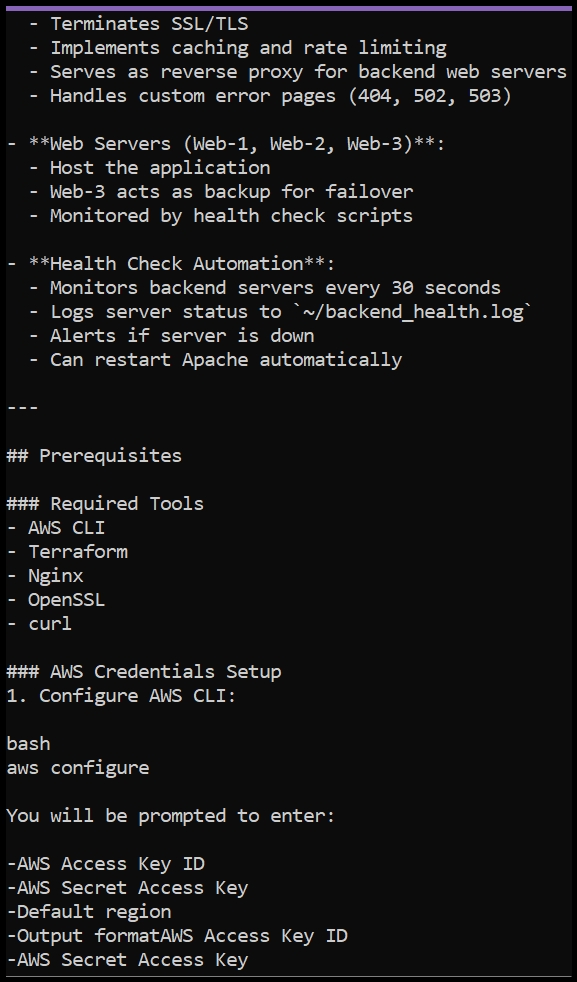
****

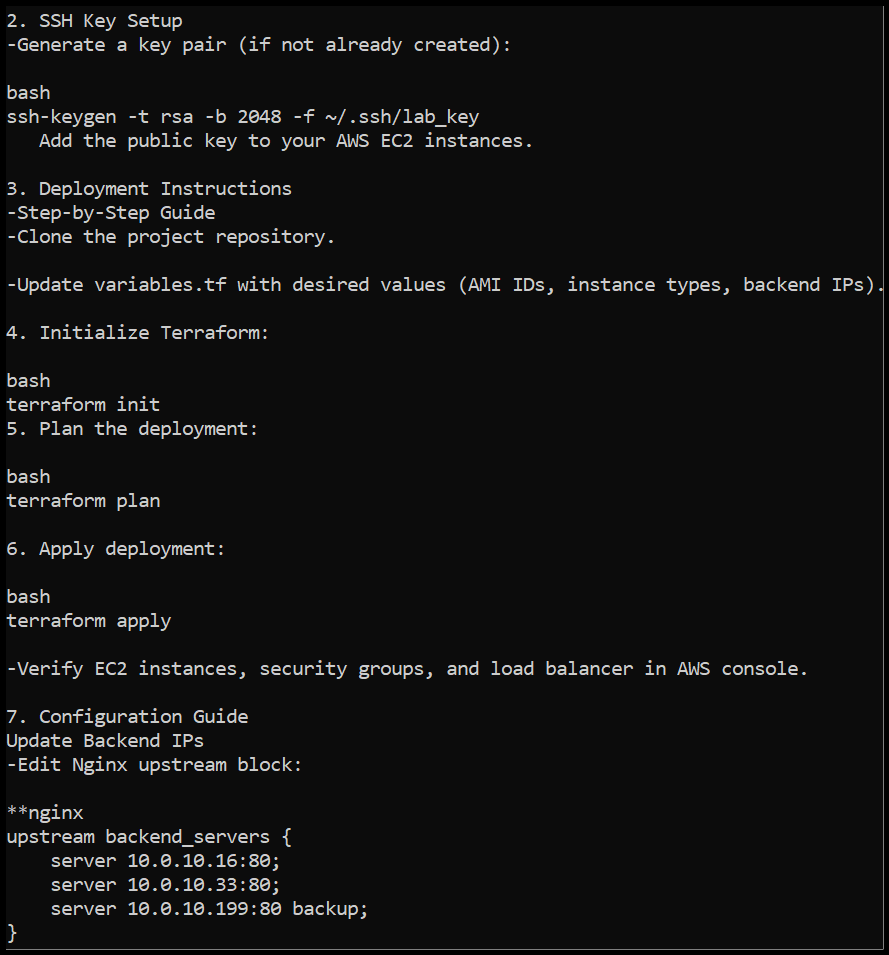
## **Part 6 : Documentation & Cleanup**

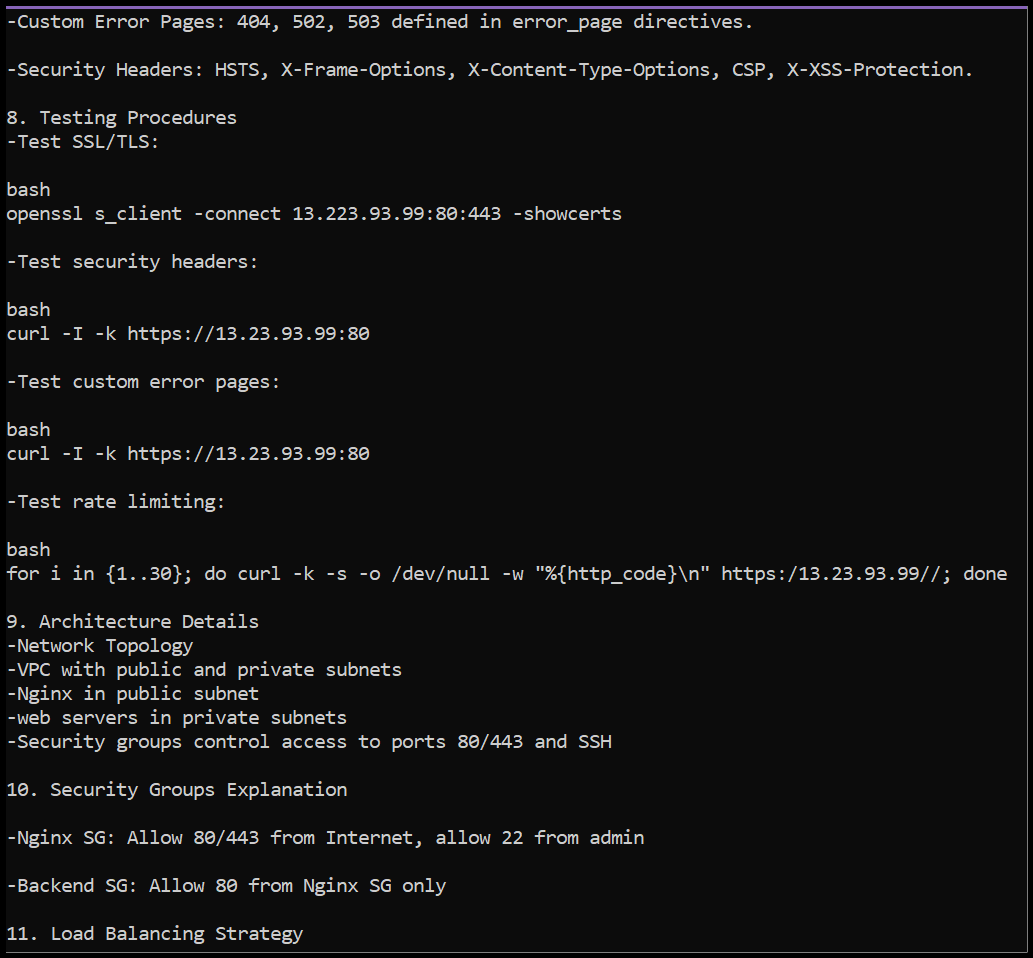
**6.1 README Documentation**

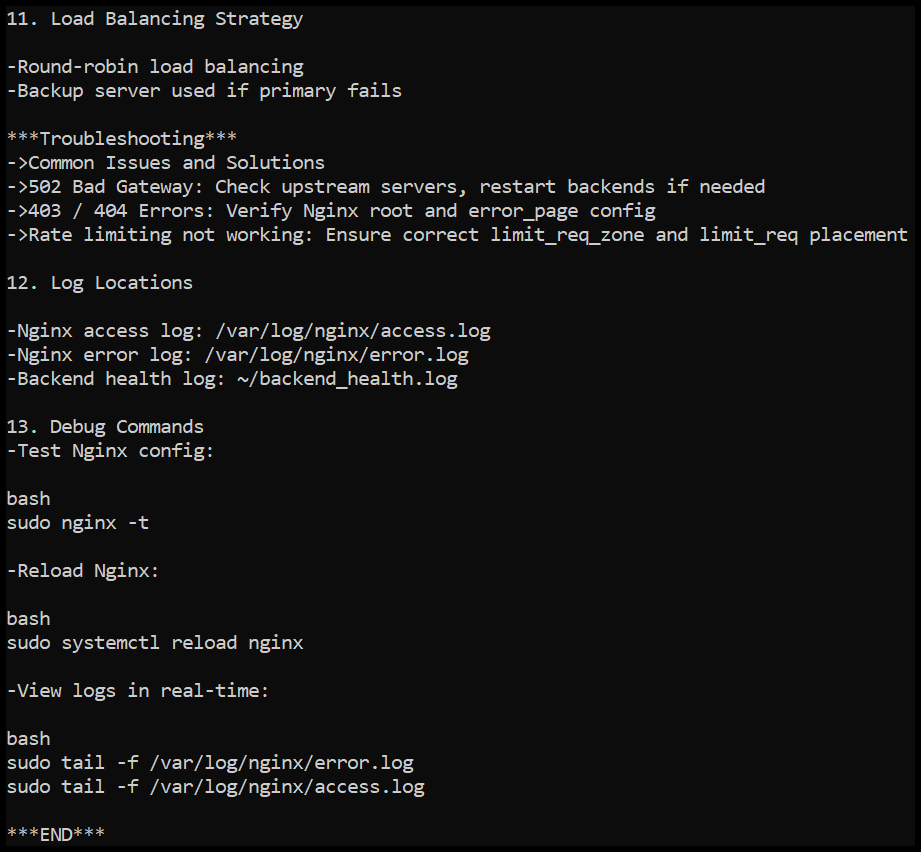
****

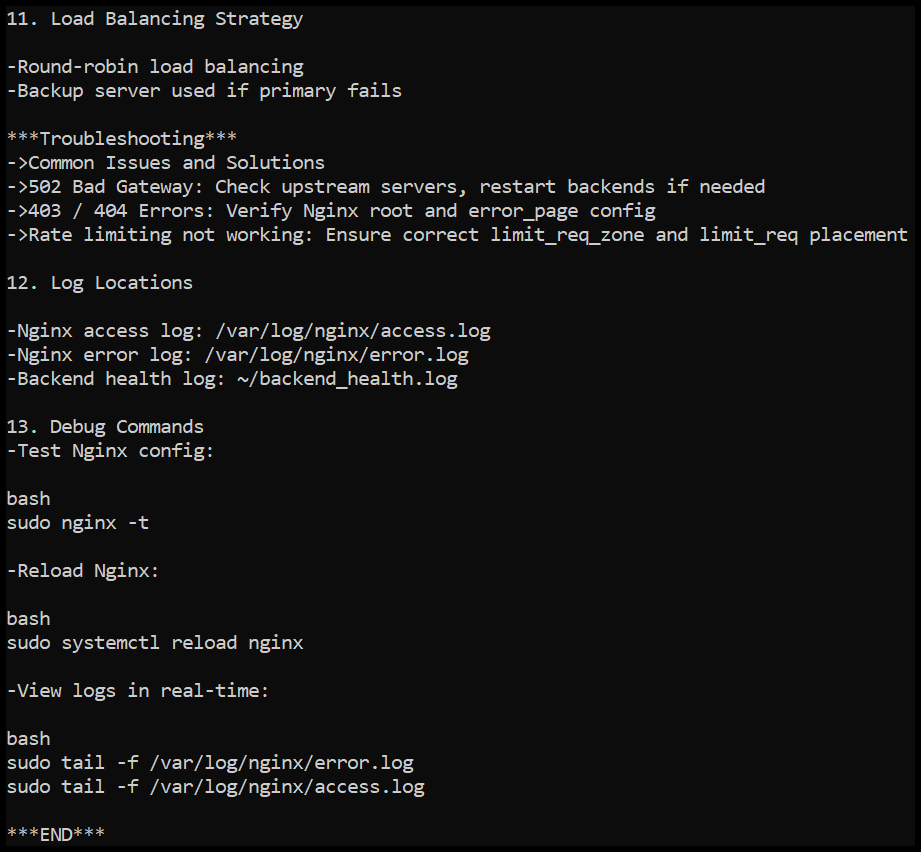
****

****

****

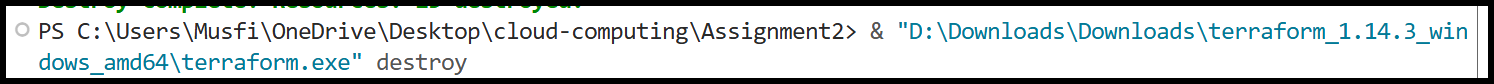
****

****

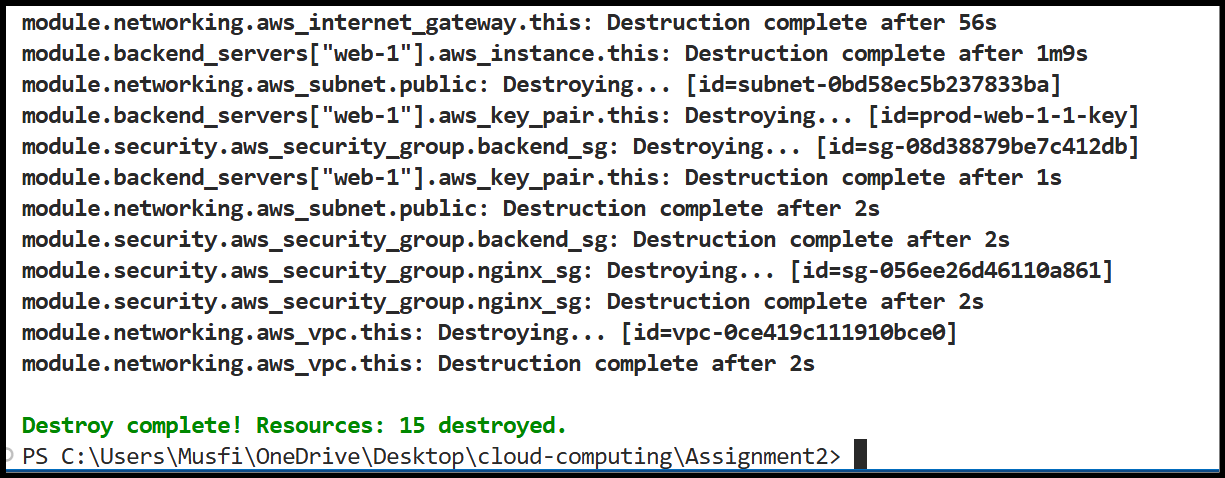
****

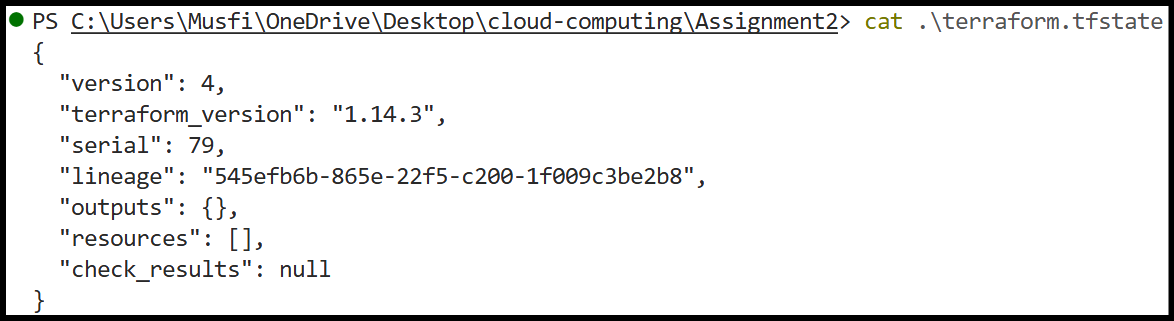
**6.2 Infrastructure Cleanup**

* **terraform\_destroy\_prompt:**

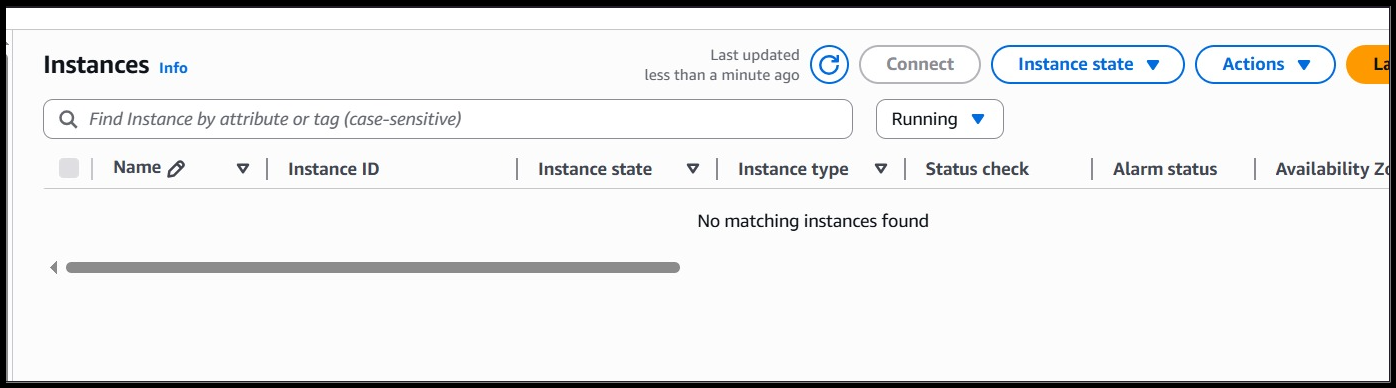
****

* **terraform\_destroy\_complete:**

****

****

* **aws\_instances\_destroyed:**

****

# ****4. Testing Results****

## **4.1 Load Balancing Validation**

Traffic was successfully shared across multiple backend servers. Server responses and logs confirmed that requests were rotating correctly between available instances.

## **4.2 Caching Efficiency**

Nginx caching improved performance by serving repeated requests faster and reducing the number of requests sent to backend servers.

## **4.3 High Availability Verification**

When one or more primary web servers were stopped, the system continued working without interruption by forwarding traffic to the backup server.

## **4.4 Security Verification**

* HTTPS connections were enforced
* Security headers were properly applied
* Direct access to backend servers was restricted

# ****5. Challenges & Solutions****

**Challenges Faced**

* Managing Terraform state files correctly
* Uploading large numbers of screenshots to GitHub
* Writing and debugging complex Nginx configurations
* Configuring AWS security groups accurately

**How These Issues Were Solved**

* Used structured and modular Terraform files
* Excluded unnecessary files using .gitignore
* Tested Nginx configurations step-by-step before deployment
* Carefully reviewed and validated security group rules

**Key Lessons Learned**

* Automation simplifies infrastructure management
* Modular design improves clarity and reusability
* Security planning is critical in cloud deployments

# ****6. Conclusion****

This assignment successfully demonstrated the deployment of a secure, scalable, and highly available cloud infrastructure using Terraform and Nginx. Practical experience was gained in cloud networking, infrastructure automation, load balancing, and system administration.

# 7. Appendices

## **Appendix A: Code Listings**

* All Terraform configuration files (main.tf, variables.tf, outputs.tf, locals.tf)
* All module files (networking/, security/, webserver/)
* Server scripts (nginx-setup.sh, apache-setup.sh)

## **Appendix B: Configuration Files**

* Example terraform.tfvars file
* .gitignore file

## **Appendix C: Additional Screenshots**

* Screenshots for each part of the assignment.
* Screenshots of Nginx configuration, SSL certificates, logs, and Terraform outputs