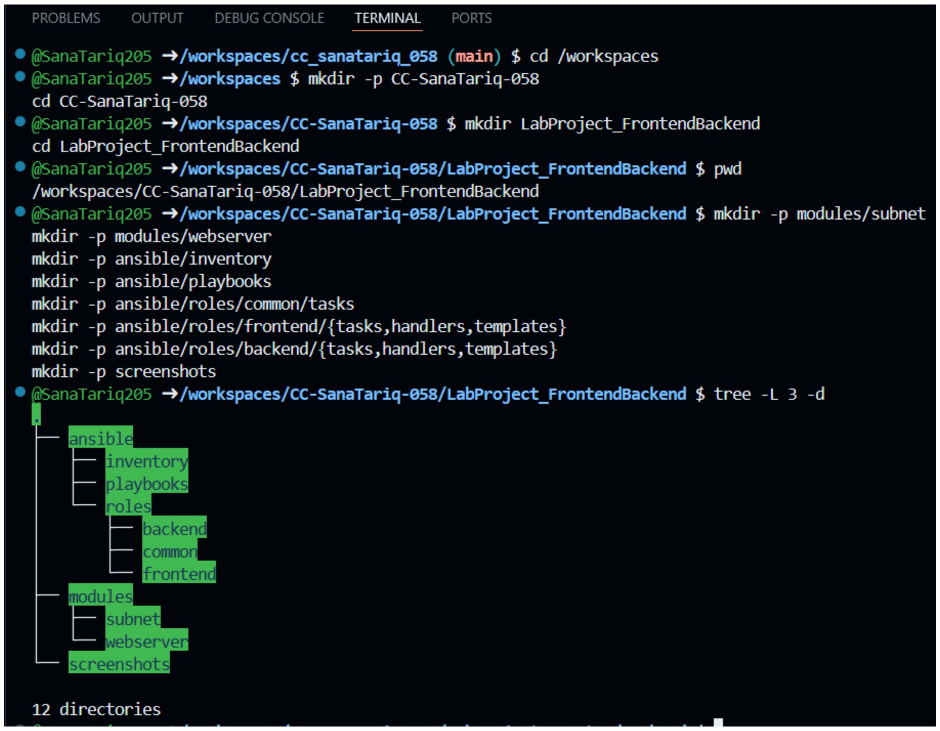
NAME: SANA TARIQ
CLASS: BSE-5B
REG.NO: 058
SUBJECT: CLOUD COMPUTING

LAB PROJECT

PHASE 1: Initial Setup & Repository

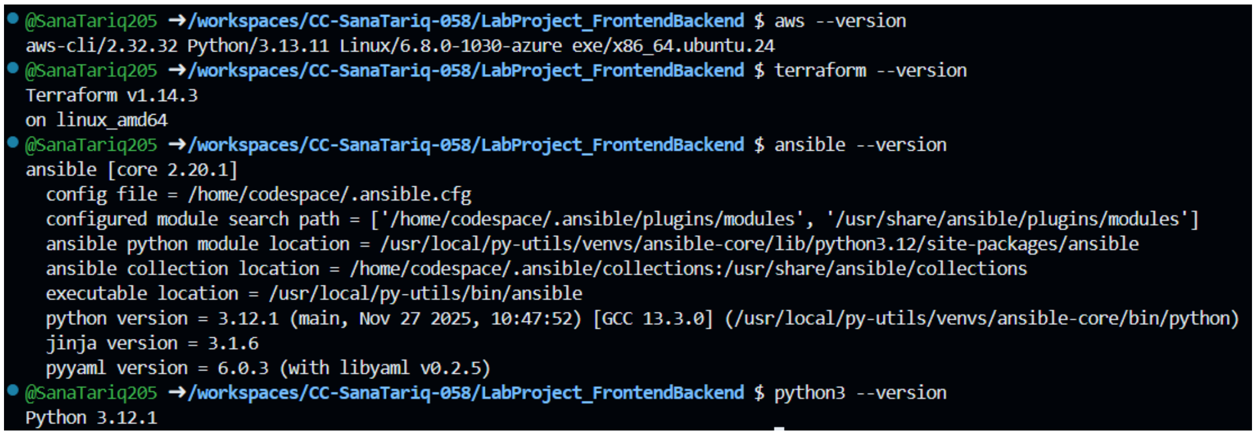
Step 1: Setup Repository Structure

Step 2: Create Complete Directory Structure



```
@SanaTariq205 ➜/workspaces/cc_sanatariq_058 (main) $ cd /workspaces
@SanaTariq205 ➜/workspaces $ mkdir -p CC-SanaTariq-058
cd CC-SanaTariq-058
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058 $ mkdir LabProject_FrontendBackend
cd LabProject_FrontendBackend
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ pwd
/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ mkdir -p modules/subnet
mkdir -p modules/webserver
mkdir -p ansible/inventory
mkdir -p ansible/playbooks
mkdir -p ansible/roles/common/tasks
mkdir -p ansible/roles/frontend/{tasks,handlers,templates}
mkdir -p ansible/roles/backend/{tasks,handlers,templates}
mkdir -p screenshots
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ tree -L 3 -d
.
├── ansible
│   ├── inventory
│   ├── playbooks
│   └── roles
│       ├── backend
│       ├── common
│       └── frontend
├── modules
│   ├── subnet
│   └── webserver
└── screenshots

12 directories
```

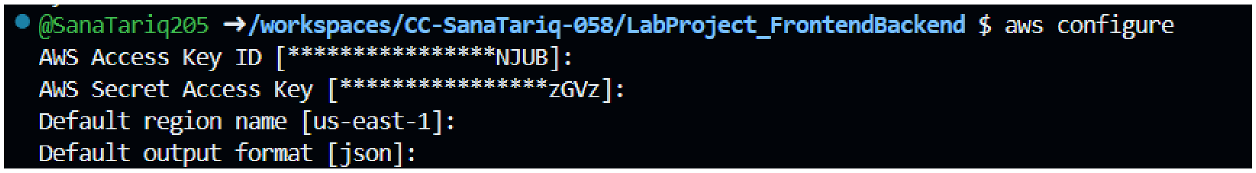
Step 3: Verify Required Tools



```
● @SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ aws --version
  aws-cli/2.32.32 Python/3.13.11 Linux/6.8.0-1030-azure exe/x86_64.ubuntu.24
● @SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ terraform --version
  Terraform v1.14.3
  on linux_amd64
● @SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ ansible --version
  ansible [core 2.20.1]
    config file = /home/codespace/.ansible.cfg
    configured module search path = ['/home/codespace/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
    ansible python module location = /usr/local/py-utils/venvs/ansible-core/lib/python3.12/site-packages/ansible
    ansible collection location = /home/codespace/.ansible/collections:/usr/share/ansible/collections
    executable location = /usr/local/py-utils/bin/ansible
    python version = 3.12.1 (main, Nov 27 2025, 10:47:52) [GCC 13.3.0] (/usr/local/py-utils/venvs/ansible-core/bin/python)
    jinja version = 3.1.6
    pyyaml version = 6.0.3 (with libyaml v0.2.5)
● @SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ python3 --version
  Python 3.12.1
```

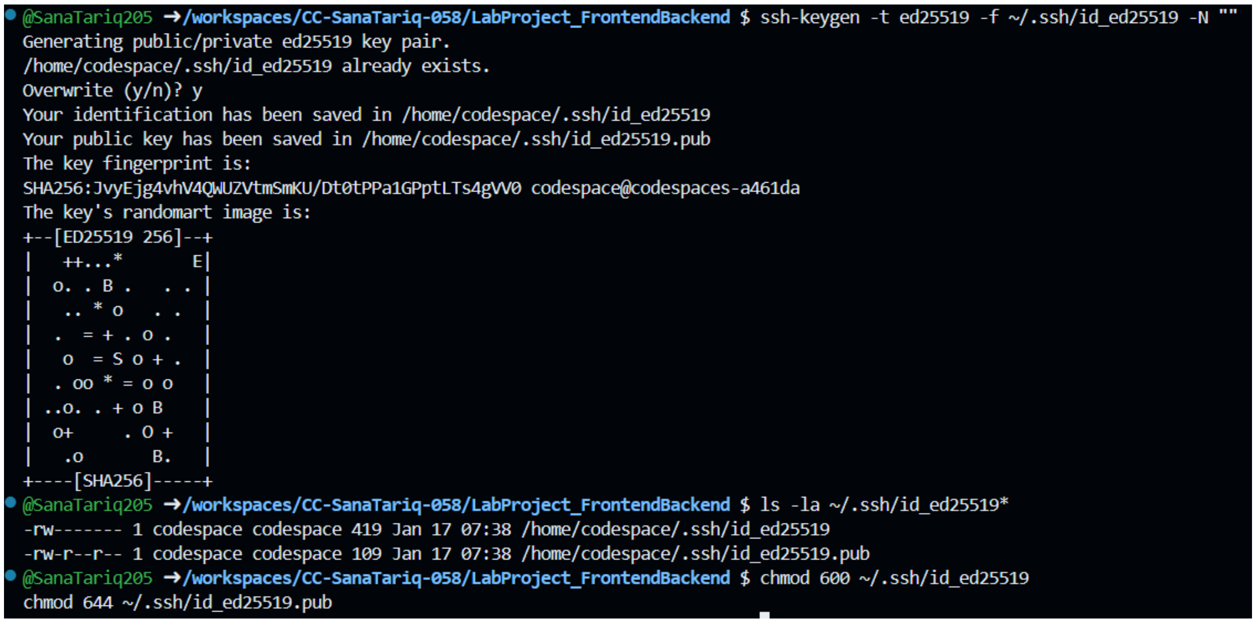
PHASE 2: AWS Configuration

Step 4: Configure AWS Credentials



```
● @SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ aws configure
  AWS Access Key ID [****************NJUB]:
  AWS Secret Access Key [****************zGVz]:
  Default region name [us-east-1]:
  Default output format [json]:
```

Step 5: Generate SSH Key Pair

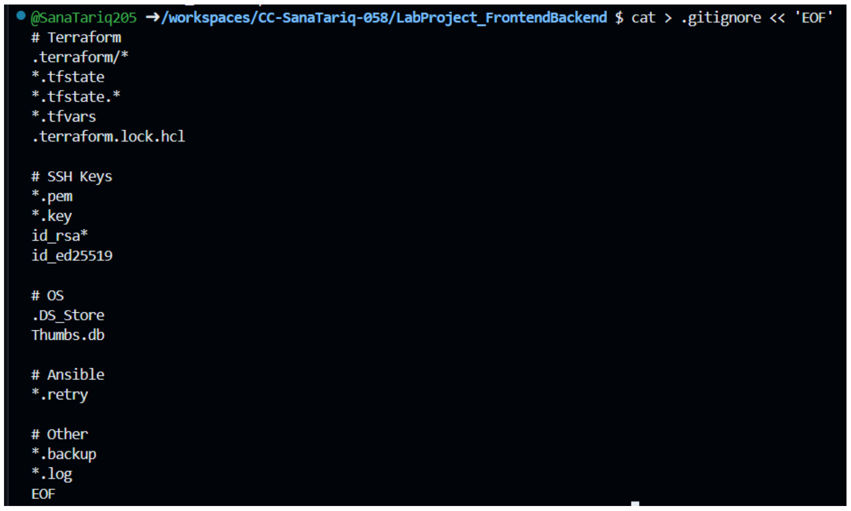


```
● @SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ ssh-keygen -t ed25519 -f ~/.ssh/id_ed25519 -N ""
  Generating public/private ed25519 key pair.
  /home/codespace/.ssh/id_ed25519 already exists.
  Overwrite (y/n)? y
  Your identification has been saved in /home/codespace/.ssh/id_ed25519
  Your public key has been saved in /home/codespace/.ssh/id_ed25519.pub
  The key fingerprint is:
  SHA256:JvyEjg4vhV4QWUZVtmSmKU/Dt0tPPa1GPptLTs4gVV0 codespace@codespaces-a461da
  The key's randomart image is:
  +--[ED25519 256]--+
  |   ++...*       E|
  |  o. . B .   . . |
  |   .. * o   . .  |
  |  .  = + . o .   |
  |   o  = S o + .  |
  |  . oo * = o o   |
  | ..o. . + o B    |
  |  o+     . O +   |
  |   .o       B.   |
  +----[SHA256]-----+
● @SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ ls -la ~/.ssh/id_ed25519*
  -rw------- 1 codespace codespace 419 Jan 17 07:38 /home/codespace/.ssh/id_ed25519
  -rw-r--r-- 1 codespace codespace 109 Jan 17 07:38 /home/codespace/.ssh/id_ed25519.pub
● @SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ chmod 600 ~/.ssh/id_ed25519
  chmod 644 ~/.ssh/id_ed25519.pub
```

PHASE 3: Create Core Terraform Files (90 minutes)

Step 6: Create `.gitignore`

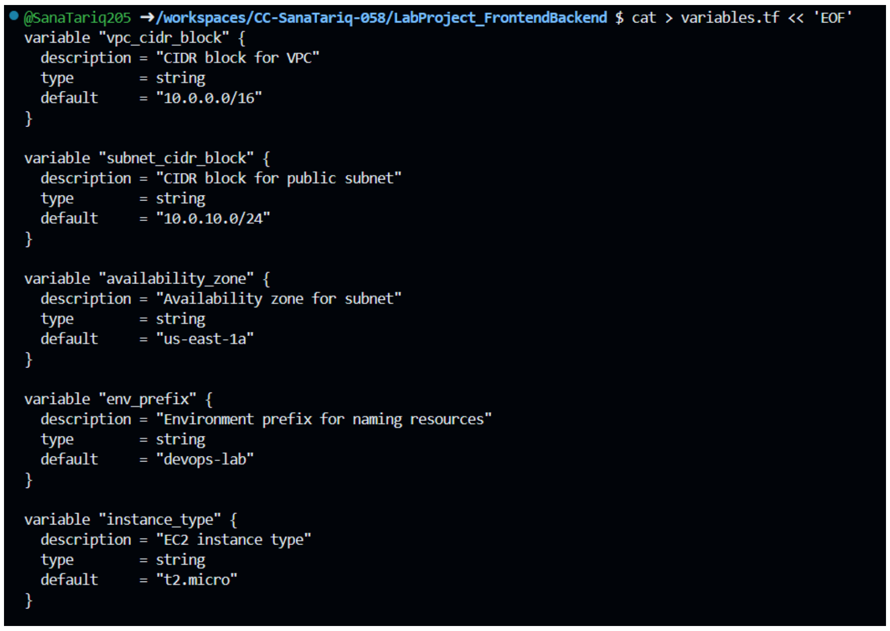
Location:
`/workspaces/CC_<YourName>_<YourRollNumber>/LabProject_FrontendBackend/.gitignore`



```
@SanaTariq205 →/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ cat > .gitignore << 'EOF'
# Terraform
.terraform/*
*.tfstate
*.tfstate.*
*.tfvars
.terraform.lock.hcl

# SSH Keys
*.pem
*.key
id_rsa*
id_ed25519

# OS
.DS_Store
Thumbs.db

# Ansible
*.retry

# Other
*.backup
*.log
EOF
```

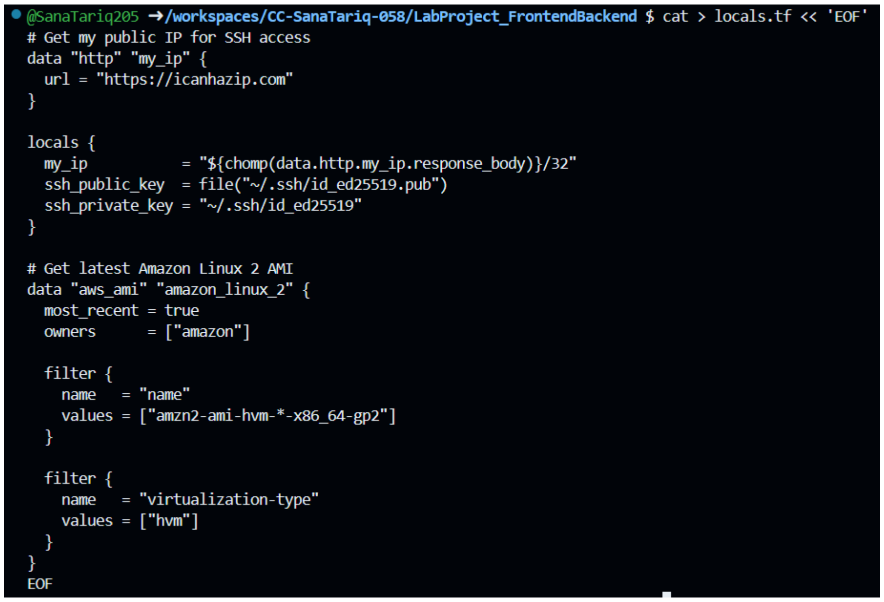
Step 7: Create `variables.tf`

Location: Same directory, create file:



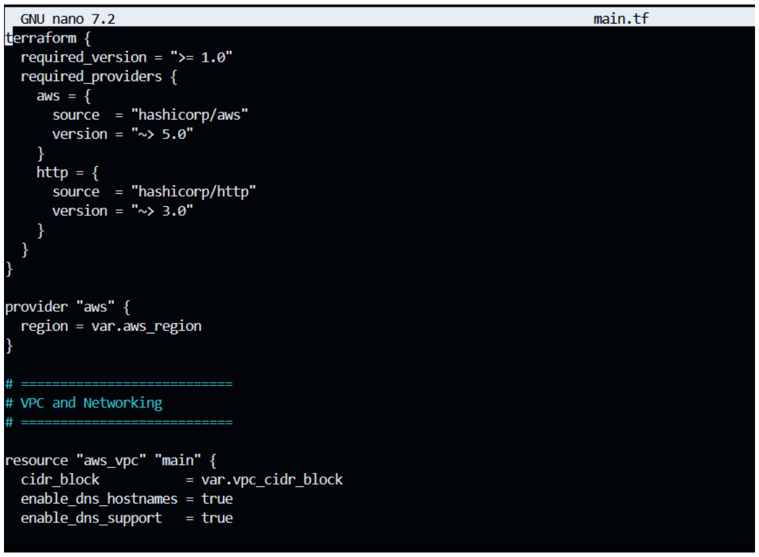
```
@SanaTariq205 →/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ cat > variables.tf << 'EOF'
variable "vpc_cidr_block" {
  description = "CIDR block for VPC"
  type        = string
  default     = "10.0.0.0/16"
}

variable "subnet_cidr_block" {
  description = "CIDR block for public subnet"
  type        = string
  default     = "10.0.10.0/24"
}

variable "availability_zone" {
  description = "Availability zone for subnet"
  type        = string
  default     = "us-east-1a"
}

variable "env_prefix" {
  description = "Environment prefix for naming resources"
  type        = string
  default     = "devops-lab"
}

variable "instance_type" {
  description = "EC2 instance type"
  type        = string
  default     = "t2.micro"
}
```

Step 8: Create `locals.tf`



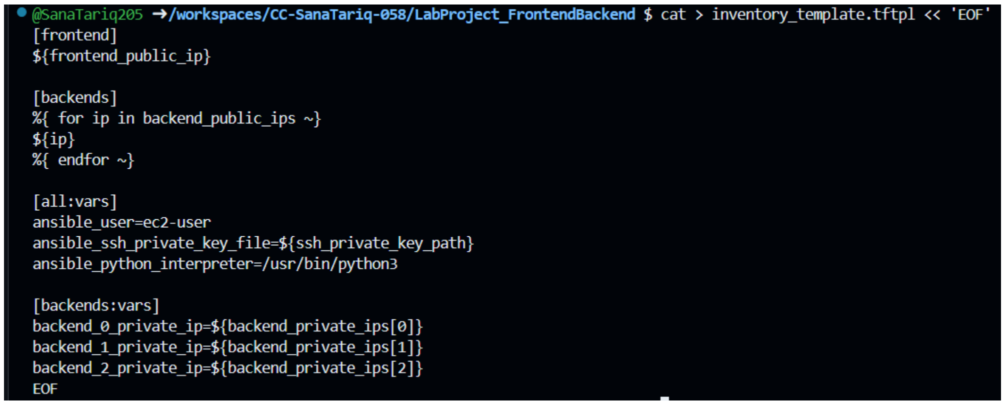
```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ cat > locals.tf << 'EOF'
  # Get my public IP for SSH access
  data "http" "my_ip" {
    url = "https://icanhazip.com"
  }

  locals {
    my_ip           = "${chomp(data.http.my_ip.response_body)}/32"
    ssh_public_key  = file("~/.ssh/id_ed25519.pub")
    ssh_private_key = "~/.ssh/id_ed25519"
  }

  # Get latest Amazon Linux 2 AMI
  data "aws_ami" "amazon_linux_2" {
    most_recent = true
    owners      = ["amazon"]

    filter {
      name   = "name"
      values = ["amzn2-ami-hvm-*-x86_64-gp2"]
    }

    filter {
      name   = "virtualization-type"
      values = ["hvm"]
    }
  }
  EOF
```

Step 9: Create `main.tf`



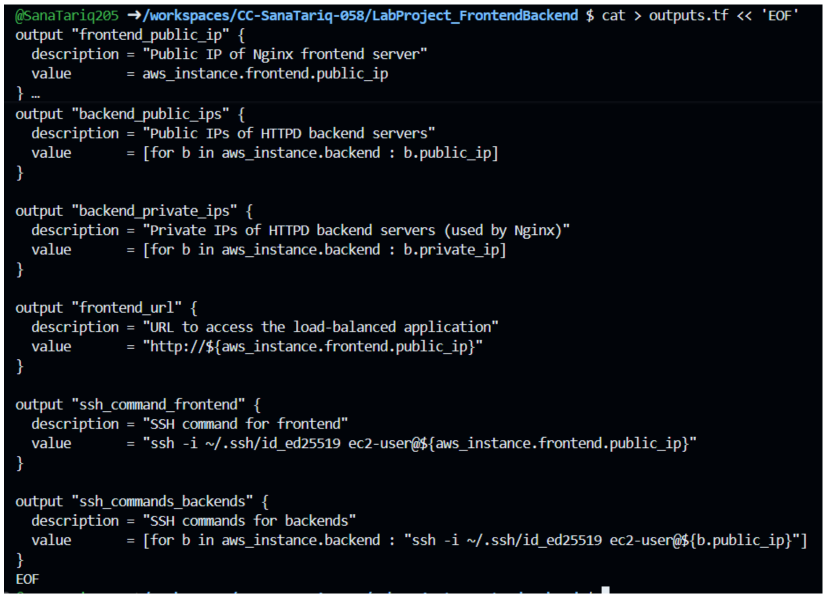
```
  GNU nano 7.2                                                  main.tf
terraform {
  required_version = ">= 1.0"
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
    http = {
      source  = "hashicorp/http"
      version = "~> 3.0"
    }
  }
}

provider "aws" {
  region = var.aws_region
}

# ==========================
# VPC and Networking
# ==========================

resource "aws_vpc" "main" {
  cidr_block           = var.vpc_cidr_block
  enable_dns_hostnames = true
  enable_dns_support   = true
```

Step 10: Create Inventory Template



```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ cat > inventory_template.tftpl << 'EOF'
  [frontend]
  ${frontend_public_ip}

  [backends]
  %{ for ip in backend_public_ips ~}
  ${ip}
  %{ endfor ~}

  [all:vars]
  ansible_user=ec2-user
  ansible_ssh_private_key_file=${ssh_private_key_path}
  ansible_python_interpreter=/usr/bin/python3

  [backends:vars]
  backend_0_private_ip=${backend_private_ips[0]}
  backend_1_private_ip=${backend_private_ips[1]}
  backend_2_private_ip=${backend_private_ips[2]}
  EOF
```

Step 11: Create `outputs.tf`

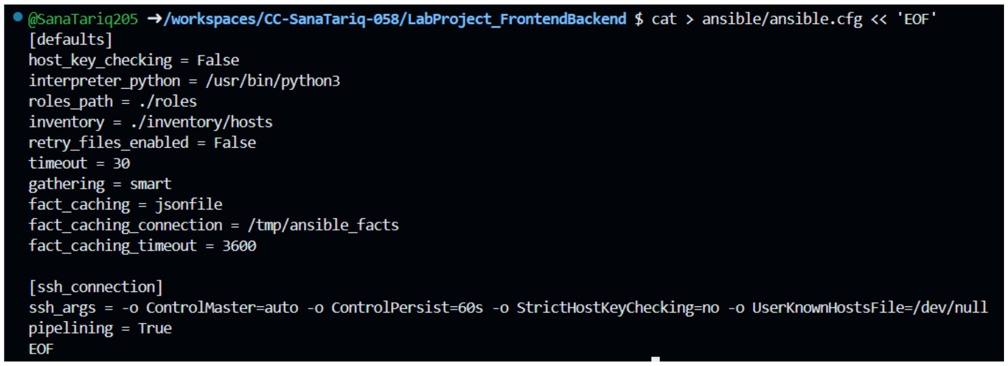


```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ cat > outputs.tf << 'EOF'
output "frontend_public_ip" {
  description = "Public IP of Nginx frontend server"
  value       = aws_instance.frontend.public_ip
} …
output "backend_public_ips" {
  description = "Public IPs of HTTPD backend servers"
  value       = [for b in aws_instance.backend : b.public_ip]
}

output "backend_private_ips" {
  description = "Private IPs of HTTPD backend servers (used by Nginx)"
  value       = [for b in aws_instance.backend : b.private_ip]
}

output "frontend_url" {
  description = "URL to access the load-balanced application"
  value       = "http://${aws_instance.frontend.public_ip}"
}

output "ssh_command_frontend" {
  description = "SSH command for frontend"
  value       = "ssh -i ~/.ssh/id_ed25519 ec2-user@${aws_instance.frontend.public_ip}"
}

output "ssh_commands_backends" {
  description = "SSH commands for backends"
  value       = [for b in aws_instance.backend : "ssh -i ~/.ssh/id_ed25519 ec2-user@${b.public_ip}"]
}
EOF
```

PHASE 4: Create Ansible Configuration

Step 12: Create **`ansible.cfg`**

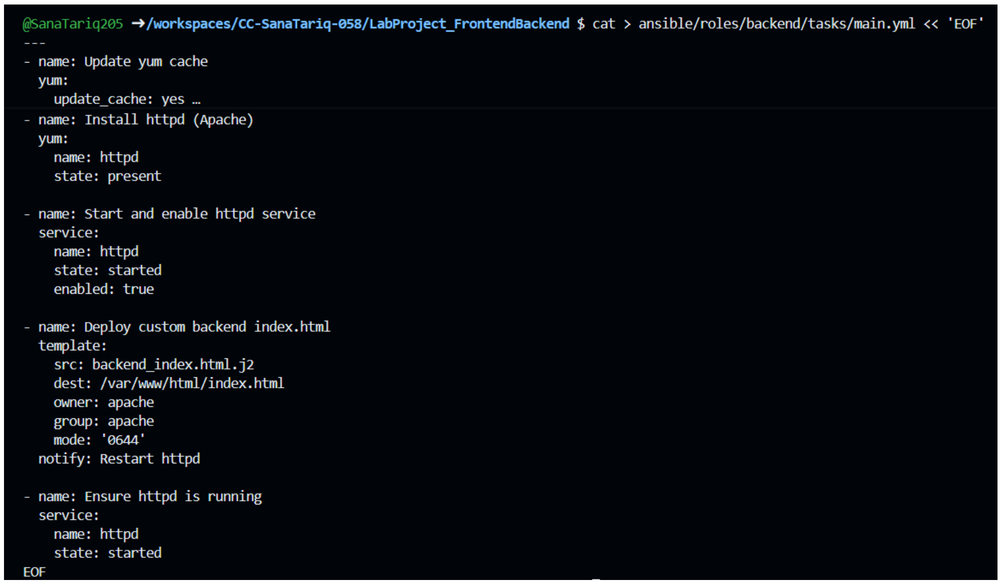
Location: `ansible/ansible.cfg`



```
● @SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ cat > ansible/ansible.cfg << 'EOF'
  [defaults]
  host_key_checking = False
  interpreter_python = /usr/bin/python3
  roles_path = ./roles
  inventory = ./inventory/hosts
  retry_files_enabled = False
  timeout = 30
  gathering = smart
  fact_caching = jsonfile
  fact_caching_connection = /tmp/ansible_facts
  fact_caching_timeout = 3600

  [ssh_connection]
  ssh_args = -o ControlMaster=auto -o ControlPersist=60s -o StrictHostKeyChecking=no -o UserKnownHostsFile=/dev/null
  pipelining = True
  EOF
```

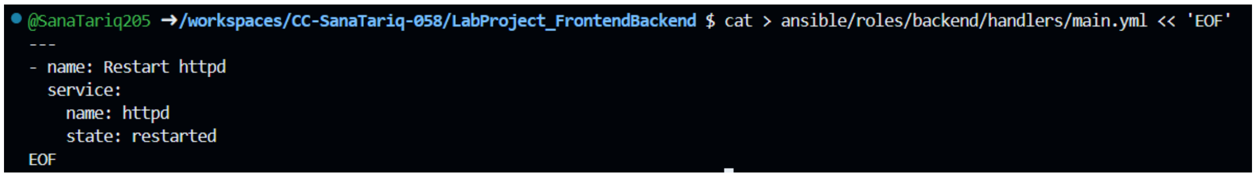
Step 13: Create Backend Role

File: `ansible/roles/backend/tasks/main.yml`



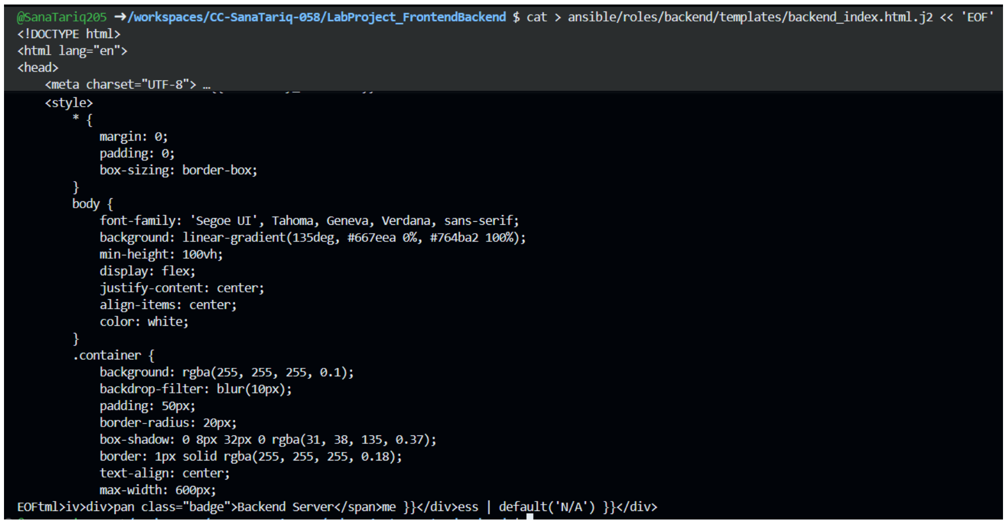
```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ cat > ansible/roles/backend/tasks/main.yml << 'EOF'
---
- name: Update yum cache
  yum:
    update_cache: yes …
- name: Install httpd (Apache)
  yum:
    name: httpd
    state: present

- name: Start and enable httpd service
  service:
    name: httpd
    state: started
    enabled: true

- name: Deploy custom backend index.html
  template:
    src: backend_index.html.j2
    dest: /var/www/html/index.html
    owner: apache
    group: apache
    mode: '0644'
  notify: Restart httpd

- name: Ensure httpd is running
  service:
    name: httpd
    state: started
EOF
```

File: `ansible/roles/backend/handlers/main.yml`



```
● @SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ cat > ansible/roles/backend/handlers/main.yml << 'EOF'
 ---
 - name: Restart httpd
   service:
     name: httpd
     state: restarted
 EOF
```

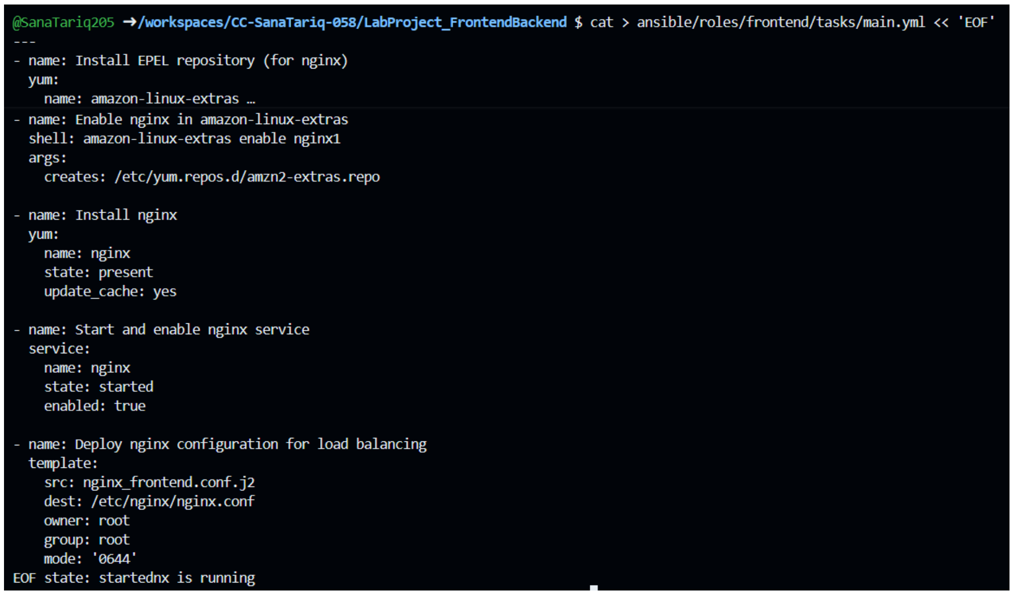
File: `ansible/roles/backend/templates/backend_index.html.j2`



```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ cat > ansible/roles/backend/templates/backend_index.html.j2 << 'EOF'
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8"> …
    <style>
        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }
        body {
            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
            background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
            min-height: 100vh;
            display: flex;
            justify-content: center;
            align-items: center;
            color: white;
        }
        .container {
            background: rgba(255, 255, 255, 0.1);
            backdrop-filter: blur(10px);
            padding: 50px;
            border-radius: 20px;
            box-shadow: 0 8px 32px 0 rgba(31, 38, 135, 0.37);
            border: 1px solid rgba(255, 255, 255, 0.18);
            text-align: center;
            max-width: 600px;
EOFtml>iv>div>pan class="badge">Backend Server</span>me }}</div>ess | default('N/A') }}</div>
```

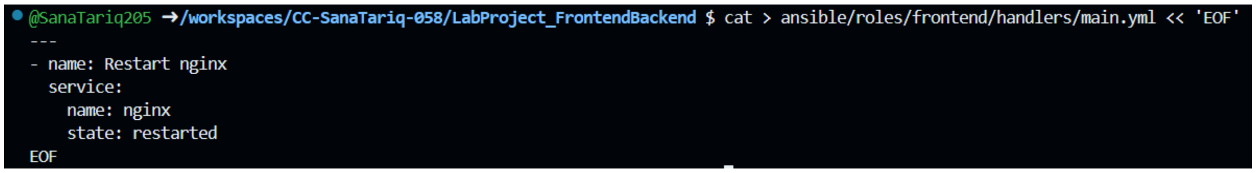
Step 14: Create Frontend Role

File: `ansible/roles/frontend/tasks/main.yml`



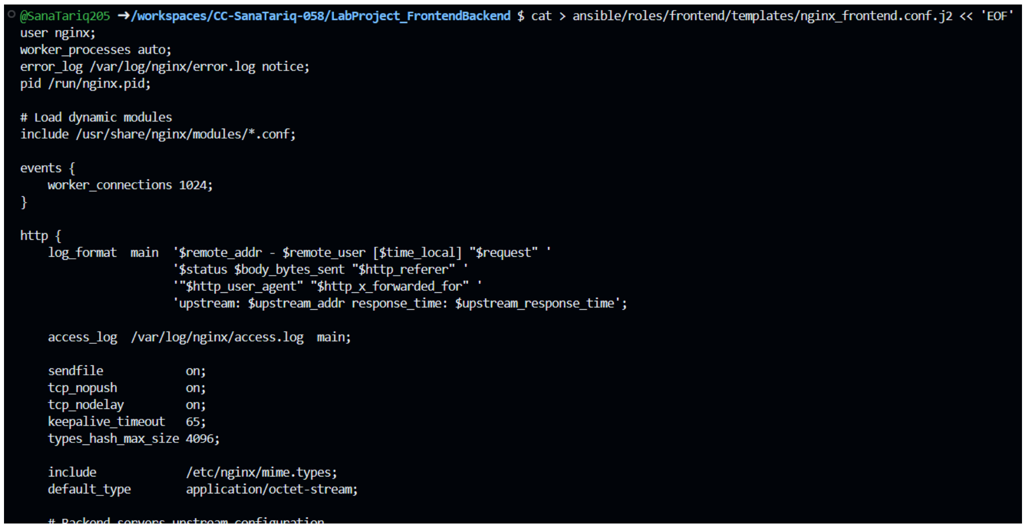
```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ cat > ansible/roles/frontend/tasks/main.yml << 'EOF'
---
- name: Install EPEL repository (for nginx)
  yum:
    name: amazon-linux-extras …
- name: Enable nginx in amazon-linux-extras
  shell: amazon-linux-extras enable nginx1
  args:
    creates: /etc/yum.repos.d/amzn2-extras.repo

- name: Install nginx
  yum:
    name: nginx
    state: present
    update_cache: yes

- name: Start and enable nginx service
  service:
    name: nginx
    state: started
    enabled: true

- name: Deploy nginx configuration for load balancing
  template:
    src: nginx_frontend.conf.j2
    dest: /etc/nginx/nginx.conf
    owner: root
    group: root
    mode: '0644'
EOF state: startednx is running
```

File: `ansible/roles/frontend/handlers/main.yml`



```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ cat > ansible/roles/frontend/handlers/main.yml << 'EOF'
---
- name: Restart nginx
  service:
    name: nginx
    state: restarted
EOF
```

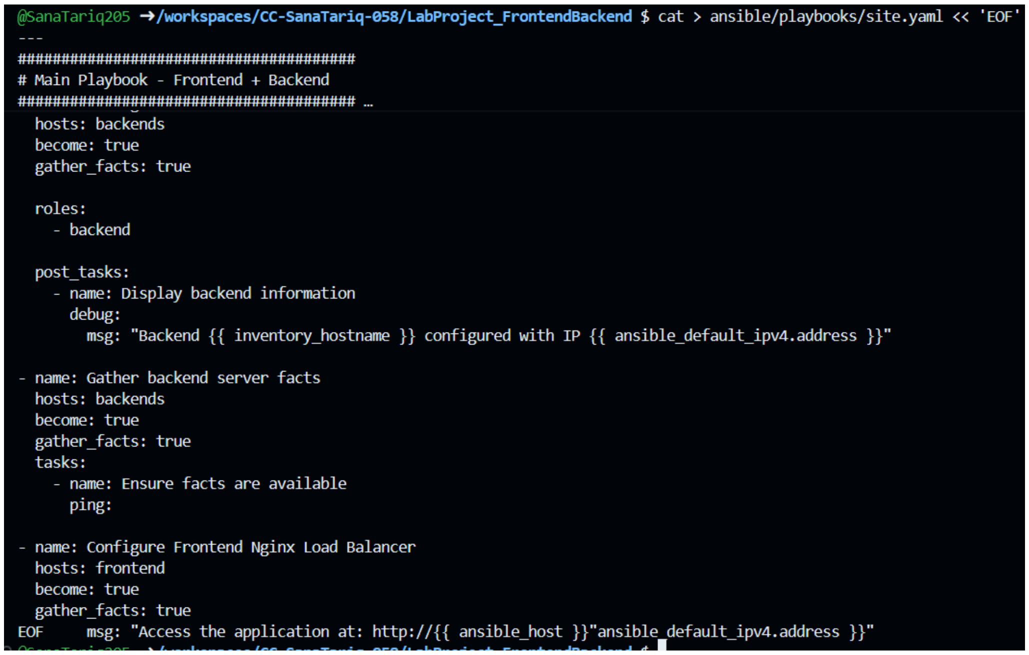
File: `ansible/roles/frontend/templates/nginx_frontend.conf.j2`



```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ cat > ansible/roles/frontend/templates/nginx_frontend.conf.j2 << 'EOF'
user nginx;
worker_processes auto;
error_log /var/log/nginx/error.log notice;
pid /run/nginx.pid;

# Load dynamic modules
include /usr/share/nginx/modules/*.conf;

events {
    worker_connections 1024;
}

http {
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for" '
                      'upstream: $upstream_addr response_time: $upstream_response_time';

    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;

    # Backend servers upstream configuration
```

Step 15: Create Main Ansible Playbook

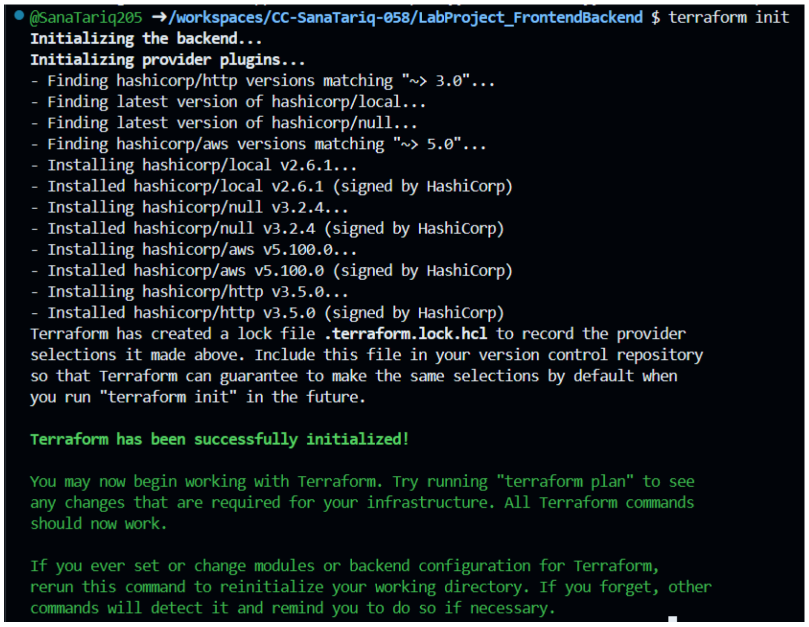
File: `ansible/playbooks/site.yaml`



```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ cat > ansible/playbooks/site.yaml << 'EOF'
---
########################################
# Main Playbook - Frontend + Backend
######################################## …
  hosts: backends
  become: true
  gather_facts: true

  roles:
    - backend

  post_tasks:
    - name: Display backend information
      debug:
        msg: "Backend {{ inventory_hostname }} configured with IP {{ ansible_default_ipv4.address }}"

- name: Gather backend server facts
  hosts: backends
  become: true
  gather_facts: true
  tasks:
    - name: Ensure facts are available
      ping:

- name: Configure Frontend Nginx Load Balancer
  hosts: frontend
  become: true
  gather_facts: true
EOF     msg: "Access the application at: http://{{ ansible_host }}"ansible_default_ipv4.address }}"
```

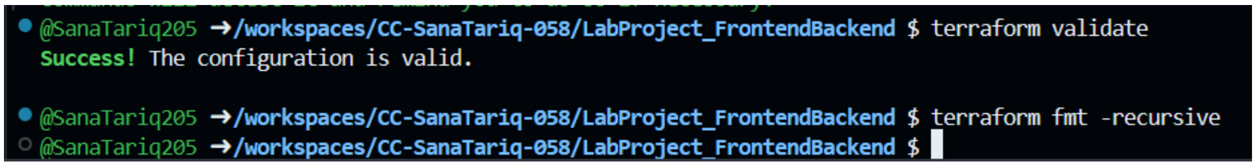
PHASE 5: Deploy everything

Step 16: Initialize Terraform



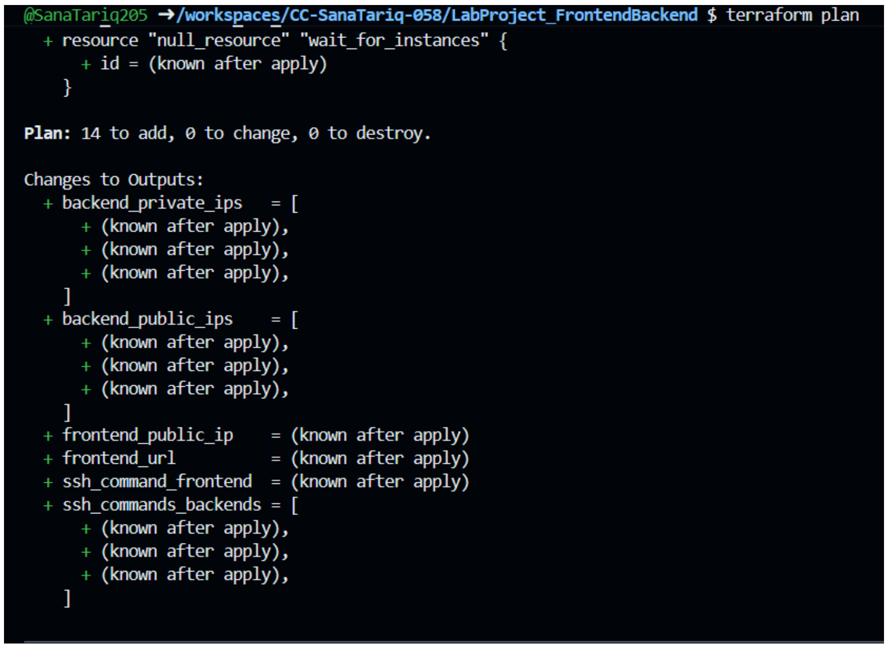
```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/http versions matching "~> 3.0"...
- Finding latest version of hashicorp/local...
- Finding latest version of hashicorp/null...
- Finding hashicorp/aws versions matching "~> 5.0"...
- Installing hashicorp/local v2.6.1...
- Installed hashicorp/local v2.6.1 (signed by HashiCorp)
- Installing hashicorp/null v3.2.4...
- Installed hashicorp/null v3.2.4 (signed by HashiCorp)
- Installing hashicorp/aws v5.100.0...
- Installed hashicorp/aws v5.100.0 (signed by HashiCorp)
- Installing hashicorp/http v3.5.0...
- Installed hashicorp/http v3.5.0 (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.
```

Step 17: Validate Configuration



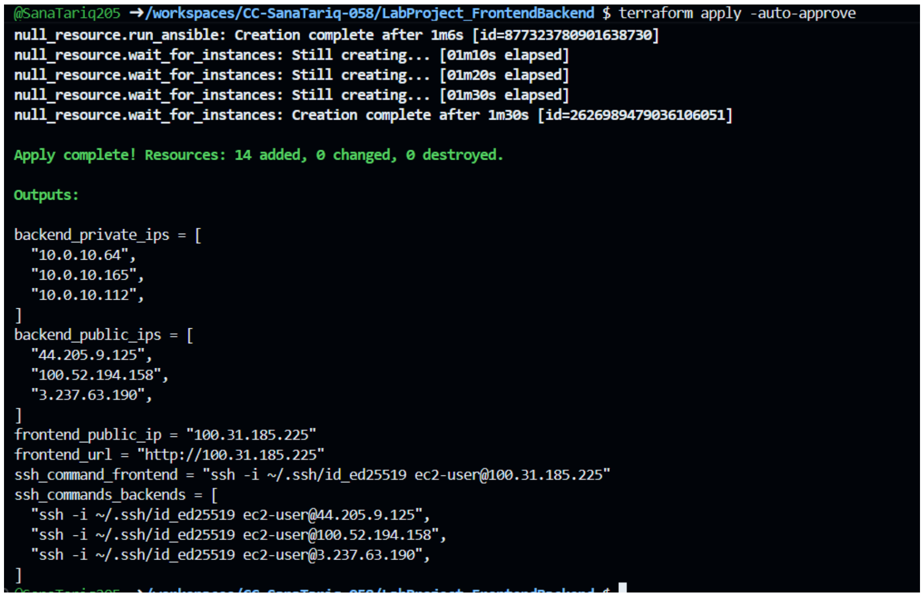
```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ terraform validate
Success! The configuration is valid.

@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ terraform fmt -recursive
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $
```

Step 18: Plan



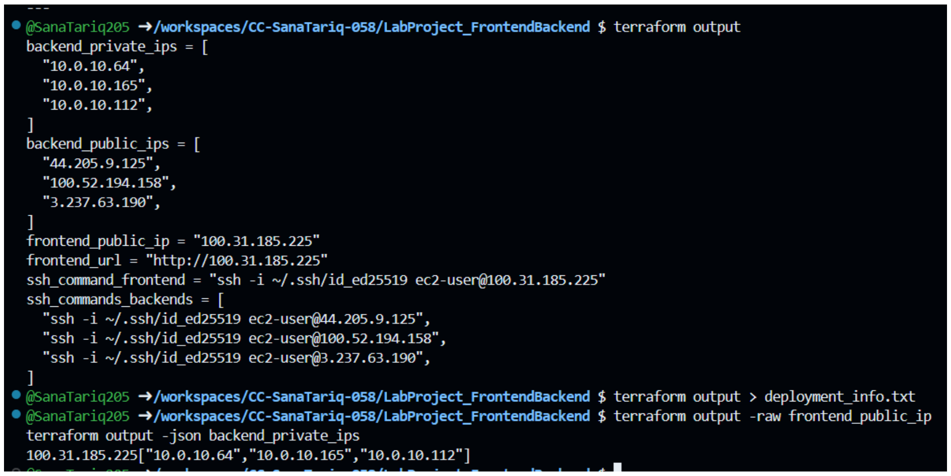
```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ terraform plan
  + resource "null_resource" "wait_for_instances" {
      + id = (known after apply)
    }

Plan: 14 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + backend_private_ips   = [
      + (known after apply),
      + (known after apply),
      + (known after apply),
    ]
  + backend_public_ips    = [
      + (known after apply),
      + (known after apply),
      + (known after apply),
    ]
  + frontend_public_ip    = (known after apply)
  + frontend_url          = (known after apply)
  + ssh_command_frontend  = (known after apply)
  + ssh_commands_backends = [
      + (known after apply),
      + (known after apply),
      + (known after apply),
    ]
```

Step 19: Apply and Deploy EVERYTHING



```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ terraform apply -auto-approve
null_resource.run_ansible: Creation complete after 1m6s [id=877323780901638730]
null_resource.wait_for_instances: Still creating... [01m10s elapsed]
null_resource.wait_for_instances: Still creating... [01m20s elapsed]
null_resource.wait_for_instances: Still creating... [01m30s elapsed]
null_resource.wait_for_instances: Creation complete after 1m30s [id=2626989479036106051]

Apply complete! Resources: 14 added, 0 changed, 0 destroyed.

Outputs:

backend_private_ips = [
  "10.0.10.64",
  "10.0.10.165",
  "10.0.10.112",
]
backend_public_ips = [
  "44.205.9.125",
  "100.52.194.158",
  "3.237.63.190",
]
frontend_public_ip = "100.31.185.225"
frontend_url = "http://100.31.185.225"
ssh_command_frontend = "ssh -i ~/.ssh/id_ed25519 ec2-user@100.31.185.225"
ssh_commands_backends = [
  "ssh -i ~/.ssh/id_ed25519 ec2-user@44.205.9.125",
  "ssh -i ~/.ssh/id_ed25519 ec2-user@100.52.194.158",
  "ssh -i ~/.ssh/id_ed25519 ec2-user@3.237.63.190",
]
```

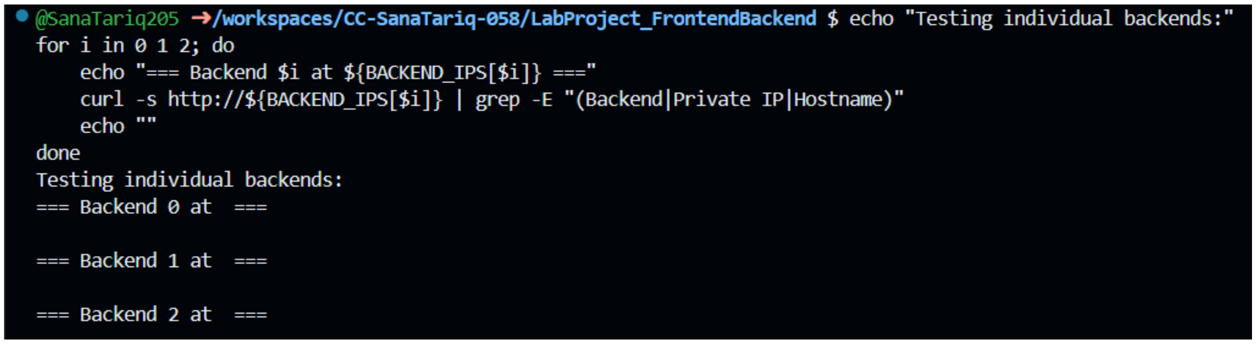
Step 20: Save Outputs



```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ terraform output
backend_private_ips = [
  "10.0.10.64",
  "10.0.10.165",
  "10.0.10.112",
]
backend_public_ips = [
  "44.205.9.125",
  "100.52.194.158",
  "3.237.63.190",
]
frontend_public_ip = "100.31.185.225"
frontend_url = "http://100.31.185.225"
ssh_command_frontend = "ssh -i ~/.ssh/id_ed25519 ec2-user@100.31.185.225"
ssh_commands_backends = [
  "ssh -i ~/.ssh/id_ed25519 ec2-user@44.205.9.125",
  "ssh -i ~/.ssh/id_ed25519 ec2-user@100.52.194.158",
  "ssh -i ~/.ssh/id_ed25519 ec2-user@3.237.63.190",
]
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ terraform output > deployment_info.txt
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ terraform output -raw frontend_public_ip
terraform output -json backend_private_ips
100.31.185.225["10.0.10.64","10.0.10.165","10.0.10.112"]
```

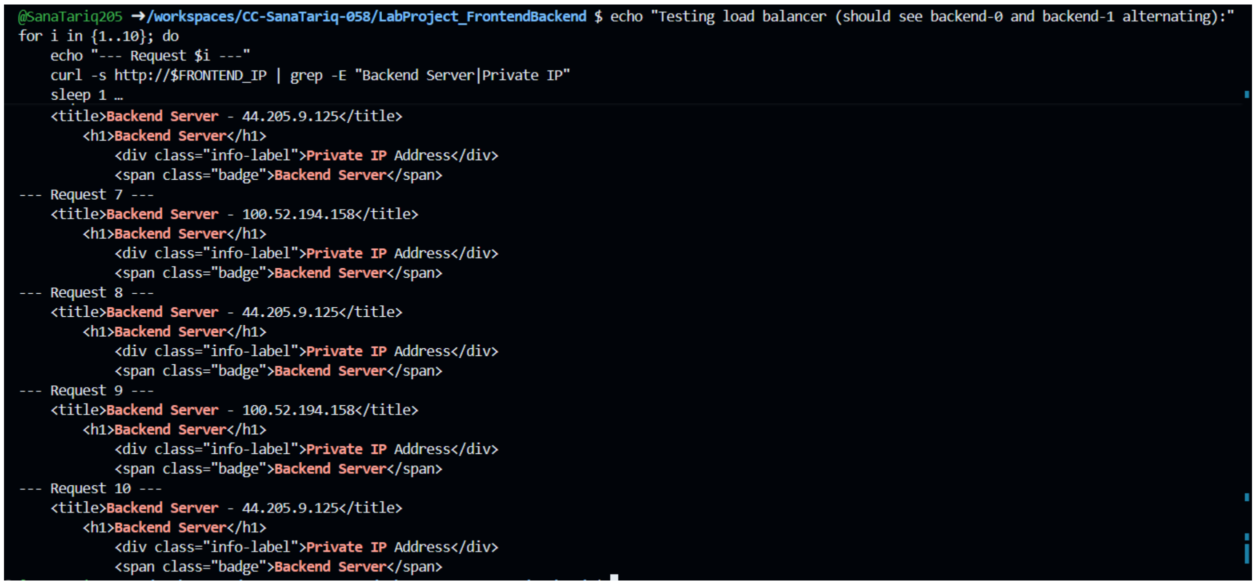
PHASE 6: Testing and Verification

Step 21: Test Individual Backends



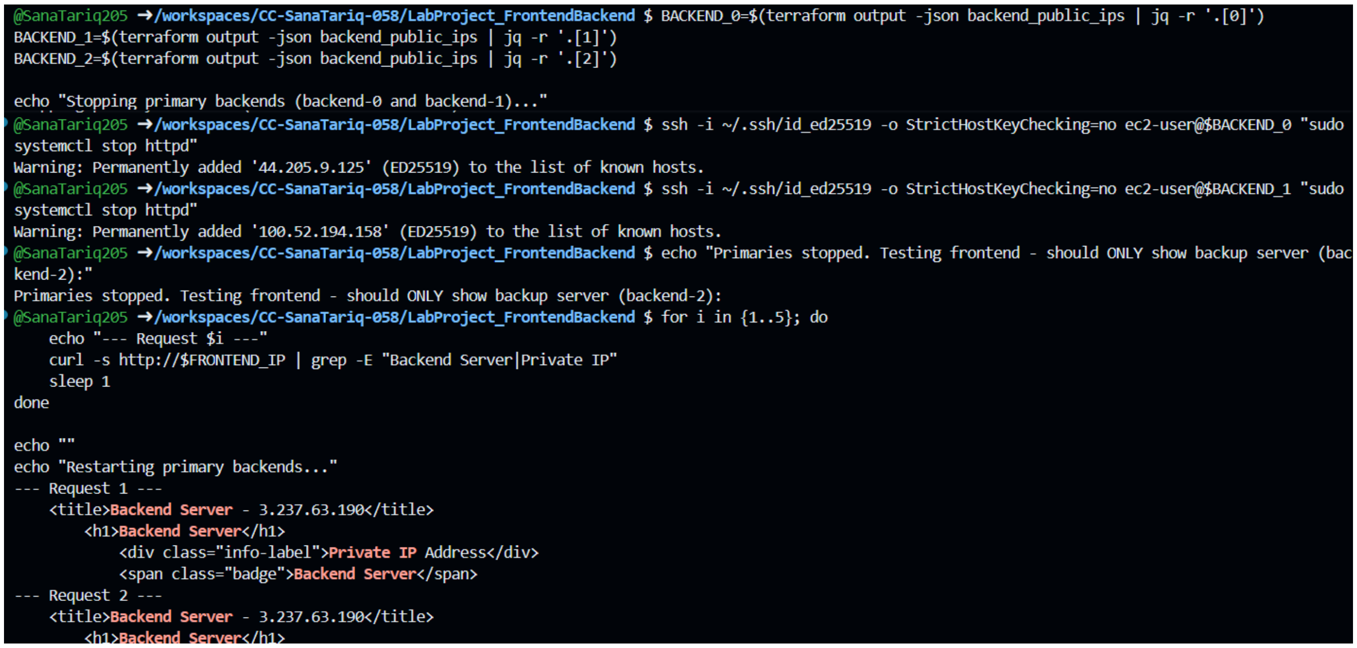
```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ echo "Testing individual backends:"
for i in 0 1 2; do
    echo "=== Backend $i at ${BACKEND_IPS[$i]} ==="
    curl -s http://${BACKEND_IPS[$i]} | grep -E "(Backend|Private IP|Hostname)"
    echo ""
done
Testing individual backends:
=== Backend 0 at  ===

=== Backend 1 at  ===

=== Backend 2 at  ===
```

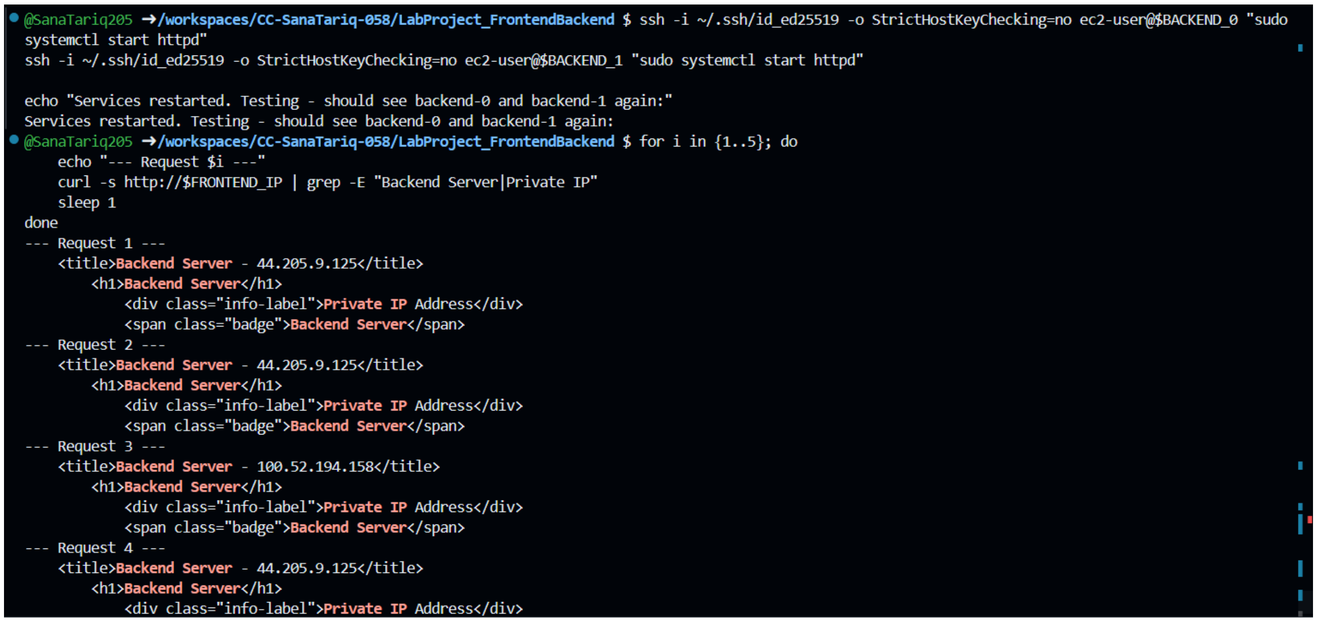
Step 22: Test Frontend Load Balancer (Normal Operation)



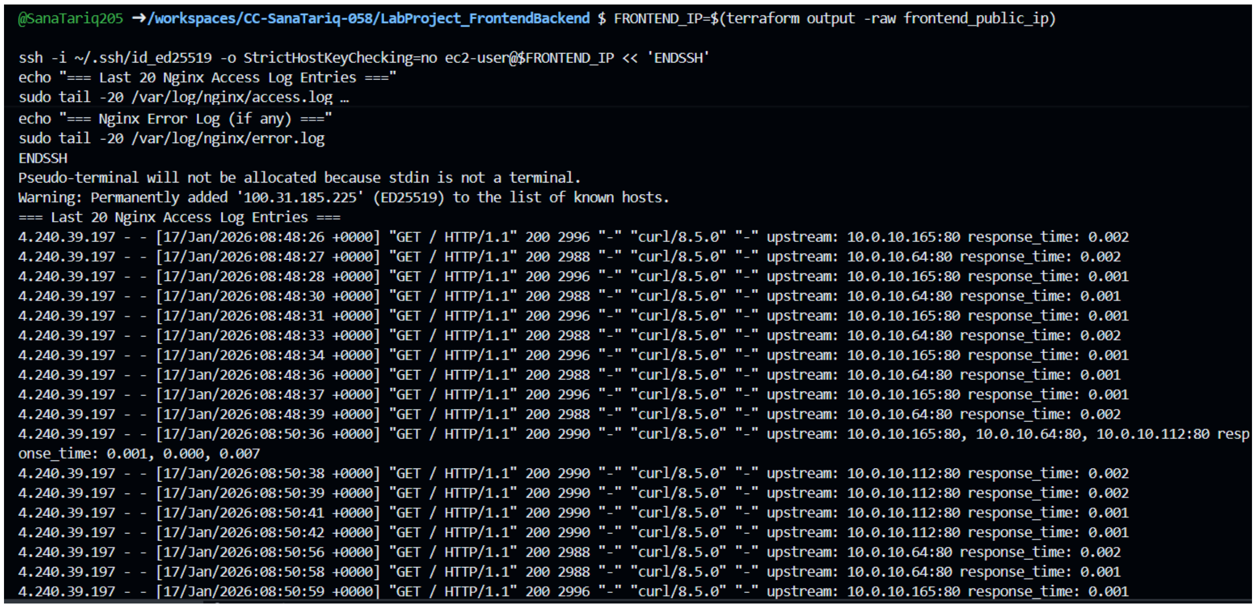
```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ echo "Testing load balancer (should see backend-0 and backend-1 alternating):"
for i in {1..10}; do
    echo "--- Request $i ---"
    curl -s http://$FRONTEND_IP | grep -E "Backend Server|Private IP"
    sleep 1 …
    <title>Backend Server - 44.205.9.125</title>
        <h1>Backend Server</h1>
            <div class="info-label">Private IP Address</div>
            <span class="badge">Backend Server</span>
--- Request 7 ---
    <title>Backend Server - 100.52.194.158</title>
        <h1>Backend Server</h1>
            <div class="info-label">Private IP Address</div>
            <span class="badge">Backend Server</span>
--- Request 8 ---
    <title>Backend Server - 44.205.9.125</title>
        <h1>Backend Server</h1>
            <div class="info-label">Private IP Address</div>
            <span class="badge">Backend Server</span>
--- Request 9 ---
    <title>Backend Server - 100.52.194.158</title>
        <h1>Backend Server</h1>
            <div class="info-label">Private IP Address</div>
            <span class="badge">Backend Server</span>
--- Request 10 ---
    <title>Backend Server - 44.205.9.125</title>
        <h1>Backend Server</h1>
            <div class="info-label">Private IP Address</div>
            <span class="badge">Backend Server</span>
```

Step 23: Test Backup Server Functionality


```
@SanaTariq205 →/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ BACKEND_0=$(terraform output -json backend_public_ips | jq -r '.[0]')
BACKEND_1=$(terraform output -json backend_public_ips | jq -r '.[1]')
BACKEND_2=$(terraform output -json backend_public_ips | jq -r '.[2]')

echo "Stopping primary backends (backend-0 and backend-1)..."
@SanaTariq205 →/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ ssh -i ~/.ssh/id_ed25519 -o StrictHostKeyChecking=no ec2-user@$BACKEND_0 "sudo
systemctl stop httpd"
Warning: Permanently added '44.205.9.125' (ED25519) to the list of known hosts.
@SanaTariq205 →/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ ssh -i ~/.ssh/id_ed25519 -o StrictHostKeyChecking=no ec2-user@$BACKEND_1 "sudo
systemctl stop httpd"
Warning: Permanently added '100.52.194.158' (ED25519) to the list of known hosts.
@SanaTariq205 →/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ echo "Primaries stopped. Testing frontend - should ONLY show backup server (bac
kend-2):"
Primaries stopped. Testing frontend - should ONLY show backup server (backend-2):
@SanaTariq205 →/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ for i in {1..5}; do
    echo "--- Request $i ---"
    curl -s http://$FRONTEND_IP | grep -E "Backend Server|Private IP"
    sleep 1
done

echo ""
echo "Restarting primary backends..."
--- Request 1 ---
    <title>Backend Server - 3.237.63.190</title>
        <h1>Backend Server</h1>
            <div class="info-label">Private IP Address</div>
            <span class="badge">Backend Server</span>
--- Request 2 ---
    <title>Backend Server - 3.237.63.190</title>
        <h1>Backend Server</h1>
```


```
@SanaTariq205 →/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ ssh -i ~/.ssh/id_ed25519 -o StrictHostKeyChecking=no ec2-user@$BACKEND_0 "sudo
systemctl start httpd"
ssh -i ~/.ssh/id_ed25519 -o StrictHostKeyChecking=no ec2-user@$BACKEND_1 "sudo systemctl start httpd"

echo "Services restarted. Testing - should see backend-0 and backend-1 again:"
Services restarted. Testing - should see backend-0 and backend-1 again:
@SanaTariq205 →/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ for i in {1..5}; do
    echo "--- Request $i ---"
    curl -s http://$FRONTEND_IP | grep -E "Backend Server|Private IP"
    sleep 1
done
--- Request 1 ---
    <title>Backend Server - 44.205.9.125</title>
        <h1>Backend Server</h1>
            <div class="info-label">Private IP Address</div>
            <span class="badge">Backend Server</span>
--- Request 2 ---
    <title>Backend Server - 44.205.9.125</title>
        <h1>Backend Server</h1>
            <div class="info-label">Private IP Address</div>
            <span class="badge">Backend Server</span>
--- Request 3 ---
    <title>Backend Server - 100.52.194.158</title>
        <h1>Backend Server</h1>
            <div class="info-label">Private IP Address</div>
            <span class="badge">Backend Server</span>
--- Request 4 ---
    <title>Backend Server - 44.205.9.125</title>
        <h1>Backend Server</h1>
            <div class="info-label">Private IP Address</div>
```

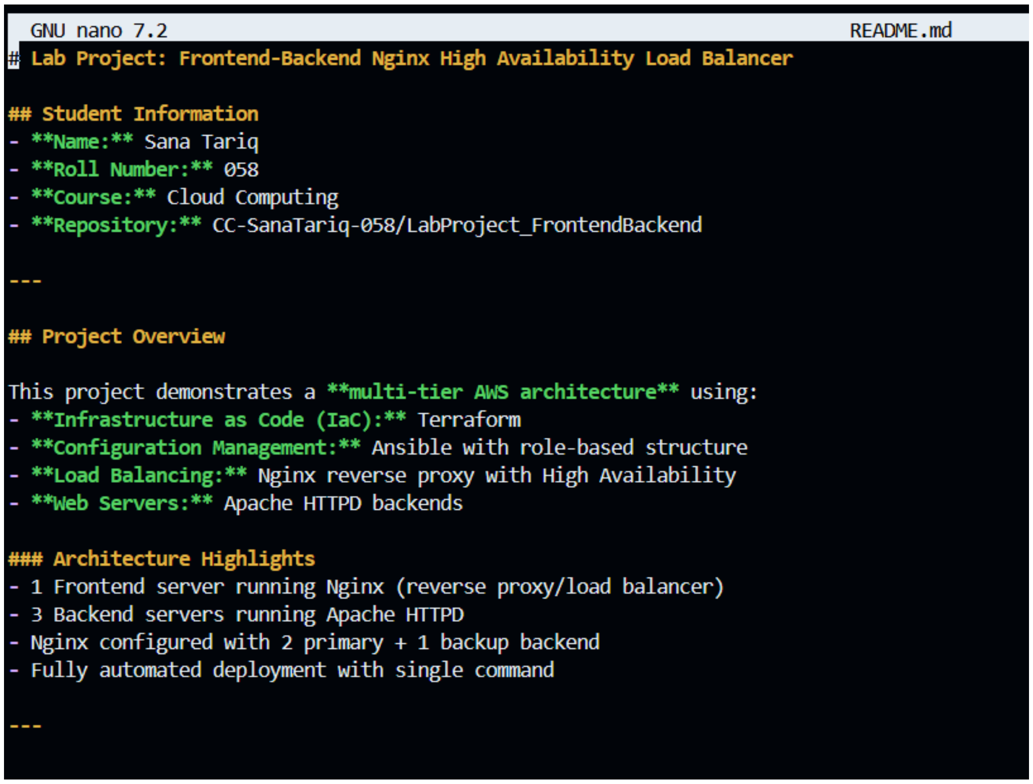
Step 24: Check Nginx Logs



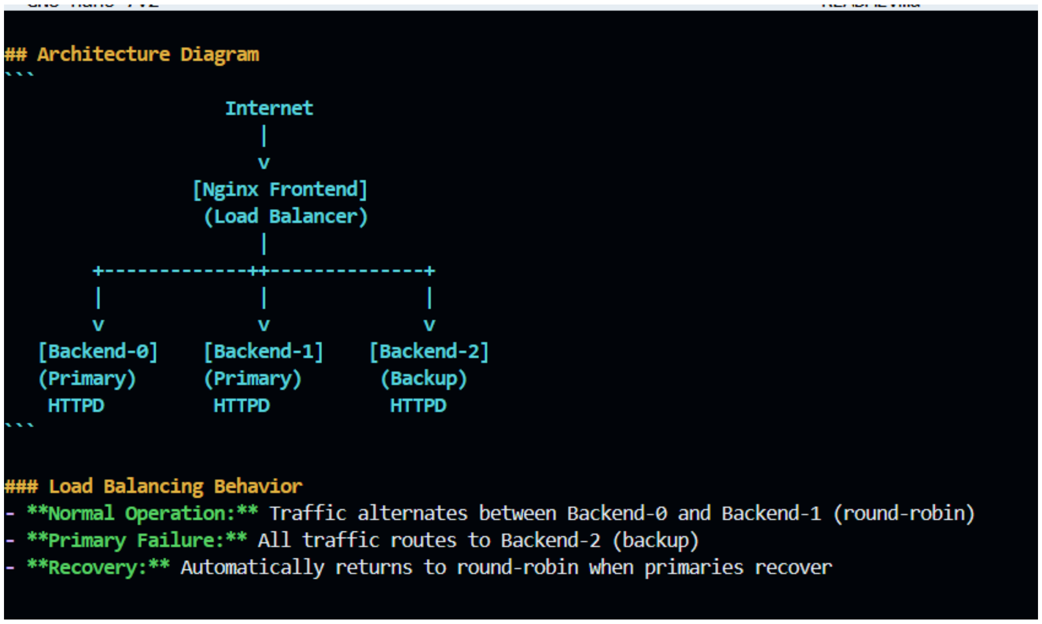
```
@SanaTariq205 ➜/workspaces/CC-SanaTariq-058/LabProject_FrontendBackend $ FRONTEND_IP=$(terraform output -raw frontend_public_ip)

ssh -i ~/.ssh/id_ed25519 -o StrictHostKeyChecking=no ec2-user@$FRONTEND_IP << 'ENDSSH'
echo "=== Last 20 Nginx Access Log Entries ==="
sudo tail -20 /var/log/nginx/access.log …
echo "=== Nginx Error Log (if any) ==="
sudo tail -20 /var/log/nginx/error.log
ENDSSH
Pseudo-terminal will not be allocated because stdin is not a terminal.
Warning: Permanently added '100.31.185.225' (ED25519) to the list of known hosts.
=== Last 20 Nginx Access Log Entries ===
4.240.39.197 - - [17/Jan/2026:08:48:26 +0000] "GET / HTTP/1.1" 200 2996 "-" "curl/8.5.0" "-" upstream: 10.0.10.165:80 response_time: 0.002
4.240.39.197 - - [17/Jan/2026:08:48:27 +0000] "GET / HTTP/1.1" 200 2988 "-" "curl/8.5.0" "-" upstream: 10.0.10.64:80 response_time: 0.002
4.240.39.197 - - [17/Jan/2026:08:48:28 +0000] "GET / HTTP/1.1" 200 2996 "-" "curl/8.5.0" "-" upstream: 10.0.10.165:80 response_time: 0.001
4.240.39.197 - - [17/Jan/2026:08:48:30 +0000] "GET / HTTP/1.1" 200 2988 "-" "curl/8.5.0" "-" upstream: 10.0.10.64:80 response_time: 0.001
4.240.39.197 - - [17/Jan/2026:08:48:31 +0000] "GET / HTTP/1.1" 200 2996 "-" "curl/8.5.0" "-" upstream: 10.0.10.165:80 response_time: 0.001
4.240.39.197 - - [17/Jan/2026:08:48:33 +0000] "GET / HTTP/1.1" 200 2988 "-" "curl/8.5.0" "-" upstream: 10.0.10.64:80 response_time: 0.002
4.240.39.197 - - [17/Jan/2026:08:48:34 +0000] "GET / HTTP/1.1" 200 2996 "-" "curl/8.5.0" "-" upstream: 10.0.10.165:80 response_time: 0.001
4.240.39.197 - - [17/Jan/2026:08:48:36 +0000] "GET / HTTP/1.1" 200 2988 "-" "curl/8.5.0" "-" upstream: 10.0.10.64:80 response_time: 0.001
4.240.39.197 - - [17/Jan/2026:08:48:37 +0000] "GET / HTTP/1.1" 200 2996 "-" "curl/8.5.0" "-" upstream: 10.0.10.165:80 response_time: 0.001
4.240.39.197 - - [17/Jan/2026:08:48:39 +0000] "GET / HTTP/1.1" 200 2988 "-" "curl/8.5.0" "-" upstream: 10.0.10.64:80 response_time: 0.002
4.240.39.197 - - [17/Jan/2026:08:50:36 +0000] "GET / HTTP/1.1" 200 2990 "-" "curl/8.5.0" "-" upstream: 10.0.10.165:80, 10.0.10.64:80, 10.0.10.112:80 resp
onse_time: 0.001, 0.000, 0.007
4.240.39.197 - - [17/Jan/2026:08:50:38 +0000] "GET / HTTP/1.1" 200 2990 "-" "curl/8.5.0" "-" upstream: 10.0.10.112:80 response_time: 0.002
4.240.39.197 - - [17/Jan/2026:08:50:39 +0000] "GET / HTTP/1.1" 200 2990 "-" "curl/8.5.0" "-" upstream: 10.0.10.112:80 response_time: 0.002
4.240.39.197 - - [17/Jan/2026:08:50:41 +0000] "GET / HTTP/1.1" 200 2990 "-" "curl/8.5.0" "-" upstream: 10.0.10.112:80 response_time: 0.001
4.240.39.197 - - [17/Jan/2026:08:50:42 +0000] "GET / HTTP/1.1" 200 2990 "-" "curl/8.5.0" "-" upstream: 10.0.10.112:80 response_time: 0.001
4.240.39.197 - - [17/Jan/2026:08:50:56 +0000] "GET / HTTP/1.1" 200 2988 "-" "curl/8.5.0" "-" upstream: 10.0.10.64:80 response_time: 0.002
4.240.39.197 - - [17/Jan/2026:08:50:58 +0000] "GET / HTTP/1.1" 200 2988 "-" "curl/8.5.0" "-" upstream: 10.0.10.64:80 response_time: 0.001
4.240.39.197 - - [17/Jan/2026:08:50:59 +0000] "GET / HTTP/1.1" 200 2996 "-" "curl/8.5.0" "-" upstream: 10.0.10.165:80 response_time: 0.001
```

PHASE 7: Documentation & Screenshots

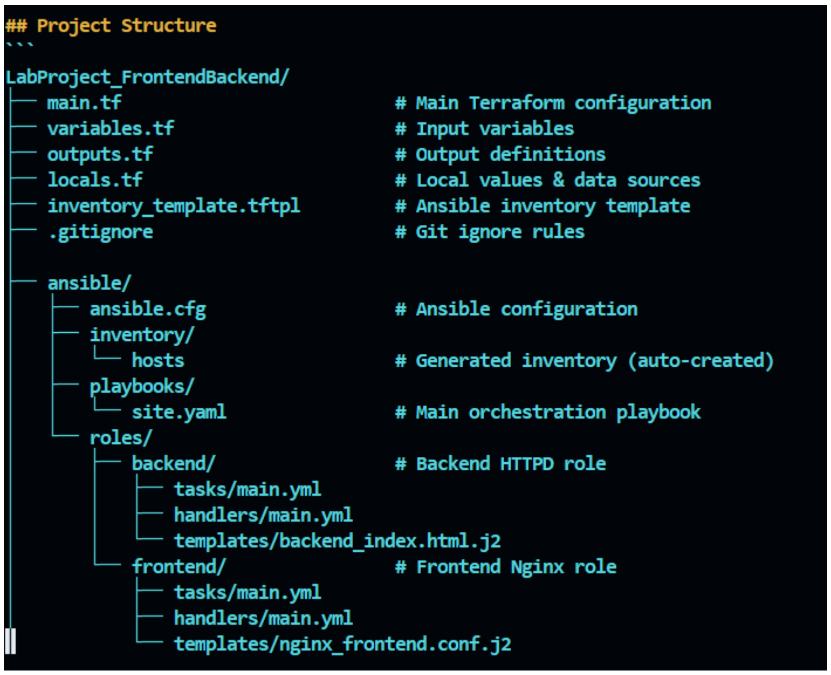
Step 25: Create README.md



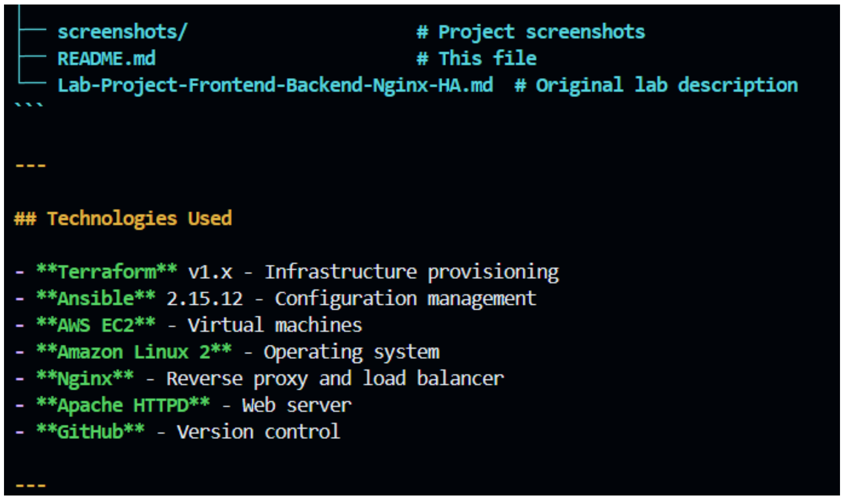
```
  GNU nano 7.2                                                        README.md
# Lab Project: Frontend-Backend Nginx High Availability Load Balancer

## Student Information
- **Name:** Sana Tariq
- **Roll Number:** 058
- **Course:** Cloud Computing
- **Repository:** CC-SanaTariq-058/LabProject_FrontendBackend

---

## Project Overview

This project demonstrates a **multi-tier AWS architecture** using:
- **Infrastructure as Code (IaC):** Terraform
- **Configuration Management:** Ansible with role-based structure
- **Load Balancing:** Nginx reverse proxy with High Availability
- **Web Servers:** Apache HTTPD backends

### Architecture Highlights
- 1 Frontend server running Nginx (reverse proxy/load balancer)
- 3 Backend servers running Apache HTTPD
- Nginx configured with 2 primary + 1 backup backend
- Fully automated deployment with single command

---
```



~~~
## Architecture Diagram
```
                    Internet
                       |
                       v
                 [Nginx Frontend]
                  (Load Balancer)
                       |
        +-------------++--------------+
        |              |              |
        v              v              v
   [Backend-0]    [Backend-1]    [Backend-2]
   (Primary)      (Primary)       (Backup)
    HTTPD          HTTPD           HTTPD
```

### Load Balancing Behavior
- **Normal Operation:** Traffic alternates between Backend-0 and Backend-1 (round-robin)
- **Primary Failure:** All traffic routes to Backend-2 (backup)
- **Recovery:** Automatically returns to round-robin when primaries recover
~~~



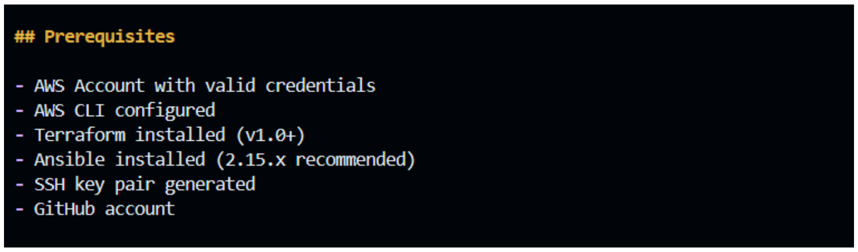
~~~
## Project Structure
```
LabProject_FrontendBackend/
├── main.tf                            # Main Terraform configuration
├── variables.tf                       # Input variables
├── outputs.tf                         # Output definitions
├── locals.tf                          # Local values & data sources
├── inventory_template.tftpl           # Ansible inventory template
├── .gitignore                         # Git ignore rules
│
├── ansible/
│   ├── ansible.cfg                    # Ansible configuration
│   ├── inventory/
│   │   └── hosts                      # Generated inventory (auto-created)
│   ├── playbooks/
│   │   └── site.yaml                  # Main orchestration playbook
│   └── roles/
│       ├── backend/                   # Backend HTTPD role
│       │   ├── tasks/main.yml
│       │   ├── handlers/main.yml
│       │   └── templates/backend_index.html.j2
│       └── frontend/                  # Frontend Nginx role
│           ├── tasks/main.yml
│           ├── handlers/main.yml
│           └── templates/nginx_frontend.conf.j2
~~~

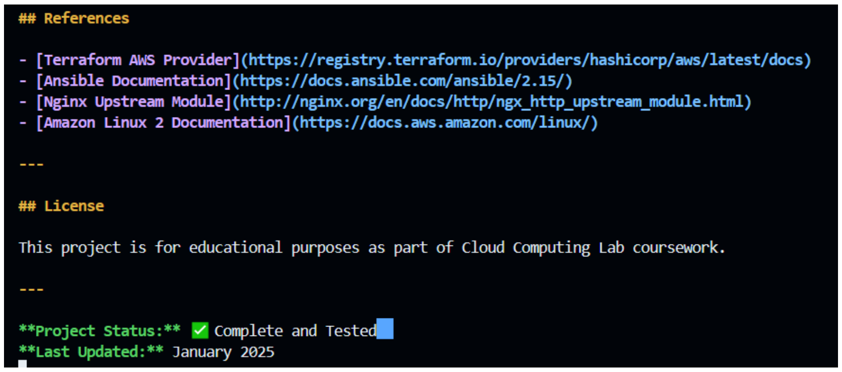


```
├── screenshots/                    # Project screenshots
├── README.md                       # This file
└── Lab-Project-Frontend-Backend-Nginx-HA.md  # Original lab description
```

Technologies Used

- **Terraform** v1.x - Infrastructure provisioning
- **Ansible** 2.15.12 - Configuration management
- **AWS EC2** - Virtual machines
- **Amazon Linux 2** - Operating system
- **Nginx** - Reverse proxy and load balancer
- **Apache HTTPD** - Web server
- **GitHub** - Version control

```



```
## Prerequisites

- AWS Account with valid credentials
- AWS CLI configured
- Terraform installed (v1.0+)
- Ansible installed (2.15.x recommended)
- SSH key pair generated
- GitHub account
```



```
## References

- [Terraform AWS Provider](https://registry.terraform.io/providers/hashicorp/aws/latest/docs)
- [Ansible Documentation](https://docs.ansible.com/ansible/2.15/)
- [Nginx Upstream Module](http://nginx.org/en/docs/http/ngx_http_upstream_module.html)
- [Amazon Linux 2 Documentation](https://docs.aws.amazon.com/linux/)

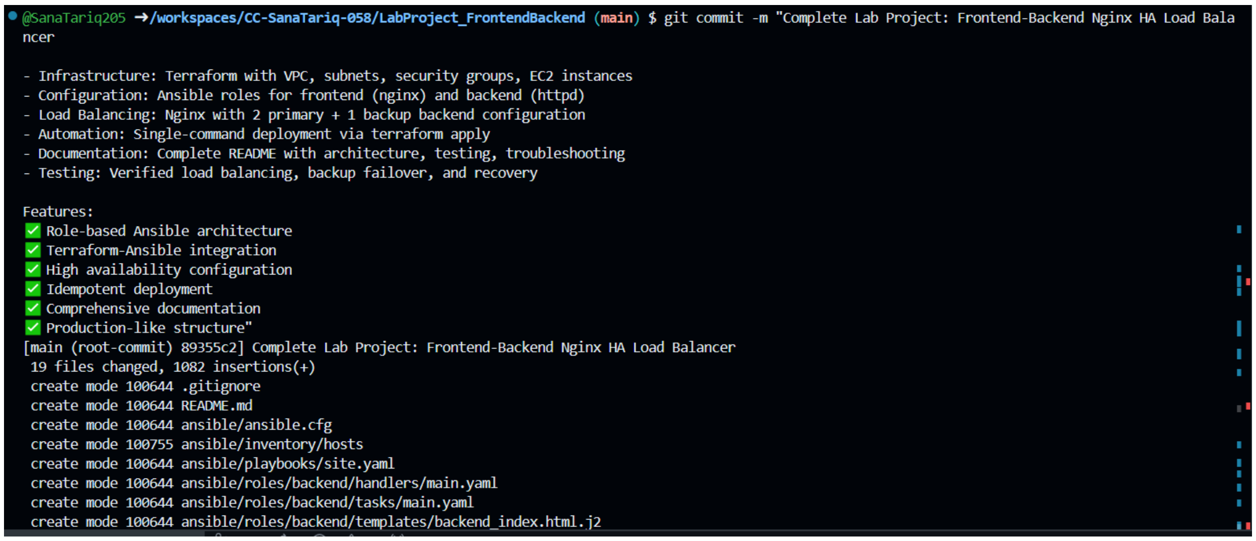
---

## License

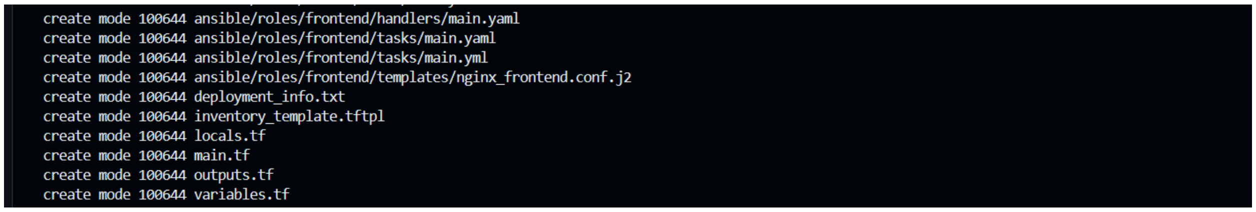
This project is for educational purposes as part of Cloud Computing Lab coursework.

---

**Project Status:** ✅ Complete and Tested
**Last Updated:** January 2025
```
```



```
@SanaTariq205 ➜ /workspaces/CC-SanaTariq-058/LabProject_FrontendBackend (main) $ git commit -m "Complete Lab Project: Frontend-Backend Nginx HA Load Bala
ncer

- Infrastructure: Terraform with VPC, subnets, security groups, EC2 instances
- Configuration: Ansible roles for frontend (nginx) and backend (httpd)
- Load Balancing: Nginx with 2 primary + 1 backup backend configuration
- Automation: Single-command deployment via terraform apply
- Documentation: Complete README with architecture, testing, troubleshooting
- Testing: Verified load balancing, backup failover, and recovery

Features:
✅ Role-based Ansible architecture
✅ Terraform-Ansible integration
✅ High availability configuration
✅ Idempotent deployment
✅ Comprehensive documentation
✅ Production-like structure"
[main (root-commit) 89355c2] Complete Lab Project: Frontend-Backend Nginx HA Load Balancer
 19 files changed, 1082 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 README.md
 create mode 100644 ansible/ansible.cfg
 create mode 100755 ansible/inventory/hosts
 create mode 100644 ansible/playbooks/site.yaml
 create mode 100644 ansible/roles/backend/handlers/main.yaml
 create mode 100644 ansible/roles/backend/tasks/main.yaml
 create mode 100644 ansible/roles/backend/templates/backend_index.html.j2
```



```
 create mode 100644 ansible/roles/frontend/handlers/main.yaml
 create mode 100644 ansible/roles/frontend/tasks/main.yaml
 create mode 100644 ansible/roles/frontend/tasks/main.yml
 create mode 100644 ansible/roles/frontend/templates/nginx_frontend.conf.j2
 create mode 100644 deployment_info.txt
 create mode 100644 inventory_template.tftpl
 create mode 100644 locals.tf
 create mode 100644 main.tf
 create mode 100644 outputs.tf
 create mode 100644 variables.tf
```