

Fatima Jinnah Women University

Department of Software Engineering



SUBJECT: CC LAB

SUBMITTED TO: SIR SHOAIB

SUBMITTED BY: ZUNAIRA NOOR

REGISTRATION NO: 2023-BSE-075

SEMESTER: V-B

STEP 0: Prerequisites

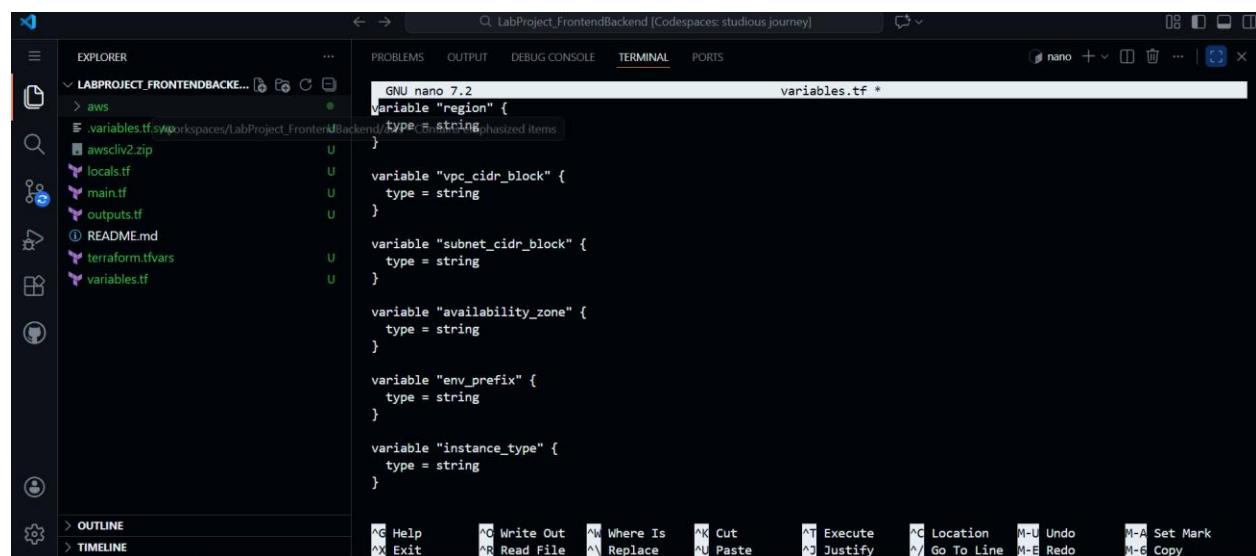
```
@Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend (main) $ terraform --version
Terraform v1.14.3
on linux_amd64
@Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend (main) $ ansible --version
ansible [core 2.16.3]
  config file = None
  configured module search path = ['~/home/codespace/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/codespace/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Jan 8 2026, 11:30:50) [GCC 13.3.0] (/usr/bin/python3)
  jinja version = 3.1.6
  libyaml = True
@Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend (main) $ aws --version
aws-cli/2.33.0 Python/3.13.11 Linux/6.8.0-1030-azure exe/x86_64_ubuntu.24
@Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend (main) $ python3 --version
Python 3.12.1
@Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend (main) $
```

STEP 1 – Terraform: Networking & Common Settings

STEP 1.0 – Confirm Root Files Exist

- @Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend (main) \$ touch main.tf
- @Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend (main) \$ touch variables.tf
- @Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend (main) \$ touch locals.tf
- @Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend (main) \$ touch outputs.tf
- @Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend (main) \$ touch terraform.tfvars

STEP 1.1 – variables.tf



```
variable "region" {
  type = string
}

variable "vpc_cidr_block" {
  type = string
}

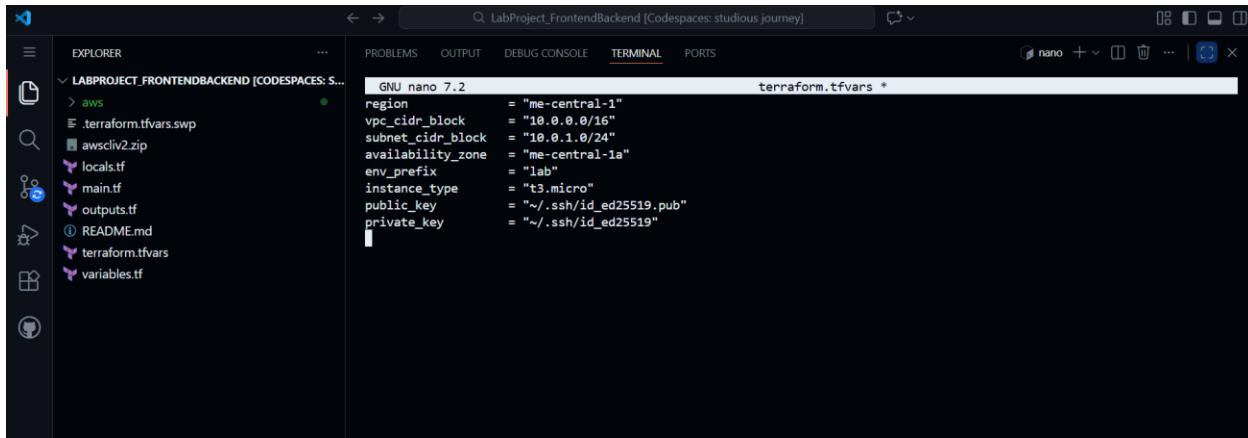
variable "subnet_cidr_block" {
  type = string
}

variable "availability_zone" {
  type = string
}

variable "env_prefix" {
  type = string
}

variable "instance_type" {
  type = string
}
```

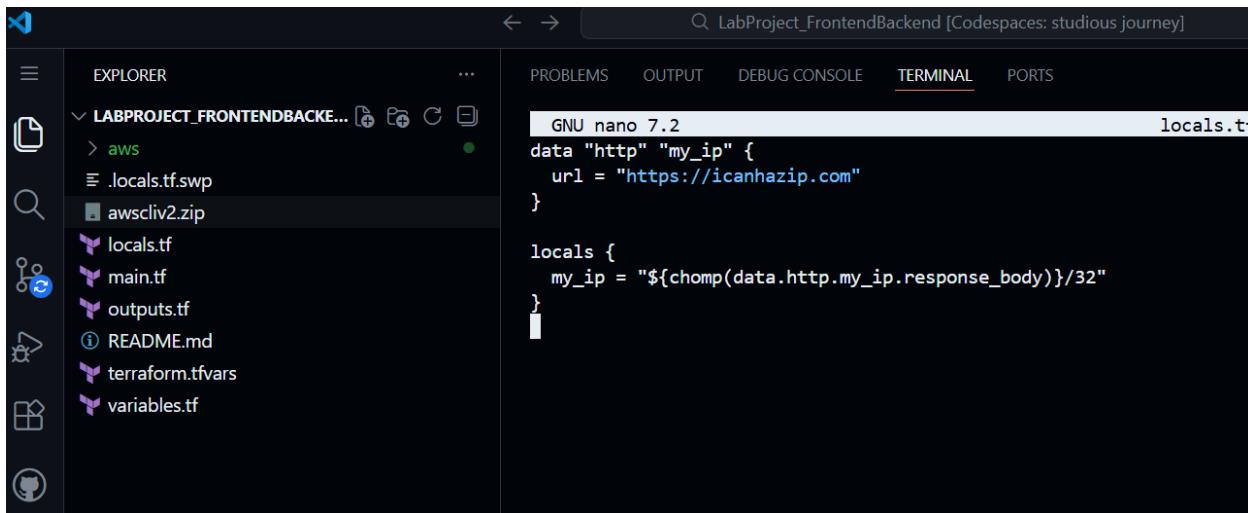
STEP 1.2 – terraform.tfvars (me-central-1)



The screenshot shows the VS Code interface with the terminal tab selected. The terminal window displays the contents of the `terraform.tfvars` file, which defines several variables for an AWS VPC:

```
region      = "me-central-1"
vpc_cidr_block = "10.0.0.0/16"
subnet_cidr_block = "10.0.1.0/24"
availability_zone = "me-central-1a"
env_prefix      = "lab"
instance_type    = "t3.micro"
public_key       = "~/.ssh/id_ed25519.pub"
private_key      = "~/.ssh/id_ed25519"
```

STEP 1.3 – locals.tf

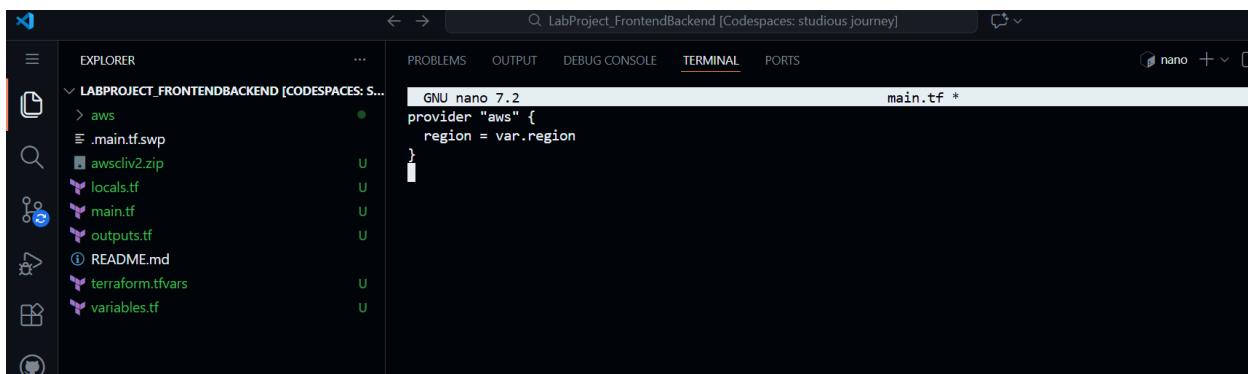


The screenshot shows the VS Code interface with the terminal tab selected. The terminal window displays the contents of the `locals.tf` file, which uses a `data` block to fetch the IP address of the `icanhazip.com` website and then defines a `locals` block to store it:

```
data "http" "my_ip" {
  url = "https://icanhazip.com"
}

locals {
  my_ip = "${chomp(data.http.my_ip.response_body)}/32"
}
```

STEP 1.4 – Provider Setup



The screenshot shows the VS Code interface with the terminal tab selected. The terminal window displays the contents of the `main.tf` file, which sets up an AWS provider block:

```
provider "aws" {
  region = var.region
}
```

STEP 1.5 – Create VPC

```
provider "aws" {
    region = var.region
}
resource "aws_vpc" "main" {
    cidr_block      = var.vpc_cidr_block
    enable_dns_support  = true
    enable_dns_hostnames = true
    tags = {
        Name = "${var.env_prefix}-vpc"
    }
}
```

STEP 1.6 – Internet Gateway

```
resource "aws_internet_gateway" "igw" {
    vpc_id = aws_vpc.main.id

    tags = {
        Name = "${var.env_prefix}-igw"
    }
}
```

STEP 1.7 – Route Table with Internet Access

```
}
resource "aws_route_table" "public_rt" {
    vpc_id = aws_vpc.main.id

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

    tags = {
        Name = "${var.env_prefix}-public-rt"
    }
}
```

STEP 1.8 – Public Subnet

```
resource "aws_route_table" "public_rt" {
    vpc_id = aws_vpc.main.id

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

    tags = {
        Name = "${var.env_prefix}-public-rt"
    }
}

resource "aws_subnet" "public" {
    vpc_id           = aws_vpc.main.id
    cidr_block       = var.subnet_cidr_block
    availability_zone = var.availability_zone
    map_public_ip_on_launch = true

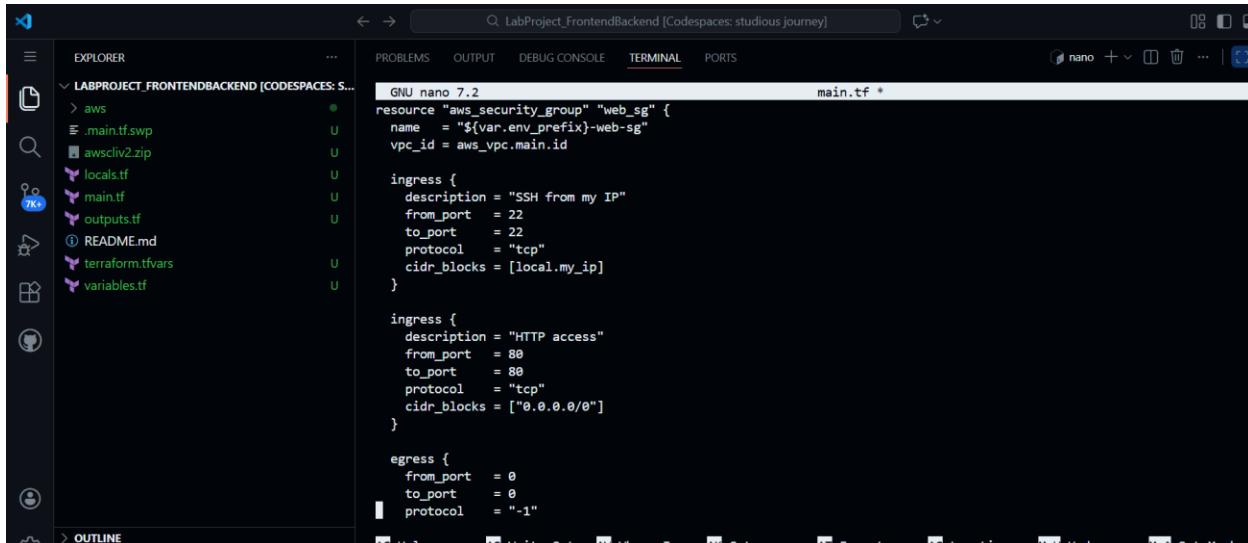
    tags = {
        Name = "${var.env_prefix}-public-subnet"
    }
}
```

STEP 1.9 – Route Table Association

```
}
```

```
resource "aws_route_table_association" "public_assoc" {
    subnet_id      = aws_subnet.public.id
    route_table_id = aws_route_table.public_rt.id
}
```

STEP 1.10 – Security Group



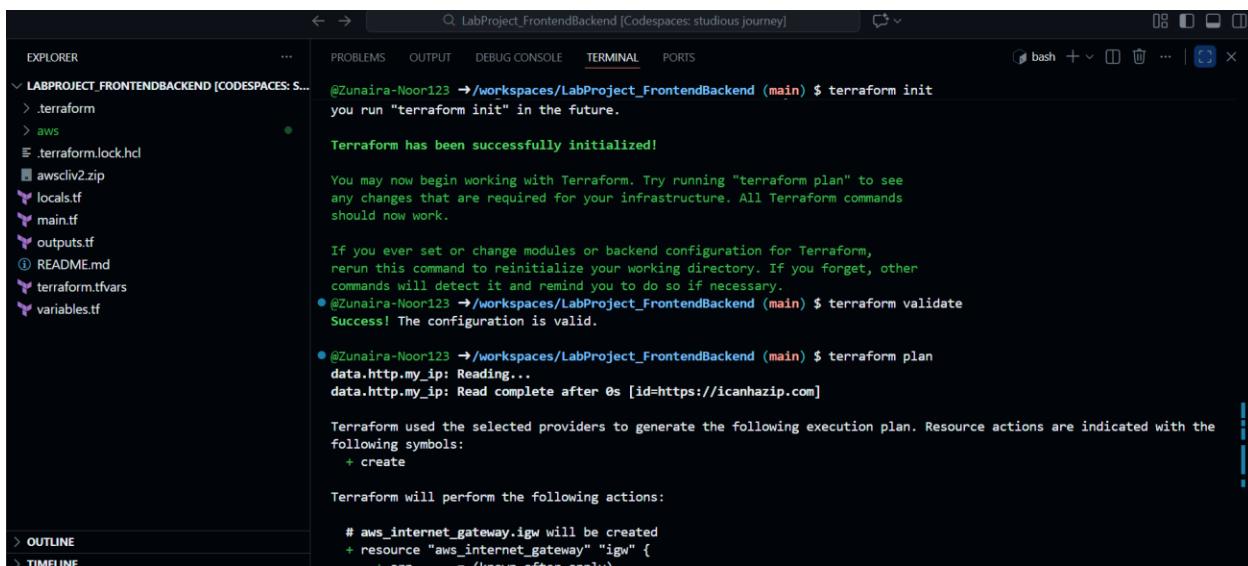
```
GNU nano 7.2
resource "aws_security_group" "web_sg" {
  name      = "${var.env_prefix}-web-sg"
  vpc_id    = aws_vpc.main.id

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

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

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
  }
}
```

STEP 1.11 – Test Terraform



```
@Zunaira-Noor123 ~/workspaces/LabProject_FrontendBackend (main) $ terraform init
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.

@Zunaira-Noor123 ~/workspaces/LabProject_FrontendBackend (main) $ terraform validate
Success! The configuration is valid.

@Zunaira-Noor123 ~/workspaces/LabProject_FrontendBackend (main) $ terraform plan
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 0s [id=https://icanhazip.com]

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:

# aws_internet_gateway.igw will be created
+ resource "aws_internet_gateway" "igw" {
    + arn      = (known after apply)
```

STEP 2 – Terraform: Frontend & Backend EC2 Instances

STEP 2.1 – Find Amazon Linux 2 AMI

```
●   region = var.region
U }
U data "aws_ami" "amazon_linux" {
U   most_recent = true
U   owners      = ["amazon"]
U
U   filter {
U     name    = "name"
U     values  = ["amzn2-ami-hvm-*-x86_64-gp2"]
U   }
U
U   filter {
U     name    = "virtualization-type"
```

STEP 2.2 – Create Key Pair Resource

```
        }
      }
resource "aws_key_pair" "deployer" {
  key_name    = "${var.env_prefix}-key"
  public_key  = file(var.public_key)
}
```

STEP 2.3 – Frontend EC2 Instance

```
resource "aws_instance" "frontend" {
  ami           = data.aws_ami.amazon_linux.id
  instance_type = var.instance_type
  subnet_id     = aws_subnet.public.id
  vpc_security_group_ids = [aws_security_group.web_sg.id]
  key_name      = aws_key_pair.deployer.key_name

  tags = {
    Name = "${var.env_prefix}-frontend"
    Role = "frontend"
  }
}
```

STEP 2.4 – Backend EC2 Instances (Count = 3)

```

resource "aws_instance" "backend" {
    count                  = 3
    ami                    = data.aws_ami.amazon_linux.id
    instance_type          = var.instance_type
    subnet_id              = aws_subnet.public.id
    vpc_security_group_ids = [aws_security_group.web_sg.id]
    key_name               = aws_key_pair.deployer.key_name

    tags = {
        Name = "${var.env_prefix}-backend-${count.index}"
        Role = "backend"
    }
}

```

STEP 2.5 – Outputs

```

GNU nano 7.2
output "frontend_public_ip" {
  value = aws_instance.frontend.public_ip
}

output "backend_public_ips" {
  value = [for b in aws_instance.backend : b.public_ip]
}

output "backend_private_ips" {
  value = [for b in aws_instance.backend : b.private_ip]
}

```

STEP 2.6 – Terraform Test

```

@Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend (main) $ terraform apply -auto-approve
aws_instance.backend[2]: Creating...
aws_route_table_association.public_assoc: Creation complete after 0s [id=rtbassoc-0a40a302b90eb8c20]
aws_instance.backend[0]: Still creating... [0m10s elapsed]
aws_instance.backend[1]: Still creating... [0m10s elapsed]
aws_instance.backend[2]: Still creating... [0m10s elapsed]
aws_instance.frontend: Still creating... [0m10s elapsed]
aws_instance.backend[0]: Creation complete after 12s [id=i-022fdc9dd8255f73b]
aws_instance.backend[2]: Creation complete after 12s [id=i-06b3d8b98637225dc]
aws_instance.frontend: Creation complete after 12s [id=i-07ceb322566dd9ca1]
aws_instance.backend[1]: Creation complete after 12s [id=i-03ae7a32b523d4cff]

Apply complete! Resources: 11 added, 0 changed, 0 destroyed.

Outputs:

backend_private_ips = [
  "10.0.1.36",
  "10.0.1.122",
  "10.0.1.102",
]
backend_public_ips = [
  "40.172.186.28",
  "3.28.131.7",
  "3.28.132.35",
]
frontend_public_ip = "3.28.40.0"

@Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend (main) $

```

STEP 3 – Ansible: Global Config & Inventory

STEP 3.1 – Create Ansible Folder Structure

STEP 3.2 – ansible.cfg

The screenshot shows the VS Code interface with the 'TERMINAL' tab selected. The terminal window displays the following content:

```
GNU nano 7.2
[defaults]
host_key_checking = False
interpreter_python = /usr/bin/python3
```

The left side of the screen shows a file tree for a project named 'LABPROJECT_FRONTENDBACKEND'. The files listed include .terraform, ansible, aws, .terraform.lock.hcl, awscli2.zip, locals.tf, main.tf, outputs.tf, README.md, terraform.tfstate, and terraform.tfvars.

STEP 3.3 – Inventory File

The screenshot shows the VS Code interface with the 'TERMINAL' tab selected. The terminal window displays the following content:

```
GNU nano 7.2
[frontend]
3.28.40.0

[backends]
40.172.186.28
3.28.131.7
3.28.132.35

[all:vars]
ansible_user=ec2-user
ansible_ssh_private_key_file=~/.ssh/id_ed25519
```

The left side of the screen shows a file tree for a project named 'PROJECT_FRONTENDBACKEND'. The files listed include .terraform, ansible, aws, .terraform.lock.hcl, awscli2.zip, locals.tf, main.tf, outputs.tf, README.md, terraform.tfstate, terraform.tfvars, and variables.tf.

STEP 3.5 – Main Playbook

```
GNU nano 7.2                               ansible/playbooks/site.yaml *
```

```
---
- name: Configure backend HTTPD servers
  hosts: backends
  become: true
  roles:
    - backend

- name: Configure frontend Nginx load balancer
  hosts: frontend
  become: true
  vars:
    backend1_private_ip: "{{ hostvars[groups['backends'][0]].ansible_default_ipv4.address }}"
    backend2_private_ip: "{{ hostvars[groups['backends'][1]].ansible_default_ipv4.address }}"
    backup_backend_private_ip: "{{ hostvars[groups['backends'][2]].ansible_default_ipv4.address }}"
  roles:
    - frontend
```

STEP 3.6 – Test Inventory

```
@Zunaira-Noor123 →/workspaces/LabProject_FrontendBackend/ansible (main) $ chmod 755 /workspaces/LabProject_FrontendBackend/ansible
@Zunaira-Noor123 →/workspaces/LabProject_FrontendBackend/ansible (main) $ ansible -i inventory/hosts all -m ping
3.28.40.0 | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
40.172.186.28 | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
3.28.131.7 | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
3.28.132.35 | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
@Zunaira-Noor123 →/workspaces/LabProject_FrontendBackend/ansible (main) $
```

STEP 4 – Backend Role (HTTPD)

4.1 – roles/backend/tasks/main.yml

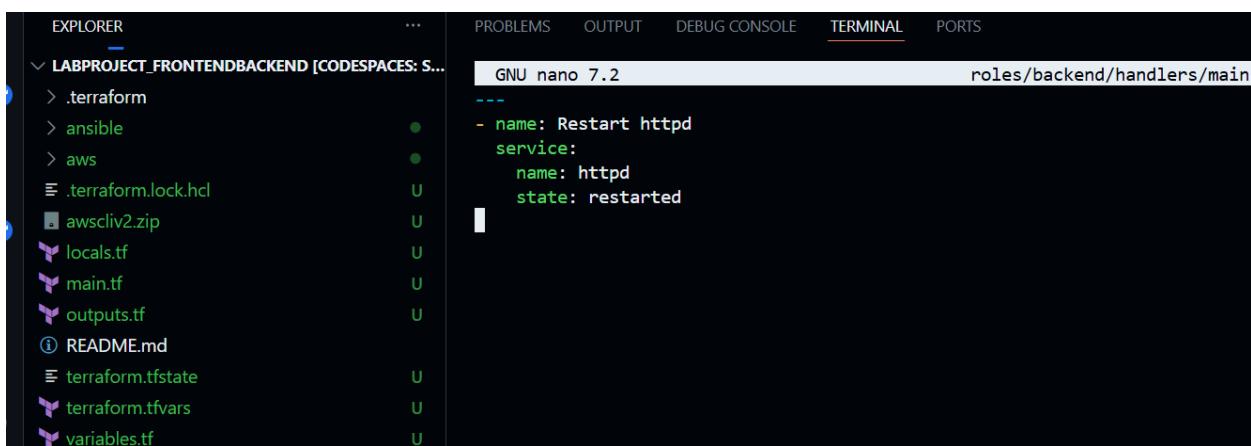
```
GNU nano 7.2                                         roles/backend/tasks/main
---
- name: Install httpd
  yum:
    name: httpd
    state: present
    update_cache: yes

- name: Enable and start httpd
  service:
    name: httpd
    state: started
    enabled: true

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

^G Help      ^O Write Out   ^W Where Is   ^K Cut        ^T Execute
^X Exit     ^R Read File   ^\ Replace    ^U Paste      ^J Justify
```

4.2 – roles/backend/handlers/main.yml



```
EXPLORER          ...          PROBLEMS      OUTPUT      DEBUG CONSOLE      TERMINAL      PORTS
LABPROJECT_FRONTENDBACKEND [CODESPACES: S...          GNU nano 7.2                                         roles/backend/handlers/main
---
- name: Restart httpd
  service:
    name: httpd
    state: restarted
```

4.3 – roles/backend/templates/backend_index.html.j2

The screenshot shows the VS Code interface with the following tabs active: EXPLORER, PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The TERMINAL tab is currently selected and displays the command 'ansible-playbook -i inventory/hosts playbooks/site.yaml'. The output of the command is visible below the command line.

4.4 – Test Backend Role

```
@Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend/ansible (main) $ ansible-playbook -i inventory/hosts playbooks/site.yaml

PLAY [Configure backend HTTPD servers] ****
TASK [Gathering Facts] ****
ok: [3.28.132.35]
ok: [40.172.186.28]
ok: [3.28.131.7]

TASK [backend : Install httpd] ****
changed: [3.28.132.35]
changed: [3.28.131.7]
changed: [40.172.186.28]

TASK [backend : Enable and start httpd] ****
changed: [3.28.131.7]
changed: [40.172.186.28]
changed: [3.28.132.35]

TASK [backend : Deploy backend index page] ****
changed: [40.172.186.28]
changed: [3.28.132.35]
changed: [3.28.131.7]

RUNNING HANDLER [backend : Restart httpd] ****
changed: [3.28.131.7]
```

After success, you can visit the backend public IPs in a browser:

Backend server: 3.28.131.7

Private IP: 10.0.1.122

Backend server: 3.28.132.35

Private IP: 10.0.1.102

Backend server: 40.172.186.28

Private IP: 10.0.1.36

STEP 5 – Frontend Role (Nginx Load Balancer)

5.1 – roles/frontend/tasks/main.yml

The screenshot shows the VS Code interface with the 'TERMINAL' tab selected. The terminal window displays the YAML configuration for a task named 'Install nginx'. The configuration includes installing nginx via yum, enabling and starting the service, and deploying a frontend config template.

```
GNU nano 7.2
---
- name: Install nginx
  yum:
    name: nginx
    state: present
    update_cache: yes

- name: Enable and start nginx
  service:
    name: nginx
    state: started
    enabled: true

- name: Deploy nginx frontend config
  template:
    src: nginx_frontend.conf.j2
    dest: /etc/nginx/nginx.conf
    owner: root
    group: root
    mode: '0644'
  notify: Restart nginx
```

5.2 – roles/frontend/handlers/main.yml

The screenshot shows the VS Code interface with the 'TERMINAL' tab selected. The terminal window displays the YAML configuration for a handler named 'Restart nginx'. It specifies a service named 'nginx' with the state set to 'restarted'.

```
GNU nano 7.2
---
- name: Restart nginx
  service:
    name: nginx
    state: restarted
```

5.3 – roles/frontend/templates/nginx_frontend.conf.j2

```
GNU nano 7.2                                         roles/frontend/templates/nginx_frontend.conf.j2 *
```

```
server {  
    listen 80;  
    server_name _;  
  
    location / {  
        proxy_pass http://backend_servers;  
        proxy_set_header Host $host;  
        proxy_set_header X-Real-IP $remote_addr;  
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;  
    }  
}
```

Ansible Main Playbook Using Roles

5.4 – Update playbooks/site.yaml for Frontend

```
GNU nano 7.2                                         playbooks/site.yaml  
---  
- name: Configure backend HTTPD servers  
  hosts: backends  
  become: true  
  roles:  
    - backend  
  
- name: Configure frontend Nginx load balancer  
  hosts: frontend  
  become: true  
  vars:  
    backend1_private_ip: "{{ hostvars[groups['backends'][0]].ansible_de...  
    backend2_private_ip: "{{ hostvars[groups['backends'][1]].ansible_de...  
    backup_backend_private_ip: "{{ hostvars[groups['backends'][2]].ansi...  
  roles:  
    - frontend
```

5.5 – Run Full Playbook

```

@Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend/ansible (main) $ ansible-playbook -i inventory/hosts playbooks/site.yaml --tags frontend

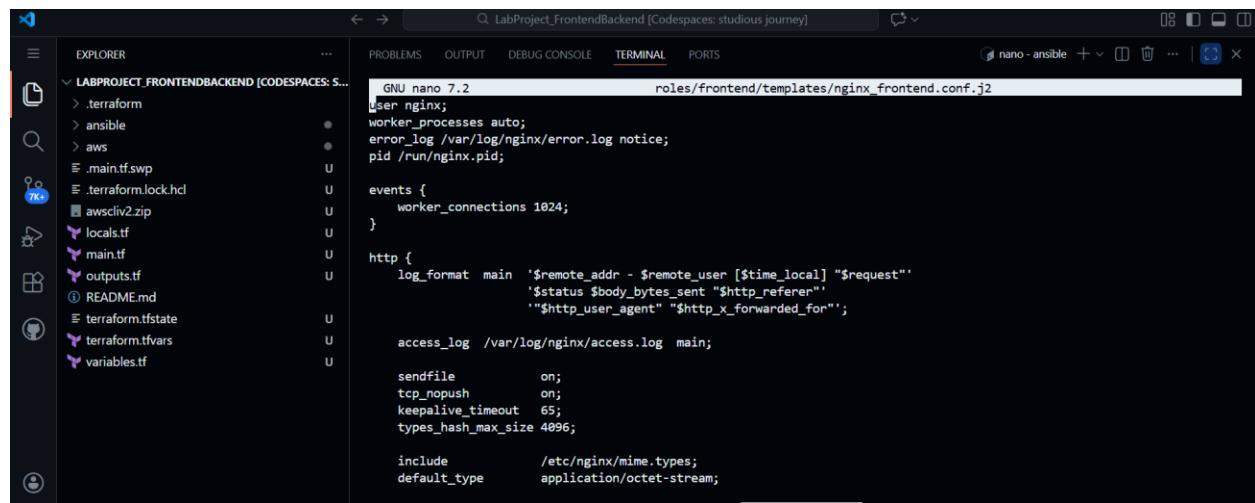
PLAY [Configure backend HTTPD servers] ****
TASK [Gathering Facts] ****
ok: [40.172.186.28]
ok: [3.28.132.35]
ok: [3.28.131.7]

PLAY [Configure frontend Nginx load balancer] ****
TASK [Gathering Facts] ****
ok: [3.28.40.0]

PLAY RECAP ****
3.28.131.7 : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
3.28.132.35 : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
3.28.40.0   : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
40.172.186.28: ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

```

roles/frontend/templates/nginx_frontend.conf.j2



```

GNU nano 7.2
User nginx;
worker_processes auto;
error_log /var/log/nginx/error.log notice;
pid /run/nginx.pid;

events {
    worker_connections 1024;
}

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

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

    sendfile      on;
    tcp_nopush    on;
    keepalive_timeout 65;
    types_hash_max_size 4096;

    include       /etc/nginx/mime.types;
    default_type  application/octet-stream;
}

```

Terraform Test

```
① @Zunaira-Noor123 →/workspaces/LabProject_FrontendBackend (main) $ terraform init
  Initializing the backend...
  Initializing provider plugins...
    - Reusing previous version of hashicorp/aws from the dependency lock file
    - Reusing previous version of hashicorp/http from the dependency lock file
    - Using previously-installed hashicorp/aws v6.28.0
    - Using previously-installed hashicorp/http v3.5.0

  Terraform has been successfully initialized!

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

  If you ever set or change modules or backend configuration for Terraform,
  rerun this command to reinitialize your working directory. If you forget, other
  commands will detect it and remind you to do so if necessary.
② @Zunaira-Noor123 →/workspaces/LabProject_FrontendBackend (main) $ terraform validate
  Success! The configuration is valid.

③ @Zunaira-Noor123 →/workspaces/LabProject_FrontendBackend (main) $
```

```
④ @Zunaira-Noor123 →/workspaces/LabProject_FrontendBackend (main) $ terraform apply -auto-approve
  name          = "lab-web-sg"
  tags          = {
    "Name" = "lab-sg"
  }
  # (9 unchanged attributes hidden)
}

Plan: 0 to add, 1 to change, 0 to destroy.
aws_security_group.web_sg: Modifying... [id=sg-09430f46473ac8ee3]
aws_security_group.web_sg: Modifications complete after 1s [id=sg-09430f46473ac8ee3]

Apply complete! Resources: 0 added, 1 changed, 0 destroyed.

Outputs:
backend_private_ips = [
  "10.0.1.36",
  "10.0.1.122",
  "10.0.1.102",
]
backend_public_ips = [
  "40.172.186.28",
  "3.28.131.7",
  "3.28.132.35",
]
frontend_public_ip = "3.28.40.0"

⑤ @Zunaira-Noor123 →/workspaces/LabProject_FrontendBackend (main) $
```

Connection through ssh

```
ssh: connect to host 3.28.42.68 port 22: Connection timed out
@Zunaira-Noor123 ➔ /workspaces/LabProject_FrontendBackend (main) $ ssh -i ~/.ssh/id_ed25519 ec2-user@3.28.40.0
Last login: Thu Jan 15 17:46:29 2026 from 20.192.21.48
,
      #
~\_ ####_      Amazon Linux 2
~~ \####\_
~~ \|##|      AL2 End of Life is 2026-06-30.
~~ \|#/ ___
~~ V~' '->
~~ /      A newer version of Amazon Linux is available!
~~-. /_
/_/_      Amazon Linux 2023, GA and supported until 2028-03-15.
/_m/      https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-10-0-1-114 ~]$
```



Backend server: 40.172.186.28

Private IP: 10.0.1.36



Backend server: 3.28.131.7

Private IP: 10.0.1.122

Terraform–Ansible Integration (Automation)

Use null_resource with local-exec

ONTENDBACK... ⌂ ⌂ ⌂ ⌂

GNU nano 7.2 main.tf *

```
    Role = "backend"
}
}
resource "null_resource" "ansible_config" {
  triggers = {
    frontend_ip = aws_instance.frontend.public_ip
    backend_ips = join(",", [for b in aws_instance.backend : b.public_ip])
  }

  depends_on = [
    aws_instance.frontend,
    aws_instance.backend
  ]

  provisioner "local-exec" {
    command = << EOT
    cd ansible
    ANSIBLE_HOST_KEY_CHECKING=False ansible-playbook \
      -i "../generated_hosts.ini" \
      playbooks/site.yaml
    EOT
  }
}
```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark
^X Exit ^R Read File ^V Replace ^L Paste ^J Justify ^I Go To Line M-F Redo M-G Copy

Generate generated_hosts.ini dynamically

```
        -i "../generated_hosts.ini" \
        playbooks/site.yaml
    EOT
}
}
data "template_file" "hosts_ini" {
    template = <<EOT
[frontend]
${aws_instance.frontend.public_ip}

[backend]
${{ for ip in aws_instance.backend[*].public_ip ~}}
${ip}
${{ endfor ~}}
EOT
}

resource "local_file" "hosts_ini" {
    content  = data.template_file.hosts_ini.rendered
    filename = "${path.module}/generated_hosts.ini"
}
```

```
@Zunaira-Noor123 → /workspaces/LabProject_FrontendBackend (main) $ terraform apply -auto-approve

Plan: 2 to add, 0 to change, 0 to destroy.
local_file.hosts_ini: Creating...
null_resource.ansible_config: Creating...
local_file.hosts_ini: Creation complete after 0s [id=3f5c2b637a033ae0c766008ade65244ab86cf9e5]
null_resource.ansible_config: Provisioning with 'local-exec'...
null_resource.ansible_config (local-exec): Executing: ["./bin/sh" "-c" "cd ansible\nANSIBLE_HOST_KEY_CHECKING=False ansible -playbook \\\n -i \"..../generated_hosts.ini\" \\\n playbooks/site.yaml\\n"]
null_resource.ansible_config (local-exec): [WARNING]: Could not match supplied host pattern, ignoring: backends

null_resource.ansible_config (local-exec): PLAY [Configure backend HTTPD servers] ****
*
null_resource.ansible_config (local-exec): skipping: no hosts matched

null_resource.ansible_config (local-exec): PLAY [Configure frontend Nginx load balancer] ****
*
null_resource.ansible_config (local-exec): TASK [Gathering Facts] ****
*
null_resource.ansible_config (local-exec): fatal: [3.28.40.0]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: codespace@3.28.40.0: Permission denied (publickey,gssapi-keyex,gssapi-with-mic).", "unreachable": true}

null_resource.ansible_config (local-exec): PLAY RECAP ****
*
null_resource.ansible_config (local-exec): 3.28.40.0 : ok=0    changed=0    unreachable=1    failed=0
ok=0    changed=0    ignored=0
```