Fatima Jinnah Women University

Opening Portals of Excellence Through Higher Education

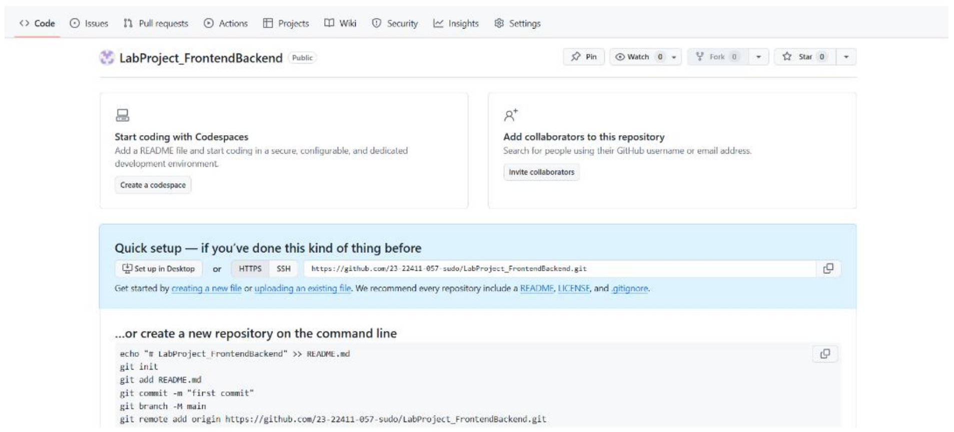
Name: Zuha Irfan
Roll No: 2023-BSE-073
Submitted To: Sir Shoaib
Semester & Section: V-B
Subject: Cloud Computing
Open Ended Lab & Project

<u>Terraform + Ansible Roles: Nginx Frontend with 3 Backend HTTPD Servers (HA + Auto-Config)</u>

Project Setup (Folder Structure & Git Initialization)

And inside it:

1.1 GitHub Repository Creation



Start coding with Codespaces
Add a README file and start coding in a secure, configurable, and dedicated development environment.

Add collaborators to this repository
Search for people using their GitHub username or email address.

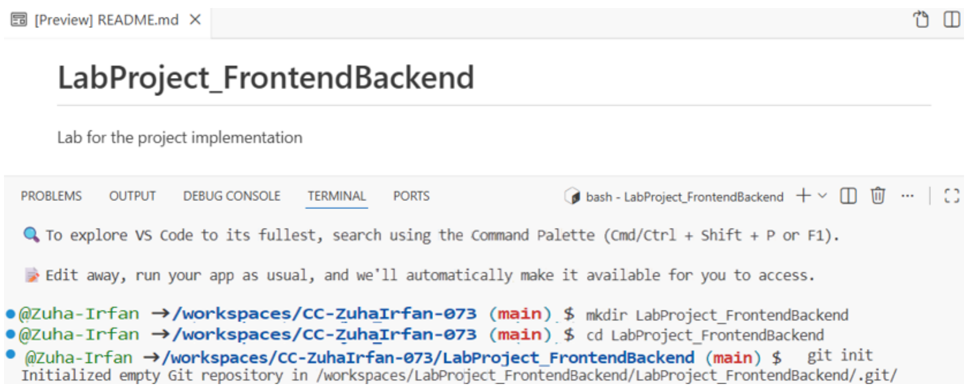
Quick setup — if you've done this kind of thing before

https://github.com/23-22411-057-sudo/LabProject_FrontendBackend.git

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

```
echo "# LabProject_FrontendBackend" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/23-22411-057-sudo/LabProject_FrontendBackend.git
```

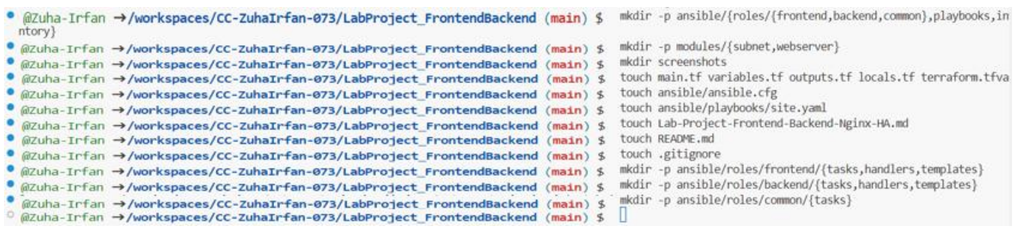


LabProject_FrontendBackend

Lab for the project implementation

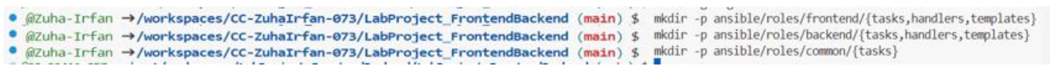
```
🔍 To explore VS Code to its fullest, search using the Command Palette (Cmd/Ctrl + Shift + P or F1).

📝 Edit away, run your app as usual, and we'll automatically make it available for you to access.

● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073 (main) $ mkdir LabProject_FrontendBackend
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073 (main) $ cd LabProject_FrontendBackend
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  git init
Initialized empty Git repository in /workspaces/LabProject_FrontendBackend/LabProject_FrontendBackend/.git/
```

Directory Structure



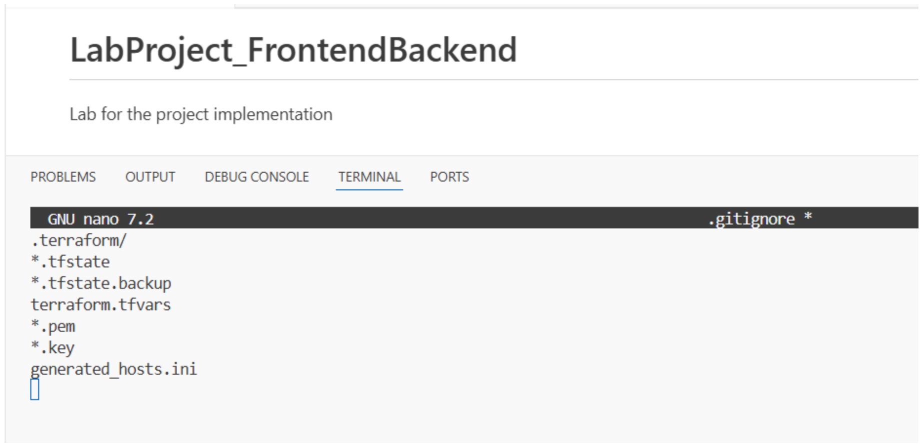
```
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  mkdir -p ansible/{roles/{frontend,backend,common},playbooks,in
ntory}
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  mkdir -p modules/{subnet,webserver}
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  mkdir screenshots
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  touch main.tf variables.tf outputs.tf locals.tf terraform.tfva
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  touch ansible/ansible.cfg
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  touch ansible/playbooks/site.yaml
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  touch Lab-Project-Frontend-Backend-Nginx-HA.md
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  touch README.md
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  touch .gitignore
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  mkdir -p ansible/roles/frontend/{tasks,handlers,templates}
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  mkdir -p ansible/roles/backend/{tasks,handlers,templates}
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  mkdir -p ansible/roles/common/{tasks}
○ @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $
```

Now create role subfolders:



```
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  mkdir -p ansible/roles/frontend/{tasks,handlers,templates}
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  mkdir -p ansible/roles/backend/{tasks,handlers,templates}
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  mkdir -p ansible/roles/common/{tasks}
```

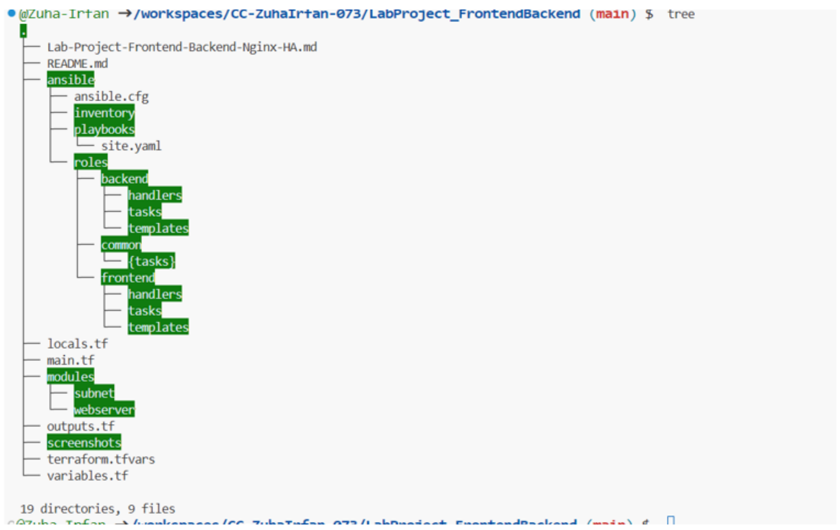
Git Ignore Configuration:


LabProject_FrontendBackend

Lab for the project implementation

```
  GNU nano 7.2                                                  .gitignore *
.terraform/
*.tfstate
*.tfstate.backup
terraform.tfvars
*.pem
*.key
generated_hosts.ini
```

Verification



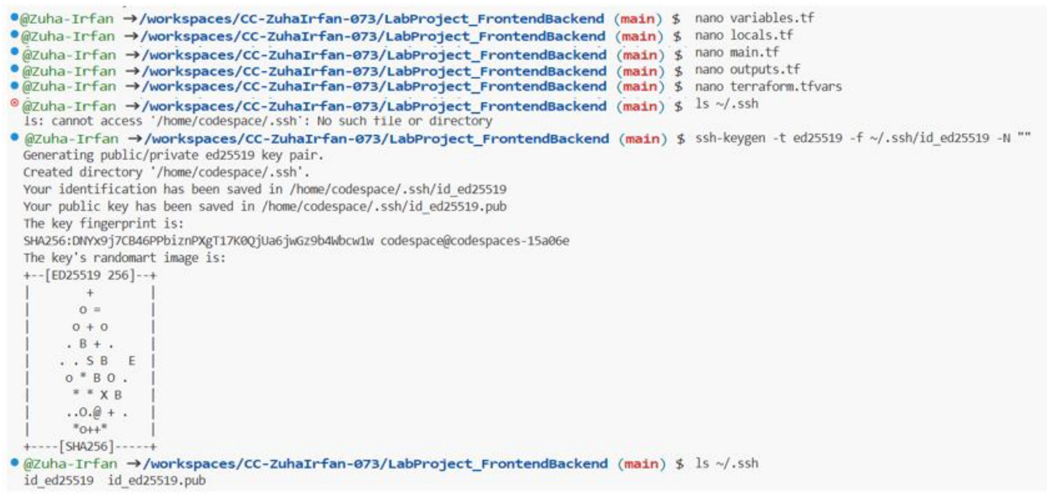
```
@Zuha-Irfan ➜ /workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ tree
.
├── Lab-Project-Frontend-Backend-Nginx-HA.md
├── README.md
├── ansible
│   ├── ansible.cfg
│   ├── inventory
│   ├── playbooks
│   │   └── site.yaml
│   └── roles
│       ├── backend
│       │   ├── handlers
│       │   ├── tasks
│       │   └── templates
│       ├── common
│       │   └── {tasks}
│       └── frontend
│           ├── handlers
│           ├── tasks
│           └── templates
├── locals.tf
├── main.tf
├── modules
│   ├── subnet
│   └── webserver
├── outputs.tf
├── screenshots
├── terraform.tfvars
└── variables.tf

19 directories, 9 files
```

<u>**Terraform – Networking & Common Settings (Architecture Definition)**</u>

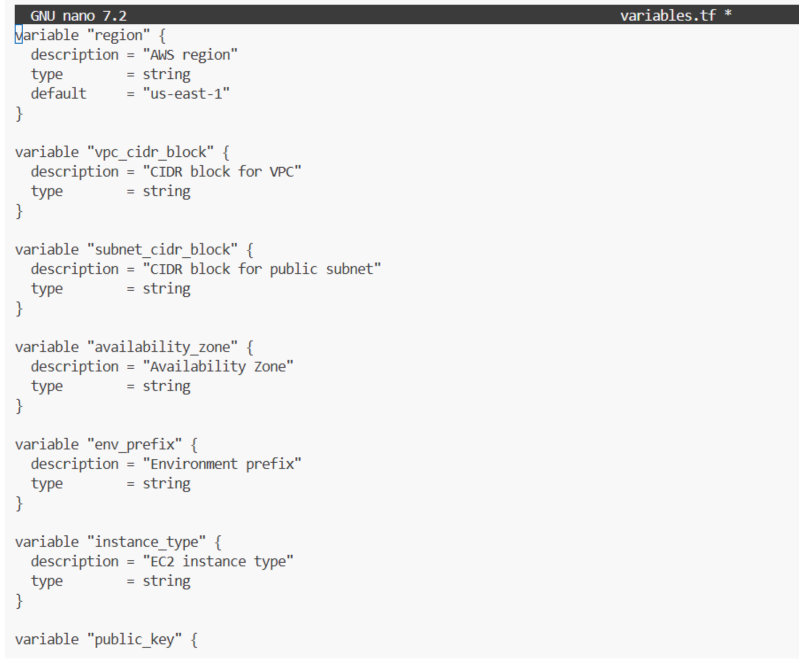
Requirements:

1. Variables setup (variables.tf):

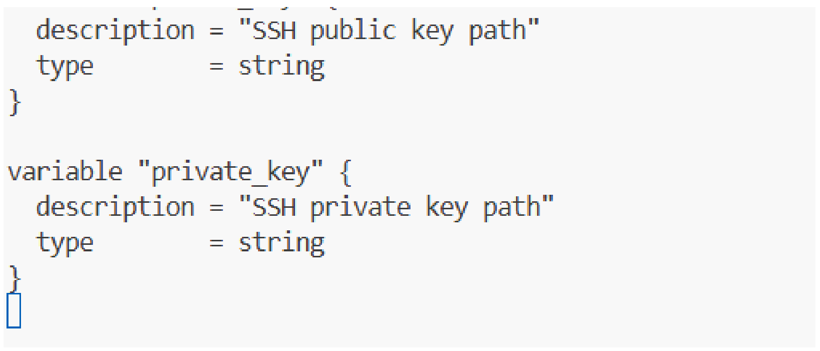
- vpc_cidr_block, subnet_cidr_block, availability_zone, env_prefix, instance_type, public_key, private_key.



```
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  nano variables.tf
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  nano locals.tf
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  nano main.tf
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  nano outputs.tf
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  nano terraform.tfvars
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  ls ~/.ssh
ls: cannot access '/home/codespace/.ssh': No such file or directory
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ ssh-keygen -t ed25519 -f ~/.ssh/id_ed25519 -N ""
Generating public/private ed25519 key pair.
Created directory '/home/codespace/.ssh'.
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:DNYx9j7CB46PPbiznPXgT17K0QjUa6jwGz9b4Wbcw1w codespace@codespaces-15a06e
The key's randomart image is:
+--[ED25519 256]--+
|       +         |
|      o =        |
|     o + o       |
|    . B + .      |
|   . . S B   E   |
|    o * B O .    |
|     * * X B     |
|    ..O.@ + .    |
|     *o++*       |
+----[SHA256]-----+
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  ls ~/.ssh
id_ed25519  id_ed25519.pub
```



```
 GNU nano 7.2                                                  variables.tf *
variable "region" {
  description = "AWS region"
  type        = string
  default     = "us-east-1"
}

variable "vpc_cidr_block" {
  description = "CIDR block for VPC"
  type        = string
}

variable "subnet_cidr_block" {
  description = "CIDR block for public subnet"
  type        = string
}

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

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

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

variable "public_key" {
```

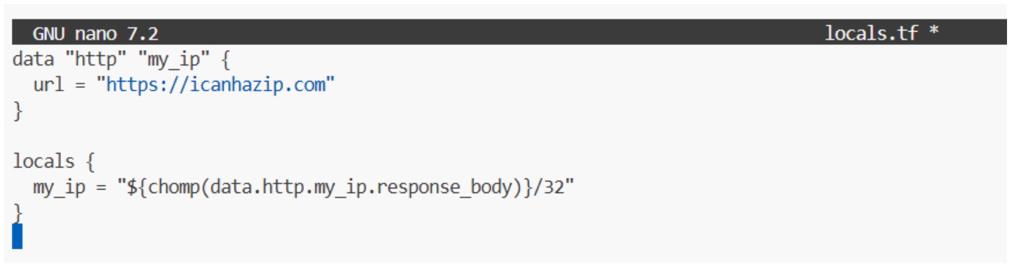


```
  description = "SSH public key path"
  type        = string
}

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

Locals (locals.tf):

Determine your public IP:

```
locals {
  my_ip = "${chomp(data.http.my_ip.response_body)}/32"
}
data "http" "my_ip" {
  url = "https://icanhazip.com"
}
```



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

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

VPC & Subnet:

- Either inline in main.tf or via modules/subnet.

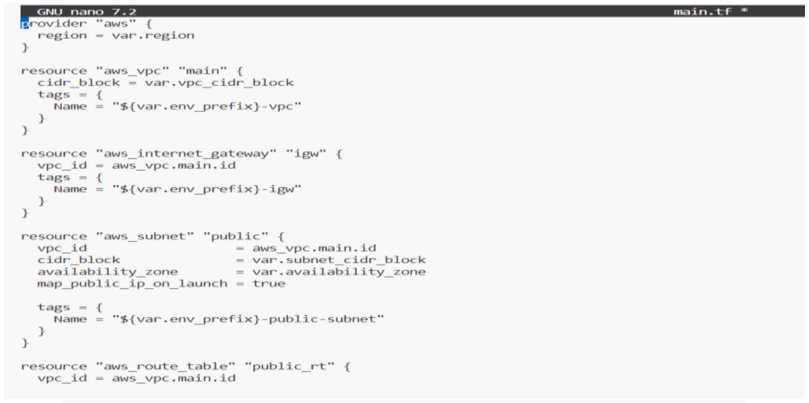
  - VPC must have:
  - CIDR from var.vpc_cidr_block.

- Internet gateway and default route for 0.0.0.0/0.
- Subnet must be public (map public IPs on launch or use IGW + route table).

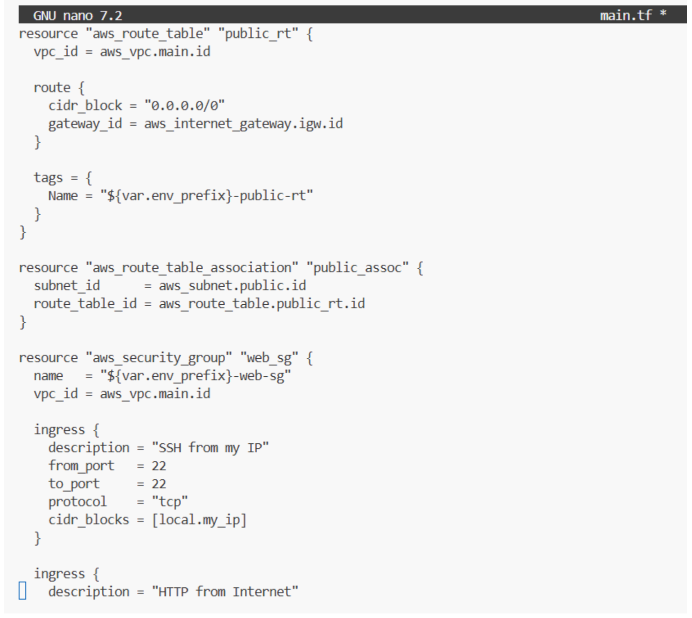
2. Security Group:

1. SSH: port 22 from local.my_ip.
2. HTTP: port 80 from 0.0.0.0/0.

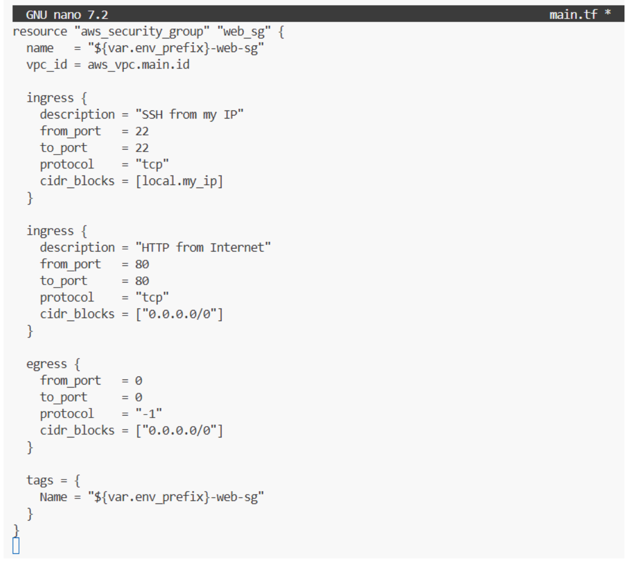
Main.tf



```
  GNU nano 7.2                                                    main.tf *
provider "aws" {
  region = var.region
}

resource "aws_vpc" "main" {
  cidr_block = var.vpc_cidr_block
  tags = {
    Name = "${var.env_prefix}-vpc"
  }
}

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

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"
  }
}

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

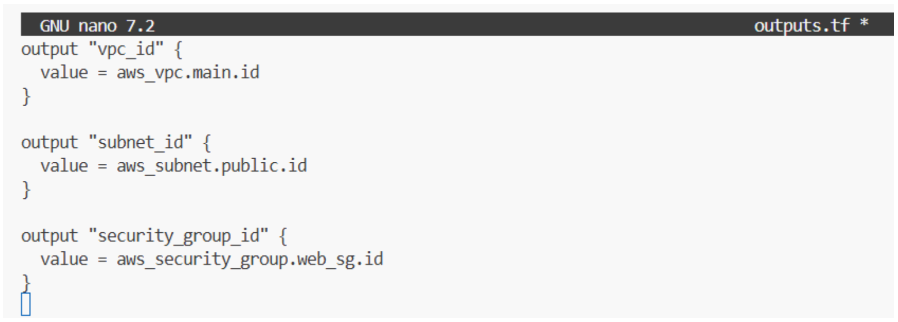


```
  GNU nano 7.2                                              main.tf *
resource "aws_route_table" "public_rt" {
  vpc_id = aws_vpc.main.id

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

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

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

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 from Internet"
```

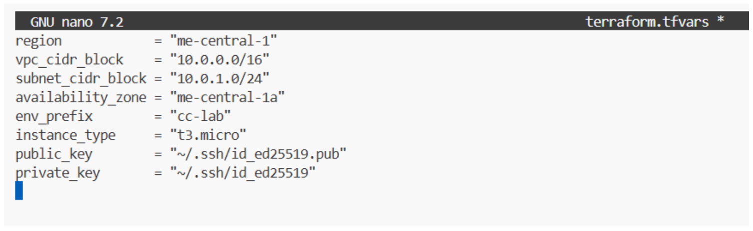


```
  GNU nano 7.2                                              main.tf *
resource "aws_security_group" "web_sg" {
  name   = "${var.env_prefix}-web-sg"
  vpc_id = aws_vpc.main.id

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

  ingress {
    description = "HTTP from Internet"
    from_port   = 80
    to_port     = 80
    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}-web-sg"
  }
}
```

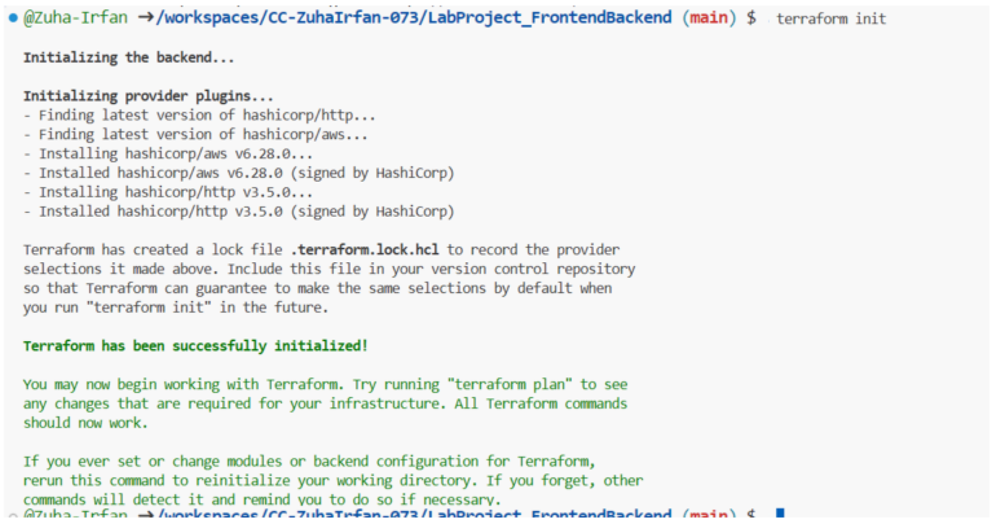
Outputs.tf:


```
GNU nano 7.2                                                    outputs.tf *
output "vpc_id" {
  value = aws_vpc.main.id
}

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

output "security_group_id" {
  value = aws_security_group.web_sg.id
}
```

terraform.tfvars:


```
GNU nano 7.2                                          terraform.tfvars *
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        = "cc-lab"
instance_type     = "t3.micro"
public_key        = "~/.ssh/id_ed25519.pub"
private_key       = "~/.ssh/id_ed25519"
```

Terraform init:

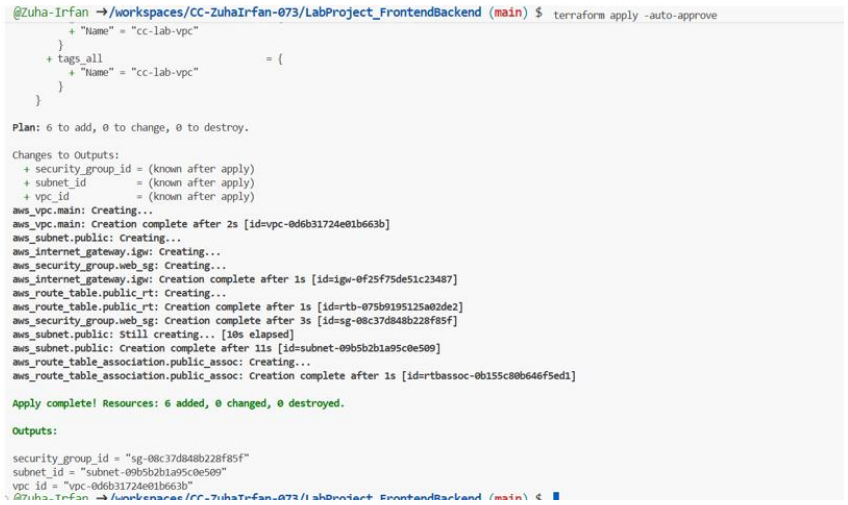

```
● @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/http...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.28.0...
- Installed hashicorp/aws v6.28.0 (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.
○ @Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $
```

User: user2

Access: AKIAVHLXGUTLCT2TNU6C

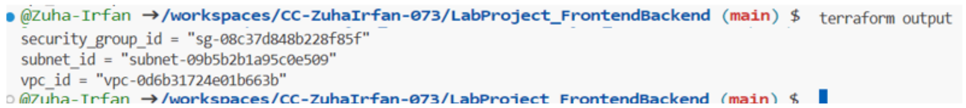
Secret: UObA4XHRIs1mQI4QENMQC2lK9A84V3F5evch8Yun

Terraform apply:



```
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  terraform apply -auto-approve
          + "Name" = "cc-lab-vpc"
        }
      + tags_all                             = {
          + "Name" = "cc-lab-vpc"
        }
    }

Plan: 6 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + security_group_id = (known after apply)
  + subnet_id         = (known after apply)
  + vpc_id            = (known after apply)
aws_vpc.main: Creating...
aws_vpc.main: Creation complete after 2s [id=vpc-0d6b31724e01b663b]
aws_subnet.public: Creating...
aws_internet_gateway.igw: Creating...
aws_security_group.web_sg: Creating...
aws_internet_gateway.igw: Creation complete after 1s [id=igw-0f25f75de51c23487]
aws_route_table.public_rt: Creating...
aws_route_table.public_rt: Creation complete after 1s [id=rtb-075b9195125a02de2]
aws_security_group.web_sg: Creation complete after 3s [id=sg-08c37d848b228f85f]
aws_subnet.public: Still creating... [10s elapsed]
aws_subnet.public: Creation complete after 11s [id=subnet-09b5b2b1a95c0e509]
aws_route_table_association.public_assoc: Creating...
aws_route_table_association.public_assoc: Creation complete after 1s [id=rtbassoc-0b155c80b646f5ed1]

Apply complete! Resources: 6 added, 0 changed, 0 destroyed.

Outputs:

security_group_id = "sg-08c37d848b228f85f"
subnet_id = "subnet-09b5b2b1a95c0e509"
vpc_id = "vpc-0d6b31724e01b663b"
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $
```

Terraform Output:



```
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  terraform output
security_group_id = "sg-08c37d848b228f85f"
subnet_id = "subnet-09b5b2b1a95c0e509"
vpc_id = "vpc-0d6b31724e01b663b"
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $
```

Terraform – Frontend & Backend EC2 Instances

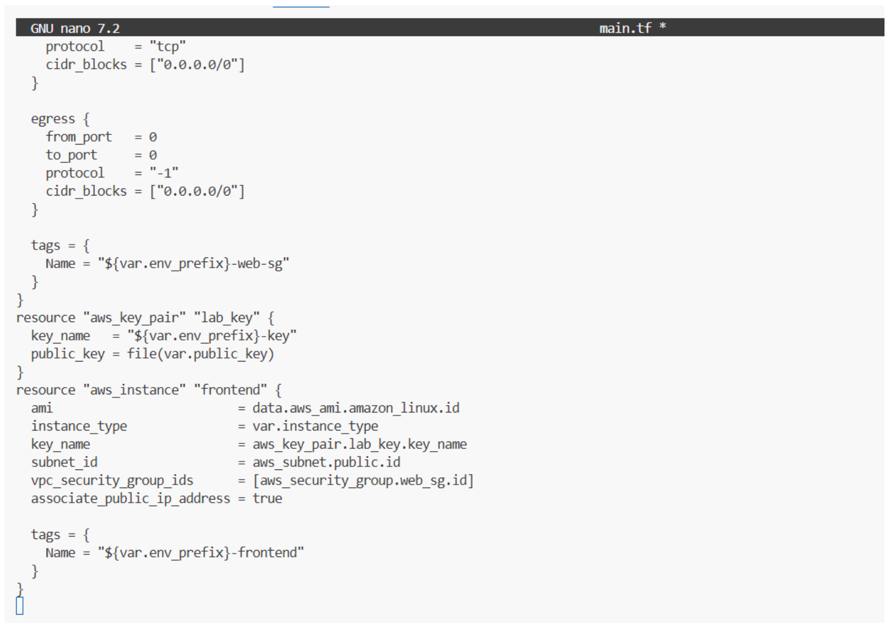
Requirements

Create AWS Key Pair in Terraform

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

Frontend EC2 Instance:

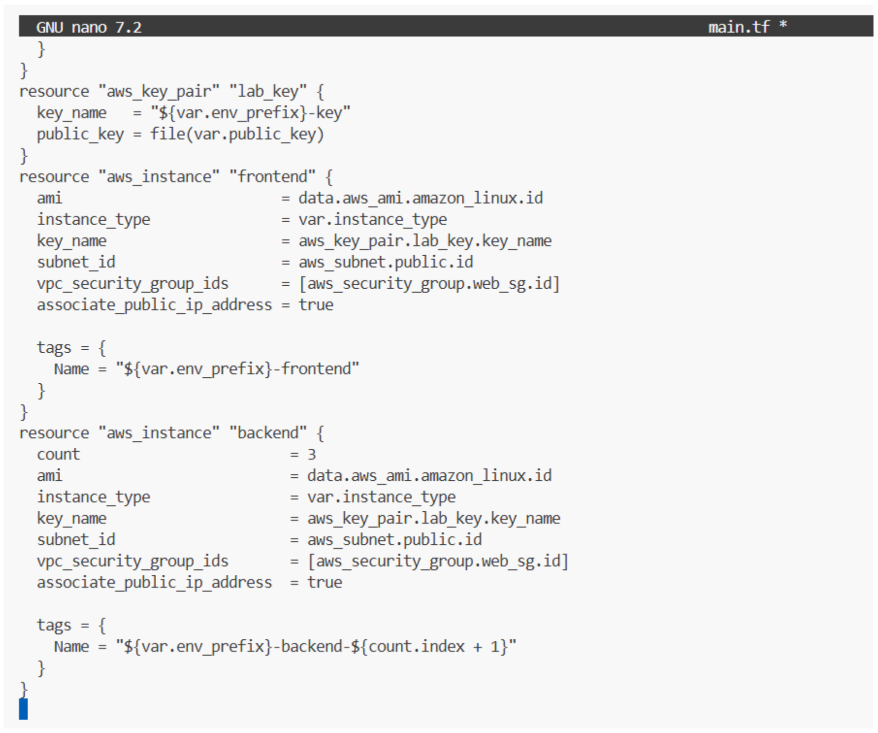
1. Use aws_instance or a webserver module.
2. Tag: Name = "${var.env_prefix}-frontend".
3. Attach security group.
4. Use SSH key pair from var.public_key.



```
GNU nano 7.2                                                          main.tf *
    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 = "${var.env_prefix}-web-sg"
  }
}
resource "aws_key_pair" "lab_key" {
  key_name   = "${var.env_prefix}-key"
  public_key = file(var.public_key)
}
resource "aws_instance" "frontend" {
  ami                         = data.aws_ami.amazon_linux.id
  instance_type               = var.instance_type
  key_name                    = aws_key_pair.lab_key.key_name
  subnet_id                   = aws_subnet.public.id
  vpc_security_group_ids      = [aws_security_group.web_sg.id]
  associate_public_ip_address = true

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

Backend EC2 Instances:

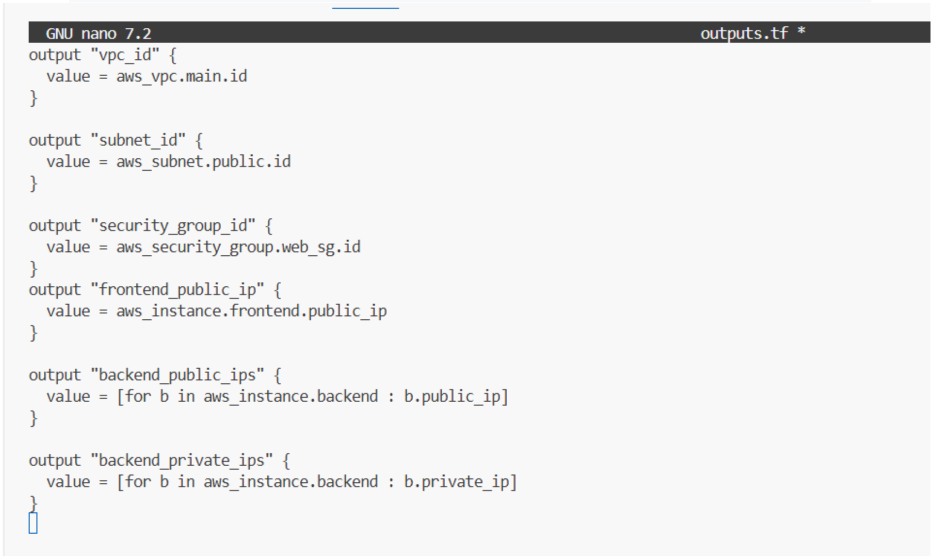
1. Use a module or count = 3.
2. Tag each as Name = "${var.env_prefix}-backend-${count.index}".
3. Use same security group or a dedicated backend SG (optional).
4. Expose their private IPs and public IPs via outputs.tf.



```
GNU nano 7.2                                                          main.tf *
  }
}
resource "aws_key_pair" "lab_key" {
  key_name   = "${var.env_prefix}-key"
  public_key = file(var.public_key)
}
resource "aws_instance" "frontend" {
  ami                         = data.aws_ami.amazon_linux.id
  instance_type               = var.instance_type
  key_name                    = aws_key_pair.lab_key.key_name
  subnet_id                   = aws_subnet.public.id
  vpc_security_group_ids      = [aws_security_group.web_sg.id]
  associate_public_ip_address = true

  tags = {
    Name = "${var.env_prefix}-frontend"
  }
}
resource "aws_instance" "backend" {
  count                        = 3
  ami                          = data.aws_ami.amazon_linux.id
  instance_type                = var.instance_type
  key_name                     = aws_key_pair.lab_key.key_name
  subnet_id                    = aws_subnet.public.id
  vpc_security_group_ids       = [aws_security_group.web_sg.id]
  associate_public_ip_address  = true

  tags = {
    Name = "${var.env_prefix}-backend-${count.index + 1}"
  }
}
```

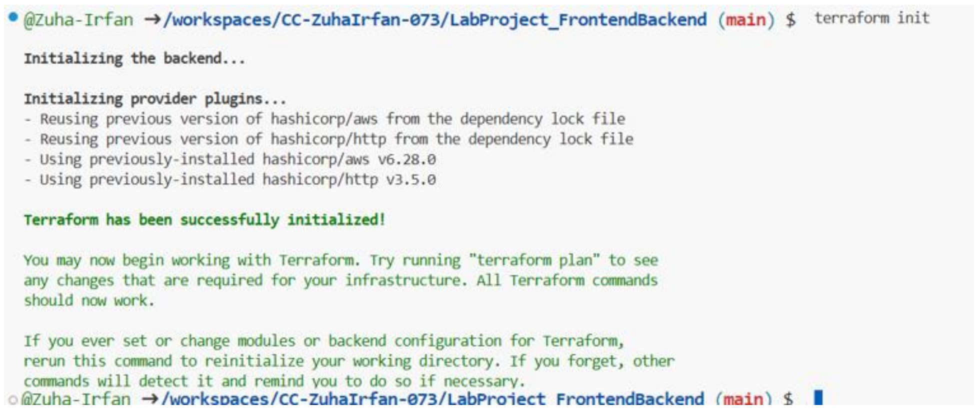
Example Outputs:

```
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]
}
```



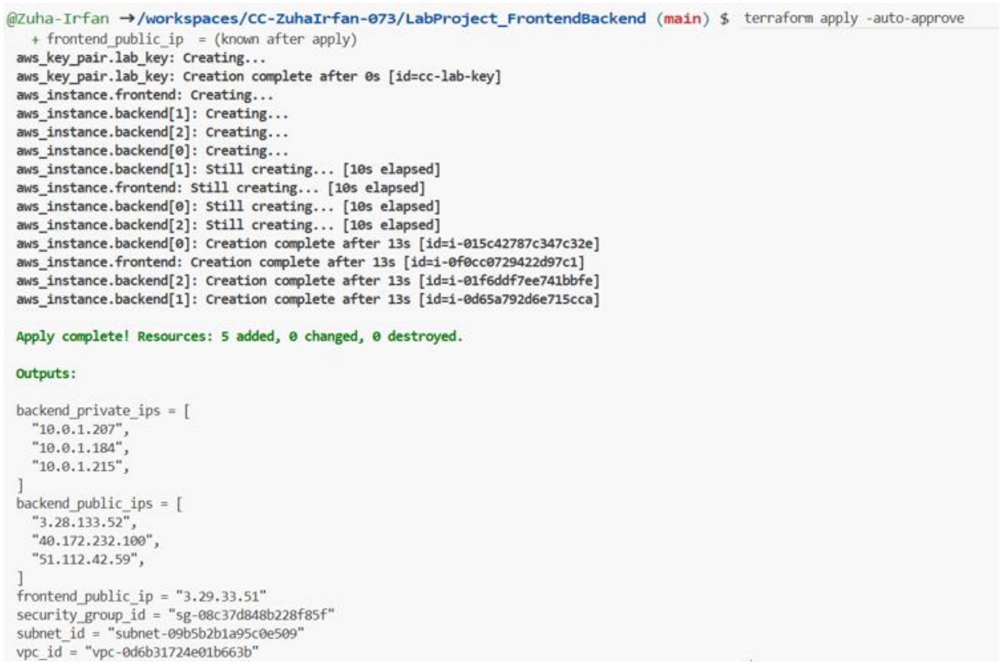
```
  GNU nano 7.2                                              outputs.tf *
output "vpc_id" {
  value = aws_vpc.main.id
}

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

output "security_group_id" {
  value = aws_security_group.web_sg.id
}
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]
}
```

Terraform init:



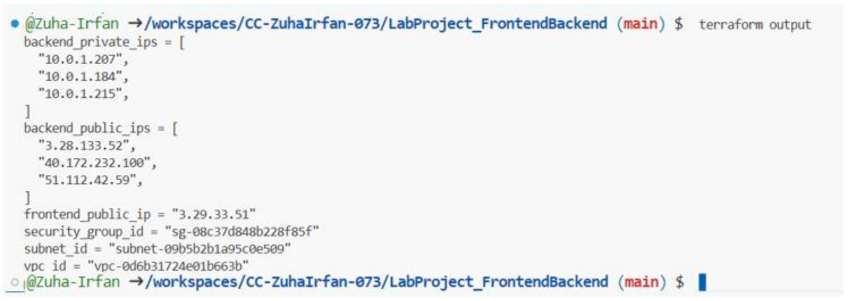
```
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  terraform init

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.
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $
```

Terraform apply:



```
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ terraform apply -auto-approve
  + frontend_public_ip  = (known after apply)
aws_key_pair.lab_key: Creating...
aws_key_pair.lab_key: Creation complete after 0s [id=cc-lab-key]
aws_instance.frontend: Creating...
aws_instance.backend[1]: Creating...
aws_instance.backend[2]: Creating...
aws_instance.backend[0]: Creating...
aws_instance.backend[1]: Still creating... [10s elapsed]
aws_instance.frontend: Still creating... [10s elapsed]
aws_instance.backend[0]: Still creating... [10s elapsed]
aws_instance.backend[2]: Still creating... [10s elapsed]
aws_instance.backend[0]: Creation complete after 13s [id=i-015c42787c347c32e]
aws_instance.frontend: Creation complete after 13s [id=i-0f0cc0729422d97c1]
aws_instance.backend[2]: Creation complete after 13s [id=i-01f6ddf7ee741bbfe]
aws_instance.backend[1]: Creation complete after 13s [id=i-0d65a792d6e715cca]

Apply complete! Resources: 5 added, 0 changed, 0 destroyed.

Outputs:

backend_private_ips = [
  "10.0.1.207",
  "10.0.1.184",
  "10.0.1.215",
]
backend_public_ips = [
  "3.28.133.52",
  "40.172.232.100",
  "51.112.42.59",
]
frontend_public_ip = "3.29.33.51"
security_group_id = "sg-08c37d848b228f85f"
subnet_id = "subnet-09b5b2b1a95c0e509"
vpc_id = "vpc-0d6b31724e01b663b"
```

Terraform output:



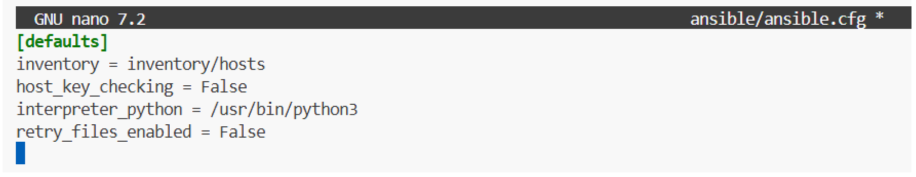
```
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ terraform output
backend_private_ips = [
  "10.0.1.207",
  "10.0.1.184",
  "10.0.1.215",
]
backend_public_ips = [
  "3.28.133.52",
  "40.172.232.100",
  "51.112.42.59",
]
frontend_public_ip = "3.29.33.51"
security_group_id = "sg-08c37d848b228f85f"
subnet_id = "subnet-09b5b2b1a95c0e509"
vpc_id = "vpc-0d6b31724e01b663b"
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $
```

<u>Ansible – Global Config & Inventory</u>

Requirements:

ansible/ansible.cfg must at least include:

```
[defaults]host_key_checking = Falseinterpreter_python = /usr/bin/python3
```



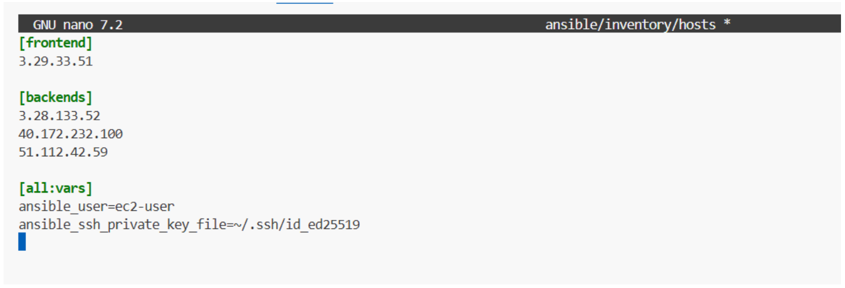
```
GNU nano 7.2                                                  ansible/ansible.cfg *
[defaults]
inventory = inventory/hosts
host_key_checking = False
interpreter_python = /usr/bin/python3
retry_files_enabled = False
```

Inventory (ansible/inventory/hosts) – choose:

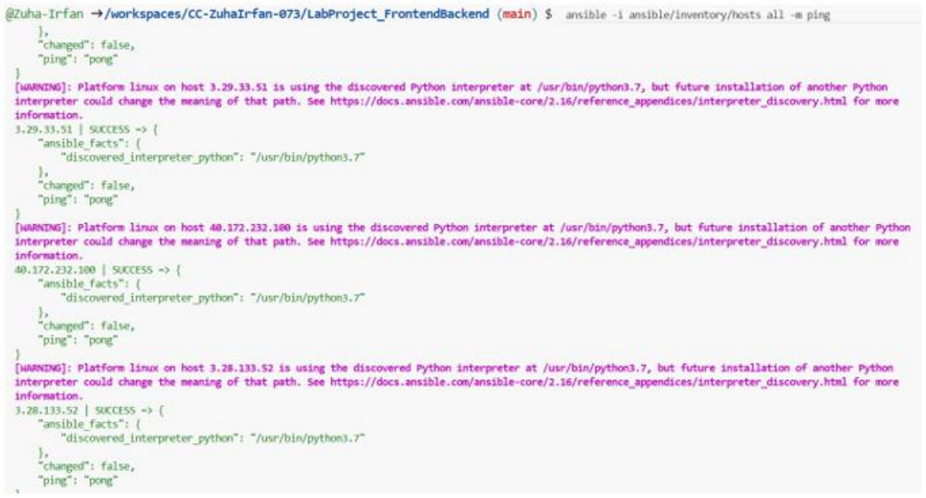
1. Static inventory filled manually using terraform output, or
2. Inline inventory via Terraform -i 'ip1,ip2,ip3'.

For easier marking, we recommend a static file:

```
[frontend]
<frontend-public-ip>
[backends]
<backend1-public-ip>
<backend2-public-ip>
<backend3-public-ip>
[all:vars]ansible_user=ec2-useransible_ssh_private_key_file=~/.ssh/id_ed25519
```

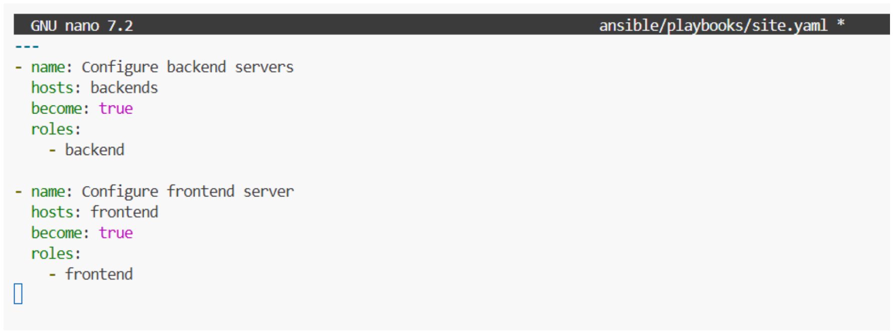


```
GNU nano 7.2                                   ansible/inventory/hosts *
[frontend]
3.29.33.51

[backends]
3.28.133.52
40.172.232.100
51.112.42.59

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



```
@Zuha-Irfan ➜/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  ansible -i ansible/inventory/hosts all -m ping
    },
    "changed": false,
    "ping": "pong"
}
[WARNING]: Platform linux on host 3.29.33.51 is using the discovered Python interpreter at /usr/bin/python3.7, but future installation of another Python
interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.16/reference_appendices/interpreter_discovery.html for more
information.
3.29.33.51 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3.7"
    },
    "changed": false,
    "ping": "pong"
}
[WARNING]: Platform linux on host 40.172.232.100 is using the discovered Python interpreter at /usr/bin/python3.7, but future installation of another Python
interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.16/reference_appendices/interpreter_discovery.html for more
information.
40.172.232.100 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3.7"
    },
    "changed": false,
    "ping": "pong"
}
[WARNING]: Platform linux on host 3.28.133.52 is using the discovered Python interpreter at /usr/bin/python3.7, but future installation of another Python
interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.16/reference_appendices/interpreter_discovery.html for more
information.
3.28.133.52 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3.7"
    },
    "changed": false,
    "ping": "pong"
```

Create site.yaml:



```
GNU nano 7.2                                   ansible/playbooks/site.yaml *
---
- name: Configure backend servers
  hosts: backends
  become: true
  roles:
    - backend

- name: Configure frontend server
  hosts: frontend
  become: true
  roles:
    - frontend
```

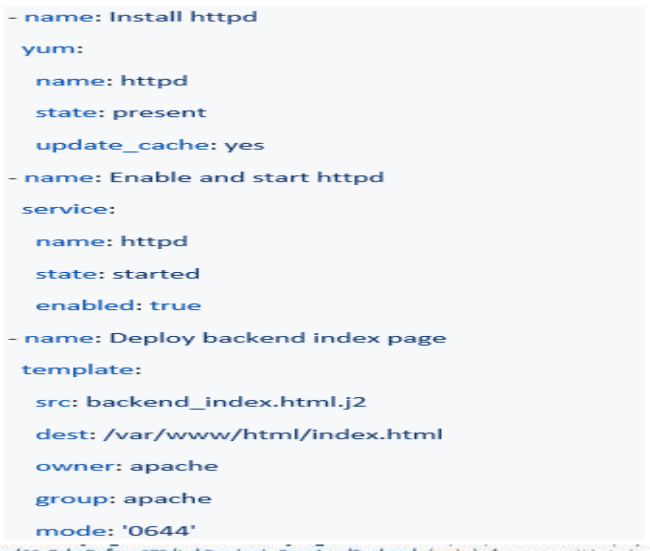
Ansible Roles – Backend HTTPD Role (backend/)

Backend Role Requirements (must be in ansible/roles/backend/):

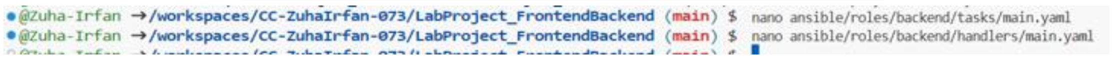
tasks/main.yml must:

1. Install Apache (httpd) and ensure service enabled & started.
2. Deploy a distinct index page per backend using a template.

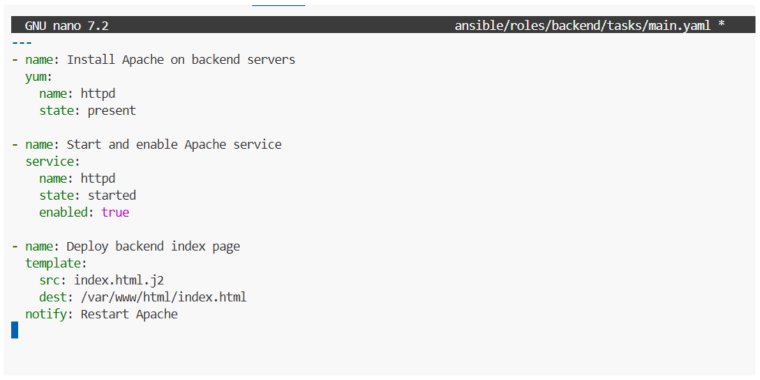
Example:



```
- 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'
```

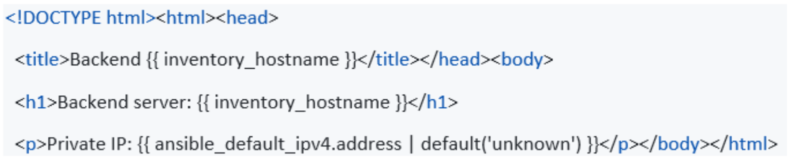


```
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ nano ansible/roles/backend/tasks/main.yaml
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ nano ansible/roles/backend/handlers/main.yaml
```

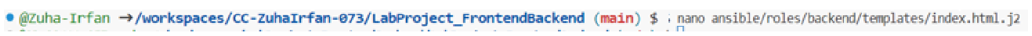


```
GNU nano 7.2                                       ansible/roles/backend/tasks/main.yaml *
---
- name: Install Apache on backend servers
  yum:
    name: httpd
    state: present

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

- name: Deploy backend index page
  template:
    src: index.html.j2
    dest: /var/www/html/index.html
  notify: Restart Apache
```

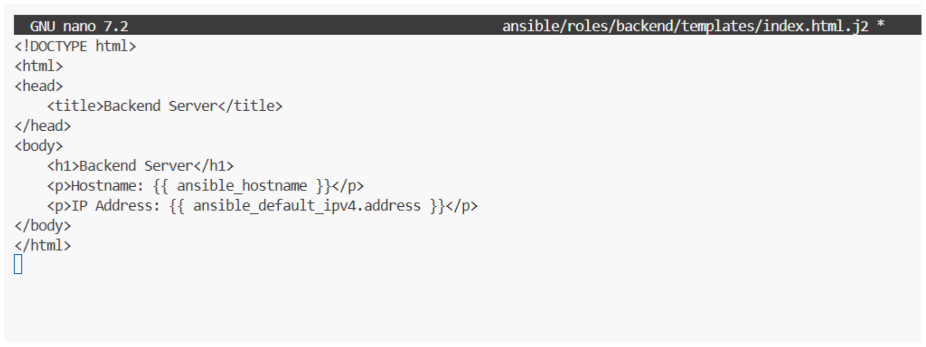
templates/backend_index.html.j2 must contain at least:



```
<!DOCTYPE html><html><head>
 <title>Backend {{ inventory_hostname }}</title></head><body>
 <h1>Backend server: {{ inventory_hostname }}</h1>
 <p>Private IP: {{ ansible_default_ipv4.address | default('unknown') }}</p></body></html>
```

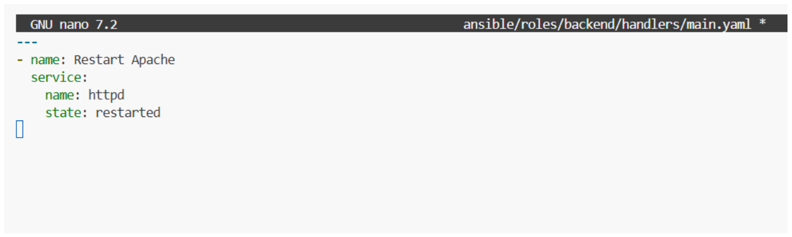


```
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ nano ansible/roles/backend/templates/index.html.j2
```

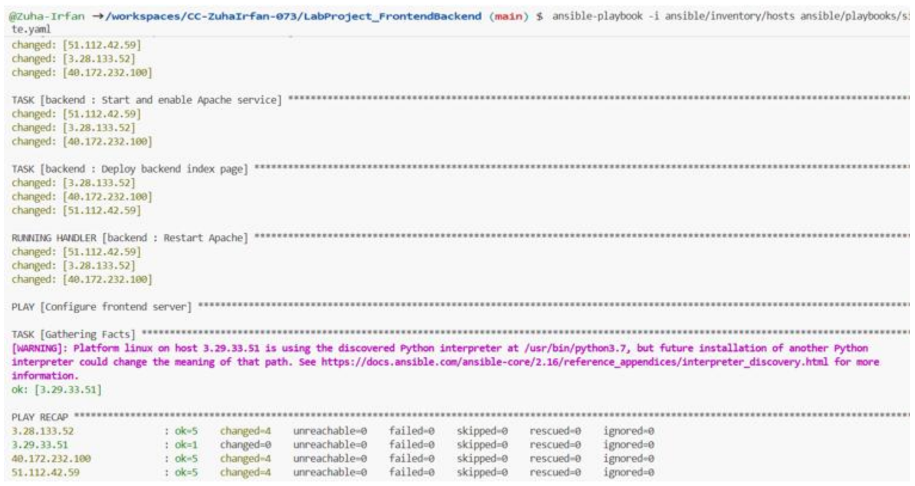


```
GNU nano 7.2                                   ansible/roles/backend/templates/index.html.j2 *
<!DOCTYPE html>
<html>
<head>
    <title>Backend Server</title>
</head>
<body>
    <h1>Backend Server</h1>
    <p>Hostname: {{ ansible_hostname }}</p>
    <p>IP Address: {{ ansible_default_ipv4.address }}</p>
</body>
</html>
```

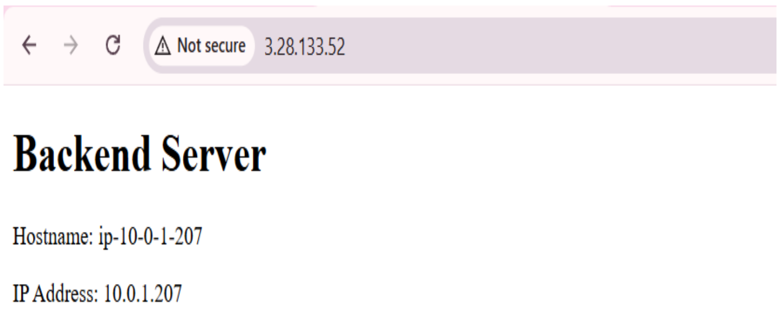
If you need restart behavior, configure handlers/main.yml (optional) to restart httpd when template changes.



```
GNU nano 7.2                                   ansible/roles/backend/handlers/main.yaml *
---
- name: Restart Apache
  service:
    name: httpd
    state: restarted
```



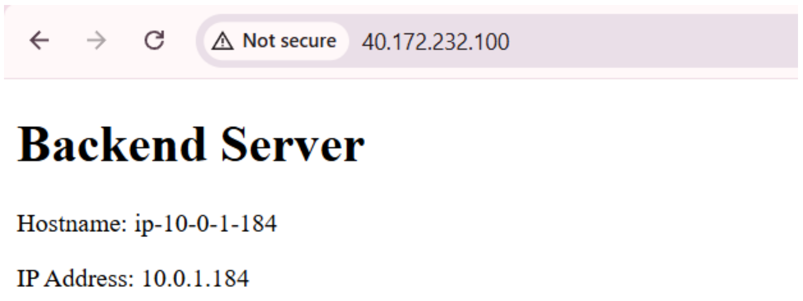
```
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ ansible-playbook -i ansible/inventory/hosts ansible/playbooks/si
te.yaml
changed: [51.112.42.59]
changed: [3.28.133.52]
changed: [40.172.232.100]

TASK [backend : Start and enable Apache service] ****************************************************************
changed: [51.112.42.59]
changed: [3.28.133.52]
changed: [40.172.232.100]

TASK [backend : Deploy backend index page] **********************************************************************
changed: [3.28.133.52]
changed: [40.172.232.100]
changed: [51.112.42.59]

RUNNING HANDLER [backend : Restart Apache] **********************************************************************
changed: [51.112.42.59]
changed: [3.28.133.52]
changed: [40.172.232.100]

PLAY [Configure frontend server] ********************************************************************************

TASK [Gathering Facts] ******************************************************************************************
[WARNING]: Platform linux on host 3.29.33.51 is using the discovered Python interpreter at /usr/bin/python3.7, but future installation of another Python
interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.16/reference_appendices/interpreter_discovery.html for more
information.
ok: [3.29.33.51]

PLAY RECAP ******************************************************************************************************
3.28.133.52                : ok=5    changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
3.29.33.51                 : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
40.172.232.100             : ok=5    changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
51.112.42.59               : ok=5    changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```



Backend Server

Hostname: ip-10-0-1-207

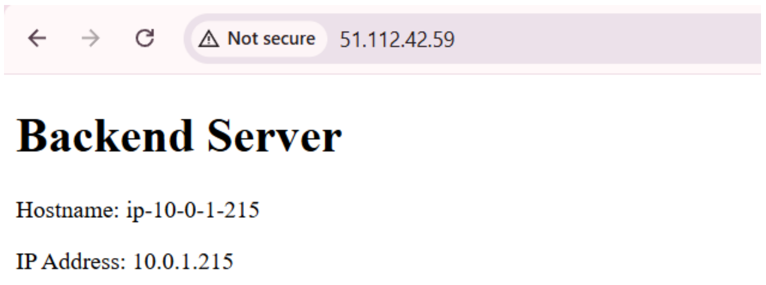
IP Address: 10.0.1.207



Backend Server

Hostname: ip-10-0-1-184

IP Address: 10.0.1.184



Backend Server

Hostname: ip-10-0-1-215

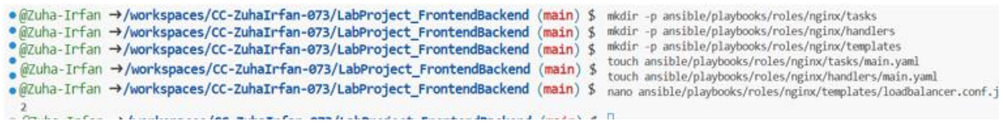
IP Address: 10.0.1.215

Ansible Roles – Frontend Nginx Role (frontend/)

Frontend Role Requirements (must be in ansible/roles/frontend/):

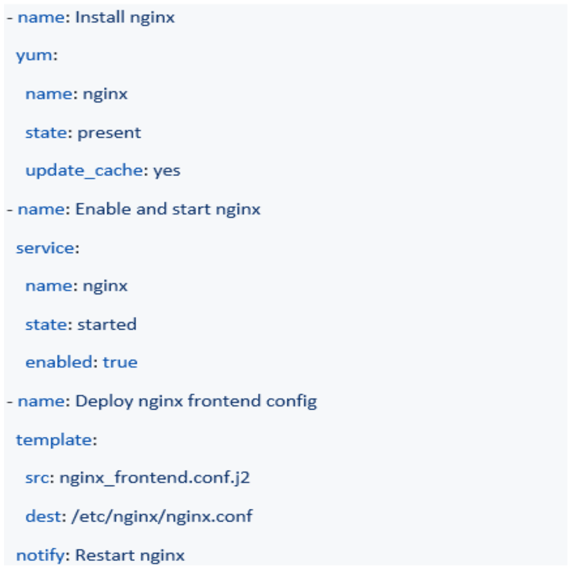
tasks/main.yml must:

1. Install Nginx.
2. Enable and start Nginx.
3. Deploy Nginx config via template nginx_frontend.conf.j2.
4. Use backend private IPs in Nginx upstream (not public IPs).

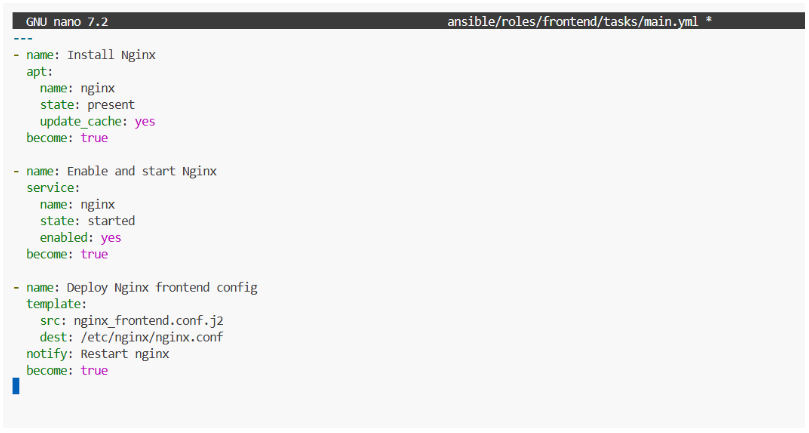


```
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ mkdir -p ansible/playbooks/roles/nginx/tasks
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ mkdir -p ansible/playbooks/roles/nginx/handlers
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ mkdir -p ansible/playbooks/roles/nginx/templates
                                                                               touch ansible/playbooks/roles/nginx/tasks/main.yaml
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ touch ansible/playbooks/roles/nginx/handlers/main.yaml
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ nano ansible/playbooks/roles/nginx/templates/loadbalancer.conf.j
2
```

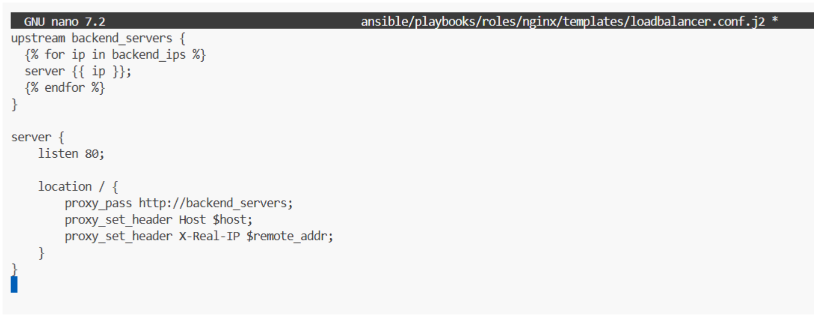
Example (outline):



```yaml
- 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
 notify: Restart nginx
```

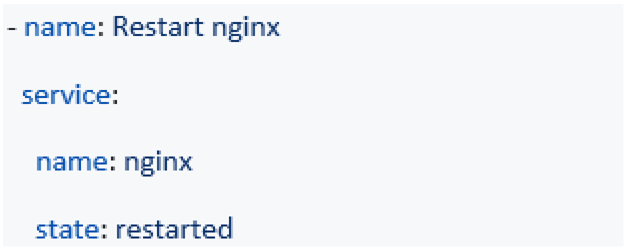


```
 GNU nano 7.2                                    ansible/roles/frontend/tasks/main.yml *
---
- name: Install Nginx
  apt:
    name: nginx
    state: present
    update_cache: yes
  become: true

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

- name: Deploy Nginx frontend config
  template:
    src: nginx_frontend.conf.j2
    dest: /etc/nginx/nginx.conf
  notify: Restart nginx
  become: true
```

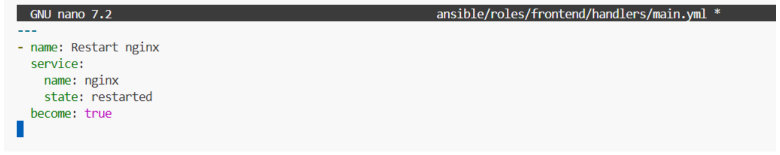


```
 GNU nano 7.2                          ansible/playbooks/roles/nginx/templates/loadbalancer.conf.j2 *
upstream backend_servers {
  {% for ip in backend_ips %}
  server {{ ip }};
  {% endfor %}
}

server {
    listen 80;

    location / {
        proxy_pass http://backend_servers;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
    }
}
```

handlers/main.yml:



```yaml
- name: Restart nginx
 service:
  name: nginx
  state: restarted
```



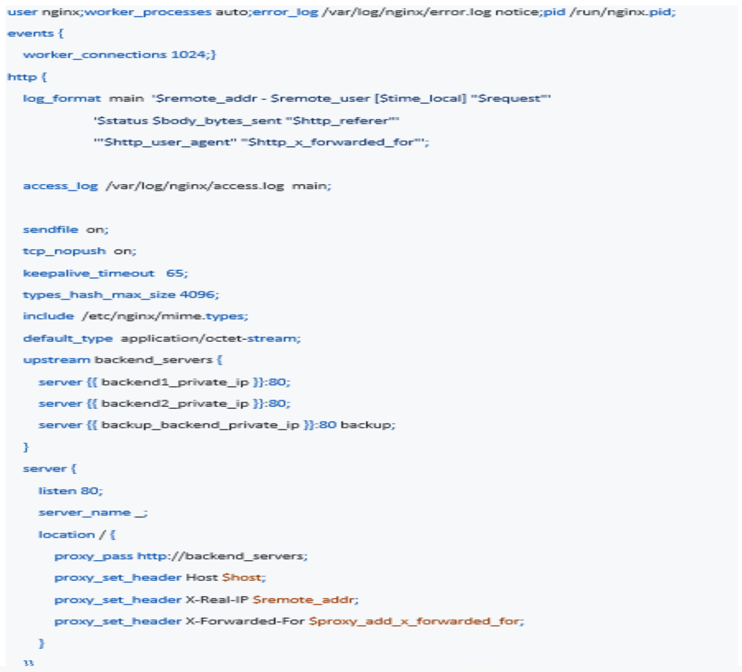
```
GNU nano 7.2                                   ansible/roles/frontend/handlers/main.yml *
---
- name: Restart nginx
  service:
    name: nginx
    state: restarted
  become: true
```

templates/nginx_frontend.conf.j2 must:

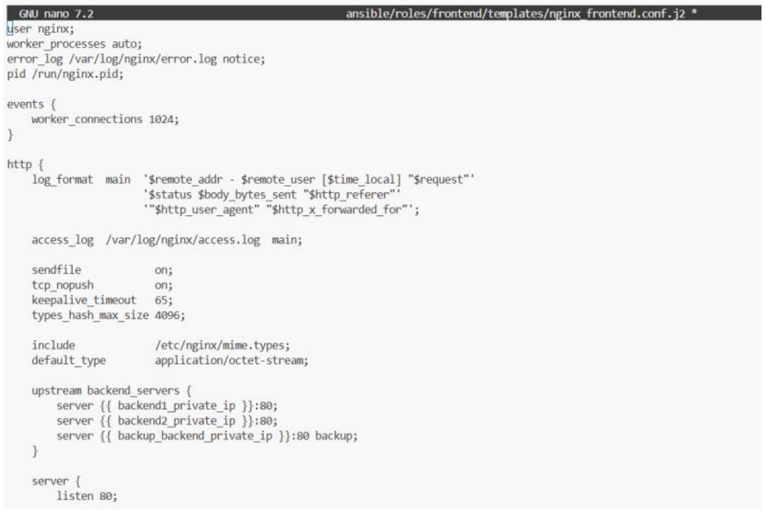
Define upstream with all 3 backends:

1. Two normal servers.
2. One with backup.

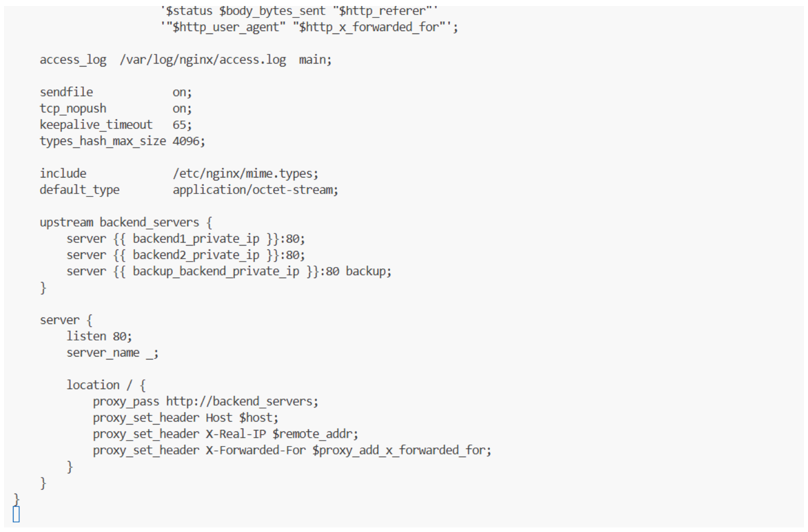
Example:



```
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;
  upstream backend_servers {
    server {{ backend1_private_ip }}:80;
    server {{ backend2_private_ip }}:80;
    server {{ backup_backend_private_ip }}:80 backup;
  }
  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;
    }
```



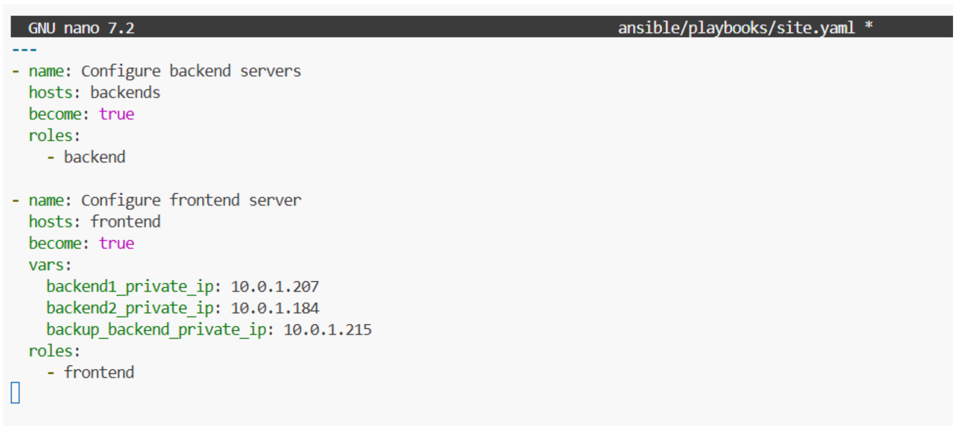
```
GNU nano 7.2                        ansible/roles/frontend/templates/nginx_frontend.conf.j2 *
user nginx;
worker_processes auto;
error_log /var/log/nginx/error.log notice;
pid /run/nginx.pid;

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;

    upstream backend_servers {
        server {{ backend1_private_ip }}:80;
        server {{ backend2_private_ip }}:80;
        server {{ backup_backend_private_ip }}:80 backup;
    }

    server {
        listen 80;
```



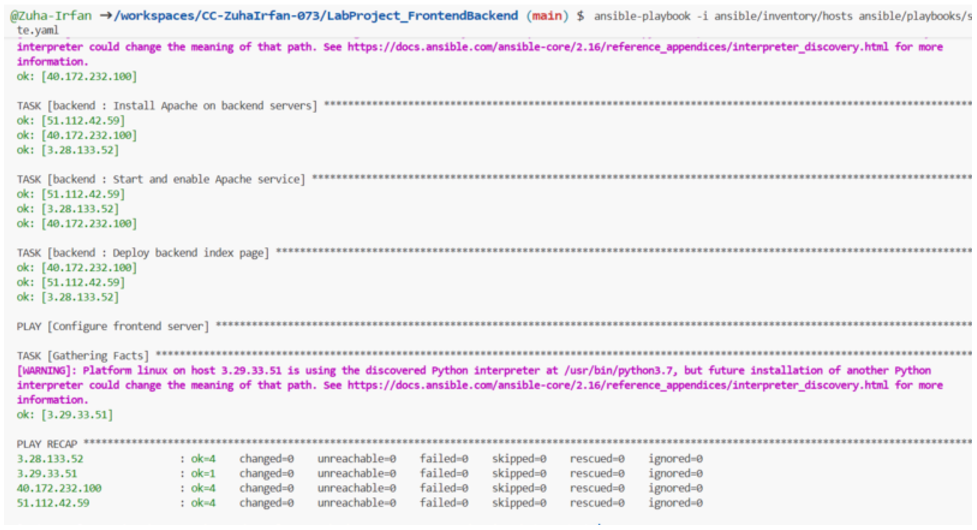
```
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    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;

    upstream backend_servers {
        server {{ backend1_private_ip }}:80;
        server {{ backend2_private_ip }}:80;
        server {{ backup_backend_private_ip }}:80 backup;
    }

    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;
        }
    }
}
```

You must pass in backend1_private_ip, backend2_private_ip, and backup_backend_private_ip into the role via variables or facts.

ansible-playbook -i ansible/inventory/hosts ansible/playbooks/site.yaml



```
  GNU nano 7.2                                          ansible/playbooks/site.yaml *
---
- name: Configure backend servers
  hosts: backends
  become: true
  roles:
    - backend

- name: Configure frontend server
  hosts: frontend
  become: true
  vars:
    backend1_private_ip: 10.0.1.207
    backend2_private_ip: 10.0.1.184
    backup_backend_private_ip: 10.0.1.215
  roles:
    - frontend
```



```
@Zuha-Irfan →/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $ ansible-playbook -i ansible/inventory/hosts ansible/playbooks/s
te.yaml
interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.16/reference_appendices/interpreter_discovery.html for more
information.
ok: [40.172.232.100]

TASK [backend : Install Apache on backend servers] ****************************************************************************************
ok: [51.112.42.59]
ok: [40.172.232.100]
ok: [3.28.133.52]

TASK [backend : Start and enable Apache service] ******************************************************************************************
ok: [51.112.42.59]
ok: [3.28.133.52]
ok: [40.172.232.100]

TASK [backend : Deploy backend index page] ************************************************************************************************
ok: [40.172.232.100]
ok: [51.112.42.59]
ok: [3.28.133.52]

PLAY [Configure frontend server] **********************************************************************************************************

TASK [Gathering Facts] ********************************************************************************************************************
[WARNING]: Platform linux on host 3.29.33.51 is using the discovered Python interpreter at /usr/bin/python3.7, but future installation of another Python
interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.16/reference_appendices/interpreter_discovery.html for more
information.
ok: [3.29.33.51]

PLAY RECAP ********************************************************************************************************************************
3.28.133.52                : ok=4    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
3.29.33.51                 : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
40.172.232.100             : ok=4    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
51.112.42.59               : ok=4    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Ansible Main Playbook Using Roles

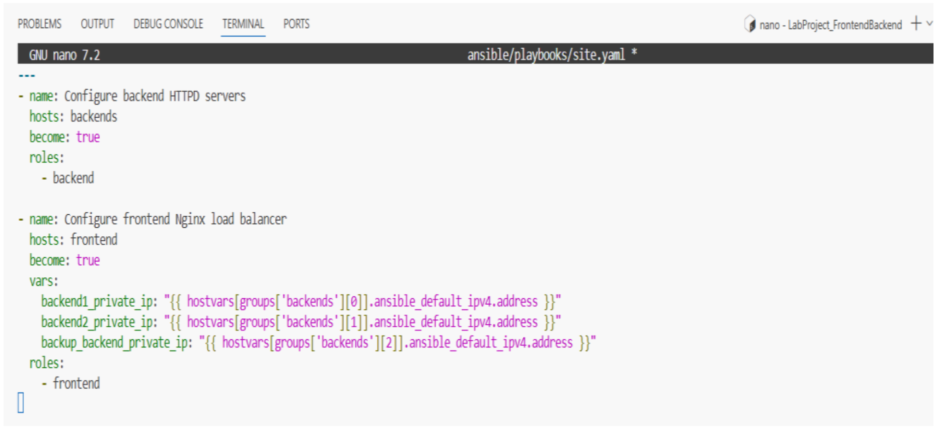
Your main playbook e.g. ansible/playbooks/site.yaml must use roles. For example:

```
- 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: "<private-ip-backend-1 or from hostvars>"
    backend2_private_ip: "<private-ip-backend-2 or from hostvars>"
    backup_backend_private_ip: "<private-ip-backend-3 or from hostvars>"
  roles:
    - frontend
```

Better is to dynamically gather backend IPs from inventory using hostvars / groups in the play:

```
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 }}"
```

Using roles in this way is mandatory. If you do not use roles for frontend and backend, you will lose most or all marks from Section B.



```
 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
```

Terraform–Ansible Integration (Automation)

Requirement:

Terraform must trigger Ansible automatically after EC2 instances are ready.

Implementation hint:

Add a null_resource to main.tf that:

- Has depends_on for frontend and backend instances.
- Has triggers that include instance IPs.
- Uses local-exec to run Ansible.

Example:

```
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-playbook \      -i ../generated_hosts.ini \      playbooks/site.yaml    EOT
 }}
```

You can either:

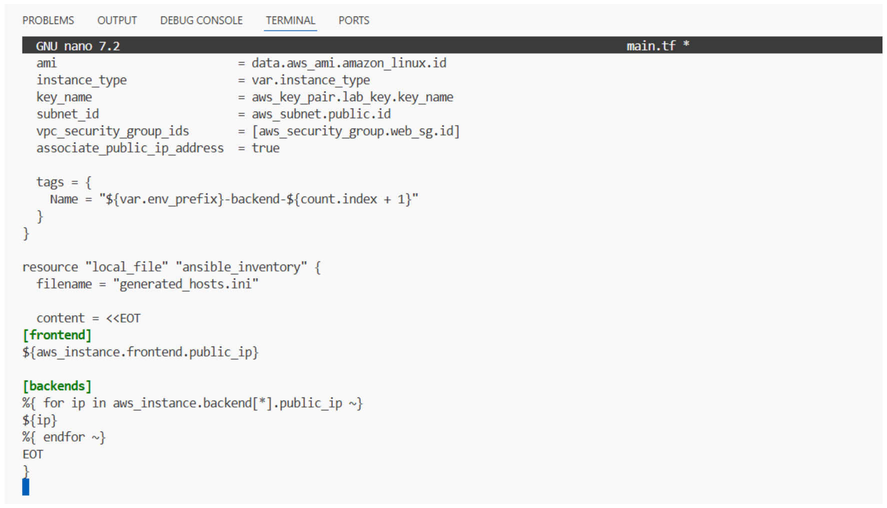
- Generate generated_hosts.ini via a Terraform template file (templatefile function + local_file resource), or
- Directly use inline inventory; e.g.,

```
provisioner "local-exec" {
 command = <<-EOT    cd ansible    ANSIBLE_HOST_KEY_CHECKING=False ansible-playbook \     -i
"${self.triggers.frontend_ip}," \     playbooks/site.yaml  EOT
}
```

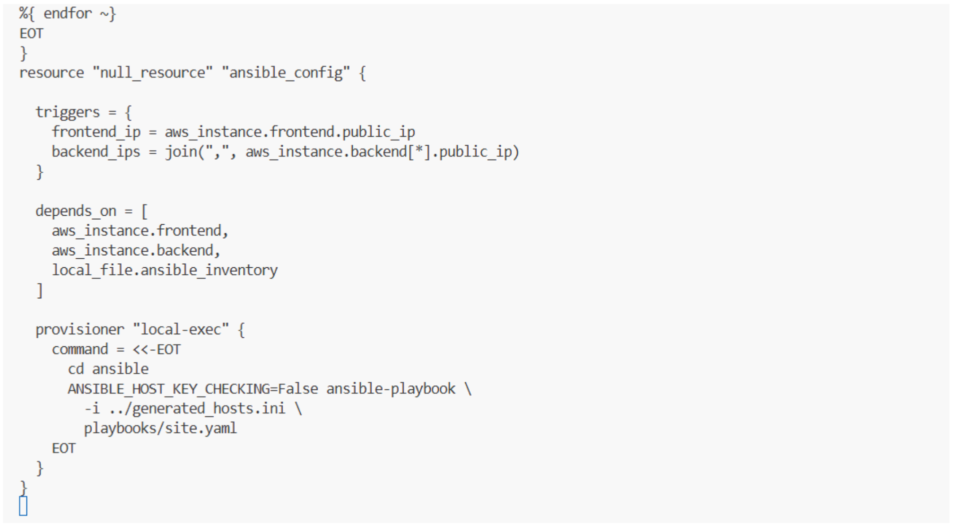
(But you need all 4 hosts; so static inventory + playbook is usually easier.)

Key evaluation criteria:

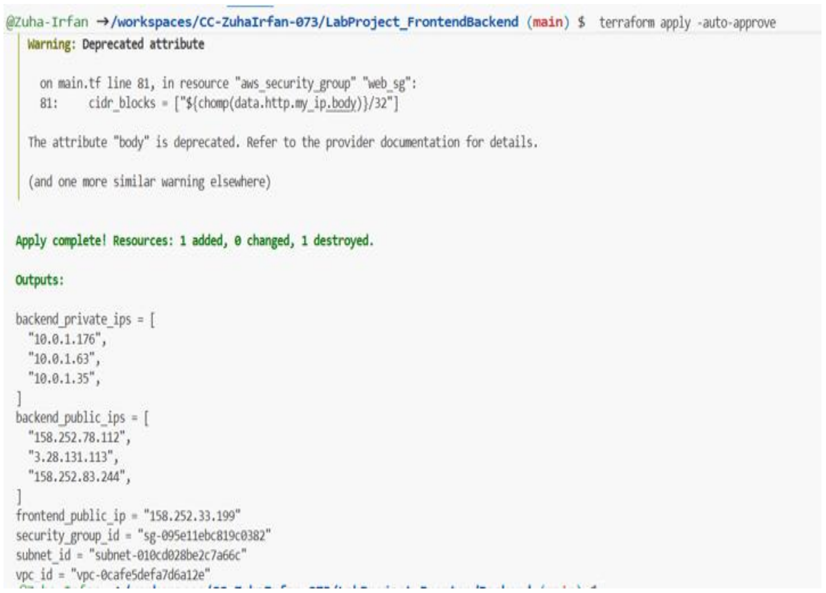
- After terraform apply -auto-approve, do not manually call ansible-playbook.
- All configuration must be applied already.



```
  GNU nano 7.2                                                                         main.tf *
  ami                           = data.aws_ami.amazon_linux.id
  instance_type                 = var.instance_type
  key_name                      = aws_key_pair.lab_key.key_name
  subnet_id                     = aws_subnet.public.id
  vpc_security_group_ids        = [aws_security_group.web_sg.id]
  associate_public_ip_address   = true

  tags = {
    Name = "${var.env_prefix}-backend-${count.index + 1}"
  }
}

resource "local_file" "ansible_inventory" {
  filename = "generated_hosts.ini"

  content = <<EOT
[frontend]
${aws_instance.frontend.public_ip}

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



```
%{ endfor ~}
EOT
}
resource "null_resource" "ansible_config" {

  triggers = {
    frontend_ip = aws_instance.frontend.public_ip
    backend_ips = join(",", aws_instance.backend[*].public_ip)
  }

  depends_on = [
    aws_instance.frontend,
    aws_instance.backend,
    local_file.ansible_inventory
  ]

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

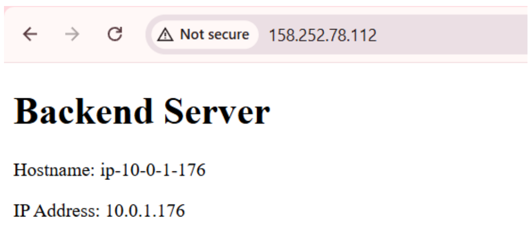


```
@Zuha-Irfan ➜/workspaces/CC-ZuhaIrfan-073/LabProject_FrontendBackend (main) $  terraform apply -auto-approve
│ Warning: Deprecated attribute
│
│   on main.tf line 81, in resource "aws_security_group" "web_sg":
│   81:     cidr_blocks = ["${chomp(data.http.my_ip.body)}/32"]
│
│ The attribute "body" is deprecated. Refer to the provider documentation for details.
│
│ (and one more similar warning elsewhere)


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

Outputs:

backend_private_ips = [
  "10.0.1.176",
  "10.0.1.63",
  "10.0.1.35",
]
backend_public_ips = [
  "158.252.78.112",
  "3.28.131.113",
  "158.252.83.244",
]
frontend_public_ip = "158.252.33.199"
security_group_id = "sg-095e11ebc819c0382"
subnet_id = "subnet-010cd028be2c7a66c"
vpc_id = "vpc-0cafe5defa7d6a12e"
```

RESULTS

- Backend ip:

http://158.252.78.112/



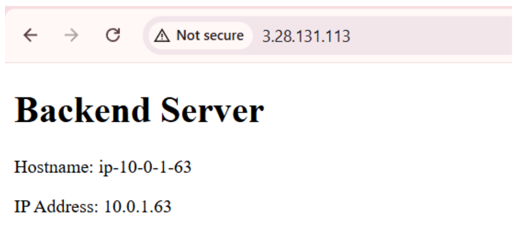
Not secure 158.252.78.112

Backend Server

Hostname: ip-10-0-1-176

IP Address: 10.0.1.176

http://3.28.131.113/



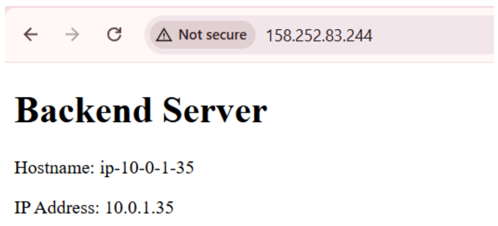
Not secure 3.28.131.113

Backend Server

Hostname: ip-10-0-1-63

IP Address: 10.0.1.63

http://158.252.83.244/



Not secure 158.252.83.244

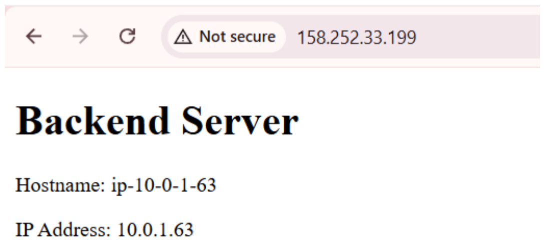
Backend Server

Hostname: ip-10-0-1-35

IP Address: 10.0.1.35

- Frontend Ip:

http://158.252.33.199/



Not secure 158.252.33.199

Backend Server

Hostname: ip-10-0-1-63

IP Address: 10.0.1.63