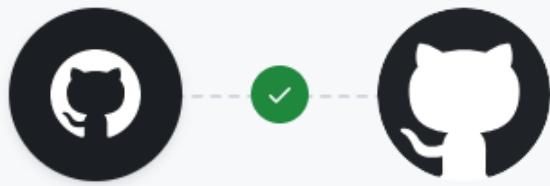


**LAB 12**  
**CLOUD COMPUTING**

SUBMITTED BY:  
AREEJ FATIMA  
BSE-2023-010  
SEMESTER VA



## Authorize GitHub CLI

This authorization was requested from Rawalpindi 125.209.64.164 on December 30th, 2025 at 16:21 (PKT)  
Make sure you trust this device as it will get access to your account.



**GitHub CLI by GitHub**

wants to access your areej-10 account



**Gists**

Read and write access



**Organizations and teams**

Read-only access

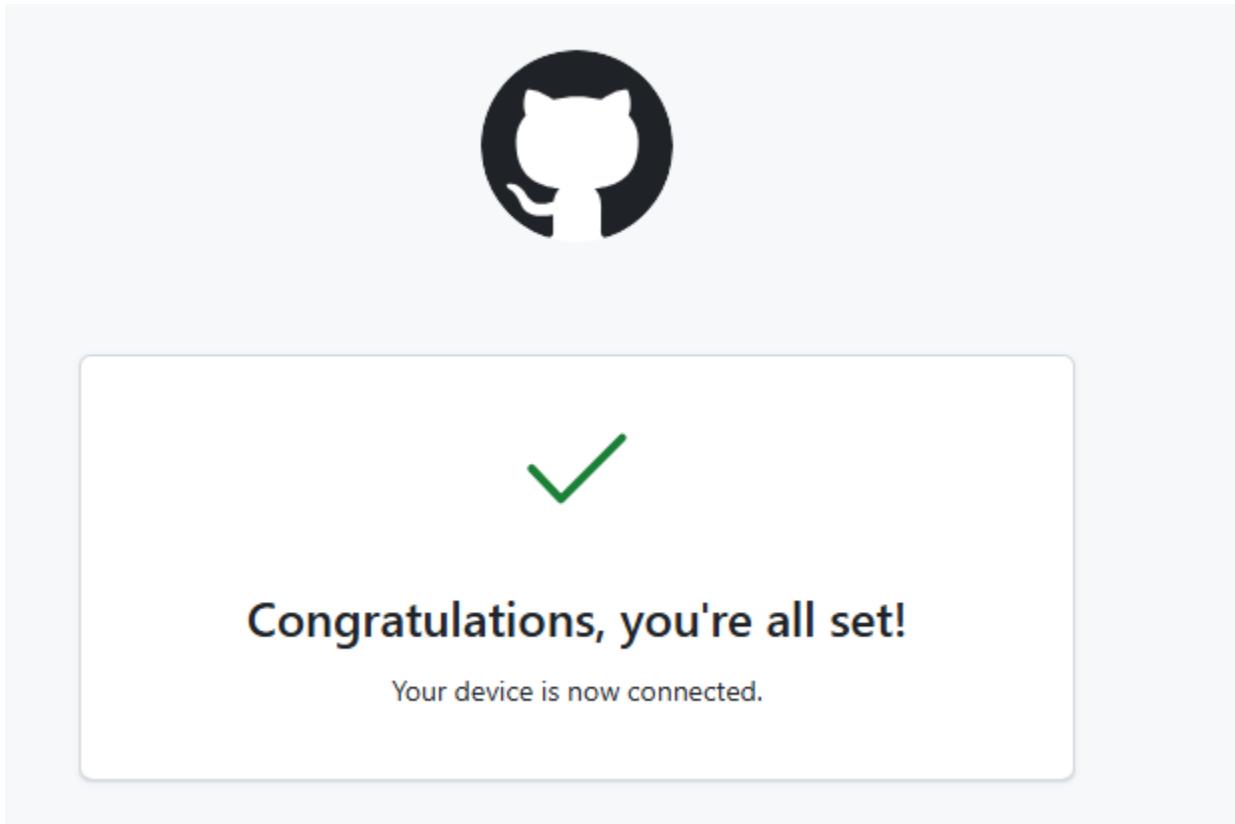


**Repositories**

Public and private



**Workflow**



```
Dell@DESKTOP-JNCVEJH MINGW64 ~ (main)
$ gh repo create areej-10/CC_areej_010_Lab12 --public
/ Created repository areej-10/CC_areej_010_Lab12 on github.com
https://github.com/areej-10/CC_areej_010_Lab12
```

```
Dell@DESKTOP-JNCVEJH MINGW64 ~ (main)
$ gh repo create areej-10/CC_areej_010_Lab12 --public
GraphQL: Name already exists on this account (createRepository)
```

```
Dell@DESKTOP-JNCVEJH MINGW64 ~ (main)
$ |
```

task0\_codespace\_create\_and\_list.png

```
Dell@DESKTOP-JNCVEJH MINGW64 ~ (main)
$ gh codespace list
NAME      DISPLAY NAME  REPOSITORY  BRANCH  STATE  CREATED AT
urban-carnival-...  urban carnival  areej-10/are...  main  Shutdown  about 19 hou...
```

task0\_codespace\_ssh\_connected.png

```

Dell@DESKTOP-JNCVEJH MINGW64 ~ (main)
$ gh codespace ssh -c urban-carnival-r45r7xvwwxvh565g
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-1030-azure x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

@areej-10 → /workspaces/areej-10-lab6 (main) $ |

```

task1\_project\_directory.png

```

Dell@DESKTOP-JNCVEJH MINGW64 ~ (main)
$ # Go to your desired location (e.g., Documents)
cd ~/Documents

# Create project folder
mkdir -p CC_areej_10/Assignment2
cd CC_areej_10/Assignment2

Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ 

```

task1\_ssh\_keygen.png

```

Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ # Generate ED25519 key (more secure than RSA)
ssh-keygen -t ed25519 -C "areejfatimaa101@gmail.com" -f ~/.ssh/id_ed25519
Generating public/private ed25519 key pair.
/c/Users/Dell/.ssh/id_ed25519 already exists.
Overwrite (y/n)? y
Enter passphrase for "/c/Users/Dell/.ssh/id_ed25519" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/Dell/.ssh/id_ed25519
Your public key has been saved in /c/Users/Dell/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:8nRrIL11JnuuPOBPMpFQVnL2t2kn7qKd7M4tw9eWC5s areejfatimaa101@gmail.com
The key's randomart image is:
+--[ED25519 256]--+
|       .o+o+   |
|       .o+ .    |
|       . . .    |
|       .+ . . o |
|       o.S== = .|
|      =.X.. o o |
|      +.+. ....|
|      +.+==.o| |
|      .o+=BEoo.|
+---[SHA256]-----+

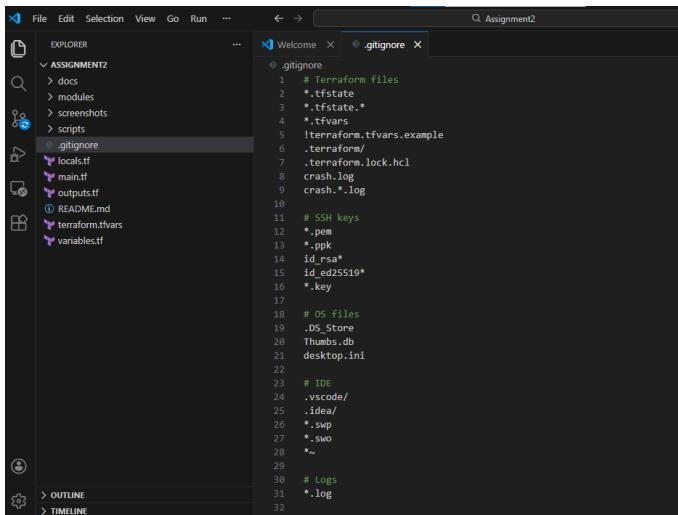
```

```
Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ ls -la ~/.ssh/
total 26
drwxr-xr-x 1 Dell 197121 0 Oct 24 12:38 ../
drwxr-xr-x 1 Dell 197121 0 Dec 28 16:09 ./
-rw-r--r-- 1 Dell 197121 419 Dec 28 16:13 id_ed25519
-rw-r--r-- 1 Dell 197121 107 Dec 28 16:13 id_ed25519.pub
-rw-r--r-- 1 Dell 197121 1674 Oct 24 12:38 known_hosts
-rw-r--r-- 1 Dell 197121 925 Oct 24 12:38 known_hosts.old

Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ cat ~/.ssh/id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAQAIcxgqdS2YYrtrFoZAnsYRnRk9xJ0DcgNim9hvWmJj7 areejfatimaa101@gmail
.com
```

task1\_files\_created.png

```
$ ls -R
.:
README.md  docs/  locals.tf  main.tf  modules/  outputs.tf  screenshots/  scripts/  terraform.tfvars  variables.tf
./docs:
architecture.md  troubleshooting.md
./modules:
networking/  security/  webserver/
./modules/networking:
main.tf  outputs.tf  variables.tf
./modules/security:
main.tf  outputs.tf  variables.tf
./modules/webserver:
main.tf  outputs.tf  variables.tf
./screenshots:
bonus/  part1/  part2/  part3/  part4/  part5/  part6/
./screenshots/bonus:
./screenshots/part1:
./screenshots/part2:
./screenshots/part3:
./screenshots/part4:
./screenshots/part5:
./screenshots/part6:
./scripts:
apache-setup.sh  nginx-setup.sh
```



task1\_variables\_tf.png

```
variables.tf
1  # VPC Configuration
2  variable "vpc_cidr_block" {
3    description = "CIDR block for VPC"
4    type        = string
5    default     = "10.0.0.0/16"
6
7    validation {
8      condition   = can(cidrhost(var.vpc_cidr_block, 0))
9      error_message = "Must be a valid IPv4 CIDR block."
10     }
11   }
12
13 variable "subnet_cidr_block" {
14   description = "CIDR block for subnet"
15   type        = string
16   default     = "10.0.10.0/24"
17
18   validation {
19     condition   = can(cidrhost(var.subnet_cidr_block, 0))
20     error_message = "Must be a valid IPv4 CIDR block."
21   }
22 }
23
24 variable "availability_zone" {
25   description = "AWS availability zone"
26   type        = string
27   default     = "me-central-1a"
28 }
29
30 # Environment Configuration
31 variable "env_prefix" {
32   description = "Environment prefix for resource naming"
```

The git repository a  
changes, only a sub

Source: Git

task1\_terraform\_tfvars.png

```
>Welcome .gitignore variables.tf terraform.tfvars X
terraform.tfvars
1 vpc_cidr_block = "10.0.0.0/16"
2 subnet_cidr_block = "10.0.10.0/24"
3 availability_zone = "me-central-1a"
4 env_prefix = "prod"
5 instance_type = "t3.micro"
6 public_key = "~/.ssh/id_ed25519.pub"
7 private_key = "~/.ssh/id_ed25519"
```

task1\_locals\_tf.png

```
locals {
  # Dynamically get your public IP
  my_ip = "${chomp(data.http.my_ip.response_body)}/32"

  # Common tags for all resources
  common_tags = []
  Environment = var.env_prefix
  Project     = "Assignment-2"
  ManagedBy   = "Terraform"
  CreatedBy   = "Areej Fatima"
}

# Backend server configurations
backend_servers = [
  {
    name      = "web-1"
    suffix    = "1"
    script_path = "./scripts/apache-setup.sh"
  },
  {
    name      = "web-2"
    suffix    = "2"
    script_path = "./scripts/apache-setup.sh"
  },
  {
    name      = "web-3"
    suffix    = "3"
    script_path = "./scripts/apache-setup.sh"
  }
]
```

task1\_variables\_tf.png

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

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

variable "availability_zone" {
  description = "Availability zone for subnet"
  type        = string
}

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

variable "common_tags" {
  description = "Common tags for all resources"
  type        = map(string)
  default     = {}
}
```

task1\_main\_tf.png

The screenshot shows a code editor interface with the title bar "Assignment2". The left sidebar lists files and folders: ".es6tf", ".terraform", ".locals", ".variables", ".networking", and ".main". The main editor area displays the "main.tf" file under the ".networking" folder. The code in "main.tf" is as follows:

```
modules > networking > main.tf
31 resource "aws_internet_gateway" "main" {
32   tags = merge(
33     var.common_tags,
34     {
35       Name = "${var.env_prefix}-igw"
36     }
37   )
38
39   # Create Route Table
40   resource "aws_route_table" "main" {
41     vpc_id = aws_vpc.main.id
42
43     route {
44       cidr_block = "0.0.0.0/0"
45       gateway_id = aws_internet_gateway.main.id
46     }
47
48     tags = merge(
49       var.common_tags,
50       {
51         Name = "${var.env_prefix}-rt"
52       }
53     )
54
55   # Associate Route Table with Subnet
56   resource "aws_route_table_association" "main" {
57     subnet_id = aws_subnet.main.id
58     route_table_id = aws_route_table.main.id
59   }
60 }
```

task1\_outputs\_tf.png

The screenshot shows a code editor interface with the title bar "Assignment2". The left sidebar lists files and folders: ".es6tf", ".terraform", ".locals", ".variables", ".networking", and ".outputs". The main editor area displays the "outputs.tf" file under the ".networking" folder. The code in "outputs.tf" is as follows:

```
modules > networking > outputs.tf
1 output "vpc_id" {
2   description = "ID of the VPC"
3   value       = aws_vpc.main.id
4 }
5
6 output "subnet_id" {
7   description = "ID of the subnet"
8   value       = aws_subnet.main.id
9 }
10
11 output "igw_id" {
12   description = "ID of the Internet Gateway"
13   value       = aws_internet_gateway.main.id
14 }
15
16 output "route_table_id" {
17   description = "ID of the route table"
18   value       = aws_route_table.main.id
19 }
```

task1\_entry\_script.png

```
scripts > $ apache-setup.sh
30 cat > /var/www/html/index.html <<EOF
31 <!DOCTYPE html>
32 <html>
33 <head>
34   <title>Backend Web Server - $HOSTNAME</title>
35   <style>
36     body {
37       font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif
38       margin: 0;
39       padding: 0;
40       background: linear-gradient(135deg, #667eea 0%, #764ba2 100%
41       min-height: 100vh;
42       display: flex;
43       justify-content: center;
44       align-items: center;
45     }
46     .container {
47       background: rgba(255, 255, 255, 0.95);
48       padding: 40px;
49       border-radius: 20px;
50       box-shadow: 0 20px 60px rgba(0, 0, 0, 0.3);
51       max-width: 800px;
52       width: 90%;
53     }
54     h1 {
55       color: #667eea;
56       text-align: center;
57       margin-bottom: 30px;
58       font-size: 2.5em;
59       text-shadow: 2px 2px 4px rgba(0,0,0,0.1);
60     }
61     .info-grid {
```

task1\_terraform\_init.png

```
Dell1@DESKTOP-3INCVEJH MINGW64 ~/Documents/CC_arreej_10/Assignment2 (main)
$ # Initialize Terraform
terraform init
Initializing the backend...
Initializing modules...
- backend_servers in modules\webserver
- networking in modules\networking
- nginx_server in modules\webserver
- security in modules\security
Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 5.0"...
- Finding latest version of hashicorp/http...
- Installing hashicorp/http v3.5.0...
- Installed hashicorp/http v3.5.0 (signed by HashiCorp)
- Installing hashicorp/aws v5.100.0...
- Installed hashicorp/aws v5.100.0 (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.
```

task1\_terraform\_apply.png

```

MINGW64:/c/Users/Dell/Documents/CC_areej_10/Assignment2
    "public_ip" = "35.175.111.154"
}
"web-3" = {
    "instance_id" = "i-09ade9131df87388b"
    "private_ip" = "10.0.10.180"
    "public_ip" = "44.204.96.43"
}
configuration_guide = <<EOT
=====
↳ DEPLOYMENT SUCCESSFUL!
=====

⌚ Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180

4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx

5. Test in browser:
   https://3.230.127.48

⌚ Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
- web-2: 35.175.111.154 (private: 10.0.10.86)
- web-3: 44.204.96.43 (private: 10.0.10.180)

=====
EOT
nginx_instance_id = "i-0e668eb55f575d3c4"
nginx_public_ip = "3.230.127.48"
subnet_id = "subnet-0d8278da93817bc6b"
vpc_id = "vpc-0df4a962ae2d7806a"

```

task1\_terraform\_output.png

```

xelli@DESKTOP-JNCVE3H MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform output
backend_servers_info = {
    "web-1" = {
        "instance_id" = "i-049a54d483fd72337"
        "private_ip" = "10.0.10.67"
        "public_ip" = "98.92.216.122"
    }
    "web-2" = {
        "instance_id" = "i-02234c4362bf54681"
        "private_ip" = "10.0.10.86"
        "public_ip" = "35.175.111.154"
    }
    "web-3" = {
        "instance_id" = "i-09ade9131df87388b"
        "private_ip" = "10.0.10.180"
        "public_ip" = "44.204.96.43"
    }
}
configuration_guide = <<EOT
=====
↳ DEPLOYMENT SUCCESSFUL!
=====

⌚ Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180

4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx

5. Test in browser:
   https://3.230.127.48

⌚ Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
- web-2: 35.175.111.154 (private: 10.0.10.86)

```

task1\_nginx\_browser.png



task1\_terraform\_destroy.png

```
Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform destroy
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 0s [id=https://icanhazip.com]
module.nginx_server.data.aws_amazon_linux: Reading...
module.backend_servers["web-1"].data.aws_amazon_linux: Reading...
module.backend_servers["web-2"].data.aws_amazon_linux: Reading...
module.backend_servers["web-3"].data.aws_amazon_linux: Reading...
module.backend_servers["web-2"].aws_key_pair.server_key: Refreshing state... [id=...
module.networking.aws_vpc.main: Refreshing state... [id=vpc-0df4a962ae2d7806a]
```

task2\_terraform\_apply.png

```
弇 MINGW64:/c/Users/Dell/Documents/CC_areej_10/Assignment2
    "public_ip" = "35.175.111.154"
}
"web-3" = {
    "instance_id" = "i-09ade9131df87388b"
    "private_ip" = "10.0.10.180"
    "public_ip" = "44.204.96.43"
}
configuration_guide = <<EOT
=====
→ DEPLOYMENT SUCCESSFUL!
=====

↳ Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48
2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf
3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180
4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx
5. Test in browser:
   https://3.230.127.48

Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
- web-2: 35.175.111.154 (private: 10.0.10.86)
- web-3: 44.204.96.43 (private: 10.0.10.180)

=====
EOT
nginx_instance_id = "i-0e668eb55f575d3c4"
nginx_public_ip = "3.230.127.48"
subnet_id = "subnet-0d8278da93817bc6b"
vpc_id = "vpc-0df4a962ae2d7806a"
```

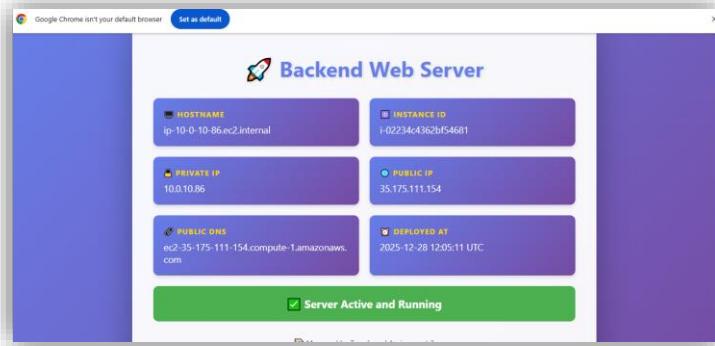
task2\_terraform\_output.png

```
bell@DESKTOP-JNCVE3H MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform output
backend_servers_info = {
  "web-1" = {
    "instance_id" = "i-049a54d483fd72337"
    "private_ip" = "10.0.10.67"
    "public_ip" = "98.92.216.122"
  }
  "web-2" = {
    "instance_id" = "i-02234c4362bf54681"
    "private_ip" = "10.0.10.86"
    "public_ip" = "35.175.111.154"
  }
  "web-3" = {
    "instance_id" = "i-09ade9131df87388b"
    "private_ip" = "10.0.10.180"
    "public_ip" = "44.204.96.43"
  }
}
configuration_guide = <>EOT
=====
⇒ DEPLOYMENT SUCCESSFUL!
=====

Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48
2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf
3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180
4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx
5. Test in browser:
   https://3.230.127.48

Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
  - web-2: 35.175.111.154 (private: 10.0.10.86)
```

task2\_nginx\_browser.png



Task3\_terraform\_apply.png

```

MINGW64:/c/Users/Dell/Documents/CC_areej_10/Assignment2
    "public_ip" = "35.175.111.154"
}
"web-3" = {
    "instance_id" = "i-09ade9131df87388b"
    "private_ip" = "10.0.10.180"
    "public_ip" = "44.204.96.43"
}
configuration_guide = <<EOT
=====
↳ DEPLOYMENT SUCCESSFUL!
=====

⌚ Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180

4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx

5. Test in browser:
   https://3.230.127.48

⌚ Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
- web-2: 35.175.111.154 (private: 10.0.10.86)
- web-3: 44.204.96.43 (private: 10.0.10.180)

=====
EOT
nginx_instance_id = "i-0e668eb55f575d3c4"
nginx_public_ip = "3.230.127.48"
subnet_id = "subnet-0d8278da93817bc6b"
vpc_id = "vpc-0df4a962ae2d7806a"

```

Task3\_terraform\_output.png

```

kelli@DESKTOP-JNCVE3H MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform output
backend_servers_info = {
    "web-1" = {
        "instance_id" = "i-049a54d483fd72337"
        "private_ip" = "10.0.10.67"
        "public_ip" = "98.92.216.122"
    }
    "web-2" = {
        "instance_id" = "i-02234c4362bf54681"
        "private_ip" = "10.0.10.86"
        "public_ip" = "35.175.111.154"
    }
    "web-3" = {
        "instance_id" = "i-09ade9131df87388b"
        "private_ip" = "10.0.10.180"
        "public_ip" = "44.204.96.43"
    }
}
configuration_guide = <<EOT
=====
↳ DEPLOYMENT SUCCESSFUL!
=====

⌚ Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180

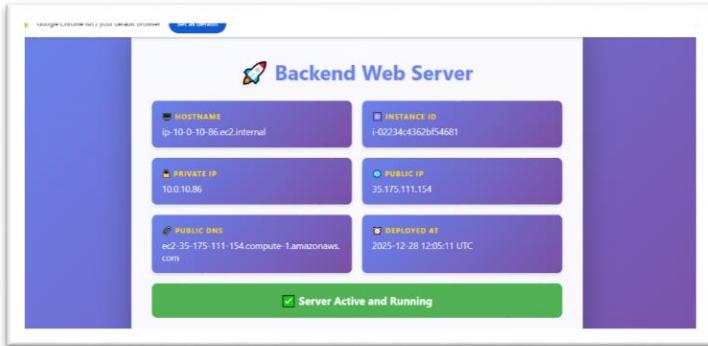
4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx

5. Test in browser:
   https://3.230.127.48

⌚ Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
- web-2: 35.175.111.154 (private: 10.0.10.86)

```

Task3\_nginx\_browser.png



Task3\_terraform\_destroy.png

```
Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform destroy
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 0s [id=https://icanhazip.com]
module.nginx_server.data.aws ami.amazon_linux: Reading...
module.backend_servers["web-1"].data.aws ami.amazon_linux: Reading...
module.backend_servers["web-2"].data.aws ami.amazon_linux: Reading...
module.backend_servers["web-3"].data.aws ami.amazon_linux: Reading...
module.backend_servers["web-2"].aws_key_pair.server_key: Refreshing state... [id=...
module.networking.aws_vpc.main: Refreshing state... [id=vpc-0df4a962ae2d7806a]
```

task3\_main\_tf\_restored.png

A screenshot of a code editor displaying the main.tf file of a Terraform configuration. The file contains the following code:

```
ver outputs.tf ..\webservice    $ apache-setup.sh    $ nginx-setup.sh    main.tf x ... 
 1 terraform {
 2   required_providers {
 3     aws = {
 4       source  = "hashicorp/aws"
 5       version = ">= 5.0"
 6     }
 7   }
 8   required_version = ">= 1.0"
 9 }
10 provider "aws" {
11   region = "me-central-1"
12 }
13
14 # Networking Module
15 module "networking" {
16   source = "./modules/networking"
17
18   vpc_cidr_block      = var.vpc_cidr_block
19   subnet_cidr_block  = var.subnet_cidr_block
20   availability_zone  = var.availability_zone
21   env_prefix          = var.env_prefix
22   common_tags         = local.common_tags
23 }
24
25
26 # Security Module
27 module "security" {
28   source = "./modules/security"
29
30   vpc_id      = module.networking.vpc_id
31   env_prefix  = var.env_prefix
32   my_ip       = local.my_ip
```

## TASK 4 - Subnet Module

task4\_module\_structure.png

```
$ ls -R
.:
README.md  docs/  locals.tf  main.tf  modules/  outputs.tf  screenshots/  scripts/  terraform.tfvars  variables.tf
./docs:
architecture.md  troubleshooting.md

./modules:
networking/  security/  webserver/

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

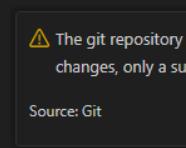
./modules/security:
main.tf  outputs.tf  variables.tf

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

./screenshots:
bonus/  part1/  part2/  part3/  part4/  part5/  part6/
./screenshots/bonus:
./screenshots/part1:
./screenshots/part2:
./screenshots/part3:
./screenshots/part4:
./screenshots/part5:
./screenshots/part6:
./scripts:
apache-setup.sh  nginx-setup.sh
```

task4\_subnet\_variables.png

```
variables.tf
1  # VPC Configuration
2  variable "vpc_cidr_block" {
3    description = "CIDR block for VPC"
4    type        = string
5    default     = "10.0.0.0/16"
6
7    validation {
8      condition    = can(cidrhost(var.vpc_cidr_block, 0))
9      error_message = "Must be a valid IPv4 CIDR block."
10   }
11 }
12
13 variable "subnet_cidr_block" {
14   description = "CIDR block for subnet"
15   type        = string
16   default     = "10.0.10.0/24"
17
18   validation {
19     condition    = can(cidrhost(var.subnet_cidr_block, 0))
20     error_message = "Must be a valid IPv4 CIDR block."
21   }
22 }
23
24 variable "availability_zone" {
25   description = "AWS availability zone"
26   type        = string
27   default     = "me-central-1a"
28 }
29
30 # Environment Configuration
31 variable "env_prefix" {
32   description = "Environment prefix for resource naming"
```



Source: Git

task4\_subnet\_main.png

The screenshot shows a code editor interface with a dark theme. In the top navigation bar, there are tabs for 'File', 'Edit', 'Selection', 'View', 'Go', 'Run', and '...'. Below the tabs, there's a search bar with the placeholder 'Q Assignment2'. The left sidebar, titled 'EXPLORER', lists several files and folders under 'ASSIGNMENT2': 'docs', 'modules', 'networking' (which is expanded), 'main.tf', 'outputs.tf', 'variables.tf', 'security', 'webserver', 'screenshots', 'scripts', '.gitignore', 'locals.tf', 'main.tf', 'outputs.tf', 'README.md', 'terraform.tfvars', and 'variables.tf'. The 'main.tf' file in the 'networking' folder is currently selected and highlighted in blue. The main editor area displays the contents of the 'main.tf' file:

```
modules > networking > main.tf
31 resource "aws_internet_gateway" "main" {
32   tags = merge(
33     var.common_tags,
34   )
35 }
36 # Create Route Table
37 resource "aws_route_table" "main" {
38   vpc_id = aws_vpc.main.id
39
40   route {
41     cidr_block = "0.0.0.0/0"
42     gateway_id = aws_internet_gateway.main.id
43   }
44
45   tags = merge(
46     var.common_tags,
47   {
48     Name = "${var.env_prefix}-rt"
49   }
50 }
51
52 # Associate Route Table with Subnet
53 resource "aws_route_table_association" "main" {
54   subnet_id    = aws_subnet.main.id
55   route_table_id = aws_route_table.main.id
56 }
```

task4\_subnet\_outputs.png

The screenshot shows a code editor interface with a dark theme. In the top navigation bar, there are tabs for 'File', 'Edit', 'Selection', 'View', 'Go', 'Run', and '...'. Below the tabs, there's a search bar with the placeholder 'Q Assignment2'. The left sidebar, titled 'EXPLORER', lists several files and folders under 'ASSIGNMENT2': 'docs', 'modules', 'networking' (which is expanded), 'main.tf', 'outputs.tf', 'variables.tf', 'security', 'webserver', 'screenshots', 'scripts', '.gitignore', 'locals.tf', 'main.tf', 'outputs.tf', 'README.md', 'terraform.tfvars', and 'variables.tf'. The 'outputs.tf' file in the 'networking' folder is currently selected and highlighted in blue. The main editor area displays the contents of the 'outputs.tf' file:

```
modules > networking > outputs.tf
1 output "vpc_id" {
2   description = "ID of the VPC"
3   value       = aws_vpc.main.id
4 }
5
6 output "subnet_id" {
7   description = "ID of the subnet"
8   value       = aws_subnet.main.id
9 }
10
11 output "igw_id" {
12   description = "ID of the Internet Gateway"
13   value       = aws_internet_gateway.main.id
14 }
15
16 output "route_table_id" [
17   description = "ID of the route table"
18   value       = aws_route_table.main.id
19 ]
```

task4\_main\_tf\_with\_module.png

The screenshot shows a code editor with four tabs open. The active tab is 'main.tf' which contains the following Terraform code:

```
ver outputs.tf ...\\webserver $ apache-setup.sh $ nginx-setup.sh main.tf X ...  
main.tf  
1  terraform {  
2    required_providers {  
3      aws = {  
4        source  = "hashicorp/aws"  
5        version = "~> 5.0"  
6      }  
7    }  
8    required_version = ">= 1.0"  
9  }  
10  
11 provider "aws" {  
12   region = "me-central-1"  
13 }  
14  
15 # Networking Module  
16 module "networking" {  
17   source = "./modules/networking"  
18  
19   vpc_cidr_block    = var.vpc_cidr_block  
20   subnet_cidr_block = var.subnet_cidr_block  
21   availability_zone = var.availability_zone  
22   env_prefix        = var.env_prefix  
23   common_tags       = local.common_tags  
24 }  
25  
26 # Security Module  
27 module "security" {  
28   source = "./modules/security"  
29  
30   vpc_id      = module.networking.vpc_id  
31   env_prefix  = var.env_prefix  
32   my_ip       = local.my_ip
```

Task4\_terraform\_init.png

The screenshot shows a terminal window with the following output:

```
DELL@DESKTOP-JNCVEJH MINGW64 ~\Documents\CC_areej_10\Assignment2 (main)  
$ # Initialize Terraform  
terraform init  
Initializing the backend...  
Initializing modules...  
- backend_servers in modules\webserver  
- networking in modules\networking  
- nginx_server in modules\webserver  
- security in modules\security  
Initializing provider plugins...  
- Finding hashicorp/aws versions matching "~> 5.0"...  
- Finding latest version of hashicorp/http...  
- Installing hashicorp/http v3.5.0...  
- Installed hashicorp/http v3.5.0 (signed by HashiCorp)  
- Installing hashicorp/aws v5.100.0...  
- Installed hashicorp/aws v5.100.0 (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.
```

Task4\_terraform\_apply.png

```

MINGW64:/c/Users/Dell/Documents/CC_areej_10/Assignment2
    "public_ip" = "35.175.111.154"
}
"web-3" = {
    "instance_id" = "i-09ade9131df87388b"
    "private_ip" = "10.0.10.180"
    "public_ip" = "44.204.96.43"
}
configuration_guide = <<EOT
=====
↳ DEPLOYMENT SUCCESSFUL!
=====

⌚ Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180

4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx

5. Test in browser:
   https://3.230.127.48

⌚ Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
- web-2: 35.175.111.154 (private: 10.0.10.86)
- web-3: 44.204.96.43 (private: 10.0.10.180)

=====
EOT
nginx_instance_id = "i-0e668eb55f575d3c4"
nginx_public_ip = "3.230.127.48"
subnet_id = "subnet-0d8278da93817bc6b"
vpc_id = "vpc-0df4a962ae2d7806a"

```

Task4\_terraform\_output.png

```

kelli@DESKTOP-JNCVE3H MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform output
backend_servers_info = {
    "web-1" = {
        "instance_id" = "i-049a54d483fd72337"
        "private_ip" = "10.0.10.67"
        "public_ip" = "98.92.216.122"
    }
    "web-2" = {
        "instance_id" = "i-02234c4362bf54681"
        "private_ip" = "10.0.10.86"
        "public_ip" = "35.175.111.154"
    }
    "web-3" = {
        "instance_id" = "i-09ade9131df87388b"
        "private_ip" = "10.0.10.180"
        "public_ip" = "44.204.96.43"
    }
}
configuration_guide = <<EOT
=====
↳ DEPLOYMENT SUCCESSFUL!
=====

⌚ Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180

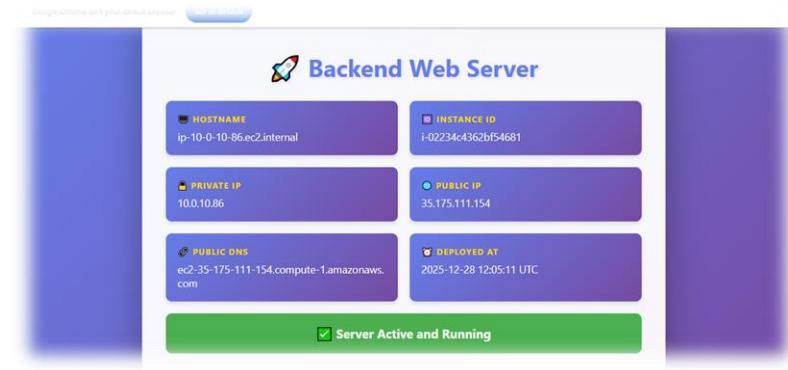
4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx

5. Test in browser:
   https://3.230.127.48

⌚ Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
- web-2: 35.175.111.154 (private: 10.0.10.86)

```

Task4\_nginx\_browser.png



Task4\_terraform\_destroy.png

```
Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform destroy
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 0s [id=https://icanhazip.com]
module.nginx_server.data.aws_amis.amazon_linux: Reading...
module.backend_servers["web-1"].data.aws_amis.amazon_linux: Reading...
module.backend_servers["web-2"].data.aws_amis.amazon_linux: Reading...
module.backend_servers["web-3"].data.aws_amis.amazon_linux: Reading...
module.backend_servers["web-2"].aws_key_pair.server_key: Refreshing state... [id=...
module.networking.aws_vpc.main: Refreshing state... [id=vpc-0df4a962ae2d7806a]
```

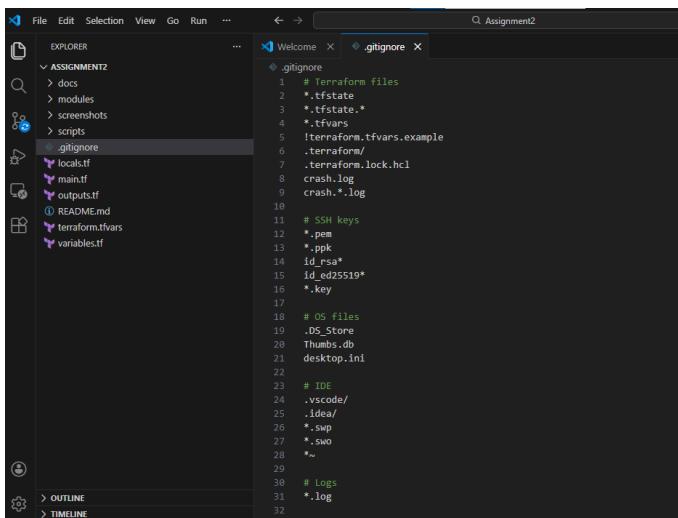
## TASK 5 - Webserver Module

task5\_webserver\_module\_structure.png

```
Dell@DESKTOP-JNCVEJH MINGW64 ~ (main)
$ # Go to your desired location (e.g., Documents)
cd ~/Documents

# Create project folder
mkdir -p CC_areej_10/Assignment2
cd CC_areej_10/Assignment2

Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$
```



task5\_webserver\_variables.png

```
# VPC Configuration
variable "vpc_cidr_block" {
  description = "CIDR block for VPC"
  type        = string
  default     = "10.0.0.0/16"

  validation {
    condition  = can(cidrhost(var.vpc_cidr_block, 0))
    error_message = "Must be a valid IPv4 CIDR block."
  }
}

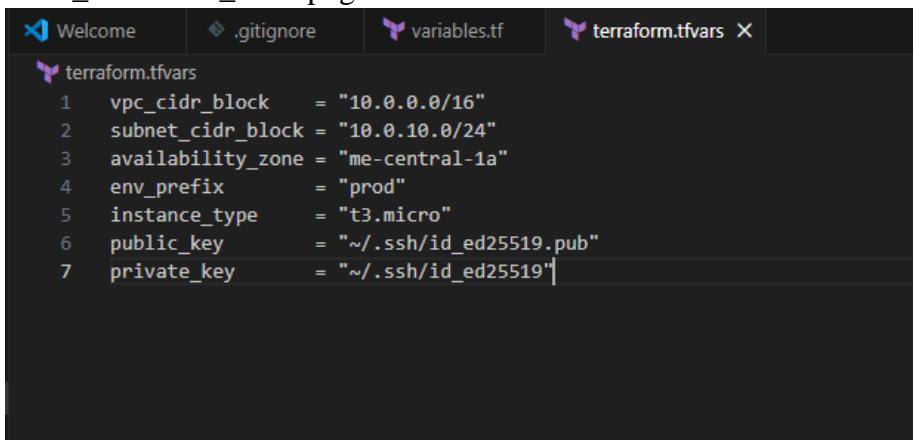
variable "subnet_cidr_block" {
  description = "CIDR block for subnet"
  type        = string
  default     = "10.0.10.0/24"

  validation {
    condition  = can(cidrhost(var.subnet_cidr_block, 0))
    error_message = "Must be a valid IPv4 CIDR block."
  }
}

variable "availability_zone" {
  description = "AWS availability zone"
  type        = string
  default     = "me-central-1a"
}

# Environment Configuration
variable "env_prefix" {
  description = "Environment prefix for resource naming"
}
```

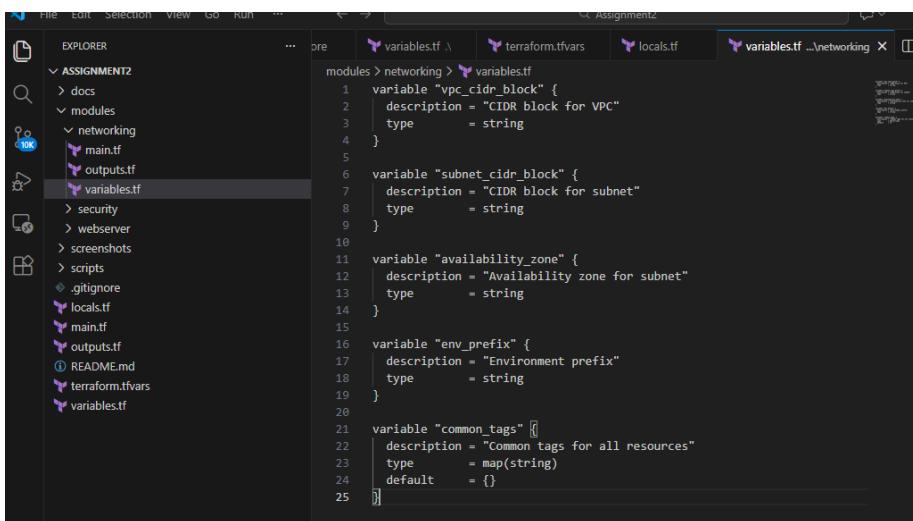
task5\_webserver\_main.png



A screenshot of a code editor showing a Terraform variables file named `variables.tf`. The file contains the following configuration:

```
vpc_cidr_block  = "10.0.0.0/16"
subnet_cidr_block = "10.0.10.0/24"
availability_zone = "me-central-1a"
env_prefix        = "prod"
instance_type     = "t3.micro"
public_key        = "./ssh/id_ed25519.pub"
private_key        = "./ssh/id_ed25519"
```

task1\_variables\_tf.png



A screenshot of a code editor showing a Terraform variables file named `variables.tf` located within a `networking` module. The file contains the following configuration:

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

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

variable "availability_zone" {
  description = "Availability zone for subnet"
  type        = string
}

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

variable "common_tags" [
  description = "Common tags for all resources"
  type        = map(string)
  default     = {}
]
```

task5\_outputs\_updated.png

```

modules > networking > 🎨 outputs.tf
  1  output "vpc_id" {
  2    description = "ID of the VPC"
  3    value        = aws_vpc.main.id
  4  }
  5
  6  output "subnet_id" {
  7    description = "ID of the subnet"
  8    value        = aws_subnet.main.id
  9  }
 10
 11 output "igw_id" {
 12   description = "ID of the Internet Gateway"
 13   value        = aws_internet_gateway.main.id
 14 }
 15
 16 output "route_table_id" []
 17   description = "ID of the route table"
 18   value        = aws_route_table.main.id
 19 }
```

assignment\_part4\_terraform\_init.png

```

Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ # Initialize Terraform
terraform init
Initializing the backend...
Initializing modules...
- backend_servers in modules\webserver
- networking in modules\networking
- nginx_server in modules\webserver
- security in modules\security
Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 5.0"...
- Finding latest version of hashicorp/http...
- Installing hashicorp/http v3.5.0...
- Installed hashicorp/http v3.5.0 (signed by HashiCorp)
- Installing hashicorp/aws v5.100.0...
- Installed hashicorp/aws v5.100.0 (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.
```

Task5\_terraform\_apply.png

```

MINGW64:/c/Users/Dell/Documents/CC_areej_10/Assignment2
    "public_ip" = "35.175.111.154"
}
"web-3" = {
    "instance_id" = "i-09ade9131df87388b"
    "private_ip" = "10.0.10.180"
    "public_ip" = "44.204.96.43"
}
configuration_guide = <<EOT
=====
↳ DEPLOYMENT SUCCESSFUL!
=====

⌚ Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180

4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx

5. Test in browser:
   https://3.230.127.48

⌚ Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
- web-2: 35.175.111.154 (private: 10.0.10.86)
- web-3: 44.204.96.43 (private: 10.0.10.180)

=====
EOT
nginx_instance_id = "i-0e668eb55f575d3c4"
nginx_public_ip = "3.230.127.48"
subnet_id = "subnet-0d8278da93817bc6b"
vpc_id = "vpc-0df4a962ae2d7806a"

```

Task5\_terraform\_output.png

```

kelli@DESKTOP-JNCVE3H MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform output
backend_servers_info = {
    "web-1" = {
        "instance_id" = "i-049a54d483fd72337"
        "private_ip" = "10.0.10.67"
        "public_ip" = "98.92.216.122"
    }
    "web-2" = {
        "instance_id" = "i-02234c4362bf54681"
        "private_ip" = "10.0.10.86"
        "public_ip" = "35.175.111.154"
    }
    "web-3" = {
        "instance_id" = "i-09ade9131df87388b"
        "private_ip" = "10.0.10.180"
        "public_ip" = "44.204.96.43"
    }
}
configuration_guide = <<EOT
=====
↳ DEPLOYMENT SUCCESSFUL!
=====

⌚ Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180

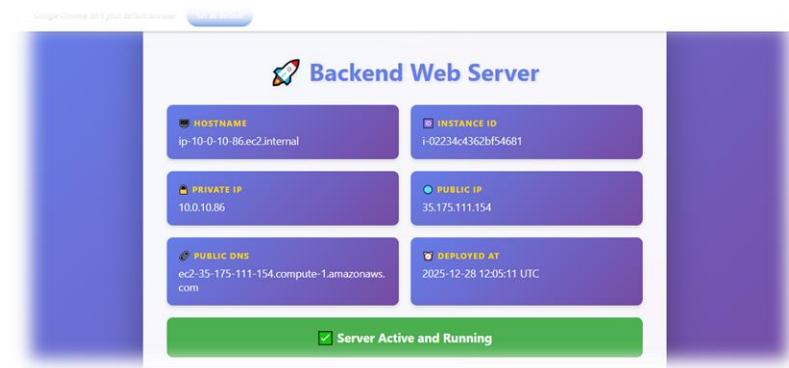
4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx

5. Test in browser:
   https://3.230.127.48

⌚ Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
- web-2: 35.175.111.154 (private: 10.0.10.86)

```

Task5\_nginx\_browser.png



Task5\_terraform\_destroy.png

```
Dell1@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform destroy
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 0s [id=https://icanhazip.com]
module.nginx_server.data.aws_ami.amazon_linux: Reading...
module.backend_servers["web-1"].data.aws_ami.amazon_linux: Reading...
module.backend_servers["web-2"].data.aws_ami.amazon_linux: Reading...
module.backend_servers["web-3"].data.aws_ami.amazon_linux: Reading...
module.backend_servers["web-2"].aws_key_pair.server_key: Refreshing state... [id=...
module.networking.aws_vpc.main: Refreshing state... [id=vpc-0df4a962ae2d7806a]
```

task6\_entry\_script\_https.png

```
scripts > $ nginx-setup.sh
1  #!/bin/bash
2  set -e
3
4  # Update and install Nginx
5  yum update -y
6  yum install -y nginx openssl
7  systemctl start nginx
8  systemctl enable nginx
9
10 # Create SSL directories
11 mkdir -p /etc/ssl/private
12 mkdir -p /etc/ssl/certs
13
14 # Get metadata token
15 TOKEN=$(curl -s -X PUT "http://169.254.169.254/latest/api/token" \
16   -H "X-aws-ec2-metadata-token-ttl-seconds: 21600")
17
18 # Get public IP
19 PUBLIC_IP=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" \
20   http://169.254.169.254/latest/meta-data/public-ipv4)
21
22 # Generate self-signed certificate
23 openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
24   -keyout /etc/ssl/private/selfsigned.key \
25   -out /etc/ssl/certs/selfsigned.crt \
26   -subj "/C=US/ST=State/L=City/O=Organization/CN=$PUBLIC_IP" \
27   -addext "subjectAltName=IP:$PUBLIC_IP" \
28   -addext "basicConstraints=CA:FALSE" \
29   -addext "keyUsage=digitalSignature,keyEncipherment" \
30   -addext "extendedKeyUsage=serverAuth"
31
32 echo "Self-signed certificate created for IP: $PUBLIC_IP"
--
```

task6\_terraform\_apply.png

```
MINGW64:/c/Users/Dell/Documents/CC_areej_10/Assignment2
    "public_ip" = "35.175.111.154"
}
"web-3" = {
    "instance_id" = "i-09ade9131df87388b"
    "private_ip" = "10.0.10.180"
    "public_ip" = "44.204.96.43"
}
configuration_guide = <<EOT
=====
» DEPLOYMENT SUCCESSFUL!
=====

» Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48
2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf
3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180
4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx
5. Test in browser:
   https://3.230.127.48
» Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
  - web-2: 35.175.111.154 (private: 10.0.10.86)
  - web-3: 44.204.96.43 (private: 10.0.10.180)
=====
EOT
nginx_instance_id = "i-0e668eb55f575d3c4"
nginx_public_ip = "3.230.127.48"
subnet_id = "subnet-0d8278da93817bc6b"
vpc_id = "vpc-0df4a962ae2d7806a"
```

task6\_terraform\_output.png

```

dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform output
backend_servers_info = {
  "web-1" = {
    "instance_id" = "i-049a54d483fd72337"
    "private_ip" = "10.0.10.67"
    "public_ip" = "98.92.216.122"
  }
  "web-2" = {
    "instance_id" = "i-02234c4362bf54681"
    "private_ip" = "10.0.10.86"
    "public_ip" = "35.175.111.154"
  }
  "web-3" = {
    "instance_id" = "i-09ade9131df87388b"
    "private_ip" = "10.0.10.180"
    "public_ip" = "44.204.96.43"
  }
}
configuration_guide = <<EOT
=====
→ DEPLOYMENT SUCCESSFUL!
=====

💡 Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48
2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf
3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180
4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx
5. Test in browser:
   https://3.230.127.48

💡 Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
- web-2: 35.175.111.154 (private: 10.0.10.86)

```

task6\_nginx\_https\_browser.png



task6\_http\_redirect.png

```
Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ curl -I http://3.230.127.48
HTTP/1.1 301 Moved Permanently
Server: nginx/1.28.0
Date: Sun, 28 Dec 2025 15:17:18 GMT
Content-Type: text/html
Content-Length: 169
Connection: keep-alive
Location: https://3.230.127.48/
```

## TASK 7 - Reverse Proxy

task7\_apache\_script.png

```
scripts > $ apache-setup.sh
30  cat > /var/www/html/index.html <<EOF
31  <!DOCTYPE html>
32  <html>
33  <head>
34  <title>Backend Web Server - $HOSTNAME</title>
35  <style>
36  body {
37      font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
38      margin: 0;
39      padding: 0;
40      background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
41      min-height: 100vh;
42      display: flex;
43      justify-content: center;
44      align-items: center;
45  }
46  .container {
47      background: rgba(255, 255, 255, 0.95);
48      padding: 40px;
49      border-radius: 20px;
50      box-shadow: 0 20px 60px rgba(0, 0, 0, 0.3);
51      max-width: 800px;
52      width: 90%;
53  }
54  h1 {
55      color: #667eea;
56      text-align: center;
57      margin-bottom: 30px;
58      font-size: 2.5em;
59      text-shadow: 2px 2px 4px rgba(0,0,0,0.1);
60  }
61  .info-grid {
```

task7\_main\_tf\_web1.png

```
ver  outputs.tf ...\\webserver  $ apache-setup.sh  $ nginx-setup.sh  main.tf  X  ⌂  ...
└── main.tf
    1  terraform {
    2      required_providers {
    3          aws = {
    4              source  = "hashicorp/aws"
    5              version = "~> 5.0"
    6          }
    7      }
    8      required_version = ">= 1.0"
    9  }
   10
   11 provider "aws" {
   12     region = "me-central-1"
   13 }
   14
   15 # Networking Module
   16 module "networking" {
   17     source = "./modules/networking"
   18
   19     vpc_cidr_block    = var.vpc_cidr_block
   20     subnet_cidr_block = var.subnet_cidr_block
   21     availability_zone = var.availability_zone
   22     env_prefix        = var.env_prefix
   23     common_tags       = local.common_tags
   24 }
   25
   26 # Security Module
   27 module "security" {
   28     source = "./modules/security"
   29
   30     vpc_id      = module.networking.vpc_id
   31     env_prefix  = var.env_prefix
   32     my_ip       = local.my_ip
```

task7\_outputs\_web1.png

```
dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform output
backend_servers_info = {
  "web-1" = {
    "instance_id" = "i-049a54d483fd72337"
    "private_ip" = "10.0.10.67"
    "public_ip" = "98.92.216.122"
  }
  "web-2" = {
    "instance_id" = "i-02234c4362bf54681"
    "private_ip" = "10.0.10.86"
    "public_ip" = "35.175.111.154"
  }
  "web-3" = {
    "instance_id" = "i-09ade9131df87388b"
    "private_ip" = "10.0.10.180"
    "public_ip" = "44.204.96.43"
  }
}
configuration_guide = <<EOT
=====
❯ DEPLOYMENT SUCCESSFUL!
=====

❯ Next Steps:
1. SSH into Nginx server:
  ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
  sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
  Replace:
  - BACKEND_IP_1 with: 10.0.10.67
  - BACKEND_IP_2 with: 10.0.10.86
  - BACKEND_IP_3 with: 10.0.10.180

4. Test and restart Nginx:
  sudo nginx -t
  sudo systemctl restart nginx

5. Test in browser:
  https://3.230.127.48

❯ Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
- web-2: 35.175.111.154 (private: 10.0.10.86)
```

task7\_terraform\_apply.png

```

MINGW64:/c/Users/Dell/Documents/CC_areej_10/Assignment2
    "public_ip" = "35.175.111.154"
}
"web-3" = {
    "instance_id" = "i-09ade9131df87388b"
    "private_ip" = "10.0.10.180"
    "public_ip" = "44.204.96.43"
}
configuration_guide = <<EOT
=====
↳ DEPLOYMENT SUCCESSFUL!
=====

⌚ Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180

4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx

5. Test in browser:
   https://3.230.127.48

⌚ Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
  - web-2: 35.175.111.154 (private: 10.0.10.86)
  - web-3: 44.204.96.43 (private: 10.0.10.180)

=====
EOT
nginx_instance_id = "i-0e668eb55f575d3c4"
nginx_public_ip = "3.230.127.48"
subnet_id = "subnet-0d8278da93817bc6b"
vpc_id = "vpc-0df4a962ae2d7806a"

```

task7\_terraform\_output.png

```

bell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform output
backend_servers_info = {
    "web-1" = {
        "instance_id" = "i-049a54d483fd72337"
        "private_ip" = "10.0.10.67"
        "public_ip" = "98.92.216.122"
    }
    "web-2" = {
        "instance_id" = "i-02234c4362bf54681"
        "private_ip" = "10.0.10.86"
        "public_ip" = "35.175.111.154"
    }
    "web-3" = {
        "instance_id" = "i-09ade9131df87388b"
        "private_ip" = "10.0.10.180"
        "public_ip" = "44.204.96.43"
    }
}
configuration_guide = <<EOT
=====
↳ DEPLOYMENT SUCCESSFUL!
=====

⌚ Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180

4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx

5. Test in browser:
   https://3.230.127.48

⌚ Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
  - web-2: 35.175.111.154 (private: 10.0.10.86)

```

### task7\_ssh\_webserver.png

```
Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform output nginx_public_ip
"3.230.127.48"

Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48
The authenticity of host '3.230.127.48 (3.230.127.48)' can't be established.
ED25519 key fingerprint is SHA256:DOrAdRFQ6TRCoq/DNp520s0Y5IR36cTo4dHCpqGE6Yg.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '3.230.127.48' (ED25519) to the list of known hosts.

#_
~\_\ #####
~~ \#####
~~ \|#
~~ \|# /__ Amazon Linux 2023 (ECS Optimized)
~~ \|# V~ ' .->
~~ \|_/
~~ \|_/
~/m/|_/

For documentation, visit http://aws.amazon.com/documentation/ecs
[ec2-user@ip-10-0-10-158 ~]$
```

### task7\_nginx\_conf\_reverse\_proxy.png

```
sudo nginx -t
nginx: [warn] the "listen ... http2" directive is deprecated, use the "http2" directive instead in /etc/nginx/nginx.conf:52
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
[ec2-user@ip-10-0-10-158 ~]$
```

### task7\_nginx\_restart.png

```
nginx: configuration file /etc/nginx/nginx.conf test is successful
[ec2-user@ip-10-0-10-158 ~]$ sudo systemctl restart nginx
[ec2-user@ip-10-0-10-158 ~]$ systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
   Active: active (running) since Sun 2025-12-28 13:18:41 UTC; 13s ago
     Process: 68828 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
    Process: 68829 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
    Process: 68827 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 68828 (nginx)
      Tasks: 5 (limit: 1065)
     Memory: 4.4M
        CPU: 0.000 CPU(s)
       CGroup: /system.slice/nginx.service
               ├─68828 "nginx: master process /usr/sbin/nginx"
               ├─68829 "nginx: worker process"
               ├─68830 "nginx: worker process"
               ├─68831 "nginx: cache manager process"
               ├─68832 "nginx: cache loader process"

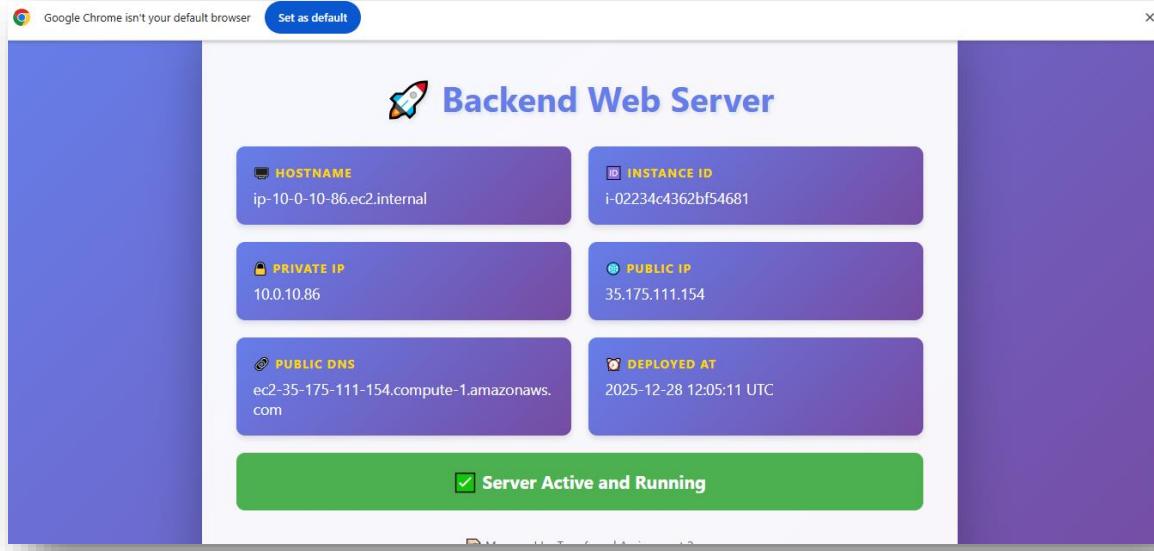
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal nginx[68828]: nginx: [warn] the "listen ... http2" directive is deprecated, use the "http2" directive instead in /etc/nginx/nginx.conf:52
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal nginx[68828]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal nginx[68827]: nginx: [warn] the "listen ... http2" directive is deprecated, use the "http2" directive instead in /etc/nginx/nginx.conf:52
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[ec2-user@ip-10-0-10-158 ~]$
```

### task7\_ssl\_cert.png

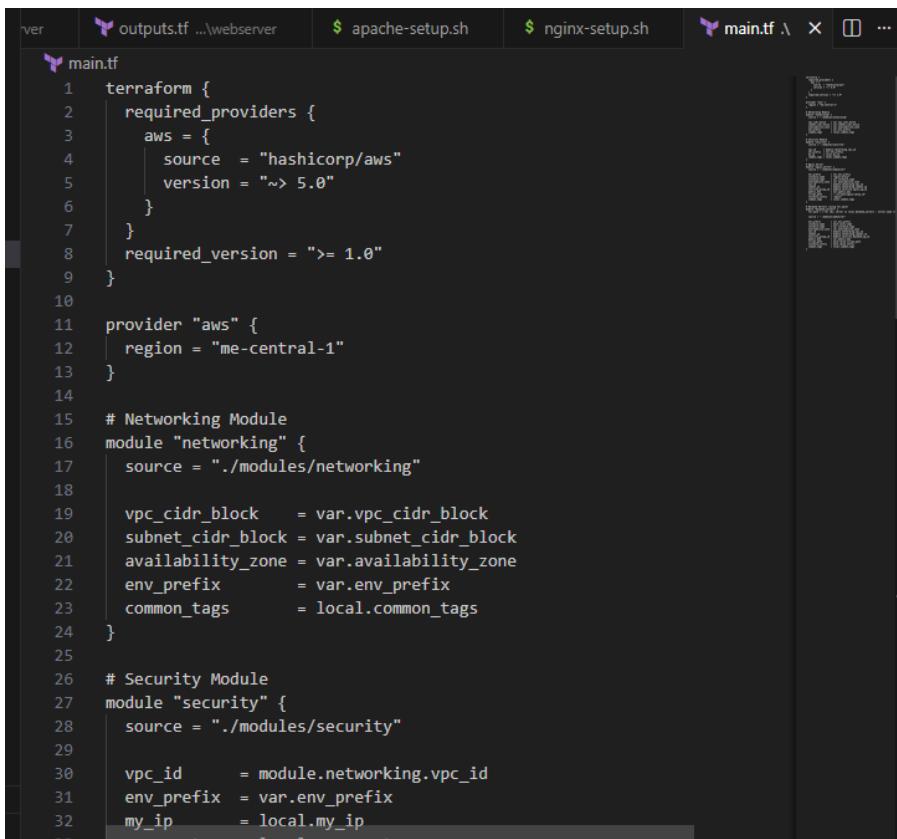
```

For documentation, visit http://aws.amazon.com/documentation/ecs
Last login: Sun Dec 28 14:56:47 2025 from 203.215.174.31
[ec2-user@ip-10-0-10-158 ~]$ sudo openssl x509 -in /etc/ssl/certs/selfsigned.crt -text -noout
Certificate:
Data:
    Version: 3 (0x2)
    Serial Number:
        46:d2:69:60:96:0b:85:aa:f1:9f:fa:81:bc:f9:6e:a3:79:30:1c:17
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: C=US, ST=State, L=City, O=Organization, CN=3.230.127.48
    Validity
        Not Before: Dec 28 12:05:22 2025 GMT
        Not After : Dec 28 12:05:22 2026 GMT
    Subject: C=US, ST=State, L=City, O=Organization, CN=3.230.127.48
    Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
        Public-Key: (2048 bit)
            Modulus:
                00:ca:06:aa:d5:a8:08:2c:1d:7c:97:e8:ba:7a:e4:
                16:f8:29:f3:6e:d1:36:69:c8:33:18:f0:3d:ad:82:
                6c:19:76:bf:81:07:23:7d:9f:95:5a:2b:32:2d:a3:
                5e:26:38:ae:bf:e7:fd:e7:f5:f7:74:e6:55:0f:9d:
                f4:11:8a:72:5b:04:c6:6d:83:50:69:f7:0c:de:20:
                ad:cb:9b:25:f8:fb:d0:4b:47:a7:d3:db:34:2e:6e:
                96:07:86:86:bd:77:91:06:ea:5a:58:d6:71:25:57:
                ae:f7:33:a1:48:e2:58:83:f7:a9:60:77:97:15:2f:
                4a:15:97:bd:71:a5:85:9b:59:2d:f8:f2:3b:a9:50:
                fe:2e:7f:2c:7b:19:40:di:1e:4a:90:72:14:a2:
                51:50:90:f0:ab:15:b2:a2:ef:31:aa:2b:24:d5:f8:
                3a:87:03:10:19:04:e3:67:66:7e:9a:07:70:d4:a3:
                2c:fd:d5:ce:d7:51:0a:6e:8e:a0:a1:62:a3:7d:2c:
                7c:c3:12:e0:8b:65:ds:11:a4:46:b2:32:el:97:ac:
                e5:ae:cd:0e:e2:bc:e0:1a:92:29:ab:3e:1f:f1:46:
                45:3b:61:61:be:79:8b:68:48:5b:fc:fd:80:ab:78:
                92:85:58:c5:35:9b:fa:35:91:e7:b:88:e5:dc:74:
                c1:df
            Exponent: 65537 (0x10001)
X509v3 extensions:
    X509v3 Subject Key Identifier:
        28:D9:40:D9:86:08:D3:35:69:25:31:FB:C9:E8:07:3D:74:A9:F6:56
    X509v3 Authority Key Identifier:
        28:D9:40:D9:86:08:D3:35:69:25:31:FB:C9:E8:07:3D:74:A9:F6:56
    X509v3 Subject Alternative Name:
        IP Address:3.230.127.48
    X509v3 Basic Constraints:
```

task7\_reverse\_proxy\_browser.png



Task8\_main\_tf\_web2.png



```
ver outputs.tf ...\\webserver $ apache-setup.sh $ nginx-setup.sh main.tf X ...  
main.tf  
1 terraform {  
2   required_providers {  
3     aws = {  
4       source  = "hashicorp/aws"  
5       version = "~> 5.0"  
6     }  
7   }  
8   required_version = ">= 1.0"  
9 }  
10  
11 provider "aws" {  
12   region = "me-central-1"  
13 }  
14  
15 # Networking Module  
16 module "networking" {  
17   source = "./modules/networking"  
18  
19   vpc_cidr_block    = var.vpc_cidr_block  
20   subnet_cidr_block = var.subnet_cidr_block  
21   availability_zone = var.availability_zone  
22   env_prefix        = var.env_prefix  
23   common_tags       = local.common_tags  
24 }  
25  
26 # Security Module  
27 module "security" {  
28   source = "./modules/security"  
29  
30   vpc_id      = module.networking.vpc_id  
31   env_prefix  = var.env_prefix  
32   my_ip       = local.my_ip
```

Task\_outputs\_we21.png

```
dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform output
backend_servers_info = {
  "web-1" = {
    "instance_id" = "i-049a54d483fd72337"
    "private_ip" = "10.0.10.67"
    "public_ip" = "98.92.216.122"
  }
  "web-2" = {
    "instance_id" = "i-02234c4362bf54681"
    "private_ip" = "10.0.10.86"
    "public_ip" = "35.175.111.154"
  }
  "web-3" = {
    "instance_id" = "i-09ade9131df87388b"
    "private_ip" = "10.0.10.180"
    "public_ip" = "44.204.96.43"
  }
}
configuration_guide = <<EOT
=====
❯ DEPLOYMENT SUCCESSFUL!
=====

💡 Next Steps:
1. SSH into Nginx server:
  ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
  sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
  Replace:
  - BACKEND_IP_1 with: 10.0.10.67
  - BACKEND_IP_2 with: 10.0.10.86
  - BACKEND_IP_3 with: 10.0.10.180

4. Test and restart Nginx:
  sudo nginx -t
  sudo systemctl restart nginx

5. Test in browser:
  https://3.230.127.48

💡 Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
- web-2: 35.175.111.154 (private: 10.0.10.86)
```

Task8\_terraform\_apply.png

```

MINGW64:/c/Users/Dell/Documents/CC_areej_10/Assignment2
    "public_ip" = "35.175.111.154"
}
"web-3" = {
    "instance_id" = "i-09ade9131df87388b"
    "private_ip" = "10.0.10.180"
    "public_ip" = "44.204.96.43"
}
configuration_guide = <<EOT
=====
↳ DEPLOYMENT SUCCESSFUL!
=====

⌚ Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180

4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx

5. Test in browser:
   https://3.230.127.48

⌚ Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
  - web-2: 35.175.111.154 (private: 10.0.10.86)
  - web-3: 44.204.96.43 (private: 10.0.10.180)

=====
EOT
nginx_instance_id = "i-0e668eb55f575d3c4"
nginx_public_ip = "3.230.127.48"
subnet_id = "subnet-0d8278da93817bc6b"
vpc_id = "vpc-0df4a962ae2d7806a"

```

Task8\_terraform\_output.png

```

bell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform output
backend_servers_info = {
    "web-1" = {
        "instance_id" = "i-049a54d483fd72337"
        "private_ip" = "10.0.10.67"
        "public_ip" = "98.92.216.122"
    }
    "web-2" = {
        "instance_id" = "i-02234c4362bf54681"
        "private_ip" = "10.0.10.86"
        "public_ip" = "35.175.111.154"
    }
    "web-3" = {
        "instance_id" = "i-09ade9131df87388b"
        "private_ip" = "10.0.10.180"
        "public_ip" = "44.204.96.43"
    }
}
configuration_guide = <<EOT
=====
↳ DEPLOYMENT SUCCESSFUL!
=====

⌚ Next Steps:
1. SSH into Nginx server:
   ssh -i ~/.ssh/id_ed25519 ec2-user@3.230.127.48

2. Edit Nginx config:
   sudo vi /etc/nginx/nginx.conf

3. Update backend IPs in upstream block:
   Replace:
   - BACKEND_IP_1 with: 10.0.10.67
   - BACKEND_IP_2 with: 10.0.10.86
   - BACKEND_IP_3 with: 10.0.10.180

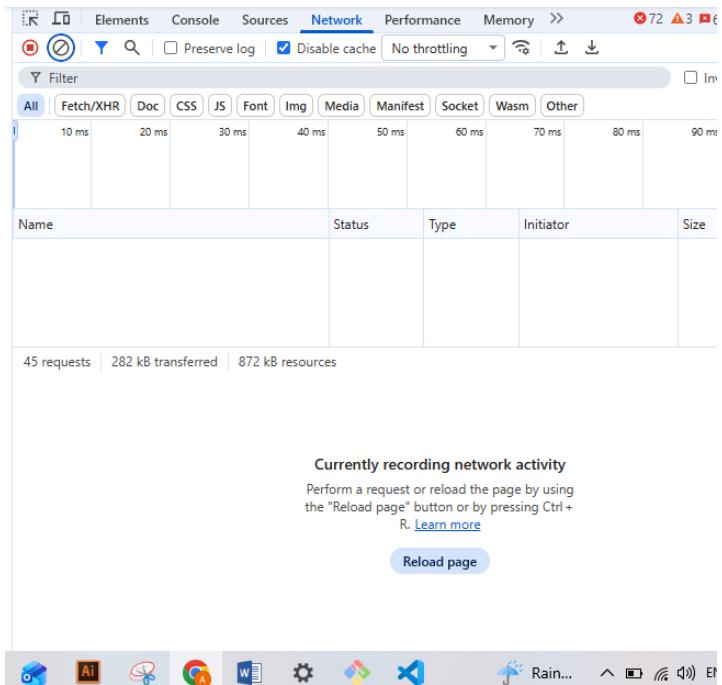
4. Test and restart Nginx:
   sudo nginx -t
   sudo systemctl restart nginx

5. Test in browser:
   https://3.230.127.48

⌚ Backend Servers:
- web-1: 98.92.216.122 (private: 10.0.10.67)
  - web-2: 35.175.111.154 (private: 10.0.10.86)

```

task8\_nginx\_conf\_load\_balancer.png



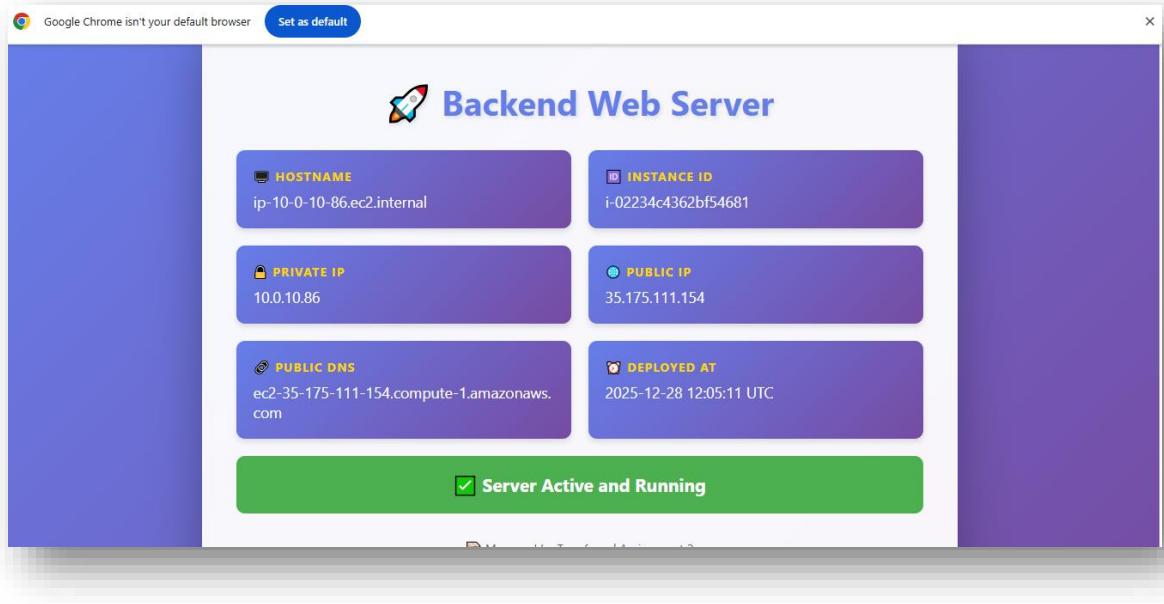
Task8\_nginx\_restart.png

```
nginx: configuration file /etc/nginx/nginx.conf test is successful
[ec2-user@ip-10-0-10-158 ~]$ sudo systemctl test nginx
[ec2-user@ip-10-0-10-158 ~]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
   Active: active (running) since Sun 2025-12-28 13:18:41 UTC; 13s ago
     Process: 68826 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
    Process: 68827 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 68828 (nginx)
      Tasks: 5 (limit: 1065)
     Memory: 4.4M
        CPU: 63ms
       CGroup: /system.slice/nginx.service
               ├─68828 "nginx: master process /usr/sbin/nginx"
               ├─68829 "nginx: worker process"
               ├─68830 "nginx: worker process"
               ├─68831 "nginx: cache manager process"
               └─68832 "nginx: cache loader process"

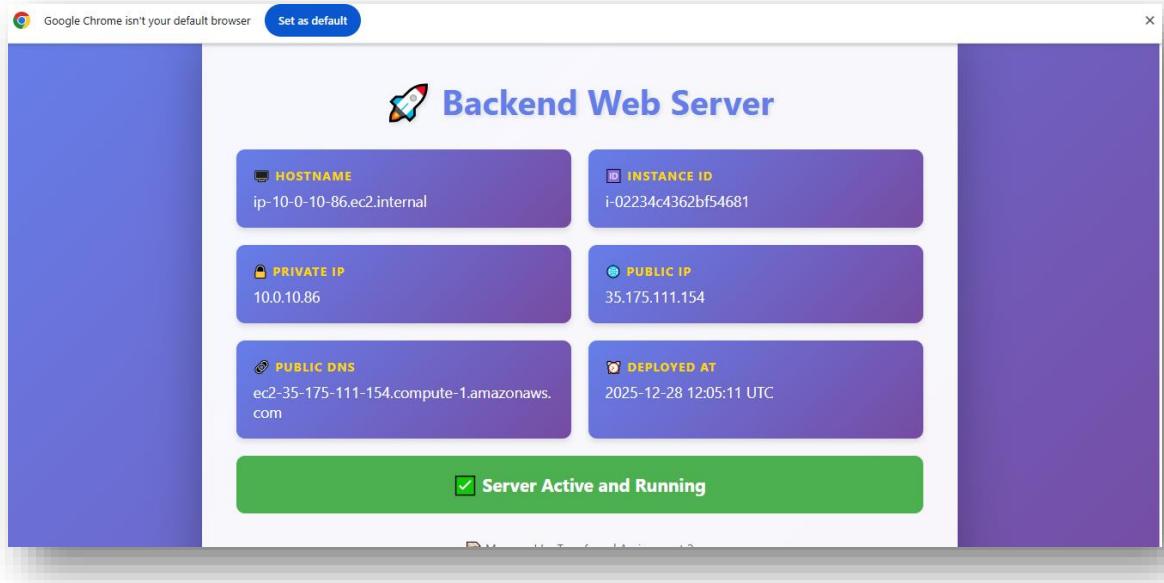
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal nginx[68826]: nginx: [warn] the "listen http" directive is deprecated, use the "http2" directive instead in /etc/nginx/nginx.conf:52
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal nginx[68826]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal nginx[68827]: nginx: [warn] the "listen ... http2" directive is deprecated, use the "http2" directive instead in /etc/nginx/nginx.conf:52
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[ec2-user@ip-10-0-10-158 ~]$
```

## TASK 9 - High Availability

task9\_ha\_web1\_only.png



task9\_ha\_web2\_only.png



## TASK 10 – Caching

task10\_nginx\_restart.png

```

nginx: configuration file /etc/nginx/nginx.conf test is successful
[ec2-user@ip-10-0-10-158 ~]$ sudo systemctl restart nginx
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
   Active: active (running) since Sun 2025-12-28 13:18:41 UTC; 13s ago
     Process: 68826 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
    Process: 68827 ExecStart=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
   Main PID: 68828 (nginx)
      Tasks: 1 (limit: 1065)
        Memory: 4.4M
          CPU: 6ms
         CGroup: /system.slice/nginx.service
             └─68828 "nginx: master process /usr/sbin/nginx"
               ├─68829 "nginx: worker process"
               ├─68830 "nginx: worker process"
               ├─68831 "nginx: cache manager process"
               └─68832 "nginx: cache loader process"

Dec 28 13:18:41 ip-10-0-10-158.ec2.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal nginx[68826]: nginx: [warn] the "listen ..." directive is deprecated, use the "http2" directive instead in /etc/nginx/nginx.conf:52
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal nginx[68826]: nginx: configuration file /etc/nginx/nginx.conf syntax is ok
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal nginx[68826]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal nginx[68827]: nginx: [warn] the "listen ... http2" directive is deprecated, use the "http2" directive instead in /etc/nginx/nginx.conf:52
Dec 28 13:18:41 ip-10-0-10-158.ec2.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[ec2-user@ip-10-0-10-158 ~]$
```

task10\_cache\_miss.png, task10\_cache\_hit.png

```

Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ # Get Nginx IP
NGINX_IP=$(terraform output -raw nginx_public_ip)

# First request - should show MISS
curl -k -I https://$NGINX_IP 2>&1 | grep -i "x-cache"

# Wait 2 seconds, then second request - should show HIT
sleep 2
curl -k -I https://$NGINX_IP 2>&1 | grep -i "x-cache"

# Third request - should show HIT
curl -k -I https://$NGINX_IP 2>&1 | grep -i "x-cache"
X-Cache-Status: MISS
X-Cache-Status: HIT
X-Cache-Status: HIT
```

task10\_cache\_directory.png

```

Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ # Get Nginx IP
NGINX_IP=$(terraform output -raw nginx_public_ip)

# First request - should show MISS
curl -k -I https://$NGINX_IP 2>&1 | grep -i "x-cache"

# Wait 2 seconds, then second request - should show HIT
sleep 2
curl -k -I https://$NGINX_IP 2>&1 | grep -i "x-cache"

# Third request - should show HIT
curl -k -I https://$NGINX_IP 2>&1 | grep -i "x-cache"
X-Cache-Status: MISS
X-Cache-Status: HIT
X-Cache-Status: HIT
```

cleanup\_destroy\_prompt.png

```

Dell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ terraform destroy
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 0s [id=https://icanhazip.com]
module.nginx_server.data.aws_amazon_linux: Reading...
module.backend_servers["web-1"].data.aws_amazon_linux: Reading...
module.backend_servers["web-2"].data.aws_amazon_linux: Reading...
module.backend_servers["web-3"].data.aws_amazon_linux: Reading...
module.backend_servers["web-2"].aws_key_pair.server_key: Refreshing state... [id=]
module.networking.aws_vpc.main: Refreshing state... [id=vpc-0df4a962ae2d7806a]
```

### cleanup\_destroy\_complete.png

```
module.nginx_server.aws_instance.server: Still destroying... [id=i-0e668eb55f575d3c4, 00m10s elapsed]
module.backend_servers["web-2"].aws_instance.server: Still destroying... [id=i-02234c4362bf54681, 00m10s elapsed]
module.backend_servers["web-3"].aws_instance.server: Still destroying... [id=i-09ade9131df87388b, 00m10s elapsed]
module.backend_servers["web-1"].aws_instance.server: Still destroying... [id=i-049a54d483fd72337, 00m10s elapsed]
module.networking.aws_internet_gateway.main: Still destroying... [id=igw-0c91f1b04f7851086, 00m10s elapsed]
module.nginx_server.aws_instance.server: Still destroying... [id=i-0e668eb55f575d3c4, 00m20s elapsed]
module.backend_servers["web-2"].aws_instance.server: Still destroying... [id=i-02234c4362bf54681, 00m20s elapsed]
module.backend_servers["web-3"].aws_instance.server: Still destroying... [id=i-09ade9131df87388b, 00m20s elapsed]
module.backend_servers["web-1"].aws_instance.server: Still destroying... [id=i-049a54d483fd72337, 00m20s elapsed]
module.networking.aws_internet_gateway.main: Still destroying... [id=igw-0c91f1b04f7851086, 00m20s elapsed]
module.backend_servers["web-2"].aws_instance.server: Still destroying... [id=i-02234c4362bf54681, 00m30s elapsed]
module.nginx_server.aws_instance.server: Still destroying... [id=i-0e668eb55f575d3c4, 00m30s elapsed]
module.backend_servers["web-3"].aws_instance.server: Still destroying... [id=i-09ade9131df87388b, 00m30s elapsed]
module.backend_servers["web-1"].aws_instance.server: Still destroying... [id=i-049a54d483fd72337, 00m30s elapsed]
module.networking.aws_internet_gateway.main: Still destroying... [id=igw-0c91f1b04f7851086, 00m30s elapsed]
module.nginx_server.aws_instance.server: Destruction complete after 36s
module.nginx_server.aws_key_pair.server_key: Destroying... [id=prod-key-nginx]
module.nginx_server.aws_key_pair.server_key: Destruction complete after 1s
module.backend_servers["web-2"].aws_instance.server: Still destroying... [id=i-02234c4362bf54681, 00m40s elapsed]
module.backend_servers["web-3"].aws_instance.server: Still destroying... [id=i-09ade9131df87388b, 00m40s elapsed]
module.backend_servers["web-1"].aws_instance.server: Still destroying... [id=i-049a54d483fd72337, 00m40s elapsed]
module.networking.aws_internet_gateway.main: Still destroying... [id=igw-0c91f1b04f7851086, 00m40s elapsed]
module.backend_servers["web-2"].aws_instance.server: Destruction complete after 47s
module.backend_servers["web-3"].aws_instance.server: Destruction complete after 1s
module.backend_servers["web-1"].aws_instance.server: Still destroying... [id=i-049a54d483fd72337, 00m50s elapsed]
module.backend_servers["web-3"].aws_instance.server: Still destroying... [id=i-09ade9131df87388b, 00m50s elapsed]
module.networking.aws_internet_gateway.main: Still destroying... [id=igw-0c91f1b04f7851086, 00m50s elapsed]
module.backend_servers["web-1"].aws_instance.server: Destruction complete after 56s
module.backend_servers["web-3"].aws_key_pair.server_key: Destroying... [id=prod-key-3]
module.networking.aws_internet_gateway.main: Destruction complete after 53s
module.backend_servers["web-3"].aws_key_pair.server_key: Destruction complete after 1s
module.backend_servers["web-1"].aws_instance.server: Still destroying... [id=i-049a54d483fd72337, 01m00s elapsed]
module.backend_servers["web-1"].aws_instance.server: Still destroying... [id=i-049a54d483fd72337, 01m10s elapsed]
module.backend_servers["web-1"].aws_instance.server: Destruction complete after 1m18s
module.backend_servers["web-1"].aws_key_pair.server_key: Destroying... [id=prod-key-1]
module.networking.aws_subnet.main: Destroying... [id=subnet-0d8278da93817bc6b]
module.security.aws_security_group.backend: Destroying... [id=sg-0496a4a11080a0c67]
module.backend.servers["web-1"].aws_key_pair.server_key: Destruction complete after 1s
module.networking.aws_subnet.main: Destruction complete after 1s
module.security.aws_security_group.backend: Destruction complete after 1s
module.security.aws_security_group.nginx: Destroying... [id=sq-0c21a7dd44f73d0c]
module.security.aws_security_group.nginx: Destruction complete after 2s
module.networking.aws_vpc.main: Destroying... [id=vpc-0df4a962ae2d7806a]
module.networking.aws_vpc.main: Destruction complete after 2s
```

```
Destroy complete! Resources: 15 destroyed.
```

### cleanup\_state\_empty.png

```
jell@DESKTOP-JNCVEJH MINGW64 ~/Documents/CC_areej_10/Assignment2 (main)
$ cat terraform.tfstate
{
  "version": 4,
  "terraform_version": "1.14.3",
  "serial": 42,
  "lineage": "f9e697a3-fddb-6bda-705c-b2cea6f29e4d",
  "outputs": {},
  "resources": [],
  "check_results": [
    {
      "object_kind": "var",
      "config_addr": "var.vpc_cidr_block",
      "status": "unknown",
      "objects": null
    },
    {
      "object_kind": "var",
      "config_addr": "var.subnet_cidr_block",
      "status": "unknown",
      "objects": null
    }
  ]
}
```

