LAB 10 - Cloud Computing

Course Student Name:

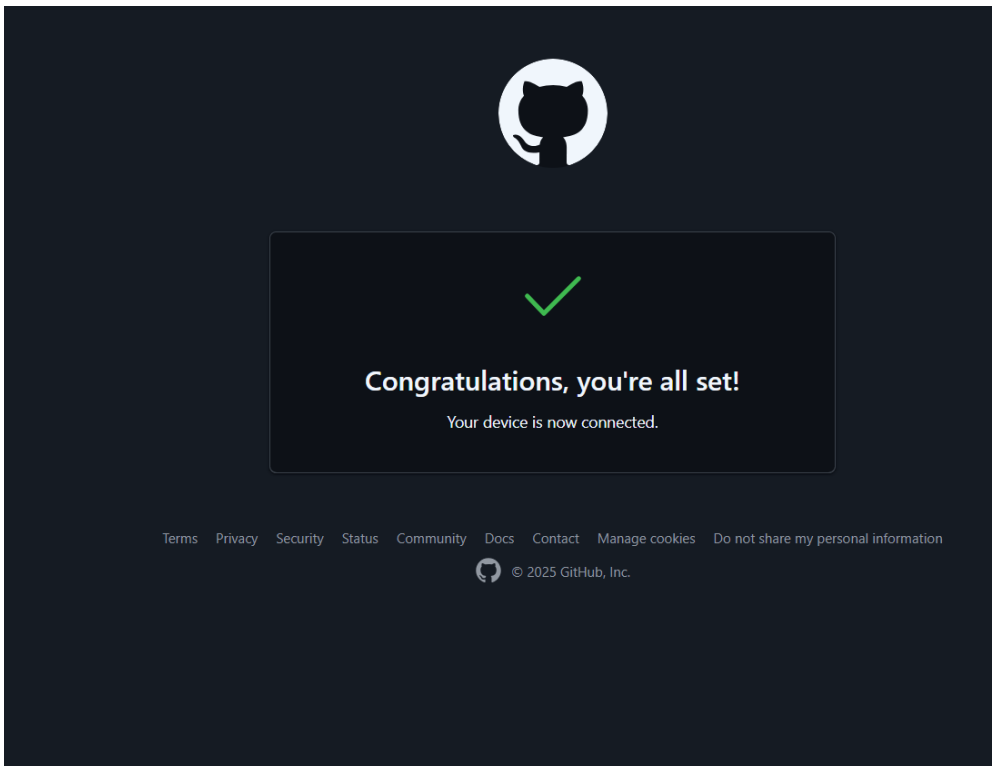
Arshia Jadoon
Roll Number: 014

Course:
CloudComputing
Instructor:

Sir Waqas Saleem

Institution: FJWU

task1_gh_auth_login.png



task1_codespace_list.png

```
@arshiajadoon ➜ /workspaces/CC_014_Arshia-Lab10 (main) $ gh codespace list
NAME                          DISPLAY NAME  REPOSITORY                       BRANCH  STATE      CREATED AT
shiny-winner-q754q6rjw9qwhxp9  shiny winner  arshiajadoon/CC_014_Arshia-Lab10  main    Available  about 3 minutes ago
@arshiajadoon ➜ /workspaces/CC_014_Arshia-Lab10 (main) $
```

task1_codespace_ssh_connected.png

```
  ✓ Codespaces usage for this repository is paid for by arshiajadoon
? Choose Machine Type: 2 cores, 8 GB RAM, 32 GB storage
shiny-winner-q754q6rjw9qwhxp9
PS C:\Users\hp> gh codespace ssh -c shiny-winner-q754q6rjw9qwhxp9
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.

@arshiajadoon ➜ /workspaces/CC_014_Arshia-Lab10 (main) $
```

task2_aws_install_and_version.png

```
PS C:\Users\hp> aws configure
AWS Access Key ID [None]: AKIAVEWO74P2FM5KLY4V
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json
PS C:\Users\hp> aws sts get-caller-identity
{
    "UserId": "353695163380",
    "Account": "353695163380",
    "Arn": "arn:aws:iam::353695163380:root"
}

PS C:\Users\hp>
```

task2_aws_configure_and_files.png

```
Dell@DESKTOP-JNCVEJH MINGW64 ~ (main)
$ aws configure
AWS Access Key ID [None]: AKIAVLLVFEFY4DCHTS4F
AWS Secret Access Key [None]: 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
Default region name [None]: me-central-1
Default output format [None]: json

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

task2_aws_get_caller_identity.png

```
PS C:\Windows\system32> aws --version
>> terraform --version
aws-cli/2.32.24 Python/3.13.11 Windows/10 exe/AMD64
Terraform v1.14.3
on windows_amd64
PS C:\Windows\system32>
```

task2_terraform_install_and_version.png

```
PS C:\Users\hp\Desktop\Assignment2> 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/aws v5.100.0...
- Installed hashicorp/aws v5.100.0 (signed by HashiCorp)
- Installing hashicorp/http v3.5.0...
- Installed hashicorp/http v3.5.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

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

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

task2_provider_file_creation.png

main.tf

```hcl
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
  required_version = ">= 1.0"
}

provider "aws" {
  region = "me-central-1"
}

# Networking Module
module "networking" {
  source = "./modules/networking"

  vpc_cidr_block    = var.vpc_cidr_block
  subnet_cidr_block = var.subnet_cidr_block
  availability_zone = var.availability_zone
  env_prefix        = var.env_prefix
  common_tags       = local.common_tags
}

# Security Module
module "security" {
  source = "./modules/security"

  vpc_id      = module.networking.vpc_id
  env_prefix  = var.env_prefix
  my_ip       = local.my_ip
  common_tags = local.common_tags
}
```

task2_provider_block.png

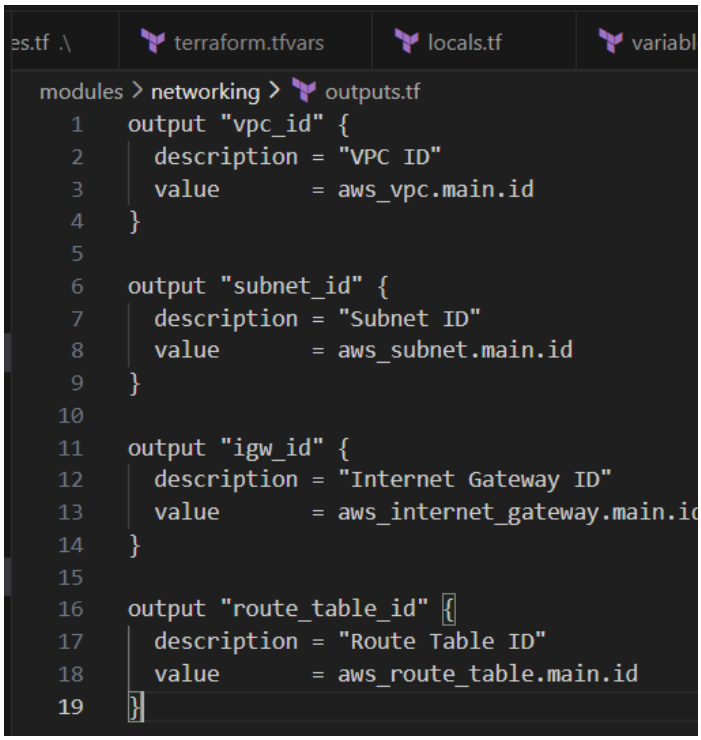
main.tf

```hcl
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
  required_version = ">= 1.0"
}

provider "aws" {
  region = "me-central-1"
}

# Networking Module
module "networking" {
  source = "./modules/networking"

  vpc_cidr_block    = var.vpc_cidr_block
  subnet_cidr_block = var.subnet_cidr_block
  availability_zone = var.availability_zone
  env_prefix        = var.env_prefix
  common_tags       = local.common_tags
}

# Security Module
module "security" {
  source = "./modules/security"

  vpc_id      = module.networking.vpc_id
  env_prefix  = var.env_prefix
  my_ip       = local.my_ip
  common_tags = local.common_tags
}
```

task2_terraform_init_output.png

```
PS C:\Users\hp\Desktop\Assignment2> 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/aws v5.100.0...
- Installed hashicorp/aws v5.100.0 (signed by HashiCorp)
- Installing hashicorp/http v3.5.0...
- Installed hashicorp/http v3.5.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

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

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

task3_main_tf_resource_add.png

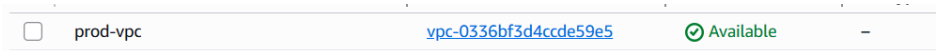


task3_terraform_apply_vpc_subnet.png

```
configuration_guide = <<EOT

========================================
DEPLOYMENT SUCCESSFUL!
========================================

Next Steps:
1. SSH into Nginx server: ssh -i C:/Users/hp/.ssh/id_ed25519 ec2-user@3.235.237.169
2. Edit Nginx config: sudo vim /etc/nginx/nginx.conf
3. Update backend IPs in upstream block:
   - BACKEND_IP_1: 10.0.10.64
   - BACKEND_IP_2: 10.0.10.135
   - BACKEND_IP_3: 10.0.10.214
4. Restart Nginx: sudo systemctl restart nginx
5. Test: https://3.235.237.169

Backend Servers:
- web-1: 3.219.218.255 (private: 10.0.10.64)
    - web-2: 44.200.134.1 (private: 10.0.10.135)
    - web-3: 98.84.56.143 (private: 10.0.10.214)

========================================

EOT
nginx_instance_id = "i-0a844d0411e192d31"
nginx_public_ip = "3.235.237.169"
subnet_id = "subnet-0b87e8a2dac16cd92"
vpc_id = "vpc-0336bf3d4ccde59e5"
PS C:\Users\hp\Desktop\Assignment2>
```

task3_aws_cli_verify_subnet.png

task3_aws_cli_verify_vpc.png



task4_terraform_apply_datasource_resource.png

```
configuration_guide = <<EOT

========================================
DEPLOYMENT SUCCESSFUL!
========================================

Next Steps:
1. SSH into Nginx server: ssh -i C:/Users/hp/.ssh/id_ed25519 ec2-user@3.235.237.169
2. Edit Nginx config: sudo vim /etc/nginx/nginx.conf
3. Update backend IPs in upstream block:
   - BACKEND_IP_1: 10.0.10.64
   - BACKEND_IP_2: 10.0.10.135
   - BACKEND_IP_3: 10.0.10.214
4. Restart Nginx: sudo systemctl restart nginx
5. Test: https://3.235.237.169

Backend Servers:
- web-1: 3.219.218.255 (private: 10.0.10.64)
    - web-2: 44.200.134.1 (private: 10.0.10.135)
    - web-3: 98.84.56.143 (private: 10.0.10.214)

========================================

EOT
nginx_instance_id = "i-0a844d0411e192d31"
nginx_public_ip = "3.235.237.169"
subnet_id = "subnet-0b87e8a2dac16cd92"
vpc_id = "vpc-0336bf3d4ccde59e5"
PS C:\Users\hp\Desktop\Assignment2>
```

task4_terraform_destroy_targeted.png

```
hp@DESKTOP-44CCB99 MINGW64 ~/Desktop/Assignment2 (main)
$ curl -I http://3.235.237.169
curl: (7) Failed to connect to 3.235.237.169 port 80 after 3219 ms: Could not co
nnect to server

hp@DESKTOP-44CCB99 MINGW64 ~/Desktop/Assignment2 (main)
$
```

task4_terraform_apply_after_refresh.png

```
configuration_guide = <<EOT

========================================
DEPLOYMENT SUCCESSFUL!
========================================

Next Steps:
1. SSH into Nginx server: ssh -i C:/Users/hp/.ssh/id_ed25519 ec2-user@3.235.237.169
2. Edit Nginx config: sudo vim /etc/nginx/nginx.conf
3. Update backend IPs in upstream block:
   - BACKEND_IP_1: 10.0.10.64
   - BACKEND_IP_2: 10.0.10.135
   - BACKEND_IP_3: 10.0.10.214
4. Restart Nginx: sudo systemctl restart nginx
5. Test: https://3.235.237.169

Backend Servers:
- web-1: 3.219.218.255 (private: 10.0.10.64)
    - web-2: 44.200.134.1 (private: 10.0.10.135)
    - web-3: 98.84.56.143 (private: 10.0.10.214)

========================================

EOT
nginx_instance_id = "i-0a844d0411e192d31"
nginx_public_ip = "3.235.237.169"
subnet_id = "subnet-0b87e8a2dac16cd92"
vpc_id = "vpc-0336bf3d4ccde59e5"
PS C:\Users\hp\Desktop\Assignment2>
```

task4_terraform_destroy_all.png

```
]
module.security.aws_security_group.nginx: Destruction complete after 1s
module.networking.aws_vpc.main: Destroying... [id=vpc-0336bf3d4ccde59e5]
module.networking.aws_vpc.main: Destruction complete after 1s

Destroy complete! Resources: 15 destroyed.
```

task4_terraform_plan_output.png

```
PS C:\Users\hp\Desktop\Assignment2> terraform plan
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 1s [id=https://icanhazip.com]
data.aws_ami.amazon_linux: Reading...
data.aws_ami.amazon_linux: Read complete after 2s [id=ami-0ca688f4217aab61b]

Terraform used the selected providers to generate the following execution plan
following symbols:
  + create

Terraform planned the following actions, but then encountered a problem:

  # module.networking.aws_internet_gateway.main will be created
  + resource "aws_internet_gateway" "main" {
      + arn      = (known after apply)
      + id       = (known after apply)
      + owner_id = (known after apply)
      + tags     = {
          + "Environment" = "prod"
          + "ManagedBy"   = "Terraform"
          + "Name"        = "prod-igw"
          + "Project"     = "Assignment-2"
        }
      + tags_all = {
          + "Environment" = "prod"
```

```
  # module.networking.aws_route_table.main will be created
  + resource "aws_route_table" "main" {
      + arn              = (known after apply)
      + id               = (known after apply)
      + owner_id         = (known after apply)
      + propagating_vgws = (known after apply)
      + route            = [
          + {
              + cidr_block                = "0.0.0.0/0"
              + gateway_id                = (known after apply)
                # (11 unchanged attributes hidden)
            },
        ]
      + tags             = {
          + "Environment" = "prod"
          + "ManagedBy"   = "Terraform"
          + "Name"        = "prod-rtb"
          + "Project"     = "Assignment-2"
        }
      + tags_all         = {
          + "Environment" = "prod"
          + "ManagedBy"   = "Terraform"
          + "Name"        = "prod-rtb"
          + "Project"     = "Assignment-2"
        }
      + vpc_id           = (known after apply)
    }

  # module.networking.aws_route_table_association.main will be created
```

```
  + resource "aws_route_table_association" "main" {
      + id             = (known after apply)
      + route_table_id = (known after apply)
      + subnet_id      = (known after apply)
    }

  # module.networking.aws_subnet.main will be created
  + resource "aws_subnet" "main" {
      + arn                                            = (known after apply)
      + assign_ipv6_address_on_creation                = false
      + availability_zone                              = "us-east-1"
      + availability_zone_id                           = (known after apply)
      + cidr_block                                     = "10.0.10.0/24"
      + enable_dns64                                   = false
      + enable_resource_name_dns_a_record_on_launch    = false
      + enable_resource_name_dns_aaaa_record_on_launch = false
      + id                                             = (known after apply)
      + ipv6_cidr_block_association_id                 = (known after apply)
      + ipv6_native                                    = false
      + map_public_ip_on_launch                        = true
      + owner_id                                       = (known after apply)
      + private_dns_hostname_type_on_launch            = (known after apply)
      + tags                                           = {
          + "Environment" = "prod"
          + "ManagedBy"   = "Terraform"
          + "Name"        = "prod-subnet"
          + "Project"     = "Assignment-2"
        }
      + tags_all                                       = {
          + "Environment" = "prod"
```

task4_terraform_apply_after_destroy.png

```
configuration_guide = <<EOT

========================================
DEPLOYMENT SUCCESSFUL!
========================================

Next Steps:
1. SSH into Nginx server: ssh -i C:/Users/hp/.ssh/id_ed25519 ec2-user@3.235.237.169
2. Edit Nginx config: sudo vim /etc/nginx/nginx.conf
3. Update backend IPs in upstream block:
   - BACKEND_IP_1: 10.0.10.64
   - BACKEND_IP_2: 10.0.10.135
   - BACKEND_IP_3: 10.0.10.214
4. Restart Nginx: sudo systemctl restart nginx
5. Test: https://3.235.237.169

Backend Servers:
- web-1: 3.219.218.255 (private: 10.0.10.64)
    - web-2: 44.200.134.1 (private: 10.0.10.135)
    - web-3: 98.84.56.143 (private: 10.0.10.214)

========================================

EOT
nginx_instance_id = "i-0a844d0411e192d31"
nginx_public_ip = "3.235.237.169"
subnet_id = "subnet-0b87e8a2dac16cd92"
vpc_id = "vpc-0336bf3d4ccde59e5"
PS C:\Users\hp\Desktop\Assignment2>
```

task4_terraform_apply_tagging.png

```
configuration_guide = <<EOT

========================================
DEPLOYMENT SUCCESSFUL!
========================================

Next Steps:
1. SSH into Nginx server: ssh -i C:/Users/hp/.ssh/id_ed25519 ec2-user@3.235.237.169
2. Edit Nginx config: sudo vim /etc/nginx/nginx.conf
3. Update backend IPs in upstream block:
   - BACKEND_IP_1: 10.0.10.64
   - BACKEND_IP_2: 10.0.10.135
   - BACKEND_IP_3: 10.0.10.214
4. Restart Nginx: sudo systemctl restart nginx
5. Test: https://3.235.237.169

Backend Servers:
- web-1: 3.219.218.255 (private: 10.0.10.64)
    - web-2: 44.200.134.1 (private: 10.0.10.135)
    - web-3: 98.84.56.143 (private: 10.0.10.214)

========================================

EOT
nginx_instance_id = "i-0a844d0411e192d31"
nginx_public_ip = "3.235.237.169"
subnet_id = "subnet-0b87e8a2dac16cd92"
vpc_id = "vpc-0336bf3d4ccde59e5"
PS C:\Users\hp\Desktop\Assignment2>
```

task4_terraform_plan_remove_tag.png

```
PS C:\Users\hp\Desktop\Assignment2> terraform plan
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 1s [id=https://icanhazip.com]
data.aws_ami.amazon_linux: Reading...
data.aws_ami.amazon_linux: Read complete after 2s [id=ami-0ca688f4217aab61b]

Terraform used the selected providers to generate the following execution plan
following symbols:
  + create

Terraform planned the following actions, but then encountered a problem:

  # module.networking.aws_internet_gateway.main will be created
  + resource "aws_internet_gateway" "main" {
      + arn      = (known after apply)
      + id       = (known after apply)
      + owner_id = (known after apply)
      + tags     = {
          + "Environment" = "prod"
          + "ManagedBy"   = "Terraform"
          + "Name"        = "prod-igw"
          + "Project"     = "Assignment-2"
        }
      + tags_all = {
          + "Environment" = "prod"
```

```
  # module.networking.aws_route_table.main will be created
  + resource "aws_route_table" "main" {
      + arn              = (known after apply)
      + id               = (known after apply)
      + owner_id         = (known after apply)
      + propagating_vgws = (known after apply)
      + route            = [
          + {
              + cidr_block                = "0.0.0.0/0"
              + gateway_id                = (known after apply)
                # (11 unchanged attributes hidden)
            },
        ]
      + tags             = {
          + "Environment" = "prod"
          + "ManagedBy"   = "Terraform"
          + "Name"        = "prod-rtb"
          + "Project"     = "Assignment-2"
        }
      + tags_all         = {
          + "Environment" = "prod"
          + "ManagedBy"   = "Terraform"
          + "Name"        = "prod-rtb"
          + "Project"     = "Assignment-2"
        }
      + vpc_id           = (known after apply)
    }

  # module.networking.aws_route_table_association.main will be created
```

```
+ resource "aws_route_table_association" "main" {
    + id             = (known after apply)
    + route_table_id = (known after apply)
    + subnet_id      = (known after apply)
  }

# module.networking.aws_subnet.main will be created
+ resource "aws_subnet" "main" {
    + arn                                            = (known after apply)
    + assign_ipv6_address_on_creation                = false
    + availability_zone                              = "us-east-1"
    + availability_zone_id                           = (known after apply)
    + cidr_block                                     = "10.0.10.0/24"
    + enable_dns64                                   = false
    + enable_resource_name_dns_a_record_on_launch    = false
    + enable_resource_name_dns_aaaa_record_on_launch = false
    + id                                             = (known after apply)
    + ipv6_cidr_block_association_id                 = (known after apply)
    + ipv6_native                                    = false
    + map_public_ip_on_launch                        = true
    + owner_id                                       = (known after apply)
    + private_dns_hostname_type_on_launch            = (known after apply)
    + tags                                           = {
        + "Environment" = "prod"
        + "ManagedBy"   = "Terraform"
        + "Name"        = "prod-subnet"
        + "Project"     = "Assignment-2"
      }
    + tags_all                                       = {
        + "Environment" = "prod"
```

task4_terraform_apply_remove_tag.png

```
configuration_guide = <<EOT

========================================
DEPLOYMENT SUCCESSFUL!
========================================

Next Steps:
1. SSH into Nginx server: ssh -i C:/Users/hp/.ssh/id_ed25519 ec2-user@3.235.237.169
2. Edit Nginx config: sudo vim /etc/nginx/nginx.conf
3. Update backend IPs in upstream block:
   - BACKEND_IP_1: 10.0.10.64
   - BACKEND_IP_2: 10.0.10.135
   - BACKEND_IP_3: 10.0.10.214
4. Restart Nginx: sudo systemctl restart nginx
5. Test: https://3.235.237.169

Backend Servers:
- web-1: 3.219.218.255 (private: 10.0.10.64)
    - web-2: 44.200.134.1 (private: 10.0.10.135)
    - web-3: 98.84.56.143 (private: 10.0.10.214)

========================================

EOT
nginx_instance_id = "i-0a844d0411e192d31"
nginx_public_ip = "3.235.237.169"
subnet_id = "subnet-0b87e8a2dac16cd92"
vpc_id = "vpc-0336bf3d4ccde59e5"
PS C:\Users\hp\Desktop\Assignment2>
```

task5_terraform_destroy.png

```
]
module.security.aws_security_group.nginx: Destruction complete after 1s
module.networking.aws_vpc.main: Destroying... [id=vpc-0336bf3d4ccde59e5]
module.networking.aws_vpc.main: Destruction complete after 1s

Destroy complete! Resources: 15 destroyed.
```

task5_terraform_state_file_empty.png

```
PS C:\Users\hp\Desktop\Assignment2> 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/aws v5.100.0...
- Installed hashicorp/aws v5.100.0 (signed by HashiCorp)
- Installing hashicorp/http v3.5.0...
- Installed hashicorp/http v3.5.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

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

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

task5_terraform_state_backup_prev.png

```
PS C:\Users\hp\Desktop\Assignment2> terraform validate
Success! The configuration is valid.
```

task5_terraform_apply_recreated.png

```
configuration_guide = <<EOT

========================================
DEPLOYMENT SUCCESSFUL!
========================================

Next Steps:
1. SSH into Nginx server: ssh -i C:/Users/hp/.ssh/id_ed25519 ec2-user@3.235.237.169
2. Edit Nginx config: sudo vim /etc/nginx/nginx.conf
3. Update backend IPs in upstream block:
   - BACKEND_IP_1: 10.0.10.64
   - BACKEND_IP_2: 10.0.10.135
   - BACKEND_IP_3: 10.0.10.214
4. Restart Nginx: sudo systemctl restart nginx
5. Test: https://3.235.237.169

Backend Servers:
- web-1: 3.219.218.255 (private: 10.0.10.64)
    - web-2: 44.200.134.1 (private: 10.0.10.135)
    - web-3: 98.84.56.143 (private: 10.0.10.214)

========================================

EOT
nginx_instance_id = "i-0a844d0411e192d31"
nginx_public_ip = "3.235.237.169"
subnet_id = "subnet-0b87e8a2dac16cd92"
vpc_id = "vpc-0336bf3d4ccde59e5"
PS C:\Users\hp\Desktop\Assignment2>
```

task5_terraform_state_file_populated.png

```
PS C:\Users\hp\Desktop\Assignment2> terraform plan
data.http.my_ip: Reading...
data.http.my_ip: Read complete after 1s [id=https://icanhazip.com]
data.aws_ami.amazon_linux: Reading...
data.aws_ami.amazon_linux: Read complete after 2s [id=ami-0ca688f4217aab61b]

Terraform used the selected providers to generate the following execution plan
following symbols:
  + create

Terraform planned the following actions, but then encountered a problem:

  # module.networking.aws_internet_gateway.main will be created
  + resource "aws_internet_gateway" "main" {
      + arn      = (known after apply)
      + id       = (known after apply)
      + owner_id = (known after apply)
      + tags     = {
          + "Environment" = "prod"
          + "ManagedBy"   = "Terraform"
          + "Name"        = "prod-igw"
          + "Project"     = "Assignment-2"
        }
      + tags_all = {
          + "Environment" = "prod"
```

task5_terraform_state_backup_empty.png

```
  # module.networking.aws_route_table.main will be created
  + resource "aws_route_table" "main" {
      + arn              = (known after apply)
      + id               = (known after apply)
      + owner_id         = (known after apply)
      + propagating_vgws = (known after apply)
      + route            = [
          + {
              + cidr_block                = "0.0.0.0/0"
              + gateway_id                = (known after apply)
                # (11 unchanged attributes hidden)
            },
        ]
      + tags             = {
          + "Environment" = "prod"
          + "ManagedBy"   = "Terraform"
          + "Name"        = "prod-rtb"
          + "Project"     = "Assignment-2"
        }
      + tags_all         = {
          + "Environment" = "prod"
          + "ManagedBy"   = "Terraform"
          + "Name"        = "prod-rtb"
          + "Project"     = "Assignment-2"
        }
      + vpc_id           = (known after apply)
    }

  # module.networking.aws_route_table_association.main will be created
```

task5_terraform_state_list.png

```
+ resource "aws_route_table_association" "main" {
    + id             = (known after apply)
    + route_table_id = (known after apply)
    + subnet_id      = (known after apply)
  }

# module.networking.aws_subnet.main will be created
+ resource "aws_subnet" "main" {
    + arn                                            = (known after apply)
    + assign_ipv6_address_on_creation                = false
    + availability_zone                              = "us-east-1"
    + availability_zone_id                           = (known after apply)
    + cidr_block                                     = "10.0.10.0/24"
    + enable_dns64                                   = false
    + enable_resource_name_dns_a_record_on_launch    = false
    + enable_resource_name_dns_aaaa_record_on_launch = false
    + id                                             = (known after apply)
    + ipv6_cidr_block_association_id                 = (known after apply)
    + ipv6_native                                    = false
    + map_public_ip_on_launch                        = true
    + owner_id                                       = (known after apply)
    + private_dns_hostname_type_on_launch            = (known after apply)
    + tags                                           = {
        + "Environment" = "prod"
        + "ManagedBy"   = "Terraform"
        + "Name"        = "prod-subnet"
        + "Project"     = "Assignment-2"
      }
    + tags_all                                       = {
        + "Environment" = "prod"
```

task5_terraform_state_show_resource.png

```
============\r\n    \r\nNext Steps:\r\n1. SSH into Nginx server: ssh -i C:/Users/hp/.ssh/
169\r\n2. Edit Nginx config: sudo vim /etc/nginx/nginx.conf\r\n3. Update backend IPs in up
_IP_1: 10.0.10.64\r\n   - BACKEND_IP_2: 10.0.10.135\r\n   - BACKEND_IP_3: 10.0.10.214\r\n4
l restart nginx\r\n5. Test: https://3.235.237.169\r\n    \r\nBackend Servers:\r\n- web-1:
0.64)\n    - web-2: 44.200.134.1 (private: 10.0.10.135)\n    - web-3: 98.84.56.143 (privat
====================================\r\n"
  },
  "nginx_instance_id": {
    "sensitive": false,
    "type": "string",
    "value": "i-0a844d0411e192d31"
  },
  "nginx_public_ip": {
    "sensitive": false,
    "type": "string",
    "value": "3.235.237.169"
  },
  "subnet_id": {
    "sensitive": false,
    "type": "string",
    "value": "subnet-0b87e8a2dac16cd92"
  },
  "vpc_id": {
    "sensitive": false,
    "type": "string",
    "value": "vpc-0336bf3d4ccde59e5"
  }
}
```

task6_terraform_outputs_basic.png

```
========================================
DEPLOYMENT SUCCESSFUL!
========================================

Next Steps:
1. SSH into Nginx server: ssh -i C:/Users/hp/.ssh/id_ed25519 ec2-user@3.235.237.169
2. Edit Nginx config: sudo vim /etc/nginx/nginx.conf
3. Update backend IPs in upstream block:
   - BACKEND_IP_1: 10.0.10.64
   - BACKEND_IP_2: 10.0.10.135
   - BACKEND_IP_3: 10.0.10.214
4. Restart Nginx: sudo systemctl restart nginx
5. Test: https://3.235.237.169

Backend Servers:
- web-1: 3.219.218.255 (private: 10.0.10.64)
    - web-2: 44.200.134.1 (private: 10.0.10.135)
    - web-3: 98.84.56.143 (private: 10.0.10.214)

========================================

EOT
nginx_instance_id = "i-0a844d0411e192d31"
nginx_public_ip = "3.235.237.169"
subnet_id = "subnet-0b87e8a2dac16cd92"
vpc_id = "vpc-0336bf3d4ccde59e5"
PS C:\Users\hp\Desktop\Assignment2>
```

task6_expanded_outputs.png

cleanup_destroy_resources.png