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

LAB#11

TASK 1: Provider & Variable Precedence

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ touch main.tf
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ vim main.tf
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ cat vim.tf
cat: vim.tf: No such file or directory
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ cat main.tf
provider "aws" {
  shared_config_files      = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
}
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.27.0...
- Installed hashicorp/aws v6.27.0 (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.
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve
var.subnet_cidr_block
  Enter a value: 10.0.10.0/24


Changes to Outputs:
  + subnet_cidr_block_output = "10.0.10.0/24"

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

Outputs:

subnet_cidr_block_output = "10.0.10.0/24"
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ export TF_VAR_subnet_cidr_block=10.0.20.0/24
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve

Changes to Outputs:
  ~ subnet_cidr_block_output = "10.0.0.0/24" -> "10.0.20.0/24"

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

Outputs:

subnet_cidr_block_output = "10.0.20.0/24"
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ touch terraform.tfvars
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ vim terraform.tfvars
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve

Changes to Outputs:
  ~ subnet_cidr_block_output = "10.0.20.0/24" -> "10.0.30.0/24"

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

Outputs:

subnet_cidr_block_output = "10.0.30.0/24"
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve -var "subnet_cidr_block=10.0.40.0/24
"

Changes to Outputs:
  ~ subnet_cidr_block_output = "10.0.30.0/24" -> "10.0.40.0/24"

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

Outputs:

subnet_cidr_block_output = "10.0.40.0/24"
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ printenv | grep TF_VAR_
TF_VAR_subnet_cidr_block=10.0.20.0/24
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ unset TF_VAR_subnet_cidr_block
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ printenv | grep TF_VAR_
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ nano main.tf
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve -var "subnet_cidr_block=10.0.0"

Error: Invalid value for variable

  on main.tf line 6:
   6: variable "subnet_cidr_block" {
    ├────────────────
    │ var.subnet_cidr_block is "10.0.0"

The subnet_cidr_block must be a valid CIDR notation string, such as 10.0.0.0/24.

This was checked by the validation rule at main.tf:14,3-13.
```

TASK 2: Variable Validation & Sensitive Variables

```
  GNU nano 7.2                                          main.tf
provider "aws" {
  shared_config_files      = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
}

variable "subnet_cidr_block" {
  type        = string
  default     = ""
  description = "CIDR block to assign to the application subnet"
  sensitive   = false
  nullable    = false
  ephemeral   = false

  validation {
    condition     = can(regex("^([0-9]{1,3}\\.){3}[0-9]{1,3}/[0-9]+$", var.subnet_cidr_block))
    error_message = "The subnet_cidr_block must be a valid CIDR notation string, such as 10.0.0.0/24."
  }
}

output "subnet_cidr_block_output" {
  value = var.subnet_cidr_block
}
variable "api_session_token" {
  type        = string
  default     = ""
  description = "Short□lived API session token used during apply operations"
  sensitive   = true
  nullable    = false
  ephemeral   = false

  validation {
    condition     = can(regex("^[A-Za-z0-9-_]{20,}$", var.api_session_token))
    error_message = "The API session token must be at least 20 characters and contain only letters, numbers, hyphens, o>
  }
}

output "api_session_token_output" {
  value     = var.api_session_token
  sensitive = true
}
```

```
@SanaTariq205 □ /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve -var "api_session_token=my_API_sessi
on_Token_12345"

Changes to Outputs:
  + api_session_token_output = (sensitive value)
  ~ subnet_cidr_block_output = "10.0.40.0/24" -> "10.0.30.0/24"

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

Outputs:

api_session_token_output = <sensitive>
subnet_cidr_block_output = "10.0.30.0/24"
```

```
@SanaTariq205 □ /workspaces/cc_sanatariq_058 (main) $ cat terraform.tfstate | grep -A 5 "api_session_token_output"
    "api_session_token_output": {
      "value": "my_API_session_Token_12345",
      "type": "string",
      "sensitive": true
    },
    "subnet_cidr_block_output": {
```

```
variable "api_session_token" {
  type        = string
  default     = ""
  description = "Short□lived API session token used during apply operations"
  sensitive   = true
  nullable    = false
  ephemeral   = true
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve -var "api_session_token=my_API_sessi
on_Token_12345"

  Error: Ephemeral value not allowed

    on main.tf line 38, in output "api_session_token_output":
    38:   value     = var.api_session_token

  This output value is not declared as returning an ephemeral value, so it cannot be set to a result derived from an
  ephemeral value.
```

```
variable "api_session_token" {
  type        = string
  default     = "my_API_session_Token_12345"
  description = "Short-lived API session token used during apply operations"
  sensitive   = true
  nullable    = false
  ephemeral   = false

  validation {
    condition     = can(regex("^[A-Za-z0-9-_]{20,}$", var.api_session_token))
    error_message = "The API session token must be at least 20 characters and contain only letters, numbers, hyphens, o
  }
}
```

TASK 3: Project Variables, Locals, Outputs

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are
needed.

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

Outputs:

api_session_token_output = <sensitive>
subnet_cidr_block_output = "10.0.30.0/24"
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ aws ec2 describe-subnets \
>   --filters "Name=availability-zone,Values=us-east-1a" \
>   --query "Subnets[].SubnetId" \
>   --output text
subnet-0640d6a9b71ac5322        subnet-00447af4668f7212a        subnet-0ea857b872b
```

```
  GNU nano 7.2                                                terraform.tfvars
environment = "dev"
project_name = "lab_work"
primary_subnet_id = "subnet-0640d6a9b71ac5322"
subnet_count = 3
monitoring = true
```

```
  GNU nano 7.2                                                  locals.tf *
locals {
  resource_name = "${var.project_name}-${var.environment}"
  primary_public_subnet = var.primary_subnet_id
  subnet_count          = var.subnet_count
  is_production         = var.environment == "prod"
  monitoring_enabled    = var.monitoring || local.is_production
}
```

```
  GNU nano 7.2                                                main.tf

output "subnet_cidr_block_output" {
  value = var.subnet_cidr_block
}
variable "api_session_token" {
  type        = string
  default     = "my_API_session_Token_12345"
  description = "Short□lived API session token used during apply opera
  sensitive   = true
  nullable    = false
  ephemeral   = false

  validation {
    condition     = can(regex("^[A-Za-z0-9-_]{20,}$", var.api_session_
    error_message = "The API session token must be at least 20 charact
  }
}

output "api_session_token_output" {
  value     = var.api_session_token
  sensitive = true
}
variable "environment" {}
variable "project_name" {}
variable "primary_subnet_id" {}
variable "subnet_count" {}
variable "monitoring" {}
output "resource_name" {
  value = local.resource_name
}

output "primary_public_subnet" {
  value = local.primary_public_subnet
}

output "subnet_count" {
  value = local.subnet_count
}

output "is_production" {
  value = local.is_production
}

output "monitoring_enabled" {
  value = local.monitoring_enabled
}
```

```
@SanaTariq205 □ /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve

Changes to Outputs:
  + is_production            = false
  + monitoring_enabled       = true
  + primary_public_subnet    = "subnet-0640d6a9b71ac5322"
  + resource_name            = "lab_work-dev"
  ~ subnet_cidr_block_output = "10.0.30.0/24" -> "10.0.20.0/24"
  + subnet_count             = 3

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

Outputs:

api_session_token_output = <sensitive>
is_production = false
monitoring_enabled = true
primary_public_subnet = "subnet-0640d6a9b71ac5322"
resource_name = "lab_work-dev"
subnet_cidr_block_output = "10.0.20.0/24"
subnet_count = 3
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve

Changes to Outputs:
  + tags                      = {
      + Environment = "dev"
      + Owner       = "platform-team"
      + Project     = "sample-app"
    }

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

Outputs:

api_session_token_output = <sensitive>
is_production = false
monitoring_enabled = true
primary_public_subnet = "subnet-0640d6a9b71ac5322"
resource_name = "lab_work-dev"
subnet_cidr_block_output = "10.0.20.0/24"
subnet_count = 3
tags = tomap({
  "Environment" = "dev"
  "Owner" = "platform-team"
  "Project" = "sample-app"
})
```

TASK 4: Maps and Objects

```
 GNU nano 7.2                                          main.tf
provider "aws" {
  shared_config_files      = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
}

variable "subnet_cidr_block" {
  type        = string
  default     = "10.0.20.0/24"
  description = "CIDR block to assign to the application subnet"
  sensitive   = false
  nullable    = false
  ephemeral   = false

  validation {
    condition     = can(regex("^([0-9]{1,3}\\.){3}[0-9]{1,3}/[0-9]+$", var.subnet_cidr_block))
    error_message = "The subnet_cidr_block must be a valid CIDR notation string, such as 10.0.0.0/24."
  }
}

output "subnet_cidr_block_output" {
  value = var.subnet_cidr_block
}
variable "api_session_token" {
  type        = string
  default     = "my_API_session_Token_12345"
  description = "Short@lived API session token used during apply operations"
  sensitive   = true
  nullable    = false
  ephemeral   = false

  validation {
    condition     = can(regex("^[A-Za-z0-9-_]{20,}$", var.api_session_token))
    error_message = "The API session token must be at least 20 characters and contain only letters, numbers, hyphens, o
  }
}

output "api_session_token_output" {
  value     = var.api_session_token
  sensitive = true
}
variable "environment" {}
variable "project_name" {}
variable "primary_subnet_id" {}
variable "subnet_count" {}
variable "monitoring" {}
output "resource_name" {
```

```
output "resource_name" {
  value = local.resource_name
}
variable "tags" {
  type = map(string)
}
variable "server_config" {
  type = object({
    name            = string
    instance_type   = string
    monitoring      = bool
    storage_gb      = number
    backup_enabled  = bool
  })
}

output "server_config" {
  value = var.server_config
}
output "tags" {
  value = var.tags
}
output "primary_public_subnet" {
  value = local.primary_public_subne
}

output "subnet_count" {
  value = local.subnet_count
}

output "is_production" {
  value = local.is_production
}

output "monitoring_enabled" {
  value = local.monitoring_enabled
```

```
  GNU nano 7.2                                          terraform.tfvars
environment = "dev"
project_name = "lab_work"
primary_subnet_id = "subnet-0640d6a9b71ac5322"
subnet_count = 3
monitoring = true
tags = {
  Environment = "dev"
  Project     = "sample-app"
  Owner       = "platform-team"
}
server_config = {
  name            = "web-server"
  instance_type   = "t3.micro"
  monitoring      = true
  storage_gb      = 20
  backup_enabled  = false
}
```

```
@SanaTariq205 /workspaces/cc_sanatariq_098 (main) $ terraform apply -auto-approve

Changes to Outputs:
  + server_config           = {
      + backup_enabled = false
      + instance_type  = "t3.micro"
      + monitoring     = true
      + name           = "web-server"
      + storage_gb     = 20
    }

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

Outputs:

api_session_token_output = <sensitive>
is_production = false
monitoring_enabled = true
primary_public_subnet = "subnet-0640d6a9b71ac5322"
resource_name = "lab_work-dev"
server_config = {
  "backup_enabled" = false
  "instance_type" = "t3.micro"
  "monitoring" = true
  "name" = "web-server"
  "storage_gb" = 20
}
subnet_cidr_block_output = "10.0.20.0/24"
subnet_count = 3
tags = tomap({
  "Environment" = "dev"
  "Owner" = "platform-team"
  "Project" = "sample-app"
})
```

TASK 5: Collections (List, Tuple, Set)

```
variable "server_config" {
  type = object({
    name            = string
    instance_type   = string
    monitoring      = bool
    storage_gb      = number
    backup_enabled  = bool
  })
}
variable "server_names" {
  type = list(string)
  default = ["web-2", "web-1", "web-2"]
}

variable "server_metadata" {
  type = tuple([string, number, bool])
  default = ["web-1", 4, true]
}

variable "availability_zones" {
  type = set(string)
  default = ["us-east-1b", "us-east-1a", "us-east-1b"]
}

output "compare_collections" {
  value = {
    list_example  = var.server_names
    tuple_example = var.server_metadata
    set_example   = var.availability_zones
  }
}
output "server_config" {
  value = var.server_config
}
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve

Changes to Outputs:
  + compare_collections      = {
      + list_example  = [
          + "web-2",
          + "web-1",
          + "web-2",
        ]
      + set_example   = [
          + "us-east-1a",
          + "us-east-1b",
        ]
      + tuple_example = [
          + "web-1",
          + 4,
          + true,
        ]
    }

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

```
Outputs:

api_session_token_output = <sensitive>
compare_collections = {
  "list_example" = tolist([
    "web-2",
    "web-1",
    "web-2",
  ])
  "set_example" = toset([
    "us-east-1a",
    "us-east-1b",
  ])
  "tuple_example" = [
    "web-1",
    4,
    true,
  ]
}
is_production = false
monitoring_enabled = true
primary_public_subnet = "subnet-0640d6a9b71ac5322"
resource_name = "lab_work-dev"
server_config = {
  "backup_enabled" = false
  "instance_type" = "t3.micro"
  "monitoring" = true
  "name" = "web-server"
  "storage_gb" = 20
}
subnet_cidr_block_output = "10.0.20.0/24"
subnet_count = 3
tags = tomap({
  "Environment" = "dev"
  "Owner" = "platform-team"
  "Project" = "sample-app"
})
```

```
  GNU nano 7.2                                          locals.tf
locals {
  resource_name = "${var.project_name}-${var.environment}"
  primary_public_subnet = var.primary_subnet_id
  subnet_count          = var.subnet_count
  is_production         = var.environment == "prod"
  monitoring_enabled    = var.monitoring || local.is_production
}
mutated_list  = setunion(var.server_names, ["web-3"])
  mutated_tuple = setunion(var.server_metadata, ["web-2"])
  mutated_set   = setunion(var.availability_zones, ["us-east-1c"])
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ cat > locals.tf << 'EOF'
locals {
  resource_name = "${var.project_name}-${var.environment}"
  primary_public_subnet = var.primary_subnet_id
  subnet_count          = var.subnet_count
  is_production         = var.environment == "prod"
  monitoring_enabled    = var.monitoring || local.is_production

  # These were outside the locals block - now fixed
  mutated_list  = setunion(var.server_names, ["web-3"])
  mutated_tuple = setunion(var.server_metadata, ["web-2"])
  mutated_set   = setunion(var.availability_zones, ["us-east-1c"])
}
EOF
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve

Changes to Outputs:
  + mutation_comparison      = {
      + mutated_tuple  = [
          + "4",
          + "true",
          + "web-1",
          + "web-2",
        ]
      + original_tuple = [
          + "web-1",
          + 4,
          + true,
        ]
    }

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

```
Outputs:

api_session_token_output = <sensitive>
compare_collections = {
  "list_example" = tolist([
    "web-2",
    "web-1",
    "web-2",
  ])
  "set_example" = toset([
    "us-east-1a",
    "us-east-1b",
  ])
  "tuple_example" = [
    "web-1",
    4,
    true,
  ]
}
is_production = false
monitoring_enabled = true
mutation_comparison = {
  "mutated_tuple" = toset([
    "4",
    "true",
    "web-1",
    "web-2",
  ])
  "original_tuple" = [
    "web-1",
    4,
    true,
  ]
}
```

TASK 6: Null & Any Type

```
primary_public_subnet = "subnet-0640d6a9b71ac5322"
resource_name = "lab_work-dev"
server_config = {
  "backup_enabled" = false
  "instance_type" = "t3.micro"
  "monitoring" = true
  "name" = "web-server"
  "storage_gb" = 20
}
subnet_cidr_block_output = "10.0.20.0/24"
subnet_count = 3
tags = tomap({
  "Environment" = "dev"
  "Owner" = "platform-team"
  "Project" = "sample-app"
})
```

```
  GNU nano 7.2                                                  locals.tf
locals {
  resource_name = "${var.project_name}-${var.environment}"
  primary_public_subnet = var.primary_subnet_id
  subnet_count          = var.subnet_count
  is_production         = var.environment == "prod"
  monitoring_enabled    = var.monitoring || local.is_production

  # These were outside the locals block - now fixed
  mutated_list  = setunion(var.server_names, ["web-3"])
  mutated_tuple = setunion(var.server_metadata, ["web-2"])
  mutated_set   = setunion(var.availability_zones, ["us-east-1c"])
}
server_tags = merge(
    { Name = "web-server" },
    var.optional_tag != null ? { Custom = var.optional_tag } : {}
  )
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve

Changes to Outputs:
  + optional_tag            = {
      + Name = "web-server"
    }

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ nano terraform.tfvars
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve

Changes to Outputs:
  ~ optional_tag            = {
      + Custom = "dev"
        # (1 unchanged attribute hidden)
    }

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

```
output "compare_collections" {
  value = {
    list_example  = var.server_names
    tuple_example = var.server_metadata
    set_example   = var.availability_zones
  }
}
variable "optional_tag" {
  type        = string
  description = "A tag that may or may not be provided"
  default     = null
}
variable "dynamic_value" {
  type        = any
  description = "A variable that can accept any data type"
  default     = null
}

output "value_received" {
  value = var.dynamic_value
}
output "server_config" {
  value = var.server_config
}
output "tags" {
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ nano main.tf
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ nano terraform.tfvars
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve

Changes to Outputs:
  + value_received           = "hello"

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ nano terraform.tfvars
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve

Changes to Outputs:
  ~ value_received           = "hello" -> 42

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ nano terraform.tfvars
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve

Changes to Outputs:
  ~ value_received           = 42 -> [
      + "a",
      + "b",
      + "c",
    ]

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ nano terraform.tfvars
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform apply -auto-approve

Changes to Outputs:
  ~ value_received           = [
      - "a",
      - "b",
      - "c",
    ] -> {
      + cpu  = 4
      + name = "server"
    }

You can apply this plan to save these new output values to the Terraform state, without changing any real
infrastructure.

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

TASK 7: Git Ignore

```
  GNU nano 7.2                                                .gitignore
.terraform/*
*.tfstate
*.tfstate.*
*.tfvars
*.pem
```

```
provider "aws" {
  shared_config_files      = ["~/.aws/config"]
  shared_credentials_files = ["~/.aws/credentials"]
}
variable "vpc_cidr_block" {}
variable "subnet_cidr_block" {}
variable "availability_zone" {}
variable "env_prefix" {}

resource "aws_vpc" "myapp_vpc" {
  cidr_block = var.vpc_cidr_block
  tags = {
    Name = "${var.env_prefix}-vpc"
  }
}
resource "aws_subnet" "myapp_subnet_1" {
  vpc_id            = aws_vpc.myapp_vpc.id
  cidr_block        = var.subnet_cidr_block
  availability_zone = var.availability_zone
  tags = {
    Name = "${var.env_prefix}-subnet-1"
  }
}
```

```
  GNU nano 7.2                                        terraform.tfvars *
vpc_cidr_block      = "10.0.0.0/16"
subnet_cidr_block   = "10.0.10.0/24"
availability_zone   = "us-east-1a"
env_prefix          = "dev"
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v6.27.0

Terraform has been successfully initialized!
```

prod-vpc	vpc-064f9b1c7aff6e7b9	Available	–
dev-vpc	vpc-0ab502d1a9c554adc	Available	–
prod-vpc	vpc-09c8d1c64aa864608	Available	–

TASK 8: Clean Up & Build Real Infrastructure

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

resource "aws_route_table" "myapp_route_table" {
  vpc_id = aws_vpc.myapp_vpc.id

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

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

```
aws_vpc.myapp_vpc: Creating...
aws_vpc.myapp_vpc: Creation complete after 4s [id=vpc-0d27b1
aws_internet_gateway.myapp_igw: Creating...
aws_subnet.myapp_subnet_1: Creating...
aws_internet_gateway.myapp_igw: Creation complete after 1s [
aws_route_table.myapp_route_table: Creating...
aws_subnet.myapp_subnet_1: Creation complete after 1s [id=su
aws_route_table.myapp_route_table: Creation complete after 3

Apply complete! Resources: 4 added, 0 changed, 0 destroyed.
```

```
resource "aws_route_table_association" "a_rtb_subnet" {
  subnet_id      = aws_subnet.myapp_subnet_1.id
  route_table_id = aws_route_table.myapp_route_table.id
}
```

```
Plan: 1 to add, 0 to change, 0 to destroy.
aws_route_table_association.a_rtb_subnet: Creating...
aws_route_table_association.a_rtb_subnet: Creation complete a

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

```
resource "aws_internet_gateway" "myapp_igw" {
  vpc_id = aws_vpc.myapp_vpc.id
  tags = {
    Name = "${var.env_prefix}-igw"
  }
}
resource "aws_default_route_table" "main_rt" {
  default_route_table_id = aws_vpc.myapp_vpc.default_route_table_id

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

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

```
Plan: 1 to add, 0 to change, 2 to destroy.
aws_route_table_association.a_rtb_subnet: Destroying... [id=rtbassoc-00f4e0ed5250d2b03]
aws_default_route_table.main_rt: Creating...
aws_route_table_association.a_rtb_subnet: Destruction complete after 1s
aws_route_table.myapp_route_table: Destroying... [id=rtb-02a800bb1d87b7667]
aws_route_table.myapp_route_table: Destruction complete after 2s
aws_default_route_table.main_rt: Creation complete after 3s [id=rtb-02fc47fd7782b5b3d]

Apply complete! Resources: 1 added, 0 changed, 2 destroyed.
```

```
variable "vpc_cidr_block" {}
variable "subnet_cidr_block" {}
variable "availability_zone" {}
variable "env_prefix" {}
variable "my_ip" {}
variable "instance_type" {}
```

```
  GNU nano 7.2                                        terraform.tfvars *
vpc_cidr_block     = "10.0.0.0/16"
subnet_cidr_block  = "10.0.10.0/24"
availability_zone  = "us-east-1a"
env_prefix         = "dev"
my_ip = "4.240.39.192/32"
instance_type = "t3.micro"
```

```
resource "aws_default_security_group" "default_sg" {
  vpc_id = aws_vpc.myapp_vpc.id

  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = [var.my_ip]
  }

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

  egress {
    from_port       = 0
    to_port         = 0
    protocol        = "-1"
    cidr_blocks     = ["0.0.0.0/0"]
    prefix_list_ids = []
  }

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

```
Plan: 1 to add, 0 to change, 0 to destroy.
aws_default_security_group.default_sg: Creating...
aws_default_security_group.default_sg: Creation complete after 4s [id=sg-0df6a696ac2f452f3]

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

TASK 9: Security Group, Key Pair, EC2

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ aws ec2 create-key-pair \
>   --key-name MyED25519Key \
>   --key-type ed25519 \
>   --key-format pem \
>   --query 'KeyMaterial' \
>   --output text > MyED25519Key.pem
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ chmod 600 MyED25519Key.pem
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ cat .gitignore | grep pem
*.pem
```

```
resource "aws_instance" "myapp-server" {
  ami                         = "ami-05524d6658fcf35b6"
  instance_type               = var.instance_type
  subnet_id                   = aws_subnet.myapp_subnet_1.id
  vpc_security_group_ids      = [aws_default_security_group.default_sg.id]
  availability_zone           = var.availability_zone
  associate_public_ip_address = true
  key_name                    = "MyED25519Key"

  tags = {
    Name = "${var.env_prefix}-ec2-instance"
  }
}

output "aws_instance_public_ip" {
  value = aws_instance.myapp-server.public_ip
}
```

```
Changes to Outputs:
  + aws_instance_public_ip = (known after apply)
aws_subnet.myapp_subnet_1: Destroying... [id=subnet-0306292fae497fe3c]
aws_subnet.myapp_subnet_1: Destruction complete after 1s
aws_vpc.myapp_vpc: Modifying... [id=vpc-0d27b10b6441529b7]
aws_vpc.myapp_vpc: Modifications complete after 2s [id=vpc-0d27b10b6441529b7]
aws_internet_gateway.myapp_igw: Modifying... [id=igw-0f261a67bda9f3d51]
aws_subnet.myapp_subnet_1: Creating...
aws_default_security_group.default_sg: Modifying... [id=sg-0df6a696ac2f452f3]
aws_internet_gateway.myapp_igw: Modifications complete after 1s [id=igw-0f261a67bda9f3d51]
aws_default_route_table.main_rt: Modifying... [id=rtb-02fc47fd7782b5b3d]
aws_subnet.myapp_subnet_1: Creation complete after 2s [id=subnet-0de0097d4c8a66dcd]
aws_default_security_group.default_sg: Modifications complete after 2s [id=sg-0df6a696ac2f452f3]
aws_instance.myapp-server: Creating...
aws_default_route_table.main_rt: Modifications complete after 0s [id=rtb-02fc47fd7782b5b3d]
aws_instance.myapp-server: Still creating... [00m10s elapsed]
aws_instance.myapp-server: Still creating... [00m20s elapsed]
aws_instance.myapp-server: Still creating... [00m30s elapsed]
aws_instance.myapp-server: Creation complete after 35s [id=i-07bd150cf45f4b88f]

Apply complete! Resources: 2 added, 4 changed, 1 destroyed.

Outputs:

aws_instance_public_ip = "100.48.210.62"
```

```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-10-0-20-165 ~]$
```

```
@SanaTariq205 /workspaces/cc_sanatariq_058 (main) $ ssh-keygen
Generating public/private ed25519 key pair.
Your identification has been saved in /home/codespace/.ssh/id_ed2
Your public key has been saved in /home/codespace/.ssh/id_ed25519
The key fingerprint is:
SHA256:FTMZYPheZr8d5kAsQlL9RwX1OvAUCXGcGvnjDyMBFAI codespace@code
The key's randomart image is:
+--[ED25519 256]--+
|      E+=+B+o=+*o|
|      o...++o.=o |
|       +  .+.=. .|
|        o.= *++. |
|       .S= + ++. |
|        .   + *. |
|            B = |
|           . o .|
|                 |
+----[SHA256]-----+
```

```
resource "aws_key_pair" "ssh_key" {
  key_name   = "serverkey"
  public_key = file("~/.ssh/id_ed25519.pub")
}
resource "aws_instance" "myapp-server" {
  ami                         = data.aws_ami.amazon_linux.id
  instance_type               = var.instance_type
  subnet_id                   = aws_subnet.myapp_subnet_1.id
  vpc_security_group_ids      = [aws_default_security_group.de
  availability_zone           = var.availability_zone
  associate_public_ip_address = true
  key_name                    = aws_key_pair.ssh_key.key_name

  tags = {
    Name = "${var.env_prefix}-ec2-instance"
  }
}
```

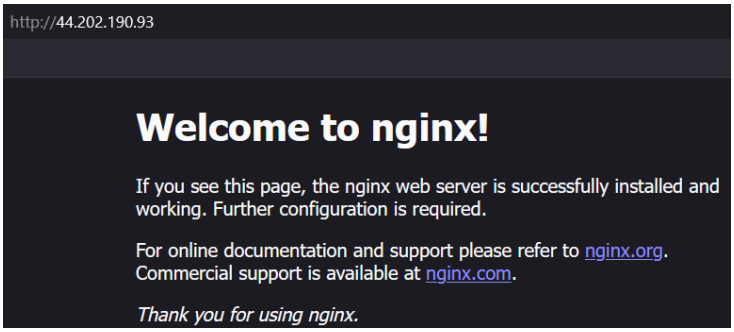
```
Apply complete! Resources: 2 added, 0 changed, 1 destroyed.

Outputs:

aws_instance_public_ip = "3.234.178.183"
```

```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/li
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-10-0-20-77 ~]$
```

```
resource "aws_instance" "myapp-server" {
  ami                         = data.aws_ami.amazon_linux.id  # Use data so
  instance_type               = var.instance_type
  subnet_id                   = aws_subnet.myapp_subnet_1.id
  vpc_security_group_ids      = [aws_default_security_group.default_sg.id]
  availability_zone           = var.availability_zone
  associate_public_ip_address = true
  key_name                    = aws_key_pair.ssh_key.key_name
  user_data = <<-EOF
              #!/bin/bash
              yum update -y
              yum install -y nginx
              systemctl start nginx
              systemctl enable nginx
            EOF
  tags = {
    Name = "${var.env_prefix}-ec2-instance"
```

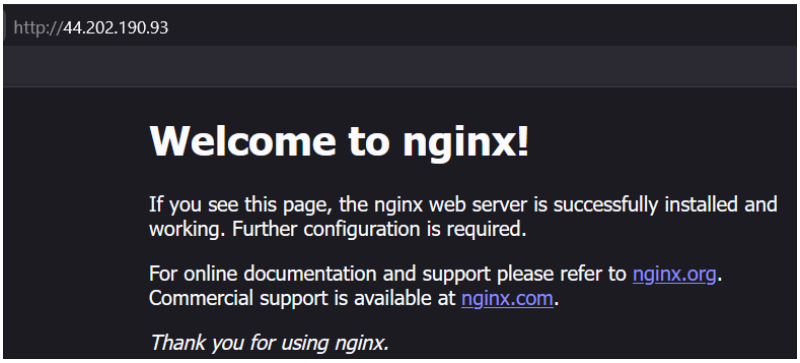
```
aws_instance.myapp-server: Modifying... [id=i-02b1f1af1044ea7cf]
aws_instance.myapp-server: Still modifying... [id=i-02b1f1af1044ea7cf, 00m10s elapsed]
aws_instance.myapp-server: Still modifying... [id=i-02b1f1af1044ea7cf, 00m20s elapsed]
aws_instance.myapp-server: Still modifying... [id=i-02b1f1af1044ea7cf, 00m30s elapsed]
aws_instance.myapp-server: Still modifying... [id=i-02b1f1af1044ea7cf, 00m40s elapsed]
aws_instance.myapp-server: Still modifying... [id=i-02b1f1af1044ea7cf, 00m50s elapsed]
aws_instance.myapp-server: Still modifying... [id=i-02b1f1af1044ea7cf, 01m00s elapsed]
aws_instance.myapp-server: Modifications complete after 1m6s [id=i-02b1f1af1044ea7cf]

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

Outputs:
```

```
[ec2-user@ip-10-0-20-77 ~]$ sudo systemctl start nginx
[ec2-user@ip-10-0-20-77 ~]$ curl localhost
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
[ec2-user@ip-10-0-20-77 ~]$
```




```
resource "aws_instance" "myapp-server" {
  ami                         = data.aws_ami.amazon_linux.id  # Use data s
  instance_type               = var.instance_type
  subnet_id                   = aws_subnet.myapp_subnet_1.id
  vpc_security_group_ids      = [aws_default_security_group.default_sg.id]
  availability_zone           = var.availability_zone
  associate_public_ip_address = true
  key_name                    = aws_key_pair.ssh_key.key_name
  user_data = file("entry-script.sh")
  tags = {
    Name = "${var.env_prefix}-ec2-instance"
  }
}
```

```
No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your c
needed.

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

Outputs:
```


http://44.202.190.93
Welcome to nginx!
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.
Thank you for using nginx.


Destroy complete! Resources: 7 destroyed.