

Bootcamp-Project 3

AUTOMATE INFRASTRUCTURE PROVISIONING USING
TERRAFORM AND AZURE DEVOPS WITH MULTIPLE
WORKSPACES

Mohanramrajan Erran Bothalraj

Table of Contents

Introduction	2
Project Objectives	2
Expected Outcome	2
Pre-requisite:.....	2
Architecture Diagram	3
Solution Steps	3
A. Terraform State Management	3
B. Terraform Scripting	4
C. Terraform Three-Tier Architecture	4
D. Azure DevOps Pipeline	5
Validation	11
APPENDIX	13
Main.tf	13
Variables	24
TFVARS file	25

Introduction

In today's cloud-first development landscape, automating infrastructure provisioning is essential for delivering scalable, secure, and repeatable environments. This project demonstrates how to implement Infrastructure as Code (IaC) using **Terraform** integrated with **Azure DevOps**, enabling consistent and automated infrastructure deployment across multiple environments.

Project Objectives

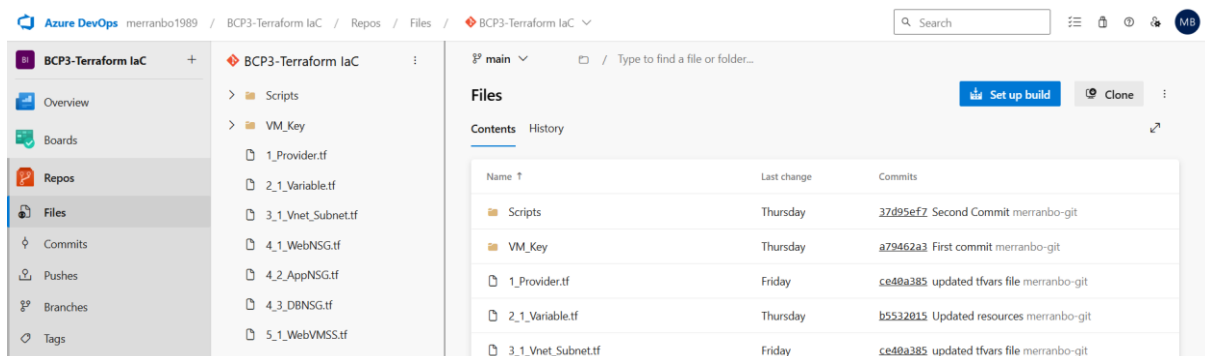
- **Three-Tier Architecture:** Deploy a secure and highly available three-tier architecture (Web, App, DB) across **Dev**, **Stage**, and **Production** environments using a single configuration file and separate workspaces.
- **Terraform Scripting:** Define and provision Azure resources such as Virtual Machines, Virtual Networks, Storage Accounts, Load Balancers, NSGs, and Scale Sets using reusable and modular Terraform scripts.
- **Terraform State Management:** Securely store Terraform state files in Azure Storage with proper **locking strategies** to ensure consistency in team environments.
- **Azure DevOps Pipelines:** Automate the validation, planning, and applying of Terraform configurations.
- **Scalability and Security:** Implement network isolation using subnets, firewalls, and NSGs, while ensuring high availability with VM Scale Sets and Load Balancers.

Expected Outcome

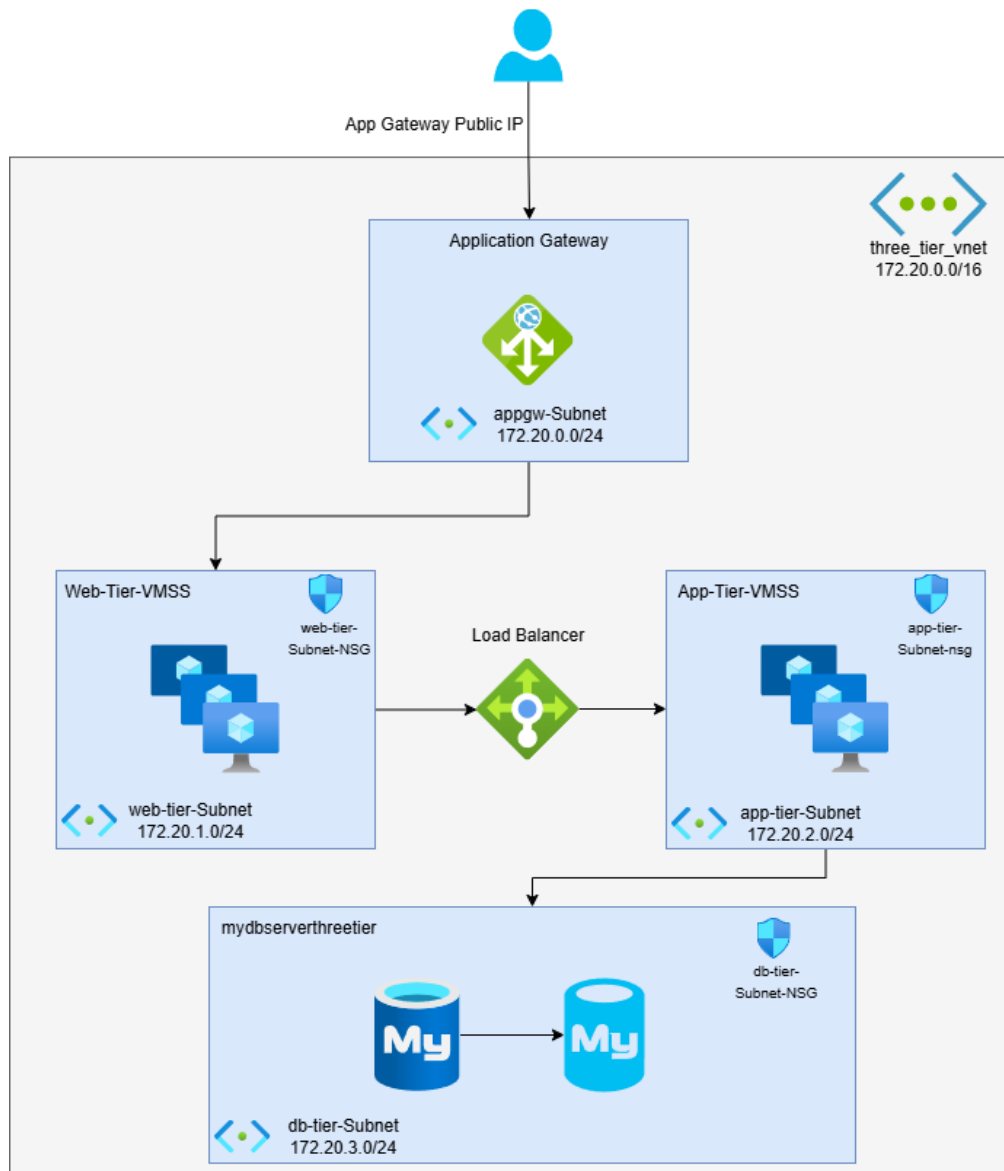
This project enables streamlined, automated infrastructure deployments using modern DevOps practices. By integrating Terraform with Azure DevOps pipelines, the solution promotes scalability, maintainability, and governance while reducing manual intervention and provisioning errors across development lifecycles.

Pre-requisite:

- GitHub Repository for terraform template file: <https://github.com/merranbo-git/bcpp3-terraform>
- Import the terraform code into Azure Repos after creating the project



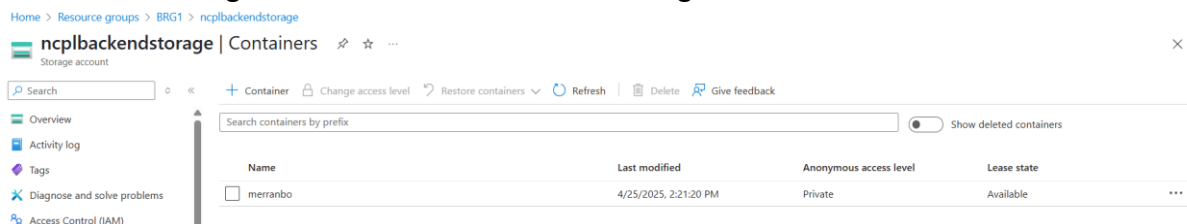
Architecture Diagram



Solution Steps

A. Terraform State Management

Creation of Storage Account → container for storing “terraform.tfstate” file.

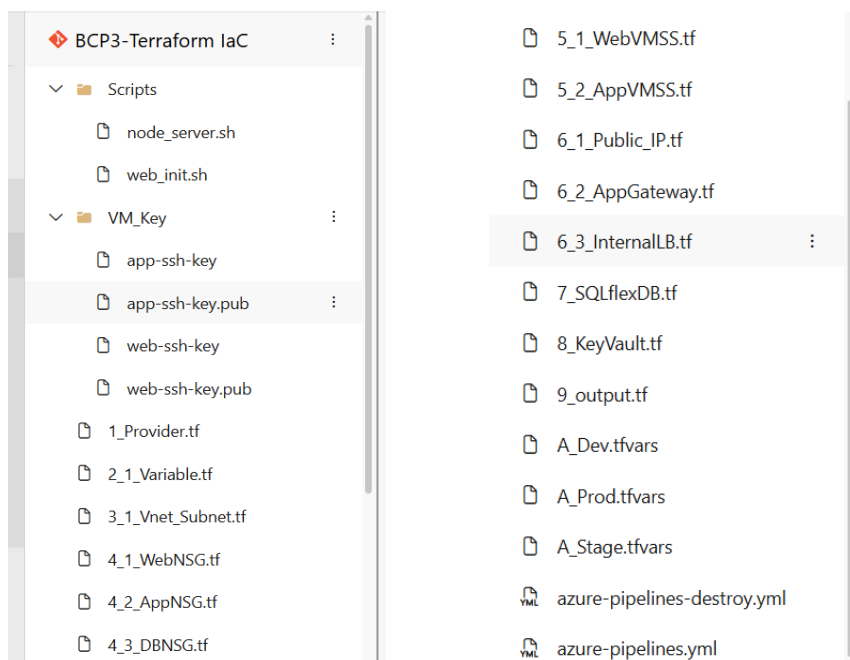


B. Terraform Scripting

1. Develop terraform scripting using Visual studio code to provision the resources required for the three-tier architecture. Following are the resources to be provisioned:
 - a. Resource Group
 - b. Vnet and Subnets
 - c. Network Security groups
 - d. Virtual Machine Scale Sets (for Web tier and App tier)
 - i. Deploy custom script using provisioner in both VMSS
 - e. Application Gateway
 - f. Public Ip
 - g. Load Balancers for internal communication between VMSS
 - h. MySQL flexible server with database
 - i. Key Vault for storing secrets

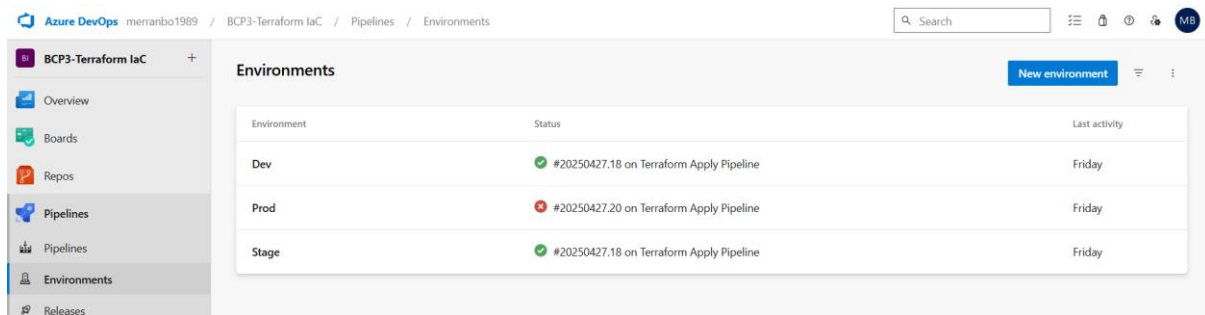
Please Note: Terraform codes for above resources are available in the Appendix Section

2. Below is the file structure for the terraform project.



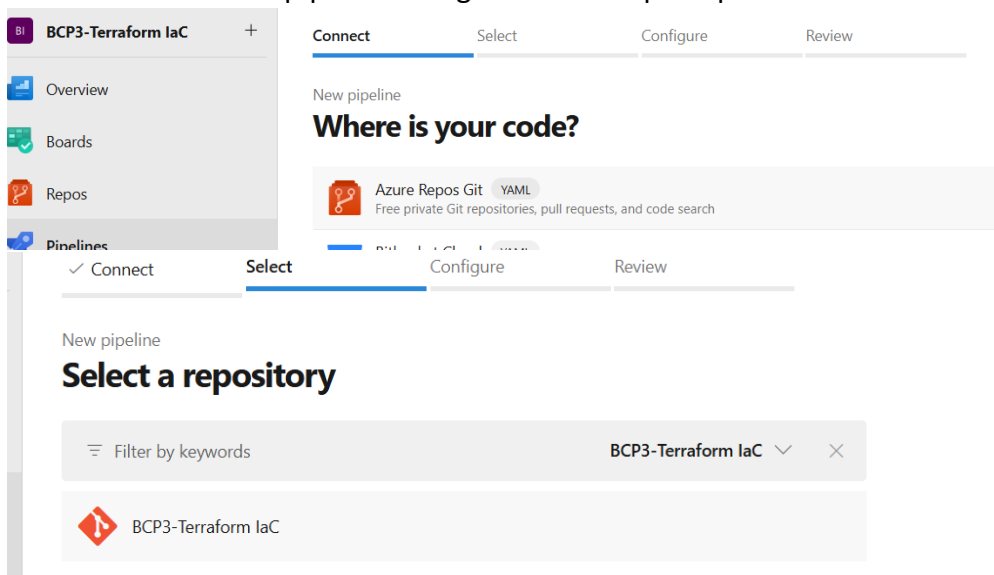
C. Terraform Three-Tier Architecture

1. As per the project requirement, we need to deploy a secure and highly available three-tier architecture (Web, App, DB) across **Dev**, **Stage**, and **Production** environments using a single configuration file and separate workspaces.
2. To Achieve this, we will be creating three Environments in Azure DevOps pipeline

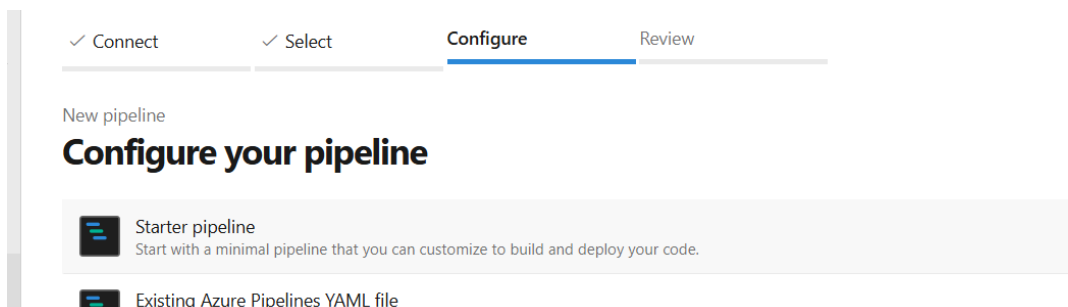


D. Azure DevOps Pipeline

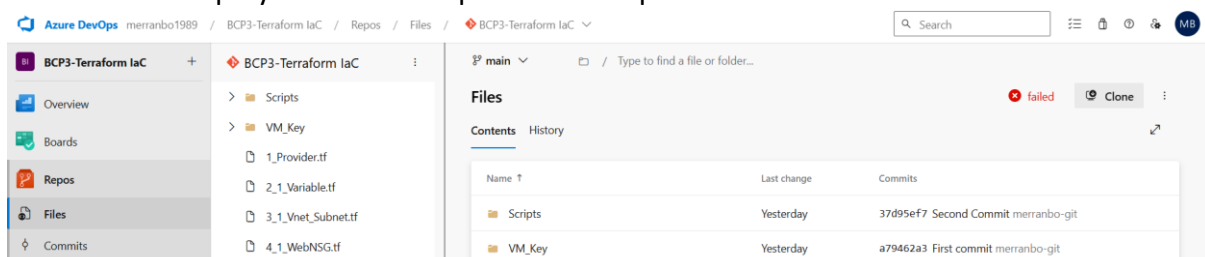
1. Create a new pipeline using the Azure repos uploaded with the terraform files

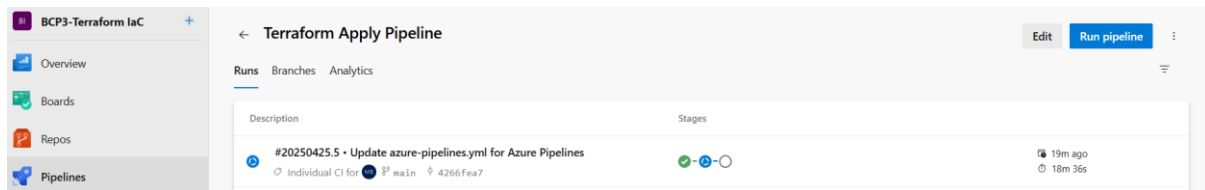


2. Use the “Starter Pipeline” option to get an empty YAML file

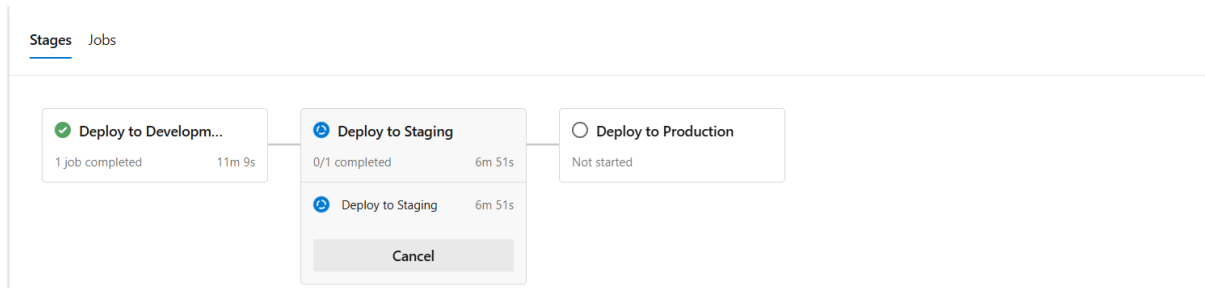


3. Configure the YAML file with separate stages for “Dev”, “Stage”, “Prod” environments to be deployed in their respective workspace.





4. Trigger the pipeline after configuring the YAML file.



***** Code starts here *****

Starter pipeline

Start with a minimal pipeline that you can customize to build and deploy your code.

Add steps that build, run tests, deploy, and more:

<https://aka.ms/yaml>

trigger: none

pool: NewAgent

variables:

bkstrgname : 'BRG1'

bkstrgacc : 'ncplbackendstorage'

bkcontainer : 'merranbo'

#bkfile : '2_2_Values.tfvars'

stages:

- stage: Dev

displayName: Deploy to Development

variables:

env: 'Dev'

bkkey : 'dev.terraform.tfstate'

varfile: 'A_Dev.tfvars'

jobs:

- deployment: DeployDev

displayName: Deploy to Dev

environment: 'Dev'

continueOnError: false

strategy:

runOnce:

```

deploy:
  steps:
    - checkout: self
    - task: TerraformInstaller@1
      displayName: Terraform Install
      inputs:
        terraformVersion: 'latest'
    - task: TerraformTaskV4@4
      displayName: Terraform Init
      inputs:
        provider: 'azurerm'
        command: 'init'
        backendServiceArm: 'Azure subscription 1(3f48112c-a56d-44e0-9a80-1ed5fbce3aac)'
        backendAzureRmResourceGroupName: '$(bkstrgname)'
        backendAzureRmStorageAccountName: '$(bkstrgacc)'
        backendAzureRmContainerName: '$(bkcontainer)'
        backendAzureRmKey: '$(bkkey)'
    - task: TerraformTaskV4@4
      displayName: Terraform Validate
      inputs:
        provider: 'azurerm'
        command: 'validate'
    - task: Bash@3
      displayName: Workspace Creation
      inputs:
        targetType: 'inline'
        script: 'terraform workspace new $(env)'
    - task: TerraformTaskV4@4
      displayName: Terraform Plan
      inputs:
        provider: 'azurerm'
        command: 'plan'
        commandOptions: '-var-file="$(varfile)"'
        environmentServiceNameAzureRM: 'Azure subscription 1(3f48112c-a56d-44e0-9a80-1ed5fbce3aac)'
    - task: TerraformTaskV4@4
      displayName: Terraform Apply
      inputs:
        provider: 'azurerm'
        command: 'apply'
        commandOptions: '-var-file="$(varfile)"'
        environmentServiceNameAzureRM: 'Azure subscription 1(3f48112c-a56d-44e0-9a80-1ed5fbce3aac)'

```


- stage: Stage
 - displayName: Deploy to Staging
 - condition: succeeded ('Dev')
 - dependsOn: Dev
 - variables:
 - env: 'Stage'
 - bkkey : 'stage.terraform.tfstate'
 - varfile: 'A_Stage.tfvars'
 - jobs:
 - deployment: DeployStage
 - displayName: Deploy to Staging
 - environment: 'Stage'
 - continueOnError: false
 - strategy:
 - runOnce:
 - deploy:
 - steps:
 - checkout: self
 - task: TerraformInstaller@1
 - displayName: Terraform Install
 - inputs:
 - terraformVersion: 'latest'
 - task: TerraformTaskV4@4
 - displayName: Terraform Init
 - inputs:
 - provider: 'azurerm'
 - command: 'init'
 - backendServiceArm: 'Azure subscription 1(3f48112c-a56d-44e0-9a80-1ed5fbce3aac)'
 - backendAzureRmResourceGroupName: '\$(bkstrgname)'
 - backendAzureRmStorageAccountName: '\$(bkstrgacc)'
 - backendAzureRmContainerName: '\$(bkcontainer)'
 - backendAzureRmKey: '\$(bkkey)'
 - task: TerraformTaskV4@4
 - displayName: Terraform Validate
 - inputs:
 - provider: 'azurerm'
 - command: 'validate'
 - task: Bash@3
 - displayName: Workspace Creation
 - inputs:
 - targetType: 'inline'
 - script: 'terraform workspace new \$(env)'

- task: TerraformTaskV4@4
 - displayName: Terraform Plan
 - inputs:
 - provider: 'azurerm'
 - command: 'plan'
 - commandOptions: '-var-file="\$(varfile)'"
 - environmentServiceNameAzureRM: 'Azure subscription 1(3f48112c-a56d-44e0-9a80-1ed5fbce3aac)'
- task: TerraformTaskV4@4
 - displayName: Terraform Apply
 - inputs:
 - provider: 'azurerm'
 - command: 'apply'
 - commandOptions: '-var-file="\$(varfile)'"
 - environmentServiceNameAzureRM: 'Azure subscription 1(3f48112c-a56d-44e0-9a80-1ed5fbce3aac)'

- stage: Prod

- displayName: Deploy to Production
- condition: succeeded ('Stage')
- dependsOn: Stage
- variables:
 - env: 'Prod'
 - bkkey : 'prod.terraform.tfstate'
 - varfile: 'A_Prod.tfvars'

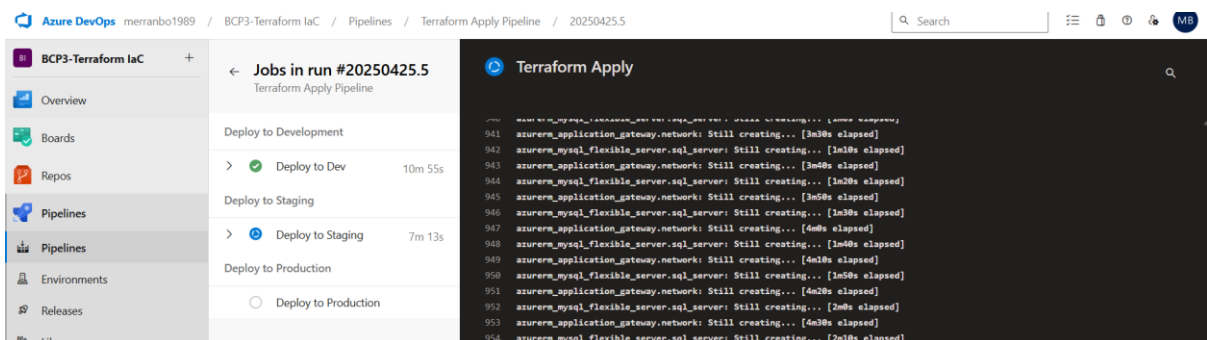
jobs:

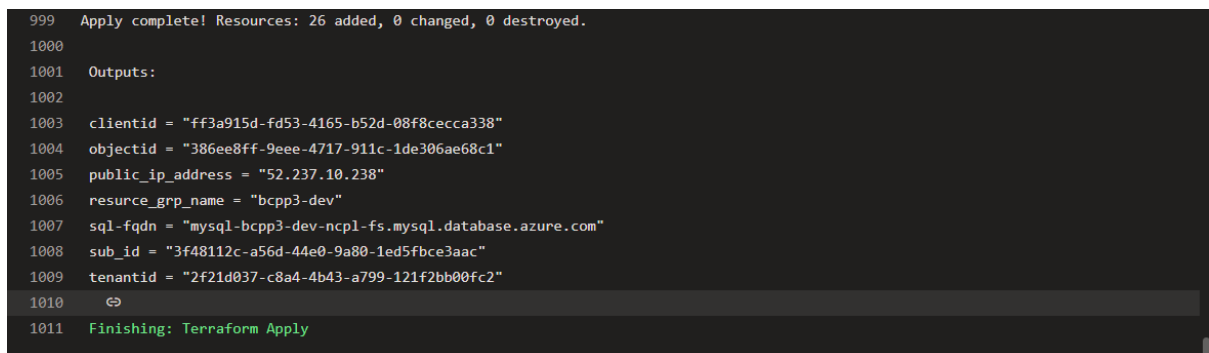
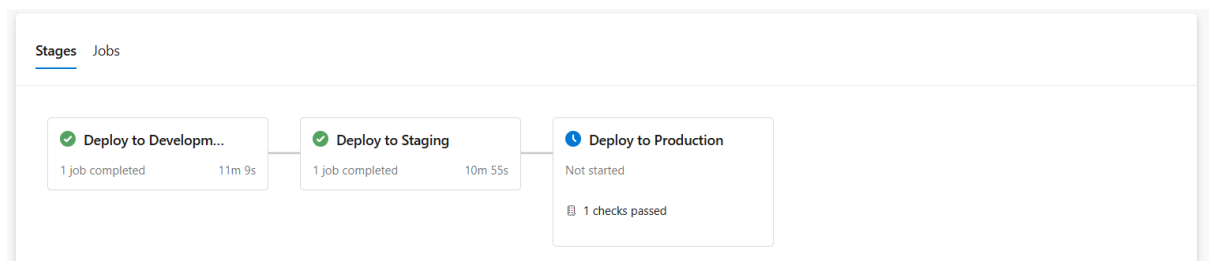
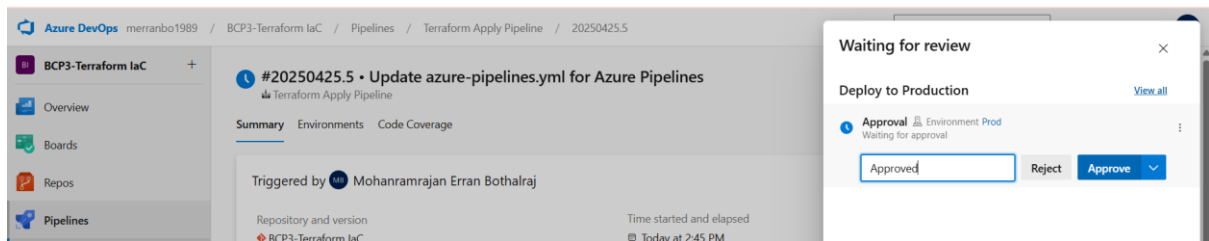
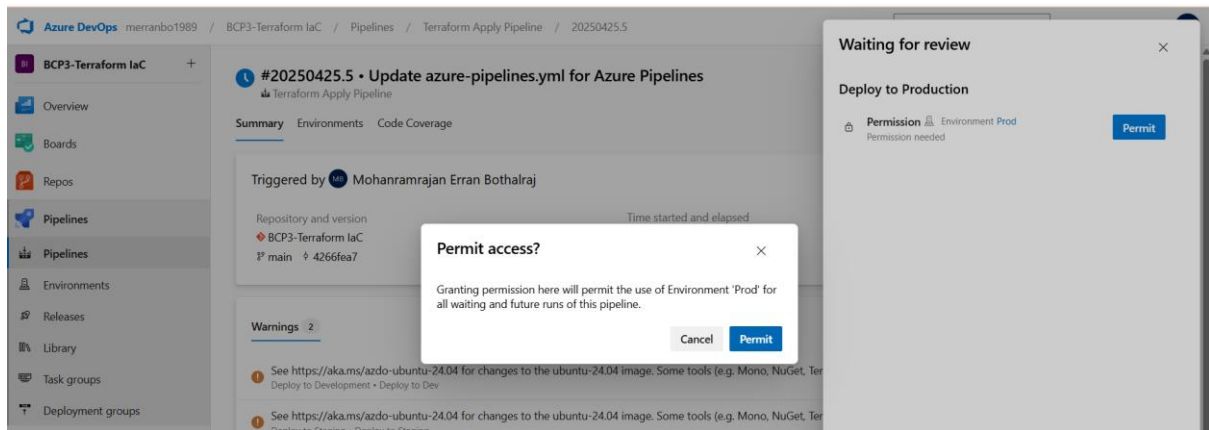
- deployment: DeployProd
 - displayName: Deploy to Production
 - environment: 'Prod'
 - continueOnError: false
 - strategy:
 - runOnce:
 - deploy:
 - steps:
 - checkout: self
 - task: TerraformInstaller@1
 - displayName: Terraform Install
 - inputs:
 - terraformVersion: 'latest'
 - task: TerraformTaskV4@4
 - displayName: Terraform Init
 - inputs:
 - provider: 'azurerm'
 - command: 'init'

```

    backendServiceArm: 'Azure subscription 1(3f48112c-a56d-44e0-9a80-1ed5fbce3aac)'
    backendAzureRmResourceGroupName: '${bkstrgname}'
    backendAzureRmStorageAccountName: '${bkstrgacc}'
    backendAzureRmContainerName: '${bkcontainer}'
    backendAzureRmKey: '${bkkey}'
- task: TerraformTaskV4@4
  displayName: Terraform Validate
  inputs:
    provider: 'azurerm'
    command: 'validate'
- task: Bash@3
  displayName: Workspace Creation
  inputs:
    targetType: 'inline'
    script: 'terraform workspace new $(env)'
- task: TerraformTaskV4@4
  displayName: Terraform Plan
  inputs:
    provider: 'azurerm'
    command: 'plan'
    commandOptions: '-var-file="$(varfile)'"
    environmentServiceNameAzureRM: 'Azure subscription 1(3f48112c-a56d-44e0-9a80-1ed5fbce3aac)'
- task: TerraformTaskV4@4
  displayName: Terraform Apply
  inputs:
    provider: 'azurerm'
    command: 'apply'
    commandOptions: '-var-file="$(varfile)'"
    environmentServiceNameAzureRM: 'Azure subscription 1(3f48112c-a56d-44e0-9a80-1ed5fbce3aac)'
***** Code Ends here *****

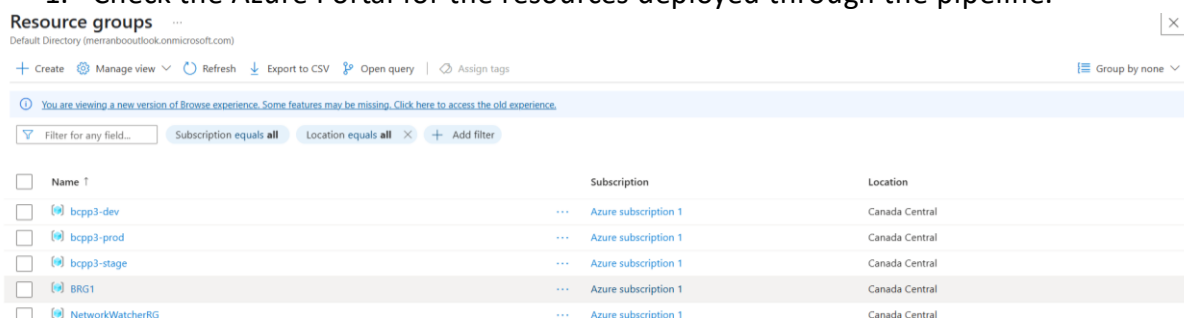
```





Validation

1. Check the Azure Portal for the resources deployed through the pipeline.



2. Check if the Environment workspace is added in the tags section

The screenshot shows the Azure portal interface for a resource named 'app-vmss' (Virtual machine scale set). The left sidebar contains navigation links: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Instances, Resource visualizer, Networking, Settings, and Availability + scale. The main content area shows the 'Overview' tab with an 'Essentials' section. Under 'Essentials', the 'Tags' section is expanded, displaying a single tag: 'environment: Dev'. Other details visible include the resource group 'bcpp3-dev', location 'Canada Central', and subscription ID '3f48112c-a56d-44e0-9a80-1ed5fbc3aac'.

The screenshot shows the Azure portal interface for a resource group named 'bcpp3-dev'. The left sidebar contains navigation links: Overview, Activity log, Access control (IAM), Tags, Resource visualizer, Events, Settings, Cost Management, Monitoring, Automation, and Help. The main content area shows the 'Overview' tab with an 'Essentials' section. Under 'Essentials', the 'Resources' section is expanded, displaying a list of resources. The resources listed are: app-nsg (Network security group), app-vmss (Virtual machine scale set), appgw-public-ip (Public IP address), bcpp3-appgateway (Application gateway), bcpp3-dev-kv (Key vault), db-nsg (Network security group), internal-loadbalancer (Load balancer), and mysql-bcpp3-dev-ncpl-fs (Azure Database for MySQL flexible server). All resources are located in 'Canada Central'.

3. Create a Destroy pipeline to remove the resources created through the pipeline

The screenshot shows the Azure Pipelines interface. The top bar has a 'New pipeline' button. Below it, the 'Pipelines' tab is selected, showing a list of pipelines. The 'Recently run pipelines' section lists two runs of the 'Terraform Destroy Pipeline'. The first run is #20250425.2, which was manually triggered for the 'main' branch and completed 6m ago. The second run is #20250427.4, which was also manually triggered for the 'main' branch and completed 9h ago. The 'Runs' tab is also visible, showing a detailed view of a pipeline run. The 'Runs' tab has sub-tabs for 'Runs', 'Branches', and 'Analytics'. The 'Runs' sub-tab is selected, showing a list of runs. The first run is #20250427.4, which was manually triggered for the 'main' branch and completed 9h ago. The run status is shown as a green checkmark, indicating success.

Summary Environments Code Coverage

Manually run by Mohanramrajan Erran Bothalraj [View 49 changes](#)

| Repository and version | Time started and elapsed | Related | Tests and coverage |
|-------------------------------------|-----------------------------|-----------------------------|-----------------------------|
| BCP3-Terraform IaC
main a6d5ae65 | Today at 2:34 PM
43m 45s | 0 work items
0 artifacts | Get started |

Stages Jobs

Destroy Resources fr...
1 job completed 15m 6s

Destroy resources fro...
1 job completed 14m 57s

Destroy resources fro...
1 job completed 12m 27s
1 checks passed

Azure DevOps merranbo1989 / BCP3-Terraform IaC / Pipelines / Terraform Destroy Pipeline / 20250427.4

BCP3-Terraform IaC Overview Boards Repos Pipelines Pipelines Environments Releases

Jobs in run #20250427.4
Terraform Destroy Pipeline

Destroy Resources from Development

- Destroy Dev Resources 15m 3s

Destroy resources from Staging

- Destroy Staging Res... 14m 54s

Destroy resources from Production

- Destroy Production ... 12m 24s

Destroy Production Resources

View raw log

```

1 Pool: HostAgent
2 Agent: agent001
3 Started: Today at 3:06 PM
4 Duration: 12m 24s
  
```

APPENDIX

Main.tf

```

terraform {
  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "=3.0.0"
    }
  }
  backend "azurerm" {}
}

provider "azurerm" {
  subscription_id = "3f48112c-a56d-44e0-9a80-1ed5fbce3aac"
  tenant_id       = "2f21d037-c8a4-4b43-a799-121f2bb00fc2"
  client_id       = "ff3a915d-fd53-4165-b52d-08f8cecca338"
  client_secret   = "IEe8Q~0oB2SA5fdjcQuIieqFLWp6b~Ff9X01gapW"

  features {
    key_vault {
      purge_soft_delete_on_destroy    = true
      recover_soft_deleted_key_vaults = true
    }
  }
}

data "azurerm_client_config" "current" {}
  
```

```

resource "azurerm_resource_group" "res_grp" {
  name      = var.res_grp_name
  location  = var.location
}
resource "azurerm_virtual_network" "myvnet" {
  name                = var.vnet_name
  address_space       = ["170.20.0.0/16"]
  location            = var.location
  resource_group_name = var.res_grp_name
  depends_on = [ azurerm_resource_group.res_grp ]
}
resource "azurerm_subnet" "appgateway_subnet" {
  name                = var.vnet_appgw
  resource_group_name = var.res_grp_name
  virtual_network_name = azurerm_virtual_network.myvnet.name
  address_prefixes     = ["170.20.0.0/24"]
  depends_on = [ azurerm_resource_group.res_grp ]
}
resource "azurerm_subnet" "web_subnet" {
  name                = var.vnet_web
  resource_group_name = var.res_grp_name
  virtual_network_name = azurerm_virtual_network.myvnet.name
  address_prefixes     = ["170.20.1.0/24"]
  depends_on = [ azurerm_resource_group.res_grp ]
}
resource "azurerm_subnet" "app_subnet" {
  name                = var.vnet_app
  resource_group_name = var.res_grp_name
  virtual_network_name = azurerm_virtual_network.myvnet.name
  address_prefixes     = ["170.20.2.0/24"]
  depends_on = [ azurerm_resource_group.res_grp ]
}
resource "azurerm_subnet" "db_subnet" {
  name                = var.vnet_db
  resource_group_name = var.res_grp_name
  virtual_network_name = azurerm_virtual_network.myvnet.name
  address_prefixes     = ["170.20.3.0/24"]

  service_endpoints = ["Microsoft.Storage"]
  delegation {
    name = "fs"
    service_delegation {
      name = "Microsoft.DBforMySQL/flexibleServers"
      actions = [
        "Microsoft.Network/virtualNetworks/subnets/join/action",
      ]
    }
  }
}
depends_on = [ azurerm_resource_group.res_grp ]
}
resource "azurerm_network_security_group" "webnsg" {
  name                = "web-nsg"
  location            = var.location
  resource_group_name = var.res_grp_name

  security_rule {
    name                = "AllowSSH"
    priority            = 1001
  }
}

```

```

        direction          = "Inbound"
        access              = "Allow"
        protocol            = "Tcp"
        source_port_range   = "*"
        destination_port_range = "22"
        source_address_prefix = "*"
        destination_address_prefix = "*"
    }

    security_rule {
        name          = "AllowHTTP"
        priority       = 1002
        direction      = "Inbound"
        access         = "Allow"
        protocol       = "Tcp"
        source_port_range = "*"
        destination_port_range = "80"
        source_address_prefix = "*"
        destination_address_prefix = "*"
    }
    security_rule {
        name          = "AllowNodeJS"
        priority       = 104
        direction      = "Inbound"
        access         = "Allow"
        protocol       = "Tcp"
        source_port_range = "*"
        destination_port_range = "5000"
        source_address_prefix = "*"
        destination_address_prefix = "*"
    }
    depends_on = [ azurerm_resource_group.res_grp ]

}

resource "azurerm_subnet_network_security_group_association" "webnsgconn" {
    subnet_id = azurerm_subnet.web_subnet.id
    network_security_group_id = azurerm_network_security_group.webnsg.id
    depends_on = [ azurerm_network_security_group.webnsg ]
}

resource "azurerm_network_security_group" "appnsg" {
    name          = "app-nsg"
    location      = var.location
    resource_group_name = var.res_grp_name

    security_rule {
        name          = "AllowHTTP"
        priority       = 1003
        direction      = "Inbound"
        access         = "Allow"
        protocol       = "Tcp"
        source_port_range = "*"
        destination_port_range = "80"
        source_address_prefix = azurerm_subnet.web_subnet.address_prefixes[0]
        destination_address_prefix = "*"
    }

    security_rule {

```



```

        name                = "AllowMySQL"
        priority             = 1004
        direction            = "Outbound"
        access               = "Allow"
        protocol             = "Tcp"
        source_port_range    = "*"
        destination_port_range = "3306"
        source_address_prefix = "*"
        destination_address_prefix = "*"
    }
    security_rule {
        name                = "AllowFlakySQL"
        priority             = 1010
        direction            = "Inbound"
        access               = "Allow"
        protocol             = "Tcp"
        source_port_range    = "*"
        destination_port_range = "5000"
        source_address_prefix = azurerm_subnet.web_subnet.address_prefixes[0]
        destination_address_prefix = "*"
    }
    security_rule {
        name                = "AllowOutboundDB"
        priority             = 102
        direction            = "Outbound"
        access               = "Allow"
        protocol             = "Tcp"
        source_port_range    = "*"
        destination_port_range = "1433"
        source_address_prefix = azurerm_subnet.app_subnet.address_prefixes[0]
        destination_address_prefix = azurerm_subnet.db_subnet.address_prefixes[0]
    }
    depends_on = [ azurerm_resource_group.res_grp ]
}
resource "azurerm_subnet_network_security_group_association" "appnsgconn" {
    subnet_id = azurerm_subnet.app_subnet.id
    network_security_group_id = azurerm_network_security_group.appnsg.id
    depends_on = [ azurerm_network_security_group.appnsg ]
}

resource "azurerm_network_security_group" "dbnsg" {
    name                = "db-nsg"
    location             = var.location
    resource_group_name = var.res_grp_name

    security_rule {
        name                = "AllowSQLRule"
        priority             = 1005
        direction            = "Inbound"
        access               = "Allow"
        protocol             = "Tcp"
        source_port_range    = "*"
        destination_port_range = "3306"
        source_address_prefix = azurerm_subnet.app_subnet.address_prefixes[0]
        destination_address_prefix = "*"
    }
    depends_on = [ azurerm_resource_group.res_grp ]
}

```

```

}
resource "azurerm_subnet_network_security_group_association" "dbnsgconn" {
  subnet_id = azurerm_subnet.db_subnet.id
  network_security_group_id = azurerm_network_security_group.dbnsg.id
  depends_on = [ azurerm_network_security_group.dbnsg ]
}

locals {
  is_production = contains(["Prod", "Production", "prod", "production"], terraform.workspace)
}

resource "azurerm_linux_virtual_machine_scale_set" "web_vmss" {
  name = "web-vmss"
  resource_group_name = var.res_grp_name
  location = var.location
  sku = "Standard_B1s"
  instances = 2
  admin_username = var.admin_username
  upgrade_mode = "Manual"

  zones = local.is_production ? ["1", "2"] : null

  source_image_reference {
    publisher = "Canonical"
    offer = "0001-com-ubuntu-server-jammy"
    sku = "22_04-lts"
    version = "latest"
  }

  admin_ssh_key {
    username = var.admin_username
    public_key = file("${path.module}/${var.web_ssh_key_path}")
  }

  os_disk {
    caching = "ReadWrite"
    storage_account_type = "Standard_LRS"
  }

  network_interface {
    name = "web-vmss-nic"
    primary = true

    ip_configuration {
      name = "web-nic-ip"
      subnet_id = azurerm_subnet.web_subnet.id
      application_gateway_backend_address_pool_ids = [for pool in azurerm_application_gateway.network.backend_address_pool : pool.id]
      load_balancer_backend_address_pool_ids = [azurerm_lb_backend_address_pool.backend_pool.id]
    }
  }

  tags = {
    environment = terraform.workspace
  }
}

```

```

custom_data = filebase64("${path.module}/Scripts/web_init.sh")

depends_on = [
  azurerm_application_gateway.network,
  azurerm_resource_group.res_grp,
  azurerm_linux_virtual_machine_scale_set.app_vmss
]
}

resource "azurerm_monitor_autoscale_setting" "vmss_autoscale" {
  name                = "web-vmss-autoscale"
  location            = var.location
  resource_group_name = var.res_grp_name
  target_resource_id  = azurerm_linux_virtual_machine_scale_set.web_vmss.id

  profile {
    name = "defaultProfile"
    capacity {
      minimum = "2"
      maximum = "5"
      default = "2"
    }

    rule {
      metric_trigger {
        metric_name      = "Percentage CPU"
        metric_resource_id = azurerm_linux_virtual_machine_scale_set.web_vmss.id
        time_grain       = "PT1M"
        statistic         = "Average"
        time_window       = "PT5M"
        time_aggregation = "Average"
        operator          = "GreaterThan"
        threshold         = 75
      }

      scale_action {
        direction = "Increase"
        type      = "ChangeCount"
        value     = "1"
        cooldown  = "PT5M"
      }
    }
  }

  rule {
    metric_trigger {
      metric_name      = "Percentage CPU"
      metric_resource_id = azurerm_linux_virtual_machine_scale_set.web_vmss.id
      time_grain       = "PT1M"
      statistic         = "Average"
      time_window       = "PT5M"
      time_aggregation = "Average"
      operator          = "LessThan"
      threshold         = 25
    }

    scale_action {
      direction = "Decrease"
      type      = "ChangeCount"
    }
  }
}

```

```

        value      = "1"
        cooldown   = "PT5M"
    }
}
}

tags = {
    environment = terraform.workspace
}

depends_on = [azurerm_linux_virtual_machine_scale_set.web_vmss]
}
resource "azurerm_linux_virtual_machine_scale_set" "app_vmss" {
    name                = "app-vmss"
    resource_group_name = var.res_grp_name
    location            = var.location
    sku                 = "Standard_B1s"
    instances           = 2
    admin_username      = var.admin_username
    upgrade_mode        = "Manual"

    zones = local.is_production ? ["1", "2"] : null

    admin_ssh_key {
        username   = var.admin_username
        public_key = file("${path.module}/${var.app_ssh_key_path}")
    }

    source_image_reference {
        publisher = "Canonical"
        offer     = "0001-com-ubuntu-server-jammy"
        sku       = "22_04-lts"
        version   = "latest"
    }

    os_disk {
        caching              = "ReadWrite"
        storage_account_type = "Standard_LRS"
    }

    network_interface {
        name      = "app-vmss-nic"
        primary   = true

        ip_configuration {
            name                          = "app-nic-ip"
            subnet_id                    = azurerm_subnet.app_subnet.id
            load_balancer_backend_address_pool_ids = [azurerm_lb_backend_address_pool.backend_pool.id]
        }
    }
}

custom_data = base64encode(templatefile("${path.module}/Scripts/node_server.sh", {
    db_host = azurerm_mysql_flexible_server.sql_server.fqdn,
    db_user = "${azurerm_mysql_flexible_server.sql_server.administrator_login}@${azurerm_mysql_flexible_server.sql_server.name}",
    db_pswd = azurerm_key_vault_secret.db_pass.value
}))

```

```

tags = {
  environment = terraform.workspace
}
depends_on = [azurerm_resource_group.res_grp, azurerm_mysql_flexible_database.sql_db]
}

resource "azurerm_monitor_autoscale_setting" "app_vmss_autoscale" {
  name                = "app-vmss-autoscale"
  location            = var.location
  resource_group_name = var.res_grp_name
  target_resource_id  = azurerm_linux_virtual_machine_scale_set.app_vmss.id

  profile {
    name = "defaultProfile"
    capacity {
      minimum = "2"
      maximum = "5"
      default = "2"
    }
  }

  rule {
    metric_trigger {
      metric_name      = "Percentage CPU"
      metric_resource_id = azurerm_linux_virtual_machine_scale_set.app_vmss.id
      time_grain       = "PT1M"
      statistic        = "Average"
      time_window      = "PT5M"
      time_aggregation = "Average"
      operator          = "GreaterThan"
      threshold        = 75
    }

    scale_action {
      direction = "Increase"
      type      = "ChangeCount"
      value     = "1"
      cooldown  = "PT5M"
    }
  }

  rule {
    metric_trigger {
      metric_name      = "Percentage CPU"
      metric_resource_id = azurerm_linux_virtual_machine_scale_set.app_vmss.id
      time_grain       = "PT1M"
      statistic        = "Average"
      time_window      = "PT5M"
      time_aggregation = "Average"
      operator          = "LessThan"
      threshold        = 25
    }

    scale_action {
      direction = "Decrease"
      type      = "ChangeCount"
      value     = "1"
      cooldown  = "PT5M"
    }
  }
}

```

```

    }
  }
}

tags = {
  environment = terraform.workspace
}

depends_on = [azurerm_linux_virtual_machine_scale_set.app_vmss]
}

resource "azurerm_public_ip" "pubip" {
  name                = "appgw-public-ip"
  location            = var.location
  resource_group_name = var.res_grp_name
  allocation_method   = "Static"
  sku                 = "Standard"
  depends_on = [ azurerm_resource_group.res_grp ]
}

locals {
  backend_address_pool_name      = "${azurerm_virtual_network.myvnet.name}-beap"
  frontend_port_name            = "${azurerm_virtual_network.myvnet.name}-feport"
  frontend_ip_configuration_name = "${azurerm_virtual_network.myvnet.name}-feip"
  http_setting_name             = "${azurerm_virtual_network.myvnet.name}-be-htst"
  listener_name                 = "${azurerm_virtual_network.myvnet.name}-httplstn"
  request_routing_rule_name     = "${azurerm_virtual_network.myvnet.name}-rqrt"
}

resource "azurerm_application_gateway" "network" {
  name                = "bcpp3-appgateway"
  resource_group_name = var.res_grp_name
  location            = var.location

  sku {
    name     = "Standard_v2"
    tier      = "Standard_v2"
    capacity = 1
  }

  gateway_ip_configuration {
    name        = "my-gateway-ip-configuration"
    subnet_id   = azurerm_subnet.appgateway_subnet.id
  }

  frontend_port {
    name = local.frontend_port_name
    port = 80
  }

  frontend_ip_configuration {
    name                = local.frontend_ip_configuration_name
    public_ip_address_id = azurerm_public_ip.pubip.id
  }

  backend_address_pool {
    name = local.backend_address_pool_name
  }
}

```

```

probe {
  name = "appgw-health-probe"
  protocol = "Http"
  path = "/"
  timeout = 30
  interval = 30
  unhealthy_threshold = 3
  pick_host_name_from_backend_http_settings = false
  host = "172.20.2.100"
  minimum_servers = 0
  match {
    status_code = ["200","201"]
    body = "*"
  }
}

backend_http_settings {
  name = local.http_setting_name
  cookie_based_affinity = "Disabled"
  port = 80
  protocol = "Http"
  request_timeout = 30
  probe_name = "appgw-health-probe"
}

http_listener {
  name = local.listener_name
  frontend_ip_configuration_name = local.frontend_ip_configuration_name
  frontend_port_name = local.frontend_port_name
  protocol = "Http"
}

request_routing_rule {
  name = local.request_routing_rule_name
  rule_type = "Basic"
  priority = 25
  http_listener_name = local.listener_name
  backend_address_pool_name = local.backend_address_pool_name
  backend_http_settings_name = local.http_setting_name
}
depends_on = [ azurerm_resource_group.res_grp ]
}

# Internal Load Balancer
resource "azurerm_lb" "internal_lb" {
  name = "internal-loadbalancer"
  location = var.location
  resource_group_name = var.res_grp_name
  sku = "Standard"

  frontend_ip_configuration {
    name = "internal-lb-fe"
    subnet_id = azurerm_subnet.app_subnet.id
    private_ip_address_allocation = "Static"
    private_ip_address = "170.20.2.100" # use any available IP in the app subnet
  }
}

```

```

}

# Backend Address Pool
resource "azurerm_lb_backend_address_pool" "backend_pool" {
  loadbalancer_id = azurerm_lb.internal_lb.id
  name            = "backend-pool"
}

# Health Probe
resource "azurerm_lb_probe" "health_probe" {
  loadbalancer_id = azurerm_lb.internal_lb.id
  name            = "http-probe"
  protocol        = "Http"
  port            = 80
  request_path    = "/"
  interval_in_seconds = 15
  number_of_probes = 2
}

# Load Balancer Rule
resource "azurerm_lb_rule" "http_rule" {
  name                        = "http-rule"
  loadbalancer_id            = azurerm_lb.internal_lb.id
  protocol                   = "Tcp"
  frontend_port              = 80
  backend_port               = 80
  frontend_ip_configuration_name = "internal-lb-fe"
  backend_address_pool_ids    = [azurerm_lb_backend_address_pool.backend_pool.id]
  probe_id                   = azurerm_lb_probe.health_probe.id
}

resource "azurerm_mysql_flexible_server" "sql_server" {
  name                        = "mysql-${var.res_grp_name}-ncpl-fs"
  resource_group_name        = var.res_grp_name
  location                   = var.location
  administrator_login        = var.admin_username
  administrator_password     = azurerm_key_vault_secret.db_pass.value
  backup_retention_days      = 7
  delegated_subnet_id        = azurerm_subnet.db_subnet.id
  private_dns_zone_id        = null
  sku_name                   = "B_Standard_B1ms"

  depends_on = [ azurerm_resource_group.res_grp ]
}

resource "azurerm_mysql_flexible_database" "sql_db" {
  name                        = "userdetails"
  resource_group_name        = var.res_grp_name
  server_name                 = azurerm_mysql_flexible_server.sql_server.name
  charset                     = "utf8"
  collation                   = "utf8_unicode_ci"

  depends_on = [ azurerm_mysql_flexible_server.sql_server ]
}

resource "azurerm_key_vault" "kv" {
  name                        = "${var.res_grp_name}-kv"
  location                   = var.location
  resource_group_name        = var.res_grp_name

```



```

tenant_id          = data.azurearm_client_config.current.tenant_id
sku_name           = "standard"
soft_delete_retention_days = 7
purge_protection_enabled = false

access_policy {
  tenant_id = data.azurearm_client_config.current.tenant_id
  object_id = data.azurearm_client_config.current.object_id

  key_permissions = [
    "Get", "List", "Create", "Update", "Delete", "Purge",
  ]

  secret_permissions = [
    "Get", "Set", "List", "Backup", "Delete", "Restore", "Purge",
  ]
}

depends_on = [ azurearm_resource_group.res_grp ,data.azurearm_client_config.current]
}

resource "azurearm_key_vault_secret" "db_pass" {
  name      = "mysql-password"
  value     = var.db_password
  key_vault_id = azurearm_key_vault.kv.id
  depends_on = [ azurearm_resource_group.res_grp, azurearm_key_vault.kv ]
}

output "public_ip_address" {
  value = azurearm_public_ip.pubip.ip_address
}
output "resource_grp_name" {
  value = azurearm_resource_group.res_grp.name
}
output "sql-fqdn" {
  value = azurearm_mysql_flexible_server.sql_server.fqdn
}
output "sub_id" {
  value = data.azurearm_client_config.current.subscription_id
}
output "clientid" {
  value = data.azurearm_client_config.current.client_id
}
output "objectid" {
  value = data.azurearm_client_config.current.object_id
}
output "tenantid" {
  value = data.azurearm_client_config.current.tenant_id
}

```

Variables

```

variable "res_grp_name" {}
variable "location" {}
variable "db_password" {}
variable "environment" {}
variable "admin_username" {}

```

```

variable "web_ssh_key_path" {}
variable "app_ssh_key_path" {}
variable "vnet_name" {}
variable "vnet_web" {}
variable "vnet_app" {}
variable "vnet_db" {}
variable "vnet_appgw" {}

```

TFVARS file

1. Dev Environment

```

db_password      = "Mohan20ram1989*"
res_grp_name     = "bcpp3-dev"
location        = "Canada Central"
environment      = "dev"
admin_username   = "adminuser"
web_ssh_key_path = "VM_Key/web-ssh-key.pub"
app_ssh_key_path = "VM_Key/app-ssh-key.pub"
vnet_name        = "myvnet-dev"
vnet_web         = "web-subnet-dev"
vnet_app         = "app-subnet-dev"
vnet_db          = "db-subnet-dev"
vnet_appgw       = "appgw-subnet-dev"

```

2. Staging Environment

```

db_password      = "Mohan20ram1989*"
res_grp_name     = "bcpp3-stage"
location        = "Canada Central"
environment      = "stage"
admin_username   = "adminuser"
web_ssh_key_path = "VM_Key/web-ssh-key.pub"
app_ssh_key_path = "VM_Key/app-ssh-key.pub"
vnet_name        = "myvnet-stage"
vnet_web         = "web-subnet-stage"
vnet_app         = "app-subnet-stage"
vnet_db          = "db-subnet-stage"
vnet_appgw       = "appgw-subnet-stage"

```

3. Prod Environment

```

db_password      = "Mohan20ram1989*"
res_grp_name     = "bcpp3-prod"
location        = "Central US"
environment      = "prod"
admin_username   = "adminuser"
web_ssh_key_path = "VM_Key/web-ssh-key.pub"
app_ssh_key_path = "VM_Key/app-ssh-key.pub"
vnet_name        = "myvnet-prod"
vnet_web         = "web-subnet-prod"
vnet_app         = "app-subnet-prod"
vnet_db          = "db-subnet-prod"
vnet_appgw       = "appgw-subnet-prod"

```